ΠΑΝΕΠΙΣΤΗΜΙΟ ΠΕΙΡΑΙΩΣ – ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ

Πρόγραμμα Μεταπτυχιακών Σπουδών

«Πληροφορική»

Μεταπτυχιακή Διατριβή

Τίτλος Διατριβής	**Εφαρμογή για Android κινητές συσκευές με ερωτήσεις εκμάθησης ιστορίας γραμμένη σε kotlin** **Android application for mobile devices with questions for learning purposes about history written in kotlin**
Ονοματεπώνυμο Φοιτητή	**Βασιλόπουλος Αντώνιος**
Πατρώνυμο	**Δημήτριος**
Αριθμός Μητρώου	**ΜΠΠΛ 19005**
Επιβλέπων	**Ευθύμιος Αλέπης, Καθηγητής**

Ημερομηνία Παράδοσης **Ιανουάριος 2023**

Τριμελής Εξεταστική Επιτροπή

Ευθύμιος Αλέπης Καθηγητής	Μαρία Βίρβου Καθηγήτρια	Ευάγγελος Σακκόπουλος Αναπληρωτής Καθηγητής

Table of Contents

Ευχαριστίες

Θα ήθελα να ευχαριστήσω τον κ. Αλέπη για την ανάθεση της μεταπτυχιακής μου διατριβής, καθώς και για την επίβλεψη και την επιστημονικά άρτια καθοδήγηση του κατά τη διάρκεια του σχεδιασμού και της ανάπτυξης αυτής της εργασίας, αλλά και για τις καίριες συμβουλές που μου έδωσε για τη βελτίωση της εφαρμογής που αναπτύχθηκε.

Τέλος, ευχαριστίες οφείλω στην οικογένειά μου και στους φίλους μου, για την ηθική υποστήριξη και βοήθεια όλα τα χρόνια των σπουδών, όπως επίσης για την κατανόηση και την ενθάρρυνση κατά τη διάρκεια εκπόνησης της μεταπτυχιακής διατριβής.

Περίληψη

Στόχος της παρούσας μεταπτυχιακής διατριβής αποτελεί η ανάπτυξη εφαρμογής βασιζόμενη σε λειτουργικό σύστημα android και η οποία θα αφορά ερωτήσεις γνώσης που θα απευθύνονται όχι μόνο σε μαθητές, αλλά και σε ενήλικες. Η θεματολογία των ερωτήσεων αναφέρεται στην χρονολογική περίοδο της Ελληνικής Επανάστασης του 1821 και ο χρήστης καλείται να επιλέξει την κατάλληλη απάντηση σε ερωτήσεις τύπου πολλαπλών επιλογών (“multiple choice”).

Αρχικά, παρατίθενται εισαγωγικές πληροφορίες που αφορούν τα έξυπνα τηλέφωνα και την εξέλιξή τους με την πάροδο των ετών. Έπειτα αναλύεται το λειτουργικό σύστημα android και η αρχιτεκτονική του συστήματος. Στη συνέχεια περιγράφονται τα στάδια υλοποίησης της εφαρμογής σε επίπεδο αισθητικής παρουσίασης και τεχνικής υλοποίησης.

Συνοπτικά, αναφορικά με την εφαρμογή, ο χρήστης θα εισέρχεται με ένα όνομα χρήστη, θα επιλέγει κάποιες γενικές κατηγορίες όπως ηλικία, φύλο και επίπεδο μόρφωσης, θα μπορεί να επιλέξει θεματολογία ερωτήσεων που επιθυμεί και θα μπορεί να μοιραστεί τις επιδόσεις του. Θα έχει τη δυνατότητα, επίσης, να παρακολουθήσει κάποια στατιστικά χρήσης της εφαρμογής.

Στην εφαρμογή έχουν προστεθεί λειτουργίες που βελτιώνουν τη διεπαφή χρήστη (User Interface) της εφαρμογής με αποτέλεσμα να παρέχεται βελτιωμένη εμπειρία χρήστη (User Experience) καθώς και αξιοποιούνται τεχνολογίες, όπως αυτή της cloud βάσης της Firebase για την αποθήκευση δεδομένων. Με το πέρας της διπλωματικής εργασίας, θα γίνει παρουσίαση των συμπερασμάτων καθώς και προοπτικές για το μέλλον της εφαρμογής.

Abstract

The goal of this thesis is to develop an application based on the Android operating system that deals with knowledge questions aimed not only at students but also at adults. The topic of the questions refers to the chronological period of the Greek Revolution of 1821 and the user is prompted to select the appropriate answer in multiple choice questions.

Initially, introductory information is provided about smartphones and their evolution over the years. Then, the Android operating system and its architecture are analyzed. Next, the stages of implementation of the application in terms of visual presentation and technical implementation are described.

In summary, regarding the application, the user will log in with a username, will be able to select some general categories such as age, gender, and education level, will be able to select the topic of questions the user wants and will be able to share his performance. The user will also be able to view some usage statistics of the application.

The application has added features that improve the User Interface of the application resulting in improved User Experience. Also, technologies like Firebase cloud database have been utilized for data storage. At the end of the thesis, the conclusions and prospects of the application will be presented.

1. Εισαγωγή

Το «έξυπνο τηλέφωνο» (“smartphone”) έχει μετατραπεί σε παντοδύναμο εργαλείο που όχι μόνο ακολουθεί τις εξελίξεις της εποχής σε κάθε τομέα της οικονομικής και κοινωνικής ζωής λόγω των απεριόριστων τεχνολογικών του δυνατοτήτων, στις οποίες συνεχώς προστίθενται νέες με την πάροδο των ετών. Η εμφάνιση του 5G και του Διαδικτύου των Πραγμάτων (Internet of Things), έχουν διευκολύνει τον χρήστη όχι μόνο στην τηλεφωνική επικοινωνία, αλλά και στην πρόσβαση και έλεγχο χιλιάδων αντικειμένων και συσκευών, όπως των «έξυπνων» οικιακών συσκευών, στην ενημέρωση και πληροφόρηση μέσω αισθητήρων, όπως η «έξυπνη» γεωργία, η «έξυπνη» βιομηχανία και οι «έξυπνες πόλεις», στην πραγματοποίηση των συναλλαγών (e-banking). Πλέον αφιερώνουμε μεγάλο μέρος και στην ψυχαγωγία και στην εκμάθηση νέων γνώσεων μέσω των smartphones.

Οι χρήστες μπορούν να εγκαθιστούν στις συσκευές τους εφαρμογές εκπαιδευτικού ή ψυχαγωγικού τύπου, να αποκτήσουν νέες γνώσεις μέσω ερωτήσεων σε μορφή πολλαπλών επιλογών και να ελέγξουν τις επιδόσεις τους. Κατά αυτόν τον τρόπο η εκμάθηση αποκτά «ρόλο» παιχνιδιού και γίνεται πιο εύκολη και πιο ευχάριστη. Μπορεί να αποδειχθούν ευεργετικές όχι μόνο για τους μαθητές αλλά και κάθε λογής ηλικίες, βελτιώνοντας τις ατομικές τους δεξιότητες και αναβαθμίζοντας τις γνώσεις τους. Το «έξυπνο τηλέφωνο» είναι κατά κάποιον τρόπο ένα «δώρο γνώσης» για όλους εκείνους που την αναζητούν, προσφέροντας τους πρόσβαση στην μεγαλύτερη βιβλιοθήκη του κόσμου.

1.1 Στόχοι της εργασίας

Αντικείμενο της μεταπτυχιακής διατριβής αποτελεί η ανάπτυξη εφαρμογής ερωτήσεων γνώσης σε λειτουργικό σύστημα Android χρησιμοποιώντας τους κατάλληλους πόρους για να μην επιβαρύνονται οι επιδόσεις της συσκευής. Στόχος της εφαρμογής είναι να παράσχει στον χρήστη μια ξεχωριστή και ευχάριστη εμπειρία αξιοποιώντας αρκετές νέες δυνατότητες, όπως προσαρμογή της εφαρμογής στις διαστάσεις της οθόνης της εκάστοτε συσκευής και αποθήκευση δεδομένων αξιοποιώντας τις δυνατότητες των cloud τεχνολογιών.

2. Έξυπνα τηλέφωνα (Smartphones) και λειτουργικά συστήματα

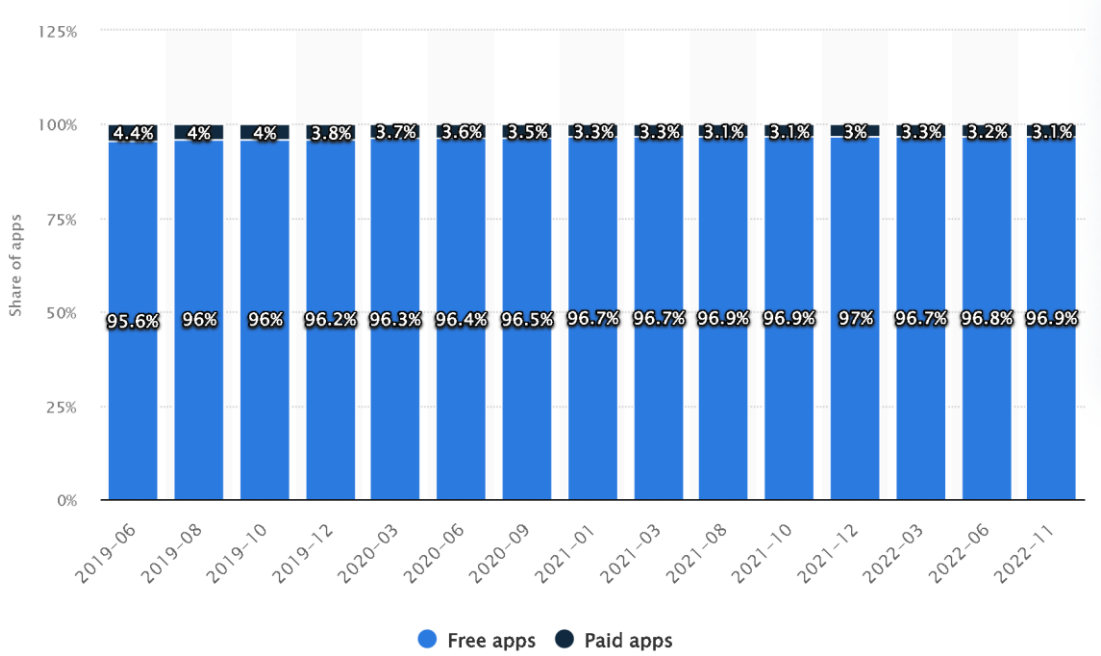
Στην παρούσα ενότητα παρουσιάζονται κάποια γενικά χαρακτηριστικά των «έξυπνων τηλεφώνων» και που τα κάνει να ξεχωρίζουν από τα πρώτα κινητά τηλέφωνα. Έπειτα

αναλύονται τα πιο διαδεδομένα λειτουργικά συστήματα, καθώς και η εξέλιξη τους με την πάροδο των ετών, για να φτάσουμε στην σημερινή εποχή.

2.1 Έξυπνο τηλέφωνο (Smartphone)

Έξυπνο τηλέφωνο είναι μια φορητή συσκευή που συνδυάζει χαρακτηριστικά που διαθέτει και ένας ηλεκτρονικός υπολογιστής με χαρακτηριστικά ενός κινητού τηλεφώνου. Σήμερα τα έξυπνα τηλέφωνα μπορούν να συνδεθούν στο διαδίκτυο με τεχνολογίες 4G/5G, διαθέτουν οθόνη αφής, ενσωματωμένο Wi-Fi, GPS, ενσωματωμένη κάμερα και πολλούς αισθητήρες, όπως δακτυλικό αποτύπωμα και αισθητήρα κίνησης. Ο χρήστης μπορεί να εγκαταστήσει εφαρμογές που επηρεάζουν και βελτιώνουν την καθημερινότητά του.

Το μεγαλύτερο ποσοστό των εφαρμογών διατίθεται δωρεάν. Στον παρακάτω πίνακα φαίνεται το ποσοστό που κατέχουν οι δωρεάν εφαρμογές στο Google Play Store και στο Apple App Store από τον Ιούνιο του έτους 2019 έως τον Νοέμβρη του έτους 2022. Το ποσοστό των δωρεάν εφαρμογών ήταν στο 95.6% στα μέσα του 2019 και αυξήθηκε στο 96.9% το 2022.



Εικόνα 1 Distribution of free and paid iOS apps in the Apple App Store as of November 2022[1]

2.2 Λειτουργικά συστήματα για έξυπνα τηλέφωνα

Παλαιότερα, με τα συμβατικά κινητά τηλέφωνα, οι δυνατότητες περιορίζονταν μόνο στην αξιοποίηση του τηλεφώνου ως μιας φορητής συσκευής όπου ο χρήστης θα μπορούσε να κάνει κλήσεις ή να στείλει μηνύματα SMS από όποιο μέρος και αν

[1] https://www.statista.com/statistics/1020996/distribution-of-free-and-paid-ios-apps/

βρίσκεται. Αν και αυτό ήταν μια επανάσταση για την εποχή, πλέον οι δυνατότητες του κινητού τηλεφώνου έχουν επεκταθεί σε τεράστιο βαθμό.

Τα έξυπνα τηλέφωνα σχεδιάστηκαν για να παρέχουν στον χρήστη δυνατότητες που μπορεί να προσφέρει και ένας σύγχρονος ηλεκτρονικός υπολογιστής. Ο χρήστης με το κινητό του τηλέφωνο μπορεί να έχει πρόσβαση σε πληθώρα εφαρμογών, οι περισσότερες από τις οποίες διατίθενται δωρεάν σε ηλεκτρονικά καταστήματα, όπως στο Google Play Store.

Με την έλευση των πρώτων έξυπνων κινητών τηλεφώνων, ο κάθε κατασκευαστής κινητών συσκευών smartphones ανέπτυσσε ένα δικό του λειτουργικό σύστημα. Αρκετά δημοφιλή λειτουργικά συστήματα ήταν το Symbian OS, το οποίο αναπτύχθηκε από την Symbian Ltd και χρησιμοποιήθηκε από πολλούς κατασκευαστές κινητών τηλεφώνων, όπως Nokia, Sony Ericsson, Sharp και Motorola καθώς και το λειτουργικό σύστημα BlackBerry OS. Το Symbian OS αποτέλεσε δημοφιλές smartphone OS μέχρι το τέλος του 2010, οπότε επικράτησαν τα λειτουργικά συστήματα iOS, Android και Windows Phone, με το τελευταίο να μην μπορεί να «αντέξει» τον ανταγωνισμό των άλλων δύο και να ανακοινώσει την διακοπή κυκλοφορίας του το 2017.

2.2.1 Χαρακτηριστικά των πρώτων «έξυπνων τηλεφώνων» και η εξέλιξη στο σήμερα.

1992: Το πρώτο κινητό τηλέφωνο σε μορφή PDA ανακοινώθηκε από την IBM. Διέθετε οθόνη αφής που μπορούσε να χρησιμοποιηθεί μόνο με ειδικά στυλό. Συγκέντρωνε κάποιες λειτουργίες που είχε και ένας ηλεκτρονικός υπολογιστής, όπως ατζέντα, σημειωματάριο ημερολόγιο και ηλεκτρονικό βιβλίο διευθύνσεων. Δεν είχε όμως την δυνατότητα να συνδεθεί στο διαδίκτυο.

1993: Η Apple παρουσίασε το PDA “Newton MessagePad”. Διέθετε οθόνη αφής που μπορούσε να χρησιμοποιηθεί με ειδικά στυλό.

1994: Η IBM παρουσίασε το πρώτο τηλέφωνο με την επωνυμία Simon Personal Communicator(SPC) που χαρακτηρίζεται ως το πρώτο smartphone. Η συσκευή συμπεριλάμβανε διάφορες εφαρμογές για βασική χρήση, όπως ημερολόγιο και σημειωματάριο. Μπορούσε να στέλνει και να λαμβάνει fax και e-mails.

Εικόνα 2 The first Smartphone[2]

1996: Η BlackBerry ανέπτυξε το δικό της λειτουργικό σύστημα, το BlackBerry OS. Μπήκε στο κομμάτι του ανταγωνισμού και παρουσίασε πρώτη διαδραστικές οθόνες. Ο χρήστης μπορούσε με αυτόν τρόπο να γνωρίζει αν έχει λάβει κάποιο email. Το 2002 η BlackBerry κατάφερε να γίνει ηγέτης στα «έξυπνα τηλέφωνα» εκείνης της εποχής μέχρι το 2007 όπου δεν άντεξε τον ανταγωνισμό του πρώτου iPhone.

Εικόνα 3 BlackBerry Smartphone, 2002[3]

2007: Η Apple Inc. κυκλοφόρησε το πρώτο iPhone. Η χρήση των smartphones αρχίζει να γίνεται ευρέως διαδεδομένη έναντι των συμβατικών τηλεφώνων. Η Apple για πρώτη φορά χρησιμοποίησε την multi-touch οθόνη αφής χωρίς να χρειάζεται η χρήση των stylus pen ή πληκτρολόγιο τύπου QUERTY. Επίσης διέθεταν GPS και ο χρήστης μπορούσε να πλοηγηθεί στο διαδίκτυο. Τα iPhones διαθέτουν δικό τους λειτουργικό σύστημα, το iOS. Το iOS είναι κλειστού κώδικα και η γλώσσα προγραμματισμού είναι η Objective-C. Για τον λόγο αυτό το λειτουργικό της σύστημα καθίσταται ασφαλές σε ιούς και κακόβουλα λογισμικά.

[2] https://simpletexting.com/where-have-we-come-since-the-first-smartphone/

[3] https://books.google.gr/books?hl=el&lr=&id=DYd-DwAAQBAJ&oi=fnd&pg=PT28&dq=history+of+smartphones&ots=n_afoA24dN&sig=_8LNg9lzfCWeUrMzSyMKHfHBTE0&redir_esc=y#v=onepage&q=history%20of%20smartphones&f=false

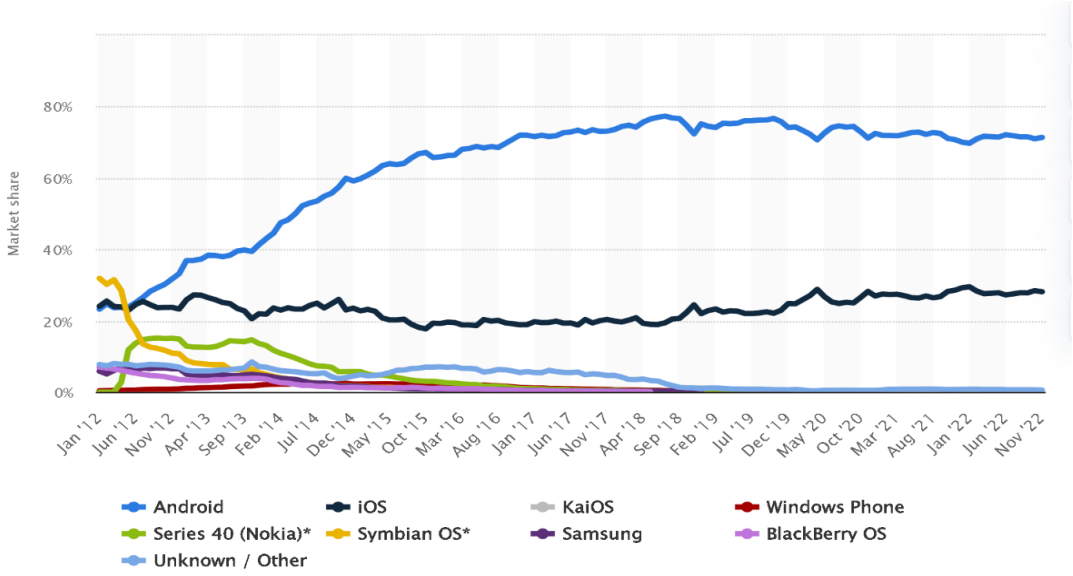
Εικόνα 4 Apple iPhone I, 2007[4]

2008: Η Google ανέπτυξε το Android λειτουργικό σύστημα. Το Android αναπτύχθηκε από μια κοινοπραξία προγραμματιστών γνωστή ως Open Handset Alliance και εμπορικά χορηγείται από την Google. Η Google διέθετε την πλατφόρμα σε κατασκευαστές κινητών συσκευών με την υπόσχεση να παρέχει ένα ευέλικτο λειτουργικό σύστημα βασισμένο στο λειτουργικό Linux Kernel.

2010: Η Microsoft με στόχο να μπει και αυτή στο παιχνίδι, παρουσίασε πρώτη φορά λειτουργικό σύστημα Windows για smartphones. Η Microsoft μάλιστα συνεργάστηκε με την Nokia, η οποία και θα αντιπροσώπευε την εταιρεία που θα λάνσαρε την πλειονότητα των συσκευών smartphones. Το γραφικό περιβάλλον των windows phones ήταν πολύ κοντά σε αυτό που είχαν οι υπολογιστές. Λόγω όμως της επικράτησης των λειτουργικών συστημάτων Android και iOS, το ενδιαφέρον της Microsoft για την περαιτέρω ανάπτυξη των Windows Phones μειώθηκε, με αποτέλεσμα περίπου στα μέσα της δεκαετίας να ανακοινώσει την οριστική διακοπή. Το βάρος δόθηκε στην ανάπτυξη λογισμικού που θα ενσωματωνόταν σε Android και iOS συσκευές.

[4] https://books.google.gr/books?hl=el&lr=&id=DYd-DwAAQBAJ&oi=fnd&pg=PT28&dq=history+of+smartphones&ots=n_afoA24dN&sig=_8LNg9lzfCWeUrMzSyMKHfHBTE0&redir_esc=y#v=onepage&q=history%20of%20smartphones&f=false

Εικόνα 5 Microsoft Windows Phone[5]



Εικόνα 6 Statistics[6]

2.3 Android και iOS

Στην παρούσα ενότητα θα γίνει παρουσίαση των πλεονεκτημάτων και μειονεκτημάτων των δύο λειτουργικών συστημάτων, του Android της Google και του iOS της Apple. Η Google και η Apple ακολουθούν δύο διαφορετικές στρατηγικές και απευθύνονται σε διαφορετικό κοινό η καθεμιά, με το Android να επικρατεί σε πωλήσεις λόγω της δωρεάν διάθεσής του σε πλήθος συσκευών διαφορετικών brands, η αξία των οποίων μπορεί να κυμαίνεται, από πολύ φθηνές έως πολύ ακριβές.

2.3.1 Πλεονεκτήματα Android

Το android βασίζεται στην Java ή την Kotlin, γλώσσες προγραμματισμού που έχουν δοκιμαστεί και αναπτύσσονται συνεχώς εδώ και αρκετά χρόνια. Ο χρήστης έχει την δυνατότητα να κατεβάσει εφαρμογές android είτε από το Google Play Store είτε από

[5] https://mspoweruser.com/a-history-of-windows-phone-the-road-to-threshold/
[6] https://www.statistica.com/en/

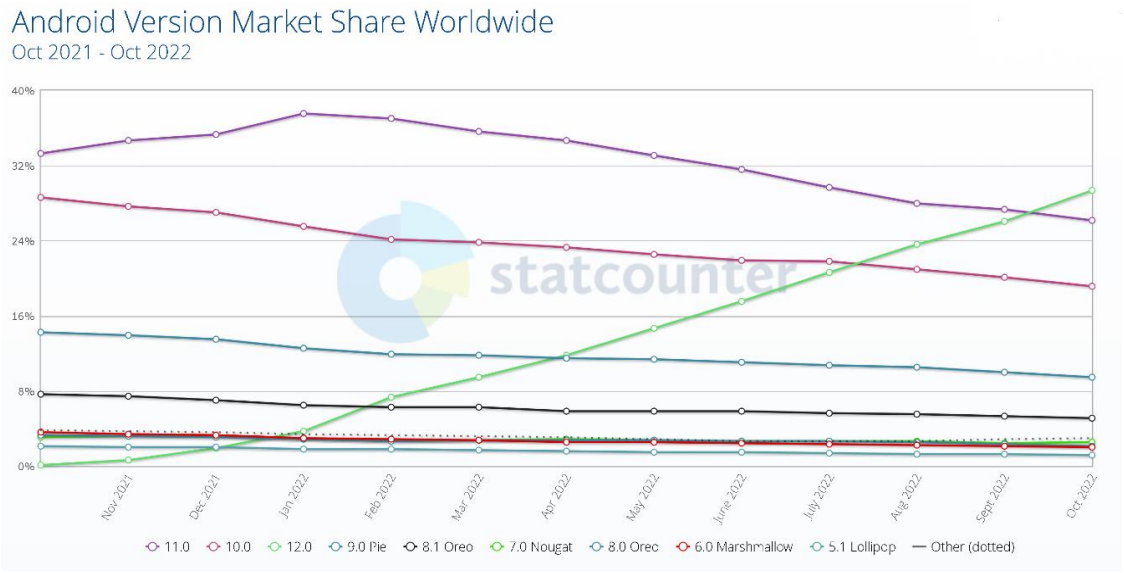
εξωτερικές βιβλιοθήκες. Σε αντίθεση με το iOS, το Android είναι «ανοικτού κώδικα» και ο προγραμματιστής μπορεί να έχει εύκολη πρόσβαση σε πληθώρα πόρων για την ανάπτυξη μιας εφαρμογής. Αυτό έχει σαν αποτέλεσμα οι τελικοί χρήστες να απολαμβάνουν τεράστια ποικιλία εφαρμογών. Επίσης τα περισσότερα smartphones στην αγορά είναι συμβατά με Android σε σύγκριση με το iOS, που αναπτύσσεται από την Apple εξ' ολοκλήρου για τις συσκευές της. Επομένως, ο χρήστης μπορεί να επιλέξει ανάμεσα σε πληθώρα συσκευών και χαρακτηριστικών hardware σε ένα εύρος τιμών αγοράς.

2.3.2 Μειονεκτήματα Android

Ο χρήστης ενός Android θα πρέπει να έχει λογαριασμό Google για να μπορεί να έχει πρόσβαση στο Google Play Store και σε άλλες υπηρεσίες της. Για κάποιους χρήστες, που ενδεχομένως να μην επιθυμούν να έχουν λογαριασμό Google, αυτό να είναι ένα κώλυμα. Επίσης, συνήθως οι κατασκευαστές Android smartphones δεν υπόσχονται αναβαθμίσεις νέων εκδόσεων για παραπάνω από δύο χρόνια και δεν γίνονται συχνές ενημερώσεις ασφαλείας. Το τελευταίο γεγονός μπορεί να καταστήσει το τηλέφωνο πιο ευάλωτο σε κακόβουλο λογισμικό (malware), πόσο μάλλον η πιθανότητα αυτή αυξάνεται όταν ο χρήστης ενσωματώνει εφαρμογές που βρίσκονται ελεύθερες στο διαδίκτυο σε μορφή .apk.

Επιπλέον, παρόλο που οι περισσότερες εφαρμογές παρέχονται δωρεάν, στοχεύουν στο κέρδος μόνο μέσα από τις διαφημίσεις που ενσωματώνουν. Αυτό έχει σαν αποτέλεσμα ο χρήστης να βομβαρδίζεται από πληθώρα διαφημίσεων, γεγονός που μειώνει την εμπειρία του χρήστη ("User Experience").

Ακόμη, ένα μεγάλο εμπόδιο για τους προγραμματιστές που αναπτύσσουν εφαρμογές Android είναι το πρόβλημα του «Κατακερματισμού» ("Fragmentation"). Αφορά το γεγονός ότι οι προγραμματιστές θα πρέπει να λάβουν υπόψιν όλα τα χαρακτηριστικά και τις λειτουργίες των συσκευών που χρησιμοποιούν οι περισσότεροι χρήστες, καθώς και το εύρος των διαφορετικών εκδόσεων Android που διαθέτουν, είτε παλαιότερες είτε νεότερες. Οι νεότερες συσκευές είτε μπορεί να διαθέτουν λειτουργίες που να μην διαθέτουν οι παλαιότερες, είτε να λειτουργούν με διαφορετικό τρόπο. Επομένως, αυτός είναι ένας επιπλέον «πονοκέφαλος», καθώς οι εφαρμογές θα πρέπει να τρέχουν αποτελεσματικά σε όλες τις συσκευές. Στο παρακάτω διάγραμμα αποδεικνύεται μεταξύ του Οκτώβρη του έτους 2021 και του Οκτώβρη του 2022 ποιες εκδόσεις Android χρησιμοποιούν ακόμη οι χρήστες.



Εικόνα 7 Mobile Operating System Market Share Worldwide[7]

2.3.3 Πλεονεκτήματα iOS

Το μεγαλύτερο πλεονέκτημα του iOS είναι ότι οι χρήστες Apple iPhones απολαμβάνουν ενημερώσεις λογισμικού αλλά και ενημερώσεις ασφαλείας ακόμη και ύστερα από πέντε με έξι χρόνια από την ημερομηνία κυκλοφορίας τους. Οι εφαρμογές είναι συνήθως καλύτερες σε σχέση με αυτές που προσφέρει το Google Play Store καθώς οι προγραμματιστές στρέφουν περισσότερο το ενδιαφέρον τους προς τα εκεί, καθώς το κέρδος για την ανάπτυξη εφαρμογών είναι κατά κύριο λόγο μεγαλύτερο σε σχέση με αυτό της ανάπτυξης εφαρμογών για Android. Επίσης, το γεγονός ότι το iOS αναπτύσσεται μόνο για τα προϊόντα της Apple, έχει ως αποτέλεσμα να προσφέρεται ένα πιο άρτιο προϊόν, χωρίς σφάλματα (bugs) ή κενά στην ασφάλεια. Οι ενημερώσεις γίνονται απευθείας από την ίδια, χωρίς να χρειάζεται η διαμεσολάβηση του κατασκευαστή της συσκευής, όπως συμβαίνει με τις Android συσκευές.

2.3.4 Μειονεκτήματα iOS

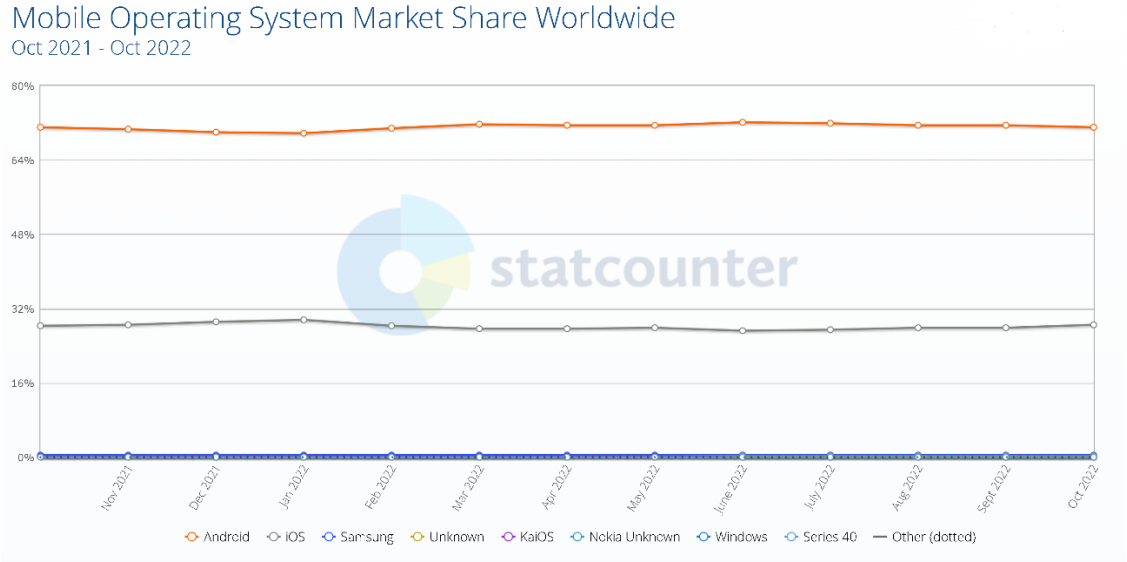
Το μεγαλύτερο πρόβλημα με το iOS είναι η έλλειψη ικανοποιητικής συνδεσιμότητας με άλλες συσκευές πέραν του της γκάμας που προσφέρει η Apple. Τα iPhones και όλες οι συσκευές που έχουν iOS είναι πιο ακριβές για τον μέσο καταναλωτή. Ενώ προσφέρεται ένα «value-based» προϊόν, δεν είναι όμως ιδιαίτερα προσιτές, σε αντίθεση με τις Android συσκευές που μπορεί να προσφέρουν οικονομικότερες λύσεις και ισάξιες σε τεχνικά χαρακτηριστικά hardware. Επίσης οι χρήστες δεν χαίρουν τεράστιες

[7] https://gs.statcounter.com/os-market-share/mobile/worldwide/#monthly-202110-202210

επιλογές εξατομίκευσης (customization), όπως για παράδειγμα αλλαγές στο γραφικό περιβάλλον.

2.3.5 Μερίδιο αγοράς Android και iOS

Το Android έως και σήμερα έχει ηγετική θέση παγκοσμίως συγκριτικά με τα υπόλοιπα λειτουργικά συστήματα. Σε μέτρηση που διεξήχθη τον Οκτώβριο του 2022, το Android OS καταλαμβάνει το 70% του συνολικού μεριδίου αγοράς, ενώ ακολουθεί το iOS λειτουργικό σύστημα με ποσοστό 28%. Το γεγονός αυτό, καθιστά το Android ένα λειτουργικό σύστημα αρκετά δημοφιλές με ένα ευοίωνο μέλλον για την περαιτέρω ανάπτυξη εφαρμογών και λειτουργιών.



Εικόνα 8 Mobile Operating System Market Share Worldwide[8]

[8] https://gs.statcounter.com/os-market-share/mobile/worldwide

3. Ανάπτυξη εφαρμογών Android

Η ανάπτυξη εφαρμογών Android έχει περάσει από αρκετά στάδια για να φτάσει στο σημείο που βρίσκεται σήμερα. Αρχικά, οι προγραμματιστές για την ανάπτυξή τους έκαναν χρήση του εργαλείου ανάπτυξης Eclipse (IDE), το οποίο χρειαζόταν κάποια επιπλέον Add-ons. Αργότερα, η Google βασιζόμενη στο εργαλείο της JetBrains, το IDE IntelliJ IDEA, κυκλοφόρησε το Android Studio, το οποίο και χρησιμοποιεί ακόμη και έως σήμερα.

3.1 Γλώσσες προγραμματισμού Android

Οι γλώσσες προγραμματισμού που μπορούν να χρησιμοποιηθούν για την ανάπτυξη εφαρμογών Android είναι αρκετές. Παρακάτω θα γίνει αναφορά στις βασικότερες, καθώς οι υπόλοιπες είναι αδύνατο να χρησιμοποιηθούν εξ' ολοκλήρου, χωρίς τη χρήση κάποιου ενδιάμεσου εργαλείου ή πλαισίου (Framework). Ωστόσο, η επιλογή της γλώσσας σχετίζεται με το αντικείμενο της εφαρμογής και τις επιλογές για το εκάστοτε project. Οι βασικότερες γλώσσες για τις οποίες θα γίνει αναφορά είναι η Java, η Kotlin, η C# και η Dart.

3.1.1 Java

Ήταν η πρώτη επίσημη γλώσσα προγραμματισμού για την ανάπτυξη εφαρμογών Android, η οποία αργότερα αντικαταστάθηκε από την Kotlin. Διαθέτει πληθώρα βιβλιοθηκών και είναι εύκολο για έναν νέο προγραμματιστή να αναζητήσει πληροφορίες στο διαδίκτυο για την ανάπτυξη της εφαρμογής του, καθώς χρησιμοποιείται εδώ και πολλά χρόνια. Επιπλέον παρέχει την καλύτερη συμβατότητα στις εφαρμογές για android συσκευές. Ωστόσο, είναι πιο περίπλοκη γλώσσα για έναν προγραμματιστή που κάνει τώρα τα πρώτα του «βήματα», καθώς έχει πιο σύνθετη δομή, όπως κατασκευαστές (constructors) και εξαιρέσεις μηδενικών δεικτών(null pointer exceptions).

3.1.2 Kotlin

Η Kotlin είναι αντικειμενοστρεφής γλώσσα προγραμματισμού και βασίζεται στην εικονική μηχανή της Java (JVM). Δημιουργήθηκε από την JetBrains το 2017, είναι «ανοικτού κώδικα» και είναι η πιο δημοφιλής γλώσσα λόγω των εκτεταμένων βιβλιοθηκών που διαθέτει και της περιεκτικής σύνταξης κώδικα. Από το 2019 η Google την χρησιμοποιεί ως κύρια γλώσσα προγραμματισμού ανάπτυξης Android εφαρμογών και μπορεί να χρησιμοποιηθεί αντί της Java. Είναι αρκετά ευέλικτη γλώσσα, καθώς μπορεί εύκολα ο προγραμματιστής να την μετατρέψει σε Java και το αντίστροφο. Η

μόνη σημαντική διαφορά σε σχέση με την Java είναι ότι αφαιρεί πολλά περιττά χαρακτηριστικά της, όπως οι εξαιρέσεις του μηδενικού δείκτη και η ανάγκη να τελειώνει κάθε γραμμή με ερωτηματικό, κάνοντας την κατά αυτόν τον τρόπο πιο απλή για έναν «αρχάριο» προγραμματιστή.

3.1.3 C#

Ισχυρά αντικειμενοστρεφής γλώσσα προγραμματισμού που αναπτύχθηκε από την Microsoft.Αρχικά, το μεγαλύτερο μειονέκτημά της ήταν ότι μπορούσε να «τρέξει» μόνο σε συστήματα Windows καθώς χρησιμοποιούσε το .NET Framework. Ωστόσο, αυτό το πρόβλημα αντιμετωπίστηκε καθώς, πλέον, βασίζεται σε "Common Language Infrastructure", που σημαίνει ότι μπορεί να χρησιμοποιηθεί σε πολλές πλατφόρμες, αξιοποιώντας τα εργαλεία Android. Επιπλέον, το γεγονός ότι διαθέτει πολλά κοινά με την Java καθώς και ότι είναι πιο απλή και καθαρή από την Java, την καθιστά συγκριτικά πιο εύκολη για την ανάπτυξη εφαρμογών Android.

3.1.4 Dart

Η Dart είναι γλώσσα προγραμματισμού «ανοικτού κώδικα» και έχει σχεδιαστεί από την Google ως ταχύτερη γλώσσα προγραμματισμού. Χρησιμοποιεί το "Flutter Framework", με το οποίο ο προγραμματιστής μπορεί να «χτίσει» πολύ γρήγορα και εύκολα εγγενείς εφαρμογές και εφαρμογές ιστού. Εφοδιάζεται με εργαλεία που διευκολύνουν τους προγραμματιστές, ιδιαίτερα στην ανάπτυξη των διεπαφών χρήστη (User Interface). Απευθύνεται αρκετά και σε προγραμματιστές δημιουργίας ηλεκτρονικών παιχνιδιών. Παρέχει την ευελιξία να μπορεί κανείς να την δουλέψει σε συνδυασμό και με άλλες τεχνολογίες που χρησιμοποιούν γλώσσες προγραμματισμού όπως JavaScript, HTML5 και C++, με αποτέλεσμα να μπορεί να δημιουργήσει εφαρμογές είτε πιο απλές είτε πιο απαιτητικές, όπως αυτές των παιχνιδιών.

3.2 Τύποι Εφαρμογών κινητών συσκευών

Τρία είναι τα είδη εφαρμογών για τις συσκευές, οι εγγενείς εφαρμογές (Native Applications), οι εφαρμογές ιστού (Web Applications) και οι υβριδικές εφαρμογές (Hybrid Applications). Παρακάτω θα αναφερθούν οι διαφορές μεταξύ των τριών τύπων καθώς και τα πλεονεκτήματα και μειονεκτήματα τους.

3.2.1 Εγγενείς Εφαρμογές (Native Apps)

Οι εγγενείς εφαρμογές κατασκευάζονται και απευθύνονται σε συγκεκριμένα λειτουργικά συστήματα. Για παράδειγμα, υπάρχουν εγγενείς εφαρμογές για Android

συσκευές και εφαρμογές για λειτουργικό σύστημα iOS. Δεν είναι δυνατόν μια εφαρμογή κατασκευασμένη για Android να μπορεί να χρησιμοποιηθεί και για iOS. Είναι γραμμένες σε γλώσσα προγραμματισμού που υποστηρίζει το λειτουργικό σύστημα στο οποίο απευθύνονται. Οι γλώσσες προγραμματισμού που χρησιμοποιούν τα πιο διαδεδομένα λειτουργικά συστήματα είναι η Java, η Kotlin, η C++, η Objective-C, η Swift, η Python και η React.

Οι εγγενείς εφαρμογές προτιμώνται για την ανάπτυξη εφαρμογών για κινητά λόγω της αξιοπιστίας τους και της ταχύτητάς τους όσον αφορά την απόδοσή τους. Αναπτύσσονται για συγκεκριμένο λειτουργικό σύστημα, με αποτέλεσμα να αξιοποιούνται πλήρως όλοι οι πόροι της συσκευής. Αξιοποιούν πλήρως την εγγενή διεπαφή χρήστη (User Interface), με αποτέλεσμα να παρέχεται βελτιωμένη εμπειρία χρήστη. Η Google και η Apple αναπτύσσουν τα δικά τους εργαλεία ανάπτυξης εφαρμογών, το SDK και τα χαρακτηριστικά που εμπλουτίζουν το UI των εφαρμογών.

Το σημαντικότερο μειονέκτημά τους είναι ότι δεν μπορούν να χρησιμοποιηθούν σε άλλο λειτουργικό σύστημα, με αποτέλεσμα να αυξάνεται το κόστος και η προσπάθεια ανάπτυξης και συντήρησης μιας εφαρμογής που θα πρέπει να απευθύνεται σε πολλά λειτουργικά συστήματα. Σε σύγκριση με τις εφαρμογές ιστού, στις οποίες οι ενημερώσεις γίνονται αυτόματα, για την ενημέρωση μιας εγγενούς εφαρμογής χρειάζεται να μπει ο χρήστης στο κατάστημα για να κάνει λήψη της τελευταίας έκδοσης. Παραδείγματα εγγενών εφαρμογών είναι η εφαρμογή WhatsApp και το Spotify.

Εικόνα 9 Spotify[9]

Εικόνα 10 WhatsApp[10]

[9] https://www.emizentech.com/blog/types-of-mobile-apps.html
[10] https://www.emizentech.com/blog/types-of-mobile-apps.html

3.2.2 Εφαρμογές Ιστού(Web Applications)

Οι εφαρμογές ιστού για κινητές συσκευές είναι ειδικά σχεδιασμένες εφαρμογές για να ανοίγουν μέσω περιηγητή. Η διεπαφή χρήστη (User Interface) είναι έτσι σχεδιασμένη ώστε η εφαρμογή να προσαρμόζεται στην οθόνη της εκάστοτε συσκευής. Δεν χρειάζεται να εγκατασταθεί στην συσκευή καθώς αρκεί μόνο το URL για να επισκεφθεί ο χρήστης την ιστοσελίδα. Κατά κύριο λόγο, οι εφαρμογές ιστού είναι γραμμένες σε γλώσσα προγραμματισμού HTML5, CSS, Ruby και JavaScript.

Το βασικότερο πλεονέκτημα τους, όσον αφορά τον προγραμματιστή, είναι το μειωμένο κόστος ανάπτυξης και συντήρησης καθώς δεν χρειάζεται να απευθύνονται σε συγκεκριμένο λειτουργικό σύστημα. Επομένως, λύνεται το ζήτημα του «Κατακερματισμού» (“Fragmentation”) καθώς διαθέτουν τεχνικά χαρακτηριστικά και λειτουργίες που μπορούν να υποστηρίξουν οι περισσότερες συσκευές, ανεξαρτήτως λειτουργικού συστήματος και οθόνης της συσκευής. Αναφορικά με τον χρήστη, δεν χρειάζεται να δεσμεύσει αποθηκευτικό χώρο και μνήμη RAM στη συσκευή του, καθώς δεν χρειάζεται να γίνει εγκατάσταση της εφαρμογής.

Το κυριότερο μειονέκτημα είναι ότι τις εφαρμογές ιστού μπορεί να τις επισκεφθεί κανείς μόνο μέσω περιηγητή. Αυτό σημαίνει ότι μπορούν να λειτουργήσουν μόνο μέσω internet. Ένα άλλο μειονέκτημα για τις εφαρμογές ιστού είναι ότι δεν είναι τόσο ασφαλείς όσο οι εγγενείς, καθώς εκτίθενται πιο εύκολα σε κακόβουλο λογισμικό και δεν επιτυγχάνουν τον βαθμό διαδραστικότητας που έχουν οι εγγενείς εφαρμογές. Παραδείγματα εφαρμογής ιστού είναι η Google Gmail και η Netflix:

Εικόνα 11 Netflix[11]

Εικόνα 12 Gmail[12]

[11] https://www.emizentech.com/blog/types-of-mobile-apps.html
[12] https://www.emizentech.com/blog/types-of-mobile-apps.html

3.2.3 Υβριδικές Εφαρμογές(Hybrid Applications)

Οι υβριδικές εφαρμογές είναι εφαρμογές ιστού που συμπεριφέρονται σαν εγγενείς εφαρμογές. Οι χρήστες τις εγκαθιστούν στη συσκευή τους όπως ακριβώς τις εγγενείς εφαρμογές, αλλά επί της ουσίας πρόκειται για εφαρμογές ιστού. Είναι «χτισμένες» σε γλώσσα προγραμματισμού HTML, CSS και JavaScript και «τρέχουν» σε ένα WebView. Μπορούν να χρησιμοποιήσουν το GPS, την κάμερα, το μικρόφωνο αλλά και άλλες λειτουργίες της συσκευής. Ένα είδος εφαρμογής ιστού είναι η «προοδευτική εφαρμογή ιστού» (PWA), η οποία είναι κατά βάση μια εγγενής εφαρμογή που εκτελείται μέσα από περιηγητή.

Μερικά από τα πλεονεκτήματα τους είναι η γρήγορη και μικρή σε κόστος ανάπτυξή τους, μπορεί να αναπτυχθεί για πολλά λειτουργικά συστήματα και μπορεί να χρησιμοποιεί τους πόρους της συσκευής. Επιπλέον, δεν χρειάζεται να αποκτήσει πρόσβαση ο χρήστης μέσω περιηγητή, σε αντίθεση με τις εφαρμογές ιστού. Ωστόσο, αποτελούν πιο ακριβή λύση σε σύγκριση με τις εφαρμογές ιστού και είναι πιο αργές και λιγότερο διαδραστικές και παραμετροποιήσιμες συγκριτικά με τις εγγενείς εφαρμογές.

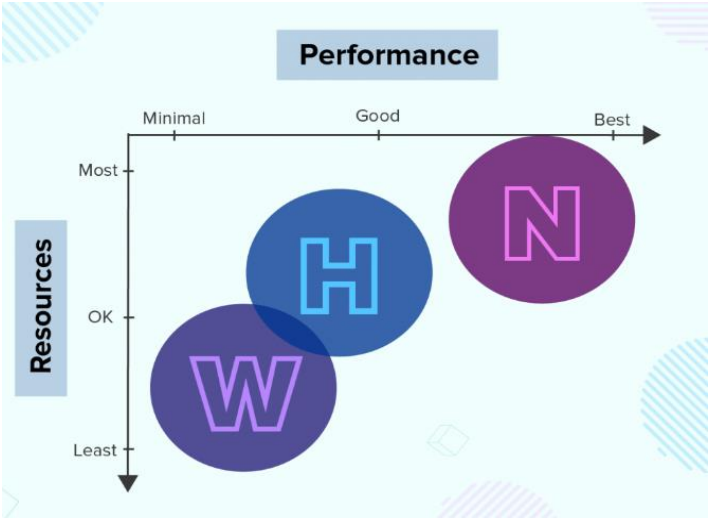
Εικόνα 13 Twitter[13]

Εικόνα 14 Facebook Instagram[14]

3.2.4 Καταλληλότερος Τύπος

Η επιλογή του κατάλληλου τύπου εξαρτάται από πολλές παραμέτρους, οι οποίες πρέπει να εξεταστούν. Τέτοιες παράμετροι είναι το “budget” που είναι διαθέσιμο για την ανάπτυξη της εφαρμογής, ο βαθμός της εμπειρίας χρήστη (User Experience) που θα προσφέρεται και οι απαιτήσεις της εφαρμογής σε τεχνικά χαρακτηριστικά και λειτουργίες. Στο παρακάτω διάγραμμα είναι εμφανείς οι απαιτήσεις σε πόρους και επιδόσεις του εκάστοτε τύπου εφαρμογών. Οι εφαρμογές ιστού έχουν τις λιγότερες απαιτήσεις σε επιδόσεις και πόρους σε αντίθεση με τις εγγενείς εφαρμογές. Οι υβριδικές εφαρμογές βρίσκονται περίπου στη μέση συγκριτικά με τους άλλους δύο τύπους.

[13] https://www.emizentech.com/blog/types-of-mobile-apps.html
[14] https://elphicks.co.uk/social-media/insta-amd-facebk/



Εικόνα 15 Performance[15]

4. Λειτουργικό Σύστημα Android και Τεχνολογίες

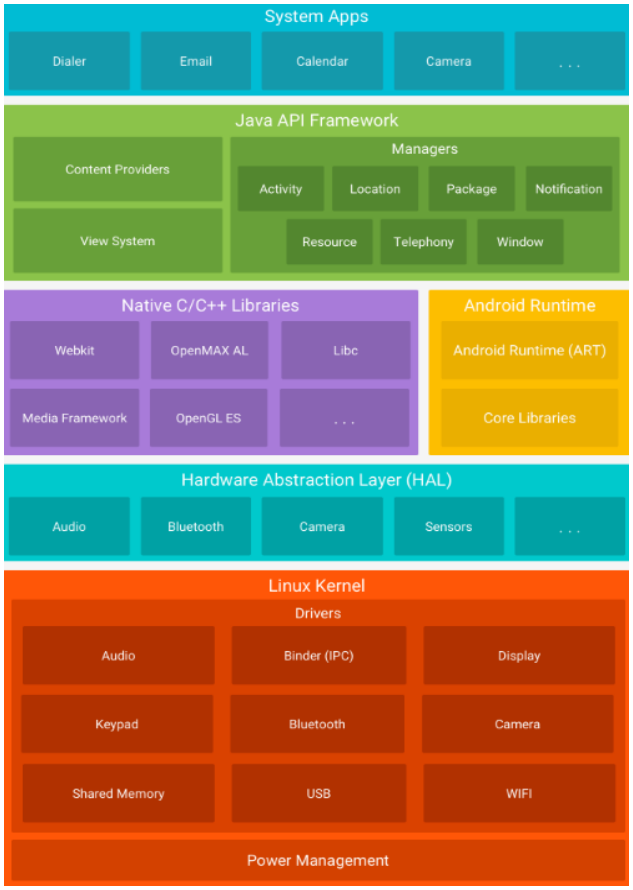
Το Android είναι λειτουργικό σύστημα «ανοικτού κώδικα» που βασίζεται στον πυρήνα του λειτουργικού Linux σχεδιασμένο για να απευθύνεται σε ευρύ φάσμα συσκευών πέραν των έξυπνων τηλεφώνων, όπως τηλεοράσεις, ρολόγια, tablets και πλέον και στα αυτοκίνητα.

Το Android αρχικά αναπτυσσόταν από την "Android Incorporation". Αργότερα, το 2008, εξαγοράστηκε από την Google και αναπτύχθηκε από μια κοινοπραξία προγραμματιστών γνωστή ως Open Handset Alliance. Η πρώτη Android συσκευή παρουσιάστηκε από την HTC στα τέλη του 2008 και ακολούθησαν και άλλοι κατασκευαστές. Η Google ξεκίνησε να παρέχει το λειτουργικό σύστημα σε κατασκευαστές με την υπόσχεση να παρέχει ένα ευέλικτο λειτουργικό σύστημα βασισμένο στο λειτουργικό πυρήνα Linux. Το γεγονός αυτό σύντομα προσέλκυσε πολλούς κατασκευαστές κινητής και αρκετούς προγραμματιστές να συμβάλλουν στην ανάπτυξη εφαρμογών. Σήμερα, το Android έχει σημαντικές βελτιώσεις και πρόσθετες λειτουργίες, με το μέλλον του λειτουργικού να είναι αισιόδοξο.

[15] https://clevertap.com/blog/types-of-mobile-apps/

4.1 Κύρια Στοιχεία Αρχιτεκτονικής του Λειτουργικού Συστήματος Android

Όπως φαίνεται και στο παρακάτω διάγραμμα, το λειτουργικό σύστημα Android απαρτίζεται από κάποια κύρια επίπεδα, το καθένα από τα οποία απαρτίζεται από άλλα υποστοιχεία. Κάθε ανώτερο επίπεδο χρησιμοποιεί τις υπηρεσίες του κατώτερου του επιπέδου.



Εικόνα 16 Platform Architecture[16][17]

4.1.1 Linux Kernel

Ο Linux Kernel αποτελεί το χαμηλότερο επίπεδο της αρχιτεκτονικής του λειτουργικού συστήματος. Είναι ένας μονολιθικός πυρήνας που διαχειρίζεται την επικοινωνία μεταξύ του hardware και του software. Περιλαμβάνει όλες τις απαραίτητες πληροφορίες, όπως hardware drivers για την κάμερα, το Bluetooth και τα ηχεία ώστε να υπάρχει επικοινωνία με το software, και αντιστρόφως. Διαθέτει κάποιες ειδικές δυνατότητες για βέλτιστη διαχείριση μνήμης, εξοικονόμηση ενέργειας αλλά και άλλες διεργασίες.

[16] https://developer.android.com/guide/platform
[17] https://www.tech-term.in/2018/01/22/art-android-runtime/

4.1.2 Hardware Abstraction Layer (HAL)

Το HAL είναι ένα επίπεδο που γεφυρώνει το χάσμα στην επικοινωνία μεταξύ του software και του hardware. Μια εφαρμογή android επικοινωνεί με το hardware μέσω Java APIs. Το HAL Layer παρέχει τις τυπικές διεπαφές (interfaces) που εκθέτουν τις δυνατότητες υλικού (hardware) της συσκευής στο υψηλότερο επίπεδο πλαισίου Java API (Java API Framework). Αποτελείται από πολλές βιβλιοθήκες, καθεμιά από τις οποίες υλοποιεί μια διεπαφή για έναν συγκεκριμένο τύπο hardware, όπως η κάμερα ή το Bluetooth. Όταν ένα πλαίσιο API πραγματοποιεί αίτημα για πρόσβαση στο υλικό (hardware), τότε φορτώνεται από το σύστημα η συγκεκριμένη βιβλιοθήκη.

4.1.3 Android Runtime

Το Android Runtime είναι περιβάλλον που χρησιμοποιείται από το λειτουργικό σύστημα Android για την εκτέλεση των εφαρμογών. Είναι διάδοχος της προηγούμενης εικονικής μηχανής “Dalvik” που ήταν μέρος του λειτουργικού συστήματος μέχρι την έκδοση 4.4 (KitKat). Το ART (Android Runtime) εισήγαγε την μεταγλώττιση των εφαρμογών “Ahead of Time”(AOT). Όταν μια εφαρμογή είναι εγκατεστημένη, δημιουργεί αρχεία τύπου .dex κατά τη διάρκεια της μεταγλώττισης, τα οποία αναλαμβάνει να εκτελέσει το λειτουργικό σύστημα. Το αποτέλεσμα είναι ότι βελτιώνεται σε μεγάλο βαθμό η απόδοση εκτέλεσης της εφαρμογής με χαμηλότερες απαιτήσεις σε μνήμη RAM και επεξεργαστή και μικρότερη κατανάλωση ενέργειας. Μεταξύ των άλλων πλεονεκτημάτων, παρέχει δυνατότητες εντοπισμού και συλλογής σφαλμάτων με αποτέλεσμα να μην παρεμποδίζεται η εκτέλεση των υπολοίπων αρχείων.

4.1.4 Native Libraries

Πολλά δομικά στοιχεία του android, όπως το HAL και το ART, χρησιμοποιούν βιβλιοθήκες που είναι γραμμένες σε γλώσσα προγραμματισμού C και C++. Αυτές οι εγγενείς βιβλιοθήκες μπορούν να χρησιμοποιηθούν από τις εφαρμογές μόνο μέσω Java Framework APIs. Για την μεταγλώττιση και απασφαλμάτωση του εγγενούς κώδικα, χρειάζεται το εργαλείο “Native Development Kit”(NDK), το οποίο επιτρέπει την χρήση κώδικα C και C++.

4.1.5 Java API Framework

Το πλαίσιο Java API διαθέτει όλα τα χαρακτηριστικά, περιλαμβανομένων των βιβλιοθηκών, που απαιτούνται για την αξιοποίηση των υλικών (hardware) της συσκευής. Τέτοιοι πόροι είναι οι λειτουργίες GPS και εντοπισμού θέσης και ο τρόπος

που γίνονται οι μετρήσεις της τοποθεσίας. Επίσης, η διαχείριση της τηλεφωνίας και για το πως συνδυάζεται το υλικό (hardware), όπως η κάρτα sim, το μικρόφωνο και η κάμερα, με τα στοιχεία λογισμικού(software), όπως το πληκτρολόγιο κλήσης, η ειδοποίηση, το λογισμικό που χρησιμοποιεί την κάμερα και άλλες λειτουργικότητες.

4.1.6 Android Applications

Είναι το υψηλότερο επίπεδο αρχιτεκτονικής και πάνω στο οποίο αλληλεπιδρούν οι τελικοί χρήστες. Εδώ οι προγραμματιστές εμφανίζουν τις εφαρμογές ώστε να μπορούν να χρησιμοποιηθούν από τον χρήστη. Υπάρχουν ήδη και προεγκατεστημένες εφαρμογές, όπως εφαρμογή με τις επαφές, εφαρμογή αποστολής SMS, email και περιηγητής Internet.

4.2 Δομικά Στοιχεία εφαρμογών και AndroidManifest.xml

Κάθε εφαρμογή έχει ένα αρχείο AndroidManifest.xml, το οποίο εμπεριέχει απαραίτητες πληροφορίες σχετικά με την εφαρμογή. Σε αυτό το αρχείο ορίζονται οι δραστηριότητες (Activities), οι υπηρεσίες (Services), οι προθέσεις (Intents), οι δέκτες μηνυμάτων Broadcast (Broadcast Receivers) και οι πάροχοι περιεχομένου (Content Providers) που θα χρησιμοποιηθούν. Εξαρτάται από τα χαρακτηριστικά της εφαρμογής για το ποια στοιχεία θα χρησιμοποιηθούν καθώς και ο τρόπος που θα χρησιμοποιηθούν. Κάθε στοιχείο από αυτά ορίζει βασικές ιδιότητες και δυνατότητες, όπως για παράδειγμα είναι οι διαμορφώσεις της συσκευής και τα φίλτρα πρόθεσης (Intent Filters), τα οποία ορίζουν πως να εκκινείται το εκάστοτε στοιχείο. Επίσης, σε αυτό το αρχείο μπορούν να οριστούν τα δικαιώματα (permissions) που χρειάζεται η εφαρμογή αναφορικά με την πρόσβαση σε προστατευμένα μέρη του συστήματος, την πρόσβαση της σε άλλες εφαρμογές και πιο ειδικά ορίζει ποια δικαιώματα πρέπει να έχουν άλλες εφαρμογές. Επιπλέον, μπορεί να δηλωθεί ποιες είναι οι απαιτήσεις σε υλικό και λογισμικό που χρειάζεται η εφαρμογή στην Android συσκευή. Παρακάτω, ακολουθεί η αναλυτική επισκόπηση αυτών των στοιχείων.

4.2.1 Activity

Αντιπροσωπεύει την οθόνη διεπαφής του χρήστη. Είναι παραθυρικού τύπου και μέσω αυτής αλληλεπιδρούν οι χρήστες, όπως το να στείλουν ένα email, να τραβήξουν φωτογραφία ή να πάρουν τηλέφωνο. Μια εφαρμογή μπορεί να αποτελείται από πολλές δραστηριότητες, οι οποίες και συνδέονται μεταξύ τους με μια λογική αλληλουχία. Ορίζεται μια δραστηριότητα ως κύρια (Main Activity), η οποία συνδέεται με άλλες

δραστηριότητες. Κάθε δραστηριότητα μπορεί να εκκινήσει μια ή πολλές δραστηριότητες οι οποίες και πραγματοποιούν διαφορετικές εντολές. Όταν εκκινείται μια νέα δραστηριότητα, τότε η προηγούμενη σταματά να τρέχει, αλλά το σύστημα την διατηρεί σε κατάσταση αναμονής, το λεγόμενο “back stack”. Ακολουθείται η λογική LIFO (last in first out) και όταν ο χρήστης πατήσει το «κουμπί επιστροφής», τότε επιστρέφει και μπορεί να συνεχίσει στην προηγούμενη δραστηριότητα.

4.2.2 Υπηρεσίες (Services)

Είναι στοιχείο της εφαρμογής το οποίο εκτελεί λειτουργίες ακόμη και στο παρασκήνιο χωρίς να χρειάζεται να είναι ανοικτή η οθόνη διεπαφής της εφαρμογής. Οι υπηρεσίες δεν παρέχουν οθόνη διεπαφής του χρήστη (user interface). Μια υπηρεσία μπορεί να εκκινηθεί από μια εφαρμογή και να συνεχίσει να εκτελείται και να εμφανίσει αποτελέσματα, ακόμη και όταν ο χρήστης έχει μεταβεί σε άλλη εφαρμογή. Για παράδειγμα, ο χρήστης μπορεί να ακούει μουσική στο παρασκήνιο ή να δεχτεί κάποιο email, ακόμη και όταν δεν έχει ανοικτή την συγκεκριμένη εφαρμογή.

4.2.3 Προθέσεις (Intents)

Τα τρία από τα πέντε στοιχεία της εφαρμογής, οι δραστηριότητες, οι υπηρεσίες και οι δέκτες μηνυμάτων Broadcast, ενεργοποιούνται μέσω μηνυμάτων από τις προθέσεις(Intents). Οι προθέσεις δεν παρέχουν διεπαφές χρήστη. Μια πρόθεση περιέχει την πληροφορία για την ενέργεια που πρέπει να εκτελεστεί, δεδομένα για αυτή την ενέργεια αλλά και πολλές άλλες πληροφορίες. Με αυτό τον τρόπο μεταφέρονται πληροφορίες από μια δραστηριότητα σε μια άλλη ή μπορούν και να επιστραφούν πληροφορίες από μια άλλη δραστηριότητα. Το ίδιο ισχύει και για μια υπηρεσία. Μια πρόθεση μπορεί να εκκινήσει μια υπηρεσία ή να μεταφέρει πληροφορίες σε μια νέα υπηρεσία, δημιουργώντας έναν δεσμό μεταξύ των υπηρεσιών.

4.2.4 Δέκτες μηνυμάτων Broadcast

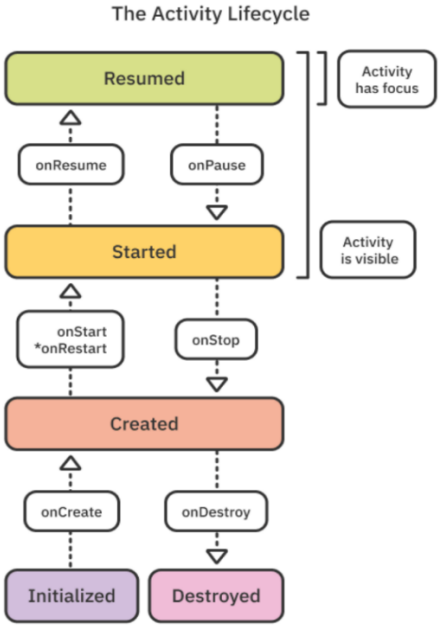
Οι δέκτες εκπομπής μηνυμάτων Broadcast απαντούν σε μηνύματα εκπομπής από άλλες εφαρμογές. Για παράδειγμα, μπορεί μια εφαρμογή να ξεκινήσει να εκπέμπει κάποιο μήνυμα για να ενημερώσει άλλη εφαρμογή ότι έχει γίνει λήψη ενός αρχείου στη συσκευή και είναι διαθέσιμη προς χρήση. Άλλες τέτοια παραδείγματα είναι η ένδειξη της στάθμης της μπαταρίας και ειδοποιήσεις από άλλες εφαρμογές.

4.2.5 Πάροχοι Περιεχομένου (Content Providers)

Οι πάροχοι περιεχομένου διαχειρίζονται δομημένους αποθηκευτικούς χώρους που μπορεί να χρησιμοποιήσουν πολλές εφαρμογές. Παρέχουν μηχανισμούς για την ασφάλεια των δεδομένων. Για να αποκτήσει μια εφαρμογή πρόσβαση στα δεδομένα του παρόχου περιεχομένου, χρησιμοποιεί το αντικείμενο “ContentResolver”, που αυτό με την σειρά του επικοινωνεί με το αντικείμενο του παρόχου, το οποίο είναι μια κλάση. Κατά αυτόν τον τρόπο οι εφαρμογές μπορούν μέσω των κλάσεων να αποκτήσουν πρόσβαση σε συγκεκριμένα δεδομένα. Το ίδιο το Android περιλαμβάνει παρόχους περιεχομένου που διαχειρίζεται δεδομένα, όπως εικόνες, βίντεο, ήχο και άλλα δεδομένα.

4.3 Κύκλος ζωής δραστηριοτήτων (The Activity lifecycle)

Οι δραστηριότητες είναι τα βασικά στοιχεία που δημιουργούν το περιβάλλον διεπαφής του χρήστη, το “User Interface”. Όπως έχει αναφερθεί και παραπάνω, μια εφαρμογή μπορεί να αποτελείται από πολλές δραστηριότητες. Όταν εκκινείται η εφαρμογή, υπάρχει μια κύρια δραστηριότητα (Main Activity), η οποία με την σειρά της εκκινεί άλλες δραστηριότητες που συνδέονται με αυτήν. Καθώς ο χρήστης πλοηγείται στις διάφορες δραστηριότητες, αυτές περνούν μέσα από διαφορετικές καταστάσεις του κύκλου ζωής. Στο παρακάτω διάγραμμα παρουσιάζονται οι διαφορετικές καταστάσεις που καλούνται κατά τη διάρκεια ζωής μιας δραστηριότητας.



Εικόνα 17 Exploring the Activity Lifecycle[18]

[18] https://www.kodeco.com/21382977-android-lifecycle

4.3.1 onInitialize:

Αρχικοποιείται η περίπτωση (Instance) της δραστηριότητας και αρχικοποιούνται οι ιδιότητες.

4.3.2 onCreate:

Η δραστηριότητα πλέον αφού έχει αρχικοποιηθεί, διαμορφώνει το περιβάλλον διεπαφής χρήστη. Εδώ αρχικοποιούνται οι βασικές μεταβλητές της δραστηριότητας και που σχετίζονται με το γραφικό περιβάλλον. Καλείται με το που πατήσει ο χρήστης το εικονίδιο της εφαρμογής. Εκτελείται μόνο μια φορά όταν εκκινείται η εφαρμογή.

4.3.3 onStart:

Αμέσως μετά την onCreate, καλείται η μέθοδος αυτή. Η μέθοδος καλείται λίγο πριν γίνει ορατή η δραστηριότητα στον χρήστη. Εκτελούνται συγκεκριμένες εργασίες για να μπει η δραστηριότητα στο προσκήνιο και να αρχίσει να αλληλεπιδρά ο χρήστης με αυτήν.

4.3.4 onResume:

Σε αυτή την κατάσταση ο χρήστης βρίσκεται στην δραστηριότητα και αλληλεπιδρά με αυτήν. Στην περίπτωση που ο χρήστης λάβει μια κλήση στη συσκευή του, η δραστηριότητα «παγώνει» και με το πέρας της κλήσης η δραστηριότητα θα επιστρέψει στο σημείο που βρισκόταν χωρίς να έχει χαθεί κάποια πληροφορία. Η μέθοδος αυτή, επομένως, καλείται όταν ο χρήστης επιστρέψει σε αυτή και στο σημείο που είχε μείνει.

4.3.5 onPause:

Η μέθοδος αυτή καλείται όταν η δραστηριότητα τεθεί στο παρασκήνιο και γίνει αόρατη για να πάρει την θέση της μια άλλη εφαρμογή. Η δραστηριότητα θα παραμείνει σε αυτήν την κατάσταση έως ότου ο χρήστης να επιστρέψει πάλι σε αυτή στην κατάσταση που την άφησε.

4.3.6 onStop:

Η μέθοδος αυτή καλείται όταν η δραστηριότητα διακοπεί από τον χρήστη. Ο χρήστης πλέον δεν θα βρίσκεται πλέον σε αυτή. Ακολουθείται από την μέθοδο onRestart() όταν ανακαλείται η δραστηριότητα ή από την μέθοδο onDestroy() όταν τερματίζεται.

4.3.7 onDestroy:

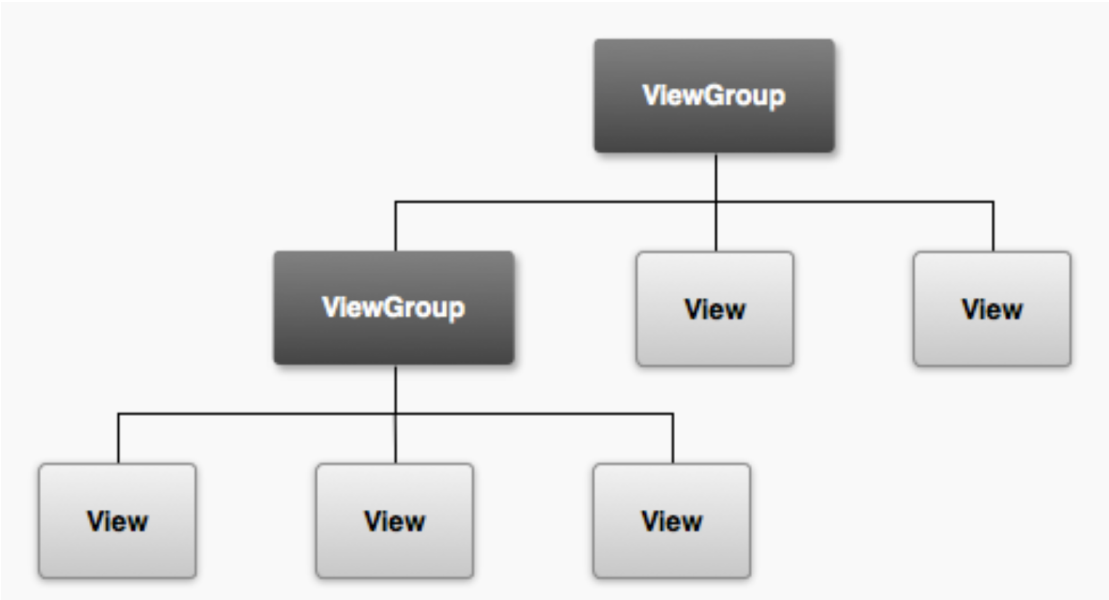
Η μέθοδος καλείται προτού μια δραστηριότητα καταστραφεί. Αυτό μπορεί να οφείλεται είτε στο γεγονός ότι ο χρήστης απορρίπτει εντελώς την δραστηριότητα, είτε

το ίδιο το σύστημα καταστρέφει προσωρινά τη δραστηριότητα λόγω κάποιας μεταβολής της διαμόρφωσης, όπως στην περίπτωση που περιστραφεί η συσκευή ή κατά την λειτουργία πολλαπλών παραθύρων.

4.4 Διεπαφή Χρήστη (User Interface)

Η διεπαφή χρήστη περιλαμβάνει όλο το γραφικό περιβάλλον με το οποίο αλληλεπιδρά ο χρήστης. Το Android περιλαμβάνει έτοιμα στοιχεία, όπως διατάξεις (layouts), μενού και ειδοποιήσεις, τα οποία επιτρέπουν στον προγραμματιστή να «χτίσει» ένα γραφικό περιβάλλον διεπαφής για την εφαρμογή.

Όλα τα στοιχεία διεπαφής του χρήστη σε μια Android εφαρμογή είναι «χτισμένα» σε προβολές (Views) και ομάδες προβολών (Viewgroups). Προβολή είναι ένα γραφικό αντικείμενο στην οθόνη, με το οποίο αλληλεπιδρά ο χρήστης, όπως είναι ένα κουμπί ή ένα πεδίο κειμένου(text field). Επιπλέον, μια δομή δεδομένων όπου αποθηκεύονται οι παράμετροι και το περιεχόμενο της διάταξης (layout). Ομάδα προβολής είναι ένα αντικείμενο που απαρτίζει προβολές ή ακόμη και ομάδες προβολών προκειμένου να ορίσει την διάταξη (layout) της διεπαφής.



Εικόνα 18 Illustration of a view hierarchy, which defines a UI layout[19]

4.5 Παράμετροι Διάταξης (Layout Parameters)

Όπως αναφέρθηκε και παραπάνω, η διεπαφή χρήστη είναι οργανωμένη ιεραρχικά σε προβολές, οι οποίες ανήκουν σε ομάδες προβολών. Αντιστοίχως, κάθε ομάδα προβολής αποτελείται από πολλές προβολές. Το γραφικό περιβάλλον δημιουργείται

[19] https://stuff.mit.edu/afs/sipb/project/android/docs/guide/topics/ui/overview.html

μέσω αρχείων .xml. Τα χαρακτηριστικά των διατάξεων .xml ορίζουν τις κατάλληλες παραμέτρους της προβολής, η οποία θα ανήκει σε μια ομάδα προβολών. Όπως φαίνεται και στην παρακάτω εικόνα, κάθε ομάδα προβολών ορίζει παραμέτρους διατάξεων για κάθε επιμέρους προβολή.

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_marginHorizontal="20dp"
    android:layout_marginTop="10dp">

    <com.example.filopaideusismvvm.views.MaskedCardView
        android:id="@+id/sectionsButton"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:clickable="true"
        app:cardElevation="2dp"
        app:cardPreventCornerOverlap="false"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:shapeAppearanceOverlay="@style/ShapeAppearance.Sunflower.Card">

        <androidx.constraintlayout.widget.ConstraintLayout
            style="@style/BackgroundColorSectionItem"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:minHeight="50dp">

            <TextView
                android:id="@+id/sectionsButtonText"
                style="@style/TextColorWhite.big"
                android:layout_width="0dp"
                android:layout_height="wrap_content"
                android:layout_marginStart="10dp"
                android:maxLines="3"
                app:layout_constraintEnd_toStartOf="@+id/infoButton"
                app:layout_constraintStart_toStartOf="parent"
                app:layout_constraintTop_toTopOf="parent"
                tools:text="Title" />
```

Εικόνα 19 Παράδειγμα XML κώδικα

4.6 Τα είδη των διατάξεων (layouts)

Παρακάτω θα γίνει αναφορά στα είδη των διατάξεων καθώς και τα χαρακτηριστικά τους.

- Γραμμική Διάταξη (LinearLayout)
- Σχετική Διάταξη (Relative Layout)
- Διάταξη Πλαισίου (Frame Layout)
- Διάταξη Πίνακα (Table Layout)

4.6.1 Γραμμική διάταξη (Linear Layout)

Είναι μια ομάδα προβολής που ορίζει για τις επιμέρους προβολές της να είναι ευθυγραμμισμένες σε μια ενιαία κατεύθυνση, είτε κάθετα είτε οριζόντια. Η κατεύθυνση της διάταξης ορίζεται με την παράμετρο android:orientation.

4.6.2 Σχετική Διάταξη (Relative Layout)

Είναι μια ομάδα προβολών που ορίζει παραμέτρους για την θέση των επιμέρους προβολών. Η θέση τους είναι «σχετική» καθώς είναι δυνατόν να οριστεί σε οποιαδήποτε θέση επιθυμεί ο χρήστης ή να οριστεί η θέση σε σχέση με κάποιο άλλο αντικείμενο. Παρόμοια και με τη Σχετική Διάταξη είναι και η Διάταξη Περιορισμού(Constraint Layout). Επιτρέπει να δημιουργούνται σύνθετες και ευέλικτες διατάξεις με ιεραρχία επίπεδης προβολής και χωρίς εμφωλευμένες ομάδες προβολών. Όλες οι προβολές διατάσσονται σύμφωνα με τις σχέσεις που έχουν οι «αδελφικές» προβολές με τις γονικές διατάξεις.

```
<androidx.constraintlayout.widget.ConstraintLayout
    style="@style/BackgroundColorSectionItem"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:minHeight="50dp">

    <TextView
        android:id="@+id/sectionsButtonText"
        style="@style/TextColorWhite.big"
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:layout_marginStart="10dp"
        android:maxLines="3"
        app:layout_constraintEnd_toStartOf="@+id/infoButton"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        tools:text="Title" />

    <TextView
        android:id="@+id/sectionsButtonSubText"
        style="@style/NormalText"
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:layout_marginStart="5dp"
        android:maxLines="3"
        app:layout_constraintEnd_toStartOf="@+id/infoButton"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/sectionsButtonText"
        tools:text="Sub-title" />

    <ImageView
        android:id="@+id/infoButton"
        android:layout_width="40dp"
        android:layout_height="40dp"
        android:src="@drawable/ic_baseline_info_24"
        android:visibility="gone"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintTop_toTopOf="parent"
```

Εικόνα 20 Παράδειγμα XML κώδικα

4.6.3 Διάταξη Πλαισίου(Frame Layout)

Δεσμεύει έναν κενό χώρο στην οθόνη για να μπορεί να προβληθεί κάποιο αντικείμενο, όπως μια εικόνα. Είναι μια ομάδα προβολών όπου καθορίζεται η θέση των επιμέρους της προβολών σε μια ενιαία προβολή στην οθόνη. Όλες οι επιμέρους προβολές προστίθενται σε μορφή στοίβας, που σημαίνει ότι η προβολή που βρίσκεται στο επάνω μέρος της στοίβας, θα εμφανίζεται στο επάνω μέρος της οθόνης. Ωστόσο μπορεί να οριστεί και η θέση της κάθε προβολής, χρησιμοποιώντας την παράμετρο «βαρύτητας»(“gravity attribute”).

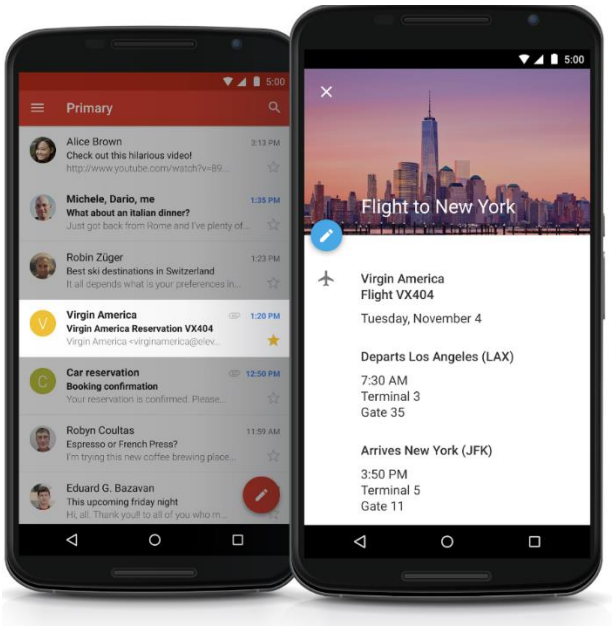
4.6.4 Διάταξη Πίνακα (Table Layout)

Είναι μια ομάδα προβολών που ορίζει τη θέση των επιμέρους προβολών σε γραμμές και στήλες. Για τις γραμμές, τις στήλες και τα κελιά τους δεν εμφανίζονται γραμμές περιγράμματος (borders). Ο πίνακας έχει τόσες στήλες, όσες έχει και η γραμμή με τα περισσότερα κελιά. Ένας πίνακας μπορεί να αφήσει άδεια κελιά. Τα κελιά μπορούν να εκτείνονται σε πολλές στήλες, όπως μπορούν και στην HTML, χρησιμοποιώντας το πεδίο span στην κλάση TableRow.LayoutParams.

4.7 Material Design

Αναπτύχθηκε από μια ομάδα προγραμματιστών και σχεδιαστών UX της Google. Πρόκειται για σχεδιαστικούς κανόνες αναφορικά με το UI, ώστε να γίνεται πιο εύχρηστο και φιλικό το γραφικό περιβάλλον στον χρήστη. Βασίζεται στο σχεδιασμό της διάταξης πίνακα (grid layout) που είναι χαρακτηριστικό που χρησιμοποιείται στο «μοντέλο των καρτών».

Κύρια χαρακτηριστικά του Material Design είναι τα διάφορα εφέ των γραφικών στοιχείων όταν ο χρήστης πλοηγείται μέσα στην εφαρμογή, όπως αντικείμενα να αιωρούνται, σκίαση σε ένα μέρος της οθόνης και παραπάνω φωτεινότητα σε άλλο σημείο, διαβαθμισμένα γραφικά, τρισδιάστατη απεικόνιση χρωμάτων, ώστε από αισθητικής πλευράς να δίνεται μια όσο το δυνατόν πιο ευχάριστη απεικόνιση στον χρήστη. Το Material Design εμφανίστηκε πρώτη φορά στην έκδοση Android 5.0.1 (Lollipop).



Εικόνα 21 Material Design[20]

4.8 Δικαιώματα (Permissions)

Όταν η εφαρμογή χρειάζεται να αποκτήσει πρόσβαση σε κάποιες από τις προστατευμένες λειτουργίες της εφαρμογής, όπως πρόσβαση στην κάμερα ή στην αποστολή SMS, τότε ο χρήστης να δώσει την απαραίτητη άδεια για να το κάνει.

Παλαιότερα, πριν από την έκδοση Android Marshmallow, τα δικαιώματα καθορίζονταν κατά την εγκατάσταση και προσδιορίζονταν από τον προγραμματιστή στο αρχείο AndroidManifest.xml εντός του project της εφαρμογής, όπου εκεί παρουσιάζεται ολόκληρη η λίστα των δικαιωμάτων. Για τον ίδιο τον χρήστη, όταν εγκαθιστούσε μια εφαρμογή από το Google Play Store, παρουσιαζόταν μια λίστα όλων των αδειών που απαιτούσε η εφαρμογή. Αν δεν γινόταν αποδοχή όλων των αδειών, ο χρήστης δεν μπορούσε να προχωρήσει στην εγκατάσταση της εφαρμογής.

Μετά την έκδοση Android Marshmallow, τα δικαιώματα θα ζητηθούν κατά την εκτέλεση της εφαρμογής (runtime permissions), σε αντίθεση με τον τρόπο που ζητούνταν μέχρι εκείνη την έκδοση. Ο χρήστης έχει την δυνατότητα εκείνη την στιγμή να επιτρέψει ή να απαγορέψει το δικαίωμα για πρόσβαση της εφαρμογής σε λειτουργίες, όπως για παράδειγμα στην χρήση της τοποθεσίας (GPS). Έχει το δικαίωμα ακόμη και σε ύστερο χρόνο να απαγορέψει κάποιο δικαίωμα που είχε αποδεχθεί παλαιότερα.

[20] https://eu.usatoday.com/story/tech/talkingtech/2018/04/13/use-gmail-googles-email-service-expected-get-new-look/513956002/

Παρακάτω θα γίνει αναφορά στα είδη των δικαιωμάτων από την έκδοση Marshmallow και μετά. Αυτά διακρίνονται στα «κανονικά δικαιώματα» (“Normal Permissions”) και στα «επικίνδυνα δικαιώματα» (“Runtime permissions”).

4.8.1 Κανονικά δικαιώματα (Normal Permissions)

Αφορά συγκεκριμένα δικαιώματα, τα οποία χαρακτηρίζει η Google ως ασφαλή. Αφορούν την πρόσβαση της εφαρμογής σε πόρους ή δεδομένα όπου η πιθανότητα να «βλάψουν» είναι πολύ μικρή, όπως είναι η πρόσβαση στην κατάσταση του internet (ACCESS_NETWORK_STATE). Δεν ζητείται άδεια από τον χρήστη και χορηγούνται αυτόματα κατά την εγκατάσταση της εφαρμογής. Τα κανονικά δικαιώματα ορίζονται στο αρχείο AndroidManifest.xml. Στο παρακάτω στιγμιότυπο φαίνεται το δικαίωμα πρόσβασης στην κατάσταση του Internet, όπως έχει οριστεί στην εφαρμογή.

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.example.filopaideusismvvm">

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

Εικόνα 22 AndroidManifest.xml

4.8.2 Επικίνδυνα δικαιώματα (Runtime permissions)

Είναι τα δικαιώματα που χρειάζονται την έγκριση του χρήστη κατά την εκτέλεση της εφαρμογής. Αμέσως πριν την πρόσβαση της εφαρμογής σε συγκεκριμένη λειτουργία, ζητείται μέσω «διαλόγου» η άδεια από τον χρήστη. Στην περίπτωση που δεν αποδεχτεί το δικαίωμα, δεν θα μπορεί να έχει πρόσβαση στην συγκεκριμένη λειτουργία. Επικίνδυνα δικαιώματα θεωρούνται η πρόσβαση στην κάμερα και στην τοποθεσία, η πρόσβαση σε αισθητήρες, η ανάγνωση και εγγραφή στις επαφές και στα μηνύματα.

Για να καθοριστεί το δικαίωμα, θα πρέπει να το προσθέσει ο προγραμματιστής στο αρχείο AndroidManifest.xml, όπως στο παρακάτω παράδειγμα το οποίο αφορά το δικαίωμα για πρόσβαση στις επαφές της συσκευής. Έπειτα θα πρέπει να αρχικοποιήσει το αίτημα του δικαιώματος μέσα στην δραστηριότητα (Activity) ή στο πλαίσιο (Fragment).

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

    <uses-permission android:name="android.permission.READ_CONTACTS" />
    ...
</manifest>
```

Εικόνα 23 Read Contacts Permission21

4.9 Παράθυρα διαλόγου (Dialogs)

Διάλογοι είναι μικρά παράθυρα τα οποία αναδύονται στην τρέχουσα δραστηριότητα. Όταν αναδύονται, ο χρήστης δεν έχει τη δυνατότητα να αλληλεπιδράσει με τη δραστηριότητα αλλά θα πρέπει να αλληλεπιδράσει με το παράθυρο διαλόγου. Συνήθως χρησιμοποιείται είτε για την ειδοποίηση του χρήστη για κάποιο γεγονός που θα ακολουθήσει είτε για να θέσει ερώτημα στον χρήστη για κρίσιμη απόφαση που αφορά την επόμενη ενέργειά του.

Ο διάλογος ειδοποίησης (Alert dialog) και ο διάλογος προόδου (Process Dialog) είναι τα βασικά είδη διαλόγου.

4.9.1 Alert Dialog

Είναι παράθυρο διαλόγου που έχει την δυνατότητα να εμφανίσει έναν τίτλο, ένα μήνυμα, έως τρία κουμπιά, μια λίστα από επιλογές ή μια εξατομικευμένη (custom) διάταξη. Η κλάση AlertDialog.Builder παρέχει πλαίσια API (API frameworks) τα οποία επιτρέπουν την δημιουργία όλων των παραπάνω ειδών περιεχομένου που αφορούν έναν διάλογο ειδοποίησης.

4.9.2 Progress Dialog

Χρησιμοποιείται κυρίως για να απεικονίσει την πρόοδο κάποιας ενέργειας συνήθως είτε με την μορφή περιστρεφόμενου τροχού είτε με τη μορφή μπάρας. Μπορεί επίσης να παρέχει κουμπιά, όπως κουμπί για την ακύρωση της λήψης.

4.10 CardView και RecyclerView

Η Google προκειμένου να βελτιώσει το γραφικό περιβάλλον του λειτουργικού συστήματος δημιούργησε ένα Framework που δίνει τη δυνατότητα στους προγραμματιστές να δημιουργούν περίπλοκες λίστες. Αυτό επιτεύχθηκε μέσω του RecyclerView σε συνεργασία με το CardView.

[21] https://guides.codepath.com/android/Understanding-App-Permissions

4.10.1 CardView

Το CardView προστέθηκε μαζί με το Material Design στην έκδοση Android Lollipop 5.0. Αφορά γραφικό στοιχείο που μας επιτρέπει να ελέγχουμε το χρώμα φόντου, τη σκιά και άλλα γραφικά. Παρακάτω αναφέρονται δύο παραδείγματα χρήσης του.

- card_view:cardCornerRadius : Εμφανίζει οποιοδήποτε στοιχείο με μια διάταξη στρογγυλεμένης γωνίας και μπορεί να απεικονίζει προβολές την μία πάνω στην άλλη. Σκοπός είναι να δώσει μια πιο πλούσια εμφάνιση στη διεπαφή χρήστη.
- card_view:cardBackgroundColor : Χρησιμοποιείται για τον ορισμό του χρώματος φόντου της προβολής.

```
<androidx.cardview.widget.CardView
    android:id="@+id/backButton"
    style="@style/ButtonColor"
    android:layout_gravity="center"
    android:layout_marginTop="50dp"
    android:layout_marginBottom="20dp"
    android:clickable="true"
    app:layout_constraintBottom_toBottomOf="parent"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintStart_toStartOf="parent">

    <TextView
        android:id="@+id/backButtonText"
        style="@style/TextColorWhite.button"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Επιστροφή στην Αρχική" />
</androidx.cardview.widget.CardView>
```

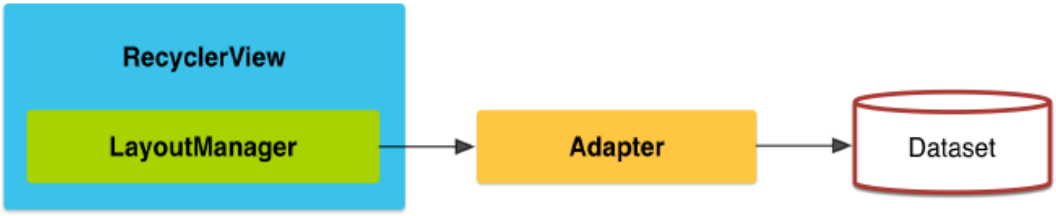
Εικόνα 24 CardView

4.10.2 RecyclerView

Το RecyclerView είναι μια πιο εκτεταμένη έκδοση του ListView και του GridView και προσθέτει πολλές δυνατότητες στις λίστες των δεδομένων, όπως ομαλή κύλιση τους και δυνατότητα πολλών επιλογών σε animations και σκιές. Επίσης, ένα σημαντικό πλεονέκτημα είναι ότι επιτυγχάνεται η ανακύκλωση προβολών που δεν είναι ορατές στην οθόνη του χρήστη. Για παράδειγμα, εάν ο χρήστης έκανε κύλιση μιας λίστας προς τα κάτω όπου τα στοιχεία 3 και 4 είναι ορατά, τότε τα στοιχεία 1 και 2 θα διαγραφούν από την μνήμη.

```
<androidx.recyclerview.widget.RecyclerView
    android:id="@+id/recyclerView_sections"
    android:layout_width="0dp"
    android:layout_height="match_parent"
    android:layout_marginTop="30dp"
    android:layout_marginBottom="5dp"
    app:layoutManager="androidx.recyclerview.widget.LinearLayoutManager"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toTopOf="parent"
    tools:listitem="@layout/design_sections" />
```

Εικόνα 25 Παράδειγμα κώδικα



Εικόνα 26 Recycle View[22]

Μοντέλο ενός RecyclerView

Για τη χρήση ενός RecyclerView απαιτούνται:

- RecyclerView: Αναλαμβάνει τη διαχείριση των δεδομένων ώστε να τα εμφανίσει στη λίστα.
- Layout Manager: Λειτουργεί υποστηρικτικά στη τοποθέτηση των δεδομένων.
- ItemAnimator: Παρέχει τα animations για διάφορες λειτουργίες όπως είναι η προσθήκη και η αφαίρεση ενός στοιχείου.

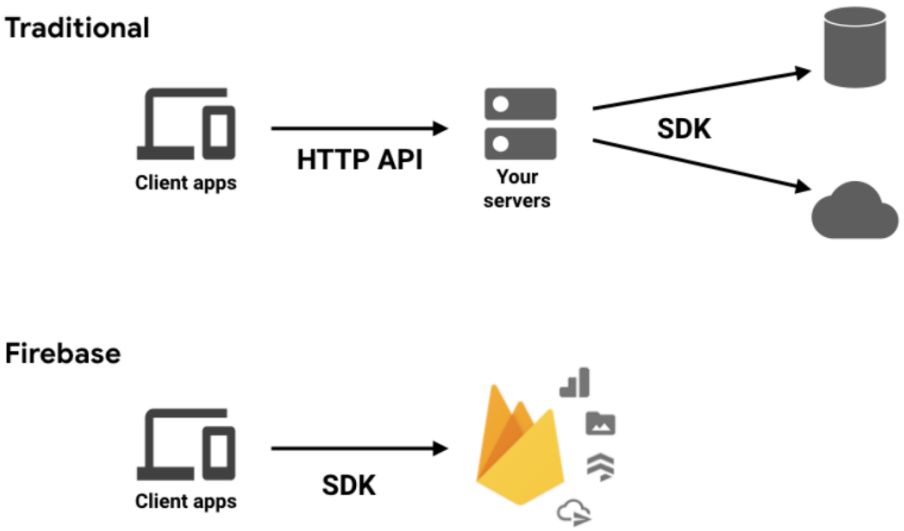
4.11 Google Firebase

Είναι μια cloud υπηρεσία που περιλαμβάνει ένα σύνολο εργαλείων (Software Development Kit) για την ανάπτυξη εφαρμογών. Οι προγραμματιστές χωρίς αυτά τα εργαλεία, θα έπρεπε να τα αναπτύξουν οι ίδιοι δαπανώντας χρόνο και κόστος. Τα εργαλεία αυτά (SDKs) αναπτύσσονται και συντηρούνται εξ' ολοκλήρου από την Google και φιλοξενούνται στο cloud. Παρέχονται από την Firebase και αλληλεπιδρούν απευθείας με τις υπηρεσίες (Services), χωρίς να χρειάζεται να δημιουργηθεί μια ενδιάμεση εφαρμογή μεταξύ της εφαρμογής και της υπηρεσίας.

[22] https://silsly.medium.com/sqlite-with-recycler-view-845cf67e0e8e

Ενώ με την παραδοσιακή ανάπτυξη εφαρμογών, οι προγραμματιστές θα έπρεπε να αναπτύξουν τόσο το front-end όσο και το back-end, τώρα με την αξιοποίηση των εργαλείων που προσφέρει η Firebase, το βάρος ανάπτυξης για τους προγραμματιστές μεταφέρεται από το back-end στο front-end. Η Firebase ενσωματώνεται πλήρως στο λειτουργικό σύστημα Android και έχει καταστεί μια πλήρης σουίτα ανάπτυξης εφαρμογών. Το αποτέλεσμα είναι ότι πλέον οι προγραμματιστές, αφού δεν χρειάζεται να αφιερώνουν χρόνο στην ανάπτυξη του back-end, εστιάζονται στην βελτίωση της διεπαφής του χρήστη (User Interface) εξοικονομώντας χρόνο και κόστος.

Στην παρακάτω εικόνα παρατηρούμε ότι με την παραδοσιακή ανάπτυξη εφαρμογών συμπεριλαμβάνεται η σύνταξη κώδικα τόσο στο front-end όσο και στο back-end. Ο κώδικας στο front-end καλεί τα APIs που εκτίθενται από το back-end με αποτέλεσμα την περισσότερη εργασία να την κάνει το back-end. Μέσω της Firebase ο προγραμματιστής έχει στη διάθεσή του όλα τα εργαλεία που θα αξιοποιήσει για την ανάπτυξη της διεπαφής χρήστη.



Εικόνα 27 Firebase[23]

4.12 Realtime Database

Η Firebase Realtime Database είναι μια μη σχεσιακή βάση (NoSQL) δεδομένων που φιλοξενείται στο cloud και μπορεί να υποστηρίξει εφαρμογές που είναι γραμμένες για Android, iOS, JavaScript, Java, Objective-C, C++ και Unity. Στην παρούσα εφαρμογή έχει γίνει χρήση της Realtime Database.

[23] https://medium.com/firebase-developers/what-is-firebase-the-complete-story-abridged-bcc730c5f2c0

Με την Realtime Database, μπορούν οι προγραμματιστές να κάνουν αλλαγές στη βάση δεδομένων και αυτόματα, σε πραγματικό χρόνο, θα ενημερωθούν και όλες οι συσκευές. Επιπλέον, όταν η συσκευή δεν είναι συνδεδεμένη στο διαδίκτυο, η εφαρμογή χρησιμοποιεί τα δεδομένα από τη μνήμη, επιτρέποντας την ομαλή λειτουργία στην εφαρμογή, και, μόλις συνδεθεί στο διαδίκτυο, η Realtime Database συγχρονίζει τις αλλαγές των δεδομένων που έγιναν τοπικά στη συσκευή με τις τυχόν απομακρυσμένες ενημερώσεις που πραγματοποιήθηκαν ενώ η συσκευή βρισκόταν εκτός σύνδεσης.

Όπως γίνεται κατανοητό, οι συσκευές έχουν άμεση πρόσβαση στη βάση δεδομένων, χωρίς να απαιτείται application server. Οι προγραμματιστές, μέσα από κανόνες ασφαλείας, μπορούν να καθορίσουν ποιος θα έχει πρόσβαση και σε ποια δεδομένα και πώς μπορεί να έχει πρόσβαση σε αυτά.

Τα δεδομένα στην Realtime Database αποθηκεύονται σε μορφή JSON. Η βάση δεδομένων μπορεί να αποτελέσει ένα μεγάλο δέντρο JSON, σε αντίθεση με την παραδοσιακή δομή μιας σχεσιακής βάσης δεδομένων (RDMS).

```
{
  "rules": {
    ".read": true,
    ".write": true
  }
}
```

Εικόνα 28 Realtime Database rules

5. Ανάλυση των απαιτήσεων της εφαρμογής

Στο παρόν κεφάλαιο θα γίνει αναφορά στην ανάλυση απαιτήσεων και στον σχεδιασμό της βάσης δεδομένων.

5.1 Ανάλυση Απαιτήσεων

Στην παρούσα ενότητα γίνεται συνοπτική ανάλυση των απαιτήσεων της εφαρμογής. Ο χρήστης μπορεί να κάνει είσοδο στην εφαρμογή, να επιλέξει κάποια προσωπικά στοιχεία του, τη θεματολογία των ερωτήσεων που επιθυμεί να απαντήσει, να ελέγξει τα αποτελέσματά του καθώς και να τα κοινοποιήσει σε εφαρμογές κοινωνικής δικτύωσης ή να τα αποστείλει μέσω mail, καθώς επίσης να παρακολουθήσει κάποια στατιστικά. Παρακάτω παρατίθενται οι απαιτήσεις της εφαρμογής.

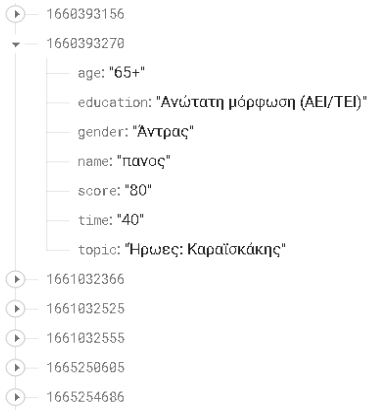
- Το γραφικό περιβάλλον έχει γίνει όσο γίνεται πιο απλό και εύχρηστο για τον χρήστη.
- Αξιοποιεί τις δυνατότητες των εργαλείων που παρέχει η cloud βάση της Firebase.
- Μπορεί να υποστηρίξει συσκευές με διαφορετικά τεχνικά χαρακτηριστικά, διαφορετικές εκδόσεις λογισμικού ή ακόμη και διαφορετικό μέγεθος οθόνης.
- Χρήση των σύγχρονων βιβλιοθηκών και τεχνολογιών για βέλτιστη λειτουργία της εφαρμογής.
- Δυνατότητα κοινοποίησης.
- Τα δεδομένα να είναι ασφαλή.
- Ανάλυση δεδομένων που αφορούν προσωπικά στοιχεία των χρηστών.
- Χρήση της εφαρμογής χωρίς να είναι απαραίτητο η συσκευή να είναι συνδεδεμένη στο διαδίκτυο.
- Δυνατότητα αναζήτησης εντός της εφαρμογής.
- Material Components της Google.
- Κατάλληλο optimization για τα δεδομένα και τη χρήση του υλικού.

Για την ανάπτυξη της εφαρμογής έχει χρησιμοποιηθεί τοπική βάση δεδομένων, ιδιαίτερα για τις ερωτήσεις γνώσης της εφαρμογής. Με αυτόν τον τρόπο γίνεται εφικτή η λειτουργία της εφαρμογής χωρίς τη χρήση του Internet. Ακόμη, αξιοποιούνται τα εργαλεία που προσφέρει η Firebase της Google, ιδιαίτερα στην αποθήκευση δεδομένων για τα στατιστικά στοιχεία που συγκεντρώνει η εφαρμογή αναφορικά με τους χρήστες. Επίσης, γίνεται εκτεταμένη χρήση του RecyclerView και των Material Components της Google.

5.2 Σχεδιασμός της Βάσης Δεδομένων

Για την αποθήκευση των στατιστικών της εφαρμογής έγινε αξιοποίηση των δυνατοτήτων που προσφέρει η Realtime Database της Firebase, παρέχοντας με αυτόν τον τρόπο την δυνατότητα να ενημερώνονται άμεσα όλες οι συσκευές για οποιαδήποτε αλλαγή στη βάση δεδομένων με ασφάλεια. Σε κάθε είσοδο χρήστη, δίνεται ένα μοναδικό id. Κάτω από αυτό το μοναδικό id έχει χτιστεί ένα δέντρο με όλες τις απαραίτητες πληροφορίες για τα στατιστικά. Οι πληροφορίες που αποθηκεύονται στην βάση δεδομένων είναι:

- Ηλικία
- Εκπαίδευση
- Γένος
- Επίδοση (score)
- Χρόνος ολοκλήρωσης των ερωτήσεων
- Κατηγορία ερωτήσεων που επέλεξε ο χρήστης



Εικόνα 29 Παράδειγμα αποθήκευσης δεδομένων στη Firebase

6. Παρουσίαση της εφαρμογής

Σε αυτό το κεφάλαιο γίνεται παρουσίαση της λειτουργικότητας και των δυνατοτήτων της εφαρμογής. Για την εύκολη κατανόηση θα γίνει χρήση βοηθητικών στιγμιότυπων (screenshots). Η παρουσίαση αυτή μπορεί να χρησιμοποιηθεί και ως εγχειρίδιο χρήστη (user manual).

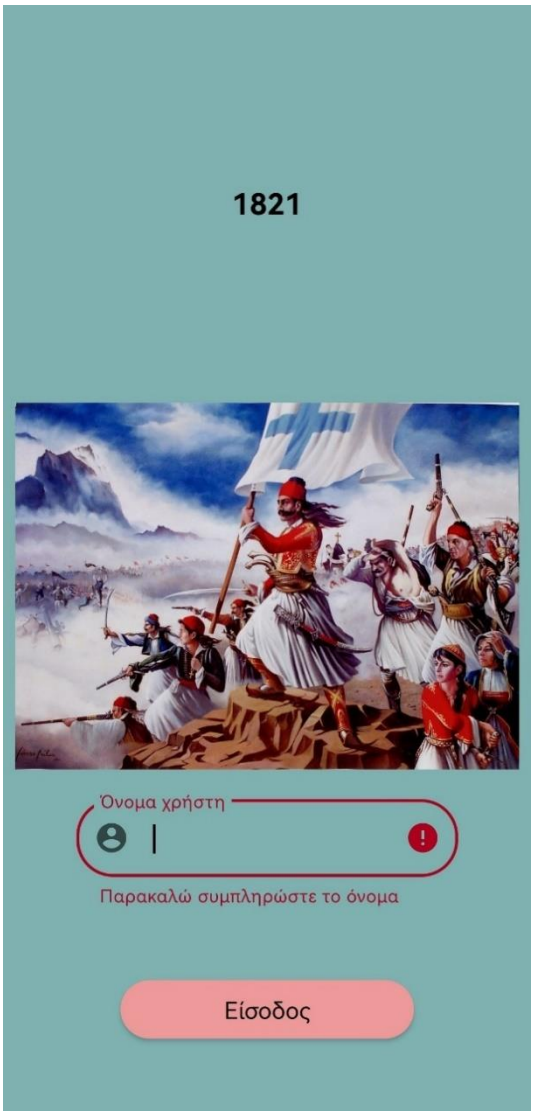
Στόχος της μεταπτυχιακής διατριβής είναι να προσφέρει μια εφαρμογή που θα βελτιώσει την εμπειρία του χρήστη σε ψυχαγωγικού τύπου παιχνίδι ερωτήσεων γνώσης. Μια εύχρηστη εφαρμογή που δίνει την δυνατότητα στον χρήστη να την επισκεφθεί χωρίς να χρειάζεται να δημιουργήσει λογαριασμό. Μπορεί να απαντήσει σε ερωτήσεις πολλαπλών επιλογών (“multiple choice”), να αναζητήσει την θεματολογία των ερωτήσεων που επιθυμεί, να κοινοποιήσει τις απαντήσεις και τα αποτελέσματα των απαντήσεών του καθώς και να παρακολουθήσει στατιστικά στοιχεία.

Όλα τα παραπάνω υλοποιούνται σε μια εφαρμογή σχεδιασμένη με όσο το δυνατόν λιγότερη πολυπλοκότητα με στόχο να καταστεί φιλική, ευχάριστη και εύχρηστη για τον απλό χρήστη.

6.1 Είσοδος της εφαρμογής

Κατά την έναρξη της εφαρμογής από τον χρήστη, εμφανίζεται η οθόνη της εισόδου. Αποτελείται από το λογότυπο της εφαρμογής ένα πεδίο εισαγωγής ονόματος χρήστη, το κουμπί εισόδου και το κουμπί που αναφέρει στατιστικά δεδομένα. Το πεδίο συμπλήρωσης του ονόματος χρήστη είναι υποχρεωτικό και μόνο εάν συμπληρωθεί ο χρήστης μεταφέρεται στην επόμενη σελίδα. Στην περίπτωση που δεν συμπληρωθεί εμφανίζεται το σχετικό μήνυμα λάθους.

Εικόνα 30 Αρχική διεπαφή εφαρμογής



Εικόνα 31 Αρχική διεπαφή εφαρμογής

6.2 Σελίδα εισαγωγής στοιχείων

Η σελίδα εισαγωγής στοιχείων αποτελείται από τρεις κατηγορίες, τις οποίες ο χρήστης καλείται να απαντήσει. Αφορούν την επιλογή του φύλου του(άντρας/γυναίκα), την ηλικιακή του ομάδα καθώς και το μορφωτικό επίπεδο που έχει. Οι κατηγορίες αυτές θα αποτελέσουν και την πηγή πληροφορίας για τα στατιστικά δεδομένα της εφαρμογής. Στο πάνω μέρος της σελίδας παρουσιάζεται ένα μήνυμα, το οποίο αναφέρει την έκφραση «Καλώς ήλθες» και προσθέτει ακριβώς δίπλα από την έκφραση αυτή, το όνομα που έχει δηλώσει ο χρήστης. Οι ηλικιακές χωρίζονται σε εκείνους που είναι κάτω από 18 ετών, σε εκείνους που είναι μεταξύ 18 και 24, 25 με 35, 36 με 50, 51 με 65 και τους άνω των 65 ετών. Το μορφωτικό επίπεδο χωρίζεται σε αποφοίτους δημοτικού, αποφοίτους γυμνασίου, αποφοίτους λυκείου, σε εκείνους που έχουν ανώτατη μόρφωση (ΑΕΙ/ΤΕΙ) και τέλος σε εκείνους που έχουν μεταπτυχιακές σπουδές ή διδακτορικό.

Η επιλογή και των τριών κατηγοριών είναι υποχρεωτική και ο χρήστης χρειάζεται να κάνει κύλιση προς τα δεξιά για να τις δει όλες. Όταν επιλέγει ο χρήστης κάποια επιλογή, εμφανίζεται με πιο σκούρο χρώμα. Στην περίπτωση που ο χρήστης δεν επιλέξει κάποια κατηγορία, εμφανίζεται σχετικό μήνυμα λάθους και είναι αδύνατο να μεταφερθεί στην επόμενη σελίδα. Στο κάτω μέρος της οθόνης υπάρχει κουμπί «Συνέχεια» το οποίο μεταφέρει τον χρήστη στο κεντρικό μενού.

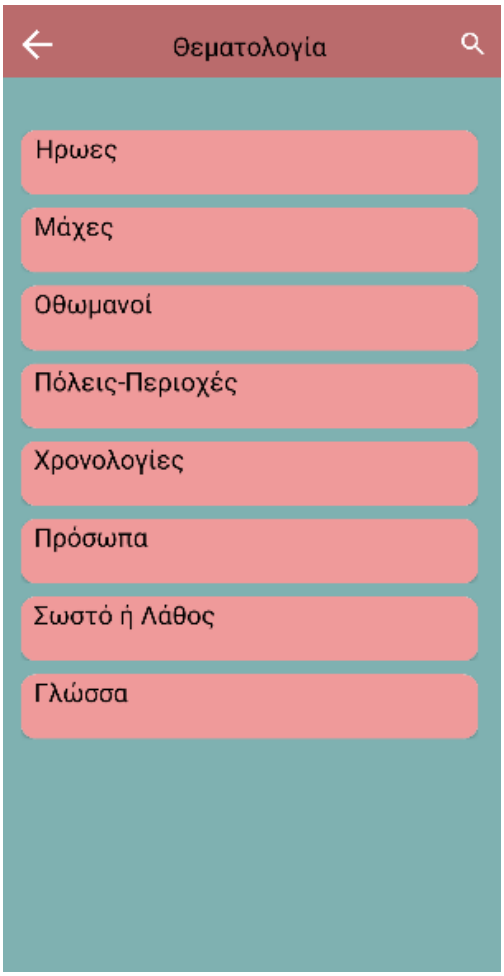
Εικόνα 32 Οθόνη επιλογής κατηγοριών

Εικόνα 33 Οθόνη επιλογής κατηγοριών

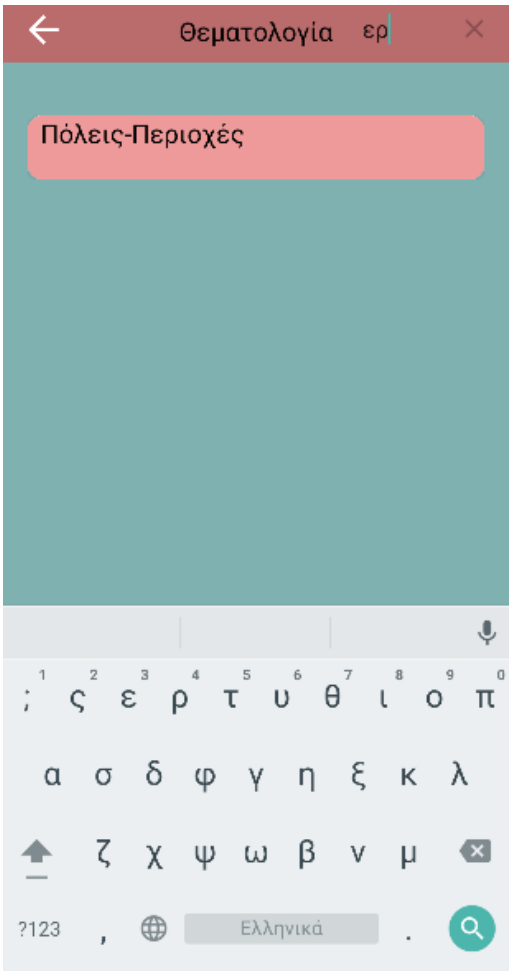
6.3 Κεντρικό μενού θεματολογίας ερωτήσεων

Αποτελεί την κεντρική σελίδα της εφαρμογής. Ο χρήστης καλείται να επιλέξει μια από τις θεματολογίες της αρεσκείας του. Οι κατηγορίες έχουν επιλεγεί με τέτοιο τρόπο ώστε να κεντρίζουν το ενδιαφέρον του χρήστη. Στο επάνω μέρος αναφέρεται ως header ο τίτλος «θεματολογία» και στο επάνω και αριστερό μέρος ο χρήστης έχει την δυνατότητα να επιστρέψει στην προηγούμενη οθόνη. Στο επάνω δεξιό μέρος της οθόνης εμφανίζεται το σύμβολο του μεγεθυντικού φακού, το οποίο πατώντας το ο χρήστης μπορεί να κάνει γρήγορη αναζήτηση μιας θεματολογίας βάσει του τίτλου της. Ο χρήστης μπορεί να αναζητήσει είτε με βάση ολόκληρο τον τίτλο είτε να γράψει ένα μέρος αυτού. Καθώς γράφει ο χρήστης στην αναζήτηση, αυτόματα εμφανίζονται οι πλησιέστερες αναζητήσεις βάσει αυτού που έχει γράψει. Ο χρήστης όταν επιλέξει μια

κατηγορία, τότε αυτόματα μεταφέρεται στην επόμενη σελίδα χωρίς να χρειάζεται να πατήσει κάποιο κουμπί που θα τον μεταφέρει.



Εικόνα 34 Κεντρικό Μενού



Εικόνα 35 Κεντρικό Μενού

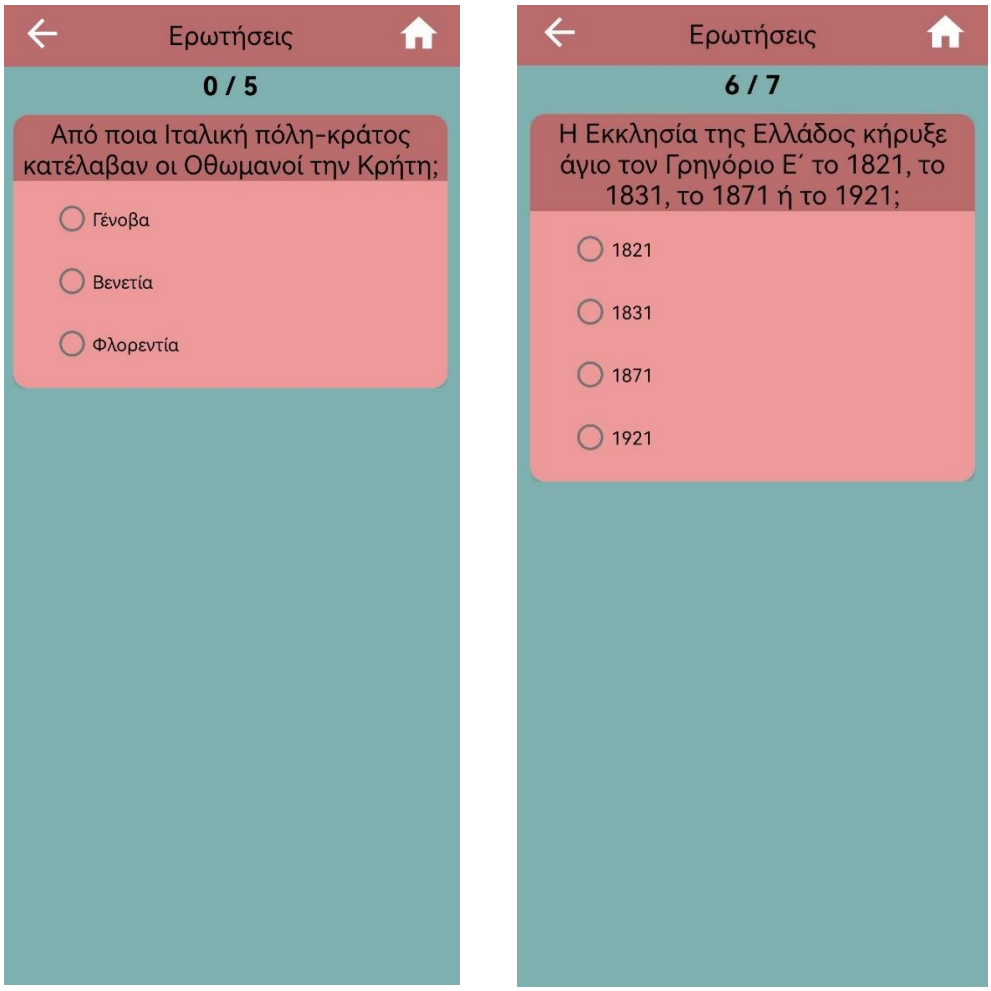
6.3.1 Κατηγορίες

Κατά τον ίδιο τρόπο με τις θεματολογίες, ο χρήστης καλείται να επιλέξει μια κατηγορία ερωτήσεων βάσει της θεματολογίας που επέλεξε προηγουμένως. Όπως ακριβώς και στην επιλογή της θεματολογίας, έτσι και στην επιλογή της κατηγορίας υπάρχει λειτουργία αναζήτησης και επιλογή επιστροφής στην προηγούμενη σελίδα. Μόλις επιλέξει ο χρήστης την κατηγορία που επιθυμεί, μεταφέρεται αυτόματα στην επόμενη σελίδα.

6.4 Σελίδα ερωτήσεων

Αφού ο χρήστης επέλεξε κατηγορία ερωτήσεων, μεταφέρεται αυτόματα στο παιχνίδι ερωτήσεων. Εδώ, ο χρήστης καλείται να απαντήσει ανάμεσα σε απαντήσεις πολλαπλής επιλογής (“multiple choice”). Στο header της οθόνης ο χρήστης βλέπει σαν τίτλο το όνομα της σελίδας («Ερωτήσεις»). Στο επάνω αριστερό μέρος υπάρχει το κουμπί της

επιστροφής στην προηγούμενη σελίδα και στο επάνω δεξιό μέρος εμφανίζεται ένα εικονίδιο στο σχήμα σπιτιού, όπου ο χρήστης όταν το πατήσει μεταφέρεται στην αρχική οθόνη. Πάνω ακριβώς από την κάθε ερώτηση εμφανίζεται ο συνολικός αριθμός ερωτήσεων στις οποίες θα χρειαστεί να απαντήσει, καθώς και η πληροφορία σε ποια ερώτηση βρίσκεται σε σχέση με τον συνολικό αριθμό. Η κάθε ερώτηση εμφανίζεται ξεχωριστά και στο κάτω μέρος της κάθε ερώτησης εμφανίζονται οι απαντήσεις σε μορφή “radio buttons”. Όταν επιλέξει την κατάλληλη απάντηση, μεταφέρεται αυτόματα στην επόμενη ερώτηση. Ο αριθμός των επιλογών των ερωτήσεων κυμαίνεται και δεν είναι σταθερός.

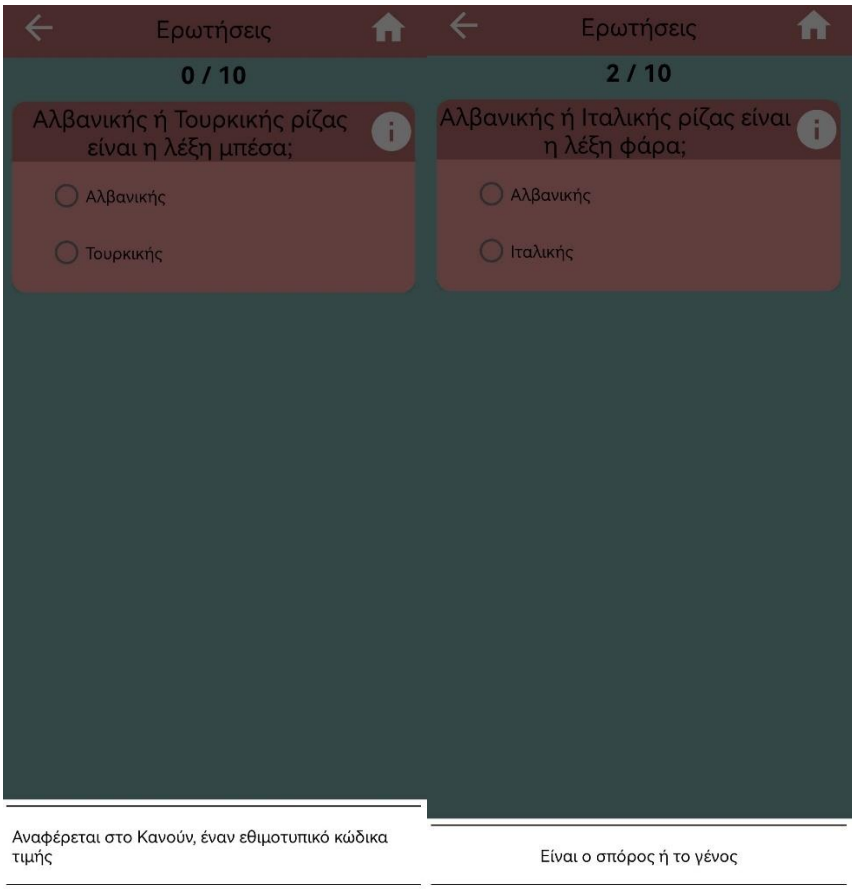


Εικόνα 36 Παραδείγματα οθόνης ερωτήσεων εφαρμογής

6.5 Hints

Σε αρκετές ερωτήσεις και ακριβώς δεξιά από την τοποθέτηση της ερώτησης, υπάρχει ένα κουμπί με το σύμβολο “i” το οποίο μπορεί όταν το πατήσει ο χρήστης, αναδύεται ένα μήνυμα-hint στο κάτω μέρος της οθόνης, το οποίο υπαινίσσεται κάποιο στοιχείο το οποίο μπορεί να οδηγήσει τον χρήστη στην επιλογή ορθής απάντησης σχετικά με την ερώτηση που του τίθεται. Όταν πατήσει ο χρήστης το hint τότε φαίνεται πιο σκούρο

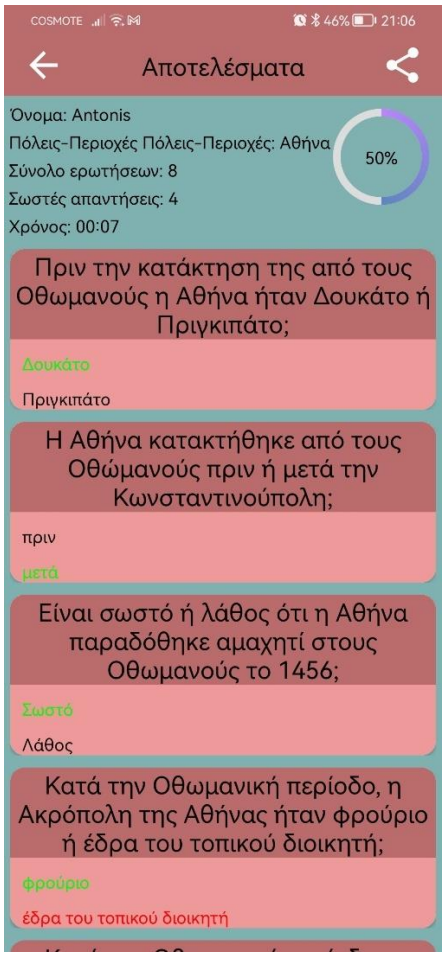
το υπόλοιπο μέρος της οθόνης και φαίνεται «καθαρά» μόνο το hint. Παρακάτω ακολουθεί στιγμιότυπο ενός hint από μια ερώτηση της εφαρμογής.



Εικόνα 37 Παράδειγμα Hint

6.6 Οθόνη αποτελεσμάτων

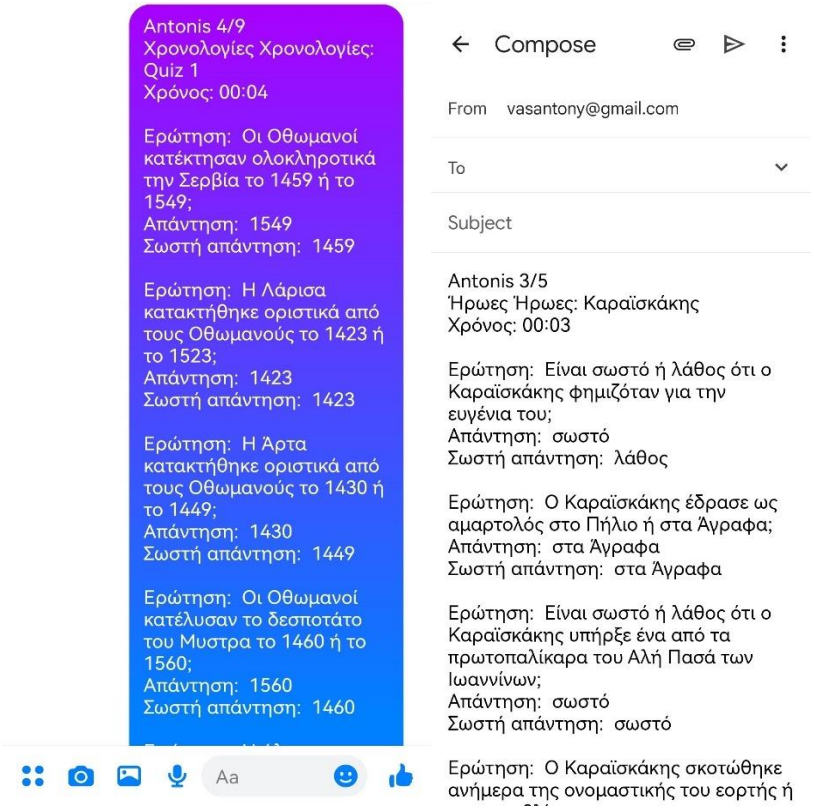
Όταν ο χρήστης απαντήσει σε όλες τις ερωτήσεις, αυτόματα μεταφέρεται στην οθόνη των αποτελεσμάτων. Στο επάνω μέρος εμφανίζεται ως header το σχετικό μήνυμα «Αποτελέσματα» ως πληροφορία για τον χρήστη. Όπως ακριβώς και στις υπόλοιπες οθόνες, ο χρήστης έχει την δυνατότητα επιστροφής στην προηγούμενη οθόνη μέσω του κουμπιού επιστροφής. Στο πάνω μέρος της οθόνης εμφανίζονται πληροφορίες σχετικά με το όνομα του χρήστη, ποια θεματολογία και κατηγορία έχει επιλέξει, το σύνολο των ερωτήσεων της συγκεκριμένης κατηγορίας, τον αριθμό των σωστών απαντήσεων και τον χρόνο ολοκλήρωσης των απαντήσεων. Ακριβώς δεξιά από αυτές τις πληροφορίες, εμφανίζεται σε μορφή «πίτας» το ποσοστό που «έπιασε» ο χρήστης. Τέλος, εμφανίζονται αναλυτικά όλες οι ερωτήσεις με τις απαντήσεις που έδωσε ο χρήστης. Οι απαντήσεις που έδωσε ο χρήστης είναι με πράσινο χρώμα γραμματοσειράς. Στην περίπτωση που ο χρήστης έχει δώσει λάθος απάντηση, τότε θα εμφανιστεί η σωστή απάντηση με κόκκινο χρώμα. Οι απαντήσεις εμφανίζονται σε RecyclerView και ο χρήστης μπορεί να κάνει κύλιση (scroll) για να τις δει όλες.



Εικόνα 38 Παράδειγμα οθόνης αποτελεσμάτων

6.7 Δυνατότητα κοινοποίησης

Η εφαρμογή δίνει το δικαίωμα στον χρήστη να κάνει κοινοποίηση των αποτελεσμάτων του. Στην οθόνη των αποτελεσμάτων που έγινε αναφορά παραπάνω, στο πάνω δεξιό μέρος, εμφανίζεται το εικονίδιο της κοινοποίησης. Μόλις το πατήσει ο χρήστης, έχει την δυνατότητα να μοιραστεί τα αποτελέσματά του στα διάφορα μέσα κοινωνικής δικτύωσης (Social Networks) ή να τα αποστείλει μέσω email. Στις πληροφορίες που κοινοποιούνται εμφανίζονται σε μορφή κειμένου όλες οι πληροφορίες που φαίνονται και στην οθόνη αποτελεσμάτων του χρήστη, όπως το όνομα χρήστη, πόσες σωστές απαντήσεις έπιασε ο χρήστης, η θεματολογία και η κατηγορία που έχει επιλέξει ο χρήστης καθώς και ο χρόνος ολοκλήρωσης των απαντήσεων. Πιο κάτω, αναφέρονται αναλυτικά οι ερωτήσεις καθώς και ποια απάντηση έδωσε ο χρήστης και αν ήταν σωστή ή λανθασμένη. Στο παρακάτω screenshot φαίνεται η μορφή της αποστολής μέσω Facebook messenger και Gmail.

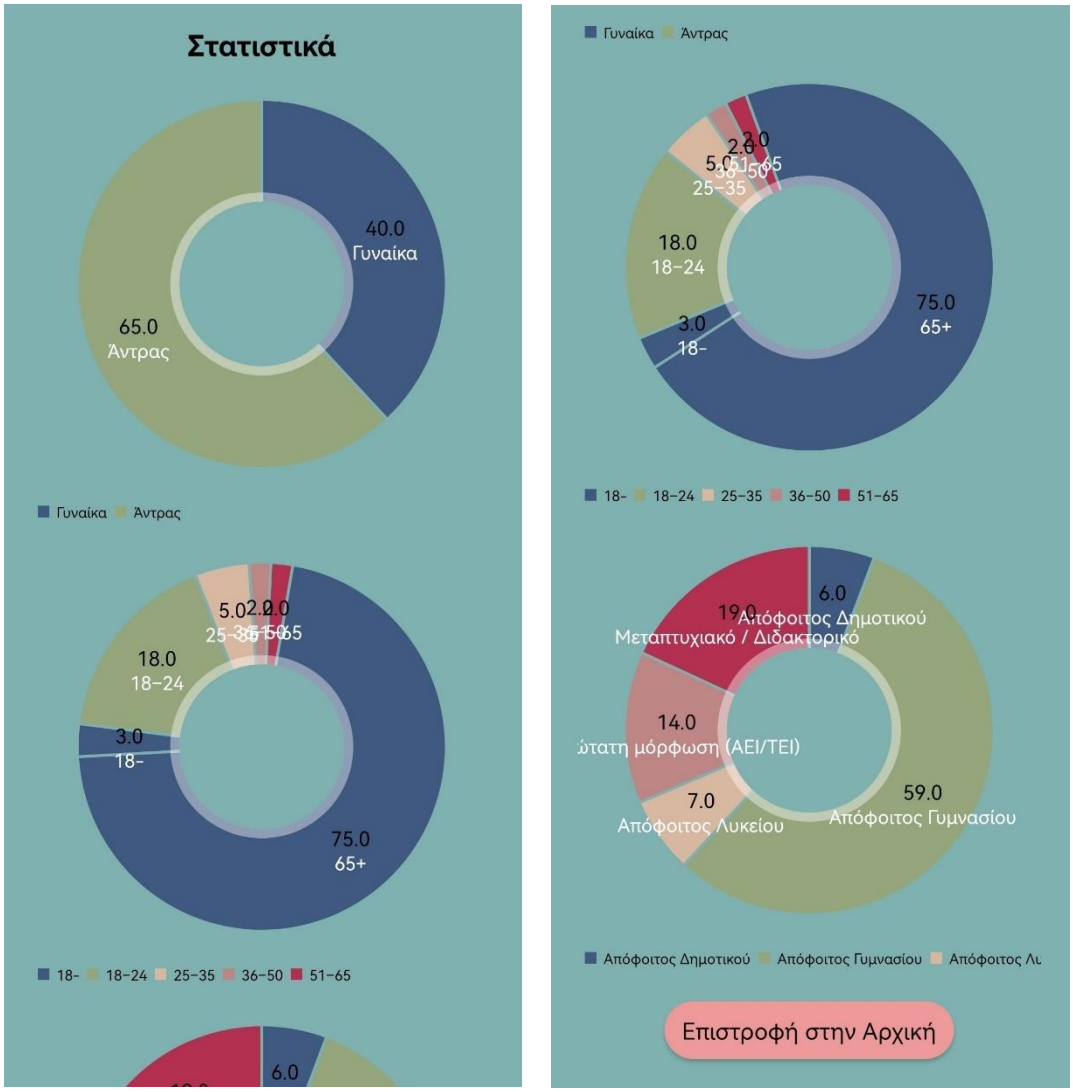


Εικόνα 39 Παραδείγματα κοινοποίησης μέσω Facebook Messenger και μέσω Google Gmail

6.8 Οθόνη στατιστικών

Στην οθόνη εισόδου της εφαρμογής υπάρχει το κουμπί των στατιστικών. Η οθόνη των στατιστικών είναι μια ανάλυση που γίνεται στο φύλο, στην ηλικία και στο μορφωτικό επίπεδο των χρηστών. Τα στοιχεία αυτά μπορούν όχι μόνο να πληροφορήσουν τον απλό χρήστη αλλά και να αξιοποιηθούν από την ίδια την εφαρμογή για τις ανάγκες βελτίωσης. Τα στατιστικά των τριών κατηγοριών απεικονίζονται σε dashboards. Τα στοιχεία καταγράφονται σε μη σχεσιακή βάση δεδομένων “Firebase”, στιγμιότυπα της οποίας θα παρουσιαστούν παρακάτω κατά την ανάλυση του κώδικα. Στο κάτω μέρος της οθόνης μπορεί να πατήσει ο χρήστης το αντίστοιχο κουμπί για να επιστρέψει στην αρχική οθόνη εισόδου.

Πιο αναλυτικά, στο ακόλουθο παράδειγμα με στιγμιότυπα της εφαρμογής έχουμε 65 άνδρες και 40 γυναίκες που έχουν χρησιμοποιήσει την εφαρμογή. Οι ηλικίες αυτών είναι πάνω από 65 ετών για τους 75 χρήστες, 18 για τους χρήστες ηλικίας από 18 έως 24, 3 χρήστες για τους κάτω των 18, 5 χρήστες για ηλικίες μεταξύ 25 και 35 ετών, 2 χρήστες για ηλικίες 36 με 50 ετών και τέλος 2 χρήστες για ηλικίες 51 με 65 ετών. Αναφορικά με την εκπαίδευση τους, οι 6 είναι απόφοιτοι Δημοτικού, οι 59 είναι απόφοιτοι Γυμνασίου, οι 7 είναι απόφοιτοι Λυκείου, οι 14 έχουν ανώτατη μόρφωση (ΑΕΙ/ΤΕΙ) και οι 19 έχουν μεταπτυχιακό/Διδακτορικό.



Εικόνα 40 Οθόνη στατιστικών

7. Επεξήγηση κώδικα

Στον παρόν κεφάλαιο θα γίνει αναφορά στα πιο σημαντικά σημεία του κώδικα της εφαρμογής.

7.1 Dark Mode

Οι σύγχρονες εφαρμογές είναι σύνηθες να έχουν δύο διαφορετικά θέματα-mode εμφάνισης, ενός κανονικού θέματος και ενός σκοτεινού. Το σκοτεινό προτιμάται από τους χρήστες για μικρότερη ταλαιπωρία της όρασης. Στην εφαρμογή δηλώνονται τα δύο διαφορετικά θέματα: Το κανονικό θέμα στο αρχείο themes.xml και το σκοτεινό θέμα (dark mode) στο αρχείο themes.xml (night). Η δήλωση των χρωμάτων σαν styleable resources γίνεται στο αρχείο attrs.xml. Ακολουθούν στιγμιότυπα από τα παραπάνω αρχεία.

```xml
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <declare-styleable name="ds">
        <attr name="BackgroundColor" format="color" />
        <attr name="BackgroundColorElement" format="color" />
        <attr name="TextColor" format="color" />
        <attr name="VectorColor" format="color" />
        <attr name="ButtonColor" format="color" />
        <attr name="HeaderColor" format="color" />
        <attr name="ListButtonColor" format="color" />
    </declare-styleable>
</resources>
```

Εικόνα 41 Attrs.xml

night\themes.xml

```xml
<resources xmlns:tools="http://schemas.android.com/tools">
    <!-- Base application theme. -->
    <style name="Theme.FilopaideusisMVVM" parent="Theme.MaterialComponents.DayNight.NoActionBar">
        <!-- Primary brand color. -->
        <item name="colorPrimary">#fff</item>
        <item name="colorPrimaryVariant">@color/purple_700</item>
        <item name="colorOnPrimary">@color/white</item>
        <!-- Secondary brand color. -->
        <item name="colorSecondary">@color/teal_200</item>
        <item name="colorSecondaryVariant">@color/teal_700</item>
        <item name="colorOnSecondary">@color/black</item>
        <!-- Status bar color. -->
        <item name="android:statusBarColor" tools:targetApi="l">#212121</item>
        <!-- Customize your theme here. -->
        <item name="BackgroundColor">#303030</item>
        <item name="BackgroundColorElement">#a2acbd</item>
        <item name="TextColor">#fff</item>
        <item name="VectorColor">#fff</item>
        <item name="ButtonColor">#424242</item>
        <item name="ListButtonColor">#424242</item>
        <item name="HeaderColor">#212121</item>
```

Εικόνα 42 themes.xml(night)

themes.xml

```xml
<resources xmlns:tools="http://schemas.android.com/tools">
    <!-- Base application theme. -->
    <style name="Theme.FilopaideusisMVVM" parent="Theme.MaterialComponents.DayNight.NoActionBar">
        <!-- Primary brand color. -->
        <item name="colorPrimary">@color/black</item>
        <item name="colorPrimaryVariant">@color/purple_700</item>
        <item name="colorOnPrimary">@color/black</item>
        <!-- Secondary brand color. -->
        <item name="colorSecondary">@color/teal_200</item>
        <item name="colorSecondaryVariant">@color/teal_700</item>
        <item name="colorOnSecondary">@color/black</item>
        <!-- Status bar color. -->
        <item name="android:statusBarColor" tools:targetApi="l">#ba6b6c</item>
        <!-- Customize your theme here. -->
        <item name="BackgroundColor">#7FB1B1</item>
        <item name="BackgroundColorElement">#ef9a9a</item>
        <item name="TextColor">@color/black</item>
        <item name="VectorColor">@color/black</item>
        <item name="ButtonColor">#ef9a9a</item>
        <item name="ListButtonColor">#ef9a9a</item>
        <item name="HeaderColor">#ba6b6c</item>
    </style>
```

Εικόνα 43 themes.xml

7.2 SafeClickListener

Listener είναι μια διεπαφή που παρέχει μια συγκεκριμένη λειτουργικότητα για κάποιο διαδραστικό στοιχείο διεπαφής χρήστη (π.χ. ένα κουμπί). Ανιχνεύει τυχόν συμβάντα που σχετίζονται με τη διεπαφή χρήστη, όπως για παράδειγμα όταν αγγίζει ο χρήστης ένα μέρος της οθόνης να εκτελείται μια συγκεκριμένη ενέργεια. Επειδή με τον Listener δεν περιορίζεται ο αριθμός των διεπαφών του χρήστη για την ίδια ενέργεια, αυτό έχει σαν αποτέλεσμα να αποστέλλονται πολλά αιτήματα για την ίδια ενέργεια. Με τον

SafeClickListener περιορίζεται αυτό το φαινόμενο και δεν επιτρέπεται να πραγματοποιηθούν πολλά αιτήματα σε σύντομο χρονικό διάστημα.

Όπως φαίνεται και παρακάτω, ο SafeClickListener κληρονομεί τον OnClickListener, ο οποίος έχει την μέθοδο onClick (View v), η οποία καλείται όταν ένα στοιχείο (view) καλείται.

```
class SafeClickListener(
    private val defaultInterval: Int,
    private val onSafeClick: (View) -> Unit
) : View.OnClickListener {

    private var lastTimeClicked: Long = 0

    override fun onClick(v: View?) {

        if (SystemClock.elapsedRealtime() - lastTimeClicked < defaultInterval) {
            return
        }
        lastTimeClicked = SystemClock.elapsedRealtime()
        v?.let { onSafeClick(it) }
    }
}
```

Εικόνα 44 SafeClickListener

7.3 Πλοήγηση (Navigation)

Αναφέρεται στις αλληλεπιδράσεις που επιτρέπουν στους χρήστες να πλοηγούνται μεταξύ των διαφόρων θέσεων-προορισμών εντός της εφαρμογής. Για την πλοήγηση μεταξύ των προορισμών εντός της εφαρμογής γίνεται χρήση ενός πλαισίου (framework) που προσφέρεται ως δυνατότητα στους προγραμματιστές. Το framework αυτό παρέχει ένα σταθερό API όπου οι προορισμοί υλοποιούνται ως Fragments. Παρακάτω θα γίνει αναφορά στα τρία βασικά μέρη της Πλοήγησης.

7.3.1 Navigation Graph

Είναι ένας πόρος που περιέχει όλες τις πληροφορίες που σχετίζονται με την πλοήγηση σε μια κεντρική τοποθεσία. Περιλαμβάνει όλες τις οθόνες εντός της εφαρμογής στις οποίες μπορεί να πλοηγηθεί ο χρήστης, οι οποίες αποκαλούνται προορισμοί (destinations), και τις πιθανές διαδρομές που θα μπορούσε να ακολουθήσει μέσω της εφαρμογής.

7.3.2 NavHost

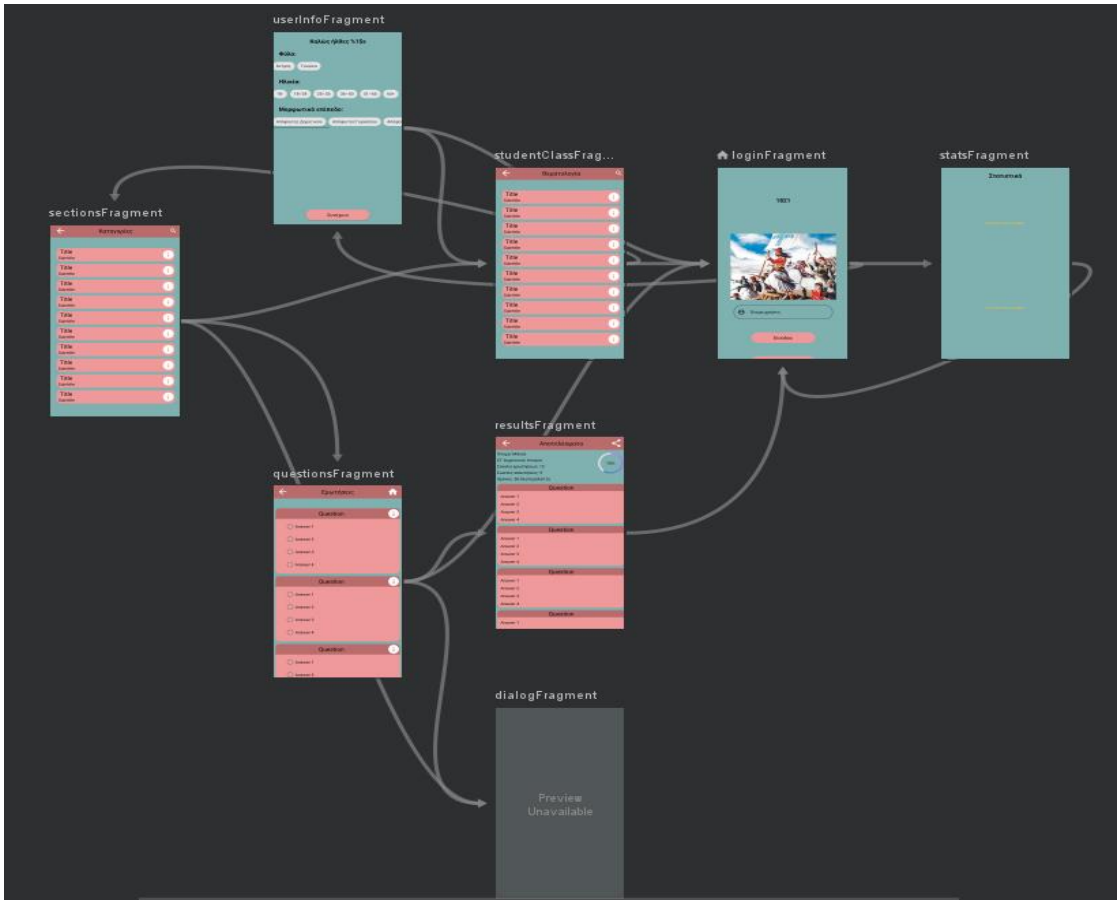
Απεικονίζει τους διαφορετικούς προορισμούς από το Navigation Graph. Η Πλοήγηση περιέχει μια προεπιλεγμένη υλοποίηση NavHost, την NavHostFragment, η οποία απεικονίζει τους προορισμούς.

7.3.3 NavController

Είναι ένα αντικείμενο που διαχειρίζεται την πλοήγηση της εφαρμογής εντός ενός NavHost. Ενορχηστρώνει την εναλλαγή του περιεχομένου του NavHost καθώς οι χρήστες πλοηγούνται στην εφαρμογή.

7.3.4 Γράφημα Πλοήγησης Εφαρμογής

Παρακάτω απεικονίζεται το γράφημα πλοήγησης (Navigation Graph) της εφαρμογής το οποίο περιλαμβάνει τους προορισμούς (destinations) καθώς και τις συνδέσεις μεταξύ των προορισμών (Actions). Ο κάθε προορισμός στο γράφημα απεικονίζεται σαν μικρογραφία και οι συνδέσεις απεικονίζονται ως βέλη που δείχνουν πώς οι χρήστες μπορούν να πλοηγηθούν από τον έναν προορισμό στον άλλο.



Εικόνα 45Γράφημα Πλοήγησης εφαρμογής

7.3.5 Ανάλυση Κώδικα Πλοήγησης

Ξεκινώντας με το στοιχείο <navigation>, καθώς προστίθενται νέοι προορισμοί με τις μεταξύ τους συνδέσεις, παρατηρούμε ότι προστίθενται αυτόματα και τα αντίστοιχα στοιχεία <destination> και <action> ως θυγατρικά στοιχεία. Στο <fragment> του κάθε θυγατρικού στοιχείου δίνεται ένα μοναδικό id. Χρησιμοποιούνται 4 παράμετροι εντός του κάθε fragment:

- **android:id:** Δίνεται ένα μοναδικό id για το κάθε fragment.
- **android:name:** Είναι το πλήρως αναγνωρισμένο όνομα της κλάσης του fragment σε γλώσσα kotlin.
- **android:label:** Είναι μια συμβολοσειρά (string) για την ταυτοποίηση του fragment.
- **tools:layout:** Ένα αναγνωριστικό αρχείου πόρων διάταξης από το res/layout.

```
<?xml version="1.0" encoding="utf-8"?>
<navigation xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/nav_graph"
    app:startDestination="@id/loginFragment">

    <fragment
        android:id="@+id/loginFragment"
        android:name="com.example.filopaideusismvvm.ui.login.LoginFragment"
        android:label="fragment_login"
        tools:layout="@layout/fragment_login">
        <action
            android:id="@+id/action_loginFragment_to_userInfoFragment"
            app:destination="@id/userInfoFragment"
            app:enterAnim="@anim/slide_in_right"
            app:exitAnim="@anim/slide_out_left"
            app:popEnterAnim="@anim/slide_in_left"
            app:popExitAnim="@anim/slide_out_right" />
        <action
            android:id="@+id/action_loginFragment_to_statsFragment"
            app:destination="@id/statsFragment"
            app:enterAnim="@anim/slide_in_right"
            app:exitAnim="@anim/slide_out_left"
            app:popEnterAnim="@anim/slide_in_left"
            app:popExitAnim="@anim/slide_out_right" />
    </fragment>
```

Εικόνα 46 Παράδειγμα κώδικα Navigation

Action

Όπως αναφέρθηκε και παραπάνω, το Navigation επιτρέπει να μετακινηθεί κανείς μέσω ενεργειών (Actions). Οι γραμμές στο γράφημα είναι γραφικές αναπαραστάσεις των ενεργειών, όπου η άκρη της κάθε γραμμής αποκαλείται προορισμός. Για να προσθέσουμε μια ενέργεια σε ένα fragment χρησιμοποιούμε το <action/> tag μέσα στο

<fragment/> tag. Μπορούμε να προσθέσουμε περισσότερο από μια ενέργεια με διαφορετικό όνομα id.

Οι παράμετροι που χρησιμοποιούνται για το <action/> είναι οι εξής:

- **android:id**: Είναι το μοναδικό id για το Action, όπως ακριβώς συμβαίνει και με τα fragments.
- **app:destination**: Είναι το μοναδικό id για το fragment που δείχνει ο προορισμός. Αυτό δηλώνει ότι από την τρέχουσα οθόνη που βρισκόμαστε, θα μετακινηθούμε στο fragment που δείχνει ο προορισμός.
- **app:enterAnim, app:exitAnim, app:popEnterAnim, app:popExitAnim:** Το Navigation επιτρέπει να προσθέσουμε animations στα Actions. Περιλαμβάνονται διάφορα προεπιλεγμένα animations τα οποία μπορούμε να καθορίσουμε όταν εισερχόμαστε σε έναν προορισμό ή όταν βγαίνουμε από έναν προορισμό. Στην εφαρμογή έχει γίνει χρήση των Animations. Όταν προχωρούμε στην επόμενη οθόνη, τότε αυτή «ξεγλιστρά» στα αριστερά, ενώ όταν πηγαίνουμε στην προηγούμενη οθόνη, τότε αυτή «ξεγλιστρά» στα δεξιά.

```
<action
    android:id="@+id/action_loginFragment_to_statsFragment"
    app:destination="@id/statsFragment"
    app:enterAnim="@anim/slide_in_right"
    app:exitAnim="@anim/slide_out_left"
    app:popEnterAnim="@anim/slide_in_left"
    app:popExitAnim="@anim/slide_out_right" />
```

Εικόνα 47 Παράδειγμα κώδικα Action

7.4 Multiple View in a RecyclerView

Μπορούμε να προσθέσουμε πολλαπλές προβολές σε ένα RecyclerView. Με αυτή την δυνατότητα μπορούμε για παράδειγμα να προσθέσουμε μια εικόνα, ένα κείμενο ή και συνδυασμό όλων αυτών στο ίδιο RecyclerView. Επομένως με την χρήση των πολλαπλών προβολών στο ίδιο RecyclerView, μπορούμε να εμφανίσουμε διαφορετικούς τύπους προβολών. Με αυτόν τον τρόπο μπορούμε να δημιουργήσουμε πιο διαδραστικές και επεκτάσιμες εφαρμογές. Παραδείγματα εκτεταμένης χρήσης αυτής της λειτουργίας παρατηρούμε σε διάφορες εφαρμογές. Μια τέτοια εφαρμογή είναι η εφαρμογή συνομιλίας “WhatApp”. Το μήνυμα που αποστέλλουμε εμείς ως χρήστες εμφανίζεται στην δεξιά πλευρά της οθόνης και το μήνυμα που λαμβάνουμε εμφανίζεται στην αριστερή πλευρά. Κατά αυτόν τον τρόπο αλλάζει ο τύπος της προβολής.

Στην παρούσα εφαρμογή γίνεται χρήση των πολλαπλών προβολών σε ένα RecyclerView. Η παρακάτω εικόνα αποτελεί παράδειγμα εφαρμογής των πολλαπλών προβολών για την περίπτωση που μια ερώτηση στο quiz αποτελείται από δύο, τρείς ή ακόμη και τέσσερις διαφορετικές απαντήσεις.

```
override fun onCreateViewHolder(parent: ViewGroup, viewType: Int): BaseQuestionViewHolder<*, *, *> {
    return when (viewType) {
        FOUR_ANSWER_RADIO -> {
            val binding =
                DesignQuestionBinding.inflate(LayoutInflater.from(parent.context), parent, attachToParent: false)
            FourQuestionsRadioViewHolder(binding)
        }
        THREE_ANSWER_RADIO -> {
            val binding =
                DesignThreeQuestionsBinding.inflate(LayoutInflater.from(parent.context), parent, attachToParent: false)
            ThreeQuestionsRadioViewHolder(binding)
        }
        TWO_ANSWER_RADIO -> {
            val binding =
                DesignTwoQuestionsBinding.inflate(LayoutInflater.from(parent.context), parent, attachToParent: false)
            TwoQuestionsRadioViewHolder(binding)
        }
        else -> throw IllegalArgumentException("Invalid view type")
    }
}
```

Εικόνα 48 Παράδειγμα κώδικα MultipleView σε RecyclerView

7.5 Αποθήκευση δεδομένων σε τοπική βάση με την χρήση του Δωματίου (Room)

Στην παρούσα εφαρμογή οι ερωτήσεις γνώσεως και οι απαντήσεις τους αποθηκεύονται σε τοπική βάση με τη χρήση του δωματίου (Room). Οι εφαρμογές μπορούν να ωφεληθούν από την αποθήκευση πολλών δεδομένων τοπικά. Σημαντικό προτέρημα της προσωρινής αποθήκευσης δεδομένων είναι ότι ακόμη και αν η συσκευή δεν έχει πρόσβαση στο διαδίκτυο, ο χρήστης να μπορεί να περιηγηθεί στο περιεχόμενο ακόμη και εάν είναι εκτός σύνδεσης. Room είναι μια βιβλιοθήκη Object Relational Mapping (ORM). Με άλλα λόγια, το Room αντιστοιχεί τα αντικείμενα της βάσης δεδομένων με τα αντικείμενα της Java. Παρέχει ένα επίπεδο αφαίρεσης (abstraction layer) πάνω από το SQLite που του επιτρέπει την εύκολη πρόσβαση στην βάση δεδομένων και φυσικά με την αξιοποίηση της πλήρους ισχύος της SQLite. Κατά αυτόν τον τρόπο γίνεται πιο εύκολη η χρήση των αντικειμένων της SQLite βάσης δεδομένων, μειώνοντας τα σφάλματα σε κώδικα boilerplate.

Ειδικότερα τα βασικά πλεονεκτήματα στη χρήση του Room είναι:

- Γρήγορη ταχύτητα μεταγλώττισης για τα SQL queries. Κάθε @Query και κάθε @Entity ελέγχεται κατά την μεταγλώττιση για να αποτρέπονται σφάλματα κατά τη διάρκεια της εκτέλεσης, λάθη στην σύνταξη και ελλείψεις στους πίνακες.
- Ευανάγνωστες παρατηρήσεις που αποτρέπουν τον επαναλαμβανόμενο και επιρρεπή σε σφάλματα κώδικα "boilerplate".
- Εύκολη ενσωμάτωση με άλλα στοιχεία αρχιτεκτονικής (όπως το LiveData).

Κύρια προβλήματα στη χρήση της SQLite χωρίς την χρήση του Room είναι:

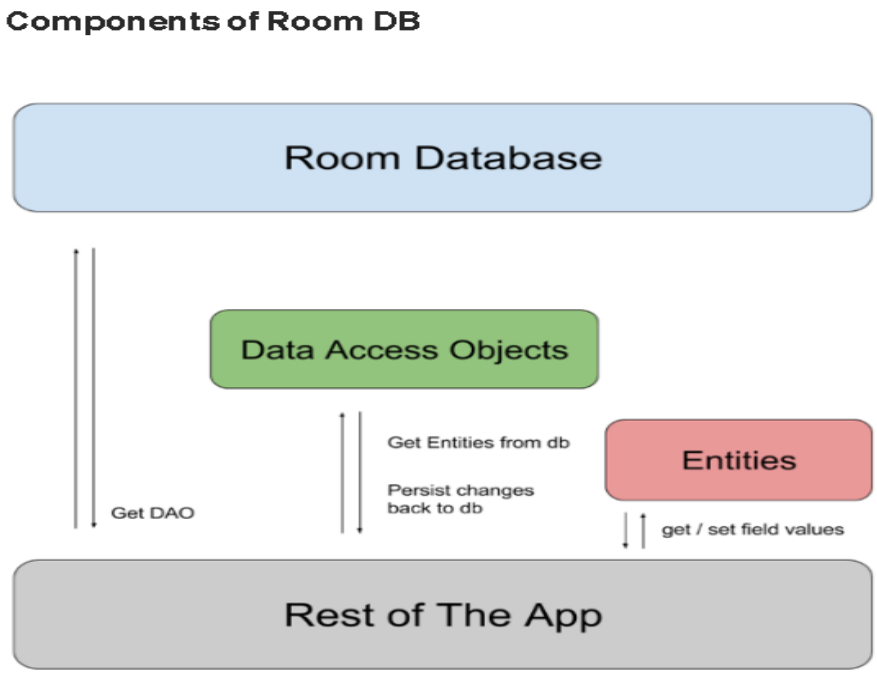
- Δεν γίνεται έλεγχος κατά την μεταγλώττιση για τις γραμμές των SQL queries. Για παράδειγμα, στην περίπτωση που υπάρχει λάθος στο όνομα μιας στήλης, η οποία δεν υπάρχει στην κανονική βάση, τότε κατά τη μεταγλώττιση θα δοθεί εξαίρεση και δεν θα επισημανθεί το σφάλμα.
- Στην περίπτωση που αλλάξει κάτι στην βάση δεδομένων, θα πρέπει να γίνουν χειροκίνητα και οι αλλαγές στα SQL queries. Αυτή η διαδικασία μπορεί να είναι χρονοβόρα και επιρρεπής σε σφάλματα.
- Εκτεταμένη χρήση κώδικα "boilerplate" για την μετατροπή μεταξύ των SQL queries και των αντικειμένων των δεδομένων Java (POJO).

Για την αποφυγή των παραπάνω, το Room βοηθά στην αντιστοίχιση των αντικειμένων της βάσης δεδομένων με τα αντικείμενα της Java, χωρίς να χρειάζεται να γίνει εκτεταμένη χρήση του κώδικα "boilerplate", σε αντίθεση με ό,τι συμβαίνει με την απευθείας χρήση της SQLite. Βασικό προτέρημα του είναι η ενσωμάτωσή του με στοιχεία αρχιτεκτονικής LiveData και RxJava, γεγονός που δεν συμβαίνει με την χρήση αυτό καθαυτό της SQLite.

7.5.1 Αρχιτεκτονική της βάσης δεδομένων του Δωματίου (Room)

3 είναι τα κύρια χαρακτηριστικά της βάσης δεδομένων του Δωματίου:

- Οντότητα (Entity)
- Dao
- Βάση δεδομένων (Database)



Εικόνα 49 Components of Room DB[24]

Entity

Αντιπροσωπεύει τον πίνακα (table) στη βάση δεδομένων. Το Room δημιουργεί έναν πίνακα για κάθε κλάση μιας οντότητας. Το κάθε πεδίο αντιστοιχεί σε μια στήλη στον πίνακα. Επομένως, κάθε κλάση μιας οντότητας περιέχει τον πίνακα με όλα τα δεδομένα της οντότητας στη βάση δεδομένων. Στο παρακάτω παράδειγμα της εφαρμογής έχει δοθεί το όνομα του πίνακα “Table_Questions” . Εντός της κλάσης έχουν δοθεί τα ονόματα των στηλών.

```
@Entity(tableName = TABLE_QUESTIONS)
@Parcelize
data class QuestionData(
    @PrimaryKey(autoGenerate = true) val id: Int = 0,
    @ColumnInfo(name = "list_questions_id") val listQuestionsId: Int? = 0,
    @ColumnInfo(name = "question") val question: String? = null,
    @ColumnInfo(name = "answer_1") val answer1: String? = null,
    @ColumnInfo(name = "answer_2") val answer2: String? = null,
    @ColumnInfo(name = "answer_3") val answer3: String? = null,
    @ColumnInfo(name = "answer_4") val answer4: String? = null,
    @ColumnInfo(name = "correct_answer") val correctAnswer: String? = null,
    @ColumnInfo(name = "submitted_answer") var submittedAnswer: String? = null,
    @ColumnInfo(name = "checked_1") var checked1: Boolean? = false,
    @ColumnInfo(name = "checked_2") var checked2: Boolean? = false,
    @ColumnInfo(name = "checked_3") var checked3: Boolean? = false,
    @ColumnInfo(name = "checked_4") var checked4: Boolean? = false,
    @ColumnInfo(name = "hint") val hint: String? = null,
    @ColumnInfo(name = "chapter") val chapter: String? = null,
    @ColumnInfo(name = "question_type") val questionType: Int = 0
) : Parcelable
```

Εικόνα 50 Παράδειγμα Entity (Room)

[24] https://betterprogramming.pub/save-data-in-your-local-database-using-the-room-persistence-library-a9630c977234

Dao

Υπεύθυνο για τον καθορισμό των αφηρημένων μεθόδων (abstract methods) πρόσβασης στην βάση δεδομένων. Για κάθε DAO κλάση που σχετίζεται με τη βάση δεδομένων, η κλάση βάσης δεδομένων πρέπει να ορίσει μια αφηρημένη μέθοδο. Για να δημιουργηθεί ένα Dao θα πρέπει να δημιουργήσουμε ένα interface αποκαλώντας το @Dao.

```
@Dao
interface QuestionDao {

    @Query("SELECT * FROM $TABLE_QUESTIONS WHERE list_questions_id = :listQuestionsId")
    fun getQuestion(listQuestionsId: Int): Flow<List<QuestionData>>

    @Query("SELECT COUNT(*) FROM $TABLE_QUESTIONS WHERE list_questions_id = :listQuestionsId")
    fun getTotalQuestion(listQuestionsId: Int): Flow<Int>

    @Query("SELECT * FROM $TABLE_QUESTIONS")
    fun getAllQuestion(): Flow<List<QuestionData>>

    @Insert(onConflict = OnConflictStrategy.REPLACE)
    suspend fun insertList(questionData: List<QuestionData>)

    @Insert(onConflict = OnConflictStrategy.REPLACE)
    suspend fun insert(questionData: QuestionData)

    @Update
    suspend fun update(questionData: QuestionData)

    @Query("DELETE FROM $TABLE_QUESTIONS")
    suspend fun deleteAllQuestions()
}
```

Εικόνα 51 Παράδειγμα Interface (Dao)

Database

Στο παρακάτω παράδειγμα της εφαρμογής ορίζεται μια αφηρημένη κλάση AppDatabase, η οποία κληρονομεί την Roomdatabase. Αυτή παραθέτει την λίστα με τις οντότητες που σχετίζονται με την βάση δεδομένων καθώς και τις DAO μεθόδους.

```
@Database(
    entities = [StudentClassData::class, SectionsData::class, QuestionData::class],
    version = 1,
    exportSchema = false
)
abstract class AppDatabase : RoomDatabase() {

    abstract fun studentClassDao(): StudentClassDao
    abstract fun sectionsDao(): SectionsDao
    abstract fun questionDao(): QuestionDao

    class Callback @Inject constructor(
        private val database: Provider<AppDatabase>,
        @ApplicationScope private val applicationScope: CoroutineScope
    ) : RoomDatabase.Callback() {

        override fun onCreate(db: SupportSQLiteDatabase) {
            super.onCreate(db)

            val studentClass = database.get().studentClassDao()
            val sections = database.get().sectionsDao()
            val question = database.get().questionDao()

            applicationScope.launch {  this: CoroutineScope
                studentClass.insert(StudentClassData( id: 1,  title: "Ήρωες",  visibility: true))
                studentClass.insert(StudentClassData( id: 2,  title: "Μάχες",  visibility: true))
                studentClass.insert(StudentClassData( id: 3,  title: "Οθωμανοί",  visibility: true))
                studentClass.insert(StudentClassData( id: 4,  title: "Πόλεις-Περιοχές",  visibility: true))
                studentClass.insert(StudentClassData( id: 5,  title: "Χρονολογίες",  visibility: true))
                studentClass.insert(StudentClassData( id: 6,  title: "Πρόσωπα",  visibility: true))
                studentClass.insert(StudentClassData( id: 7,  title: "Σωστό ή Λάθος",  visibility: true))
                studentClass.insert(StudentClassData( id: 8,  title: "Γλώσσα",  visibility: true))
```

Εικόνα 52 Παράδειγμα κώδικα κληρονομικότητας της Roomdatabase από την κλάση AppDatabase

7.6 DialogFragment

Το DialogFragment είναι μια κλάση που κληρονομεί την κλάση Fragment. Χρησιμοποιείται για την εμφάνιση ενός παραθύρου που όταν αναδύεται, θα επικαλύπτει την δραστηριότητα. Ουσιαστικά το DialogFragment εμφανίζει έναν διάλογο μέσα σε ένα Fragment. Στην παρούσα εφαρμογή έχει χρησιμοποιηθεί για να εμφανίζει hints σε κάποιες ερωτήσεις. Τα hints αυτά είναι παράθυρα διαλόγου κειμένου (text) και καλούνται σε αρκετές ερωτήσεις μέσα στην εφαρμογή. Παρακάτω, παρατηρούμε στιγμιότυπο από την παρούσα εφαρμογή.

```
class DialogFragment : BottomSheetDialogFragment() {

    private val args: DialogFragmentArgs by navArgs()
    private lateinit var binding: FragmentDialogBinding
    override fun onCreateView(inflater: LayoutInflater, container: ViewGroup?, savedInstanceState: Bundle?): View {
        binding = FragmentDialogBinding.inflate(layoutInflater)
        return binding.root
    }

    override fun onViewCreated(view: View, savedInstanceState: Bundle?) {
        super.onViewCreated(view, savedInstanceState)
        binding.dialog.text = args.hint
    }
}
```

Εικόνα 53 Παράδειγμα κώδικα DialogFragment

8. Συμπεράσματα

Με την πάροδο των ετών, οι ανάγκες του ανθρώπου αυξάνονται σε όλους τους τομείς. Τα smartphones, τα οποία έχουν γνωρίσει τεράστια εξέλιξη τα τελευταία χρόνια, έχουν συμβάλει στη κάλυψη των αναγκών του ανθρώπου με αποτέλεσμα να αποτελούν αναπόσπαστο κομμάτι στην καθημερινότητά του.

Πλέον οι χρήστες, εκτός από την κάλυψη των καθημερινών τους αναγκών, αναζητούν και πιο μοντέρνες και εναλλακτικές μεθόδους για την ψυχαγωγία τους. Δεν αποκλείεται αυτές οι μέθοδοι πέρα από την ψυχαγωγία να προσφέρουν ταυτόχρονα στον χρήστη τη δυνατότητα για απόκτηση γνώσεων. Η παρούσα εφαρμογή για Android συσκευές που αφορά ερωτήσεις γνώσεων αναφορικά με την ιστορία του 1821, σχεδιάστηκε και υλοποιήθηκε με τέτοιο τρόπο, ώστε να παρέχεται μια ευχάριστη εμπειρία στον χρήστη. Αξιοποιεί όλες τις σύγχρονες τεχνολογίες που αξιοποιούν τα smartphones, ώστε να προσελκύουν όσο το δυνατόν περισσότερους χρήστες.

Σε αυτό συμβάλλει η σχεδίαση της διεπαφής χρήστη (UI) το οποίο πετυχαίνει θετικά αποτελέσματα στη βελτίωση της εμπειρίας χρήστη (UX) καθιστώντας την εφαρμογή όσο το δυνατόν πιο ευχάριστη και φιλική. Η ενσωμάτωση σύγχρονων εργαλείων στον κώδικα, πετυχαίνουν τον στόχο για τη δημιουργία μιας ευέλικτης εφαρμογής που θα απευθύνεται σε συσκευές μεγάλου εύρους δυνατοτήτων σε τεχνικά χαρακτηριστικά.

9. Μελλοντικές επεκτάσεις

Εξαιτίας του ανταγωνισμού και της συνεχούς εξέλιξης των smartphones και των υπηρεσιών που προσφέρουν, πρέπει να αναβαθμίζονται και να βελτιώνονται και οι εφαρμογές με επιπλέον δυνατότητες και χαρακτηριστικά για να συμβαδίζουν με την εξέλιξη αυτή. Παρακάτω αναφέρονται επιγραμματικά μερικά πλάνα για την μελλοντική επέκταση της εφαρμογής.

- Βελτίωση του γραφικού περιβάλλοντος (User Interface) και της εμπειρίας χρήστη (User Experience).
- Δημιουργία chat ώστε οι χρήστες να επικοινωνούν μεταξύ τους κατά τη διάρκεια της κατάταξης. Αυτό προϋποθέτει να γίνεται εγγραφή χρήστη κατά την είσοδο.
- Σειρά κατάταξης των εγγεγραμμένων χρηστών ανάλογα με τα αποτελέσματα που φέρουν.
- Ηχητική εκφώνηση των ερωτήσεων και των αποτελεσμάτων καθώς και ηχητικές εντολές, ώστε να μπορούν να τη χειρίζονται και άτομα με ειδικές ανάγκες.
- Διόρθωση τυχόν bugs που προκύψουν.

10. Βιβλιογραφικές αναφορές

1. History of smartphones

   https://books.google.gr/books?hl=el&lr=&id=DYd-DwAAQBAJ&oi=fnd&pg=PT28&dq=history+of+smartphones&ots=n_afoA24dN&sig=_8LNg9lzfCWeUrMzSyMKHfHBTE0&redir_esc=y#v=onepage&q=history%20of%20smartphones&f=false

2. Designveloper.com - Top 5 Best Android App Development Languages for 2023

   https://www.designveloper.com/blog/android-app-development-languages/

3. Simpletexting.com - Smartphone History and Evolution

   https://simpletexting.com/where-have-we-come-since-the-first-smartphone/

4. Developer.android.com - Platform Architecture
   https://developer.android.com/guide/platform

5. ART – Android RunTime
   https://www.tech-term.in/2018/01/22/art-android-runtime/

6. NDK and CMake
   https://developer.android.com/studio/projects/install-ndk?gclid=Cj0KCQiAmaibBhCAARIsAKUlaKTdto8KXNyJ_HUpkq6nymoho6P_IfHk89ckj8dQd6BGnLHzn-aJ1zgaArMPEALw_wcB&gclsrc=aw.ds

7. Android Architecture: Application Layers, Framework, Component
   https://www.guru99.com/android-architecture.html

8. App Manifest Overview
   https://developer.android.com/guide/topics/manifest/manifest-intro

9. Developers Activities
   https://stuff.mit.edu/afs/sipb/project/android/docs/guide/components/activities.html

10. Developers User Interface
    https://stuff.mit.edu/afs/sipb/project/android/docs/guide/topics/ui/index.html

11. UI Overview
    https://stuff.mit.edu/afs/sipb/project/android/docs/guide/topics/ui/overview.html

12. Mobile Operating System Market Share Worldwide
    https://gs.statcounter.com/os-market-share/mobile/worldwide

13. A History of Windows Phone: The life and death of Microsoft’s mobile platform
    https://mspoweruser.com/a-history-of-windows-phone-the-road-to-threshold/

14. iPhone 14
    https://www.apple.com/iphone-14/

15. Android Lifecycle
https://www.kodeco.com/21382977-android-lifecycle

16. Activity Lifecycle in Android with Demo App
https://www.geeksforgeeks.org/activity-lifecycle-in-android-with-demo-app/

17. Linear Layout
https://developer.android.com/develop/ui/views/layout/linear

18. Understanding App Permissions
https://guides.codepath.com/android/Understanding-App-Permissions

19. Permissions on Android
https://developer.android.com/guide/topics/permissions/overview

20. Dialogs
https://developer.android.com/develop/ui/views/components/dialogs

21. Creating Dialogs
http://www.dre.vanderbilt.edu/~schmidt/android/android-4.0/out/target/common/docs/doc-comment-check/guide/topics/ui/dialogs.html

22. Different Types Of Mobile Apps
https://www.emizentech.com/blog/types-of-mobile-apps.html

23. What Are the Different Types of Mobile Apps
https://clevertap.com/blog/types-of-mobile-apps/

24. Τα smartphones κατακτούν και την Ελλάδα - Εξι στους δέκα είναι online αγοραστές
https://www.imerisia.gr/tehnologia/47153_ta-smartphones-kataktoyn-kai-tin-ellada-exi-stoys-deka-einai-online-agorastes

25. What is Firebase
https://medium.com/firebase-developers/what-is-firebase-the-complete-story-abridged-bcc730c5f2c0

26. React Native Firebase
https://rnfirebase.io/auth/usage

27. Introduction to Material Design in Android
https://www.geeksforgeeks.org/introduction-to-material-design-in-android/

28. RecyclerView Multiple View Types in Android
https://blog.mindorks.com/recyclerview-multiple-view-types-in-android/

29. Animate transitions between destinations
https://developer.android.com/guide/navigation/navigation-animate-transitions

30. Using Room Database | Android Jetpack

https://medium.com/mindorks/using-room-database-android-jetpack-675a89a0e942

31. Save data in a local database using Room
https://developer.android.com/training/data-storage/room

32. What is an OnClickListener
https://www.educative.io/answers/what-is-an-onclicklistener

33. Top Programming Languages for Android App Development
https://www.geeksforgeeks.org/top-programming-languages-for-android-app-development/

34. Introduction to Firebase Realtime Database
https://www.kodeco.com/books/saving-data-on-android/v1.0/chapters/12-introduction-to-firebase-realtime-database