ΣΧΟΛΗ ΜΗΧΑΝΙΚΩΝ

ΤΜΗΜΑ ΜΗΧΑΝΙΚΩΝ ΠΛΗΡΟΦΟΡΙΚΗΣ
ΚΑΙ ΗΛΕΚΤΡΟΝΙΚΩΝ ΣΥΣΤΗΜΑΤΩΝ

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

«Σύστημα Διαχείρισης Αθλητικών Δραστηριοτήτων για android συσκευές»

Του φοιτητή

Τούμπα Παναγιώτη

Αρ. Μητρώου: 144261

Επιβλέπουσα

Ελβίρα-Μαρία Αρβανίτου

10 Σεπτεμβρίου 2023

Τίτλος Π.Ε. **Σύστημα Διαχείρισης Αθλητικών Δραστηριοτήτων για Android συσκευές**

Κωδικός Π.Ε. **23161**

Φοιτητής: **Τούμπας Παναγιώτης**

Εισηγητής: **Ελβίρα-Μαρία Αρβανίτου**

Ημερομηνία ανάληψης: Π.Ε. 21-03-2023

Ημερομηνία περάτωσης Π.Ε. 10-09-2023

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.

Πρόλογος

Ονομάζομαι Τούμπας Παναγιώτης και επέλεξα αυτό το θέμα για την πτυχιακή, μου διότι με ενδιαφέρει το κομμάτι της σωματικής υγείας και της φυσικής κατάστασης του ανθρώπου. Με βάση αυτή μου την προτίμηση, παρακολουθώ την άνοδο των Mobile Health and Fitness εφαρμογών, στην χρονική περίοδο που διανύουμε. Γι'αυτό σκέφτηκα να υλοποιήσω την προσωπική μου Android Fitness Tracking εφαρμογή. Έτσι θα έχω την ευκαιρία να βιώσω τις δυσκολίες που αντιμετωπίζουν οι εταιρείες, που προσφέρουν τέτοιες εφαρμογές στο ευρύ κοινό, καθώς και να κατανοήσω την αγορά εμπορικών Android εφαρμογών. Μιας και έχω ασχοληθεί στον ελεύθερο μου χρόνο με την ανάπτυξη Android εφαρμογών, θα ήθελα να εμβαθύνω περαιτέρω τις γνώσεις για την πλατφόρμα του Android. Να μελετήσω τις πιο πρόσφατες τεχνολογίες που υπάρχουν σε αυτήν. Ακόμα στοχεύω στην μελέτη μιας client-oriented back-end υποδομής που λέγεται Firebase. Στοχεύω να την χρησιμοποιήσω για να απαλλαχτώ από την πολυπλοκότητα μιας κανονικής back-end ανάπτυξης, που περιλαμβάνει στήσιμο της σχεσιακής βάσης δεδομένων, καθώς και την γνώση μιας επιπλέον γλώσσας προγραμματισμού back-end. Έτσι, θα εστιάσω την προσοχή μου, στο να κατακτήσω μια γλώσσα προγραμματισμού front end mobile και να γίνω ειδικός σε αυτήν. Με αυτόν τον τρόπο, θα έχω περισσότερες πιθανότητες να βρω θέση εργασίας σε ένα συγκεκριμένο πεδίο, αφού έχω την αυτοπεποίθηση να πω ότι έχω σχετική εμπειρία και είμαι κατάλληλος για αυτή.

Περίληψη

Στην παρακάτω πτυχιακή εργασία παρουσιάζεται η υλοποίηση μιας Android εφαρμογής, με θέμα την παρακολούθηση της φυσικής δραστηριότητας και την καταγραφή διατροφικών συνηθειών. Αρχικά διενεργήθηκε μια ανασκόπηση της πυροδότησης των έξυπνων τηλεφώνων στο ευρύ κοινό και κατ' επέκταση την άνοδο και την εξέλιξη των Android εφαρμογών που ακολούθησε. Ταυτόχρονα επισημάνθηκε το ενδιαφέρον μου γύρω από την αγορά των εφαρμογών σωματικής υγείας και φυσικής κατάστασης, καθώς και το πάθος μου γύρω από οτιδήποτε αφορά την σωματική και ψυχική εξύψωση του οργανισμού. Πραγματοποίησα μια έρευνα στις υπάρχουσες λύσεις που προσφέρει το Google Play Store, πάνω στο θέμα που με απασχόλησε. Ανακάλυψα ότι οι επιλογές που παρέχονται μπορούν να συνδυαστούν μεταξύ τους και να παρέχουν ένα πιο ολοκληρωμένο πακέτο στον χρήστη που ενδιαφέρεται για την ολότητα της υγείας του. Έτσι βασίστηκα στην δική μου ικανότητα να φτιάξω μια τέτοια εφαρμογή. Στα επόμενα κεφάλαια αναλύω τις σχεδιαστικές μεθόδους και τα εργαλεία που χρησιμοποίησα, για να φέρω το έργο αυτό που είχα στο μυαλό μου εις πέρας. Στις σχεδιαστικές μεθόδους ακολούθησα το MVVM και ως ένα βαθμό το Clean Architecture. Για να αντλώ δεδομένα χρειάστηκα ένα RESTful API για την συμπλήρωση της λίστα τροφίμων. Για το κύριο κομμάτι της εφαρμογής που είναι τα διαγράμματα, εισήγαγα μια third party βιβλιοθήκη σύνθεσης και απεικόνισης γραφημάτων. Αν και με διευκόλυνε η επιλογή αυτής της βιβλιοθήκης, χρειάστηκε αρκετή μελέτη του εγχειριδίου, που παρείχε ο δημιουργός του. Έπειτα δείχνω παραδείγματα πηγαίου κώδικα, παρουσιάζω τις τελικές οθόνες που δημιούργησα και εξηγώ ποιες ανάγκες μπορεί να εκπληρώσει ο χρήστης πλοηγούμενος σε αυτές. Τέλος, αξιολογώ αν το αρχικό μου όραμα επιτεύχθηκε ή όχι. Καταλήγω στο συμπέρασμα ότι το τελικό αποτέλεσμα χρειάζεται ακόμα παραπάνω λειτουργίες και βελτιώσεις στην εμφάνιση της διεπαφής χρήστη, για να πλησιάσει το επιθυμητό. Παρ' όλα αυτά κατάφερα να φτιάξω μια αρκετά καλή βάση, η οποία έχει αμέτρητες προοπτικές εξέλιξης.

«Sport Activity Management System for Android devices»

«Toumpas Panagiotis»

Abstract

In the following thesis, the implementation of an Android application for monitoring physical activity and recording dietary habits is presented. Initially, a review of the proliferation of smartphones in the general public and the subsequent rise and evolution of Android applications is conducted. Concurrently, my interest in the market for health and fitness applications, as well as my passion for anything related to physical and mental well-being, is highlighted. I conducted research into the existing solutions offered by the Google Play Store on the topic that intrigued me. I discovered that the options available could be combined to provide a more comprehensive package for users interested in their overall health. Thus, I relied on my ability to create such an application. In the following chapters, I analyze the design methods and tools I used to bring this project to fruition. In terms of design methods, I followed the MVVM and Clean Architecture to a certain extent. To retrieve data, I needed a RESTful API to populate the food list. For the main part of the application, which is the charts, I introduced a third-party chart composition and rendering library. Although the library facilitated my work, it required a considerable amount of study from the creator's manual. I provide examples of source code, present the final screens I created, and explain the user needs that can be fulfilled by navigating through them. Finally, I evaluate whether my initial vision was achieved or not. I conclude that the final result still requires additional features and improvements in the user interface to approach the desired level of quality. Nevertheless, I managed to create a solid foundation with numerous prospects for further development.

Περιεχόμενα

Πρόλογος iii
Περίληψη iv
Abstract v
Περιεχόμενα vi
Κατάλογος Σχημάτων viii
Συντομογραφίες xi
Κεφάλαιο 1ο: Εισαγωγή 1
1.1 Android 1
1.2 Αθλητικές Δραστηριότητες 5
1.3: Κίνητρο Έρευνας 5
1.4: Δομή Πτυχιακής 7
Κεφάλαιο 2ο: Εισαγωγή στην Εφαρμογή 9
2.1 Παρόμοιες Εφαρμογές / Ανάλυση Ανταγωνισμού 9
2.1.1 MyFitnessPal 9
2.1.2 Sports Tracker Running Cycling 10
2.2: Απαιτήσεις της Εφαρμογής 11
2.2.1 Ευχρηστία της Εφαρμογής 14
2.2.2 Material Design 18
2.2.3 Κατευθυντήριες Γραμμές Jetpack Compose 20
Κεφάλαιο 3: Μεθοδολογία και σχεδιαστικές αποφάσεις 23
3.1: Κεντρική Ιδέα 23
3.1.1 Πρότυπα Σχεδίασης (Design Patterns) 23
3.2: Επιλογή Εργαλείων / Τεχνολογιών / Γλωσσών Προγραμματισμού 26
3.2.1 Android Studio 26
3.2.2 Android SDK 26
3.2.3 Android Gradle 27
3.2.4 Java 27
3.2.5 Kotlin 28
3.2.6 Jetpack Compose 28
3.2.7 GIT 29
3.2.8 GitHub 32

3.2.9 Kotlin Coroutines 32
3.2.10 ProGuard 32
3.2.11 R8 33
3.2.12 Dagger Hilt 34
3.2.13 Vico 34
3.2.14 YCharts 35
3.2.15: BACK-END - Firebase 35
Κεφάλαιο 4ο: Ανάπτυξη Εφαρμογής και Περιγραφή Λειτουργίας 37
4.1 ΑΝΑΠΤΥΞΗ BACK-END 37
4.1.1 ER Diagram 37
4.1.2 Σχεδίαση Βάση Δεδομένων - Προσδιορισμός Οντοτήτων 38
4.2 ΑΝΑΠΤΥΞΗ FRONT-END 41
4.2.1 Πρότυπο Σχεδίασης 41
4.2.2 Android Studio Gradle 42
4.2.3 AndroidManifest 44
4.2.4 Application, Activity και ViewModel Class 46
4.2.5 Jetpack Compose UI 49
4.2.6 Android Resources 52
4.2.7 Σύνδεση εφαρμογής με εξωτερικές πηγές 53
4.2.8 Services 55
4.3 ΟΘΟΝΕΣ ΔΙΕΠΑΦΗΣ ΧΡΗΣΤΩΝ ΚΑΙ ΠΕΡΙΓΡΑΦΗ ΠΕΡΙΠΤΩΣΕΩΝ ΧΡΗΣΗΣ 57
Κεφάλαιο 5ο: Συμπεράσματα και Μελλοντική Εργασία 77
ΒΙΒΛΙΟΓΡΑΦΙΑ 79

Κατάλογος Σχημάτων

Σχήμα 1.1: Ανάπτυξη αγοράς Android 2

Σχήμα 1.2. Δημοσκόπηση σε προγραμματιστές εφαρμογών σε smartphones - Ποσοστά πλατφόρμας απασχόλησης 3

Σχήμα 2.1. MyFitnessPal αρχική οθόνη. 10

Σχήμα 2.2. Sports Tracker αρχική οθόνη 11

Σχήμα 2.3. Διάγραμμα ροής δοκιμής ευχρηστίας. 16

Σχήμα 2.4. Χρωματική αντίθεση κειμένου σε φόντο 17

Σχήμα 2.5. Περιγραφή στοιχείου οθόνης 18

Σχήμα 2.6. Material Design βασικά στοιχεία οθόνης ευρέως χρησιμοποιούμενα. 19

Σχήμα 3.1. Σύνδεση και επικοινωνία μεταξύ των συστατικών του MVC 24

Σχήμα 3.2. Σύνδεση και επικοινωνία μεταξύ των συστατικών του MVVM 25

Σχήμα 3.3. Σημασιολογικό UI δέντρο Layout κόμβων κατά την εκτέλεση των Compose μεθόδων. .. 29

Σχήμα 3.4. Δέντρο UI συστατικών Compose μεθόδων (Composables). 29

Σχήμα 3.5. Δημιουργία διαφορετικών παρακλαδίων για την διόρθωση σφάλματος ή την υλοποίηση νέου χαρακτηριστικού 30

Σχήμα 3.6. Βασικές ενέργειες χρήσης απομακρυσμένου αποθετηρίου Git 31

Σχήμα 3.7. Git δέντρο μέσα από το Android Studio λογισμικό της εφαρμογής Thesis Fitness. 31

Σχήμα 3.8. Η διαδικασία μετατροπής Kotlin κώδικα σε βελτιστοποιημένο κώδικα με Proguard 33

Σχήμα 3.9. Η διαδικασία μετατροπής Kotlin κώδικα σε βελτιστοποιημένο κώδικα με R8 33

Σχήμα 4.1. Πίνακας εγγεγραμμένων χρηστών μέσα στο Firebase Authentication της υπηρεσίας της εφαρμογής 37

Σχήμα 4.2: ER Διάγραμμα της εφαρμογής Fitness 38

Σχήμα 4.3. Η συλλογή των χρηστών μέσα στην Firestore βάση, με επιλεγμένο τον πρώτο χρήστη. .. 39

Σχήμα 4.4. Εγγραφή μιας αθλητικής δραστηριότητας και συγκεκριμένα το περπάτημα. 40

Σχήμα 4.5. Gradle αρχείο Project επιπέδου 42

Σχήμα 4.6. Gradle αρχείο επιπέδου app. Ρυθμίσεις για τις Android εκδόσεις λογισμικού που υποστηρίζει η εφαρμογή. 43

Σχήμα 4.7. Gradle αρχείο app επιπέδου. Εισαγωγή first party βιβλιοθηκών απαραίτητων για την ανάπτυξη του κώδικα της εφαρμογής. 44

Σχήμα 4.8. AndroidManifest, application δομή. 45

Σχήμα 4.9. AndroidManifest. Permissions για την λειτουργία των χαρακτηριστικών που χρησιμοποιούν το λειτουργικό του Android. 45

Σχήμα 4.10. Thesis Fitness App Application Class. 47

Σχήμα 4.11. AppActivity. Βασικό Activity που διαχειρίζεται όλο το UI της εφαρμογής. 48
Σχήμα 4.12. AppViewModel κλάση. Υπεύθυνη για την διαχείριση όλης της εφαρμογής. 49
Σχήμα 4.13. Jetpack Compose NavHost της εφαρμογής. 50
Σχήμα 4.14. Παράδειγμα Composable της Home οθόνης της εφαρμογής. 51
Σχήμα 4.15. Παλέτα χρωμάτων της εφαρμογής 52
Σχήμα 4.16. Android resource Strings file 52
Σχήμα 4.17. Android drawable files 53
Σχήμα 4.18. Set query αποθήκευσης πληροφοριών χρήστη στο Firebase Firestore. 54
Σχήμα 4.19. Get query ανάκτηση πληροφοριών χρήστη από το Firebase Firestore 54
Σχήμα 4.20. Δηλώσεις request στο public REST API για τα τρόφιμα. 54
Σχήμα 4.21. Ασύγχρονη κλήση της μεθόδου του FoodApi. 55
Σχήμα 4.22. Ασύγχρονη σύνδεση REST API δεδομένων με το UI μέσω του ViewModel. 55
Σχήμα 4.23. Υλοποίηση SportSessionService. 56
Σχήμα 4.24. Ενημέρωση 56
Σχήμα 4.25. SportSessionModule. Singleton αντικείμενο που παρέχει ίδιο instance αντικειμένων σε όποιον τα ζητάει. 57
Σχήμα 4.26. Οθόνη Login και παράθυρο επιλογής google λογαριασμού 57
Σχήμα 4.27 Home βασική οθόνη του χρήστη στην εφαρμογή 58
Σχήμα 4.28. Πρώτο βήμα του Wizard της εφαρμογής 59
Σχήμα 4.29. Wizard προαιρετικό βήμα 3 ορισμός ημερήσιων βημάτων και θερμίδων 59
Σχήμα 4.30. Βήμα 4ο του Wizard - Επιλογή αγαπημένων αθλημάτων 60
Σχήμα 4.31. Τελευταίο βήμα του Wizard - Καθορισμός ημερήσιων μακροθρεπτικών στόχων. 60
Σχήμα 4.32. Βασική Home οθόνη της εφαρμογής. 61
Εικόνα 4.33. Η πρόοδος των στόχων του χρήστη φαίνεται με χρώμα. 62
Σχήμα 4.34. Η οθόνη Activity 63
Σχήμα 4.35. Οθόνη ActivityPreCustomization 64
Σχήμα 4.36. Οθόνη Activity λειτουργία αλλαγής αγαπημένων αθλημάτων 64
Σχήμα 4.37. Οθόνη Home μετά την αλλαγή στα αγαπημένα αθλήματα 65
Σχήμα 4.38. Οθόνη Activity με το ημερολόγιο του ιστορικού ανοιχτό 65
Σχήμα 4.39. ActivityPreCustomization 12 λεπτά στόχος διάρκειας πριν ξεκινήσει το PingPong 66
Σχήμα 4.40. Οθόνη ActivitySession με το 3.2.1- Go animation να τελειώνει πριν ξεκινήσει το χρόνομετρο 66
Σχήμα 4.41. Οθόνη ActivitySession σε εξέλιξη 67

Σχήμα 4.42. Η ειδοποίηση που δείχνει το χρονόμετρο όσο η αθλητική συνεδρία τρέχει από το Notification Drawer του Android OS 68
Σχήμα 4.43. Οθόνη ιστορικού για την αθλητική δραστηριότητα. 69
Σχήμα 4.44. Φιλτράρισμα του ιστορικού αθλημάτων με βάση το άθλημα τρέξιμο. 70
Σχήμα 4.45. Η υπόλοιπη ActivityHistory οθόνη. 71
Σχήμα 4.46. Diet οθόνη 72
Σχήμα 4.47. Παράθυρο pop up χειροκίνητης ενημέρωσης κατανάλωσης μικροθρεπτικών στοιχείων. 73
Σχήμα 4.48. Λειτουργία υπολογισμού κατανάλωσης μακροθρεπτικών από την επιλογή φαγητού. 74
Σχήμα 4.50. Οθόνη ιστορικού διατροφής. 76
Σχήμα 4.51. Επισκόπηση διατροφής παρελθοντικής ημέρας στην οθόνη Diet. 76
Σχήμα 4.52. Οθόνη Προφίλ. 76

Συντομογραφίες

Π.Ε.	Πτυχιακή Εργασία
A.P.I.	Application Programming Interface
S.D.K.	Software Development Kit
IoT	Internet of Things
OHA	Open Handset Alliance
iOS	iPhone Operating System
VR	Virtual Reality
AR	Augmented Reality
GPS	Global Positioning System
ERD	Entity Relationship Diagram

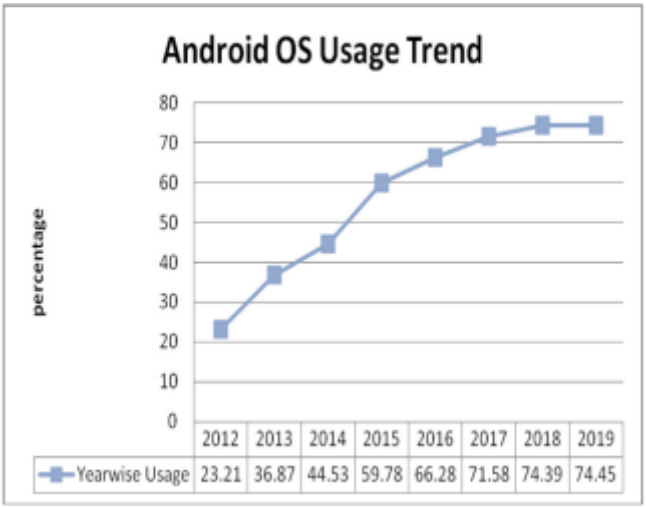
Κεφάλαιο 1ο: Εισαγωγή

Στην εποχή μας, η ανάπτυξη κινητών εφαρμογών για smartphones έχει υποστεί μια εντυπωσιακή μεταμόρφωση, καθώς την υποστηρίζει η ευρεία υιοθέτηση και δημοφιλία μεταξύ των τελικών χρηστών [7]. Αυτό το αυξημένο ενδιαφέρον έχει κατακτήσει την προσοχή των προγραμματιστών κινητών εφαρμογών τα τελευταία χρόνια [8], [9]. Επί του παρόντος, εκατομμύρια κινητές εφαρμογές κατεβάζονται και χρησιμοποιούνται από χρήστες σε ολόκληρο τον κόσμο. Διάφορα καταστήματα εφαρμογών για κινητά, όπως το Google Play Store, το Apple App Store και το Windows Phone Store, παρέχουν τόσο δωρεάν όσο και επί πληρωμή κινητές εφαρμογές στους χρήστες [10]. Οι κινητές εφαρμογές δημιουργούνται κυρίως με τρεις προσεγγίσεις: τοπικές (native), βασισμένες στον ιστό και υβριδικές κινητές εφαρμογές. Η τοπική κινητή εφαρμογή λειτουργεί στο λειτουργικό σύστημα της συσκευής και απαιτεί προσαρμογή σε διάφορες στόχευστες συσκευές. Η ανάπτυξη "τοπικών" εφαρμογών επιτρέπει στους προγραμματιστές να αξιοποιήσουν πλήρως τις δυνατότητες της συσκευής, με αποτέλεσμα καλύτερη απόδοση και διαθεσιμότητα μέσω των αφιερωμένων καταστημάτων εφαρμογών κάθε πλατφόρμας. Ωστόσο, η ανάπτυξη μιας τοπικής εφαρμογής για πολλές πλατφόρμες απαιτεί εμπειρία στις γλώσσες προγραμματισμού, τις διεπαφές προγραμματισμού εφαρμογών (Application Programming Interface—API) και τα Σετ Εργαλείων Ανάπτυξης (Software Development Kit - SDK) κάθε πλατφόρμας, με το αντίστοιχο κόστος (δεξιότητες, χρόνος και έξοδα) [8], [11]. Η δεύτερη προσέγγιση, οι κινητές εφαρμογές ιστού, λειτουργούν σε διακομιστές ιστού και είναι προσβάσιμες μέσω κινητών περιηγητών, προσφέροντας υψηλή φορητότητα σε διάφορες κινητές πλατφόρμες. Αυτή η προσέγγιση μειώνει το κόστος και τον χρόνο ανάπτυξης, επιτρέποντας τη διασυνοριακή χρήση. Παρ' όλα αυτά, οι εφαρμογές "Κινητού Δικτύου" δεν έχουν πρόσβαση σε συσκευασμένα χαρακτηριστικά υλικού της συσκευής, όπως η κάμερα ή το επιταχυνσιόμετρο. Η τρίτη προσέγγιση αφορά τις υβριδικές κινητές εφαρμογές, που εγκαθίστανται στις συσκευές και συσκευάζονται εντός του ελέγχου περιηγητή της πλατφόρμας, λειτουργώντας ουσιαστικά ως "τοπικές" κινητές εφαρμογές ιστού. Αυτές οι κινητές εφαρμογές μπορούν να έχουν πρόσβαση σε χαρακτηριστικά υλικού της συσκευής και είναι διαθέσιμες για λήψη μέσω του καταστήματος διανομής εφαρμογών της πλατφόρμας [8], [11]. Έτσι η εν λόγω ανάπτυξη των κινητών εφαρμογών έπαιξε ένα σημαντικό ρόλο στον σχεδιασμό του οικοσυστήματος του Android. Καθώς οι κινητές εφαρμογές εξελίσσονταν το ίδιο έκανε και το Android. Η εξέλιξη του Android συνδέεται αναπόσπαστα με την εξέλιξη της ανάπτυξης κινητών εφαρμογών. Αντικατοπτρίζει πώς η ευρεία υιοθέτηση και η δημοφιλία των smartphones ώθησαν την καινοτομία στεγανοποίησης μεθοδολογιών ανάπτυξης εφαρμογών. Εκατομμύρια κινητές εφαρμογές, αναπτυγμένες με διάφορες μεθόδους όπως προαναφερόμενες, βρήκαν τον δρόμο τους στα χέρια των χρηστών παγκοσμίως μέσω του Play Store και αυτό αποδεικνύεται από την ιστορία.

1.1 Android

Η ιστορία του Android χαρακτηρίζεται από συνεχή καινοτομία, εξέλιξη και προσαρμογή στον αλλαγόμενο τοπίο της βιομηχανίας κινητής τεχνολογίας. Παραμένει μια κυρίαρχη δύναμη στην αγορά των λειτουργικών συστημάτων κινητών, διαμορφώνοντας τον τρόπο που χρησιμοποιούμε και αλληλεπιδρούμε με την τεχνολογία [23]. Έχει παγκόσμια επιρροή, τροφοδοτώντας δισεκατομμύρια συσκευές, συμπεριλαμβανομένων smartphone, tablet, smart TV και συσκευών IoT (Internet of

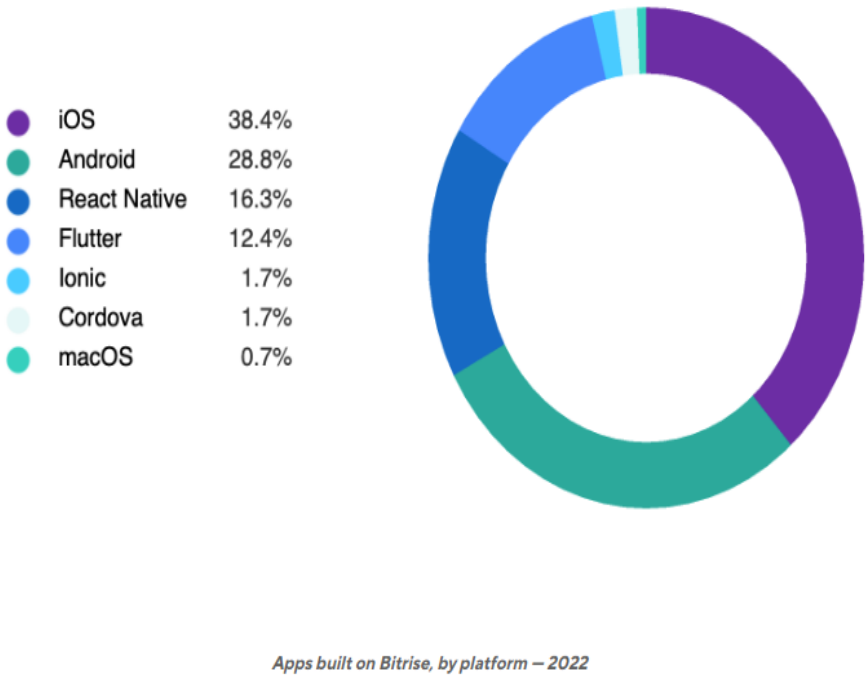
Things). Επίσης, έχει παίξει καθοριστικό ρόλο στη διάθεση προσιτών smartphone στις αναδυόμενες αγορές. Από το 2008 με την πρώτη έκδοση του Android και το λανσάρισμα της πρώτης Android συσκευής (T-Mobile G1) η μερίδα της αγοράς των smartphones που διαθέτουν αυτό το λειτουργικό σύστημα αυξήθηκε από περίπου 4% το 2009 σε ένα εντυπωσιακό 75% το 2019 [31,32]. Αυτή η εκπληκτική αύξηση μπορεί να αποδοθεί σε αρκετά πλεονεκτήματα που προσφέρει έναντι των ανταγωνιστικών λειτουργικών συστημάτων. Ο κύριος λόγος για αυτήν την εντυπωσιακή αύξηση του μεριδίου της αγοράς του Android είναι η υποστήριξη της Ομάδας Ανοικτού Κινητού Τηλεφώνου (Open Handset Alliance - OHA). Η OHA είναι ένα διεθνές κονσόρτσιουμ που αποτελείται από κατασκευαστές συσκευών, κατασκευαστές εξαρτημάτων, αναπτυσσόμενους προγραμματιστές λογισμικού, παρόχους δικτύου κ.λπ., το οποίο στοχεύει στην ανάπτυξη ανοικτών προτύπων για κινητές συσκευές. Ουσιαστικά, όμως, η ανάπτυξη και επιτυχία του Android δεν εξαρτώνται από μια περιορισμένη ποικιλία συσκευών. Η ποικιλία του υλικού καλύπτει τις ανάγκες διαφόρων καταναλωτών με βάση την απόδοση, την προσιτότητα, την αισθητική κ.λπ. [33] (βλ. σχήμα 1.1).



Σχήμα 1.1: Ανάπτυξη αγοράς Android

Η παγκόσμια κοινότητα χτίζει γνώση και εμπειρία πάνω στην χρήση των Android λειτουργικών συστημάτων και εφαρμογών. Την ίδια χρονιά, η Google ξεκίνησε το Android Market (γνωστό τώρα ως Google Play Store), όπου οι χρήστες μπορούσαν να κατεβάζουν και να εγκαθιστούν εφαρμογές για τις συσκευές τους με Android. Αυτό αποτέλεσε ένα σημαντικό βήμα στη δημιουργία ενός ακμάζοντος οικοσυστήματος εφαρμογών για το Android. Το Android βασίζεται σε έναν πυρήνα Linux και είναι γνωστό για τον ανοικτό του χαρακτήρα, επιτρέποντας στους κατασκευαστές και τους προγραμματιστές να τροποποιούν και να προσαρμόζουν το λειτουργικό σύστημα για τις συσκευές τους. Αυτό το ανοιχτό οικοσύστημα έχει συμβάλει στην ευρεία διάδοση της πλατφόρμας. Με βάση τον πυρήνα Linux, η Android Inc. ανέπτυξε το λειτουργικό σύστημα Android, το οποίο πουλήθηκε στην Google. Το λειτουργικό σύστημα Android έχει σχεδιαστεί κυρίως για συσκευές mobile που εφαρμόζουν μια διεπαφή εισαγωγής σε οθόνης αφής. Επίσης έχει διευρυνθεί σε διάφορες μορφές, συμπεριλαμβανομένων του Android Wear (για έξυπνα ρολόγια), του Android TV (για τηλεοπτικές πλατφόρμες) και του Android Auto (για συστήματα ψυχαγωγίας στο αυτοκίνητο). Η Google συνεχίζει να κυκλοφορεί τακτικές ενημερώσεις και νέες εκδόσεις του Android, προσθέτοντας χαρακτηριστικά όπως βελτιωμένη ασφάλεια, βελτιωμένο περιβάλλον χρήστη και άλλα. Επιπλέον, η Project Treble εισήχθη για να διευκολύνει τους κατασκευαστές συσκευών στην παροχή έγκαιρων ενημερώσεων στις συσκευές τους. Ο ανοικτός και προσαρμόσιμος χαρακτήρας του Android, σε συνδυασμό με τη διαθεσιμότητα συσκευών από διάφορους κατασκευαστές, βοήθησε στην κατάκτηση

της θέσης του ως το κυρίαρχο κινητό λειτουργικό σύστημα παγκοσμίως. Ξεπέρασε το iOS σε μερίδιο αγοράς και έγινε το προτιμώμενο λειτουργικό σύστημα για πολλούς χρήστες smartphone. Σήμερα μεταξύ πολλών λειτουργικών συστημάτων για κινητά, το Android είναι ακόμα το στα κορυφαία δύο πιο δημοφιλή λειτουργικά συστήματα, ανταγωνιζόμενο το iOS για συσκευές Apple. Μια έρευνα προγραμματιστών που πραγματοποιήθηκε το 2022 διαπίστωσε ότι το 28,8% των προγραμματιστών κινητής τηλεφωνίας χρησιμοποιούν το Android ως προτιμώμενη πλατφόρμα, πράγμα που το καθιστά δεύτερη δημοφιλέστερη επιλογή [17] (βλ σχήμα 1.2).



Σχήμα 1.2. Δημοσκόπηση σε προγραμματιστές εφαρμογών σε smartphones - Ποσοστά πλατφόρμας απασχόλησης.

Με την πάροδο των χρόνων, το Play Store διευρύνθηκε και πρόσθεσε πολλές κατηγορίες εφαρμογών. Αρχικά, περιλάμβανε κυρίως εφαρμογές που βοηθούσαν στην παραγωγικότητα και εφαρμογές εργαλεία, αλλά γρήγορα διεύρυνε το φάσμα των κατηγοριών της. Οι κατηγορίες αυτές είναι:

- *Παιχνίδια:* Αυτή η κατηγορία περιλαμβάνει μια ευρεία επιλογή από κινητά παιχνίδια, κυμαίνοντας από απλά παιχνίδια και παζλ έως δράσης και παιχνίδια πολλαπλών παικτών.
- *Διασκέδαση:* Αυτή η κατηγορία περιλαμβάνει εφαρμογές για τη ροή μουσικής, ταινιών, τηλεοπτικών εκπομπών και άλλων μορφών ψυχαγωγίας. Μπορεί επίσης να περιλαμβάνει εφαρμογές εικονικής πραγματικότητας (Virtual Reality - VR) και επαυξημένης πραγματικότητας (Augmented Reality - AR).
- *Κοινωνικά:* Οι εφαρμογές κοινωνικών μέσων, οι εφαρμογές μηνυμάτων και τα εργαλεία επικοινωνίας βρίσκονται σε αυτήν την κατηγορία, επιτρέποντας στους χρήστες να συνδέονται με φίλους και οικογένεια.

- *Νέα και Περιοδικά*: Οι εφαρμογές σε αυτήν την κατηγορία παρέχουν πρόσβαση σε άρθρα ειδήσεων, περιοδικά και ψηφιακές εκδόσεις, επιτρέποντας στους χρήστες να παραμένουν ενημερωμένοι για τα τρέχοντα γεγονότα και θέματα ενδιαφέροντος.
- *Υγεία και Φυσική Κατάσταση*: Αυτές οι εφαρμογές βοηθούν τους χρήστες να παρακολουθούν τα ρουτίνες φυσικής κατάστασης, να παρακολουθούν μετρήσεις υγείας και να έχουν πρόσβαση σε προγράμματα προπόνησης και πληροφορίες διατροφής.
- *Ταξίδια και Πλοήγηση*: Οι εφαρμογές ταξιδιού και πλοήγησης προσφέρουν χαρακτηριστικά όπως χάρτες, πλοήγηση GPS, κρατήσεις πτήσεων, κρατήσεις ξενοδοχείων και οδηγοί ταξιδιού.
- *Εκπαίδευση*: Οι εκπαιδευτικές εφαρμογές καλύπτουν μια ευρεία γκάμα θεμάτων και ηλικιακών ομάδων, παρέχοντας υλικό μάθησης, μαθήματα και εργαλεία για φοιτητές και εκπαιδευτικούς.
- *Οικονομία*: Οι εφαρμογές οικονομίας περιλαμβάνουν τραπεζικές υπηρεσίες, διαχείριση προϋπολογισμού, επενδυτικά και εργαλεία διαχείρισης χρημάτων για τη βοήθεια των χρηστών στη διαχείριση των οικονομικών τους.
- *Αγορές*: Οι εφαρμογές αγορών επιτρέπουν στους χρήστες να περιηγούνται και να αγοράζουν προϊόντα online, να αποκτούν πρόσβαση σε προσφορές και να διαχειρίζονται τις λίστες αγορών τους.
- *Τρόπος Ζωής*: Οι εφαρμογές τρόπου ζωής περιλαμβάνουν διάφορα ενδιαφέροντα, όπως ραντεβού, μόδα, μαγειρική και βελτίωση του σπιτιού.
- *Φωτογραφία*: Οι εφαρμογές φωτογραφίας προσφέρουν επεξεργασία φωτογραφιών, βελτιώσεις της κάμερας και φίλτρα για την καταγραφή και επεξεργασία εικόνων.
- *Επιχείρηση*: Οι εφαρμογές επιχείρησης απευθύνονται σε επαγγελματίες και επιχειρηματίες, παρέχοντας εργαλεία για τη διαχείριση έργων, την επικοινωνία και τη γραφειακή αποτελεσματικότητα.
- *Καιρός*: Οι εφαρμογές καιρού παρέχουν πρόγνωση καιρού σε πραγματικό χρόνο, χάρτες ραντάρ και πληροφορίες σχετικά με τον καιρό.
- *Αθλητικά*: Οι εφαρμογές αθλητικών επιτρέπουν την κάλυψη μιας ευρείας γκάμας αθλημάτων και παρέχουν νέα, σκορ, ζωντανές ενημερώσεις και βίντεο αποσπάσματα.
- *Ταξίδι και Τοπικά*: Αυτές οι εφαρμογές παρέχουν πληροφορίες για τοπικές επιχειρήσεις, εστιατόρια, αξιοθέατα και εκδηλώσεις σε συγκεκριμένες τοποθεσίες.
- *Προσαρμογή*: Οι εφαρμογές προσαρμογής περιλαμβάνουν ταπετσαρίες, θέματα και εργαλεία προσαρμογής για την προσαρμογή της εμφάνισης της συσκευής Android.
- *Βιβλία και Αναφορά*: Αυτή η κατηγορία περιλαμβάνει ηλεκτρονικά βιβλία, ηχητικά βιβλία, λεξικά και αναφορικά υλικά για ανάγνωση και μάθηση.
- *Κόμικς*: Οι αναγνώστες κόμικς και ψηφιακά κόμικς μπορούν να βρεθούν σε αυτήν την κατηγορία για τους λάτρεις των κόμικς.
- *Γονεϊκότητα*: Οι εφαρμογές γονεϊκότητας παρέχουν πόρους και συμβουλές για γονείς, καλύπτοντας θέματα όπως η εγκυμοσύνη, η ανάπτυξη του παιδιού και συμβουλές για τη γονική φροντίδα.
- *Τέχνη και Σχεδιασμός*: Οι εφαρμογές τέχνης και σχεδιασμού είναι κατάλληλες για καλλιτέχνες, σχεδιαστές και δημιουργικά άτομα, προσφέροντας εργαλεία για ψηφιακή τέχνη και σχεδιασμό.
- *Βιβλιοθήκες και Επίδειξη*: Αυτή η κατηγορία μπορεί να περιλαμβάνει βιβλιοθήκες και εφαρμογές επίδειξης για προγραμματιστές και χρήστες που ενδιαφέρονται να δοκιμάσουν νέες τεχνολογίες [24].

1.2 Αθλητικές Δραστηριότητες

Για τους λάτρεις της σωματικής άσκησης, το να παρακολουθούν την εξέλιξη της φυσικής τους κατάστασης ήταν πάντα ουσιώδες. Η ανάπτυξη του διαδικτύου και των κινητών εφαρμογών έκανε αυτά τα εργαλεία προσβάσιμα σε όλους, αυξάνοντας τον ανταγωνισμό στο πεδίο.
Με το ξεκίνημα της πανδημίας και των πρώτων lockdowns το 2020, η βιομηχανία της αθλητικής δραστηριότητας και της φυσικής κατάστασης γνώρισε σημαντικό αντίκτυπο, ίσως περισσότερο από άλλους τομείς [12]. Ως απάντηση σε αυτήν την πρόκληση, γίναμε μάρτυρες της εμφάνισης της διαδικτυακής προπόνησης, της δημοτικότητας των γυμναστηρίων στο σπίτι και της αυξημένης έμφασης στην άσκηση για την ενίσχυση της υγείας του ανοσοποιητικού και της συνολικής ευεξίας. Αυτές οι τάσεις φυσικής κατάστασης παρέμειναν το 2021 και το 2022, με μια συνεχώς διευρυνόμενη σειρά ψηφιακών προσφορών γυμναστικής να γίνεται προσβάσιμη. Οι εξελισσόμενες συνήθειες και προτιμήσεις συνεχίζουν να διαμορφώνουν τη βιομηχανία της γυμναστικής καθώς περάσαμε στην χρόνια του 2023.
Σύμφωνα με μια έρευνα από την αμερικανική κυβέρνηση που διεξήχθη τον Αύγουστο του 2022 [6] εντοπίστηκε ευνοϊκή επιρροή των εφαρμογών φυσικής κατάστασης για κινητά στα επίπεδα σωματικής δραστηριότητας σε μια ομάδα ατόμων που έχουν επίγνωση της υγείας τους και τείνουν προς την τεχνολογία. Τα ποιοτικά δεδομένα υπογράμμισαν περαιτέρω την ακαμψία αυτών των εφαρμογών και συσκευών για κινητά, τονίζοντας την ανάγκη για αυτές τις τεχνολογίες να προσαρμόζονται άμεσα σε εξελισσόμενες καταστάσεις ζωής. Πιο συγκεκριμένα σε σύγκριση με τις περιόδους πριν από την πανδημία, σχεδόν οι μισοί (47,8%) χρησιμοποιούσαν πιο συχνά εφαρμογές για κινητά για λόγους υγείας κατά τη διάρκεια της πανδημίας COVID-19. Το σαράντα τοις εκατό (40,9%) άρχισε να χρησιμοποιεί μια νέα εφαρμογή για κινητά για σκοπούς που σχετίζονται με την υγεία από το ξέσπασμα του COVID-19. Μεταξύ εκείνων που χρησιμοποιούσαν εφαρμογές για σωματική δραστηριότητα, η πλειονότητα τις χρησιμοποίησε για την παρακολούθηση των επιπέδων δραστηριότητας (79,7%) ή για την παρακολούθηση ενός βίντεο άσκησης (60,1%) [6].
Τα άτομα που χρησιμοποιούσαν μια εφαρμογή για κινητά που περιείχε πρόγραμμα παρακολούθησης φυσικής κατάστασης κατά τη διάρκεια της πανδημίας είχαν σχεδόν διπλάσιες πιθανότητες να τηρήσουν τις οδηγίες αερόβιας σωματικής δραστηριότητας σε σύγκριση με μη χρήστες. Τα άτομα που άρχισαν να χρησιμοποιούν μια νέα τέτοια εφαρμογή κατά τον COVID-19 είχαν 1,7 φορές περισσότερες πιθανότητες να τηρήσουν τις οδηγίες αερόβιας σωματικής δραστηριότητας σε σχέση με άτομα που δεν το έκαναν. Τα άτομα που είχαν χρησιμοποιήσει εφαρμογές με προγράμματα παρακολούθησης για κινητά πριν από τον COVID-19 είχαν περισσότερες από 2 φορές περισσότερες πιθανότητες να τηρήσουν τις οδηγίες αερόβιας σωματικής δραστηριότητας από τους μη χρήστες [6].

1.3: Κίνητρο Έρευνας

Έχετε παρατηρήσει τον πρόσφατο πολλαπλασιασμό των εφαρμογών για κινητά; Η τεχνολογία έχει την αξιοσημείωτη ικανότητα να εξελίσσεται με γρήγορους ρυθμούς, υφίσταται αλλαγές περίπου κάθε δύο εβδομάδες. Αυτό το φαινόμενο είναι πραγματικά εντυπωσιακό. Πράγματι, κάθε πτυχή της τεχνολογίας βρίσκεται σε συνεχή ροή και αναμφίβολα θα συνεχίσει να εξελίσσεται στο μέλλον. Με την εισαγωγή νέων εφαρμογών και διαδικασιών, αυτή η τάση γίνεται ένα εγγενές χαρακτηριστικό του σημερινού ψηφιακού τοπίου. Αυτό που μόλις πριν από πέντε χρόνια θεωρούταν αιχμή έχει πλέον αντικατασταθεί από εντελώς διαφορετικές καινοτομίες, σηματοδοτώντας ότι αυτή η τάση θα

συνεχιστεί. Οι ειδικοί και οι μελετητές πιστεύουν ότι η τεχνολογία, ειδικά οι εφαρμογές για κινητά, θα ακολουθήσουν τις ιστορικές τάσεις μέχρι τώρα. Η καινοτομία χρησιμεύει ως κινητήρια δύναμη πίσω από αυτόν τον μετασχηματισμό, προχωρώντας με γρήγορους ρυθμούς. Αυτό οδήγησε στην κυκλοφορία πολλών νέων εφαρμογών που χρησιμοποιούμε επί του παρόντος. Επιπλέον, η διαφορά μεταξύ των σημερινών εφαρμογών και εκείνων των προηγούμενων ετών είναι τόσο μεγάλη όσο η αντίθεση μεταξύ του ήλιου και της σελήνης. Κάποτε οι εφαρμογές έδειχναν απλώς το δικό τους περιεχόμενο, αντί να το προσαρμόσουν στις συγκεκριμένες προτιμήσεις των χρηστών. Στις μέρες μας, η κατάσταση έχει εξελιχθεί σημαντικά. Οι εφαρμογές προσφέρουν πλέον πολλές επιλογές, καθοδηγούμενες από μια διαισθητική κατανόηση που ευθυγραμμίζεται με συνέπεια στις προσδοκίες των χρηστών. Τεχνολογίες όπως τα γυροσκόπια και τα δεδομένα τοποθεσίας έχουν βελτιώσει τη λειτουργικότητα, την καινοτομία και την εμπειρία του χρήστη. Επιπλέον, αυτές οι εφαρμογές δεν περιορίζονται σε μία μόνο ιδέα, καθιστώντας τις μια πιο ελκυστική επιλογή για καταναλωτές με γνώσεις τεχνολογίας. Οι άνθρωποι αναζητούν μια απρόσκοπτη και εξαιρετική εμπειρία εφαρμογής. Όταν σκεφτόμαστε την τεχνολογία, το μυαλό μας στρέφεται αμέσως στις εφαρμογές smartphone. Όλοι χρησιμοποιούμε πολλές εφαρμογές καθημερινά, καθεμία από τις οποίες συμβάλλει στην διευκόλυνση της καθημερινότητας της ζωής μας. Με την έλευση του «Internet of Things», όπου τα πάντα είναι στα χέρια μας, η τεχνολογία έχει ανέβει σε ένα νέο επίπεδο. Οι εφαρμογές αποτελούν αναπόσπαστο μέρος της ζωής μας, στις οποίες βασιζόμαστε για διάφορους σκοπούς, είτε πρόκειται για εφαρμογές χρησιμότητας, είτε για εμπορικές εφαρμογές ή για παιχνίδια στις κινητές συσκευές μας. Η όρεξή μας για αυτές παραμένει ακόρεστη, καθώς λαχταρούμε συνεχώς όλο και περισσότερα. Με αφορμή τους τωρινούς γρήγορους ρυθμούς της ζωής, η επίτευξη μιας αρμονικής ισορροπίας μεταξύ επαγγελματικής και προσωπικής ζωής έχει γίνει σταδιακά πιο απαιτητική. Οι απαιτήσεις των χαοτικών προγραμμάτων, των μακρών μετακινήσεων και του φόρτου εργασίας υψηλής πίεσης μπορούν να επηρεάσουν σημαντικά την ευημερία και την ψυχική υγεία των εργαζομένων. Ωστόσο, χάρη στο κύμα της τεχνολογίας, οι εργαζόμενοι άνθρωποι έχουν πλέον την ευκαιρία να χρησιμοποιήσουν μια ποικιλία εφαρμογών γυμναστικής που μπορούν να τους βοηθήσουν στη διατήρηση των στόχων ευεξίας τους. Υπάρχει ανάγκη δημιουργίας εφαρμογών γυμναστικής που θα προσφέρουν στους χρήστες ένα βολικό, εξατομικευμένο και αποτελεσματικό μέσο για να παραμείνουν σε φόρμα και να προάγουν τη συνολική υγεία. Αυτές οι εφαρμογές πρέπει να είναι εύκολα προσβάσιμες και φιλικές προς το χρήστη. Οι εφαρμογή αυτή που υλοποιήθηκε στην παρούσα εργασία μπορεί να συμβάλει στην ενίσχυση της ευημερίας και της ψυχικής υγείας των εργαζομένων με ποικίλους τρόπους. Ένας από αυτούς είναι η άνεση της έναρξης γυμναστικής και της παρακολούθησης της με λίγα μόνο αγγίγματα στο smartphone, με σειρά προπονήσεων, προκλήσεων και προγραμμάτων φυσικής κατάστασης. Αυτή η ευκολία είναι ιδιαίτερα πολύτιμη για τους υπαλλήλους που κάνουν ταχυδακτυλουργίες με πολυάσχολα προγράμματα ή για όσους έχουν περιορισμένη πρόσβαση σε γυμναστήριο. Οι εφαρμογές γυμναστικής τους δίνουν τη δυνατότητα να ασκούνται από οποιαδήποτε τοποθεσία και ανά πάσα στιγμή, παρέχοντας την απαραίτητη ευελιξία για να διατηρήσουν την φυσική τους κατάσταση. Οι εφαρμογές γυμναστικής γενικότερα μπορούν επίσης να χρησιμεύσουν ως εξαιρετικές πηγές κινήτρων για την μακροχρόνια συνείδηση του ανθρώπου ως προς την σωματική του υγεία. Πολλές από αυτές τις εφαρμογές ενσωματώνουν λειτουργίες που επιτρέπουν στους χρήστες να δημιουργήσουν συνδέσεις με φίλους, οικογένεια ή συνανθρώπους που αγωνίζονται για παρόμοιους στόχους φυσικής κατάστασης. Αυτό καλλιεργεί μια αίσθηση κοινότητας, ευθύνης και έμπνευσης, που μπορεί να βοηθήσει σημαντικά τους χρήστες να τηρήσουν τις φιλοδοξίες τους για τη φυσική κατάσταση. Οι εφαρμογές γυμναστικής επιτρέπουν στους χρήστες να παρακολουθούν την πρόοδό τους και να παρακολουθούν στενά τους στόχους της φυσικής τους κατάστασης. Οι χρήστες μπορούν να καταγράφουν αβίαστα λεπτομέρειες σχετικά με τις

προπονήσεις τους, τις διατροφικές τους επιλογές και διάφορες μετρήσεις υγείας, όπως το βάρος, την αρτηριακή πίεση και τον καρδιακό ρυθμό. Αυτή η ικανότητα τους δίνει τη δυνατότητα να οπτικοποιούν την πρόοδό τους και να κάνουν τις απαραίτητες τροποποιήσεις στο πρόγραμμα φυσικής τους κατάστασης όταν απαιτείται. Η πράξη της παρακολούθησης της προόδου μπορεί επίσης να χρησιμεύσει ως ισχυρό κίνητρο, ωθώντας τους χρήστες να επιμείνουν στην επιδίωξη των στόχων τους. Πολυάριθμες μελέτες έχουν αποκαλύψει ότι η άσκηση έχει ευεργετικά αποτελέσματα στην ψυχική υγεία και οι εφαρμογές Fitness μπορούν να παίξουν ρόλο στην ανακούφιση του στρες και του άγχους. Με την ενσωμάτωση τεχνικών ενσυνειδητότητας και διαλογισμού σε αυτές τις εφαρμογές μπορούμε να βοηθήσουμε τους χρήστες να διαχειριστούν το άγχος και να ενισχύσουν τη συνολική τους αίσθηση ευεξίας. Αυτές οι πρακτικές μπορεί να είναι ιδιαίτερα πολύτιμες για τους υπαλλήλους που αντιμετωπίζουν υψηλά επίπεδα άγχους που προέρχονται από τις επαγγελματικές ή προσωπικές τους συνθήκες. Η εφαρμογή αυτή αποτελεί ένα οικονομικό υποκατάστατο για τις συνδρομές στο γυμναστήριο ή τους προσωπικούς προπονητές. Παρέχει δωρεάν ή φιλικές προς τον προϋπολογισμό επιλογές. Κατά συνέπεια, η διατήρηση της φυσικής κατάστασης γίνεται πιο εφικτή και φιλική προς τον προϋπολογισμό, ανεξάρτητα από τους οικονομικούς περιορισμούς τους. Συμπεραίνουμε πως η παρούσα εφαρμογή μπορεί να βοηθήσει τους χρήστες να ενισχύσουν την καρδιαγγειακή τους φυσική κατάσταση, να χτίσουν δύναμη και αντοχή και να μειώσουν τον κίνδυνο χρόνιων παθήσεων όπως το διαβήτη και η καρδιακή νόσος. Με τη βελτίωση της φυσικής τους κατάστασης, οι εργαζόμενοι μπορούν να δώσουν βελτιωμένη ενέργεια, υψηλότερη αίσθηση ευημερίας και αυξημένη παραγωγικότητα στο χώρο εργασίας τους. Τέλος αποτελεί έναν πολύτιμο πόρο για τη βελτίωση της ευημερίας και της ψυχικής υγείας των υπαλλήλων. Προσφέρει διευκόλυνση, προσαρμογή, κίνητρο, δυνατότητα παρακολούθησης, δωρεάν επιλογές και τα οφέλη της βελτιωμένης φυσικής υγείας.

1.4: Δομή Πτυχιακής

Στο 2^{ο} κεφάλαιο πραγματοποιείται έρευνα σχετικά με τις δημοφιλέστερες και πιο παρόμοιες εφαρμογές στην αγορά. Εξετάζονται οι λειτουργίες τους και τα μειονεκτήματα τους. Έπειτα αντλούντε μέσα από αυτήν την σύγκριση οι απαιτήσεις της εφαρμογής που υλοποίησα. Ακολούθησε μια μελέτη πάνω στις προτεινόμενες κατευθυντήριες γραμμές, στις οποίες πρέπει να δίνεται ιδιαίτερη προσοχή, κατά την ανάπτυξη μια Android εφαρμογής. Αυτές χωρίζονται σε οδηγίες που αφορούν την λειτουργικότητα μιας εφαρμογής και σε αυτές που αφορούν την εμφάνιση μιας εφαρμογής σύμφωνα με το Material Design. Περνώντας στο 3^{ο} κεφάλαιο περιγράφεται η διαδικασία της σχεδίασης της εφαρμογής με βάση τις δυσκολίες που μπορούν να προκύψουν και τις προϋπάρχουσες γνώσεις που διέθετα. Αυτές οι αποφάσεις περιλαμβάνουν την επιλογή γλώσσας προγραμματισμού που ήταν η Kotlin, την επιλογή βάσης δεδομένων που ήταν ο συνδυασμός Firebase Firestore και Firebase Storage και την επιλογή εργαλείων και τεχνολογιών που ήταν απαραίτητα για την ανάπτυξη. Στα εργαλεία δίνονται αναλυτικά όλα τα τμήματα της υποδομής που απαρτίζουν μια android εφαρμογή και υποβοηθούν στην συντήρηση της, καθώς και οι συνδεδεμένες βιβλιοθήκες, είτε της Google, είτε third party που επιτρέπουν στην ταχύτερη ανάπτυξη πολύπλοκων διαδικασιών της διεπαφής χρήστη. Στο 4^{ο} κεφάλαιο αναλύεται όλη η διαδικασία ανάπτυξης της εφαρμογής, η οποία απαρτίζεται από δύο τμήματα. Το πρώτο είναι η ανάπτυξη του backend μαζί με την βάση δεδομένων. Σε αυτό το τμήμα επεξηγώ πως αποφάσισα την οντότητες της βάσης δεδομένων και πως τις σύνδεσα για να έχουν αξία στην εφαρμογή. Το δεύτερο είναι η ανάπτυξη του front-end client που είναι η εφαρμογή. Παρουσιάζω τις οθόνες και πως ο χρήστης αλληλεπιδρά με αυτές και εξηγώ με ποιον τρόπο της

απεικόνισα. Ακόμα δείχνω την σύνδεση με της εφαρμογής με το backend καθώς και με τις διάφορες υπηρεσίες του λειτουργικού συστήματος (όπως το GPS) είναι απαραίτητα για την καταγραφή συγκεκριμένων πληροφοριών χρήσης. Εναποθέτω κάθε περίπτωση χρήσης και την αλληλουχία ενεργειών που πρέπει να ακολουθήσει κάποιος που έχει την εφαρμογή στα χέρια του για να την ολοκληρώσει. Εν κατακλείδι στο 5^{ο} και τελευταίο κεφάλαιο αναλογίζομαι για μελλοντική επεκτασιμότητα της εφαρμογής με βάση την εξέλιξη της πλατφόρμας του Android και των τεχνολογιών που την αποτελούν.

Κεφάλαιο 2ο: Εισαγωγή στην Εφαρμογή

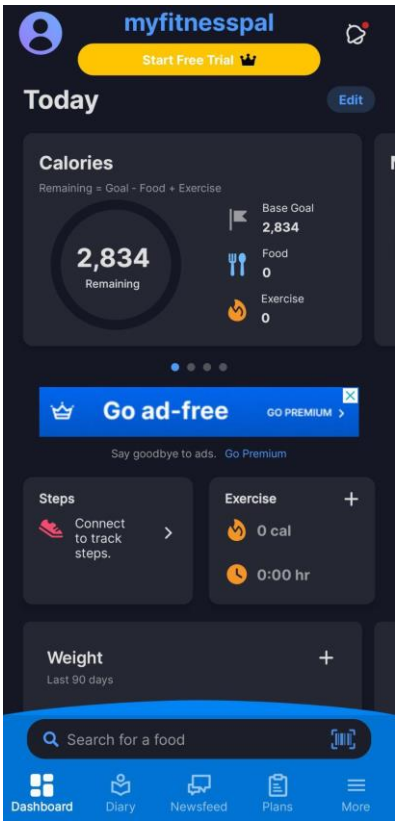
2.1 Παρόμοιες Εφαρμογές / Ανάλυση Ανταγωνισμού

Η αγορά του Fitness στις mobile εφαρμογές χωρίζεται σε δύο κατηγορίες: trackers και ευέλικτες πλατφόρμες γυμναστικής. Το τελευταίο δίνει την δυνατότητα στους χρήστες να ελέγχουν το χρόνο, την ένταση και παρέχει εξατομικευμένα προγράμματα σύμφωνα με τις προτιμήσεις τους. Αυτά τα προσαρμοσμένα πακέτα συνδυάζουν την τεχνητή νοημοσύνη με την ανθρώπινη αλληλεπίδραση για να αναπαράγουν την εμπειρία ενός συμβατικού προσωπικού γυμναστή.

2.1.1 MyFitnessPal

Το MyFitnessPal πρωτοστάτησε στις υπηρεσίες παρακολούθησης φυσικής κατάστασης και υγείας, επιτρέποντας στους χρήστες να δημιουργούν τα δικά τους γεύματα και προπονήσεις. Είναι μια εφαρμογή γυμναστικής που επικεντρώνεται κυρίως στην παρακολούθηση της κατανάλωσης και της δαπάνης θερμίδων, βοηθώντας τους χρήστες να διαχειριστούν το βάρος τους - είτε πρόκειται για απώλεια βάρους, αύξηση ή διατήρησή του. Οι χρήστες μπορούν να καταγράφουν την καθημερινή κατανάλωση τροφίμων και ποτών τους. Διαθέτει μια εκτενή βάση δεδομένων με εκατομμύρια είδη τροφίμων, συμπεριλαμβανομένων επώνυμων και γενικών επιλογών. Συγκεκριμένα πάνω από 6 εκατομμύρια τρόφιμα σε πολλές γλώσσες. Οι χρήστες απλώς καταχωρούν τα γεύματά τους, και η εφαρμογή υπολογίζει αυτόματα την περιεκτικότητα σε θερμίδες, παρέχοντας πληροφορίες σχετικά με τη διατροφική αξία κάθε τροφής. Υπάρχει λειτουργία που επιτρέπει στους χρήστες να προσθέτουν γεύματα από τα αγαπημένα τους εστιατόρια, αρκεί να υπάρχουν στη βάση δεδομένων. Αν ένα γεύμα δεν βρίσκεται στη βάση δεδομένων της εφαρμογής, οι χρήστες έχουν την ευελιξία να δημιουργήσουν τις δικές τους συνταγές. Η εφαρμογή ακόμα υπολογίζει και παρακολουθεί την ημερήσια πρόσληψη και δαπάνη θερμίδων. Οι χρήστες μπορούν να παρακολουθούν συγκεκριμένα την πρόσληψη μακροθρεπτικών συστατικών (υδατάνθρακες, πρωτεΐνες και λίπη) καθώς και μικροθρεπτικά συστατικά (βιταμίνες και μέταλλα). Προσαρμόσιμοι στόχοι είναι διαθέσιμοι για στόχους θερμίδων και θρεπτικών συστατικών. Επιτρέπει στους χρήστες να καταγράφουν διάφορους τύπους σωματικών δραστηριοτήτων και προπονήσεων. Περιλαμβάνει μια βιβλιοθήκη ασκήσεων με εκτιμήσεις θερμίδων για να βοηθήσει τους χρήστες να παρακολουθούν την καύση θερμίδων που σχετίζεται με την άσκηση. Οι χρήστες μπορούν να θέσουν συγκεκριμένους στόχους υγείας και φυσικής κατάστασης, είτε πρόκειται για απώλεια βάρους, συντήρηση ή αύξηση μυών. Η εφαρμογή παρέχει οπτικά εργαλεία παρακολούθησης, διαγράμματα και γραφήματα για την παρακολούθηση της προόδου με την πάροδο του χρόνου. Η παρακολούθηση βάρους βοηθά τους χρήστες να δουν τις τάσεις και τις αλλαγές στο σωματικό τους βάρος. Παρέχει ένα κοινωνικό στοιχείο που επιτρέπει στους χρήστες να συνδέονται με φίλους, να συμμετέχουν σε ομάδες και να μοιράζονται ενημερώσεις και επιτεύγματα. Οι χρήστες μπορούν να προσφέρουν και να λαμβάνουν υποστήριξη, κίνητρα και συμβουλές από την κοινότητα. Η εφαρμογή έχει μια λειτουργία συνταγών, επιτρέποντας στους χρήστες να βρίσκουν, να αποθηκεύουν και να καταγράφουν συνταγές με λεπτομερείς διατροφικές πληροφορίες. Τα εργαλεία προγραμματισμού γευμάτων βοηθούν τους χρήστες να οργανώνουν τα καθημερινά ή εβδομαδιαία γεύματα και τα σνακ τους. Περιλαμβάνει μια λειτουργία παρακολούθησης νερού για την παρακολούθηση της ημερήσιας πρόσληψης νερού. Μπορεί να συγχρονιστεί με δημοφιλείς ιχνηλάτες γυμναστικής και έξυπνα ρολόγια, όπως τα Fitbit, Garmin και Apple Watch, για να εισάγει αυτόματα

δεδομένα δραστηριότητας και βημάτων. Επίσης οι χρήστες μπορούν να σαρώσουν barcodes σε συσκευασμένα τρόφιμα για να τους προσθέσουν γρήγορα στο ημερολόγιο τροφίμων τους, απλοποιώντας τη διαδικασία καταγραφής. Τέλος προσφέρει δυνατότητα προσαρμογής των ημερήσιων στόχων διατροφής, των ονομάτων των γευμάτων και τα μεγεθών των μερίδων τους ώστε να ευθυγραμμίζονται με τις συγκεκριμένες διατροφικές προτιμήσεις και περιορισμούς του χρήστη. Ορισμένα από τα αρνητικά στοιχεία, κυρίως αφορούν την εκτεταμένη βάση δεδομένων τροφίμων και τον αυξημένο αριθμό των λειτουργιών που προαναφέρθηκαν. Υπάρχουν υπερβολικά πολλές επιλογές για το κάθε τρόφιμο που αναζητείται. Ίσως χρειάζεται να βρεθεί ένας τρόπος να παρουσιάζονται πιο κατανοητά τα διάφορα χαρακτηριστικά και λειτουργίες που παρέχονται και να επεξηγείται καλύτερα στους χρήστες η κάθε πολύπλοκη λειτουργία της. Το MyFitnessPal είναι κατά ένα μέρος δωρέαν, παρέχοντας τις βασικές λειτουργίες από αυτές που περιέγραψε, ενώ προσφέρει και μια premium συνδρομή που ξεκλειδώνει τις προχωρημένες λειτουργίες, όπως προηγμένες διατροφικές πληροφορίες, προγραμματισμός γευμάτων και εμπειρία χωρίς διαφημίσεις.

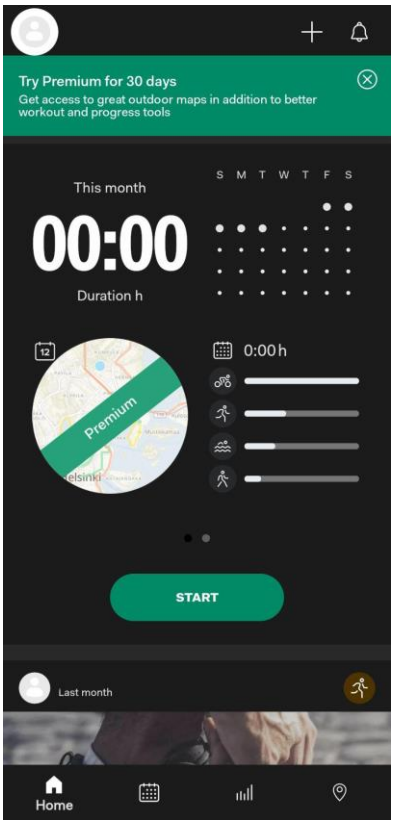


Σχήμα 2.1. MyFitnessPal αρχική οθόνη.

2.1.2 Sports Tracker Running Cycling

Το Sports Tracker Running Cycling περιέχει αθλητικές δραστηριότητες όπως τρέξιμο, ποδηλασία, περπάτημα, πεζοπορία, ποδηλασία βουνού, σκι, κ.α. Καταγράφει την προπόνησή του χρήστη χρησιμοποιώντας ισχυρό GPS και χάρτες, αναλύει από κατανάλωση θερμίδων έως μέση ταχύτητα και υψόμετρο, παρακολουθεί την πρόοδό του χρήστη καθώς εργάζεται προς την επίτευξη των φυσικών του στόχων. Επιτρέπει το διαμοιρασμό της προόδου της προπόνησής και φωτογραφίες με φίλους και ακόλουθους. Καταγράφει και αναλύστε τις επιδόσεις της προπόνησής. Παρακολουθεί την πρόοδο της φυσικής κατάστασης του χρήστη. Συλλέγει δεδομένα σε Ημερολόγιο Προπόνησής για ανάλυση της προόδου. Παρακολουθεί τις θερμίδες που καίγονται, τη μέση ταχύτητα και την ταχύτητα ποδηλασίας,

το ρυθμό τρεξίματος, το υψόμετρο κ.α. Χρησιμοποιεί χάρτες GPS, χρονόμετρο και υπολογισμό αποστάσεων. Εκφωνεί σχόλια κατά τη διάρκεια της προπόνησης. Περιέχει σύστημα κοινότητας που επιτρέπει το follow μεταξύ χρηστών, την θέαση της προόδου άλλου χρήστη, τον διαμοιρασμό φωτογραφιών, αγαπημένων χαρτών και άλλων στοιχείων. Διαθέτει εγγραφή με χρηματικό αντίτυπο για την αφαίρεση των διαφημίσεων, για την άντληση συμπερασμάτων για την εξέλιξη της φυσικής κατάστασης στην διάρκεια του χρόνου, για την εμφάνιση 3D χαρτών, για την παροχή προκαθορισμένων διαδρομών πεζοπορίας, σκι, ποδηλασίας κ.α., για την πρόσβαση στο ιστορικό και άλλων προχωρημένων χαρακτηριστικών. Στα αρνητικά στοιχεία περιλαμβάνονται η φτωχή φωνητική ανατροφοδότηση κατα της διάρκεια της αθλητικής δραστηριότητας και η συνεχής παρακολούθηση της ευαίσθητων δεδομένων του χρήστη, χωρίς να είναι ξεκάθαρο ποια λειτουργία. Τέλος υπάρχουν πολλά παράπονα που αναφέρουν ότι το premium κομμάτι της εφαρμογής εξαπλώνεται ακόμα και σε βασικές λειτουργίες που έχει ανάγκη οποιοσδήποτε χρήστης, είτε είναι καινούργιος στην γυμναστική, είτε έμπειρος στην άθληση και την καταγραφή της.



Σχήμα 2.2. Sports Tracker αρχική οθόνη.

Όλες οι εφαρμογές που προσπαθούν να κλέψουν μερίδιο της αγοράς απαιτούν την χρηματική συνεισφορά του χρήστη για να παρέχουν στοιχεία της εφαρμογής, όπως για παράδειγμα παρακολούθηση της κατανάλωσης μακροθρεπτικών και μικροθρεπτικών στοιχείων.

2.2: Απαιτήσεις της Εφαρμογής

Η παρούσα εργασία στοχεύει να ικανοποιήσει τα κριτήρια μιας ωφέλιμης εφαρμογής καταγραφής της ***αθλητικής δραστηριότητας*** και ***παρακολούθησης διατροφής***. Όταν πρόκειται για παρακολούθηση

της καθημερινής δραστηριότητας του χρήστη και περαιτέρω αξιολόγηση της φυσικής κατάστασης του χρήστη, η εφαρμογή πρέπει να έχει προσωπικά στοιχεία του χρήστη και με αυτά να επιτρέπει στον χρήστη να προσαρμόσει την εφαρμογή σύμφωνα με τις ανάγκες του. Τέτοιες ανάγκες είναι τα εξατομικευμένα προγράμματα διατροφής που έχουν σχεδιαστεί για τις προτιμήσεις, τις τροφικές δυσανεξίες, και τις συγκεκριμένες απαιτήσεις ή τροφικές αλλεργίες του καθενός. Θα είναι πιο πιθανό να τηρήσει ένα σχέδιο που περιλαμβάνει τα είδη των φαγητών που απολαμβάνει. Με προσαρμοσμένα προγράμματα διατροφής, μπορούν να αποφευχθούν αβίαστα οι παγίδες που έρχονται με την απλή περιήγηση στην αγορά χωρίς προφανή πρόθεση ή σκοπό. Γι' αυτό η λίστα αγορών και οι συνταγές που συνοδεύουν ένα προσαρμοσμένο πρόγραμμα διατροφής είναι τόσο χρήσιμες. Εξοικονομούν χρόνο κάνοντας τον προγραμματισμό και την εμπειρία αγορών ευκολότερη και πιο σκόπιμη.

Η διατροφή είναι ο πυρήνας κάθε δραστηριότητας γυμναστικής. Υπάρχουν πολλές εφαρμογές παρακολούθησης διατροφής διαθέσιμες στην αγορά που επιτρέπουν στους χρήστες σας να ελέγχουν το βάρος τους παρέχοντάς τους καθημερινά δεδομένα για τις καμένες θερμίδες, ελέγχοντας το ισοζύγιο νερού και παρακινώντας τους χρήστες να ενθαρρύνουν υγιεινές διατροφικές συνήθειες. Εάν ένα άτομο αντιμετωπίζει προβλήματα με το να τηρεί την υγιεινή διατροφή στην διάρκεια του χρόνου, η εφαρμογή θα βοηθήσει να γνωρίζει αν σε κάποιο χρονικό διάστημα, παρ όλες τις διατροφικές του ατασθαλίες, η διατροφή του κινήθηκε προς την κατεύθυνση του στόχου του. Χρήσιμη δυνατότητα αποτελεί η παρακολούθησης της γεωγραφικής θέσης που επιτρέπει στον χρήστη να παρακολουθεί τις διαδρομές του περπατήματος του και να καταγράφει τις προπονήσεις. Χρησιμοποιώντας αυτή την πλατφόρμα, η εφαρμογή παρακολούθησης φυσικής κατάστασης μπορεί να δημιουργήσει έναν ακατανίκητο χώρο στις καρδιές των χρηστών της, καθώς αυτό θα τους επιτρέψει να παρακολουθούν την εξέλιξη της φυσικής τους κατάστασης σε μεγαλύτερες χρονικές περιόδους.

Όποιος προπονείται τακτικά δεν θα ήθελε ποτέ να χάσει μια συνεδρία. Ωστόσο, συμβαδίζοντας με τις πολυάσχολες ζωές μας, μερικές φορές, ξεφεύγει από το μυαλό μας. Σε τέτοιες περιπτώσεις, μια ειδοποίηση push από την εφαρμογή μπορεί να είναι πολύ χρήσιμη για να μας υπενθυμίσει την προπόνησή μας. Αυτός είναι ακριβώς ο λόγος για τον οποίο πολλές εταιρείες ανάπτυξης εφαρμογών Fitness επενδύουν πολύ κεφάλαιο και χρόνο στο σύστημα ειδοποιήσεων. Ωστόσο, πρέπει να έχουμε κατά νου ένα πράγμα: Οι ειδοποιήσεις ώθησης λειτουργούν ως δίκοπο μαχαίρι. Η υπερβολική ώθηση των ειδοποιήσεων Push θα μπορούσε να ενοχλήσει τους χρήστες.

Όλοι έχουμε κάνει επιπλέον πέντε push up τη στιγμή που το γυμναστήριο παίζει το αγαπημένο μας τραγούδι. Επιπλέον, τείνουμε να πιέζουμε περισσότερο τον εαυτό μας κάθε φορά που υπάρχει ένας στόχος να επιτύχουμε. Έτσι αν μπορούν οι χρήστες μέσα από την εφαρμογή να ολοκληρώσουν αυτούς τους στόχους και να κερδίζουν ανταμοιβές θα βοηθήσει πολύ στην αίσθηση της ολοκλήρωσης.

Τέτοιο είδους εφαρμογές μπορούν να επιτρέψουν στους χρήστες να συγχρονίσουν τα δεδομένα τους με διάφορες κινητές συσκευές. Αυτή η δυνατότητα θα δίνει ένα επιπλέον πλεονέκτημα, καθώς με αυτόν τον τρόπο η εφαρμογή παρέχει στους χρήστες την δυνατότητα να συλλέγουν πληροφορίες από πολλαπλές πηγές και να συσχετίζουν τα δεδομένα και να έχουν καλύτερη συνολική εικόνα της φυσικής του κατάστασης.

Ένας ακόμα στόχους αυτών των εφαρμογών είναι το να βοηθήσουν τους χρήστες να φέρουν μετρήσιμα αποτελέσματα από τις καθημερινές τους προπονήσεις. Ο χρήστης ορίζει μόνος του τα επιθυμητά αποτελέσματα, επομένως η διαδικασία καθορισμού στόχων πρέπει να είναι απλή και σαφής. Εκτός από αυτό, οι εφαρμογές γυμναστικής εστιάζουν επίσης σε αθλητικές δραστηριότητες

και ρυθμίσεις διατροφικών στόχων. Για την αποτελεσματική χρήση τέτοιων εφαρμογών, ο χρήστης πρέπει να μπορεί να επιλέξει μια καθημερινή αθλητική δραστηριότητα, να εισαγάγει μια τιμή στόχου ή να επιλέγει μια συγκεκριμένη ημερομηνία για την επίτευξη του στόχου αυτού. Επιπλέον, θα πρέπει να εμφανίζονται τα στατιστικά της απόδοσης της δραστηριότητας

Δεν μπορούν να επιτευχθούν οι στόχοι της φυσικής κατάστασης του χρήστη αν δεν μπορεί να μετράει την πρόοδό τους. Επαναλήψεις, σετ, θερμίδες, ώρες, χιλιόμετρα, κιλά, κιλά – όλα μπορούν να μετρηθούν. Οι κάτοχοι τέτοιων εφαρμογών παρακινούνται από αυτές τις πληροφορίες και να συνεχίζουν να χρησιμοποιούν την εφαρμογή για να επιτύχουν περισσότερα. Το πιο σημαντικό εδώ είναι ότι οι εφαρμογές γυμναστικής δεν παρακολουθούν περιττές ποσότητες δεδομένων για να αποφύγουν την υπερφόρτωση δεδομένων. Αρμόζουσα λύση είναι η γραφική επισκόπηση των μετρήσεων της αθλητικής άσκησης που παρέχει ολόκληρη τη σύνοψη της στα χέρια του χρήστη, συμπεριλαμβανομένου του χάρτη διαδρομής.

Για πολλά χρόνια γιατροί, αθλητές και αστροναύτες χρησιμοποιούν συσκευές παρακολούθησης καρδιακών παλμών για τη μέτρηση του στρες και άλλων μετρήσεων υγείας. Σήμερα, η πλειονότητα των εφαρμογών φυσικής κατάστασης χρησιμοποιούν οπτικές συσκευές παρακολούθησης καρδιακών παλμών.
Ακόμα η παρακολούθηση βημάτων είναι μια απαραίτητη δυνατότητα για νεοσύστατες επιχειρήσεις που θέλουν να δημιουργήσουν μια εφαρμογή που βοηθά τους χρήστες με την παρακολούθηση της προπόνησής τους. Με αυτήν τη δυνατότητα μιας εφαρμογής παρακολούθησης φυσικής κατάστασης, οι χρήστες μπορούν να ελέγξουν τα συνολικά βήματα μιας ολόκληρης μέρας και τις θερμίδες που καίγονται ενώ κάνουν διάφορες δραστηριότητες φυσικής κατάστασης όπως περπάτημα, τρέξιμο και ποδηλασία.
Η λειτουργία του αλτίμετρου είναι χρήσιμη για τους χρήστες που ανεβαίνουν σκάλες ή σκαρφαλώνουν. Το χαρακτηριστικό αυτό μετρά και την αλλαγή ύψους (υψόμετρο). Αυτή η λειτουργία είναι επίσης χρήσιμη κατά το τρέξιμο, προκειμένου να μετρηθούν οι προσπάθειες που καταβάλλονται και την ολοκλήρωσή του.

Με της λειτουργίας της παρακολούθησης φαγητού, η εφαρμογή επιτρέπει στους χρήστες να καταγράφουν την πρόσληψη τροφής. Αυτό θα τους βοηθήσει να παρακολουθούν σε πραγματικό χρόνο την πρόσληψη θερμίδων και την ακριβή απαίτηση σύμφωνα με το σώμα τους. Η ενσωμάτωση της παρακινεί και καθοδηγεί τον χρήστη.

Τέλος ο ιχνηλάτης νερού είναι ένα μοναδικό χαρακτηριστικό που είναι βέβαιο ότι θα λάβει πράσινο σήμα από τους περισσότερους χρήστες. Οι ιχνηλάτες γυμναστικής μπορούν να το συμπεριλάβουν και να το εκμεταλλευτούν. Το νερό είναι ένα πολύ απαραίτητο αλλά παραμελημένο στοιχείο της διατροφής μας. Όλοι γνωρίζουμε ότι περίπου το 60% του σώματός μας αποτελείται από νερό. Ένα ανθρώπινο σώμα χρειάζεται 2-3 λίτρα νερό την ημέρα. Μια εφαρμογή πρέπει να υπενθυμίζει στον χρήστη να πίνει νερό και να παραμένει ενυδατωμένος σε διάφορα στάδια της ημέρας. Βοηθά σημαντικά στο να βοηθήσει τον χρήστη να επιτύχει τους στόχους του διασφαλίζοντας τη σωστή πρόσληψη νερού στο σώμα του.

2.2.1 Ευχρηστία της Εφαρμογής

Από τότε μέχρι σήμερα οι Android προγραμματιστές με την βοήθεια της ανατροφοδότησης των χρηστών έχουν συλλέξει αρκετή γνώση για να βγάλουν κάποιες βασικές αρχές που πρέπει να ακολουθούμε για να παράγουμε *εύχρηστες Android εφαρμογές*.

Συγκεκριμένα το 1990 ο Jakob Nielsen και ο Rolf Molich πρότειναν 10 κανόνες για την σχεδίαση ψηφιακών διεπαφών χρηστών, τους οποίους αν τους επεκτείνουμε για τις mobile εφαρμογές καταλήγουμε σε 5 σύμφωνα με την έρευνα της [2].

1. Αποτελεσματικότητα: Να επεκτείνονται οι πόροι σε σχέση με την ακρίβεια και την πληρότητα με την οποία οι χρήστες επιτυγχάνουν στόχους.
2. Ικανοποίηση: Οι χρήστες να έχουν ελευθερία από δυσφορία και θετική στάση απέναντι στη χρήση του προϊόν.
3. Δυνατότητα εκμάθησης: Το σύστημα θα πρέπει να είναι εύκολο στην εκμάθηση, έτσι ώστε ο χρήστης να μπορεί γρήγορα να αρχίσει να ολοκληρώνει τις εργασίες του με αυτό.
4. Δυνατότητα ενθύμησης: Το σύστημα θα πρέπει να θυμάται εύκολα, έτσι ώστε ο απλός χρήστης να μπορεί να επιστρέψει στο σύστημα μετά από κάποιο χρονικό διάστημα, χωρίς να χρειάζεται να μάθει τα πάντα από την αρχή.
5. Διαχείριση λαθών: Το σύστημα θα πρέπει να έχει χαμηλό ποσοστό σφαλμάτων, έτσι ώστε οι χρήστες να κάνουν σφάλματα και να μπορούν εύκολα να τα ανακτήσουν. Επιπλέον, δεν πρέπει να συμβαίνουν καταστροφικά σφάλματα.
6. Βεβαιωθείτε ότι το περιεχόμενό σας είναι δομημένο και κατανεμημένο κατάλληλα για τις διαστάσεις οθόνης κάθε συσκευής.
7. Ακολουθήστε τις οδηγίες διεπαφής χρήστη του κλάδου και τους κανονισμούς της εκάστοτε κυβέρνησης στην ανάπτυξη του προϊόντος σας για κινητά.
8. Αξιοποιήστε τις δυνατότητες της συσκευής για χρηστικότητα και προσβασιμότητα.
9. Δοκιμάστε την εφαρμογή σας σε πολλαπλές στιγμές κατά τη διάρκεια της σχεδίασης και ανάπτυξης της.
10. Συλλέξτε πληροφορίες (ποσοτικές και ποιοτικές) από την εμπειρία των χρηστών για να καθορίσετε που και ποιο είναι το περιεχόμενο που θέλουν να βλέπουν.
11. Αναπτύξτε μηχανισμούς ασφαλείας με βάσει τις κατευθυντήριες γραμμές ιδιωτικότητας σε κάθε χαρακτηριστικό της εφαρμογής που διαχειρίζεται ευαίσθητα δεδομένα χρηστών και κυβερνητικά συστήματα.

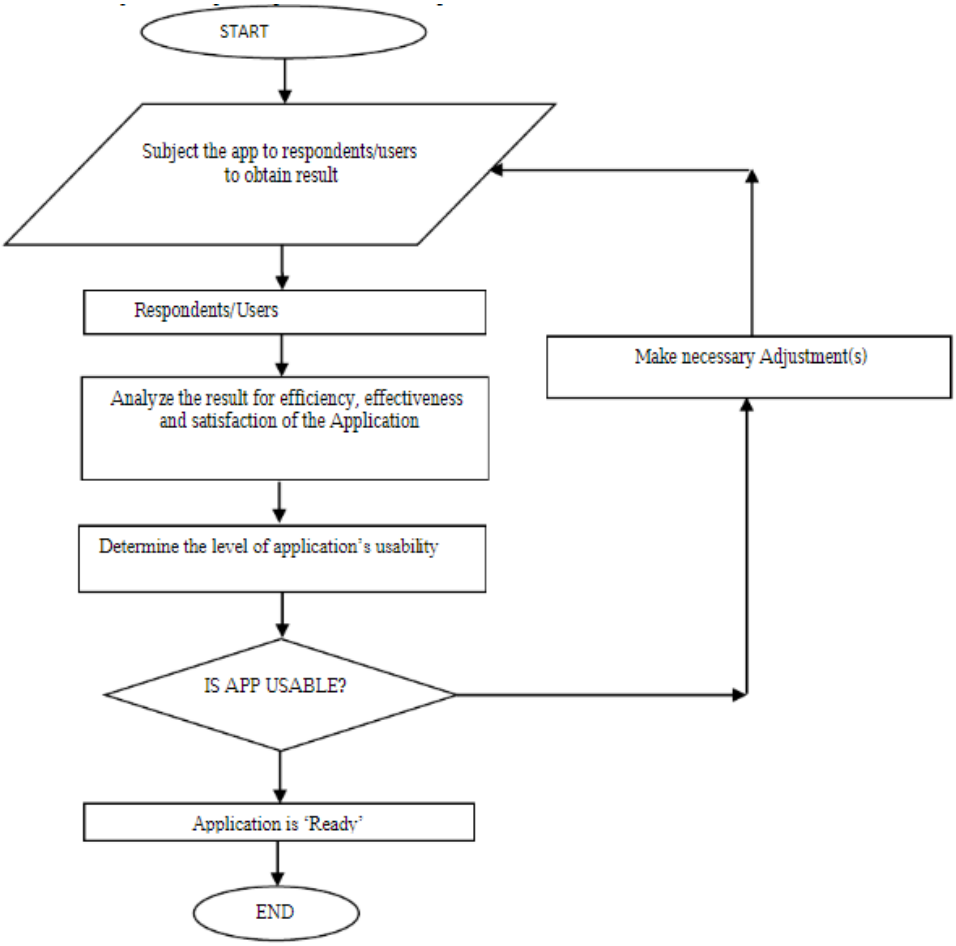
Αυτές οι γενικές αρχές πρέπει να ισχύουν σαν κανόνας για όλες τις mobile εφαρμογές. Παρόλα αυτά υπάρχουν και άλλες που θα αναφέρω και αφορούν περισσότερο τις *εμπορικές android εφαρμογές*. Στόχος τους η προώθηση υπηρεσιών και προϊόντων προς κάποιο τμήμα αγοραστικού κοινού:

1. **Συνέπεια και εξοικείωση:** Οι χρήστες πρέπει να νιώθουν ότι έχουν αλληλεπιδράσει με την εφαρμογή σας ξανά. Να αντιλαμβάνονται ασυναίσθητα που βρίσκεται η πληροφορία που ψάχνουν, τι είδους πληροφορία τους παρουσιάζεται, καθώς και να προβλέπουν τι θα συμβεί αν κάνουν κάποια ενέργεια (π.χ. πάτημα ενός κουμπιού). Για να πετύχει αυτό, οι προγραμματιστές θα πρέπει να ακολουθούν με συνέπεια τα εργαλεία που παρέχει η πλατφόρμα του Android, τα οποία χρησιμοποιούν οι υπόλοιπες εφαρμογές.
2. **Σχεδιασμός Πληροφορίας:** Σιγουρευτείτε ότι παρουσιάζεται η πληροφορία με τον καλύτερο δυνατό τρόπο για να την καταλάβουν οι χρήστες. Μπορείτε να πετύχετε αυτόν τον στόχο χρησιμοποιώντας γραφήματα, χάρτες, λίστες, πίνακες, φωτογραφίες κ.τ.λ. Ωστόσο θα πρέπει

να αξιοποιείται οτιδήποτε έτοιμο και επαναχρησιμοποιήσιμο παρέχει η πλατφόρμα του Android για την παρουσίαση αυτού του είδους της πληροφορίας που σκέφτεστε.

3. **Branding:** Είναι σημαντικό να μεταδώσετε την επωνυμία σας στους χρήστες, ώστε η εφαρμογή σας να είναι άμεσα αναγνωρίσιμη. Αυτό το πετυχαίνεται συνδυάζοντας την οπτική σας ταυτότητα με τη σχεδίαση του λειτουργικού συστήματος, όπως προσθέτοντας δικές σας χρωματικές αποχρώσεις σε έτοιμα κουμπιά, ετικέτες και γραφικά στοιχεία που παρέχει το Android ως native widgets του Material Design.
4. **Απλή ροή πλοήγησης:** Οι χρήστες θα πρέπει να γνωρίζουν ακριβώς σε ποια οθόνη βρίσκονται στην εφαρμογή σας, πόσο βαθιά είναι αυτή η οθόνη σε σχέση με την αρχική οθόνη καθώς και πως να επιστρέψουν πίσω στις οθόνες από τις οποίες οδηγήθηκαν εδώ. Πρέπει να τους είναι εύκολο να επιστρέψουν σε εκείνη, την επόμενη φορά που θα ανοίξουν την εφαρμογή σας. Θα είναι σε θέση να το κάνουν αυτό, πολύ πιο εύκολα εάν κάνετε την πλοήγηση στην εφαρμογή σας απλή και ξεκάθαρη για αυτούς.
5. **Εξάλειψη αγγαρειών:** Προσπαθήστε να πάρετε από τα χέρια του χρήστη κάθε δύσκολη, χρονοβόρα και επαναλαμβανόμενη διαδικασία που πρέπει να εκτελεί για να πετύχει κάποιο σκοπό. Μην ζητάτε πληροφορίες από αυτόν, τις οποίες μπορείτε να εκμαιεύσετε με κάποιον τρόπο μόνοι σας. Για παράδειγμα μην ζητάτε την ακριβή τοποθεσία του στο χάρτη, ενώ μπορείτε να χρησιμοποιήσετε το Location της Android συσκευής. Μην ζητάτε για το νόμισμα που χρησιμοποιούν, ενώ μπορείτε να το βρείτε από το που βρίσκονται στον κόσμο.
6. **Δυνατότητα αναίρεσης:** Δώστε στους χρήστες τη δυνατότητα να αναιρέσουν την προηγούμενη τους κρίσιμη ενέργεια ή προειδοποιήστε τους με παράθυρα επιβεβαίωσης. Προσέξτε όμως να μην το παρακάνετε.
7. **Συντομία και απλή γλώσσα:** Χρησιμοποιήστε μικρές και περιεκτικές προτάσεις για να μιλήσετε στον χρήστη μέσα από την εφαρμογή. Μην βάζετε αχρείαστη εξειδικευμένη ορολογία. Μεταφράστε τα κείμενα στην γλώσσα ομιλίας των χρηστών.
8. **Παροχή βοήθειας χρήσης:** Παρέχεται σύντομα και ανάλαφρα tutorials για το πως να χρησιμοποιήσουν τα χαρακτηριστικά της εφαρμογής σας για να οδηγηθούν στο αναμενόμενο για αυτούς αποτέλεσμα. Αυτό μπορεί να επιτευχθεί για παράδειγμα, με οθόνες wizard πριν το άνοιγμα ενός περίπλοκου χαρακτηριστικού της εφαρμογής. Επίσης πολύ χρήσιμα αποδεικνύονται τα συννεφάκια υπόδειξης που εμφανίζονται από πάνω όταν ο χρήστης ακουμπάει με το δάχτυλο του κάποιο από τα στοιχεία της οθόνης (buble hints, tips).
9. **Οι άδειες είναι τρομακτικές:** Όταν οι χρήστες καλούνται να παρέχουν μια άδεια στο λειτουργικό της εφαρμογής για να χρησιμοποιήσουν ένα χαρακτηριστικό θα πρέπει να νιώθουν ότι η άδεια που τους ζητείται είναι απαραίτητη και αφορά την πληροφορία που θα τους παρουσιαστεί στο σημείο που το ζητούν. Οι πολλαπλές άδειες ή οι άδειες που αιτούνται από την εφαρμογή σε λάθος σημεία τρομάζουν τους χρήστες και πατούν απόρριψη.
10. **Μην μπλοκάρετε τους χρήστες:** Αν οι χρήστες δοκιμάζουν την εφαρμογή σας για πρώτη φορά, δεν έχουν αποφασίσει ακόμα αν αυτή τους βολεύει για την ανάγκη που ψάχνουν να ικανοποιήσουν. Γι΄αυτό θα πρέπει να είναι ελεύθεροι να την χρησιμοποιήσουν χωρίς δεσμεύσεις στο βαθμό που είναι δυνατό. Επομένως η εγγραφή των χρηστών και η δημιουργία λογαριασμού προφιλ θα πρέπει να μην είναι απαραίτητη προϋπόθεση ώστε να αλληλεπιδράσουν οι χρήστες με τα βασικά χαρακτηριστικά της εφαρμογής σας.

11. **Εναλλακτικές και error handling:** Όταν οι χρήστες κάνουν κάτι λάθος ή για κάποιο αστάθμητο παράγοντα δεν μπορούν να χρησιμοποιήσουν κάποια χαρακτηριστικά της εφαρμογής, θα πρέπει να παρέχονται τρόποι για να συνεχίσουν να την χρησιμοποιούν κανονικά όπως και πριν. Για παράδειγμα αν ξαφνικά κοπεί η σύνδεση στο internet να σώζονται οι πληροφορίες που έβλεπε προηγουμένως σε μια τοπική βάση δεδομένων στο κινητό για να μην κοπεί η ροή της χρήσης του. Επιπροσθέτως αν προκύψει κάποιο πρόβλημα και η εφαρμογή δεν μπορεί να επεξεργαστεί και να παράγει τα επιθυμητά αποτελέσματα για τον χρήστη θα πρέπει να τον ενημερώνει ότι συνέβει κάτι απρόοπτο και ότι αυτή δεν είναι η συνηθισμένη κατάσταση της εφαρμογής. Έτσι να μπορεί να του δίνει την δυνατότητα να επαναλάβει την τελευταία του ενέργεια μετά από ένα εύλογο χρονικό διάστημα που όλες οι συνθήκες είναι όπως πρέπει για να συνεχιστεί η χρήση.
12. **Μην σπαμάρετε:** Μην εκνευρίζεται τον χρήστη με επαναλαμβανόμενες ενημερωτικές ειδοποιήσεις push που βγάζουν ήχο και δονούν τα κινητό του κάθε 1 δευτερόλεπτο. Μειώστε τες και στέλνετε μόνο την απαραίτητη στιγμή που χρειάζεται ή όταν είναι επείγον για αυτόν.
13. **Εντυπωσιάστε:** Κάντε την εφαρμογή σας εντυπωσιακή, διασκεδαστική και γρήγορη για τον χρήστη χωρίς υπερβολές. Χρησιμοποιήστε animations, όμορφα χρώματα και γραφικά που ταιριάζουν μεταξύ τους για να κάνετε τον χρήστη να θέλει να ξαναβιώσει την εμπειρία που του παρείχε η εφαρμογή σας [3].



Σχήμα 2.3. Διάγραμμα ροής δοκιμής ευχρηστίας.

Ένα πολύ σημαντικό κομμάτι του Android UI design είναι η *προσβασιμότητα*. Η Google προτείνει κάποιους τρόπους για να πετύχει η κατανόηση του περιεχομένου μιας Android εφαρμογής από όλους τους χρήστες:

- *Αυξήστε την ορατότητα του κειμένου*: Για κάθε σύνολο κειμένων εντός της εφαρμογής σας, συνιστούμε ότι η χρωματική αντίθεση —ή η διαφορά στην αντιληπτή φωτεινότητα μεταξύ του χρώματος του κειμένου και του χρώματος του φόντου πίσω από το κείμενο— πρέπει να είναι πάνω από ένα συγκεκριμένο όριο. Το ακριβές όριο εξαρτάται από το μέγεθος της γραμματοσειράς του κειμένου και από το αν το κείμενο εμφανίζεται με έντονη γραφή:

1. Εάν το κείμενο είναι μικρότερο από 18 pt ή εάν το κείμενο είναι έντονο και μικρότερο από 14 pt, ορίστε την αναλογία αντίθεσης χρώματος σε τουλάχιστον 4,5:1.
2. Για όλο το άλλο κείμενο, ορίστε την αναλογία χρωματικής αντίθεσης σε τουλάχιστον 3:1.

Η παρακάτω εικόνα δείχνει δύο παραδείγματα χρωματικής αντίθεσης κειμένου σε φόντο:

Σχήμα 2.4. Χρωματική αντίθεση κειμένου σε φόντο

- *Χρησιμοποιήστε μεγάλα, απλά στοιχεία ελέγχου*: Η διεπαφή χρήστη της εφαρμογής σας είναι πιο εύκολη στη χρήση, εάν τα στοιχεία ελέγχου της είναι πιο εύκολα ορατά για να τα πατήσει ο χρήστης. Συνιστούμε κάθε στοιχείο διαδραστικής διεπαφής χρήστη να έχει μια περιοχή με δυνατότητα εστίασης ή μέγεθος στόχου αφής τουλάχιστον 48dpx48dp. Το μεγαλύτερο είναι ακόμα καλύτερο. Για να έχει ένα δεδομένο στοιχείο διεπαφής χρήστη ένα αρκετά μεγάλο μέγεθος στόχου αφής, θα πρέπει να ισχύουν και οι δύο παρακάτω συνθήκες:
    1. Το άθροισμα των τιμών των android:paddingLeft, android:minWidth και android:paddingRight είναι μεγαλύτερο ή ίσο με 48dp.
    2. Το άθροισμα των τιμών των android:paddingTop, android:minHeight και android:paddingBottom είναι μεγαλύτερο ή ίσο με 48dp.
    3. Οι τιμές συμπλήρωσης επιτρέπουν το ορατό μέγεθος ενός αντικειμένου να είναι μικρότερο από 48dpx48dp, ενώ εξακολουθεί να έχει το προτεινόμενο μέγεθος στόχου αφής.
- *Περιγράψτε κάθε στοιχείο διεπαφής χρήστη*: Για κάθε στοιχείο διεπαφής χρήστη στην εφαρμογή σας, συμπεριλάβετε μια περιγραφή που περιγράφει τον σκοπό του στοιχείου. Στις περισσότερες περιπτώσεις, συμπεριλαμβάνετε αυτήν την περιγραφή στο χαρακτηριστικό contentDescription του στοιχείου, όπως φαίνεται στο ακόλουθο απόσπασμα κώδικα:

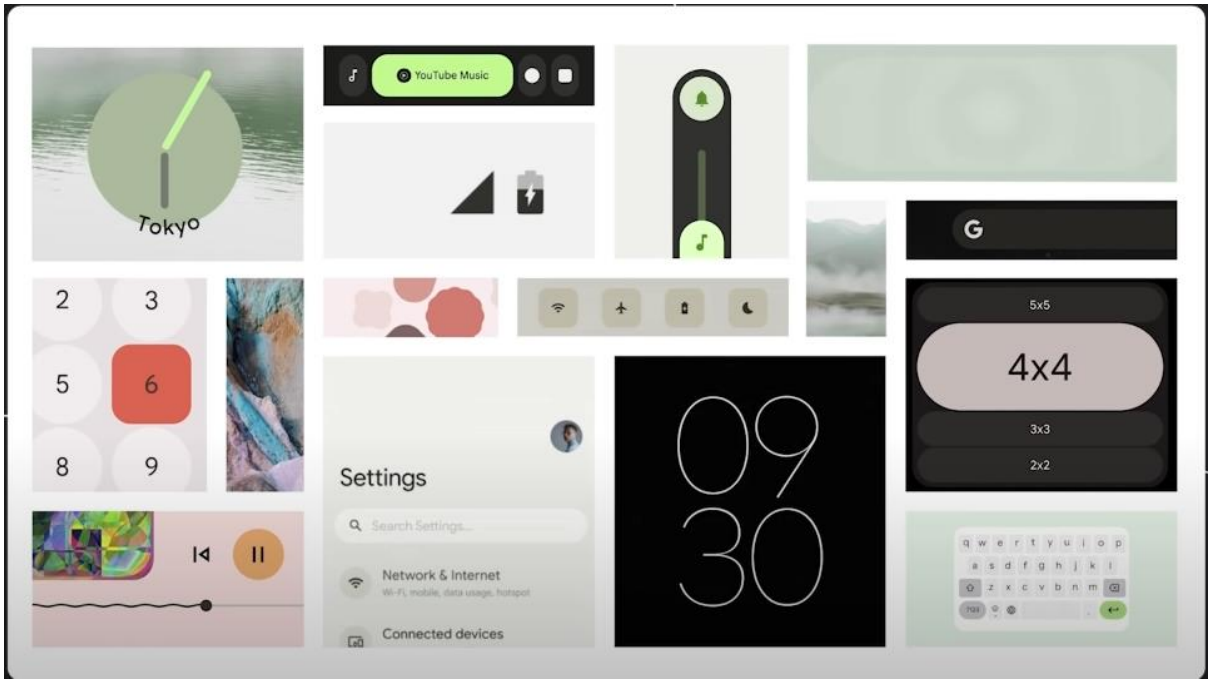
```
<!-- Use string resources for easier localization. -->
<!-- The en-US value for the following string is "Inspect". -->
<ImageView
    ...
    android:contentDescription="@string/inspect" />
```

Σχήμα 2.5. Περιγραφή στοιχείου οθόνης

- Όταν προσθέτετε περιγραφές στα στοιχεία διεπαφής χρήστη της εφαρμογής σας, λάβετε υπόψη τις ακόλουθες βέλτιστες πρακτικές:
    1. Μην συμπεριλάβετε τον τύπο του στοιχείου διεπαφής χρήστη στην περιγραφή περιεχομένου. Οι αναγνώστες οθόνης ανακοινώνουν αυτόματα τόσο τον τύπο όσο και την περιγραφή του στοιχείου. Για παράδειγμα, εάν η επιλογή ενός κουμπιού προκαλεί την ενέργεια "υποβολή" στην εφαρμογή σας, κάντε την περιγραφή του κουμπιού "Υποβολή" και όχι "Κουμπί υποβολής".
    2. Κάθε περιγραφή πρέπει να είναι μοναδική. Με αυτόν τον τρόπο, όταν οι χρήστες του προγράμματος ανάγνωσης οθόνης αντιμετωπίζουν μια επαναλαμβανόμενη περιγραφή στοιχείου, αναγνωρίζουν σωστά ότι η εστίαση είναι σε ένα στοιχείο που είχε ήδη εστίαση νωρίτερα. Συγκεκριμένα, κάθε στοιχείο σε μια ομάδα προβολών όπως το RecyclerView/LazyList πρέπει να έχει διαφορετική περιγραφή. Κάθε περιγραφή πρέπει να αντικατοπτρίζει το περιεχόμενο που είναι μοναδικό για ένα δεδομένο στοιχείο, όπως το όνομα μιας πόλης σε μια λίστα τοποθεσιών.
    3. Μπορείτε να ορίσετε το χαρακτηριστικό android:importantForAccessibility σε "όχι" για γραφικά στοιχεία που χρησιμοποιούνται μόνο για διακοσμητικά εφέ.

2.2.2 Material Design

Η Google από το 2014 έχει εισάγει ένα σύστημα σχεδίασης διεπαφής χρήστη στα Android κινητά που ονομάζεται Material Design [25]. Σκοπός του είναι να χτιστούν τολμηρές, όμορφες και συνεπείς μεταξύ τους, ψηφιακές εμπειρίες μέσα από τις Android συσκευές. Την σχεδίαση αυτή ακολουθούν όλα τα Android λογισμικά περιβάλλοντα από το Android 5.0 (Lollipop) μέχρι σήμερα. Τον Οκτώβριο του 2022 με το λανσάρισμα του Android 12 (Snow Cone), η Google αναβάθμισε το πρότυπο αυτό με την έκδοση Material Design 3. Προτείνει σε όλους τους προγραμματιστές να ακολουθούν αυτούς τους νέους κανόνες όταν σχεδιάζουν τις οθόνες των εφαρμογών τους. Το πρότυπο αυτό προσαρμόζει τα UI στοιχεία που υπάρχουν διαθέσιμα μέσα από το Android Studio για να τα χρησιμοποιήσουμε για τις διάφορες ανάγκες της σχεδίασης μας. Περιλαμβάνει αλλαγές στις χρωματικές παλέτες, στους συνδυασμούς χρωμάτων, στα μεγέθη και στα σχήματα των UI συστατικών. Ακόμα συστήνει νέες προδιαγραφές για τα animations που αφορούν την κίνηση των συστατικών στην οθόνη, καθώς και την μετάβαση από μια οθόνη σε μια άλλη.



Σχήμα 2.6. Material Design βασικά στοιχεία οθόνης ευρέως χρησιμοποιούμενα.

Σημαντικό κομμάτι του Material Design αποτελεί και η μορφοποίηση των γραμματοσειρών. Με την 3η έκδοση η Google προσάρμοσε τις γραμματοσειρές της ώστε να γίνουν πιο ελκυστικές και ευανάγνωστες.
Υπάρχουν βασικές αρχές του Material Design που υπόκεινται στα διάφορα μέρη της οθόνης μιας android εφαρμογής. Τα μέρη αυτά περιλαμβάνουν τις επιφάνειες, την τυπογραφία, το χρώμα, τα εικονίδια και τις εικόνες, τα Layouts και τα Grids, την κίνηση και τα animations των στοιχείων της οθόνης, την προσβασιμότητα, την ανατροφοδότηση του χρήστη, τον προσαρμοστικό σχεδιασμό, της δοκιμές του χρήστη.

Ξεκινώντας από το πρώτο που είναι οι επιφάνειες υλικού το Material Design τονίζει τη χρήση φυσικών επιφανειών και σκιών για να δημιουργήσει αίσθηση βάθους και ιεραρχίας στη διεπαφή χρήστη. Τα αντικείμενα εντός της διεπαφής πρέπει να μιμούνται υλικά από τον πραγματικό κόσμο και να αντιδρούν στο φως και τη σκιά για να παρέχουν στους χρήστες μια ψηφιακή εμπειρία που ανταποκρίνεται στην αφή και είναι ευαίσθητη και εύχρηστη. Περνώντας στην τυπογραφία πρέπει να χρησιμοποιείται για να καθιερώσει ιεραρχία και αναγνωσιμότητα. Πρέπει να επιλέγονται γραμματοσειρές που είναι αναγνώσιμες σε διάφορα μεγέθη οθονών και να διατηρείται μια συνεπή κλίμακα γραμματοσειρών σε όλη την εφαρμογή. Όσον αφορά το χρώμα το Material Design ενθαρρύνει τη χρήση ζωηρών και νοήμον χρωμάτων. Ζητάει την δημιουργία μιας παλέτας χρωμάτων που συμβαδίζει με το branding της εκάστοτε εφαρμογής ενώ ταυτόχρονα διασφαλίζει την προσβασιμότητα για όλους τους χρήστες, συμπεριλαμβανομένων εκείνων με προβλήματα όρασης. Συνεχίζοντας με την επόμενη αρχή τα εικονίδια πρέπει να είναι καθαρά, εύληπτα και να ακολουθούν τις οδηγίες εικονογραφίας του Material Design. Προτείνεται η χρήση της εικονογραφίας με σκέψη για να βελτιώνεται η εμπειρία του χρήστη και να διατηρείται η οπτική συνέπεια. Τα Layout και τα Grids είναι δύο από τα πιο χρήσιμα στοιχεία διάταξης που περικλείουν εσωτερικά βασικών στοιχεία (π.χ. κουμπιά, κείμενα) και πρέπει να χρησιμοποιούνται διατάξεις που ανταποκρίνονται στις ενέργειες του

χρήστη (όπως το scrolling) και να τηρείται ένα συνεπές σύστημα πλέγματος για τον οργανισμό του περιεχομένου. Τονίζεται να ακολουθούμε τις οδηγίες για τα διαστήματα και την ευθυγράμμιση, προκειμένου να δημιουργούμε μια καθαρή και αισθητικά ευχάριστη διεπαφή για τον χρήστη. Για την κίνηση και τα animations πρέπει να συμπεριλαμβάνονται διακριτικές κινήσεις της θέσης και διακριτές μεταβάσεις της κατάστασης των στοιχείων της οθόνης για να παρέχετε ανατροφοδότηση και ξεκάθαρο πλαίσιο χώρου στο οποίο οι χρήστες μπορούν να αλληλεπιδρούν με τα στοιχεία που υπέστησαν κάποιο animation. Η κίνηση πρέπει να αισθάνεται φυσική και να ενισχύει την κατανόηση των λειτουργιών της εφαρμογής από τον χρήστη. Περί προσβασιμότητας ζητάει να υπάρχει βεβαιότητα ότι η εφαρμογή είναι προσβάσιμη για όλους τους χρήστες, συμπεριλαμβανομένων εκείνων με αναπηρίες. Ζητάει να ακολουθούμε τις οδηγίες προσβασιμότητας για να παρέχετε εναλλακτικό κείμενο για εικόνες, να διατηρείτε κατάλληλη αντίθεση και να προσφέρετε υποστήριξη για αναγνωστές οθονών. Ακόμα εντείνει να παρέχεται σαφή και έγκαιρη ανατροφοδότηση στους χρήστες όταν αλληλεπιδρούν με την εφαρμογή. Τέτοια ανατροφοδότηση περιλαμβάνει κινήσεις, μικρές αλληλεπιδράσεις και οπτικές ενδείξεις για να υποδεικνύεται η κατάσταση των στοιχείων της διεπαφής.

Για να επιτύχουμε προσαρμοστικό σχεδιασμό πρέπει να λάβουμε υπόψη τα διάφορα μεγέθη οθόνης, προσανατολισμούς (portrait, landscape) και δυνατότητες συσκευής (αν έχουμε τους απαραίτητους σένσορες ή εφαρμογές διαχείρισης βασικών λειτουργιών του Android συστήματος) κατά το σχεδιασμό της εφαρμογής. Δημιουργήστε διατάξεις και σχέδια που προσαρμόζονται με χάρη σε διαφορετικούς παράγοντες μορφής (Light/Dark mode, low blue color emission mode, do not disturb mode κ.α.).

Τέλος αλλά εξίσου σημαντικό είναι η αναδρομική πραγματοποίηση δοκιμών χρηστικότητας με πραγματικούς χρήστες για να συγκεντρώνονται σχόλια και να επανασχεδιάζεται η διεπαφή της εφαρμογής με βάση τις εμπειρίες και τις ανάγκες τους.

2.2.3 Κατευθυντήριες Γραμμές Jetpack Compose

Το Jetpack Compose είναι μια σύγχρονη εργαλειοθήκη Android UI που αναπτύχθηκε από την Google για τη δημιουργία εγγενών διεπαφών χρήστη με πιο δηλωτικό και συνθετικό τρόπο. Τον Μάρτιο του 2021 η Google δημοσίευσε κάποιες βασικές αρχές που πρέπει να ακολουθούν οι Android προγραμματιστές όταν χτίζουν μια διεπαφή με Jetpack Compose [26]. Ειδικότερα θα πρέπει να αξιοποιείται η δηλωτική φύση του Jetpack Compose. Να περιγράφεται η διεπαφή ως μια συνάρτηση της τρέχουσας κατάστασης της αφού το Compose διαχειρίζεται αξιόπιστα την απεικόνιση της και τις ενημερώσεις στα στοιχεία της. Να ορίζονται τα στοιχεία της διεπαφής ως Composables που έχουν την ετικέτα @Composable. Αυτές οι συναρτήσεις πρέπει να είναι μικρές, εστιασμένες και εύκολες στην κατανόηση. Να μεταχειρίζονται τα στοιχεία της διεπαφής ως αμετάβλητα. Όταν χρειάζεται να ενημερωθεί η διεπαφή, να δημιουργείται μια νέα κατάσταση που περιγράφει την οθόνη με τις επιθυμητές αλλαγές. Να αποφεύγεται η τροποποίηση των στοιχείων της διεπαφής στη μέσα από την Composable μέθοδο που τα δημιουργεί. Να δημιουργούνται επαναχρησιμοποιήσιμα Composables με κοινά στοιχεία ή μοτίβα της διεπαφής (π.χ., κουμπιά, κάρτες). Αυτό προάγει την επαναχρησιμοποίηση του κώδικα και τη καλύτερη συντήρηση του. Να χρησιμοποιείται η μέθοδος “remember” για τη εσωτερική διαχείριση της κατάστασης των Composable μεθόδων. Για την κατάσταση της εφαρμογής, συνιστάται η χρήση του ViewModel ή άλλων στοιχείων αρχιτεκτονικής. Όταν χρησιμοποιείται ViewModel, να χρησιμοποιούνται συναρτήσεις viewModel κατασκευαστές και observeAsState για να συνδέονται τα στοιχεία της διεπαφής Compose με τα δεδομένα του ViewModel. Να καθορίζονται και να χρησιμοποιούνται Themes και Styles για να διασφαλίζεται μια

συνεπής εμφάνιση στην εφαρμογή. Το Compose καθιστά εύκολη την εφαρμογή στοιχείων του Theme στα στοιχεία της διεπαφής. Να βελτιστοποιείται η συγγραφή του κώδικα που περιγράφει την διεπαφή για απόδοση, είτε στον καθορισμό της UI κατάστασης, είτε εσωτερικά στα Composables. Το Compose είναι σχεδιασμένο να ενημερώνει αποτελεσματικά μόνο τα τμήματα της διεπαφής που έχουν αλλάξει. Ωστόσο αυτό έχει το ρίσκο πολλαπλά Composables να επανασχεδιάζονται αυτόματα χωρίς να αλλάζουν τα δεδομένα που επεξεργάζονται. Να χρησιμοποιείται το API Modifier για τη βελτιστοποίηση της απεικόνισης. Να συντάσσονται δοκιμές διεπαφής για τα Composables χρησιμοποιώντας εργαλεία όπως το compose-test. Να ελέγχονται τόσο η αναμενόμενη συμπεριφορά όσο και οι αλληλεπιδράσεις των στοιχείων της διεπαφής. Να διασφαλίζεται ότι η διεπαφή είναι προσβάσιμη για όλους τους χρήστες. Να χρησιμοποιούνται οι Modifiers προσβασιμότητας και να ελέγχεται η εφαρμογή με υπηρεσίες προσβασιμότητας. Να εφαρμόζεται κατάλληλη διαχείριση σφαλμάτων και επικύρωση για να παρέχετε μια ομαλή εμπειρία στον χρήστη. Να εξετάζεται η χρήση μεθόδων σφαλμάτων για την εξακρίβωση σφαλμάτων. Να χρησιμοποιείται το Compose Navigation API για την πλοήγηση μεταξύ διαφορετικών οθονών και προορισμών στην εφαρμογή. Να παρέχεται ανατροφοδότηση στους χρήστες μέσω animation, μικρών αλληλεπιδράσεων και οπτικών ενδείξεων. Να χρησιμοποιείται το API Transition για πολύπλοκα animations. Να σχεδιάζεται η διεπαφή ώστε να προσαρμόζεται αρμονικά σε διάφορα μεγέθη οθόνης, προσανατολίσεις (Portrait, Landscape) και δυνατότητες συσκευής (υπάρχων σένσορες, υπάρχουσες υπηρεσίες διαχείρισης πόρων του Android λειτουργικού). Να γίνεται διενέργεια δοκιμών με πραγματικούς χρήστες για να συλλέγονται δεδομένα ανατροφοδότηση και να εξελίσσεται ο σχεδιασμός της εφαρμογής με μια αναδρομική διαδικασία βάσει των εμπειριών και των αναγκών τους.

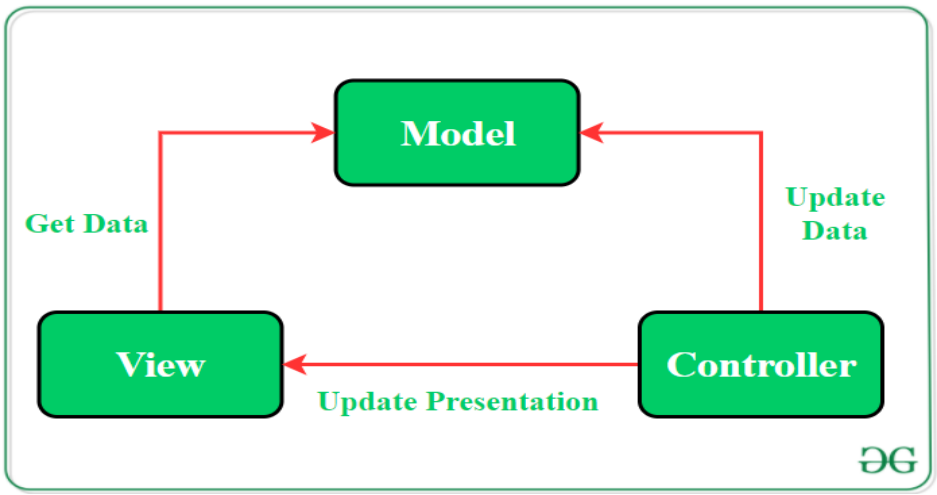
Κεφάλαιο 3: Μεθοδολογία και σχεδιαστικές αποφάσεις

3.1: Κεντρική Ιδέα

3.1.1 Πρότυπα Σχεδίασης (Design Patterns)

Η ανάπτυξη μιας εφαρμογής Android με την εφαρμογή ενός μοτίβου αρχιτεκτονικής λογισμικού προτιμάται πάντα από τους προγραμματιστές. Ένα μοτίβο αρχιτεκτονικής δίνει σπονδυλωτότητα στα αρχεία του έργου και διασφαλίζει ότι όλα τα κομμάτια κώδικα καλύπτονται από το Unit Testing (κώδικας που ανατρέχει τον κώδικα και επαληθεύει την ορθότητα του με βάση το αναμενόμενο αποτέλεσμα). Διευκολύνει τους προγραμματιστές να συντηρήσουν το λογισμικό και να επεκτείνουν τις δυνατότητες της εφαρμογής στο μέλλον. Υπάρχουν ορισμένες αρχιτεκτονικές που είναι πολύ δημοφιλείς μεταξύ των προγραμματιστών. Θα σας περιγράψω πως έφτασε στο Android να χρησιμοποιείται το MVVM (Model-View-ViewModel) το οποίο καθιερώθηκε και προτείνεται από την Google σε όλα της τα βίντεο και άρθρα τεκμηρίωσης που δημοσιεύει. Εδώ και πολλά χρόνια στο web development υπήρχε το μοτίβο Model—View—Controller(MVC) το οποίο αρχικά πέρασε και στο android development. Το μοτίβο MVC προτείνει τον διαχωρισμό του κώδικα σε 3 στοιχεία [15]. Κατά τη δημιουργία της κλάσης/αρχείου της εφαρμογής, ο προγραμματιστής πρέπει να την κατηγοριοποιήσει σε ένα από τα ακόλουθα τρία επίπεδα:

- Model: Αυτό το στοιχείο αποθηκεύει τα δεδομένα της εφαρμογής. Δεν έχει γνώση για τη διεπαφή. Το μοντέλο είναι υπεύθυνο για το χειρισμό της λογικής του τομέα (Business Logic) και την επικοινωνία με τη βάση δεδομένων και τα επίπεδα δικτύου.
- View: Είναι το επίπεδο UI (User Interface) που περιέχει στοιχεία που είναι ορατά στην οθόνη. Επιπλέον, παρέχει την οπτικοποίηση των δεδομένων που είναι αποθηκευμένα στο Μοντέλο και προσφέρει αλληλεπίδραση στον χρήστη.
- Controller: Αυτό το στοιχείο καθορίζει τη σχέση μεταξύ της προβολής και του μοντέλου. Περιέχει τη βασική λογική της εφαρμογής και λαμβάνει τη συμπεριφορά του χρήστη και ενημερώνει το Model ανάλογα με τις ανάγκες.



Σχήμα 3.1. Σύνδεση και επικοινωνία μεταξύ των συστατικών του MVC

Στην αρχιτεκτονική MVC, τα δεδομένα της εφαρμογής ενημερώνονται από τον Controller και το View λαμβάνει τα δεδομένα. Εφόσον το Model είναι διαχωρισμένο, θα μπορούσε να ελεγχθεί ανεξάρτητα από τη διεπαφή χρήστη. Επιπλέον, εάν το επίπεδο προβολής σέβεται την αρχή της ενιαίας ευθύνης (Single Responsibility Principle) , τότε ο ρόλος τους είναι απλώς να ενημερώνουν τον ελεγκτή για κάθε συμβάν χρήστη και απλώς να εμφανίζουν δεδομένα από το Μοντέλο, χωρίς να εφαρμόζουν κανένα Business Logic. Σε αυτήν την περίπτωση, τα UI Tests θα πρέπει να είναι αρκετά για να καλύψουν τις λειτουργίες του View [15].

Χρησιμοποιώντας το MVC ως αρχιτεκτονική λογισμικού, οι προγραμματιστές καταλήγουν με τις ακόλουθες δυσκολίες:

- Το μεγαλύτερο μέρος του Business Logic βρίσκεται στον Controller. Κατά τη διάρκεια ζωής μιας εφαρμογής, αυτό το αρχείο μεγαλώνει και γίνεται δύσκολη η διατήρηση του κώδικα.
- Λόγω των στενά συνδεδεμένων μηχανισμών διεπαφής χρήστη και πρόσβασης δεδομένων, τόσο το επίπεδο Controller όσο και το επίπεδο View εμπίπτουν στο ίδιο Activity ή Fragment. Αυτό προκαλεί πρόβλημα στην πραγματοποίηση αλλαγών κατά την συντήρηση της εφαρμογής.
- Γίνεται δύσκολο να πραγματοποιηθεί Unit Testing στο διάφορα επίπεδα, καθώς το μεγαλύτερο μέρος του τμήματος που βρίσκεται υπό δοκιμή χρειάζεται στοιχεία Android SDK.

Το μοτίβο MVP ξεπερνά αυτές τις προκλήσεις του MVC και παρέχει έναν εύκολο τρόπο δομής του κώδικα του έργου. Ο λόγος για τον οποίο το MVP είναι ευρέως αποδεκτό είναι ότι παρέχει αρθρωτότητα, δυνατότητα δοκιμής και μια πιο καθαρή και συντηρήσιμη βάση κώδικα. Αποτελείται από τα ακόλουθα τρία συστατικά:

- Model: Επίπεδο για αποθήκευση δεδομένων. Είναι υπεύθυνο για το χειρισμό της λογικής του τομέα (real Business Rules) και της επικοινωνίας με τη βάση δεδομένων και τα επίπεδα δικτύου.
- View: Επίπεδο διεπαφής χρήστη (User Interface). Παρέχει την οπτικοποίηση των δεδομένων και παρακολουθεί τη δράση του χρήστη προκειμένου να ειδοποιήσει τον Παρουσιαστή.
- Presenter: Λάβετε τα δεδομένα από το μοντέλο και εφαρμόζετε τη λογική διεπαφής χρήστη (UI Logic) για να αποφασίσετε τι θα εμφανιστεί. Διαχειρίζεται την κατάσταση του View και πραγματοποιεί ενέργειες σύμφωνα με την ειδοποίηση εισαγωγής του χρήστη από το View.

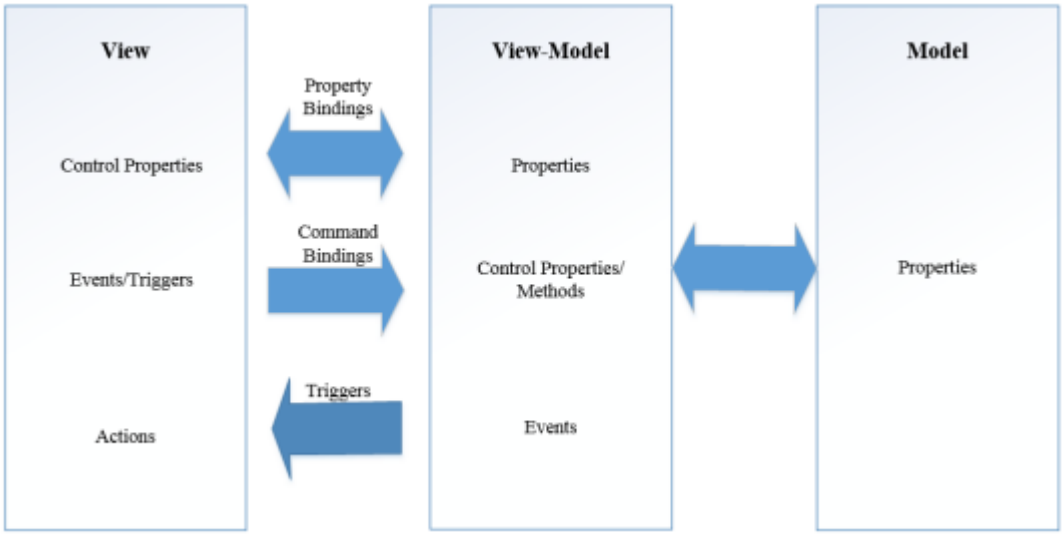
Πλεονεκτήματα της αρχιτεκτονικής MVP

- Καμία εννοιολογική σχέση στα στοιχεία Android
- Εύκολη συντήρηση και δοκιμή κώδικα καθώς το μοντέλο, η προβολή και το επίπεδο παρουσίασης της εφαρμογής διαχωρίζονται.

Μειονεκτήματα της αρχιτεκτονικής MVP

Εάν ο προγραμματιστής δεν ακολουθεί την αρχή SRP (Single Responsibility Principle) για να σπάσει τον κώδικα, τότε το επίπεδο Presenter τείνει να επεκταθεί σε μια τεράστια κλάση που γνωρίζει τα πάντα (God Class), πράγμα που την κάνει δύσκολα συντηρήσιμη [16].
Το MVVM ελαχιστοποιεί περαιτέρω τον κώδικα View Binding, δηλαδή τον τρόπο σύνδεσης του View με το Model. Η αρχιτεκτονική MVVM έχει τρία στοιχεία, όπως δηλώνει το όνομά του, Model, View και ViewModel. Το View εμφανίζει τη διεπαφή χρήστη της εφαρμογής. Στο MVVM, ο στόχος

είναι να είναι πιο φιλικό προς τους σχεδιαστές και να μπορεί εύκολα να υλοποιηθεί από τον σχεδιαστή αντί από τον προγραμματιστή κώδικα. Το Model αντιπροσωπεύει τα δεδομένα, το ίδιο με το μοντέλο που εξηγείται στις προηγούμενες αρχιτεκτονικές. Το View-Model, το οποίο σημαίνει μοντέλο προβολής, προορίζεται να διαχειρίζεται τις αλλαγές στην τρέχουμε κατάσταση του View (UI state). Θα περάσει τα δεδομένα και τις λειτουργίες για να προβληθούν από το View επίπεδο και επίσης θα διαχειριστεί τη λογική και τη συμπεριφορά του View.
Ένα καλό στοιχείο View-Model θα πρέπει περιέχει δεδομένα που αφορούν μόνο το UI state και όχι την προβολή των δεδομένων στο UI τόσο ως προς την ονομασία όσο και ως προς τον τύπο. Για παράδειγμα, εάν θέλουμε να αποθηκεύσουμε δεδομένα όταν είναι ενεργοποιημένο το κουμπί αποθήκευσης, αντί να ονομάσουμε τη μεταβλητή isSaveButtonEnabled, προτιμάτε τα ένα όνομα που αφορά την κατάσταση του UI όπως π.χ. canSave. Οι αλληλεπιδράσεις μεταξύ αυτών των συστατικών εξηγούνται στο Σχήμα 2.7. Η σύνδεση μεταξύ του ViewModel και του View είναι πιο πολύπλοκη από ό,τι στην αρχιτεκτονική MVP. Υπάρχουν δύο τύποι συνδέσεων: παραδοσιακή σύνδεση και data binding σύνδεση. Οι παραδοσιακές συνδέσεις είναι παρόμοιες στο MVC και στο MVP ότι ένα στοιχείο ViewModel θα αλλάξει το View.



Σχήμα 3.2. Σύνδεση και επικοινωνία μεταξύ των συστατικών του MVVM

Το Databinding είναι ένας νέος μηχανισμός που εισήχθη στο MVVM. Επιτρέπει στο View να έχει άμεση πρόσβαση στις ιδιότητες και τις λειτουργίες του View-Model. Με το Databinding, το View-Model δεν χρειάζεται να ειδοποιεί τις αλλαγές του View μέσω κώδικα. Αντ'αυτού το View γνωρίζει ότι τα δεδομένα έχουν φορτωθεί και εμφανίζει τα δεδομένα από μόνο του. Για παράδειγμα. Στο MVP και αρχιτεκτονική MVC, μετά τη φόρτωση των δεδομένων, ο παρουσιαστής και ο ελεγκτής θα ορίσουν το View σε κώδικα. Με τον μηχανισμό σύνδεσης δεδομένων (Data binding), το View παρακολουθεί τις αλλαγές σε αυτά τα δεδομένα και όταν τα δεδομένα έχουν φορτωθεί, η προβολή τους θα αλλάξει αυτόματα. Η σύνδεση δεδομένων μεταξύ του View και του View-Model μπορεί να είναι κατευθυντική και αμφίδρομη. Όταν τα δεδομένα που συνδέονται με το View αλλάζουν, τα δεδομένα στο View-Model γνωρίζουν επίσης την αλλαγή. Αυτή η δέσμευση, που αναφέρεται ως δεσμευτική ιδιοκτησία(property binding) είναι αμφίδρομη. Υπάρχει επίσης Data binding που ονομάζεται δέσμευση λειτουργίας (operation binding) και είναι κατευθυντήρια. Μια λειτουργία που

δημιουργήθηκε στο View-Model συνδέθηκε σε ένα γραφικό στοιχείο στο View. Αυτή η σύνδεση είναι σαν τον κώδικα JavaScript σε προβολή φόρμας HTML. Όταν ο χρήστης κάνει κλικ στο κουμπί αποθήκευσης σε μια φόρμα, θα καλέσει τη μέθοδο που έχει προκαθοριστεί στον κώδικα JavaScript. Με αυτόν τον τρόπο, στο View-Model, δεν χρειάζεται να γράψουμε κώδικα για να καταγράψουμε το καταγράψουμε το συμβάν του κλικ.
Για το MVP και το MVVM, δεν υπάρχουν στοιχεία που να δείχνουν ότι το ένα είναι ανώτερο από το άλλο. Αυτές οι δύο αρχιτεκτονικές έχουν παρόμοια απόδοση, ενώ το MVP παρέχει καλύτερη δυνατότητα τροποποίησης και το MVVM παρέχει καλύτερη δυνατότητα δοκιμής (Code Testing) χάρη στην καλύτερη του οργάνωση και τον εκτεταμένο boilerplate κώδικα (κώδικας που επαναλαμβάνεται και γράφεται χωρίς να επηρεάζει το αποτέλεσμα που θα δει ο χρήστης στο View της εφαρμογής αλλά γράφεται για να προσδώσει καλύτερη οργάνωση στα επίπεδα του κώδικα και να διαχωρίσει καλύτερα τα διάφορα modules του έργου) [4].

3.2: Επιλογή Εργαλείων / Τεχνολογιών / Γλωσσών Προγραμματισμού

3.2.1 Android Studio

Το Android Studio είναι το ενσωματωμένο περιβάλλον ανάπτυξης για την πλατφόρμα Android[1] της Google. Χρησιμοποιεί το λογισμικό της JetBrains' IntelliJ IDEA που είναι ένα ολοκληρωμένο περιβάλλον ανάπτυξης και προσθέτει στοιχεία πάνω σε αυτό που κάνει την ανάπτυξη εφαρμογών για Android συσκευές πιο εύκολη. Το περιβάλλον αυτό βοηθάει του προγραμματιστές να παράγουν κώδικα πιο γρήγορα και αποδοτικά, παρέχοντας επεξεργαστή κειμένου, μεταφραστή κώδικα, εντοπιστή σφαλμάτων κ.α. Οι εκδόσεις του Android Studio είναι συμβατές με ορισμένα MacOS, Windows και Linux. Με υποστήριξη για την πλατφόρμα Google Cloud και την ενσωμάτωση εφαρμογών Google, το Android Studio προσφέρει στους προγραμματιστές μια καλά εφοδιασμένη εργαλειοθήκη για τη δημιουργία εφαρμογών Android ή άλλων έργων και αποτελεί αναπόσπαστο μέρος της ανάπτυξης Android από το 2013 [19].

3.2.2 Android SDK

Το Android SDK (κιτ ανάπτυξης λογισμικού) είναι ένα σύνολο εργαλείων ανάπτυξης που χρησιμοποιούνται για την ανάπτυξη εφαρμογών για την πλατφόρμα Android [20]. Το Android SDK περιλαμβάνει τα εξής:

- Απαιτούμενες βιβλιοθήκες.
- Εντοπιστής σφαλμάτων.
- Εξομοιωτή Android συσκευών.
- Έγγραφα εγχειριδίων για τις διεπαφές προγραμμάτων εφαρμογών Android (API).
- Δείγματα πηγαίου κώδικα.
- Οδηγίες για το λειτουργικό σύστημα Android.

Ειδικότερα το μέσα από το Android Studio μπορούμε να κατεβάσουμε την αντίστοιχη SDK πλατφόρμα για κάθε έκδοση Android που κυκλοφορεί η Google. Το τρέχον νεότερο είναι το Android

[1] Στον σύνδεσμο θα βρείτε το έγγραφο οδηγό για την μελέτη της Android πλατφόρμας https://developer.android.com/docs

API 34 (Android 14). Ακόμα βλέπουμε τα SDK εργαλεία που μπορούμε να κατεβάσουμε όπως τα Google Play Services (χωρίς το οποίο δεν μπορούμε να χρησιμοποιήσουμε τις ανοιχτές υπηρεσίες του Google π.χ. Maps, Geofencing, Analytics κ.τ.λ.). Απαραίτητο εργαλείο είναι το Android SDK Platform-Tools, με το οποίο συνδέουμε τις φυσικές συσκευές όπως τα κινητά στο Desktop σύστημα μας για να περάσουμε με καλώδιο το APK και να τρέξουμε την εφαρμογή.

3.2.3 Android Gradle

Το Android Gradle συλλέγει όλα τα απαραίτητα αρχεία που απαρτίζουν το έργο της εφαρμογής, όπως τα αρχεία εικόνων, ήχου, κειμένου, τις εξωτερικές βιβλιοθήκες, της σχεδίασης οθονών, των κανόνων αποφυγής σμίκρυνσης κομματιών κώδικα κ.α. και τα ενοποιεί σε ένα APK αρχείο που χρησιμοποιούν οι συσκευές Android για να ανοίξουν την εφαρμογή.

3.2.4 Java

Η Java είναι μια γλώσσα προγραμματισμού γενικού σκοπού που αναπτύχθηκε από την Sun Microsystems και κυκλοφόρησε το 1995. Επηρεάζεται από την C++. Το κύριο κίνητρο πίσω από την Java ήταν να δημιουργήσει μια γλώσσα προγραμματισμού που επιτρέπει στους χρήστες της να «γράψουν μια φορά, να τρέξουν παντού». Ο τρόπος με τον οποίο λειτουργούσε η Java έναντι άλλων γλωσσών προγραμματισμού εκείνη την εποχή ήταν επαναστατικός. Αντί να μεταγλωττιστεί σε έναν συγκεκριμένο κώδικα μηχανής, ο μεταγλωττιστής Java μετατρέπει τον κώδικα σε κάτι που ονομάζεται Bytecode, το οποίο στη συνέχεια ερμηνεύεται από ένα λογισμικό που ονομάζεται Java Runtime Environment JRE, ή Java Virtual Machine JVM. Το JRE λειτουργεί ως εικονικός υπολογιστής που ερμηνεύει τον Bytecode και τον μεταφράζει για τον κεντρικό υπολογιστή. Εξαιτίας αυτού, ο κώδικας Java μπορεί να γραφτεί με τον ίδιο τρόπο για πολλές πλατφόρμες ("γράψτε μια φορά, εκτελέστε οπουδήποτε"), γεγονός που βοήθησε στη δημοτικότητά του για χρήση στο Διαδίκτυο, όπου πολλοί διαφορετικοί τύποι υπολογιστών μπορούν να ανακτήσουν την ίδια ιστοσελίδα. Η Java είναι η τρίτη γλώσσα που χρησιμοποιείται περισσότερο τα τελευταία 5 χρόνια στο GitHub. Διαφορετικές πλατφόρμες Java είναι διαθέσιμες ανάλογα με τον τύπο της εφαρμογής για την οποία θα χρησιμοποιηθεί η Java για την ανάπτυξη.

Υπάρχουν τέσσερις πλατφόρμες της γλώσσας προγραμματισμού Java [13]:

- Τα Java standard edition SE είναι το τυπικό API της γλώσσας μαζί με το JVM και το Java Compiler.
- Η εταιρική έκδοση Java EE είναι χτισμένη στην κορυφή της Java SE με επιπλέον στοιχεία για την υποστήριξη εταιρικών έργων που απαιτούν εφαρμογές δικτύου μεγάλης κλίμακας, πολλαπλών επιπέδων, επεκτάσιμων, αξιόπιστων και ασφαλών εφαρμογών.
- Το Java micro edition ME παρέχει ένα API και ένα JVM μικρού μήκους για την εκτέλεση εφαρμογών Java σε μικρές συσκευές, όπως κινητά τηλέφωνα. Το API είναι ένα υποσύνολο του Java SE API. Οι εφαρμογές Java ME είναι συχνά πελάτες υπηρεσιών πλατφόρμας Java EE.
- Το JavaFX είναι μια πλατφόρμα για τη δημιουργία πλούσιων διαδικτυακών εφαρμογών χρησιμοποιώντας ένα ελαφρύ API διεπαφής χρήστη.

3.2.5 Kotlin

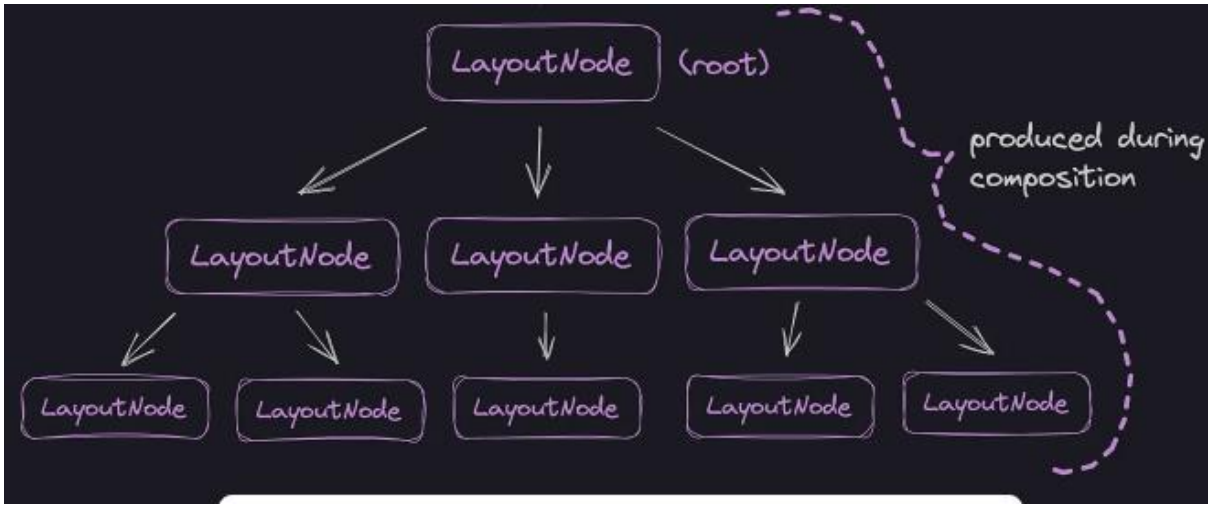
Είναι μια στατικά πληκτρολογημένη τεχνητή γλώσσα ανοιχτού κώδικα, δηλαδή οι τύποι μεταβλητών δηλώνονται ρητά και επομένως προσδιορίζονται κατά το χρόνο μεταγλώττισης. Σχεδιάστηκε από την JetBrains, που δημιούργησε και το IntelliJ IDEA, το καλύτερο IDE για JAVA. Η Kotlin είναι αντικειμενοστραφής και υποστηρίζει προγραμματισμό με μεθόδους που δημιουργεί διατηρητέο κώδικα. Έχει σχεδιαστεί για το JVM (Java Virtual Machine). Αυτό είναι πλήρως συμβατό με JAVA. Στην πραγματικότητα, η Kotlin είναι Java με μερικά επιπλέον κομμάτια. Οι δύο γλώσσες χρησιμοποιούνται συχνά παράλληλα στο ίδιο έργο. Για παράδειγμα, θα μπορούν να χρησιμοποιηθούν εύκολα JAVA βιβλιοθήκες σε έργο που είναι γραμμένο σε Kotlin. Επικεντρώνεται στη διαλειτουργικότητα, την ασφάλεια, τη σαφήνεια και υποστήριξη εργαλείων. Μπορεί να χρησιμοποιηθεί για να προγραμματίσει Back-end servers, Android εφαρμογές ή ακόμα και εφαρμογή iOS χρησιμοποιώντας Kotlin Native. Αυτό που την κάνει ανώτερη από την Java είναι η σύνταξη. Όταν εργαζόμαστε με την Kotlin, είμαστε σε θέση να συμπτύξουμε πολλές γραμμές κώδικα. Ο κώδικας της είναι πιο διαφανείς και ευθύς στην κατανόηση. Η JAVA θα μπορούσε να είναι μια πολύ ωραία γλώσσα, αλλά όσο το έργο επεκτείνεται, ο αριθμός των γραμμών κώδικα αυξάνεται λογαριθμικά. Τελικά, θα χρειάζεται πολύς χρόνος για να αναζητήσετε αυτό που ψάχνετε στον κώδικα. Με την Kotlin όλος αυτός ο χρόνος εξοικονομείται και μετατρέπεται σε παραγωγικό χρόνο [5].

3.2.6 Jetpack Compose

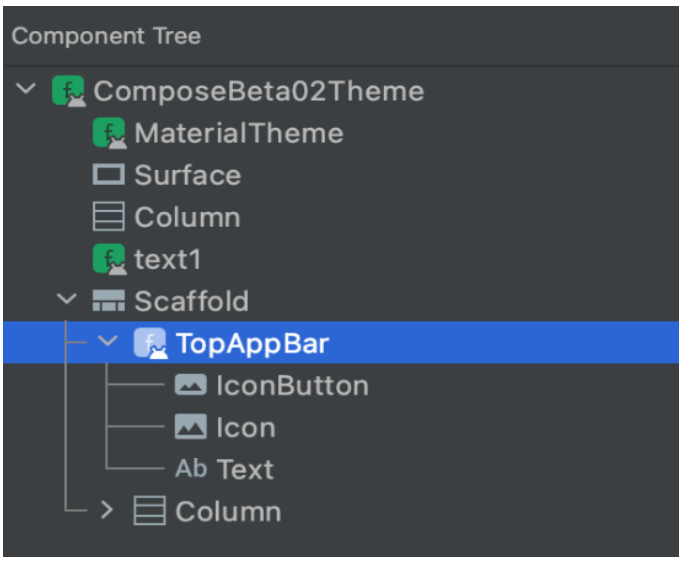
Οι προσδοκίες σχετικά με την ανάπτυξη διεπαφής χρήστη έχουν αυξηθεί από την δημιουργία του πρώτου UI Toolkit. Σήμερα, δεν μπορούμε να δημιουργήσουμε μια εφαρμογή και να ανταποκριθούμε στις ανάγκες του χρήστη, χωρίς να έχουμε ένα εξευγενισμένο περιβάλλον χρήστη με κινούμενα σχέδια και κίνηση. Αυτές οι απαιτήσεις δεν υπήρχαν όταν δημιουργήθηκε το τρέχον UI Toolkit. Για να αντιμετωπίσουμε τις τεχνικές προκλήσεις της δημιουργίας ενός γυαλιστερού περιβάλλοντος χρήστη γρήγορα και αποτελεσματικά, δημιουργήθηκε το Jetpack Compose. Μια σύγχρονη εργαλειοθήκη διεπαφής χρήστη που προσαρμόζει τους προγραμματιστές εφαρμογών σε αυτό το νέο τοπίο.

Η στρατηγική δημιουργίας αυτής της τεχνολογίας βασίστηκε στον διαχωρισμό των ανησυχιών (Separation of Concerns). Είναι μια πολύ γνωστή αρχή σχεδιασμού λογισμικού. Είναι ένα από τα θεμελιώδη πράγματα που μαθαίνουμε ως προγραμματιστές εφαρμογών. Παρά το γεγονός ότι είναι ευρέως γνωστό, είναι συχνά δύσκολο να κατανοήσουμε εάν αυτή η αρχή τηρείται ή όχι στην πράξη. Μπορεί να είναι χρήσιμο να σκεφτούμε αυτήν την αρχή με όρους «Σύζευξης» και «Συνοχής». Όταν γράφουμε κώδικα, δημιουργούμε ενότητες που αποτελούνται από πολλαπλές μονάδες. Η σύζευξη είναι η εξάρτηση μεταξύ των μονάδων, που ανήκουν σε διαφορετικές ενότητες, αντικατοπτρίζει τους τρόπους με τους οποίους τμήματα μιας μονάδας επηρεάζουν μέρη άλλων μονάδων. Για παράδειγμα όταν μια κλάση “A” στον αντικειμενοστραφή προγραμματισμό, χρειάζεται σαν παράμετρο ή σαν χαρακτηριστικό, άλλη κλάση “B”, τότε η κλάση “A” έχει εξάρτηση από την κλάση “B”. Η συνοχή αντίθετα, είναι η σχέση μεταξύ των μονάδων μέσα σε μια ενότητα και υποδεικνύει πόσο καλά ομαδοποιούνται οι μονάδες μέσα σε αυτήν.
Όταν γράφουμε λογισμικό για να παρέχουμε δυνατότητα συντήρησης, στόχος μας είναι να ελαχιστοποιήσουμε τη σύζευξη και να μεγιστοποιήσουμε τη συνοχή.
Έτσι ήρθε το Jetpack Compose που αντικατέστησε το XML layout για την σχεδίαση διεπαφών χρήστη. Παλαιότερα γράφαμε κώδικα για την σχεδίαση των οθονών της εφαρμογής σε XML, ενώ την λογική και την λειτουργικότητα της οθόνης σε Kotlin. Πλέον το Jetpack Compose μας επιτρέπει να

γράφουμε και τα δύο σε Kotlin. Αυτό συντέλεσε στην μείωση των αρχείων, στην καλύτερη οργάνωση του έργο και ως τελικό αποτέλεσμα στην καλύτερη συντήρηση του. Το Compose απαρτίζεται από μεθόδους που ονομάζονται Composable Functions, κάθε μια από την οποίες παράγει έναν κόμβο σε ένα UI δέντρο με ρίζα την πρώτη Composable μέθοδο που λέγεται SetContent {} και καλείται από την κλάση ComposeView που κληρονομεί την κλάση AndroidView, η οποία είναι μέρος του παλαιότερου UI Toolkit [18].



Σχήμα 3.3. Σημασιολογικό UI δέντρο Layout κόμβων κατά την εκτέλεση των Compose μεθόδων.

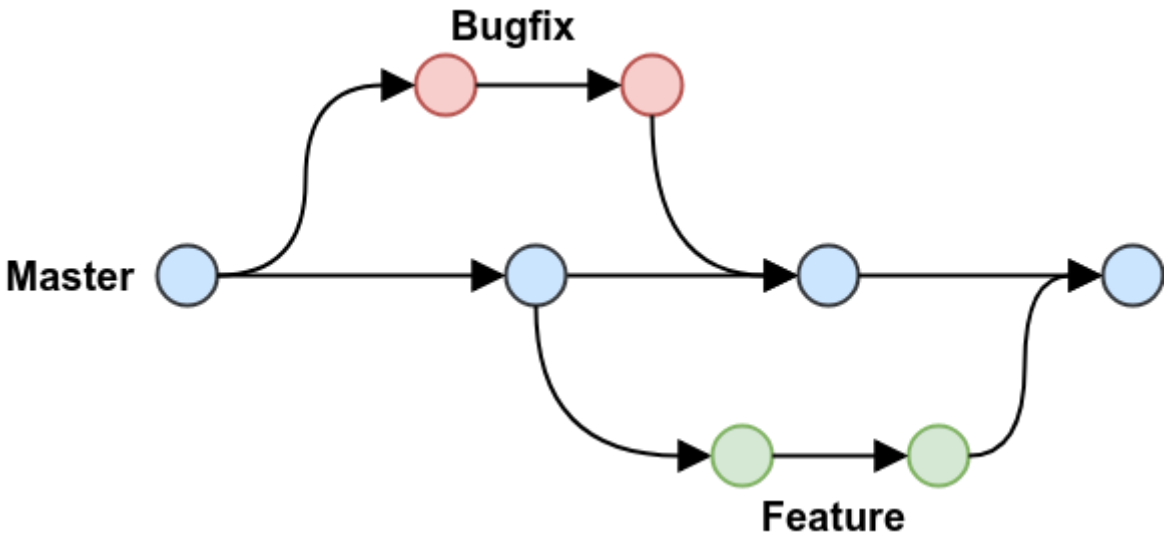


Σχήμα 3.4. Δέντρο UI συστατικών Compose μεθόδων (Composables).

3.2.7 GIT

Όταν εργαζόμαστε πάνω σε ένα έργο και χρειαζόμαστε ένα τρόπο να καταγράφουμε τις καταστάσεις του έργου σε διάφορα στιγμιότυπα στην διάρκεια της εξέλιξης του. Έτσι μπορούμε να ξέρουμε ποια αλλαγή οδήγησε σε ποια κατάσταση του συστήματος για να μπορούμε να επιστρέψουμε πίσω σε αυτήν. Παλαιότερα η μέθοδος που χρησιμοποιούσαμε ήταν να κρατάμε αντίγραφα των ίδιων αρχείων για παράδειγμα code_1.txt, code_2.txt, code_3.txt κτλ κάθε φορά που το αλλάξαμε. Αλλά επειδή αυτό

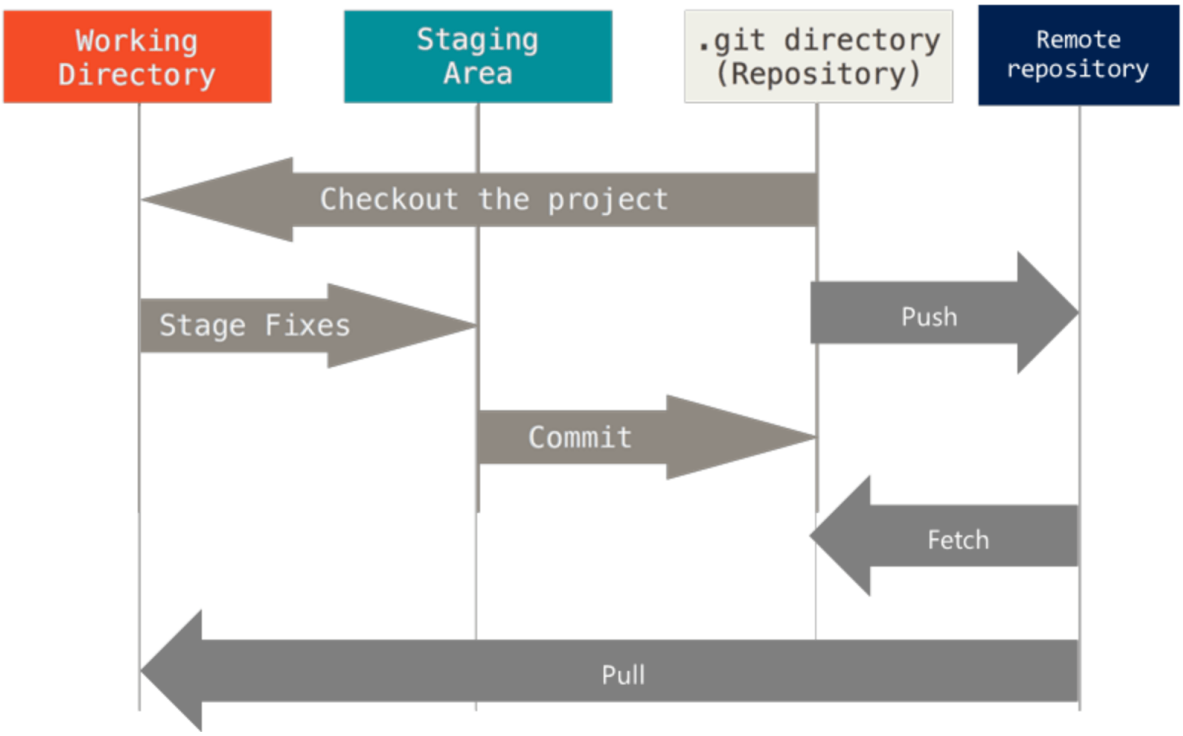
οδηγεί σε πάρα πολλές εκδοχές του ίδιου αρχείου και στην δυσκολία οργάνωση τους υπήρξε η ανάγκη ενός καλύτερου συστήματος ελέγχου έκδοσης του έργου. Γι'αυτό οι μηχανικοί λογισμικού δημιούργησαν το Git που είναι το πιο ευρέως διαδεδομένο τέτοιο σύστημα. Με την βοήθεια του Git μπορούμε να ελέγχουμε και να διαχειριζόμαστε τα διάφορα στιγμιότυπα κατάστασης του έργου. Ακόμα μπορούμε να δουλεύουμε και να συνεισφέρουμε πολλοί προγραμματιστές ταυτόχρονα στην εξέλιξη του ίδιου έργου.
Όπως φαίνεται στο σχήμα 2.4, το Master προσομοιώνει την κύρια διαδρομή, από την οποία ξεκινήσαμε να δημιουργούμε και να μεταποιούμε τα πρώτα αρχεία του έργου. Όταν κάποια στιγμή αποφασίσαμε να υλοποιήσουμε κάποιο καινούργιο χαρακτηριστικό στην εφαρμογή μας, για να μην επηρεάσουμε απευθείας την υπάρχων κατάσταση της, δημιουργήσαμε ένα καινούργιο branch (παρακλάδι) που λέγεται Feature (όνομα του χαρακτηριστικού). Σε αυτό πάνω δουλέψαμε μέχρι που το ολοκληρώσουμε και να ξανά ενώσαμε το παρακλάδι με την κύρια αποδεκτή εξέλιξη της εφαρμογής. Το ίδιο πράγμα κάναμε και με το παρακλάδι που δημιουργήσαμε για να διορθώσουμε ένα σφάλμα που υπήρχε στην εφαρμογή. Και έτσι αν κάποια στιγμή μετανιώσουμε και το χαρακτηριστικό που υλοποιήσαμε δεν το θέλουμε πια, απλά επιστρέφουμε στην στιγμή πριν δημιουργήσουμε το παρακλάδι και δεν χρειάζεται να αφιερώσουμε χρόνο διαγράφοντας κώδικα και αρχεία που είχαμε γράψει.



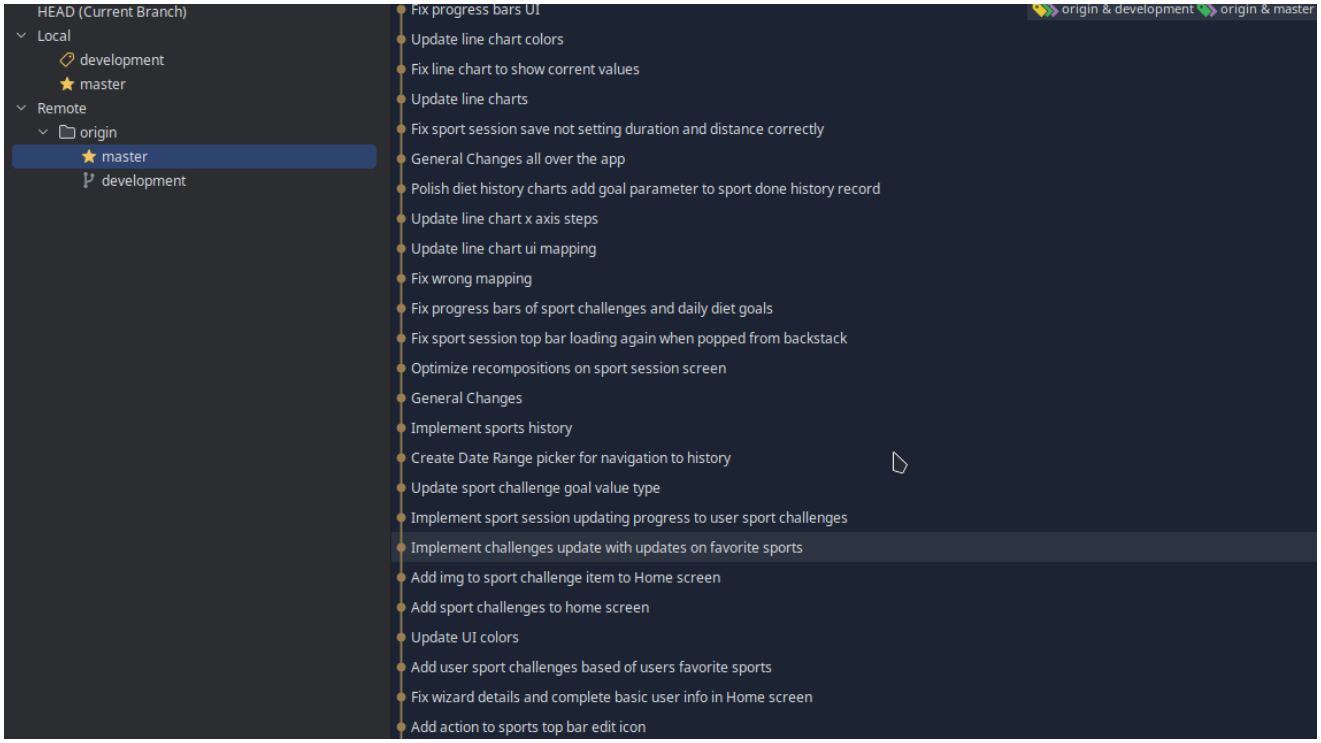
Σχήμα 3.5. Δημιουργία διαφορετικών παρακλαδίων για την διόρθωση σφάλματος ή την υλοποίηση νέου χαρακτηριστικού.

Για να ξεκινήσετε να ενσωματώνεται το Git στον κώδικα σας, μεταβείτε στον νέο σας φάκελο που περιέχει όλα τα αρχεία που θέλετε να εκδίδεται σε στιγμιότυπα. Ανοίξτε την γραμμή εντολών και εκτελέστε την εντολή git init για να αρχικοποιήσετε τον τρέχοντα φάκελο ως αποθετήριο Git. Στην συνέχεια όταν θέλετε να αποθηκεύσετε ένα στιγμιότυπο αυτού του φακέλου εκτελέστε git add όλα τα αρχεία στην περιοχή σταδίου που θέλετε να δεσμεύσετε (για παράδειγμα, έχετε ενημερώσει το αρχείο process.kt). Δεύτερον, πληκτρολογήστε git commit με ένα μήνυμα. Θα δεσμευτούν μόνο τα αρχεία που προστέθηκαν στην περιοχή σταδιοποίησης. Όλα τα προηγούμενα commits βρίσκονται στον κρυφό κατάλογο .git στο αποθετήριο σας. Έπειτα αν θέλετε να δουλέψετε στον ίδιο φάκελο με κάποιον συνάδελφο χρησιμοποιήστε ένα απομακρυσμένο αποθετήριο. Έτσι κάθε φορά που θέλετε να συνεχίσετε να δουλέψετε πάνω στις τελευταίες αλλαγές που έκανε ο συνάδελφός σας χρησιμοποιήστε

την εντολή git pull για να ενημερωθούν τα τοπικά αρχεία από το απομακρυσμένο αποθετήριο. Το σχήμα 5 δείχνει αυτές τις ενέργειες.



Σχήμα 3.6. Βασικές ενέργειες χρήσης απομακρυσμένου αποθετηρίου Git



Σχήμα 3.7. Git δέντρο μέσα από το Android Studio λογισμικό της εφαρμογής Thesis Fitness.

3.2.8 GitHub

Όπως αναφέραμε παραπάνω το απομακρυσμένο αποθετήριο αποτελεί ο πλέον διαδεδομένος τρόπος να γράφετε κώδικα ομαδικά. Η πιο δημοφιλής ιστοσελίδα που παρέχει την υπηρεσία φιλοξενίας αποθετηρίων είναι το GitHub. Δημιουργώντας έναν δωρεάν λογαριασμού η υπηρεσία είναι ανοιχτή προς όλους και μπορεί να συνδεθεί με το Android Studio για να μπορούμε να επαναλαμβάνουμε τις ενέργειες του Git εύκολα και γρήγορα.

3.2.9 Kotlin Coroutines

Σε πραγματικά έργα χρειαζόμαστε πολλές φορές να τρέξουμε κάτι στον κώδικα ασύγχρονα. Μόλις πάρουμε το αποτέλεσμα της ασύγχρονης διαδικασίας κάποια στιγμή στο μέλλον, έπειτα να συνεχίσουμε με την ροή της εφαρμογής. Για παράδειγμα μια συχνή αλληλουχία είναι να καλέσουμε ένα API για να πάρουμε κάποια δεδομένα που θέλουμε να δείξουμε στην οθόνη του κινητού. Όσο περιμένουμε για αυτά τα δεδομένα (internet latency) θέλουμε ο χρήστης να βλέπει έναν loader που φορτώνει. Έπειτα μόλις λάβουμε τα δεδομένα να εξαφανίσουμε τον loader και να εμφανίσουμε τα δεδομένα σε κάποια ευανάγνωστη μορφή. Για να υλοποιήσουμε μία τέτοια περίπτωση χρήσης (Use Case) αλλά και πολλές ακόμα στο Android χρησιμοποιούμε τα Coroutines της Kotlin. Η υποστήριξη των κορουτίνων από την Kotlin ξεκίνησε τον Οκτώβριο του 2018 με την έκδοση 1.3. Η υπεροχή τους απέναντι σε προηγούμενες υλοποιήσεις ασύγχρονου κώδικα όπως RxJava (JVM Threads) είναι η απλότητα και η ευκολία στην συγγραφή τους. Χρειαζόμαστε απλά ένα πλαίσιο στο οποίο μπορεί να μένει “ανοιχτή” η αναμονή του αποτελέσματος της κορουτίνας, ένα thread στο οποίο θα τρέχει η επεξεργασία της κορουτίνας (UI Thread, Main Thread, Default Thread ή IO Thread) και δύο σγουρές αγκύλες ({}) που περικλείουν το κομμάτι κώδικα που θέλουμε να τρέχει ασύγχρονα.

3.2.10 ProGuard

Κατά τη σύνταξη του κώδικα για μια εφαρμογή Android, ενδέχεται να υπάρχουν ορισμένες γραμμές κώδικα που δεν είναι απαραίτητες και αυξάνουν το μέγεθος του APK της εφαρμογής. Εκτός από τον άχρηστο κώδικα, μπορεί να υπάρχουν πολλές βιβλιοθήκες ενσωματωμένες στον πηγαία κώδικα αλλά να μην χρησιμοποιούνται όλες οι λειτουργίες που παρέχουν. Επίσης, μπορεί να υπάρχει κώδικας που θα είναι παρωχημένος στο μέλλον και να μην γίνεται να διαγραφεί εύκολα. Αυτοί οι παράγοντες ευθύνονται για το αυξημένο μέγεθος του APK της εφαρμογής. Το Android έχει την ικανότητα με το Proguard να ελαχιστοποιεί το μέγεθος του APK. Το ProGuard είναι μια δωρεάν εφαρμογή Java για Android που μας επιτρέπει να κάνουμε τα εξής:

- Μείωση (ελαχιστοποίηση) του κώδικα: Ο μη χρησιμοποιημένος κώδικας στο έργο πρέπει να αφαιρεθεί.
- Συσκότιση κώδικα: Μετονομασία των ονομάτων κλάσεων, πεδίων και ούτω καθεξής.
- Βελτίωση του κώδικα: π.χ. Επανασυγγραφή μεθόδων σε μια γραμμή (inline functions)

Τα κύρια οφέλη που μας προσφέρουν αυτές οι λειτουργίες του ProGuard είναι ότι:

- Μειώνει το μέγεθος της εφαρμογής.
- Καταργεί τις αχρησιμοποίητες κλάσεις και μεθόδους που συμβάλλουν στο όριο μέτρησης μεθόδων 64K μιας εφαρμογής Android.
- Η συσκότιση του κώδικα, καθιστά δύσκολη την αντίστροφη μηχανική της εφαρμογής (reverse engineering).

3.2.11 R8

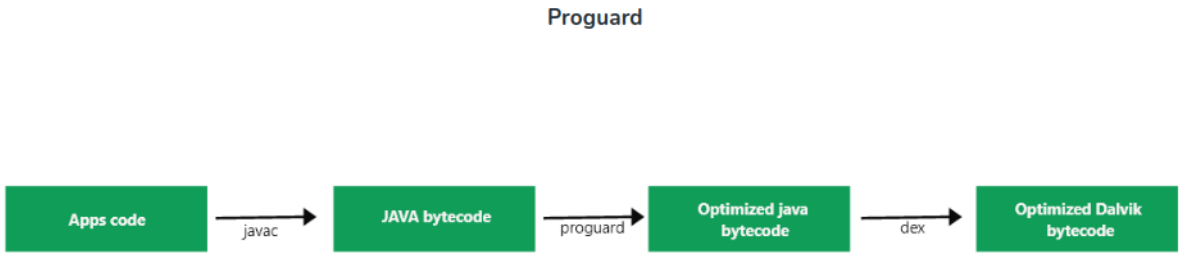
Το R8 είναι ένα άλλο εργαλείο που θα μετατρέψει τον κώδικα java byte σε μια βελτιστοποιημένη μορφή κώδικα dex. Θα ελέγξει ολόκληρη την εφαρμογή και θα αφαιρέσει τις αχρησιμοποίητες κλάσεις και μεθόδους. Μας βοηθά να μειώσουμε το μέγεθος του APK μας και να κάνουμε την εφαρμογή μας πιο ασφαλή. Το R8 χρησιμοποιεί παρόμοιους ProGuard κανόνες για να τροποποιήσει την προεπιλεγμένη συμπεριφορά του.

Η εφαρμογή Android που έχει Gradle plugin έκδοση πάνω από 3.4.0 ή νεότερη, τότε το έργο χρησιμοποιεί το R8 από προεπιλογή μόνο με κανόνες Proguard.

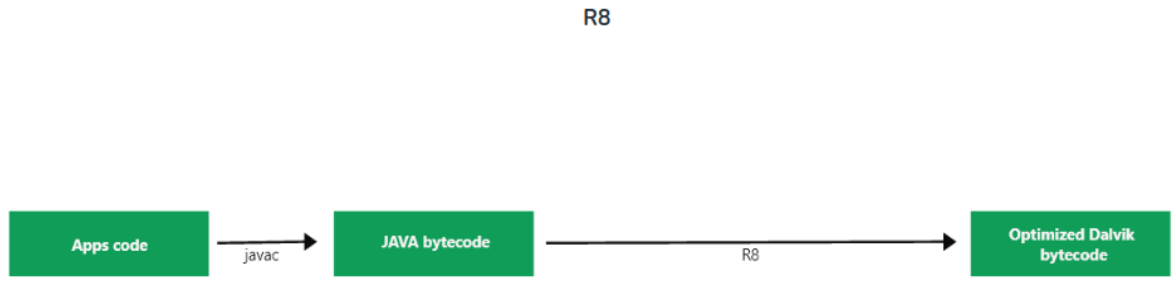
Διαφορά μεταξύ Proguard και R8.

- Το R8 έχει περισσότερη υποστήριξη Kotlin από αυτή της Proguard.
- Το R8 έχει ταχύτερο χρόνο επεξεργασίας από το Proguard, γεγονός που μειώνει τον χρόνο κατασκευής.
- Το R8 δίνει καλύτερα αποτελέσματα εξόδου από το Proguard.
- Το R8 μειώνει το μέγεθος της εφαρμογής κατά 10%, ενώ η Proguard μειώνει το μέγεθος της εφαρμογής κατά 8,5%.

(Difference Between Proguard and R8 in Android, 2021)



Σχήμα 3.8. Η διαδικασία μετατροπής Kotlin κώδικα σε βελτιστοποιημένο κώδικα με Proguard



Σχήμα 3.9. Η διαδικασία μετατροπής Kotlin κώδικα σε βελτιστοποιημένο κώδικα με R8

3.2.12 Dagger Hilt

Το Dagger Hilt είναι ένα dependency injection (DI) framework για το Android που απλοποιεί και βελτιστοποιεί τη διαδικασία διαχείρισης των εξαρτήσεων σε εφαρμογές Android [30]. Είναι χτισμένο πάνω στη δημοφιλή βιβλιοθήκη DI Dagger 2 και σχεδιάστηκε ειδικά για την ανάπτυξη εφαρμογών Android. Ακολουθεί τις αρχές του dependency injection, επιτρέποντάς να ορίζονται οι εξαρτήσεις που χρειάζονται τα συστατικά μιας Android εφαρμογής για να τις παρέχει αργότερα αυτόματα σε αυτά. Αυτό προάγει καθαρότερο, πιο αρθρωτό και εύκολα εξεταζόμενο κώδικα. Το Dagger Hilt ενσωματώνεται άψογα με τα συστατικά του Android που έχουν συγκεκριμένο κύκλο ζωής, όπως Activities, Fragments, ViewModels, Services και άλλα. Αυτή η ενσωμάτωση απλοποιεί τη διαδικασία εγκατάστασης και διαχείρισης των εξαρτήσεων εντός αυτών των συστατικών. Το Dagger Hilt βασίζεται σε ένα σύνολο ευκολόχρηστων ετικετών (λέξεις που ακολουθουν το @) για να δηλώνονται και να διαχειρίζονται οι εξαρτήσεις. Ορισμένες από τις συχνά χρησιμοποιούμενες ετικέτες είναι το @HiltAndroidApp, @AndroidEntryPoint και @Inject όπως θα δείξω και στην ανάλυση του κώδικα της εφαρμογής που ακολουθεί. Το Dagger Hilt απλοποιεί τη διαδικασία, που ο προκάτοχος Dagger 2 επιτύγχανε συχνά με εκτεταμένες ενότητες Dagger για να ορίσει συνδέσεις και να παρέχει εξαρτήσεις ρητά. Δημιουργεί αυτόματα ενότητες για τα στοιχεία του Android, μειώνοντας τον αντίστοιχο υπερβολικό κώδικα. Το Dagger Hilt παρέχει ένα σύνολο προκαθορισμένων και προσαρμόσιμων Scopes για τα συστατικά-κλάσεις του Android, επιτρέποντάς την διαχείριση του κύκλο ζωής των εξαρτήσεων. Συχνά χρησιμοποιούμενα Scopes περιλαμβάνουν τα @Singleton, @ActivityScoped και @FragmentScoped. Επιπροσθέτως, το framework αυτό απλοποιεί την ενσωμάτωση των ViewModel στα UI συστατικά του Android όπως τα Activities, τα Fragments και τα Composables. Παρέχει αυτόματα τα αντικείμενα ViewModel με το σωστό πεδίο ορατότητας και διασφαλίζει ότι διατηρούνται ανάλογα μετά από κάποιο configuration change του κινητού, όπως τον προσανατολισμό της οθόνης από portrait σε landscape και το αντίθετο. Το Dagger Hilt καθιστά ευκολότερη την εγγραφή unit tests για τα στοιχεία του Android παρέχοντας ενσωματωμένη υποστήριξη για test κλάσεις. Υποστηρίζει την δημιουργία test modules για να αντικαθίστανται οι εξαρτήσεις για λόγους δοκιμής. Η προσθήκη του Dagger Hilt σε ένα έργο Android είναι απλή, χάρη στο Gradle plugin. Ένα από τα σημαντικά χαρακτηριστικά του είναι ότι είναι ανοιχτού κώδικα και συντηρείται από τη Google και την κοινότητα του Android. Αυτό εξασφαλίζει ότι θα λαμβάνει ενημερώσεις και βελτιώσεις με την πάροδο του χρόνου και ότι θα έχει ισχυρή υποστήριξη από την κοινότητα. Η ανάπτυξη του συνεχίζει να εξελίσσεται με ενημερώσεις και βελτιώσεις, κάνοντάς το μια αξιόπιστη επιλογή για τη διαχείριση των εξαρτήσεων στη σύγχρονη ανάπτυξη εφαρμογών Android.

3.2.13 Vico

Το Vico είναι μια ελαφριά και επεκτάσιμη βιβλιοθήκη γραφικών για το Android [35]. Είναι συμβατό με το Jetpack Compose και το Android View System αλλά τα δύο κύρια του πεδία εφαρμογής, composables και views, είναι ανεξάρτητα.

Ως βιβλιοθήκη που είναι συμβατή και με το Jetpack Compose και με το Android View System, το Vico είναι αρκετά μοναδικό. Δεν εξαρτάται από την αλληλεπικοινωνία μεταξύ των δύο συστημάτων διεπαφής χρήστη (UI). Η κύρια κοινή λογική βρίσκεται στον πυρήνα (core) του και εξαρτάται από το Android SDK. Δεν γνωρίζει τίποτα για τα composables ή τα views. Επίσης, η ενότητα Compose δεν εξαρτάται από το View System και η ενότητα views δεν εξαρτάται από το Jetpack Compose.

Ο πυρήνας (core) χρησιμοποιεί την android.graphics.Canvas (που χρησιμοποιείται επίσης από τις προβολές) για το σχεδιασμό των γραφημάτων, και το DrawScope (που χρησιμοποιείται από το Jetpack Compose) εκθέτει ένα αντίγραφο της android.graphics.Canvas μέσω του DrawScope#canvas#nativeCanvas. Είναι παρόμοιο και για άλλες διεπαφές προγραμματισμού, όπως η Path. Αυτή η προσέγγιση ενθαρρύνει ένα υψηλότερο επίπεδο αφαιρετικότητας και προωθεί την σχεδιαστική αρχή του διαχωρισμού των ανησυχιών(SOP - separation of concerns). Επίσης, έκανα το API του Vico εξαιρετικά επεκτάσιμο. Την εν λόγω βιβλιοθήκη την χρησιμοποίησα για να απεικονίσω τα δεδομένα ιστορικού αθλητικής δραστηριοποίησης.

3.2.14 YCharts

Το Y-Charts είναι μια βιβλιοθήκη για γραφήματα και διαγράμματα που βασίζεται στο Jetpack Compose και επιτρέπει στους προγραμματιστές να ενσωματώνουν ομαλά στην υπάρχουσα διεπαφή χρήστη διάφορα είδη γραφημάτων και διαγραμμάτων για την οπτική αναπαράσταση στατιστικών δεδομένων [36]. Το Jetpack Compose είναι ένα σύγχρονο εργαλείο του Android για την κατασκευή φυσικών διεπαφών χρήστη. Εκμεταλλεύεται τις ισχυρές δυνατότητες του Jetpack Compose για την παροχή ακριβών και χρήσιμων αποτελεσμάτων, και οι πολλές επαναχρησιμοποιήσιμες συνιστώσες του εξασφαλίζουν αποτελεσματικότητα και απλότητα, ενώ παράλληλα παρέχουν στους προγραμματιστές την ευελιξία να εξατομικεύουν και να εισάγουν νέα γραφήματα με ευκολία. Υποστηρίζει:

- Καρτεσιανούς πίνακες, γνωστούς επίσης ως συντεταγμένους πίνακες ή XY-γραφικούς πίνακες που χρησιμοποιούν ένα ορθογώνιο σύστημα συντεταγμένων με δύο κάθετους άξονες: τον άξονα x (οριζόντιο) και τον άξονα y (κατακόρυφο). Αυτοί οι πίνακες είναι ιδιαίτερα αποτελεσματικοί για την απεικόνιση ποσοτικών δεδομένων και σχέσεων μεταξύ μεταβλητών.
- Πολικούς γραφικούς πίνακες, γνωστοί και ως ακτινικοί πίνακες που χρησιμοποιούν ένα πολικό σύστημα συντεταγμένων με ένα κεντρικό σημείο και ακτινικούς άξονες. Αυτοί οι πίνακες είναι αποτελεσματικοί για την αναπαράσταση κυκλικών ή γωνιακών δεδομένων, καθώς και για τη σύγκριση πολλαπλών μεταβλητών γύρω από ένα κεντρικό σημείο. Αυτή την κατηγορία γραφήματος χρησιμοποιώ στην εφαρμογή για την απεικόνιση ιστορικού διατροφής

3.2.15: BACK-END - Firebase

Μια mobile εφαρμογή όπως όλες η εφαρμογές έχει ως στόχο να λάβει και να επεξεργαστεί μεγάλους όγκους δεδομένων τυχαίου τύπου, όπως βίντεο, φωτογραφίες, ήχο, κείμενο, αρχεία κ.α. Είναι δύσκολο για ένα σύστημα διαχείρισης σχεσιακής βάσης δεδομένων (RDBMS) να διαχειριστεί αυτά τα δεδομένα. Το Firebase είναι μια νέα τεχνολογία για τη διαχείριση μεγάλων ποσοτήτων τυχαίων δεδομένων. Πολύ γρηγορότερο σε σύγκριση με το RDBMS. Το Google Firebase είναι ένα λογισμικό ανάπτυξης εφαρμογών που υποστηρίζεται από την Google και επιτρέπει στους προγραμματιστές να αναπτύσσουν εφαρμογές IOS, Android και Web. Το Firebase παρέχει εργαλεία για την παρακολούθηση αναλυτικών στοιχείων, την αναφορά και τη διόρθωση σφαλμάτων εφαρμογών, δημιουργία πειράματος μάρκετινγκ και προϊόντων.

Analytics: Το Google Firebase Analytics προσφέρει δωρεάν, απεριόριστες αναφορές σε έως και 500 ή περισσότερα ξεχωριστά συμβάντα κατά την χρήση μιας εφαρμογής. Τα στατιστικά στοιχεία

παρουσιάζουν περιεχόμενο σχετικά με τη συμπεριφορά των χρηστών σε εφαρμογές IOS και Android για την καλύτερη δυνατή λήψη αποφάσεων σχετικά με τη βελτίωση της απόδοσης μιας εφαρμογής και το μάρκετινγκ.

Authorization: Το Firebase Authorization διευκολύνει τους προγραμματιστές να προστατεύουν τα συστήματα ελέγχου ταυτότητας και βελτιώνει την εμπειρία σύνδεσης και επιβίβασης σε μια εφαρμογή για τους χρήστες. Αυτή η δυνατότητα προσφέρει μια λύση ταυτότητας all-in-one, υποστήριξη λογαριασμών email και κωδικού πρόσβασης, έλεγχος ταυτότητας τηλεφώνου, σύνδεση μέσω Google, Facebook, GitHub, Twitter κι αλλα.

Crashlytics: Το Firebase Crashlytics μπορεί να είναι ένας αγγελιοφόρος crash σε πραγματικό χρόνο που βοηθά τους μηχανικούς να παρακολουθούν, να δίνουν προτεραιότητα και να επιδιορθώνουν προβλήματα που μειώνουν την ποιότητα των εφαρμογών τους. Με το Crashlytics, οι μηχανικοί ξοδεύουν λιγότερο χρόνο στον σχεδιασμό και στην επίλυση σφαλμάτων, μακροχρόνιων λειτουργιών των εφαρμογών τους [4].

Firestore: Το Firebase Firestore είναι η υπηρεσία ειδικά για εφαρμογές πελατών, η οποία είναι βαθιά ενσωματωμένη με τα άλλα προϊόντα Firebase που αναφέρθηκαν. Είναι μια βάση δεδομένων NoSQL χωρίς διακομιστή που φιλοξενείται και διαχειρίζεται πλήρως από τη Google. Μας επιτρέπει να δημιουργήσουμε μια βάση δεδομένων υψηλής απόδοσης σε λίγα λεπτά. Δεν χρειάζεται να ανησυχούμε για τη διαχείριση των διακομιστών και την επεκτασιμότητα.
Το Cloud Firestore είναι ο διάδοχος της Firebase Realtime Database, με μεγαλύτερη ευελιξία και επεκτασιμότητα. Το Cloud Firestore χρησιμοποιείται για την ανάπτυξη εφαρμογών για κινητές συσκευές, ιστού και IoT για να συγχρονίζει δεδομένα άμεσα σε όλες τις συσκευές χρησιμοποιώντας συσκευές ακρόασης σε πραγματικό χρόνο. Χρησιμοποιώντας εγγενή SDK που παρέχονται από το firebase, οι εφαρμογές για κινητά, web και IoT μπορούν να έχουν απευθείας πρόσβαση στη βάση δεδομένων [14].

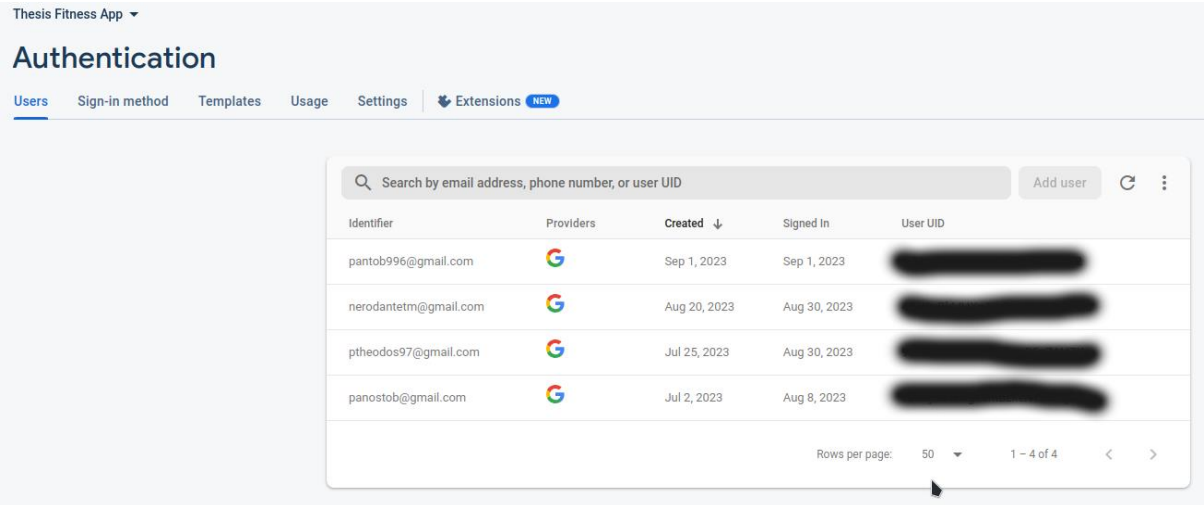
Κεφάλαιο 4ο: Ανάπτυξη Εφαρμογής και Περιγραφή Λειτουργίας

4.1 ΑΝΑΠΤΥΞΗ BACK-END

Για την ανάπτυξη της υπηρεσίας του Back-End χρησιμοποίησα την υπηρεσία Firebase της Google. Η υπηρεσία αυτή έχει δημιουργηθεί για προγραμματιστές που δεν θέλουν να μπλέξουν με την πολυπλοκότητα ανάπτυξης μιας back-end υποδομής. Συνεργάζεται τέλεια με την ανάπτυξη μιας Android Mobile εφαρμογής, καθώς είναι και οι δύο πλατφόρμες της ίδιας εταιρείας και το Firebase δημιουργήθηκε για να πλαισιώνει το Android.

Χρειάστηκε να μελετήσω την αρχιτεκτονική του Firebase Firestore, το οποίο έπαιξε τον ρόλο της βάσης δεδομένων της back-end υπηρεσίας μου. Το μόνο μειονέκτημα αυτής της επιλογής, ήταν ότι έπρεπε να δώσω προσοχή στην σχεδίαση των οντοτήτων και στις εγγραφές που θα αποθηκεύω. Το Firebase χρησιμοποιεί Collection, που περιέχουν σύνολα εγγραφών, με το όνομα Documents. Έτσι έπρεπε να αποφασίσω προσεκτικά τι δεδομένα θα περιέχει κάθε οντότητα, που αντιπροσωπεύουν αυτά τα Documents.

Για την αποθήκευση μεγάλων δεδομένων, όπως τις εικόνες της εφαρμογής χρησιμοποίησα το Firebase Storage. Είναι η αποθήκη που παρέχει η Google, για δεδομένα μεγάλου μεγέθους. Τέλος, απαραίτητη ήταν η αυθεντικοποίηση του χρήστη και η επόμενη ενθύμηση της ύπαρξης του, κάθε φορά που ανοίγει την εφαρμογή. Για αυτήν την λειτουργία, βασίστηκα στην υπηρεσία Firebase Authentication, η οποία συνδέεται με τον Google Sign In Client. Με τις πληροφορίες που παίρνει από την Google Account σύνδεση του χρήστη, δημιουργεί εγγραφή στον πίνακα χρηστών της εφαρμογής.



Σχήμα 4.1. Πίνακας εγγεγραμμένων χρηστών μέσα στο Firebase Authentication της υπηρεσίας της εφαρμογής

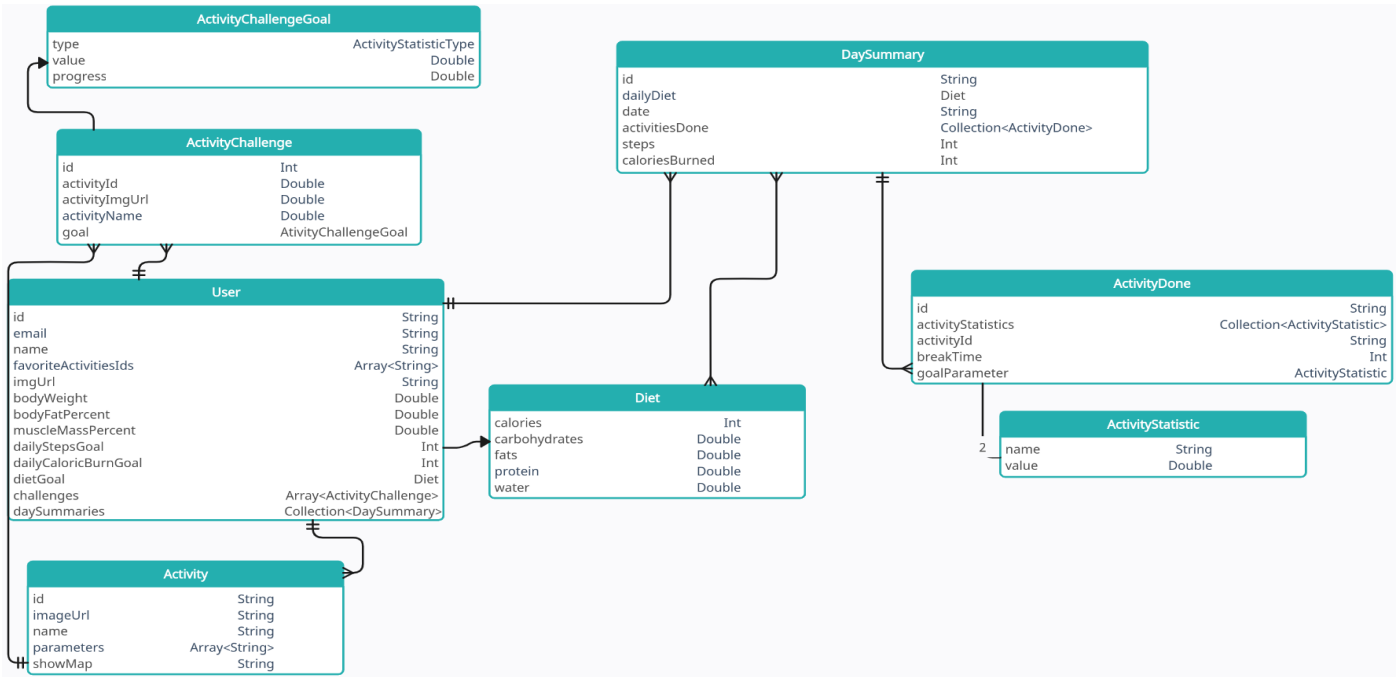
4.1.1 ER Diagram

Ένα διάγραμμα σχέσης οντοτήτων (ERD – Entity Relationship Diagram) μπορεί να περιγραφεί ως μια παραλλαγή ενός διαγράμματος ροής, που αναπαριστά οπτικά τις συνδέσεις μεταξύ οντοτήτων, οι οποίες μπορούν να περιλαμβάνουν άτομα, αντικείμενα ή έννοιες, σε ένα δεδομένο σύστημα. Αυτά τα διαγράμματα βρίσκουν την κύρια εφαρμογή τους, στη δημιουργία και την αντιμετώπιση προβλημάτων σχεσιακών βάσεων δεδομένων, που κυριαρχούν σε τομείς όπως, η μηχανική

λογισμικού, τα συστήματα πληροφοριών επιχειρήσεων, η εκπαίδευση και η έρευνα. Αναγνωρισμένα από εναλλακτικά ονόματα όπως ERD ή ER Models, χρησιμοποιούν ένα συγκεκριμένο σύνολο συμβόλων όπως ορθογώνια, διαμάντια, οβάλ και γραμμές σύνδεσης. Αυτή η οπτικοποιημένη γλώσσα, αποτυπώνει αποτελεσματικά τις αλληλεξαρτήσεις μεταξύ των οντοτήτων, των σχέσεων και των σχετικών χαρακτηριστικών τους, αντικατοπτρίζοντας στενά τη δομή της γραμματικής, όπου οι οντότητες είναι παρόμοιες με ουσιαστικά και οι σχέσεις παρόμοιες με τα ρήματα.

4.1.2 Σχεδίαση Βάση Δεδομένων - Προσδιορισμός Οντοτήτων

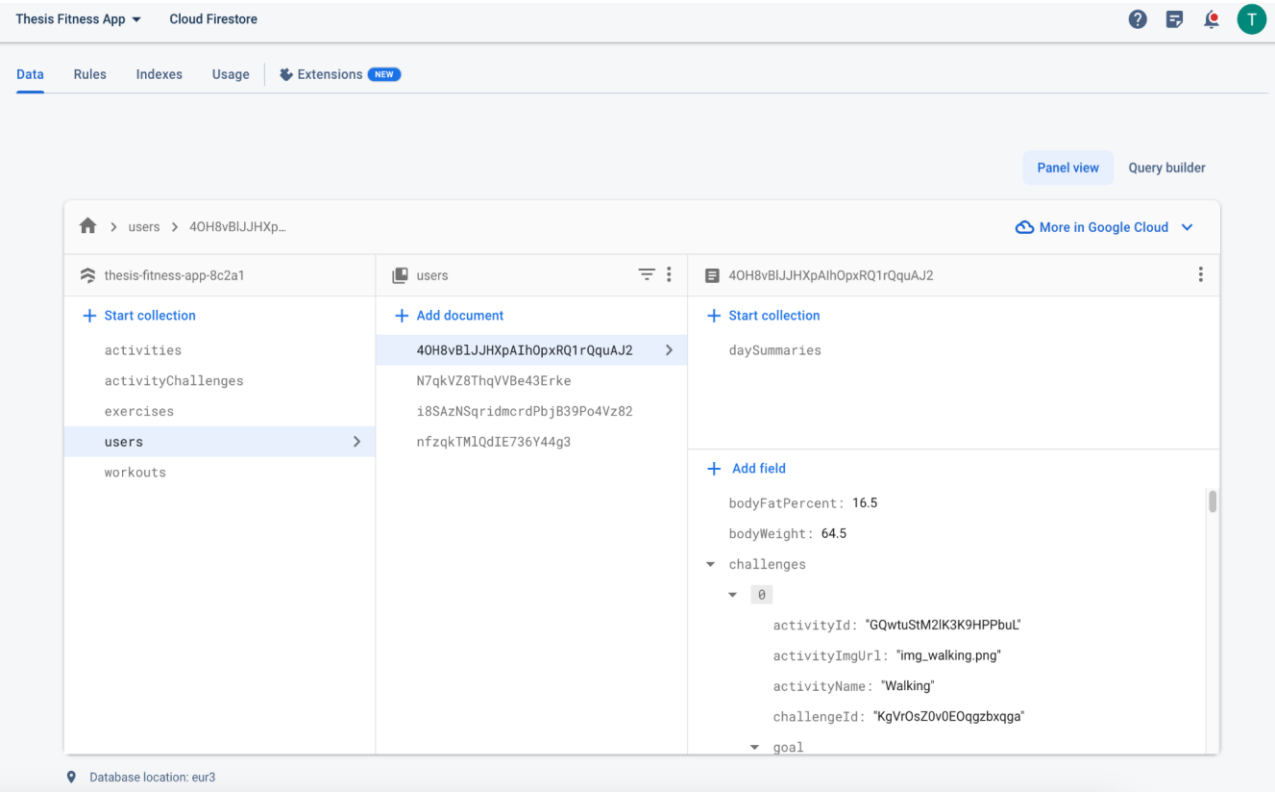
Στο ER διάγραμμα 4.2 παρουσιάζουμε με ποιον τρόπο ο χρήστης, που είναι η κύρια οντότητα της εφαρμογής μας, σχετίζεται με τους υπόλοιπους εννοιολογικούς πίνακες της βάσης μας στο Firebase Firestore.



Σχήμα 4.2: ER Διάγραμμα της εφαρμογής Fitness

Ο χρήστης διαφοροποιείται από τους υπόλοιπους με το id του Document, που τον εκπροσωπεί στο Firestore collection “users”. Όπως φαίνεται στο σχήμα 4.3 οι χρήστες είναι αποθηκευμένοι σαν documents, μέσα στην συλλογή (collection) των χρηστών. Επιλέγοντας την συλλογή των χρηστών, εμφανίζεται η στήλη δεξιά με τα ids των documents και αντίστοιχα των χρηστών, το ένα κάτω από το άλλο. Αφού διαλέξουμε ένα από αυτά τα ids, εμφανίζεται δεξιά η τρίτη στήλη, που χωρίζεται σε δύο τμήματα. Το πάνω μέρος δείχνει τις εσωτερικές συλλογές που ανήκουν στην οντότητα του χρήστη. Το κάτω τμήμα δείχνει τα υπόλοιπα στοιχεία του χρήστη, που βρίσκονται στο ίδιο επίπεδο. Αναφέρω τον όρο επίπεδο, για να επισημάνω ότι οι εσωτερική συλλογή “daySummaries”, βρίσκεται ένα επίπεδο πιο μέσα από τις υπόλοιπες πληροφορίες του χρήστη (bodyFatPercent, bodyWeight κ.τ.λ.). Η σημασία του επιπέδου παίζει ρόλο στα Get αιτήματα, που ασκούμε από την mobile εφαρμογή στον χρήστη μας. Όταν στέλνουμε ένα get αίτημα στην υπηρεσία του Firestore για να πάρουμε έναν χρήστη, οι πληροφορίες που επιστρέφονται είναι μόνο αυτές που ανήκουν στο πρώτο επίπεδο. Αυτό δείχνει ότι τα αιτήματα στο Firestore είναι “ρηχά”, που σημαίνει ότι οι εγγραφές από τα εσωτερικά collections (π.χ. το DaySummary από τα daySummaries) δεν λαμβάνονται με ένα μόνο αίτημα. Θα

πρέπει να γίνει δεύτερο αίτημα, που θα διαλέγει πρώτα τον χρήστη και μετά την συλλογή daySummaries, για να πάρουμε τα εσωτερικά documents.



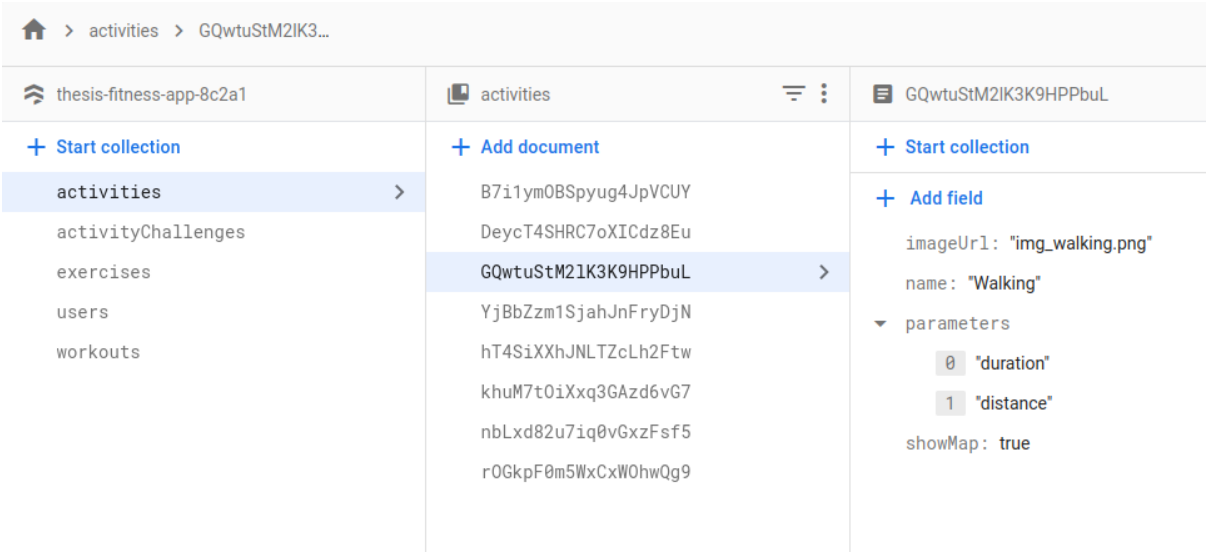
Σχήμα 4.3. Η συλλογή των χρηστών μέσα στην Firestore βάση, με επιλεγμένο τον πρώτο χρήστη.

Συνεχίζοντας λοιπόν με τις υπόλοιπες οντότητες, αφού το αναφέραμε παραπάνω, υπάρχει το DaySummary. Αυτή η οντότητα αντιπροσωπεύει την περίληψη της δραστηριότητας του στην διάρκεια μιας ημέρας. Αυτή η οντότητα χρησιμεύει στην εμφάνιση του ιστορικού του χρήστη, εντός της εφαρμογής. Βλέποντας το σχήμα 4.2, περιέχει πληροφορίες όπως την σημερινή λήψη μακροθρεπτικών στοιχείων, τις σημερινές αθλητικές δραστηριότητες που ολοκλήρωσε, ημερήσια βήματα και θερμίδες που έκαψε, και τέλος την ημερομηνία της ημερας, που αφορά αυτές τις πληροφορίες.

Κάθε ημερήσια λήψη μαρκοθρεπτικών, αποτελεί με την σειρά της μια οντότητα με όνομα “Diet”. Τα πεδία του Diet είναι συγκεκριμένα. Οι θερμίδες, η πρωτεΐνη, οι υδατάνθρακες, τα λίπη και το νερό που κατανάλωσε ο χρήστης για μια συγκεκριμένη μέρα. Εκτός από το DaySummary, το Diet χρησιμοποιείται και μέσα στον χρήστη ως DietGoal, που αντιπροσωπεύει τον στόχο που έχει βάλει ο χρήστης για κάθε μέρα, όσον αφορά τα μακροθρεπτικά που θέλει να λαμβάνει.

Για να ολοκληρώσω με το DaySummary, πρέπει να περιγράψω πρώτα μια άλλη βασική οντότητα της εφαρμογής. Αυτή η οντότητα ονομάζεται “Activity” και ανήκει στην top level συλλογή “activities”. Κάθε ένα Activity έχει ένα όνομα και μια εικόνα, για να αναγνωρίζεται εύκολα από τα υπόλοιπα. Ακόμα, διαθέτει την showMap Boolean μεταβλητή, που όταν υπάρχει και είναι true, σημαίνει ότι σε αυτό το άθλημα, όταν ξεκινάει να καταγράφεται από τον χρήστη, εμφανίζεται Google Map στην οθόνη του αθλήματος ο οποίος καταγράφει την απόσταση που διένυσε. Ένα τέτοιο παράδειγμα

αθλητικής δραστηριότητας φαίνεται στο 4.4. Αυτή η δραστηριότητα στο πεδίο “parameters”, που είναι το τελευταίο πεδίο που θα εξηγήσω, έχει duration και distance σαν κελιά σε πίνακα. Ουσιαστικά, δείχνει αναφορικά ότι αυτή η δραστηριότητα μπορεί να καταγράψει την απόσταση που διανύθηκε, καθώς και τον χρόνο που ο χρήστης ασχολήθηκε με αυτήν. Τα Activities που δεν έχουν το distance στον πίνακα parameters, δεν εμφανίζουν χάρτη (που σημαίνει ότι δεν έχουν το showMap πεδίο) και δεν καταγράφουν απόσταση, παρά μόνο χρόνο άσκησης.



Σχήμα 4.4. Εγγραφή μιας αθλητικής δραστηριότητας και συγκεκριμένα το περπάτημα.

Τελειώνοντας με το DaySummary, ανέφερα ότι έχει και τα αθλήματα που έχει εμπλακεί ο χρήστης στην διάρκεια μιας ημέρας. Τα αθλήματα αυτά σαν οντότητα είναι το ActivityDone, η οποία αναφέρεται σε ένα από τα top level activities που περιέγραψα μόλις. Η σύνδεση γίνεται με το πεδίο activityId, το οποίο είναι αναφορά σε ένα από τα document id μέσα στην συλλογή activities. Επίσης αναφορές αποτελούν και οι εγγραφές της εσωτερικής συλλογής του ενός ActivityDone, που λέγεται activityStatistics. Οι εγγραφές αυτής της συλλογής μπορεί να είναι από μία έως δύο, καθώς αναφέρονται στον πίνακα parameters που έχει ένα Activity. Δηλαδή, οι δύο εγγραφές έχουν το όνομα της παραμέτρου είτε distance, είτε duration και την αντίστοιχη τιμή που καταγράφηκε, κατά της λήξη της συνεδρίας μιας αθλητικής δραστηριότητας, που ξεκίνησε από την εφαρμογή ο χρήστης. Έπειτα, έχουμε το goalParameter και το breakTime. Το τελευταίο είναι ο χρόνος που ο χρήστης έκανε διάλειμμα εν μέσω της δραστηριότητας, ενώ το goalParameter αναφέρεται στον στόχο που θέτει ο χρήστης πριν ξεκινήσει την συνεδρία κάποιου αθλήματος. Ο στόχος αυτός μπορεί να είναι ένας από τις επιλογές του parameters πίνακα, που διαθέτει το αντίστοιχο Activity document.

Προχωρώντας, έχουμε το ActivityChallenge, που είναι εγγραφή στο top level collection activityChallenges. Η συγκεκριμένη εγγραφή παίζει τον ρόλο του προκαθορισμένου στόχου από την εφαρμογή, σε κάθε δραστηριότητα που παρέχει. Συνδέεται άμεσα με το πεδίο activityId, που αναφέρεται σε ένα συγκεκριμένο άθλημα από τα activities. Κατέχει ακόμα το όνομα και την εικόνα του συγκεκριμένου Activity, για λόγους ταχύτητας. Όταν λαμβάνουμε για να εμφανίσουμε το ActivityChallenge να παίρνουμε και τις αναγκαίες πληροφορίες του Activity, χωρίς να χρειάζεται να κάνουμε ένα επιπλέον αίτημα, για να πάρουμε αυτές τις λεπτομέρειες από το activities collection.

Τέλος, περιέχει το goal που είναι δική του οντότητα ονόματι ActivityChallengeGoal, η οποία δείχνει για το αντίστοιχο challenge την σημαντική πληροφορία, που δείχνει ποιά είναι η τιμή του στόχου - “value” και ποια είναι τιμή εξέλιξης - “progress” που έχει φτάσει μέχρι τώρα ο χρήστης, σε σχέση με τον στόχο. Το πεδίο type αφορά ποια από τις επιλογές του πίνακα parameters στο Activity, δηλαδή αν είναι στόχος για την διάρκεια - duration ή την απόσταση - distance του αθλήματος.

Αφού περιέγραψα κάθε υπαρκτή οντότητα, θα ολοκληρώσω με τις πληροφορίες της οντότητας User. Για τον χρήστη λοιπόν, αποθηκεύω το email, το όνομα και την εικόνα από τον λογαριασμό Google που συνδέθηκε. Επίσης μπορεί να θέσει αγαπημένα αθλήματα, τα οποία αποθηκεύονται ως id αναφορές στο πεδίο - πίνακα “favoriteActivitiesIds”. Αποθηκεύω λεπτομέρειες σώματος, όπως τα “bodyWeight”, “bodyFatPercent”, “muscleMassPercent”. Μαζί με αυτά που ανέφερα προηγουμένως, ολοκληρώνεται ο χρήστης με τα “dailyStepsGoal” και “dailyCaloricBurnGoal”, τα οποία θέτουν τον στόχο για τα “steps” και τα “caloriesBurned” αντίστοιχα, από την ημερήσια περίληψη (DaySummary).

Εκτός από τις οντότητες, το σχήμα 4.2 απεικονίζει και τις σχέσεις εξάρτησης μεταξύ τους. Η βασική top level εγγραφή του χρήστη έχει ακριβώς έναν διατροφικό - στόχο dietGoal. Άρα η σχέση του με το Diet είναι ένα προς ένα. Ακόμα μπορεί να θέσει ως αγαπημένα του, από κανένα έως όλα τα Activity. Επομένως, η σχέση User με Activity είναι ένα προς πολλά. Επίσης ο χρήστης μπορεί να έχει από κανένα εώς πολλά ActivityChallenge, άρα μιλάμε για ένα προς πολλά εξάρτηση. Με τη σειρά του ActivityChallenge, συνδέεται ένα προς ένα με το ActivityChallengeGoal. Τελευταία εξάρτηση του User είναι στο DaySummary, από τα οποία έχει ένα για κάθε ημέρα που περνάει. Το πρώτο DaySummary δημιουργείται την ώρα της αποθήκευσής του στην βάση Firestore. Έτσι η σχέση User με DaySummary ειναι ένα προς πολλά. Η επόμενη top level οντότητα Activity εκτός από τον User συνδέεται και με το ActivityChallenge, καθώς κάθε Activity μπορεί να έχει ένα έως πολλά ActivityChallenge για τον χρήστη. Οι σχέσεις του DaySummary ολοκληρώνουν τις σχέσεις των οντοτήτων. Αυτές αυτές είναι ότι το DaySummary, μπορεί να έχει ένα προς ένα σχέση με το Diet, αφού κάθε μέρα έχει μια διατροφική αξία σε φαγητά. Μπορεί να έχει ένα προς πολλά ActivityDone, αφού κάθε μέρα ο χρήστης μπορεί να δραστηριοποιηθεί με πολλά αθλήματα. Κάθε ένα ActivityDone με τη σειρά του μπορεί να έχει, όπως είπα παραπάνω τον λόγο, από ένα έως δύο ActivityStatistic.

4.2 ΑΝΑΠΤΥΞΗ FRONT-END

Σε αυτό το κεφάλαιο θα αναλύσω τα διαφορετικά οικοδομικά υλικά της εφαρμογής. Θα περιγράψω τον τρόπο που αποφάσισα να τα συνδέσω και να τα οργανώσω, για να μεγιστοποιήσω το readability, τον reusable κώδικα σε πολλά σημεία, την συντηρησιμότητα και την επεκτασιμότητα της.

4.2.1 Πρότυπο Σχεδίασης

Πρώτα θα εξηγήσω γιατί επέλεξα το πρότυπο MVVM για την σχεδίαση κάθε οθόνης της εφαρμογής. Το μοντέλο αυτό, όπως ανέφερα και στο 3 κεφάλαιο είναι το πιο κατάλληλο για μεγάλης κλίμακας εφαρμογές καθώς χωρίζει το κομμάτια κώδικα της εφαρμογής περισσότερο από τα υπόλοιπα διαθέσιμα και έτσι επιτυγχάνει την αρχή SRP (Single Responsibility Principle). Κάθε κομμάτι απομονώνεται από τα υπόλοιπα και είναι αρμόδιο για να εκπληρώνει μόνο έναν συγκεκριμένο σκοπό εντός της εφαρμογής [1]. Θα φτάσω όμως εκεί μόλις μιλήσω για τις οθόνες.

4.2.2 Android Studio Gradle

Αρχικά θα δείξω τι χρειάζεται για να γίνει το deploy της εφαρμογής. Μιας η mobile εφαρμογή που ανέπτυξα είναι Android, το μοναδικό εργαλείο στην αγορά που είναι υπεύθυνο για το deploy είναι το Android Studio. Για την δημιουργία λοιπόν ενός καινούργιου έργου ακολούθησα το wizard αυτού του λογισμικού. Με κάποια εύκολα βήματα [27] δημιουργεί τον απαραίτητο πηγαίο κώδικα για να μπορεί να εκτελεστεί ένα άδειο, χωρίς καμία λειτουργικότητα έργο, με μία μόνο οθόνη. Ωστόσο αυτό το έκανα για να γλιτώσω χρόνο διότι γιατί θα μου έπαιρνε πάρα πολύ ώρα να δημιουργεί τα gradle αρχεία και το σύστημα φακέλων που είναι απαραίτητο για να διαχειρίζεται ο compiler του android studio τα μέρη του έργου. Στην συνέχεια μετέτραψα τα gradle αρχείο τα οποία είναι γραμμένα με την γλώσσα Groovy, στην γλώσσα Kotlin για να μπορώ να τα διατηρώ πιο εύκολα μιας και η Kotlin είναι η γλώσσα που καταλαβαίνω καλύτερα. Τα αρχεία Gradle που φαίνονται στα σχήματα 4.5, 4.6 και 4.7 είναι υπεύθυνα για τις ρυθμίσεις που ορίζουν το εκτελέσιμο του προγράμματος, όπως σε ποιες εκδόσεις του Android (targetSdk) θα μπορεί να εκτελεστεί, (βλ. Σχήμα 4.6) ποιες είναι οι εκδόσεις γλωσσών προγραμματισμού (kotlinVersion) που χρησιμοποιούνται για την ανάπτυξη (βλ. Σχήμα 4.5), το αναγνωριστικό της εφαρμογής (applicationId) στην αγορά (βλ Σχήμα 4.6) και τις βιβλιοθήκες που χρησιμοποιήθηκαν για την ανάπτυξη (βλ. Σχήμα 4.7).

```
buildscript { this: ScriptHandlerScope
    val kotlinVersion by rootProject.extra { "1.8.21" }
    val hiltVersion by rootProject.extra { "2.46.1" }
    val navVersion by rootProject.extra { "2.6.0" }

    repositories { this: RepositoryHandler
        google()
        mavenCentral()
    }

    dependencies { this: DependencyHandlerScope
        classpath("com.android.tools.build:gradle:8.0.2")
        classpath("org.jetbrains.kotlin:kotlin-gradle-plugin:${kotlinVersion}")
        classpath("com.google.dagger:hilt-android-gradle-plugin:${hiltVersion}")
        classpath("androidx.navigation:navigation-safe-args-gradle-plugin:${navVersion}")
        classpath("com.google.gms:google-services:4.3.15")
        classpath("com.google.firebase:firebase-crashlytics-gradle:2.9.6")
    }
}

allprojects { this: Project
    repositories { this: RepositoryHandler
        google()
        mavenCentral()
    }
}

task<Delete>( name: "clean") { this: Delete
    delete(rootProject.buildDir)
}
```

Σχήμα 4.5. Gradle αρχείο Project επιπέδου

Το επόμενο απαραίτητο στοιχείο μιας οποιασδήποτε Android εφαρμογής είναι το AndroidManifest αρχείο (βλ 4.8, 4.9). Σε αυτό ορίζουμε το όνομα και το εικονίδιο της εφαρμογής, το βασικό Activity της εφαρμογής, το οποίο θα εξετάσουμε παρακάτω, τα απαραίτητα Services για τις εργασίας παρασκηνίου και κάποια metadata που χρειάζονται για να λειτουργήσουν βοηθητικές υπηρεσίες τα Google Maps ή το Firebase. Επίσης σημειώνουμε και τις άδειες που πρέπει να ζητήσει η εφαρμογή από το λειτουργικό του Android της συσκευής που εκτελείται για να έχει πρόσβαση σε κάποιον πόρο

του κινητού. Ένα κύριο παράδειγμα είναι η πρόσβαση στο Internet που αποκτάται με την άδεια <uses-permission android:name="android.permission.INTERNET" />, ή την πρόσβαση στο GPS για τον χάρτη με τις άδειες <uses-permission android:name="android.permission.ACCESS_FINE_LOCATION"/> και <uses-permission android:name="android.permission.ACCESS_COARSE_LOCATION"/>.

```
namespace = "gr.dipae.thesisfitnessapp"
compileSdk = 33

defaultConfig { this: ApplicationDefaultConfig
    applicationId = "gr.dipae.thesisfitnessapp"
    minSdk = 26
    targetSdk = 33
    versionCode = 1
    versionName = "1.0"

    testInstrumentationRunner = "androidx.test.runner.AndroidJUnitRunner"
    vectorDrawables { this: VectorDrawables
        useSupportLibrary = true
    }
    signingConfig = signingConfigs.getByName( name: "debug")
}

buildFeatures { this: ApplicationBuildFeatures
    compose = true
    viewBinding = true
    buildConfig = true
}

lint { this: Lint
    abortOnError = false
}
```

Σχήμα 4.6. Gradle αρχείο επιπέδου app. Ρυθμίσεις για τις Android εκδόσεις λογισμικού που υποστηρίζει η εφαρμογή.

```
dependencies { this: DependencyHandlerScope
    //CORE LIBS
    implementation("androidx.core:core-ktx:1.10.1")
    implementation("org.jetbrains.kotlin:kotlin-bom:1.8.0")
    implementation("androidx.lifecycle:lifecycle-runtime-ktx:2.6.1")
    implementation("androidx.fragment:fragment-ktx:1.6.0")

    val lifecycleVersion by extra { "2.6.1" }
    implementation("androidx.lifecycle:lifecycle-livedata-ktx:$lifecycleVersion")
    implementation("androidx.lifecycle:lifecycle-viewmodel-ktx:$lifecycleVersion")
    implementation("androidx.lifecycle:lifecycle-viewmodel-savedstate:$lifecycleVersion")
    implementation("androidx.lifecycle:lifecycle-runtime-compose:$lifecycleVersion")

    implementation("androidx.navigation:navigation-fragment-ktx:${rootProject.extra.get("navVersion")}")
    implementation("androidx.navigation:navigation-ui-ktx:${rootProject.extra.get("navVersion")}")
    implementation("androidx.navigation:navigation-compose:${rootProject.extra.get("navVersion")}")

    //COMPOSE LIBS
    implementation("androidx.compose:compose-bom:2023.06.01")
    implementation("androidx.activity:activity-compose:1.7.2")
    implementation("androidx.lifecycle:lifecycle-viewmodel-compose:2.6.1")
    implementation("androidx.compose.runtime:runtime-livedata:1.4.3")
    implementation("androidx.compose.ui:ui")
    implementation("androidx.compose.ui:ui-graphics")
    implementation("androidx.compose.ui:ui-tooling-preview")
    implementation("androidx.compose.material3:material3")

    val accompanistVersion by extra { "0.30.1" }
    implementation("com.google.accompanist:accompanist-pager-indicators:$accompanistVersion")
    implementation("com.google.accompanist:accompanist-swiperefresh:$accompanistVersion")
    implementation("com.google.accompanist:accompanist-flowlayout:$accompanistVersion")
    implementation("com.google.accompanist:accompanist-permissions:$accompanistVersion")
    implementation("com.google.accompanist:accompanist-insets:$accompanistVersion")
```

Σχήμα 4.7. Gradle αρχείο app επιπέδου. Εισαγωγή first party βιβλιοθηκών απαραίτητων για την ανάπτυξη του κώδικα της εφαρμογής.

4.2.3 AndroidManifest

Αφού εξετάσαμε και το AndroidManifest που ολοκληρώνει τις ρυθμίσεις της εφαρμογής, ας περάσουμε στην λειτουργικότητα της εφαρμογής και την πρώτη κλάση που χρειάζεται υλοποίηση, το Application class. Όπως δείχνω στο σχήμα 4.10, η κλάση αυτή υλοποιείται και ορίζεται από το AndroidManifest να ξεκινήσει την εφαρμογή. Στην OnCreate μέθοδο αυτής της κλάσεις βάζουμε βασικά πράγματα που χρειάζεται όλη η εφαρμογή όπως στην περίπτωση μου την αλλαγή γλώσσας με την setupLocale() μέθοδο. Η μέθοδος αυτή χρησιμοποιεί τα SharedPreferences που είναι ένα αρχείο μόνιμης αποθήκευσης στον εσωτερικό χώρο της εφαρμογής (σκληρός δίσκος) για να κρατάει και να μεταβάλει την υπάρχουσα γλώσσα της εφαρμογής.

```
<application
    android:name=".ThesisFitnessApplication"
    android:allowBackup="true"
    android:dataExtractionRules="@xml/data_extraction_rules"
    android:fullBackupContent="@xml/backup_rules"
    android:icon="@mipmap/ic_launcher"
    android:label="Thesis Fitness DEV"
    android:resizeableActivity="false"
    android:roundIcon="@mipmap/ic_launcher_round"
    android:supportsRtl="true"
    android:theme="@style/Theme.ThesisFitnessApp"
    tools:targetApi="31">
    <activity
        android:name=".ui.app.AppActivity"
        android:configChanges="orientation|keyboardHidden|screenSize|screenLayout|uiMode"
        android:exported="true"
        android:screenOrientation="portrait"
        android:theme="@style/Theme.ThesisFitnessApp"
        android:windowSoftInputMode="adjustResize">
        <intent-filter>
            <action android:name="android.intent.action.MAIN" />

            <category android:name="android.intent.category.LAUNCHER" />
        </intent-filter>
    </activity>

    <service
        android:name=".framework.service.SportSessionService"
        android:enabled="true"
        android:exported="false" />

    <meta-data
        android:name="com.google.android.gms.version"
        android:value="12451000" />
    <meta-data
        android:name="com.google.android.maps.v2.API_KEY"
        android:value="@string/google_api_key" />
</application>
```

Σχήμα 4.8. AndroidManifest, application δομή.

```
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools">

    <uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />
    <uses-permission android:name="android.permission.INTERNET" />
    <uses-permission android:name="android.permission.READ_EXTERNAL_STORAGE" />

    <!-- Maps -->
    <uses-permission android:name="android.permission.ACCESS_FINE_LOCATION" />
    <uses-permission android:name="android.permission.ACCESS_COARSE_LOCATION" />

    <!-- Steps Tracking -->
    <uses-permission android:name="android.permission.ACTIVITY_RECOGNITION" />

    <queries>
        <package android:name="com.google.android.apps.maps" />
    </queries>

    <!-- Notifications -->
    <uses-permission android:name="android.permission.WAKE_LOCK" />
    <uses-permission android:name="android.permission.VIBRATE" />
    <uses-permission android:name="android.permission.POST_NOTIFICATIONS" />
```

Σχήμα 4.9. AndroidManifest. Permissions για την λειτουργία των χαρακτηριστικών που χρησιμοποιούν το λειτουργικό του Android.

4.2.4 Application, Activity και ViewModel Class

Μετά την υλοποίηση του Application Class έρχεται η ανάγκη υλοποίησης του Activity class που ορίζει πάλι το AndroidManifest ως το αρμόδιο τμήμα της εφαρμογής που θα εκτελέσει την διεργασία της εμφάνισης της διεπαφής χρήστη.

Χρησιμοποίησα το Design Pattern του Single Activity που σημαίνει ότι η οθόνες μου όλες διαχειρίζονται από ένα μόνο activity το AppActivity που φαίνεται στο σχήμα 4.11. Σε αυτήν την κλάση συνδέω το πρώτο ViewModel της εφαρμογής που είναι υπεύθυνο για την διαχείριση και την παροχή γενικών δεδομένων που μπορεί να χρειαστεί οποιαδήποτε οθόνη της εφαρμογής. Το AppViewModel όπως το ονόμασα φαίνεται στο σχήμα 4.12. και στον κατασκευαστή του με τον όρο constructor() αρχικοποιεί απαραίτητα πεδία κλάσεις που κάνουν διάφορες άλλες χρήσιμες λειτουργίες. Για παράδειγμα η NetworkLiveData κλάσση είναι υπεύθυνη για να ενημερώνει τον χρήστη όταν η πρόσβαση στο Internet κόβεται ή επανέρχεται. Το SignInUseCase είναι υπεύθυνο για την σύνδεση της υπηρεσίας Google Authentication με το κουμπί στην Login οθόνη που συνδέει τον χρήστη στην εφαρμογή μέσω google λογαριασμού.

Ο τρόπος επικοινωνίας ενός ViewModel, που είναι ένα κομμάτι επεξεργασίας δεδομένων και παροχής των στο Activity και τις οθόνες του, γίνεται μέσα των LiveData και των States. Το LiveData `val restartApp: LiveData<Unit> = _restartApp` φαίνεται στο σχήμα 4.12 και είναι μια μεταβλητή που ορίζεται στο ViewModel και το Activity παρακολουθεί τις αλλαγές στην τιμή αυτής μεταβλητής με την εντολή που φαίνεται στο σχήμα 4.11 `restartApp.observe(this@AppActivity,Observer(this@AppActivity::restartApp))` που σημαίνει ότι κάθε φορά που αλλάζει η τιμή του restartApp θα καλείται η μέθοδος του Activity restartApp με παράμετρο η αλλαγμένη τιμή της μεταβλητής. Η μέθοδος αυτή με την σειρά της θα επανεκκινεί την εφαρμογή με την αλληλουχία εντολών

```
fun Activity.restartApp() {
  finishAffinity()
  val intent = Intent(this, AppActivity::class.java)
  intent.flags = Intent.FLAG_ACTIVITY_CLEAR_TASK
  startActivity(intent)
}
```

```kotlin
@HiltAndroidApp
class ThesisFitnessApplication : Application() {

    @Inject
    @GeneralSharedPrefs
    lateinit var prefs: Lazy<SharedPreferences>

    ptoumpas +1
    override fun onCreate() {
        super.onCreate()

        // Debug Only
        if (BuildConfig.DEBUG) {
            Timber.plant(Timber.DebugTree())
        }

        setupLocale(prefs.get())
    }
}

ptoumpas
private fun setupLocale(prefs: SharedPreferences) {

    if (prefs[PREFS_LANGUAGE, ""] == "") {
        val language = Locale.getDefault().language
        prefs[PREFS_LANGUAGE] = when (language) {
            LanguageResult.GREEK.value -> LanguageResult.GREEK.value
            else -> LanguageResult.ENGLISH.value
        }
    }
}
```

Σχήμα 4.10. Thesis Fitness App Application Class.

```kotlin
@ExperimentalMaterialApi
@ExperimentalPermissionsApi
@DelicateCoroutinesApi
@ExperimentalMaterial3Api
@ExperimentalFoundationApi
@ExperimentalComposeUiApi
@AndroidEntryPoint
class AppActivity : BaseActivity() {

    private val viewModel: AppViewModel by viewModels()

    private var googleSignInIntentListener: ActivityResultLauncher<IntentSenderRequest>? = null

    PanosTob
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setupObservers()
        registerGoogleSignInIntentListener()
        setContent {
            ThesisFitnessAppTheme(statusBarColor = Color.Black) {
                ActivityProgressContainer(progressDisplayed = LoadingLiveData.observeAsState( initial: false).value) {
                    AppNavHost(viewModel)
                }
            }
        }
    }

    PanosTob
    private fun setupObservers() {
        with(viewModel) { this: AppViewModel
            restartApp.observe( owner: this@AppActivity, Observer(this@AppActivity::restartApp))
            recreateApp.observe( owner: this@AppActivity, Observer(this@AppActivity::recreateApp))
            networkLiveData.observe( owner: this@AppActivity, Observer(this@AppActivity::onNetworkConnectivityChange))
            submitLanguageChange.observe( owner: this@AppActivity, Observer(::submitLanguageChange))
            initiateGoogleSignInWindow.observe( owner: this@AppActivity, Observer(::initiateGoogleSignInWindow))
            startStopWatch.observe( owner: this@AppActivity, Observer(::startStopWatch))
            pauseStopWatch.observe( owner: this@AppActivity, Observer(::pauseStopWatch))
            stopStopWatch.observe( owner: this@AppActivity, Observer(::stopStopWatch))
        }
    }
```

Σχήμα 4.11. AppActivity. Βασικό Activity που διαχειρίζεται όλο το UI της εφαρμογής.

```
@HiltViewModel
class AppViewModel @Inject constructor(
    savedStateHandle: SavedStateHandle,
    val networkLiveData: NetworkLiveData,
    private val podHelper: PoDHelper,
    private val signInUserUseCase: SignInUserUseCase,
    private val logoutUserUseCase: LogoutUserUseCase,
    private val disableGoogleSignIfUserDenialsExceedsLimitUseCase: DisableGoogleSignIfUserDenialsExceedsLimitUseCase
) : BaseViewModel() {

    private val _restartApp = SingleLiveEvent<Unit>()
    val restartApp: LiveData<Unit> = _restartApp

    private val _recreateApp = SingleLiveEvent<Unit>()
    val recreateApp: LiveData<Unit> = _recreateApp

    private val _startStopWatch = SingleLiveEvent<Unit>()
    val startStopWatch: LiveData<Unit> = _startStopWatch

    private val _pauseStopWatch = SingleLiveEvent<Unit>()
    val pauseStopWatch: LiveData<Unit> = _pauseStopWatch

    private val _stopStopWatch = SingleLiveEvent<Unit>()
    val stopStopWatch: LiveData<Unit> = _stopStopWatch

    private val _submitLanguageChange = MutableLiveData<Unit>()
    val submitLanguageChange: LiveData<Unit> = _submitLanguageChange

    private val _initiateGoogleSignInWindow = SingleLiveEvent<IntentSender>()
    val initiateGoogleSignInWindow: LiveData<IntentSender> = _initiateGoogleSignInWindow

    private val _appUiState: MutableState<AppUiState> = mutableStateOf(AppUiState())
    val appUiState: State<AppUiState> = _appUiState
```

Σχήμα 4.12. AppViewModel κλάση. Υπεύθυνη για την διαχείριση όλης της εφαρμογής.

4.2.5 Jetpack Compose UI

Το State αντίστοιχα, είναι το ίδιο με το LiveData απλώς τις αλλαγές στην τιμή του παρακολουθούν τα Composables που είναι υπεύθυνη για την απεικόνιση των συστατικών της οθόνης. Έτσι το `val appUiState: State<AppUiState> = _appUiState` ορίζει την κατάσταση των οθονών της εφαρμογής την οποία επεξεργάζεται το NavHost composable για να ορίσει σε ποιό κομμάτι οθονών να πλοηγηθεί ο χρήστης. Το NavHost (βλ 4.13) αρχικοποιείται στην OnCreate μέθοδο του Activity (βλ 4.10) μέσα στην setContent μέθοδο και είναι ο κοντέινερ όλων των οθονών την εφαρμογής που χωρίζονται σε δύο κομμάτια, το OnboardingNavHost, το LobbyNavHost. Το OnboardingNavHost έχει τις οθόνες Splash, Login και Wizard, ενώ το LobbyNavHost έχει όλες τις εσωτερικές οθόνες μετά την αυθεντικοποίηση και την δημιουργία λογαριασμού του χρήστη.

Το NavHost είναι Composable του καινούργιου UI Toolkit Jetpack Compose αντίστοιχο του προκάτοχου NavGraph των Android Views Toolkit [28].

Τα Composables είναι το βασικό συστατικό στοιχείο με το οποίο έχτισα την δομή της διεπαφής χρήστη. Ένα βασικό παράδειγμα είναι αυτό της Home οθόνης στο σχήμα 4.14, που δημιουργεί την οθόνη με εσωτερικές κλήσεις σε άλλα Composables που με την σειρά τους χτίζουν μέρη της οθόνης όπως το `HomeUserDetails(uiState.userDetails)`. Αυτά τα μέρη διατάσσονται στον

χώρο της οθόνης με το βασικό composable διάταξης Column, που περιέχει όλα τα υπόλοιπα composables και καλείται πρώτο πρώτο μετά το άνοιγμα του block της μεθόδου HomeContent(). Η Modifier παράμετρος είναι υπεύθυνη για να εμπλουτίσει το Column composable με επιπλέον λειτουργικότητας και τροποποιήσεις στον τρόπο εμφάνισης του όπως το ότι θα πιάνει όλο το μέρος της οθόνης `Modifier.fillMaxSize()` ότι θα έχει την δυνατότητα να σκρολλαρει ο χρήστης σέρνωντας το δάχτυλο πάνω του `.verticalScroll(rememberScrollState())` και με το ότι το χρώμα του θα είναι αυτό της βασικής παλέτας χρωμάτων της εφαρμογής `.background(color = MaterialTheme.colorScheme.background)`.

Όσον αφορά την παλέτα χρωμάτων ακολουθείται το guideline του Material Design και δημιουργήθηκε μια παλέτα χρωμάτων που χρησιμοποιούν όλα τα Composables της εφαρμογής χρειάζονται χρωματισμό. Αυτή η παλέτα φαίνεται στο σχήμα 4.15. και διαθέτει δύο εναλλαγές χρωμάτων για το Light και Dark mode της εφαρμογής.

```
NavHost(
    modifier = Modifier
        .fillMaxSize()
        .background(MaterialTheme.colorScheme.background),
    navController = navController,
    startDestination = OnBoardingNavHostRoute
) { this: NavGraphBuilder
    onBoardingNavHost(
        onGoogleSignInClicked = { viewModel.onGoogleSignInClicked(it) },
        onUserAlreadySignIn = { navController.navigateToLobbyNavHost() }
    )
    lobbyNavHost(
        startStopWatch = { viewModel.startStopWatch() },
        pauseStopWatch = { viewModel.pauseStopWatch() },
        stopStopWatch = { viewModel.stopStopWatch() },
        logoutUser = { viewModel.logoutUser() }
    )
}
```

Σχήμα 4.13. Jetpack Compose NavHost της εφαρμογής.

```
@ExperimentalMaterial3Api
@Composable
fun HomeContent(
    uiState: HomeUiState
) {
    Column(
        Modifier
            .fillMaxSize()
            .verticalScroll(rememberScrollState())
            .background(color = MaterialTheme.colorScheme.background)
            .padding(horizontal = SpacingCustom_24dp)
    ) { this: ColumnScope
        HomeUserDetails(uiState.userDetails)
        VerticalSpacerDouble()
        HomeDailyMovementStats(uiState.userMovementTracking)

        VerticalSpacerDefault()
        WidthAdjustedDivider(Modifier.fillMaxWidth())
        VerticalSpacerDefault()

        HomeSportChallenges(uiState.sportChallenges)
    }
}
```

Σχήμα 4.14. Παράδειγμα Composable της Home οθόνης της εφαρμογής.

```
val lightColorsPalette = lightColorScheme(
    primary = ColorPrimary,
    secondary = ColorSecondary,
    tertiary = ColorBottomNavBar,
    background = Color.White,
    surface = Color.Black,
    outline = Color.Yellow
)

val darkColorsPalette = darkColorScheme(
    primary = ColorPrimaryDark,
    secondary = ColorSecondary,
    background = Color.Black,
    surface = Color.White,
    tertiary = Color.White,
    outline = Color.Yellow
)
```

Σχήμα 4.15. Παλέτα χρωμάτων της εφαρμογής

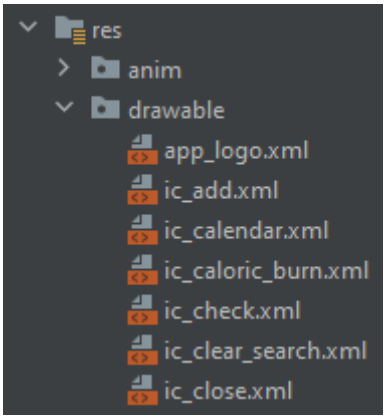
4.2.6 Android Resources

Τα κείμενα και οι εικόνες που εμφανίζονται στο UI της εφαρμογής δημιουργούνται στον φάκελο αποθήκευσης resources. Στο αρχείο strings.xml ορίζονται τα κείμενα που έχουν πρόσβαση τα Composables

```
<resources>
    <string name="app_name">ThesisFitnessApp</string>

    <string name="empty" translatable="false" />
    <!-- Login Screen -->
    <string name="login_title">Welcome</string>
    <string name="login_sign_in_btn">Sign in with Google</string>
    <string name="login_guest_btn">Or continue as Guest</string>
```

Σχήμα 4.16. Android resource Strings file

και στην τα αρχεία των εικόνων ορίζονται στο μέσα στον υποφάκελο του resources, με όνομα drawable.



Σχήμα 4.17. Android drawable files

4.2.7 Σύνδεση εφαρμογής με εξωτερικές πηγές

Αφού ανέλυσα το UI, θα ήθελα να εξηγήσω και πως λαμβάνω τα δεδομένα από το Backend της εφαρμογής που είναι το Firebase. Με βάση τα σχήματα 4.18. και 4.19 γίνονται όλα τα queries στην απομακρυσμένης πηγή δεδομένων Firestore του Firebase. Ωστόσο στην εφαρμογή υλοποίησα και την σύνδεση third party απομακρυσμένης πηγής δημοσίου REST API, από το οποίο εμφανίζω στον χρήστη την λίστα με τα φαγητά που μπορεί να επιλέξει για την διατροφή του. Για αυτήν την σύνδεση χρησιμοποίησα το Retrofit 2 που είναι η προτεινόμενη βιβλιοθήκη από την google για να την εκπόνηση αιτημάτων σε RESTFUL API. Το στήσιμο αυτής της λειτουργίας γίνεται με αρχικά με την δημιουργία των αιτημάτων που δηλώνει ο προγραμματιστής ως ένα interface με μη υλοποιημένες μεθόδους για κάθε αίτημα που θα επικοινωνεί με το API. Σχήμα 4.20 φαίνονται οι δηλώσεις των αιτημάτων που χρειάστηκα. Η πρώτη μέθοδος είναι για την εμφάνιση και το pagination της λίστας με τα φαγητά, καθώς ο χρήστης σκρολάρει. Η δεύτερη μέθοδος δείχνει το αίτημα της αναζήτησης φαγητού. Για να γραφεί η κλήση αυτών των μεθόδων χρειάστηκα τις Kotlin κορουτίνες που εξήγησα στο 3ο κεφάλαιο. Αυτές είναι αναγκαίες για την ασύγχρονη διεργασία ανάκτησης των δεδομένων που επιστρέφει το REST API. Για να τρέξει μια κορουτίνα χρειάζεται ένα CoroutineContext που ορίζει σε ποιο thread θα τρέχει αυτή, ενώ το UI thread ασχολείται με την αλληλεπίδραση του χρήστη με την διεπαφή. Εγώ όπως φαίνεται στο σχήμα 4.21 διάλεξα το IO thread που είναι αποδοτικό για την διαχείριση εισόδου και εξόδου δεδομένων από την εφαρμογή. Στο αποτέλεσμα της ασύγχρονης αυτής κλήσης “ακούει” το ViewModel που ανέφερα προηγουμένως ως διαχειριστή δεδομένων. Αυτό γίνεται στο σχήμα 4.22 με με την μέθοδο του viewModelScope (CoroutineScope), launch {}, το οποίο ξεκινάει την κορουτίνα λαμβάνοντας το CoroutineContext, που θέσαμε προηγουμένως και εμφανίζει τα δεδομένα στο UI όταν αυτά ανακτηθούν από το internet.

```
override suspend fun registerUser() {
    val firebaseUserId = getFirebaseUserId()

    fireStore.collection(USERS_COLLECTION).document(firebaseUserId)
        .set(
            hashMapOf(
                USER_EMAIL to auth.currentUser?.email,
                USER_NAME to auth.currentUser?.displayName,
                USER_IMG_URL to auth.currentUser?.photoUrl
            )
        ).await()
}
```

Σχήμα 4.18. Set query αποθήκευσης πληροφοριών χρήστη στο Firebase Firestore.

```
override suspend fun getUser(): RemoteUser? {
    val firebaseUserId = auth.currentUser?.uid ?: throw FirebaseNoSignedInUserException("")
    return fireStore.collection(USERS_COLLECTION).document(firebaseUserId).getDocumentResponse<RemoteUser>()
}
```

Σχήμα 4.19. Get query ανάκτηση πληροφοριών χρήστη από το Firebase Firestore

```
interface FoodApi {

    PanosTob
    @GET("v1/foods/list?dataType=Foundation")
    suspend fun getFoodList(
        @Query("pageSize") pageSize: Int = 25,
        @Query("pageNumber") page: Int
    ): Response<List<RemoteFood>>

    PanosTob
    @GET("/fdc/v1/foods/search?dataType=Foundation")
    suspend fun getFoodByName(
        @Query("query") foodNameQuery: String
    ): Response<RemoteSearchFoodResponse>
}
```

Σχήμα 4.20. Δηλώσεις request στο public REST API για τα τρόφιμα.

```
return withContext(Dispatchers.IO) { this: CoroutineScop
    api.getFoodList(page = page).requireNotNull()
}
```

Σχήμα 4.21. Ασύγχρονη κλήση της μεθόδου του FoodApi.

με την μέθοδο του viewModelScope (CoroutineScope), launch {}, το οποίο ξεκινάει την κορουτίνα λαμβάνοντας το CoroutineContext, που θέσαμε προηγουμένως και εμφανίζει τα δεδομένα στο UI όταν αυτά ανακτηθούν από το internet.

```
private val _uiState: MutableState<FoodSelectionUiState> = mutableStateOf(FoodSelectionUiState())
val uiState: State<FoodSelectionUiState> = _uiState

private var pageFetched = 1

 PanosTob *
fun init() {
    viewModelScope.launch { this: CoroutineScope
        val foodList = foodSelectionUiMapper(getFoodListUseCase(pageFetched))
        _uiState.value = foodList
    }.apply { this: Job
        invokeOnCompletion { it: Throwable?
            LoadingLiveData.postValue( value: false)
        }
    }
}
```

Σχήμα 4.22. Ασύγχρονη σύνδεση REST API δεδομένων με το UI μέσω του ViewModel.

4.2.8 Services

Στο σχήμα 4.8 φαίνεται και ένα ακόμα βασικό συστατικό της εφαρμογής που λέγεται SportSessionService. Κληρονομεί την κλάση Service, η οποία είναι η προτεινόμενη κλάση που έχει επιλέξει η Google για την διαχείριση διεργασιών παρασκηνίου [29]. Μέσα σε αυτήν την κλάση υλοποιώ τον χρόνομετρο της συνεδρίας αθλητικής δραστηριότητας, και την συνεχή ενημέρωση της τοποθεσίας του χρήστη έτσι ώστε να καταγράφεται η διάρκεια και η απόσταση του αθλήματος ακόμα και αν ο χρήστης έχει κλείσει την οθόνη του κινητού ή έχει βάλει την εφαρμογή στο παρασκήνιο. Αυτή η διαδικασία φαίνεται στο σχήμα 4.23, όπου γίνεται η αρχικοποίηση του Timer που κρατάει τον χρόνο, του NotificationManager που εμφανίζει την ειδοποίηση του χρονομέτρου (βλ. επόμενη ενότητα) και του LocationProvider που κρατάει την τοποθεσία του χρήστη με βάση το GPS. Έτσι όταν υπάρχουν αλλαγές σε αυτά τα δεδομένα, το ίδιο το service ενημερώνει τα δύο αντικείμενα κλάσεων StopWatchBroadcast και SportSessionLocationProvider, στα οποία ακούν ταυτόχρονα οι οθόνη του UI που δείχνει το χρονόμετρο και τον χάρτη του αθλήματος σε εξέλιξη, μέσω του ViewModel της οθόνης. Οι ενημερώσεις φαίνονται στο σχήμα 4.24. Για να έχουν οι δύο κλάσεις SportSessionService και SportSessionViewModel το ίδιο instance ενός αντικειμένου χρησιμοποιούν τον Singleton provider του Hilt Module SportSessionModule που απεικονίζει το σχήμα 4.25. Με το object SportSessionModule που είναι αντικείμενο του SingletonComponent, μιας κλάσης που δημιουργεί το Hilt under-the-hood στην onCreate μέθοδο του Application class, οι δύο αυτές κλάσεις λαμβάνουν τα αντικείμενα SportSessionLocationProvider και StopWatchBroadcast μέσω των αντίστοιχων provider μεθόδων που είναι υλοποιημένα μέσα.

```kotlin
@DelicateCoroutinesApi
@AndroidEntryPoint
class SportSessionService : Service() {

    @Inject
    lateinit var stopWatchBroadcast: StopWatchBroadcast

    @Inject
    lateinit var notification: NotificationCompat.Builder

    @Inject
    lateinit var sportSessionLocationProvider: SportSessionLocationProvider

    private lateinit var notificationManager: NotificationManager

    private var duration: Duration = ZERO
    private lateinit var timer: Timer

    PanosTob
    override fun onBind(intent: Intent?): IBinder? {
        return null
    }

    PanosTob
    override fun onCreate() {
        super.onCreate()
        notificationManager = getSystemService(Context.NOTIFICATION_SERVICE) as NotificationManager
    }
```

Σχήμα 4.23. Υλοποίηση SportSessionService.

```kotlin
private fun updateSportSessionStopWatch() {
    stopWatchBroadcast.refreshStopWatchMillisPassed(duration.inWholeMilliseconds)
}
```

```kotlin
when (intent.action) {
    STOP_WATCH_ACTION_START -> {
        sportSessionLocationProvider.startUserLocationUpdates( locationUpdateIntervalMillis: 5000)
        startTimer()
    }

    STOP_WATCH_ACTION_PAUSE -> {
        sportSessionLocationProvider.stopTracking()
        pauseTimer()
    }

    STOP_WATCH_ACTION_STOP -> {
        clearService()
    }
}
```

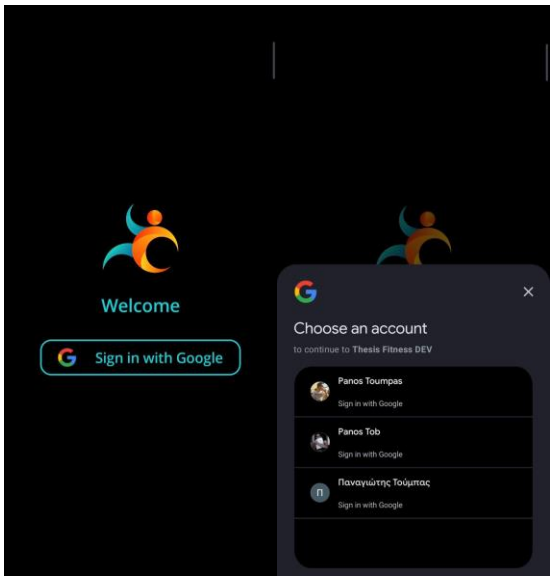
Σχήμα 4.24. Ενημέρωση

```kotlin
@DelicateCoroutinesApi
@Module
@InstallIn(SingletonComponent::class)
object SportSessionModule {
    ptoumpas
    @Provides
    fun provideSportSessionBroadcast(): SportSessionBroadcast {
        return SportSessionBroadcast.getInstance()
    }

    PanosTob +1
    @Provides
    fun provideStopWatchBroadcast(): StopWatchBroadcast {
        return StopWatchBroadcast.getInstance()
    }

    PanosTob
    @Provides
    fun provideSportLocationProvider(@ApplicationContext context: Context): SportSessionLocationProvider {
        return SportSessionLocationProvider.getInstance(context)
    }
```

Σχήμα 4.25. SportSessionModule. Singleton αντικείμενο που παρέχει ίδιο instance αντικειμένων σε όποιον τα ζητάει.

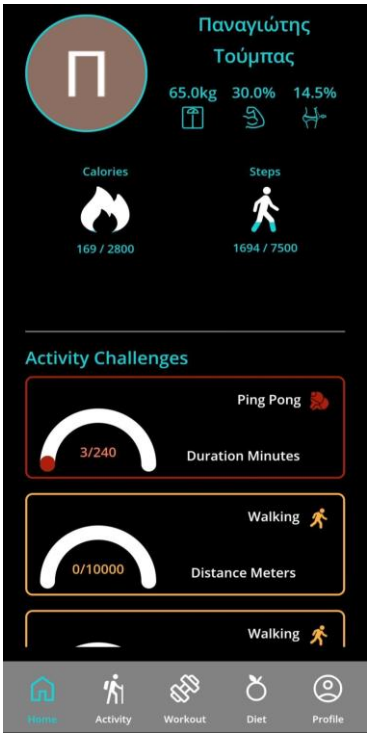
4.3 ΟΘΟΝΕΣ ΔΙΕΠΑΦΗΣ ΧΡΗΣΤΩΝ ΚΑΙ ΠΕΡΙΓΡΑΦΗ ΠΕΡΙΠΤΩΣΕΩΝ ΧΡΗΣΗΣ

Κάθε φορά που ο χρήστης ανοίγει την εφαρμογή, αν είναι ήδη συνδεδεμένος στην υπηρεσία του Firebase Authentication που διαχειρίζομαι, εισέρχεται στην Home οθόνη (βλ. σχήμα 4.27). Ειδάλλως έρχεται αντιμέτωπος με την Login οθόνη (βλ. σχήμα 4.26) με μόνη επιλογή το να συνδεθεί με έναν από τους λογαριασμούς Google που διαθέτει.



Σχήμα 4.26. Οθόνη Login και παράθυρο επιλογής google λογαριασμού.

Μόλις επιλέξει με ποιον λογαριασμό θέλει να συνδεθεί, αν ο λογαριασμός υπάρχει (έχει κάνει εγγραφή στο παρελθόν) μπαίνει απευθείας στην Home οθόνη του σχήματος 4.27. Ωστόσο αν είναι η πρώτη φορά που συνδέεται με αυτόν τον λογαριασμό, θεωρείται σαν εγγραφή λογαριασμού στην πλατφόρμα της εφαρμογής και μεταβαίνει στην οθόνη του σχήματος 4.28. Εκεί ακολουθεί κάποια βήματα για να συμπληρώσει κάποιες απαραίτητες πληροφορίες για το προφίλ του. Η αλληλουχία οθονών αυτών ονομάζεται Wizard και κάθε βούλα που εμφανίζεται στο κάτω μέρος αντιστοιχεί σε

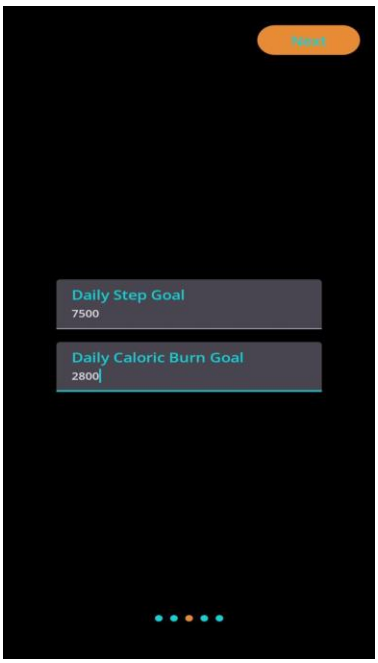


Σχήμα 4.27 Home βασική οθόνη του χρήστη στην εφαρμογή

ένα βήμα. Το πρώτο βήμα του σχήματος 4.28 του επιτρέπει να αλλάξει το όνομα του αν θέλει, για να είναι διαφορετικό από αυτό που πήρα από τον google λογαριασμό του όταν συνδέεται. Το βήμα αυτό δεν είναι προαιρετικό.

Σχήμα 4.28. Πρώτο βήμα του Wizard της εφαρμογής

Στο δεύτερο βήμα ο χρήστης καλείται να συμπληρώσει κάποιες βασικές πληροφορίες για την χτίσιμο της σωματικής του διάπλασης, με δεδομένα που ενδεχομένως έχει λάβει από τον διατροφολόγο του. Το σχήμα 4.29 απεικονίζει τα τρία “TextField” πέδια που απαρτίζουν αυτό το βήμα, από τα οποία το πρώτο, που αναφέρεται στο σωματικό βάρος, είναι υποχρεωτικό για τον υπολογισμό των θερμίδων που καίει ο χρήστης σε κάθε του διασκελισμό. Η τρίτη κουκίδα που είναι το τρίτο σε σειρά βήμα (βλ. σχήμα 4.29) αφήνει στον χρήστη προαιρετικά να ορίσει τους στόχους του ως προς το πόσα ημερήσια βήματα θέλει να κάνει και πόσες θερμίδες θέλει να κάνει καύση ημερησίως.

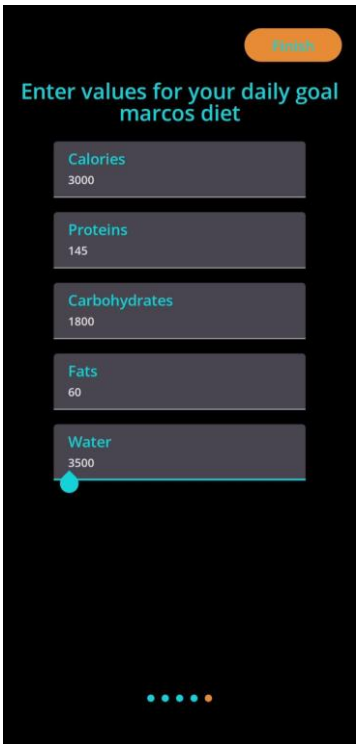


Σχήμα 4.29. Wizard προαιρετικό βήμα 3 ορισμός ημερήσιων βημάτων και θερμίδων

Βήμα 4ο είναι η επιλογή των αγαπημένων αθλημάτων. Όπως φαίνεται στο 4.30 τα πορτοκαλί αθλήματα είναι τα επιλεγμένα ως αγαπημένα και τα άσπρα ως μη επιλεγμένα. Δεν υπάρχει υποχρέωση από τον σύστημα να διαλέξει κάποιο άθλημα. Μπορεί απλά να το προσπεράσει και να μην έχει αγαπημένα αθλήματα.

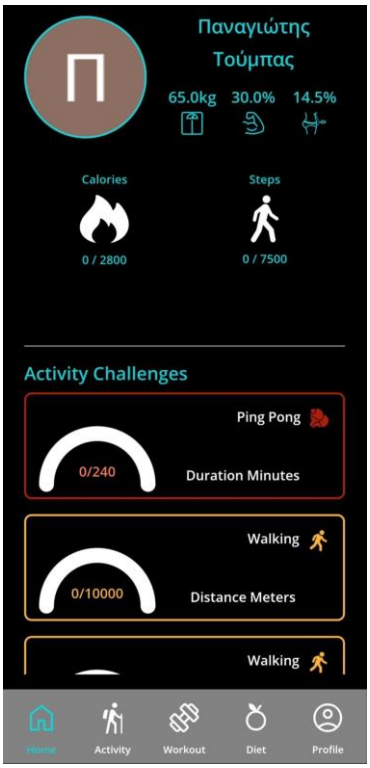
Σχήμα 4.30. Βήμα 4ο του Wizard - Επιλογή αγαπημένων αθλημάτων

Πατώντας το κουμπί next ή σέρνοντας με το δάχτυλο την οθόνη προς τα αριστερά καταλήγει στο τελευταίο βήμα που αφορά τους διατροφικούς του στόχους. Κάθε ένα πεδίο από αυτά του σχήματος 4.31 επιτρέπει στον χρήστη να βάλει σε κάποιο μακροθρεπτικό στοιχείο της ημερήσιας διατροφής του (της θερμίδες, την πρωτεΐνη, τους υδατάνθρακες, τα λίπη και το νερό) τιμή στόχου. Οι εισαγωγή τιμών δεν είναι υποχρεωτική σε κανένα από τα πεδία. Αν και να μην βάλει στόχους, η εφαρμογή θα του παρέχει κανονικά καταγραφή και εμφάνιση τον ημερήσιων θερμίδων του. Απλά δεν θα έχει με κάτι να το συγκρίνει.



Σχήμα 4.31. Τελευταίο βήμα του Wizard - Καθορισμός ημερήσιων μακροθρεπτικών στόχων.

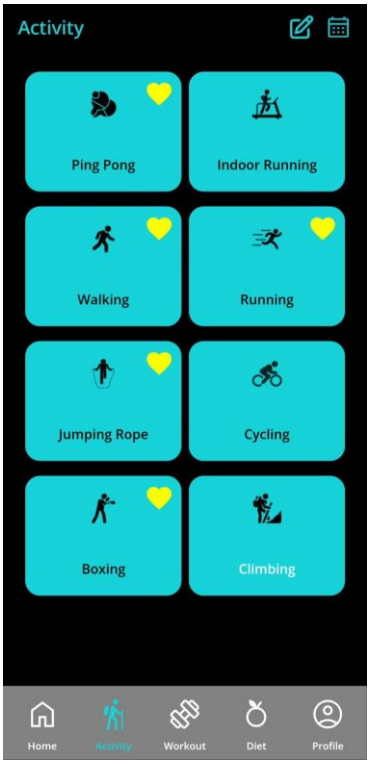
Αν το θελήσει ο χρήστης μπορεί να επιστρέψει σε κάποιο από τα προηγούμενα βήματα σέρνοντας το δάχτυλο του αριστερά για να συμπληρώσει κάποια τίμη αν το μετάνιωσε. Μόλις είναι σίγουρος για τα στοιχεία που έδωσε μπορεί να μεταφερθεί στο τελευταίο βήμα όπου το κουμπί από “Next” μεταβάλλεται σε “Finish” και πατώντας το ολοκληρώνει το Wizard και καταχωρείται η εγγραφή του στην cloud βάση δεδομένων της εφαρμογής. Όλη η προηγούμενη διαδικασία που περιέγραψα ονομάζεται OnBoarding του χρήστη που κατεβάζει την εφαρμογή για πρώτη φορά. Μπαίνοντας στην κύρια οθόνη της εφαρμογής ο χρήστης βλέπει στην βασική οθόνη (που παραθέτω στο σχήμα 4.32), την καθημερινή εξέλιξη της δραστηριότητάς του. Πάνω αριστερά φαίνεται εικόνα του προφίλ του και αριστερά το όνομα του και η σωματική του ανάλυση. Ακριβώς από κάτω εμφανίζονται οι πληρότητα των στόχων του στα βήματα που έχει κάνει και στις θερμίδες που έχει κάψει. Όσο ο αριστερός αριθμός από την κάθετο ανεβαίνει, η εικόνα χρωματίζεται με το βασικό χρώμα της εφαρμογής. Πιο κάτω μετά την διαχωριστική γραμμή είναι οι στόχοι του στα αθλήματα που έχει επιλέξει για αγαπημένα από το Wizard που ανέφερα προηγουμένως. Κάθε άθλημα αντιστοιχεί και σε ένα διαφορετικό χρώμα για να ξεχωρίζει. Όπως και στα βήματα, όσο οι στόχοι των αθλητικών δραστηριοτήτων συμπληρώνονται η άσπρη καμάρα που δείχνει την πρόοδο του στόχου γεμίζει με το χρώμα του αθλήματος.



Σχήμα 4.32. Βασική Home οθόνη της εφαρμογής.

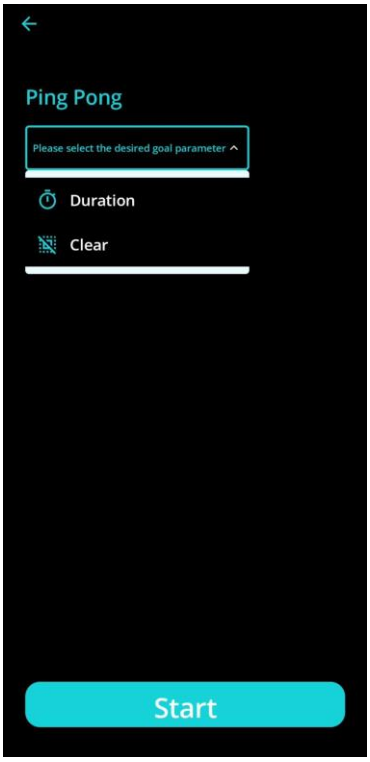
Εικόνα 4.33. Η πρόοδος των στόχων του χρήστη φαίνεται με χρώμα.

Στο κάτω μέρος της οθόνης του Home (στο σχήμα 4.33) βρίσκεται το BottomNavigation με το οποίο ο χρήστης πλοηγείται στις υπόλοιπες οθόνες της εφαρμογής. Το χρωματισμένο εικονίδια από τα πέντε του BottomNavigation δείχνει σε ποια οθόνη βρίσκεται ο χρήστης. Δίπλα από την Home βρίσκεται η Activity οθόνη και πατώντας το εικονίδιο οδηγείται στην οθόνη του σχήματος 4.34.

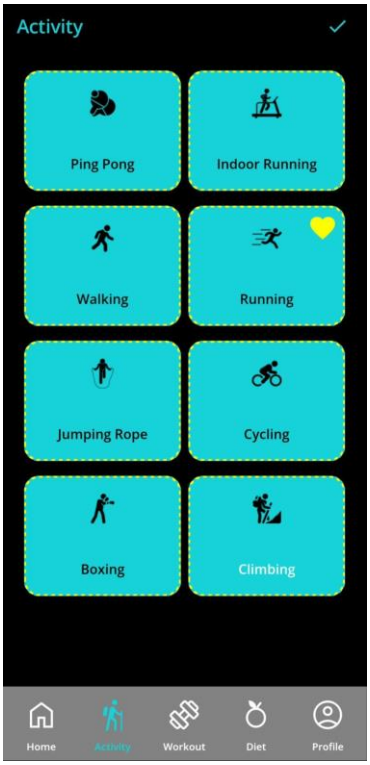


Σχήμα 4.34. Η οθόνη Activity

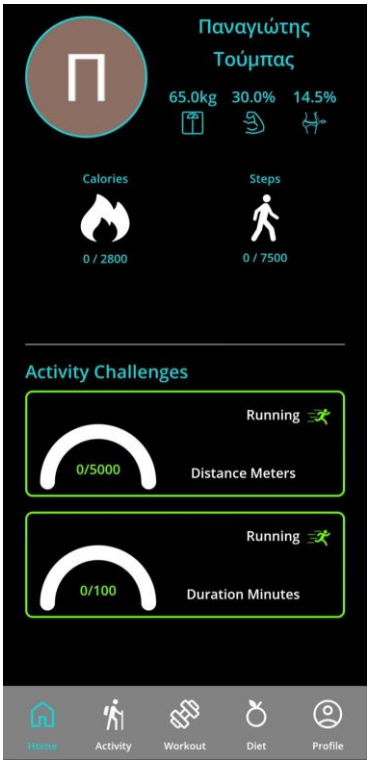
Σε αυτήν την οθόνη εμφανίζονται σε διάταξη πίνακα οι αθλητικές δραστηριότητες που υποστηρίζει η εφαρμογή. Η κίτρινη καρδούλα σε κάθε άθλημα υποδηλώνει ότι ο χρήστης το έχει προσθέσει στα αγαπημένα του. Από ο χρήστης έχει τρεις επιλογές. Η πρώτη είναι να κλικ στο τετράγωνο κάποιου αθλήματος και να οδηγηθεί στην οθόνη ActivityPreCustomization στο σχήμα 4.35. Η δεύτερη είναι να πατήσει το εικονίδιο επεξεργασίας αγαπημένων αθλημάτων που βρίσκεται πάνω δεξιά το πρώτο από τα αριστερά. Έτσι θα ενεργοποιήσει αυτήν την λειτουργία που φαίνεται στο σχήμα 4.36. Η ενεργοποίηση υποδηλώνεται από το διακεκομμένο σύνορο που εμφανίζεται γύρω από κάθε πλακάκι αθλήματος και από το κουμπί “check” που πάνω δεξιά στο toolbar που παίρνει την θέση των δύο προηγούμενων εικονιδίων. Κάθε φορά που ο χρήστης πατάει σε κάποιο κουτάκι η καρδούλα εμφανίζεται και εξαφανίζεται για να δηλώσει ότι το άθλημα επιλέχθει να είναι αγαπημένο ή οχι αγαπημένο αντίστοιχα. Όταν ο χρήστης καταλήξει στο ποια θέλει να είναι τα καινούργια αγαπημένα του μπορεί να πατήσει το “check” εικονίδιο και να οριστικοποιήσει την απόφαση του ανανεώνοντας την βάση δεδομένων. Στο σχήμα 4.36 φαίνεται ότι ο χρήστης επέλεξε μόνο το τρέξιμο να είναι αγαπημένο του. Έτσι στην Home οθόνη ανανεώνονται οι στόχοι και μένουν μόνο αυτοί που υπάρχουν για το τρέξιμο (βλ. σχήμα 4.37). Η τρίτη επιλογή του χρήστη είναι να πατήσει το εικονίδιο με το ημερολόγιο και να ανοίξει το παράθυρο ημερολογίου που δείχνω στο σχήμα 4.38. Ανοίγωντας το ημερολόγιο μπορεί να επιλέξει ένα διάστημα ημερών όσο μεγάλο και αν θέλει από το οποίο ενδιαφέρεται να μάθει το ιστορικό του για την αθλητική δραστηριότητα.



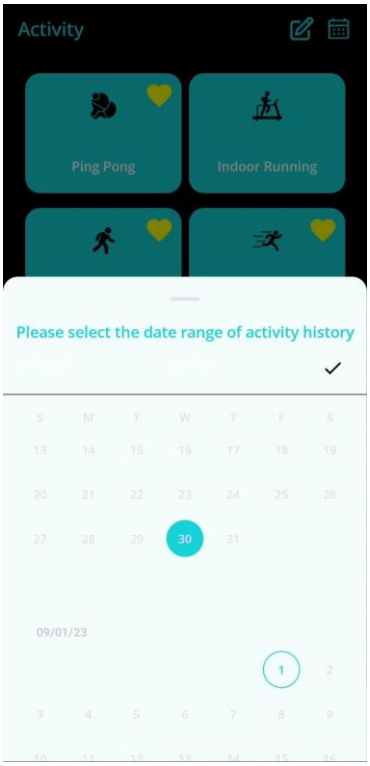
Σχήμα 4.35. Οθόνη ActivityPreCustomization



Σχήμα 4.36. Οθόνη Activity λειτουργία αλλαγής αγαπημένων αθλημάτων



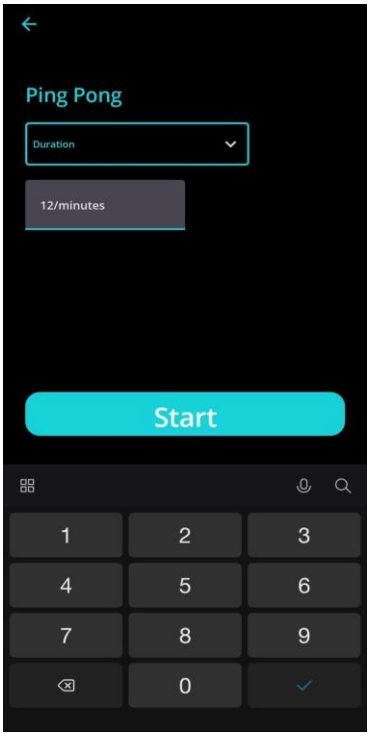
Σχήμα 4.37. Οθόνη Home μετά την αλλαγή στα αγαπημένα αθλήματα



Σχήμα 4.38. Οθόνη Activity με το ημερολόγιο του ιστορικού ανοιχτό

Στην οθόνη ActivityPreCustomization ο χρήστης μπορεί να επιλέξει έναν από τους στόχους που θέλει να θέσει πριν ξεκινήσει το συγκεκριμένο άθλημα. Για παράδειγμα στο PingPong που είναι άθλημα

χωρίς απόσταση, ο χρήστης μπορεί να επιλέξει μόνο την διάρκεια να βάλει στόχο και η τιμή του στόχου εισάγεται από τον ίδιο στο πεδίο που βρίσκεται κάτω από το dropdown με τις επιλογές (βλ. σχήμα 4.39).

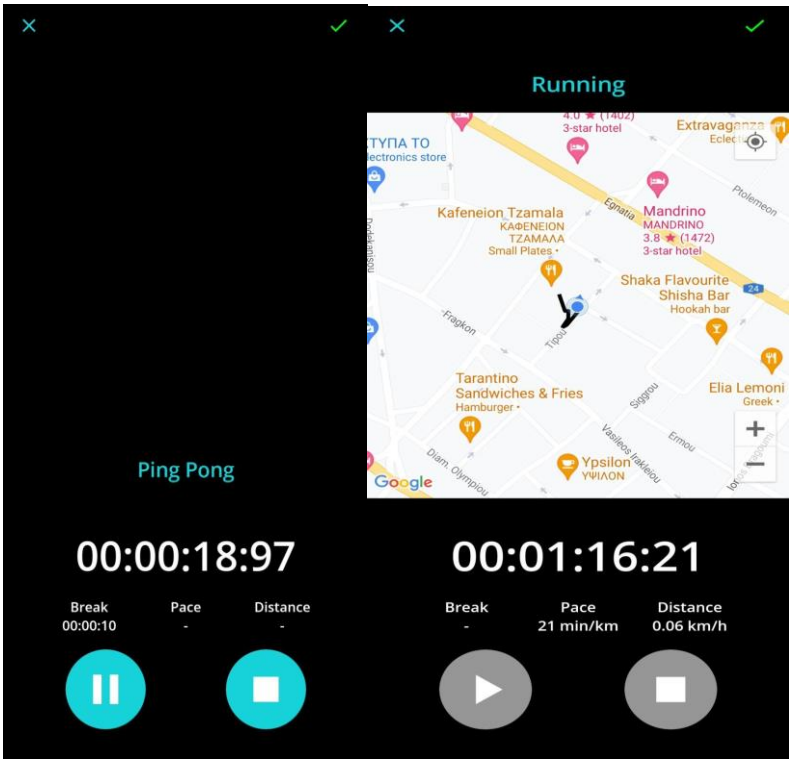


Σχήμα 4.39. ActivityPreCustomization 12 λεπτά στόχος διάρκειας πριν ξεκινήσει το PingPong

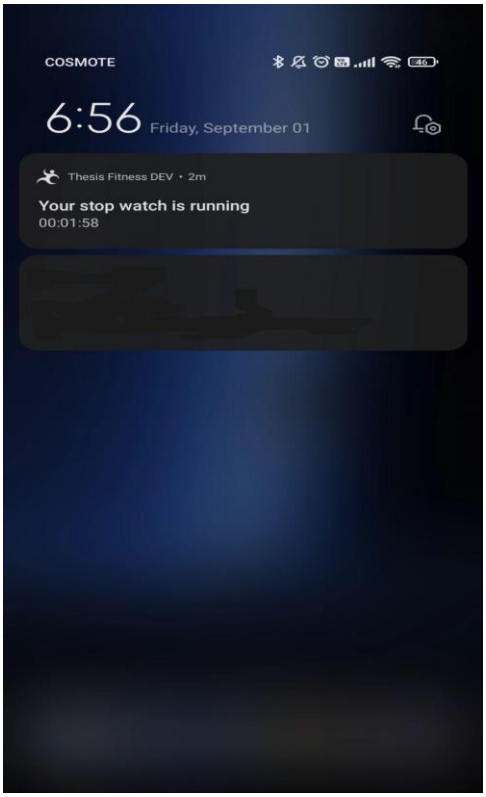
Πατώντας Start σε αυτήν την οθόνη εμφανίζεται η οθόνη ActivitySession του σχήματος 4.40.

Σχήμα 4.40. Οθόνη ActivitySession με το 3.2.1- Go animation να τελειώνει πριν ξεκινήσει το χρόνομετρο

Ξεκινώντας το ActivitySession του PingPong βλέπουμε στο σχήμα 4.41 στην αριστερή εικόνα, ότι αρχίζει να μετράει το χρονόμετρο, το οποίο βρίσκεται στο κέντρο του μήκους της οθόνης με τα πιο μεγάλα γράμματα. Από κάτω βρίσκεται το χρονόμετρο διαλείμματος που ξεκινάει όταν ο χρήστης πατήσει το κουμπί της Παύσης που βρίσκεται κάτω αριστερά. Όταν θελήσει να σταματήσει να καταγράφει την άσκηση του ο χρήστης μπορεί είτε να πατήσει το Stop κουμπί δεξιά από το Pause για να παγώσουν τα χρονόμετρα, είτε απευθείας να πατήσει το πράσινο "check" κουμπί πάνω δεξιά στο toolbar για να αποθηκεύσει την συνεδρία του. Αν οποιαδήποτε στιγμή θελήσει να ακυρώσει την συνεδρία τότε πατώντας το κουμπί X - close πάνω αριστερά στην οθόνη κλείνει η οθόνη χωρίς να αποθηκευτεί τίποτα και επιστρέφει στην οθόνη με τον πίνακα αθλητικών δραστηριοτήτων. Μια άλλη παραλλαγή αυτής της οθόνης είναι το σχήμα 4.41 στην δεξιά εικόνα. Επειδή το τρέξιμο καταγράφει και την απόστασης εμφανίζω τον χάρτη και την τοποθεσία του χρήστη καθώς κινείται. Η μαύρη γραμμή πάνω στο χάρτη δείχνει την διαδρομή που διέσχισε ο χρήστης κατά την διάρκεια της συνεδρίας. Κάτω από το βασικό χρονόμετρο και δεξιά από το Break χρονόμετρο υπολογίζεται και εμφανίζεται ο ρυθμός τρεξίματος του χρήστη καθώς και η απόσταση που διανύει σε πραγματικό χρόνο, κατά την διάρκεια της άσκησης. Οι δύο εικόνες του σχήματος αντιπροσωπεύουν τις δύο κατηγορίες αθλημάτων που ανήκουν στις δύο παραλλαγές αυτής της οθόνης. Στο σχήμα 4.42 φαίνεται η ειδοποίηση που εμφανίζω στο Notification Drawer του λειτουργικό συστήματος του κινητού έτσι ώστε ο χρήστης αν θέλει μπορεί να βάλει την εφαρμογή στο background κατά της διάρκεια της άθλησης και να ρίχνει μια ματιά στο χρονόμετρο όποτε θέλει. Αν πατήσει βέβαια την ειδοποίηση αυτή θα ανοίξει την εφαρμογή πίσω στην αθλητική συνεδρία που βρίσκεται σε εξέλιξη.

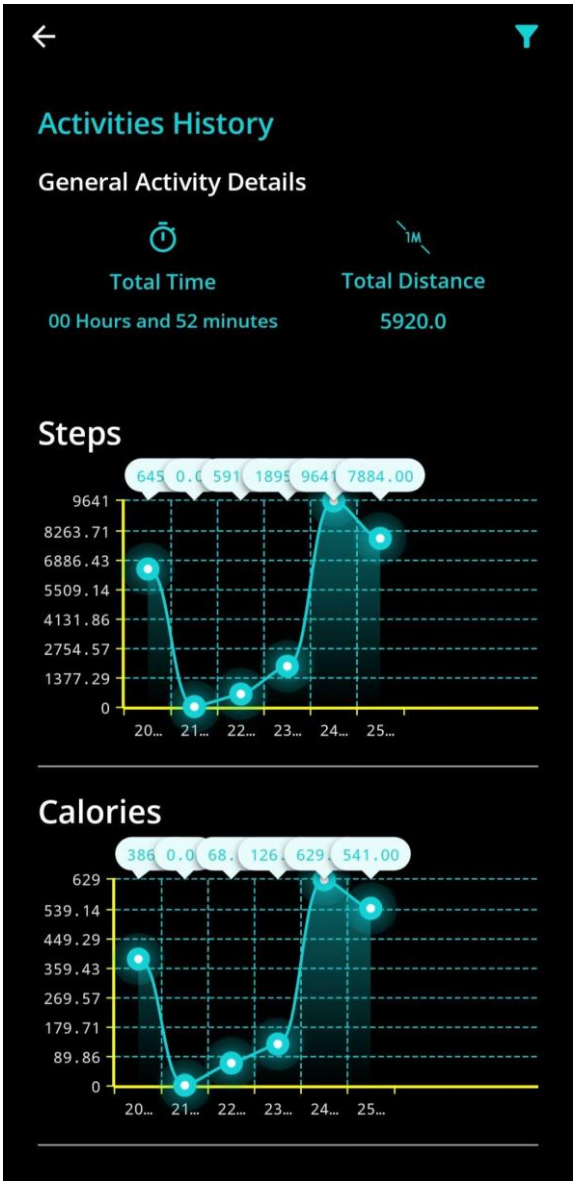


Σχήμα 4.41. Οθόνη ActivitySession σε εξέλιξη



Σχήμα 4.42. Η ειδοποίηση που δείχνει το χρονόμετρο όσο η αθλητική συνεδρία τρέχει από το Notification Drawer του Android OS

Πατώντας ωστόσο το ημερολόγιο και διαλέγοντας ένα διάστημα ημερών ο χρήστης οδηγείται στην οθόνη ActivityHistory - σχήμα 4.43. Σε αυτήν την οθόνη αρχικά στο πάνω μέρος έχω τοποθετήσει τον συνολικό χρόνο άσκησης για τις επιλεγμένες ημέρες και την συνολική απόσταση που καλύφθηκε από τα αθλήματα που περιλαμβάνουν απόσταση. Ακριβώς μετά έχω δύο διαγράμματα που δείχνουν τα βήματα και τις θερμίδες που έχει κάψει κάθε μέρα από αυτές που διάλεξε. Σκρολάροντας με το δάχτυλο πιο κάτω φαίνεται η υπόλοιπη οθόνη στο σχήμα 4.45. Εκεί πρώτα εμφανίζω ένα γράφημα πίτας που κάθε κομμάτι της απεικονίζει πόσες μέρες, από τις συνολικές που επιλέχθηκαν για το ιστορικό, έχει αθληθεί σε κάθε άθλημα και πόσο ποσοστό της πίτας καλύπτει. Αμέσως μετά δείχνω κάθε εγγραφή αθλητικής συνεδρίας που καταγράφηκε για αυτές τις μέρες ταξινομημένες από την πιο πρόσφατη σε αύξουσα σειρά. Κάθε κουτάκι δίνει πληροφορίες αριστερά για την διάρκεια και την απόσταση αν υπάρχει, ενώ δεξιά για τον στόχο που έχει θέσει ο χρήστης για την συνεδρία. Αν ο στόχος επετεύχθει εμφανίζεται το κίτρινο κυπελάκι. Πάνω αριστερά στο toolbar αυτής της οθόνης υπάρχει το κουμπί του φίλτρου, με το οποίο μπορεί ο χρήστης να επιλέξει από ένα παράθυρο να φιλτράρει κάποιο από τα αθλήματα που περιλαμβάνουν απόσταση και να αλλάξει την οθόνη στο σχήμα 4.44. Με αυτόν τον τρόπο τα γενικά στατιστικά αφορούν πλέον μόνο το άθλημα που φιλτράρει όπως και οι καταγεγραμμένες συνεδρίες κάτω από το γράφημα. Η πίτα μετατρέπεται σε γραμμικό γράφημα που δείχνει την απόσταση που σημείωσε ο χρήστης στο συγκεκριμένο άθλημα κάθε ημέρα του ιστορικού που ζητήθηκε νωρίτερα στο ημερολόγιο. Τέλος από κάτω εμφανίζονται οι καταγραφές των συνεδριών μόνο του συγκεκριμένου αθλήματος πάνω στο εφαρμόστηκε το φίλτρο.



Σχήμα 4.43. Οθόνη ιστορικού για την αθλητική δραστηριότητα.

Σχήμα 4.44. Φιλτράρισμα του ιστορικού αθλημάτων με βάση το άθλημα τρέξιμο.

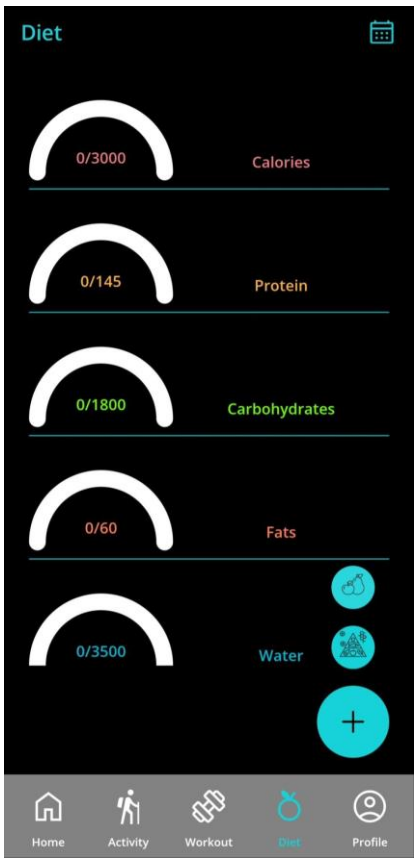
Σχήμα 4.45. Η υπόλοιπη ActivityHistory οθόνη.

Περνάω λοιπόν στην τελευταία λειτουργία της εφαρμογής που είναι η διατροφή. Είναι το δεύτερο εικονίδιο από τα δεξιά του BottomNavigation και αν πατηθεί εμφανίζεται η οθόνη στο σχήμα 4.46. Σε αυτήν την οθόνη βλέπουμε τους μακροθρεπτικούς στόχους των έναν κάτω από τον άλλον και της πρόοδο τους για σήμερα. Κάτω δεξιά υπάρχει το Floating Action Button με το + icon, που πατώντας το υπάρχουν δύο επιλογές από πάνω. Η πρώτη από κάτω επιλογή ανοίγει το παράθυρο με την χειροκίνητη ενημέρωση μακροθρεπτικών για τους πιο έμπειρους. Αυτό φαίνεται στο σχήμα 4.47. Ενώ πατώντας το ακριβώς από πάνω επιλογή οδηγούμαστε στην οθόνη με την λίστα φαγητών όπου ο χρήστης μπορεί να αναζητήσει, είτε σκρολλάρωντας, είτε μέσω της μπάρας αναζήτησης. Μόλις κλικάρει κάποιο τρόφιμο εμφανίζεται ένα παράθυρο για να εισάγει τα γραμμάρια που κατανάλωσε από αυτό. Αφού πατήσει το κουμπί “Save” υπολογίζονται και ενημερώνονται τα μακροθρεπτικά αυτόματα για το γέμισμα των Progress bars στην Diet οθόνη. Η λειτουργικότητα αυτή φαίνεται στο σχήμα 4.48.

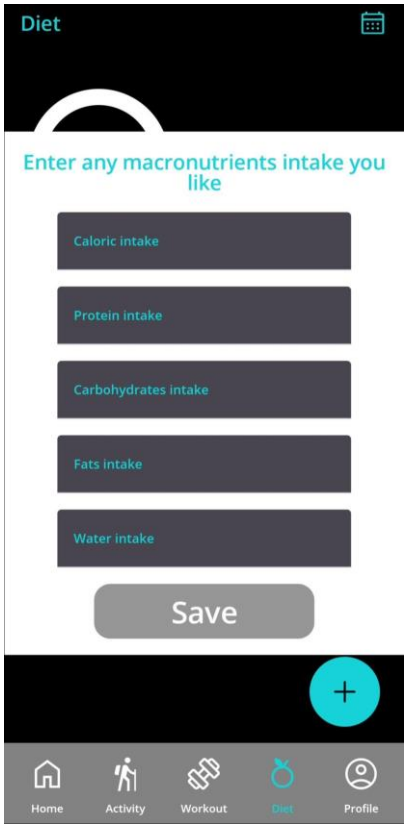
Στην οθόνη Diet ακόμα υπάρχει αντίστοιχα το ημερολόγιο πάνω δεξιά που εμφανίζεται όπως και στην Activity οθόνη, μόνο που εδώ αφορά το ιστορικό για την διατροφή. Έτσι όταν ο χρήστης

επιλέξει ημέρες, ανοίγει η οθόνη στο σχήμα 4.50. Η οθόνη αυτή εμφανίζεται ένα γραμμικό διάγραμμα για κάθε μακροθρεπτικό στοιχείο της διατροφής με x άξονα τις ημέρες που επιλέχθηκε και y άξονα τις τιμές κατανάλωσης που καταγράφηκαν για αυτές τις ημέρες. Υπάρχει μια ακόμα επιλογή για το ιστορικό διατροφής του χρήστη. Αν από το ημερολόγιο επιλέξει μόνο μία ημέρα, εμφανίζεται η ανάλυση εκείνης της ημέρας που διάλεξε. Σε αυτήν υπάρχει ένα κείμενο από πάνω που εξηγεί για ποια ημερομηνία γίνεται η επισκόπηση και ένα κουμπί πάνω αριστερά με το εικονία X - close για να επιστρέψει στην σημερινή ημερομηνία (βλ. σχήμα 4.51). Τέλος από το BottomNavigation υπάρχει το προφίλ από το οποίο ο χρήστης μπορεί να κάνει κάποιες αλλαγές που συνδέονται με τις προαναφερόμενες λειτουργίας που προσφέρει η εφαρμογή. Αυτές φαίνονται στο σχήμα 4.52 και επιτρέπουν στον χρήστη με την σειρά από πάνω μέχρι κάτω, να:

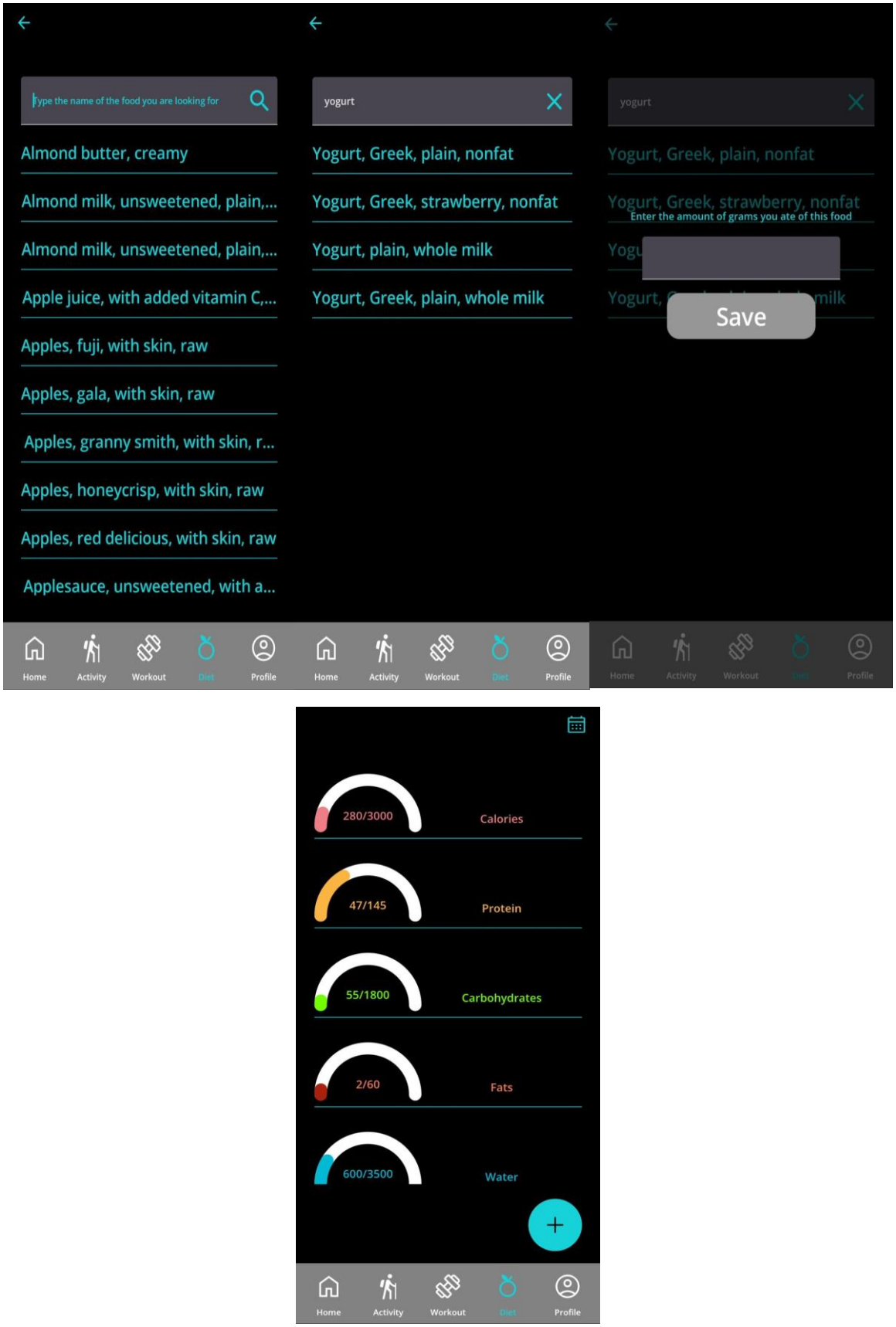
- Αλλάξη την λεπτομέρειες του σώματος του, δηλαδή το βάρος, την μυϊκή μάζα και το λίπος.
- Τους ημερήσιους στόχους του στα βήματα και τις θερμίδες.
- Τους ημερήσιους στόχους του στα μικροθρεπτικά στοιχεία
- Την αλλαγή γλώσσας από Ελληνικά/Αγγλικά
- Την αποσύνδεση του από την εφαρμογή που τον οδηγεί πίσω στην Login οθόνη
- Και με κόκκινο χρώμα την διαγραφή λογαριασμού που εμφανίζει μήνυμα προειδοποίησης και επιβεβαίωσης αν πατηθεί καταλάθος.



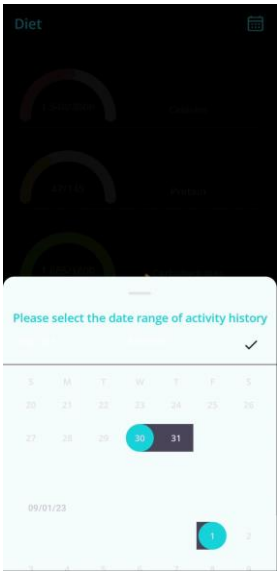
Σχήμα 4.46. Diet οθόνη



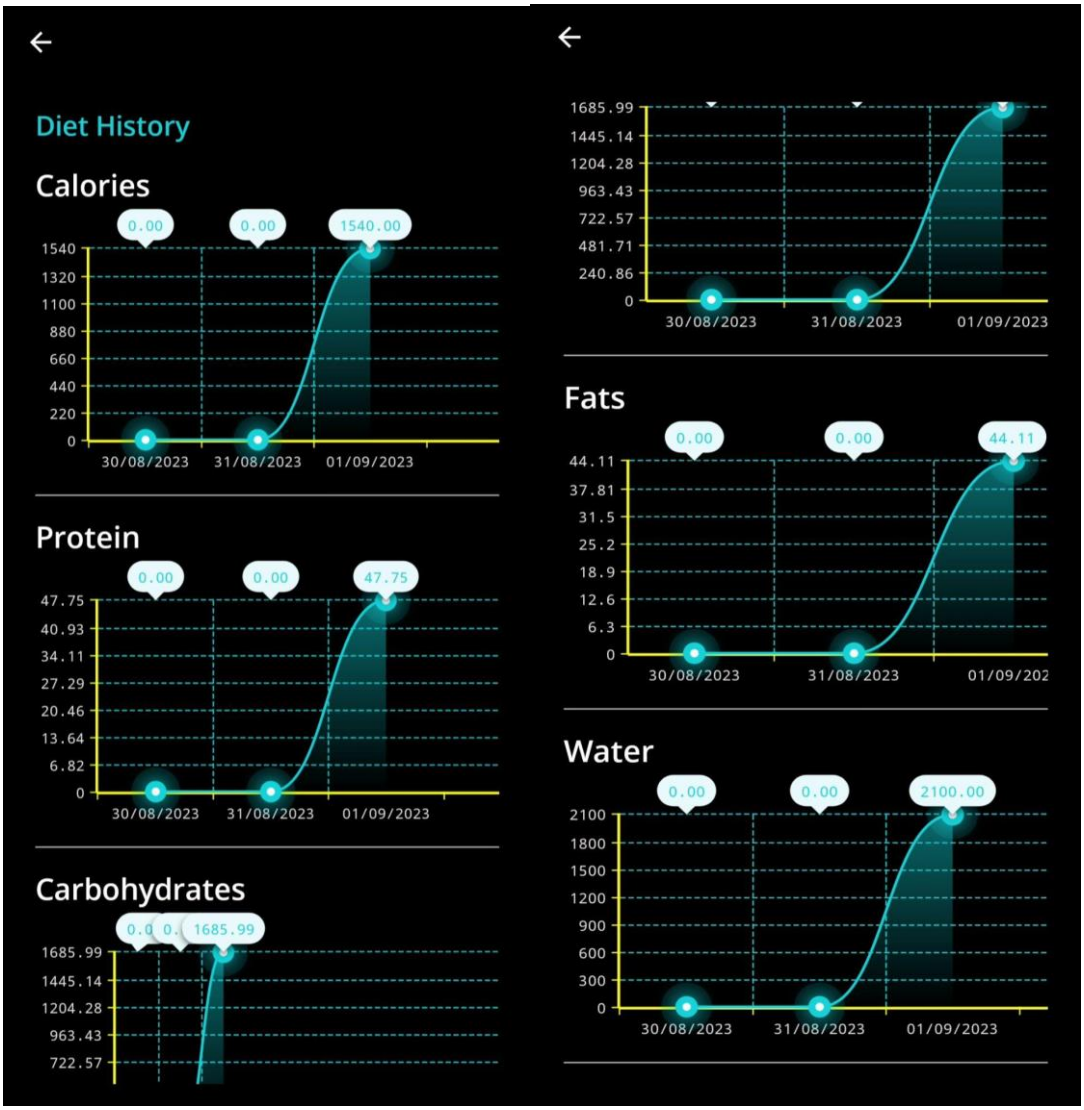
Σχήμα 4.47. Παράθυρο pop up χειροκίνητης ενημέρωσης κατανάλωσης μικροθρεπτικών στοιχείων.



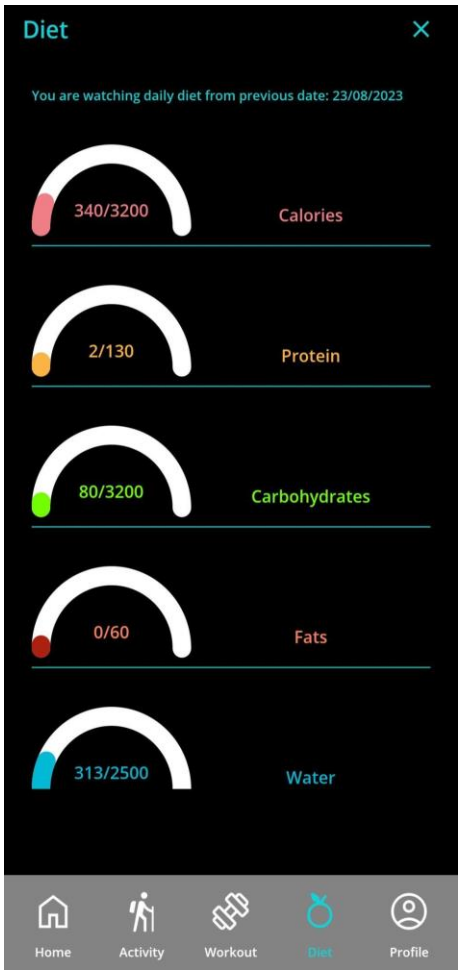
Σχήμα 4.48. Λειτουργία υπολογισμού κατανάλωσης μακροθρεπτικών από την επιλογή φαγητού.



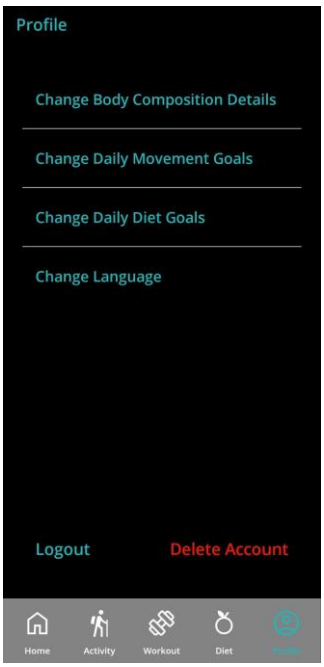
Σχήμα 4.49. Ημερολόγιο και επιλογή ημερών για εμφάνιση ιστορικού διατροφής.



Σχήμα 4.50. Οθόνη ιστορικού διατροφής.



Σχήμα 4.51. Επισκόπηση διατροφής παρελθοντικής ημέρας στην οθόνη Diet.



Σχήμα 4.52. Οθόνη Προφίλ.

Κεφάλαιο 5ο: Συμπεράσματα και Μελλοντική Εργασία

Κάνοντας μια ανασκόπηση την εκπόνησης της συγκεκριμένης Πτυχιακής Εργασίας και βλέποντας το αποτέλεσμα της εφαρμογής, πιστεύω ότι εκπλήρωσα μόνο ένα κομμάτι της ανάγκης των καταναλωτών της αγοράς, σχετικά με τις εφαρμογές fitness/diet tracking. Θα ήθελα να είχα αναπτύξει περισσότερο το κομμάτι της γυμναστικής στο σπίτι, με ή χωρίς εξοπλισμό, παρέχοντας βίντεο και φωνητικές εντολές ανατροφοδότησης του χρήστη στην εφαρμογή, που θα παίζει τον ρόλο του προσωπικού προπονητή. Ακόμα ευελπιστούσα στην πρόσθεση συστήματος τεχνητής νοημοσύνης, που αναγνωρίζει τους διατροφικούς πίνακες πάνω σε τρόφιμα της λιανικής αγοράς, μέσω σκαναρίσματος από την κάμερα του κινητού. Έτσι, η διαδικασία αποθήκευσης της ημερήσιας κατανάλωσης ενός προϊόντος θα γινόταν πιο εύκολη για τον χρήστη. Σε άλλη σκοπιά, τίθεται έντονη η ανάγκη ωραιοποίησης και καλλιτεχνικού εμπλουτισμού της εμφάνισης της διεπαφής του χρήστη που ανέπτυξα, ώστε να είναι πιο ξεκάθαρη η πληροφορία που παρουσιάζεται, να συμβάλει στην διαισθητική αλληλεπίδραση και να αποπνέει περισσότερη ώθηση προς δραστηριοποίηση και γαλήνη στον χρήστη. Ωστόσο, η εφαρμογή ακόμα και στην κατάσταση που βρίσκεται αυτήν την στιγμή, προσφέρει λειτουργίες που άλλες εφαρμογές της αγοράς ζητούν μηνιαίο χρηματικό αντίτιμο, για να ξεκλειδώσουν οι χρήστες. Επίσης κατέληξα σε ένα αρκετά οργανωμένο έργο με προδιαγραφές μεγάλης κλίμακας επιχειρησιακού έργου εφαρμογής Android. Με λίγη καλύτερη τακτοποίηση του πηγαίου κώδικα, θα ήταν ένα πολύ ευανάγνωστο και κατ' επέκταση εύκολα επεκτάσιμο, για την μελλοντική υλοποίηση αυτών των χαρακτηριστικών, που προηγουμένως ανέφερα. Το έργο το ανέπτυξα με το μοντέλο αρχιτεκτονικής Clean Code Architecture, που οδηγεί σε πιο testable πηγαίο κώδικα, διαχωρίζοντας τα συστατικά του έργου, ώστε να έχει το καθένα μια αρμοδιότητα και να μην υπάρχει σύμπλεξη μεταξύ τους με αμφίδρομες αλληλοεξαρτήσεις [1]. Μελέτησα αρκετά αυτό το μοντέλο και στο εξής, όλες οι εφαρμογές που θα αναπτύσσω, είτε είναι mobile, desktop ή web, θα ήθελα να ικανοποιούν τις αρχές που το Clean Code διδάσκει. Πρωτίστως, το μονοπάτι των fitness εφαρμογών θα ήθελα να το ακολουθήσω μέχρι τέλους και να ανέβει ολοκληρωμένη η παραγωγική έκδοση της εφαρμογής ThesisFitnessApp στο Google Play Store, έτσι ώστε να εκπληρωθεί το όνειρο της εύκολα προσβάσιμης προσωπικής γυμναστικής και διατροφικής υγείας από όλους, χωρίς χρηματικά κόστη. Σίγουρα όμως, το αίσθημα της κοινότητας με οποιοδήποτε χόμπι καταπιανόμαστε είναι σημαντικό. Γι' αυτό θα ήθελα η έκδοση αυτή που θα ανέβει, να παρέχει και στοιχεία που κάνουν τον χρήστη, εκτός από το να νιώθει γεμάτος, να βιώνει την προσφορά του στην συλλογική ενθάρρυνση και εξέλιξη της υγείας και της φυσικής κατάστασης του συνόλου των χρηστών. Ένα από αυτά τα στοιχεία, θα ήταν η ενσωμάτωση τροφοδοσίας δημοσιεύσεων, που διαμοιράζονται από όλους τους χρήστες. Η δυνοτότητα θετικής ανταπόκρισης με μηχανισμό "like" στις δημοσιεύσεις και "follow" στο προφίλ των χρηστών. Το μέλλον είναι αβέβαιο για τις εφαρμογές καταγραφής φυσικής κατάστασης, δραστηριότητας και διατροφής. Η παγκόσμια αγορά αυτών των εφαρμογών πάντως, φαίνεται ότι έχει προοπτικές για τα επόμενα 5 χρόνια [34]. Το 2022, η παγκόσμια αγορά των εφαρμογών για την φυσική κατάσταση εκτιμήθηκε στα 3.698 εκατομμύρια δολάρια ΗΠΑ και αναμένεται να φτάσει τα 18.374 εκατομμύρια δολάρια ΗΠΑ το 2028, με μια ετήσια ανάπτυξη ποσοστού 30,63% κατά τα έτη πρόβλεψης. Μένει να φανεί σε μελλοντικές έρευνες εάν η σχέση μεταξύ της χρήσης κινητών συσκευών φυσικής δραστηριότητας διατηρείται μακροπρόθεσμα. Τέλος θα μπορούσα να επεκταθώ στον τομέα της υγείας μέσω της εφαρμογής, υλοποιώντας ένα καθολικό API, που ανατρέχει τους διαθέσιμους φυσικοθεραπευτές, για πιθανούς τραυματισμούς, που

συμβαίνουν στους χρήστες. Μιας και η εφαρμογή ενθαρρύνει την γυμναστική χωρίς ειδικούς προπονητές, όσο αποτελεσματικά και αν διδάσκει τις σωματικές ασκήσεις με βίντεο και λεπτομερείς περιγραφές, σίγουρα δεν μπορεί να αντικαταστήσει τους επαγγελματίες προπονητές στην γνώση και την άμεση υποβολή διορθώσεων, που προσφέρουν σε αρχάριους γυμναζόμενους. Με αυτό το χαρακτηριστικό αν συμβεί κάποιο ατύχημα την ώρα της γυμναστικής, οι χρήστες θα εισάγουν το πρόβλημα σε μια οθόνη αναζήτησης, και θα τους προτείνονται οι κατάλληλοι γιατροί ή φυσικοθεραπευτές.

ΒΙΒΛΙΟΓΡΑΦΙΑ

[1] Robert C. Martin, Clean Architecture: A Craftsman's Guide to Software Structure and Design, Pearson; 1st edition (13 September 2017); Pearson Education (I) Pvt. Ltd.

[2] Samar Swaid, “Usability of Mobile Apps: An Integrated Approach”, 2017 AHFE, July 2017, The Westin Bonaventure Hotel, Los Angeles, USA

[4] Tian Lou, (2016, September 6). Eindhoven University of Technology MASTER “A comparison of Android Native App Architecture MVC, MVP and MVVM”, Eindhoven University of Technology

[3] James Olujoba Adegboye, “Usability Analysis for An Android Based Application”, Journal of Women in Technical Education and Employment (JOWITED), The Federal Polytechnic, July 2020

[4] Pankaj Chougale, Vaibhav Yadav, Dr. Anil Gaikwad, “FIREBASE - OVERVIEW AND USAGE”, International Research Journal of Modernization in Engineering Technology and Science, December 2021

[5] Siddhi Sanjay Shinde & Pratibha Adkar, “A Review Paper on Kotlin Programming Language”, International Journal of Trend in Scientific Research and Development (IJTSRD), June 2021

[6] Huong Ly Tong & Carol Maher & Kate Parker & Tien Dung Pham & Ana Luisa Neves & Benjamin Riordan & Clara K. Chow & Liliana Laranjo & Juan C. Quiroz, “The use of mobile apps and fitness trackers to promote healthy behaviors during COVID-19: A cross-sectional survey”, Public Library of Science, Digital Health, August 2022

[7] Arshad Ahmad & Kan Li & Chong Feng & Syed Mohammad Asim & Abdallah Yousif & Shi Ge, “An Empirical Study of Investigating Mobile Applications Development Challenges”, IEEE Access, March 2018

[8] M. E. Joorabchi, A. Mesbah, and P. Kruchten, ‘‘Real challenges in mobile app development,’’ in Proc. ACM/IEEE Int. Symp. Empirical Softw. Eng. Meas., Oct. 2013, pp. 15–24.

[9] I. Malavolta, S. Ruberto, T. Soru, and V. Terragni, ‘‘End users’ perception of hybrid mobile apps in the Google Play Store,’’ in Proc. IEEE Int. Conf. Mobile Services, New York, NY, USA, Jun./Jul. 2015, pp. 25–32.

[10] I. Malavolta, S. Ruberto, T. Soru, and V. Terragni, ‘‘Hybrid mobile apps in the Google Play Store: An exploratory investigation,’’ in Proc. 2nd ACM Int. Conf. Mobile Softw. Eng. Syst., 2015, pp. 56–59.

[11] M. Ali and A. Mesbah, ‘‘Mining and characterizing hybrid apps,’’ in Proc. WAMA, Seattle, WA, USA, 2016, pp. 50–56.

[12] Yanlong Guo, Denghang Chen, Han Zhang, “Factors Influencing Use of Fitness Apps by Adults under Influence of COVID-19”, Int J Environ Res Public Health, November 2022

[13] Mahmud Jabri, (2023, January 5). Java Programming Language Report. ResearchGate. Retrieved January, 2021, Utrecht University

[14] Santhosh Dayala, “Firebase — Cloud Firestore. Cloud Firestore is one of the services…”, Wenable Medium, 2019. [Online]. Available: https://medium.com

[15] RISHU MISHRA, “MVC (Model View Controller) Architecture Pattern in Android”, GeeksforGeeks. 2020. [Online]. Available https://www.geeksforgeeks.org

[16] RISHU MISHRA, “MVP (Model View Presenter) Architecture Pattern in Android with Example”. GeeksforGeeks. 2020. [Online]. Available: https://www.geeksforgeeks.org

[17] Moataz Nabil, “The State of Mobile App Development in 2022”. Bitrise. 2023. [Online]. Available: https://bitrise.io

[18] Leland Richardson, “Understanding Jetpack Compose — part 1 of 2”. Medium. 2020. [Online]. Available: https://medium.com

[19] Margaret Rouse, “What is Android Studio? - Definition from Techopedia”. Techopedia. 2019. [Online]. Available: https://www.techopedia.com

[20] Margaret Rouse, “What is Android SDK? - Definition from Techopedia”. Techopedia. 2020. [Online]. Available: https://www.techopedia.com

[21] Google, “API Guidelines for Jetpack Compose”, March 2021. [Online]. Available: https://github.com/androidx/androidx/blob/androidx-main/compose/docs/

[22] Google, “API Guidelines for @Composable components in Jetpack Compose”, July 2023. [Online]. Available: https://github.com/androidx/androidx/blob/androidx-main/compose/docs/

[23] John Callaham, “The history of Android: The evolution of the biggest mobile OS in the world”, April 2023. [Online]. Available: https://www.androidauthority.com/

[24] Cihan Ilter, “What Are Google Play Store Tags and Categories?”, January 2021. [Online]. Available: https://metrikal.io/blog

[25] Lucie Loubet, “Everything you need to know about Material Design 3”, January 2022. [Online]. Available: https://medium.com/

[26] Google, “API Guidelines for Jetpack Compose”, March 2021. [Online]. Available: https://github.com/androidx/androidx/blob/androidx-main/compose/docs/

[27] Google, “Create a project”, April 2023. [Online]. Available: https://developer.android.com/studio/projects/create-project

[28] Google, “Get started with the Navigation component”, September 2023. [Online]. Available: https://developer.android.com/guide/navigation/get-started

[29] Google, “Services overview”, July 2023. [Online]. Available: https://developer.android.com/guide/components/services

[30] Google, “Dependency Injection with Hilt.”, June 2021. [Online]. Available: https://developer.android.com/training/dependencyinjection/hilt-android

[31] Statista Research Department, “Global market share held by mobile operating systems from 1st quarter 2009 to 2nd quarter 2023”, April 2020. [Online]. Available:

https://www.statista.com/statistics/272698/global-market-share-held-by-mobile-operatingsystems-since-2009/

[32] Laurence Goasduff & Rob van der Meulen,“Gartner Says Huawei Secured No. 2 Worldwide Smartphone Vendor Spot, Surpassing Apple in Second Quarter 2018”, August 2018. [Online]. Available: https://www.gartner.com/en/newsroom/press-releases/2018-08-28-gartner-says-huawei-secured-no-2-worldwide-smartphone-vendorspot-surpassing-apple- in-second-quarter.

[33] J. Winter & S. Battisti & T. Burström & S. Luukkainen, “Exploring the Success Factors of Mobile Business Ecosystems”, Int. J. Innov. Technol. Manage. 15 (3) (2018) 1–23.

[34] Market Growth Reports, “Global Fitness App Industry Research Report, In-depth Analysis of Current Status and Outlook of Key Countries 2023-2028”, SKU ID : Maia-23071014, March 2023. [Online]. Available: https://www.marketgrowthreports.com/global-fitness-app-industry-23071014

[35] Michal Bialas, “The Hottest Jetpack Compose Projects And Libraries on GitHub in 2023”, July 2023. [Online]. Available: https://medium.com/

[36] Codeangi, “Meet “Y-Charts”: an Opensource Jetpack Compose chart library”, July 2023. [Online]. Available: https://medium.com/