Εθνικό Μετσόβιο Πολυτεχνείο
Σχολή Ηλεκτρολόγων Μηχανικών
και Μηχανικών Υπολογιστών
Τομέας Επικοινωνιών, Ηλεκτρονικής & Συστημάτων
Πληροφορικής

Διαδραστική Πρόσβαση Σε Πολιτιστικές Βάσεις

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

Αλέξανδρος Π. Σωτήραλης

Επιβλέπων : Γεώργιος Στασινόπουλος
Καθηγητής Ε.Μ.Π.

Αθήνα, Νοέμβριος 2009

Εθνικό Μετσόβιο Πολυτεχνείο
Σχολή Ηλεκτρολόγων Μηχανικών
και Μηχανικών Υπολογιστών
Τομέας Επικοινωνιών, Ηλεκτρονικής & Συστημάτων
Πληροφορικής

Διαδραστική Πρόσβαση Σε Πολιτιστικές Βάσεις

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

Αλέξανδρος Π. Σωτήραλης

Επιβλέπων : Γεώργιος Στασινόπουλος
Καθηγητής Ε.Μ.Π.

Εγκρίθηκε από την τριμελή εξεταστική επιτροπή την 25η Νοέμβρη 2009

.....	.....	.....
Γεώργιος Στασινόπουλος	Ευστάθιος Συκάς	Μιχαήλ Θεολόγου
Καθηγητής Ε.Μ.Π.	Καθηγητής Ε.Μ.Π.	Καθηγητής Ε.Μ.Π.

Αθήνα, Νοέμβριος 2009

.....
Αλέξανδρος Π. Σωτήραλης

Διπλωματούχος Ηλεκτρολόγος Μηχανικός και Μηχανικός Υπολογιστών Ε.Μ.Π.

Περίληψη

Στην παρούσα διπλωματική εργασία υλοποιήθηκε μια διαδικτυακή εφαρμογή πρόσβασης (για προβολή, προσθήκη ή τροποποίηση) σε πολιτιστικά δεδομένα.

Τα δεδομένα αυτά είναι σφραγίδες της οθωμανικής αυτοκρατίας. Η βάση στην οποία είναι αποθηκευμένα τα στοιχεία για τις σφραγίδες περιλαμβάνει περισσότερες από 6800 σφραγίδες που προέρχονται από περισσότερα από 1800 διαφορετικά μέρη της Οθωμανικής αυτοκρατρίας στα Βαλκάνια, στη Εγγύς και τη Μέση Ανατολή. Η περιοχή αυτή περιλαμβάνει διαφορετικά εθνικά, γλωσσικά και πολιτιστικά τοπονύμια, τα οποία μάλιστα άλλαζαν μέσα στα χρόνια με τη δημιουργία νέων εθνών-κρατών. Οι γλώσσες που χρησιμοποιούνται για τις σφραγίδες προέρχονται από δύο διαφορετικά αλφάβητα, το οθωμανικό/τουρκικό(αραβικά) και το γαλλικό(λατινικό). Τα πολυπολιτισμικά αυτά χαρακτηριστικά υπαγορεύουν και μια πολυγλωσσική προσέγγιση για την εφαρμογή που αναπτύχθηκε. Η χρησιμότητα των σφραγίδων μπορεί να έγκειται σε ιστορικούς λόγους ή ως χόμπυ για συλλέκτες.

Η εφαρμογή είναι διαδικτυακή, δηλαδή μπορεί ο κάθε χρήστης να την προσπελάσει μέσα από το διαδίκτυο. Η εφαρμογή τρέχει σε εναν server και ο χρήστης έχει πρόσβαση σε αυτήν μέσω ενός browser, χωρίς να χρειάζεται κάποιο εξειδικευμένο λογισμικό. Η εφαρμογή στον server δέχεται τις αιτήσεις του χρήστη, τις επεξεργάζεται, προσπελαύνει τη βάση δεδομένων και απαντά στο χρήστη με σελίδες html.

Η εφαρμογή είναι διαδραστική και φιλική προς το χρήστη. Μέσα από διάφορα μενού ο χρήστης μπορεί να προβεί εύκολα σε εξειδικευμένες αναζητήσεις και να δει τα αποτελέσματα. Παρέχεται ακόμα δυνατότητα επιλογής ανάμεσα σε λατινικά ή πολυγλωσσικά μενού.

Οι υπηρεσίες που παρέχει η εφαρμογή στο χρήστη είναι η αναζήτηση σφραγίδων και η προβολή πληροφοριών και φωτογραφιών σχετικά με αυτές, καθώς και η δυνατότητα τροποποίησης στοιχείων σφραγίδων και εισαγωγή νέων στη βάση. Λειτουργεί έτσι σαν ηλεκτρονικός κατάλογος-album σφραγίδων.

Λέξεις-κλειδιά :

διαδίκτυο, εφαρμογή, σφραγίδες, διαδραστικότητα, πολιτισμικά, οθωμανική, αυτοκρατορία, πολυγλωσσικά

Abstract

The subject of this diploma thesis is the development of an internet application for accessing cultural data (viewing, adding, modifying).

The cultural data are cancellations of the Ottoman Empire. Over 6800 Ottoman cancellations from more than 1800 sites of the Ottoman Empire in the Balkans, Near and Middle East are included. This wide geographical area involves multinational, multilingual, and multicultural toponyms, frequently changed over the years by the formation of new nation-states. Two alphabets are simultaneously used on the cancellations, Ottoman/Turkish(Arabic) and French(Latin). These multicultural characteristics lead to a multilingual approach to the application developped. Cancellations can be useful for historic reasons or collectors.

The application is an internet application, meaning that any one interested can use it over the internet. It runs on a server and the user can access it using a common internet browser without the need of any separate software. The server receives clients requests, processes them, accesses the database and responds to the client with html pages.

The application is interactive and user friendly. Using various menus, the user can easily specialize searches and view the results. A choice between latinised and multilingual menus is available.

Services offered by the application are searching cancellations and viewing information and photos about them, modifying existing cancellations (modifying data or adding photos), and inserting new cancellations. The application can play the role of an electronic album for cancellations.

Keywords :

internet, application, interactive, cancellation, stamp, cancel, ottoman, empire, cultural, multilingual, multicultural, 3-tier architecture

Ευχαριστίες

Θα ήθελα να ευχαριστήσω τον κ. Στασινόπουλο Γεώργιο, καθηγητή ΕΜΠ, για τις συμβουλές, την καθοδήγηση και την αμέριστη συμπαράστασή του κατά τη διάρκεια της εκπόνησης της παρούσας διπλωματικής εργασίας.

ΠΕΡΙΕΧΟΜΕΝΑ

ΚΕΦΑΛΑΙΟ 1 : εισαγωγή

Στο κεφάλαιο 1 περιγράφεται το αντικείμενο της παρούσας διπλωματικής (παράγραφος 1.1), καθώς και η οργάνωση του κειμένου που κρατάτε στα χέρια σας (παράγραφος 1.2).

1.1 – αντικείμενο διπλωματικής

Αντικείμενο της παρούσας διπλωματικής είναι η ανάπτυξη μιας εφαρμογής για πρόσβαση σε πολιτιστικά δεδομένα. Τα δεδομένα αυτά είναι σφραγίδες της οθωμανικής αυτοκρατορίας. Η πρόσβαση περιλαμβάνει δυνατότητες όπως προβολή, τροποποίηση, προσθήκη, διαγραφή πληροφοριών/εικόνων σχετικών με τις σφραγίδες.

Η εφαρμογή πρέπει να έχει κάποια χαρακτηριστικά :

i. Πρέπει να είναι διαδικτυακή. Σε μια εποχή όπου το διαδίκτυο γνωρίζει τεράστια εξάπλωση και χρησιμοποίηση, η δημοσίευση στο διαδίκτυο κάποιας εφαρμογής την κάνει άμεσα διαθέσιμη σε όλους τους ενδιαφερόμενους.
ii. Πρέπει να είναι διαδραστική. (υπάρχουν διαφωνίες σχετικά με τον ορισμό της έννοιας της διαδραστικότητας). Ένας ορισμός αφορά τις εφαρμογές που δέχονται εισόδους (δεδομένα, εντολές) από χρήστες και απαντούν σε αυτές. Ένας πιο αυστηρός ορισμός απαιτεί κάθε απάντηση της εφαρμογής στις αιτήσεις των χρηστών να σχετίζεται με και να λαμβάνει υπόψη μια σειρά από εισόδους του χρήστη (καθώς και τις σχέσεις ανάμεσα σε αυτές τις εισόδους) κι όχι μόνο την τελευταία είσοδο. Όποιος ορισμός κι αν υιοθετηθεί η παρούσα εφαρμογή αποτελεί μια διαδραστική εφαρμογή.
iii. Πρέπει να είναι φιλική προς το χρήστη. Η φιλικότητα προς το χρήστη είναι μια απαίτηση που αφορά όλες τις εφαρμογές που δεν στοχεύουν σε ένα εξειδικευμένο επιστημονικό κοινό. Αυτή επιτυγχάνεται με χρήση φιλικών διεπαφών χρήστη, οι οποίες διευκολύνουν τη χρήση της εφαρμογής. Στη συγκεκριμένη περίπτωση, αυτή η απαίτηση επετείνεται και λόγω της φύσης των δεδομένων (πολυγλωσσικά, πολυπολιτισμικά).

Η εφαρμογή ανήκει σε μια ευρύτερη κατηγορία, αυτήν των εφαρμογών με αρχιτεκτονική 3-tier. Αυτές οι εφαρμογές σπάνε σε τρία διακριτά λογικά επίπεδα όπου το κάθε επίπεδο αναλαμβάνει ένα ρόλο. Πρώτο επίπεδο είναι το επίπεδο παρουσιάσης, το οποίο αναλαμβάνει την επικοινωνία με τον τελικό χρήστη (άνθρωπος). Αυτό είναι συνήθως ένας web browser. Δεύτερο επίπεδο είναι ένας server που δέχεται αιτήσεις από το πρώτο επίπεδο, τις επεξεργάζεται και απαντάει σ' αυτές. Τρίτο επίπεδο είναι το επίπεδο των δεδομένων της εφαρμογής, όπου ανατρέχει το επίπεδο δύο για να ανακτήσει τα δεδομένα που του χρειάζονται για να ικανοποιήσει τις αιτήσεις του πρώτου επιπέδου.(περισσότερα για την αρχιτεκτονική στο κεφάλαιο 4).

Προφανώς υπάρχουν κι εναλλακτικές επιλογές όσον αφορά την αρχιτεκτονική της εφαρμογής, αλλά αυτή ενδείκνυται.

1.2 – οργάνωση κειμένου

Το κείμενο της διπλωματικής έχει οργανωθεί σε 7 κεφάλαια.

Στο κεφάλαιο 1 γίνεται μια μικρή εισαγωγή στην παρούσα διπλωματική.

Στο κεφάλαιο 2 δίνονται κάποιες πληροφορίες για τα πολιτιστικά δεδομένα της εφαρμογής και τις απαιτήσεις που εγείρει η φύση αυτών (ιστορικά, πολυπολιτισμικά, πολυγλωσσικά,).

Στο κεφάλαιο 3 γίνεται μια επίδειξη λειτουργίας της εφαρμογής μέσω screenshots. Το κεφάλαιο αυτό μπορεί να χρησιμοποιηθεί και σαν εγχειρίδιο χρήσης αφού επιδεικνύει κι επεξηγεί τις επιλογές που έχει ο χρήστης σε κάθε σελίδα της εφαρμογής.

Στο κεφάλαιο 4 περιγράφεται η αρχιτεκτονική της εφαρμογής.

Στο κεφάλαιο 5 περιγράφεται η υλοποίηση της εφαρμογής. Καταγράφονται θέματα όπως τα προγραμματιστικά εργαλεία που χρησιμοποιήθηκαν κατά την ανάπτυξη της εφαρμογής, χρήσιμα χαρακτηριστικά του asp .net που χρησιμοποιήθηκαν και η γνώση των οποίων είναι απαραίτητη για την κατανόηση της εφαρμογής, η βάση δεδομένων της εφαρμογής. Τέλος αφιερώνεται μια παράγραφος σε κάθε κλάση της εφαρμογής όπου εξηγείται ο κώδικας. Ιδιαίτερη έμφαση δίνεται στην επικοινωνία ανάμεσα στις σελίδες και πως αυτή υλοποιείται με χρήση των αντικειμένων Session και ViewState.

Στο κεφάλαιο 6 περιγράφεται εν συντομία η διαδικασία που ακολουθήθηκε για τον έλεγχο της ορθότητας της εφαρμογής.

Στο κεφάλαιο 7 προτείνονται κάποιες πιθανές προεκτάσεις της εφαρμογής (παράγραφος 7.2), και μια διαφορετική προσέγγιση που θα μπορούσε να ακολουθηθεί στην ανάπτυξή της (παράγραφος 7.1).

Ακολουθεί η βιβλιογραφία της διπλωματικής.

Τέλος στο παράρτημα Α υπάρχει ο πλήρης κώδικας της εφαρμογής.

ΚΕΦΑΛΑΙΟ 2 : για τα πολιτιστικά δεδομένα της εφαρμογής

Τα πολιτιστικά δεδομένα της εφαρμογής είναι σφραγίδες της οθωμανικής αυτοκρατορίας, μερικώς ή ολικώς συντηρημένες. Περισσότερες από 6800 σφραγίδες που προέρχονται από περισσότερα από 1800 μέρη της οθωμανικής αυτοκρατορίας στα Βαλκάνια, στην Εγγύ και Μέση ανατολή είναι αποθηκευμένες στη βάση δεδομένων της εφαρμογής. Μπορούν να αξιοποιηθούν για ιστορικούς πολιτιστικούς λόγους ή να αποτελέσουν αντικείμενα συλλογής για φιλοτελιστές.

Οι οθωμανικές σφραγίδες καλύπτουν μια ευρεία γεωγραφική περιοχή που περιλαμβάνει πολυεθνικά, πολυγλωσσικά και πολυπολιτισμικά τοπονύμια (γεωγραφικά ονόματα ταχυδρομείων) που συχνά άλλαξαν όνομα την περίοδο 1863-1922. Αυτήν ήταν η περίοδος σχηματισμού για έναν αριθμό από έθνη-κράτη στην Νοτιοανατολική Ευρώπη, στη Μέση Ανατολή και στη Βόρεια Αφρική, γεγονός που αναδεικνύει περιφερειακά και εθνικά θέματα.

Σημαντικό πολιτιστικό χαρακτηριστικό των σφραγίδων είναι η χρήση δύο αλφαβήτων, οθωμανικό/τούρκικο(αραβικά) και γαλλικό(λατινικά). Αρχικά χρησιμοποιήτο το αραβικό αλφάβητο. Η περίοδος αυτή είναι γνωστή ως η Brandt περίοδος. Στη συνέχεια χρησιμοποιήθηκαν ταυτόχρονα αραβικά και λατινικά, ειδικά σε σφαιρικές σφραγίδες, με τα αραβικά να εμφανίζονται στο πάνω μέρος και τα λατινικά στο κάτω μέρος των σφραγίδων. Επίσης έχουν

χρησιμοποιηθεί διαφορετικοί καλλιγραφικοί τύποι, κυρίως rika και αργότερα

nasx. Rika είναι ένα είδος αραβικής στενογραφίας που επιτρέπει γρήγορη αλλά και καλαίσθητη γραφή. Είναι τουρκικής προέλευσης και είχε χρησιμοποιηθεί ευρέως από την τουρκική διοίκηση, κι επομένως και στις σφραγίδες του οθωμανικού ταχυδρομείου που διανέμονταν σε όλη την αυτοκρατορία.

Ολόκληρες σφραγίδες με καθαρά πεδία είναι σχετικά σπάνιες. Ωστόσο υπάρχουν πολλές δυσανάγνωστες, μερικώς συντηρημένες που είναι εύκολο να βρεθούν. Επειδή τα πεδία τους είναι κατακερματισμένα ή θολά χρειάζεται αρκετή προσπάθεια για να αναγνωριστούν στοιχεία όπως το ταχυδρομείο ή η χρονική περίοδος στην οποία ανήκουν.

Η βάση δεδομένων περιέχει φωτογραφίες από κομμάτια των χτυπημάτων των σφραγίδων σε γραμματόσημα ή φακέλους. Στις περισσότερες περιπτώσεις δεν διασώνεται ολόκληρη η σφραγίδα, αλλά κάποιο μέρος αυτής. Αν λείπει το κάτω μέρος μιας σφραγίδας, μόνο το αραβικό κείμενο διασώνεται (το οποιό είναι μερικές φορές δυσανάγνωστο). Αν το αριστερό (δεξί) μέρος λείπει, η αρχή(τέλος) του γαλλικού κειμένου μαζί με το τέλος(αρχή) του αραβικού κειμένου του ταχυδρομείου είναι αναγνώσιμα.

Βλέποντας τα αραβικά κείμενα , παρατηρούμε ονόματα ταχυδρομείων (που συχνά διαφέρουν από τα τοπονύμια των σύγχρονων χαρτών) και συνήθεις γεωγραφικές/διοικητικές/πολιτιστικές εκφράσεις. Τα γεωγραφικά ονόματα

είναι ένα δύσκολο (και ορισμένες φορές πολιτικά φορτισμένο) θέμα. Τα οθωμανικά ταχυδρομεία αποθηκεύονται με τουλάχιστον τέσερα διαφορετικά ονόματα (Ottoman, Turkish, Latinised, Multilingual, Other). Ottoman είναι το επίσημο όνομα που χρησιμοποιούσε η οθωμανική διοίκηση. Turkish είναι το ίδιο όνομα σε σύγχρονα τούρκικα, το οποίο δεν ταυτίζεται απαραίτητα με τη σύγχρονη ονομασία της τούρκικης περιοχής. Latinised είναι το τοπονύμιο που χρησιμοποιείται σήμερα σε κάθε χώρα, ύστερα από διάφορες πιθανές μετονομασίες που οφείλονται σε ιστορικούς, πολιτιστικούς, εθνικούς ή άλλους λόγους (δηλαδή αυτό που θα συναντούσαμε σε ένα σύγχρονο χάρτη). Multilingual είναι το ίδιο όνομα όπως θα το συναντούσαμε σε χάρτες χωρών που χρησιμοποιούν τη δικιά τους γλώσσα. Σε περιπτώσεις που χρειάζεται κάποια επιπλέον ονομασία χρησιμοποιείται το Other (π.χ. σε περιπτώσεις που μια εναλλακτική ονομασία χρησιμοποιείται για ένα χρονικό διάστημα ή σε κάποια περιοχή, παράλληλα με την 'κυρίαρχη' ονομασία).

Ottoman	**Turkish**	**Latinised**	**Multilingual**	**Other**
زاغور	Zağor	Tsepelovo	Τσεπέλοβον	Τσεπέλοβον
اورته کوی	Ortaköy (Edirne V.)	Ivailovgrad	Ивайловград	Орта-кьои
قلعه سلطانیه	Kale-i Sultaniye	Çanakkale	Çanakkale	Ἄβυδος, Δαρδανέλλια
عکا	Akka	Akka	עַכּוֹ, عکا	Πτολεμαΐς
آحی چلبی	Paşmaklı	Smoljan	Смоля	
کوک تپه	Göktepe	Zvecdec	Звеэдец	

πίνακας 2.1

Επιπλέον το ίδιο όνομα μπορεί να εμφανίζεται σε διαφορετικές σφραγίδες με διαφορετική χρήση γραμμάτων, π.χ. λατινικό κείμενο χωρίς/με τη χρήση ειδικών τουρκικών χαρακτήρων που εισήχθησαν μετά το 1293. Οι διάφορες τοποθεσίες μπορεί να είναι μικρές, με ένα μόνο ταχυδρομείο, αλλά τα μεγάλα κέντρα εμφανίζονται στις σφραγίδες με πολλαπλά ονόματα (π.χ. Der Saadet, Der Aliye, Constantinople, Stamboul, İstanbul). Μία μεγάλη πόλη μπορεί να χωρίζεται σε διαφορετικές περιοχές, κάθε μια με τα δικά της ταχυδρομεία, χρησιμοποιώντας δεκάδες διαφορετικών σφραγίδων.

Πολύ συχνά εμφανίζονται κάποιες κοινές εκφράσεις, οι οποίες μπορεί να βρίσκονται σε διάφορα μέρη των σφραγίδων και σε διαφορετικούς συνδυασμούς. Στη βάση δεδομένων της εφαρμογής είναι αποθηκευμένες τέτοιες εκφράσεις μαζί με την ερμηνεία τους στα τουρκικά αλλά και στα αγγλικά για την ευκολότερη κατανόηση και αναζήτηση από τους χρήστες. Μερικά παραδείγματα φαίνονται στον πίνακα 2.2 .

Ottoman	Turkish	Meaning	there-of form
اطه	ada	island	اطه سی
باش	baş	head	باشی
خانه	hane	office	خانه سی
حکومت	hükümet	government	حکومتی
قپو	kapı	gate	قاپوسی

πίνακας 2.2

Για περισσότερες πληροφορίες σχετικά με τις σφραγίδες της οθωμανικής αυτοκρατορίας δείτε το [Sta] και τη βιβλιογραφία του.

ΚΕΦΑΛΑΙΟ 3 : επίδειξη λειτουργίας – εγχειρίδιο χρήσης

Σε αυτό το κεφάλαιο θα γίνει μια μικρή παρουσίαση της λειτουργίας της εφαρμογής μέσα από screenshots, η οποία αναδεικνύει τις διάφορες δυνατότητες που έχει ο χρήστης, κι επομένως μπορεί να χρησιμοποιηθεί και σαν εγχειρίδιο χρήσης.

Η αρχική σελίδα που εμφανίζεται στον browser όταν πληκτρολογούμε τη διεύθυνση του site φαίνεται στην εικόνα 3.1 .

Σε αυτήν παρατηρούμε ότι έχουμε μια σειρά από επιλογές για αναζητήσεις, όπως

- αναζήτηση με βάση κάποιο αραβικό ή λατινικό κείμενο που βρίσκεται πάνω στην σφραγίδα (αριστερά στην παραπάνω εικόνα)
- αναζήτηση με βάση κάποιο vilaet, country ή post office (οι DropDown λίστες στο κέντρο της παραπάνω εικόνας)
- αναζήτηση με βάση ημερομηνίες ή το ID που έχει η σφραγίδα στη βάση δεδομένων (τα κουμπιά στην κορυφή στο κέντρο της φόρμας)
- αναζήτηση για αρνητικές σφραγίδες (τα κουμπιά στο μαύρο πλαίσιο στα δεξιά της φόρμας) όπου μπορούμε να δούμε όλες τις αρνητικές ή να αναζητήσουμε με βάση το σχήμα της σφραγίδας(shape search) ή με βάση κάποια έκφραση που βρίσκεται πάνω στην σφραγίδα(expression search)

Αν χρησιμοποιήσουμε την drop down λίστα για τις χώρες κι επιλέξουμε τη χώρα Greece εμφανίζεται μία λίστα με τις αντίστοιχες σφραγίδες όπως φαίνεται στην εικόνα 3.2 .

Στη λίστα εμφανίζονται κάποιες πληροφορίες για όλες τις σφραγίδες που βρέθηκαν στη βάση, ενώ αν πατήσουμε το κουμπί details εμφανίζονται όλα τα στοιχεία της επιλέγμενης εγγραφής σε μια νέα φόρμα (εικόνα 3.6).

Στο κέντρο της παραπάνω εικόνας υπάρχουν δύο επιλογές latinised/multilingual οι οποίες αλλάζουν τη γλώσσα στην οποία παρουσιάζονται τα στοιχεία vilaet, country, post office.Έτσι αν επιλέξουμε multilingual θα πάρουμε την εικόνα 3.3 .

Οι επιλογές cw και brandt που φαίνονται στα αριστερά του μαύρου πλαισίου μας επιτρέπουν να φιλτράρουμε τα αποτελέσματα μιας αναζήτησης με βάση τα αντίστοιχα πεδία. Το coles walker help κουμπί οδηγεί σε μια νέα φόρμα όπου μέσα από κάποια menu γίνεται πιο εύκολη η επιλογή των τιμών γι αυτά τα πεδία. Αν φιλτράρουμε την παραπάνω αναζήτηση με βάση το cw code θα πάρουμε την εικόνα 3.4 (επιλέξαμε cw=S3 όπως φαίνεται από τη λίστα) .

εικόνα 3.1 : αρχική σελίδα

ID		Normalised Latin	Normalised Arabic	Post Office	CW	Br	
6846	=		بابه بوسنه سی ؛؛	Baya	S4		details
3315	=		غولس	Golos	AV3	XXII	details
3328		VOLOS	غولس	Golos	C2		details
5569	=		ڭلغراف خانه و بوسنه غولس ٣١٠	Golos	S3		details
4278	=		غولس ڭلغراف و بوسنه خانه سی ١٣١٤	Golos	S3	XXVII	details
3330	=		غولس ڭلغراف و بوسنه خانه سی ١٣١٤	Golos	S3	XXVII	details
3325		HALMYROS	ارمبه	Halmiros, Ermiye	C2		details
3314	=		ارمبه	Halmiros, Ermiye	AC2		details
6832	=		ڭلغراف خانه و بوسنه ارمبه ٣١٥	Halmiros, Ermiye	S3		details
6833	=		ڭلغراف و بوسنه خانه ارمبه ١٣٩٠	Halmiros, Ermiye	SM		details
3616	=		اسفاكيه سلطانيه	Isfakya Sultaniye	AO2	XV	details

εικόνα 3.2 : αναζήτηση με βάση country

εικόνα 3.3 : επιλογή multilingual

ID		Normalised Latin	Normalised Arabic	Post Office	CW	Br	
5569	=		ڭلغراف خانه و پوسته غوس ۳۱۰	Golos	S3		details
4278	=		غوس ڭلغراف و پوسته خانه سی ۱۳۱٤	Golos	S3	XXVII	details
3330	=		غوس ڭلغراف و پوسته خانه سی ۱۳۱٤	Golos	S3	XXVII	details
6832	=		ڭلغراف خانه و پوسته ارمبه ۳۱۵	Halmiros, Ermiye	S3		details
6834	=		ڭلغراف خانه و پوسته قردجه ۱۳۱۵	Kardiçe	S3		details
6867	=		ڭلغراف خانه و پوسته قسطل ۱۳۱۱	Kastel kariyesi	S3		details
4276	=		يكشهر ڭلغراف پوسته خانه سی ۱۳۱٤	Yeñişehir (Larisa)	S3		details
3331	=		يكشهر ڭلغراف پوسته خانه سی ۱۳۱٤	Yeñişehir (Larisa)	S3		details

εικόνα 3.4 : φιλτράρισμα αναζήτησης με βάση το cw code

Στα αριστερά της φόρμας (εικόνα 3.4 - πλαίσιο επιλογής με βάση οθωμανικό/λατινικό κείμενο) υπάρχει ένα κουμπί keyboard. Αν το πατήσουμε εμφανίζεται ένα πληκτρολόγιο οθωμανικών χαρακτήρων το οποίο μας επιτρέπει να εισάγουμε κείμενο για την αναζήτηση. Το κείμενο εισάγεται απευθείας στο πλαίσιο κειμένου ottoman που βρίσκεται κάτω από το κουμπί Η προσθήκη αυτή κρίθηκε απαραίτητη για τη διευκόλυνση του χρήστη επειδή πολλές φορές η πληκτρολόγηση οθωμανικών χαρακτήρων είναι μια δύσκολη υπόθεση. Στην εικόνα 3.5 βλέπουμε το εν λόγω πληκτρολόγιο.

Εκτός από τους διάφορους χαρακτήρες το πληκτρολόγιο παρέχει και την επιλογή ανάμεσα από κάποιους κοινούς όρους, η εμφάνιση των οποίων στις σφραγίδες είναι συχνή. Οι οροί αυτοί είναι διαθέσιμοι σε τρεις διαφορετικές γλώσσες (ottoman, turkish, english) για τη διευκόλυνση οποιουδήποτε χρήστη. Η επιλογή γίνεται από τις drop down λίστες που φαίνονται στο κάτω μέρος του πληκτρολογίου.

Τα πλαίσια κειμένου ottoman/french μπορούν να δεχθούν και wildcard χαρακτήρες (όπως % _) για γενικότερη αναζήτηση.

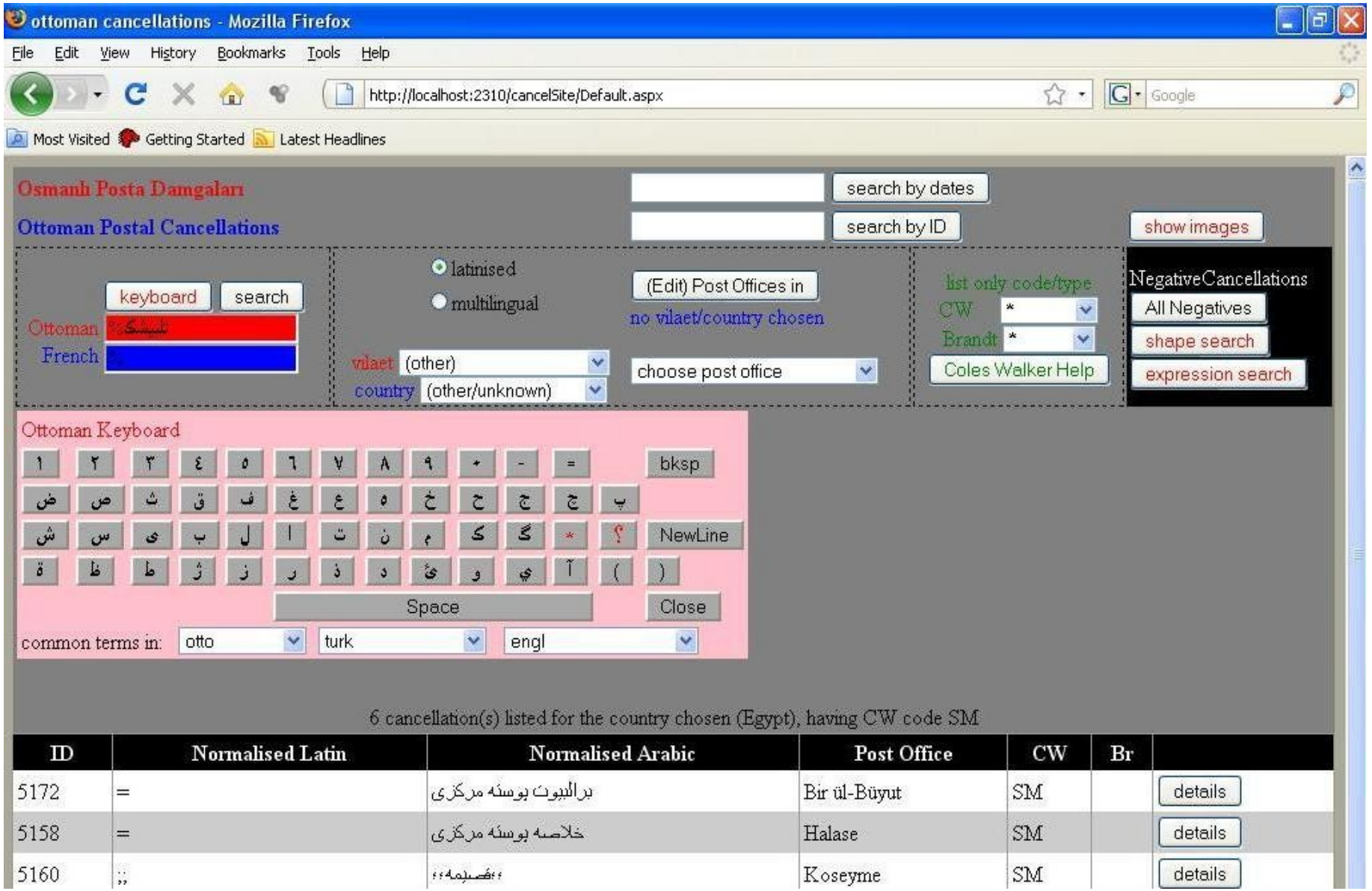
Πατώντας το κουμπί details (από τη λίστα που περιέχει τα αποτελέσματα κάποιας αναζήτησης) βλέπουμε αναλυτικά τις λεπτομέρειες της επιλεγμένης σφραγίδας (εικόνα 3.6). Εμφανίζονται μόνο όσα πεδία είναι γνωστά (π.χ. στην εικόνα 3.6 απουσιάζουν πεδία όπως brandt, nuhoglou, ...).

Κάτω από τα στοιχεία της εγγραφής, εμφανίζεται η λίστα με τα αποτέλεσματα της προηγούμενης αναζήτησης, από όπου μπορούμε να επιλέξουμε να δούμε αναλυτικά κάποια άλλη εγγραφή πατώντας το αντίστοιχο κουμπί details.

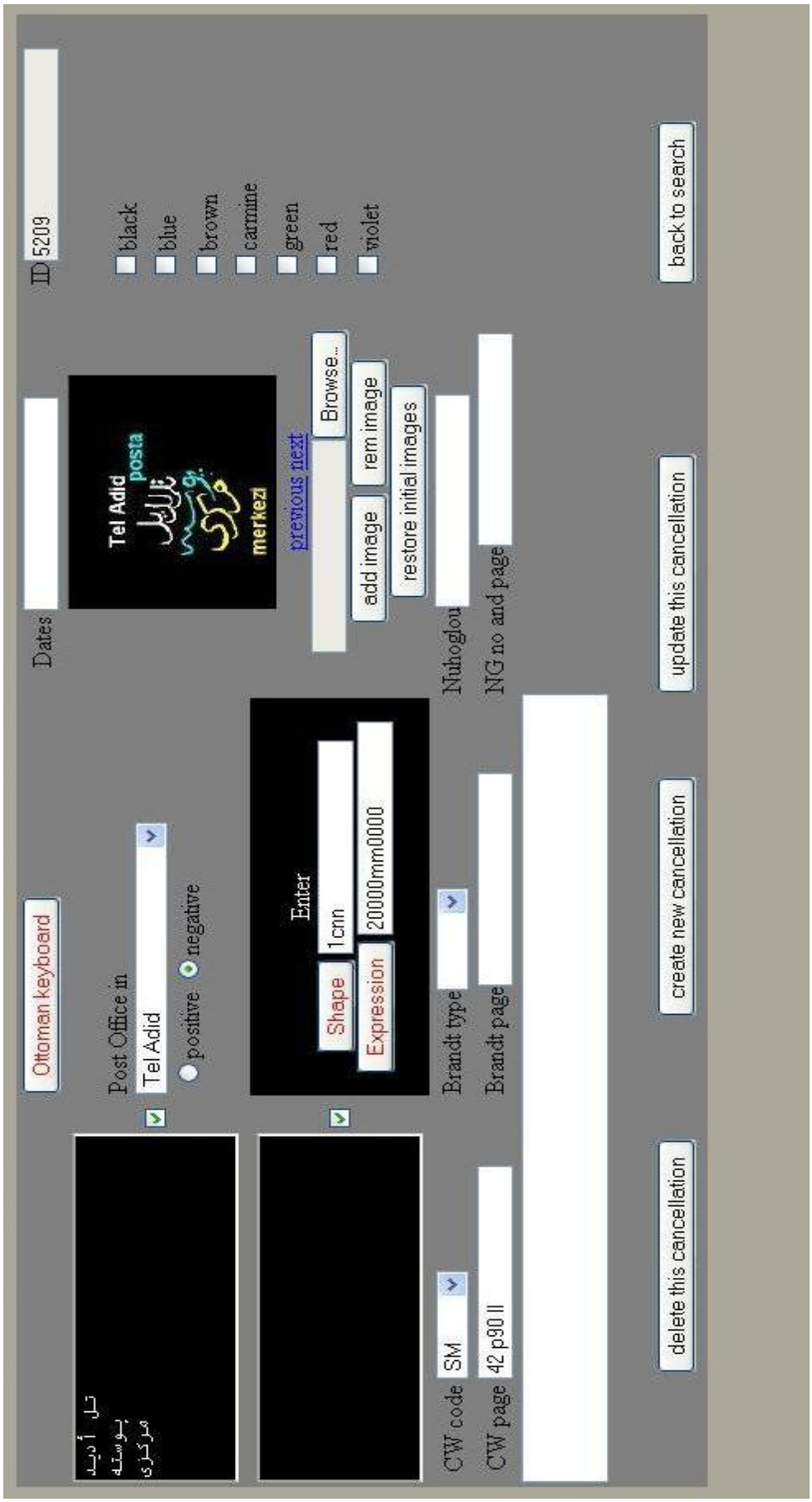
Βλέπουμε επίσης τις εικόνες που υπάρχουν διαθέσιμες γι αυτήν την σφραγίδα (πάνω δεξιά στην οθόνη). Μπορούμε να περιηγηθούμε στις εικόνες (πιθανόν περισσότερες από μια διαθέσιμες) με τους συνδέσμους previous, next που βρίσκονται κάτω από τις εικόνες.

Με το κουμπί back to search επιστρέφουμε στο προηγούμενο menu (αρχική σελίδα με αποτελέσματα τελευταίας αναζήτησης), ενώ με το κουμπί enter new data οδηγούμαστε σε μια καινούρια φόρμα όπου μπορούμε να τροποποιήσουμε την εγγραφή. Η σελίδα αυτή φαίνεται στην εικόνα 3.7 .

εικόνα 3.5 : οθωμανικό ηλεκτρολόγιο



εικόνα 3.6 : λεπτομέρειες επιλεγμένης σφραγίδας



εικόνα 3.7 : φόρμα για τροποποίηση σφραγίδων (update, create, delete)

Σ' αυτή τη σελίδα (εικόνα 3.7) εμφανίζονται όλα τα πεδία μιας εγγραφής. Ο χρήστης έχει δυνατότητα να δημιουργήσει νέες εγγραφές, να ενημερώσει υπάρχουσες, αλλά και να διαγράψει. Οι αλλαγές στέλνονται στη βάση δεδομένων μετά το πάτημα ενός εκ των τριών κουμπιών που βρίσκονται στο κάτω μέρος της φόρμας.

Υπάρχει δυνατότητα προσθήκης νέων εικόνων, αφαίρεση υπαρχουσών εικόνων αλλά και επαναφορά των αρχικών εικόνων που είχε η εγγραφή πριν την επεξεργαστεί ο χρήστης (κουμπιά add/remove/restore image). Σε κάθε περίπτωση η προεπίσκοπηση των εικόνων γίνεται με τον ίδιο τρόπο που γινόταν και στην προηγούμενη φόρμα (σύνδεσμοι previous,next). Οι νέες εικόνες που προσθέτει ο χρήστης ανεβαίνουν και αποθηκεύονται στον server.

Και σε αυτή τη φόρμα υπάρχει το κουμπί ottoman keyboard, που εμφανίζει το οθωμανικό πληκτρολόγιο. Οι οθωμανικοί χαρακτήρες εισάγονται στο πλαίσιο κειμένου που βρίσκεται πάνω αριστερά.

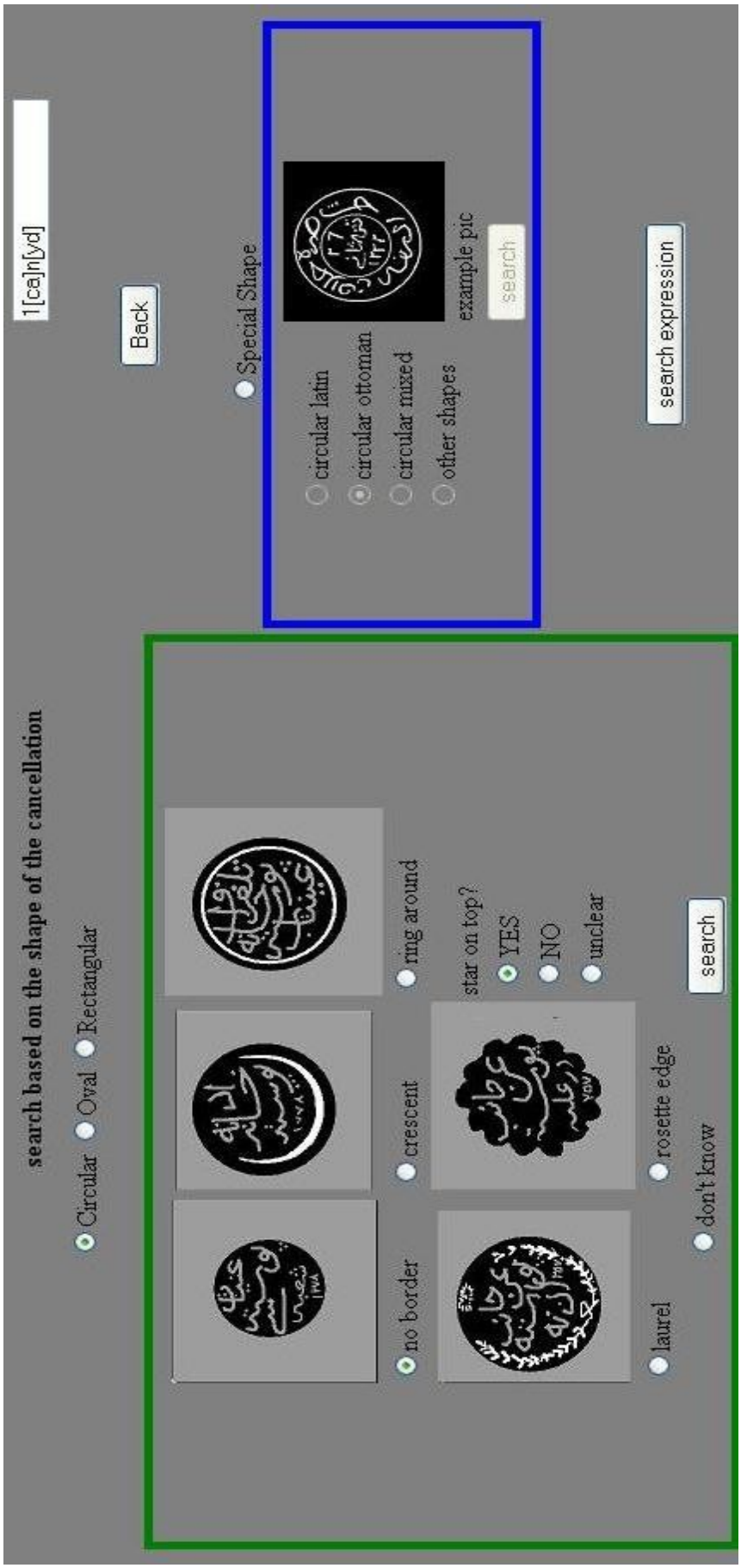
Η επιλογή positive/negative τροποποιεί την εμφάνιση της φόρμας όπως φαίνεται στην εικόνα 3.8(π.χ. positive εγγραφές δεν έχουν shape ή expression, οπότε τα αντίστοιχα κουμπιά γίνονται ανενεργά , κτλ...).

Πριν την αποστολή των νέων δεδομένων στη βάση ελέγχεται η εγκυρότητά τους. Αν δεν είναι έγκυρα τυπώνεται κατάλληλο μήνυμα στον χρήστη. Π.χ. στην εικόνα 3.8 ο χρήστης έχει επιλέξει positive και cwCode=SM, τα οποία είναι ασύμβατα μεταξύ τους, με αποτέλεσμα να παίρνει το μήνυμα που φαίνεται στο κάτω μέρος της φόρμας.

Τέλος τα κουμπιά shape και expression οδηγούν σε νέες φόρμες στις οποίες ο χρήστης διευκολύνεται στην εισαγωγή των αντίστοιχων πεδίων μέσω ειδικών επιλογών. Οι φόρμες αυτές είναι σχεδόν ίδιες με τις φόρμες αναζήτησης shape/expression (που παρουσιάζονται στα αμέσως επόμενα) αφού έχουν τον ίδιο σκοπό, δηλαδή τη διευκόλυνση στην εισαγωγή/αναζήτηση μέσω διεπαφής χρήστη.

Η φόρμα αναζήτησης shape εμφανίζεται αν από την αρχική σελίδα (εικόνα 3.1) πατήσουμε το κουμπί shape search που βρίσκεται στο μαύρο πλαίσιο στα δεξιά της σελίδας. Η φόρμα αυτή φαίνεται στην εικόνα 3.9 .

εικόνα 3.8 : έλεγχος εγκυρότητας εισαγόμενων δεδομένων



εικόνα 3.9 : φόρμα για αναζήτηση με βάση το σχήμα σφραγίδας

Στη φόρμα αυτή (εικόνα 3.9) υπάρχουν δύο πλαίσια (κάθε στιγμή είναι ενεργό το ένα από τα δύο) για επιλογή συνηθισμένου ή ειδικού σχήματος. Πατώντας το κουμπί search του αντίστοιχου πλαισίου, γίνεται αναζήτηση σφραγίδων με βάση τις επιλογές που έχουμε κάνει στο συγκεκριμένο πλαίσιο και επιστρέφουμε στην αρχική σελίδα όπου εμφανίζεται η λίστα με τα αποτελέσματα της αναζήτησης.

Οι επιλογές που μπορούμε να κάνουμε στο πρώτο πλαίσιο έχουν σχέση με το σχήμα της σφραγίδας(circular/oval/rectangular), το περίγραμμα (no border/crescent/ring around/laurel/rosette edge/don't know), καθώς και το αν εμφανίζεται αστέρι στην κορυφή της σφραγίδας ή όχι. Οι επιλογές στο δεύτερο πλαίσιο (ειδικό σχήμα) αφορούν το περίγραμμα της σφραγίδας (κυκλικό ή όχι) και το αν εμφανίζεται σ' αυτήν οθωμανικό/λατινικό κείμενο.

Μετά την επιλογή κάποιου σχήματος, αντί να πατήσουμε το αντίστοιχο κουμπί search, μπορούμε να πατήσουμε το κουμπί search expression, το οποίο μας οδηγεί στη φόρμα αναζήτησης expression. Στην αναζήτηση που θα κάνουμε σ' αυτήν τη φόρμα λαμβάνονται υπόψη τόσο το σχήμα που ήδη έχουμε επιλέξει όσο και η έκφραση που πρόκειται να επιλέξουμε. Η φόρμα search expression φαίνεται στην εικόνα 3.10 .

Σ' αυτήν τη φόρμα μπορούμε να βρεθούμε είτε από τη φόρμα αναζήτησης με βάση το σχήμα (εικόνα 3.9), είτε από την αρχική σελίδα (εικόνα 3.1) πατώντας το κουμπί expression search. Στην πρώτη περίπτωση λαμβάνεται υπόψη για την αναζήτηση και το σχήμα που έχουμε επιλέξει, ενώ στη δεύτερη μόνο οι εκφράσεις που θα επιλέξουμε τώρα.

Με βάση τα radio buttons αυτής της φόρμας επιλέγουμε αν υπάρχουν κάποιες συνηθισμένες εκφράσεις (existent/non existent) και η θέση στην οποία υπάρχουν πάνω στην σφραγίδα (top/middle/bottom/unclear αν είναι ασαφές). Οι εκφράσεις αυτές φαίνονται στις εικόνες. Κάνοντας click πάνω στις εικόνες εμφανίζονται άλλες παραλλαγές της ίδιας έκφρασης.

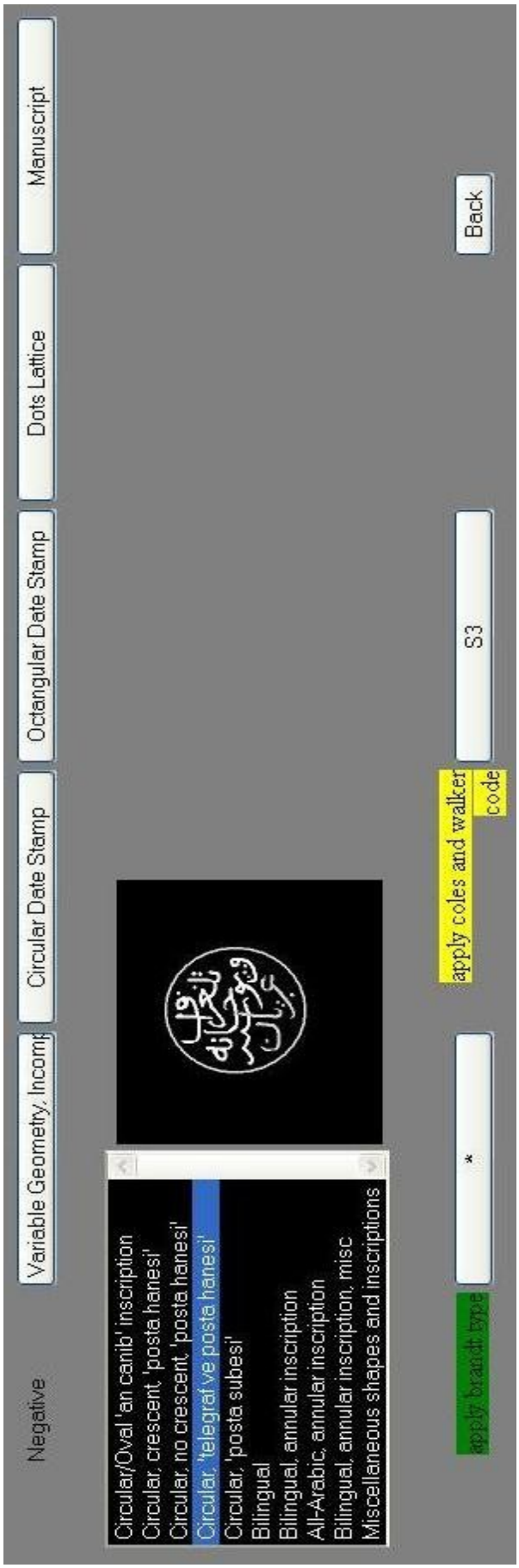
Τέλος μπορούμε να αναζητήσουμε σφραγίδες με βάση την εμφάνιση αριθμών σ' αυτές. Υπάρχει δυνατότητα εισαγωγής τόσο σε arabic όσο και σε hindi script

Πατώντας το κουμπί search επιστρέφουμε στην αρχική σελίδα (εικόνα 3.1), όπου εμφανίζονται τα αποτελέσματα της αναζήτησης με βάση τις επιλογές που έχουμε κάνει. Αν πατήσουμε το κουμπί back to search επιστρέφουμε στην αρχική σελίδα χωρίς να γίνει αναζήτηση (εμφανίζονται τα αποτελέσματα της τελευταίας αναζήτησης που έχει πραγματοποιηθεί).

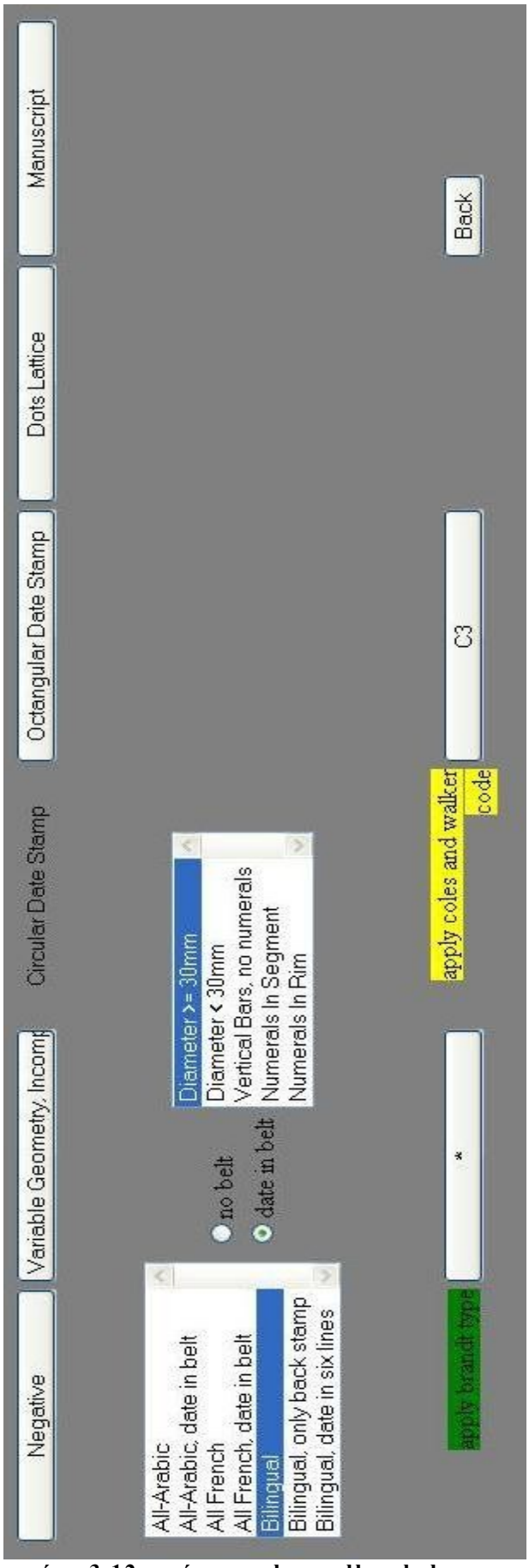
εικόνα 3.10 : αναζήτηση με βάση την εμφάνιση κάποιας έκφρασης

Επιστρέφοντας στην αρχική σελίδα (εικόνα 3.1), βλέπουμε ότι κάτω από τις drop down λίστες υπάρχει ένα κουμπί coles walker help. Η φόρμα που εμφανίζεται πατώντας το, φαίνεται στην εικόνα 3.11 και έχει σκοπό να διευκολύνει το χρήστη στην επιλογή coles walker code για φιλτράρισμα των αποτελεσμάτων μιας αναζήτησης με βάση αυτόν τον κωδικό.

Στην κορυφή της παραπάνω φόρμας υπάρχει μια σειρά από κουμπιά που μας διευκολύνει στο φιλτράρισμα (αν ψάχνουμε για negative/circular/... σφραγίδες). Το πάτημα κάθε κουμπιού εμφανίζει και ένα διαφορετικό menu επιλογών από κάτω. Π.χ. στην εικόνα 3.11 έχουμε επιλέξει αρνητικές σφραγίδες. Μια άλλη επιλογή φαίνεται στην εικόνα 3.12 . Κάθε μενού έχει διάφορες επιπλέον επιλογές για περαιτέρω φιλτράρισμα των αποτελεσμάτων. Η κάθε επιλογή που κάνουμε αντιστοιχεί σε κάποιο coles walker code που εμφανίζεται στο κουμπί δίπλα στο κείμενο apply coles and walker code. Στην εικόνα 3.11 η επιλογή Circular, 'telegraf ve posta hanesi' αντιστοιχεί στον κωδικό S3 που εμφανίζεται στο κουμπί. Πατώντας αυτό το κουμπί γίνεται επιστροφή στην αρχική σελίδα, όπου τα αποτελέσματα της προηγούμενης αναζήτησης έχουν φιλτραριστεί με βάση την επιλογή cw code (S3 στην προκειμένη περίπτωση) και εμφανίζονται στη γνώριμη λίστα. Αν δεν έχει γίνει κάποια επιλογή εμφανίζεται ο wildcard χαρακτήρας * που σημαίνει όλα τα cw codes, δηλαδή επί της ουσίας δε γίνεται κανένα φιλτράρισμα.



εικόνα 3.11 : φόρμα coles walker help



εικόνα 3.12 : φόρμα coles walker help

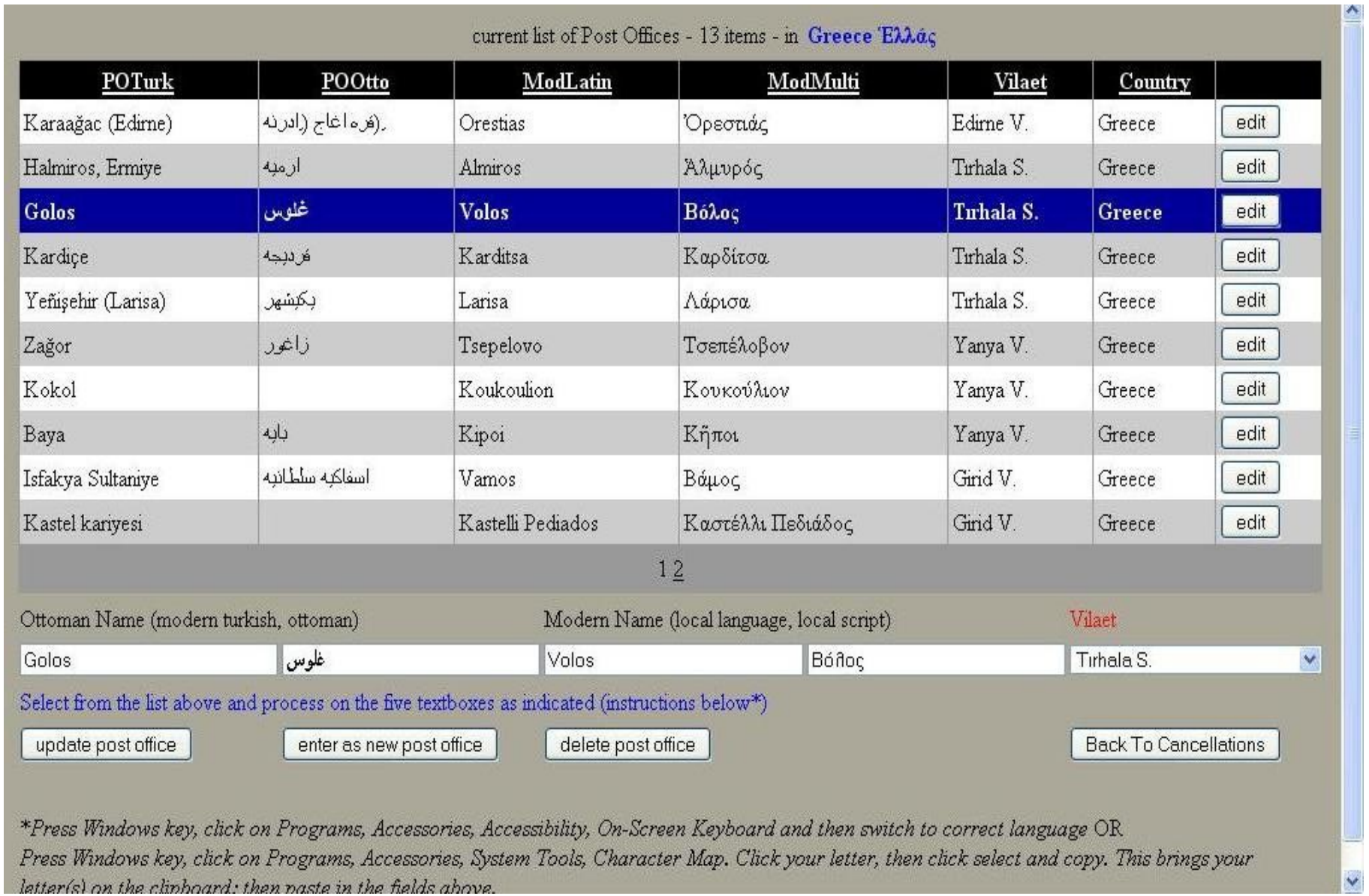
Επιστρέφουμε στην αρχική σελίδα (εικόνα 3.4). Στο κεντρικό πλαίσιο της φόρμας αναζήτησης υπάρχει ένα κουμπί με το κείμενο edit post offices in. Ακριβώς κάτω από αυτό το κουμπί υπάρχει η επιλογή χώρας που έχει γίνει (Greece στη συγκεκριμένη περίπτωση). Αν είχαμε επιλέξει vilaet από την αντίστοιχη drop down λίστα, τότε κάτω από αυτό το κουμπί θα εμφανιζόταν το όνομα του επιλεγμένου vilaet. Πατώντας το συγκεκριμένο κουμπί εμφανίζεται μια νέα σελίδα για τροποποίηση των ταχυδρομείων που βρίσκονται στη συγκεκριμένη χώρα(ή vilaet). Για την επιλογή Greece παίρνουμε τη φόρμα που φαίνεται στην εικόνα 3.13.

Αυτή η φόρμα περιλαμβάνει μια λίστα με όλα τα post offices που βρίσκονται στην επιλεγμένη χώρα. Στη συγκεκριμένη περίπτωση είναι 13, και επειδή δεν χωρούν όλα σε μια σελίδα, υπάρχει επιλογή μετάβασης στη δεύτερη σελίδα της λίστας (αριθμοί-links στο κάτω μέρος της λίστας).

Αν επιλέξουμε ένα ταχυδρομείο από τη λίστα (Golos στην παραπάνω εικόνα) με χρήση του edit κουμπιού που βρίσκεται στην τελευταία στήλη, τότε τα στοιχεία αυτού μεταφέρονται στα πλαίσια κειμένου που βρίσκονται κάτω από τη λίστα. Εκεί έχουμε τη δυνατότητα να τα τροποποιήσουμε, καθώς και να επιλέξουμε vilaet(ή country). Μετά την πιθανή επεξεργασία των στοιχείων, υπάρχει δυνατότητα update, create, delete του post office με χρήση των αντίστοιχων κουμπιών.

Όπως και στην περίπτωση της εισαγωγής νέων δεδομένων για σφραγίδες (εικόνα 3.8), πριν την ενημέρωση της βάσης δεδομένων γίνονται έλεγχοι εγκυρότητας και επιστρέφονται αντίστοιχα μηνύματα (π.χ. δεν επιτρέπεται εισαγωγή post office που έχει ίδιο όνομα με κάποιο υπάρχον post office,...).

εικόνα 3.13 : φόρμα για επεξεργασία των post offices



current list of Post Offices - 13 items - in **Greece Ἑλλάς**

POTurk	POOtto	ModLatin	ModMulti	Vilaet	Country	
Karaağac (Edirne)	قره اغاج (ادرنه).	Orestias	Ὀρεστιάς	Edirne V.	Greece	edit
Halmiros, Ermiye	ارمیه	Almiros	Ἁλμυρός	Tırhala S.	Greece	edit
Golos	**غلوس**	**Volos**	**Βόλος**	**Tırhala S.**	**Greece**	edit
Kardiçe	قردیجه	Karditsa	Καρδίτσα	Tırhala S.	Greece	edit
Yeñişehir (Larisa)	یکیشهر	Larisa	Λάρισα	Tırhala S.	Greece	edit
Zağor	زاغور	Tsepelovo	Τσεπέλοβον	Yanya V.	Greece	edit
Kokol		Koukoulion	Κουκούλιον	Yanya V.	Greece	edit
Baya	بایه	Kipoi	Κῆποι	Yanya V.	Greece	edit
Isfakya Sultaniye	اسفاکیه سلطانیه	Vamos	Βάμος	Girid V.	Greece	edit
Kastel kariyesi		Kastelli Pediados	Καστέλλι Πεδιάδος	Girid V.	Greece	edit

1 2

Ottoman Name (modern turkish, ottoman) | Modern Name (local language, local script) | Vilaet

Golos | غلوس | Volos | Βόλος | Tırhala S.

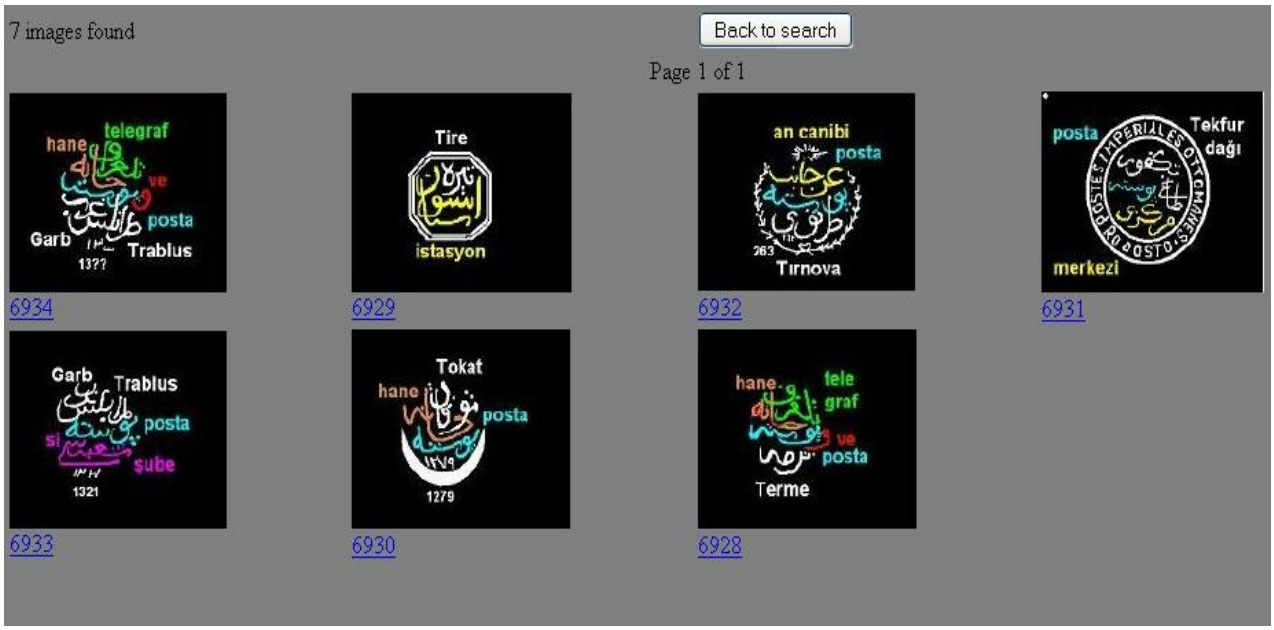
Select from the list above and process on the five textboxes as indicated (instructions below*)

update post office | enter as new post office | delete post office | Back To Cancellations

**Press Windows key, click on Programs, Accessories, Accessibility, On-Screen Keyboard and then switch to correct language* OR
Press Windows key, click on Programs, Accessories, System Tools, Character Map. Click your letter, then click select and copy. This brings your letter(s) on the clipboard; then paste in the fields above.

Επιστρέφουμε και πάλι στην αρχική σελίδα (εικόνα 3.1), όπου στην πάνω δεξιά γωνία υπάρχει ένα κουμπί show images. Η λειτουργία αυτού του κουμπιού είναι (μετά από μια αναζήτηση που έχει επιστρέψει κάποιες εγγραφές στη λίστα της αρχικής σελίδας) να εμφανίζει όσες εικόνες ανήκουν σ' αυτές τις εγγραφές και είναι διαθέσιμες. Η νέα σελίδα που επιστρέφει φαίνεται στην εικόνα 3.14 .

Στην εικόνα 3.14 φαίνονται οι εικόνες που βρέθηκαν στη βάση δεδομένων. Κάτω από κάθε εικόνα υπάρχει ένας αριθμός-σύνδεσμος. Αυτός ο αριθμός είναι το ID της αντίστοιχης εικόνας. Πατώντας τον, οδηγεί στη φόρμα που δείχνει τις λεπτομέρειες της συγκεκριμένης εγγραφής (εικόνα 3.6).

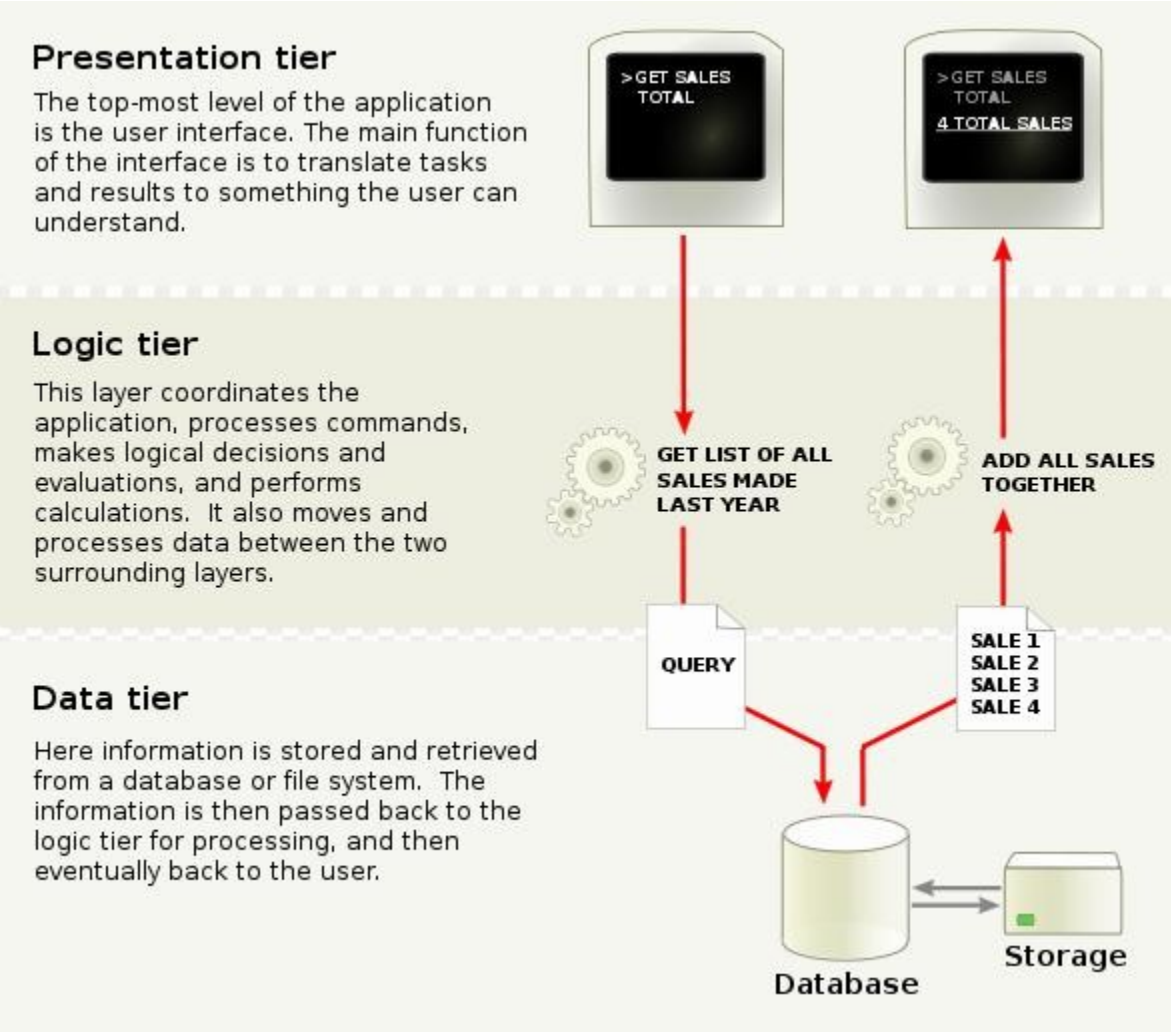


εικόνα 3.14 : προβολή εικόνων από επιλεγμένες σφραγίδες

Τέλος σημειώνεται ότι η περιήγηση στις διάφορες σελίδες της εφαρμογής πρέπει να γίνεται με χρήση των κουμπιών της εφαρμογής (π.χ. κουμπιά back) κι όχι των κουμπιών του browser.

ΚΕΦΑΛΑΙΟ 4 : αρχιτεκτονική εφαρμογής

Η αρχιτεκτονική της εφαρμογής κατατάσσεται στις αρχιτεκτονικές 3-tier (3 επιπέδων, σειρών). Μια οπτική αναπαράσταση αυτής της αρχιτεκτονικής φαίνεται στην εικόνα 4.1 .



εικόνα 4.1 : αρχιτεκτονική 3-tier

Οι εφαρμογές συνήθως "σπάνε" σε μικρότερα λογικά μέρη, όπου κάθε μέρος αναλαμβάνει κάποιο διακριτό ρόλο. Τα μέρη αυτά λέγονται tiers. Οι λόγοι που υιοθετείται μια τέτοια αρχιτεκτονική είναι η εξής :

- Ευκολότερη διαδικασία ανάπτυξης, αφού η υλοποίηση κάθε μέρους είναι ανεξάρτητη από την υλοποίηση των άλλων μερών. Αυτό που μας ενδιαφέρει είναι οι συμβάσεις επικοινωνίας ανάμεσά τους. Έτσι κάθε μέρος μπορεί να αναπτυχθεί από διαφορετικούς προγραμματιστές.
- Ευκολότερος έλεγχος της ορθής λειτουργίας των επιμέρους tiers.
- Εύκολότερη διόρθωση πιθανών σφαλμάτων που θα αναγνωριστούν αφού χρειάζεται να διορθωθεί μόνο ένα μέρος.

- Δυνατότητα αντικατάστασης ενός μέρους από ένα άλλο το οποίο εσωτερικά έχει υλοποιηθεί διαφορετικά (αρκεί να συμμορφώνεται με τις συμβάσεις επικοινωνίας). Ακόμα δυνατότητα χρησιμοποίησης έτοιμων μερών που έχουν αναπτυχθεί σε άλλα projects.
- Ανάλογα με τον τύπο της εφαρμογής, βελτίωση της απόδοσης.
- Ανάγκη για εφαρμογές που εξυπηρετούν χρήστες που βρίσκονται σε διαφορετικά γεωγραφικά μέρη.

Παραδοσιακά οι εφαρμογές αποτελούνταν μόνο από ένα tier που βρίσκεται στον υπολογιστή του χρήστη, αλλά οι διαδικτυακές εφαρμογές τείνουν από τη φύση τους να υιοθετούν μια αρχιτεκτονική πολλών tiers. Πολλές εναλλακτικές αρχιτεκτονικές είνια πιθανές (n-tier). Μία συχνή για τις διαδικτυακές εφαρμογές είναι η 3-tier. Τα τρία αυτά επίπεδα είναι :

i. **presentation tier** (επίπεδο παρουσίασης). Το επίπεδο αυτό προβάλει πληροφορίες στο τελικό χρήστη της εφαρμογής. Συνήθως είναι ένας web browser. Αποτελείται από κάποια διεπαφή χρήστη (φόρμες) μέσα από τις οποίες ο χρήστης ζητάει και βλέπει πληροφορίες. Επικοινωνεί με το logic tier.
ii. **logic tier** (επίπεδο λογικής, επίσης business logic). Συντονίζει την εφαρμογή, επεξεργάζεται εντολές, κάνει λογικές αποφάσεις και αποτιμήσεις, πραγματοποιεί υπολογισμούς. Επικοινωνεί και ενώνει τα άλλα δύο επίπεδα, μεταφέροντας δεδομένα από και προς αυτά.
iii. **data tier** (επίπεδο δεδομένων). Εδώ αποθηκεύονται και ανακτώνται δεδομένα. Συνήθως είναι μια βάση δεδομένων ή ένα σύστημα αρχείων. Επικοινωνεί με το logic tier. Του περνάει δεδομένα για επεξεργασία τα οποία στη συνέχεια προωθούνται στο presentation tier.

Τα επίπεδα επικοινωνούν μόνο με τα αμέσως γειτονικά τους, δηλαδή δεν επιτρέπεται η άμεση επικοινωνία μεταξύ των επιπέδων presentation tier και data tier. Η επικοινωνία ανάμεσα σ' αυτά τα επίπεδα γίνεται μέσω του logic tier. Η επικοινωνία ανάμεσα σε δύο γειτονικά επίπεδα γίνεται με το μοντέλο πελάτη/εξηπυρετητή(client/server). Συγκεκριμένα το presentation λειτουργεί σαν client του logic tier κάνοντας αιτήσεις σε αυτό. Το logic tier χρειάζεται δεδομένα για να απαντήσει σ' αυτές τις αιτήσεις, και γίνεται με τη σειρά του client στο επίπεδο data tier.

Τα τρία λογικά επίπεδα δεν είναι απαραίτητα και τρία διαφορετικά φυσικά επίπεδα. Δηλαδή κάποια επίπεδα μπορεί να βρίσκονται στο ίδιο υπολογιστικό σύστημα. Αυτό που μας ενδιαφέρει είναι ο λογικός τους διαχωρισμός.

Στην εφαρμογή που αναπτύχθηκε **presentation tier** μπορεί να είναι οποιοσδήποτε web browser. Μέσα από αυτόν ο τελικός χρήστης βλέπει τις html σελίδες της εφαρμογής απ' όπου μπορεί να κάνει αιτήσεις για δεδομένα και να τα δει να προβάλονται στην οθόνη του. Εδώ βλέπουμε ένα πλεονέκτημα της αρχιτεκτονικής, δεν χρειάζεται να αναπτυχθεί αυτό το tier καθώς υπάρχουν ήδη πληθώρα έτοιμων λύσεων που μπορούν να επικοινωνήσουν με το logic tier μέσω του πρωτοκόλλου http.

Logic tier είναι οι aspx σελίδες που υλοποιήθηκαν και οι οποίες υλοποιούν όλη την επεξεργαστική λογική, καθώς και την επικοινωνία ανάμεσα στα άλλα δύο επίπεδα. Υπάρχει μια σειρά από περιβάλλοντα και προγραμματιστικές γλώσσες που μπορούν να υλοποιήσουν αυτό το μέρος όπως ASP, ASP .NET, CGI, ColdFusion, JSP/Java, PHP, Perl, Python, Ruby on Rails, Struts.

Τέλος το **data tier** αποτελείται από μια βάση δεδομένων υλοποιημένη σε microsoft access και ένα σύστημα αρχείων για την αποθήκευση και ανάκτηση των εικόνων της εφαρμογής(που έχει τοποθετηθεί στον server). Η επικοινωνία ανάμεσα στο logic tier και το τη βάση δεδομένων γίνεται με χρήση ενός data provider.

ΚΕΦΑΛΑΙΟ 5 : υλοποίηση

Στο κεφάλαιο αυτό καταγράφονται θέματα που άπτονται της υλοποίησης της εφαρμογής που αναπτύχθηκε, αλλά και της διαδικασίας εγκατάστασης για οποιονδήποτε θέλει να τη χρησιμοποιήσει. Υπάρχουν θέματα σχετικά με τη βάση δεδομένων της εφαρμογής, τον κώδικα, το περιβάλλον ανάπτυξης και τη δυνατότητα χρήσης της εφαρμογής.

5.1 – πλατφόρμα και προγραμματιστικά εργαλεία

Η εφαρμογή αναπτύχθηκε με χρήση του περιβάλλοντος ανάπτυξης εφαρμογών visual studio 2005. Αφού πρόκειται για διαδικτυακή εφαρμογή (στην ουσία website) χρησιμοποιήθηκε το asp .net κομμάτι του visual studio. Για την επεξεργασία του project δεν είναι απαραίτητη η χρήση του visual studio, αλλά αρκεί η χρήση των .NET Framework και Visual Web Developer. Η έκδοση του asp .net που χρησιμοποιείται στο visual studio 2005 (επομένως και στην εφαρμογή που αναπτύχθηκε) είναι η 2.0 . Η γλώσσα προγραμματισμού που χρησιμοποιήθηκε για τον code behind της εφαρμογής είναι η c sharp.

5.2 – χρήση εφαρμογής

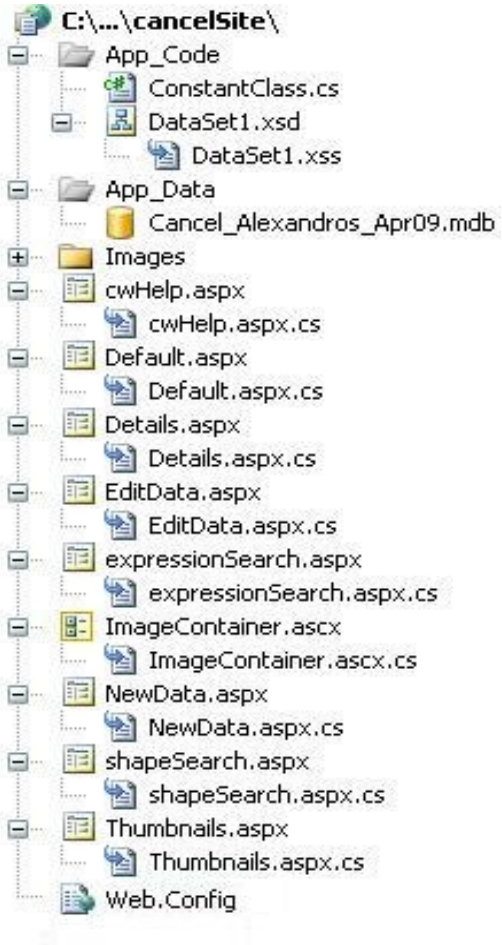
Για τη χρήση της εφαρμογής απαιτείται ένας απλός web browser και μια σύνδεση στο internet, από τη στιγμή που η εφαρμογή θα έχει δημοσιευθεί στο internet. Όλες οι λειτουργίες γίνονται στην πλευρά του server (server side programming) και το μόνο που βλέπει ο χρήστης της εφαρμογής είναι σελίδες html. Για τρέξιμο της εφαρμογής στον υπολογιστή του χρήστη δείτε την παράγραφο 5.3 (επεξεργασία εφαρμογής).

5.3 – επεξεργασία εφαρμογής

Για χρήση της εφαρμογής σε ένα υπολογιστικό σύστημα (είτε για να τρέξει σ' αυτό, είτε για να τροποποιηθεί) χρειάζεται το περιβάλλον ανάπτυξης visual studio. Οι οδηγίες που ακολουθούν αφορούν το visual studio 2005 και μπορεί να διαφέρουν σε άλλες εκδόσεις του visual studio ή αν χρησιμοποιηθεί το Visual Web Developer.

i. Εκκίνηση του visual studio 2005
ii. File -> New -> Web Site
iii. δίνουμε ένα όνομα και δημιουργούμε το project
iv. στο καινούριο project που δημιουργήθηκε στο προηγούμενο βήμα, προσθέτουμε τον κώδικα και τη βάση δεδομένων της εφαρμογής διατηρώντας τη σχετική δομή των φακέλων στους οποίους αυτά περιέχονται. Ο κώδικας και η βάση δεδομένων μπορούν να βρεθούν στο cd-rom που συνοδεύει την παρούσα διπλωματική. Η μεταφορά μπορεί να γίνει από το menu WebSite -> Add Existing Item, από όπου και μεταφέρουμε όλα τα αρχεία, ή απλά ρίχνοντας όλα τα αρχεία στον φάκελο όπου έχει δημιουργηθεί για το συγκεκριμένο project. Η δομή των αρχείων όπως πρέπει να φαίνεται μέσα από το visual studio απεικονίζεται στην εικόνα 5.1 .

v. Μπορούμε πλέον να μεταγλωττίσουμε και να τρέξουμε την εφαρμογή. Το site σηκώνεται στον server του asp .net και είναι προσπελάσιμο στην διεύθυνση localhost στο κατάλληλο port.

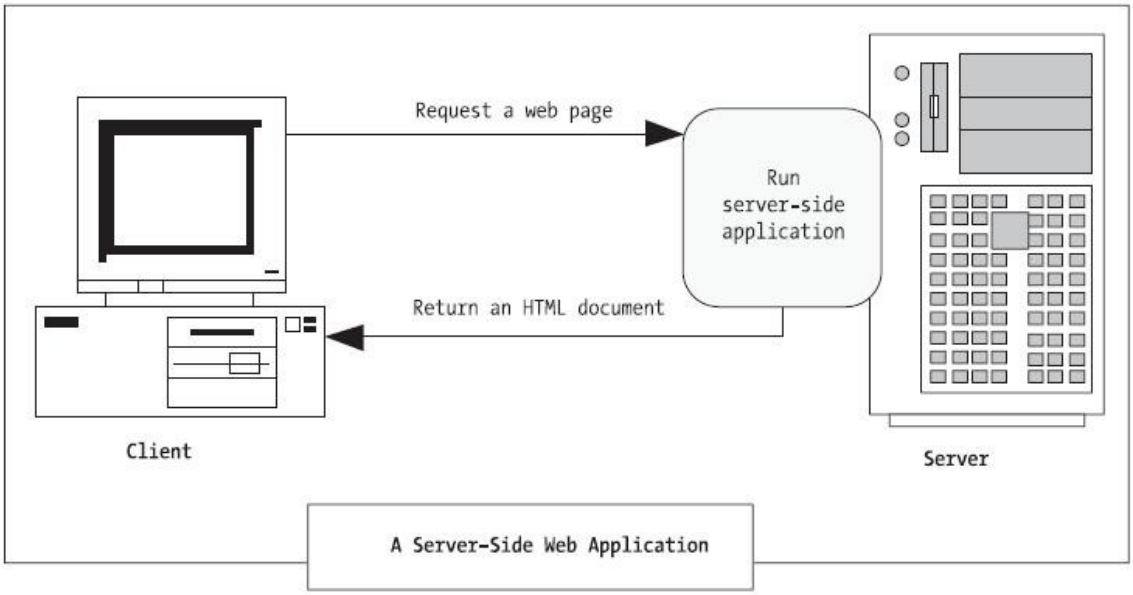


εικόνα 5.1 : δομή αρχείων του project

5.4 – Λίγα λόγια για το asp .net

Το .net επιτρέπει τη συνύπαρξη και συνεργασία κώδικα γραμμένου σε διαφορετικές αντικειμενοστρεφείς γλώσσες προγραμματισμού (Visual Basic, C sharp, C++, J sharp...). Κάθε κλάσση γραμμένη σε οποιαδήποτε από τις παραπάνω γλώσσες μεταγλωττίζεται αρχικά σε κάποια κοινή γλώσσα ενδιάμεσου κώδικα ο οποίος εκτελείται απο το Common Language Runtime(CLR). Για την ανάπτυξη της εφαρμογής χρησιμοποιήθηκε αποκλειστικά c sharp.

Το μοντέλο προγραμματισμού είναι server side, δηλαδή τα πάντα εκτελούνται στην μεριά του server. Αυτό σχηματικά περιγράφεται από την εικόνα 5.2 . Πλεονεκτήματα που προκύπτουν από αυτό είναι ότι ο χρήστης δε χρειάζεται να έχει εξειδικευμένο λογισμικό, ζητήματα ασφάλειας, απόκρυψης πληροφοριών,...



εικόνα 5.2 : server side programming

Ο κώδικας της εφαρμογής αποτελείται από σελίδες-φόρμες (αρχεία .aspx) τα οποία περιέχουν τα διάφορα αντικείμενα των σελίδων που εμφανίζονται στο χρήστη. Αυτά τα αντικείμενα λέγονται controls. Η βιβλιοθήκη του .net περιέχει ένα μεγάλο αριθμό από controls που υλοποιούν λειτουργίες που χρειάζονται συχνά σε εφαρμογές. Π.χ. controls για κουμπιά, πλαίσια κειμένου, πίνακες δεδομένων, λίστες, υποδοχείς για εικόνες, αλλά και controls που σχετίζονται με βάσεις δεδομένων και διευκολύνουν την επικοινωνία με αυτές (ανάγνωση, τροποποίηση, παρουσίαση δεδομένων). Κάθε control συνδέεται με έναν αριθμό από ιδιότητες, αλλά και έναν αριθμό από γεγονότα που εγείρονται μετά από κάποιο ερέθισμα (π.χ. ένα κουμπί 'πετάει' ένα γεγονός όταν το πατήσει ο χρήστης). Αυτό το γεγονός μπορεί να συνδεθεί με μια συνάρτηση η οποία εκτελείται όταν πυροδοτείται το γεγονός.

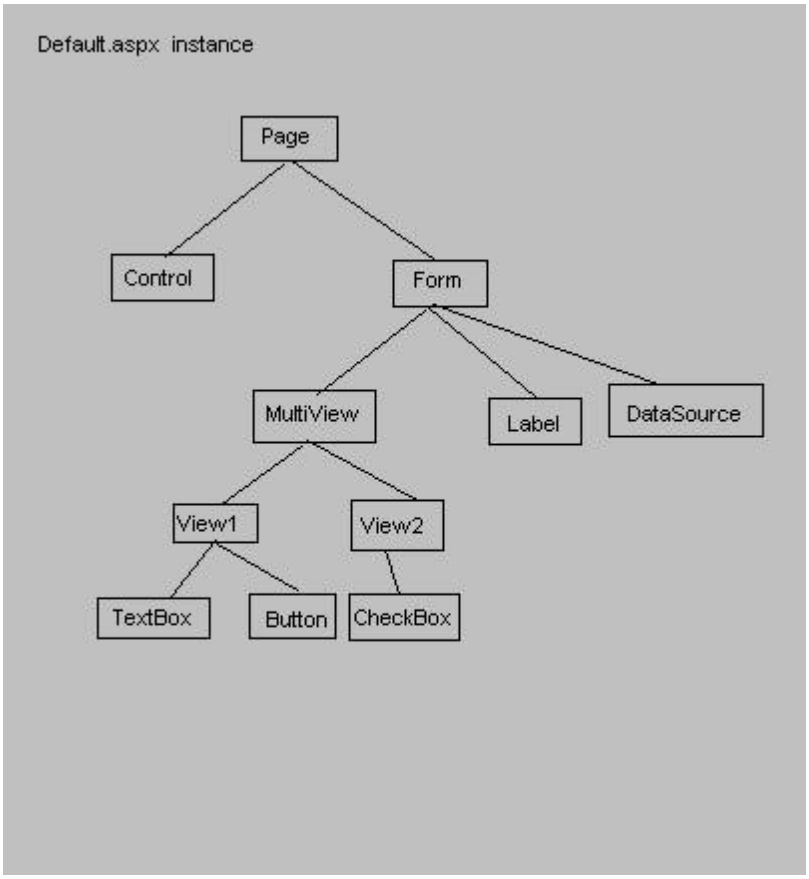
Με κάθε αρχείο aspx μπορεί να συσχετιστεί ένα αρχείο aspx.cs . Όπως υποδηλώνει το cs πρόκειται για κώδικα γραμμένο σε γλώσσα c sharp. Αυτός ο κώδικας λέγεται code behind και μπορεί να ενσωματώνει τη λειτουργικότητα που απαιτείται για τη συγκεκριμένη σελίδα (π.χ. συναρτήσεις που εκτελούνται μετά από πυροδότηση γεγονότων). Έτσι μπορεί να επιτυγχάνεται μια διάκριση ανάμεσα στην εμφάνιση της σελίδας (αρχεία aspx) και την λειτουργικότητα (αρχεία aspx.cs). Αυτή η διάκριση βέβαια δεν γίνεται να είναι απόλυτη. Άλλωστε, μέρος της λειτουργικότητας μιας εφαρμογής είναι και η τροποποίηση της εμφάνισης, ειδικά αν πρόκειται για μια διαδραστική, φιλική προς το χρήστη εφαρμογή.

Εκτός από τα controls της βιβλιοθήκης του .net, ο προγραμματιστής έχει τη δυνατότητα να ορίσει και νέα δικά του controls για να προτυποποιήσει λειτουργικότητα που του χρειάζεται συχνά. Υπάρχουν δύο είδη τέτοιων controls, τα user controls και τα custom controls. Τα user controls είναι άμεσα συσχετισμένα με μια εφαρμογή και υλοποιούν κάποια λειτουργικότητα που χρειάζεται σε περισσότερες από μια σελίδες της εφαρμογής. Έτσι ο προγραμματιστής φτιάχνει ένα user control αντί να αντιγράφει παρόμοιο κώδικα σε όλες τις σελίδες. Τα custom controls δεν σχετίζονται με μια συγκεκριμένη εφαρμογή, αλλά υλοποιούν λειτουργικότητα γενικότερη, η οποία μπορεί να χρειάζεται σε διαφορετικές εφαρμογές. Με αυτόν τον τρόπο μπορεί να επεκτείνει τις βιβλιοθήκες με controls για το περιβάλλον .net .

5.5 – μοντέλο εκτέλεσης σελίδας και κύκλος ζωής των controls

Το μοντέλο εκτέλεσης σελίδας (page execution model) είναι ο τρόπος με τον οποίο δημιουργείται μια σελίδα από τον server για απάντηση σε μια εισερχόμενη αίτηση. Το πρωτόκολλο που χρησιμοποιείται είναι το http. Ο χρήστης (μέσω ενός web browser) στέλνει μια αίτηση http στον server ζητώντας μια σελίδα. Ο server δέχεται την αίτηση, την επεξεργάζεται και παράγει μια σελίδα http σαν απάντηση, την οποία αποστέλει πίσω στον browser του χρήστη για προβολή.

Μια σελίδα αποτελείται από εναν αριθμό controls τα οποία είναι διατεταγμένα σε μια ιεραρχική μορφή, δηλαδή κάθε control μπορεί να περιέχει άλλα controls. Η ίδια η σελίδα είναι ένα control με τις δικές της ιδότητες και γεγονότα. Η ιεραρχική αυτή μορφή φαίνεται στην εικόνα 5.3 .

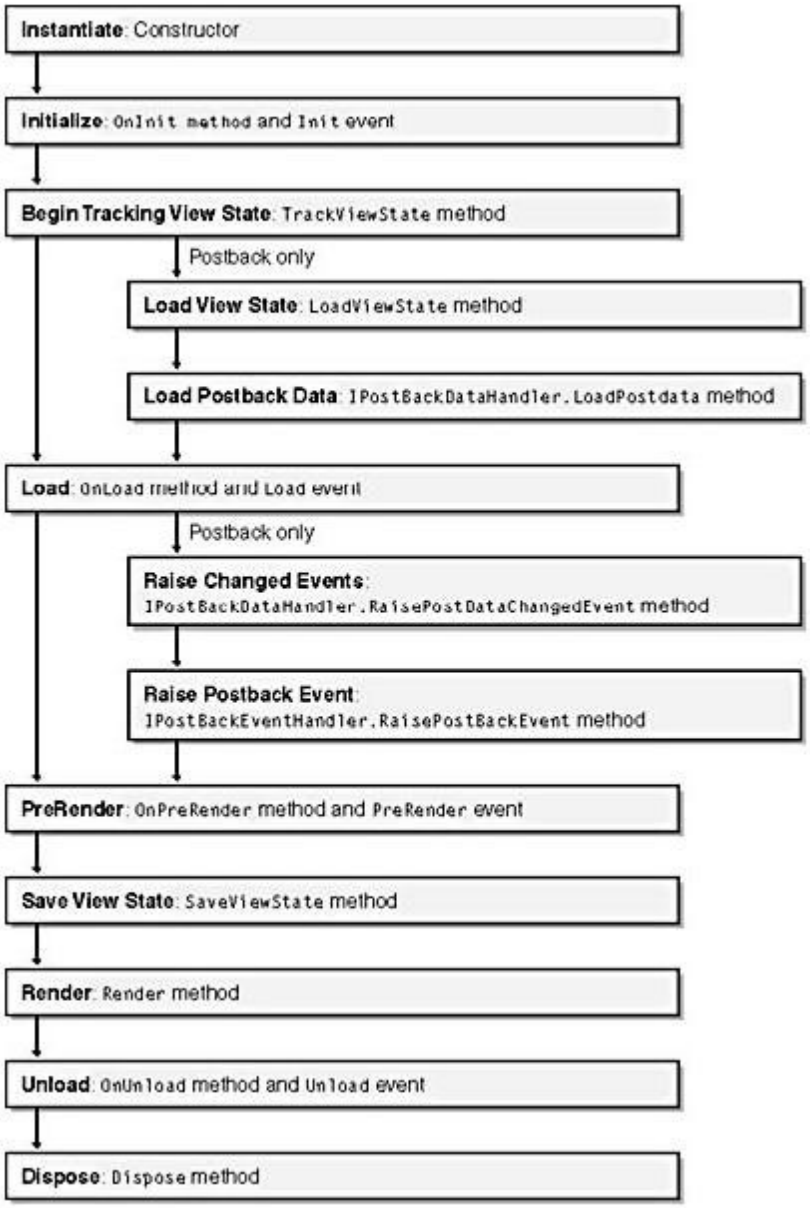


εικόνα 5.3 : ιεραρχική μορφή controls σελίδας

Κάθε control μπορεί να προκαλεί postbacks, δηλαδή εκ νέου αιτήσεις στον server. Τις αιτήσεις αυτές χειρίζεται ο server όπως τις αρχικές. Το αποτέλεσμα τους μπορεί να είναι η εκτέλεση διάφορων διεργασιών στον server και η επιστροφή της ίδιας ή άλλης σελίδας στον browser του χρήστη (π.χ. μετά το πάτημα ενός κουμπιού μπορεί να βλέπουμε την ίδια σελίδα τροποποιημένη ή μια καινουρια σελίδα).

Η δημιουργία της σελίδας συνίσταται στη δημιουργία των controls που περιέχονται στην ιεραρχία και στην εκτέλεση του κώδικα που σχετίζεται με διάφορα γεγονότα. Κατά τη δημιουργία κάθε σελίδας πυροδοτούνται κάποια συγκεκριμένα γεγονότα (π.χ. Initialize,

Page Load, Render) αλλά και όσα πυροδοτούνται ως αποτέλεσμα ενεργειών του χρήστη (π.χ. πάτημα ενός κουμπιού). Κάθε control (όπως και η σελίδα) έχει το δικό του κύκλο ζωής. Ο κύκλος αυτός φαίνεται στην εικόνα 5.4 .



εικόνα 5.4 : κύκλος ζωής ενός control

- Instantiate
  Δημιουργείται το control μετά από κλήση του constructor του. Η κλήση αυτή μπορεί να γίνει είτε από τη σελίδα είτε από άλλο control.
- Initialize
  Αρχικοποιείται το control. Παίρνει αρχικές τιμές με βάση τις ιδιότητες στις οποίες έχουν ανατεθεί τιμές δηλωτικά-declaratively (δηλαδή με βάση τις τιμές που υπάρχουν στο αρχείο .aspx στη δήλωση του συγκεκριμένου control). Πυροδοτείται το event Init. Αν πρόκειται για σελίδα εκτελείται η μέθοδος Page_Init, αλλιώς μπορούμε να κάνουμε override τη μέθοδο OnInit του control για να προσθέσουμε κάποια λογική που πρέπει να εκτελείται σ' αυτή τη φάση.

- Begin Tracking View State
  εκτελείται η μέθοδος TrackViewState. Τιμές ιδιοτήτων που αφορούν το συγκεκριμένο control και έχουν αποθηκευθεί στο ViewState επανααναθέτονται. (για περισσότερες πληροφορίες για το ViewState δες παράγραφο 5.6)
- Load
  Σ' αυτή τη φάση τα controls έχουν επαναφερθεί στην κατάσταση που είχαν πριν το postback(ή στην αρχική κατάσταση αν δεν έχει γίνει postback) και είναι ασφαλές να προσπελαστούν. Πυροδοτείται το load event. Αν πρόκειται για σελίδα εκτελείται η μέθοδος Page_Load, αλλιώς μπορούμε να κάνουμε override τη μέθοδο OnLoad του control
- Raise Changed Events (postback only, optional)
  Αν το control προκαλεί γεγονότα όταν αλλάζει η κατάστασή του, δηλαδή κάποιες τιμές ιδιοτήτων του, τότε αυτή είναι η φάση στην οποία πυροδοτούνται αυτά τα γεγονότα.
- Raise PostBack Events (postback only, optional)
  Αν το control προκαλεί postbacks μετά από κάποια ενέργεια του χρήστη, τότε αυτή είναι η φάση στην οποία πυροδοτείται το αντίστοιχο γεγονός. Π.χ. αν το πάτημα ενός κουμπιού σε μια φόρμα προκαλεί postback στον server, τότε σε αυτή τη φάση πυροδοτείται το γεγονός και εκτελείται η μέθοδος που έχει δεθεί με αυτό οτ γεγονός.
- PreRender
  Πυροδοτείται το PreRender γεγονός (override τη μέθοδο OnPreRender)
- SaveViewState
  Σώζονται οι πληροφορίες που αφορούν το αντικείμενο ViewState από όσα controls το χρησιμοποιούν.
- Render
  Παράγεται html κείμενο που ενσωματώνεται στην σελίδα που θα επιστραφεί στο χρήστη και αφορά το συγκεκριμένο control.
- Unload
  'Καθάρισμα' της σελίδας ή του control
- Dispose
  Απελευθέρωση πόρων που πιθανώς δεσμεύει το control.

Σε κάθε postback η κάθε σελίδα δημιουργείται εκ νέου με επαναφορά των αρχικών τιμών (αυτών που περιέχονται στο δηλωτικό κομμάτι του κώδικα). Έτσι χάνονται οι πιθανές αλλαγές-επιλογές του χρήστη (π.χ. το επιλεγμένο στοιχείο από μια λίστα). Γι αυτό υπάρχει ανάγκη για έναν μηχανισμό που θα αποθηκεύει και θα επαναφέρει τις συγκεκριμένες τιμές. Αυτόν το ρόλο τον παίζει το αντικείμενο ViewState. Επίσης υπάρχει ανάγκη για επικοινωνία ανάμεσα σε διαφορετικές σελίδες (π.χ. κράτηση επιλογών που ο χρήστης έχει κάνει σε μια προηγούμενη σελίδα). Αυτόν το ρόλο το παίζει το αντικείμενο Session. Τα αντικείμενα αυτά περιγράφονται στην επόμενη παράγραφο.

5.6 - Τα αντικείμενα ViewState και Session

Το πρωτόκολλο http δεν κρατάει την κατάσταση της επικοινωνίας (stateless). Το ίδιο κάνουν και οι aspx σελίδες. Δημιουργούνται, αρχικοποιούνται, εκτελούνται, προβάλλονται και διαγράφονται σε κάθε αίτηση στον server. Οι διάφορες εφαρμογές έχουν ανάγκη να κρατούν με κάποιο τρόπο την κατάσταση της εφαρμογής. Π.χ. έστω μια σελίδα που είναι μια φόρμα στοιχείων την οποία συμπληρώνει και υποβάλει στον server ο χρήστης. Κατά την συμπλήρωση ο χρήστης κάνει ένα λάθος (βάζει μια μη αποδεκτή τιμή σε ένα πεδίο). Ο server απαντά με την ίδια φόρμα και ένα μήνυμα λάθους. Αν δεν υπάρχει τρόπος να κρατηθεί η κατάσταση της φόρμας (τιμές συμπληρωμένων πεδίων) τότε ο χρήστης πρέπει να συμπληρώσει από την αρχή όλα τα πεδία. Αν μπορεί να κρατηθεί, τότε ο χρήστης απλώς διορθώνει το λανθασμένο πεδίο. Έτσι γίνεται προφανές ότι ένας μηχανισμός που θα κρατάει την κατάσταση των controls των σελίδων είναι απαραίτητος. Αυτό το ρόλο παίζει το ViewState, το οποίο είναι ενσωματωμένο σε κάθε control, δηλαδή τα controls μπορούν να αποθηκεύουν την κατάστασή τους στο ViewState και να την ξαναφορτώνουν από εκεί σε κάθε postback.

Όμως πέρα από την κατάσταση των controls, ο προγραμματιστής μπορεί να χρειάζεται να κρατά και επιπλέον πληροφορίες για την εφαρμογή (όπως π.χ. για αντικείμενα που δημιουργεί αυτός προγραμματιστικά) έτσι ώστε να τα έχει διαθέσιμα μετά από πιθανά postbacks. Και σ' αυτήν την περίπτωση μπορεί να χρησιμοποιηθεί το ViewState. Το ViewState είναι μια συλλογή από αντικείμενα στην οποία ο προγραμματιστής μπορεί να προσθέσει δικά του με εύκολο τρόπο. Π.χ. ένα string που περιγράφει το όνομα μιας χώρας μπορεί να αποθηκευθεί στο ViewState με τον εξής τρόπο :

ViewState[“CountryName”] = “Greece”;

Η τιμή αυτή δεν αλλάζει μετά από postbacks στην ίδια σελίδα.Αντίθετα θα χαθεί αν γίνει postback που θα οδηγήσει σε νέα σελίδα. Δηλαδή το ViewState αφορά μια συγκεκριμένη σελίδα και διαγράφεται μαζί με τη σελίδα όταν ο χρήστης ζητήσει μια νέα σελίδα.

Σε πολλές περιπτώσεις όμως είναι χρήσιμο να μπορούν να κρατηθούν δεδομένα και ανάμεσα σε διαφορετικές σελίδες (π.χ. μια επιλογή του χρήστη σε μια αρχική φόρμα μπορεί να τροποποιεί την εμφάνιση όλου του site και επομένως όλων των επόμενων σελίδων που θα ζητήσει ο χρήστης). Αυτό όπως αναφέρθηκε παραπάνω δεν μπορεί να υλοποιηθεί με χρήση του ViewState. Γι αυτό το λόγο μπορεί να χρησιμοποιηθεί ένα άλλο αντικείμενο, το Session.

Session (σύνοδος? στα ελληνικά) είναι μια περίοδος χρόνου που ένας μοναδικός χρήστης επικοινωνεί με μια διαδικτυακή εφαρμογή (web site). Από εκεί αντλεί το όνομα του το αντικείμενο Session. Η χρονική αυτή περίοδος μπορεί να τελειώνει μετά το κλείσιμο του web browser του χρήστη ή μετά από κάποιο χρονικό διάστημα (π.χ. 20 λεπτά) κατά το οποίο ο browser δεν έχει στείλει κάποια αίτηση στον server. Το αντικείμενο αυτό (που στην ουσία είναι μια συλλογή αντικειμένων) διατηρείται καθόλη τη διάρκεια της συνόδου, εν αντιθέσει με το ViewState που διατηρείται μόνο όσο ο χρήστης κάνει postbacks ζητώντας την ίδια σελίδα.Η ανάθεση ή ανάγνωση μιας τιμής μπορεί να γίνει με τον ίδιο απλό τρόπο που γίνεται και για το ViewState :

string name = Session[“CountryName”];

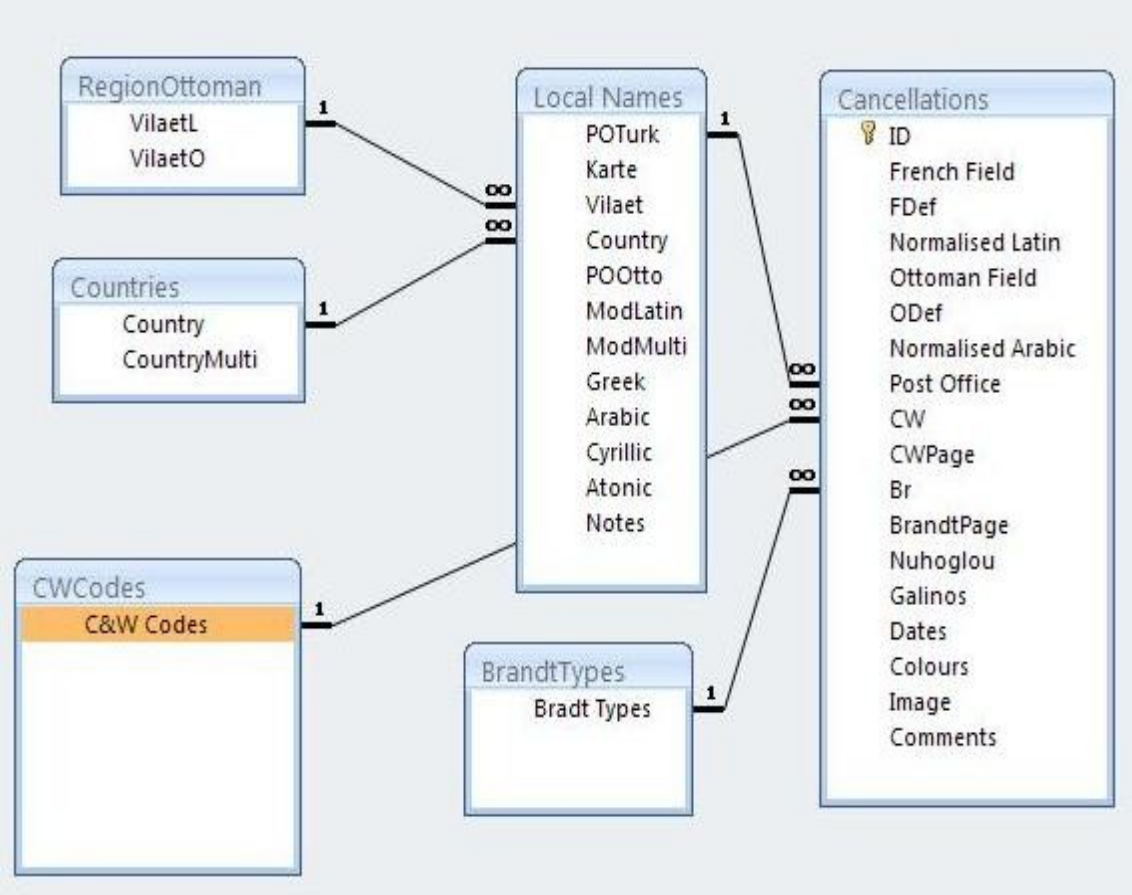
Εκτός από την διαφορά στη διάρκεια ζωής των αντικειμένων Session και ViewState (ολόκληρη σύνοδος για το πρώτο, ίδια σελίδα για το δεύτερο) υπάρχουν κι άλλα χαρακτηριστικά που οδηγούν στην προτίμηση του ενός από το άλλο για μια συγκεκριμένη περίπτωση. Αυτά είναι τα εξής :

- Μέγεθος δεδομένων. Οι πληροφορίες που αποθηκεύονται στο ViewState κωδικοποιούνται και αποστέλονται μαζί με τη σελίδα, τόσο όταν η σελίδα είναι μια απάντηση από τον server στον browser, όσο και όταν η σελίδα είναι ένα postback από τον browser στον server. Επομένως μεγάλο μέγεθος δεδομένων, δεν είναι αποδοτικό να αποθηκεύεται στο ViewState, αλλά στο Session το οποίο είναι προσβάσιμο μόνο στον server και δεν αποστέλεται στον browser σε κάθε επικοινωνία.
- Ασφάλεια δεδομένων που δεν προβάλλονται στη διεπαφή χρήστη της εφαρμογής. Αν και με το αντικείμενο ViewState υπάρχει δυνατότητα κωδικοποίησης και κρυπτογράφησης των δεδομένων, είναι πιο ασφαλές να χρησιμοποιείται το αντικείμενο Session (το οποίο δεν αποστέλεται καθόλου στον browser) όταν η ασφάλεια των δεδομένων κρίνεται σημαντική.

Κατά την ανάπτυξη της παρούσας διπλωματικής έχουν χρησιμοποιηθεί τόσο το ViewState όσο και το Session όπως περιγράφεται παρακάτω.

5.7 – η βάση δεδομένων

Το σχήμα της βάσης δεδομένων φαίνεται στην εικόνα 5.5 .



εικόνα 5.5 : σχήμα βάσης δεδομένων

Η βάση έχει δημιουργηθεί με χρήση της microsoft access.

Ο πίνακας RegionOttoman περιέχει τα ονόματα των vilaet. VilaetL είναι το όνομα στα λατινικά και VilaetO είναι το όνομα στα οθωμανικά. Το VilaetL είναι unique(μοναδικό) στον πίνακα RegionOttoman για να γίνεται η συσχέτιση πολλά προς ένα από τον πίνακα LocalNames προς τον πίνακα RegionOttoman. Δηλαδή δεν μπορούν να υπάρχουν δύο εγγραφές με το ίδιο VilaetL στον πίνακα RegionOttoman.

Ο πίνακας Countries περιέχει τα ονόματα των χωρών, σε λατινικά (ιδιότητα Country) και πολυγλωσσικά (ιδιότητα CountryMulti). Η ιδιότητα Country είναι μοναδική.

Ο πίνακας LocalNames περιέχει τα ονόματα των ταχυδρομείων. Η ιδιότητα POTurk είναι μοναδική. Το όνομα των ταχυδρομείων αποθηκεύεται σε διαφορετικές διαλέκτους (POTurk, POOtto, ModLatin, ModMulti, Greek, Arabic, Cyrillic, Atonic) όπως έχει αναφερθεί στο κεφάλαιο 2.

Ο πίνακας CWCodes περιέχει τους κωδικούς coles and walker, ενώ ο πίνακας BrandtTypes τους κωδικούς brandt.

Ο πίνακας Cancellations περιέχει τις πληροφορίες για τις σφραγίδες. Τα διάφορα πεδία αυτού του πίνακα συσχετίζουν μια εγγραφή με άλλους πίνακες εκεί που οι τιμές των ιδιοτήτων προέρχονται από μια προκαθορισμένη λίστα (Post Office, CW, Br). Το πεδίο Image περιέχει διευθύνσεις εικόνων της σφραγίδας που χωρίζονται από τον χαρακτήρα ',' .

Η ιδιότητα Colours είναι κατάλληλα κωδικοποιημένη ώστε να περιέχει πληροφορίες για τα χρώματα που εμφανίζονται σε μια σφραγίδα, το σχήμα της και τις εκφράσεις που εμφανίζονται σ΄αυτήν. Οι εκφράσεις και το σχήμα αφορούν μόνο τις αρνητικές σφραγίδες.Αν μια σφραγίδα είναι αρνητική, τότε το πεδίο Colours αρχίζει με το string "neg" . Στη συνέχεια ακολουθούν τα διάφορα χρώματα που εμφανίζονται σ' αυτήν χωρισμένα με κόμα (blk, blu, brn, crm, grn, red, vlt για τα χρώματα black, blue, brown, carmine, green, red, violet αντίστοιχα). Στη συνέχεια μπορεί να περιέχει έναν κωδικό της μορφής 1xxx που περιγράφει το σχήμα. Το πρώτο x αντιστοιχεί στο σχήμα της σφραγίδας, το δεύτερο x στο περίγραμμα, το τρίτο x στο αν έχει αστέρι στην κορυφή ή όχι. Τέλος μπορεί να περιέχει έναν κωδικό της μορφής 2xxxxxxxxxx όπου κάθε x αντιστοιχεί σε μία έκφραση και δείχνει αν αυτή εμφανίζεται και σε ποιο μέρος πάνω στη σφραγίδα. Οι επιτρεπόμενες τιμές του x σ' αυτήν την περίπτωση είναι t (top), m (middle), b (bottom), w (unclear), 0 (non existent). Μη αρνητικές σφραγίδες περιέχουν στο πεδίο Colours μόνο μια σειρά από χρώματα.Παρακάτω παρατίθενται ενδεικτικά κάποιες τιμές της ιδιότητας Colours :

blk,blu,vlt,
blu,blk,
neg,
neg,1acn,20000bb00mm,
neg,1cnn,
neg,1cnn,20000mmbm00,
neg,blk,blu,grn,1cry,200ww00ww00,

Η σύνδεση της εφαρμογής με τη βάση δεδομένων γίνεται μέσω του data provider Microsoft.Jet.OLEDB.4.0 . Αξίζει να σημειωθεί ότι ο data provider χρησιμοποιεί διαφορετικά wildcards από τη βάση δεδομένων access (π.χ. % αντί *, _ αντί ?).

5.8 – λεπτομέρειες υλοποίησης - περιγραφή κώδικα

Σ' αυτήν την ενότητα γίνεται αναφορά στις λεπτομέρειες της υλοποίησης, περιγράφοντας τι κάνει το κάθε αρχείο/κλάση κώδικα και πως αυτά επικοινωνούν μεταξύ τους για την παραγωγή της επιθυμητής λειτουργικότητας.

5.8.1 – κλάση ConstantClass.cs

Στην κλάση αυτή έχουν δημιουργηθεί διάφορες μέθοδοι οι οποίες είναι χρήσιμες σε άλλες σελίδες της εφαρμογής. Η κλάση αυτή έχει τοποθετηθεί στο φάκελο App_Code έτσι ώστε να είνια προσβάσιμη από όλες τις σελίδες της εφαρμογής που θέλουν να χρησιμοποιήσουν τις μεθόδους της. Οι μέθοδοι δηλώνονται ως public static για να μπορεί να τις προσπελαύνει οποιαδήποτε σελίδα με τον εξής απλό τρόπο :

ConstantClass.NameOfMethod(parameter1, parameter2, ...) ;

Οι μέθοδοι που είναι μέλη αυτής της κλάσης υλοποιούν κυρίως επεξεργασία πάνω σε strings που έχει σχέση με γλωσσικές απαιτήσεις της εφαρμογής ή γενίκευσης των αναζητήσεων μέσω της τροποποίησης των strings που περνούν ως παράμετροι στα διάφορα queries. Τέτοιο παράδειγμα είναι η επέκταση των χαρακτήρων ενός string αναζήτησης έτσι ώστε να περιέχει και παρόμοιους με τους αρχικούς χαρακτήρες, π.χ. αν κάποιος δίνει το string “παράδειγμα” για αναζήτηση τότε μια συνάρτηση επέκτασης θα μπορούσε να επίστρεφει το string “[πΠ][αΑάΆ][ρP][αΑάΆ][δΔ][εΕέΕ][ιΙίΙ][γΓ][μM][αΑάΆ]” το οποίο μπορεί να επιστρέψει περισσότερα αποτελέσματα. Οι χαρακτήρες '[' και ']' αποτελούν wildcards που μπορούν να σταλούν στη βάση και σημαίνουν ένα οποιοδήποτε από τα σύμβολα που περιέχονται ανάμεσά τους. Δηλαδή μια αναζήτηση με το επεκταμένο string ταιριάζει με strings όπως “παράδειγμα”, “Παραδειγμα”, “ΠΑΡΑΔΕΙΓΜΑ”, ... , ενώ αναζήτηση με το αρχικό string ταιριάζει μόνο με το πρώτο από τα παραπάνω strings. Άλλες συναρτήσεις μετατρέπουν λατινικούς χαρακτήρες ενός string σε αραβικούς και αντίστροφα, προσθέτουν κατάλληλα wildcards, ελέγχουν για ύπαρξη κάποιων συγκεκριμένων χαρακτήρων σε ένα string, αφαιρούν συγκεκριμένους χαρακτήρες από strings, ...

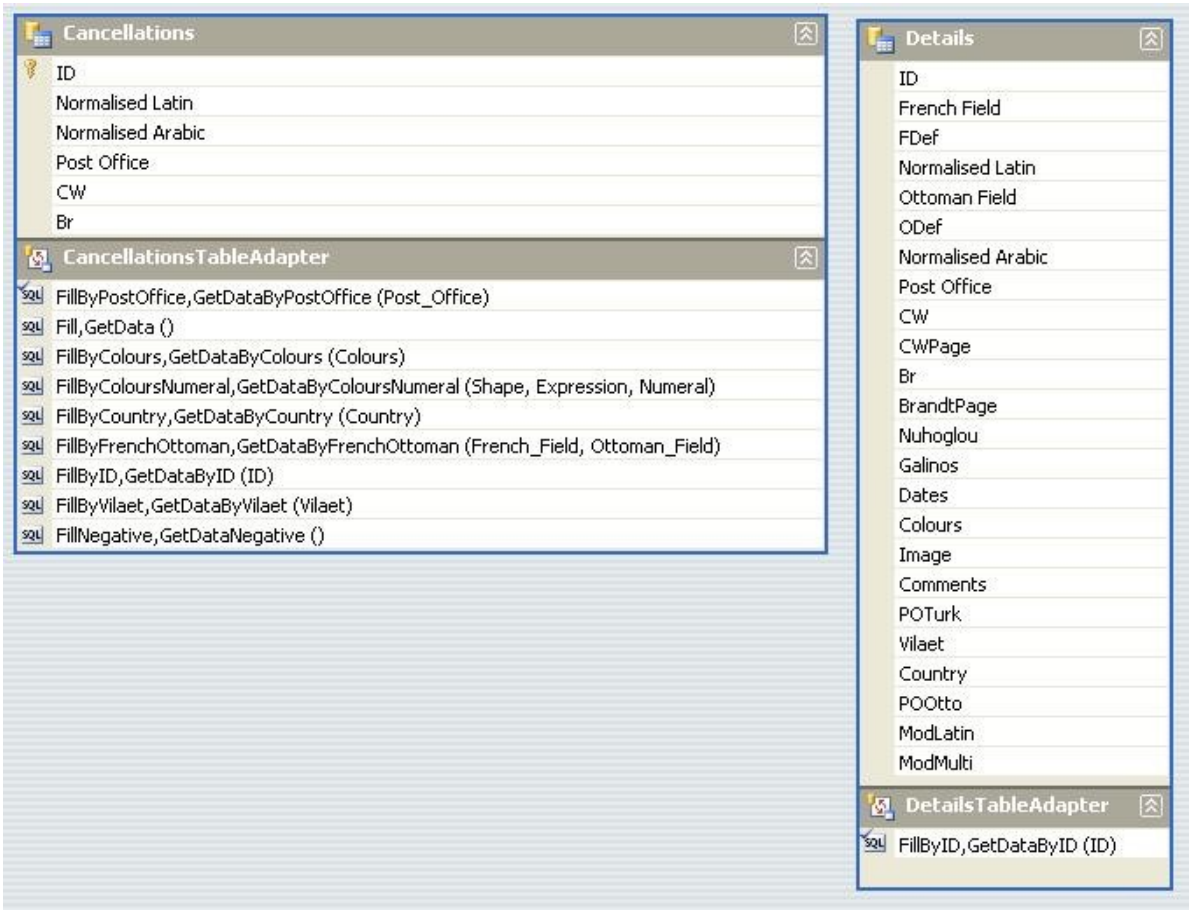
Οι τίτλοι των μεθόδων είναι περιγραφικοί και φαίνονται παρακάτω :

```
string strOneLine(string s)
bool containsFrench(string s)
string removedFrench(string s)
bool containsOttoman(string s)
string removedOttoman(string s)
string expandedFrench(string frenchText)
string expandedOttoman(string ottoText)
string wildcarded_fr(string expandedtext)
string wildcarded_ot(string expandedText)
string strAllArabised(string text)
string strNumArabised(string s)
string strParArabised(string s)
string strStarArabised(string s)
string strQmrkArabised(string s)
string strSemicolonArabised(string s)
string strColonArabised(string s)
string strNumWesternised(string s)
string strStarQmrkWesternised(string s)
```

5.8.2 - αρχείο DataSet1.xsd

DateSet είναι μια κλάση που αναπαριστά την αποθήκευση στη μνήμη (in-memory cache) δεδομένων που έχουν ανακτηθεί από μια πηγή δεδομένων (π.χ. βάση δεδομένων). Η χρησιμοποίησή των DataSet μειώνει τις προσβάσεις στη βάση και αυξάνει την ταχύτητα της εφαρμογής. Ένα DataSet αποτελείται από μια συλλογή πινάκων δεδομένων (DataTable) στους οποίους αποθηκεύονται τα δεδομένα που αντλούνται από τη βάση (ή προστίθενται και από το χρήστη). Η επικοινωνία με τη βάση γίνεται μέσω DataAdapters.

Για την εφαρμογή έχουν δημιουργηθεί δύο DataTables που απεικονίζονται στην επόμενη εικόνα :



εικόνα 5.6 : DataSet1.xsd

Κάθε DataTable αποτελείται από εγγραφές (γραμμές-rows) που έχουν μια σειρά από ιδιότητες. Το DataTable Cancellations (εικόνα 5.6) έχει 6 ιδιότητες, ενώ το Details 24 ιδιότητες. Οι μέθοδοι που εμφανίζονται από κάτω (FillBy, GetDataBy) είναι sql queries που απευθύνονται στη βάση (με πέρασμα κατάλληλων παραμέτρων) για να ανακτηθούν οι εγγραφές και να γεμίσουν τα DataTables. Στο DataTable Details υπάρχει μόνο ένα query που ανακτά τις λεπτομέρειες μιας εγγραφής (σφραγίδας) περνώντας ως παράμετρο το ID της εγγραφής. Στο DataTable Cancellations υπάρχουν πολλά queries με διαφορετικές παραμέτρους που δείχνουν τις διαφορετικές αναζητήσεις που μπορεί να κάνει ο χρήστης στη βάση δεδομένων.

Π.χ. το query GetDataByID(ID) του DataTable Details είναι το εξής :

```
SELECT     Cancellations.ID, Cancellations.[French Field], Cancellations.FDef,
           Cancellations.[Normalised Latin], Cancellations.[Ottoman Field],
           Cancellations.ODef, Cancellations.[Normalised Arabic],
           Cancellations.[Post Office], Cancellations.CW, Cancellations.CWPage,
           Cancellations.Br, Cancellations.BrandtPage, Cancellations.Nuhoglou,
           Cancellations.Galinos, Cancellations.Dates, Cancellations.Colours,
           Cancellations.[Image], Cancellations.Comments,
           [Local Names].POTurk, [Local Names].Vilaet, [Local Names].Country,
           [Local Names].POOtto, [Local Names].ModLatin, [Local Names].ModMulti
FROM       (Cancellations INNER JOIN
             [Local Names] ON Cancellations.[Post Office] = [Local Names].POTurk)
WHERE      (Cancellations.ID = ?)
```

ενώ το query GetDataByID(ID) του DataTable Cancellations είναι το εξής :

```
SELECT Br, CW, ID, [Normalised Arabic], [Normalised Latin], [Post Office]
FROM Cancellations
WHERE (ID = ?)
```

Το DataTable Cancellations χρειάζεται για να γεμίζει τις γραμμές της λίστας της αρχικής σελίδας (Default.aspx , εικόνα 3.2). Από αυτή τη λίστα ο χρήστης έχει τη δυνατότητα να επιλέξει μια γραμμή (μία εγγραφή) και να δει περισσότερες λεπτομέρειες γι αυτήν. Η επιλογή των λεπτομερειών γίνεται με βάση το ID της γραμμής και το DataTable Details (query GetDataByID(ID)) και αυτές προβάλλονται σε άλλη φόρμα (εικόνα 3.6).

5.8.3 – Default.aspx και Default.aspx.cs

Είναι η αρχική σελίδα που εμφανίζεται στο χρήστη και κεντρική σελίδα της εφαρμογής. Περιέχει διάφορα πλαίσια από τα οποία μπορούν να γίνουν διαφορετικές αναζητήσεις και μια λίστα στην οποία εμφανίζονται τα αποτελέσματα αυτών των αναζητήσεων. Περιέχει επίσης ένα οθωμανικό πληκτρολόγιο το οποίο περιγράφεται σε επόμενη παράγραφο (ottoman keyboard).

Η λίστα (που φαίνεται στην εικόνα 3.2) έχει υλοποιηθεί με τη χρήση του control GridView. Το όνομα αυτού του αντικειμένου είναι GridView1. Το GridView1 αντλεί τα δεδομένα των γραμμών του από ένα control τύπου ObjectDataSource με όνομα cancelDataSource. Η cancelDataSource αντλεί τα δεδομένα από το CancellationsTableAdapter που φαίνεται στην εικόνα 5.6 και έχει περιγραφεί στην προηγούμενη παράγραφο (5.8.2). Το query με βάση το οποίο αντλούνται τα δεδομένα αλλάζει για διαφορετικές αναζητήσεις του χρήστη (αναζήτηση με βάση post office/country/id/....). Οπότε με κάθε νέα αναζήτηση του χρήστη πρέπει να αλλάζουν και οι παράμετροι της cancelDataSource.

Επίσης η τελευταία αναζήτηση του χρήστη πρέπει να κρατείται όταν αυτός εγκαταλείπει την αρχική σελίδα (για να του ξαναπαρουσιαστεί όταν επιστρέψει στην αρχική σελίδα). Όπως έχει αναφερθεί παραπάνω οι σελίδες είναι stateless, δηλαδή μετά την μετακίνηση σε νέα σελίδα χάνονται τα δεδομένα τους, κι επομένως και τα περιεχόμενα του GridView1. Άρα πρέπει να αναπτυχθεί ένας μηχανισμός που να κάνει δυνατή την επαναφορά των γραμμών του GridView1 όταν ο χρήστης επιστρέφει από κάποια άλλη σελίδα της εφαρμογής. Θα μπορούσαμε να αποθηκεύουμε τα ίδια τα δεδομένα του GridView1 όταν

φεύγουμε από τη σελίδα, αλλά αυτό δε θα ήταν αποδοτικό αφού ο όγκος τους μπορεί να είναι πολύ μεγάλος (ανάλογα με την αναζήτηση). Αντ' αυτού επιλέχθηκε να αποθηκεύεται το τελευταίο query string του χρήστη και όταν αυτός επιστρέφει στην αρχική σελίδα, να ξαναεκτελείται (το query) για να ανακτηθούν τα δεδομένα. Επί της ουσίας, δεν αποθηκεύεται ακριβώς το query string, αλλά κάποιες παράμετροι της cancelDataSource οι οποίες επαναφέρονται κατά την επιστροφή στην αρχική σελίδα.Η αποθήκευση αυτών των παραμέτρων γίνεται στο Session αντικείμενο το οποίο δε διαγράφεται ανάμεσα στις μεταβάσεις από μια σελίδα της εφαρμογής σε άλλη.

Οι παράμετροι αυτές είναι

- η SelectMethod (αποθηκεύεται στο **Session[“selectMethod”]**)
  που είναι το query που χρησιμοποιείται για άντληση των δεδομένων (π.χ. GetDataByVilaet, GetDataByID,... δες εικόνα 5.6)
- ο αριθμός παραμέτρων που θέλει το συγκεκριμένο query (αποθηκεύεται στο **Session[“selectParamCount”]**)
  π.χ. GetDataByVilaet παίρνει μια παράμετρο, ενώ GetDataByFrenchOttoman παίρνει δύο
- Για την πρώτη παράμετρο αποθηκεύεται το όνομα της παραμέτρου, ο τύπος της παραμέτρου και η τιμή της παραμέτρου στα **Session[“name0”], Session[“type0”], Session[“value0”]**. Αν υπάρχει και δεύτερη παράμετρος, αυτή αποθηκεύεται στα **Session[“name1”], Session[“type1”], Session[“value1”]**. Και συνεχίζεται η αρίθμηση με τον ίδιο τρόπο για περισσότερες παραμέτρους.

Π.χ. για αναζήτηση με βάση λατινικό/οθωμανικό κείμενο η αποθήκευση γίνεται με τον παρακάτω κώδικα (από μέθοδο setSourceAsFrenchOttoman)

```
Session["name0"] = "French_Field";
Session["type0"] = TypeCode.String;
Session["value0"] = "";
Session["name1"] = "Ottoman_Field";
Session["type1"] = TypeCode.String;
Session["value1"] = "";
Session["selectParamCount"] = 2;
cancelDataSource.SelectMethod = "GetDataByFrenchOttoman";
Session["selectMethod"] = "GetDataByFrenchOttoman";
```

Επομένως κάθε φορά που ο χρήστης κάνει μια αναζήτηση ανανεώνουμε τις παραμέτρους της cancelDataSource και τις αποθηκεύουμε στο Session για μελλοντική αναφορά. Αυτό φαίνεται στον κώδικα και τις κλήσεις των συναρτήσεων setSourceAsXXX όπου XXX διαφέρει ανάλογα με την αναζήτηση (δείτε τον πλήρη κώδικα για περισσότερες λεπτομέρειες).

Όταν ο χρήστης επιστρέφει στην αρχική σελίδα, το GridView1 ανακτά τα προηγούμενα δεδομένα του. Αυτό υλοποιείται από τη συνάρτηση restoreGridViewData :

```
for (int i = cancelDataSource.SelectParameters.Count - 1; i >= 0; i--)
    cancelDataSource.SelectParameters.RemoveAt(i);
for (int i = 0; i < (int)Session["selectParamCount"]; i++)
{
    cancelDataSource.SelectParameters.Add((string)Session["name" + i], (TypeCode)Session["type" + i]
                                          (string)Session["value" + i]);
}
cancelDataSource.SelectMethod = (string)Session["selectMethod"];
```

Όταν ο χρήστης φιλτράρει τα δεδομένα μιας αναζήτησης, αυτό υλοποιείται με χρήση της FilterExpression ιδιότητας της cancelDataSource. π.χ.
cancelDataSource.FilterExpression = "CW IS NULL OR CW=' '";
Και αυτή η ιδιότητα αποθηκεύεται στο Session για επαναφορά κατά την επιστροφή στην αρχική σελίδα (**Session[“filterExpr”]**).

Τέλος το κείμενο που εμφανίζεται πάνω από το GridView1 και καταγράφει τον αριθμό των αποτελεσμάτων μιας αναζήτησης αποθηκεύεται κι αυτό στο Session για ανάκτηση. Μάλιστα αυτό μερικές φορές διαμορφώνεται από διαφορετικές σελίδες (π.χ. δες shapeSearch.aspx.cs ή expressionSearch.aspx.cs). Παράδειγμα τέτοιου κειμένου είναι το εξής :

7 circular negative cancellation(s) listed without border and star on top (with the identified standard expression(s)), having CW code S2

Το κείμενο αυτό αποθηκεύεται στα **Session[“searchText”]** και **Session[“filterText”]**. Στο παραπάνω παράδειγμα

Session[“searchText”] == “circular negative cancellation(s) listed without border and star on top (with the identified standard expression(s))”
Session[“filterText”] == “having CW code S2”

Το Session[“searchText”] έχει διαμορφωθεί από τις σελίδες shapeSearch.aspx και expressionSearch.aspx , ενώ το Session[“filterText”] από τη σελίδα cwHelp.aspx .

5.8.4 – οθωμανικό πληκτρολόγιο

Το οθωμανικό πληκτρολόγιο φαίνεται στην επόμενη εικόνα.

εικόνα 5.7 : οθωμανικό πληκτρολόγιο

Στην ουσία είναι ένας πίνακας από κουμπιά με μεθόδους που δένονται πάνω στο γεγονός πατήματος αυτών. Όταν ο χρήστης πατήσει ένα κουμπί, προκαλείται postback και το γράμμα που πάτησε ο χρήστης προστίθεται σε ένα πεδίο κειμένου (textbox). Το πληκτρολόγιο είναι αρχικά αόρατο (ιδιότητα Visible==false) και εμφανίζεται μετά το πάτημα κάποιου κουμπιού της φόρμας που περιέχει και το πληκτρολόγιο. Όταν πατηθεί το κουμπί Close του πληκτρολογίου, το πληκτρολόγιο κρύβεται ξανά. Το πλήκτρο bksp (backspace) σβήνει τον τελευταίο χαρακτήρα του παραπάνω πεδίου κειμένου. Επίσης υπάρχουν οι drop down λίστες που φαίνονται στο κάτω μέρος του πληκτρολογίου, από τις

οποίες μπορούν να επιλεχθούν κάποιες συχνές εκφράσεις. Κάθε τέτοια έκφραση που επιλέγεται προστίθεται στο παραπάνω πεδίο κειμένου (πάντα η έκφραση otto ανεξάρτητα από το αν επιλέγεται αυτή ή η αντίστοιχη έκφραση turk/engl).

Οι συναρτήσεις που σχετίζονται με το πληκτρολόγιο είναι οι :
keyboardDDL_DataBound
keyButton_Click
commonTermChosen
initKeyboard

Μια καλύτερη ιδέα θα ήταν η υλοποίηση του πληκτρολογίου ως user control ή ως custom server control, για να μπορεί να χρησιμοποιηθεί ευκολότερα σε περισσότερες σελίδες ή ακόμα και εφαρμογές.

5.8.5 – Details.aspx και Details.aspx.cs

Η σελίδα αυτή εμφανίζει τις λεπτομέρειες μιας επιλεγμένης εγγραφής (η φόρμα φαίνεται στην εικόνα 3.6). Αποτελείται από κάποια controls στα οποία τοποθετούνται οι επιμέρους ιδιότητες της σφραγίδας (πάνω μέρος της φόρμας) και τη λίστα από την οποία επιλέγη η συγκεκριμένη σφραγίδα (κάτω μέρος φόρμας). Η λίστα αυτή χρησιμεύει στο χρήστη για να μπορεί να δει τις λεπτομέρειες μιας άλλης εγγραφής η οποία ανήκει στα αποτέλεσματα της ίδιας αναζήτησης χωρίς να χρειάζεται να κάνει εκ νέου την αναζήτηση.

Η λίστα υλοποιείται με ένα control τύπου GridView (όνομα αντικειμένου GridView1) το οποίο αντλεί τα δεδομένα του από μια ObjectDataSource (με όνομα cancelDataSource). Η λογική είναι ίδια με αυτή που έχει περιγραφεί στην παράγραφο 5.8.3 για τη σελίδα Default.aspx. Το query αντλείται από το Session αντικείμενο και τα δεδομένα φορτώνονται στη λίστα από τη συνάρτηση restoreGridViewData.

Κατά την επιλογή μιας εγγραφής από τη λίστα (με χρήση του κουμπιού details) ανακτώνται από τη βάση δεδομένων οι λεπτομέρειες της εγγραφής με βάση το ID της επιλεγμένης γραμμής της λίστας. Η ανάκτηση των λεπτομερειών από τη βάση γίνεται από την ObjectDataSource με όνομα detailsDataSource η οποία απευθύνει το query στη βάση με παράμετρο το ID της εγγραφής. Τα αποτέλεσματα του select query αποθηκεύονται προσωρινά σε μια DataView. Από εκεί χρησιμοποιούνται για να γεμίσουν τα διάφορα πεδία της φόρμας (συνάρτηση loadDetails). Παράλληλα αποθηκεύονται και στη συλλογή Session για να είναι διαθέσιμα στη σελίδα EnterNewData αν ο χρήστης αποφασίσει να επεξεργαστεί την εγγραφή (περιγραφή της EnterNewData.aspx στην παράγραφο 5.8.7). Τα πεδία του Session που χρησιμοποιούνται είναι τα εξής :

Session["selectedCancelID"]
Session["selectedCancelDates"]
Session["selectedCancelCW"]
Session["selectedCancelCWPage"]
Session["selectedCancelBr"]
Session["selectedCancelBrPage"]
Session["selectedCancelNuhoglou"]
Session["selectedCancelGalinos"]
Session["selectedCancelComments"]
Session["selectedCancelFDef"]
Session["selectedCancelODef"]

Session["selectedCancelFrenchField"]
Session["selectedCancelOttoField"]
Session["selectedCancelPostOffice"]
Session["selectedCancelColours"]
Session["selectedCancelImage"]

Στη σελίδα περιλαμβάνεται ένα user contol (ImageContainer) για προβολή των εικόνων της σφραγίδας. Το control αυτό περιγράφεται στην παράγραφο 5.8.6 .

Τέλος έχουν υλοποιηθεί κάποιες συναρτήσεις για μετατροπή των string που αναπαριστούν τα χρώματα της σφραγίδας στη βάση, σε html literals έτσι ώστε να μπορούν αυτά να προβληθούν στο χρήστη. Οι συναρτήσεις αυτές είναι οι
colorsToString
colorAcronymToString
colorAcronymToCode

5.8.6 – ImageContainer.ascx και imageContainer.ascx.cs

Το δηλωτικό μέρος του user control αποτελείται από μια εικόνα (control asp:Image), και δύο συνδέσμους με μορφή κειμένου (control asp:LinkButton) που χρησιμοποιούνται για πλοήγηση ανάμεσα στις εικόνες μιας εγγραφής (previous, next). Το ImageContainer φαίνεται στην εικόνα 3.6 . Στην εφαρμογή χρησιμοποιείται στις σελίδες Details.aspx και NewData.aspx .

Βασική ιδιότητα του user control είναι η ImagesUrlsString η οποία αποθηκεύεται και ανακτάται από το ViewState (όπως φαίνεται από τις μεθόδους get, set). Αυτή είναι ένα string που αναπαριστά τις διευθύνσεις των εικόνων που σχετίζονται με μια εγγραφή όπως αυτό αποθηκεύεται στη βάση δεδομένων (μια σειρά από διευθύνσεις που χωρίζονται μεταξύ τους με τον χαρακτήρα ',').

Η κλάση ImageContainer περιλαμβάνει μία private ArrayList (δεν είναι προσβάσιμη από τους χρήστες του user control) με όνομα urls, η οποία περιέχει τις πραγματικές διευθύνσεις-url των εικονών, όπως αυτές είναι αποθηκευμένες στα directories του server. Η εξαγωγή αυτών των διευθύνσεων γίνεται από το string ImagesUrlsString με χρήση της συνάρτησης loadUrlsState. Οι εικόνες στον server αποθηκεύονται όλες κάτω από το directory Images.

Η συνάρτηση fillImageContainer τοποθετεί την πρώτη εικόνα που θα βρει (από τη λίστα urls) στο control asp:Image. Αυτή είναι η εικόνα που βλέπει ο χρήστης. Αν δεν βρει καμιά εικόνα στα directories του server τότε τοποθετεί μια εικόνα-μήνυμα η οποία αναφέρει ότι δεν υπάρχουν διαθέσιμες εικόνες για τη συγκεκριμένη σφραγίδα.

Η πλοήγηση ανάμεσα στις εικόνες της σφραγίδας γίνεται με χρήση των συνδέσμων previous, next. Όταν πατηθούν οι παραπάνω σύνδεσμοι εκτελούνται οι μέθοδοι prevButton_Click, nextButton_Click οι οποίες προβάλουν την προηγούμενη, επόμενη διαθέσιμη εικόνα από αυτές που περιλαμβάνονται στη λίστα urls.

Τέλος η μέθοδος getDisplayedImageUrl επιστρέφει ένα string το οποίο αναπαριστά τη διεύθυνση της εικόνας η οποία προβάλεται.

5.8.7 – NewData.aspx και NewData.aspx.cs

Αυτή η σελίδα δίνει τη δυνατότητα στο χρήστη να τροποποιήσει τη βάση δεδομένων προσθέτοντας, διαγράφοντας ή ενημερώνοντας εγγραφές. Έχει τρεις διαφορετικές όψεις (φόρμες). Η πρώτη είναι η κύρια φόρμα που περιλαμβάνει όλα τα πεδία της εγγραφής (εικόνα 3.7). Πατώντας το κουμπί Shape εμφανίζεται μια άλλη φόρμα (η δεύτερη όψη, παραμένουμε στην ίδια σελίδα) . Αυτή η φόρμα χρησιμεύει στην εισαγωγή του σχήματος της σφραγίδας μέσω μιας γραφικής διεπαφής χρήστη. Η τρίτη φόρμα-όψη της σελίδας εμφανίζεται πατώντας το κουμπί Expression και χρησιμοποιείται για την εισαγωγή του string που περιγράφει ποιες εκφράσεις εμφανίζονται πάνω στην σφραγίδα. Οι όψεις για εισαγωγή shape και expression μοιάζουν με τις φόρμες για αναζήτηση shape και expression (εικόνες 3.9 και 3.10 αντίστοιχα).

Οι διαφορετικές όψεις της ίδιας σελίδας υλοποιούνται με το control MultiView. Αυτό το control είναι ένα "πλαίσιο" στο οποίο τοποθετούνται controls τύπου View. Κάθε στιγμή είναι ενεργό (δηλαδή ορατό στο χρήστη) μόνο ένα View. Κάθε View περιέχει άλλα controls, τα οποία αποτελούν τη φόρμα που βλέπει ο χρήστης. Στη συγκεκριμένη εφαρμογή έχουν υλοποιηθεί τρία View, με ονόματα View1, shapeView, exprView. Το View1 είναι αυτό που φορτώνεται στο αρχικό φόρτωμα της σελίδας .

Η **όψη View1** (εικόνα 3.7) περιέχει τα διάφορα πεδία της σφραγίδας, τα οποία μπορεί να τροποποιήσει ο χρήστης. Οι αρχικές τιμές αυτών των πεδίων λαμβάνονται από το Session αντικείμενο όπου τις είχε τοποθετήσει η σελίδα Details.aspx (η διαδικασία αυτή έχει περιγραφεί στην παράγραφο 5.8.3). Η σελίδα NewData.aspx είναι προσβάσιμη μόνο μέσα από τη σελίδα Details.aspx , δηλαδή δεν υπάρχει άλλη σελίδα της εφαρμογής που να έχει σύνδεσμο προς τη σελίδα NewData.aspx . Στην όψη υπάρχει το γνώριμο οθωμανικό πληκτρολόγιο που εμφανίζεται μετά το πάτημα του αντίστοιχου κουμπιού (για το πληκτρολόγιο δείτε παράγραφο 5.8.4). Στο κάτω μέρος της όψης εμφανίζονται τα κουμπιά υποβολής των αλλαγών στη βάση δεδομένων (delete, create, update). Πριν την αποστολή των αλλαγών στη βάση δεδομένων γίνεται έλεγχος της εγκυρότητας των νέων τιμών από τη συνάρτηση checkValuesAccepted, η οποία τυπώνει κατάλληλα μηνύματα λάθους.

Στην όψη View1 υπάρχει επίσης το γνώριμο user control ImageContainer (δείτε παράγραφο 5.8.6) από το οποίο ο χρήστης μπορεί να δει τις εικόνες μιας εγγραφής. Κάτω από αυτό υπάρχουν κουμπιά για προσθαφαίρεση εικόνων. Το πλήκτρο remove image αφαιρεί από το string ImagesUrlsString του ImageContainer τη διεύθυνση της εικόνας που προβάλεται τώρα, και ξαναφορτώνει τις εικόνες στο ImageContainer (δηλαδή το νέο μειωμένο σύνολο εικόνων). Το πλήκτρο add image προσθέτει τη διεύθυνση της νέας εικόνας στο string ImagesUrlsString αφού πρώτα ανεβάσει τη νέα εικόνα στον server (και συγκεκριμένα στο directory Images/Uploaded/). Για την επιλογή της εικόνας στον browser του χρήστη, χρησιμοποιήθηκε το control asp:FileUpload. Τέλος το κουμπί restore initial images επαναφέρει το ImagesUrlsString στην αρχική του τιμή (δηλαδή αναιρεί όποιες προσθαφαιρέσεις εικόνων έχει κάνει ο χρήστης) την οποία βρίσκει στο Session αντικείμενο (εκεί που την είχε βρει αρχικά κατά το φόρτωμα των πεδίων της εγγραφής).

Η συνάρτηση getColoursParameter κατασκευάζει την τιμή του πεδίου Colours με βάση τα χρώματα της σφραγίδας (CheckBoxes στα δεξιά του ImageContainer) και το αν αυτή είναι αρνητική (πεδία κειμένου shape και expression που συμπληρώνονται από τις άλλες όψεις ή το χρήστη manually).

Η **όψη shapeView** χρησιμοποιείται για την εισαγωγή του σχήματος της σφραγίδας. Οι επιλογές του χρήστη από τις διάφορες RadioButtonLists αποθηκεύονται στο ViewsState και συγκεκριμένα στα εξής ονόματα :
ViewState["shape"], ViewState["border"], ViewState["star"].
Το συνολικό string που αναπαριστά το σχήμα προκύπτει απλώς από την συνένωση των παραπάνω strings :

```
Text = "1" + ViewState["shape"] + ViewState["border"] + ViewState["star"];
```

Η **όψη exprView** χρησιμοποιείται για την εισαγωγή του πεδίου expression. Το τελικό string προέρχεται από τη συνάρτηση scanExpression που σκανάρει τις επιλογές του χρήστη. Στη διεπαφή χρήστη εμφανίζονται εικόνες των εκφράσεων σε διάφορες παραλλαγές. Πατώντας πάνω σε μια εικόνα βλέπουμε τις παραλλαγές της. Για να ξέρουμε ποια εικόνα προβάλεται κι επομένως ποια είναι η επόμενη παραλλαγή που πρέπει να προβληθεί κρατάμε στο ViewState τον αύξοντα αριθμό της τρέχουσας εικόνας. Π.χ. αν προβάλεται η εικόνα telegr1.bmp τότε έχουμε τον αύξοντα αριθμό 1 και επόμενη παραλλαγή είναι η telegr2.bmp. Οι αποθηκευμένες παραλλαγές ακολουθούν την παραπάνω σύμβαση την ονομασία τους (δείτε κώδικα στις μεθόδους Page_Load και image_Click).

5.8.8 – shapeSearch.aspx και shapeSearch.aspx.cs

Είναι η φόρμα από την οποία ο χρήστης μπορεί να αναζητήσει σφραγίδες με βάση το σχήμα τους (εικόνα 3.9). Σε κάθε αναζήτηση φτιάχνεται το query που θα σταλεί στη βάση, καθώς και το κείμενο που θα τυπωθεί πάνω από τη λίστα αποτελεσμάτων στην αρχική σελίδα. Επειδή μετά την αναζήτηση γίνεται επιστροφή στην αρχική σελίδα για εμφάνιση των αποτελεσμάτων, δεν έχει νόημα να κάνουμε αναζήτηση σ' αυτή τη σελίδα. Επομένως αποθηκεύονται οι παράμετροι του query στο Session και η αναζήτηση θα γίνει στην Default.aspx από τη συνάρτηση restoreGridViewData όταν επιστρέψουμε σ' αυτή (η διαδικασία αυτή περιγράφεται στην παράγραφο 5.8.3).

Κάθε επιλογή του χρήστη από τα διάφορα RadioButtons της φόρμας αποθηκεύεται στο ViewState, με τα εξής ονόματα :
ViewState["shape"], ViewState["border"], ViewState["star"] για το query, και ViewState["shapeText"], ViewState["borderText"], ViewState["starText"] για το κείμενο που περιγράφει την αναζήτηση.

Όταν ο χρήστης πατήσει το search κουμπί γίνεται επιστροφή στην αρχική σελίδα για να γίνει η αναζήτηση. Πριν γίνει η επιστροφή πρέπει να μεταφέρουμε αυτά που αποθηκεύσαμε από το ViewState στο Session.Αυτό γίνεται με τη συνάρτηση setSourceAsColours :

```
    Session["selectMethod"] = "GetDataByColours";
    Session["name0"] = "Colours";
    Session["type0"] = TypeCode.String;
    Session["value0"] = "1" + ViewState["shape"] + ViewState["border"] + ViewState["star"];
    Session["selectParamCount"] = 1;
    Session["filterExpr"] = "null";
```

Επίσης πρέπει να μεταφέρουμε το κείμενο αναζήτησης στο Session.Αυτό γίνεται με τον εξής απλό τρόπο :

```
Session["searchText"] = (string)ViewState["shapeText"] + ViewState["borderText"] + ViewState["starText"];
Session["filterText"] = "";
```

Ο χρήστης έχει τη δυνατότητα να κάνει αναζήτηση λαμβάνοντας υπόψη ταυτόχρονα το σχήμα της σφραγίδας και τις εκφράσεις που εμφανίζονται σ' αυτή. Αυτό γίνεται όταν αντί να πατήσει ένα από τα δύο search κουμπιά της φόρμας, πατάει το search expression κουμπί. Σ' αυτήν την περίπτωση δε ακολουθεί επιστροφή στην αρχική φόρμα, αλλά φορτώνεται η expressionSearch.aspx σελίδα. Επομένως πρέπει να περάσουμε τις επιλογές του χρήστη (όσον αφορά το σχήμα της σφραγίδας) σ' αυτήν τη σελίδα.Η διαδικασία αυτή φαίνεται στη μέθοδο exprButton_Click, όπου οι επιλογές του χρήστη αποθηκεύονται στα Session[“shapeSearchString”] και Session["shapeSearchText"].
Π.χ.

```
Session["shapeSearchText"] = ViewState["shapeText"] + ViewState["borderText"] + ViewState["starText"]
Session["shapeSearchString"] = "1" + ViewState["shape"] + ViewState["border"] + ViewState["star"];
```

Από το Session θα πάρει τις τιμές αυτές η expressionSearch.aspx και θα τις τροποποιήσει ώστε να συμπεριλάβει και τις επιλογές του χρήστη για τις εκφράσεις. Μετά θα αποθηκεύσει τις νέες τιμές στο Session ώστε να τις πάρει η restoreGridViewData της αρχικής σελίδας που θα πραγματοποιήσει τελικά την αναζήτηση. Η διαδικασία αυτή περιγράφεται στην επόμενη παράγραφο.

5.8.9 – expressionSearch.aspx και expressionSearch.aspx.cs

Η σελίδα expressionSearch.aspx είναι η φόρμα μέσω της οποίας ο χρήστης μπορεί να κάνει αναζήτηση με βάση τις εκφράσεις που εμφανίζονται πάνω στις σφραγίδες (εικόνα 3.10). Ο χρήστης μπορεί να φτάσει σ' αυτήν τη σελίδα είτε από την αρχική σελίδα (Default.aspx) είτε από τη σελίδα shapeSearch.aspx . Σε κάθε περίπτωση η προηγούμενη σελίδα έχει διαμορφώσει κατάλληλα το πεδίο shapeSearchString του Session έτσι ώστε η expressionSearch.aspx να “καταλάβει” ποια ήταν η προηγούμενη σελίδα. Συγκεκριμένα η αρχική σελίδα έχει θέσει :

```
Session["shapeSearchString"] == null
```

και η expressionSearch.aspx δε θα λάβει υπόψη στην αναζήτηση που θα γίνει το σχήμα της σφραγίδας (γενική αναζήτηση για οποιοδήποτε σχήμα).
Αν προηγούμενη σελίδα ήταν η shapeSearch.aspx , τότε θα έχει θέσει κάποια τιμή στο Session[“shapeSearchString”] (διάφορη του null) και αυτή θα ληφθεί υπόψη κατά την αναζήτηση (αναζήτηση για συγκεκριμένο σχήμα). Αυτή η διάκριση γίνεται στο πρώτο άνοιγμα της σελίδας (μέθοδος Page_Load).

Στην Page_Load αρχικοποιούνται επίσης τα πεδία του ViewState που έχουν σχέση με την πλοήγηση ανάμεσα στις εικόνες που αποτελούν παραλλαγές μιας συγκεκριμένης έκφρασης (δείτε παράγραφο 5.8.7 για την όψη exprView).

Η έκφραση που εισάγει ο χρήστης μέσω της γραφικής διεπαφής, μετατρέπεται σε string μέσω της μεθόδου scanExpression. Το string αυτό ακολουθεί τις συμβάσεις αποθήκευσης στη βάση δεδομένων έτσι ώστε να χρησιμοποιηθεί για αναζήτηση σ'αυτήν.

Η γραφική διεπαφή περιλαμβάνει επίσης δύο πεδία κειμένου για εισαγωγή αριθμών σε hindi και arabic. Σε κάθε περίπτωση οι αριθμοί μετατρέπονται σε arabic και αντιπαραβάλλονται με το πεδίο Normalised Arabic του πίνακα Cancellations της βάσης δεδομένων.

Επομένως έχουμε συνολικά τρία πεδία με βάση τα οποία γίνεται αναζήτηση από αυτή τη φόρμα (shape, expression, Normalised Arabic). Αν κάποιο από αυτά δεν έχει συμπληρωθεί (π.χ. ο χρήστης ήρθε εδώ από την αρχική σελίδα κι επομένως δεν έχει επιλέξει shape), τότε αντικαθίσταται από το wildcard %. Το query που στέλνεται στη βάση είναι το εξής :

```
SELECT Br, CW, ID, [Normalised Arabic], [Normalised Latin], [Post Office]
FROM Cancellations
WHERE (Colours LIKE '%' + ? + '%')
        AND (Colours LIKE '%' + ? + '%')
        AND ([Normalised Arabic] LIKE '%' + ? + '%')
```

όπου τα τρία ερωτηματικά αντιστοιχούν με τη σειρά στα shape, expression, normalised arabic.

Τις παραμέτρους αυτές περνάει στην αρχική σελίδα (εκεί θα γίνει η αναζήτηση και προβολή των αποτελεσμάτων με χρήση της restoreGridViewData) η μέθοδος setSourceAsColoursNumeral :

```
Session["selectMethod"] = "GetDataByColoursNumeral";
Session["name0"] = "Shape";
Session["type0"] = TypeCode.String;
Session["value0"] = shape;
Session["name1"] = "Expression";
Session["type1"] = TypeCode.String;
Session["value1"] = expr;
Session["name2"] = "Numeral";
Session["type2"] = TypeCode.String;
Session["value2"] = numeral;
Session["selectParamCount"] = 3;
Session["filterExpr"] = "null";
```

Τέλος η expressionSearch.aspx φτιάχνει και τα Session[“searchText”] (λαμβάνοντας υπόψη και το Session[“shapeSearchText”] που της έχει περάσει η shapeSearch.aspx) και Session[“filterText”] που θα χρησιμοποιήσει και πάλι η Default.aspx .

5.8.10 – Thumbnails.aspx και Thumbnails.aspx.cs

Αυτή η σελίδα (εικόνα 3.14) είναι η φόρμα που προβάλει όσες εικόνες υπάρχουν από ένα σύνολο αποτελεσμάτων που έχουν επιστραφεί στη λίστα της αρχικής σελίδας.

Στο πρώτο άνοιγμα της σελίδας ανακτώνται οι διευθύνσεις των εικόνων από τη βάση δεδομένων. Αυτό γίνεται χρησιμοποιώντας το query της τελευταίας αναζήτησης. Για την ακρίβεια διαβάζεται η Session[“selectMethod”] (όπου έχει αποθηκευτεί το τελευταίο query από την Default.aspx), φτιάχνεται ένα νέο query το οποίο ανακτά μόνο τα πεδία ID και Image της κάθε εγγραφής των αποτελεσμάτων. Το query αυτό απευθύνεται στη βάση μέσω του control AccessDataSource με όνομα AccessDataSource1. Τα ζευγάρια (ID,Image) που επιστρέφονται αποθηκεύονται στη συλλογή Session (Session["idArray"], Session["imgArray"]), για μελλοντική πρόσβαση. Στις επόμενες επισκέψεις στην ίδια σελίδα (postbacks) οι εικόνες ανακτώνται από το Session, αντί να εκτελείται εκ νέου query στη βάση. Προτιμήθηκε η χρήση του Session αντί του ViewState όχι επειδή χρειάζεται να διατηρούνται τα δεδομένα στη μνήμη ανάμεσα σε επισκέψεις σε διαφορετικές σελίδες της εφαρμογής, αλλά λόγω του πιθανού μεγέθους των αποτελεσμάτων (Το ViewState θα μετέφερε μαζί με τη σελίδα και τα δεδομένα από και προς τον server, επιβάρυνση που δεν

είναι θεμιτή). Η διαδικασία ανάκτησης των διευθύνσεων των εικόνων, σε κάθε περίπτωση, γίνεται από τη συνάρτηση retrieveImages.

Μετά την ανάκτηση των εικόνων προβάλονται οι 12 πρώτες. Κάτω από κάθε εικόνα εμφανίζεται ένα κείμενο-σύνδεσμος που αναπαριστά το ID της εγγραφής στην οποία ανήκει αυτή η εικόνα. Αν πατηθεί αυτός ο σύνδεσμος, γίνεται μετάβαση στη σελίδα Details.aspx όπου προβάλονται οι λεπτομέρειες αυτής της εγγραφής. Πριν γίνει η μετάβαση αποθηκεύεται στο Session[“detailsID”] το επιλεγμένο ID ώστε να ξέρει η Details.aspx ποιας εγγραφής τις λεπτομέρειες να προβάλει.

Στη σελίδα υπάρχουν κάποιες πληροφορίες σχετικά με τις εικόνες που βρέθηκαν (αριθμός εικόνων, αριθμός σελίδας, σελίδα που προβάλεται) και κουμπιά πλοήγησης ανάμεσα στο σύνολο των σελίδων (12άδες εικόνων). Για το σκοπό αυτό κρατούνται κάποιοι αριθμοί στη μνήμη (αριθμός εικόνων, τρέχουσα σελίδα, ...) η οποίοι αναθέτονται από τη συνάρτηση updateNumbers.

Τέλος η displayImages είναι η μέθοδος που (με βάση τον αριθμό της σελίδας προς προβολή) γεμίζει την εκάστοτε σελίδα με τις κατάλληλες 12 εικόνες.

5.8.11 – EditData.aspx και EditData.aspx.cs

Σ' αυτήν τη σελίδα (εικόνα 3.13) ο χρήστης μπορεί να επεξεργαστεί τα post offices της βάσης δεδομένων. Η σελίδα περιλαμβάνει ένα GridView (με όνομα GridView1) στο οποίο προβάλονται οι ιδιότητες POTurk, POOtto, ModLatin, ModMulti, Vilaet, Country των επιλεγμένων post offices. Κάτω από το GridView1 υπάρχουν κάποια πεδία κειμένου όπου ο χρήστης μπορεί να επεξεργαστεί τις παραπάνω ιδιότητες.

Σ' αυτήν τη σελίδα υπάρχει πρόσβαση μόνο από την Default.aspx, και μόνο εφόσον ο χρήστης έχει επιλέξει ένα vilaet ή μια country προηγουμένως. Αυτές οι επιλογές του χρήστη αποθηκεύονται στη συλλογή Session από τη σελίδα Default.aspx ως εξής :

- Στο Session[“vilaetSelected”] true ή false, ανάλογα με το αν έχει επιλεγεί vialet ή όχι.
- Στο Session[“countrySelected”] true ή false, ανάλογα με το αν έχει επιλεγεί country ή όχι. Προφανώς τα δύο αυτά στοιχεία δεν μπορεί να είναι ταυτόχρονα true.
- Στο Session[“selectedLatin”] η επιλογή που έχει γίνει από το χρήστη (vilaet ή country) σε λατινικούς χαρακτήρες.

Το GridView1 γεμίζει με τα post offices που ανήκουν στον επιλεγμένο vilaet/country με χρήση της AccessDataSource1. Το query της AccessDataSource1 κατασκευάζεται κατά το άνοιγμα της σελίδας και αποθηκεύεται στο ViewState[“selectCommand”] για ανάκτηση στα postbakcs. Η ενημέρωση της βάσης δεδομένων γίνεται μέσω της AccessDataSource localNamesDSource με χρήση της κατάλληλης μεθόδου (Update, Delete, Insert) ανάλογα με την επιλογή του χρήστη (κουμπιά στο κάτω μέρος της φόρμας). Η localNamesDSource χρησιμοποιείται επίσης για να ελέγχεται αν υπάρχει ένα post office πριν τη διαγραφή/ενημέρωσή/δημιουργία του (μέθοδος existsPostOffice).

5.8.12 – cwHelp.aspx και cwHelp.aspx.cs

Πρόκειται για τη σελίδα που διευκολύνει στην επιλογή ενός cw κωδικού για φιλτράρισμα των αποτελεσμάτων μιας αναζήτησης (εικόνες 3.11 και 3.12).

Η φόρμα αποτελείται από μια σειρά κουμπιών στην κορυφή της, μια σειρά κουμπιών στο κάτω μέρος και ένα control MultiView ανάμεσα από αυτές. Ανάλογα με το ποιο κουμπί (από αυτά της κορυφής) είναι πατημένο, αλλάζει και η όψη που εμφανίζεται στο MultiView.

Οι επιλογές που γίνονται αλλάζουν το κείμενο του κουμπιού που βρίσκεται δεξιά του label apply coles and walker code. Αν πατηθεί αυτό το κουμπί γίνεται επιστροφή στην αρχική σελίδα αφού πρώτα έχει περαστεί σ' αυτήν η επιλογή του χρήστη. Αυτό γίνεται από τη συνάρτηση cwButton_Click η οποία θέτει τις τιμές των Session[“filterExpr”] και Session[“filterText”]. Στην αρχική σελίδα η restoreGridViewData θα διαβάσει αυτές τις τιμές και θα ανανεώσει (φιλτράρει στη συγκεκριμένη περίπτωση) τα αποτελέσματα της αναζήτησης (γι αυτή τη διαδικασία δείτε παράγραφο 5.8.3).

ΚΕΦΑΛΑΙΟ 6 : έλεγχος εφαρμογής

Σ' αυτό το κεφάλαιο γίνεται μια σύντομη περιγραφή των ελέγχων που έγιναν για να διαπιστωθεί η ορθή λειτουργία της εφαρμογής.

Ο έλεγχος ορθότητας έγινε με βάση το μοντέλο του **black box**. Σ' αυτόν τον έλεγχο η εφαρμογή θεωρείται ως ένα black box του οποίου ο πηγαίος κώδικας δεν εξετάζεται. Οι έλεγχοι επομένως γίνονται από τη σκοπιά του τελικού χρήστη. Εκτελούνται όλα τα πιθανά σενάρια χρήσης της εφαρμογής για να διαπιστωθεί αν αυτή λειτουργεί με βάση τις προδιαγραφές.

Ο έλεγχος που έγινε ήταν εξαντλητικός, δηλαδή εξετάστηκαν όλες οι πιθανές τιμές εισόδου. Στην προκειμένη περίπτωση τιμές εισόδου είναι οι αλληλουχίες εντολών που δίνει ο χρήστης, δηλαδή τα κουμπιά που πατάει, τα δεδομένα που συμπληρώνει ή ζητάει από τις διάφορες φόρμες, καθώς και όλες οι πιθανές μεταβάσεις ανάμεσα σε διαφορετικές ή και την ίδια σελίδα (π.χ. postback στην ίδια σελίδα). Εξαντλητικός δε σημαίνει όλους τους πιθανούς συνδυασμούς, αφού αυτοί είναι άπειροι, αλλά όλα τα σενάρια διαδοχής πράξεων του χρήστη. Π.χ. μετά από μια αναζήτηση μπορεί να ακολουθεί ίδια αναζήτηση με άλλες παραμέτρους, άλλου τύπου αναζήτηση, μετάβαση σε άλλη σελίδα, επιστροφή σε κάποια προηγούμενη σελίδα κτλ...

Ο αριθμός αυτών των σεναρίων είναι μεγάλος και γι αυτό το λόγο δεν καταγράφονται εδώ. Κάτι τέτοιο θα απαιτούσε πολύ χώρο και χρόνο αν αναλογιστούμε και τα screenshots που θα έπρεπε να χρησιμοποιηθούν.

Εδώ είναι χρήσιμοι δύο ορισμοί :
Θετικός είναι ο **έλεγχος** ορθότητας ενός προγράμματος που καλύπτει το φάσμα όλων των νόμιμων (αποδεκτών) τιμών εισόδου και καταδεικνύει τη σωστή λειτουργία του γι αυτές τις τιμές.
Αρνητικός είναι ο **έλεγχος** ορθότητας ενός προγράμματος που καλύπτει μη νόμιμες (μη αποδεκτές) τιμές εισόδου με σκοπό να καταδείξει ότι το πρόγραμμα συμπεριφέρεται λογικά σε περιπτώσεις λάθους από το χρήστη.

Ο έλεγχος που πραγματοποιήθηκε ήταν θετικός όσον αφορά τις μεταβάσεις από σελίδα σε σελίδα της εφαρμογής. Υπενθυμίζεται ότι για τις μεταβάσεις από σελίδα σε σελίδα πρέπει να χρησιμοποιούνται αποκλειστικά τα κουμπιά των φορμών κι όχι τα κουμπιά του browser. Επίσης κάθε session πρέπει να ξεκινάει από την αρχική σελίδα και μέσω αυτής να μεταβαίνει στις υπόλοιπες. Δεν πρέπει να υπάρχουν απευθείας αιτήσεις σελίδων (μέσω των urls τους) εκτός από την Default.aspx. Επομένως δεν έχει γίνει αρνητικός έλεγχος γι αυτές τις περιπτώσεις και η συμπεριφορά της εφαρμογής είναι άγνωστη (αν χρησιμοποιηθούν τα κουμπιά back, forward του web browser ή η μπάρα διευθύνσεων για τη μετάβαση ανάμεσα στις σελίδες).

Ο έλεγχος που πραγματοποιήθηκε όσον αφορά τα δεδομένα που εισάγονται στις διάφορες φόρμες της εφαρμογής και αποστέλονται στον server είναι και “αρνητικός”, δηλαδή εξετάστηκαν και περιπτώσεις όπου ο χρήστης εισάγει μη αποδεκτά δεδομένα. Αυτό έγινε μόνο για περιπτώσεις που είχε προβλεφθεί ένα τέτοιο ενδεχόμενο, και είχαν καθοριστεί κάποια μηνύματα λάθους για επιστροφή του χρήστη. Στην ουσία, και αυτός ο έλεγχος

ανήκει στην κατηγορία του θετικού ελέγχου, αφού η συμπεριφορά έχει προβλεφθεί κατά τη σχεδίαση των σελίδων (γι αυτό και το αρνητικός σε εισαγωγικά).

Τέλος χρησιμοποιήθηκε σε μικρό βαθμό και έλεγχος τύπου **white box** (βασίζεται στον έλεγχο του πηγαίου κώδικα). Αυτό έγινε κατά την ανάπτυξη κάποιων μεθόδων κλάσεων. Έγινε έλεγχος για τη λειτουργία τους με τη (βηματική σε debug mode ή/και κανονική) εκτέλεσή τους σε διαφορετικές συνθήκες (π.χ. διαφορετικές παραμέτρους). Ο έλεγχος αυτός σε καμία περίπτωση δεν μπορεί να θεωρηθεί επαρκής, αφού δεν περιέλαβε όλα τα κομμάτια κώδικα.. (για να θεωρηθεί πλήρης πρέπει να περιλαμβάνει όλες τις μονάδες κώδικα, όλα τα πιθανά μονοπάτια εκτέλεσης (π.χ. εντολές διακλάδωσης if,else), καθώς και τις μεγαλύτερες μονάδες κώδικα που προκύπτουν από τη συνένωση των αρχικών μονάδων)

ΚΕΦΑΛΑΙΟ 7 : πιθανές επεκτάσεις - διαφορετικές προσεγγίσεις

Η εφαρμογή που αναπτύχθηκε μπορεί να προσελκύσει το ενδιαφέρον ατόμων που έχουν ως hobby τους την ενασχόληση-συλλογή οθωμανικών σφραγίδων. Από αυτό το γεγονός προκύπτουν κάποιες ιδέες για πιθανές επεκτάσεις της εφαρμογής ή διαφορετικές προσεγγίσεις στην αρχιτεκτονική της, ώστε να μπορούν να συνδράμουν στη συλλογή στοιχείων και εικόνων οι συλλέκτες που ενδιαφέρονται.

Στην τωρινή προσέγγιση, οποιοσδήποτε χρήστης μπορεί να προσθέσει πληροφορίες ή/και εικόνες στη βάση δεδομένων ανεβάζοντας τες στο server. Μπορεί αυτό να μην είναι θεμιτό (η τροποποίηση της βάσης από οποιονδήποτε χρήστη του site). Εξάλλου μπορεί να υιοθετηθεί μια λιγότερο συγκεντρωτική προσέγγιση στην αποθήκευση των δεδομένων. Σ' αυτό το κεφάλαιο αναφέρονται κάποιες σχετικές ιδέες.

7.1 – διαφορετική προσέγγιση 1

Μια διαφορετική προσέγγιση είναι μια αποκεντρωμένη εφαρμογή (τώρα έχουμε μια συγκεντρωτική προσέγγιση αφού τα πάντα εξαρτόνται από τον server). Γι αυτό το σκοπό μπορούν να αξιοποιηθούν τα δίκτυα peer to peer (P2P), τα οποία αποτελούν αντικείμενο εκτεταμένης έρευνας τα τελευταία χρόνια.

Με βάση αυτή τη λογική μπορεί να φτιαχθεί ένα δίκτυο στο οποίο συμμετέχουν όσοι έχουν κάποιες σφραγίδες στη συλλογή τους. Πρέπει να αναπτυχθεί ένα πρόγραμμα, το οποίο θα έχει ο κάθε χρήστης μαζί με μια βάση δεδομένων. Το πρόγραμμα αυτό μαζί με το σχήμα της βάσης δεδομένων μπορεί να είναι για όλους κοινό ή να επιτρέπονται και διαφορετικές υλοποιήσεις. Στη δεύτερη περίπτωση θα πρέπει να έχει οριστεί ένα πρότυπο επικοινωνίας ανάμεσα στα διαφορετικά προγράμματα. Οι αναζητήσεις σφραγίδων θα γίνονται ανάμεσα στους συνδεδεμένους χρήστες πλήρως αποκεντρωμένα.

Μπορεί να εφαρμοστεί μια πλήρως κατανεμημένη προσέγγιση, όπου κανένας χρήστης δεν παίζει κεντρικό ρόλο (όπως αυτή που αναφέρθηκε προηγουμένως) ή μια προσέγγιση όπου θα υπάρχει κάποιος server από όπου θα αντλούνται οι εγγραφές. Σ' αυτήν την περίπτωση τα δεδομένα είναι κατανεμημένα στους χρήστες, αλλά η λογική εκτελείται από τον server. Έτσι ένας απλός χρήστης μπορεί να χρησιμοποιεί μόνο έναν web browser από τον οποίο στέλνει αιτήσεις στον server. Ο server μαζεύει όσες εγγραφές ικανοποιούν την αναζήτηση του χρήστη και τις στέλνει στον αιτούντα. Ένα άλλο είδος χρήστη, κατέχει ένα επιπλέον πρόγραμμα (μαζί με μια βάση δεδομένων) εκτός από τον web browser. Το πρόγραμμα αυτό είναι υπεύθυνο για την επικοινωνία με τον κεντρικό server της εφαρμογής και την αποστολή σ' αυτόν των εγγραφών που του ζητούνται.

7.2 – διαφορετική προσέγγιση 2

Αν δεν είναι αποδεκτή η τροποποίηση της βάσης από κάθε χρήστη του site, τότε μπορούμε απλώς να καταργήσουμε τις σελίδες που χρησιμοποιούνται γι αυτό το σκοπό.

Μπορούμε όμως να ακολουθήσουμε μια προσέγγιση, περισσότερη προσαρμοσμένη στο χρήστη. Ο κάθε χρήστης μπορεί να βλέπει τις εγγραφές της βάσης χωρίς να τις τροποποιεί και παράλληλα να προσθέτει δικές του εγγραφές πάνω στις οποίες έχει κάθε δικαίωμα. Επιλέγει αν αυτές οι εγγραφές είναι ορατές στους άλλους χρήστες της εφαρμογής και τι δικαιώματα έχουν πάνω σε αυτές (προβολή,προσθήκη εικόνων, τροποποίηση, διαγραφή).

Αυτή η προσέγγιση απαιτεί τροποποίηση της βάσης δεδομένων. Μαζί με κάθε εγγραφή πρέπει να αποθηκεύεται και ο κάτοχός της, όπως και τα δικαιώματα άλλων χρηστών πάνω σε αυτήν. Θα χρειαζόταν επίσης να κρατούνται κάποια στοιχεία για τους χρήστες στη βάση δεδομένων. Θα μπορούσαν να δημιουργηθούν ομάδες δικαιωμάτων που θα διαχώριζαν τους χρήστες ανάλογα με τα δικαιώματά τους πάνω στη βάση ή σε ξεχωριστές εγγραφές.

Θα έπρεπε να δημιουργηθεί μια υπηρεσία ταυτοποίησης χρηστών (π.χ. με username και password). Οι μη εγγεγραμένοι χρήστες του site δε θα είχαν δικαίωμα τροποποίησης, αλλά μόνο προβολής των δεδομένων.

ΒΙΒΛΙΟΓΡΑΦΙΑ

[Στα08] Γ. Ι. Στασινόπουλος, Σημειώσεις για το μάθημα "Διαδίκτυο και εφαρμογές" της σχολής ΗΜΜΥ του ΕΜΠ
http://courses.cn.ntua.gr/mod/resource/view.php?id=87

[Sta] George I. Stassinopoulos, Identifying and Searching Ottoman Cancellations

[NiVa02] Nikhil Kothari, Vandana Datje, Developing Microsoft ASP.NET Server Controls and Components

[DuMa02] G. Andrew Duthie, Matthew MacDonald, ASP.NET in a nutshell : a desktop quick reference

[DeDe06] Harvey M. Deitel, Paul J. Deitel, C# for programmers, 2nd ed.

[Mac06] Matthew MacDonald, Beginning ASP.NET 2.0 in C Sharp From Novice to Professional

ΠΑΡΑΡΤΗΜΑ Α : πλήρης κώδικας εφαρμογής

Α1) αρχείο cwHelp.aspx :

```
<%@ Page Language="C#" AutoEventWireup="true" CodeFile="cwHelp.aspx.cs"
Inherits="cwHelp" %>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml" >
<head runat="server">
   <title>Coles Walker Help</title>
</head>
<body style="background-color:Gray">
   <form id="form1" runat="server">
   <div>

   <table style="background-color:Gray" width="100%">
   <tr>
      <td style="width:17%">
         <asp:Button ID="Button1" runat="server" Text="Negative" ToolTip="Negative" BackColor="gray"
                  BorderStyle="none" Height="100%" Width="100%" OnClick="tabButton_Click" />
      </td>
      <td style="width:17%">
         <asp:Button ID="Button2" runat="server" Text="Variable Geometry, Incomplete/Absent Date"
          ToolTip="Variable Geometry, Incomplete/Absent Date" Height="100%" Width="100%"
          OnClick="tabButton_Click" />
      </td>
      <td style="width:17%">
         <asp:Button ID="Button3" runat="server" Text="Circular Date Stamp" ToolTip="Circular Date
          Stamp" Height="100%" Width="100%" OnClick="tabButton_Click" />
      </td>
      <td style="width:17%">
         <asp:Button ID="Button4" runat="server" Text="Octangular Date Stamp" ToolTip="Octangular Date
          Stamp" Height="100%" Width="100%" OnClick="tabButton_Click" />
      </td>
      <td style="width:16%">
         <asp:Button ID="Button5" runat="server" Text="Dots Lattice" ToolTip="Dots Lattice"
          Height="100%" Width="100%" OnClick="tabButton_Click" />
      </td>
      <td style="width:16%">
         <asp:Button ID="Button6" runat="server" Text="Manuscript" ToolTip="Manuscript" Height="100%"
          Width="100%" OnClick="tabButton_Click" />
      </td>
   </tr>
   <tr>
      <td colspan="6">
         <br />
      </td>
   </tr>
   <tr>
      <td colspan="6">
         <asp:MultiView ID="MultiView1" runat="server" ActiveViewIndex="0">
            <asp:View ID="View1" runat="server" OnActivate="view1_Activate">
               <asp:ListBox ID="negativeListBox" AutoPostBack="true" runat="server" ForeColor="white"
                     BackColor="black" Rows="10"
                     OnSelectedIndexChanged="negativeListBox_ItemSelected">
```

```
        <asp:ListItem>Circular/Oval 'an canib' inscription</asp:ListItem>
        <asp:ListItem>Circular, crescent, 'posta hanesi'</asp:ListItem>
        <asp:ListItem>Circular, no crescent, 'posta hanesi'</asp:ListItem>
        <asp:ListItem>Circular, 'telegraf ve posta hanesi'</asp:ListItem>
        <asp:ListItem>Circular, 'posta subesi'</asp:ListItem>
        <asp:ListItem>Bilingual</asp:ListItem>
        <asp:ListItem>Bilingual, annular inscription</asp:ListItem>
        <asp:ListItem>All-Arabic, annular inscription</asp:ListItem>
        <asp:ListItem>Bilingual, annular inscription, misc</asp:ListItem>
        <asp:ListItem>Miscellaneous shapes and inscriptions</asp:ListItem>
    </asp:ListBox>
    <asp:Image ID="negativeImage" runat="server" />
</asp:View>
<asp:View ID="View2" runat="server" OnActivate="view2_Activate">
    <table>
    <tr>
    <td>
        <br />
        <asp:ListBox ID="variableLB" Rows="8" runat="server"
                     OnSelectedIndexChanged="geometry_Updated">
            <asp:ListItem>Square</asp:ListItem>
            <asp:ListItem>Circular</asp:ListItem>
            <asp:ListItem>Rectangular</asp:ListItem>
            <asp:ListItem>Oval</asp:ListItem>
            <asp:ListItem>Hexagonal</asp:ListItem>
            <asp:ListItem>Octagonal</asp:ListItem>
            <asp:ListItem>Triangular</asp:ListItem>
            <asp:ListItem>Miscellaneous</asp:ListItem>
        </asp:ListBox>
        <br />
    </td>
    <td>
        <asp:RadioButtonList ID="variableRBL" runat="server"
                     OnSelectedIndexChanged="geometry_Updated">
            <asp:ListItem>with</asp:ListItem>
            <asp:ListItem Selected="True">without</asp:ListItem>
        </asp:RadioButtonList>
    </td>
    <td>
        Grids and Grilles and
    </td>
    <td>
        <asp:DropDownList ID="variableDDL" runat="server"
                     OnSelectedIndexChanged="geometry_Updated">
            <asp:ListItem>?</asp:ListItem>
            <asp:ListItem>All-Arabic, Unframed</asp:ListItem>
            <asp:ListItem>All-Arabic, Framed, Single Line</asp:ListItem>
            <asp:ListItem>All-Arabic, Framed, Double Lines</asp:ListItem>
            <asp:ListItem>All-Arabic, Framed, Triple Lines</asp:ListItem>
            <asp:ListItem>All-Arabic, Framed, Quad Lines</asp:ListItem>
            <asp:ListItem>All-French, Unframed</asp:ListItem>
            <asp:ListItem>All-French, Framed</asp:ListItem>
            <asp:ListItem>All-Italian, Inner, Outer Inscription</asp:ListItem>
            <asp:ListItem>Bilingual, Unframed</asp:ListItem>
            <asp:ListItem>Bilingual, Framed, Single Line</asp:ListItem>
            <asp:ListItem>Bilingual, Framed, Double Lines</asp:ListItem>
            <asp:ListItem>Bilingual, Miscellaneous Frames</asp:ListItem>
            <asp:ListItem>Trilingual, Framed, Single Line</asp:ListItem>
        </asp:DropDownList>
    </td>
```

```
    <td>
      <asp:Button ID="variableButton" runat="server" Text="submit combination"
          OnClick="geometry_Updated" />
      <br /><asp:Label ID="variableLabel" runat="server" ForeColor="red" Visible="false"
                  Text=""></asp:Label>
    </td>
    </tr>
    </table>
</asp:View>
<asp:View ID="View3" runat="server" OnActivate="view3_Activate">
    <table>
    <tr>
    <td>
      <br />
      <asp:ListBox ID="circularListBox" runat="server" AutoPostBack="True"
                  OnSelectedIndexChanged="circular_SelectedChanged" Rows="7">
        <asp:ListItem>All-Arabic</asp:ListItem>
        <asp:ListItem>All-Arabic, date in belt</asp:ListItem>
        <asp:ListItem>All French</asp:ListItem>
        <asp:ListItem>All French, date in belt</asp:ListItem>
        <asp:ListItem>Bilingual</asp:ListItem>
        <asp:ListItem>Bilingual, only back stamp</asp:ListItem>
        <asp:ListItem>Bilingual, date in six lines</asp:ListItem>
      </asp:ListBox>
      <br /><br />
    </td>
    <td>
      <asp:RadioButtonList ID="circularRadioButtonList" runat="server" AutoPostBack="True"
                  OnSelectedIndexChanged="circularRBL_SelectedChanged">
        <asp:ListItem>no belt</asp:ListItem>
        <asp:ListItem>date in belt</asp:ListItem>
      </asp:RadioButtonList>
    </td>
    <td>
      <asp:ListBox ID="dateBeltListBox" runat="server" AutoPostBack="True"
                  OnSelectedIndexChanged="dateBelt_SelectedChanged" Rows="5">
        <asp:ListItem>Diameter &gt;= 30mm</asp:ListItem>
        <asp:ListItem>Diameter &lt; 30mm</asp:ListItem>
        <asp:ListItem>Vertical Bars, no numerals</asp:ListItem>
        <asp:ListItem>Numerals In Segment</asp:ListItem>
        <asp:ListItem>Numerals In Rim</asp:ListItem>
      </asp:ListBox>
    </td>
    </tr>
    </table>
</asp:View>
<asp:View ID="View4" runat="server" OnActivate="view4_Activate">
    <br /><br /><br />some text
    <asp:ListBox ID="octangularListBox" Rows="2" runat="server" AutoPostBack="True"
          OnSelectedIndexChanged="octangular_SelectedChanged">
      <asp:ListItem>All-Arabic</asp:ListItem>
      <asp:ListItem>Bilingual</asp:ListItem>
    </asp:ListBox>
    <br /><br /><br /><br /><br />
</asp:View>
<asp:View ID="View5" runat="server" OnActivate="view5_Activate">
    <br /><br />
    <asp:CheckBox ID="dmCheckBox" runat="server" AutoPostBack="true" Text="DM?"
          OnCheckedChanged="DM_CheckedChanged" />
    <br /><br /><br /><br /><br /><br />
```

```
            </asp:View>
            <asp:View ID="View6" runat="server" OnActivate="view6_Activate">
              <br /><br /><br /><br /><br /><br /><br /><br />
            </asp:View>
          </asp:MultiView>
        </td>
      </tr>
      <tr>
        <td colspan="6">
          <br />
        </td>
      </tr>
      <tr>
        <td align="right">
          <asp:Label ID="brLabel" runat="server" BackColor="green" Text="apply brandt type"></asp:Label>
        </td>
        <td align="left" >
          <asp:Button ID="brButton" runat="server" Text="*" Width="100%" OnClick="cwButton_Click"
            ToolTip="press to submit current filtering" />
        </td>
        <td align="right">
          <asp:Label ID="cwLabel" runat="server" BackColor="yellow" Text="apply coles and walker
                  code"></asp:Label>
        </td>
        <td align="left" >
          <asp:Button ID="cwButton" runat="server" Text="*" Width="100%" OnClick="cwButton_Click"
            ToolTip="press to submit current filtering" />
        </td>
        <td>
        </td>
        <td>
          <asp:Button ID="backButton" runat="server" Text="Back" PostBackUrl="~/Default.aspx"
            ToolTip="return to cancellations" />
        </td>
      </tr>
      </table>

      </div>
      </form>
</body>
</html>
```

A2) αρχείο cwHelp.aspx.cs :

```
using System;
using System.Data;
using System.Configuration;
using System.Collections;
using System.Web;
using System.Web.Security;
using System.Web.UI;
using System.Web.UI.WebControls;
using System.Web.UI.WebControls.WebParts;
using System.Web.UI.HtmlControls;

public partial class cwHelp : System.Web.UI.Page
{
   private static string imgPath = "Images/CWBrEx/";
```

```
protected void Page_Load(object sender, EventArgs e)
{   }

protected void tabButton_Click(object sender, EventArgs e)
{
   Button1.BackColor = System.Drawing.Color.Empty;
   Button2.BackColor = System.Drawing.Color.Empty;
   Button3.BackColor = System.Drawing.Color.Empty;
   Button4.BackColor = System.Drawing.Color.Empty;
   Button5.BackColor = System.Drawing.Color.Empty;
   Button6.BackColor = System.Drawing.Color.Empty;
   Button1.BorderStyle = BorderStyle.NotSet;
   Button2.BorderStyle = BorderStyle.NotSet;
   Button3.BorderStyle = BorderStyle.NotSet;
   Button4.BorderStyle = BorderStyle.NotSet;
   Button5.BorderStyle = BorderStyle.NotSet;
   Button6.BorderStyle = BorderStyle.NotSet;

   Button but = (Button)sender;
   but.BackColor = System.Drawing.Color.Gray;
   but.BorderStyle = BorderStyle.None;

   switch(but.ID)
   {
      case "Button1":
         MultiView1.ActiveViewIndex = 0;
         break;
      case "Button2":
         MultiView1.ActiveViewIndex = 1;
         break;
      case "Button3":
         MultiView1.ActiveViewIndex = 2;
         break;
      case "Button4":
         MultiView1.ActiveViewIndex = 3;
         break;
      case "Button5":
         MultiView1.ActiveViewIndex = 4;
         break;
      case "Button6":
         MultiView1.ActiveViewIndex = 5;
         break;
   }
}

protected void negativeListBox_ItemSelected(object sender, EventArgs e)
{
   switch (negativeListBox.SelectedIndex)
   {
      case 0:
         negativeImage.ImageUrl = imgPath + "S1_h.bmp";
         cwButton.Text = "S1";
         break;
      case 1:
         negativeImage.ImageUrl = imgPath + "S2_h.bmp";
         cwButton.Text = "S2";
         break;
      case 2:
         negativeImage.ImageUrl = imgPath + "S2A_h.bmp";
         cwButton.Text = "S2A";
```

```
            break;
        case 3:
            negativeImage.ImageUrl = imgPath + "S3_h.bmp";
            cwButton.Text = "S3";
            break;
        case 4:
            negativeImage.ImageUrl = imgPath + "S4_h.bmp";
            cwButton.Text = "S4";
            break;
        case 5:
            negativeImage.ImageUrl = imgPath + "SB1_h.bmp";
            cwButton.Text = "SB1";
            break;
        case 6:
            negativeImage.ImageUrl = imgPath + "SB2_h.bmp";
            cwButton.Text = "SB2";
            break;
        case 7:
            negativeImage.ImageUrl = imgPath + "SM_h.bmp";
            cwButton.Text = "SM2";
            break;
        case 8:
            negativeImage.ImageUrl = imgPath + "SM2_h.bmp";
            cwButton.Text = "SBM2";
            break;
        case 9:
            negativeImage.ImageUrl = imgPath + "SM2_h.bmp";
            cwButton.Text = "SM";
            break;
        default:
            break;
    }
}

protected void view6_Activate(object sender, EventArgs e)
{
    cwButton.Text = "Man.";
}
protected void view5_Activate(object sender, EventArgs e)
{
    cwButton.Text = "D";
}
protected void view4_Activate(object sender, EventArgs e)
{
    cwButton.Text = "*";
}
protected void view1_Activate(object sender, EventArgs e)
{
    cwButton.Text = "*";
}
protected void view3_Activate(object sender, EventArgs e)
{
    cwButton.Text = "*";
    dateBeltListBox.Visible = false;
    circularRadioButtonList.Visible = false;
}
protected void view2_Activate(object sender, EventArgs e)
{
    cwButton.Text = "*";
    variableLabel.Visible = false;
```

```
}

protected void DM_CheckedChanged(object sender, EventArgs e)
{
   if (dmCheckBox.Checked)
      cwButton.Text = "DM";
   else
      cwButton.Text = "D";
}

protected void octangular_SelectedChanged(object sender, EventArgs e)
{
   if ( octangularListBox.SelectedIndex == 0 )
      cwButton.Text = "AO*";
   else
      cwButton.Text = "BO*";
}

protected void circular_SelectedChanged(object sender, EventArgs e)
{
   switch (circularListBox.SelectedIndex)
   {
      case 0:
         cwButton.Text = "CA";
         circularRadioButtonList.Visible = false;
         dateBeltListBox.Visible = false;
         break;
      case 1:
         cwButton.Text = "C6/CA";
         circularRadioButtonList.Visible = false;
         dateBeltListBox.Visible = false;
         break;
      case 2:
         cwButton.Text = "CF";
         circularRadioButtonList.Visible = false;
         dateBeltListBox.Visible = false;
         break;
      case 3:
         cwButton.Text = "CF/C5";
         circularRadioButtonList.Visible = false;
         dateBeltListBox.Visible = false;
         break;
      case 4:
         cwButton.Text = "C2";
         circularRadioButtonList.Visible = true;
         dateBeltListBox.Visible = false;
         circularRadioButtonList.SelectedIndex = 0;
         break;
      case 5:
         cwButton.Text = "C1";
         circularRadioButtonList.Visible = false;
         dateBeltListBox.Visible = false;
         break;
      case 6:
         cwButton.Text = "C2";
         circularRadioButtonList.Visible = false;
         dateBeltListBox.Visible = false;
         break;
   }
}
```

```
protected void dateBelt_SelectedChanged(object sender, EventArgs e)
{
   switch (dateBeltListBox.SelectedIndex)
   {
      case 0:
         cwButton.Text = "C3";
         break;
      case 1:
         cwButton.Text = "C4";
         break;
      case 2:
         cwButton.Text = "C5";
         break;
      case 3:
         cwButton.Text = "C6";
         break;
      case 4:
         cwButton.Text = "C6R";
         break;
   }
}
protected void circularRBL_SelectedChanged(object sender, EventArgs e)
{
   if (circularRadioButtonList.SelectedIndex == 0)
   {
      cwButton.Text = "C2";
      dateBeltListBox.Visible = false;
   }
   else
   {
      cwButton.Text = "C?";
      dateBeltListBox.Visible = true;
   }
}

protected void geometry_Updated(object sender, EventArgs e)
{
   variableLabel.Visible = false;

   string geoPrefix;
   if (variableLB.SelectedValue.Equals(""))
      geoPrefix = "?";
   else if (variableLB.SelectedValue.Equals("Oval"))
      geoPrefix = "V";
   else
      geoPrefix = variableLB.SelectedValue.Substring(0, 1);

   string hcaption="*";

   switch(variableDDL.SelectedValue)
   {
      case "?":
         hcaption = "?";
         break;
      case "All-Arabic, Unframed":
         hcaption = "AX";
         if ( geoPrefix.Equals("M") )
            hcaption = "AM";
         break;
      case "All-Arabic, Framed, Single Line":
```

```
            hcaption = "A" + geoPrefix + "1";
            break;
        case "All-Arabic, Framed, Double Lines":
            hcaption = "A" + geoPrefix + "2";
            break;
        case "All-Arabic, Framed, Triple Lines":
            hcaption = "A" + geoPrefix + "3";
            break;
        case "All-Arabic, Framed, Quad Lines":
            hcaption = "A" + geoPrefix + "4";
            break;
        case "All-French, Unframed":
            hcaption = "F1";
            break;
        case "All-French, Framed":
             hcaption = "F2";
            break;
        case "All-Italian, Inner, Outer Inscription":
            if (geoPrefix.Equals("C"))
                hcaption = "M";
            break;
        case "Bilingual, Unframed":
            hcaption = "BX";
            break;
        case "Bilingual, Framed, Single Line":
            hcaption = "B" + geoPrefix + "1";
            break;
        case "Bilingual, Framed, Double Lines":
            hcaption = "B" + geoPrefix + "2";
            break;
        case "Bilingual, Miscellaneous Frames":
            hcaption = "B" + geoPrefix + "M";
            if (geoPrefix.Equals("M"))
                hcaption = "BM";
            break;
        case "Trilingual, Framed, Single Line":
            hcaption = "T" + geoPrefix + "1";
            break;
        default:
            break;
    }

    if ( variableRBL.SelectedValue.Equals("with") )
        switch(hcaption)
        {
            case "AR1":
                hcaption = "GA";
                break;
            case "AX":
                hcaption = "GA";
                break;
            case "F1":
                hcaption = "GF";
                break;
            case "AV1":
                hcaption = "GAV1";
                break;
            case "BX":
                hcaption = "GB";
                break;
```

```
            case "?":
               hcaption = "*";
               break;
            default:
               hcaption = "*";
               variableLabel.Text = "This combination with Grids and Grilles" +
                  " does not exist.\nPlease choose the appropriate case.";
               variableLabel.Visible = true;
               break;
         }

      cwButton.Text = hcaption;
   }

   protected void cwButton_Click(object sender, EventArgs e)
   {
      string temp = cwButton.Text;
      if (temp.Equals("*"))
      {
         Session["filterExpr"] = "null";
         Session["filterText"] = "";
         Response.Redirect("Default.aspx");
      }

      temp = temp.Replace('*','%');
      temp = temp.Replace('?','%');      //den doulevoun ?,_

      if (temp.Contains("/"))    //dyo kodikoi pou xorizontai apo /
      {
         int i = temp.IndexOf('/');
         string temp1 = temp.Substring(0, i);
         temp = temp.Substring(i + 1);
         Session["filterExpr"] = "(CW LIKE '" + temp1 +
                     "') OR (CW LIKE '" + temp + "')";
         Session["filterText"] = ", having CW code " +
            temp1 + " or " + temp;
      }
      else
      {
         Session["filterExpr"] = "CW LIKE '" + temp + "'";
         Session["filterText"] = ", having CW code " + temp;
      }

      Response.Redirect("Default.aspx");
   }
}
```

A3) αρχείο Default.aspx :

```
<%@ Page Language="C#" ValidateRequest="false" Trace="false"
AutoEventWireup="true"  CodeFile="Default.aspx.cs" Inherits="_Default" %>
<%@ Register TagPrefix="uc" TagName="ImageContainer" Src="~/ImageContainer.ascx" %>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml" >
<head runat="server">
   <title>ottoman cancellations</title>
```

```
    <style type="text/css">
        td.cellLeft {border-bottom: black dashed thin;
                     border-top: black dashed thin;
                     border-left: black dashed thin;
                     border-right: none;
                    }
        td.cellRight{border-bottom: black dashed thin;
                     border-top: black dashed thin;
                     border-left: none;
                     border-right: black dashed thin;
                    }
            td.cellLR{border-bottom: black dashed thin;
                     border-top: black dashed thin;
                     border-left: black dashed thin;
                     border-right: black dashed thin;
                    }
    </style>
</head>
<body style="background-color:GrayText">
    <form id="form1" runat="server">
    <div>

    <table id="menuTable" runat="server" style="background-color:Gray" width="100%" >
    <tr>
    <td colspan="3">
        <asp:Label ID="titleLabel1" Font-Bold="true" ForeColor="Red" runat="server"></asp:Label>
    </td>
    <td colspan="2">
        <asp:TextBox ID="dateTextBox" runat="server" ToolTip="give year range (eg.
1897-1903)"></asp:TextBox>
        <asp:Button ID="dateButton" runat="server" Text="search by dates" ToolTip="give year range (eg.
1897-1903)" OnClick="dateButton_Click" />
        <asp:Label ID="dateLabel" runat="server" ForeColor="DarkRed" Visible="false"></asp:Label>
    </td>
    <td>
    </td>
    </tr>
    <tr>
    <td colspan="3">
        <asp:Label ID="titleLabel2" Font-Bold="true" ForeColor="Blue" runat="server" ></asp:Label>
    </td>
    <td colspan="2">
        <asp:TextBox ID="idTextBox" runat="server" ToolTip="give cancellation id (0 for all
cancellations)"></asp:TextBox>
        <asp:Button ID="idButton" runat="server" Text="search by ID" ToolTip="give cancellation id (0 for all
cancellations)" OnClick="idButton_Clicked" />
        <asp:Label ID="idLabel" runat="server" ForeColor="DarkRed" Visible="false"></asp:Label>
    </td>
    <td>
        <asp:Button ID="imgButton" ForeColor="Red" PostBackUrl="~/Thumbnails.aspx" runat="server"
Text="show images" ToolTip="list all (available) images for cancellations listed below" />
    </td>
    </tr>
    <tr>
    <td class="cellLeft" align="right" >
        <br />
        <asp:Label ID="ottomanLabel" ForeColor="Red" runat="server" Text="Ottoman"></asp:Label><br />
        <asp:Label ID="frenchLabel" ForeColor="Blue" runat="server" Text="French"></asp:Label>
    </td>
    <td class="cellRight"  >
```

```
      <asp:Button ID="keyboardButton" ForeColor="Red" runat="server" Text="keyboard" ToolTip="click
for keyboard to fill in ottoman text" OnClick="keyboardButton_Click" />
      <asp:Button ID="searchButton" runat="server" Text="search" ToolTip="enter ottoman AND french text,
then Search" OnClick="searchButton_Click"/><br />
      <asp:TextBox ID="ottomanTextBox" BackColor="Red" runat="server" Text="%" ToolTip="use
keyboard above, for empty field put '=""></asp:TextBox><br />
      <asp:TextBox ID="frenchTextBox" BackColor="Blue" runat="server" Text="%" ToolTip="use western
keyboard, _ means ANY CHARACTER, % means ANY SEQUENCE OF CHARACTERS"></asp:TextBox>
   </td>
   <td  class="cellLeft" align="center" >
      <asp:RadioButtonList ID="RadioButtonList1" runat="server" AutoPostBack="True"
OnSelectedIndexChanged="languageChanged">
         <asp:ListItem Selected="True">latinised</asp:ListItem>
         <asp:ListItem>multilingual</asp:ListItem>
      </asp:RadioButtonList>
      <br />
      <asp:Label ID="vilaetLabel" ForeColor="Red" runat="server" Text="vilaet"></asp:Label>
      <asp:DropDownList ID="vilaetDDL" runat="server" AutoPostBack="True"
DataSourceID="vilaetDataSource" DataTextField="VilaetL" DataValueField="VilaetO"
OnSelectedIndexChanged="vilaetChosen" ToolTip="choose vilaet.sanjak,etc...">
      </asp:DropDownList>
      <asp:AccessDataSource ID="vilaetDataSource" runat="server"
DataFile="~/App_Data/Cancel_Alexandros_Apr09.mdb" SelectCommand="SELECT * FROM
[RegionOttoman] ORDER BY [VilaetL]"></asp:AccessDataSource>
      <br />
      <asp:Label ID="countryLabel" ForeColor="Blue" runat="server" Text="country"></asp:Label>
      <asp:DropDownList ID="countryDDL" runat="server" AutoPostBack="True"
DataSourceID="countryDataSource" DataTextField="Country" DataValueField="CountryMulti"
OnSelectedIndexChanged="countryChosen" ToolTip="choose a country">
      </asp:DropDownList>
      <asp:AccessDataSource ID="countryDataSource" runat="server"
DataFile="~/App_Data/Cancel_Alexandros_Apr09.mdb" SelectCommand="SELECT * FROM [Countries]
ORDER BY [Country]"></asp:AccessDataSource>
   </td>
   <td class="cellRight" >
      <asp:Button ID="editButton" runat="server" Text="(Edit) Post Offices in" OnClick="editButton_Click"
ToolTip="create new/update/delete post offices in selected country/region" />
      <br />
      <asp:Label ID="chosenLabel" runat="server" Text="no vilaet/country chosen"
ForeColor="Blue"></asp:Label>
      <br /><br />
      <asp:DropDownList ID="postOfficeDDL" runat="server" AutoPostBack="True"
DataSourceID="localNamesDataSource" DataTextField="POTurk" DataValueField="POTurk"
OnSelectedIndexChanged="postOfficeChosen" ToolTip="search based on post office"
OnDataBound="postOfficeDDL_DataBound">
      </asp:DropDownList>
      <asp:AccessDataSource ID="localNamesDataSource" runat="server"
DataFile="~/App_Data/Cancel_Alexandros_Apr09.mdb" SelectCommand="SELECT [POTurk], [POOtto]
FROM [Local Names] WHERE ([POTurk] IS NOT NULL) ORDER BY
[POTurk]"></asp:AccessDataSource>
   </td>
   <td class="cellLR" style="color:Green;" align="center">
      <asp:Label ID="cwBrandtLabel" runat="server" Text="list only code/type"></asp:Label>
      <br />
      <asp:Label ID="cwLabel" runat="server" Text="CW"></asp:Label>
          <asp:DropDownList ID="cwDDL" runat="server" AutoPostBack="True"
DataSourceID="cwDataSource" DataTextField="column1" DataValueField="column1"
OnSelectedIndexChanged="cwChosen" ToolTip="filtering cancellations according to code/type"
OnDataBound="cwDDL_DataBound">
      </asp:DropDownList>
```

```
        <asp:AccessDataSource ID="cwDataSource" runat="server"
DataFile="~/App_Data/Cancel_Alexandros_Apr09.mdb"
            SelectCommand="SELECT [C&W Codes] AS column1 FROM [CWCodes] ORDER BY [C&W
Codes]">
        </asp:AccessDataSource>
        <br />
        <asp:Label ID="brandLabel" runat="server" Text="Brandt"></asp:Label>
        <asp:DropDownList ID="brandtDDL" runat="server" AutoPostBack="True"
DataSourceID="brandtDataSource" DataTextField="Bradt_Types" DataValueField="Bradt_Types"
OnSelectedIndexChanged="brandtChosen" ToolTip="filtering cancellations according to code/type"
OnDataBound="brandtDDL_DataBound">
        </asp:DropDownList>
        <asp:AccessDataSource ID="brandtDataSource" runat="server"
DataFile="~/App_Data/Cancel_Alexandros_Apr09.mdb"
            SelectCommand="SELECT [Bradt Types] AS Bradt_Types FROM [BrandtTypes] ORDER BY [Bradt
Types]"></asp:AccessDataSource>
        <br />
        <asp:Button ID="cwHelpButton" ForeColor="Green" runat="server" Text="Coles Walker Help"
ToolTip="help for selecting the correct CW code" PostBackUrl="~/cwHelp.aspx" />
    </td>
    <td class="cellLR" style="background-color:Black;color:White;">
        <asp:Label ID="negativeLabel" runat="server" Text="NegativeCancellations"></asp:Label>
        <br />
        <asp:Button ID="negativeButton" runat="server" Text="All Negatives"
OnClick="negativeButton_Click" ToolTip="lists all negative cancellations" />
        <br />
        <asp:Button ID="shapeButton" ForeColor="Red" runat="server" Text="shape search" ToolTip="search
based on shape" PostBackUrl="~/shapeSearch.aspx" />
        <br />
        <asp:Button ID="expressionButton" ForeColor="Red" runat="server" Text="expression search"
ToolTip="search based on expression" OnClick="expressionButton_Click" />
    </td>
    </tr>
    <tr>
    <td colspan="6">
        <asp:Table ID="keyboardTable" runat="server" Visible="false" BackColor="Pink">
    <asp:TableRow>
        <asp:TableCell ColumnSpan="8">
            <asp:Label ID="keyboardTitleLabel" runat="server" Text="Ottoman Keyboard"
ForeColor="Red"></asp:Label>
        </asp:TableCell>
    </asp:TableRow>
    <asp:TableRow>
        <asp:TableCell>
            <asp:Button ID="keyButton1" runat="server" OnClick="keyButton_Click" BackColor="DarkGray" />
        </asp:TableCell>
        <asp:TableCell>
            <asp:Button ID="keyButton2" runat="server" OnClick="keyButton_Click" BackColor="DarkGray" />
        </asp:TableCell>
        <asp:TableCell>
            <asp:Button ID="keyButton3" runat="server" OnClick="keyButton_Click" BackColor="DarkGray" />
        </asp:TableCell>
        <asp:TableCell>
            <asp:Button ID="keyButton4" runat="server" OnClick="keyButton_Click" BackColor="DarkGray" />
        </asp:TableCell>
        <asp:TableCell>
            <asp:Button ID="keyButton5" runat="server" OnClick="keyButton_Click" BackColor="DarkGray" />
        </asp:TableCell>
        <asp:TableCell>
            <asp:Button ID="keyButton6" runat="server" OnClick="keyButton_Click" BackColor="DarkGray" />
```

```
        </asp:TableCell>
        <asp:TableCell>
          <asp:Button ID="keyButton7" runat="server" OnClick="keyButton_Click" BackColor="DarkGray" />
        </asp:TableCell>
        <asp:TableCell>
          <asp:Button ID="keyButton8" runat="server" OnClick="keyButton_Click" BackColor="DarkGray" />
        </asp:TableCell>
        <asp:TableCell>
          <asp:Button ID="keyButton9" runat="server" OnClick="keyButton_Click" BackColor="DarkGray" />
        </asp:TableCell>
        <asp:TableCell>
          <asp:Button ID="keyButton10" runat="server" OnClick="keyButton_Click" BackColor="DarkGray"
/>
        </asp:TableCell>
        <asp:TableCell>
          <asp:Button ID="keyButton11" runat="server" Text="-" OnClick="keyButton_Click"
BackColor="DarkGray" />
        </asp:TableCell>
        <asp:TableCell>
          <asp:Button ID="keyButton12" runat="server" Text="=" OnClick="keyButton_Click"
BackColor="DarkGray" />
        </asp:TableCell>
        <asp:TableCell>
        </asp:TableCell>
        <asp:TableCell>
          <asp:Button ID="keyButton13" runat="server" Text="bksp" OnClick="keyButton_Click"
BackColor="DarkGray" />
        </asp:TableCell>
      </asp:TableRow>
      <asp:TableRow>
        <asp:TableCell>
          <asp:Button ID="keyButton14" runat="server" OnClick="keyButton_Click" BackColor="DarkGray" />
        </asp:TableCell>
        <asp:TableCell>
          <asp:Button ID="keyButton15" runat="server" OnClick="keyButton_Click" BackColor="DarkGray" />
        </asp:TableCell>
        <asp:TableCell>
          <asp:Button ID="keyButton16" runat="server" OnClick="keyButton_Click" BackColor="DarkGray" />
        </asp:TableCell>
        <asp:TableCell>
          <asp:Button ID="keyButton17" runat="server" OnClick="keyButton_Click" BackColor="DarkGray" />
        </asp:TableCell>
        <asp:TableCell>
          <asp:Button ID="keyButton18" runat="server" OnClick="keyButton_Click" BackColor="DarkGray" />
        </asp:TableCell>
        <asp:TableCell>
          <asp:Button ID="keyButton19" runat="server" OnClick="keyButton_Click" BackColor="DarkGray" />
        </asp:TableCell>
        <asp:TableCell>
          <asp:Button ID="keyButton20" runat="server" OnClick="keyButton_Click" BackColor="DarkGray" />
        </asp:TableCell>
        <asp:TableCell>
          <asp:Button ID="keyButton21" runat="server" OnClick="keyButton_Click" BackColor="DarkGray" />
        </asp:TableCell>
        <asp:TableCell>
          <asp:Button ID="keyButton22" runat="server" OnClick="keyButton_Click" BackColor="DarkGray" />
        </asp:TableCell>
        <asp:TableCell>
          <asp:Button ID="keyButton23" runat="server" OnClick="keyButton_Click" BackColor="DarkGray" />
        </asp:TableCell>
```

```
      <asp:TableCell>
        <asp:Button ID="keyButton24" runat="server" OnClick="keyButton_Click" BackColor="DarkGray" />
      </asp:TableCell>
      <asp:TableCell>
        <asp:Button ID="keyButton25" runat="server" OnClick="keyButton_Click" BackColor="DarkGray" />
      </asp:TableCell>
      <asp:TableCell>
        <asp:Button ID="keyButton26" runat="server" OnClick="keyButton_Click" BackColor="DarkGray" />
      </asp:TableCell>
    </asp:TableRow>
    <asp:TableRow>
      <asp:TableCell>
        <asp:Button ID="keyButton27" runat="server" OnClick="keyButton_Click" BackColor="DarkGray" />
      </asp:TableCell>
      <asp:TableCell>
        <asp:Button ID="keyButton28" runat="server" OnClick="keyButton_Click" BackColor="DarkGray" />
      </asp:TableCell>
      <asp:TableCell>
        <asp:Button ID="keyButton29" runat="server" OnClick="keyButton_Click" BackColor="DarkGray" />
      </asp:TableCell>
      <asp:TableCell>
        <asp:Button ID="keyButton30" runat="server" OnClick="keyButton_Click" BackColor="DarkGray" />
      </asp:TableCell>
      <asp:TableCell>
        <asp:Button ID="keyButton31" runat="server" OnClick="keyButton_Click" BackColor="DarkGray" />
      </asp:TableCell>
      <asp:TableCell>
        <asp:Button ID="keyButton32" runat="server" OnClick="keyButton_Click" BackColor="DarkGray" />
      </asp:TableCell>
      <asp:TableCell>
        <asp:Button ID="keyButton33" runat="server" OnClick="keyButton_Click" BackColor="DarkGray" />
      </asp:TableCell>
      <asp:TableCell>
        <asp:Button ID="keyButton34" runat="server" OnClick="keyButton_Click" BackColor="DarkGray" />
      </asp:TableCell>
      <asp:TableCell>
        <asp:Button ID="keyButton35" runat="server" OnClick="keyButton_Click" BackColor="DarkGray" />
      </asp:TableCell>
      <asp:TableCell>
        <asp:Button ID="keyButton36" runat="server" OnClick="keyButton_Click" BackColor="DarkGray" />
      </asp:TableCell>
      <asp:TableCell>
        <asp:Button ID="keyButton37" runat="server" OnClick="keyButton_Click" BackColor="DarkGray" />
      </asp:TableCell>
      <asp:TableCell>
        <asp:Button ID="keyButton38" runat="server" ForeColor="Red" OnClick="keyButton_Click"
BackColor="DarkGray" />
      </asp:TableCell>
      <asp:TableCell>
        <asp:Button ID="keyButton39" runat="server" ForeColor="Red" OnClick="keyButton_Click"
BackColor="DarkGray" />
      </asp:TableCell>
      <asp:TableCell>
        <asp:Button ID="keyButton40" runat="server" OnClick="keyButton_Click" Text="NEWLINE"
BackColor="DarkGray" />
      </asp:TableCell>
    </asp:TableRow>
    <asp:TableRow>
      <asp:TableCell>
        <asp:Button ID="keyButton41" runat="server" OnClick="keyButton_Click" BackColor="DarkGray" />
```

```
      </asp:TableCell>
      <asp:TableCell>
        <asp:Button ID="keyButton42" runat="server" OnClick="keyButton_Click" BackColor="DarkGray" />
      </asp:TableCell>
      <asp:TableCell>
        <asp:Button ID="keyButton43" runat="server" OnClick="keyButton_Click" BackColor="DarkGray" />
      </asp:TableCell>
      <asp:TableCell>
        <asp:Button ID="keyButton44" runat="server" OnClick="keyButton_Click" BackColor="DarkGray" />
      </asp:TableCell>
      <asp:TableCell>
        <asp:Button ID="keyButton45" runat="server" OnClick="keyButton_Click" BackColor="DarkGray" />
      </asp:TableCell>
      <asp:TableCell>
        <asp:Button ID="keyButton46" runat="server" OnClick="keyButton_Click" BackColor="DarkGray" />
      </asp:TableCell>
      <asp:TableCell>
        <asp:Button ID="keyButton47" runat="server" OnClick="keyButton_Click" BackColor="DarkGray" />
      </asp:TableCell>
      <asp:TableCell>
        <asp:Button ID="keyButton48" runat="server" OnClick="keyButton_Click" BackColor="DarkGray" />
      </asp:TableCell>
      <asp:TableCell>
        <asp:Button ID="keyButton49" runat="server" OnClick="keyButton_Click" BackColor="DarkGray" />
      </asp:TableCell>
      <asp:TableCell>
        <asp:Button ID="keyButton50" runat="server" OnClick="keyButton_Click" BackColor="DarkGray" />
      </asp:TableCell>
      <asp:TableCell>
        <asp:Button ID="keyButton51" runat="server" OnClick="keyButton_Click" BackColor="DarkGray" />
      </asp:TableCell>
      <asp:TableCell>
        <asp:Button ID="keyButton52" runat="server" OnClick="keyButton_Click" BackColor="DarkGray" />
      </asp:TableCell>
      <asp:TableCell>
        <asp:Button ID="keyButton53" runat="server" Text="(" OnClick="keyButton_Click"
BackColor="DarkGray" />
      </asp:TableCell>
      <asp:TableCell>
        <asp:Button ID="keyButton54" runat="server" Text=")" OnClick="keyButton_Click"
BackColor="DarkGray" />
      </asp:TableCell>
    </asp:TableRow>
    <asp:TableRow>
      <asp:TableCell ColumnSpan="5">
      </asp:TableCell>
      <asp:TableCell ColumnSpan="7">
        <asp:Button ID="keySpaceButton" runat="server" Text="Space" BackColor="DarkGray"
OnClick="keyButton_Click" Width="100%" />
      </asp:TableCell>
      <asp:TableCell>
      </asp:TableCell>
      <asp:TableCell>
        <asp:Button ID="keyCloseButton" runat="server" Text="Close" OnClick="keyButton_Click"
BackColor="DarkGray" />
      </asp:TableCell>
    </asp:TableRow>
    <asp:TableRow>
      <asp:TableCell ColumnSpan="3">common terms in:</asp:TableCell>
      <asp:TableCell ColumnSpan="3">
```

```
        <asp:DropDownList ID="keyboardOttoDDL" runat="server" AutoPostBack="True"
DataSourceID="keyboardDataSource" DataTextField="Term_Otto" DataValueField="Term_Otto"
OnDataBound="keyboardDDL_DataBound"
OnSelectedIndexChanged="commonTermChosen"></asp:DropDownList>
        <asp:AccessDataSource ID="keyboardDataSource" runat="server"
DataFile="~/App_Data/Cancel_Alexandros_Apr09.mdb"
          SelectCommand="SELECT [Term_Otto], [Term_Turk], [Term_Meaning] FROM [Terms] ORDER
BY [Term_Turk]">
        </asp:AccessDataSource>
      </asp:TableCell>
      <asp:TableCell ColumnSpan="4">
        <asp:DropDownList ID="keyboardTurkDDL" runat="server" AutoPostBack="True"
DataSourceID="keyboardDataSource" DataTextField="Term_Turk" DataValueField="Term_Turk"
OnDataBound="keyboardDDL_DataBound" OnSelectedIndexChanged="commonTermChosen">
        </asp:DropDownList>
      </asp:TableCell>
      <asp:TableCell ColumnSpan="4">
        <asp:DropDownList ID="keyboardEnglDDL" runat="server" AutoPostBack="True"
DataSourceID="keyboardDataSource" DataTextField="Term_Meaning" DataValueField="Term_Meaning"
OnDataBound="keyboardDDL_DataBound" OnSelectedIndexChanged="commonTermChosen">
        </asp:DropDownList>
      </asp:TableCell>
    </asp:TableRow>
      </asp:Table>
    </td>
    </tr>
    <tr>
      <td colspan="6">
      <asp:TextBox ID="warningTextBox" TextMode="MultiLine" ReadOnly="true" Visible="false"
ForeColor="red" runat="server" Width="100%"></asp:TextBox>
      </td>
    </tr>
    <tr>
      <td><br /></td>
      <td><br /></td>
      <td><br /></td>
      <td><br /></td>
      <td><br /></td>
      <td><br /></td>
    </tr>
    <tr>
      <td colspan="6" align="center">
        <asp:Label ID="foundLabel1" runat="server"></asp:Label>
      </td>
    </tr>
    </table>


      <asp:ObjectDataSource ID="detailsDataSource" runat="server"
OldValuesParameterFormatString="original_{0}" SelectMethod="GetDataByID"
TypeName="DataSet1TableAdapters.DetailsTableAdapter">
        <SelectParameters>
          <asp:Parameter Name="ID" Type="Int32" />
        </SelectParameters>
      </asp:ObjectDataSource>
    <table id="detailsTable" visible="false" width="100%" runat="server">
    <tr>
      <td style="height: 10px"></td>
      <td style="height: 10px"></td>
      <td align="right" style="height: 10px">
```

```
            <asp:Label ID="detailsIDLabel" runat="server" Text="ID"></asp:Label>
        </td>
        <td style="height: 10px">
            <asp:TextBox ID="detailsIDTextBox" runat="server" ReadOnly="true"></asp:TextBox>
        </td>
        <td style="height: 10px"></td>
        <td align="right" style="height: 10px">
            <asp:Label ID="detailsDatesLabel" runat="server" Text="Dates"></asp:Label>
        </td>
        <td style="height: 10px">
            <asp:TextBox ID="detailsDatesTextBox" runat="server" ReadOnly="true"></asp:TextBox>
        </td>
    </tr>
    <tr>
        <td colspan="2">
            <asp:TextBox ID="detailsOttoFieldTextBox" TextMode="MultiLine" Rows="5" Width="100%"
runat="server" ReadOnly="true"></asp:TextBox>
        </td>
        <td align="left">
            <asp:CheckBox ID="detailsOttoCheckBox" runat="server"/>
         </td>
        <td colspan="2">
            <asp:Label ID="detailsPOLabel" runat="server" Text="Post Office in:"></asp:Label><br/>
            <asp:TextBox ID="detailsPOTextBox" TextMode="MultiLine" Rows="4" Width="100%"
runat="server" ReadOnly="true"></asp:TextBox>
        </td>
        <td rowspan="2" colspan="2" align="center">
            <asp:Literal ID="detailsColorsLiteral" runat="server"></asp:Literal><br/>
            <uc:ImageContainer ID="imageContainer" runat="server" />
        </td>
    </tr>
    <tr>
        <td colspan="2">
            <asp:TextBox ID="detailsFrenchFieldTextBox" TextMode="MultiLine" Rows="5" Width="100%"
runat="server" ReadOnly="true"></asp:TextBox>
        </td>
        <td align="left">
            <asp:CheckBox ID="detailsFrenchCheckBox" runat="server" />
        </td>
        <td colspan="2">
            <asp:Label ID="detailsLocalNamesLabel" runat="server" Text="now in:"></asp:Label><br/>
            <asp:TextBox ID="detailsLocalNamesTextBox" TextMode="MultiLine" Rows="3" Width="100%"
runat="server" ReadOnly="true"></asp:TextBox>
        </td>
    </tr>
    <tr>
        <td align="right">
            <asp:Label ID="detailsCWLabel" runat="server" Text="CW code"></asp:Label>
        </td>
        <td>
            <asp:TextBox ID="detailsCWTextBox" runat="server" ReadOnly="true"></asp:TextBox>
        </td>
        <td></td>
        <td align="right">
            <asp:Label ID="detailsBrLabel" runat="server" Text="Br type"></asp:Label>
        </td>
        <td>
            <asp:TextBox ID="detailsBrTextBox" runat="server" ReadOnly="true"></asp:TextBox>
        </td>
        <td align="right">
```

```
            <asp:Label ID="detailsNuhoglouLabel" runat="server" Text="Nuhoglou page"></asp:Label>
        </td>
        <td>
            <asp:TextBox ID="detailsNuhoglouTextBox" runat="server" ReadOnly="true"></asp:TextBox>

        </td>
    </tr>
    <tr>
        <td align="right">
            <asp:Label ID="detailsCWPageLabel" runat="server" Text="CW page"></asp:Label>
        </td>
        <td>
            <asp:TextBox ID="detailsCWPageTextBox" runat="server" ReadOnly="true"></asp:TextBox>
        </td>
        <td></td>
        <td align="right">
            <asp:Label ID="detailsBrPageLabel" runat="server" Text="Br page"></asp:Label>
        </td>
        <td>
            <asp:TextBox ID="detailsBrPageTextBox" runat="server" ReadOnly="true"></asp:TextBox>
        </td>
        <td align="right">
            <asp:Label ID="detailsGalinosLabel" runat="server" Text="Galinos page"></asp:Label>
        </td>
        <td>
            <asp:TextBox ID="detailsGalinosTextBox" runat="server" ReadOnly="true"></asp:TextBox>
        </td>
    </tr>
    <tr>
        <td rowspan="2" colspan="5">
            <asp:TextBox ID="detailsCommentsTextBox" TextMode="MultiLine" Width="100%" runat="server"
ReadOnly="true"></asp:TextBox>
        </td>
        <td colspan="2" align="center">
            <asp:Button ID="detailsBackButton" runat="server" Text="back to search"
OnClick="detailsBackButton_Click" />
        </td>
    </tr>
    <tr>
        <td colspan="2" align="center">
            <asp:Button ID="detailsNewDataButton" runat="server" Text="enter new data"
PostBackUrl="~/NewData.aspx" />
        </td>
    </tr>
    <tr>
    <td colspan="7" style="height: 40px">
        <br/>
    </td>
    </tr>
    </table>


        <asp:GridView ID="GridView1" Width="100%" runat="server" AutoGenerateColumns="False"
DataSourceID="" BackColor="White" BorderColor="#999999" BorderStyle="Solid" BorderWidth="1px"
CellPadding="3" ForeColor="Black" GridLines="Vertical" AllowPaging="true" PageSize="50"
DataKeyNames="ID" OnSelectedIndexChanged="rowSelected" OnDataBound="GridView1_DataBound" >
            <FooterStyle BackColor="#CCCCCC" />
            <SelectedRowStyle BackColor="#000099" Font-Bold="True" ForeColor="White" />
            <PagerStyle BackColor="#999999" ForeColor="Black" HorizontalAlign="Center" />
            <HeaderStyle BackColor="Black" Font-Bold="True" ForeColor="White" />
```

```
        <AlternatingRowStyle BackColor="#CCCCCC" />
        <Columns>
          <asp:BoundField DataField="ID" HeaderText="ID" InsertVisible="False" ReadOnly="True"
            SortExpression="ID" />
          <asp:BoundField DataField="Normalised Latin" HeaderText="Normalised Latin"
SortExpression="Normalised Latin" />
          <asp:BoundField DataField="Normalised Arabic" HeaderText="Normalised Arabic"
SortExpression="Normalised Arabic" />
          <asp:BoundField DataField="Post Office" HeaderText="Post Office" SortExpression="Post Office"
/>
          <asp:BoundField DataField="CW" HeaderText="CW" SortExpression="CW" />
          <asp:BoundField DataField="Br" HeaderText="Br" SortExpression="Br" />
          <asp:ButtonField ButtonType="Button" CommandName="Select" Text="details" />
        </Columns>
      </asp:GridView>
      <asp:ObjectDataSource ID="cancelDataSource" runat="server"
OldValuesParameterFormatString="original_{0}"
        TypeName="DataSet1TableAdapters.CancellationsTableAdapter" >
        <SelectParameters>
          <asp:Parameter Name="Post_Office" Type="String" />
        </SelectParameters>
      </asp:ObjectDataSource>

    </div>
    </form>
</body>
</html>
```

A4) αρχείο Default.aspx.cs :

```
using System;
using System.Collections;
using System.Data;
using System.Configuration;
using System.Web;
using System.Web.Security;
using System.Web.UI;
using System.Web.UI.WebControls;
using System.Web.UI.WebControls.WebParts;
using System.Web.UI.HtmlControls;
using System.IO;

public partial class _Default : System.Web.UI.Page
{
    protected static string title1 = "Osmanl\u0131 Posta Damgalar\u0131";
    protected static string title2 =
"\u0639\u062B\u0645\u0627\u0646\u0644\u06CC\u0020\u0627\u0645\u067E\u0631\u062A\u0648\u0631\u0
644\u06A9\u06CC";
    protected static string title3 = "Ottoman Postal Cancellations";

    protected void Page_Load(object sender, EventArgs e)
    {
        if (Session.Count == 0)     //new session object(opening site)
        {
            Session["selectMethod"] = "";
            Session["filterExpr"] = "null";
            Session["selectParamCount"] = 0;
            Session["searchText"] = "";
            Session["filterText"] = "";
```

```
        }
        restoreGridViewData();

        if (!IsPostBack)
        {
            titleLabel1.Text = title1;
            titleLabel2.Text = title3;
            Session["vilaetSelected"] = false;
            Session["countrySelected"] = false;
            ViewState["postOfficeSelected"] = false;
            initKeyboard();
        }
        else
        {
            idLabel.Visible = false;
            dateLabel.Visible = false;
            warningTextBox.Visible = false;
        }
    }

    protected void languageChanged(object sender, EventArgs e)
    {
        if (RadioButtonList1.SelectedValue.Equals("latinised"))
        {
            titleLabel1.Text = title1;
            titleLabel2.Text = title3;
            vilaetDataSource.SelectCommand = "SELECT * FROM [RegionOttoman] ORDER BY [VilaetL]";
            countryDataSource.SelectCommand = "SELECT * FROM [Countries] ORDER BY [Country]";
            vilaetDDL.DataTextField = "VilaetL";
            vilaetDDL.DataValueField = "VilaetO";
            countryDDL.DataTextField = "Country";
            countryDDL.DataValueField = "CountryMulti";

            if ((bool)Session["vilaetSelected"])
            {
                chosenLabel.Text = (string)Session["selectedLatin"];
                if (Session["selectedLatin"].Equals("*"))
                    localNamesDataSource.SelectCommand = "SELECT [POTurk], [POOtto] FROM [Local Names]
ORDER BY [POTurk]";
                else
                    localNamesDataSource.SelectCommand = "SELECT [POTurk], [POOtto] FROM [Local Names]
WHERE ([Vilaet] = \"" +
                                Session["selectedLatin"] + "\") ORDER BY [POTurk]";
                postOfficeDDL.DataTextField = "POTurk";
                postOfficeDDL.DataValueField = "POTurk";
            }
            else if ((bool)Session["countrySelected"])
            {
                chosenLabel.Text = (string)Session["selectedLatin"];
                localNamesDataSource.SelectCommand = "SELECT [POTurk], [ModLatin] FROM [Local Names]
WHERE ([Country] = \"" +
                                Session["selectedLatin"] + "\") ORDER BY [ModLatin]";
                postOfficeDDL.DataTextField = "ModLatin";
                postOfficeDDL.DataValueField = "POTurk";
            }
            else
            {
                localNamesDataSource.SelectCommand = "SELECT [POTurk], [POOtto] FROM [Local Names]
WHERE ([POTurk] IS NOT NULL) ORDER BY [POTurk]";
                postOfficeDDL.DataTextField = "POTurk";
```

```
            postOfficeDDL.DataValueField = "POTurk";
        }
        if ((bool)ViewState["postOfficeSelected"])
        {
            postOfficeDDL.SelectedValue = (string)ViewState["selectedPOTurk"];
        }
    }
    else if (RadioButtonList1.SelectedValue.Equals("multilingual"))
    {
        titleLabel1.Text = title2;
        titleLabel2.Text = "";
        vilaetDataSource.SelectCommand = "SELECT * FROM [RegionOttoman] ORDER BY [VilaetO]";
        countryDataSource.SelectCommand = "SELECT * FROM [Countries] ORDER BY [CountryMulti]";
        vilaetDDL.DataTextField = "VilaetO";
        vilaetDDL.DataValueField = "VilaetL";
        countryDDL.DataTextField = "CountryMulti";
        countryDDL.DataValueField = "Country";

        if ((bool)Session["vilaetSelected"])
        {
            chosenLabel.Text = (string)Session["selectedMulti"];
            if (Session["selectedLatin"].Equals("*"))
                localNamesDataSource.SelectCommand = "SELECT [POTurk], [POOtto] FROM [Local Names]
ORDER BY [POOtto]";
            else
                localNamesDataSource.SelectCommand = "SELECT [POTurk], [POOtto] FROM [Local Names]
WHERE ([Vilaet] = \"" +
                                Session["selectedLatin"] + "\") ORDER BY [POOtto]";
            postOfficeDDL.DataTextField = "POOtto";
            postOfficeDDL.DataValueField = "POTurk";
        }
        else if ((bool)Session["countrySelected"])
        {
            chosenLabel.Text = (string)Session["selectedMulti"];
            localNamesDataSource.SelectCommand = "SELECT [POTurk], [ModMulti] FROM [Local Names]
WHERE ([Country] = \"" +
                                 Session["selectedLatin"] + "\") ORDER BY [ModMulti]";
            postOfficeDDL.DataTextField = "ModMulti";
            postOfficeDDL.DataValueField = "POTurk";
        }
        else
        {
            localNamesDataSource.SelectCommand = "SELECT [POTurk], [POOtto] FROM [Local Names]
WHERE ([POOtto] IS NOT NULL) ORDER BY [POOtto]";
            postOfficeDDL.DataTextField = "POOtto";
            postOfficeDDL.DataValueField = "POTurk";
        }
        if ((bool)ViewState["postOfficeSelected"])
        {
            postOfficeDDL.SelectedValue = (string)ViewState["selectedPOTurk"];
        }
    }
}

protected void vilaetChosen(object sender, EventArgs e)
{
    if (RadioButtonList1.SelectedValue.Equals("latinised"))
    {
        Session["selectedLatin"] = "" + vilaetDDL.SelectedItem;
        Session["selectedMulti"] = vilaetDDL.SelectedValue;
```

```
            chosenLabel.Text = (string)Session["selectedLatin"];
            if (Session["selectedLatin"].Equals("*"))
                localNamesDataSource.SelectCommand = "SELECT [POTurk], [POOtto] FROM [Local Names]
ORDER BY [POTurk]";
            else
                localNamesDataSource.SelectCommand = "SELECT [POTurk], [POOtto] FROM [Local Names]
WHERE ([Vilaet] = \"" +
                                    vilaetDDL.SelectedItem + "\") ORDER BY [POTurk]";
            postOfficeDDL.DataTextField = "POTurk";
            postOfficeDDL.DataValueField = "POTurk";
        }
        else if (RadioButtonList1.SelectedValue.Equals("multilingual"))
        {
            Session["selectedMulti"] = "" + vilaetDDL.SelectedItem;
            Session["selectedLatin"] = vilaetDDL.SelectedValue;
            chosenLabel.Text = (string)Session["selectedMulti"];
            if (vilaetDDL.SelectedValue.Equals("*"))
                localNamesDataSource.SelectCommand = "SELECT [POTurk], [POOtto] FROM [Local Names]
ORDER BY [POOtto]";
            else
                localNamesDataSource.SelectCommand = "SELECT [POTurk], [POOtto] FROM [Local Names]
WHERE ([Vilaet] = \"" +
                                    vilaetDDL.SelectedValue + "\") ORDER BY [POOtto]";
            postOfficeDDL.DataTextField = "POOtto";
            postOfficeDDL.DataValueField = "POTurk";
        }
        Session["countrySelected"] = false;
        Session["vilaetSelected"] = true;
        ViewState["postOfficeSelected"] = false;
        ViewState["selectedPOTurk"] = null;
        chosenLabel.ForeColor = System.Drawing.Color.Red;

        if (Session["selectedLatin"].Equals("*"))
        {
            setSourceAsIdAll();
        }
        else
        {
            setSourceAsVilaet();
            cancelDataSource.SelectParameters["Vilaet"].DefaultValue = (string)Session["selectedLatin"];
            Session["value0"] = (string)Session["selectedLatin"];
        }
        resetFilterExpression();
        GridView1.DataSourceID = "cancelDataSource";
    }

    protected void countryChosen(object sender, EventArgs e)
    {
        if (RadioButtonList1.SelectedValue.Equals("latinised"))
        {
            Session["selectedLatin"] = "" + countryDDL.SelectedItem;
            Session["selectedMulti"] = countryDDL.SelectedValue;
            chosenLabel.Text = (string)Session["selectedLatin"];
            localNamesDataSource.SelectCommand = "SELECT [POTurk], [ModLatin] FROM [Local Names]
WHERE ([Country] = \"" +
                                    countryDDL.SelectedItem + "\") ORDER BY [ModLatin]";
            postOfficeDDL.DataTextField = "ModLatin";
            postOfficeDDL.DataValueField = "POTurk";
        }
        else if (RadioButtonList1.SelectedValue.Equals("multilingual"))
```

```
        {
            Session["selectedMulti"] = "" + countryDDL.SelectedItem;
            Session["selectedLatin"] = countryDDL.SelectedValue;
            chosenLabel.Text = (string)Session["selectedMulti"];
            localNamesDataSource.SelectCommand = "SELECT [POTurk], [ModMulti] FROM [Local Names]
WHERE ([Country] = \"" +
                                 countryDDL.SelectedValue + "\") ORDER BY [ModMulti]";
            postOfficeDDL.DataTextField = "ModMulti";
            postOfficeDDL.DataValueField = "POTurk";
        }
        Session["vilaetSelected"] = false;
        Session["countrySelected"] = true;
        ViewState["postOfficeSelected"] = false;
        ViewState["selectedPOTurk"] = null;
        chosenLabel.ForeColor = System.Drawing.Color.Blue;

        setSourceAsCountry();
        cancelDataSource.SelectParameters["Country"].DefaultValue = (string)Session["selectedLatin"];
        Session["value0"] = (string)Session["selectedLatin"];
        resetFilterExpression();
        GridView1.DataSourceID = "cancelDataSource";
    }

    protected void postOfficeChosen(object sender, EventArgs e)
    {
        ViewState["postOfficeSelected"] = true;
        ViewState["selectedPOTurk"] = postOfficeDDL.SelectedValue;
        setSourceAsPostOffice();
        cancelDataSource.SelectParameters["Post_Office"].DefaultValue = postOfficeDDL.SelectedValue;
        Session["value0"] = postOfficeDDL.SelectedValue;
        resetFilterExpression();
        GridView1.DataSourceID = "cancelDataSource";
    }

    protected void editButton_Click(object sender, EventArgs e)
    {
        if ( (bool)Session["countrySelected"] ||
             (bool)Session["vilaetSelected"] )
        {
            Response.Redirect("EditData.aspx");
        }
    }

    protected void cwDDL_DataBound(object sender, EventArgs e)
    {
        cwDDL.Items.Remove("?");
    }
    protected void brandtDDL_DataBound(object sender, EventArgs e)
    {
        brandtDDL.Items.Remove("-");
        brandtDDL.Items.Remove("?");
    }

    protected void cwChosen(object sender, EventArgs e)
    {
        brandtDDL.SelectedIndex = 0;       //filter only according to CW
        if (cwDDL.SelectedValue.Equals("none"))
        {
            Session["filterExpr"] = "CW IS NULL OR CW=' '";
            Session["filterText"] = ", having no CW code";
```

```
            cancelDataSource.FilterExpression = "CW IS NULL OR CW=' '";
            return;
        }
        if (cwDDL.SelectedValue.Equals("*"))
        {
            resetFilterExpression();
            return;
        }

        string temp = cwDDL.SelectedValue;
        temp = temp.Replace('*', '%');
        temp = temp.Replace('?', '%');       //den doulevoun ?,_

        if (temp.Contains("/"))    //dyo kodikoi pou xorizontai apo /
        {
            int i = temp.IndexOf('/');
            string temp1 = temp.Substring(0, i);
            temp = temp.Substring(i + 1);
            Session["filterExpr"] = "(CW LIKE '" + temp1 +
                        "') OR (CW LIKE '" + temp + "')";
            Session["filterText"] = ", having CW code " +
               temp1 + " or " + temp;
        }
        else
        {
            Session["filterExpr"] = "CW LIKE '" + temp + "'";
            Session["filterText"] = ", having CW code " + temp;
        }

        cancelDataSource.FilterExpression = (string)Session["filterExpr"];
    }

    protected void brandtChosen(object sender, EventArgs e)
    {
        cwDDL.SelectedIndex = 0;       //filter only according to Br
        if (brandtDDL.SelectedValue.Equals("*"))
        {
            resetFilterExpression();
            return;
        }
        if (brandtDDL.SelectedValue.Equals("none"))
        {
            Session["filterExpr"] = "Br IS NULL OR Br=' '";
            Session["filterText"] = ", having no brand type";
            cancelDataSource.FilterExpression = "Br IS NULL OR Br=' '";
            return;
        }
        Session["filterExpr"] = "Br='" + brandtDDL.SelectedValue + "'";
        Session["filterText"] = ", having brand type " + brandtDDL.SelectedValue;
        cancelDataSource.FilterExpression = "Br='" + brandtDDL.SelectedValue + "'";
    }

    protected void idButton_Clicked(object sender, EventArgs e)
    {
        int id;
        try
        {
            id = Int32.Parse(idTextBox.Text);
        }
        catch (Exception ex)
```

```
    {
        idLabel.Text = "id must be numeric and not null!";
        idLabel.Visible = true;
        GridView1.DataSourceID = "";
        Session["selectMethod"] = "";
        Session["selectParamCount"] = 0;
        resetFilterExpression();
        return;
    }
    if (id == 0)
    {
        setSourceAsIdAll();
    }
    else
    {
        setSourceAsId();
        cancelDataSource.SelectParameters["ID"].DefaultValue = "" + id;
        Session["value0"] = "" + id;
    }
    resetFilterExpression();
    GridView1.DataSourceID = "cancelDataSource";
}

protected void searchButton_Click(object sender, EventArgs e)
{
    string frenchInput = frenchTextBox.Text.Trim();
    string ottoInput = ottomanTextBox.Text.Trim();

    if (frenchInput.Equals(String.Empty) || frenchInput.Equals("="))
    {
        frenchInput = "=";
    }
    else
    {
        if (ConstantClass.containsOttoman(frenchInput))
        {
            warningTextBox.Rows = 1;
            warningTextBox.Text = "Please remove ottoman characters from french textField.\n";
            warningTextBox.Visible = true;
            frenchInput = ConstantClass.removedOttoman(frenchInput);
        }
        else
        {
            warningTextBox.Rows = 0;
            warningTextBox.Text = "";
        }
        frenchInput = ConstantClass.strOneLine(frenchInput);
        frenchInput = ConstantClass.expandedFrench(frenchInput);
        frenchInput = ConstantClass.wildcarded_fr(frenchInput);
    }

    if (ottoInput.Equals(String.Empty) || ottoInput.Equals("="))
    {
        ottoInput = "=";
    }
    else
    {
        if (ConstantClass.containsFrench(ottoInput))
        {
            warningTextBox.Visible = true;
```

```
            warningTextBox.Rows = warningTextBox.Rows + 1;
            warningTextBox.Text = warningTextBox.Text +
                "Please remove latin characters from ottoman textField.\n";
            ottoInput = ConstantClass.removedFrench(ottoInput);
        }
        ottoInput = ConstantClass.strOneLine(ottoInput);
        ottoInput = ConstantClass.strAllArabised(ottoInput);
        ottoInput = ConstantClass.expandedOttoman(ottoInput);
        ottoInput = ottoInput.Replace("\u061f", "?");
        ottoInput = ottoInput.Replace("\u066d", "*");
        ottoInput = ConstantClass.wildcarded_ot(ottoInput);
    }

    setSourceAsFrenchOttoman();
    cancelDataSource.SelectParameters["French_Field"].DefaultValue = frenchInput;
    Session["value0"] = frenchInput;
    cancelDataSource.SelectParameters["Ottoman_Field"].DefaultValue = ottoInput;
    Session["value1"] = ottoInput;
    resetFilterExpression();
    GridView1.DataSourceID = "cancelDataSource";
}

protected void negativeButton_Click(object sender, EventArgs e)
{
    setSourceAsNegative();
    resetFilterExpression();
    GridView1.DataSourceID = "cancelDataSource";
}

protected void expressionButton_Click(object sender, EventArgs e)
{
    Session["shapeSearchString"] = "%";
    Session["shapeSearchText"] = " negative cancellation(s) " +
            "listed with unspecified shape and";
    Response.Redirect("expressionSearch.aspx");
}

protected void postOfficeDDL_DataBound(object sender, EventArgs e)
{
    postOfficeDDL.Items.Insert(0, "choose post office");
}

protected void keyboardDDL_DataBound(object sender, EventArgs e)
{
    DropDownList ddl = (DropDownList)sender;
    string text="";

    if (ddl.ID.Equals("keyboardOttoDDL"))
        text = "otto";
    else if (ddl.ID.Equals("keyboardTurkDDL"))
        text = "turk";
    else if (ddl.ID.Equals("keyboardEnglDDL"))
        text = "engl";

    ddl.Items.Insert(0, text);
}

protected void keyboardButton_Click(object sender, EventArgs e)
{
    keyboardTable.Visible = true;
```

```
}

protected void keyButton_Click(object sender, EventArgs e)
{
   Button but = (Button)sender;
   string temp = ottomanTextBox.Text;

   if (but.ID.Equals("keyCloseButton"))
      keyboardTable.Visible = false;
   else if (but.ID.Equals("keySpaceButton"))
      ottomanTextBox.Text = temp + " ";
   else if (but.ID.Equals("keyButton13"))          //backspace
   {
      if (temp.Length > 1)
         ottomanTextBox.Text = temp.Substring(0, temp.Length - 1);
      else
         ottomanTextBox.Text = "";
   }
   else if (but.ID.Equals("keyButton40"))          //newline
   {
      ottomanTextBox.Text = temp + "\n";
   }
   else
      ottomanTextBox.Text = temp + but.Text;
}

protected void commonTermChosen(object sender, EventArgs e)
{
   int i = ((DropDownList)sender).SelectedIndex;
   keyboardOttoDDL.SelectedIndex = i;
   keyboardTurkDDL.SelectedIndex = i;
   keyboardEnglDDL.SelectedIndex = i;
   if (!keyboardOttoDDL.SelectedValue.Equals("otto"))
      ottomanTextBox.Text = keyboardOttoDDL.SelectedValue;
}

protected void rowSelected(object sender, EventArgs e)
{
   GridViewRow row = GridView1.SelectedRow;
   menuTable.Visible = false;
   detailsTable.Visible = true;

   detailsDataSource.SelectParameters["ID"].DefaultValue = row.Cells[0].Text;
   DataView dv = (DataView)detailsDataSource.Select();

   if (dv.Table.Rows[0]["Dates"] is System.DBNull)
   {
      detailsDatesLabel.Visible = false;
      detailsDatesTextBox.Visible = false;
      Session["selectedCancelDates"] = "";
   }
   else
   {
      string temp = (string)dv.Table.Rows[0]["Dates"];
      detailsDatesLabel.Visible = true;
      detailsDatesTextBox.Visible = true;
      detailsDatesTextBox.Text = temp;
      Session["selectedCancelDates"] = temp;
   }
```

```
if (dv.Table.Rows[0]["ID"] is System.DBNull)
{
   detailsIDLabel.Visible = false;
   detailsIDTextBox.Visible = false;
   Session["selectedCancelID"] = "";
}
else
{
   int temp = (int)dv.Table.Rows[0]["ID"];
   detailsIDLabel.Visible = true;
   detailsIDTextBox.Visible = true;
   detailsIDTextBox.Text = "" + temp;
   Session["selectedCancelID"] = temp;
}

if (dv.Table.Rows[0]["CW"] is System.DBNull)
{
   detailsCWLabel.Visible = false;
   detailsCWTextBox.Visible = false;
   Session["selectedCancelCW"] = "";
}
else
{
   string temp = (string)dv.Table.Rows[0]["CW"];
   detailsCWLabel.Visible = true;
   detailsCWTextBox.Visible = true;
   detailsCWTextBox.Text = temp;
   Session["selectedCancelCW"] = temp;
   if ( temp.StartsWith("S") )
   {
      detailsOttoFieldTextBox.BackColor = System.Drawing.Color.Black;
      detailsOttoFieldTextBox.ForeColor = System.Drawing.Color.White;
      detailsFrenchFieldTextBox.BackColor = System.Drawing.Color.Black;
      detailsFrenchFieldTextBox.ForeColor = System.Drawing.Color.White;
   }
   else
   {
      detailsOttoFieldTextBox.BackColor = System.Drawing.Color.White;
      detailsOttoFieldTextBox.ForeColor = System.Drawing.Color.Black;
      detailsFrenchFieldTextBox.BackColor = System.Drawing.Color.White;
      detailsFrenchFieldTextBox.ForeColor = System.Drawing.Color.Black;
   }
}

if (dv.Table.Rows[0]["CWPage"] is System.DBNull)
{
   detailsCWPageLabel.Visible = false;
   detailsCWPageTextBox.Visible = false;
   Session["selectedCancelCWPage"] = "";
}
else
{
   string temp = (string)dv.Table.Rows[0]["CWPage"];
   detailsCWPageLabel.Visible = true;
   detailsCWPageTextBox.Visible = true;
   detailsCWPageTextBox.Text = temp;
   Session["selectedCancelCWPage"] = temp;
}

if (dv.Table.Rows[0]["Br"] is System.DBNull)
```

```
{
    detailsBrLabel.Visible = false;
    detailsBrTextBox.Visible = false;
    Session["selectedCancelBr"] = "";
}
else
{
    string temp = (string)dv.Table.Rows[0]["Br"];
    detailsBrLabel.Visible = true;
    detailsBrTextBox.Visible = true;
    detailsBrTextBox.Text = temp;
    Session["selectedCancelBr"] = temp;
}

if (dv.Table.Rows[0]["BrandtPage"] is System.DBNull)
{
    detailsBrPageLabel.Visible = false;
    detailsBrPageTextBox.Visible = false;
    Session["selectedCancelBrPage"] = "";
}
else
{
    string temp = (string)dv.Table.Rows[0]["BrandtPage"];
    detailsBrPageLabel.Visible = true;
    detailsBrPageTextBox.Visible = true;
    detailsBrPageTextBox.Text = temp;
    Session["selectedCancelBrPage"] = temp;
}

if (dv.Table.Rows[0]["Nuhoglou"] is System.DBNull)
{
    detailsNuhoglouLabel.Visible = false;
    detailsNuhoglouTextBox.Visible = false;
    Session["selectedCancelNuhoglou"] = "";
}
else
{
    string temp = (string)dv.Table.Rows[0]["Nuhoglou"];
    detailsNuhoglouLabel.Visible = true;
    detailsNuhoglouTextBox.Visible = true;
    detailsNuhoglouTextBox.Text = temp;
    Session["selectedCancelNuhoglou"] = temp;
}

if (dv.Table.Rows[0]["Galinos"] is System.DBNull)
{
    detailsGalinosLabel.Visible = false;
    detailsGalinosTextBox.Visible = false;
    Session["selectedCancelGalinos"] = "";
}
else
{
    string temp = (string)dv.Table.Rows[0]["Galinos"];
    detailsGalinosLabel.Visible = true;
    detailsGalinosTextBox.Visible = true;
    detailsGalinosTextBox.Text = temp;
    Session["selectedCancelGalinos"] = temp;
}

if (dv.Table.Rows[0]["Comments"] is System.DBNull)
```

```
{
    detailsCommentsTextBox.Visible = false;
    Session["selectedCancelComments"] = "";
}
else
{
    string temp = (string)dv.Table.Rows[0]["Comments"];
    detailsCommentsTextBox.Visible = true;
    detailsCommentsTextBox.Text = temp;
    Session["selectedCancelComments"] = temp;
}

if ((bool)dv.Table.Rows[0]["FDef"])
{
    detailsFrenchCheckBox.Checked = true;
    Session["selectedCancelFDef"] = true;
}
else
{
    detailsFrenchCheckBox.Checked = false;
    Session["selectedCancelFDef"] = false;
}

if ((bool)dv.Table.Rows[0]["ODef"])
{
    detailsOttoCheckBox.Checked = true;
    Session["selectedCancelODef"] = true;
}
else
{
    detailsOttoCheckBox.Checked = false;
    Session["selectedCancelODef"] = false;
}

if (dv.Table.Rows[0]["French Field"].Equals("="))
{
    detailsFrenchFieldTextBox.Visible = false;
    detailsFrenchCheckBox.Visible = false;
    Session["selectedCancelFrenchField"] = "";
}
else
{
    string temp = (string)dv.Table.Rows[0]["French Field"];
    detailsFrenchFieldTextBox.Text = temp;
    detailsFrenchFieldTextBox.Visible = true;
    detailsFrenchCheckBox.Visible = true;
    Session["selectedCancelFrenchField"] = temp;
}

if (dv.Table.Rows[0]["Ottoman Field"].Equals("="))
{
    detailsOttoFieldTextBox.Visible = false;
    detailsOttoCheckBox.Visible = false;
    Session["selectedCancelOttoField"] = "";
}
else
{
    string temp = (string)dv.Table.Rows[0]["Ottoman Field"];
    detailsOttoFieldTextBox.Text = temp;
    detailsOttoFieldTextBox.Visible = true;
```

```
            detailsOttoCheckBox.Visible = true;
            Session["selectedCancelOttoField"] = temp;
        }

        Session["selectedCancelPostOffice"] = "" + dv.Table.Rows[0]["POTurk"];
        string vilaet = (string)dv.Table.Rows[0]["Vilaet"];
        if (vilaet.EndsWith("S."))
            detailsPOTextBox.Text = "" + dv.Table.Rows[0]["POTurk"] +
                    "\n" + dv.Table.Rows[0]["POOtto"] +
                    "\n" + vilaet.Substring(0, vilaet.Length - 3) +
                    "\n" + "Sancak";
        else if (vilaet.EndsWith("V."))
            detailsPOTextBox.Text = "" + dv.Table.Rows[0]["POTurk"] +
                    "\n" + dv.Table.Rows[0]["POOtto"] +
                    "\n" + vilaet.Substring(0, vilaet.Length - 3) +
                    "\n" + "Vil\u03b2yeti";
        else
            detailsPOTextBox.Text = "" + dv.Table.Rows[0]["POTurk"] +
                    "\n" + dv.Table.Rows[0]["POOtto"] +
                    "\n" + vilaet;

        detailsLocalNamesTextBox.Text = "" + dv.Table.Rows[0]["Country"] +
                    ":\n" + dv.Table.Rows[0]["ModMulti"] +
                    "\n" + dv.Table.Rows[0]["ModLatin"];

        if (dv.Table.Rows[0]["Colours"] is System.DBNull)
        {
            Session["selectedCancelColours"] = "";
        }
        else
        {
            string temp = (string)dv.Table.Rows[0]["Colours"];
            detailsColorsLiteral.Text = colorsToString( temp );
            Session["selectedCancelColours"] = temp;
        }

        if( dv.Table.Rows[0]["Image"] is System.DBNull )
        {
            Session["selectedCancelImage"] = "";
            imageContainer.ImagesUrlsString = "";
        }
        else
        {
            string temp = (string)dv.Table.Rows[0]["Image"];
            imageContainer.ImagesUrlsString = temp;
            Session["selectedCancelImage"] = temp;
        }
    }

    protected void detailsBackButton_Click(object sender, EventArgs e)
    {
        detailsTable.Visible = false;
        menuTable.Visible = true;
    }


    private void setSourceAsPostOffice()
    {
        if (cancelDataSource.SelectMethod.Equals("GetDataByPostOffice"))
            return;
```

```
    for (int i = cancelDataSource.SelectParameters.Count - 1; i >= 0; i--)
        cancelDataSource.SelectParameters.RemoveAt(i);
    cancelDataSource.SelectParameters.Add("Post_Office", TypeCode.String, "");
    Session["name0"] = "Post_Office";
    Session["type0"] = TypeCode.String;
    Session["value0"] = "";
    Session["selectParamCount"] = 1;
    cancelDataSource.SelectMethod = "GetDataByPostOffice";
    Session["selectMethod"] = "GetDataByPostOffice";
}
private void setSourceAsId()
{
    if (cancelDataSource.SelectMethod.Equals("GetDataByID"))
        return;
    for (int i = cancelDataSource.SelectParameters.Count - 1; i >= 0; i--)
        cancelDataSource.SelectParameters.RemoveAt(i);
    cancelDataSource.SelectParameters.Add("ID", TypeCode.Int32, "1");
    Session["name0"] = "ID";
    Session["type0"] = TypeCode.Int32;
    Session["value0"] = "1";
    Session["selectParamCount"] = 1;
    cancelDataSource.SelectMethod = "GetDataByID";
    Session["selectMethod"] = "GetDataByID";
}
private void setSourceAsIdAll()
{
    if (cancelDataSource.SelectMethod.Equals("GetData"))
        return;
    for (int i = cancelDataSource.SelectParameters.Count - 1; i >= 0; i--)
        cancelDataSource.SelectParameters.RemoveAt(i);
    Session["selectParamCount"] = 0;
    cancelDataSource.SelectMethod = "GetData";
    Session["selectMethod"] = "GetData";
}
private void setSourceAsFrenchOttoman()
{
    if (cancelDataSource.SelectMethod.Equals("GetDataByFrenchOttoman"))
        return;
    for (int i = cancelDataSource.SelectParameters.Count - 1; i >= 0; i--)
        cancelDataSource.SelectParameters.RemoveAt(i);
    cancelDataSource.SelectParameters.Add("French_Field", TypeCode.String, "");
    cancelDataSource.SelectParameters.Add("Ottoman_Field", TypeCode.String, "");
    Session["name0"] = "French_Field";
    Session["type0"] = TypeCode.String;
    Session["value0"] = "";
    Session["name1"] = "Ottoman_Field";
    Session["type1"] = TypeCode.String;
    Session["value1"] = "";
    Session["selectParamCount"] = 2;
    cancelDataSource.SelectMethod = "GetDataByFrenchOttoman";
    Session["selectMethod"] = "GetDataByFrenchOttoman";
}
private void setSourceAsNegative()
{
    if (cancelDataSource.SelectMethod.Equals("GetDataNegative"))
        return;
    for (int i = cancelDataSource.SelectParameters.Count - 1; i >= 0; i--)
        cancelDataSource.SelectParameters.RemoveAt(i);
    Session["selectParamCount"] = 0;
    cancelDataSource.SelectMethod = "GetDataNegative";
```

```
    Session["selectMethod"] = "GetDataNegative";
}
private void setSourceAsCountry()
{
    if (cancelDataSource.SelectMethod.Equals("GetDataByCountry"))
        return;
    for (int i = cancelDataSource.SelectParameters.Count - 1; i >= 0; i--)
        cancelDataSource.SelectParameters.RemoveAt(i);

    cancelDataSource.SelectParameters.Add("Country", TypeCode.String, "");
    Session["name0"] = "Country";
    Session["type0"] = TypeCode.String;
    Session["value0"] = "";
    Session["selectParamCount"] = 1;
    cancelDataSource.SelectMethod = "GetDataByCountry";
    Session["selectMethod"] = "GetDataByCountry";
}
private void setSourceAsVilaet()
{
    if (cancelDataSource.SelectMethod.Equals("GetDataByVilaet"))
        return;
    for (int i = cancelDataSource.SelectParameters.Count - 1; i >= 0; i--)
        cancelDataSource.SelectParameters.RemoveAt(i);
    cancelDataSource.SelectParameters.Add("Vilaet", TypeCode.String, "");
    Session["name0"] = "Vilaet";
    Session["type0"] = TypeCode.String;
    Session["value0"] = "";
    Session["selectParamCount"] = 1;
    cancelDataSource.SelectMethod = "GetDataByVilaet";
    Session["selectMethod"] = "GetDataByVilaet";
}

private void resetFilterExpression()
{
    Session["filterText"] = "";
    Session["filterExpr"] = "null";
    cancelDataSource.FilterExpression = null;
    cwDDL.SelectedIndex = 0;
    brandtDDL.SelectedIndex = 0;
}

private static string ChrW(int i)
{
    return "\\u" + String.Format("{0:x4}", i);
}

private static string colorsToString(string col)
{
    string[] tokens = col.Split(',');
    string results = "";

    for (int i = 0; i < tokens.Length; i++)
    {
        string temp = colorAcronymToString(tokens[i]);
        if( !temp.Equals(String.Empty) )
            results += "<font color=\"" + colorAcronymToCode(tokens[i]) +
                "\">" + temp + "</font> ";
    }
        return results;
}
```

```
private static string colorAcronymToString(string col)
{
   switch (col)
   {
      case "blk":
         return "black";
      case "blu":
         return "blue";
      case "brn":
         return "brown";
      case "crm":
         return "carmine";
      case "grn":
         return "green";
      case "red":
         return "red";
      case "vlt":
         return "violet";
      default:
         return "";
   }
}
private static string colorAcronymToCode(string col)
{
   switch (col)
   {
      case "blk":
         return "#000000";
      case "blu":
         return "#0000FF";
      case "brn":
         return "#A52A2A";
      case "crm":
         return "#960018";
      case "grn":
         return "#00FF00";
      case "red":
         return "#FF0000";
      case "vlt":
         return "#EE82EE";
      default:
         return "";
   }
}

private void initKeyboard()
{
   keyButton1.Text = "\u0661";
   keyButton2.Text = "\u0662";
   keyButton3.Text = "\u0663";
   keyButton4.Text = "\u0664";
   keyButton5.Text = "\u0665";
   keyButton6.Text = "\u0666";
   keyButton7.Text = "\u0667";
   keyButton8.Text = "\u0668";
   keyButton9.Text = "\u0669";
   keyButton10.Text = "\u0660";
   keyButton11.Text = "\u002d";
   keyButton12.Text = "\u003d";
   keyButton13.Text = "bksp";
```

```
    keyButton14.Text = "\u0636";
    keyButton15.Text = "\u0635";
    keyButton16.Text = "\u062b";
    keyButton17.Text = "\u0642";
    keyButton18.Text = "\u0641";
    keyButton19.Text = "\u063a";
    keyButton20.Text = "\u0639";
    keyButton21.Text = "\u0647";
    keyButton22.Text = "\u062e";
    keyButton23.Text = "\u062d";
    keyButton24.Text = "\u062c";
    keyButton25.Text = "\u0686";
    keyButton26.Text = "\u067e";

    keyButton27.Text = "\u0634";
    keyButton28.Text = "\u0633";
    keyButton29.Text = "\u06cc";
    keyButton30.Text = "\u0628";
    keyButton31.Text = "\u0644";
    keyButton32.Text = "\u0627";
    keyButton33.Text = "\u062a";
    keyButton34.Text = "\u0646";
    keyButton35.Text = "\u0645";
    keyButton36.Text = "\u06a9";
    keyButton37.Text = "\u06af";
    keyButton38.Text = "\u066d";
    keyButton39.Text = "\u061f";
    keyButton40.Text = "NewLine";

    keyButton41.Text = "\u0629";
    keyButton42.Text = "\u0638";
    keyButton43.Text = "\u0637";
    keyButton44.Text = "\u0698";
    keyButton45.Text = "\u0632";
    keyButton46.Text = "\u0631";
    keyButton47.Text = "\u0630";
    keyButton48.Text = "\u062f";
    keyButton49.Text = "\u0626";
    keyButton50.Text = "\u0648";
    keyButton51.Text = "\u064a";
    keyButton52.Text = "\u0622";
    keyButton53.Text = "\u0028";
    keyButton54.Text = "\u0029";
}

private void restoreGridViewData()
{
    if (Session["selectMethod"].Equals(""))
    {
        Session["searchText"] = "";
        Session["filterText"] = "";
        foundLabel1.Text = "";
        return;
    }

    for (int i = cancelDataSource.SelectParameters.Count - 1; i >= 0; i--)
        cancelDataSource.SelectParameters.RemoveAt(i);
    for (int i = 0; i < (int)Session["selectParamCount"]; i++)
    {
```

```
            cancelDataSource.SelectParameters.Add((string)Session["name" + i], (TypeCode)Session["type" + i],
(string)Session["value" + i]);
        }
        cancelDataSource.SelectMethod = (string)Session["selectMethod"];

        if (Session["filterExpr"].Equals("null"))
            cancelDataSource.FilterExpression = null;
        else
            cancelDataSource.FilterExpression = (string)Session["filterExpr"];

        GridView1.DataSourceID = "cancelDataSource";
    }

    protected void GridView1_DataBound(object sender, EventArgs e)
    {
        if (Session["selectMethod"].Equals(""))
        {
            Session["searchText"] = "";
            Session["filterText"] = "";
            foundLabel1.Text = "";
            return;
        }

        int listed = GridView1.Rows.Count;
        switch (cancelDataSource.SelectMethod)
        {
            case "GetDataByCountry":
                Session["searchText"] = listed +
                    " cancellation(s) listed for the country chosen (" +
                    Session["value0"] + ")";
                break;
            case "GetDataByVilaet":
                Session["searchText"] = listed +
                    " cancellation(s) listed for the vilaet chosen (" +
                    Session["value0"] + ")";
                break;
            case "GetDataByPostOffice":
                Session["searchText"] = listed +
                    " cancellation(s) listed for the post offices of the requested place (" +
                    Session["value0"] + ")";
                break;
            case "GetDataNegative":
                Session["searchText"] = listed +
                    " negative cancellation(s) listed";
                break;
            case "GetDataByFrenchOttoman":
                Session["searchText"] = listed +
                    " cancellation(s) listed fitting the specified ottoman "+
                    "and french text";
                break;
            case "GetDataByID":
                Session["searchText"] = listed +
                    " cancellation(s) listed with chosen id("+
                    Session["value0"] + ")";
                break;
            case "GetData":
                Session["searchText"] = listed +
                    " cancellation(s) listed";
                break;
            case "GetDataByColours":
```

```
                foundLabel1.Text = listed + (string)Session["searchText"] +
                    Session["filterText"];
                return;
                break;
            case "GetDataByColoursNumeral":
                foundLabel1.Text = listed + (string)Session["searchText"] +
                    Session["filterText"];
                return;
                break;
            default:
                Session["searchText"] = "";
                Session["filterText"] = "";
                break;
        }
        foundLabel1.Text = (string)Session["searchText"] +
                        Session["filterText"];
    }

    private static Control GetPostBackControl(Page page)
    {
        Control postbackControlInstance = null;
        string postbackControlName = page.Request.Params.Get("__EVENTTARGET");

        if (postbackControlName != null && postbackControlName != string.Empty)
        {
            postbackControlInstance = page.FindControl(postbackControlName);
        }

        else
        {
            // handle the Button control postbacks
            for (int i = 0; i < page.Request.Form.Keys.Count; i++)
            {
                postbackControlInstance = page.FindControl(page.Request.Form.Keys[i]);
                if (postbackControlInstance is System.Web.UI.WebControls.Button)
                {
                    return postbackControlInstance;
                }
            }
        }
        // handle the ImageButton postbacks
        if (postbackControlInstance == null)
        {
            for (int i = 0; i < page.Request.Form.Count; i++)
            {
                if ((page.Request.Form.Keys[i].EndsWith(".x")) || (page.Request.Form.Keys[i].EndsWith(".y")))
                {
                    postbackControlInstance = page.FindControl(page.Request.Form.Keys[i].Substring(0,
page.Request.Form.Keys[i].Length - 2));
                    return postbackControlInstance;
                }
            }
        }
        return postbackControlInstance;
    }

    protected void dateButton_Click(object sender, EventArgs e)
    {
    }
}
```

A5) αρχείο Details.aspx :

```
<%@ Page Language="C#" ValidateRequest="false" Trace="false"
AutoEventWireup="true" CodeFile="Details.aspx.cs" Inherits="Details" %>
<%@ Register TagPrefix="uc" TagName="ImageContainer" Src="~/ImageContainer.ascx" %>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml" >
<head runat="server">
   <title>details of cancellation</title>
</head>
<body style="background-color:Gray">
   <form id="form1" runat="server">
   <div>

   <table id="detailsTable" width="100%" runat="server">
   <tr>
      <td ></td>
      <td ></td>
      <td align="right" >
         <asp:Label ID="detailsIDLabel" runat="server" Text="ID"></asp:Label>
      </td>
      <td >
         <asp:TextBox ID="detailsIDTextBox" runat="server" ReadOnly="true"></asp:TextBox>
      </td>
      <td ></td>
      <td align="right" >
         <asp:Label ID="detailsDatesLabel" runat="server" Text="Dates"></asp:Label>
      </td>
      <td >
         <asp:TextBox ID="detailsDatesTextBox" runat="server" ReadOnly="true"></asp:TextBox>
      </td>
   </tr>
   <tr>
      <td colspan="2">
         <asp:TextBox ID="detailsOttoFieldTextBox" TextMode="MultiLine" Rows="5" Width="100%"
runat="server" ReadOnly="true"></asp:TextBox>
      </td>
      <td align="left">
         <asp:CheckBox ID="detailsOttoCheckBox" runat="server"/>
       </td>
      <td colspan="2">
         <asp:Label ID="detailsPOLabel" runat="server" Text="Post Office in:"></asp:Label><br/>
         <asp:TextBox ID="detailsPOTextBox" TextMode="MultiLine" Rows="4" Width="100%"
runat="server" ReadOnly="true"></asp:TextBox>
      </td>
      <td rowspan="2" colspan="2" align="center">
         <asp:Literal ID="detailsColorsLiteral" runat="server"></asp:Literal><br/>
         <uc:ImageContainer ID="imageContainer" runat="server" />
      </td>
   </tr>
   <tr>
      <td colspan="2">
         <asp:TextBox ID="detailsFrenchFieldTextBox" TextMode="MultiLine" Rows="5" Width="100%"
runat="server" ReadOnly="true"></asp:TextBox>
      </td>
      <td align="left">
```

```
            <asp:CheckBox ID="detailsFrenchCheckBox" runat="server" />
        </td>
        <td colspan="2">
            <asp:Label ID="detailsLocalNamesLabel" runat="server" Text="now in:"></asp:Label><br/>
            <asp:TextBox ID="detailsLocalNamesTextBox" TextMode="MultiLine" Rows="3" Width="100%"
runat="server" ReadOnly="true"></asp:TextBox>
        </td>
    </tr>
    <tr>
        <td align="right">
            <asp:Label ID="detailsCWLabel" runat="server" Text="CW code"></asp:Label>
        </td>
        <td>
            <asp:TextBox ID="detailsCWTextBox" runat="server" ReadOnly="true"></asp:TextBox>
        </td>
        <td></td>
        <td align="right">
            <asp:Label ID="detailsBrLabel" runat="server" Text="Br type"></asp:Label>
        </td>
        <td>
            <asp:TextBox ID="detailsBrTextBox" runat="server" ReadOnly="true"></asp:TextBox>
        </td>
        <td align="right">
            <asp:Label ID="detailsNuhoglouLabel" runat="server" Text="Nuhoglou page"></asp:Label>
        </td>
        <td>
            <asp:TextBox ID="detailsNuhoglouTextBox" runat="server" ReadOnly="true"></asp:TextBox>
        </td>
    </tr>
    <tr>
        <td align="right">
            <asp:Label ID="detailsCWPageLabel" runat="server" Text="CW page"></asp:Label>
        </td>
        <td>
            <asp:TextBox ID="detailsCWPageTextBox" runat="server" ReadOnly="true"></asp:TextBox>
        </td>
        <td></td>
        <td align="right">
            <asp:Label ID="detailsBrPageLabel" runat="server" Text="Br page"></asp:Label>
        </td>
        <td>
            <asp:TextBox ID="detailsBrPageTextBox" runat="server" ReadOnly="true"></asp:TextBox>
        </td>
        <td align="right">
            <asp:Label ID="detailsGalinosLabel" runat="server" Text="Galinos page"></asp:Label>
        </td>
        <td>
            <asp:TextBox ID="detailsGalinosTextBox" runat="server" ReadOnly="true"></asp:TextBox>
        </td>
    </tr>
    <tr>
        <td rowspan="2" colspan="5">
            <asp:TextBox ID="detailsCommentsTextBox" TextMode="MultiLine" Width="100%" runat="server"
ReadOnly="true"></asp:TextBox>
        </td>
        <td colspan="2" align="center">
            <asp:Button ID="detailsBackButton" runat="server" Text="back"
PostBackUrl="~/Thumbnails.aspx"/>
        </td>
    </tr>
```

```
    <tr>
      <td colspan="2" align="center">
        <asp:Button ID="detailsNewDataButton" runat="server" Text="enter new data"
PostBackUrl="~/NewData.aspx" />
      </td>
    </tr>
    <tr>
    <td colspan="7" >
      <br/>
      <asp:ObjectDataSource ID="detailsDataSource" runat="server"
OldValuesParameterFormatString="original_{0}" SelectMethod="GetDataByID"
TypeName="DataSet1TableAdapters.DetailsTableAdapter">
        <SelectParameters>
          <asp:Parameter Name="ID" Type="Int32" />
        </SelectParameters>
      </asp:ObjectDataSource>
    </td>
    </tr>
    <tr>
      <td colspan="7">
      <asp:GridView ID="GridView1" Width="100%" runat="server" AutoGenerateColumns="False"
DataSourceID="" BackColor="White" BorderColor="#999999" BorderStyle="Solid" BorderWidth="1px"
CellPadding="3" ForeColor="Black" GridLines="Vertical" AllowPaging="True" PageSize="50"
DataKeyNames="ID" OnSelectedIndexChanged="rowSelected" OnPageIndexChanged="pageChanged" >
        <FooterStyle BackColor="#CCCCCC" />
        <SelectedRowStyle BackColor="#000099" Font-Bold="True" ForeColor="White" />
        <PagerStyle BackColor="#999999" ForeColor="Black" HorizontalAlign="Center" />
        <HeaderStyle BackColor="Black" Font-Bold="True" ForeColor="White" />
        <AlternatingRowStyle BackColor="#CCCCCC" />
        <Columns>
          <asp:BoundField DataField="ID" HeaderText="ID" InsertVisible="False" ReadOnly="True"
            SortExpression="ID" />
          <asp:BoundField DataField="Normalised Latin" HeaderText="Normalised Latin"
SortExpression="Normalised Latin" />
          <asp:BoundField DataField="Normalised Arabic" HeaderText="Normalised Arabic"
SortExpression="Normalised Arabic" />
          <asp:BoundField DataField="Post Office" HeaderText="Post Office" SortExpression="Post Office"
/>
          <asp:BoundField DataField="CW" HeaderText="CW" SortExpression="CW" />
          <asp:BoundField DataField="Br" HeaderText="Br" SortExpression="Br" />
          <asp:ButtonField ButtonType="Button" CommandName="Select" Text="details" />
        </Columns>
      </asp:GridView>
      <asp:ObjectDataSource ID="cancelDataSource" runat="server"
OldValuesParameterFormatString="original_{0}"
        SelectMethod="GetDataByPostOffice"
TypeName="DataSet1TableAdapters.CancellationsTableAdapter" >
        <SelectParameters>
          <asp:Parameter Name="Post_Office" Type="String" />
        </SelectParameters>
      </asp:ObjectDataSource>
      </td>
    </tr>
    </table>


    </div>
    </form>
</body>
</html>
```

Α6) αρχείο Details.aspx.cs :

```
using System;
using System.Data;
using System.Configuration;
using System.Collections;
using System.Web;
using System.Web.Security;
using System.Web.UI;
using System.Web.UI.WebControls;
using System.Web.UI.WebControls.WebParts;
using System.Web.UI.HtmlControls;

public partial class Details : System.Web.UI.Page
{
   protected void Page_Load(object sender, EventArgs e)
   {
      if (!IsPostBack)
      {
         detailsDataSource.SelectParameters["ID"].DefaultValue = (string)Session["detailsID"];
         loadDetails();
      }
      restoreGridViewData();
      if(!IsPostBack)
         setSelectedRow();
   }

   private void loadDetails()
   {
      DataView dv = (DataView)detailsDataSource.Select();

      if (dv.Table.Rows[0]["Dates"] is System.DBNull)
      {
         detailsDatesLabel.Visible = false;
         detailsDatesTextBox.Visible = false;
         Session["selectedCancelDates"] = "";
      }
      else
      {
         string temp = (string)dv.Table.Rows[0]["Dates"];
         detailsDatesLabel.Visible = true;
         detailsDatesTextBox.Visible = true;
         detailsDatesTextBox.Text = temp;
         Session["selectedCancelDates"] = temp;
      }

      if (dv.Table.Rows[0]["ID"] is System.DBNull)
      {
         detailsIDLabel.Visible = false;
         detailsIDTextBox.Visible = false;
         Session["selectedCancelID"] = "";
      }
      else
      {
         int temp = (int)dv.Table.Rows[0]["ID"];
         detailsIDLabel.Visible = true;
         detailsIDTextBox.Visible = true;
         detailsIDTextBox.Text = "" + temp;
         Session["selectedCancelID"] = temp;
```

```
}

if (dv.Table.Rows[0]["CW"] is System.DBNull)
{
   detailsCWLabel.Visible = false;
   detailsCWTextBox.Visible = false;
   Session["selectedCancelCW"] = "";
}
else
{
   string temp = (string)dv.Table.Rows[0]["CW"];
   detailsCWLabel.Visible = true;
   detailsCWTextBox.Visible = true;
   detailsCWTextBox.Text = temp;
   Session["selectedCancelCW"] = temp;
   if (temp.StartsWith("S"))
   {
      detailsOttoFieldTextBox.BackColor = System.Drawing.Color.Black;
      detailsOttoFieldTextBox.ForeColor = System.Drawing.Color.White;
      detailsFrenchFieldTextBox.BackColor = System.Drawing.Color.Black;
      detailsFrenchFieldTextBox.ForeColor = System.Drawing.Color.White;
   }
   else
   {
      detailsOttoFieldTextBox.BackColor = System.Drawing.Color.White;
      detailsOttoFieldTextBox.ForeColor = System.Drawing.Color.Black;
      detailsFrenchFieldTextBox.BackColor = System.Drawing.Color.White;
      detailsFrenchFieldTextBox.ForeColor = System.Drawing.Color.Black;
   }
}

if (dv.Table.Rows[0]["CWPage"] is System.DBNull)
{
   detailsCWPageLabel.Visible = false;
   detailsCWPageTextBox.Visible = false;
   Session["selectedCancelCWPage"] = "";
}
else
{
   string temp = (string)dv.Table.Rows[0]["CWPage"];
   detailsCWPageLabel.Visible = true;
   detailsCWPageTextBox.Visible = true;
   detailsCWPageTextBox.Text = temp;
   Session["selectedCancelCWPage"] = temp;
}

if (dv.Table.Rows[0]["Br"] is System.DBNull)
{
   detailsBrLabel.Visible = false;
   detailsBrTextBox.Visible = false;
   Session["selectedCancelBr"] = "";
}
else
{
   string temp = (string)dv.Table.Rows[0]["Br"];
   detailsBrLabel.Visible = true;
   detailsBrTextBox.Visible = true;
   detailsBrTextBox.Text = temp;
   Session["selectedCancelBr"] = temp;
}
```

```
if (dv.Table.Rows[0]["BrandtPage"] is System.DBNull)
{
   detailsBrPageLabel.Visible = false;
   detailsBrPageTextBox.Visible = false;
   Session["selectedCancelBrPage"] = "";
}
else
{
   string temp = (string)dv.Table.Rows[0]["BrandtPage"];
   detailsBrPageLabel.Visible = true;
   detailsBrPageTextBox.Visible = true;
   detailsBrPageTextBox.Text = temp;
   Session["selectedCancelBrPage"] = temp;
}

if (dv.Table.Rows[0]["Nuhoglou"] is System.DBNull)
{
   detailsNuhoglouLabel.Visible = false;
   detailsNuhoglouTextBox.Visible = false;
   Session["selectedCancelNuhoglou"] = "";
}
else
{
   string temp = (string)dv.Table.Rows[0]["Nuhoglou"];
   detailsNuhoglouLabel.Visible = true;
   detailsNuhoglouTextBox.Visible = true;
   detailsNuhoglouTextBox.Text = temp;
   Session["selectedCancelNuhoglou"] = temp;
}

if (dv.Table.Rows[0]["Galinos"] is System.DBNull)
{
   detailsGalinosLabel.Visible = false;
   detailsGalinosTextBox.Visible = false;
   Session["selectedCancelGalinos"] = "";
}
else
{
   string temp = (string)dv.Table.Rows[0]["Galinos"];
   detailsGalinosLabel.Visible = true;
   detailsGalinosTextBox.Visible = true;
   detailsGalinosTextBox.Text = temp;
   Session["selectedCancelGalinos"] = temp;
}

if (dv.Table.Rows[0]["Comments"] is System.DBNull)
{
   detailsCommentsTextBox.Visible = false;
   Session["selectedCancelComments"] = "";
}
else
{
   string temp = (string)dv.Table.Rows[0]["Comments"];
   detailsCommentsTextBox.Visible = true;
   detailsCommentsTextBox.Text = temp;
   Session["selectedCancelComments"] = temp;
}

if ((bool)dv.Table.Rows[0]["FDef"])
```

```
{
    detailsFrenchCheckBox.Checked = true;
    Session["selectedCancelFDef"] = true;
}
else
{
    detailsFrenchCheckBox.Checked = false;
    Session["selectedCancelFDef"] = false;
}

if ((bool)dv.Table.Rows[0]["ODef"])
{
    detailsOttoCheckBox.Checked = true;
    Session["selectedCancelODef"] = true;
}
else
{
    detailsOttoCheckBox.Checked = false;
    Session["selectedCancelODef"] = false;
}

if (dv.Table.Rows[0]["French Field"].Equals("="))
{
    detailsFrenchFieldTextBox.Visible = false;
    detailsFrenchCheckBox.Visible = false;
    Session["selectedCancelFrenchField"] = "";
}
else
{
    string temp = (string)dv.Table.Rows[0]["French Field"];
    detailsFrenchFieldTextBox.Text = temp;
    detailsFrenchFieldTextBox.Visible = true;
    detailsFrenchCheckBox.Visible = true;
    Session["selectedCancelFrenchField"] = temp;
}

if (dv.Table.Rows[0]["Ottoman Field"].Equals("="))
{
    detailsOttoFieldTextBox.Visible = false;
    detailsOttoCheckBox.Visible = false;
    Session["selectedCancelOttoField"] = "";
}
else
{
    string temp = (string)dv.Table.Rows[0]["Ottoman Field"];
    detailsOttoFieldTextBox.Text = temp;
    detailsOttoFieldTextBox.Visible = true;
    detailsOttoCheckBox.Visible = true;
    Session["selectedCancelOttoField"] = temp;
}

Session["selectedCancelPostOffice"] = "" + dv.Table.Rows[0]["POTurk"];
string vilaet = (string)dv.Table.Rows[0]["Vilaet"];
if (vilaet.EndsWith("S."))
    detailsPOTextBox.Text = "" + dv.Table.Rows[0]["POTurk"] +
            "\n" + dv.Table.Rows[0]["POOtto"] +
            "\n" + vilaet.Substring(0, vilaet.Length - 3) +
            "\n" + "Sancak";
else if (vilaet.EndsWith("V."))
    detailsPOTextBox.Text = "" + dv.Table.Rows[0]["POTurk"] +
```

```
                "\n" + dv.Table.Rows[0]["POOtto"] +
                "\n" + vilaet.Substring(0, vilaet.Length - 3) +
                "\n" + "Vil\u03b2yeti";
        else
            detailsPOTextBox.Text = "" + dv.Table.Rows[0]["POTurk"] +
                "\n" + dv.Table.Rows[0]["POOtto"] +
                "\n" + vilaet;

        detailsLocalNamesTextBox.Text = "" + dv.Table.Rows[0]["Country"] +
                ":\n" + dv.Table.Rows[0]["ModMulti"] +
                "\n" + dv.Table.Rows[0]["ModLatin"];

        if (dv.Table.Rows[0]["Colours"] is System.DBNull)
        {
            Session["selectedCancelColours"] = "";
        }
        else
        {
            string temp = (string)dv.Table.Rows[0]["Colours"];
            detailsColorsLiteral.Text = colorsToString(temp);
            Session["selectedCancelColours"] = temp;
        }

        if (dv.Table.Rows[0]["Image"] is System.DBNull)
        {
            Session["selectedCancelImage"] = "";
            imageContainer.ImagesUrlsString = "";
        }
        else
        {
            string temp = (string)dv.Table.Rows[0]["Image"];
            imageContainer.ImagesUrlsString = temp;
            Session["selectedCancelImage"] = temp;
        }
    }

    private static string colorsToString(string col)
    {
        string[] tokens = col.Split(',');
        string results = "";

        for (int i = 0; i < tokens.Length; i++)
        {
            string temp = colorAcronymToString(tokens[i]);
            if (!temp.Equals(String.Empty))
                results += "<font color=\"" + colorAcronymToCode(tokens[i]) +
                    "\">" + temp + "</font> ";
        }
        return results;
    }
    private static string colorAcronymToString(string col)
    {
        switch (col)
        {
            case "blk":
                return "black";
            case "blu":
                return "blue";
            case "brn":
                return "brown";
```

```
            case "crm":
                return "carmine";
            case "grn":
                return "green";
            case "red":
                return "red";
            case "vlt":
                return "violet";
            default:
                return "";
        }
    }
    private static string colorAcronymToCode(string col)
    {
        switch (col)
        {
            case "blk":
                return "#000000";
            case "blu":
                return "#0000FF";
            case "brn":
                return "#A52A2A";
            case "crm":
                return "#960018";
            case "grn":
                return "#00FF00";
            case "red":
                return "#FF0000";
            case "vlt":
                return "#EE82EE";
            default:
                return "";
        }
    }

    protected void rowSelected(object sender, EventArgs e)
    {
        GridViewRow row = GridView1.SelectedRow;
        detailsDataSource.SelectParameters["ID"].DefaultValue = row.Cells[0].Text;
        loadDetails();
    }

    private void restoreGridViewData()
    {
        if (Session["selectMethod"].Equals(""))
            return;

        for (int i = cancelDataSource.SelectParameters.Count - 1; i >= 0; i--)
            cancelDataSource.SelectParameters.RemoveAt(i);
        for (int i = 0; i < (int)Session["selectParamCount"]; i++)
        {
            cancelDataSource.SelectParameters.Add((string)Session["name" + i], (TypeCode)Session["type" + i],
(string)Session["value" + i]);
        }
        cancelDataSource.SelectMethod = (string)Session["selectMethod"];

        if (Session["filterExpr"].Equals("null"))
            cancelDataSource.FilterExpression = null;
        else
            cancelDataSource.FilterExpression = (string)Session["filterExpr"];
```

```
        GridView1.DataSourceID = "cancelDataSource";
    }

    private void setSelectedRow()
    {
        GridView1.AllowPaging = false;
        int index = -1;

        for (int i = 0; i < GridView1.Rows.Count; i++)
        {
            GridViewRow row = GridView1.Rows[i];
            if (row.Cells[0].Text.Equals((string)Session["detailsID"]))
            {
                index = i;
                break;
            }
        }
        GridView1.AllowPaging = true;

        if (index != -1)
        {
            GridView1.PageIndex = index / GridView1.PageSize;
            GridView1.SelectedIndex = index % GridView1.PageSize;
        }
    }

    protected void pageChanged(object sender, EventArgs e)
    {
        GridView1.SelectedIndex = -1;
    }
}
```

A7) αρχείο EditData.aspx :

```
<%@ Page Language="C#" ValidateRequest="false" Trace="false"
AutoEventWireup="true" CodeFile="EditData.aspx.cs" Inherits="EditData" %>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml" >
<head runat="server">
    <title>Form for editing Post Offices</title>
</head>
<body style="background-color:GrayText">
    <form id="form1" runat="server">
    <div>
    <table width="100%" cellspacing="5" >
        <tr>
            <td colspan="3" align="right" >
                <asp:Label ID="titleLabel" runat="server" Text=""></asp:Label>
            </td>
            <td colspan="2" align="left" >
                <asp:Label ID="title1Label" Font-Bold="true" runat="server" Text=""></asp:Label>
            </td>
        </tr>
        <tr>
            <td colspan="5" style="height: 197px">
```

```
            <asp:GridView ID="GridView1" Width="100%" runat="server" AutoGenerateColumns="False"
BackColor="White" BorderColor="#999999" BorderStyle="Solid" BorderWidth="1px" CellPadding="3"
DataSourceID="AccessDataSource1" ForeColor="Black" GridLines="Vertical" AllowSorting="True"
OnSelectedIndexChanged="gridViewRowSelected" OnPageIndexChanged="gridView_PageChanged"
OnSorted="gridView_RowSorted">
                <FooterStyle BackColor="#CCCCCC" />
                <Columns>
                    <asp:BoundField DataField="POTurk" HeaderText="POTurk" SortExpression="POTurk" />
                    <asp:BoundField DataField="POOtto" HeaderText="POOtto" SortExpression="POOtto" />
                    <asp:BoundField DataField="ModLatin" HeaderText="ModLatin"
SortExpression="ModLatin" />
                    <asp:BoundField DataField="ModMulti" HeaderText="ModMulti"
SortExpression="ModMulti" />
                    <asp:BoundField DataField="Vilaet" HeaderText="Vilaet" SortExpression="Vilaet" />
                    <asp:BoundField DataField="Country" HeaderText="Country" SortExpression="Country" />
                    <asp:ButtonField ButtonType="Button" CommandName="Select" Text="edit" />
                </Columns>
                <SelectedRowStyle BackColor="#000099" Font-Bold="True" ForeColor="White" />
                <PagerStyle BackColor="#999999" ForeColor="Black" HorizontalAlign="Center" />
                <HeaderStyle BackColor="Black" Font-Bold="True" ForeColor="White" />
                <AlternatingRowStyle BackColor="#CCCCCC" />
            </asp:GridView>
            <asp:AccessDataSource ID="AccessDataSource1" runat="server"
DataFile="~/App_Data/Cancel_Alexandros_Apr09.mdb">
            </asp:AccessDataSource>
        </td>
    </tr>
    <tr>
        <td colspan="2">
            Ottoman Name (modern turkish, ottoman)
        </td>
        <td colspan="2">
            Modern Name (local language, local script)
        </td>
        <td>
            <asp:Label ID="vilaetCountryLabel" runat="server" Text=""></asp:Label>
        </td>
    </tr>
    <tr>
        <td style="width:20%">
            <asp:TextBox ID="otto1TextBox" Width="100%" runat="server"></asp:TextBox>
        </td>
        <td style="width:20%">
            <asp:TextBox ID="otto2TextBox" Width="100%" runat="server"></asp:TextBox>
        </td>
        <td style="width:20%">
            <asp:TextBox ID="modern1TextBox" Width="100%" runat="server"></asp:TextBox>
        </td>
        <td style="width:20%">
            <asp:TextBox ID="modern2TextBox" Width="100%" runat="server"></asp:TextBox>
        </td>
        <td style="width:20%">
            <asp:DropDownList ID="vilaetCountryDDL" Width="100%" runat="server"
OnDataBound="vilaetCountryDDL_DataBound">
            </asp:DropDownList>
            <asp:AccessDataSource ID="vilaetDataSource" runat="server"
DataFile="~/App_Data/Cancel_Alexandros_Apr09.mdb" SelectCommand="SELECT [VilaetL] FROM
[RegionOttoman] ORDER BY [VilaetL]"></asp:AccessDataSource>
```

```
            <asp:AccessDataSource ID="countryDataSource" runat="server"
DataFile="~/App_Data/Cancel_Alexandros_Apr09.mdb" SelectCommand="SELECT [Country] FROM
[Countries] ORDER BY [Country]"></asp:AccessDataSource>
          </td>
        </tr>
        <tr>
          <td colspan="5" style="color:Blue">
            Select from the list above and process on the five textboxes as indicated (instructions below*)
          </td>
        </tr>
        <tr>
          <td>
            <asp:Button ID="updateButton" runat="server" Text="update post office"
OnClick="updateButton_Click" />
          </td>
          <td>
            <asp:Button ID="createButton" runat="server" Text="enter as new post office"
OnClick="createButton_Click" />
          </td>
          <td>
            <asp:Button ID="deleteButton" runat="server" Text="delete post office"
OnClick="deleteButton_Click" />
          </td>
          <td>
            <asp:AccessDataSource ID="localNamesDSource" runat="server"
DataFile="~/App_Data/Cancel_Alexandros_Apr09.mdb"
              DeleteCommand="DELETE FROM [Local Names] WHERE (POTurk = ?)"
              InsertCommand="INSERT INTO [Local Names] (POTurk, Vilaet, Country, POOtto, ModLatin,
ModMulti) &#13;&#10;VALUES (@poturk,@vilaet,@country,@pootto,@modlatin,@modmulti)"
              SelectCommand="SELECT POTurk, Vilaet, Country, POOtto, ModLatin, ModMulti
&#13;&#10;FROM [Local Names] WHERE (POTurk = ?)"
              UpdateCommand="UPDATE [Local Names] SET POTurk=@pot, Vilaet=@vil, Country=@cou,
POOtto=@poo, ModLatin=@modl, ModMulti=@modm WHERE (POTurk = ?)">
              <DeleteParameters>
                <asp:ControlParameter ControlID="otto1TextBox" Name="POTurk" PropertyName="Text" />
              </DeleteParameters>
              <InsertParameters>
                <asp:ControlParameter ControlID="otto1TextBox" Name="poturk" PropertyName="Text" />
                <asp:Parameter DefaultValue="" Name="vilaet" />
                <asp:Parameter DefaultValue="" Name="country" />
                <asp:ControlParameter ControlID="otto2TextBox" Name="pootto" PropertyName="Text" />
                <asp:ControlParameter ControlID="modern1TextBox" Name="modlatin"
PropertyName="Text" />
                <asp:ControlParameter ControlID="modern2TextBox" Name="modmulti"
PropertyName="Text" />
              </InsertParameters>
              <SelectParameters>
                <asp:Parameter DefaultValue="" Name="POTurk" />
              </SelectParameters>
              <UpdateParameters>
                <asp:ControlParameter ControlID="otto1TextBox" Name="pot" PropertyName="Text" />
                <asp:Parameter DefaultValue="" Name="vil" />
                <asp:Parameter DefaultValue="" Name="cou" />
                <asp:ControlParameter ControlID="otto2TextBox" Name="poo" PropertyName="Text" />
                <asp:ControlParameter ControlID="modern1TextBox" Name="modl" PropertyName="Text"/>
                <asp:ControlParameter ControlID="modern2TextBox" Name="modm" PropertyName="Text"
/>
                <asp:Parameter DefaultValue="" Name="POTurk" />
              </UpdateParameters>
            </asp:AccessDataSource>
```

```
        </td>
        <td>
          <asp:Button ID="Button4" runat="server" PostBackUrl="~/Default.aspx" Text="Back To
Cancellations" />
        </td>
      </tr>
      <tr>
        <td colspan="5">
          <asp:TextBox ID="warningTextBox" BackColor="gray" ReadOnly="true" ForeColor="red"
Width="100%" runat="server"></asp:TextBox>
        </td>
      </tr>
      <tr>
        <td colspan="5">
          <br/>*<i>Press Windows key, click on Programs, Accessories, Accessibility, On-Screen Keyboard
and then switch to correct language</i> OR<br/>
          <i>Press Windows key, click on Programs, Accessories, System Tools, Character Map. Click your
letter, then click select and copy. This brings your letter(s) on the clipboard; then paste in the fields above.</i>
        </td>
      </tr>
    </table>
    </div>
    </form>
</body>
</html>
```

A8) αρχείο EditData.aspx.cs :

```
using System;
using System.Data;
using System.Configuration;
using System.Collections;
using System.Web;
using System.Web.Security;
using System.Web.UI;
using System.Web.UI.WebControls;
using System.Web.UI.WebControls.WebParts;
using System.Web.UI.HtmlControls;

public partial class EditData : System.Web.UI.Page
{
    protected void Page_Load(object sender, EventArgs e)
    {
        warningTextBox.Visible = false;

        if (!IsPostBack)
        {
            title1Label.Text = (string)Session["selectedLatin"] + " " +
                               (string)Session["selectedMulti"];
            string param = (string)Session["selectedLatin"];

            if ((bool)Session["vilaetSelected"])
            {
                title1Label.ForeColor = System.Drawing.Color.Red;
                vilaetCountryLabel.ForeColor = System.Drawing.Color.Blue;
                vilaetCountryLabel.Text = "Country";

                AccessDataSource1.SelectCommand = "SELECT [POTurk], [POOtto], [ModLatin], [ModMulti],
[Vilaet], [Country] FROM [Local Names] WHERE ([Vilaet] = '" + param + "')";
```

```
            ViewState["selectCommand"] = "SELECT [POTurk], [POOtto], [ModLatin], [ModMulti], [Vilaet],
[Country] FROM [Local Names] WHERE ([Vilaet] = '" + param + "')";
            GridView1.DataBind();
            vilaetCountryDDL.DataSourceID = "countryDataSource";
            vilaetCountryDDL.DataTextField = "Country";
        }
        else if ((bool)Session["countrySelected"])
        {
            title1Label.ForeColor = System.Drawing.Color.Blue;
            vilaetCountryLabel.ForeColor = System.Drawing.Color.Red;
            vilaetCountryLabel.Text = "Vilaet";

            AccessDataSource1.SelectCommand = "SELECT [POTurk], [POOtto], [ModLatin], [ModMulti],
[Vilaet], [Country] FROM [Local Names] WHERE ([Country] = '" + param + "')";
            ViewState["selectCommand"] = "SELECT [POTurk], [POOtto], [ModLatin], [ModMulti], [Vilaet],
[Country] FROM [Local Names] WHERE ([Country] = '" + param + "')";
            GridView1.DataBind();
            vilaetCountryDDL.DataSourceID = "vilaetDataSource";
            vilaetCountryDDL.DataTextField = "VilaetL";
        }

        ViewState["rowCounter"] = "" + GridView1.Rows.Count;
        titleLabel.Text = "current list of Post Offices - " + ViewState["rowCounter"] + " items - in ";
        GridView1.AllowPaging = true;
    }
    else
    {
        AccessDataSource1.SelectCommand = (string)ViewState["selectCommand"];
        GridView1.DataBind();
    }
}

protected void vilaetCountryDDL_DataBound(object sender, EventArgs e)
{
    vilaetCountryDDL.Items.Remove("*");
}

protected void gridViewRowSelected(object sender, EventArgs e)
{
    GridViewRow row = GridView1.SelectedRow;

    otto1TextBox.Text = HttpUtility.HtmlDecode(row.Cells[0].Text);
    otto2TextBox.Text = HttpUtility.HtmlDecode(row.Cells[1].Text);
    modern1TextBox.Text = HttpUtility.HtmlDecode(row.Cells[2].Text);
    modern2TextBox.Text = HttpUtility.HtmlDecode(row.Cells[3].Text);

    if ((bool)Session["vilaetSelected"])
    {
        vilaetCountryDDL.SelectedValue = HttpUtility.HtmlDecode(row.Cells[5].Text);
    }
    else if ((bool)Session["countrySelected"])
    {
        vilaetCountryDDL.SelectedValue = HttpUtility.HtmlDecode(row.Cells[4].Text);
    }
}

protected void updateButton_Click(object sender, EventArgs e)
{
    if (isEmptyPOTextBox())
        return;
```

```
        if (!existsPostOffice(otto1TextBox.Text))
        {
            warningTextBox.Visible = true;
            warningTextBox.Text =
                "no such post office exists. if you want to create it, use the enter new post office button";
            return;
        }

        if ((bool)Session["vilaetSelected"])
        {
            localNamesDSource.UpdateParameters["vil"].DefaultValue = (string)Session["selectedLatin"];
            localNamesDSource.UpdateParameters["cou"].DefaultValue = vilaetCountryDDL.SelectedValue;
        }
        else if ((bool)Session["countrySelected"])
        {
            localNamesDSource.UpdateParameters["vil"].DefaultValue = vilaetCountryDDL.SelectedValue;
            localNamesDSource.UpdateParameters["cou"].DefaultValue = (string)Session["selectedLatin"];
        }

        localNamesDSource.UpdateParameters["POTurk"].DefaultValue = otto1TextBox.Text;
        localNamesDSource.Update();
        GridView1.DataBind();
    }
    protected void createButton_Click(object sender, EventArgs e)
    {
        if (isEmptyPOTextBox())
            return;
        if (existsPostOffice(otto1TextBox.Text))
        {
            warningTextBox.Visible = true;
            warningTextBox.Text = "post office already exists. if you want to change its properties, use the update
button";
            return;
        }

        if ( (bool)Session["vilaetSelected"] )
        {
            localNamesDSource.InsertParameters["vilaet"].DefaultValue = (string)Session["selectedLatin"];
            localNamesDSource.InsertParameters["country"].DefaultValue = vilaetCountryDDL.SelectedValue;
        }
        else if ( (bool)Session["countrySelected"] )
        {
            localNamesDSource.InsertParameters["vilaet"].DefaultValue = vilaetCountryDDL.SelectedValue;
            localNamesDSource.InsertParameters["country"].DefaultValue = (string)Session["selectedLatin"];
        }

        localNamesDSource.Insert();
        increaseRowsCounter();
        GridView1.DataBind();
    }
    protected void deleteButton_Click(object sender, EventArgs e)
    {
        if (isEmptyPOTextBox())
            return;
        if ( !existsPostOffice(otto1TextBox.Text) )
            return;
        localNamesDSource.Delete();
        decreaseRowsCounter();
        GridView1.DataBind();
    }
```

```
    private bool existsPostOffice(string poturk)
    {
        localNamesDSource.SelectParameters["POTurk"].DefaultValue = poturk;
        DataView dv = (DataView)localNamesDSource.Select(DataSourceSelectArguments.Empty);
        if (dv.Table.Rows.Count > 0)
            return true;
        else
            return false;
    }
    private bool isEmptyPOTextBox()
    {
        if (otto1TextBox.Text.Trim().Equals(string.Empty))
        {
            warningTextBox.Visible = true;
            warningTextBox.Text = "ottoman name field (in modern turkish) cannot be empty";
            return true;
        }
        return false;
    }

    private void increaseRowsCounter()
    {
        int i = Int32.Parse( (string)ViewState["rowCounter"] );
        ViewState["rowCounter"] = "" + (i + 1);
        titleLabel.Text = "current list of Post Offices - " + ViewState["rowCounter"] + " items - in ";
    }
    private void decreaseRowsCounter()
    {
        int i = Int32.Parse( (string)ViewState["rowCounter"] );
        ViewState["rowCounter"] = "" + (i - 1);
        titleLabel.Text = "current list of Post Offices - " + ViewState["rowCounter"] + " items - in ";
    }

    protected void gridView_PageChanged(object sender, EventArgs e)
    {
        GridView1.SelectedIndex = -1;
    }
    protected void gridView_RowSorted(object sender, EventArgs e)
    {
        GridView1.SelectedIndex = -1;
    }
}
```

A9) αρχείο expressionSearch.aspx :

```
<%@ Page Language="C#" AutoEventWireup="true" CodeFile="expressionSearch.aspx.cs"
Inherits="expressionSearch" %>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml" >
<head runat="server">
    <title>search based on expression</title>
</head>
<body style="background-color:Gray">
    <form id="form1" runat="server">
```

```
    <div>

    <table width="100%">
    <tr>
        <td colspan="10">
            <asp:Label ID="title" Font-Size="Large" Font-Bold="true" runat="server" Text="Search based on the
placement of Ottoman Post expressions on the cancellation"></asp:Label>
        </td>
        <td></td>
        <td colspan="2">
            <asp:Button ID="exprBackButton" runat="server" Text="back to search"
PostBackUrl="~/Default.aspx" />
        </td>
        <td></td>
    </tr>
    <tr>
        <td colspan="14"><br /></td>
    </tr>
    <tr align="center">
        <td></td>
        <td></td>
        <td colspan="2">
            <asp:Label ID="anCanibLabel" BackColor="black" ForeColor="white" Font-Bold="true"
Width="100%" runat="server" Text="an canib"></asp:Label>
        </td>
        <td></td>
        <td></td>
        <td colspan="2">
            <asp:Label ID="telegrafLabel" BackColor="black" ForeColor="white" Font-Bold="true"
Width="100%" runat="server" Text="telegraf"></asp:Label>
        </td>
        <td></td>
        <td></td>
        <td colspan="2">
            <asp:Label ID="postaLabel" BackColor="black" ForeColor="white" Font-Bold="true" Width="100%"
runat="server" Text="posta"></asp:Label>
        </td>
        <td></td>
        <td></td>
    </tr>
    <tr>
        <td align="right">
            top<br />
            middle<br />
            bottom<br />
            unclear
        </td>
        <td align="right">
            <asp:RadioButtonList ID="anCanibRBL1" runat="server">
                <asp:ListItem Value="top" Text=""></asp:ListItem>
                <asp:ListItem Value="middle" Text=""></asp:ListItem>
                <asp:ListItem Value="bottom" Text=""></asp:ListItem>
                <asp:ListItem Value="unclear" Text="" Selected="True"></asp:ListItem>
            </asp:RadioButtonList>
        </td>
        <td>
            <asp:ImageButton ID="anCanibImage1" runat="server" ToolTip="click for other variants of this
expression" OnClick="image_Click" />
        </td>
        <td align="right">
```

```
        <asp:ImageButton ID="anCanibImage2" runat="server" ToolTip="click for other variants of this
expression" OnClick="image_Click" />
      </td>
      <td align="left">
        <asp:RadioButtonList ID="anCanibRBL2" runat="server">
          <asp:ListItem Value="top" Text=""></asp:ListItem>
          <asp:ListItem Value="middle" Text=""></asp:ListItem>
          <asp:ListItem Value="bottom" Text=""></asp:ListItem>
          <asp:ListItem Value="unclear" Text="" Selected="True"></asp:ListItem>
        </asp:RadioButtonList>
      </td>
      <td align="right">
        <asp:RadioButtonList ID="telegrafRBL1" runat="server">
          <asp:ListItem Value="top" Text=""></asp:ListItem>
          <asp:ListItem Value="middle" Text=""></asp:ListItem>
          <asp:ListItem Value="bottom" Text=""></asp:ListItem>
          <asp:ListItem Value="unclear" Text="" Selected="True"></asp:ListItem>
        </asp:RadioButtonList>
      </td>
      <td>
        <asp:ImageButton ID="telegrafImage1" runat="server" ToolTip="click for other variants of this
expression" OnClick="image_Click" />
      </td>
      <td align="right">
        <asp:ImageButton ID="telegrafImage2" runat="server" ToolTip="click for other variants of this
expression" OnClick="image_Click" />
      </td>
      <td align="left">
        <asp:RadioButtonList ID="telegrafRBL2" runat="server">
          <asp:ListItem Value="top" Text=""></asp:ListItem>
          <asp:ListItem Value="middle" Text=""></asp:ListItem>
          <asp:ListItem Value="bottom" Text=""></asp:ListItem>
          <asp:ListItem Value="unclear" Text="" Selected="True"></asp:ListItem>
        </asp:RadioButtonList>
      </td>
      <td align="right">
        <asp:RadioButtonList ID="postaRBL1" runat="server">
          <asp:ListItem Value="top" Text=""></asp:ListItem>
          <asp:ListItem Value="middle" Text=""></asp:ListItem>
          <asp:ListItem Value="bottom" Text=""></asp:ListItem>
          <asp:ListItem Value="unclear" Text="" Selected="True"></asp:ListItem>
        </asp:RadioButtonList>
      </td>
      <td>
        <asp:ImageButton ID="postaImage1" runat="server" ToolTip="click for other variants of this
expression" OnClick="image_Click" />
      </td>
      <td align="right">
        <asp:ImageButton ID="postaImage2" runat="server" ToolTip="click for other variants of this
expression" OnClick="image_Click" />
      </td>
      <td  align="left">
        <asp:RadioButtonList ID="postaRBL2" runat="server">
          <asp:ListItem Value="top" Text=""></asp:ListItem>
          <asp:ListItem Value="middle" Text=""></asp:ListItem>
          <asp:ListItem Value="bottom" Text=""></asp:ListItem>
          <asp:ListItem Value="unclear" Text="" Selected="True"></asp:ListItem>
        </asp:RadioButtonList>
      </td>
      <td>
```

```
            top<br />
            middle<br />
            bottom<br />
            unclear
        </td>
    </tr>
    <tr align="center">
        <td>
        </td>
        <td>
        </td>
        <td colspan="2">
            <asp:RadioButtonList ID="anCanibExistRBL" AutoPostBack="true"
OnSelectedIndexChanged="anCanibExist_Changed" runat="server">
                <asp:ListItem Selected="True">existent</asp:ListItem>
                <asp:ListItem>nonexistent</asp:ListItem>
            </asp:RadioButtonList>
        </td>
        <td>
        </td>
        <td>
        </td>
        <td colspan="2">
            <asp:RadioButtonList ID="telegrafExistRBL" runat="server" AutoPostBack="True"
OnSelectedIndexChanged="telegrafExist_Changed">
                <asp:ListItem Selected="True">existent</asp:ListItem>
                <asp:ListItem>nonexistent</asp:ListItem>
            </asp:RadioButtonList>
        </td>
        <td>
        </td>
        <td>
        </td>
        <td colspan="2">
            <asp:RadioButtonList ID="postaExistRBL" runat="server" AutoPostBack="True"
OnSelectedIndexChanged="postaExist_Changed">
                <asp:ListItem Selected="True">existent</asp:ListItem>
                <asp:ListItem>nonexistent</asp:ListItem>
            </asp:RadioButtonList>
        </td>
        <td>
        </td>
        <td>
        </td>
    </tr>
    <tr>
        <td colspan="14"><br /></td>
    </tr>
    <tr align="center">
        <td></td>
        <td></td>
        <td colspan="2">
            <asp:Label ID="subesiLabel" BackColor="black" ForeColor="white" Font-Bold="true"
Width="100%" runat="server" Text="subesi"></asp:Label>
        </td>
        <td></td>
        <td></td>
        <td colspan="2">
            <asp:Label ID="hanesiLabel" BackColor="black" ForeColor="white" Font-Bold="true"
Width="100%" runat="server" Text="hane(si)"></asp:Label>
```

```
      </td>
      <td></td>
      <td></td>
      <td colspan="2">
        <asp:Label ID="numeralsLabel" BackColor="black" ForeColor="white" Font-Bold="true"
Width="100%" runat="server" Text="Numerals"
          ToolTip="search taking also into account the hindi numeral on a negative cancellation. Use hindi or
arabic textbox below">
        </asp:Label>
      </td>
      <td></td>
      <td></td>
    </tr>
    <tr>
      <td align="right">
        top<br />
        middle<br />
        bottom<br />
        unclear
      </td>
      <td align="right">
        <asp:RadioButtonList ID="subesiRBL1" runat="server">
          <asp:ListItem Value="top" Text=""></asp:ListItem>
          <asp:ListItem Value="middle" Text=""></asp:ListItem>
          <asp:ListItem Value="bottom" Text=""></asp:ListItem>
          <asp:ListItem Value="unclear" Text="" Selected="True"></asp:ListItem>
        </asp:RadioButtonList>
      </td>
      <td>
        <asp:ImageButton ID="subesiImage1" runat="server" ToolTip="click for other variants of this
expression" OnClick="image_Click" />
      </td>
      <td align="right">
        <asp:ImageButton ID="subesiImage2" runat="server" ToolTip="click for other variants of this
expression" OnClick="image_Click" />
      </td>
      <td align="left">
        <asp:RadioButtonList ID="subesiRBL2" runat="server">
          <asp:ListItem Value="top" Text=""></asp:ListItem>
          <asp:ListItem Value="middle" Text=""></asp:ListItem>
          <asp:ListItem Value="bottom" Text=""></asp:ListItem>
          <asp:ListItem Value="unclear" Text="" Selected="True"></asp:ListItem>
        </asp:RadioButtonList>
      </td>
      <td align="right">
        <asp:RadioButtonList ID="hanesiRBL1" runat="server">
          <asp:ListItem Value="top" Text=""></asp:ListItem>
          <asp:ListItem Value="middle" Text=""></asp:ListItem>
          <asp:ListItem Value="bottom" Text=""></asp:ListItem>
          <asp:ListItem Value="unclear" Text="" Selected="True"></asp:ListItem>
        </asp:RadioButtonList>
      </td>
      <td>
        <asp:ImageButton ID="hanesiImage1" runat="server" ToolTip="click for other variants of this
expression" OnClick="image_Click" />
      </td>
      <td align="right">
        <asp:ImageButton ID="hanesiImage2" runat="server" ToolTip="click for other variants of this
expression" OnClick="image_Click" />
      </td>
```

```
      <td align="left">
        <asp:RadioButtonList ID="hanesiRBL2" runat="server">
          <asp:ListItem Value="top" Text=""></asp:ListItem>
          <asp:ListItem Value="middle" Text=""></asp:ListItem>
          <asp:ListItem Value="bottom" Text=""></asp:ListItem>
          <asp:ListItem Value="unclear" Text="" Selected="True"></asp:ListItem>
        </asp:RadioButtonList>
      </td>
      <td></td>
      <td colspan="2">
        <asp:TextBox ID="hindiTextBox" BackColor="black" ForeColor="white" runat="server"
ToolTip="optionally enter clearly identifiable numerals using '%' for unclear ones (e.g.
1%04)"></asp:TextBox>hindi<br />
        <asp:TextBox ID="arabicTextBox" BackColor="black" ForeColor="white" runat="server"
ToolTip="optionally enter clearly identifiable numerals using '%' for unclear ones (e.g.
1%04)"></asp:TextBox>arabic
      </td>
      <td></td>
      <td></td>
    </tr>
    <tr align="center">
      <td></td>
      <td></td>
      <td colspan="2">
        <asp:RadioButtonList ID="subesiExistRBL" AutoPostBack="true"
OnSelectedIndexChanged="subesiExist_Changed" runat="server">
          <asp:ListItem Selected="True">existent</asp:ListItem>
          <asp:ListItem>nonexistent</asp:ListItem>
        </asp:RadioButtonList>
      </td>
      <td></td>
      <td></td>
      <td colspan="2">
        <asp:RadioButtonList ID="hanesiExistRBL" runat="server" AutoPostBack="True"
OnSelectedIndexChanged="hanesiExist_Changed">
          <asp:ListItem Selected="True">existent</asp:ListItem>
          <asp:ListItem>nonexistent</asp:ListItem>
        </asp:RadioButtonList>
      </td>
      <td></td>
      <td></td>
      <td colspan="2">
        <asp:Button ID="exprSearchButton" runat="server" Text="search" OnClick="exprSearch_Click" />
      </td>
      <td></td>
      <td></td>
    </tr>
    </table>

    </div>
    </form>
</body>
</html>
```

A10) αρχείο expressionSearch.aspx.cs :

```
using System;
using System.Data;
```

```
using System.Configuration;
using System.Collections;
using System.Web;
using System.Web.Security;
using System.Web.UI;
using System.Web.UI.WebControls;
using System.Web.UI.WebControls.WebParts;
using System.Web.UI.HtmlControls;
using System.IO;

public partial class expressionSearch : System.Web.UI.Page
{
   protected void Page_Load(object sender, EventArgs e)
   {
      if (!IsPostBack)
      {
         if (Session["shapeSearchString"] == null)
         {
            Session["shapeSearchString"] = "%";
            Session["shapeSearchText"] = " negative cancellation(s) " +
               "listed with unspecified shape and";
         }

         anCanibImage1.ImageUrl = "Images/NegSamples/an/an1.bmp";
         anCanibImage2.ImageUrl = "Images/NegSamples/janib/janib1.bmp";
         telegrafImage1.ImageUrl = "Images/NegSamples/telegr/telegr1.bmp";
         telegrafImage2.ImageUrl = "Images/NegSamples/af/af1.bmp";
         postaImage1.ImageUrl = "Images/NegSamples/po/po1.bmp";
         postaImage2.ImageUrl = "Images/NegSamples/sta/sta1.bmp";
         subesiImage1.ImageUrl = "Images/NegSamples/sube/sube1.bmp";
         subesiImage2.ImageUrl = "Images/NegSamples/si/si1.bmp";
         hanesiImage1.ImageUrl = "Images/NegSamples/ha/ha1.bmp";
         hanesiImage2.ImageUrl = "Images/NegSamples/neyi/neyi1.bmp";
         ViewState["an1"] = 1;
         ViewState["an2"] = 1;
         ViewState["tel1"] = 1;
         ViewState["tel2"] = 1;
         ViewState["post1"] = 1;
         ViewState["post2"] = 1;
         ViewState["sub1"] = 1;
         ViewState["sub2"] = 1;
         ViewState["han1"] = 1;
         ViewState["han2"] = 1;
         ViewState["prefix"] = Server.MapPath("Images/NegSamples/");
      }
   }

   protected void anCanibExist_Changed(object sender, EventArgs e)
   {
      if (anCanibExistRBL.SelectedIndex == 0)
      {
         anCanibRBL1.Visible = true;
         anCanibRBL2.Visible = true;
      }
      else
      {
         anCanibRBL1.Visible = false;
         anCanibRBL2.Visible = false;
      }
```

```
}
protected void telegrafExist_Changed(object sender, EventArgs e)
{

   if (telegrafExistRBL.SelectedIndex == 0)
   {
      telegrafRBL1.Visible = true;
      telegrafRBL2.Visible = true;
   }
   else
   {
      telegrafRBL1.Visible = false;
      telegrafRBL2.Visible = false;
   }
}
protected void postaExist_Changed(object sender, EventArgs e)
{
   if (postaExistRBL.SelectedIndex == 0)
   {
      postaRBL1.Visible = true;
      postaRBL2.Visible = true;
   }
   else
   {
      postaRBL1.Visible = false;
      postaRBL2.Visible = false;
   }
}
protected void subesiExist_Changed(object sender, EventArgs e)
{
   if (subesiExistRBL.SelectedIndex == 0)
   {
      subesiRBL1.Visible = true;
      subesiRBL2.Visible = true;
   }
   else
   {
      subesiRBL1.Visible = false;
      subesiRBL2.Visible = false;
   }
}
protected void hanesiExist_Changed(object sender, EventArgs e)
{
   if (hanesiExistRBL.SelectedIndex == 0)
   {
      hanesiRBL1.Visible = true;
      hanesiRBL2.Visible = true;
   }
   else
   {
      hanesiRBL1.Visible = false;
      hanesiRBL2.Visible = false;
   }
}

protected void image_Click(object sender, ImageClickEventArgs e)
{
   int i;
   string addr;
   ImageButton ib = (ImageButton)sender;
```

```
switch (ib.ID)
{
   case "anCanibImage1":
      i = (int)ViewState["an1"] + 1;
      addr = ViewState["prefix"] + "an/an" + i +".bmp";
      if(File.Exists(addr))
      {
         anCanibImage1.ImageUrl = "Images/NegSamples/an/an" +
               i + ".bmp" ;
         ViewState["an1"] = i;
      }
      else
      {
         anCanibImage1.ImageUrl = "Images/NegSamples/an/an1.bmp";
         ViewState["an1"] = 1;
      }
      break;
   case "anCanibImage2":
      i = (int)ViewState["an2"] + 1;
      addr = ViewState["prefix"] + "janib/janib" + i + ".bmp";
      if (File.Exists(addr))
      {
         anCanibImage2.ImageUrl = "Images/NegSamples/janib/janib" +
               i + ".bmp";
         ViewState["an2"] = i;
      }
      else
      {
         anCanibImage2.ImageUrl = "Images/NegSamples/janib/janib1.bmp";
         ViewState["an2"] = 1;
      }
      break;
   case "telegrafImage1":
      i = (int)ViewState["tel1"] + 1;
      addr = ViewState["prefix"] + "telegr/telegr" + i + ".bmp";
      if (File.Exists(addr))
      {
         telegrafImage1.ImageUrl = "Images/NegSamples/telegr/telegr" +
               i + ".bmp";
         ViewState["tel1"] = i;
      }
      else
      {
         telegrafImage1.ImageUrl = "Images/NegSamples/telegr/telegr1.bmp";
         ViewState["tel1"] = 1;
      }
      break;
   case "telegrafImage2":
      i = (int)ViewState["tel2"] + 1;
      addr = ViewState["prefix"] + "af/af" + i + ".bmp";
      if (File.Exists(addr))
      {
         telegrafImage2.ImageUrl = "Images/NegSamples/af/af" +
               i + ".bmp";
         ViewState["tel2"] = i;
      }
      else
      {
         telegrafImage2.ImageUrl = "Images/NegSamples/af/af1.bmp";
         ViewState["tel2"] = 1;
```

```
        }
        break;
    case "postaImage1":
        i = (int)ViewState["post1"] + 1;
        addr = ViewState["prefix"] + "po/po" + i + ".bmp";
        if (File.Exists(addr))
        {
            postaImage1.ImageUrl = "Images/NegSamples/po/po" +
                i + ".bmp";
            ViewState["post1"] = i;
        }
        else
        {
            postaImage1.ImageUrl = "Images/NegSamples/po/po1.bmp";
            ViewState["post1"] = 1;
        }
        break;
    case "postaImage2":
        i = (int)ViewState["post2"] + 1;
        addr = ViewState["prefix"] + "sta/sta" + i + ".bmp";
        if (File.Exists(addr))
        {
            postaImage2.ImageUrl = "Images/NegSamples/sta/sta" +
                i + ".bmp";
            ViewState["post2"] = i;
        }
        else
        {
            postaImage2.ImageUrl = "Images/NegSamples/sta/sta1.bmp";
            ViewState["post2"] = 1;
        }
        break;
    case "subesiImage1":
        i = (int)ViewState["sub1"] + 1;
        addr = ViewState["prefix"] + "sube/sube" + i + ".bmp";
        if (File.Exists(addr))
        {
            subesiImage1.ImageUrl = "Images/NegSamples/sube/sube" +
                i + ".bmp";
            ViewState["sub1"] = i;
        }
        else
        {
            subesiImage1.ImageUrl = "Images/NegSamples/sube/sube1.bmp";
            ViewState["sub1"] = 1;
        }
        break;
    case "subesiImage2":
        i = (int)ViewState["sub2"] + 1;
        addr = ViewState["prefix"] + "si/si" + i + ".bmp";
        if (File.Exists(addr))
        {
            subesiImage2.ImageUrl = "Images/NegSamples/si/si" +
                i + ".bmp";
            ViewState["sub2"] = i;
        }
        else
        {
            subesiImage2.ImageUrl = "Images/NegSamples/si/si1.bmp";
            ViewState["sub2"] = 1;
```

```
            }
            break;
        case "hanesiImage1":
            i = (int)ViewState["han1"] + 1;
            addr = ViewState["prefix"] + "ha/ha" + i + ".bmp";
            if (File.Exists(addr))
            {
                hanesiImage1.ImageUrl = "Images/NegSamples/ha/ha" +
                    i + ".bmp";
                ViewState["han1"] = i;
            }
            else
            {
                hanesiImage1.ImageUrl = "Images/NegSamples/ha/ha1.bmp";
                ViewState["han1"] = 1;
            }
            break;
        case "hanesiImage2":
            i = (int)ViewState["han2"] + 1;
            addr = ViewState["prefix"] + "neyi/neyi" + i + ".bmp";
            if (File.Exists(addr))
            {
                hanesiImage2.ImageUrl = "Images/NegSamples/neyi/neyi" +
                    i + ".bmp";
                ViewState["han2"] = i;
            }
            else
            {
                hanesiImage2.ImageUrl = "Images/NegSamples/neyi/neyi1.bmp";
                ViewState["han2"] = 1;
            }
            break;
    }
}

protected void exprSearch_Click(object sender, EventArgs e)
{
    string expr = scanExpression();

    ArabicHindiUpdate();
    string numeral = wildcarded_ot(hindiTextBox.Text);
    if (numeral.Equals(string.Empty))
        numeral = "%";

    string shape = (string)Session["shapeSearchString"];

    Session["searchText"] = (string)Session["shapeSearchText"] +
        " (with the identified standard expression(s) )";
    Session["filterText"] = "";

    setSourceAsColoursNumeral(shape,expr,numeral);
    Response.Redirect("Default.aspx");
}

private string scanExpression()
{
    string temp = "2";
    if (anCanibExistRBL.SelectedIndex == 1)
        temp += "00";
    else
```

```
{
    switch(anCanibRBL1.SelectedIndex)
    {
        case 0:
            temp += "[tw]";
            break;
        case 1:
            temp += "[mw]";
            break;
        case 2:
            temp += "[bw]";
            break;
        case 3:
            temp += "[tmbw]";
            break;
    }
    switch (anCanibRBL2.SelectedIndex)
    {
        case 0:
            temp += "[tw]";
            break;
        case 1:
            temp += "[mw]";
            break;
        case 2:
            temp += "[bw]";
            break;
        case 3:
            temp += "[tmbw]";
            break;
    }
}
if (telegrafExistRBL.SelectedIndex == 1)
    temp += "00";
else
{
    switch (telegrafRBL1.SelectedIndex)
    {
        case 0:
            temp += "[tw]";
            break;
        case 1:
            temp += "[mw]";
            break;
        case 2:
            temp += "[bw]";
            break;
        case 3:
            temp += "[tmbw]";
            break;
    }
    switch (telegrafRBL2.SelectedIndex)
    {
        case 0:
            temp += "[tw]";
            break;
        case 1:
            temp += "[mw]";
            break;
        case 2:
```

```
            temp += "[bw]";
            break;
        case 3:
            temp += "[tmbw]";
            break;
    }
}
if (postaExistRBL.SelectedIndex == 1)
    temp += "00";
else
{
    switch (postaRBL1.SelectedIndex)
    {
        case 0:
            temp += "[tw]";
            break;
        case 1:
            temp += "[mw]";
            break;
        case 2:
            temp += "[bw]";
            break;
        case 3:
            temp += "[tmbw]";
            break;
    }
    switch (postaRBL2.SelectedIndex)
    {
        case 0:
            temp += "[tw]";
            break;
        case 1:
            temp += "[mw]";
            break;
        case 2:
            temp += "[bw]";
            break;
        case 3:
            temp += "[tmbw]";
            break;
    }
}
if (subesiExistRBL.SelectedIndex == 1)
    temp += "00";
else
{
    switch (subesiRBL1.SelectedIndex)
    {
        case 0:
            temp += "[tw]";
            break;
        case 1:
            temp += "[mw]";
            break;
        case 2:
            temp += "[bw]";
            break;
        case 3:
            temp += "[tmbw]";
            break;
```

```
        }
        switch (subesiRBL2.SelectedIndex)
        {
            case 0:
                temp += "[tw]";
                break;
            case 1:
                temp += "[mw]";
                break;
            case 2:
                temp += "[bw]";
                break;
            case 3:
                temp += "[tmbw]";
                break;
        }
    }
    if (hanesiExistRBL.SelectedIndex == 1)
        temp += "00";
    else
    {
        switch (hanesiRBL1.SelectedIndex)
        {
            case 0:
                temp += "[tw]";
                break;
            case 1:
                temp += "[mw]";
                break;
            case 2:
                temp += "[bw]";
                break;
            case 3:
                temp += "[tmbw]";
                break;
        }
        switch (hanesiRBL2.SelectedIndex)
        {
            case 0:
                temp += "[tw]";
                break;
            case 1:
                temp += "[mw]";
                break;
            case 2:
                temp += "[bw]";
                break;
            case 3:
                temp += "[tmbw]";
                break;
        }
    }
    return temp;
}

private void setSourceAsColoursNumeral(string shape, string expr, string numeral)
{
    Session["selectMethod"] = "GetDataByColoursNumeral";
    Session["name0"] = "Shape";
    Session["type0"] = TypeCode.String;
```

```
    Session["value0"] = shape;
    Session["name1"] = "Expression";
    Session["type1"] = TypeCode.String;
    Session["value1"] = expr;
    Session["name2"] = "Numeral";
    Session["type2"] = TypeCode.String;
    Session["value2"] = numeral;
    Session["selectParamCount"] = 3;
    Session["filterExpr"] = "null";
}

private string strNumWesternised(string s)
{
    // hindi(1632,1633,...,1641) -> arabic(48,49,...,57)
    s = s.Replace('\u0660', '\u0030');
    s = s.Replace('\u0661', '\u0031');
    s = s.Replace('\u0662', '\u0032');
    s = s.Replace('\u0663', '\u0033');
    s = s.Replace('\u0664', '\u0034');
    s = s.Replace('\u0665', '\u0035');
    s = s.Replace('\u0666', '\u0036');
    s = s.Replace('\u0667', '\u0037');
    s = s.Replace('\u0668', '\u0038');
    s = s.Replace('\u0669', '\u0039');
    return s;
}
private string strNumArabised(string s)
{
    // arabic(48,49,...,57) -> hindi(1632,1633,...,1641)
    s = s.Replace('\u0030', '\u0660');
    s = s.Replace('\u0031', '\u0661');
    s = s.Replace('\u0032', '\u0662');
    s = s.Replace('\u0033', '\u0663');
    s = s.Replace('\u0034', '\u0664');
    s = s.Replace('\u0035', '\u0665');
    s = s.Replace('\u0036', '\u0666');
    s = s.Replace('\u0037', '\u0667');
    s = s.Replace('\u0038', '\u0668');
    s = s.Replace('\u0039', '\u0669');
    return s;
}
private string wildcarded_ot(string s)
{
    int inside = 0;
    string hexp = "";
    string wildCardedText = "";

    for (int i = 0; i < s.Length; i++)
    {
        hexp = s.Substring(i, 1);
        switch (hexp)
        {
            case "!":
            case "=":
            case " ":
            case "%":
            case "_":
                break;
            case "[":
                if (inside == 0) hexp = "[\u061b";
```

```
                else hexp = "";
                inside++;
                break;
            case "]":
                if (inside > 1) hexp = "";
                inside--;
                break;
            case "0":
            case "1":
            case "2":
            case "3":
            case "4":
            case "5":
            case "6":
            case "7":
            case "8":
            case "9":
                if (inside == 0) hexp = "[\u066b" + hexp + "]";
                break;
            case "\u0660":
            case "\u0661":
            case "\u0662":
            case "\u0663":
            case "\u0664":
            case "\u0665":
            case "\u0666":
            case "\u0667":
            case "\u0668":
            case "\u0669":
                if (inside == 0)
                    hexp = "[\u066b" + hexp + "]";
                break;
            default:
                if (inside == 0)
                    hexp = "[\u061b" + hexp + "]";
                break;
        }
        wildCardedText += hexp;
    }
    return wildCardedText;
}
private void ArabicHindiUpdate()
{
    if (!arabicTextBox.Text.Equals(string.Empty))
    {
        hindiTextBox.Text = strNumArabised(arabicTextBox.Text);
        arabicTextBox.Text = strNumWesternised(hindiTextBox.Text);
    }
    else
    {
        arabicTextBox.Text = strNumWesternised(hindiTextBox.Text);
        hindiTextBox.Text = strNumArabised(arabicTextBox.Text);
    }
}
}
```

A11) αρχείο ImageContainer.ascx :

```
<%@ Control Language="C#" AutoEventWireup="true" CodeFile="ImageContainer.ascx.cs"
Inherits="ImageContainer" %>

<asp:Image ID="cancelImage" runat="server" />
<br/>
<asp:LinkButton ID="prevButton" runat="server" Text="previous" OnClick="prevButton_Click" />
<asp:LinkButton ID="nextButton" runat="server" Text="next" OnClick="nextButton_Click" />
```

A12) αρχείο ImageContainer.ascx.cs :

```
using System;
using System.Data;
using System.Configuration;
using System.Collections;
using System.IO;
using System.Web;
using System.Web.Security;
using System.Web.UI;
using System.Web.UI.WebControls;
using System.Web.UI.WebControls.WebParts;
using System.Web.UI.HtmlControls;

public partial class ImageContainer : System.Web.UI.UserControl
{
   public String ImagesUrlsString
   {
      get
      {
         String s = (String)ViewState["ImagesUrlsString"];
         return ((s == null) ? String.Empty : s);
      }
      set
      {
         string oldValue = ImagesUrlsString;
         ViewState["ImagesUrlsString"] = value;
         if (!oldValue.Equals(value))
         {
            ViewState["currentImage"] = -1;
            loadUrlsState();
            fillImageContainer();
         }
      }
   }

   private static string imgDir = "Images/";
   private ArrayList urls = new ArrayList();
   //private int currentImage=-1; viewstate

   protected void Page_Load(object sender, EventArgs e)
   {
      loadUrlsState();
      if (!IsPostBack)
      {
         ViewState["currentImage"] = -1;
         fillImageContainer();
      }
   }
```

```
private void fillImageContainer()
{
   if (ImagesUrlsString.Equals(String.Empty))
   {
      cancelImage.ImageUrl = imgDir + "Various/no_image.gif";
      ViewState["currentImage"] = -1;
      return;
   }
   string urlsString = ImagesUrlsString;

   string imgDirFullPath = Server.MapPath(imgDir);
   string[] tokens = urlsString.Split(',');
   bool found = false;

   for (int i = 0; i < tokens.Length - 1; i++)
   {
      string temp = imgDirFullPath + urls[i];
      if (File.Exists(temp))
      {
         cancelImage.ImageUrl = imgDir + urls[i];
         ViewState["currentImage"] = i;
         found = true;
         break;
      }
   }

   if (!found)
   {
      cancelImage.ImageUrl = imgDir + "Various/no_image.gif";
      ViewState["currentImage"] = -1;
   }
}

private void loadUrlsState()
{
   string urlsString = ImagesUrlsString;
   string imgDirFullPath = Server.MapPath(imgDir);
   string[] tokens = urlsString.Split(',');

   urls.Clear();
   for (int i = 0; i < tokens.Length - 1; i++)
   {
      tokens[i] = tokens[i].Replace("\\", "/");
      urls.Add(tokens[i].Substring(5));
   }
}

protected void nextButton_Click(object sender, EventArgs e)
{
   if ( (int)ViewState["currentImage"] == -1 || urls.Count==0)
   {
      return;
   }

   int i = ( (int)ViewState["currentImage"]+1)%urls.Count;
   string imgDirFullPath = Server.MapPath(imgDir);
   for (int j = 0; j < urls.Count; j++)
   {
      string temp = imgDirFullPath + urls[i];
      if (File.Exists(temp))
```

```
            {
                cancelImage.ImageUrl = imgDir + urls[i];
                ViewState["currentImage"] = i;
                return;
            }
            i = (i + 1) % urls.Count;
        }
        cancelImage.ImageUrl = imgDir + "Various/no_image.gif";
        ViewState["currentImage"] = -1;
    }
    protected void prevButton_Click(object sender, EventArgs e)
    {
        if ( (int)ViewState["currentImage"] == -1 || urls.Count==0)
        {
            return;
        }

        int i = (int)ViewState["currentImage"] - 1;
        if (i < 0)
            i = urls.Count - 1;
        string imgDirFullPath = Server.MapPath(imgDir);
        for (int j = 0; j < urls.Count; j++)
        {
            string temp = imgDirFullPath + urls[i];
            if (File.Exists(temp))
            {
                cancelImage.ImageUrl = imgDir + urls[i];
                ViewState["currentImage"] = i;
                return;
            }
            i = i-1;
            if (i < 0)
                i = urls.Count-1;
        }
        cancelImage.ImageUrl = imgDir + "Various/no_image.gif";
        ViewState["currentImage"] = -1;
    }

    public string getDisplayedImageUrl()
    {
        return cancelImage.ImageUrl;
    }
}
```

A13) αρχείο NewData.aspx :

```
<%@ Page Language="C#" Trace="false" AutoEventWireup="true" CodeFile="NewData.aspx.cs"
Inherits="NewData" %>
<%@ Register TagPrefix="uc" TagName="ImageContainer" Src="~/ImageContainer.ascx" %>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml" >
<head runat="server">
    <title>enter new cancellation data</title>
</head>
<body style="background-color:GrayText">
```

```
    <form id="form1" runat="server">
    <div>

    <asp:MultiView ID="MultiView1" ActiveViewIndex="0" runat="server">

    <asp:View ID="View1" runat="server" OnActivate="defaultView_Activate">
      <table width="100%" style="background-color:Gray">
      <tr>
        <td>        </td>
        <td>        </td>
        <td>        </td>
        <td>
          <asp:Button ID="keyboardButton" runat="server" ForeColor="Red" Text="Ottoman keyboard"
OnClick="keyboardButton_Click" />
        </td>
        <td align="right">
          <asp:Label ID="datesLabel" runat="server" Text="Dates"></asp:Label>
        </td>
        <td>
          <asp:TextBox ID="datesTextBox" runat="server"></asp:TextBox>
        </td>
        <td>
          ID<asp:TextBox ID="idTextBox" runat="server" ReadOnly="true"></asp:TextBox>
        </td>
      </tr>
      <tr>
        <td colspan="2">
          <asp:TextBox ID="ottoFieldTextBox" TextMode="MultiLine" Rows="5" runat="server"
Width="100%"></asp:TextBox>
        </td>
        <td align="left">
          <asp:CheckBox ID="ottoCheckBox" runat="server" />
        </td>
        <td>
          Post Office in<br/>
          <asp:DropDownList ID="poDropDownList" runat="server" DataSourceID="poDataSource"
DataTextField="POTurk" DataValueField="POTurk"></asp:DropDownList><br/>
          <asp:AccessDataSource ID="poDataSource" runat="server"
DataFile="~/App_Data/Cancel_Alexandros_Apr09.mdb" SelectCommand="SELECT [POTurk] FROM
[Local Names] ORDER BY [POTurk]"></asp:AccessDataSource>
          <asp:RadioButtonList RepeatDirection="Horizontal" ID="posNegRadioButtonList" runat="server"
AutoPostBack="True" OnSelectedIndexChanged="posNeg_Changed">
            <asp:ListItem>positive</asp:ListItem>
            <asp:ListItem>negative</asp:ListItem>
          </asp:RadioButtonList>
        </td>
        <td rowspan="2" colspan="2" align="center">
          <uc:ImageContainer ID="imgContainer" runat="server" /><br/>
          <asp:FileUpload ID="uploadFile" runat="server" ToolTip="select an image to upload, then press
'add image' button" /><br/>
          <asp:Button ID="addImgButton" runat="server" Text="add image"
OnClick="addImgButton_Click" ToolTip="add image of the above path to this cancellation" />
          <asp:Button ID="remImgButton" runat="server" Text="rem image"
OnClick="remImgButton_Click" ToolTip="remove currently displayed image from this cancellation" /><br/>
          <asp:Button ID="restoreImgButton" runat="server" Text="restore initial images"
OnClick="restoreImgButton_Click" ToolTip="restore initial images of this cancellation(rollback any changes
made)" />
        </td>
        <td rowspan="2">
          <asp:CheckBoxList ID="colorCheckBoxList" runat="server">
```

```
                <asp:ListItem Value="blk">black</asp:ListItem>
                <asp:ListItem Value="blu">blue</asp:ListItem>
                <asp:ListItem Value="brn">brown</asp:ListItem>
                <asp:ListItem Value="crm">carmine</asp:ListItem>
                <asp:ListItem Value="grn">green</asp:ListItem>
                <asp:ListItem>red</asp:ListItem>
                <asp:ListItem Value="vlt">violet</asp:ListItem>
            </asp:CheckBoxList>
        </td>
    </tr>
    <tr>
        <td colspan="2">
            <asp:TextBox ID="frenchFieldTextBox" TextMode="MultiLine" Rows="5" runat="server"
Width="100%"></asp:TextBox>
        </td>
        <td align="left">
            <asp:CheckBox ID="frenchCheckBox" runat="server" />
        </td>
        <td id="shapeExprCell" align="center" style="background-color:Black; color:White; ">
            Enter<br/>
            <asp:Button ID="shapeButton" runat="server" Text="Shape" ForeColor="Red"
OnClick="shapeButton_Click" ToolTip="form for entering shape" />
            <asp:TextBox ID="shapeEnteredTextBox" runat="server" ToolTip="current shape entered, you
can change it manually or by clicking shape button"></asp:TextBox><br />
            <asp:Button ID="exprButton" runat="server" Text="Expression" ForeColor="Red"
OnClick="exprButton_Click" ToolTip="form for entering expression" />
            <asp:TextBox ID="exprEnteredTextBox" runat="server" ToolTip="current expression entered,
you can change it manually or by clicking expression button"></asp:TextBox>
        </td>
    </tr>
    <tr>
        <td align="right">
            CW code
        </td>
        <td>
            <asp:DropDownList ID="cwDropDownList" runat="server" DataSourceID="cwDataSource"
DataTextField="column1" DataValueField="column1"
OnDataBound="cwDDL_DataBound"></asp:DropDownList>
            <asp:AccessDataSource ID="cwDataSource" runat="server"
DataFile="~/App_Data/Cancel_Alexandros_Apr09.mdb"
                SelectCommand="SELECT [C&W Codes] AS column1 FROM [CWCodes] ORDER BY [C&W
Codes]">
            </asp:AccessDataSource>
        </td>
        <td>          </td>
        <td>
            Brandt type<asp:DropDownList ID="brDropDownList" runat="server"
DataSourceID="brDataSource" DataTextField="Bradt_Types" DataValueField="Bradt_Types"
OnDataBound="brDDL_DataBound"></asp:DropDownList>
            <asp:AccessDataSource ID="brDataSource" runat="server"
DataFile="~/App_Data/Cancel_Alexandros_Apr09.mdb"
                SelectCommand="SELECT [Bradt Types] AS Bradt_Types FROM [BrandtTypes] ORDER BY
[Bradt Types]">
            </asp:AccessDataSource>
        </td>
        <td colspan="2">
            Nuhoglou<asp:TextBox ID="nuhoglouTextBox" runat="server"></asp:TextBox>
        </td>
        <td>          </td>
    </tr>
```

```
        <tr>
          <td align="right">
            CW page
          </td>
          <td>
            <asp:TextBox ID="cwPageTextBox" runat="server"></asp:TextBox>
          </td>
          <td>          </td>
          <td>
            Brandt page<asp:TextBox ID="brandtPageTextBox" runat="server"></asp:TextBox>
          </td>
          <td colspan="2">
            NG no and page<asp:TextBox ID="galinosTextBox" runat="server"></asp:TextBox>
          </td>
          <td>          </td>
        </tr>
        <tr>
          <td colspan="4">
            <asp:TextBox ID="commentsTextBox" TextMode="MultiLine" runat="server" Width="100%"
ToolTip="place comments about the cancellation here"></asp:TextBox>
          </td>
          <td>          </td>
          <td>          </td>
          <td>          </td>
        </tr>
        <tr>
          <td colspan="7"><br/></td>
        </tr>
        <tr>
          <td colspan="2" align="right">
            <asp:Button ID="delButton" runat="server" Text="delete this cancellation"
OnClick="delButton_Click" />
          </td>
          <td>          </td>
          <td colspan="2" align="center">
            <asp:Button ID="createButton" runat="server" Text="create new cancellation"
OnClick="createButton_Click" />
          </td>
          <td colspan="2" align="left">
            <asp:Button ID="updateButton" runat="server" Text="update this cancellation"
OnClick="updateButton_Click" />
          </td>
          <td>
            <asp:Button ID="backButton" runat="server" Text="back to search" PostBackUrl="~/Default.aspx"
/>
          </td>
        </tr>
        <tr>
          <td colspan="7">
            <asp:TextBox ID="warningTextBox" runat="server" TextMode="MultiLine" width="100%"
ForeColor="Red" Font-Bold="true" Visible="false"></asp:TextBox>
          </td>
        </tr>
        </table>
        <asp:AccessDataSource ID="cancelDataSource" runat="server"
DataFile="~/App_Data/Cancel_Alexandros_Apr09.mdb"
          DeleteCommand="DELETE FROM Cancellations WHERE (ID = ?)"
          InsertCommand="INSERT INTO Cancellations([French Field], FDef, [Normalised Latin], [Ottoman
Field], ODef, [Normalised Arabic], [Post Office], CW, CWPage, Br, BrandtPage, Nuhoglou, Galinos, Dates,
Colours, [Image], Comments) VALUES
```

```
(@frenchField,@fdef,@normalisedLatin,@ottoField,@odef,@normalisedArabic,@postOffice,@cw,@cwPage
,@br,@brPage,@nuhoglou,@galinos,@dates,@colours,@image,@comments)"
        UpdateCommand="UPDATE Cancellations SET [French Field] = ?, FDef = ?, [Normalised Latin] = ?,
[Ottoman Field] = ?, ODef = ?, [Normalised Arabic] = ?, [Post Office] = ?, CW = ?, CWPage = ?, Br = ?,
BrandtPage = ?, Nuhoglou = ?, Galinos = ?, Dates = ?, Colours = ?, [Image] = ?, Comments = ? WHERE (ID
= ?)"
        SelectCommand="SELECT * FROM [Cancellations]" >
        <DeleteParameters>
          <asp:ControlParameter ControlID="idTextBox" Name="ID" PropertyName="Text" />
        </DeleteParameters>
        <InsertParameters>
          <asp:ControlParameter ControlID="frenchFieldTextBox" Name="frenchField"
PropertyName="Text" />
          <asp:ControlParameter ControlID="frenchCheckBox" Name="fdef" PropertyName="Checked" />
          <asp:Parameter DefaultValue="" Name="normalisedLatin" />
          <asp:ControlParameter ControlID="ottoFieldTextBox" DefaultValue="" Name="ottoField"
            PropertyName="Text" />
          <asp:ControlParameter ControlID="ottoCheckBox" Name="odef" PropertyName="Checked" />
          <asp:Parameter DefaultValue="" Name="normalisedArabic" />
          <asp:ControlParameter ControlID="poDropDownList" DefaultValue="" Name="postOffice"
            PropertyName="SelectedValue" />
          <asp:ControlParameter ControlID="cwDropDownList" Name="cw"
PropertyName="SelectedValue" />
          <asp:ControlParameter ControlID="cwPageTextBox" Name="cwPage" PropertyName="Text" />
          <asp:ControlParameter ControlID="brDropDownList" Name="br" PropertyName="SelectedValue"
/>
          <asp:ControlParameter ControlID="brandtPageTextBox" Name="brPage" PropertyName="Text" />
          <asp:ControlParameter ControlID="nuhoglouTextBox" Name="nuhoglou" PropertyName="Text"/>
          <asp:ControlParameter ControlID="galinosTextBox" Name="galinos" PropertyName="Text" />
          <asp:ControlParameter ControlID="datesTextBox" Name="dates" PropertyName="Text" />
          <asp:Parameter DefaultValue="" Name="colours" />
          <asp:Parameter DefaultValue="" Name="image" />
          <asp:ControlParameter ControlID="commentsTextBox" Name="comments" PropertyName="Text"
/>
        </InsertParameters>
        <UpdateParameters>
          <asp:ControlParameter ControlID="frenchFieldTextBox" Name="frenchField"
PropertyName="Text" />
          <asp:ControlParameter ControlID="frenchCheckBox" Name="fdef" PropertyName="Checked" />
          <asp:Parameter DefaultValue="" Name="normalisedLatin" />
          <asp:ControlParameter ControlID="ottoFieldTextBox" DefaultValue="" Name="ottoField"
            PropertyName="Text" />
          <asp:ControlParameter ControlID="ottoCheckBox" Name="odef" PropertyName="Checked" />
          <asp:Parameter DefaultValue="" Name="normalisedArabic" />
          <asp:ControlParameter ControlID="poDropDownList" DefaultValue="" Name="postOffice"
            PropertyName="SelectedValue" />
          <asp:ControlParameter ControlID="cwDropDownList" Name="cw"
PropertyName="SelectedValue" />
          <asp:ControlParameter ControlID="cwPageTextBox" Name="cwPage" PropertyName="Text" />
          <asp:ControlParameter ControlID="brDropDownList" Name="br" PropertyName="SelectedValue"
/>
          <asp:ControlParameter ControlID="brandtPageTextBox" Name="brPage" PropertyName="Text" />
          <asp:ControlParameter ControlID="nuhoglouTextBox" Name="nuhoglou" PropertyName="Text"/>
          <asp:ControlParameter ControlID="galinosTextBox" Name="galinos" PropertyName="Text" />
          <asp:ControlParameter ControlID="datesTextBox" Name="dates" PropertyName="Text" />
          <asp:Parameter DefaultValue="" Name="colours" />
          <asp:Parameter DefaultValue="" Name="image" />
          <asp:ControlParameter ControlID="commentsTextBox" Name="comments" PropertyName="Text"
/>
          <asp:ControlParameter ControlID="idTextBox" Name="ID" PropertyName="Text" />
```

```
        </UpdateParameters>
      </asp:AccessDataSource>

      <br/>
   <table id="keyboardTable" style="background-color:#FF00FF;" runat="server" visible="false">
   <tr>
      <td colspan="8">
        <asp:Label ID="keyboardTitleLabel" runat="server" Text="Ottoman Keyboard"
ForeColor="Red"></asp:Label>
      </td>
   </tr>
   <tr>
      <td>
        <asp:Button ID="keyButton1" runat="server" OnClick="keyButton_Click" BackColor="DarkGray" />
      </td>
      <td>
        <asp:Button ID="keyButton2" runat="server" OnClick="keyButton_Click" BackColor="DarkGray" />
      </td>
      <td>
        <asp:Button ID="keyButton3" runat="server" OnClick="keyButton_Click" BackColor="DarkGray" />
      </td>
      <td>
        <asp:Button ID="keyButton4" runat="server" OnClick="keyButton_Click" BackColor="DarkGray" />
      </td>
      <td>
        <asp:Button ID="keyButton5" runat="server" OnClick="keyButton_Click" BackColor="DarkGray" />
      </td>
      <td>
        <asp:Button ID="keyButton6" runat="server" OnClick="keyButton_Click" BackColor="DarkGray" />
      </td>
      <td>
        <asp:Button ID="keyButton7" runat="server" OnClick="keyButton_Click" BackColor="DarkGray" />
      </td>
      <td>
        <asp:Button ID="keyButton8" runat="server" OnClick="keyButton_Click" BackColor="DarkGray" />
      </td>
      <td>
        <asp:Button ID="keyButton9" runat="server" OnClick="keyButton_Click" BackColor="DarkGray" />
      </td>
      <td>
       <asp:Button ID="keyButton10" runat="server" OnClick="keyButton_Click" BackColor="DarkGray"/>
      </td>
      <td>
        <asp:Button ID="keyButton11" runat="server" Text="-" OnClick="keyButton_Click"
BackColor="DarkGray" />
      </td>
      <td>
        <asp:Button ID="keyButton12" runat="server" Text="=" OnClick="keyButton_Click"
BackColor="DarkGray" />
      </td>
      <td>      </td>
      <td>
        <asp:Button ID="keyButton13" runat="server" Text="bksp" OnClick="keyButton_Click"
BackColor="DarkGray" />
      </td>
   </tr>
   <tr>
      <td>
       <asp:Button ID="keyButton14" runat="server" OnClick="keyButton_Click" BackColor="DarkGray" />
      </td>
```

```
        <td>
          <asp:Button ID="keyButton15" runat="server" OnClick="keyButton_Click" BackColor="DarkGray" />
        </td>
        <td>
          <asp:Button ID="keyButton16" runat="server" OnClick="keyButton_Click" BackColor="DarkGray" />
        </td>
        <td>
          <asp:Button ID="keyButton17" runat="server" OnClick="keyButton_Click" BackColor="DarkGray" />
        </td>
        <td>
          <asp:Button ID="keyButton18" runat="server" OnClick="keyButton_Click" BackColor="DarkGray" />
        </td>
        <td>
          <asp:Button ID="keyButton19" runat="server" OnClick="keyButton_Click" BackColor="DarkGray" />
        </td>
        <td>
          <asp:Button ID="keyButton20" runat="server" OnClick="keyButton_Click" BackColor="DarkGray" />
        </td>
        <td>
          <asp:Button ID="keyButton21" runat="server" OnClick="keyButton_Click" BackColor="DarkGray" />
        </td>
        <td>
          <asp:Button ID="keyButton22" runat="server" OnClick="keyButton_Click" BackColor="DarkGray" />
        </td>
        <td>
          <asp:Button ID="keyButton23" runat="server" OnClick="keyButton_Click" BackColor="DarkGray" />
        </td>
        <td>
          <asp:Button ID="keyButton24" runat="server" OnClick="keyButton_Click" BackColor="DarkGray" />
        </td>
        <td>
          <asp:Button ID="keyButton25" runat="server" OnClick="keyButton_Click" BackColor="DarkGray" />
        </td>
        <td>
          <asp:Button ID="keyButton26" runat="server" OnClick="keyButton_Click" BackColor="DarkGray" />
        </td>
        <td>
          <asp:Button ID="keyButton55" runat="server" ForeColor="Red" OnClick="keyButton_Click"
BackColor="DarkGray" Text="," />
        </td>
      </tr>
      <tr>
        <td>
          <asp:Button ID="keyButton27" runat="server" OnClick="keyButton_Click" BackColor="DarkGray" />
        </td>
        <td>
          <asp:Button ID="keyButton28" runat="server" OnClick="keyButton_Click" BackColor="DarkGray" />
        </td>
        <td>
          <asp:Button ID="keyButton29" runat="server" OnClick="keyButton_Click" BackColor="DarkGray" />
        </td>
        <td>
          <asp:Button ID="keyButton30" runat="server" OnClick="keyButton_Click" BackColor="DarkGray" />
        </td>
        <td>
          <asp:Button ID="keyButton31" runat="server" OnClick="keyButton_Click" BackColor="DarkGray" />
        </td>
        <td>
          <asp:Button ID="keyButton32" runat="server" OnClick="keyButton_Click" BackColor="DarkGray" />
        </td>
```

```
      <td>
        <asp:Button ID="keyButton33" runat="server" OnClick="keyButton_Click" BackColor="DarkGray" />
      </td>
      <td>
        <asp:Button ID="keyButton34" runat="server" OnClick="keyButton_Click" BackColor="DarkGray" />
      </td>
      <td>
        <asp:Button ID="keyButton35" runat="server" OnClick="keyButton_Click" BackColor="DarkGray" />
      </td>
      <td>
        <asp:Button ID="keyButton36" runat="server" OnClick="keyButton_Click" BackColor="DarkGray" />
      </td>
      <td>
        <asp:Button ID="keyButton37" runat="server" OnClick="keyButton_Click" BackColor="DarkGray" />
      </td>
      <td>
        <asp:Button ID="keyButton38" runat="server" OnClick="keyButton_Click" BackColor="DarkGray" />
      </td>
      <td>
        <asp:Button ID="keyButton39" runat="server" ForeColor="Red" OnClick="keyButton_Click"
BackColor="DarkGray" />
      </td>
      <td>
        <asp:Button ID="keyButton40" runat="server" OnClick="keyButton_Click" Text="NEWLINE"
BackColor="DarkGray" />
      </td>
    </tr>
    <tr>
      <td>
        <asp:Button ID="keyButton41" runat="server" OnClick="keyButton_Click" BackColor="DarkGray" />
      </td>
      <td>
        <asp:Button ID="keyButton42" runat="server" OnClick="keyButton_Click" BackColor="DarkGray" />
      </td>
      <td>
        <asp:Button ID="keyButton43" runat="server" OnClick="keyButton_Click" BackColor="DarkGray" />
      </td>
      <td>
        <asp:Button ID="keyButton44" runat="server" OnClick="keyButton_Click" BackColor="DarkGray" />
      </td>
      <td>
        <asp:Button ID="keyButton45" runat="server" OnClick="keyButton_Click" BackColor="DarkGray" />
      </td>
      <td>
        <asp:Button ID="keyButton46" runat="server" OnClick="keyButton_Click" BackColor="DarkGray" />
      </td>
      <td>
        <asp:Button ID="keyButton47" runat="server" OnClick="keyButton_Click" BackColor="DarkGray" />
      </td>
      <td>
        <asp:Button ID="keyButton48" runat="server" OnClick="keyButton_Click" BackColor="DarkGray" />
      </td>
      <td>
        <asp:Button ID="keyButton49" runat="server" OnClick="keyButton_Click" BackColor="DarkGray" />
      </td>
      <td>
        <asp:Button ID="keyButton50" runat="server" OnClick="keyButton_Click" BackColor="DarkGray" />
      </td>
      <td>
        <asp:Button ID="keyButton51" runat="server" OnClick="keyButton_Click" BackColor="DarkGray" />
```

```
      </td>
      <td>
        <asp:Button ID="keyButton52" runat="server" OnClick="keyButton_Click" BackColor="DarkGray" />
      </td>
      <td>
        <asp:Button ID="keyButton53" runat="server" Text="(" OnClick="keyButton_Click"
BackColor="DarkGray" />
      </td>
      <td>
        <asp:Button ID="keyButton54" runat="server" Text=")" OnClick="keyButton_Click"
BackColor="DarkGray" />
      </td>
    </tr>
    <tr>
      <td colspan="5">
      </td>
      <td colspan="7">
        <asp:Button ID="keySpaceButton" runat="server" Text="Space" BackColor="DarkGray"
OnClick="keyButton_Click" Width="100%" />
      </td>
      <td>
      </td>
      <td>
        <asp:Button ID="keyCloseButton" runat="server" Text="Close" OnClick="keyButton_Click"
BackColor="DarkGray" />
      </td>
    </tr>
    <tr>
      <td colspan="3">common terms in:</td>
      <td colspan="3">
        <asp:DropDownList ID="keyboardOttoDDL" runat="server" AutoPostBack="True"
DataSourceID="keyboardDataSource" DataTextField="Term_Otto" DataValueField="Term_Otto"
OnDataBound="keyboardDDL_DataBound"
OnSelectedIndexChanged="commonTermChosen"></asp:DropDownList>
        <asp:AccessDataSource ID="keyboardDataSource" runat="server"
DataFile="~/App_Data/Cancel_Alexandros_Apr09.mdb"
          SelectCommand="SELECT [Term_Otto], [Term_Turk], [Term_Meaning] FROM [Terms] ORDER
BY [Term_Turk]">
        </asp:AccessDataSource>
      </td>
      <td colspan="4">
        <asp:DropDownList ID="keyboardTurkDDL" runat="server" AutoPostBack="True"
DataSourceID="keyboardDataSource" DataTextField="Term_Turk" DataValueField="Term_Turk"
OnDataBound="keyboardDDL_DataBound" OnSelectedIndexChanged="commonTermChosen">
        </asp:DropDownList>
      </td>
      <td colspan="4">
        <asp:DropDownList ID="keyboardEnglDDL" runat="server" AutoPostBack="True"
DataSourceID="keyboardDataSource" DataTextField="Term_Meaning" DataValueField="Term_Meaning"
OnDataBound="keyboardDDL_DataBound" OnSelectedIndexChanged="commonTermChosen">
        </asp:DropDownList>
      </td>
    </tr>
  </table>
  </asp:View>

  <asp:View ID="shapeView" runat="server" OnActivate="shapeView_Activate">
    <table width="100%">
    <tr>
      <td colspan="4" align="center">
```

```
            <asp:Label ID="titleLabel" Font-Bold="true" runat="server" Text="Entering the shape of the
cancellation"></asp:Label>
          </td>
          <td>
            <asp:TextBox ID="shapeTextBox" runat="server"></asp:TextBox>
          </td>
        </tr>
        <tr>
          <td style="width:20%">          </td>
          <td style="width:20%">          </td>
          <td style="width:20%">          </td>
          <td style="width:20%">          </td>
          <td style="width:20%">          </td>
        </tr>
        <tr>
          <td colspan="3" align="center">
          <asp:RadioButtonList ID="shapeRBL" RepeatDirection="Horizontal" runat="server"
AutoPostBack="True" OnSelectedIndexChanged="shapeRBLSelected">
            <asp:ListItem>Circular</asp:ListItem>
            <asp:ListItem>Oval</asp:ListItem>
            <asp:ListItem>Rectangular</asp:ListItem>
          </asp:RadioButtonList><br />
          <asp:Panel ID="greenPanel" BorderColor="green" BorderStyle="Solid" BorderWidth="5px"
runat="server">
            <table>
            <tr>
              <td style="width:33%" align="right">
                <asp:CheckBox ID="circOvalCB" TextAlign="Left" Text="circular/oval unclear"
AutoPostBack="true" runat="server" OnCheckedChanged="circOvalChecked" />
              </td>
              <td style="width:33%"></td>
              <td style="width:34%" align="left">
                <asp:CheckBox ID="ovalRectCB" TextAlign="Right" Text="oval/rectangular unclear"
AutoPostBack="true" runat="server" OnCheckedChanged="ovalRectChecked" />
              </td>
            </tr>
            <tr>
              <td>
                <asp:Image ID="noBorderImage" ImageUrl="Images/Shapes/n1_h.bmp" runat="server" />
              </td>
              <td>
                <asp:Image ID="crescentImage" ImageUrl="Images/Shapes/c1_h.bmp" runat="server" />
              </td>
              <td>
                <asp:Image ID="ringImage" ImageUrl="Images/Shapes/r1_h.bmp" runat="server" />
              </td>
            </tr>
            <tr>
              <td>
                <asp:RadioButton ID="noBorderRB" Text="no border" runat="server" AutoPostBack="True"
OnCheckedChanged="shapeBorderChanged" />
              </td>
              <td>
                <asp:RadioButton ID="crescentRB" Text="crescent" runat="server" AutoPostBack="True"
OnCheckedChanged="shapeBorderChanged" />
              </td>
              <td>
                <asp:RadioButton ID="ringRB" Text="ring around" runat="server" AutoPostBack="True"
OnCheckedChanged="shapeBorderChanged" />
                <br />
```

```
                <asp:CheckBox ID="ringCheckBox" Text="ring not clear" runat="server"
AutoPostBack="true" OnCheckedChanged="ringUnclearChecked" />
              </td>
            </tr>
            <tr>
              <td>
                <asp:Image ID="laurelImage" ImageUrl="Images/Shapes/l1_h.bmp" runat="server" />
              </td>
              <td>
                <asp:Image ID="rosetteImage" ImageUrl="Images/Shapes/t1_h.bmp" runat="server" />
              </td>
              <td>
                star on top?<br />
                <asp:RadioButtonList ID="starRBL" runat="server" AutoPostBack="True"
OnSelectedIndexChanged="starChanged">
                  <asp:ListItem>YES</asp:ListItem>
                  <asp:ListItem>NO</asp:ListItem>
                  <asp:ListItem>unclear</asp:ListItem>
                </asp:RadioButtonList>
              </td>
            </tr>
            <tr>
              <td>
                <asp:RadioButton ID="laurelRB" Text="laurel" runat="server" AutoPostBack="True"
OnCheckedChanged="shapeBorderChanged" />
              </td>
              <td>
                <asp:RadioButton ID="rosetteRB" Text="rosette edge" runat="server" AutoPostBack="True"
OnCheckedChanged="shapeBorderChanged" />
              </td>
              <td></td>
            </tr>
            <tr>
              <td colspan="2" align="center">
                <asp:RadioButton ID="dontKnowRB" Text="don't know" runat="server"
AutoPostBack="True" OnCheckedChanged="shapeBorderChanged" />
              </td>
              <td>
                <asp:Button ID="shapeEnterButton" runat="server" Text="enter shape"
OnClick="shapeEnter_Click" />
              </td>
            </tr>
            </table>
          </asp:Panel>
          </td>

          <td colspan="2" align="center">
            <asp:Button ID="shapeBackButton" runat="server" Text="Back"
OnClick="shapeBackButton_Click" />
            <br /><br /><br /><br />
            <asp:RadioButton ID="specialShapeRB" Text="Special Shape" runat="server"
AutoPostBack="True" OnCheckedChanged="speciaShapeChecked" /><br />
          <asp:Panel ID="bluePanel" BorderColor="blue" BorderStyle="Solid" BorderWidth="5px"
runat="server">
            <table>
            <tr>
              <td style="width:50%"></td>
              <td style="width:50%"></td>
            </tr>
            <tr>
```

```
            <td>
              <asp:RadioButtonList ID="specialShapeRBL" runat="server" AutoPostBack="True"
OnSelectedIndexChanged="specialShapeRBLchanged">
                <asp:ListItem>circular latin</asp:ListItem>
                <asp:ListItem>circular ottoman</asp:ListItem>
                <asp:ListItem>circular mixed</asp:ListItem>
                <asp:ListItem>other shapes</asp:ListItem>
              </asp:RadioButtonList>
            </td>
            <td>
              <asp:Image ID="specialImage" runat="server" />
              <br/>example pic
            </td>
          </tr>
          <tr>
            <td>                </td>
            <td>
              <asp:Button ID="specialEnterButton" runat="server" Text="enter shape"
OnClick="shapeEnter_Click" />
            </td>
          </tr>
          </table>
        </asp:Panel>
        </td>
      </tr>
      </table>
    </asp:View>

    <asp:View ID="exprView" runat="server" OnActivate="exprView_Activate">
      <table width="100%">
      <tr>
        <td colspan="10">
          <asp:Label ID="title" Font-Size="Large" Font-Bold="true" runat="server" Text="Entering the
placement of Ottoman Post expressions on the cancellation"></asp:Label>
        </td>
        <td></td>
        <td colspan="2">
          <asp:Button ID="exprBackButton" runat="server" Text="Back" OnClick="exprBackButton_Click"/>
        </td>
        <td></td>
      </tr>
      <tr>
        <td colspan="14"><br /></td>
      </tr>
      <tr align="center">
        <td></td>
        <td></td>
        <td colspan="2">
          <asp:Label ID="anCanibLabel" BackColor="black" ForeColor="white" Font-Bold="true"
Width="100%" runat="server" Text="an canib"></asp:Label>
        </td>
        <td></td>
        <td></td>
        <td colspan="2">
          <asp:Label ID="telegrafLabel" BackColor="black" ForeColor="white" Font-Bold="true"
Width="100%" runat="server" Text="telegraf"></asp:Label>
        </td>
        <td></td>
        <td></td>
        <td colspan="2">
```

```
            <asp:Label ID="postaLabel" BackColor="black" ForeColor="white" Font-Bold="true"
Width="100%" runat="server" Text="posta"></asp:Label>
          </td>
          <td></td>
          <td></td>
        </tr>
        <tr>
          <td align="right">
            top<br />
            middle<br />
            bottom<br />
            unclear
          </td>
          <td align="right">
            <asp:RadioButtonList ID="anCanibRBL1" runat="server">
              <asp:ListItem Value="top" Text=""></asp:ListItem>
              <asp:ListItem Value="middle" Text=""></asp:ListItem>
              <asp:ListItem Value="bottom" Text=""></asp:ListItem>
              <asp:ListItem Value="unclear" Text="" Selected="True"></asp:ListItem>
            </asp:RadioButtonList>
          </td>
          <td>
            <asp:ImageButton ID="anCanibImage1" runat="server" ToolTip="click for other variants of this
expression" OnClick="image_Click" />
          </td>
          <td align="right">
            <asp:ImageButton ID="anCanibImage2" runat="server" ToolTip="click for other variants of this
expression" OnClick="image_Click" />
          </td>
          <td align="left">
            <asp:RadioButtonList ID="anCanibRBL2" runat="server">
              <asp:ListItem Value="top" Text=""></asp:ListItem>
              <asp:ListItem Value="middle" Text=""></asp:ListItem>
              <asp:ListItem Value="bottom" Text=""></asp:ListItem>
              <asp:ListItem Value="unclear" Text="" Selected="True"></asp:ListItem>
            </asp:RadioButtonList>
          </td>
          <td align="right">
            <asp:RadioButtonList ID="telegrafRBL1" runat="server">
              <asp:ListItem Value="top" Text=""></asp:ListItem>
              <asp:ListItem Value="middle" Text=""></asp:ListItem>
              <asp:ListItem Value="bottom" Text=""></asp:ListItem>
              <asp:ListItem Value="unclear" Text="" Selected="True"></asp:ListItem>
            </asp:RadioButtonList>
          </td>
          <td>
            <asp:ImageButton ID="telegrafImage1" runat="server" ToolTip="click for other variants of this
expression" OnClick="image_Click" />
          </td>
          <td align="right">
            <asp:ImageButton ID="telegrafImage2" runat="server" ToolTip="click for other variants of this
expression" OnClick="image_Click" />
          </td>
          <td align="left">
            <asp:RadioButtonList ID="telegrafRBL2" runat="server">
              <asp:ListItem Value="top" Text=""></asp:ListItem>
              <asp:ListItem Value="middle" Text=""></asp:ListItem>
              <asp:ListItem Value="bottom" Text=""></asp:ListItem>
              <asp:ListItem Value="unclear" Text="" Selected="True"></asp:ListItem>
            </asp:RadioButtonList>
```

```
        </td>
        <td align="right">
          <asp:RadioButtonList ID="postaRBL1" runat="server">
            <asp:ListItem Value="top" Text=""></asp:ListItem>
            <asp:ListItem Value="middle" Text=""></asp:ListItem>
            <asp:ListItem Value="bottom" Text=""></asp:ListItem>
            <asp:ListItem Value="unclear" Text="" Selected="True"></asp:ListItem>
          </asp:RadioButtonList>
        </td>
        <td>
          <asp:ImageButton ID="postaImage1" runat="server" ToolTip="click for other variants of this
expression" OnClick="image_Click" />
        </td>
        <td align="right">
          <asp:ImageButton ID="postaImage2" runat="server" ToolTip="click for other variants of this
expression" OnClick="image_Click" />
        </td>
        <td  align="left">
          <asp:RadioButtonList ID="postaRBL2" runat="server">
            <asp:ListItem Value="top" Text=""></asp:ListItem>
            <asp:ListItem Value="middle" Text=""></asp:ListItem>
            <asp:ListItem Value="bottom" Text=""></asp:ListItem>
            <asp:ListItem Value="unclear" Text="" Selected="True"></asp:ListItem>
          </asp:RadioButtonList>
        </td>
        <td>
          top<br />
          middle<br />
          bottom<br />
          unclear
        </td>
      </tr>
      <tr align="center">
        <td>          </td>
        <td>          </td>
        <td colspan="2">
          <asp:RadioButtonList ID="anCanibExistRBL" AutoPostBack="true"
OnSelectedIndexChanged="anCanibExist_Changed" runat="server">
            <asp:ListItem Selected="True">existent</asp:ListItem>
            <asp:ListItem>nonexistent</asp:ListItem>
          </asp:RadioButtonList>
        </td>
        <td>          </td>
        <td>          </td>
        <td colspan="2">
          <asp:RadioButtonList ID="telegrafExistRBL" runat="server" AutoPostBack="True"
OnSelectedIndexChanged="telegrafExist_Changed">
            <asp:ListItem Selected="True">existent</asp:ListItem>
            <asp:ListItem>nonexistent</asp:ListItem>
          </asp:RadioButtonList>
        </td>
        <td>          </td>
        <td>          </td>
        <td colspan="2">
          <asp:RadioButtonList ID="postaExistRBL" runat="server" AutoPostBack="True"
OnSelectedIndexChanged="postaExist_Changed">
            <asp:ListItem Selected="True">existent</asp:ListItem>
            <asp:ListItem>nonexistent</asp:ListItem>
          </asp:RadioButtonList>
        </td>
```

```
        <td>          </td>
        <td>          </td>
      </tr>
      <tr>
        <td colspan="14"><br /></td>
      </tr>
      <tr align="center">
        <td></td>
        <td></td>
        <td colspan="2">
          <asp:Label ID="subesiLabel" BackColor="black" ForeColor="white" Font-Bold="true"
Width="100%" runat="server" Text="subesi"></asp:Label>
        </td>
        <td></td>
        <td></td>
        <td colspan="2">
          <asp:Label ID="hanesiLabel" BackColor="black" ForeColor="white" Font-Bold="true"
Width="100%" runat="server" Text="hane(si)"></asp:Label>
        </td>
        <td></td>
        <td></td>
        <td colspan="2">
          <asp:Label ID="numeralsLabel" BackColor="black" ForeColor="white" Font-Bold="true"
Width="100%" runat="server" Text="Numerals"
            ToolTip="search taking also into account the hindi numeral on a negative cancellation. Use hindi
or arabic textbox below">
          </asp:Label>
        </td>
        <td></td>
        <td></td>
      </tr>
      <tr>
        <td align="right">
          top<br />
          middle<br />
          bottom<br />
          unclear
        </td>
        <td align="right">
          <asp:RadioButtonList ID="subesiRBL1" runat="server">
            <asp:ListItem Value="top" Text=""></asp:ListItem>
            <asp:ListItem Value="middle" Text=""></asp:ListItem>
            <asp:ListItem Value="bottom" Text=""></asp:ListItem>
            <asp:ListItem Value="unclear" Text="" Selected="True"></asp:ListItem>
          </asp:RadioButtonList>
        </td>
        <td>
          <asp:ImageButton ID="subesiImage1" runat="server" ToolTip="click for other variants of this
expression" OnClick="image_Click" />
        </td>
        <td align="right">
          <asp:ImageButton ID="subesiImage2" runat="server" ToolTip="click for other variants of this
expression" OnClick="image_Click" />
        </td>
        <td align="left">
          <asp:RadioButtonList ID="subesiRBL2" runat="server">
            <asp:ListItem Value="top" Text=""></asp:ListItem>
            <asp:ListItem Value="middle" Text=""></asp:ListItem>
            <asp:ListItem Value="bottom" Text=""></asp:ListItem>
            <asp:ListItem Value="unclear" Text="" Selected="True"></asp:ListItem>
```

```
            </asp:RadioButtonList>
          </td>
          <td align="right">
            <asp:RadioButtonList ID="hanesiRBL1" runat="server">
              <asp:ListItem Value="top" Text=""></asp:ListItem>
              <asp:ListItem Value="middle" Text=""></asp:ListItem>
              <asp:ListItem Value="bottom" Text=""></asp:ListItem>
              <asp:ListItem Value="unclear" Text="" Selected="True"></asp:ListItem>
            </asp:RadioButtonList>
          </td>
          <td>
            <asp:ImageButton ID="hanesiImage1" runat="server" ToolTip="click for other variants of this
expression" OnClick="image_Click" />
          </td>
          <td align="right">
            <asp:ImageButton ID="hanesiImage2" runat="server" ToolTip="click for other variants of this
expression" OnClick="image_Click" />
          </td>
          <td align="left">
            <asp:RadioButtonList ID="hanesiRBL2" runat="server">
              <asp:ListItem Value="top" Text=""></asp:ListItem>
              <asp:ListItem Value="middle" Text=""></asp:ListItem>
              <asp:ListItem Value="bottom" Text=""></asp:ListItem>
              <asp:ListItem Value="unclear" Text="" Selected="True"></asp:ListItem>
            </asp:RadioButtonList>
          </td>
          <td></td>
          <td></td>
          <td></td>
          <td></td>
          <td></td>
        </tr>
        <tr align="center">
          <td></td>
          <td></td>
          <td colspan="2">
            <asp:RadioButtonList ID="subesiExistRBL" AutoPostBack="true"
OnSelectedIndexChanged="subesiExist_Changed" runat="server">
              <asp:ListItem Selected="True">existent</asp:ListItem>
              <asp:ListItem>nonexistent</asp:ListItem>
            </asp:RadioButtonList>
          </td>
          <td></td>
          <td></td>
          <td colspan="2">
            <asp:RadioButtonList ID="hanesiExistRBL" runat="server" AutoPostBack="True"
OnSelectedIndexChanged="hanesiExist_Changed">
              <asp:ListItem Selected="True">existent</asp:ListItem>
              <asp:ListItem>nonexistent</asp:ListItem>
            </asp:RadioButtonList>
          </td>
          <td></td>
          <td></td>
          <td colspan="2">
            <asp:Button ID="exprEnterButton" runat="server" Text="enter expression"
OnClick="exprEnter_Click" />
          </td>
          <td></td>
          <td></td>
        </tr>
```

```
        </table>

    </asp:View>

    </asp:MultiView>

    </div>
    </form>
</body>
</html>
```

A14) αρχείο NewData.aspx.cs :

```
using System;
using System.Data;
using System.Configuration;
using System.Collections;
using System.IO;
using System.Web;
using System.Web.Security;
using System.Web.UI;
using System.Web.UI.WebControls;
using System.Web.UI.WebControls.WebParts;
using System.Web.UI.HtmlControls;

public partial class NewData : System.Web.UI.Page
{

    protected void Page_Load(object sender, EventArgs e)
    {
        if (!IsPostBack)
        {
            initKeyboard();
            string temp;

            datesTextBox.Text = (string)Session["selectedCancelDates"];
            idTextBox.Text = "" + (int)Session["selectedCancelID"];
            frenchFieldTextBox.Text = (string)Session["selectedCancelFrenchField"];
            ottoFieldTextBox.Text = (string)Session["selectedCancelOttoField"];
            frenchCheckBox.Checked = (bool)Session["selectedCancelFDef"];
            ottoCheckBox.Checked = (bool)Session["selectedCancelODef"];

            temp = (string)Session["selectedCancelCW"];
            cwDropDownList.SelectedValue = temp;
            if (temp.StartsWith("S"))
            {
                posNegRadioButtonList.SelectedValue = "negative";
                frenchFieldTextBox.BackColor = System.Drawing.Color.Black;
                frenchFieldTextBox.ForeColor = System.Drawing.Color.White;
                ottoFieldTextBox.BackColor = System.Drawing.Color.Black;
                ottoFieldTextBox.ForeColor = System.Drawing.Color.White;
            }
            else
            {
                posNegRadioButtonList.SelectedValue = "positive";
                frenchFieldTextBox.BackColor = System.Drawing.Color.White;
                frenchFieldTextBox.ForeColor = System.Drawing.Color.Black;
                ottoFieldTextBox.BackColor = System.Drawing.Color.White;
```

```
    ottoFieldTextBox.ForeColor = System.Drawing.Color.Black;
    shapeButton.Enabled = false;
    shapeEnteredTextBox.Enabled = false;
    exprButton.Enabled = false;
    exprEnteredTextBox.Enabled = false;

}
cwPageTextBox.Text = (string)Session["selectedCancelCWPage"];

temp = (string)Session["selectedCancelBr"];
if (!temp.Equals(String.Empty))
    brDropDownList.SelectedValue = temp;

brandtPageTextBox.Text = (string)Session["selectedCancelBrPage"];
nuhoglouTextBox.Text = (string)Session["selectedCancelNuhoglou"];
galinosTextBox.Text = (string)Session["selectedCancelGalinos"];
commentsTextBox.Text = (string)Session["selectedCancelComments"];
poDropDownList.SelectedValue = (string)Session["selectedCancelPostOffice"];

string[] tokens = ((string)Session["selectedCancelColours"]).Split(',');
for ( int i=0; i<tokens.Length; i++ )
    switch (tokens[i])
    {
        case "blk":
            colorCheckBoxList.Items[0].Selected = true;
            break;
        case "blu":
            colorCheckBoxList.Items[1].Selected = true;
            break;
        case "brn":
            colorCheckBoxList.Items[2].Selected = true;
            break;
        case "crm":
            colorCheckBoxList.Items[3].Selected = true;
            break;
        case "grn":
            colorCheckBoxList.Items[4].Selected = true;
            break;
        case "red":
            colorCheckBoxList.Items[5].Selected = true;
            break;
        case "vlt":
            colorCheckBoxList.Items[6].Selected = true;
            break;
        default:
            if(tokens[i].StartsWith("1"))
                shapeEnteredTextBox.Text = tokens[i];
            else if (tokens[i].StartsWith("2"))
                exprEnteredTextBox.Text = tokens[i];
            break;
    }

imgContainer.ImagesUrlsString = (string)Session["selectedCancelImage"];

//
//expression view
anCanibImage1.ImageUrl = "Images/NegSamples/an/an1.bmp";
anCanibImage2.ImageUrl = "Images/NegSamples/janib/janib1.bmp";
telegrafImage1.ImageUrl = "Images/NegSamples/telegr/telegr1.bmp";
telegrafImage2.ImageUrl = "Images/NegSamples/af/af1.bmp";
```

```
        postaImage1.ImageUrl = "Images/NegSamples/po/po1.bmp";
        postaImage2.ImageUrl = "Images/NegSamples/sta/sta1.bmp";
        subesiImage1.ImageUrl = "Images/NegSamples/sube/sube1.bmp";
        subesiImage2.ImageUrl = "Images/NegSamples/si/si1.bmp";
        hanesiImage1.ImageUrl = "Images/NegSamples/ha/ha1.bmp";
        hanesiImage2.ImageUrl = "Images/NegSamples/neyi/neyi1.bmp";
        ViewState["an1"] = 1;
        ViewState["an2"] = 1;
        ViewState["tel1"] = 1;
        ViewState["tel2"] = 1;
        ViewState["post1"] = 1;
        ViewState["post2"] = 1;
        ViewState["sub1"] = 1;
        ViewState["sub2"] = 1;
        ViewState["han1"] = 1;
        ViewState["han2"] = 1;
        ViewState["prefix"] = Server.MapPath("Images/NegSamples/");
    }

    warningTextBox.Visible = false;
}

protected void cwDDL_DataBound(object sender, EventArgs e)
{
    cwDropDownList.Items.Remove("*");
    cwDropDownList.Items.Remove("?");
    cwDropDownList.Items.Remove("C?");
    cwDropDownList.Items.Remove("none");
}
protected void brDDL_DataBound(object sender, EventArgs e)
{
    brDropDownList.Items.Remove("-");
    brDropDownList.Items.Remove("*");
    brDropDownList.Items.Remove("?");
    brDropDownList.Items.Remove("none");
    brDropDownList.Items.Insert(0, "");
}

protected void delButton_Click(object sender, EventArgs e)
{
    cancelDataSource.Delete();
}
protected void updateButton_Click(object sender, EventArgs e)
{
    //elegxos an yparxei cancellation me to ID oste
    //na epistrepsei katallilo minima sto xristi
    //an den yparxei

    if (!checkValuesAccepted())
        return;

    if (frenchFieldTextBox.Text.Trim().Equals(""))
        frenchFieldTextBox.Text = "=";
    else
    {
        if (frenchFieldTextBox.Text.Contains(";"))
            frenchCheckBox.Checked = false;
    }
    if (ottoFieldTextBox.Text.Trim().Equals(""))
        ottoFieldTextBox.Text = "=";
```

```
    else
    {
        ottoFieldTextBox.Text = ottoFieldTextBox.Text.Replace('?', '\u061b');
        ottoFieldTextBox.Text = ConstantClass.strAllArabised(ottoFieldTextBox.Text);
        if (ottoFieldTextBox.Text.Contains("\u061b"))
            ottoCheckBox.Checked = false;
    }

    cancelDataSource.UpdateParameters["normalisedLatin"].DefaultValue =
        ConstantClass.strOneLine(frenchFieldTextBox.Text);
    cancelDataSource.UpdateParameters["normalisedArabic"].DefaultValue =
        ConstantClass.strOneLine(ottoFieldTextBox.Text);
    cancelDataSource.UpdateParameters["colours"].DefaultValue = getColoursParameter();
    cancelDataSource.UpdateParameters["image"].DefaultValue = imgContainer.ImagesUrlsString;

    cancelDataSource.Update();
}
protected void createButton_Click(object sender, EventArgs e)
{
    if (!checkValuesAccepted())
        return;

    if (frenchFieldTextBox.Text.Trim().Equals(""))
        frenchFieldTextBox.Text = "=";
    else
    {
        if (frenchFieldTextBox.Text.Contains(";"))
            frenchCheckBox.Checked = false;
    }
    if (ottoFieldTextBox.Text.Trim().Equals(""))
        ottoFieldTextBox.Text = "=";
    else
    {
        ottoFieldTextBox.Text = ottoFieldTextBox.Text.Replace('?','\u061b');
        ottoFieldTextBox.Text = ConstantClass.strAllArabised(ottoFieldTextBox.Text);
        if (ottoFieldTextBox.Text.Contains("\u061b"))
            ottoCheckBox.Checked = false;
    }

    cancelDataSource.InsertParameters["normalisedLatin"].DefaultValue =
        ConstantClass.strOneLine(frenchFieldTextBox.Text);
    cancelDataSource.InsertParameters["normalisedArabic"].DefaultValue =
        ConstantClass.strOneLine(ottoFieldTextBox.Text);
    cancelDataSource.InsertParameters["colours"].DefaultValue = getColoursParameter();
    cancelDataSource.InsertParameters["image"].DefaultValue = imgContainer.ImagesUrlsString;

    cancelDataSource.Insert();
}
private bool checkValuesAccepted()
{
    if (cwDropDownList.SelectedValue.StartsWith("S"))
    {
        if (posNegRadioButtonList.SelectedValue.Equals("positive"))
        {
            warningTextBox.Text = "cw codes that start with S belong to negative cancellations\n" +
                "look at cw code DropDownList and positive/negative RadioButtonList above";
            warningTextBox.Visible = true;
            return false;
        }
    }
```

```
    else
    {
        if (posNegRadioButtonList.SelectedValue.Equals("negative"))
        {
            warningTextBox.Text = "cw codes for negative cancellations must start with S\n" +
                "look at cw code DropDownList and positive/negative RadioButtonList above";
            warningTextBox.Visible = true;
            return false;
        }
        if (!shapeEnteredTextBox.Text.Trim().Equals(string.Empty))
        {
            warningTextBox.Text = "do not enter shape for positive " +
                        "cancellations.\nShape TextBox emptied.";
            warningTextBox.Visible = true;
            shapeEnteredTextBox.Text = "";
            return false;
        }
        if (!exprEnteredTextBox.Text.Trim().Equals(string.Empty))
        {
            warningTextBox.Text = "do not enter expression for positive " +
                        "cancellations.\nExpression TextBox emptied.";
            warningTextBox.Visible = true;
            exprEnteredTextBox.Text = "";
            return false;
        }
    }

    if (brDropDownList.SelectedValue.Equals(""))
        if (!brandtPageTextBox.Text.Equals(""))
        {
            warningTextBox.Text = "Brandt type missing, \n" +
                "so brandt page will be erased";
            warningTextBox.Visible = true;
            brandtPageTextBox.Text = "";
            return false;
        }

    if (ConstantClass.containsOttoman(frenchFieldTextBox.Text))
    {
        warningTextBox.Text = "No ottoman characters allowed in french field\n" +
            "Please remove ottoman characters.";
        warningTextBox.Visible = true;
        return false;
    }
    if (ConstantClass.containsFrench(ottoFieldTextBox.Text))
    {
        warningTextBox.Text = "No latin characters allowed in ottoman field\n" +
            "Please remove latin characters.";
        warningTextBox.Visible = true;
        return false;
    }

    return true;
}

protected void keyButton_Click(object sender, EventArgs e)
{
    Button but = (Button)sender;
    string temp = ottoFieldTextBox.Text;
```

```
    if (but.ID.Equals("keyCloseButton"))
        keyboardTable.Visible = false;
    else if (but.ID.Equals("keySpaceButton"))
        ottoFieldTextBox.Text = temp + " ";
    else if (but.ID.Equals("keyButton13"))          //backspace
    {
        if (temp.Length > 1)
            ottoFieldTextBox.Text = temp.Substring(0, temp.Length - 1);
        else
            ottoFieldTextBox.Text = "";
    }
    else if (but.ID.Equals("keyButton40"))          //newline
    {
        ottoFieldTextBox.Text = temp + "\n";
    }
    else
        ottoFieldTextBox.Text = temp + but.Text;
}
protected void keyboardDDL_DataBound(object sender, EventArgs e)
{
    DropDownList ddl = (DropDownList)sender;
    string text = "";

    if (ddl.ID.Equals("keyboardOttoDDL"))
        text = "otto";
    else if (ddl.ID.Equals("keyboardTurkDDL"))
        text = "turk";
    else if (ddl.ID.Equals("keyboardEnglDDL"))
        text = "engl";

    ddl.Items.Insert(0, text);
}
protected void commonTermChosen(object sender, EventArgs e)
{
    int i = ((DropDownList)sender).SelectedIndex;
    keyboardOttoDDL.SelectedIndex = i;
    keyboardTurkDDL.SelectedIndex = i;
    keyboardEnglDDL.SelectedIndex = i;
    if (!keyboardOttoDDL.SelectedValue.Equals("otto"))
        ottoFieldTextBox.Text += keyboardOttoDDL.SelectedValue;
}
protected void keyboardButton_Click(object sender, EventArgs e)
{
    keyboardTable.Visible = true;
}
private void initKeyboard()
{
    keyButton1.Text = "\u0661";
    keyButton2.Text = "\u0662";
    keyButton3.Text = "\u0663";
    keyButton4.Text = "\u0664";
    keyButton5.Text = "\u0665";
    keyButton6.Text = "\u0666";
    keyButton7.Text = "\u0667";
    keyButton8.Text = "\u0668";
    keyButton9.Text = "\u0669";
    keyButton10.Text = "\u0660";
    keyButton11.Text = "\u002d";
    keyButton12.Text = "\u003d";
    keyButton13.Text = "bksp";
```

```
    keyButton14.Text = "\u0636";
    keyButton15.Text = "\u0635";
    keyButton16.Text = "\u062b";
    keyButton17.Text = "\u0642";
    keyButton18.Text = "\u0641";
    keyButton19.Text = "\u063a";
    keyButton20.Text = "\u0639";
    keyButton21.Text = "\u0647";
    keyButton22.Text = "\u062e";
    keyButton23.Text = "\u062d";
    keyButton24.Text = "\u062c";
    keyButton25.Text = "\u0686";
    keyButton26.Text = "\u067e";

    keyButton27.Text = "\u0634";
    keyButton28.Text = "\u0633";
    keyButton29.Text = "\u06cc";
    keyButton30.Text = "\u0628";
    keyButton31.Text = "\u0644";
    keyButton32.Text = "\u0627";
    keyButton33.Text = "\u062a";
    keyButton34.Text = "\u0646";
    keyButton35.Text = "\u0645";
    keyButton36.Text = "\u06a9";
    keyButton37.Text = "\u06af";
    keyButton38.Text = "\u066d";
    keyButton39.Text = "\u061b";
    keyButton40.Text = "NewLine";

    keyButton41.Text = "\u0629";
    keyButton42.Text = "\u0638";
    keyButton43.Text = "\u0637";
    keyButton44.Text = "\u0698";
    keyButton45.Text = "\u0632";
    keyButton46.Text = "\u0631";
    keyButton47.Text = "\u0630";
    keyButton48.Text = "\u062f";
    keyButton49.Text = "\u0626";
    keyButton50.Text = "\u0648";
    keyButton51.Text = "\u064a";
    keyButton52.Text = "\u0622";
    keyButton53.Text = "\u0028";
    keyButton54.Text = "\u0029";
}

private string getColoursParameter()
{
    string col = "";

    if (posNegRadioButtonList.SelectedValue.Equals("negative"))
        col = "neg,";

    for (int i = 0; i < colorCheckBoxList.Items.Count; i++)
    {
        if (colorCheckBoxList.Items[i].Selected)
            col += colorCheckBoxList.Items[i].Value + ",";
    }

    if (posNegRadioButtonList.SelectedValue.Equals("negative"))
```

```
    {
        if (!shapeEnteredTextBox.Text.Trim().Equals(string.Empty))
            col += shapeEnteredTextBox.Text + ",";
        if (!exprEnteredTextBox.Text.Trim().Equals(string.Empty))
            col += exprEnteredTextBox.Text + ",";
    }

    return col;
}

protected void addImgButton_Click(object sender, EventArgs e)
{
    if (uploadFile.FileName.Equals(""))
    {
        warningTextBox.Text = "you have not given a filename for the image to add\n" +
            "use the browse button above";
        warningTextBox.Visible = true;
        return;
    }

    string path = Server.MapPath("Images/Uploaded/") + uploadFile.FileName;
    int start = path.LastIndexOf('.');
    string suffix = path.Substring(start);
    string fileName = path.Remove(start);
    int i=0;
    while (File.Exists(path))
    {
        i++;
        path = fileName + "_" + i + suffix;
    }
    uploadFile.SaveAs(path);

    start = path.LastIndexOf("\\Uploaded\\");
    path = "Ci.m" + path.Substring(start) + ",";
    imgContainer.ImagesUrlsString += path;
}
protected void remImgButton_Click(object sender, EventArgs e)
{
    string displayedImg = imgContainer.getDisplayedImageUrl();

    if (displayedImg.StartsWith("Images/Various/"))
    {
        warningTextBox.Text = "cannot remove this image because this is not \n" +
            "a cancellation image, but a message image";
        warningTextBox.Visible = true;
        return;
    }

    displayedImg = displayedImg.Replace("Images/","");
    displayedImg = displayedImg.Replace("/","\\");
    displayedImg = "Ci.m\\" + displayedImg;

    string currentUrlsString = imgContainer.ImagesUrlsString;
    currentUrlsString = currentUrlsString.Replace(displayedImg+",","");
    imgContainer.ImagesUrlsString = currentUrlsString;

}
protected void restoreImgButton_Click(object sender, EventArgs e)
{
    imgContainer.ImagesUrlsString = (string)Session["selectedCancelImage"];
```

```
}

protected void posNeg_Changed(object sender, EventArgs e)
{
   if (posNegRadioButtonList.SelectedValue.Equals("positive"))
   {
      frenchFieldTextBox.BackColor = System.Drawing.Color.White;
      frenchFieldTextBox.ForeColor = System.Drawing.Color.Black;
      ottoFieldTextBox.BackColor = System.Drawing.Color.White;
      ottoFieldTextBox.ForeColor = System.Drawing.Color.Black;
      shapeButton.Enabled = false;
      shapeEnteredTextBox.Enabled = false;
      shapeEnteredTextBox.Text = "";
      exprButton.Enabled = false;
      exprEnteredTextBox.Enabled = false;
      exprEnteredTextBox.Text = "";
   }
   else
   {
      frenchFieldTextBox.BackColor = System.Drawing.Color.Black;
      frenchFieldTextBox.ForeColor = System.Drawing.Color.White;
      ottoFieldTextBox.BackColor = System.Drawing.Color.Black;
      ottoFieldTextBox.ForeColor = System.Drawing.Color.White;
      shapeButton.Enabled = true;
      shapeEnteredTextBox.Enabled = true;
      exprButton.Enabled = true;
      exprEnteredTextBox.Enabled = true;
   }
}

protected void shapeButton_Click(object sender, EventArgs e)
{
   MultiView1.ActiveViewIndex = 1;
}
protected void exprButton_Click(object sender, EventArgs e)
{
   MultiView1.ActiveViewIndex = 2;
}

protected void defaultView_Activate(object sender, EventArgs e)
{
}

//
//shape view
protected void shapeView_Activate(object sender, EventArgs e)
{
   greenPanel.Enabled = false;
   bluePanel.Enabled = false;
   specialShapeRB.Checked = false;
   shapeRBL.ClearSelection();
   ringCheckBox.Visible = false;
   circOvalCB.Visible = false;
   ovalRectCB.Visible = false;
   dontKnowRB.Visible = false;

   noBorderRB.Checked = false;
   crescentRB.Checked = false;
   ringRB.Checked = false;
   ringCheckBox.Checked = false;
```

```
    laurelRB.Checked = false;
    rosetteRB.Checked = false;
    starRBL.ClearSelection();

    specialShapeRBL.ClearSelection();

    ViewState["shape"] = "[corab]";
    ViewState["border"] = "[ncrult]";
    ViewState["star"] = "[ynd]";
}

protected void specialShapeRBLchanged(object sender, EventArgs e)
{
    switch (specialShapeRBL.SelectedIndex)
    {
        case 0:
            specialImage.ImageUrl = "Images/CWBrEx/no_ex_h.bmp";
            shapeTextBox.Text = "1slx";
            break;
        case 1:
            specialImage.ImageUrl = "Images/CWBrEx/SM_h.bmp";
            shapeTextBox.Text = "1sox";
            break;
        case 2:
            specialImage.ImageUrl = "Images/CWBrEx/SBM2_h.bmp";
            shapeTextBox.Text = "1smx";
            break;
        case 3:
            specialImage.ImageUrl = "Images/CWBrEx/NegHexagonal_h.bmp";
            shapeTextBox.Text = "1sqx";
            break;
    }
}
protected void speciaShapeChecked(object sender, EventArgs e)
{
    greenPanel.Enabled = false;
    shapeRBL.ClearSelection();
    bluePanel.Enabled = true;

    switch (specialShapeRBL.SelectedIndex)
    {
        case 0:
            shapeTextBox.Text = "1slx";
            break;
        case 1:
            shapeTextBox.Text = "1sox";
            break;
        case 2:
            shapeTextBox.Text = "1smx";
            break;
        case 3:
            shapeTextBox.Text = "1sqx";
            break;
        default:
            shapeTextBox.Text = "1s_x";
            break;
    }
}

protected void shapeRBLSelected(object sender, EventArgs e)
```

```
{
   bluePanel.Enabled = false;
   specialShapeRB.Checked = false;
   greenPanel.Enabled = true;
   if (shapeRBL.SelectedValue.Equals("Circular"))
   {
      circOvalCB.Visible = true;
      circOvalCB.Checked = false;
      ViewState["shape"] = "c";
      ovalRectCB.Visible = false;
   }
   else if (shapeRBL.SelectedValue.Equals("Oval"))
   {
      circOvalCB.Visible = true;
      circOvalCB.Checked = false;
      ovalRectCB.Visible = true;
      ovalRectCB.Checked = false;
      ViewState["shape"] = "o";
   }
   else if (shapeRBL.SelectedValue.Equals("Rectangular"))
   {
      circOvalCB.Visible = false;
      ovalRectCB.Visible = true;
      ovalRectCB.Checked = false;
      ViewState["shape"] = "r";
   }

   shapeTextBox.Text = "1" + ViewState["shape"] +
               ViewState["border"] + ViewState["star"];
}
protected void circOvalChecked(object sender, EventArgs e)
{
   if (circOvalCB.Checked)
   {
      ovalRectCB.Checked = false;
      ViewState["shape"] = "a";
   }
   else
   {
      if(shapeRBL.SelectedValue.Equals("Circular"))
         ViewState["shape"] = "c";
      else
         ViewState["shape"] = "o";
   }
   shapeTextBox.Text = "1" + ViewState["shape"] +
               ViewState["border"] + ViewState["star"];
}
protected void ovalRectChecked(object sender, EventArgs e)
{
   if (ovalRectCB.Checked)
   {
      circOvalCB.Checked = false;
      ViewState["shape"] = "b";
   }
   else
   {
      if (shapeRBL.SelectedValue.Equals("Rectangular"))
         ViewState["shape"] = "r";
      else
         ViewState["shape"] = "o";
```

```
    }
    shapeTextBox.Text = "1" + ViewState["shape"] +
                ViewState["border"] + ViewState["star"];
}

protected void shapeBorderChanged(object sender, EventArgs e)
{
    noBorderRB.Checked = false;
    crescentRB.Checked = false;
    ringRB.Checked = false;
    laurelRB.Checked = false;
    rosetteRB.Checked = false;
    dontKnowRB.Checked = false;

    RadioButton rb = (RadioButton)sender;
    rb.Checked = true;
    if (rb.ID.Equals("ringRB"))
        ringCheckBox.Visible = true;
    else
        ringCheckBox.Visible = false;

    switch (rb.ID)
    {
        case "noBorderRB":
            ViewState["border"] = "n";
            break;
        case "crescentRB":
            ViewState["border"] = "c";
            break;
        case "ringRB":
            if (ringCheckBox.Checked)
                ViewState["border"] = "u";
            else
                ViewState["border"] = "r";
            break;
        case "laurelRB":
            ViewState["border"] = "l";
            break;
        case "rosetteRB":
            ViewState["border"] = "t";
            break;
        default:
            ViewState["border"] = "?";
            break;
    }

    shapeTextBox.Text = "1" + ViewState["shape"] +
            ViewState["border"] + ViewState["star"];
}
protected void ringUnclearChecked(object sender, EventArgs e)
{
    if (ringCheckBox.Checked)
        ViewState["border"] = "u";
    else
        ViewState["border"] = "r";

    shapeTextBox.Text = "1" + ViewState["shape"] +
            ViewState["border"] + ViewState["star"];
}
```

```
protected void starChanged(object sender, EventArgs e)
{
   switch (starRBL.SelectedValue)
   {
      case "YES":
         ViewState["star"] = "y";
         break;
      case "NO":
         ViewState["star"] = "n";
         break;
      case "unclear":
         ViewState["star"] = "d";
         break;
   }
   shapeTextBox.Text = "1" + ViewState["shape"] +
         ViewState["border"] + ViewState["star"];
}

protected void shapeBackButton_Click(object sender, EventArgs e)
{
   MultiView1.ActiveViewIndex = 0;
}
protected void shapeEnter_Click(object sender, EventArgs e)
{
   if (shapeTextBox.Text.Contains("?"))
   {

   }
   shapeEnteredTextBox.Text = shapeTextBox.Text;
   MultiView1.ActiveViewIndex = 0;
}

//
//expression view
protected void exprView_Activate(object sender, EventArgs e)
{

}

protected void exprBackButton_Click(object sender, EventArgs e)
{
   MultiView1.ActiveViewIndex = 0;
}
protected void exprEnter_Click(object sender, EventArgs e)
{
   exprEnteredTextBox.Text = scanExpression();
   MultiView1.ActiveViewIndex = 0;
}

private string scanExpression()
{
   string temp = "2";
   if (anCanibExistRBL.SelectedIndex == 1)
      temp += "00";
   else
   {
      switch (anCanibRBL1.SelectedIndex)
      {
         case 0:
            temp += "t";
```

```
            break;
        case 1:
            temp += "m";
            break;
        case 2:
            temp += "b";
            break;
        case 3:
            temp += "w";
            break;
    }
    switch (anCanibRBL2.SelectedIndex)
    {
        case 0:
            temp += "t";
            break;
        case 1:
            temp += "m";
            break;
        case 2:
            temp += "b";
            break;
        case 3:
            temp += "w";
            break;
    }
}
if (telegrafExistRBL.SelectedIndex == 1)
    temp += "00";
else
{
    switch (telegrafRBL1.SelectedIndex)
    {
        case 0:
            temp += "t";
            break;
        case 1:
            temp += "m";
            break;
        case 2:
            temp += "b";
            break;
        case 3:
            temp += "w";
            break;
    }
    switch (telegrafRBL2.SelectedIndex)
    {
        case 0:
            temp += "t";
            break;
        case 1:
            temp += "m";
            break;
        case 2:
            temp += "b";
            break;
        case 3:
            temp += "w";
            break;
```

```
    }
}
if (postaExistRBL.SelectedIndex == 1)
    temp += "00";
else
{
    switch (postaRBL1.SelectedIndex)
    {
        case 0:
            temp += "t";
            break;
        case 1:
            temp += "m";
            break;
        case 2:
            temp += "b";
            break;
        case 3:
            temp += "w";
            break;
    }
    switch (postaRBL2.SelectedIndex)
    {
        case 0:
            temp += "t";
            break;
        case 1:
            temp += "m";
            break;
        case 2:
            temp += "b";
            break;
        case 3:
            temp += "w";
            break;
    }
}
if (subesiExistRBL.SelectedIndex == 1)
    temp += "00";
else
{
    switch (subesiRBL1.SelectedIndex)
    {
        case 0:
            temp += "t";
            break;
        case 1:
            temp += "m";
            break;
        case 2:
            temp += "b";
            break;
        case 3:
            temp += "w";
            break;
    }
    switch (subesiRBL2.SelectedIndex)
    {
        case 0:
            temp += "t";
```

```
                break;
            case 1:
                temp += "m";
                break;
            case 2:
                temp += "b";
                break;
            case 3:
                temp += "w";
                break;
        }
    }
    if (hanesiExistRBL.SelectedIndex == 1)
        temp += "00";
    else
    {
        switch (hanesiRBL1.SelectedIndex)
        {
            case 0:
                temp += "t";
                break;
            case 1:
                temp += "m";
                break;
            case 2:
                temp += "b";
                break;
            case 3:
                temp += "w";
                break;
        }
        switch (hanesiRBL2.SelectedIndex)
        {
            case 0:
                temp += "t";
                break;
            case 1:
                temp += "m";
                break;
            case 2:
                temp += "b";
                break;
            case 3:
                temp += "w";
                break;
        }
    }
    return temp;
}

protected void image_Click(object sender, ImageClickEventArgs e)
{
    int i;
    string addr;
    ImageButton ib = (ImageButton)sender;
    switch (ib.ID)
    {
        case "anCanibImage1":
            i = (int)ViewState["an1"] + 1;
            addr = ViewState["prefix"] + "an/an" + i + ".bmp";
```

```
    if (File.Exists(addr))
    {
        anCanibImage1.ImageUrl = "Images/NegSamples/an/an" +
            i + ".bmp";
        ViewState["an1"] = i;
    }
    else
    {
        anCanibImage1.ImageUrl = "Images/NegSamples/an/an1.bmp";
        ViewState["an1"] = 1;
    }
    break;
case "anCanibImage2":
    i = (int)ViewState["an2"] + 1;
    addr = ViewState["prefix"] + "janib/janib" + i + ".bmp";
    if (File.Exists(addr))
    {
        anCanibImage2.ImageUrl = "Images/NegSamples/janib/janib" +
            i + ".bmp";
        ViewState["an2"] = i;
    }
    else
    {
        anCanibImage2.ImageUrl = "Images/NegSamples/janib/janib1.bmp";
        ViewState["an2"] = 1;
    }
    break;
case "telegrafImage1":
    i = (int)ViewState["tel1"] + 1;
    addr = ViewState["prefix"] + "telegr/telegr" + i + ".bmp";
    if (File.Exists(addr))
    {
        telegrafImage1.ImageUrl = "Images/NegSamples/telegr/telegr" +
            i + ".bmp";
        ViewState["tel1"] = i;
    }
    else
    {
        telegrafImage1.ImageUrl = "Images/NegSamples/telegr/telegr1.bmp";
        ViewState["tel1"] = 1;
    }
    break;
case "telegrafImage2":
    i = (int)ViewState["tel2"] + 1;
    addr = ViewState["prefix"] + "af/af" + i + ".bmp";
    if (File.Exists(addr))
    {
        telegrafImage2.ImageUrl = "Images/NegSamples/af/af" +
            i + ".bmp";
        ViewState["tel2"] = i;
    }
    else
    {
        telegrafImage2.ImageUrl = "Images/NegSamples/af/af1.bmp";
        ViewState["tel2"] = 1;
    }
    break;
case "postaImage1":
    i = (int)ViewState["post1"] + 1;
    addr = ViewState["prefix"] + "po/po" + i + ".bmp";
```

```
        if (File.Exists(addr))
        {
            postaImage1.ImageUrl = "Images/NegSamples/po/po" +
                i + ".bmp";
            ViewState["post1"] = i;
        }
        else
        {
            postaImage1.ImageUrl = "Images/NegSamples/po/po1.bmp";
            ViewState["post1"] = 1;
        }
        break;
    case "postaImage2":
        i = (int)ViewState["post2"] + 1;
        addr = ViewState["prefix"] + "sta/sta" + i + ".bmp";
        if (File.Exists(addr))
        {
            postaImage2.ImageUrl = "Images/NegSamples/sta/sta" +
                i + ".bmp";
            ViewState["post2"] = i;
        }
        else
        {
            postaImage2.ImageUrl = "Images/NegSamples/sta/sta1.bmp";
            ViewState["post2"] = 1;
        }
        break;
    case "subesiImage1":
        i = (int)ViewState["sub1"] + 1;
        addr = ViewState["prefix"] + "sube/sube" + i + ".bmp";
        if (File.Exists(addr))
        {
            subesiImage1.ImageUrl = "Images/NegSamples/sube/sube" +
                i + ".bmp";
            ViewState["sub1"] = i;
        }
        else
        {
            subesiImage1.ImageUrl = "Images/NegSamples/sube/sube1.bmp";
            ViewState["sub1"] = 1;
        }
        break;
    case "subesiImage2":
        i = (int)ViewState["sub2"] + 1;
        addr = ViewState["prefix"] + "si/si" + i + ".bmp";
        if (File.Exists(addr))
        {
            subesiImage2.ImageUrl = "Images/NegSamples/si/si" +
                i + ".bmp";
            ViewState["sub2"] = i;
        }
        else
        {
            subesiImage2.ImageUrl = "Images/NegSamples/si/si1.bmp";
            ViewState["sub2"] = 1;
        }
        break;
    case "hanesiImage1":
        i = (int)ViewState["han1"] + 1;
        addr = ViewState["prefix"] + "ha/ha" + i + ".bmp";
```

```
            if (File.Exists(addr))
            {
               hanesiImage1.ImageUrl = "Images/NegSamples/ha/ha" +
                     i + ".bmp";
               ViewState["han1"] = i;
            }
            else
            {
               hanesiImage1.ImageUrl = "Images/NegSamples/ha/ha1.bmp";
               ViewState["han1"] = 1;
            }
            break;
         case "hanesiImage2":
            i = (int)ViewState["han2"] + 1;
            addr = ViewState["prefix"] + "neyi/neyi" + i + ".bmp";
            if (File.Exists(addr))
            {
               hanesiImage2.ImageUrl = "Images/NegSamples/neyi/neyi" +
                     i + ".bmp";
               ViewState["han2"] = i;
            }
            else
            {
               hanesiImage2.ImageUrl = "Images/NegSamples/neyi/neyi1.bmp";
               ViewState["han2"] = 1;
            }
            break;
      }
   }

   protected void anCanibExist_Changed(object sender, EventArgs e)
   {
      if (anCanibExistRBL.SelectedIndex == 0)
      {
         anCanibRBL1.Visible = true;
         anCanibRBL2.Visible = true;
      }
      else
      {
         anCanibRBL1.Visible = false;
         anCanibRBL2.Visible = false;
      }

   }
   protected void telegrafExist_Changed(object sender, EventArgs e)
   {

      if (telegrafExistRBL.SelectedIndex == 0)
      {
         telegrafRBL1.Visible = true;
         telegrafRBL2.Visible = true;
      }
      else
      {
         telegrafRBL1.Visible = false;
         telegrafRBL2.Visible = false;
      }
   }
   protected void postaExist_Changed(object sender, EventArgs e)
   {
```

```
        if (postaExistRBL.SelectedIndex == 0)
        {
            postaRBL1.Visible = true;
            postaRBL2.Visible = true;
        }
        else
        {
            postaRBL1.Visible = false;
            postaRBL2.Visible = false;
        }
    }
    protected void subesiExist_Changed(object sender, EventArgs e)
    {
        if (subesiExistRBL.SelectedIndex == 0)
        {
            subesiRBL1.Visible = true;
            subesiRBL2.Visible = true;
        }
        else
        {
            subesiRBL1.Visible = false;
            subesiRBL2.Visible = false;
        }
    }
    protected void hanesiExist_Changed(object sender, EventArgs e)
    {
        if (hanesiExistRBL.SelectedIndex == 0)
        {
            hanesiRBL1.Visible = true;
            hanesiRBL2.Visible = true;
        }
        else
        {
            hanesiRBL1.Visible = false;
            hanesiRBL2.Visible = false;
        }
    }
}
```

A15) αρχείο shapeSearch.aspx :

```
<%@ Page Language="C#" AutoEventWireup="true" CodeFile="shapeSearch.aspx.cs"
Inherits="shapeSearch" %>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml" >
<head runat="server">
    <title>search based on the shape of the cancellation</title>
</head>
<body style="background-color:Gray">
    <form id="form1" runat="server">
    <div>

    <table width="100%">
    <tr>
        <td colspan="4" align="center">
```

```
        <asp:Label ID="titleLabel" Font-Bold="true" runat="server" Text="search based on the shape of the
cancellation"></asp:Label>
      </td>
      <td>
        <asp:TextBox ID="TextBox1" runat="server"></asp:TextBox>
      </td>
    </tr>
    <tr>
      <td style="width:20%">      </td>
      <td style="width:20%">      </td>
      <td style="width:20%">      </td>
      <td style="width:20%">      </td>
      <td style="width:20%">      </td>
    </tr>
    <tr>
      <td colspan="3" align="center">
      <asp:RadioButtonList ID="shapeRBL" RepeatDirection="Horizontal" runat="server"
AutoPostBack="True" OnSelectedIndexChanged="shapeRBLSelected">
        <asp:ListItem>Circular</asp:ListItem>
        <asp:ListItem>Oval</asp:ListItem>
        <asp:ListItem>Rectangular</asp:ListItem>
      </asp:RadioButtonList><br />
      <asp:Panel ID="greenPanel" BorderColor="green" BorderStyle="Solid" BorderWidth="5px"
runat="server">
        <table>
        <tr>
          <td style="width:33%" align="right">
            <asp:CheckBox ID="circOvalCB" TextAlign="Left" Text="circular/oval unclear" runat="server"
/>
          </td>
          <td style="width:33%"></td>
          <td style="width:34%" align="left">
            <asp:CheckBox ID="ovalRectCB" TextAlign="Right" Text="oval/rectangular unclear"
runat="server" />
          </td>
        </tr>
        <tr>
          <td>
            <asp:Image ID="noBorderImage" ImageUrl="Images/Shapes/n1_h.bmp" runat="server" />
          </td>
          <td>
            <asp:Image ID="crescentImage" ImageUrl="Images/Shapes/c1_h.bmp" runat="server" />
          </td>
          <td>
            <asp:Image ID="ringImage" ImageUrl="Images/Shapes/r1_h.bmp" runat="server" />
          </td>
        </tr>
        <tr>
          <td>
            <asp:RadioButton ID="noBorderRB" Text="no border" runat="server" AutoPostBack="True"
OnCheckedChanged="shapeBorderChanged" />
          </td>
          <td>
            <asp:RadioButton ID="crescentRB" Text="crescent" runat="server" AutoPostBack="True"
OnCheckedChanged="shapeBorderChanged" />
          </td>
          <td>
            <asp:RadioButton ID="ringRB" Text="ring around" runat="server" AutoPostBack="True"
OnCheckedChanged="shapeBorderChanged" />
            <br />
```

```
            <asp:CheckBox ID="ringCheckBox" Text="ring not clear" runat="server" />
          </td>
        </tr>
        <tr>
          <td>
            <asp:Image ID="laurelImage" ImageUrl="Images/Shapes/l1_h.bmp" runat="server" />
          </td>
          <td>
            <asp:Image ID="rosetteImage" ImageUrl="Images/Shapes/t1_h.bmp" runat="server" />
          </td>
          <td>
            star on top?<br />
            <asp:RadioButtonList ID="starRBL" runat="server" AutoPostBack="True"
OnSelectedIndexChanged="starChanged">
              <asp:ListItem>YES</asp:ListItem>
              <asp:ListItem>NO</asp:ListItem>
              <asp:ListItem>unclear</asp:ListItem>
            </asp:RadioButtonList>
          </td>
        </tr>
        <tr>
          <td>
            <asp:RadioButton ID="laurelRB" Text="laurel" runat="server" AutoPostBack="True"
OnCheckedChanged="shapeBorderChanged" />
          </td>
          <td>
            <asp:RadioButton ID="rosetteRB" Text="rosette edge" runat="server" AutoPostBack="True"
OnCheckedChanged="shapeBorderChanged" />
          </td>
          <td></td>
        </tr>
        <tr>
          <td colspan="2" align="center">
            <asp:RadioButton ID="dontKnowRB" Text="don't know" runat="server" AutoPostBack="True"
OnCheckedChanged="shapeBorderChanged" />
          </td>
          <td>
            <asp:Button ID="searchButton" runat="server" Text="search"
OnCommand="searchButton_Click" />
          </td>
        </tr>
        </table>
      </asp:Panel>
      </td>

      <td colspan="2" align="center">
        <asp:Button ID="backButton" runat="server" Text="Back" PostBackUrl="~/Default.aspx" />
        <br /><br /><br />
        <asp:RadioButton ID="specialShapeRB" Text="Special Shape" runat="server" AutoPostBack="True"
OnCheckedChanged="speciaShapeChecked" /><br />
      <asp:Panel ID="bluePanel" BorderColor="blue" BorderStyle="Solid" BorderWidth="5px"
runat="server">
        <table>
        <tr>
          <td style="width:50%"></td>
          <td style="width:50%"></td>
        </tr>
        <tr>
          <td>
```

```
            <asp:RadioButtonList ID="specialShapeRBL" runat="server" AutoPostBack="True"
OnSelectedIndexChanged="specialShapeRBLchanged">
                <asp:ListItem>circular latin</asp:ListItem>
                <asp:ListItem>circular ottoman</asp:ListItem>
                <asp:ListItem>circular mixed</asp:ListItem>
                <asp:ListItem>other shapes</asp:ListItem>
            </asp:RadioButtonList>
          </td>
          <td>
            <asp:Image ID="specialImage" runat="server" />
            <br/>example pic
          </td>
        </tr>
        <tr>
          <td>            </td>
          <td>
            <asp:Button ID="specialSearchButton" runat="server" Text="search"
OnClick="specialSearch_Click" />
          </td>
        </tr>
        </table>
      </asp:Panel>
        <br /><br /><br />
        <asp:Button ID="exprButton" runat="server" Text="search expression" ToolTip="keep current shape
choice and add expression choice (new form)" OnClick="exprButton_Click" />
      </td>
  </tr>
  </table>

  </div>
  </form>
</body>
</html>
```

A16) αρχείο shapeSearch.aspx.cs :

```
using System;
using System.Data;
using System.Configuration;
using System.Collections;
using System.Web;
using System.Web.Security;
using System.Web.UI;
using System.Web.UI.WebControls;
using System.Web.UI.WebControls.WebParts;
using System.Web.UI.HtmlControls;

public partial class shapeSearch : System.Web.UI.Page
{
  protected void Page_Load(object sender, EventArgs e)
  {
    if (!IsPostBack)
    {
      greenPanel.Enabled = false;
      bluePanel.Enabled = false;

      //needed only for inserting new cancellation
      ringCheckBox.Visible = false;
```

```
        circOvalCB.Visible = false;
        ovalRectCB.Visible = false;
    }
}

protected void specialShapeRBLchanged(object sender, EventArgs e)
{
    switch (specialShapeRBL.SelectedIndex)
    {
        case 0:
            specialImage.ImageUrl = "Images/CWBrEx/no_ex_h.bmp";
            break;
        case 1:
            specialImage.ImageUrl = "Images/CWBrEx/SM_h.bmp";
            break;
        case 2:
            specialImage.ImageUrl = "Images/CWBrEx/SBM2_h.bmp";
            break;
        case 3:
            specialImage.ImageUrl = "Images/CWBrEx/NegHexagonal_h.bmp";
            break;
    }
    updateShapeString();
}

protected void speciaShapeChecked(object sender, EventArgs e)
{
    greenPanel.Enabled = false;
    shapeRBL.ClearSelection();
    bluePanel.Enabled = true;

    updateShapeString();
}

protected void shapeRBLSelected(object sender, EventArgs e)
{
    bluePanel.Enabled = false;
    specialShapeRB.Checked = false;
    greenPanel.Enabled = true;

    updateShapeString();
}

protected void shapeBorderChanged(object sender, EventArgs e)
{
    noBorderRB.Checked = false;
    crescentRB.Checked = false;
    ringRB.Checked = false;
    laurelRB.Checked = false;
    rosetteRB.Checked = false;
    dontKnowRB.Checked = false;

    RadioButton rb = (RadioButton)sender;
    rb.Checked = true;

    switch (rb.ID)
    {
        case "noBorderRB":
            ViewState["border"] = "n";
            ViewState["borderText"] = "without border and ";
```

```
            break;
        case "crescentRB":
            ViewState["border"] = "c";
            ViewState["borderText"] = "with crescent border and ";
            break;
        case "ringRB":
            ViewState["border"] = "[ru]";
            ViewState["borderText"] = "with ring border and ";
            break;
        case "laurelRB":
            ViewState["border"] = "l";
            ViewState["borderText"] = "with laurel border and ";
            break;
        case "rosetteRB":
            ViewState["border"] = "t";
            ViewState["borderText"] = "with rosette shaped border and ";
            break;
        case "dontKnowRB":
            ViewState["border"] = "[encrult]";
            ViewState["borderText"] = "with unspecified border and ";
            break;
    }

    TextBox1.Text = "1" + ViewState["shape"] +
            ViewState["border"] + ViewState["star"];
}

protected void starChanged(object sender, EventArgs e)
{
    switch (starRBL.SelectedValue)
    {
        case "YES":
            ViewState["star"] = "[yd]";
            ViewState["starText"] = "star on top";
            break;
        case "NO":
            ViewState["star"] = "[nd]";
            ViewState["starText"] = "no star on top";
            break;
        case "unclear":
            ViewState["star"] = "[ynd]";
            ViewState["starText"] = "unspecified if star on top";
            break;
    }

    TextBox1.Text = "1" + ViewState["shape"] +
            ViewState["border"] + ViewState["star"];
}

protected void searchButton_Click(object sender, CommandEventArgs e)
{
    setSourceAsColours(TextBox1.Text);
    Session["filterText"] = "";
    Session["searchText"] = (string)ViewState["shapeText"] +
        ViewState["borderText"] + ViewState["starText"];
    Response.Redirect("Default.aspx");
}
protected void specialSearch_Click(object sender, EventArgs e)
{
    setSourceAsColours(TextBox1.Text);
```

```
    Session["filterText"] = "";
    switch (specialShapeRBL.SelectedIndex)
    {
        case 0:
            Session["searchText"] = " negative cancellation(s) listed " +
                "of special shape and with only latin text";
            break;
        case 1:
            Session["searchText"] = " negative cancellation(s) listed " +
                "of special shape and with only ottoman text";
            break;
        case 2:
            Session["searchText"] = " negative cancellation(s) listed " +
                "of special shape and with mixed latin/ottoman text";
            break;
        case 3:
            Session["searchText"] = " negative cancellation(s) listed " +
                "of special shape(other, no circular shape)";
            break;
        default:
            Session["searchText"] = " negative cancellation(s) listed " +
                "of special shape";
            break;
    }
    Response.Redirect("Default.aspx");
}

protected void exprButton_Click(object sender, EventArgs e)
{
    if (TextBox1.Text.Equals(string.Empty))
    {
        Session["shapeSearchString"] = "%";
        Session["shapeSearchText"] = " negative cancellation(s) listed " +
            "with unspecified shape and";
    }
    else
    {
        Session["shapeSearchString"] = TextBox1.Text;

        if (greenPanel.Enabled)
        {
            Session["shapeSearchText"] = (string)ViewState["shapeText"] +
                ViewState["borderText"] + ViewState["starText"];
        }
        else if (bluePanel.Enabled)
        {
            switch (specialShapeRBL.SelectedIndex)
            {
                case 0:
                    Session["shapeSearchText"] = " negative cancellation(s) listed " +
                        "of special shape and with only latin text";
                    break;
                case 1:
                    Session["shapeSearchText"] = " negative cancellation(s) listed " +
                        "of special shape and with only ottoman text";
                    break;
                case 2:
                    Session["shapeSearchText"] = " negative cancellation(s) listed " +
                        "of special shape and with mixed latin/ottoman text";
                    break;
```

```
            case 3:
                Session["shapeSearchText"] = " negative cancellation(s) listed " +
                    "of special shape(other, no circular shape)";
                break;
            default:
                Session["shapeSearchText"] = " negative cancellation(s) listed " +
                    "of special shape";
                break;
        }
    }
  }
  Response.Redirect("expressionSearch.aspx");
}

private void updateShapeString()
{
  if (specialShapeRB.Checked)     //blue panel
  {
    switch (specialShapeRBL.SelectedIndex)
    {
      case 0:
        TextBox1.Text = "1slx";
        break;
      case 1:
        TextBox1.Text = "1sox";
        break;
      case 2:
        TextBox1.Text = "1smx";
        break;
      case 3:
        TextBox1.Text = "1sqx";
        break;
      default:
        TextBox1.Text = "1s_x";
        break;
    }
    return;
  }

  //green panel
  switch (shapeRBL.SelectedValue)
  {
    case "Circular":
      ViewState["shape"] = "[ca]";
      ViewState["shapeText"] = " circular negative cancellation(s) listed ";
      break;
    case "Oval":
      ViewState["shape"] = "[oab]";
      ViewState["shapeText"] = " oval negative cancellation(s) listed ";
      break;
    case "Rectangular":
      ViewState["shape"] = "[rb]";
      ViewState["shapeText"] = " rectangular negative cancellation(s) listed ";
      break;
    default:
      ViewState["shape"] = "[corab]";
      ViewState["shapeText"] = " of unclear shape negative cancellation(s) listed ";
      break;
  }
```

```
    if (noBorderRB.Checked)
    {
        ViewState["border"] = "n";
        ViewState["borderText"] = "without border and ";
    }
    else if (crescentRB.Checked)
    {
        ViewState["border"] = "c";
        ViewState["borderText"] = "with crescent border and ";
    }
    else if (ringRB.Checked)
    {
        ViewState["border"] = "[ru]";
        ViewState["borderText"] = "with ring border and ";
    }
    else if (laurelRB.Checked)
    {
        ViewState["border"] = "l";
        ViewState["borderText"] = "with laurel border and ";
    }
    else if (rosetteRB.Checked)
    {
        ViewState["border"] = "t";
        ViewState["borderText"] = "with rosette shaped border and ";
    }
    else
    {
        ViewState["border"] = "[enrcltu]";    //e?????
        ViewState["borderText"] = "with unspecified border and ";
    }

    switch (starRBL.SelectedValue)
    {
        case "YES":
            ViewState["star"] = "[yd]";
            ViewState["starText"] = "star on top";
            break;
        case "NO":
            ViewState["star"] = "[nd]";
            ViewState["starText"] = "no star on top";
            break;
        case "unclear":
            ViewState["star"] = "[ynd]";
            ViewState["starText"] = "unspecified if star on top";
            break;
        default:
            ViewState["star"] = "[ynd]";
            ViewState["starText"] = "unspecified if star on top";
            break;
    }

    TextBox1.Text = "1" + ViewState["shape"] +
            ViewState["border"] + ViewState["star"];
}

private void setSourceAsColours( string colours)
{
    Session["selectMethod"] = "GetDataByColours";
    Session["name0"] = "Colours";
    Session["type0"] = TypeCode.String;
```

```
        Session["value0"] = colours;
        Session["selectParamCount"] = 1;
        Session["filterExpr"] = "null";
    }
}
```

A17) αρχείο ThumbNails.aspx :

```
<%@ Page Language="C#" AutoEventWireup="true" CodeFile="Thumbnails.aspx.cs" Inherits="Thumbnails"
%>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml" >
<head runat="server">
    <title>Cancellations Images</title>
</head>
<body style="background-color:Gray">
    <form id="form1" runat="server">
    <div>

    <table width="100%">
    <tr>
        <td colspan="2" align="left">
            <asp:Label ID="titleLabel" runat="server" Text="titleLabel"></asp:Label>
        </td>
        <td colspan="2" align="left">
            <asp:Button ID="backButton" runat="server" Text="Back to search" PostBackUrl="~/Default.aspx" />
        </td>
    </tr>
    <tr>
        <td colspan="4" align="center">
            <asp:Button ID="previousButton" runat="server" Text="Previous" OnClick="previous_Click" />
            <asp:Label ID="pageLabel" runat="server" Text="pageLabel"></asp:Label>
            <asp:Button ID="nextButton" runat="server" Text="Next" OnClick="next_Click" />
            <asp:Button ID="firstButton" runat="server" Text="first" OnClick="first_Click" />
            <asp:Button ID="middleButton" runat="server" Text="middle" OnClick="middle_Click" />
            <asp:Button ID="lastButton" runat="server" Text="last" OnClick="last_Click" />
        </td>
    </tr>
    <tr>
        <td>
            <asp:Image ID="Image1" runat="server" /><br />
            <asp:LinkButton ID="LinkButton1" runat="server" OnClick="LinkButton_Click"></asp:LinkButton>
        </td>
        <td>
            <asp:Image ID="Image2" runat="server" /><br />
            <asp:LinkButton ID="LinkButton2" runat="server" OnClick="LinkButton_Click"></asp:LinkButton>
        </td>
        <td>
            <asp:Image ID="Image3" runat="server" /><br />
            <asp:LinkButton ID="LinkButton3" runat="server" OnClick="LinkButton_Click"></asp:LinkButton>
        </td>
        <td>
            <asp:Image ID="Image4" runat="server" /><br />
            <asp:LinkButton ID="LinkButton4" runat="server" OnClick="LinkButton_Click"></asp:LinkButton>
        </td>
```

```
    </tr>
    <tr>
      <td>
        <asp:Image ID="Image5" runat="server" /><br />
        <asp:LinkButton ID="LinkButton5" runat="server" OnClick="LinkButton_Click"></asp:LinkButton>
      </td>
      <td>
        <asp:Image ID="Image6" runat="server" /><br />
        <asp:LinkButton ID="LinkButton6" runat="server" OnClick="LinkButton_Click"></asp:LinkButton>
      </td>
      <td>
        <asp:Image ID="Image7" runat="server" /><br />
        <asp:LinkButton ID="LinkButton7" runat="server" OnClick="LinkButton_Click"></asp:LinkButton>
      </td>
      <td>
        <asp:Image ID="Image8" runat="server" /><br />
        <asp:LinkButton ID="LinkButton8" runat="server" OnClick="LinkButton_Click"></asp:LinkButton>
      </td>
    </tr>
    <tr>
      <td>
        <asp:Image ID="Image9" runat="server" /><br />
        <asp:LinkButton ID="LinkButton9" runat="server" OnClick="LinkButton_Click"></asp:LinkButton>
      </td>
      <td>
        <asp:Image ID="Image10" runat="server" /><br />
        <asp:LinkButton ID="LinkButton10" runat="server"
OnClick="LinkButton_Click"></asp:LinkButton>
      </td>
      <td>
        <asp:Image ID="Image11" runat="server" /><br />
        <asp:LinkButton ID="LinkButton11" runat="server"
OnClick="LinkButton_Click"></asp:LinkButton>
      </td>
      <td>
        <asp:Image ID="Image12" runat="server" /><br />
        <asp:LinkButton ID="LinkButton12" runat="server"
OnClick="LinkButton_Click"></asp:LinkButton>
      </td>
    </tr>

    </table>

        <asp:AccessDataSource ID="AccessDataSource1" runat="server"
DataFile="~/App_Data/Cancel_Alexandros_Apr09.mdb" SelectCommand="SELECT ID, [Image] FROM
Cancellations WHERE ([Image] IS NOT NULL) AND ([Image] <> ' ')"></asp:AccessDataSource>

    </div>
    </form>
</body>
</html>
```

A18) αρχείο Thumbnails.aspx.cs :

```
using System;
using System.Data;
using System.Configuration;
using System.Collections;
```

```
using System.Web;
using System.Web.Security;
using System.Web.UI;
using System.Web.UI.WebControls;
using System.Web.UI.WebControls.WebParts;
using System.Web.UI.HtmlControls;
using System.IO;

public partial class Thumbnails : System.Web.UI.Page
{
   private ArrayList idArray = new ArrayList();    //int
   private ArrayList imgArray = new ArrayList();   //string
   private int numberOfPages = 0;
   private int numberOfImages = 0;
   private int lastPage = 0;
   private int currentPage = 0;
   private int middlePage = 0;

   protected void Page_Load(object sender, EventArgs e)
   {
      retrieveImages();
      updateNumbers();
      if (!IsPostBack)
      {
         setButtonVisibility();

         titleLabel.Text = numberOfImages + " images found";
         pageLabel.Text = "Page " + (currentPage + 1) + " of " + numberOfPages;

         if (numberOfImages > 0)
            displayImages();
         else
            pageLabel.Text = "Page 0 of 0";
      }
   }

   private void updateNumbers()
   {
      if (ViewState["currentPage"] == null)
         ViewState["currentPage"] = 0;
      currentPage = (int)ViewState["currentPage"];

      numberOfPages = idArray.Count / 12;
      if (idArray.Count % 12 > 0)
         numberOfPages++;

      numberOfImages = idArray.Count;
      if (numberOfPages > 0)
         lastPage = numberOfPages - 1;
      else
         lastPage = 0;
      middlePage = lastPage / 2;
   }

   private void setButtonVisibility()
   {
      if (currentPage == 0)
      {
         firstButton.Visible = false;
         previousButton.Visible = false;
```

```
    }
    else
    {
        firstButton.Visible = true;
        previousButton.Visible = true;
    }

    if (currentPage == lastPage)
    {
        nextButton.Visible = false;
        lastButton.Visible = false;
    }
    else
    {
        nextButton.Visible = true;
        lastButton.Visible = true;
    }

    if (currentPage == middlePage)
        middleButton.Visible = false;
    else
        middleButton.Visible = true;
}

private void displayImages()
{
    Image1.ImageUrl = "";
    Image2.ImageUrl = "";
    Image3.ImageUrl = "";
    Image4.ImageUrl = "";
    Image5.ImageUrl = "";
    Image6.ImageUrl = "";
    Image7.ImageUrl = "";
    Image8.ImageUrl = "";
    Image9.ImageUrl = "";
    Image10.ImageUrl = "";
    Image11.ImageUrl = "";
    Image12.ImageUrl = "";
    LinkButton1.Text = "";
    LinkButton2.Text = "";
    LinkButton3.Text = "";
    LinkButton4.Text = "";
    LinkButton5.Text = "";
    LinkButton6.Text = "";
    LinkButton7.Text = "";
    LinkButton8.Text = "";
    LinkButton9.Text = "";
    LinkButton10.Text = "";
    LinkButton11.Text = "";
    LinkButton12.Text = "";

    int imgBase = currentPage * 12;
    if (imgBase < numberOfImages)
    {
        Image1.ImageUrl = "Images/" + imgArray[imgBase];
        LinkButton1.Text = "" + idArray[imgBase];
        imgBase++;
    }
    if (imgBase < numberOfImages)
    {
```

```
    Image2.ImageUrl = "Images/" + imgArray[imgBase];
    LinkButton2.Text = "" + idArray[imgBase];
    imgBase++;
}
if (imgBase < numberOfImages)
{
    Image3.ImageUrl = "Images/" + imgArray[imgBase];
    LinkButton3.Text = "" + idArray[imgBase];
    imgBase++;
}
if (imgBase < numberOfImages)
{
    Image4.ImageUrl = "Images/" + imgArray[imgBase];
    LinkButton4.Text = "" + idArray[imgBase];
    imgBase++;
}
if (imgBase < numberOfImages)
{
    Image5.ImageUrl = "Images/" + imgArray[imgBase];
    LinkButton5.Text = "" + idArray[imgBase];
    imgBase++;
}
if (imgBase < numberOfImages)
{
    Image6.ImageUrl = "Images/" + imgArray[imgBase];
    LinkButton6.Text = "" + idArray[imgBase];
    imgBase++;
}
if (imgBase < numberOfImages)
{
    Image7.ImageUrl = "Images/" + imgArray[imgBase];
    LinkButton7.Text = "" + idArray[imgBase];
    imgBase++;
}
if (imgBase < numberOfImages)
{
    Image8.ImageUrl = "Images/" + imgArray[imgBase];
    LinkButton8.Text = "" + idArray[imgBase];
    imgBase++;
}
if (imgBase < numberOfImages)
{
    Image9.ImageUrl = "Images/" + imgArray[imgBase];
    LinkButton9.Text = "" + idArray[imgBase];
    imgBase++;
}
if (imgBase < numberOfImages)
{
    Image10.ImageUrl = "Images/" + imgArray[imgBase];
    LinkButton10.Text = "" + idArray[imgBase];
    imgBase++;
}
if (imgBase < numberOfImages)
{
    Image11.ImageUrl = "Images/" + imgArray[imgBase];
    LinkButton11.Text = "" + idArray[imgBase];
    imgBase++;
}
if (imgBase < numberOfImages)
{
```

```
        Image12.ImageUrl = "Images/" + imgArray[imgBase];
        LinkButton12.Text = "" + idArray[imgBase];
        imgBase++;
    }
}

private void retrieveImages()
{
    if (IsPostBack)
    {
        idArray = (ArrayList)Session["idArray"];
        imgArray = (ArrayList)Session["imgArray"];
        return;
    }

    string filterExpr = (string)Session["filterExpr"];
    string command = "";
    switch ((string)Session["selectMethod"])
    {
        case "GetDataByPostOffice":
            command =
                "SELECT ID, Image " +
                "FROM Cancellations " +
                "WHERE ([Post Office] ='" +
                Session["value0"] +"')";
            if (filterExpr.Equals("null"))
                AccessDataSource1.SelectCommand = command;
            else
            {
                AccessDataSource1.SelectCommand = command +
                    " AND (" + filterExpr + ")";
            }
            break;
        case "GetData":
            if (filterExpr.Equals("null"))
                AccessDataSource1.SelectCommand =
                    "SELECT ID, Image " +
                    "FROM Cancellations";
            else
                AccessDataSource1.SelectCommand =
                    "SELECT ID, Image " +
                    "FROM Cancellations " +
                    "WHERE " + filterExpr;
            break;
        case "GetDataByColours":
            command =
                "SELECT ID, Image " +
                "FROM Cancellations " +
                "WHERE (Colours LIKE '%" +
                Session["value0"] + "%')";
            if (filterExpr.Equals("null"))
                AccessDataSource1.SelectCommand = command;
            else
                AccessDataSource1.SelectCommand = command +
                    " AND(" +filterExpr+ ")";
            break;
        case "GetDataByColoursNumeral":
            command =
                "SELECT ID, Image " +
                "FROM Cancellations " +
```

```
        "WHERE (Colours LIKE '%" +
        Session["value0"] + "%') " +
                "AND (Colours LIKE '%" +
        Session["value1"] + "%') " +
                        "AND ([Normalised Arabic] LIKE '%" +
        Session["value2"] + "%')";
    if (filterExpr.Equals("null"))
        AccessDataSource1.SelectCommand = command;
    else
        AccessDataSource1.SelectCommand = command +
          " AND(" + filterExpr + ")";
    break;
case "GetDataByCountry":
    command =
        "SELECT Cancellations.ID, Cancellations.Image " +
        "FROM (Cancellations INNER JOIN "+
        "[Local Names] ON Cancellations.[Post Office] = [Local Names].POTurk) "+
        "WHERE ([Local Names].Country = '" +
        Session["value0"] +
        "')";
    if (filterExpr.Equals("null"))
        AccessDataSource1.SelectCommand = command +
          " ORDER BY Cancellations.[Post Office]";
    else
        AccessDataSource1.SelectCommand = command +
          " AND(" + filterExpr + ")" +
          " ORDER BY Cancellations.[Post Office]";
    break;
case "GetDataByFrenchOttoman":
    command =
        "SELECT ID, Image " +
        "FROM Cancellations " +
        "WHERE ([French Field] LIKE '%" +
        Session["value0"] + "%') " +
        "AND ([Ottoman Field] LIKE '%" +
        Session["value1"] + "%')";
    if (filterExpr.Equals("null"))
        AccessDataSource1.SelectCommand = command;
    else
        AccessDataSource1.SelectCommand = command +
          " AND(" + filterExpr + ")";
    break;
case "GetDataByID":
    command =
        "SELECT ID,Image " +
        "FROM Cancellations " +
        "WHERE (ID = " +
        Session["value0"] + ")";
    if (filterExpr.Equals("null"))
        AccessDataSource1.SelectCommand = command;
    else
        AccessDataSource1.SelectCommand = command +
          " AND(" + filterExpr + ")";
    break;
case "GetDataByVilaet":
    command =
        "SELECT Cancellations.ID, Cancellations.Image " +
        "FROM (Cancellations INNER JOIN " +
        "[Local Names] ON Cancellations.[Post Office] = [Local Names].POTurk) " +
        "WHERE ([Local Names].Vilaet = '" +
```

```
            Session["value0"] +"')";
        if (filterExpr.Equals("null"))
            AccessDataSource1.SelectCommand = command +
              " ORDER BY Cancellations.[Post Office]";
        else
            AccessDataSource1.SelectCommand = command +
              " AND(" + filterExpr + ")" +
              " ORDER BY Cancellations.[Post Office]";
        break;
      case "GetDataNegative":
        command =
          "SELECT ID, Image " +
          "FROM Cancellations " +
          "WHERE (CW LIKE 'S%')";
        if (filterExpr.Equals("null"))
            AccessDataSource1.SelectCommand = command +
              " ORDER BY ID";
        else
            AccessDataSource1.SelectCommand = command +
              " AND(" + filterExpr + ")" +
              " ORDER BY ID";
        break;
    }

    DataView dv = (DataView)AccessDataSource1.Select(DataSourceSelectArguments.Empty);

    for (int i = 0; i < dv.Count; i++)
    {
      if( !(dv.Table.Rows[i]["Image"] is DBNull) )
      {
        string img = (string)dv.Table.Rows[i]["Image"];
        img = img.Trim();
        char[] param = { ',' };
        string[] tokens = img.Split(param,StringSplitOptions.RemoveEmptyEntries);
        for (int j = 0; j < tokens.Length; j++)
        {
          string url = tokens[j].Substring(5).Replace('\\','/');
          if (File.Exists(Server.MapPath("Images/") + url))
          {
            idArray.Add(dv.Table.Rows[i]["ID"]);
            imgArray.Add(url);
          }
        }
      }
    }

    Session["idArray"] = idArray;
    Session["imgArray"] = imgArray;
}

protected void previous_Click(object sender, EventArgs e)
{
    if (currentPage == 0)
      return;

    currentPage--;
    ViewState["currentPage"] = currentPage;

    setButtonVisibility();
    pageLabel.Text = "Page " + (currentPage + 1) + " of " + numberOfPages;
```

```
        displayImages();
    }
    protected void next_Click(object sender, EventArgs e)
    {
        if (currentPage == lastPage)
            return;

        currentPage++;
        ViewState["currentPage"] = currentPage;

        setButtonVisibility();
        pageLabel.Text = "Page " + (currentPage + 1) + " of " + numberOfPages;
        displayImages();
    }
    protected void first_Click(object sender, EventArgs e)
    {
        if (currentPage == 0)
            return;

        currentPage = 0;
        ViewState["currentPage"] = currentPage;

        setButtonVisibility();
        pageLabel.Text = "Page " + (currentPage + 1) + " of " + numberOfPages;
        displayImages();
    }
    protected void middle_Click(object sender, EventArgs e)
    {
        if (currentPage == middlePage)
            return;

        currentPage = middlePage;
        ViewState["currentPage"] = currentPage;

        setButtonVisibility();
        pageLabel.Text = "Page " + (currentPage + 1) + " of " + numberOfPages;
        displayImages();
    }
    protected void last_Click(object sender, EventArgs e)
    {
        if (currentPage == lastPage)
            return;

        currentPage = lastPage;
        ViewState["currentPage"] = currentPage;

        setButtonVisibility();
        pageLabel.Text = "Page " + (currentPage + 1) + " of " + numberOfPages;
        displayImages();
    }

    protected void LinkButton_Click(object sender, EventArgs e)
    {
        LinkButton lb = (LinkButton)sender;
        Session["detailsID"] = lb.Text;
        Response.Redirect("Details.aspx");
    }
}
```

A19) αρχείο ConstantClass.cs :

```
using System;
using System.Data;
using System.Configuration;
using System.Web;
using System.Web.Security;
using System.Web.UI;
using System.Web.UI.WebControls;
using System.Web.UI.WebControls.WebParts;
using System.Web.UI.HtmlControls;

public class ConstantClass
{
            public ConstantClass()
            {           }

   public static string strOneLine(string s)
   {
      if (s == null)
         return s;
      return s.Replace("\n","  ");
   }

   public static bool containsFrench(string s)
   {
      if (s == null)
         return false;

      char[] temp = s.ToCharArray();

      for (int i = 0; i < temp.Length; i++)
         if ( (temp[i] >= 'A' && temp[i] <= 'Z') ||
            (temp[i] >= 'a' && temp[i] <= 'z'))
            return true;

      return false;
   }
   public static string removedFrench(string s)
   {
      if(s==null)
         return null;
      string result = "";

      char[] temp = s.ToCharArray();

      for (int i = 0; i < temp.Length; i++)
      {
         if ((temp[i] >= 'A' && temp[i] <= 'Z') ||
            (temp[i] >= 'a' && temp[i] <= 'z'))
         {
            ;
         }
         else
            result = result + temp[i];
      }

      return result;
```

```
}
public static bool containsOttoman(string s)
{
   if (s == null)
      return false;

   char[] temp = s.ToCharArray();

   for (int i = 0; i < temp.Length; i++)
      if (temp[i] >= 1548 && temp[i] <= 1785)
         return true;

   return false;
}
public static string removedOttoman(string s)
{
   if (s == null)
      return null;

   string result = "";
   char[] temp = s.ToCharArray();

   for (int i = 0; i < temp.Length; i++)
   {
      if (temp[i] >= 1548 && temp[i] <= 1785)
      {    ;      }
      else
         result = result + temp[i];
   }

   return result;
}

public static string expandedFrench(string frenchText)
{
   string newValue;

   newValue = frenchText.Replace("d", "[\u0111d]");
   newValue = newValue.Replace("N", "[N\u0418\u00f1]");
   newValue = newValue.Replace("A", "[A\u00c0\u00c1\u00c2\u00c3\u00c4\u01fa]");
   newValue = newValue.Replace("E", "[E\u00c8\u00c9\u00ca\u00cb]");
   newValue = newValue.Replace("I", "[I\u00cc\u00cd\u00ce\u00cf\u0130\u0131]");
   newValue = newValue.Replace("O", "[O\u00d2\u00d3\u00d4\u00d5\u00d6]");
   newValue = newValue.Replace("U", "[U\u00d9\u00da\u00db\u00dc]");
   newValue = newValue.Replace("Y", "[Y\u00dd\u0178]");
   newValue = newValue.Replace("C", "[\u00c7\u00e7\u010d\u0107C]");
   newValue = newValue.Replace("s", "[\u015f\u0161s]");
   newValue = newValue.Replace("g", "[\u011fg]");
   newValue = newValue.Replace("z", "[\u017ez]");
   newValue = newValue.Replace("\u00c6", "AE");
   newValue = newValue.Replace("\u0152", "OE");

   return newValue;
}
public static string expandedOttoman(string ottoText)
{
   string newValue;

   newValue = ottoText.Replace("\u062d\u0627\u0646\u0647", "[\u062d\u062e]\u0627\u0646\u0647");
   newValue = newValue.Replace("\u062e\u0627\u0646\u0647", "[\u062d\u062e]\u0627\u0646\u0647");
```

```
    newValue = newValue.Replace("\u0634\u0639\u0628 ", "[\u0634\u0633]\u0639\u0628");
    newValue = newValue.Replace("\u0633\u0639\u0628", "[\u0634\u0633]\u0639\u0628");
    newValue = newValue.Replace("\u0627", "[\u0622\u0627\u0623\u0625]");

    return newValue;
}
public static string wildcarded_fr(string expandedtext)
{
    int inside = 0;
    string hexp = "";
    string wildCardedText = "";

    for (int i = 0; i < expandedtext.Length; i++)
    {
        hexp = expandedtext.Substring(i, 1);
        switch (hexp)
        {
            case "!":
            case "=":
            case " ":
            case "%":
            case "_":
                break;
            case "[":
                if (inside == 0) hexp = "[;";
                else hexp = "";
                inside++;
                break;
            case "]":
                if (inside > 1) hexp = "";
                inside--;
                break;
            default:
                if (inside == 0) hexp = "[;" + hexp + "]";
                break;
        }
        wildCardedText += hexp;
    }
    return wildCardedText;
}
public static string wildcarded_ot(string expandedText)
{
    int inside = 0;
    string hexp = "";
    string wildCardedText = "";

    for (int i = 0; i < expandedText.Length; i++)
    {
        hexp = expandedText.Substring(i, 1);
        switch (hexp)
        {
            case "!":
            case "=":
            case " ":
            case "%":
            case "_":
                break;
            case "[":
                if (inside == 0) hexp = "[\u061b";
                else hexp = "";
```

```
                inside++;
                break;
            case "]":
                if (inside > 1) hexp = "";
                inside--;
                break;
            case "0":
            case "1":
            case "2":
            case "3":
            case "4":
            case "5":
            case "6":
            case "7":
            case "8":
            case "9":
                if (inside == 0) hexp = "[\u066b" + hexp + "]";
                break;
            case "\u0660":
            case "\u0661":
            case "\u0662":
            case "\u0663":
            case "\u0664":
            case "\u0665":
            case "\u0666":
            case "\u0667":
            case "\u0668":
            case "\u0669":
                if (inside == 0)
                    hexp = "[\u066b" + hexp + "]";
                break;
            default:
                if (inside == 0)
                    hexp = "[\u061b" + hexp + "]";
                break;
        }
        wildCardedText += hexp;
    }
    return wildCardedText;
}

public static string strAllArabised(string text)
{
    if(text == null)
        return null;
    string temp = text;

    if ( !text.Equals(string.Empty)  )
    {
        temp = strNumArabised(temp);
        temp = strParArabised(temp);
        temp = strStarArabised(temp);
        temp = strQmrkArabised(temp);
        temp = strSemicolonArabised(temp);
        temp = strColonArabised(temp);
    }
    return temp;
}
public static string strNumArabised(string s)
{
```

```
    // arabic(48,49,...,57) -> hindi(1632,1633,...,1641)
    s = s.Replace('\u0030', '\u0660');
    s = s.Replace('\u0031', '\u0661');
    s = s.Replace('\u0032', '\u0662');
    s = s.Replace('\u0033', '\u0663');
    s = s.Replace('\u0034', '\u0664');
    s = s.Replace('\u0035', '\u0665');
    s = s.Replace('\u0036', '\u0666');
    s = s.Replace('\u0037', '\u0667');
    s = s.Replace('\u0038', '\u0668');
    s = s.Replace('\u0039', '\u0669');
    return s;
}
public static string strParArabised(string s)
{
    if(s==null)
        return s;
    if (s.Equals(string.Empty))
        return s;

    int i;
    char[] temp = s.ToCharArray();
    string text = "";

    for (i = 0; i < temp.Length-1; i++)
    {
        if (temp[i].Equals('\u0028'))
        {
            if (temp[i+1].Equals('\u0650'))
                text = text + temp[i];
            else
                text = text + temp[i] + '\u0650';
        }
        else if (temp[i].Equals('\u0029'))
        {
            if (temp[i + 1].Equals('\u0650'))
                text = text + temp[i];
            else
                text = text + temp[i] + '\u0650';
        }
        else
        {
            text = text + temp[i];
        }
    }
        //last character of string
    if (temp[i].Equals('\u0028') || temp[i].Equals('\u0029'))
        text = text + temp[i] + '\u0650';
    else
        text = text + temp[i];

    return text;
}
public static string strStarArabised(string s)
{
    if (s == null)
        return s;

    return s.Replace('*','\u066d');
}
```

```
    public static string strQmrkArabised(string s)
    {
        if (s == null)
            return s;

        return s.Replace('?', '\u061f');
    }
    public static string strSemicolonArabised(string s)
    {
        if (s == null)
            return s;

        return s.Replace(';', '\u061b');
    }
    public static string strColonArabised(string s)
    {
        if (s == null)
            return s;

        return s.Replace(',', '\u066b');
    }

    public static string strNumWesternised(string s)
    {
        // hindi(1632,1633,...,1641) -> arabic(48,49,...,57)
        s = s.Replace('\u0660', '\u0030');
        s = s.Replace('\u0661', '\u0031');
        s = s.Replace('\u0662', '\u0032');
        s = s.Replace('\u0663', '\u0033');
        s = s.Replace('\u0664', '\u0034');
        s = s.Replace('\u0665', '\u0035');
        s = s.Replace('\u0666', '\u0036');
        s = s.Replace('\u0667', '\u0037');
        s = s.Replace('\u0668', '\u0038');
        s = s.Replace('\u0669', '\u0039');
        return s;
    }
    public static string strStarQmrkWesternised(string s)
    {
        if (s == null)
            return null;
        string temp = s.Replace('\u066d', '*');
        return temp.Replace('\u061f','?');
    }

    public static void testtt()
    {
    }
}
```

A20) αρχείο DataSet1.xsd :

```
<?xml version="1.0" encoding="utf-8"?>
<xs:schema id="DataSet1" targetNamespace="http://tempuri.org/DataSet1.xsd"
xmlns:mstns="http://tempuri.org/DataSet1.xsd" xmlns="http://tempuri.org/DataSet1.xsd"
xmlns:xs="http://www.w3.org/2001/XMLSchema" xmlns:msdata="urn:schemas-microsoft-com:xml-msdata"
xmlns:msprop="urn:schemas-microsoft-com:xml-msprop" attributeFormDefault="qualified"
elementFormDefault="qualified">
```

```
  <xs:annotation>
    <xs:appinfo source="urn:schemas-microsoft-com:xml-msdatasource">
      <DataSource DefaultConnectionIndex="0" FunctionsComponentName="QueriesTableAdapter"
GeneratorFunctionsComponentClassName="QueriesTableAdapter" Modifier="AutoLayout, AnsiClass, Class,
Public" SchemaSerializationMode="IncludeSchema" UserFunctionsComponentName="QueriesTableAdapter"
xmlns="urn:schemas-microsoft-com:xml-msdatasource">
        <Connections>
          <Connection AppSettingsObjectName="Web.config"
AppSettingsPropertyName="Cancel_Alexandros_Apr09ConnectionString" ConnectionStringObject=""
IsAppSettingsProperty="True" Modifier="Assembly" Name="Cancel_Alexandros_Apr09ConnectionString
(Web.config)" PropertyReference="AppConfig.System.Configuration.ConfigurationManager.
0.ConnectionStrings.Cancel_Alexandros_Apr09ConnectionString.ConnectionString"
Provider="System.Data.OleDb">
          </Connection>
        </Connections>
        <Tables>
          <TableAdapter BaseClass="System.ComponentModel.Component"
DataAccessorModifier="AutoLayout, AnsiClass, Class, Public"
DataAccessorName="CancellationsTableAdapter"
GeneratorDataComponentClassName="CancellationsTableAdapter" Name="Cancellations"
UserDataComponentName="CancellationsTableAdapter">
            <MainSource>
              <DbSource ConnectionRef="Cancel_Alexandros_Apr09ConnectionString (Web.config)"
DbObjectName="Cancellations" DbObjectType="Table" FillMethodModifier="Public"
FillMethodName="FillByPostOffice" GenerateMethods="Both" GenerateShortCommands="False"
GeneratorGetMethodName="GetDataByPostOffice" GeneratorSourceName="FillByPostOffice"
GetMethodModifier="Public" GetMethodName="GetDataByPostOffice" QueryType="Rowset"
ScalarCallRetval="System.Object, mscorlib, Version=2.0.0.0, Culture=neutral,
PublicKeyToken=b77a5c561934e089" UseOptimisticConcurrency="False"
UserGetMethodName="GetDataByPostOffice" UserSourceName="FillByPostOffice">
                <DeleteCommand>
                  <DbCommand CommandType="Text" ModifiedByUser="False">
                    <CommandText>DELETE FROM `Cancellations` WHERE ((`ID` = ?))</CommandText>
                    <Parameters>
                      <Parameter AllowDbNull="False" AutogeneratedName="" DataSourceName=""
DbType="Int32" Direction="Input" ParameterName="Original_ID" Precision="0" ProviderType="Integer"
Scale="0" Size="0" SourceColumn="ID" SourceColumnNullMapping="False" SourceVersion="Original">
                      </Parameter>
                    </Parameters>
                  </DbCommand>
                </DeleteCommand>
                <SelectCommand>
                  <DbCommand CommandType="Text" ModifiedByUser="False">
                    <CommandText>SELECT     ID, [Normalised Latin], [Normalised Arabic], [Post Office], CW, Br
FROM       Cancellations
WHERE     ([Post Office] = ?)</CommandText>
                    <Parameters>
                      <Parameter AllowDbNull="True" AutogeneratedName="Param1" ColumnName="Post Office"
DataSourceName="Cancellations" DataTypeServer="DbType.StringFixedLength(90)" DbType="String"
Direction="Input" ParameterName="Post_Office" Precision="0" ProviderType="WChar" Scale="0"
Size="90" SourceColumn="Post Office" SourceColumnNullMapping="False" SourceVersion="Current">
                      </Parameter>
                    </Parameters>
                  </DbCommand>
                </SelectCommand>
                <UpdateCommand>
                  <DbCommand CommandType="Text" ModifiedByUser="False">
                    <CommandText>UPDATE `Cancellations` SET `Normalised Latin` = ?, `Normalised Arabic` = ?,
`Post Office` = ?, `CW` = ?, `Br` = ? WHERE ((`ID` = ?))</CommandText>
                    <Parameters>
```

```
                <Parameter AllowDbNull="True" AutogeneratedName="" DataSourceName=""
DbType="String" Direction="Input" ParameterName="Normalised_Latin" Precision="0"
ProviderType="VarWChar" Scale="0" Size="0" SourceColumn="Normalised Latin"
SourceColumnNullMapping="False" SourceVersion="Current">
                </Parameter>
                <Parameter AllowDbNull="True" AutogeneratedName="" DataSourceName=""
DbType="String" Direction="Input" ParameterName="Normalised_Arabic" Precision="0"
ProviderType="VarWChar" Scale="0" Size="0" SourceColumn="Normalised Arabic"
SourceColumnNullMapping="False" SourceVersion="Current">
                </Parameter>
                <Parameter AllowDbNull="True" AutogeneratedName="" DataSourceName=""
DbType="String" Direction="Input" ParameterName="Post_Office" Precision="0"
ProviderType="VarWChar" Scale="0" Size="0" SourceColumn="Post Office"
SourceColumnNullMapping="False" SourceVersion="Current">
                </Parameter>
                <Parameter AllowDbNull="True" AutogeneratedName="" DataSourceName=""
DbType="String" Direction="Input" ParameterName="CW" Precision="0" ProviderType="VarWChar"
Scale="0" Size="0" SourceColumn="CW" SourceColumnNullMapping="False" SourceVersion="Current">
                </Parameter>
                <Parameter AllowDbNull="True" AutogeneratedName="" DataSourceName=""
DbType="String" Direction="Input" ParameterName="Br" Precision="0" ProviderType="VarWChar"
Scale="0" Size="0" SourceColumn="Br" SourceColumnNullMapping="False" SourceVersion="Current">
                </Parameter>
                <Parameter AllowDbNull="False" AutogeneratedName="" DataSourceName=""
DbType="Int32" Direction="Input" ParameterName="Original_ID" Precision="0" ProviderType="Integer"
Scale="0" Size="0" SourceColumn="ID" SourceColumnNullMapping="False" SourceVersion="Original">
                </Parameter>
              </Parameters>
            </DbCommand>
          </UpdateCommand>
        </DbSource>
      </MainSource>
      <Mappings>
        <Mapping SourceColumn="ID" DataSetColumn="ID" />
        <Mapping SourceColumn="Normalised Latin" DataSetColumn="Normalised Latin" />
        <Mapping SourceColumn="Normalised Arabic" DataSetColumn="Normalised Arabic" />
        <Mapping SourceColumn="Post Office" DataSetColumn="Post Office" />
        <Mapping SourceColumn="CW" DataSetColumn="CW" />
        <Mapping SourceColumn="Br" DataSetColumn="Br" />
      </Mappings>
      <Sources>
        <DbSource ConnectionRef="Cancel_Alexandros_Apr09ConnectionString (Web.config)"
DbObjectName="Cancellations" DbObjectType="Table" FillMethodModifier="Public"
FillMethodName="Fill" GenerateMethods="Both" GenerateShortCommands="True"
GeneratorGetMethodName="GetData" GeneratorSourceName="Fill" GetMethodModifier="Public"
GetMethodName="GetData" QueryType="Rowset" ScalarCallRetval="System.Object, mscorlib,
Version=2.0.0.0, Culture=neutral, PublicKeyToken=b77a5c561934e089" UseOptimisticConcurrency="True"
UserGetMethodName="GetData" UserSourceName="Fill">
          <SelectCommand>
            <DbCommand CommandType="Text" ModifiedByUser="True">
              <CommandText>SELECT Br, CW, ID, [Normalised Arabic], [Normalised Latin], [Post Office]
FROM Cancellations</CommandText>
              <Parameters>
              </Parameters>
            </DbCommand>
          </SelectCommand>
        </DbSource>
        <DbSource ConnectionRef="Cancel_Alexandros_Apr09ConnectionString (Web.config)"
DbObjectName="Cancellations" DbObjectType="Table" FillMethodModifier="Public"
FillMethodName="FillByColours" GenerateMethods="Both" GenerateShortCommands="True"
```

```
GeneratorGetMethodName="GetDataByColours" GeneratorSourceName="FillByColours"
GetMethodModifier="Public" GetMethodName="GetDataByColours" QueryType="Rowset"
ScalarCallRetval="System.Object, mscorlib, Version=2.0.0.0, Culture=neutral,
PublicKeyToken=b77a5c561934e089" UseOptimisticConcurrency="True"
UserGetMethodName="GetDataByColours" UserSourceName="FillByColours">
            <SelectCommand>
              <DbCommand CommandType="Text" ModifiedByUser="True">
                <CommandText>SELECT Br, CW, ID, [Normalised Arabic], [Normalised Latin], [Post Office]
FROM Cancellations WHERE (Colours LIKE '%' + ? + '%')</CommandText>
                <Parameters>
                  <Parameter AllowDbNull="True" AutogeneratedName="Param1" ColumnName="Colours"
DataSourceName="Cancellations" DataTypeServer="DbType.StringFixedLength(250)" DbType="String"
Direction="Input" ParameterName="Colours" Precision="0" ProviderType="WChar" Scale="0" Size="250"
SourceColumn="Colours" SourceColumnNullMapping="False" SourceVersion="Current">
                  </Parameter>
                </Parameters>
              </DbCommand>
            </SelectCommand>
          </DbSource>
          <DbSource ConnectionRef="Cancel_Alexandros_Apr09ConnectionString (Web.config)"
DbObjectName="Cancellations" DbObjectType="Table" FillMethodModifier="Public"
FillMethodName="FillByColoursNumeral" GenerateMethods="Both" GenerateShortCommands="True"
GeneratorGetMethodName="GetDataByColoursNumeral" GeneratorSourceName="FillByColoursNumeral"
GetMethodModifier="Public" GetMethodName="GetDataByColoursNumeral" QueryType="Rowset"
ScalarCallRetval="System.Object, mscorlib, Version=2.0.0.0, Culture=neutral,
PublicKeyToken=b77a5c561934e089" UseOptimisticConcurrency="True"
UserGetMethodName="GetDataByColoursNumeral" UserSourceName="FillByColoursNumeral">
            <SelectCommand>
              <DbCommand CommandType="Text" ModifiedByUser="True">
                <CommandText>SELECT Br, CW, ID, [Normalised Arabic], [Normalised Latin], [Post Office]
FROM Cancellations WHERE (Colours LIKE '%' + ? + '%') AND (Colours LIKE '%' + ? + '%') AND
([Normalised Arabic] LIKE '%' + ? + '%')</CommandText>
                <Parameters>
                  <Parameter AllowDbNull="True" AutogeneratedName="Param1" ColumnName="Colours"
DataSourceName="Cancellations" DataTypeServer="DbType.StringFixedLength(250)" DbType="String"
Direction="Input" ParameterName="Shape" Precision="0" ProviderType="WChar" Scale="0" Size="250"
SourceColumn="Colours" SourceColumnNullMapping="False" SourceVersion="Current">
                  </Parameter>
                  <Parameter AllowDbNull="True" AutogeneratedName="Param2" ColumnName="Colours"
DataSourceName="Cancellations" DataTypeServer="DbType.StringFixedLength(250)" DbType="String"
Direction="Input" ParameterName="Expression" Precision="0" ProviderType="WChar" Scale="0"
Size="250" SourceColumn="Colours" SourceColumnNullMapping="False" SourceVersion="Current">
                  </Parameter>
                  <Parameter AllowDbNull="True" AutogeneratedName="Param3" ColumnName="Normalised
Arabic" DataSourceName="Cancellations" DataTypeServer="DbType.StringFixedLength(250)"
DbType="String" Direction="Input" ParameterName="Numeral" Precision="0" ProviderType="WChar"
Scale="0" Size="250" SourceColumn="Normalised Arabic" SourceColumnNullMapping="False"
SourceVersion="Current">
                  </Parameter>
                </Parameters>
              </DbCommand>
            </SelectCommand>
          </DbSource>
          <DbSource ConnectionRef="Cancel_Alexandros_Apr09ConnectionString (Web.config)"
DbObjectName="Cancellations" DbObjectType="Table" FillMethodModifier="Public"
FillMethodName="FillByCountry" GenerateMethods="Both" GenerateShortCommands="True"
GeneratorGetMethodName="GetDataByCountry" GeneratorSourceName="FillByCountry"
GetMethodModifier="Public" GetMethodName="GetDataByCountry" QueryType="Rowset"
ScalarCallRetval="System.Object, mscorlib, Version=2.0.0.0, Culture=neutral,
```

```
PublicKeyToken=b77a5c561934e089" UseOptimisticConcurrency="True"
UserGetMethodName="GetDataByCountry" UserSourceName="FillByCountry">
            <SelectCommand>
              <DbCommand CommandType="Text" ModifiedByUser="True">
                <CommandText>SELECT     Cancellations.Br, Cancellations.CW, Cancellations.ID,
Cancellations.[Normalised Arabic], Cancellations.[Normalised Latin],
                  Cancellations.[Post Office]
FROM        (Cancellations INNER JOIN
                  [Local Names] ON Cancellations.[Post Office] = [Local Names].POTurk)
WHERE     ([Local Names].Country = ?)
ORDER BY Cancellations.[Post Office]</CommandText>
                <Parameters>
                  <Parameter AllowDbNull="True" AutogeneratedName="Param1" ColumnName="Country"
DataSourceName="Local Names" DataTypeServer="DbType.StringFixedLength(50)" DbType="String"
Direction="Input" ParameterName="Country" Precision="0" ProviderType="WChar" Scale="0" Size="50"
SourceColumn="Country" SourceColumnNullMapping="False" SourceVersion="Current">
                  </Parameter>
                </Parameters>
              </DbCommand>
            </SelectCommand>
          </DbSource>
          <DbSource ConnectionRef="Cancel_Alexandros_Apr09ConnectionString (Web.config)"
DbObjectName="Cancellations" DbObjectType="Table" FillMethodModifier="Public"
FillMethodName="FillByFrenchOttoman" GenerateMethods="Both" GenerateShortCommands="True"
GeneratorGetMethodName="GetDataByFrenchOttoman" GeneratorSourceName="FillByFrenchOttoman"
GetMethodModifier="Public" GetMethodName="GetDataByFrenchOttoman" QueryType="Rowset"
ScalarCallRetval="System.Object, mscorlib, Version=2.0.0.0, Culture=neutral,
PublicKeyToken=b77a5c561934e089" UseOptimisticConcurrency="True"
UserGetMethodName="GetDataByFrenchOttoman" UserSourceName="FillByFrenchOttoman">
            <SelectCommand>
              <DbCommand CommandType="Text" ModifiedByUser="True">
                <CommandText>SELECT Br, CW, ID, [Normalised Arabic], [Normalised Latin], [Post Office]
FROM Cancellations WHERE ([French Field] LIKE '%' + ? + '%') AND ([Ottoman Field] LIKE '%' + ? +
'%')</CommandText>
                <Parameters>
                  <Parameter AllowDbNull="True" AutogeneratedName="Param1" ColumnName="French Field"
DataSourceName="Cancellations" DataTypeServer="DbType.StringFixedLength(250)" DbType="String"
Direction="Input" ParameterName="French_Field" Precision="0" ProviderType="WChar" Scale="0"
Size="250" SourceColumn="French Field" SourceColumnNullMapping="False" SourceVersion="Current">
                  </Parameter>
                  <Parameter AllowDbNull="True" AutogeneratedName="Param2" ColumnName="Ottoman
Field" DataSourceName="Cancellations" DataTypeServer="DbType.StringFixedLength(250)"
DbType="String" Direction="Input" ParameterName="Ottoman_Field" Precision="0"
ProviderType="WChar" Scale="0" Size="250" SourceColumn="Ottoman Field"
SourceColumnNullMapping="False" SourceVersion="Current">
                  </Parameter>
                </Parameters>
              </DbCommand>
            </SelectCommand>
          </DbSource>
          <DbSource ConnectionRef="Cancel_Alexandros_Apr09ConnectionString (Web.config)"
DbObjectName="Cancellations" DbObjectType="Table" FillMethodModifier="Public"
FillMethodName="FillByID" GenerateMethods="Both" GenerateShortCommands="True"
GeneratorGetMethodName="GetDataByID" GeneratorSourceName="FillByID"
GetMethodModifier="Public" GetMethodName="GetDataByID" QueryType="Rowset"
ScalarCallRetval="System.Object, mscorlib, Version=2.0.0.0, Culture=neutral,
PublicKeyToken=b77a5c561934e089" UseOptimisticConcurrency="True"
UserGetMethodName="GetDataByID" UserSourceName="FillByID">
            <SelectCommand>
              <DbCommand CommandType="Text" ModifiedByUser="True">
```

<CommandText>SELECT Br, CW, ID, [Normalised Arabic], [Normalised Latin], [Post Office] FROM Cancellations WHERE (ID = ?)</CommandText>
<Parameters>
<Parameter AllowDbNull="False" AutogeneratedName="Param1" ColumnName="ID" DataSourceName="Cancellations" DataTypeServer="DbType.Int32" DbType="Int32" Direction="Input" ParameterName="ID" Precision="0" ProviderType="Integer" Scale="0" Size="0" SourceColumn="ID" SourceColumnNullMapping="False" SourceVersion="Current">
</Parameter>
</Parameters>
</DbCommand>
</SelectCommand>
</DbSource>
<DbSource ConnectionRef="Cancel_Alexandros_Apr09ConnectionString (Web.config)" DbObjectName="Cancellations" DbObjectType="Table" FillMethodModifier="Public" FillMethodName="FillByVilaet" GenerateMethods="Both" GenerateShortCommands="True" GeneratorGetMethodName="GetDataByVilaet" GeneratorSourceName="FillByVilaet" GetMethodModifier="Public" GetMethodName="GetDataByVilaet" QueryType="Rowset" ScalarCallRetval="System.Object, mscorlib, Version=2.0.0.0, Culture=neutral, PublicKeyToken=b77a5c561934e089" UseOptimisticConcurrency="True" UserGetMethodName="GetDataByVilaet" UserSourceName="FillByVilaet">
<SelectCommand>
<DbCommand CommandType="Text" ModifiedByUser="True">
<CommandText>SELECT Cancellations.Br, Cancellations.CW, Cancellations.ID, Cancellations.[Normalised Arabic], Cancellations.[Normalised Latin], Cancellations.[Post Office] FROM (Cancellations INNER JOIN [Local Names] ON Cancellations.[Post Office] = [Local Names].POTurk) WHERE ([Local Names].Vilaet = ?) ORDER BY Cancellations.[Post Office]</CommandText>
<Parameters>
<Parameter AllowDbNull="True" AutogeneratedName="Param1" ColumnName="Vilaet" DataSourceName="Local Names" DataTypeServer="DbType.StringFixedLength(50)" DbType="String" Direction="Input" ParameterName="Vilaet" Precision="0" ProviderType="WChar" Scale="0" Size="50" SourceColumn="Vilaet" SourceColumnNullMapping="False" SourceVersion="Current">
</Parameter>
</Parameters>
</DbCommand>
</SelectCommand>
</DbSource>
<DbSource ConnectionRef="Cancel_Alexandros_Apr09ConnectionString (Web.config)" DbObjectName="Cancellations" DbObjectType="Table" FillMethodModifier="Public" FillMethodName="FillNegative" GenerateMethods="Both" GenerateShortCommands="True" GeneratorGetMethodName="GetDataNegative" GeneratorSourceName="FillNegative" GetMethodModifier="Public" GetMethodName="GetDataNegative" QueryType="Rowset" ScalarCallRetval="System.Object, mscorlib, Version=2.0.0.0, Culture=neutral, PublicKeyToken=b77a5c561934e089" UseOptimisticConcurrency="True" UserGetMethodName="GetDataNegative" UserSourceName="FillNegative">
<SelectCommand>
<DbCommand CommandType="Text" ModifiedByUser="True">
<CommandText>SELECT Br, CW, ID, [Normalised Arabic], [Normalised Latin], [Post Office] FROM Cancellations WHERE (CW LIKE 'S%') ORDER BY ID</CommandText>
<Parameters>
</Parameters>
</DbCommand>
</SelectCommand>
</DbSource>
</Sources>
</TableAdapter>
<TableAdapter BaseClass="System.ComponentModel.Component" DataAccessorModifier="AutoLayout, AnsiClass, Class, Public" DataAccessorName="DetailsTableAdapter" GeneratorDataComponentClassName="DetailsTableAdapter" Name="Details" UserDataComponentName="DetailsTableAdapter">
<MainSource>

```
        <DbSource ConnectionRef="Cancel_Alexandros_Apr09ConnectionString (Web.config)"
DbObjectType="Unknown" FillMethodModifier="Public" FillMethodName="FillByID"
GenerateMethods="Both" GenerateShortCommands="False" GeneratorGetMethodName="GetDataByID"
GeneratorSourceName="FillByID" GetMethodModifier="Public" GetMethodName="GetDataByID"
QueryType="Rowset" ScalarCallRetval="System.Object, mscorlib, Version=2.0.0.0, Culture=neutral,
PublicKeyToken=b77a5c561934e089" UseOptimisticConcurrency="False"
UserGetMethodName="GetDataByID" UserSourceName="FillByID">
          <SelectCommand>
            <DbCommand CommandType="Text" ModifiedByUser="True">
              <CommandText>SELECT     Cancellations.ID, Cancellations.[French Field], Cancellations.FDef,
Cancellations.[Normalised Latin], Cancellations.[Ottoman Field], Cancellations.ODef,
                Cancellations.[Normalised Arabic], Cancellations.[Post Office], Cancellations.CW,
Cancellations.CWPage, Cancellations.Br, Cancellations.BrandtPage,
                Cancellations.Nuhoglou, Cancellations.Galinos, Cancellations.Dates, Cancellations.Colours,
Cancellations.[Image], Cancellations.Comments,
                [Local Names].POTurk, [Local Names].Vilaet, [Local Names].Country, [Local Names].POOtto,
[Local Names].ModLatin,
                [Local Names].ModMulti
FROM        (Cancellations INNER JOIN
                [Local Names] ON Cancellations.[Post Office] = [Local Names].POTurk)
WHERE     (Cancellations.ID = ?)</CommandText>
              <Parameters>
                <Parameter AllowDbNull="False" AutogeneratedName="Param1" ColumnName="ID"
DataSourceName="Cancellations" DataTypeServer="DbType.Int32" DbType="Int32" Direction="Input"
ParameterName="ID" Precision="0" ProviderType="Integer" Scale="0" Size="0" SourceColumn="ID"
SourceColumnNullMapping="False" SourceVersion="Current">
                </Parameter>
              </Parameters>
            </DbCommand>
          </SelectCommand>
        </DbSource>
      </MainSource>
      <Mappings>
        <Mapping SourceColumn="ID" DataSetColumn="ID" />
        <Mapping SourceColumn="French Field" DataSetColumn="French Field" />
        <Mapping SourceColumn="FDef" DataSetColumn="FDef" />
        <Mapping SourceColumn="Normalised Latin" DataSetColumn="Normalised Latin" />
        <Mapping SourceColumn="Ottoman Field" DataSetColumn="Ottoman Field" />
        <Mapping SourceColumn="ODef" DataSetColumn="ODef" />
        <Mapping SourceColumn="Normalised Arabic" DataSetColumn="Normalised Arabic" />
        <Mapping SourceColumn="Post Office" DataSetColumn="Post Office" />
        <Mapping SourceColumn="CW" DataSetColumn="CW" />
        <Mapping SourceColumn="CWPage" DataSetColumn="CWPage" />
        <Mapping SourceColumn="Br" DataSetColumn="Br" />
        <Mapping SourceColumn="BrandtPage" DataSetColumn="BrandtPage" />
        <Mapping SourceColumn="Nuhoglou" DataSetColumn="Nuhoglou" />
        <Mapping SourceColumn="Galinos" DataSetColumn="Galinos" />
        <Mapping SourceColumn="Dates" DataSetColumn="Dates" />
        <Mapping SourceColumn="Colours" DataSetColumn="Colours" />
        <Mapping SourceColumn="Image" DataSetColumn="Image" />
        <Mapping SourceColumn="Comments" DataSetColumn="Comments" />
        <Mapping SourceColumn="POTurk" DataSetColumn="POTurk" />
        <Mapping SourceColumn="Vilaet" DataSetColumn="Vilaet" />
        <Mapping SourceColumn="Country" DataSetColumn="Country" />
        <Mapping SourceColumn="POOtto" DataSetColumn="POOtto" />
        <Mapping SourceColumn="ModLatin" DataSetColumn="ModLatin" />
        <Mapping SourceColumn="ModMulti" DataSetColumn="ModMulti" />
      </Mappings>
      <Sources>
      </Sources>
```

```
        </TableAdapter>
      </Tables>
      <Sources>
      </Sources>
    </DataSource>
  </xs:appinfo>
 </xs:annotation>
 <xs:element name="DataSet1" msdata:IsDataSet="true" msdata:UseCurrentLocale="true"
msprop:Generator_UserDSName="DataSet1" msprop:Generator_DataSetName="DataSet1">
  <xs:complexType>
   <xs:choice minOccurs="0" maxOccurs="unbounded">
    <xs:element name="Cancellations" msprop:Generator_UserTableName="Cancellations"
msprop:Generator_RowDeletedName="CancellationsRowDeleted"
msprop:Generator_RowChangedName="CancellationsRowChanged"
msprop:Generator_RowClassName="CancellationsRow"
msprop:Generator_RowChangingName="CancellationsRowChanging"
msprop:Generator_RowEvArgName="CancellationsRowChangeEvent"
msprop:Generator_RowEvHandlerName="CancellationsRowChangeEventHandler"
msprop:Generator_TableClassName="CancellationsDataTable"
msprop:Generator_TableVarName="tableCancellations"
msprop:Generator_RowDeletingName="CancellationsRowDeleting"
msprop:Generator_TablePropName="Cancellations">
     <xs:complexType>
      <xs:sequence>
       <xs:element name="ID" msdata:AutoIncrement="true" msprop:Generator_UserColumnName="ID"
msprop:Generator_ColumnPropNameInRow="ID" msprop:Generator_ColumnVarNameInTable="columnID"
msprop:Generator_ColumnPropNameInTable="IDColumn" type="xs:int" />
       <xs:element name="Normalised_x0020_Latin" msprop:Generator_UserColumnName="Normalised
Latin" msprop:Generator_ColumnPropNameInRow="Normalised_Latin"
msprop:Generator_ColumnVarNameInTable="columnNormalised_Latin"
msprop:Generator_ColumnPropNameInTable="Normalised_LatinColumn" minOccurs="0">
        <xs:simpleType>
         <xs:restriction base="xs:string">
          <xs:maxLength value="250" />
         </xs:restriction>
        </xs:simpleType>
       </xs:element>
       <xs:element name="Normalised_x0020_Arabic" msprop:Generator_UserColumnName="Normalised
Arabic" msprop:Generator_ColumnPropNameInRow="Normalised_Arabic"
msprop:Generator_ColumnVarNameInTable="columnNormalised_Arabic"
msprop:Generator_ColumnPropNameInTable="Normalised_ArabicColumn" minOccurs="0">
        <xs:simpleType>
         <xs:restriction base="xs:string">
          <xs:maxLength value="250" />
         </xs:restriction>
        </xs:simpleType>
       </xs:element>
       <xs:element name="Post_x0020_Office" msprop:Generator_UserColumnName="Post Office"
msprop:Generator_ColumnPropNameInRow="Post_Office"
msprop:Generator_ColumnVarNameInTable="columnPost_Office"
msprop:Generator_ColumnPropNameInTable="Post_OfficeColumn" minOccurs="0">
        <xs:simpleType>
         <xs:restriction base="xs:string">
          <xs:maxLength value="90" />
         </xs:restriction>
        </xs:simpleType>
       </xs:element>
       <xs:element name="CW" msprop:Generator_UserColumnName="CW"
msprop:Generator_ColumnPropNameInRow="CW"
```

```
msprop:Generator_ColumnVarNameInTable="columnCW"
msprop:Generator_ColumnPropNameInTable="CWColumn" minOccurs="0">
            <xs:simpleType>
              <xs:restriction base="xs:string">
                <xs:maxLength value="50" />
              </xs:restriction>
            </xs:simpleType>
          </xs:element>
          <xs:element name="Br" msprop:Generator_UserColumnName="Br"
msprop:Generator_ColumnPropNameInRow="Br" msprop:Generator_ColumnVarNameInTable="columnBr"
msprop:Generator_ColumnPropNameInTable="BrColumn" minOccurs="0">
            <xs:simpleType>
              <xs:restriction base="xs:string">
                <xs:maxLength value="50" />
              </xs:restriction>
            </xs:simpleType>
          </xs:element>
        </xs:sequence>
      </xs:complexType>
    </xs:element>
    <xs:element name="Details" msprop:Generator_UserTableName="Details"
msprop:Generator_RowDeletedName="DetailsRowDeleted"
msprop:Generator_RowChangedName="DetailsRowChanged"
msprop:Generator_RowClassName="DetailsRow"
msprop:Generator_RowChangingName="DetailsRowChanging"
msprop:Generator_RowEvArgName="DetailsRowChangeEvent"
msprop:Generator_RowEvHandlerName="DetailsRowChangeEventHandler"
msprop:Generator_TableClassName="DetailsDataTable" msprop:Generator_TableVarName="tableDetails"
msprop:Generator_RowDeletingName="DetailsRowDeleting" msprop:Generator_TablePropName="Details">
      <xs:complexType>
        <xs:sequence>
          <xs:element name="ID" msdata:AutoIncrement="true" msprop:Generator_UserColumnName="ID"
msprop:Generator_ColumnPropNameInRow="ID" msprop:Generator_ColumnVarNameInTable="columnID"
msprop:Generator_ColumnPropNameInTable="IDColumn" type="xs:int" minOccurs="0" />
          <xs:element name="French_x0020_Field" msprop:Generator_UserColumnName="French Field"
msprop:Generator_ColumnPropNameInRow="French_Field"
msprop:Generator_ColumnVarNameInTable="columnFrench_Field"
msprop:Generator_ColumnPropNameInTable="French_FieldColumn" minOccurs="0">
            <xs:simpleType>
              <xs:restriction base="xs:string">
                <xs:maxLength value="250" />
              </xs:restriction>
            </xs:simpleType>
          </xs:element>
          <xs:element name="FDef" msprop:Generator_UserColumnName="FDef"
msprop:Generator_ColumnPropNameInRow="FDef"
msprop:Generator_ColumnVarNameInTable="columnFDef"
msprop:Generator_ColumnPropNameInTable="FDefColumn" type="xs:boolean" minOccurs="0" />
          <xs:element name="Normalised_x0020_Latin" msprop:Generator_UserColumnName="Normalised
Latin" msprop:Generator_ColumnPropNameInRow="Normalised_Latin"
msprop:Generator_ColumnVarNameInTable="columnNormalised_Latin"
msprop:Generator_ColumnPropNameInTable="Normalised_LatinColumn" minOccurs="0">
            <xs:simpleType>
              <xs:restriction base="xs:string">
                <xs:maxLength value="250" />
              </xs:restriction>
            </xs:simpleType>
          </xs:element>
          <xs:element name="Ottoman_x0020_Field" msprop:Generator_UserColumnName="Ottoman Field"
msprop:Generator_ColumnPropNameInRow="Ottoman_Field"
```

```
msprop:Generator_ColumnVarNameInTable="columnOttoman_Field"
msprop:Generator_ColumnPropNameInTable="Ottoman_FieldColumn" minOccurs="0">
            <xs:simpleType>
              <xs:restriction base="xs:string">
                <xs:maxLength value="250" />
              </xs:restriction>
            </xs:simpleType>
          </xs:element>
          <xs:element name="ODef" msprop:Generator_UserColumnName="ODef"
msprop:Generator_ColumnPropNameInRow="ODef"
msprop:Generator_ColumnVarNameInTable="columnODef"
msprop:Generator_ColumnPropNameInTable="ODefColumn" type="xs:boolean" minOccurs="0" />
          <xs:element name="Normalised_x0020_Arabic" msprop:Generator_UserColumnName="Normalised
Arabic" msprop:Generator_ColumnPropNameInRow="Normalised_Arabic"
msprop:Generator_ColumnVarNameInTable="columnNormalised_Arabic"
msprop:Generator_ColumnPropNameInTable="Normalised_ArabicColumn" minOccurs="0">
            <xs:simpleType>
              <xs:restriction base="xs:string">
                <xs:maxLength value="250" />
              </xs:restriction>
            </xs:simpleType>
          </xs:element>
          <xs:element name="Post_x0020_Office" msprop:Generator_UserColumnName="Post Office"
msprop:Generator_ColumnPropNameInRow="Post_Office"
msprop:Generator_ColumnVarNameInTable="columnPost_Office"
msprop:Generator_ColumnPropNameInTable="Post_OfficeColumn" minOccurs="0">
            <xs:simpleType>
              <xs:restriction base="xs:string">
                <xs:maxLength value="90" />
              </xs:restriction>
            </xs:simpleType>
          </xs:element>
          <xs:element name="CW" msprop:Generator_UserColumnName="CW"
msprop:Generator_ColumnPropNameInRow="CW"
msprop:Generator_ColumnVarNameInTable="columnCW"
msprop:Generator_ColumnPropNameInTable="CWColumn" minOccurs="0">
            <xs:simpleType>
              <xs:restriction base="xs:string">
                <xs:maxLength value="50" />
              </xs:restriction>
            </xs:simpleType>
          </xs:element>
          <xs:element name="CWPage" msprop:Generator_UserColumnName="CWPage"
msprop:Generator_ColumnPropNameInRow="CWPage"
msprop:Generator_ColumnVarNameInTable="columnCWPage"
msprop:Generator_ColumnPropNameInTable="CWPageColumn" minOccurs="0">
            <xs:simpleType>
              <xs:restriction base="xs:string">
                <xs:maxLength value="50" />
              </xs:restriction>
            </xs:simpleType>
          </xs:element>
          <xs:element name="Br" msprop:Generator_UserColumnName="Br"
msprop:Generator_ColumnPropNameInRow="Br" msprop:Generator_ColumnVarNameInTable="columnBr"
msprop:Generator_ColumnPropNameInTable="BrColumn" minOccurs="0">
            <xs:simpleType>
              <xs:restriction base="xs:string">
                <xs:maxLength value="50" />
              </xs:restriction>
            </xs:simpleType>
```

```
          </xs:element>
          <xs:element name="BrandtPage" msprop:Generator_UserColumnName="BrandtPage"
msprop:Generator_ColumnPropNameInRow="BrandtPage"
msprop:Generator_ColumnVarNameInTable="columnBrandtPage"
msprop:Generator_ColumnPropNameInTable="BrandtPageColumn" minOccurs="0">
            <xs:simpleType>
              <xs:restriction base="xs:string">
                <xs:maxLength value="50" />
              </xs:restriction>
            </xs:simpleType>
          </xs:element>
          <xs:element name="Nuhoglou" msprop:Generator_UserColumnName="Nuhoglou"
msprop:Generator_ColumnPropNameInRow="Nuhoglou"
msprop:Generator_ColumnVarNameInTable="columnNuhoglou"
msprop:Generator_ColumnPropNameInTable="NuhoglouColumn" minOccurs="0">
            <xs:simpleType>
              <xs:restriction base="xs:string">
                <xs:maxLength value="50" />
              </xs:restriction>
            </xs:simpleType>
          </xs:element>
          <xs:element name="Galinos" msprop:Generator_UserColumnName="Galinos"
msprop:Generator_ColumnPropNameInRow="Galinos"
msprop:Generator_ColumnVarNameInTable="columnGalinos"
msprop:Generator_ColumnPropNameInTable="GalinosColumn" minOccurs="0">
            <xs:simpleType>
              <xs:restriction base="xs:string">
                <xs:maxLength value="50" />
              </xs:restriction>
            </xs:simpleType>
          </xs:element>
          <xs:element name="Dates" msprop:Generator_UserColumnName="Dates"
msprop:Generator_ColumnPropNameInRow="Dates"
msprop:Generator_ColumnVarNameInTable="columnDates"
msprop:Generator_ColumnPropNameInTable="DatesColumn" minOccurs="0">
            <xs:simpleType>
              <xs:restriction base="xs:string">
                <xs:maxLength value="50" />
              </xs:restriction>
            </xs:simpleType>
          </xs:element>
          <xs:element name="Colours" msprop:Generator_UserColumnName="Colours"
msprop:Generator_ColumnPropNameInRow="Colours"
msprop:Generator_ColumnVarNameInTable="columnColours"
msprop:Generator_ColumnPropNameInTable="ColoursColumn" minOccurs="0">
            <xs:simpleType>
              <xs:restriction base="xs:string">
                <xs:maxLength value="250" />
              </xs:restriction>
            </xs:simpleType>
          </xs:element>
          <xs:element name="Image" msprop:Generator_UserColumnName="Image"
msprop:Generator_ColumnPropNameInRow="Image"
msprop:Generator_ColumnVarNameInTable="columnImage"
msprop:Generator_ColumnPropNameInTable="ImageColumn" minOccurs="0">
            <xs:simpleType>
              <xs:restriction base="xs:string">
                <xs:maxLength value="250" />
              </xs:restriction>
            </xs:simpleType>
```

```
        </xs:element>
        <xs:element name="Comments" msprop:Generator_UserColumnName="Comments"
msprop:Generator_ColumnPropNameInRow="Comments"
msprop:Generator_ColumnVarNameInTable="columnComments"
msprop:Generator_ColumnPropNameInTable="CommentsColumn" minOccurs="0">
          <xs:simpleType>
            <xs:restriction base="xs:string">
              <xs:maxLength value="250" />
            </xs:restriction>
          </xs:simpleType>
        </xs:element>
        <xs:element name="POTurk" msprop:Generator_UserColumnName="POTurk"
msprop:Generator_ColumnPropNameInRow="POTurk"
msprop:Generator_ColumnVarNameInTable="columnPOTurk"
msprop:Generator_ColumnPropNameInTable="POTurkColumn" minOccurs="0">
          <xs:simpleType>
            <xs:restriction base="xs:string">
              <xs:maxLength value="90" />
            </xs:restriction>
          </xs:simpleType>
        </xs:element>
        <xs:element name="Vilaet" msprop:Generator_UserColumnName="Vilaet"
msprop:Generator_ColumnPropNameInRow="Vilaet"
msprop:Generator_ColumnVarNameInTable="columnVilaet"
msprop:Generator_ColumnPropNameInTable="VilaetColumn" minOccurs="0">
          <xs:simpleType>
            <xs:restriction base="xs:string">
              <xs:maxLength value="50" />
            </xs:restriction>
          </xs:simpleType>
        </xs:element>
        <xs:element name="Country" msprop:Generator_UserColumnName="Country"
msprop:Generator_ColumnPropNameInRow="Country"
msprop:Generator_ColumnVarNameInTable="columnCountry"
msprop:Generator_ColumnPropNameInTable="CountryColumn" minOccurs="0">
          <xs:simpleType>
            <xs:restriction base="xs:string">
              <xs:maxLength value="50" />
            </xs:restriction>
          </xs:simpleType>
        </xs:element>
        <xs:element name="POOtto" msprop:Generator_UserColumnName="POOtto"
msprop:Generator_ColumnPropNameInRow="POOtto"
msprop:Generator_ColumnVarNameInTable="columnPOOtto"
msprop:Generator_ColumnPropNameInTable="POOttoColumn" minOccurs="0">
          <xs:simpleType>
            <xs:restriction base="xs:string">
              <xs:maxLength value="50" />
            </xs:restriction>
          </xs:simpleType>
        </xs:element>
        <xs:element name="ModLatin" msprop:Generator_UserColumnName="ModLatin"
msprop:Generator_ColumnPropNameInRow="ModLatin"
msprop:Generator_ColumnVarNameInTable="columnModLatin"
msprop:Generator_ColumnPropNameInTable="ModLatinColumn" minOccurs="0">
          <xs:simpleType>
            <xs:restriction base="xs:string">
              <xs:maxLength value="50" />
            </xs:restriction>
          </xs:simpleType>
```

```
            </xs:element>
            <xs:element name="ModMulti" msprop:Generator_UserColumnName="ModMulti"
msprop:Generator_ColumnPropNameInRow="ModMulti"
msprop:Generator_ColumnVarNameInTable="columnModMulti"
msprop:Generator_ColumnPropNameInTable="ModMultiColumn" minOccurs="0">
              <xs:simpleType>
                <xs:restriction base="xs:string">
                  <xs:maxLength value="50" />
                </xs:restriction>
              </xs:simpleType>
            </xs:element>
          </xs:sequence>
        </xs:complexType>
      </xs:element>
    </xs:choice>
  </xs:complexType>
  <xs:unique name="Constraint1" msdata:PrimaryKey="true">
    <xs:selector xpath=".//mstns:Cancellations" />
    <xs:field xpath="mstns:ID" />
  </xs:unique>
 </xs:element>
</xs:schema>
```

A21) αρχείο Web.Config :

```
<?xml version="1.0"?>
<configuration>
            <appSettings/>
            <connectionStrings>
  <add name="Cancel_Alexandros_Apr09ConnectionString"
connectionString="Provider=Microsoft.Jet.OLEDB.4.0;Data Source=|
DataDirectory|\Cancel_Alexandros_Apr09.mdb"
   providerName="System.Data.OleDb" />
 </connectionStrings>
            <system.web>
                        <compilation debug="true"/>
                        <authentication mode="Windows"/>
            </system.web>
</configuration>
```