

ΠΑΝΕΠΙΣΤΗΜΙΟ ΠΕΙΡΑΙΩΣ
Τμήμα Πληροφορικής



ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

***Ανάπτυξη εφαρμογής διαχείρισης κρατήσεων και συνδρομών
γυμναστηρίου με χρήση Spring Boot, σε λειτουργικό σύστημα Android
Android Application for managing booking and subscriptions of a fitness
center using Spring Boot.***

***Μάριος – Ιωάννης Σταματόπουλος,
Π18144***

Επιβλέπων: Ευθύμιος Αλέπης

Πειραιάς, Ιούνιος 2022





Επιτελική Σύνοψη

Αντικείμενο της παρούσας πτυχιακής εργασίας είναι η ανάπτυξη μιας εφαρμογής κινητού, η οποία εστιάζει στην διαχείριση των κρατήσεων και των συνδρομών ενός γυμναστηρίου. Η εφαρμογή προορίζεται τόσο για τους κατόχους όσο και για τους αθλούμενους του γυμναστηρίου καθώς διαθέτει όλες τις απαραίτητες λειτουργίες και αυτοματισμούς ώστε να πραγματοποιηθεί με ευκολία η διαχείριση των συνδρομών και η κράτηση μαθημάτων. Συγκεκριμένα για τους κατόχους του γυμναστηρίου, η εφαρμογή διαθέτει τον ρόλο του διαχειριστή. Ο ρόλος αυτός περιλαμβάνει λειτουργίες όπως προσθήκη διαχείριση και επεξεργασία νέων μαθημάτων και συνδρομών, προσθήκη διαθεσιμότητας σε υπάρχοντα προγράμματα και εξαγωγή στατιστικών. Επιπλέον διατίθεται και ο ρόλος του πελάτη τον οποίο αποκτούν οι αθλούμενοι με την εγγραφή τους στην εφαρμογή δίνοντας τους δυνατότητες όπως περιήγηση στο εβδομαδιαίο πρόγραμμα των μαθημάτων του γυμναστηρίου, online κράτησης θέσεων στα διάφορα μαθήματα, εμφάνιση της συνδρομής τους καθώς και λεπτομέρειες για την κατάσταση της. Σε επόμενες ενότητες της πτυχιακής παρουσιάζονται αναλυτικότερα οι προαναφερθέντες λειτουργίες και δυνατότητες της εφαρμογής και αναφέρονται οι τεχνολογίες και οι τεχνικές υλοποίησης που χρησιμοποιήθηκαν. Επίσης γίνεται μερική αναφορά σε τεχνικά σημεία της εφαρμογής και η επεξήγηση τους όπου αυτή είναι απαραίτητη. Τέλος παρατίθενται πιθανές βελτιώσεις της εφαρμογής και ιδέες για μελλοντική επέκταση της δίνοντας έτσι μια ολοκληρωμένη και πλήρη εικόνα για το εύρος των δυνατοτήτων που θα μπορούσε να καλύψει.

Abstract

The purpose of this thesis is the development of a mobile application, which focuses on the reservation and membership management of a fitness center. The application is intended for both owners and members of the gym as it provides all the necessary features and functions to easily manage memberships and book classes. More specifically for gym owners, the application has the role of administrator. This role includes functionalities such as adding managing and editing of new courses and subscriptions, adding availability to existing programs and exporting statistics. In addition, the customer role is also available, which is acquired by the gym members with their registration to the application, giving them features such as browsing the weekly schedule of the gym's classes, online booking of the various lessons, displaying their membership and details of its status. In the following sections of the thesis, the aforementioned functions and features of the application and also the technologies and implementation techniques are presented in greater detail. There is also a partial reference to technical aspects of the application and their explanation where necessary. Finally, possible improvements to the application and ideas for future extensions are listed, thus giving a comprehensive and complete picture of the range of possibilities it could cover.



Περιεχόμενα

1	Εισαγωγή	5
1.1	Δυνατότητες εφαρμογής	5
1.2	Τεχνολογίες που χρησιμοποιήθηκαν	6
1.2.1	Java Spring Boot.....	6
1.2.2	Android.....	7
2	Ανάλυση σχεδιασμού.....	8
2.1	Backend Spring Boot – PostgreSQL	8
2.1.1	Αρχιτεκτονική Spring Boot	8
2.1.2	Η Δομή του Project	9
2.1.3	Μηχανισμοί ασφάλειας	13
2.1.4	Παρουσίαση βάσης δεδομένων	16
2.2	Frontend Android	17
2.2.1	Αρχιτεκτονική Android	17
2.2.2	Η δομή της εφαρμογής	18
2.2.3	Παρουσίαση των packages	19
3	Παρουσίαση εφαρμογής.....	24
4	Συμπεράσματα και μελλοντικές επεκτάσεις	45

Κεφάλαιο 1^ο

1 Εισαγωγή

Είναι γεγονός πως τα τελευταία δύο χρόνια άλλαξαν δραματικά την σχέση του ανθρώπου με την τεχνολογία καθώς η πανδημία του κορονοϊού έφερε σε περιορισμό τις καθημερινές δραστηριότητες, υποχρεώσεις και κοινωνικές συναναστροφές. Πολλές επιχειρήσεις πάγωσαν και εργαζόμενοι βρέθηκαν σε απόγνωση εφόσον δεν υπήρχαν οι κατάλληλες υποδομές για εξ αποστάσεως εργασία ή για εργασία με περιοριστικά μέτρα και σε ορισμένες περιπτώσεις αυτό ήταν σχεδόν αδύνατο να υλοποιηθεί.

Πλέον εσωτερικοί χώροι, όπως για παράδειγμα χώροι άθλησης, βρέθηκαν αντιμέτωποι με τις δυσκολίες που είχε η επιστροφή στην πραγματικότητα καθώς έπρεπε να συμβαδίζουν και να λειτουργούν με τα περιοριστικά μέτρα εξάπλωσης του κορονοϊού. Με την αξιοποίηση της τεχνολογίας υπήρξε μεγάλη ζήτηση για την ανάπτυξη έξυπνων εφαρμογών και υλοποιήσεων που θα μπορούσαν να προσφέρουν άμεσες και επικερδείς λύσεις στα προβλήματα που εμφανίστηκαν και να καταστήσουν ως ένα βαθμό εφικτή την υλοποίηση των επαγγελματικών δραστηριοτήτων. Μέσα στο πλαίσιο της κατάστασης αυτής βασίστηκε η υλοποίηση της παρούσας εφαρμογής στην οποία οι αθλούμενοι θα μπορούν να επιλέξουν τις ώρες που θέλουν να γυμναστούν και θα πραγματοποιούν την κράτηση τους με ευκολία, οπουδήποτε και οποτεδήποτε με την χρήση κινητής συσκευής. Ως αποτέλεσμα οι διαχειριστές από την μεριά τους θα έχουν την δυνατότητα να επεξεργάζονται τις ώρες μαθημάτων καθώς και να διατρέχουν κάποια χρήσιμα στατιστικά για τα μαθήματα και τις ώρες που αυτά πραγματοποιούνται, με σκοπό να τροποποιούν κατάλληλα και με ευελιξία τα προγράμματα των μαθημάτων έτσι ώστε να μην παρατηρείται συνωστισμός στους διάφορους χώρους προπόνησης του γυμναστηρίου.

1.1 Δυνατότητες εφαρμογής

Η εφαρμογή ανάλογα με την ιδιότητα του χρήστη παρέχει ξεχωριστές λειτουργίες για τον καθένα. Παρακάτω αναγράφονται αναφορικά οι λειτουργίες αυτές.

Δυνατότητες Διαχειριστή

- Σύνδεση στην εφαρμογή
- Προβολή του προσωπικού προφίλ και επεξεργασία των στοιχείων του
- Προβολή των μαθημάτων
- Εισαγωγή νέου μαθήματος
- Επεξεργασία των στοιχείων του μαθήματος
- Εισαγωγή μέρας και ώρας που θα είναι διαθέσιμο κάποιο μάθημα
- Διαγραφή διαθεσιμότητας
- Εισαγωγή νέας συνδρομής για έναν πελάτη
- Ανανέωση συνδρομής
- Εμφάνιση στατιστικών για τις συνδρομές
- Εμφάνιση στατιστικών για τις κρατήσεις
- Εμφάνιση δημοφιλών ωρών κράτησης



Δυνατότητες Πελάτη

- Εγγραφή στην εφαρμογή
- Σύνδεση
- Προβολή του προσωπικού προφίλ και επεξεργασία των στοιχείων του
- Προβολή του εβδομαδιαίου προγράμματος
- Επιλογή μέρας και ώρας για την κράτηση μαθημάτων
- Προβολή των τρεχουσών κρατήσεων και της κατάστασής τους
- Ακύρωση τρέχουσας κράτησης
- Προβολή παλαιότερων κρατήσεων
- Προβολή της συνδρομής του καθώς και λεπτομέρειες για αυτή

1.2 Τεχνολογίες που χρησιμοποιήθηκαν

Η υλοποίηση της εφαρμογής χωρίστηκε σε δύο μέρη, το backend και το frontend. Ως backend εννοούμε το επίπεδο που διαχειρίζεται τα δεδομένα, εκεί βρίσκεται το REST API της εφαρμογής το οποίο υλοποιήθηκε με το Spring Framework της Java. Το front end πρόκειται για το επίπεδο παρουσίασης της εφαρμογής, σε αυτό χρησιμοποιήθηκε το Android το οποίο λειτουργεί ως λειτουργικό σύστημα αλλά και ως πλατφόρμα στις κινητές συσκευές. Η γλώσσα προγραμματισμού που έκτισε την εφαρμογή είναι η Java και τα περιβάλλοντα ανάπτυξης του λογισμικού που βοήθησαν στην ανάπτυξη του κώδικα είναι το IntelliJ IDEA και το Android Studio. Τέλος ως βάση δεδομένων χρησιμοποιήθηκε η σχεσιακή βάση PostgreSQL. Παρακάτω γίνεται αναφορά στα σημαντικότερα εργαλεία που βοήθησαν στην ανάπτυξη της εφαρμογής.

1.2.1 Java Spring Boot

Το Java Spring Boot ή απλώς Spring Boot, πρόκειται για ένα δημοφιλές εργαλείο το οποίο καθιστά την ανάπτυξη έτοιμων για την παραγωγή, εφαρμογών και μικρο-υπηρεσιών εύκολη και γρήγορη, με ελάχιστες διαμορφώσεις και ρυθμίσεις, χάρη στις βασικές δυνατότητες και τα χαρακτηριστικά του. Το εργαλείο αυτό βασίζεται στο Java Spring Framework το οποίο είναι ανοικτού κώδικα, επιχειρησιακού επιπέδου framework για τη δημιουργία αυτόνομων και παραγωγικών εφαρμογών που εκτελούνται στην Εικονική Μηχανή Java (JVM). Μερικά από τα χαρακτηριστικά και τα πλεονεκτήματα που προσφέρει είναι τα εξής:

- Παρέχει έναν ευέλικτο τρόπο διαμόρφωσης των Java Beans, των XML configuration αρχείων και των συναλλαγών στην βάση δεδομένων.
- Περιέχει dependencies τα οποία διαχειρίζεται και απλοποιούν την ανάπτυξη και την παραμετροποίηση της εφαρμογής.
- Περιλαμβάνει ενσωματωμένο Servlet Container.
- Παρέχει ενσωματωμένο server και διαχειρίζεται τερματικά σημεία REST.
- Παρέχει annotations.
- Προσφέρει την δυνατότητα παροχής μετρήσεων (metrics) και ελέγχου κατάστασης.

Ο λόγος που επιλέχθηκε η χρήση αυτού του framework βασίζεται στα πλεονεκτήματά του και το πλήθος των βιβλιοθηκών και dependencies που υπάρχουν για την εύκολη υλοποίηση των διάφορων λειτουργιών. Επίσης πρόκειται για ένα δημοφιλές framework στον χώρο των REST APIs, και των Web εφαρμογών το οποίο συνεχίζει να εξελίσσεται και να βελτιώνεται.



1.2.2 Android

Το Android πρόκειται για ένα λειτουργικό σύστημα ανοικτού κώδικα, βασισμένο σε Linux και πλέον εντοπίζεται σε μια μεγάλη γκάμα έξυπνων συσκευών, smartphones, tablets, laptops wearables κ.α. Πρωτοεμφανίστηκε για πρώτη φορά το 2007 από την Google ενώ η πρώτη εμπορική έκδοση το Android 1.0 κυκλοφόρησε ένα χρόνο μετά. Με την πάροδο των χρόνων το Android εξελισσόταν και βελτιωνόταν σε διάφορους τομείς με επίκεντρο την καθημερινή χρήση και διεπαφή του χρήστη με το λογισμικό, την ασφάλεια, την ιδιωτικότητα την προσβασιμότητα κ.α. Κάποια από τα πιο σημαντικά χαρακτηριστικά του Android από τα οποία απέκτησε μεγάλη διασημότητα στον χώρο της τεχνολογίας και προτιμήθηκε ως front end υλοποίηση για την παρούσα πτυχιακή είναι τα εξής:

- Περιέχει εύχρηστο, απλό, ευέλικτο και διαισθητικό περιβάλλον εργασίας χρήστη. Το UI του βασίζεται στον άμεσο χειρισμό, που σημαίνει ότι χρησιμοποιώντας την αφή και διάφορες χειρονομίες ο χρήστης μπορεί να πλοηγηθεί στις διάφορες λειτουργίες της οθόνης.
- Λόγω του ελεύθερου διαθέσιμου λογισμικού και των δυνατοτήτων που αυτό παρέχει συνεχώς αυξάνεται ο αριθμός των διαθέσιμων εφαρμογών που υπάρχουν για το Android. Επομένως οι χρήστες μέσα από καταστήματα εφαρμογών όπως το Google Play έχουν την δυνατότητα να αναζητούν και κατεβάζουν τις εφαρμογές είτε δωρεάν είτε επί πληρωμή τις διάφορες εφαρμογές που επιθυμούν
- Προσφέρει multi-tasking καθώς ο χρήστης μπορεί εύκολα να μεταβεί από την μία εφαρμογή στην άλλη και ταυτόχρονα διάφορες εφαρμογές μπορούν να εκτελούνται παράλληλα είτε στο παρασκήνιο είτε στην οθόνη του χρήστη.
- Παρέχει υποστήριξη σε πολλές γλώσσες, επομένως προσφέρει την δυνατότητα σε ανθρώπους διαφόρων εθνικοτήτων να μπορούν να γνωρίσουν και να χρησιμοποιήσουν το Android στην μητρική τους γλώσσα.
- Μπορεί και επικοινωνεί με το Internet με ασφάλεια. Με την χρήση των κατάλληλων εργαλείων και την ασύγχρονη λειτουργία καθιστά εύκολη την επικοινωνία μεταξύ κάποιου web server και την κατανάλωση APIs
- Τέλος ένας ακόμη λόγος που το Android είναι δημοφιλές από την μεριά των προγραμματιστών είναι το ισχυρό περιβάλλον ανάπτυξης του. Παρόλο που υπάρχουν πολλά περιβάλλοντα ανάπτυξης Android ο κυρίαρχος τρόπος είναι η χρήση του Android SDK ή του Android Studio. Τα εργαλεία που παρέχονται προσφέρουν ένα ολοκληρωμένο και πλήρες περιβάλλον ανάπτυξης με πολλές δυνατότητες.

Κεφάλαιο 2^ο

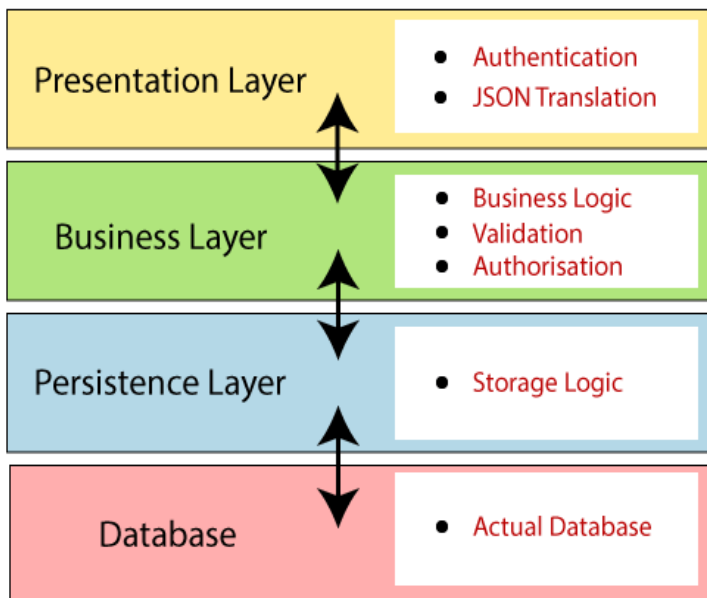
2 Ανάλυση σχεδιασμού

2.1 Backend Spring Boot – PostgreSQL

Στο κεφάλαιο αυτό αρχικά θα παρουσιαστούν τα διάφορα επίπεδα που ακολουθεί η πολυεπίπεδη αρχιτεκτονική του Spring Boot σχετίζοντάς την με την αρχιτεκτονική της παρούσας εφαρμογής. Στη συνέχεια θα περιγράψουμε πως τα διάφορα αυτά επίπεδα επικοινωνούν μεταξύ τους και πως είναι δομημένα στα διάφορα packages της εφαρμογής. Επιπλέον θα γίνει αναφορά στους μηχανισμούς ασφάλειας της εφαρμογής και στο πως αυτοί υλοποιούνται. Τέλος θα δοθεί μια εικόνα για την δομή της βάσης δεδομένων και τους πίνακες που αποτελείται.

2.1.1 Αρχιτεκτονική Spring Boot

Η αρχιτεκτονική του Spring Boot βασίζεται στο παρακάτω σχεδιάγραμμα.



Εικόνα 2.1 Η αρχιτεκτονική μιας Spring Boot εφαρμογής

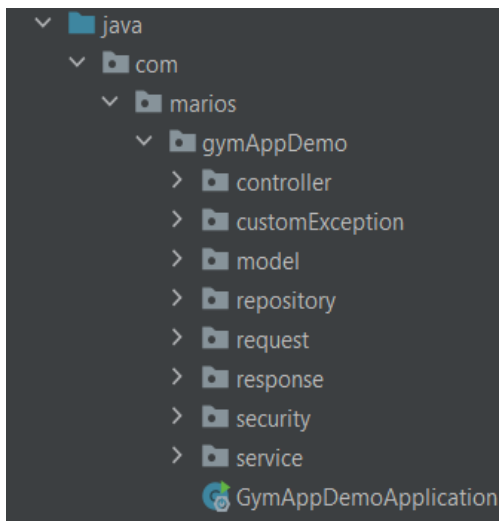
- **Presentation Layer:** Το επίπεδο παρουσίασης αφορά τις λειτουργίες που συμβαίνουν στο frontend κομμάτι και προβάλλονται στην μεριά του χρήστη. Το επίπεδο αυτό διαχειρίζεται HTTP requests, μπορεί και μεταφράζει για παράδειγμα μία JSON παράμετρο σε ένα αντικείμενο της Java, μπορεί να ελέγχει την αυθεντικότητα του μηνύματος και να επικοινωνεί με το Business Layer ώστε να στείλει ή να λάβει δεδομένα.
- **Business Layer:** Το επίπεδο αυτό χειρίζεται όλη την επιχειρησιακή λογική. Αποτελείται από κλάσεις και διάφορες υπηρεσίες οι οποίες επεξεργάζονται όλη την πληροφορία που μπορεί

να δέχονται από το επίπεδο παρουσίασης και ύστερα την επιστρέφει ή την μεταφέρει σε άλλο επίπεδο ώστε να επιτευχθεί η αποθήκευσή της. Στο επίπεδο αυτό επίσης εκτελούνται υπηρεσίες εξουσιοδότησης και η επικύρωση δεδομένων.

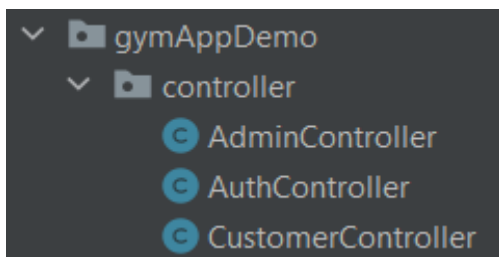
- **Persistence Layer:** Σε αυτό το επίπεδο περιέχεται όλη η λογική της αποθήκευσης και της άντλησης δεδομένων από την βάση.
- **Database Layer:** Το επίπεδο της βάσης δεδομένων αφορά όλες τις CRUD λειτουργίες (Create, Retrieve, Update, Delete) που πραγματοποιούνται στα δεδομένα της βάσης.

2.1.2 Η Δομή του Project

Η παραπάνω αρχιτεκτονική εκφράζει μια γενική εικόνα των επιπέδων σε μία εφαρμογή. Τα packages controller, model, repository και service που φαίνονται στην εικόνα, καθώς και οι επιμέρους κλάσεις τους, έχουν δημιουργηθεί με τέτοιο τρόπο ώστε να ακολουθούν την αρχιτεκτονική εφαρμογής που περιγράψαμε. Τα επίπεδα που παρατηρούνται είναι το Controller Layer, το Service Layer και το Data Access Layer.



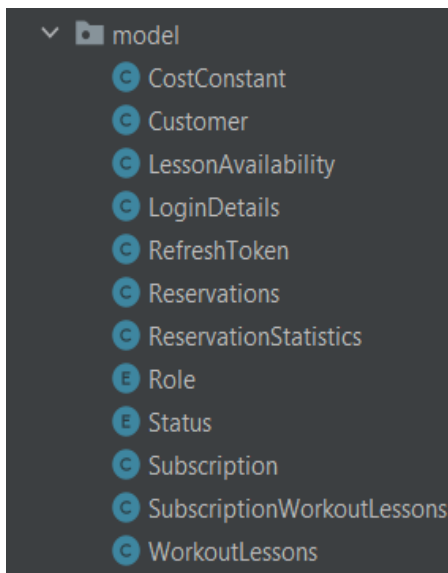
Εικόνα 2.2 Η ιεραρχία των packages



Εικόνα 2.3 Controller package

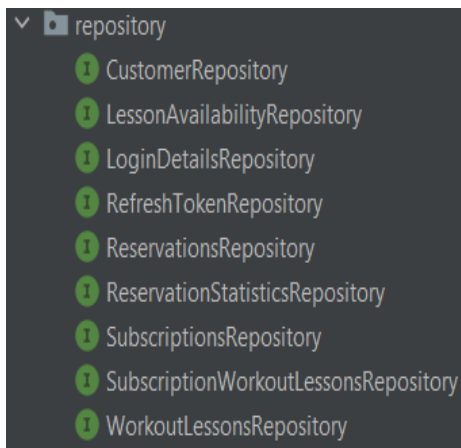
Το Controller package (**Εικόνα 2.3**) αποτελεί το Controller Layer της αρχιτεκτονικής και περιέχει κλάσεις οι οποίες είναι υπεύθυνες να επεξεργάζονται εισερχόμενα REST API requests, και να επιστρέφουν δεδομένα ως απάντηση στο frontend μέρος της εφαρμογής. Οι κλάσεις που εμπεριέχονται είναι οι AdminController, AuthController, CustomerController. Ο κάθε controller χειρίζεται την πληροφορία που διαβιβάζεται από τα διάφορα HTTP requests διαφορετικά και ανάλογα με την οντότητα που προσπαθεί να στείλει ή να πάρει δεδομένα καλεί μία ή περισσότερες υπηρεσίες από το κατάλληλο Service Layer. Παρακάτω περιγράφονται οι controllers που αναφέρθηκαν.

- **Auth Controller:** Η κλάση αυτή αποτελεί τον κόμβο επικοινωνίας των Admin και Customers με το σύστημα ώστε να εκτελεστούν λειτουργίες εγγραφής, σύνδεσης, αποσύνδεσης και ανανέωσης του JWT token.
- **Admin Controller:** Στην κλάση αυτή δίνεται πρόσβαση σε λειτουργίες που αφορούν μόνο τον Admin όπως για παράδειγμα εγγραφή νέου προγράμματος, εισαγωγή συνδρομής πελάτη, εύρεση συνδρομής κ.α.
- **Customer Controller:** Ο controller αυτός αφορά έναν Customer και περιλαμβάνει λειτουργίες όπως προβολή διαθέσιμων μαθημάτων για κράτηση, προβολή συνδρομής, πληροφορίες για την κράτηση κ.α.



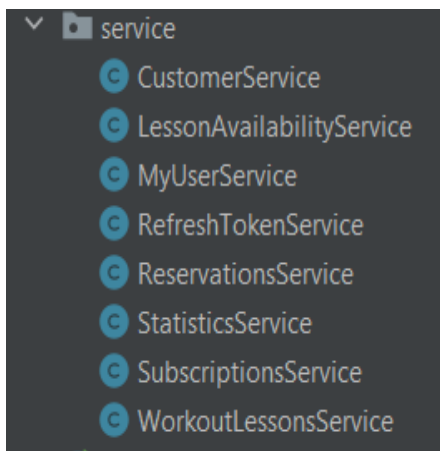
Εικόνα 2.4 Model package

Το Model package (**Εικόνα 2.4**) περιέχει όλες τις οντότητες (Entities) της εφαρμογής μας καθώς και κάποια χρήσιμα Enums και Constants. Κάθε οντότητα είναι ένα POJO που αναπαριστά δεδομένα και αντιστοιχεί σε έναν πίνακα στην βάση.



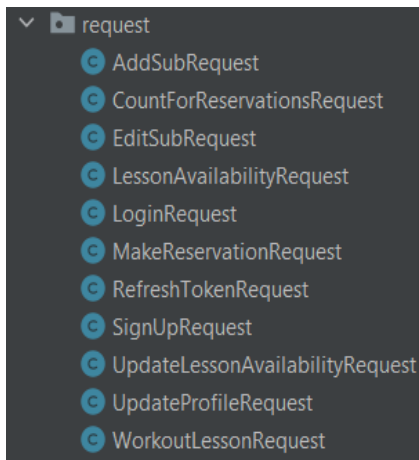
Εικόνα 2.5 Repository package

Τα Repositories (**Εικόνα 2.5**) αποτελούν τους κόμβους σύνδεσης του back end μέρους (Service layer), με την βάση (Data Access Layer). Υποστηρίζονται από την Spring για να διευκολύνουν την χρήση διαφορετικών τεχνολογιών πρόσβασης σε δεδομένα, όπως το JDBC, το Hibernate, το JPA. Στην παρούσα εφαρμογή χρησιμοποιείται το JpaRepository interface το οποίο να προσφέρει αυτοματοποιημένα και γρήγορα CRUD λειτουργίες σε κάθε οντότητα χωρίς να χρειάζεται η δημιουργία SQL queries από την μεριά του προγραμματιστή.

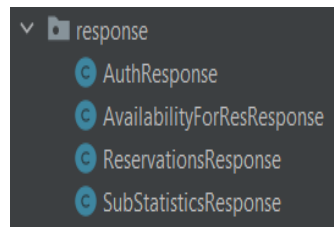


Εικόνα 2.6 Service package

Το Service package (**Εικόνα 2.6**) αναπαριστά το Service Layer. Περιέχει τις κλάσεις service όπου ενθυλακώνουν την επιχειρηματική λογική της εφαρμογής. Τα services διαχειρίζονται και επεξεργάζονται τα διάφορα δεδομένα που αποστέλλονται από και προς την μεριά του client (Controller) και του Data Access Layer (Repository). Για παράδειγμα περιλαμβάνουν λειτουργίες όπως, επικοινωνία με την βάση δεδομένων μέσω των repositories και η επιστροφή των δεδομένων της βάσης ως αντικείμενα της Java, ή λίστες αντικειμένων, αφού γίνουν οι απαραίτητοι έλεγχοι αλλά και λειτουργίες επεξεργασίας δεδομένων που αφορούν είτε δεδομένα τα οποία στάλθηκαν από τους controllers είτε αποκτήθηκαν από μια αναζήτηση στην βάση δεδομένων.

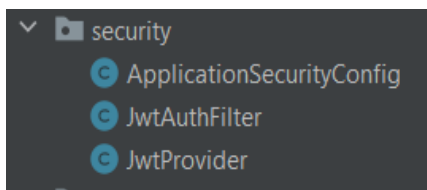


Εικόνα 2.7 Request package



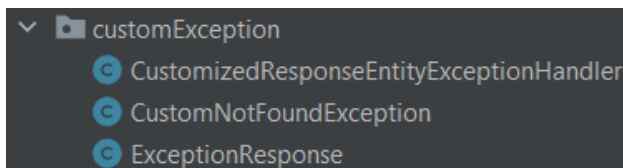
Εικόνα 2.8 Response package

Στα Request, Response packages (**Εικόνες 2.7, 2.8**) βρίσκονται οι κλάσεις που τροφοδοτούνται με δεδομένα που είτε προέρχονται από requests στη μεριά του client (JSON δεδομένα τα οποία μετατρέπονται σε αντικείμενα Java), είτε προέρχονται από την μεριά του Business layer της εφαρμογής ώστε να μετατραπούν σε JSON κατά την αποστολή ενός request. Πρόκειται δηλαδή για DTO(Data Transfer Object) αντικείμενα τα οποία μεταφέρουν δεδομένα μεταξύ των διάφορων επιπέδων στην εφαρμογή.



Εικόνα 2.9 Security package

Το Security package (**Εικόνα 2.9**) Περιέχει τις κλάσεις στις οποίες υλοποιούνται οι απαραίτητοι μηχανισμοί ασφάλειας (πχ Authentication, Authorization)



Εικόνα 2.10 Custom Exception package

Το CustomException package (**Εικόνα 2.10**) Περιέχει κλάσεις οι οποίες διαχειρίζονται τα διάφορα exceptions που μπορεί να συμβούν κατά την λειτουργία της εφαρμογής ώστε να επιστραφούν κατάλληλα μηνύματα στον χρήστη.



2.1.3 Μηχανισμοί ασφάλειας

Για την υλοποίηση των μηχανισμών ασφαλείας για το back end χρησιμοποιήθηκε το Spring Security framework το οποίο περιλαμβάνει ένα εύρος εργαλείων και αποτελεί πρότυπο για την ασφάλεια των εφαρμογών που βασίζονται σε Spring. Οι κύριες απαιτήσεις ασφαλείας που υλοποιήθηκαν με το Spring Security είναι η αυθεντικοποίηση (Authentication) και η εξουσιοδότηση (Authorization). Με τον όρο αυθεντικοποίηση εννοούμε την διαδικασία της ταυτοποίησης ενός χρήστη που θέλει να αποκτήσει πρόσβαση σε κάποια πληροφορία, ενώ εξουσιοδότηση είναι η διαδικασία του να επιτρέπουμε την πρόσβαση σε συγκεκριμένους πόρους της εφαρμογής μας μόνο σε χρήστες που έχουν το δικαίωμα να αποκτήσουν πρόσβαση σε αυτούς.

JWT Tokens

Τα JWT tokens χρησιμοποιούνται ως μηχανισμός επαλήθευσης του ιδιοκτήτη κάποιων δεδομένων (πχ JSON data), τα οποία μπορεί να στέλνει σε ένα end point του API ή να περιμένει να τα παραλάβει από αυτό. Πρόκειται για μια κωδικοποιημένη συμβολοσειρά η οποία περιέχει απεριόριστη πληροφορία. Τα JWT tokens αποτελούνται από 3 μέρη Header, Payload και Signature όπου χωρίζονται μεταξύ τους με τελεία Header.Payload.Signature (**Εικόνα 2.11**).

- **Header:** Περιέχει τον τύπο του token (JWT) και τον αλγόριθμο υπογραφής που χρησιμοποιείται. Στην περίπτωσή μας χρησιμοποιείται ο RS256 αλγόριθμος.
- **Payload:** Περιέχει στοιχεία σχετικά με την οντότητα η και άλλες πρόσθετες σημαντικές πληροφορίες. Στην περίπτωσή μας περιέχεται ο ρόλος του χρήστη, το timestamp της στιγμής δημιουργίας του token και το timestamp της λήξης του.
- **Signature:** Η υπογραφή χρησιμοποιείται ως διαπιστευτήριο ότι το μήνυμα δεν έχει αλλοιωθεί και ότι ο αποστολέας του μηνύματος είναι αυτός που ισχυρίζεται ότι είναι. Η υπογραφή δημιουργείται λαμβάνοντας σε κωδικοποιημένη μορφή Base64 το μέρος header, το payload, ένα μυστικό κωδικό και τον αλγόριθμο που ορίζεται στην επικεφαλίδα.

Encoded

PASTE A TOKEN HERE

```
eyJhbGciOiJSUzI1NiJ9.eyJzdWIiOiJhZG1pbI  
IsIm1hdCI6MTY1MDk4NjU2OCwiZXhwIjoxNjUxM  
Dc2NTY4fQ.Dca_hYAc5JDzPi-iodNK0m-  
gPm6JdVQp6r5368eekMJysi4FKlJg055tavx8Ux  
RdqbejwZSgwUVXN8qGDgkmm4e7F1UAY5-  
5yD3gaZ1lVXJAmt5o40gtHwwujyORkM7gbWD6pD  
cgLBVO_B6ujDpRBUhbAF0ymS60qx2eKtFdjCI00  
jDrFgfzI_9y1ax3xuC0jcLa_mmkE0h3ajGJ2B12  
cQdMc57Ez1kPy5xK02p9FZKZBqbPUXqD6C1Botw  
E5xmN9xV14mY8ngngs07xWygwHmeheRfrdHIIbM  
VdNwsM7rVF0CyVNHKdP5bqtaP-  
sA4xksDqMSL8RZMTiU70dY7dSw
```

Decoded

EDIT THE PAYLOAD AND SECRET

HEADER: ALGORITHM & TOKEN TYPE

```
{  
  "alg": "RS256"  
}
```

PAYLOAD: DATA

```
{  
  "sub": "admin",  
  "iat": 1650986568,  
  "exp": 1651076568  
}
```

VERIFY SIGNATURE

```
RSASHA256(  
  base64UrlEncode(header) + "." +  
  base64UrlEncode(payload),  
  Public Key in SPKI, PKCS #1,  
  X.509 Certificate, or JWK string  
  format.  
  
  Private Key in PKCS #8, PKCS #  
  1, or JWK string format. The k  
  ey never leaves your browser.  
)
```

Εικόνα 2.11 Παρουσίαση ενός JWT Token σε encoded και decoded μορφή

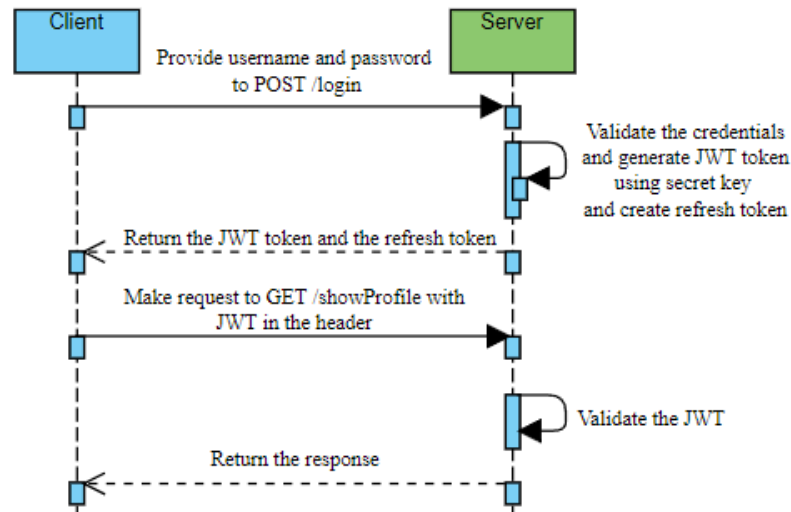
Authentication

Το spring security framework προσφέρει μια αλυσίδα φίλτρων η οποία παρακολουθεί όλα τα εισερχόμενα requests. Η αλυσίδα αυτή μπορεί να περιέχει διάφορα φίλτρα και καθένα από αυτά να διαχειρίζεται διαφορετικές λειτουργίες. Στην περίπτωση της παρούσας εφαρμογής η αυθεντικοποίηση επιτυγχάνεται με JWT tokens και η διαδικασία της υλοποιείται σε συγκεκριμένα end points στα οποία ο χρήστης πρέπει να είναι αυθεντικοποιημένος και να έχει κάποιο ρόλο για να πάρει πρόσβαση σε πληροφορίες. Για παράδειγμα σημεία πρόσβασης όπως η φόρμα εγγραφής ή η φόρμα σύνδεσης, είναι ανοικτά σε οποιοδήποτε χρήστη και δεν απαιτούν αυθεντικοποίηση με κάποιο JWT token. Η διαδικασία αυθεντικοποίησης γίνεται ως εξής (Εικόνα 2.12).

- Αρχικά ένας εγγεγραμμένος χρήστης στέλνει τα στοιχεία του (username, password) στον server για να πιστοποιηθεί (Login request).
- Το end point που είναι υπεύθυνο για να καταναλώσει ένα αίτημα login διαβάζει τα στοιχεία του χρήστη και εφόσον αυτά πιστοποιηθούν και είναι έγκυρα τότε δημιουργείται ένα JWT token και ένα refresh token και επιστρέφονται. Το refresh token πρόκειται για ένα UUID, (αναγνωριστικό κλειδί) το οποίο αποθηκεύετε στην βάση και συσχετίζεται με τον χρήστη. Επιτρέπει στην δημιουργία νέων JWT tokens όταν αυτά λήξουν χωρίς να ζητηθεί η επανασύνδεση και επαναπιστοποίηση του χρήστη στην εφαρμογή.
- Ο client για να μπορέσει να πάρει πρόσβαση στους διάφορους κόμβους, παρέχει το JWT token που το δόθηκε, στην επικεφαλίδα Authorization του HTTP request στην μορφή "Bearer TOKEN". Το back end αφού φιλτράρει το request, θα ελέγξει την εγκυρότητα του

token και τα στοιχεία που δίνονται για τον χρήστη στο payload μέρος(username, role..). Εάν ο έλεγχος είναι επιτυχής και τα στοιχεία του χρήστη επιβεβαιωθούν το αίτημα θα εξουσιοδοτηθεί.

- Κατά την διαδικασία της αποσύνδεσης (logout) το refresh token διαγράφεται από την βάση δεδομένων, και κλείνει το session του χρήστη. Ως αποτέλεσμα, ο χρήστης δεν θα μπορέσει να πάρει πρόσβαση σε end point το οποίο προστατεύεται, εάν στείλει κάποιο request. Επίσης με την διαγραφή του refresh token δεν θα μπορέσει να γίνει η ανανέωση του JWT token εάν αυτό λήξει οπότε ο χρήστης θα πρέπει να επανασυνδεθεί με τα στοιχεία του.



Εικόνα 2.12 Διάγραμμα αυθεντικοποίησης

Τέλος ενδιαφέρον εμφανίζει η αποθήκευση των προσωπικών στοιχείων του χρήστη, όταν αυτός εγγράφεται στην βάση. Κατά την εγγραφή του χρήστη δημιουργείται μια νέα προσθήκη στον πίνακα που είναι υπεύθυνος να κρατάει τα στοιχεία του χρήστη (username, password, role, κ.α.). Ο κωδικός του κρυπτογραφείται με BCrypt αλγόριθμο one way κρυπτογράφησης και έχει την μορφή `$2b$[cost]$[22 character salt][31 character hash]`.

Κατά την αυθεντικοποίηση του χρήστη εφόσον επικυρωθεί το JWT token, ο πίνακας αυτός διατρέχεται ώστε να αναζητηθεί ο χρήστης και να γίνει η ταυτοποίηση των στοιχείων του.

Authorization

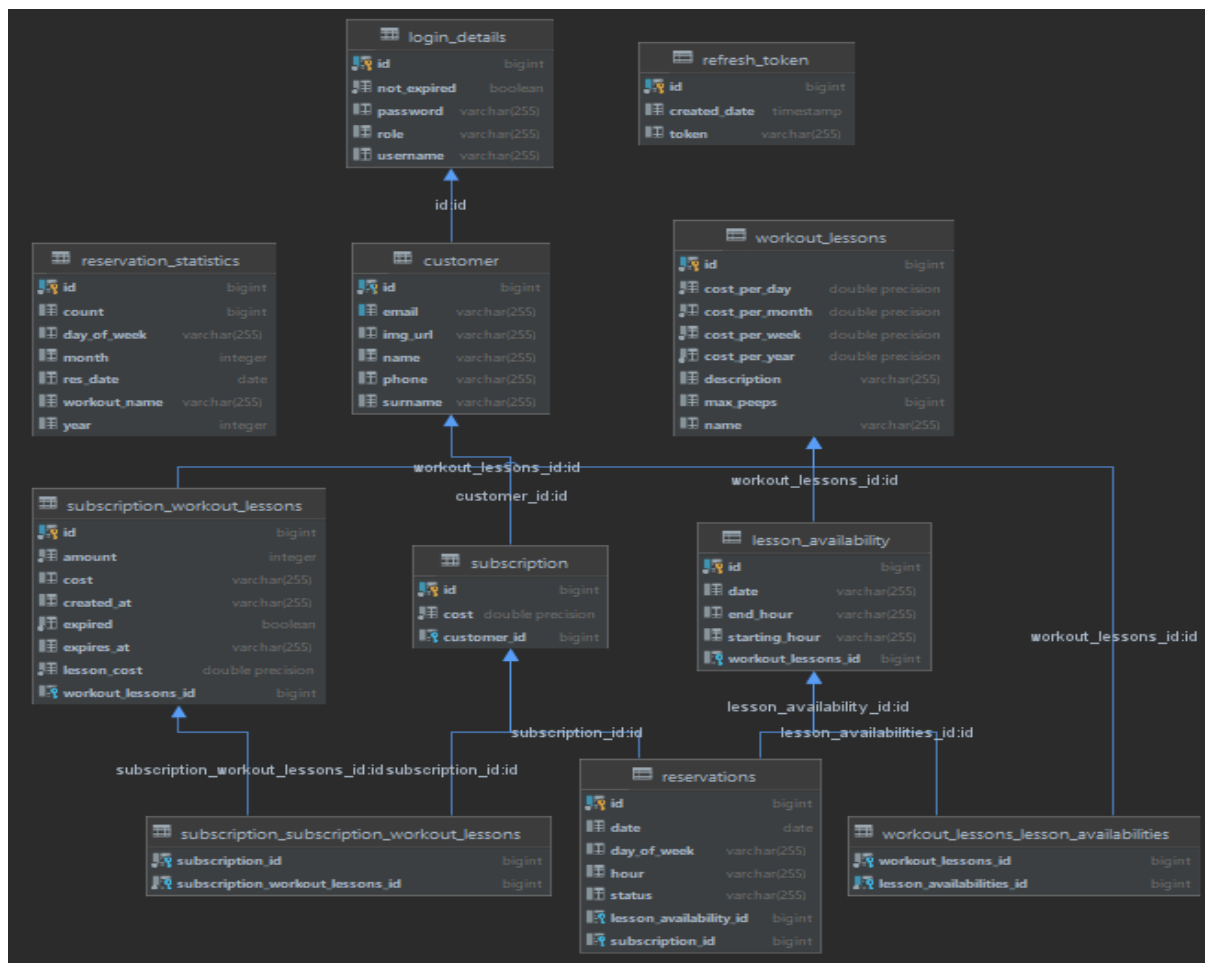
Ο χρήστης αποκτά συγκεκριμένα authorities στην εφαρμογή που αντιστοιχούν στον ρόλο του Πελάτη ή του Διαχειριστή ώστε να μπορεί έχει πρόσβαση σε συγκεκριμένες λειτουργίες. Αυτό επιτυγχάνεται μέσω URL securing σε ορισμένα end points προκειμένου να απορριφθεί η πρόσβαση σε μη εξουσιοδοτημένους χρήστες. Επιπλέον μέσω του payload στο JWT token που στέλνει ο χρήστης υπάρχει η πληροφορία για τον ρόλο του.

2.1.4 Παρουσίαση βάσης δεδομένων

Spring Data JPA

Πρόκειται για ένα dependency του Spring boot το οποίο παρέχει λειτουργικότητα και κάποια πρότυπα στα εργαλεία ORM(Object Relational Mapping). Χρησιμοποιείται για τον έλεγχο και την αντιστοίχιση των δεδομένων μεταξύ αντικειμένων της Java και των σχεσιακών βάσεων. Τα εργαλεία ORM όπως για παράδειγμα το Hibernate υλοποιούν τις προδιαγραφές JPA και βοηθούν στην ανάπτυξη της Java εφαρμογής και στην αλληλεπίδρασή της με την βάση δεδομένων. Στην παρούσα εφαρμογή μας βοηθάει στην αντιστοίχιση των διάφορων οντοτήτων (πχ. Customer, Lessons) σε τύπους δεδομένων SQL, χωρίς να χρειάζεται να γράψουμε SQL κώδικα. Επίσης για την προσθήκη συγκεκριμένων ιδιοτήτων και περιορισμών στις διάφορες στήλες του πίνακα (π.χ. primary keys, unique keys, κ.α.) μας προσφέρει ένα σύνολο από JPA Annotations. Σαν τελευταίο επίπεδο πριν την επικοινωνία με την βάση δεδομένων χρησιμοποιείται το JDBC (Java Database Connectivity) ώστε να δημιουργηθεί επιτυχώς η σύνδεση με το DBMS(Database Management System) της βάσης.

Παρακάτω (Εικόνα 2.13) δίνεται το σχεσιακό σχήμα της ΒΔ, στο οποίο αναπαρίστανται οι πίνακες που έχουν δημιουργηθεί, καθώς και οι σχέσεις μεταξύ τους.



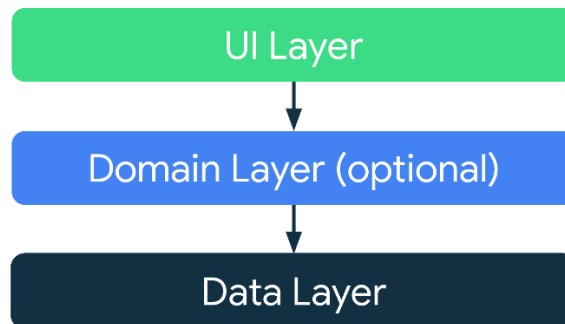
Εικόνα 2.13 Σχεσιακό σχήμα της ΒΔ

2.2 Frontend Android

Στο κεφάλαιο αυτό θα αναλυθεί η βασική αρχιτεκτονική μιας εφαρμογής σε Android σύμφωνα με κάποιες αρχιτεκτονικές αρχές και θα επιτευχθεί η σύγκριση της μεταξύ της αρχιτεκτονικής της παρούσας εφαρμογής. Στην συνέχεια θα αναλυθεί η δομή της παρούσας εφαρμογής παρουσιάζοντας τα περιεχόμενα των packages του project. Τέλος γίνεται αναφορά στην επικοινωνία του backend με το frontend μέρος καθώς και στις βιβλιοθήκες που χρησιμοποιήθηκαν, για την πλήρη λειτουργία της εφαρμογής.

2.2.1 Αρχιτεκτονική Android

Η βέλτιστες αρχιτεκτονικές για κάθε εφαρμογή είναι διαφορετικές και εξαρτώνται από τις απαιτήσεις και της ανάγκης του κάθε προγραμματιστή. Η βασική ιδέα είναι να οργανωθούν τα διάφορα components που συνυπάρχουν στην εφαρμογή με τέτοιο τρόπο ώστε να διευκολύνουν γενικότερα την καλύτερη λειτουργία της. Μια καλή αρχιτεκτονική επιτρέπει στην προσθήκη και στην αφαίρεση χαρακτηριστικών χωρίς να δημιουργούνται conflicts, την συντήρηση της εφαρμογής, και την εύκολη παρακολούθηση της. Η παρακάτω προτεινόμενη αρχιτεκτονική εκφράζει μια γενικότερη ιδέα για την δομή των επιπέδων μιας εφαρμογής και εστιάζει στην αρχή του διαχωρισμού των διεργασιών, δηλαδή ότι κάθε επίπεδο θα πρέπει να έχει το δικό του μοντέλο δεδομένων.



Εικόνα 2.14 Διάγραμμα default αρχιτεκτονικής Android

- Ο ρόλος του UI επιπέδου είναι να εμφανίζει τα αποτελέσματα της εφαρμογής στην οθόνη του χρήστη. Είναι το κεντρικό σημείο αλληλεπίδρασης του χρήστη με την εφαρμογή. Οι αλλαγές των δεδομένων που γίνονται κάθε στιγμή, είτε λόγω αλληλεπίδρασης του χρήστη, είτε αυτές προέρχονται από κάποια εξωτερική πηγή (Web server), αντικατοπτρίζονται κατάλληλα προς τον χρήστη. Επίσης στο επίπεδο αυτό τα δεδομένα μετατρέπονται σε μορφή που μπορεί να τα παρουσιάσει το UI και στην συνέχεια τα εμφανίζει. Το μεγαλύτερο μέρος του UI ορίζεται σε XML αρχεία τα οποία περιγράφουν τις ιδιότητες και τα χαρακτηριστικά των διαφόρων components που θα δημιουργηθούν στην οθόνη του χρήστη.
- Το επίπεδο Domain Layer περιέχει την σύνθετη η και την απλή επιχειρηματική λογική της εφαρμογής και επικοινωνεί με το επίπεδο UI και Data. Το επίπεδο αυτό είναι προαιρετικό ως προς την υλοποίηση του καθώς εξαρτάται από τις απαιτήσεις της εφαρμογής.
- Το επίπεδο Data Layer περιέχει την επιχειρησιακή λογική. Μπορεί να περιέχει Repositories στα οποία γίνεται η δημιουργία, η αποθήκευση και η αλλαγή των δεδομένων.

2.2.2 Η δομή της εφαρμογής

Στην παρούσα εφαρμογή του Android μέρους εστιάζουμε στο UI επίπεδο χωρίς να δίνουμε έμφαση στα επίπεδα που χειρίζονται την επιχειρηματική λογική, καθώς η εφαρμογή στο Android αποτελεί κυρίως το σημείο αναπαράστασης των δεδομένων στον χρήστη αλλά και το σημείο αλληλεπίδρασης του χρήστη με τα δεδομένα αυτά. Ύστερα τα δεδομένα αυτά στέλνονται στο back end μέρος μέσω API calls όπου και υλοποιείται η επιχειρηματική λογική

Activities, Fragments

Τα Activities είναι τα κεντρικά σημεία στην εφαρμογή τα οποία παρέχουν μια οθόνη, ένα “παράθυρο” με λειτουργικότητα στον χρήστη για να αλληλεπιδράσει. Μια εφαρμογή μπορεί να περιέχει πολλά Activities τα οποία είτε εξυπηρετούν σε μεμονωμένες λειτουργίες είτε συνδέονται μεταξύ τους για την καλύτερη διεκπεραίωση των διάφορων λειτουργιών. Στην εφαρμογή μας ορίζουμε ένα αρχικό Activity το οποίο παρουσιάζεται στο χρήστη κατά την εκκίνηση της εφαρμογής και αυτό με την σειρά του καλεί άλλα activities για την εγγραφή ή την σύνδεση του χρήστη. Στη συνέχεια κατά την σύνδεση του χρήστη στην εφαρμογή, εμφανίζεται επιπλέον ένα Activity το οποίο λειτουργεί ως κεντρικό μενού για την εκκίνηση Fragments και την εναλλαγή μεταξύ αυτών. Τα Fragments θα μπορούσαν να θεωρηθούν ως sub Activities καθώς λειτουργούν σαν τα Activities, έχουν το δικό τους lifecycle, την δική τους διάταξη, συμπεριφορά και events. Τα Fragments δημιουργούνται από ένα Activity και είναι επαναχρησιμοποιήσιμα όσο αυτό εκτελείτε. Μπορούμε να τα χειριστούμε ανεξάρτητα, να προσθέσουμε καινούργια ή να αφαιρέσουμε. Επίσης υπάρχει η δυνατότητα προσθήκης των fragment σε στοίβα back stack καθώς εναλλασσόμαστε σε αυτά, ώστε να επιτρέπεται η επιστροφή στο fragment που εμφανίστηκε στην οθόνη την τελευταία φορά. Για παράδειγμα εάν από ένα fragment μεταβούμε σε ένα άλλο τότε με το back button δίνεται η δυνατότητα επιστροφής στο αρχικό. Στην παρούσα εφαρμογή το Activity που ενεργεί ως κεντρικό μενού προσθέτει και αφαιρεί από την οθόνη τα διάφορα Fragments ανάλογα με τις λειτουργίες που επιλέγει ο χρήστης.

Επικοινωνία με το API του web sever

Η εφαρμογή για την υλοποίηση των διάφορων σύνθετων λειτουργιών ή για την επεξεργασία, την αναζήτηση και την αποθήκευση δεδομένων επικοινωνεί με το backend το οποίο περιέχει όλη την επιχειρηματική λογική. Αυτό επιτυγχάνεται μέσω της βιβλιοθήκης Retrofit η οποία στέλνει και δέχεται πληροφορία από τα διάφορα end points.

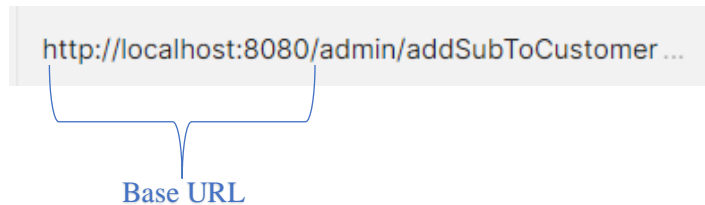
Retrofit

Πιο συγκεκριμένα η Retrofit οποία αποσκοπεί στη εύκολη και γρήγορη κατανάλωση RESTful υπηρεσιών ιστού σε μια Android εφαρμογή. Στην εφαρμογή μας χρησιμοποιείται η Retrofit 2 βιβλιοθήκη ώστε να μετατρέπει αυτόματα τα JSON requests που λαμβάνει από τον εξυπηρετητή σε POJO(Plain Old Java Object). Πρόκειται δηλαδή για τον συνδετικό κρίκο ανάμεσα στον web server μας και την εφαρμογή στο Android. Η retrofit αξιοποιεί το OkHttp ως επίπεδο δικτύωσης και βασίζεται σε αυτό. Το OkHttp είναι μια third party βιβλιοθήκη που είναι υπεύθυνη για την αποστολή και την λήψη αιτημάτων HTTP. Παρέχει μια υλοποίηση των interfaces HttpURLConnection και Apache Client που έχει ενσωματωμένα το Android, δουλεύοντας απευθείας πάνω στο Java Socket χωρίς να χρησιμοποιεί επιπλέον dependencies. Ορισμένα από τα πλεονεκτήματά του είναι η υποστήριξη σύγχρονων και ασύγχρονων κλήσεων, προσωρινή αποθήκευση δεδομένων, connection pooling.

Η υλοποίηση του retrofit

Για την επίτευξη της επικοινωνίας με τον Web Server υλοποιούνται τα εξής:

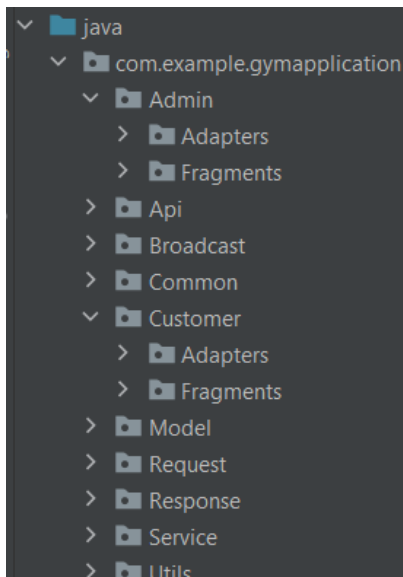
- Κλάσεις POJO οι οποίες θα χρησιμοποιηθούν για τον μετασχηματισμό των JSON δεδομένων σε αντικείμενα Java.
- Interfaces τα οποία διαθέτουν μεθόδους που εκτελούν αιτήματα HTTP όπως (Get, Post, Put, Delete κ.λπ.)
- Κλάση που περιέχει το Retrofit Builder instance το οποίο χρησιμοποιεί τα interfaces που περιεγράφηκαν και το Builder API το οποίο ορίζει το κεντρικό URL endpoint στο οποίο θα κατευθύνονται τα HTTP αιτήματα. Στα interfaces το URL path μπορεί να αλλάζει ανάλογα με την ιδιότητα του χρήστη και τις ενέργειες που θέλουμε να εκτελέσουμε



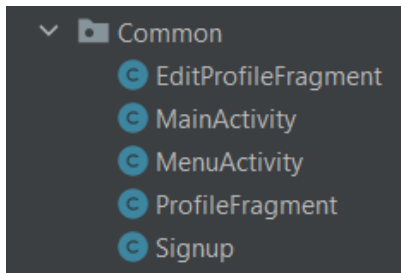
Εικόνα 2.15 Παράδειγμα HTTP request

2.2.3 Παρουσίαση των packages

Για την καλύτερη κατανόηση της δομής της εφαρμογής θα παρουσιαστεί η ιεραρχία των packages εντός του project καθώς και τα components που αυτά περιέχουν.

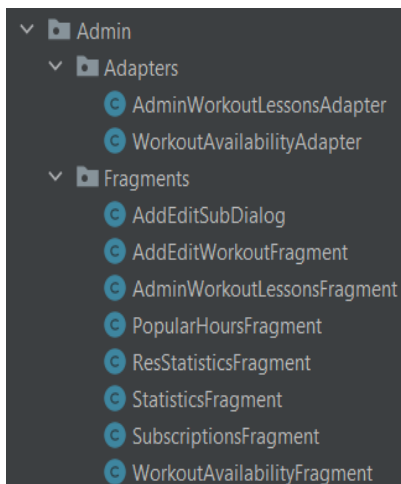


Εικόνα 2.16 Η ιεραρχία των packages



Εικόνα 2.17 Common package

Στο Common package (Εικόνα 2.17) βρίσκονται τα activities και fragments τα οποία είναι κοινά για τους χρήστες ανεξαρτήτως ρόλου. Για παράδειγμα τα Profile Fragment και Edit Profile Fragment παρέχουν λειτουργίες όπως προβολή προφίλ χρήστη και επεξεργασία των στοιχείων του. Το Main Activity περιέχει την αρχική οθόνη που εμφανίζεται κατά την εκκίνηση της εφαρμογής μαζί με την δυνατότητα login. Το Menu Activity είναι αυτό που περιέχει το Navigation View ένα μενού στο οποίο οι χρήστες ανάλογα με τον ρόλο τους μπορούν να μεταβούν σε άλλα Fragments. Το signup activity εμφανίζεται και στους διαχειριστές καθώς βρίσκεται στην αρχική οθόνη αλλά χρησιμοποιείται για εγγραφή μόνο πελατών.

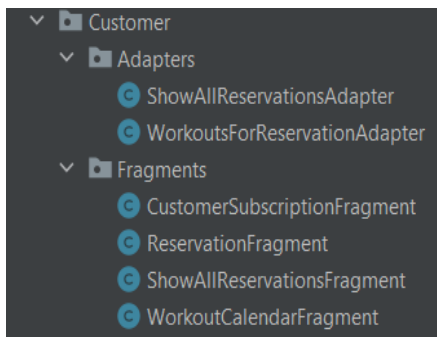


Εικόνα 2.18 Admin package

Το Admin package (Εικόνα 2.18) περιλαμβάνει τους Adapters και τα Fragments που κατέχουν πληροφορία τις λειτουργίες για έναν Admin. Οι Adapters κάνουν extend τον RecyclerViewAdapter ο οποίος διευκολύνει την εμφάνιση μεγάλου όγκου δεδομένων σε λίστες με δυναμικό τρόπο. Επαναχρησιμοποιεί τα views των αντικειμένων που αποκρύπτονται από την οθόνη, καθώς ο χρήστης κάνει scroll την λίστα των αντικειμένων, για να εμφανίσει τα υπόλοιπα στοιχεία που υπάρχουν στην λίστα, βελτιώνοντας σημαντικά έτσι την απόδοση της εφαρμογής. Η κλάση AdminWorkoutLessonsAdapter συνεπώς κρατάει την λίστα των μαθημάτων και η κλάση WorkoutAvailabilityAdapter την λίστα με τις μέρες και τις ώρες που είναι διαθέσιμο ένα πρόγραμμα. Παρακάτω δίνεται μια σύντομη περιγραφή για τα fragments

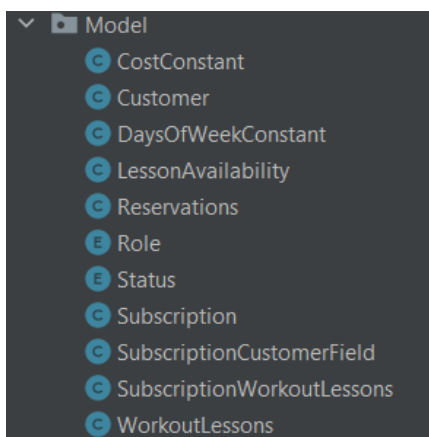
- **AddEditSubDialog:** Αφορά λειτουργίες προσθήκης νέας συνδρομής ή ανανέωση υπάρχουσας.

- **AddEditWorkoutFragment:** Αφορά λειτουργίες προσθήκης και επεξεργασίας μαθήματος.
- **AdminWorkoutLessonsFragment:** Περιλαμβάνει το RecyclerView στο οποίο ο AdminWorkoutLessonAdapter θα φορτώσει την λίστα με τα υπάρχοντα μαθήματα.
- **PopularHoursFragment:** Εμφανίζει στατιστικά σχετικά με βάση τις δημοφιλέστερες ώρες κράτησης.
- **ResStatisticsFragment:** Εμφανίζει στατιστικά για τις κρατήσεις των μαθημάτων.
- **StatisticsFragment:** Εμφανίζει στατιστικά για τις συνδρομές.
- **SubscriptionsFragment:** Υλοποιεί τη λειτουργία της αναζήτησης της συνδρομής ενός πελάτη και της προβολής της.
- **WorkoutAvailabilityFragment:** Περιλαμβάνει το RecyclerView όπου περιέχει την λίστα που φορτώνει ο WorkoutAvailabilityAdapter.



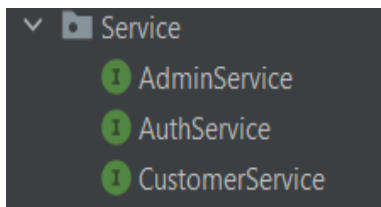
Εικόνα 2.19 Customer package

Αντίστοιχα με το Admin package στο Customer package (Εικόνα 2.19) περιλαμβάνονται οι Adapters και τα Fragments του Customer. Το ShowAllReservationsAdapter συνδέεται με το ShowAllReservationsFragment και αποσκοπούν στην εμφάνιση των κρατήσεων του χρήστη ενώ το WorkoutsForReservationAdapter με το WorkoutCalendarFragment εμφανίζουν τις διαθέσιμες ώρες και μέρες του κάθε μαθήματος με σκοπό την δημιουργία κράτησης στο ReservationFragment. Τέλος το CustomerSubscriptionFragment αφορά την προβολή της συνδρομής του χρήστη.



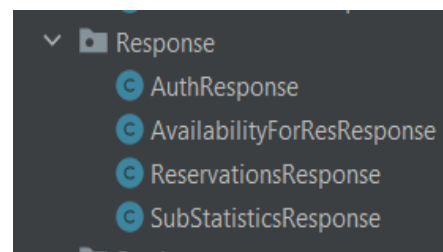
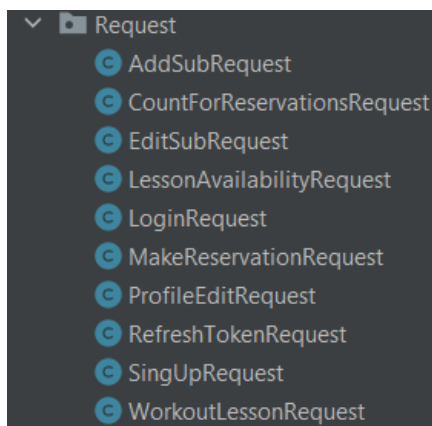
Εικόνα 2.20 Model package

Το Model package (Εικόνα 2.20) περιέχει όλες τις οντότητες (Entities) της εφαρμογής όπως και στο backend μέρος καθώς και κάποια χρήσιμα Enums και Constants.



Εικόνα 2.21 Service package

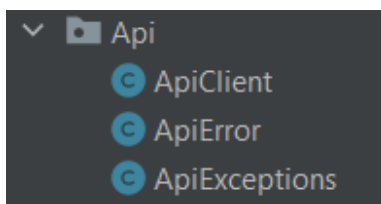
Το Service package (Εικόνα 2.21) περιλαμβάνει τα interfaces που ορίζουν τα endpoint από τα οποία γίνονται τα διάφορα HTTP Get, Post, Put, Delete requests στο API. Το Admin και Customer Service περιλαμβάνουν ξεχωριστά requests του Admin και του Customer αντίστοιχα ενώ το AuthService αφορά τα requests για την ανανέωση του JWT token του χρήστη την εγγραφή, την σύνδεση, και την αποσύνδεση του από την εφαρμογή.



Εικόνα 2.23 Response package

Εικόνα 2.22 Request package

Στα Request Response packages (Εικόνες 2.22, 2.23) βρίσκονται κλάσεις από τις οποίες δημιουργούνται DTO αντικείμενα είτε κατά την δημιουργία ενός request από το API η κατά την δημιουργία response προς το API.

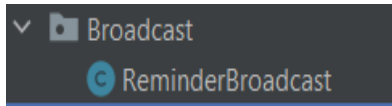


Εικόνα 2.24 Api package

Το Api package (Εικόνα 2.24) αποτελείται από τα εξής:

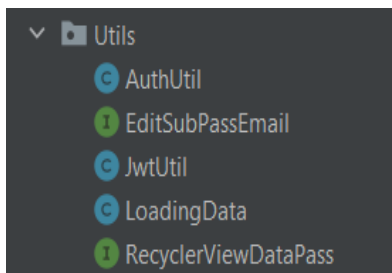
- **ApiClient:** Πρόκειται για την κλάση που καθορίζει το κεντρικό URL στο οποίο θα στέλνονται τα requests.
- **ApiError:** Περιέχει μια μεταβλητή message όπου θα παίρνει το μήνυμα σφάλματος που μπορεί να επιστρέψει ένα request στο API.

- **ApiExceptions:** Περιλαμβάνει τις κωδικές ονομασίες σε static final strings, από όλα τα πιθανά exceptions που μπορεί να δημιουργηθούν από κάποιο request στο API, ώστε να εμφανιστεί το κατάλληλο μήνυμα σφάλματος στον χρήστη.



Εικόνα 2.25 Broadcast package

Στο Broadcast package (Εικόνα 2.25) η κλάση ReminderBroadcast κάνει extend την BroadcastReceiver και επιτρέπει την δημιουργία ειδοποιήσεων κατά το runtime της εφαρμογής. Στην παρούσα εφαρμογή ο πελάτης λαμβάνει μια ειδοποίηση όταν πλησιάσει η ώρα που ξεκινάει το μάθημα του.



Εικόνα 2.26 Utils package

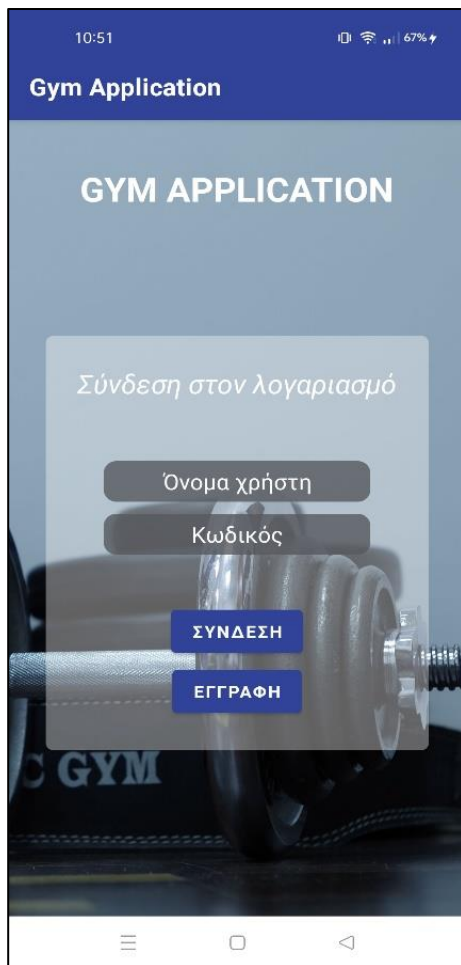
Το Utils package (Εικόνα 2.26) αποτελείται από τα εξής:

- **AuthUtil:** Η κλάση αυτή κρατάει τις πληροφορίες σχετικά με τη αυθεντικοποίηση του χρήστη. Αποθηκεύει ιδιωτικά, σε SharedPreferences τα στοιχεία (Username, Role, JWT token, Refresh Token) έτσι ώστε να μπορούν να αξιοποιηθούν σε διάφορες δραστηριότητες εντός της εφαρμογής. Κατά την αποσύνδεση του χρήστη από την εφαρμογή αυτά τα στοιχεία διαγράφονται.
- **JwtUtil:** Περιέχει λειτουργίες σχετικά με τον έλεγχο και την ανανέωση του JWT token του χρήστη.
- **EditSubPassEmail, RecyclerViewDataPass:** Τα interfaces περιέχουν κάποιες μεθόδους που αποσκοπούν στην μεταφορά ορισμένης πληροφορίας μεταξύ components.
- **LoadingData:** Διαχειρίζεται την εμφάνιση ενός Progress Bar κατά την διαδικασία της επικοινωνίας του Android με το API μέχρι και την φόρτωση των δεδομένων στην οθόνη του.

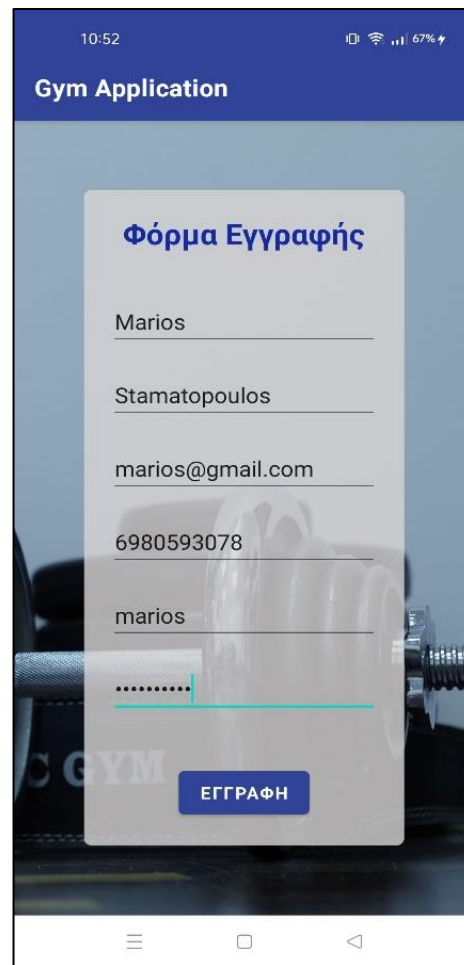
Κεφάλαιο 3^ο

3 Παρουσίαση εφαρμογής

Κατά την εκκίνηση της εφαρμογής εμφανίζεται στον χρήστη η φόρμα σύνδεσης (**Εικόνα 3.1**) στην οποία πρέπει να συμπληρώσει τα στοιχεία όνομα χρήστη και κωδικό πρόσβασης ώστε να του επιτραπεί η είσοδος στο προφίλ του. Εάν δεν είναι εγγεγραμμένος πατώντας το κουμπί Εγγραφή μεταφέρεται στην φόρμα εγγραφής (**Εικόνα 3.2**) όπου καλείται να εισάγει τα προσωπικά του στοιχεία για να εγγραφεί στο σύστημα ως πελάτης. Εφόσον συμπληρώσει επιτυχώς τα στοιχεία του και γίνουν οι απαραίτητοι έλεγχοι ότι δεν υπάρχει χρήστης με ίδια στοιχεία username και email στην βάση, πατώντας το κουμπί register θα οδηγηθεί ξανά στην φόρμα σύνδεσης με ένα μήνυμα επιτυχίας. Ο Διαχειριστής βρίσκεται εγγεγραμμένος στην βάση εξαρχής επομένως μπορεί να αποκτήσει πρόσβαση στην εφαρμογή πληκτρολογώντας τα στοιχεία του στην φόρμα σύνδεσης.



Εικόνα 3.1 Φόρμα Σύνδεσης



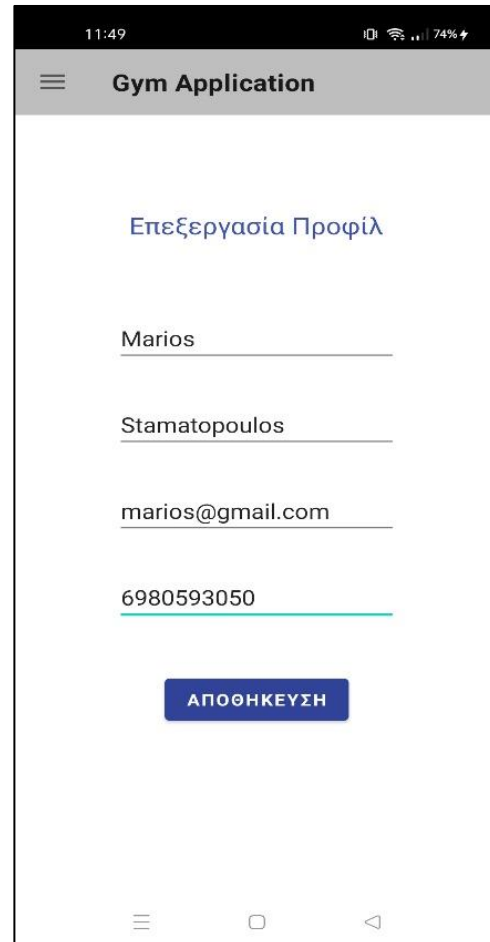
Εικόνα 3.2 Φόρμα Εγγραφής

Λειτουργίες Πελάτη

Κατά την επιτυχή σύνδεση του Πελάτη στην εφαρμογή εμφανίζεται το προφίλ του (**Εικόνα 3.3**), μέσω του οποίου μπορεί να επεξεργαστεί τα προσωπικά του στοιχεία πατώντας το αντίστοιχο κουμπί. Στην συνέχεια αφού εμφανιστεί η οθόνη με τα στοιχεία του τα οποία μπορεί να επεξεργαστεί πατώντας το κουμπί αποθήκευση αποθηκεύονται στη βάση η αλλαγές (**Εικόνα 3.4**). Επιπλέον δίνεται η δυνατότητα αλλαγής εικόνας προφίλ επιλέγοντας από τον εσωτερικό χώρο του κινητού, και αποθήκευσής της (**Εικόνες 3.5, 3.6**).



Εικόνα 3.3 Προφίλ



Εικόνα 3.4 Επεξεργασία προφίλ

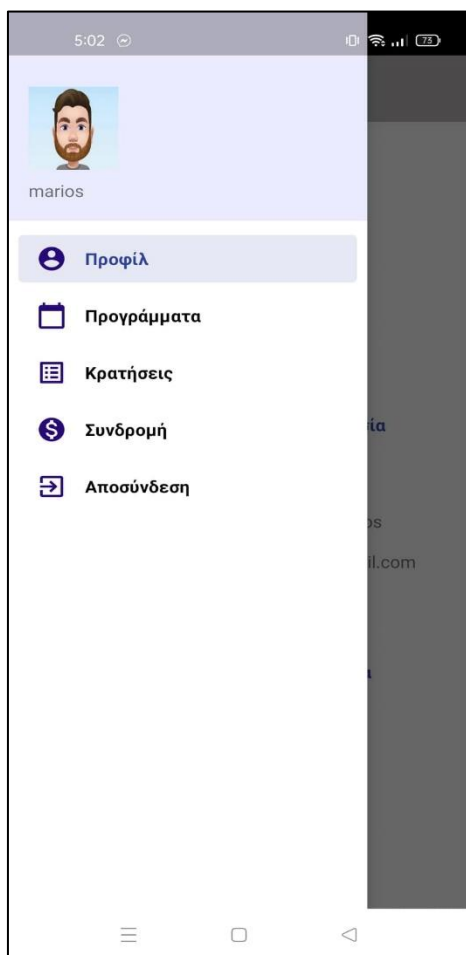


Εικόνα 3.5 Επεξεργασία εικόνας



Εικόνα 3.6 Αποθηκευμένη εικόνα

Ο χρήστης πατώντας στις τρεις γραμμές πάνω αριστερά στην οθόνη εμφανίζεται μια καρτέλα μενού με τις διαθέσιμες επιλογές (Εικόνα 3.7). Επιλέγοντας τα προγράμματα, παρουσιάζεται στην οθόνη ένα εβδομαδιαίο ημερολόγιο που έχει σκιαγραφημένη την τωρινή μέρα καθώς και μια λίστα με όλα τα διαθέσιμα προγράμματα της επιλεγμένης μέρας (Εικόνα 3.8). Ο χρήστης μπορεί εύκολα πατώντας πάνω στα νούμερα που αντιπροσωπεύουν την ημέρα να εξερευνήσει τα προγράμματα άλλων ημερών και με τα βελάκια να πλοηγηθεί σε επόμενα ή προηγούμενα εβδομαδιαία προγράμματα. Για να προβεί στην κράτηση προϋποθέτουμε ότι ο χρήστης διαλέγει κάποιο πρόγραμμα που δεν έχει τελειώσει και βρίσκεται εντός της τωρινής του μέρας (Εικόνες 3.9, 3.10). Πατώντας πάνω στο μπλε πλαίσιο των μαθημάτων, όπου αναγράφει το όνομα και την ώρα διαθεσιμότητας του προγράμματος, ο χρήστης μεταφέρεται στην σελίδα για την πραγματοποίηση της κράτησης του (Εικόνα 3.11).



Εικόνα 3.7 Μενού επιλογών



Εικόνα 3.8 Εμφάνιση διαθέσιμων προγραμμάτων

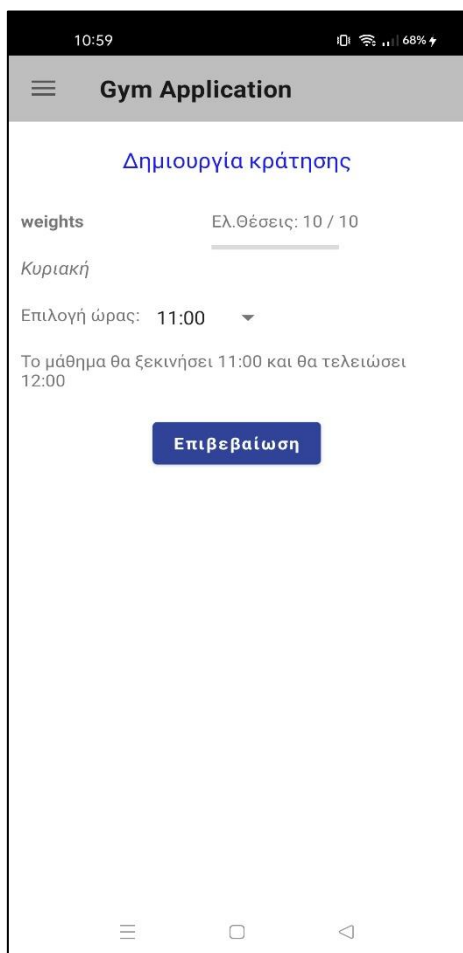


Εικόνα 3.9 Επιλογή μαθήματος επόμενης μέρας

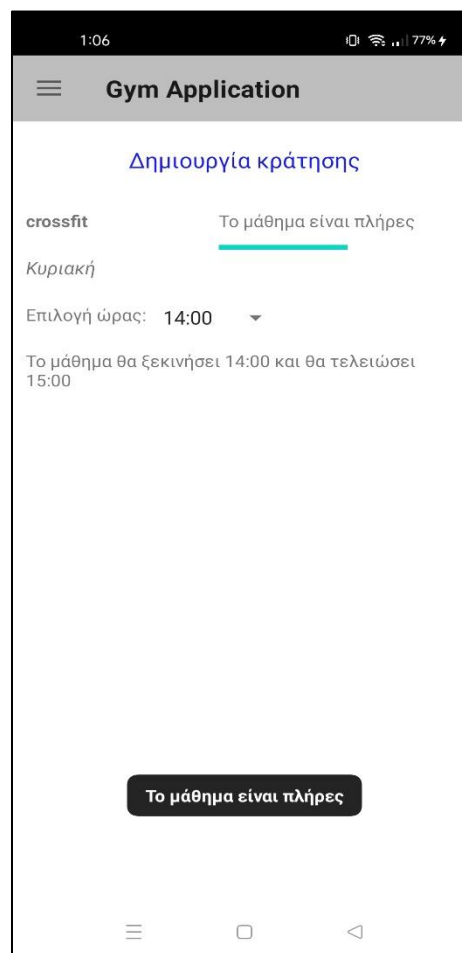


Εικόνα 3.10 Επιλογή μαθήματος προηγούμενης ημέρας

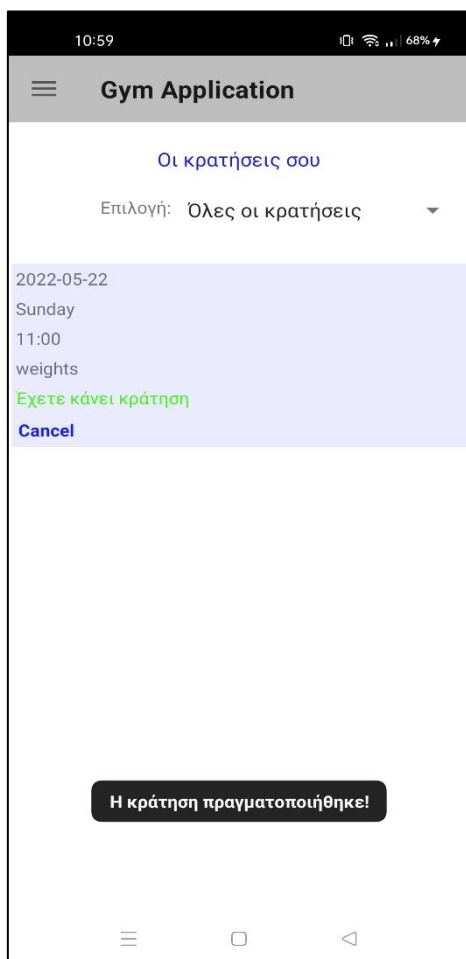
Κατά την δημιουργία κράτησης εμφανίζονται αναλυτικά τα στοιχεία του προγράμματος που έχει επιλέξει για κράτηση και τις θέσεις διαθεσιμότητας που υπάρχουν. Ο χρήστης μπορεί να επιλέξει την ώρα που επιθυμεί να κάνει κράτηση μέσα από μια dropdown λίστα. Εάν το μάθημα είναι πλήρες για εκείνη την ώρα τότε εμφανίζεται αντίστοιχο μήνυμα και αποτρέπει τον χρήστη από το να πραγματοποιήσει την κράτηση (Εικόνα 3.12). Εφόσον πραγματοποιηθεί επιτυχώς η κράτηση εμφανίζεται μήνυμα επιτυχίας και μεταβαίνει στην σελίδα με τις κρατήσεις όπου εμφανίζονται όλες οι κρατήσεις παλαιότερες και τωρινές (Εικόνα 3.13). Στην σελίδα αυτή μπορεί να πάρει κανείς πρόσβαση και από το μενού επιλογών. Επιπλέον εμφανίζεται ειδοποίηση στο κινητό του χρήστη μερικά λεπτά πριν την αρχή του μαθήματος ώστε να υπενθυμίσει στον πελάτη την ώρα που ξεκινάει το μάθημα (Εικόνα 3.14).



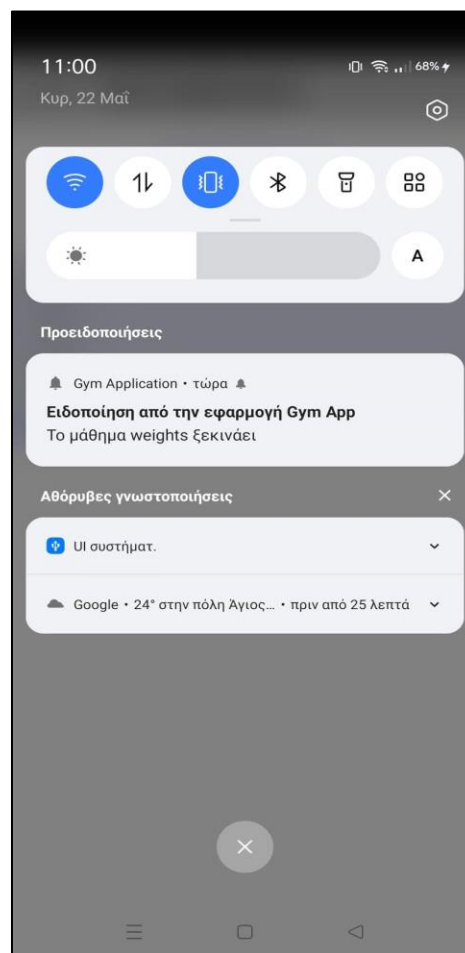
Εικόνα 3.11 Δημιουργία Κράτησης



Εικόνα 3.12 Μάθημα πλήρες για την επιλεγμένη ώρα



Εικόνα 3.13 Επιτυχής κράτηση



Εικόνα 3.14 Ειδοποίηση αρχής μαθήματος

Τέλος δίνεται η δυνατότητα στον πελάτη να εμφανιστεί η συνδρομή του με τα μαθήματα στα οποία έχει εγγραφεί (Εικόνα 3.15). Εάν κάποιο από τα μαθήματα έχει λήξει ο χρήστης ενημερώνεται κατά την σύνδεση του στην εφαρμογή με μήνυμα και στην συνδρομή του εμφανίζεται κάτω από το αντίστοιχο μάθημα το προειδοποιητικό μήνυμα ότι έχει λήξει (Εικόνες 3.16, 3.17). Η ανανέωσή της γίνεται από την μεριά του Διαχειριστή και παρουσιάζεται παρακάτω.



Εικόνα 3.15 Η συνδρομή του πελάτη

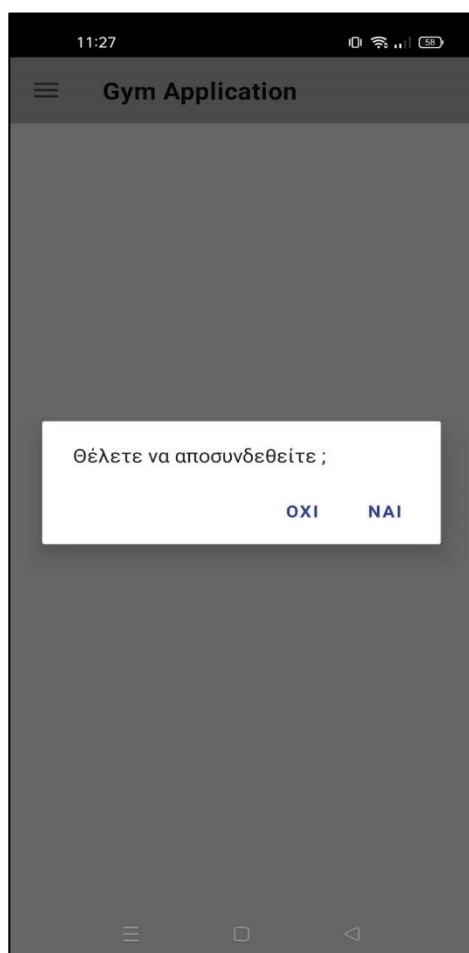


Εικόνα 3.16 Μήνυμα λήξης προγράμματος κατά την σύνδεση



Εικόνα 3.17 Εμφάνιση του μαθήματος που έχει λήξει η συνδρομή

Κατά την αποσύνδεση οποιουδήποτε χρήστη από την εφαρμογή χάνεται η ταυτοποίηση του και ανακατευθύνεται στην αρχική φόρμα σύνδεσης όπου καλείται εισάγει ξανά τα στοιχεία του (Εικόνα 3.18). Εάν όμως πραγματοποιηθεί έξοδος από την εφαρμογή χωρίς να έχει γίνει αποσύνδεση, ο χρήστης παραμένει συνδεδεμένος και την επόμενη φορά που θα ανοίξει την εφαρμογή θα επανασυνδεθεί (Εικόνες 3.19, 3.20).



Εικόνα 3.18 Αποσύνδεση



Εικόνα 3.19 Έξοδος από την εφαρμογή πατώντας το back button



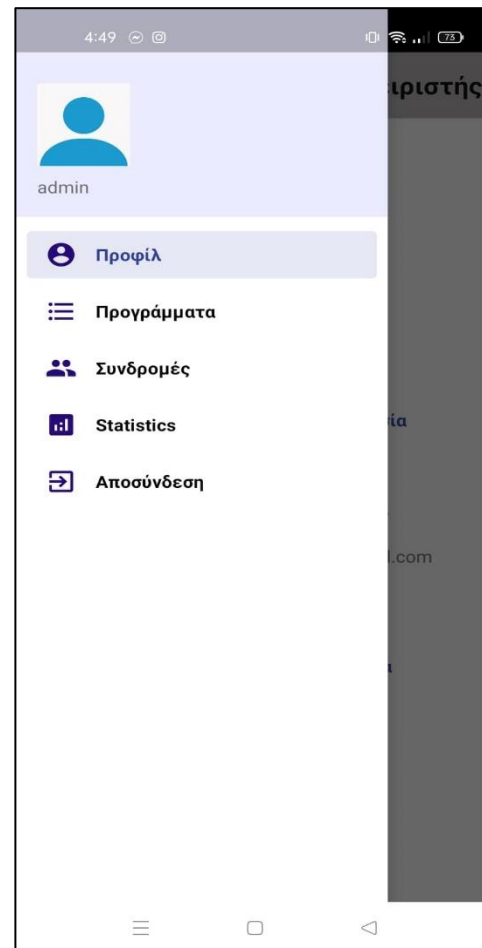
Εικόνα 3.20 Άνοιγμα εφαρμογής και επανασύνδεση

Λειτουργίες Διαχειριστή

Ομοίως κατά την σύνδεση του διαχειριστή στην εφαρμογή γίνεται ανακατεύθυνση στο προφίλ του, όπου μπορεί να πραγματοποιήσει επεξεργασία των στοιχείων και της εικόνας προφίλ του (**Εικόνα 3.21**). Στην συνέχεια ανοίγοντας το μενού εμφανίζονται οι διαθέσιμες επιλογές του (**Εικόνα 3.22**).

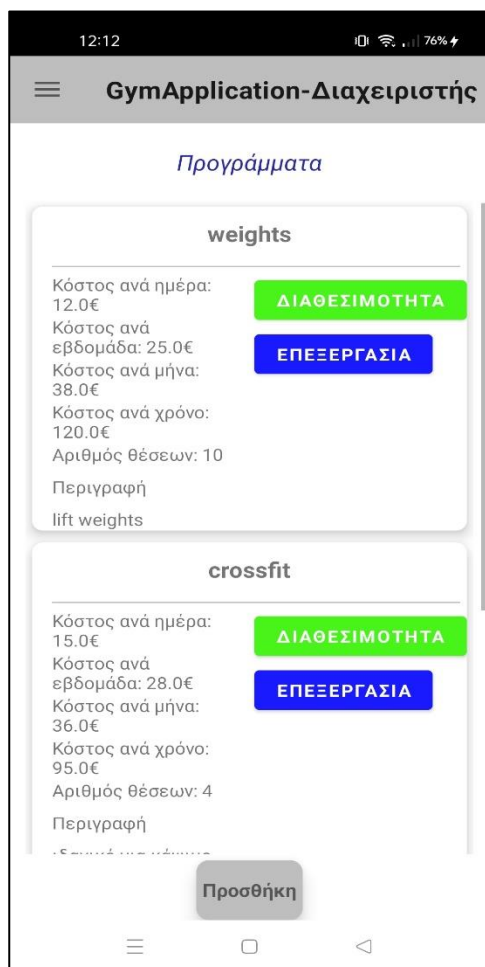


Εικόνα 3.21 Προφίλ Διαχειριστή



Εικόνα 3.22 Μενού επιλογών Διαχειριστή

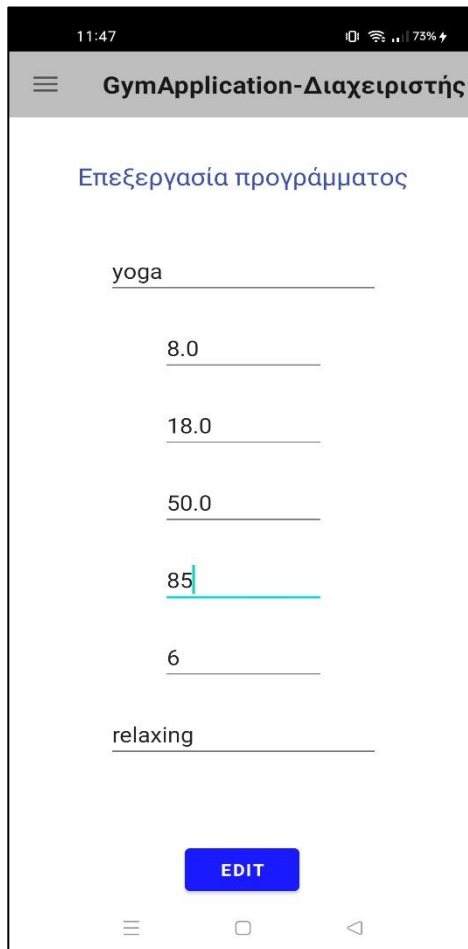
Η επιλογή προγράμματα εμφανίζει την λίστα με τα μαθήματα εφόσον αυτά υπάρχουν (**Εικόνα 3.23**). Με το κουμπί προσθήκη προβάλλεται μια φόρμα στη οποία ο διαχειριστής μπορεί να προσθέσει ένα νέο μάθημα (**Εικόνα 3.24**). Με την επιτυχή προσθήκη του μαθήματος επιστρέφει στην λίστα με τα μαθήματα. Σε κάθε μάθημα υπάρχει το κουμπί διαθεσιμότητα και επεξεργασία. Το κουμπί επεξεργασία προβάλλει μια φόρμα με τα στοιχεία του αντίστοιχου μαθήματος όπου επιτρέπει την δημιουργία αλλαγών και την αποθήκευσή τους (**Εικόνα 3.25**).



Εικόνα 3.23 Η λίστα με τα προγράμματα



Εικόνα 3.24 Φόρμα προσθήκης νέου προγράμματος



11:47 73%

GymApplication-Διαχειριστής

Επεξεργασία προγράμματος

yoga

8.0

18.0

50.0

85

6

relaxing

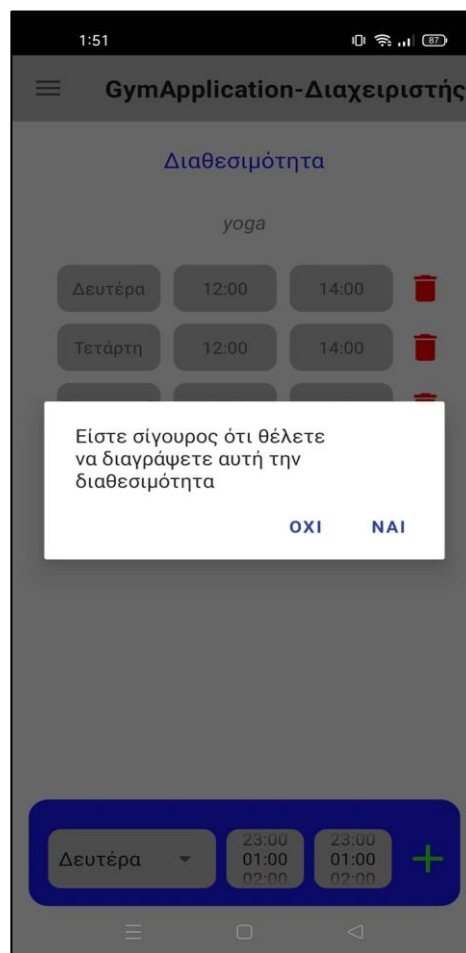
EDIT

Εικόνα 3.25 Επεξεργασία του κόστους ανά χρόνο για το πρόγραμμα yoga

Πατώντας το κουμπί διαθεσιμότητα εμφανίζεται μια σελίδα στην οποία ο διαχειριστής μπορεί να επιλέξει διαθέσιμη μέρα και ώρα αρχής και τέλους για το κάθε μάθημα (Εικόνα 3.26). Υποθέτοντας ότι τα μαθήματα διαρκούν μία ώρα, ο διαχειριστής εισάγει την περίοδο από την ώρα την οποία θα είναι διαθέσιμο για κράτηση μέχρι και την ώρα που λήξει. Μπορεί να προσθέσει διαθεσιμότητα και να αφαιρέσει κάποια πατώντας πάνω στα κατάλληλα εικονίδια (Εικόνες 3.27, 3.28). Ένα μάθημα μπορεί να έχει μόνο μία διαθεσιμότητα για μια συγκεκριμένη μέρα. (Εικόνα 3.29).



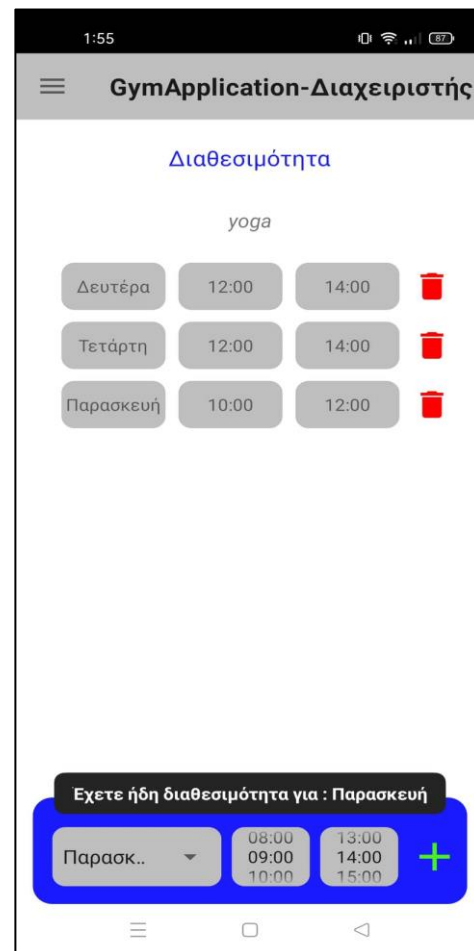
Εικόνα 3.26 Σελίδα διαθεσιμότητας για το πρόγραμμα yoga



Εικόνα 3.27 Διαγραφή διαθεσιμότητας με ημέρα Σάββατο και ώρα 17:00 – 20:00

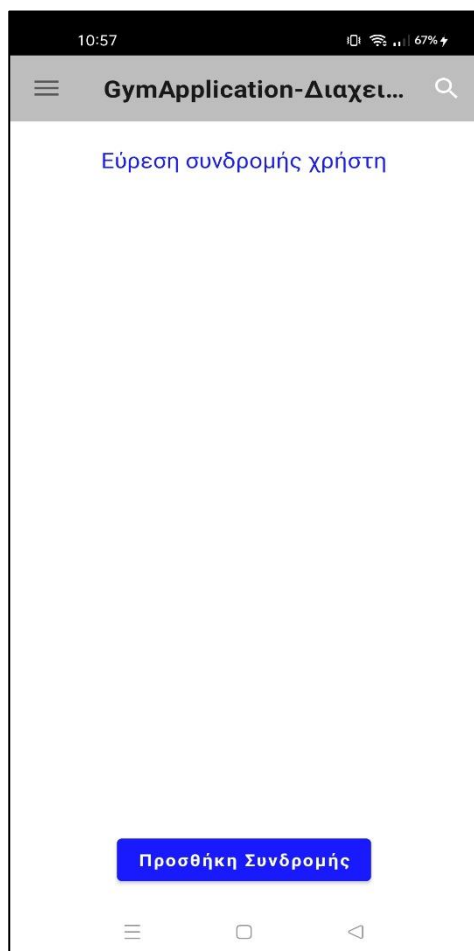


Εικόνα 3.28 Επιτυχής διαγραφή διαθεσιμότητας

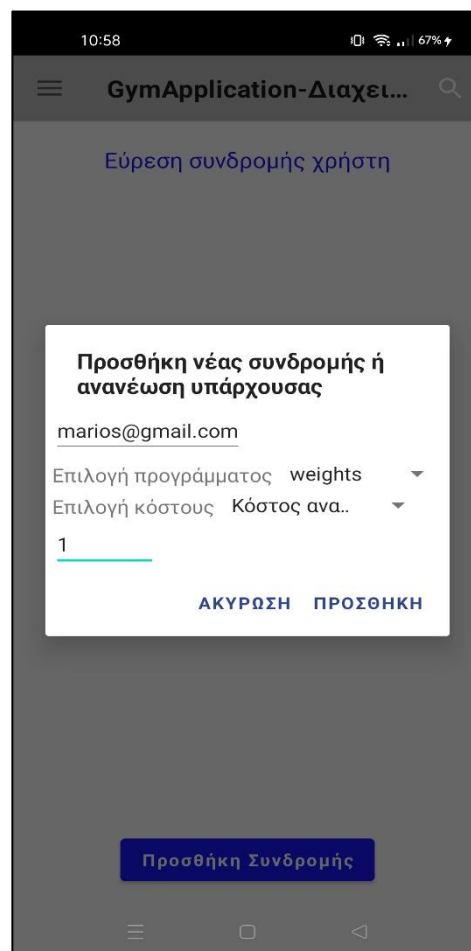


Εικόνα 3.29 Προσπάθεια εισαγωγής υπάρχουσας διαθεσιμότητας

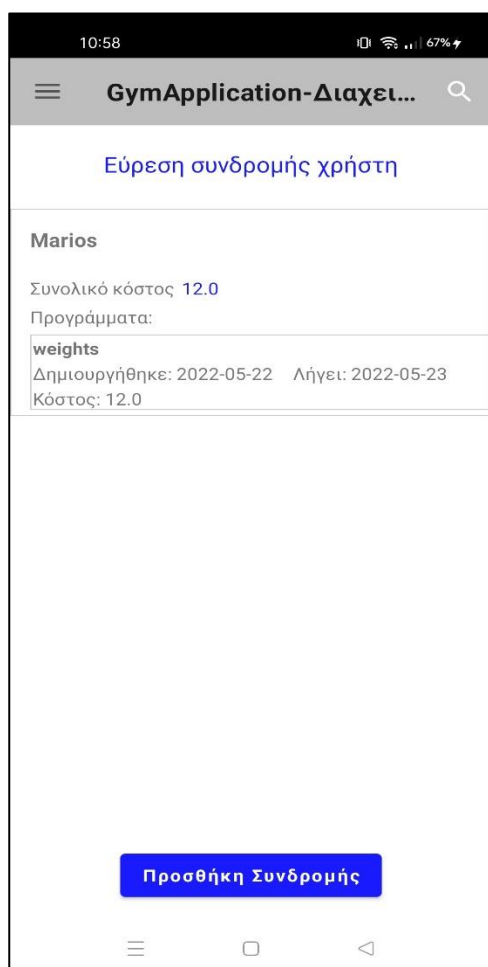
Η επιλογή συνδρομές από το μενού επιλογών, δίνει στον διαχειριστή τις λειτουργίες της προσθήκης της ανανέωσης και της εύρεσης συνδρομής πελάτη (Εικόνα 3.30). Πατώντας το κουμπί προσθήκη εμφανίζεται ένα Pop up παράθυρο με το οποίο γίνεται η προσθήκη μαθήματος στην συνδρομή του πελάτη. Στο πρώτο πεδίο συμπληρώνεται το email του πελάτη, ύστερα επιλέγεται κάποιο μάθημα από την λίστα μαθημάτων, το κόστος του μαθήματος από τις επιλογές (Κόστος ανά ημέρα, κόστος ανά εβδομάδα, κόστος ανά μήνα, κόστος ανά χρόνο) και την ποσότητα (Εικόνες 3.31, 3.32). Για παράδειγμα η επιλογή κόστος ανά ημέρα με ποσότητα 2, σημαίνει ότι ο πελάτης θα πληρώσει ημερήσια συνδρομή για 2 μέρες. Η αναζήτηση μια συνδρομής γίνεται στο πάνω μέρος την σελίδας πληκτρολογώντας το email του πελάτη στο πλαίσιο και πατώντας το εικονίδιο για αναζήτηση. Στην συνέχεια εμφανίζονται πληροφορίες για την συνδρομή του και την δυνατότητα ενημέρωσης κάποιου μαθήματος που έχει λήξει (Εικόνα 3.33). Η διαδικασία ανανέωσης της συνδρομής είναι ίδια με αυτή της προσθήκης.



Εικόνα 3.30 Σελίδα εύρεσης/προσθήκης/ανανέωσης συνδρομής



Εικόνα 3.31 Προσθήκη συνδρομής στον πελάτη

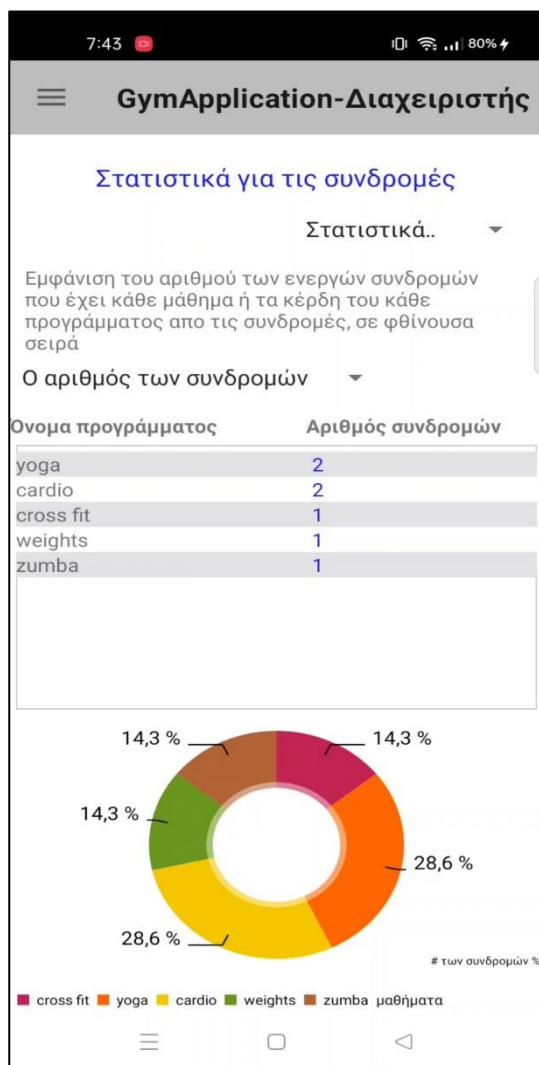


Εικόνα 3.32 Επιτυχής προσθήκης προγράμματος στην συνδρομή του χρήστη

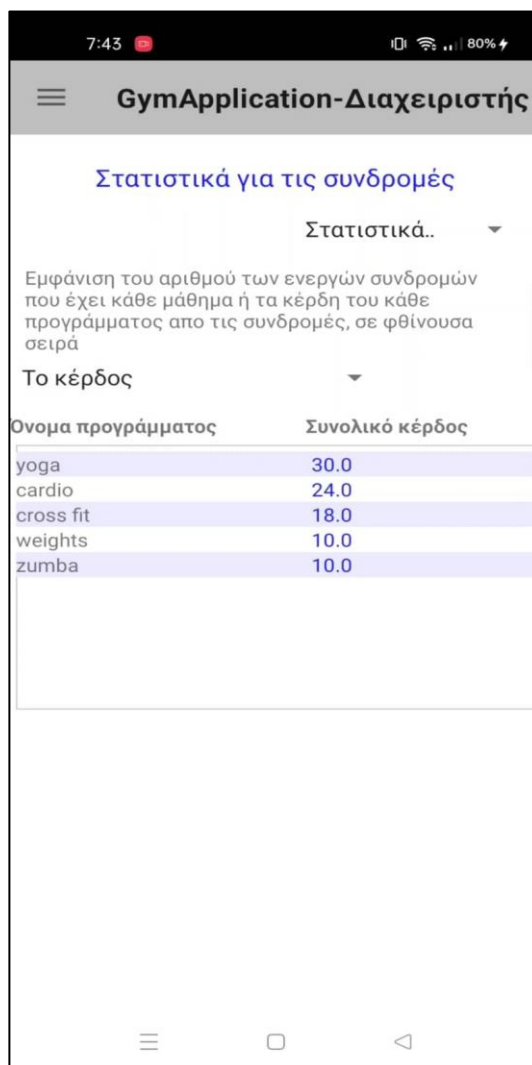


Εικόνα 3.33 Εύρεση συνδρομής χρήστη και εμφάνιση λήξης προγράμματος

Με την επιλογή στατιστικά ο χρήστης μπορεί να περιηγηθεί στα διάφορα στατιστικά της εφαρμογής σχετικά με τις συνδρομές τις κρατήσεις και τις ώρες κράτησης μαθημάτων. Αρχικά εμφανίζεται μία λίστα με τον αριθμό των ενεργών συνδρομών για κάθε μάθημα ταξινομημένο με φθίνουσα σειρά, καθώς και ένα διάγραμμα πίτας όπου εμφανίζει σε ποσοστιαία τιμή τον αριθμό των συνδρομών (Εικόνα 3.34). Επίσης επιλέγοντας το κέρδος από την dropdown λίστα φαίνονται ταξινομημένα κατά φθίνουσα σειρά τα κέρδη από τις συνδρομές του κάθε μαθήματος (Εικόνα 3.35). Επιλέγοντας τα στατιστικά για της κρατήσεις, προκύπτουν για κάθε μάθημα ο αριθμός των κρατήσεων που πραγματοποιήθηκαν την συγκεκριμένη μέρα ή μήνα, και παρουσιάζονται μέσω ενός ραβδογράμματος (Εικόνες 3.36, 3.37). Επιλέγοντας τις δημοφιλείς ώρες κράτησης παρουσιάζονται συνολικά οι πραγματοποιημένες κρατήσεις, κάθε ώρα για οποιοδήποτε μάθημα, την συγκεκριμένη ημέρα που έχει επιλεγεί (Εικόνα 3.38).



Εικόνα 3.34 Στατιστικά για τις συνδρομές



Εικόνα 3.35 Το κέρδος κάθε μαθήματος από τις συνδρομές



Εικόνα 3.36 Στατιστικά για τις κρατήσεις μιας επιλεγμένης ημέρας



Εικόνα 3.37 Στατιστικά για τις κρατήσεις ενός επιλεγμένου μήνα



Εικόνα 3.38 Στατιστικά σχετικά με τις δημοφιλείς ώρες κράτησης μια επιλεγμένης ημέρας



Κεφάλαιο 4^ο

4 Συμπεράσματα και μελλοντικές επεκτάσεις

Η παρούσα εφαρμογή έρχεται να καλύψει τις βασικότερες λειτουργίες ενός διαχειριστή γυμναστηρίου και των αθλούμενων. Επίσης αποτελεί ένα πρακτικό εργαλείο για την αποφυγή συνωστισμού στους χώρους του γυμναστηρίου, καθώς μέσω των κρατήσεων ελέγχεται και οργανώνεται καλύτερα ο αριθμός των ατόμων που μπορούν να παρευρεθούν σε κάποιο μάθημα. Πλέον οι κινητές συσκευές βρίσκονται στην καθημερινή μας ζωή, επομένως η υλοποίηση της σε Android προσφέρει την δυνατότητα στους χρήστες μέσα από την ευκολία μιας οθόνης να πραγματοποιούν τις διάφορες λειτουργίες τους.

Η εφαρμογή αυτή θέτει τα θεμέλια για τον εμπλουτισμό και την επέκτασή της σε ένα ολοκληρωμένο σύστημα διαχείρισης γυμναστηρίου με περισσότερες δυνατότητες. Μια καλή επέκταση της εφαρμογής, θα ήταν η ανάπτυξη και μιας web εφαρμογής (ιστοσελίδας), παράλληλα με την εφαρμογή Android στην οποία οι χρήστες θα μπορούσαν να συνδεθούν και από άλλες συσκευές οι οποίες έχουν πρόσβαση στο διαδίκτυο. Στην web εφαρμογή οι διαχειριστές θα μπορούσαν να έχουν επιπλέον δυνατότητες και να προσφέρεται μια ευέλικτη αναπαράσταση της πληροφορίας, με σκοπό την καλύτερη παρακολούθηση και διαχείριση των συνδρομών και των κρατήσεων. Επιπλέον θα μπορούσε να υποστηρίζεται ένα ηλεκτρονικό σύστημα πληρωμών στο οποίο θα παρέχεται η δυνατότητα στους πελάτες να ανανεώνουν ηλεκτρονικά την συνδρομή τους. Τέλος η εφαρμογή θα μπορούσε υλοποιεί την διαδικασία του check in παράγοντας κάποιο barcode και κατά την είσοδο του πελάτη στο γυμναστήριο με κάποιο scanner να πραγματοποιούνταν η διαδικασία του check in και να ενημερωνόταν το σύστημα για την άφιξη του αθλούμενου στον χώρο του γυμναστηρίου.



5 Βιβλιογραφία

<https://spring.io/>

<https://www.javaguides.net/>

<https://www.baeldung.com/>

<https://www.javatpoint.com/>

<https://www.tutorialspoint.com/index.htm>

<https://jwt.io/>

<https://dzone.com/>

<https://en.wikipedia.org/>