ΔΙΕΘΝΕΣ ΠΑΝΕΠΙΣΤΗΜΙΟ ΤΗΣ ΕΛΛΑΔΟΣ

ΣΧΟΛΗ ΜΗΧΑΝΙΚΩΝ

ΤΜΗΜΑ ΜΗΧΑΝΙΚΩΝ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΗΛΕΚΤΡΟΝΙΚΩΝ ΣΥΣΤΗΜΑΤΩΝ

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

«ΕΦΑΡΜΟΓΗ ΑΝΑΖΗΤΗΣΗΣ ΙΑΤΡΟΥ ΣΕ ΕΞΥΠΝΟ ΚΙΝΗΤΟ»

Του φοιτητή
ΠΕΡΤΣΕΛΗ ΝΙΚΗΦΟΡΟΥ
Αρ. Μητρώου: 133991

Επιβλέπων
ΣΙΔΗΡΟΠΟΥΛΟΣ ΑΝΤΩΝΙΟΣ
Επίκουρος Καθηγητής

Ημερομηνία 07-01-2022

Τίτλος Δ.Ε.ΕΦΑΡΜΟΓΗ ΑΝΑΖΗΤΗΣΗΣ ΙΑΤΡΟΥ ΣΕ ΕΞΥΠΝΟ ΚΙΝΗΤΟ
Κωδικός Δ.Ε. 20158
Ονοματεπώνυμο φοιτητή ΝΙΚΗΦΟΡΟΣ ΠΕΡΤΣΕΛΗΣ
Ονοματεπώνυμο εισηγητή ΑΝΤΩΝΙΟΣ ΣΙΔΗΡΟΠΟΥΛΟΣ
Ημερομηνία ανάληψης Δ.Ε. 12-04-2020
Ημερομηνία περάτωσης Δ.Ε. 07-01-2022

Βεβαιώνω ότι είμαι ο συγγραφέας αυτής της εργασίας και ότι κάθε βοήθεια την οποία είχα για την προετοιμασία της είναι πλήρως αναγνωρισμένη και αναφέρεται στην εργασία. Επίσης, έχω καταγράψει τις όποιες πηγές από τις οποίες έκανα χρήση δεδομένων, ιδεών, εικόνων και κειμένου, είτε αυτές αναφέρονται ακριβώς είτε παραφρασμένες. Επιπλέον, βεβαιώνω ότι αυτή η εργασία προετοιμάστηκε από εμένα προσωπικά, ειδικά ως διπλωματική εργασία, στο Τμήμα Μηχανικών Πληροφορικής και Ηλεκτρονικών Συστημάτων του ΔΙ.ΠΑ.Ε.

Η έγκριση της διπλωματικής εργασίας από το Τμήμα Μηχανικών Πληροφορικής και Ηλεκτρονικών Συστημάτων του Διεθνούς Πανεπιστημίου της Ελλάδος, δεν υποδηλώνει απαραιτήτως και αποδοχή των απόψεων του συγγραφέα, εκ μέρους του Τμήματος.

Στο τέλος μίας όμορφης συνεργασίας

Πρόλογος

Η ραγδαία ανάπτυξη της τεχνολογίας στο τομέα της πληροφορικής, έχει οδηγήσει στη βελτίωση των κινητών τηλεφώνων, μετατρέποντάς τα σε έξυπνα κινητά. Η χρήση τους δεν αφορά μόνο την επικοινωνία των ανθρώπων αλλά αποτελούν εργαλείο για τη διευκόλυνση της καθημερινότητας των χρηστών. Στις μέρες μας όλο και περισσότεροι επαγγελματίες του τομέα της υγείας χρησιμοποιούν τα έξυπνα κινητά και τις σχετικές εφαρμογές για να βοηθήσουν τους υγειονομικούς σε πολλές σημαντικές εργασίες τους. Οι νέες εφαρμογές έχουν τεράστιες δυνατότητες στην υγειονομική περίθαλψη των ασθενών, βελτιώνοντας τη ποιότητα των παρεχόμενων υπηρεσιών στους ασθενής, μειώνοντας, ταυτόχρονα, το χρόνο και το κόστος της περίθαλψης. Επιπλέον, η χρήση ιατρικών εφαρμογών ενισχύει και διευκολύνει την επικοινωνία μεταξύ ιατρού και ασθενή, προωθώντας την ενημέρωση των χρηστών για θέματα υγείας.

Περίληψη

Αντικείμενο της παρούσας εργασίας αποτέλεσε ο σχεδιασμός, η ανάπτυξη και η υλοποίηση μίας εφαρμογής αναφορικά με την αναζήτηση ιατρών. Η συγκεκριμένη εφαρμογή ορίζει ως στόχους τη βελτίωση του τρόπου αναζήτησης ιατρών και τη διευκόλυνση της διαδικασίας καθορισμού των ραντεβού. Ταυτόχρονα, δίνει τη δυνατότητα στους χρήστες της να έχουν πρόσβαση και να μοιράζονται μεταξύ τους πληροφορίες για την υγεία και την ιατρική σε ψηφιακή μορφή. Πιο συγκεκριμένα, η εφαρμογή, αποτελείται από δύο μέρη, την χρήση από τη μεριά του ιατρού και από τη μεριά του ασθενή. Οι δύο μεριές έχουν κοινές αλλά και διαφορετικές δυνατότητες και λειτουργίες. Η εφαρμογή χαρακτηρίζεται ως πολλαπλών πλατφορμών (cross platform), διότι υποστηρίζει ταυτόχρονα τα λογισμικά Android και iOS. Για την ανάπτυξη της εφαρμογής χρησιμοποιήθηκαν τεχνολογίες της Google, όπως το Flutter Framework και το Firebase. Από τη πλατφόρμα Firebase χρησιμοποιήθηκε το Firestore, μία ευέλικτη NoSQL cloud βάση δεδομένων, για την αποθήκευση και το συγχρονισμό δεδομένων. Το Flutter Framework χρησιμοποιήθηκε για την ανάπτυξη της εφαρμογής, αξιοποιώντας τη γλώσσα προγραμματισμού Dart για τη συγγραφή του πηγαίου κώδικα. Η εφαρμογή είναι εφοδιασμένη με τα κατάλληλα εργαλεία προκειμένου οι ιατροί να ορίζουν και να διαχειρίζονται το διαθέσιμο πρόγραμμα επισκέψεων και από την άλλη οι ασθενείς να αναζητούν, να κλείνουν ραντεβού και τελικώς να αξιολογούν τους ιατρούς. Παράλληλα δίνεται η δυνατότητα για όλους τους χρήστες να μοιράζονται πληροφορίες και εμπειρίες σε ένα κοινό ψηφιακό τόπο συζητήσεως. Στη παρούσα εργασία, αρχικά περιγράφονται και αναλύονται οι απαιτήσεις του συστήματος. Στη συνέχεια, αναλύονται όλες οι τεχνολογίες που λαμβάνουν μέρος στην εφαρμογή. Έπειτα παρουσιάζεται η μελέτη της υλοποίησης της εφαρμογής, εμβαθύνοντας στο τρόπο που χρησιμοποιήθηκαν και αναπτύχθηκαν οι τεχνολογίες. Τέλος, περιγράφεται η εφαρμογή, παρουσιάζοντας με εικόνες και επεξηγήσεις τη λειτουργικότητά της.

DOCTOR SEARCH APPLICATION ON A SMART MOBILE PHONE

NIKIFOROS PERTSELIS

Abstract

The subject of this final year thesis project was the design, the development and the implementation of a doctor search mobile application. This application aims to improve the search process and the doctor's appointment scheduling. Simultaneously, the users are able to access and share health and medical information in digital form. This application consists of two parts, the utilization of the app by doctors and patients. Regarding to these parts, similarities but also a few differences in the way that doctors and patients operate on this application are noticed. This mobile application is defined as cross platform software since it can work across multiple types of platforms such as Android and iOS operating systems. Google technologies such as Flutter Framework and Firebase were used for the development of this project. The storage and synchronization of the data were based on the Firebase Firestore, a flexible NoSQL cloud database. Additionally, the Flutter Framework was used in order to fulfill the application. In particular, the source code was written based on the Dart programming language. The application is equipped with the appropriate tools for the doctors to set the available schedule of appointments and for the patients to search, make appointments and finally evaluate the doctors. Moreover, all users are able to share information and experiences in an online discussion forum. In the first chapter of this dissertation, the system requirements are described and analyzed. Furthermore, all the technologies which are used for the implementation of this project are presented thoroughly. In the third chapter, the execution phase of this project is elaborated emphasizing on the way that aforementioned technologies were used and developed. Last but not least, this application is illustrated with images and clarifications on its functionality.

Ευχαριστίες

Με την περάτωση της παρούσας πτυχιακής εργασίας θα ήθελα να εκφράσω τις ειλικρινείς και θερμές ευχαριστίες μου σε όσους συνέβαλαν στην ολοκλήρωση αυτής της προσπάθειας. Θα ήθελα αρχικά να ευχαριστήσω τον επιβλέποντα καθηγητή, κύριο Αντώνιο Σιδηρόπουλο, για την ανάθεση αυτής της πτυχιακής εργασίας αλλά και για την πολύτιμη καθοδήγησή του και το εξαιρετικό κλίμα συνεργασίας που διαμόρφωσε. Τέλος, θα ήθελα να ευχαριστήσω την οικογένειά μου, καθώς και σε όλους εκείνους που στάθηκαν κοντά μου, για τη συμπαράσταση και την υπομονή τους.

Περιεχόμενα

Κατάλογος Σχημάτων

Κατάλογος Εικόνων

Κατάλογος Πινάκων

Κατάλογος Κώδικα

Συντομογραφίες

Δ.Ε.	Διπλωματική Εργασία
SDK	Software Development Kit

Κεφάλαιο 1ο: Ανάλυση απαιτήσεων του λογισμικού

1.1 Θεμελιώδης σκοπός της εφαρμογής

Θεμελιώδης σκοπός της εφαρμογής είναι να προσφέρει στο τελικό χρήστη τη βέλτιστη λειτουργικότητα και απόδοση. Ταυτόχρονα, για να εμπιστεύονται οι χρήστες την εφαρμογή, είναι αναγκαίο να προσφέρει αξιοπιστία. Η εφαρμογή είναι απαραίτητο να είναι εύκολη στη συντήρηση ώστε να μπορεί να εξελίσσεται και να καλύπτει καινούριες ανάγκες που μπορούν να προκύψουν από τους χρήστες. Τέλος, οι πόροι του συστήματος θα πρέπει να αξιοποιούνται με τον καλύτερο δυνατό τρόπο, έτσι ώστε να παρέχεται η μέγιστη αποδοτικότητα, η οποία θα οδηγήσει στην ικανοποίηση του τελικού χρήστη. Για να επιτευχθούν τα παραπάνω, αρχικά πραγματοποιείται η ανάλυση των απαιτήσεων του συστήματος.

1.2 Απαιτήσεις λογισμικού μίας εφαρμογής

Μία απαίτηση είναι μία πρόταση η οποία περιγράφει είτε μία πτυχή του τι πρέπει να κάνει το σύστημα ή έναν περιορισμό στην ανάπτυξη του συστήματος [1]. Πρακτικά αυτό σημαίνει ότι οι απαιτήσεις λογισμικού μίας εφαρμογής αποτελούν τη διαδικασία της περιγραφής και της ανάλυσης όλων των απαραίτητων ενεργειών που χρειάζεται να εκτελέσει το σύστημα. Προκειμένου να ικανοποιηθεί ο τελικός χρήστης, εκτός από την ανάλυση των απαιτήσεων είναι απαραίτητη και η περιγραφή όλων των πιθανών περιορισμών, βάσει των οποίων αναπτύσσεται και λειτουργεί το ίδιο το σύστημα.

Το σύστημα είναι απαραίτητο να συμβάλλει στην ικανοποιητική αντιμετώπιση των προβλημάτων του χρήστη. Επομένως, μία απαίτηση μπορεί να υπάρχει μόνο όταν μπορεί να επιλύσει, σε μεγάλο βαθμό, τα προβλήματα και τις ανάγκες του τελικού χρήστη. Για τον λόγο αυτό, η ανάλυση των απαιτήσεων αποτελεί ένα πολύ σημαντικό παράγοντα στην υλοποίηση του έργου, διότι κάθε απαίτηση αποτελεί τη βάση της διαπραγμάτευσης μεταξύ όλων των εμπλεκομένων πλευρών που ασχολούνται με την υλοποίηση του έργου και για αυτό το λόγο θα πρέπει να είναι μία ακριβής πληροφορία, η οποία καλύπτει όλες τις περιπτώσεις και δεν αντιβαίνει με άλλες απαιτήσεις [1].

1.3 Οι κατηγορίες των απαιτήσεων

Τα είδη των απαιτήσεων διαχωρίζονται σε λειτουργικές απαιτήσεις και μη λειτουργικές απαιτήσεις.

Οι λειτουργικές απαιτήσεις αποτελούνται από τις απαιτήσεις οι οποίες καθορίζουν τις υπηρεσίες που παρέχει το σύστημα και περιγράφουν τον τρόπο συμπεριφοράς του συστήματος κατά την αλληλεπίδρασή του με συγκεκριμένες πληροφορίες και καταστάσεις.

Οι μη λειτουργικές απαιτήσεις περιγράφουν ιδιότητες του συστήματος με βάση τις οποίες κρίνεται η σωστή υποστήριξη των λειτουργικών απαιτήσεων από το σύστημα. Τέτοιες ιδιότητες είναι η ασφάλεια, η απόδοση, η αποτελεσματικότητα, η αξιοπιστία και η επαναχρησιμοποίηση του συστήματος. Οι απαιτήσεις αυτές, περιορίζουν τους τρόπους με τους οποίους μπορούν να ολοκληρωθούν οι λειτουργικές απαιτήσεις και για το λόγο αυτό θεωρούνται περιορισμοί του συστήματος. [2]

1.3.1 Λειτουργικές απαιτήσεις της εφαρμογής

Η παρούσα εφαρμογή υλοποιείται ταυτόχρονα σε δύο πλατφόρμες (Android, iOS) χρησιμοποιώντας το Flutter Framework, το οποίο αναλύεται στη συνέχεια. Σκοπός της εφαρμογής είναι να πληροί όλες τις απαραίτητες προϋποθέσεις και τις ανάγκες που πρέπει να ικανοποιεί μία εφαρμογή αναζήτησης ιατρού. Οι λειτουργικές απαιτήσεις της παρούσας εφαρμογής μπορούν να αναλυθούν ως εξής:

1) Διαπίστευση του χρήστη (user authentication)
   Η διαδικασία της διαπίστωσης του χρήστη είναι η πρώτη λειτουργία που πραγματοποιείται για κάθε εγγραφή στο σύστημα. Αποτελεί μία από τις βασικότερες λειτουργίες του συστήματος διότι εμποδίζει τους μη εξουσιοδοτημένους χρήστες να αποκτήσουν πρόσβαση σε ευαίσθητες πληροφορίες. Για τον λόγο αυτό, οι περισσότερες εφαρμογές πρέπει να γνωρίζουν την ταυτότητα του κάθε χρήστη. Η γνώση της ταυτότητας των χρηστών, επιτρέπει σε μια εφαρμογή να αποθηκεύει με ασφάλεια τα δεδομένα τους στη βάση δεδομένων και να τα προστατεύει από επιθέσεις. Για να συνδεθεί ένας χρήστης στην εφαρμογή, θα πρέπει η εφαρμογή να λάβει τα απαραίτητα διαπιστευτήρια. Τα διαπιστευτήρια είναι η διεύθυνση email και ο κωδικός πρόσβασης που έχει δηλώσει ο χρήστης κατά την εγγραφή του, τα οποία μεταβιβάζονται στο Firebase Authentication SDK και εφόσον επαληθευτούν, ο χρήστης συνδέεται επιτυχώς στην εφαρμογή.

2) Δικαιοδοσία χρηστών (user authorization)
   Η δικαιοδοσία των χρηστών μεριμνά, επίσης, για τη ταυτότητα του χρήστη, αλλά χρησιμοποιείται για τον καθορισμό των λειτουργιών, των ενεργειών και των δεδομένων στα οποία παρέχεται πρόσβαση στον χρήστη. Στη παρούσα εφαρμογή, η δικαιοδοσία των χρηστών καθορίζεται με την καταγραφή κανόνων ασφαλείας του Firebase.

3) Προβολή των ιατρών ως λίστα
   Από τη στιγμή που ο εγγεγραμμένος χρήστης συνδεθεί επιτυχώς, δίνεται η δυνατότητα της προβολής όλων των ιατρών που είναι εγγεγραμμένοι στο σύστημα. Ο χρήστης πλοηγείται στην οθόνη αναζήτησης γιατρού και εκεί του προβάλλεται μια λίστα από όλους τους διαθέσιμους γιατρούς. Η εφαρμογή δίνει τη δυνατότητα στον χρήστη να ταξινομήσει τους γιατρούς με βάση την βαθμολογία ή τον συνολικό αριθμό αξιολογήσεων που έχει αποκτήσει ο κάθε γιατρός από τις αξιολογήσεις άλλων χρηστών. Ο χρήστης μπορεί να αναζητήσει έναν ιατρό με τέσσερις διαφορετικούς τρόπους.
   Αρχικά, ο χρήστης μπορεί να επιλέξει τον ιατρό που επιθυμεί από την προβαλλόμενη λίστα. Επιλέγοντας ένα ιατρό, πλοηγείται στην οθόνη του ιατρού και του παρέχονται όλες οι διαθέσιμες πληροφορίες του επιλεγμένου γιατρού.
   Ο δεύτερος τρόπος για να αναζητήσει έναν ιατρό είναι με αναζήτηση κειμένου. Η αναζήτηση πραγματοποιείται με βάση το επώνυμο του ιατρού και η εφαρμογή εμφανίζει μία αναπτυσσόμενη λίστα με τα αποτελέσματα της αναζήτησης. Ο χρήστης μπορεί να επιλέξει έναν ιατρό από αυτή τη λίστα και να πλοηγηθεί στην οθόνη του αντίστοιχου γιατρού.

   Ο τρίτος τρόπος αναζήτησης ιατρού είναι με βάση την ειδικότητά του. Ο χρήστης, αφού επιλέξει την ειδικότητα για την οποία ενδιαφέρεται, μεταφέρεται σε νέα οθόνη όπου εμφανίζεται μία λίστα με τους ιατρούς της συγκεκριμένης ειδικότητας.

   Ο τέταρτος και τελευταίος τρόπος αναζήτησης ιατρού είναι μέσω ενός γεωγραφικού χάρτη. Η εφαρμογή, αρχικά, ζητάει άδεια της τοποθεσίας του χρήστη. Εάν ο χρήστης παραχωρήσει την άδεια της τοποθεσίας του, τότε ο χάρτης εμφανίζει όλους τους ιατρούς που βρίσκονται κοντά του. Διαφορετικά ο χρήστης μπορεί να πλοηγηθεί στον χάρτη και να αναζητήσει γιατρούς στη περιοχή που επιθυμεί. Επιλέγοντας μία πινέζα στον χάρτη, εμφανίζεται ο αντίστοιχος ιατρός, όπου πατώντας επάνω, ο χρήστης πλοηγείται στην οθόνη του αντίστοιχου ιατρού.

4) Δημιουργία ραντεβού
   Ο εγγεγραμμένος χρήστης, έχει τη δυνατότητα να κλείσει ραντεβού στον ιατρό που επιθυμεί. Ο χρήστης αφού περιηγηθεί στην οθόνη του γιατρού, μπορεί να επιλέξει την ημερομηνία και την ημέρα που επιθυμεί, μέσα από το ημερολόγιο των διαθέσιμων ωρών. Τα στοιχεία που απαιτούνται για την ολοκλήρωση του ραντεβού είναι το ονοματεπώνυμο του χρήστη, το τηλέφωνό του και το email του.

5) Προβολή των ραντεβού ως λίστα
   Ο εγγεγραμμένος χρήστης με την ιδιότητα του ασθενή, εφόσον έχει κλείσει ραντεβού σε κάποιον ιατρό, μπορεί να δει τη λίστα με τα προσεχή ραντεβού του. Το σύστημα του δίνει τη δυνατότητα να ταξινομήσει τα ραντεβού του με βάση την ημερομηνία. Στη περίπτωση του χρήστη με την ιδιότητα του ιατρού, δίνεται η δυνατότητα προβολής των ραντεβού που αφορούν τον ίδιο. Τα ραντεβού μίας συγκεκριμένης ημέρας, προβάλλονται σε λίστα, ταξινομημένα με βάση την ώρα.

6) Προσθήκη διαθέσιμων ημερομηνιών για ραντεβού
   Ο εγγεγραμμένος χρήστης με την ιδιότητα του ιατρού, μπορεί να ενημερώσει τις ημέρες και τις ώρες κατά τις οποίες δέχεται ραντεβού. Τα δεδομένα ενημερώνονται και επεξεργάζονται από τον γιατρό, ενώ δίνεται η δυνατότητα να προσθέσει ή να αφαιρέσει κάποια συγκεκριμένη ώρα από τις ήδη υπάρχουσες.

7) Προβολή αγαπημένων γιατρών
   Ο χρήστης ο οποίος έχει συνδεθεί ως ασθενής, μπορεί να προσθέσει στα αγαπημένα του, όποιον γιατρό επιθυμεί. Κάθε φορά που προσθέτει έναν γιατρό στα αγαπημένα, δημιουργείται μία λίστα η οποία είναι προσβάσιμη στον χρήστη, έτσι ώστε να έχει συγκεντρωμένους τους γιατρούς που επιθυμεί.

8) Επεξεργασία δεδομένων του χρήστη
   Σε αυτή τη λειτουργία ο χρήστης ως ασθενής μπορεί να επεξεργαστεί διαφορετικά στοιχεία από τον χρήστη με την ιδιότητα του ιατρού. Στη περίπτωση που ο χρήστης είναι ασθενής, του παρέχεται η δυνατότητα να επεξεργαστεί την εικόνα, το ύψος και το βάρος του. Στην άλλη περίπτωση, ο χρήστης ιατρός μπορεί να τροποποιήσει την εικόνα του, την περιγραφή του και το ωράριο λειτουργίας του. Το email του χρήστη δεν μπορεί να τροποποιηθεί διότι χρησιμοποιείται για την διαπίστευση του χρήστη.

9) Δημοσίευση και σχολιασμός
   Αυτή η λειτουργία δίνει τη δυνατότητα σε κάθε εγγεγραμμένο χρήστη, να μπορεί να δημοσιεύει τις σκέψεις και τα ερωτήματά του σχετικά με ιατρικά θέματα, καθώς, επίσης, δύναται να σχολιάσει δημοσιεύσεις, ανταλλάσσοντας πληροφορίες και ιδέες με άλλους χρήστες.

1.3.2 Μη λειτουργικές απαιτήσεις της εφαρμογής

Παρακάτω αναφέρονται και αναλύονται οι μη λειτουργικές απαιτήσεις της εφαρμογής:

1) Ασφάλεια
   Η ασφάλεια ενός λογισμικού αποτελεί έναν από τους βασικότερους παράγοντες. Η εφαρμογή περιέχει προσωπικά δεδομένα των χρηστών και για το λόγο αυτό χρειάζεται να διασφαλιστεί η ασφάλεια και η ακεραιότητα των δεδομένων. Ένας χρήστης για να έχει πρόσβαση στη παρούσα εφαρμογή, θα πρέπει να συνδεθεί με το προσωπικό του email και τον κωδικό πρόσβασης. Για να μπορεί ένας χρήστης να συνδεθεί στην εφαρμογή είναι αναγκαία η εγγραφή του στο σύστημα, να γνωρίζει τα προσωπικά του διαπιστευτήρια και να μην είναι αποκλεισμένος από το σύστημα. Με τον τρόπο αυτό διασφαλίζεται η διαπίστευση του χρήστη. Παράλληλα, έχουν αναπτυχθεί κανόνες ασφαλείας με τη χρήση του Firestore Rules, έτσι ώστε ένας μη εξουσιοδοτημένος χρήστης να μη έχει τη δυνατότητα να διαβάσει ή να γράψει στη βάση δεδομένων.

2) Απόδοση και αποτελεσματικότητα
Η απόδοση και η αποτελεσματικότητα διαδραματίζουν σημαντικό ρόλο στους πελάτες διότι η απόδοση της εφαρμογής επηρεάζει τον χρόνο της κεντρικής μονάδας ελέγχου, τη μνήμη, τον χώρο αποθήκευσης που καταλαμβάνει η εφαρμογή καθώς και άλλους πόρους που επηρεάζουν το τελικό κόστος της εφαρμογής. Το λογισμικό που θα αναπτυχθεί θα πρέπει να είναι απλό στη χρήση και γρήγορο στην πλοήγηση και τη προβολή των δεδομένων.

3) Αξιοπιστία
Το πλήθος και ο τύπος των σφαλμάτων ενός λογισμικού καθορίζουν την αξιοπιστία του. Ένα λογισμικό είναι πιο αξιόπιστο, όταν παρουσιάζει όσο το δυνατόν λιγότερες αποτυχίες. Η αξιοπιστία εξασφαλίζεται όταν το λογισμικό είναι υλοποιήσιμο και τροποποιήσιμο με τέτοιο τρόπο, ώστε να μπορεί να διαχειριστεί τυχόν σφάλματα που θα προκύψουν. Στην τρέχουσα εφαρμογή, τα δεδομένα είναι συνεχώς διαθέσιμα προς τους χρήστες, ακόμα και όταν δεν υπάρχει πρόσβαση στο διαδίκτυο.

4) Επαναχρησιμοποίηση
Η επαναχρησιμοποίηση του λογισμικού σχετίζεται με τη δυνατότητα χρήσης του ίδιου λογισμικού σε διαφορετικές πλατφόρμες. Η παρούσα εφαρμογή είναι διαθέσιμη για έξυπνα κινητά και ταμπλέτες, σε λειτουργικό Android και iOS ταυτόχρονα, χρησιμοποιώντας τον ίδιο πηγαίο κώδικα.

1.4 Περιπτώσεις χρήσης

Πίνακας 1.1 Περίπτωση χρήσης: Εγγραφή του ανώνυμου χρήστη στο σύστημα

Περίπτωση χρήσης	**Εγγραφή του ανώνυμου χρήστη στο σύστημα**
Χρήστης	Ανώνυμος χρήστης
Προϋποθέσεις	1) Σύνδεση στο διαδίκτυο
Κύρια ροή γεγονότων	1) Ο ανώνυμος χρήστης επιλέγει «Εγγραφή» από την οθόνη σύνδεσης της εφαρμογής 2) Ο χρήστης επιλέγει από τις δύο καρτέλες, εάν πρόκειται για ασθενή ή γιατρό 3) Το σύστημα εμφανίζει μία φόρμα για την εισαγωγή των απαραίτητων στοιχείων 4) Ο ανώνυμος χρήστης πατάει το κουμπί «Συνέχεια» 5) Το σύστημα δημιουργεί και αποθηκεύει τον λογαριασμό του χρήστη 6) Το σύστημα ανακατευθύνει τον χρήστη στην αρχική οθόνη
Εναλλακτικές ροές γεγονότων	1) Στο βήμα 4, το σύστημα διαπιστώνει ότι ο χρήστης έχει εισάγει email που υπάρχει ήδη. a) Το σύστημα ενημερώνει τον χρήστη με μήνυμα «Το email που επιλέξατε χρησιμοποιείται ήδη» b) Ο χρήστης εισάγει νέο email c) Επαναφορά στο βήμα 3 2) Στο βήμα 4, το σύστημα διαπιστώνει ότι ο χρήστης έχει εισάγει λάθος συνθηματικά a) Το σύστημα ενημερώνει τον χρήστη με μήνυμα «Οι κωδικοί δεν ταιριάζουν» b) Ο χρήστης εισάγει ξανά τα συνθηματικά c) Επαναφορά στο βήμα 3 3) Στο βήμα 4, το σύστημα διαπιστώνει ότι ο χρήστης έχει εισάγει κωδικό πρόσβασης ο οποίος περιλαμβάνει λιγότερο από 6 χαρακτήρες. a) Το σύστημα ενημερώνει τον χρήστη με μήνυμα «Ο κωδικός πρόσβασης πρέπει να περιέχει τουλάχιστον 6 χαρακτήρες» b) Ο χρήστης εισάγει ξανά τον κωδικό πρόσβασης c) Επαναφορά στο βήμα 3

	4) Στο βήμα 4, το σύστημα διαπιστώνει ότι το κάποιο πεδίο είναι κενό ή λάθος. a) Το σύστημα ενημερώνει τον χρήστη με μήνυμα ότι το αντίστοιχο πεδίο χρειάζεται να συμπληρωθεί. b) Ο χρήστης συμπληρώνει το πεδίο c) Το σύστημα επαληθεύει την ορθότητα των στοιχείων d) Επαναφορά στο βήμα 3 5) Στο βήμα 4, δεν υπάρχει σύνδεση στο διαδίκτυο. a) Το σύστημα αποτυγχάνει να εισάγει τον χρήστη στη βάση δεδομένων b) Η συσκευή αποκτάει πρόσβαση στο διαδίκτυο c) Επαναφορά στο βήμα 3
Μετά-συνθήκες	Ο ανώνυμος χρήστης έχει εγγραφεί στο σύστημα

Πίνακας 1.2 Περίπτωση χρήσης: Σύνδεση εγγεγραμμένου χρήστη στο σύστημα

Περίπτωση χρήσης	**Σύνδεση εγγεγραμμένου χρήστη στο σύστημα**
Χρήστης	Εγγεγραμμένος χρήστης
Προϋποθέσεις	1) Π.Χ. 01 Εγγραφή χρήστη 2) Σύνδεση στο διαδίκτυο
Κύρια ροή γεγονότων	1) Ο εγγεγραμμένος χρήστης πλοηγείται στην οθόνη σύνδεσης της εφαρμογής 2) Το σύστημα εμφανίζει φόρμα εισαγωγής στοιχείων 3) Ο εγγεγραμμένος χρήστης συμπληρώνει τα διαπιστευτήριά του (email, κωδικός πρόσβασης) 4) Ο εγγεγραμμένος χρήστης πατάει το κουμπί «Σύνδεση» 5) Το σύστημα επιβεβαιώνει τα διαπιστευτήρια του εγγεγραμμένου χρήστη 6) Το σύστημα ανακατευθύνει τον χρήστη στην αρχική οθόνη της εφαρμογής
Εναλλακτικές ροές γεγονότων	1) Στο βήμα 5, το σύστημα διαπιστώνει ότι το email ή/και ο κωδικός πρόσβασης είναι λάθος. a) Το σύστημα ενημερώνει τον χρήστη με μήνυμα «Το email ή ο κωδικός πρόσβασης που εισάγατε είναι λάθος» b) Επαναφορά στο βήμα 3 2) Στο βήμα 4, δεν υπάρχει σύνδεση στο διαδίκτυο. a) Το σύστημα αποτυγχάνει να εισάγει τον χρήστη στη βάση δεδομένων b) Η συσκευή αποκτάει πρόσβαση στο διαδίκτυο c) Επαναφορά στο βήμα 3
Μετά-συνθήκες	Ο εγγεγραμμένος χρήστης έχει συνδεθεί στο λογαριασμό του και έχει ανακατευθυνθεί στην αρχική οθόνη της εφαρμογής

Πίνακας 1.3 Περίπτωση χρήσης: Υποβολή μίας δημοσίευσης

Περίπτωση χρήσης	**Υποβολή μίας δημοσίευσης**
Χρήστης	Εγγεγραμμένος χρήστης
Προϋποθέσεις	1) Π.Χ. 02 Σύνδεση εγγεγραμμένου χρήστη στο σύστημα
Κύρια ροή γεγονότων	1) Ο εγγεγραμμένος χρήστης πλοηγείται στην αρχική οθόνη της εφαρμογής. 2) Το σύστημα εμφανίζει γραμμή πλοήγησης με επιλεγμένη την αρχική οθόνη 3) Ο εγγεγραμμένος χρήστης πληκτρολογεί στο πεδίο δημοσίευσης 4) Ο εγγεγραμμένος χρήστης εάν επιθυμεί, πατάει στο κουμπί για να επιλέξει μία φωτογραφία από τα αρχεία της συσκευής του και να τη

	δημοσιεύσει. 5) Το σύστημα ζητάει από τον χρήστη να του επιτρέπει την πρόσβαση στα αρχεία της συσκευής (μέχρι ο χρήστης παραχωρήσει τη πρόσβαση στα αρχεία της συσκευής) 6) Ο χρήστης επιτρέπει στη συσκευή να έχει πρόσβαση στα αρχεία της συσκευής 7) Το σύστημα ανοίγει την πλοήγηση των αρχείων 8) Ο εγγεγραμμένος χρήστης επιλέγει μία φωτογραφία από τα αρχεία της συσκευής 9) Η επιλεγμένη φωτογραφία προστίθεται στο πεδίο δημοσίευσης 10) Ο εγγεγραμμένος χρήστης πατάει το κουμπί «Δημοσίευση» 11) Το σύστημα εμφανίζει τη δημοσίευση του χρήστη
Εναλλακτικές ροές γεγονότων	1) Στο βήμα 6, ο χρήστης απορρίπτει στην εφαρμογή να έχει πρόσβαση στα αρχεία της συσκευής a) Επαναφορά στο βήμα 2 2) Στο βήμα 4, ο χρήστης αποφασίζει ότι δεν επιθυμεί να προσθέσει εικόνα στη δημοσίευσή του. a) Επαναφορά στο βήμα 2.
Μετά-συνθήκες	Ο εγγεγραμμένος χρήστης έχει δημοσιεύσει ένα νέο το οποίο προβάλλεται στην αρχική οθόνη.

Πίνακας 1.4 Περίπτωση χρήσης: Προβολή αγαπημένων γιατρών

Περίπτωση χρήσης	**Προβολή αγαπημένων γιατρών**
Χρήστης	Εγγεγραμμένος χρήστης
Προϋποθέσεις	1) Π.Χ. 02 Σύνδεση χρήστη 2) Σύνδεση στο διαδίκτυο
Κύρια ροή γεγονότων	1) Ο εγγεγραμμένος χρήστης πλοηγείται στην αρχική οθόνη της εφαρμογής. 2) Το σύστημα εμφανίζει γραμμή πλοήγησης με επιλεγμένη την οθόνη αγαπημένων 3) Το σύστημα προβάλει μία λίστα με τους αγαπημένους ιατρούς του εγγεγραμμένου χρήστη
Εναλλακτικές ροές γεγονότων	1) Στο βήμα 2, δεν υπάρχει σύνδεση στο διαδίκτυο. a) Το σύστημα δεν προβάλλει καμία λίστα αγαπημένων b) Η συσκευή αποκτάει πρόσβαση στο διαδίκτυο 2) Στο βήμα 3, το σύστημα δεν προβάλλει καμία λίστα, διότι ο χρήστης δεν έχει προσθέσει στα αγαπημένα κανέναν ιατρό. a) Επαναφορά στο βήμα 2
Μετά-συνθήκες	Ο εγγεγραμμένος χρήστης παρακολουθεί τη λίστα με τους αγαπημένους του ιατρούς.

Πίνακας 1.5 Περίπτωση χρήσης: Αναζήτηση ιατρού από τη λίστα

Περίπτωση χρήσης	**Αναζήτηση ιατρού από τη λίστα**
Χρήστης	Εγγεγραμμένος χρήστης
Προϋποθέσεις	1) Π.Χ. 02 Σύνδεση χρήστη
Κύρια ροή γεγονότων	1) Ο εγγεγραμμένος χρήστης πλοηγείται στην αρχική οθόνη της εφαρμογής. 2) Το σύστημα εμφανίζει γραμμή πλοήγησης με επιλεγμένη την οθόνη αναζήτησης 3) Το σύστημα προβάλει μία λίστα με τους ιατρούς που είναι εγγεγραμμένοι στο σύστημα

	4) Ο εγγεγραμμένος χρήστης επιλέγει τον ιατρό που επιθυμεί από τη προβαλλόμενη λίστα. 5) Το σύστημα ανακατευθύνει τον χρήστη στην οθόνη του επιλεγμένου ιατρού.
Εναλλακτικές ροές γεγονότων	1) Στο βήμα 3, το σύστημα δεν προβάλλει καμία λίστα, δεν υπάρχει εγγεγραμμένος ιατρός στο σύστημα a) Επαναφορά στο βήμα 2 2) Στο βήμα 4, ο χρήστης δεν επιλέγει κάποιον ιατρό από τα λίστα. a) Επαναφορά στο βήμα 2.
Μετά-συνθήκες	Ο εγγεγραμμένος χρήστης πλοηγείται στην οθόνη του επιλεγμένου ιατρού.

Πίνακας 1.6 Περίπτωση χρήσης: Αναζήτηση ιατρού με βάση το ονοματεπώνυμο

Περίπτωση χρήσης	**Αναζήτηση ιατρού με βάση το ονοματεπώνυμο**
Χρήστης	Εγγεγραμμένος χρήστης
Προϋποθέσεις	1) Π.Χ. 02 Σύνδεση χρήστη
Κύρια ροή γεγονότων	1) Ο εγγεγραμμένος χρήστης πλοηγείται στην αρχική οθόνη της εφαρμογής. 2) Το σύστημα εμφανίζει γραμμή πλοήγησης με επιλεγμένη την οθόνη αναζήτησης 3) Το σύστημα προβάλει ένα πεδίο για να πληκτρολογήσει ο χρήστης το επώνυμο του ιατρού 4) Ο χρήστης πληκτρολογεί το ονοματεπώνυμο του ιατρού που επιθυμεί 5) Το σύστημα εμφανίζει μία λίστα με τους ιατρούς που είναι εγγεγραμμένοι στο σύστημα και το ονοματεπώνυμό τους ταιριάζει με το κείμενο της αναζήτησης. 6) Ο εγγεγραμμένος χρήστης επιλέγει τον ιατρό που επιθυμεί από τη προβαλλόμενη λίστα. 7) Το σύστημα ανακατευθύνει τον χρήστη στην οθόνη του επιλεγμένου ιατρού.
Εναλλακτικές ροές γεγονότων	1) Στο βήμα 4, ο χρήστης δεν πληκτρολογεί κάποιο κείμενο a) Επαναφορά στο βήμα 3 2) Στο βήμα 5, δεν υπάρχει ιατρός που είναι εγγεγραμμένος στο σύστημα και το επώνυμό του να αντιστοιχεί με το κείμενο αναζήτησης. Το σύστημα εμφανίζει μήνυμα «Δεν βρέθηκε ιατρός». a) Επαναφορά στο βήμα 3. 3) Στο βήμα 6, ο χρήστης δεν επιλέγει κάποιον ιατρό από τα λίστα. a) Επαναφορά στο βήμα 2.
Μετά-συνθήκες	Ο εγγεγραμμένος χρήστης πλοηγείται στην οθόνη του επιλεγμένου ιατρού.

Πίνακας 1.7 Περίπτωση χρήσης: Αναζήτηση ιατρού με βάση την ειδικότητα

Περίπτωση χρήσης	**Αναζήτηση ιατρού με βάση την ειδικότητα**
Χρήστης	Εγγεγραμμένος χρήστης
Προϋποθέσεις	1) Π.Χ. 02 Σύνδεση χρήστη
Κύρια ροή γεγονότων	1) Ο εγγεγραμμένος χρήστης πλοηγείται στην αρχική οθόνη της εφαρμογής. 2) Το σύστημα εμφανίζει γραμμή πλοήγησης με επιλεγμένη την οθόνη αναζήτησης 3) Το σύστημα προβάλει ένα μενού το οποίο περιέχει τις ειδικότητες ιατρικής 4) Ο εγγεγραμμένος χρήστης επιλέγει την ειδικότητα που τον ενδιαφέρει 5) Το σύστημα ανακατευθύνει τον χρήστη στην οθόνη αποτελεσμάτων της αναζήτησης

	6) Το σύστημα εμφανίζει μία λίστα με τους ιατρούς που είναι εγγεγραμμένοι στο σύστημα και η ειδικότητά τους ταιριάζει με την επιλεγμένη ειδικότητα. 7) Ο εγγεγραμμένος χρήστης επιλέγει τον ιατρό που επιθυμεί από τη προβαλλόμενη λίστα. 8) Το σύστημα ανακατευθύνει τον χρήστη στην οθόνη του επιλεγμένου ιατρού.
Εναλλακτικές ροές γεγονότων	1) Στο βήμα 4, ο χρήστης δεν επιλέγει κάποια ειδικότητα a) Επαναφορά στο βήμα 3 2) Στο βήμα 6, δεν υπάρχει ιατρός που είναι εγγεγραμμένος στο σύστημα και η ειδικότητά του να αντιστοιχεί με την επιλεγμένη ειδικότητα. Το σύστημα εμφανίζει μήνυμα «Δεν υπάρχουν ιατροί». a) Επαναφορά στο βήμα 4 3) Στο βήμα 7, ο χρήστης δεν επιλέγει κάποιον ιατρό από τα λίστα. a) Επαναφορά στο βήμα 6.
Μετά-συνθήκες	Ο εγγεγραμμένος χρήστης πλοηγείται στην οθόνη του επιλεγμένου ιατρού.

Πίνακας 1.8 Περίπτωση χρήσης: Αναζήτηση ιατρού από τον χάρτη

Περίπτωση χρήσης	**Αναζήτηση ιατρού από τον χάρτη**
Χρήστης	Εγγεγραμμένος χρήστης
Προϋποθέσεις	1) Π.Χ. 02 Σύνδεση χρήστη
Κύρια ροή γεγονότων	1) Ο εγγεγραμμένος χρήστης πλοηγείται στην αρχική οθόνη της εφαρμογής. 2) Το σύστημα εμφανίζει γραμμή πλοήγησης με επιλεγμένη την οθόνη αναζήτησης 3) Το σύστημα προβάλει ένα κουμπί με εικόνα χάρτη. 4) Ο εγγεγραμμένος χρήστης πατάει το κουμπί του χάρτη 5) Το σύστημα ανακατευθύνει τον χρήστη στην οθόνη του χάρτη 6) Το σύστημα ζητάει πρόσβαση στη τοποθεσία του χρήστη (μόνο τη πρώτη φορά ή μέχρι ο χρήστης να παραχωρήσει τη πρόσβαση στην τοποθεσία του) 7) Ο εγγεγραμμένος χρήστης επιτρέπει στο σύστημα να έχει πρόσβαση στη τοποθεσία του 8) Ο χάρτης πλοηγείται κοντά στη τοποθεσία του 9) Το σύστημα εμφανίζει δείκτες επάνω στον χάρτη, οι οποίοι αντιστοιχούν σε ιατρούς. 10) Ο εγγεγραμμένος χρήστης επιλέγει έναν δείκτη 11) Το σύστημα εμφανίζει τη κάρτα του ιατρού (ονοματεπώνυμο, αξιολόγηση, ειδικότητα, διεύθυνση) που αντιστοιχεί στον δείκτη του χάρτη 12) Ο εγγεγραμμένος χρήστης επιλέγει τον ιατρό από τη προβαλλόμενη κάρτα. 13) Το σύστημα ανακατευθύνει τον χρήστη στην οθόνη του επιλεγμένου ιατρού.
Εναλλακτικές ροές γεγονότων	1) Στο βήμα 4, ο χρήστης δεν πατάει το κουμπί του χάρτη a) Επαναφορά στο βήμα 3 2) Στο βήμα 7, ο χρήστης δεν επιτρέπει στο σύστημα να έχει πρόσβαση στη τοποθεσία της συσκευής του χρήστη a) Μετάβαση στο βήμα 9. 3) Στο βήμα 9, το σύστημα δεν εμφανίζει δείκτες διότι δεν υπάρχουν εγγεγραμμένοι ιατροί 4) Στο βήμα 10, ο εγγεγραμμένος χρήστης δεν επιλέγει κάποιο δείκτη a) Επαναφορά στο βήμα 9.

	5) Στο βήμα 12, ο χρήστης δεν επιλέγει τον ιατρό από τη προβαλλόμενη κάρτα a) Επαναφορά στο βήμα 9.
Μετά-συνθήκες	Ο εγγεγραμμένος χρήστης πλοηγείται στην οθόνη του επιλεγμένου ιατρού.

Πίνακας 1.9 Περίπτωση χρήσης: Προβολή των ραντεβού του ασθενή

Περίπτωση χρήσης	**Προβολή των ραντεβού του ασθενή**
Χρήστης	Εγγεγραμμένος χρήστης-ασθενής
Προϋποθέσεις	1) Π.Χ. 02 Σύνδεση χρήστη
Κύρια ροή γεγονότων	1) Ο εγγεγραμμένος χρήστης πατάει το κουμπί για να ανοίξει το πλαϊνό μενού 2) Το σύστημα εμφανίζει το πλαϊνό μενού 3) Ο εγγεγραμμένος χρήστης πατάει το κουμπί «ΤΑ ΡΑΝΤΕΒΟΥ ΜΟΥ» 4) Το σύστημα ανακατευθύνει τον χρήστη στην οθόνη των ραντεβού του ασθενή 5) Το σύστημα εμφανίζει μία λίστα με κάρτες οι οποίες περιέχουν τα ραντεβού που έχει κλείσει ο ασθενής(ημερομηνία, ώρα, ονοματεπώνυμο και διεύθυνση ιατρού.
Εναλλακτικές ροές γεγονότων	1) Στο βήμα 1, ο χρήστης δεν πατάει το κουμπί για να ανοίξει το πλαϊνό μενού 2) Στο βήμα 3, ο χρήστης δεν πατάει το κουμπί «ΤΑ ΡΑΝΤΕΒΟΥ ΜΟΥ» a) Επαναφορά στο βήμα 2 3) Στο βήμα 5, το σύστημα δεν εμφανίζει κάποια λίστα διότι δεν υπάρχουν ραντεβού
Μετά-συνθήκες	Ο εγγεγραμμένος χρήστης πλοηγείται στην οθόνη των ραντεβού του.

Πίνακας 1.10 Περίπτωση χρήσης: Προβολή των ραντεβού του γιατρού

Περίπτωση χρήσης	**Προβολή των ραντεβού του γιατρού**
Χρήστης	Εγγεγραμμένος χρήστης-ιατρός
Προϋποθέσεις	1) Π.Χ. 02 Σύνδεση χρήστη
Κύρια ροή γεγονότων	1) Ο εγγεγραμμένος χρήστης πλοηγείται στην αρχική οθόνη της εφαρμογής 2) Το σύστημα εμφανίζει γραμμή πλοήγησης με επιλεγμένη την οθόνη των ραντεβού 3) Το σύστημα εμφανίζει ένα ημερολόγιο με προεπιλεγμένη τη τρέχουσα ημέρα 4) Ο χρήστης-ιατρός επιλέγει μία ημέρα από το ημερολόγιο 5) Το σύστημα εμφανίζει μία λίστα με τα διαθέσιμα και μη ραντεβού του ιατρού για την επιλεγμένη ημέρα.
Εναλλακτικές ροές γεγονότων	1) Στο βήμα 5, ο ιατρός δεν έχει ορίσει ώρες για ραντεβού. Το σύστημα εμφανίζει μήνυμα «Δεν υπάρχουν διαθέσιμα ραντεβού»
Μετά-συνθήκες	Ο εγγεγραμμένος χρήστης-ιατρός πλοηγείται στην οθόνη των ραντεβού του.

Πίνακας 1.11 Περίπτωση χρήσης: Προσθήκη διαθέσιμων ωρών

Περίπτωση χρήσης	**Προσθήκη διαθέσιμων ωρών**
Χρήστης	Εγγεγραμμένος χρήστης-ιατρός
Προϋποθέσεις	1) Π.Χ. 02 Σύνδεση χρήστη

	2) Σύνδεση στο διαδίκτυο
Κύρια ροή γεγονότων	1) Ο εγγεγραμμένος χρήστης πλοηγείται στην αρχική οθόνη της εφαρμογής 2) Το σύστημα εμφανίζει γραμμή πλοήγησης με επιλεγμένη την οθόνη των ραντεβού 3) Το σύστημα εμφανίζει το κουμπί «Προσθήκη ωρών 4) Ο χρήστης-ιατρός πατάει το κουμπί «Προσθήκη ωρών» 5) Το σύστημα ανακατευθύνει τον χρήστη στη σελίδα επιλογής ωρών επισκέψεως 6) Το σύστημα δίνει την επιλογή στον χρήστη να επιλέξει τη χρονική απόσταση μεταξύ των ραντεβού του 7) Το σύστημα εμφανίζει το κουμπί «Φόρτωση» 8) Ο χρήστης πατάει το κουμπί «Φόρτωση» 9) Το σύστημα δημιουργεί τις διαθέσιμες ώρες του γιατρού, με βάση το ωράριο εργασίας που έχει ορίσει 10) Το σύστημα εμφανίζει τις διαθέσιμες ώρες του γιατρού 11) Το σύστημα δίνει την δυνατότητα στον χρήστη να επιλέξει αν θα συμπεριλαμβάνονται τα Σαββατοκύριακα στις διαθέσιμες ημέρες 12) Το σύστημα εμφανίζει το κουμπί «+»για την προσθήκη επιπλέον ώρας, εκτός από αυτές του ωραρίου τυ χρήστη 13) Ο χρήστης πατάει στο κουμπί «+» 14) Το σύστημα εμφανίζει δύο στήλες που αντιστοιχούν σε ώρες και λεπτά αντίστοιχα 15) Ο χρήστης επιλέγει την ώρα που επιθυμεί να προστεθεί στις διαθέσιμες ώρες επισκέψεως. 16) Το σύστημα εμφανίζει το κουμπί «Αποθήκευση ώρας» και το κουμπί «Άκυρο» 17) Ο χρήστης πατάει το κουμπί «Αποθήκευση ώρας» 18) Το σύστημα προσθέτει την συγκεκριμένη ώρα στις συνολικά διαθέσιμες ώρες επισκέψεως 19) Το σύστημα εμφανίζει το κουμπί «Αποθήκευση» 20) Ο χρήστης πατάει το κουμπί «Αποθήκευση» 21) Το σύστημα ενημερώνει και προσθέτει για κάθε διαθέσιμη ημέρα, τις διαθέσιμες ώρες που πρόσθεσε ο χρήστης 22) Το σύστημα εμφανίζει μήνυμα επιτυχίας 23) Το σύστημα ανακατευθύνει τον χρήστη στην οθόνη επιλογής ωρών επισκέψεως
Εναλλακτικές ροές γεγονότων	1) Στο βήμα 4 ο χρήστης δεν πατάει το κουμπί. a) Επαναφορά στο βήμα 3 2) Στο βήμα 8, ο χρήστης δεν πατάει το κουμπί. a) Επαναφορά στο βήμα 7 3) Στο βήμα 8, δεν υπάρχει σύνδεση στο διαδίκτυο. a) Επαναφορά στο βήμα 7 4) Στο βήμα 13, ο χρήστης δεν πατάει το κουμπί «+». a) Επαναφορά στο βήμα 12 5) Στο βήμα 15, ο χρήστης δεν επιλέγει ώρα a) Επαναφορά στο βήμα 12. 6) Στο βήμα 17, ο χρήστης πατάει το κουμπί «Άκυρο». a) Επαναφορά στο βήμα 12. 7) Στο βήμα 17, δεν υπάρχει σύνδεση στο διαδίκτυο. a) Επαναφορά στο βήμα 16. 8) Στο βήμα 20, ο χρήστης δεν πατάει το κουμπί «Αποθήκευση» a) Επαναφορά στο βήμα 19 9) Στο βήμα 20, δεν υπάρχει σύνδεση στο διαδίκτυο. a) Επαναφορά στο βήμα 19
Μετά-συνθήκες	Ο εγγεγραμμένος χρήστης-ιατρός πλοηγείται στην οθόνη των ραντεβού του.

	Προβάλλονται τα διαθέσιμα και μη ραντεβού.

Πίνακας 1.12 Περίπτωση χρήσης: Προσθήκη διαθέσιμης ώρας

Περίπτωση χρήσης	**Προσθήκη διαθέσιμης ώρας**
Χρήστης	Εγγεγραμμένος χρήστης-ιατρός
Προϋποθέσεις	1) Π.Χ. 02 Σύνδεση χρήστη 2) Σύνδεση στο διαδίκτυο
Κύρια ροή γεγονότων	1) Ο εγγεγραμμένος χρήστης πλοηγείται στην αρχική οθόνη της εφαρμογής 2) Το σύστημα εμφανίζει γραμμή πλοήγησης με επιλεγμένη την οθόνη των ραντεβού 3) Το σύστημα εμφανίζει το κουμπί «Προσθήκη ώρας» 4) Ο χρήστης πατάει το κουμπί «Προσθήκη ώρας» 5) Το σύστημα εμφανίζει δύο στήλες που αντιστοιχούν σε ώρες και λεπτά αντίστοιχα 6) Ο χρήστης επιλέγει την ώρα που επιθυμεί να προστεθεί στις διαθέσιμες ώρες επισκέψεως. 7) Το σύστημα εμφανίζει το κουμπί «Αποθήκευση ώρας» και το κουμπί «Άκυρο» 8) Ο χρήστης πατάει το κουμπί «Αποθήκευση ώρας» 9) Το σύστημα προσθέτει την συγκεκριμένη ώρα στις συνολικά διαθέσιμες ώρες επισκέψεως
Εναλλακτικές ροές γεγονότων	1) Στο βήμα 4, ο χρήστης δεν πατάει το κουμπί. a) Επαναφορά στο βήμα 7 2) Στο βήμα 6, ο χρήστης δεν επιλέγει ώρα a) Επαναφορά στο βήμα 5 3) Στο βήμα 8, ο χρήστης πατάει το κουμπί «Άκυρο». a) Επαναφορά στο βήμα 3. 4) Στο βήμα 8, δεν υπάρχει σύνδεση στο διαδίκτυο. a) Επαναφορά στο βήμα 7.
Μετά-συνθήκες	Ο εγγεγραμμένος χρήστης-ιατρός πλοηγείται στην οθόνη των ραντεβού του. Προβάλλονται τα διαθέσιμα και μη ραντεβού.

Πίνακας 1.13 Περίπτωση χρήσης: Δημιουργία ραντεβού

Περίπτωση χρήσης	**Δημιουργία ραντεβού**
Χρήστης	Εγγεγραμμένος χρήστης
Προϋποθέσεις	1) Π.Χ. 02 Σύνδεση χρήστη 2) Σύνδεση στο διαδίκτυο 3) Επιλογή ιατρού
Κύρια ροή γεγονότων	1) Ο εγγεγραμμένος χρήστης πλοηγείται στην οθόνη του ιατρού που έχει επιλέξει 2) Το σύστημα εμφανίζει το εικονίδιο που αντιστοιχεί για τη δημιουργία νέου ραντεβού 3) Ο χρήστης πατάει το εικονίδιο 4) Το σύστημα ανακατευθύνει τον χρήστη στην οθόνη των ραντεβού. 5) Το σύστημα εμφανίζει ένα ημερολόγιο με προεπιλεγμένη την τρέχουσα ημέρα 6) Ο χρήστης επιλέγει την ημέρα που επιθυμεί 7) Το σύστημα εμφανίζει μία λίστα με τις διαθέσιμες ώρες επισκέψεως. 8) Ο χρήστης επιλέγει την ώρα που επιθυμεί

	9) Το σύστημα εμφανίζει μια φόρμα η οποία είναι προσυμπληρωμένη με τα στοιχεία του χρήστη (όνομα, επώνυμο, τηλέφωνο, email) 10) Ο χρήστης πατάει το κουμπί «Ολοκλήρωση». 11) Το σύστημα εμφανίζει μήνυμα επιτυχίας και ανακατευθύνει τον χρήστη στην οθόνη των ραντεβού
Εναλλακτικές ροές γεγονότων	1) Στο βήμα 3 ο χρήστης δεν πατάει το εικονίδιο a) Επαναφορά στο βήμα 2 2) Στο βήμα 7, το σύστημα δεν εμφανίζει κάποια λίστα, διότι δεν υπάρχουν διαθέσιμες ώρες επισκέψεως. Εμφανίζεται το μήνυμα «Δεν υπάρχουν διαθέσιμα ραντεβού» a) Επαναφορά στο βήμα 6 3) Στο βήμα 7, το σύστημα εμφανίζει μήνυμα «Δοκιμάστε άλλη μέρα», διότι ο ιατρός δεν έχει ορίσει τις διαθέσιμες ώρες για την συγκεκριμένη μέρα a) Επαναφορά στο βήμα 6 4) Στο βήμα 10, ο χρήστης κλείνει τη φόρμα. Το ραντεβού ακυρώνεται. a) Επαναφορά στο βήμα 6 5) Στο βήμα 10, δεν υπάρχει σύνδεση στο διαδίκτυο. a) Επαναφορά στο βήμα 9.
Μετά-συνθήκες	Ο εγγεγραμμένος χρήστης έχει κλείσει ραντεβού για μία συγκεκριμένη ημερομηνία, για τον ιατρό που επιθυμεί.

Πίνακας 1.14 Περίπτωση χρήσης: Δημιουργία αξιολόγησης

Περίπτωση χρήσης	**Δημιουργία αξιολόγησης**
Χρήστης	Εγγεγραμμένος χρήστης
Προϋποθέσεις	1) Π.Χ. 02 Σύνδεση χρήστη 2) Σύνδεση στο διαδίκτυο 3) Επιλογή ιατρού
Κύρια ροή γεγονότων	1) Ο εγγεγραμμένος χρήστης πλοηγείται στην οθόνη του ιατρού που έχει επιλέξει 2) Το σύστημα εμφανίζει τη καρτέλα «Πληροφορίες» και τη καρτέλα «Αξιολογήσεις» 3) Ο χρήστης επιλέγει τη καρτέλα «Αξιολογήσεις» 4) Το σύστημα αλλάζει το περιεχόμενο της οθόνης με βάση την επιλεγμένη καρτέλα 5) Το σύστημα εμφανίζει μία λίστα με όλες τις αξιολογήσεις που αφορούν τον συγκεκριμένο ιατρό 6) Το σύστημα εμφανίζει το κουμπί «Αφήστε μία αξιολόγηση» 7) Ο χρήστης πατάει το κουμπί «Αφήστε μία αξιολόγηση» 8) Το σύστημα εμφανίζει μία φόρμα (αστέρια βαθμολογίας, κείμενο αξιολόγησης) 9) Ο χρήστης συμπληρώνει τη φόρμα 10) Ο χρήστης πατάει το κουμπί «Αποθήκευση» 11) Το σύστημα αποθηκεύει την αξιολόγηση του χρήστη για τον συγκεκριμένο ιατρό 12) Το σύστημα ενημερώνει τη λίστα με όλες τις αξιολογήσεις.
Εναλλακτικές ροές γεγονότων	1) Στο βήμα 3 ο χρήστης δεν επιλέγει την καρτέλα «Αξιολογήσεις» a) Επαναφορά στο βήμα 2. 2) Στο βήμα 5, το σύστημα δεν εμφανίζει κάποια λίστα, διότι δεν υπάρχουν αξιολογήσεις για τον συγκεκριμένο ιατρό. Εμφανίζεται μήνυμα «Δεν υπάρχουν αξιολογήσεις» a) Επαναφορά στο βήμα 4

	3) Στο βήμα 6, το σύστημα δεν εμφανίζεται το κουμπί «Αφήστε μία αξιολόγηση» διότι ο χρήστης έχει δημιουργήσει ήδη μία αξιολόγηση για τον συγκεκριμένο ιατρό. a) Επαναφορά στο βήμα 5. 4) Στο βήμα 10, ο χρήστης κλείνει τη φόρμα. Η αξιολόγηση ακυρώνεται a) Επαναφορά στο βήμα 6. 5) Στο βήμα 10, ο χρήστης πατάει το κουμπί «Άκυρο» και η φόρμα κλείνει. Η αξιολόγηση ακυρώνεται. a) Επαναφορά στο βήμα 6. 6) Στο βήμα 10, δεν υπάρχει σύνδεση στο διαδίκτυο. a) Επαναφορά στο βήμα 9.
Μετά-συνθήκες	Ο εγγεγραμμένος χρήστης έχει υποβάλει μία αξιολόγηση για τον ιατρό που επιθυμεί.

Πίνακας 1.15 περίπτωση χρήσης: Επικοινωνία μέσω ηλεκτρονικού ταχυδρομείου

Περίπτωση χρήσης	**Επικοινωνία μέσω ηλεκτρονικού ταχυδρομείου**
Χρήστης	Εγγεγραμμένος χρήστης
Προϋποθέσεις	1) Π.Χ. 02 Σύνδεση χρήστη 2) Σύνδεση στο διαδίκτυο 3) Επιλογή ιατρού
Κύρια ροή γεγονότων	1) Ο εγγεγραμμένος χρήστης πλοηγείται στην οθόνη του ιατρού που έχει επιλέξει 2) Το σύστημα εμφανίζει το εικονίδιο που αντιστοιχεί για την επικοινωνία μέσω email 3) Ο χρήστης επιλέγει το εικονίδιο 4) Το σύστημα ανοίγει την προεπιλεγμένη εφαρμογή για αποστολή mail, που είναι εγκατεστημένη στο κινητό τηλέφωνο. 5) Ο χρήστης συμπληρώνει το θέμα και το μήνυμα που επιθυμεί να γράψει. 6) Ο χρήστης αποστέλλει το email μέσω της εφαρμογής του έξυπνου κινητού
Εναλλακτικές ροές γεγονότων	1) Στο βήμα 3, ο χρήστης δεν πατάει στο εικονίδιο a) Επαναφορά στο βήμα 2. 2) Στο βήμα 6, ο χρήστης αποφασίζει ότι δεν επιθυμεί να στείλει email. a) Επαναφορά στο βήμα 2. 3) Στο βήμα 6,δεν υπάρχει σύνδεση στο διαδίκτυο. Το email δεν αποστέλλεται.
Μετά-συνθήκες	Ο εγγεγραμμένος χρήστης έχει αποστείλει email στον ιατρό που επιθυμεί.

Πίνακας 1.16 Περίπτωση χρήσης: Επικοινωνία μέσω τηλεφωνικής κλήσης

Περίπτωση χρήσης	**Επικοινωνία μέσω τηλεφωνική κλήσης**
Χρήστης	Εγγεγραμμένος χρήστης
Προϋποθέσεις	1) Π.Χ. 02 Σύνδεση χρήστη 2) Επιλογή ιατρού
Κύρια ροή γεγονότων	1) Ο εγγεγραμμένος χρήστης πλοηγείται στην οθόνη του ιατρού που έχει επιλέξει 2) Το σύστημα εμφανίζει το εικονίδιο που αντιστοιχεί για την επικοινωνία μέσω τηλεφωνικής κλήσης 3) Ο χρήστης επιλέγει το εικονίδιο 4) Το σύστημα ανοίγει την προεπιλεγμένη εφαρμογή για κλήσεις. Το τηλέφωνο του ιατρού είναι προσυμπληρωμένο.

	5) Ο χρήστης πατάει το κουμπί για να πραγματοποιήσει κλήση. 6) Το κινητό τηλέφωνο πραγματοποιεί την τηλεφωνική κλήση προς τον ιατρό
Εναλλακτικές ροές γεγονότων	1) Στο βήμα 3 ο χρήστης δεν πατάει στο εικονίδιο a) Επαναφορά στο βήμα 2. 2) Στο βήμα 6 ο χρήστης αποφασίζει ότι δεν επιθυμεί να πραγματοποιήσει τηλεφωνική κλήση. a) Επαναφορά στο βήμα 2.
Μετά-συνθήκες	Ο εγγεγραμμένος χρήστης έχει πραγματοποιήσει τηλεφωνική κλήση με τον ιατρό που επιθυμεί.

Πίνακας 1.17 Περίπτωση χρήσης: Προσθήκη γιατρού στα αγαπημένα

Περίπτωση χρήσης	**Προσθήκη γιατρού στα αγαπημένα**
Χρήστης	Εγγεγραμμένος χρήστης
Προϋποθέσεις	1) Π.Χ. 02 Σύνδεση χρήστη 2) Σύνδεση στο διαδίκτυο 3) Επιλογή ιατρού
Κύρια ροή γεγονότων	1) Ο εγγεγραμμένος χρήστης πλοηγείται στην οθόνη του ιατρού που έχει επιλέξει 2) Το σύστημα εμφανίζει το εικονίδιο που αντιστοιχεί στα αγαπημένα 3) Ο χρήστης πατάει στο εικονίδιο 4) Το σύστημα αποθηκεύει τον ιατρό στα αγαπημένα του χρήστη. 5) Το σύστημα εμφανίζει μήνυμα «Προστέθηκε στα αγαπημένα!»
Εναλλακτικές ροές γεγονότων	1) Στο βήμα 3 ο χρήστης δεν επιλέγει το εικονίδιο a) Επαναφορά στο βήμα 2. 2) Στο βήμα 4 ο χρήστης έχει προσθέσει ήδη τον ιατρό στα αγαπημένα του a) Το σύστημα αφαιρεί τον ιατρό από τα αγαπημένα b) Επαναφορά στο βήμα 2. 3) Στο βήμα 4, δεν υπάρχει σύνδεση στο διαδίκτυο.
Μετά-συνθήκες	Ο εγγεγραμμένος χρήστης έχει αποθηκεύσει στα αγαπημένα, τον ιατρό που επιθυμεί.

Πίνακας 1.18 Περίπτωση χρήσης: Ενημέρωση των προσωπικών στοιχείων του ασθενή

Περίπτωση χρήσης	**Ενημέρωση των προσωπικών στοιχείων του ασθενή**
Χρήστης	Εγγεγραμμένος χρήστης
Προϋποθέσεις	1) Π.Χ. 02 Σύνδεση χρήστη 2) Σύνδεση χρήστη ως ασθενής
Κύρια ροή γεγονότων	1) Ο εγγεγραμμένος χρήστης πλοηγείται στην αρχική οθόνη της εφαρμογής 2) Το σύστημα εμφανίζει γραμμή πλοήγησης με επιλεγμένη την οθόνη των προσωπικών πληροφοριών 3) Το σύστημα εμφανίζει τα προσωπικά στοιχεία του χρήστη 4) Ο χρήστης επιλέγει την προσωπική φωτογραφία του για να την αντικαταστήσει με μία νέα 5) Το σύστημα ζητάει από τον χρήστη να του επιτρέπει την πρόσβαση στα αρχεία της συσκευής (μέχρι ο χρήστης παραχωρήσει τη πρόσβαση στα αρχεία της συσκευής) 6) Ο χρήστης επιτρέπει στη συσκευή να έχει πρόσβαση στα αρχεία της συσκευής 7) Το σύστημα ανοίγει την πλοήγηση των αρχείων

	8) Ο εγγεγραμμένος χρήστης επιλέγει μία φωτογραφία από τα αρχεία της συσκευής 9) Η φωτογραφία του χρήστη αλλάζει 10) Ο χρήστης, πατάει στο πεδίο «Ύψος» 11) Το σύστημα εμφανίζει μία λίστα προκειμένου ο χρήστης να επιλέξει το ύψος του. 12) Ο χρήστης επιλέγει το ύψος του. 13) Ο χρήστης πατάει το κουμπί «ΟΚ» 14) Το σύστημα ενημερώνει τη τιμή του ύψους. 15) Ο χρήστης, πατάει στο πεδίο «Βάρος» 16) Το σύστημα εμφανίζει μία λίστα προκειμένου ο χρήστης να επιλέξει το βάρος του. 17) Ο χρήστης επιλέγει το βάρος του. 18) Ο χρήστης πατάει το κουμπί «ΟΚ» 19) Το σύστημα ενημερώνει τη τιμή του βάρους.
Εναλλακτικές ροές γεγονότων	1) Στο βήμα 4 δεν επιλέγει τη προσωπική του φωτογραφία για να την αντικαταστήσει. a) Μετάβαση στο βήμα 12. 2) Στο βήμα 6 ο χρήστης δεν επιτρέπει την πρόσβαση στα αρχεία της συσκευής a) Επαναφορά στο βήμα 3. 3) Στο βήμα 8 ο χρήστης δεν επιλέγει κάποια φωτογραφία από τα αρχεία της συσκευής a) Επαναφορά στο βήμα 3. 4) Στο βήμα 10 ο χρήστης δεν πατάει στο πεδίο «Ύψος». a) Μετάβαση στο βήμα 15 5) Στο βήμα 12, ο χρήστης δεν επιλέγει ύψος και πατάει το κουμπί «Άκυρο» a) Επαναφορά στο βήμα 3. 6) Στο βήμα 15, ο χρήστης δεν πατάει στο πεδίο «Βάρος» a) Επαναφορά στο βήμα 3. 7) Στο βήμα 17, ο χρήστης δεν επιλέγει βάρος και πατάει το κουμπί «Άκυρο». a) Επαναφορά στο βήμα 3.
Μετά-συνθήκες	Ο εγγεγραμμένος χρήστης έχει αλλάξει τα προσωπικά του στοιχεία

Πίνακας 1.19 Περίπτωση χρήσης: Ενημέρωση των προσωπικών στοιχείων του γιατρού

Περίπτωση χρήσης	**Ενημέρωση των προσωπικών στοιχείων του γιατρού**
Χρήστης	Εγγεγραμμένος χρήστης
Προϋποθέσεις	1) Π.Χ. 02 Σύνδεση χρήστη 2) Σύνδεση χρήστη ως ιατρός
Κύρια ροή γεγονότων	1) Ο εγγεγραμμένος χρήστης πλοηγείται στην αρχική οθόνη της εφαρμογής 2) Το σύστημα εμφανίζει γραμμή πλοήγησης με επιλεγμένη την οθόνη των προσωπικών πληροφοριών 3) Το σύστημα εμφανίζει τα προσωπικά στοιχεία του χρήστη 4) Ο χρήστης επιλέγει την προσωπική φωτογραφία του για να την αντικαταστήσει με μία νέα 5) Το σύστημα ζητάει από τον χρήστη να του επιτρέπει την πρόσβαση στα αρχεία της συσκευής (μέχρι ο χρήστης παραχωρήσει τη πρόσβαση στα αρχεία της συσκευής) 6) Ο χρήστης επιτρέπει στη συσκευή να έχει πρόσβαση στα αρχεία της συσκευής

	7) Το σύστημα ανοίγει την πλοήγηση των αρχείων 8) Ο εγγεγραμμένος χρήστης επιλέγει μία φωτογραφία από τα αρχεία της συσκευής 9) Η φωτογραφία του χρήστη αλλάζει 10) Ο χρήστης επιλέγει το πεδίο «Περιγραφή» 11) Το σύστημα εμφανίζει μία αναδυόμενη φόρμα εισαγωγής κειμένου. 12) Ο χρήστης να πληκτρολογεί τη νέα περιγραφή. 13) Ο χρήστης πατάει το κουμπί «Αποθήκευση» 14) Το σύστημα ενημερώνει τη περιγραφή του χρήστη. 15) Ο χρήστης επιλέγει το πεδίο «Ωράριο» 16) Το σύστημα εμφανίζει δύο λίστες προκειμένου ο χρήστης να επιλέξει το ωράριό του. 17) Ο χρήστης επιλέγει την ώρα και τα λεπτά. 18) Ο χρήστης πατάει το κουμπί «Αποθήκευση» 19) Το σύστημα ενημερώνει τη τιμή του ωραρίου.
Εναλλακτικές ροές γεγονότων	1) Στο βήμα 4, ο χρήστης δεν επιλέγει τη προσωπική του φωτογραφία για να την αντικαταστήσει. a) Μετάβαση στο βήμα 10. 2) Στο βήμα 6 ο χρήστης δεν επιτρέπει την πρόσβαση στα αρχεία της συσκευής a) Επαναφορά στο βήμα 3. 3) Στο βήμα 8 ο χρήστης δεν επιλέγει κάποια φωτογραφία από τα αρχεία της συσκευής a) Επαναφορά στο βήμα 3. 4) Στο βήμα 10 ο χρήστης δεν επιλέγει το πεδίο «Περιγραφή». a) Μετάβαση στο βήμα 15 5) Στο βήμα 12 ο χρήστης πατάει το κουμπί «Άκυρο» και κλείνει η φόρμα εισαγωγής κειμένου. a) Επαναφορά στο βήμα 3. 6) Στο βήμα 15, ο χρήστης δεν πατάει στο πεδίο «Ωράριο» a) Επαναφορά στο βήμα 3. 7) Στο βήμα 17, ο χρήστης πατάει το κουμπί «Άκυρο» και δεν επιλέγει να επεξεργαστεί το ωράριο a) Επαναφορά στο βήμα 3.
Μετά-συνθήκες	Ο εγγεγραμμένος χρήστης έχει αλλάξει τα προσωπικά του στοιχεία

1.5 Επίλογος

Στο παρόν κεφάλαιο παρουσιάστηκαν και αναλύθηκαν ο σκοπός και οι απαιτήσεις της εφαρμογής. Επιπλέον περιεγράφηκαν πλήρως οι περιπτώσεις χρήσης των υπηρεσιών της παρούσας εφαρμογής. Στο παρακάτω κεφάλαιο παρουσιάζονται και αναλύονται οι τεχνολογίες που χρησιμοποιήθηκαν για την ανάπτυξη της εφαρμογής.

Κεφάλαιο 2ο: Περιγραφή των τεχνολογιών

2.1 Εισαγωγή

Η παρούσα εφαρμογή αναπτύχθηκε με τη χρήση του Flutter Framework. Το Flutter Framework αποτελεί ένα εργαλείο το οποίο επιτρέπει να δημιουργούνται εφαρμογές, ικανές να λειτουργούν σε πληθώρα πλατφόρμων, χρησιμοποιώντας μία κοινή γλώσσα προγραμματισμού. Το Flutter Framework χρησιμοποιεί τη γλώσσα προγραμματισμού Dart, η οποία είναι μία αντικειμενοστραφής γλώσσα προγραμματισμού. Ως βάση δεδομένων χρησιμοποιήθηκε το Firestore του Firebase το οποίο είναι μια NoSQL βάση δεδομένων που φιλοξενείται στο cloud για την αποθήκευση και τον συγχρονισμό των δεδομένων. Για την συγγραφή και την επεξεργασία του πηγαίου κώδικα, χρησιμοποιήθηκε το πρόγραμμα Visual Studio Code.

2.2 Flutter

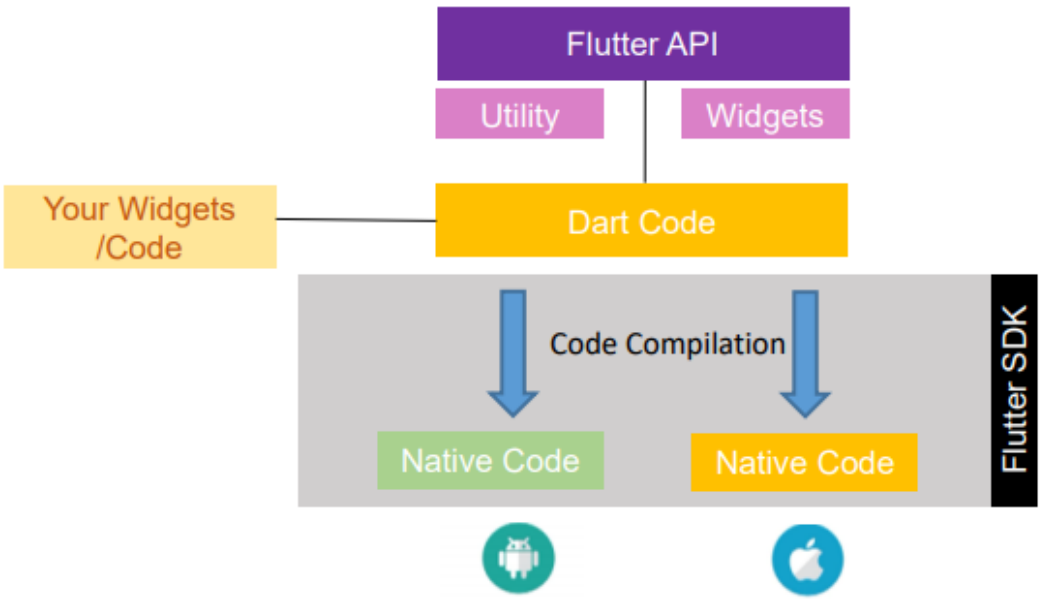
Μία εφαρμογή που κάνει άμεση χρήση ενός συγκεκριμένου λειτουργικού συστήματος ονομάζεται native εφαρμογή [3]. Στις εφαρμογές οι οποίες σχεδιάζονται και αναπτύσσονται για έξυπνα κινητά τηλέφωνα, τα πιο διαδεδομένα λειτουργικά συστήματα είναι το Android και το iOS. Μία native εφαρμογή δεν μπορεί να εκτελεστεί σε μία συσκευή η οποία χρησιμοποιεί διαφορετική πλατφόρμα. Συγκεκριμένα μία εφαρμογή η οποία εκτελείται σε λειτουργικό σύστημα Android, δεν μπορεί να λειτουργήσει σε μία iOS συσκευή. Αντίστοιχα, μία iOS εφαρμογή δεν μπορεί να λειτουργήσει σε μία Android συσκευή.

Στις μέρες μας, όλο και πιο συχνά διαδίδεται η έννοια των εφαρμογών πολλαπλών πλατφορμών ή αλλιώς cross-platform. Λογισμικό πολλαπλών πλατφορμών είναι οποιαδήποτε εφαρμογή λογισμικού που λειτουργεί ταυτόχρονα σε πολλαπλά λειτουργικά συστήματα ή συσκευές, που συχνά αναφέρονται ως πλατφόρμες. Πλατφόρμα σημαίνει ένα λειτουργικό σύστημα όπως Android, iOS, Windows ή MacOS [4]. Οι cross-platform εφαρμογές προκειμένου να εκτελούνται ταυτόχρονα σε διαφορετικά λειτουργικά συστήματα, χρησιμοποιούν την ίδια γλώσσα προγραμματισμού και βάση κώδικα. Με αυτό τον τρόπο, μειώνεται δραματικά ο χρόνος υλοποίησης μίας εφαρμογής διότι διαφορετικά θα χρειαζόταν να δημιουργηθούν δύο ή περισσότερες διαφορετικές εφαρμογές. Ταυτόχρονα, μία εφαρμογή η οποία εκτελείται σε όλα τα λειτουργικά συστήματα, με τον ίδιο πηγαίο κώδικα, έχει ως αποτέλεσμα τη μείωση του κόστους της δημιουργίας της.

Το Flutter Framework αποτελεί ένα cross-platform εργαλείο διεπαφής χρήστη που αναπτύχθηκε από την Google για τη δημιουργία εφαρμογών για έξυπνα κινητά (Android και iOS), εφαρμογές ιστού και επιτραπέζιους υπολογιστές, από μία μόνο βάση κώδικα [5]. Στο παράρτημα Α παρουσιάζεται ο οδηγός της εγκατάστασης του Flutter Framework.

Για να δημιουργηθεί μία εφαρμογή θα πρέπει να χρησιμοποιηθεί μία συγκεκριμένη γλώσσα προγραμματισμού. Για την ανάπτυξη εφαρμογών για συσκευές με το iOS λειτουργικό σύστημα, χρησιμοποιείται είτε η γλώσσα Swift , είτε Objective C/ Objective C++ , ενώ για την ανάπτυξη Android εφαρμογών χρησιμοποιείται είτε η γλώσσα Java είτε η γλώσσα Kotlin. Αντιθέτως, το Flutter Framework χρησιμοποιεί την γλώσσα προγραμματισμού Dart με τέτοιο τρόπο ώστε να μπορεί να δημιουργήσει, ταυτόχρονα και με τον ίδιο ακριβώς πηγαίο κώδικα, native εφαρμογές για τις πλατφόρμες Android και iOS. Για να πραγματοποιηθεί αυτό, το Flutter, μεταγλωττίζει τον Dart κώδικα σε native κώδικα του κάθε λειτουργικού συστήματος και αυτό γίνεται με την βοήθεια του

Flutter SDK. Το Flutter χωρίζεται σε δύο σημαντικά τμήματα, το Flutter SDK και το Flutter Framework.



Σχήμα 2.1 Μεταγλώττιση του Dart κώδικα σε εγγενή κώδικα[1]

2.2.1 Flutter SDK

Το Flutter SDK είναι ένα απαραίτητο εργαλείο και αποτελεί μία συλλογή από επιμέρους εργαλεία τα οποία βοηθούν στην ανάπτυξη της εφαρμογής. Η συλλογή αυτή περιλαμβάνει όλα τα απαραίτητα εργαλεία τα οποία χρειάζονται για την μεταγλώττιση του κώδικα Dart σε native κώδικα, είτε για Android, είτε για iOS. Συγκεκριμένα το Flutter SDK περιλαμβάνει [6]:

1) Τη δική του μηχανή απόδοσης OpenGL, χάρις την οποία δεν χρειάζεται να χρησιμοποιεί γέφυρα σε native SDKs για τη δημιουργία και την αλληλεπίδραση με την εφαρμογή
2) Συλλογή πληθώρας γραφικών στοιχείων που υλοποιούν το Material (Android) και το Cupertino (iOS) σχεδιασμό
3) API για καλύτερο έλεγχο της εφαρμογής (testing)
4) Το εργαλείο Dart Dev Tools. Χρησιμοποιείται για έλεγχο στην διάταξη της διεπαφής χρήστη, εντοπισμό σφαλμάτων αλλά και για την προβολή γενικών πληροφοριών, όπως η απόδοση της εφαρμογής [7]
5) Εργαλεία γραμμής εντολών για τη δημιουργία τη δοκιμή και τη μεταγλώττιση της εφαρμογής

2.2.2 Flutter Framework

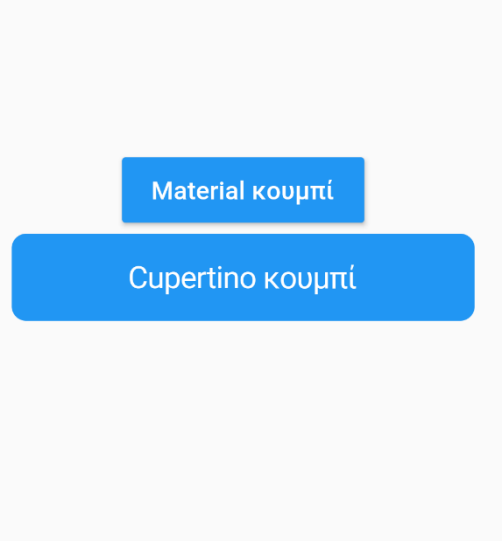
Το Flutter Framework αποτελεί μία βιβλιοθήκη από εργαλεία για τη γλώσσα προγραμματισμού Dart, με τα οποία δημιουργείται και διαμορφώνεται η εφαρμογή. Πρόκειται για μία συλλογή η οποία περιέχει μεγάλο πλήθος επαναχρησιμοποιήσιμων εργαλείων τα οποία ονομάζονται widgets. Τα widgets είναι εργαλεία το οποία χρησιμοποιούνται από τους προγραμματιστές, για την διαμόρφωση της διεπαφής του χρήστη. Τέτοια widgets μπορούν να είναι κουμπιά, καρτέλες, πεδία κειμένου και μπορούν να διαμορφωθούν με τέτοιο τρόπο ώστε να μπορούν να προσαρμοστούν στις ανάγκες του χρήστη, δίνοντάς τους λειτουργικότητα. Το Flutter Framework παρέχει στον προγραμματιστή τον

[1] https://miro.medium.com/max/555/1*mqm3Dmplx9SjhKI8k5Pf3Q.png

Material σχεδιασμό, ο οποίος εφαρμόζεται κυρίως σε Android συσκευές, αλλά και τον Cupertino σχεδιασμό ο οποίος εφαρμόζεται σε iOS. Ο προγραμματιστής έχει τη δυνατότητα να χρησιμοποιήσει τα widgets, όπως ο ίδιος επιθυμεί. Ακόμη, μπορεί να δημιουργήσει δικά του widget [8].

Κώδικας 2.1 Widget κουμπιών στον πηγαίο κώδικα σε Material και Cupertino

```
RaisedButton(
  child: Text("Material κουμπί"),
  onPressed: () {
    print("Material κουμπί");
  },
  color: Colors.blue,
  textColor: Colors.white,
),
CupertinoButton(
  child: Text("Material κουμπί"),
  onPressed: () {
    print("Material κουμπί");
  },
  color: Colors.blue,
)
```



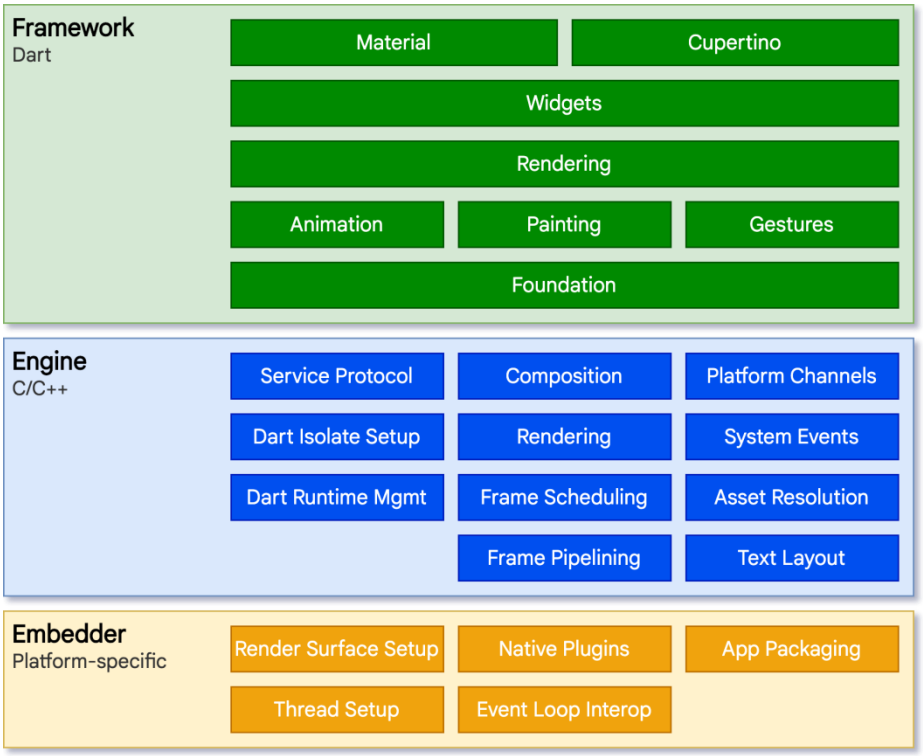
Εικόνα 2.1 Widget κουμπιών σχεδιασμένα σε Material και Cupertino

2.2.3 Αρχιτεκτονική

Κατά τη δημιουργία ενός Flutter έργου, η εφαρμογή εκτελείται παράλληλα σε μία εικονική μηχανή. Ο λόγος που εκτελείται στην εικονική μηχανή είναι η λειτουργία του hot reload. Αυτή η λειτουργία αναπτύχθηκε με σκοπό την άμεση επαναφόρτωση των αλλαγών, χωρίς να απαιτείται μεταγλώττιση και υπάρχει μόνο κατά την ανάπτυξη της εφαρμογής και όχι στην παραγωγική της μορφή. Έτσι, το μέγεθος της εφαρμογής είναι σημαντικά μεγαλύτερο κατά τη διάρκεια που προγραμματίζεται, συγκριτικά με το μέγεθός της όταν κυκλοφορήσει.

Το Flutter σχεδιάστηκε ως ένα πολυεπίπεδο σύστημα. Τα επίπεδα αυτά είναι ο ενσωματωτής (Embedder), η μηχανή (Engine) και το Framework.

Ξεκινώντας από το επίπεδο του ενσωματωτή, το οποίο αποτελεί το χαμηλότερο επίπεδο, το Flutter παρέχει ένα κέλυφος (shell) το οποίο περιέχει την εικονική μηχανή όπου πραγματοποιείται η μεταγλώττιση του κώδικα. Ο ενσωματωτής είναι συγκεκριμένος για κάθε πλατφόρμα (platform-specific) και έχει δημιουργηθεί με βάση τη κατάλληλη γλώσσα προγραμματισμού για κάθε πλατφόρμα ώστε να παρέχει πρόσβαση στο native API της αντίστοιχης πλατφόρμας.



Σχήμα 2.2 Αρχιτεκτονική του Flutter Framework[2]

Το μεσαίο επίπεδο είναι αυτό την μηχανής και είναι γραμμένη κυρίως στη γλώσσα προγραμματισμού C++. Η μηχανή του Flutter αποτελεί ένα μηχανισμό 2D γραφικών και συντονίζει λειτουργίες χαμηλότερου επιπέδου του Flutter API όπως η εικονική μηχανή, τα γραφικά εισόδου και εξόδου, τη διάταξη του κειμένου, τη προσβασιμότητα, την αρχιτεκτονική των πρόσθετων plugin και τον χρόνο εκτέλεσης του κώδικα Dart . Επίσης, η μηχανή Flutter είναι υπεύθυνη για την ραστεροποίηση σκηνών η οποία οδηγεί στην γρήγορη απόδοση της οθόνης.

Το τελευταίο και υψηλότερο επίπεδο είναι το Framework, το οποίο περιεγράφηκε παραπάνω. Μόλις ολοκληρωθεί η διεπαφή χρήστη, το Flutter αποδίνει τα widgets στη μηχανή του. Εκεί μεταγλωττίζονται και εν συνεχεία επιστρέφονται στο Framework [9].

2.2.4 Λόγοι επιλογής του Flutter

Ανακαλύπτοντας το Flutter, γίνονται αντιληπτά τα πλεονεκτήματα που παρουσιάζει. Όπως έχει αναφερθεί παραπάνω, το Flutter αποτελεί ανοιχτού κώδικα cross-platform εργαλείο. Με τον ίδιο πηγαίο κώδικα, δημιουργούνται δύο εφαρμογές για τις πλατφόρμες Android και iOS είναι διαθέσιμο για εφαρμογές ιστού, Windows, macOS και Linux. Η γλώσσα προγραμματισμού που χρησιμοποιεί το Flutter είναι η Dart και αποτελεί μία αντικειμενοστραφής γλώσσα προγραμματισμού, ενώ παρουσιάζει αρκετές ομοιότητες με τη γλώσσα Java. Παράλληλα, το Flutter διαφέρει από άλλα

[2] https://docs.flutter.dev/assets/images/docs/arch-overview/archdiagram.png

frameworks δημιουργίας εφαρμογών. Συγκεκριμένα, το Flutter χρησιμοποιεί τη δική του μηχανή για εμφάνιση στοιχείων στην οθόνη. Αυτό έχει ως αποτέλεσμα την υψηλή αποδοτικότητα της εφαρμογής, η οποία είναι σχεδόν ίση με τις native εφαρμογές.

2.3 Firebase

Το Firebase είναι ένα προϊόν της Google το οποίο παρέχει πολλαπλές τεχνολογίες και υπηρεσίες. Από τις υπηρεσίες που προσφέρει, η εφαρμογή χρησιμοποιεί το Authentication, το Cloud Firestore, το Storage και το Cloud Messaging.

2.3.1 Authentication

Όπως προαναφέρθηκε στο πρώτο κεφάλαιο, η αυθεντικοποίηση των χρηστών αποτελεί μία από τις πιο σημαντικές λειτουργίες σε μία εφαρμογή. Προκειμένου να διασφαλιστεί η ασφάλεια των δεδομένων, είναι απαραίτητο για την εφαρμογή να γνωρίζει τη ταυτότητα του κάθε χρήστη που χρησιμοποιεί την εφαρμογή [10]. Η γνώση της ταυτότητας ενός χρήστη επιτρέπει στην εφαρμογή την ασφαλή αποθήκευση των δεδομένων του.

Το Firebase Authentication προσφέρει αυθεντικοποίηση με πολλούς τρόπους [11]. Οι πιο συνηθισμένοι είναι με email/κωδικό πρόσβασης, Google, AppleId, Facebook, SMS. Ο παρακάτω πίνακας παρουσιάζει όλους του τρόπους με τους οποίους μπορεί να επιτευχθεί η αυθεντικοποίηση του χρήστη από το Firebase. Η παρούσα εφαρμογή πραγματοποιεί την αυθεντικοποίηση των χρηστών με βάση το email και τον κωδικό πρόσβασής τους, επιτυγχάνοντας την ασφαλή αποθήκευση των δεδομένων του χρήστη στο cloud.

Πίνακας 2.1 Μέθοδοι αυθεντικοποίησης του χρήστη

Τρόπος σύνδεσης	Κατάσταση
Email/Password	Ενεργοποιημένο
Τηλέφωνο	Απενεργοποιημένο
Google	Απενεργοποιημένο
Apple	Απενεργοποιημένο
Ανώνυμα	Απενεργοποιημένο
Game Center	Απενεργοποιημένο
Facebook	Απενεργοποιημένο
Github	Απενεργοποιημένο
Yahoo	Απενεργοποιημένο
Play Games	Απενεργοποιημένο
Microsoft	Απενεργοποιημένο
Twitter	Απενεργοποιημένο

Εκτός από την δημιουργία και την αυθεντικοποίηση του χρήστη, το Firebase Authentication προσφέρει επιπλέον λειτουργίες οι οποίες αφορούν τον τελικό χρήστη και είναι απαραίτητες για την υλοποίηση, τη λειτουργικότητα και την αξιοπιστία μίας εφαρμογής [12]. Παρακάτω παρουσιάζονται αυτές οι λειτουργίες οι οποίες θα αναλυθούν αργότερα στο Κεφάλαιο 3:

1) Αποσύνδεση του χρήστη
2) Λήψη πληροφοριών σχετικά με τον συνδεδεμένο χρήστη
3) Ενημέρωση των πληροφοριών του συνδεδεμένου χρήστη
4) Αποστολή email για την επαναφορά του κωδικού πρόσβασης

2.3.2 Cloud Firestore

Το Cloud Firestore είναι μία NoSQL βάση δεδομένων, η οποία συγχρονίζει τα δεδομένα σε πραγματικό χρόνο, παρέχοντας ταυτόχρονα εκτός σύνδεσης υποστήριξη [13]. Επιπροσθέτως, το Cloud Firestore μέσω της αυθεντικοποίησης των χρηστών αλλά και των κανόνων που ορίζονται, προσφέρει, αναμφίβολα, ασφάλεια στην εφαρμογή. Στο παράρτημα Β παρουσιάζεται ο οδηγός της εγκατάστασης του Cloud Firestore.

Σε αντίθεση με μία SQL βάση δεδομένων, τα δεδομένα δεν αποθηκεύονται σε πίνακες και γραμμές αλλά σε έγγραφα (documents), τα οποία οργανώνονται σε συλλογές (collections). Κάθε έγγραφο περιέχει ένα σύνολο ζευγών κλειδιού και τιμής [14]. Όλα τα έγγραφα αποθηκεύονται σε συλλογές και μπορούν να περιέχουν υποσυλλογές και αντικείμενα, στοχεύοντας στην ευκολότερη ιεραρχική οργάνωση των πιο σύνθετων και πολύπλοκων δεδομένων. Επειδή το Cloud Firestore δεν έχει σχήμα, τα δεδομένα μπορούν να αποτελούνται από διαφορετικούς τύπους. Οι υποστηριζόμενοι τύποι δεδομένων είναι: boolean, αριθμός (number), κείμενο (string), geo point, binary blob, timestamp, πίνακες και αντικείμενα (maps) [15].

Πίνακας 2.2 Αποθηκευμένο έγγραφο της συλλογής Users

Πεδίο	Τιμή
age	"14-01-1995"
bloodType	"0+"
email	"nikiforosper@yahoo.gr"
firstName	"John"
gender	"Άνδρας"
height	175
imageUrl	"https://firebasestorage.googleapis.com/v0/b/finddoctor-7e356.appspot.com/o/photos%2FWZ9AlRfkQKYqbDyDaGzSxfj4ssv1%2Fdefault_user.jpg?alt=media&token=d05d0c55-962d-463c-ae3a-8099afa60770”
lastName	"Doe"
phone	"6977777777"
uid	"WZ9AlRfkQKYqbDyDaGzSxfj4ssv1"
weight	70

Με το Cloud Firestore, η εφαρμογή συνδέεται απευθείας με τη βάση δεδομένων στο Firestore, χωρίς ενδιάμεσους διακομιστές ανάμεσα στην εφαρμογή και στο Firestore. Την στιγμή που η βάση δεδομένων δέχεται κάποια αλλαγή στα δεδομένα της, τότε τα δεδομένα μεταφέρονται απευθείας στους χρήστες της εφαρμογής.

Το Cloud Firestore λειτουργεί ακόμα και όταν δεν υπάρχει σύνδεση στο διαδίκτυο. Σε αυτή τη περίπτωση τα δεδομένα δεν λαμβάνονται από τη βάση δεδομένων, αλλά από τη κρυφή μνήμη. Το

Firestore αποθηκεύει ένα αντίγραφο των δεδομένων, στο οποίο η εφαρμογή έχει πρόσβαση. Όταν η συσκευή βρίσκεται εκτός σύνδεσης, ο χρήστης έχει τη δυνατότητα να διαβάσει και να γράψει στα αποθηκευμένα δεδομένα. Όταν η συσκευή επανασυνδεθεί στο διαδίκτυο, το Cloud Firestore συγχρονίζει τυχόν αλλαγές που πραγματοποιήθηκαν όσο η συσκευή βρισκόταν εκτός σύνδεσης.

Για την αποθήκευση δεδομένων στη βάση δεδομένων του Cloud Firestore, υπάρχουν ορισμένοι περιορισμοί, οι οποίοι παρουσιάζονται στον παρακάτω πίνακα. Για την ανάπτυξη της παρούσας εφαρμογής, χρησιμοποιήθηκε το δωρεάν πρόγραμμα του Firestore [16]. Η δωρεάν έκδοση περιέχει περιορισμούς κυρίως για τον καθημερινό αριθμό ανάγνωσης και εγγραφής δεδομένων, όπου περιορίζονται σε 50.000 και 20.000 αντίστοιχα.

Πίνακας 2.3 Δωρεάν όρια του Firestore

Λειτουργία	Ποσοστό
Αποθηκευμένα δεδομένα	1 Gb
Διάβασμα δεδομένων	50.000 / ημέρα
Εγγραφή δεδομένων	20.000 / ημέρα
Διαγραφή δεδομένων	20.000 / ημέρα
Χρήση δικτύου	10 Gb / μήνα

Πίνακας 2.4 Συγκεκριμένα όρια του Firestore στις συλλογές, τα έγγραφα και τα πεδία

Όριο	Λεπτομέρειες
Περιορισμοί στα μοναδικά αναγνωριστικά των συλλογών	1) Πρέπει να είναι έγκυροι χαρακτήρες UTF-8 2) Δεν πρέπει να είναι μεγαλύτερο από 1.500 byte 3) Δεν μπορεί να περιέχει κάθετο προς τα εμπρός (/) 4) Δεν μπορεί να αποτελείται αποκλειστικά από μία μόνο τελεία (.) ή δύο τελείες (..) 5) Δεν μπορεί να είναι της μορφής __.*__
Μέγιστο βάθος των υποσυλλογών	100
Περιορισμοί στα μοναδικά αναγνωριστικά των εγγράφων	1) Πρέπει να είναι έγκυροι χαρακτήρες UTF-8 2) Δεν πρέπει να είναι μεγαλύτερο από 1.500 byte 3) Δεν μπορεί να περιέχει κάθετο προς τα εμπρός (/) 4) Δεν μπορεί να αποτελείται αποκλειστικά από μία μόνο τελεία (.) ή δύο τελείες (..) 5) Δεν μπορεί να είναι της μορφής __.*__ 6) Εάν εισάγετε οντότητες του Datastore σε μια βάση δεδομένων Firestore, τα αναγνωριστικά αριθμητικών οντοτήτων είναι της μορφής: __id[0-9]+__
Μέγιστο μέγεθος του ονόματος ενός εγγράφου	6 Kb
Μέγιστο μέγεθος ενός εγγράφου	1 Mb
Περιορισμοί στα ονόματα των πεδίων	Πρέπει να είναι έγκυροι χαρακτήρες UTF-8
Μέγιστο μέγεθος του ονόματος ενός πεδίου	1.500 Bytes
Περιορισμοί στη διαδρομή ενός πεδίου	1) Πρέπει να διαχωρίζονται τα ονόματα των πεδίων με μία μόνο τελεία (.) 2) Μπορεί να μεταβιβαστεί ως συμβολοσειρά όταν όλα τα ονόματα των πεδίων στη διαδρομή είναι απλά. Διαφορετικά πρέπει να μεταβιβαστούν ως αντικείμενο FieldPath

	3) Περιέχει μόνο τους χαρακτήρες a-z, A-Z, 0-9 και κάτω παύλα (_) 4) Δεν ξεκινά με 0-9
Μέγιστο μέγεθος της διαδρομής ενός πεδίου	1.500 Bytes
Μέγιστο μέγεθος της τιμής ενός πεδίου	1 Mb
Μέγιστο βάθος των πεδίων σε ένα map ή ένα πίνακα δεδομένων	20

2.3.3 Storage

Το Firebase Storage αποτελεί υπηρεσία του Firestore με την οποία καθίστανται δυνατή η αποθήκευση αρχείων στο cloud [17]. Η αποθήκευση των δεδομένων πραγματοποιείται με μεγάλη ασφάλεια καθώς ορίζονται κανόνες για την πρόσβαση στα δεδομένα. Στη παρούσα εφαρμογή, μόνο οι εγγεγραμμένοι χρήστες έχουν τη δυνατότητα να αποθηκεύουν δεδομένα στο Storage, καλύπτοντας έτσι την απαίτηση της ασφάλειας. Οι πιο συνηθισμένοι τύποι αρχείων για αποθήκευση είναι φωτογραφίες και βίντεο, χωρίς όμως να υπάρχει περιορισμός. Ο χρήστης έχει τη δυνατότητα να αποθηκεύσει οποιουδήποτε τύπου αρχεία.

Πίνακας 2.5 Δωρεάν όρια του Firebase Storage [18]

Λειτουργία	Ποσοστό
Μέγεθος αποθηκευμένων δεδομένων	5 Gb
Μέγεθος ληφθέντων δεδομένων	1 Gb / ημέρα
Μεταμόρφωση δεδομένων	20.000 / ημέρα
Ληφθέντα δεδομένα	50.000 / ημέρα

2.3.4 Cloud Messaging

Το Firebase Cloud Messaging αποτελεί μία ακόμη υπηρεσία του Firebase. Πιο συγκεκριμένα, το Firebase Cloud Messaging είναι μία cross-platform υπηρεσία η οποία επιτρέπει την αποστολή ειδοποιήσεων στις συσκευές των χρηστών [19]. Οι ειδοποιήσεις διαχειρίζονται αυτόματα από το Firebase Cloud Messaging SDK προκειμένου να εμφανιστούν στην εφαρμογή. Τα μηνύματα ειδοποιήσεων περιέχουν ένα σύνολο από κλειδιά και τιμές. Παρακάτω δίνεται δύο παραδείγματα που αφορούν τη δομή μηνυμάτων ειδοποίησης με και χωρίς δεδομένα:

Κώδικας 2.2 Μορφή ειδοποίησης στο Firebase Cloud Messaging [20]

```
{
  "message":{
    "token":"bk3RNwTe3H0:CI2k_HHwgIpoDKCIZvvDMExUdFQ3P1",
    "notification":{
      "title":"Έχετε νέο ραντεβού στις 07/01/2022 09:00",
      "body":"Από Περτσέλης Νικηφόρος"
    }
  }
}
```

2.4 Γλώσσα προγραμματισμού Dart

Η γλώσσα Dart αποτελεί μία αντικειμενοστραφή γλώσσα προγραμματισμού [21]. Όπως οι περισσότερες αντικειμενοστραφής γλώσσες προγραμματισμού, η Dart έχει μία πολύ βασική δομή, καθιστώντας την μία γλώσσα προγραμματισμού η οποία είναι εύκολη στην εκμάθησή της, ιδιαίτερα για άτομα με εμπειρία σε γλώσσες προγραμματισμού όπως η Java, η C#, η C++ ή άλλες αντικειμενοστραφής γλώσσες.

Η Dart παρουσιάστηκε το 2011 [22] αλλά κυκλοφόρησε επίσημα το 2013 [23]. Σχεδιάστηκε και αναπτύχθηκε από την Google όπως το Flutter Framework και το Firebase. Αρχικά η Dart έδωσε έμφαση στην ανάπτυξη εφαρμογών ιστού, με σκοπό την αντικατάσταση της Javascript. Πλέον επικεντρώνεται στην ανάπτυξη εφαρμογών για κινητά τηλέφωνα και συγκεκριμένα στο Flutter Framework. Η ομάδα του Flutter, έπρεπε να επιλέξουν κάποια γλώσσα προγραμματισμού η οποία θα χρησιμοποιούταν από το Flutter Framework. Η επικρατέστερη γλώσσα ήταν η Dart. Οι λόγοι που οδήγησαν στη ομάδα να επιλέξει την Dart είναι επειδή θεώρησαν ότι εξοικονομούν πόρους, λόγω της δυνατότητας της γλώσσας να δημιουργεί εφαρμογές σε Android και iOS. Επίσης, η αντικειμενοστρέφεια της γλώσσας αλλά και η υψηλή απόδοσή της, διαδραματίζουν καθοριστικό ρόλο στην επιλογή της συγκεκριμένης γλώσσας προγραμματισμού [24].

2.5 Visual Studio Code

Το Visual Studio Code είναι ένα δωρεάν και ανοιχτού κώδικα πρόγραμμα το οποίο δημιουργήθηκε από τη Microsoft για την ανάπτυξη, την επεξεργασία και την εκτέλεση πηγαίου κώδικα [25]. Το περιβάλλον αυτό, προσφέρει πολλές δυνατότητες. Μία από τις βασικές δυνατότητες του περιβάλλοντος είναι ο εντοπισμός σφαλμάτων (debugging), κατά τον οποίο μπορεί κανείς να εξετάσει τις κλήσεις ή τις μεταβλητές κατά τον χρόνο εκτέλεσης και να θέσει σε παύση την εκτέλεση του κώδικα.

Το Visual Studio Code υποστηρίζει λειτουργίες όπως είναι η επισήμανση της σύνταξης του κώδικα, η αυτόματη συμπλήρωση κώδικα και ο αυτόματος εντοπισμός SDK από τη μεταβλητή περιβάλλοντος PATH. Επίσης προσφέρει ποικιλία επεκτάσεων με τις οποίες καθίστανται πιο ευέλικτος και γρήγορος ο προγραμματισμός. Παράλληλα ενσωματώνει τερματικό για την εκτέλεση γραμμών εντολών. Στο παράρτημα Γ παρουσιάζεται ο οδηγός της εγκατάστασης του Visual Studio Code.

2.6 Επίλογος

Στο κεφάλαιο αυτό παρουσιάστηκαν και αναλύθηκαν οι τεχνολογίες και τα εργαλεία που χρησιμοποιήθηκαν για την δημιουργία της τρέχουσας εφαρμογής. Συγκεκριμένα, χρησιμοποιήθηκαν τεχνολογίες οι οποίες έχουν αναπτυχθεί από τη Google, προσφέροντας απόλυτη συμβατότητα μεταξύ τους. Στο επόμενο κεφάλαιο περιγράφεται ο τρόπος με τον οποίο χρησιμοποιήθηκαν οι τεχνολογίες που παρουσιάστηκαν σε αυτό το κεφάλαιο.

Κεφάλαιο 3ο: Υλοποίηση της εφαρμογής

3.1 Εισαγωγή

Στο κεφάλαιο αυτό περιγράφεται και αναλύεται ο τρόπος με τον οποίο χρησιμοποιήθηκαν οι τεχνολογίες για την ανάπτυξη της εφαρμογής. Συγκεκριμένα δίνεται πλήρης περιγραφή της δομής, του σχεδιασμού και του τρόπου υλοποίησης των τεχνολογιών που χρησιμοποιεί η εφαρμογή.

3.2 Μοντελοποίηση κλάσεων

Κατά την ανάπτυξη της εφαρμογής δημιουργήθηκαν εννέα βασικές κλάσεις:

3.2.1 Κλάση UserProfile

Η κλάση αυτή περιγράφει τα στοιχεία του χρήστη ο οποίος έχει εγγραφεί ως ασθενής. Τα στοιχεία αυτά είναι τα στοιχεία τα οποία έχει εισάγει ο χρήστης κατά την εγγραφή του στην εφαρμογή και είναι τα εξής:

Πίνακας 3.1 Κλάση UserProfile

Όνομα	Τύπος	Περιγραφή
uid	Συμβολοσειρά (String)	Το αναγνωριστικό κάθε χρήστη. Το πεδίο αυτό είναι μοναδικό για κάθε χρήστη και αποτελεί τη ταυτότητα του χρήστη για την εφαρμογή
email	Συμβολοσειρά (String)	Το email του χρήστη
firstName	Συμβολοσειρά (String)	Το όνομα του χρήστη
lastName	Συμβολοσειρά (String)	Το επώνυμο του χρήστη.
phone	Συμβολοσειρά (String)	Το τηλέφωνο του χρήστη.
age	Ακέραιος αριθμός (Integer)	Η ηλικία του χρήστη.
bloodType	Συμβολοσειρά (String)	Η ομάδα αίματος του χρήστη
imageUrl	Συμβολοσειρά (String)	Η διεύθυνση στην οποία είναι αποθηκευμένη η φωτογραφία του χρήστη στο Firebase Storage. Αν ο χρήστης δεν προσθέσει κάποια φωτογραφία, τότε το πεδίο αυτό παραμένει κενό. (προαιρετικό)
height	Ακέραιος αριθμός (Integer)	Το ύψος του χρήστη.
weight	Ακέραιος αριθμός (Integer)	Το βάρος του χρήστη.
gender	Συμβολοσειρά (String)	Το φύλο του χρήστη.

3.2.2 Κλάση DoctorProfile

Η κλάση αυτή περιγράφει τα στοιχεία του χρήστη ο οποίος έχει εγγραφεί ως ιατρός. Τα στοιχεία αυτά είναι τα στοιχεία τα οποία έχει εισάγει ο χρήστης κατά την εγγραφή του στην εφαρμογή και είναι τα εξής:

Πίνακας 3.2 Κλάση DoctorProfile

Όνομα	Τύπος	Περιγραφή
dId	Συμβολοσειρά (String)	Το αναγνωριστικό κάθε χρήστη-ιατρού. Το πεδίο αυτό είναι μοναδικό για κάθε χρήστη και αποτελεί τη ταυτότητα του χρήστη για την εφαρμογή
email	Συμβολοσειρά (String)	Το email του ιατρού.
firstName	Συμβολοσειρά (String)	Το όνομα του ιατρού.

lastName	Συμβολοσειρά (String)	Το επώνυμο του ιατρού.
address	Συμβολοσειρά (String)	Η διεύθυνση του ιατρού.
description	Συμβολοσειρά (String)	Η περιγραφή που ορίζει ο ιατρός, περιγράφοντας με λίγα λόγια στοιχεία της προσωπικότητάς του.
phone	Συμβολοσειρά (String)	Το τηλέφωνο του ιατρού.
workTime	Συμβολοσειρά (String)	Το καθημερινό ωράριο εργασίας του ιατρού.
specialty	Συμβολοσειρά (String)	Η ειδικότητα του ιατρού.
imageUrl	Συμβολοσειρά (String)	Η διεύθυνση στην οποία είναι αποθηκευμένη η φωτογραφία του ιατρού στο Firebase Storage. (προαιρετικό)
avgRating	Ακέραιος αριθμός (Integer)	Ο μέσος όρος των αξιολογήσεων που έχει λάβει ο ιατρός.
numRatings	Ακέραιος αριθμός (Integer)	Ο συνολικός αριθμός των αξιολογήσεων που έχει λάβει ο ιατρός.
lat	Συμβολοσειρά (String)	Το γεωγραφικό πλάτος του ιατρείου.
lng	Συμβολοσειρά (String)	Το γεωγραφικό μήκος του ιατρείου.

3.2.3 Κλάση Post

Η κλάση αυτή αναφέρεται στη δημοσίευση περιεχομένου που μπορεί να κάνουν όλοι οι χρήστες. Τα πεδία της κλάσης είναι τα εξής:

Πίνακας 3.3 Κλάση Post

Όνομα	Τύπος	Περιγραφή
uid	Συμβολοσειρά (String)	Το αναγνωριστικό του χρήστη
timestamp	Ακέραιος αριθμός (Integer)	Η χρονική στιγμή της δημιουργίας της δημοσίευσης.
post	Συμβολοσειρά (String)	Το κείμενο της δημοσίευσης.
imageUrl	Συμβολοσειρά (String)	Η διεύθυνση της επισυναπτόμενης φωτογραφίας της δημοσίευσης. (προαιρετικό)
likedBy	Λίστα τύπου συμβολοσειράς (Array of strings)	Λίστα από τους χρήστες στους οποίους αρέσει η αναρτημένη δημοσίευση. Η λίστα περιλαμβάνει ένα σύνολο από τα αναγνωριστικά (uid/dId) αυτών των χρηστών.

3.2.4 Κλάση Comment

Η κλάση αυτή αναφέρεται στο σχόλιο που έχει λάβει μία δημοσίευση. Τα πεδία της κλάσης είναι τα εξής:

Πίνακας 3.4 Κλάση Comment

Όνομα	Τύπος	Περιγραφή
uid	Συμβολοσειρά (String)	Το αναγνωριστικό του χρήστη ο οποίος δημιούργησε ένα σχόλιο για μία δημοσίευση.
timestamp	Ακέραιος αριθμός (Integer)	Η χρονική στιγμή της δημιουργίας του σχολίου.
commentText	Συμβολοσειρά (String)	Το κείμενο-σχόλιο που έχει γράψει ο χρήστης για μία δημοσίευση.

3.2.5 Κλάση Favorites

Η κλάση αυτή αναφέρεται στη λίστα των αγαπημένων ιατρών για κάθε χρήστη. Τα πεδία της κλάσης είναι τα εξής:

Πίνακας 3.5 Κλάση Favorites

Όνομα	Τύπος	Περιγραφή
doctorId	Συμβολοσειρά (String)	Το αναγνωριστικό του ιατρού τον οποίο ο χρήστης επιθυμεί να προσθέσει στα αγαπημένα του.
isFavorite	Boolean	Το πεδίο αυτό δείχνει εάν ο ιατρός ανήκει στη λίστα αγαπημένων του χρήστη.

3.2.6 Κλάση Review

Η κλάση αυτή αναφέρεται στην αξιολόγηση γράφει ένας χρήστης σε κάποιον ιατρό. Τα πεδία της κλάσης είναι τα εξής:

Πίνακας 3.6 Κλάση Review

Όνομα	Τύπος	Περιγραφή
id	Συμβολοσειρά (String)	Το μοναδικό αναγνωριστικό της αξιολόγησης.
did	Ακέραιος αριθμός (Integer)	Το μοναδικό αναγνωριστικό του γιατρού για τον οποίο πραγματοποιείται η αξιολόγηση
uid	Συμβολοσειρά (String)	Το μοναδικό αναγνωριστικό του χρήστη ο οποίος αξιολογεί έναν γιατρό.
rating	Ακέραιος αριθμός (Integer)	Ο βαθμός αξιολόγησης του γιατρού. Οι διαθέσιμες τιμές για την αξιολόγηση είναι από μηδέν έως πέντε (0-5)
text	Συμβολοσειρά (String)	Το κείμενο της αξιολόγησης (προαιρετικό)
timestamp	Ημερομηνία (Timestamp)	Η ημερομηνία κατά την οποία δημιουργήθηκε η αξιολόγηση.

3.2.7 Κλάση UserAppointment

Η κλάση αυτή αναφέρεται στο ραντεβού που κλείνει ο χρήστης. Τα πεδία της κλάσης είναι τα εξής:

Πίνακας 3.7 Κλάση UserAppointment

Όνομα	Τύπος	Περιγραφή
doctorId	Συμβολοσειρά (String)	Το αναγνωριστικό του ιατρού στον οποίο επιθυμεί να κλείσει ραντεβού ο χρήστης.
date	Συμβολοσειρά (String)	Η ημερομηνία και η ώρα του ραντεβού που επιθυμεί να κλείσει ο χρήστης.

3.2.8 Κλάση Patient

Η κλάση αναφέρεται στον χρήστη που κλείνει ραντεβού σε ένα ιατρό. Όταν ένας χρήστης κλείσει ραντεβού σε ένα ιατρό, μερικά από τα στοιχεία του χρήστη είναι διαθέσιμα στον ιατρό. Τα πεδία της κλάσης είναι τα εξής:

Πίνακας 3.8 Κλάση Patient

Όνομα	Τύπος	Περιγραφή
uid	Συμβολοσειρά (String)	Το αναγνωριστικό του χρήστη που έκλεισε ραντεβού στον ιατρό.
email	Συμβολοσειρά (String)	Το email του χρήστη
firstName	Συμβολοσειρά (String)	Το όνομα του χρήστη.
lastName	Συμβολοσειρά (String)	Το επώνυμο του χρήστη.
phone	Συμβολοσειρά (String)	Το τηλέφωνο του χρήστη.

3.2.9 Κλάση PushNotificationMessage

Η κλάση αυτή αναφέρεται στις ειδοποιήσεις που λαμβάνει ο ιατρός όταν έχει ένα νέο ραντεβού. Τα πεδία της κλάσης είναι τα εξής:

Πίνακας 3.9 Κλάση PushNotificationMessage

Όνομα	Τύπος	Περιγραφή
title	Συμβολοσειρά (String)	Ο τίτλος της ειδοποίησης.
body	Συμβολοσειρά (String)	Το κύριο μέρος της ειδοποίησης.

3.3 Βάση Δεδομένων

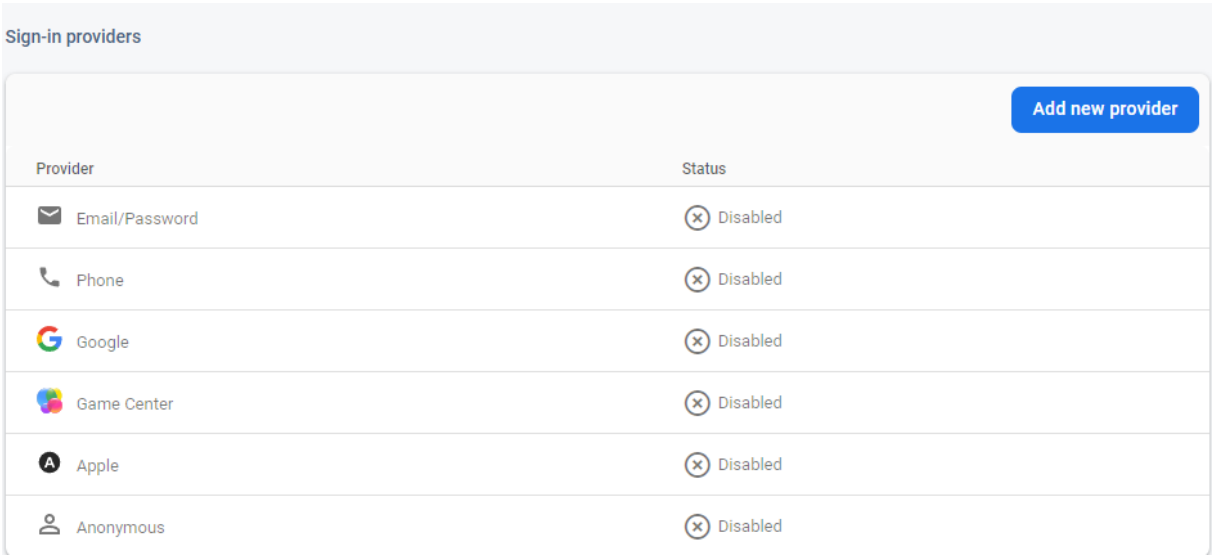
Όπως έχει αναφερθεί σε προηγούμενα κεφάλαια, ως βάση δεδομένων χρησιμοποιείται η υπηρεσία Firestore του Firebase. Το Firestore αποτελεί μία NoSQL βάση δεδομένων που φιλοξενείται στο cloud για αποθήκευση και συγχρονισμό δεδομένων.

3.3.1 Authentication

Όπως αναφέρθηκε στο Κεφάλαιο 2, το Firebase Authentication, περιέχει αρκετούς τρόπους για την σύνδεση του χρήστη. Η εφαρμογή χρησιμοποιεί την αυθεντικοποίηση μέσω της εισαγωγής του προσωπικού email και του κωδικού πρόσβασης του χρήστη.

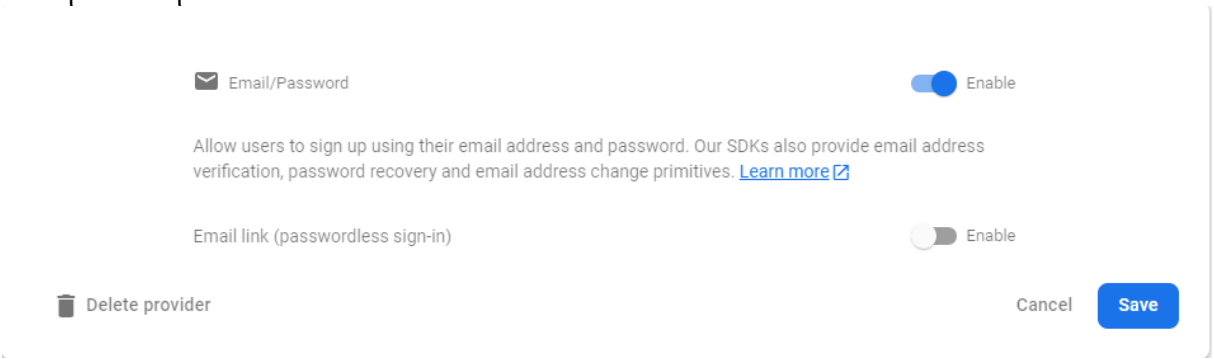
Για να μπορεί η εφαρμογή να πραγματοποιήσει με επιτυχία την αυθεντικοποίηση, χρειάζεται να ενεργοποιηθεί ο τρόπος με τον οποίο θα επιτευχθεί αυτό. Παρακάτω παρουσιάζονται τα βασικά βήματα για την ενεργοποίηση της αυθεντικοποίησης:

1) Χρησιμοποιείται η κονσόλα του Firebase και συγκεκριμένα η σελίδα «Authentication».
2) Από τις διαθέσιμες καρτέλες, επιλέγεται η καρτέλα «Sign-in method». Στη σελίδα αυτή, παρουσιάζονται μερικοί από τους βασικούς τρόπους αυθεντικοποίησης του χρήστη.



Εικόνα 3.1 Προβολή των πιθανών τρόπων αυθεντικοποίησης του χρήστη

3) Από τη λίστα, επιλέγεται ο τρόπος αυθεντικοποίησης. Στη περίπτωση της εφαρμογής, επιλέγεται η σύνδεση Email/Password



Εικόνα 3.2 Ενεργοποίηση της αυθεντικοποίησης του χρήστη

4) Ενεργοποιείται ο τρόπος αυθεντικοποίησης

Από τη στιγμή που έχει ενεργοποιηθεί τουλάχιστον ένας τρόπος σύνδεσης, τότε η εφαρμογή είναι έτοιμη για την εγγραφή και την σύνδεση χρηστών.

Προκειμένου να καλύπτονται οι απαιτήσεις της εφαρμογής, προκειμένου ο χρήστης να έχει πρόσβαση στην εφαρμογή, είναι απαραίτητο να εγγραφεί αρχικά στο σύστημα. Ο χρήστης εγγράφεται στην εφαρμογή, εισάγοντας το προσωπικό του email και τον κωδικό πρόσβασης. Η παρακάτω μέθοδος αποτελεί τμήμα του πηγαίου κώδικα της εφαρμογής και χρησιμοποιείται για την δημιουργία του χρήστη στο σύστημα. Η μέθοδος *signUp()* περιέχει δύο παραμέτρους, το email και τον κωδικό πρόσβασης του χρήστη και εκτελεί τη μέθοδο *createUserWithEmailAndPassword() του Firebase* . Αν η εγγραφή του χρήστη είναι επιτυχημένη, τότε η μέθοδος επιστρέφει τις πληροφορίες που αφορούν τον εγγεγραμμένου, πλέον, χρήστη. Σε περίπτωση αποτυχίας, η μέθοδος επιστρέφει το μήνυμα του σφάλματος [26].

Κώδικας 3.1 Μέθοδος signUp η οποία χρησιμοποιείται για την δημιουργία και την εγγραφή του χρήστη στο σύστημα

```
Future<dynamic> signUp(String email, String password) async {
  try {
    AuthResult result = await FirebaseAuth.instance
        .createUserWithEmailAndPassword(email: email, password:
password);

    FirebaseUser user = result.user;
    return user;
  } catch (error) {
    return error.message;
  }
}
```

Πίνακας 3.10 Σφάλματα που μπορεί να προκύψουν κατά την εγγραφή του χρήστη στο σύστημα

Κωδικός σφάλματος	Περιγραφή
email-already-exists	Εμφανίζεται όταν το email που καταχωρεί ο χρήστης, χρησιμοποιείται από κάποιον άλλο χρήστη
invalid-email	Εμφανίζεται όταν δεν είναι έγκυρο το email που καταχωρεί ο χρήστης
operation-not-allowed	Εμφανίζεται όταν δεν είναι ενεργοποιημένο το email ή ο κωδικός πρόσβασης που καταχωρεί ο χρήστης. Αυτό συμβαίνει όταν δεν υπάρχει ενεργοποιημένος τρόπος σύνδεσης στο Firebase
invalid-password	Εμφανίζεται όταν δεν είναι έγκυρος ο κωδικός πρόσβασης που καταχωρεί ο χρήστης. Ο κωδικός πρόσβασης θα πρέπει να είναι μεγαλύτερος από 6 χαρακτήρες

Από τη στιγμή που ο χρήστης έχει εγγραφεί με επιτυχία στο σύστημα, το επόμενο βήμα είναι να συνδεθεί στην εφαρμογή, ώστε να είναι προσβάσιμες όλες οι λειτουργίες και πληροφορίες της εφαρμογής. Παρακάτω παρουσιάζεται η μέθοδος *signIn()* η οποία περιέχει δύο παραμέτρους, το email και τον κωδικό πρόσβασης του χρήστη. Η μέθοδος αυτή εκτελεί τη μέθοδο *signInWithEmailAndPassword* του Firebase και χρησιμοποιείται για την σύνδεση του χρήστη στο σύστημα. Αν η σύνδεση του χρήστη στο σύστημα είναι επιτυχής, τότε η μέθοδος επιστρέφει τις πληροφορίες του χρήστη. Αν αποτύχει η σύνδεση του χρήστη, τότε η μέθοδος επιστρέφει το αντίστοιχο μήνυμα σφάλματος [26].

Κώδικας 3.2 Μέθοδος signIn η οποία χρησιμοποιείται για την σύνδεση του χρήστη στο σύστημα

```
Future<dynamic> signIn(String email, String password) async {
  try {
    AuthResult  result = await FirebaseAuth.instance
        .signInWithEmailAndPassword(email: email, password:
password);

    FirebaseUser  user = result.user;
    return user;
  } catch (error) {
    return error.message;
  }
}
```

Πίνακας 3.11 Σφάλματα που μπορεί να προκύψουν κατά την σύνδεση του χρήστη στο σύστημα

Κωδικός σφάλματος	Περιγραφή
invalid-email	Εμφανίζεται όταν δεν είναι έγκυρο το email που καταχωρεί ο χρήστης
user-disabled	Εμφανίζεται όταν ο χρήστης έχει απενεργοποιηθεί από τον διαχειριστή
user-not-found	Εμφανίζεται όταν δεν υπάρχει κάποιος χρήστης ο οποίος να αντιστοιχίζεται με το email που έχει καταχωρήσει ο χρήστης
wrong-password	Εμφανίζεται όταν ο κωδικός πρόσβασης είναι λανθασμένος για το email που έχει καταχωρήσει ο χρήστης

Όπως αναφέρθηκε στο Κεφάλαιο 2, το Firebase Authentication, εκτός από την εγγραφή και τη σύνδεση ενός χρήστη στην εφαρμογή, παρέχει πολλές και χρήσιμες λειτουργίες, οι οποίες βελτιστοποιούν την εφαρμογή και την εμπειρία του τελικού χρήστη [12]. Οι λειτουργίες αυτές είναι οι εξής:

- Αποσύνδεση του χρήστη
  Η εφαρμογή θα πρέπει να παρέχει στον χρήστη την δυνατότητα να αποσυνδεθεί οποιαδήποτε στιγμή από την εφαρμογή. Η μέθοδος *signOut()* εκτελεί την μέθοδο *signOut()* του Firebase και χρησιμοποιείται για την αποσύνδεση του χρήστη από την εφαρμογή.

Κώδικας 3.3 Μέθοδος signOut η οποία χρησιμοποιείται για την αποσύνδεση του χρήστη από το σύστημα

```
Future<void> signOut() async {
    try {
      return await FirebaseAuth.instance.signOut();
    } catch (error) {
      throw error;
    }
}
```

- Λήψη πληροφοριών σχετικά με τον συνδεδεμένο χρήστη

  Η εφαρμογή χρειάζεται να έχει πρόσβαση σε κάποιες πληροφορίες του χρήστη όπως το μοναδικό αναγνωριστικό του, ή το email του. Η μέθοδος *getFirebaseUser()* χρησιμοποιείται για να έχει πρόσβαση στο Firebase προφίλ του συνδεδεμένου χρήστη. Οι πληροφορίες του χρήστη που λαμβάνονται είναι οι εξής:
  - uid: Το μοναδικό αναγνωριστικό του χρήστη
  - email: Το email του χρήστη
  - displayName: Το όνομα του χρήστη
  - emailVerified: Επιστρέφει εάν η διεύθυνση email του χρήστη έχει επαληθευτεί.
  - metadata: Μεταδεδομένα τα οποία περιλαμβάνουν επιπλέον πληροφορίες
    - creationTime: Ημερομηνία την οποία εγγράφηκε ο χρήστης στο σύστημα
    - lastSignInTime: Η τελευταία ημερομηνία κατά την οποία συνδέθηκε στο σύστημα ο χρήστης
  - phoneNumber: Ο αριθμός του τηλεφώνου του χρήστη
  - photoURL: Ο σύνδεσμος της φωτογραφίας του χρήστη

Κώδικας 3.4 Λήψη του προφίλ του Firebase χρήστη

```
Future<User> getFirebaseUser() async {
  try {
    final user = await FirebaseAuth.instance.currentUser();
    return user;
  } catch (e) {
    return null;
  }
}
```

Παρακάτω παρουσιάζεται ο τρόπος με τον οποίο μπορούν να ληφθούν οι παραπάνω πληροφορίες.

Κώδικας 3.5 Οι πληροφορίες του προφίλ του Firebase χρήστη

```
final user = await getFirebaseUser();
String name = user.displayName;
String email = user.email;
String photoUrl = user.photoUrl;
String uid = user.uid;
```

- Ενημέρωση των πληροφοριών του συνδεδεμένου χρήστη

Από τις πληροφορίες του Firebase χρήστη, η εφαρμογή δύναται να επεξεργαστεί μόνο τον κωδικό πρόσβασης του χρήστη. Ο χρήστης, μπορεί εφόσον είναι συνδεδεμένος, να αλλάξει τον προσωπικό του κωδικό πρόσβασης. Η μέθοδος *updatePassword()* διαβάζει τον νέο κωδικό από το πεδίο κειμένου και προσπαθεί να αλλάξει τον κωδικό πρόσβασης. Εάν ο κωδικός αλλάξει με επιτυχία εμφανίζεται μήνυμα επιτυχίας αλλιώς εμφανίζεται αντίστοιχο μήνυμα σφάλματος [26].

Κώδικας 3.6 Αλλαγή του κωδικού πρόσβασης του χρήστη

```
Future<void> updatePassword() async {
  try {
    final user = await FirebaseAuth.instance.currentUser();
    await user.updatePassword(newPasswordController.text.trim());
  } catch (error) {
    throw error;
  }
}
```

Πίνακας 3.12 Σφάλματα που μπορούν να προκύψουν από την αλλαγή του κωδικού πρόσβασης

Κωδικός σφάλματος	Περιγραφή
invalid-password	Εμφανίζεται όταν δεν είναι έγκυρος ο κωδικός πρόσβασης που καταχωρεί ο χρήστης. Ο κωδικός πρόσβασης θα πρέπει να είναι μεγαλύτερος από 6 χαρακτήρες
requires-recent-login	Εμφανίζεται όταν ο χρόνος της τελευταίας σύνδεσης του χρήστη δεν πληροί το όριο ασφαλείας

- Αποστολή email για την επαναφορά του κωδικού πρόσβασης

Κάθε εφαρμογή η οποία διαχειρίζεται χρήστες χρειάζεται να προσφέρει τη δυνατότητα της επαναφοράς του κωδικού πρόσβασης, όταν κάποιος χρήστης δεν μπορεί να συνδεθεί στο σύστημα. Η μέθοδος *resetPassword()* περιέχει ως μοναδική παράμετρο το email του χρήστη που θέλει να ανακτήσει τον προσωπικό του κωδικό πρόσβασης. Μέσω αυτής της μεθόδου, εκτελείται η μέθοδος *sendPasswordResetEmail()* του Firebase και χρησιμοποιείται για την ανάκτηση του κωδικού πρόσβασης. Αν εκτελεστεί σωστά η μέθοδος, τότε θα σταλεί ένα email στη διεύθυνση email του χρήστη. Το email περιλαμβάνει οδηγίες για την επαναφορά του κωδικού πρόσβασης. Αν αποτύχει η επαναφορά του κωδικού, τότε εμφανίζεται μήνυμα σφάλματος.

Κώδικας 3.7 Επαναφορά του κωδικού πρόσβασης του χρήστη

```
Future<void> resetPassword(String email) async {
  try {
    await FirebaseAuth.instance.sendPasswordResetEmail(email:
email);
  } catch (error) {
    throw error;
  }
}
```

Πίνακας 3.13 Σφάλματα που μπορούν να προκύψουν κατά την ανάκτηση του κωδικού πρόσβασης

Κωδικός σφάλματος	Περιγραφή
invalid-email	Εμφανίζεται όταν δεν είναι έγκυρο το email που καταχωρεί ο χρήστης
user-not-found	Εμφανίζεται όταν δεν υπάρχει κάποιος χρήστης ο οποίος να αντιστοιχίζεται με το email που έχει καταχωρήσει ο χρήστης

3.3.2 Δομή της βάσης δεδομένων

Όπως έχει αναφερθεί, το Cloud Firestore είναι μία NoSQL βάση δεδομένων όπου τα δεδομένα αποθηκεύονται σε συλλογές (collections), οι οποίες περιέχουν έγγραφα ή/και υποσυλλογές. Στη παρούσα εφαρμογή, η βάση δεδομένων περιέχει πέντε βασικές συλλογές. Οι συλλογές αυτές είναι η συλλογή doctors, posts, specialties, tokens και users.

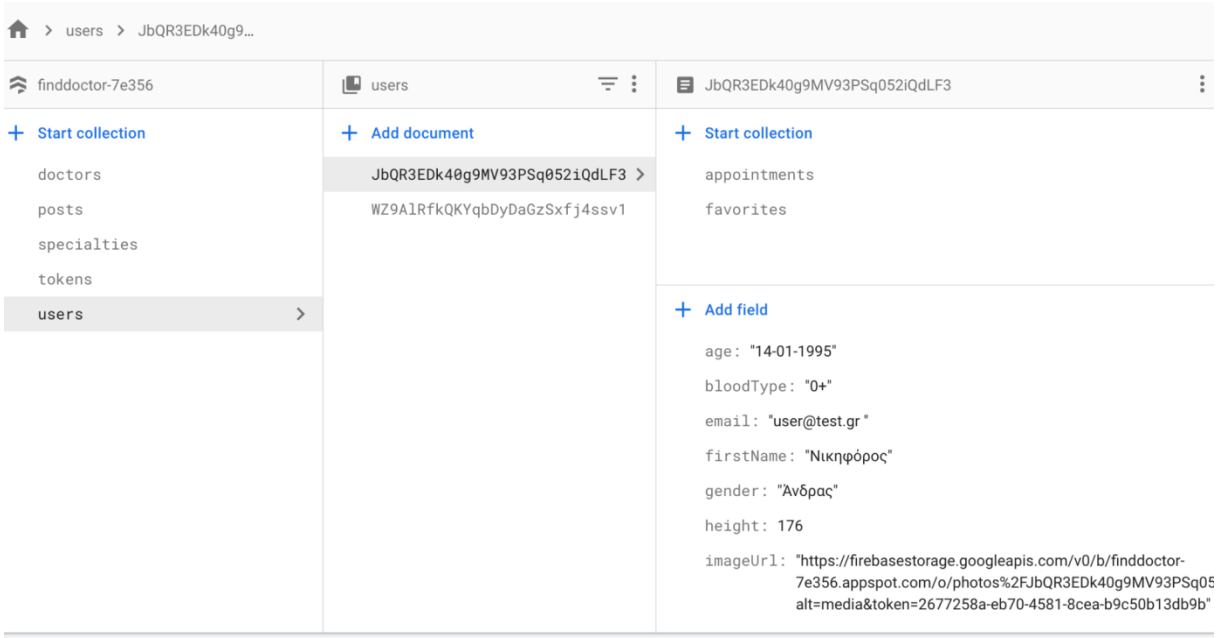
Ο χρήστης κατά της είσοδό του στην εφαρμογή μπορεί να εγγραφεί στο σύστημα είτε ως ασθενής είτε ως ιατρός. Κατά την εγγραφή του χρήστη ως ασθενής, δημιουργείται η συλλογή users. Τη στιγμή που ο χρήστης συμπληρώσει και διαμορφώσει το προφίλ του, τότε καταχωρούνται τα στοιχεία του στη συλλογή users.

Η εγγραφή των δεδομένων του χρήστη στη βάση δεδομένων πραγματοποιείται με την μέθοδο *createUserProfile()*. Η μέθοδος λαμβάνει ως παράμετρο το προφίλ του χρήστη, σύμφωνα με τα στοιχεία που έχει συμπληρώσει.

Κώδικας 3.8 Μέθοδος δημιουργίας της συλλογής users

```
Future<void> createUserProfile(UserProfile userProfile) async {
  FirebaseUser currentUser = await AuthService().getCurrentUser();
  await Firestore.instance
      .collection("users")
      .document(currentUser.uid)
      .setData(userProfile.toMap());}
```

Η συλλογή περιέχει έγγραφα τα οποία περιέχουν ως κλειδί το id του χρήστη, το οποίο παρέχεται αυτόματα από το Firebase και είναι μοναδικό για κάθε χρήστη. Η τιμή του εγγράφου που αντιστοιχεί σε κάθε χρήστη, περιέχει τα προσωπικά του στοιχεία, τα οποία προστέθηκαν κατά την εγγραφή του. Τα πεδία αυτά αντιστοιχούν με τα πεδία της κλάσης UserProfile, η οποία παρουσιάστηκε παραπάνω. Επιπροσθέτως, το έγγραφο περιέχει δύο υποσυλλογές, την συλλογή appointments και την favorites. Οι υποσυλλογές αυτές, αρχικά δεν υφίστανται αλλά δημιουργούνται μετά από συγκεκριμένες ενέργειες του χρήστη. Παρακάτω παρουσιάζεται το στιγμιότυπο της συλλογής users όπου διακρίνονται τα έγγραφα αλλά και οι υποσυλλογές appointments και favorites.



Εικόνα 3.3 Συλλογή Users

Η συλλογή appointments βρίσκεται μέσα στη συλλογή “users” και δημιουργείται όταν ένα ασθενής δημιουργήσει ένα ραντεβού σε κάποιον ιατρό. Για την δημιουργία του ραντεβού, χρησιμοποιείται η μέθοδος *addToUserAppointments()*, η οποία λαμβάνει ως παραμέτρους το μοναδικό αναγνωριστικό του επιλεγμένου γιατρού και την ημερομηνία. Η μέθοδος αυτή, χρησιμοποιεί την λειτουργία του transaction για την εγγραφή του ραντεβού στη βάση δεδομένων. Η λειτουργία transaction, χρησιμοποιείται για τη σωστή αποθήκευση των δεδομένων, σε περίπτωση που υπάρχουν ταυτόχρονες αλλαγές στα δεδομένα. Αν τροποποιηθούν κάποια δεδομένα που αφορούν το transaction, τότε το transaction θα εκτελεστεί ξανά, λαμβάνοντας υπόψιν τις νέες τιμές των δεδομένων. Στη περίπτωση της δημιουργίας ενός ραντεβού, υπάρχει η πιθανότητα, δύο ή περισσότεροι χρήστες να προσπαθήσουν ακριβώς την ίδια χρονική στιγμή, να κλείσουν ραντεβού για τον ίδιο γιατρό για την ίδια ημερομηνία.

Κώδικας 3.9 Μέθοδος δημιουργίας της συλλογής appointments για κάθε χρήστη

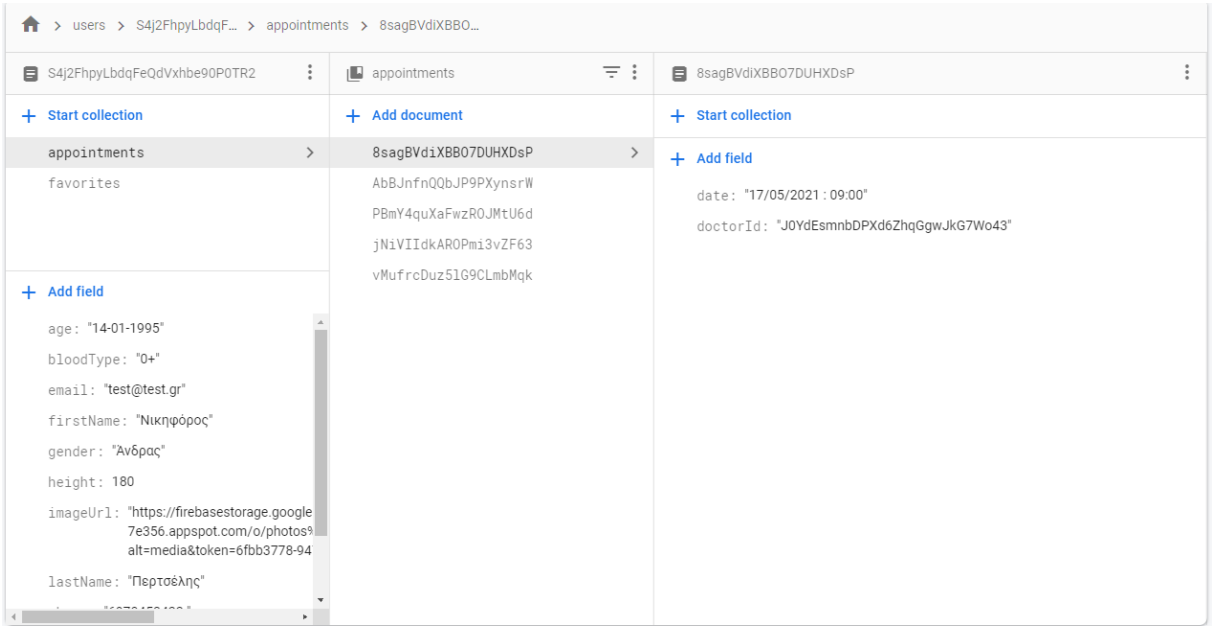
```
Future<void> addToUserAppointments(String doctorId, String date)
async {
  String userId = await FirebaseAuth.getUserId();

  DocumentReference user =
      Firestore.instance.collection('users').document(userId);

  DocumentReference newAppointment =
```

```
        user.collection('appointments').document();
    return Firestore.instance.runTransaction((transaction) {
      return transaction.set(newAppointment, {
        'doctorId': doctorId,
        'date': date,
      });
    });
}
```

Η συλλογή “appointments” περιέχει έγγραφα τα οποία αντιστοιχούν σε ένα ραντεβού του χρήστη. Κάθε έγγραφο αποτελείται από ένα μοναδικό κλειδί, το οποίο παράγεται τυχαία από το Firebase και περιλαμβάνει τα δεδομένα που αφορούν το συγκεκριμένο ραντεβού. Πιο συγκεκριμένα, το κάθε έγγραφο περιέχει την ημερομηνία (date) του ραντεβού και το μοναδικό αναγνωριστικό του γιατρού (doctorId).



Εικόνα 3.4 Υποσυλλογή Appointments

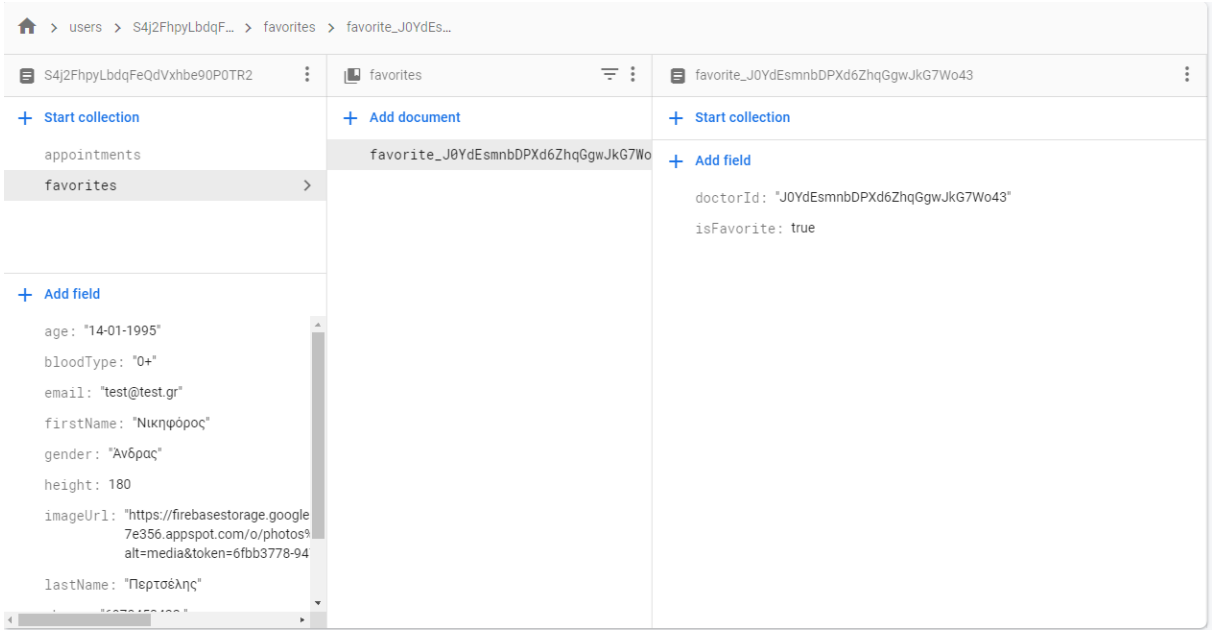
Ομοίως με τη συλλογή appointments, η συλλογή favorites βρίσκεται και αυτή μέσα στη συλλογή “users” και δημιουργείται όταν ένας ασθενής προσθέσει έναν ιατρό στα αγαπημένα του. Για την προσθήκη ενός γιατρού στα αγαπημένα του χρήστη, χρησιμοποιείται η μέθοδος *addToFavorites()*, η οποία λαμβάνει παραμέτρους το μοναδικό αναγνωριστικό του γιατρού (doctorId) και μία δυαδική μεταβλητή, με δύο πιθανές τιμές true ή false, οι οποίες υποδηλώνουν εάν ο ιατρός έχει τοποθετηθεί στα αγαπημένα του χρήστη.

Κώδικας 3.10 Μέθοδος δημιουργίας της συλλογής favorites για κάθε χρήστη

```
Future<void> addToFavorites({String doctorId, bool isFavorite})
async {
  String userId = await Firestore.instance.getUserId();
  DocumentReference user =
      Firestore.instance.collection('users').document(userId);
  DocumentReference newFavoriteDocument =
      user.collection('favorites').document("favorite_$doctorId");
  await newFavoriteDocument.setData({
```

```
      'doctorId': doctorId,
      'isFavorite': isFavorite,
    });
}
```

Η συλλογή περιέχει έγγραφα τα οποία περιέχουν ένα μοναδικό κλειδί και δεδομένα. Το κλειδί αντιστοιχεί στο μοναδικό αναγνωριστικό του ιατρού και είναι της μορφής «favorite_*DOCTOR_ID*». Το έγγραφο περιέχει τα πεδία με τις τιμές που χρησιμοποιεί η μέθοδος *addToFavorites()*, δηλαδή το μοναδικό αναγνωριστικό του ιατρού και τη δυαδική μεταβλητή.



Εικόνα 3.5 Υποσυλλογή Favorites

Στη περίπτωση που ο χρήστης εγγραφεί στο σύστημα και συμπληρώσει τα στοιχεία του ως ιατρός, τότε τα προσωπικά του στοιχεία καταχωρούνται στη συλλογή doctors.

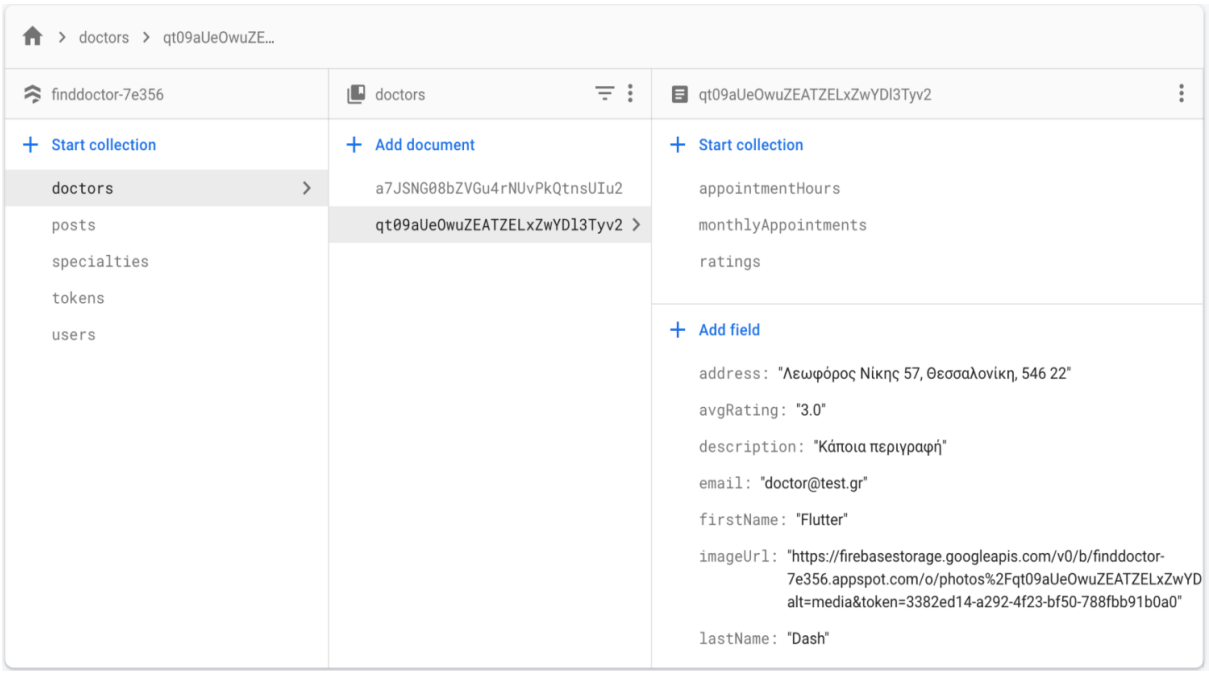
Η δημιουργία του προφίλ του γιατρού, πραγματοποιείται με τη μέθοδο *createDoctorProfile()*. Η μέθοδος περιέχει ως παράμετρο το προφίλ του γιατρού, το οποίο θα αποθηκευτεί στη συλλογή doctors.

Κώδικας 3.11 Μέθοδος δημιουργίας της συλλογής doctors

```
Future<void> createDoctorProfile(DoctorProfile doctorProfile) async
{
  FirebaseUser user = await _authService.getCurrentUser();
  await Firestore.instance
      .collection("doctors")
      .document(user.uid)
      .setData(doctorProfile.toMap());
}
```

Η συλλογή αυτή αφορά τους γιατρούς που είναι καταχωρημένοι στη βάση δεδομένων και περιέχει έγγραφα τα οποία περιλαμβάνουν ως κλειδί το μοναδικό αναγνωριστικό του κάθε γιατρού. Κάθε έγγραφο περιέχει τις βασικές πληροφορίες του ιατρού. Οι πληροφορίες αυτές είναι τα στοιχεία που έχει εισάγει ο ιατρός κατά την εγγραφή του και αφορούν τα πεδία της κλάσης DoctorProfile, η οποία παρουσιάστηκε παραπάνω. Επιπλέον, το κάθε έγγραφο της συλλογής «doctors» εκτός από τις

πληροφορίες του ιατρού, περιέχει τρεις υποσυλλογές, την συλλογή appointmentHours, monthlyAppointments και ratings. Παρακάτω παρουσιάζεται το στιγμιότυπο της συλλογής doctors όπου διακρίνονται τα έγγραφα και οι υποσυλλογές appointmentHours, ratings και monthlyAppointments.



Εικόνα 3.6 Συλλογή Doctors

Για να μπορεί ο χρήστης-ιατρός να δέχεται ραντεβού, είναι απαραίτητη η δήλωση των διαθέσιμων ωρών του. Η εφαρμογή, λαμβάνοντας υπόψιν τις ώρες εργασίας που έχει ορίσει ο χρήστης κατά την εγγραφή του, δημιουργεί αυτόματα τα έγγραφα, τα οποία περιέχουν τις διαθέσιμες ώρες του χρήστη, κατά τις οποίες μπορεί να δεχθεί επισκέψεις. Η υποσυλλογή «appointmentHours» περιέχει αυτά τα έγγραφα των οποίων τα κλειδιά παράγονται αυτόματα από το ίδιο το Firebase. Κάθε έγγραφο της συλλογής, περιέχει ένα πεδίο, το οποία ονομάζεται appointmentHour και υποδηλώνει τη διαθέσιμη ώρα. Για την αποθήκευση των κάθε διαθέσιμης ώρας του γιατρού, καλείται η μέθοδος *setAppointmentHours()*, η οποία δέχεται ως παραμέτρους το μοναδικό αναγνωριστικό του γιατρού και την διαθέσιμη ώρα.

Κώδικας 3.12 Μέθοδος δημιουργίας της συλλογής appointmentHours για κάθε γιατρό

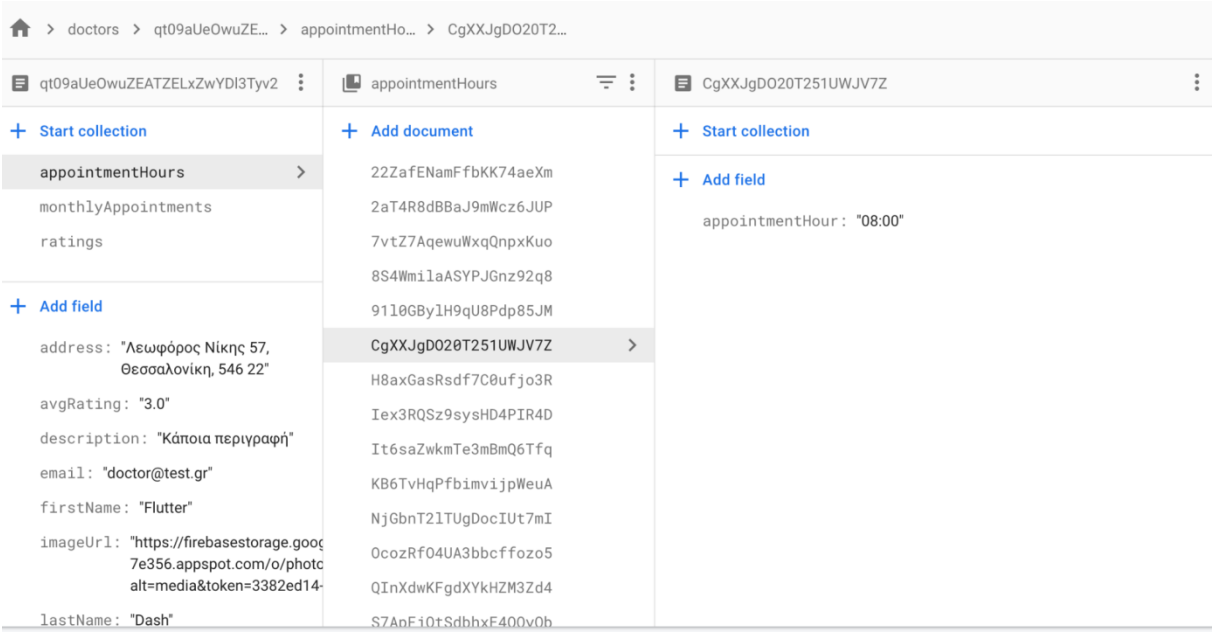
```
Future<void>setAppointmentHours(String doctorId, String
appointmentHour) async {
    DocumentReference doctor = await Firestore.instance
        .collection('doctors')
        .document(doctorId);

    Firestore.instance.collection("doctors").document(doctorId);

    DocumentReference appointmentHoursDocument =
    doctor.collection("appointmentHours").document();

    await appointmentHoursDocument,setData({"appointmentHour" :
appointmentHour});
}
```

Παρακάτω παρουσιάζεται το στιγμιότυπο της υποσυλλογής appointmentHours, στην οποία διακρίνονται τα έγγραφα με τις διαθέσιμες ώρες του γιατρού, καθώς και το πεδίο appointmentHour, το οποίο αντιστοιχεί σε μία συγκεκριμένη ώρα.



Εικόνα 3.7 Συλλογή AppointmentHours

Η συλλογή appointmentHours συνδέεται άμεσα με τη συλλογή monthlyAppointments. Μόλις ο γιατρός ορίσει τις διαθέσιμες ώρες για τις οποίες δέχεται επισκέψεις, τότε στη συλλογή «monthlyAppointments» δημιουργούνται, αυτόματα, κάποια έγγραφα. Τα έγγραφα αυτά, έχουν ως μοναδικό κλειδί τις ημερομηνίες των ραντεβού και περιέχουν ως δεδομένα, μία λίστα (appointmentHours) με τις διαθέσιμες ώρες για κάθε διαθέσιμη ημέρα. Κάθε στοιχείο της λίστας των διαθέσιμων ωρών, αποτελείται από τέσσερα πεδία. Τα πεδία αυτά είναι το μοναδικό αναγνωριστικό του γιατρού (doctorId), η ώρα (time), το μοναδικό αναγνωριστικό του χρήστη-ασθενή (uid) και μία δυαδική μεταβλητή (isAvailable), με δύο πιθανές τιμές true ή false, οι οποίες υποδηλώνουν εάν η συγκεκριμένη ώρα είναι διαθέσιμη για ραντεβού.

Η μέθοδος *setMonthlyAppointments()* χρησιμοποιείται για ολοκλήρωση της παραπάνω διαδικασίας. Συγκεκριμένα, περιλαμβάνει τρεις παραμέτρους, το μοναδικό αναγνωριστικό του γιατρού, τη λίστα με τις διαθέσιμες ώρες, και την ημερομηνία για την οποία θα οριστεί ως διαθέσιμη. Όταν δημιουργούνται τα έγγραφα, τότε όλες οι ημερομηνίες και ώρες είναι προκαθορισμένες έτσι ώστε να είναι διαθέσιμες για ραντεβού. Εξαιρείται η περίπτωση που ο γιατρός ορίσει ως διαθέσιμη ημέρα το Σάββατο, ή την Κυριακή ή ολόκληρο το Σαββατοκύριακο. Πιο συγκεκριμένα, η τιμή του πεδίου isAvailable είναι προκαθορισμένη ως “true”, ενώ το πεδίο uid είναι κενό. Όταν ο χρήστης κλείσει ραντεβού σε έναν γιατρό, τότε τα πεδία αυτά ενημερώνονται και το πεδίο isAvailable λαμβάνει τη τιμή false, ενώ το πεδίο uid συμπληρώνεται με το μοναδικό αναγνωριστικό του χρήστη. Με αυτόν τον τρόπο, η συγκεκριμένη ώρα της αντίστοιχης ημέρας, δηλώνεται ως μη διαθέσιμη για ραντεβού.

Παρακάτω παρουσιάζεται η μέθοδος *setMonthlyAppointments(),* η οποία στη συγκεκριμένη περίπτωση ορίζει διαθέσιμα τα ραντεβού για τις καθημερινές, ενώ τα Σαββατοκύριακα ορίζονται ως μη διαθέσιμα.

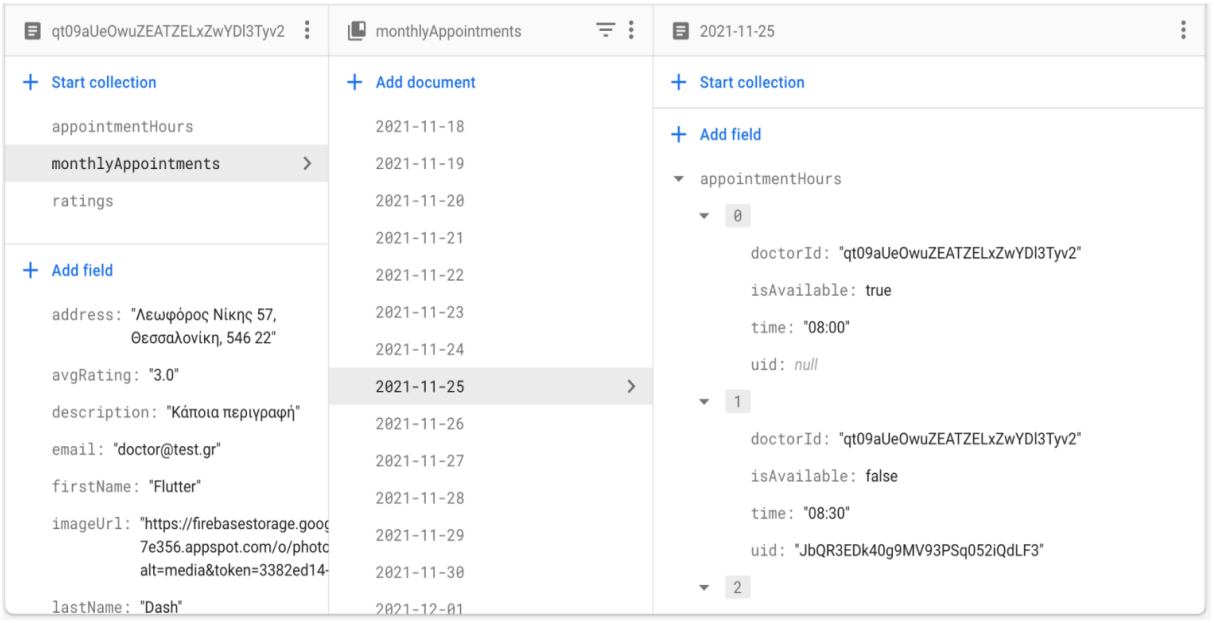
Κώδικας 3.13 Μέθοδος δημιουργίας της συλλογής monthlyAppointments για κάθε γιατρό

```
Future<void> setMonthlyAppointments(String doctorId,
```

```
List<DocumentSnapshot> appointments, DateTime day) async {
    List<Map<String,dynamic>> appointmentHours = [];
    Map<String,dynamic>> map = {};
    DocumentReference doctor =
        Firestore.instance.collection('doctors').document(doctorId)
;
    String formattedDay = DateFormat("yyyy-MM-dd").format(day);

    DocumentReference dayAppointment =
        doctor.collection("monthlyAppointments").document('$formatt
edDay');
    if (day.weekday <= 5) {
        for (var hour in appointments) {
          map = {
            "doctorId": doctorId,
            "time": hour['appointmentHour'],
            "isAvailable": true,
            "uid": null,
          };
          appointmentHours.add(map);
        }
      } else {
          map = {
            "doctorId": doctorId,
            "time": null,
            "isAvailable": false,
            "uid": null,
          };
          appointmentHours.add(map);
        }
        await dayAppointment.setData(
              {"appointmentHours": appointmentHours});
}
```

Παρακάτω φαίνεται το στιγμιότυπο της συλλογής monthlyAppointments ύστερα από την πετυχημένη εισαγωγή των διαθέσιμων ημερομηνιών.



Εικόνα 3.8 Υποσυλλογή MonthlyAppointments

Η υποσυλλογή ratings αφορά τις αξιολογήσεις που πραγματοποιούνται από τους χρήστες προς τους γιατρούς. Η υποσυλλογή αυτή διαθέτει όλες τις αξιολογήσεις που έχει λάβει ο γιατρός. Συγκεκριμένα, περιέχει έγγραφα των οποίων τα κλειδιά αντιστοιχούν στο μοναδικό αναγνωριστικό κάθε αξιολόγησης. Το αναγνωριστικό κάθε αξιολόγησης αντιστοιχεί στον συνδυασμό του μοναδικού αναγνωριστικού του χρήστη και του μοναδικού αναγνωριστικού του γιατρού και είναι της μορφής *UserId_DoctorId*. Κάθε έγγραφο της υποσυλλογής περιέχει δεδομένα, τα οποία αντιστοιχούν στα πεδία της κλάσης “Review”, η οποία αναλύθηκε παραπάνω.

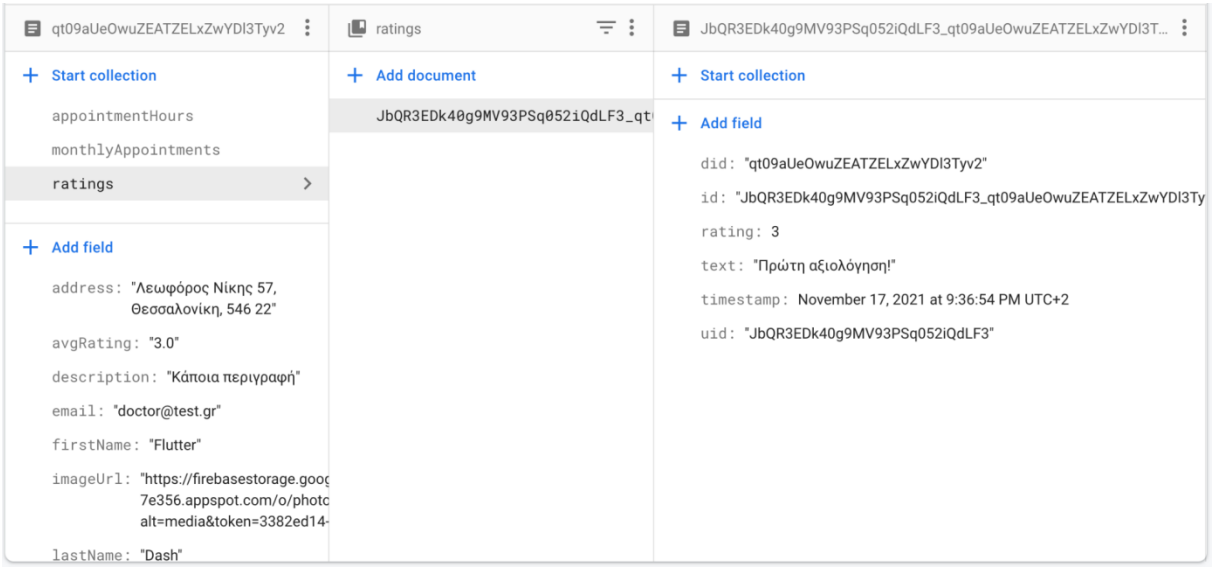
Για την δημιουργία μίας αξιολόγησης ενός χρήστη για ένα γιατρό, χρησιμοποιείται η μέθοδος *addReview()*. Η μέθοδος αυτή περιέχει δύο παραμέτρους, το μοναδικό αναγνωριστικό του γιατρού και τα δεδομένα της κλάσης “Review”. Κατά την κλήση της μεθόδου, δημιουργείται μία νέα αξιολόγηση του χρήστη προς τον γιατρό που τον ενδιαφέρει. Εκτός από αυτό, την ίδια στιγμή, υπολογίζεται ο νέος συνολικός αριθμός των αξιολογήσεων του γιατρού. Εν συνεχεία ενημερώνεται η μέση τιμή του βαθμού όλων των αξιολογήσεων που έχει λάβει ο γιατρός. Η συγκεκριμένη μέθοδος παρουσιάζεται στο παρακάτω στιγμιότυπο.

Κώδικας 3.14 Μέθοδος δημιουργίας μίας αξιολόγησης σε ένα γιατρό

```
Future<void> addReview({String doctorId, Review review}) async {
  String userId = await Firestore.instance.getUserId();
  DocumentReference doctor =
      Firestore.instance.collection('doctors').document(doctorId);
  DocumentReference ratingsDocument =
      doctor.collection('ratings').document("$userId\_$doctorId");
  await Firestore.instance.runTransaction((transaction) {
    return transaction
        .get(doctor)
        .then((DocumentSnapshot doc) =>
            DoctorProfile.fromSnapshot(doc))
            .then((DoctorProfile doctorProfile) {
            int newNumRatings = doctorProfile.numRatings + 1;
            String newAverage =
```

```
                (((doctorProfile.numRatings *
                   double.parse(doctorProfile.avgRating)) +
                   review.rating) / newRatings).toStringAsFixed(1);
                transaction.update(doctor, {
                 'numRatings': newNumRatings,
                 'avgRating': newAverage,
                });
                return transaction.set(ratingsDocument, {
                'id': "$userId\_$doctorId",
                'rating': review.rating,
                'text': review.text,
                'timestamp': review.timestamp ??
                             FieldValue.serverTimestamp(),
                'did': doctorId,
                'uid': review.userId,
                });
      });
   });
}
```

Παρακάτω παρουσιάζεται το στιγμιότυπο της υποσυλλογής “Ratings”, όπου διακρίνονται τα έγγραφα και τα πεδία της κλάσης “Review”.



Εικόνα 3.9 Υποσυλλογή Ratings

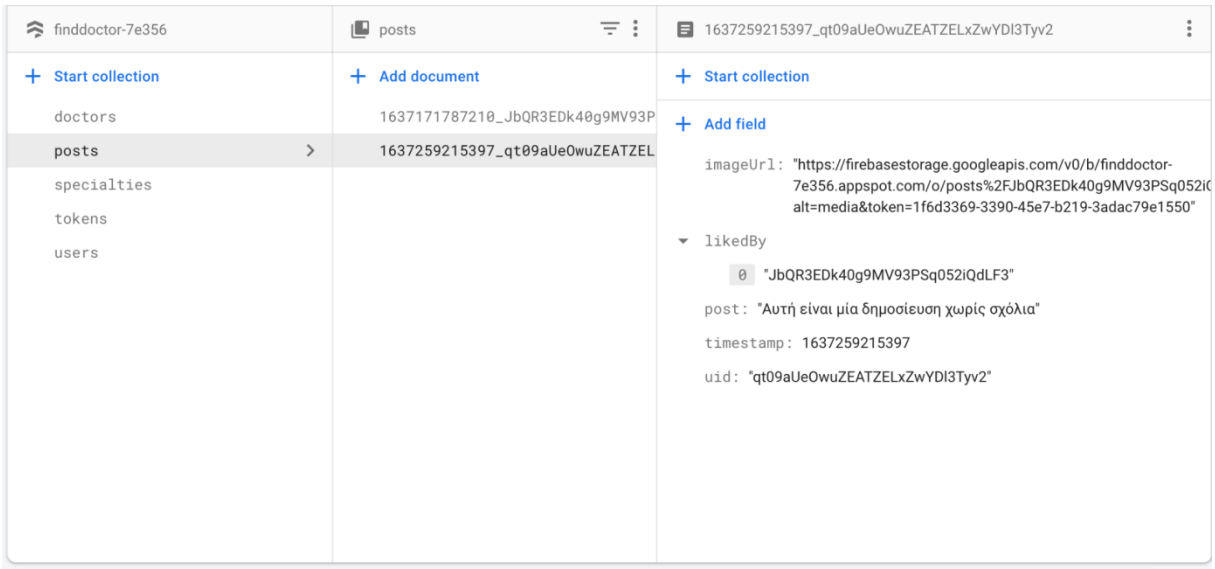
Η εφαρμογή παρέχει τη δυνατότητα στους χρήστες να αλληλοεπιδρούν μεταξύ τους. Οι χρήστες μπορούν να δημοσιεύουν κείμενα και φωτογραφίες, ενώ ταυτόχρονα είναι δυνατόν να σχολιάζουν και να αντιδρούν στις δημοσιεύσεις. Για αυτή την λειτουργικότητα, υπάρχει η συλλογή posts η οποία αφορά τις δημοσιεύσεις των χρηστών.

Η συλλογή αυτή περιέχει ως κλειδί τον συνδυασμό της ημερομηνίας της δημοσίευσης και του μοναδικού αναγνωριστικού του χρήστη, ο οποίος δημιούργησε τη δημοσίευση. Πιο συγκεκριμένα, η ημερομηνία καταγράφεται με τη μορφή timestamp, δηλαδή ως ένας ακέραιος αριθμός σε χιλιοστά του δευτερολέπτου. Επομένως το κλειδί κάθε εγγράφου έχει την εξής μορφή: *timestamp_UserId*. Για τη δημιουργία μίας δημοσίευσης, χρησιμοποιείται η μέθοδος *createPost()*, η οποία περιλαμβάνει τα πεδία της κλάσης «Post», η οποία παρουσιάστηκε παραπάνω.

Κώδικας 3.15 Μέθοδος δημιουργίας μίας δημοσίευσης

```
Future<void> createPost(Post post) async {
  await Firestore.instance
      .collection("posts")
      .document("${post.timestamp}_${post.uid}")
      .setData(post.toMap());
}
```

Μία επιτυχημένη δημιουργία δημοσίευσης σημαίνει ότι κάθε έγγραφο περιέχει δεδομένα που εισήγαγε ο χρήστης και αντιστοιχούν στα πεδία της κλάσης «Post». Παρακάτω παρουσιάζεται το στιγμιότυπο της συλλογής “posts”:



Εικόνα 3.10 Συλλογή Posts

Εκτός από τα δεδομένα της κλάσης «Post», κάθε έγγραφο της συλλογής posts, μπορεί να περιέχει την υποσυλλογή comments. Αυτή η υποσυλλογή δημιουργείται όταν κάποιος χρήστης σχολιάσει σε μία δημοσίευση. Για την προσθήκη ενός σχολίου σε μία δημοσίευση χρησιμοποιείται η μέθοδος *addComment()*, η οποία περιέχει ως παραμέτρους το μοναδικό αναγνωριστικό της δημοσίευσης και τα δεδομένα της κλάσης Comment, η οποία αναλύθηκε παραπάνω.

Η επιτυχημένη εισαγωγή ενός σχολίου σημαίνει ότι η υποσυλλογή comments, περιέχει έγγραφα τα οποία χρησιμοποιούν ως κλειδί, το μοναδικό αναγνωριστικό του χρήστη, ο οποίος έχει δημιουργήσει ένα σχόλιο σε κάποια δημοσίευση. Το κλειδί κάθε εγγράφου είναι της μορφής “comment_UserId”, ενώ τα δεδομένα αντιστοιχούν στα δεδομένα της κλάσης Comment, με τις αντίστοιχες τιμές που έχει ορίσει ο χρήστης στο σχόλιό του.

Κώδικας 3.16 Μέθοδος εισαγωγής σχολίου σε μία δημοσίευση

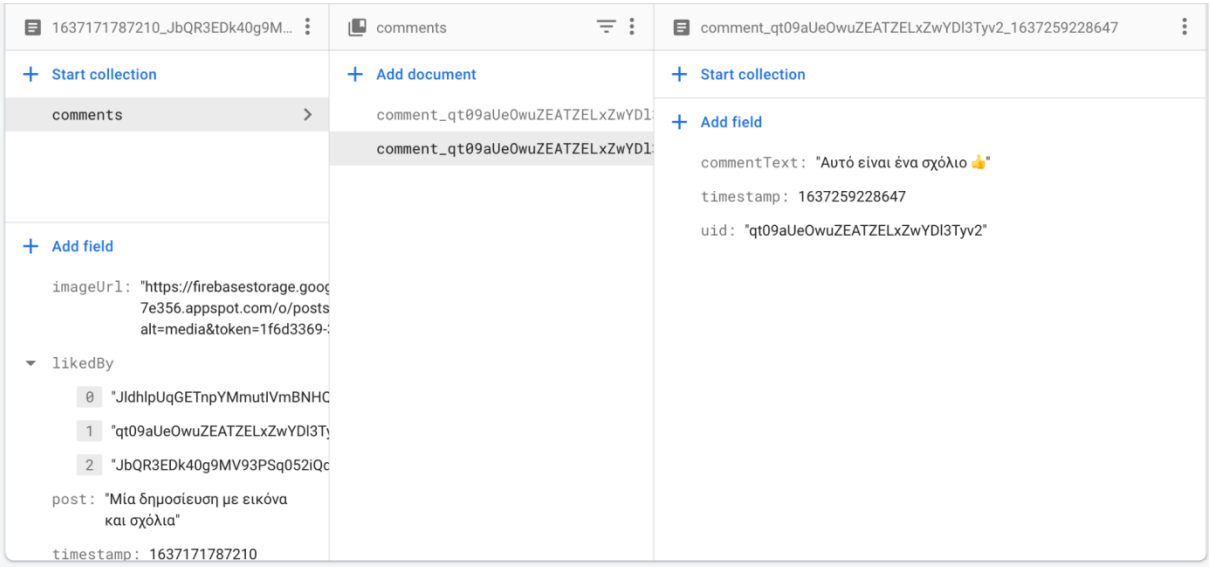
```
Future<void> addComment({String postId, Comment comment}) async {
  DocumentReference post =
      Firestore.instance.collection('posts').document(postId);

  FirebaseUser user = await Firestore.instance.getCurrentUser();

  DocumentReference commentDocument =
      post.collection('comments').document(
      "comment_${user.uid}_${comment.timestamp ??
```

```
        FieldValue.serverTimestamp()}");
    await commentDocument.setData({
      'commentText': comment.commentText,
      'timestamp': comment.timestamp ?? FieldValue.serverTimestamp(),
      'uid': user.uid,
    });
}
```

Παρακάτω προβάλλεται στιγμιότυπο της υποσυλλογής comments:



Εικόνα 3.11 Υποσυλλογή Comments

Όταν ένας χρήστης ως ασθενής κλείσει ένα ραντεβού σε κάποιον γιατρό, τότε χρειάζεται να ενημερωθεί ο γιατρός. Ο τρόπος με τον οποίο η εφαρμογή ενημερώνει τους γιατρούς όταν δέχονται νέα ραντεβού είναι μέσω ειδοποιήσεων στα κινητά τους τηλέφωνα. Όπως παρουσιάστηκε στο κεφάλαιο 2, προκειμένου να ενεργοποιηθούν οι ειδοποιήσεις σε έξυπνα κινητά, η εφαρμογή χρησιμοποιεί την πλατφόρμα του Firebase και συγκεκριμένα το εργαλείο Cloud Messaging.

Για την αποστολή ειδοποιήσεων σε μία συσκευή το Firebase χρειάζεται να γνωρίζει σε ποια συσκευή θα σταλεί η κατάλληλη ειδοποίηση. Για να πραγματοποιηθεί αυτό, η εφαρμογή, μέσω του Firebase, δημιουργεί ένα μοναδικό firebase token, το οποίο αντιστοιχεί στην έξυπνη συσκευή, στην οποία είναι εγκατεστημένη η εφαρμογή.

Κώδικας 3.17 Λήψη του Cloud Firebase Messaging Token

```
String token = await FirebaseMessaging().getToken();
```

Αυτό το μοναδικό firebase token δημιουργείται μόνο για τους γιατρούς διότι μόνο οι γιατροί λαμβάνουν ειδοποιήσεις στις συσκευές τους. Ένας χρήστης γιατρός μπορεί να έχει πολλά firebase tokens. Ο αριθμός των firebase token που αντιστοιχούν σε έναν γιατρό, εξαρτάται από το πλήθος των συσκευών στις οποίες είναι εγκατεστημένη η εφαρμογή και έχει συνδεθεί ο γιατρός.

Τα μοναδικά firebase tokens των γιατρών αποθηκεύονται στη συλλογή tokens. Η συλλογή αυτή περιέχει έγγραφα τα οποία περιέχουν ως κλειδί, το μοναδικό αναγνωριστικό του κάθε γιατρού. Κάθε έγγραφο περιέχει δεδομένα τα οποία αποτελούν μία λίστα με τα firebase tokens των συσκευών που

χρησιμοποιεί ο γιατρός. Το firebase token δημιουργείται κατά την σύνδεση του γιατρού και εγγράφεται στη συλλογή. Αντιθέτως, όταν ο γιατρός αποσυνδεθεί από την εφαρμογή, τότε το firebase token αφαιρείται από τη συλλογή. Συνεπώς, όταν ο γιατρός έχει αποσυνδεθεί από την εφαρμογή, τότε δεν λαμβάνει ειδοποιήσεις.

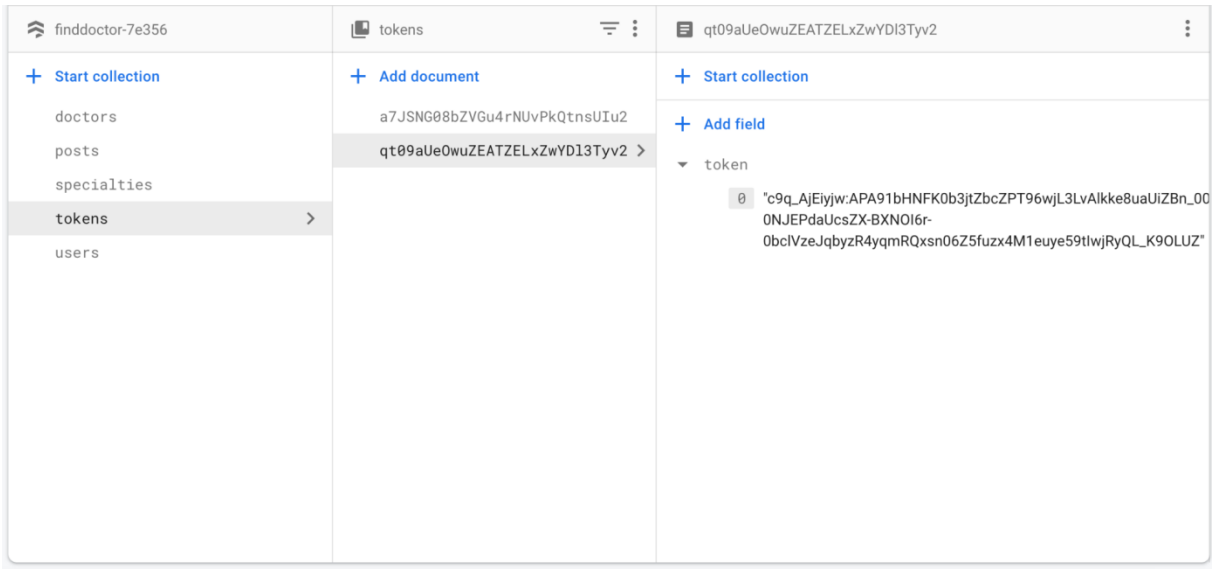
Η μέθοδος *setToken()* χρησιμοποιείται για την λήψη και την αποθήκευση του firebase token. Η μέθοδος αυτή, αρχικά λαμβάνει το firebase token για την συσκευή και εν συνεχεία το αποθηκεύει στη λίστα με τα tokens του χρήστη.

Κώδικας 3.18 Μέθοδος αποθήκευσης του Firebase token

```
Future<void> setToken(String token) async {
  String token = await FirebaseMessaging().getToken();
  final doctor = await _authService.getCurrentUser();
  List tokens = [];
  DocumentSnapshot document =
      await Firestore.instance.collection("tokens")
      .document(doctor.uid).get();

  if (document!=null&&document.data!=null&&document.data[0]!= null)
  {
    tokens = document.data[0].data["token"];
  }
  tokens.add(token);
  await Firestore.instance
      .collection("tokens")
      .document(doctor.uid)
      .setData({"token": tokens});
}
```

Παρακάτω παρουσιάζεται η συλλογή tokens, ύστερα από την επιτυχημένη αποθήκευση του firebase token ενός χρήστη.

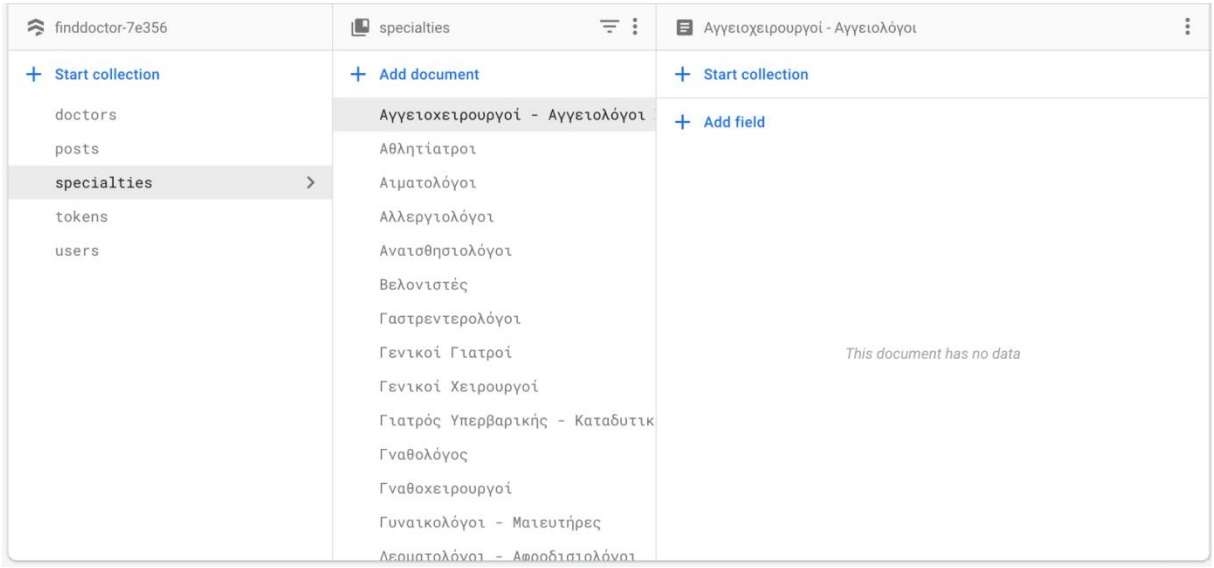


Εικόνα 3.12 Συλλογή Tokens

Η τελευταία συλλογή της εφαρμογής είναι η συλλογή “specialties”. Η συλλογή αφορά όλες τις ειδικότητες των γιατρών. Πιο συγκεκριμένα, η συλλογή αυτή, περιέχει έγγραφα τα οποία περιέχουν μόνο κλειδιά. Κάθε κλειδί αντιστοιχεί σε μία ειδικότητα γιατρού. Οι ειδικότητες των γιατρών

αποθηκεύονται στη βάση δεδομένων, έτσι ώστε η εφαρμογή να φορτώνει δυναμικά τις πληροφορίες. Με αυτόν τον τρόπο, οποιαδήποτε αλλαγή στις ειδικότητες, θα πραγματοποιηθεί άμεσα, χωρίς κάποια ενημέρωση της συσκευής. Αντιθέτως, αν οι ειδικότητες αποθηκεύονταν στατικά στην εφαρμογή, τότε για οποιαδήποτε αλλαγή στις ειδικότητες, θα χρειαζόταν ενημέρωση της εφαρμογής.

Παρακάτω παρουσιάζεται το στιγμιότυπο της συλλογής specialties:



Εικόνα 3.13 Συλλογή Specialties

3.4 Επίλογος

Στο κεφάλαιο αυτό παρουσιάστηκε και αναλύθηκε ο τρόπος με τον οποίο χρησιμοποιήθηκαν οι τεχνολογίες για την ανάπτυξη της εφαρμογής. Συγκεκριμένα, περιεγράφηκαν τα μοντέλα των κλάσεων της εφαρμογής. Στη συνέχεια, αναλύθηκε το Firebase και οι λειτουργίες του, δίνοντας έμφαση στο τρόπο με τον οποίο ενσωματώνονται στην εφαρμογή. Τέλος παρουσιάστηκε η δομή της βάσης δεδομένων. Στο επόμενο κεφάλαιο παρουσιάζονται όλες οι οθόνες και οι λειτουργικότητες της εφαρμογής.

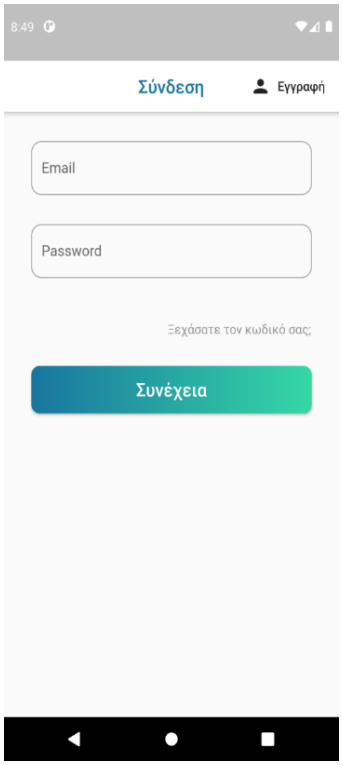
Κεφάλαιο 4ο: Παρουσίαση και περιγραφή

4.1 Εισαγωγή

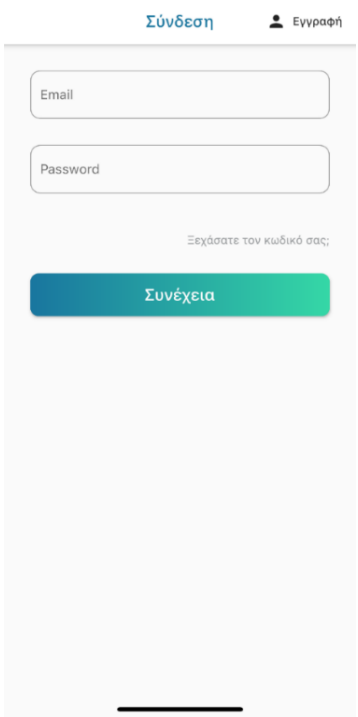
Σε αυτό το κεφάλαιο παρουσιάζονται και περιγράφονται όλες οι οθόνες και οι λειτουργίες της εφαρμογής, τόσο από τη μεριά του ασθενή, όσο και από τη μεριά του γιατρού. Κατά τον σχεδιασμό και την ανάπτυξη της εφαρμογή, δόθηκε ιδιαίτερη σημασία στη διεπαφή χρήστη, διότι μία καλαίσθητη και λειτουργική εφαρμογή προσελκύει και ικανοποιεί τον τελικό χρήστη.

4.2 Οθόνη σύνδεσης

Όταν ο χρήστης χρησιμοποιήσει για πρώτη φορά την εφαρμογή, τότε η πρώτη οθόνη που εμφανίζεται είναι η οθόνη σύνδεσης. Προκειμένου ένας χρήστης να έχει πρόσβαση στην εφαρμογή, πρέπει πρώτα να συνδεθεί στο σύστημα. Σε αυτή την οθόνη, ο χρήστης εισάγει το προσωπικό του email και τον κωδικό πρόσβασής του. Πατώντας το κουμπί “Συνέχεια”, ελέγχονται τα διαπιστευτήριά του και συνδέεται στην εφαρμογή, αν τα διαπιστευτήριά του είναι σωστά.



Εικόνα 4.1 Οθόνη σύνδεσης στο Android



Εικόνα 4.2 Οθόνη σύνδεσης στο iOS

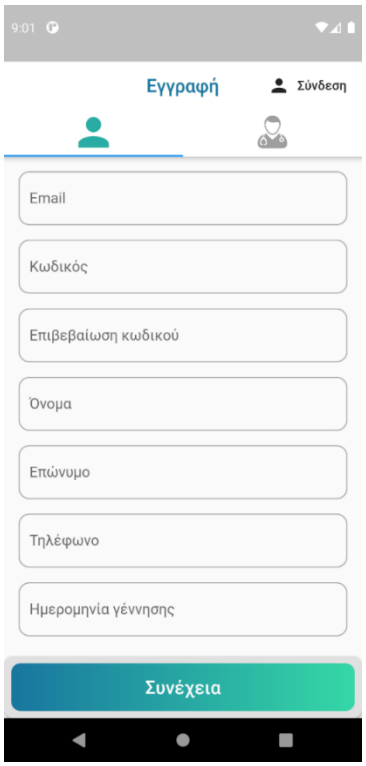
4.3 Οθόνη εγγραφής

Για να μπορέσει ο χρήστης να συνδεθεί στην εφαρμογή, χρειάζεται πρώτα να δημιουργήσει τον προσωπικό του λογαριασμό στην εφαρμογή. Η οθόνη της σύνδεσης (Εικόνα 4.1, Εικόνα 4.2), περιέχει το κουμπί “Εγγραφή”, το οποίο οδηγεί τον χρήστη στην οθόνη της εγγραφής του χρήστη. Η οθόνη αυτή περιέχει δύο καρτέλες, όπου η πρώτη καρτέλα αντιστοιχεί στην δημιουργία λογαριασμού ως ασθενής και η δεύτερη καρτέλα αντιστοιχεί στην δημιουργία λογαριασμού ως γιατρός.

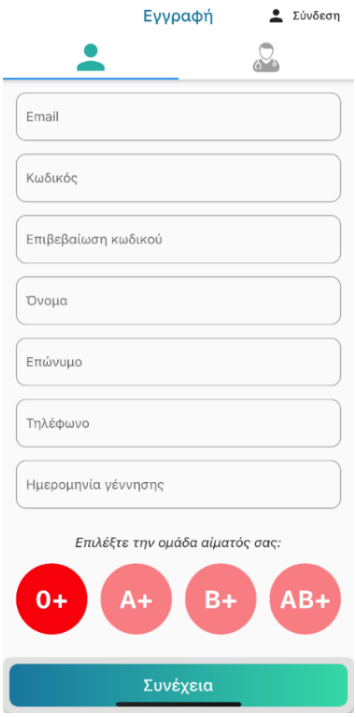
Στην οθόνη της εγγραφής του ασθενή, ο χρήστης εισάγει τις απαραίτητες πληροφορίες για την δημιουργία του λογαριασμού του ως ασθενής. Πιο συγκεκριμένα ο χρήστης θα πρέπει να εισάγει υποχρεωτικά τα εξής:

- το email του
- τον προσωπικό του κωδικό πρόσβασης,
- το ονοματεπώνυμό του
- ένα τηλέφωνο επικοινωνίας
- την ηλικία του
- την ομάδα αίματός του
- το ύψος του
- το βάρος του
- το φύλο του

Επιπλέον, δίνεται η δυνατότητα να εισάγει μία φωτογραφία για το προφίλ του. Σε περίπτωση που δεν εισαχθεί κάποια φωτογραφία, η εφαρμογή θα αναπαριστά τον χρήστη με μία προεπιλεγμένη εικόνα.



Εικόνα 4.3 Οθόνη εγγραφής του ασθενή στο Android



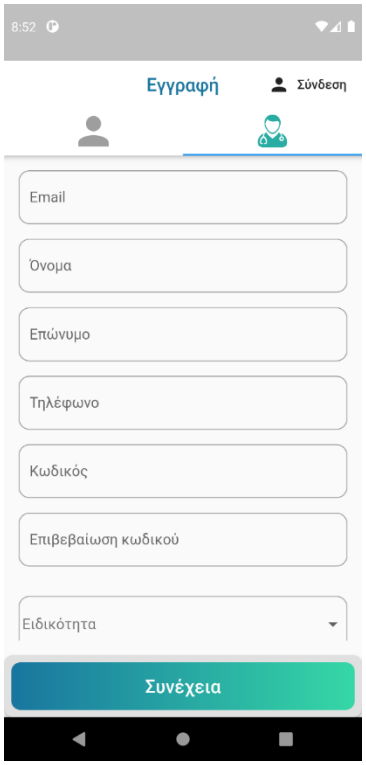
Εικόνα 4.4 Οθόνη εγγραφής του ασθενή στο iOS

Στην οθόνη της εγγραφής του γιατρού, ο χρήστης εισάγει τις απαραίτητες πληροφορίες για την δημιουργία του λογαριασμού του ως γιατρός. Πιο συγκεκριμένα ο χρήστης θα πρέπει να εισάγει υποχρεωτικά τα εξής:

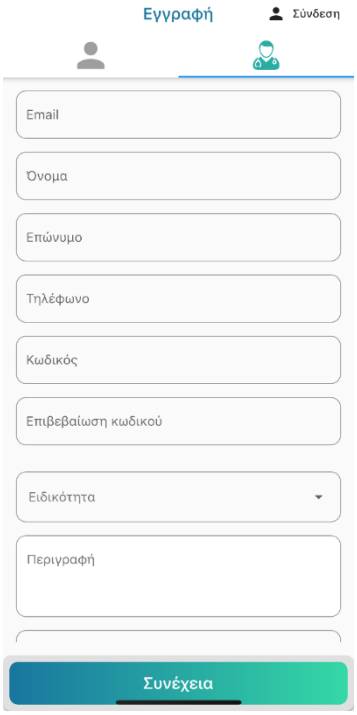
- το email του
- το ονοματεπώνυμό του
- το τηλέφωνό του
- τον προσωπικό του κωδικό πρόσβασης,
- ένα τηλέφωνο επικοινωνίας
- την ειδικότητά του
- μία περιγραφή

- την διεύθυνσή του
- το ωράριο εργασίας του

Επιπλέον, δίνεται η δυνατότητα να εισάγει μία φωτογραφία για το προφίλ του. Σε περίπτωση που δεν εισαχθεί κάποια φωτογραφία, η εφαρμογή θα αναπαριστά τον χρήστη με μία προεπιλεγμένη εικόνα.



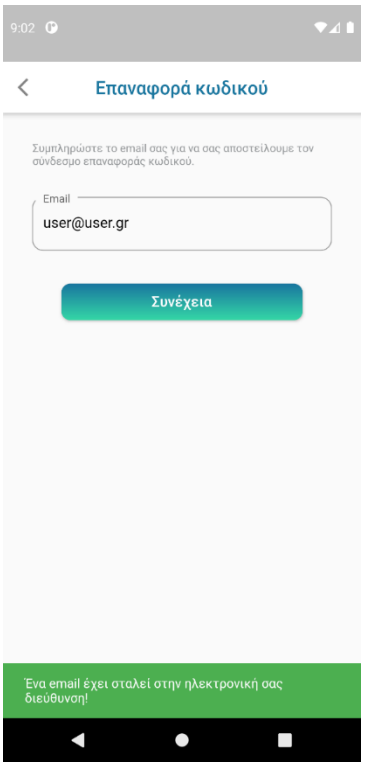
Εικόνα 4.5 Οθόνη εγγραφής του γιατρού στο Android



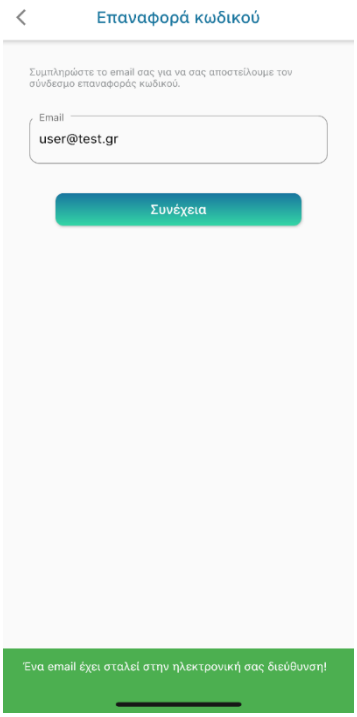
Εικόνα 4.6 Οθόνη εγγραφής του γιατρού στο iOS

4.4 Οθόνη επαναφοράς του κωδικού πρόσβασης

Στη περίπτωση όπου ο χρήστης δεν θυμάται τον προσωπικό του κωδικό και επομένως δεν μπορεί να συνδεθεί στην εφαρμογή, υπάρχει το κουμπί «΄Ξεχάσατε τον κωδικό σας;». Το κουμπί αυτό οδηγεί στην οθόνη επαναφοράς του κωδικού πρόσβασης ενός χρήστη. Στην οθόνη αυτή, ο εγγεγραμμένος χρήστης πληκτρολογεί το email του. Το email που εισάγει θα πρέπει να αντιστοιχεί με το email σύμφωνα με το οποίο ο χρήστης εγγράφηκε στην εφαρμογή. Αν το email του χρήστη είναι σωστό, τότε εμφανίζεται ένα μήνυμα επιτυχίας. Την ίδια στιγμή, ο χρήστης θα λάβει ένα mail το οποίο περιέχει τις απαραίτητες οδηγίες για την επαναφορά του κωδικού του χρήστη.



Εικόνα 4.7 Οθόνη επαναφοράς του κωδικού πρόσβασης σο Android

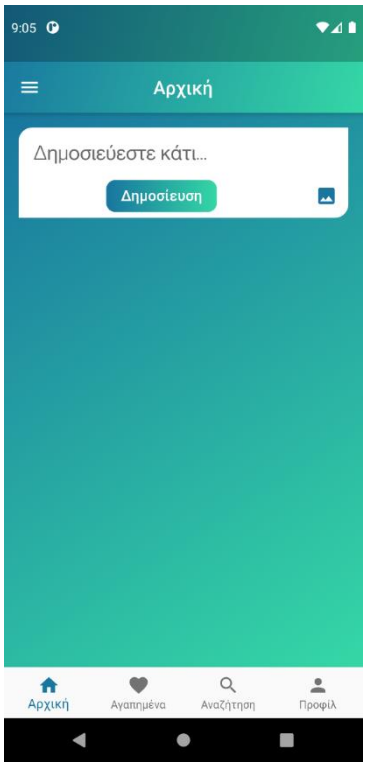


Εικόνα 4.8 Οθόνη επαναφοράς του κωδικού πρόσβασης στο iOS

4.5 Αρχική οθόνη

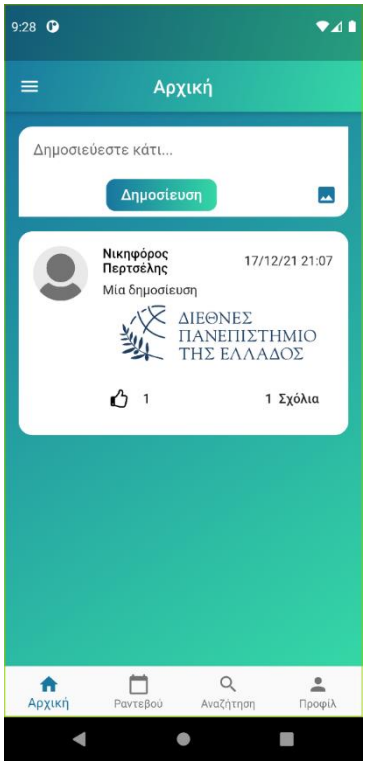
Όταν ο εγγεγραμμένος χρήστης συνδέεται επιτυχώς στην εφαρμογή, τότε οδηγείται στην αρχική οθόνη. Στην αρχική οθόνη, παρουσιάζονται οι πιο πρόσφατες δημοσιεύσεις όλων των χρηστών. Επίσης, υπάρχει ένα πεδίο κειμένου όπου ο εγγεγραμμένος χρήστης μπορεί να δημοσιεύσει ένα κείμενο αλλά και μία φωτογραφία.

Παράλληλα, περιλαμβάνεται το κεντρικό μενού της εφαρμογής, το οποίο είναι τοποθετημένο στο κάτω μέρος της κάθε οθόνης. Το κεντρικό μενού είναι διαφορετικό ανάμεσα στους χρήστες που είναι εγγεγραμμένοι ως ασθενείς και γιατροί. Το μενού στους ασθενείς περιέχει την αρχική οθόνη, την οθόνη των αγαπημένων γιατρών, την οθόνη αναζήτησης γιατρού και το προφίλ του χρήστη. Αντίστοιχα, το μενού στους γιατρούς είναι σχεδόν όμοιο με αυτό των ασθενών, με την μοναδική διαφορά ότι αντί για την οθόνη των αγαπημένων, περιέχεται η οθόνη της διαχείρισης των ραντεβού του χρήστη.

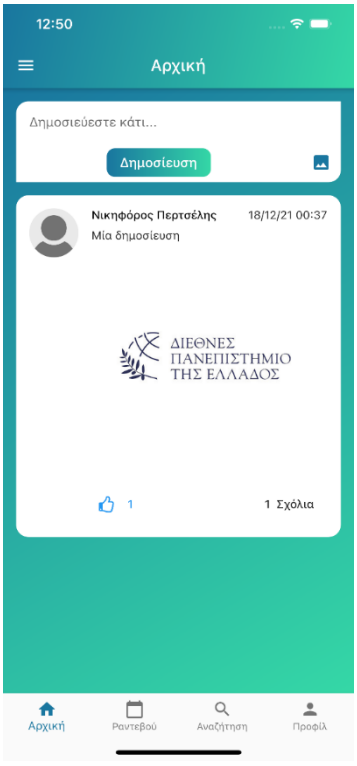


Εικόνα 4.9 Αρχική οθόνη του ασθενή στο Android

Εικόνα 4.10 Αρχική οθόνη του ασθενή στο iOS



Εικόνα 4.11 Αρχική οθόνη του γιατρού στο Android



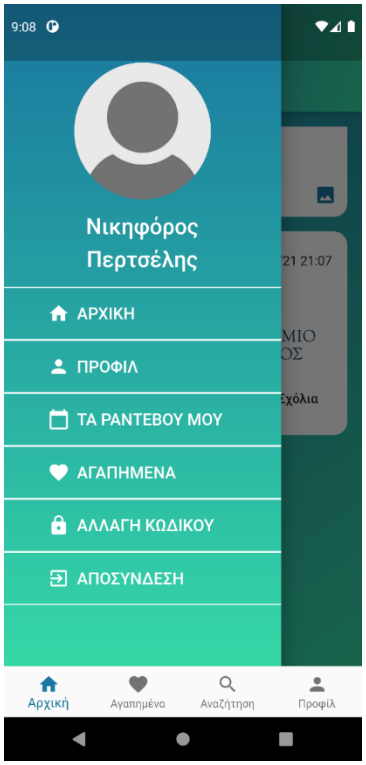
Εικόνα 4.12 Αρχική οθόνη του γιατρού στο iOS

4.6 Πλαϊνό μενού

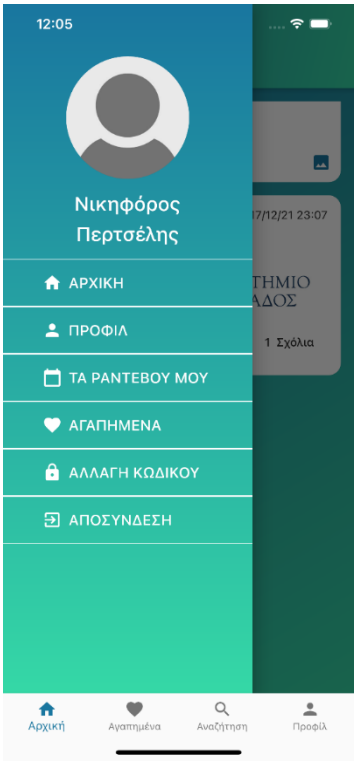
Σε όλες τις οθόνες της εφαρμογής, υπάρχει ένα εικονίδιο με τρεις οριζόντιες γραμμές, το οποίο βρίσκεται στο πάνω αριστερό μέρος κάθε οθόνης και εμφανίζει το πλαϊνό μενού της εφαρμογής. Το πλαϊνό μενού είναι διαφορετικό για τους χρήστες οι οποίοι είναι ασθενής και διαφορετικό για τους χρήστες γιατρούς.

Το πλαϊνό μενού που προβάλλεται στον ασθενή, περιέχει επτά επιλογές:

1) Αρχική. Ο χρήστης πλοηγείται στην αρχική οθόνη.
2) Προφίλ. Ο χρήστης πλοηγείται στην οθόνη του προφίλ του.
3) Τα ραντεβού μου. Ο χρήστης πλοηγείται στην οθόνη των ραντεβού του.
4) Αγαπημένα. Ο χρήστης πλοηγείται στην οθόνη με τους αγαπημένους του γιατρούς.
5) Αλλαγή κωδικού πρόσβασης. Ο χρήστης πλοηγείται στην οθόνη αλλαγής του προσωπικού του κωδικού πρόσβασης.
6) Αποσύνδεση. Ο χρήστης αποσυνδέεται από την εφαρμογή και πλοηγείται στην οθόνη σύνδεσης.



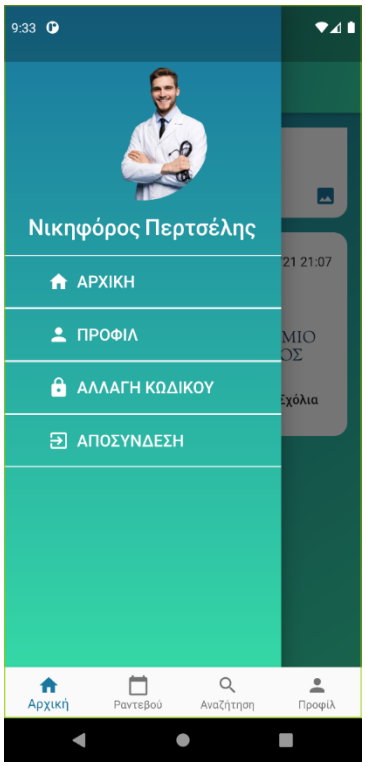
Εικόνα 4.13 Πλαϊνό μενού του ασθενή στο Android



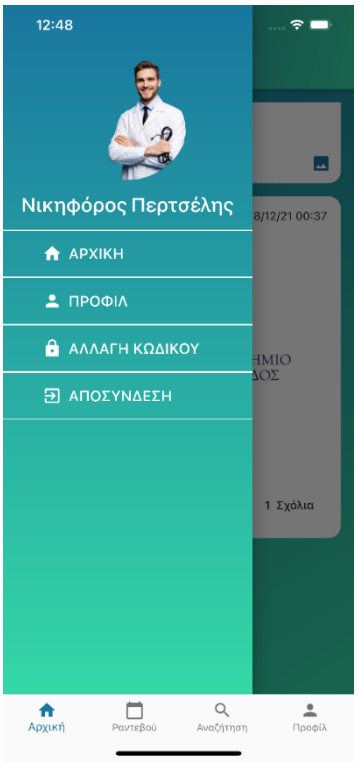
Εικόνα 4.14 Πλαϊνό μενού του ασθενή στο iOS

Το πλαϊνό μενού που προβάλλεται στον γιατρό, περιέχει τέσσερις επιλογές:

1) Αρχική. Ο χρήστης πλοηγείται στην αρχική οθόνη.
2) Προφίλ. Ο χρήστης πλοηγείται στην οθόνη του προφίλ του.
3) Αλλαγή κωδικού πρόσβασης. Ο χρήστης πλοηγείται στην οθόνη αλλαγής του προσωπικού του κωδικού πρόσβασης.
4) Αποσύνδεση. Ο χρήστης αποσυνδέεται από την εφαρμογή και πλοηγείται στην οθόνη σύνδεσης.



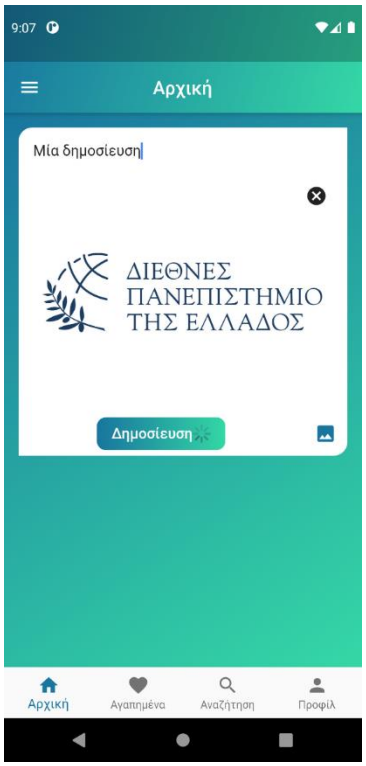
Εικόνα 4.15 Πλαϊνό μενού του γιατρού στο Android



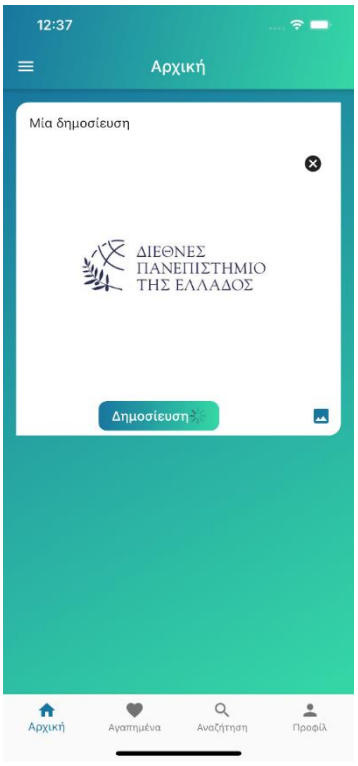
Εικόνα 4.16 Πλαϊνό μενού του γιατρού στο iOS

4.7 Οθόνη δημιουργίας μίας δημοσίευσης

Όπως προαναφέρθηκε, στην αρχική οθόνη υπάρχει η δυνατότητα της δημιουργίας μίας δημοσίευσης από τους εγγεγραμμένους χρήστες. Ο χρήστης έχει τη δυνατότητα να γράψει ένα κείμενο ή/και να επισυνάψει μία φωτογραφία από τα αρχεία της συσκευής του. Όταν πατήσει το κουμπί «Δημοσίευση», τότε η δημοσίευσή του εμφανίζεται με τις υπόλοιπες δημοσιεύσεις όλων των χρηστών.



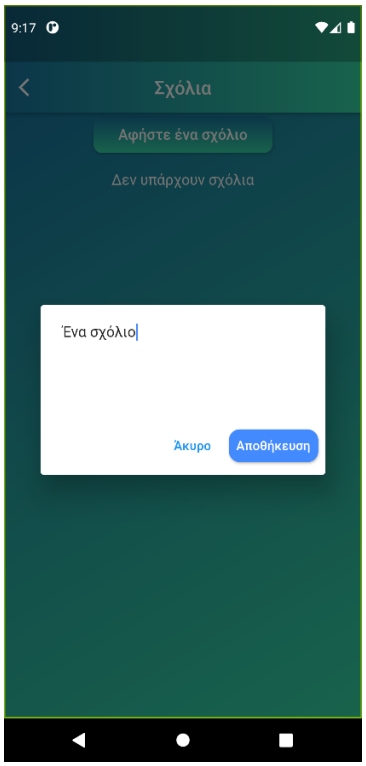
Εικόνα 4.17 Δημιουργία μίας δημοσίευση στο Android



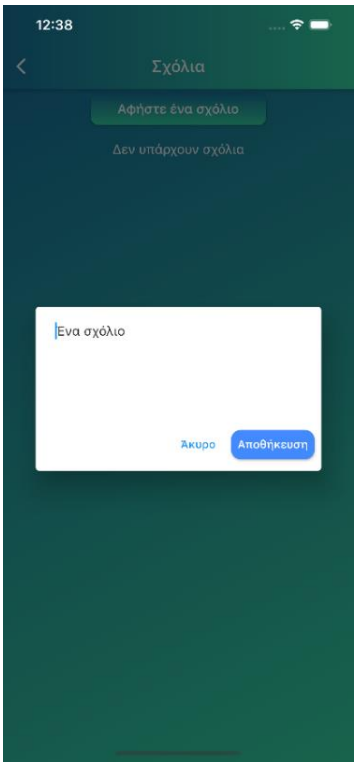
Εικόνα 4.18 Δημιουργία μίας δημοσίευση στο iOS

4.8 Οθόνη δημιουργίας ενός σχολίου σε μία δημοσίευση

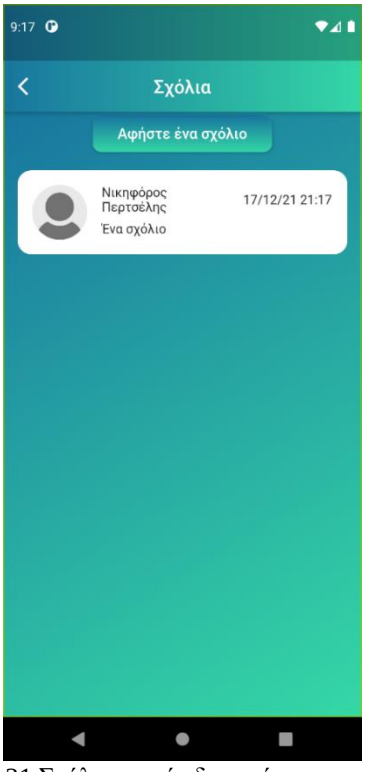
Ο εγγεγραμμένος χρήστης, έχει τη δυνατότητα να σχολιάσει σε οποιαδήποτε δημοσίευση πατώντας το κουμπί «Σχόλια». Πατώντας το κουμπί, ο χρήστης πλοηγείται στην οθόνη «Σχόλια». Η οθόνη αυτή περιέχει όλα τα σχόλια της αντίστοιχης δημοσίευσης που επέλεξε ο χρήστης. Ο χρήστης έχει, επιπλέον, τη δυνατότητα να σχολιάσει στην δημοσίευση, πατώντας το κουμπί «Αφήστε ένα σχόλιο», όπου πατώντας το, εμφανίζεται μία αναδυόμενη φόρμα εισαγωγής κειμένου. Εκεί ο χρήστης πληκτρολογεί το κείμενό του. Πατώντας το κουμπί «Αποθήκευση», δημιουργείται το σχόλιό του.



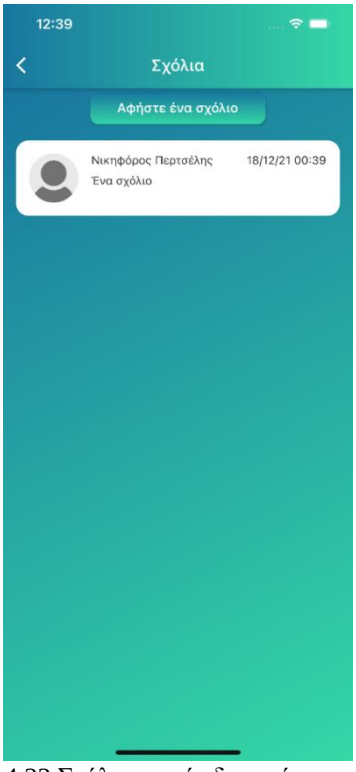
Εικόνα 4.19 Δημιουργία σχολίου σε μία δημοσίευση στο Android



Εικόνα 4.20 Δημιουργία σχολίου σε μία δημοσίευση στο iOS



Εικόνα 4.21 Σχόλιο σε μία δημοσίευση στο Android

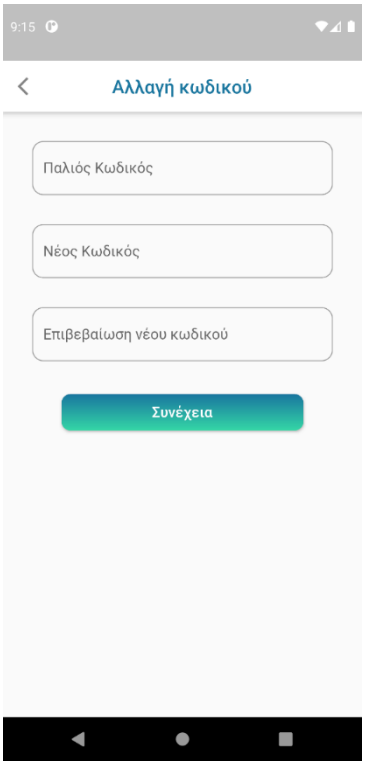


Εικόνα 4.22 Σχόλιο σε μία δημοσίευση στο iOS

4.9 Οθόνη αλλαγής κωδικού πρόσβασης

Στη περίπτωση όπου ο εγγεγραμμένος χρήστης επιθυμεί να τροποποιήσει τον προσωπικό του κωδικό πρόσβασης, υπάρχει το κουμπί «Αλλαγή κωδικού πρόσβασης», το οποίο βρίσκεται στο πλαϊνό μενού.

Το κουμπί αυτό οδηγεί στην οθόνη αλλαγής του κωδικού πρόσβασης του χρήστη. Στην οθόνη αυτή, ο εγγεγραμμένος χρήστης θα πρέπει να εισάγει τον παλαιό και τον νέο κωδικό που επιθυμεί. Από τη στιγμή που θα εισαχθούν σωστά τα πεδία, πατώντας το κουμπί «Συνέχεια», ο κωδικός του χρήστη ενημερώνεται.



Εικόνα 4.23 Αλλαγή του κωδικού πρόσβασης στο Android

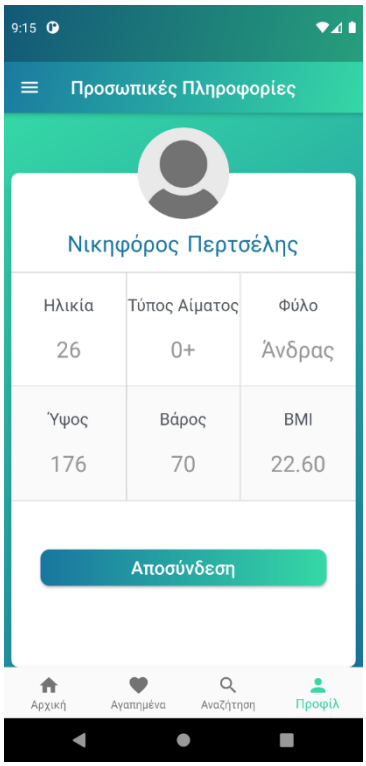
Εικόνα 4.24 Αλλαγή του κωδικού πρόσβασης στο iOS

4.10 Οθόνη του προφίλ του ασθενή

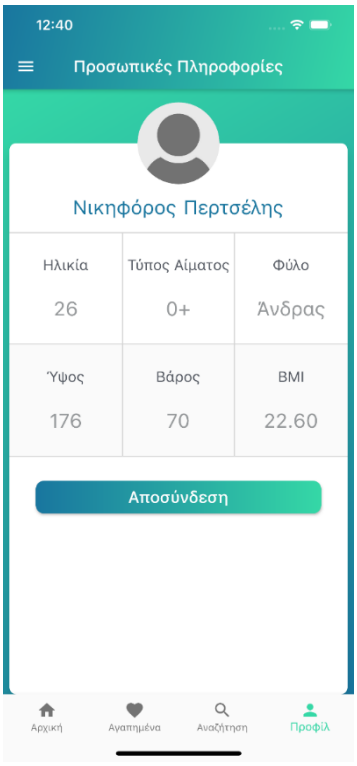
Ο εγγεγραμμένος ασθενής μπορεί να περιηγηθεί στο προφίλ του από το κεντρικό αλλά και το πλαϊνό μενού. Στην οθόνη αυτή, παρουσιάζονται οι προσωπικές του πληροφορίες και το κουμπί της αποσύνδεσης. Συγκεκριμένα φαίνεται η φωτογραφία του, η ηλικία του, η ομάδα αίματος, το φύλο, το ύψος, το βάρος και ο δείκτης μάζας του σώματος.

Ο χρήστης μπορεί να επεξεργαστεί την φωτογραφία του πατώντας επάνω στην ήδη υπάρχουσα, επιλέγοντας στη συνέχεια μία φωτογραφία μέσα από τη συσκευή του. Επιπλέον, μπορεί να τροποποιήσει το ύψος και το βάρος του, πατώντας επάνω στο αντίστοιχο πεδίο και επιλέγοντας εν συνεχεία την τιμή που επιθυμεί.

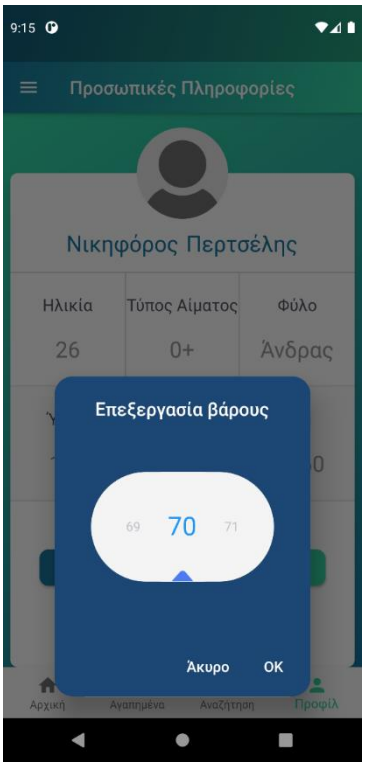
Στο κάτω μέρος της οθόνης βρίσκεται το κουμπί «Αποσύνδεση», όπου πατώντας το, ο χρήστης αποσυνδέεται από την εφαρμογή και πλοηγείται εκ νέου στην οθόνη σύνδεσης.



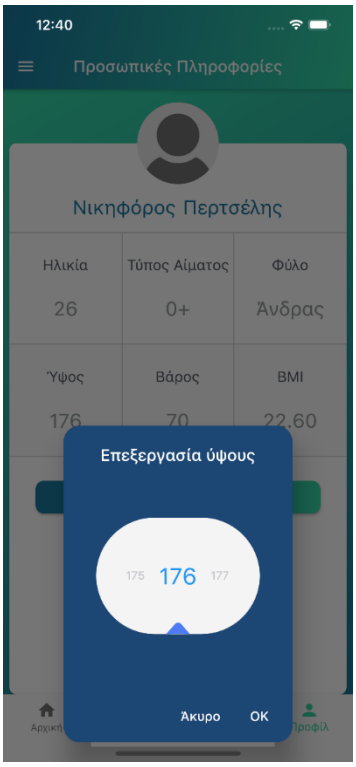
Εικόνα 4.25 Οθόνη του προφίλ του ασθενή στο Android



Εικόνα 4.26 Οθόνη του προφίλ του ασθενή στο iOS



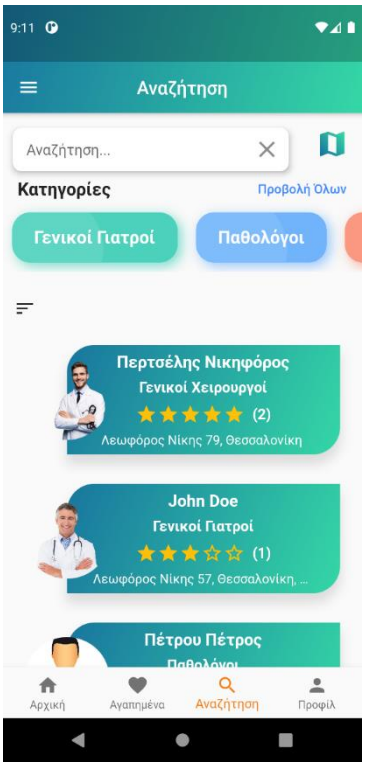
Εικόνα 4.27 Επεξεργασία των προσωπικών πληροφοριών του ασθενή στο Android



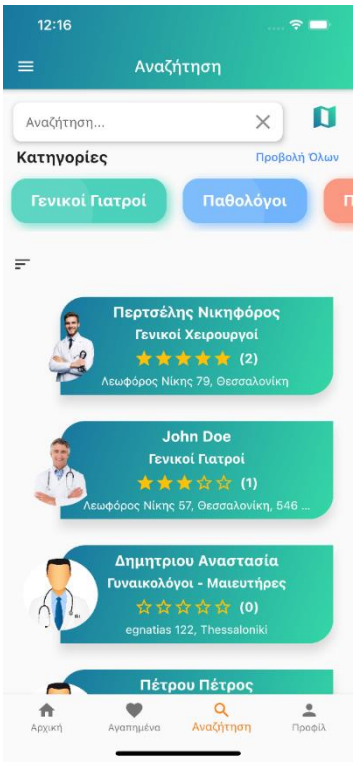
Εικόνα 4.28 Επεξεργασία των προσωπικών πληροφοριών του ασθενή στο iOS

4.11 Οθόνη αναζήτησης των γιατρών

Σε αυτή την οθόνη εμφανίζεται μία λίστα από όλους τους γιατρούς, ταξινομημένοι με βάση τον μέσο όρο της συνολικής βαθμολογίας των αξιολογήσεων που έχουν λάβει.

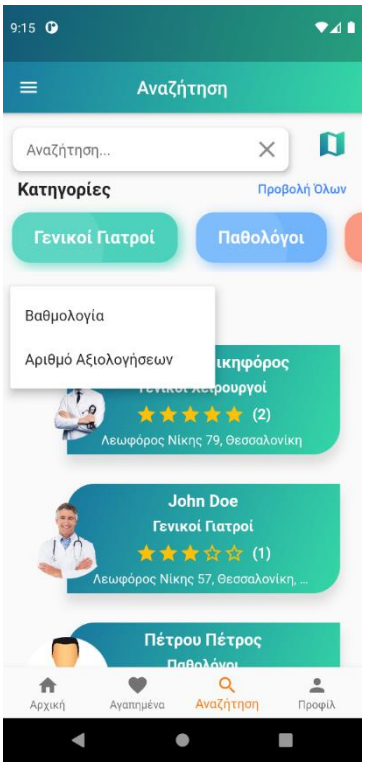


Εικόνα 4.29 Οθόνη αναζήτησης γιατρών στο Android

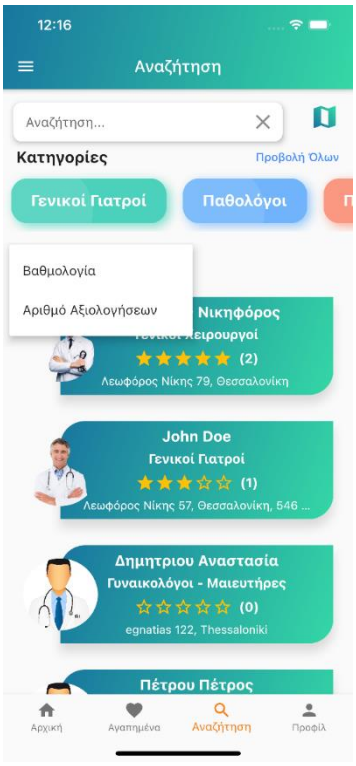


Εικόνα 4.30 Οθόνη αναζήτησης γιατρών στο iOS

Ο χρήστης μπορεί να τροποποιήσει και να αλλάξει την προβολή της λίστας των γιατρών και να την ταξινομήσει με βάση τον συνολικό αριθμό των αξιολογήσεων κάθε γιατρού.



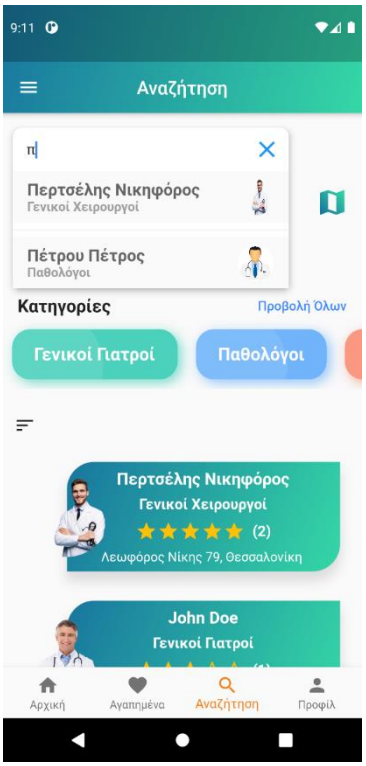
Εικόνα 4.31 Ταξινόμηση των γιατρών στο Android



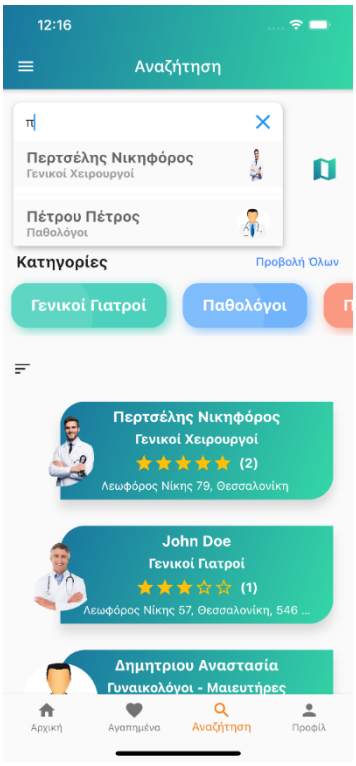
Εικόνα 4.32 Ταξινόμηση των γιατρών στο iOS

Παράλληλα, υπάρχουν άλλοι τρεις τρόποι προκειμένου ο χρήστης να αναζητήσει έναν γιατρό.

Ο εγγεγραμμένος χρήστης μπορεί να αναζητήσει γιατρούς με βάση το ονοματεπώνυμό τους. Στην οθόνη της αναζήτησης γιατρών, υπάρχει ένα πεδίο αναζήτησης. Ο χρήστης πληκτρολογεί το ονοματεπώνυμο του γιατρού. Το σύστημα πραγματοποιεί αναζήτηση με βάση το κείμενο που έχει εισάγει ο χρήστης και εμφανίζει μία αναπτυσσόμενη λίστα η οποία περιέχει τους γιατρούς που το επίθετο ή το όνομά τους αντιστοιχεί στο κείμενο που έχει πληκτρολογήσει ο χρήστης.

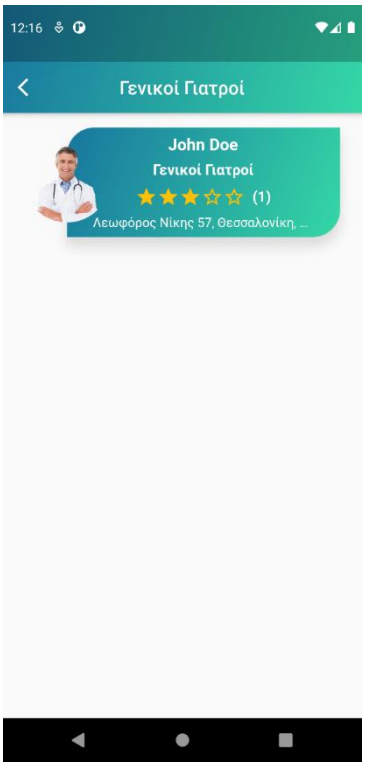


Εικόνα 4.33 Αναζήτηση γιατρών με βάση το όνομα στο Android

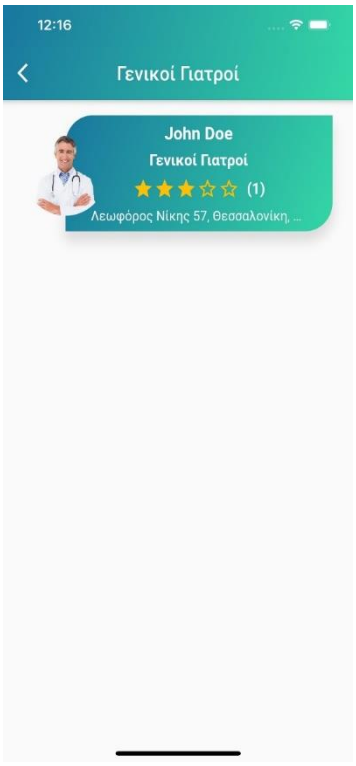


Εικόνα 4.34 Αναζήτηση γιατρών με βάση το όνομα στο iOS

Η τρίτη περίπτωση της αναζήτησης των γιατρών είναι με βάση τις ειδικότητες. Η οθόνη της αναζήτησης γιατρών περιέχει μία οριζόντια λίστα με τις πιο συνηθισμένες ειδικότητες γιατρών. Ο χρήστης επιλέγοντας την ειδικότητα που επιθυμεί, πλοηγείται στην οθόνη της αντίστοιχης ειδικότητας, όπου παρουσιάζεται μία λίστα με όλους τους γιατρούς οι οποίοι έχουν την αντίστοιχη ειδικότητα.

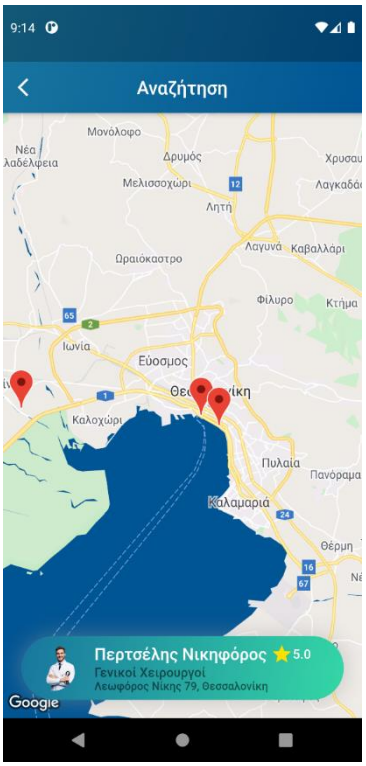


Εικόνα 4.35 Αναζήτηση γιατρών με βάση την ειδικότητα στο Android

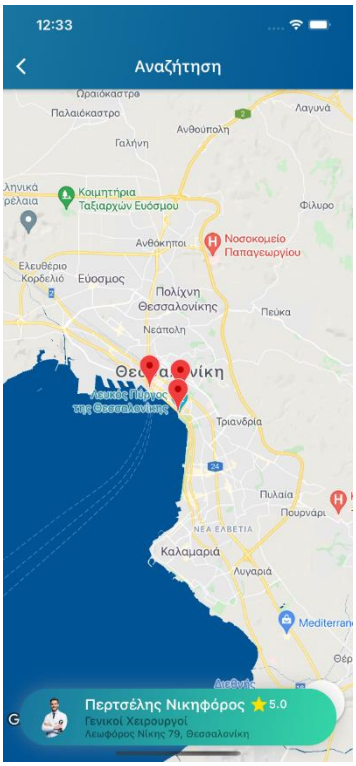


Εικόνα 4.36 Αναζήτηση γιατρών με βάση την ειδικότητα στο iOS

Η τέταρτη και τελευταία μέθοδος αναζήτησης των γιατρών, πραγματοποιείται μέσω του χάρτη. Ο χρήστης επιλέγει το εικονίδιο του χάρτη το οποίο βρίσκεται στο επάνω δεξιό μέρος της οθόνης αναζήτησης. Με το πάτημα του εικονιδίου, ο χρήστης πλοηγείται στην οθόνη του χάρτη. Σε αυτή την οθόνη, εμφανίζονται ορισμένοι δείκτες επάνω στον χάρτη, τα οποία αντιστοιχούν στις διευθύνσεις όλων των γιατρών. Πατώντας επάνω σε ένα δείκτη, εμφανίζεται μία αναδυόμενη κάρτα η οποία περιέχει τις βασικές πληροφορίες του γιατρού, ονοματεπώνυμο, μέσος όρος αξιολογήσεων, ειδικότητα και διεύθυνση.



Εικόνα 4.37 Οθόνη αναζήτησης γιατρών στον χάρτη στο Android



Εικόνα 4.38 Οθόνη αναζήτησης γιατρών στον χάρτη στο iOS

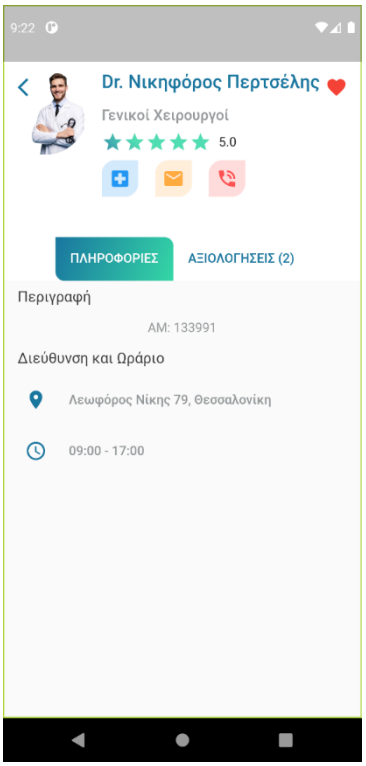
4.12 Οθόνη του προφίλ ενός γιατρού

Όταν ο εγγεγραμμένος χρήστης επιλέξει έναν γιατρό τότε πλοηγείται στην οθόνη του προφίλ του επιλεγμένου γιατρού. Στην οθόνη αυτή, εμφανίζονται το ονοματεπώνυμο του γιατρού, η ειδικότητά του και ο μέσος όρος των αξιολογήσεών του. Δίπλα εμφανίζεται ένα εικονίδιο σε σχήμα καρδίας, το οποίο χρησιμοποιείται για την προσθήκη του γιατρού στα αγαπημένα του χρήστη.

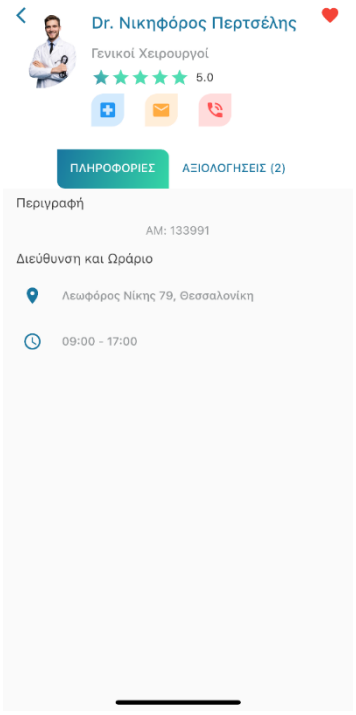
Επιπλέον, παρακάτω υπάρχουν τρία βασικά εικονίδια, τα οποία αντιστοιχούν στην δημιουργία ενός ραντεβού, στην αποστολή email και στην τηλεφωνική κλήση αντίστοιχα.

Επιπροσθέτως, η οθόνη περιέχει δύο καρτέλες, την καρτέλα «Πληροφορίες» και την καρτέλα «Αξιολογήσεις». Η καρτέλα των πληροφοριών περιέχει τις βασικές πληροφορίες του γιατρού, Συγκεκριμένα, παρουσιάζεται η περιγραφή του γιατρού, η διεύθυνσή του και το ωράριο εργασίας του γιατρού.

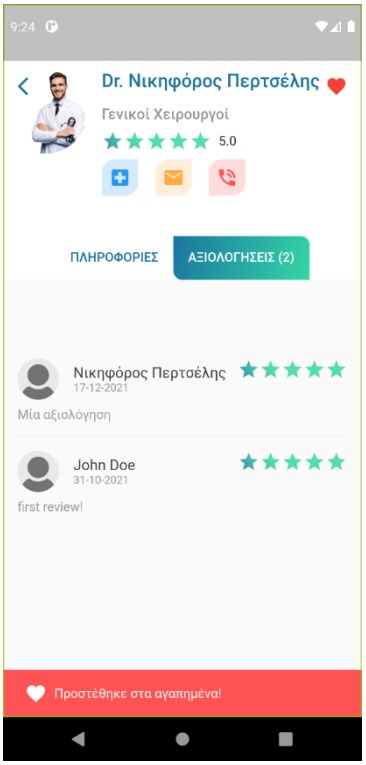
Η καρτέλα των αξιολογήσεων, εμφανίζει μία λίστα με όλες τις αξιολογήσεις που έχουν πραγματοποιηθεί και αφορούν τον επιλεγμένο γιατρό.



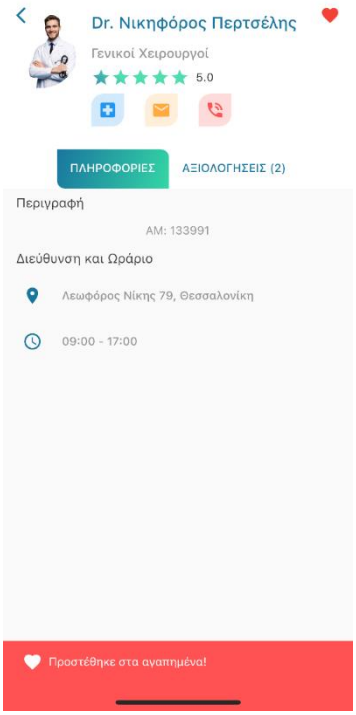
Εικόνα 4.39 Προβολή ενός γιατρού στο Android



Εικόνα 4.40 Προβολή ενός γιατρού στο iOS



Εικόνα 4.41 Προσθήκη γιατρού στα αγαπημένα στο Android



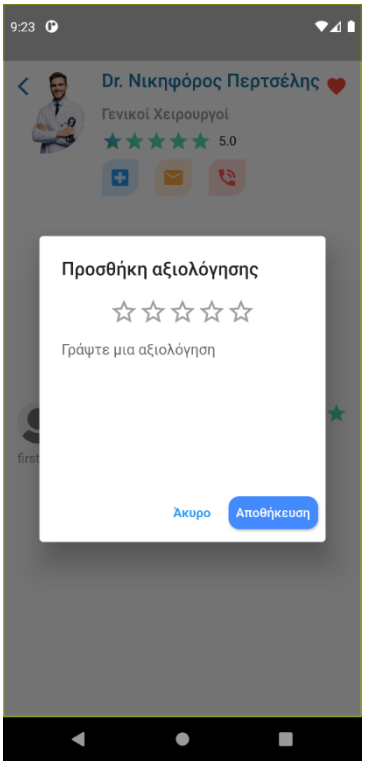
Εικόνα 4.42 Προσθήκη γιατρού στα αγαπημένα στο iOS

4.13 Δημιουργία αξιολόγησης ενός γιατρού

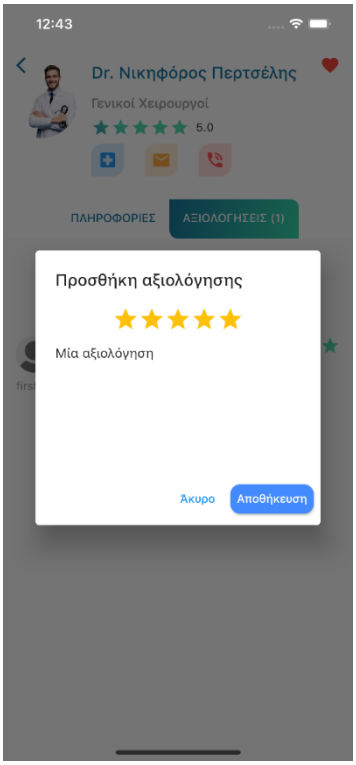
Όταν επιλέγεται η καρτέλα των αξιολογήσεων, εκτός από τη λίστα όλων των αξιολογήσεων, υπάρχει το κουμπί «Προσθήκη αξιολόγησης» το οποίο χρησιμοποιείται για την δημιουργία μίας αξιολόγησης

προς τον επιλεγμένο γιατρό. Αυτό το κουμπί δεν είναι ορατό, όταν ο εγγεγραμμένος χρήστης έχει αξιολογήσει ήδη τον γιατρό.

Πατώντας το κουμπί για την προσθήκη μίας αξιολόγησης, εμφανίζεται μία αναδυόμενη φόρμα. Ο χρήστης επιλέγει τα αστέρια τα οποία αντιστοιχούν στην βαθμολογία από ένα έως πέντε. Προαιρετικά μπορεί να εισάγει ένα κείμενο στην αξιολόγησή του, το οποίο να περιγράφει την εμπειρία και τις εντυπώσεις του για τον γιατρό.



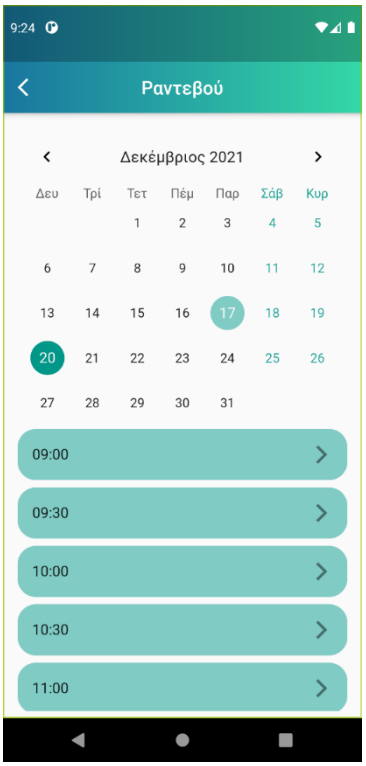
Εικόνα 4.43 Δημιουργία αξιολόγησης ενός γιατρού στο Android



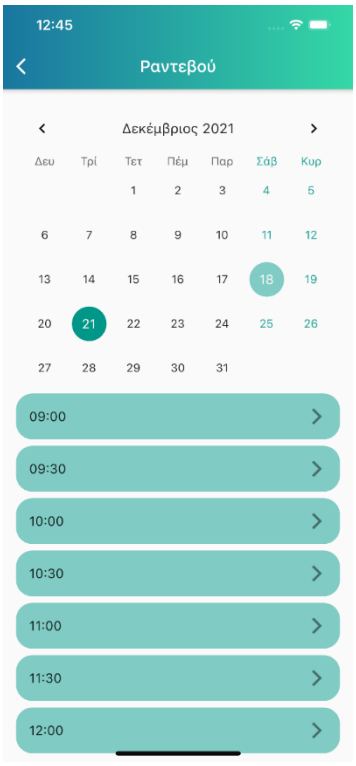
Εικόνα 4.44 Δημιουργία αξιολόγησης ενός γιατρού στο iOS

4.14 Προβολή των διαθέσιμων ημερομηνιών ενός γιατρού και δημιουργία ενός ραντεβού

Όπως προαναφέρθηκε, το μπλε εικονίδιο χρησιμοποιείται για την δημιουργία ενός ραντεβού. Πατώντας αυτό το εικονίδιο, ο χρήστης πλοηγείται στην οθόνη των διαθέσιμων ραντεβού. Η οθόνη αποτελείται από ένα ημερολόγιο, παρουσιάζοντας μία λίστα με τις διαθέσιμες ώρες προς ραντεβού, για την επιλεγμένη ημέρα.

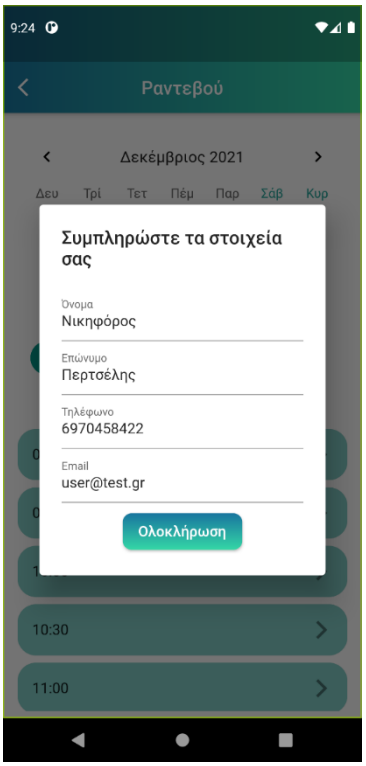


Εικόνα 4.45 Προβολή των διαθέσιμων ημερομηνιών ενός γιατρού στο Android

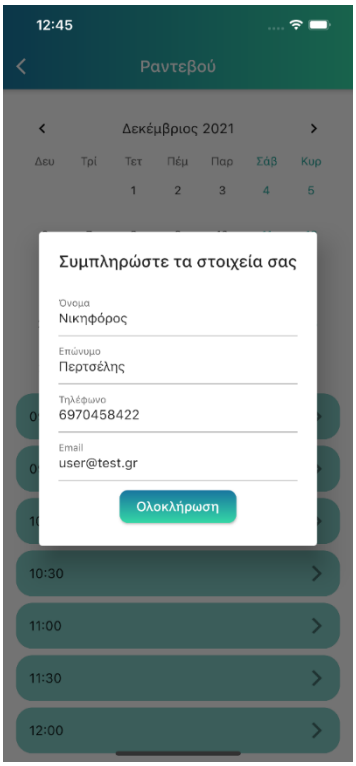


Εικόνα 4.46 Προβολή των διαθέσιμων ημερομηνιών ενός γιατρού στο iOS

Προκειμένου ο εγγεγραμμένος χρήστης να κλείσει ένα ραντεβού για τον επιλεγμένο γιατρό, χρειάζεται αρχικά να επιλέξει μία ώρα που επιθυμεί, από τις διαθέσιμες που υπάρχουν. Επιλέγοντας μία συγκεκριμένη ώρα, η εφαρμογή εμφανίζει μία αναδυόμενη φόρμα. Η φόρμα αυτή, περιέχει τέσσερα πεδία, το όνομα, το επώνυμο, το τηλέφωνο και το email του χρήστη. Τα πεδία είναι αυτόματα προσυμπληρωμένα με βάση τα στοιχεία που έχει καταχωρήσει ο εγγεγραμμένος χρήστης. Επιπλέον η φόρμα περιέχει το κουμπί «Ολοκλήρωση», όπου πατώντας το, δημιουργείται το ραντεβού του χρήστη, ενώ την ίδια στιγμή, μέσω του Firebase, ο γιατρός λαμβάνει μία ειδοποίηση για το ραντεβού που δημιουργήθηκε.



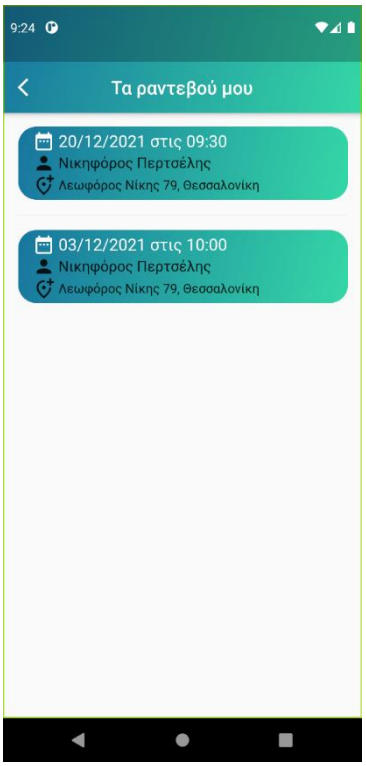
Εικόνα 4.47 Δημιουργία ενός ραντεβού στο Android



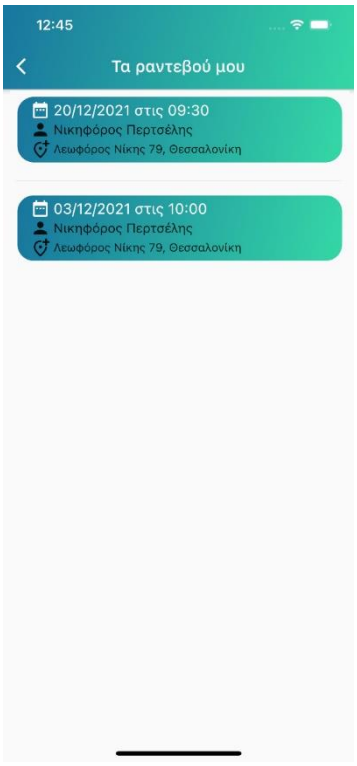
Εικόνα 4.48 Δημιουργία ενός ραντεβού στο iOS

4.15 Προβολή του ιστορικού των ραντεβού ενός ασθενή

Όπως έχει προαναφερθεί, στο πλαϊνό μενού του εγγεγραμμένου ασθενή, μία από τις επτά επιλογές είναι η επιλογή «Τα ραντεβού μου». Πατώντας αυτή την επιλογή, ο χρήστης πλοηγείται στην οθόνη των προσωπικών του ραντεβού. Η οθόνη αυτή αποτελείται από μία λίστα των ραντεβού του εγγεγραμμένου ασθενή, ταξινομημένη ως προς το πιο πρόσφατο ραντεβού.



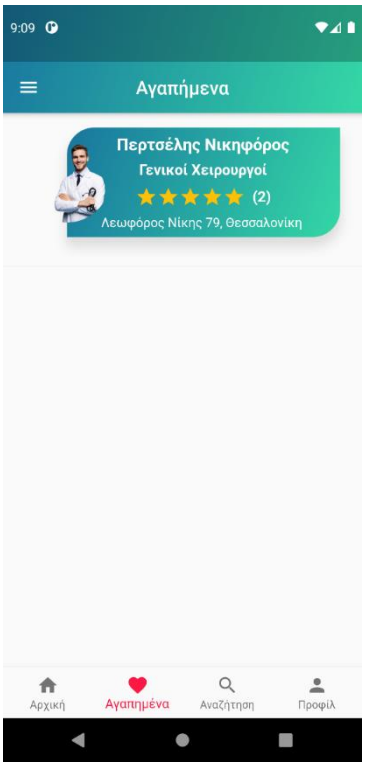
Εικόνα 4.49 Ιστορικό των ραντεβού ενός ασθενή στο Android



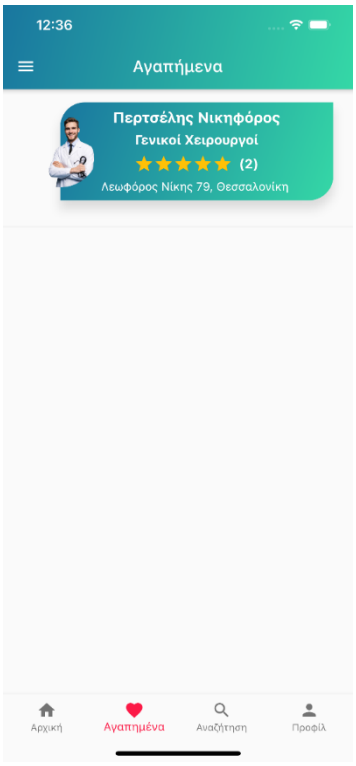
Εικόνα 4.50 Ιστορικό των ραντεβού ενός ασθενή στο iOS

4.16 Οθόνη των αγαπημένων

Ο εγγεγραμμένος ασθενής, έχει τη δυνατότητα να αποθηκεύει γιατρούς στα αγαπημένα του. Όπως παρουσιάστηκε παραπάνω, ο χρήστης τοποθετεί έναν γιατρό στα αγαπημένα του, μέσα από το προφίλ του επιλεγμένου γιατρού. Το κεντρικό αλλά και το πλαϊνό μενού του χρήστη, περιέχουν την επιλογή για την πλοήγηση στην οθόνη των αγαπημένων. Στην οθόνη των αγαπημένων, παρουσιάζεται μία λίστα με τις βασικές πληροφορίες των γιατρών οι οποίοι έχουν τοποθετηθεί στα αγαπημένα του χρήστη.



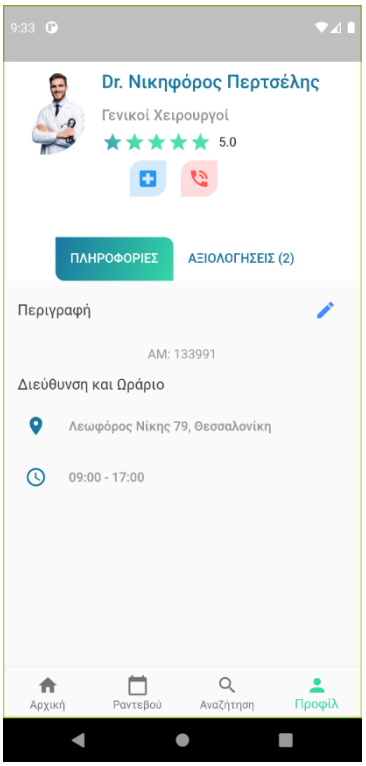
Εικόνα 4.51 Οθόνη των αγαπημένων γιατρών στο Android



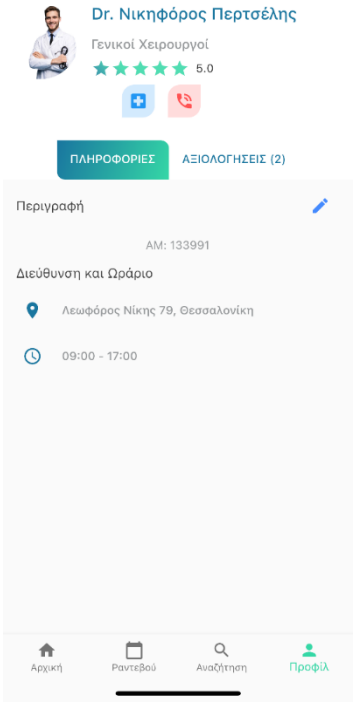
Εικόνα 4.52 Οθόνη των αγαπημένων γιατρών στο iOS

4.17 Προβολή του προσωπικού προφίλ ενός γιατρού

Ο εγγεγραμμένος γιατρός μπορεί να περιηγηθεί στο προφίλ του από το κεντρικό αλλά και το πλαϊνό μενού. Στην οθόνη αυτή, παρουσιάζονται οι προσωπικές του πληροφορίες. Συγκεκριμένα φαίνεται η φωτογραφία του, η ειδικότητά του, η συνολική του βαθμολογία από τις αξιολογήσεις που έχει λάβει, η περιγραφή που έχει ορίσει, η διεύθυνσή του, το ωράριο εργασίας του, οι αξιολογήσεις του, καθώς και δύο εικονίδια που αντιστοιχούν στη διαχείριση των ραντεβού του και το τηλέφωνό του.

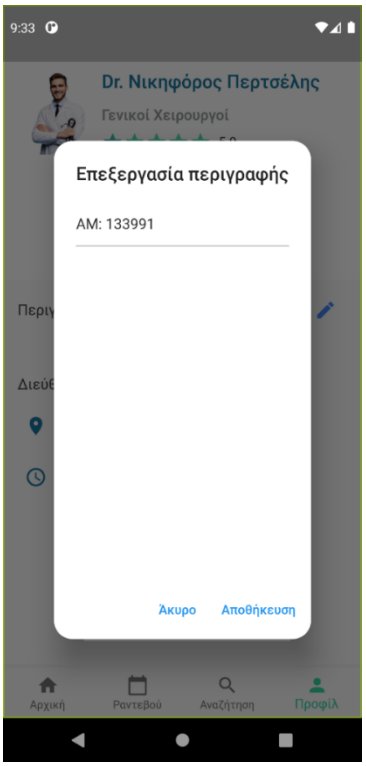


Εικόνα 4.53 Προβολή του προσωπικού προφίλ ενός γιατρού στο Android

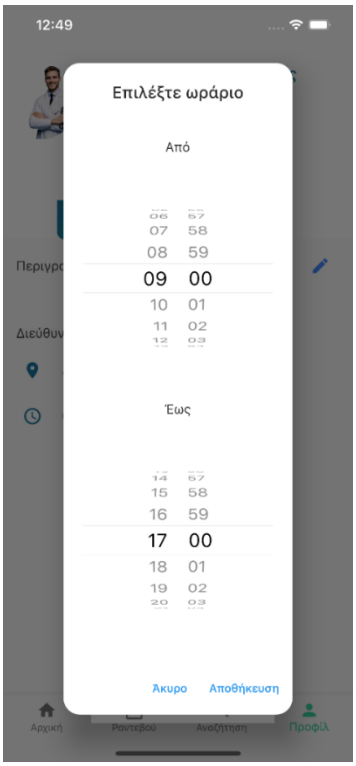


Εικόνα 4.54 Προβολή του προσωπικού προφίλ ενός γιατρού στο iOS

Ο χρήστης μπορεί να επεξεργαστεί την φωτογραφία του πατώντας επάνω στην ήδη υπάρχουσα, επιλέγοντας στη συνέχεια μία φωτογραφία μέσα από τη συσκευή του. Επιπλέον, μπορεί να τροποποιήσει το τηλέφωνό του, την περιγραφή του και το ωράριο εργασίας του. Πατώντας επάνω στο αντίστοιχο πεδίο, εμφανίζεται μία αναδυόμενη φόρμα όπου ο χρήστης τροποποιεί το κάθε πεδίο, όπως ο ίδιος επιθυμεί.



Εικόνα 4.55 Επεξεργασία των προσωπικών πληροφοριών ενός γιατρού στο Android

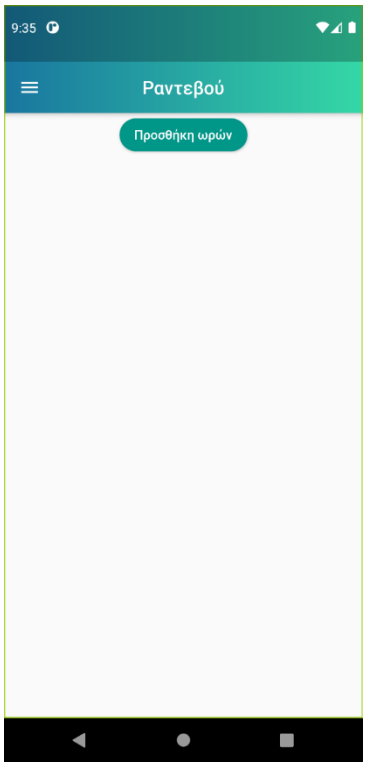


Εικόνα 4.56 Επεξεργασία των προσωπικών πληροφοριών ενός γιατρού στο iOS

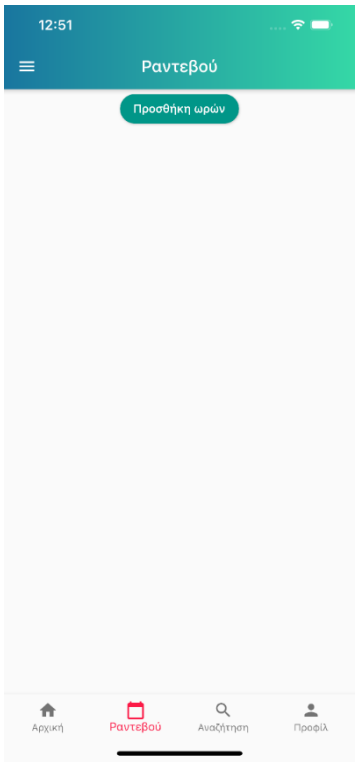
4.18 Αρχικοποίηση των διαθέσιμων ημερομηνιών του γιατρού

Κατά τη δημιουργία του λογαριασμού του, ο εγγεγραμμένος γιατρός, καθορίζει το ωράριο εργασίας του. Για να μπορεί να δέχεται ραντεβού από τους ασθενής, χρειάζεται να ορίσει τις ημέρες κατά τις οποίες θα μπορεί να δέχεται ραντεβού.

Αρχικά, ο εγγεγραμμένος γιατρός πλοηγείται στην οθόνη των ραντεβού. Η οθόνη των ραντεβού θα διαμορφωθεί όταν ο γιατρός ορίσει τις ημέρες και τις ώρες που δέχεται τα ραντεβού του. Στην οθόνη υπάρχει το κουμπί «Προσθήκη ωρών». Πατώντας αυτό το κουμπί, ο χρήστης πλοηγείται σε μία άλλη οθόνη, ώστε να καθορίσει τις ημερομηνίες που επιθυμεί.



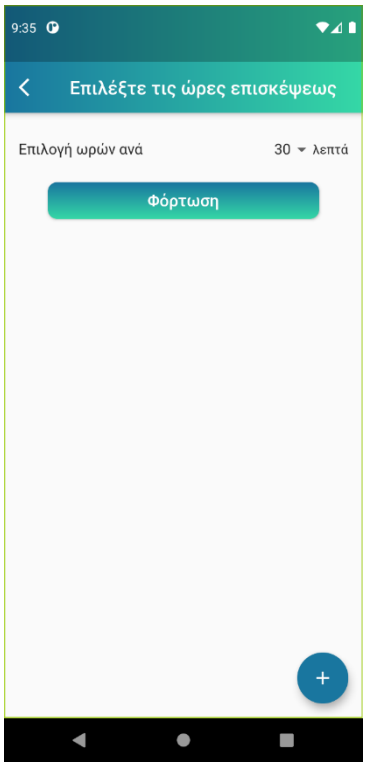
Εικόνα 4.57 Οθόνη των ραντεβού ενός γιατρού, πριν τον καθορισμό των διαθέσιμων ημερών και ωρών, στο Android



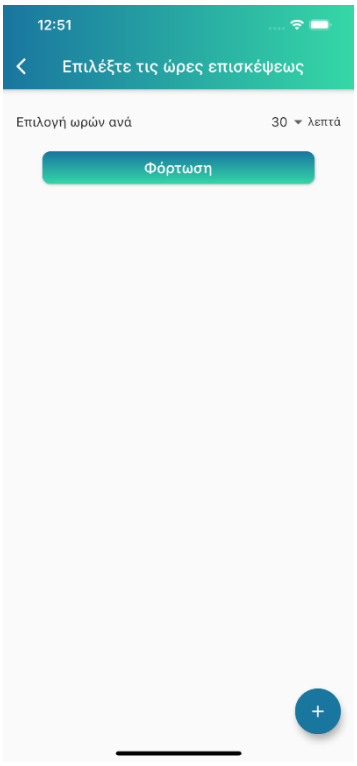
Εικόνα 4.58 Οθόνη των ραντεβού ενός γιατρού, πριν τον καθορισμό των διαθέσιμων ημερών και ωρών, στο iOS

Στην νέα οθόνη, πρώτα, δίνεται η επιλογή στον χρήστη να επιλέξει το χρονικό διάστημα μεταξύ των καθημερινών του ραντεβού. Οι επιλογές που παρέχει η εφαρμογή είναι κάθε 15, 30, 45 ή 60 λεπτά.

Στη συνέχεια, ο χρήστης χρειάζεται να πατήσει το κουμπί «Φόρτωση». Πατώντας αυτό το κουμπί, η εφαρμογή λαμβάνει το ωράριο εργασίας που έχει καθορίσει ο γιατρός και υπολογίζει τις διαθέσιμες ώρες με βάση το ωράριο εργασίας του και το χρονικό διάστημα που επέλεξε παραπάνω.

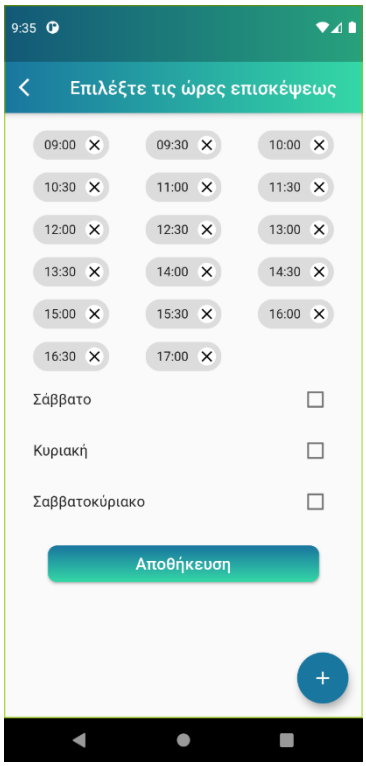


Εικόνα 4.59 Οθόνη στην οποία ο γιατρός καθορίζει το χρονικό διάστημα μεταξύ των καθημερινών του ραντεβού, στο Android

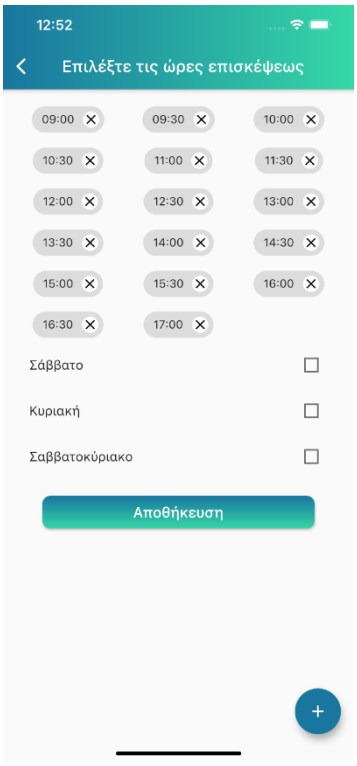


Εικόνα 4.60 Οθόνη στην οποία ο γιατρός καθορίζει το χρονικό διάστημα μεταξύ των καθημερινών του ραντεβού, στο iOS

Έπειτα, η οθόνη εμφανίζει τις διαθέσιμες ώρες που υπολογίστηκαν. Ο χρήστης μπορεί να τροποποιήσει τις ώρες αυτές, προσθέτοντας ή αφαιρώντας κάποια ώρα που επιθυμεί. Επίσης, δίνεται η δυνατότητα στον χρήστη να επιλέξει αν θα μπορεί να δέχεται ραντεβού, εκτός των καθημερινών, δηλαδή είτε Σάββατο, είτε Κυριακή είτε και τις δύο ημέρες. Όταν ο χρήστης αποφασίσει και ορίσει τις ημέρες και τις ώρες που επιθυμεί, τότε χρειάζεται να πατήσει το κουμπί «Αποθήκευση». Πατώντας το κουμπί, δημιουργούνται τα διαθέσιμα ραντεβού του χρήστη, σύμφωνα με τις επιλογές του και ο χρήστης πλοηγείται ξανά στην οθόνη των ραντεβού, η οποία έχει διαμορφωθεί.

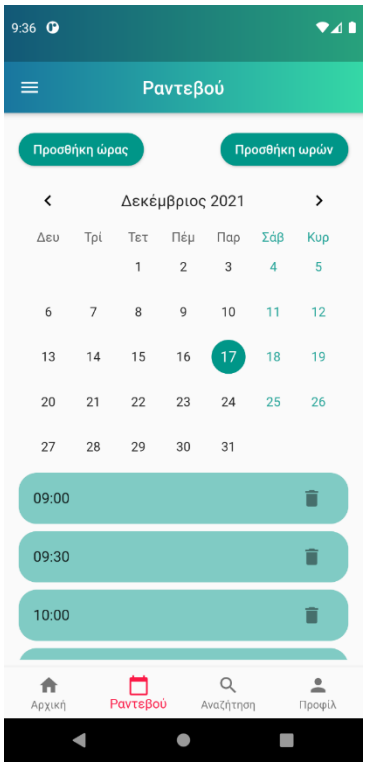


Εικόνα 4.61 Οθόνη στην οποία ο γιατρός καθορίζει τις ημέρες για τις οποίες δέχεται ραντεβού, στο Android

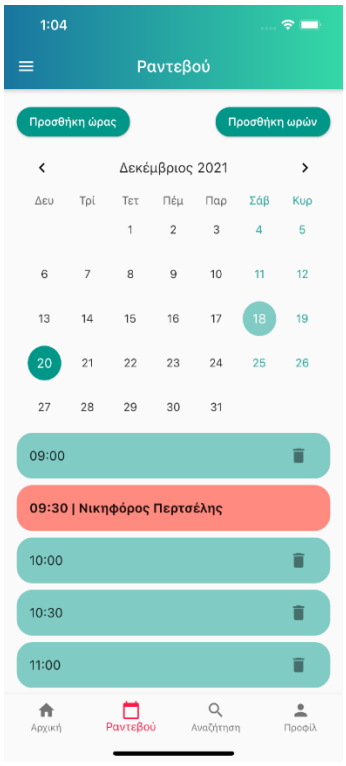


Εικόνα 4.62 Οθόνη στην οποία ο γιατρός καθορίζει τις ημέρες σύμφωνα με τις οποίες δέχεται ραντεβού, στο iOS

Η οθόνη των ραντεβού, πλέον, αποτελείται από ένα ημερολόγιο, παρουσιάζοντας μία λίστα με τα ραντεβού του γιατρού, όσον αφορά την επιλεγμένη ημέρα. Τα διαθέσιμα ραντεβού παρουσιάζονται με ανοικτό πράσινο χρώμα, ενώ τα καθορισμένα ραντεβού, παρουσιάζονται με ανοικτό κόκκινο, περιλαμβάνοντας και το ονοματεπώνυμο του ασθενή.



Εικόνα 4.63 Οθόνη των ραντεβού ενός γιατρού, μετά τον καθορισμό των διαθέσιμων ημερών και ωρών, στο Android



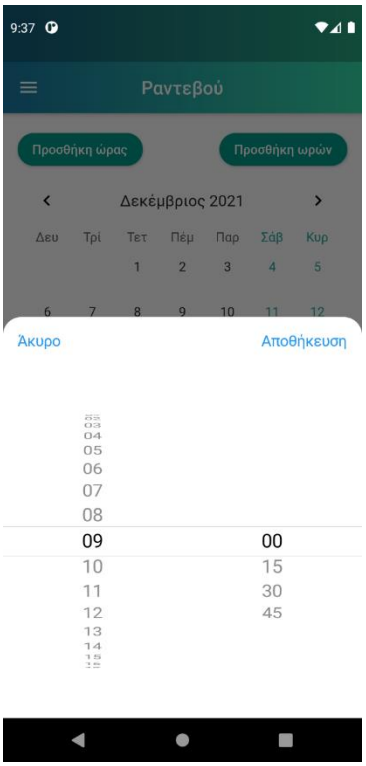
Εικόνα 4.64 Οθόνη των ραντεβού ενός γιατρού, μετά τον καθορισμό των διαθέσιμων ημερών και ωρών, στο iOS

Επιπλέον, ο χρήστης μπορεί να τροποποιήσει το ημερολόγιο των ραντεβού. Η οθόνη των ραντεβού, παρέχει στον χρήστη, τρεις τρόπους επεξεργασίας.

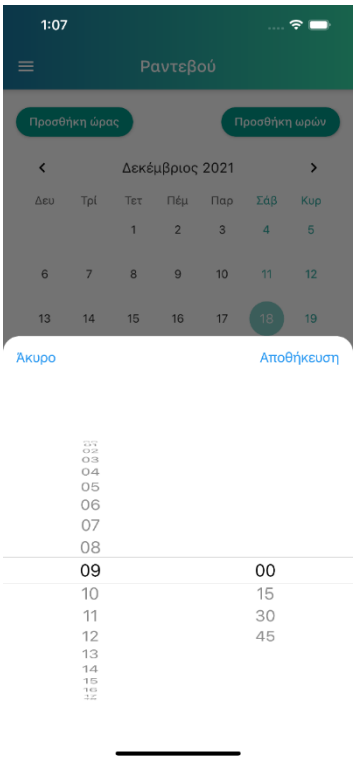
Αρχικά, ο χρήστης μπορεί να διαγράψει κάποιο από τα διαθέσιμα ραντεβού, αλλά δεν μπορεί να τροποποιήσει ένα καθορισμένο ραντεβού.

Επιπλέον, ο χρήστης μπορεί να επιλέξει το κουμπί «Προσθήκη ώρας», το οποίο χρησιμοποιείται για την προσθήκη μίας διαθέσιμης ώρας για την επιλεγμένη ημέρα.

Ταυτόχρονα, υπάρχει το κουμπί «Προσθήκη ωρών», το οποίο χρησιμοποιείται για τον εκ νέου καθορισμό των διαθέσιμων ωρών και ημερών, όπως περιεγράφηκε παραπάνω.



Εικόνα 4.65 Προσθήκη μίας διαθέσιμης ώρας για μία επιλεγμένη ημέρα στο Android



Εικόνα 4.66 Προσθήκη μίας διαθέσιμης ώρας για μία επιλεγμένη ημέρα στο iOS

4.19 Επίλογος

Στο κεφάλαιο αυτό παρουσιάστηκαν και περιεγράφηκαν λεπτομερώς όλες οι οθόνες και οι λειτουργίες της εφαρμογής, τόσο από τη μεριά του ασθενή, όσο και από τη μεριά του γιατρού.

Κεφάλαιο 5ο: Επίλογος

5.1 Σύνοψη

Η παρούσα εργασία είχε σκοπό τον σχεδιασμό και την υλοποίηση μίας εφαρμογής για έξυπνα κινητά, η οποία αφορά την αναζήτηση γιατρών. Η εφαρμογή αποτελείται από δύο βασικά μέρη, στα οποία ο χρήστης συνδέεται ως γιατρός ή ως ασθενής. Η ανάπτυξη της εφαρμογής στοχεύει στην άμεση προβολή και τη διαχείριση του προγράμματος των επισκέψεων του γιατρού αλλά και στην εύκολη και γρήγορη αναζήτηση των γιατρών, δίνοντας έμφαση στις αξιολογήσεις και την βαθμολογία τους, σύμφωνα με τις οποίες ο τελικός χρήστης μπορεί να αποκτήσει μεγαλύτερη αξιοπιστία για τους διαθέσιμους γιατρούς και τελικώς να κλείσει κάποιο ραντεβού. Η αρχική οθόνη περιλαμβάνει δημοσιεύσεις από όλους τους χρήστες, ενισχύοντας, έτσι, την επικοινωνία και την ανταλλαγή απόψεων μεταξύ των χρηστών, σε ένα κοινό ψηφιακό τόπο συζητήσεως, όπου οι χρήστες μπορούν να αλληλοεπιδράσουν στις δημοσιεύσεις, μέσω του κουμπιού «μου αρέσει» αλλά και με την υποβολή σχολίων. Παράλληλα, κάθε χρήστης μπορεί να προβάλει το προσωπικό του προφίλ και να επεξεργαστεί τις πληροφορίες του. Επιπρόσθετα, ο εγγεγραμμένος γιατρός, καθορίζει τις διαθέσιμες ημέρες και ώρες κατά τις οποίες δέχεται επισκέψεις. Επιπλέον, κάθε χρήστης μπορεί να αναζητήσει γιατρούς από την οθόνη της αναζήτησης. Η αναζήτηση πραγματοποιείται με τέσσερις διαφορετικούς τρόπους. Συγκεκριμένα, η αναζήτηση επιτυγχάνεται με τη βοήθεια μίας κάθετης λίστας όλων των γιατρών. Εναλλακτικά, ο χρήστης μπορεί να αναζητήσει έναν γιατρό με βάση την ειδικότητα, το ονοματεπώνυμο ή μέσω γεωγραφικού χάρτη. Πραγματοποιώντας την αναζήτηση, οι χρήστες πλοηγούνται στην οθόνη του επιλεγμένου γιατρού και δύναται να ορίσουν μία επίσκεψη.

5.2 Μελλοντικές βελτιώσεις

Για την ανάπτυξη της εφαρμογής, χρησιμοποιήθηκε το Flutter Framework, το οποίο αποτελεί ένα καινούριο και ταυτόχρονα αναπτυσσόμενο εργαλείο. Κατά την υλοποίηση της εφαρμογής, το Flutter ήταν διαθέσιμο μόνο για έξυπνα κινητά Android και iOS. Πλέον, το Flutter υποστηρίζει όλα τα λειτουργικά συστήματα και καθιστά δυνατή την δημιουργία cross platform εφαρμογών για Android, iOS, Web, Windows, macOS και Linux. Συνεπώς, μελλοντικά, η εφαρμογή θα μπορεί να είναι διαθέσιμη σε όλες τις πλατφόρμες.

Επιπλέον, αντί η εφαρμογή να διασπάται σε δύο μέρη, θα μπορούσε να αφορά μόνο τους ασθενής. Σύμφωνα με τα παραπάνω, ένα σημαντικό βήμα για την εξέλιξη της εφαρμογής, θα ήταν η δημιουργία μίας ιστοσελίδας, η οποία θα αφορά αποκλειστικά και μόνο τους γιατρούς. Οι λειτουργίες που σχετίζονται με την δημιουργία του γιατρού, την επεξεργασία των δεδομένων του, ο καθορισμός και η διαχείριση των προσωπικών του ραντεβού, θα μπορούσαν να γίνουν πιο εύχρηστες με την υλοποίηση μίας διαχειριστικής ιστοσελίδας.

ΒΙΒΛΙΟΓΡΑΦΙΑ

[1] T. C. Lathbrige και R. Laganiere, Object-Oriented Software Engineering: Practical Software Development Using UML and Java 2nd Edition, McGraw-Hill Publishing Company, 2004.

[2] K. E. Wiegers και J. Beatty, Software Requirements, WA, United States: Microsoft Press, Div. of Microsoft Corp. One Microsoft Way Redmond, 2013.

[3] A. S. Gillis, «Native app,» [Ηλεκτρονικό]. Available: https://searchsoftwarequality.techtarget.com/definition/native-application-native-app.

[4] «What is Cross-Platform Software?,» [Ηλεκτρονικό]. Available: https://www.bobology.com/public/What-is-CrossPlatform-Software.cfm.

[5] M. L. Napoli, Beginning Flutter: A Hands on Guide to App Development, 2019.

[6] «What is inside the flutter sdk,» [Ηλεκτρονικό]. Available: https://flutter.dev/docs/resources/faq#what-is-inside-the-flutter-sdk .

[7] «DevTools,» [Ηλεκτρονικό]. Available: https://docs.flutter.dev/development/tools/devtools/overview.

[8] «Introduction to widgets,» [Ηλεκτρονικό]. Available: https://docs.flutter.dev/development/ui/widgets-intro.

[9] «Flutter architectural overview,» [Ηλεκτρονικό]. Available: https://flutter.dev/docs/resources/architectural-overview.

[10] «Firebase Authentication,» [Ηλεκτρονικό]. Available: https://firebase.google.com/docs/auth.

[11] «Where do I start with Firebase Authentication?,» [Ηλεκτρονικό]. Available: https://firebase.google.com/docs/auth/where-to-start.

[12] «Manage Users in Firebase,» [Ηλεκτρονικό]. Available: https://firebase.google.com/docs/auth/android/manage-users.

[13] «Cloud Firestore,» [Ηλεκτρονικό]. Available: https://firebase.google.com/docs/firestore.

[14] «Cloud Firestore Data model,» [Ηλεκτρονικό]. Available: https://firebase.google.com/docs/firestore/data-model.

[15] «Supported data types,» [Ηλεκτρονικό]. Available: https://firebase.google.com/docs/firestore/manage-data/data-types.

[16] «Usage and limits,» [Ηλεκτρονικό]. Available: https://firebase.google.com/docs/firestore/quotas.

[17] «Cloud Storage for Firebase,» [Ηλεκτρονικό]. Available: https://firebase.google.com/docs/storage.

[18] «Cloud Storage Pricing,» [Ηλεκτρονικό]. Available: https://firebase.google.com/pricing.

[19] «Firebase Cloud Messaging,» [Ηλεκτρονικό]. Available: https://firebase.google.com/docs/cloud-messaging.

[20] «About FCM messages,» [Ηλεκτρονικό]. Available: https://firebase.google.com/docs/cloud-messaging/concept-options.

[21] «A tour of the Dart language,» [Ηλεκτρονικό]. Available: https://dart.dev/guides/language/language-tour#important-concepts.

[22] «Dart: A language for structured web programming,» [Ηλεκτρονικό]. Available: https://blog.chromium.org/2011/10/dart-language-for-structured.html.

[23] «Dart 1.0: A stable SDK for structured web apps,» [Ηλεκτρονικό]. Available: https://news.dartlang.org/2013/11/dart-10-stable-sdk-for-structured-web.html.

[24] «Why did Flutter choose to use Dart?,» [Ηλεκτρονικό]. Available: https://flutter.dev/docs/resources/faq#why-did-flutter-choose-to-use-dart.

[25] «Microsoft Launches Visual Studio Code, A Free Cross-Platform Code Editor For OS X, Linux And Windows,» [Ηλεκτρονικό]. Available: https://techcrunch.com/2015/04/29/microsoft-shocks-the-world-with-visual-studio-code-a-free-code-editor-for-os-x-linux-and-windows.

[26] «Admin Authentication API Errors,» [Ηλεκτρονικό]. Available: https://firebase.google.com/docs/auth/admin/errors.

[27] «Add Firebase to your Android project,» [Ηλεκτρονικό]. Available: https://firebase.google.com/docs/android/setup.

[28] «Vistual Studio Code,» [Ηλεκτρονικό]. Available: https://code.visualstudio.com .

[29] «Dart Plugin,» [Ηλεκτρονικό]. Available: https://marketplace.visualstudio.com/items?itemName=Dart-Code.dart-code.

[30] «Flutter plugin,» [Ηλεκτρονικό]. Available: https://marketplace.visualstudio.com/items?itemName=Dart-Code.flutter.

[31] «Flutter install,» [Ηλεκτρονικό]. Available: https://docs.flutter.dev/get-started/install.

[32] «Flutter Github,» [Ηλεκτρονικό]. Available: https://github.com/flutter/flutter.

ΠΑΡΑΡΤΗΜΑ Α: Οδηγός εγκατάστασης του Flutter Framework

Το Flutter Framework είναι ένα cross-platform εργαλείο διεπαφής χρήστη που αναπτύχθηκε από την Google για τη δημιουργία native εφαρμογών για κινητά τηλέφωνα. Προκειμένου να καταστεί δυνατή η συγγραφή κώδικα Dart για τη παραγωγή μίας εφαρμογής, είναι απαραίτητη η εγκατάσταση του Flutter SDK. Το Flutter SDK παρέχεται δωρεάν μέσω της επίσημης ιστοσελίδας του [31] ή μέσω ανοικτού κώδικα στη πλατφόρμα Github [32]. Από τη στιγμή που αποθηκευτεί στον υπολογιστή το Flutter SDK, θα πρέπει να ενημερωθεί το μονοπάτι μεταβλητών περιβάλλοντος προκειμένου να είναι δυνατή η εκτέλεση εντολών του Flutter.

ΠΑΡΑΡΤΗΜΑ Β: Οδηγός εγκατάστασης της βάσης δεδομένων

Η εγκατάσταση του Firebase είναι εύκολη και γρήγορη. Αρχικά από τη κονσόλα του Firebase, θα πρέπει να δημιουργηθεί ένα καινούριο έργο (Project). Αφού δοθεί το όνομα του έργου τότε δημιουργείται το Firebase έργο. Στη συνέχεια, το Firebase παρέχει ένα αρχείο για κάθε πλατφόρμα. Το αρχείο *google-services.json* εξάγεται για το Android και το αρχείο *GoogleService-Info.plist* για iOS. Αυτά τα αρχεία χρειάζονται για την αντιστοίχιση και τον συγχρονισμό του Firebase με τον πηγαίο κώδικα, καθώς περιλαμβάνει πληροφορίες που αφορούν τις υπηρεσίες που προσφέρονται.

Προκειμένου να γίνει η απαραίτητη αντιστοίχιση, θα πρέπει το κάθε αρχείο να ενσωματωθεί στον κατάλληλο φάκελο. Συγκεκριμένα, το αρχείο *google-services.json* τοποθετείται στον φάκελο android/app και το αρχείο *GoogleService-Info.plist* στον root φάκελο του έργου.

Παρακάτω παρουσιάζεται η διαδικασία αποθήκευσης του αρχείου στις πλατφόρμες Android και iOS αντίστοιχα [27].

ΠΑΡΑΡΤΗΜΑ Γ: Οδηγός Visual Studio Code

Όπως αναλύθηκε στο προηγούμενο κεφάλαιο, το Visual Studio Code αποτελεί ένα προγραμματιστικό περιβάλλον για την ανάπτυξη, την επεξεργασία και την εκτέλεση πηγαίου κώδικα. Το πρόγραμμα αυτό διατίθεται δωρεάν από την επίσημη ιστοσελίδα του [28].

Από τη στιγμή που εγκατασταθεί το πρόγραμμα στον υπολογιστή, απαιτούνται ορισμένες ενέργειες προκειμένου το πρόγραμμα να μπορεί να συγγράψει και να εκτελέσει κώδικα γραμμένο στη γλώσσα προγραμματισμού Dart, χρησιμοποιώντας το Flutter Framework.

Αρχικά, θα πρέπει να ενεργοποιηθούν οι εξής επεκτάσεις:

- Dart [29]
  Η επέκταση αυτή απαιτείται για την υποστήριξη της γλώσσας προγραμματισμού Dart, παρέχοντας όλα τα απαραίτητα εργαλεία για την ανάπτυξη, την επεξεργασία και την εκτέλεση πηγαίου κώδικα.

- Flutter [30]

  Η επέκταση αυτή χρειάζεται για την υποστήριξη του Flutter Framework από το Visual Studio Code, προκειμένου να είναι εφικτή η επεξεργασία και η εκτέλεση εφαρμογών για κινητά τηλέφωνα.

Μετά την εγκατάσταση των παραπάνω επεκτάσεων, το περιβάλλον είναι έτοιμο για την δημιουργία ενός έργου το οποίο θα χρησιμοποιεί το Flutter Framework και τη γλώσσα προγραμματισμού Dart.

Τα βήματα για την δημιουργία ενός καινούριου έργου είναι τα εξής:

1. Ο προγραμματιστής πατάει στο εικονίδιο των ρυθμίσεων το οποίο βρίσκεται στη κάτω αριστερή πλευρά.
2. Ο προγραμματιστής επιλέγει την επιλογή της παλέτας εντολών (Command Palette)
3. Από τη λίστα επιλογών, ο προγραμματιστής επιλέγει το “Flutter: New Application Project”
4. Το Visual Studio Code έχει δημιουργήσει ένα νέο έργο