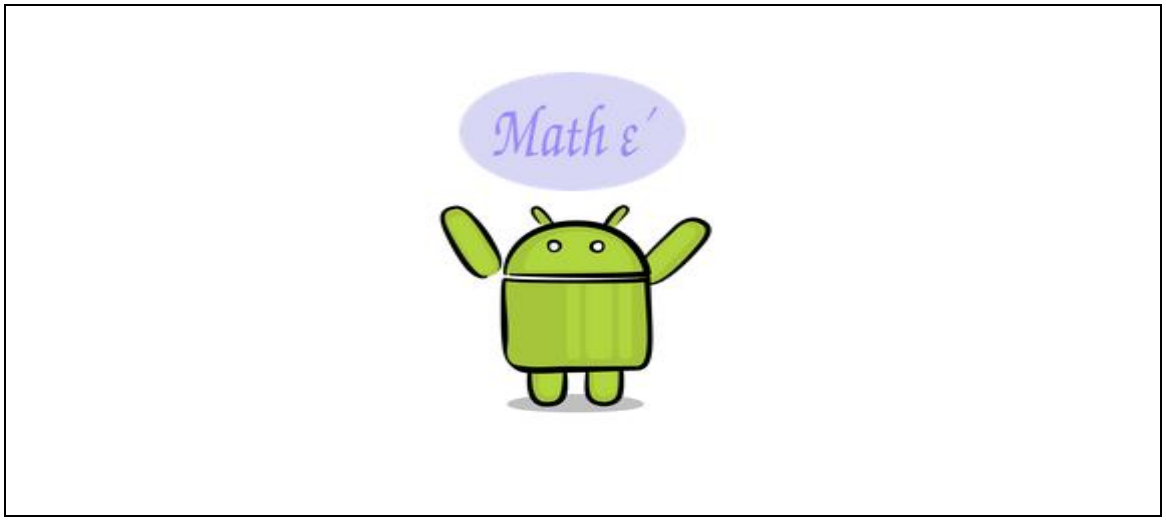
ΔΙΕΘΝΕΣ
ΠΑΝΕΠΙΣΤΗΜΙΟ
ΤΗΣ ΕΛΛΑΔΟΣ

ΣΧΟΛΗ ΜΗΧΑΝΙΚΩΝ

ΤΜΗΜΑ ΜΗΧΑΝΙΚΩΝ ΠΛΗΡΟΦΟΡΙΚΗΣ
ΚΑΙ ΗΛΕΚΤΡΟΝΙΚΩΝ ΣΥΣΤΗΜΑΤΩΝ

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

«ΑΝΑΠΤΥΞΗ ANDROID ΕΦΑΡΜΟΓΗΣ ΓΙΑ ΤΗΝ ΥΠΟΒΟΗΘΗΣΗ ΤΗΣ ΔΙΔΑΣΚΑΛΙΑΣ ΤΩΝ ΜΑΘΗΜΑΤΙΚΩΝ ΤΗΣ Ε' ΔΗΜΟΤΙΚΟΥ»



Του φοιτητή
Ζώτη Αγγελή
Αρ. Μητρώου: 164626

Επιβλέπων:
Ιορδάνης Κιοσκερίδης

Βαθμίδα Καθηγητής

Ημερομηνία Σεπτέμβριος 2023

Τίτλος Δ.Ε.: Ανάπτυξη android εφαρμογής για την υποβοήθηση της διδασκαλίας των μαθηματικών της Ε' Δημοτικού

Κωδικός Δ.Ε. 22304

Ονοματεπώνυμο φοιτητή: Ζώτη Αγγελή

Ονοματεπώνυμο εισηγητή: Ιορδάνης Κιοσκερίδης

Ημερομηνία ανάληψης Δ.Ε.: 27/10/2022

Ημερομηνία περάτωσης Δ.Ε.: 14/9/2023

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

«Στον Πατέρα μου»

Πρόλογος

Η εκπαίδευση αποτελεί το θεμέλιο για την ανάπτυξη και την εξέλιξη της κοινωνίας μας. Στον σύγχρονο κόσμο, η τεχνολογία αναδεικνύεται ως ισχυρό εκπαιδευτικό εργαλείο, ανοίγοντας νέους ορίζοντες στην προσέγγιση της μάθησης.

Η παρούσα εργασία συνδυάζει την πρωτοποριακή τεχνολογία των Android εφαρμογών με την παιδαγωγική αφοσίωση για τη διδασκαλία των μαθηματικών στην ε΄ τάξη του δημοτικού.

Επιπλέον, μέσω ανταγωνισμού, παιχνιδιού και αλληλεπίδρασης μεταξύ των μαθητών, η εφαρμογή παρέχει έναν ενθαρρυντικό κίνητρο για τη βελτίωση των επιδόσεων τους στα μαθηματικά.

Τέλος, διασκεδάζοντας με την διαδικασία, οι μαθητές ανακαλύπτουν έναν πιο ενδιαφέροντα και εναλλακτικό τρόπο να αντιμετωπίζουν αυτό το σημαντικό μάθημα.

Περίληψη

Η παρούσα πτυχιακή εργασία αναφέρεται στο σχεδιασμό και στην ανάπτυξη εφαρμογής για κινητές συσκευές, με σκοπό τη βελτίωση της εκπαιδευτικής διαδικασίας στα δημοτικά σχολεία με χρήση των νέων τεχνολογιών. Η εφαρμογή από τη μεριά των εκπαιδευτικών (καθηγητών) επιταχύνει και βελτιώνει της διαδικασία της αξιολόγησης των μαθητών και βοηθάει στη καλύτερη οπτικοποίηση της απόδοσης των μαθητών ανά ενότητα και ανά μάθημα, ώστε να έχουν μια καλύτερη εικόνα για την πορεία τους. Επίσης, από τη μεριά των μαθητών, επειδή αλληλεπιδρούν με την εφαρμογή, είναι πιο ελκυστική η διαδικασία εκμάθησης για τους μαθητές και παρέχεται ένας καλύτερος προσανατολισμός ως αναφορά για την πρόσβαση στη γνώση. Η εφαρμογή διαθέτει μια τοπική βάση όπου αποθηκεύονται οι χρήστες της εφαρμογής και οι επιδόσεις τους ανά ενότητα, αλλά διαθέτει και τα κατάλληλα εργαλεία για να μπορεί να κατευθύνει τους μαθητές στο επιθυμητό εκπαιδευτικό υλικό που διατίθεται στο διαδίκτυο.

Παρουσιάζονται και συγκρίνονται οι αντίστοιχες εφαρμογές που εξυπηρετούν τον ίδιο σκοπό.

Τέλος, γίνεται μια ανάλυση της δομής της εφαρμογής και των τεχνολογιών που χρησιμοποιήθηκαν κατά την διάρκεια της ανάπτυξης.

«DEVELOPMENT OF AN ANDROID APPLICATION TO SUPPORT THE TEACHING OF 5TH GRADE MATHEMATICS»

«ZOTIS ANGELIS»

Abstract

The present bachelor's thesis concerns the design and development of a mobile application aimed at enhancing the educational process in elementary schools through the use of new technologies. The application, from the educators' perspective, accelerates and improves the assessment process of students, aiding in better visualization of their performance by unit and subject, providing them with a clearer understanding of their progress. Additionally, from the students' perspective, as they interact with the application, it makes the learning process more engaging and offers better guidance in accessing knowledge. Our application possesses a local database where user information and their performance by unit are stored, but also provides the necessary tools to direct students to the desired educational material available on the internet. Comparable applications serving the same purpose are presented and compared. Finally, an analysis of the application's structure and the technologies utilized during development is provided.

Ευχαριστίες

Αρχικά, θα ήθελα να εκφράσω τις ειλικρινές μου ευχαριστίες προς τον Επιβλέποντα καθηγητή μου κ. Ιορδάνη Κιοσκερίδη για την πολύτιμη καθοδήγηση, την εμπιστοσύνη και τη στήριξή του κατά τη διάρκεια της πτυχιακής μου εργασίας. Οι συμβουλές του ήταν κρίσιμες για την ολοκλήρωση αυτού του έργου.

Επιπλέον, θέλω να ευχαριστήσω την οικογένεια μου και ιδιαίτερα τον αγαπημένο μου πατέρα, την αγαπημένη αδερφή μου και τον σύζυγό της. Η αμέριστη στήριξη που μου πρόσφεραν καθ' όλη τη διάρκεια αυτής της πορείας ήταν πολύτιμη. Οι συμβουλές, οι λέξεις καθησυχασμού και η αγάπη τους με έκαναν να νιώθω δυνατός και ανυπέρβλητος.

Τέλος, δεν μπορώ παρά να ευχαριστήσω την αγαπημένη μου, η οποία με την αφοσίωση και την αγάπη της με στήριξε σε κάθε βήμα της πορείας μου. Οι στιγμές που μοιραστήκαμε και οι όμορφες στιγμές που αναμφίβολα θα έρθουν στο μέλλον θα μείνουν αξέχαστες.

Σας ευχαριστώ όλους για όλα, και είμαι ευγνώμον που έχω ανθρώπους όπως εσείς στη ζωή μου. Είμαι βέβαιος ότι η αγάπη και η στήριξή σας θα με συνοδεύουν πάντα.

Περιεχόμενα

Κατάλογος εικόνων

ΚΕΦΑΛΑΙΟ 1 : ΕΙΣΑΓΩΓΗ

1.1 Σκοπός της πτυχιακής εργασίας

Σκοπός αυτής της πτυχιακής εργασίας είναι η ανάπτυξη μιας κινητής εφαρμογής για την βελτίωση της εκπαιδευτικής διαδικασίας στα δημοτικά σχολεία. Η εφαρμογή μας σκοπεύει την διευκόλυνση των καθημερινών αναγκών των εκπαιδευτικών αλλά βοηθάει και στην εκπαιδευτική διαδικασία προσφέροντας τις δυνατότητες των νέων τεχνολογιών στην διαδικασία της εκμάθησης. Η εφαρμογή μας αναπτύχθηκε για το λειτουργικό σύστημα Android. Οι μαθητές μέσω της εφαρμογής μπορούν να έχουν ένα καλύτερο προσανατολισμό για την πρόσβαση στο κατάλληλο εκπαιδευτικό υλικό .

1.2 Η μεθοδολογία

Ο πρωταρχικός σκοπός της εφαρμογής μας είναι να διευκολύνει την εκπαιδευτική διαδικασία αλλά και να κάνει πιο δυνατή την επικοινωνία μεταξύ των εκπαιδευτικών , των μαθητών αλλά και των γονέων. Βοηθάει τους εκπαιδευτικούς να έχουν μια καλύτερη εικόνα για την επίδοση των μαθητών, να εφαρμόζουν την διαδικασία της ανατροφοδότησης (εξέταση) των μαθητών με πιο γρήγορο και αυτοματοποιημένο τρόπο, να τραβήξουν το ενδιαφέρον των μαθητών για το διδασκόμενο αντικείμενο αλλά να έχουν οι γονείς μια ενημέρωση για την πορεία των παιδιών τους στο σχολείο.

Παράλληλα θα γίνει έρευνα και μελέτη των παρόμοιων εφαρμογών που υπάρχουν παγκοσμίως οι οποίες εκπληρούν τους ίδιους στόχους και θα προστεθούν επιπλέον λειτουργίες που θα διακρίνουν την εφαρμογή μας από τις υπόλοιπες. Κατά την ανάπτυξη της εφαρμογής θα γίνουν αρκετές βελτιώσεις διότι μπορεί να προκύπτουν καινούριες ανάγκες και να δοθούν καινούριες λύσεις με σκοπό τη βελτίωση των αρχικών σχεδίων. Μετά την ολοκλήρωση της εφαρμογής, αφού ξεπεράστηκαν κάποια προβλήματα, έγινε η αξιολόγηση του συστήματος. Στην αξιολόγηση αναδεικνύονται τα δυνατά όσο και τα αδύναμα σημεία της εφαρμογής μας. Τέλος, ο στόχος μας μετά την αξιολόγηση να προκύψουν παρατηρήσεις και να προετοιμαστεί το έδαφος για μελλοντικές επεκτάσεις.

1.3 Η Δομή της πτυχιακής

Η δομή της πτυχιακής μας προβάλει την διαδικασία και τα στάδια που ακολουθήσαμε κατά την διάρκεια της δημιουργίας της πτυχιακής

- Στο **πρώτο κεφάλαιο** κάνουμε μια εισαγωγή αναφέροντας τον σκοπό της εφαρμογής, ποια συνεισφορά θα έχει στην εκπαιδευτική διαδικασία και ποιες είναι οι δυσκολίες που αντιμετωπίσαμε.
- Στο **δεύτερο κεφάλαιο** αναλύουμε το τι είναι το λειτουργικό σύστημα Android, κάνουμε μια γενική επισκόπηση και ιστορική αναδρομή των εφαρμογών για έξυπνες συσκευές και κινητά τηλέφωνα (smartphones) και κάνουμε μια αναφορά στο τεχνολογικό υπόβαθρο του λειτουργικού συστήματος.
- Στο **τρίτο κεφάλαιο** γίνεται η ανάλυση των τεχνολογιών που χρησιμοποιούνται στις κινητές εφαρμογές για τις βάσεις δεδομένων, σύγκριση των διαθέσιμων τεχνολογιών και ο λόγος που επιλέξαμε τη συγκεκριμένη τεχνολογία για την εφαρμογή μας. Επίσης, θα αναφερθούμε στις υπηρεσίες που χρησιμοποιούνται για την ταυτοποίηση των χρηστών και προσπέλασης των δεδομένων από έναν διακομιστή.
- Στο **τέταρτο κεφάλαιο** γίνεται η παρουσίαση των βασικών λειτουργιών της εφαρμογής μας με λεπτομέρεια και παρουσιάζεται η τελική μορφή της εφαρμογής. Παρουσιάζονται κάποια στιγμιότυπα οθόνης (screenshots) που πιστοποιούν τη σωστή λειτουργία της εφαρμογής μας.
- Στο **πέμπτο κεφάλαιο** θα γίνει μια αναφορά στις αδυναμίες που έχει η εφαρμογή μας και θα αναφερθούμε στις μελλοντικές επεκτάσεις και ιδέες που θα βοηθήσουν στην αποτελεσματικότερη λειτουργία της εφαρμογής μας

ΚΕΦΑΛΑΙΟ 2: ΑΡΧΙΤΕΚΤΟΝΙΚΗ ΚΑΙ ΕΦΑΡΜΟΓΕΣ ΤΟΥ ΛΕΙΤΟΥΡΓΙΚΟΥ ΣΥΣΤΗΜΑΤΟΣ ANDROID

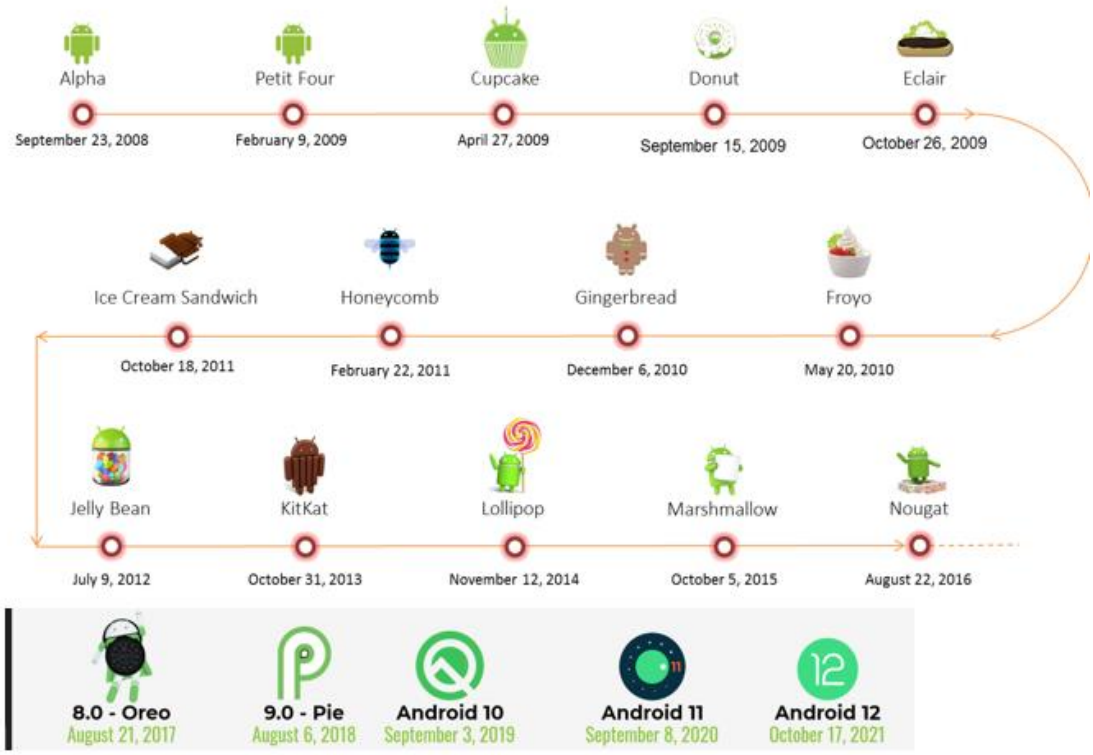
2.1 Ορισμός του λειτουργικού συστήματος Android

Το λειτουργικό σύστημα Android είναι ένα λειτουργικό σύστημα που έχει αναπτυχθεί για κινητές εφαρμογές και τρέχει επάνω στον πυρήνα του Linux (Λειτουργικό σύστημα ανοιχτού κώδικα). Μπορούμε να το παρομοιάσουμε με τα λειτουργικά συστήματα όπως είναι τα Windows, ή το MacOS (Apple). Με τη βοήθεια των ειδικών εργαλειοθηκών ανάπτυξης συστήματος λογισμικού (Software Development Kit) οι προγραμματιστές εφαρμογών έχουν την δυνατότητα να δημιουργούν πρωτοποριακές εφαρμογές. Παρουσιάστηκε πρώτη φορά στις 5 Νοεμβρίου 2007.

Το Android παρέχει τη δυνατότητα στους προγραμματιστές να αναπτύξουν εφαρμογές με χρήση της γλώσσας προγραμματισμού Java ή Kotlin, ελέγχοντας τη συσκευή μέσω των κατάλληλων βιβλιοθηκών που έχουν αναπτυχθεί από την Google.

Η πλατφόρμα Android είναι ένα λειτουργικό σύστημα, ένα ενδιάμεσο λογισμικό που επιτρέπει στους προγραμματιστές τη δημιουργία των νέων και πρότυπων εφαρμογών που αξιοποιούν όλες τις δυνατότητες μιας συσκευής (όπως σύνδεση στο διαδίκτυο ή αποθήκευση στην τοπική βάση).

2.2 Η ιστορική επισκόπηση του λειτουργικού συστήματος



Εικόνα 2.1: Οι διάφορες εκδόσεις του λειτουργικού συστήματος Android ανά χρόνο εμφάνισης στην αγορά

2.2.1 "Android 4.3 "

Το Android 4.3 αποτελεί τη νέα πιο 'Ευχάριστη' αναβάθμιση του Jelly Bean και το συναντάμε κυρίως σε Nexus κινητά και ταμπλέτες (Tablets). Ορισμένα νέα χαρακτηριστικά που διαθέτει αυτή η έκδοση είναι :

- 3D ρεαλιστικά γραφικά από την OpenGL ES 3.0 που κάνουν τις εφαρμογές με παιχνίδια ακόμα πιο διασκεδαστικές.
- Μεγάλη βελτίωση στην απόδοση του ήχου.
- Γραφικά τελευταίας γενιάς από την OPEN GL 3.0
- Αυτόματη προβολή της οθόνης εργασίας του tablet μας σε τηλεόραση για το Nexus 7 και το Nexus 10.
- Χρήση της δυνατότητας 'περιορισμένου προφίλ', όπου αν θέσουμε το κινητό ή το τάμπλετ σε μια συγκεκριμένη λειτουργία , περιορίζει την πρόσβαση σε διάφορες εφαρμογές και τα περιεχόμενα.
- Bluetooth Smart, που ελαχιστοποιεί τη χρήση της ενέργειας κατά τη μέτρηση και διαβίβαση δεδομένων για αισθητήρες γυμναστικής.

2.2.2 "Android 4.4 (KitKat)"

Εικόνα 2.2: Android Kit Kat

Το Android 4.4, με ψευδώνυμο KitKat παρουσιάστηκε την πρώτη φορά το Σεπτέμβριο του 2013.

Το Android 4.4 είχε αρκετές βελτιώσεις στα γραφικά ,με εκλεπτυσμένα εικονίδια και γενικά μια επανασχεδιασμένη αρχική οθόνη.
Τα χαρακτηριστικά που προστέθηκαν είναι :

- Αναλόγως με τις ειδοποιήσεις αλλάζει η εμφάνιση των μπαρών.
- Βελτιώσεις στο Google Map
- Ιδιαίτερες βελτιώσεις στην ασύρματη υποστήριξη.

Η πρώτη συσκευή που κυκλοφόρησε με το Android KitKat είναι η LG-made Nexus

2.2.3 “Android 8.0 (Oreo) αποτελεί την όγδοη έκδοση του γνωστικού λειτουργικού”

Εικόνα 2.3: Android Oreo

Η έκδοση αυτή συνεχίζει την παράδοση να δίνονται ονόματα “γλυκών”, και το Μάρτιο του 2017 κυκλοφόρησε η έκδοση το Android Oreo που πήρε το όνομα από τη γνωστή μάρκα μπισκότων.
Τα σημαντικότερα χαρακτηριστικά της έκδοσης Oreo είναι τα παρακάτω:

- **Λειτουργία Picture in Picture(PiP):** Με λειτουργία αυτή μπορούμε να χρησιμοποιούμε δύο εφαρμογές ταυτόχρονα, ελαχιστοποιώντας τη μία σε ένα μικρό τετράγωνο στο κάτω μέρος της οθόνης. Έτσι μπορούμε για παράδειγμα να ακούσουμε ένα τραγούδι στο Youtube και συγχρόνως να βρούμε κάποια επαφή.
- **Γρηγορότερη εκκίνηση συσκευής**: Συνήθως ο χρόνος εκκίνησης της συσκευής εξαρτάται από το hardware της συσκευής αλλά η έκδοση αυτή έχει κάνει σημαντικές βελτιώσεις.
- **Βελτιωμένη χρήση μπαταρίας:** Το Android 8.0 βελτιώνει τη διαχείριση της μπαταρίας της συσκευής, ώστε να κρατάει περισσότερο. Αυτό το καταφέρνει περιορίζοντας τον αριθμό των διεργασιών που τρέχουν στο παρασκήνιο.
- **Έξυπνη επιλογή κειμένου:** Το λειτουργικό σύστημα μπορεί να αναγνωρίζει το κείμενο που επιλέγουμε στη συσκευή και να αποφασίζει ποια εφαρμογή είναι κατάλληλη για αυτή για να γίνει κάποια ενέργεια (π.χ. Αντιγραφή αριθμός τηλεφώνου και εμφάνιση επιλογών κλήσεων).
- **Καλύτερη υποστήριξη Bluetooth για συσκευές ήχου**: Η έκδοση αυτή υποστηρίζει τη χρήση υψηλής ποιότητας ακουστικών και ηχείων που συνδέονται μέσω Bluetooth.
- **Αυτόματο Wi-Fi**: Η έκδοση αυτή του Android δεν απαιτεί την εγκατάσταση μιας ξεχωριστής εφαρμογής για την προστασία από κακόβουλο λογισμικό. Η εφαρμογή VITALS είναι ήδη εγκατεστημένη στην έκδοση αυτή.

2.2.4 “Android 9.0(Pie) ”

Εικόνα 2.4: Android Pie

Τα σημαντικότερα χαρακτηριστικά της έκδοσης αυτής:

- **Νέα γραμμή πλοήγησης:** Τα τρία εικονίδια πλοήγησης που συναντάγαμε μέχρι τώρα στο κάτω μέρος της οθόνης και μετά οποία μπορούμε να πάμε στην κεντρική οθόνη, να γυρίσουμε πίσω και να δούμε τις εφαρμογές που χρησιμοποιήθηκαν πρόσφατα, **αντικαθίστανται τώρα από ένα κουμπί πλοήγησης.**
- **Έλεγχος Μπαταρίας:** Με τη χρήση της τεχνητής νοημοσύνης (AI) προκειμένου να χρησιμοποιείται 'έξυπνα' η μπαταρία της συσκευής**, με τη λειτουργία Adaptive Battery**. Το λειτουργικό "μαθαίνει" ποιες εφαρμογές χρησιμοποιείται συχνότερα κα κλείνει τις υπόλοιπες που τρέχουν στο παρασκήνιο.

- **Έλεγχος φωτεινότητας οθόνης:** Η τεχνολογία μηχανικής μάθησης (machine learning) χρησιμοποιείται για να προσαρμόζεται η φωτεινότητα της συσκευής ανάλογα με το περιβάλλον και τις προτιμήσεις του χρήστη. Η λειτουργία Adaptive Brightness ρυθμίζει τη φωτεινότητα της οθόνης ανάλογα με το περιβάλλον και τις προτιμήσεις.

2.2.5 "Android 10 "

Εικόνα 2.5: Android 10

Η Google μαζί με την νέα έκδοση φέρνει στους χρήστες νέα χαρακτηριστικά, βελτιώσεις και πολυσυζητημένα gestures. Οι χειρονομίες αντικαθιστούν το παλιό σχεδιασμό των κουμπιών και κάνουν πιο εύκολη την πλοήγηση στο περιβάλλον του λειτουργικού ,ιδιαίτερα Full Screen Smartphones.

Πλέον για την πλοήγηση του συστήματος υπάρχουν 3 επιλογές:
Η πρώτη επιλογή είναι η κλασική πλοήγηση με 3 κουμπιά.
Η δεύτερη είναι μια παραλλαγή που προσφέρει πλοήγηση με 2 κουμπιά.
Η τρίτη επιλογή (με τη χρήση των χειρονομιών) είναι ο νέος τρόπος πλοήγησης του Android 10.
Οι προεπιλεγμένες χειρονομίες του Android 10 είναι οι παρακάτω. Εκτελούνται με τον τρόπο που περιγράφουμε και πραγματοποιούν τις ενέργειες που αναφέραμε:

- Κάνοντας Swipe up από το κάτω μέρος της οθόνης , αναλόγως ποια εφαρμογή είναι ανοιχτή σε μεταφέρει στην αρχική οθόνη.

- Με Swipe στα δεξιά ή στα αριστερά από την άκρη της οθόνης ο χρήστης μεταφέρεται στην προηγούμενη σελίδα ή αν υπάρχει κάποια ανοιχτή εφαρμογή κλείνει.
- Η εναλλαγή μεταξύ των ανοιχτών εφαρμογών γίνεται με swipe up και τη διατήρηση του δακτύλου στην οθόνη.
- Η ενεργοποίηση του Google Assistant γίνεται με swipe down στη δεξιά ή αριστερά στη κάτω γωνία της οθόνης.
- Επιπλέον, αν μία εφαρμογή έχει ενεργοποιημένο τον χειρισμό μέσω gestures τότε ο χρήστης αποκτά πρόσβαση στο βασικό μενού κάνοντας swipe από την αριστερή πλευρά με δύο δάχτυλα.
- Ένας ακόμη τρόπος για την εμφάνιση του βασικού μενού είναι το παρατεταμένο πάτημα στην αριστερή πλευρά της οθόνης και swipe up, αυτή η μέθοδος είναι ιδιαίτερα χρήσιμη στον χειρισμό της συσκευής με ένα χέρι.

2.2.6 “Android 11 ”

Εικόνα 2.6: Android 11

Η εταιρεία έχει συμπεριλάβει πολλά νέα χαρακτηριστικά στις πρώτες εκδόσεις του νέου λειτουργικού συστήματος.

- **Native screen recording (Εγγενής εγγραφή οθόνης)**

Τώρα υπάρχει ένα νέο εικονίδιο εγγραφής οθόνης στις επιλογές γρήγορων ρυθμίσεων στην κορυφή της οθόνης.

- **Chat Bubbles (Φυσαλίδες συνομηλίας)**

Οι φυσαλίδες είναι βασικά σαν αυτά τα περιφερόμενα εικονίδια συνομιλίας του Facebook Messenger που βρίσκονται και σε άλλες εφαρμογές. Η λειτουργία ήταν έτοιμη από το Android 10.

- **Share menu pinning(καρφίτσωμα μενού κοινής χρήσης)**

Το μενού κοινής χρήσης παρουσιάστηκε στο Android 7 Nougat. Έως 4 εφαρμογές κοινής χρήσης μπορούμε να καρφώσουμε στην κορυφή του μενού κοινής χρήσης.

- **Σίγαση ειδοποιήσεων κατά την εγγραφή βίντεο.**

Πολλές φορές οι ειδοποιήσεις είναι ενοχλητικές, ειδικά όταν προσπαθούμε να βγάλουμε φωτογραφίες ή να βγάλουμε βίντεο. Με το Android 11 έρχεται νέο Api κάμερας που έχει SetCameraAudioRestriction. Επιτρέπει στους προγραμματιστές να θέσουν τις εφαρμογές τους σε σίγαση όταν η κάμερα είναι ανοιχτή.

- **Το Bluetooth παραμένει ανοιχτό σε λειτουργία πτήσης**

Μέχρι και τώρα με το Android 10, η ενεργοποίηση της λειτουργίας πτήσης κλείνει τις ασύρματες συνδέσεις, ακόμη και το Bluetooth. Ωστόσο, αυτό αλλάζει με το Android 11. Τώρα, το Bluetooth θα παραμένει ενεργό όταν είναι ενεργοποιημένη η λειτουργία πτήσης.

- **Ιστορικό ειδοποιήσεων**
  Η νέα έκδοση του λειτουργικού συστήματος επιτρέπει τον έλεγχο όλων των ειδοποιήσεων που απορρίψαμε στη νέα λειτουργία ιστορικό ειδοποιήσεων.

2.2.7 "Android 12 "

Εικόνα 2.7: Android 12

Το Android 12 έχει αμέτρητα νέα βελτιωμένα χαρακτηριστικά:

- **Νέο UI και Material NEXT DESIGN**
- **Βελτιώσεις για μεγάλες οθόνες και lockscreen**
- **Νέα γραφική διεπαφή (UI) ειδοποιήσεων και μενού ρυθμίσεων**
- **Καλύτερα Screenshots κα ευκολότερη κοινοποίηση Wi-Fi**
- **Βελτιώσεις στον ήχο**
- **Αναβαθμίσεις μέσω Google Play**

2.3 ΑΡΧΙΤΕΚΤΟΝΙΚΗ ANDROID

Το Android λειτουργικό σύστημα έχει ως κέντρο του τον "πυρήνα" (Linux Kernel) και αποτελείται από μια στοίβα λογισμικού. Η αρχή της αρχιτεκτονικής είναι η επαναχρησιμοποίηση του κώδικα, που μας παρέχει τη δυνατότητα της δημοσίευσης και μοιράσματος μεταξύ των :

- Δραστηριοτήτων
- Υπηρεσιών
- Δεδομένων

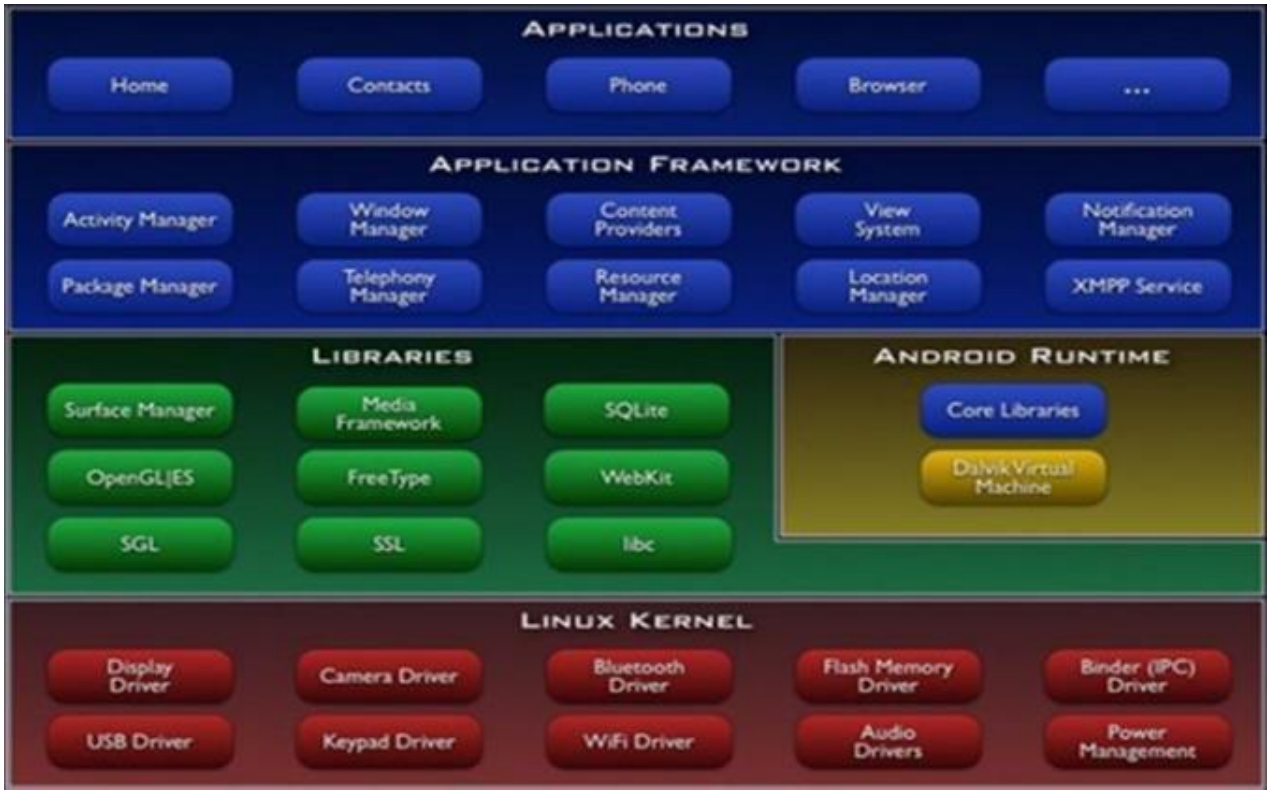
με άλλες εφαρμογές που τρέχουν επάνω στο λειτουργικό σύστημα.

Χάρη στη δυνατότητα αυτή μπορούμε να κάνουμε επέκταση ή βελτιστοποίηση των ήδη υπαρχόντων εφαρμογών που έχουν αναπτυχθεί, ή μπορούν να δημιουργηθούν από την αρχή.

Το λειτουργικό σύστημα Android δεν είναι απλά ένα ενιαίο λειτουργικό σύστημα, αλλά είναι μια στοίβα λογισμικού που αποτελείται από διάφορα επίπεδα, όπου σε κάθε επίπεδο εκτελείται μια ξεχωριστή λειτουργία. Οι υπηρεσίες διασύνδεσης με τις εφαρμογές και οι κύριες εφαρμογές που είναι (CORE):

- Email client

- Η εφαρμογή διαχείρισης SMS
- Το ημερολόγιο
- Ο browser
- Η εφαρμογή διαχείρισης επαφών και άλλες που είναι προεγκατεστημένες.



Εικόνα 2.8: Τα επίπεδα (layers) του λειτουργικού συστήματος Android

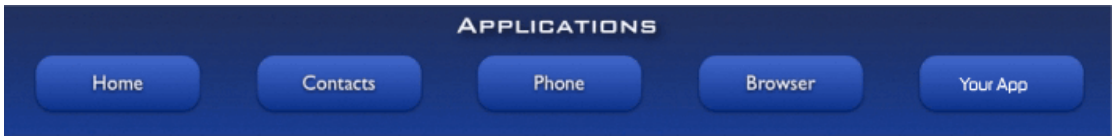
Η αρχιτεκτονική του λειτουργικού συστήματος Android υποστηρίζει:

- Παράλληλη εκτέλεση πολλαπλών εφαρμογών
- Ισότητα για την εκτέλεση της κάθε εφαρμογής
- Εκτέλεση εφαρμογών χωρίς περιορισμό.

Το Android λειτουργικό σύστημα αποτελείται από πέντε επίπεδα:

1-) Επίπεδο εφαρμογών (Application)

Στο επίπεδο αυτό μερικές βασικές εφαρμογές που διαθέτει το λειτουργικό σύστημα είναι:



Εικόνα 2.9: Android Operating System Application Layer

- E-mail clients
- Αποστολή SMS μηνύματα
- Το ημερολόγιο
- Οι Χάρτες (Google Maps)
- Browser
- Πρόγραμμα για επαφές χρηστών

2-) Επίπεδο πλαισίου εφαρμογών (Application Framework)

Εικόνα 2.10: Android Operating System Application Framework Layer

Το επίπεδο αυτό μας παρέχει υψηλού επίπεδου δομικές μονάδες που δίνουν τη δυνατότητα στους προγραμματιστές για τη δημιουργία των καινοτόμων εφαρμογών. Στο επίπεδο αυτό οι προγραμματιστές μπορούν να εκμεταλλευτούν τις δυνατότητες του υλικού των κινητών συσκευών (π.χ. όπως ο Χάρτης ή οι αισθητήρες που έχουν οι συσκευές για τον εντοπισμό θέσεων μέσω GPS). Στο επίπεδο αυτό κάθε εφαρμογή που εκτελείται στον λειτουργικό σύστημα μπορεί να χρησιμοποιεί τις δυνατότητες μιας άλλης εφαρμογής.

Τα βασικά δομικά στοιχεία του πλαισίου εφαρμογών είναι:

- ***Διαχειριστής περιεχομένου (Content Manager)***
  Είναι η δυνατότητα που παρέχει στον προγραμματιστή εφαρμογών να διαμοιράζονται τα δεδομένα από μια εφαρμογή σε μια άλλη. Τέτοια δεδομένα μπορεί να είναι οι επαφές του χρήστη στις βάσεις δεδομένων.
- ***Διαχειριστής πόρων (Resource Manager)***
  Επιτρέπει την πρόσβαση :
  - *String*
  - *Εικόνες*
  - *Layout Files*
- ***Διαχειριστής τοποθεσίας (Location Manager)***
  Παρέχει την δυνατότητα μέσω του κατάλληλου υλικού που διαθέτουν οι κινητές συσκευές (π.χ. GPS) για τον καθορισμό της τοποθεσίας επάνω στον χάρτη .
- ***Διαχειριστής ειδοποιήσεων (Notification Manager)***
  Επιτρέπει στους χρήστες των κινητών συσκευών να ενημερώνονται για τα διάφορα γεγονότα που συμβαίνουν. Οι ειδοποιήσεις αυτές μπορεί να είναι :
  - Μπάρα κατάστασης (Status Bar)
  - Toast μηνυμάτων τα οποία εμφανίζονται στο κάτω μέρος της οθόνης

- Δόνηση ή ενεργοποίηση της οθόνης

- ***Διαχειριστής δραστηριοτήτων (Activity Manager)***
  Ρυθμίζει το χρόνο ζωής των εφαρμογών και επιτρέπει στο χρήστη να πλοηγείται στις προηγούμενες εφαρμογές
- ***Σύστημα προβολών (View System)***

Περιέχει το σύνολο των αντικειμένων που χρησιμοποιούνται στο σχεδιασμό της οπτικής διεπαφής όπου αλληλεπιδρά με τον χρήστη. Μερικά από αυτά είναι τα πεδία εισαγωγής κειμένου, το πλέγμα , οι λίστες.

- ***Ικανότητα δικτύωσης (Connectivity Manager)***

  Περιέχει πληροφορίες για τις δυνατές συνδέσεις μιας συσκευής, καθώς και την κατάσταση κάθε σύνδεσης.

3-) Επίπεδο Χρόνος Εκτέλεσης Εφαρμογών (Android Runtime)

Εικόνα 2.11: Android Operating System Runtime Layer

Περιέχει ένα σύνολο βασικών βιβλιοθηκών και την **Dalvik Virtual Machine .**

Το Dalvik Virtual Machine είναι μια εικονική μηχανή Java για φορητές συσκευές. Σε αυτή εκτελείται ο κώδικας bytecode των εφαρμογών. Κάθε εφαρμογή δεν έχει επαφή με την άλλη παρόλο που εκτελούνται ταυτόχρονα.

4-) Επίπεδο βιβλιοθηκών (Libraries)

Εικόνα 2.12: Οι βιβλιοθήκες του Android

Οι κύριες βιβλιοθήκες που διαθέτει το λειτουργικό σύστημα Android είναι :

1) **System C Library :**
   Είναι η ενσωματωμένη βιβλιοθήκη της C, διαμορφωμένη έτσι ώστε να είναι κατάλληλη για κινητές συσκευές που βασίζονται σε Linux. Η κύρια αρμοδιότητα της βιβλιοθήκης είναι οι λειτουργίες που αφορούν την μείωση της μνήμης.
2) **Surface Manager:**
   Είναι ο διαχειριστής των σχεδιαστικών επιφανειών. Η σχεδιαστική επιφάνεια είναι η οθόνη της συσκευής που έχει δεσμευθεί για να κρατήσει τα δεδομένα προς απεικόνιση. Ο Διαχειριστής των επιφανειών σε συνδυασμό με το διαχειριστή παραθύρων (Window Manager) καθορίζουν τη σειρά σχεδίασης των παραθύρων, έτσι ώστε να υπάρχει μια σωστή αλληλουχία.
3) **SQLITE:** αποτελεί μια πολύ ισχυρή βάση δεδομένων που χρησιμοποιείται από κινητές εφαρμογές. Εκτελεί τις λειτουργίες όπως η εγγραφή - ανάγνωση και αποθήκευση των δεδομένων.
4) **Βιβλιοθήκες 3D:** είναι οι βιβλιοθήκες αυτές που χρησιμοποιούνται για την τρισδιάστατη απεικόνιση**.**
5) **Βιβλιοθήκες πολυμέσων:** είναι η βιβλιοθήκη που επιτρέπει την αναπαραγωγή και εγγραφή πολλαπλών μέσων εικόνας και ήχου.
6) **SGL:** είναι μια μηχανή για που παρέχει τη δυνατότητα σχεδίασης των δισδιάστατων γραφικών**.**
7) **FreeType:** προσφέρει ευκρίνεια στα γραφικά που χρησιμοποιούνται στα bitmaps και στις γραμματοσειρές των εφαρμογών.
8) **LibWebCore:** παρέχει την πλοήγηση στο διαδίκτυο και χρησιμοποιείται από τον browser του Android και τις Web Views.
9) **WebKit:** είναι η μηχανή διάταξης των γραφικών που δίνει την δυνατότητα στον browser να απεικονίσει τις ιστοσελίδες.

5-)Επίπεδο Πυρήνα του LINUX

Εικόνα 2.13: Linux kernel in Android Operating System

Το λογισμικό του Android βασίζεται στον πυρήνα Linux,έκδοση 4.9 που επιτρέπει να μπορεί να χρησιμοποιηθεί σε ένα μεγάλος εύρος από πλατφόρμες.

Ειδικότερα χρησιμοποιείται :

- Για τη διαχείριση μνήμης
- Για τη διαχείριση των διεργασιών

- Τις λειτουργίες του δικτύου
- Για την ασφάλεια του λειτουργικού συστήματος
- Οδηγίες για υλικό

2.3.1 ΤΟ ΣΧΕΔΙΟ ΣΧΕΔΙΑΣΗΣ (DESIGN PATERNS) MVC

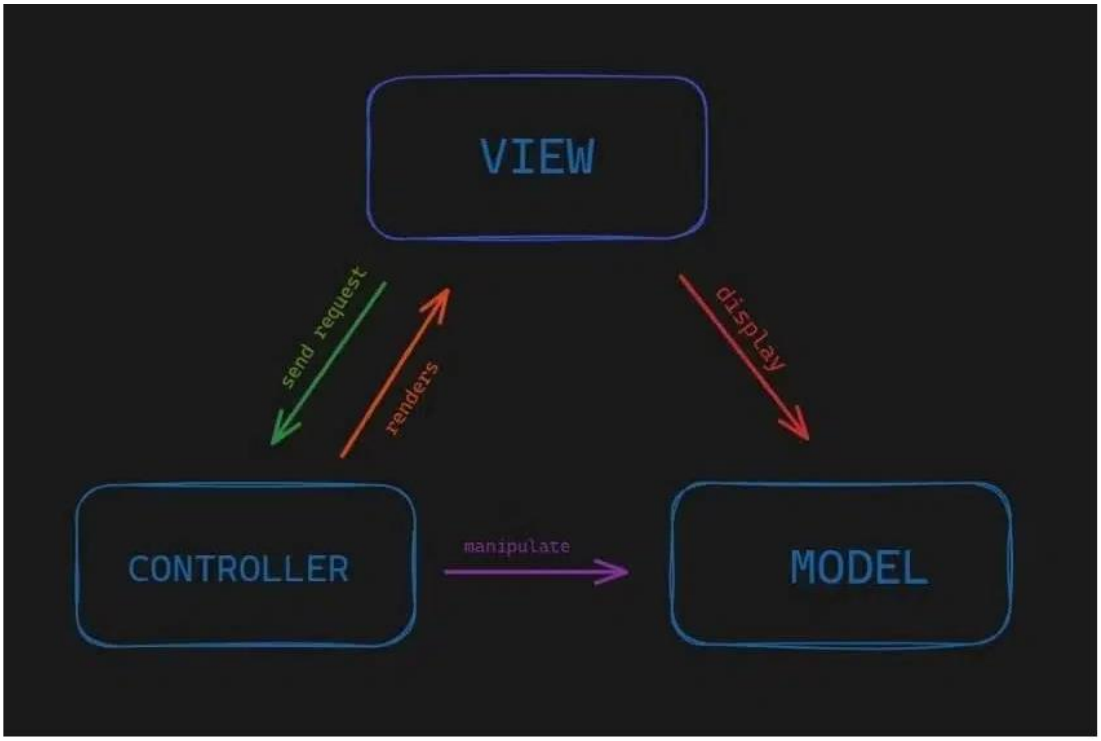
Το MVC είναι ένας όρος ο οποίος σημαίνει Model View Controller και αποτελεί την αρχιτεκτονική που χρησιμοποιεί το Android λειτουργικό σύστημα. Το MVC μοντέλο δεν θεωρείται μια καινούρια έννοια στον προγραμματισμό, αλλά ειδικά τα τελευταία χρόνια με την εξέλιξη του αντικειμενοστραφή προγραμματισμού έχει έρθει στην επικαιρότητα των προγραμματιστών και άρχισαν να καταλαβαίνουν τα πλεονεκτήματα που προσφέρει στους προγραμματιστές κατά την διάρκεια της συγγραφής κώδικα.

Ας ξεκινήσουμε αρχικά από το κομμάτι του μοντέλου (MODEL) της αρχιτεκτονικής MVC. Το μοντέλο έχει δημιουργηθεί ώστε να διευκολύνει την επικοινωνία της εφαρμογής μας με τη βάση δεδομένων και διευκολύνει τους προγραμματιστές να προσαρμόσουν τη δομή της σχεσιακής βάσης να προσαρμόσουν στην εφαρμογή τους. Είναι το κύριο μέρος του μοντέλου να διαχειρίζεται την ανάκτηση και την αποθήκευση δεδομένων στο σύστημα. Το Model στην ουσία αναπαριστά τα δεδομένα μας. Πολλές φορές οι κανόνες που ισχύουν στις σχεσιακές βάσεις δεδομένων είναι δύσκολο να εφαρμοστούν σε αντικειμενοστραφή προγραμματισμό. Όποτε στο σημείο αυτό ακολουθώντας το μοντέλο MVC αναπτύχθηκαν διάφοροι μέθοδοι και στρατηγικές για να διευκολύνουν την επικοινωνία της εφαρμογής με τη βάση χωρίς ιδιαίτερη προσπάθεια.

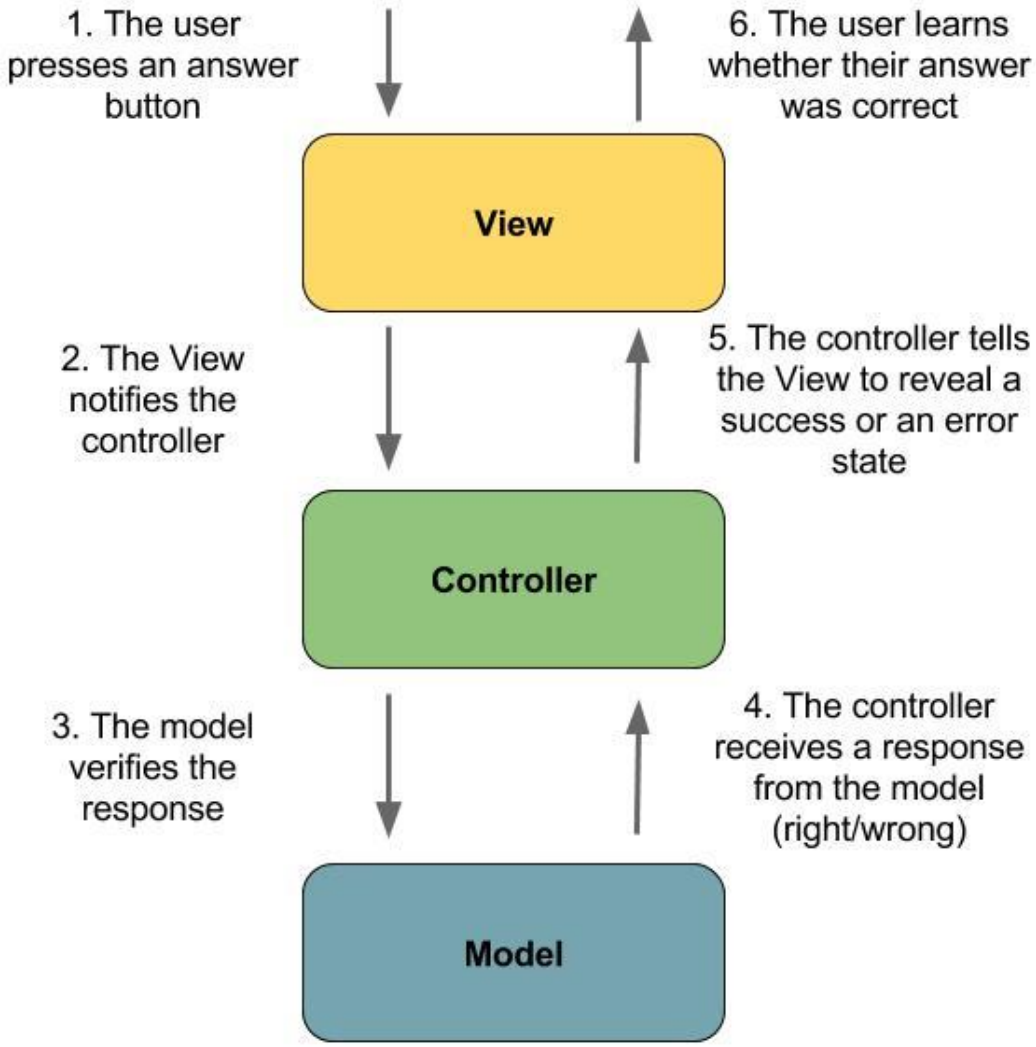
Το View είναι το οπτικό μέρος της εφαρμογής, έρχεται σε αλλεπίδραση με τον χρήστη μέσω των διάφορων διεπαφών που παρέχουν τα frameworks. Για την δημιουργία των διεπαφών για να αλληλεπιδράσουν με τους τελικούς χρήστες συνήθως χρησιμοποιείται η γλώσσα XML. Προβάλλουν τα δεδομένα που έρχονται από τα μοντέλα της αρχιτεκτονικής MVC όταν είναι για αναπαράσταση των δεδομένων ενώ όταν είναι να καταχωρήσουν δεδομένα στην αρχιτεκτονική μας προωθούν τα δεδομένα στα κατάλληλα Controller για περαιτέρω επεξεργασία και τα Controlllers με βάση των κατάλληλων κανόνων που έχουν οριστεί εκτελούν ή όχι την συγκεκριμένη επιθυμητή ενέργεια.

Τέλος, ο Controller εκτελεί τον ρόλο του διαχειριστή μεταξύ του Μοντέλου (Model) και της Οπτικής διεπαφής (View). Οι λειτουργικές ενέργειες στην εφαρμογή μας εκτελούνται μέσω των Controllers. Οι Controllers καθορίζουν την επικοινωνία με τα μοντέλα, προωθούν τα αιτήματα για τη λήψη των

δεδομένων και μετά από την κατάλληλη επεξεργασία τα στέλνουν για απεικόνιση στην οπτική διεπαφή χρηστών (View).



Εικόνα 2.14: Android Operating System Runtime Layer



Εικόνα 2.15: MVC αρχιτεκτονική και η σειρά εκτέλεσης λειτουργιών.

Ας δούμε με ποια σειρά εκτελούνται τα αντίστοιχα πεδία της αρχιτεκτονικής MVC. Θα μελετήσουμε την περίπτωση ο χρήστης της εφαρμογής να κάνει την ενέργεια να δει το μενού του εστιατορίου.

- Αρχικά ο Χρήστης της εφαρμογής στέλνει ένα αίτημα στη διεπαφή μας μέσω του View για την αναπαράσταση των κατάλληλων δεδομένων και ενεργοποιεί την αντίστοιχη μέθοδο του Controller για τη λήψη των δεδομένων από τη βάση.
- Controller από τη στιγμή που εκτελέσει την κατάλληλη επεξεργασία του αιτήματος (π.χ. αν έχει δικαίωμα να το κάνει αυτό ο συγκεκριμένος χρήστης ή όχι), επικοινωνεί με το κατάλληλο Μοντέλο για την λήψη των δεδομένων (χρήση μεταβλητής).
- Παίρνει τα δεδομένα από το Μοντέλο προσαρμόζοντας αυτά που ζητάει ο Χρήστης.
- Ο Controller προωθεί τα ζητούμενα δεδομένα στα κατάλληλα πεδία γραφικής διεπαφής ώστε ο χρήστης να έχει πρόσβαση στα ζητούμενα δεδομένα.

- **Διαχωρισμός προβλημάτων:**

  Το μοντέλο MVC αποτελείται από τρία επίπεδα. Έχει ανατεθεί σε κάθε επίπεδο μια συγκεκριμένη λειτουργία αλλά και το κάθε επίπεδο είναι σε στενή επαφή με τα υπόλοιπα επίπεδα της αρχιτεκτονικής. Διευκολύνει τους προγραμματιστές για τον εντοπισμό των προβλημάτων.

- **Επεκτασιμότητα:**

  Επίσης ένα από τα βασικά πλεονεκτήματα της αρχιτεκτονικής MVC είναι και η **επεκτασιμότητα**. Πρέπει κάποιες λειτουργίες που διαθέτει η εφαρμογή μας να έχουν σχεδιαστεί με τέτοιο αφηρημένο τρόπο, δηλαδή όχι να περιοριστούμε σε κάτι συγκεκριμένο, αλλά να το έχουμε σχεδιάσει με πολύ γενικό τρόπο.

- **Ελεγξιμότητα:**

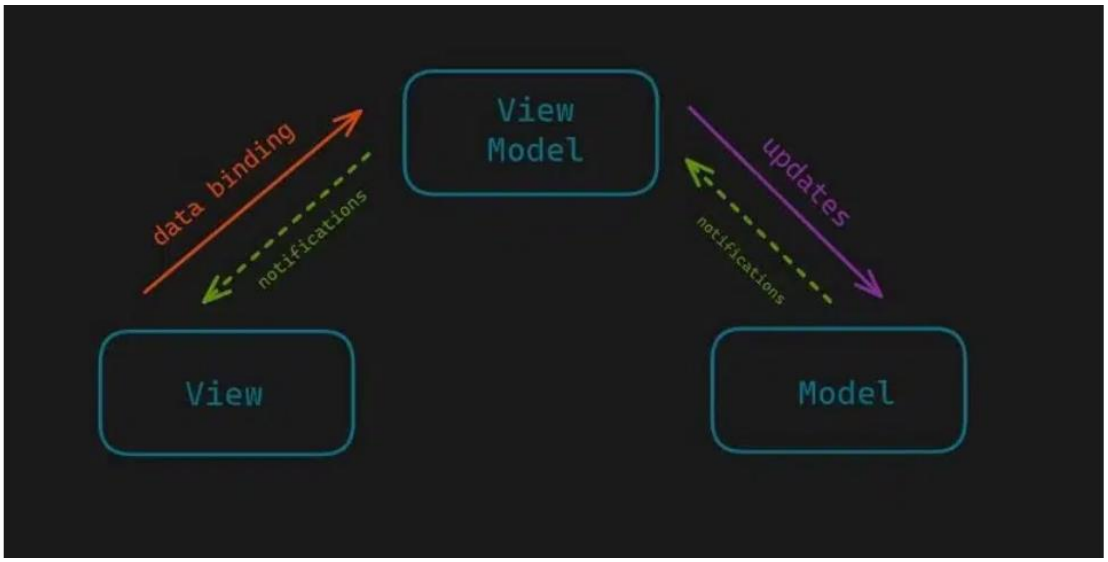
  Η πιο σημαντική ιδιότητα της αρχιτεκτονικής MVC είναι η **ελεγξιμότητα**. Οι εφαρμογές που δημιουργήθηκαν με βάση τους κανόνες της συγκεκριμένης αρχιτεκτονικής παρέχουν μεγάλη ευκολία στους προγραμματιστές στον εντοπισμό των προβλημάτων και στη συντήρηση τους .

2.3.2 ΤΟ ΣΧΕΔΙΟ ΣΧΕΔΙΑΣΗΣ (DESIGN PATERN) MVVM

Το MVVM είναι η εξέλιξη του προτύπου ανάπτυξης του MVC. Ο βασικός σκοπός του MVVM είναι να παρέχει έναν σαφή διαχωρισμό μεταξύ του τομέα λογικού και του επιπέδου προβολής. Το MVVM παρέχει αμφίδρομη σύνδεση δεδομένων μεταξύ της προβολής αυτής και του μοντέλου προβολής.

Το μοτίβο MVVM μας επιτρέπει να διαχωρίζουμε την προβολή κώδικα και το μοντέλο μας. Κάτι το οποίο σημαίνει ότι όταν αλλάζει το μοντέλο, η προβολή δεν χρειάζεται. Με το μοντέλο προβολής, μπορούμε να εκτελούμε δοκιμές και να ελέγχουμε τη λογική συμπεριφορά.

- Τα βασικά στοιχεία του MVVM μοντέλου είναι :
- **Το Μοντέλο** -> Διατηρεί τα δεδομένα
- **Το μοντέλο προβολής** -> Λειτουργεί ως σύνδεση μεταξύ του μοντέλου και της προβολής
- **Προβολή** -> Διατηρεί τις διεπαφές του Χρήστη



Εικόνα 2.16: MVVM αρχιτεκτονική

2.4 ΤΟ ΠΑΙΔΑΓΩΓΙΚΟ ΠΛΑΙΣΙΟ

2.4.1 Βασικές Αρχές

Οι εφαρμογές κινητών συσκευών αρχίζει όλο και περισσότερο να λαμβάνει μέρος στην εκπαιδευτική διαδικασία. Με βάση τις πρόσφατες μελέτες που έχουν γίνει από τους ερευνητές (J. Hung, 2012; J.-L. Hung & Zhang, 2012), υπάρχουν αρκετές δημοσιεύσεις που αφορούν την κινητή μάθηση (mobile learning). Οι επιστημονικές δημοσιεύσεις (Roschelle, 2003; M. Sharples, Arnedillo-Sánchez, Milrad, & Vavoula, 2009; M. Sharples et al., 2005; Mike Sharples, 2000; Zurita & Nussbaum, 2004) αναφέρονται στα θετικά αποτελέσματα των εφαρμογών την κινητών συσκευών, στη μαθησιακή και στη διδακτική διαδικασία.

Η ερευνητική δραστηριότητα των προγραμματιστών επικεντρώνεται στα εξής:

- Σχεδίαση κινητών εφαρμογών (Mike Sharples, Corlett, & Westmancott, 2002)
- Θέματα που αφορούν τη διαχείριση της εκπαιδευτικής διαδικασίας εντός και εκτός σχολείου από τους εκπαιδευτικούς και από τους ίδιους τους μαθητές.

2.4.2 Παιδαγωγικές χρήσεις

Η αξιοποίηση των κινητών συσκευών στην εκπαιδευτική διαδικασία γίνεται στο πλαίσιο καινοτόμων παιδαγωγικών δραστηριοτήτων που συμβαίνουν τόσο μέσα στην τάξη όσο και εκτός αυτής.

Τέτοιες δραστηριότητες αφορούν :

- Πρόσβαση και διαχείριση του εκπαιδευτικού υλικού
- Επικοινωνία και συνεργασία στο πλαίσιο της μαθησιακής διαδικασίας.

Οι κινητές συσκευές βοηθάνε στην αυτόνομη και στην ανεξάρτητη μάθηση, λόγω του χαμηλού κόστους τους και της φορητότητάς τους και προσφέρουν στους μαθητές εύκολη και συνεχόμενη πρόσβαση.

Χρησιμοποιώντας αυτές τις συσκευές μπορούμε να έχουμε :

- **Πρόσβαση σε πληροφορίες:**
  Χρησιμοποιώντας αυτές της συσκευές μπορούμε να έχουμε εύκολη πρόσβαση σε εκπαιδευτικό υλικό (π.χ. ηλεκτρονικά βιβλία, ηλεκτρονικά λεξικά, εγχειρίδια κλπ.)

- **Μεταφορά δεδομένων :**
  Λήψη και αποστολή εκπαιδευτικού υλικού μεταξύ των μαθητών και του εκπαιδευτικού με γρήγορο και εύκολο τρόπο.

- **Καταχώρηση πληροφοριών :**
  Μπορούν να χρησιμοποιηθούν για συγγραφή κειμένων, για καταχώρηση προσωπικών δεδομένων.

- **Συλλογή δεδομένων :** είναι κατάλληλα εργαλεία για συλλογή δεδομένων με αυτόνομο τρόπο (ηχογράφηση, φωτογράφηση, βιντεοσκόπηση).

- **Αναζήτηση πληροφοριών και επικοινωνία:**
  Μπορούν να συνδεθούν στο διαδίκτυο και να χρησιμοποιηθούν ως εργαλεία πρόσβασης στις υπηρεσίες του.
- **Αξιολόγηση:** ο εκπαιδευτικός μπορεί εύκολα να στείλει και να παραλάβει ασκήσεις που έχουν λυθεί από τους μαθητές και έτσι με αυτό τον τρόπο επιταχύνει την διαδικασία της επανατροφοδότησης.

- **Διαχείριση:** Οργάνωση των εκπαιδευτικών δραστηριοτήτων και του προγράμματος.
  Οι μαθητές χρησιμοποιώντας κινητές συσκευές εκτός από τα προηγούμενα που αναφέραμε μπορούν βελτιώσουν την ατομική και την συνεργατική μάθηση, αλλά να συμβάλλουν στην οικοδόμηση των γνώσεων και απόκτηση ικανότητας υψηλής επιστημονικής κατανόησης. Οι συσκευές αυτές διαθέτουν τα κατάλληλα εργαλεία για επιστημονικούς υπολογισμούς αλλά και για την οπτικοποίηση των δεδομένων.

2.4.3 Μάθηση μέσω κινητών συσκευών.

Μάθηση μέσω κινητών συσκευών είναι μια μορφή μάθησης που πραγματοποιείται χωρίς ο εκπαιδευόμενος να χρειάζεται να βρίσκεται σε προκαθορισμένα σημεία και χρησιμοποιώντας τις δυνατότητες που προσφέρουν οι ασύρματες φορητές τεχνολογίες οι συσκευές (Vavoula & Karagiannidis, 2005).

Οι κινητές συσκευές διαθέτουν μια σειρά από χαρακτηριστικά όπως:

- Είναι φθηνές σε σχέση με του υπολογιστές και μπορούν να μεταφερθούν εύκολα.
- Προσφέρουν πρόσβαση σε πληροφορίες λόγω του χαμηλού κόστους.
- Προσφέρουν δυνατότητες για ανεξάρτητη μάθηση (independent learning)
- Διευκολύνουν τα άτομα με ειδικές ανάγκες.

Τέσσερα κύρια επίπεδα αξιοποίησης των κινητών συσκευών στην εκπαιδευτική διαδικασία καταγράφονται στη βιβλιογραφία (Gay et al., 2002).

Η παραγωγικότητα:

- Ημερολόγια
- Επικοινωνία
- Βαθμολόγηση
- Αξιοποίηση
- Χρονοπρογραμματισμός

Ευέλικτη πρόσβαση:

- Τοπικές βάσεις δεδομένων
- Διαδραστικές εφαρμογές
- Ασύγχρονη εκπαίδευση

Συλλογή δεδομένων:

- Διαδικτυακές βάσεις δεδομένων
- Συλλογή δεδομένων
- Σύνθεση δεδομένων
- Mobile βιβλιοθήκες

Επικοινωνία και συνεργασία:

- Επικοινωνία πραγματικού χρόνου
- Σημειώσεις
- Μοίρασμα δεδομένων
- Ασύρματη επικοινωνία

2.4.4 Πλεονεκτήματα των Κινητών συσκευών στην Εκπαίδευση

Οι κινητές συσκευές παρέχουν τα ακόλουθα πλεονεκτήματα:

- Η συνεισφορά τους στη διαδικασία της εκμάθησης.
- Εξαιτίας της διαδραστικότητας προκαλούν κίνητρο για εκμάθηση.
- Ευκολία χρήσης των εφαρμογών.
- Ευκολία και χρήση της διαδικασίας της ανατροφοδότησης (π.χ. εξετάσεις).
- Η ευελιξία.
- Η διαθεσιμότητα.
- Το ενδιαφέρον .
- Εύκολη πρόσβαση σε σύγκριση με τους υπολογιστές.

2.4.5 Μειονεκτήματα των Κινητών συσκευών στην Εκπαίδευση

Η χρήση των κινητών συσκευών εκτός από τα πλεονεκτήματα μπορούν να παρουσιάσουν και μειονεκτήματα όπως:

- Το κόστος (Αλλάζει συνέχεια η τεχνολογία οπότε θα χρειαστώ τη κατάλληλη συσκευή)
- Ευαισθησία (για το σχολικό περιβάλλον)
- Τεχνικά προβλήματα (π.χ. μπαταρία, έλλειψη διαθέσιμης μνήμης)
- Έλλειψη κατάλληλου λογισμικού.

2.5 ΠΑΡΟΜΟΙΕΣ ΕΦΑΡΜΟΓΕΣ

Τα εκπαιδευτικά λογισμικά αποτελούν ένα χρήσιμο εργαλείο στα χέρια των εκπαιδευτικών αλλά και των γονιών. Βελτιώνουν την διαδικασία της μάθησης και την κάνουν πιο ενδιαφέρουσα και διασκεδαστική. Με αυτό τον τρόπο, οι μαθητές μαθαίνουν τις έννοιες με ευχάριστο τρόπο, ενώ παράλληλα ενεργοποιούν τις αισθήσεις τους και ακολουθούν τους δικούς τους ρυθμούς μάθησης.

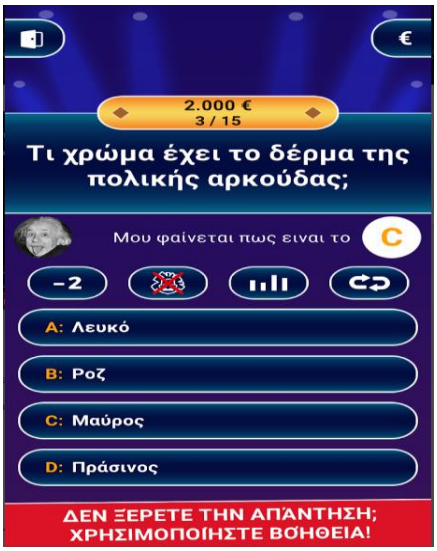
2.5.1 Ο Βασιλιάς των μαθηματικών .

Είναι ένα παιχνίδι μαθηματικών σε μεσαιωνικό περιβάλλον. Ανεβαίνεις στην κοινωνική κλίμακα απαντώντας σε ερωτήσεις μαθηματικών και λύνοντας παζλ. Συλλέγεις αστέρια, κερδίζεις μετάλλια και συναγωνίζεσαι με φίλους και την οικογένεια. Είναι κατάλληλο για παιδιά άνω των 6 ετών και παρουσιάζει τα μαθηματικά με ένα προσιτό και εμπνευσμένο τρόπο. Οι παίκτες ενθαρρύνονται να σκέφτονται για τους ίδιους και να βλέπουν τις μαθηματικές έννοιες από διαφορετική σκοπιά, λύνοντας προβλήματα σε πολλούς τομείς.

Εικόνα 2.17: Ο βασιλιάς των μαθηματικών

2.5.2 Εκατομμυριούχος 2023 .

Το Παιχνίδι έχει 15 ερωτήματα και 4 βοήθειες. Όποιος απαντάει σωστά σε 15 ερωτήσεις κερδίζει το παιχνίδι. Υπάρχουν πάνω από 100 ερωτήσεις πολλαπλής επιλογής κατάλληλα επιλεγμένες για τον διασκεδαστικό αλλά και επιμορφωτικό τους χαρακτήρα. Παρέχεται καταγραφή των υψηλότερων σκορ, το γίνε εκατομμυριούχος και το σύγκρινε το σκορ σου με τους άλλους παίκτες σε όλη την Ελλάδα.



Εικόνα 2.18: Εκατομμυριούχος

ΚΕΦΑΛΑΙΟ 3: ΤΕΧΝΟΛΟΓΙΕΣ

3.1 Απαιτήσεις και Προδιαγραφές

Στο κεφάλαιο αυτό γίνεται περιγραφή και ανάλυση των κριτηρίων επιλογής όλων αυτών των τεχνολογιών, που χρειάστηκαν για την υλοποίηση του συστήματος, όπως είναι η Kotlin και RoomDB για την τοπική αποθήκευση των δεδομένων και το Firebase για αποθήκευση των δεδομένων σε υπολογιστικό νέφος (Cloud) και άλλες τεχνολογίες που χρησιμοποιήθηκαν για την προσπέλαση των δεδομένων.
Η επιλογή της γλώσσας Kotlin αντί για Java και ROOMDB αντί για SQLITE έγινε γιατί μετά την διεξαγωγή μιας μικρής έρευνας, καταλήξαμε ότι αυτές οι τεχνολογίες υπερτερούν σε σύγκριση με τις υπόλοιπες.

3.2 Η γλώσσα προγραμματισμού KOTLIN

Οι προγραμματιστές ανάπτυξης εφαρμογών εδώ και καιρό χρησιμοποιούν τη γλώσσα Java για τη δημιουργία των εφαρμογών Android.Ο τομέας της πληροφορικής συνεχώς εξελίσσεται .Εξαιτίας αυτής της πραγματικότητας η Google ανακοίνωσε το 2017 ότι θα υποστηρίξει τη γλώσσα προγραμματισμού Kotlin στο περιβάλλον Android.

Σε αυτό το κεφάλαιο θα εξηγήσουμε λίγο τη δομή της γλώσσας, τα πλεονεκτήματα που έχει και τις διαφορές που έχει με τη Java. Η Kotlin είναι μια νέα γλώσσα προγραμματισμού που αναπτύσσεται από την εταιρεία JetBrains το 2011 και η πρώτη σταθερή έκδοση (v1.0) κυκλοφόρησε το 2016. Όπως αναπτύσσεται εφαρμογές στο περιβάλλον Android με τη γλώσσα Java μπορείτε να αναπτύξετε εφαρμογές και με τη γλώσσα Kotlin. Επίσης, μπορεί να υλοποιηθεί για ανάπτυξη διαδικτυακών (Server-Side) εφαρμογών.

Στη γλώσσα kotlin ορίζουμε πρώτα το όνομα της μεταβλητής και μετά τον τύπο της μεταβλητής μας ανάλογα με τον τύπο των δεδομένων. Ακόμα και αν δεν γράψουμε τον τύπο της μεταβλητής μας ,η γλώσσα Kotlin εκχωρεί τον τύπο της ,σύμφωνα με την τιμή που έχει εκχωρηθεί στη μεταβλητή .

Τα πλεονεκτήματα της γλώσσας:

- Η Kotlin. όπως και η Java, είναι μια γλώσσα που μπορεί να εκτελεστεί σε οποιαδήποτε περιβάλλον (Ανέξάρτητη πλατφόρμα).
- Μπορεί να μεταγλωττιστεί σε κώδικα Java.
- Όλες οι βιβλιοθήκες και framework που είναι για Java μπορούν να ενσωματωθούν και να χρησιμοποιηθούν από τη γλώσσα Kotlin.
- Η γλώσσα Kotlin μπορεί εύκολα να ενσωματωθεί με Maven,Gradle και με άλλα συστήματα.
- Μαθαίνεται εύκολα σαν γλώσσα.
- Είναι απλή στη χρήση και στη κατανόηση.
- Είναι μια γλώσσα Ανοιχτού κώδικα.

Features	Kotlin	Java
1. Extension Functions	It is already available in Kotlin	In java, we need to create class
2. Null Safety	It is available in Kotlin	It is not available in Java
3. Static Members	Kotlin doesn't have a static member for a class	It is available in Java
4. String Templates	Yes, there are two types of string literals in Kotlin	It is available in Java too but it doesn't support expression like Kotlin
5. Wildcard Types	It is not available in Kotlin	Available in Java
6. Smartcasts	Available in Kotlin	Not Available in Java
7. No Checked Exceptions	Kotlin removed exceptions entirely	It is problematic in Java
8. Operator Overloading	Kotlin allows users to provide a way to invoke functions	Operators are tied to particular Java Types
9. Constructors	It has primary constructor and secondary constructor	Constructors can be used to take parameters to initialize attributes
10. Type System	It gives nullability support, type inference, and universal guards	There are other kinds of reference types related to the basic concept of class

Εικόνα 3.1: Kotlin vs JAVA

Οι κύριες διαφορές τους:

- **Null Safety :** Η εξαίρεση όπως το NullPointerException έχει καταργηθεί. Αυτό συμβαίνει διότι δεν μπορείτε να εκχωρήσετε null τιμές σε καμία μεταβλητή ούτε οι μέθοδοι μπορούν να πάρουν null τιμές.
- **Data Classes:** δεν χρειάζεται να γράψουμε get/set μεθόδους.
- **Type inference:** είναι μια ωραία ιδιότητα να καθορίσουμε τον τύπο κάθε μεταβλητής. Εάν θέλουμε να ορίσουμε ρητά έναν τύπο δεδομέων, στη γλώσσα Kotlin μπορούμε να το κάνουμε με εύκολο τρόπο.
- **High-order functions and lambdas:** Επιτρέπει υπερφόρτωση τελεστών και των συναρτήσεων**.**

3.3 ROOMDB

3.3.1 Ο Ορισμός της βιβλιοθήκης ROOM

Το Room είναι μια βιβλιοθήκη ORM (Object Relational Mapping). Το ORM είναι μια γέφυρα μεταξύ μπλοκ κώδικα και της βάσης δεδομένων. Το Room επιτρέπει τη μετατροπή των πεδίων (του πίνακα στη βάση) σε αντικείμενα στη Java. Στην ουσία βοηθάει στον προγραμματιστή των εφαρμογών με αφηρημένο τρόπο να μπορεί να δημιουργεί ερωτήματα στη βάση χωρίς να έχει ιδιαίτερες γνώσεις Sql.

Η πιο συνηθισμένη περίπτωση χρήσης είναι η προσωρινή αποθήκευση δεδομένων. Με αυτό τον τρόπο ο χρήστης μπορεί να έχει πρόσβαση στα δεδομένα ακόμα και όταν δεν έχει πρόσβαση στο διαδίκτυο. Η βιβλιοθήκη αυτή μας παρέχει ένα επίπεδο αφαίρεσης επάνω στην τοπική βάση SQLITE. Τα βασικά μέρη της βιβλιοθήκης αυτής είναι :

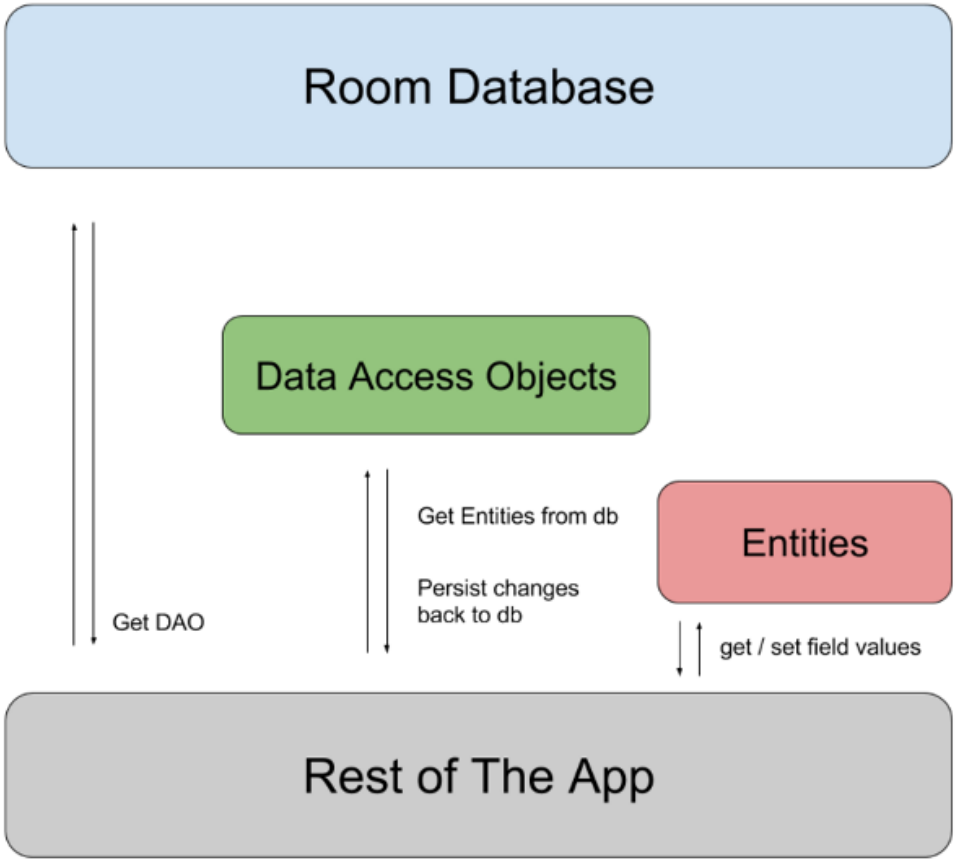
- **Η κλάση βάσης δεδομένων** (**Database Class**): είναι μια αφηρημένη κλάση όπου κρατάει όλη τη πληροφορία που αφορά γενικά τη βάση και τη δομή των πινάκων που υπάρχει στη βάση μας.
- **Οι οντότητες δεδομένων (Data Entities**): αντιστοιχούν στους πίνακες δεδομένων της

εφαρμογής μας. Το όνομα του Μοντέλου (Entities) αντιστοιχεί σε όνομα του αντίστοιχου πίνακα που υπάρχει στη βάση .

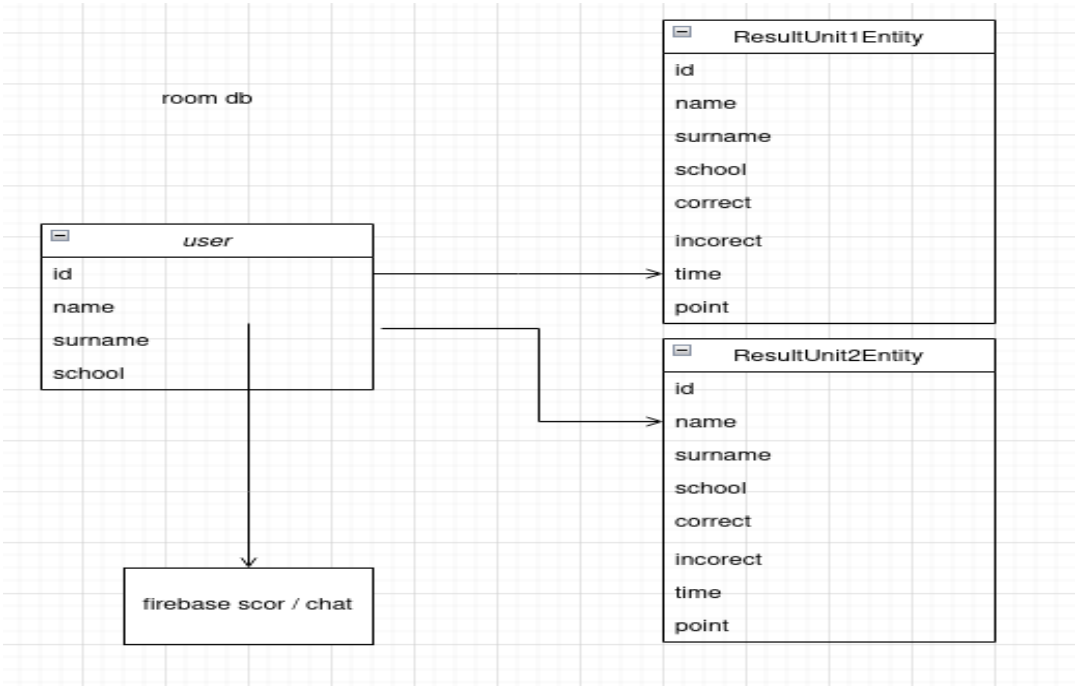
- **Τα Αντικείμενα πρόσβασης των δεδομένων (Data Acess Object(Dao)**: παρέχουν μεθόδους στην εφαρμογή μας που εκτελούν τις λειτουργίες εισαγωγής, ενημέρωσης και διαγραφής των δεδομένων από την βάση δεδομένων. Είναι ο τρόπος προσπέλασης των δεδομένων από τη βάση. Είναι ο διαμεσολαβητής μεταξύ της βάσης και του χρήστη.

Η κλάση βάσης δεδομένων παρέχει στην εφαρμογή μας τα αντικείμενα τύπου DAO που σχετίζονται με την βάση δεδομένων. Η εφαρμογή μας χρησιμοποιεί τα αντικείμενα DAO για να ανακτήσει τα δεδομένα από τη βάση ως παρουσίες των σχετιζομένων αντικειμένων με τη βάση. Η εφαρμογή μπορεί επίσης να χρησιμοποιήσει τις καθορισμένες οντότητες δεδομένων για να ενημερώσει τα γνωρίσματα (rows)από τους αντίστοιχους πίνακες που έχουμε στη βάση μας ή να δημιουργήσει καινούριες για εισαγωγή νέων δεδομένων.

Το παρακάτω σχήμα μας απεικονίζει τη σχέση μεταξύ των διαφορετικών στοιχείων της βιβλιοθήκης ROOM.



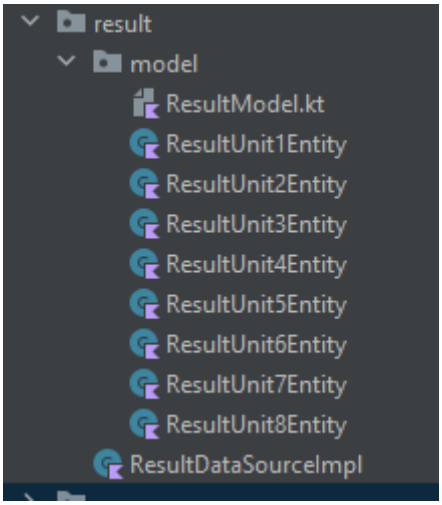
Εικόνα 3.2: Αρχιτεκτονική ROOMDB



Εικόνα 3.3: Σχεσιακός πίνακας στη βάση ROOMDB

Στην παραπάνω εικόνα ο τοπικός πίνακας Χρήστης τροφοδοτεί τους υπόλοιπους πίνακες όπως φαίνεται στο σχήμα.

ENTITIES:



Εικόνα 3.4: Τα διάφορα Entities της εφαρμογής μας

Εικόνα 3.5: Η δομή ενός μοντέλου μαζί με τις ετικέτες Room

Πάμε να εξηγήσουμε κάποιες βασικές έννοιες από το μοντέλο Users:
@ Entity:
Η προσθήκη αυτού του στοιχείου μέσα στο μοντέλο σημαίνει ότι θα δημιουργηθεί ένας αντίστοιχος πίνακας στη βάση δεδομένων (με το όνομα που δηλώνεται μέσα στην παρένθεση) και τα γνωρίσματα του πίνακα στη βάση δεδομένων καθορίζονται σύμφωνα με τους μεταβλητές της κλάσης. Με την λέξη "tableName" δηλώνουμε το όνομα του πίνακα που θα δημιουργηθεί.
@ColumnInfo:
Η ετικέτα αυτή καθορίζει τις στήλες (τα γνωρίσματα) του πίνακα που θα δημιουργηθεί στη βάση δεδομένων. Με την δεσμευμένη λέξη της βιβλιοθήκης μπορώ να ορίσω το όνομα της στήλης.
@ NonNull:
Εάν δεν θέλουμε η συγκεκριμένη στήλη του πίνακα να μείνει άδεια (κενή) προσθέτουμε την αντίστοιχη ετικέτα επάνω στην μεταβλητή της κλάσης.
@PrimaryKey:
Καθορίζει το πρωτεύον κλειδί του πίνακα. Θυμίζουμε ότι τα πρωτεύοντα κλειδιά στις βάσεις δεδομένων είναι τα δεδομένα του πίνακα που είναι μοναδικά για κάθε στοιχείο που θα προστεθεί στον πίνακα, κάτι το οποίο διευκολύνει την ανάκτηση των δεδομένων και αποτρέπει την απόκτηση λανθασμένων δεδομένων από τη βάση. Έχουμε την δεσμευμένη λέξη "**autoGenerate", όταν έχει την τιμή 1 αυξάνει την τιμή του πρωτεύοντος κλειδιού αυτόματα.**

DAO:

```
@Dao
interface UserDao {

    @Insert(onConflict = OnConflictStrategy.REPLACE)
    fun insertData(vararg users: UserEntity)

    @Delete
    fun deleteData(user: UserEntity)

    @Query("SELECT * FROM users")
    fun getData(): Flow<List<UserEntity>>

    @Update
    fun updateData(vararg users: UserEntity)

    @Query("SELECT * FROM users ORDER BY name DESC LIMIT 1")
    suspend fun getLastUser(): UserEntity?
```

Εικόνα 3.6: Η δομή του UserDao

Σε αυτό το σημείο προσθέτουμε τις διάφορες ενέργειες που θα κάνουμε στη βάση δεδομένων όπως η εισαγωγή νέων δεδομένων, η αναζήτηση και η διαγραφή. θα χρειαστούμε και την υλοποίηση των διεπαφών (Interfaces) για να έχουμε πιο ασφαλή πρόσβαση στη βάση. Επίσης μπορούμε να δημιουργήσουμε τα δικά μας ερωτήματα sql και να συσχετίσουμε αυτά με τους μεθόδους.

```
@Insert(onConflict = OnConflictStrategy.REPLACE)
fun insertData(vararg users: UserEntity)
```

Εικόνα 3.7: Η δομή της μεθόδου για εισαγωγή δεδομένων

@Insert(onConflict = onConflictStrategy.Replace):
Η χρήση αυτής της ετικέτας προσθέτει την επιπλέον λειτουργία στη μέθοδο, κάθε φορά που θα κάνει εισαγωγή των νέων δεδομένων στη βάση θα πάει να αντικαταστήσει τις παλιές που έχει και θα συνεχίσει τη διαδικασία της εισαγωγής των νέων δεδομένων.

@Query:

```
@Query("SELECT * FROM users")
fun getData(): Flow<List<UserEntity>>
```

Εικόνα 3.8 :Η δομή της μεθόδου

Με τη χρήση αυτής της ετικέτας μπορούμε να δημιουργήσουμε τα δικά μας προσαρμοσμένα ερωτήματα sql ως προς τη βάση και να τα συσχετίσουμε αυτά τα ερωτήματα με τη μέθοδο που έχω δημιουργήσει. Στην περίπτωση μας εδώ έχω μια μέθοδο που ονομάζεται **getData()**, δημιουργεί ένα αντικείμενο για κάθε χρήστη που φέρνει από τη βάση δεδομένων και τα αντικείμενα αυτά τα κρατάει σε μια λίστα. Κάθε φορά που θα κληθεί αυτή η συνάρτηση θα έχουμε μια λίστα από αντικείμενα τύπου **UserEntity (μοντέλο)**.

Database: Στο σημείο αυτό θα δημιουργήσουμε την αφηρημένη (Abstract) κλάση η οποία έχει όνομα RoomDb και κληρονομεί (extend) από την γενική κλάση RoomDatabase. Επάνω στην κλάση προσθέτουμε μια ετικέτα όπου αναφέρουμε όλα τα μοντέλα / οντότητες (Entitites) που θέλουμε να συμπεριληφθούν σε αυτήν. Με τη μεταβλητή **version** καθορίζουμε την εκδοχή της βάσης δεδομένων. Κάθε φορά που θα κάνουμε λειτουργικές αλλαγές πρέπει να αλλάξουμε και αυτό τον αριθμό ώστε να προσαρμοστούν οι καινούριες αλλαγές.

```
@Database(
    entities = [UserEntity::class, ResultUnit1Entity::class, ResultUnit2Entity::class,
        ResultUnit3Entity::class, ResultUnit4Entity::class, ResultUnit5Entity::class,
        ResultUnit6Entity::class, ResultUnit7Entity::class, ResultUnit8Entity::class],
    version = 4,
    exportSchema = false
)
abstract class RoomDb : RoomDatabase() {

    abstract fun quizDao(): UserDao
    abstract fun resultUnit1Dao(): ResultUnit1Dao
    abstract fun resultUnit2Dao(): ResultUnit2Dao
    abstract fun resultUnit3Dao(): ResultUnit3Dao
    abstract fun resultUnit4Dao(): ResultUnit4Dao
    abstract fun resultUnit5Dao(): ResultUnit5Dao
    abstract fun resultUnit6Dao(): ResultUnit6Dao
    abstract fun resultUnit7Dao(): ResultUnit7Dao
    abstract fun resultUnit8Dao(): ResultUnit8Dao

    companion object {
        @Volatile
        private var INSTANCE: RoomDb? = null

        fun getDatabase(context: Context): RoomDb {
            if (INSTANCE == null) {
                synchronized( lock: this) {
                    INSTANCE = buildDatabase(context)
                }
            }
            return INSTANCE!!
        }

        private fun buildDatabase(context: Context): RoomDb {
            return Room.databaseBuilder(
                context.applicationContext,
                RoomDb::class.java,
                 name: "UserRoomDatabase"
            )
                .fallbackToDestructiveMigration() // This line allow destructive migration
                .build()
```

Εικόνα 3.9: Η δομή της κλάσης Database

3.3.2 Τα πλεονεκτήματα της βιβλιοθήκης ROOM

- Κάθε Annotation (**@Query,@Entity**) ελέγχεται κατά τη διάρκεια της μεταγλώττισης, κάτι το οποίο προστατεύει την εφαρμογή μας από σφάλματα αλλά ταυτόχρονα ελέγχει και την ύπαρξη των πινάκων στη βάση.
- Ελαχιστοποιεί την επανάληψη του κώδικα και αποκλείει τα πιθανά λάθη κατά την δημιουργία της εφαρμογής.
- Εύκολη ενσωμάτωση με τις υπόλοιπες αρχιτεκτονικές (LiveData, ViewModel).

3.4 FIREBASE

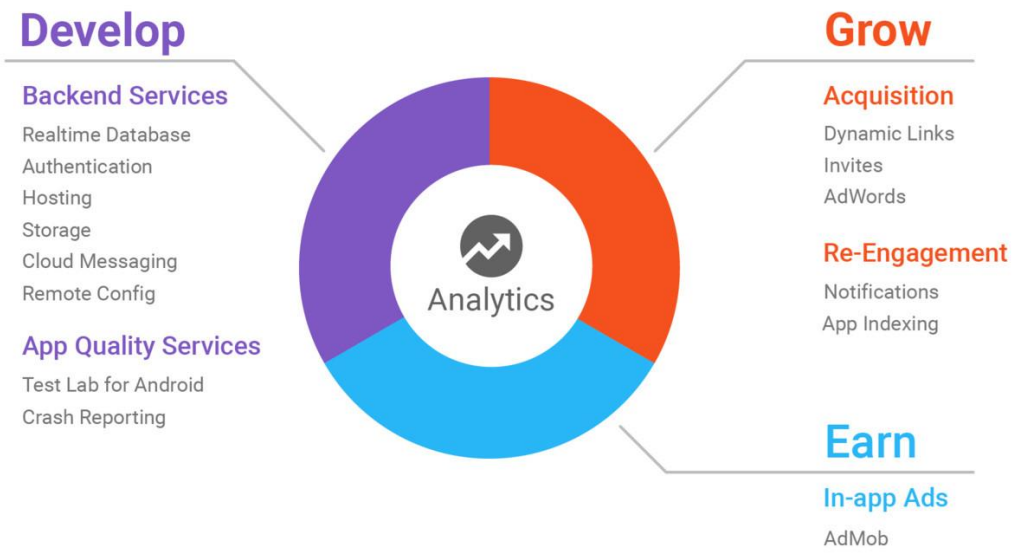
Εικόνα 3.10: Firebase Icon

Είναι μια πλατφόρμα που παρέχει υπηρεσίες backend για τους προγραμματιστές λογισμικού που αναπτύσσουν εφαρμογές ή προγράμματα σε τομείς (java, flutter, Android). Προσφέρει τη δυνατότητα απλοποίησης πολλών διεργασιών. Επιπλέον, υπάρχει η σημαντική επιλογή της συνέπειας ανάλυσης δεδομένων στις εμφανίσεις των χρηστών.

Ας δούμε λίγο την ιστορική εξέλιξη της πλατφόρμας.

Η πλατφόρμα ιδρύθηκε το 2011 από δύο startups. Το πρώτο όνομα εκείνη την εποχή ήτανε ENVOLVE. Ο κύριος σκοπός της πλατφόρμας ήτανε να μπορούν οι προγραμματιστές λογισμικού να προσθέσουν μια εφαρμογή συνομιλίας (CHAT) στις εφαρμογές τους με απλοϊκό τρόπο.

Το 2014 εξαγοράστηκε από την Google και προστέθηκαν επιπλέον λειτουργίες όπως:



Εικόνα 3.11: Firebase Services

Όπως φαίνεται στην παραπάνω εικόνα, υπάρχουν πολλές υπηρεσίες που παρέχει. Εκτός από την παροχή δωρεάν χρήσης, μπορούμε να έχουμε πρόσβαση στη βάση δεδομένων όπου καταχωρούνται τα δεδομένα χρήστη, εγγραφή, σύνδεση (login).

3.4.1 ΤΑ ΒΑΣΙΚΑ ΧΑΡΑΚΤΗΡΙΣΤΙΚΑ ΤΗΣ GOOGLE FIREBASE:

- Παράλληλη (σε πραγματικό χρόνο) αποθήκευση δεδομένων.
- Ταυτοποίηση χρηστών κατά την είσοδο
- Αποθήκευση δεδομένων (Υπηρεσίες Νέφους)
- Μηχανική μάθηση
- Ειδοποιήσεις (Notifications)
- Διαχειριστικό περιβάλλον (Admin panel)

3.4.2 Τι μπορούμε να κάνουμε μέσω Firebase:

- Μπορούμε να φτιάξουμε μια εφαρμογή συνομιλίας (Zoom, Telegram, Whatsapp)
- Μπορούμε να φτιάξουμε μια πλατφόρμα κοινωνικής δικτύωσης όπου οι χρήστες μπορούν να μοιράζονται δημοσιεύσεις / κοινοποιήσεις και οι άλλοι χρήστες να μπορούν να τα δουν αυτές τις κοινοποιήσεις άμεσα (Instagram, Facebook, Twitter).
- Μπορούμε να φτιάξουμε μια διαδικτυακή εφαρμογή διαγωνισμού όπου οι χρήστες ανταγωνίζονται μεταξύ τους.
- Μπορούμε να δημιουργήσουμε ένα σύστημα παρακολούθησης αποθέματος στο οποίο μπορούν να έχουν πρόσβαση ταυτόχρονα πολλά άτομα όσο από κινητές συσκευές αλλά όσο και από υπολογιστές.

3.4.3 Οι εφαρμογές που χρησιμοποιούν Firebase:

Εικόνα 3.12: Κάποιες εφαρμογές που χρησιμοποιούν Firebase

3.4.4 Οι υπηρεσίες της πλατφόρμας Firebase :

- **Authentication (Ταυτοποίηση χρήστη):** Μπορούμε να κάνουμε περιήγηση στις πληροφορίες των χρηστών που έχουν κάνει εγγραφή στην εφαρμογή μας. Επίσης μπορούμε να κάνουμε ταυτοποίηση των χρηστών. Μέσω της πλατφόρμας μπορούμε να δούμε τα στοιχεία των χρηστών όπως το email, την ημερομηνία εγγραφής. Με αυτή την υπηρεσία παρέχει λειτουργίες όπως επαλήθευση email, επαναφορά κωδικού πρόσβασης.

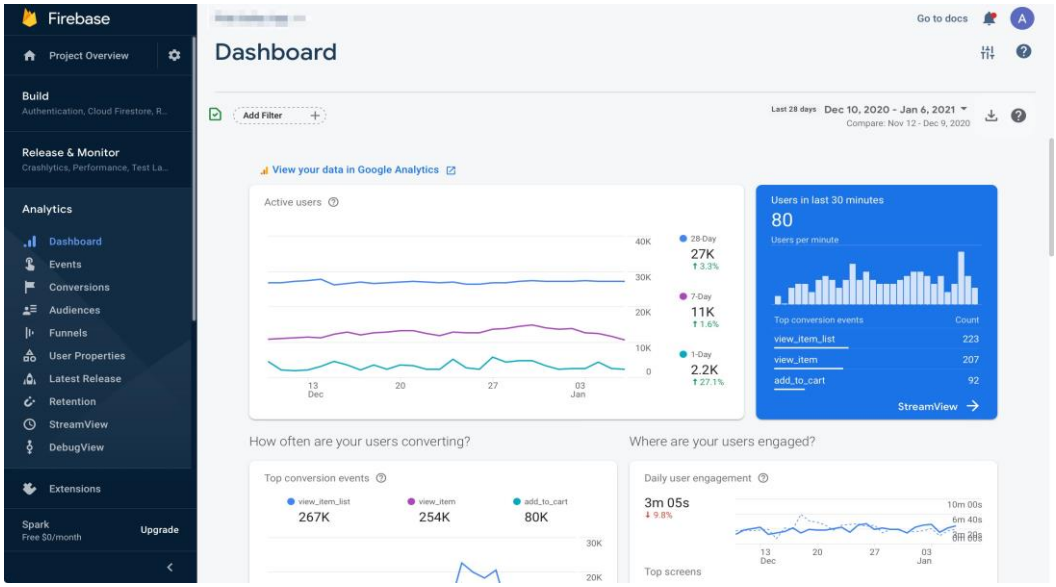
- **Database (Βάση δεδομένων):** Χάρη σε αυτή την υπηρεσία, το Firebase παρέχει στους χρήστες μια υπηρεσία βάσης δεδομένων NoSql ταυτόχρονης (σε πραγματικό χρόνο) πρόσβασης, που λειτουργεί σε ασύγχρονη (ξεχωριστή) δομή.

- **Storage (Μέσο αποθήκευσης):** Μέσω αυτής της υπηρεσίας μπορούμε να κάνουμε αποθήκευση ή αντιγραφή των δεδομένων (εικόνα η βίντεο) που έχουμε στον υπολογιστή μας στο νέφος (Cloud) και να έχουμε πρόσβαση από όπου θέλουμε.

- **Notification (Ειδοποιήσεις):** Εάν θέλουμε να επικοινωνήσουμε άμεσα με τους χρήστες της εφαρμογής μας, είτε στέλνοντας ειδοποιήσεις, είτε στέλνοντας μηνύματα, αυτή η υπηρεσία θα είναι η καλύτερη επιλογή.

- **Admob:** Μέσω αυτής της υπηρεσίας μπορούμε να προσθέσουμε διαφημίσεις της Google στην εφαρμογή μας και να κερδίσουμε χρήματα.

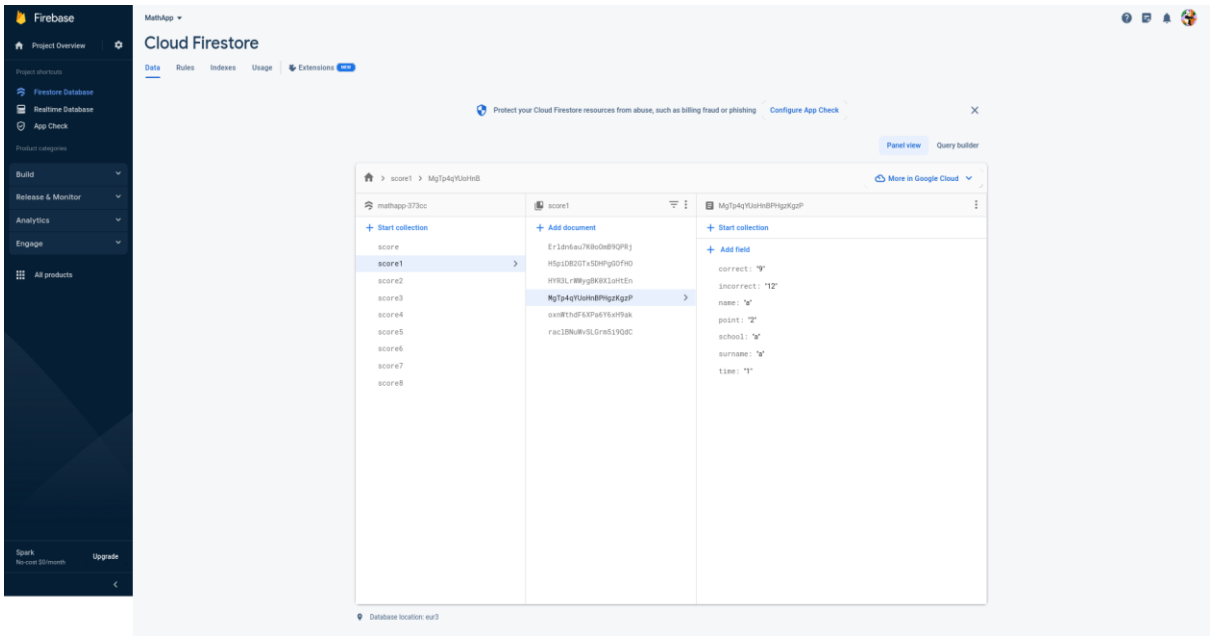
- **Firebase Analytics (Ανάλυση δεδομένων μέσω Firebase):** Με αυτή τη δομή, μπορούμε να έχουμε άμεση πρόσβαση σε πολλές πληροφορίες όπως ο αριθμός των ενεργών χρηστών, οι αλληλεπιδράσεις μεταξύ τους και πολλά άλλα στατιστικά που αφορούν τη λειτουργία της εφαρμογής μας.



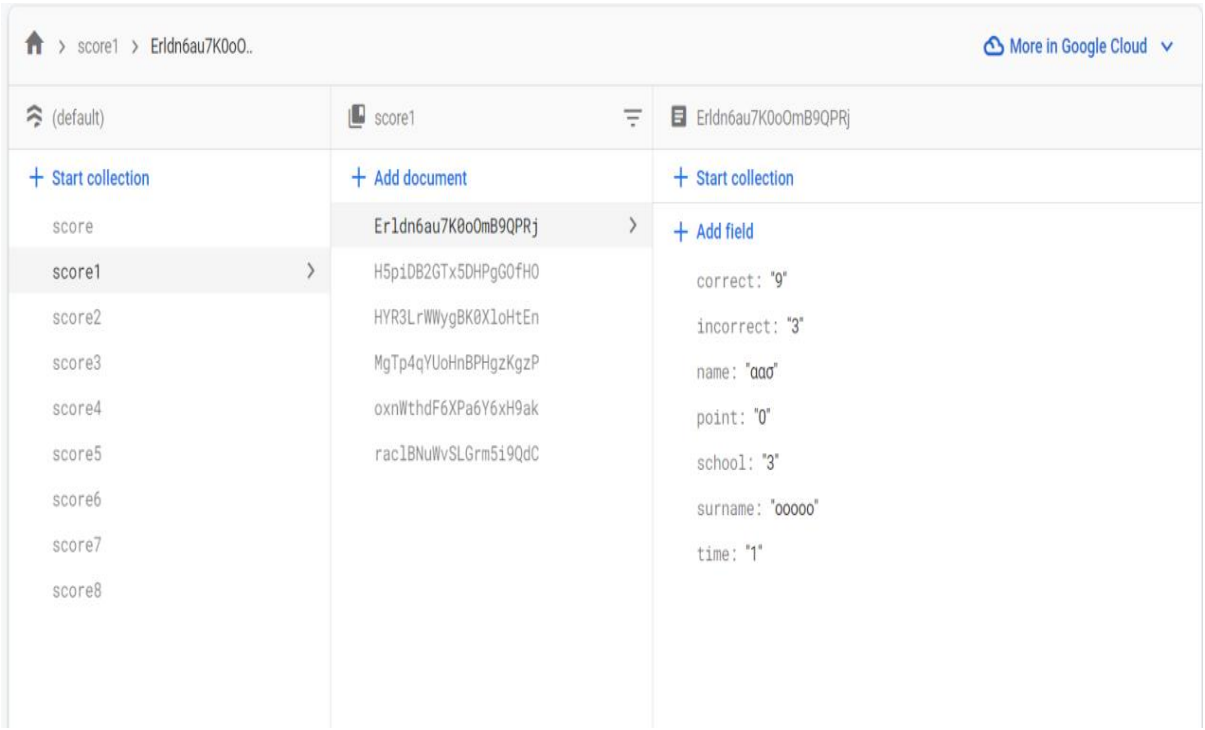
Εικόνα 3.13: Firebase Dashboard

```kotlin
package com.example.mathapp.framework.score.external

import ...


class ScoreExternalDataSourceImpl @Inject constructor(
    private val firestore: FirebaseFirestore
) : ScoreExternalDataSource {

    override suspend fun getScore(unit:Int): List<ResultModel> {
        val userModels = mutableListOf<ResultModel>()
        val collectionName = "score$unit"
        val querySnapshot = firestore.collection(collectionName).get().await()
        for (document in querySnapshot.documents) {
            val name = document.getString( field: "name") ?: ""
            val surname = document.getString( field: "surname") ?: ""
            val school = document.getString( field: "school") ?: ""
            val correct = document.getString( field: "correct") ?: ""
            val incorrect = document.getString( field: "incorrect") ?: ""
            val time = document.getString( field: "time") ?: ""
            val point = document.getString( field: "point") ?: ""
            val userModel = ResultModel(name = name,surname = surname,school = school,correct = correct,incorrect = incorrect,time = time, point = point)
            userModels.add(userModel)
        }
        return userModels
    }
}
```

Εικόνα 3.14: Κώδικας Firebase



Εικόνα 3.15: Firestore Database



Εικόνα 3.16: Firestore Databese score1

Στην παραπάνω εικόνα βλέπουμε τη δομή των δεδομένων στο περιβάλλον Firestore. Στην πρώτη στήλη γίνεται η ταξινόμηση των βαθμολογιών των χρηστών ανά ενότητα. Η δεύτερη στήλη αντιστοιχεί σε string id του κάθε χρήστη και στην τελευταία στήλη έχουμε τη αναλυτική παρουσίαση των στοιχείων του χρήστη.

Επίσης για την συνομιλία των χρηστών της εφαρμογής έχει δημιουργηθεί μια βάση πραγματικού χρόνου(Real Time) και έχει την δομή που εμφανίζεται στην παρακάτω εικόνα:

Εικόνα 3.17: RealTime Database

3.5 JETPACK COMPOSE

Εικόνα 3.18: Jetpack Compose

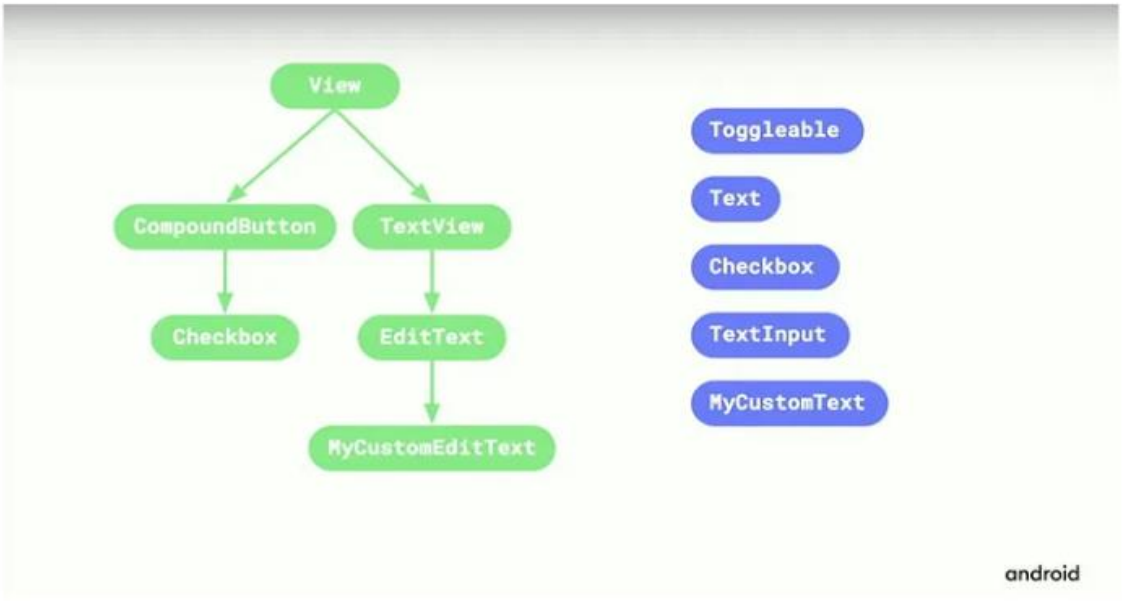
Το Jetpack Compose είναι ένα σύγχρονο εργαλείο για την ανάπτυξη των γραφικών διεπαφών για τις εφαρμογές των κινητών συσκευών και αναπτύχθηκε από τους προγραμματιστές της Google.
Ο ορισμός που δίνεται στην επίσημη ιστοσελίδα της Google:

> Είναι μια σύγχρονη εργαλειοθήκη (Toolkit) για την δημιουργία εγγενών(Native) γραφικών διεπαφών (UI).Το Jetpack Compose απλοποιεί και επιταχύνει την ανάπτυξη της διεπαφής χρήστη στις εφαρμογές. Υλοποιήστε την εφαρμογή σας με ταχύτερο τρόπο, λιγότερο κώδικα και με ισχυρά εργαλεία που παρέχει η γλώσσα Kotlin.

Το Jetpack Compose προσφέρει τα παρακάτω πλεονεκτήματα:
Λιγότερος κώδικας: Με αυτό τον τρόπο αποφεύγουμε τα λάθη που θα προκύψουν κατά τη διάρκεια της ανάπτυξης της εφαρμογής, απλούστερος κώδικας και εύκολη συντήρηση**.**
Διαισθητικότητα: Σχεδιάζουμε μόνο την διεπαφή χρήστη και τις υπόλοιπες λειτουργίες τα αναλαμβάνει το Compose. Κάθε φορά αλλάζει κατάσταση της διεπαφής ενημερώνεται το compose αυτόματα.

Γρήγορη σχεδίαση: Είναι συμβατό με τα υπάρχοντα εργαλεία. Μπορούμε να προβάλουμε γρήγορα τα αποτελέσματα με ζωντανή προεπισκόπηση.

Πλήρης έλεγχος: Δημιουργία διαδραστικών εφαρμογών με άμεση πρόσβαση στα API της τεχνολογίας και ενσωματωμένη υποστήριξη για τα Material Design.



View Hierarchy

Εικόνα 3.19: View Hierarchy

Κατά την ανάπτυξη της εφαρμογής, τα στοιχεία προβολής που είναι διαθέσιμα για την ανάπτυξη της διεπαφής του χρήστη τηρούν μια ιεραρχία (Κληρονομούνται από πάνω προς τα κάτω). Για παράδειγμα, το FrameLayot κληρονομεί από το ViewGroup ή το TextView κληρονομείται από το View. Στο Jetpack Compose τα πράγματα είναι λίγο διαφορετικά.
Στην kotlin οι συναρτήσεις δεν πρέπει δεσμεύονται με τις κλάσεις αλλά μπορούν να κληθούν σαν συναρτήσεις οπουδήποτε μέσα στην εφαρμογή μας.

Top Level Widget

Εικόνα 3.20: Compose Example

Στην ανάπτυξη εφαρμογών προτού να περάσουμε στη γλώσσα Kotlin, την διεπαφή του χρήστη την γράφαμε σε xml μορφή και αργότερα κάναμε τη σύνδεση (inflate) με κάποια δραστηριότητα (Activity). Στο JetPacK Compose τα πράγματα είναι λίγο διαφορετικά. Πλέον έχει αλλάξει η λογική

αυτή και έχουμε κάποιες συναρτήσεις που αλληλεπιδρούν με τα εργαλεία που είναι διαθέσιμα επάνω στην διεπαφή του χρήστη. Η αλληλεπίδραση με την οπτική διεπαφή μπορεί να γίνει απ' οπουδήποτε μέσω κατάλληλων κλήσεων συναρτήσεων.

Τι είναι μια @Composable συνάρτηση;

Ένας από τους στόχους του JetPack Compose είναι να αποκτήσει μια αρθρωτή (Modular) δομή. Κατά τη δημιουργία αυτής της δομής , συμπληρώνουμε τις ιδιότητες του κάθε στοιχείου προβολής κατάλληλα για την εμφάνισης του επιθυμητού αποτελέσματος.

Αν πρόκειται να δημιουργήσουμε μια προβολή κειμένου, ορίζουμε τις ιδιότητες του κειμένου, όπως το χρώμα, την γραμματοσειρά ή το μέγεθος. Αυτές οι συναρτήσεις ονομάζονται συνθετικές (Composable) συναρτήσεις. Η ετικέτα @Composable δηλώνει ότι πρόκειται για μια συνάρτηση συνθετική (Composable) και διαθέτει αυτή η συνάρτηση τουλάχιστον ένα στοιχείο προβολής (view).

Όταν προστεθεί η επεξήγηση @Preview μπορούμε να δούμε γρήγορα την έξοδο της σύνθετης (composable) συνάρτησης που δημιουργήθηκε στα δεξιά.

```
47  @Composable
48  fun HomeScreen(viewModel: LobbyViewModel) {
49      val musicEnabled = remember { mutableStateOf(SettingsManager.isMusicEnabled()) }
50      val examBoolean by viewModel.dashboardGetExamBool().collectAsState(initial = false)
51      val context = LocalContext.current
52      val switchText = if (musicEnabled.value) { R.string.dashboard_music_enable_text} else { R.string.dashboard_musi
53
54      Column(
55          modifier = Modifier
56              .fillMaxWidth()
57              .background(BabyBluePurple2)
58              .padding(horizontal = SpacingCustom_20dp),
59          verticalArrangement = Arrangement.Center
60      ) {
61          Row(
62              modifier = Modifier.fillMaxWidth(),
63              horizontalArrangement = Arrangement.Center,
64              verticalAlignment = Alignment.CenterVertically
65          ) {
66              Text(
67                  maxLines = 1,
68                  text = stringResource(id = R.string.app_name),
69                  color = BabyBluePurple5,
70                  fontWeight = FontWeight.Bold,
71                  style = MaterialTheme.typography.displayLarge,
72                  fontFamily = FontFamily.Cursive,
73                  modifier = Modifier.padding(SpacingDefault_16dp)
74              )
```

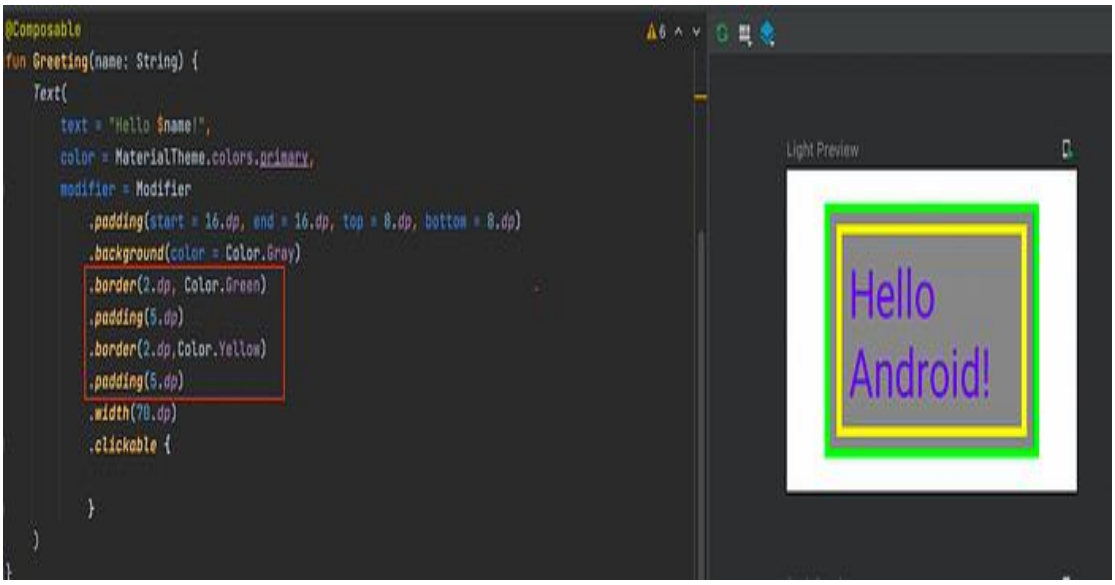
Εικόνα 3.21: Compose HomeScreen

@row: Αντιστοιχεί LinearLayout orientation = vertical. Ταξινομεί όλα τα στοιχεία που υπάρχουν επάνω στην γραφική διεπαφή στήλη προς στήλη.

@Column: Αντιστοιχεί LinearLayout orientation = horizontal. Ταξινομεί όλα τα στοιχεία που υπάρχουν επάνω στην γραφική διεπαφή γραμμή προς γραμμή.

Modifier: Είναι η μέθοδος που μας βοηθάει για την στοίχιση των οπτικών στοιχείων (δίνοντας τιμές στους διάφορους παραμέτρους της συνάρτησης όπως είναι το padding,) που εμφανίζονται στην γραφική

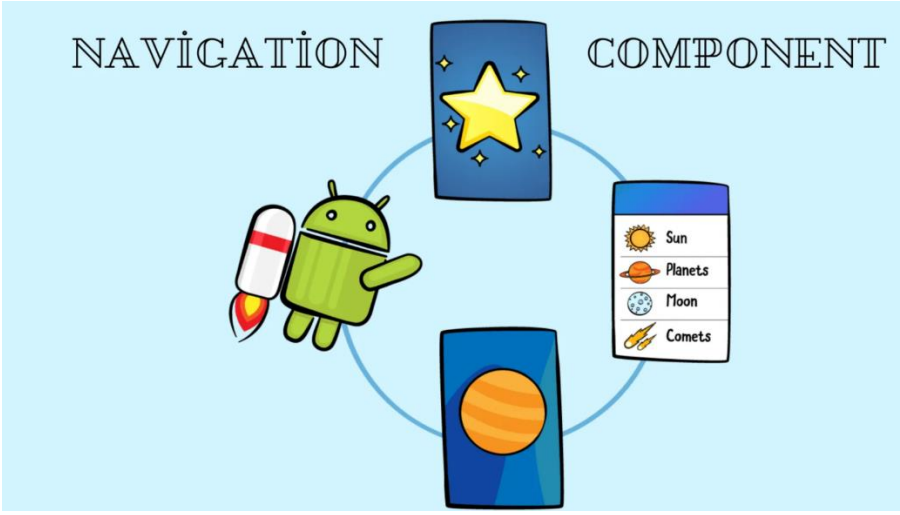
διεπαφή του χρήστη.



Εικόνα 3.22: Modifier Example

Αφού έχουμε καταλάβει τα βασικά ως προς τη στοίχιση επάνω στην γραφική διεπαφή, πάμε να δούμε τώρα την έννοια της διαισθητικότητας. Διαισθητικό σημαίνει όταν το περιεχόμενο ενός μοντέλου ή μιας μεταβλητής υφίσταται οποιαδήποτε αλλαγή κατάστασης και τότε η κάθε οπτική διεπαφή του χρήστη που συσχετίζεται με το μοντέλο αλλάζει με αυτόματο τρόπο, χωρίς να κάνουμε εμείς σε κάποια ενέργεια. Αυτό επιτυγχάνεται με την ετικέτα μέσα στο Μοντέλο με χρήση Modifier.

3.6 NAVIGATION COMPONENT



Εικόνα 3.23: Navigation Component Icon

Το Navigation Component είναι μια από τις βιβλιοθήκες Android Jetpack που μας προσφέρει η Google. Το Navigation Component είναι μία από τις λειτουργίες του λειτουργικού συστήματος που μας διευκολύνει στις μεταβάσεις και τις εναλλαγές μεταξύ των οθονών των εφαρμογών και τη ροή δεδομένων μεταξύ τους.

Το Navigation Graph (Γράφημα μεταβάσεων) είναι ένα xml αρχείο όπου εμπεριέχει την συνολική εικόνα μεταξύ των διεπαφών του χρήστη και της ροής των δεδομένων.

Το εργαλείο Navigation αποτελείται από τρία βασικά σημεία:

1. **Navigation Graph** (Γραφικό περιβάλλον για τη σχεδίαση): είναι ένα αρχείο xml που περιέχει όλα τα μονοπάτια πλοήγησης που μπορεί να ακολουθήσει ο χρήστης μέσα στην εφαρμογή μας.

2. **NavHostFragment** (Δηλώνει τους προορισμούς στο γράφημα): είναι ο προορισμός, συνήθως η κενή καρτέλα (container fragment) όπου θα τοποθετηθεί το Fragment.

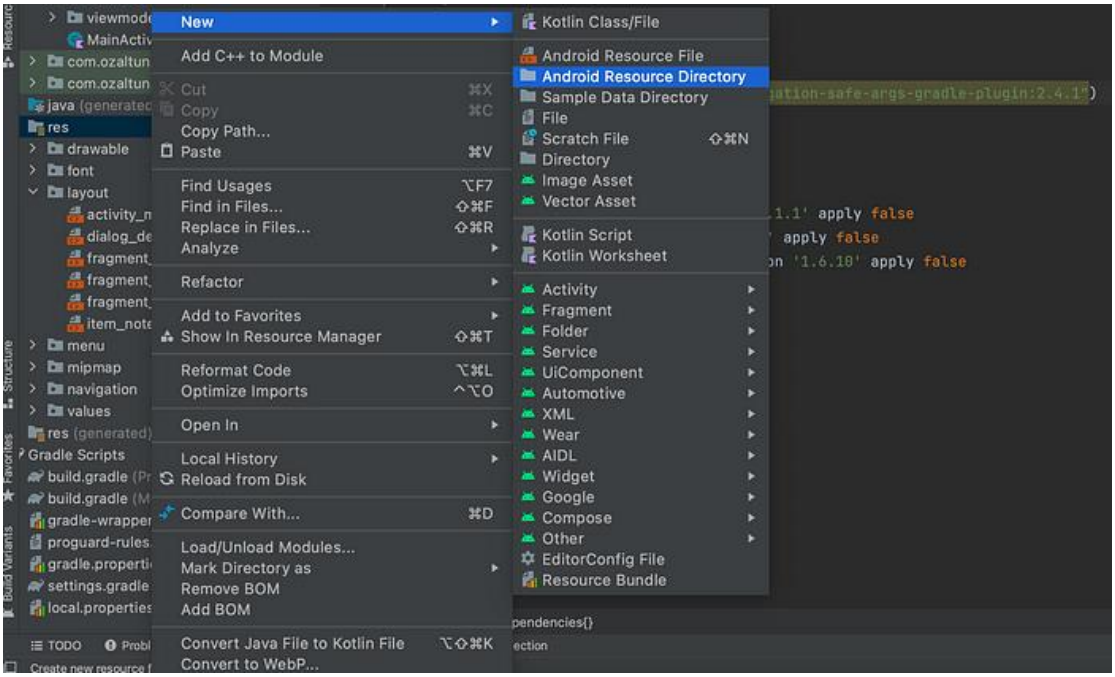
3. **NavController**(): είναι η μέθοδος που διαχειρίζεται τις μεταβάσεις μεταξύ των fragments.

Υπάρχουν πολλοί τρόποι για να παρέχουμε τα δεδομένα που χρειαζόμαστε σε μια οθόνη (Intent-Bundle, Shared Preferences, Content Provider). Το στοιχείο Navigation προσθέτει ακόμα ένα καινούριο στοιχείο σε αυτή τη λίστα, το Safe Args. Με αυτή τη δυνατότητα, μπορούμε να προσδιορίσουμε τα δεδομένα που χρειαζόμαστε στις οθόνες μέσω του γραφήματος πλοήγησης.

Τα βήματα που ακολουθούμε κατά τη δημιουργία Navigation Component :

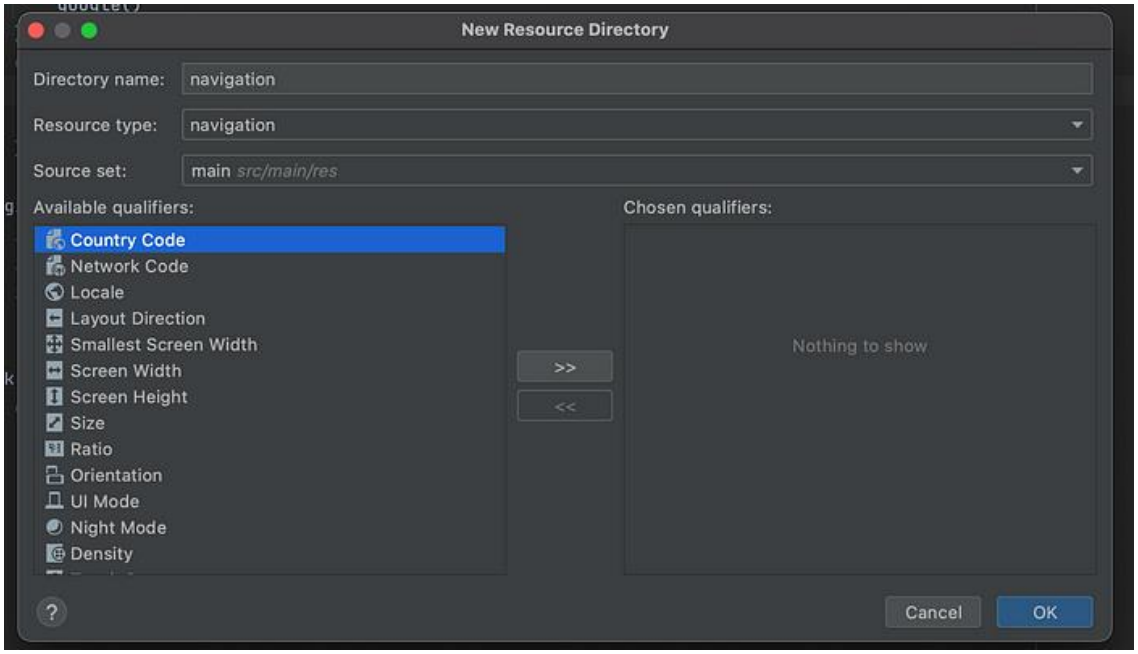
ΒΗΜΑ 1: Αρχικά πρέπει να δημιουργηθούν τα Fragments.

ΒΗΜΑ 2: Δεξί κλίκ στο φάκελο res και δημιουργία αρχείου “Android Resource Directory ”.



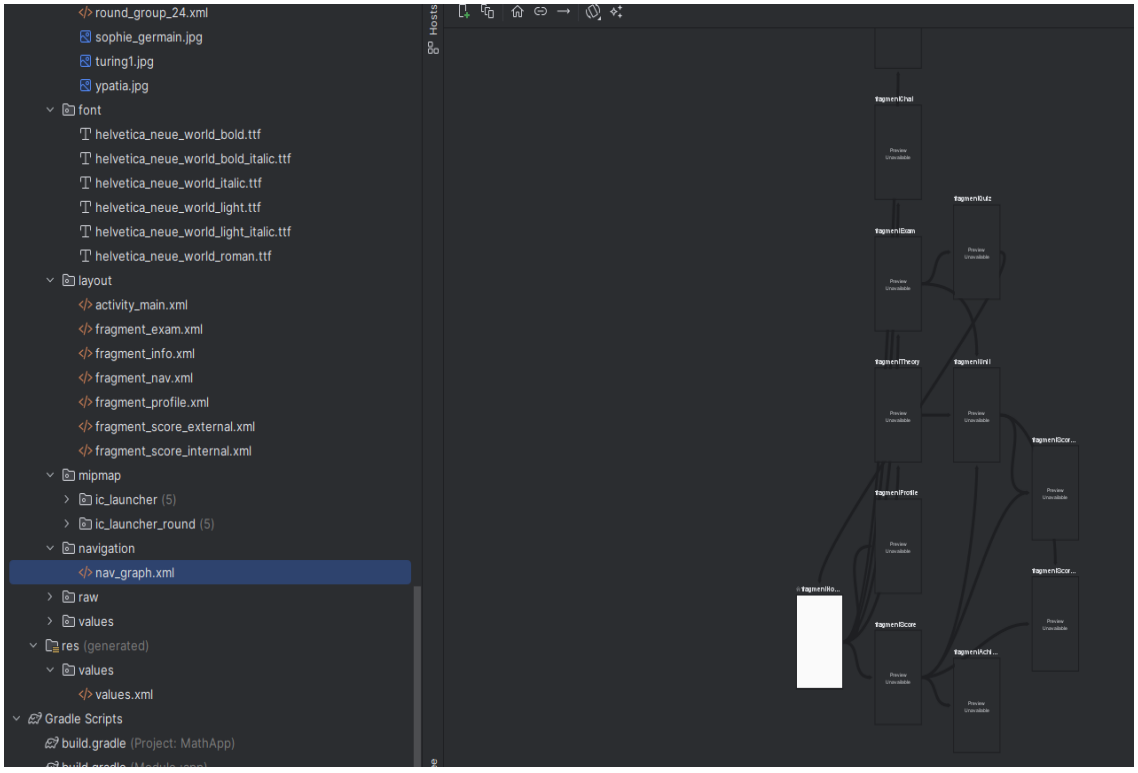
Εικόνα 3.24: Fragment Create Example

ΒΗΜΑ 3: Στο παράθυρο που μου εμφανίζεται δηλώνω το όνομα του φακέλου”Navigation”.



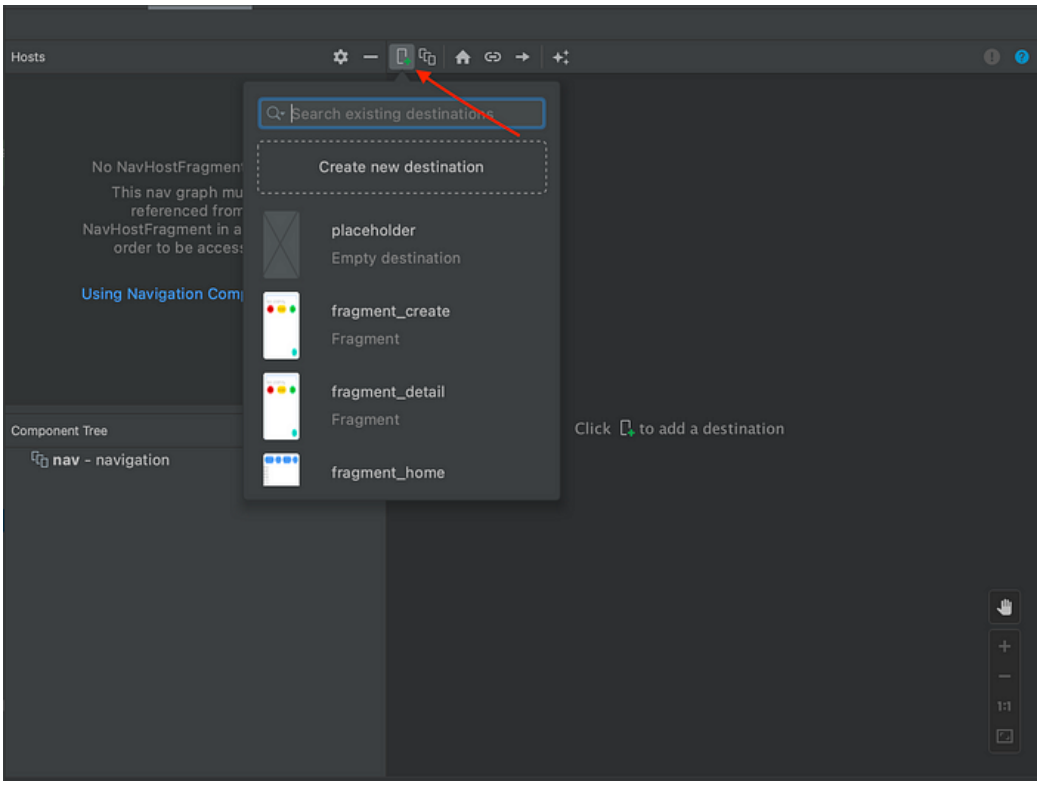
Εικόνα 3.25: Fragment Nav Example

ΒΗΜΑ 4: Κάτω από το navigation θα δημιουργηθεί ένα xml αρχείο nav_graph .



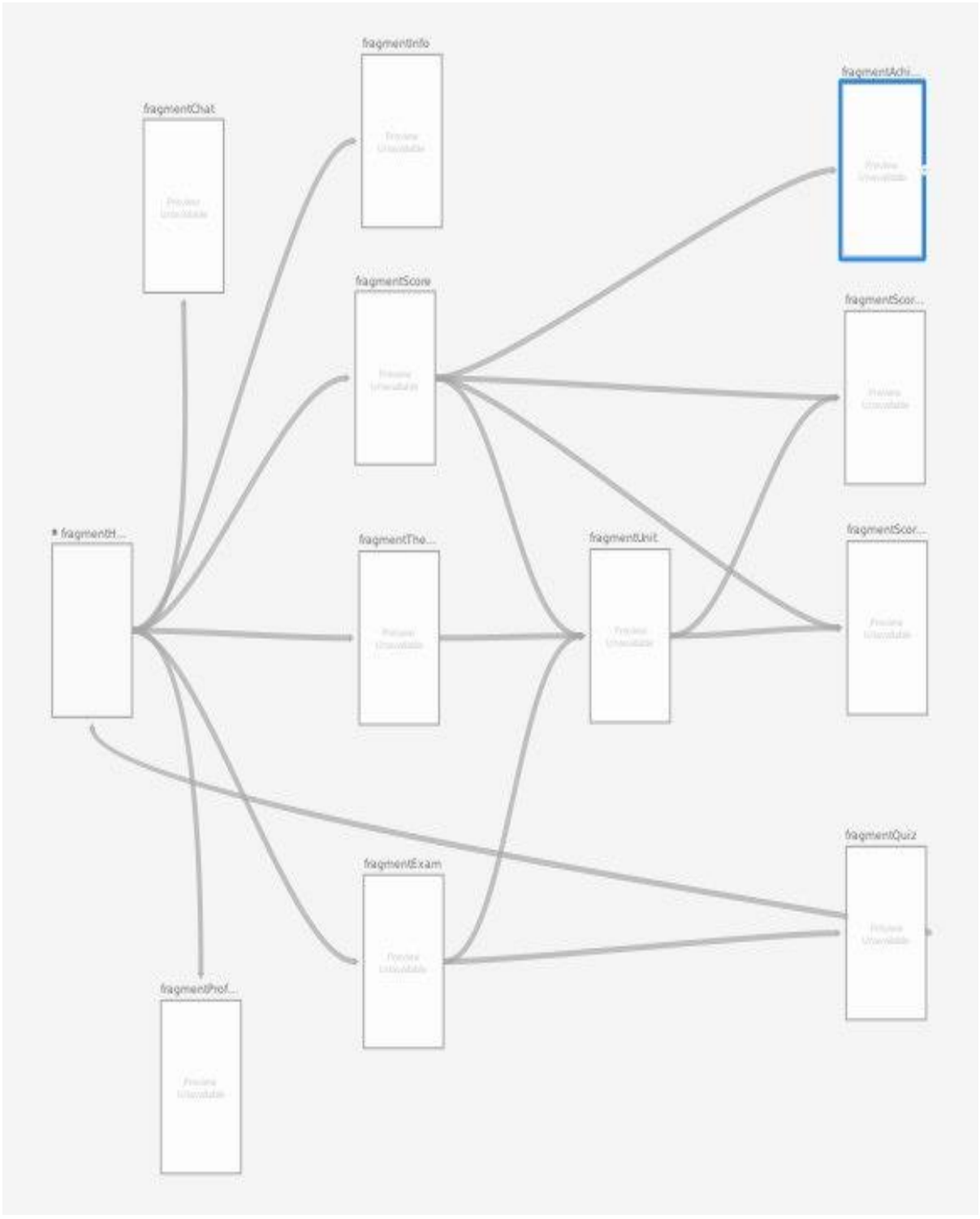
Εικόνα 3.26: nav_graph.xml in navigation folder

ΒΗΜΑ 5: Στο σημείο αυτό προσθέτουμε τα fragments που έχουμε δημιουργήσει νωρίτερα.



Εικόνα 3.27: Navigation Add Component Example

ΒΗΜΑ 6: Στο σημείο αυτό αφού έχουμε ολοκληρώσει τη διαδικασία της προσθήκης των Fragments μπορούν να δημιουργηθούν μονοπάτια μεταξύ τους. Για κάθε διαδρομή που θα σχηματιστεί μεταξύ δύο Fagments θα δοθεί ένα μοναδικό αναγνωριστικό(Id). Αργότερα, θα κάνουμε τον έλεγχο κάθε σελίδας μέσω αυτού του μοναδικού αριθμού.



Εικόνα 3.28: Complete Navigation Graph

3.7 HILT

3.7.1 Τι είναι το Hilt;

Εικόνα 3.29: Hilt Dagger Icon

Σε αυτό το σημείο θα μιλήσουμε για τη βιβλιοθήκη Hilt, η οποία είναι μια από τις βιβλιοθήκες Dependency Injection στο Android. Η βιβλιοθήκη Hilt είναι μια βιβλιοθήκη που προτείνεται από την Jetpack και είναι χτισμένη στο Dagger. Ο πιο σημαντικός λόγος για τον οποίο προτιμάμε τη βιβλιοθήκη Hilt, είναι ότι μπορεί να ενσωματωθεί πιο εύκολα από το Dagger. Μειώνει το μέγεθος του περιττού κώδικα (Boilerplate Code).

Τα πλεονεκτήματα της βιβλιοθήκης:

- Η Επεκτασιμότητα
- Υψηλός χρόνος εκτέλεσης
- Γρήγορες αρχικές ρυθμίσεις
- Επαναχρησιμοποίηση του κώδικα
- Εύκολη διόρθωση κώδικα

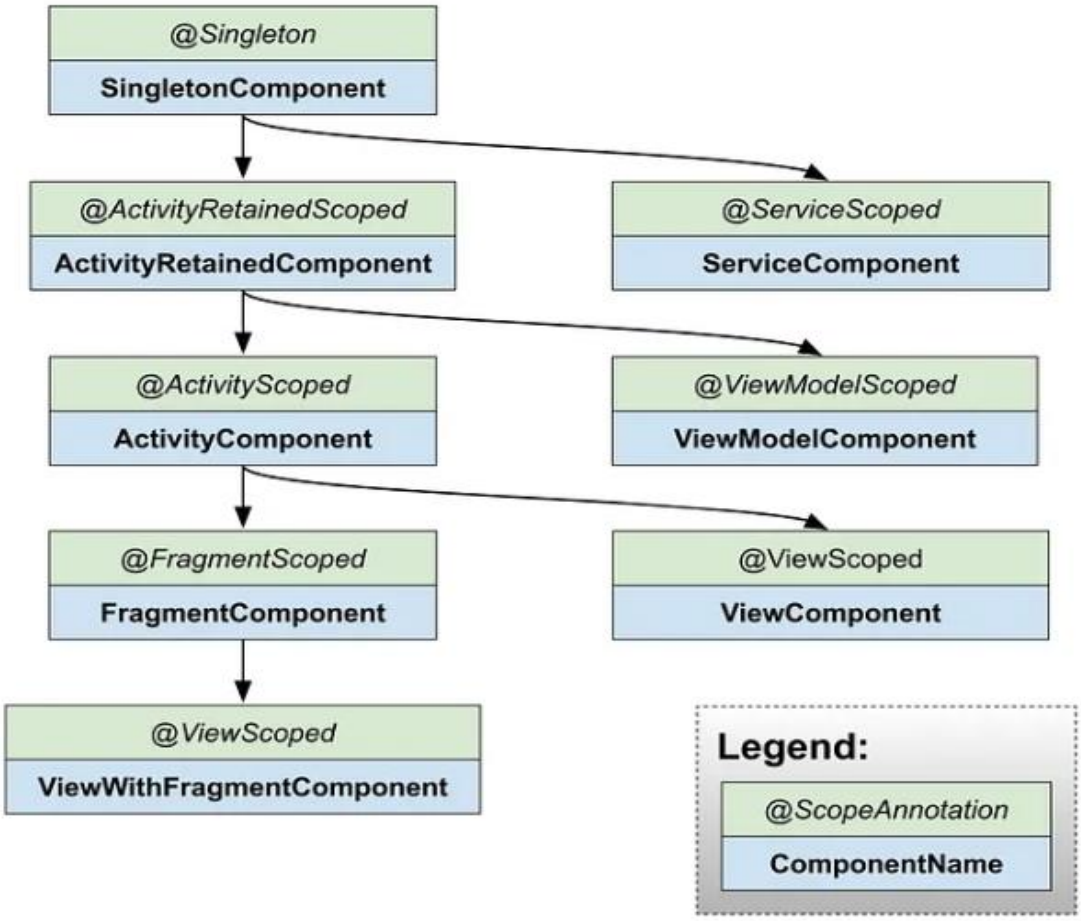
Με τη βιβλιοθήκη αυτή οι εξαρτήσεις που υπάρχουν στον κώδικα μας πρέπει να καθοριστούν με χειροκίνητο τρόπο, κάτι το οποίο αυξάνει το μέγεθος του κώδικα. Ένας από τους κυριότερους λόγους που χρησιμοποιούμε τη βιβλιοθήκη Hilt είναι οι ετικέτες (Annotations). Πρόκειται για το εργαλείο που βοηθάει στην εξοικονόμηση χρόνου.
Μπορεί να ενσωματωθεί εύκολα με τη βιβλιοθήκη Jetpack.

Οι βιβλιοθήκες Jetpack συμβατές με τη βιβλιθήκη Hilt:

- ViewModel
- Navigation
- Compose
- WorkManager

Οι λειτουργίες του λειτουργικού συστήματος android που υποστηρίζονται από τη βιβλιοθήκη hilt:

- Activity
- Fragment
- View
- Service
- BroadcastReciever



Εικόνα 3.30: Hilt Dagger Explain

@ Inject:
Χρησιμοποιούμε τις ετικέτες (annotations) για να εισάγουμε εξαρτήσεις στις εξαρτώμενες κλάσεις μας. Οι εξαρτήσεις μπορούν να προστεθούν στα σημεία όπου δηλώνουμε τους δημιουργούς ή τις μεθόδους όπου δηλώνουμε τις εξαρτήσεις.
@ Module:
Προστίθενται στην αρχή των κλάσεων, εκεί όπου θα δημιουργηθεί ένα αντικείμενο μέσα στη κλάση.
@ Installn: Με βάση αυτή την ετικέτα καθορίζουμε το Module που έχουμε και σε ποιο Component θα εμφανιστεί. Κάθε Component έχει ένα κύκλο ζωής.
@ Provides: Η ετικέτα αυτή χρησιμοποιείται επάνω στις μεθόδους των αντικειμένων που έχουν δημιουργηθεί**.**

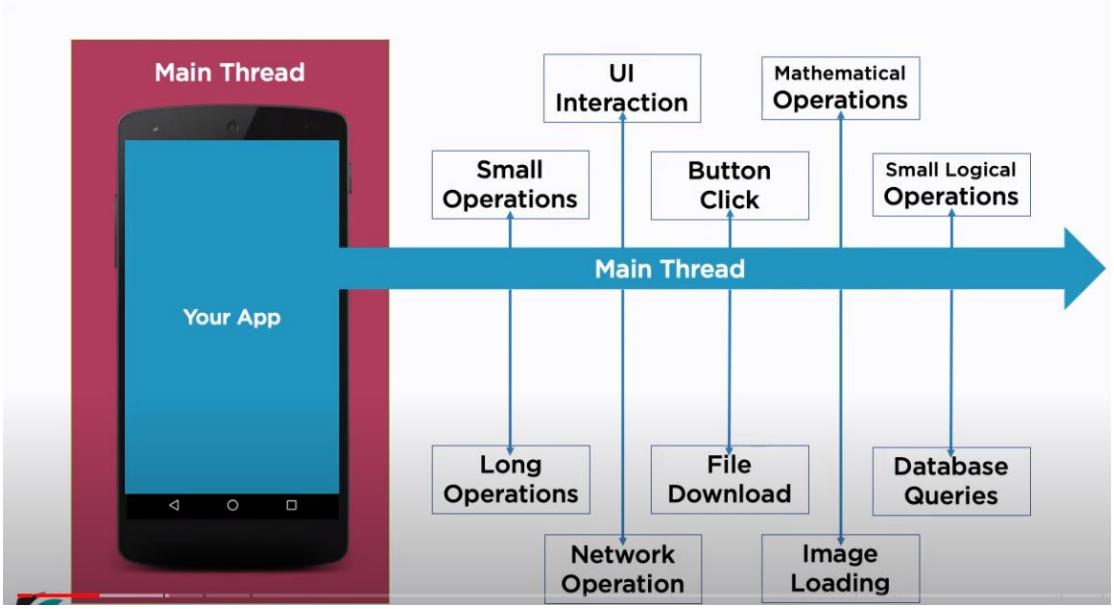
3.8 COROUTINES

3.8.1 Τι είναι τα coroutines;

Όταν ο χρήστης εκκινεί την εφαρμογή, δημιουργείται ένα προεπιλεγμένο νήμα. Αυτό το νήμα ονομάζεται το **Κύριο Νήμα (Main Thread)**. Αυτό το κύριο νήμα έχει μια διάρκεια ζωής (διαρκεί για κάποια δευτερόλεπτα). Στο παρακάτω σχήμα αναπαριστούμε το κύριο νήμα με το μπλε βέλος. Στο νήμα αυτό εκτελούμε πολύ μικρές και ελαφριές λειτουργίες (Αλληλεπίδραση με τον χρήστη).

Οι αλληλεπιδράσεις αυτές είναι όπως:

- *Μικρές διεργασίες.*
- *Αλληλεπιδράσεις διεπαφής χρήστη.*
- *Πάτημα των κουμπιών.*
- *Μαθηματικές Πράξεις .*
- *Απλές λογικές πράξεις.*

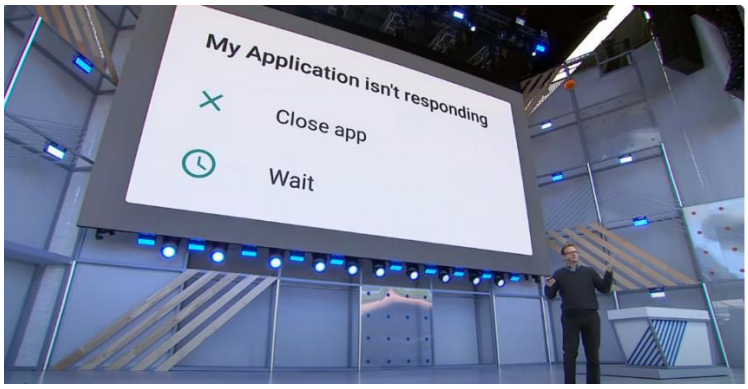


Εικόνα 3.31: Main Thread on My App

Τι γίνεται όμως με τις βαριές διεργασίες που εκτελούνται σε αυτό το νήμα όπως:

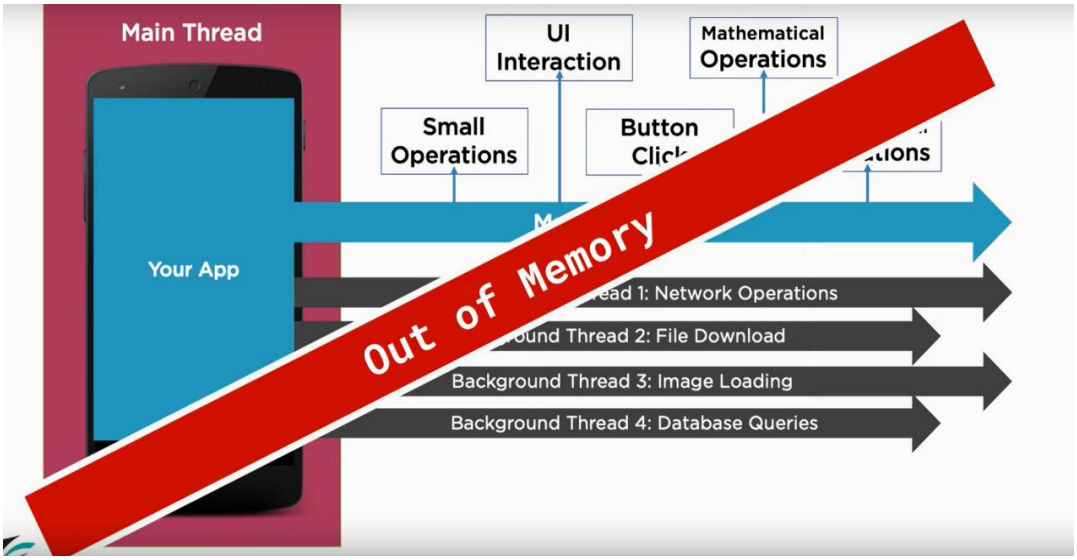
- *Μεγάλες διεργασίες.*
- *Υπηρεσίες Διαδικτύου.*
- *Κατέβασμα αρχείων .*
- *Φόρτωση εικόνας .*
- *Ερωτήματα (queries) ως προς τη βάση.*

Εάν εκτελέσουμε αυτές τις βαριές λειτουργίες επάνω στο κύριο νήμα, τότε το κύριο νήμα μετά από κάποια δευτερόλεπτα θα είναι εκτός λειτουργίας (θα μπλοκάρει από το λειτουργικό). Σε μια τέτοια περίπτωση η εφαρμογή μας θα παγώσει και το πιο πιθανό ένα αποκλειστεί για παραπάνω από 50 δευτερόλεπτα. Θα μας εμφανίσει το σφάλμα που εμφανίζεται στην παρακάτω εικόνα.



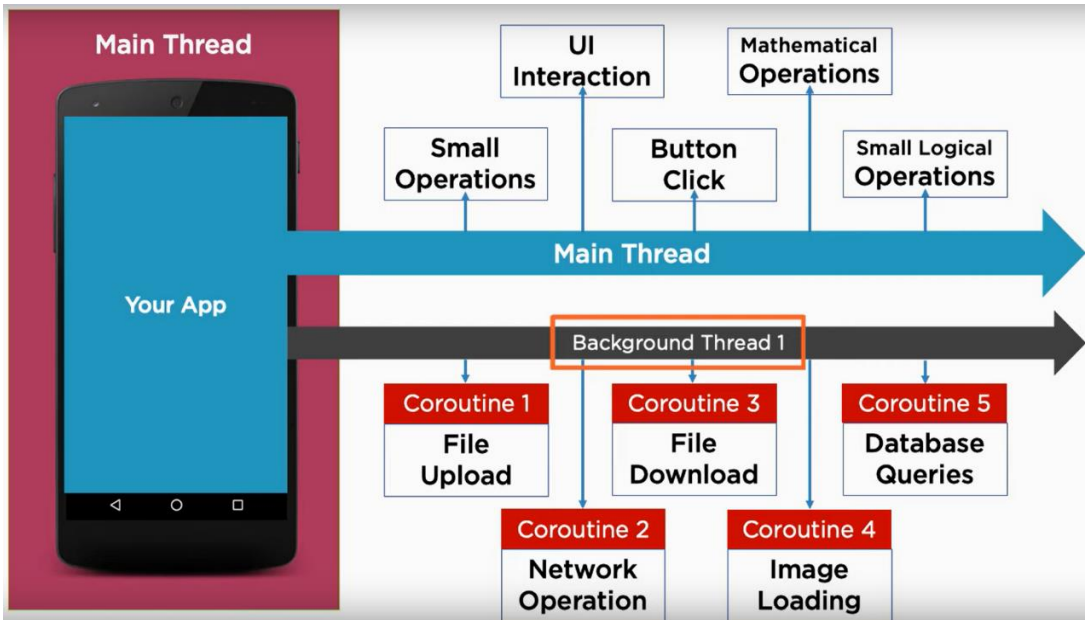
Εικόνα 3.32: Application not responding

Η εκτέλεση των βαρέων διεργασιών επάνω στο κύριο νήμα είναι μια κακή ιδέα. Για την αντιμετώπιση αυτής της κατάστασης υπάρχει λύση. Η λύση αυτή είναι δημιουργία κάποιου δευτερεύον νήματος (background thread) που τρέχει με ασύγχρονο τρόπο και δεν επιβαρύνει τις λειτουργίες του κύριου νήματος. Μια λογική για την αντιμετώπιση αυτής της κατάστασης, είναι η δημιουργία ξεχωριστού νήματος για κάθε βαριά διεργασία που εκτελείται στην εφαρμογή μας. Για κάθε μια καινούρια διεργασία δημιουργείται και ένα αντίστοιχο νήμα χωρίς να επηρεάζεται η λειτουργία του κύριου νήματος.
Σαν λύση μπορεί να ακούγεται ότι είναι αποτελεσματική, αλλά υπάρχει ένα άνω όριο στον αριθμό την νημάτων που μπορούμε να δημιουργήσουμε κάθε φορά. Το κόστος δημιουργίας νημάτων είναι **πάρα πολύ ακριβή και** όσο περισσότερα νήματα δημιουργούνται, τότε θα υπάρξει μια στιγμή που η μνήμη της συσκευής θα **εξαντληθεί. Είναι μια κακή ιδέα.**



Εικόνα 3.33: Threads out of memory

Σε αυτό το σημείο οι coroutines λύνουν τα χέρια των προγραμματιστών. Όταν χρησιμοποιούμε coroutines, τότε δεν χρειάζεται να δημιουργούμε τόσα νήματα για κάθε λειτουργία. Αντίθετα, μπορούμε να έχουμε μόνο ένα νήμα φόντου και σε αυτό το φόντο μπορούμε να εκκινήσουμε coroutines όπως φαίνεται στην παρακάτω εικόνα.



Εικόνα 3.34: Background Thread

- ***Coroutine1 (File Upload):*** εκτελεί τη λειτουργία φόρτωσης του αρχείου
- ***Coroutine2 (File Download):*** εκτελεί τη λειτουργία λήψης του αρχείου
- ***Coroutine3 (Network operations):*** εκτελεί τις λειτουργίες διαδικτύου.
- ***Coroutine4 (Image Loadings)****:* εκτελεί τη λειτουργία φόρτωσης εικόνων.

Άρα μπορούμε να έχουμε περισσότερες coroutines για να εκτελέσουμε άλλες λειτουργίες. Εν ολίγοις, με την κατανάλωση ενός νήματος φόντου, μπορούμε να εκτελέσουμε τόσες πολλές βαριές λειτουργίες.

Τα χαρακτηριστικά των coroutines:

- Είναι πιο ελαφριά από τα νήματα
- Όπως τα νήματα μπορεί να τρέχουν παράλληλα να περιμένουν η μία την άλλη να ολοκληρώσουν τις λειτουργίες τους ακόμα και να έχουν επικοινωνία μεταξύ τους .
- Δεν είναι νήματα (Threads)
- Είναι πάρα πολύ φτηνές. Είναι σχεδόν δωρεάν και δεν μας απασχολεί το θέμα της μνήμης.

ΚΕΦΑΛΑΙΟ 4: ΑΝΑΠΤΥΞΗ ΤΗΣ ΕΦΑΡΜΟΓΗΣ

Στο κεφάλαιο αυτό θα γίνει περαιτέρω ανάλυση και επεξήγηση των λειτουργιών της εκπαιδευτικής εφαρμογής. Θα γίνουν αναφορές στα κρίσιμα σημεία της υλοποίησης και με χρήση του κατάλληλου οπτικού υλικού σκοπεύουμε την καλύτερη κατανόηση από τη μεριά των χρηστών. Σκοπός της συγκεκριμένης εφαρμογής είναι η ανάλυση της χρήσης των κινητών εφαρμογών στη σχολική τάξη και η μελέτη των δυνατών μαθησιακών χρήσεων των κινητών εφαρμογών στο δημοτικό σχολείο.

Πιο συγκεκριμένα τα ερευνητικά ερωτήματα που μπορούν να τεθούν είναι:

- Πώς αλληλεπιδρούν οι μαθητές μέσα στη σχολική τάξη από τη χρήση των κινητών συσκευών και πώς οργανώνεται η τάξη για την ενσωμάτωση αυτών των εφαρμογών.
- Πώς ενσωματώνονται οι δυνατότητες των κινητών εφαρμογών σε επιμέρους γνωστικά αντικείμενα του Δημοτικού σχολείου.
- Αν είναι αποδεκτή η χρήση των κινητών συσκευών από τους μαθητές και από τους εκπαιδευτικούς και για ποιους λόγους.

Στόχος μας είναι να αναδειχθούν οι αλληλεπιδράσεις ανάμεσα σε χρήστες (μαθητές) και εργαλεία (κινητές συσκευές) όταν ασχολούνται για την υλοποίηση των μαθησιακών στόχων. Επιπλέον, θα αναφερθούμε στις μαθησιακές χρήσεις των κινητών συσκευών με στόχο τη διατύπωση διδακτικών προτάσεων για την ένταξη σε μια διαφορετικά γνωστικά αντικείμενα.
Όπως έχουμε περιγράψει και στο κεφάλαιο 2, η εφαρμογή μας διαθέτει μέσα πολλές λειτουργίες, όπως η αξιολόγηση των μαθητών και καθοδήγηση των μαθητών ώστε να έχουν καλύτερα μαθησιακά αποτελέσματα. Επίσης, διαθέτει και την δυνατότητα ανταλλαγής απόψεων και γνώσεων μεταξύ των μαθητών.

4.1 Είσοδος στην εφαρμογή

Μετά την εκκίνηση της εφαρμογής κάθε φορά εμφανίζεται η αρχική οθόνη στην οποία καλείται να επιλέξει ο χρήστης της εφαρμογής μας μια από τις διαθέσιμες επιλογές ανεξαρτήτως αν έχει κάνει εγγραφή ή όχι. Αν ο χρήστης παραμένει “GUEST” τότε έχει πρόσβαση μόνο στην πρώτη επιλογή, δηλαδή μπορεί να δει μόνο τα περιεχόμενα του διαθέσιμου εκπαιδευτικού υλικό και δεν μπορεί να εκμεταλλευτεί πλήρως όλες τις λειτουργίες της εφαρμογής. Αν ο επισκέπτης χρήστης μας θέλει να εκμεταλλευτεί όλες τις λειτουργίες της εφαρμογής μας πρέπει να μεταβεί σε ενέργεια δημιουργίας προφίλ χρήστη ακολουθώντας τα κατάλληλα βήματα.

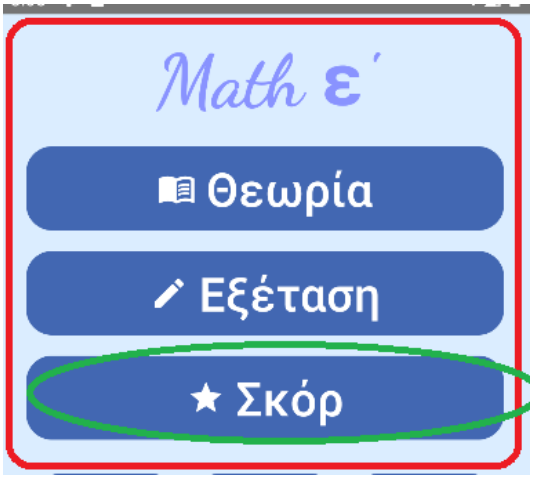
Μια γενική επεξήγηση των βασικών σημείων:

- **Data:**Τα μοντέλα που αλληλεπιδρούν με τους πίνακες της βάσης
- **DI:** Περιέχει τα module μέσα για το hilt injection
- **Domain:** Περιέχει τα interface για τα repository
- **Framework:** Στο παρόν γίνεται η υλοποίηση του datasource interface από το data file.
- **UI:** Περιέχει τα viewModel , compose και fragment
- **Usecase:** Περιέχει τις χρήσεις των repositories.
- **Util:** Περιέχει λειτουργίες που δεν ανήκουν στα παραπάνω.

Εικόνα 4.1: Main Menu

Πρόκειται για την κεντρική οθόνη (Εικόνα 26) της εφαρμογής, την οποία θα συναντήσει οποιοσδήποτε χρήστης εισέλθει στην εφαρμογή, είτε είναι εγγεγραμμένος είτε όχι.

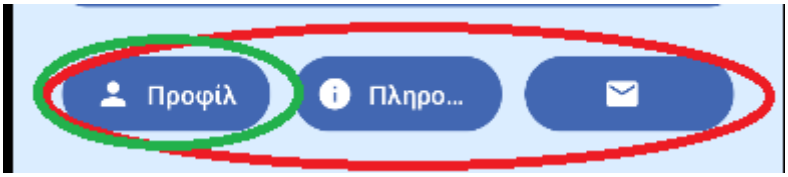
Αρχικά υπάρχουν τα τρία κουμπιά όπου κατευθύνουν τον μαθητή στο κατάλληλο πεδίο αναλόγως με την επιλογή του :



Εικόνα 4.2.: Main Menu Score

- **Η Θεωρία:**
  Ανοίγοντας μια καινούρια οθόνη μας κατευθύνει στις κατάλληλες ενότητες όπου υπάρχει το κατάλληλο εκπαιδευτικό υλικό για τον μαθητή.
- **Η εξέταση:**
  Μεταβαίνοντας στην οθόνη που εκτελεί τη διαδικασία εξέτασης, όπου ο χρήστης αφού έχει ολοκληρώσει την διαδικασία εγγραφής στην εφαρμογή**.**
- **Το σκορ:**
  Είναι η οθόνη όπου εμφανίζονται τα αποτελέσματα της εξέτασης.

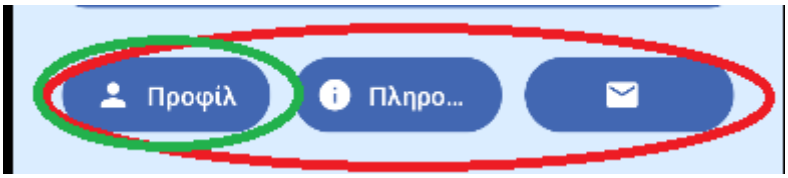
Επιπλέον υπάρχουν και τα τρία εικονίδια πιο κάτω όπου εξηγούνται πιο αναλυτικά η κάθε λειτουργία που εκτελείται σε κάθε οθόνη:



Εικόνα 4.3: Main Menu Profile

- **Το Προφίλ:** Πατώντας στο εικονίδιο αυτό ο επισκέπτης χρήστης μας μπορεί να κάνει εγγραφή στην εφαρμογή μας.
- **Οι Πληροφορίες:** Πατώντας στο εικονίδιο πληροφορίες μεταβαίνουμε σε μια καινούρια οθόνη όπου μας δίνει οδηγίες για την χρήση της εφαρμογής.
- **Η Συνομιλία:** Ο εγγεγραμμένος μαθητής μπορεί να ανταλλάξει μηνύματα με τους υπόλοιπους συμμαθητές του.

4.1.1 Εγγραφή στην εφαρμογή (Το παράθυρο που εμφανίζεται μόλις πατήσει ο χρήστης το εικονίδιο προφίλ)



Εικόνα 4.4: Main Menu Profile Second

Εικόνα 4.5: Profile Screen

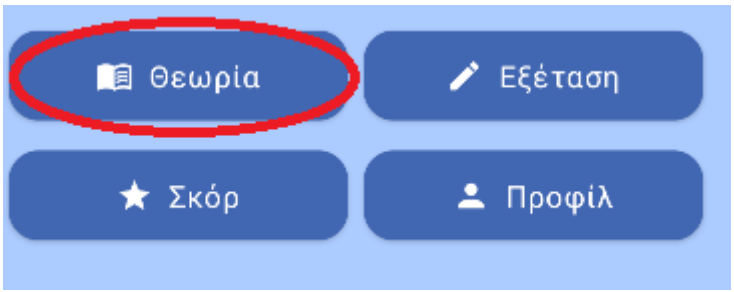
Θεωρείται μια από τις βασικές ενέργειες που πρέπει να κάνει ο χρήστης προκειμένου να δημιουργήσει προσωπικό του λογαριασμό στην εφαρμογή. Στην συγκεκριμένη οθόνη, αυτό που χρειάζεται να κάνει ο μαθητής είναι να συμπληρώσει τα απαραίτητα πεδία (Σχήμα 27).

Τα στοιχεία συμπλήρωσης αφορούν τα στοιχεία που πρόκειται θα χρησιμοποιηθούν για ταυτοποίηση των χρηστών της εφαρμογής κατά την είσοδο του στην εφαρμογή. Πιο συγκεκριμένα, τα στοιχεία όνομα και επίθετο και το σχολείο χρησιμοποιούνται για ταυτοποίηση του κάθε μαθητή. Ο μαθητής με το κουμπί " **Καθαρισμός πεδίων**" διαγράφει όλα τα δεδομένα που υπάρχουν στην φόρμα ενώ το κουμπί 'αποθήκευση' αποθηκεύει το προφίλ του μαθητή.

4.1.2 Πληροφορίες για κάθε πεδίο της εφαρμογή (Το παράθυρο που εμφανίζεται μόλις πατήσει ο χρήστης το εικονίδιο πληροφορίες)

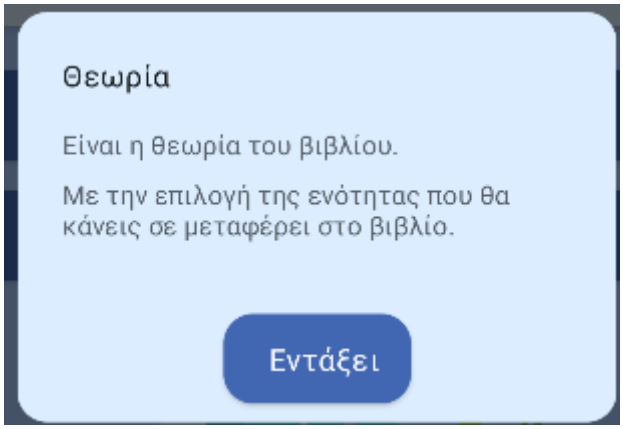
Εικόνα 4.6: Main Menu Information

Εικόνα 4.7: Information Screen

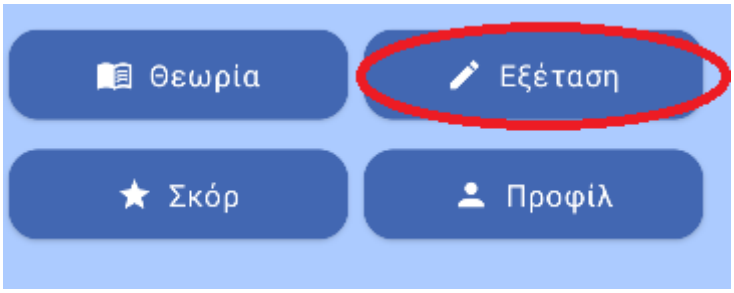


Εικόνα 4.8: Information Theory button

Πατώντας στο κουμπί (button) με τίτλο «Θεωρία» μας εμφανίζει το αναδυόμενο παράθυρο και ενημερώνει τον μαθητή ως αναφορά για το περιεχόμενο της οθόνης και για τους κανόνες που ισχύουν στο συγκεκριμένο πεδίο .

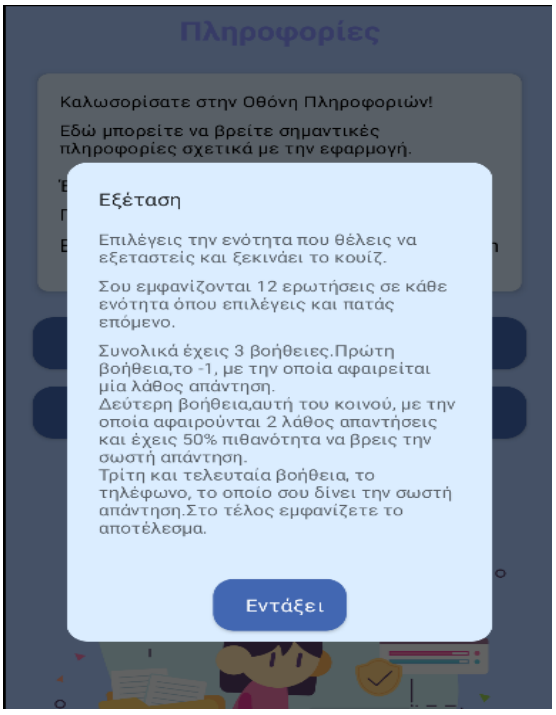


Εικόνα 4.9: Information Theory Screen



Εικόνα 4.10: Information Exam Button

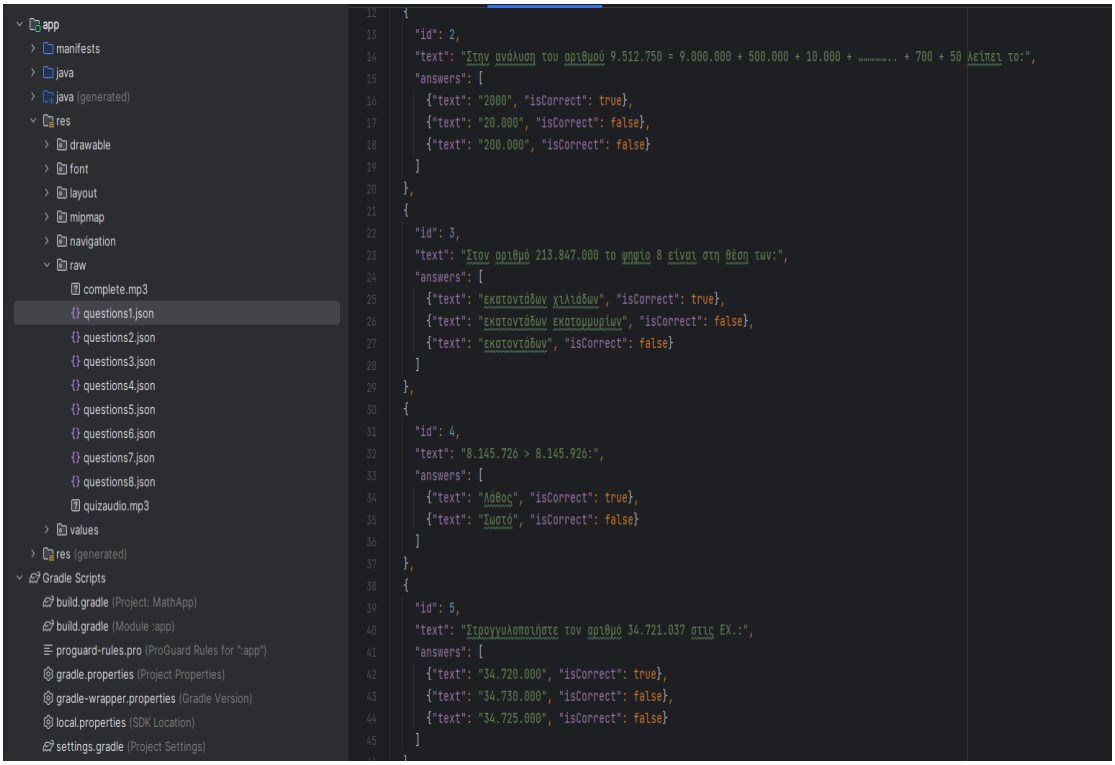
Πατώντας στο κουμπί (button) με τίτλο «εξέταση» μας εμφανίζει το αναδυόμενο παράθυρο και ενημερώνει τον μαθητή ως αναφορά για το περιεχόμενο της οθόνης και για τους κανόνες που ισχύουν το συγκεκριμένο πεδίο .



Εικόνα 4.11: Information Exam Screen

Το αναδυόμενο παράθυρο (Pop-Up Menu) μας δίνει τις πληροφορίες:

Το πλήθος των ερωτήσεων και τα απαραίτητα βήματα που πρέπει να κάνει ο εξεταζόμενος. Επίσης, εξηγεί τους κανόνες που ισχύουν κατά τη διάρκεια της εξέτασης και τις ευκολίες που δικαιούται ο μαθητής κατά τη διάρκεια της εξέτασης.



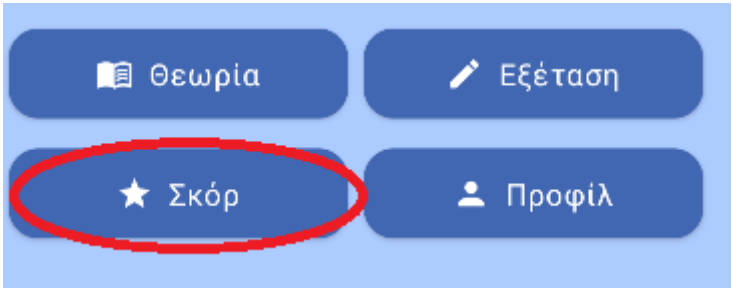
Εικόνα 4.12: Οι ερωτήσεις ανά ενότητα json

Οι ερωτήσεις αποθηκεύονται σε μορφή JSON αρχείου στο φάκελο raw. Κάθε ερώτηση αντιστοιχεί σε ένα αντικείμενο τύπου JSON(JSON object).

Και αποτελείται από τα παρακάτω πεδία:

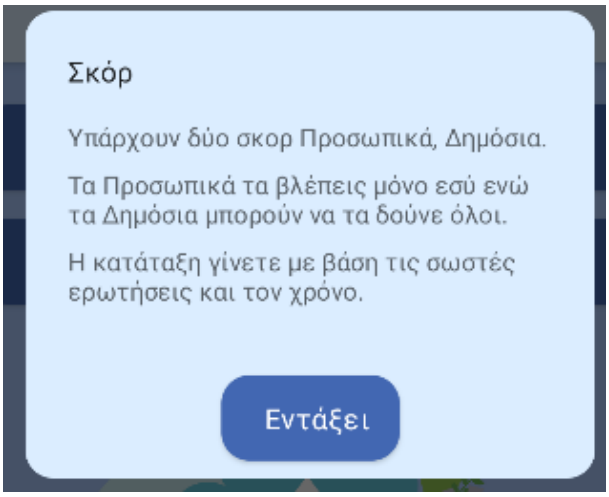
- Id: Είναι το αναγνωριστικό του κάθε αντικειμένου τύπου Json.
- Text: Περιέχει την εκφώνηση της ερώτησης.

Answer: Αποτελείται από ένα αντικείμενο που περιέχει τα αντικείμενα. Κάθε αντικείμενο περιέχει τις απαντήσεις και τη σωστή απάντηση.



Εικόνα 4.13: Information score button

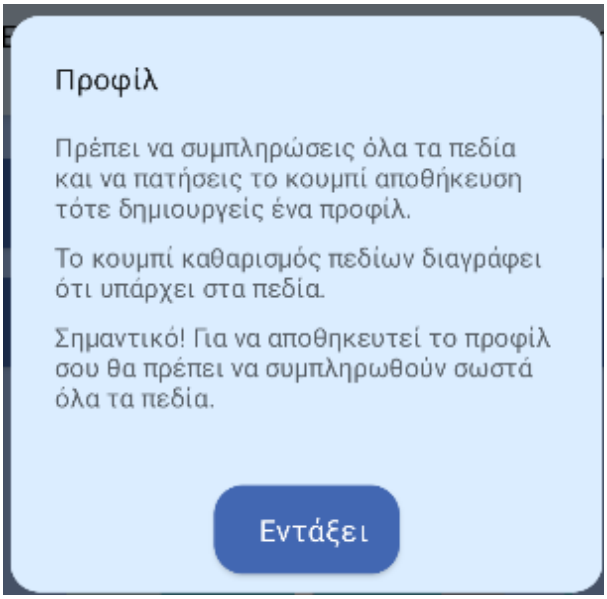
Πατώντας κουμπί (button) με τίτλο «σκορ» μας εμφανίζει το αναδυόμενο παράθυρο και ενημερώνει τον μαθητή ως αναφορά για το περιεχόμενο της οθόνης και για τους κανόνες που ισχύουν στο συγκεκριμένο πεδίο .



Εικόνα 4.14: Information Score Screen

Εικόνα 4.15: Information Profile Button

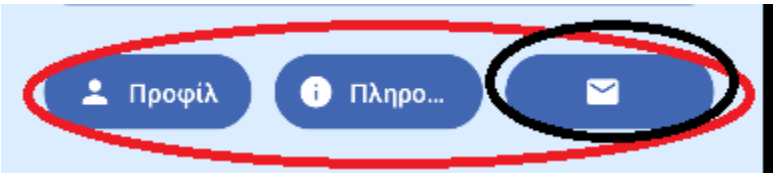
Πατώντας κουμπί (button) με τίτλο «Προφίλ» μας εμφανίζει το αναδυόμενο παράθυρο και ενημερώνει τον μαθητή ως αναφορά για το περιεχόμενο της οθόνης και για τους κανόνες που ισχύουν στο συγκεκριμένο πεδίο .



Εικόνα 4.16: Information Profile Screen

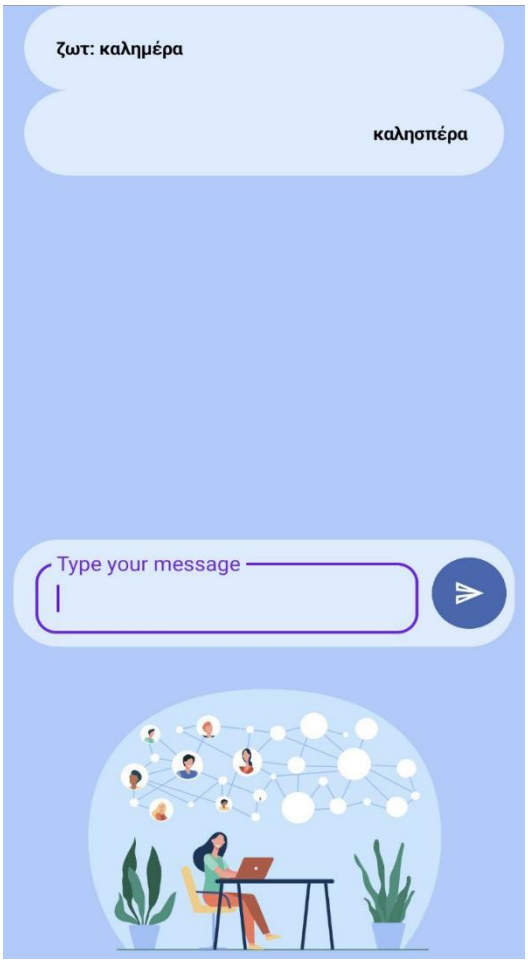
Εξηγεί τα βήματα και τους κανόνες που ισχύουν για την ορθή καταχώρηση των προσωπικών δεδομένων.

4.1.3 Πεδίο συνομιλιών (Το παράθυρο που εμφανίζεται μόλις πατήσει ο χρήστης το εικονίδιο MAIL)



Εικόνα 4.17: Main Menu Chat

Στο σημείο αυτό η εφαρμογή μας παρέχει τη δυνατότητα στους μαθητές και στους εκπαιδευτικούς να μπορούν να αλλάξουν τις απόψεις τους και με τους υπόλοιπους συμμαθητές τους, να έχουν 24-ωρη καθοδήγηση από τους εκπαιδευτικούς αλλά και οι εκπαιδευτικοί έχουν την δυνατότητα επικοινωνίας με τους μαθητές τους εκτός σχολείου.

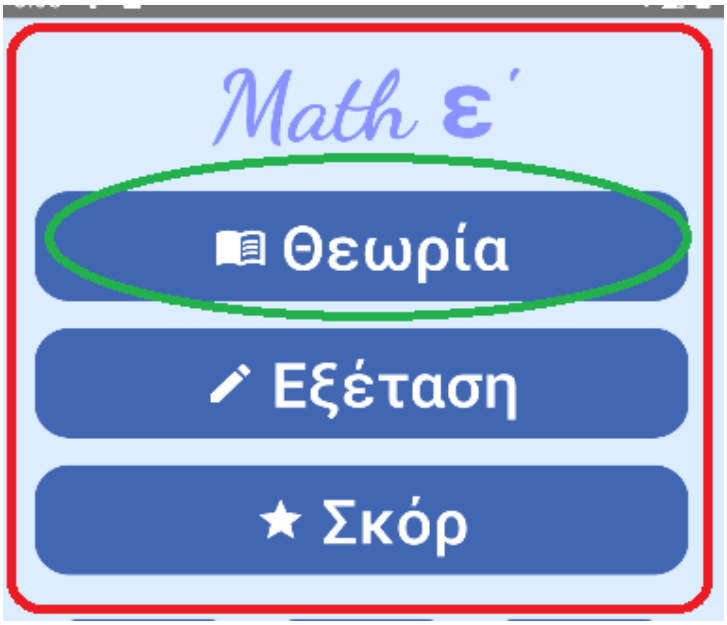


Εικόνα 4.18: Chat Screen

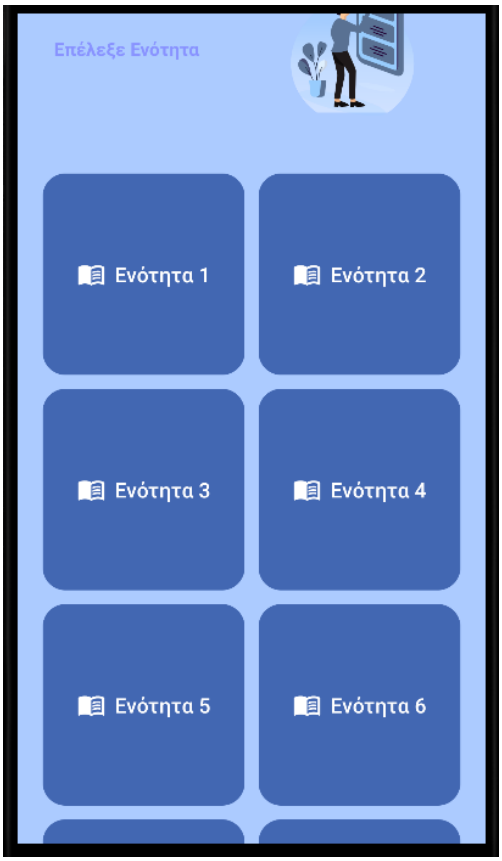
```
@HiltViewModel
class ChatViewModel @Inject constructor(
    private val getUsersUseCase: GetUsersUseCase
) : ViewModel() {
    private val database = FirebaseDatabase.getInstance().reference.child( pathString: "messages")
    private val _messageList = MutableLiveData<List<ChatMessage>>()
    val messageList: LiveData<List<ChatMessage>> get() = _messageList

    private val _getUserData = MutableLiveData<List<UserEntity>>()
    val getUserData: LiveData<List<UserEntity>> = _getUserData

    init {
        getUser()
        // Set up Firebase Realtime Database Listener for new messages
        database.addChildEventListener(object : ChildEventListener {
            override fun onChildAdded(snapshot: DataSnapshot, previousChildName: String?) {
                val chatMessage = snapshot.getValue(ChatMessage::class.java)
                chatMessage?.let { it: ChatMessage
                    val updatedList = _messageList.value?.toMutableList() ?: mutableListOf()
                    updatedList.add(it)
                    _messageList.value = updatedList
                }
            }

            override fun onChildChanged(snapshot: DataSnapshot, previousChildName: String?) {}
            override fun onChildRemoved(snapshot: DataSnapshot) {}
            override fun onChildMoved(snapshot: DataSnapshot, previousChildName: String?) {}
            override fun onCancelled(error: DatabaseError) {}
        })
    }

    fun sendMessage(messageText: String) {

        //val user = FirebaseAuth.getInstance().currentUser
        //val userId = user?.uid ?: ""
        val userName =  _getUserData.value?.lastOrNull()?.surname ?: "Anonymous"//user?.displayName ?: "Anonymous"

        val chatMessage = ChatMessage( userId: "userId", userName, messageText)
        database.push().setValue(chatMessage)
    }

> chat > ChatViewModel > sendMessage
```

Εικόνα 4.19: Chat ViewModel

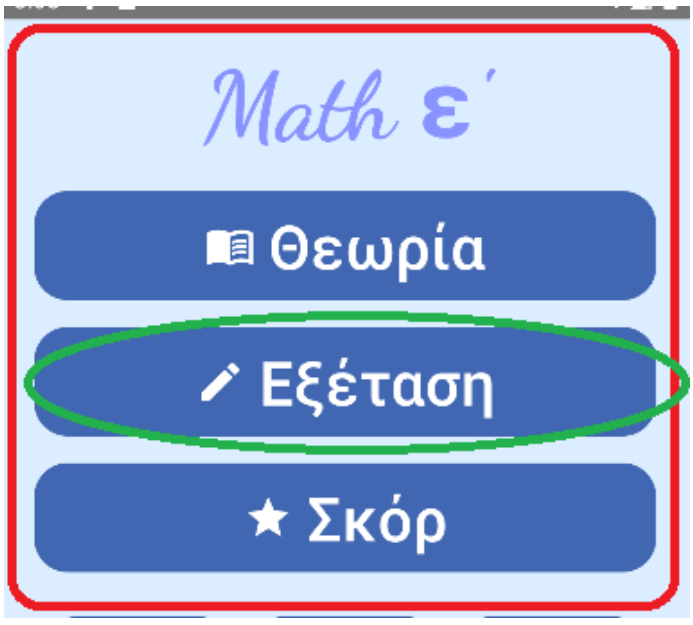
4.2 Οι βασικές λειτουργίες της Εφαρμογής (Main Screen)



Εικόνα 4.20: Main Menu Theory Button

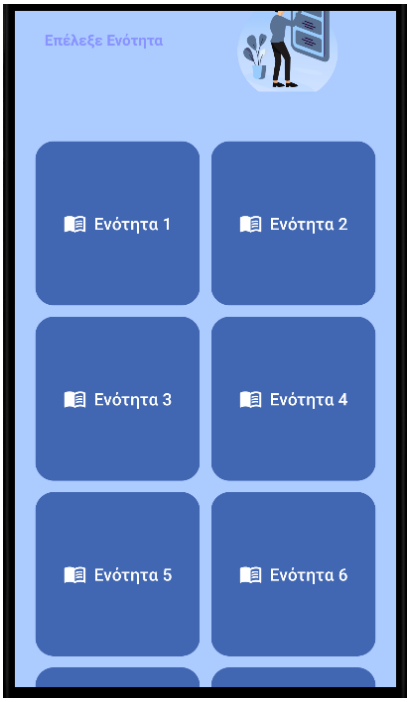


Εικόνα 4.21: Theory Screen



Εικόνα 4.22: Main Menu Exam Button

Πατώντας κουμπί (button) με τίτλο «Εξέταση» η εφαρμογή μας μεταβαίνει σε μία καινούρια οθόνη η οποία περιέχει τις ενότητες ανά κεφάλαιο και αναλόγως με την επιλογή της κατάλληλης ενότητας εμφανίζονται τα ερωτήματα.

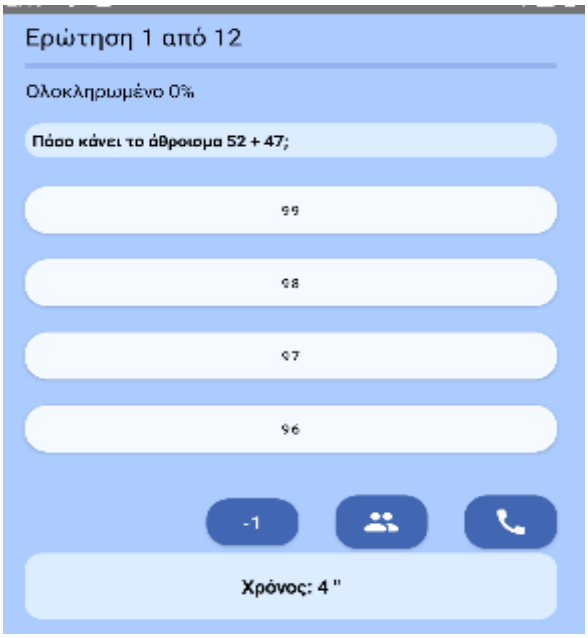


Εικόνα 4.23: Exam Unit Screen

Επιλέγει ο χρήστης τη κατάλληλη ενότητα και του εμφανίζεται ένα ενημερωτικό παράθυρο όπου εξηγεί τους κανόνες της εξέτασης.

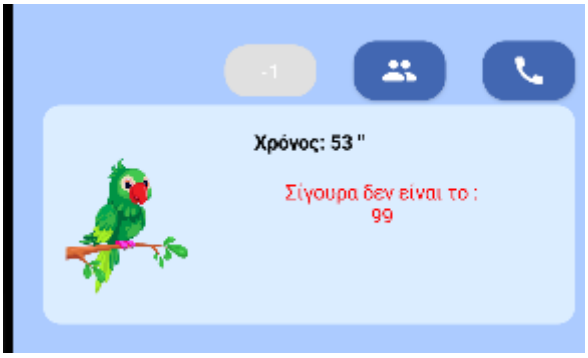
Εικόνα 4.24: Κανόνες της εξέτασης

Ο Χρήστης αφού έχει διαβάσει το ενημερωτικό κείμενο πατώντας στο κουμπί της εκκίνησης ξεκινάει η διαδικασία της εξέτασης και εμφανίζεται η οθόνη που έχουμε στην εικόνα 43.



Εικόνα 4.25: Η οθόνη απαντήσεων

Υπάρχουν 4 επιλογές. Η μία από αυτές αντιστοιχεί σε σωστή απάντηση. Υπάρχει ένα πεδίο στο κάτω μέρος της οθόνης όπου μετράει το χρόνο από την ώρα της εξέτασης.



Εικόνα 4.26: Η Βοήθεια αφαίρεση μιας λαν/μένης απάντησης

Επιλέγοντας τη βοήθεια αυτή αφαιρείται μια λανθασμένη απάντηση.

Εικόνα 4.27: Βοήθεια από το κοινό

Επιλέγοντας τη βοήθεια αυτή το σύστημα απευθύνεται στο κοινό και μας εμφανίζει τις απαντήσεις που έχει λάβει από το κοινό.

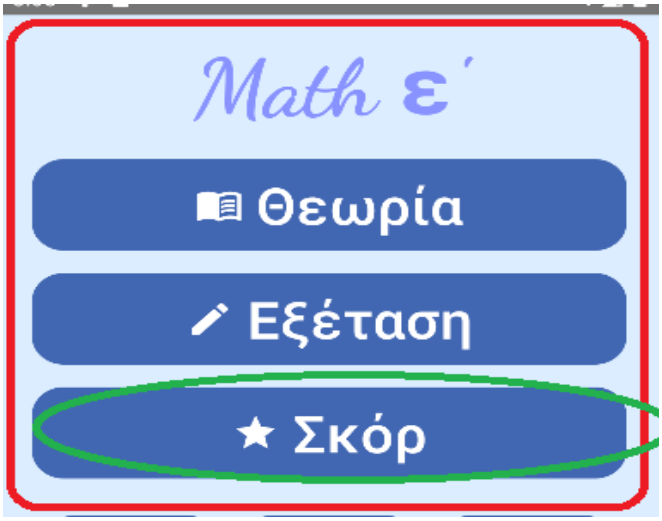
Εικόνα 4.28: Η βοήθεια Τηλέφωνο

Επιλέγοντας τη βοήθεια αυτή το σύστημα απευθύνεται υποτίθεται σ΄ ένα γνωστό μέσω τηλεφώνου και μας εμφανίζει την πιο πιθανή σωστή απάντηση που δίνεται στην συγκεκριμένη ερώτηση.

Εικόνα 4.29: Τελική οθόνη τεστ

Εφόσον έχει ολοκληρωθεί η διαδικασία της εξέτασης, στο τέλος μας εμφανίζεται η οθόνη που εμφανίζεται στην εικόνα 42. Μας εμφανίζεται η απόδοση του μαθητή, πόσες σωστές και πόσες λανθασμένες απαντήσεις υπάρχουν και σε πόσο χρόνο το ολοκλήρωσε.
Στη μέση της οθόνης εμφανίζονται δύο κουμπιά:

- ***Προσωπικά:***
  Κρατάει την πληροφορία μόνο στην τοπική βάση και δεν έχει πρόσβαση από τους υπόλοιπους μαθητές.
- ***Δημόσια:***
  Στέλνει το αποτέλεσμα σε νέφος και στην συγκεκριμένη πληροφορία μπορούν να έχουν όλοι οι χρήστες της εφαρμογής έχουν πρόσβαση από τους υπόλοιπους μαθητές.

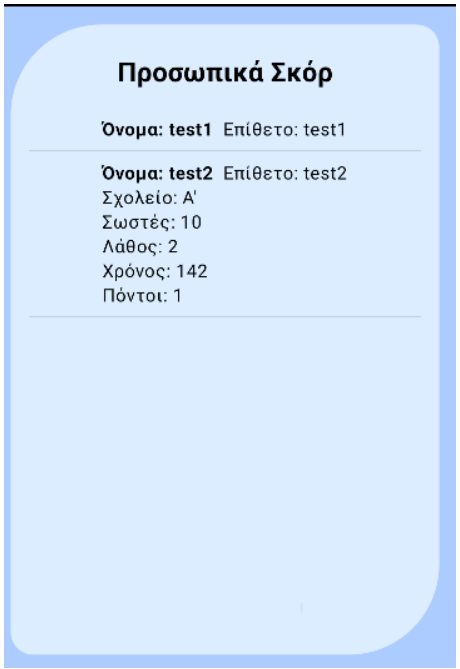


Εικόνα 4.30: Main Menu Score Buttons

Εικόνα 4.31: Score Screen

Ο χρήστης πατώντας το κουμπί «Σκορ» μεταφέρεται στην παραπάνω εικόνα. Στο σημείο αυτό έχει τρεις επιλογές:

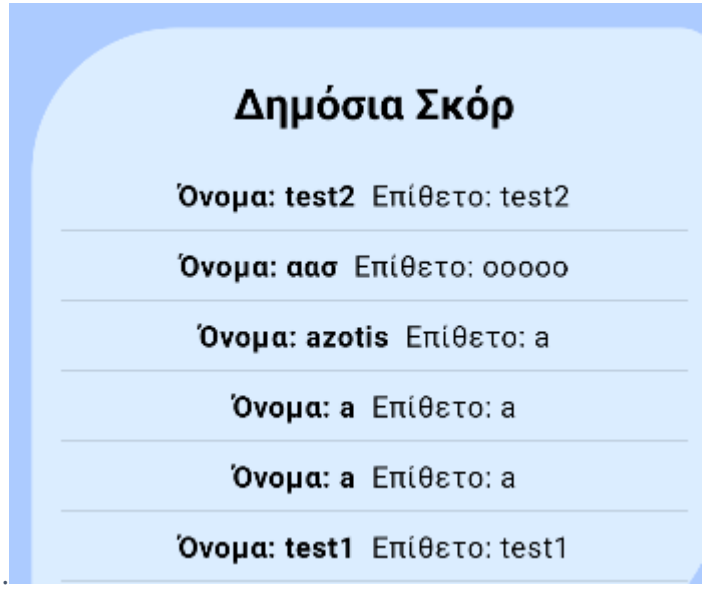
- **Προσωπικά :** Περιέχει τις δοκιμασίες που έχει κάνει ο χρήστης, τα κρατάει στην τοπική βάση δηλαδή πρόσβαση έχει μόνο ο ίδιος.



Εικόνα 4.32: Personal Score

- **Δημόσια:** Περιέχει τις δοκιμασίες που έχει κάνει ο χρήστης, τα κρατάει σε μια απομακρυσμένη

βάση, δηλαδή πρόσβαση έχει ο ίδιος και οι υπόλοιποι χρήστες που είναι εγγεγραμμένοι στην εφαρμογή.



Εικόνα 4.33: Δημόσια Σκόρ Screen

- **Τρόπαια:** Εμφανίζονται τα τρόπαια που έχει κατακτήσει ο χρήστης. Για να κατακτήσει ένα τρόπαιο πρέπει σε κάθε ενότητα να έχει μαζέψει 10 πόντους.

Εικόνα 4.34: Achievements Screen

Ο χρήστης μόλις έχει μαζέψει 10 πόντους ξεκλειδώνει την ενότητα και εμφανίζεται η παρακάτω εικόνα.

Εικόνα 4.35: Τρόπαιο

ΚΕΦΑΛΑΙΟ 5: ΣΥΜΠΕΡΑΣΜΑΤΑ ΚΑΙ ΕΠΕΚΤΑΣΕΙΣ

Στο κεφάλαιο αυτό, θα εξετάσουμε τα συμπεράσματα, τις πιθανές μελλοντικές αλλαγές που μπορούν να γίνουν, καθώς και τις καινοτομίες σε σχέση με τις υπάρχουσες εφαρμογές.

5.1 Συμπεράσματα

Στα πλαίσια της παρούσας πτυχιακής εργασίας αναπτύχθηκε μια Mobile εφαρμογή για τα παιδιά του δημοτικού με σκοπό τη βελτίωση της εκπαιδευτικής διαδικασίας κάνοντας πιο ελκυστικό από τη μεριά των μαθητών τη διαδικασία της μάθησης. Επίσης. από τη μεριά των καθηγητών η εφαρμογή διευκολύνει και επιταχύνει πάρα πολύ την διαδικασία της επανατροφοδότησης. Με αυτό το τρόπο οι καθηγητές μπορούν να έχουν μια γενική εκτίμηση για τον εκπαιδευτικό τους έργο ,αν έχει πετύχει τους στόχους του ή όχι. Αργότερα. μπορεί να προτεθεί η ιδιότητα να μπορεί ο εκπαιδευτικός να καθορίσει το περιεχόμενο της ενότητας και το υλικό που πρέπει να μελετήσει ο μαθητής.

5.2 Επεκτάσεις

Η εφαρμογή ‘Math ε΄’ αναπτύχθηκε στα πλαίσια μιας πτυχιακής εργασίας, πράγμα που σημαίνει ότι δεν ήταν δυνατόν να γίνει με το βέλτιστο τρόπο. Οπότε καλό θα ήταν να αναφερθούμε σε κάποιες προτάσεις σχετικά με την περαιτέρω βελτίωση στις λειτουργίες της ,που ίσως με τη βοήθεια κάποιου ειδικού της εκπαίδευσης να έχει καλύτερα αποτελέσματα.

Η εφαρμογή θα μπορούσε να βελτιωθεί με την προσθήκη καινούριων λειτουργιών. Αρχικά θα μπορούσε να συνδεθεί με τα μέσα κοινωνικής δικτύωσης, για την προσέλκυση περισσότερων ατόμων αλλά και για την εκμετάλλευση άλλων δυνατοτήτων που παρέχουν τα μέσα αυτά. Επίσης, η προσθήκη αυξημένης πραγματικότητας(AR) σε κάποιες λειτουργίες της εφαρμογής σίγουρα θα τραβήξουν το ενδιαφέρον των μαθητών.

Το σημαντικότερο από όλα είναι η προσθήκη των λειτουργιών για την ειδική αγωγή (τα παιδιά με αναπηρία). Θα μπορούσε να υπάρχει ένα ξεχωριστό πεδίο κατάλληλα σχεδιασμένο με στήριξη και εξωτερικών συσκευών για να βοηθήσουμε και να κάνουμε την διαδικασία της εκμάθησης πιο ευχάριστη.

5.3 Καινοτομίες

Σε σχέση με άλλες εκπαιδευτικές εφαρμογές, η παρούσα εφαρμογή ξεχωρίζει λόγω των παρακάτω σημαντικών παραγόντων:

- Ανταγωνιστικό Στοιχείο: Ο ανταγωνισμός αποτελεί έναν σημαντικό παράγοντα για την ενίσχυση της μάθησης. Οι διαγωνισμοί και το παιχνίδι ενθαρρύνουν τους μαθητές.
- Διασκεδαστική Μάθηση: Η εφαρμογή προσφέρει μια ενδιαφέρουσα και διασκεδαστική προσέγγιση της μάθησης, βοηθώντας τους μαθητές να εναρμονίσουν την εκμάθηση με την διασκέδαση.
- Αλληλεπίδραση μέσω του συνομιλίας: Οι μαθητές μπορούν να ανταλλάσσουν ιδέες, να λύνουν απορίες και να συνεργάζονται για την καλύτερη κατανόηση των μαθηματικών.
- Εξειδίκευση στα Μαθηματικά της Ε' Δημοτικού: Η εφαρμογή επικεντρώνεται αποκλειστικά στα μαθηματικά της ε΄ τάξης του δημοτικού, προσφέροντας ένα περιβάλλον ειδικά σχεδιασμένο για αυτή την ηλικιακή ομάδα. Επιπλέον, παρέχει παραπομπές στη σχολική ύλη, προσφέροντας στους μαθητές πρόσβαση στο σχολικό βιβλίο .

Συνολικά, η εφαρμογή αυτή δημιουργεί ένα ενθαρρυντικό και ανανεωτικό περιβάλλον για την μάθηση των μαθηματικών, προσφέροντας ταυτόχρονα εξατομικευμένη υποστήριξη και πρόκληση για κάθε μαθητή.

ΒΙΒΛΙΟΓΡΑΦΙΑ:

- [1] GOOGLE:
  - www.google.com
- [2] ANDROID:
  - Developer.android.com
- [3] https://developer.android.com/jetpack/compose
- [4] GESTURES :
- https://techblog.gr/software/android-10-ola-osa-prepei-na-gnorizoyme-gia-ta-gestures/
- [5] https://www.android.com/android-11/
- [6].KOTLIN
- [7] https://medium.com/@ajaysaini.official/building-database-with-room-persistence-library-ecf7d0b8f3e9
- [8] https://riggaroo.co.za/android-architecture-components-looking-room-livedata-part-1/
- [9] https://developer.android.com/training/data-storage/room

- [10] https://codelabs.developers.google.com/codelabs/jetpack-compose-basics
- [11] https://developer.android.com/jetpack/androidx/releases/ui
- [12] https://joebirch.co/android/exploring-jetpack-compose-row-column/

- [13] *https://developer.android.com/training/dependency-injection/hilt-android*
- [14] *https://medium.com/androiddevelopers/dependency-injection-on-android-with-hilt-67b6031e62d*

- [15] Kotlin image: https://www.geeksforgeeks.org/difference-between-java-and-kotlin-in-android-with-examples/

- [16] https://developer.android.com/jetpack/compose/mental-model