ΕΘΝΙΚΌ ΜΕΤΣΌΒΙΟ ΠΟΛΥΤΕΧΝΕΊΟ
ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ
ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ
ΤΟΜΕΑΣ ΣΥΣΤΗΜΑΤΩΝ ΜΕΤΑΔΟΣΗΣ ΠΛΗΡΟΦΟΡΙΑΣ
ΚΑΙ ΤΕΧΝΟΛΟΓΙΑΣ ΥΛΙΚΩΝ

Ανάπτυξη πλατφόρμας καταγραφής και επεργασίας ιατρικών ερωτηματολογίων

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

Φώτιος Κ. Τσουκαλάς

Επιβλέπων : Γεώργιος Κ. Ματσόπουλος
Καθηγητης Ε.Μ.Π.

Αθήνα,Ιούνιος 2022

ΕΘΝΙΚΌ ΜΕΤΣΌΒΙΟ ΠΟΛΥΤΕΧΝΕΊΟ
ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ
ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ
ΤΟΜΕΑΣ ΣΥΣΤΗΜΑΤΩΝ ΜΕΤΑΔΟΣΗΣ ΠΛΗΡΟΦΟΡΙΑΣ
ΚΑΙ ΤΕΧΝΟΛΟΓΙΑΣ ΥΛΙΚΩΝ

Ανάπτυξη πλατφόρμας καταγραφής και επεργασίας ιατρικών ερωτηματολογίων

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

Φώτιος Κ. Τσουκαλάς

Επιβλέπων : Γεώργιος Κ. Ματσόπουλος
Καθηγητής Ε.Μ.Π.

Εγκρίθηκε από την τριμελή εξεταστική επιτροπή την 29η Ιουνίου 2022

……	……	…….
Διονύσιος-Δημήτριος Κουτσούρης **Καθηγητής ΕΜΠ**	**Γεώργιος Ματσόπουλος** **Καθηγητής ΕΜΠ**	**ΠαναγιώτηςΤσανάκας** **Καθηγητής ΕΜΠ**

Αθήνα,Ιούνιος 2022

.....

Φώτιος Τσουκαλάς

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

Περίληψη

Το αντικείμενο της παρούσας εργασίας είναι το ιατρικό ερωτηματολόγιο και τρόποι τυποποίησης της αποθήκευσής του αλλά και χρήσης του. Στα πλάισια αυτής της εργασίας έγινε ανάλυση των διαφόρων ερωτηματολογίων που ήδη υπάρχουν, ως προς τον τρόπο σύνταξής τους, το περιεχόμενο των ερωτήσεων και τον τρόπο απάντησης τους, με σκοπό την κατασκευή μίας τυποποιημένης μεθόδου αποθήκευσής τους. Βρέθηκε ότι τα ιατρικά ερωτηματολόγια δεν διαφέρουν πολύ μεταξύ τους ως προς τον τύπο των ερωτήσεων ή ως προς τη γενικότερη μορφή τους, αλλά μόνο ως προς το περιεχόμενό τους, με αποτέλεσμα να είναι εφικτή η τυποποίησή τους .

Κατασκευάστηκε ένα πρότυπο αποθήκευσης κάθε ερωτηματολογίου σε JSON, με μεγάλη ευελιξία διαμόρφωσης, που μπορεί να περιγράψει μια πληθώρα ερωτηματολογίων , ανεξαρτήτως του ιατρικού πεδίου που θα χρησιμοποιηθούν , του τρόπου που λαμβάνονται αλλά και της κατηγορίας του ασθενούς. Έπειτα κατασκευάστηκε κατάλληλη εφαρμογή ,η οποία, βασιζόμενη στο νέο πρότυπο , επιτρέπει την κατασκευή του αρχείου JSON που περιγράφει το ερωτηματολόγιο μέσα από μία εύχρηστη φόρμα, καθώς και την αποθήκευσή του σε βάση δεδομένων. Με αυτόν τον τρόπο γίνεται αυτόματα, γρήγορα και εύκολα η επίπονη διαδικασία σύνταξης του αρχείου JSON που περιγράφει το ερωτηματολόγιο.

Η ίδια εφαρμογή, χάριν επίδειξης της λειτουργικότητας του νέου προτύπου, παρέχει στον χρήστη μία βασική ικανότητα ανάθεσης ερωτηματολογίου σε ασθενή, καθώς και αποτελεί διεπαφή χρήστη για την συμπλήρωση του ερωτηματολογίου από τον ασθενή. Τέλος, στα πλαίσια της τελευταίας λειτουργίας, αποθηκεύονται οι απαντήσεις σε επίσης τυποποιημένη και συμβατή μορφή στην ίδια βάση δεδομένων.

Βασικός γνώμονας όλης της εργασίας είναι η διαλειτουργικότητα του προτύπου, φαίνεται πρωταρχικά από τις ελάχιστες παραδοχές που έγιναν για την κατασκευή του προτύπου αποθήκευσης σε JSON. Όλη η πληροφορία για ένα ερωτηματολόγιο βρίσκεται στο JSON αρχείο του, δεν χρειάζεται κάποιος προγραμματιστής να κάνει επιπλέον παραδοχές για την χρήση του προτύπου. Η διαλειτουργικότητα αυτή γίνεται εμφανής στη δεύτερη διεπαφή χρήστη για την συμπλήρωση ερωτηματολογίου που κατασκευάστηκε σε μια εντελώς διαφορετική γλώσσα από την πρώτη.

Λέξεις κλειδιά: ιατρικό ερωτηματολόγιο,εφαρμογή, συμπλήρωση , αποθήκευση, διαχείριση

Abstract

The subject of this thesis is the medical questionnaire and ways to standardize its storage and use. In this project, various questionnaires that already exist were analysed , in terms of how they are structured, of the content of their questions and the way a patient should answer every one of them, in order to construct a standard method of storing them. It was found that medical questionnaires do not differ much from each other in terms of the type of questions made , but only in terms of their content, so it is easy to store them in a standardised form .

A template for storing each questionnaire in JSON was developed, which provides great flexibility around the question types and amount, and therefore can describe a variety of questionnaires, regardless of the medical field they are used for , their way of completion and the patient they are offered to. Afterwards,an appropriate application was built, which fulfills the construction of the JSON file based on the new standard, by providing an easy-to-use form to create the JSON file, as well as managing its storage in a database. This allows the process of writing the JSON file to happen automatically and in a few minutes.

The same application also features an interface of assigning a questionnaire to a patient and of completing the questionnaire by the patient in order to demonstrate the functionality of the new standard. Additionally, the responses of the patient are stored in a JSON in a standard manner . The application provides the user with response management.

The main strength of this project is the interoperability of the JSON template, that is used to store the questionnaire definitions. Minimal assumptions were made in the construction of the JSON storage template. The entirety of the information concerning a questionnaire is in its JSON file, any developer that may use this template should not need to make any additional assumptions about it. This interoperability becomes apparent with the second user interface that was created, that provides an alternate way of completing a questionnaire, while built on a fundamentally different framework.

Keywords: medical questionnaire, application,completion,storing,management

Ευχαριστίες

Με το τέλος αυτού του κεφαλαίου της ζωής μου θα ήθελα να ευχαριστήσω τους ανθρώπους που με έφεραν ως εδώ.

Πρώτα από όλους , ευχαριστώ τους γονείς μου, τον Κώστα και την Ειρήνη, που με υπομονή , επιμονή και πείσμα καλλιέργησαν την πλειονότητα των ικανοτήτων μου.

Τον αδερφό μου, Σπύρο, που ως συνοδοιπόρος και συναγωνιστής μου με συντρόφεψε όλα τα προηγούμενα χρόνια και με ώθησε να είμαι καλύτερος.

Τους νονούς μου , Κώστα και Γιάννη , των οποίων η καθοδήγηση και αρωγή ήταν ανεκτίμητης αξίας.

Επιπλέον ,θέλω να ευχαριστήσω τους καθηγητές κ. Κουτσούρη και κ. Ματσόπουλο για την ευκαιρία να δουλέψω πάνω σε αυτή την εργασία.

Τέλος , ευχαριστώ τον κ. Ιωάννη Κουρή, ο οποίος με βοήθησε καθ'όλη τη διάρκεια αυτής της εργασίας, παρέχοντάς μου πολύτιμες συμβουλές αλλά και οδηγώντας με να ανακαλύψω εργαλεία που θα μου είναι σίγουρα χρήσιμα στο μέλλον.

Κατάλογος περιεχομένων

Κατάλογος εικόνων

Ευρετήριο πινάκων

Εισαγωγή

1.Σκοπός της διπλωματικής και περιγραφή του προβλήματος

Η παρούσα διπλωματική πραγματεύεται το ιατρικό ερωτηματολόγιο. Ένα ιατρικό ερωτηματολόγιο είναι μία συγκεκριμένη αλληλουχία ερωτήσεων που απευθύνονται σε ασθενείς ατομικά, με σκοπό τη συλλογή δεδομένων, στα πλαίσια είτε ιατρικής έρευνας ή διαδικασίας διάγνωσης του ασθενούς. Τα ιατρικά ερωτηματολόγια μπορεί να είναι ανώνυμα ή επώνυμα, συμπληρώνονται από τον ασθενή ή συμπληρώνονται από κάποιον υγειονομικό σε συνέντευξη [1]με τον ασθενή. Περιέχουν συνήθως ερωτήσεις που αντλούν δεδομένα για τα προσωπικά χαρακτηριστικά του ασθενή, όπως η ηλικία και το φύλο, αλλά κυρίως ερωτήσεις που αφορούν άμεσα την υγεία του, όπως “Πόσο συχνά αισθάνεστε δυσφορία;”

Τα ιατρικά ερωτηματολόγια , παρά τα πολλά κοινά χαρακτηριστικά που εμφανίζουν μεταξύ τους ανεξαρτήτως πεδίου χρήσης, είτε ποιοτικά ή ποσοτικά, κατασκευάζονται κατά ανάγκη από το μηδέν . Αυτό οδηγεί σε ασυμβατότητα μεταξύ τους σε επίπεδο μορφής και σε επίπεδο κωδικοποίησης .Δεν επιτρέπει άμεση σύγκριση μεταξύ τους χωρίς την παρέμβαση ανθρώπου.

Σκοπός της συγκεκριμένης εργασίας είναι η τυποποίηση του ιατρικού ερωτηματολογίου .Στα πλαίσια αυτής της τυποποίησης , προτείνεται μία κατάλληλη μέθοδος ηλεκτρονικής αποθήκευσης ερωτηματολογίων σε json που καταφέρνει να καλύψει μια ευρεία γκάμα ιατρικών ερωτηματολογίων που υπάρχουν. Σε δεύτερο λόγο κατασκευάστηκε εφαρμογή δημιουργίας και διαχείρισης ερωτηματολογίων για χρήση από ερευνητικό ή ιατρο-νοσηλευτικό προσωπικό καθώς και παροχής ερωτηματολογίων σε ασθενείς. Η εφαρμογή αυτή καταδεικνύει πλήρως την δημιουργία και επεργασία των ερωτηματολογίων στα όρια της τυποποίησης που προτείνεται, περιλαμβάνει βάση δεδομένων που δέχεται τα ερωτηματολόγια καθώς και παρέχει περιορισμένο έλεγχο χρηστών και ασθενών. Η εφαρμογή αυτή προσφέρεται σε 2 πλατφόρμες αλληλεπίδρασης: σε web(Blazor) και mobile (Flutter). Η υλοποίηση σε μορφή σελίδας διαδικτύου εμπεριέχει όλες τις λειτουργίες που

προαναφέρθηκαν, η υλοποίηση σε εφαρμογή κινητού τηλεφώνου εμπεριέχει μόνο την λειτουργία συμπλήρωσης ερωτηματολογίου από ασθενή.

2.Περιγραφή της προτεινόμενης λύσης

Η λύση που προτάθηκε στην παρούσα εργασία για την περιγραφή ενός ερωτηματολογίου είναι η ολοκληρωμένη κωδικοποίηση του περιεχομένου του σε ένα προκαθορισμένο τύπο JSON. Η λογική πίσω από αυτή τη λύση είναι η επάρκεια της πληροφορίας ενός και μόνο JSON αρχείου ενός ερωτηματολογίου για την ανασύνταξη του σε χρήσιμη μορφή. Δηλαδή, ένας ερευνητής που θα αποκτήσει το JSON που αντιστοιχεί σε ένα συγκεκριμένο ερωτηματολόγιο, δεν χρειάζεται καμία παραπάνω πληροφορία για να το μεταφέρει σε χρήσιμη μορφή . Το JSON παρέχει όλες τις οδηγίες, τις ερωτήσεις και τις απαντήσεις -ή τον τύπου απάντησης αν είναι ανοικτού τύπου ερώτηση– κάθε ζητούμενου,τις προϋποθέσεις για διακλάδωση ως προς την άσκηση ή όχι κάθε ερώτησης, καθώς και τον τρόπο βαθμολογίας κάθε κλίμακας του ερωτηματολογίου. Με αυτόν τον τρόπο δε χρειάζεται ο ερευνητής να αναζητήσει κάποιο επιπλέον έγγραφο για να χρησιμοποιήσει το ερωτηματολόγιο.

Σε δεύτερο λόγο , η τυποποίηση του ερωτηματολογίου σε μία τόσο πλήρη μορφή επιτρέπει και σε προγραμματιστές συστημάτων και τεχνολογίας υγείας να δημιουργήσουν και εφαρμογές γύρω από αυτήν. Γίνεται εμφανές στο δεύτερο σκέλος της παρούσας εργασίας πόσο απρόσκοπτα το ίδιο αρχείο JSON μπορεί να χρησιμοποιηθεί σε δύο διαφορετικά πλαίσια προγραμματισμού με τον ίδιο ακριβώς τρόπο .Η σύνταξη και απάντηση του ερωτηματολογίου γίνεται το ίδιο εύκολα σε διαφορετικές πλατφόρμες , ανεξαρτήτως του περιεχομένου και της φύσης του ερωτηματολογίου. Προβάλλεται στην εργασία αυτή η αξία μέσα από την λειτουργικότητα των εφαρμογών.

3.Διάρθρωση διπλωματικής και σύντομη ανασκόπηση των κεφαλαίων

Με την σειρά που ακολουθούν, σε αυτή τη διπλωματική εργασία:

- αναλύονται ποιοτικά και ποσοτικά τα μοτίβα που διέπουν τα ιατρικά ερωτηματολόγια

- αναλύεται η τυποποίηση που προτείνεται , με παραδείγματα τυποποίησης ερωτηματολογίων

- αναλύεται η δομή και λειτουργία της εφαρμογής που κατασκευάστηκε και που αναδεικνύει αυτή την τυποποίηση.

Κεφάλαιο 1: Τα ιατρικά ερωτηματολόγια

1.1 Τι είναι το ερωτηματολόγιο

Το ερωτηματολόγιο είναι ένα ερευνητικό εργαλείο που αποτελείται από ένα σύνολο ερωτήσεων (ή άλλων τύπων προτροπών) με σκοπό τη συλλογή πληροφοριών από τους ερωτηθέντες στα πλαίσια έρευνας ή στατιστικής μελέτης [2]. Ένα ερευνητικό ερωτηματολόγιο είναι συνήθως ένας συνδυασμός ερωτήσεων κλειστού τύπου και ερωτήσεων ανοιχτού τύπου. Οι ανοιχτές ερωτήσεις προσφέρουν στον ερωτώμενο την ικανότητα να εκφραστεί σε μεγαλύτερο βαθμό . Το ερευνητικό ερωτηματολόγιο αναπτύχθηκε από τη Στατιστική Εταιρεία του Λονδίνου το 1838.Στα πλεονεκτήματα της χρήσης ενός ερωτηματολογίου για την συλλογή πληροφορίας είναι το σχετικά χαμηλό κόστος , η συνήθως εύκολη και γρήγορη συμπλήρωσή τους από τον ερωτώμενο , αλλά και η απλότητα της συγκέντρωσης δεδομένων, ειδικά στην περίπτωση των ερωτήσεων κλειστού τύπου. Στα μειονεκτήματα της χρήσης του ερωτηματολογίου, είναι η προαπαίτηση το άτομο που το συμπληρώνει να έχει την ικανότητα να το απαντήσει , καθώς και ο περιορισμός που επιβάλλουν στα δεδομένα που συγκεντρώνονται οι συγκεκριμένες ερωτήσεις, όπως και οι απαντήσεις των ερωτήσεων κλειστού τύπου. Ένα ερωτηματολόγιο απαιτεί πρωταρχικά την ικανότητα ανάγνωσής του .

1.2 Πως λειτουργεί το ιατρικό ερωτηματολόγιο

Το ιατρικό ερωτηματολόγιο ακολουθεί την ίδια μορφή , αλλά πέρα από τη χρήση για στατιστική μελέτη ή έρευνα , χρησιμοποιείται και ως διαγνωστικό εργαλείο[3]–[5]. Η χρήση του γίνεται είτε αυτοτελώς, δηλαδή δίνεται στον ασθενή για να το συμπληρώσει μόνος του, ή ως εργαλείο από τον επαγγελματία υγείας, του οποίου η θέση είναι να ρωτάει τον ασθενή ενώ κατευθύνεται από αυτό και να συγκεντρώσει τα επιθυμητά .Το δεύτερο ενδεχόμενο συνήθως υλοποιείται σε περιπτώσεις που ο ασθενής θεωρείται ότι δε μπορεί να το απαντήσει μόνος του, ή όταν ο ερευνητής πρέπει να κρίνει την απόκριση του ασθενούς στην προτροπή (ή ερώτηση).

Τα ιατρικά ερωτηματολόγια συνήθως διαθέτουν κλίμακες , στις οποίες βαθμολογούνται οι απαντήσεις των ερωτηθέντων. Οι κλίμακες αυτές μπορεί να είναι κρυφές ή φανερές στον ασθενή. Στις ερωτήσεις κλειστού τύπου , κάθε απάντηση συνήθως συνοδεύεται από μια διαφορετική βαθμολογία . Με την δημιουργία του ερωτηματολογίου διεξάγεται συνήθως μία στατιστική μελέτη σε ένα όσο δυνατόν πιο

αντιπροσωπευτικό κατά τον ερευνητή σύνολο ανθρώπων, και ανάλογα των απαντήσεων τους , της βαθμολογίας τους και της κατάστασής τους, εξάγονται συμπεράσματα συνδεδεμένα με τις κλίμακες . Για παράδειγμα,στο ερωτηματολόγιο GAD-7 (Generalized Anxiety Disorder 7)[6] , το αποτέλεσμα στην βασική κλίμακα συσχετίζεται με το ρίσκο της κατάστασης του ασθενούς : έως 4 λίγο ως ανύπαρκτο, 5-9 ελαφρύ,10-14 μέτριο,15+ έντονο. Ένα ιατρικό ερωτηματολόγιο μπορεί να έχει και περισσότερες από μία κλίμακες. Όπως για παράδειγμα το QLQ-PR25[7] ,το οποίο διαθέτει 6 διαφορετικές κλίμακες , κάθε μία εξαρτώμενη από διαφορετικές ερωτήσεις ,δημιουργώντας ένα ευρύ φάσμα δεικτών.

Τα ιατρικά ερωτηματολόγια χρησιμοποιούνται σε διάφορους τομείς της υγείας. Χρησιμοποιούνται στον ερευνητικό τομέα, όπως παραδείγματος χάριν στην έρευνα για ένα νέο φάρμακο . Στη διάγνωση ασθένειας, ψυχολογικής όπως και με το προαναφερθέν GAD-7, αλλά και άλλης μορφής , όπως γηριατρικά σύνδρομα , με το RGA (Rapid Geriatric Assessment),παθολογικής φύσεως ,κ.α. [8]–[10]

1.3 Ανάλυση των ερωτηματολογίων

Τα ερωτηματολόγια , όπως προαναφέρθηκε , στην ουσία τους είναι απλά ένα συγκεκριμένο σύνολο ερωτήσεων , το οποίο μπορεί να συνοδεύεται από κάποιες επιπλέον οδηγίες. Η ανάλυση των ιατρικών ερωτηματολογίων έγινε περισσότερο σε ποσοτικό άξονα παρά ποιοτικό: δεν εξαρτάται από τη φύση του πεδίου στο οποίο τοποθετείται η ερώτηση, ούτε από το σκοπό για τον οποίο δημιουργήθηκε το ερωτηματολόγιο, παρά από το πλήθος των ερωτήσεων, τη διαρρύθμιση τους και τον τύπο της απάντησης κάθε μίας από αυτές. Επιπλέον , δόθηκε προσοχή στην καταγραφή των μετα-δεδομένων των ερωτηματολογίων, όπως το αναγνωριστικό τους, έκδοση, πληροφορίες δημιουργού και άδεια χρήσης.

Οι ερωτήσεις διακρίθηκαν ως προς τον τύπο της απάντησης σε :

- Ανοικτού τύπου:

- Κειμένου
- Αμιγώς αριθμητικές

- Κλειστού τύπου:
  - Πολλαπλής επιλογής με πολλές επιλογές
  - Πολλαπλής επιλογής με μοναδική επιλογή

Τα περισσότερα ιατρικά ερωτηματολόγια από αυτά που μελετήθηκαν διαθέτουν κυρίως ερωτήσεις κλειστού τύπου μοναδικής επιλογής, όμως έγινε τέτοια ανάλυση ώστε να καλυφθεί πληθώρα τύπων ερωτηματολογίων.

Ως προς τη διαρρύθμιση των ερωτηματολογίων, παρατηρήθηκε η ομαδοποίηση των ερωτήσεων[3] , συνήθως σε συνδυασμό με μία αποκλειστική σε αυτές προτροπή ή οδηγία. Γι'αυτόν τον λόγο θεωρήθηκε ένα μοντέλο ομάδας ερωτήσεων το οποίο ομαδοποιεί τις ερωτήσεις που θέλει ο δημιουργός του να συνοδεύονται από κάποιο κείμενο. Το μοντέλο αυτό ,πέρα από την κανονική του λειτουργία, είτε μπορεί να αγνοηθεί εντελώς, συγκεντρώνοντας όλες τις ερωτήσεις ενός ερωτηματολογίου σε μία ομάδα, είτε μπορεί να λειτουργήσει και σαν δοχείο ενδιάμεσων οδηγιών , αν δεν προστεθούν ερωτήσεις στην ομάδα, παρά διαθέτει μόνο το κείμενο.

Τέλος, επειδή σε κάποια ερωτηματολόγια[7] υπάρχουν ερωτήσεις που πρέπει να απαντηθούν μόνο περιστασιακά και ερωτήσεις που δεν είναι απαραίτητο να συμπληρωθούν, καταστρώθηκε ένα σύστημα σήμανσης των ερωτήσεων , το οποίο τις ελέγχει δυναμικά στο τελική διεπαφή χρήστη.

Διακρίθηκαν οι ερωτήσεις σε υποχρεωτικές και μη, καθώς και σε ερωτήσεις με προυπόθεση ή όχι.Για παράδειγμα , στο ερωτηματολόγιο….

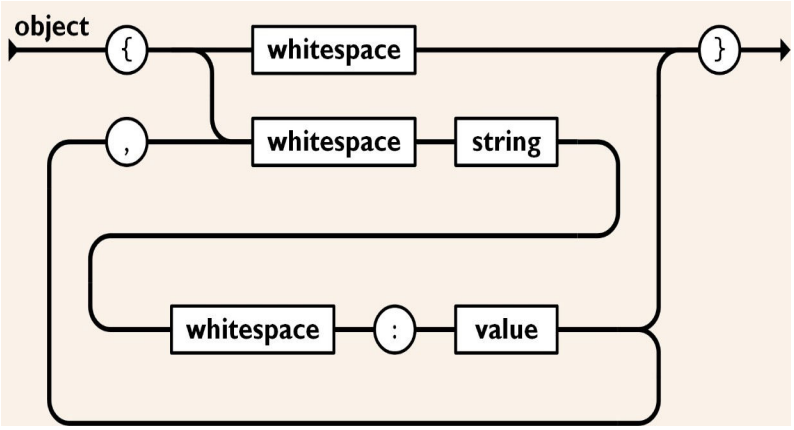
Ως προς τη βαθμολόγηση του ερωτηματολογίου, δημιουργήθηκε ένας τρόπος περιγραφής κλιμάκων και καταγραφής βαθμολογίας ανά ερώτηση. Κάθε κλίμακα διαθέτει το αναγνωριστικό των ερωτήσεων από τις οποίες βαθμολογείται καθώς και τη συνάρτηση που χρησιμοποιείται για την βαθμολόγηση. Οι συναρτήσεις περιγράφονται με αναγνωριστικό όνομα , όπως sum(άθροισμα) και avg(μέσος όρος).Σε κάθε ερώτηση κλειστού τύπου κάθε απάντηση συνοδεύεται από την βαθμολογική της αξία. Οι ερωτήσεις ανοικτού τύπου κειμένου θεωρήθηκε ότι δεν μπορούν να βαθμολογηθούν αυτόματα, ενώ οι αριθμητικές παρέχουν ως βαθμό την τιμή που απαντάται από τον ασθενή.

Κεφάλαιο 2:Τεχνολογίες υλοποίησης

2.1 Το πρότυπο JSON

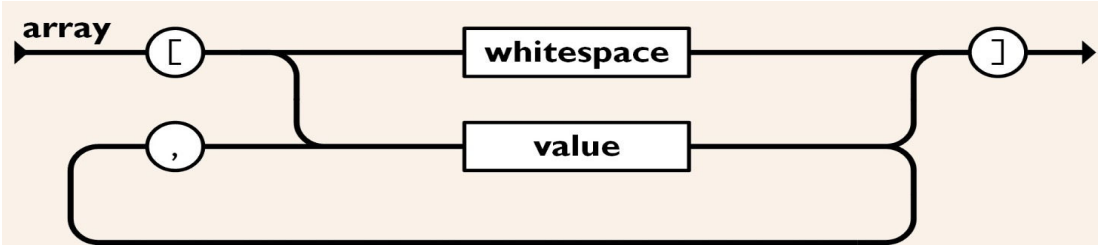
Το πρότυπο JSON (JavaScript Object Notation) είναι ένα ανοιχτό πρότυπο αρχείου που χρησιμοποιείται για την ανταλλαγή δεδομένων[11].Χρησιμοποιεί κείμενο που μπορεί να διαβαστεί από τον άνθρωπο και αποτελείται από ζεύγη ιδιοτήτων-τιμών και πινάκων ή γενικά αριθμήσιμων οντοτήτων.Είναι ανεξάρτητο από γλώσσες προγραμματισμού, κατα συνέπεια μπορεί να χρησιμοποιηθεί από πληθώρα προγραμμάτων . Διαθέτει συγκεκριμένη δομή , η οποία έχει προσυμφωνηθεί και υποστηρίζεται στις γλώσσες προγραμματισμού της οικογένειας C. Συνοπτικά, η δομή ενός JSON αρχείου έχει ως εξής :

- Ένα αντικείμενο (object) είναι ένα άτακτο σύνολο από ζευγάρια ονομάτων/τιμών (Εικόνα 1). Ένα αντικείμενο (object) ξεκινάει με αριστερό άγκιστρο και τελειώνει με δεξιό άγκιστρο. Κάθε όνομα ακολουθείται από άνω-κάτω τελεία και τα ζευγάρια ονόματος/τιμής χωρίζονται από κόμμα.



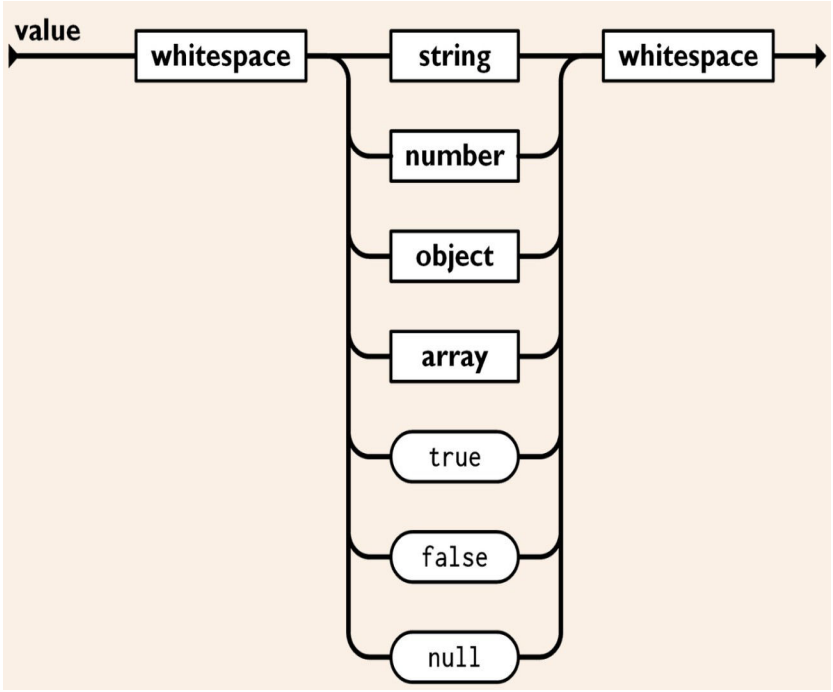
Εικόνα 1: Το αντικείμενο σε JSON

- Ένας πίνακας (array) είναι μια συλλογή από τιμές σε σειρά (Εικόνα 2). Ένας πίνακας (array) ξεκινάει με αριστερή αγκύλη και τελειώνει με δεξιά αγκύλη. Οι τιμές χωρίζονται με κόμμα.



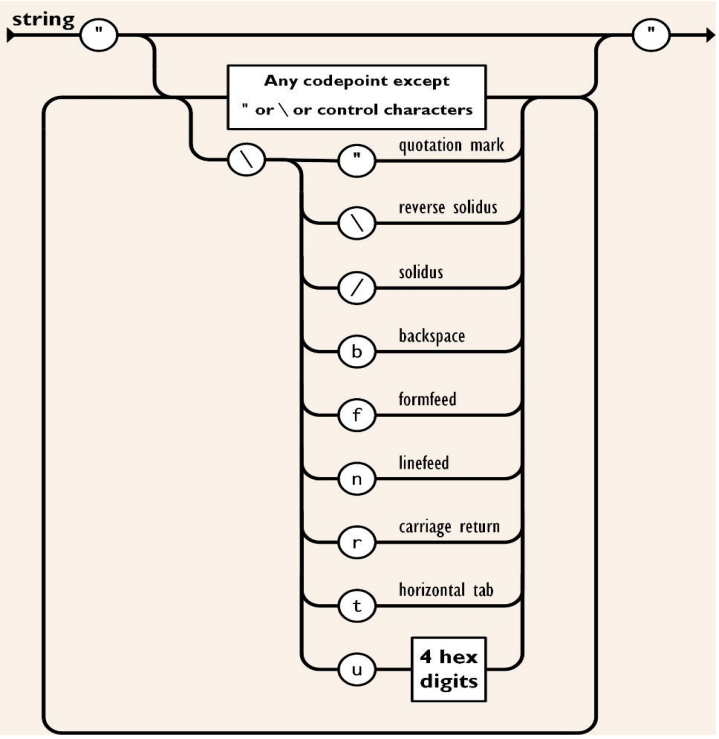
Εικόνα 2: Πίνακας σε JSON

- Μία τιμή μπορεί να είναι string μέσα σε διπλά quotes, ή αριθμός (number), ή true ή false ή null, ή αντικείμενο (object) ή πίνακας (array). Αυτές οι τιμές μπορεί να είναι και ανακατεμένες(Εικόνα 3).



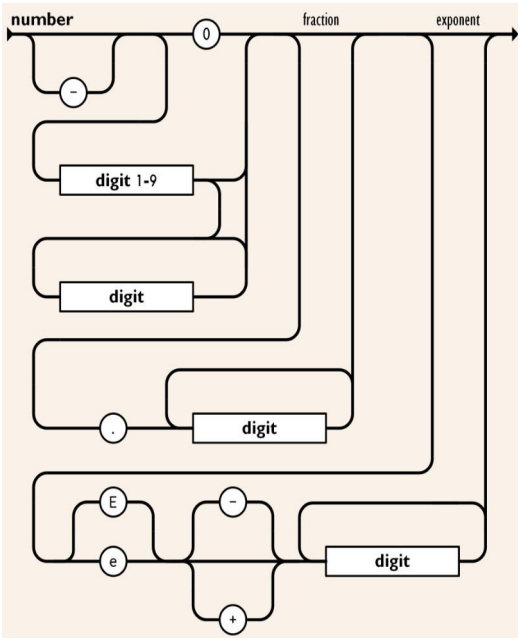
Εικόνα 3: Ενδεχόμενα τιμής σε JSON

- Ένα string είναι μια συλλογή από μηδέν ή περισσότερους Unicode χαρακτήρες, μέσα σε διπλά quotes, χρησιμοποιώντας αντίστροφους κάθετους \ backslash για escapes(Εικόνα 4). Ένας χαρακτήρας αντιπροσωπεύεται ως ένας μονός χαρακτήρας string. Ένα string μοιάζει πολύ σαν ένα C ή Java string.



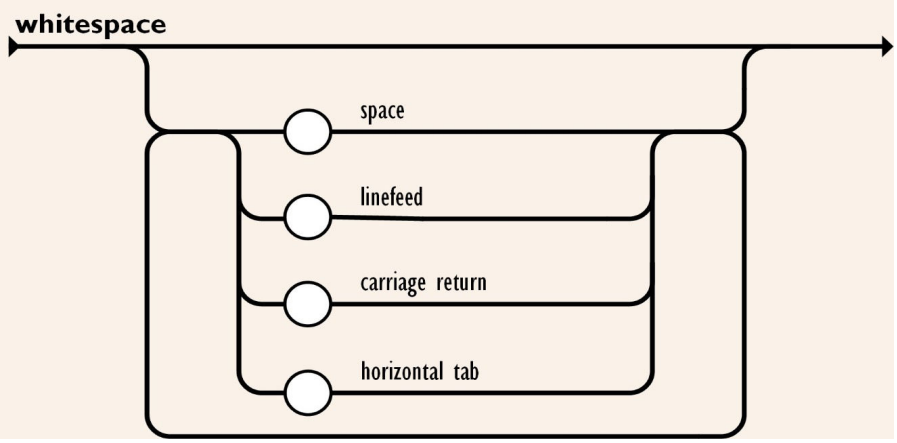
Εικόνα 4: Συμβολοσειρές σε JSON

- Ένας αριθμός (number) μοιάζει πάρα πολύ με ένα C ή Java αριθμό (number), με την διαφορά πως τα οκταδικά και δεκαεξαδικά συστήματα δεν χρησιμοποιούνται (Εικόνα 5).



Εικόνα 5: Οι αριθμοί σε JSON

- Τα κενά (whitespace) μπορούν να εισαχθούν ανάμεσα σε οποιοδήποτε ζευγάρι tokens (Εικόνα 6). Με εξαίρεση μερικών λεπτομερειών κωδικοποίησης (encoding), αυτό περιγράφει γενικότερα την γλώσσα (προγραμματισμού).



Εικόνα 6: Κενός χώρος σε JSON

2.2 Καθορισμός προδιαγραφών για την δημιουργία ερωτηματολογίου

Για την περιγραφή του ορισμού κάθε ερωτηματολογίου προτείνεται μία συγκεκριμένη δομή JSON αρχείου. Στόχος είναι για κάθε ερωτηματολόγιο να απαιτείται μόνο ένα JSON αρχείο και κάθε JSON αρχείο να παρέχει όλη την απαραίτητη πληροφορία για την δημιουργία και χρήση ενός ερωτηματολογίου. Το πρότυπο δημιουργήθηκε έχοντας κατα νου ως πρωταρχικό χαρακτηριστικό την

Κάθε JSON αποτελείται από ένα αντικείμενο,το αντικείμενο του ερωτηματολογίου. Τ αντικείμενο αυτό έχει την παρακάτω δομή:

- questionnaire_definition (string) :Η κωδική ονομασία του ερωτηματολογίου
- version (int): Ο αριθμός έκδοσης του ερωτηματολογίου
- language (string) :Η γλώσσα του ερωτηματολογίου κωδικοποιημένη σε ISO 639-1
- question_group (πίνακας αντικειμένων question_group) :

  Κάθε αντικείμενο question_group είναι της μορφής:
  - title (string):

- questions (πίνακας αντικειμένων question):
  Κάθε αντικείμενο question είναι της μορφής:
  - number (int) :το αριθμητικό αναγνωριστικό της ερώτησης
  - question (string): Η ερώτηση
  - type (string): Ο τύπος της ερώτησης,αναλύονται παρακάτω ποιοί τύποι έχουν οριστεί.
  - choices (πίνακας αντικειμένων choice): Οι πιθανές απαντήσεις στην ερώτηση, αν είναι ανοικτού τύπου είναι κενός πίνακας.
    Κάθε αντικείμενο choice είναι της μορφής:
    - point (int):Ο βαθμός της απάντησης
    - content (string):Το περιεχόμενο της απάντησης
  - nullable (boolean): Αν η ερώτηση μπορεί να μην απαντηθεί
  - condition (αντικείμενο τύπου condition): Παρέχει πληροφορία ως προς την προυπόθεση ,αν υπάρχει , να εμφανιστεί η ερώτηση.
    - hascondition (boolean):Αν υπάρχει προυπόθεση
    - groupnumber (int) :Σε ποια ομάδα ανήκει η ερώτηση ελεγκτής
    - questionumber (int) :Η θέση της εντός της ομάδας.
    - qvalue (string): Η τιμή της απάντησης που χρειάζεται για να εμφανιστεί η ερώτηση. Αναλύεται παρακάτω πως λειτουργεί σε αριθμητικές ερωτήσεις-ελεγκτές.

- text (string):Πεδίο εισαγωγής γενικών οδηγιών προς τον ασθενή ως προς το ερωτηματολόγιο.

- scales (πίνακας αντικειμένων scale):Το σύνολο των βαθμολογικών κλιμάκων.
  Κάθε αντικείμενο scale αποτελείται από:
  - name (string): Το όνομα της κλίμακας
  - type (string): Ο τύπος υπολογισμού της κλίμακας. Οι τύποι διαθέτουν ονόματα και αποτελούν μία από τις παραδοχές του μοντέλου.Αναλύεται παρακάτω ποιοί έχουν οριστεί και πως μπορούν να επεκταθούν.
  - questions (πίνακας από int): περιέχει τα αριθμητικά αναγνωριστικά των ερωτήσεων που αφορά η κλίμακα

- CreatorInformation (string):Τρόποι επικοινωνίας και γενικές πληροφορίας για τον δημιουργό του ερωτηματολογίου.

- CopyrightNotice (string):Η πλήρης δήλωση πνευματικών δικαιωμάτων

- CopyrightLicence (string): Το αναγνωριστικό άδειας χρήσης, οπως π.χ. All Rights Reserved

2.3 Παράδειγμα μεταφοράς ορισμού ερωτηματολογίου σε JSON

Λόγω του μεγέθους του ερωτηματολογίου , θα παρουσιαστει ένα αντιπροσωπευτικό της πλειονότητας, το ESS, μερικώς (Εικόνα 7).

2.3.1 ESS (Epworth Sleepiness Scale)

How likely are you to doze off or fall asleep in the following situations, in contrast to feeling just tired?

This refers to your usual way of life in recent times.

Even if you haven't done some of these things recently try to work out how they would have affected you.

Use the following scale to choose the **most appropriate number** for each situation:

0 = would **never** doze
1 = **slight chance** of dozing
2 = **moderate chance** of dozing
3 = **high chance** of dozing

It is important that you answer each question as best you can.

Situation	**Chance of Dozing (0-3)**
Sitting and reading	___
Watching TV	___

Εικόνα 7: Τμήμα του ερωτηματολογίου ESS

Αντιστοίχηση σε JSON:

```
{
"questionaire_definition": "ESS",
   "version": 1,
   "language": "en",
   "question_group": [
      {
         "title": "It is important that you answer each question as best you can",
         "questions": [
            {
               "number": 1,
               "question": "Sitting and reading",
               "type": "radiobutton",
               "choices": [
                  {
                     "point": 0,
                     "content": "would never doze "
                  },
                  {
                     "point": 1,
                     "content": "slight chance of dozing"
                  },
                  {
                     "point": 2,
                     "content": "moderate chance of dozing"
                  },
                  {
                     "point": 3,
                     "content": "high chance of dozing"
                  }
               ],
               "nullable": false,
               "condition": {
                  "hascondition": false,
                  "groupnumber": 0,
                  "questionumber": 0,
                  "qvalue": ""
               }
            },{
               "number": 2,
               "question": "Watching TV",
               "type": "radiobutton",
               "choices": [
                  {
                     "point": 0,
                     "content": "would never doze "
                  },
                  {
                     "point": 1,
                     "content": "slight chance of dozing"
                  },
```

```
                {
                  "point": 2,
                  "content": "moderate chance of dozing"
                },
                {
                  "point": 3,
                  "content": "high chance of dozing"
                }
              ],
              "nullable": false,
              "condition": {
                "hascondition": false,
                "groupnumber": 0,
                "questionumber": 0,
                "qvalue": ""
              }
            },…]
"text": "How likely are you to doze off or fall asleep in the following situations, in contrast to feelin
g just  tired?  This refers to your usual way of life in recent times.  Even if you haven't done some o
f these things recently try to work out how they would have affected  you. ",
  "scales": [
    {
      "name": "Sum",
      "type": "sum",
      "questions": [
        0,
        1,
        2,
        3,
        4,
        5,
        6,
        7
      ]
    }
"CreatorInformation": null,
  "CopyrightNotice": "Copyright M.W. Johns 1990-97",
  "CopyrightLicence": "All Rights Reserved"

}
```

2.4 Τεχνολογίες υλοποίησης εφαρμογής

Η εφαρμογή σε προγραμματιστικό περιβάλλον που κατασκευάστηκε για την παρούσα εργασία λειτουργεί με το μοντέλο πελάτη-διακομιστή. Το μοντέλο πελάτη διακομιστή είναι δομή κατανεμημένου συστήματος, στην οποία ο διακομιστής που παρέχει τις υπηρεσίες και ο πελάτης που τις ζητά τρέχουν ανεξάρτητα. Ο διακομιστής επικοινωνεί τοπικά με την βάση δεδομένων ,ενώ μέσα από διεπαφή προγραμματισμού εφαρμογών (Application Programming Interface) λαμβάνει και στέλνει κατά παραγγελία δεδομένα στους πελάτες. Η βάση δεδομένων κατασκευάστηκε με MariaDB σε Docker , ο διακομιστής σε ASP.NET (C#) , ενώ το κομμάτι του πελάτη (client-side) σε Blazor (C#) και Flutter(dart).

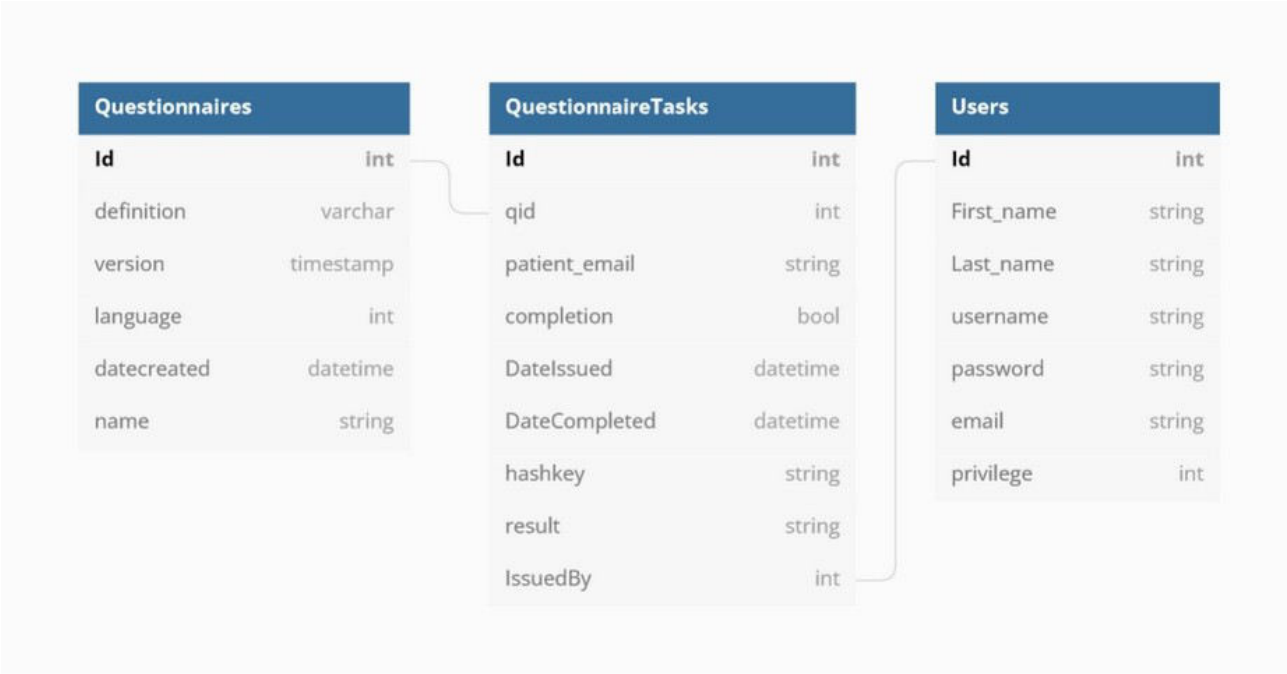
2.4.1 Η βάση δεδομένων

Η βάση δεδομένων δημιουργήθηκε πάνω σε MariaDB. Η MariaDB είναι ένα σύστημα διαχείρισης σχεσιακών βάσεων δεδομένων[12] το οποίο είναι βασισμένο στην MySQL. Η επιλογή του έγινε επειδή είναι δωρεάν και ανοικτού κώδικα , καθώς και επειδή προσφέρει μεγάλη συμβατότητα με πολλά διαθέσιμα API (Application Programming Interface).Στα πλαίσια της εργασίας δεν έγινε εγκατάσταση της βάσης, παρά χρησιμοποιήθηκε ένα έτοιμο docker container [13]το οποίο είναι ελεύθερα διαθέσιμο στην ιστοσελίδα της MariaDB. Για την αλληλεπίδραση της εφαρμογής με την βάση , το container εκτελέστηκε σε Docker[14], το οποίο είναι μια πλατφόρμα που μπορει να τρέχει εικόνες συστημάτων .

Η βάση δεδομένων περιέχει συνολικά τρεις πίνακες :

- Questionnaires (πίνακας ορισμών ερωτηματολογίου)
- QuestionnaireTasks (πίνακας διεργασιών ερωτηματολογίου)
- Users (πίνακας χρηστών)

Το σχήμα της βάσης παρουσιάζεται στην επόμενη εικόνα.(Εικόνα 8)



Εικόνα 8: Το σχήμα της βάσης δεδομένων

Πίνακας Questionnaires:

Ο πίνακας που περιέχει τον ορισμό κάθε ερωτηματολογίου. Ο ορισμός αυτός είναι αποθηκευμένος σε JSON μορφή στο πεδίο definition. Τα υπόλοιπα πεδία του πίνακα είναι μεταδεδομένα για το ερωτηματολόγιο .Συνολικά έχει τα πεδία:

- **Id**(int):Αύξων αριθμός κάθε καταχώρησης και κύριο κλειδί.
- definition(string): Περιέχει το JSON στο οποίο περιγράφεται το ερωτηματολόγιο,δημιουργείται αυτόματα από τη διεπαφή χρήστη σύμφωνα με τις επιλογές του τελευταίου.
- version(string): σειρά έκδοσης του ερωτηματολογίου, παρέχεται κατά τη δημιουργία του ερωτηματολογίου από τον χρήστη.
- language(string): γλώσσα γραφής του ερωτηματολογίου,παρέχεται κατά τη δημιουργία του ερωτηματολογίου από τον χρήστη.
- datecreated(DateTime): χρονική στιγμή δημιουργίας του ερωτηματολογίου,καταγράφεται από το back-end τη στιγμή δημιουργίας του ερωτηματολογίου.

- name(string): αναγνωριστικό του ερωτηματολογίου για εύκολη αναγνώριση από τον χρήστη

Πίνακας QuestionnaireTasks:

Ο πίνακας που περιέχει όλα τα δεδομένα που αφορούν κάθε μεμονωμένη διεργασία συμπλήρωσης ενός ερωτηματολογίου από έναν ασθενή. Περιέχει τα πεδία:

- **Id** (int) : Αύξων αριθμός κάθε καταχώρησης και κύριο κλειδί.
- *qid* (int) : Κλειδί πίνακα Questionnaires,δηλώνει ποιο ερωτηματολόγιο θα χρησιμοποιηθεί.
- patient_email (string) : Διεύθυνση ηλεκτρονικής αλληλογραφίας του ασθενή.
- completion (bool): Ένδειξη ολοκλήρωσης του ερωτηματολογίου από τον ασθενή.
- DateIssued (DateTime): χρονική στιγμή δημιουργίας της διεργασίας από τον χρήστη.
- DateCompleted (DateTime): Χρονική στιγμή ολοκλήρωσης του ερωτηματολογίου από τον ασθενή.
- hashkey (string): Κωδικός σύνδεσης του ασθενή με την διεπαφή, προκύπτει με hashing sha256[15] της χρονικής στιγμής που δημιουργήθηκε η διεργασία.
- result (string): JSON που καταγράφει τις απαντήσεις που έδωσε ο ασθενής κατά την συμπλήρωσή του
- *IssuedBy* (int): Κλειδί πίνακα Users. Το αναγνωριστικό του χρήστη που δημιούργησε την διεργασία.

Πίνακας Users:

Ο πίνακας στον οποίο καταγράφονται οι λ οι διαχειριστές (admins) και οι γιατροί (doctors).Περιέχει τα πεδία:

- **Id**(int):Αύξων αριθμός κάθε καταχώρησης και κύριο κλειδί.
- First_name(string): Όνομα του χρήστη
- Last_name(string):Επώνυμο του χρήστη
- username(string): όνομα χρήστη κατά τη σύνδεση στην εφαρμογή
- password(string):κωδικός χρήστη κατά τη σύνδεση στην εφαρμογή
  Είναι αποθηκευμένος με κρυπτογράφηση SHA256
- email(string): Διεύθυνση ηλεκτρονικής αλληλογραφίας του χρήστη
- privilege(int): Αναγνωριστικό προνομίων του χρήστη, 0 για διαχειριστή, 1 για γιατρό

2.4.2 Ο διακομιστής

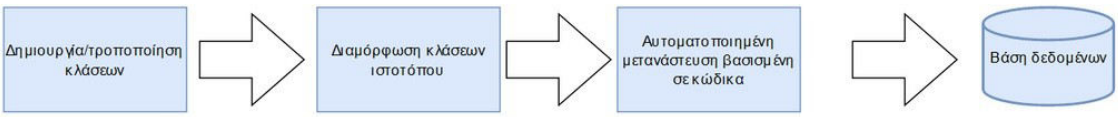
Ο διακομιστής (back-end) προγραμματίστηκε σε γλώσσα C# (ASP.NET), χρησιμοποιώντας το Entity Framework Core (.NET Core) για την διεπαφή με τη βάση δεδομένων. Η διαχείριση της βάσης δεδομένων γίνεται με το μοντέλο code-first. Το .NET Core είναι ένα σύνολο στοιχείων χρόνου εκτέλεσης, βιβλιοθήκης και μεταγλωττιστή που μπορούν να χρησιμοποιηθούν σε διάφορες ρυθμίσεις παραμέτρων για φόρτους εργασίας συσκευής και cloud. Το .NET Core, το οποίο υποστηρίζεται σε πολλές πλατφόρμες και είναι ανοιχτού κώδικα, παρέχει ένα μοντέλο ελαφριάς ανάπτυξης και την ευελιξία χρήσης διαφόρων πλατφορμών λειτουργικών συστημάτων για εργαλεία ανάπτυξης.

Το Entity Framework Core[16] παρέχει την κλάση DbContext μέσω των μεθόδων της οποίας γίνεται η ανταλλαγή δεδομένων μεταξύ βάσης και διακομιστή. Ο προγραμματιστής δεν χρειάζεται να γράψει sql ερωτήματα για να λάβει ή να παρέχει δεδομένα. Τα ερωτήματα γράφονται αυτόματα , αναλόγως των παραμέτρων που δίνονται στις μεθόδους. Για παράδειγμα δίνεται ο τρόπος που γίνεται εκτέλεση ενός ερωτήματος SELECT για την επιλογή ενός συγκεκριμένου ερωτηματολογίου(Πίνακας 1):

SQL	Entity Framework Core
SELECT * FROM Questionnaires WHERE Id=i	one = new QDbContext (); one.Questionnaires.Find(i);

Πίνακας 1: Ερωτήματα SQL σε EF Core

Στο μοντέλο code-first διαχείρισης της βάσης δεδομένων ,δουλειά του προγραμματιστή είναι μόνο να δημιουργεί και να τροποποιεί-διαμορφώνει τις κλάσεις που περιγράφουν τα αντικείμενα που θέλει να αποθηκευτούν στη βάση δεδομένων . Έπειτα δίνει ο προγραμματιστής την εντολή της δημιουργίας μετανάστευσης(add-migration) . η οποία περιέχει όλες τις μεταβολές στο μοντέλο της βάσης δεδομένων . Τέλος με την εντολή ενημέρωσης της βάσης (update-database) , το framework εκτελεί τις απαραίτητες εντολές , ώστε να λάβει η βάση τη νέα της μορφή (Εικόνα 9).



Εικόνα 9: Το μοντέλο code-first

Με αυτόν τον τρόπο ο προγραμματιστής δεν χρειάζεται να τροποποιεί μόνος του τις παραμέτρους της βάσης δεδομένων .Επικεντρώνεται στην ανάπτυξη της εφαρμογής μόνο στο κομμάτι των κλάσεων του προγράμματος, κάνοντας την ανάπτυξη της εφαρμογής ταχύτερη και πιο ευέλικτη.

2.4.2.1 Το API του διακομιστή

Για την επικοινωνία διακομιστή-πελάτη χρησιμοποιούνται οι μέθοδοι GET και POST μέσα στο πρωτόκολλο HTTP. Ανάλογα με τη διεύθυνση και τη μέθοδο που καλείται, εκτελεί ο διακομιστής την ανάλογη λειτουργία.

Ακολουθεί συνοπτική λίστα με τις διευθύνσεις GET και POST που προσφέρονται από το διακομιστή μαζί με μια επεξήγηση της λειτουργικότητάς τους. Η υλοποίηση σε κώδικα δίνεται στο παράρτημα Β.

GET

- /questionnaire/id={}
  Επιστρέφει το JSON του ερωτηματολογίου βάσει κυρίου κλειδιού της βάσης.
- /list
  Επιστρέφει τη πλήρη λίστα με όλα τα μετα-δεδομένα των ερωτηματολογίων της βάσης
- /users
  Επιστρέφει τη πλήρη λίστα των χρηστών της βάσης
- "/patient/id={id}"
  Επιστρέφει το ερωτηματολόγιο που αντιστοιχεί στον κωδικό σύνδεσης ασθενή (patient token)
- /tasks/id={id}
  Επιστρέφει τις διεργασίες για τις οποίες διαθέτει δικαιώματα ο χρήστης(διαχειριστής ή γιατρός)

POST

- /newuser
  Προσθέτει νέο χρήστη σύμφωνα με τα δεδομένα που λαμβάνει σε JSON
- /newpatient
  Προσθέτει νέα διεργασία ερωτηματολογίου βάσει δεδομένων που λαμβάνει σε JSON
- /task_removal
  Αφαιρεί διεργασία/ες από την βάση δεδομένων βάσει εισόδου
- /user_removal
  Αφαιρεί χρήστη(ες) απο την βάση δεδομένων βάσει εισόδου.
- /posting
  Αποθηκεύει νέο ερωτηματολόγιο βάσει του JSON που έρχεται από τη διεπαφή χρήστη στη βάση δεδομένων
- /response
  Λαμβάνει την απάντηση του ασθενή, αξιολογεί τις απαντήσεις βάσει των κλιμάκων, ενημερώνει τον πίνακα διεργασιών με την απάντηση του ασθενή.

- /login
  Επαφή ταυτοποίησης χρηστών,ταυτοποιεί τα στοιχεία σύνδεσης στον πίνακα χρηστών.

2.4.2.2Λειτουργίες Διακομιστή

Πέρα από μεσάζων μεταξύ διεπαφής χρήστη και βάσης δεδομένων, ο διακομιστής έχει και το ρόλο βαθμολογητη-ελεγκτή εγκυρότητας των απαντήσεων σε ένα ερωτηματολόγιο.

Ο διακομιστής τρέχει την συνάρτηση εγκυρότητας ξανά, μιας και εχει τρέξει και στην διεπαφή χρήστη, με σκοπό την αποφυγή κακόβουλων επιθέσεων και συυπληρώσεων ερωτηματολογίου. Έπειτα , καλεί την συνάρτηση της βαθμολογίας, ώστε να υπολογίσει τα διάφορα αποτελέσματα για κάθε κλίμακα του ερωτηματολογίου που χρησιμοποιήθηκε και ύστερα, μαζί με τις απαντήσεις, ενημερώνει την κατάσταση στον πίνακα questionnairetasks ,καθώς και το πεδίο result.

Κεφάλαιο 3 : Διεπαφές χρηστών

3.1 Διάκριση χρηστών

Οι χρήστες της εφαρμογής έχουν χωριστεί σε τρεις κατηγορίες:

- Διαχειριστές

- Ιατρούς
- Ασθενείς

Στην περίπτωση που ο χρήστης είναι διαχειριστής της πλατφόρμας, έχει πλήρη δικαιώματα ως προς όλες τις λειτουργίες της.Του δίνεται η δυνατότητα να σχεδιάσει ένα νέο ερωτηματολόγιο, να δημιουργήσει και να διαχειριστεί διεργασίες απάντησης ερωτηματολογίου και να δημιουργήσει και να διαχειριστεί τους χρήστες της πλατφόρμας, διαχειριστές και γιατρους,που θα έχουν με τη σειρά τους πρόσβαση στην πλατφόρμα .

Στην περίπτωση που ο χρήστης είναι ιατρός, διαθέτει την ίδια ικανότητα δημιουργίας ερωτηματολογίου και ανάθεσης διεργασίας συμπλήρωσής του σε κάποιον ασθενή, όμως έχει πρόσβαση μόνο στις διεργασίες απάντησης ερωτηματολογίου που ο ίδιος δημιούργησε,σε αντίθεση με τους διαχειριστές που έχουν πρόσβαση σε όλες τις διεργασίες, ανεξαρτήτως του χρήστη που τις δημιούργησε.Επίσης , δεν μπορεί να προσθαφαιρέσει διαχειριστές ή γιατρούς.

Τέλος, στην περίπτωση που ο χρήστης είναι ασθενής, η μόνη αλληλεπίδραση που έχει με την πλατφόρμα είναι η συμπλήρωση του ερωτηματολογίου και η υποβολή του στο σύστημα.

Οι λειτουργίες για διαχειριστή και γιατρο έχουν υλοποιηθεί μόνο σε web(blazor) πλατφόρμα, επειδή έγινε η υπόθεση ότι οι εργασίες αυτές λαμβάνουν χώρα σε περιβάλλον γραφείου. Αντιθέτως, για τους ασθενείς εχουν δημιουργηθεί διεπαφές σε 2 πλατφόρμες, η πρώτη είναι ενσωματωμένη στην πλατφόρμα web που προαναφέρθηκε, ενώ η δεύτερη έχει γίνει σε mobile ready υλοποίηση χρησιμοποιώντας Flutter(dart)[17] .

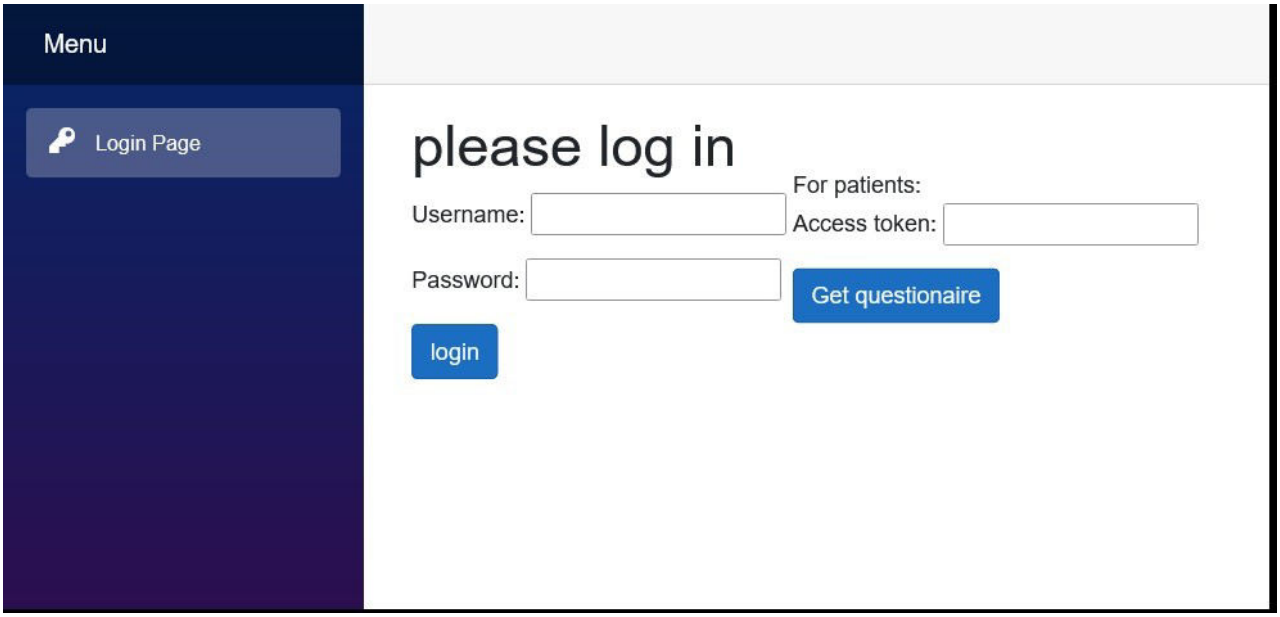
3.2 Η διεπαφή χρήστη σε Blazor

Συνολικά , η διεπαφή χρήστη σε Blazor διαθέτει 6 διαφορετικές σελίδες, η κάθε μία με την δική της χρήση

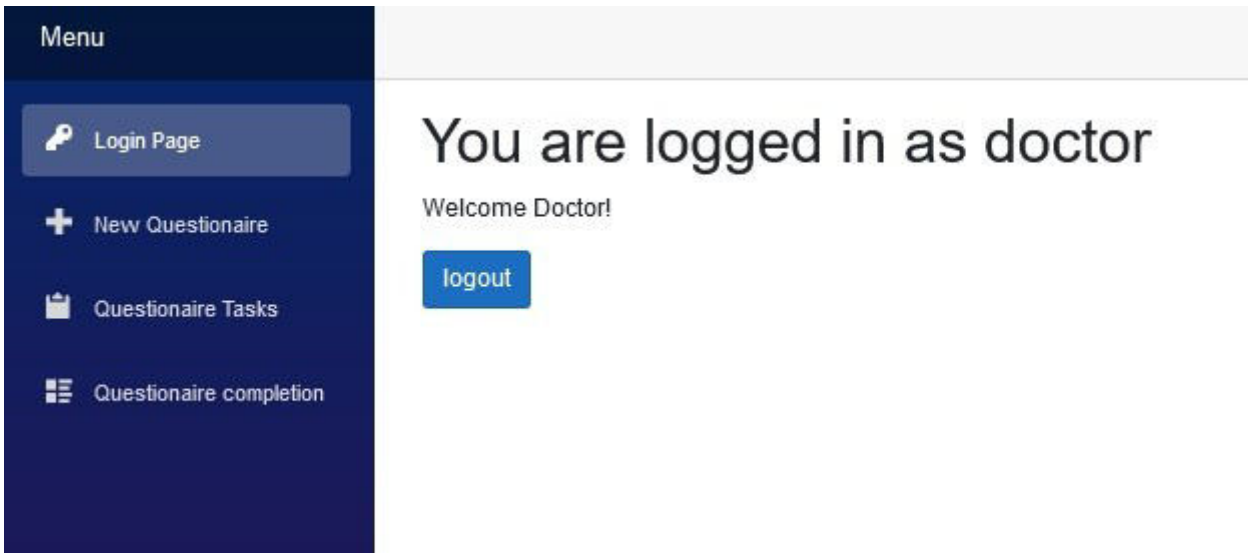
1. Σελίδα σύνδεσης χρήστη.
2. Διαχείριση χρηστών
3. Δημιουργία ερωτηματολογίου
4. Διαχείριση διεργασιών ερωτηματολογίου
5. Προβολή ερωτηματολογίου
6. Συμπλήρωση ερωτηματολογίου.

3.2.1 Σελίδα σύνδεσης χρήστη

Στην σελίδα σύνδεσης (Εικόνα 10) δίνονται υπάρχουν 2 τρόποι σύνδεσης, για διαχειριστές ή γιατρούς και για ασθενείς. Στην 1η περίπτωση η σύνδεση γίνεται με το όνομα χρήστη(username) και κωδικό πρόσβασης(password) . Μετά την επικύρωσή τους από τον διακομιστή αλλάζει η κατάσταση της σελίδας σε συνδεδεμένου χρήστη (Εικόνα 11) και μπορούν να χρησιμοποιήσουν τις διάφορες λειτουργίες της εφαρμογής αναλόγως της ιδιότητάς τους.Οι ασθενείς χρειάζεται απλά να υποβάλλουν τον κωδικό(access token) που είναι μοναδικά ταυτισμένος στην βάση δεδομένων με την διεργασία συμπλήρωσης ερωτηματολογίου, για να οδηγηθούν στην σελίδα εκτέλεσής της,



Εικόνα 10: Αρχική σελίδα σύνδεσης

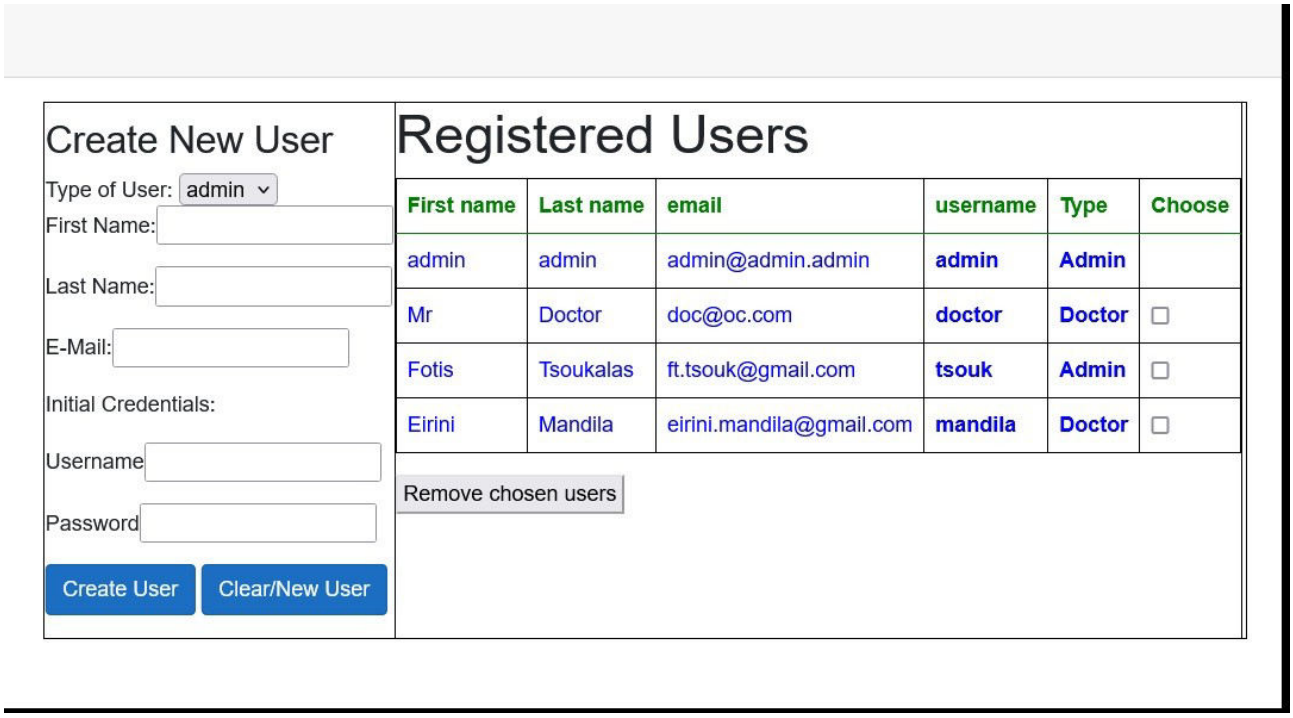


Εικόνα 11: Αρχική σελίδα έπειτα από σύνδεση γιατρού

3.2.2 Διαχείριση χρηστών

Η σελίδα διαχείρισης χρηστών (Εικόνα 12) , η οποία είναι προσβάσιμη μόνο στους διαχειριστές , επιτρέπει την δημιουργία, ανασκόπηση και διαγραφή των διαχειριστών και των ιατρών.Διαθέτει 2 μέρη, το κομμάτι που επιτρέπει την εγγραφή ενός νέου χρήστη, με πεδία κειμένου τα ακόλουθα:

1. Όνομα (name)
2. Επώνυμο(surname)
3. Διεύθυνση Ηλεκτρονικού Ταχυδρομείου (email)
4. Όνομα χρήστη (username)
5. Κωδικός πρόσβασης (password)



Εικόνα 12: Σελίδα διαχείρισης χρηστών

Ακολουθούν το κουμπί υποβολής (create user),με το πάτημα του οποίου γίνεται η υποβολή των στοιχείων στο σύστημα και το κουμπί διαγραφής (clear) που αδειάζει οτι εχει συμπληρώσει ο χρήστης στα πεδία.Κατα την υποβολή εμφανίζεται μήνυμα επιβεβαίωσης αν είναι επιτυχής και μήνυμα σφάλματος αν έχει γίνει λάθος. Σε όλα τα πεδία γίνεται έλεγχος κατά την υποβολή στο εμπρός μέρος της εφαρμογής ως προς αν έχει μείνει κενό κάποιο πεδίο και επιστρέφει μηνυμα τότε. Στο πίσω μέρος γίνεται ο έλεγχος ως προς την μη σύνδεση του e-mail που δόθηκε με κάποια προηγούμενη εγγραφή χρήστη , καθώς και για τη μη προηγούμενη χρήση του ιδίου ονόματος χρήστη. Και στις δύο περιπτώσεις προβάλλεται αντίστοιχο μήνυμα στον χρήστη αν υπάρχει πρόβλημα.

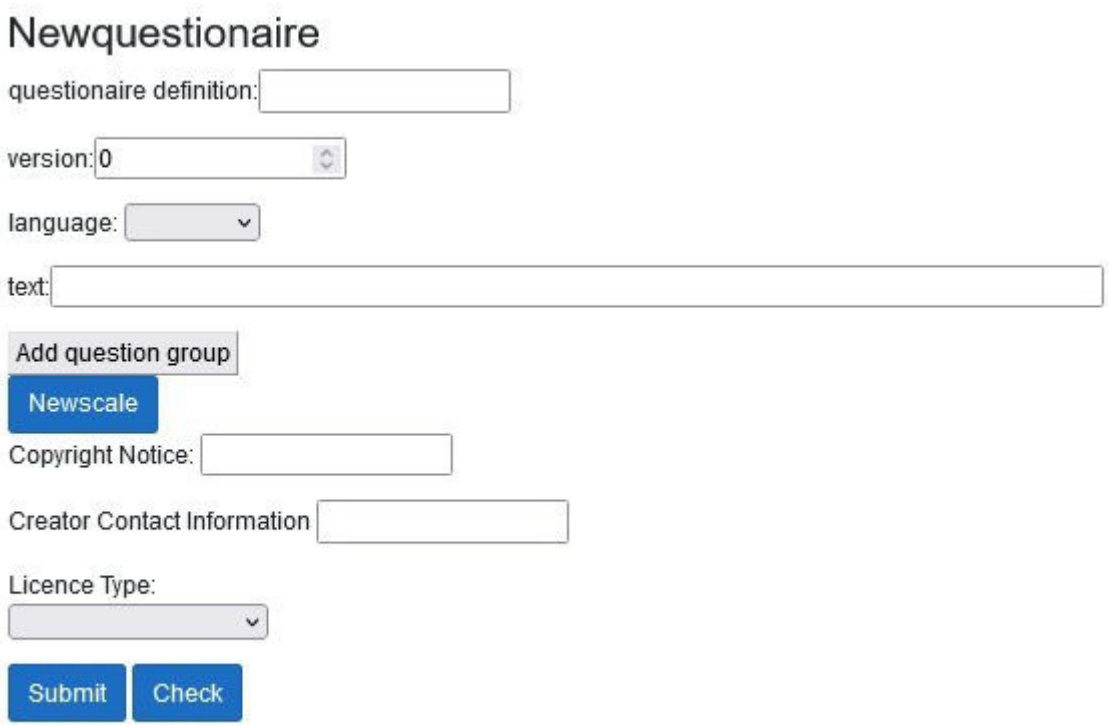
Στα δεξιά της σελίδας υπάρχει πίνακας με τους υπάρχοντες χρήστες και τα στοιχεία τους, πλην του κωδικού πρόσβασής τους.Ο κάθε διαχειριστής μπορεί να επιλέξει χρήστες και να τους διαγράψει με το πάτημα του κουμπιού “remove chosen users” .Κατά την δημιουργία χρήστη ή την διαγραφή χρήστη/ων ενημερώνεται αυτόματα η λίστα με τους υπάρχοντες χρήστες. Ο κωδικός που δίνεται μετά αποστέλλεται. Για την είσοδο των χρηστών, το frontend κάθε φορά κρυπτογραφεί

με SHA256 τον κωδικό πρόσβασης, πριν τον στείλει στο διακομιστή για ταυτοποίηση.

3.2.3 Δημιουργία ερωτηματολογίου

Η βασική λειτουργία της εφαρμογής είναι η μεταφορά του ερωτηματολογίου από μη δομημένη μορφή στην προκαθορισμένη μορφή JSON που προτείνεται στην παρούσα εργασία. Η μετατροπή αυτή γίνεται στην σελίδα που θα περιγραφεί εδώ.

Η σελίδα αυτή αποτελείται από μία διαδραστική φόρμα (Εικόνα 13) με κάποια προϋπάρχοντα πεδία και κάποια που δημιουργούνται κατά βούληση του χρήστη.



Εικόνα 13: Η αρχική μορφή της σελίδας δημιουργίας ερωτηματολογίου

Τα προυπάρχοντα πεδία που είναι υποχρεωτικό να συμπληρωθούν σε κάθε δημιουργία ερωτηματολογίου είναι:

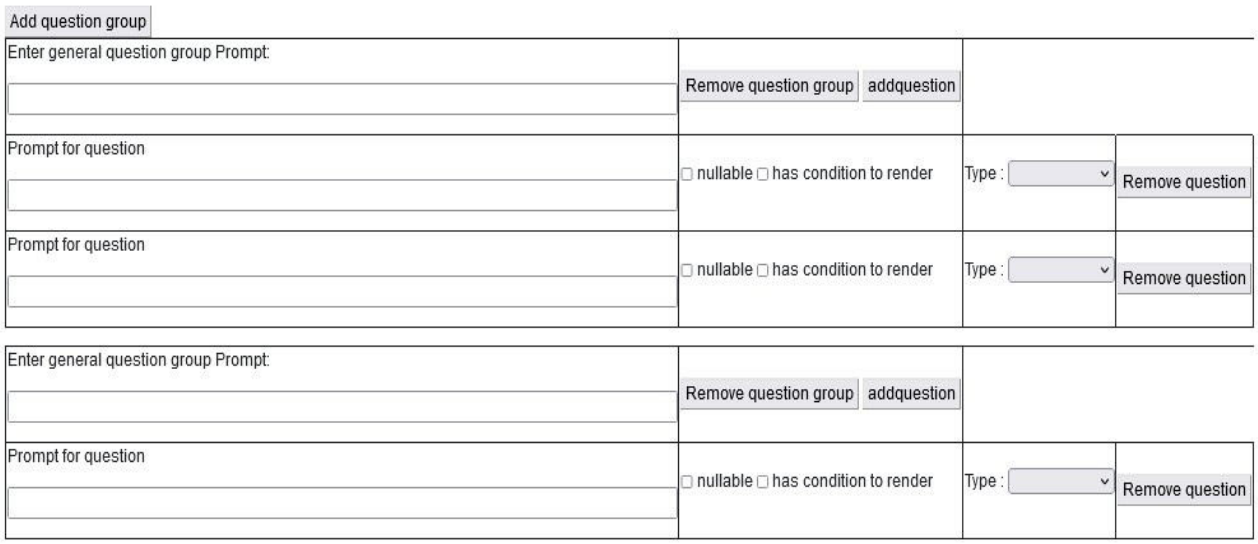
1. Το όνομα του ερωτηματολογίου (questionnaire definition), το οποίο δίνει όσο πιο σύντομα γίνεται την ονομασία του ερωτηματολογίου. Υπάρχει όριο χαρακτήρων (30) ως προς την είσοδο .
2. Η έκδοση του ερωτηματολογίου (version), που δίνεται με αριθμητική είσοδο. Η εφαρμογή δεν δέχεται τιμή 0.
3. Η γλώσσα (language) του ερωτηματολογίου. Η εφαρμογή παρουσιάζει αναπτυσσόμενο μενού με τις υπάρχουσες γλώσσες στην Αγγλική. Κάθε επιλογή οδηγεί σε μεταφορά της κωδικοποίησης κατά ISO 639-I [18]στο JSON.

Επιπλέον προσφέρεται ένα πεδίο κειμένου (text), το οποίο δεν είναι απαραίτητο να συμπληρωθεί ,για την παροχή του εισαγωγικού κειμένου του ερωτηματολογίου αν υπάρχει, καθώς και των γενικών οδηγιών που δίνονται και αφορούν ολόκληρο το ερωτηματολόγιο.

Στο τέλος της φόρμας υπάρχουν άλλα 3 πεδία παροχής δεδομένων, τα οποία είναι προαιρετικά . Το πρώτο αφορά την παροχή στοιχείων επικοινωνίας του δημιουργού του ερωτηματολογίου και είναι κειμένου ανοιχτού τύπου ενώ ακολουθούν πληροφορίες για τα πνευματικά δικαιώματα, σε πρώτο λόγο με το πεδίο γραφής της δήλωσης πνευματικών δικαιωμάτων και σε δεύτερο λόγο με αναπτυσσόμενο μενού με τις διαφορετικούς τύπους αδειών,π.χ. All rights reserved, creative commons,Free to Use κλπ. .

Ενδιάμεσα από τα δύο σταθερά τμήματα της φόρμας παρεμβάλλεται το διαμορφώσιμο της κομμάτι, το οποίο αφορά τις ερωτήσεις (Εικόνα 14) .Το κομμάτι αυτό ακολουθεί την ίδια λογική που έχει περιγραφεί στη δομή του JSON. Δηλαδή έχει ως βασικό χαρακτηριστικό τις ομάδες ερωτήσεων, οι οποίες αυξάνονται και αφαιρούνται με το πάτημα ενός κουμπιού (create question group και remove question group αντίστοιχα). Για κάθε ομάδα ερωτήσεων προσφέρεται ένα προαιρετικό πεδίο εισαγωγής κειμένου(title) για την παροχή ενος τίτλου ή/και των ειδικών οδηγιών που

αφορούν μόνο τις ερωτήσεις της συγκεκριμένης ομάδας . Το ίδιο πεδίο μπορεί να χρησιμοποιηθεί και για την παροχή ενδιάμεσων οδηγιών[1] ή κειμένου για το ερωτηματολόγιο, αν δεν προστεθουν ερωτήσεις στην ομάδα ερωτήσεων,αν μείνει δηλαδή κενό το πεδίο των ερωτήσεων στην ομάδα. Δεν έχει κατασκευαστεί με πρωταρχικά αυτη τη λειτουργία, και αντιβαίνει μερικώς την λογική κατασκευής του JSON, όμως θεωρήθηκε κατα την συγγραφή του παρόντος ότι η ευελιξία που παρέχεται με αυτόν τον τρόπο οδηγεί στην επιτυχέστερη και ακριβέστερη τυποποίηση περισσότερων ερωτηματολογίων



Εικόνα 14: Διαχείριση ερωτήσεων και ομάδων ερωτήσεων

Κάθε ομάδα ερωτήσεων που δημιουργείται διαθέτει το δικό της κουμπί προσθήκης ερώτησης στην ομάδα.Κάθε φορά που πατιέται αυτό το κουμπί εμφανίζονται σε μορφή γραμμής σε πίνακα τα πεδία που αφορούν την καινούρια ερώτηση.

Αυτά είναι :

1. Το πεδίο κειμένου για την ερώτηση (Prompt) με όριο χαρακτήρων(400) και υποχρεωτικότητα συμπλήρωσης

2. 2 κουμπιά επιλογής , που καθορίζουν αν επιτρέπεται αυτή η ερώτηση να μείνει κενή (nullable) και αν η εμφάνισή της εξαρτάται από την απάντηση του ασθενή σε κάποια άλλη ερώτηση(has condition to render). Με την επιλογή του τελευταίου εμφανίζεται ένα διπλό , αναπτυσσόμενο μενού που πρέπει ο συντάκτης να επιλέξει την ερώτηση που ελέγχει την προβολή της παρούσας.
    - Αν η ερώτηση αυτή είναι αριθμητική ανοικτού τύπου, του δίνεται η επιλογή αριθμού με πεδίο ανοικτού τύπου, ο οποίος ελέγχει το όριο για την γενική σχέση που εμφανίζεται σε αναπτυσσόμενο μενου. Η σχέση αυτή μπορεί να είναι μεγαλύτερη/μικρότερη ή και ίση από τον αριθμό(5 σχέσεις).
    - Αν η ερώτηση αυτή είναι κλειστού τύπου πολλαπλής επιλογής , εμφανίζονται δυναμικά με κουμπιά μοναδικής επιλογής οι διάφορες επιλογές της ερώτησης και ο χρήστης πρέπει να διαλέξει μία από αυτές.
3. Ενα κουμπί που αφορά τον τύπο απάντησης της ερώτησης. Εχει υποτεθεί από την κατασκευή του προτύπου JSON 4 διαφορετικοί τύποι, οι οποίοι εμφανίζονται σε αναπτυσσόμενο μενου :
    - Text
    - Radiobutton
    - Multi
    - Number
4. Ενα κουμπί αφαίρεσης ερώτησης (remove question)

Στο τέλος της φόρμας υπάρχει κουμπί υποβολής (submit) του ερωτηματολογίου στο σύστημα. Αν η φόρμα είναι έγκυρη , εμφανίζεται μήνυμα επιβεβαίωσης (Submitted) κάτω από αυτό, αν δεν είναι έγκυρη η συμπλήρωση της φόρμας εμφανίζεται μήνυμα προβλήματος .

3.2.4 Διαχείριση διεργασιών ερωτηματολογίου

Οι διεργασίες ερωτηματολογίου είναι εγγραφές που κάθε μία αφορά έναν ασθενή και ένα ερωτηματολόγιο. Μπορούν να δημιουργηθούν εξίσου από διαχειριστή ή γιατρό, όμως ο πρώτος έχει πρόσβαση σε όλες τις διεργασίες που υπάρχουν στο σύστημα,

ενώ ο γιατρός μόνο στις διεργασίες που έχει δημιουργήσει. Η διάκριση αυτή γίνεται όταν ζητείται η λίστα με τις διεργασίες από τον διακομιστή.

Εικόνα 15: Σελίδα διαχείρισης διεργασιών ερωτηματολογίου

Η σελίδα (Εικόνα 15) που δημιουργεί μία διεργασία αποτελείται από 2 κύρια μέρη , τον δημιουργό διεργασίας και τον πίνακα που αναρτάται η κατάσταση κάθε διεργασίας. Για την δημιουργία της διεργασίας χρειάζεται μόνο η επιλογή του ερωτηματολογίου από αναπτυσσόμενο μενού με όλα τα διαθέσιμα ερωτηματολόγια και η εισαγωγή e-mail ασθενή. Επειδή η συμπλήρωση του ονόματος έχει αφεθεί στον ασθενή , χρησιμοποιείται ένα πιο ανώνυμο αναγνωριστικό , όπως η διεύθυνση ηλεκτρονικού ταχυδρομείου.

Στον πίνακα με τις υπάρχουσες διεργασίες αναρτάται για κάθε μία : η διεύθυνση ηλεκτρονικού ταχυδρομείου του ασθενή, το όνομα χρήστη του δημιουργού της διεργασίας (αν χρησιμοποιείται από διαχειριστή, οι γιατροί βλέπουν μόνο τις δικές τους), η ώρα δημιουργίας, η κατάσταση ολοκλήρωσης, η ώρα ολοκλήρωσης αν υπάρχει, η απάντηση του ασθενούς και επιπρόσθετα δίνεται η ικανότητα επιλογής και μετέπειτα διαγραφής των επιλεγμένων με το κουμπί Remove chosen tasks.

3.2.5 Προβολή ερωτηματολογίου

Στην σελίδα προεπισκόπησης ερωτηματολογίου μπορούν οι διαχειριστές και οι γιατροί να κάνουν προεπισκόπηση κάθε ερωτηματολογίου που βρίσκεται διαθέσιμο από τον διακομιστή. Περιέχει ένα αναπτυσσόμενο μενού επιλογής του χαρακτηρισμού του επιθυμητού ερωτηματολογίου και το κουμπί με το οποίο δίνεται η εντολή λήψης και προεπισκόπησής του. Ο χρήστης μπορεί να συμπληρώσει το ερωτηματολόγιο και να κάνει δοκιμή υποβολής , ώστε να δοκιμάσει τη λειτουργικότητά του . Το αποτέλεσμα δεν αποθηκεύεται κάπου.

Η συγκεκριμένη σελίδα έχει πιο πολύ ως σκοπό τον έλεγχο του αποτελέσματος της δημιουργίας ερωτηματολογίου από τον χρήστη , αποφεύγοντας την δημιουργία διεργασίας ερωτηματολογίου και την αποσύνδεση/επανασύνδεση ως ασθενή με τον κωδικό της διεργασίας. Η προεπισκόπηση χρησιμοποιεί το ίδιο εξάρτημα που προβάλλει στους ασθενείς το ερωτηματολόγιο, συνεπώς προσφέρει απολύτως ίδια εμπειρία με αυτή που θα λάβει ο ασθενής.

3.2.6 Σελίδα απάντησης ερωτηματολογίου

Η σελίδα απάντησης του ερωτηματολογίου (Εικόνα 16) είναι προσβάσιμη από τον ασθενή κατά την σύνδεσή του με τον μοναδικά αντιστοιχισμένο κωδικό (patient token).Αν ο ασθενής έχει απαντήσει το ερωτηματολόγιο προηγουμένως ή έβαλε λάθος κωδικό, η σελίδα αυτή απλά εμφανίζει κείμενο που δηλώνει ότι ο κωδικός δεν αντιστοιχεί σε κάποιο ερωτηματολόγιο ή ότι το ερωτηματολόγιο στο οποίο αντιστοιχεί έχει ήδη απαντηθεί.(This questionnaire task has already been completed or is not available).Ο ασθενής μπορεί τότε ή να πατήσει ένα κουμπί που τον επιστρέφει στην αρχική σελίδα ή να ανανεώσει τη σελίδα, το οποίο θα μηδενίσει την κατάστασή της και θα τον επιστρέψει στην αρχική.

Αν η διεργασία συμπλήρωσης δεν έχει ολοκληρωθεί προηγουμένως, αντλείται το ερωτηματολόγιο σε μορφή JSON από τον διακομιστή και προβάλλεται το περιεχόμενό του. Με την σειρά, εκτυπώνεται το γενικό κείμενο (text) και ακολουθούν οι ομάδες ερωτήσεων με τη σειρά που βρίσκονται στο JSON. Για κάθε ομάδα πρώτα τυπώνεται το συνοδευτικό κείμενο(title στο JSON) , και μετά σειριακά κάθε ερώτηση της ομάδας με το πεδίο απάντησης. Με την εκτύπωση και της τελευταίας ερώτησης μίας ομάδας ακολουθούν το κείμενο και οι ερωτήσεις της επόμενης. Ως προς τις ερωτήσεις, για κάθε ζητούμενο τυπώνονται στην πρώτη σειρά ο αριθμός της ερώτησης μαζί με την ερώτηση, ενώ ακολουθεί το πεδίο απάντησης στην επόμενη σειρά, το οποίο εξαρτάται από τον τύπο (type) της ερώτησης.

Please enter your date of birth: 06 / 28 / 2022

Enter your name :

How likely are you to doze off or fall asleep in the following situations, in contrast to feeling just tired? This refers to your usual way of life in recent times. Even if you haven't done some of these things recently try to work out how they would have affected you.

It is important that you answer each question as best you can

1 . Sitting and reading

○ would never doze○ slight chance of dozing○ moderate chance of dozing○ high chance of dozing

2 . Watching TV

○ would never doze○ slight chance of dozing○ moderate chance of dozing○ high chance of dozing

3 . Sitting, inactive in a public place (e.g. a theatre or a meeting)

○ would never doze○ slight chance of dozing○ moderate chance of dozing○ high chance of dozing

4 . As a passenger in a car for an hour without a break

○ would never doze○ slight chance of dozing○ moderate chance of dozing○ high chance of dozing

5 . Lying down to rest in the afternoon when circumstances permit

○ would never doze○ slight chance of dozing○ moderate chance of dozing○ high chance of dozing

6 . Sitting and talking to someone

○ would never doze○ slight chance of dozing○ moderate chance of dozing○ high chance of dozing

7 . Sitting quietly after a lunch without alcohol

○ would never doze○ slight chance of dozing○ moderate chance of dozing○ high chance of dozing

8 . In a car, while stopped for a few minutes in the traffic

○ would never doze○ slight chance of dozing○ moderate chance of dozing○ high chance of dozing

Submit Clear

Εικόνα 16: Σελίδα απάντησης ερωτηματολογίου ESS

Αν είναι μοναδικής επιλογής (radiobutton), εμφανίζονται οι διάφορες επιλογές με κομβία και με το πάτημα καθενός επιλέγεται μόνο αυτό.

Αν είναι πολλαπλής επιλογής (multi),εμφανίζονται οι διάφορες επιλογές με κομβία και ο ασθενής μπορεί να επιλέξει 1 ή και περισσότερα.

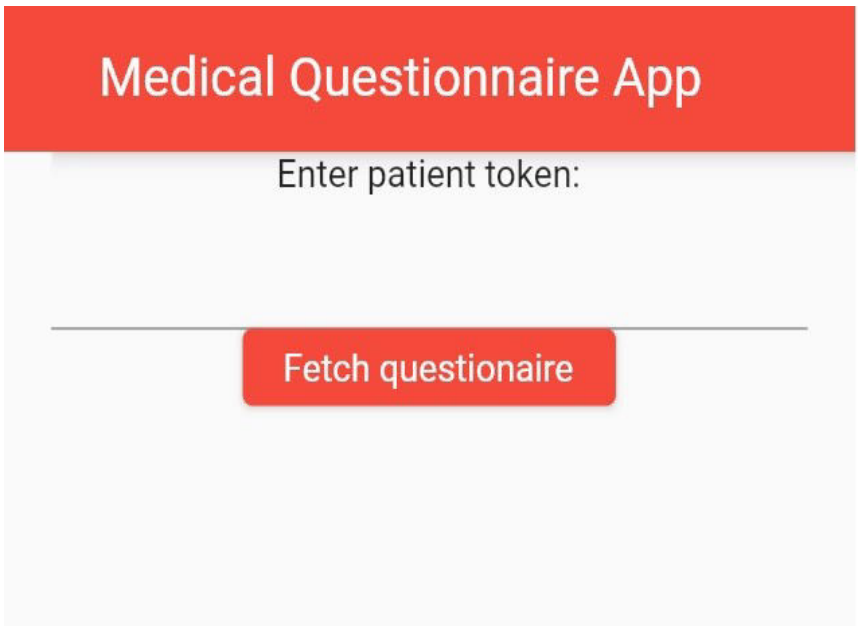
Όσες ερωτήσεις έχουν προϋπόθεση εμφάνισης δεν παρουσιάζονται εξαρχής,παρά μόνο αν ο ασθενής επιλέξει απάντηση που τις εμφανίζει. Αν τότε αλλάξει την επιλογή του , αυτές οι ερωτήσεις εξαφανίζονται και στον υπολογισμό της βαθμολογίας του ερωτηματολογίου δεν λαμβάνονται υπόψιν.

Μετά από όλες τις ερωτήσεις ακολουθούν 2 κουμπιά:

1. Κουμπί υποβολής (submit),με το πάτημα του οποίου γίνεται ένας πρωταρχικός έλεγχος των απαντήσεων και αν είναι έγκυρο υποβάλλεται στο σύστημα. Με το πάτημά του ανεξαρτήτως αποτελέσματος προβάλλονται από κάτω οι απαντήσεις του ασθενή στις ερωτήσεις. Αν δεν έχει συμπληρώσει κάποια υποχρεωτική ερώτηση, προηγείται των απαντήσεων μήνυμα που γνωστοποιεί ποιές ερωτήσεις δεν έχει απαντήσει.
2. Κουμπί μηδενισμού (Clear), με το πάτημα του οποίου εξαφανίζονται όλες οι απαντήσεις που έχει δώσει ο ασθενής στη φόρμα. Δεν επηρεάζει προηγούμενη υποβολή.

3.3 Η διεπαφή χρήστη σε Flutter

Σε flutter υλοποιήθηκε μόνο το κομμάτι της εφαρμογής που αφορά την συμπλήρωση του ερωτηματολογίου και διαθέτει ακριβώς την ίδια λογική με την υλοποίηση σε Blazor.
Η αρχική σελίδα (Εικόνα 17)διαθέτει ένα κενό πεδίο στο οποίο ο ασθενής καλείται να βάλει τον κωδικό διεργασίας ερωτηματολογίου (patient token) και έπειτα να πατήσει το κουμπί υποβολής (Submit).



Εικόνα 17: Αρχική σελίδα διεπαφής χρήστη σε Flutter

Αν ο κωδικός είναι έγκυρος, το ερωτηματολόγιο θα εμφανιστεί ακριβώς με τις ίδιες ιδιότητες και χαρακτηριστικά που περιγράφηκαν για τη σελίδα σε Blazor. Δηλαδή,με την σειρά, εκτυπώνεται το γενικό κείμενο (text) και ακολουθούν οι ομάδες ερωτήσεων με τη σειρά που βρίσκονται στο JSON.Για κάθε ομάδα πρώτα τυπώνεται το συνοδευτικό κείμενο(title στο JSON) , και μετά σειριακά κάθε ερώτηση της ομάδας με το πεδίο απάντησης. Με την εκτύπωση και της τελευταίας ερώτησης μίας ομάδας ακολουθούν το κείμενο και οι ερωτήσεις της επόμενης. Ως προς τις ερωτήσεις, για κάθε ζητούμενο τυπώνονται στην πρώτη σειρά ο αριθμός της ερώτησης μαζί με την ερώτηση, ενώ ακολουθεί το πεδίο απάντησης στην επόμενη σειρά, το οποίο εξαρτάται από τον τύπο (type) της ερώτησης.

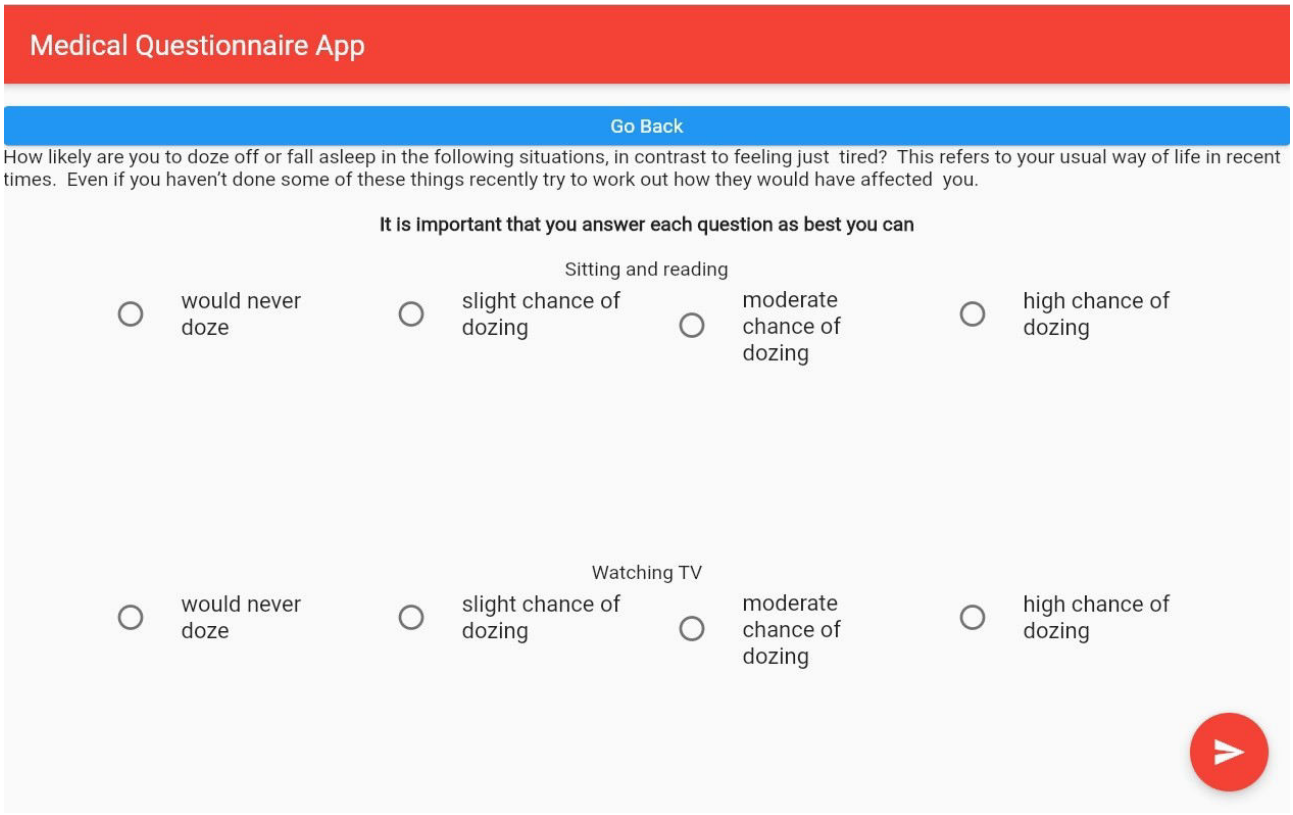
Αν είναι ανοικτού τύπου (text), δίνεται κενό πεδίο κειμένου.

Αν είναι αριθμητική ερώτηση ανοιχτού τύπου, προβάλλεται αριθμητικό πεδίο, το οποίο στα πλαίσια της υλοποίησης δέχεται μόνο ακεραίους αριθμούς. Στο flutter, πέρα από τον έλεγχο ειδόδου που γίνεται , στην mobile εκδοχή εμφανίζεται και αμιγώς αριθμητικό πληκτρολόγιο.

Αν είναι μοναδικής επιλογής (radiobutton), εμφανίζονται οι διάφορες επιλογές με κομβία και με το πάτημα καθενός επιλέγεται μονο αυτό.

Αν είναι πολλαπλής επιλογής (multi),εμφανίζονται οι διάφορες επιλογές με κομβία και ο ασθενής μπορεί να επιλέξει 1 ή και περισσότερα.

Όσες ερωτήσεις έχουν προϋπόθεση εμφάνισης δεν παρουσιάζονται εξαρχής,παρα μόνο αν ο ασθενής επιλέξει απάντηση που τις εμφανίζει. Αν τότε αλλάξει την επιλογή του , αυτές οι ερωτήσεις εξαφανίζονται και στον υπολογισμό της βαθμολογίας του ερωτηματολογίου δεν λαμβάνονται υπόψιν.



Εικόνα 18: Σελίδα απάντησης ερωτηματολογίου ESS σε Flutter

Στο Flutter ,για ευκολία υπάρχουν 2 κουμπιά: ένα για αποστολή των απαντήσεων κάτω δεξιά και ένα πάνω (Εικόνα 18)που επιστρέφει τον ασθενή στην αρχική σελίδα .

Κεφαλαιο 4:Συμπεράσματα και επεκτάσεις

4.1 Συμπεράσματα

Το πρότυπο JSON για την περιγραφή του ορισμού ενός ιατρικού ερωτηματολογίου που κατασκευάστηκε διαθέτει μεγάλη ικανότητα περιγραφής πολλών διαφορετικών ερωτηματολογίων. Η διαμορφωσιμότητα και η απλότητα του επιτρέπουν στους χρήστες να το χρησιμοποιήσουν με πολλούς διαφορετικούς τρόπους, ώστε να απεικονιστούν και ερωτηματολόγια που δεν εξερευνήθηκαν από την παρούσα εργασία.

Επιπρόσθετα γίνεται εμφανής η διαλειτουργικότητά του συστήματός, από την στιγμή που πολύ εύκολα μπορεί να χρησιμοποιηθεί σε δύο εντελώς διαφορετικές πλατφόρμες ,Blazor (web) και Flutter (mobile ready).

Εμφανής αδυναμία είναι η ανικανότητα αναπαράστασης ερωτηματολογίων που η απάντηση δεν περιγράφεται με συμβολοσειρά. Ένα από αυτά τα ερωτηματολόγια είναι το CDT (Clock Drawing Test)[19], στο οποίο οι ασθενείς απαιτείται να σχεδιάσουν ένα ρολόι που δείχνει συγκεκριμένη ώρα. Ούτε το πρότυπο του ορισμου ερωτηματολογίου σε JSON έχει κατασκευαστεί για να περιέχει εικόνες, αλλά και η εφαρμογή δεν έχει κατασκευαστεί για να τις υποστηρίζει.

4.2 Επεκτάσεις

Θα μπορούσε να επεκταθεί η εφαρμογή και το πρότυπο , ώστε να περιέχει εικόνες ή να επιτρέπει στον ασθενή να σχεδιάζει εικόνες, αλλά αναγνωρίστηκε ως στόχος έξω από τα χρονικά περιθώρια της εργασίας.
Επίσης, το πρότυπο μπορεί να επεκταθεί ως προς τους διαθέσιμους τρόπους βαθμολογίας . Υπάρχουν ερωτηματολόγια που χρησιμοποιούν κλίμακες εντελώς διαφορετικές από το άθροισμα ή τον μέσο όρο κάποιων ερωτήσεων. Για παράδειγμα , το QLQ-PR25 [7]διαθέτει σταθμισμένες κλίμακες, οι οποίες περιγράφονται από γραφική παράσταση.

Σε επίπεδο ασφάλειας, η εφαρμογή είναι πολύ διάτρητη, μιας και μεγάλο μέρος του API δεν απαιτεί κάποιο αναγνωριστικό κωδικό για την επιστροφή δεδομένων. Έγινε μία προσπάθεια απόκρυψης των κωδικών στην βάση δεδομένων με SHA256 αλλά αυτό είναι μόνο ένα μέτρο ασφάλειας. Η ασφάλεια του προγράμματος χρησιμοποιώντας την ίδια λογική , ή ακόμα και με JWT [20]για επικοινωνία, θα μπορούσε να το καταστήσει βιώσιμο στον παγκόσμιο ιστό.

Παράρτημα Α – Βασικές Κλάσεις Υλοποίησης

```
public class Questionnaire
   {
      [Required]
      [StringLength(30)]
      public string questionnaire_definition { get; set; }
      [Required]
      [Range(1, 10)]
      public int version { get; set; }
      [Required]
      public string language { get; set; }
      [Required]
      public List<QuestionGroup> question_group { get; set; }

      public string text { get; set; }

      public List<Scale> scales { get; set; }

      public string CreatorInformation { get; set; }

      public string CopyrightNotice { get; set; }

      public string CopyrightLicence { get; set; }

   }
   public class QuestionGroup
   {
      [Required]
      [StringLength(400)]
      public string title  { get;set; }

      [Required]
      public List<Question> questions { get; set; }
      public List<int> gnums { get; set; }
      public int x { get; set; }
   }
   public class Condition
   {
      public bool hascondition { get; set;  }

      public int groupnumber { get; set; }

      public int questionumber { get; set; }

      public string qvalue { get; set;  }
   }
```

```
public class Question
{
    public int number { get; set; }
    [Required]
    [StringLength(400)]
    public string question { get; set; }
    [Required]
    public string type { get; set; }

    public List<Choice> choices { get; set; }

    public bool nullable { get; set; }

    public Condition condition { get; set;  }

}
public class Choice
{
    public int point { get; set; }
    public string content { get; set; }
}
public class Scale
{
    public string name { get; set;  }

    public string type { get; set; }

    public List<int> questions { get; set; }
}
```

Παραρτημα Β – Υλοποίηση σημείων επαφής- ελεγκτή

Σημεία επαφής

```
endpoints.MapGet("/questionnaire/id={id:int}", (int id) => a.act(id));
endpoints.MapGet("/list", () => a.listings());
endpoints.MapGet("/users", () => a.users());
endpoints.MapGet("/tasks/id={id}", (string id) => a.qtasks(id));
endpoints.MapGet("/patient/id={id}", (string id) => a.gettask(id));


endpoints.MapPost("/newuser", async context =>
    {
        if (!context.Request.HasJsonContentType())
        {
            context.Response.StatusCode = (int)HttpStatusCode.UnsupportedMediaType;
            return;
        }

        string definition = await new        StreamReader(context.Request.Body).ReadToEndAsync();
        string status = a.newusr(definition);
        Console.WriteLine(definition);
        context.Response.StatusCode = (int)HttpStatusCode.Accepted;
        byte[] byteArray = Encoding.ASCII.GetBytes(status.ToString());
        await context.Response.Body.WriteAsync(byteArray);
    });
    endpoints.MapPost("/newpatient", async context =>
    {
        if (!context.Request.HasJsonContentType())
        {
            context.Response.StatusCode = (int)HttpStatusCode.UnsupportedMediaType;
            return;
        }

        string definition = await new StreamReader(context.Request.Body).ReadToEndAsync();
        string hash=a.newpatient(definition);
        Console.WriteLine(definition);

        context.Response.StatusCode = (int)HttpStatusCode.Accepted;
        string message = "Uploaded Succesfully";
        byte[] byteArray = Encoding.ASCII.GetBytes(hash);
        await context.Response.Body.WriteAsync(byteArray);
    });
endpoints.MapPost("/task_removal", async context =>
    {
        if (!context.Request.HasJsonContentType())
```

```
        {
            context.Response.StatusCode = (int)HttpStatusCode.UnsupportedMediaType;
            return;
        }
        List<int> i = await context.Request.ReadFromJsonAsync<List<int>>();
        a.task_removal(i);

        context.Response.StatusCode = (int)HttpStatusCode.Accepted;
    });
endpoints.MapPost("/user_removal", async context =>
    {
        if (!context.Request.HasJsonContentType())
        {
            context.Response.StatusCode = (int)HttpStatusCode.UnsupportedMediaType;
            return;
        }
        List <int> i = await context.Request.ReadFromJsonAsync< List<int>>();
        a.user_removal(i);

        context.Response.StatusCode = (int)HttpStatusCode.Accepted;
    });

endpoints.MapPost("/posting", async context =>
    {
        if (!context.Request.HasJsonContentType())
        {
            context.Response.StatusCode = (int)HttpStatusCode.UnsupportedMediaType;
            return;
        }

        string definition = await new StreamReader(context.Request.Body).ReadToEndAsync();

        a.store(definition);
        Console.WriteLine(definition);
        context.Response.StatusCode = (int)HttpStatusCode.Accepted;
    });

    endpoints.MapPost("/response", async context =>
    {
        //gets response from patient,scores and stores it
        if (!context.Request.HasJsonContentType())
        {
            context.Response.StatusCode = (int)HttpStatusCode.UnsupportedMediaType;
            return;
        }
        string response = await new StreamReader(context.Request.Body).ReadToEndAsync();

        //string score = a.evaluate(response);//evaluation, not ready yet()?
        string score = ""; //fix later
```

```
        a.store_response(response);
        Console.WriteLine(score);

        context.Response.StatusCode = (int)HttpStatusCode.Accepted;
        string message = "Uploaded Succesfully";
        byte[] byteArray = Encoding.ASCII.GetBytes(score);
        await context.Response.Body.WriteAsync(byteArray);
    });

endpoints.MapPost("/login", async context =>
    {
        string log = await new StreamReader(context.Request.Body).ReadToEndAsync();
        User login = new User();
        login = JsonConvert.DeserializeObject<User>(log);
        if (login is null)
            context.Response.StatusCode = (int)HttpStatusCode.BadRequest;
        else

        {
            Credentials creds = a.loggin(login.username, login.password);
            int x = creds.x;

            if (x > -1)
            {
                context.Response.StatusCode = (int)HttpStatusCode.Accepted;
                string credentials = JsonConvert.SerializeObject(creds);
                byte[] byteArray = Encoding.ASCII.GetBytes(credentials);
                await context.Response.Body.WriteAsync(byteArray);

            }

            else
                context.Response.StatusCode = (int)HttpStatusCode.Unauthorized;
        }
    });
```

Κλάση ελεγκτή

```
public class NewController

    {

        static string ComputeSha256Hash(string rawData)

        {

            // Create a SHA256

            using (SHA256 sha256Hash = SHA256.Create())

            {

                // ComputeHash - returns byte array
```

```
        byte[] bytes = sha256Hash.ComputeHash(Encoding.UTF8.GetBytes(rawData));

        // Convert byte array to a string
        StringBuilder builder = new StringBuilder();
        for (int i = 0; i < bytes.Length; i++)
        {
            builder.Append(bytes[i].ToString("x2"));
        }
        return builder.ToString();
    }
}
public string act(int i)
//fetching questionnaire for frontend
{

    QDbContext one = new QDbContext();
    Questionnaire s = one.Questionnaires.Find(i);
    if (s != null)
    {
        return  s.definition;
    }
    return "No questionnaire like that";
}
public Credentials loggin(string usr, string psw)
{
    QDbContext one = new QDbContext();
    var query = one.Users.Where(s => s.username == usr & s.password == psw).ToList();
    Credentials b = new Credentials();
    if (query.Count == 0)
    {
        b.x = -1;
```

```
        return b;
    }

    else
    {
        b.x = query[0].privilege;
        b.id = query[0].Id;
        b.hash = ComputeSha256Hash(DateTime.Now.ToString());
        return b;
    }
}
public string newusr(string definition)
{
    //store new questionnaire in true db
    User q = JsonConvert.DeserializeObject<User>(definition);
    QDbContext one = new QDbContext();
    if (one.Users.Any(c => c.email == q.email))
    {
        return "1";
    }
    if (one.Users.Any(c => c.username == q.username))
    {
        return "2";
    }
    one.Users.Add(q);
    one.SaveChanges();
    return "0";
}
public string newpatient(string definition)
{
    //store new questionnaire in true db
```

```
        QuestionnaireTask q = JsonConvert.DeserializeObject<QuestionnaireTask>(definition);
        QDbContext one = new QDbContext();
        q.DateIssued = DateTime.UtcNow;
        q.DateCompleted = DateTime.MinValue;
        q.completion = false;
        q.result = "";
        q.hashkey = ComputeSha256Hash(q.DateIssued.ToString());
        one.QuestionnaireTasks.Add(q);
        one.SaveChanges();
        return q.hashkey;
    }
    public string gettask(string hashkey)
    {
        QDbContext one = new QDbContext();
        try
        {
            var task = one.QuestionnaireTasks.Where(s => s.hashkey == hashkey).First();
            if (task.completion == false)
            {
                var qid = task.qid;
                var definition = one.Questionnaires.Where(s => s.Id == qid).First().definition;
                    return definition.Substring(0, definition.Length - 1) + ",\"questionnaire_id\":" +
qid.ToString() + "}";
            }
            else
            {
                return "{}";
            }
        }
        catch
        {
```

```
            return "{}";
        }
    }
    public string qtasks(string id)
    {
        QDbContext one = new QDbContext();
        var usr = one.Users.Where(s => s.username == id).FirstOrDefault();
        //get list of tasks according to who asks
        string result = "";
        string jsonString;
        if (usr.privilege == 0)
        {
            foreach (var q in one.QuestionnaireTasks.ToList())
            {
                jsonString = System.Text.Json.JsonSerializer.Serialize(q);
                result = result + jsonString + ",";
            }
        }
        else
        {
                            foreach (var q in one.QuestionnaireTasks.Where(s => s.issued_by
==usr.username ).ToList())
            {
                jsonString = System.Text.Json.JsonSerializer.Serialize(q);
                result = result + jsonString + ",";
            }

        }
        if (result.Length == 0)
            return "[]";
        return "[" + result.Substring(0, result.Length - 1) + "]";
```

```
    }
    public string store(string definition)
    {
        //store new questionnaire in true db
        questionnaire q = JsonConvert.DeserializeObject<questionnaire>(definition);
        string cleandefinition = JsonConvert.SerializeObject(q);

        QDbContext one = new QDbContext();

         Questionnaire s = new Questionnaire {  definition = cleandefinition, version =1, language = q.language,name=q.questionnaire_definition, date= DateTime.UtcNow };
        one.Questionnaires.Add(s);
        one.SaveChanges();
        return "Done";
    }
    private class respond
    {
        public int questionnaire_id { get; set; }
        public List<Answer> answers { get; set; }
        public DateTime dateofbirth { get; set; }
        public string subjectname { get; set; }
        public string hashkey { get; set;  }
    }
    private class questionnaire
    {
        public string questionnaire_definition { get; set; }
        public int version { get; set; }

        public string language { get; set; }
        public List<QuestionGroup> question_group { get; set; }
        public string text { get; set; }
```

```
    public List<Scale> scales { get; set; }
    public string CreatorInformation { get; set; }
    public string CopyrightNotice { get; set; }
    public string CopyrightLicence { get; set; }
}
private class token
{
    public string hashkey { get; set; }
}
private class Answer
{
    public int group_n { get; set; }
    public int question_n { get; set; }

    public string value { get; set; }
}
public string store_response(string definition)
{
    //store response in response db
    respond r = JsonConvert.DeserializeObject<respond>(definition);
    string hashkey = r.hashkey;
    QDbContext one = new QDbContext();
    QuestionnaireTask s = one.QuestionnaireTasks.Where(s => s.hashkey == hashkey).First();
    s.result = definition;
    s.DateCompleted= DateTime.UtcNow;
    s.completion = true;
    one.QuestionnaireTasks.Update(s);
    one.SaveChanges();
    return "Done";
}
public string evaluate(string definition)
```

```
    {
        respond r = JsonConvert.DeserializeObject<respond>(definition);
                                                    TempQuestionnaire questionnaire=
JsonConvert.DeserializeObject<TempQuestionnaire>(act(r.questionnaire_id));
        string answer_value="0";
        foreach (var scale in questionnaire.scales)
        {
            scale.result = 0;
            foreach (int question_id in scale.questions)
            {
                int qgroup = question_id / 100;
                int qnumber = question_id % 100;
                foreach (var answ  in r.answers)
                {
                    if (answ.group_n == qgroup & answ.question_n == qnumber)
                    {
                        answer_value = answ.value;
                        break;
                    }
                }
                if (questionnaire.question_group[qgroup].questions[qnumber].type == "number")
                {
                    scale.result += Convert.ToInt32(answer_value);
                }
                if (questionnaire.question_group[qgroup].questions[qnumber].type == "radiobutton")
                {
                                                        foreach (Choice i in
questionnaire.question_group[qgroup].questions[qnumber].choices)
                    {
                        if (answer_value == i.content)
                        {
```

```
                    scale.result += i.point;
                    break;
                }
            }
        }
        if (scale.type == "average")
        {
            scale.result = scale.result / scale.questions.Count;
        }
    }
    answer_value = "";
    foreach (var scale in questionnaire.scales)
    {
        answer_value += "{\"name\":\""+scale.name;
        answer_value += "\",\"result\":";
         answer_value += scale.result.ToString()+"}";
        answer_value += ",";
    }
    return answer_value.Substring(0,answer_value.Length-1);
}
public int user_removal(List<int> i)
{
    QDbContext one = new QDbContext();
    foreach (int id in i)
    {
        one.Users.Remove(new User { Id=id});
    }
    one.SaveChanges();
    return 0;
}
```

```
public int task_removal(List<int> i)
{
    QDbContext one = new QDbContext();
    foreach (int id in i)
    {
        one.QuestionnaireTasks.Remove(new QuestionnaireTask { Id = id });
    }
    one.SaveChanges();
    return 0;
}
public string listings()
{
    //get list of questionnaires from db
    string result="";
    QDbContext one = new QDbContext();
    string jsonString;
            foreach (var q in one.Questionnaires.Select(p => new { Id = p.Id ,name= p.name ,language=p.language}).ToList())
    {
        jsonString = System.Text.Json.JsonSerializer.Serialize(q);
        result = result + jsonString + ",";
    }
    return "["+result.Substring(0, result.Length - 1)+"]";
}
public string users()
{
    //get list of users for admin
    string result = "";
    QDbContext one = new QDbContext();
    string jsonString;
```

```
            foreach (var q in one.Users.Select(p => new { Id = p.Id, First_name = p.First_name, Last_name = p.Last_name, username=p.username,email=p.email, privilege = p.privilege }).ToList())
        {
            jsonString = System.Text.Json.JsonSerializer.Serialize(q);
            result = result + jsonString + ",";
        }
        return "[" + result.Substring(0, result.Length - 1) + "]";
    }
}
```

Βιβλιογραφία

[1] J. B. W. Williams, “A Structured Interview Guide for the Hamilton Depression Rating Scale,” *Arch. Gen. Psychiatry*, vol. 45, no. 8, p. 742, Aug. 1988, doi: 10.1001/archpsyc.1988.01800320058007.

[2] R. H. Gault, “A History of the Questionnaire Method of Research in Psychology,” *Pedagog. Semin.*, vol. 14, no. 3, pp. 366–383, Sep. 1907, doi: 10.1080/08919402.1907.10532551.

[3] R. L. Spitzer, “Validation and Utility of a Self-report Version of PRIME-MD<SUBTITLE>The PHQ Primary Care Study</SUBTITLE>,” *JAMA*, vol. 282, no. 18, p. 1737, Nov. 1999, doi: 10.1001/jama.282.18.1737.

[4] M. A. Robinson, “Using multi-item psychometric scales for research and practice in human resource management,” *Hum. Resour. Manage.*, vol. 57, no. 3, pp. 739–750, May 2018, doi: 10.1002/hrm.21852.

[5] J. Richardson, A. Iezzi, M. A. Khan, and A. Maxwell, “Validity and Reliability of the Assessment of Quality of Life (AQoL)-8D Multi-Attribute Utility Instrument,” *Patient - Patient-Centered Outcomes Res.*, vol. 7, no. 1, pp. 85–96, Mar. 2014, doi: 10.1007/s40271-013-0036-x.

[6] R. L. Spitzer, K. Kroenke, J. B. W. Williams, and B. Löwe, “A Brief Measure for Assessing Generalized Anxiety Disorder,” *Arch. Intern. Med.*, vol. 166, no. 10, p. 1092, May 2006, doi: 10.1001/archinte.166.10.1092.

[7] G. van Andel *et al.*, “An international field study of the EORTC QLQ-PR25: A questionnaire for assessing the health-related quality of life of patients with prostate cancer,” *Eur. J. Cancer*, vol. 44, no. 16, pp. 2418–2424, Nov. 2008, doi: 10.1016/j.ejca.2008.07.030.

[8] H.-Y. Chiu *et al.*, “Diagnostic accuracy of the Berlin questionnaire, STOP-BANG, STOP, and Epworth sleepiness scale in detecting obstructive sleep apnea: A bivariate meta-analysis,” *Sleep Med. Rev.*, vol. 36, pp. 57–70, Dec. 2017, doi: 10.1016/j.smrv.2016.10.004.

[9] E. L. Rhoden, C. Telöken, P. R. Sogari, and C. A. Vargas Souto, “The use of the simplified International Index of Erectile Function (IIEF-5) as a diagnostic tool to study the prevalence of erectile dysfunction,” *Int. J. Impot. Res.*, vol. 14, no. 4, pp. 245–250, Aug. 2002, doi: 10.1038/sj.ijir.3900859.

[10] M. O. Little, “The Rapid Geriatric Assessment: A Quick Screen for Geriatric Syndromes.,” *Mo. Med.*, vol. 114, no. 2, pp. 101–104, [Online]. Available: http://www.ncbi.nlm.nih.gov/pubmed/30228554

[11] “JSON Official Website.” https://www.json.org/json-en.html

[12] “MariaDB Official Website.” https://mariadb.org/

[13] “MariaDB Docker Container Download Page.” https://hub.docker.com/_/mariadb

[14] “Docker Documentation Website.” https://docs.docker.com/

[15] H. Handschuh, “SHA Family (Secure Hash Algorithm),” in *Encyclopedia of Cryptography and Security*, Springer US, pp. 565–567. doi: 10.1007/0-387-23483-7_388.

[16] “Entity Framework Core Documentation”, [Online]. Available: https://docs.microsoft.com/en-us/ef/core/

[17] “Flutter Official Webpage.” https://flutter.de/

[18] “ISO 639-1.” https://www.iso.org/standard/22109.html

[19] B. K. P. Woo, V. A. Rice, S. A. Legendre, D. P. Salmon, D. V. Jeste, and D. D. Sewell, “The Clock Drawing Test as a Measure of Executive Dysfunction in Elderly Depressed Patients,” *J. Geriatr. Psychiatry Neurol.*, vol. 17, no. 4, pp. 190–194, Dec. 2004, doi: 10.1177/0891988704269820.

[20] M. Jones, J. Bradley, and N. Sakimura, “JSON Web Token (JWT),” May 2015. doi: 10.17487/RFC7519.