Ελληνικό Μεσογειακό Πανεπιστήμιο

Σχολή Τεχνολογικών Εφαρμογών

Τμήμα Μηχανικών Πληροφορικής

Πτυχιακή Εργασία
Ιατρική Ατζέντα σε Android

(Αποτελεί τμήμα κοινής εργασίας)

Επιμελητές: Ριτζάκης Γεώργιος (ΑΜ: 3888)
Κριτσωτάκης Νικόλαος (ΑΜ: 3794)

Επιβλέπων εκπαιδευτικός: Παπαδάκης Νικόλαος

Επιτροπή Αξιολόγησης: Παπαδάκης Νικόλαος
Χαράλαμπος Τσαγκαράκης

Ημερομηνία Παρουσίασης: 01-07-2020

Ευχαριστίες

Θα θέλαμε αμφότεροι να ευχαριστήσουμε αρχικά, τους καθηγητές μας που μας στάθηκαν καθ' όλη την διάρκεια της ακαδημαϊκής μας πορείας με την απλόχερη παραχώρηση των γνώσεων τους, και συγκεκριμένα τον κ. Παπαδάκη Νικόλαο για την ουσιαστική βοήθεια και καθοδήγηση του στην εκπόνηση της πτυχιακής εργασίας. Έπειτα, τις οικογένειες μας και κυρίως τους γονείς μας Κωνσταντίνος Κριτσωτάκης, Ειρήνη Μανουσάκη – Ιωάννης Ριτζάκης, Μαρία Μάγου για την αμέριστη στήριξη καθ' όλη την διάρκεια των σπουδών μας. Εν συνεχεία, αφιερώνουμε το παρόν σύγγραμμα στους αγαπημένους μας Στυλιανό Μανουσάκη, Μαρία Μανουσάκη, Νικόλαο Κριτσωτάκη, Αικατερίνη Κριτσωτάκη, – Αργυρώ Ριτζάκη.

Abstract

Thesis is based on an application running on an Android environment. The programming language used for its development is Kotlin. The main purpose of this innovative idea is to facilitate its user in organizing health related issues. This is a patient-user medical file with various functions. The application features, a user registration base, where anyone can create his own medical account. The user gains access to specific features with his own personal account. He can plan his weekly diet, use of medications, doctor appointments, sugar measurement or pressure, where he will receive a push-notification on his smartphone as a reminder, depending on the settings he had set. In addition, user will be able to enter the values of his examinations, which are subject to blood, biochemical and general tests where the application will inform him whether he is in the desired range or not. Finally, the user has the ability to calculate his body mass index (BMI) and body fat percentage through the application.

Σύνοψη

Η παρούσα πτυχιακή εργασία είναι βασισμένη σε εφαρμογή που λειτουργεί σε περιβάλλον Android. Η προγραμματιστική γλώσσα που χρησιμοποιήθηκε για την ανάπτυξη της, είναι η Kotlin. Ο βασικός σκοπός που οραματίστηκε η συγκεκριμένη καινοτόμος ιδέα είναι η διευκόλυνση του χρήστη της όσον αφορά την οργάνωση θεμάτων που σχετίζονται με την υγεία του. Πρόκειται για έναν ιατρικό φάκελο ασθενή - χρήστη με διάφορες λειτουργίες. Στην εφαρμογή παρουσιάζεται μία βάση εγγραφής χρηστών, όπου ο κάθε χρήστης μπορεί να δημιουργήσει το δικό του ιατρικό λογαριασμό. Ο χρήστης με το δικό του προσωπικό λογαριασμό αποκτά πρόσβαση σε συγκεκριμένες λειτουργίες. Μπορεί να προγραμματίσει το εβδομαδιαίο του πρόγραμμα διατροφής, λήψης φαρμάκων, ραντεβού με ιατρό και μετρήσεις σακχάρου ή πίεσης, όπου θα λαμβάνει ειδοποίηση στο κινητό του σαν υπενθύμιση ανάλογα με τις ρυθμίσεις που ο ίδιος έχει ορίσει. Επιπροσθέτως, ο χρήστης θα είναι σε θέση να εισάγει τις τιμές των εξετάσεων, που αυτές υπάγονται στις αιματολογικές, βιοχημικές και γενικές εξετάσεις, όπου η εφαρμογή θα τον ενημερώνει αν αυτές βρίσκονται σε επιθυμητά πλαίσια ή όχι. Τέλος, ο χρήστης έχει τη δυνατότητα μέσω της εφαρμογής, να υπολογίσει το δείκτη μάζας σώματός του (ΔΜΣ) καθώς και το ποσοστό του λίπους του.

Περιεχόμενα

Λίστα Εικόνων

Λίστα Πινάκων

1. Εισαγωγή

Όπως είναι ευρέως γνωστό, στη σημερινή εποχή τα κινητά τηλέφωνα διαδραματίζουν μείζονα ρόλο στη ζωή των ανθρώπων. Σύμφωνα με γνωστές εταιρείες παραγωγής στατιστικών, γίνεται γνωστό, ότι ο μέσος χρήστης ενασχολείται με το κινητό τηλέφωνό του, άνω των τεσσάρων (4) ωρών ημερησίως. Αυτό συμβαίνει διότι η κοινωνικοποίηση, η ψυχαγωγία, η ενημέρωση κ.α. που άλλοτε γίνονταν δια ζώσης, τώρα έχουν μεταφερθεί στον ψηφιακό κόσμο. Βάση αυτού του σκεπτικού, υλοποιήθηκε η παρούσα πτυχιακή εργασία, με τίτλο Ιατρική Ατζέντα, σε περιβάλλον Android. Πρόκειται για μία εφαρμογή η οποία θα βρίσκει χρησιμότητα σε οποιαδήποτε συσκευή που έχει ως λειτουργικό σύστημα το Android. Σκοπός της εφαρμογής που αναπτύχθηκε, είναι η διευκόλυνση του χρήστη δια μέσου μιας συσκευής Android που χρησιμοποιεί στην καθημερινή του ζωή, να μπορεί να έχει πρόσβαση και να διαχειρίζεται με ευκολία τον δικό του ιατρικό φάκελο, ο οποίος προσφέρει κάποιες λειτουργίες, όπου και θα αναφερθούν λεπτομερώς παρακάτω.

1.1 Περίληψη

Σκοπός της παρούσας πτυχιακής εργασίας είναι η μελέτη, σχεδίαση και ανάπτυξη μιας android εφαρμογής, η οποία διευκολύνει τον χρήστη, όσον αφορά την τακτοποίηση στοιχείων του, που σχετίζονται με την υγεία και τον τρόπο ζωής. Οι λειτουργίες που προσφέρονται από την «Ιατρική Ατζέντα», είναι η αποθήκευση των προσωπικών στοιχείων του χρήστη, το ιστορικό των εξετάσεων και των λοιπών μετρήσεων, στη βάση δεδομένων. Ο χρήστης μπορεί να πληκτρολογήσει τα αποτελέσματα των αιματολογικών, βιοχημικών και γενικών εξετάσεων και να διαπιστώσει αν βρίσκεται μέσα στα επιθυμητά όρια. Οι εξετάσεις θα αποθηκεύονται στη βάση δεδομένων της «Ιατρικής Ατζέντας» και θα δύναται να ανατρέξει σε αυτές όποτε επιθυμεί, για να δει παλαιότερα αποτελέσματα και να τα συγκρίνει με νέα. Επίσης, με τις απαραίτητες πληροφορίες που θα εισάγονται από το χρήστη στα ανάλογα πεδία της εφαρμογής, θα μπορεί να υπολογιστεί ο δείκτης μάζας σώματος και το ποσοστό λίπους στο σώμα. Τέλος, πολύ σημαντική λειτουργία της εν λόγω εφαρμογής αποτελεί η λήψη ειδοποιήσεων-υπενθυμίσεων σε μορφή «push» από τα εκ των προτέρων προγραμματισμένα γεγονότα από το χρήστη, για τα ημερήσια γεύματα, για τη λήψη φαρμάκων, για τη μέτρηση σακχάρου ή πίεσης και για τα ραντεβού με ιατρούς.

1.2 Κίνητρο για την Διεξαγωγή της Εργασίας – Στόχοι

Στόχος της πτυχιακής εργασίας είναι να έχει τη δυνατότητα ο χρήστης της εφαρμογής να έχει πρόσβαση στα ιατρικά του στοιχεία και να τα αρχειοθετεί, δίχως να υπάρχει κίνδυνος απώλειας τους, εφόσον δεν είναι σε έντυπη μορφή. Χάρη της εφαρμογής υπάρχει και σαφώς μειωμένος χρόνος στην αναζήτηση παλαιότερων εξετάσεων και μετρήσεων του, καθώς και καλύτερη οργάνωση στα ιατρικά του καθήκοντα, όπως προαναφέρθηκαν.

1.3 Δομή Εργασίας

Κεφάλαιο 1 – Εισαγωγή: Επρόκειτο για γενικές πληροφορίες για την εργασία.

Κεφάλαιο 2 – Android: Λεπτομέρειες λειτουργικού συστήματος Android.

Κεφάλαιο 3 – Εργαλεία Υλοποίησης Πτυχιακής Εργασίας: Πληροφορίες για την γλώσσα προγραμματισμού που χρησιμοποιήθηκε και το περιβάλλον ανάπτυξής της.

Κεφάλαιο 4 – Παρουσίαση Εφαρμογής «Ιατρική Ατζέντα – Medical Agenda»: Παρουσίαση των λειτουργιών της εφαρμογής.

2. Android

Εικόνα 1. Λογότυπο πλατφόρμας Android
(https://en.wikipedia.org/wiki/Android_(operating_system))

2.1 Γενικές Πληροφορίες

Το Android αρχικά είναι ένα λειτουργικό σύστημα (Operating System) στηριγμένο σε μια παραλλαγή του λειτουργικού συστήματος Linux, σχεδιασμένο κατά κύριο λόγο για φορητές συσκευές που λειτουργούν με την αφή, όπως smartphones και tablets. Το λειτουργικό σύστημα Android αναπτύσσεται από μια συνεργασία προγραμματιστών, γνωστή ως Open Handset Alliance, με εμπορικό πρωταγωνιστή να είναι η Google.[1]

Η εταιρία που ανέπτυξε το Android ήταν η Android Inc., την οποία το 2005 την εξαγόρασε η Google. Το 2007 έγινε η πρώτη παρουσίαση του Android και τον Σεπτέμβριου του 2008 ξεκίνησε η πρώτη εμπορική συσκευή Android. Το Android έχει ως πηγαίο κώδικα τον Android Open Source Project (AOSP), ο οποίος έχει κυρίως άδεια από την Apache. Αυτό επέτρεψε την ανάπτυξη παραλλαγών του Android σε άλλες ηλεκτρονικές συσκευές, όπως οι παιχνιδοκονσόλες, οι ψηφιακές φωτογραφικές μηχανές, οι υπολογιστές κ.α. Το Android έχει επίσης εφαρμογή στις τηλεοράσεις Android TV, και Wear OS για αξεσουάρ χειρός (όπως ρολόγια), τα οποία έχουν αναπτυχθεί από την Google.

Ο πηγαίος κώδικας του Android χρησιμοποιήθηκε ως βάση για διαφορετικά συστήματα, κυρίως για την Google, ο οποίος συσχετίζεται με μια βιβλιοθήκη λογισμικού με όνομα Google Mobile Services (GMS), που συχνά προστίθεται στις εν λόγω συσκευές. Το οποίο περιλαμβάνει βασικές εφαρμογές όπως το Youtube, Gmail, Google Maps, το Google Play (ψηφιακό κατάστημα παιχνιδιών και εφαρμογών) και συνήθως εφαρμογές όπως το Google Chrome. Σε γενική ομολογία το GMS είναι ένα πακέτο εφαρμογών που έρχονται μαζί με κάθε συσκευή που χρησιμοποιεί Android λογισμικό. Οι συγκεκριμένες εφαρμογές, παρέχουν άδεια χρήσης από κατασκευαστές Android συσκευών, που έχουν πιστοποιηθεί σύμφωνα με πρότυπα που έχουν επιβληθεί από την Google. Άλλη ανταγωνιστική εταιρεία παραγωγής προϊόντων Android αποτελεί για παράδειγμα το FireOS (που αναπτύχθηκε από την Amazon) ή το LineageOS . Η διανομή λογισμικού προσφέρεται γενικά μέσω ιδιόκτητων καταστημάτων εφαρμογών (Google Play Store, Samsung Galaxy Store ή πλατφόρμες ανοιχτού κώδικα όπως Aptoide, F-Droid) που χρησιμοποιούν πακέτα λογισμικού στη μορφή APK.[2]

Το λειτουργικό Android έχει τις περισσότερες πωλήσεις σε ολόκληρο τον κόσμο, από το 2011 σε κινητά τηλέφωνα και από το 2013 σε tablets. Από τον Μάιο του 2017, κατάφερε να αποκτήσει πάνω από 2 δις. μηνιαίους χρήστες και την ευρύτερη βάση δεδομένων από οποιοδήποτε άλλο λειτουργικό σύστημα. Από τον Ιανουάριο του 2020, το Play Store της Google περιέχει πάνω από 3 εκ. εφαρμογές.[3]

2.2 Ιστορική Αναδρομή

Εικόνα 2. Εξέλιξη λογότυπου Android
(https://en.wikipedia.org/wiki/Android_(operating_system))

Τον Οκτώβριο του 2003 στην Καλιφόρνια και συγκεκριμένα στο Palo Alto, οι Andy Rubin, Chris White, Nick Sears και Rich Miner ίδρυσαν την εταιρία «Android Inc.»[4][5] Ο Andy δήλωσε χαρακτηριστικά ότι το Android είναι ένα τεράστιος κινητήριος μοχλός για την δημιουργία smartphones που θα παρέχουν σημαντικές ευκολίες στους κατόχους τους.[5] Αρχικά η εταιρία είχε ως σκοπό την ανάπτυξη ενός εξελιγμένου λειτουργικού συστήματος για ψηφιακές φωτογραφικές μηχανές όπου από το 2004 οι επιχειρηματίες που στήριξαν αυτό το όραμα βασίστηκαν σε αυτή την αρχική ιδέα.[6] Η εταιρεία τότε λοιπόν αποφάσισε, ότι η αγορά φωτογραφικών μηχανών δεν εκπλήρωνε τις υψηλές της προσδοκίες και πέντε μήνες αργότερα έθεσε άλλες βλέψεις και ανέβασε το Android ως λειτουργικό σύστημα για φορητές ηλεκτρονικές συσκευές, που θα ανταγωνιζόταν τις Symbian και Microsoft Windows Mobile.[6][7]

Μέχρι και το Δεκέμβριο του 2006, συνέχιζαν να χτίζονται υποθέσεις, σχετικά με την πεποίθηση της Google να «εισβάλει» στην αγορά των τηλεπικοινωνιών. Ένα από τα πρώτα τηλέφωνα Android ξεκίνησε χωρίς οθόνη αφής και ένα φυσικό QWERTY πληκτρολόγιο και για όσους γνωρίζουν μπορούμε να πούμε ότι έμοιαζε με ένα BlackBerry τηλέφωνο, αλλά η άφιξη του iPhone της Apple το 2007 σήμαινε ότι το Android έπρεπε να αλλάξει τις σχεδιαστικές της αρχές για λόγους ανταγωνισμού.[8][9] Η Google δήλωσε αργότερα πως θα υποστηρίζει συσκευές με οθόνη αφής.[10] Μέχρι το 2008, άλλες 2 μεγάλες εταιρίες, οι BlackBerry-Nokia ανακοίνωσαν κινητά τηλέφωνα με υποστήριξη οθόνη αφής, για να ανταγωνιστούν της Apple το iPhone 3G, και η εστίαση του Android τελικά μεταβλήθηκε από τις ανάγκες που έφερνε στην επιφάνεια η ανταγωνιστική αγορά, σε τεχνολογίες εφάμιλλες αυτών, της οθόνης αφής. Το πρώτο smartphone που διατέθηκε στην αγορά και που είχε ως λειτουργικό σύστημα το Android ανακοινώθηκε 23 Σεπτεμβρίου του 2008 και ήταν το HTC Dream, επίσης πασίγνωστο ως T-Mobile G1, αυτή η ημερομηνία σήμανε την τρανταχτή είσοδο των Android στην αγορά.[11][12]

Εικόνα 3. Το πρώτο κινητό Android HTC Dream – Google G1
(https://news.softpedia.com/news/Remember-The-First-Android-Smartphone-Was-the-HTC-Dream-or-Google-G1-472839.shtml)

Στις 5 Νοεμβρίου του 2007, το Open Handset Alliance (OHA), μια συνεργασία των τεχνολογικών εταιρειών (όπως Google), κατασκευαστές συσκευών (όπως HTC,

Motorola, Samsung), ασύρματοι φορείς (όπως Sprint, T-Mobile) και chipset κατασκευαστές (όπως Qualcomm, Texas Instruments) αποκάλυψαν τον στόχο να αναπτύξουν για πρώτη φορά μια ολοκληρωμένη πλατφόρμα με ανοικτό κώδικα, για κινητά τηλέφωνα.[13][14][15] Μέσα σε ένα χρόνο, διακρίθηκαν άλλοι 2 ανταγωνιστές της Open Handset Alliance, η Symbian και η LiMo Foundation, η LiMo αναπτύσσει επίσης ένα παρόμοιο λειτουργικό σύστημα, όπως της Google. Το InformationWeek, τον Σεπτέμβριο του 2007, έκανε μια μελέτη αξιολόγησης, αναφέροντας ότι η Google είχε υποβάλει αρκετές αιτήσεις ευρεσιτεχνιών στον τομέα της κινητής τηλεφωνίας.[16]

Έως και το 2008, στο Android έχουν πραγματοποιηθεί αρκετές αναβαθμίσεις, οι οποίες βελτίωναν σταδιακά το λειτουργικό του σύστημα, εισάγοντας καινούρια στοιχεία και διορθώνοντας κάποια σφάλματα σε προηγούμενες εκδόσεις. Κάθε δημοφιλής κυκλοφορία έκδοσης του Android ταξινομείται σε αλφαβητική σειρά με όνομα από επιδόρπιο ή γλύκισμα. Με τις πρώτες εκδόσεις του Android που κυκλοφόρησαν στην αγορά, να παίρνουν την ονομασία "Cupcake", "Donut", "Eclair" και "Froyo", με τη συγκεκριμένη σειρά. Πριν από αυτές τις επίσημες εκδόσεις, γινόντουσαν δοκιμές εκ του Android με ονομασίες εξ' αυτών "Alpha" (23 Σεπτεμβρίου 2008) και "Beta" (9 Φεβρουαρίου 2009). Το 2013 που ανακοινώθηκε το Android KitKat, η Google διευκρίνισε ότι «Δεδομένου ότι αυτές οι συσκευές κάνουν τη ζωή μας τόσο γλυκιά, κάθε έκδοση Android παίρνει το όνομα της από ένα επιδόρπιο».[17]

Η Google παρουσίασε το 2010 μία νέα σειρά συσκευών, με ονομασία Nexus, όπου συνεργάστηκε με διάφορους κατασκευαστές για την κατασκευή νέων συσκευών και την παραγωγή νέων εκδόσεων του Android. Αυτή η σειρά που προαναφέρθηκε, έχει ειπωθεί ότι «έχει διαδραματίσει καθοριστικό ρόλο στην ιστορία του Android εισάγοντας νέες εκδόσεις λογισμικού και προδιαγραφών υλικού σε όλους τους τομείς» και έγινε γνωστή για το λογισμικό "free-bloat" με άμεσες ενημερώσεις (updates).[18] Τον Μάϊο του 2013, η Google ανακοίνωσε μια διαφορετική ιδέα για την έκδοση του Samsung Galaxy S4. Αντί η Samsung να χρησιμοποιήσει τη δική της έκδοση Android, το τηλέφωνο έτρεξε "stock Android" και υποσχέθηκαν ότι θα ενημερώσουν το λειτουργικό σύστημα έγκαιρα.[19] Η συσκευή αυτή έγινε η βάση για την πλατφόρμα Google Play, όπου η εν λόγω πλατφόρμα ακολουθήθηκε και από άλλες συσκευές αργότερα, όπως η έκδοση HTC One Google Play,[20] και η έκδοση MotoG Google Play.[21]

Ο Hugo Barra, τις χρονιές από 2008 έως 2013, εργάστηκε ως αντιπρόσωπος των προϊόντων, και εκπροσώπησε το Android σε συνεντεύξεις τύπου καθώς και στο συνέδριο που γίνεται κάθε χρόνο για προγραμματιστές, Google I-O. Τον Αύγουστο του 2013, αποχώρησε από την Google και μεταπήδησε επαγγελματικά σε μία άλλη εταιρεία κατασκευής τηλεφώνων, με έδρα την Κίνα, την Xiaomi.[22] Νωρίτερα, πριν από έξι μήνες, ο Larry Page, διευθύνων σύμβουλος της Google, ανάρτησε σε blog, ότι ο Andy Rubin ανέλαβε νέα καθήκοντα σε νέα πρότζεκτ της Google και αποχώρησε από το τμήμα Android. Επίσης, ο Sundar Pichai έγινε ο νέος καθοδηγητής του Android.[23][24] Ο ίδιος ο Pichai άλλαξε τελικά την θέση του και έγινε ο νέος διευθύνων σύμβουλος της εταιρίας, τον Αύγουστο του 2015, μετά την συνένωση της εταιρείας με την εταιρία Alphabet,[25][26] καθιστώντας τον Hiroshi Lockheimer ως νέο επικεφαλή του Android.[27][28]

Η Google, το καλοκαίρι του 2014, ελευθέρωσε στο εμπόριο το Android One, ένα νέο μοντέλο υλικού (hardware) που επέτρεπε στους κατασκευαστές να αναπτύξουν εύκολα συσκευές υψηλής τεχνολογίας, με ελάχιστο κόστος, σχεδιασμένα για αγοραστές στις υπό ανάπτυξη χώρες.[29][30][31] Το Σεπτέμβριο του ίδιου έτους, η Google ανακοίνωσε το πρώτο σετ Android One κινητών, για κυκλοφορία στην Ινδία.[32][33]

Τον Οκτώβριο του 2016, η Google ανακοίνωσε τα νέα smartphones, Pixel και Pixel XL τα οποία εντάχθηκαν στο εμπόριο ως τα πρώτα τηλέφωνα που δημιουργήθηκαν από την ίδια

την Google,[34][35] και παρουσίασαν αποκλειστικά μερικές λειτουργίες λογισμικού, όπως το "Google Assistant", πριν από την ανάπτυξη περαιτέρω λειτουργιών λογισμικού.[36][37] Τον Οκτώβριο του 2017,[39] το Pixel αντικατέστησε τη σειρά Nexus,[38] με μια νέα γενιά Pixel τηλεφώνων.

Τον Μάιο του 2019, το λειτουργικό σύστημα Android μπλέχτηκε στον εμπορικό πόλεμο μεταξύ Κίνας και Ηνωμένων Πολιτειών, με τη εταιρία Huawei, η οποία όπως και πολλές άλλες εταιρείες τεχνολογίας έχουν εξαρτηθεί από την πρόσβαση στην πλατφόρμα Android.[40][41] Το καλοκαίρι του 2019, η Huawei ανακοίνωσε ότι θα δημιουργήσει ένα εναλλακτικό λειτουργικό σύστημα για Android[42][43] γνωστό ως Harmony OS,[44] και έχουν κατατεθεί για τα δικαιώματα πνευματικής ιδιοκτησίας σε μεγάλες παγκόσμιες αγορές.[45][46] Η Huawei δεν έχει επί του παρόντος σχέδια να αντικαταστήσει το Android στο εγγύς μέλλον, καθώς το Harmony OS σχεδιάζεται για το "Internet of Things"[47] (το διαδίκτυο των πραγμάτων (IoT) είναι ένα σύστημα με αλληλένδετες υπολογιστικές συσκευές, μηχανικές και ψηφιακές μηχανές, εφοδιασμένες η κάθε μία με ένα αναγνωριστικό στοιχείο ταυτότητας (UID) και με τη δυνατότητα μεταφοράς δεδομένων μέσω δικτύου, χωρίς να είναι απαραίτητη η αλληλεπίδραση μεταξύ ανθρώπων ή ανθρώπων με υπολογιστές).

Στις 22 Αυγούστου 2019, ανακοινώθηκε ότι το Android "Q" θα κυκλοφορήσει επισήμως ως Android 10, δίνοντας ένα τέλος στις ονομασίες των νέων εκδόσεων Android με ονόματα επιδορπίων-γλυκισμάτων. Η Google δήλωσε ότι αυτά τα ονόματα δεν ήταν αποκλειστικά για τους διεθνείς χρήστες (λόγω είτε των προαναφερθέντων τροφίμων που δεν είναι γνωστά διεθνώς, είτε είναι δύσκολο να διατυπωθούν σε ορισμένες γλώσσες).[48][49] Την ίδια ημέρα, ο τύπος Android Police ανέφερε ότι η Google είχε αναθέσει την εγκατάσταση ενός γιγαντιαίου αγάλματος με τον αριθμό "10" στο προθάλαμο του νέου γραφείου των προγραμματιστών.[50] Το Android 10 κυκλοφόρησε στις 3 Σεπτεμβρίου 2019, αρχικά σε κινητά τηλέφωνα Google Pixel.

2.3 Εξέλιξη των εκδόσεων Android

Ονομασία έκδοσης	Αριθμός έκδοσης	Ημερομηνία κυκλοφορίας	Επίπεδο API
Alpha	1.0	23 Σεπτεμβρίου 2008	1
Beta	1.1	9 Φεβρουάριου 2009	2
Cupcake	1.5	27 Απριλίου 2009	3
Donut	1.6	15 Σεπτεμβρίου 2009	4
Eclair	2.0 – 2.1	26 Οκτωβρίου 2009	5 – 7
Froyo	2.2 – 2.2.3	20 Μαΐου 2010	8
Gingerbread	2.3 – 2.3.7	6 Δεκεμβρίου 2010	9 – 10
Honeycomb	3.0 – 3.2.6	22 Φεβρουάριου 2011	11 – 13
Ice Cream Sandwich	4.0 – 4.0.4	18 Οκτωβρίου 2011	14 – 15
Jelly Bean	4.1 – 4.3.1	9 Ιουλίου 2012	16 – 18
KitKat	4.4 – 4.4.4	31 Οκτωβρίου 2013	19 – 20
Lollipop	5.0 – 5.1.1	12 Νοεμβρίου 2014	21 – 22
Marshmallow	6.0 – 6.0.1	5 Οκτωβρίου 2015	23
Nougat	7.0 – 7.1.2	22 Αυγούστου 2016	24 – 25
Oreo	8.0 – 8.1	21 Αυγούστου 2017	26 – 27
Pie	9.0	6 Αυγούστου 2018	28
Q	10.0	3 Σεπτεμβρίου 2019	29
R	11.0	19 Φεβρουαρίου 2020	30

Πίνακας 1. Εξέλιξη των εκδόσεων Android[51]

Εικόνα 4. Εξέλιξη λογότυπων εκδόσεων Android
(https://mindster.com/evolution-android/)

2.3.1 Android Alpha - Beta (version 1.0 – 1.1)

Το Android Alpha πρωτοπαρουσιάστηκε στις 23 Σεπτεμβρίου του 2008 με ονομασία Android 1.0 ή Android Alpha, μια έκδοση που δεν είχε κάποιο κωδικό όνομα. Η έκδοση Alpha δεν περιείχε όλα τα δυνατά χαρακτηριστικά που σχεδιάστηκαν για την ολοκληρωμένη έκδοση. Η πρώτη έκδοση (στην συγκεκριμένη περίπτωση: Android 1.0 "Alpha"), δοκιμαζόταν μόνο από το προσωπικό του οργανισμού που προγραμμάτιζε το λογισμικό.

Το Android Beta είναι η 1.1 σε αριθμόν έκδοση και η δεύτερη γενική έκδοση για το λειτουργικό σύστημα Android. Ανακοινώθηκε τον Φεβρουάριο του 2009 "με επίπεδο API 2" και είναι η επόμενη έκδοση του Android Alpha και προηγούμενη έκδοση του Android 1.5 "Cupcake". Η δεύτερη έκδοση (Android 1.1 "Beta"), περιλαμβάνει έναν μικρό ποσοστό χρηστών. Το λογισμικό τότε περιείχε μια ακολουθία από πρώϊμες εφαρμογές όπως Google, Gmail, Χάρτες, Ημερολόγιο και το Youtube, οι οποίες εμπεριέχθηκαν στο λειτουργικό σύστημα.[60][61]

2.3.2 Android Cupcake (version 1.5)

Το Android 1.5 "Cupcake" ήταν η τρίτη στην σειρά έκδοση του Android που δημιουργήθηκε από την Google. Κυκλοφόρησε 27 Απριλίου του 2009 και είναι μια μεγάλη αναβαθμισμένη πλατφόρμα για τις συσκευές με λειτουργικό σύστημα το Android. Η έκδοση περιλαμβάνει αλλαγές στο Android API και νέα χαρακτηριστικά για τους χρήστες και τους προγραμματιστές. Το Cupcake βελτίωσε αρκετά το UI (user interface / διεπαφή χρήστη) του Android, συμπεριλαμβανομένου του πρώτου ψηφιακού πληκτρολογίου, το οποίο ήταν απαραίτητο, καθώς τα κινητά τηλέφωνα πλέον δεν έχουν φυσικό πληκτρολόγιο. Το Cupcake επίσης εισήγαγε τις widget εφαρμογές, οι οποίες μετατράπηκαν σε ένα από τα πιο διακριτά στοιχεία του Android. Τέλος, ήταν η πρώτη έκδοση που παρείχε την επιλογή για εγγραφή βίντεο και την υποστήριξη Bluetooth.[52][60][61]

2.3.3 Android Donut (version 1.6)

Το Android Donut 1.6, κυκλοφόρησε 15 Σεπτεμβρίου του 2009. Η οποία έκδοση, κάλυψε μερικά κενά στο Android. Ένα από αυτά ήταν η δυνατότητα του λειτουργικού συστήματος να δουλέψει σε μια ποικιλία με διαφορετικά μεγέθη οθονών, όπου είχε σημαντικό ρόλο για τα επόμενα χρόνια. Το Donut επίσης, προσθέτει υποστήριξη για δίκτυα CDMA, ενός δείκτη μπαταρίας και υποστήριξη μετατροπής κειμένου σε ομιλία.[53][60][61]

2.3.4 Android Eclair (version 2.0 – 2.1)

Στις 26 Οκτωβρίου 2009, έξι εβδομάδες μετά την κυκλοφορία του Android Donut, κυκλοφόρησε το Android 2.0 Eclair. Η αναβαθμισμένη έκδοση "2.1", που ονομάζεται επίσης Eclair, εμφανίστηκε λίγους μήνες αργότερα. Μερικά από τα νέα προστεθέντα χαρακτηριστικά είναι το "home screen" του Eclair που εμφανίζει την αναζήτηση Google (Google Search) στην κορυφή της οθόνης, επίσης η εφαρμογή Camera έχει επαναπροσδιοριστεί με αρκετές νέες προδιαγραφές για φωτογραφίες, συμπεριλαμβανομένου του ψηφιακού zoom, εφέ χρώματος, της macro-εστίασης, Flash και της λειτουργίας σκηνής. Η εφαρμογή Gallery, εμπεριέχει επίσης βασικά εργαλεία επεξεργασίας φωτογραφιών. Εκτός από τις ζωντανές ταπετσαρίες, προσθέτει τις κινούμενες εικόνες φόντου στην αρχική οθόνη. Για πρώτη φορά παρουσιάστηκε η ομιλία σε κείμενο, αντικαθιστώντας το πλήκτρο κόμμα. Τέλος, το Android Eclair υιοθετεί χαρακτηριστικά στην πλατφόρμα από την έκδοση Donut και προσθέτει την δυνατότητα αναζήτησης σε όλα τα αποθηκευμένα μηνύματα SMS και MMS, υποστήριξης για επικοινωνία κοντινού πεδίου (NFC), όπως επίσης της βελτίωσης της υποστήριξης Exchange της εφαρμογής Email και του Google Maps 3.1.2.[54][60][61]

2.3.5 Android Froyo (version 2.2)

Στις 20 Μαΐου του 2010 η Google έφερε στο παρασκήνιο το Android 2.2 εν ονόματι Froyo, το οποίο ασχολήθηκε περισσότερο σε βελτιώσεις όσον αναφορά τις επιδόσεις της συσκευής. Το Froyo έκανε σημαντικές αλλαγές στα front-facing χαρακτηριστικά, συμπεριλαμβανομένης της προσθήκης των εφαρμογών στο κάτω μέρος της αρχικής οθόνης, τα Push Notifications, καθώς και της ενσωμάτωσης των φωνητικών ενεργειών που επιτρέπουν την εκτέλεση βασικών λειτουργιών. Μία από τις πιο σημαντικές αλλαγές στην κυκλοφορία του Froyo ήταν η υποστήριξη για το Flash στον περιηγητή ιστού του Android, η πρόσδεση με USB καλώδιο και η λειτουργία του Wi-Fi hotspot. Από τις 9 Ιανουαρίου του 2016 τα στατιστικά στοιχεία που εκδίδονται από την Google, επιβεβαιώνουν ότι λιγότερο από το 0,1% όλων των συσκευών Android που έχουν πρόσβαση στο Google Play διαθέτουν την έκδοση Froyo, κάτι που σημαίνει ότι αυτή η έκδοση δεν είναι πλέον σε χρήση.[55][60][61]

2.3.6 Android Gingerbread (version 2.3)

Στις 6 Δεκεμβρίου του 2010, η έκδοση Gingerbread, ήταν η αρχή της επικέντρωσης της οπτικής ταυτότητας του Android. Μαζί με την κυκλοφορία του Gingerbread, νέα χαρακτηριστικά ήρθαν στην επιφάνια του λειτουργικού συστήματος, όπως το SIP (Session Initiation Protocol / Πρωτόκολλο εκκίνησης συνόδου) που χρησιμοποιείται στην τηλεφωνία VoIP μέσω internet και η υποστήριξη του NFC (Near-Feald Communication / Επικοινωνία Κοντινού Πεδίου) που χρησιμοποιείται κυρίως σε πληρωμές μέσω κινητών. Το 2010, κυκλοφόρησε το smart phone Nexus S, ήταν το πρώτο κινητό τηλέφωνο από τη σειρά Nexus που είχε την έκδοση Gingerbread και η πρώτη συσκευή με ενσωματωμένη λειτουργία NFC. Το user interface του Gingerbread είχε τελειοποιηθεί με πολλές μεθόδους, πράγμα που το καθιστά ευκολότερο και πιο γρήγορο στη χρήση και πιο αποδοτικό ενεργειακά. Η βελτιστοποιημένη παλέτα με τα χρώματα έδωσε ζωντάνια και αντίθεση στη μπάρα με τις ειδοποιήσεις, στο μενού και σε άλλα στοιχεία του user interface. Η βελτίωση στις ρυθμίσεις και στο μενού οδήγησαν στην ευκολότερη πλοήγηση και τον έλεγχο του συστήματος για τον χρήστη. Τα χρώματα,

πράσινο και μαύρο εισχωρήσαν σε όλο το user interface και τότε το Android άρχισε να ξεχωρίζει όσον αναφορά τον σχεδιασμό του.[60][61]

2.3.7 Android Honeycomb (version 3.0 – 3.2)

Στις 22 Φεβρουαρίου του 2011, κυκλοφόρησε το Android Honeycomb, αλλά μόνο για tablets, που συνόδεψε το ντεμπούτο του Motorolo Xoom και με τις ακόλουθες ενημερώσεις 3.1 και 3.2 συνέχισε να είναι μια έκδοση μόνο για tablets. Με τον Matias Duarte, ως νέος επικεφαλής στον τομέα του σχεδίου, η Honeycomb έκανε ενσωμάτωση ενός δραματικά επαναπροσδιορισμένου user interface για το λειτουργικό Android. Διέθετε ένα "ολογραφικό" σχεδιασμό και ένα μοντέλο αλληλεπίδρασης που ενσωματώθηκε πάνω στα βασικά χαρακτηριστικά του λειτουργικού, όπως τα widgets, τις κοινοποιήσεις και το multitasking. Ενώ στην αρχή η ιδέα αφορούσε μόνο τα tablets, το Honeycomb έθεσε τα θεμέλια για το Android που όλοι γνωρίζουν σήμερα. Ήταν το πρώτο λογισμικό που ενσωμάτωσε τα ψηφιακά κουμπιά στην οθόνη (Back, Home, Switch app) για τις βασικές εντολές του Android.[57][58][60][61]

2.3.8 Android Ice Cream Sandwitch (version 4.0)

Μετά το Honeycomb, στις 18 Οκτωβρίου του 2011, κυκλοφόρησε η νέα έκδοση του Android, που είχε ως όνομα, Ice Cream Sandwich. Με την έκδοση αυτή, βελτιώθηκαν οι οπτικές έννοιες που εισήχθησαν από τα tablet με την έκδοση Honeycomb και κινητά με ένα ενιαίο user interface. Το Ice Cream Sandwich τροποποίησε την εμφάνιση της Honeycomb, αλλά κράτησε κάποια χαρακτηριστικά όπως το μπλε χρώμα. Έφερε βασικά στοιχεία στο σύστημα, όπως την εναλλαγή εφαρμογών μέσω μία κάρτας. Η συγκεκριμένη έκδοση υιοθέτησε την ιδέα του swipe away ειδοποιήσεων και πρόσφατων εφαρμογών. Επίσης, έγινε η αρχή του τυποποιημένου πλαισίου στον σχεδιασμό, γνωστόν ως “Holo” σε όλο το λειτουργικό σύστημα. Με την κυκλοφορία της έκδοσης αυτής εγκαταστάθηκαν αρκετά νέα χαρακτηριστικά, συμπεριλαμβανομένης της δυνατότητας ελέγχου της μουσικής και της πρόσβασης στην Camera από την οθόνη κλειδώματος, μιας ανανεωμένης αρχικής οθόνης, της δυνατότητας παρακολούθησης και περιορισμού της χρήσης δεδομένων κινητής επικοινωνίας, της επιλογής αναγνώρισης προσώπου για ξεκλείδωμα οθόνης (Face Unlock), μίας ενημερωμένης εφαρμογής web browser, μίας νέας εφαρμογής για τις επαφές με την ενσωμάτωση των κοινωνικών δικτύων, της βελτίωσης της επικοινωνίας κοντινού πεδίου (NFC), της οπτικής υποστήριξης τηλεφωνητή, και άλλων εσωτερικών βελτιώσεων.[59][60][61]

2.3.9 Android Jelly Bean (version 4.1 – 4.3)

Η έκδοση Jelly Bean κυκλοφόρησε σε 3 διαφορετικές εκδόσεις. Στις 9 Ιουλίου του 2012 και στις αρχές του 2013 πραγματοποιήθηκαν σημαντικοί βηματισμοί στην τελειοποίηση και την οικοδόμηση των καινούριων αυτών εκδόσεων. Οι εκδόσεις αυτές είχαν ως προσπάθεια να κάνουν τον χρήστη, πιο φιλόξενο με το Android. Το Jelly Bean ενσωμάτωσε το Google Play, που στην αρχή είχε ως σκοπό την πρόβλεψη ευφυΐας, αλλά τελικά κατέληξε σε μια εφαρμογή ειδήσεων. Επίσης έδωσε στον χρήστη την εκτεταμένη και διαδραστική ειδοποίηση, την δυνατότητα για φωνητική αναζήτηση και ένα πιο προηγμένο σύστημα για να εμφανίζει τα αποτελέσματα μίας αναζήτησης, με ιδιαίτερη σημασία στα αποτελέσματα. Όσον αναφορά τα tablets, ενσωματώθηκε η υποστήριξη πολλαπλών χρηστών. Ακόμη διακρίθηκε μια πρώιμη έκδοση του πίνακα για τις γρήγορες ρυθμίσεις στην συρόμενη μπάρα του Android. Τέλος,

τοποθετήθηκαν widgets στην οθόνη κλειδώματος, τα οποία εξαφανίστηκαν στις επόμενες εκδόσεις όπως και άλλα 27 χαρακτηριστικά.

2.3.10 Android KitKat (version 4.4)

Στις 3 Σεπτεμβρίου του 2013, παρουσιάστηκε η έκδοση KitKat που κυκλοφόρησε στις 31 Οκτωβρίου του 2013, το μαύρο χρώμα του Gingerbread και το μπλε χρώμα της Honeycomb εν τέλει αποσύρθηκαν από το user interface του λειτουργικού συστήματος. Μία διάφανη γραμμή κατάστασης και λευκά εικονίδια έδωσαν στο λειτουργικό σύστημα ένα πιο μοντέρνο σχεδιασμό. Η έκδοση KitKat ακολούθησε επίσης μαζί με την πρώτη έκδοση του Google Assistant με την εντολή "OK, Google", αλλά στην συγκεκριμένη έκδοση, η εντολή μπορούσε να λειτουργήσει μόνο όταν η οθόνη ήταν σε αδράνεια.[60][61]

2.3.11 Android Lollipop (version 5.0 – 5.1)

Η Google διαμόρφωσε και πάλι το Android με το Android Lollipop που κυκλοφόρησε στις 12 Νοεμβρίου 2014. Ένα από τα κύρια χαρακτηριστικά του Lollipop είναι ένα επανασχεδιασμένο περιβάλλον εργασίας χρήστη κατασκευασμένο γύρω από μια σχεδιαστική γλώσσα, γνωστή ως Material Design, η οποία κατασκευάστηκε για την διατήρηση μιας αίσθησης χαρτιού στο περιβάλλον. Μερικές αλλαγές ακόμα είναι οι βελτιώσεις στις ειδοποιήσεις, οι οποίες είναι προσβάσιμες από την οθόνη κλειδώματος. Η Google έκανε επίσης εσωτερικές αλλαγές στην πλατφόρμα, με το Android Runtime (ART) αντικατέστησε το Dalvik για να βελτιώσει την απόδοση των εφαρμογών, καθώς και τη βελτίωση και βελτιστοποίηση της χρήσης μπαταρίας.[60][61]

2.3.12 Android Marshmallow (version 6.0)

Το Marshmallow κυκλοφόρησε στις 5 Οκτωβρίου του 2015 και επικεντρώθηκε κυρίως να βελτιώσει την συνολική εμπειρία του χρήστη, σε σύγκριση με την προηγούμενη έκδοση, Lollipop. Πρόσθεσε μια καινούρια αρχιτεκτονική στα δικαιώματα των εφαρμογών, νέα APIs, εγγενή υποστήριξη για την αναγνώριση δακτυλικών αποτυπωμάτων και USB type-C, ένα νέο σύστημα για την διαχείριση ενέργειας που μειώνει την δραστηριότητα του παρασκηνίου εφόσον μια εφαρμογή δεν είναι σε χρήση, τη δυνατότητα μεταφοράς των δεδομένων και εφαρμογών σε μια κάρτα μνήμης microSD, και άλλες εσωτερικές αλλαγές. Η συγκεκριμένη έκδοση ποτέ δεν ολοκληρώθηκε και κατέληξε να αποσυρθεί από την εμπορική του δραστηριότητα όταν κυκλοφόρησε η καινούρια έκδοση το αμέσως επόμενο έτος.[60][61]

2.3.13 Android Nougat (version 7.0 – 7.1)

Το Android Nougat κυκλοφόρησε για ως δοκιμαστική έκδοση στις 9 Μαρτίου 2016 και στις 22 Αυγούστου 2016 ως επίσημη έκδοση. Αν και το LG V20 ήταν το πρώτο smartphone που κυκλοφόρησε με Nougat, οι συσκευές Nexus ήταν οι πρώτες που ενημερώθηκαν με το νέο λογισμικό. Το Nougat ενσωματώνει σημαντικές αλλαγές του λειτουργικού συστήματος και της πλατφόρμας ανάπτυξής του, συμπεριλαμβάνεται και η δυνατότητα εμφάνισης πολλαπλών εφαρμογών στην οθόνη (Multitasking) με μία προβολή που χωρίζει την οθόνη στην μέση, καθώς και το περιβάλλον Java που είναι βασισμένο στο OpenJDK αλλά και η υποστήριξη

απόδοσης γραφικών Vulkan API και ενημέρωσης συστήματος για υποστηριζόμενες συσκευές, ένας νέος τρόπος οργάνωσης ειδοποιήσεων και λειτουργίας για την εξοικονόμηση της μπαταρίας. Δύο μήνες μετά την κυκλοφορία του Nougat, ανακοινώθηκε το πρώτο αυτοτελές τηλέφωνο της Google, με ονομασία Pixel. Παράλληλα έγινε σημαντική βελτίωση στο Google Assistant, που θα συνεχίσει να είναι ένα σημαντικό στοιχείο του Android και είναι αναμφισβήτητα μία από τις μεγαλύτερες προσπάθειες της Google μέχρι και σήμερα.[60][61]

2.3.14 Android Oreo (version 8.0 – 8.1)

Το Android Oreo που κυκλοφόρησε στις 21 Αυγούστου του 2017, έκανε μια προσθήκη λεπτών χαρακτηριστικών, όπως την λειτουργία picture-in-picture, την επιλογή του χρήστη για αναβολή των ειδοποιήσεων και έναν νέο τρόπο που προσφέρει εξαιρετικό έλεγχο στον χρήστη για τον έλεγχο του τρόπου ειδοποίησης από τις εφαρμογών. Η κυκλοφορία αυτής της έκδοσης, συμπεριλάμβανε επίσης κάποια στοιχεία που είχαν ως σκοπό την ευθυγράμμιση του Android και του Chrome και την βελτίωση της εμπειρίας χρήσης των εφαρμογών Android σε Chromebook. Επιπλέον, έγινε για πρώτη φορά η απόπειρα να δημιουργηθεί μια αρθρωτή βάση του κώδικα του Android (Project Treble), ώστε να γίνονται έγκαιρες ενημερώσεις στο λογισμικό από τους κατασκευαστές.[60][61]

2.3.15 Android Pie (version 9.0)

Το Android 9 κυκλοφόρησε στις 6 Αυγούστου το 2018. Οι πιο σημαντικές αλλαγές στην καινούρια έκδοση, ήταν μια ακολουθία από εντολές που εκτελούνται με κάποιες συγκεκριμένες χειρονομίες και το πολύ-λειτουργικό Home button που αντικατέστησε τα 3 βασικά κουμπιά των προηγούμενων εκδόσεων. Το Android Pie περιέχει επίσης κάποια καινούρια χαρακτηριστικά, όπως μία μέθοδο για τη λήψη στιγμιότυπών (Screenshots), καλύτερα συστήματα εξοικονόμησης ενέργειας, έναν έξυπνο τρόπο για έλεγχο της φωτεινότητας και προτεινόμενες απαντήσεις μηνυμάτων. Επίσης, μικρές αλλά σημαντικές εξελίξεις έγιναν και στον αισθητήρα δακτυλικών αποτυπωμάτων χρησιμοποιώντας την αφή, της λειτουργίας για εξοικονόμηση μπαταρίας και στο χειρισμό του Wi-fi hotspot. Τέλος, το Android Pie περιέχει βελτιώσεις στην προστασία για τα προσωπικά δεδομένα και την ασφάλεια.[60][61]

2.3.16 Android Q (version 10.0)

Τον Μάρτιο του 2019, κυκλοφόρησε το Android Q, σε beta έκδοση για αρχή και υπήρξαν σταδιακές ενημερώσεις τον Απρίλιο και τον Μάιο. Οι πρώτες κυκλοφορίες αφορούσαν μόνο τα κινητά Pixel, συμπεριλαμβανομένων τις συσκευές Pixel και Pixel XL από πρώτη γενιά. Ένα από τα πιο σημαντικά χαρακτηριστικά του Q, ήταν η αντικατάσταση του ψηφιακού πλήκτρου Back σε έναν διαφορετικό τρόπο πλοήγησης ίδιου με των iphone (δηλαδή με ένα home button). Το Android Q εγκαθιστά μια νέα ρύθμιση για την ενημέρωση της ασφάλειας που θα επιτρέπει ταχύτερη και συνεπέστερη ενημέρωση σε όλο το σύστημα. Επίσης, παρέχει πολλές βελτιώσεις όσον αναφορά την παραχώρηση δικαιωμάτων σε εφαρμογές, ενισχύοντας την ασφάλεια της συσκευής. Τέλος, δίνετε η δυνατότητα να γίνει μια αλλαγή στις ρυθμίσεις του συστήματος χωρίς να χρειάζεται να κλείσει η εφαρμογή που είναι στην οθόνη εκείνη την στιγμή. Η επίσημη ημερομηνία κυκλοφορίας του ήταν στις 3 Σεπτεμβρίου 2019.[60][61]

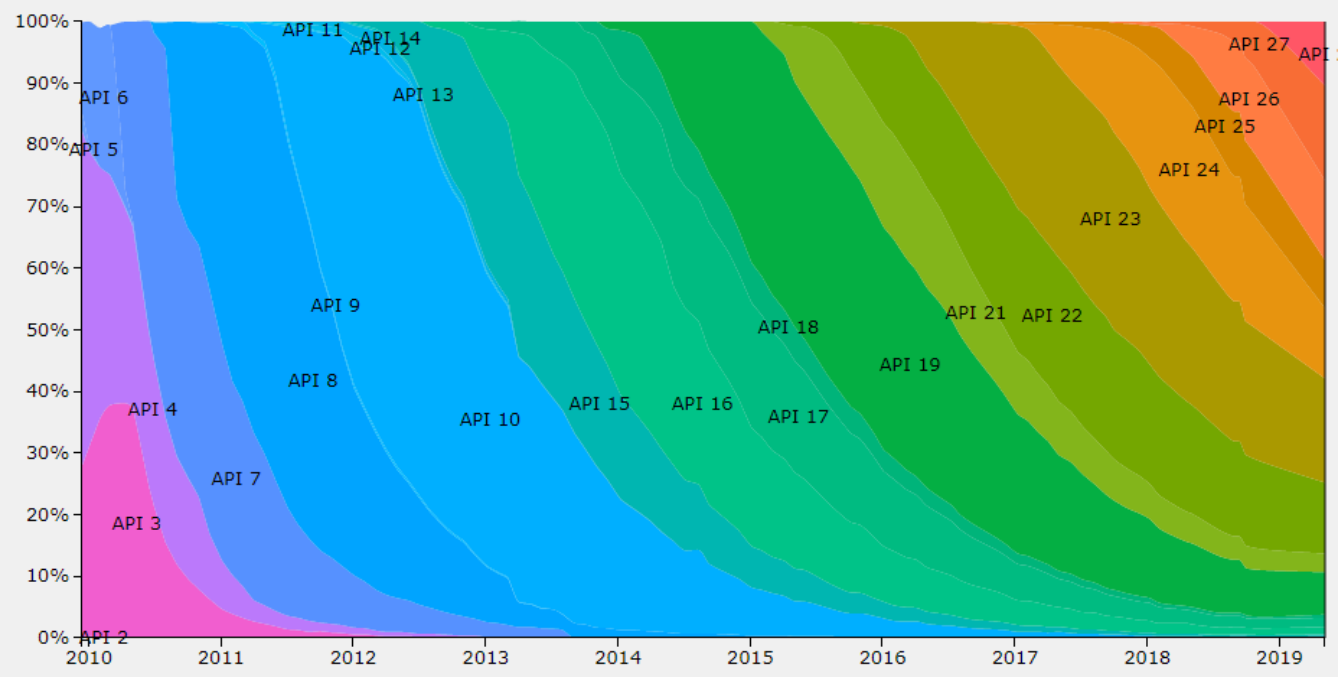
2.3.17 Android R (version 11.0)

Το Android R είναι η 11η αριθμητικά έκδοση και η 18η γενική έκδοση του λειτουργικού συστήματος Android. Κυκλοφόρησε στις 19 Φεβρουαρίου 2020 με επίπεδο API 30. Η Google, στις 4 Μαρτίου του 2020, κυκλοφόρησε την νέα έκδοση του Android R, την έκδοση 1.1. Δεν περιλάμβανε νέα χαρακτηριστικά, απλά επικεντρώθηκαν να διορθώσουν κάποια σφάλματα.[61]

Στατιστικός Πίνακας Διανομής Εκδόσεων Android

07-05-2019			
API	**Codename**	**Versions**	**Share**
10	Gingerbread	2.3.3 - 2.3.7	0.3%
15	Ice Cream Sandwich	4.0.3 - 4.0.4	0.3%
16	Jelly Bean	4.1	1.2%
17	Jelly Bean	4.2	1.5%
18	Jelly Bean	4.3	0.5%
19	KitKat	4.4	6.9%
21	Lollipop	5.0	3.0%
22	Lollipop	5.1	11.5%
23	Marshmallow	6.0	16.9%
24	Nougat	7.0	11.4%
25	Nougat	7.1	7.8%
26	Oreo	8.0	12.9%
27	Oreo	8.1	15.4%
28	Pie	9	10.4%

Πίνακας 2. Στατιστικός πίνακας διανομής εκδόσεων Android (2019)[62]



Εικόνα 5. Το ιστορικό διανομής μεταξύ των εκδόσεων Android (2009 – 2019)
(https://www.bidouille.org/misc/androidcharts)

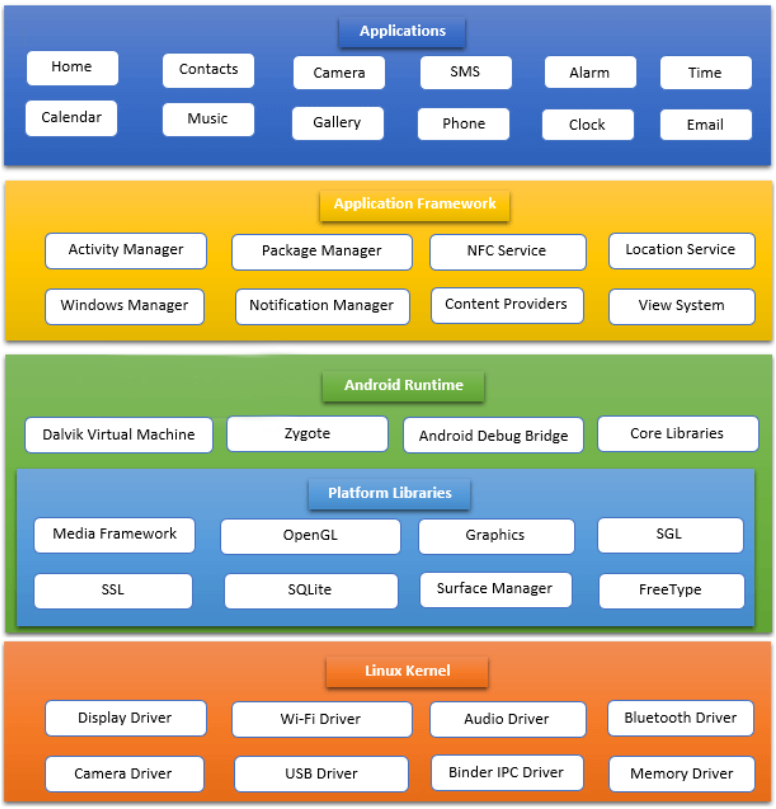
Συμπεράσματα

Σε γενικές γραμμές κάθε νέα έκδοση του Android έχει να προτείνει τις δικές της τεχνολογικές αναβαθμίσεις είτε αυτές υπόκεινται σε επίπεδο εμφάνισης και αισθητικής (UI), είτε αυτές υπόκεινται σε επίπεδο λογισμικού που δημιουργούν περισσότερες ανέσεις και επιλογές στον χρήστη στο να περιηγηθεί και να διαχειριστεί το περιβάλλον του Android. Στις παραπάνω παραγράφους αναφέρθηκαν περιληπτικά οι καινοτομίες που έφερε κάθε νέα έκδοση Android μέχρι τον Φεβρουάριο 2020. Το περιβάλλον χρήστη - user interface (UI), οι προεγκατεστημένες εφαρμογές, χαρακτηριστικά-features (όπως live wallpapers, φακός, widgets), Wi-Fi, NFC, Mobile Hotspot, VPN, Bluetooth, καλύτερη διαχείριση της μπαταρίας όπως και άλλες πολλές αναβαθμίσεις υλικολογισμικού έχουν έρθει σταδιακά με κάθε νέα έκδοση. Σημαντικό επίσης να αναφερθεί ότι στο προσκήνιο κάθε νέας έκδοσης του εν λόγο λογισμικού, υπήρχε η αναβάθμιση του Linux Kernel και Framework API, ορισμοί οι οποίοι θα αναφερθούν παρακάτω.

2.4 Επίπεδα Αρχιτεκτονικής Λειτουργικού Συστήματος Android

Η αρχιτεκτονική Android είναι μια στοίβα λογισμικού με επίπεδα που υποστηρίζουν τις ανάγκες των φορητών συσκευών. Η εν λόγω στοίβα λογισμικού Android περιέχει έναν πυρήνα Linux, συλλογή βιβλιοθηκών c / c++ που είναι διαθέσιμες μέσω υπηρεσιών πλαισίου εφαρμογών (API), χρόνου εκτέλεσης και εφαρμογής. Τα κύρια επίπεδα της αρχιτεκτονικής του Android όπως φαίνονται στην εικόνα που παρατίθεται αμέσως παρακάτω, είναι οι Εφαρμογές/Applications, τα Android Πλαίσια/Android Frameworks, ο Χρόνος

Εκτέλεσης/Android Runtime, οι Βιβλιοθήκες Πλατφόρμας/Platform Libraries και το Linux Kernel. Σε αυτά τα στοιχεία, το Linux Kernel είναι το βασικότερο επίπεδο του Android για να παρέχει τις λειτουργίες του λειτουργικού του συστήματος σε κινητό και το Dalvik Virtual Machine (DVM), το οποίο είναι υπεύθυνο για την εκτέλεση μιας εφαρμογής για κινητά.[63]



Εικόνα 6. Τα επίπεδα αρχιτεκτονικής Android
(https://www.tutlane.com/tutorial/android/android-architecture)

2.4.1 Applications / Εφαρμογές

Το κορυφαίο επίπεδο της αρχιτεκτονικής του Android είναι το επίπεδο Applications. Τα Applications των εγγενών και των τρίτων, όπως οι επαφές, το ηλεκτρονικό ταχυδρομείο, η μουσική, η γκαλερί, το ρολόι, τα παιχνίδια κ.λπ.. Ό,τι θα χτίσουμε, θα εγκατασταθούν σε αυτό το επίπεδο. Το επίπεδο Applications λειτουργεί εντός του χρόνου εκτέλεσης του Android χρησιμοποιώντας τις κλάσεις και τις υπηρεσίες που διατίθενται από το Application Framework.[63]

2.4.2 Application Framework / Πλαίσιο εφαρμογής (API)

Το Application Framework παρέχει τις κλάσεις που χρησιμοποιούνται για τη δημιουργία εφαρμογών Android. Παρέχει επίσης την δυνατότητα για την πρόσβαση στο υλικό και διαχειρίζεται τη διασύνδεση χρήστη και τους πόρους της εφαρμογής. Δηλαδή οι προγραμματιστές έχουν δικαίωμα πρόσβασης σε οποιοδήποτε API που χρησιμοποιούν οι ενσωματωμένες εφαρμογές. Παρέχει τις υπηρεσίες μέσω των οποίων μπορούμε να δημιουργήσουμε μια συγκεκριμένη κλάση και να την καταστήσουμε χρήσιμη, για τη δημιουργία μιας εφαρμογής. Η δομή των ενσωματωμένων εφαρμογών ευνοεί την επαναχρησιμοποίηση των δυνατοτήτων τους από άλλες εφαρμογές. Το Application Framework περιλαμβάνει υπηρεσίες όπως οι υπηρεσίες τηλεφωνίας, οι υπηρεσίες εντοπισμού θέσης, η διαχείριση ειδοποιήσεων, η υπηρεσία NFC, το σύστημα προβολής κ.λπ., που μπορούμε να χρησιμοποιήσουμε για την ανάπτυξη εφαρμογών σύμφωνα με τις απαιτήσεις μας.[63] Τα πιο σημαντικά δομικά χαρακτηριστικά του Application Framework είναι τα εξής:

- View System – Είναι ένα μεγάλο «πλαίσιο» από αντικείμενα GUI τα οποία δύναται να χρησιμοποιηθούν κατά την ανάπτυξη μιας εφαρμογής. Εν παραδείγματι, το πλαίσιο προβολών (GridView), πεδία για εισαγωγή text, κουμπιά, οι λίστες (listView), κλπ.
- Content Provider – Παρέχει την ευχέρεια στις εφαρμογές να μοιραστούν ή να ανταλλάξουν δεδομένα για μια συγκεκριμένη μορφή, η οποία καθορίζεται από τον πάροχο. Για παράδειγμα, δεδομένα είναι οι βάσεις δεδομένων των εφαρμογών και οι επαφές χρήστη.
- Resource Manager – Δίνεται η πρόσβαση σε «αντικείμενα» τα οποία δεν έχουν μορφή κώδικα όπως για παράδειγμα, πίνακες χαρακτήρων, εικόνες, αρχεία xml, κλπ.
- Notification Manager – Παρέχει στις εφαρμογές "access" στις υπηρεσίες ειδοποιήσεων χρήστη. Τέτοιες είναι, η δόνηση του κινητού και η ενεργοποίηση της οθόνης, οι ειδοποιήσεις στη μπάρα ειδοποιήσεων, τα pop μηνύματα στην κάτω πλευρά της οθόνης, κλπ.
- Activity Manager – Ελέγχει τον κύκλο ζωής των «activities» και δίνει την δυνατότητα μεταφοράς από activity σε activity κρατώντας στη μνήμη τη σειρά εκτέλεσης αυτών.

2.4.3 Android Runtime / Χρόνος εκτέλεσης

Το περιβάλλον Android Runtime είναι ένα σημαντικό κομμάτι του Android και περιέχει στοιχεία όπως, οι βασικές βιβλιοθήκες και η εικονική μηχανή Dalvik. Το Android Runtime είναι η κινητήρια δύναμη που εξάγει τις εφαρμογές μαζί με τις βιβλιοθήκες και αποτελεί τη βάση για το Application Framework. Το Virtual Machine Dalvik (DVM) είναι μια εικονική μηχανή που βασίζεται σε μητρώα όπως η Java Virtual Machine (JVM). Είναι ειδικά σχεδιασμένο και βελτιστοποιημένο για το Android για να διασφαλίσει ότι μια συσκευή μπορεί να τρέχει πολλές περιπτώσεις (instances) αποτελεσματικά. Βασίζεται στον πυρήνα του Linux για τη διαχείρηση νημάτων (threading) και τη διαχείριση μνήμης χαμηλού επιπέδου. Οι βασικές βιβλιοθήκες του Android Runtime, θα μας επιτρέψουν να υλοποιήσουμε εφαρμογές Android χρησιμοποιώντας την πρότυπη γλώσσα προγραμματισμού JAVA.[63]

2.4.4 Platform Libraries / Βιβλιοθήκες πλατφόρμας

Οι Platform Libraries περιλαμβάνουν διάφορες βιβλιοθήκες C / C++ και βιβλιοθήκες που βασίζονται σε Java όπως SSL, libc, γραφικά, SQLite, Webkit, Media, Surface Manager, OpenGL κ.λπ., για την υποστήριξη της ανάπτυξης του Android. Τα ακόλουθα, είναι οι συνοπτικές λεπτομέρειες ορισμένων βασικών βιβλιοθηκών που είναι διαθέσιμες για ανάπτυξη Android.[63]

- System C library: Μια ενσωμάτωση της βασικής βιβλιοθήκης συστήματος της C (libc) τροποποιημένη για κινητές συσκευές βασισμένες στο Linux.
- Media Library: Μία βιβλιοθήκη αναπαραγωγής και εγγραφής μορφών ήχου, βίντεο και εικόνων, όπως PNG, AMR, MP3, AAC, JPG, MPEG4 και H.264.
- Surface Manager: Για την διαχείριση της πρόσβασης στο υποσύστημα προβολής και σύνθεσης δισδιάστατων και τρισδιάστατων επιπέδων για γραφικά τα οποία προέρχονται από πολλαπλές εφαρμογές.
- SGL και OpenGL Graphics: Είναι υπεύθυνες για γραφικά 2D και 3D.
- SQLite: Μια σχεσιακή βάση δεδομένων.
- FreeType: Παρέχει ευκρίνεια στα γραφικά του bitmaps και στις γραμματοσειρές των εφαρμογών του συστήματος.
- Web-Kit: Για υποστήριξη του ενσωματωμένου web browser και SSL για ασφάλεια στο Internet.

2.4.5 Linux Kernel

Το Linux Kernel είναι το κατώτατο στρώμα και η καρδιά της αρχιτεκτονικής του Android. Διαχειρίζεται όλα τα προγράμματα οδήγησης (drivers), όπως προγράμματα οδήγησης οθόνης, προγράμματα οδήγησης κάμερας, προγράμματα οδήγησης Bluetooth, προγράμματα οδήγησης ήχου, προγράμματα οδήγησης μνήμης κ.λπ., τα οποία απαιτούνται κυρίως για τη συσκευή Android κατά τη διάρκεια εκκίνησης. Το Linux Kernel θα παρέχει ένα στρώμα επικοινωνίας μεταξύ του υλικού της συσκευής και της υπόλοιπης στοίβας. Είναι υπεύθυνο για τη διαχείριση μνήμης, τη διαχείριση ισχύος, τη διαχείριση συσκευών, την πρόσβαση σε πόρους κ.λπ..[63]

2.5 Αρχιτεκτονικά Μοντέλα Ανάπτυξης Εφαρμογών

Το πιο σημαντικό πράγμα κατά την έναρξη δημιουργίας νέας εφαρμογής είναι πρώτα η σκέψη της αρχιτεκτονικής. Το σημαντικότερο λάθος που μπορεί κάποιος να κάνει είναι να δημιουργήσει μία εφαρμογή, χωρίς μοντέλο-μοτίβο αρχιτεκτονικής. Το θέμα επιλογής μοντέλου αρχιτεκτονικής έγινε μείζον ζήτημα για την κοινότητα Android τα τελευταία χρόνια, που ακόμη και η Google αποφάσισε να συμμετάσχει, και το 2017 πρότεινε τη δική της προσέγγιση τυποποιημένης αρχιτεκτονικής, απελευθερώνοντας για πρώτη φορά τα Android Architecture Components που θα διευκολύνουν τη ζωή των προγραμματιστών, τα οποία θα αναφερθούν στο κεφάλαιο 2.7. Σε αυτή την ενότητα θα αναφερθεί γιατί πρέπει να σχεδιάζονται οι εφαρμογές σύμφωνα με κάποιο αρχιτεκτονικό μοτίβο και ποιες είναι οι επιλογές αυτές.[66]

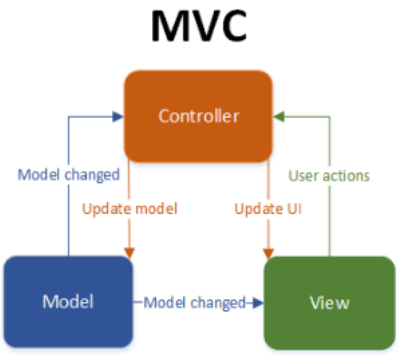
Οι λόγοι που χρειάζονται για μια καλή αρχιτεκτονική είναι για να αποφευχθούν κάποια συγκεκριμένα προβλήματα, όπως οι κώδικες να μην καλύπτονται από Test Units, να είναι δύσκολο να εντοπιστεί και να γίνει αποσφαλμάτωση (debug) κάποιας κλάσης επειδή περιέχει έναν τεράστιο αριθμό λειτουργιών, να μην είναι κατανοητή η παρακολούθηση της λογικής μέσα σε αυτήν την τεράστια κλάση και τέλος, θα είναι δύσκολο άλλοι προγραμματιστές να διαβάσουν τον κώδικα για να διατηρήσουν και να προσθέσουν νέες λειτουργίες στην εφαρμογή.

Τα πλεονεκτήματα που θα αποκομιθούν εφόσον χρησιμοποιηθεί μια σωστή αρχιτεκτονική, θα είναι η απλότητα του προγράμματος εφόσον θα είναι πιο εύκολο να πλοηγηθεί κάποιος μέσα στον κώδικα και να αναγνωρίσει τις λειτουργίες που του προσφέρει. Μπορεί επίσης να κάνει αλλαγές στο προγραμματιστικό μέρος με μεγαλύτερη άνεση. Τέλος, η συντήρηση χαμηλού κόστους είναι σαφέστατα το μεγαλύτερο πλεονέκτημα που χαρίζει ένα σωστό αρχιτεκτονικό μοντέλο, καθώς είναι εύκολο να προστεθούν και να αφαιρεθούν λειτουργίες και να παρατηρηθούν σημαντικά λογικά μέρη.[67]

Τα πιο γνωστά αρχιτεκτονικά μοντέλα είναι κλασικές αρχιτεκτονικές τριών επιπέδων όπως, το MVC (Model-View-Controller), το MVP (Model-View-Presenter), και το MVVM (Model-View-ViewModel). Όλα αυτά τα μοτίβα αντιπροσωπεύουν την κύρια παρόμοια ιδέα να δομήσουν τον κώδικα του έργου έτσι ώστε να διαχωρίζεται στα διαφορετικά γενικά επίπεδα. Κάθε επίπεδο έχει τη δική του ευθύνη, γι' αυτό το έργο γίνεται αρθρωτό, χωριστά τμήματα κώδικα, που μπορούν να δοκιμαστούν, και η εφαρμογή είναι αρκετά ευέλικτη για συνεχείς αλλαγές.[66]

2.5.1 MVC: Model-View-Controller

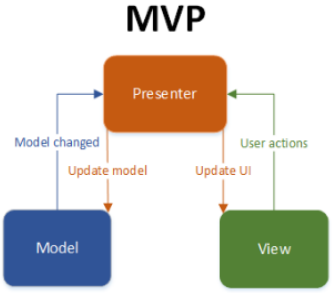
Αυτό το μοτίβο ήταν η πρώτη προσέγγιση της αρχιτεκτονικής εφαρμογών Android. Προτείνει να διαχωριστεί ο κώδικας σε 3 διαφορετικά επίπεδα. Το επίπεδο Model, που είναι το επίπεδο δεδομένων και είναι υπεύθυνο για το χειρισμό της επιχειρηματικής λογικής και της επικοινωνίας με τα επίπεδα δικτύου και βάσης δεδομένων. Το επίπεδο View που είναι το επίπεδο διεπαφής χρήστη (UI) και είναι μια απλή απεικόνιση των δεδομένων από το επίπεδο Model. Το επίπεδο Controller που είναι το λογικό επίπεδο και ενημερώνεται για τη συμπεριφορά του χρήστη και ενημερώνει το επίπεδο Model όπως απαιτείται. Το μοντέλο MVC είναι αρκετά συγκεχυμένο. Υπάρχουν διάφορες προσεγγίσεις για την εφαρμογή, του εν λόγω μοτίβου. Η πρώτη προσέγγιση είναι ότι τα Activities και τα Fragments ενεργούν όπως το επίπεδο Controller. Είναι υπεύθυνα να επεξεργάζονται τα δεδομένα και να ενημερώνουν το επίπεδο View. Το πρόβλημα με αυτήν την αρχιτεκτονική προσέγγιση είναι ότι τα Activities και τα Fragments μπορούν να γίνουν αρκετά μεγάλα και πολύ δύσκολο να ελεγχθούν. Μια δεύτερη προσέγγιση που φαίνεται πιο λογική και πιο σωστή είναι ότι τα Activities και τα Fragments πρέπει να είναι το επίπεδο View στον κόσμο του MVC. Τα Controllers θα πρέπει να είναι ξεχωριστές κλάσεις που δεν επεκτείνονται ή δεν χρησιμοποιούν οποιαδήποτε κλάση Android και το ίδιο ισχύει για τα Models.[66]



Εικόνα 7. MVC Architecture Pattern
(https://android.jlelse.eu/basic-concepts-of-software-architecture-patterns-in-android-c76e53f46cba)

2.5.2 MVP: Model-View-Presenter

Μετά την πρώτη προσέγγιση που δεν λειτούργησε, οι προγραμματιστές Android προσπάθησαν να χρησιμοποιήσουν ένα από τα πιο δημοφιλή αρχιτεκτονικά μοτίβα, το MVP. Αυτό το μοτίβο χρησιμοποιήθηκε ευρέως και εξακολουθεί να συνιστάται, ειδικά για όποιον ξεκινά την ανάπτυξη Android, καθώς είναι εύκολο να το μάθει κάποιος. Και αυτό το μοντέλο διαχωρίζεται σε 3 διαφορετικά επίπεδα. Το επίπεδο Model που είναι το επίπεδο δεδομένων, το οποίο είναι ίδιο με το μοτίβο MVC. Υπεύθυνο για το χειρισμό της επιχειρηματικής λογικής και της επικοινωνίας με τα επίπεδα δικτύου και βάσης δεδομένων. Το επίπεδο View που είναι το επίπεδο διεπαφής χρήστη (UI) που εμφανίζει τα δεδομένα και ειδοποιεί το επίπεδο Presenter για τις ενέργειες του χρήστη. Το επίπεδο Presenter που ανακτά τα δεδομένα από το Model, εφαρμόζει τη λογική διεπαφής (UI Logic) και διαχειρίζεται την κατάσταση του επιπέδου View, αποφασίζει τι θα εμφανίσει και αντιδρά στις ειδοποιήσεις εισαγωγής χρήστη από το επίπεδο View. Αυτό είναι ουσιαστικά το επίπεδο Controller από το MVC, εκτός του ότι δεν είναι καθόλου συνδεδεμένο με το επίπεδο View, απλώς είναι μια διεπαφή. Αυτό το μοτίβο έχει ένα σημαντικό αλλά ελεγχόμενο μειονέκτημα ότι το Presenter τείνει να επεκταθεί σε μια τεράστια γνωστή κλάση εάν δεν είμαστε αρκετά προσεκτικοί και δεν διαχωρίσουμε τον κώδικά μας. Ωστόσο, σε γενικές γραμμές, το μοτίβο MVP προσφέρει έναν πολύ καλό διαχωρισμό και με σιγουριά θα μπορούσε να είναι η κύρια επιλογή για δημιουργία εφαρμογών.[66]



Εικόνα 8. MVP Architecture Pattern
(https://android.jlelse.eu/basic-concepts-of-software-architecture-patterns-in-android-c76e53f46cba)

2.5.3 MVVM: Model-View-ViewModel

Το μοτίβο MVVM είναι η τρίτη προσέγγιση αρχιτεκτονικής, το οποίο έγινε προτεινόμενο πρότυπο αρχιτεκτονικής από την ομάδα Android με την έκδοση Android Architecture Components. **Να αναφερθεί ότι για την ανάπτυξη της εφαρμογής μας «Ιατρική Ατζέντα σε Android» χρησιμοποιήθηκε αυτό το αρχιτεκτονικό μοντέλο.** Τα επίπεδα που περικλείει το MVVM είναι, το επίπεδο Model, που αφαιρεί την πηγή δεδομένων, το επίπεδο ViewModel, που συνεργάζεται με το επίπεδο Model για τη λήψη και αποθήκευση των δεδομένων, το επίπεδο View, που ενημερώνει το επίπεδο ViewModel για τις ενέργειες των χρηστών και το επίπεδο ViewModel, που εκθέτει ροές δεδομένων που σχετίζονται με το επίπεδο View. Μία κύρια διαφορά σε σχέση με το μοτίβο MVP είναι ότι το MVVM, στο επίπεδο ViewModel δεν περιέχει αναφορά στο επίπεδο View όπως συμβαίνει με το Presenter. Στο μοτίβο MVVM το επίπεδο View έχει αναφορά στο επίπεδο ViewModel, αλλά το επίπεδο ViewModel δεν έχει καμία πληροφορία σχετικά με το επίπεδο View και υπάρχει σχέση πολλά-προς-ένα μεταξύ των View και ViewModel.[66]

Εικόνα 9. MVVM Architecture Pattern
(https://www.baruckis.com/android/kriptofolio-app-series-part-3/)

2.5.4 Σύγκριση MVC – MVP – MVVM

Μοντέλο/Pattern	Εξάρτηση από Android API	Πολυπλοκότητα XML	Δοκιμή Μονάδας	Ευελιξία αλλαγών δομής κώδικα
MVC	Υψηλή	Χαμηλή	Δύσκολη	Όχι
MVP	Χαμηλή	Χαμηλή	Καλή	Ναι
MVVM	Χαμηλή ή Καμία	Μέτρια ή Υψηλή	Τέλεια	Ναι

Πίνακας 3. Συγκριτικός πίνακας MVC-MVP-MVVM[68]

2.5.5 Συμπεράσματα

Τόσο το MVP όσο και το MVVM, προσφέρουν πιο εύπλαστη αρχιτεκτονική κώδικα συγκριτικά με το MVC. Ωστόσο, τείνουν επίσης να προσθέτουν περισσότερη πολυπλοκότητα στην εφαρμογή μας. Σε απλούστερες εφαρμογές που περιλαμβάνουν λιγότερο κώδικα, το MVC μπορεί να λειτουργήσει καλά στο Android. Ενώ σε πιο περίπλοκες περιπτώσεις όπου η εφαρμογή μας πρέπει να αναπτυχθεί λαμβάνοντας υπόψη την προσθήκη περισσότερων δυνατοτήτων στο μέλλον, το MVVM με τη χρήση του Data Binding θα μας κάνει να γράφουμε μικρότερο κώδικα. Για αυτόν ακριβώς τον λόγο, χρησιμοποιήσαμε το αρχιτεκτονικό μοτίβο MVVM στην εφαρμογή «Ιατρική Ατζέντα σε Android» που αναπτύξαμε εμείς. Στην πραγματικότητα, οι απαιτήσεις της εφαρμογής, ο χρόνος, η ομάδα λογισμικού, οι διαθέσιμοι πόροι κ.λπ., είναι παράγοντες οι οποίοι καθορίζουν και την επιλογή του αρχιτεκτονικού μοντέλου που θα χρησιμοποιηθεί, αλλά σαφώς ο καθένας μπορεί να επιλέξει όποιο επιθυμεί.

2.6 Δομικά Μέρη – Ανατομία Εφαρμογών

Ο σκοπός της ενότητας αυτής, είναι να προσφέρει μια εξοικείωση με τις έννοιες υψηλού επιπέδου, πίσω από την ανατομία των εφαρμογών Android. Με αυτόν τον τρόπο, θα διερευνηθούν λεπτομερώς, τόσο τα διάφορα στοιχεία που δύναται να χρησιμοποιηθούν για την δημιουργία μιας εφαρμογής, όσο και οι μηχανισμοί που επιτρέπουν τη συνεργασία για τη δημιουργία μιας εφαρμογής.

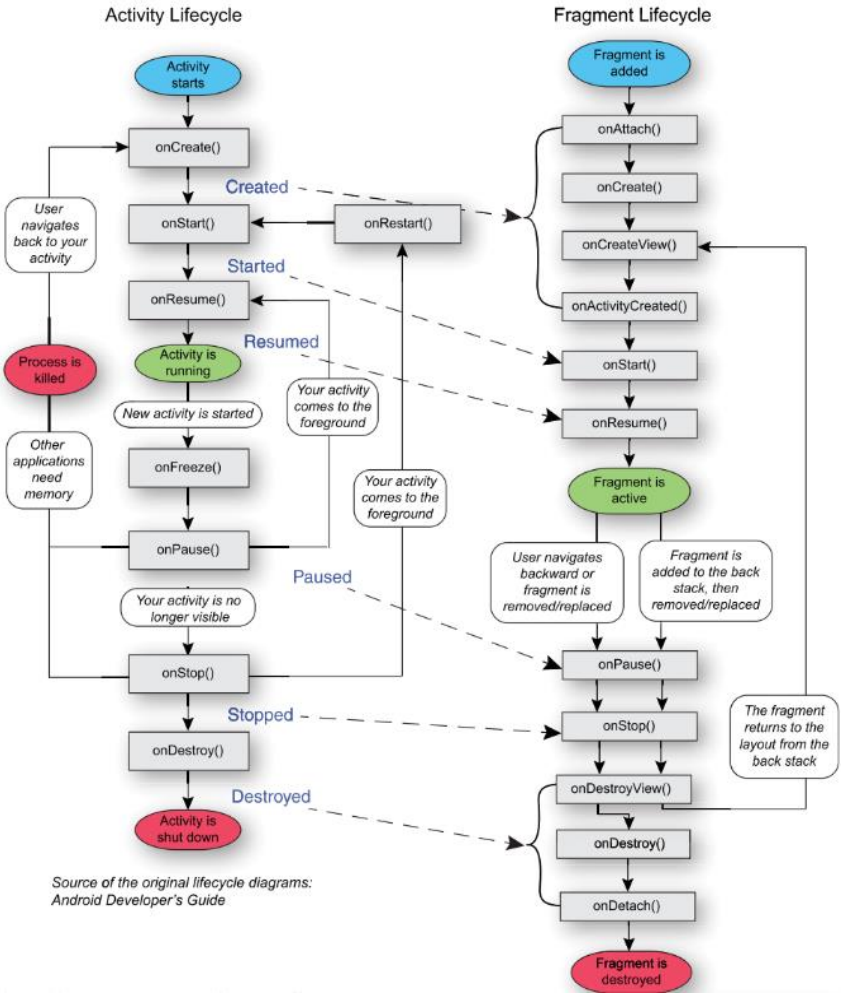
2.6.1 Android Activities

Οι εφαρμογές Android δημιουργούνται συνδυάζοντας ένα ή περισσότερα στοιχεία γνωστά ως Activities. Ένα Activity είναι μια μεμονωμένη ενότητα λειτουργικότητας εφαρμογών, που συνήθως σχετίζεται άμεσα με μια οθόνη διεπαφής χρήστη (User Interface) και την αντίστοιχη λειτουργικότητά της. Μια εφαρμογή για ραντεβού με γιατρούς μπορεί, για παράδειγμα, να έχει μια οθόνη δραστηριότητας (Activity) που εμφανίζει τα ραντεβού που έχουν ρυθμιστεί για την τρέχουσα ημέρα. Η εφαρμογή μπορεί επίσης να χρησιμοποιεί μια δεύτερη δραστηριότητα που αποτελείται από μια οθόνη όπου μπορούν να εισαχθούν νέα ραντεβού από τον χρήστη. Τα Activities προορίζονται ως πλήρως επαναχρησιμοποιήσιμα και εναλλάξιμα δομικά στοιχεία που μπορούν να μοιραστούν μεταξύ διαφορετικών εφαρμογών. Μια υπάρχουσα εφαρμογή email, για παράδειγμα, μπορεί να περιέχει μια δραστηριότητα ειδικά για τη σύνθεση και την αποστολή ενός email. Ένας προγραμματιστής μπορεί να συντάσσει μια εφαρμογή που έχει επίσης την υποχρέωση να στείλει ένα email. Αντί να αναπτύξει μια δραστηριότητα σύνθεσης email ειδικά για τη νέα εφαρμογή, ο προγραμματιστής μπορεί απλά να χρησιμοποιήσει τη δραστηριότητα από την υπάρχουσα εφαρμογή email. Τα Activities δημιουργούνται ως υποκλάσεις, από την κλάση Android Activity και πρέπει να υλοποιούνται έτσι ώστε να είναι εντελώς ανεξάρτητες από άλλα Activities στην εφαρμογή. Με άλλα λόγια, ένα κοινό Activity δεν μπορεί να βασιστεί στο να κληθεί σε ένα γνωστό σημείο σε μια ροή κώδικα (δεδομένου ότι άλλες εφαρμογές, μπορεί να κάνουν χρήση του Activity με διαφορετικούς τρόπους) και ένα Activity δεν μπορεί να καλέσει απευθείας μεθόδους ή να αποκτήσει πρόσβαση σε δεδομένα παρουσίας ενός άλλου Activity. Αυτό, αντίθετα, επιτυγχάνεται με τη χρήση Intents και Content Providers, όροι που θα αναλύσουμε παρακάτω. Από προεπιλογή, ένα Activity δεν μπορεί να επιστρέψει αποτελέσματα

στο Activity από το οποίο κλήθηκε. Εάν απαιτείται αυτή η λειτουργικότητα, το Activity πρέπει να ξεκινήσει συγκεκριμένα ως sub-Activity (υπο-δραστηριότητα) του αρχικού Activity.[68]

2.6.2 Android Fragments

Ένα Activity, όπως περιγράφεται παραπάνω, συνήθως αντιπροσωπεύει μια οθόνη διεπαφής χρήστη (User Interface) σε μια εφαρμογή. Μια επιλογή είναι να δημιουργηθεί το Activity χρησιμοποιώντας ένα User Interface layout και ένα αντίστοιχο Activity αρχείου κλάσης. Μια καλύτερη εναλλακτική λύση, ωστόσο, είναι να χωριστεί το Activity σε διαφορετικές ενότητες. Κάθε μία από αυτές τις ενότητες αναφέρεται ως Fragment, καθένα από τα οποία αποτελούν μέρος του User Interface layout και ενός αντίστοιχου αρχείου κλάσης (δηλώνεται ως υποκλάση της κλάσης Android Fragment). Σε αυτό το σενάριο, ένα Activity γίνεται απλά ένας χώρος στο οποίο ενσωματώνεται ένα ή περισσότερα Fragments. Στην πραγματικότητα, τα Fragments παρέχουν μια αποτελεσματική εναλλακτική λύση για την παρουσίαση κάθε οθόνης διεπαφής χρήστη από ένα ξεχωριστό Activity. Εναλλακτικά, μια εφαρμογή μπορεί να αποτελείται από ένα Activity που εναλλάσσεται μεταξύ διαφορετικών Fragments, το καθένα αντιπροσωπεύει μια διαφορετική οθόνη της εφαρμογής.[68]



Εικόνα 10. Activity & Fragment Lifecycle
(https://blog.avenuecode.com/android-basics-activities-fragments)

2.6.3 Android Intents

Το Intent παρέχει μια δυνατότητα για την εκτέλεση καθυστερημένης δέσμευσης χρόνου εκτέλεσης μεταξύ του κώδικα σε διαφορετικές εφαρμογές. Η πιο σημαντική χρήση του, είναι στην έναρξη των Activities, όπου μπορεί να θεωρηθεί ως η "κόλλα" μεταξύ των Activities. Είναι βασικά μια παθητική δομή δεδομένων που περιέχει μια αφηρημένη περιγραφή μιας δράσης που πρέπει να εκτελεστεί.[69]

2.6.4 Broadcast Intents

Ένας άλλος τύπος, είναι το Broadcast Intent, που είναι ένα Intent σε όλο το σύστημα, που αποστέλλεται σε όλες τις εφαρμογές που έχουν εισάγει έναν «ενδιαφερόμενο» Broadcast Receiver. Το σύστημα Android, για παράδειγμα, συνήθως αποστέλλει Broadcast Intents για να επισημάνει αλλαγές στην κατάσταση της συσκευής, όπως η ολοκλήρωση της εκκίνησης του συστήματος, η σύνδεση μιας εξωτερικής πηγής τροφοδοσίας με τη συσκευή, ή η οθόνη που ενεργοποιείται ή απενεργοποιείται. Ένα Broadcast Intent ή αποστέλλεται σε όλους τους ενδιαφερόμενους Broadcast Receivers περίπου ίδια στιγμή, ή αποστέλλεται σε έναν Receiver τη στιγμή που μπορεί να υποβληθεί σε επεξεργασία. Και στη συνέχεια είτε ακυρώνεται, είτε επιτρέπεται να περάσει στον επόμενο Broadcast Receiver.[68]

2.6.5 Broadcast Receivers

Οι Broadcast Receivers είναι ένας μηχανισμός με τον οποίο οι εφαρμογές μπορούν να ανταποκριθούν στα Broadcast Intents. Ένας Broadcast Receiver πρέπει να καταχωρηθεί από μια εφαρμογή και να διαμορφωθεί με ένα Intent Filter για να υποδείξει τους τύπους Broadcast που τον ενδιαφέρουν. Όταν ένα Intent κάνει Broadcast, ο Receiver θα κληθεί από το Android Runtime ανεξάρτητα από το εάν η εφαρμογή που καταχώρησε τον Receiver εκτελείται αυτήν τη στιγμή. Ο Receiver τότε, έχει 5 δευτερόλεπτα για να ολοκληρώσει οποιεσδήποτε διεργασίες απαιτούνται από αυτόν (όπως, έναρξη λειτουργίας μιας υπηρεσίας-Service, ενημέρωση δεδομένων ή εμφάνιση μιας ειδοποίησης στον χρήστη) πριν επανέρθει. Οι Broadcast Receivers λειτουργούν στο παρασκήνιο και δεν διαθέτουν διεπαφή χρήστη.[68]

2.6.6 Android Services

Τα Android Services είναι διαδικασίες που εκτελούνται στο παρασκήνιο και δεν διαθέτουν διεπαφή χρήστη. Μπορούν να ξεκινήσουν και στη συνέχεια να διαχειριστούν από τα Activities, τους Broadcast Receivers ή άλλα Services. Τα Android Services είναι ιδανικά για καταστάσεις όπου μια εφαρμογή πρέπει να συνεχίσει να εκτελεί διεργασίες, αλλά δεν χρειάζεται απαραίτητα ένα User Interface για να είναι ορατά στον χρήστη. Παρόλο που τα Services δεν διαθέτουν User Interface, εξακολουθούν να μπορούν να ειδοποιούν τον χρήστη για συμβάντα χρησιμοποιώντας ειδοποιήσεις και toasts (μικρά μηνύματα ειδοποίησης που εμφανίζονται στην οθόνη χωρίς να διακόπτεται η τρέχουσα ορατή δραστηριότητα). Τα Services έχουν υψηλότερη προτεραιότητα από το Android Runtime από ό, τι πολλές άλλες διαδικασίες

και θα τερματιστούν μόνο ως έσχατη λύση από το σύστημα προκειμένου να απελευθερωθούν πόροι. Σε περίπτωση που το Runtime χρειάζεται να τερματίσει ένα Service, ωστόσο, θα επανεκκινηθεί αυτόματα μόλις καταστούν διαθέσιμοι επαρκείς πόροι. Ένα Service μπορεί να μειώσει τον κίνδυνο τερματισμού του, δηλώνοντας ότι χρειάζεται να εκτελεστεί στο προσκήνιο. Αυτό επιτυγχάνεται κάνοντας μια κλήση στο startForeground(). Αυτό συνιστάται μόνο για καταστάσεις όπου ο τερματισμός θα ήταν επιζήμιος για την εμπειρία του χρήστη (για παράδειγμα, εάν ο χρήστης ακούει τον ήχο να μεταδίδεται από το Service). Παραδείγματα καταστάσεων όπου ένα Service μπορεί να είναι μια πρακτική λύση, αποτελούν, όπως αναφέρθηκε προηγουμένως, η ροή ήχου όπου θα πρέπει να συνεχιστεί, όταν η εφαρμογή δεν είναι πλέον ενεργή ή μια εφαρμογή παρακολούθησης τιμών από online-market, που πρέπει να ειδοποιήσει τον χρήστη όταν ένα προϊόν φτάσει σε μια καθορισμένη τιμή.[68]

2.6.7 Content Providers

Τα Content Providers εφαρμόζουν έναν μηχανισμό για την κοινή χρήση δεδομένων μεταξύ εφαρμογών. Οποιαδήποτε εφαρμογή μπορεί να προσφέρει σε άλλες εφαρμογές προσβασιμότητα στα δεδομένα της, μέσω της εφαρμογής ενός Content Provider, συμπεριλαμβανομένης της δυνατότητας, προσθήκης, αφαίρεσης και ερώτησης των δεδομένων της βάσης (βάσει αδειών). Η πρόσβαση στα δεδομένα δίνεται μέσω ενός Universal Resource Identifier (URI) που ορίζεται από τον Content Provider. Τα δεδομένα μπορούν να κοινοποιηθούν με τη μορφή αρχείου ή ολόκληρης της βάσης δεδομένων SQLite. Οι εφαρμογές Android περιλαμβάνουν έναν συγκεκριμένο αριθμό Content Provider, που δίνουν δικαίωμα στις εφαρμογές να έχουν πρόσβαση σε δεδομένα όπως επαφές και αρχεία πολυμέσων. Τα Content Providers που είναι διαθέσιμα επί του παρόντος σε ένα σύστημα Android, ενδέχεται να εντοπιστούν χρησιμοποιώντας έναν Content Resolver.[68]

2.6.8 The Application Manifest

Η συνδετικός κρίκος που συγκεντρώνει τα διάφορα στοιχεία που αποτελούν μια εφαρμογή είναι το Application Manifest. Σε αυτό το αρχείο που βασίζεται σε XML, η εφαρμογή περιγράφει τα Activities, τα Services, τα Broadcast Receivers, τα Content Providers και τα δικαιώματα που αποτελούν την ολοκληρωμένη εφαρμογή.[68]

2.6.9 Application Resources

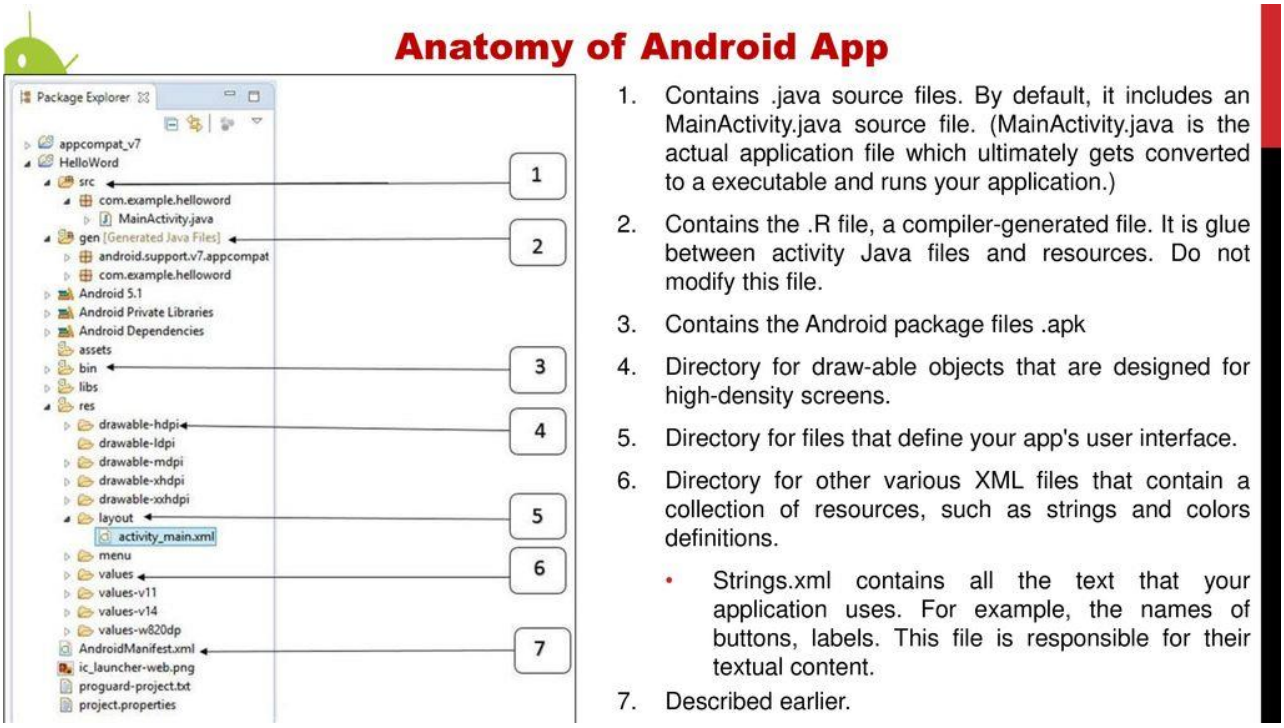
Εκτός από το Application Manifest, ένα πακέτο εφαρμογών Android θα περιέχει επίσης μια συλλογή αρχείων. Αυτά τα αρχεία περιέχουν Resources, όπως τις συμβολοσειρές, τις εικόνες, τις γραμματοσειρές και τα χρώματα που εμφανίζονται στη διεπαφή χρήστη μαζί με την αναπαράσταση XML των layout. Από προεπιλογή, αυτά τα αρχεία αποθηκεύονται στον υπο-κατάλογο / res της ιεραρχίας του project.[68]

2.6.10 Application Context

Όταν γίνεται compile μια εφαρμογή, δημιουργείται μια κλάση με το όνομα R, που περιέχει αναφορές του καταλόγου /res (Application Resources). Το αρχείο AndroidManifest.xml και τα Resources συγχωνεύονται για να δημιουργήσουν αυτό που είναι γνωστό, ως Application Context. Αυτό το Context, που αντιπροσωπεύεται από την κλάση Android Context, μπορεί να χρησιμοποιηθεί στον κώδικα της εφαρμογής για να αποκτήσει πρόσβαση στα Application Resources κατά το χρόνο εκτέλεσης της εφαρμογής. Επιπλέον, ένα ευρύ φάσμα μεθόδων μπορεί να ζητηθεί στο πλαίσιο μιας εφαρμογής για τη συλλογή πληροφοριών και την πραγματοποίηση αλλαγών στο περιβάλλον της εφαρμογής κατά το χρόνο εκτέλεσης.[68]

2.6.11 Συμπεράσματα

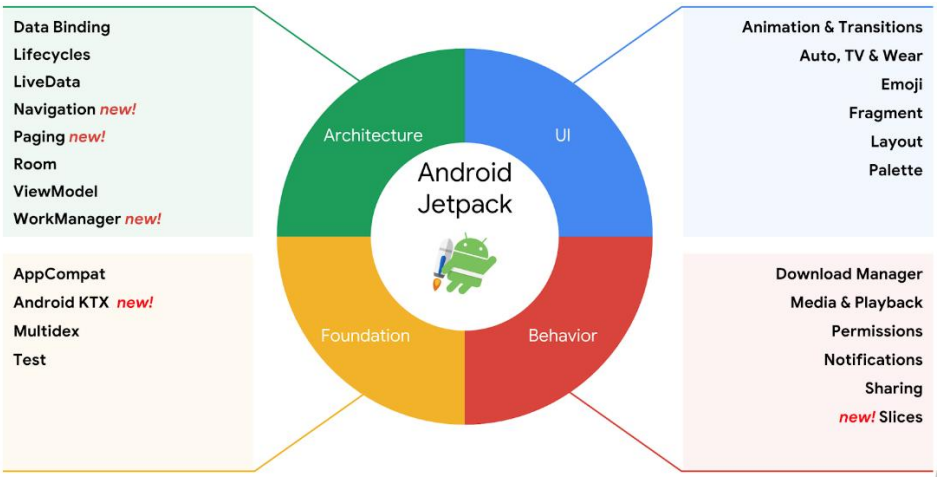
Σε αυτήν την ενότητα, παραθέσαμε μια επισκόπηση υψηλού επιπέδου των Activities, Fragments, Services, Intents και Broadcast Receivers, μαζί με μια επισκόπηση του αρχείου Manifest και των Resources μιας εφαρμογής. Η μέγιστη επαναχρησιμοποίηση και διαλειτουργικότητα προωθούνται μέσω της δημιουργίας μεμονωμένων, αυτόνομων ενοτήτων λειτουργικότητας, με τη μορφή Activities και Intents, ενώ η κοινή χρήση δεδομένων μεταξύ εφαρμογών επιτυγχάνεται με την εφαρμογή Context Providers. Ενώ τα Activities επικεντρώνονται σε περιοχές όπου ο χρήστης αλληλοεπιδρά με την εφαρμογή (ένα Activity που ουσιαστικά ισοδυναμεί με μια οθόνη διεπαφής χρήστη και συχνά αποτελείται από ένα ή περισσότερα Fragments), η επεξεργασία παρασκηνίου χειρίζονται από τα Services και τα Broadcast Receivers. Τα στοιχεία που συνθέτουν την εφαρμογή περιγράφονται για το Android Runtime System σε ένα αρχείο δήλωσης (Manifest) το οποίο, σε συνδυασμό με τα Resources, αντιπροσωπεύουν το περιβάλλον της εφαρμογής.



Εικόνα 11. Anatomy Of Android App
(https://slideplayer.com/slide/12814845/)

2.7 Σουΐτα Βιβλιοθηκών – Android Jetpack

Τα στοιχεία του Android Jetpack περιλαμβάνουν εκτός από τις κατηγορίες "Foundation", "Behavior" και "UI", την κατηγορία "Architecture", όπου είναι μια συλλογή από βιβλιοθήκες που υιοθετούνται και κατασκευάζονται μεμονωμένα για να συνεργαστούν με άλλα στοιχεία, αξιοποιώντας ταυτόχρονα τα χαρακτηριστικά της γλώσσας Kotlin. Θα αναλύσουμε σε αυτήν την ενότητα τα στοιχεία αρχιτεκτονικής που βοηθάνε να σχεδιάσουμε και να αναπτύξουμε ισχυρές εφαρμογές με την δυνατότητα δοκιμής και συντήρησης, μετά την κατασκευή τους.[64][65]



Εικόνα 12. Andoid Jetpack
(https://medium.com/@balakrishnanpt/android-jetpack-work-manager-de6909eb684d)

2.7.1 Data Binding

Το Data Binding είναι μια βιβλιοθήκη που επιτρέπει τη δέσμευση των στοιχείων UI στην διάταξη (layout) των δικών μας πηγών δεδομένων, χρησιμοποιώντας μια διακοσμητική μορφή. Αυτό επιτρέπει την αυτόματη ενημέρωση των τιμών UI με μία μορφή δήλωσης, χωρίς χειροκίνητη προγραμματιστική παρέμβαση, η οποία μειώνει αισθητά τον κώδικα του boilerplate της εφαρμογής.[64][65]

2.7.2 LifeCycle Awareness

Κάθε δραστηριότητα (Activity) ή κατακερματισμός (Fragment) σε μια εφαρμογή Android έχει έναν κύκλο ζωής (LifeCycle). Η κατάσταση του κύκλου ζωής της δραστηριότητας για κατακερματισμό, αλλάζει πάντα. Όταν μπορούμε να δημιουργήσουμε ξεχωριστές κλάσεις με LifeCycle Awareness (παρατήρηση κύκλου ζωής), μπορούμε να δημιουργήσουμε ξεχωριστές κλάσεις για να παρατηρήσουμε τις αλλαγές στην κατάσταση κύκλου ζωής της δραστηριότητας και να δράσουμε αναλόγως. Αυτό βοηθά στη δημιουργία πιο οργανωμένου και ελαφρύ κώδικα, που είναι ευκολότερο να διατηρηθεί.[64][65]

2.7.3 ViewModel

Το ViewModel είναι μια κλάση σχεδιασμένη για να αποθηκεύει και να διαχειρίζεται δεδομένα που σχετίζονται με το UI. Εάν απαιτείται, μπορούμε να δημιουργήσουμε ξεχωριστό ViewModel για κάθε Activity και Fragment στην εφαρμογή μας. Αυτή η κλάση έχει σχεδιαστεί για να αποθηκεύει και να διαχειρίζεται δεδομένα που σχετίζονται με το UI και να διατηρούν σταθερότητα στον κύκλο ζωής. Η κλάση ViewModel επιτρέπει δεδομένα σε αλλαγές διαμόρφωσης υπηρεσιών. Όπως η περιστροφή της οθόνης, αλλαγή γλώσσας πληκτρολογίου και η ενεργοποίηση της λειτουργίας πολλαπλών παραθύρων.[64][65]

2.7.4 Live Data

Το Live Data είναι μια κλάση για παρατήρηση δεδομένων. Μπορούμε να χρησιμοποιήσουμε την κλάση Live Data για να λάβουμε σε πραγματικό χρόνο ενημέρωση των χρηστών που χρησιμοποιούν τις διεπαφές (Interfaces). Το Live Data ενημερώνει στοιχεία εφαρμογών, που παρατηρούνται μόνο σε κατάσταση ενεργού κύκλου ζωής δραστηριότητας. Τα Live Data δεν αντικαθιστούν το RxJava, αλλά μας επιτρέπουν να εκτελέσουμε κάποια εργασία από την RxJava με απλούστερο τρόπο.[64][65]

2.7.5 Room Persistence Library

Το Room Persistence Library παρέχει ένα επίπεδο αφαιρετικότητας πάνω από το SQLite για να επιτρέψει την πιο ισχυρή πρόσβαση στην βάση δεδομένων ενώ εκμεταλλεύεται

την πλήρη ισχύ του SQLite. Το Room, μας βοηθάει να δημιουργήσουμε μια προσωρινή μνήμη (cached memory) δεδομένων μιας εφαρμογής, σε μια συσκευή που εκτελεί την εφαρμογή. Αυτή η κρυφή μνήμη επιτρέπει στους χρήστες να βλέπουν ένα συνεπές αντίγραφο βασικών πληροφοριών μέσα στην εφαρμογή, ανεξάρτητα από το αν οι χρήστες έχουν σύνδεση στο διαδίκτυο.[64][65]

2.7.6 Navigation

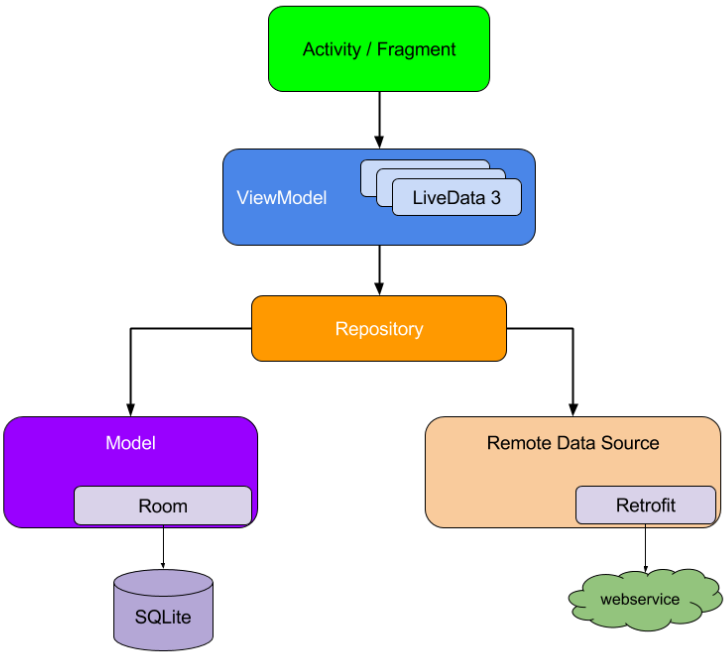
Το Navigation βοηθάει τους προγραμματιστές να σχεδιάσουν τις αλληλεπιδράσεις που επιτρέπουν στους χρήστες να μετακινηθούν, προς και από διαφορετικό περιεχόμενο εντός της εφαρμογής. Περιλαμβάνει την κλάση UI Navigation, η οποία περιέχει στατικές μεθόδους που διαχειρίζονται την πλοήγηση με το AppCompact bar, το Navigation Drawer και το Bottom Navigation.[64][65]

2.7.7 Work Manager

Υπάρχουν πολλές διεργασίες επεξεργασίας παρασκηνίου σε μια εφαρμογή Android, που φορτώνουν δεδομένα και συγχρονίζουν την βάση δεδομένων με απομακρυσμένο server. Μπορούμε να δημιουργήσουμε μια διεργασία παρασκηνίου και να την παραδώσουμε στο Work Manager, για να εκτελεστεί αμέσως ή την κατάλληλη στιγμή. Το Work Manager είναι ο πιο πρόσφατος και καλύτερος τρόπος για να εκτελέσουμε διεργασίες παρασκηνίου βασισμένες σε διαφορετικούς παράγοντες, όπως το επίπεδο API της συσκευής και η κατάσταση εφαρμογής.[64][65]

2.7.8 Paging Library

Η Paging Library μας βοηθάει να φορτώνουμε και να εμφανίζουμε μικρά κομμάτια δεδομένων κάθε φορά. Η φόρτωση μερικών δεδομένων στη ζήτηση, μειώνει τη χρήση του εύρους ζώνης δικτύου και των πόρων του συστήματος.[64][65]



Εικόνα 13. Συνδεσμολογία οντοτήτων μιας Android εφαρμογής (https://developer.android.com/jetpack/docs/guide)

3. Εργαλεία Υλοποίησης Πτυχιακής Εργασίας

Στο συγκεκριμένο κεφάλαιο, θα αναλυθεί γιατί το λειτουργικό σύστημα Android είναι κατάλληλο για την φιλοξενία της εφαρμογής «Ιατρική Ατζέντα» σε σχέση με άλλα λειτουργικά συστήματα, γιατί η προγραμματιστική γλώσσα Kotlin είναι η ιδανικότερη για την ανάπτυξη της σε σχέση με την Java, γιατί επιλέχθηκε να χρησιμοποιηθεί η Kotlin στο περιβάλλον του Android Studio και όχι σε κάποιο άλλο περιβάλλον, καθώς και η παρουσίαση του περιβάλλοντος του Android Studio.

3.1 Γιατί επιλέξαμε Android;

Το Android δεν είναι πλέον μόνο ένα λειτουργικό σύστημα. Έχει γίνει τρόπος ζωής, λόγω της εξαιρετικής δημοτικότητας και διαθεσιμότητάς του σε μια μεγάλη ποικιλία πλατφορμών. Ανεξάρτητα από τις συσκευές που χρησιμοποιούνται, είτε πρόκειται για smartphone, tablet είτε για smartwatch, θα βρεθεί ένα λειτουργικό σύστημα Android σε αυτό. Ο λόγος πίσω από την αποδοχή του είναι κυρίως η απρόσκοπτη λειτουργική εμπειρία που αποκτούν οι τελικοί χρήστες μετά τη χρήση του συστήματος. Επιπλέον, το λειτουργικό σύστημα Android είναι εξαιρετικά φιλικό προς το χρήστη (user-friendly) και εξαιρετικά προσαρμόσιμο, που λειτουργεί ως καταλύτης για την ακραία δημοτικότητά του σε όλο τον κόσμο. Τα πλεονεκτήματα που συναντούμε στο λειτουργικό σύστημα Android παρατίθενται παρακάτω.

Ανοιχτό Οικοσύστημα: Το λειτουργικό σύστημα Android δίνει πολλές επιλογές όσον αφορά την εγκατάσταση των εφαρμογών. Το Android διαθέτει ενσωματωμένο Google Play Store, το οποίο είναι το επίσημο κατάστημα για τη λήψη εφαρμογών "από το σπίτι της Google". Αλλά εκτός από αυτό, έχει επίσης την επιλογή να κατεβάσει ο κάθε χρήστης τις αγαπημένες του εφαρμογές από οποιοδήποτε κατάστημα εφαρμογών τρίτων. Μπορεί απλά να κατεβάσει μια συγκεκριμένη εφαρμογή από τον ιστότοπο προγραμματιστή και να τη μεταφέρει στη συσκευή του για να την χρησιμοποιήσει χωρίς προβλήματα. Παρόλο που υπάρχουν ορισμένοι κίνδυνοι εγκατάστασης εφαρμογών από ιστότοπο τρίτων, προειδοποιεί εκ των προτέρων πριν από την εγκατάσταση και πρέπει ο χρήστης να ενεργοποιήσει την επιλογή αδειοδότησης εφαρμογών τρίτων στη συσκευή του.

Διαφορετικές επιλογές τηλεφώνου: Σε αντίθεση με την Apple, όπου το λειτουργικό σύστημα IOS περιορίζεται μόνο σε χρήστες iPhone, το Android είναι διαθέσιμο ως λειτουργικό σύστημα σε διαφορετικά τηλέφωνα, από μια μεγάλη ποικιλία κατασκευαστών. Αυτό σημαίνει ότι μπορεί ο κάθε χρήστης να επιλέξετε το αγαπημένο του κινητό τηλέφωνο με βάση την προτίμηση της επωνυμίας του κατασκευαστή και να έχει προεγκατεστημένο το Android σύστημα σε αυτό. Επιπλέον, κάθε κατασκευαστής μπορεί να προσαρμόσει το UI του τηλεφώνου του με το δικό του στυλ. Για παράδειγμα, η Motorola διαθέτει Motoblur, η Sony Ericsson έχει Timescale και η MI έχει προεγκατεστημένο MIUI, για να βελτιώσει την εμπειρία του χρήστη.

Λειτουργικό σύστημα ανοιχτού κώδικα: Είναι ένα μεγάλο πλεονεκτήματα του λειτουργικού συστήματος Android. Καθώς ο πηγαίος κώδικας είναι ανοιχτός για όλους, οι προγραμματιστές και οι κατασκευαστές συσκευών μπορούν εύκολα να αποκτήσουν πρόσβαση στον πηγαίο κώδικα και να κάνουν τις απαραίτητες αλλαγές σύμφωνα με τη συμβατότητα του υλικού τους, εάν απαιτείται. Αυτό καθιστά το λειτουργικό σύστημα εξαιρετικά προσαρμόσιμο και προσανατολισμένο στην έρευνα. Ακόμη και η Google μπορεί να λάβει προτάσεις και σχόλια από προγραμματιστές, δοκιμαστές και κατασκευαστές συσκευών και να χρησιμοποιήσει αυτές τις αναφορές για τη βελτίωση του λειτουργικού συστήματος Android.

Προσαρμοσμένο ή τροποποιημένο Rom: Εάν κάποιος χρήστης έχει βαρεθεί να χρησιμοποιεί το stock Android και θέλει να έχει μερικές νέες και συναρπαστικές δυνατότητες, μπορεί εύκολα να εγκαταστήσει οποιοδήποτε προσαρμοσμένο ROM και να βελτιώσει την εμπειρία χρήστη του. Υπάρχουν πολλές εφαρμογές τρίτων που βοηθούν στη χρήση των προηγμένων λειτουργιών του Android, χρησιμοποιώντας ένα προσαρμοσμένο ROM. Αυτό είναι ένα εξαιρετικά χρήσιμο χαρακτηριστικό, επειδή επιτρέπει στον χρήστη να τροποποιήσει το stock Android παντού και μπορεί να αποκτήσει μια εντελώς νέα εμπειρία στην ίδια παλιά συσκευή του, εγκαθιστώντας ένα προσαρμοσμένο ROM. Έτσι, το λειτουργικό σύστημα Android, του επιτρέπει να κάνει πολλές τροποποιήσεις και βελτιώσεις σε επίπεδο συστήματος, που δεν θα ήταν ποτέ δυνατές σε άλλα λειτουργικά συστήματα όπως IOS ή Windows.

Φιλικό προς το χρήστη Play Store: Τόσο το Google play store όσο και το Apple store διαθέτει μια συλλογή από περισσότερες από 1 εκατομμύριο εφαρμογές που μπορούν να ληφθούν και να χρησιμοποιηθούν από τον χρήστη σύμφωνα με τις απαιτήσεις του. Ωστόσο, το Google play store είναι ευκολότερο στην χρήση και πιο φιλικό στον χρήστη σε σύγκριση με το App Store της Apple. Αυτό συμβαίνει επειδή η Apple είναι αρκετά αυστηρή κατά τη λήψη εφαρμογών και έχει πολύ περισσότερους περιορισμούς. Για παράδειγμα, ο χρήστης της Apple πρέπει να διαθέτει Apple iTunes για να κάνει λήψη και αναπαραγωγή ταινίας στη συσκευή

του. Από την άλλη πλευρά, το κατάστημα παιχνιδιών Android είναι πολύ πιο φιλικό προς το χρήστη με μια ανοιχτή διεπαφή που επιτρέπει στο χρήστη να αναπαράγει ταινίες σε οποιαδήποτε συσκευή με ένα απλό πρόγραμμα.

Εύκολη & προσιτή ανάπτυξη εφαρμογών: Η ανάπτυξη εφαρμογών είναι εξαιρετικά εύκολη και προσιτή όταν πρόκειται για Android. Μπορεί ο χρήστης εύκολα να αναπτύξει τη δική του εφαρμογή και να την καταχωρίσει στο play store, χωρίς να χρειάζεται να πληρώσει σημαντικά έξοδα ανάπτυξης. Το περιβάλλον εφαρμογών δεν χρεώνεται και έχει την επιλογή να αναπτύξει όσες εφαρμογές θέλει και να τις καταχωρίσει στο Android Play Store. Οι εφαρμογές εξακολουθούν να είναι διαχωρισμένες σε κατηγορίες όταν πρόκειται για εφαρμογές ψυχαγωγίας, εφαρμογές παιχνιδιών, εφαρμογές βιβλίων κ.λπ.. Δεν υπάρχει όριο ή περιορισμός στον αριθμό των εφαρμογών που αναπτύσσονται και κυκλοφορούν στην αγορά Android. Αυτό εξηγεί ότι ο χρήστης έχει την επιλογή να εξοικονομήσει πολλά χρήματα και να συνεχίσει να αναπτύσσει τις δικές του εφαρμογές στην πλατφόρμα Android.

Widgets: Widgets, ή αυτόνομα προγράμματα, είναι εξαιρετικά χρήσιμα για τη λειτουργικότητα. Τα widgets βοηθούν στην παροχή καλύτερης εμπειρίας χρήστη και εύκολων λειτουργιών πλοήγησης. Επιπλέον, προσθέτει ευελιξία στη συσκευή Android καθιστώντας την, φιλική προς το χρήστη και εύχρηστη. Όταν υπάρχει ένα widget απευθείας στην αρχική οθόνη, παρέχει στο χρήστη πρόσβαση και μπορεί χρησιμοποιεί διάφορες ρυθμίσεις και λειτουργίες με ένα μόνο άγγιγμα, το οποίο καθίσταται εξαιρετικά χρήσιμο όταν έχει λιγότερο χρόνο και χρειάζεστε τη συσκευή του να λειτουργεί πιο γρήγορα.

Επιλογή επέκτασης αποθηκευτικού χώρου: Όποτε θέλει ο χρήστης να αυξήσει τη χωρητικότητα αποθήκευσης οποιασδήποτε συσκευής Apple, όπως το iPhone ή το iPad, πρέπει να κάνει για μια δαπανηρή αναβάθμιση και έχει υπάρχει μεγάλη ταλαιπωρία. Αλλά στην περίπτωση συσκευών Android, ο χρήστης απλά εισάγει μια κάρτα μνήμης micro SD, στην υποδοχή κάρτας της συσκευής του και επεκτείνει τη μνήμη σύμφωνα με τις απαιτήσεις του. Αυτό είναι πραγματικά ένα πολύ χρήσιμο χαρακτηριστικό, που θα εντοπίσει στις περισσότερες συσκευές Android στην αγορά.[70]

3.2 Γιατί επιλέξαμε Kotlin;

Εικόνα 14. Λογότυπα Kotlin & Java
(https://appsoft.pro/kotlin-vs-java-what-is-better-for-android-development/)

3.2.1 Τι είναι η Java;

Η Java είναι μια προγραμματιστική γλώσσα OOP (αντικειμενοστρεφή γλώσσα προγραμματισμού) που άρχισε να χρησιμοποιείται το 1995. Η Java δημιουργήθηκε από την Sun Microsystems που αργότερα εξαγοράστηκε από την Oracle. Εάν θέλουμε να εκτελέσουμε ένα πρόγραμμα Java στο πρόγραμμα περιήγησης (web-browser), πρέπει να χρησιμοποιήσουμε εφαρμογές Java που είναι ενσωματωμένες ως plugins. Έτσι, η Java έχει χρησιμότητα ως επί το πλείστον για αυτόνομες desktop εφαρμογές ή back-end προγραμματισμό. Η Java σχεδιάστηκε από τον James Gosling και η βασική εφαρμογή της ήταν το OpenJDK. Η Java είναι η κύρια επιλογή για τους πιο πολλούς προγραμματιστές όσον αφορά την δημιουργία εφαρμογών Android, καθώς το ίδιο το Android είναι γραμμένο σε γλώσσα Java.[71]

3.2.2 Τι είναι η Kotlin;

Η Kotlin είναι μια νέα προγραμματιστική γλώσσα που αναπτύχθηκε από προγραμματιστές της IDE Jet Brains και χειρίζεται ορισμένα σύγχρονα χαρακτηριστικά. Εμφανίστηκε για πρώτη φορά το έτος 2011, η επίσημη κυκλοφορία της είναι το 2016 και είναι μια γλώσσα ανοιχτού κώδικα. Η Kotlin είναι επίσης μια στατικά δομημένη γλώσσα προγραμματισμού όπως οι Java, C++, η οποία είναι βασισμένη σε JVM (Java Virtual Machine), αλλά μπορεί να μεταγλωττιστεί και σε JavaScript, Android και Native, επίσης, για την κατασκευή κώδικα, και να εκτελείται σε λειτουργικό σύστημα iOS. Η Kotlin είναι πλήρως συμβατή με την Java, δηλαδή έναν κώδικα γραμμένο σε Java έχουμε την δυνατότητα να τον μετατρέψουμε σε Kotlin. Η μετατροπή αυτή είναι πολύ εύκολη καθώς απλά πρέπει να εγκαταστήσουμε ένα plugin. Κατά τη διάρκεια ενός εκ των συνεδρίων της Google I/O, ανακοινώθηκε ότι καθίσταται η Kotlin επίσημα υποστηριζόμενη γλώσσα για την ανάπτυξη εφαρμογών Android.[71]

3.2.3 Σύγκριση Java – Kotlin

Checked Exceptions: Μια σημαντική διαφορά μεταξύ των Java, Kotlin είναι ότι η τελευταία δεν έχει μηχανισμό για checked exceptions. Να αναφέρουμε ότι τα exceptions είναι μία μέθοδος εύρεσης σφαλμάτων, τα οποία δεν είναι κρίσιμα (critical) για την λειτουργία του προγράμματος κατά το compile ενός κώδικα, με αποτέλεσμα να μην αναγράφονται στο error section του compiler. Για παράδειγμα, αν ο προγραμματιστής έχει γράψει κάποιο κώδικα ο οποίος κάνει καταχωρήσεις σε μία βάση δεδομένων, τότε μπορούμε να χρησιμοποιήσουμε την μέθοδο try/catch για να δούμε αν όντως η βάση δεδομένων αποθηκεύει τις τιμές στα πεδία της. Ως εκ τούτου, δεν υπάρχει καμία ανάγκη να πιαστεί (catch) ή να δηλωθεί (declare) τυχόν exception . Εάν ένας προγραμματιστής που εργάζεται στην Java το θεωρεί ενοχλητικό να χρησιμοποιήσει try/catch στον κώδικα, τότε η παράλειψη που έκανε η Kotlin μπορεί να θεωρηθεί ευπρόσδεκτη αλλαγή.

Code Conciseness: Η σύγκριση μιας κλάσης Java με μια ισοδύναμη κλάση Kotlin καταδεικνύει τη συντομία του κώδικα Kotlin. Για την εκτέλεση της ίδιας λειτουργίας που κάνει η κλάση Java, μια κλάση Kotlin απαιτεί λιγότερο κώδικα. Για παράδειγμα, ένα συγκεκριμένο τμήμα όπου το Kotlin μπορεί να μειώσει σημαντικά τη συνολική ποσότητα του κώδικα boilerplate είναι το findViewByIds. Τα Android Extensions της Kotlin επιτρέπουν την εισαγωγή αναφοράς σε προβολή (View) στο αρχείο δραστηριότητας (Activity).

Coroutines: Οι εργασίες της CPU και το I/O δικτύου είναι χρονοβόρες λειτουργίες. Το νήμα που καλείται (calling thread) μπλοκάρεται έως ότου ολοκληρωθεί ολόκληρη η λειτουργία. Καθώς το Android λειτουργεί κάτω από την υπηρεσία ενός νήματος (single-threaded) από προεπιλογή, το περιβάλλον εργασίας χρήστη (UI) μιας εφαρμογής παγώνει εντελώς όσο το κύριο νήμα είναι μπλοκαρισμένο. Η κλασική λύση για το πρόβλημα αυτό από την Java είναι να δημιουργηθεί ένα background νήμα, για τη χρονοβόρα ή εντατική λειτουργία. Ωστόσο, η διαχείριση πολλών νημάτων οδηγεί σε αύξηση της πολυπλοκότητας, καθώς και σφαλμάτων στον κώδικα. Η Kotlin επιτρέπει επίσης τη δημιουργία πρόσθετων νημάτων. Ωστόσο, υπάρχει ένας καλύτερος τρόπος διαχείρισης εντατικών λειτουργιών στην Kotlin, γνωστός ως Coroutines. Τα Coroutines δεν λειτουργούν με την λογική μιας στοίβας, πράγμα που σημαίνει ότι απαιτούν χαμηλότερη χρήση μνήμης, σε σύγκριση με τα νήματα. Τα Coroutines είναι σε θέση να εκτελούν χρονοβόρες και εντατικές λειτουργίες, αναστέλλοντας την λειτουργία που μέλει να εκτελεσθεί χωρίς να μπλοκάρεται το νήμα και να συνεχίσουν την εκτέλεση κάποια στιγμή αργότερα. Επιτρέπει τη δημιουργία ασύγχρονου κώδικα χωρίς τον αποκλεισμό νημάτων. Ο κώδικας που χρησιμοποιεί Coroutines δεν είναι μόνο σαφής αλλά και συνοπτικός. Επιπλέον, τα Coroutines επιτρέπουν τη δημιουργία πρόσθετων μορφών ασύγχρονου προγραμματισμού όπως το async/await.

Data classes: Μεγάλες εφαρμογές έχουν πολλές κλάσεις που προορίζονται αποκλειστικά για τη διατήρηση δεδομένων. Παρόλο που αυτές οι κλάσεις έχουν ελάχιστη ή καθόλου λειτουργικότητα, ένας προγραμματιστής πρέπει να γράψει πολύ boilerplate κώδικα στην Java. Συνήθως, ένας προγραμματιστής χρειάζεται να ορίσει ένα κονστράκτορα (constructor), πολλά πεδία για να αποθηκεύσει τα δεδομένα, getter και setter συναρτήσεις για καθένα από τα πεδία, equals(), hashCode(), και toString() συναρτήσεις. Η Kotlin έχει έναν πολύ απλό τρόπο δημιουργίας τέτοιων κλάσεων. Ο προγραμματιστής πρέπει πολύ πιο απλά να περιλαμβάνει μόνο τη λέξη-κλειδί δεδομένων στον ορισμό της κλάσης και ο μεταγλωττιστής θα φροντίσει μόνος του όλη την διεργασία.

Higher-Order Functions and Lambdas: Μια συνάρτηση υψηλότερης προτεραιότητας, είναι εκείνη που λαμβάνει συναρτήσεις ως παραμέτρους ή επιστρέφει μια συνάρτηση. Επίσης, οι συναρτήσεις της Kotlin είναι πρώτης κλάσης. Αυτό σημαίνει ότι μπορούν να αποθηκευτούν σε δομές δεδομένων και μεταβλητές, οι οποίες μπορούν να μεταδοθούν ως ορίσματα και να επιστραφούν από άλλες συναρτήσεις υψηλότερης προτεραιότητας. Ως στατική γλώσσα προγραμματισμού, η Kotlin χρησιμοποιεί ένα εύρος τύπων συναρτήσεων για την αναπαράσταση άλλων συναρτήσεων. Επιπλέον, έρχεται με ένα σύνολο εξειδικευμένων γλωσσικών κατασκευών, όπως οι εκφράσεις Lambdas. Οι ανώνυμες συναρτήσεις και οι εκφράσεις Lambda, είναι επίσης γνωστές ως λεξιλόγια συνάρτησης. Αυτές είναι συναρτήσεις που δεν δηλώνονται, αλλά μεταβιβάζονται αμέσως ως έκφραση.

Implicit Widening Conversions: Δεν υπάρχει υποστήριξη για έμμεσες μετατροπές αριθμών στην Kotlin. Έτσι, οι μικρότεροι τύποι, δεν μπορούν να μετατραπούν σε μεγαλύτερους τύπους. Ενώ η Java έχει υποστήριξη για έμμεσες μετατροπές, η Kotlin απαιτεί να εκτελέσει μια άμεση μετατροπή για επιτύχει αυτή την μετατροπή.

Inline Functions: Οι μεταβλητές που έχουν πρόσβαση στην συνάρτηση είναι γνωστές ως closures. Η χρήση λειτουργιών υψηλότερης προτεραιότητας μπορεί να επιβάλει αρκετές κυρώσεις χρόνου εκτέλεσης. Κάθε συνάρτηση στην Kotlin είναι ένα αντικείμενο και καταγράφει ένα closure. Τόσο οι κλάσεις όσο και οι συναρτήσεις αντικειμένων απαιτούν εκχώρηση μνήμης. Αυτές μαζί με εικονικές κλήσεις εισάγουν γενικά το χρόνο εκτέλεσης. Μια

τέτοια επιπλέον επιβάρυνση μπορεί να αποφευχθεί με την κλίση των εκφράσεων Lambda στην Kotlin. Ένα τέτοιο παράδειγμα είναι η συνάρτηση lock(). Σε αντίθεση με το Kotlin, η Java δεν παρέχει υποστήριξη για ενσωματωμένες συναρτήσεις.

Native Support for Delegation: Στην ορολογία προγραμματισμού, το Delegation αντιπροσωπεύει τη διαδικασία όπου ένα αντικείμενο μεταβιβάζει τις λειτουργίες του σε ένα δεύτερο αντικείμενο. Η Kotlin υποστηρίζει τη σύνθεση του κληρονομικού σχεδιασμού μέσω της πρώτης κλάσης, γνωστή και ως έμμεση εκχώρηση. Το class Degelation είναι μια εναλλακτική λύση για την κληρονομικότητα στην Kotlin. Αυτό καθιστά δυνατή τη χρήση πολλαπλών κληρονομικοτήτων.

Null Safety: Ένα από τα πιο ενοχλητικά ζητήματα που αφορούν την Java για προγραμματιστές είναι το NullPointerExceptions. Η Java επιτρέπει στους προγραμματιστές να εκχωρούν μια τιμή null σε οποιαδήποτε μεταβλητή. Ωστόσο, εάν προσπαθήσουν να χρησιμοποιήσουν μια αναφορά αντικειμένου που έχει μηδενική τιμή, έρχεται το NullPointerException! Σε αντίθεση με την Java, όλοι οι τύποι είναι μηδενικοί στην Kotlin από προεπιλογή. Εάν οι προγραμματιστές προσπαθήσουν να εκχωρήσουν ή να επιστρέψουν μηδενικά στον κώδικα Kotlin, θα αποτύχει κατά τη στιγμή της μεταγλώττισης. Ωστόσο, υπάρχει ένας τρόπος, για να εκχωρήθεί μια τιμή null σε μια μεταβλητή στην Kotlin, απαιτείται να επισημανθεί ρητά αυτή η μεταβλητή ως nullable. Αυτό γίνεται προσθέτοντας ένα ερωτηματικό μετά τον τύπο, για παράδειγμα: val number: Int? = null. Επομένως, δεν υπάρχει NullPointerException στην Kotlin.

Primitive Types: Υπάρχουν 8 τύποι δεδομένων, συμπεριλαμβανομένων των char, double, float και int. Σε αντίθεση με την Kotlin, μεταβλητές ενός τύπου δεν είναι αντικείμενα σε Java. Αυτό σημαίνει ότι δεν αποτελούν αντικείμενο που δημιουργείται από μια κλάση ή μια δομή.

Smart Casts: Προτού μπορέσει να μεταδώθεί ένα αντικείμενο σε Java, είναι υποχρεωτικό να ελεγχτεί ο τύπος. Σε αντίθεση με την Java, το Kotlin έρχεται με τη δυνατότητα smart cast, η οποία χειρίζεται αυτόματα τέτοια περιττά cast. Δεν χρειάζεται να χρησιμοποιηθεί μια δήλωση, με την προϋπόθεση ότι έχει ήδη ελεγχθεί με το «is operator» στην Kotlin.

Static Members: Η Kotlin δεν έχει καμία πρόβλεψη για στατικά μέλη. Ωστόσο, στη γλώσσα προγραμματισμού Java η λέξη-κλειδί «static» αντικατοπτρίζει ότι το συγκεκριμένο μέλος με το οποίο χρησιμοποιείται η λέξη-κλειδί ανήκει σε έναν ίδιο τύπο αντί για μια παρουσία αυτού του τύπου. Σημαίνει απλώς ότι μία και μόνο μία παρουσία αυτού του στατικού μέλους δημιουργείται και κοινοποιείται σε όλες τις εμφανίσεις της κλάσης.

Support for Constructors: Μια κλάση Kotlin, σε αντίθεση με μια κλάση Java, μπορεί να έχει έναν ή περισσότερους δευτερεύοντες constructors εκτός από έναν κύριο constuctor. Αυτό γίνεται συμπεριλαμβάνοντας αυτούς τους δευτερεύοντες constructors στη δήλωση κλάσης.

Ternary Operator: Σε αντίθεση με την Kotlin, η Java διαθέτει έναν τριαδικό τελεστή (Ternary Operator). Ο τριαδικός τελεστής της Java λειτουργεί απλώς σαν μια βασική δήλωση if. Αποτελείται από μια συνθήκη που αξιολογείται ως αληθής ή ψευδής. Επιπλέον, ο τριαδικός τελεστής Java έχει δύο τιμές. Μόνο μία από αυτές επιστρέφεται, ανάλογα με το αν η κατάσταση είναι αληθής ή ψευδής. Η σύνταξη για τον τριαδικό τελεστή Java είναι: (condition) ? (value1) : (value 2)

Wildcard Types: Σε γενικό κώδικα, το "?" αντιπροσωπεύει έναν άγνωστο τύπο. Είναι γνωστό ως Wildcard. Υπάρχουν πολλές χρήσεις ενός Wildcard, συμπεριλαμβανομένου και του τύπου ενός πεδίου, της τοπικής μεταβλητής ή παραμέτρου. Ενώ το σύστημα τύπου Java προσφέρει Wildcard τύπους, ενώ η Kotlin δεν το προσφέρει. Ωστόσο, έχει δύο διαφορετικά πράγματα. Declaration-site variance και type projections ως εναλλακτική λύση για τους τύπους Wildcards.

Χαρακτηριστικά	Java	Kotlin
Checked Exceptions	✓	X
Code Conciseness	Εκτενής κώδικας	Σύντομος κώδικας
Coroutines	X	✓
Data Classes	Απαιτείται για τη σύνταξη πολλή κώδικα boilerplate	Απαιτείται προσθήκη μόνο της λέξης-κλειδιού δεδομένων στον ορισμό κλάσης
Extension Functions	X	✓
Higher-Order Functions and Lambdas	Εφαρμόζονται χρησιμοποιώντας Callables. Οι εκφράσεις Lambdas εισάγονται στο Java 8	Είναι ένα από τα προκαθορισμένα χαρακτηριστικά
Implicit Widening Conversions	✓	X
Inline Functions	X	✓
Native Support for Delegation	X	✓
NullPointerExceptions	✓	X
Primitive Types	Δεν είναι αντικείμενα	Είναι αντικείμενα
Smart Casts	X	✓
Static Members	✓	X
Support for Constructors	Δεν είναι δυνατή η δημιουργία δευτερευόντων Constructors	Μπορεί να έχει έναν ή περισσότερους δευτερεύοντες Constructors
Ternary Operator	✓	X

Wildcard Types	✓	X Έχει εναλλακτική δυνατότητα declaration-site variance και types ως εναλλακτική λύση

Πίνακας 4. Συγκριτικός πίνακας Java-Kotlin[72]

3.2.4 Συμπεράσματα

Χωρίς αμφιβολία, υπάρχουν πολλά χαρακτηριστικά στην Kotlin που την καθιστούν πρακτική. Αν και η Kotlin δεν είναι κτήμα της Google, θα συνεχίσει να αναπτύσσεται από την προγραμματιστική ομάδα JetBrains. Όπως είδαμε παραπάνω, η Java και η Kotlin έχουν τις δικές τους αδυναμίες και δυνατότητες η κάθε μία ξεχωριστά. Παρ' όλ' αυτά, για δημιουργία εφαρμογών Android, η Kotlin έχει το πάνω χέρι, καθώς προσφέρει αναμφισβήτητα πιο συνοπτικό και οργανωμένο κώδικα, με αποτέλεσμα να μειώνεται αισθητά ο χρόνος υλοποίησης μίας εφαρμογής. Με αυτή την λογική, προχωρήσαμε και εμείς στην υλοποίηση του θέματος της Π.Ε. μας στη προγραμματιστική γλώσσα Kotlin.

3.3 Γιατί επιλέξαμε Android Studio;

Σε αυτή την ενότητα θα αναλύσουμε γιατί καταλήξαμε στην επιλογή της πλατφόρμας Android Studio για την ανάπτυξη της Android εφαρμογής μας, σε γλώσσα Kotlin. Επίσης, θα κάνουμε μία σύντομη σύγκριση μεταξύ του Android Studio και του Eclipse IDE, που είναι ανάμεσα από όλα τα περιβάλλοντα ανάπτυξης εφαρμογών, οι δύο επικρατέστεροι κολοσσοί.

Εικόνα 15. Android Studio logo
(https://techcrunch.com/2017/02/19/why-is-android-studio-still-such-a-gruesome-embarrassment/)

3.3.1 Σύγκριση Android Studio – Eclipse

Το Eclipse είναι, εδώ και χρόνια, το προτιμώμενο ολοκληρωμένο περιβάλλον ανάπτυξης (IDE) για προγραμματιστές Android. Ωστόσο, αυτή τη στιγμή αντιμετωπίζει σοβαρό ανταγωνισμό από το Android Studio της Google. Υπάρχει μια γενική αίσθηση, ότι το Android Studio σταδιακά «εκλείπει» τη δημοτικότητα του Eclipse, χάρη στη περισσότερα φιλική, προς τον χρήστη φύση του. Ας αναλύσουμε μερικά από τα βασικά χαρακτηριστικά των παραπάνω περιβαλλόντων ανάπτυξης εφαρμογών.[73]

Διεπαφή χρήστη (UI): Το Eclipse, σε αντίθεση με το Android Studio, δεν είναι προγραμματισμένο για Android IDE (είναι προγραμματισμένο για Java IDE και είναι συμβατό με πολλές πλατφόρμες). Ως εκ τούτου, είναι γεγονός ότι η συνολική διάταξη της διεπαφής (κουμπιά, καρτέλες και γενική πλοήγηση) είναι λίγο πιο περίπλοκη από αυτήν του Android Studio. Οι προγραμματιστές εφαρμογών Android, βρίσκουν το Android Studio IDE πιο εύκολο και πιο γρήγορο. Επιπλέον, το Android Studio δημιουργήθηκε συγκεκριμένα για Android, ενώ το Eclipse δημιουργήθηκε για IDE, για όλες τις χρήσεις που μπορεί να χρησιμοποιηθεί με οποιαδήποτε γλώσσα και πλατφόρμα.[73]

Apache Ant vs Gradle: Οι προγραμματιστές Java με τουλάχιστον κάποια εμπειρία, θα ήταν εξοικειωμένοι με το σύστημα κατασκευής Apache Ant (εργαλείο για την αυτοματοποίηση των διεργασιών δημιουργίας λογισμικού), με το οποίο έρχεται το Eclipse (μέσω ενός plugin). Το Ant βασίζεται σε XML και κατατάσσεται ψηλά στην αξιοπιστία και στις αποδόσεις. Παρόλα αυτά, το ολοκαίνουργιο σύστημα κατασκευής Gradle (σύστημα αυτοματοποίησης δημιουργίας ανοιχτού κώδικα) του Android Studio όπου βασίζεται στο Groovy αντί για την φόρμα XML, εμφανίζεται ως πιο προηγμένο και προσφέρει μεγαλύτερες ανέσεις στους προγραμματιστές εφαρμογών για κινητά. Το Gradle χρησιμοποιεί ένα κατευθυνόμενο ακυκλικό γράφημα για να προσδιορίσει τη σειρά με την οποία μπορούν να εκτελεστούν οι διεργασίες. Υπάρχουν ακόμη και επιλογές αυτοματοποίησης για τη μεταφόρτωση της έκδοσης beta των αρχείων *.apk* στο TestFlight (προσομοιωτή της εφαρμογής). Πράγμα το οποίο δεν είναι δυνατό με το Eclipse.[73]

Προηγμένη ολοκλήρωση κώδικα (Advanced Code Completion): Τόσο το Android Studio όσο και το Eclipse διαθέτουν την τυπική αυτόματη συμπλήρωση κώδικα Java. Ωστόσο, στο Android Studio η ολοκλήρωση του κώδικα είναι πολύ καλύτερη σε σύγκριση με το Eclipse, που φαίνεται να μπερδεύεται σε μερικά σημεία και δεν παρέχει ακριβή αποτελέσματα τις περισσότερες φορές.[74]

Σταθερότητα συστήματος (System Stability): Το Eclipse είναι λογισμικό που βασίζεται σε Java και σε μεγαλύτερο IDE σε σύγκριση με το Android Studio, οπότε χρειάζεται πολύ υψηλότερος χώρος RAM με υψηλή ταχύτητα CPU για να λειτουργεί σωστά. Η μη τήρηση αυτού του κριτηρίου προκαλεί το Eclipse να καταρρέει και να μην ανταποκρίνεται. Από την άλλη πλευρά, το Android Studio κυκλοφορεί τώρα με πολύ λιγότερα σφάλματα και παρέχει μια πιο σταθερή εγγύηση απόδοσης, από ότι το Eclipse και οι ανάγκες του συστήματος είναι επίσης χαμηλότερες. Τέλος, ενώ στο Eclipse για ένα σύνθετο προγραμματιστικό έργο χρειάζονται περίπου 1 με 2 λεπτά για να φορτωθεί, για το ίδιο προγραμματιστικό έργο, στο Android Studio με το ίδιο σύστημα χρειάζονται περίπου 30 δευτερόλεπτα για την φόρτωση του.[74]

Drag and Drop: Το Android Studio διαθέτει GUI (γραφικό περιβάλλον εργασίας χρήστη), αλλά το Eclipse δεν διαθέτει. Ωστόσο, η λειτουργία drag and drop δεν είναι απαραίτητη για τους προγραμματιστές, οι οποίοι δεν ενδιαφέρονται πολύ για τα οπτικά στοιχεία των εφαρμογών τους. Ένας προγραμματιστής πρέπει να έχει λεπτομερείς γνώσεις της Visual Basic, έτσι ώστε ο προγραμματιστής να μπορεί να χρησιμοποιήσει κατάλληλα τη δυνατότητα drag and drop. Είναι μια νέα δυνατότητα στο Android Studio, αλλά η κατάσταση της απουσίας του στο Eclipse δεν έχει μεγάλη σημασία.[74]

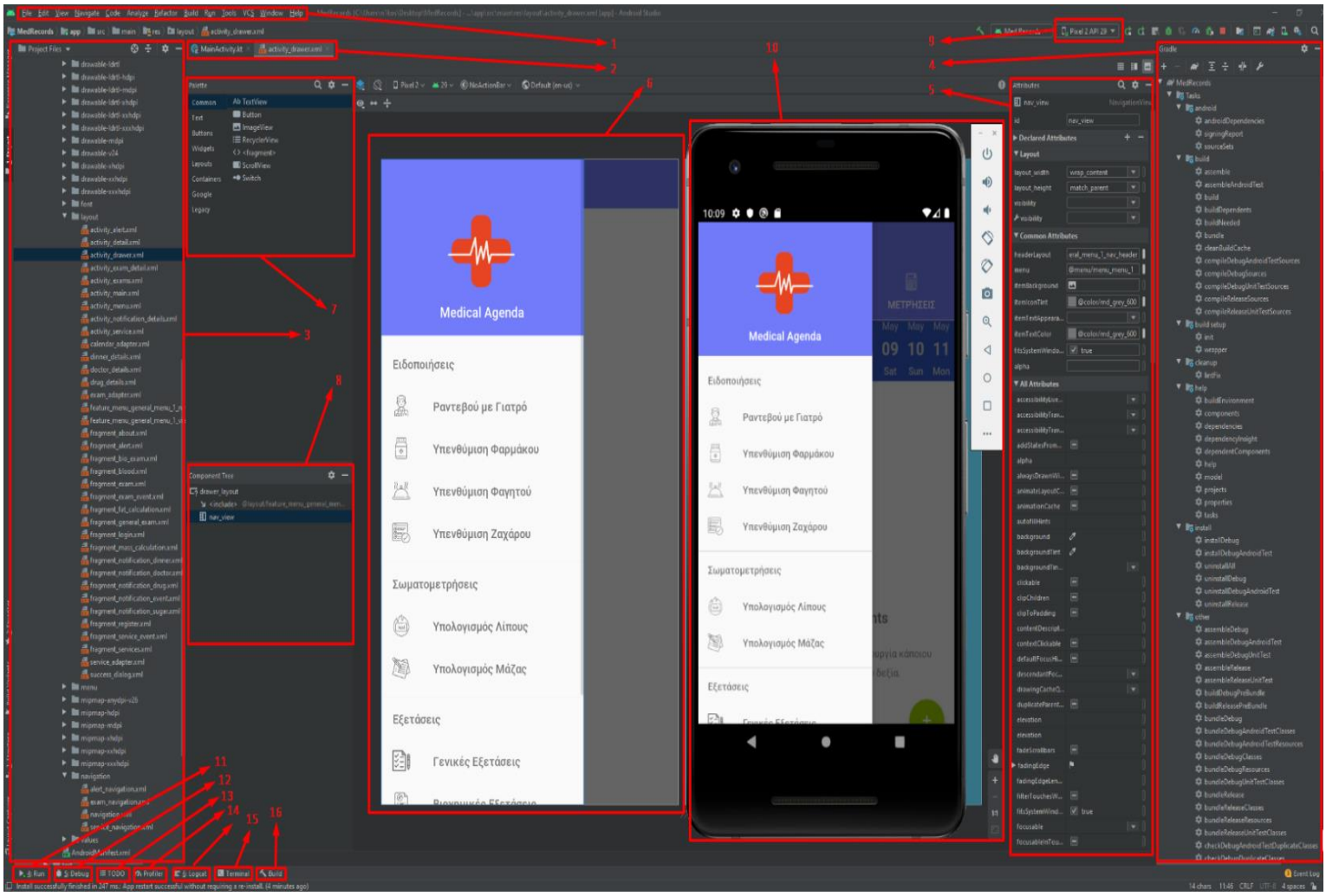
Οργάνωση Έργου (Organization of Project): Παρόλο που και τα δύο IDE λειτουργούν με διαφορετικό τρόπο για να βοηθήσουν τους προγραμματιστές στη διαχείριση και την

οργάνωση των προγραμματιστικών έργων, όταν θελήσουν να εργαστούν σε πολλά έργα στο Eclipse θα πρέπει να τα συγχωνεύσουν σε ένα workspace. Σε μια προσπάθεια μετάβασης σε διαφορετικό workspace, πρέπει να επιλέξουν το path, μετά την επανεκκίνηση του Eclipse και αυτό είναι χρονοβόρο και μη βολικό. Από την άλλη πλευρά, το Android Studio χρησιμοποιεί λειτουργικές μονάδες (modules) για τη διαχείριση και την οργάνωση των ενοτήτων κώδικα που έχουν τα δικά τους αρχεία κατασκευής Gradle. Συγκριτικά, το Andoid Studio φαίνεται πιο λειτουργικό για ανάπτυξη προγραμμάτων Android, απλά για κάποιον προγραμματιστή που δεν το έχει ξανά χρησιμοποιήσει, ίσως απαιτείται ορισμένος χρόνος για να εκμάθει το περιβάλλον αυτό.[74]

Δοκιμή εφαρμογών και εντοπισμός σφαλμάτων (App testing and debugging): Ο σωστός έλεγχος εφαρμογών πριν από την κυκλοφορία αποτελεί μεγάλη ανησυχία των προγραμματιστών. Η λειτουργικότητα Unit Tests του Android Studio έλαβε λαμπερά σχόλια, ακριβώς λόγω αυτού. Με τη ρύθμιση test classes και τη συμπερίληψή τους στη διαμόρφωση εκτέλεσης έργων, τα σφάλματα προγραμμάτων μπορούν να εντοπιστούν και να διορθωθούν στο Android Studio με ευκολία. Οι δοκιμές μονάδας (Unit Tests) μπορούν να ξεκινήσουν ενώ οι εφαρμογές βρίσκονται στο στάδιο κατασκευής (αυτό διασφαλίζει ότι τα σφάλματα δεν εισέρχονται στη φάση παραγωγής). Το Eclipse δεν διαθέτει παρόμοιο εργαλείο για εύκολη δοκιμή εφαρμογών. Αυτό είναι ένα ακόμη μειονέκτημα του Eclipse IDE.[73]

3.3.2 Παρουσίαση Android Studio

Σε αυτήν την ενότητα παρουσιάζουμε συνοπτικά το περιβάλλον του Android Studio συμπεριλαμβανομένων κάποιων βασικών εργαλειών που μας βοηθάνε στην ανάπτυξη λογισμικού.



Εικόνα 16. Περιβάλλον Android Studio

1) Tool bar: Είναι μια συλλογή από πολλά εργαλεία όπως cut, copy, paste run, debug κ.α.

2) Navigation bar: Βοηθάει τον χρήστη να πλοηγηθεί στα πρόσφατα ανοιχτά αρχεία του project (αρχεία κώδικα).

3) Project Files: Είναι η ιεραρχία των φακέλων του Project.

4) Gradle bar: Δείχνει τα διαθέσιμα Gradle tasks και ο χρήστης μπορεί να τα τρέξει απευθείας από εκεί.

5) Attributes: Έλεγχος για τα χαρακτηριστικά του επιλεγμένου layout.

6) View Mode: Δείχνει το layout είτε σε κώδικα, είτε σε σχεδίαση, ή και τα δύο μαζί όπου ονομάζεται Split.

7) Palette: Περιέχει διάφορες προβολές που μπορούν να μεταφερθούν στο layout.

8) Component Tree: Δείχνει ιεραρχικά τα στοιχεία του layout.

9) AVD Manager (Android Virtual Device): Δημιουργείται ένα ή περισσότερα ψηφιακά κινητά της επιλογής του χρήστη με τις απαραίτητες ρυθμίσεις που ορίζει (android API, μέγεθος μνήμης, μέγεθος οθόνης κ.α.) με σκοπό τον έλεγχο της εφαρμογής σε ψηφιακή αναπαράσταση.

10) Emulator: Είναι το γραφικό περιβάλλον του ψηφιακού κινητού τηλεφώνου που έχει επιλέξει ο χρήστης για να ελέγξει την εφαρμογή του.

11) Run: Το run tab είναι διαθέσιμο όταν μία εφαρμογή τρέχει και παρέχονται πληροφορίες και αποτελέσματα από την εκτέλεση του καθώς και επιλογές διακοπής ή επανεκκίνησης κάποιας διεργασίας. Επίσης, αν μία εφαρμογή αποτύχει να εγκαταστήσει και να τρέξει κάποιον εξομοιωτή, το παράθυρο run θα δώσει απαραίτητες πληροφορίες για να εντοπιστεί το πρόβλημα.

12) Debug: Αυτό το παράθυρο δείχνει τον εντοπισμό σφαλμάτων με τις ακόλουθες δυνατότητες. Το Frames tab όπου δείχνει την στοίβα με τα frames που εκτελούνται για ένα νήμα. Η εκτελέσιμη στοίβα εμφανίζει κάθε κλάση και μέθοδο που έχει κληθεί από την εφαρμογή στο Android Runtime, με την πιο πρόσφατη μέθοδο στην κορυφή της στοίβας. Το Watches button όπου δείχνει τις αλλαγές τιμών μίας μεταβλητής καθ’όλη την διάρκεια εκτέλεσης του προγράμματος. Τέλος, το Variables πλαίσιο όπου μας δείχνει τις τιμές που έχουν πάρει οι μεταβλητές μας, και πατώντας πάνω σε κάποια μεταβλητή παρουσιάζονται οι περαιτέρω ιδιότητες της.

12) TODO: Αυτό το εργαλείο σαρώνει τον κώδικα για σχόλια "TODO" και εμφανίζονται στο παράθυρο TODO οι εξής καρτέλες. Project, Current File, Scope Based. Στο Project δείχνονται τα στοιχεία TODO που υπάρχουν σε όλο το έργο. Στο Current File παρουσιάζονται τα στοιχεία TODO που υπάρχουν στο τρέχον αρχείο του editor. Στο Scope Based παρουσιάζονται τα στοιχεία TODO υπό συγκεκριμένης σκοπιάς, επιλεγμένα από τη λίστα. Με λίγα λόγια το TODO είναι σχόλια που έχει ο χρήστης τοποθετήσει στον κώδικα ως υπενθύμιση για επικείμενες αλλαγές που θέλει να κάνει.

13) Profiler: Είναι ένα εργαλείο που παρέχεται σε πραγματικό χρόνο και αναλύει προβλήματα όταν τρέχουν οι εφαρμογές, όπως CPU, μνήμη και χρήση δικτύου.

14) Logcat: Το συγκεκριμένο εργαλείο προσφέρει την δυνατότητα να βγάλει ο χρήστης σε βίντεο την εφαρμογή του ή να δημιουργήσει κάποιο στιγμιότυπο οθόνης. Επίσης έχει την δυνατότητα να σταματήσει ή να επανεκκινήσει μία διεργασία.

15) Terminal: Παρέχεται η πρόσβαση στο τερματικό (Command Window Promt) του συστήματος πάνω στο οποίο τρέχει η εφαρμογή.

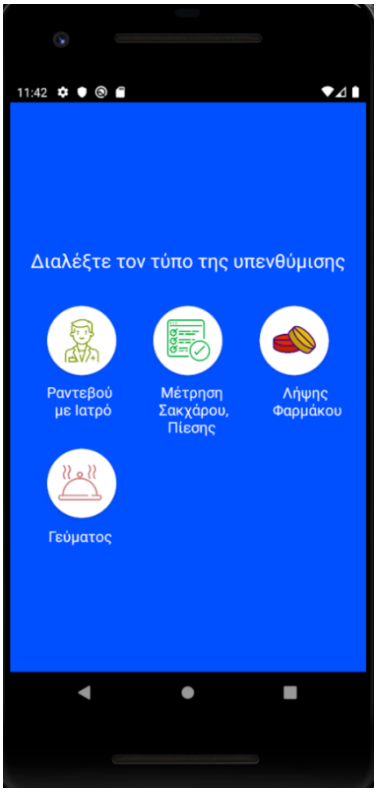
16) Build: Αυτό το εργαλείο απεικονίζει πληροφορίες σχετικά με το "χτίσιμο-build" του προγράμματος καθώς ένα project γίνεται compile (ελέγχεται για συντακτικά λάθη) και χωρίζεται σε πακέτα (packaged), καθώς αναφέρονται στο παράθυρο build, τυχόν λάθη που παρουσιάζονται κατά την διάρκεια του build-process.

3.3.3 Συμπεράσματα

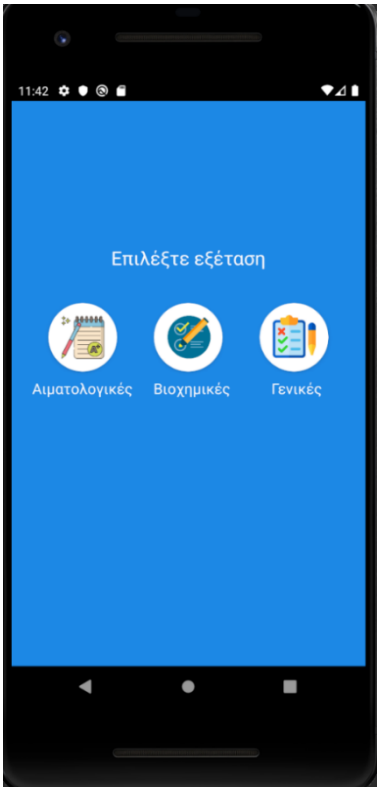
Το Android Studio είναι ένα βήμα μπροστά από το Eclipse, όσον αφορά τις προεπισκοπήσεις γραφικών επιλογών μεταξύ των δύο IDE. Η επιλογή εισαγωγής library resources (πηγές βιβλιοθηκών) απευθείας από το Maven Central είναι ένα άλλο πλεονέκτημα του Android Studio. Από τότε που το Android Studio ανακοινώθηκε κατά τη διάρκεια του Google I/O 2013, υπήρξε μεγάλη διαφημιστική εκστρατεία και αισιοδοξία μεταξύ των προγραμματιστών εφαρμογών Android, και το IDE σίγουρα ανταποκρίνεται στις περισσότερες προσδοκίες. Το Eclipse μπορεί να είναι παλιό, με λιγότερες δυνατότητες σε σχέση με το Android Studio, αλλά η επιλογή του κατάλληλου προγραμματιστικού περιβάλλοντος εξαρτάται και από το ποιόν της εφαρμογής που επιθυμείτε να αναπτυχθεί. Στην συγκεκριμένη πτυχιακή που αναπτύχθηκε εφαρμογή Android, το Android Studio ήταν, ιδανικά, η πρώτη επιλογή. Προσφέρει περισσότερη υποστήριξη σε όλους τους προγραμματιστές σε αντίθεση με το Eclipse.

4. Παρουσίαση Εφαρμογής «Ιατρική Ατζέντα – Medical Agenda»

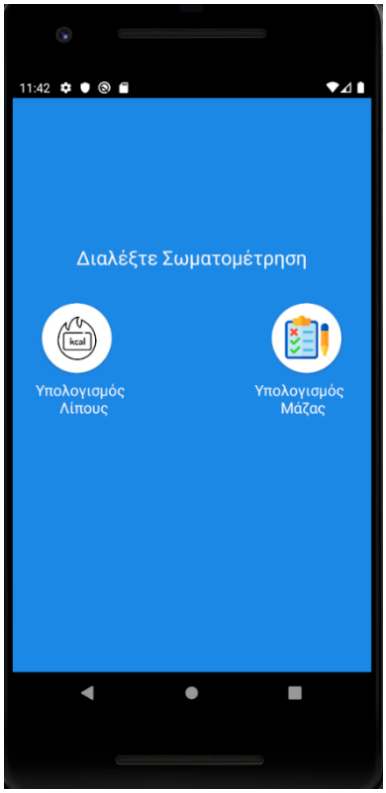
Σε αυτήν την ενότητα, παρουσιάζεται η εφαρμογή «Ιατρική Ατζέντα – Medical Agenda» καθώς και οι λειτουργίες της. Η βασική ιδέα της συγκεκριμένης Android εφαρμογής, είναι η διευκόλυνση του χρήστη της αναφορικά, με την υπενθύμιση των γευμάτων του, της μέτρησης σακχάρου ή πίεσης, των ραντεβού με ιατρούς, τον έλεγχο - αποθήκευση των ιατρικών του δεδομένων και τον υπολογισμό ορισμένων σωματομετρήσεων. Ιατρικά δεδομένα στην παρούσα εφαρμογή ορίζονται, εξετάσεις αίματος όπως, γενικές, βιοχημικές και αιματολογικές εξετάσεις. Αναλυτικότερα, με την συγκεκριμένη εφαρμογή, ο χρήστης θα έχει την δυνατότητα να υπενθυμίζεται από την Android συσκευή του έγκαιρα, για γεύματα, για μέτρηση σακχάρου ή πίεσης, για λήψη φαρμάκων και για ραντεβού με ιατρούς, όπου ο ίδιος έχει προγραμματίσει με την μορφή push-notification. Πέραν από τις ειδοποιήσεις push που λαμβάνει ο χρήστης από την εφαρμογή, μπορεί επίσης να εισάγει τις τιμές των αιματολογικών του εξετάσεων ανάλογα τον τύπο και να ελέγξει ποιες τιμές είναι εκτός ορίων και ποιες όχι, καθώς και την αποθήκευση των αποτελεσμάτων στην βάση δεδομένων της εφαρμογής. Τέλος, μία ακόμα λειτουργία της εφαρμογής, είναι ο υπολογισμός λίπους ή μάζας του σώματος και η αποθήκευση των αποτελεσμάτων στην βάση δεδομένων.



Εικόνα 17. Layout Επιλογής Υπενθύμισης

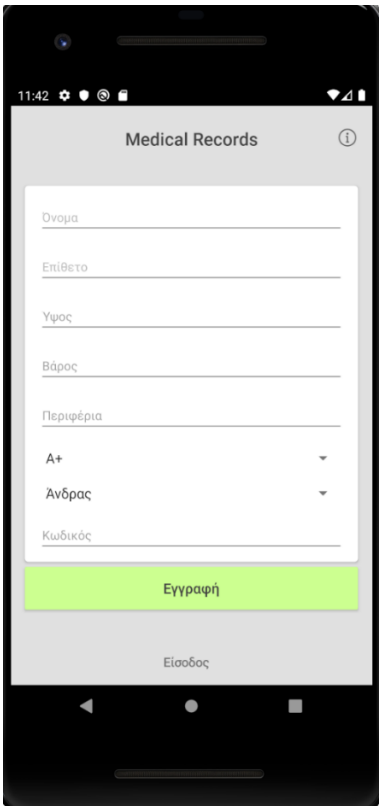


Εικόνα 18. Layout Επιλογής Εξέτασης

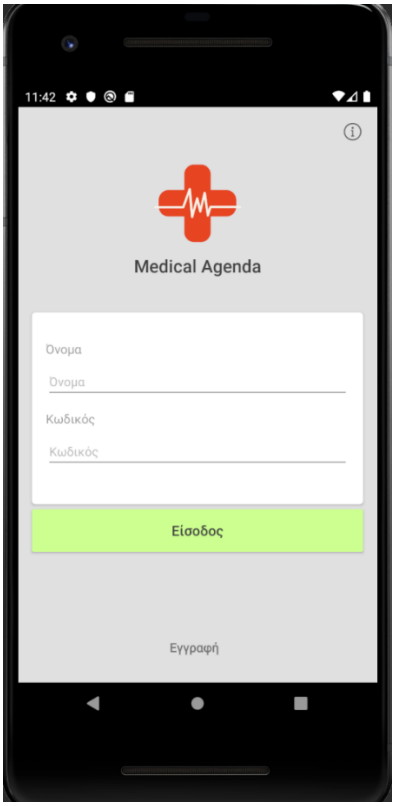


Εικόνα 19, Layout Επιλογής Σωματομέτρησης

4.1 Εγγραφή και Είσοδος χρήστη



Εικόνα 20. Layout Εγγραφής χρήστη

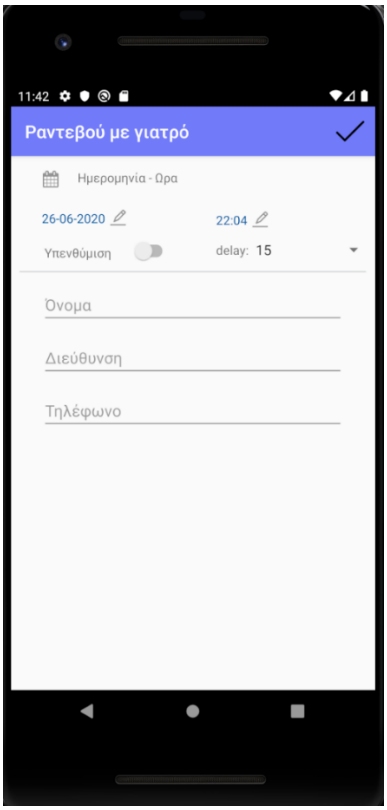


Εικόνα 21. Layout Εισόδου χρήστη

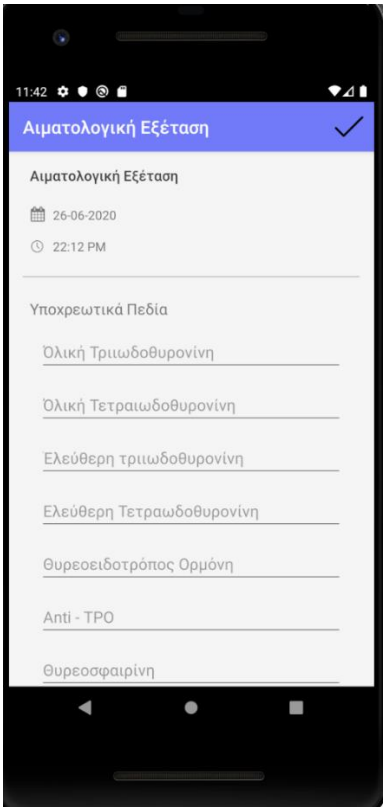
Στην Εικόνα 20 αποτυπώνεται η οθόνη εγγραφής του χρήστη στην εφαρμογή. Για την ολοκλήρωση της εγγραφής, ο χρήστης θα πρέπει να επιλέξει το κουμπί «εγγραφή» όπως διακρίνεται στην Εικόνα 21. Έπειτα, ο χρήστης θα πρέπει να συμπληρώσει τα στοιχεία του (Όνομα, Επίθετο, Ύψος, Βάρος, Περιφέρεια, Ομάδα αίματος, Φύλο και Κωδικό) για να δημιουργήσει νέο λογαριασμό. Ο χρήστης είναι υποχρεωμένος να συμπληρώσει όλα τα πεδία ώστε να ολοκληρωθεί η εγγραφή, αλλιώς σε περίπτωση ελλιπής συμπλήρωσης, εμφανίζεται μήνυμα «Συμπληρώστε το πεδίο». Μετά την ολοκλήρωση της εγγραφής, η εφαρμογή ανακατευθύνει τον χρήστη στην οθόνη εισόδου, όπως φαίνεται στην Εικόνα 21, όπου ο χρήστης θα υποχρεώνεται να πληκτρολογήσει το όνομα και τον κωδικό του. Εφόσον τα στοιχεία που εισάγει είναι σωστά, θα μπορεί να πλοηγηθεί στα ενδότερα της εφαρμογής. Σε περίπτωση λανθασμένων στοιχείων, ο χρήστης θα ειδοποιείται με μήνυμα «Ο κωδικός ή το όνομα είναι λάθος». Να σημειωθεί ότι η βάση δεδομένων που αποθηκεύονται οι εγγραφές των χρηστών, είναι τοπική (Room Database), άρα ο χρήστης, για να έχει πρόσβαση στην εφαρμογή δεν είναι υποχρεωτικό να έχει πρόσβαση στο διαδίκτυο. Τέλος, πάνω δεξιά όπως φαίνεται στις εικόνες, πατώντας το κουμπί Information (i), ο χρήστης μπορεί να δει πληροφορίες σχετικά με την εφαρμογή, τους δημιουργούς και τα Social Media της εφαρμογής, όπως φαίνεται στην Εικόνα 22.

Εικόνα 22. Layout με Γενικές Πληροφορίες Εφαρμογής

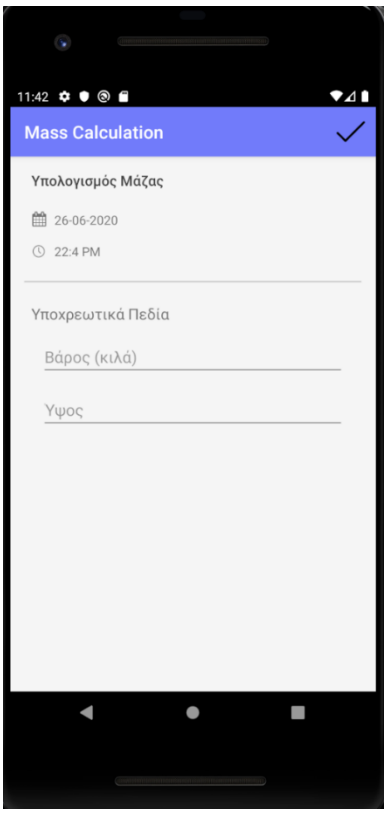
4.2 Λειτουργίες Εφαρμογής



Εικόνα 23. Layout Καταχώρησης Ραντεβού

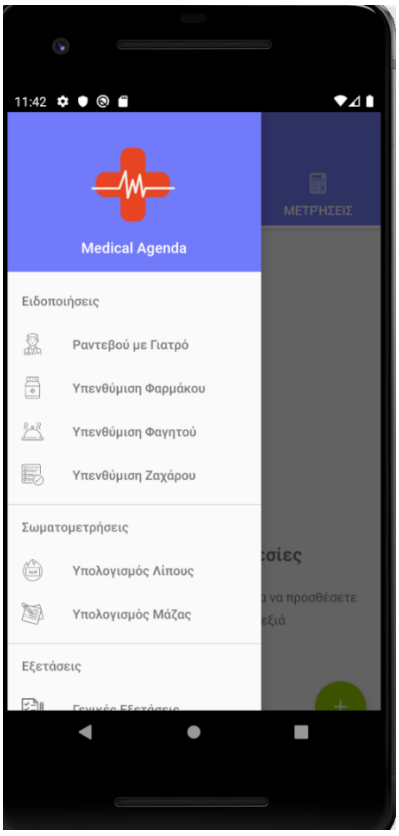


Εικόνα 24. Layout Καταχώρησης Εξέτασης

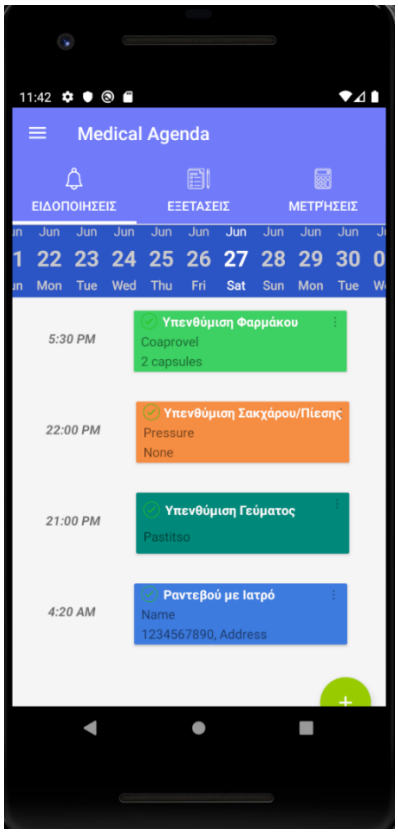


Εικόνα 25. Layout Υπολογισμού Μάζας

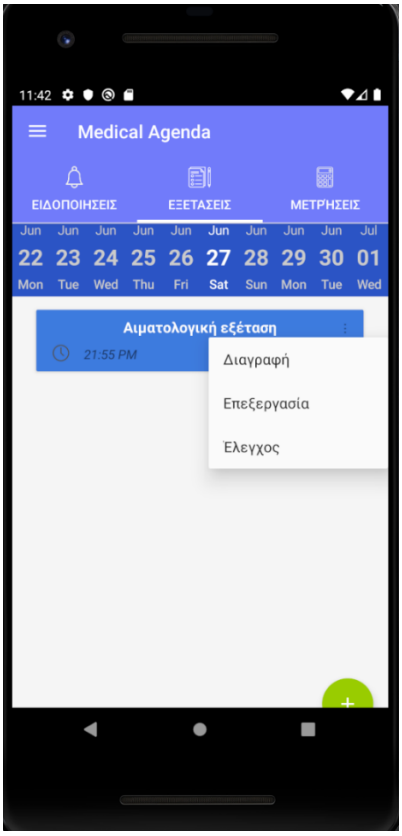
Όταν ο χρήστης αποκτήσει πρόσβαση στην εφαρμογή, όπως αποτυπώνεται στις παραπάνω εικόνες (Εικόνα 23, Εικόνα 24, Εικόνα 25), θα έχει την δυνατότητα πληκτρολογώντας τα απαραίτητα πεδία, ανάλογα την λειτουργία που έχει επιλέξει, να ορίσει τον τύπο υπενθύμισης που επιθυμεί όπως φαίνεται στην Εικόνα 17, ή να ελέγξει και να αποθηκεύσει την εξέταση αίματος που έχει επιλέξει όπως φαίνεται στην Εικόνα 18, ή να πραγματοποιήσει κάποια σωματομέτρηση όπως απεικονίζεται στην Εικόνα 19.



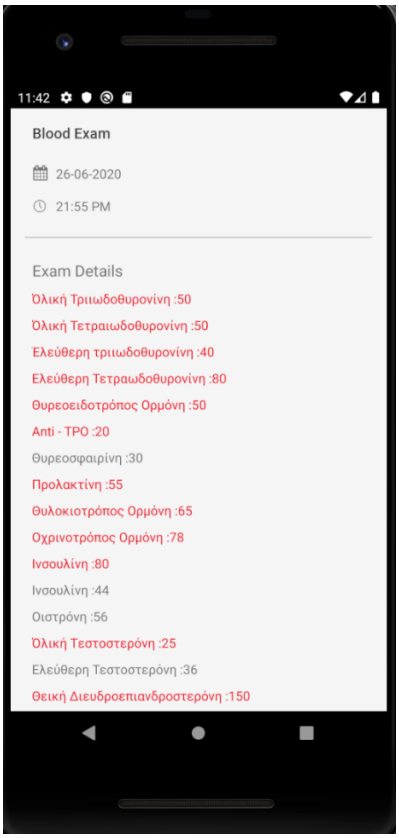
Εικόνα 26. Μενού Λειτουργιών



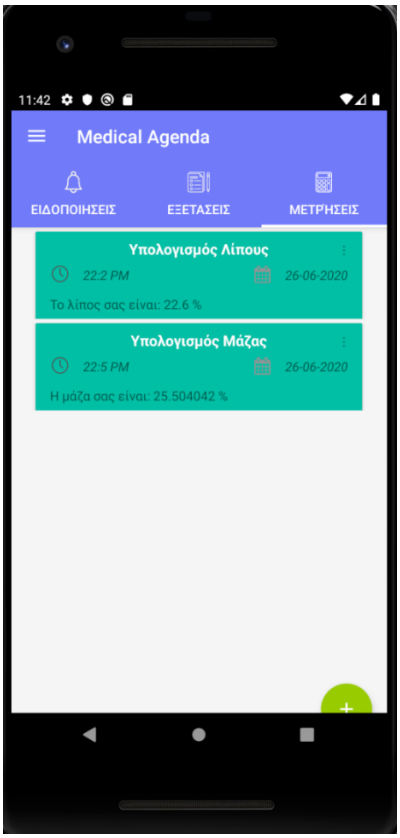
Εικόνα 27. Καταχωρημένες Υπενθυμίσεις



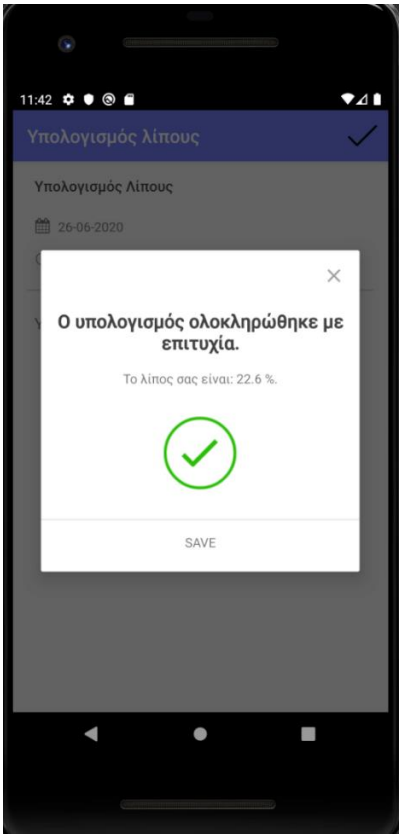
Εικόνα 28. Καταχωρημένες Εξετάσεις



Εικόνα 29. Έλεγχος Αιματολογικών Εξετάσεων



Εικόνα 30. Καταχωρημένες Μετρήσεις



Εικόνα 31. Υπολογισμός Μέτρησης Λίπους

Το γρήγορο μενού λειτουργιών της εφαρμογής, απεικονίζεται στην Εικόνα 26. Στην Εικόνα 27, φαίνονται ενδεικτικά κάποιες ορισμένες υπενθυμίσεις, οι οποίες ανάλογα τον τύπο τους φέρουν διαφορετικό χρωματισμό. Στην Εικόνα 28, έχει καταχωρηθεί με τυχαίες τιμές μια αιματολογική εξέταση, η οποία αποθηκεύεται στην ημέρα που έχει καταχωρηθεί και εν συνεχεία, ο χρήστης έχει την δυνατότητα να την διαγράψει, επεξεργαστεί ή πραγματοποιήσει τον έλεγχο που διακρίνεται στην Εικόνα 29. Οι τιμές με ερυθρό χρωματισμό βρίσκονται εκτός των επιθυμητών ορίων. Τέλος, στην Εικόνα 30 φαίνονται δύο τυχαίες αποθηκευμένες σωματομετρήσεις ποσοστού λίπους και δείκτη μάζας σώματος, ενώ στην Εικόνα 31 το μήνυμα επιτυχίας υπολογισμού.

Βιβλιογραφία – Παραπομπές

[1] King, B., 2016. Is Android Really Open Source? And Does It Even Matter? *MakeUseOf*. Available at: https://www.makeuseof.com/tag/android-really-open-source-matter/ [Accessed April 3, 2020].

[2] Anon, Google Mobile Services. *Android*. Available at: https://www.android.com/gms/ [Accessed April 3, 2020].

[3] Clement, J., 2020. Google Play Store: number of apps 2019. *Statista*. Available at: https://www.statista.com/statistics/266210/number-of-available-applications-in-the-google-play-store/ [Accessed April 3, 2020].

[4] FAUguy, 2011. Google's Android OS: Past, Present, and Future. *Phone Arena*. Available at: https://www.phonearena.com/news/Googles-Android-OS-Past-Present-and-Future_id21273 [Accessed April 3, 2020].

[5] Elgin, B., 2005. *Google Buys Android for Its Mobile Arsenal*. Available at: https://web.archive.org/web/20110205190729/http:/www.businessweek.com/technology/content/aug2005/tc20050817_0949_tc024.htm [Accessed April 3, 2020].

[6] Alabaster, J., 2013. Android founder: We aimed to make a camera OS. *PCWorld*. Available at: https://www.pcworld.com/article/2034723/android-founder-we-aimed-to-make-a-camera-os.html [Accessed April 3, 2020].

[7] Welch, C., 2013. Before it took over smartphones, Android was originally destined for cameras. *The Verge*. Available at: https://www.theverge.com/2013/4/16/4230468/android-originally-designed-for-cameras-before-smartphones [Accessed April 3, 2020].

[8] Ionescu, D., 2012. Original Android Prototype Revealed During Google, Oracle Trial. *PCWorld*. Available at: https://www.pcworld.com/article/254539/original_android_prototype_revealed_during_google_oracle_trial.html [Accessed April 3, 2020].

[9] Ziegler, C., 2012. This was the original 'Google Phone' presented in 2006. *The Verge*. Available at: https://www.theverge.com/2012/4/25/2974676/this-was-the-original-google-phone-presented-in-2006 [Accessed April 3, 2020].

[10] Ziegler, C., 2012. Google in 2007: 'a touchscreen cannot completely replace physical buttons'. *The Verge*. Available at: https://www.theverge.com/2012/4/25/2974843/google-in-2007-a-touchscreen-cannot-completely-replace-physical/in/2731667 [Accessed April 3, 2020].

[11] Anon, 2008. T-Mobile officially announces the G1 Android phone. *TechCrunch*. Available at: https://techcrunch.com/2008/09/23/t-mobile-officially-announces-the-g1-android-phone/ [Accessed April 3, 2020].

[12] Gao, Richard, 2016. Android and its first purchasable product, the T-Mobile G1, celebrate their 8th birthdays today. *Android Police*. Available at: https://www.androidpolice.com/2016/09/23/android-first-purchasable-product-t-mobile-g1-celebrate-8th-birthdays-today/ [Accessed April 3, 2020].

[13] Anon, 2007. Industry Leaders Announce Open Platform for Mobile Devices: Open Handset Alliance. *Industry Leaders Announce Open Platform for Mobile Devices | Open Handset Alliance*. Available at: http://www.openhandsetalliance.com/press_110507.html [Accessed April 3, 2020].

[14] Schonfeld, E., 2007. Breaking: Google Announces Android and Open Handset Alliance. *TechCrunch*. Available at: https://techcrunch.com/2007/11/05/breaking-google-announces-android-and-open-handset-alliance/ [Accessed April 3, 2020].

[15] Andy Rubin, 2007. Where's my Gphone? *Official Google Blog*. Available at: https://googleblog.blogspot.com/2007/11/wheres-my-gphone.html [Accessed April 3, 2020].

[16] Staff, E., 2007. Googles Strong Mobile-Related Patent Portfolio. *Gigaom*. Available at: https://gigaom.com/2007/09/20/419-googles-strong-mobile-related-patent-portfolio/ [Accessed April 3, 2020].

[17] Menon, M.K., 2016. Android Nougat: Here's why Google names the OS after sweets. *The Indian Express*. Available at: https://indianexpress.com/article/lifestyle/food-wine/from-donut-to-nougat-why-are-android-versions-named-after-sweets-2887237/ [Accessed April 3, 2020].

[18] Florence Ion - May 15, 2013 1:01 pm U.T.C., 2013. From Nexus One to Nexus 10: a brief history of Google's flagship devices. *Ars Technica*. Available at: https://arstechnica.com/gadgets/2013/05/from-the-nexus-one-to-the-nexus-10-a-brief-history-of-nexus-devices/ [Accessed April 3, 2020].

[19] Hollister, S., 2013. Google turns the Samsung Galaxy S4 into a Nexus phone, coming June 26th for $649. *The Verge*. Available at: https://www.theverge.com/2013/5/15/4333716/galaxy-s4-stock-android-google-io-2013 [Accessed April 3, 2020].
[20] Cunningham, A., 2013. Review: The HTC One Google Play edition offers the best of both worlds. *Ars Technica*. Available at: https://arstechnica.com/gadgets/2013/07/review-the-htc-one-google-play-edition-offers-the-best-of-both-worlds/ [Accessed April 3, 2020].
[21] Cunningham, A., 2014. Moto G Google Play edition replaces near-stock Android with stock Android. *Ars Technica*. Available at: https://arstechnica.com/gadgets/2014/01/moto-g-google-play-edition-replaces-near-stock-android-with-stock-android/ [Accessed April 3, 2020].
[22] Smith, M., 2013. Android VP Hugo Barra leaves Google, joins Chinese phone maker Xiaomi (updated). *Engadget*. Available at: https://www.engadget.com/2013-08-28-android-vp-hugo-barra-report-leaves-google-xiaomi.html [Accessed April 3, 2020].
[23] Anon, 2013. Update from the CEO. *Official Google Blog*. Available at: https://googleblog.blogspot.com/2013/03/update-from-ceo.html [Accessed April 3, 2020].
[24] Arthur, C., 2013. Andy Rubin moved from Android to take on 'moonshots' at Google. *The Guardian*. Available at: https://www.theguardian.com/technology/2013/mar/13/andy-rubin-google-move [Accessed April 3, 2020].
[25] Brandom, R., 2015. Google is reorganizing and Sundar Pichai will become new CEO. *The Verge*. Available at: https://www.theverge.com/2015/8/10/9128083/sundar-pichai-ceo-google-larry-page-sergey-brin [Accessed April 3, 2020].
[26] Conditt, J., 2015. Google gets an overhaul and a new CEO: Sundar Pichai. *Engadget*. Available at: https://www.engadget.com/2015-08-10-google-gets-an-overhaul-and-a-new-ceo-sundar-pichai.html [Accessed April 3, 2020].
[27] Bergen, M., 2015. New Google CEO Sundar Pichai Makes First Major Executive Picks. *Vox*. Available at: https://www.vox.com/2015/10/9/11619448/new-google-ceo-sundar-pichai-makes-first-major-executive-picks [Accessed April 3, 2020].
[28] Martonik, A., 2015. Sundar Pichai promotes Hiroshi Lockheimer to oversee Android. *Android Central*. Available at: https://www.androidcentral.com/sundar-pichai-promotes-hiroshi-lockheimer-oversee-android-chrome-os-and-chromecast [Accessed April 3, 2020].
[29] Kastrenakes, J., 2014. Android One will help manufacturers build low-cost phones for developing markets. *The Verge*. Available at: https://www.theverge.com/2014/6/25/5841976/android-one-hardware-reference-program [Accessed April 3, 2020].
[30] Seifert, D., 2014. With Android One, Google is poised to own the entire world. *The Verge*. Available at: https://www.theverge.com/2014/6/26/5845562/android-one-google-the-next-billion [Accessed April 3, 2020].
[31] Woods, B., 2014. Google Announces Android One Standard for Affordable Devices. *The Next Web*. Available at: https://thenextweb.com/google/2014/06/25/google-announces-android-one-standard-affordable-devices-arriving-first-india-100/ [Accessed April 3, 2020].
[32] Pichai, Sundar., 2014. For the next five billion: Android One. *Official Google Blog*. Available at: https://googleblog.blogspot.com/2014/09/for-next-five-billion-android-one.html [Accessed April 3, 2020].
[33] Shilpa Kannan, 2014. Android One smartphones released in India. *BBC News*. Available at: https://www.bbc.com/news/av/technology-29209103/android-one-smartphones-released-in-india [Accessed April 3, 2020].
[34] Savov, V., 2016. Pixel 'phone by Google' announced. *The Verge*. Available at: https://www.theverge.com/2016/10/4/13161028/google-phone-announced-pixel-xl-price-release-date-specs [Accessed April 3, 2020].
[35] Lawler, R., 2016. Google's Pixel phones make their debut. *Engadget*. Available at: https://www.engadget.com/2016-10-04-made-by-google-pixel-pixel-xl.html [Accessed April 3, 2020].
[36] Seifert, D., 2016. Google's new Pixel phones come with Android 7.1 Nougat. *The Verge*. Available at: https://www.theverge.com/2016/10/4/13098314/google-android-update-7-1-nougat-new-features [Accessed April 3, 2020].
[37] Ng, A., 2016. Pixel won't share Google Assistant with other Android phones. *CNET*. Available at: https://www.cnet.com/news/google-pixel-android-7-1-assistant-now/ [Accessed April 3, 2020].

[38] Bohn, D., 2016. The inside story of the Google phone. *TheVerge.com*. Available at: https://www.theverge.com/a/google-pixel-phone-new-hardware-interview-2016 [Accessed April 3, 2020].
[39] Kastrenakes, J., 2017. Google Pixel 2 and 2 XL phones announced. *The Verge*. Available at: https://www.theverge.com/circuitbreaker/2017/10/4/16408962/new-google-pixel-2-phone-announced-price-release-date-features [Accessed April 3, 2020].
[40] Sottek, T.C., 2019. Google pulls Huawei's Android license, forcing it to use open source version. *The Verge*. Available at: https://www.theverge.com/2019/5/19/18631558/google-huawei-android-suspension [Accessed April 3, 2020].
[41] Cook, J., 2019. Google restricts Huawei from using Android: Here's what that could mean for you. *The Telegraph*. Available at: https://www.telegraph.co.uk/technology/2019/05/20/google-restricts-huawei-using-android-could-mean/ [Accessed April 3, 2020].
[42] MA SI, 2019. Huawei's operating system in pipeline. *Huawei's operating system in pipeline - Chinadaily.com.cn*. Available at: http://www.chinadaily.com.cn/a/201906/11/WS5cfeeebda31017657723067f.html [Accessed April 3, 2020].
[43] Reichert, C., 2019. Huawei's Hongmeng OS could be trademarked in Peru. *CNET*. Available at: https://www.cnet.com/news/huawei-files-trademark-for-hongmeng-os-while-objecting-to-us-ban-reports-say/ [Accessed April 3, 2020].
[44] Sohail, O., 2019. Huawei's Own Smartphone Operating System Reportedly Named HongMeng OS, According to Foreign Sources. *Wccftech*. Available at: https://wccftech.com/huawei-hongmeng-os-for-smartphones/ [Accessed April 3, 2020].
[45] Jie, Yang, 2019. Who Needs Google's Android? Huawei Trademarks Its Own Smartphone OS. *The Wall Street Journal*. Available at: https://www.wsj.com/articles/who-needs-googles-android-huawei-trademarks-its-own-smartphone-os-11558693195 [Accessed April 3, 2020].
[46] England, J., 2019. Huawei begins trademarking its Android replacement OS - HongMeng. *Android Central*. Available at: https://www.androidcentral.com/huawei-begins-trademarking-its-android-replacement-os-several-countries [Accessed April 3, 2020].
[47] Porter, J., 2019. Huawei's new operating system is called HarmonyOS. *The Verge*. Available at: https://www.theverge.com/2019/8/9/20798251/huawei-harmonyos-hongmengos-smartphones-internet-of-things-operating-system-android [Accessed April 3, 2020].
[48] Bohn, D., 2019. Google deserts desserts: Android 10 is the official name for Android Q. *The Verge*. Available at: https://www.theverge.com/2019/8/22/20827231/android-10-q-google-name-officially-announced-new-logo-wordmark-desserts [Accessed April 3, 2020].
[49] Amadeo, R., 2019. Unsweetened: Android swaps sugary codenames for boring numbers. *Ars Technica*. Available at: https://arstechnica.com/gadgets/2019/08/unsweetened-android-swaps-sugary-codenames-for-boring-numbers/ [Accessed April 3, 2020].
[50] David.ruddock, 2020. Android's iconic dessert names are going away, starting with Android 10. *Android Police*. Available at: https://www.androidpolice.com/2019/08/22/androids-iconic-dessert-names-are-going-away-starting-with-android-10/ [Accessed April 3, 2020].
[51] Anon, 2019. Android versions comparison: Comparison tables. *SocialCompare*. Available at: https://socialcompare.com/en/comparison/android-versions-comparison [Accessed April 5, 2020].
[52] Anon, 2009. Android 1.5 Platform Highlights : Android Developers. *Android Developers*. Available at: https://developer.android.com/about/versions/android-1.5-highlights.html [Accessed April 5, 2020].
[53] Anon, 2016. Android 1.6 Platform Highlights : Android Developers. *Android Developers*. Available at: https://developer.android.com/about/versions/android-1.6-highlights.html [Accessed April 5, 2020].
[54] Anon, 2016. Android 2.0 Platform Highlights : Android Developers. *Android Developers*. Available at: https://developer.android.com/about/versions/android-2.0-highlights.html [Accessed April 5, 2020].
[55] Anon, 2016. Android 2.2 Platform Highlights : Android Developers. *Android Developers*. Available at: https://developer.android.com/about/versions/android-2.2-highlights.html [Accessed April 5, 2020].

[56] Anon, 2016. Gingerbread : Android Developers. *Android Developers*. Available at: https://developer.android.com/about/versions/android-2.3-highlights.html [Accessed April 5, 2020].
[57] Anon, 2016. Honeycomb : Android Developers. *Android Developers*. Available at: https://developer.android.com/about/versions/android-3.0-highlights.html [Accessed April 5, 2020].
[58] Anon, 2016. Honeycomb MR1 : Android Developers. *Android Developers*. Available at: https://developer.android.com/about/versions/android-3.1-highlights.html [Accessed April 5, 2020].
[59] Anon, 2016. Ice Cream Sandwich : Android Developers. *Android Developers*. Available at: https://developer.android.com/about/versions/android-4.0-highlights.html [Accessed April 5, 2020].
[60] Raphael, J.R., 2020. Android versions: A living history from 1.0 to 11. *Computerworld*. Available at: https://www.computerworld.com/article/3235946/android-versions-a-living-history-from-1-0-to-today.html [Accessed April 5, 2020].
[61] Anon, 2020. Ιστορία εκδόσεων του Android. *Wikipedia*. Available at: https://el.wikipedia.org/wiki/Ιστορία_εκδόσεων_του_Android [Accessed April 5, 2020].
[62] Anon, Comments. *Android version distribution history - Bidouille.org*. Available at: https://www.bidouille.org/misc/androidcharts [Accessed April 5, 2020].
[63] Anon, Android Architecture. *Tutlane*. Available at: https://www.tutlane.com/tutorial/android/android-architecture [Accessed April 7, 2020].
[64] Anon, 2019. Android Jetpack : Android Developers. *Android Developers*. Available at: https://developer.android.com/jetpack [Accessed April 10, 2020].
[65] Anon, 2019. Android Architecture Components. *AndroidWave*. Available at: https://androidwave.com/android-architecture-components/ [Accessed April 10, 2020].
[66] Baruckis Author, A., 2019. Kriptofolio app series - Part 3: Architecture patterns MVC, MVP, MVVM with Android Architecture Components - ViewModel, LiveData, Data Binding, Room. *Andrius Baruckis blog*. Available at: https://www.baruckis.com/android/kriptofolio-app-series-part-3/ [Accessed April 22, 2020].
[67] Nguyen, Q., 2019. Architecture patterns in Android. *Medium*. Available at: https://android.jlelse.eu/architecture-patterns-in-android-abf99f2b6f70 [Accessed April 22, 2020].
[68] Anon, The Anatomy of an Android Application. *Techotopia*. Available at: https://www.techotopia.com/index.php/The_Anatomy_of_an_Android_Application [Accessed April 24, 2020].
[69] Anon, 2020. Intent Android Developers. *Android Developers*. Available at: https://developer.android.com/reference/android/content/Intent [Accessed April 24, 2020].
[70] Chris Yackulic, 2018. Prime Advantages Of Android Operating System. *Android Headlines*. Available at: https://www.androidheadlines.com/2018/10/prime-advantages-of-android-operating-system.html [Accessed April 25, 2020].

[71] Anon, 2020. Java vs Kotlin: 8 Most Amazing Differences You Should Know. *EDUCBA*. Available at: https://www.educba.com/java-vs-kotlin/ [Accessed April 26, 2020].

[72] Vijay Singh, 2020. Kotlin vs Java: Most Important Differences That You Must Know. *Hackr.io*. Available at: https://hackr.io/blog/kotlin-vs-java [Accessed April 26, 2020].

[73] Rajput, M., 2015. Why Android Studio Is Better For Android Developers Instead Of Eclipse - DZone Mobile. *dzone.com*. Available at: https://dzone.com/articles/why-android-studio-better [Accessed April 28, 2020].

[74] Fakhruddin, H., 2015. Eclipse vs Android Studio: Which IDE Is Better For Android Developers? *Teknowledge Software : iPhone Application Development Company India*. Available at: https://teks.co.in/site/blog/eclipse-vs-android-studio-ide-better-android-developers/ [Accessed April 28, 2020].