



ΠΑΝΕΠΙΣΤΗΜΙΟ ΠΕΙΡΑΙΩΣ

UNIVERSITY OF PIRAEUS

ΠΑΝΕΠΙΣΤΗΜΙΟ ΠΕΙΡΑΙΩΣ - ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ:

*Ανάπτυξη εφαρμογής διαχείρισης
προόδου μαθημάτων σε Android και
ιστοσελίδας: UNIPi GRADES*

ΟΝΟΜΑΤΕΠΩΝΥΜΟ ΦΟΙΤΗΤΗ: ΚΩΝΣΤΑΝΤΙΝΟΣ-ΠΩΛ ΠΑΠΑΝΙΚΟΣ

ΑΡΙΘΜΟΣ ΜΗΤΡΩΟΥ :Π14142

ΕΠΙΒΛΕΠΩΝ ΚΑΘΗΓΗΤΗΣ : ΕΥΘΥΜΗΣ ΑΛΕΠΗΣ

Πειραιάς, Ιούνιος 2019



Περίληψη

Σκοπός της παρούσας πτυχιακής εργασίας είναι η υλοποίηση και η κατασκευή μιας mobile εφαρμογής σε MVVM μοντέλο χρησιμοποιώντας την βιβλιοθήκη android architecture components. Οι χρήστες έχουν την δυνατότητα να αποθηκεύουν βαθμούς και άλλα στοιχεία σχετικά με μαθήματα που παρακολουθούν . Αναπτύχθηκε επίσης μία ιστοσελίδα η οποία συγχρονίζεται με την εφαρμογή, χρησιμοποιώντας τα δεδομένα της, διαμορφώνει διαφορετικά στατιστικά στοιχεία που αφορούν τον χρήστη. Η εφαρμογή υλοποιήθηκε στο προγραμματιστικό περιβάλλον Android Studio. Καταγράφονται επίσης κενά στην αγορά καθώς και εφαρμογές με παράλληλες λειτουργίες οι οποίες λειτούργησαν και σαν μοντέλα για την παρούσα πτυχιακή εργασία.



Περιεχόμενα

1. Εισαγωγή	5
1.1 Κίνητρο διεξαγωγής εργασίας	5
1.2 Σκοπός και στόχοι τις εργασίας	5
1.3 Δομή εργασίας	6
2. Ανασκόπηση πεδίων	7
2.1 Grade Tracker Pro	7
2.2 My Grades	11
3. Παρουσίαση της εφαρμογής και ιστοσελίδας	14
3.1 Χρήση της εφαρμογής	14
3.2 Δραστηριότητα Login	14
3.3 Δραστηριότητα Register	17
3.4 Δραστηριότητα Home	18
3.5 Δραστηριότητα Add Note	20
3.6 Χρήση της ιστοσελίδας	21
3.7 Δραστηριότητα Login της ιστοσελίδας	22
3.8 Δραστηριότητα Home της ιστοσελίδας	22
3.9 Δραστηριότητα Search a user της ιστοσελίδας	24
3.10 Δραστηριότητα Search a course της ιστοσελίδας	25
4. Αρχιτεκτονική συστήματος	27
4.1 Ανάλυση αρχιτεκτονικής της εφαρμογής Unipi Grades	27
4.1.1 Εισαγωγή	27
4.1.2 Ο τρόπος λειτουργίας του MVVM	28



4.1.3 Η εφαρμογή του μοντέλου MVVM στο Unipi Grades	29
4.1.4 Ο τρόπος λειτουργίας του Android Jetpack Architecture components	33
4.1.5 Android Jetpack Architecture components στην εφαρμογή Unipi Grades	37
4.1.6 Διάγραμμα κλάσης	41
4.2 Τρόπος σύνδεσης εφαρμογής και ιστοσελίδας.....	42
4.3 Ανάλυση αρχιτεκτονικής της ιστοσελίδας Unipi Grades	43
4.3.1 Εισαγωγή.....	43
4.3.2 Λειτουργία login.....	43
4.3.3 Λειτουργία main	45
4.3.4 Λειτουργία search_user	47
4.3.5 Λειτουργία search_course	49
4.3.6 Use case Diagram	51
5. Εργαλεία και Τεχνολογίες	53
5.1 Γλώσσα Προγραμματισμού	53
5.2 Βάση Δεδομένων	54
5.3 Περιβάλλον	54
5.4 Plugins	55
6. Συμπεράσματα και Μελλοντικές επεκτάσεις.....	56
7. Βιβλιογραφία	57
8. Παράρτημα	58
8.1 Γλωσσάριο.....	58

1. Εισαγωγή

1.1 Κίνητρο διεξαγωγής εργασίας

Την εποχή που διανύουμε, τα έξυπνα κινητά ή smartphones είναι ιδιαίτερα δημοφιλή και αποτελούν αναπόσπαστο κομμάτι της καθημερινότητάς μας. Σε ότι αφορά τα λειτουργικά συστήματα των κινητών τηλεφώνων, το Android κατέχει εξέχουσα θέση μίας και από το 2013 κατέχει το μεγαλύτερο μερίδιο της αγοράς. Συνεπώς, η ανάπτυξη επιχειρηματικής δραστηριότητας στην αγορά των εφαρμογών για Android είναι πολλά υποσχόμενη.

1.2 Σκοπός και στόχοι τις εργασίας

Σκοπός της παρούσας εργασίας, είναι η δημιουργία μιας εφαρμογής αποθήκευσης, παρουσίασης και επεξεργασίας βαθμών ενός φοιτητή, ανεξαρτήτου σχολής. Η εφαρμογή έχει την δυνατότητα σύνδεσης με μια ιστοσελίδα, που παρουσιάζει χρήσιμα στατιστικά στοιχεία στον χρήστη. Η εφαρμογή είναι δομημένη βάση της αρχιτεκτονικής **Android Jetpack Components**, η οποία ευνοεί την σχεδίαση ισχυρών και διατηρήσιμων εφαρμογών με το επιπλέον πλεονέκτημα, την ικανότητα δηλαδή να δοκιμάζονται στα στάδια ανάπτυξης τους. Για τον συγχρονισμό της ιστοσελίδας με την εφαρμογή, χρησιμοποιήθηκε η βάση **Firestore**.

1.3 Δομή εργασίας

Κεφάλαιο 1- Εισαγωγή:

Στο πρώτο κεφάλαιο, γίνεται μια εισαγωγή για την εφαρμογή που υλοποιήθηκε καθώς και ποιες ανάγκες εξυπηρετεί ή πρόκειται να εξυπηρετήσει στο μέλλον.

Κεφάλαιο 2- Ανασκόπηση πεδίου:

Στο δεύτερο κεφάλαιο, γίνεται μια παρουσίαση και αναφορά σε παρόμοιες εφαρμογές που υπάρχουν στην Ελλάδα και στο εξωτερικό.

Κεφάλαιο 3- Παρουσίαση και χρήση της εφαρμογής και της ιστοσελίδας:

Στο τρίτο κεφάλαιο, το οποίο αποτελεί το εγχειρίδιο χρήσης της εφαρμογής και της ιστοσελίδας καθώς παρουσιάζονται επεξηγηματικά, με την χρήση εικόνων, οι λειτουργίες της εφαρμογής, γνωστό ως User Manual.

Κεφάλαιο 4- Αρχιτεκτονική συστήματος:

Στο τέταρτο κεφάλαιο, παρουσιάζεται η αρχιτεκτονική του συστήματος καθώς και κάποια σχεδιαγράμματα για την πλήρη κατανόηση του συστήματος. Επιπλέον γίνεται αναφορά στις τεχνολογίες που χρησιμοποιήθηκαν όπως η γλώσσα, η βάση καθώς και το πως φτιάχτηκε.

Κεφάλαιο 5- Εργαλεία και Τεχνολογίες:

Στο πέμπτο κεφάλαιο, γίνεται αναφορά σε όλες τις τεχνολογίες και τα εργαλεία που χρησιμοποιήσαμε όπως βάση δεδομένων, frameworks, προγραμματιστικό περιβάλλον, plugins. ISOSSS

Κεφάλαιο 6- Συμπεράσματα και μελλοντικές επεκτάσεις:

Τέλος στο έκτο κεφάλαιο έχουμε τα συμπεράσματα και τις μελλοντικές επεκτάσεις της εφαρμογής.

Κεφάλαιο 7- Βιβλιογραφία:

Αναφορά στις πηγές που χρησιμοποιήθηκαν για την εκπόνηση της εργασίας.

Κεφάλαιο 8- Παραρτήματα:

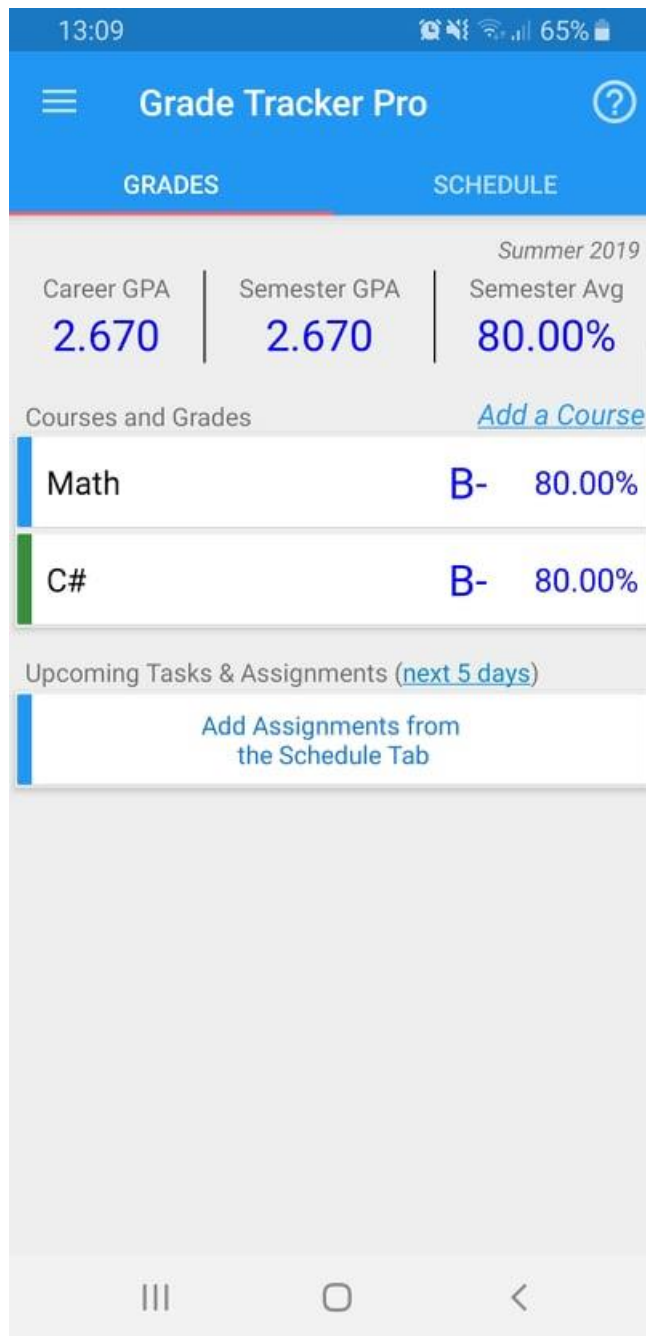
Περιέχει τα Γλωσσάριο.

2. Ανασκόπηση πεδίων

Οι εφαρμογές στις οποίες υπάρχει η δυνατότητα καταγραφής βαθμών , είναι ευρέως διαδεδομένες, καθώς πολλές φορές θέλουμε να έχουμε μια καλύτερη εικόνα στην απόδοση μας και διάφορα στατιστικά που προκύπτουν από αυτά τα δεδομένα.

2.1 Grade Tracker Pro

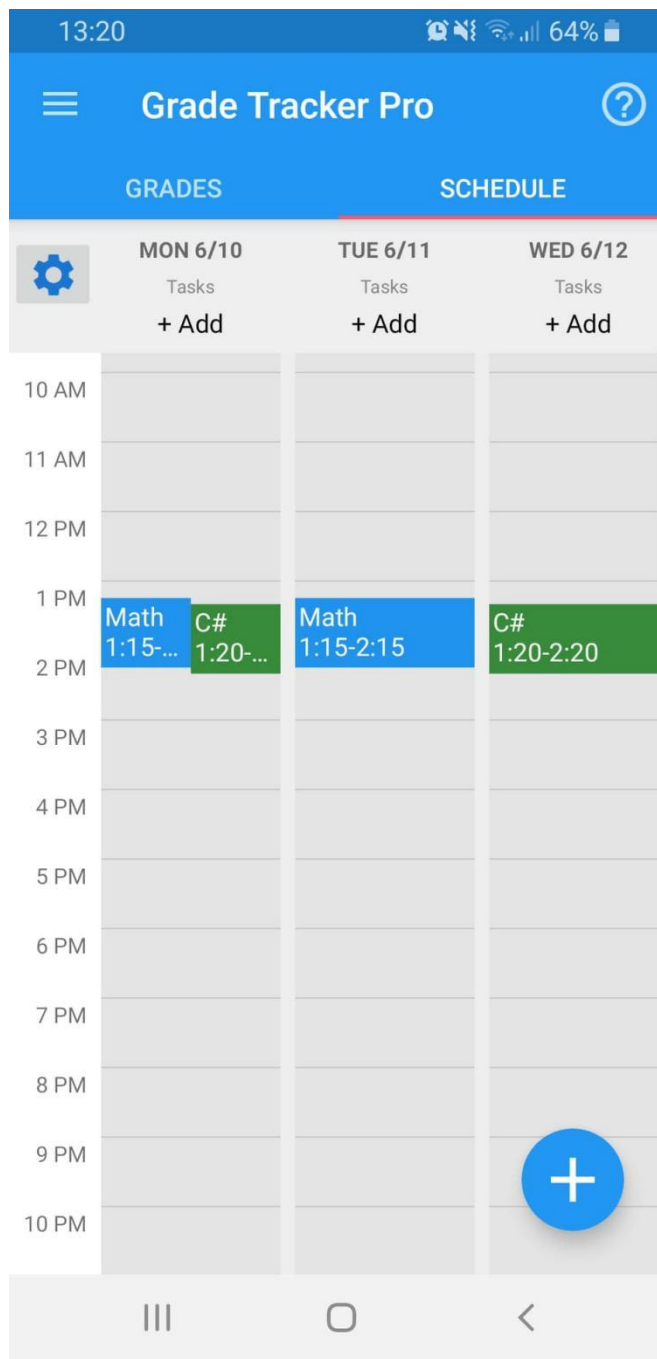
Μία από τις δημοφιλέστερες εφαρμογές, που η λειτουργία της είναι η οργάνωση προσωπικών δεδομένων, συγκεκριμένα βαθμών του χρήστη, είναι το **Grade Tracker Pro**. Το **Grade Tracker Pro** βρίσκεται διαθέσιμο στο **Google Play store** και είναι από τις υψηλότερα βαθμολογημένες, δωρεάν εφαρμογές στην πλατφόρμα. Οι δυνατότητες που προσφέρει στον χρήστη είναι η αποθήκευση και η οργάνωση των βαθμών που επιλέγει να εισάγει. Στην εικόνα 1, εμφανίζεται η κεντρική οθόνη της εφαρμογής. Στην κεντρική οθόνη, παρουσιάζεται στον χρήστη ένας πίνακας με τα δεδομένα που εισήγαγε και επιπρόσθετα έχει αυτομάτως ενσωματωμένους υπολογισμούς του μέσου όρου των δεδομένων αυτών, του ποσοστού επιτυχίας αλλά και με βάση το σύστημα βαθμών του Αμερικάνικου συστήματος κεντρικής βαθμολογίας.



Εικόνα 1. Κύρια οθόνη εφαρμογής Grade Tracker Pro

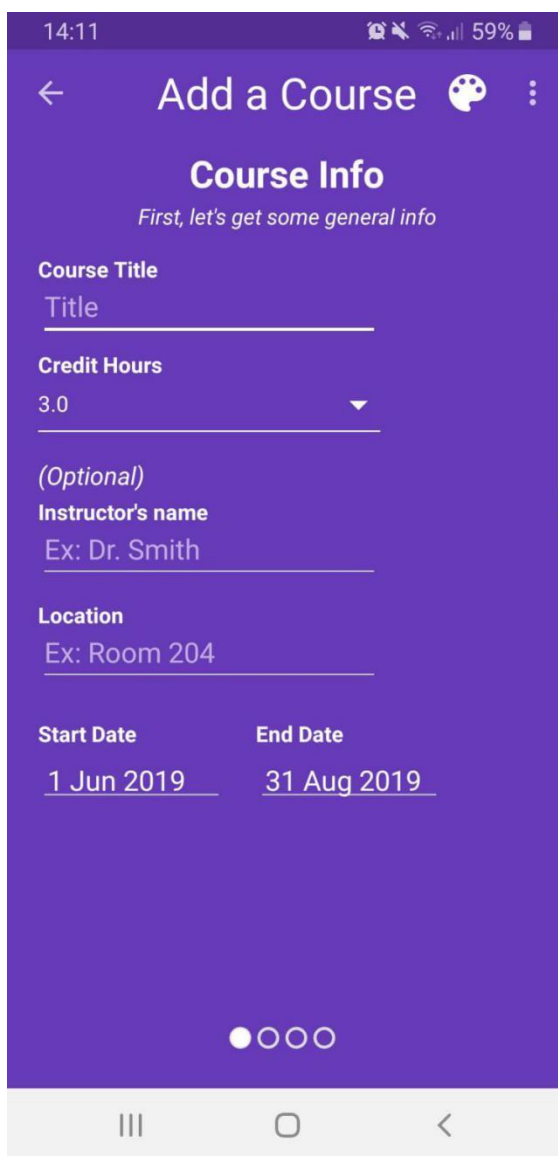
Επιλέγοντας ένα από τα μαθήματα που έχει εισάγει ο χρήστης, του προσφέρεται η επιλογή να τα επεξεργαστεί. Του παρέχεται η δυνατότητα να προσθέσει περισσότερα πεδία, όπως για παράδειγμα με ποια μέθοδο διεξήχθη η εξέταση. Επιπρόσθετα υπάρχει η επιλογή προσθήκης

επιπλέον μαθημάτων που έχουν εξεταστεί. Όπως παρουσιάζεται στην εικόνα 2, στον χρήστη παρέχεται και η δυνατότητα στησίματος ενός χρονοδιαγράμματος. Η διάταξη του επιτρέπει την προσθήκη διαλέξεων ή εξετάσεων των μαθημάτων που έχουν προστεθεί στην εφαρμογή.



Εικόνα 2. Χρονοδιάγραμμα

Στην εικόνα 3, παρουσιάζεται η επιλογή “**SCHEDULE**”. Η επιλογή αυτή, ενεργοποιείται επιλέγοντας το μπλε κουμπί, το οποίο βρίσκεται στην κάτω δεξιά γωνία της εφαρμογής. Η επιλογή επιτρέπει την διαμόρφωση οποιουδήποτε από τα μάθημα που έχει προσθέσει προηγουμένως ο χρήστης. Καθώς και η εισαγωγή του στο χρονοδιάγραμμα, με επιλογές όπως ώρες και μέρες διαλέξεων και εξετάσεων.

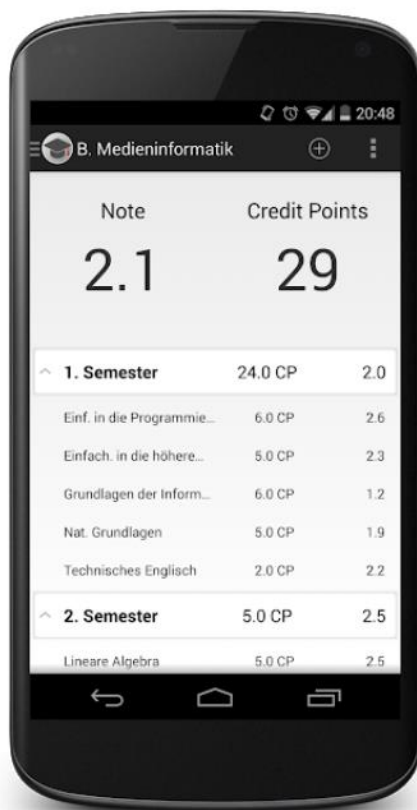


Εικόνα 3. Πρόσθεση μαθήματος

Στην συνέχεια βρισκόμαστε στην επιλογή πρόσθεσης μαθήματος. Εδώ υπάρχει μεγάλος αριθμός παραμέτρων και πληροφοριών που μπορούμε να προσθέσουμε όπως τίτλος μαθήματος, διάρκεια των μαθημάτων κλπ. Επιπρόσθετα μπορούμε να τροποποιήσουμε τον τρόπο υπολογισμού του συνολικού βαθμού του μαθήματος, για παράδειγμα 40% να είναι βαθμός εργαστηρίου 10% προφορικός βαθμός και 50% γραπτό, όπου το σύνολο είναι το 100%

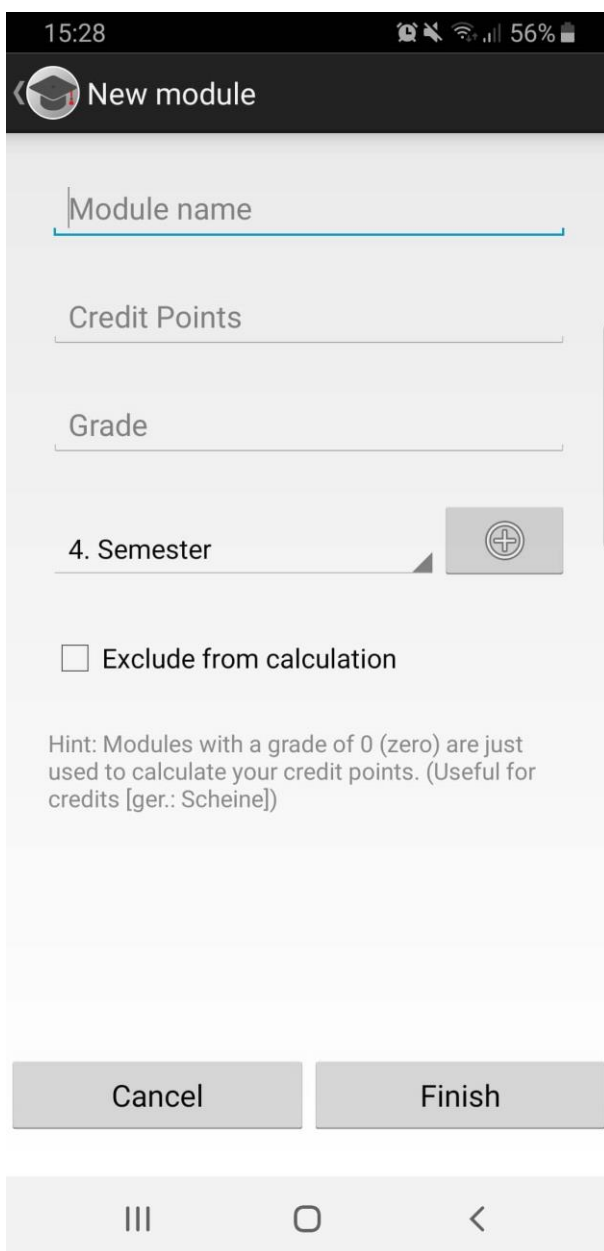
2.2 My Grades

Μία άλλη εφαρμογή της ίδιας κατηγορίας είναι το **My Grades**. Το **My Grades** είναι μια δωρεάν εφαρμογή η οποία είναι διαθέσιμη στο **Google Play store**. Η συγκεκριμένη εφαρμογή είναι προσωπικός καταχωρητής βαθμών ο οποίος παρουσιάζει τα μαθήματα σε μια οργανωμένη δομή ανά εξάμηνο. Στην παρακάτω εικόνα φαίνεται η κατηγοριοποίηση των βαθμών.



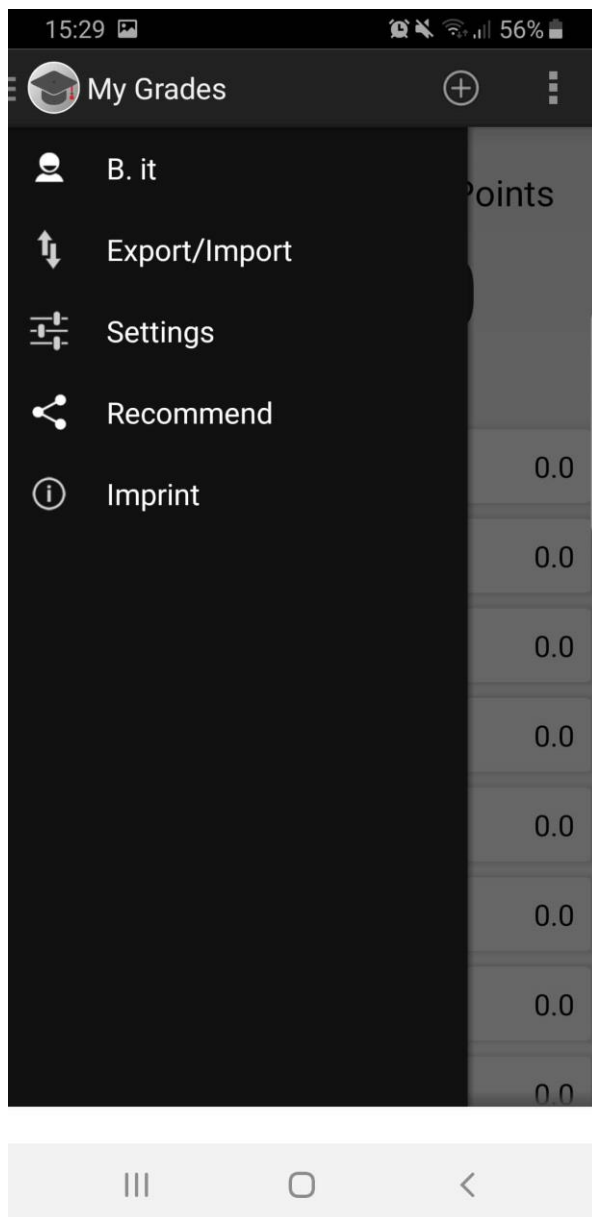
Εικόνα 4. Κύρια οθόνη εφαρμογής My Grades

Η πρώτη εικόνα που αντικρίζουν οι χρήστες είναι ένα περιβάλλον όπου είναι δομημένο ανά εξάμηνα , σε κάθε εξάμηνο μπορούν να τοποθετήσουν μαθήματα, δηλαδή βαθμούς και διδακτικές μονάδες. Στο κέντρο της οθόνης φαίνεται ο μέσος όρος καθώς και οι συνολικές διδακτικές μονάδες . Υπάρχει η δυνατότητα να προστεθούν νέα μαθήματα με το κουμπί πρόσθεσης που βρίσκεται στην αριστερή κορυφή της εφαρμογής.



Εικόνα 5. Πρόσθεση μαθήματος

Οι χρήστες συμπληρώνουν τα πεδία για να δημιουργήσουν ένα καινούργιο μάθημα στην αρχική σελίδα. Αυτά τα πεδία όπως φαίνονται στην εικόνα είναι το όνομα ,διδασκτικές μονάδες, βαθμός και τέλος εξάμηνο.



Εικόνα 6. Ρυθμίσεις

Επιπλέον η εφαρμογή παρέχει και την δυνατότητα εισαγωγής και εξαγωγής δεδομένων σε μορφή JSON .Επιπρόσθετα στην επιλογή **“settings”** ο χρήστης μπορεί να αλλάξει ρυθμίσεις όπως με πόσα ψηφία ακρίβειας να εμφανίζεται ο μέσος όρος του.

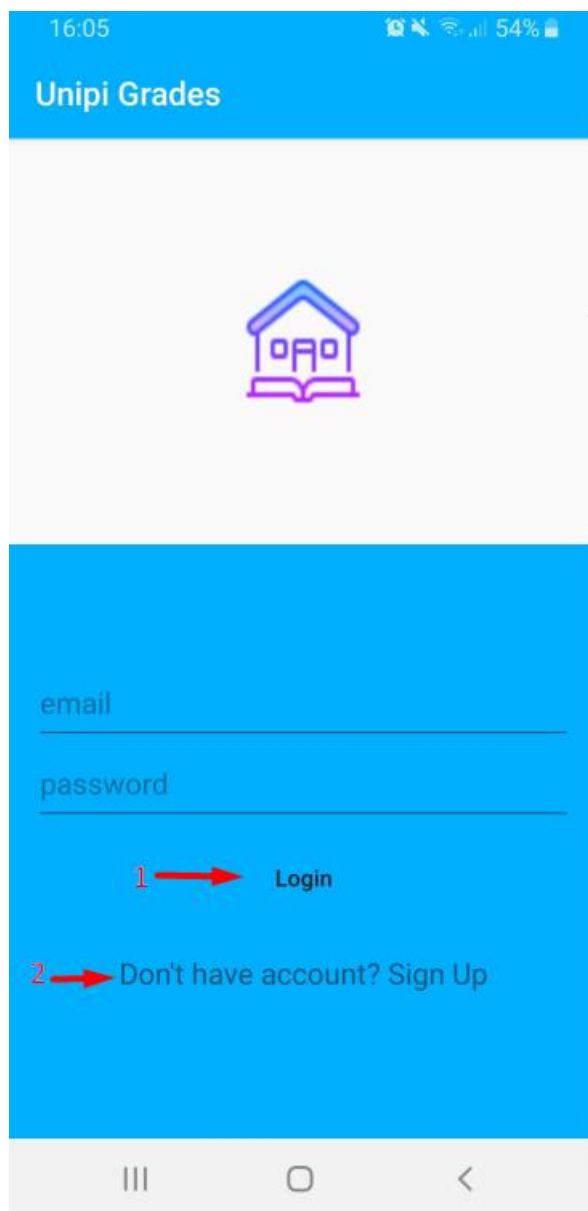
3. Παρουσίαση της εφαρμογής και ιστοσελίδας

3.1 Χρήση της εφαρμογής

Η χρήση της εφαρμογής προορίζεται για χρήστες οι οποίοι θέλουν να καταχωρήσουν τις επιδώσεις τους στα μαθήματα καθώς και άλλα στοιχεία . Αυτό πραγματοποιείτε με σκοπό να δημιουργηθούν κάποια στατιστικά στοιχεία ,τα οποία στατιστικά στοιχεία εμφανίζονται στην ιστοσελίδα όπου οι χρήστες μπορούν να συγκρίνουν τις επιδώσεις τους και να έχουν μία γενική εικόνα πού χρειάζεται να βελτιωθούν.

3.2 Δραστηριότητα Login

Η πρώτη οθόνη της εφαρμογής που συναντάει ο χρήστης είναι η δραστηριότητα **Login**. Σύμφωνα με την παρακάτω εικόνα παρέχονται κάποιες επιλογές, οι οποίες υπογραμμίζονται με τα κόκκινα βέλη. Οι επιλογές είναι 1) **Login** και 2) **Sign up**.



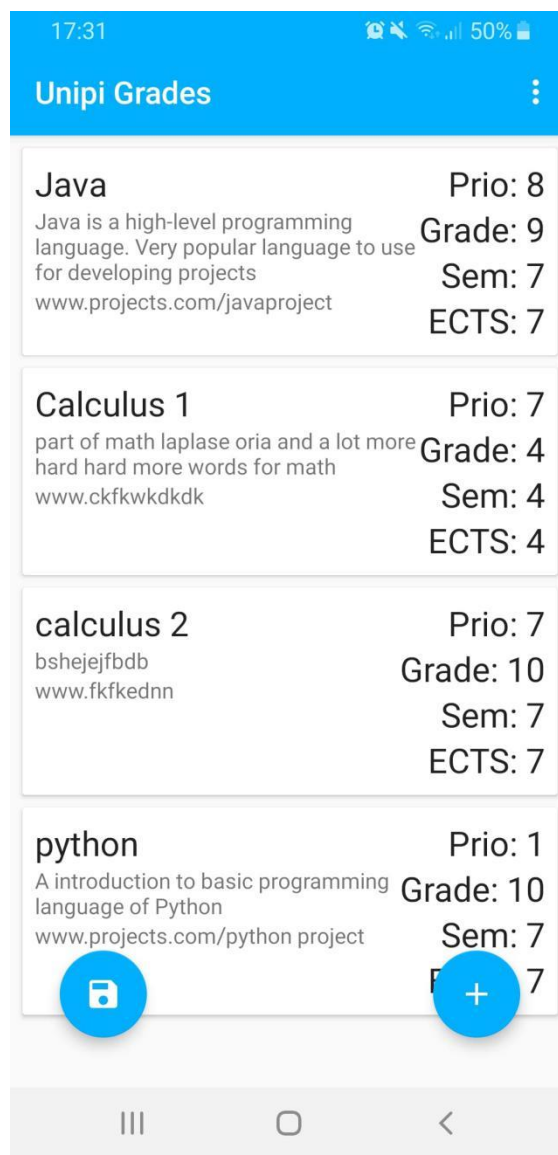
Εικόνα 7. Οθόνη σύνδεσης Unipi Grades

Βήμα 1)

Στο πεδίο τα οποία αναγράφεται <<Email>> ο χρήστης συμπληρώνει την ηλεκτρονική διεύθυνση με την οποία έχει εγγραφεί στην εφαρμογή. Στο επόμενο πεδίο στο οποίο αναγράφεται <<Password>> συμπληρώνει τον κωδικό που δήλωσε κατά την εγγραφή του στην εφαρμογή.

Βήμα 2)

Στην συνέχεια αφού έχει συμπληρώσει τα στοιχεία σωστά, πατάει το κουμπί **Login**. Με αυτήν την ενέργεια η εφαρμογή εκτελεί τους κατάλληλους ελέγχους για την είσοδο του. Εάν η σύνδεση είναι επιτυχής τότε θα μεταφερθεί στην κύρια οθόνη της εφαρμογής. Στην παρακάτω εικόνα φαίνεται η κυρίως οθόνη της εφαρμογής.



Εικόνα 8. Κύρια οθόνη εφαρμογής Unipi Grades

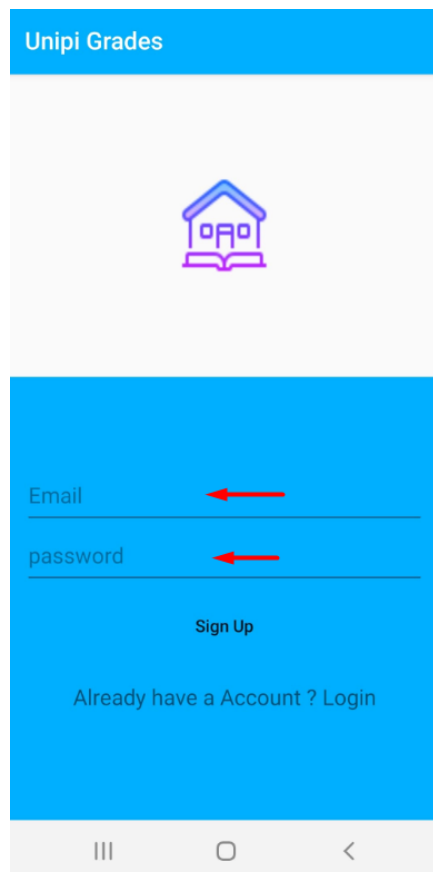
Περίπτωση 2)

Εάν ο χρήστης δεν έχει λογαριασμό στην εφαρμογή, θα πρέπει να πατήσει το βελάκι με τον αριθμό 2 όπως φαίνεται στην εικόνα 7. Στην συνέχεια ο χρήστης θα μεταφερθεί σε μία νέα φόρμα της οποίας η λειτουργία θα αναλυθεί στην επόμενη υπο-ενότητα.

3.3 Δραστηριότητα Register

Στην παρακάτω δραστηριότητα ο χρήστης έχει την δυνατότητα να εγγραφεί για πρώτη φορά στην εφαρμογή ώστε να είναι σε θέση να την χρησιμοποιήσει. Στην παρακάτω φόρμα ο χρήστης συμπληρώνει τα στοιχεία του αντίστοιχα, όπως θα έκανε και στην φόρμα του **Login**.

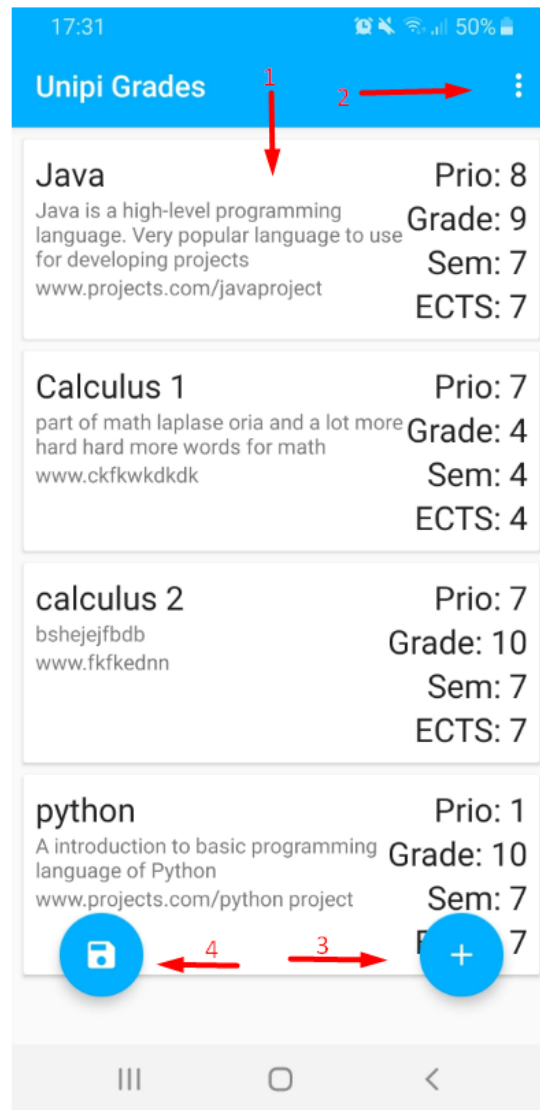
Τα κόκκινα βελάκια υποδεικνύουν τα πεδία που θα πρέπει να τοποθετηθεί ο κωδικός και το Email.



Εικόνα 9. Οθόνη εγγραφής Unipi Grades

3.4 Δραστηριότητα Home

Αυτή η δραστηριότητα αποτελεί τον πυρήνα της εφαρμογής ,αφού όλες οι ενέργειες καταγραφής των βαθμών πραγματοποιούνται σε αυτήν την δραστηριότητα. Στην παρακάτω εικόνα φαίνονται οι κόμβοι που περιέχουν τις πληροφορίες για τα μαθήματα και τα κουμπιά πλοήγησης.



Εικόνα 10. Κύρια οθόνη εφαρμογής Unipi Grades

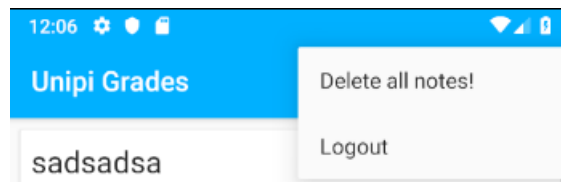
Λειτουργία αριθμημένων εικονοστοιχείων:

1)Κόμβος μαθήματος

Σε αυτήν αυτών τον κόμβο αναγράφονται όλες οι πληροφορίες όπου περιέχει κάθε μάθημα. Στα αριστερά βρίσκεται τίτλος του μαθήματος ,η περιγραφή του και εργασίες που έχουν υλοποιηθεί . Στην δεξιά βρίσκονται το “**prio**” όπου είναι ο αριθμός προτεραιότητας του μαθήματος ,”**grade**” ο βαθμός ,” **sem**” όπου αντιστοιχεί σε ποιο εξάμηνο ο χρήστης διδάχτηκε το μάθημα και τέλος το “**ects**” όπου αντιστοιχεί στις διδακτικές μονάδες του. Αν ο χρήστης πατήσει πάνω σε κάποιο κόμβο μπορεί μεταβάλλει αυτές τις πληροφορίες.

2)Καρτέλα επιλογών

Εδώ ο χρήστης μπορεί να επιλέξει ανάμεσα σε 2 ενέργειες την “**Logout**” όπου πραγματοποιεί αποσύνδεση και “**Delete all notes!**” όπου διαγράφει όλους τους κόμβους των μαθημάτων του.



Εικόνα 11. Καρτέλα επιλογών

3)Εισαγωγή καινούργιου μαθήματος

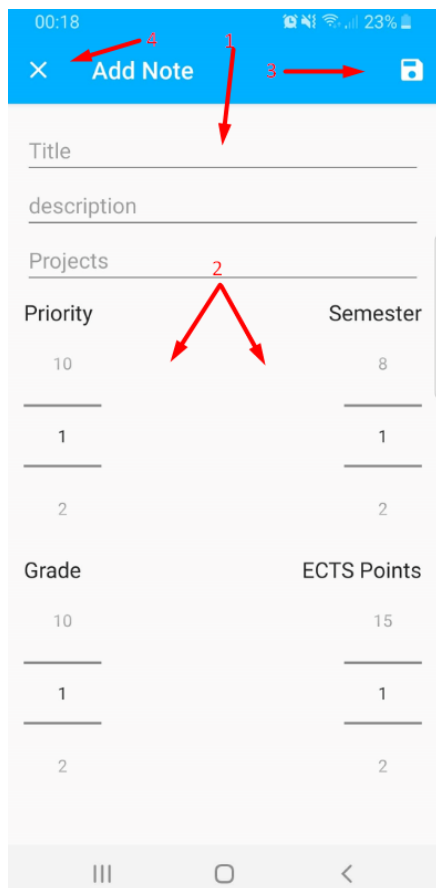
Με το πάτημα αυτού του κουμπιού ο χρήστης μεταφέρετε σε μια νέα καρτέλα όπου έχει την δυνατότητα να προσθέσει ένα νέο μάθημα. Η δραστηριότητα αυτή θα αναλυθεί στην επόμενη υπο-ενότητα.

4)Αποθήκευση

Αφού ο χρήστης τελειώσει την διαδικασία και προσθέσει όλα τα απαραίτητα μαθήματα , τότε πατάει αυτό το κουμπί για να αποθηκευτούν όλα τα μαθήματα στην βάση δεδομένων και να είναι διαθέσιμα στην ιστοσελίδα. Αν η αποθήκευσή γίνει επιτυχημένα τότε ο χρήστης θα λάβει ένα κατάλληλο μήνυμα.

3.5 Δραστηριότητα Add Note

Αυτή η δραστηριότητα αποτελεί την κύρια ενέργεια της εφαρμογής, οι χρήστες σε αυτήν την δραστηριότητα μπορούν να προσθέσουν νέα μαθήματα. Στην παρακάτω εικόνα φαίνονται οι κόμβοι που περιέχουν τις πληροφορίες για τα μαθήματα και τα κουμπιά πλοήγησης.



Priority	Semester
10	8
1	1
2	2

Grade	ECTS Points
10	15
1	1
2	2

Εικόνα 12. Πρόσθεση μαθήματος

Λειτουργία αριθμημένων εικονοστοιχείων:

1)Στοιχεία μαθήματος

Στα ενδεδειγμένα κενά ο χρήστης συμπληρώνει τα στοιχεία του μαθήματος τίτλο , περιγραφή και ένα σύνδεσμο σε εργασία που υλοποιήθηκε σε σχέση με αυτό το μάθημα.

2)Επιπλέον στοιχεία μαθήματος

Εδώ ο χρήστης επιλέγει την προτεραιότητα που έχει για τον ίδιο αυτό το μάθημα **“Priority”**, τον βαθμό του στο μάθημα **“Grade”** , σε ποιο εξάμηνο το διδάχτηκε **“Semester”** και την ποσότητα των διδακτικών μονάδων **“ECTS Points”** που αντιστοιχίζει στο μάθημα.

3)Αποθήκευση καινούργιου μαθήματος

Με το πάτημα αυτού του κουμπιού ο χρήστης αποθηκεύει το μάθημα στην τοπική βάση και εφόσον όλα τα στοιχεία που έχει εισάγει είναι σωστά θα λάβει ένα μήνυμά επιτυχίας.Στην συνέχεια θα μεταφερθεί στην κύρια σελίδα **“Home”** όπου και θα μπορεί να δει το μάθημά αυτό.

4)Ακύρωση

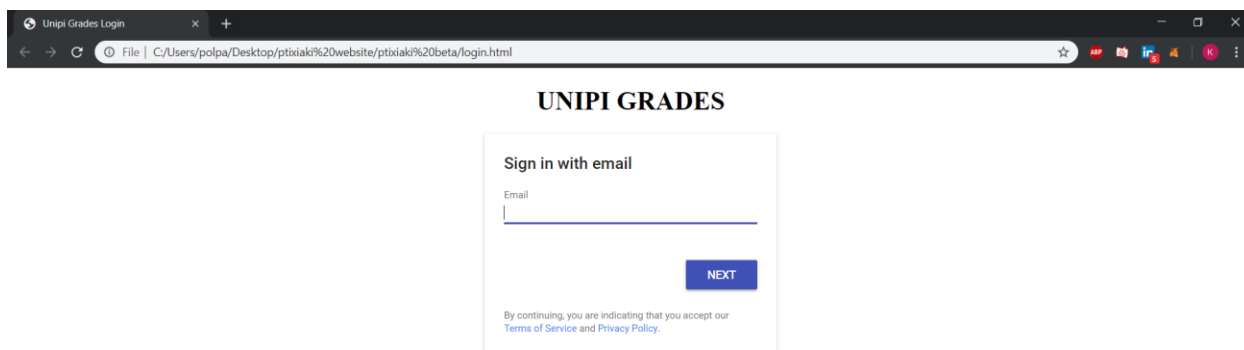
Ο χρήστης με αυτήν την ενέργεια θα επιστρέψει στην αρχική σελίδα **“Home”** χωρίς να έχει αποθηκευτεί οτιδήποτε έχει γράψει προηγουμένους.

3.6 Χρήση της ιστοσελίδας

Η χρήση της ιστοσελίδας προορίζεται για χρήστες οι οποίοι θέλουν να έχουν μία επισκόπηση των επιδόσεων τους στα μαθήματα, καθώς και άλλα στοιχεία . Αυτό πραγματοποιείται παρατηρώντας τα στατιστικά στοιχεία τα όποια είναι διαθέσιμα στην ιστοσελίδα. Οι χρήστες μπορούν να τα εξετάσουν και να συγκρίνουν τις επιδόσεις τους με άλλους φοιτητές και να βγάλουν συμπεράσματα στο πού χρειάζονται να βελτιωθούν.

3.7 Δραστηριότητα Login της ιστοσελίδας

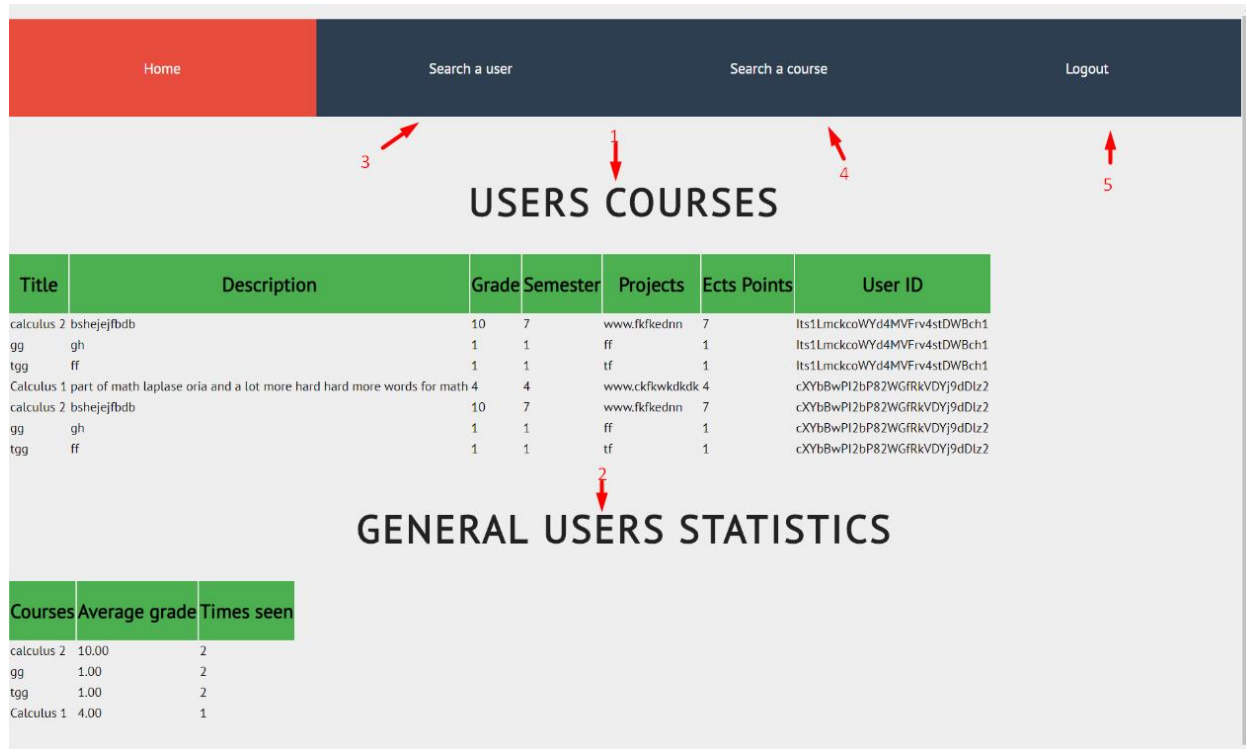
Η πρώτη οθόνη της ιστοσελίδας που συναντάει ο χρήστης είναι η δραστηριότητα **Login**. Σύμφωνα με την παρακάτω εικόνα ο χρήστης συμπληρώνει τα στοιχεία του. Το πεδίο τα οποία αναγράφεται <<**Email**>> ο χρήστης συμπληρώνει την ηλεκτρονική διεύθυνση με την οποία έχει εγγραφεί στην εφαρμογή. Πατώντας το κουμπί “**next**” εμφανίζεται το επόμενο πεδίο στο οποίο αναγράφεται <<**Password**>> , συμπληρώνει τον κωδικό που δήλωσε κατά την εγγραφή του στην εφαρμογή και πραγματοποιείτε η σύνδεση στην ιστοσελίδα.



Εικόνα 13 . Είσοδος στην ιστοσελίδα Unipi Grades

3.8 Δραστηριότητα Home της ιστοσελίδας

Αυτή η δραστηριότητα αποτελεί τον κύριο πυλώνα της ιστοσελίδας, εφόσον όλες οι πληροφορίες των μαθημάτων βρίσκονται ομαδοποιημένες εδώ. Στην παρακάτω εικόνα φαίνονται οι κόμβοι που περιέχουν τις πληροφορίες για τα μαθήματα και τα κουμπιά πλοήγησης.



Title	Description	Grade	Semester	Projects	Ects Points	User ID
calculus 2	bshejefbdb	10	7	www.fkfkednn	7	Its1LmckcoWYd4MVFr4stDWBch1
gg	gh	1	1	ff	1	Its1LmckcoWYd4MVFr4stDWBch1
tgg	ff	1	1	tf	1	Its1LmckcoWYd4MVFr4stDWBch1
Calculus 1	part of math laplace oria and a lot more hard hard more words for math	4	4	www.ckflwkdldk	4	cXYbBwPI2bP82WGfRkVDYf9dDlz2
calculus 2	bshejefbdb	10	7	www.fkfkednn	7	cXYbBwPI2bP82WGfRkVDYf9dDlz2
gg	gh	1	1	ff	1	cXYbBwPI2bP82WGfRkVDYf9dDlz2
tgg	ff	1	1	tf	1	cXYbBwPI2bP82WGfRkVDYf9dDlz2

Courses	Average grade	Times seen
calculus 2	10.00	2
gg	1.00	2
tgg	1.00	2
Calculus 1	4.00	1

Εικόνα 14. Αρχική σελίδα Unipi Grades

Λειτουργία αριθμημένων εικονοστοιχείων:

1)Στοιχεία χρηστών

Εδώ φαίνονται όλες οι πληροφορίες των μαθημάτων για τον κάθε χρήστη ξεχωριστά. Αυτές οι πληροφορίες είναι ομαδοποιημένες βάσει του **“User ID”**, που είναι το ξεχωριστό κλειδί του κάθε χρήστη. Οι πληροφορίες που εμφανίζονται είναι ο τίτλος του μαθήματος, περιγραφή, βαθμός, εξάμηνο, εργασία, διδακτικές μονάδες και τέλος το κλειδί του χρήστη.

2)Γενικά στατιστικά χρηστών

Εδώ εμφανίζονται στατιστικά των μαθημάτων που υπάρχουν στο σύστημα, το όνομα του μαθήματος **“Courses”**, ο μέσος όρος του μαθήματος **“Average Grade”** και πόσοι διαφορετικοί χρήστες το έχουν δηλώσει **“Times seen”**.

3)Αναζήτηση ως προς χρήστη

Με το πάτημα αυτού του κουμπιού ο χρήστης θα μεταφερθεί στην δραστηριότητα **“Search user”** που θα αναλυθεί στην επόμενη υπο-ενότητα 3.9.

4) Αναζήτηση ως προς μάθημα

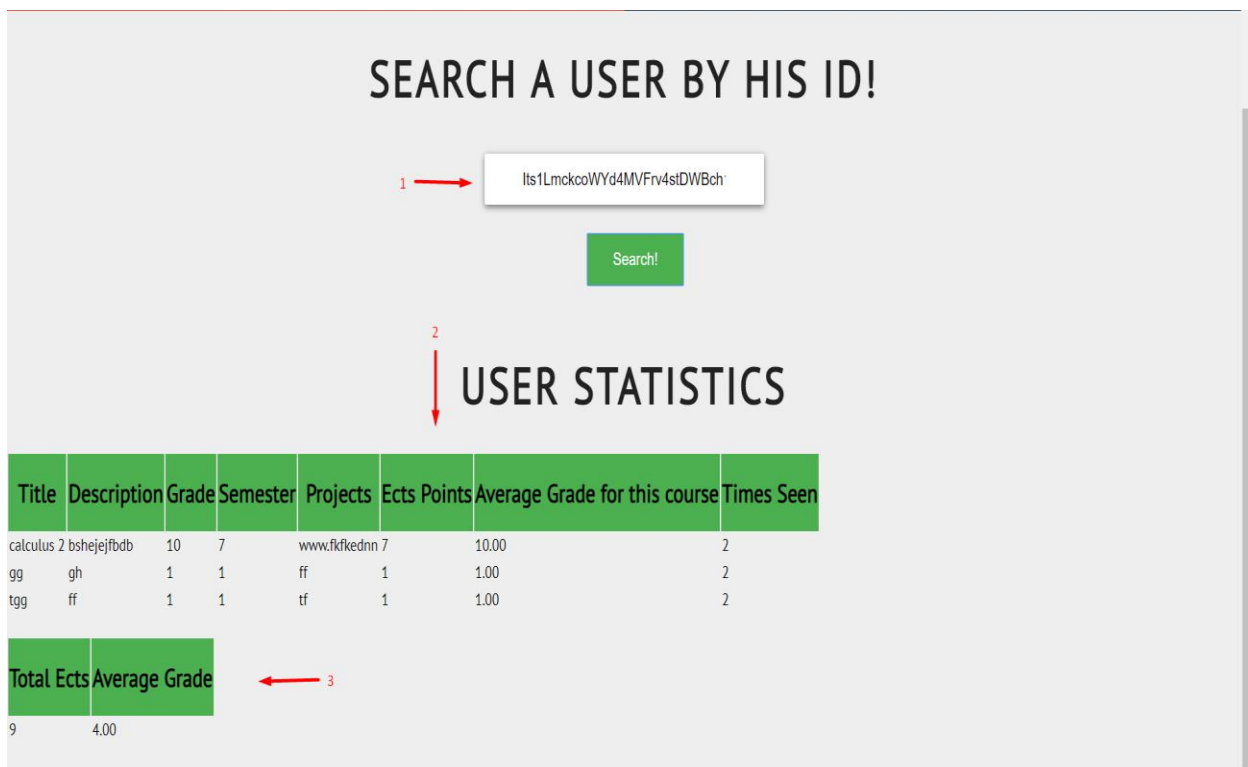
Με το πάτημα αυτού του κουμπιού ο χρήστης θα μεταφερθεί στην δραστηριότητα “**Search course**” που θα αναλυθεί παρακάτω στην υπο-ενότητα 3.10.

5) Αποσύνδεση

Με το πάτημα αυτού του κουμπιού ο χρήστης θα αποσυνδεθεί από την σελίδα

3.9 Δραστηριότητα Search a user της ιστοσελίδας

Σε αυτήν την δραστηριότητα οι χρήστες έχουν την δυνατότητα να αναζητήσουν τα στατιστικά συγκεκριμένου χρήστη με βάση το κλειδί του χρήστη “**User ID**”. Στην συνέχεια ο χρήστης μπορεί να μελετήσει τα αποτελέσματα του και σε σύγκριση με ανώνυμα στατιστικά στοιχεία χρηστών στο σύνολο της εφαρμογής. Στην παρακάτω εικόνα φαίνονται αναλυτικά.



Title	Description	Grade	Semester	Projects	Ects Points	Average Grade for this course	Times Seen
calculus 2	bshejejbdb	10	7	www.fkfkednn	7	10.00	2
gg	gh	1	1	ff	1	1.00	2
tgg	ff	1	1	tf	1	1.00	2

Total Ects	Average Grade
9	4.00

Εικόνα 15. Αναζήτηση χρήστη

Λειτουργία αριθμημένων εικονοστοιχείων:

1) Αναζήτηση χρήστη

Ο χρήστης πληκτρολογεί το “**User ID**” που αντιστοιχεί στον χρήστη που θέλει να αναζητήσει, στην συνέχεια πατάει το κουμπί “**Search**” και αν υπάρχει στην βάση δεδομένων τότε θα εμφανιστούν τα αποτελέσματα.

2) Στατιστικά χρήστη που αναζητήθηκε

Εδώ φαίνονται οι πληροφορίες για τον χρήστη που αναζητήθηκε. Είναι ο τίτλος , η περιγραφή , ο βαθμός που έχει ο χρήστης σε αυτό το μάθημα , το εξάμηνο , οι εργασίες , οι διδακτικές μονάδες , ο μέσος όρος βαθμού αυτού του μαθήματος και τέλος πόσοι χρήστες έχουν δηλώσει αυτό το μάθημα.

3) Γενικά στατιστικά χρήστη

Σε αυτό το κομμάτι φαίνονται τα συνολικά στατιστικά του χρήστη. Οι συνολικές διδακτικές μονάδες και ο συνολικός μέσος όρος του χρήστη.

3.10 Δραστηριότητα Search a course της ιστοσελίδας

Σε αυτήν την δραστηριότητα οι χρήστες έχουν την δυνατότητα να αναζητήσουν τα στατιστικά συγκεκριμένου μαθήματος , αναζητώντας με βάση το όνομα του μαθήματος. Στην συνέχεια ο χρήστης μπορεί να μελετήσει τα αποτελέσματα του και σε σύγκριση με ανώνυμα στατιστικά στοιχεία χρηστών στο σύνολο της εφαρμογής. Στην παρακάτω εικόνα φαίνονται αναλυτικά.

SEARCH A COURSE!

1 →

2 → **COURSE STATISTICS**

Title	Description	Grade	Semester	Projects	Ects Points	Average Grade for this course	User id
calculus 2 bshejefbdb		10	7	www.fkfkednn	7	10.00	Its1LmckcoWYd4MVFrV4stDWBch1
calculus 2 bshejefbdb		10	7	www.fkfkednn	7	10.00	cXYbBwPI2bP82WGRkVDYj9dDlz2

Total Ects	Average Grade
14	10.00

3 →

Εικόνα 16. Αναζήτηση μαθήματος

Λειτουργία αριθμημένων εικονοστοιχείων:

1)Αναζήτηση μαθήματος

Ο χρήστης πληκτρολογεί το μάθημα που θέλει να αναζητήσει, στην συνέχεια πατάει το κουμπί “Search” και αν υπάρχει στην βάση δεδομένων τότε θα εμφανιστούν τα αποτελέσματα.

2)Στατιστικά μαθήματος που αναζητήθηκε

Εδώ φαίνονται οι πληροφορίες για το μάθημα που αναζητήθηκε. Είναι ο τίτλος , η περιγραφή , ο βαθμός που έχει ο χρήστης σε αυτό το μάθημα ,το εξάμηνο , οι εργασίες , οι διδακτικές μονάδες , ο μέσος όρος βαθμού αυτού του μαθήματος και τέλος το κλειδί του χρήστη που αντιστοιχεί αυτό το μάθημα.

3)Γενικά στατιστικά μαθήματος

Σε αυτό το κομμάτι φαίνονται τα συνολικά στατιστικά του μαθήματος. Οι συνολικές διδακτικές μονάδες και ο συνολικός μέσος όρος του συγκεκριμένου μαθήματος.

4. Αρχιτεκτονική συστήματος

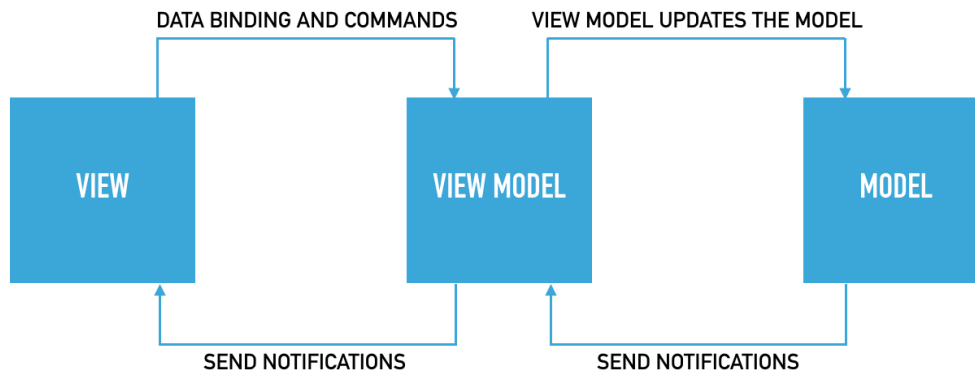
4.1 Ανάλυση αρχιτεκτονικής της εφαρμογής Unipi Grades

4.1.1 Εισαγωγή

Η ενότητα αυτή εξετάζει την αρχιτεκτονική της εφαρμογής , πιο συγκεκριμένα θα αναλυθούν το μοντέλο **MVVM** (**model view viewmodel**) που χρησιμοποιήθηκε στην ανάπτυξη της εφαρμογής καθώς και η βιβλιοθήκη **Android Jetpack**.

Ένα αρχιτεκτονικό μοτίβο είναι μια γενική, επαναχρησιμοποιήσιμη λύση σε ένα πρόβλημα που παρουσιάζεται συνήθως στην αρχιτεκτονική του λογισμικού, εντός ενός δεδομένου πλαισίου. Η καλή αρχιτεκτονική λογισμικού είναι ένας από τους βασικούς παράγοντες που συμβάλλουν στην επιτυχία του λογισμικού. Επιπλέον, η αρχιτεκτονική του συστήματος λογισμικού έχει σημαντικό αντίκτυπο στην ποιότητα του λογισμικού.

4.1.2 Ο τρόπος λειτουργίας του MVVM



Εικόνα 17. Τρόπος σύνδεσης MVVM

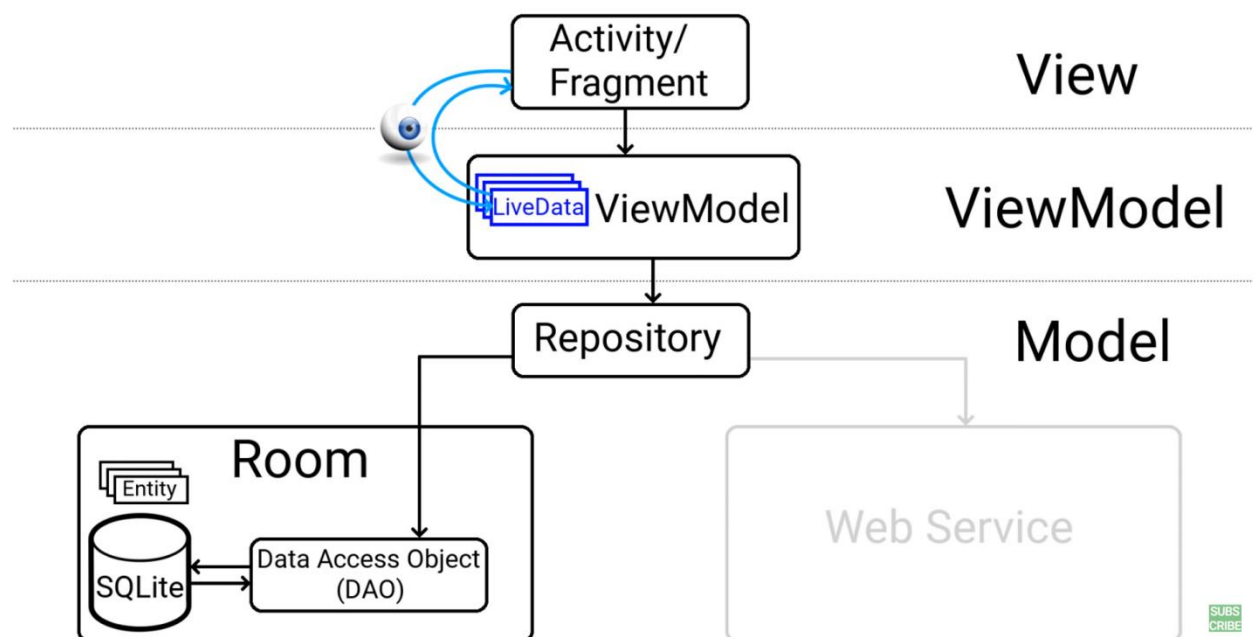
Η αρχιτεκτονική MVVM έχει τρία συστατικά, όπως δηλώνει το όνομά της, **Model**, **View** και **View Model**. Το στοιχείο **View** εμφανίζει το περιβάλλον στον χρήστη της εφαρμογής. Το **MVVM** είναι πιο φιλικό προς το σχεδιαστή, θα μπορέσει εύκολα να εφαρμοστεί από έναν σχεδιαστή αντί για έναν προγραμματιστή. Το **model** αντιπροσωπεύει τα δεδομένα. Το **View-Model**, το οποίο αντιπροσωπεύει ένα μοντέλο **view**, προορίζεται να διαχειρίζεται την κατάσταση του **View**. Θα περάσει τα δεδομένα και τις λειτουργίες για το **view** και επίσης θα διαχειριστεί τη λογική και τη συμπεριφορά του **view**. Ένα καλό **view-model** θα πρέπει να περιέχει μόνο τα δεδομένα που αφορούν την συγκεκριμένη κατάσταση, αντί για τα δεδομένα του **view** που αφορούν στην ονομασία και στον τύπο τους. Για παράδειγμα, εάν θέλουμε να αποθηκεύσουμε δεδομένα όταν είναι ενεργοποιημένο το κουμπί αποθήκευσης, αντί να ονομάσουμε τη μεταβλητή **isSaveButtonEnabled**, η ονομασία της κατάστασης **canSave** πρέπει να προτιμηθεί. Η σύνδεση μεταξύ του **ViewModel** και του **View** είναι πιο πολύπλοκη από ό,τι στην αρχιτεκτονική MVP. Υπάρχουν δύο τύποι συνδέσεων: η παραδοσιακή σύνδεση και η σύνδεση δεδομένων. Παραδοσιακές συνδέσεις είναι παρόμοιες σε **MVC** και **MVP**.

Το **Databinding** είναι ένας νέος μηχανισμός που εισήχθη στο **MVVM**. Επιτρέπει την άμεση σύνδεση του **view** με τις ιδιότητες και τις λειτουργίες του **view-model**. Με το **databinding** τα στοιχεία **model-view** δεν χρειάζεται να ειδοποιούν τις αλλαγές του **view** μέσω κώδικα, βλέπει ότι τα δεδομένα είναι φορτωμένα και εμφανίζει τα δεδομένα από το **view**. Για παράδειγμα, στο MVP και αρχιτεκτονική MVC, αφού φορτωθούν τα δεδομένα, ο presenter και ο controller θα ρυθμίσουν το **view** στον κώδικα. Με το μηχανισμό **databinding**, το **view** δεσμεύεται με αυτά τα δεδομένα και όταν αυτά φορτώνονται το **view** θα αλλάξει αυτόματα. Το **databinding** μεταξύ

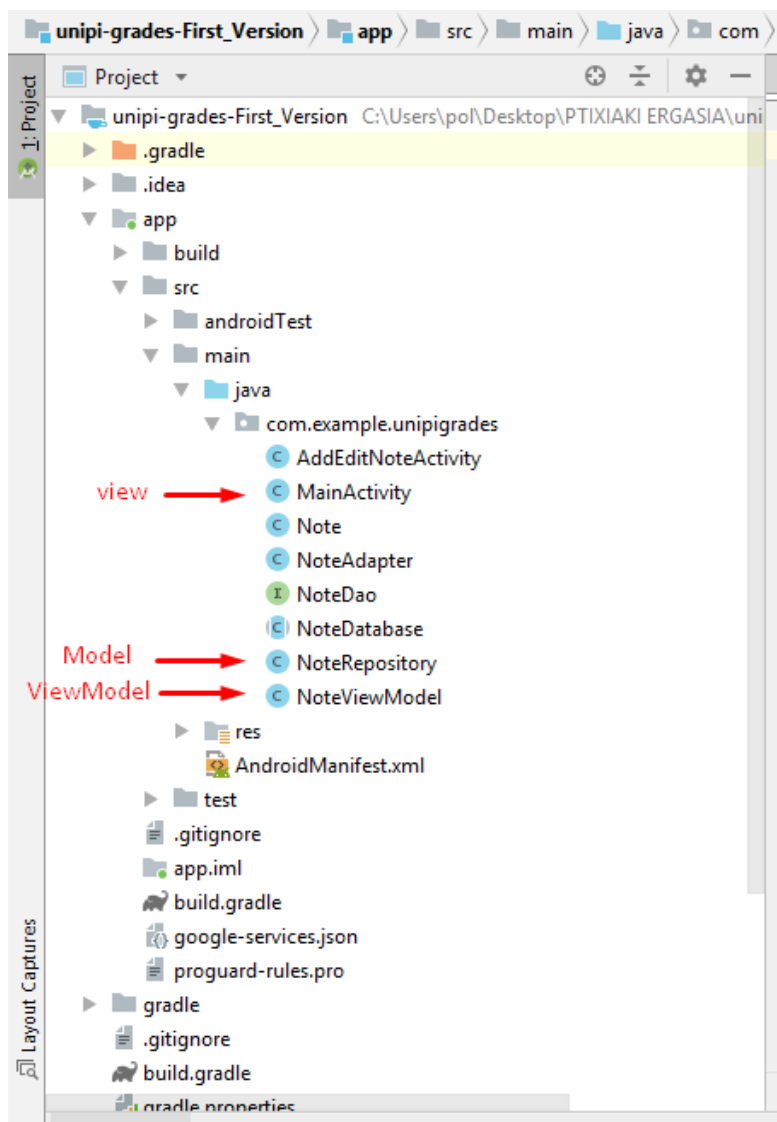
του **view-model** και **view** μπορεί να είναι κατευθυντικό και αμφίδρομο. Όταν αλλάζουν τα δεδομένα που είναι δεσμευμένα, τα δεδομένα στο **model-view** στοιχεία γνωρίζουν επίσης την αλλαγή. Αυτή η δέσμευση, η οποία αναφέρεται ως δεσμευτική ιδιότητα, είναι αμφίδρομη.

4.1.3 Η εφαρμογή του μοντέλου MVVM στο Unipi Grades

Η αρχιτεκτονική που χρησιμοποιήθηκε **MVVM** , **Model** , **View** και **ViewModel** έχει μεγάλη προσαρμοστικότητα με την βιβλιοθήκη android jetpack architecture components για αυτόν τον λόγο και χρησιμοποιήθηκε. Έχει επίσης και αρκετά θετικά καθώς παίρνει τις σωστές αποφάσεις στον κύκλο ζωής των activities και fragments, προλαμβάνει πιθανά σφάλματα της εφαρμογής, προβλήματα στον κύκλο ζωής και διαρροές μνήμης. Στην παρακάτω εικόνα φαίνεται πιο αναλυτικά η αρχιτεκτονική.



Εικόνα 18. Τρόπος εφαρμογής MVVM στο Unipi Grades



Εικόνα 19. Οι κλάσεις της εφαρμογής

Στις 2 παραπάνω εικόνες δίνεται ο τρόπος σύνδεσης μιας MVVM αρχιτεκτονικής και πώς εφαρμόστηκε αυτό στην περίπτωση του Unipi Grades. Οι κύριες εργασίες αυτών είναι οι παρακάτω :

- **ViewModel**, κύριο έργο της είναι να διατηρεί προσωρινά και να επεξεργάζεται τα δεδομένα για το **UserInterface**, έτσι ώστε να μην χρειάζεται να περνούν κατευθείαν στα activities και fragments που συνήθως δημιουργεί προβλήματα.

- **View**, αποτελείται από όλα τα activity, fragments της εφαρμογής, και το MainActivity. Το **View** λαμβάνει όλα του τα δεδομένα απευθείας μέσω του **ViewModel**. Τα καθήκοντα του είναι να δείχνει τις πληροφορίες στην οθόνη και να αναφέρει όλες τις αλληλεπιδράσεις του χρήστη με την εφαρμογή, πίσω στο **ViewModel**.
- **Model**, είναι ένα επιπλέον στρώμα το οποίο μεσολαβεί μεταξύ των πηγών δεδομένων και πιο συγκεκριμένα της SQLite. Η εργασία αυτής της κλάσης είναι να αποθηκεύει δεδομένα, μακροπρόθεσμα και κυρίως ώστε το **ViewModel** να μην αλληλοεπιδρά με αυτά τα δεδομένα απευθείας.

Για παράδειγμα στην κυρίως κλάση δημιουργείτε ένα object της κλάσης **noteViewModel**:

```
private NoteViewModel noteViewModel;

@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);

    FloatingActionButton buttonAddnote = findViewById(R.id.button_add_note);
    buttonAddnote.setOnClickListener((v) -> {
        Intent intent = new Intent( packageContext: MainActivity.this, AddEditNoteActivity.class);
        startActivityForResult(intent, ADD_NOTE_REQUEST);
    });

    RecyclerView recyclerView= findViewById(R.id.recycler_view);
    recyclerView.setLayoutManager(new LinearLayoutManager( context: this));
    recyclerView.setHasFixedSize(true);

    final NoteAdapter adapter = new NoteAdapter();
    recyclerView.setAdapter(adapter);

    noteViewModel = ViewModelProviders.of( activity: this).get(NoteViewModel.class);
    noteViewModel.getAllNotes().observe( owner: this, (notes) -> {
        adapter.submitList(notes);
    });

    new ItemTouchHelper(new ItemTouchHelper.SimpleCallback( dragDirs: 0,
        swipeDirs: ItemTouchHelper.LEFT| ItemTouchHelper.RIGHT) {
        @Override
        public boolean onMove(@NonNull RecyclerView recyclerView, @NonNull RecyclerView.ViewHolder viewHolder, @No
            return false;
    })
}
```

Εικόνα 20. Κλάση Main Activity

Στην συνέχεια γίνεται κλήση στην μέθοδο **getAllNotes()**. Τα δεδομένα που προκύπτουν είναι αυτά τα οποία εμφανίζονται στην κύρια οθόνη μέσω του **recycleView**.

Στην κλάση **NoteViewModel** για να γίνει κλήση στην μέθοδο **getAllNotes()** και να έχει τα δεδομένα που είναι αποθηκευμένα στην βάση πρέπει να έχει πρόσβαση στα δεδομένα **allNotes** που βρίσκονται αποθηκευμένα στην κλάση **NoteRepository**.

```
public NoteViewModel(@NonNull Application application) {  
    super(application);  
    repository = new NoteRepository(application);  
    allNotes= repository.getAllNotes();  
}  
  
public void insert(Note note) { repository.insert(note); }  
  
public void update(Note note) { repository.update(note); }  
  
public void delete(Note note) {  
    repository.delete(note);  
}  
  
public void deleteAllNotes() { repository.deleteAllNotes(); }  
  
public LiveData<List<Note>> getAllNotes() { return allNotes; }  
}
```

Εικόνα 21. Κλάση NoteViewModel

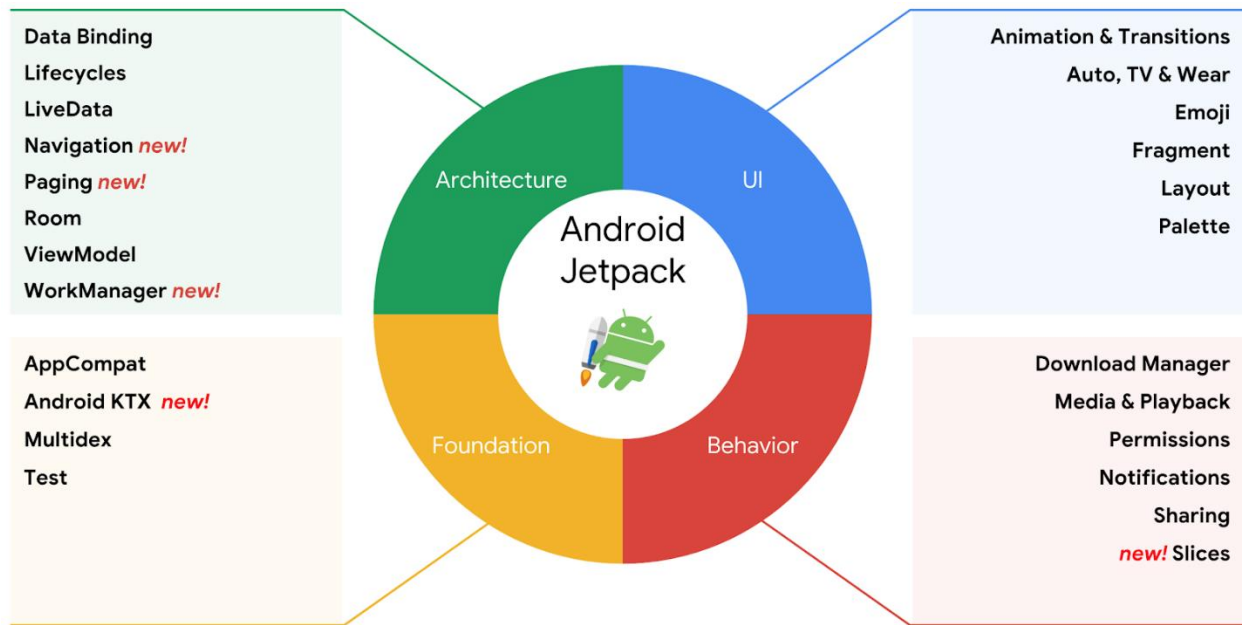
Όλα τα δεδομένα που είναι αποθηκευμένα στην βάση δεδομένων μεταβιβάζονται στην **allNotes** λίστα που στην συνέχεια χρησιμοποιούνται από την κλάση **getAllNotes()**.


```
import ...  
  
public class NoteRepository {  
  
    private NoteDao noteDao;  
    private LiveData<List<Note>> allNotes;  
  
    public NoteRepository(Application application) {  
        NoteDatabase database= NoteDatabase.getInstance(application);  
        noteDao = database.noteDao();  
        allNotes =noteDao.getAllNotes();  
    }  
}
```

Εικόνα 22. Κλάση NoteRepository

4.1.4 Ο τρόπος λειτουργίας του Android Jetpack Architecture components

Τα στοιχεία αρχιτεκτονικής Android είναι μια συλλογή βιβλιοθηκών τα οποία βοηθούν να σχεδιάσουν ισχυρές εφαρμογές, που έχουν την δυνατότητα να δοκιμαστούν και είναι διατηρήσιμες. Οι κλάσεις είναι η αρχή για τη διαχείριση του κύκλου ζωής των στοιχείων του περιβάλλοντος του χρήστη και για τη διαχείριση των δεδομένων.



Εικόνα 23. Τα ξεχωριστά μέρη του Android Jetpack

- Η βιβλιοθήκη **Room** παρέχει ένα «φανταστικό» στρώμα πάνω από το SQLite για να επιτρέψει μια πιο ισχυρή πρόσβαση στην βάση δεδομένων ενώ αξιοποιεί την πλήρη ισχύ του SQLite. Η βιβλιοθήκη βοηθά να δημιουργηθεί μια προσωρινή μνήμη των δεδομένων της εφαρμογής σε μια συσκευή που εκτελεί την εφαρμογή. Αυτή η κρυφή μνήμη, η οποία χρησιμεύει ως μοναδική πηγή αλήθειας της εφαρμογής, επιτρέπει στους χρήστες να βλέπουν ένα σταθερό αντίγραφο βασικών πληροφοριών μέσα στην εφαρμογή, ανεξαρτητως από το αν οι χρήστες έχουν σύνδεση στο διαδίκτυο.

- Η βιβλιοθήκη **Data Binding** είναι μια βιβλιοθήκη υποστήριξης που επιτρέπει να δεσμεύονται τα στοιχεία UI στις διατάξεις με τις πηγές δεδομένων στην εφαρμογή χρησιμοποιώντας δηλωτική μορφή και όχι προγραμματιστική.

Οι διατάξεις συχνά ορίζονται σε δραστηριότητες με κώδικα που καλεί τις μεθόδους στο πλαίσιο του UI.

- Η κλάση **ViewModel** έχει σχεδιαστεί για να αποθηκεύει και να διαχειρίζεται δεδομένα που σχετίζονται με το UI με συνειδητό τρόπο ζωής. Η κλάση **ViewModel** επιτρέπει στα

δεδομένα να επιβιώσουν από τις αλλαγές διαμόρφωσης όπως πχ οι περιστροφές της οθόνης.

Το πλαίσιο Android διαχειρίζεται τους κύκλους ζωής των ελεγκτών UI, όπως activities και fragments. Το πλαίσιο μπορεί να αποφασίσει να καταστρέψει ή να αναδημιουργήσει έναν ελεγκτή UI σε απόκριση ορισμένων ενεργειών του χρήστη ή συμβάντων της συσκευής που είναι εντελώς εκτός ελέγχου.

Εάν το σύστημα καταστρέψει ή αναδημιουργήσει έναν ελεγκτή UI, τυχόν μεταβατικά δεδομένα που σχετίζονται με το UI που αποθηκεύονται σε αυτά θα χαθούν. Για παράδειγμα, η εφαρμογή μπορεί να περιλαμβάνει μια λίστα χρηστών σε μια από τις δραστηριότητές της. Όταν η δραστηριότητα δημιουργηθεί ξανά για μια αλλαγή διαμόρφωσης, η νέα δραστηριότητα πρέπει να επανακτήσει τη λίστα των χρηστών. Για απλά δεδομένα, η δραστηριότητα μπορεί να χρησιμοποιήσει τη μέθοδο **onSaveInstanceState ()** και να επαναφέρει τα δεδομένα της από τη δέσμη στο **onCreate ()**, αλλά αυτή η προσέγγιση είναι κατάλληλη μόνο για μικρές ποσότητες δεδομένων, όχι για μεγάλες ποσότητες δεδομένων όπως μια λίστα χρηστών ή **bitmap**. Ένα άλλο πρόβλημα είναι ότι οι ελεγκτές UI πρέπει συχνά να κάνουν ασύγχρονες κλήσεις που μπορεί να χρειαστούν κάποιο χρόνο για να επιστρέψουν. Ο ελεγκτής UI πρέπει να διαχειριστεί αυτές τις κλήσεις και να διασφαλίσει ότι το σύστημα θα καθαρίσει αφού καταστραφεί για να αποφευχθούν τυχόν διαρροές μνήμης. Αυτή η διαχείριση απαιτεί μεγάλη συντήρηση και στην περίπτωση που το αντικείμενο αναδημιουργείται για μια αλλαγή διαμόρφωσης. Αυτό προκαλεί σπατάλη πόρων, αφού το αντικείμενο ενδέχεται να χρειαστεί να επαναλάβει τις κλήσεις που έχει ήδη πραγματοποιήσει. Οι ελεγκτές UI, όπως τα activities και τα fragments, αποσκοπούν κυρίως στην απεικόνιση δεδομένων UI, στην αντίδραση των ενεργειών των χρηστών ή στην επεξεργασία της επικοινωνίας του λειτουργικού συστήματος, όπως αιτήματα αδειών. Η απαίτηση για τους υπεύθυνους επεξεργασίας UI να είναι επίσης υπεύθυνοι για τη φόρτωση δεδομένων από μια βάση δεδομένων ή ένα δίκτυο προσθέτει φούσκωμα στην κλάση. Η ανάθεση υπερβολικής υπευθυνότητας στους ελεγκτές UI μπορεί να οδηγήσει σε μια ενιαία κλάση που προσπαθεί να χειριστεί από μόνη της το έργο μιας εφαρμογής αντί να μεταβιβάσει την εργασία σε άλλες κλάσεις. Η ανάθεση υπερβολικής υπευθυνότητας στους ελεγκτές UI με τον τρόπο αυτό καθιστά επίσης πολύ πιο δύσκολο τον έλεγχο. Είναι πιο εύκολο και αποτελεσματικότερο να διαχωριστεί η ιδιοκτησία δεδομένων **view** από τη λογική του ελεγκτή UI.

- Το **LiveData** είναι μια κλάση που κρατάει δεδομένα. Το **LiveData** έχει πλήρη γνώση του κύκλου ζωής, που σημαίνει ότι σέβεται τον κύκλο ζωής άλλων κλάσεων της εφαρμογής,

όπως *activities*, *fragments* ή υπηρεσίες. Αυτή η επίγνωση διασφαλίζει ότι μόνο το **LiveData** ενημερώνει τους παρατηρητές των κλάσεων της εφαρμογής που βρίσκονται σε κατάσταση ενεργού κύκλου ζωής. Το **LiveData** θεωρεί έναν παρατηρητή, ο οποίος εκπροσωπείται από την τάξη **Observer**, να βρίσκεται σε ενεργή κατάσταση ακόμη και αν ο κύκλος ζωής του είναι στην κατάσταση *STARTED* ή *RESUMED*. Το **LiveData** ειδοποιεί μόνο ενεργούς παρατηρητές για ενημερώσεις. Οι ανενεργοί παρατηρητές, που έχουν εγγραφεί για να παρακολουθήσουν αντικείμενα *LiveData*, δεν ενημερώνονται για αλλαγές.

- Τα **Lifecycle-aware** έχουν επίγνωση του κύκλου ζωής των εφαρμογών και εκτελούν ενέργειες ως απάντηση σε μια αλλαγή της κατάστασης του κύκλου ζωής ενός άλλου στοιχείου, όπως *activities* και *fragments*. Αυτά τα στοιχεία βοηθούν να παραχθούν ένας πιο οργανωμένος και συχνά ελαφρύτερος κώδικας, ο οποίος είναι ευκολότερος στη συντήρηση. Ένα κοινό πρότυπο είναι να υλοποιηθούν οι ενέργειες των εξαρτημένων στοιχείων στις μεθόδους των *activities* και των *fragments* του κύκλου ζωής. Ωστόσο, αυτό το μοντέλο οδηγεί σε κακή οργάνωση του κώδικα και στη διάδοση των σφαλμάτων. Χρησιμοποιώντας εξαρτήματα με γνώση του κύκλου ζωής, μπορεί να μετακινηθεί ο κώδικας των εξαρτημένων στοιχείων εκτός των μεθόδων του κύκλου ζωής και στα ίδια τα στοιχεία. Το πακέτο `android.arch.lifecycle` παρέχει κλάσεις και διεπαφές που επιτρέπουν να δημιουργηθούν συστατικά γνωρίσματα του κύκλου ζωής, τα οποία μπορούν να προσαρμόσουν αυτόματα τη συμπεριφορά τους, με βάση την τρέχουσα κατάσταση κύκλου ζωής ενός *activity* ή ενός *fragment*. Τα περισσότερα από τα συστατικά στοιχεία της εφαρμογής που ορίζονται στο *Android Framework* έχουν κύκλους ζωής συνδεδεμένους με αυτά. Οι κύκλοι ζωής διαχειρίζονται από το λειτουργικό σύστημα ή από τον κώδικα πλαισίου που εκτελείται στη διαδικασία. Είναι σημαντικοί για το πώς λειτουργεί το *Android* και η εφαρμογή πρέπει να τους σέβεται. Εάν δεν γίνετε αυτό, ενδέχεται να προκληθούν διαρροές μνήμης ή ακόμα και διακοπές λειτουργίας.

4.1.5 Android Jetpack Architecture components στην εφαρμογή Unipi Grades

Για την εφαρμογή **Unipi Grades** χρησιμοποιήθηκαν κάποια από τα Android jetpack architecture components , συγκεκριμένα χρησιμοποιήθηκαν τα **Room** , **ViewModel** και **LiveData**. Σε συνδυασμό με το **MVVM** μοντέλο που αναλύθηκε παραπάνω δίνουν πολλά πλεονεκτήματα στην εφαρμογή καθώς είναι φτιαγμένα ώστε να λειτουργούν σε συνεργασία. Η εφαρμογή, λόγω αυτού του καθαρού σχεδιασμού, δεν περιέχει bugs και μπορεί να λειτουργήσει σωστά σε πολλές διαφορετικές καταστάσεις.

Για παράδειγμα στην κλάση **Note** γίνεται η αρχικοποίηση της κλάσης και χρησιμοποιώντας την βιβλιοθήκη **Room** η βάση δεδομένων και συγκεκριμένα το **note_table** χρησιμοποιείται ως ένα αντικείμενο.

```
package com.example.unipigrades;

import ...

@Entity(tableName = "note_table")
public class Note {

    @PrimaryKey(autoGenerate = true)
    private int id;

    private String title;

    private String description;

    private int priority;

    private int grade;

    private String projects;

    public Note(String title, String description, int priority, int grade , String projects) {
        this.title = title;
        this.description = description;
        this.priority = priority;
        this.grade = grade;
        this.projects = projects;
    }

    public void setId(int id) { this.id = id; }

    public int getId() { return id; }
```

Εικόνα 24. Κλάση Note

Στην συνέχεια χρησιμοποιώντας την κλάση **NoteDao**, αντί να επιστρέφεται πίσω κάποιος κέρσοντας, το **Room** δίνει την δυνατότητα τα αποτελέσματα που έρχονται πίσω να είναι κατευθείαν της μορφής **note java** αντικείμενα.

```
@Dao
public interface NoteDao {

    @Insert
    void insert(Note note);

    @Update
    void update(Note note);

    @Delete
    void delete(Note note);

    @Query("DELETE FROM note_table")
    void deleteAllNotes();

    @Query("SELECT * FROM note_table ORDER BY priority DESC")
    LiveData<List<Note>> getAllNotes();
}
```

Εικόνα 25. Κλάση NoteDao

Η κλάση **NoteViewModel** είναι πολύ χρήσιμη όταν υπάρχει κάποια αλλαγή ρυθμίσεων, όπως για παράδειγμα όταν γίνεται αλλαγή προσανατολισμού της συσκευής και καλεστεί η **OnDestroy()**, τότε χάνονται όλα τα δεδομένα από τα activities. Όμως τα δεδομένα του **NoteViewModel** δεν χάνονται. Έτσι η χρήση του, ως μεσολαβητής των δεδομένων στα activities, αποφέρει την διατήρηση των δεδομένων που δεν χάνονται ποτέ.

```
1 package com.example.unipigrades;
2
3 import ...
4
5
6
7
8
9
10 public class NoteViewModel extends AndroidViewModel {
11     public NoteRepository repository;
12     private LiveData<List<Note>> allNotes;
13
14     public NoteViewModel(@NonNull Application application) {
15         super(application);
16         repository = new NoteRepository(application);
17         allNotes= repository.getAllNotes();
18     }
19
20     public void insert(Note note) { repository.insert(note); }
21
22
23
24     public void update(Note note) { repository.update(note); }
25
26
27
28     public void delete(Note note){
29         repository.delete(note);
30     }
31
32     public void deleteAllNotes() { repository.deleteAllNotes(); }
33
34
35
36     public LiveData<List<Note>> getAllNotes() { return allNotes; }
37
38 }
39
40
```

Εικόνα 26. Κλάση NoteViewModel

Από την παραπάνω εικόνα φαίνεται πως γίνεται η αρχικοποίηση της κλάσης και πώς όλα τα δεδομένα που υπάρχουν στην **SQLite** βάση μπορούν να ανακτηθούν μέσω του **NoteViewModel** και να χρησιμοποιηθούν από τα άλλα activities.

```
noteViewModel = ViewModelProviders.of( activity: this).get(NoteViewModel.class);
noteViewModel.getAllNotes().observe( owner: this, (notes) → {
    adapter.submitList(notes);
});
```

Εικόνα 27. Χρήση της κλάσης NoteViewModel στην κλάση MainActivity

Έδω για παράδειγμα φαίνεται πώς χρησιμοποιείται στην **MainActivity**.

Τα **LiveData** είναι ένα περιτύλιγμα που μπορεί να χρησιμοποιηθεί με οποιαδήποτε δεδομένα, συμπεριλαμβανομένων αντικειμένων που υλοποιούν `collections` και `lists`. Ένα αντικείμενο **LiveData** συνήθως αποθηκεύεται μέσα σε ένα αντικείμενο **NoteViewModel** και προσεγγίζεται μέσω μιας μεθόδου `getter`, όπως φαίνεται στο παρακάτω παράδειγμα:

```
import ...

public class NoteViewModel extends AndroidViewModel {
    public NoteRepository repository;
    private LiveData<List<Note>> allNotes;

    public NoteViewModel(@NonNull Application application) {
        super(application);
        repository = new NoteRepository(application);
        allNotes = repository.getAllNotes();
    }

    public void insert(Note note) { repository.insert(note); }

    public void update(Note note) { repository.update(note); }

    public void delete(Note note) {
        repository.delete(note);
    }

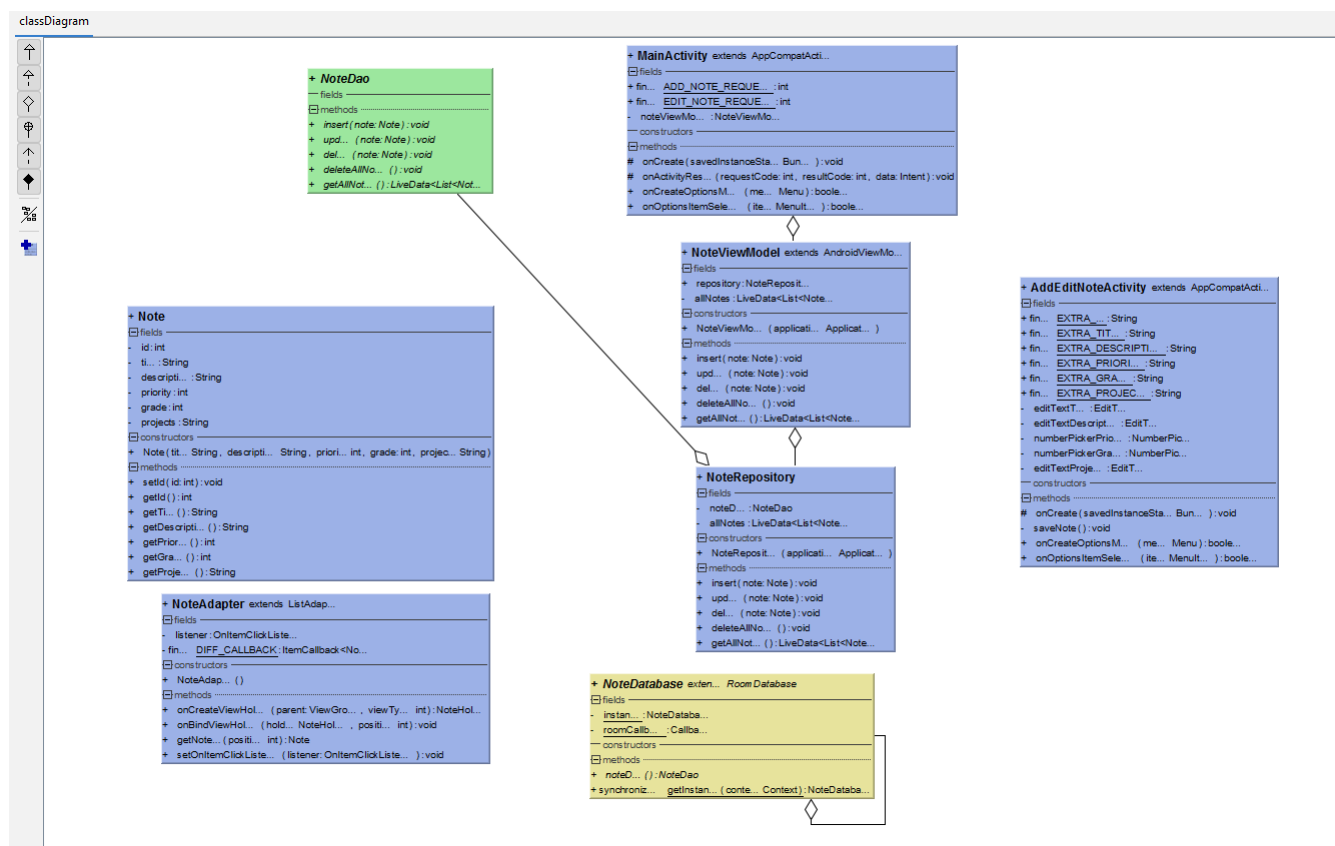
    public void deleteAllNotes() { repository.deleteAllNotes(); }

    public LiveData<List<Note>> getAllNotes() { return allNotes; }
}
```

Εικόνα 28. Χρήση LiveData στην κλάση NoteViewModel

4.1.6 Διάγραμμα κλάσης

Στο παρακάτω διάγραμμα κλάσης απεικονίζεται το **MVVM** μοντέλο . Από το διάγραμμα φαίνεται πώς από το **NoteRepository** τα δεδομένα κατευθύνονται στο **NoteViewModel** και στην συνέχεια στο **MainActivity**.



Εικόνα 29. Διάγραμμα κλάσης Unipi Grades

4.2 Τρόπος σύνδεσης εφαρμογής και ιστοσελίδας

Για την καλύτερη κατανόηση της αρχιτεκτονικής της ιστοσελίδας **Unipi grades**, που θα είναι το επόμενο κεφάλαιο, είναι απαραίτητη η διερεύνηση της σύνδεσης μεταξύ της εφαρμογής και της ιστοσελίδας.

Όταν αποθηκεύονται τα δεδομένα της εφαρμογής στην **Sqlite** βάση δεδομένων, μέσω της αρχιτεκτονικής **MVVM** και **android jetpack components**, τα δεδομένα αποθηκεύονται τοπικά στην συσκευή, έτσι μόνο με αυτόν τον τρόπο δεν μπορούν να γίνουν διαθέσιμα για την ιστοσελίδα. Έτσι χρησιμοποιείται μια **Cloud** βάση δεδομένων η **Firebase**, αφού αποθηκευτούν πρώτα τα δεδομένα τοπικά στην **Sqlite**.

```
save_toFirebase.setOnClickListener(view -> {  
  
    allNotess= noteViewModel.getAllNotes();  
    // GETs user unique ID  
  
    myRef.setValue(allNotess).addOnCompleteListener((task) -> {  
        if (task.isSuccessful()) {  
            Toast.makeText(context: MainActivity.this, text: "Data Successfully saved!", Toast.LENGTH_SHORT).show();  
        } else if (task.getException() instanceof FirebaseAuthUserCollisionException) {  
            Toast.makeText(context: MainActivity.this, text: "Data not saved!", Toast.LENGTH_SHORT).show();  
        }  
    });  
});
```

Εικόνα 30. Τρόπος σύνδεσης εφαρμογής και ιστοσελίδας

Τα δεδομένα αποθηκεύονται σαν αντικείμενο στην Firebase με κλειδί το ID του χρήστη. Στην συνέχεια τα δεδομένα είναι διαθέσιμα στο διαδίκτυο και μπορούν να ανακτηθούν από την ιστοσελίδα Unipi grades.

4.3 Ανάλυση αρχιτεκτονικής της ιστοσελίδας Unipi Grades

4.3.1 Εισαγωγή

Η ενότητα αυτή εξετάζει την αρχιτεκτονική της ιστοσελίδας. Πιο συγκεκριμένα θα γίνει ανάλυση πως τα κομμάτια του κώδικα της HTML, CSS και JS λειτουργούν, καθώς και με ποιο τρόπο συνδέονται μεταξύ τους. Η ιστοσελίδα αποτελείται από 4 σελίδες, τις **Main.html**, **login.html**, **search_course.html** και την **search_user.html**, η κάθε μία από αυτές τις σελίδες περιέχει τον δικό της ξεχωριστό κώδικα **javascript**. Πιο συγκεκριμένα θα αναλυθούν τα κομμάτια **javascript** που απευθύνονται στην λειτουργικότητα της ιστοσελίδας.

4.3.2 Λειτουργία login

Η λειτουργία Login εφαρμόζεται με την χρήση του **FirestoreUI**, το οποίο είναι μια βιβλιοθήκη που είναι χτισμένη πάνω από το **SDK Firebase Authentication SDK** που παρέχει ροές UI drop-in. Το **FirestoreUI** παρέχει πλεονεκτήματα, όπως πολλαπλοί χρήστες - ροές σύνδεσης για email / password και σύνδεση ηλεκτρονικού ταχυδρομείου με ασφαλή σύνδεση λογαριασμών μεταξύ διαφορετικών παροχών ταυτότητας.

Με την χρήση του **Authentication** της **Firebase** στο παρακάτω κομμάτι κώδικα **javascript**, καλείται η **signInSuccessWithAuthResult** για να διασταυρώσει τα στοιχεία που έχει εισάγει ο χρήστης και αν υπάρχει στην βάση δεδομένων τότε μεταφέρεται στην **Main.html**.

```
1 (function(){
2   // Initialize the FirebaseUI Widget using Firebase.
3   var ui = new firebaseui.auth.AuthUI(firebase.auth());
4   var uiConfig = {
5     callbacks: {
6       signInSuccessWithAuthResult: function(authResult, redirectUrl) {
7         // User successfully signed in.
8         // Return type determines whether we continue the redirect automatically
9         // or whether we leave that to developer to handle.
10        return true;
11      },
12      uiShown: function() {
13        // The widget is rendered.
14        // Hide the loader.
15        document.getElementById('loader').style.display = 'none';
16      }
17    },
18    // Will use popup for IDP Providers sign-in flow instead of the default, redirect.
19    signInFlow: 'popup',
20    signInSuccessUrl: 'main.html',
21    signInOptions: [
22      // Leave the lines as is for the providers you want to offer your users.
23      //firebase.auth.GoogleAuthProvider.PROVIDER_ID,
24      // firebase.auth.FacebookAuthProvider.PROVIDER_ID,
25      // firebase.auth.TwitterAuthProvider.PROVIDER_ID,
26      //firebase.auth.GithubAuthProvider.PROVIDER_ID,
27      firebase.auth.EmailAuthProvider.PROVIDER_ID,
28      // firebase.auth.PhoneAuthProvider.PROVIDER_ID
29    ],
30    // Terms of service url.
31    tosUrl: '<your-tos-url>',
32    // Privacy policy url.
33    privacyPolicyUrl: 'main.html'
34  };
35
36  // The start method will wait until the DOM is loaded.
37  ui.start('#firebaseui-auth-container', uiConfig);
38
39 })()
40
41
```

Εικόνα 31.Κείμενο κώδικα Firebase.js

4.3.3 Λειτουργία main

Στην κύρια σελίδα της ιστοσελίδας **Main.html** εμφανίζονται τα στατιστικά στοιχεία όλων των χρηστών. Αυτό γίνεται χρησιμοποιώντας 2 πίνακες με id **table_body** και **table_body2**. Στον κώδικα **Javascript** καλείται ένα στιγμιότυπο της βάσης δεδομένων για να μετρηθούν πόσοι χρήστες υπάρχουν. Στην συνέχεια αποθηκεύονται σε μεταβλητές, επαναληπτικά, τα στατιστικά για κάθε χρήστη που χρειάζονται για την δημιουργία γενικών στατιστικών στοιχείων.

```
25
26
27 → firebase.database().ref('User_data').once('value').then(function(snapshot) {
28
29     snapshot.forEach(function(snapshot1) {
30
31         var uid= snapshot1.key;
32         //saveState.uid = uid;
33
34         firebase.database().ref('User_data').on("value", function(snapshot) {
35             tsnapshot.numChildren()
36         });
37     });
38
39
40
41
42
43
44 → for (var i = 0; i < snapshot1.val().value.length ; i++) {
45
46     const dbRefList = dbRefObject.child(uid).child('value').child(i);
47
48
49
50     dbRefList.once('value' , function(snapshot)
51     {
52         if (snapshot.exists()) {
53             k=k+1;
54             var flag = 0;
55             var description = (snapshot.val().description)
56             var ects = (snapshot.val().ects)
57             var grade = (snapshot.val().grade)
58             var projects = (snapshot.val().projects)
59             var semester = (snapshot.val().semester)
60             var title = (snapshot.val().title)
61
62
63             saveStateTitle[p]=title;
64             saveStateGrades[p]= grade;
65
66
```

Εικόνα 32. Κείμενο κώδικα main.js , μέτρηση χρηστών

Οι μεταβλητές αυτές (εικόνα 32) γίνονται **append** στους πίνακες και έτσι η ιστοσελίδα λειτουργεί όσο πιο δυναμικά γίνεται.

```
const tr = document.createElement('tr');
const td1 = document.createElement('td');
const td2 = document.createElement('td');
const td3 = document.createElement('td');
const td4 = document.createElement('td');
const td5 = document.createElement('td');
const td6 = document.createElement('td');
const td9 = document.createElement('td');

td1.innerText = title;
uList.appendChild(td1);
td2.innerText = description;
uList.appendChild(td2);
td3.innerText = grade;
uList.appendChild(td3);
td4.innerText= semester;
uList.appendChild(td4);
td5.innerText= projects;
uList.appendChild(td5);
td6.innerText = ect5;
uList.appendChild(td6);
td9.innerText=uid;
uList.appendChild(td9);
uList.appendChild(tr);

p=p+1;
```

Εικόνα 33. Κείμενο κώδικα main.js , δημιουργία κόμβων των πινάκων

4.3.4 Λειτουργία search_user

Σε αυτήν την σελίδα οι χρήστες έχουν την δυνατότητα να αναζητήσουν τα στατιστικά συγκεκριμένου χρήστη με βάση το κλειδί του χρήστη “**User ID**”. Το κλειδί αυτό είναι το ξεχωριστό κλειδί για κάθε χρήστη στην βάση **Firebase**. Στον κώδικα **Javascript** θα γίνει έλεγχος εάν υπάρχει το κλειδί στην βάση δεδομένων. Εάν υπάρχει, οι μεταβλητές για τα στατιστικά του θα αποθηκευτούν σε ξεχωριστές μεταβλητές, επειδή θα πρέπει να ξεχωρισθούν για να γίνει σύγκριση με τα συνολικά στατιστικά στοιχεία όλων των χρηστών. Εάν το κλειδί δεν υπάρχει, δεν θα εμφανιστεί τίποτα.

```
if (uid==uid123) { ←
    l=l+1;

    title1[l]=title;
    description1[l]=description;
    grade1[l]=grade;
    projects1[l]= projects;
    ect1[l]=ects;
    semester1[l]= semester;
    total_ects = total_ects + ect1;
    avg_grade = avg_grade + grade;
}

p=p+1;
```

Εικόνα 34. Κείμενο κώδικα search.js , έλεγχος αναζήτησης στην βάση firebase

Στην συνέχεια τα στατιστικά στοιχεία του χρήστη που αναζητήθηκε και τα συνολικά στατιστικά στοιχεία όλων των χρηστών της βάσης δεδομένων, θα στοιχηθούν κατάλληλα στους 2 πίνακες με τον ίδιο τρόπο όπως και στην **home**.

```
0 → for (var r = 0; r <= Object.keys(title1).length; r++) {  
1  
2  
3     if (saveStateTitleStats[u]==title1[r]){  
4  
5  
6         const td7 = document.createElement('td');  
7         const td8 = document.createElement('td');  
8         const td10 = document.createElement('td');  
9         const tr1 = document.createElement('tr');  
10        const tr = document.createElement('tr');  
11        const td2 = document.createElement('td');  
12        const td3 = document.createElement('td');  
13        const td4 = document.createElement('td');  
14        const td5 = document.createElement('td');  
15        const td6 = document.createElement('td');  
16        const td1 = document.getElementById('td7');  
17        const td9 = document.getElementById('td8');  
18  
19        td7.innerText = saveStateTitleStats[u];  
20        uList.appendChild(td7);  
21        td2.innerText = description1[r];  
22        uList.appendChild(td2);  
23        td3.innerText = grade1[r];  
24        uList.appendChild(td3);  
25        td4.innerText= semester1[r];  
26        uList.appendChild(td4);  
27        td5.innerText= projects1[r];  
28        uList.appendChild(td5);  
29        td6.innerText = ects1[r];  
30        uList.appendChild(td6);  
31        grades_stats =saveStateGradesStats[u]/saveStateTimesSeenStats[u];  
32        td8.innerText = grades_stats.toFixed(2);  
33        uList.appendChild(td8);  
34        td10.innerText = saveStateTimesSeenStats[u];  
35        uList.appendChild(td10);  
36        uList.appendChild(tr1);  
37  
38        td1.innerText= total_ects;  
39        uList2.appendChild(td1);  
40        avg_grade1= avg_grade/1;  
41        td9.innerText= avg_grade1.toFixed(2);  
42        uList2.appendChild(td9);  
43    }  
44 }  
45 }  
46
```

Εικόνα 35. Κείμενο κώδικα search.js , δημιουργία κόμβων των πινάκων

4.3.5 Λειτουργία search_course

Σε αυτήν την σελίδα οι χρήστες έχουν την δυνατότητα να αναζητήσουν τα στατιστικά ενός συγκεκριμένου μαθήματος βάσει του ονόματος του μαθήματος. Η λειτουργία του **search_course** είναι παρόμοια με αυτήν του **search_user**. Στον κώδικα **Javascript** θα γίνει έλεγχος εάν υπάρχει το μάθημα στην βάση δεδομένων. Εάν υπάρχει, τότε οι μεταβλητές για τα στατιστικά του θα αποθηκευτούν σε ξεχωριστές μεταβλητές, εφόσον θα πρέπει να ξεχωρισθούν για να γίνει σύγκριση με τα συνολικά στατιστικά στοιχεία όλων των χρηστών. Εάν δεν υπάρχει τότε δεν θα εμφανιστεί τίποτα.

```
dbRefList.once('value' , function(snapshot)
{
    if (snapshot.exists()) {
        k=k+1;
        var flag = 0;
        var description = (snapshot.val().description)
        var ects = (snapshot.val().ects)
        var grade = (snapshot.val().grade)
        var projects = (snapshot.val().projects)
        var semester = (snapshot.val().semester)
        var title = (snapshot.val().title)

        saveStateTitle[p]=title;
        saveStateGrades[p]= grade;
```

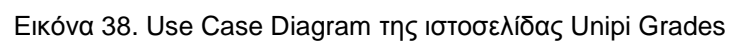
Εικόνα 36. Κείμενο κώδικα search_course.js , έλεγχος αναζήτησης στην βάση firebase

Οπότε με κριτήριο την αναζήτηση του μαθήματος δημιουργούνται τα κελιά του πίνακα όσο πιο δυναμικά γίνεται.

```
    }  
  
    for (var u = 0; u < Object.keys(saveStateTitleStats).length; u++) {  
        for (var r = 0; r <= Object.keys(title1).length; r++) {  
  
            if (saveStateTitleStats[u]==title1[r]){  
  
                const td7 = document.createElement('td');  
                const td8 = document.createElement('td');  
                const td10 = document.createElement('td');  
                const tr1 = document.createElement('tr');  
                const tr = document.createElement('tr');  
                const td2 = document.createElement('td');  
                const td3 = document.createElement('td');  
                const td4 = document.createElement('td');  
                const td5 = document.createElement('td');  
                const td6 = document.createElement('td');  
                const td1 = document.getElementById('td7');  
                const td9 = document.getElementById('td8');  
  
                td7.innerText = saveStateTitleStats[u];  
                uList.appendChild(td7);  
                td2.innerText = description1[r];  
                uList.appendChild(td2);  
                td3.innerText = grade1[r];  
                uList.appendChild(td3);  
                td4.innerText= semester1[r];  
                uList.appendChild(td4);  
                td5.innerText= projects1[r];  
                uList.appendChild(td5);  
                td6.innerText = ects1[r];  
                uList.appendChild(td6);  
                grades_stats =saveStateGradesStats[u]/saveStateTimesSeenStats[u];  
                td8.innerText = grades_stats.toFixed(2);  
                uList.appendChild(td8);  
            }  
        }  
    }  
}
```

Εικόνα 37. Κείμενο κώδικα search_course.js ,δημιουργία κόμβων των πινάκων

Στο παρακάτω **Use case** διάγραμμα φαίνεται πως αλληλοεπιδρούν οι οντότητες μεταξύ τους αλλά και με το σύστημα.



Από το παραπάνω διάγραμμα παρέχονται οι εξής πληροφορίες,

Ο χρήστης(**User**) μπορεί:

- Να κάνει **Login**
- Να κάνει **Logout**

Το σύστημα ταυτοποίησης(**Firebase Authentication Service**) μπορεί:

- Να κάνει έλεγχο στοιχείων όταν κάποιος χρήστης προσπαθεί να κάνει **Login**
- Να δώσει πρόσβαση στην εφαρμογή όταν ο κωδικός είναι έγκυρος
- Να εμφανίσει μήνυμα ανεπιτυχούς εισόδου όταν ο κωδικός δεν είναι έγκυρος

Εφόσον ο χρήστης ταυτοποιηθεί από το σύστημα μπορεί:

- Να κάνει **Search a user**
- Να κάνει **Search a course**
- Να βλέπει να γενικά στατιστικά χρηστών στην **main**
- Να κάνει **Logout**

5. Εργαλεία και Τεχνολογίες

5.1 Γλώσσα Προγραμματισμού

Η **γλώσσα προγραμματισμού** που χρησιμοποιήθηκε για την ανάπτυξη τα εφαρμογής είναι η **Java Standard Edition 8**. Αυτή η γλώσσα παρέχεται από την **Oracle**. Η έκδοση αυτή προσφέρει πολλές λειτουργίες οι οποίες ήταν χρήσιμες για την διεκπεραίωση αυτής της εργασίας. Περισσότερες πληροφορίες για αυτήν την έκδοση μπορούν να βρεθούν στο παρακάτω link:

<https://www.oracle.com/technetwork/java/javase/8-whats-new-2157071.html>

Η **JavaScript** χρησιμοποιήθηκε για την ανάπτυξη της ιστοσελίδας, συχνά συντομευμένο ως JS. Είναι μια γλώσσα προγραμματισμού υψηλού επιπέδου που ερμηνεύεται σύμφωνα με την προδιαγραφή **ECMAScript**. Η **JavaScript** έχει σύντομη σύνταξη, δυναμική πληκτρολόγηση, προσανατολισμό αντικειμένων με βάση πρωτότυπο και λειτουργίες πρώτης τάξης. Περισσότερες πληροφορίες για αυτήν την έκδοση μπορούν να βρεθούν στο παρακάτω link:

<https://javascript.info/js>



Εικόνα 39. Λογότυπο Java

5.2 Βάση Δεδομένων

Η βάση δεδομένων που χρησιμοποιήθηκε για την ανάπτυξη αυτού του Project είναι το **Firebase Database**. Το προϊόν διατίθεται από την **Firebase, Inc** η οποία είναι μία θυγατρική της **Google**. Αυτή η βάση δεδομένων προσφέρει μία πολύ γρήγορη και εύκολη λύση για την εφαρμογή Android που υλοποιήθηκε. Η έκδοση που χρησιμοποιήθηκε στην εφαρμογή είναι η **10.0.1**.



Εικόνα 40. Λογότυπο βάσης Firebase

5.3 Περιβάλλον

Το περιβάλλον που χρησιμοποιήθηκε για την ανάπτυξη της εφαρμογής είναι το **Android Studio**. Το Android Studio είναι ένα λογισμικό βασισμένο στο **JetBrains** και το **IntelliJ IDEA**. Το λογισμικό αυτό παρέχεται από την **Google** και είναι αποκλειστικά για προγραμματισμό σε **Android**. Η έκδοση που χρησιμοποιήθηκε για την υλοποίηση αυτού του Project είναι η **3.2.1**.

Για την ανάπτυξη της ιστοσελίδας χρησιμοποιήθηκε το **Sublime Text** όπου είναι ένας επεξεργαστής πηγαίου κώδικα πολλαπλών πλατφορμών. Υποστηρίζει εγγενώς πολλές γλώσσες προγραμματισμού και γλώσσες σήμανσης για την ιστοσελίδα Unipi Grades χρησιμοποιήθηκε για την ανάπτυξη κώδικα HTML , CSS ,JS. Η έκδοση που χρησιμοποιήθηκε για την υλοποίηση αυτού του Project είναι η **3.1.1**.



Εικόνα 41 . Λογότυπο Android Studio

5.4 Plugins

Τα plugins που χρησιμοποιήθηκαν στην συγκεκριμένη εφαρμογή και ιστοσελίδα είναι τα παρακάτω:

- SimpleUML για την δημιουργία των class diagrams.
- LucidChart για την δημιουργία του Use Case Diagram.
- Firebase-core version 10.0.1 για την εισαγωγή των βασικών λειτουργιών του Firebase Database
- Firebase-client-android version 2.5.2
- Firebase-ui-database version 0.4.0

6. Συμπεράσματα και Μελλοντικές επεκτάσεις

Στην παρούσα εργασία αναπτύχθηκε μία εφαρμογή δημιουργίας και αποθήκευσης βαθμών, που ανταποκρίνονται σε μαθήματα που παρακολουθεί ο χρήστης. Ο χρήστης έχει επίσης την δυνατότητα να εισάγει περιφερειακές πληροφορίες για τα μαθήματα, τα οποία συγχρονίζονται με μια ιστοσελίδα που ανταποκρίνεται με την εφαρμογή.

Βασικό στοιχείο της εργασίας ήταν η εξοικείωση με τους τρόπους ανάπτυξης και δομής μίας Android εφαρμογής καθώς και ιστοσελίδας που να ανταποκρίνεται με εκείνη.

Μέσα από την ανάπτυξη της εφαρμογής αποκτήθηκαν οι απαραίτητες γνώσεις, συγκεκριμένα έγινε εξερεύνηση της πλατφόρμας Android, με παράλληλη εξερεύνηση των επιπλέον εφαρμογών και λειτουργιών που διαθέτει. Χρησιμοποιήθηκαν βιβλιοθήκες Android architecture components σε MVVM μοντέλο. Επίσης έγινε μεγάλη κατανόηση των αντικειμενοστραφών γλωσσών προγραμματισμού Java και Javascript αφού ήταν οι γλώσσες υλοποίησης της εφαρμογής και της ιστοσελίδας αντίστοιχα.

Επιπλέον, υπήρξε μεγάλη τριβή με αποδοτικούς τρόπους υλοποίησης προγραμμάτων χρησιμοποιώντας κληρονομικότητα, interfaces, models, abstract και classes.

Όσο αναφορά στις μελλοντικές επεκτάσεις της εφαρμογής και της ιστοσελίδας είναι γεγονός ότι υπάρχουν περιθώρια βελτίωσης. Κάποιες επιπλέον λειτουργίες μπορούν να προστεθούν, όπως για παράδειγμα οι χρήστες να έχουν την δυνατότητα διαμόρφωσης ενός ημερολογίου για τις ώρες μαθημάτων, εργαστηρίων και εξετάσεων όπως και στην εφαρμογή Grade Tracker Pro. Τέλος, θα ήταν πολύ χρήσιμο η εφαρμογή να υποστηρίζει ξένες γλώσσες ώστε να στοχεύει σε χρήστες εκτός Ελλάδας.

7. Βιβλιογραφία

Βιβλία:

- [1] David Griffiths and Dawn Griffiths ,*"Head First Android Development: A Brain-Friendly Guide"* , 2015

Links:

- [2]"Android Jetpack", <https://developer.android.com/jetpack>
[3]"Android", <https://www.android.com/>
[4]"Firebase", <https://firebase.google.com/>
[5]"Javascript", <https://www.javascript.com/>

Άρθρα :

- [6] Model-View-ViewModel (MVVM) Explained by Jeremy Likness
, <https://www.wintellect.com/model-view-viewmodel-mvvm-explained/>
[7] Introduction to Android Jetpack, Ιούλιος 2018, <https://www.raywenderlich.com/5376-introduction-to-android-jetpack>
[8]Thanos Karpouzis,Android Architecture, Αύγουστος 2015, <https://android.jlelse.eu/android-architecture-2f12e1c7d4db>
[8]Android Jetpack Architecture Components: Getting Started ,Οκτώβριος 2018,
<https://www.raywenderlich.com/6729-android-jetpack-architecture-components-getting-started>
[9] Mikael Konutgan, Firebase Tutorial: Getting Started , Μάιος 2018,
<https://www.raywenderlich.com/3-firebase-tutorial-getting-started>
[10]Peter Ekene Eze, Add Firebase Authentication to your App in 7minutes , Οκτώβριος 2017,
<https://medium.com/@peterekeneeze/add-firebase-authentication-to-your-app-in-7minutes-c13df58994bd>
[11]Clive Sargent, Using a Firebase Realtime Database in your app, Αύγουστος 2017,
<https://www.101apps.co.za/index.php/item/182-firebase-realtime-database-tutorial.html>

8. Παράρτημα

8.1 Γλωσσάριο

Ο παρακάτω πίνακας δίνει τις μεταφράσεις όρων που ήταν στα αγγλικά μεταφρασμένους στα ελληνικά.

Αγγλικά	Ελληνικά
Mobile	Κινητό
Login	Είσοδος
Update	Εκσυγχρόνιση
Smartphones	Έξυπνο κινητό τηλέφωνο
Delete	Διαγράψω
Email	Ηλεκτρονικό Ταχυδρομείο
Password	Κωδικός
Reset	Επαναφορά
Register	Εγγραφή
Home	Αρχική Σελίδα
User Manual	Εγχειρίδιο Χρήστη
Schedule	Χρονοδιάγραμμα
Sign up	Εγγραφή
Type	Τύπος
Note	Σημείωση
Framework	Πλαίσιο Εργασίας
Object	Αντικείμενο
Grade	Βαθμός
Next	Επόμενο
Model	Μοντέλο
View	Προβολή

View-Model	Μοντέλο Προβολής
User ID	Ταυτότητα Χρήστη
Interface	Διεπαφή
Activity	Δραστηριότητα
Layers	Επίπεδα
Behavioral	Συμπεριφορικό
Abstract	Αφηρημένο
Course	Μάθημα
Fragment	Τεμάχιο
Textbox	Πλαίσιο Κειμένου
Average Grade	Μέσος Βαθμός
Databinding	Δέσμευση δεδομένων
Observer	Παρατηρητής
Times Seen	Φορές εμφάνισης
Search User	Αναζήτηση Χρήστη
Search Course	Αναζήτηση Μαθήματος
RecyclerView	Ανακυκλωμένη Όψη
Bitmap	Χάρτης των Μπιτ
Cloud	Σύννεφο
Drop-in	Πτώση
Append	Προσαρτώ
Use Case Diagram	Διαγράμματα περιπτώσεων χρήσης
UI	Διεπαφή χρήστη
Sent	Απεσταλμένα