**ΠΑΝΕΠΙΣΤΗΜΙΟ ΔΥΤΙΚΗΣ ΜΑΚΕΔΟΝΙΑΣ**
**ΣΧΟΛΗ ΘΕΤΙΚΩΝ ΕΠΙΣΤΗΜΩΝ**
**ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ**

# Μελέτη και χρήση εργαλείων για ανάπτυξη εφαρμογών σε περιβάλλον Android
# Chat application – WhatsApp Clone

**ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ**

του

**ΑΛΠΙΔΗΣ ΠΑΝΑΓΙΩΤΗΣ (ΑΕΜ:1755)**

**ΑΛΠΙΔΟΥ ΧΡΥΣΗ (ΑΕΜ:1681)**

**Επιβλέπων: Δημήτριος Ι. Βέργαδος**
**Επίκουρος Καθηγητής**

Καστοριά Ιούλιος - 2023

**ΠΑΝΕΠΙΣΤΗΜΙΟ ΔΥΤΙΚΗΣ ΜΑΚΕΔΟΝΙΑΣ**

**ΣΧΟΛΗ ΘΕΤΙΚΩΝ ΕΠΙΣΤΗΜΩΝ**

**ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ**

# Μελέτη και χρήση εργαλείων για ανάπτυξη εφαρμογών σε περιβάλλον Android
# Chat application – WhatsApp Clone

**ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ**

Του

**ΑΛΠΙΔΗΣ ΠΑΝΑΓΙΩΤΗΣ (ΑΕΜ:1755)**

**ΑΛΠΙΔΟΥ ΧΡΥΣΗ (ΑΕΜ:1681)**

**Επιβλέπων: Δημήτριος Ι. Βέργαδος**
**Επίκουρος Καθηγητής**

Εγκρίθηκε από την τριμελή εξεταστική επιτροπή την 05/07/2023

| ..... | ..... | ..... |
|---|---|---|
| Δημήτριος Ι. Βέργαδος | Νίκος Δημόκας | Σπυρίδων Νικολάου |
| Επίκουρος Καθηγητής | Επίκουρος Καθηγητής | Λέκτορας |

Καστοριά Ιούλιος - 2023

## Περίληψη

Σκοπός της παρούσας πτυχιακής εργασίας είναι η κατασκευή μιας mobile chat εφαρμογής παρόμοια με αυτή της WhatsApp , με την χρήση του Android SDK. Στο πρώτο κεφάλαιο γίνεται μια εισαγωγή στις mobile εφαρμογές και το λειτουργικό σύστημα Android. Στο δεύτερο κεφάλαιο παρουσιάζονται εφαρμογές παρόμοιες με αυτήν που υλοποιήθηκε για τις ανάγκες της τρέχουσας εργασίας και γίνεται μελέτη των δυνατοτήτων τους. Το τρίτο κεφάλαιο περιλαμβάνει την διαδικασία εγκατάστασης των εργαλείων για την υλοποίησή της και την περιγραφή των σημαντικότερων λειτουργιών της με κομμάτια κώδικα. Το τέταρτο και τελευταίο κεφάλαιο, είναι τα συμπεράσματα από την υλοποίηση της εφαρμογής και κάποιες προτάσεις για μελλοντικές βελτιώσεις και προσθήκες.

## Abstract

The purpose of this dissertation is to build a mobile chat application similar to that of WhatsApp, using the Android SDK. The first chapter is an introduction to mobile applications and the Android operating system. The second chapter presents applications similar to the one that was implemented for the needs of the current work and their possibilities are studied. The third chapter includes the process of installing the tools for its implementation and the description of its most important functions with pieces of code. The fourth and last chapter, are the conclusions from the implementation of the application and some suggestions for future improvements and additions.

## Ευχαριστίες

Σε αυτό το σημείο θα ήθελα να ευχαριστήσω τον καθηγητή Δημήτριο Βέργαδο για την ανάθεση της πτυχιακής μου εργασίας και την επίβλεψη καθώς και σημαντική καθοδήγησή του. Επίσης θα ήθελα να ευχαριστήσω, την οικογένεια μου, για την υποστήριξη και βοήθεια κατά την διάρκεια των σπουδών μου στο Πανεπιστήμιο. Τέλος, θα ήθελα να

ευχαριστήσω τους φίλους μου και του συμφοιτητές μου αλλά και όλους όσους με βοήθησαν και με στήριξαν κατά την διάρκεια των σπουδών μου.

# Πίνακας Περιεχομένων

# Πίνακας Εικόνων

# ΚΕΦΑΛΑΙΟ 1 - ΕΙΣΑΓΩΓΗ

## 1.1 Τι είναι ένα Mobile App

Mobile App ονομάζεται μια εφαρμογή η οποία έχει κατασκευαστεί για να εκτελείται σε φορητές συσκευές όπως τα smartphones ή τα tablets. Φιλοδοξεί να καλύψει ανάγκες των χρηστών που θα απαιτούσαν τη χρήση υπολογιστών. Αποτελείται από μικρές μονάδες λογισμικού με περιορισμένη λειτουργία. Εναλλακτικές ονομασίες μιας εφαρμογής για κινητά είναι οι παρακάτω: application, web application, application iPhone ή application smartphone (1).

Η βασική στόχευση των Mobile apps - τουλάχιστον το πρώτο διάστημα- αφορούσε τη δημιουργία εφαρμογών γενικού ενδιαφέροντος των χρηστών, όπως για παράδειγμα τη δημιουργία εφαρμογών για e-mails, ημερολόγια, αριθμομηχανή, πρόγνωση καιρού, επαφές κτλ. Με την πάροδο των ετών και την ολοένα αυξανόμενη χρήση κινητών συσκευών οι κατηγορίες των εφαρμογών διευρύνθηκαν και αναπτύχθηκαν οι παρακάτω τομείς: gaming, χάρτες (GPS), τραπεζικές εφαρμογές, online παραγγελίες κτλ (2).

## 1.2 History of Mobile Application Development

Όσο αναπόσπαστο μέρος της ζωής μας έχουν γίνει, δεν ήταν πολύ καιρό πριν, οι εφαρμογές για κινητά ήταν σπάνιες. Τα κινητά τηλέφωνα ενδέχεται να είχαν αποκλειστικές λειτουργίες ή λειτουργίες (για παράδειγμα μια αριθμομηχανή), αλλά οι εφαρμογές όπως τις γνωρίζουμε πρόσφατα ξεκίνησαν μαζί με smartphone και PDA. Παρά το μικρό χρονικό διάστημα που πέρασαν, είχαν μια υπέροχη ιστορία.

Το Motorola DynaTAC 8000X το 1983 ήταν το κινητό που έκανε την αρχή και άνοιξε το δρόμο για όλα τα άλλα κινητά τηλέφωνα, όπως των smartphone και των εφαρμογών που οι σημερινοί χρήστες δεν μπορούν να ζήσουν χωρίς αυτά. Παρόλο που ήταν ένα αρκετά μεγάλο κινητό τηλέφωνο δεν έκανε κάτι παραπάνω από το να κάνει κλήσεις και έτσι κάποιοι κατασκευαστές αποφάσισαν να προσθέσουν απλά παιχνίδια όπως το Snake, το Pong και το Tic-Tac-Toe για περισσότερη λειτουργικότητα (3).

Οι εφαρμογές αυτές ήταν αρκετά εύκολες και έπαιζαν με τη δημοτικότητα των παιχνιδιών που ήταν ήδη στην αγορά για κονσόλες. Επίσης, άλλαξαν τον τρόπο που χρησιμοποιούσαν οι χρήστες τα τηλέφωνα και άνοιξαν τις πόρτες στην ανάπτυξη εφαρμογών. Αυτό είχε σαν αποτέλεσμα οι καταναλωτές να απαιτούν περισσότερα και οι προγραμματιστές εφαρμογών θα έκαναν πολύ καιρό ακόμη για να καλύψουν τις ανάγκες στους.

Αρχικά, οι χρήστες κινητών είχαν πρόσβαση σε απλές εφαρμογές όπως αριθμομηχανές, δημιουργούς ήχων κλήσης, βασικά παιχνίδια arcade και ημερολόγια αν και μερικές φορές ήταν δύσκολες στη χρήση. Ο απόλυτος ανταγωνισμός ήταν τη δεκαετία του '80 και του '90 στην αγορά κινητής τηλεφωνίας οι κατασκευαστές κινητών τηλεφώνων φύλαγαν προσεκτικά τα μυστικά τους και δεν άφησαν τις πλατφόρμες τους ανοιχτές για ανάπτυξη. Ως αποτέλεσμα, η ανάπτυξη εφαρμογών πραγματοποιήθηκε μόνο εσωτερικά (4).

Ωστόσο, οι εταιρείες κινητής τηλεφωνίας προσπάθησαν να προσφέρουν στους πελάτες περισσότερες εφαρμογές μέσω του Διαδικτύου χρησιμοποιώντας το Wireless Application Protocol (WAP). Σχεδιάστηκε για να αντιμετωπίσει τα προβλήματα που είχαν τα τηλέφωνα της δεκαετίας του 90, όπως περιορισμένη ισχύς επεξεργασίας, μονόχρωμες οθόνες και λίγη

αποθήκευση. Ενώ στην αρχή φαινόταν υπέροχο, οι πελάτες είδαν τα μειονεκτήματα γρήγορα, συμπεριλαμβανομένων εφαρμογών που περιορίζονται σε εφαρμογές ενός κατασκευαστή, υψηλές χρεώσεις χρήσης και σύνθετες μεθόδους λήψης εφαρμογών. Από την στιγμή που υπήρξε πληθώρα κινητών τηλεφώνων, το κόστος ανάπτυξης μειώθηκε και κάποιες συσκευές χρησιμοποιούσαν διάφορες γνωστές πλατφόρμες για ανάπτυξη εφαρμογών, όπως Linux και Windows. Έτσι άνοιξε τις πόρτες σε μια νέα γενιά ανάπτυξης εφαρμογών (5).

## 1.3 The Rise of Mobile Apps

Η IBM το 1993 τόλμησε να δημιουργήσει κάτι νέο μια κινητή συσκευή με οθόνη αφής. Ονομάστηκε Simon, περιελάμβανε πρόσβαση σε προεγκατεστημένες εφαρμογές για τις εργασίες που ήθελαν περισσότερο οι χρήστες, όπως ημερολόγιο, σημειωματάριο, ρολόι, email, επαφές, παιχνίδια και ακόμη και φαξ. Ήταν συσκευή ευκολότερη στη χρήση καθώς και η πρόσβαση σε εφαρμογές με καλύτερη εμφάνιση, την έκανε εξαιρετικά δημοφιλή. Η RIM βασίστηκε σε αυτήν την επιτυχία με το πρώτο Blackberry που ήταν αφιερωμένο στο email, μια ακόμη ευρέως χρησιμοποιούμενη εφαρμογή. Οι χρήστες δεν θα είχαν πραγματικά γεύση από αυτό που πραγματικά ήθελαν μέχρι το 2007 και το 2008, όταν η Apple έφερε στην αγορά το πρώτο iPhone που ακολουθείται από το App Store Ξαφνικά, οι χρήστες είχαν άμεση πρόσβαση σε μια ολόκληρη αγορά εφαρμογών με εύκολη πρόσβαση και εγκατάσταση (6).

Ενώ ακόμη οι επιλογές ήταν περιορισμένες, το μέλλον ξαφνικά έγινε σαφές για τις εταιρείες ανάπτυξης εφαρμογών, χρειαζόταν μόνο να δώσει στους χρήστες την ποικιλία και ευκολία χρήσης που ζητούν από τη δεκαετία του '80. Η αγορά Android ακολούθησε λίγους μήνες αργότερα. Η HTC τον ίδιο μήνα κυκλοφόρησε το πρώτο εμπορικά διαθέσιμο Android τηλέφωνο, δίνοντας στην Apple ανταγωνισμό λίγους μήνες μετά τις δικές τους κυκλοφορίες. Και οι δύο αγορές αυτές έφτασαν γρήγορα σε ένα δισεκατομμύριο λήψεις εφαρμογών με την Apple να σημειώνει το 2009 και το Android μόλις ένα χρόνο αργότερα. Κατά τη διάρκεια αυτής της περιόδου, η Apple άλλαξε επίσης το παιχνίδι απελευθερώνοντας το iPad, δίνοντας στους χρήστες έναν ακόμη τρόπο να χρησιμοποιούν εφαρμογές.

## 1.4 Αύξηση του ανταγωνισμού

Μέχρι το 2011, υπήρχαν πάνω από ένα εκατομμύριο διαθέσιμες εφαρμογές και η χρήση εφαρμογών ξεπέρασε τελικά τη χρήση ιστού για κινητά. Η Apple και η Google είδαν τη

δημοτικότητα των αγορών τους να αυξάνεται με πάνω από 15 δισεκατομμύρια λήψεις χρόνο με το χρόνο.

Μερικές δημοφιλείς εφαρμογές όπως το Instagram και το Draw Something ξεπέρασαν τα 50 εκατομμύρια λήψεις και το ιογενές παιχνίδι Angry Birds, έχει φτάσει πάνω από ένα δισεκατομμύριο λήψεις. Οι εταιρείες ανάπτυξης εφαρμογών διευκολύνονται από τις ανοιχτές πλατφόρμες για να εμπορεύονται και να δημιουργούν τις εφαρμογές τους σε πολλές συσκευές. Ωστόσο, η ποικιλία συσκευών και λειτουργικών συστημάτων καθιστά δύσκολη τη διασφάλιση ότι οι εφαρμογές είναι πάντα συμβατές. Ένα παράδειγμα που αντιμετωπίζουν σήμερα οι εταιρείες ανάπτυξης εφαρμογών που έχουν στόχο το Android είναι οι διαφορετικές εκδόσεις λειτουργικού συστήματος Android.

Οι εταιρείες ανάπτυξης εφαρμογών βρίσκονται σε συνεχή ανταγωνισμό, αλλά κάθε εταιρεία έχει το ίδιο διαθέσιμο κοινό για κινητές συσκευές. Βλέποντας το ιστορικό ανάπτυξης εφαρμογών, οι προγραμματιστές εφαρμογών μπορούν να μάθουν τι είδους εφαρμογές θέλουν περισσότερο οι χρήστες. Με κάθε εφαρμογή που γίνεται δημοφιλής, οι προγραμματιστές έχουν μια ακόμη ευκαιρία να δημιουργήσουν μια καλύτερη έκδοση. Με κάθε συσκευή που κυκλοφορεί, οι προγραμματιστές έχουν έναν νέο τρόπο εμφάνισης εφαρμογών.

## 1.5 Mobile Apps Today

Από τη στιγμή που ξεκίνησαν τα ακατέργαστα παιχνίδια, στα κινητά οι εφαρμογές έχουν παρουσιάσει αλλαγές. Στόχος τους είναι οι χρήστες να αισθάνονται απαραίτητη τη χρήση τους. Έτσι, διαπιστώθηκε ότι ξοδεύουν το 87% του χρόνου τους σε εφαρμογές και το 13% του χρόνου στο διαδίκτυο για τα κινητά. Ακολούθως, το 2015 οι επιχειρήσεις προσπάθησαν να ακολουθήσουν το πλάνο “πρώτα για κινητά”. Στη πορεία αυτό αποδείχτηκε σωτήριο, διότι με τις εφαρμογές οι επιχειρήσεις κατάφεραν σε σημαντικό βαθμό να αλλάξουν τη ζωή των πελατών τους. Ακόμη, οι πελάτες μπορούν ανά πάσα στιγμή και από κάθε χώρο να έχουν γρήγορη και εύκολη πρόσβαση μέσω των εφαρμογών στα κινητά τους.

Η Unifunds διαπιστώνει την τεράστια χρησιμότητα των εφαρμογών στα κινητά αλλά και την σημαντική εξέλιξη τους στο μέλλον. Επιπρόσθετα, το διαδίκτυο αποτελεί σημείο καμπής στην ανάπτυξη των επιχειρηματικών δραστηριοτήτων και της χρήσης τους διαδικτύου για επιπλέον ενημέρωση και ορθή καθοδήγηση (7).

## 1.6 ANDROID - Τι είναι και λίγα λόγια για αυτό


*Figure 2: Android*

Με μια σύντομη ανασκόπηση στον χρόνο μπορούμε να διαπιστώσουμε ότι το Android δημιουργήθηκε σε πρώτο χρόνο από την Google και αργότερα από την Open Handset Alliance. Θεωρείται το πιο δημοφιλές λογισμικό παγκοσμίως, αν αναλογιστεί κανείς ότι οι συσκευές με Android ξεπερνούν σε πωλήσεις όλες τις άλλες συσκευές μαζί (Windows, iOS και Mac OS X). Τρέχει τον πυρήνα του λειτουργικού Linux και χρησιμοποιείται από κινητές συσκευές. Ο κώδικας γράφεται σε γλώσσα Java με τους προγραμματιστές να αξιοποιούν μια πληθώρα βιβλιοθηκών που προσφέρει η Google. Αν και το λογισμικό Android αφορά κατά βάση συσκευές με οθόνη αφής , όπως smart phones, tablets, οθόνες αυτοκινήτων, τηλεοράσεις και smart watches, χρησιμοποιείται και σε άλλες ηλεκτρονικές συσκευές (όπως π.χ. φωτογραφικές μηχανές, H/Y, gaming κονσόλες κτλ.) (1, 2).

Η πρώτη παρουσίαση του νέου αυτού λογισμικού έγινε στις 5 Νοεμβρίου 2007. Η Google κοινοποίησε σχεδόν όλο το τμήμα του κώδικα του Android υπό τους όρους μιας ελεύθερης άδειας λογισμικού (Apache License). Το λογότυπο για το λογισμικό Android το σχεδίασε η Ιρίνα Μπλόκ και απεικονίζει ένα ρομπότ σε χρώμα πράσινου μήλου.

## 1.7 Γιατί ανάπτυξη σε Android και όχι σε IOS

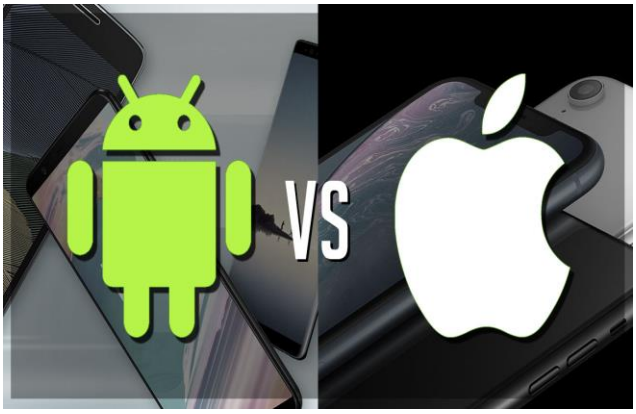



*Figure 3: Android*

Διότι όπως αναφέρθηκε και παραπάνω το Android είναι ένα πλήρες, ελεύθερο και προσβάσιμο λογισμικό για φορητές συσκευές που εμπεριέχει ένα λειτουργικό σύστημα (OS), βιβλιοθήκες, το αναγκαίο ενδιάμεσο λογισμικό και βασικές εφαρμογές. Αξιοποιώντας το Android System Development Kit οι προγραμματιστές έχουν τη δυνατότητα να δημιουργήσουν Android λογισμικό με τη χρήση της γλώσσας Java. Συζητάμε λόγους για τους οποίους πιστεύουμε ότι οι προγραμματιστές πρέπει πρώτα να αναπτύξουν εφαρμογές για συσκευές Android, αντί για iOS, Windows ή / και Blackberry (3).

***Φορητότητα***

Οι εγγενείς εφαρμογές Android αναπτύσσονται χρησιμοποιώντας τη γλώσσα προγραμματισμού Java και μπορούν εύκολα να μεταφερθούν σε άλλα λειτουργικά συστήματα για κινητά όπως το Blackberry, το Symbian και το Ubuntu. Επιπλέον, οι εφαρμογές Android μπορούν επίσης να μεταφερθούν εύκολα στο Chrome OS. Δεν αποτελεί έκπληξη το γεγονός ότι η Microsoft ανακοίνωσε επίσης ότι θα παρέχει μια εύκολη μέθοδο μεταφοράς εφαρμογών Android σε συσκευές Windows 10.

***Διαδραστικότητα***

Ένας προγραμματιστής Android δεν έχει περιορισμούς ως προς την πολυπλοκότητα, την καινοτομία και την λειτουργικότητα των εφαρμογών που κατασκευάζει. Οι εφαρμογές Android μπορούν να μεταφέρουν δεδομένα από και προς το κινητό μέσω διαδικτύου. Έτσι, μπορούν να ενημερώνονται εφαρμογές όπως το ημερολόγιο, οι επαφές, οι φωτογραφίες, το gps κτλ.

**Profitability**

Η γενική συναίνεση ήταν πάντοτε ότι το iPhone χρησιμοποιείται από πλουσιότερους και πιο εύπορους χρήστες, και έτσι, οι χρήστες iPhone είναι πιο πιθανό να ξοδέψουν χρήματα σε εφαρμογές από τους χρήστες Android. Αυτό μπορεί να ισχύει στο παρελθόν, αλλά όχι πια. Στις περισσότερες κατηγορίες εφαρμογών, οι εφαρμογές Android βρέθηκαν να είναι τόσο επικερδείς (ακόμη πιο επικερδείς σε ορισμένες περιπτώσεις) όσο οι εφαρμογές iPhone, τόσο για αρχικές αγορές εφαρμογών όσο και για αγορές εντός εφαρμογής.

Επίσης, με πολλές εφαρμογές να χρησιμοποιούν ένα μοντέλο δωρεάν με διαφημίσεις, εφόσον οι διαφημίσεις εμφανίζονται στους χρήστες της εφαρμογής, η εφαρμογή δημιουργεί έσοδα. Ο μέσος όρος εσόδων ανά χρήστη για παιχνίδια Android ήταν ελαφρώς 20% αυτού από τα παιχνίδια iOS τον Ιανουάριο του 2014. Μέχρι τον Δεκέμβριο του 2014, ο αριθμός είχε φτάσει στο 65%. Επιπλέον, το κόστος διαφήμισης είναι γενικά χαμηλότερο σε συσκευές Android, πράγμα που σημαίνει ότι οι εφαρμογές μπορούν να διαφημίζονται σε περισσότερους χρήστες σε συσκευές Android από χρήστες σε συσκευές iOS για το ίδιο ποσό.

***Απλούστερη ανάπτυξη εφαρμογών κινητών***

Το λογισμικό Android παρέχει μια πληθώρα βιβλιοθηκών οι οποίες μπορούν να αξιοποιηθούν από τον προγραμματιστή για τη δημιουργία νέων και καινοτόμων εφαρμογών. Παρέχει, επίσης, βοηθητικά εργαλεία που μπορούν να οδηγήσουν στην ανάπτυξη πιο περίπλοκου λογισμικού (δηλαδή ταχύτερου και με όσο το δυνατόν λιγότερα σφάλματα).

# ΚΕΦΑΛΑΙΟ – 2 – ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΣ ΣΕ ΠΕΡΙΒΑΛΛΟΝ ANDROID

## 2.1 Εισαγωγή Κεφαλαίου

Οι γλώσσες προγραμματισμού δημιουργήθηκαν για να ελέγχουν μηχανές όπως έναν ηλεκτρονικό υπολογιστή (Yoav, 2016). Μία γλώσσα προγραμματισμού όπως και οι φυσικές γλώσσες, έχουν συντακτικούς και εννοιολογικούς κανόνες. Υπάρχουν διάφορες κατηγορίες γλωσσών προγραμματισμούς. Κάποιες κατηγορίες γλωσσών είναι οι εξής (1, 2).

- ✓ Διαδικαστικές γλώσσες
- ✓ Αντικειμενοστραφείς γλώσσες
- ✓ Συναρτησιακές γλώσσες.
- ✓ Γλώσσες μηχανής
- ✓ Assembly
- ✓ Προγραμματισμού ιστοσελίδας
- ✓ Βάσεων δεδομένων

Το 1940 δημιουργήθηκαν, οι πρώτοι σύγχρονοι ηλεκτρονικοί υπολογιστές .Η περιορισμένη ταχύτητα και μνήμη που κατείχαν, οδήγησαν τους προγραμματιστές στην συγγραφή προγραμμάτων γλώσσας *assembly* και στην συνεχή ρύθμιση τους από τους ιδίους κάτι το οποίο απαιτούσε μεγάλη πνευματική προσπάθεια (Wayner, 2015). Οι πρώτες λειτουργικές γλώσσες προγραμματισμού που σχεδιάστηκαν για να δίνουν εντολές σε ηλεκτρονικό υπολογιστή, δημιουργήθηκαν στις αρχές του 1950.

Σε αντίθεση με τη γλώσσα μηχανής, τα μικρά κομμάτια κώδικα αντιπροσωπεύαν μαθηματικές εκφράσεις σε κατανοητή μορφή. Ο κώδικας όμως έπρεπε να μεταφράζεται σε γλώσσα μηχανής, κάθε φορά που εκτελούταν κάνοντας την διαδικασία πιο αργή. Αυτό το πρόβλημα λύθηκε στις αρχές της δεκαετίας του 1950 όταν ο Alick Glennie ανέπτυξε το Autocode την πιθανώς πρώτη μεταγλωττισμένη γλώσσα προγραμματισμού.

Έπειτα την περίοδο 1955-1959 στις ΗΠΑ, η Grace Hopper δημιούργησε την γλώσσα FLOW-MATIC. Ο μεταγλωττιστής της γλώσσας έγινε διαθέσιμος στις αρχές του 1958 και ολοκληρώθηκε το 1959. Γλώσσες που χρησιμοποιούνται μέχρι σήμερα είναι η LISP που

δημιουργήθηκε το 1958 από τον John McCarthy και η COBOL που δημιουργήθηκε από την Sort range committee. Το 1960, δημιουργήθηκε η Simula στο Νορβηγικό Υπολογιστικό Κέντρο στο Όσλο από τους Ole-Johan Dahl και Kristen Nygaard. Η Simula είναι όνομα δυο γλωσσών προγραμματισμού προσομοίωσης της Simula I και Simula 67.Η πρώτη έκδοση της Simula (1962)όπως υποδηλώνει και το όνομα της σχεδιάστηκε για να κάνει προσομοιώσεις ωστόσο θεωρείται η πρώτη αντικειμενοστραφής γλώσσα προγραμματισμού.

Συγκεκριμένα, η Simula 67 εισήγαγε την έννοια του αντικειμένου της κλάσης της κληρονομικότητας και της υποκλάσης. Τέλος, το 1964 στο Dartmouth College δημιουργήθηκε η γλώσσα BASIC (Beginner's All-purpose Sympolic Instruction Code) από τους John G. Kemeny και Thomas E. Kurtz. Η δημιουργία της BASIC είχε ως στόχο να δώσει τη δυνατότητα σε μαθητές που δεν ανήκαν σε επιστημονικά πεδία να χρησιμοποιήσουν ηλεκτρονικούς υπολογιστές. Εκείνη της εποχή σχεδόν όλοι οι υπολογιστές απαιτούσαν την συγγραφή προσαρμοσμένου λογισμικού κάτι το οποίο μέχρι τότε γνώριζαν μόνο επιστήμονες και μαθηματικοί (Gewirtz, 2017). Το 1990, μειώθηκε η χρήση της BASIC καθώς δημιουργήθηκαν ισχυροί μικροϋπολογιστές που απαιτούσαν γλώσσες προγραμματισμού με πιο προηγμένα χαρακτηριστικά.

Η πρώτη γλώσσα προγραμματισμού υψηλού επιπέδου, ήταν η Plankalkul που δημιουργήθηκε από τον Konrad Zuse μεταξύ 1942-1945 αλλά δεν εφαρμόσθηκε εκείνη την εποχή. Η πρώτη γλώσσα υψηλού επιπέδου η οποία είχε συσχετισμένο μεταγλωττιστεί, δημιουργήθηκε από τον Corrado Bohm. Η πρώτη εμπορικά διαθέσιμη γλώσσα ήταν η Fortran η οποία αναπτύχθηκε το 1954 με το πρώτο εγχειρίδιο να εμφανίζεται το 1956 από μια ομάδα με επικεφαλής τον John Backus (Wexelblat, 2014).

Ήταν η πρώτη ευρέως χρησιμοποιούμενη γλώσσα προγραμματισμού γενικού σκοπού, υψηλού επιπέδου που είχε λειτουργική υλοποίηση σε αντίθεση με μία απλή σχεδίαση σε χαρτί. Στην αρχή αντιμετωπίστηκε με σκεπτικισμό λόγω σφαλμάτων, καθυστερήσεων στην ανάπτυξη και συγκρίσεων με τον μέχρι τότε χειροκίνητων προγραμμάτων που χρησιμοποιούσαν assembly. Με την ανάπτυξη του hardware όμως η Fortran αποδείχθηκε αποτελεσματική και χρησιμοποιείται μέχρι και σήμερα σε υπολογιστές υψηλής απόδοσης (Pontin, 2018).

## 2.2 ANDROID SOFTWARE DEVELOPMENT

Μέσω του λογισμικού Android κατασκευάζονται εφαρμογές για συσκευές που εκτελούν το λειτουργικό σύστημα Android. Με βάση την Google οι εφαρμογές Android μπορούν να γραφτούν κατά βάση στις εξής γλώσσες: Kotlin, Java και C ++ χρησιμοποιώντας το SDK. Όλες

οι γλώσσες εκτός JVM, όπως Go, JavaScript, C, C ++ ή συγκρότημα, έχουν ανάγκη τη συμβολή του κώδικα γλώσσας JVM, που μπορεί να δίνεται από εργαλεία, ενδεχομένως με περιορισμένη υποστήριξη API. Ο συμβατικός τρόπος παράδοσης εφαρμογών Android στους τελικούς χρήστες είναι το Google Play, μέσω του οποίου μπορούν να κυκλοφορήσουν σταδιακά οι εφαρμογές ή να διανεμηθούν σε δοκιμαστές (7).

## 2.3 PROGRAMMING LANGUAGES

### 1. JAVA

*Figure 4: Java*

Η Java είναι μια αντικειμενοστραφή γλώσσα προγραμματισμού πολλαπλών πλατφορμών που κυκλοφόρησε από την Sun Microsystems το έτος 1995. Η Java σήμερα είναι απαραίτητη για την εκτέλεση διαφόρων εφαρμογών όπως παιχνίδια, , εφαρμογές ήχου και βίντεο, εφαρμογές κοινωνικών μέσων κ.λπ. Είναι αντικειμενοστραφής γλώσσα παρόμοια με την C ++, αλλά με προηγμένες και απλουστευμένες δυνατότητες. Αυτή η γλώσσα είναι ελεύθερη για πρόσβαση και μπορεί να εκτελεστεί σε όλες τις πλατφόρμες. Η Java έχει κάνει ευκολότερη τη ζωή αφαιρώντας όλες τις πολυπλοκότητες, όπως δείκτες, υπερφόρτωση χειριστή όπως βλέπει κανείς στη C ++ ή οποιαδήποτε άλλη γλώσσα προγραμματισμού (4, 5).

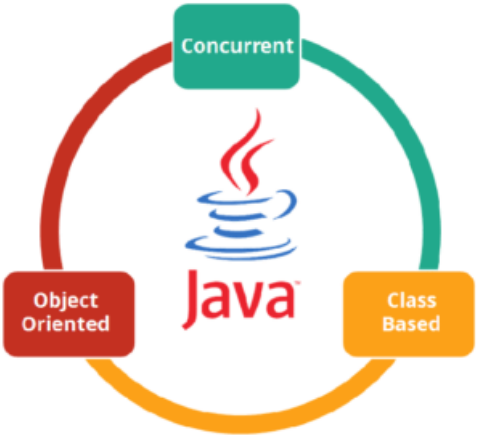



*Figure 5: Java 2*

**Σε τι χρησιμοποιείται η Java;**

Μερικές από τις εφαρμογές παρατίθενται παρακάτω:

- Τραπεζική: Για να ασχοληθείτε με τη διαχείριση συναλλαγών:
- Οι εφαρμογές χρέωσης που βλέπετε σε ένα κατάστημα / εστιατόριο είναι εντελώς γραμμένες στην Java.
- Πληροφορική: Η Java έχει σχεδιαστεί για την επίλυση εξαρτήσεων εφαρμογής.
- Android: Οι εφαρμογές γράφονται είτε σε Java είτε χρησιμοποιούν Java API.
- Χρηματοοικονομικές υπηρεσίες: Χρησιμοποιείται σε εφαρμογές διακομιστή.
- Χρηματιστήριο: Για να γράψετε αλγόριθμους σε ποια εταιρεία πρέπει να επενδύσουν.
- Big Data: Το πλαίσιο Hadoop MapReduce γράφεται χρησιμοποιώντας Java.
- Επιστημονική και ερευνητική κοινότητα: Για την αντιμετώπιση τεράστιου όγκου δεδομένων.

Τα εργαλεία εκείνη την εποχή ήταν γλώσσες όπως η γλώσσα προγραμματισμού C++ και η C (6). Μετά από διάφορα πειράματα, εξήλθε το συμπέρασμα ότι οι υπάρχουσες *γλώσσες* δεν μπορούσαν να καλύψουν τις ανάγκες τους. Ο Γκόσλινγκ, ο «πατέρας» της Java που εργαζόταν στη *Sun* εκείνη την εποχή, είχε ήδη αρχίσει να πειραματίζεται με τη γλώσσα προγραμματισμού C++ και περιστασιακά πρόσφερε κάποια πειραματική γλώσσα (C++++), ως πρότυπο για νέα εργαλεία που αναζητούσαν στη Sun. Μετά από λίγη ώρα κατέληξαν σε μια πρόταση για τους υπαλλήλους της εταιρείας που ήταν μια γλώσσα Oak. Πήρε το όνομά του από το ομώνυμο δέντρο (βελανιδιά) που φαίνεται καθημερινά έξω από το γραφείο του Γκόσλινγκ. Ωστόσο, ένα από τα βασικά πλεονεκτήματα της Java σε σχέση με τις περισσότερες άλλες γλώσσες, είναι το λειτουργικό σύστημα και η ανεξαρτησία της πλατφόρμας.

Τα προγράμματα που είναι γραμμένα σε Java εκτελούνται πανομοιότυπα σε Windows, Linux, Unix και Macintosh (*τρέχουν* στο Playstation και σε άλλες κονσόλες παιχνιδιών), χωρίς να χρειάζεται να γίνει εκ νέου μεταγλώττιση ή αλλαγή πηγαίου κώδικα για κάθε διαφορετικό λειτουργικό σύστημα. Αλλά για να συμβεί αυτό, χρειάζεται κάποιος τρόπος ώστε οποιοσδήποτε υπολογιστής να μπορεί να «κατανοήσει» ένα πρόγραμμα γραμμένο σε Java, ανεξάρτητα από τον τύπο επεξεργαστή (Intel x86, IBM, Sun SPARC, Motorola) και λειτουργικού συστήματος (Windows, Unix, Linux, BSD, MacOS).

Ο λόγος είναι ότι κάθε CPU κατανοεί διαφορετικό κώδικα μηχανής. Ο κώδικας συναρμολόγησης που μεταγλωττίζεται και εκτελείται στα Windows διαφέρει από τον κώδικα συναρμολόγησης που μεταγλωττίζεται και εκτελείται σε υπολογιστές Macintosh. Η λύση ήρθε με την ανάπτυξη εικονικών μηχανών (Virtual Machine ή VM ή ΕΜ στα ελληνικά).

Αφού γραφτεί ένα πρόγραμμα σε Java, μεταγλωττίζεται από τον μεταγλωττιστή javac για τη δημιουργία πολλών αρχείων .class (bytecode ή bytecode). Το Bytecode είναι η μορφή με την οποία μεταγλωττίζεται ο πηγαίος κώδικας Java. Όταν μια εφαρμογή πρόκειται να εκτελεστεί σε ένα μηχάνημα, η εικονική μηχανή Java, η οποία πρέπει να εγκατασταθεί στο μηχάνημα, θα είναι υπεύθυνη για την ανάγνωση των αρχείων .class (7, 8).

Στη συνέχεια, τα μεταφράζει στη γλώσσα προγραμματισμού που υποστηρίζεται από το λειτουργικό σύστημα και τον επεξεργαστή, ώστε να μπορεί να εκτελεστεί (πρόκειται για μια παραδοσιακή εικονική μηχανή). Οι πιο σύγχρονες εφαρμογές εικονικών μηχανών, μπορούν και κάνουν μεταγλώττιση από τμήματα bytecode υψηλού επιπέδου απευθείας σε κώδικα εντός της μηχανής (εγγενής κώδικας), αυξάνοντας έτσι την ταχύτητα.

Χωρίς αυτό, είναι αδύνατο να εκτελεστεί λογισμικό γραμμένο σε γλώσσα προγραμματισμού Java. Από την άλλη, οι χρήστες δεν μπορούν να *κατεβάσουν* «μη ορθό» κώδικα από το δίκτυο και να τον εκτελέσουν. Αυτό είναι ιδιαίτερα χρήσιμο για μεγάλα κατανεμημένα συστήματα όπου πολλοί χρήστες χρησιμοποιούν το ίδιο πρόγραμμα ταυτόχρονα.

Μια άλλη ιδέα πίσω από την Java είναι η ύπαρξη ενός συλλέκτη απορριμμάτων (garbage collector). Η συλλογή απορριμμάτων είναι μια γενική ονομασία που χρησιμοποιείται στους υπολογιστές για να υποδηλώσει την *απελευθέρωση* τμημάτων μνήμης από δεδομένα που δεν χρειάζονται και δεν χρησιμοποιούνται πλέον. Αυτή η κατανομή μνήμης σε Java είναι αυτόματη και γίνεται από τον συλλέκτη σκουπιδιών. Η ευθύνη για αυτό είναι και πάλι η

εικονική μηχανή, αφού «κατανοήσει» το σωρό μνήμης (στην γλώσσα προγραμματισμού Java η συντριπτική πλειονότητα των αντικειμένων αποθηκεύεται στο σωρό, σε αντίθεση με την C++ όπου αποθηκεύονται ως επί το πλείστων στη στοίβα) για να ενεργοποιήσει τον συλλέκτη σκουπιδιών.

Επομένως, ο προγραμματιστής δεν χρειάζεται να ανησυχεί για το πότε και αν θα ελευθερώσει ένα συγκεκριμένο κομμάτι μνήμης, ούτε χρειάζεται να ανησυχεί για σφάλματα δείκτη. Αυτό είναι ιδιαίτερα σημαντικό επειδή τα σφάλματα προγράμματος λόγω εσφαλμένου χειρισμού της μνήμης είναι κοινά.

Παρά όλα αυτά τα (και περισσότερα) πλεονεκτήματα που παρέχει μια εικονική μηχανή, η Java ήταν αρχικά πιο αργή από άλλες γλώσσες προγραμματισμού υψηλού επιπέδου όπως η γλώσσα προγραμματισμού C και η C++. Προηγούμενες εμπειρικές μετρήσεις δείχνουν ότι η C++ μπορεί να είναι αρκετές φορές ταχύτερη από την Java. Ωστόσο, η Sun εργάζεται για τη βελτιστοποίηση της εικονικής μηχανής και υπάρχουν άλλες υλοποιήσεις εικονικών μηχανών από διαφορετικές εταιρείες (όπως η IBM) που μπορεί να δώσουν καλύτερα αποτελέσματα σε ορισμένα μέρη και χειρότερα αποτελέσματα σε άλλα.

Επίσης, με την εισαγωγή των μεταγλωττιστών JIT (just-in-time) που μετατρέπουν τον bytecode απευθείας σε γλώσσα μηχανής, η διαφορά ταχύτητας με τη C++ έχει μειωθεί σημαντικά. Όλα τα εργαλεία που χρειάζονται για τη σύνταξη προγραμμάτων Java είναι δωρεάν, από περιβάλλοντα ανάπτυξης έως εργαλεία κατασκευής όπως το Apache Ant και βιβλιοθήκες, καθώς και πολλές διαφορετικές εικονικές μηχανές Java και υλοποιήσεις μεταγλωττιστών (όπως το GNU Compiler για Java).

Σύμφωνα με τα όσα αναφέρθηκαν παραπάνω λοιπόν, θα λέγαμε πως η εφημερίδα *Sun* περιγράφει την Java ως «*μια απλή, αντικειμενοστραφή, κατανεμημένη, ερμηνευμένη, συμπαγής, ασφαλής, ανεξάρτητη από αρχιτεκτονική, φορητή, υψηλής απόδοσης, πολυνηματική και δυναμική γλώσσα*».

## 2. KOTLIN

*Figure 6: Kotlin*

Το Kotlin είναι μια γλώσσα προγραμματισμού γενικής χρήσης πολλαπλής πλατφόρμας, με στατικό τύπο και συμπεράσματα. Το Kotlin έχει σχεδιαστεί για να λειτουργεί πλήρως με την Java και η έκδοση JVM της τυπικής βιβλιοθήκης του Kotlin εξαρτάται από τη Java Class Library , αλλά η συμπερίληψη τύπου επιτρέπει στη σύνταξή της να είναι πιο περιεκτική. Το Kotlin στοχεύει κυρίως την JVM, αλλά επίσης μεταγλωττίζεται σε JavaScript (π.χ. για εφαρμογές web frontend που χρησιμοποιούν το React) ή τον εγγενή κώδικα (μέσω LLVM), π.χ. για εγγενείς εφαρμογές iOS που μοιράζονται επιχειρηματική λογική με εφαρμογές Android (6).

Το κόστος ανάπτυξης της γλώσσας βαρύνει την JetBrains, ενώ το Ίδρυμα Kotlin προστατεύει το εμπορικό σήμα Kotlin. Στις 7 Μαΐου 2019, η Google ανακοίνωσε ότι η γλώσσα προγραμματισμού Kotlin είναι πλέον η προτιμώμενη γλώσσα της για προγραμματιστές εφαρμογών Android. Ως αποτέλεσμα, πολλοί προγραμματιστές έχουν μεταβεί στο Kotlin. Από την κυκλοφορία του Android Studio 3.0 τον Οκτώβριο του 2017, το Kotlin έχει συμπεριληφθεί ως εναλλακτική λύση του τυπικού μεταγλωττιστή Java. Ο μεταγλωττιστής Android Kotlin στοχεύει Java 6 από προεπιλογή, αλλά επιτρέπει στον προγραμματιστή να επιλέξει να στοχεύσει Java 8 έως 13, για βελτιστοποίηση,  ή περισσότερες δυνατότητες.

Η **JetBrains**, τον Ιούλιο του 2011 έκανε γνωστό ένα νέο Project στο οποίο έγινε χρήση του Project Kotlin για διάρκεια ενός έτους. Η **Kotlin** αποτέλεσε μία νέα γλώσσα για την **JVM**. Ο JetBrains Dmitry ανέφερε ότι εκτός από τη **Scala**, οι υπόλοιπες γλώσσες δεν πληρούσαν τις προϋποθέσεις που αναζητούσαν, όμως τόνισε ότι μειονέκτημα της Scala είναι ο αργός χρόνος σύνταξης. Από τους βασικούς στόχους της Kotlin είναι η σύνταξή της με την Java. Επίσης, η **JetBrains** εκκίνησε το project με την άδεια του **Apache 2**.

Η ονομασία προέρχεται από το νησί Kotlin, κοντά στην Αγία Πετρούπολη. Ο Αντρέι Μπρέσλαβ ανέφερε ότι η ομάδα αποφάσισε να το ονομάσει ένα νησί, όπως ακριβώς η Java πήρε το όνομά

της από το νησί της Ινδονησίας (αν και η γλώσσα προγραμματισμού Java ίσως πήρε το όνομά του από τον καφέ). Η JetBrains ελπίζει ότι η νέα γλώσσα θα οδηγήσει τις πωλήσεις IntelliJ IDEA. Στις 15 Φεβρουαρίου 2016 κυκλοφόρησε το Kotlin v1.0 . Η συγκεκριμένη έκδοση αποτελεί την πρώτη επίσημη κυκλοφορία και η JetBrains έχει ανακοινώσει ότι θα παρέχει μακροχρόνια υποστήριξη για τα προηγούμενα έτη από την συγκεκριμένη έκδοση (7).

Επιπλέον, στο Google I / O το 2017 η εταιρεία ανακοίνωσε ότι θα υποστηρίξει τη γλώσσα Kotlin σε Android. Το Kotlin v1.2 κυκλοφόρησε το Νοέμβριο του 2017. Αυτή η κυκλοφορία προστέθηκε πρόσφατα στην κοινή χρήση κώδικα κοινής χρήσης μεταξύ JVM και πλατφόρμας JavaScript (από την έκδοση 1.4 ο προγραμματισμός πολλαπλών μορφών είναι μια δυνατότητα άλφα που αναβαθμίστηκε από το "πειραματικό"). Έχει γίνει μια επίδειξη πλήρους στοίβας με το νέο Kotlin / JS Gradle Plugin. Στις 7 Μαΐου 2019, η Google ανακοίνωσε ότι η γλώσσα προγραμματισμού Kotlin είναι πλέον η προτιμώμενη γλώσσα της για προγραμματιστές εφαρμογών Android. Το Kotlin v1.4 κυκλοφόρησε τον Αύγουστο του 2020, με π.χ. κάποιες μικρές αλλαγές στην υποστήριξη για τις πλατφόρμες της Apple, δηλαδή στο Interop Objective-C / Swift (8).

**3. C++**

*Figure 7: C++*

Η C++ είναι intermediate level language (μέσου επιπέδου), δηλαδή εμπερικλείει μία σύζευξη ιδιοτήτων από high level languages και low level languages. Επίσης, είναι γλώσσα γενικού σκοπού καθώς μπορεί να καλύψει ευρύ πεδίο εφαρμογών. Επίσης, είναι μεταγλωττιζόμενη γλώσσα πολλαπλών παραδειγμάτων, με τύπους. Ο προγραμματισμός που μπορεί να υποστηρίξει είναι δομημένος, αντικειμενοστραφής και γενικός.

Η C++ δημιουργήθηκε από τον Bjarne Stroustrup (1979) και αποτελεί βελτιωμένη έκδοση της παλιότερης και ήδη υπάρχουσας γλώσσας C. Στην πραγματικότητα, πρόκειται για τη γλώσσα C με την προσθήκη κλάσεων (πχ. Simula). Αυτός είναι άλλωστε και ο λόγος που ονομάστηκε αρχικά “C with Classes”. Ο Rick Mascitti εισήγαγε για πρώτη φορά την ονομασία «C++» (1983) και από τότε καθιερώθηκε στον προγραμματιστικό χώρο. Η γλώσσα προγραμματισμού C++ σχεδιάστηκε για να είναι μια στατικά γλώσσα πληκτρολόγησης γενικής χρήσης, τόσο αποτελεσματική και φορητή όσο η C (Wayner, 2015).

Η γλώσσα προγραμματισμού C++ έχει σχεδιαστεί για να υποστηρίζει πολλά είδη προγραμματισμού άμεσα και παγκοσμίως. Η γλώσσα προγραμματισμού C++ έχει σχεδιαστεί για να δίνει στους προγραμματιστές επιλογές, ακόμα κι αν τους επιτρέπει να κάνουν λανθασμένες επιλογές.

Η γλώσσα προγραμματισμού C++ συνδυάζει χαρακτηριστικά γλωσσών υψηλού και χαμηλού επιπέδου. Είναι μια δακτυλογραφημένη, ελεύθερης μορφής, πολλαπλών παραδειγμάτων γλώσσα, της οποίας η μετάφραση (μεταγλώττιση) δημιουργεί κώδικα για συγκεκριμένο τύπο υλικού (15).

Τέλος, υποστηρίζει δομημένο, αντικειμενοστραφή και γενικό προγραμματισμό. Η γλώσσα αναπτύχθηκε από τον Bjarne Stroustrup στα εργαστήρια Bell της AT&T το 1979, ως βελτίωση της υπάρχουσας γλώσσας προγραμματισμού C, που αρχικά ονομαζόταν "C with Classes".

## 2.4 ANDROID STUDIO


*Figure 8: Android Studio*

Το πιο βασικό εργαλείο για την ανάπτυξη Android εφαρμογών είναι το Android SDK (software development kit) το οποιο μας παρε8χει τα επιπλεον εργαλει8α ω8στε να μπορου8με να

γράψουμε κώδικα συγκεκριμένα για την κατασκευή εφαρμογής Android. Συγκεκριμένα, το SDK περιλαμβάνει μια πλήρη σειρά από εργαλεία ανάπτυξης, συμπεριλαμβανομένων ενός προγράμματος εντοπισμού σφαλμάτων, βιβλιοθηκών λογισμικού, ενός εξομοιωτή βασισμένου στο QEMU, εγγράφων, δειγμάτων κώδικα, και παραδειγμάτων. Η εκκίνηση της εφαρμογής Android, μπορεί να ξεκινήσει είτε με την εγκατάσταση σε ένα smartphone είτε με την χρήση κάποιου εξομοιωτή κινητού τηλεφώνου. Στο AndroidSDK περιλαμβάνεται ένας εξομοιωτής κινητών συσκευών, μια εικονική, δηλαδή, φορητή συσκευή, η οποία εκτελείται στον υπολογιστή. Ο εξομοιωτής επιτρέπει την ανάπτυξη και τη δοκιμή εφαρμογών του Android χωρίς τη χρήση φυσικής συσκευής.

## 2.5 ΔΟΜΙΚΑ ΜΕΡΗ ΜΙΑΣ ΕΦΑΡΜΟΓΗΣ ANDROID

Η κάθε εφαρμογή Android περιέχει κάποια απαραίτητα δομικά στοιχεία. Κάθε στοιχείο είναι ένα διαφορετικό σημείο από το οποίο το σύστημα μπορεί να εισέλθει στην εφαρμογή μας. Υπάρχουν τέσσερις διαφορετικοί τύποι δομικών στοιχείων: δραστηριότητες (activities), υπηρεσίες (services), πάροχοι περιεχομένου (content providers) και παραλήπτες εκπομπών (broadcast receivers). Κάθε τύπος εξυπηρετεί ένα συγκεκριμένο σκοπό και έχει ένα ξεχωριστό κύκλο ζωής που καθορίζει το πώς το στοιχείο δημιουργείται και καταστρέφεται (10).

### 2.5.1 Δραστηριότητες - Activities

Η κατηγορία δραστηριοτήτων είναι ένα από τα πολύ σημαντικά μέρη του στοιχείου Android. Οποιαδήποτε εφαρμογή, ανεξάρτητα από το πόσο μικρή είναι (όσον αφορά τον κώδικα και την επεκτασιμότητα), έχει τουλάχιστον μία κλάση Δραστηριότητας. Σε αντίθεση με τις περισσότερες από τις γλώσσες προγραμματισμού, στις οποίες η κύρια μέθοδος είναι το σημείο εισόδου για να ξεκινήσει η εκτέλεση αυτού του προγράμματος ή της εφαρμογής, το λειτουργικό σύστημα android ξεκινά τον κώδικα σε μια παρουσία Δραστηριότητας επικαλούμενος συγκεκριμένες μεθόδους επανάκλησης που αντιστοιχούν σε συγκεκριμένα στάδια τον κύκλο ζωής του. Έτσι μπορεί να ειπωθεί ότι μια δραστηριότητα είναι το σημείο εισόδου για την αλληλεπίδραση με τον χρήστη. Κάθε δραστηριότητα περιέχει τη διάταξη, η οποία διαθέτει διεπαφή χρήστη για αλληλεπίδραση με τον χρήστη. Όπως γνωρίζουμε ότι κάθε δραστηριότητα περιέχει μια διάταξη που σχετίζεται με αυτήν, έτσι μπορούμε να πούμε ότι η κατηγορία δραστηριότητας είναι η πύλη, μέσω της οποίας ένας χρήστης μπορεί να αλληλοεπιδρά μέσω προγραμματισμού με το περιβάλλον χρήστη.

Η διάταξη για μια συγκεκριμένη δραστηριότητα ρυθμίζεται με τη βοήθεια του setContentView (). Το setContentView () είναι μια συνάρτηση που λαμβάνει το View ως παράμετρο. Η

παράμετρος προβολής περιέχει βασικά το αρχείο διάταξης για αυτήν τη δραστηριότητα. Οι παρακάτω εικόνες δείχνουν ότι το activity_main είναι το αρχείο διάταξης του MainActivity.

```
public class MainActivity extends AppCompatActivity {

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

*Figure 10: Main Activity Code*

### 2.5.2 Υπηρεσίες

Μια υπηρεσία είναι ένα στοιχείο που τρέχει στο παρασκήνιο για να εκτελέσει μμακροχρόνιες εργασίες  ή εργασίες για απομακρυσμένες διαδικασίες. Οι υπηρεσίες δεν παρέχουν διεπαφή χρήστη. Για παράδειγμα, μια υπηρεσία μπορεί να παίξει μουσική στο παρασκήνιο ενώ ο χρήστης βρίσκεται σε μια διαφορετική εφαρμογή, ή θα μπορούσε να φέρει δεδομένα μέσω δικτύου χωρίς να εμποδίζει την αλληλεπίδραση του χρήστη με κάποια δραστηριότητα. Οι υπηρεσίες εκκινούνται από άλλα στοιχεία όπως δραστηριότητες, οι οποίες στη συνέχεια τις αφήνουν να τρέξουν ή συνδέονται με αυτές, ούτως ώστε να αλληλοεπιδράσουν μεταξύ τους. Κάθε υπηρεσία υλοποιείται ως μια υποκλάση της κλάσης Service και διαθέτει δύο βασικές μεθόδους: την startService(11) και την bindService(11).

### 2.5.3 Πάροχοι Περιεχομένου – Content Providers

Ο ρόλος του παρόχου περιεχομένου είναι η διαχείριση της πρόσβασης σε ένα κεντρικό αποθετήριο δεδομένων. Επίσης, είναι τμήμα μιας Android εφαρμογής, η οποία πολλές φορές προσφέρει στο χρήστη το περιβάλλον για την διάδραση του με τα δεδομένα. Από κοινού, οι πάροχοι και οι πελάτες παροχών παρέχουν μια ακριβή, τυπική διάδραση με δεδομένα και καλύπτουν την επικοινωνία μεταξύ διεργασιών και την ασφαλή πρόσβαση στα δεδομένα (10, 11).

Ένας πάροχος περιεχομένου παρουσιάζει δεδομένα σε εξωτερικές εφαρμογές ως έναν ή περισσότερους πίνακες που είναι παρόμοιοι με τους πίνακες που βρίσκονται σε σχεσιακή βάση δεδομένων. Μια σειρά αντιπροσωπεύει μια παρουσία κάποιου τύπου δεδομένων που συλλέγει ο πάροχος και κάθε στήλη στη σειρά αντιπροσωπεύει ένα μεμονωμένο κομμάτι δεδομένων που συλλέγεται για μια παρουσία. Ένας πάροχος περιεχομένου συντονίζει την πρόσβαση στο

επίπεδο αποθήκευσης δεδομένων στην εφαρμογή σας για έναν αριθμό διαφορετικών API και στοιχείων όπως φαίνεται στο σχήμα 1, αυτά περιλαμβάνουν:

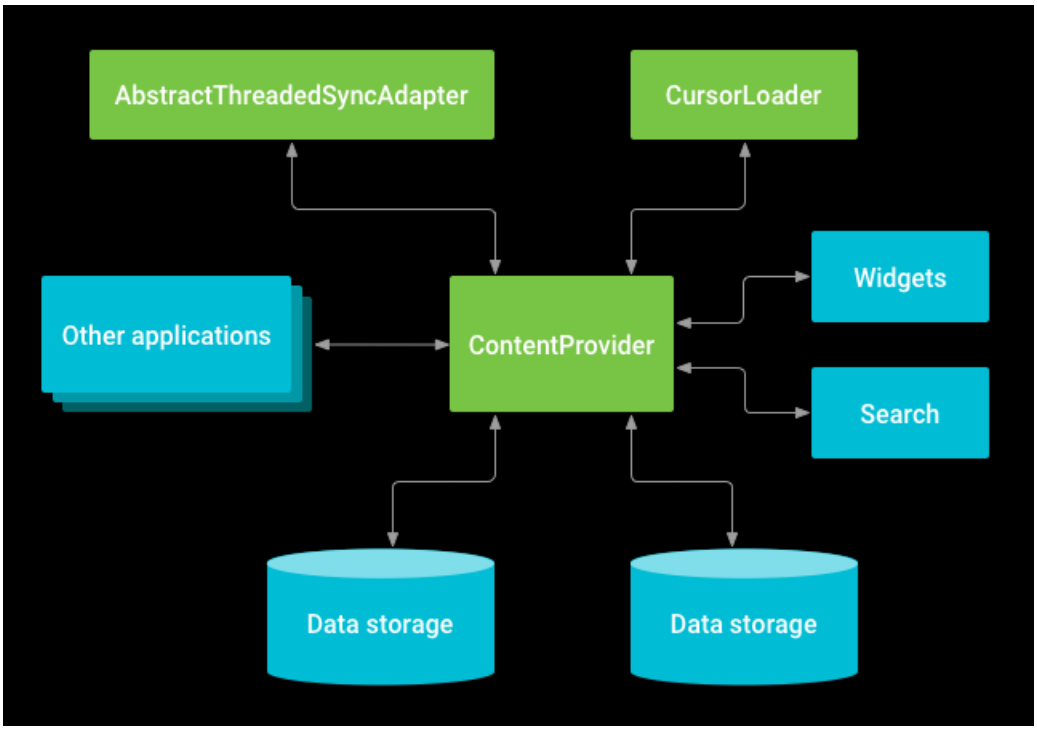



*Figure 11: Content Provider*

- Κοινή χρήση πρόσβασης στα δεδομένα της εφαρμογής σας με άλλες εφαρμογές
- Αποστολή δεδομένων σε widget
- Επιστροφή προσαρμοσμένων προτάσεων αναζήτησης για την εφαρμογή σας μέσω του πλαισίου αναζήτησης χρησιμοποιώντας το SearchRecentSuggestionsProvider
- Συγχρονισμός δεδομένων εφαρμογής με το διακομιστή σας χρησιμοποιώντας μια εφαρμογή AbstractThreadedSyncAdapter
- Φόρτωση δεδομένων στο περιβάλλον εργασίας σας χρησιμοποιώντας ένα CursorLoader

### 2.5.4 Παραλήπτες Εκπομπών

Όλοι οι εγγεγραμμένοι δέκτες για ένα συμβάν ειδοποιούνται από το χρόνο εκτέλεσης του Android μόλις συμβεί αυτό. Για παράδειγμα, οι εφαρμογές μπορούν να εγγραφούν στο συμβάν συστήματος ACTION_BOOT_COMPLETED που ενεργοποιείται μόλις το σύστημα Android ολοκληρώσει τη διαδικασία εκκίνησης.

Ένας παραλήπτης εκπομπών είναι ένα στοιχείο που ανταποκρίνεται σε όλο το σύστημα μετάδοσης ανακοινώσεων. Πολλές εκπομπές προέρχονται από το σύστημα, για παράδειγμα,

μία εκπομπή που ανακοινώνει ότι η οθόνη έχει σβήσει, ότι η μπαταρία είναι χαμηλή, ή ότι μια φωτογραφία τραβήχτηκε. Οιεκπομπές μπορεί να προέρχονται κι από εφαρμογές. Για παράδειγμα, για να ενημερώσει άλλες εφαρμογές ότι έχει κατεβάσει κάποια δεδομένα και είναι διαθέσιμα να τα χρησιμοποιήσουν.

## 2.6 ΑΡΧΕΙΟ ANDROID MANIFEST

AndroidManifest.xml: Το αρχείο αυτό παράγεται αυτόματα όταν δημιουργούμε το project. Δηλώνει τις βασικές πληροφορίες που χρειάζεται το σύστημα Android για να τρέξει την εφαρμογή: όνομα package, έκδοση, δραστηριότητες, άδειες, προθέσεις, ή απαιτούμενο υλικό.

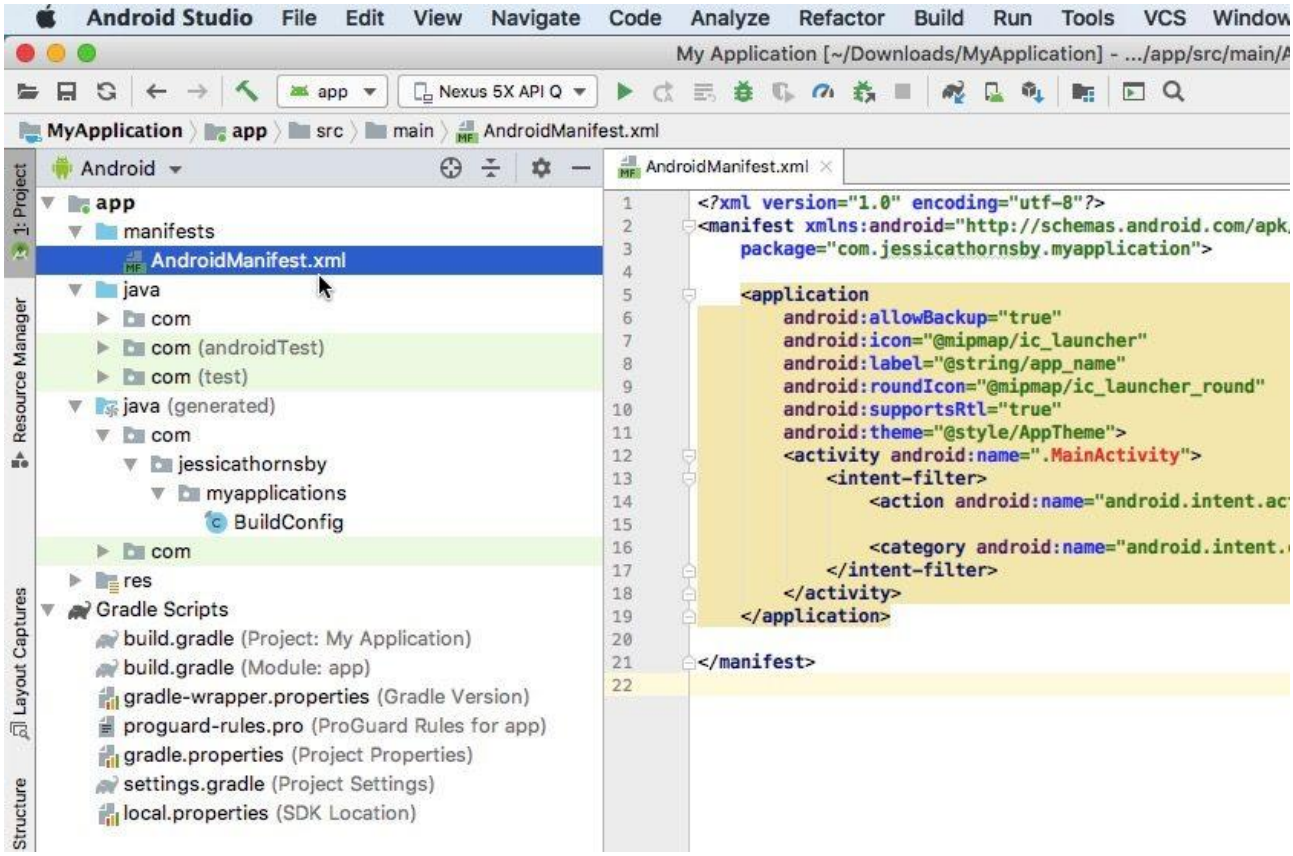


*Figure 12: Android Manifest XML*

### 2.6.1 Οι φάκελοι src και res

- src - Ο φάκελος src εμπεριέχει τα αρχεία Java που αντιστοιχούν σε όλες τις δραστηριότητες, υπηρεσίες, παρόχους περιεχομένου, βοηθητικά αρχεία, κλπ της εφαρμογής.
- res - Αρχεία πόρων που σχετίζονται με το έργο σας. Όλα τα γραφικά, οι συμβολοσειρές, οι διατάξεις και άλλα αρχεία πόρων αποθηκεύονται στην ιεραρχία αρχείων πόρων κάτω από τον κατάλογο res.

- res / layout - Αρχεία διάταξης XML που περιγράφουν τις προβολές και τις διατάξεις για κάθε δραστηριότητα και για μερικές προβολές, όπως στοιχεία λίστας. res / τιμές - XML αρχεία που αποθηκεύουν διάφορες τιμές χαρακτηριστικών Αυτά περιλαμβάνουν strings.xml, dimens.xml, styles.xml, colors.xml, themes.xml και ούτω καθεξής.
- res / drawable - Εδώ αποθηκεύουμε τα διάφορα γραφικά στοιχεία ανεξάρτητα από την πυκνότητα που χρησιμοποιούνται στην εφαρμογή μας.
- res / drawable-hdpi - Σειρά φακέλων για ειδικές εικόνες πυκνότητας για χρήση σε διάφορες αναλύσεις.
- res / mipmap - χρησιμοποιείται συνήθως για εικονίδια εφαρμογών.

# ΚΕΦΑΛΑΙΟ – 3 – ΑΡΧΙΤΕΚΤΟΝΙΚΗ ΤΗΣ ΕΦΑΡΜΟΓΗΣ

## 3.1 Εισαγωγή

Στο κεφάλαιο αυτό στην αρχή θα ασχοληθούμε με την χρήση και λειτουργία της εφαρμογής και τον τρόπο ανάπτυξής της για το λειτουργικό σύστημα Android με αναφορές σε δυνατότητες που παρέχει στον τελικό χρήστη. Στη συνέχεια θα αναλύσουμε τα αρχεία που χρησιμοποιούνται καθώς και επιμέρους ανάλυση του κάθε αρχείου ξεχωριστά. Ακόμη, θα αναλύσουμε την δομή της εφαρμογής και τις επιμέρους λειτουργικότητες που παρέχει.

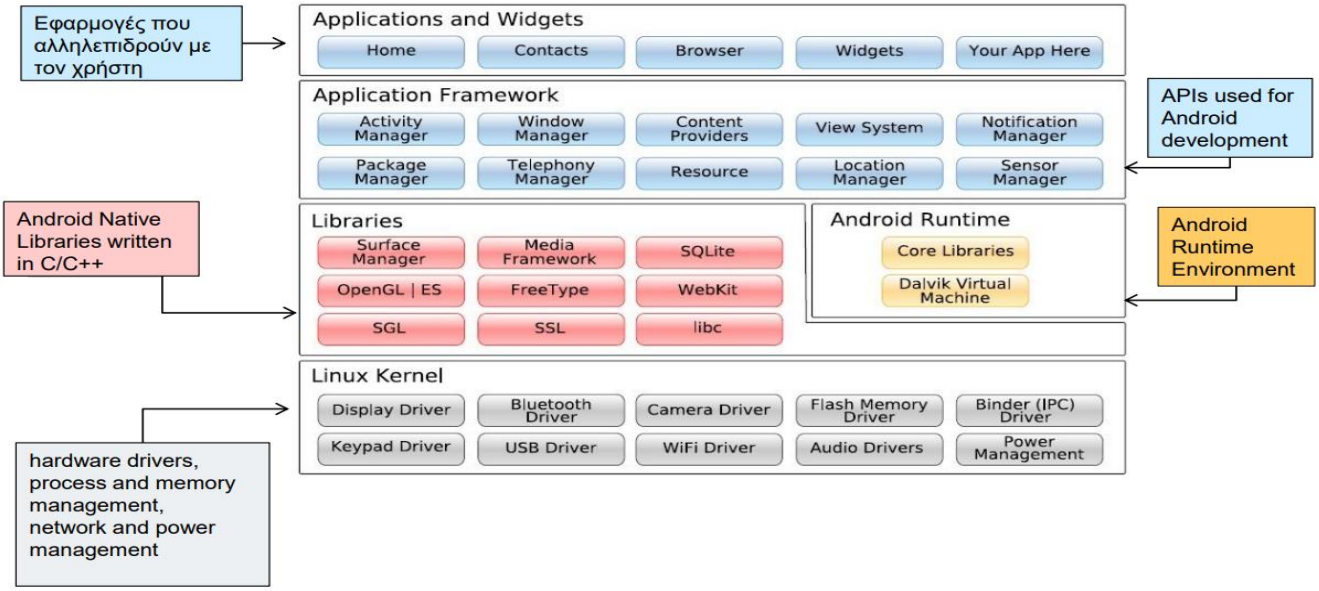
## 3.2 Περιγραφή

Κύριος σκοπός της εφαρμογής είναι να δημιουργηθεί ένα πρόγραμμα ανταλλαγής μηνυμάτων chatroom που παρέχει την δυνατότητα εισαγωγής όνομα χρήστη και κωδικού χρήστη για την είσοδο στο περιβάλλον ανταλλαγής μηνυμάτων μεταξύ των χρηστών το οποίο δίνει την δυνατότητα άμεσης αλληλεπίδρασης και επικοινωνίας μεταξύ των χρηστών. Με την εφαρμογή αυτή μπορεί ο μέσος χρήστης να συζητήσει με διάφορους χρήστες για θέματα που τον απασχολούν, είτε για εκπαιδευτικά, είτε ψυχαγωγικά θέματα, με την δυνατότητα άμεσης ανταλλαγής μηνυμάτων ο χρήστης έχει την δυνατότητα ανταλλαγής μηνυμάτων για διάφορα θέματα της καθημερινότητας και να επικοινωνήσει με διάφορους χρήστες αρκεί να έχουν κινητό android, την συγκεκριμένη εφαρμογή και πρόσβαση στο διαδίκτυο.

Έτσι ο χρήστης εφόσον κατεβάσει την εφαρμογή ή μεταφέρει το αρχείο εγκατάστασης εφαρμογής (application) .apk από τον υπολογιστή στην συσκευή του, έπειτα συνεχίζει με την εγκατάσταση της εφαρμογής και εφόσον αυτή ολοκληρωθεί επιτυχώς τότε μπορεί να ανοίξει την εφαρμογή, έχοντας πρόσβαση στο διαδίκτυο και να εισάγει το όνομα χρήστη και τον κωδικό για να εισέλθει στο chatroom application (13).

## 3.3 Δομή

### 3.3.1 Αρχιτεκτονική Android




**Διάγραμμα αρχιτεκτονικής**

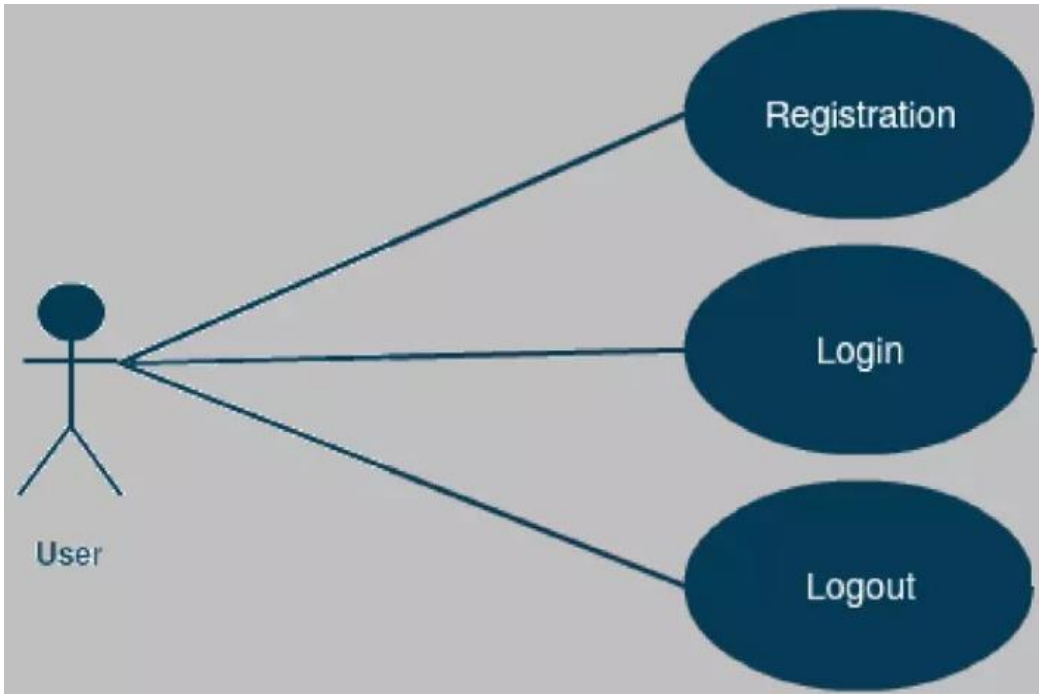
## User Case Diagrams

## User Case Table

| Level 0 | Level 1 | Level 2 | Actor |
|---|---|---|---|
| Chat Application | Authentication System | Registrar<br>Login<br>Logout | User |
| | Contacts Form | Friend List<br>Find Friend<br>Add Friend<br>Remove Friend<br>Block Friend | User |
| | Chat Form | Send Message<br>Group Chat<br>Best Friend | User |
| | Maintenance | User's Profile<br>Dtabase | Admin |
| | Monitor | Check History<br>Feedback | Admin User |

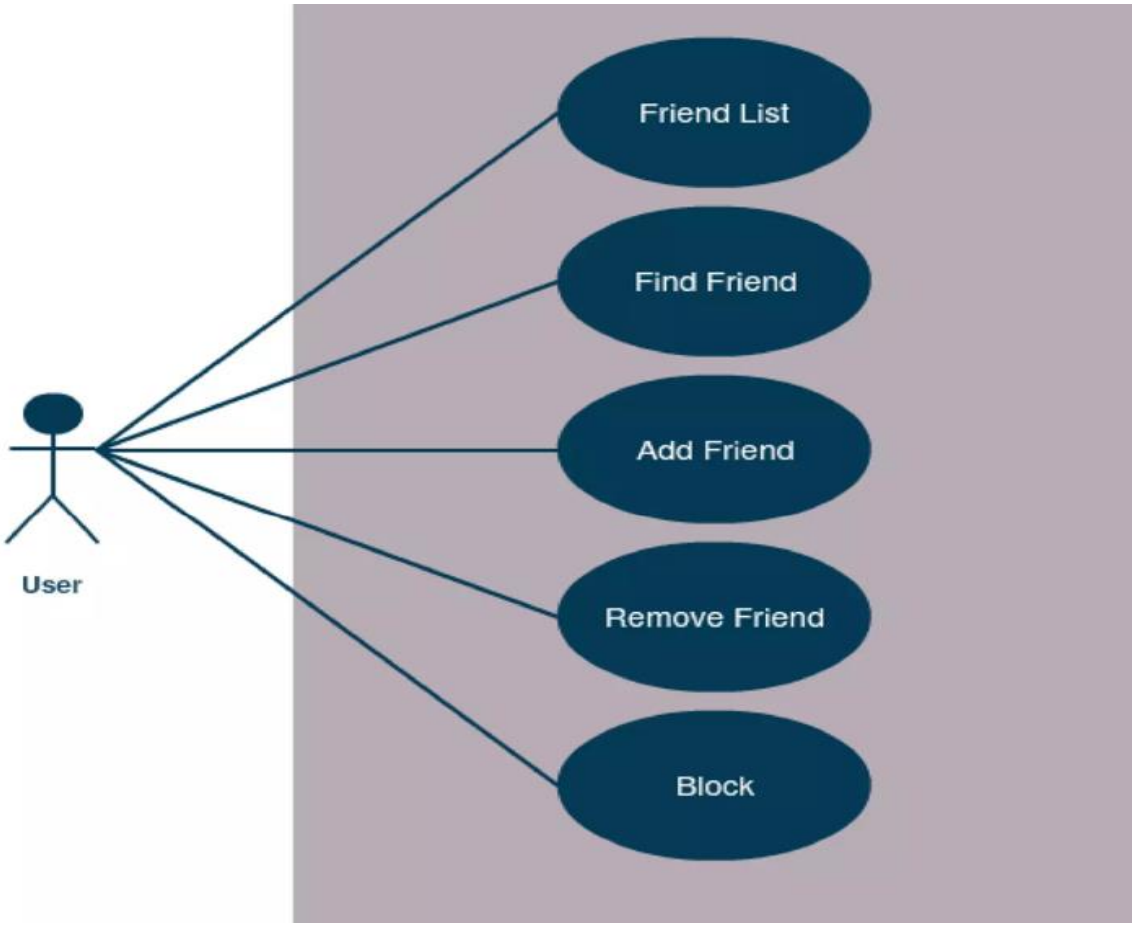
**User Case Table of Chat Application**

## Authentication System Diagram



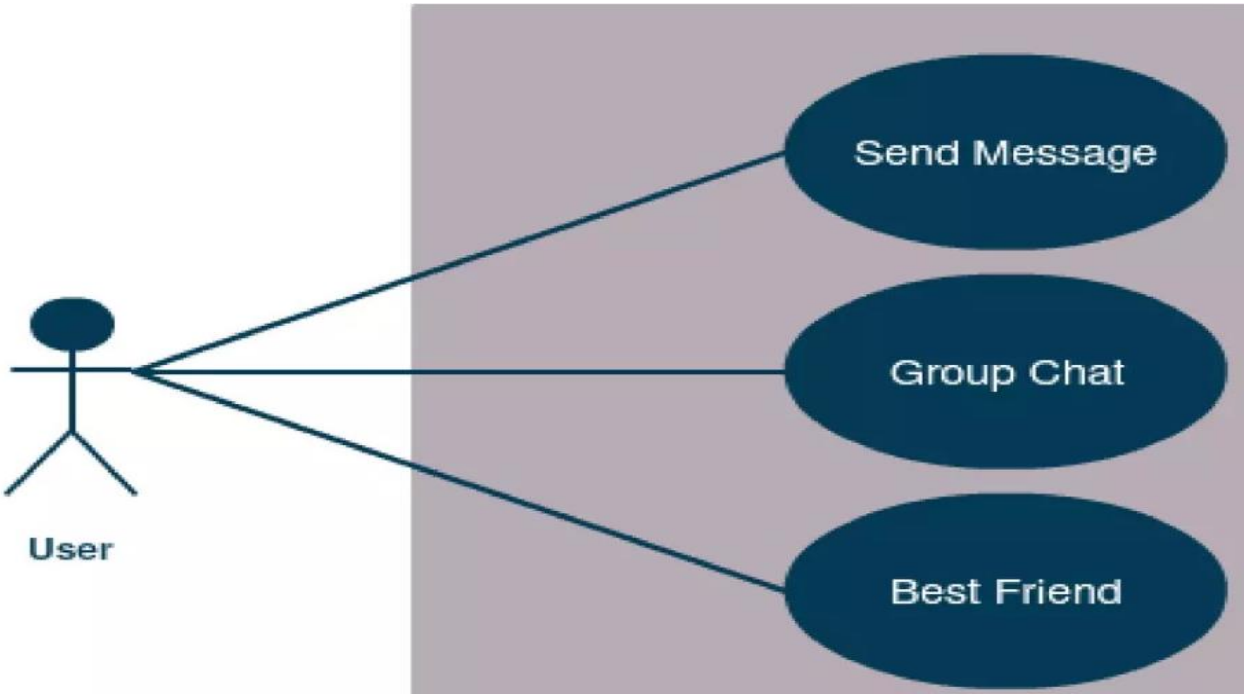
Use Case Diagram of Authentication System

**Contacts Form Diagram**



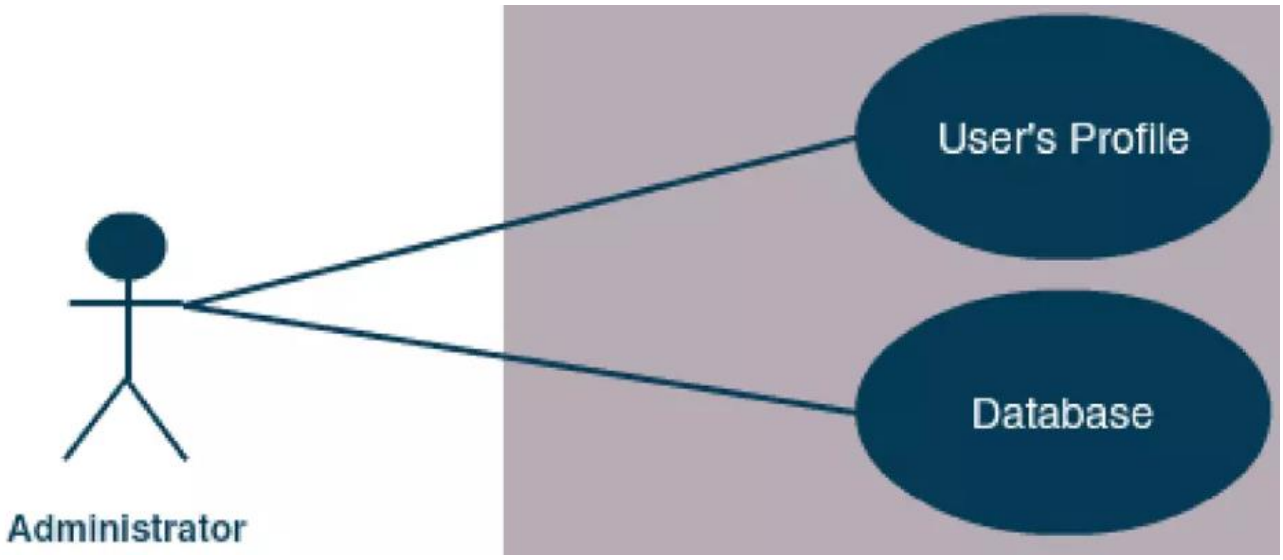

**User Case Diagram of Contacts Form**

**Chat Forms**



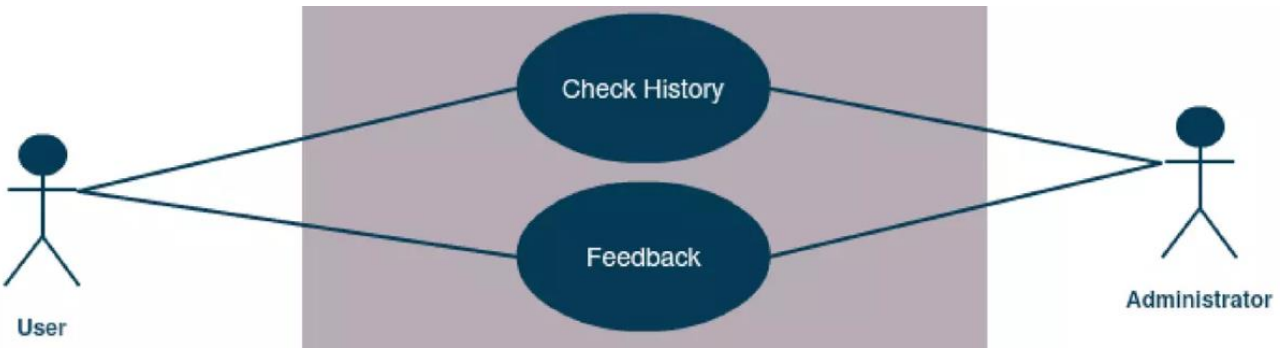

**User Case Diagram of Chat Forms**

**Maintenance**



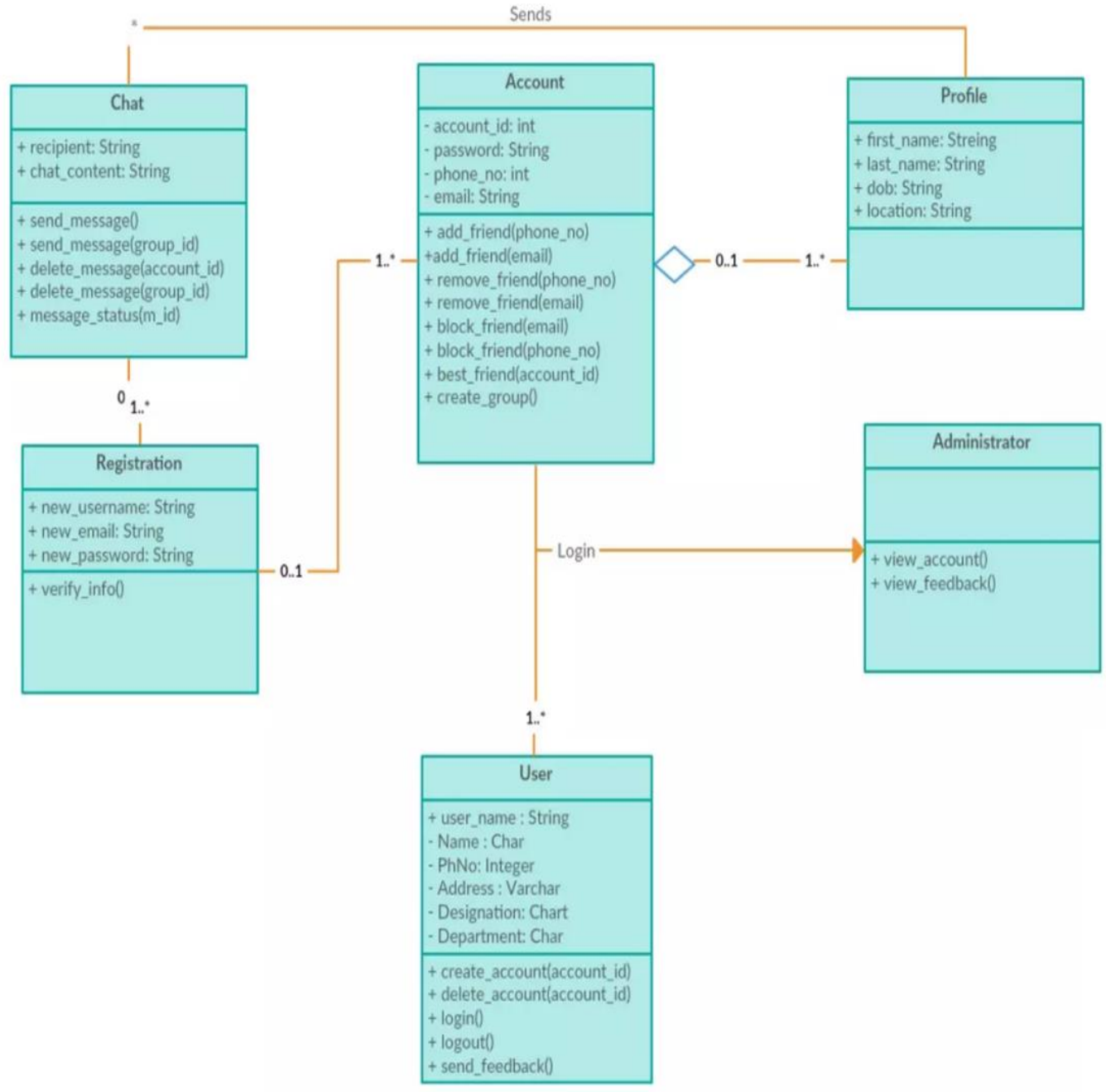

**User Case Diagram of Maintenance**

**Monitor**



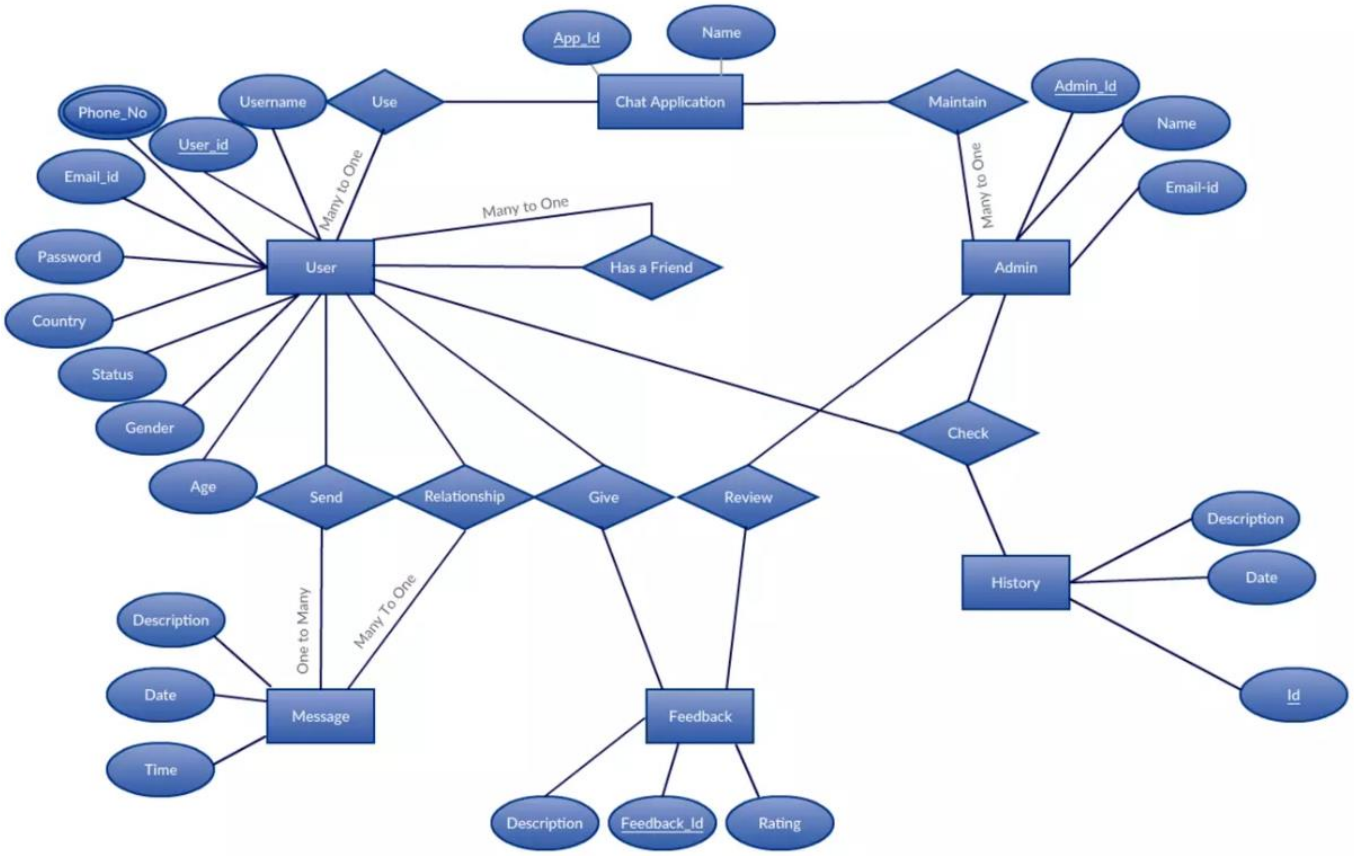

**User Case Diagram of Monitor**

**Class Diagram of Chat Application**



## Class Diagram

## Entity Relationship Diagram (ERD)




**Entity Relationship Diagram of Chat Application**

## Sequence Diagrams

Ένα διάγραμμα σχέσεων οντοτήτων (ERD), γνωστό και ως μοντέλο σχέσεων οντοτήτων, είναι μια γραφική αναπαράσταση που απεικονίζει σχέσεις μεταξύ ανθρώπων, αντικειμένων, τόπων, εννοιών ή συμβάντων μέσα σε ένα σύστημα τεχνολογίας πληροφοριών (IT). Ένα ERD χρησιμοποιεί τεχνικές μοντελοποίησης δεδομένων που μπορούν να βοηθήσουν στον καθορισμό των επιχειρηματικών διαδικασιών και να χρησιμεύσουν ως βάση για μια σχεσιακή βάση δεδομένων. Τα διαγράμματα σχέσεων οντοτήτων παρέχουν ένα οπτικό σημείο εκκίνησης για το σχεδιασμό της βάσης δεδομένων που μπορεί επίσης να χρησιμοποιηθεί για να βοηθήσει στον προσδιορισμό των απαιτήσεων του συστήματος πληροφοριών σε έναν οργανισμό. Μετά την κυκλοφορία μιας σχεσιακής βάσης δεδομένων, ένα ERD μπορεί να χρησιμεύσει ως σημείο αναφοράς, εάν χρειαστεί αργότερα τυχόν εντοπισμός σφαλμάτων ή επανασχεδιασμός της επιχειρηματικής διαδικασίας.

Ωστόσο, ενώ ένα ERD μπορεί να είναι χρήσιμο για την οργάνωση δεδομένων που μπορούν να αναπαρασταθούν από μια σχεσιακή δομή, δεν μπορεί να αντιπροσωπεύει επαρκώς ημι-

δομημένα ή μη δομημένα δεδομένα. Είναι επίσης απίθανο να είναι χρήσιμο από μόνο του για την ενσωμάτωση δεδομένων σε ένα προϋπάρχον σύστημα πληροφοριών. Τα ERD απεικονίζονται γενικά σε ένα ή περισσότερα από τα ακόλουθα μοντέλα:

Ένα εννοιολογικό μοντέλο δεδομένων, το οποίο στερείται συγκεκριμένων λεπτομερειών, αλλά παρέχει μια επισκόπηση του πεδίου εφαρμογής του έργου και του τρόπου με τον οποίο τα σύνολα δεδομένων σχετίζονται μεταξύ τους.

Ένα λογικό μοντέλο δεδομένων, το οποίο είναι πιο λεπτομερές από ένα εννοιολογικό μοντέλο δεδομένων, που απεικονίζει συγκεκριμένα χαρακτηριστικά και σχέσεις μεταξύ σημείων δεδομένων. Ενώ ένα εννοιολογικό μοντέλο δεδομένων δεν χρειάζεται να σχεδιαστεί πριν από ένα μοντέλο λογικών δεδομένων, ένα μοντέλο φυσικών δεδομένων βασίζεται σε ένα μοντέλο λογικών δεδομένων.

Ένα μοντέλο φυσικών δεδομένων, το οποίο παρέχει το προσχέδιο για μια φυσική εκδήλωση -- όπως μια σχεσιακή βάση δεδομένων -- του λογικού μοντέλου δεδομένων. Ένα ή περισσότερα μοντέλα φυσικών δεδομένων μπορούν να αναπτυχθούν με βάση ένα λογικό μοντέλο δεδομένων.

Υπάρχουν πέντε βασικά στοιχεία ενός διαγράμματος σχέσεων οντοτήτων. Παρόμοια εξαρτήματα θα χαρακτηρίζονται με το ίδιο σχήμα. Για παράδειγμα, όλοι οι τύποι οντοτήτων μπορεί να περικλείονται σε ένα ορθογώνιο, ενώ όλα τα χαρακτηριστικά περικλείονται σε ένα διαμάντι. Τα εξαρτήματα περιλαμβάνουν:

Οντότητες, οι οποίες είναι αντικείμενα ή έννοιες που μπορούν να έχουν αποθηκευμένα δεδομένα σχετικά με αυτές. Οι οντότητες αναφέρονται σε πίνακες που χρησιμοποιούνται σε βάσεις δεδομένων.

Χαρακτηριστικά, τα οποία είναι ιδιότητες ή χαρακτηριστικά οντοτήτων. Ένα χαρακτηριστικό ERD μπορεί να χαρακτηριστεί ως πρωτεύον κλειδί, το οποίο προσδιορίζει ένα μοναδικό χαρακτηριστικό, ή ένα ξένο κλειδί, το οποίο μπορεί να εκχωρηθεί σε πολλαπλά χαρακτηριστικά.

Οι σχέσεις μεταξύ και μεταξύ αυτών των οντοτήτων.

Ενέργειες, οι οποίες περιγράφουν πώς οι οντότητες μοιράζονται πληροφορίες στη βάση δεδομένων.

Γραμμές σύνδεσης

Το ERD εξηγεί οπτικά τις σχέσεις μεταξύ αντιπροσώπων πωλήσεων, πελατών και παραγγελιών προϊόντων.

Ένα διάγραμμα σχέσης οντότητας που δείχνει σχέσεις μεταξύ αντιπροσώπων πωλήσεων, πελατών και παραγγελιών προϊόντων.

Για παράδειγμα, ένα ERD που αντιπροσωπεύει το σύστημα πληροφοριών για το τμήμα πωλήσεων μιας εταιρείας μπορεί να ξεκινά με γραφικές αναπαραστάσεις οντοτήτων όπως ο αντιπρόσωπος πωλήσεων, ο πελάτης, η διεύθυνση του πελάτη, η παραγγελία του πελάτη, το προϊόν και η αποθήκη. (Βλ. παραπάνω διάγραμμα.) Στη συνέχεια, γραμμές ή άλλα σύμβολα μπορούν να χρησιμοποιηθούν για να αναπαραστήσουν τη σχέση μεταξύ οντοτήτων και το κείμενο μπορεί να χρησιμοποιηθεί για την επισήμανση των σχέσεων.

Ένας συμβολισμός καρδιαλικότητας μπορεί στη συνέχεια να ορίσει τα χαρακτηριστικά της σχέσης μεταξύ των οντοτήτων. Οι καρδινότητες μπορούν να υποδηλώνουν ότι μια οντότητα είναι προαιρετική (για παράδειγμα, ένας εκπρόσωπος πωλήσεων θα μπορούσε να μην έχει πελάτες ή θα μπορούσε να έχει πολλούς) ή υποχρεωτικό (για παράδειγμα, πρέπει να υπάρχει τουλάχιστον ένα προϊόν που αναφέρεται σε μια παραγγελία.)

Τα τρία βασικά χαρακτηριστικά είναι:

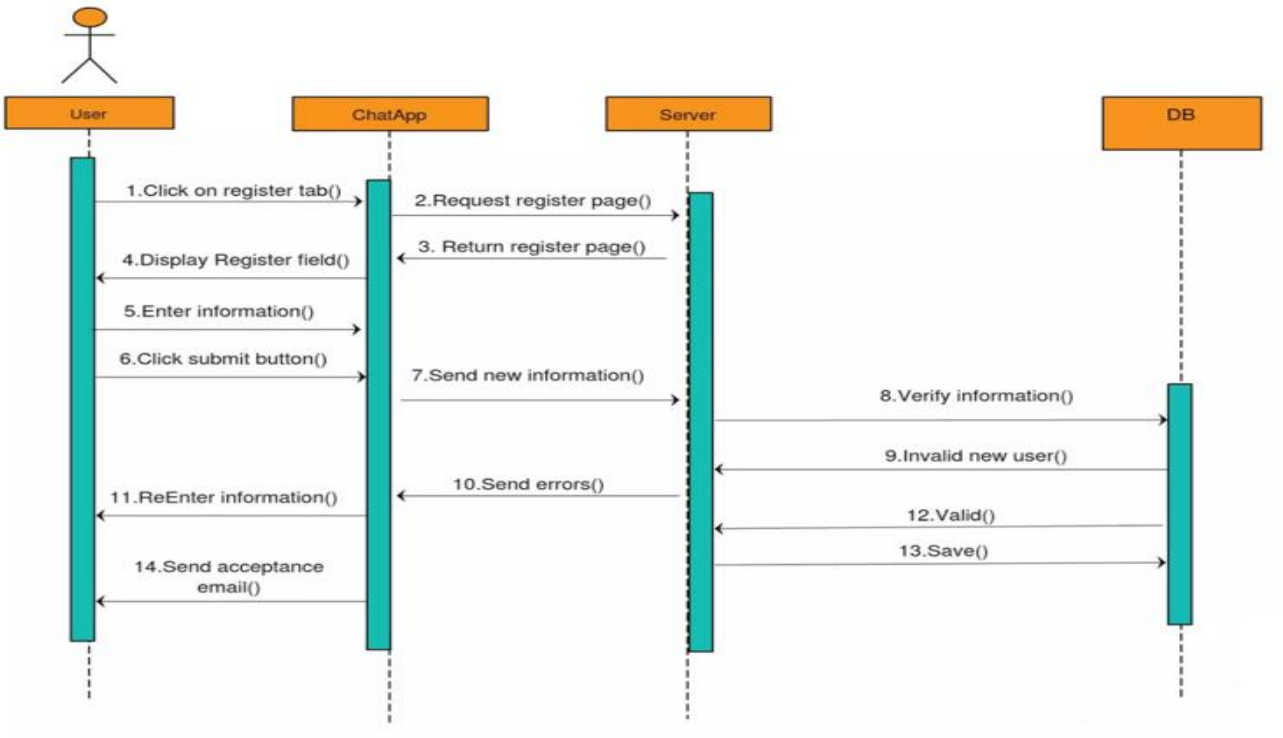
Μια σχέση ένας προς έναν (1:1). Για παράδειγμα, εάν κάθε πελάτης σε μια βάση δεδομένων συσχετίζεται με μία ταχυδρομική διεύθυνση.

Μια σχέση ένα προς πολλά (1:M). Για παράδειγμα, ένας πελάτης μπορεί να κάνει μια παραγγελία για πολλά προϊόντα. Ο πελάτης συσχετίζεται με πολλές οντότητες, αλλά όλες αυτές οι οντότητες έχουν μια ενιαία σύνδεση πίσω στον ίδιο πελάτη.

Μια σχέση πολλά-προς-πολλά (M:N). Για παράδειγμα, σε μια εταιρεία όπου όλοι οι πράκτορες τηλεφωνικών κέντρων συνεργάζονται με πολλούς πελάτες, κάθε πράκτορας συσχετίζεται με πολλούς πελάτες και πολλοί πελάτες μπορεί επίσης να συσχετίζονται με πολλούς πράκτορες.

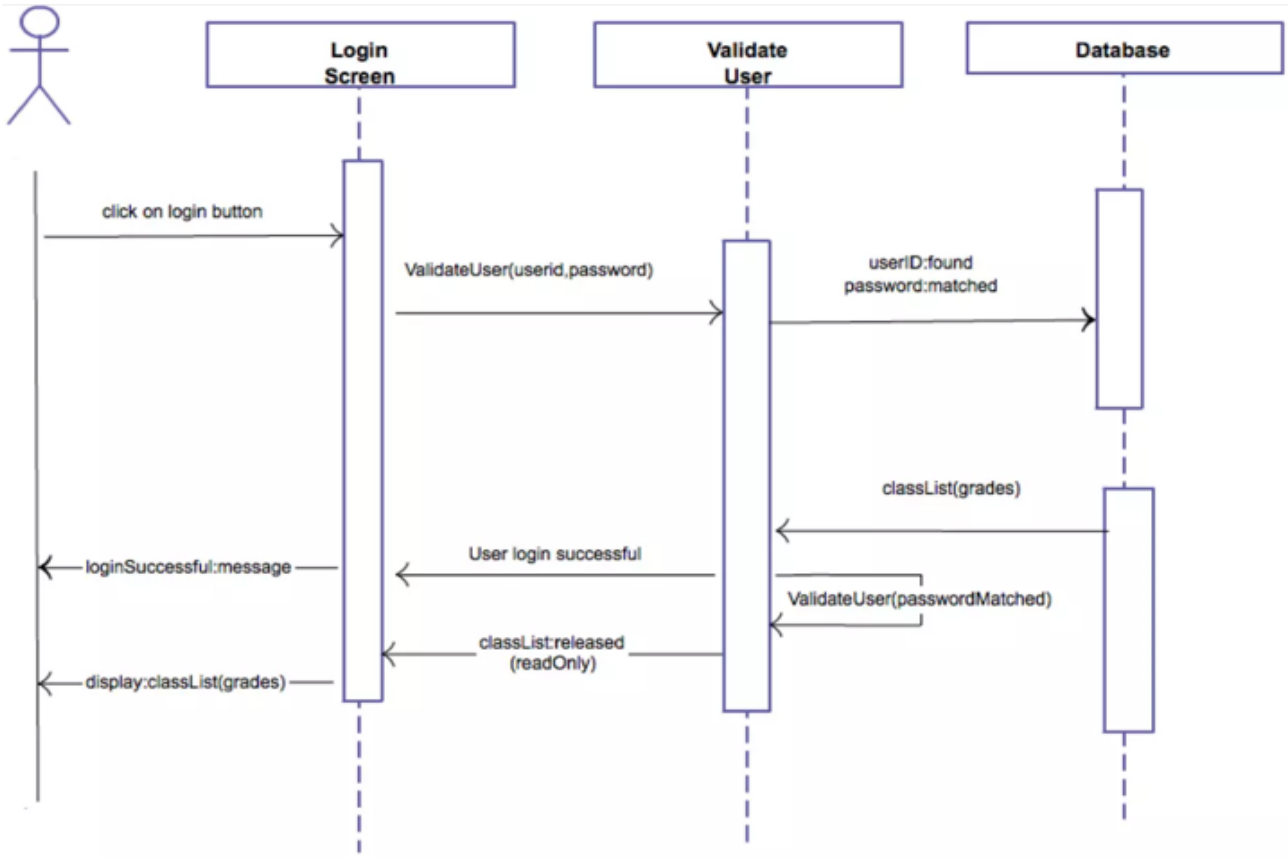
Ενώ υπάρχουν εργαλεία που βοηθούν στην κατάρτιση διαγραμμάτων σχέσεων οντοτήτων, όπως εργαλεία μηχανικής λογισμικού με τη βοήθεια υπολογιστή (CASE), ορισμένα συστήματα διαχείρισης σχεσιακών βάσεων δεδομένων (RDBMS) έχουν επίσης ενσωματωμένες δυνατότητες σχεδίασης.

## Sequence Diagram Registration



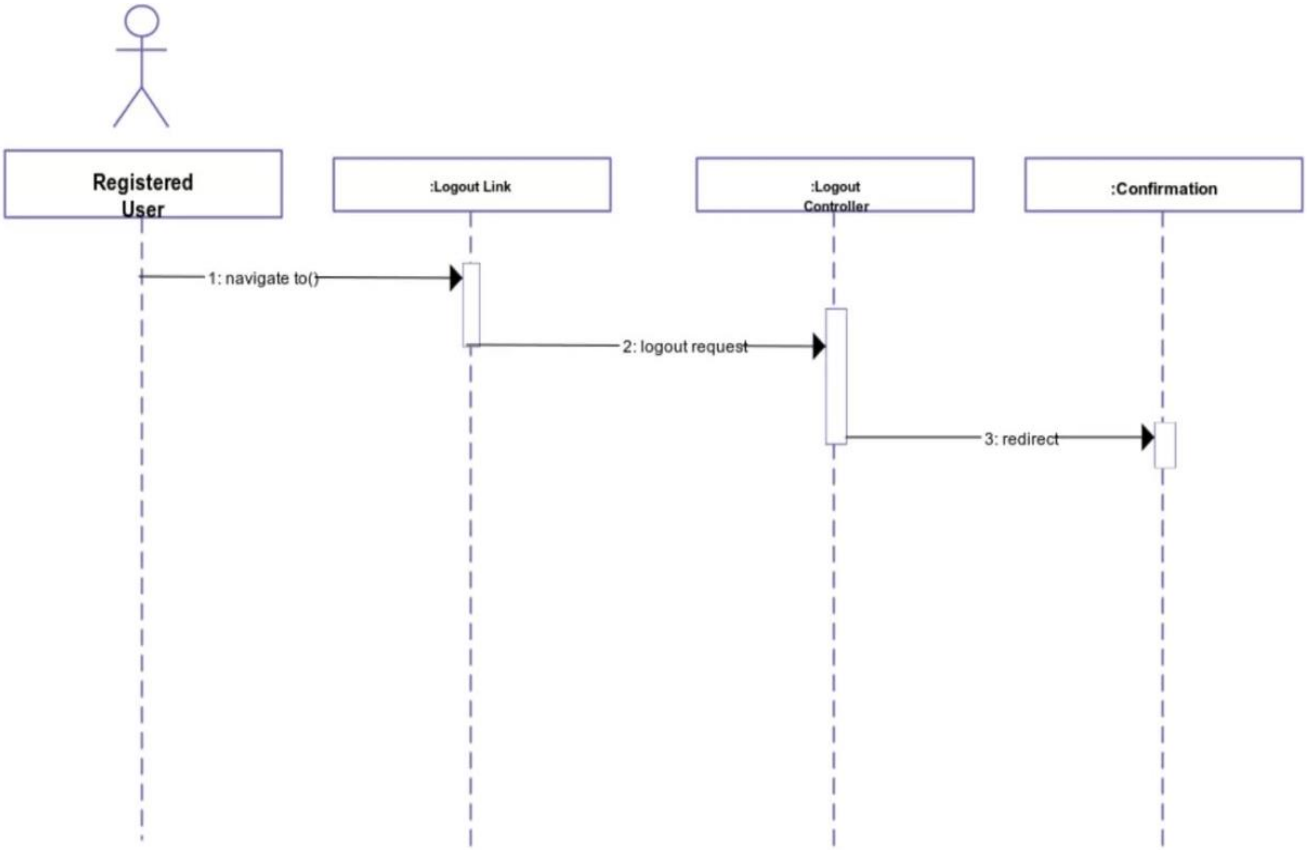

## Sequence diagram of Registration Functionality

## Sequence diagram login



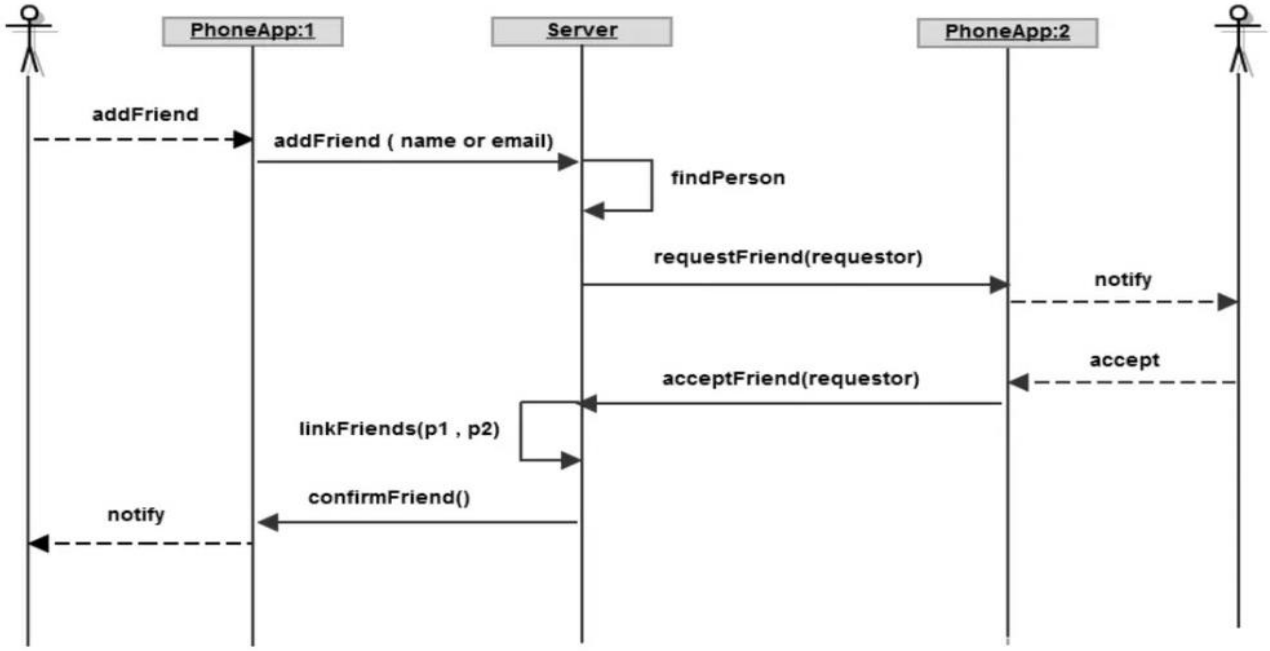
**Sequence diagram of Login Functionality**

# Sequence diagram logout



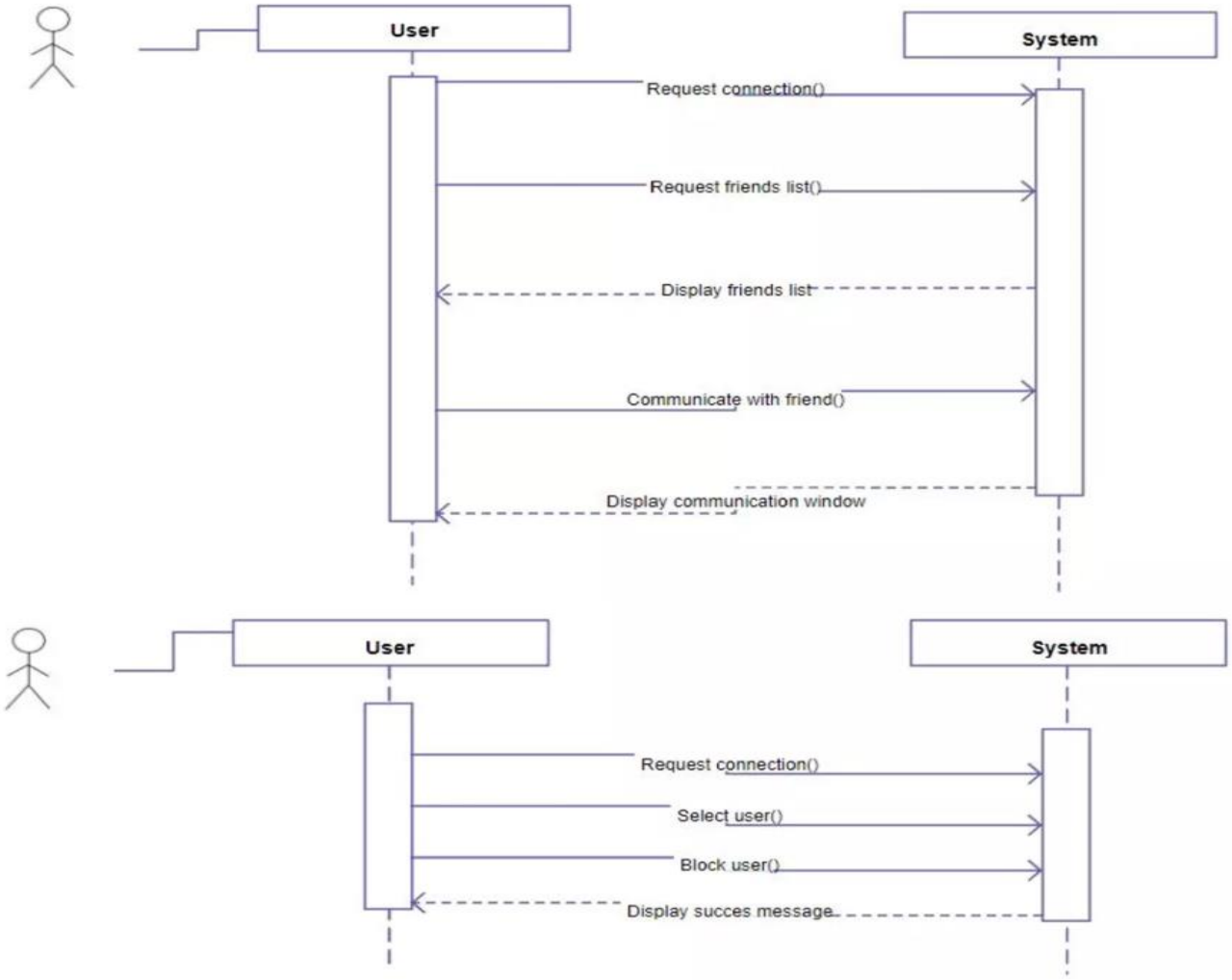

**Sequence diagram of Logout Functionality**

**Sequence diagram add friend**



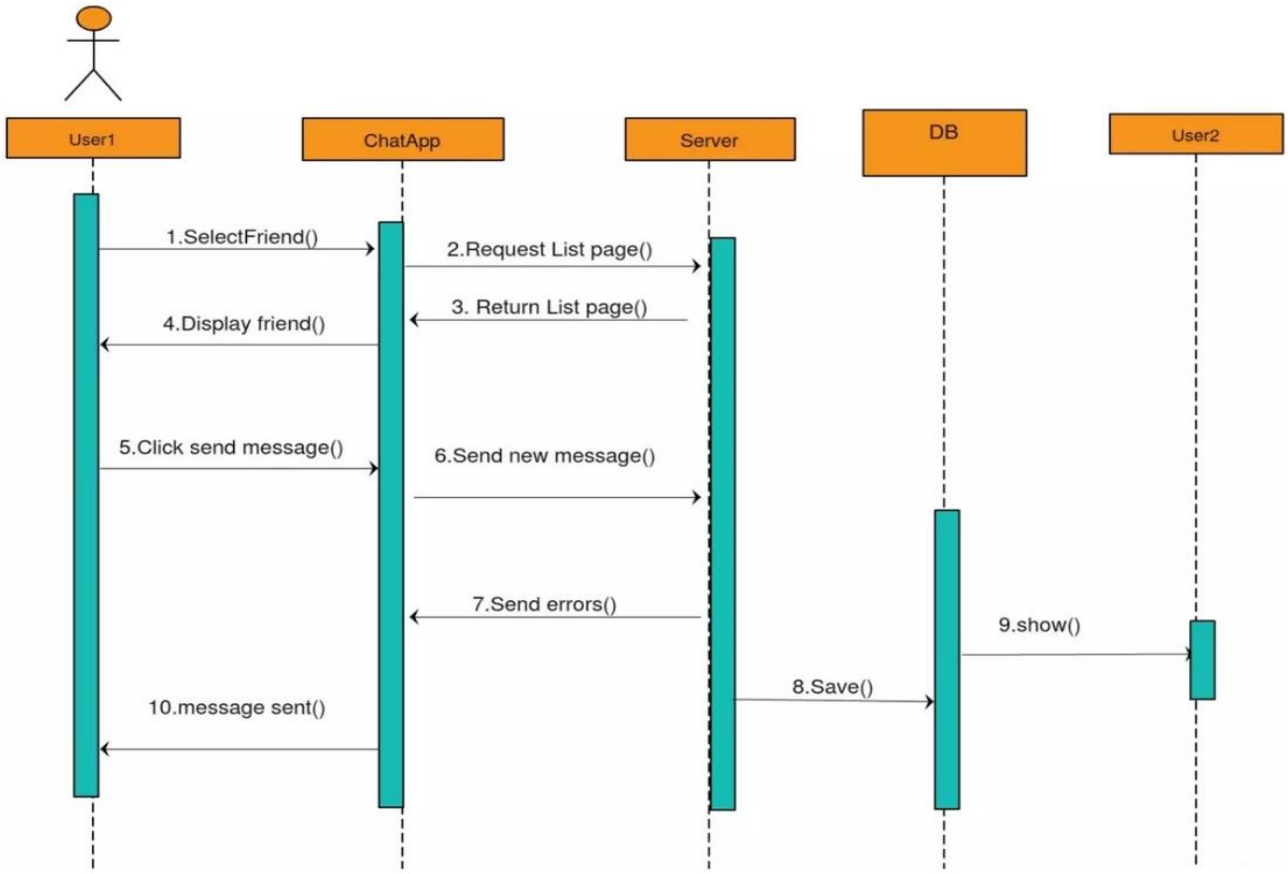

**Sequence diagram of add friend functionality**

# Sequence diagram of block friend functionality



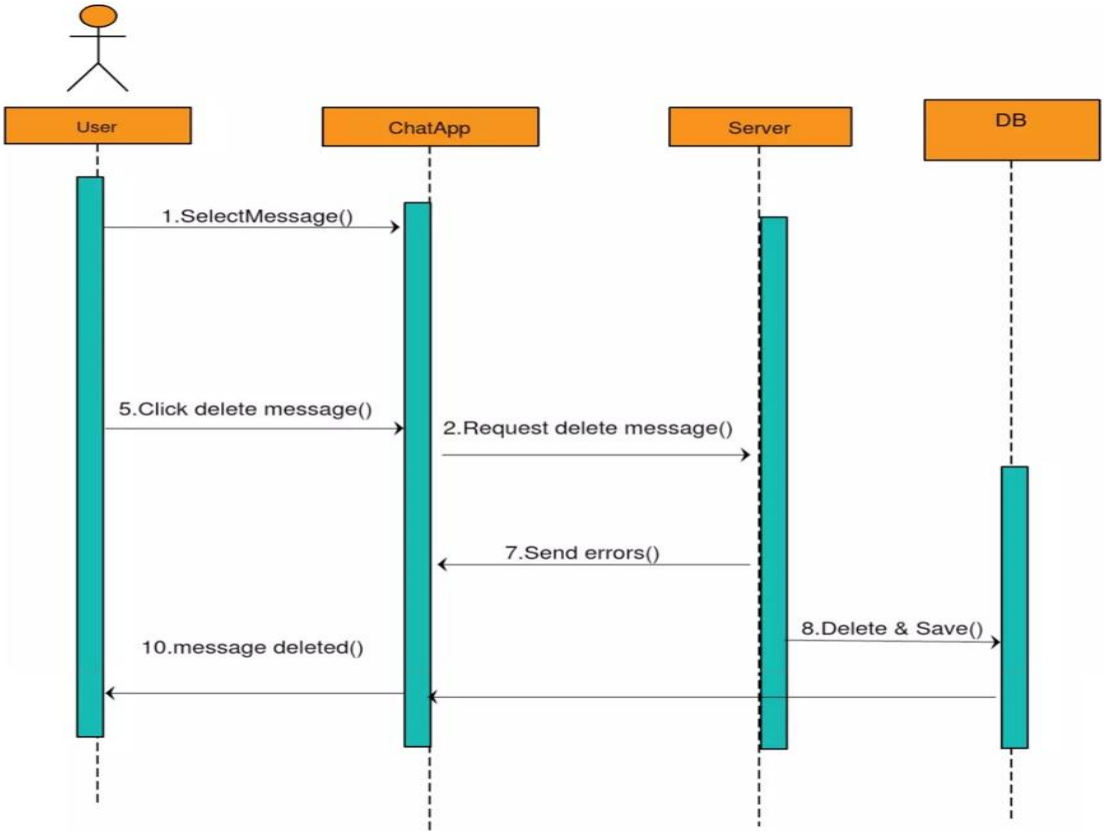

**Sequence diagram of block friend functionality**

**Sequence diagram of send message functionality**



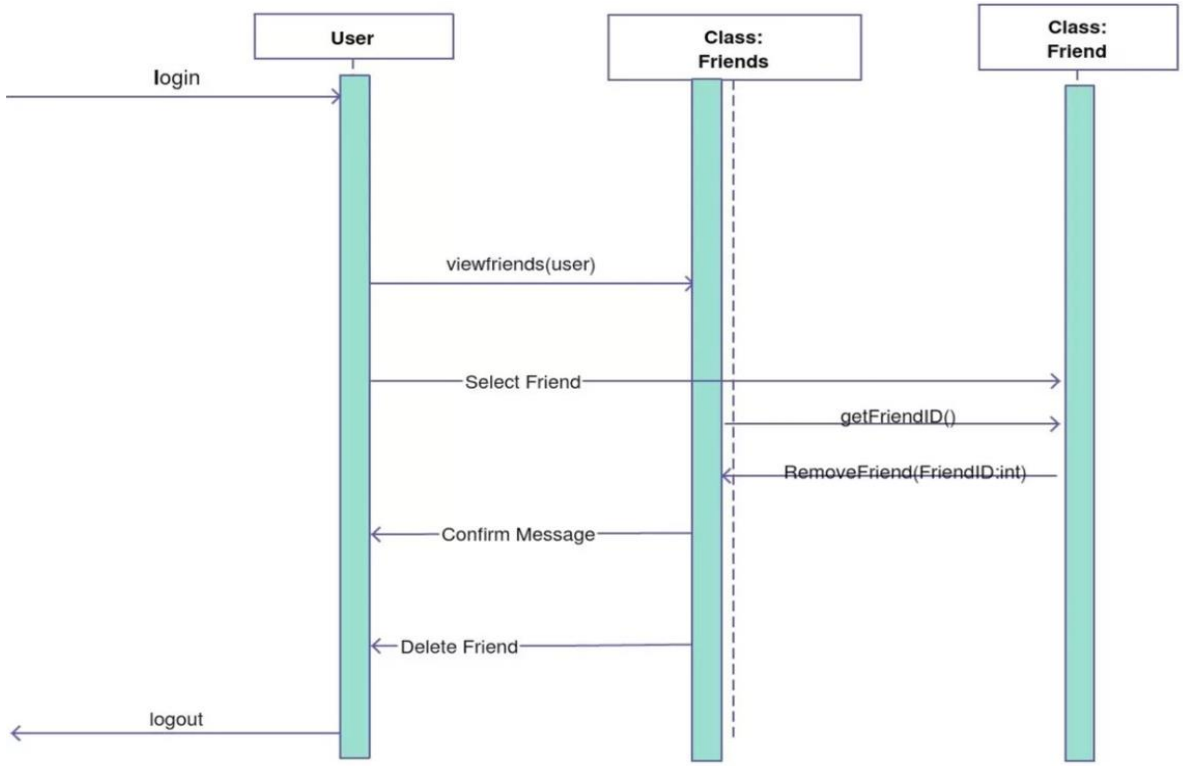

**Sequence diagram of send message functionality**

**Sequence diagram of delete message functionality**



**Sequence diagram of delete message functionality**

**Sequence diagram of remove friend functionality**




**Sequence diagram of remove friend functionality**

# Κεφάλαιο 4 – Εργαλεία Ανάπτυξης Εφαρμογής – Οδηγίες Χρήσης

Σε αυτό το κεφάλαιο θα δούμε τα βήματα που ακολουθήσαμε ώστε να φτιάξουμε την εφαρμογή. Θα ξεκινήσουμε από μία γενική παρουσίαση της εφαρμογής, θα δούμε τα εργαλεία που χρησιμοποιήσαμε και κάποια χαρακτηριστικά κομμάτια από τον κώδικα της εφαρμογής.

## 4.1 Απαραίτητα Εργαλεία

### ANDROID SDK

Τα βασικά εργαλεία ανάπτυξης της εφαρμογής ονομάζονται Android SDK (Standard Development Kit) και δίνονται όλα συγκεντρωμένα από την εταιρεία Google. Αυτό περιλαμβάνει τις βιβλιοθήκες Java του Android που θα χρησιμοποιήσει η εφαρμογή, τον emulator για την δοκιμαστική εκτέλεση της εφαρμογής χωρίς να χρειάζεται να υπάρχει συσκευή (που παρεμπιπτόντως δεν τρέχει εφαρμογές με το Google Maps), τον compiler που παράγει τον κώδικα και μία πληθώρα διαφορετικών βιβλιοθηκών. Το SDK δεν παρέχεται μόνο του αλλά έρχεται μαζί με ένα περιβάλλον ανάπτυξης. Αυτή τη στιγμή υπάρχουν δύο επιλογές, το eclipse και το Android Studio. Το Android Studio είναι ένα περιβάλλον βασισμένο στο IntelliJ IDEA και είναι το περιβάλλον που αναπτύσσεται ενεργά από τη Google. Υποστηρίζει όλες τις δυνατότητες που παρέχει και το eclipse εκτός από το NDK (14).


*Figure 13: Android SDK*

**FIREBASE**

*Figure 14: Firebase*

Το Firebase είναι ένα λογισμικό που δημιουργήθηκε από την Google για την ανάπτυξη εφαρμογών για κινητά και ιστούς. Η εταιρεία, η οποία ιδρύθηκε το 2011, ήταν αυτόνομη το πρώτο διάστημα. Ωστόσο, τρία χρόνια μετά, η Google αγόρασε το λογισμικό αυτό και πλέον είναι η πρώτη της επιλογή για την ανάπτυξη λογισμικών. Ακολούθως, προχωρά στην αποθηκευσή των δεδομένων στο cloud του Firebase. Αυτό αποτελεί πολύ σημαντικό κομμάτι στη διαχείρηση από τη μεριά των προγραμματιστών καθώς για τους προγραμματιστές λογισμικού η υλοποίηση σε πραγματικό χρόνο και η κατασκευή συνεργατικών εφαρμογών που επικοινωνούν με τη πλατφόρμα και αποστέλλουν αντίστοιχα μυνήματα αποτελεί σημείο καμπής για την εμπορικότητα των εφαρμογών και την άμεση επικοινωνία με τον τελικό χρήστη (16).

Τον Μάιο του 2012, και διάστημα 30 ημερών περίπου μετά την εμφάνιση της δοκιμαστικής έκδοσης, η εταιρία συγκέντρωσε περίπου 1 εκατομμύριο δολάρια το οποίο το συγκέντρωσε από σημαντικά fund από επιχειρηματίες κεφαλαίων, όπως είναι το Flybridge Capital Partners. Με αυτή την ενέργεια η εταιρεία εδραιώθηκε και πλέον λειτουργεί ώς κινητή backend υπηρεσία για τους ενδιαφερόμενους πελάτες. Τον Οκτώβριο του 2014, η εξαγορά του Firebase ολοκληρώθηκε από την εταιρεία της Google και πλέον αποτελεί αναπόσπαστο κομμάτι της. Ένα χρόνο αργότερα, τον Οκτώβριο του 2015, η εταιρία έκανε δικό της το Divshot, μια πλατφόρμα που φιλοξενεί HTML5. Λιγο καιρό αργότερα η εταιρεία συγχώνευσε την πλατφόρμα αυτή με το Firebase (17).

Τον Μάιο του 2016, στο Google I / O, το ετήσιο συνέδριο προγραμματιστών της εταιρείας, το Firebase παρουσίασε το Firebase Analytics και ανακοίνωσε ότι επεκτείνει τις υπηρεσίες του

για να γίνει μια ενοποιημένη πλατφόρμα backend-as-a-service (BaaS) για προγραμματιστές κινητών. Το Firebase ενοποιείται πλέον με διάφορες άλλες υπηρεσίες Google, όπως το Google Cloud Platform, το AdMob και το Google Ads, για να προσφέρουν ευρύτερα προϊόντα και κλίμακα για προγραμματιστές. Το Google Cloud Messaging, η υπηρεσία Google για αποστολή ειδοποιήσεων push σε συσκευές Android, αντικαταστάθηκε από ένα προϊόν Firebase, το Firebase Cloud Messaging, το οποίο πρόσθεσε τη λειτουργικότητα για την παροχή ειδοποιήσεων push σε συσκευές iOS και ιστού. Τον Ιανουάριο του 2017, η Google απέκτησε το Fabric και το Crashlytics από το Twitter για να προσθέσει αυτές τις υπηρεσίες στο Firebase.

Επιπλέον, το Google Cloud Messaging, που αποτελεί σημαντική υπηρεσία της Google για αποστολή ειδοποιήσεων (push notifications) σε συσκευές Android, αντικαταστάθηκε από ένα προϊόν Firebase με αναλυτικά δεδομένα αποστολής ειδοποιήσεων προς τις κινητές συσκευές που αποστέλλονται τα αντίστοιχα push notifications για ενημέρωση των εγγεγραμμένων χρηστών σε συγκεκριμένες εφαρμογές. Ένα κατατοπιστικό παράδειγμα στη χρήση του Firebase θα ήταν η ενημέρωση των εγγεγραμμένων χρηστών σε ακραία φαινόμενα σε εφαρμογή για την αντιμετώπιση ακραίων καιρικών φαινομένων όπως εφαρμογή πολιτικής προστασίας για πρόβλεψη εμφάνισης πυρκαγιάς, ή πλημμύρας σε συγκεκριμένη περιοχή και η εμφάνιση push notifications από την προηγούμενη για την επόμενη ημέρα προς τους αρμόδιους φορείς και εγγεγραμμένους χρήστες της εφαρμογής. Ακόμη, το Firebase Cloud Messaging, το οποίο εισήγαγε τη δυνατότητα λειτουργίας για την δημιουργία ειδοποιήσεων push σε συσκευές Android, iOS και ιστού. Τον Ιανουάριο του 2017, η Google απέκτησε το Fabric και το Crashlytics από το Twitter και τις πρόσθεσε τις υπηρεσίες αυτές στο Firebase (18, 19).

## 4.2 Οδηγίες Χρήσης Εφαρμογής

Στην παρούσα ενότητα θα παραθέσουμε μια εκτενή περιγραφή της εφαρμογής μας, όπως και του τρόπου με τον οποίο ο χρήστης έχει τη δυνατότητα να την αξιοποιήσει. Αναλυτικά η παρούσα εφαρμογή διαθέτει μια σειρά ενεργειών και δραστηριοτήτων που αναλύονται παρακάτω.

### 4.2.1 Η δραστηριότητα LOGIN

Η λειτουργία εισόδου βρίσκεται στην αρχική οθόνη της εφαρμογής μας. Επιπλέον αξίζει να αναφέρουμε ότι δίνει την δυνατότητα σε έναν user να προβεί σε login με τα ανάλογα credentials εισόδου για να πραγματοποιήσει είσοδο στην εφαρμογή μας δημιουργώντας δικό του όνομα χρήστη και κωδικό εισόδου στην εφαρμογή. Η εφαρμογή στην συγκεκριμένη καρτέλα εισόδου επιτρέπει στον χρήστη να προβεί σε δραστηριότητα Login όπως παρατηρούμε στην παρακάτω

Εικόνα. Ακόμη σε περίπτωση που ο χρήστης έχει πραγματοποιήσει προγενέστερα εγγραφή τότε του δίνεται η δυνατότητα “Login” να συνδεθεί στην εφαρμογή. Σε αντίθετη περίπτωση, έχει την δυνατότητα να επιλέξει το κουμπί “Register” ώστε να δημιουργήσει έναν καινούριο λογαριασμό (20).

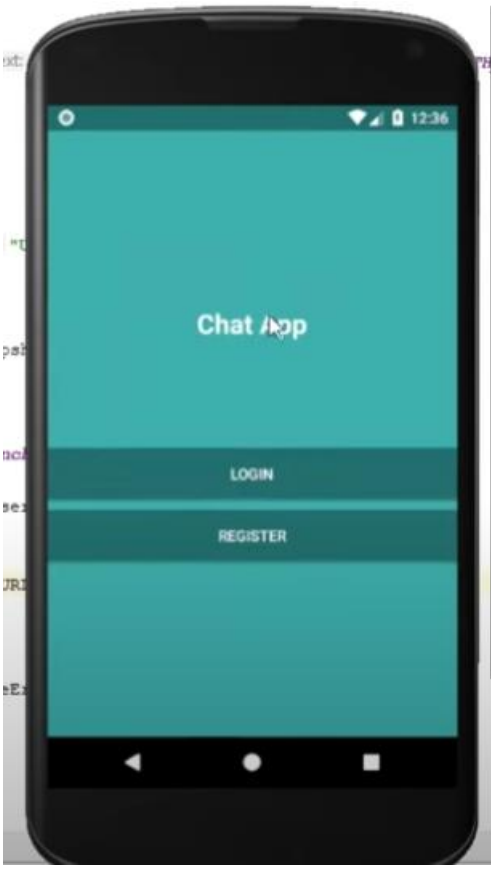



*Figure 15: Main - ChatApplication*

Εφόσον ο χρήστης πατήσει το κουμπί “Login”, οδηγείται στην οθόνη που φαίνεται στην παρακάτω εικόνα , έτσι χρειάζεται να τοποθετήσει το email του και τον κωδικό που έχει κάνει εγγραφεί για να συνδεθεί.

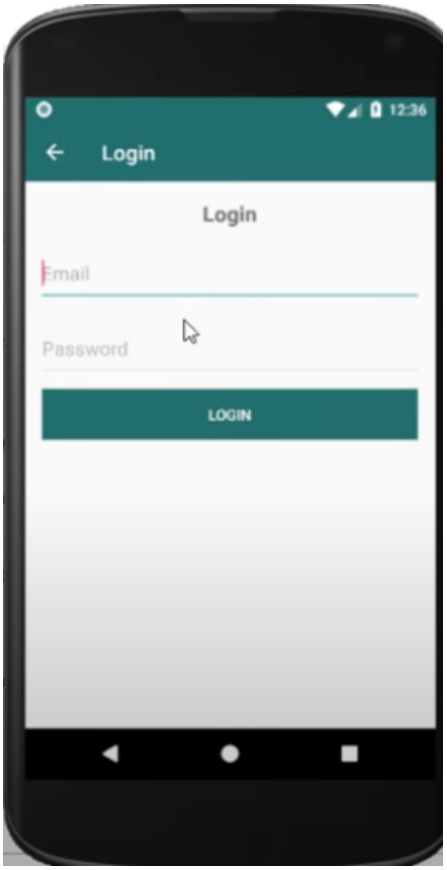



*Figure 16: Login Activity*

### 4.3 Η δραστηριότητα Register

Ο χρήστης έχει την δυνατότητα να εγγραφεί στην database της εφαρμογής και αυτό πραγματοποιείται μέσω της δραστηριότητας Register. Στην παρακάτω Εικόνα αναπαριστούμε την διεπαφή για την συγκεκριμένη δραστηριότητα για την υλοποίηση της διαδικασίας Register. Επιλέγοντας το πρώτο κουμπί, και εφόσον έχει ολοκληρωθεί ή ορθή συμπλήρωση των υποχρεωτικών πεδίων με τα παρακάτω στοιχεία αναλυτικά για τα παρακάτω πεδία, όνομα χρήστη, e-mail, και κωδικό, η εφαρμογή μετά από επικοινωνία με τον Server με την διαδικασία Parse για την εφαρμογή και εφόσον έχουν καταχωρηθεί ορθά τα προαναφερθέντα στοιχεία, στέλνει τα περιεχόμενα των τριών παραπάνω πεδίων. Στην συγκεκριμένη περίπτωση, ο κωδικός που καταχωρείται από τον χρήστη υφίσταται κρυπτογράφηση με τον αλγόριθμο SHA-512. Αν υπάρξει κάποια αναντιστοιχία η οποιοδήποτε άλλο πρόβλημα, όπως για παράδειγμα αν υφίσταται ήδη το e-mail, τότε ο χρήστης ενημερώνεται αυτόματα μέσω της εφαρμογής ότι η εγγραφή δεν ήταν επιτυχής.

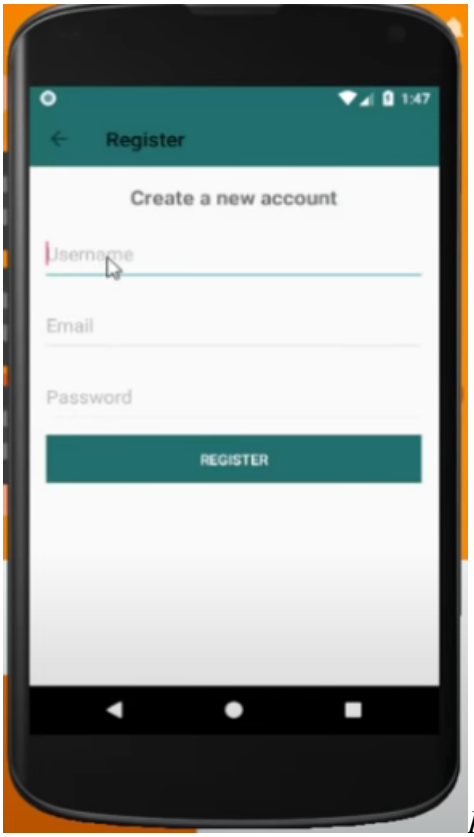


*Figure 17: Register Activity*

### 4.4 Κεντρική Οθόνη Εφαρμογής

Η κεντρική οθόνη εφαρμογής χωρίζεται σε 3 διαφορετικά Fragments. Αριστερά βρίσκεται το Chat Fragment , στο οποίο ο χρήστης μπορεί να δει τις συνομιλίες που έχει με τους φίλους του. Στη μέση βρίσκεται το Users Fragment , στο οποίο γίνεται εμφάνιση των Users της εφαρμογής. Και τέλος, δεξιά βρίσκεται το Profile Fragment , όπου ο χρήστης μπορεί να ενημερώσει το προφίλ του. Όπως να αλλάξει εικόνα προφίλ.

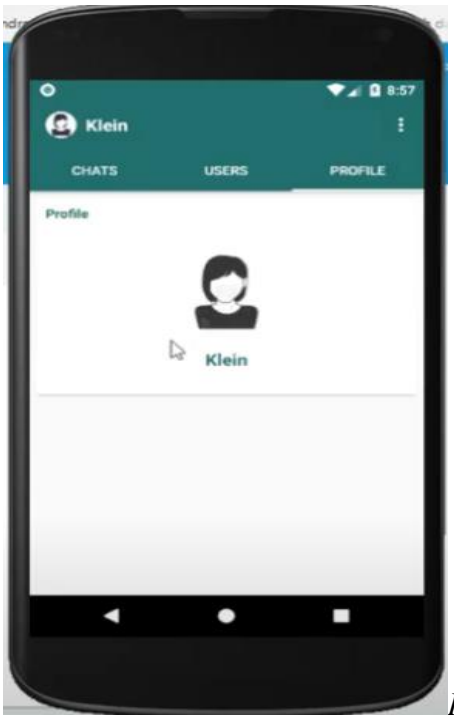


*Figure 18: Κεντρική Οθόνη*

### 4.5 Δραστηριότητα Chat

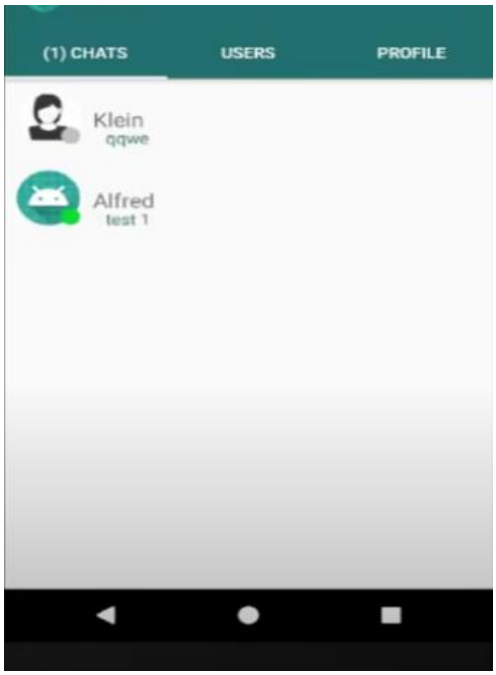


*Figure 19: Chat Activity*

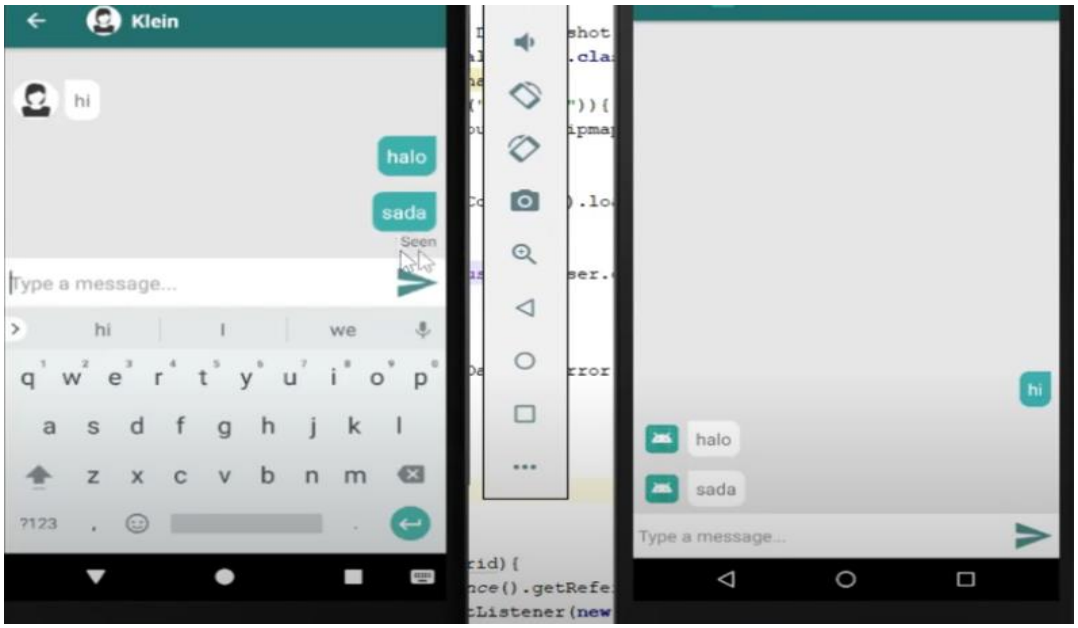



*Figure 20: Chat Activity 2*

### 4.6 Δραστηριότητα Users

Η δραστηριότητα Users παρουσιάζει μια λίστα με όλους τους χρήστες που είναι εγγεγραμμένοι στην εφαρμογή, από την οποία έχουμε την δυνατότητα να διαλέξουμε οποιονδήποτε χρήστη, με σκοπό να επικονωνήσουμε μαζί του. Εμφανίζει, επίσης, το ποιοι από τους χρήστες είναι ενεργοί ή όχι μέσα στην εφαρμογή σε πραγματικό χρόνο, με την ένδειξη ενός πράσινου χρώματος κύκλου ο οποίος γίνεται φανερός στην περίπτωση την οποία ο οποιοσδήποτε χρήστης είναι ενεργός (21).

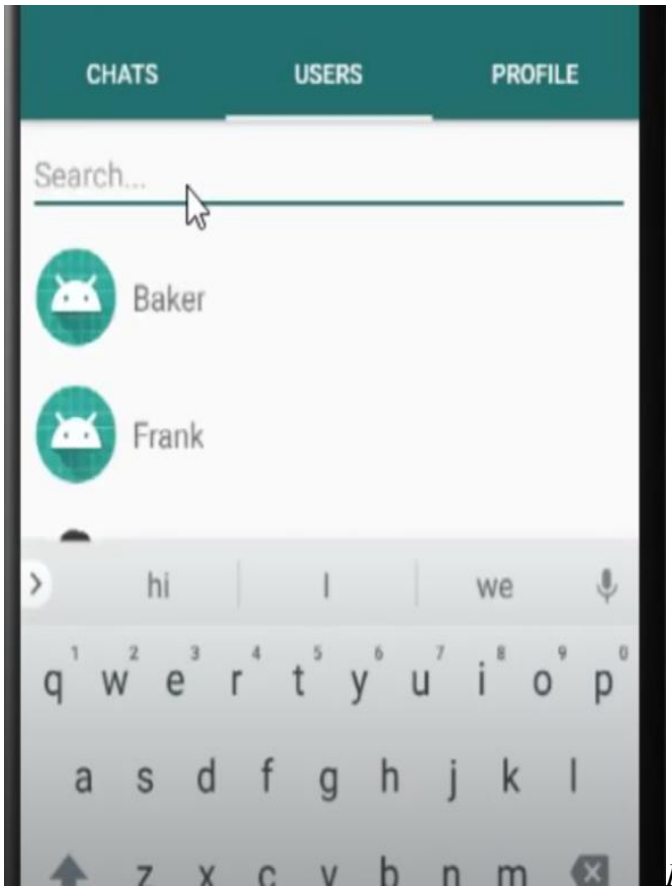


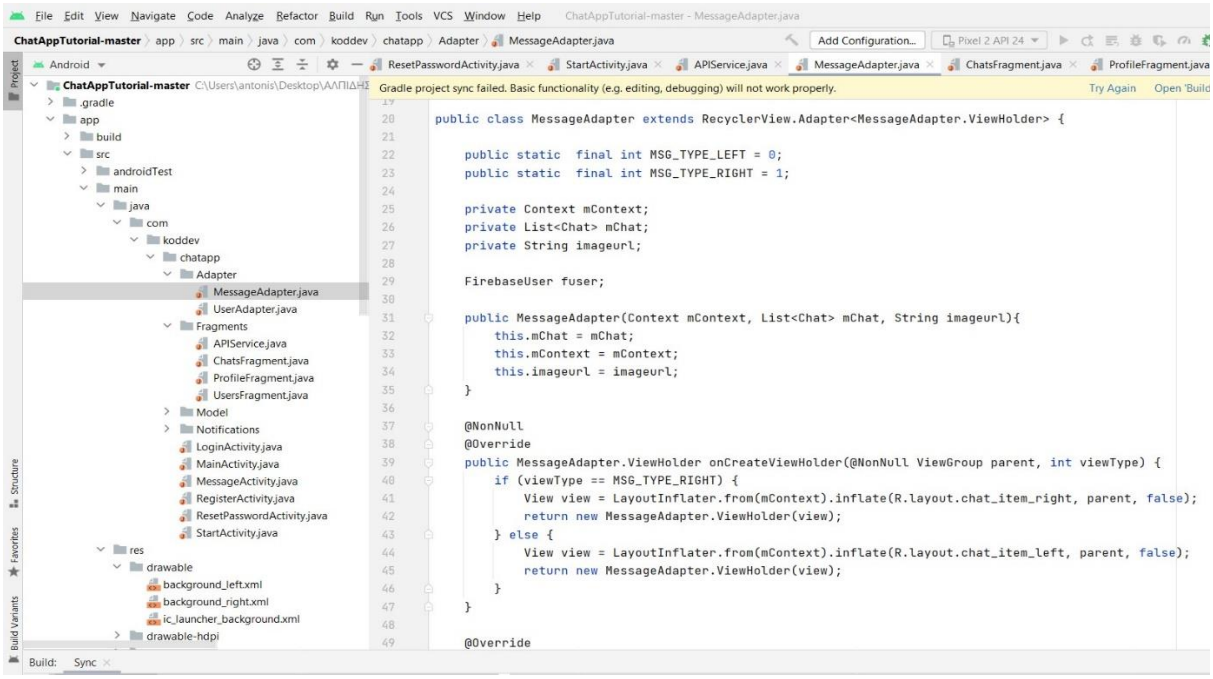
*Figure 21: User Activity*

### 4.7 Δραστηριότητα Profile

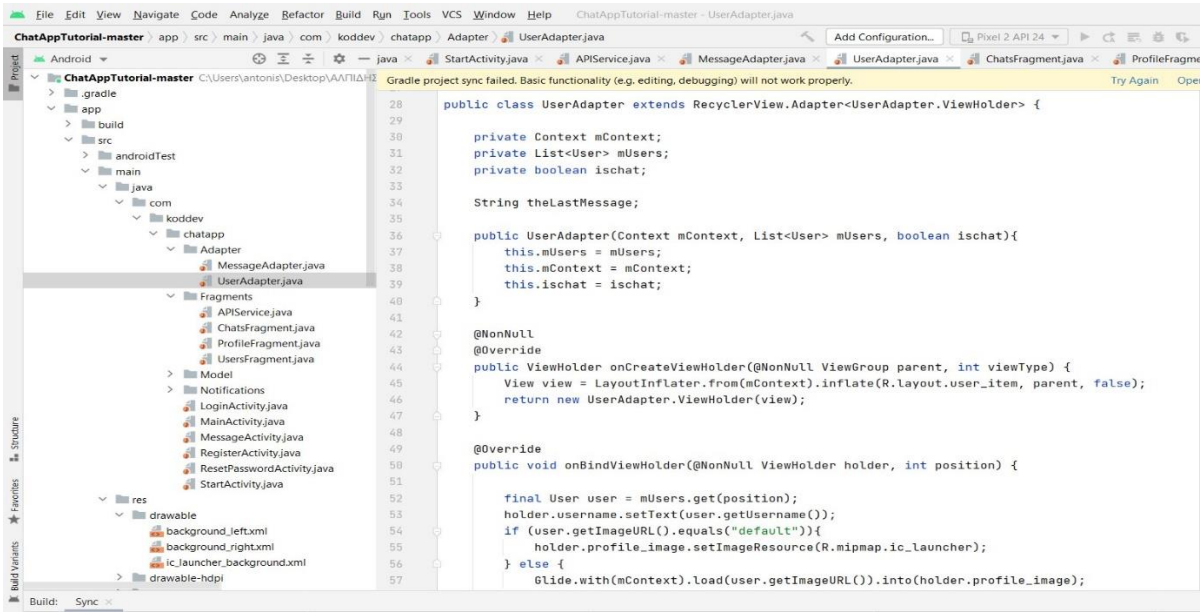
Στην δραστηριότητα Profile εμφανίζει το profile του χρήστη. Πιο συγκεκριμένα εμφανίζεται η εικόνα προφίλ του και το όνομα που χρησιμοποιεί. Ο χρήστης έχει την δυνατότητα με κλικ πάνω στην εικόνα να την αλλάξει. Η δραστηριότητα Profile φαίνεται στην παρακάτω εικόνα.

## 4.8 Manual for Chat application

### ADAPTERS




Message Adapter: Ένα Adapter το οποίο προσδιορίζει ένα Recycle View, που χρησιμοποιείται για να εκτυπώνει μηνύματα στο interface που βλέπουμε στο application. Είναι υπεύθυνο για τη διαχείριση όλων των μηνυμάτων του Recycle View, οι μόνιμες μεταβλητές MSG_TYPE_LEFT & MSG_TYPE_RIGHT προσδιορίζουν ποια μηνύματα βρίσκονται αριστερά και ποια δεξιά, παίρνει τρεις παραμέτρους Context object , Chat objects(μηνύματα) and an image URL(φωτογραφία του χρήστη), μέσα στη μέθοδο onCreateViewHolder υπάρχουν διαφορετικά layouts αναλόγως με τα μηνύματα και το LayoutInFlater προσδιορίζει το κατάλληλο layout αριστερά ή δεξιά μηνύματα αντίστοιχα. Η μέθοδος onBindViewHolder δεσμεύει τα δεδομένα (message text, image, εμφάνιση διαβασμένων μηνυμάτων), η ViewHolder μέθοδος αντιπροσωπεύει κάθε αντικείμενο ξεχωριστά μέσα στο Recycle View και τέλος η getItemViewType ελέγχει αν το μήνυμα που στάλθηκε το έστειλε ο χρήστης η όχι και επιστρέφει το κατάλληλο τύπο View.

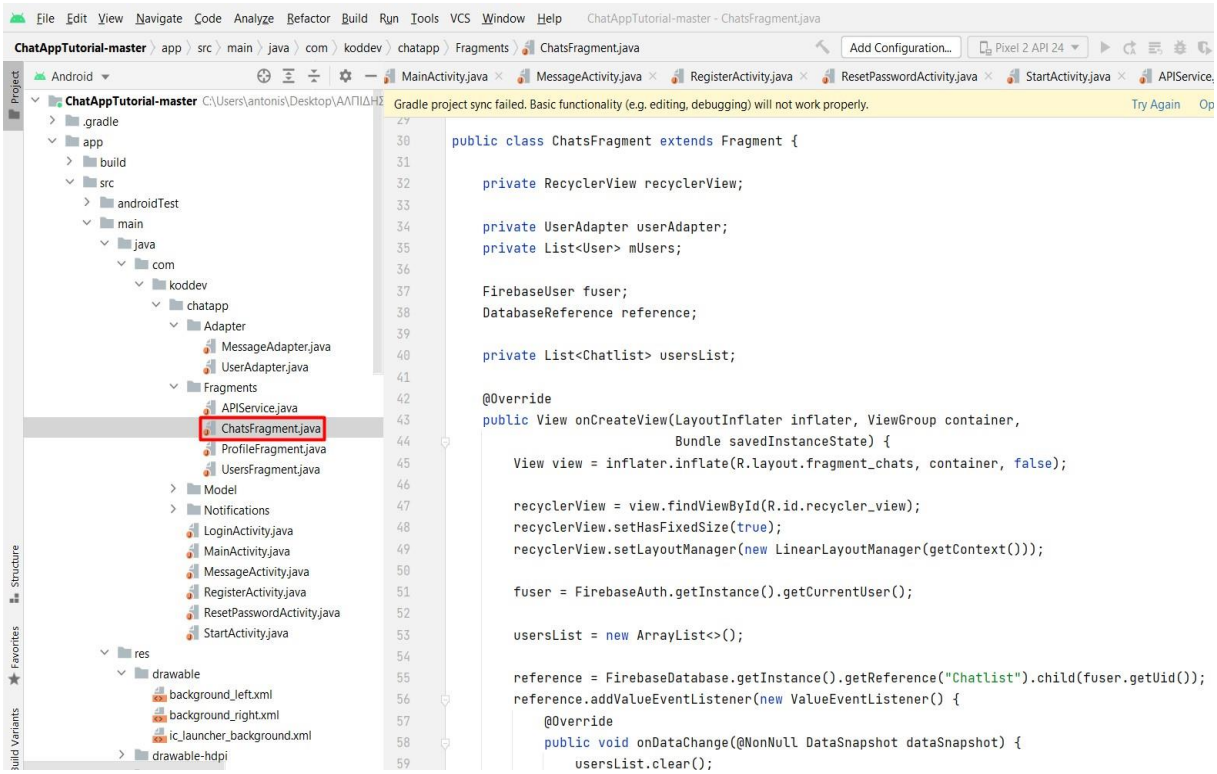



User Adapter: Ένα Adapter το οποίο προσδιορίζει ένα Recycle View, που χρησιμοποιείται για να εκτυπώνει τους χρήστες σε μορφή λίστας του application. Παίρνει τρεις παραμέτρους Context object, User objects(χρήστες) και μία Boolean μεταβλητή ischat(υποδεικνύει αν χρησιμοποιείται σε γεγονός ή όχι), μέσα στη μέθοδο onCreateViewHolder υπάρχουν διαφορετικά layouts αναλόγως με τα μηνύματα και το LayoutInFlater προσδιορίζει το κατάλληλο layout για το user.xml file.

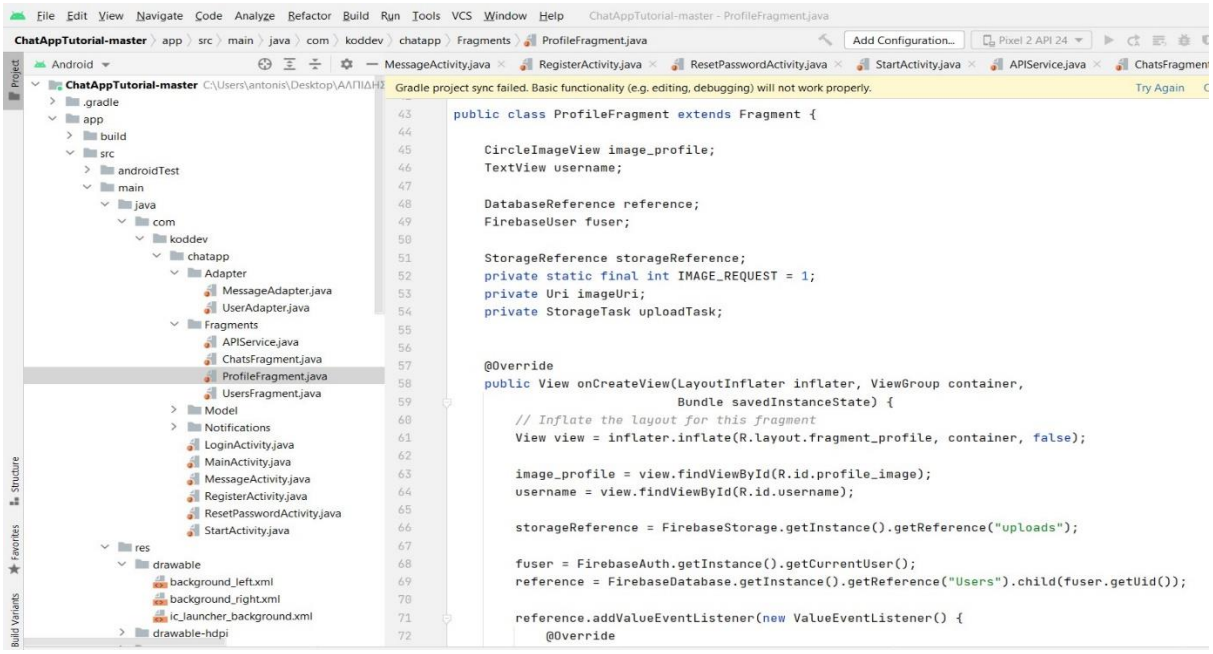
Η μέθοδος onBindViewHolder φτιάχνει τα δεδομένα ενός χρήστη(username, profile image) είναι υπεύθυνο να μας δείξει αν ένας χρήστης είναι online/offline και επιπλέον μας δείχνει το τελευταίο μήνυμα που αντάλλαξε ένας χρήστης με τον τωρινό χρήστη μας αν η μεταβλητή Boolean είναι true, η ViewHolder μέθοδος κρατάει όλες τις πληροφορίες του χρήστη μας μέσα στο Recycle View και τέλος η lastMessage μέθοδος ανακτά και εκτυπώνει τα τελευταία μηνύματα που ανταλλάχτηκαν σε ένα συγκεκριμένο χρήστη, παίρνει σαν παραμέτρους ένα user ID και ένα TextView, και επιστρέφει από τη βάση δεδομένων μας (Firebase) το τελευταίο μήνυμα ανάλογα με το ID(sender ή receiver).

## FRAGMENTS

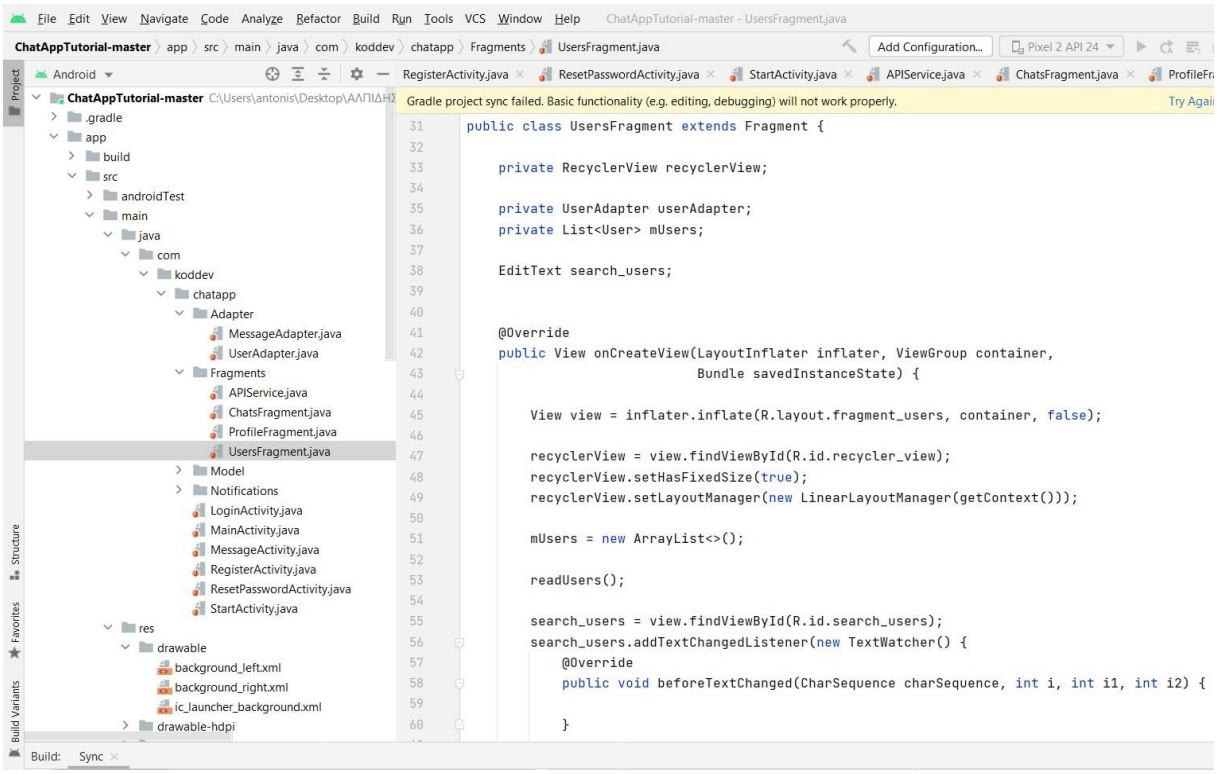
API Service: Χρησιμοποιείται για τα network requests (τροποποίηση δεδομένων API ή μέσα από έναν server) στέλνοντας ειδοποιήσεις σε ένα Adroid chat application. Γίνεται χρήση του Retrofit(κλάση που δημιουργεί αντικείμενα του API interface), που είναι επίσης μια δημοφιλής βιβλιοθήκη της JAVA. Γενικά το service καθορίζει τη δομή των network requests που στέλνουν ειδοποιήσεις μέσω του Firebase Cloud Messaging (FCM) service.




Chats Fragment: Είναι υπεύθυνο για την προβολή των συζητήσεων των χρηστών. Κληρονομεί τη κλάση Fragment και κάνει υποσκέλιση τη μέθοδο onCreateView. Η μέθοδος FirebaseAuth.getInstance().getCurrentUser() χρησιμοποιείται για να ανακαλέσει το Login του κάθε χρήστη μέσω του Firebase Authentication. Η λίστα usersList δηλώνεται για να κρατήσει όλα τη λίστα των chats. Η ValueEventListener δημιουργείται για να καλέσει τα δεδομένα τη chatlist, μέσω της μεταβλητής reference στην FireBase Realtime Database. Η μέθοδος chatlist () καλείται για να φέρει δεδομένα πραγματικών χρηστών από τη βάση δεδομένων μας, καθώς και δημιουργεί την λίστα mUsers που αναφέρεται σε αυτούς τους χρήστες. Η updateToken μέθοδος καλείται για να ανανεώσει το FCM token του κάθε χρήστη στη βάση δεδομένων με σκοπό να λαμβάνει ειδοποιήσεις. Το View object επιστρέφεται από το ChatFragment. Τέλος το ChatFragment είναι υπεύθυνο για τη προβολή της λίστας των chats ανάμεσα στους χρήστες , την ανάκτηση των δεδομένων των chats από τη βάση δεδομένων καθώς και την ανάκτηση του εκάστοτε χρήστη με τα δεδομένα του.

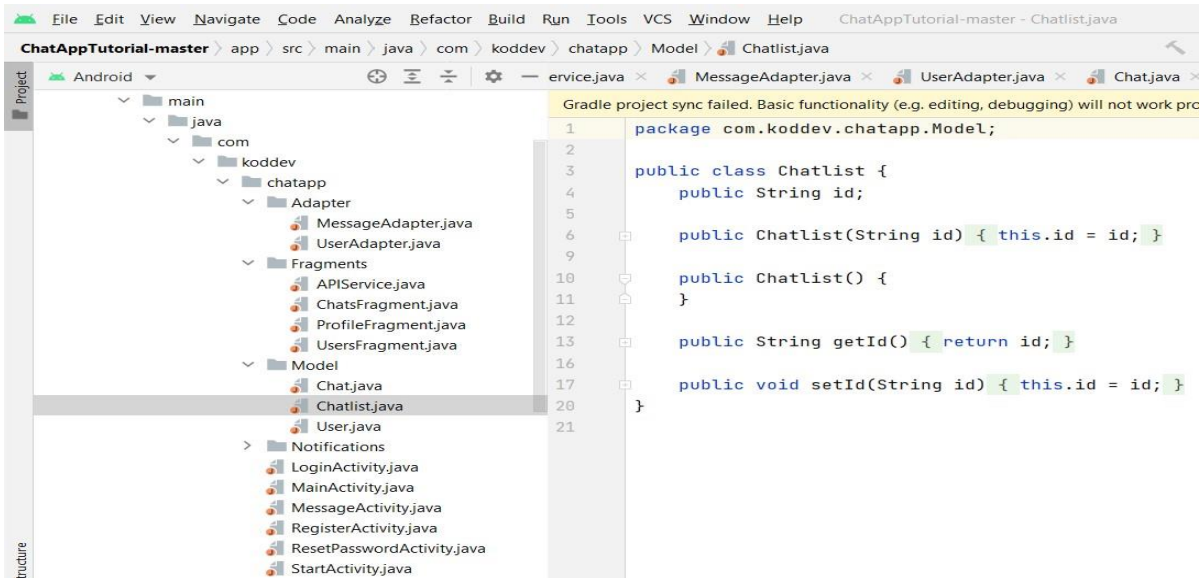



Profile Fragment: Είναι υπεύθυνο για τη προβολή του προφίλ και των πληροφοριών του χρήστη, καθώς επιτρέπει στους χρήστες να αλλάξουν ή να διαμορφώσουν την εικόνα προφίλ τους. Επικοινωνεί με το Firebase Authentication, τη βάση δεδομένων μας και το χώρο αποθήκευσης, ώστε να εισάγει τις απαραίτητες πληροφορίες και δεδομένα για αυτή τη λειτουργία




Users Fragment: Είναι υπεύθυνο για τη προβολή της λίστας των χρηστών και προμηθεύει με μία λειτουργική αναζήτηση το φιλτράρισμα των χρηστών βάση ενός ερωτήματος στη βάση δεδομένων μας. Επικοινωνεί δηλαδή με τον τρόπο που θέλουμε, με σκοπό να φέρουμε και να φιλτράρουμε τα δεδομένα των χρηστών μας.

## MODELS




Chat, Chat List, User: Τα τρία αυτά μοντέλα κλάσεων ενθυλακώνουν τα απαραίτητα properties και έχουν τις απαραίτητες μεθόδους για να έχουμε πρόσβαση στα properties τους. Επίσης, μπορούμε να δημιουργήσουμε αντικείμενα Chat , ChatList , User και να αποθηκεύσουμε ή να ανακτήσουμε δεδομένα από την βάση δεδομένων μας.

## NOTIFICATIONS

Client: Αποτελεί ένα πιο βολικό τρόπο της χρήσης της βιβλιοθήκης Retrofit, καλώντας τη μέθοδο get Client, η οποία εγγυάται ότι μόνο μία αντιγραφή(instance) δημιουργείται και χρησιμοποιείται πολλές φορές μέσα στο application. Αυτή η αντιγραφή μπορεί να χρησιμοποιηθεί για network requests σε μία συγκεκριμένη URL διεύθυνση.

Data: Η κλάση αυτή είναι υπεύθυνη για όλα τα δεδομένα που μπορεί να έχει ένας χρήστης στο application μας. Επίσης, παίρνει σαν παραμέτρους τα εισερχόμενα και εξερχόμενα μηνύματα που μπορεί να έχει ένας χρήστης.

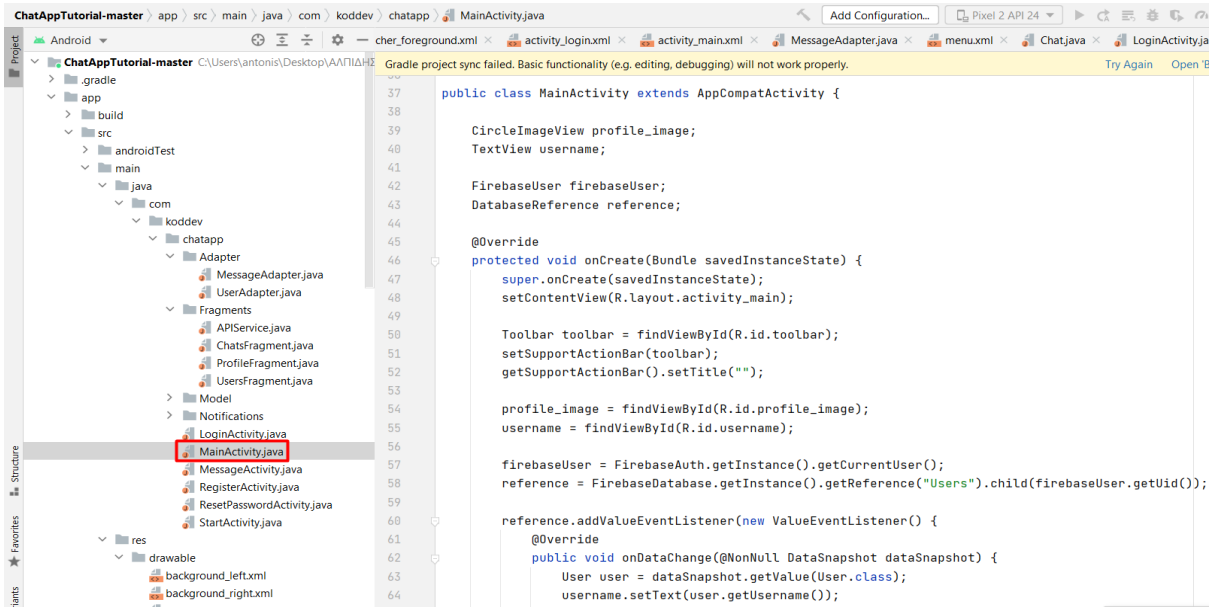
My Fire base Id Service: Το service αυτό δημιουργήθηκε για να ελέγχει το κάθε νέο token ενός χρήστη όταν η συσκευή του κάνει εγγραφή μέσω του Firebase Cloud Messaging(FCM). Παραλαμβάνει το token της εκάστοτε συσκευής του χρήστη , ελέγχει αν ο χρήστης έχει συνδεθεί , και μετά ανανεώνει το token στη βάση δεδομένων μας. Αυτό μας είναι πολύ χρήσιμο γιατί μπορούμε να διαχειριστούμε τα notifications ανάλογα με τα token του κάθε χρήστη και να στείλουμε ειδοποιήσεις σε συγκεκριμένες συσκευές ή χρήστες.
Firebase Messaging:

My Response: Έχουμε ή δεν έχουμε σύνδεση με τη βάση δεδομένων ή τον server μας(Retrofit).

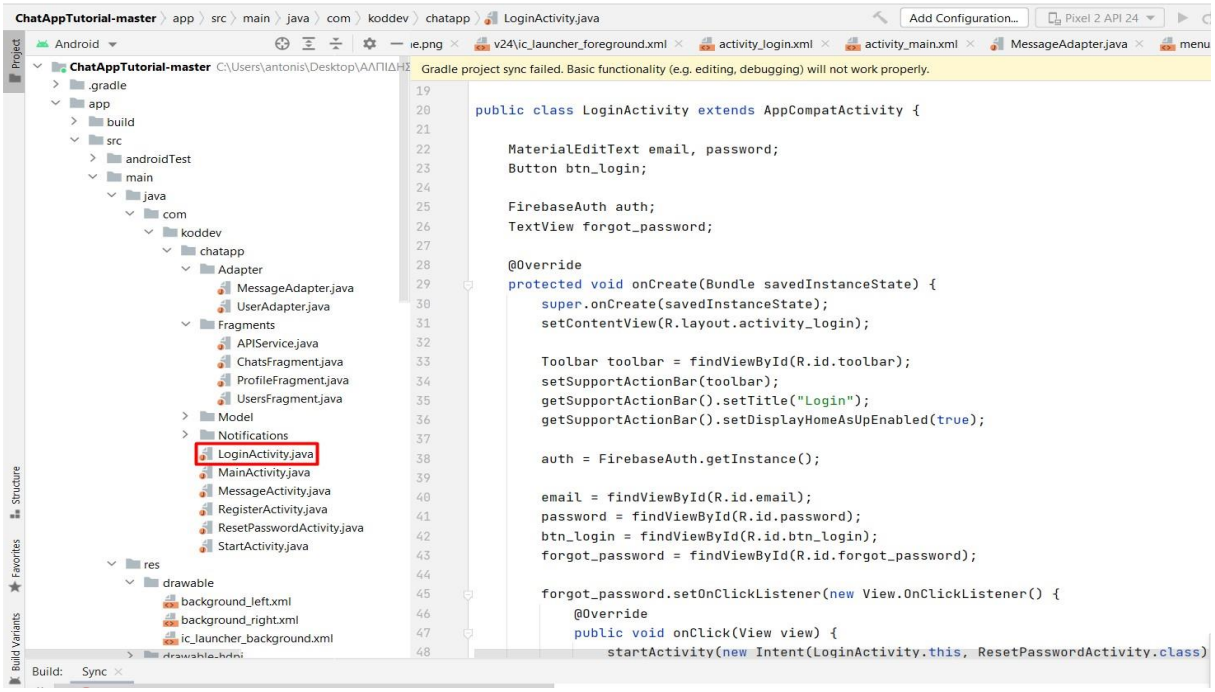
Oreo Notification: Είναι μία κλάση που δημιουργεί ειδοποιήσεις τύπου Oreo και απευθύνεται σε συσκευές Android με λογισμικό Oreo ή κατώτερο. Δημιουργεί επίσης το κανάλι των ειδοποιήσεων , ανακτά το Notification.Manager(Διαχειρίζεται τις ειδοποιήσεις μας) και μας προμηθεύει με μία μέθοδο που δημιουργεί Oreo-Style ειδοποιήσεις με τη χρήση της κλάσης Notification.Builder(χτίσιμο ειδοποίησης).

Sender: Παραλαμβάνει σαν παραμέτρους τα δεδομένα του κάθε χρήστη μας και χρησιμοποιείται για να τα περάσουμε τα δεδομένα μέσα στα notifications όπου αυτά χρειάζονται, αναλόγως με το χρήστη μας.
Token: Είναι η κλάση που δημιουργεί το token του κάθε χρήστη, που χρησιμοποιείται μέσα στο application μας.

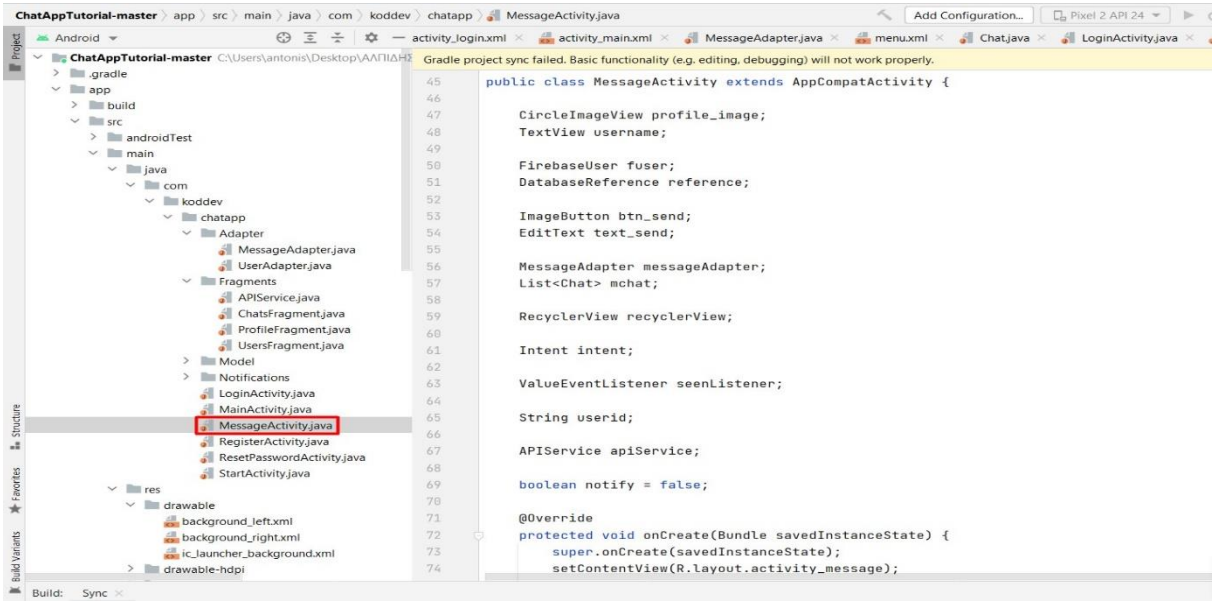
## ACTIVITIES



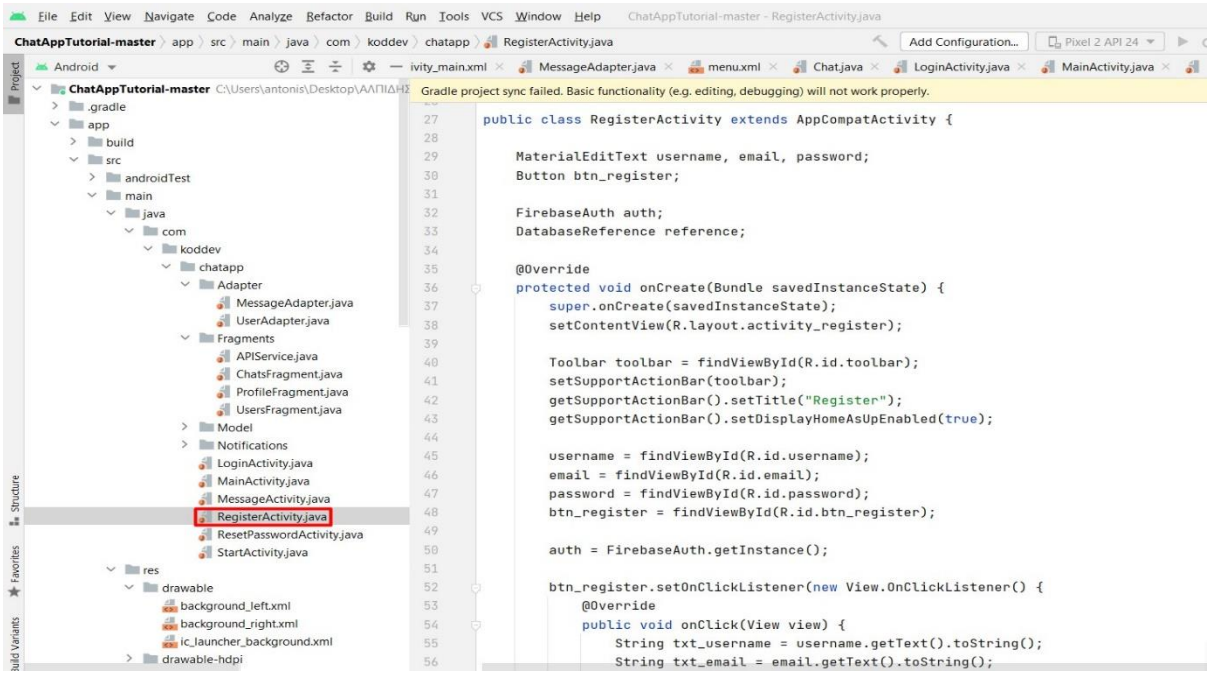

Main: Αποτελεί τη Main του application μας, η οποία διαχειρίζεται τον χρήστη μας και το interface καθώς και τη λειτουργικότητα όλων των κλάσεων της εφαρμογής μας.



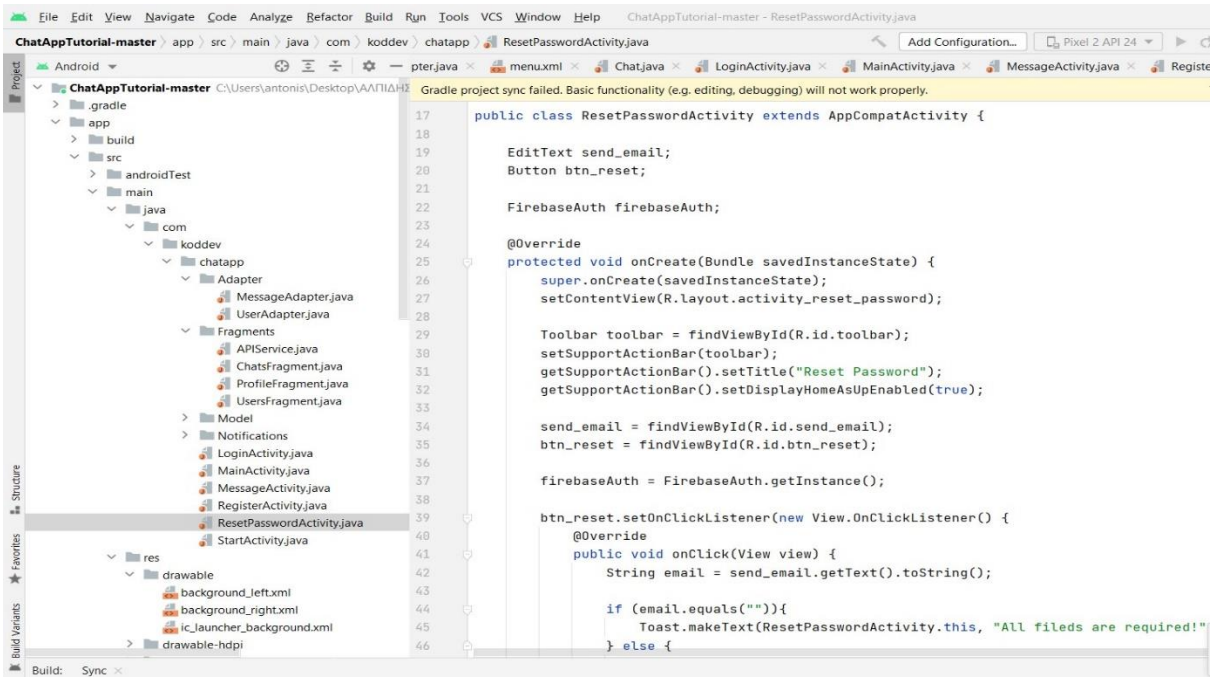

Login: Διαχείριση της εισόδου του χρήστη (log in) στο application μας. Επίσης διαχειρίζεται το User Interface της εφαρμογής , δηλαδή αυτό που βλέπει ο χρήστης μας όταν ανοίγει την εφαρμογή, επιπλέον ελέγχει τα επικυρωμένα στοιχεία του χρήστη και αν είναι σωστά (email,password,photo etc) και μέσω του Firebase Authentication ελέγχεται η αυθεντικότητα του χρήστη μας.



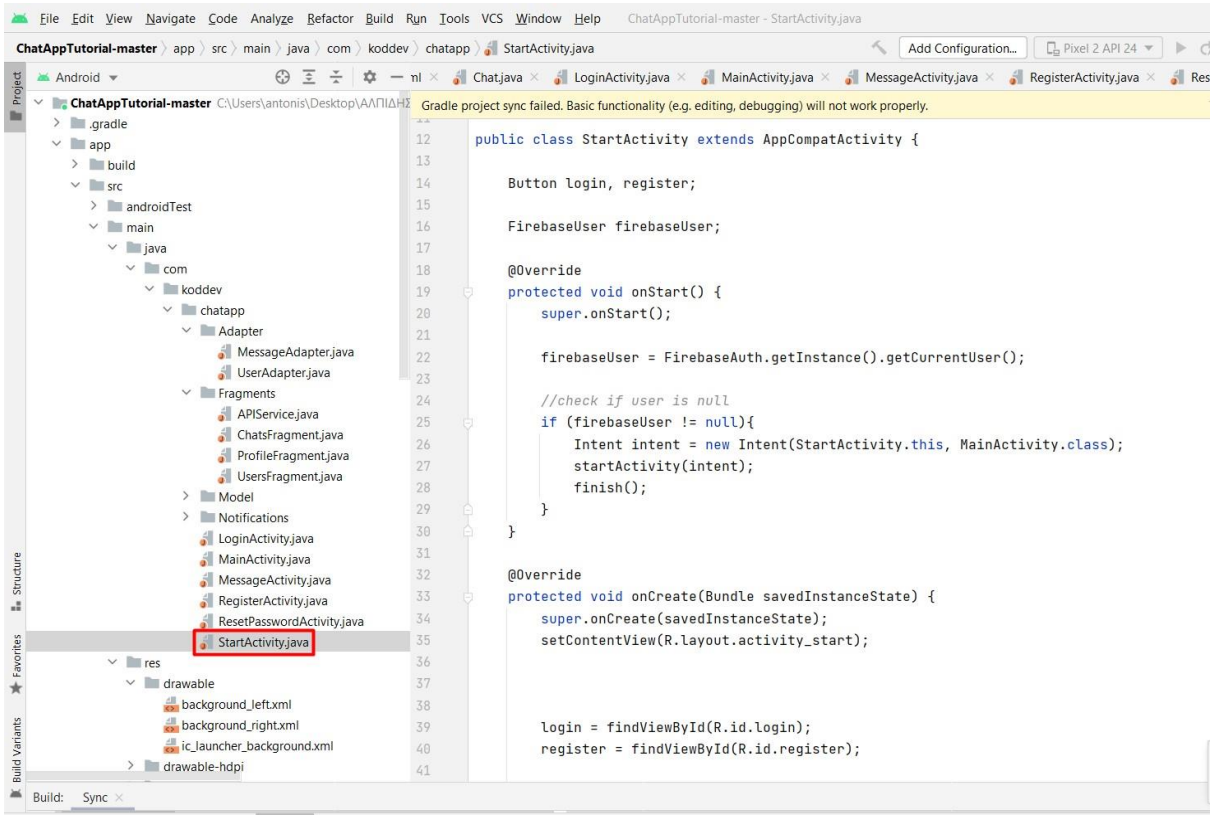

Message: Διαχείριση των μηνυμάτων μεταξύ των χρηστών μας (read, seen,send, notifications etc.). Πιο αναλυτικά , το activity Message είναι υπεύθυνο για το UI των μηνυμάτων, δηλαδή αποστολή και λήψη μηνυμάτων, ανανέωση του status των μηνυμάτων, προβολή του ιστορικού μηνυμάτων και αποστολή/λήψη ειδοποιήσεων.




Register: Διαχειρίζεται το registration του χρήστη μας. Πιο αναλυτικά, είναι υπεύθυνο για το UI registration του χρήστη, δηλαδή φτιάχνει ένα καινούργιο χρήστη μέσω του Firebase Authentication, αποθηκεύει τα δεδομένα του χρήστη στη βάση δεδομένων μας και εφόσον όλη αυτή η διαδικασία είναι επιτυχημένη καταλήγουν τα δεδομένα μας στη Main.

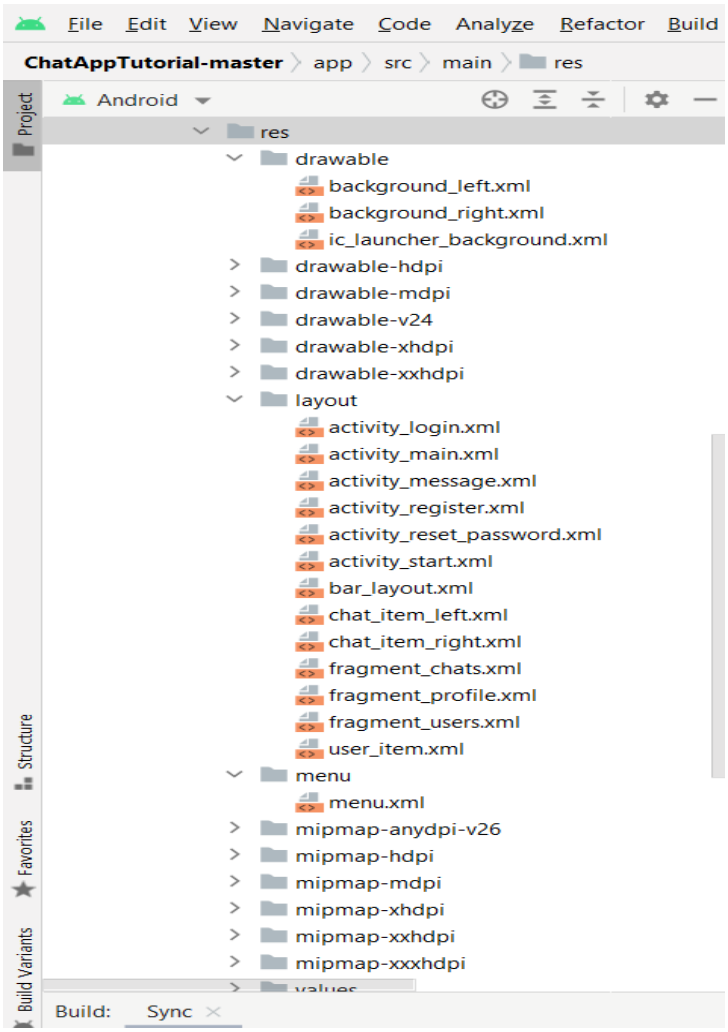



Reset Password: Διαχειρίζεται την αλλαγή κωδικού του χρήστη μας. Πιο αναλυτικά, στέλνει ένα email αλλαγής κωδικού στο χρήστη μας με τη χρήση της κλάσης Firebase AUTH, σε περίπτωση που ο χρήστης μας για τον οποιοδήποτε λόγο ξέχασε η δεν πληκτρολόγησε σωστά τον κωδικό του για να εισέλθει στην εφαρμογή.




Start: Είναι υπεύθυνο για την είσοδο μας στην εφαρμογή (entry point) και διαχειρίζεται τον προσδιορισμό για το εάν ένας χρήστης έχει κάνει login ή όχι. Ουσιαστικά, η Start Activity αποφασίζει εάν θα γίνει η εκκίνηση της Main δηλαδή της εφαρμογής μας ή εάν πρέπει να δείξει στο χρήστη μας τις επιλογές login και registration.

## DRAWABLE XML




Τα υπόλοιπα αρχεία που υπάρχουν μέσα στο πρόγραμμα είναι για τη σχεδίαση την εμφάνιση και το στυλ της εφαρμογής μας, δηλαδή τι θέλουμε εμείς να βλέπει ο χρήστης όταν ανοίγει την εφαρμογή κάνει για παράδειγμα register, log in , send a message , reset password , be offline or online etc.. Ουσιαστικά όλο το User Interface μας είναι χτισμένο στα αρχεία αυτά.

# Κεφάλαιο 5 - Συμπεράσματα και Μελλοντικές Προσθήκες

Η ανάπτυξη μία εφαρμογής Android ήταν μια διαδικασία που μας άφησε ανάμεικτες εντυπώσεις. Από την μία η ανάπτυξη μίας βασικής εφαρμογής είναι σχετικά απλή, με την εταιρεία Google να προσφέρει το απαραίτητο documentation που καθοδηγεί τον προγραμματιστή ώστε να ξεκινήσει το συντομότερο να γράφει κώδικα. Όμως η διαδικασία δεν είναι τόσο απλή όσο φαίνεται αρχικά. Αρχικά, το documentation δεν καλύπτει αναλυτικά διάφορα θέματα, κάτι το οποίο γίνεται εμφανές όταν προσπαθήσουμε να μπούμε σε λεπτομέρειες και να υλοποιήσουμε λειτουργίες που δεν εμπίπτουν στα παραδείγματα που δίνονται. Τέλος ένα απρόσμενο εμπόδιο που συναντήσαμε ήταν η μετάβαση από τις βιβλιοθήκες android σε android, που είχαν σαν αποτέλεσμα να σπαταληθεί αρκετός χρόνος.

Επιπλέον, οι τεχνολογίες σχετικά με τα έξυπνα κινητά τηλέφωνα και τον τρόπο με τον οποίο αναπτύσσονται εφαρμογές σε αυτά. Παρόλα αυτά, σημαντική θεωρούμε και την επιτυχή ανάπτυξη μιας λειτουργικής εφαρμογής ισάξιας με τις υπάρχουσες δημοφιλείς εφαρμογές, κάτι το οποίο καθιστά πρόδηλο το γεγονός ότι αρκεί να υπάρχει η κατάλληλη ιδέα και τεχνογνωσία, ώστε να δημιουργηθεί κάτι το πρωτότυπο. Συμπερασματικά, ύστερα από εκτενή ενασχόληση με την θεματική μας, η εργασία ολοκληρώθηκε προσφέροντας μας, μάλιστα, ορισμένες γνώσεις τις οποίες αναντίρρητα πρόκειται να χρησιμοποιήσουμε μελλοντικά.

Οι εφαρμογές υπολογιστών που αναπτύχθηκαν τα τελευταία χρόνια, ανεξάρτητα από το στόχο και τους τύπους χρηστών, είναι σχεδόν βέβαιο ότι θα εκτελούνται σε πολλαπλές μηχανές συνδεδεμένες σε κάποια τοπική ή ευρύτερη διεπαφή δικτύου.

Η Java για παράδειγμα, είναι μια γλώσσα προγραμματισμού που προσπαθεί να λύσει τα παραπάνω προβλήματα. Αναπτύχθηκε από την Sun Corporation και έγινε πολύ δημοφιλής σε σύντομο χρονικό διάστημα μετά την ανάπτυξή του. Αρχικά προσανατολιζόταν στην ανάπτυξη λογισμικού για οικιακά ηλεκτρονικά είδη, αλλά έχει εξελιχθεί σε μια ολοκληρωμένη γλώσσα με πολλά χαρακτηριστικά σύγχρονων γλωσσών προγραμματισμού και υποστήριξη αντικειμενοστραφούς προγραμματισμού. Η επιτυχία της γλώσσας έγκειται στο ότι μπορεί να χρησιμοποιηθεί για τη σύνταξη ασφαλών εφαρμογών Διαδικτύου υψηλής απόδοσης που μπορούν να εκτελούνται εγγενώς σε διαφορετικά περιβάλλοντα προγραμματισμού και αρχιτεκτονικές και παρέχει έναν ισχυρό τρόπο για την παροχή δυναμικού περιεχομένου σε εφαρμογές πολυμέσων.

**Μελλοντικές Προσθήκες**

Σε μελλοντικό χρόνο, η εφαρμογή διαθέτει πολλαπλές πτυχές οι οποίες δύνανται να επεκταθούν περαιτέρω, με την σταδιακή προσθήκη νέων χαρακτηριστικών. Αρχικά μια σημαντική προσθήκη θα ήταν να μπορεί ο χρήστης να εισέλθει στην εφαρμογή χρησιμοποιώντας Facebook, Google , Instagram κτλ. Έπειτα μια σημαντική προσθήκη θα ήταν η δημιουργία chat groups για πολλά άτομα μαζί , το οποίο δοκιμάστηκε αλλά συναντήσαμε αρκετές δυσκολίες. Μία άλλη ιδέα για την επέκταση της εφαρμογής είναι η ενσωμάτωση υπηρεσιών κοινωνικής δικτύωσης, για παράδειγμα ο χρήστης να μπορεί να δει τις θέσεις των φίλων του στο Facebook ή στο Google Plus. Μία τέτοια εφαρμογή όμως παρουσιάζει διάφορα θέματα προστασίας προσωπικών υπηρεσιών και χρειάζεται περισσότερη μελέτη και προετοιμασία.

# ΒΙΒΛΙΟΓΡΑΦΙΑ - REFERENCES


[1] Darcey L. and Shane C., "Αναπτυξη Εφαρμογων με το Android-Δευτερη Εκδοση", Μ.Γκιούρδας, Αθήνα, 2011

[2] Meier R. "Professional Android 4 application development". John Wiley & Sons, 2012

[3] Murphy M. "Android Programming Tutorials".CommonsWare, 2010.

[4] Lee W. "Beginning Android 4 Application Development". John Wiley & Sons, 2012.

[5] "Android Studio", https://developer.android.com/intl/zh-cn/sdk/index.html

[6] "Android", https://www.android.com/

[7] "Smartphone", Wikipedia, https://en.wikipedia.org/wiki/Smartphone

[8] Firebase Tutorial Documentation google.com

[9] The Android Developer's Cookbook Building Applications with the Android SDK

[10] S. Nasehi, J. Sillito, F. Maurer, and C. Burns. What makes a good code example? A study of programming Q&A in stackoverflow. In 28th IEEE International Conference on Software Maintenance (ICSM'12), page 25.34, 2012

[11] Why is the Android emulator so slow? http://stackoverflow.com/questions/1554099, October 2009.

[12] M. Butler, "Android: Changing the Mobile Landscape", Pervasive Computing, (2011), pp. 4-7. [13] B. Proffitt, "Open Android-For better and for worse", Spectrum, (2011), pp. 22– 24.

[14] K. W. Tracy, "Mobile Application Development Experiences on Apple's iOS and Android OS", Potentials, (2012), pp. 30 – 34.

[15] A. Shabtai, Y. Fledel, U. Kanonov, Y. Elovici, S. Dolev and C. Glezer, "Google Android: A Comprehensive Security Assessment", Security & Privacy, (2010), pp. 35 – 44.

[16] Gewirtz, D., (2017). Which programming languages are most popular (and what does that even mean). Fetched from https://www.zdnet.com/article/which-programming-languages-are-most-popular-and -what-does-that-even-mean/

[17] Pontin, J., (2018). The problem with programming, November 28, 2006. Fetch from https://www.technologyreview.com/s/406923/the-problem-with-programming/

[18] Wayner, P., (2015). 13 programming languages defining the future of coding. Fetched from https://techbeacon.com/13-programming-languages-defining-future-coding

[19] Wexelblat, R., (2014). History of programming languages. Fetched from https://books.google.es/books?id=Hy-jBQAAQBAJ&lpg=PP1&ots=jl-J6nuYU2

[20] Yoav, G., (2016). “A Primer on Neural Network Models for Natural Language Processing.” σε Journal of Artificial Intelligence Research 57, 2016