ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ
ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ
ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ
ΤΟΜΕΑΣ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΥΠΟΛΟΓΙΣΤΩΝ

Εφαρμογή φαρμακευτικής συμμόρφωσης σε αρχιτεκτονική serverless computing

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

Αντιγόνη Μοίρα

Επιβλέπων: Παναγιώτης Τσανάκας
Καθηγητής Ε.Μ.Π.

Αθήνα, Ιούλιος 2021

ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ
ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ
ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ
ΤΟΜΕΑΣ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΥΠΟΛΟΓΙΣΤΩΝ

Εφαρμογή φαρμακευτικής συμμόρφωσης σε αρχιτεκτονική serverless computing

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

Αντιγόνη Μοίρα

Επιβλέπων : Παναγιώτης Τσανάκας
Καθηγητής Ε.Μ.Π.

Εγκρίθηκε από την τριμελή εξεταστική επιτροπή την 8η Ιουλίου 2021.

.....	.....	.....
Παναγιώτης Τσανάκας	Δημήτριος-Διονύσιος Κουτσούρης	Γεώργιος Ματσόπουλος
Καθηγητής Ε.Μ.Π.	Καθηγητής Ε.Μ.Π.	Καθηγητής Ε.Μ.Π.

Αθήνα, Ιούλιος 2021

.....
Αντιγόνη Μοίρα

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

Περίληψη

Σκοπός της παρούσας διπλωματικής εργασίας είναι η σχεδίαση και ανάπτυξη μιας Android εφαρμογής για την βελτίωση της συμμόρφωσης του ασθενούς στην φαρμακευτική αγωγή που είναι διαθέσιμη στην ελληνική αγορά και, γενικότερα, η διευκόλυνση της παρακολούθησης της υγείας του. Η εφαρμογή αυτή στοχεύει να βοηθήσει στην οργάνωση της καθημερινότητας του χρήστη υπενθυμίζοντάς του τη συστηματική λήψη της αγωγής του, την ανανέωση του αποθέματός του, αλλά και τα ραντεβού του με τους θεράποντες ιατρούς. Τέλος, ένας επιπλέον στόχος είναι η σταδιακή δημιουργία μιας βάσης φωτογραφιών των συσκευασιών όλων των σκευασμάτων που κυκλοφορούν στην Ελλάδα, η οποία για την ώρα δεν υπάρχει διαθέσιμη σε κάποια άλλη πηγή.

Κίνητρο της εργασίας αποτελεί το εξής αντιφατικό φαινόμενο: ενώ η ανακάλυψη νέων φαρμάκων και η ανάπτυξη της τεχνολογίας στην Ιατρική οδηγούν σε νέες επαναστατικές μεθόδους διάγνωσης και θεραπείας, για την αντιμετώπιση διαφόρων ασθενειών, σε μεγάλο βαθμό η συμπεριφορά των ασθενών δε συμβαδίζει με τις αντίστοιχες κάθε φορά ιατρικές συστάσεις. Σύμφωνα με τον Παγκόσμιο Οργανισμό Υγείας, η φαρμακευτική συμμόρφωση συνδέεται άρρηκτα με τη λήψη του σωστού σκευάσματος, την κατάλληλη στιγμή, στην απαιτούμενη δόση και για το χρονικό διάστημα που συστήνει ο εκάστοτε θεράπων ιατρός. Αυτή η δυσκολία των ασθενών να ακολουθήσουν πιστά τις οδηγίες που τους έχουν δοθεί από κάποιον εργαζόμενο στο χώρο της υγείας, συνιστά το πρόβλημα της ελλιπούς ιατρικής και φαρμακευτικής συμμόρφωσης.

Ως σύμμαχος της Ιατρικής στην καθοδήγηση και την στήριξη των ασθενών φαίνεται να αναδεικνύονται, τα τελευταία χρόνια, διάφορες ιατρικές εφαρμογές που μπαίνουν κάτω από την ομπρέλα του όρου m-health (mobile Health). Ο πολλαπλασιασμός των δυνατοτήτων των κινητών τηλεφώνων (smartphones) με την παράλληλη ευρεία χρήση τους από το καταναλωτικό κοινό, έχει σταθεί η αφορμή για να αντιμετωπίζονται πλέον τέτοιου είδους εφαρμογές ως μέσα εκπαίδευσης και δέσμευσης των ασθενών στην φαρμακοθεραπεία τους. Ωστόσο, κάποια ιδιαίτερα χαρακτηριστικά των ατόμων που λαμβάνουν συστηματικά μια αγωγή, όπως είναι η ηλικία τους καθώς και η πολυπλοκότητα και η συχνότητα της θεραπείας τους επιβάλλουν περιορισμούς και συγκεκριμένες απαιτήσεις για την ανάπτυξη μιας τέτοιας εφαρμογής.

Ο σχεδιασμός της προτεινόμενης εφαρμογής στηρίχθηκε στο μοντέλο της serverless computing αρχιτεκτονικής και για τον λόγο αυτό επιλέχθηκαν για την υλοποίηση το Flutter SDK και το Firebase, τα οποία είναι κατάλληλα να υποστηρίξουν μια τέτοια δομή.

Λέξεις κλειδιά: φαρμακευτική συμμόρφωση, υπενθύμιση, ειδοποιήσεις, λήψη αγωγής, αυτοπαρακολούθηση, κινητή υγεία, Android εφαρμογές, Flutter, Firebase, serverless computing

Abstract

The purpose of this thesis is the design and development of an Android application for the improvement of patient adherence to medicine available on the greek market and the facilitation of health monitoring of the patient in general. The application aims at aiding the patient organise their daily routine, by reminding them to systematically take their medication, renew their prescriptions and keep their doctors' appointments. An additional objective is the creation of a database with photographs of the packages from all the pharmaceutical products sold in Greece, which at this moment cannot be found by another source.

The motivation for this thesis is the following contradictory fact: while the discovery of new medicine and the technological development in the medical field lead to revolutionary new methods for diagnosing and treating different diseases, in large the behaviour of patients does not conform to the medical recommendations provided. According to the World Health Organization, pharmaceutical adherence is inextricably linked to taking the right medicine at the appropriate moment, in the required dosage and for as long as the doctor monitoring the patient deems necessary. The patient's difficulty at diligently following the instructions of a health-care specialist sums up the problem of medical and pharmaceutical non-adherence.

As of lately, Medicine finds its ally to guiding and supporting patients in the form of various applications which fall under the more general term of m-Health (mobile health). The multitude of possibilities that smartphones provide along with their widespread usage all over the world, has now rendered these applications as a powerful means for the education and commitment of patients to their medication. However, some characteristics of the people who receive treatment regularly, such as their age as well as the complexity and frequency of their medication regime, call for certain considerations and demands in the development of such an application.

The implementation of the suggested application is based on the model of serverless computing architecture. For this reason, Flutter SDK and Firebase were chosen, as they are the suitable tools to design such a structure.

Keywords: pharmaceutical adherence, reminder, push notifications, medication treatment, self-monitoring, mobile health, Android applications, Flutter, Firebase, serverless computing

Ευχαριστίες

Θα ήθελα να ευχαριστήσω θερμά τον επιβλέποντα καθηγητή κ. Παναγιώτη Τσανάκα για την εμπιστοσύνη που μου έδειξε με την ανάθεση αυτής της διπλωματικής εργασίας, καθώς και για την πολύτιμη καθοδήγησή του κατά την εκπόνησή της.

Επίσης, θα ήθελα να απευθύνω ένα μεγάλο ευχαριστώ στην οικογένεια και τους κοντινούς μου ανθρώπους που στάθηκαν δίπλα μου καθόλη την διάρκεια των σπουδών μου.

Μοίρα Αντιγόνη,

Αθήνα, 8η Ιουλίου 2021

Περιεχόμενα

Περίληψη **5**

Abstract **7**

Ευχαριστίες **9**

Περιεχόμενα **10**

Ευρετήριο Εικόνων **13**

Ευρετήριο Πινάκων **16**

Κεφάλαιο 1: Θεωρητικό Πλαίσιο **17**

1.1 Εισαγωγή 17

1.2 Ορισμός της ιατρικής και φαρμακευτικής "συμμόρφωσης" 17

1.3 Κυριότερες αιτίες μη συμμόρφωσης 18

1.3.1 Ψυχολογικές αιτίες 18

1.3.1.1 Αίσθημα ντροπής για την ανάγκη λήψης αγωγής 18

1.3.1.2 Έλλειψη εμπιστοσύνης απέναντι στις συστάσεις του θεράποντα ιατρού 19

1.3.1.3 Παρενέργειες της αγωγής που αποθαρρύνουν τον ασθενή 19

1.3.2 Πρακτικές αιτίες 19

1.3.2.1 Πολυπλοκότητα στη συχνότητα/δοσολογία της φαρμακευτικής αγωγής 19

1.3.2.2 Λήψη φαρμακευτικής αγωγής για διαφορετικές παθήσεις & ηλικία 20

1.3.2.3 Λησμοσύνη 20

1.3.2.4 Η διάρκεια της αγωγής στις περιπτώσεις των χρόνιων νοσημάτων 20

1.4 Επιπτώσεις της μη συμμόρφωσης 21

1.4.1 Επιπτώσεις στην υγεία του ασθενούς σε διαφορετικές ασθένειες 21

1.4.1.1 Καρδειαγγειακά νοσήματα 21

1.4.1.2 Καρκίνος 21

1.4.1.3 Σακχαρώδης διαβήτης 22

1.4.1.4 Νευροεκφυλιστικές ασθένειες 23

1.4.1.5 Άλλα 24

1.4.2 Επιπτώσεις στο Σύστημα Υγείας 24

1.5 Ανάδυση τεχνολογιών mHealth 25

1.5.1 Αποσαφήνιση του όρου mHealth 25

1.5.2 mHealth και συμμόρφωση σε παγκόσμιο επίπεδο 25

Κεφάλαιο 2: Εφαρμογές υπενθύμισης λήψης φαρμακευτικής αγωγής **26**

2.1 Χαρακτηριστικά εύχρηστης εφαρμογής 26

2.2 Υπάρχουσες εφαρμογές 27

2.3 Η προτεινόμενη εφαρμογή MedCurie 27

2.4 Σύγκριση εφαρμογών 28

Κεφάλαιο 3: Αρχιτεκτονική & Εργαλεία σχεδιασμού και ανάπτυξης 29
3.1 Serverless computing architecture 29
3.2 Εργαλεία σχεδιασμού και ανάπτυξης 31
3.2.1 Flutter 31
3.2.2 Firebase 33

Κεφάλαιο 4: Μεθοδολογία και σχεδιασμός εφαρμογής MedImg_App 35
4.1 Εισαγωγή 35
4.2 Δομή εφαρμογής 35
4.2.1 Βάση Δεδομένων 35
4.2.1.1 Realtime Database 35
4.2.1.2 Firebase Storage 37
4.2.2 Authentication 38
4.2.3 App Flow 39
4.3 Οθόνες εφαρμογής 40
4.3.1 Οθόνη Σύνδεσης 40
4.3.2 Αρχική Οθόνη & Αποσύνδεση 40
4.3.3 Οθόνες καταχώρησης barcode 41
4.3.4 Οθόνη πληροφοριών φαρμάκου 42
4.3.5 Οθόνες λήψης φωτογραφίας 43
4.3.6 Οθόνη επιτυχούς καταχώρησης 44
4.4 Πακέτα (packages) & Βιβλιοθήκες 45

Κεφάλαιο 5: Μεθοδολογία και σχεδιασμός εφαρμογής MedCurie 47
5.1 Εισαγωγή 47
5.2 Δομή εφαρμογής 47
5.2.1 Authentication 47
5.2.2 Βάση Δεδομένων 48
5.2.2.1 Realtime Database 48
5.2.2.2 Firebase Storage 48
5.2.2.3 Cloud Firebase 48
5.2.3 Cloud Functions 52
5.3 Οθόνες εφαρμογής 61
5.3.1 Οθόνες Υποδοχής, Σύνδεσης & Εγγραφής 61
5.3.2 Βασικές οθόνες εφαρμογής 62
5.3.2.1 Αρχική οθόνη 62
5.3.2.2 Μενού Εφαρμογής & Προφίλ Χρήστη 64
5.3.2.2 Φάρμακα 64
5.3.2.3 Μετρήσεις 69
5.3.2.4 Σημειώσεις 72
5.3.2.5 Ιατροί 74
5.3.2.6 Φροντιστές 76
5.3.2.7 Συνταγές 78

5.3.2.8 Ραντεβού 80
5.3.3 Αποσύνδεση 82
5.3.4 Ειδοποιήσεις 83
5.4 Πακέτα (packages) & Βιβλιοθήκες 84
Κεφάλαιο 6: Σύνοψη **86**
4.1 Αποτέλεσμα 86
4.2 Αξιολόγηση 87
4.3 Προτάσεις για μελλοντικές επεκτάσεις 88
Βιβλιογραφία **90**

Ευρετήριο Εικόνων

Εικόνα 1. Λόγοι μη συμμόρφωσης ασθενών στην φαρμακευτική αγωγή [4].....18
Εικόνα 2. Συσχέτιση επικινδυνότητας και ποσοστό μη-συμμόρφωσης (αντιστρόφως ανάλογο με την κατοχή φαρμακευτικής αγωγής) σε ασθενείς που υποβλήθηκαν σε μετεγχειρητικής ορμονοθεραπεία για την αντιμετώπιση του καρκίνου του μαστού [18].....22
Εικόνα 3. Οφέλη κόστους του serverless computing [48]……..……29
Εικόνα 4. Μοντέλα FaaS και BaaS [49].....30
Εικόνα 5. Διάγραμμα Flutter Layer [51]……...……...32
Εικόνα 6. Json tree της Realtime database με τις πληροφορίες των φαρμακευτικών σκευασμάτων που κυκλοφορούν στην ελληνική αγορά…….37
Εικόνα 7. Authentication Flow της εφαρμογής MedIm_App……..38
Εικόνα 8. App Flow της εφαρμογής MedIm_App……...39
Εικόνα 9. Οθόνη σύνδεσης……...40
Εικόνα 10α. Αρχική οθόνη……...41
Εικόνα 10β. Οθόνη αποσύνδεσης…….41
Εικόνα 11α. Οθόνη σκαναρίσματος barcode……....42
Εικόνα 11β. Οθόνη πληκτρολόγησης barcode……..42
Εικόνα 12α. Οθόνη πληροφοριών όταν υπάρχει ήδη εικόνα για αυτό το barcode στη βάση μας……43
Εικόνα 12β. Οθόνη πληροφοριών όταν δεν υπάρχει εικόνα για αυτό το φάρμακο…….....43
Εικόνα 13α. Οθόνη λήψης φωτογραφίας……..44
Εικόνα 13β. Οθόνη επεξεργασίας φωτογραφίας……...44
Εικόνα 14. Οθόνη επιτυχούς καταχώρησης……...45
Εικόνα 15. Τα πακέτα που χρησιμοποιήθηκαν για την ανάπτυξη της εφαρμογής, όπως φαίνονται στο αρχείο “pubspec.yaml”[53].....46
Εικόνα 16. Όριο κλήσεων συναρτήσεων από την κονσόλα του Firebase, χωρίς χρέωση…….60
Εικόνα 17α. Authentication - Οθόνη υποδοχής.……62
Εικόνα 17β. Authentication - Οθόνη εγγραφής.……62
Εικόνα 17γ. Authentication - Οθόνη σύνδεσης.……62
Εικόνα 18α. Αρχική οθόνη - Τρέχουσα ημερομηνία…….63
Εικόνα 18β. Αρχική οθόνη - Μετέπειτα ημερομηνία……63
Εικόνα 18γ. Αρχική οθόνη - Οδηγίες λήψης δόσης φαρμάκου……63
Εικόνα 19α. Μενού και προφίλ - Side navigation menu……...64
Εικόνα 19β. Μενού και προφίλ - Οθόνη προφίλ του χρήστη……64
Εικόνα 20α. Οθόνη φαρμάκων - Λίστα φαρμάκων……...65
Εικόνα 20β. Οθόνη φαρμάκων - Λειτουργικότητες οθόνης φαρμάκων……65

Εικόνα 20γ. Οθόνη φαρμάκων - Λεπτομέρειες εγγραφής φαρμάκου……65
Εικόνα 20δ. Οθόνη φαρμάκων - Διαγραφή φαρμάκου……..66
Εικόνα 20ε. Οθόνη φαρμάκων - Βήμα 1 Drop down επιλογής ονόματος φαρμάκου……..…..66
Εικόνα 20στ. Οθόνη φαρμάκων - Επιλογή προσθήκης φωτογραφίας όταν δεν υπάρχει στην βάση……66
Εικόνα 20ζ. Οθόνη φαρμάκων - Λήψη φωτογραφίας συσκευασίας……..67
Εικόνα 20η. Οθόνη φαρμάκων - Επεξεργασία φωτογραφίας……...….67
Εικόνα 20θ. Οθόνη φαρμάκων - Επιστροφή στο Βήμα 1 της καταχώρησης μετά την αποθήκευση της φωτογραφίας στην βάση……67
Εικόνα 20ι. Οθόνη φαρμάκων - Βήμα 2 καταχώρησης φαρμάκου……..……...68
Εικόνα 20κ. Περιοδικότητα κάθε Χ ημέρες, Περιοδικότητα Επιλεγμένες μέρες της εβδομάδας, Περιοδικότητα Κύκλος: Χ πρόσληψη, Υ παύση……..68
Εικόνα 20λ. Οθόνη φαρμάκων - Βήμα 3 καταχώρησης φαρμάκου……....69
Εικόνα 21α. Οθόνη μετρήσεων - Λίστα μετρήσεων αρτηριακής πίεσης……....70
Εικόνα 21β. Οθόνη μετρήσεων - Λίστα μετρήσεων γλυκόζης αίματος…….……70
Εικόνα 21γ. Οθόνη μετρήσεων - Λίστα μετρήσεων θερμοκρασίας…….……..70
Εικόνα 21δ. Οθόνη μετρήσεων - Καταχώρηση αρτηριακής πίεσης…….……..71
Εικόνα 21ε. Οθόνη μετρήσεων - Καταχώρηση μέτρησης γλυκόζης αίματος…….……...71
Εικόνα 21στ. Οθόνη μετρήσεων - Καταχώρηση μέτρησης θερμοκρασίας…….……...71
Εικόνα 21ζ. Οθόνη μετρήσεων - Διαγραφή μιας καταγραφής.……..……...72
Εικόνα 22α. Οθόνη σημειώσεων - Λίστα σημειώσεων…….……..……..73
Εικόνα 22β. Οθόνη σημειώσεων - Λειτουργικότητες της οθόνης σημειώσεων…….……..73
Εικόνα 22γ. Οθόνη σημειώσεων - Λεπτομέρειες εγγραφής σημείωσης…….……..73
Εικόνα 22δ. Οθόνη σημειώσεων - Επεξεργασία σημείωσης…….....74
Εικόνα 22ε. Οθόνη σημειώσεων - Διαγραφή σημείωσης…….....74
Εικόνα 22στ. Οθόνη σημειώσεων - Καταχώρηση νέας σημείωσης…….....74
Εικόνα 23α. Οθόνη ιατρών - Λίστα ιατρών…….....75
Εικόνα 23β. Οθόνη ιατρών - Λειτουργικότητες οθόνης ιατρών…….....75
Εικόνα 23γ. Οθόνη ιατρών - Λεπτομέρειες εγγραφής ιατρού…….....75
Εικόνα 23δ. Οθόνη ιατρών - Επεξεργασία ιατρού…….....76
Εικόνα 23ε. Οθόνη ιατρών - Διαγραφή ιατρού…….....76
Εικόνα 23στ. Οθόνη ιατρών - Καταχώρηση νέου ιατρού…….....76
Εικόνα 24α. Οθόνη φροντιστών - Λίστα φροντιστών…….....77
Εικόνα 24β. Οθόνη φροντιστών - Λειτουργικότητες οθόνης φροντιστών…….....77
Εικόνα 24γ. Οθόνη φροντιστών - Λεπτομέρειες εγγραφής φροντιστή…….....77
Εικόνα 24δ. Οθόνη φροντιστών - Επεξεργασία φροντιστή…….....78
Εικόνα 24ε. Οθόνη φροντιστών - Διαγραφή φροντιστή…….....78
Εικόνα 24στ. Οθόνη φροντιστών - Καταχώρηση νέου φροντιστή…….....78

Εικόνα 25α. Οθόνη συνταγών - Λίστα συνταγών…….....79
Εικόνα 25β. Οθόνη συνταγών - Λειτουργικότητες οθόνης συνταγών…….....79
Εικόνα 25γ. Οθόνη συνταγών - Λεπτομέρειες εγγραφής συνταγής…….....79
Εικόνα 25δ. Οθόνη συνταγών - Διαγραφή συνταγής…….....80
Εικόνα 25ε. Οθόνη συνταγών - Καταχώρηση νέας συνταγής…….....80
Εικόνα 26α. Οθόνη ραντεβού - Λίστα ραντεβού…….....81
Εικόνα 26β. Οθόνη ραντεβού - Λειτουργικότητες οθόνης ραντεβού…….....81
Εικόνα 26γ. Οθόνη ραντεβού - Λεπτομέρειες εγγραφής ραντεβού…….....81
Εικόνα 26δ. Οθόνη ραντεβού - Επεξεργασία ραντεβού…….....82
Εικόνα 26ε. Οθόνη ραντεβού - Διαγραφή ραντεβού…….....82
Εικόνα 26στ. Οθόνη ραντεβού - Καταχώρηση νέου ραντεβού…….....82
Εικόνα 27. Οθόνη αποσύνδεσης…….....83
Εικόνα 28α. Ειδοποιήσεις - Ειδοποίηση λήψης Αγωγής…….....84
Εικόνα 28β. Ειδοποιήσεις - Ειδοποίηση έναρξης συνταγής…….....84
Εικόνα 28γ. Ειδοποιήσεις - Ειδοποίηση λήξης συνταγής…….....84
Εικόνα 28δ. Ειδοποιήσεις - Ειδοποίηση αυριανού ραντεβού με ιατρό…….....84
Εικόνα 29. Τα πακέτα που χρησιμοποιήθηκαν για την ανάπτυξη της εφαρμογής MedCurie, όπως φαίνονται στο αρχείο ‘pubspec.yaml’ [54]…….....85

Ευρετήριο Πινάκων

Πίνακας 1. Σύγκριση εφαρμογών φαρμακευτικής συμμόρφωσης.....28
Πίνακας 2. Προϊόντα Firebase……..33
Πίνακας 3. Realtime database and Storage rules……38
Πίνακας 4. Δομή της συλλογής users στο Cloud Firestore…….……49
Πίνακας 5. Δομή της συλλογής meds στο Cloud Firestore…….....49
Πίνακας 6. Δομή της συλλογής meds_notif στο Cloud Firestore…….…..49
Πίνακας 7. Δομή της συλλογής measurements στο Cloud Firestore……..50
Πίνακας 8. Δομή της συλλογής notes στο Cloud Firestore……50
Πίνακας 9. Δομή της συλλογής doctors στο Cloud Firestore…….……50
Πίνακας 10. Δομή της συλλογής assistants στο Cloud Firestore……...51
Πίνακας 11. Δομή της συλλογής prescriptions στο Cloud Firestore……..51
Πίνακας 12. Δομή της συλλογής prescriptions_notif στο Cloud Firestore……...…….51
Πίνακας 13. Δομή της συλλογής appointments στο Cloud Firestore…….51
Πίνακας 14. Δομή της συλλογής appointments_notif στο Cloud Firestore……...……52
Πίνακας 15. Cloud Firestore rules……...52

Κεφάλαιο 1: Θεωρητικό Πλαίσιο

1.1 Εισαγωγή

Μιας και τις τελευταίες δεκαετίες η ιατρική και οι βιοεπιστήμες έχουν προσφέρει σημαντικές εξελίξεις στον τομέα των φαρμακευτικών αγωγών, εγείρεται το ζήτημα του κατά πόσο αυτές ακολουθούνται σωστά από τους ασθενείς ώστε να έχουν τη μέγιστη αποτελεσματικότητα στην θεραπεία. Σύμφωνα με έκθεση του Παγκόσμιου Οργανισμού Υγείας (W.H.O.) το 2003 [1] το φαινόμενο της αυθαιρεσίας των ασθενών απέναντι στη τήρηση των οδηγιών που έχουν λάβει από επαγγελματίες υγείας είναι τόσο σημαντικό ώστε η αλλαγή νοοτροπίας των ασθενών και η σωστή λήψη της φαρμακευτικής αγωγής θα ήταν πιο ωφέλιμη για την παγκόσμια υγεία σε σχέση με την ανάπτυξη νέων θεραπειών. Έτσι, έχει αναδυθεί ο όρος της "ιατρικής" ή "φαρμακευτικής συμμόρφωσης" για να περιγράψει την συμπεριφορά του ασθενή απέναντι στις οδηγίες του θεράποντα ιατρού και στην φαρμακευτική αγωγή που πρέπει να λάβει αντίστοιχα. Στην Ελλάδα συγκεκριμένα, οι δύο συχνότερες αιτίες για τη θνησιμότητα ασθενών [2] -τα καρδιαγγειακά νοσήματα και ο καρκίνος- εμφανίζουν κάποια κοινά χαρακτηριστικά όσον αφορά τη σημασία της φαρμακευτικής αγωγής στην θετική έκβαση της θεραπείας. Άλλα χρόνια νοσήματα ή σύνδρομα που εμφανίζονται σε μεγάλο ποσοστό του πληθυσμού και στα οποία η φαρμακευτική ρύθμιση αποτελεί το σημαντικότερο παράγοντα της βελτίωσης της ποιότητας ζωής του ασθενούς είναι ο σακχαρώδης διαβήτης (τύπου Ι και τύπου ΙΙ), οι νευροεκφυλιστικές ασθένειες και άλλα.

1.2 Ορισμός της ιατρικής και φαρμακευτικής "συμμόρφωσης"

Στην έκθεση του Π.Ο.Υ. που αναφέρθηκε παραπάνω [1], ορίζεται η έννοια της ιατρικής συμμόρφωσης (medical adherence) ως εξής:

"Ο βαθμός στον οποίο η συμπεριφορά ενός ανθρώπου -λήψη της αγωγής, τήρηση συγκεκριμένης δίαιτας, ή/και πραγματοποίηση αλλαγών στον τρόπο ζωής- αντιστοιχεί στις οδηγίες που του έχουν δοθεί από κάποιον εργαζόμενο στο χώρο της υγείας" ("The extent to which a person's behaviour – taking medication, following a diet, and/or executing lifestyle changes- corresponds with agreed recommendations from a health care provider.")

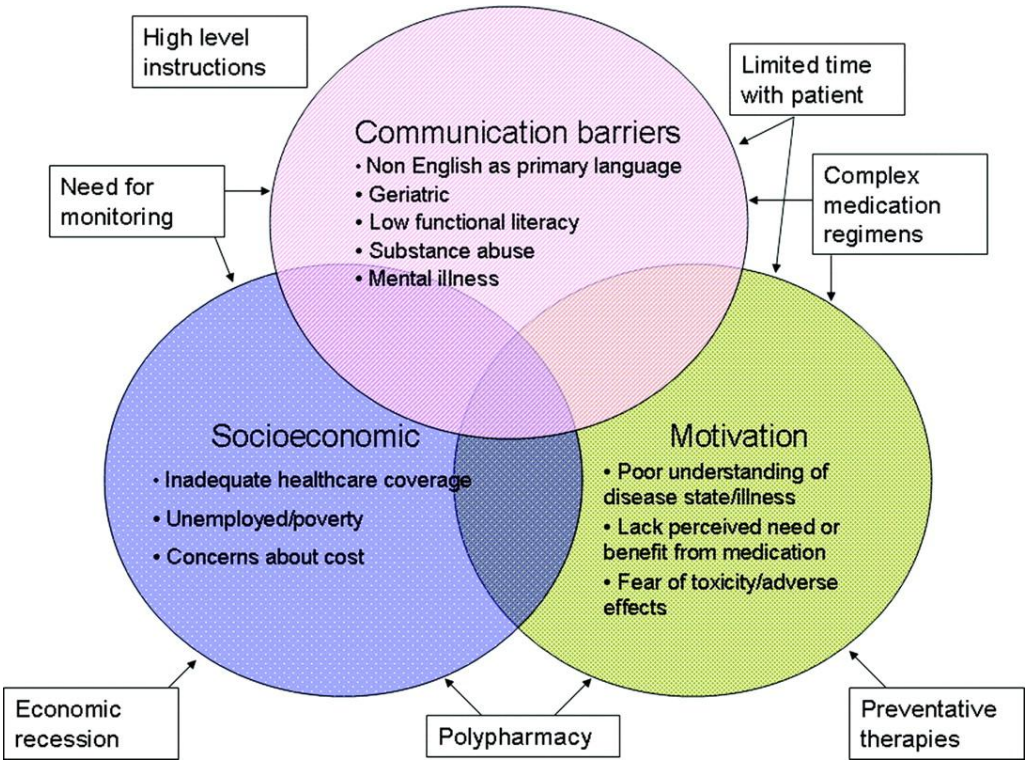
Αντίστοιχα εισάγεται και η έννοια της φαρμακευτικής συμμόρφωσης (treatment compliance): [3]

" Η φαρμακευτική συμμόρφωση συνδέεται άρρηκτα με τη λήψη του σωστού φαρμακευτικού σκευάσματος, την κατάλληλη στιγμή, στην απαιτούμενη δόση και για το χρονικό διάστημα που συστήνει ο εκάστοτε θεράπων ιατρός."

Μπορεί να διαχωριστεί σε πρωτογενή και δευτερογενή μη-συμμόρφωση. Η πρώτη περίπτωση περιγράφει το φαινόμενο κατά το οποίο ο ασθενής δεν πηγαίνει καν στο φαρμακείο για να αποκτήσει τη φαρμακευτική αγωγή που έχει συνταγογραφήσει ο θεράπων ιατρός, ενώ η δεύτερη αφορά στη μη τήρηση των οδηγιών των ιατρών ή/και στη μη συστηματική λήψη της αγωγής [4]. Η συγκεκριμένη εργασία εστιάζει στο φαινόμενο της δευτερογενούς μη-συμμόρφωσης, το ποσοστό της οποίας παρατηρείται κατά 50% σε ασθενείς με χρόνιες παθήσεις, δηλαδή η φαρμακευτική αγωγή λαμβάνεται κατά το ήμισυ -ποσοτικά ή ποιοτικά- ενώ οι περισσότερες θεραπείες είναι αποτελεσματικές για ποσοστό συμμόρφωσης τουλάχιστον στο 80% [5].

1.3 Κυριότερες αιτίες μη συμμόρφωσης

Μιας και το φαινόμενο της μη-συμμόρφωσης έχει αρχίσει να μελετάται πολύ πρόσφατα, θα ήταν επιπόλαιο να υποστηρίξει κανείς ότι γνωρίζουμε όλες τις αιτίες που συμβάλλουν στο φαινόμενο. Το παρακάτω διάγραμμα είναι κατατοπιστικό ώστε να αναδειχθεί ο πολυπαραγοντικός χαρακτήρας του προβλήματος και οι διαφορετικές διαστάσεις που περιλαμβάνει. Εξαρτάται τόσο από κοινωνικοπολιτικά ζητήματα, όσο από τη στάση του ιατρού και την ψυχολογία, ηλικία και το μορφωτικό επίπεδο του ασθενούς, καθώς και τις αλληλεπιδράσεις μεταξύ αυτών των παραγόντων.



Εικόνα 1. Λόγοι μη συμμόρφωσης ασθενών στην φαρμακευτική αγωγή [4]

Στην παρούσα εργασία όπως αναφέρθηκε δίνεται έμφαση στη δευτερογενή μη-συμμόρφωση, δηλαδή όχι στο γιατί οι ασθενείς δεν αγοράζουν την αγωγή αλλά γιατί ενώ την έχουν αγοράσει, δεν την ακολουθούν πιστά και σύμφωνα με τις οδηγίες των ειδικών. Επομένως στη συνέχεια διερευνώνται οι αιτίες αυτής της περίπτωσης.

1.3.1 Ψυχολογικές αιτίες

1.3.1.1 Αίσθημα ντροπής για την ανάγκη λήψης αγωγής

Ειδικότερα σε περιπτώσεις νεότερων ασθενών, υπάρχει η πιθανότητα να γίνονται παρεκτροπές στη λήψη της αγωγής τους λόγω του αισθήματος ντροπής που μπορεί να προκύπτει από το γεγονός ότι εξαρτώνται από αυτήν. Τέτοιες περιπτώσεις αφορούν κυρίως ασθενείς που χρήζουν ψυχιατρικής φαρμακευτικής αγωγής καθώς υπάρχει έντονο το κοινωνικό στίγμα ακόμα και στις μέρες μας σχετικά με θέματα που αφορούν την ψυχική υγεία [6]. Αντίστοιχες περιπτώσεις είναι οι ορμονικές θεραπείες νεαρών ατόμων για την ρύθμιση ενδοκρινολογικών προβλημάτων, θεραπείες που αφορούν σεξουαλικώς μεταδιδόμενα νοσήματα και άλλα.

1.3.1.2 Έλλειψη εμπιστοσύνης απέναντι στις συστάσεις του θεράποντα ιατρού

Δεν είναι λίγες οι έρευνες [7] που εντοπίζουν την αρχή του προβλήματος στο γεγονός ότι ο ιατρός αποτυγχάνει να επικοινωνήσει στον ασθενή τη σημαντικότητα της λήψης συγκεκριμένης αγωγής. Ως αποτέλεσμα, το άτομο που χρήζει ιατροφαρμακευτικής περίθαλψης δεν κατανοεί πλήρως τους λόγους για τους οποίους πρέπει να τηρηθούν οι οδηγίες που του έχουν δοθεί. Παράλληλα, ο πιο απόμακρος τρόπος κάποιων ιατρών, καθώς και το ότι έχει χαθεί ο ρόλος της αυθεντίας τους λόγω των ιατρικών πληροφοριών που πλέον βρίσκονται εύκολα προσβάσιμες σε όποιον τις αναζητήσει, συνθέτουν μια ιδιαίτερη δυναμική ανάμεσα σε θεράποντα και θεραπευόμενο που δεν συνίσταται στην άμεση επικοινωνία και την απόλυτη εμπιστοσύνη.

1.3.1.3 Παρενέργειες της αγωγής που αποθαρρύνουν τον ασθενή

Το συχνότερο παράδειγμα που αναφέρεται στην παγκόσμια βιβλιογραφία όσον αφορά τη συμμόρφωση σε σχέση με τις παρενέργειες συγκεκριμένης αγωγής δεν είναι άλλο από τις χημειοθεραπείες. Όπως και η ενδοφλέβια χορήγησή της, έτσι και η από στόματος λήψη χημειοθεραπείας έχει σημαντικές παρενέργειες οι οποίες επηρεάζουν την ποιότητα ζωής του ασθενούς. Η ναυτία και η κόπωση είναι οι συχνότερα εμφανιζόμενες παρενέργειες σε κάθε τύπο τέτοιας αγωγής. Άλλα φαινόμενα είναι η εμφάνιση πληγών στη στοματική κοιλότητα, δερματικές παθήσεις καθώς και διαταραχές στη λειτουργία του ήπατος και των νεφρών [8], τα οποία αν και δεν είναι απειλητικά για τη ζωή των ασθενών, αποτελούν σοβαρούς λόγους για τους οποίους διακόπτουν ή δεν λαμβάνουν συστηματικά την αγωγή τους χωρίς να ενημερώσουν τον ιατρό που τους παρακολουθεί.

1.3.2 Πρακτικές αιτίες

1.3.2.1 Πολυπλοκότητα στη συχνότητα/δοσολογία της φαρμακευτικής αγωγής

Ο συγκεκριμένος παράγοντας είναι από τους πρώτους που έρχονται στην επιφάνεια σε έρευνες σχετικά με τα αίτια της μη-συμμόρφωσης. Πρόκειται για μία πραγματική πρόκληση κατά το σχεδιασμό μιας θεραπείας και εξαρτάται σε μεγάλο βαθμό από τη θέληση και την οργανωτικότητα του ασθενούς (ή του φροντιστή του) απέναντι στην τήρηση του προγράμματος της αγωγής [9]. Σε πολλές περιπτώσεις απαιτείται πέρα από συγκεκριμένο ωράριο λήψης του φαρμάκου, και συγκεκριμένο ωράριο και τύπος δίαιτας, γεγονός που αυξάνει την πολυπλοκότητα της αγωγής.

Κάποια φάρμακα, όπως το lenalidomide (Revlimid), το οποίο χρησιμοποιείται για τη θεραπεία του πολλαπλού μυελώματος, ακολουθεί ένα σχετικά απλό πρόγραμμα: 2 ή 3 εβδομάδες λήψης και στη συνέχεια 1 διαλειμματική εβδομάδα. Άλλα εμφανίζουν μεγαλύτερη πολυπλοκότητα στις οδηγίες λήψης. Το TAS-102 (Lonsurf), μία από στόματος χημειοθεραπεία για τον καρκίνο του παχέος εντέρου, λαμβάνεται 2 φορές την ημέρα για 5 ημέρες, ακολουθούν 2 εβδομάδες κατά τις οποίες το φάρμακο λαμβάνεται ανά 3 ημέρες και στο τέλος του κύκλου υπάρχουν 2 εβδομάδες "ξεκούρασης", πριν επαναληφθεί πάλι από την αρχή. Μάλιστα μερικές φορές απαιτείται τροποποίηση της δοσολογίας ανάλογα με το βάρος του ασθενούς [8].

1.3.2.2 Λήψη φαρμακευτικής αγωγής για διαφορετικές παθήσεις & ηλικία

Έχει παρατηρηθεί ότι συνήθως οι ασθενείς που δυσκολεύονται να τηρήσουν την αγωγή τους είναι εκείνοι που απαιτείται να λάβουν πολλά διαφορετικά σκευάσματα για την αντιμετώπιση παράλληλων ασθενειών. Οι άνθρωποι που έχουν ανάγκη φαρμακευτικής αγωγής σε καθημερινή βάση ανήκουν κατά μεγάλο ποσοστό σε ηλικίες άνω των 60 ετών. Έρευνα που πραγματοποιήθηκε το 2006 στον Καναδά έδειξε ότι το 88% των ατόμων αυτής της ηλικίας παρουσίαζε τουλάχιστον μία ιατρική πάθηση και το 65% δύο ή περισσότερες παθήσεις [10]. Όσο αυξάνεται η ηλικία, αυξάνεται και ο αριθμός των χρόνιων ασθενειών, για τις οποίες συχνά απαιτείται λήψη διαφορετικών αγωγών. Οι άνθρωποι μεγαλύτερης ηλικίας λαμβάνουν κατά μέσο όρο 6.5 φαρμακευτικά σκευάσματα την ημέρα [11], γεγονός που αυξάνει ιδιαίτερα τη δυσκολία σωστής λήψης τους.

1.3.2.3 Λησμοσύνη

Χαρακτηριστικό παράδειγμα τέτοιας περίπτωσης είναι οι νευροεκφυλιστικές ασθένειες, οι οποίες μιας και εμφανίζονται συχνότερα σε ασθενείς μεγαλύτερης ηλικίας, αποτελούν ένα ακόμα συνδυασμό που οδηγεί στη μη-συμμόρφωση. Σε αυτές τις περιπτώσεις αναφέρονται δυσκολίες τήρησης της αγωγής για λόγους που αφορούν προβλήματα όρασης ή/και μνήμης. Συχνό φαινόμενο στα άτομα αυτά είναι η δυσχέρεια αναγνώρισης ή διάκρισης αντικειμένων [12], οπότε μπορεί να υπάρχει σύγχυση ανάμεσα σε διαφορετικά σκευάσματα. Εξάλλου αξίζει να επισημανθεί ότι τα ίδια τα φάρμακα μπορούν να παραπλανήσουν τον ασθενή: το ίδιο φάρμακο μπορεί να εμφανιστεί σε πολλές παρόμοιες συσκευασίες, αλλά με διαφορετικές δόσεις και τρόπους χορήγησης ή εντελώς διαφορετικά φάρμακα μπορεί να έχουν σχεδόν το ίδιο όνομα ή συσκευασία [36]. Τέλος οι ασθενείς, είναι πιθανό να βιώνουν αλλοιωμένη την έννοια του χρόνου, οπότε να μην λαμβάνουν την αγωγή τους στα σωστά διαστήματα.

Το πρόβλημα της λησμοσύνης όσον αφορά την λήψη της φαρμακευτικής αγωγής μπορεί εύκολα να αντιμετωπιστεί από εξωτερικούς παράγοντες όπως ένας φροντιστής για το άτομο αυτό ή χρήση μιας έξυπνης εφαρμογής που να βοηθά στην υπενθύμιση και αναγνώριση της αγωγής που πρέπει να ληφθεί.

1.3.2.4 Η διάρκεια της αγωγής στις περιπτώσεις των χρόνιων νοσημάτων

Ο παράγοντας του χρόνου είναι μια πρόκληση στη συμμόρφωση ασθενών με χρόνια νοσήματα. Έχει παρατηρηθεί ότι στις συγκεκριμένες περιπτώσεις, το ποσοστό συμμόρφωσης είναι φθίνουσα συνάρτηση του χρόνου, μιας και από κάποιο χρονικό διάστημα και μετά επέρχεται λησμοσύνη της σοβαρότητας της κατάστασης ή/και κόπωση από την ανάγκη τήρησης ενός προγράμματος αγωγής.

Ωστόσο, κοντά στις κλινικές επισκέψεις που αφορούν σε ραντεβού με θεράποντα ιατρό ή σε διαγνωστικές εξετάσεις, παρατηρείται το φαινόμενο της “λευκής συμμόρφωσης” (white-coat compliance) [13]. Αυτό αναφέρεται στην τήρηση της αγωγής πριν και μετά από κάποια προγραμματισμένη επίσκεψη σε ιατρό, το οποίο ναι μεν είναι θετικό γιατί τότε τηρείται η θεραπεία, αλλά δίνει λανθασμένη εικόνα στους επαγγελματίες υγείας με αποτέλεσμα να μην μπορούν να έχουν επαρκή εποπτεία της πορείας του ασθενούς. Πρόκειται για ψυχολογικό φαινόμενο το οποίο μπορεί να αξιοποιηθεί από κάποια τεχνολογική εφαρμογή η οποία να δίνει την αίσθηση της “επίβλεψης” στον ασθενή και άρα να τηρεί εκείνος το πρόγραμμα της αγωγής του.

1.4 Επιπτώσεις της μη συμμόρφωσης

1.4.1 Επιπτώσεις στην υγεία του ασθενούς σε διαφορετικές ασθένειες

1.4.1.1 Καρδειαγγειακά νοσήματα

Ο όρος "καρδιαγγειακά νοσήματα" περιγράφει ένα ευρύ φάσμα ασθενειών και συμπτωμάτων τα οποία κυρίως αφορούν τη στένωση των αγγείων λόγω αθηρωματικής πλάκας. Ως αποτέλεσμα, εμποδίζεται η ομαλή κυκλοφορία του αίματος, επιβαρύνοντας τη λειτουργία της καρδιάς και άλλων οργάνων. Η στεφανιαία νόσος, το αγγειακό εγκεφαλικό επεισόδιο και οι περιφερικές αγγειοπάθειες είναι τα σημαντικότερα νοσήματα αυτού του είδους και σύμφωνα με τον Π.Ο.Υ. αποτελούν την πρώτη αιτία θανάτου παγκοσμίως. Το ίδιο στατιστικό -όπως αναφέρθηκε και στην εισαγωγή- παρατηρείται και στη χώρα μας.

Η λήψη στατινών -μιας κατηγορίας φαρμάκων που μειώνουν τη χοληστερόλη του αίματος- με μέτρια ή υψηλή συχνότητα ενδείκνυται για τους ενήλικες με κλινικά διαγνωσμένη αθηροσκλήρωση [14].

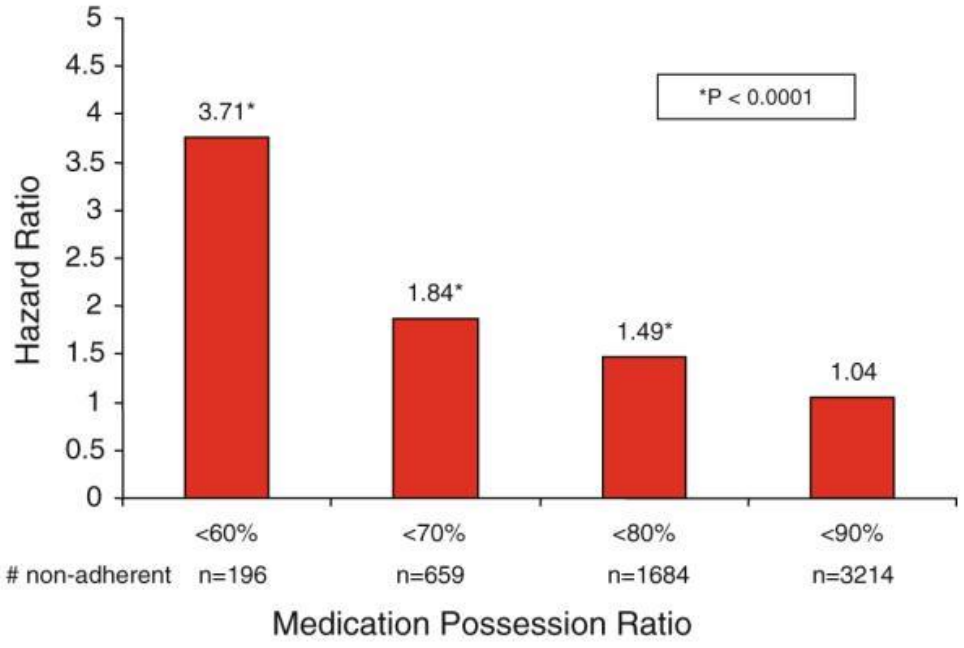
Σχετικά με τη συμμόρφωση σε αυτές τις περιπτώσεις, έχει παρατηρηθεί ότι ασθενείς με υψηλά ποσοστά συμμόρφωσης έχουν σημαντικά μικρότερο ρίσκο καρδιαγγειακών επεισοδίων σε σχέση με αυτούς με χαμηλή συμμόρφωση. Για παράδειγμα, ασθενείς που είχαν εξέλθει από το νοσοκομείο μετά από έμφραγμα του μυοκαρδίου και δεν ακολούθησαν την αγωγή τους για διάστημα 120 ημερών είχαν 80% υψηλότερα ποσοστά θνησιμότητας και όσοι την ακολούθησαν μερικώς είχαν 44% μεγαλύτερη πιθανότητα θανάτου σε σχέση με εκείνους που τήρησαν πιστά τις ιατρικές οδηγίες [4].

1.4.1.2 Καρκίνος

Ο καρκίνος -ως η δεύτερη αιτία θανάτου τόσο παγκοσμίως όσο και στη χώρα μας- περιγράφει μία κατηγορία περιπτώσεων στην οποία υπάρχει ανεξέλεγκτη αναπαραγωγή συγκεκριμένων κυττάρων (κακοήθη) τα οποία εμποδίζουν τη λειτουργία και την ανάπτυξη υγιών ιστών. Ενώ παλαιότερα αντιμετωπιζόταν σχεδόν αποκλειστικά χειρουργικά, πλέον πέρα από την ακτινοβόληση των ιστών και τη χορήγηση ενδοφλέβιας χημειοθεραπείας έχουν αναπτυχθεί πολύ αποτελεσματικά φάρμακα με από στόματος λήψη, τα οποία καθιστούν μία σχεδόν ανίατη ασθένεια σε χρόνιο αλλά αντιμετωπίσιμο νόσημα σε πολλές περιπτώσεις. Ωστόσο, η θετική επίδραση των τεχνολογιών αυτών εξαρτάται σε πολύ μεγάλο βαθμό από την τήρηση της αγωγής από τους ασθενείς [15].

Ένας τύπος καρκίνου για τον οποίο κρίνεται ιδιαίτερα αποτελεσματική η χημειοθεραπεία σε μορφή χαπιού είναι η χρόνια μυελογενής λευχαιμία (chromic myeloid leykemia -CML). Η αγωγή που ακολουθείται από τους ασθενείς ονομάζεται Imatinib και απαιτεί συστηματική λήψη για μεγάλα χρονικά διαστήματα, οπότε αποτελεί μία ευκαιρία καταγραφής της συμμόρφωσης ή μη. Από σχετικές έρευνες [16], μελετάται η συσχέτιση της τήρησης της αγωγής και του διαστήματος EFS (event free survival) δηλαδή για πόσο καιρό μπορεί ο ασθενής να ζήσει χωρίς να παρουσιάσει υποτροπή της ασθένειας. Προκύπτει λοιπόν πως το 5ετές EFS σε ασθενείς με μη-συμμόρφωση ήταν 59.8%, ενώ εκείνων που δεν διέκοψαν καθόλου την αγωγή τους το αντίστοιχο ποσοστό επιβίωσης χωρίς υποτροπή ήταν 76.7% [17].

Ένα άλλο σχετικό παράδειγμα αφορά σε ορμονοθεραπεία που ακολουθεί τη χειρουργική επέμβαση για την αντιμετώπιση του καρκίνου του μαστού (adjuvant hormonal therapy), η οποία έχει φανεί ιδιαίτερα αποτελεσματική στην αύξηση των ποσοστών επιβίωσης σε αυτές τις περιπτώσεις. Σύμφωνα με έρευνα ανάμεσα σε 8769 ασθενείς, για τις οποίες συνταγογραφήθηκε τουλάχιστον μία δόση για ορμονική αγωγή, το 31% των περιπτώσεων διέκοψε τη θεραπεία. Από όσες συνέχισαν, το 28% παρουσίασε μη-συμμόρφωση. Το εκτιμώμενο ποσοστό επιβίωσης στα 10 έτη ήταν 80.7% για όσες συνέχισαν την αγωγή σε σχέση με 73.6% για εκείνες που διέκοψαν. Από όσες συνέχισαν, το αντίστοιχο ποσοστό ήταν 81.7% και 77.8% ανάμεσα σε όσες είχαν συμμόρφωση και μη αντίστοιχα. Τα ποσοστά αυτά αποτυπώνονται και στο παρακάτω διάγραμμα.



Εικόνα 2. Συσχέτιση επικινδυνότητας και ποσοστό μη-συμμόρφωσης (αντιστρόφως ανάλογο με την κατοχή φαρμακευτικής αγωγής) σε ασθενείς που υποβλήθηκαν σε μετεγχειρητική ορμονοθεραπεία για την αντιμετώπιση του καρκίνου του μαστού [18]

Από τα παραπάνω γίνεται προφανές ότι η μη-συμμόρφωση σε περιπτώσεις για τις οποίες χορηγείται φαρμακευτική αγωγή για την αντιμετώπιση κάποιας μορφής καρκίνου, μπορεί να μειώσει σε σημαντικό βαθμό το ποσοστό επιβίωσης του ασθενούς ή να οδηγήσει σε υποτροπή της ασθένειας.

1.4.1.3 Σακχαρώδης διαβήτης

Ο σακχαρώδης διαβήτης (ΣΔ) είναι μια χρόνια νόσος που χαρακτηρίζεται από υψηλά επίπεδα γλυκόζης στο αίμα. Εμφανίζεται είτε όταν το πάγκρεας αδυνατεί να παράξει επαρκή ποσότητα ινσουλίνης (ΣΔ τύπου Ι), η οποία βοηθά στη ρύθμιση του σακχάρου στο αίμα, είτε όταν ο οργανισμός δεν μπορεί να αξιοποιήσει αποτελεσματικά την ινσουλίνη που παράγει λόγω ανεπαρκούς ευαισθησίας των κυττάρων στη δράση της (αντίσταση στην ινσουλίνη (ΣΔ τύπου ΙΙ) [19] [20].

Παρά το ότι υπάρχει ένα ευρύ φάσμα διαθέσιμων επιλογών όσον αφορά στη φαρμακευτική αγωγή για τον ΣΔ, το 50% των ασθενών δεν επιτυγχάνουν επαρκή γλυκαιμικό έλεγχο [21]. Με βάση το είδος της αγωγής, η συμμόρφωση κυμαίνεται μεταξύ 13-64% για τα αντιδιαβητικά δισκία και μεταξύ 19-46% για την ινσουλίνη [22].

Οι επιπτώσεις από τη μη-συμμόρφωση στην αντιδιαβητική αγωγή είναι κυρίως καρδιαγγειακού τύπου, με συχνότερη την εμφάνιση εμφράγματος του μυοκαρδίου και εγκεφαλικών επεισοδίων [19]. Πρόκειται για κινδύνους που θα μπορούσαν να αποφευχθούν σε ποσοστό 15% μέσω της αυστηρής ρύθμισης της γλυκοζυλιωμένης αιμοσφαιρίνης [23]. Τα χαμηλά ποσοστά συμμόρφωσης έχουν συσχετιστεί επίσης με περισσότερες επιπλοκές όπως η αμφιβληστροειδοπάθεια, τα έλκη και οι ακρωτηριασμοί, μερικά εκ των οποίων είναι μη αντιστρέψιμες περιπτώσεις [24].

1.4.1.4 Νευροεκφυλιστικές ασθένειες

Οι νευροεκφυλιστικές ασθένειες είναι ανίατες παθήσεις που επιφέρουν αναπηρία και οδηγούν στην προοδευτική εκφύλιση ή/και τον θάνατο των νευρικών κυττάρων. Αυτή η εξέλιξη προκαλεί προβλήματα στην κίνηση (αταξία) ή στη νοητική λειτουργία (άνοια) [25].

Οι διάφορες μορφές άνοιας είναι υπεύθυνες για το μεγαλύτερο φορτίο αυτών των ασθενειών, ενώ η νόσος Alzheimer (AD) αντιπροσωπεύει περίπου το 60–70% των περιπτώσεων [26] και αποτελεί μια χρόνια εκφυλιστική νόσο του εγκεφάλου με κύρια κλινική εκδήλωση τη διαταραχή στη μνήμη. Θεραπεία δεν υπάρχει μέχρι στιγμής αλλά γίνεται προσπάθεια για την επιβράδυνση της εξέλιξής της μέσω ειδικών για αυτήν την περίπτωση φαρμάκων. Επίσης στους ασθενείς χορηγούνται φάρμακα για άλλα συμπτώματα της νόσου (αγχολυτικά, αντικαταθλιπτικά, υπνωτικά, αντιψυχωσικά).

Ακριβώς επειδή πρόκειται για μία νόσο η οποία πλήττει τη μνήμη και τη νοητική διαύγεια, είναι πολύ δύσκολη η τήρηση των οδηγιών των ιατρών σε θέματα αγωγής, με το ποσοστό συμμόρφωσης να κυμαίνεται από 17% έως 42% και τη διακοπή της αγωγής πριν την ολοκλήρωσή της να παρατηρείται σε ποσοστό από 37% έως 80% [27]. Η μη συμμόρφωση συσχετίζεται με αυξημένο κίνδυνο νοσηλείας και μεγαλύτερη θνησιμότητα και επομένως σε αυτές τις περιπτώσεις κρίνεται απαραίτητος ο ρόλος ενός φροντιστή που θα έχει εποπτεία της ιατροφαρμακευτικής περίθαλψης του ασθενούς.

Η δεύτερη συχνότερη πάθηση αυτής της μορφής είναι η νόσος του Parkinson (PD), μία προοδευτική ιδιοπαθής νόσος του νευρικού συστήματος, που χαρακτηρίζεται από κινητικές και μη κινητικές εκδηλώσεις, όπως τρόμο ηρεμίας, υπερτονία, ακινησία, αμιμία προσώπου και διαταραχές στη στάση και την ισορροπία [28]. Αν και η εξέλιξη της νόσου είναι διαχειρίσιμη σε πρώιμο στάδιο μέσω θεραπείας με ένα μόνο φάρμακο, οι περισσότεροι από τους μισούς ασθενείς με PD λαμβάνουν 2 με 4 αντι-παρκινσονιακά σκευάσματα 3-4 φορές την ημέρα [29] [30].

Το παρατηρούμενο ποσοστό συμμόρφωσης στην PD είναι πολύ χαμηλό -περίπου 10%- με το 76% των ασθενών να παραδέχονται ότι έχουν χάσει ή έχουν πάρει λάθος ώρα κάποια δόση [31]. Για αγωγή με πολλαπλές δόσεις μέσα στο 24ωρο, οι ερευνητές κατέγραψαν τήρηση των ιατρικών οδηγιών σε μόλις το 3% των ασθενών [32]. Οι συνέπειες της μη-συμμόρφωσης είναι σημαντικές και περιλαμβάνουν επανεμφάνιση των κινητικών δυσλειτουργιών [33], ενώ συχνά παρατηρείται λήψη μεγαλύτερης από της αναλογούμενης ποσότητας ντοπαμινεργικής αγωγής [34] με αποτέλεσμα σοβαρή δυσκινησία, πιθανή εμφάνιση συνδρόμου άρσης αναστολών, η οποία μπορεί να οδηγήσει ακόμα και σε ψυχωτικά επεισόδια [35]. Όπως και στην περίπτωση της AD, επομένως, κρίνεται σκόπιμη η συστηματική καταγραφή της αγωγής και τήρηση όσο περισσότερο γίνεται του σωστού προγράμματος για την αποφυγή των παραπάνω παρενεργειών.

1.4.1.5 Άλλα

Υπάρχουν και άλλες πολλές περιπτώσεις στις οποίες το πρόγραμμα της φαρμακευτικής αγωγής είναι ιδιαίτερο, οδηγώντας σε αποκλίσεις. Αναφέροντας μερικές μόνο από αυτές, τέτοιες είναι οι ορμονικές θεραπείες οι οποίες συνήθως κινούνται σε κύκλους, όπως για παράδειγμα είναι η αντισυλληπτική αγωγή, τα ψυχικά νοσήματα που συχνά απαιτούνται αρκετές δοκιμές μέχρι ο θεράπων ιατρός να καταλήξει στην αγωγή που ταιριάζει στις ανάγκες του ασθενούς και φυσικά οι θεραπείες για κάποια περιστασιακή ασθένεια ή μετεγχειρητική αγωγή. Στην τελευταία περίπτωση ακριβώς λόγω του προσωρινού χαρακτήρα της αγωγής, γίνονται αρκετά λάθη επειδή πρόκειται για ένα νέο πρόγραμμα στη ζωή του ατόμου και καθώς η αγωγή δεν αποτελεί κομμάτι της συνήθους καθημερινότητάς του, πολλές φορές δεν λαμβάνεται σωστά.

1.4.2 Επιπτώσεις στο Σύστημα Υγείας

Το 2010 οι δαπάνες για συνταγογραφούμενα φάρμακα στις ΗΠΑ ήταν 259 δισεκατομμύρια δολάρια. Λαμβάνοντας υπόψη τα ποσοστά μη συμμόρφωσης που επικρατούν, οι δαπάνες αυτές μπορούν να μειωθούν αρκετά αν οι ασθενείς δείξουν μεγαλύτερη συνέπεια στην λήψη της αγωγής τους. Η πλειονότητα των δαπανών που αποδίδονται στη μη φαρμακευτική συμμόρφωση προκύπτει από την νοσηλεία ασθενών σε νοσοκομεία που θα μπορούσε να έχει αποφευχθεί [36]. Επιπλέον κόστη προκύπτουν από την πρόοδο της μέχρι τώρα ελεγχόμενης νόσου με: 1) την ανάγκη για νοσηλεία σε κέντρα φιλοξενίας ή αιμοκάθαρσης 2) αποτρέψιμο κόστος φαρμακευτικής αγωγής που σχετίζεται με την εντατικοποίηση της θεραπείας, καθώς αναπτύσσονται συννοσηρές καταστάσεις και 3) διαγνωστικούς ελέγχους που θα μπορούσαν να αποφευχθούν με τον έλεγχο της πρωτογενούς ασθένειας. Στοιχεία δείχνουν ότι το μακροπρόθεσμο κόστος των αρνητικών αποτελεσμάτων στην προσπάθεια συμμόρφωσης των ασθενών υπερβαίνει το κόστος των φαρμάκων σε πολλές χρόνιες ασθένειες [36]. Υπολογίζεται ότι το 10% των νοσηλειών σε ηλικιωμένους μπορεί να προκληθεί από την μη τήρηση της φαρμακευτικής αγωγής [37][38].

Στις ΗΠΑ, το κόστος της υγειονομικής περίθαλψης που θα μπορούσε να αποφευχθεί και αποδίδεται στη μη συμμόρφωση των ασθενών ανέρχεται μεταξύ των 100 και 300 δισεκατομμυρίων δολαρίων ετησίως [36][39]. Το κόστος αυτό αντιπροσωπεύει το 3% έως το 10% του συνολικού κόστους υγειονομικής περίθαλψης των ΗΠΑ [40].

Για την Ελλάδα δεν υπάρχουν αυτή την στιγμή διαθέσιμες επαρκή στατιστικά στοιχεία σχετικά με το ζήτημα της οικονομικής επιβάρυνσης του Εθνικού Συστήματος Υγείας (Ε.Σ.Υ.) λόγω της ελλιπούς φαρμακευτικής συμμόρφωσης.

1.5 Ανάδυση τεχνολογιών mHealth

1.5.1 Αποσαφήνιση του όρου mHealth

Ο όρος mHealth είναι μια συντομογραφία για το mobile health, που στα ελληνικά αποδίδεται ως κινητή υγεία. Ο Παγκόσμιος Οργανισμός Υγείας (Π.Ο.Υ.) ορίζει την κινητή υγεία ως "πρακτική ιατρικής και δημόσιας υγείας που υποστηρίζεται από κινητές συσκευές, όπως κινητά τηλέφωνα, συσκευές παρακολούθησης ασθενών, προσωπικούς ψηφιακούς βοηθούς και άλλες ασύρματες συσκευές". Ο όρος κινητή υγεία πρωτοεμφανίστηκε στην παγκόσμια βιβλιογραφία το 2003 από τον Robert Istepanian και αποτελεί υποσύνολο της ηλεκτρονικής υγείας. Η κινητή υγεία εμφανίστηκε στο παγκόσμιο γίγνεσθαι λόγω της ραγδαίας και δυναμικής εξάπλωσης των κινητών τεχνολογιών και επικοινωνιών.

1.5.2 mHealth και συμμόρφωση σε παγκόσμιο επίπεδο

Στη διπλωματική της εργασία με τίτλο "Μελέτη και καταγραφή των μεθόδων ιατρικής συμμόρφωσης και στατιστική ανάλυση του επιπέδου συμμόρφωσης των ασθενών μέσω mobile εφαρμογών" η Μαρία Πέτρου αναφέρει [41]:

"Σύμφωνα με πρόσφατη έρευνα του Παγκόσμιου Οργανισμού Υγείας, καταδεικνύεται ότι στις χώρες υψηλού εισοδήματος, η mHealth καθοδηγείται από την επιτακτική ανάγκη για μείωση του κόστους της υγειονομικής περίθαλψης, ενώ στις αναπτυσσόμενες χώρες, ωθείται κυρίως από την ανάγκη για πρόσβαση στην πρωτοβάθμια περίθαλψη. Η έρευνα κατέδειξε επίσης ότι ένας από τους πιο πρόσφατους κινητήριους μοχλούς της υγειονομικής περίθαλψης στην Ευρωπαϊκή Ένωση είναι τα συστήματα, τα οποία προωθούν την εξατομικευμένη φροντίδα μέσω φορετών, φορητών ή εμφυτεύσιμων συστημάτων, και δίνουν στους ασθενείς έναν πιο ενεργό ρόλο (τα αποκαλούμενα προσωπικά συστήματα υγείας). Στην Αφρική και στην Ασία, η πλειονότητα των υφιστάμενων υπηρεσιών mHealth επικεντρώνονται στη βελτίωση της αποτελεσματικότητας του εργατικού δυναμικού και των συστημάτων υγειονομικής περίθαλψης. Μια άλλη κατηγορία υπηρεσιών ιδιαίτερα σημαντική στην Ινδία, στη Νότια Αφρική και στην Κένυα περιλαμβάνει μηνύματα πρόληψης και ευαισθητοποίησης για τον περιορισμό της εξάπλωσης μολυσματικών νοσημάτων."

Κεφάλαιο 2: Εφαρμογές υπενθύμισης λήψης φαρμακευτικής αγωγής

2.1 Χαρακτηριστικά εύχρηστης εφαρμογής

Οι εφαρμογές φαρμακευτικής συμμόρφωσης, λαμβάνοντας υπόψιν το κοινό στο οποίο απευθύνονται, αλλά και τον σκοπό για τον οποίο χρησιμοποιούνται, θα πρέπει να έχουν τα εξής χαρακτηριστικά για να μπορούν να θεωρηθούν εύχρηστες [36]:

- Εξασφάλιση μεγαλύτερης αυτονομίας στους ασθενείς: Η εφαρμογή να δίνει την δυνατότητα στον χρήστη με διάφορα μέσα και συστήματα να οργανώσει την παρακολούθηση της φαρμακευτικής αγωγής του αυτόνομα και με επάρκεια. Η καταχώρηση, αλλά και η παρακολούθηση της αγωγής του, να είναι μια εύκολη και γρήγορη διαδικασία αποτελούμενη από όσο το δυνατόν λιγότερα και πιο απλά βήματα. Ο ίδιος ο ασθενής θα πρέπει να μπορεί να χειρίζεται την εφαρμογή, χωρίς να προκύπτει η ανάγκη υποστήριξής του από τον ιατρό του ή κάποιο φροντιστή.

- Οργάνωση όλης της χρήσιμης πληροφορίας που αφορά την γενική κατάσταση της υγείας του χρήστη: Είναι ιδιαίτερα βοηθητικό για τον ασθενή να μπορεί να έχει καταχωρημένες και οργανωμένες μέσω μιας εφαρμογής τις περισσότερες πληροφορίες που αφορούν την υγεία του και γενικότερα την καθημερινότητα σε σχέση με την ασθένειά του.

- Απλότητα και ευκολία χρήσης: Πρέπει να είναι από τα βασικά στοιχεία του σχεδιασμού και της ανάπτυξης της εφαρμογής, καθώς απευθύνεται σε ευρύ κοινό που περιλαμβάνει και τις μεγαλύτερες ηλικιακές ομάδες. Αυτό μεταφράζεται στο πώς θα σχεδιαστεί τόσο το γραφικό περιβάλλον της εφαρμογής όσο και η ροή χρήσης της. Ωστόσο, αυτή η απλοποίηση της εφαρμογής δεν πρέπει να πραγματοποιηθεί με την απόκρυψη πληροφοριών από τον χρήστη, αλλά με την οργάνωσή τους με τέτοιο τρόπο, ώστε να παρουσιάζονται μόνο όταν είναι απαραίτητο.

- Αξιοπιστία και ασφάλεια: Καθώς τέτοιου είδους εφαρμογές αφορούν την υγεία του χρήστη, είναι πολύ σημαντικό να λαμβάνει τις απαραίτητες ειδοποιήσεις για την αγωγή του την σωστή ώρα και η διαδικασία να εξαρτάται από όσο γίνεται λιγότερες παραμέτρους (πχ σύνδεση στο διαδίκτυο, εκτέλεση εφαρμογής στο background κλπ). Επίσης, τα προσωπικά δεδομένα του χρήστη θα πρέπει να αποθηκεύονται σε μια βάση δεδομένων με αυστηρούς κανόνες ασφαλείας και όχι στην συσκευή του από την οποία θα μπορούσαν εύκολα να διαρεύσουν.

- Φορητότητα: Ο χρήστης θα πρέπει να μπορεί να έχει πρόσβαση στα δεδομένα του από πολλές συσκευές και να μην απαιτείται η καταχώρηση από την αρχή της αγωγής του και των υπόλοιπων στοιχείων του σε περίπτωση που χρειαστεί να αλλάξει συσκευή.

- Χαμηλό κόστος: Το σύστημα να μην χρειάζεται εξειδικευμένο hardware υψηλού κόστους. Η εφαρμογή να μπορεί να εγκατασταθεί σε οποιοδήποτε smartphone, χωρίς να απαιτεί πολλούς υπολογιστικούς πόρους και μνήμη.

2.2 Υπάρχουσες εφαρμογές

Υπάρχουν πολλές διαθέσιμες εφαρμογές για smartphones που απευθύνονται στην φαρμακευτική συμμόρφωση, αλλά λίγες από αυτές έχουν χαρακτηριστικά που τις καθιστούν κάτι παραπάνω από ηλεκτρονικό ημερολόγιο. Παρακάτω παρουσιάζονται μερικές εφαρμογές που εξασφαλίζουν καλή παρακολούθηση του προγράμματος λήψης φαρμάκων, μερικές φορές προσφέροντας επιπλέον την αποθήκευση μετρήσεων για ζωτικά σημεία όπως ο καρδιακός ρυθμός, η αρτηριακή πίεση και τα επίπεδα γλυκόζης στο αίμα. Μερικές παρέχουν ακόμα την δυνατότητα της εύκολης κοινοποίησης τέτοιων πληροφοριών στον θεράποντα ιατρό. Επίσης, υπάρχουν κάποιες προηγμένες δυνατότητες όπως η πρόσβαση σε βάσεις δεδομένων φαρμάκων, η αποθήκευση δεδομένων στο cloud και η καταχώρηση παραγγελιών σε φαρμακεία (αυτή είναι μια υπηρεσία που δεν είναι χρήσιμη στην ελληνική αγορά). Μια σύντομη λίστα εμπορικών εφαρμογών είναι η ακόλουθη:

- Medisafe [42]: Δίνει υποστήριξη στους ασθενείς που έχουν ανάγκη την διαχείριση περίπλοκων φαρμακευτικών αγωγών. Μπορεί να συγχρονίσει δεδομένα με άλλες συσκευές σε πραγματικό χρόνο με σκοπό να μπορούν και άλλα οικογενειακά πρόσωπα να βοηθήσουν τον χρήστη και να ειδοποιηθούν σε περίπτωση παράβλεψεις κάποιας δόσης. Χαρακτηρίζεται επίσης από απλό και εύχρηστο interface.

- Med Helper [43]: Είναι μια δωρεάν εφαρμογή για προγραμματισμό της φαρμακευτικής αγωγής με καλά αποτελέσματα όσων αφορά την σαφήνεια και απλότητα της χρήσης. Ο χρήστης ειδοποιείται όταν πρέπει να πάρει την αγωγή του ή όταν οι συνταγές του πρόκειται να λήξουν και χρειάζονται ανανέωση.

- Pill Reminder [44]: Πρόκειται για μια δωρεάν εφαρμογή που είναι σχεδόν αποκλειστικά περιορισμένη στον προγραμματισμό και αποστολή ειδοποιήσεων για ιατρικά ραντεβού με καλά αποτελέσματα όσων αφορά την σαφήνεια και απλότητα της χρήσης.

- Pill Manager [45]: Στέλνει ειδοποιήσεις σχετικά με τον προγραμματισμό λήψης φαρμάκων. Οι χρήστες μπορούν να κάνουν preview στο πρόγραμματα της αγωγής τους και να οργανώσουν τις παραγγελίες των φαρμάκων τους από εγγεγραμμένα φαρμακεία.

- Dosecast [46]: Προσφέρει ευέλικτο πρόγραμμα λήψης φαρμακευτικής αγωγής και δίνει πρόσβαση σε μια βάση δεδομένων φαρμάκων.

- Care4today [47]: Προορίζεται ως μια εφαρμογή υγείας και ευ ζειν. Η υπενθύμιση των φαρμάκων βασίζεται σε μια βάση δεδομένων που περιέχει φωτογραφίες των συσκευασιών τις οποίες ο χρήστης εύκολα μπορεί να βρει και να προσθέσει τις δικές του. Είναι κάτι παραπάνω από μια απλή εφαρμογή ιατρικών υπενθυμίσεων καθώς μπορεί να καταγράψει προσωπικές μετρήσεις και διάφορα ιατρικά δεδομένα που καταχωρεί ο χρήστης.

2.3 Η προτεινόμενη εφαρμογή MedCurie

Στο πλαίσιο της παρούσας εργασίας, αναπτύξαμε την εφαρμογή φαρμακευτικής συμμόρφωσης MedCurie, δίνοντας έμφαση σε όλα τα χαρακτηριστικά που αναφέρθηκαν στην προηγούμενη ενότητα, ώστε να αποτελέσει ένα πρακτικό εργαλείο για τους χρήστες της.

Η εφαρμογή MedCurie απευθύνεται σε ασθενείς που έχουν πρόσβαση στην ελληνική αγορά φαρμάκων. Υποστηρίζεται από μια βάση δεδομένων με όλα τα φάρμακα που κυκλοφορούν στην αγορά και τις εικόνες των

συσκευασιών τους, σε μια προσπάθεια να διευκολύνει την διαδικασία της λήψης της αγωγής από τους ασθενείς. Η καταχώρηση των φαρμάκων που λαμβάνει κάθε ασθενής γίνεται με απλό και γρήγορο τρόπο, ενώ του δίνεται η δυνατότητα να ορίσει και σύνθετα προγράμματα πρόσληψης (σε συγκεκριμένες ημέρες της εβδομάδας, σε κύκλους κλπ), ώστε να εξυπηρετεί με τον βέλτιστο τρόπο τις ανάγκες του.

Μέσω της εφαρμογής ο ασθενής μπορεί να συγκεντρώσει όλες τις βασικές πληροφορίες που αφορούν την υγεία του και να έχει άμεση πρόσβαση σε αυτές όποτε το χρειαστεί. Η εφαρμογή υποστηρίζει την καταχώρηση μετρήσεων ζωτικών σημείων, την αποθήκευση των στοιχείων επικοινωνίας των ιατρών και των φροντιστών του, την καταγραφή σημειώσεων σχετικών με την υγεία του και την υπενθύμιση των ραντεβού του και των ημερομηνιών έναρξης και λήξης των συνταγών του. Με αυτόν τον τρόπο μπορεί να διαδραματίσει υποστηρικτικό ρόλο σε πολλά θέματα της καθημερινότητας του ασθενούς.

Όλα τα δεδομένα του ασθενή αποθηκεύονται σε μια cloud database, εξασφαλίζοντας έτσι την ασφάλεια και την πρόσβαση σε αυτά από οποιαδήποτε συσκευή επιθυμεί ο χρήστης. Τέλος, η λειτουργικότητα των ειδοποιήσεων έχει στηριχθεί σε serverless αρχιτεκτονική με αποτελέσματα την ακρίβεια και την αξιοπιστία της, καθώς και την ελάχιστη επιβάρυνση της συσκευής του χρήστη.

2.4 Σύγκριση εφαρμογών

Στον παρακάτω πίνακα παρουσιάζεται μια σύνοψη των παραπάνω εμπορικών εφαρμογών σε σύγκριση με την προτεινόμενη εφαρμογή MedCurie:

	Υπενθύμιση Αγωγής	Καταγραφή μετρήσεων ζωτικών σημείων	Cloud Database	Απομακρυσμένη παρακολούθηση από φροντιστή	Βάση δεδομένων φαρμάκων	Παραγγελία σε φαρμακείο
Medisafe	X	X	X	X	X	
Med Helper	X			X		
Pill Reminder	X					
Pill Manager	X	X				X
Dosecase	X				X	
Care4today	X		X			
MedCurie	X	X	X		X	

Πίνακας 1. Σύγκριση εφαρμογών φαρμακευτικής συμμόρφωσης

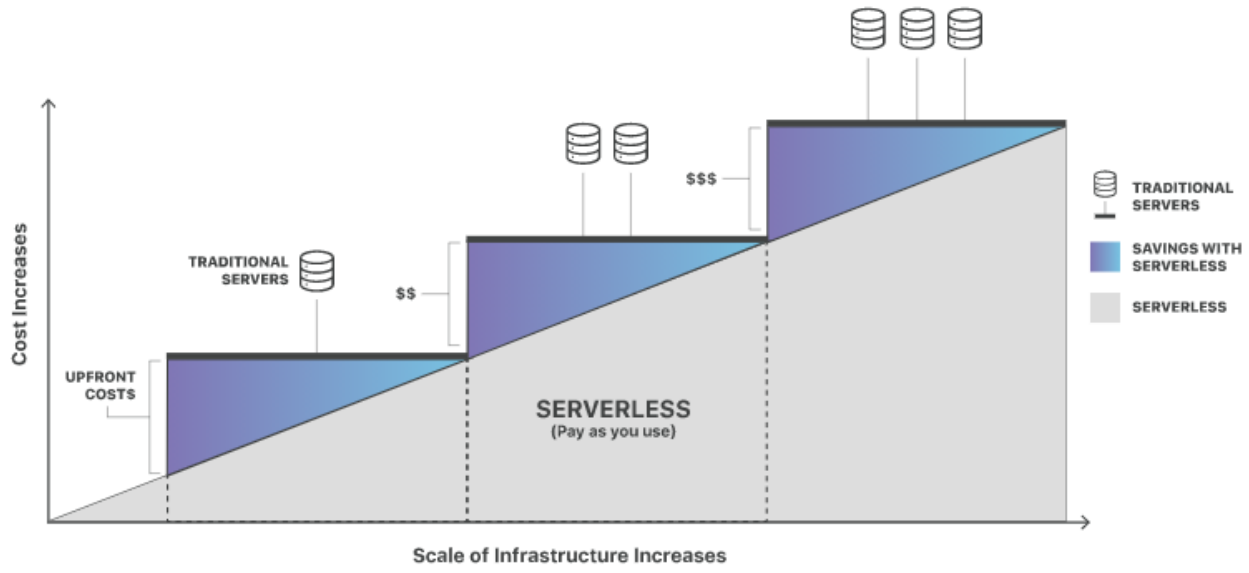
Κεφάλαιο 3: Αρχιτεκτονική & Εργαλεία σχεδιασμού και ανάπτυξης

3.1 Serverless computing architecture

Το serverless computing είναι μια μέθοδος παροχής backend υπηρεσιών σε μόνιμη βάση. Ένας πάροχος serverless υπηρεσιών επιτρέπει στους χρήστες του να γράψουν και να εκτελέσουν κώδικα χωρίς να πρέπει να ανησυχούν για την υποκείμενη υποδομή. Μια εταιρεία που χρησιμοποιεί backend υπηρεσίες από έναν πάροχο serverless χρεώνεται βάση της χρήσης και δεν χρειάζεται να κρατήσει και να πληρώσει για ένα συγκεκριμένο εύρος ή αριθμό από servers, καθώς οι πόροι κλιμακώνονται προς τα πάνω και προς τα κάτω αυτόματα για να ανταποκριθούν στην ζήτηση. Να σημειωθεί, ότι παρά το όνομα serverless, εξακολουθούν να χρησιμοποιούνται φυσικοί servers, αλλά οι προγραμματιστές δεν χρειάζεται να ανησυχούν για αυτούς. Το serverless computing είναι μια πλατφόρμα η οποία αποκρύπτει τη χρήση του server από τον προγραμματιστή.

Στις πρώτες μέρες του διαδικτύου, οποιοσδήποτε ήθελε να χτίσει μια web εφαρμογή έπρεπε να διαθέτει το φυσικό hardware που απαιτείται για να τρέξει έναν server, που είναι μια δύσκαμπτη και δαπανηρή διαδικασία.

Έπειτα επινοήθηκε το cloud computing, όπου σταθερός αριθμός από servers ή ποσοστό ενός server μπορούν να ενοικιαστούν από απόσταση. Οι προγραμματιστές και οι εταιρείες που νοικιάζουν αυτές τις σταθερές μονάδες χώρου στον server, γενικά υπερβάλλουν στις ποσότητες, ώστε να διασφαλίσουν ότι μια αύξηση της κίνησης ή της δραστηριότητας δεν θα υπερβεί τα μηνιαία όριά τους και δεν θα προκαλέσει πρόβλημα στις εφαρμογές τους. Αυτό σημαίνει ότι μεγάλο μέρος του χώρου του server που έχουν πληρώσει μπορεί να μείνει ανεκμετάλλευτο. Οι προμηθευτές cloud τεχνολογιών εισήγαγαν αυτόματα κλιμακούμενα μοντέλα για να διαχειριστούν αυτό το ζήτημα, αλλά ακόμα κι έτσι μια ανεπιθύμητη αιχμή στην δραστηριότητα, όπως μια επίθεση DDoS, μπορεί να αποβεί πολύ δαπανηρή.



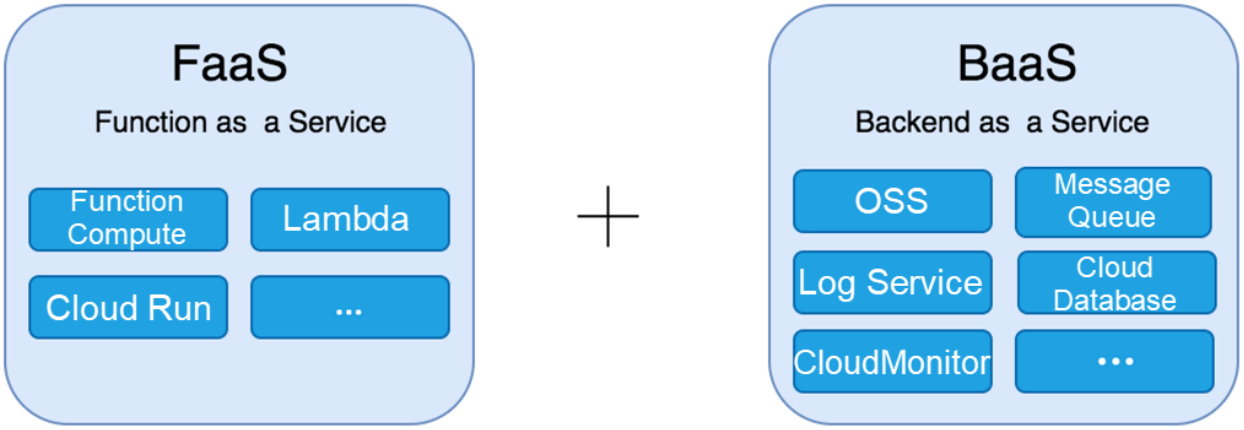
Εικόνα 3. Οφέλη κόστους του serverless computing [48]

Το serverless computing επιτρέπει στους προγραμματιστές να αγοράζουν backend υπηρεσίες με μια ευέλικτη 'pay-as-you-go' βάση, που σημαίνει ότι απαιτείται πληρωμή μόνο για τις υπηρεσίες που χρησιμοποιούν. Ο όρος "serverless" είναι κάπως παραπλανητικός, καθώς εξακολουθούν να υπάρχουν servers που παρέχουν αυτές τις backend υπηρεσίες, αλλά όλα τα προβλήματα του χώρου και της υποδομής του server αντιμετωπίζονται από τον προμηθευτή [48].

Μια serverless πλατφόρμα μπορεί να προσφέρει είτε το μοντέλο Function as a Service (FaaS), είτε το μοντέλο Backend as a Service (BaaS), είτε και τα δύο.

- Function as a Service (FaaS): Το FaaS επιτρέπει στους προγραμματιστές να αναπτύξουν διακριτά block κώδικα για εκτέλεση προς απόκριση συγκεκριμένων συμβάντων χωρίς να ανησυχούν για τη διαχείριση πόρων στο backend. Οι συναρτήσεις αυτές τρέχουν σε έτοιμα περιβάλλοντα εκτέλεσης γλωσσών προγραμματισμού όπως JavaScript, Python κλπ.

- Backend as a Service (BaaS): Την ίδια στιγμή το BaaS παρέχει υπηρεσίες API που προσφέρουν βασική λειτουργικότητα της εφαρμογής. Δεδομένου ότι τα APIs παρέχονται ως υπηρεσία κλιμακούμενη αυτόματα στο παρασκήνιο, χωρίς καμία παρέμβαση από τον προγραμματιστή, οι επιχειρήσεις μπορούν να αναπτύσσουν εφαρμογές με πολύ μεγαλύτερη ταχύτητα και ευκινησία.



Εικόνα 4. Μοντέλα FaaS και BaaS [49]

3.2 Εργαλεία σχεδιασμού και ανάπτυξης

Έχοντας ως στόχο να προσεγγίσουμε μια serverless αρχιτεκτονική για το σύστημά μας, επιλέξαμε για εργαλεία ανάπτυξης της εφαρμογής μας το Flutter SDK για το χτίσιμο κυρίως του UI και το Firebase της Google ως πάροχο υπηρεσιών backend. Το Firebase είναι μια Backend-as-a-Service (BaaS) πλατφόρμα, αλλά συνδυάζει και το μοντέλο Function as a Service (FaaS) χάρη στις Firebase Cloud Functions.

3.2.1 Flutter

Για την ανάπτυξη της εφαρμογής επιλέχθηκε το SDK (Software Development Kit) Flutter. Το Flutter είναι μια πλατφόρμα ανοιχτού κώδικα που δημιουργήθηκε από την Google και αποτελεί την ένωση του UI Framework, με το SDK. Χρησιμοποιείται για την ανάπτυξη διεπαφών χρήστη πολλαπλών πλατφορμών (cross platform) και έχει σχεδιαστεί για να επιτρέπει την επαναχρησιμοποίηση κώδικα σε διάφορα λειτουργικά συστήματα όπως το Android, το iOS, τα Linux, τα Windows και το web, ενώ επιτρέπει επίσης στις εφαρμογές να επικοινωνούν απευθείας με τις υπηρεσίες της υποκείμενης πλατφόρμας. Στόχος είναι να επιτρέπει στους προγραμματιστές να αναπτύσσουν πολύπλοκες εφαρμογές υψηλών επιδόσεων, που μπορούν να εκτελεστούν σε διαφορετικές πλατφόρμες, αφομοιώνοντας τις μεταξύ τους διαφορές, ενώ παράλληλα μοιράζονται μια κοινή βάση κώδικα.

Η πλατφόρμα του Flutter βασίζεται στην γλώσσα προγραμματισμού Dart. Η Dart είναι μια client-optimized γλώσσα προγραμματισμού. Αναπτύσσεται από την Google και χρησιμοποιείται για την κατασκευή mobile, desktop, server και web εφαρμογών. Είναι αντικειμενοστραφής και το συντακτικό της είναι παρόμοιο με αυτό της γλώσσας προγραμματισμού C. Η Dart μπορεί να μεταγλωττιστεί σε native code ή σε JavaScript.

Κατά την διάρκεια της ανάπτυξης, οι Flutter εφαρμογές τρέχουν σε μια εικονική μηχανή (VM) που προσφέρει σταθερή ενσωμάτωση στις αλλαγές (hot reload) χωρίς να απαιτείται διαρκώς πλήρης μεταγλώττιση. Οι Flutter εφαρμογές μεταγλωττίζονται κατευθείαν σε γλώσσα μηχανής, είτε σε Intel x64 ή ARM εντολές, είτε σε Javascript εάν προορίζονται για το web. Η πλατφόρμα είναι ανοιχτού κώδικα, με ανεκτική άδεια BSD, και υποστηρίζεται από μια ταχέως αναπτυσσόμενη συλλογή πακέτων τρίτων που επεκτείνουν την βασική λειτουργικότητα της βιβλιοθήκης [50].

Το Flutter παρέχει πολλά έτοιμα για χρήση widgets (μονάδες σύνθεσης), ικανά να δημιουργήσουν μια όμορφη, μοντέρνα εφαρμογή. Τα πάντα στο Flutter μπορούν να υλοποιηθούν χρησιμοποιώντα widgets. Ο προσανατολισμός, η διάταξη, τα animations, τα κουμπιά κλπ υλοποιούνται με widgets. Πιο συγκεκριμένα και η ίδια η Flutter εφαρμογή είναι ένα widget, το root widget (συνήθως MaterialApp ή CupertinoApp). Η λογική της εφαρμογής στηρίζεται στο reactive programming. Το widget μπορεί προαιρετικά να έχει μια κατάσταση (state). Αλλάζοντας το state του widget, το Flutter θα συγκρίνει αυτόματα (reactive programming) το state του widget (παλιό και νέο) και θα το απεικονίσει μόνο με τις απαραίτητες αλλαγές αντί να επαναπεικονίσει ολόκληρο το widget. Τα widgets σχηματίζουν μια ιεραρχία με βάση τη σύνθεση. Κάθε widget τοποθετείται μέσα στον γονέα του και μπορεί να λαμβάνει το περιβάλλον από αυτόν. Αυτή η δομή μεταφέρεται μέχρι το root widget.

Τρία βασικά χαρακτηριστικά της αρχιτεκτονικής του Flutter που αξίζει να αναφερθούν είναι τα εξής:

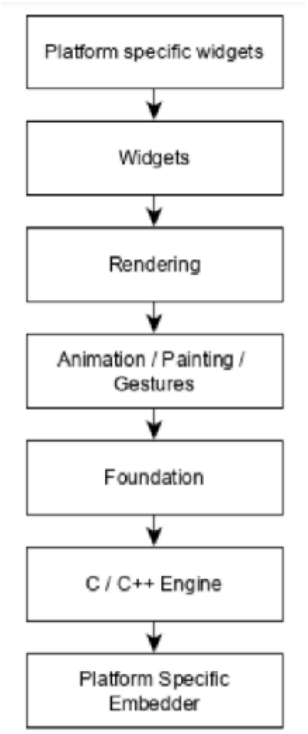
- Gestures: Τα Flutter widgets υποστηρίζουν την αλληλεπίδραση με τον χρήστη μέσω ενός ειδικού widget που ονομάζεται GestureDetector. Το GestureDetector είναι ένα αόρατο widget που έχει την ικανότητα να αντιλαμβάνεται τις κινήσεις του χρήστη (tapping, dragging κλπ), στο παιδί widget. Πολλά native widgets υποστηρίζουν την αλληλεπίδραση μέσω της χρήσης του GestureDetector. Μπορούμε επίσης να

ενσωματώσουμε διαδραστικά χαρακτηριστικά σε ένα υπάρχον widget, συνθέτοντάς το με το GestureDetector widget.

- Concept of State: Τα widgets του Flutter υποστηρίζουν την έννοια του state, παρέχοντας ένα ειδικό widget που ονομάζεται StatefulWidget. Για να μπορεί ένα widget να υποστηρίξει την λογική του state πρέπει να παραχθεί από το StatefulWidget, αλλιώς από το StatelessWidget. Το StatefullWidget θα αυτο-επαναπεικονιστεί οποτεδήποτε το εσωτερικό του state αλλάξει. Η διαδικασία αυτή βελτιστοποιείται εντοπίζοντας τις διαφορές μεταξύ του παλιού και του νέου widget UI και επαναπεικονίζει μόνο τις απαραίτητες αλλαγές.

- Layers: Η πιο σημαντική ιδέα του Flatter framework είναι ότι είναι ξεκάθαρα οργανωμένο σε επίπεδα μειούμενης πολυπλοκότητας. Ένα επίπεδο είναι χτισμένο χρησιμοποιώντας το αμέσως επόμενό του επίπεδο. Το πρώτο στην ιεραρχία είναι το widget που εξαρτάται από την πλατφόρμα (android, iOS κλπ). Το επόμενο είναι το layer που περιλαμβάνει όλα τα widgets του Flutter. Ακολουθεί το Rendering layer, που είναι ένα χαμηλού επιπέδου απεικονιστικό δομικό στοιχείο και απεικονίζει τα πάντα στην Flutter εφαρμογή. Τα επίπεδα συνεχίζουν μέχρι τον πυρήνα του κώδικα της πλατφόρμας.

  Μια γενική επισκόπηση ενός επιπέδου στο Flutter παρουσιάζεται στο παρακάτω διάγραμμα:



Εικόνα 5 Διάγραμμα Flutter Layer [51]

Όπως έχουμε, λοιπόν ήδη αναφέρει το Flutter προσφέρει πολλά ευπαρουσίαστα και προσαρμοζόμενα widgets για την υλοποίηση υψηλής απόδοσης εφαρμογών, καλύπτοντας έτσι όλες τις ανάγκες και τις απαιτήσεις του προγραμματιστή. Πέραν αυτού όμως προσφέρει και πολλά ακόμα πλεονεκτήματα:

- Η Dart διαθέτει ένα εκτεταμένο οικοσύστημα πακέτων τρίτων που επιτρέπουν στον χρήστη να επεκτείνει τις δυνατότητες της εφαρμογής του.
- Οι προγραμματιστές πρέπει να γράψουν μόνο μια κοινή βάση κώδικα για όλες τις πλατφόρμες (Android, iOS, Windows κλπ)
- Το Flutter χρειάζεται λιγότερους ελέγχους (testing). Εξαιτίας της κοινής βάσης κώδικα, αρκεί να δημιουργήσουμε αυτόματα tests κοινά για όλες τις πλατφόρμες.
- Η απλότητα του Flutter το καθιστά καλή επιλογή για γρήγορη ανάπτυξη εφαρμογών. Η προσαρμοστικότητα και η επεκτασιμότητά του το κάνουν ακόμα πιο ισχυρό.
- Με το Flutter, οι προγραμματιστές έχουν τον πλήρη έλεγχο χάρη στα widgets και την δομή που εξασφαλίζει.
- Τέλος, το Flutter παρέχει πλήθος εργαλείων για τον προγραμματιστή καθώς και την εξαιρετικά χρήσιμη δυνατότητα για αυτόματη ενσωμάτωση των αλλαγών (hot reload).

Πέραν των πολλών πλεονεκτημάτων του, όμως, το Flutter έχει και ορισμένα μειονεκτήματα:

- Από την στιγμή που ο προγραμματισμός γίνεται στην γλώσσα Dart, ο προγραμματιστής καλείται να μάθει μια καινούργια γλώσσα. Παρόλα αυτά είναι μια εύκολη γλώσσα στην εκμάθησή της.
- Τα μοντέρνα frameworks προσπαθούν να ξεχωρίσουν την λογική με το UI όσο περισσότερο μπορούν. Στο Flutter, η διεπαφή χρήστη και η λογικη αναμειγνύονται. Αυτό μπορεί να ξεπεραστεί χρησιμοποιώντας έξυπνο κώδικα και υψηλού επιπέδου modules για να ξεχωρίσουμε την διεπαφή του χρήστη από την λογική.

3.2.2 Firebase

Το Firebase είναι μια πλατφόρμα που αναπτύχθηκε από την Google για τη δημιουργία mobile και web εφαρμογών. Είναι μια Backend-as-a-Service (BaaS) πλατφόρμα ανάπτυξης εφαρμογών που παρέχει backend services όπως realtime database, cloud storage, authentication, crash reporting, machine learning, remote configuration, and hosting για τα στατικά αρχεία. Αρχικά ήταν μια ανεξάρτητη εταιρεία που ιδρύθηκε το 2011. Το 2014, η Google απέκτησε την πλατφόρμα και είναι πλέον η ναυαρχίδα της για ανάπτυξη εφαρμογών. Η πλατφόρμα διαθέτει 18 προϊόντα ταξινομημένα στις εξής κατηγορίες: Build, Release & Monitor και Engage. Αναλυτικότερα στον παρακάτω πίνακα:

Build	**Release & Monitor**	**Engage**
Realtime Database	Google Analytics	Google Analytics
Remote Config	Remote Config	Predictions
Firebase ML	Performance Monitoring	A/B Testing
Cloud Functions	Test Lab	Authentication
Authentication	App Distribution	Cloud Messaging
Cloud Messaging		Crashlytics
Hosting		Dynamic Links
Cloud Storage		In-App Messaging

Πίνακας 2. Προϊόντα Firebase

Για την ανάπτυξη της εφαρμογής χρησιμοποιήσαμε τα παρακάτω προϊόντα του Firebase:

- Firebase Authentication: Οι περισσότερες εφαρμογές χρειάζεται να γνωρίζουν την ταυτότητα του χρήστη. Γνωρίζοντας την ταυτότητα του χρήστη μια εφαρμογή μπορεί να αποθηκεύσει με ασφάλεια τα δεδομένα του στο cloud και να του παρέχει προσωποποιημένη εμπειρία σε όλες τις συσκευές του. Το Firebase Authentication παρέχει backend υπηρεσίες, εύχρηστα SDKs και προκατασκευασμένες UI βιβλιοθήκες για την ταυτοποίηση των χρηστών της εφαρμογής. Υποστηρίζει ταυτοποίηση χρησιμοποιώντας passwords, αριθμούς τηλεφώνων και δημοφιλείς παρόχους ομόσπονδης ταυτότητας όπως το Google, το Facebook, το Twitter κλπ. Το Firebase Authentication εντάσσεται και σε άλλες υπηρεσίες του Firebase και αξιοποιεί βιομηχανικά πρότυπα όπως το OAuth 2.0 και το OpenID Connect, ώστε να μπορεί εύκολα να ενσωματωθεί στο backend της εφαρμογής.

- Firebase Realtime Database: Η Firebase Realtime Database είναι μια μη σχεσιακή (NoSQL), cloud-hosted βάση δεδομένων. Τα δεδομένα αποθηκεύονται σε μορφή JSON και συγχρονίζονται σε πραγματικό χρόνο με κάθε συνδεδεμένο κόμβο (client). Όλοι οι κόμβοι μοιράζονται ένα κοινό στιγμιότυπο της Realtime Database και αυτόματα λαμβάνουν ενημερώσεις με τα νέα δεδομένα. Τα δεδομένα παραμένουν προσβάσιμα ακόμα και όταν η εφαρμογή είναι εκτός σύνδεσης (offline).

- Firebase Cloud Firestore: Η Firebase Cloud Firestore είναι μια ευέλικτη, επεκτάσιμη, μη σχεσιακή, cloud-hosted βάση δεδομένων για την αποθήκευση και τον συγχρονισμό δεδομένων για client-and server-side ανάπτυξη. Όπως και η Firebase Realtime Database, κρατάει τα δεδομένα συγχρονισμένα ανάμεσα στους κόμβους (clients) μέσω realtime listeners και παρέχει υποστήριξη εκτός σύνδεσης λειτουργίας, ώστε να μπορούν να χτιστούν εφαρμογές με καλή ανταπόκριση που θα μπορούν να λειτουργούν ανεξαρτήτως της σύνδεσης στο διαδίκτυο. Το Cloud Firestore επίσης, παρέχει απρόσκοπτη ενσωμάτωση με άλλα προϊόντα του Firebase και της Google, συμπεριλαμβανομένων και των Cloud Functions.

- Firebase Cloud Storage: Το Firebase Cloud Storage χτίστηκε για προγραμματιστές που χρειάζεται να αποθηκεύσουν και να διαμοιραστούν περιεχόμενο προερχόμενο από τους χρήστες, όπως φωτογραφίες ή βίντεο. Είναι μια ισχυρή, απλή και χαμηλού κόστους υπηρεσία αποθήκευσης. Οι Firebase εφαρμογές μπορούν μέσω αυτής να ανεβάζουν ή να κατεβάζουν αρχεία ανεξαρτήτως της ποιότητας σύνδεσης στο διαδίκτυο.

- Firebase Cloud Functions: Οι Firebase Cloud Functions είναι ένα serverless framework που επιτρέπει στον προγραμματιστή να τρέξει αυτόματα backend κώδικα ως απάντηση σε events που πυροδοτούνται από Firebase λειτουργίες και αιτήματα HTTPS. Ο JavaScript ή TypeScript κώδικας είναι αποθηκευμένος στο cloud της Google και τρέχει σε ένα διαχειριζόμενο περιβάλλον. Έτσι δεν χρειάζεται να διαχειριστεί ή να επεκτείνει τους δικούς του servers ο χρήστης.

- Firebase Cloud Messaging (FCM): Το Firebase Cloud Messaging χρησιμοποιείται για την αξιόπιστη αποστολή μηνυμάτων χωρίς κόστος. Με το FCM, μπορούμε να ενημερώσουμε μία εφαρμογή ότι ένα νέο email ή άλλα δεδομένα είναι διαθέσιμα για συγχρονισμό. Ακόμα, μπορούμε να στείλουμε ειδοποιήσεις μεταξύ των διαφόρων χρηστών και συσκευών. Το μέγεθος το μηνυμάτων που μπορεί να μεταφέρει είναι μέχρι 4KB. Το σύστημα νέφους του Firebase (Firebase cloud system) παρέχει κρυπτογράφηση SSL για τη μετάδοση των δεδομένων.

Κεφάλαιο 4: Μεθοδολογία και σχεδιασμός εφαρμογής MedImg_App

4.1 Εισαγωγή

Σε πρώτη φάση αποφασίσαμε να αναπτύξουμε την δική μας βάση με εικόνες συσκευασιών φαρμάκων, καθώς η μόνη πηγή που υπάρχει για την ελληνική αγορά είναι η πλατφόρμα επιστημονικής πληροφόρησης για τα φάρμακα ανθρώπινης χρήσης "Γαληνός Οδηγός Φαρμάκων" [52]. Στον Γαληνό υπάρχουν εικόνες για το ¼ περίπου των σκευασμάτων που κυκλοφορούν στην ελληνική αγορά. Για να μπορέσει να πραγματοποιηθεί η παραγωγή μιας τέτοιας βάσης δημιουργήσαμε μια εφαρμογή με το όνομα MedImg_App, που δίνει την δυνατότητα στον χρήστη να αναζητήσει το barcode του φαρμάκου και αν δεν υπάρχει ήδη καταχωρημένη εικόνα, να τραβήξει ο ίδιος μια φωτογραφία με το κινητό του, να την επεξεργαστεί και να την ανεβάσει στην βάση. Η εφαρμογή MedImg_App προορίζεται για επιλεγμένους χρήστες, όπως φαρμακοποιούς ή εθελοντές σε κοινωνικά φαρμακεία.

4.2 Δομή εφαρμογής

4.2.1 Βάση Δεδομένων

4.2.1.1 Realtime Database

Αρχικά, δημιουργήσαμε ένα νέο Firebase project με το όνομα NTUA Thesis. Στη συνέχεια, έπρεπε να δημιουργηθεί μια βάση δεδομένων με τις βασικές πληροφορίες για κάθε φάρμακο. Τα περισσότερα δεδομένα που χρειαζόμασταν προέκυψαν από το excel που συνόδευε την απόφαση του Υπουργείου Υγείας, με θέμα «Επικαιροποιημένο Δελτίο Τιμών Φαρμάκων Ανθρώπινης Χρήσης με ενσωμάτωση: του Δελτίου Τιμών Νέων Φαρμάκων έτους 2019, διοικητικών μεταβολών, αλλαγής Φ.Π.Α. και αναπροσαρμογής τιμών για λόγους Δημόσιας Υγείας» που δημοσιεύτηκε στις 20-03-2020 .

Για να μπορέσουμε εύκολα να δημιουργήσουμε μια Realtime Database στο Firebase, ανεβάσαμε το excel στο Google Drive και το μετατρέψαμε σε Google Sheet. Αφαιρέσαμε τις στήλες που δεν χρειαζόμασταν (τιμή, ΦΠΑ κλπ), αλλάξαμε τα ονόματα των στηλών και προσθέσαμε τρεις νέες στήλες:

- ntua_img : το link για τις εικόνες της δικής μας βάσης
- pill_img : "user" εάν η εικόνα του ntua_img έχει καταχωρηθεί από απλό χρήστη της εφαρμογής MedCurie (Κεφάλαιο 5) και "pharm" εάν έχει καταχωρηθεί από φαρμακοποιό μέσω της παρούσας εφαρμογής MedImg_App.
- gal_img: το link για την εικόνα από τον Γαληνό αν υπάρχει

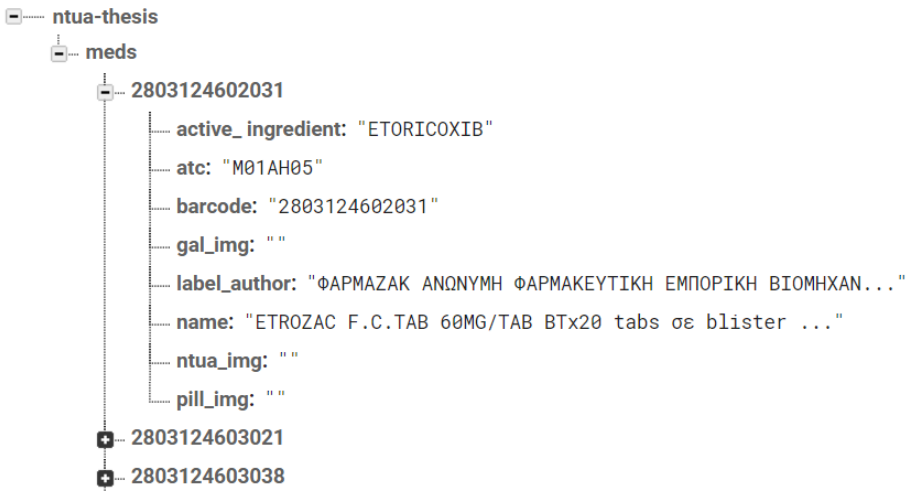
Τέλος, αλλάξαμε τα read/write permissions των rules της βάσης σε true και χρησιμοποιήσαμε το εργαλείο "Πρόγραμμα Επεξεργασίας Σεναρίων" των Google Sheets για να γράψουμε και να εκτελέσουμε τον κώδικα που δημιούργησε την βάση με τα δεδομένα του excel. Το script φαίνεται παρακάτω:

```
function getEnvironment() {
 var environment = {
   spreadsheetID: "1aZKQhytcvu-gMG89Y1LNMiMdDCi6V7IDgdPNye7ZdVY",
   firebaseUrl: "https://ntua-thesis.firebaseio.com/"
```

```
 };
 return environment;
}

// Creates a Google Sheets on change trigger for the specific sheet
function createSpreadsheetEditTrigger(sheetID) {
 var triggers = ScriptApp.getProjectTriggers();
 var triggerExists = false;
 for (var i = 0; i < triggers.length; i++) {
   if (triggers[i].getTriggerSourceId() == sheetID) {
     triggerExists = true;
     break;
   }
 }

 if (!triggerExists) {
   var spreadsheet = SpreadsheetApp.openById(sheetID);
   ScriptApp.newTrigger("importSheet")
     .forSpreadsheet(spreadsheet)
     .onChange()
     .create();
 }
}

// Delete all the existing triggers for the project
function deleteTriggers() {
 var triggers = ScriptApp.getProjectTriggers();
 for (var i = 0; i < triggers.length; i++) {
   ScriptApp.deleteTrigger(triggers[i]);
 }
}

// Initialize
function initialize(e) {
 writeDataToFirebase(getEnvironment().spreadsheetID);
}

// Write the data to the Firebase URL
function writeDataToFirebase(sheetID) {
 var ss = SpreadsheetApp.openById(sheetID);
 SpreadsheetApp.setActiveSpreadsheet(ss);
 createSpreadsheetEditTrigger(sheetID);
 var sheets = ss.getSheets();
 for (var i = 0; i < sheets.length; i++) {
   importSheet(sheets[i]);
   SpreadsheetApp.setActiveSheet(sheets[i]);
 }
}

// A utility function to generate nested object when
// given a keys in array format
function assign(obj, keyPath, value) {
 lastKeyIndex = keyPath.length - 1;
 for (var i = 0; i < lastKeyIndex; ++i) {
   key = keyPath[i];
   if (!(key in obj)) obj[key] = {};
   obj = obj[key];
 }
 obj[keyPath[lastKeyIndex]] = value;
}

// Import each sheet when there is a change
function importSheet() {
 var sheet = SpreadsheetApp.getActiveSheet();
```

```
 var name = sheet.getName();
 var data = sheet.getDataRange().getValues();

 var dataToImport = {};

 for (var i = 1; i < data.length; i++) {
   dataToImport[data[i][0]] = {};
   for (var j = 0; j < data[0].length; j++) {
     assign(dataToImport[data[i][0]], data[0][j].split("__"), data[i][j]);
   }
 }

 var token = ScriptApp.getOAuthToken();

 var firebaseUrl = getEnvironment().firebaseUrl + name;
 var base = FirebaseApp.getDatabaseByUrl(firebaseUrl, token);
 base.setData("", dataToImport);
}
```

Οπότε, το json tree της βάσης διαμορφώθηκε, όπως φαίνεται στην παρακάτω εικόνα:



Εικόνα 6. Json tree της Realtime database με τις πληροφορίες των φαρμακευτικών σκευασμάτων που κυκλοφορούν στην ελληνική αγορά

4.2.1.2 Firebase Storage

Μέσω της κονσόλας του Firebase δημιουργήσαμε στο Storage έναν φάκελο με το όνομα meds. Εδώ αποθηκεύονται οι εικόνες των φαρμάκων με όνομα <barcode>.png

4.2.2 Authentication

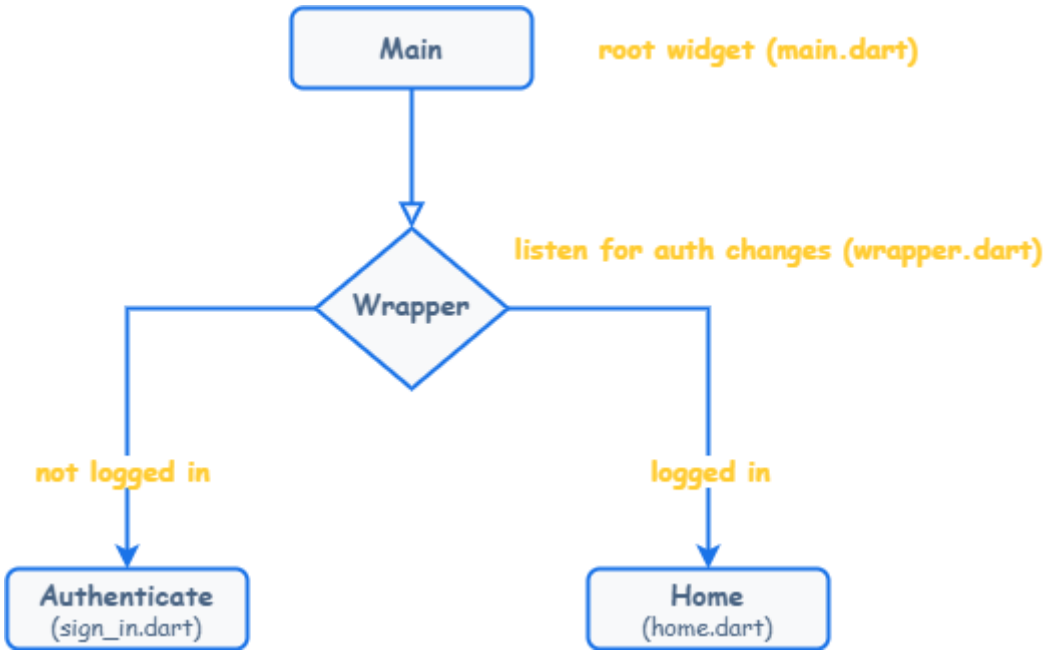
Μέσω της κονσόλας του Firebase επιλέξαμε ως Sign-in method το Email/Password. Στην συνέχεια, τροποποίησαμε τα rules για την Realtime Database και το Storage, ώστε να έχουν πρόσβαση μόνο οι χρήστες που έχουν κάνει login.

Realtime Database	Storage

{
"rules": {
".read": "auth.uid != null",
".write": "auth.uid != null"
}
}
	rules_version = '2';
	service firebase.storage {
	match /b/{bucket}/o {
	match /{allPaths=**} {
	allow read, write: if request.auth != null;
	}
	}
	}

Πίνακας 3. Realtime Database and Storage rules

Το flow του authentication των χρηστών είναι αυτό που φαίνεται στο παρακάτω λογικό διάγραμμα:

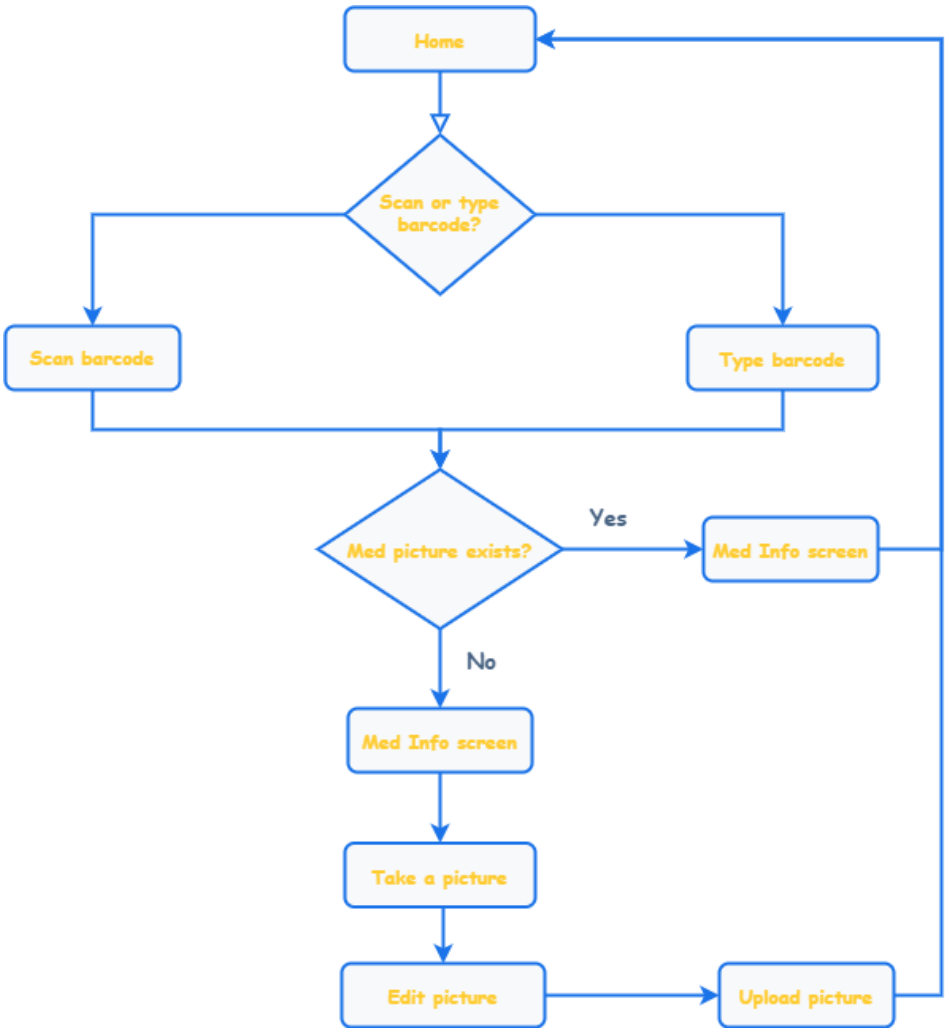


Εικόνα 7. Authentication Flow της εφαρμογής MedImg_App

4.2.3 App Flow

Στο παρακάτω λογικό διάγραμμα παρουσιάζεται η δομή της εφαρμογής, αφού έχει προηγηθεί το authentication του χρήστη. Ο χρήστης ανάλογα με τις επιλογές του, αλλά και με το αν το φάρμακο που έχει επιλέξει έχει ήδη εικόνα στην βάση μας, πλοηγείται στις οθόνες της εφαρμογής όπως φαίνεται παρακάτω. Εν συντομία η ροή εξελίσσεται ως εξής:

- Ο χρήστης επιλέγει αν θα σκανάρει ή θα πληκτρολογήσει το barcode.
- Το φάρμακο εντοπίζεται στην βάση και ανακαλούνται οι πληροφορίες που έχουμε σχετικά με αυτό.
- Αν το φάρμακο που αναζήτησε έχει ήδη εικόνα επιστρέφει στην αρχική για να αναζητήσει κάποιο άλλο, ενώ αν δεν έχει του ζητείται να χρησιμοποιήσει την κάμερα της συσκευής του για να καταχωρήσει μια.
- Αφού τραβήξει την φωτογραφία του δίνεται η δυνατότητα να την μορφοποιήσει ώστε να διορθώσει την εμφάνισή της.
- Όταν ο χρήστης δώσει την έγκρισή του ότι η εικόνα είναι έτοιμη ώστε να ανέβει στην βάση τότε η φωτογραφία συμπιέζεται από την εφαρμογή ώστε να καταλαμβάνει λιγότερο χώρο μιας και ο χώρος που δίνει το firebase στο storage χρεώνεται και δεν είναι αναγκαία η υψηλή ανάλυση για τους σκοπούς της δικής μας εφαρμογής. Τέλος, προστίθεται αυτόματα από την εφαρμογή στο κάτω δεξιά μέρος της εικόνας το logo του ΕΜΠ ως watermark για λόγους κατοχύρωσης της πηγής της.



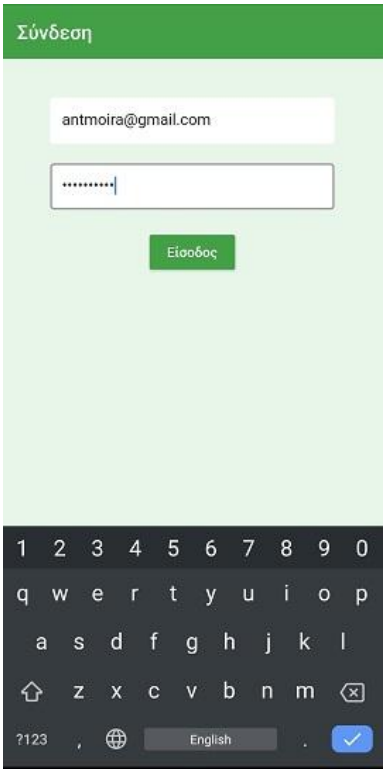
Εικόνα 8. App Flow της εφαρμογής MedImg_App

4.3 Οθόνες εφαρμογής

Σε αυτή την ενότητα παρουσιάζονται αναλυτικά οι βασικές οθόνες της εφαρμογής, ώστε να δώσουμε μια ολοκληρωμένη εικόνα των λειτουργιών της τόσο θεωρητικά όσο και οπτικά. Ακολουθούν οι εικόνες και οι περιγραφές τους.

4.3.1 Οθόνη Σύνδεσης

Μπαίνοντας στην εφαρμογή ο φαρμακοποιός καλείται να εισάγει στην φόρμα (Εικόνα 9) το Email του και το Password που του έχει στείλει ο διαχειριστής της εφαρμογής μας, ώστε να μπορέσει να συνδεθεί και να προχωρήσει στην διαδικασία της καταχώρησης φωτογραφιών για τα φαρμακευτικά σκευάσματα. Πατώντας το κουμπί "Είσοδος", εάν τα στοιχεία που έχει δώσει είναι σωστά μεταφέρεται στην αρχική οθόνη της εφαρμογής.



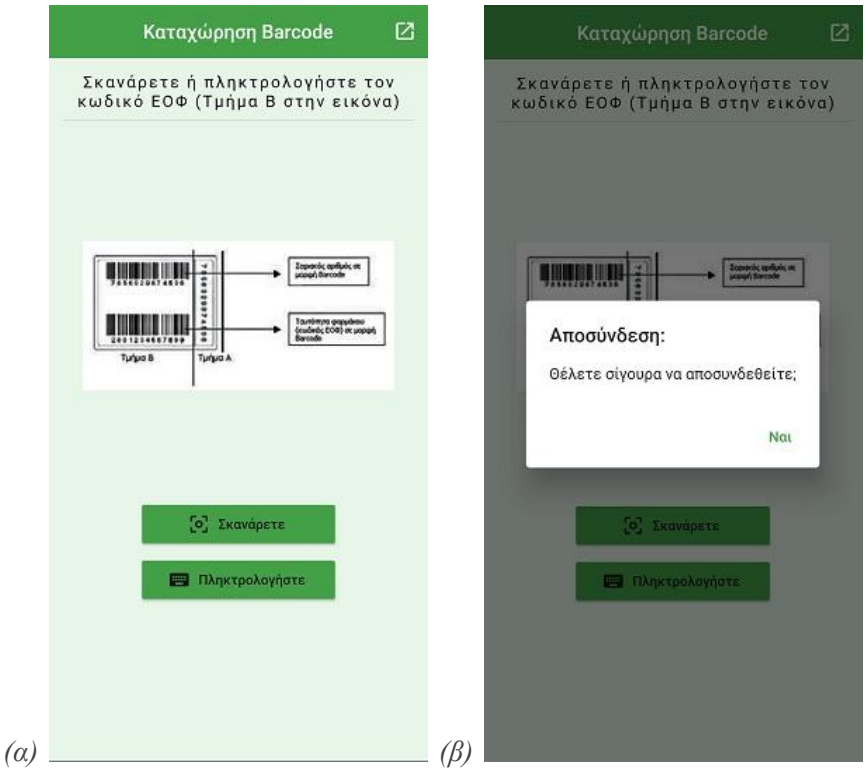
Εικόνα 9. Οθόνη σύνδεσης

4.3.2 Αρχική Οθόνη & Αποσύνδεση

Αφού ο φαρμακοποιός ολοκληρώσει την διαδικασία της σύνδεσης μεταφέρεται στην αρχική οθόνη της εφαρμογής (Εικόνα 10α). Εκεί του δίνεται η δυνατότητα να επιλέξει με ποιό τρόπο θα προχωρήσει στην αναζήτηση του σκευάσματος για το οποίο επιθυμεί να καταχωρήσει φωτογραφία. Του δίνονται οι εξής δύο επιλογές:

- Σκανάρισμα του barcode: Πατώντας στο κουμπί “Σκανάρετε” ο χρήστης μπορεί να σκανάρει το barcode της συσκευασίας, προκειμένου να εντοπιστεί το φάρμακο στην βάση μας.
- Πληκτρολόγηση του barcode: Πατώντας στο κουμπί “Πληκτρολογήστε” ζητείται από τον χρήστη να πληκτρολογήσει το barcode του σκευάσματος, ώστε να γίνει η αναζήτηση του σκευάσματος στην βάση μας.

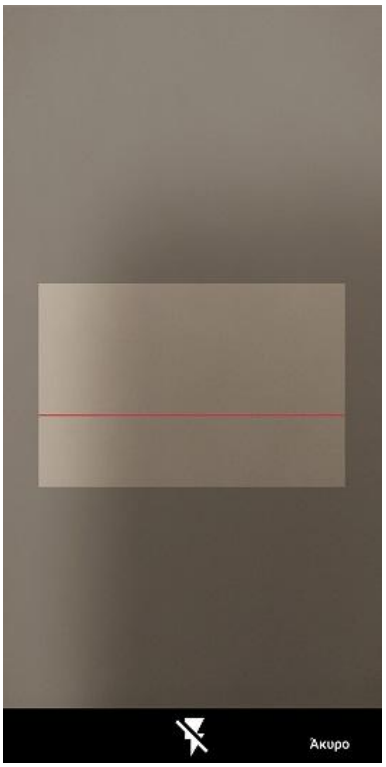
Τέλος, ο χρήστης μπορεί να αποσυνδεθεί σε περίπτωση που το επιθυμεί πατώντας το launch icon στα δεξιά του AppBar. Αφού του ζητηθεί να επιβεβαιώσει την πρόθεσή του (Εικόνα 10β) μεταφέρεται στην οθόνη σύνδεσης.

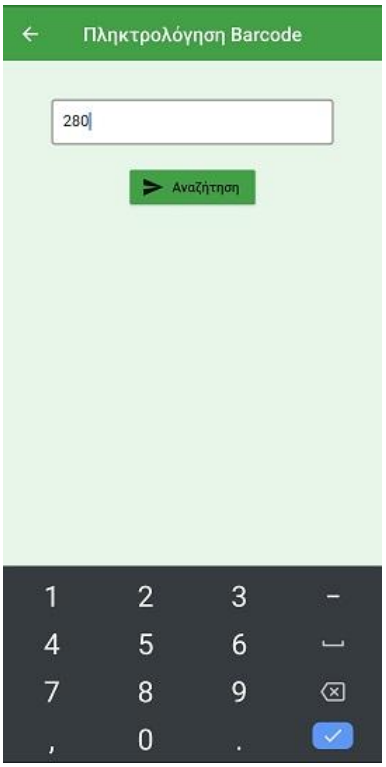


Εικόνα 10. (α) Αρχική οθόνη και (β)Οθόνη αποσύνδεσης

4.3.3 Οθόνες καταχώρησης barcode

Αναλόγως, την επιλογή που θα κάνει ο χρήστης στην αρχική οθόνη της εφαρμογής, δηλαδή αν επιθυμεί να σκανάρει ή να πληκτρολογήσει το barcode της συσκευασίας θα οδηγηθεί στην οθόνη σκαναρίσματος (Εικόνα 11α) ή στην οθόνη πληκτρολόγησης barcode (Εικόνα 11β) αντίστοιχα. Στην πρώτη περίπτωση μπορεί μέσω της κάμερας του κινητού του να σκανάρει το 13ψήφιο barcode που ξεκινάει με τα ψηφία 280 για να γίνει η αναζήτηση του σκευάσματος στην realtime database μας. Ενώ στην δεύτερη περίπτωση καλείται να πληκτρολογήσει τα δεκατρία ψηφία του barcode (αν πληκτρολογήσει λάθος αριθμό ψηφίων θα του εμφανιστεί μήνυμα λάθους) και πατώντας το κουμπί “Αναζήτηση” να γίνει η αναζήτηση στην βάση.

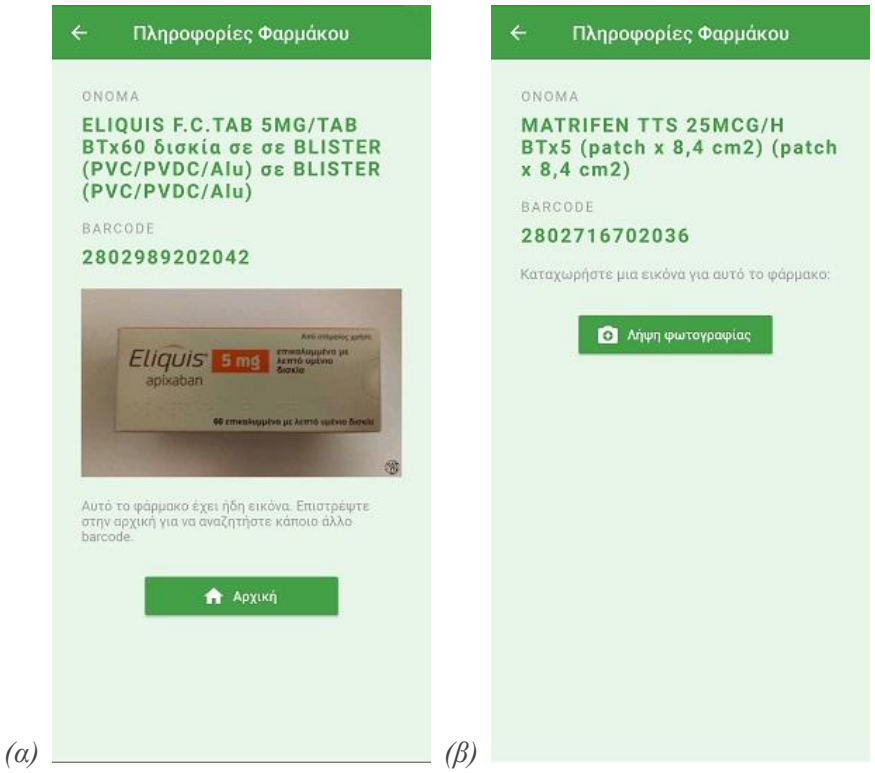
(α) 

(β) 

Εικόνα 11. (α) Οθόνη σκαναρίσματος barcode και (β)Οθόνη πληκτρολόγησης barcode

4.3.4 Οθόνη πληροφοριών φαρμάκου

Όταν για το barcode που αναζητεί ο χρήστης υπάρχει ήδη καταχωρημένη εικόνα στην βάση μας, εμφανίζεται η οθόνη πληροφοριών φαρμάκου με το όνομα του φαρμάκου, το barcode του και την εικόνα που ήδη υπάρχει (Εικόνα 12α). Σε αυτή την περίπτωση ο χρήστης πατώντας το κουμπί "Αρχική" μπορεί να επιστρέψει στην αρχική οθόνη και να αναζητήσει ένα άλλο barcode. Στην περίπτωση που δεν έχουμε εικόνα για αυτό το φάρμακο (Εικόνα 12β) ο χρήστης μπορεί πατώντας το κουμπί "Λήψη φωτογραφίας" να προχωρήσει στην διαδικασία καταχώρησης μιας φωτογραφίας για αυτό το φάρμακο.



(α) *(β)*

Εικόνα 12. (α) Οθόνη πληροφοριών όταν υπάρχει ήδη εικόνα για αυτό το barcode στην βάση μας και (β) Οθόνη πληροφοριών όταν δεν υπάρχει εικόνα για αυτό το φάρμακο

4.3.5 Οθόνες λήψης φωτογραφίας

Όταν ο χρήστης αρχίσει την διαδικασία καταχώρησης μιας φωτογραφίας για ένα σκεύασμα ενεργοποιείται αυτόματα η κάμερα της συσκευής του και μεταφέρεται στην οθόνη λήψης της φωτογραφίας (Εικόνα 13α). Εκεί αφού εστιάσει στην συσκευασία πατώντας το floatingActionButton κάτω δεξιά στην οθόνη αποθηκεύεται προσωρινά στην συσκευή του η φωτογραφία και μεταφέρεται στην οθόνη επεξεργασίας αυτής (Εικόνα 13β). Στην οθόνη αυτή μπορεί να εκτελέσει μια απο τις εξής διορθωτικές ενέργειες στην φωτογραφία που τράβηξε:

- Περικοπή: Μπορεί να περικόψει την φωτογραφία ρυθμίζοντας ανάλογα τα μπλε πλαίσια στις γωνίες της.
- Flip: Όπου μπορεί να αναποδογυρίσει την εικόνα πατώντας το πρώτο icon του μενού στο κάτω μέρος της οθόνης.
- Rotate Left: Μπορεί να την περιστρέψει προς τα αριστερά πατώντας το δεύτερο icon.
- Rotate Right: Να την περιστρέψει δεξιά με το τρίτο Icon.
- Reset: Να αναιρέσει οποιαδήποτε αλλαγή έχει κάνει και έχει μετανιώσει πατώντας το τέταρτο icon button στο μενού.

Τέλος, πατώντας ο χρήστης το tick icon στα δεξιά του AppBar αποθηκεύει την εικόνα στο storage του Firebase με το όνομα <barcode>.png, αφού πρώτα συμπιεστεί για αν μειωθεί το μέγεθός της και μπει στην κάτω δεξιά γωνία της το logo του ΕΜΠ ως watermark. Ταυτόχρονα ενημερώνεται το πεδίο ntua_img στην Realtime Database για το συγκεκριμένο φάρμακο με το link που οδηγεί στην εικόνα και το pill_img ορίζεται ως “pharm”.

(α) 

(β) 

Εικόνα 13. (α) Οθόνη λήψης φωτογραφίας και (β) Οθόνη επεξεργασίας φωτογραφίας

4.3.6 Οθόνη επιτυχούς καταχώρησης

Αφού ολοκληρωθεί η διαδικασία που προαναφέραμε ο χρήστης βλέπει στην τελική οθόνη (Εικόνα 14) την φωτογραφία που καταχώρησε για το συγκεκριμένο barcode και πατώντας το κουμπί “Αρχική” επιστρέφει στην αρχική οθόνη για να προχωρήσει στην επόμενη καταχώρηση για κάποιο άλλο σκεύασμα.

Εικόνα 14. Οθόνη επιτυχούς καταχώρησης

4.4 Πακέτα (packages) & Βιβλιοθήκες

Το Flutter υποστηρίζει τη χρήση κοινών πακέτων (shared packages) τα οποία συνεισφέρονται από προγραμματιστές στα οικοσυστήματα Flutter και Dart. Αυτό επιτρέπει τη γρήγορη δημιουργία μιας εφαρμογής χωρίς να χρειάζεται να αναπτυχθούν τα πάντα από το μηδέν. Στην συνέχεια της ενότητας αυτής ακολουθεί μια περιγραφή των πακέτων που χρησιμοποιήσαμε για την ανάπτυξη της εφαρμογής MedImg_App και των λειτουργιών που αναπτύξαμε με βάση αυτά.

Αρχικά για την ταυτοποίηση του χρήστη (user authentication) χρησιμοποιήσαμε το πακέτο “firebase_auth”, προκειμένου να υλοποιήσουμε την μέθοδο ταυτοποίησης “email/password” που παρέχει η πλατφόρμα του Firebase. Το Firebase υποστηρίζει επίσης έλεγχο ταυτότητας με χρήση αριθμών τηλεφώνου, καθώς και μέσω δημοφιλών παρόχων όπως το Google, το Facebook, η Microsoft κ.ά αξιοποιώντας το πρωτόκολλο διαδικτύου OAuth. Επίσης, με το πακέτο “firabase_database” μπορέσαμε να έχουμε πρόσβαση στην Realtime Database “meds”, με τις πληροφορίες για κάθε φάρμακο, που έχει ήδη αναφερθεί στην ενότητα 4.2.1.1. Τέλος, για να χρησιμοποιήσουμε το Firebase Cloud Storage API και να μπορεί ο χρήστης να ανεβάσει τις φωτογραφίες των φαρμάκων στον φάκελο meds, εντάξαμε στην εφαρμογή μας το πακέτο “firebase_storage”.

Για να μπορεί ο χρήστης να χρησιμοποιήσει την κάμερα και να τραβήξει φωτογραφίες εντός της εφαρμογής έγινε χρήση των πακέτων “camera”, “path_provider” και “path”. Ενώ για να του δώσουμε την δυνατότητα να επεξεργαστεί την φωτογραφία προτού την ανεβάσει επιλέξαμε το πακέτο “image_editor”, καθώς και κάποια επιπλέον βοηθητικά αυτού όπως τα “extended_image”, “image”, “isolate” και “oktoast”. Τέλος, για ό,τι αφορά την φωτογραφία του φαρμάκου, χρησιμοποιήσαμε το πακέτο “flutter_image_compress” για να μειώσουμε το μέγεθός της (quality: 75%) και να καταλαμβάνει λιγότερο χώρο στο Firebase Storage.

Επιπλέον, σημαντικό είναι να αναφερθεί ότι η δυνατότητα που έχει ο χρήστης να σκανάρει το barcode του φαρμάκου υλοποιήθηκε μέσω του πακέτου “flutter_barcode_scanner”.

Όλα τα παραπάνω πακέτα δηλώθηκαν και εγκαταστάθηκαν μέσω του αρχείου “pubspec.yaml” του κώδικα της εφαρμογής, όπως φαίνεται και στην Εικόνα 15.

Εικόνα 15. Τα πακέτα που χρησιμοποιήθηκαν για την ανάπτυξη της εφαρμογής, όπως φαίνονται στο αρχείο ‘pubspec.yaml’ [53]

Κεφάλαιο 5: Μεθοδολογία και σχεδιασμός εφαρμογής MedCurie

5.1 Εισαγωγή

Αφού υλοποιήθηκε η εφαρμογή που θα χρησιμοποιηθεί για να χτίσει την βάση με τις εικόνες των φαρμάκων, προχωρήσαμε στην ανάπτυξη της εφαρμογής που στοχεύει στην βελτίωση της φαρμακευτικής συμμόρφωσης του ασθενή. Ο βασικός στόχος της εφαρμογής είναι να βοηθήσει τον ασθενή να λαμβάνει στην ώρα του το σωστό φάρμακο και στην σωστή δόση. Αυτό προσπαθούμε να το πετύχουμε στέλνωντας στον χρήστη ειδοποιήσεις μέσω της εφαρμογής και χρησιμοποιώντας ένα εργαλείο από την εργαλειοθήκη του Firebase, το Firebase Cloud Messaging (FCM). Καθώς, όμως το πρόβλημα της φαρμακευτικής συμμόρφωσης, αλλά και γενικότερα η διαχείριση της υγείας του ασθενή, είναι πιο σύνθετο και εξαρτάται από πολλούς παράγοντες, προσθέσαμε τις εξής επιπλέον λειτουργικότητες στην εφαρμογή ώστε να διευκολύνουμε την καθημερινότητά του:

- Μετρήσεις: Όπου μπορεί να αποθηκεύει μετρήσεις (αρτηριακή πίεση, γλυκόζη αίματος και θερμοκρασία), ώστε να παρακολουθεί την πορεία της υγείας του σε σχέση με αυτές τις βασικές παραμέτρους και το πώς ανταποκρίνεται στην φαρμακευτική του αγωγή. Επίσης, συχνά ζητείται από τους ιατρούς οι ασθενείς να κάνουν καταγραφή για ένα χρονικό διάστημα και να τους την παρουσιάσουν στο επόμενο ραντεβού, οπότε σε αυτή την περίπτωση αυτή η λειτουργία θα διευκολύνει την διαδικασία για τον ασθενή.
- Σημειώσεις: Ώστε να μπορεί ο ασθενής να κρατάει σημειώσεις σχετικά με τα συμπτώματά του ή απορίες που δεν θα ήθελε να παραλείψει να αναφέρει ή να ρωτήσει στο επόμενο ραντεβού με τον ιατρό του.
- Ιατροί: Εδώ μπορεί να κρατάει έναν κατάλογο με τα στοιχεία όλων των ιατρών του ώστε να μπορεί να επικοινωνήσει μαζί τους εύκολα όταν το χρειαστεί.
- Φροντιστές: Με την ίδια λογική, εδώ αποθηκεύει τα στοιχεία των ανθρώπων που τον φροντίζουν για να μπορεί να τα βρει εύκολα αν υπάρξει ανάγκη.
- Συνταγές: Με αυτή την λειτουργία ο χρήστης αποθηκεύει όλες τις συνταγές που του έχουν γράψει οι ιατροί του, είτε σχετικά με την ανανέωση του αποθέματος των φαρμάκων του, είτε με διαγνωστικές εξετάσεις. Έτσι, πάλι μέσω ειδοποιήσεων που έχουμε προγραμματίσει να του αποστέλλονται την ημέρα έναρξης και την ημέρα λήξης της συνταγής του ο χρήστης δεν κινδυνεύει να χάσει τις σχετικές προθεσμίες.
- Ραντεβού: Τέλος, δίνεται η δυνατότητα στον χρήστη να αποθηκεύσει εδώ τα ραντεβού με τους θεράποντες ιατρούς του ή με διαγνωστικά κέντρα, ώστε να του αποστέλλεται ειδοποίηση μια μέρα πριν για να του υπενθυμίσει το ραντεβού του.

Στις επόμενες ενότητες παρουσιάζεται αναλυτικά η υλοποίηση των παραπάνω λειτουργιών, τα εργαλεία που χρησιμοποιήθηκαν και η μεθοδολογία που ακολουθήθηκε.

5.2 Δομή εφαρμογής

5.2.1 Authentication

Και στην εφαρμογή MedCurie, όπως και στην MedImg_App (Ενότητα 4.2.2), χρησιμοποιήθηκε η μέθοδος Email/Password του Firebase για την ταυτοποίηση του χρήστη. Σε αυτή την εφαρμογή όμως, δίνεται η δυνατότητα στους χρήστες να κάνουν μόνοι τους εγγραφή και να δημιουργήσουν τον λογαριασμό τους. Σε κάθε

χρήστη με την εγγραφή του, το Firebase παράγει και δίνει αυτόματα ένα μοναδικό userid (uid), το οποίο χρησιμοποιούμε όπως θα περιγραφεί στην παρακάτω ενότητα για να αποθηκεύσουμε στην Cloud Firestore βάση δεδομένων όλα τα δεδομένα του συγκεκριμένου χρήστη.

5.2.2 Βάση Δεδομένων

Στην παρούσα εφαρμογή χρησιμοποιούμε τα εργαλεία Realtime Database, Firebase Storage και Cloud Firebase της εργαλειοθήκης του Firebase για να αποθηκεύσουμε και να διαχειριστούμε όλα τα δεδομένα που απαιτούνται.

5.2.2.1 Realtime Database

Στην εφαρμογή MedCurie χρησιμοποιείται η Realtime Database meds που αναφέρθηκε αναλυτικά στην Ενότητα 4.2.2.1, για να ανακτώνται οι απαραίτητες πληροφορίες για το φάρμακο που ο ασθενής θέλει να καταχωρήσει και περιλαμβάνεται στην αγωγή του. Μέσω αυτής, μπορεί ο χρήστης να αναζητήσει το όνομα του φαρμάκου που λαμβάνει και να το εντοπίσει στην βάση μας.

5.2.2.2 Firebase Storage

Αν για το φάρμακο που αναζητά ο χρήστης υπάρχει ήδη φωτογραφία στο Storage παρουσιάζεται στον χρήστη ώστε να βεβαιωθεί ότι έχει κάνει την σωστή επιλογή, ενώ αν δεν υπάρχει μπορεί ο ίδιος να καταχωρήσει μια φωτογραφία και να την αποθηκεύσει στον φάκελο meds του Firebase Storage, με παρόμοιο τρόπο όπως οι φαρμακοποιοί με την εφαρμογή MedImg_App.

5.2.2.3 Cloud Firebase

Όλος ο όγκος των πληροφοριών που αφορούν έναν συγκεκριμένο χρήστη, δηλαδή τα φάρμακα της αγωγής του, οι σημειώσεις του, οι ιατροί του, οι συνταγές του κλπ, αποθηκεύονται δομημένα σε συλλογές (collections) και έγγραφα (documents) στο Cloud Firebase. Η δομή της βάσης αυτής έχει στηριχθεί στο μοναδικό id του χρήστη που αναφέραμε στην Εισαγωγή, ώστε να είναι εύκολος ο εντοπισμός των εγγραφών που τον αφορούν. Για κάθε κατηγορία δεδομένων η δομή της βάσης φαίνεται στους παρακάτω πίνακες.

- users: Σε αυτό το collection αποθηκεύονται τα δεδομένα που έχουν σχέση με το προφίλ του χρήστη καθώς επίσης και κάποια βοηθητικά δεδομένα για την λειτουργία της εφαρμογής, δηλαδή το “today_counter” στο οποίο αποθηκεύεται το πόσες δόσεις φαρμάκων πρέπει να πάρει σήμερα ο χρήστης, το “today_meds” που είναι μια λίστα με όλες τις δόσεις που ο χρήστης δήλωσε ότι πήρε σήμερα και τέλος το “tokens” που αποθηκεύει σε μια λίστα όλα τα device tokens του συγκεκριμένου χρήστη ώστε να μπορούμε να στέλνουμε ειδοποιήσεις σε όλες τις συσκευές που έχει εγκαταστήσει την εφαρμογή και έχει κάνει login.

Collection	Document	Fields

users	uid	uid | name | email | birthdate | gender | height | weight | today_counter | today_meds | tokens

Πίνακας 4. Δομή της συλλογής users στο Cloud Firestore

- meds: Σε αυτή την συλλογή αποθηκεύονται όλα τα φάρμακα που περιλαμβάνονται στις αγωγές των χρηστών. Μέσω του uid του κάθε χρήστης αποθηκεύει τα φάρμακα της δικής του αγωγής στην προσωπική του συλλογή με το όνομα user_meds.

Collection	**Document**	**Collection**	**Fields**
meds	uid	user_meds	medid | name | barcode | image | symptom | meal | units | dosage | periodicity | daysX | daysXY | daysWeek | dateStart | dateEnd timesperday | intaketime

Πίνακας 5. Δομή της συλλογής meds στο Cloud Firestore

- meds_notif: Όταν ο χρήστης αποθηκεύει ένα νέο φάρμακο στην βάση, δημιουργείται ταυτόχρονα και ένα νέο document με το medid του φαρμάκου στην συλλογή meds_notif. Μέσω αυτού συντονίζονται οι ειδοποιήσει που αφορούν την λήψη του συγκεκριμένου φαρμάκου, με τον τρόπο που θα αναλύσουμε στην Ενότητα 5.2.3.

Collection	**Document**	**Fields**
meds_notif	medid	notifid | uid | whenToNotify | notificationSent | currentdate | intake | body

Πίνακας 6. Δομή της συλλογής meds_notif στο Cloud Firestore

- measurements: Σε αυτή την συλλογή οι χρήστες αποθηκεύουν τις μετρήσεις τους. Για κάθε χρήστη δημιουργείται ένα document με το uid του που περιλαμβάνει τις εξής συλλογές: “user_glucose”, “user_pressure” και “user_temperature”.

Collection	Document	Collection	Fields
measurements	uid	user_glucose	pressureid | value | date
		user_pressure	glucoseid | value | date
		user_temperature	tempid | value | date

Πίνακας 7. Δομή της συλλογής measurements στο Cloud Firestore

- notes: Στην συλλογή notes αποθηκεύονται οι σημειώσεις των χρηστών, ενώ για κάθε χρήστη υπάρχει ξεχωριστή συλλογή “user_notes” στο έγγραφο του uid του.

Collection	Document	Collection	Fields
notes	uid	user_notes	noteid | text | date

Πίνακας 8. Δομή της συλλογής notes στο Cloud Firestore

- doctors: Στην συλλογή doctors αποθηκεύονται οι κατάλογοι των ιατρών των χρηστών στις υποσυλλογές “user_doctors”.

Collection	Document	Collection	Fields
doctors	uid	user_doctors	doctorid | name | specialty | mobile | phone |email | address;

Πίνακας 9. Δομή της συλλογής doctors στο Cloud Firestore

- assistants: Στην συλλογή assistants αποθηκεύονται οι φροντιστές των χρηστών στις υποσυλλογές “user_assistants”

Collection	Document	Collection	Fields
assistants	uid	user_assistants	assistantid | name | relationship | mobile | phone | email

Πίνακας 10. Δομή της συλλογής assistants στο Cloud Firestore

- prescriptions: Εδώ αποθηκεύουν οι χρήστες τις συνταγές που έχουν προς εκτέλεση στην συλλογή prescriptions.

Collection	Document	Collection	Fields
prescriptions	uid	user_prescriptions	prescriptionid | barcode | datestart | dateend | category

Πίνακας 11. Δομή της συλλογής prescriptions στο Cloud Firestore

- prescriptions_notif: Εδώ ακολουθείται μια παρόμοια λογική με τα φάρμακα του χρήστη, ώστε για κάθε συνταγή του που καταχωρεί στην συλλογή "pescriptions" να δημιουργείται ταυτόχρονα μια εγγραφή στην συλλογή "prescriptions_notif". Μέσω αυτής συντονίζονται οι ειδοποιήσεις που στέλνονται στον χρήστη για να τον ενημερώσουν για τις ημερομηνίες έναρξης και λήξης της συνταγής του.

Collection	Document	Fields
prescriptions_notif	prescriptionid	notifid | uid | whenToNotify | notificationSent | mode | body

Πίνακας 12. Δομή της συλλογής prescriptions_notif στο Cloud Firestore

- appointments: Σε αυτή την συλλογή αποθηκεύονται τα ραντεβού των ασθενών σε μια ξεχωριστή υποσυλλογή για τον καθένα που ονομάζεται "user_appointments".

Collection	Document	Collection	Fields
appointments	uid	user_appointments	appointmentid | doctorsname | specialty | address | date | note

Πίνακας 13. Δομή της συλλογής appointments στο Cloud Firestore

- appointments_notif: Επίσης, εδώ υλοποιείται μια παρόμοια λογική όπως στα φάρμακα και τις συνταγές του ασθενή. Έτσι με την καταχώρηση ενός νέου ραντεβού δημιουργείται αυτόματα για εγγραφή για το ραντεβού αυτό στην συλλογή "appointments_notif", ώστε ο χρήστης να ειδοποιηθεί για το ραντεβού του.

Collection	Document	Fields
appoinements_notif	appointmentid	notifid | uid | whenToNotify | notificationSent | body

Πίνακας 14. Δομή της συλλογής appointments_notif στο Cloud Firestore

Τέλος, για την βάση στο Cloud Firestore να αναφερθεί ότι έχουν τροποποιηθεί οι κανόνες (Rules), ώστε να έχουν πρόσβαση μόνο όσοι χρήστες έχουν ταυτοποιηθεί.

Cloud Firestore rules

```
rules_version = '2';
service cloud.firestore {
  match /databases/{database}/documents {
    match /{document=**} {
      allow read, write: if request.auth != null;
    }
  }
}
```

Πίνακας 15. Cloud Firestore rules

5.2.3 Cloud Functions

Προκειμένου να διαχειριστούμε τις καθημερινές ειδοποιήσεις που πρέπει να λαμβάνουν οι χρήστες σχετικά με την φαρμακευτική τους αγωγή, αλλά και τις συνταγές και τα ραντεβού που έχουν αποθηκεύσει στην εφαρμογή μας, αξιοποιήσαμε το εργαλείο Cloud Functions από την εργαλειοθήκη του Firebase. Με αυτόν τον τρόπο καταφέραμε οι υπολογιστικά βαριές διαδικασίες να μην εκτελούνται στην συσκευή του χρήστη αλλά στο cloud της Google. Επιπλέον, αξιοποιήσαμε το εργαλείο Cloud Messaging του Firebase και δημιουργήσαμε έτσι ένα αξιόπιστο σύστημα Push notifications στους χρήστες της εφαρμογής.

Συγκεκριμένα γράφτηκαν σε JavaScript (Node.js) και έγιναν deploy στο Firebase οι τέσσερις συναρτήσεις που παρουσιάζονται παρακάτω:

- **scheduledInitialize (Περίπου 30 κλήσεις/μήνα):** Αυτή η συνάρτηση έχει προγραμματιστεί να εκτελείται κάθε μέρα στις 00:01 ώρα Ελλάδος και να αρχικοποιεί κάποια πεδία της βάσης, που αφορούν την διαχείριση των λήψεων των ημερήσιων δόσεων φαρμάκων από τον χρήστη. Αρχικά, αδειάζει την λίστα "today_meds" με τις δόσεις που έχει πάρει ο χρήστης την προηγούμενη ημέρα και στην συνέχεια, υπολογίζει πόσες δόσεις πρέπει να πάρει ο χρήστης την τρέχουσα ημέρα και αποθηκεύει αυτόν τον αριθμό στο field "today_counter" στο document του χρήστη στην συλλογή users που αναφέραμε στην προηγούμενη ενότητα. Ακολουθεί ο κώδικας για αυτή την συνάρτηση:

```
exports.scheduledInitialize = functions.pubsub.schedule('1 0 * * *')
  .timeZone('Europe/Athens') // Users can choose timezone - default is America/Los_Angeles
  .onRun( async (context) => {
       var moment = require('moment');
       require('moment-timezone');
       const UTC = moment.tz(moment.utc(), 'Europe/Athens').utcOffset()/60;

       const query = await database.collection("users").get();

    query.forEach(async snapshot => {
        await updateCounter(snapshot.data().userid);
        await database.doc('users/' + snapshot.data().userid).update({
            "today_meds": [],
        });
    });

    async function updateCounter(uid){
               const query = await
database.collection("meds").doc(uid).collection('user_meds').get();
               var counter = 0;

               query.forEach(async snapshot => {
                      var dstart = new Date(snapshot.data().dateStart);
                      var dend = new Date(snapshot.data().dateEnd);
                      var today = new Date();
                      today.setHours( today.getHours() + UTC );
                      if(today >= dstart && today <= dend){
                             switch(snapshot.data().periodicity) {
                            //Periodicity: Every day
                               case 1:
                                     counter += snapshot.data().timesperday;
                                     break;
                               //Periodicity: On specific date
                               case 2:
                                     counter += snapshot.data().timesperday;
                                     break;
                               //Periodicity: Every X days
                               case 3:
                                     if (daysfromstart(dstart, today) % snapshot.data().daysX == 0)
                                            counter += snapshot.data().timesperday;
                                     break;
                               //Periodicity: On specific days of the week
                               case 4:
```

```
                                    if(snapshot.data().daysWeek[today.getDay() % 7]) {
                                          counter += snapshot.data().timesperday;
                                    }
                                    break;
                              //Periodicity: Cycle: X intake, Y pass
                              case 5:
                                    if (daysfromstart(dstart, today) %
(snapshot.data().daysXY[0]+snapshot.data().daysXY[1]) <= snapshot.data().daysXY[0]) {
                                          counter += snapshot.data().timesperday;
                                    }
                                    break;
                              default:
                                    // code block
                                    break;
                        }

                  }

            });

            await database.doc('users/' + uid).update({
                        "today_counter": counter,
                  });

            return console.log('End Of updateCounter('+uid+')');
    }


       function daysfromstart(dstart, today) {
            //calculate time difference
            var time_difference = today.getTime() - dstart.getTime();

        //calculate days difference by dividing total milliseconds in a day
        var days_difference = time_difference / (1000 * 60 * 60 * 24);
            return days_difference;
       }

    return console.log('End Of scheduledInitialize Function. This will be run tomorrow at 00:01
AM!');
});
```

- **AppointmentsNotifications (Περίπου 1440 κλήσεις/μήνα):** Αυτή η συνάρτηση δημιουργεί και στέλνει τις ειδοποιήσεις που αφορούν τα ραντεβού του ασθενή με τους ιατρούς του. Έχει προγραμματιστεί να τρέχει κάθε μισή ώρα και όταν βρει στην συλλογή “appointments_notif” μια εγγραφή με το πεδίο “whenToNotify” να είναι μικρότερο από την τρέχουσα ημέρα και ώρα και το πεδίο “notificationSent” false, δημιουργεί και στέλνει ειδοποίηση σε όλες τις συσκευές που χρησιμοποιεί ο χρήστης με βάση τα device tokens που είναι αποθηκευμένα στο field “tokens” του προφίλ του. Αφού ολοκληρωθεί η διαδικασία αποστολής της ειδοποίησης αλλάζει το πεδίο “notidicationSent” της συγκεκριμένης καταχώρησης σε true για να μην στείλει ξανά την επόμενη φορά που θα εκτελεστεί. Ακολουθεί ο κώδικας της συνάρτησης:

```
exports.AppointmentsNotifications = functions.pubsub.schedule('every 30 minutes').onRun(async
(context) => {

    const query = await database.collection("appointments_notif")
        .where("whenToNotify", '<=', admin.firestore.Timestamp.now())
        .where("notificationSent", "==", false).get();

    query.forEach(async snapshot => {
        await sendAppointmentNotification(snapshot.data().uid, snapshot.data().body);
        await database.doc('appointments_notif/' + snapshot.data().notifid).update({
            "notificationSent": true,
        });
    });

    async function sendAppointmentNotification(uid, body) {
        let title = "Ραντεβού";
              //get all device tokens from user's profile
              var androidNotificationTokens = [];
              await database.doc('users/' + uid).get().then((doc) => {
                     if (doc.exists) {
                            androidNotificationTokens = doc.data().tokens;
                     } else {
                            console.log("No such document!");
                     }
              }).catch((error) => {
                     console.log("Error getting document:", error);
              });

              //push notification to all user's devices
              for (var i = 0; i < androidNotificationTokens.length; i++)  {
                     var message = {
                            notification: { title: title, body: body },
                            token: androidNotificationTokens[i],
                            data: { click_action: 'FLUTTER_NOTIFICATION_CLICK', screen:
'/appointments' }
                     };

                     admin.messaging().send(message).then(response => {
                            return console.log("Successful Message Sent");
                     }).catch(error => {
                            return console.log("Error Sending Message");
                     });
              }
    }
    return console.log('End Of AppointmentsNotifications Function.');
});
```

- **PrescriptionsNotifications (Περίπου 1440 κλήσεις/μήνα):** Μια αντίστοιχη διαδικασία με την συνάρτηση που αρχικοποιεί τις ειδοποιήσεις για τα ραντεβού, υλοποιεί και η συνάρτηση που επεξεργάζεται τις ειδοποιήσεις για τις συνταγές του χρήστη. Η κύρια διαφορά τους είναι ότι στην συλλογή "prescriptions_notif" της βάσης υπάρχει ένα επιπλέον πεδίο για κάθε εγγραφή συνταγής. Το πεδίο "mode" είναι αρχικοποιημένο ως start όταν ακόμα δεν έχει φύγει η ειδοποίηση για την ημέρα

έναρξης της συνταγής, ενώ το “body” της ειδοποίησης ενημερώνει τον χρήστη ότι αυτή είναι η πρώτη μέρα εκτέλεσης της συνταγής του. Αν η συνάρτηση PrescriptionsNotifications βρει και στείλει μια ειδοποίηση με “mode” start, αμέσως μετά τροποποιεί το πεδίο “mode” σε end, αλλάζει το body της ειδοποίησης ώστε να ενημερώνεται ο χρήστης ότι αυτή είναι η τελευταία ημέρα εκτέλεσης της συνταγής του και τέλος ορίζει το πεδίο “whenToNotify” ίσο με την ημερομηνία “dateend” στις 10:00. Όταν η συνάρτηση στείλει και την δεύτερη ειδοποίηση το πεδίο “notificationSent” γίνεται true. Ο κώδικας της συνάρτησης είναι αυτός που ακολουθεί:

```
exports.PrescriptionsNotifications = functions.pubsub.schedule('every 30 minutes').onRun(async
(context) => {
       var moment = require('moment');
       require('moment-timezone');
       const UTC = moment.tz(moment.utc(), 'Europe/Athens').utcOffset()/60;

    const query = await database.collection("prescriptions_notif")
        .where("whenToNotify", '<=', admin.firestore.Timestamp.now())
        .where("notificationSent", "==", false).get();

    query.forEach(async snapshot => {
        await sendPrescriptionNotification(snapshot.data().uid, snapshot.data().body);
              if(snapshot.data().mode == 'end') {
                     await database.doc('prescriptions_notif/' +
snapshot.data().notifid).update({
                            "notificationSent": true,
                     });
              }else{
                     var newDate;
                     await
database.collection("prescriptions").doc(snapshot.data().uid).collection('user_prescriptions').doc
(snapshot.data().notifid).get().then((doc) => {
                            if (doc.exists) {
                                   newDate = new Date(doc.data().dateend);
                                   //notification for prescription end at dateend 10:00 AM
                                   newDate.setHours( newDate.getHours() + 10 - UTC);
                            } else {
                                   console.log("No such document!");
                            }
                     }).catch((error) => {
                            console.log("Error getting document:", error);
                     });
                     new Date(snapshot.data().dateStart)
                     await database
                          .doc('prescriptions_notif/' + snapshot.data().notifid).update({
                            "mode": "end",
                            "body": "Σήμερα είναι η τελευταία μέρα εκτέλεσης της συνταγής σας",
                            "whenToNotify": newDate,
                     });
              }
    });

    async function sendPrescriptionNotification(uid, body) {
        let title = "Συνταγή";

              //get all device tokens from user's profile
```

```
            var androidNotificationTokens = [];
            await database.doc('users/' + uid).get().then((doc) => {
                  if (doc.exists) {
                        androidNotificationTokens = doc.data().tokens;
                  } else {
                        console.log("No such document!");
                  }
            }).catch((error) => {
                  console.log("Error getting document:", error);
            });

            //push notification to all user's devices
            for (var i = 0; i < androidNotificationTokens.length; i++)  {
                  var message = {
                        notification: { title: title, body: body },
                        token: androidNotificationTokens[i],
                        data: { click_action: 'FLUTTER_NOTIFICATION_CLICK', screen:
'/prescriptions' }
                  };

                  admin.messaging().send(message).then(response => {
                        return console.log("Successful Message Sent");
                  }).catch(error => {
                        return console.log("Error Sending Message");
                  });
            }
    }
    return console.log('End Of PrescriptionsNotifications Function');
});
```

- **MedsNotifications (Περίπου 43200 κλήσεις/μήνα):** Η συνάρτηση MedsNotifications είναι υπεύθυνη για την αποστολή των ειδοποιήσεων που σχετίζονται με την λήψη της αγωγής του ασθενή. Είναι προγραμματισμένη να τρέχει κάθε ένα λεπτό ώστε να τσεκάρει αν ήρθε η ώρα ο ασθενής να πάρει κάποια δόση από το φάρμακό του. Αφού στείλει την ειδοποίηση, αναλαμβάνει να προγραμματίσει την ειδοποίηση για την επόμενη δόση του συγκεκριμένου φαρμάκου, με βάση την περιοδικότητα που έχει ορίσει ο χρήστης ότι πρέπει να το λαμβάνει. Όταν φτάσουμε στην ημερομηνία που ο χρήστης έχει ορίσει ως ημερομηνία λήξης της αγωγής του, απενεργοποιεί την συγκεκριμένη εγγραφή στην συλλογή "meds_notif" θέτοντας το πεδίο "notificationSent" true.

```
exports.MedsNotifications = functions.pubsub.schedule('* * * * *').onRun(async (context) => {
       var moment = require('moment');
       require('moment-timezone');
       const UTC = moment.tz(moment.utc(), 'Europe/Athens').utcOffset()/60;

       var dateFormat = require('dateformat');

    const query = await database.collection("meds_notif")
        .where("whenToNotify", '<=', admin.firestore.Timestamp.now())
```

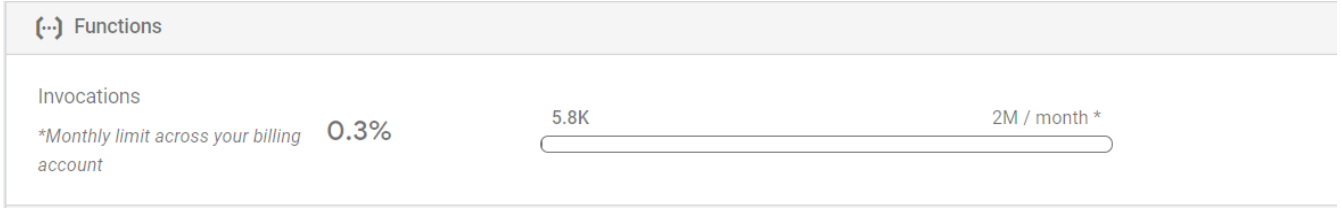
```
        .where("notificationSent", "==", false).get();

    query.forEach(async snapshot => {
        await sendMedNotification(snapshot.data().uid, snapshot.data().body);
              var currentdate = snapshot.data().currentdate;
              var intake = snapshot.data().intake;
              var newDate = new Date(currentdate);
              var notifsent = true;
              await
database.collection("meds").doc(snapshot.data().uid).collection('user_meds').doc(snapshot.data().n
otifid).get().then((doc) => {
                     if (doc.exists) {
                            intake ++;
                            if(intake<doc.data().timesperday){

newDate.setHours(parseInt(doc.data().intaketime[intake].substring(0, 2)));

newDate.setMinutes(parseInt(doc.data().intaketime[intake].substring(3, 5)));
                                   notifsent = false;
                            } else {
                                   intake = 0;
                                   newDate.setDate(newDate.getDate() + 1);

                                   var dstart = new Date(doc.data().dateStart);
                                   var dend = new Date(doc.data().dateEnd);

                                   notifsent = true;

                                   if(newDate >= dstart && newDate <= dend){
                                          switch(doc.data().periodicity) {
                                            //Periodicity: Every day
                                            case 1:

newDate.setHours(parseInt(doc.data().intaketime[intake].substring(0, 2)));

newDate.setMinutes(parseInt(doc.data().intaketime[intake].substring(3, 5)));
                                                 notifsent = false;
                                                 break;
                                            //Periodicity: On specific date
                                            case 2:
                                                 break;
                                            //Periodicity: Every X days
                                            case 3:
                                                 newDate.setDate(newDate.getDate() +
(doc.data().daysX-1));

                                                 dend.setDate(dend.getDate()+1);
                                                 if(newDate<=dend){

newDate.setHours(parseInt(doc.data().intaketime[intake].substring(0, 2)));

newDate.setMinutes(parseInt(doc.data().intaketime[intake].substring(3, 5)));
                                                        notifsent = false;
                                                 }
                                                 break;
                                            //Periodicity: On specific days of the week
                                            case 4:
```

```
                                                while(!doc.data().daysWeek[newDate.getDay() %
7]) {
                                                       newDate.setDate(newDate.getDate() + 1);
                                                }
                                                dend.setDate(dend.getDate()+1);
                                                if(newDate<=dend){

newDate.setHours(parseInt(doc.data().intaketime[intake].substring(0, 2)));

newDate.setMinutes(parseInt(doc.data().intaketime[intake].substring(3, 5)));
                                                       notifsent = false;
                                                }
                                                break;
                                          //Periodicity: Cycle: X intake, Y pass
                                          case 5:
                                                if (daysfromstart(dstart, newDate) %
(doc.data().daysXY[0]+doc.data().daysXY[1]) <= doc.data().daysXY[0]) {
                                                       newDate.setDate(newDate.getDate() +
(doc.data().daysXY[1]-1));
                                                }
                                                dend.setDate(dend.getDate()+1);
                                                if(newDate<=dend){

newDate.setHours(parseInt(doc.data().intaketime[intake].substring(0, 2)));

newDate.setMinutes(parseInt(doc.data().intaketime[intake].substring(3, 5)));
                                                       notifsent = false;
                                                }
                                                break;
                                          default:
                                                // code block
                                                break;
                                    }
                              }
                        }
                        currentdate = dateFormat(newDate, "yyyy/mm/dd");
                        newDate.setHours(newDate.getHours() - UTC);
                  } else {
                        // doc.data() will be undefined in this case
                        console.log("No such document!");
                  }
            }).catch((error) => {
                  console.log("Error getting document:", error);
            });
            new Date(snapshot.data().dateStart)
            await database.doc('meds_notif/' + snapshot.data().notifid).update({
                  "intake": intake,
                  "whenToNotify": newDate,
                  "currentdate":currentdate,
            });
    });

    async function sendMedNotification(uid, body) {
        let title = "Αγωγή";
            //get all device tokens from user's profile
            var androidNotificationTokens = [];
```

```
            await database.doc('users/' + uid).get().then((doc) => {
                  if (doc.exists) {
                        androidNotificationTokens = doc.data().tokens;
                  } else {
                        console.log("No such document!");
                  }
            }).catch((error) => {
                  console.log("Error getting document:", error);
            });

            //push notification to all user's devices
            for (var i = 0; i < androidNotificationTokens.length; i++)  {
                  var message = {
                        notification: { title: title, body: body },
                        token: androidNotificationTokens[i],
                        data: { click_action: 'FLUTTER_NOTIFICATION_CLICK', screen: '/' }
                  };

                  admin.messaging().send(message).then(response => {
                        return console.log("Successful Message Sent");
                  }).catch(error => {
                        return console.log("Error Sending Message");
                  });
            }
    }

       function daysfromstart(dstart, today) {
            //calculate time difference
            var time_difference = today.getTime() - dstart.getTime();

        //calculate days difference by dividing total milliseconds in a day
        var days_difference = time_difference / (1000 * 60 * 60 * 24);
            return days_difference;
       }

    return console.log('End Of MedsNotifications Function');
});
```

Για τις παραπάνω συναρτήσεις εγκαταστήσαμε και χρησιμοποιήσαμε τα JavaScript πακέτα “moment” & "moment-timezone”, ώστε να υπολογίζουμε το UTC offset για την Ελλάδα, καθώς και το πακέτο “dateformat” για την εύκολη μορφοποίηση των ημερομηνιών όπου υπήρξε ανάγκη πριν την καταχώρησή τους στην βάση.

Για την εφαρμογή MedCurie λοιπόν θα γίνουν συνολικά περίπου 46110 κλήσεις cloud function ανά μήνα, που είναι λίγες σε σχέση με τις δωρεάν κλήσεις που δίνει το Firebase ανά μήνα, όπως φαίνεται στην Εικόνα 16.



Εικόνα 16. Όριο κλήσεων συναρτήσεων από την κονσόλα του Firebase, χωρίς χρέωση

5.3 Οθόνες εφαρμογής

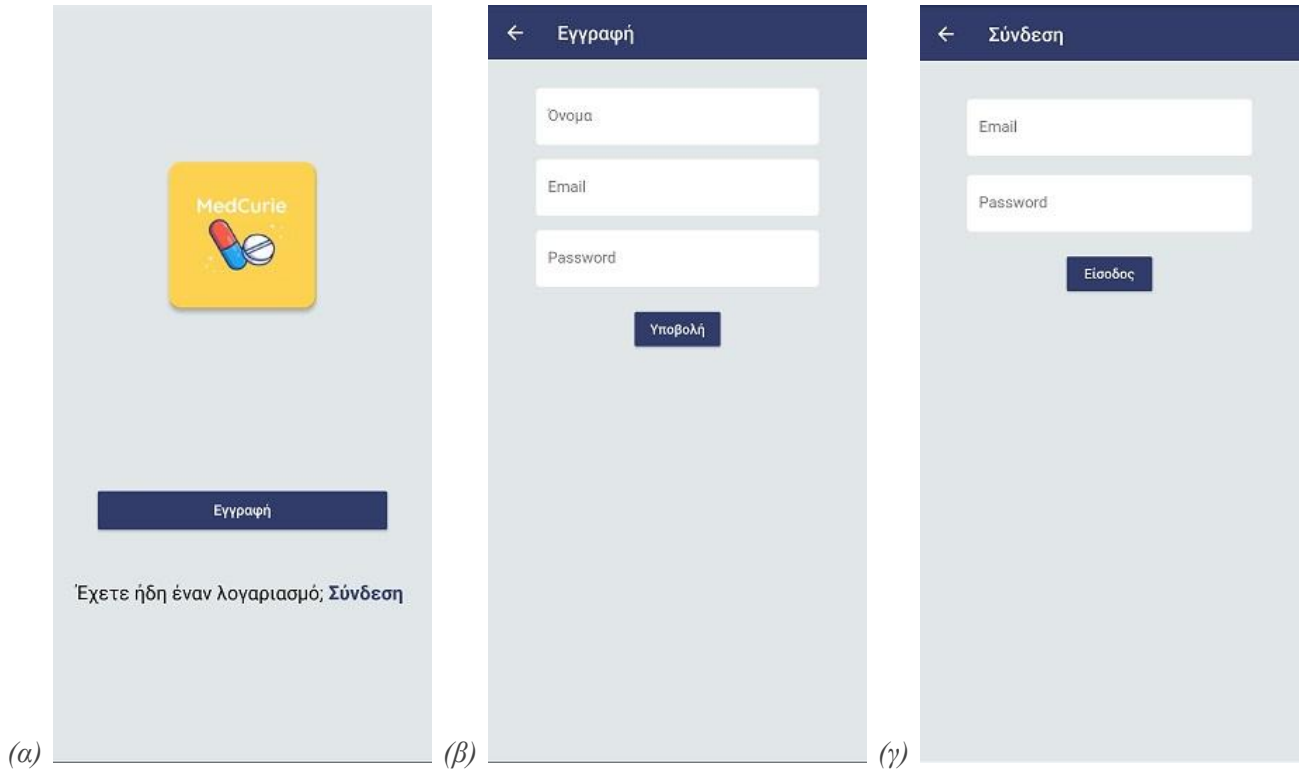
Σε αυτή την ενότητα παρουσιάζονται αναλυτικά οι βασικές οθόνες της εφαρμογής, ώστε να δώσουμε μια ολοκληρωμένη εικόνα των λειτουργιών της τόσο θεωρητικά όσο και οπτικά. Ακολουθούν οι εικόνες και οι περιγραφές τους.

5.3.1 Οθόνες Υποδοχής, Σύνδεσης & Εγγραφής

Η πρώτη επαφή του χρήστη με την εφαρμογή είναι η οθόνη υποδοχής (Εικόνα 17α). Όταν το user stream δεν περιλαμβάνει κάποιον χρήστη (δηλαδή κανένας χρήστης δεν έχει ακόμα κάνει login στην εφαρμογή), οπότε ο stream provider δίνει user == null τότε εμφανίζεται στον χρήστη η οθόνη υποδοχής, που τον παροτρύνει είτε να πατήσει τον σύνδεσμο "Σύνδεση" εάν έχει ήδη λογαριασμό στην εφαρμογή μας είτε να προχωρήσει στην εγγραφή του πατώντας το κουμπί "Εγγραφή".

Πατώντας το κουμπί "Εγγραφή" ο χρήστης μεταφέρεται στην οθόνη με την φόρμα εγγραφής (Εικόνα 17β). Εκεί καλείται να συμπληρώσει στο πρώτο πεδίο το όνομα με το οποίο επιθυμεί να εμφανίζεται στην εφαρμογή, στο δεύτερο πεδίο το Email του και στο τρίτο το Password του (απαιτούνται τουλάχιστον 6 χαρακτήρες, αλλιώς εμφανίζεται προειδοποιητικό μήνυμα λάθους). Πατώντας το κουμπί "Υποβολή" ένας νέος χρήστης με τα στοιχεία που συμπλήρωσε και ένα τυχαίο uid δημιουργείται στην βάση χρηστών του Firebase. Ένα προφίλ με dummy στοιχεία αποθηκεύεται στην συνολή "users" του Cloud Firestore με document το τυχαίο uid που δημιούργησε το Firebase κατά την εγγραφή του. Στην συνέχεια, ο χρήστης μεταφέρεται στην αρχική οθόνη της εφαρμογής που θα παρουσιάσουμε παρακάτω.

Στην περίπτωση, που ο χρήστης έχει ήδη λογαριασμό πατώντας, τον σύνδεσμο "Σύνδεση" της οθόνης υποδοχής, μεταφέρεται στην οθόνη με την φόρμα σύνδεσης (Εικόνα 17γ). Εκεί αφού συμπληρώσει το Email και το Password του και πατήσει το κουμπί "Είσοδος", εάν τα στοιχεία που έχει πληκτρολογήσει είναι σωστά και ταυτοποιηθεί από το Firebase ως έγκυρος χρήστης μεταφέρεται στην αρχική οθόνη της εφαρμογής (ανάλυση στην ενότητα που ακολουθεί). Εάν τα στοιχεία του είναι λανθασμένα εμφανίζεται προειδοποιητικό μήνυμα στον χρήστης και προτείνεται να προσπαθήσει ξανά.



(α) *(β)* *(γ)*

Εικόνα 17. Authentication (α) Οθόνη υποδοχής, (β) Οθόνη εγγραφής και (γ) Οθόνη σύνδεσης

5.3.2 Βασικές οθόνες εφαρμογής

Ακολούθως, αναλύονται οι οθόνες στις οποίες έχει πρόσβαση ο χρήστης μετά την τακτοποίησή του. Κάθε μία από τις υποενότητες 5.3.2.1-5.3.3.8 έχει ως τίτλο μια από τις επιλογές που εμφανίζονται στο side navigation menu της εφαρμογής, ενώ εντός αυτής αναλύονται όλες οι οθόνες στις οποίες μπορεί να έχει πρόσβαση ο χρήστης κάνοντας κλικ στην συγκεκριμένη επιλογή.

5.3.2.1 Αρχική οθόνη

Ο χρήστης αφού έχει ολοκληρώσει την σύνδεση ή την εγγραφή του, οδηγείται στην αρχική οθόνη της εφαρμογής (Εικόνα 18α). Η επιλεγμένη ημερομηνία στο timeline στην κορυφή της οθόνης είναι αρχικά η τρέχουσα. Για την υλοποίηση αυτού του timeline χρησιμοποιήθηκε το πακέτο “date_picker_timeline”. Στην οθόνη του λοιπόν, ο χρήστης βλέπει σε μια λίστα τις δόσεις των σκευασμάτων που περιλαμβάνονται στην αγωγή του για την τρέχουσα ημερομηνία. Κάνοντας κλικ στο πορτοκαλί check icon μπορεί να δηλώσει ότι πήρε την δόση του συγκεκριμένου φαρμάκου και αυτό να αποθηκευτεί στην λίστα με τις δόσεις που έχει πάρει για την τρέχουσα ημερομηνία και να μην εμφανιστεί για το υπόλοιπο της ημέρας στην αρχική του οθόνη.

Παρόμοια δομή έχει η αρχική οθόνη και για οποιαδήποτε μετέπειτα ημερομηνία επιλέξει ο χρήστης κάνοντας slide και κλικ στο timeline (Εικόνα 18β). Έτσι ο χρήστης μπορεί να γνωρίζει τις δόσεις της φαρμακευτικής του αγωγής για οποιαδήποτε μέρα και να προνοήσει για αυτές. Η διαφορά ανάμεσα στην λίστα της τρέχουσας ημερομηνίας και σε αυτές των επόμενων ημερομηνιών είναι ότι απουσιάζει το check icon, ώστε ο χρήστης να

μην σβήσει κατά λάθος κάποια δόση επόμενης ημέρας. Τόσο για την τρέχουσα όσο και για τις επόμενες ημερομηνίες καλείται η συνάρτηση getDateMeds [54], η οποία παίρνει σαν όρισμα μια ημερομηνία και επιστρέφει κάνοντας αναζήτηση στα φάρμακα που έχει αποθηκεύσει ο χρήστης, μια λίστα με όλες τις δόσεις από κάθε φάρμακο που πρέπει να πάρει ο χρήστης για την συγκεκριμένη ημερομηνία. Πιο αναλυτικά η συνάρτηση αυτή:

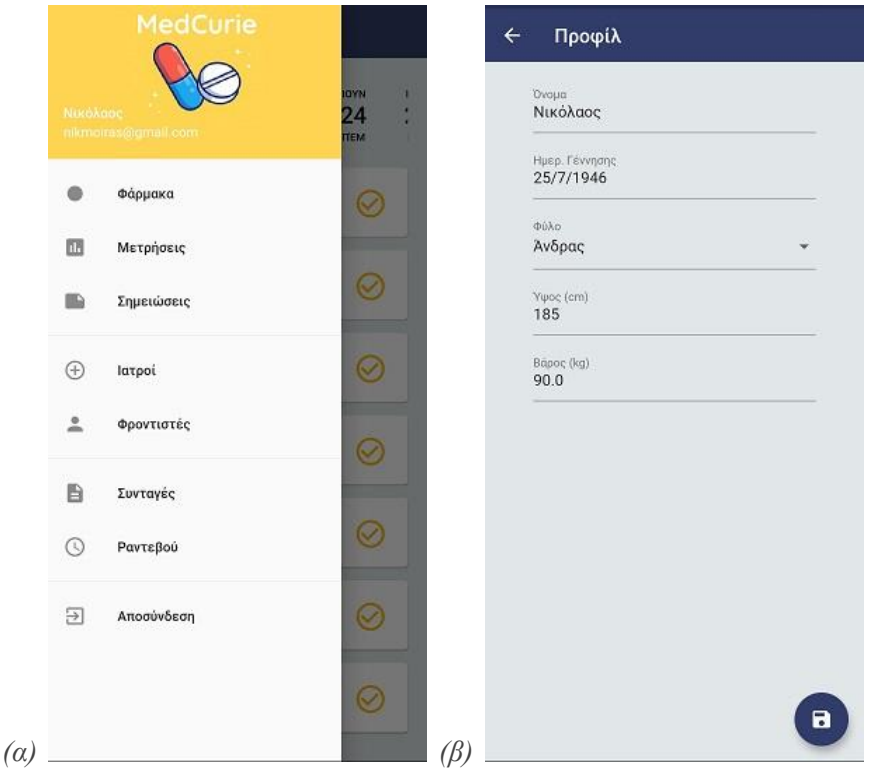
- Αρχικά, κάνει ένα query στην βάση ώστε να πάρει τα φάρμακα που η ημερομηνία που έχει επιλέξει ο χρήστης βρίσκεται ανάμεσα στις ημερομηνίες έναρξης και λήξης της αγωγής .
- Έπειτα, για τις περιπτώσεις που η αγωγή έχει περιοδικότητα κάθε Χ ημέρες τσεκάρει αν η ημερομηνία που έχει επιλέξει ο χρήστης διαφέρει από την ημερομηνία έναρξης της αγωγής κατά ημέρες πολλαπλάσιες του Χ, ενώ αν η περιοδικότητα είναι ορισμένη σε κύκλους τσεκάρει αν αυτή η ημερομηνία πέφτει σε φάση λήψης ή διακοπής της αγωγής. Επίσης, για την περίπτωση που ο χρήστης έχει δηλώσει ότι πρέπει να παίρνει το φάρμακο σε συγκεκριμένες ημέρες της εβδομάδας, ελέγχει αν η ημερομηνία που της δόθηκε ταυτίζεται σε αυτές τις ημέρες της εβδομάδας.
- Τέλος, αν το όρισμα είναι η τρέχουσα ημερομηνία αφαιρεί από την λίστα που θα επιστρέψει στην αρχική οθόνη, τις δόσεις που ο χρήστης έχει ήδη πάρει.

Επιπλέον, για να διευκολύνουμε τον χρήστη στην διαδικασία της σωστής λήψης της αγωγής του δίνεται η δυνατότητα πατώντας πάνω στο ListTile κάθε δόσης να δει λεπτομέρειες σχετικά με αυτή, συγκεκριμένα: το πλήρες όνομα του σκευάσματος, μια φωτογραφία της συσκευασίας του αν υπάρχει στην βάση μας, το σύμπτωμα που καταπολεμά με αυτό το φάρμακο, οδηγίες σχετικά με το αν πρέπει να έχει καταναλώσει τροφή πριν ή μετά την λήψη του φαρμάκου, καθώς και την δόση και την ώρα που πρέπει να την πάρει.

Εικόνα 18. Αρχική οθόνη (α) Τρέχουσα ημερομηνία, (β)Μετέπειτα ημερομηνία και (γ) Οδηγίες λήψης δόσης φαρμάκου

5.3.2.2 Μενού Εφαρμογής & Προφίλ Χρήστη

Πατώντας το hamburger icon που βρίσκεται αριστερά στο AppBar της αρχικής οθόνης (Εικόνα 18α), ανοίγει στα αριστερά της οθόνης το side navigation menu της εφαρμογής (Εικόνα 19α). Κάνοντας κλικ στις επιλογές αυτού του μενού ο χρήστης μπορεί να πλοηγηθεί σε όλες τις οθόνες της εφαρμογής που θα παρουσιαστούν αναλυτικά στις επόμενες ενότητες. Εάν επιλέξει να κάνει κλικ πάνω στο όνομά του θα οδηγηθεί στην οθόνη του προφίλ του (Εικόνα 19β), όπου μπορεί να τροποποιήσει τα προσωπικά του δεδομένα, δηλαδή το όνομα, την ημερομηνία γέννησης, το φύλο, το ύψος και το βάρος του. Αρχικά, στα πεδία της φόρμας παρουσιάζονται τα στοιχεία που είναι ήδη αποθηκευμένα στην βάση και ανακαλούνται μέσω του stream userProfile που έχουμε ορίσει. Αν αποφασίσει να τροποποιήσει κάποιο από τα παραπάνω και πατήσει το floatingActionButton κάτω δεξιά στην οθόνη του, τα στοιχεία του θα ανανεωθούν μέσω της συνάρτησης UserDatabaseService(uid: user.uid).updateUserData(name, email, birthdate, gender, height, weight, today_counter, today_meds) [54].



Εικόνα 19. Μενού και προφίλ (α) Side navigation menu και (β) Οθόνη προφίλ του χρήστη

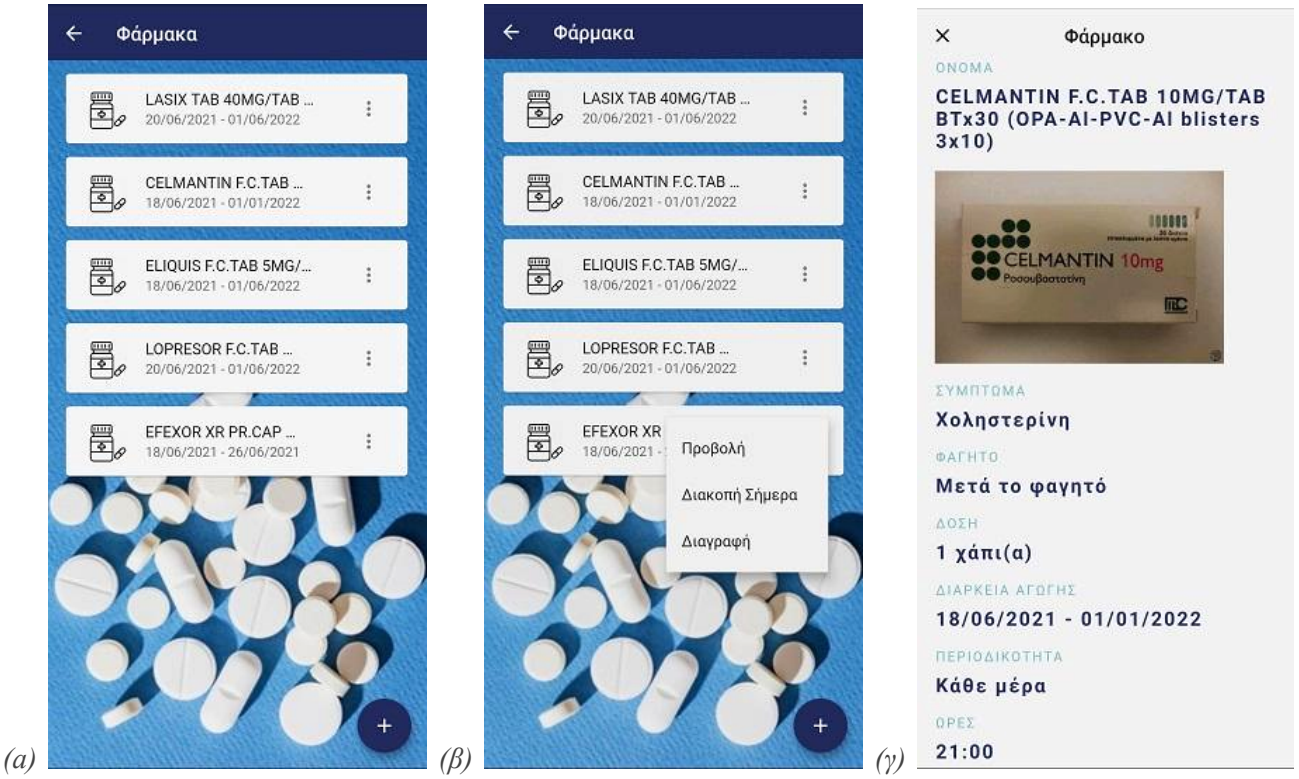
5.3.2.2 Φάρμακα

Η βασικότερη οθόνη της εφαρμογής, μετά την αρχική, είναι αυτή των Φαρμάκων (Εικόνα 20α), γι' αυτό βρίσκεται και στην κορυφή του μενού. Ο χρήστης σε αυτή την οθόνη μπορεί να δει την λίστα με τα φάρμακα που περιλαμβάνονται στην αγωγή του. Για κάθε ένα από αυτά ο χρήστης μπορεί να εκτελέσει μια από τις παρακάτω ενέργειες (Εικόνα 20β):

- Προβολή: Πατώντας στην προβολή εμφανίζονται στην οθόνη του χρήστη όλες οι πληροφορίες σχετικά με το συγκεκριμένο φάρμακο (Εικόνα 20γ).
- Διακοπή Σήμερα: Με αυτή την επιλογή δίνεται στον χρήστη η δυνατότητα να ορίσει την τρέχουσα ημέρα ως την τελευταία που λαμβάνει αυτή την αγωγή. Επιλέξαμε αυτόν τον τρόπο ώστε ο χρήστης να

μπορεί να κρατήσει στο ιστορικό του την προηγούμενη δοσολογία για αυτό το φάρμακο. Έτσι όταν για παράδειγμα ο ιατρός του του συστήσει να κρατήσει το φάρμακο αλλά να αλλάξει δοσολογία ο χρήστης μπορεί να πατήσει "Διακοπή Σήμερα" στην τρέχουσα καταχώρηση και να καταχωρήσει εκ νέου το φάρμακο με την νέα του δοσολογία.

- Διαγραφή: Σε περίπτωση που ο χρήστης επιθυμεί να διαγράψει τελείως αυτό το φάρμακο από την βάση μπορεί να επιλέξει την διαγραφή. Ζητείται η επιβεβαίωσή του (Εικόνα 20δ) ώστε να κληθεί η συνάρτηση MedDatabaseService(uid: user.uid).deleteMedData(medid) [54].



Εικόνα 20 Οθόνη φαρμάκων.(α) Λίστα φαρμάκων(β) Λειτουργικότητες οθόνης φαρμάκων και (γ) Λεπτομέρειες εγγραφής φαρμάκου

Όταν ο χρήστης θελήσει να προχωρήσει στην εισαγωγή ενός νέου φαρμάκου στην αγωγή του μπορεί να πατήσει το floatingActionButton στο κάτω δεξιά μέρος της οθόνης φαρμάκων και να μεταφερθεί στο πρώτο βήμα της νέας καταχώρησης. Εκεί η πρώτη του ενέργεια είναι να αρχίσει να πληκτρολογεί το όνομα του σκευάσματος που λαμβάνει, με κεφαλαίους, λατινικούς χαρακτήρες όπως του ζητείται. Αφού πληκτρολογήσει τους τουλάχιστον τρεις πρώτους χαρακτήρες γίνεται αναζήτηση στην βάση των φαρμάκων και του επιστρέφονται σε ένα drop down list (Εικόνα 20ε) τα πλήρη ονόματα όλων των σκευασμάτων που αρχίζουν με τους χαρακτήρες που πληκτρολόγησε, ώστε να επιλέξει αυτό που τον ενδιαφέρει. Η συνάρτηση που εκτελεί την προαναφερθείσα αναζήτηση είναι η medNames, που παίρνει ως όρισμα ένα String με τους πρώτους χαρακτήρες του ονόματος του φαρμάκου και επιστρέφει μια λίστα από Strings με όλες τις πρώτες δέκα πιθανές επιλογές ονομάτων.

```
final dbRef = FirebaseDatabase.instance.reference().child("meds");

Future<List> medNames(String currentName) async {
    List<String> nameList = [];
    try {
      await
```

```
dbRef.orderByChild("name").startAt(currentName).limitToFirst(10).once()
     .then((DataSnapshot snapshot) {
        Map<dynamic, dynamic> values = snapshot.value;
        values.forEach((key,values) {
          nameList.add(values["name"]);
          });
      });
      return nameList;
    }
    catch (e){
      print('caught error: $e');
    }
  }
```

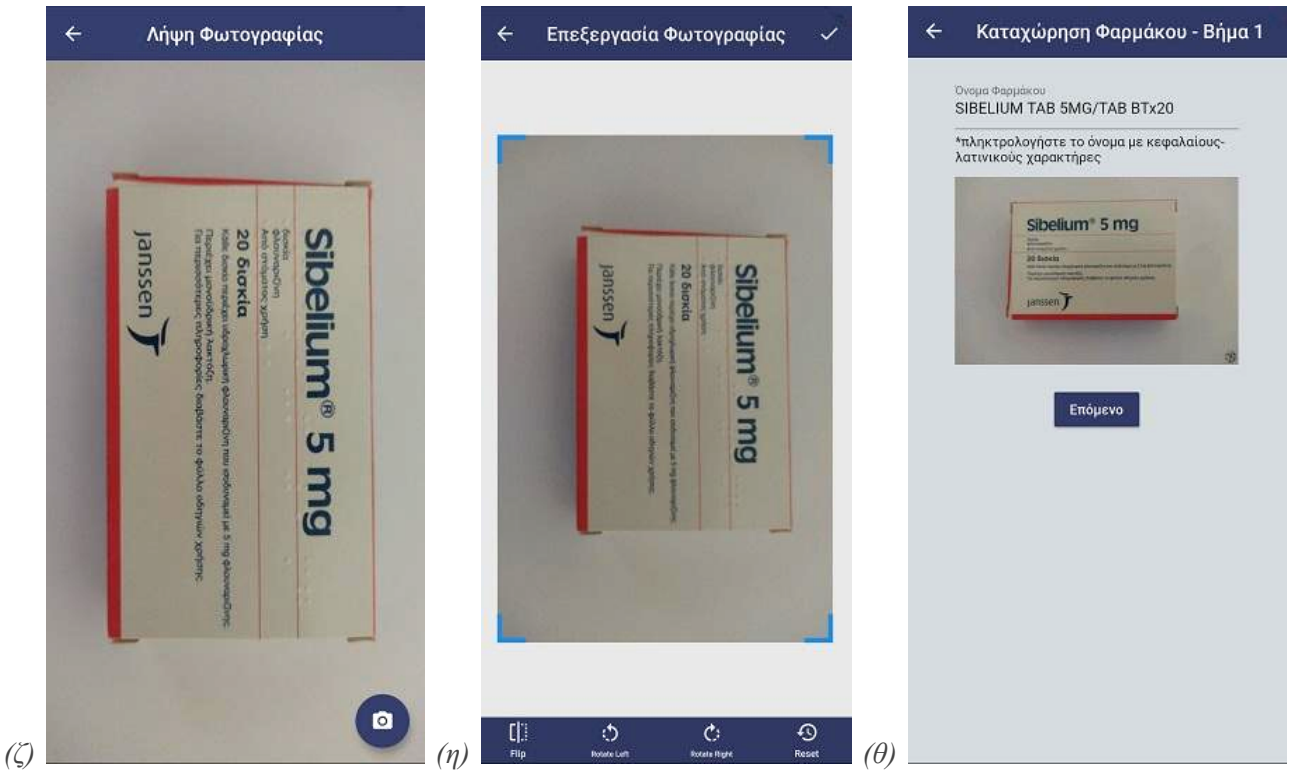
(δ) 

(ε) 

(στ) 

Εικόνα 20 Οθόνη φαρμάκων.(δ)Διαγραφή φαρμάκου, (ε) Βήμα 1 Drop down επιλογής ονόματος φαρμάκου και (στ) Επιλογή προσθήκης φωτογραφίας όταν δεν υπάρχει στην βάση

Αφού ο χρήστης επιλέξει το όνομα του φαρμάκου που θέλει να καταχωρήσει, εάν το φάρμακο αυτό έχει ήδη φωτογραφία στην βάση μας, του εμφανίζεται η φωτογραφία, αλλιώς του δίνεται η δυνατότητα εάν το επιθυμεί να καταχωρήσει ο ίδιος μια φωτογραφία πατώντας το κουμπί "Προσθέστε φωτογραφία" (Εικόνα 20στ). Επιλέγοντας την προσθήκη φωτογραφίας ενεργοποιείται η κάμερα της συσκευής του και μπορεί να τραβήξει φωτογραφία της συσκευασίας του φαρμάκου του. (Εικόνα 20ζ). Έπειτα. μπορεί να επεξεργαστεί την φωτογραφία που τράβηξε και πατώντας το tick icon που βρίσκεται δεξιά στο AppBar να καταχωρήσει την φωτογραφία στην βάση. (Εικόνα 20η). Αφού λοιπόν ολοκληρώσει την διαδικασία επιστρέφει στο Βήμα 1 της καταχώρησης όπου πλέον του εμφανίζεται η εικόνα που ο ίδιος αποθήκευσε και πατώντας το κουμπί "Επόμενο" μπορεί να προχωρήσει στο επόμενο βήμα της καταχώρησης (Εικόνα 20θ).



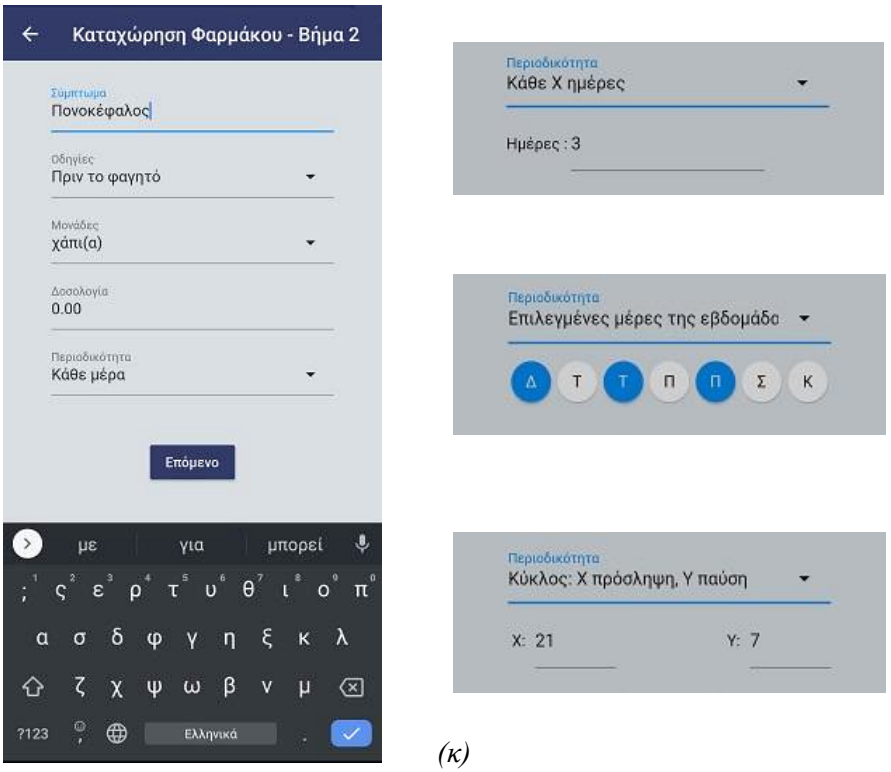
(ζ) (η) (θ)

Εικόνα 20 Οθόνη φαρμάκων.(ζ)Λήψη φωτογραφίας συσκευασίας, (η)Επεξεργασία φωτογραφίας, (θ) Επιστροφή στο Βήμα 1 της καταχώρησης μετά την αποθήκευση της φωτογραφίας στην βάση

Στο Βήμα 2, ο χρήστης μπορεί να συμπληρώσει εάν το επιθυμεί το σύμπτωμα που καταπολεμά με το συγκεκριμένο σκεύασμα και να προχωρήσει στις επιλογές που αφορούν τον τρόπο λήψης του φαρμάκου (Εικόνα 20ι). Αρχικά, από την drop down list που εμφανίζεται πατώντας πάνω στις ”Οδηγίες” μπορεί να κάνει μια από τις εξής επιλογές που έχουν να κάνουν με τις οδηγίες που του έχει δώσει ο ιατρός του σχετικά με την λήψη του φαρμάκου και τα γεύματά του: Πριν το φαγητό, Μετά το φαγητό, Κατά την διάρκεια του φαγητού και Δεν έχει σημασία.

Έπειτα πρέπει να ορίσει τις μονάδες μέτρησης του σκευάσματος επιλέγοντας από το drop down list που του εμφανίζεται πατώντας στο πεδίο “Μονάδες”: χάπι(α), κάψουλα/ες, κουταλάκι(α) του γλυκού, κουταλιά(ές) της σούπας, ένεση/εις, εισπνοή/ές, εφαρμογή/ες, σταγόνα/ες, υπόθετο/α, αμπούλα/ες, σακουλάκι(α), mg, mL, UI (Unité internationale). Στο επόμενο πεδίο, που είναι η “Δοσολογία” πληκτρολογεί την ποσότητα που του έχει συστήσει ο ιατρός του.

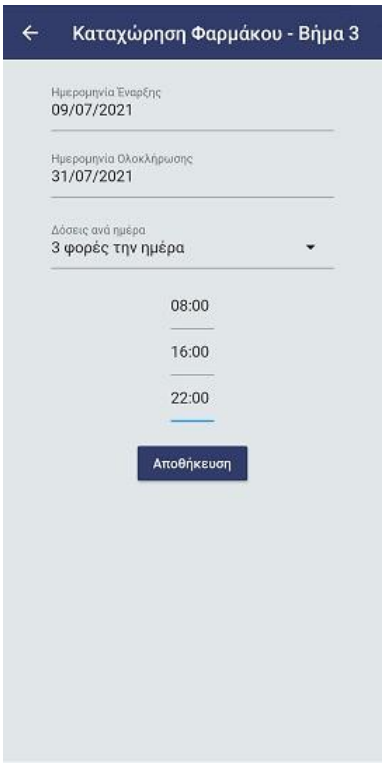
Τέλος, για αυτό το βήμα, επιλέγει από την drop down list που του εμφανίζεται πατώντας πάνω στο πεδίο “Περιοδικότητα” το σχήμα της αγωγής που πρέπει να ακολουθήσει: Κάθε μέρα, Μόνο μια μέρα, Κάθε Χ ημέρες, Επιλεγμένες μέρες της εβδομάδας, Κύκλο: Χ πρόσληψη, Υ παύση. Εάν ο χρήστης κάνει μια από τις τρεις τελευταίες επιλογές του εμφανίζει ένα ακόμα πεδίο που πρέπει να συμπληρώσει και αλλάζει αντίστοιχα για την κάθε επιλογή όπως φαίνεται στις παρακάτω εικόνες (Εικόνα 20κ).



(ι) *(κ)*

Εικόνα 20 Οθόνη φαρμάκων.(ι) Βήμα 2 καταχώρησης φαρμάκου, (κ) Περιοδικότητα κάθε Χ ημέρες, Περιοδικότητα Επιλεγμένες μέρες της εβδομάδας, Περιοδικότητα Κύκλος: Χ πρόσληψη, Υ παύση

Στο τρίτο και τελευταίο βήμα της καταχώρησης, ο χρήστης επιλέγει την ημερομηνία έναρξης και λήξης της αγωγής του (του δίνεται η δυνατότητα να επιλέξει ημερομηνία μεγαλύτερη της τρέχουσας για να διατηρηθεί η σωστή λειτουργία των ειδοποιήσεων). Επίσης, επιλέγει από το drop down list που του εμφανίζεται πατώντας πάνω στο πεδίο “Δόσεις ανά ημέρα” το πόσες φορές την ημέρα πρέπει να λαμβάνει μια δόση από το φάρμακο: 1 φορά την ημέρα, 2 φορές την ημέρα, 3 φορές την ημέρα, 4 φορές την ημέρα. Αναλόγως την επιλογή του εμφανίζονται τα αντίστοιχα πεδία στα οποία πρέπει να ορίσει την ώρα κάθε δόσης (Εικόνα 20λ).

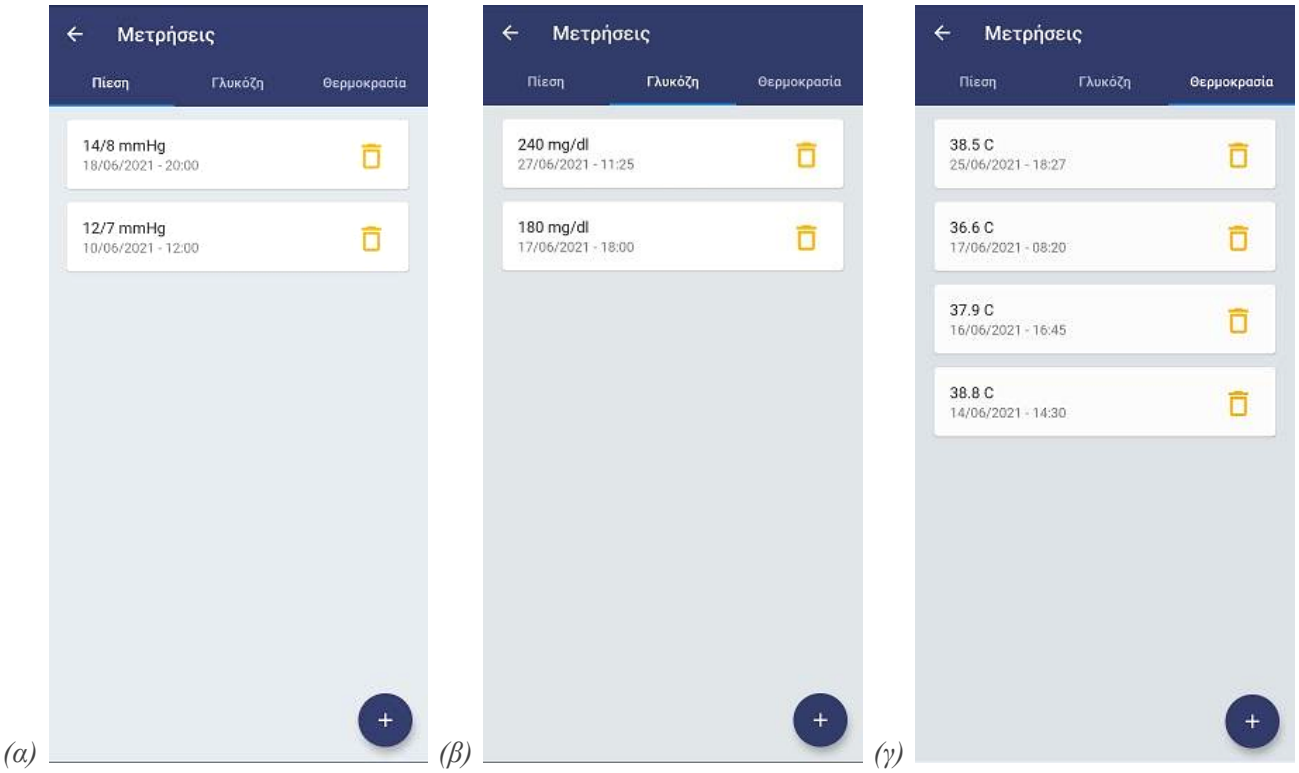
(λ) 

Εικόνα 20 Οθόνη φαρμάκων.(λ) Βήμα 3 καταχώρησης φαρμάκου

Πατώντας το κουμπί "Αποθήκευση" όταν ολοκληρώσει την διαδικασία καλείται η συνάρτηση MedDatabaseService(uid: user.uid).createMedData(med) [54] και ο χρήστης μεταφέρεται αυτόματα πίσω στην οθόνη των φαρμάκων όπου βλέπει ανανεωμένη την λίστα των φαρμάκων του.

5.3.2.3 Μετρήσεις

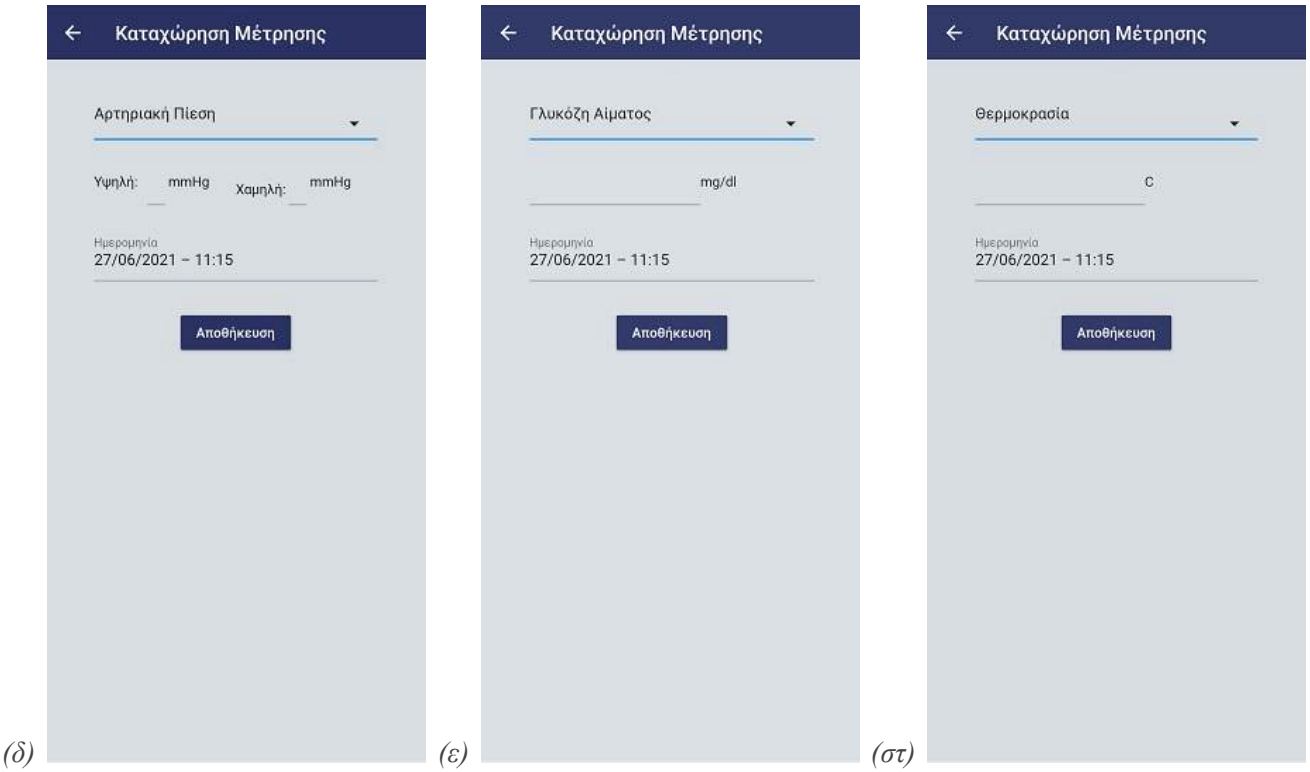
Στην οθόνη των μετρήσεων δίνεται η δυνατότητα στον χρήστη να καταχωρήσει τρία βασικά είδη μετρήσεων: την αρτηριακή του πίεση, τα επίπεδα της γλυκόζης στο αίμα του και την θερμοκρασία του. Επομένως, στην οθόνη αυτή έχουμε 3 καρτέλες: Πίεση (Εικόνα 21α), Γλυκόζη (Εικόνα 21β) και Θερμοκρασία (Εικόνα 21γ), όπου ανάλογα με το ποια έχει επιλέξει ο χρήστης του εμφανίζεται μια λίστα με τις μετρήσεις που έχει καταχωρήσει σε αυτήν την καρτέλα με φθίνουσα σειρά ημερομηνίας καταχώρησης, δηλαδή η πιο πρόσφατη είναι στην κορυφή της λίστας.



Εικόνα 21 Οθόνη μετρήσεων.(α) Λίστα μετρήσεων αρτηριακής πίεσης(β) Λίστα μετρήσεων γλυκόζης αίματος και (γ) Λίστα μετρήσεων θερμοκρασίας

Πατώντας το κουμπί "+" στο κάτω δεξιά μέρος της οθόνης, ο χρήστης μπορεί να καταχωρήσει μια νέα μέτρηση αρχικά επιλέγοντας από το drop down το είδος της μέτρησης: Αρτηριακή πίεση (Εικόνα 21δ), Γλυκόζη Αίματος (Εικόνα 21ε) και Θερμοκρασία (Εικόνα 21στ) και στην συνέχεια καταχωρώντας τις τιμές του και την ημέρα και ώρα της μέτρησης (by default στο πεδίο Ημερομηνία εμφανίζεται η τρέχουσα ημέρα και ώρα). Πατώντας το κουμπί "Αποθήκευση" η νέα καταχώρηση αποθηκεύεται στην βάση ανά περίπτωση μέσω των συναρτήσεων:

- MeasurementDatabaseService(uid: user.uid) .createPressureData(pressureHigh + '/' + pressureLow + ' mmHg', date) [54]
- MeasurementDatabaseService(uid: user.uid) .createGlucoseData(glucose + ' mg/dl', date) [54]
- MeasurementDatabaseService(uid: user.uid).createTemperatureData(temperature + ' C', date) [54]



(δ) *(ε)* *(στ)*

Εικόνα 21 Οθόνη μετρήσεων.(δ)Καταχώρηση αρτηριακής πίεσης, (ε) Καταχώρηση μέτρησης γλυκόζης αίματος και (στ) Καταχώρηση μέτρησης θερμοκρασίας

Τέλος, για κάθε καταχώρηση ο χρήστης μπορεί να προχωρήσει στην διαγραφή της από την λίστα πατώντας το πορτοκαλί delete icon στα δεξιά του ListTile. Αφού, του ζητηθεί να επιβεβαιώσει την επιλογή του (Εικόνα 21ζ) καλείται μία από τις εξής συναρτήσεις ανάλογα τον τύπο της μέτρησης:

- MeasurementDatabaseService(uid: user.uid).deletePressureData(pressureid) [54]
- MeasurementDatabaseService(uid: user.uid).deleteGlucoseData(glucoseid) [54]
- MeasurementDatabaseService(uid: user.uid).deleteTemperatureData(tempid) [54]

και ο χρήστης βλέπει πλέον την ανανεωμένη λίστα που δεν περιλαμβάνει το στοιχείο που μόλις διέγραψε.

(ζ) 

Εικόνα 21 Οθόνη μετρήσεων. (ζ)Διαγραφή μιας καταγραφής,

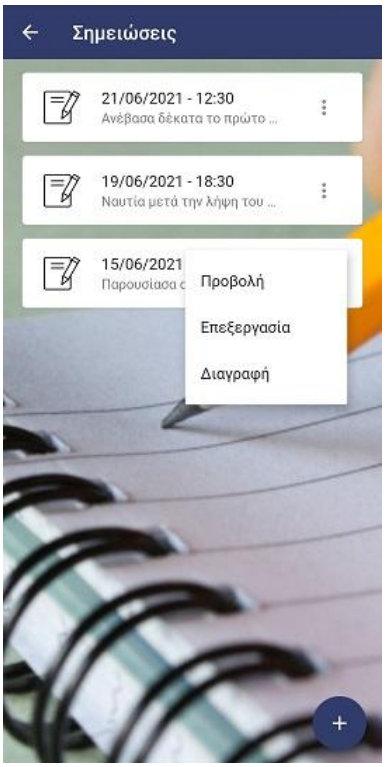
5.3.2.4 Σημειώσεις

Στην οθόνη των σημειώσεων (Εικόνα 22α) ο χρήστης έχει την δυνατότητα να καταγράψει σε μορφή σημειώσεων οτιδήποτε σχετίζεται με την πορεία της υγείας του ή είναι σχετικό με την φαρμακευτική αγωγή που λαμβάνει και ενδεχομένως θα ήθελε να μην ξεχάσει να αναφέρει στον θεράποντα ιατρό του στο επόμενο ραντεβού τους. Σε αυτή την οθόνη λοιπόν, ο χρήστης μπορεί να δει μια λίστα από τις σημειώσεις του ταξινομημένες από την πιο πρόσφατη στην πιο παλιά και πατώντας στις 3 τελείες στα δεξιά του ListTile να προβεί στις εξής ενέργειες για καθεμία από αυτές (Εικόνα 22β):

- Προβολή: Κάνοντας αυτή την επιλογή ο χρήστης μπορεί να δει ολόκληρο το κείμενο της σημείωσής του (Εικόνα 22γ).
- Επεξεργασία: Εδώ ο χρήστης μπορεί να τροποποιήσει το κείμενο και την ημερομηνία καταχώρησης της σημείωσής του (Εικόνα 22δ). Συνάρτηση NoteDatabaseService(uid: user.uid).updateNoteData(noteid, text, date) [54].
- Διαγραφή: Επιλέγοντας διαγραφή ο χρήστης αφού πρώτα επιβεβαιώσει την απόφασή του (Εικόνα 22ε), προχωράει στη διαγραφή μιας σημείωσης από την βάση με την συνάρτηση NoteDatabaseService(uid: user.uid).deleteNoteData(noteid) [54].

(α)

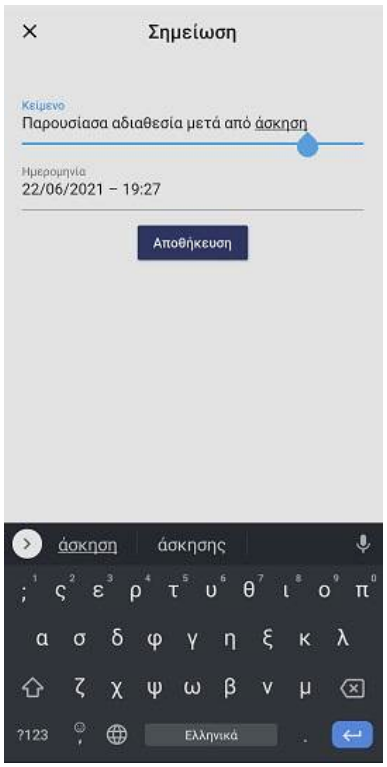
(β)



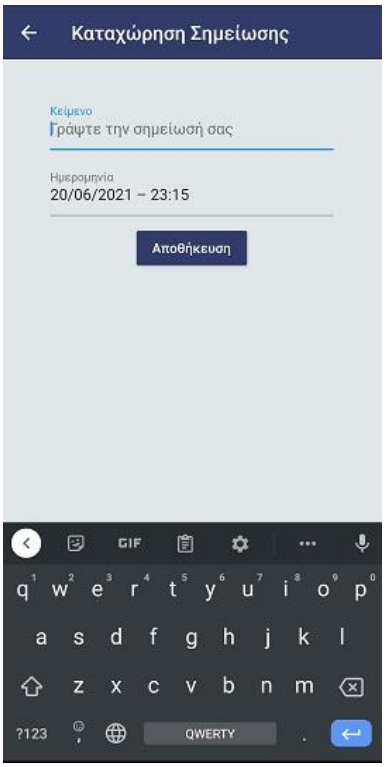
(γ)

Εικόνα 22 .Οθόνη σημειώσεων (α) Λίστα σημειώσεων (β) Λειτουργικότητες της οθόνης σημειώσεων και (γ) Λεπτομέρειες εγγραφής σημείωσης

Τέλος, ο χρήστης μπορεί πατώντας το floatingActionButton στο κάτω μέρος της οθόνης να καταχωρήσει μια νέα σημείωση γράφοντας το κείμενο που επιθυμεί στο πρώτο πεδίο και την ημερομηνία καταχώρησης στο δεύτερο (Εικόνα 22στ). Εάν δεν επιλέξει να τροποποιήσει την ημερομηνία, αποθηκεύεται στην βάση ως ημερομηνία καταχώρησης η τρέχουσα ημέρα και ώρα που είναι και η default επιλογή.

(δ) 

(ε) 

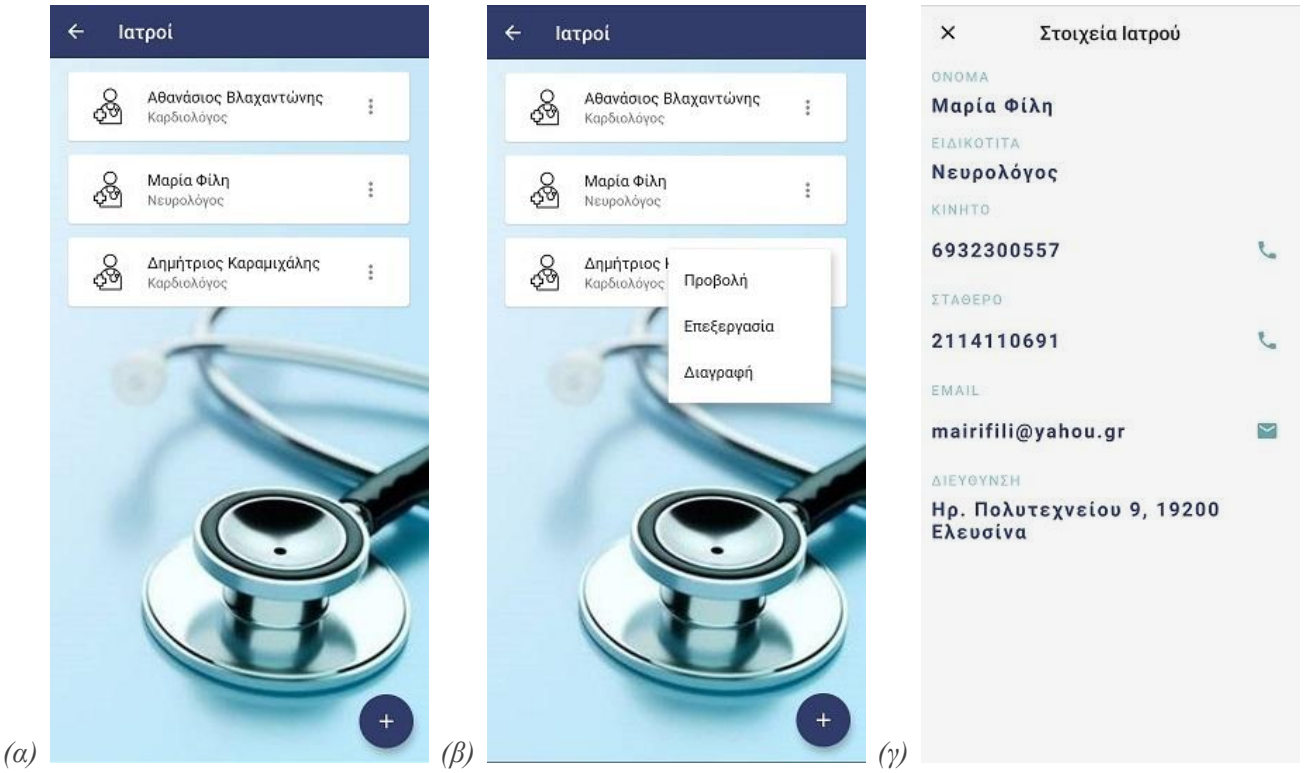
(στ) 

Εικόνα 22. Οθόνη σημειώσεων.(δ) Επεξεργασία σημείωσης, (ε) Διαγραφή σημείωσης και (στ) Καταχώρηση νέας σημείωσης

5.3.2.5 Ιατροί

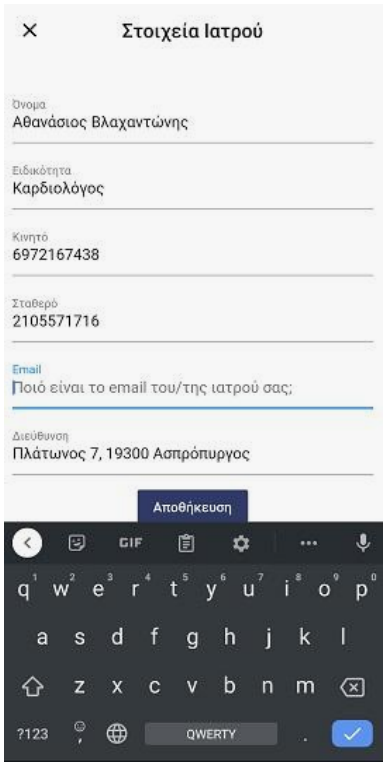
Στην οθόνη των ιατρών (Εικόνα 23α) ο χρήστης μπορεί να δημιουργήσει μια λίστα με όλους τους θεράποντες ιατρούς του ώστε να βρίσκει εύκολα τα στοιχεία τους όταν παρουσιαστεί ανάγκη και να επικοινωνεί μαζί τους. Οι ενέργειες που μπορεί να κάνει ο χρήστης για κάθε καταχώρηση ιατρού (Εικόνα 23β) είναι οι εξής:

- Προβολή: Στην οθόνη που εμφανίζεται (Εικόνα 23γ) πατώντας προβολή ο χρήστης βλέπει αναλυτικά τα στοιχεία του θεράποντα ιατρού του ενώ πατώντας στα εικονίδια δίπλα στα τηλέφωνα ή στο email ανακατευθύνεται στην αντίστοιχη εφαρμογή της συσκευής του ώστε να επικοινωνήσει άμεσα.
- Επεξεργασία: Με αυτή την επιλογή ο χρήστης μπορεί να κάνει αλλαγές στα στοιχεία του ιατρού του και να διορθώσει τυχόν λάθη (Εικόνα 23δ). Η συνάρτηση που χρησιμοποιείται για την λειτουργικότητα αυτή είναι η DoctorDatabaseService(uid: user.uid).updateDoctorData(doctorid, name, specialty, mobile, phone, email,address) [54].
- Διαγραφή: Πατώντας στην επιλογή διαγραφή ο χρήστης μπορεί να διαγράψει (αφού πρώτα του ζητηθεί επιβεβαίωση) από την λίστα τον συγκεκριμένο ιατρό (Εικόνα 23ε). Η συνάρτηση που υλοποιεί την διαγραφή είναι η DoctorDatabaseService(uid: user.uid).deleteDoctorData(doctorid) [54].

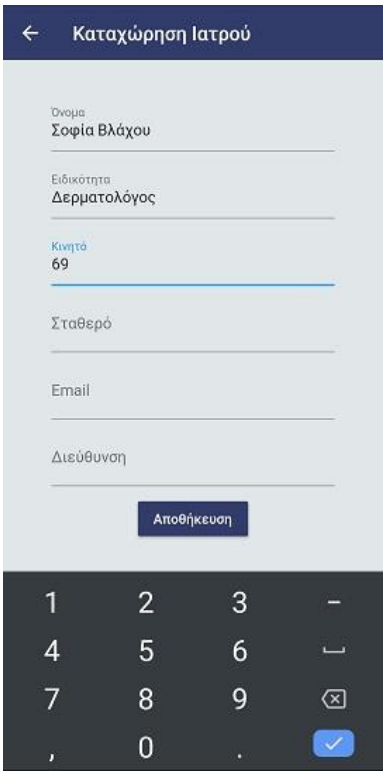


Εικόνα 23. Οθόνη ιατρών .(α) Λίστα ιατρών (β) Λειτουργικότητες οθόνης ιατρών και (γ) Λεπτομέρειες εγγραφής ιατρού

Τέλος, ο χρήστης μπορεί πατώντας το floatingActionButton στο κάτω μέρος της οθόνης να καταχωρήσει έναν νέο ιατρό στην λίστα του (Εικόνα 23στ). Τα στοιχεία που μπορεί να συμπληρώσει είναι το όνομα, η ειδικότητα, το κινητό τηλέφωνο, το σταθερό, το email και η διεύθυνση του ιατρού. Να σημειωθεί ότι στα πεδία κινητό και σταθερό τηλέφωνο ο χρήστης επιτρέπεται να πληκτρολογήσει μόνο νούμερα (επιτυγχάνεται με χρήση της παραμέτρου keyboardType: TextInputType.number στο TextFormField) [54]καθώς και ότι αυτά πρέπει να έχουν ακριβώς 10 ψηφία, αλλιώς θα εμφανιστεί στον χρήστη μήνυμα λάθους όταν προσπαθήσει να κάνει αποθήκευση. Πατώντας το κουμπί “Αποθήκευση” η νέα καταχώρηση αποθηκεύεται στην βάση μέσω της συνάρτησης DoctorDatabaseService(uid: user.uid).createDoctorData(name, specialty, mobile, phone, email,address) [54] και ο χρήστης επιστρέφει αυτόματα στην οθόνη με την λίστα των ιατρών του (Εικόνα 23α) όπου την βλέπει ήδη ανανεωμένη με την προσθήκη του νέου ιατρού.

(δ) 

(ε) 

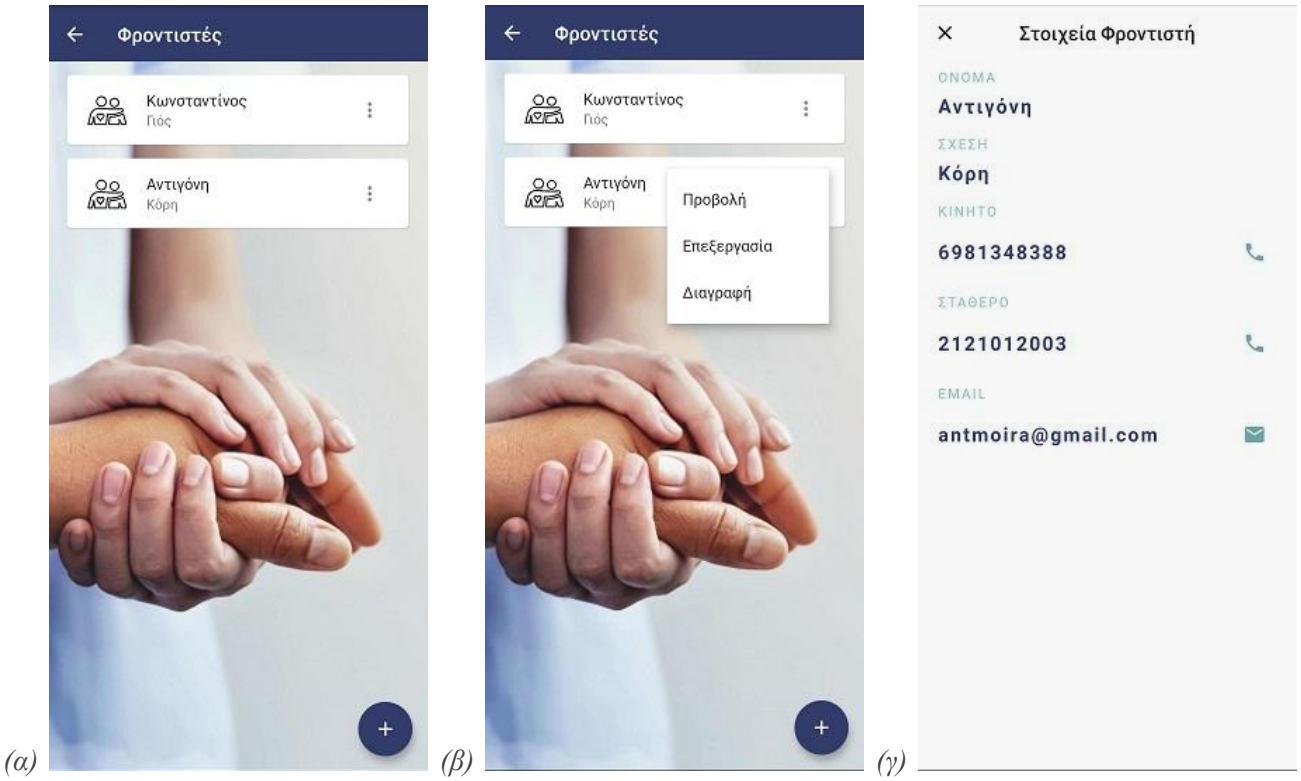
(στ) 

Εικόνα 23. Οθόνη ιατρών .(δ) Επεξεργασία ιατρού (ε) Διαγραφή ιατρού και (στ) Καταχώρηση νέου ιατρού

5.3.2.6 Φροντιστές

Στην οθόνη των φροντιστών (Εικόνα 24α), ο χρήστης μπορεί να δημιουργήσει μια λίστα με τα άτομα που τον φροντίζουν, ώστε να μπορεί να βρει τα στοιχεία επικοινωνίας τους εύκολα και να τα μοιραστεί σε περίπτωση ανάγκης. Οι ενέργειες που μπορεί να κάνει ο χρήστης για κάθε καταχώρηση φροντιστή (Εικόνα 24β) είναι οι εξής:

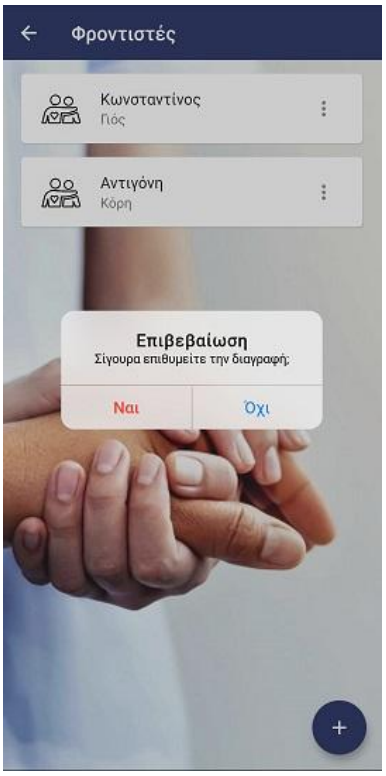
- Προβολή: Πατώντας σε αυτή την επιλογή ο χρήστης μπορεί να δει τα στοιχεία του οικείου του (Εικόνα 24γ) και πατώντας στα εικονίδια του τηλεφώνου ή του email να επικοινωνήσει μαζί του.
- Επεξεργασία: Εδώ ο χρήστης τροποποιεί τα στοιχεία ενός φροντιστή που έχει ήδη καταχωρήσει (Εικόνα 24δ). Η αλλαγή αυτή πραγματοποιείται μέσω της συνάρτησης AssistantDatabaseService(uid: user.uid) .updateAssistantData(assistantid, name,.relationship, mobile, phone, email) [54].
- Διαγραφή: Με την επιλογή διαγραφή ο χρήστης μπορεί να διαγράψει (αφού πρώτα του ζητηθεί επιβεβαίωση) από την λίστα τον συγκεκριμένο φροντιστή (Εικόνα 24ε). Η συνάρτηση που υλοποιεί την διαγραφή είναι η AssistantDatabaseService(uid: user.uid).deleteAssistantData(assistantid) [54].

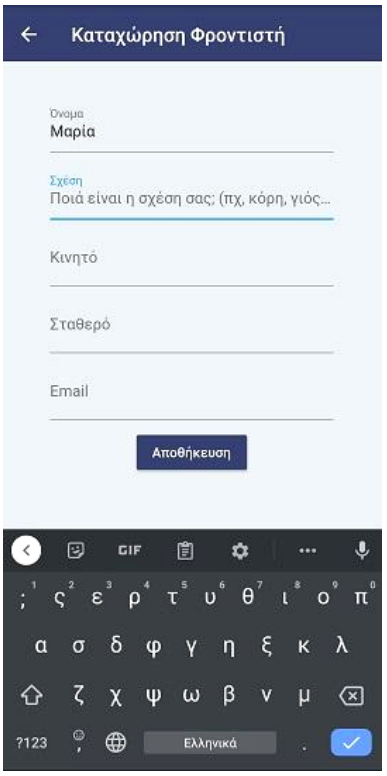


Εικόνα 24. Οθόνη φροντιστών. (α) Λίστα φροντιστών,(β) Λειτουργικότητες οθόνης φροντιστών και (γ) Λεπτομέρειες εγγραφής φροντιστή

Η καταχώρηση ενός επιπλέον ατόμου ως φροντιστή του ασθενούς γίνεται πατώντας το floatingActionButton στο κάτω δεξιά στην οθόνη. Ο χρήστης μεταφέρεται στην οθόνη που εμφανίζει την φόρμα καταχώρησης φροντιστή και εκεί καλείται να συμπληρώσει όσα από τα πέντε πεδία επιθυμεί: όνομα, την σχέση που έχει με τον συγκεκριμένο άνθρωπο, το κινητό και το σταθερό του τηλέφωνό και το email του. Στα πεδία κινητό και σταθερό επιτρέπεται μόνο η εισαγωγή αριθμών και πρέπει να αποτελούνται από ακριβώς 10 ψηφία, αλλιώς θα εμφανιστεί στον χρήστη μήνυμα λάθους κατά την αποθήκευση. Για να ολοκληρωθεί η διαδικασία ο πρέπει να πατηθεί το κουμπί “Αποθήκευση” που καλεί την συνάρτηση AssistantDatabaseService(uid: user.uid) .createAssistantData(name,.relationship, mobile, phone, email) [54] με ορίσματα τα στοιχεία που κατέγραψε ο χρήστης.

(δ) 

(ε) 

(στ) 

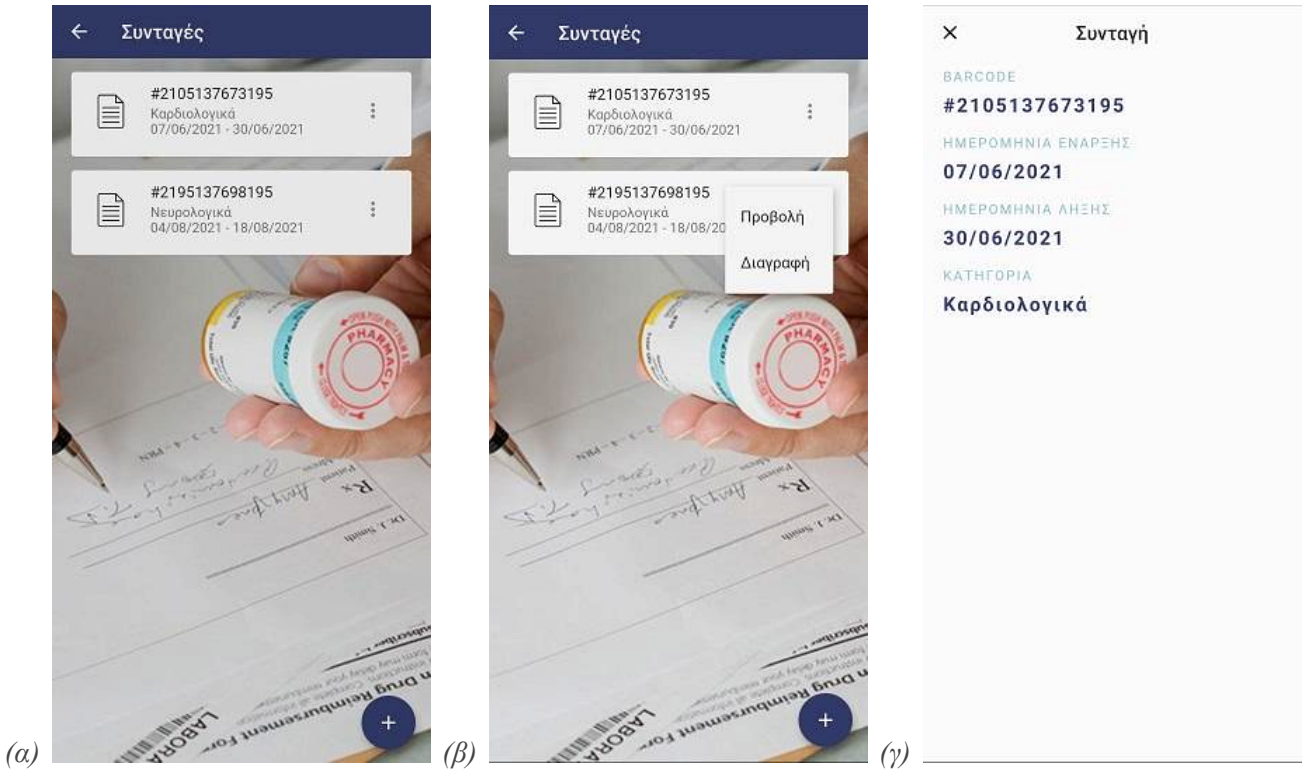
Εικόνα 24. Οθόνη φροντιστών .(δ) Επεξεργασία φροντιστή,(ε) Διαγραφή φροντιστή και (στ) Καταχώρηση νέου φροντιστή

5.3.2.7 Συνταγές

Στην οθόνη των συνταγών (Εικόνα 25α) ο χρήστης μπορεί να καταχωρήσει μια λίστα με τις συνταγές που του έχουν γράψει οι θεράποντες ιατροί με την φαρμακευτική αγωγή που πρέπει να λαμβάνει ανά μήνα. Οι συνταγές εμφανίζονται σε μια λίστα με αύξουσα ημερομηνία έναρξης εκτέλεσης, ώστε πιο πάνω στην λίστα να εμφανίζονται οι συνταγές που πρέπει να εκτελεστούν πρώτες. Ο ασθενής επομένως, πηγαίνοντας σε ένα φαρμακείο οποιαδήποτε ημέρα μεταξύ της ημερομηνίας έναρξης και λήξης της συνταγής, μπορεί να δώσει απλά το δεκατριψήφιο barcode της και να πάρει τα φάρμακά του. Η εφαρμογή στέλνει ειδοποίηση στον χρήστη στις 10:00 την ημέρα έναρξης και την ημέρα λήξης της συνταγής ώστε να μειωθεί η πιθανότητα να ξεχάσει να εκτελέσει μια συνταγή και να έχει ελλείψεις στο απόθεμα των σκευασμάτων της αγωγής του. Σε αυτή την οθόνη δίνονται δύο επιλογές στον χρήστη (Εικόνα 25β):

- Προβολή: Ώστε να μπορεί να δει όλες τις πληροφορίες που έχει καταχωρήσει και αφορούν την συγκεκριμένη συνταγή (Εικόνα 25γ)
- Διαγραφή: Όταν εκτελέσει την συνταγή του ο ασθενής μπορεί επιλέγοντας την διαγραφή να διαγράψει την συγκεκριμένη συνταγή από την λίστα του (Εικόνα 25δ). Η λειτουργία αυτή επιτυγχάνεται με την συνάρτηση PrescriptionDatabaseService(uid: user.uid).deletePrescriptionData(prescriptionid) [54].

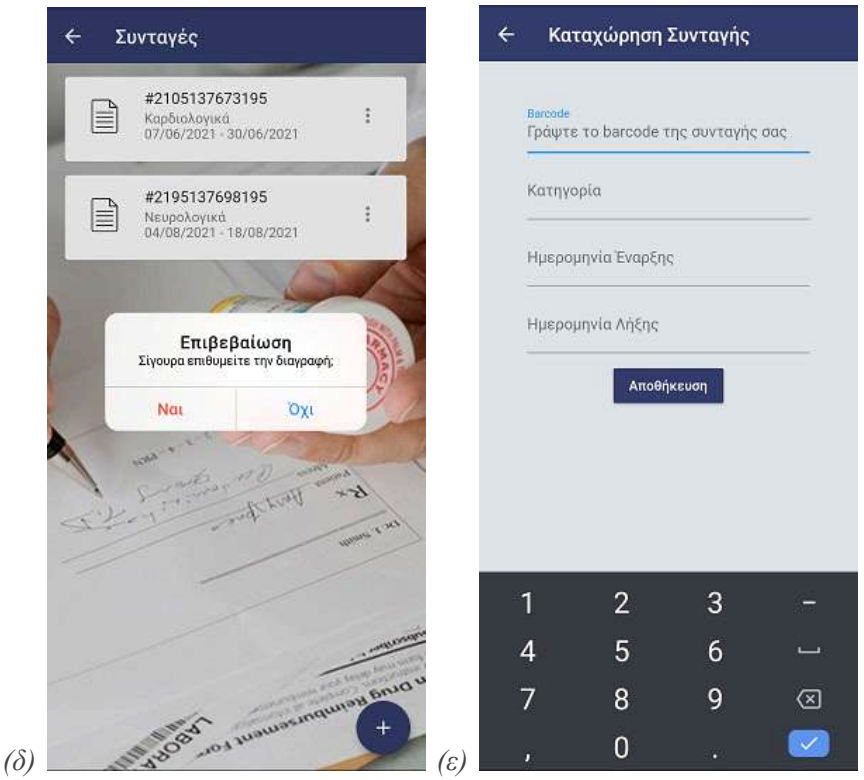
Εδώ επιλέξαμε να μην δίνουμε την δυνατότητα στον χρήστη να κάνει επεξεργασία των πληροφοριών κάθε συνταγής που έχει καταχωρήσει, καθώς τα στοιχεία αυτά παραμένουν αμετάβλητα από την στιγμή που γράφεται από τον ιατρό, οπότε μια τέτοια ανάγκη είναι απίθανο να προκύψει.



Εικόνα 25. Οθόνη συνταγών. (α) Λίστα συνταγών, (β) Λειτουργικότητες οθόνης συνταγών και (γ) Λεπτομέρειες εγγραφής συνταγής

Τέλος, ο χρήστης μπορεί πατώντας το floatingActionButton στο κάτω μέρος της οθόνης να καταχωρήσει μία νέα συνταγή στην λίστα του (Εικόνα 25ε). Τα στοιχεία που πρέπει να συμπληρώσει είναι το barcode της συνταγής, η ημερομηνία έναρξης και λήξης της και προεραιτικά μια σημείωση που μπορεί να είναι για παράδειγμα το όνομα του ιατρού που του την έγραψε ή το είδος των σκευασμάτων που αφορά η συγκεκριμένη συνταγή, ώστε να του είναι πιο εύκολο να διαχειριστεί το απόθεμα των φαρμάκων του (πχ να δει ότι δεν έχει άλλη συνταγή για τα καρδιολογικά του φάρμακα και να επισκεφτεί τον ιατρό του για να του τα συνταγογραφήσει για τους επόμενους μήνες).

Να σημειωθεί ότι στο πεδίο του barcode ο χρήστης επιτρέπεται να πληκτρολογήσει μόνο νούμερα (επιτυγχάνεται με χρήση της παραμέτρου keyboardType: TextInputType.number στο TextFormField) καθώς και ότι αυτό πρέπει να αποτελείται από ακριβώς 13 ψηφία, αλλιώς θα εμφανιστεί στον χρήστη μήνυμα λάθους όταν προσπαθήσει να κάνει αποθήκευση. Πατώντας το κουμπί “Αποθήκευση” η νέα καταχώρηση αποθηκεύεται στην βάση μέσω της συνάρτησης PrescriptionDatabaseService(uid: user.uid).createPrescriptionData(barcode, dateStart, dateEnd, category) [54] και ο χρήστης επιστρέφει αυτόματα στην οθόνη με την λίστα των συνταγών του (Εικόνα 25α) όπου την βλέπει ήδη ανανεωμένη με την προσθήκη της νέας συνταγής.



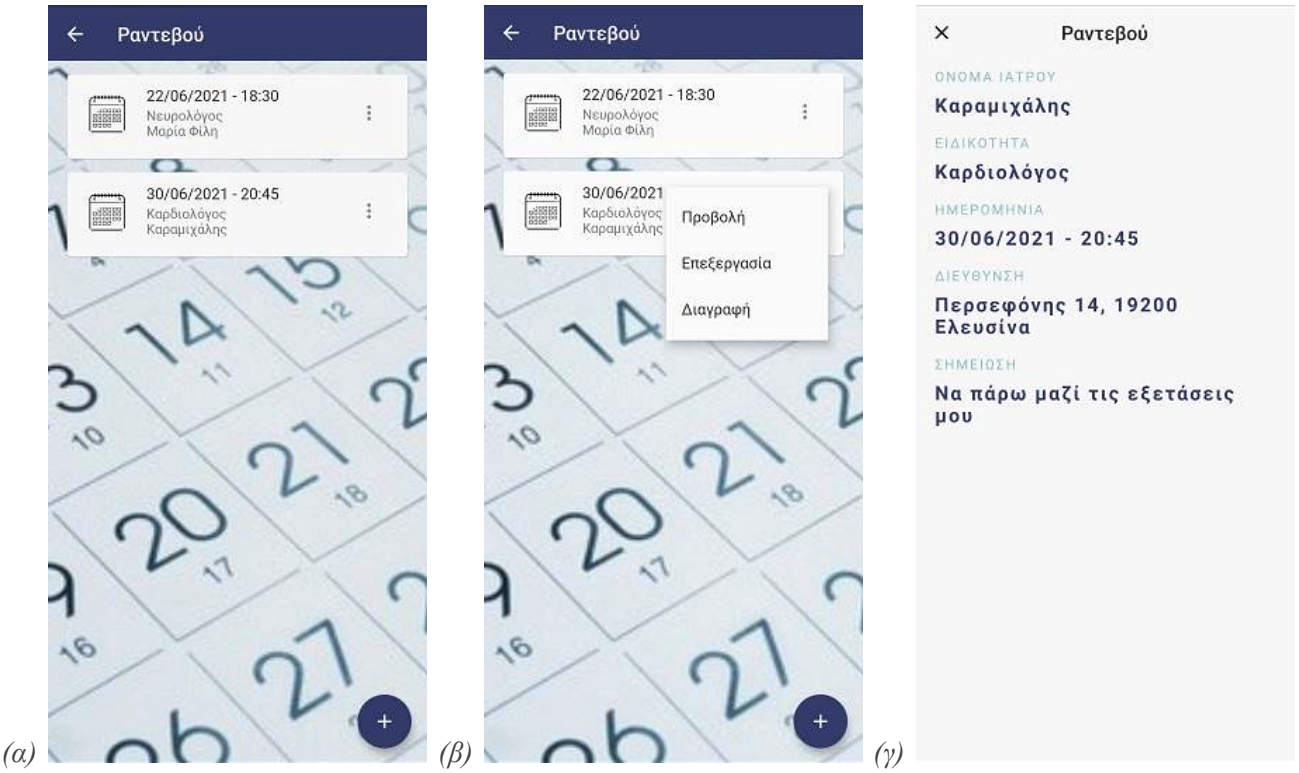
(δ) *(ε)*

Εικόνα 25. Οθόνη συνταγών.(δ) Διαγραφή συνταγής και (ε) Καταχώρηση νέας συνταγής

5.3.2.8 Ραντεβού

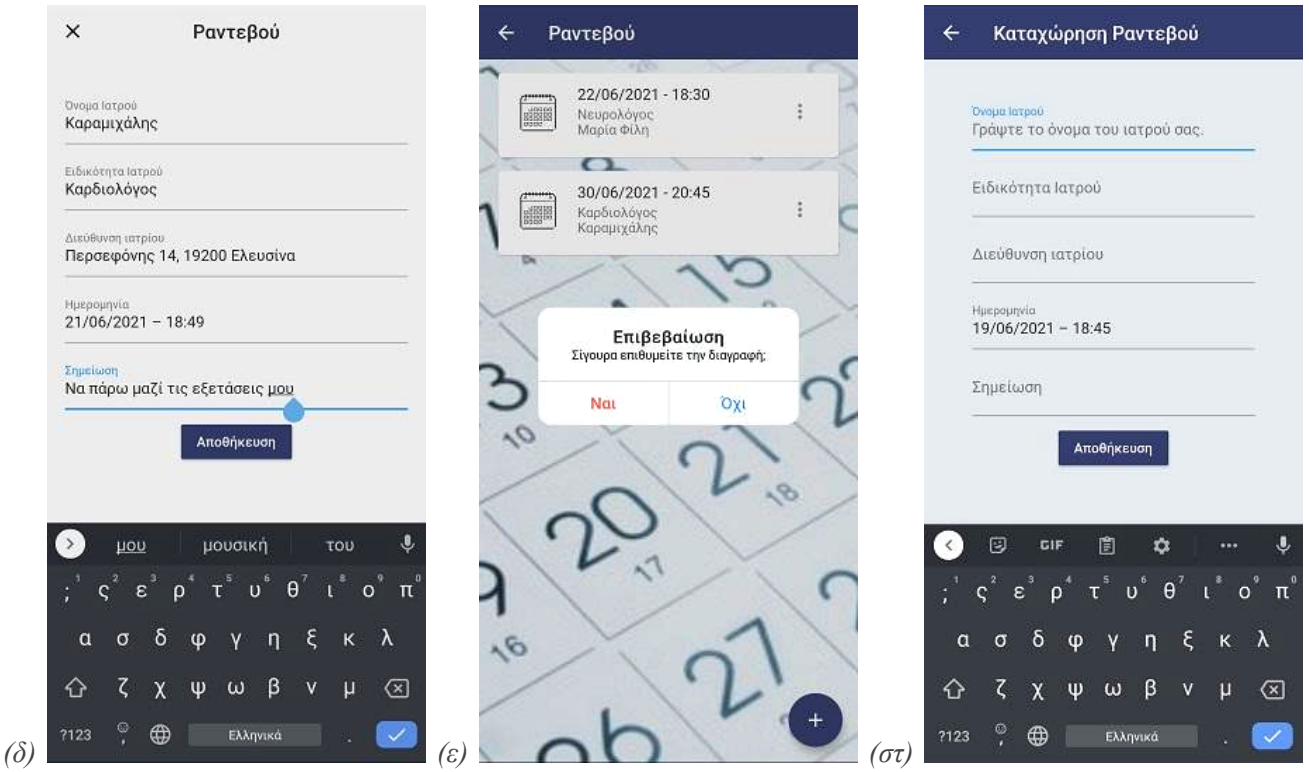
Ο χρήστης επιλέγοντας από το μενού της εφαρμογής τα "Ραντεβού" μπορεί να δει μια λίστα με τα προγραμματισμένα του ραντεβού με ιατρούς ή με διαγνωστικά κέντρα για εξετάσεις (Εικόνα 26α). Η λίστα αυτή είναι ταξινομημένη ώστε το πιο χρονικά κοντινό ραντεβού να εμφανίζεται στην κορυφή της λίστας. Πατώντας στο icon με τις τρεις τελείες στο δεξί άκρο κάθε ListTile δίνεται στον χρήστη η δυνατότητα να εκτελέσει κάποια από τις παρακάτω ενέργειες (Εικόνα 26β):

- Προβολή: Εδώ ο χρήστης μπορεί να δει τις λεπτομέρειες του ραντεβού του (Εικόνα 26γ).
- Επεξεργασία: Ο χρήστης μπορεί να τροποποιήσει τα στοιχεία ενός ραντεβού που έχει ήδη καταχωρήσει (Εικόνα 26δ). Η τροποποίηση αυτή πραγματοποιείται μέσω της συνάρτησης AppointmentDatabaseService(uid: user.uid).updateAppointmentData(appointmentid, doctorsname, specialty, address, date, note) [54].
- Διαγραφή: Με την επιλογή διαγραφή ο χρήστης μπορεί να διαγράψει (αφού πρώτα του ζητηθεί επιβεβαίωση) από την λίστα το συγκεκριμένο ραντεβού (Εικόνα 26ε). Η συνάρτηση που υλοποιεί την διαγραφή είναι η AppointmentDatabaseService(uid: user.uid).deleteAppointmentData(appointmentid) [54].



Εικόνα 26. Οθόνη ραντεβού. (α) Λίστα ραντεβού,(β) Λειτουργικότητες οθόνης ραντεβού και (γ) Λεπτομέρειες εγγραφής ραντεβού

Τέλος, ο χρήστης μπορεί πατώντας το floatingActionButton στο κάτω μέρος της οθόνης να καταχωρήσει ένα νέο ραντεβού στην λίστα του (Εικόνα 26στ). Τα στοιχεία που μπορεί να συμπληρώσει είναι το όνομα και η ειδικότητα του ιατρού, η διεύθυνση του ιατρίου του, η ημερομηνία και η ώρα του ραντεβού καθώς και μία σημείωση, όπου μπορεί να γράψει οποιαδήποτε επιπλέον πληροφορία του χρειάζεται για το ραντεβού, όπως για παράδειγμα οδηγίες που του έχουν δοθεί αν πρόκειται για μία εξέταση, το barcode της συνταγής εάν έχει γράψει την εξέταση ή ότι του έχει ζητήσει ο ιατρός του να φέρει μαζί στο ραντεβού τους. Πατώντας το κουμπί "Αποθήκευση" η νέα καταχώρηση αποθηκεύεται στην βάση μέσω της συνάρτησης AppointmentDatabaseService(uid: user.uid).createAppointmentData(doctorsname, specialty, address, date, note) [54] και ο χρήστης επιστρέφει αυτόματα στην οθόνη με την λίστα των ραντεβού του (Εικόνα 26α) όπου την βλέπει ήδη ανανεωμένη με την προσθήκη του νέου ραντεβού.

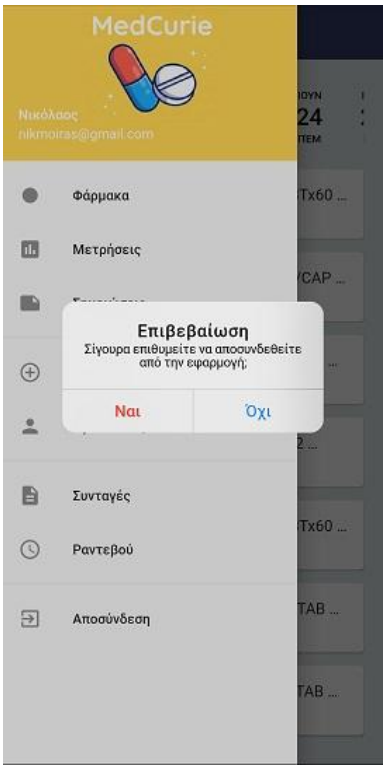


(δ) *(ε)* *(στ)*

Εικόνα 26. Οθόνη ραντεβού .(δ) Επεξεργασία ραντεβού,(ε) Διαγραφή ραντεβού και (στ) Καταχώρηση νέου ραντεβού

5.3.3 Αποσύνδεση

Ο χρήστης κάνοντας κλικ στην επιλογή “Αποσύνδεση” του μενού καλείται να επιβεβαιώσει την επιλογή του στο popup που εμφανίζεται (Εικόνα 27), ώστε να γίνει δυσκολότερη η αποσύνδεση του χρήστη κατά λάθος. Μετά την αποσύνδεση, ο χρήστης επιστρέφει στην οθόνη υποδοχής που αναλύσαμε παραπάνω, καθώς πλέον το user stream είναι κενό και ο current user είναι null.

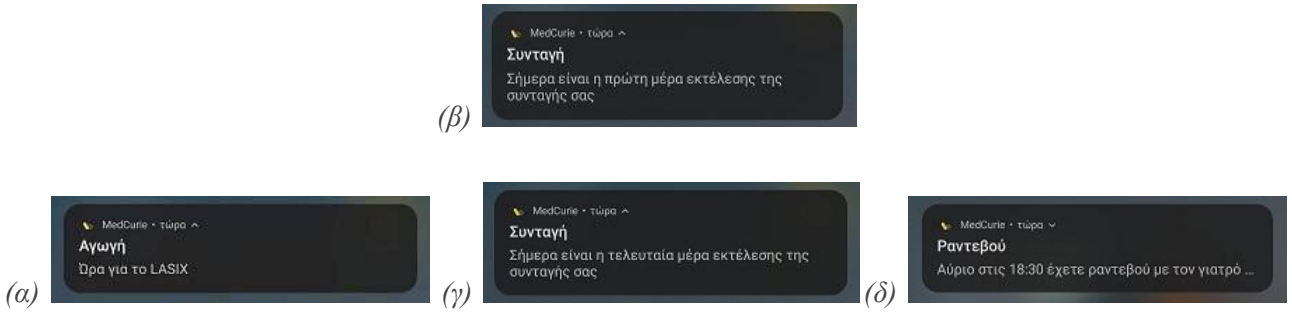


Εικόνα 27. Οθόνη αποσύνδεσης

5.3.4 Ειδοποιήσεις

Όταν είναι η ώρα να υπενθυμίσει η εφαρμογή ένα γεγονός στον χρήστη εμφανίζεται στην οθόνη της συσκευής του μια από τις εξής ειδοποιήσεις:

- Αγωγή: Όταν είναι η ώρα ο ασθενής να πάρει μια δόση από ένα φάρμακο που έχει καταχωρήσει, λαμβάνει ειδοποίηση (Εικόνα 28α) μέσω του εργαλείου Firebase Cloud Messaging (FCM) με την διαδικασία που παρουσιάστηκε στην Ενότητα 5.2.2.3. Η ειδοποίηση τον ενημερώνει ότι ήρθε η ώρα να πάρει το φάρμακό του, αναφέροντας και το όνομα του φαρμάκου. Πατώντας πάνω της η εφαρμογή ανοίγει στην αρχική οθόνη στην τρέχουσα ημερομηνία (Εικόνα 18α), ώστε ο χρήστης να μπορεί να δει περισσότερες λεπτομέρειες σχετικά με την δόση του φαρμάκου του και στην συνέχεια να την σβήσει από την λίστα με τα φάρμακα που έχει να πάρει για σήμερα.
- Συνταγή: Την ημέρα έναρξης και την ημέρα λήξης μιας συνταγής αποστέλλεται στον χρήστη που την έχει καταχωρήσει μια ειδοποίηση στις 10:00 (Εικόνα 28β και 28γ αντίστοιχα). Πατώντας πάνω σε αυτή την ειδοποίηση η εφαρμογή ανοίγει στην οθόνη των συνταγών (Εικόνα 25α), ώστε ο χρήστης να μπορέσει να δει περισσότερες πληροφορίες σχετικά με την συνταγή του.
- Ραντεβού: Μια ημέρα πριν το προγραμματισμένο του ραντεβού με έναν ιατρό ο χρήστης λαμβάνει μια ειδοποίηση στις 12:00 που τον ενημερώνει ότι αύριο έχει ένα ραντεβού καθώς και για την ώρα του ραντεβού (Εικόνα 28δ). Πατώντας πάνω στην ειδοποίηση ο χρήστης οδηγείται στην οθόνη των Ραντεβού στης εφαρμογής (Εικόνα 26α) για να δει όποια επιπλέον πληροφορία χρειάζεται για το ραντεβού του.



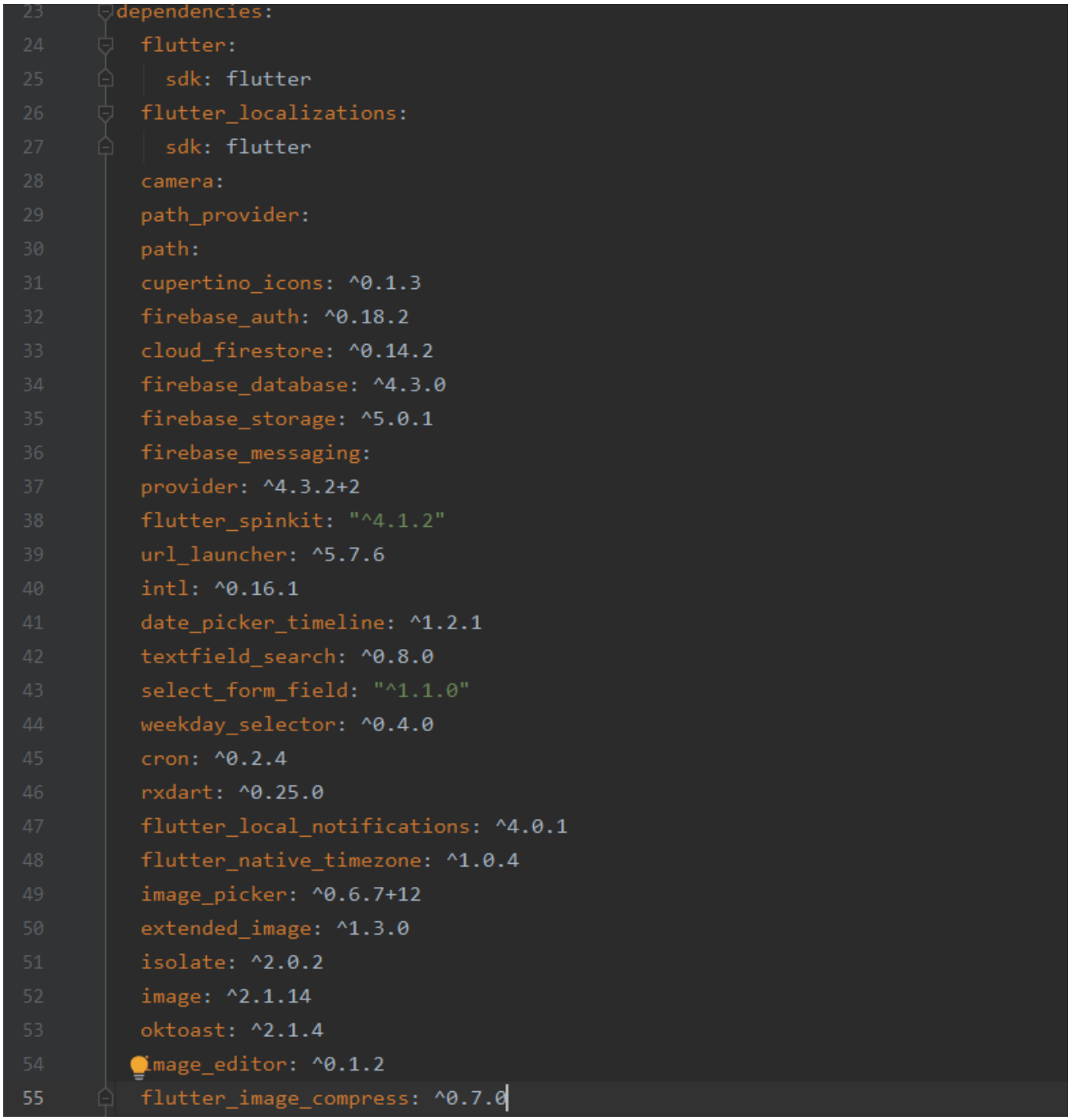
Εικόνα 28. Ειδοποιήσεις. (α) Ειδοποίηση λήψης Αγωγής, (β) Ειδοποίηση έναρξης συνταγής, (γ) Ειδοποίηση λήξης συνταγής και (δ) Ειδοποίηση αυριανού ραντεβού με ιατρό

5.4 Πακέτα (packages) & Βιβλιοθήκες

Αρχικά για την ταυτοποίηση του χρήστη (user authentication) χρησιμοποιήσαμε το πακέτο “firebase_auth”, προκειμένου να υλοποιήσουμε την μέθοδο ταυτοποίησης “email/password” που παρέχει η πλατφόρμα του Firebase. Επίσης, με το πακέτο “firabase_database” μπορέσαμε να έχουμε πρόσβαση στην Realtime Database “meds”, με τις πληροφορίες για κάθε φάρμακο. Για να χρησιμοποιήσουμε το Firebase Cloud Storage API και να μπορεί ο χρήστης να ανεβάσει τις φωτογραφίες των φαρμάκων στον φάκελο meds, εντάξαμε στην εφαρμογή μας το πακέτο “firebase_storage”. Τέλος, για την αποθήκευση όλων των δεδομένων του κάθε χρήστη αξιοποιήσαμε την Firestore Database όπως έχει ήδη αναφερθεί στην Ενότητα 5.2.2.3. Προκειμένου λοιπόν, να έχει η εφαρμογή πρόσβαση σε αυτή την NoSQL βάση και να μπορεί να δημιουργεί συλλογές και έγγραφα, να ανακτά δεδομένα, να τα διαγράφει ή να τα ανανεώνει ήταν απαραίτητη η χρήση του πακέτου “cloud_firestore”. Τέλος, από την εργαλειοθήκη του Firebase εγκαταστήσαμε το πακέτο “firebase_messaging”, ώστε να μπορεί η εφαρμογή του χρήστη να διαχειρίζεται και να του εμφανίζει τα push notifications που στέλνει η πλατφόρμα του Firebase.

Για να μπορεί ο χρήστης να χρησιμοποιήσει την κάμερα και να τραβήξει φωτογραφίες εντός της εφαρμογής έγινε χρήση των πακέτων “camera”, “path_provider” και “path”. Ενώ για να του δώσουμε την δυνατότητα να επεξεργαστεί την φωτογραφία προτού την ανεβάσει επιλέξαμε το πακέτο “image_editor”, καθώς και κάποια επιπλέον βοηθητικά αυτού όπως τα “extended_image”, “image”, “isolate” και “oktoast”. Τέλος, για ό,τι αφορά την φωτογραφία του φαρμάκου, χρησιμοποιήσαμε το πακέτο “flutter_image_compress” για να μειώσουμε το μέγεθός της (quality: 75%) και να καταλαμβάνει λιγότερο χώρο στο Firebase Storage.

Χρησιμοποιήσαμε αρκετά ακόμα πακέτα του οικοσυστήματος Flutter τα οποία έχουν σχέση κυρίως με την παροχή έτοιμων widgets για την διαμόρφωση του interface της εφαρμογής. Όλα τα παραπάνω πακέτα δηλώθηκαν και εγκαταστάθηκαν μέσω του αρχείου “pubspec.yaml” του κώδικα της εφαρμογής, όπως φαίνεται και στην Εικόνα 29.



Εικόνα 29. Τα πακέτα που χρησιμοποιήθηκαν για την ανάπτυξη της εφαρμογής MedCurie, όπως φαίνονται στο αρχείο 'pubspec.yaml' [54]

Κεφάλαιο 6: Σύνοψη

Στο κεφάλαιο αυτό ακολουθεί μια σύντομη, ολοκληρωμένη περιγραφή των εφαρμογων που αναπτύχθηκαν στο πλαίσιο της παρούσας διπλωματικής εργασίας. Στη συνέχεια, γίνεται μια σύντομη αξιολόγηση του αποτελέσματος της εργασίας και τέλος προτείνονται πιθανές μελλοντικές επεκτάσεις και βελτιώσεις της εφαρμογής MedCurie.

4.1 Αποτέλεσμα

Κύριος στόχος την διπλωματικής εργασίας ήταν η ανάπτυξη ενός συστήματος για την υποστήριξη των ασθενών που λαμβάνουν συστηματικά φαρμακευτική αγωγή, αποτελούμενη από σκευάσματα που κυκλοφορούν στην ελληνική αγορά. Οι ασθενείς αυτοί συχνά έρχονται αντιμέτωποι με το πρόβλημα της ελλιπούς φαρμακευτικής συμμόρφωσης, δηλαδή δυσκολεύονται να ακολουθήσουν τις οδηγίες του θεράποντα ιατρού τους και να λαμβάνουν το σωστό φάρμακο, στην σωστή δοσολογία και με τη σωστή συχνότητα. Υπάρχουν πολλές πιθανές αιτίες που τους ωθούν σε μια τέτοια συμπεριφορά, οι πιο σημαντικές εκ των οποίων είναι η προχωρημένη ηλικία ή οι επιπλοκές της ασθένειάς τους που οδηγούν σε έκπτωση της νοητικής λειτουργίας της μνήμης. Επιπλέον, η πολυπλοκότητα και οι συχνές αλλαγές της αγωγής τους εντείνουν την δυσκολία τους.

Στο πλαίσιο της διπλωματικής επιλέξαμε να επικεντρωθούμε στο πώς θα υποστηρίξουμε την καθημερινότητα αυτών των ανθρώπων παρέχοντάς τους έναν ηλεκτρονικό βοηθό που θα τους υπενθυμίζει τη λήψη της αγωγής τους και μέσω των φωτογραφιών και των επιπλέον πληροφοριών που παρέχονται, θα εξασφαλίζει ότι θα πάρουν το σωστό φάρμακο και στην δοσολογία που τους έχει συστήσει ο ιατρός τους. Στην πρώτη φάση ανάπτυξης λοιπόν, ήρθαμε αντιμέτωποι με την απουσία μιας ολοκληρωμένης και ενημερωμένης βάσης με φωτογραφίες για όλα τα φάρμακα που κυκλοφορούν στην Ελλάδα. Όσες πηγές καταφέραμε να βρούμε ήταν ελλιπείς ή αφορούσαν σκευάσματα που κυκλοφορούν σε ξένες αγορές, με διαφορετικά εμπορικά ονόματα και συσκευασίες. Για να ξεπεράσουμε αυτή την δυσκολία προχωρήσαμε στην ανάπτυξη μιας εφαρμογής που θα μπορούσαν να χρησιμοποιήσουν φαρμακοποιοί ή εργαζόμενοι σε κοινωνικά φαρμακεία και σταδιακά να δημιουργήσουν μια βάση δεδομένων με φωτογραφίες των συσκευασιών των φαρμάκων. Η εφαρμογή ονομάζεται MedImg_App και διευκολύνει την προαναφερθείσα διαδικασία καθώς:

- Ο φαρμακοποιός μπορεί να σκανάρει το barcode της συσκευασίας, αποφεύγοντας έτσι λάθη κατά την πληκτρολόγησή του.
- Έχει την δυνατότητα να τραβήξει την φωτογραφία μέσα από την εφαρμογή και να την επεξεργαστεί πριν την ανεβάσει, αποφεύγοντας έτσι τις επιπλέον ενέργειες που θα απαιτούσαν περισσότερο χρόνο και κόπο εάν δεν υπήρχε η εφαρμογή.
- Ο χρήστης δεν χρειάζεται να ασχοληθεί με την σωστή ονομασία του αρχείου, αφού το αναλαμβάνει η εφαρμογή ονομάζοντας αυτόματα την εικόνα ως <barcode>.png προτού την ανεβάσει στο storage.
- Τέλος, με την δυνατότητα που δίνει το Firebase για σύγχρονη και ταυτόχρονη ενημέρωση της βάσης, δεν απαιτείται επιπλέον κόπος για τον συντονισμό των φαρμακοποιών, αφού ο κάθε ένας βλέπει αν το φάρμακο που έχει επιλέξει να καταχωρήσει έχει ήδη φωτογραφία και προχωράει στο επόμενο.

Στη συνέχεια, επικεντρωθήκαμε αποκλειστικά στο βασικό πρόβλημα, που ήταν αυτό της φαρμακευτικής συμμόρφωσης του ασθενή. Για τον λόγο αυτό αναπτύξαμε την εφαρμογή MedCurie που θα μπορούσε να λειτουργήσει ως ένας βοηθός για την αυτοπαρακολούθηση του ασθενή. Τα σημαντικότερα χαρακτηριστικά της εφαρμογής είναι τα εξής:

- Συντονισμός της λήψης της αγωγής του ασθενή με ειδοποιήσεις και παροχή όλων των απαραίτητων πληροφοριών για την διευκόλυνση της διαδικασίας, όπως είναι φωτογραφία της συσκευασίας του φαρμάκου, οδηγίες για την δοσολογία και τα γεύματα που πρέπει να ακολουθήσει.
- Επιλογή για καταγραφή και παρακολούθηση ορισμένων βασικών βιομετρικών δεδομένων όπως η αρτηριακή πίεση, η γλυκόζη αίματος και η θερμοκρασία του ασθενή.
- Λειτουργία αποθήκευσης σημειώσεων από τον χρήστη σχετικών με την υγεία του.
- Δημιουργία καταλόγων με τις επαφές και τα στοιχεία των ιατρών και τον φροντιστών του για την διευκόλυνση της επικοινωνίας μαζί τους.
- Καταχώρηση των συνταγών που του έχουν γράψει οι θεράποντες ιατροί του για την ανανέωση του αποθέματος των φαρμάκων του ή για διαγνωστικές εξετάσεις. Έτσι ο χρήστης μπορεί να τις έχει συγκεντρωμένες στην εφαρμογή και να λαμβάνει ειδοποιήσεις όταν έρθει η ημερομηνία που αυτές οι συνταγές μπορούν να εκτελεστούν ή λήγουν, ώστε να μην χάνει τις προθεσμίες τους.
- Τέλος, μπορεί να καταχωρεί και να έχει συγκεντρωμένα σε ένα σημείο τα ραντεβού με τους ιατρούς του ή σε διαγνωστικά κέντρα για εξετάσεις και να λαμβάνει υπενθύμιση μια ημέρα πριν, αποτρέποντας έτσι την πιθανή ασυνέπεια σε αυτά.

4.2 Αξιολόγηση

Από όλες τις παραπάνω λειτουργίες των εφαρμογών, προκύπτουν ορισμένα συμπεράσματα, αναφορικά με τη χρησιμότητα που το σύστημα αυτό μπορεί να έχει για τους ασθενείς, καθώς και για την αξία των ιδιαίτερων χαρακτηριστικών του.

Αρχικά, το σύστημα που καταφέραμε να στήσουμε αποτελεί μια σχετικά ολοκληρωμένη πρόταση για την υποστήριξη την καθημερινότητας των ασθενών στην Ελλάδα. Μέσω της εφαρμογής MedImg_App μπορεί να αρχίσει η προσπάθεια για την ανάπτυξη μιας βάσης δεδομένων με τις φωτογραφίες των σκευασμάτων της ελληνικής αγοράς, που θα μπορούσε να φανεί χρήσιμη και σε άλλου είδους εφαρμογές ή στον ΕΟΦ.

Σε ό,τι αφορά την εφαρμογή υποστήριξης λήψης της φαρμακευτικής αγωγής MedCurie, μπορούμε να εντοπίσουμε αρκετά πλεονεκτήματα σε σχέση με παρόμοιες “ανταγωνιστικές” εφαρμογές. Έχει αναπτυχθεί για έλληνες χρήστες, αφού είναι υλοποιημένη στην ελληνική γλώσσα (οι περισσότερες αντίστοιχες εφαρμογές έχουν αναπτυχθεί στο εξωτερικό και είναι στα αγγλικά ή μπορούν να λειτουργήσουν στα ελληνικά με αυτόματη όμως μετάφραση που δυσχεραίνει την χρήση τους) και υποστηρίζεται από μια βάση δεδομένων με τις εμπορικές ονομασίες και τις φωτογραφίες που έχουν τα διάφορα σκευάσματα στην ελληνική αγορά φαρμάκων. Επίσης, το γεγονός ότι περιλαμβάνει την βάση δεδομένων, την κάνει ξεχωρίζει από εφαρμογές που αφήνουν τον χρήστη να καταχωρήσει οποιαδήποτε ονομασία φαρμάκου, χωρίς να επιβεβαιώνουν με κάποιον τρόπο την ορθότητά της και να του παρέχουν την επιπλέον πληροφορία της φωτογραφίας που τον διευκολύνει κατά την λήψη της αγωγής του.

Ένα άλλο αξιόλογο πλεονέκτημα είναι ότι ο χρήστης δημιουργεί τον δικό του λογαριασμό στην εφαρμογή και δεν χρειάζεται να καταχωρήσει ξανά όλα του τα δεδομένα σε περίπτωση που προκύψει η ανάγκη να αλλάξει συσκευή. Όλα τα δεδομένα θα εμφανιστούν αυτόματα στην νέα του συσκευή κάνοντας απλά login στον λογαριασμό του. Τέλος, το γεγονός ότι προσθέσαμε τις επιπλέον λειτουργίες όπως η καταγραφή των μετρήσεων ή οι υπενθυμίσεις για τα ραντεβού και τις συνταγές του, έχει ως αποτέλεσμα η εφαρμογή να μπορεί να υποστηρίξει ένα πιο ευρύ φάσμα προβλημάτων που έχει να διαχειριστεί ο ασθενής στην καθημερινότητά του. Εξάλλου η προσπάθεια αντιμετώπισης της ελλιπούς φαρμακευτικής συμμόρφωσης συνδέεται και με την έγκαιρη ανανέωση των αποθεμάτων των φαρμάκων του ασθενή, αλλά και με την συστηματική παρακολούθηση από τον θεράποντα ιατρό του για την επίβλεψη της φαρμακευτικής αγωγής και την τροποποίησή της όταν χρειαστεί.

Ένα χαρακτηριστικό της εφαρμογής που αφήνει πολλά περιθώρια βελτίωσης και επέκτασης είναι το γεγονός ότι δεν δίνει την δυνατότητα της επίβλεψης του ασθενή από κάποιον φροντιστή. Όπως είναι τώρα διαμορφωμένη η λογική της εφαρμογής, ο φροντιστής θα μπορούσε να την εγκαταστήσει στην δική του συσκευή, να κάνει login στον λογαριασμό του ασθενή και να παρακολουθεί την λήψη της αγωγής του ή τις μετρήσεις του. Όμως δεν στέλνονται ξεχωριστές ειδοποιήσεις στον φροντιστή όταν ο ασθενής παραλείψει ή αργήσει να πάρει μια δόση και αυτό είναι κάτι που αξίζει να επισημανθεί και να βελτιωθεί στο μέλλον.

Τέλος, αξίζει να γίνει μια πολύ σύντομη αξιολόγηση των εργαλείων που χρησιμοποιήσαμε για την ανάπτυξη των εφαρμογών μας. Χρησιμοποιώντας το Flutter SDK, η υλοποίηση του interface των εφαρμογών μας έγινε μια σχετικά εύκολη και γρήγορη διαδικασία. Επίσης, παρόλο που οι εφαρμογές έχουν αναπτυχθεί για android πλατφόρμες, μπορούν με πολύ μικρές τροποποιήσεις να λειτουργήσουν και σε iOS περιβάλλοντα. Η πλατφόρμα Firebase από την άλλη, ήταν μια εξαιρετική επιλογή για την υποστήριξη του backend των εφαρμογών μας, αφού με τα εργαλεία που παρέχει διευκόλυνε κατά πολύ την υλοποίηση και έδωσε λύσεις σε προβλήματα όπως η ανεύρεση πόρων για την υποστήριξη της βάσης δεδομένων ή η λειτουργικότητα των notifications. Η εναλλακτική, του να χρησιμοποιήσουμε local notifications μέσω του Flutter, ήταν αρκετά ασταθής. Οι ειδοποιήσεις δεν θα ήταν δυνατόν να συντονιστούν σωστά αν η εφαρμογή δεν έτρεχε στο background ή ο χρήστης δεν είχε πρόσβαση στο internet, καθιστώντας έτσι την λειτουργικότητά της αναξιόπιστη.

4.3 Προτάσεις για μελλοντικές επεκτάσεις

Παρακάτω παρουσιάζονται κάποιες προτάσεις και βελτιώσεις που μπορούν να υλοποιηθούν σε μεταγενέστερο στάδιο, με σκοπό η εφαρμογή MedCurie να γίνει πιο εύχρηστη και αποτελεσματική.

Η πρώτη πρόταση για βελτίωση είναι να προστεθούν επιπλέον ρόλοι χρηστών. Για παράδειγμα οι καταχωρήσεις του ασθενή στους ιατρούς και τους φροντιστές του, θα μπορούσαν να συνδέονται με τα αντίστοιχα προφίλ αυτών των χρηστών. Ο ρόλος του ιατρού-χρήστη θα μπορούσε να είναι να αναλαμβάνει εκείνος την καταχώρηση των οδηγιών για την φαρμακευτική αγωγή του ασθενή. Με τον τρόπο αυτό θα μπορούσε να κάνει τροποποιήσεις στην αγωγή απομακρυσμένα και να είναι βέβαιος ότι οι οδηγίες που θα καταχωρήσει θα είναι οι σωστές, ώστε ο ασθενής να επωφεληθεί στο μέγιστο από την αγωγή του. Παράλληλα, δίνοντας έναν ενεργό ρόλο στους φροντιστές ως χρήστες της εφαρμογής θα γινόταν πιο εύκολη η επίβλεψη των ασθενών τους. Θα μπορούσαν να λαμβάνουν ξεχωριστές ειδοποιήσεις όταν ο ασθενής καθυστερεί να πάρει μια δόση και να επεμβαίνουν όταν χρειαστεί για να λύσουν το όποιο θέμα μπορεί να έχει προκύψει. Τέλος, τόσο ο ιατρός όσο και ο φροντιστής θα μπορούσαν να έχουν πρόσβαση στις μετρήσεις του ασθενή και να παρακολουθούν συστηματικά την πορεία της υγείας του.

Η δεύτερη πρόταση αφορά την λειτουργία των μετρήσεων. Αρχικά θα μπορούσε να προστεθεί πλήθος άλλων κατηγοριών μετρήσεων, σημαντικών για την παρακολούθηση της εξέλιξης της υγείας του ασθενή. Τέτοια παραδείγματα θα μπορούσαν να είναι ο σφυγμός και το οξυγόνο του χρήστη, καθώς και στοιχεία σχετικά με την φυσική του κατάσταση, όπως το σωματικό του βάρος, ο δείκτης μάζας σώματος ή τα βήματα που κάνει καθημερινά. Επίσης, σε ένα επόμενο στάδιο θα μπορούσε να βελτιωθεί ο τρόπος παρουσίασης αυτών των μετρήσεων με την χρήση διαγραμμάτων.

Η τρίτη πρόταση συνδέεται και πάλι με τις μετρήσεις που προτείναμε παραπάνω και θα διευκόλυνε την καταγραφή τους, απεμπλέκοντας τον χρήστη από την διαδικασία συλλογής και καταχώρησής τους. Αυτό θα μπορούσε να συμβεί με την σύνδεση της εφαρμογής με το Smartwatch ή το Activity Tracker του χρήστη. Η σύνδεση αυτή θα μπορούσε να αφορά και την λειτουργία των ειδοποιήσεων, οι οποίες θα μπορούσαν να

προωθούνται και στο ρολόι του χρήστη, ώστε να αυξηθεί η αποτελεσματικότητά τους και στις περιπτώσεις που ο χρήστης δεν είναι κοντά στην android συσκευή του.

Τέλος, η τέταρτη πρόταση αφορά την συλλογή περισσότερων στοιχείων σχετικά με το ιστορικό του ασθενούς, που θα αποθηκεύονται στο προφίλ του. Έτσι θα μπορούσαμε να του δώσουμε την δυνατότητα να κατεβάζει σε ένα pdf το πλήρες ιστορικό του, καθώς και την αγωγή που παίρνει και τις τελευταίες του μετρήσεις (για παράδειγμα τις μετρήσεις του τελευταίου μήνα) όταν προκύψει ανάγκη, όπως στην περίπτωση που θέλει να επισκεφτεί έναν καινούργιο ιατρό και θέλει να του δώσει μια πλήρη εικόνα της υγείας του μέχρι τώρα.

Βιβλιογραφία

1. Sabaté E. Adherence To Long-Term Therapies, Evidence For Action. Switzerland: World Health Organization; 2003. https://www.who.int/chp/knowledge/publications/adherence_full_report.pdf. Accessed February 10, 2021.
2. Maresso, A., 2018. State of Health in the EU. Greece. Country Health Profile 2017. *European Journal of Public Health*, 28(suppl_4).
3. Μπισκανάκη Ε. Η Συμμόρφωση Ασθενών στη Φαρμακευτική Αγωγή: Μια Πρόκληση στις Υπηρεσίες Υγείας. Pharmacy Management και Επικοινωνία. 2021.https://www.pharmamanage.gr/. Accessed February 12, 2021.
4. Baroletti, S. and Dell'Orfano, H., 2010. Medication Adherence in Cardiovascular Disease. *Circulation*, 121(12), pp.1455-1458.
5. Kim J, Combs K, Downs J, Tillman III F. Medication Adherence: The Elephant in the Room. US Pharmacist. 2018. https://www.uspharmacist.com/article/medication-adherence-the-elephant-in-the-room. Accessed February 12, 2021.
6. Abdisa, E., Fekadu, G., Girma, S., Shibiru, T., Tilahun, T., Mohamed, H., Wakgari, A., Takele, A., Abebe, M. and Tsegaye, R., 2020. Self-stigma and medication adherence among patients with mental illness treated at Jimma University Medical Center, Southwest Ethiopia. *International Journal of Mental Health Systems*, 14(1).
7. Medication Adherence. In: CDC'S Noon Conference. Centers for Disease Control and Prevention; 2013. https://time.com/wp-content/uploads/2015/03/medication-adherence-01ccd.pdf. Accessed February 12, 2021.
8. Wimbiscus B. Patient Adherence, a Challenge of Oral Chemotherapy. Targeted Oncology. 2019. https://www.targetedonc.com/view/patient-adherence-a-challenge-of-oral-chemotherapy. Accessed February 15, 2021.
9. de Vries, S., Keers, J., Visser, R., de Zeeuw, D., Haaijer-Ruskamp, F., Voorham, J. and Denig, P., 2014. Medication beliefs, treatment complexity, and non-adherence to different drug classes in patients with type 2 diabetes. *Journal of Psychosomatic Research*, 76(2), pp.134-138.
10. The Chief Public Health Officer's Report On The State Of Public Health In Canada 2010 – Growing Older: Adding Life To Years. Government of Canada; 2014. https://www.canada.ca/en/public-health/corporate/publications/chief-public-health-officer-reports-state-public-health-canada/annual-report-on-state-public-health-canada-2010.html. Accessed February 16, 2021.
11. Proulx, J., 2018. Drug Use Among Seniors in Canada, 2016. *Value in Health*, 21, p.S146.
12. Διαταραχές Μνήμης - Neurocheck. NeuroCheck | Καψαλάκης Ιωάννης. https://neurocheck.gr/%CF%80%CE%B1%CE%B8%CE%AE%CF%83%CE%B5%CE%B9%CF%82/%CE%B4%CE%B9%CE%B1%CF%84%CE%B1%CF%81%CE%B1%CF%87%CE%AD%CF%82-%CE%BC%CE%BD%CE%AE%CE%BC%CE%B7%CF%82/. Published 2021. Accessed February 15, 2021.
13. Pharmacovigilance and Pharmacoepidemiology. Edelweiss Publications. http://edelweisspublications.com/keyword/33/1663/White-coat-adherence. Accessed February 16, 2021.
14. Benjamin E, Blaha M, Chiuve S et al. Heart Disease and Stroke Statistics—2017 Update: A Report From the American Heart Association. Circulation. 2017;135(10).
15. Perdomo Claros, M., Marín Messa, C. and García-Perdomo, H., 2019. Adherence to oral pharmacological treatment in cancer patients: Systematic review. *Oncology Reviews*, 13(1).
16. Mazzeo F, Duck L, Joosens E, et al. Nonadherence to imatinib treatment in patients with gastrointestinal stromal tumors: the ADAGIO study. Anticancer Res. 2011;31(4):1407-1409.
17. Ganesan, P., Sagar, T., Dubashi, B., Rajendranath, R., Kannan, K., Cyriac, S. and Nandennavar, M., 2011. Nonadherence to Imatinib adversely affects event free survival in chronic phase

chronic myeloid leukemia. *American Journal of Hematology*, 86(6), pp.471-474.
18. Hershman, D., Shao, T., Kushi, L., Buono, D., Tsai, W., Fehrenbacher, L., Kwan, M., Gomez, S. and Neugut, A., 2010. Early discontinuation and non-adherence to adjuvant hormonal therapy are associated with increased mortality in women with breast cancer. *Breast Cancer Research and Treatment*, 126(2), pp.529-537.
19. Πρασσά Ε. Σακχαρώδης Διαβήτης (ΣΔ) Μια παγκόσμια υγειονομική πρόκληση. MSD, Inventing for life. https://www.msd.gr/our_work/diabetes/. Accessed February 16, 2021.
20. Μελάς Ν. Το πρόβλημα της συμμόρφωσης στην αγωγή στον ΣΔ. ΦΑΡΜΑΣΕΡΒ. https://www.lilly.gr/blog/%CE%B8%CE%B5%CF%81%CE%B1%CF%80%CE%B5%CF%85%CF%84%CE%B9%CE%BA%CE%AD%CF%82-%CE%BA%CE%B1%CF%84%CE%B7%CE%B3%CE%BF%CF%81%CE%AF%CE%B5%CF%82/%CE%B4%CE%B9%CE%B1%CE%B2%CE%AE%CF%84%CE%B7%CF%82/%CF%80%CF%81%CF%8C%CE%B2%CE%BB%CE%B7%CE%BC%CE%B1-%CF%83%CF%85%CE%BC%CE%BC%CF%8C%CF%81%CF%86%CF%89%CF%83%CE%B7%CF%82-%CF%83%CF%84%CE%B7%CE%BD-%CE%B1%CE%B3%CF%89%CE%B3%CE%AE/. Published 2020. Accessed February 19, 2021.
21. Ali MK, McKeever Bullard K, Imperatore G, Barker L, Gregg EW; Centers for Disease Control and Prevention (CDC). Characteristics associated with poor glycemic control among adults with self-reported diagnosed diabetes--National Health and Nutrition Examination Survey, United States, 2007-2010. MMWR Suppl. 2012;61(2):32-37.
22. Salas, M., Hughes, D., Zuluaga, A., Vardeva, K. and Lebmeier, M., 2009. Costs of Medication Nonadherence in Patients with Diabetes Mellitus: A Systematic Review and Critical Analysis of the Literature. *Value in Health*, 12(6), pp.915-922.
23. Rodríguez-Gutiérrez R, Montori V. Glycemic Control for Patients With Type 2 Diabetes Mellitus: Our Evolving Faith in the Face of Evidence. Circulation: Cardiovascular Quality and Outcomes. 2016;9(5):504-512.
24. Gibson TB, Song X, Alemayehu B, et al. Cost sharing, adherence, and health outcomes in patients with diabetes. Am J Manag Care. 2010;16(8):589-600.
25. Τι είναι οι νευροεκφυλιστικές ασθένειες;. JPND Research. https://www.neurodegenerationresearch.eu/el/%CF%84%CE%B9-%CE%B5%CE%AF%CE%BD%CE%B1%CE%B9-%CE%BF%CE%B9-%CE%BD%CE%B5%CF%85%CF%81%CE%BF%CE%B5%CE%BA%CF%86%CF%85%CE%BB%CE%B9%CF%83%CF%84%CE%B9%CE%BA%CE%AD%CF%82-%CE%B1%CF%83%CE%B8%CE%AD%CE%BD%CE%B5/. Accessed February 21, 2021.
26. Νόσος Alzheimer. Νευροχειρουργική Νευρολογία. https://www.neurocenter.gr/alzheimer.html. Accessed February 22, 2021.
27. El-Saifi, N., Moyle, W., Jones, C. and Tuffaha, H., 2017. Medication Adherence in Older Patients With Dementia: A Systematic Literature Review. *Journal of Pharmacy Practice*, 31(3), pp.322-334.
28. Νόσος του Parkinson. Roche. https://www.roche.gr/el/health/neurology/parkinson.html. Accessed February 22, 2021.
29. Daley, D., Deane, K., Gray, R., Worth, P., Clark, A., Sabanathan, K., Pfeil, M. and Myint, P., 2011. The use of carer assisted adherence therapy for people with Parkinson's disease and their carers (CAAT-PARK): study protocol for a randomised controlled trial. *Trials*, 12(1).
30. Leoni, O., Martignoni, E., Cosentino, M., Michielotto, D., Calandrella, D., Zangaglia, R., Riboldazzi, G., Oria, C., Lecchini, S., Nappi, G. and Frigo, G., 2002. Drug prescribing patterns in Parkinson's disease: a pharmacoepidemiological survey in a cohort of ambulatory patients. *Pharmacoepidemiology and Drug Safety*, 11(2), pp.149-157.
31. Leopold, N., Polansky, M. and Hurka, M., 2004. Drug adherence in Parkinson's disease. *Movement Disorders*, 19(5), pp.513-517.
32. Grosset, K., Bone, I. and Grosset, D., 2005. Suboptimal medication adherence in Parkinson's disease. *Movement Disorders*, 20(11), pp.1502-1507.
33. Grosset, K., Reid, J. and Grosset, D., 2005. Medicine-taking behavior: Implications of suboptimal compliance in Parkinson's disease. *Movement Disorders*, 20(11), pp.1397-1404.
34. Grosset, D., Taurah, L., Burn, D., MacMahon, D., Forbes, A., Turner, K., Bowron, A., Walker,

R., Findley, L., Foster, O., Patel, K., Clough, C., Castleton, B., Smith, S., Carey, G., Murphy, T., Hill, J., Brechany, U., McGee, P., Reading, S., Brand, G., Kelly, L., Breen, K., Ford, S., Baker, M., Williams, A., Hearne, J., Qizilbash, N. and Chaudhuri, K., 2006. A multicentre longitudinal observational study of changes in self reported health status in people with Parkinson's disease left untreated at diagnosis. *Journal of Neurology, Neurosurgery & Psychiatry*, 78(5), pp.465-469.
35. O'Sullivan, S., Evans, A. and Lees, A., 2009. Dopamine Dysregulation Syndrome. *CNS Drugs*, 23(2), pp.157-170.
36. Orcioni S, Pellegrini R, Seepold R, Gaiduk M, Madrid N, Conti M. Medication adherence supported by mHealth and NFC. Inform Med Unlocked. 2021;23:100552.
37. Sokol M, McGuigan K, Verbrugge R, Epstein R. Impact of Medication Adherence on Hospitalization Risk and Healthcare Cost. Med Care. 2005;43(6):521-530.
38. Vermeire E, Avonts D, Van Royen P, Buntinx F, Denekens J. Context and health outcomes. The Lancet. 2001;357(9273):2059-2060.
39. Benjamin R. Medication Adherence: Helping Patients Take Their Medicines as Directed. Public Health Rep. 2012;127(1):2-3
40. McGuire M, Iuga. Adherence and health care costs. Risk Manag Healthc Policy. 2014:35
41. Πέτρου Μ. Μελέτη και καταγραφή των μεθόδων ιατρικής συμμόρφωσης και στατιστική ανάλυση του επιπέδου συμμόρφωσης των ασθενών μέσω mobile εφαρμογών. 2014.
42. MediSafe (2021) [Online]. Available: https://www.medisafe.com/
43. Med helper. Your healthcare assistant (2021) [Online]. Available: http://medhelperapp.com/
44. Pill reminder (2021) [Online]. Available: https://play.google.com/store/apps/details?id=it.andreacanevari.android.pillreminder
45. Pillmanager (2021) [Online]. Available: http://www.pillmanager.co.uk/
46. Dosecast (2021) [Online]. Available: http://www.montunosoftware.com/about/
47. Care4today (2021) [Online]. Available: https://www.care4today.com/
48. What is serverless computing? | Serverless definition. Cloudflare. https://www.cloudflare.com/learning/serverless/what-is-serverless/. Accessed February 25, 2021.
49. New Frontend Technology Revolution Triggered by Serverless. Alibabacloud. https://www.alibabacloud.com/help/doc-detail/194806.htm. Published 2021. Accessed April 17, 2021.
50. Flutter architectural overview. Flutter. https://flutter.dev/docs/resources/architectural-overview. Accessed April 23, 2021.
51. Flutter. Tutorials Point; 2019. https://www.tutorialspoint.com/flutter/flutter_tutorial.pdf. Accessed April 27, 2021.
52. Γαληνός. https://www.galinos.gr/. Accessed November 26, 2020.
53. Moira A. NTUA_Thesis_1. GitHub. https://github.com/AntigoniMoira/NTUA_Thesis_1. Published 2021. Accessed April 23, 2021.
54. Moira A. NTUA_Thesis_2. GitHub. https://github.com/AntigoniMoira/NTUA_Thesis_2. Published 2021. Accessed July 8, 2021.