NATIONAL AND KAPODISTRIAN UNIVERSITY OF ATHENS

SCHOOL OF SCIENCE
DEPARTMENT OF INFORMATICS AND TELECOMMUNICATION

INTERDISCIPLINARY POSTGRADUATE PROGRAM "INFORMATION TECHNOLOGIES IN MEDICINE AND BIOLOGY"

MASTER THESIS

Development of a Real-Time Gene Database for an Android Application

Joana H. Dulaj

Supervisor: **Dr. Zoe Cournia,** Researcher - Assistant Professor Level, Biomedical Research Foundation of the Academy of Athens (BRFAA)

Athens

October 2018

ΕΘΝΙΚΟ ΚΑΙ ΚΑΠΟΔΙΣΤΡΙΑΚΟ ΠΑΝΕΠΙΣΤΗΜΙΟ ΑΘΗΝΩΝ

ΣΧΟΛΗ ΘΕΤΙΚΩΝ ΕΠΙΣΤΗΜΩΝ
ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΤΗΛΕΠΙΚΟΙΝΩΝΙΩΝ

ΔΙΑΤΜΗΜΑΤΙΚΟ ΜΕΤΑΠΤΥΧΙΑΚΟ ΠΡΟΓΡΑΜΜΑ "ΤΕΧΝΟΛΟΓΙΕΣ ΠΛΗΡΟΦΟΡΙΚΗΣ ΣΤΗΝ ΙΑΤΡΙΚΗ ΚΑΙ ΤΗ ΒΙΟΛΟΓΙΑ"

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

Ανάπτυξη βάσης δεδομένων πραγματικού χρόνου για γονίδια σε εφαρμογή Android

Ιωάννα Χ. Ντουλάι

Επιβλέπο*υσα*: **Δρ. Ζωή Κούρνια,** Ερευνήτρια Γ', Ίδρυμα Ιατροβιολογικών Ερευνών Ακαδημίας Αθηνών (ΙΙΒΕΑΑ)

ΑΘΗΝΑ

Οκτώβριος 2018

MASTER THESIS

Development of real-time gene database for an Android application

Joana H. Dulaj

S.N.: PIV0159

SUPERVISOR: **Dr. Zoe Cournia**, Researcher - Assistant Professor Level, Biomedical Research Foundation of the Academy of Athens (BRFAA)

EXAMINATION COMMITTEE:

Dr. Zoe Cournia, Researcher - Assistant Professor Level, Biomedical Research Foundation of the Academy of Athens (BRFAA)

Dr. Stavros Perantonis, Research Director, Institute of Informatics and Telecommunications, National Center of Scientific Research “Demokritos”

Dr. Emma Anastasiadou, Primary Investigator - Lecturer Level, Department of Genetics, Biomedical Research Foundation of the Academy of Athens (BRFAA).

October 2018

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

Ανάπτυξη μιας βάσης δεδομένων πραγματικού χρόνου για γονίδια σε εφαρμογή Android

Ιωάννα Χ. Ντουλάι

Α.Μ.: ΠΙΒ0159

ΕΠΙΒΛΕΠΟΥΣΑ: **Δρ. Ζωή Κούρνια,** Ερευνήτρια Γ', Ίδρυμα Ιατροβιολογικών Ερευνών Ακαδημίας Αθηνών (ΙΙΒΕΑΑ)

ΕΞΕΤΑΣΤΙΚΗ ΕΠΙΤΡΟΠΗ: **Δρ. Ζωή Κούρνια,** Ερευνήτρια Γ', Ίδρυμα Ιατροβιολογικών Ερευνών Ακαδημίας Αθηνών (ΙΙΒΕΑΑ)

Δρ. Σταύρος Περαντώνης, Διευθυντής Έρευνας, Ινστιτούτο Πληροφορικής και Τηλεπικοινωνιών, ΕΚΕΦΕ "Δημόκριτος"

Δρ. Έμμα Αναστασιάδου, Ερευνήτρια Δ', Ίδρυμα Ιατροβιολογικών Ερευνών Ακαδημίας Αθηνών (ΙΙΒΕΑΑ)

Οκτώβριος 2018

ABSTRACT

Genomic profiling is increasingly incorporated into medical research and the clinical care of patients. It is also widely used in the search for biomarkers and diagnostics, and recently gene databases have been made publicly available for research purposes. At the same time, the technology of smartphones has given rise to a massive change in our lives. In this thesis, we present the development of the Geneus Android application, which provides the users with a real-time database of genes as well as a communication tool through a chat box, using the Firebase technology of Google.

The Firebase Real-time Database is a cloud-hosted database, where data is stored as Javascript Object Notation (JSON) and synchronized in real-time to every connected user. All users share one Real-time Database instance and automatically receive updates with the newest data. Firebase apps remain responsive even when offline, as the Firebase Real-time Database Software Development Kit (SDK) stores the data on disk. Once connectivity is re-established, the client user receives any missed changes, by synchronizing with the current server state.

Given the massive, rapidly changing biological information, which we experience daily, Geneus's innovation lies in using a real-time database in order for users to receive instant updates with the newest data, with no need for downloading a new version of the app. Geneus is developed as a general Android mobile app framework that could be further connected with any other patient database entries (e.g., daily blood sugar data), and it is expected to be utilized by patients, doctors, as well as field experts.

Providing real-time access to biological databases from a smartphone is foreseen to enhance their usability and access as we move towards new technological advances such as targeted next-generation DNA sequencing reactions using mobile phones, and achieving new milestones for telemedicine technologies.

SUBJECT AREA: Real-Time Database

KEYWORDS: android development, firebase, messenger, genes database, authentication, mobile app, real-time database, cloud

ΠΕΡΙΛΗΨΗ

Το γονιδιωματικό προφίλ (Genomic Profile) ενσωματώνεται όλο και περισσότερο στην ιατρική έρευνα και στην κλινική φροντίδα των ασθενών. Χρησιμοποιείται επίσης ευρέως στην έρευνα για την ανακαλυψη νέων βιοδεικτών για διαγνωστικούς και θεραπευτικούς σκοπούς. Ταυτόχρονα βάσεις δεδομένων γονιδίων με ολοένα και περισσότερες πληροφορίες δημοσιεύονται για ερευνητικούς σκοπούς. Την ίδια στιγμή, η τεχνολογία των έξυπνων-κινητών έχει επιφέρει μια τεράστια αλλαγή στη ζωή μας. Σε αυτή την εργασία, παρουσιάζουμε την ανάπτυξη της εφαρμογής Geneus σε περιβαλλον Android, η οποία προσφέρει στους χρήστες την πρόσβαση σε μία βάση δεδομένων πραγματικού χρόνου με γονίδια καθώς επίσης και την δυνατότητα επικοινωνίας μεταξύ όλων των χρηστών μέσω ενός «παραθύρου» μηνυμάτων (chat box), χρησιμοποιώντας την τεχνολογία Firebase της Google.

Στην βάση δεδομένων Firebase Realtime τα δεδομένα αποθηκεύονται σε ένα JSON αρχείο και συγχρονίζονται σε πραγματικό χρόνο για κάθε συνδεδεμένο χρήστη. Όλοι οι χρήστες μοιράζονται ένα στιγμιότυπο της βάσης και λαμβάνουν αυτόματα ενημερώσεις με τα νεότερα δεδομένα. Οι εφαρμογές που χρησιμοποιούν Firebase ανταποκρίνονται ακόμη και όταν είναι εκτός σύνδεσης στο διαδίκτυο επειδή τα δεδομένα έχουν αποθηκευτεί και στη συσκευή. Μόλις αποκατασταθεί η συνδεσιμότητα, ο χρήστης λαμβάνει οποιεσδήποτε αλλαγές έχουν συμβεί και συγχρονίζεται με την τρέχουσα κατάσταση του server (όπου και βρίσκεται η βάση δεδομένων).

Με βάση την καθημερινή ραγδαία συρροή βιολογικών δεδομένων και πληροφοριών, η καινοτομία που προσφέρει η εφαρμογή Geneus είναι η βάση δεδομένων πραγματικού χρόνου μέσω της οποίας λαμβάνουν τα νέα δεδομένα αμέσως μετά από οποιαδήποτε αλλαγή χωρίς να απαιτείται το κατέβασμα μιας νέας έκδοσης της εφαρμογής από το Google Play store. Επιπλέον, το Geneus μπορεί να χρησιμοποιηθεί σαν μια γενική δομή υποστήριξης οποιαδήποτε βάσης δεδομένων, όπως παραδείγματος χάρη για καταγραφή βιοχημικών δεικτών (πχ σακχαρου) στο αίμα, σε Android κινητά, και αναμένεται να χρησιμοποιηθεί από γιατρούς, ασθενείς και επιστήμονες.

Η πρόσβαση σε βάσεις βιολογικών δεδομένων πραγματικού χρόνου από «έξυπνα» κινητά τηλέφωνα αναμένεται να βελτιώσει την χρηστικότητα και την πρόσβασή τους καθώς προχωράμε προς νέες τεχνολογικές εξελίξεις, όπως η στοχευμένη αλληλούχιση του ανθρώπινου γονιδιώματος από κινητά τηλέφωνα και η εφαρμογή τηλεϊατρικών τεχνολογιών σε θεραπείες.

ΘΕΜΑΤΙΚΗ ΠΕΡΙΟΧΗ: Πραγματικού Χρόνου Βάση Δεδομένων

ΛΕΞΕΙΣ ΚΛΕΙΔΙΑ: προγραμματισμός σε Android, firebase, messenger, γονιδιακή βάση δεδομένων,ταυτοποίηση, εφαρμογή σε κινητό τηλέφωνο, βάση δεδομένων πραγματικού χρόνου, υπολογιστικό νέφος

To my family

ACKNOWLEDGMENTS

I would like to take this opportunity to acknowledge those who have facilitated the completion of this thesis. First and foremost, I would like to express my gratitude to my supervisor, Dr. Zoe Cournia, Researcher - Assistant Professor Level, at the Biomedical Research Foundation of the Academy of Athens (BRFAA) for her encouragement, support, and thoughtful advice that have been immensely valuable for me and my academic steps, and for turning an academic project into an applied venture. Her guidance empowered me in my research and in writing this thesis.

For their precious advice in shaping this thesis, I would also like to thank the two members of my MSc diploma thesis examination committee, Dr. Stavros Perantonis, Research Director Institute of Informatics and Telecommunications, National Center of Scientific Research “Demokritos”, and Dr. Emma Anastasiadou, Researcher - Assistant Professor Level, at the Biomedical Research Foundation of the Academy of Athens (BRFAA).

I also would like to thank my fellow lab-mate Alexios Chatzigoulas for valuable suggestions in writing this thesis, and Stamatia Zavitsanou and Christina Athanasiou for all the discussions and suggestions for my thesis presentation.

Last but not least, I would like to thank my family and all my friends for supporting me spiritually throughout this thesis and in my life in general.

CONTENTS

1. INTRODUCTION 14

1.1 Genomic profiling and medical/clinical research 15

1.2 Modern Healthcare 16

1.2.1 Healthcare mobile apps 17

1.3 Study Objectives – Motivation 19

2. METHODS 21

2.1 Android OS and Software Development for Android 21

2.1.1 Application Fundamentals 22

2.1.1.1 App Components 23

2.1.1.2 Manifest File 23

2.1.1.3 App Resources 23

2.1.1.4 Android Software Development Kit (SDK) 24

2.1.2 Gradle File 24

2.1.3 Integrated Development Environments (IDE) 24

2.2 Android Database Technologies 25

2.2.1 SQLite Database 25

2.2.1.1 SQLiteOpenHelper 26

2.2.1.2 SQLite Datatypes 27

2.2.1.3 Type of queries in SQLite 27

2.2.1.4 Query methods 28

2.2.2 Google Firebase Technology 28

2.2.3 Key Capabilities of Firebase Real-time Database 29

2.2.4 Firebase Real-time Database Security Rules 30

2.3 Firebase Authentication 30

2.4 Methodology for an online real-time database of Firebase 30

2.5 Methodology for SQLite Database 31

2.6 Geneus Database 32

3. RESULTS 33

3.1 Workflow of Geneus Development 33

3.1.1 SQLite implementation 33

3.1.1.1 Data Cleaning and Extraction 34

3.1.1.2 SQLite Database Integration 35

3.1.1.3 Issues of SQLite Database Implementation 36

3.1.2 Firebase Implementation 36

3.1.2.1 Data Cleaning and extraction 37

3.1.2.2 Database Integration to Android Firebase Console 37

3.1.2.3 Implementation of communication between Firebase Database and Geneus 39

3.1.2.4 Read Data from Firebase Database 39

3.1.2.5 Write data to Firebase Database 41

3.1.2.6 Firebase Google Authentication 42

3.1.2.7 Firebase Real-time Database Crashes 43

3.2 Geneus Real-time Database Advantages 43

3.3 Geneus: A real-time gene app 44
3.3.1 Competitive landscape 45

3.4 Geneus Activities 45
3.4.1 Geneus Sign in Activity 46
3.4.2 Geneus Main Activity 47
3.4.3 Geneus Search Activity 48
3.4.4 Gene's Profile 49
3.4.5 Community Messenger 50

CONCLUSIONS 51

FUTURE PERSPECTIVES 52

ABBREVIATIONS - ACRONYMS 53

ANNEX I 54

REFERENCES 62

LIST OF IMAGES

Figure 1: Genomic Profiling [14] 15
Figure 2: Projected total global mHealth devices and services revenue from 2014 to 2020 [19] 17
Figure 3: Patient DataViewer [20] 18
Figure 4: XCMS Mobile [21] 19
Figure 5: Android Architecture [31] 22
Figure 6: App Fundamentals 22
Figure 7: Firebase Features [42] 28
Figure 8: Firebase Real-time Database [44] 29
Figure 9: The workflow implementation of Geneus development 33
Figure 10: Data extraction workflow to create the SQLite database file 34
Figure 11: SQLite implementation 35
Figure 12: Data extraction to create the Firebase database file 37
Figure 13: Database Integration to Firebase 38
Figure 14: Implementation workflow of communication between Firebase Database and Geneus 39
Figure 15: Implementation of Reading Data from Firebase 40
Figure 16: Implementation of writing data to Firebase 41
Figure 17: Firebase Authentication Steps 42
Figure 18: Geneus Logo 44
Figure 19: SignIn Activity 46
Figure 20: SignIn Activity(2) 46
Figure 21: Main Activity 47
Figure 22: Geneus Search Activity 48
Figure 23: Gene's Profile Activity 49
Figure 24: Gene's Profile Activity(2) 49
Figure 25: Community Messenger 50

LIST OF TABLES

Table 1: SQLite Storage Classes 27
Table 2: Competitive apps.....45

PREFACE

The master thesis ‘Development of a Real-Time Gene Database for an Android Application’ has been conducted at the Biomedical Research Foundation Academy of Athens for the completion of the Postgraduate Program "Information Technologies in Medicine and Biology" (I.T.M.B.), Department of Informatics and Telecommunications, National and Kapodistrian University of Athens, Greece.

The first Chapter presents the motivation of the study, the use of genomic profiling in medical research and clinical care of patients, the modern healthcare and some mobile apps related to healthcare services and scientific activities.

In the second Chapter, the theoretical foundation of the present work is presented. Firstly, the Android Operating System and software development for mobile applications is outlined. Next, the Android Database Technologies, the Firebase Authentication and algorithmic methodologies are described. Furthermore, the details of Geneus Database implementation are presented.

The results of the present thesis are illustrated in Chapter three. Firstly, the implementation of SQLite Database and Firebase Realtime Database are presented. Next, the advantages and disadvantages of both implementations with an emphasis on the choice of the Geneus Realtime Database are described. In addition, the “Geneus” mobile application and its features are presented.

The conclusions constitute the epilogue of this thesis, along with the possible future extensions of Geneus application.

Finally, an Annex with the results of a UI/UX questionnaire based on the users’ experience by using Geneus application is presented.

1. INTRODUCTION

The collection of medical data has always been a costly and laborious process, especially in research where sizeable datasets are mandatory for extracting meaningful correlations that would aid patient care. A series of advances in medical science and IT facilitated not only the collection of biological data, but also its conversion into relevant patient insights, which may be utilized to provide more precise and personalized care. [1] Personalized medicine is a new framework for medical care that utilizes patient genomic information to diagnose, prevent and treat diseases. This new framework, includes targeted diagnostics and treatment based on patient history, ancestry, and genetic profile. Consequently, a new industry has emerged: digital health and healthcare big data analytics. This industry aims to create data-driven predictions to identify patients at risk before a health issue arises. As such, the use of big data promises a more efficient healthcare system to the benefit of patients. [2]

Moreover, recent advances in genomic research are leading to a new era in medicine and biology. [3] Genomic profiling is increasingly incorporated into medical research and the clinical care of patients. [4] In the next few years, the use of genomic data in healthcare will be increased, and decisions regarding the prevention and treatment of diseases will be highly based on an individual's genetic makeup. Therefore, in the future, genomic data is expected to be effectively used in healthcare and wellbeing, as personalized medicine will be incorporated in standard patient care through disease profiling, patient profiling, and novel therapies. [5]

To achieve the objectives of personalized medicine the development of a reference database of genomes is required in order to be used in clinical care and research. Moreover, research will be benefitted by the improved availability of genomic data and the systematic collection of genetic information that has been increased in several countries during the past recent years. Therefore, for research purposes, many databases have been made publicly available. [6][7][8]

At the same time, the technology of smartphones has brought out a massive change in our daily lives, and many innovative apps for healthcare and for optimizing scientific activities have been developed. The most popular operating system for smartphones is the Android, and its prevalence is only increasing.[9] Android is an open source operating system, and therefore applications for users can be developed and deployed at no cost using its features.[10] Developers pursue to use these features of Android to provide the users the facility of accessing personal data regardless of location and time.

Furthermore, the progress of mobile technology has minimized the reliance on large hardware instrumentation and scientists have been benefiting from the convenience of mobility in their daily lives. In the past few decades, research has become more data-intensive and specialized equipment is often shared between different groups. One major disadvantage of most bioinformatics tools is their limitation to personal computers (PCs), but when the scientists are outside of the laboratory, data and instrument access is typically limited to a Virtual Private Network (VPN) based connection. These have not only caused an increase in the necessary amount of time for analysis but also, to non-traditional working hours. Therefore, mobile devices, particularly smartphones, offer the advantage of accessibility to a wide range of operations and functionalities regardless of location and time.

As mobile OS is growing larger and their hardware is becoming as powerful as any standard computer device, data computation from mobiles is becoming part of our daily routine. Many mobile applications have to store and manage a huge amount of data, and the most common way of storing and managing data is the development of a database (DB). In a typical scenario, a mobile application uses a database that is

hosted in the cloud and connects remotely to it in order to access its data. This implies that an active and extremely fast network connection is required for a mobile app in order to be responsive and functional. Furthermore, many applications preserve DB locally or make a copy of a DB located on the cloud locally in the device which is synchronized once per day or whenever there is a network connection. Using a copy of the DB stored locally in the device will assist in developing faster and responsive applications which are responsive even when there is no or limited internet connectivity.

The databases with the above functionalities are known as embeddable databases. Mobile applications can be (statically or dynamically) linked to them in order to create and manage their own – private or shared – databases locally on the device.[11] In Android application development, there are several embeddable databases. In this thesis, SQLite and Firebase Real-time database are explored.

1.1 Genomic profiling and medical/clinical research

Genomic profiling is a laboratory method that is used to output the genes of a person's DNA or of a specific cell type and the way these genes interact with each other and with the environment. This method may be used to identify why some people get certain diseases while others do not, or why people react in different ways to the same drug. Furthermore, Genomic profiling may be used to develop new ways for diagnosing, treating, and preventing diseases, such as cancer. [13]

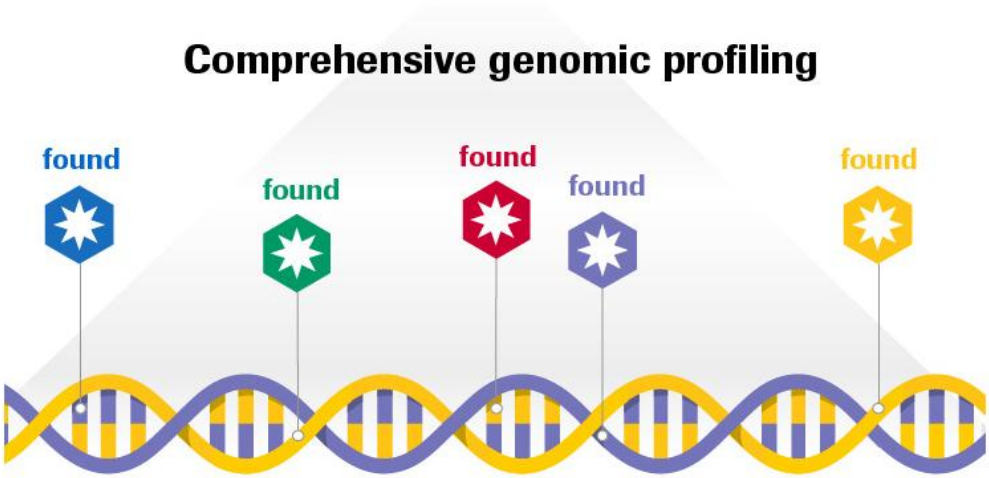


Figure 1: Genomic Profiling [14]

This technique [Figure 1] is commonly used and also benefitted many areas of biological and biomedicine research to advance, from the understanding of genetic information to the causes and treatment of human diseases.

The research of genetics creates new possibilities to identify the causes of human health and disease, as well as to create new means for disease prevention and targeted care. The use of genomic data, i.e., information about an individual's entire genetic makeup, is rapidly increasing in healthcare. In future, health promotion and treatment of diseases will be highly based on individual genetic makeup. Genomic data will aid the scientists to target the screening of the diseases more precisely, to diagnose more accurately, and to select the most effective treatment against diseases.[15]

Extensive use of genomic data is expected to produce significant benefits. Individuals will be able to receive more effective medicines and care. It will be possible to diagnose diseases at an early stage and more accurately. In addition, scientists can identify individual risks for diseases and prevent them before their onset.

The systematic collection of genetic information has increased rapidly in several countries during recent years, which have drawn up or are drawing up, strategies and action plans to utilize genomic data. For example, the British National Health Service (NHS) has launched a 300 million pound genome project for reading the genetic makeup of 100,000 people. In Germany, the Ministry of Education and Research has earmarked 360 million euros to promote personalized medicine during the next three years. Moreover, in the United States, a 215 million dollar genome project has been launched. All these projects aim at utilizing the results in the healthcare sector. At the same time, they generate data which can be used in developing new medicines and methods for early detection and treatment of diseases. [16]

1.2 Modern Healthcare

Information technology and communication technology have witnessed rapid technological breakthroughs. These breakthroughs will receive appreciable respect if they alter into beneficial use to the citizens. Inadequate healthcare facilities and a shortage of healthcare staff and experts result in difficulties in meeting the healthcare requirements of the population of underdeveloped countries or developing countries. The only way to cope up with these technologies is through the use of these advancements in the healthcare. Healthcare is directly responsible for people's health and well-being, thus it needs to be continuously developed and improved to provide not only efficiency and productivity for itself and decrease expenses, but also to create an improved quality service to improve patient safety and quality of care.

The area of healthcare has attracted the researcher's attention more than before due to many factors, for example [17]:

- The increasing number of aging population
- New types of diseases were detected because of the change in the lifestyle
- Rising costs in healthcare and its related services
- Shortage of medical resources and healthcare personnel

Furthermore, the evolution of technology has brought new fields: e-Health and m-health have arisen. E-health and m-health can be any electronic device or monitoring system that is applied by physicians in the healthcare practice or by individuals to monitor or improve their health status. E-health typically refers to online and offline computer-based applications while m-health refers to applications for mobile phones. Such tools can be used to stimulate a positive health behavior change and assist people to have a healthier lifestyle, or to support the diagnosis and treatment of diseases.[18]

E-health and m-health tools are being used increasingly, and a wide array of medical apps has been developed to provide access to patients into medical services. The e-Health and m-Health market is one of the fastest growing areas of mobile technology. There are many healthcare mobile applications providing a variety of functionalities to users such as accessing their electronic medical record, viewing laboratory results, scheduling appointments, managing chronic conditions, tracking disease outbreak information and locating clinical trials.

The following statistic [Figure 2] illustrates the global digital health market in 2015 and 2016, and a projection for 2017 until 2020, by major segment. In 2019, the mobile health market is expected to reach 37 billion U.S. dollars worldwide. The digital health market is expected to reach 206 billion U.S. dollars by 2020, driven particularly by the mobile and wireless health market. [19]

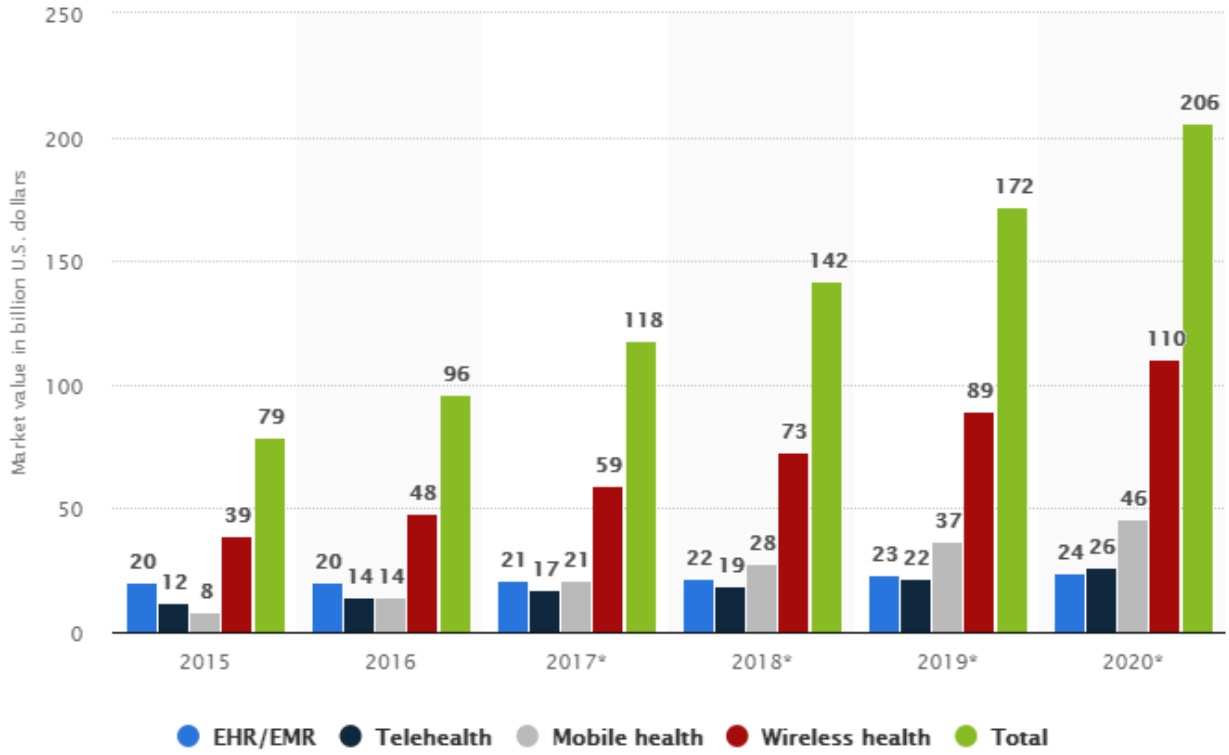


Figure 2: Projected total global mHealth devices and services revenue from 2014 to 2020 [19]

1.2.1 Healthcare mobile apps

Healthcare environments have turned to be technologically oriented. Therefore, plenty of mobile applications have been developed to provide the functionalities of optimizing the remote healthcare services for patients like accessing their electronic medical record, viewing laboratory results or scheduling appointments. One such healthcare app is Patient DataViewer. [Figure 3] Patient Data Viewer is an Android application for doctors, which provides the latest information on the patients' vital health parameters like blood pressure, heart rate, etc. Furthermore, the doctors through Patient DataViewer can create notes regarding a patient, view them and receive a notification whenever a patient's health is at risk (whenever any of the vital parameters of the patient exceed a critical limit). All this information is collected and sent to the doctor's Android device from the hospital server. [20]

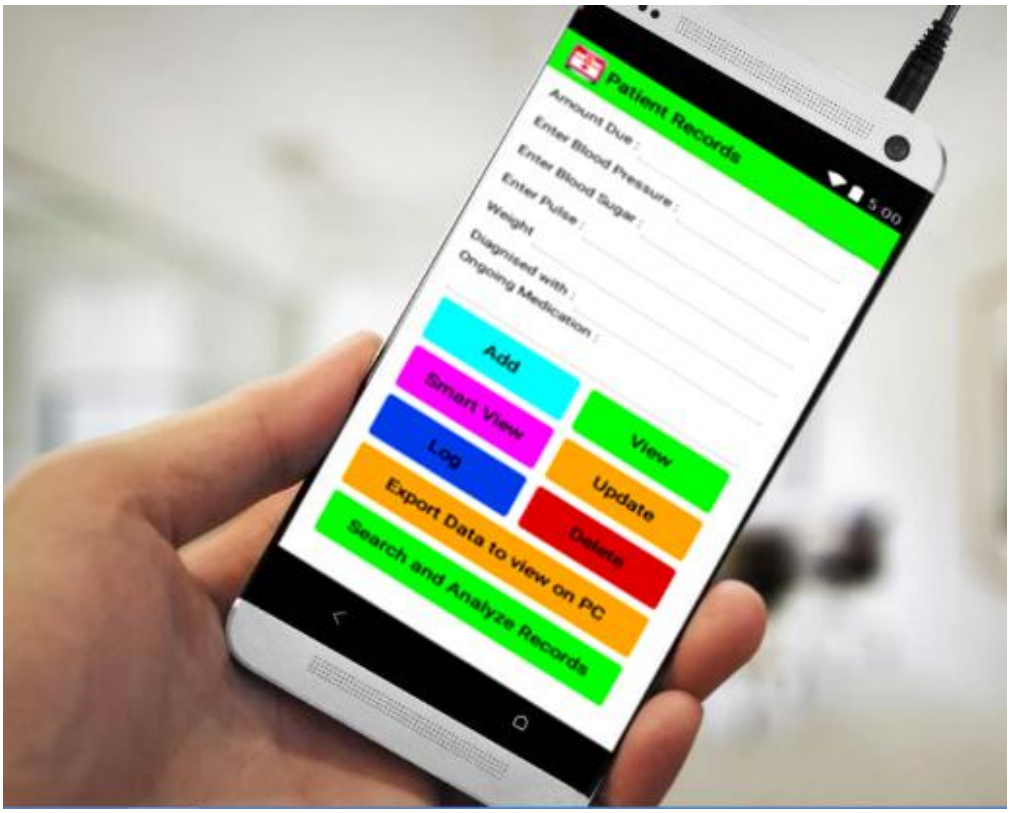


Figure 3: Patient DataViewer [20]

In addition, many bioinformatics applications have been developed for ‘on the go’ data analysis and visualization in order to raise productivity and expedite research. Active data screening is an essential component of many scientific activities, and with mobile technology, the reliance on large hardware instrumentation has been minimized.

XCMS Mobile is an application [Figure 4] that minimizes the reliance on large hardware instrumentation and provides a remote metabolomic data screening platform for mobile devices. Metabolomics is the technology for the analysis of metabolites and facilitates in identifying biomarkers. [21]

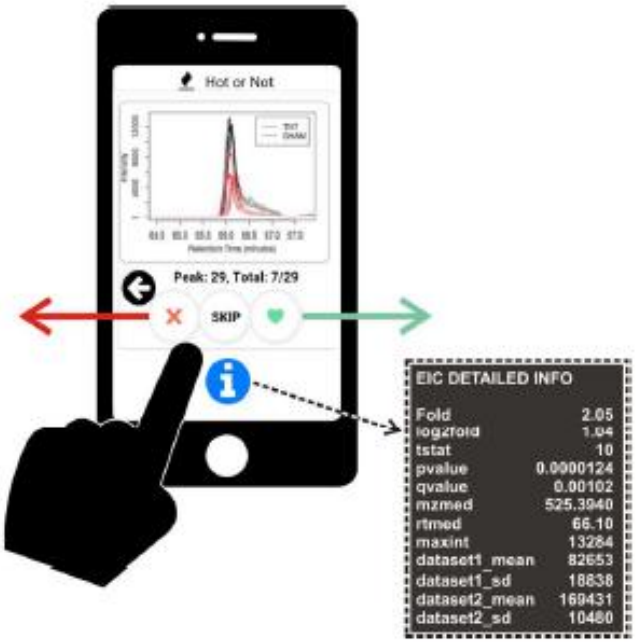


Figure 4: XCMS Mobile [21]

Specifically, XCMS Mobile provide the capabilities for remote monitoring of data processing, real-time notifications for the data processing, visualization and interactive analysis of processed data (e.g., cloud plots, principal component analysis, box-plots, extracted ion chromatograms and hierarchical cluster analysis), and database searching for metabolite identification. This application is available both on Apple iOS and Google Android operating systems.

1.3 Study Objectives – Motivation

Having in mind that the global market for the mobile medical and biological apps is projected to reach $206 billion by 2020 [Figure 2] we aimed to create an app that will reduce the reliance on personal computers for data monitoring and will minimize resources for remote data access on the move. Therefore, we designed an application for providing real-time database access integrated into an Android application in order for users to benefit from the convenience of mobility in their everyday lives.

In this study, we present the “Geneus app”, an Android application incorporating a real-time gene database that eases the access to gene’s details, and also enabling a Group Chat functionality for communication among the users. Currently, Geneus contains the NCBI gene database. [6] However, this framework could be extended as a general mobile app supporting any other patient database entries (e.g., daily blood sugar data).

Geneus potential users could be Experts, Doctors, Patients or any typical user that could be interested in genes’ information.

Through Geneus, experts such as Biologists, enjoy straightforward access to real-time gathered information such as gene details from their smartphone by avoiding the process of searching and downloading a database of genes, which may contain already obsolete information. By using Geneus, experts may not miss any data updates because new information is received automatically.

Within Geneus, patients may obtain gene information within a simple implementation and can communicate with a doctor or other patients-users through the Group Chat

activity of Geneus. In the future, Geneus could be incorporated into the software of medical centers as a service to patients. Therefore, the app could contain the personal sequencing data per each patient or medical examination results and receive real-time updates about them. Furthermore, in future implementations patients may communicate privately with their doctor about medical issues. Moreover, doctors through Geneus can communicate with their patients directly and offer medical advice. Finally, the doctors will be able to provide patients with personal medical examination data results as well as real-time updates about them. Obviously, this implementation should fully comply with the current GDPR standards and regulations and a dedicated Data Protection Officer should be staffed.

2. METHODS

An Android mobile phone application was created for this project, called “Geneus”. Geneus exploits the capabilities of the Android operating system and Android application fundamentals for the development and implementation of the functionalities that it supports. Furthermore, the implementation of its database was based on two different database technologies; SQLite database and Firebase Realtime Database.

2.1 Android OS and Software Development for Android

Android is an open source mobile operating system based on Linux Kernel OS. Open source defines that the source code for any application is not being developed for sale or profit-oriented purposes. Therefore any manufacturer can access, customize and adapt it. Thus, without any license fee, users and developers use the source code of Android.

As it is based upon Linux kernel, it maintains the Linux core features. A kernel is the first layer of software that interacts with the device hardware. In Android OS, the kernel is responsible for providing a basic architectural model for process scheduling, resource handling, memory management, networking, isolation, etc. It must be noted that while Android is built on Linux Kernel, Google has maintained its forked version of the Linux Kernel specifically for Android since 2010. [25]

For security purposes, Google has designed their OS to execute applications in specialized sandboxes, which reduce the ability of malware attacks to the applications in smartphones. [26] Furthermore, in Android OS, each application is isolated from every other application, and each one works on its personal memory space, and no application can access data from another. Additionally, message passing methods are used in Android to provide communication between the applications.[27]

Android software easily integrates with the device’s existing applications. Therefore, many healthcare and biological applications are being developed on Android due to its ability to interact with the hardware at a high level. [28] Moreover, Android applications can be executed by non-mobile household devices like ovens, AC, refrigerators, washing machines, etc.

Android applications can be written by using Kotlin, Java or C++ languages. Although, Android is highly dependent on its Java architecture, and thus Java is its primary development language. [29]

Moreover, Android contains a virtual machine, the Dalvik Virtual Machine (DVM), which is responsible for running Android applications and producing a .dex file (Dalvik Executable File). A .dex file is the Compiled Android application code file and is, in turn, zipped into a single .apk (Android Package) file. An apk file contains all the contents of an Android app and is used to install an application in Android-powered devices. [30]

The architecture of the Android OS is designed so that the communication between the end user and the application level is extremely easy. Android Architecture consists of four layers. The layer of Applications, the layer of Application Fundamentals, the layer of Native Libraries and Android Runtime, and the layer of Linux Kernel. [Figure 5] Each component of the stack and the elements within each layer are integrated and provide optimal application development and execution environment for mobile devices. [31]

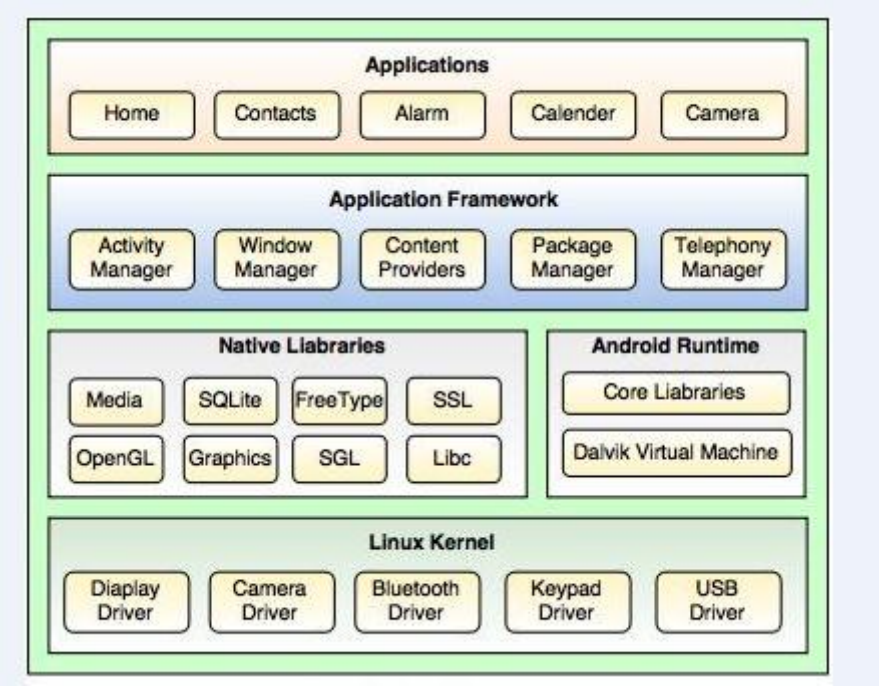


Figure 5: Android Architecture [31]

Linux is the heart of Android architecture, and it provides a level of abstraction between the hardware devices and the upper layers of the Android software stack. The native libraries such as Media, WebKit, SQLite, etc. are situated on the top of a Linux kernel. Android Runtime includes core libraries and Dalvik Virtual Machine (DVM), which are responsible for running Android applications. The Dalvik Virtual Machine (DVM) is like Java Virtual Machine (JVM) in Java, but DVM is optimized for mobile Devices. Android framework provides numerous classes and interfaces for Android application development and high level services to the applications in the form of Java classes. Finally, in the layer of Applications, the applications that we install in Android-powered devices are located.

2.1.1 Android Application Development Fundamentals

Before a developer starts the implementation of an Android application, they should study carefully Android's Application Fundamentals. Application Fundamentals of Android consists of the App Components, the Manifest File, the App Resources, and the SDK Tools. [Figure 6] Android SDK tools compile the code along with any data and resource files into an .apk file.[32]

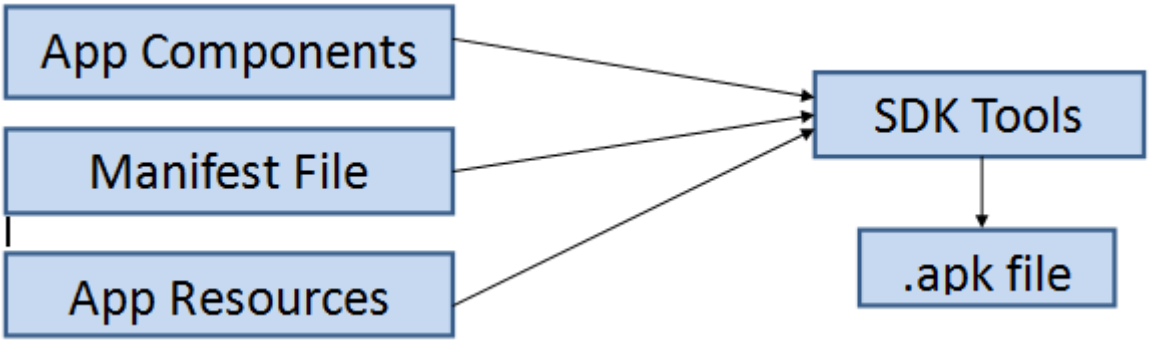


Figure 6: Android App Development Fundamentals

2.1.1.1 App Components

Any app is made of four components, the Activity class, the Service class, the Content Providers class, and the Broadcast Receiver class. [33]

The Activity class is used in order to create a screen for an Android activity. In Geneus, the Activity class is extended in order to create the screen of each application activity.

The Service class is responsible for background processing. Therefore, when a Service class is used the Android OS is informed that a process of an application or an application will run in the background (even when the user is not directly interacting with the application). In Geneus, when a user clicks on the back button or home button, Android OS of the device is informed that Geneus is not terminated and runs in the background.

The Content Providers class facilitates an application to manage access to data stored by itself, and by other apps, and finally provide a way to share data with other apps. This class encapsulates the data and provides mechanisms for defining data security. Content Provider is the interface that connects data in one process with code running in another process. Geneus uses the class of Content Providers in the implementation of SQLite database.

The Broadcast Receiver class responds to broadcast messages from other applications or from the system itself. These messages are sometimes called events or intents. For example, applications can initiate broadcasts to inform other applications that some data have been downloaded to the device and are available for use. Therefore, Broadcast Receivers are used for communication between applications.

2.1.1.2 Manifest File

Every app project must have an AndroidManifest.xml file (with precisely that name) at the root of the project source set. The manifest file describes essential information about the app to the Android build tools, the Android operating system, and Google Play. [34] More specifically a Manifest file:

- It contains information related to the package, the activities, the services, the broadcast receivers, the content providers
- It is a required xml file for every Android application
- It is located inside the root directory of an application
- In that file, the required permissions for an Android application are declared, e.g., internet permission for network access.
- It declares the Android API of the app. The API defines the version of libraries and components used for the development of the application.

2.1.1.3 App Resources

An Android app is composed of more than just code because it requires resources that are separate from the source code, such as images, audio files, and anything related to the visual presentation of the application.

Using app resources the update of various characteristics of the app is achieved without modifying many segments of the code. [35]

2.1.1.4 Android Software Development Kit (SDK)

The Android SDK (software development kit) is a set of development tools used to develop applications for the Android platform. The Android SDK includes the following tools:

- Debugger
- Required Libraries
- Handset emulator based on QEMU
- Relevant documentation for the Android application program interfaces (APIs)
- Sample code
- Tutorials

It supports older versions of the Android platform in case developers wish to target their applications at older devices. The development tools are downloadable components, so in case a developer has downloaded the latest version and platform, older platforms and tools can be downloaded for compatibility testing. [36]

2.1.2 Gradle File

After the implementation of an application's functionalities, the developer may build an .apk file that will be used to install the application in an Android device. Developers could ease that process by using Gradle files.

Gradle is an advanced build toolkit used in Android Studio IDE, to automate and manage the build process. It allows defining flexible custom build configurations. Each build configuration can define its own set of code and resources while reusing the components common to all versions of the app. Android plugin for Gradle works with the build toolkit to provide processes and configurable settings that are specific for building and testing Android applications. [37] Thus, using Gradle files the processes of Building, Testing, Publishing, and Deployment are automated.

2.1.3 Integrated Development Environments (IDE)

Android developers have the option to choose from three different IDEs, but the official and most used environment is Android Studio, which is highly integrated with external frameworks like Unity3D. The most common Android development IDEs are:

- Android Studio
- IntelliJ IDEA
- NetBeans

2.2 Android Database Technologies

Android mobile OS is continuously growing and is considered as the most used OS in the world (it has already exceeded Windows OS). At the same time, big data is growing rapidly and is predicted to grow at an annual rate of 60% for structured and unstructured data. [2] Therefore, the is an emerging and pressing need for developers to manage and store big data; the optimal way for this is by using databases.

For decades, databases were handled either on the server-side or cloud, and the communication between mobile devices and the databases was performed through the network. However, if an application relies on network connectivity, the user experience may be unpredictable. Thus, in order to design applications more responsive and less dependent on Internet connection, offline capabilities are gaining popularity. Nowadays, apps store locally data in files as a DB or produce a copy of the DB over the cloud onto the local device and synchronizes the data whenever network connectivity is re-established. In this way, the applications are responsive and functional even in the absence of network connectivity. [11]

Based on the above, mobile databases require specific features. The storage is limited on mobile devices, therefore a database should be lightweight in order to consume little memory. It is crucial that the communication between the database and the application does not rely on a server-client communication and a copy of a cloud database instance is stored in the device's memory. Furthermore, the process of searching or retrieving large amounts of data from databases should be performed in a fast and efficient way, and the data should be secured following strict security rules. Although many mobile databases exist, not all of them satisfy the mentioned requirements.

2.2.1 SQLite Database

In Android software development various database technologies exist; a commonly used one is the SQLite database system. SQLite is an open source relational database management system (RDBMS), which is by default embedded in Android and iOS. It was developed by D. Richard Hipp and is a lighter version of SQL, which was designed for mobile OS. Furthermore, it is the most widely deployed SQL database engine in the world, as it is used by several widespread browsers, operating systems, and embedded systems. SQLite is characterized as an embedded database system because it is provided in the form of a library that is linked in applications. Using SQLite, a self-contained, serverless, zero-configuration, transactional SQL database engine is implemented. [38]

- ✓ **Self-contained:** A single library contains the entire database system, which is integrated directly into a host application. Thus, it requires very minimal support from external libraries or the operating system.
- ✓ **Serverless:** SQLite does not require a separate server process or system to operate, because it accesses its storage files directly.
- ✓ **Zero-configuration:** SQLite does not require to be "installed" as there is no "setup" procedure. Thus, there is no need for an administrator to create a new database instance or assign access permissions to users. Moreover, SQLite does not use configuration files, and there is no need for the OS to be informed that SQLite is running.
- ✓ **Transactional:** SQLite transactions allow safe access from multiple processes or threads.

Additionally, SQLite can be stored both on disk and memory and is very fast on retrieving data. Using SQLite database management system, the data are being stored in tables that consist of rows and columns. The intersection of a row and a column is called field and fields contain data, references to other fields, and references to other tables. The rows contain the values of the fields and are identified by unique IDs. The columns contain the names of the fields, which are unique per each table.

2.2.1.1 SQLiteOpenHelper

The most important class of SQLite is SQLiteOpenHelper because it provides methods for handling database operations such as creating the database, upgrading it, reading data from the database and writing data in it. Its methods include:

- onCreate(): called when the database is created for the first time
- onUpgrade(): called if the application code contains a more recent database version number reference
- onOpen(): called when the database is opened
- getWritableDatabase(): opens or creates a database for reading and writing
- getReadableDatabase(): creates or opens a database for reading only
- close(): closes the database

Thus, SQLiteOpenHelper is a class, which provides the methods to manage database creation and version management. [39]

2.2.1.2 SQLite Datatypes

As it was mentioned before, using SQLite database management system the data is stored in tables. Each column of the database's tables is defined to store specific data types. Unlike other database systems, SQLite uses dynamic type system. Therefore, the value stored in a column determines its data type, not the column's data type. In addition, it is not required to declare a specific data type for a column when a table is created. In case a column with the integer data type is declared, it can store any data type such as text, blob, etc., without SQLite complaining about this. [40] The following table illustrates five storage classes in SQLite:

Table 1: SQLite Storage Classes

Storage Class	Meaning
NULL	NULL values mean missing information or unknown.
INTEGER	Integer values are whole numbers (either positive or negative). An integer can have variable sizes such as 1, 2,3, 4, or 8 bytes.
REAL	Real values are real numbers with decimal values that use 8-byte floats.
TEXT	TEXT is used to store character data. The maximum length of TEXT is unlimited. SQLite supports various character encodings.
BLOB	BLOB stands for a large binary object that can be used to store any kind of data. The maximum size of BLOBs is unlimited.

Geneus database schema contains two tables. The first table stores gene entries and contains 10.000 rows and 9 columns. The second table stores the sent messages of Geneus and the rows are dynamically added whenever a user sends a message, but the columns are static and store the message text, the username and the timestamp of the sent message. The fields of every row in both tables are defined as TEXT.

2.2.1.3 Type of queries in SQLite

As it was mentioned before the SQLite database system is used to store and manage data. This procedure is performed by executing SQLite queries. Queries are used to retrieve data from the database, to update values of the rows, to insert new rows in the database, and to delete rows. The following functions are used to execute SQLite queries:

- insert() is method that insert rows in the database
- execSQL() executes a single SQL Statement
- put() is a method that updates name/value pairs of database rows
- delete() is a method that delete rows by specifying the table name and a where clause that contains criteria of deletion

- SELECT statement is used to retrieve data from rows that meet the given criteria (criteria of where clause of a query)

2.2.1.4 Query methods

Queries can be created by using the SQLiteQueryBuilder class and return a Cursor object. Cursor object is a pointer to the location before the first result element of the result set of the query. Therefore, the Cursor should be moved to the first element of the result set, and then the data of the result set is retrieved. Finally, when the process of retrieving data ends, the cursor should be closed to release the memory.

- rawQuery(String sql, String[] selectionArgs) runs the provided SQL query and returns a Cursor of the result set
- query(String table, String[] columns, String selection, String[] selectionArgs, String groupBy, String having, String orderBy, String limit) queries the given table and returns a Cursor over the result set

2.2.2 Google Firebase Technology

Firebase is a BaaS(Backend as a Service) platform for building Web, Android, and IOS applications. It commenced as an YC11 startup and grew up into a next-generation app development platform on Google Cloud Platform. The Firebase tools and services are exposed through SDKs. The main features provided are a real-time database, storage, and authentication. [41]

Firebase has a family of products [Figure 7] that can support a developer in three fields: growing, earning, and development. Therefore, it provides developers with a number of tools and cloud services to assist them develop high-quality apps, grow their user base, and earn more profit.

Geneus exploits the features of Firebase Real-time Database and Authentication to implement its functionalities.

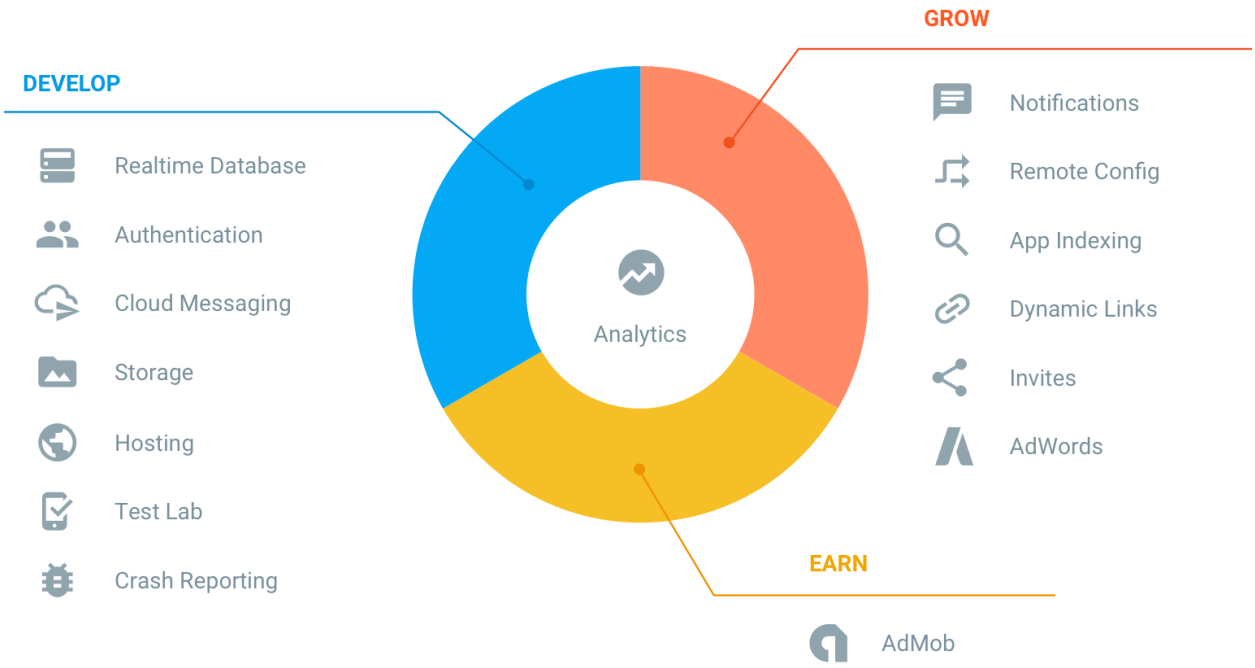


Figure 7: Firebase Features [42]

2.2.3 Key Capabilities of Firebase Real-time Database

Firebase Real-time Database is a cloud-hosted NoSQL database, where data are being stored as JSON and synchronized between connected users in real-time. Firebase synchronizes the database with every connected client instead of using HTTP requests. This means that there is only one common instance of the database for every connected user, which automatically receives updates. Each device is considered as a client for Firebase Real-time Database and data are available even when the client device is offline because of the Firebase offline capabilities. Finally, it is designed only to allow operations that will be executed in milliseconds.[43]

In details key capabilities of Firebase Real-time Database are:

- **Real-time:** Firebase Real-time Database uses data synchronization in order for every connected user/client to receive data changes within milliseconds. Thus, Firebase does not use HTTP requests.
- **Offline:** Firebase data is available even when the user is offline because Firebase Real-time Database persists data on the device (data is stored in cache memory) in order to be available even offline. The data is synced with the cloud when the device reconnects to the Internet.
- **Accessible from Client Devices:** The Firebase Real-time Database can be accessed directly from a mobile device or web browser [Figure 8], so there is no need of implementing an application server. Security and data validation are available through the Firebase Real-time Database Security Rules, which are expression-based rules that are executed when data are read or written.

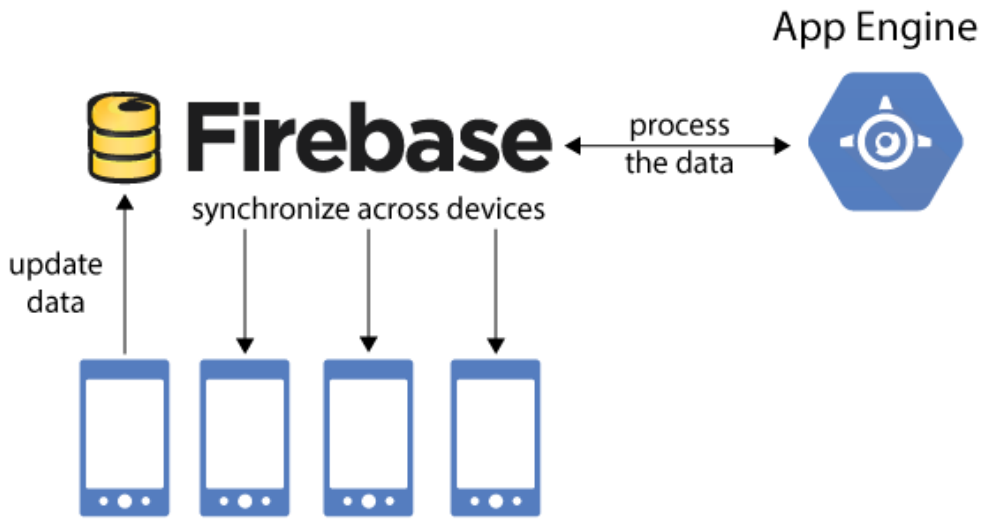


Figure 8: Firebase Real-time Database [44]

Firebase Real-time Database supports only JSON files. JSON files are schema-less database files and store the entire database. Their structure does not contain tables and records; the structure of the database is a JSON tree with nodes. Each node is a JSON object and represents a database's entry. JSON objects should be designed by avoiding nesting child nodes in many layers so that the data can be accessed easily during a read or write process.

The data that are added to a JSON tree are stored as a node in the existing JSON structure with an associated key. Different data types like String, Long, Double, Boolean, Map<String, Object>, List<Object>, Custom Java objects can be stored in Firebase Real-time database, only as String data types.

2.2.4 Firebase Real-time Database Security Rules

The Real-time Database is public facing; therefore it needs a robust security system in order to prevent an attacker from compromising the database. Firebase Real-time Database provides an expression-based rules language, called Firebase Real-time Database Security Rules, to define how the data should be structured and when data can be read or written. These rules are made up of Javascript-like expressions and are contained in a JSON document. The structure of the rules should follow the structure of the data stored in the database. Furthermore, Firebase Real-time Database Security Rules along with the Firebase Authentication feature can define who has access, to what data, and how users can access them. [45] Geneus provides read-access only to the authenticated users and do not allow any user to modify genes' details in the database of genes. Finally, it provides read and write access to the authenticated users in the database of messages in order to write messages in that database and to read the messages sent from other users.

There are three types of rules for enforcing security [46]:

- .read → Describes if and when data are allowed to be read by users
- .write → Describes if and when data are allowed to be written
- .validate → Defines what a correctly formatted value will look like, whether it has child attributes and the data type

2.3 Firebase Authentication

Most of the applications have to identify the users for providing the same personalized experience across all of the user's devices. Authenticating a user's identity allows an application to save user data in the cloud securely. Firebase Authentication provides backend services, easy-to-use SDKs, and UI libraries to authenticate users to an app by implementing its sign in methods. The sign in methods authenticate users with email addresses and passwords, phone numbers, and with popular federated identity providers, including Google Sign-In and Facebook Login and more.

2.4 Methodology for an online real-time database of Firebase

Using Real-time Database most of the system's application code is written on the client side and the developer does not have to worry about factors like hardware, uptime, and scalability of the database server. Data is persisted locally to the device, and even while offline, real-time events continue to fire, giving the end user a responsive experience. When the device is reconnected, the Real-time Database synchronizes the local data changes with the remote updates that occurred while the client was offline, merging any conflicts automatically. Data synchronization uses web sockets, allowing for very concise transactions.

The Real-time Database API is designed only to allow operations that can be executed in milliseconds. Queries fire and respond in near real-time using Real-time Database.

This enables to build a great real-time experience that can serve millions of users without compromising on responsiveness.

Based on Firebase documentation the implementation steps of creating a Realtime Database consists of 5 actions [43]:

1) Integrate the Firebase Real-time Database SDKs: In Gradle file of the application include the Firebase real-time database libraries (SDKs)
2) Create a Realtime database reference to the remote stored data. Use the reference to JSON data in order to set data or listen to data changes
3) Use database references to read/write data or listen to changes
4) Enable offline persistence in order for the application to be available when there is not an internet connection
5) Secure data by using Firebase Real-time Database Security Rules

2.5 Methodology for SQLite Database

Each application that uses an SQLite database interacts with the database, which contains the tables with data, through an activity and a database handler class (commonly named MyDBHandler class). The database handler will be a subclass of SQLiteOpenHelper and will provide an abstract layer of communication between the underlying SQLite database and the activity class. A third class (Model class) is required for storing data entries into the database, and retrieving data from the database, as they are passed between the activity and the handler.

Based on SQLite documentation the implementation steps of creating the interaction between application and SQLite database consists of 4 actions [47]:

1) Create a Model Java Class in order to manage (by using getter and setter methods) the data
2) Create another one Java Class (usually named as MyDBHandler), which extends SQLiteOpenHelper class and assists in maintaining the database and its table
3) Create one more class, which is used for performing all data retrieval and manipulation tasks (Commonly named as DBAdapter)
4) Use references to the above classes in order to handle (insert, update, delete or retrieve data) the database

2.6 Geneus Database

Geneus' database of gene details is downloaded from NCBI as a csv file. [48] The National Center for Biotechnology Information (NCBI) is part of the United States National Library of Medicine (NLM), a branch of the National Institutes of Health (NIH). The NCBI houses a series of databases relevant to biotechnology and biomedicine and is an important resource for bioinformatics tools and services. [49]

The database csv file was cleaned, and only the important information of genes was extracted from the csv file by a Python script. The Geneus database consists of 10.000 human genes; all the fields are stored as Strings and contain the following details:

- Gene Id
- Symbol
- Synonyms
- Chromosome (number of the chromosome where the gene is located)
- Map location (genomic location)
- Description
- Type of Gene
- Other Designations
- Modification Date (of data)

Geneus contains another one database that stores and manages the users' sent messages. The messages database fields are stored as Strings and contain details about the message text, the username and the timestamp of the sent message.

3. RESULTS

In this Chapter, the workflow of Geneus implementation, the use of the application and its activities, and the issues that were experienced during the development of this application and its functionalities, are presented.

3.1 Workflow of Geneus Development

The first step in developing Geneus was the creation of the database structure. For this purpose, two different database technologies; SQLite database and Realtime Firebase Database were explored. Initially, the database was implemented based on the SQLite database system, and subsequently, Realtime Firebase Database was implemented. The two database technologies are compared in the next Chapters. Following the creation of the app database, the “Search Genes” and “Community Messenger” functionalities were implemented in Geneus [Figure 9].

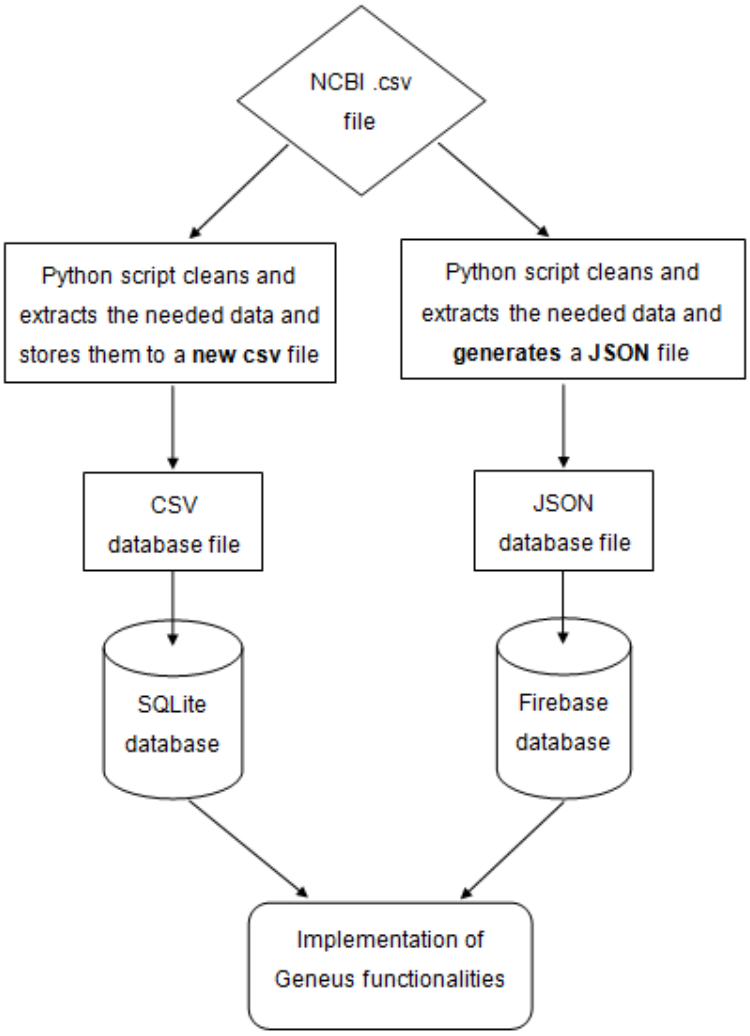


Figure 9: The workflow implementation of Geneus development

3.1.1 SQLite implementation

The implementation of SQLite consists of two steps. First, a .csv file was created by post-processing the .csv file that was downloaded from NCBI, as explained below. The

new .csv file was created by implementing a Python script. Then, the communication between the SQLite database and Geneus application was implemented.

3.1.1.1 Data Cleaning and Extraction

The original csv file that was downloaded from NCBI included 16 columns (tax_id, GeneID, Symbol, LocusTag, Synonyms, dbXrefs, chromosome, map_location, description, type_of_gene, Symbol_from_nomenclature_authority, Full_name_from_nomenclature_authority, Nomenclature_status Other_designations, Modification_date, Feature_type). A python script was developed [Figure 10], which extracts only the significant columns from the .csv file in order to provide to the user in a sufficient way the basic details of a gene. Therefore, the script creates a new csv file with 10.000 rows and 9 columns. The new .csv file is the entity of Geneus database file and is stored in the Assets folder of the Android application project. The details that the Geneus gene database contains are:

- Gene Id
- Symbol
- Synonyms
- Chromosome (number of the chromosome where the gene is located)
- Map location (genomic location)
- Description
- Type of Gene
- Other Designations
- Modification Date (of data)

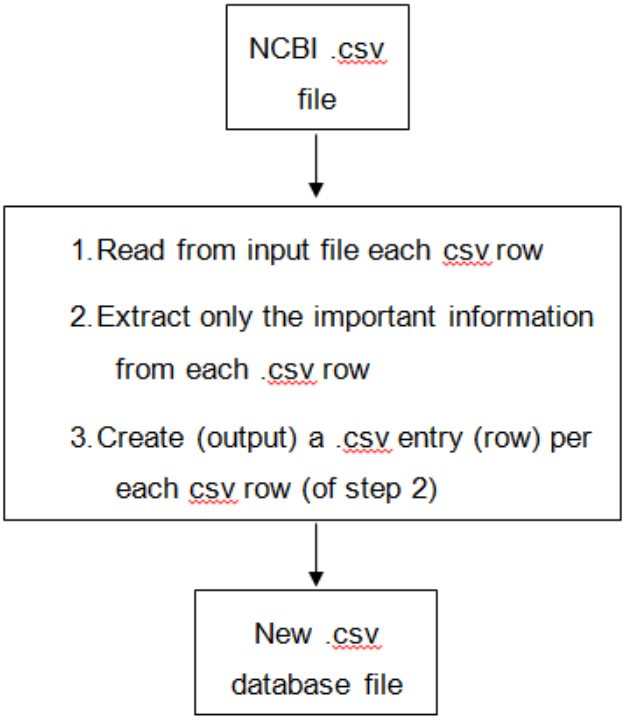


Figure 10: Data extraction workflow to create the SQLite database file

3.1.1.2 SQLite Database Integration

After the development of the SQLite database file, the next step is to perform the communication between Geneus application and SQLite database [Figure 11].

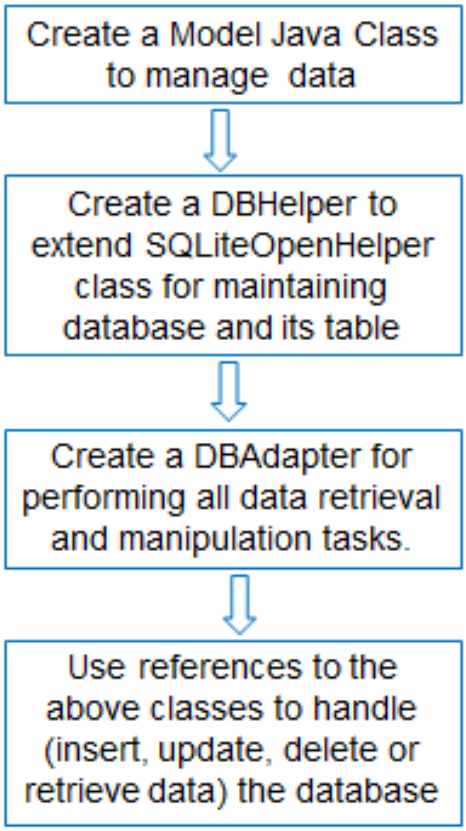


Figure 11: SQLite implementation

The communication between Geneus and the SQLite database is performed by extending the SQLiteOpenHelper class. This class performs “Create, Read, Update and Delete” (CRUD) operations on the database. The workflow of Figure 11 contains all the steps mentioned in Chapter 2.6.

In more detail, these steps describe that a model class was created to manage and store the gene data details by using setters and getters methods. This model java class contains the same fields included in the database of genes (column fields) and facilitates to store a new gene entry in the database, to retrieve gene details from the database by using get methods or to update a field of a gene entry by using set methods.

Then, a DBHelper class that extends SQLiteOpenHelper class was created. DBhelper inherits the methods of SQLiteOpenHelper class (because of extend keyword) and performs the CRUD operations on the database. In DBHelper class were implemented (overridden) the two inherited methods of SQLiteOpenHelper class; the onCreate() method that creates a table, which stores gene database details included in the csv file that was created in Chapter 3.1.1, and the onUpgrade() method that is called when an update is released in order for the user to receive the new version of gene database. Furthermore, in that class are implemented methods about gene data retrieval, insertion, deletion, and update.

The last class created was DBAdapter. This class defines how the gene's data will be displayed by the application and uses references of the above class to perform data retrieval and manipulation processes (implemented in DBHelper class).

3.1.1.3 Issues of SQLite Database Implementation

Using the implementation of SQLite database some issues were detected. As it was mentioned in Chapter 3.1.1, the csv file of the database was stored locally in an application's folder. Therefore, the size of the application was extremely increased because of the 10.000 genes' details, which are too many data for a mobile app. The size of an application could discourage the users from downloading an application that requires a large capacity of device memory. In addition, the size affects the installation, and the launching of Geneus, which has a long duration in conjunction with the size of the database file. Moreover, because of the locally stored database file during the installation of the app, if the back button is pressed the installation of the database is terminated, which results the application not to contain the entire database.

A final issue is that updates by Google Play store are received asynchronously by the app users. Users do not receive at the same time a notification by Google Play store to update an installed app to a new version. Consequently, if any change is performed to the genes' details of the database, many users will share different instances of the database.

Based on the above issues, it was decided that the SQLite database implementation was not the best database technology for Geneus functionalities.

3.1.2 Firebase Implementation

In order to provide with real-time data information Geneus users, the Google Firebase Real-time database was implemented because of its capabilities explained in Chapter 2.3.3; this is a real-time, offline, and accessible from client devices database.

The implementation of Realtime Firebase Database within Geneus consisted of two steps. First, a JSON database file was created by modifying the Python script described in Chapter 3.1.1.1 in order to generate a JSON tree database file (instead of a .csv file). Later, the communication between Firebase database and Geneus application was implemented.

3.1.2.1 Data Cleaning and extraction

Data within the Firebase Real-time Database is stored as JSON objects, and the format of the database is a JSON tree. Unlike an SQL database, tables or records are not used; adding an entry to the JSON tree creates a node in the existing JSON structure with an associated key.

The original NCBI database was downloaded as a .csv file. Therefore, in order to be imported to Firebase Database, the .csv should be converted to a JSON tree file. This conversion was achieved by modifying the Python script [Figure 12] described in Chapter 3.1.1.1 in order to create JSON objects. All the JSON objects compose the JSON tree database file of Geneus.

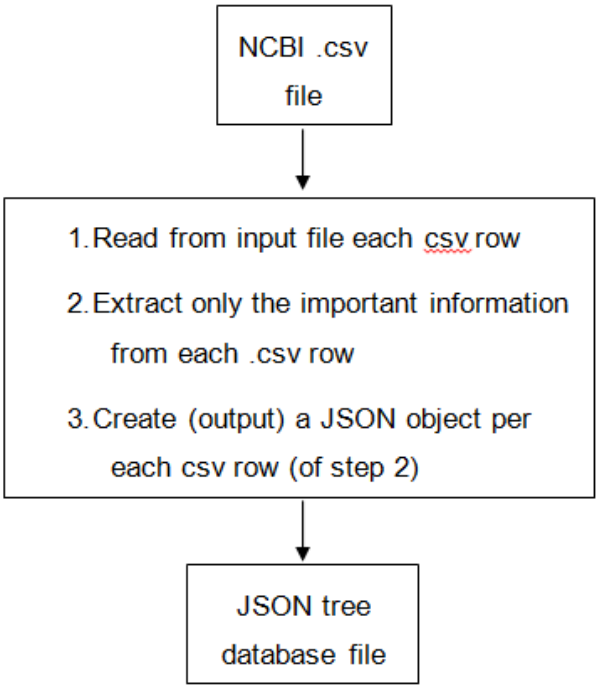


Figure 12: Data extraction to create the Firebase database file

3.1.2.2 Database Integration to Android Firebase Console

After the development of the JSON database file, the database integration to Firebase console was performed. Firebase console provides options like creating, managing and deleting Firebase project apps. By clicking on a specific Firebase project app in the Firebase dashboard, the app database can be viewed and modified in real-time, the Firebase Real-time Database Rules can be settled, the app authentication can be managed, the deployment can be performed, and the analytics can be viewed.

In order to integrate the data of JSON tree in Firebase database, a project to firebase console was created. Then, in the Gradle file of the app project the dependencies of Firebase services were added. The required dependencies of Geneus app are Authentication and Real-time database. Additionally, when access in the Firebase console, it is required to insert the project's package name and the SHA-1 key of the application. The Project package name is located in the Manifest file as well as the SHA-1 key, which is a unique signature generated during the build process of every Android app. The result of the above-mentioned steps was the generation of a configuration file that contains the Google Services. [Figure 13] This file should be

installed into the project app/module folder, being a mandatory file in every Android application that exploits features of Firebase technology because it performs the communication between the application and Firebase.

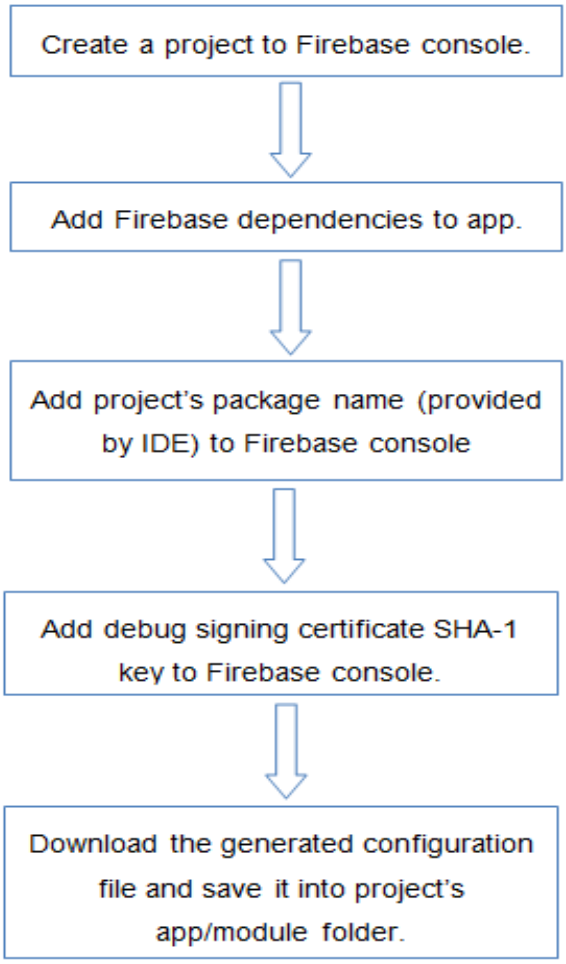


Figure 13: Database Integration to Firebase

3.1.2.3 Implementation of communication between Firebase Database and Geneus

After the integration of the database in the Firebase console, the next step is to implement the communication between Geneus and Firebase Database. [Figure 14] For this purpose, the Firebase Database libraries should be included in the application's Gradle file. Then, a reference to the Real-time Database is essential to be created in order for the option of setting, reading, or writing data to Geneus database to be possible. Therefore, the database reference is responsible for "listening" to data changes. Moreover, the offline mode is important to be enabled in order for Geneus to be responsive even when the user is offline. Finally, the data were secured by using the Firebase Real-time Database Security Rules as explained in Chapter 2.3.4.

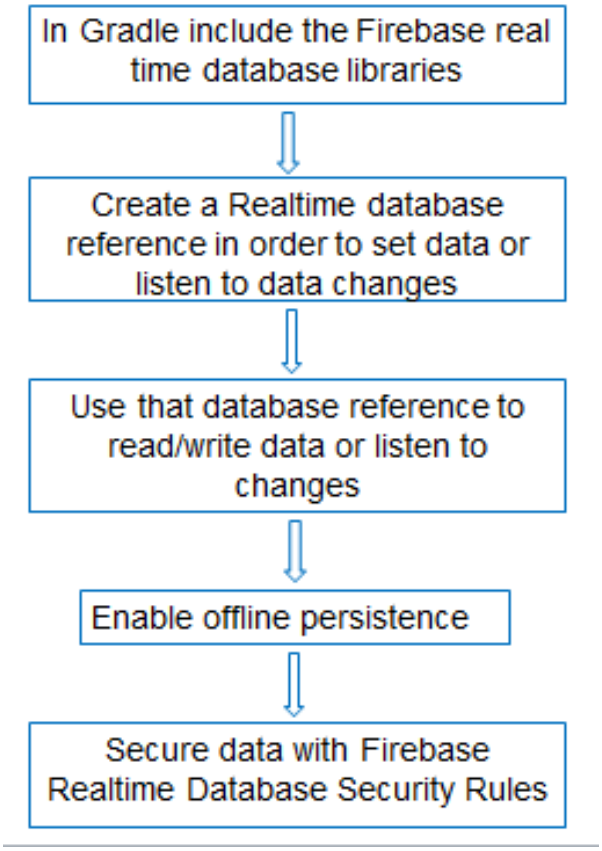


Figure 14: Implementation workflow of communication between Firebase Database and Geneus

3.1.2.4 Read Data from Firebase Database

Geneus has to read and retrieve data from Firebase Database in order to provide the users with the details of the human genes in the Search Genes Activity. The process of reading data from Firebase Database requires four steps. First, a model Java class that stores and manages the gene details was created. The model class contains the same fields included in a JSON object of Geneus Firebase database. The name compatibility between the JSON file and the model class is very important to be followed. Therefore, the name of each field in model class must be exactly the same with the name of the fields of a JSON object of Geneus Firebase database. If there is not name compatibility between the JSON file and the model class, during the reading process of genes from the Firebase database the fields of details included to a JSON node will not be matched to the fields of the model class. Therefore, the reading process does not match the first field of a JSON node to the first field of the model class, but the names of the fields are used for the matching process of the fields in order for an object of the model class to

be created and finally to store gene's details. The next step of Figure 15 requires the creation of a ListView adapter and its customization. In the ListView adapter is defined how genes' data will be displayed into the Geneus activity screen. For the reading process, a listener is essential to be added to the Firebase database reference for "listening" to any data change during this process. Finally, the data that was retrieved are stored to an ArrayList Object, and whenever a gene's details are required to be displayed into Geneus Search activity screen, the details are retrieved from that ArrayList object.

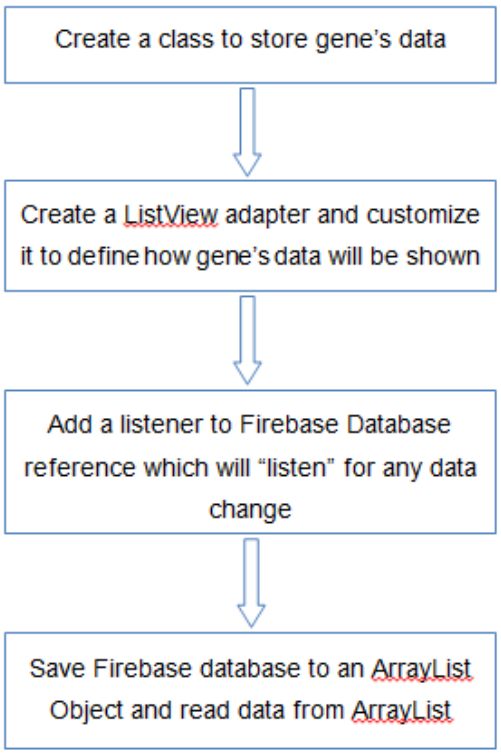


Figure 15: Implementation of Reading Data from Firebase

3.1.2.5 Write data to Firebase Database

Geneus writes data into Firebase Database because of the Community Messenger activity, and the data is written to the messages Firebase database. For the writing process, the creation of a model Java class that will store and manage message's details is required, the same as in the reading process. The next step of Figure 16 requires the creation of a ListView adapter and its customization for defining how message's details will be displayed into the Geneus activity screen. Moreover, a Fab button was created, which is like a send button. When the button is pressed, the messages are pushed to Firebase database and stored in the message's database. Finally, a function that displays all the stored messages from the message's Firebase database was implemented.

The user through Geneus Community Messenger activity can view all the history of messages already sent and the new messages. In addition, in case a user is offline and types a message when the sent button is pressed the message is stored locally to the database and when the connection is reestablished the message is sent to all the other connected users and stored to Firebase messages database. Then, the app receives the appropriate set of events so that the client user is synchronized with the current server state. Thus, the app remains responsive regardless of network latency or connectivity.

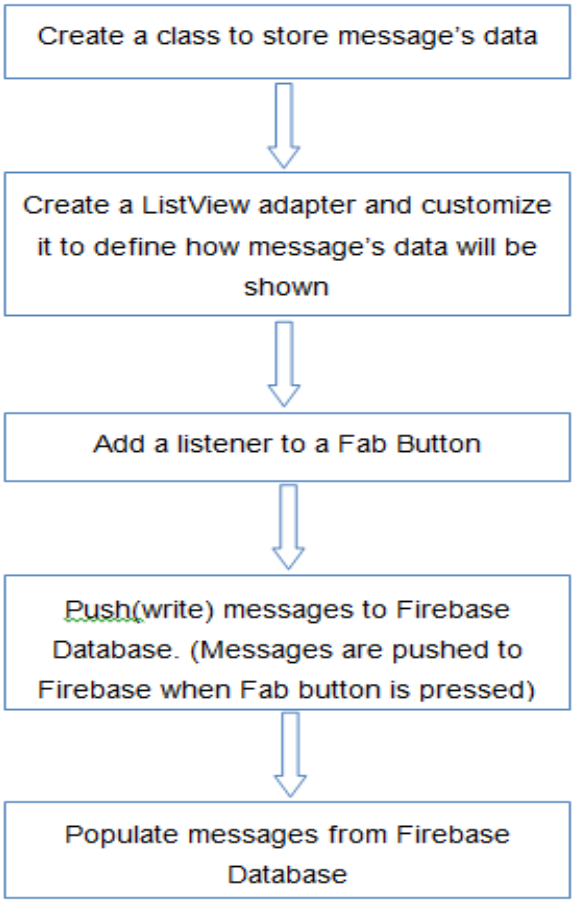


Figure 16: Implementation of writing data to Firebase

3.1.2.6 Firebase Google Authentication

Authentication is a very important step for Geneus application because by authenticating users, Geneus securely saves user's data in the cloud and it prevents the database from compromising by an attacker. To achieve this, the Firebase console requires sign-in methods to be enabled. Furthermore, in the Gradle file of Geneus Authentication must be declared authentication dependencies. Finally, the sign-in function in Geneus code project was implemented. [Figure 17]

The authentication process identifies users before they can access certain resources in an application such as a database. Geneus authenticates its users with a Gmail account. User identification is an important security concept because each user has their data and capabilities in an application. Therefore, the Community Messenger activity of Geneus along with the message text displays the username of the sender user. Moreover, authentication is a favorable practice to be followed in order to prevent and control public access to the application's resources.

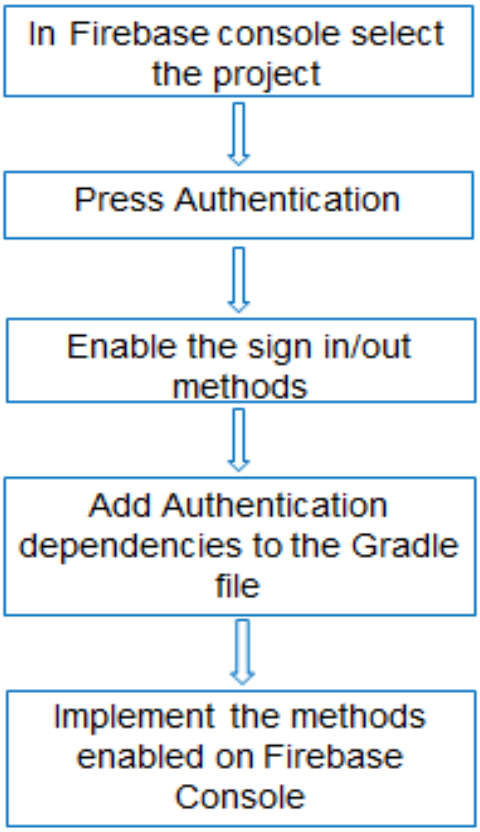


Figure 17: Firebase Authentication Steps

3.1.2.7 Firebase Real-time Database Crashes

During the Geneus implementation, several issues were encountered, which resulted to gain in-depth knowledge of Firebase features and fully comprehend its documentation in all aspects of Firebase integration methods. Two aspects that finally were overcame, critical for a developer using Firebase Real-time Database, are emphasized:

1) Name Compatibility

The problem was generated because of incorrect mapping of the field names of the database JSON file against the fields of the model class that store the information of each gene.

Therefore, the field names of the model class should be the same as the field names of the JSON file.

2) Exception using setPersistence mode

Function setPersistenceEnabled should be called only once (before any other instance of FirebaseDatabase is used) inside the activity that performs the connection between the Firebase Database and the application. Initially, before calling the setPersistenceEnabled method, it should be checked if the database reference is null. In case it is null, the data persistence during offline mode can be enabled.

3.2 Geneus Real-time Database Advantages

The Geneus database having been implemented by exploiting Real-time Database of Firebase bears many advantages. Because of Firebase, the database is cloud-hosted in Firebase servers, so there is no need for implementing an application server or creating an API to perform communication between the database stored in Firebase and the application. The genes' and messages' data are synchronized in real-time to every connected user. Therefore, all Geneus users share one Real-time database instance and automatically receive updates with the newest data when any data were changed. Consequently, there is no need for downloading a new version of Geneus from the Google Play Store when an update is performed to the data stored in the Firebase database.

Moreover, Geneus data included in the genes database and the messages database remain accessible even when the user is offline because data are cached to the device's memory. Once connectivity is reestablished, the client user receives any changes that were missed, and Geneus data is synchronized with the current state of data stored in the Firebase database. In addition, Geneus database is more than capable of handling the Real-time data updates between many devices because of Firebase Database technology.

Finally, another advantage of using the Firebase Real-time Database framework is its code portability. Geneus could be used as a general mobile app framework for any other patient database entries (e.g., daily blood sugar data) by modifying only two files of Geneus application code project: First, the Java model class that contains fields included in the JSON file must be modified. In the new model class, the fields of the new JSON database file have to be defined. Finally, the list adapter of Geneus app could be modified if another layout is more appropriate for the new database entries. The layout defines how the data will be displayed in an activity.

3.3 Geneus: A real-time gene app

Geneus [Figure 18] is an Android application that offers an online human genes database, and authenticated users can have access to 10.000 human gene profiles. Additionally, Geneus provides a Community Messenger, which is a group chat for the authenticated users to interact with each other and share useful knowledge.

Geneus potential users could be Experts, Doctors, Patients or any user who is interested in gene information.

Through Geneus, experts such as Biologists, enjoy straightforward access to real-time gathered information such as gene details from their smartphone by avoiding the process of searching and downloading a database of genes, which may contain already obsolete information. By using Geneus, experts may not miss any data updates because new information is received automatically.

Within Geneus, patients may obtain gene information within a simple implementation and can communicate with a doctor or other patients-users through the Community Messenger activity of Geneus. In the future, Geneus could be incorporated into the software of medical centers as a service to patients. Therefore, the app could contain the personal sequencing data per each patient or medical examination results and receive real-time updates about them. Furthermore, in future implementations patients may communicate privately with their doctor about medical issues. Moreover, doctors through Geneus can communicate with their patients directly and offer medical advice. Finally, the doctors will be able to provide patients with personal medical examination data results as well as real-time updates about them. Obviously, this implementation should fully comply with the current GDPR standards and regulations and a dedicated Data Protection Officer should be staffed.

Figure 18: Geneus Logo

3.3.1 Competitive landscape

Geneus is ahead of other competitors in features and functionality. None of the existing gene database apps utilize a real-time database neither do they contain a group chat for providing a communication tool among the app users. The most important difference of Geneus compared to other apps is that Geneus database is real-time and there is no need for downloading a new version of the app from Google Play store due to the fact that every change or addition related to the database is automatically synchronized to all online users. The following table demonstrates the advantages of Geneus compared to other apps.

Table 2: Competitive apps

App Name	Number of Downloads	Real-time Database	Group Chat
Geneus	Currently in Beta	✓	✓
List of Human Genes	1,000+	NO	NO
Genes	1,000+	NO	NO
Human Genes	10,000+	NO	NO

Some competitive apps of Geneus found in Google Play store are List of Human Genes app, Genes app, and Human Genes app. List of Human Genes application contains a list of human genes classified by chromosomes.[22] Geneus application presents a collection of genes and their functions illustrated with ontology terms.[23] Human Genes contains an appendix with the description of human genes.[24]

3.4 Geneus Activities

The development of Geneus was completed by implementing four activities; Sign in Activity, Main Activity, Search Activity, Gene's Profile Activity, and Community Messenger Activity. These activities are explained in detail in the next Chapters.

3.4.1 Geneus Sign in Activity

The user management of Geneus app is implemented by authenticating users with a Gmail account. Only the logged in and verified users have access to the app's database. [Figure 19-20]

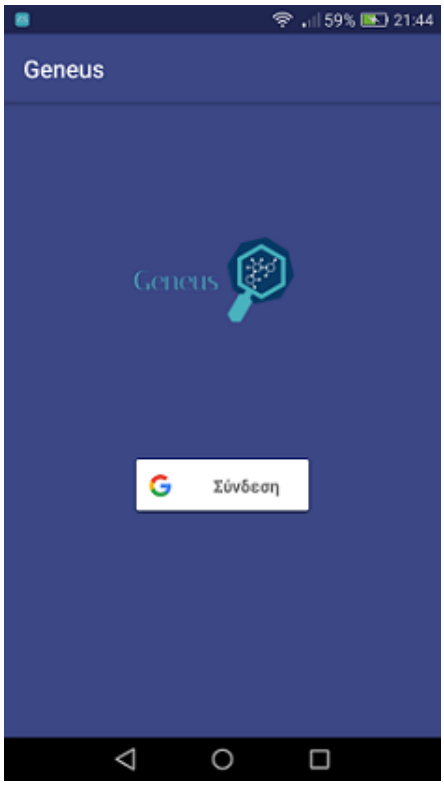


Figure 19: SignIn Activity

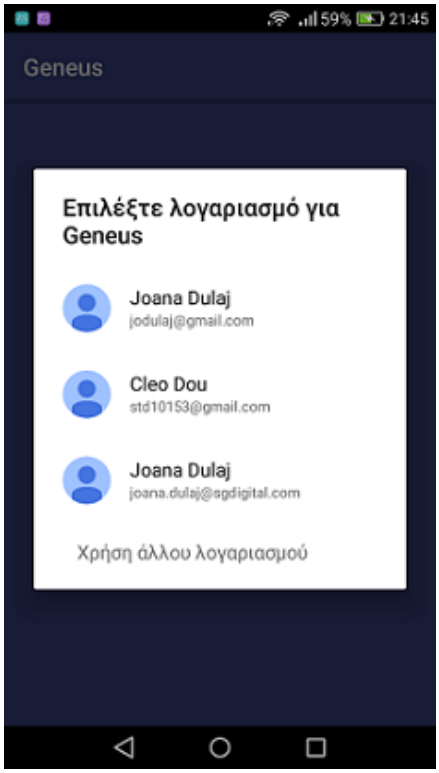


Figure 20: SignIn Activity(2)

3.4.2 Geneus Main Activity

The main activity consists of 3 buttons [Figure 21]:

- Search Genes
- Community Messenger
- Google Sign Out

Figure 21: Main Activity

3.4.3 Geneus Search Activity

Geneus Search Activity displays a list of 10.000 gene profiles, and all users share one Real-time Database instance. Because of Firebase database, the users automatically receive updates with the newest data. Gene's list displays the name of the gene and all the list items are clickable. [Figure 22] The loading of the genes in this activity takes 8-10 seconds because in the background a check is performed for any data change.

Figure 22: Geneus Search Activity

3.4.4 Gene's Profile

Geneus Profile Activity displays the details of the selected in the Search Activity gene's name. The fields of the list are clickable and expandable. [Figures 23-24]

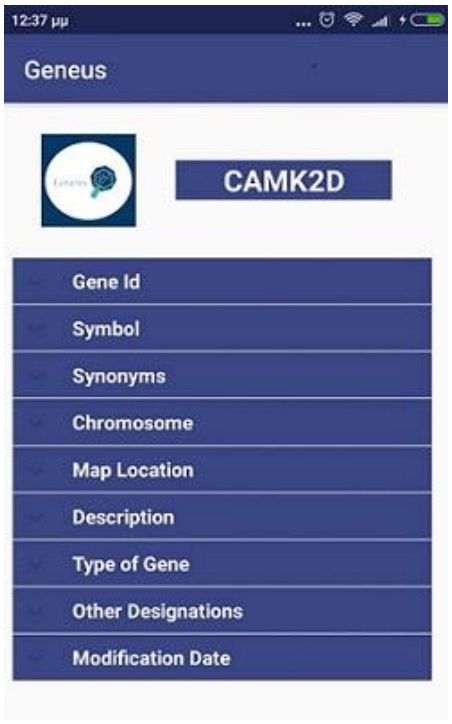


Figure 23: Gene's Profile Activity

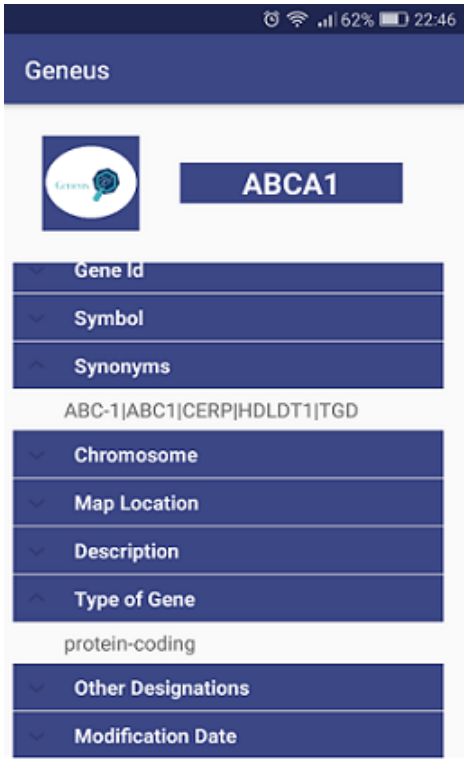


Figure 24: Gene's Profile Activity(2)

3.4.5 Community Messenger

Geneus Community Messenger (group chat) activity provides communication among all app users. The messages remain accessible even when the user is offline because they are cached to the device's memory, and when connectivity is reestablished, the user receives any message that was missed. Bubbles displayed in this activity provide information about the username, the text and the timestamp of the sent message. [Figure 25]

Figure 25: Community Messenger

CONCLUSIONS

Healthcare applications using the Android Operating System are being increasingly adopted by doctors and patients. Accessing patient data in a real-time fashion through a mobile application would be extremely time- and cost- saving as doctors would have the opportunity to monitor patients remotely. Moreover, it is vital for patients to access their personal medical information real time. This rationale was the driving force for developing the Geneus application, an Android application incorporating a real-time gene database, developed in a general framework able to host any type of medical information, such as daily blood sugar data.

E-health is being increasingly incorporated in patient care incorporating various applications worldwide. E-health is the future of health practice and, sooner or later, it will become a mandatory component in health systems worldwide. Advances in this field may bring groundbreaking leaps, evidenced by the fact that numerous applications are launched and used every year by patients, doctors, experts, and the public.

Providing real-time access to gene databases from a smartphone is expected to enhance their usability and access as we move towards new technological advances such as targeted next-generation DNA sequencing. Developing mobile apps with real-time database may be deemed convenient for genome sequencing analysis because of the real-time access to rapidly changing information. Moreover, patients could access their genome analysis from a smartphone and receive updates when new data appears.

In this study, we have exploited the Android Google Firebase, which provides data synchronization for every connected user/client who receives gene data changes within milliseconds. Moreover, through Firebase SDKs, data is available even when the user is offline because Firebase Real-time Database persists data on the device and data are synced with the cloud when the device reconnects to the Internet. Authentication is related to the Sign In process for Apps by authenticating a user, before they can access certain resources in an application such as a database. User authentication is an important security concept because each user can store their data securely in a mobile app.

The Geneus database contains information about 10.000 human genes downloaded from NCBI. With Geneus, authenticated users enjoy straightforward access to a real-time database of genomic profiling integrated into an Android application with specific information per each gene included in the database, and to a Community Messenger activity for communication with the other users. Furthermore, Geneus could be used as a general mobile app framework with any other patient database entries (e.g., daily blood sugar data). Geneus innovation is related to the use of a real-time database in order for users to receive automatically updates with the newest data with no need for downloading a new version of the app from Google Play Store.

Geneus potential users could be Experts, Doctors, Patients or any typical user that could be interested in gene information. Through Geneus, experts such as Biologists, enjoy straightforward access to real-time gathered information such as gene details from their smartphone by avoiding the process of searching and downloading a database of genes, which may contain already obsolete information. By using Geneus, experts may not miss any data updates because new information is received automatically. Within Geneus, patients may obtain gene information within a simple implementation and can communicate with a doctor or other patients-users through the Community Messenger activity of Geneus. Finally, doctors through Geneus can communicate with their patients directly and offer medical advice.

FUTURE PERSPECTIVES

Currently, the Geneus app provides to Google authenticated users access to a real-time database of genomic profiling with specific information for 10.000 human genes downloaded from NCBI. Future implementations could include a web crawler for developing a new database with more data. Web crawling is a method for collecting data and keeping up to date with the rapidly expanding Internet, as it automatically traverses the web by downloading documents and following links from page to page. It is a tool for the search engines and other information seekers to gather data for indexing and to enable them to retain their databases up to date.

Currently within Geneus the patients may obtain gene information within a simple implementation and can communicate with a doctor or other patients-users through the Community Messenger activity of Geneus. However, in the future, Geneus could be incorporated into the software of medical centers as a service to patients. Therefore, the app could contain the personal sequencing data per each patient or medical examination results and receive real-time updates about them. Furthermore, in future implementations patients may communicate privately with their doctor about medical issues. Moreover, doctors through Geneus can communicate with their patients directly and offer medical advice. Finally, the doctors will be able to provide patients with personal medical examination data results as well as real-time updates about them. Obviously, this implementation should fully comply with the current GDPR standards and regulations and a dedicated Data Protection Officer should be staffed

At the moment Geneus provides communication to all the users through a group chat activity. In later editions, this activity could be implemented as a messenger philosophy activity that will provide private communication between users.

In Geneus only the logged in and verified users have access to the app in order to be prevented the public access to the database. At present, Geneus authenticates users only with Google accounts but is possible for a user that downloads the app not to have a Google account. Therefore, in that case, the user cannot explore apps functionalities because it is mandatory for a user to first sign in with a Google account before access app functionalities. Thus, it is important more authentication methods (using Facebook, Twitter or Github account) to be implemented for user authentication in order to avoid cases when a user downloads Geneus and cannot access it because they do not have a Google account to sign in.

Another functionality that could be added in Geneus is share buttons in order for users to share data via social media.

Finally, uploading and sharing files or images could be added to messenger functionalities in order for users to have more options during their communication with other users because now they can only communicate by sending messages.

ABBREVIATIONS - ACRONYMS

SDK	Software Development Kit
IDE	Integrated Development Environments
JSON	JavaScript Object Notation
OS	Operating System
DB	Database
DVM/JVM	Dalvik Virtual Machine/ Java Virtual Machine
.dex	dalvic executable (file)
app	Application
CRUD	Create, Read, Update and Delete

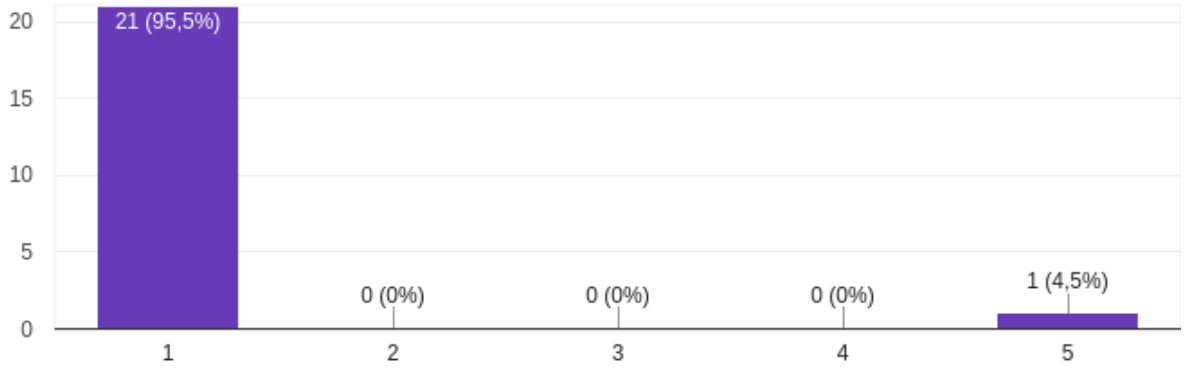
ANNEX I

Geneus App UI/UX questionnaire

Twenty two typical users, of age 25-30, participated in a Geneus UI/UX questionnaire in order to record their experience with Geneus.

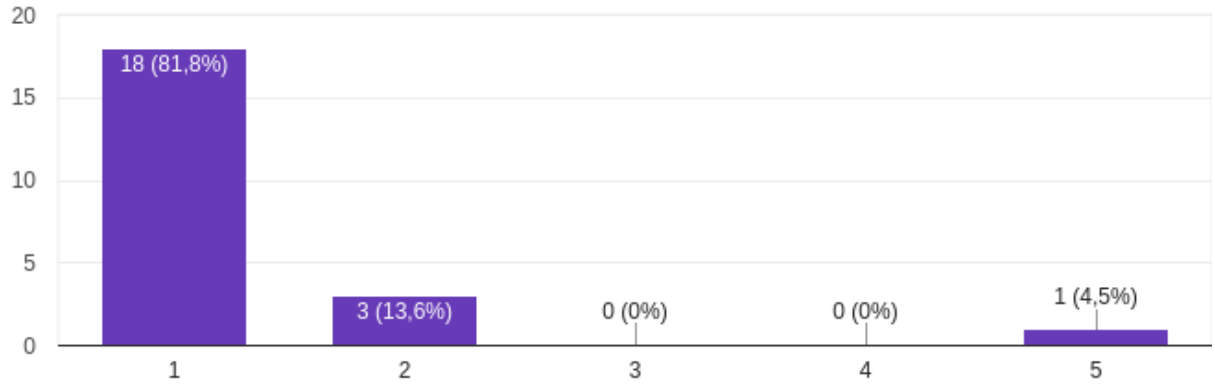
As it can be noticed in the following diagrams, Geneus is a user-friendly application without performance issues. The response time of the database is extremely high, and the “Search Genes” functionality was voted as the most important feature of the app. Finally, most of the participants are extremely likely to recommend the app to a friend or colleague and proposed a few features to be added in the future.

Diagram 1: How easy was it to install Geneus Application?



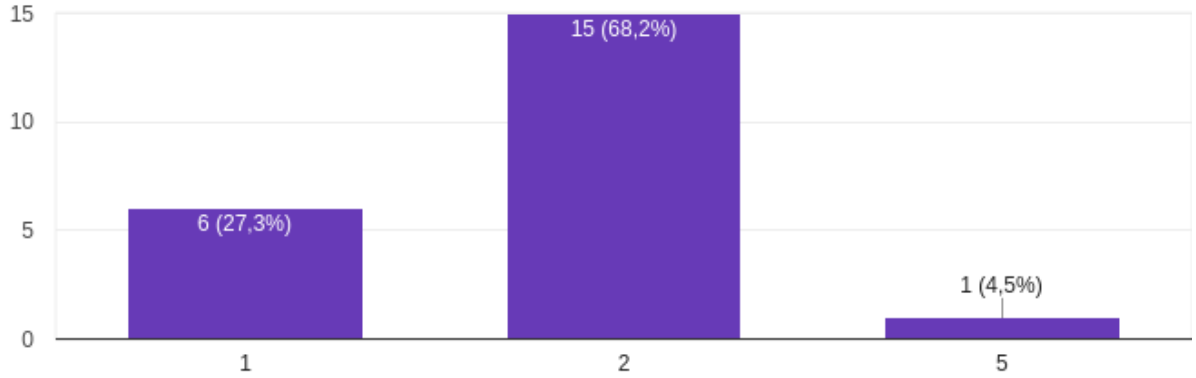
95.5% of the participants reported that Geneus was easy to install and only one participant (4.5%) reported difficult in installation [Diagram 1].

Diagram 2: How easy was it to sign in with your Gmail Account?



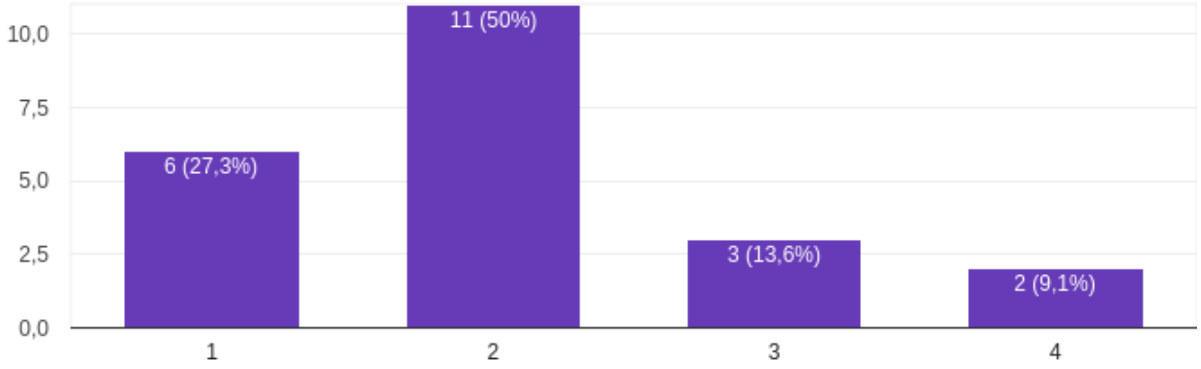
81.8% of the participants voted that was very easy to log in with a Gmail account in Geneus. 13.6% (3 of 22 users) of the participants voted that is just easy and for 4.5% (1 user) of the participants it was difficult to sign in in Geneus with a Gmail account [Diagram 2].

Diagram 3: **How many clicks you performed to find NATP gene**



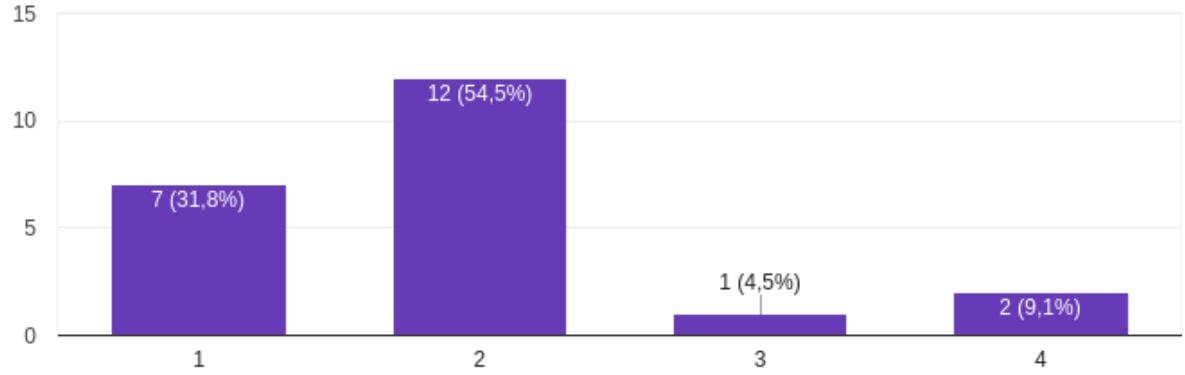
68.2% of the participants voted that with 2 clicks they found a gene in Geneus, 2.3% of the participants found NATP gene with only 1 click and 4.5% of participants found it with 5 clicks [Diagram 3].

Diagram 4: How many clicks you performed to NATP details



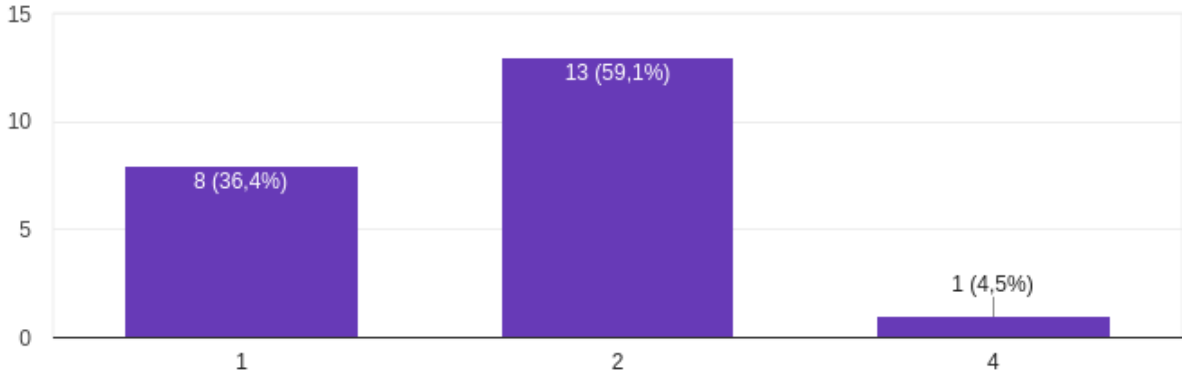
27.3% of the participants voted that with 1 click they had access to NATP details, 50% of the participants had access with 2 clicks, 13.6% with 3 clicks and 9.1% performed it with 4 clicks [Diagram 4].

Diagram 5: How many clicks you performed to find "Map Location" of NATP gene



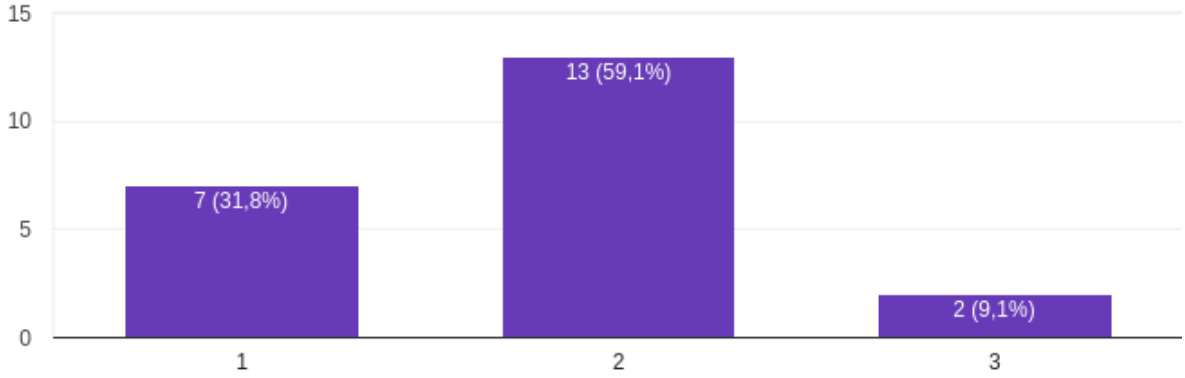
31.8% of the participants voted that with only 1 click it was achieved the access to a specific field of information of NATP gene , 54.5% of the participants voted that with 2 clicks they performed it, 4.5% performed it with 3 clicks and 9.1% with 4 clicks [Diagram 5].

Diagram 6: How many clicks you performed to find community messenger of Geneus app



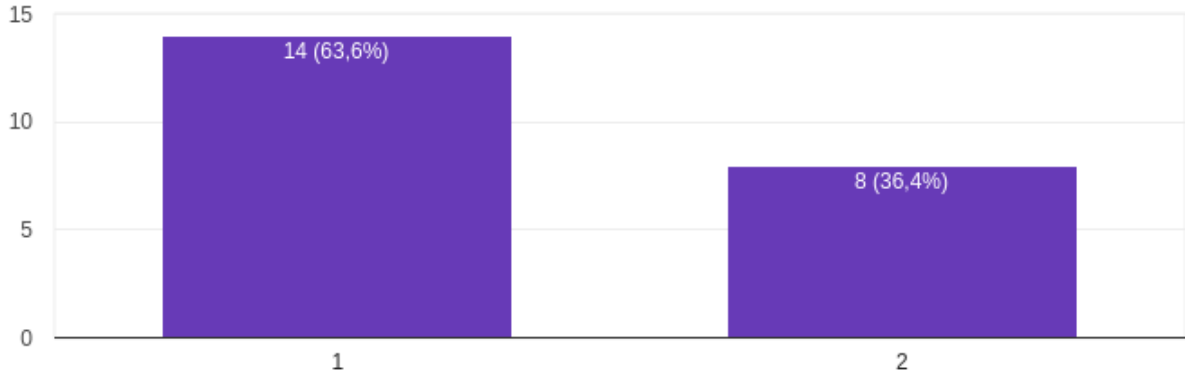
36.4% of the participants voted that with 1 click they had access to Geneus chat box , 59.1% of the participants had access with 2 clicks and 4.5% with 4 clicks [Diagram 6].

Diagram 7: How many clicks you performed to send a message over community messenger of Geneus



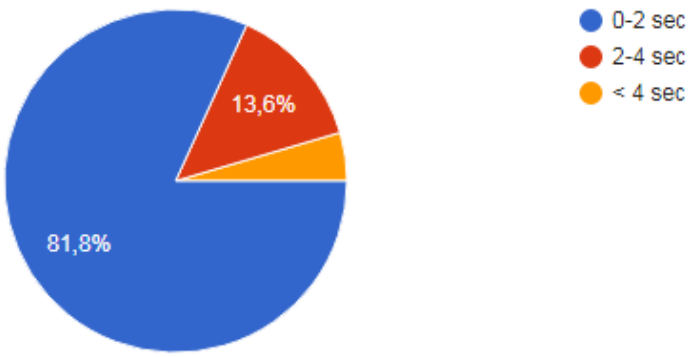
31.8% of the participants voted that with 1 click they sent a message through Geneus chat box, 59.1% of the participants sent it with 2 clicks and 2.9% with 3 clicks [Diagram 7].

Diagram 8: How many clicks you performed to Sign Out from Geneus app



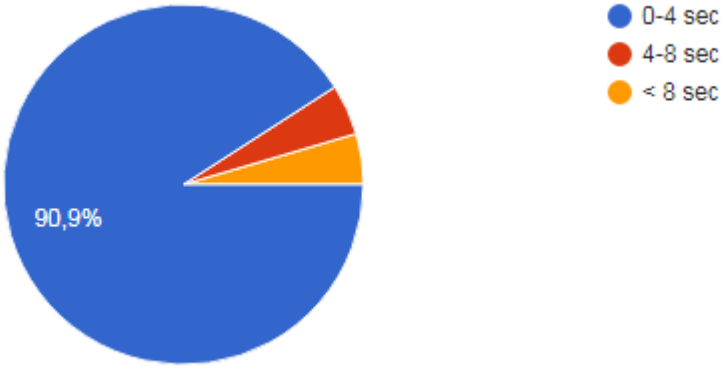
63.6% of the participants voted that with 1 click they signed out from Geneus and 36.4% performed to sign out with 2 clicks [Diagram 8].

Diagram 9: What is the time response of the search results for a gene



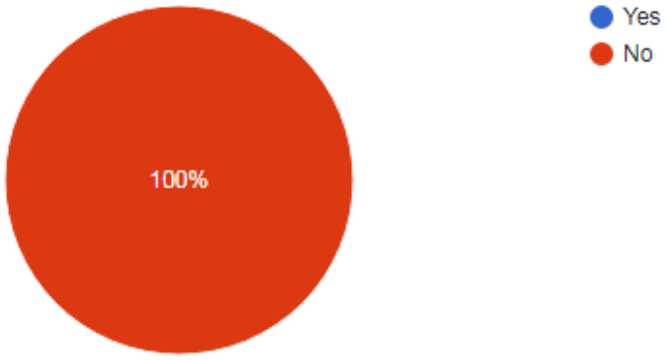
81.8% of the participants voted that the time response of search results for a gene is 0-2 seconds, 13.6% voted that 2-4 seconds took for the results of a gene and 4.6% reported that it took up to 4 seconds [Diagram 9].

Diagram 10: What is the time response of loading the entire database



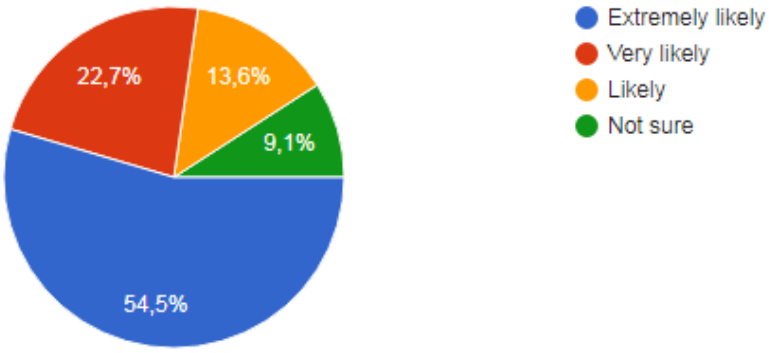
90.9% of the participants reported that the time response of the entire database loads in 0-4 seconds, for 4.55% it takes 4-8 seconds and for 4.55% it takes up to 8 seconds [Diagram 10].

Diagram 11: Do you experience any performance issue during scrolling down and up in Search Genes Activity



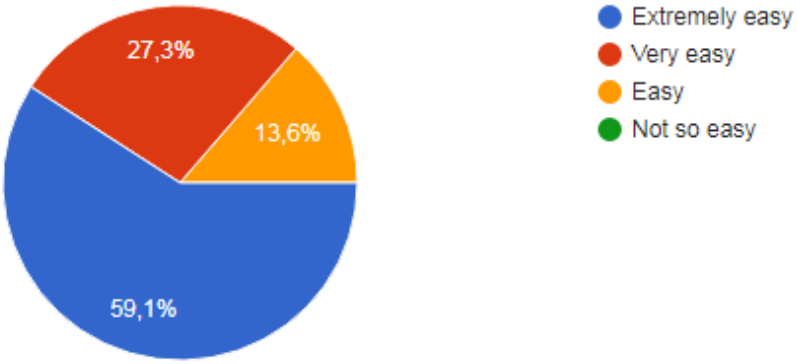
100% of the participants reported that they did not experience any performance issue during scrolling down/up genes list [Diagram 11].

Diagram 12: How likely is to recommend Geneus to a friend or a colleague



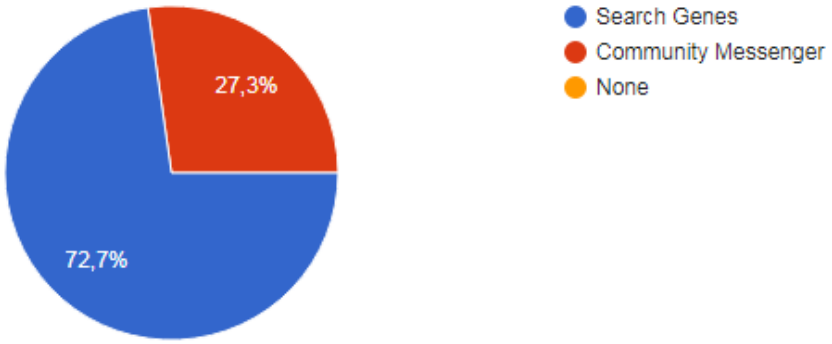
54.5% of the participants voted that it is extremely likely to recommend Geneus to a friend or colleague, 22.7% of the participants reported that it is very likely to recommend it and 9.1% that they are not sure if they will recommend it [Diagram 12].

Diagram 13: How user-friendly is Geneus



59.1% of the participants voted that Geneus is extremely user-friendly, 27.3% reported that is very user-friendly, and 13.6% that is just user-friendly [Diagram 13].

Diagram 14: Which feature of Geneus id the most important



72.7% of the participants reported that Search Genes is the most important feature of Geneus and 27.3% voted Community Messenger as the most important [Diagram 14].

Diagram 15: What features we should add in Geneus app

Sign in with Facebook etc and private messages
Share buttons.
Please insert some images
Looks complete
You can add favorites section in search mode
Please add more login authenication modes (for example facebook)
Add interactive mode to measure our vital signs for example temperature
Animation to buttons
upload files to messenger chatbox
history of search
share buttons
some more videos for educational purpose

Participants proposed many features to be added in Geneus like the sign in with a Facebook account, option to send private messages, animation buttons, history of search, etc [Diagram 15].

REFERENCES

[1] Melissa McCormack. (2018, September) https://www.softwareadvice.com. [Online]. https://www.softwareadvice.com/resources/what-is-big-data-in-healthcare-and-whos-already-doing-it

[2] Fabricio F. Costa, "Big data in biomedicine," Elsevier, vol. 00, pp. 433–440, November 2013.

[3] Schram AM, Reales D, Galle J, Cambria R, Durany R, Feldman D, Sherman E, Rosenberg J, D'Andrea G, Baxi S, Janjigian Y, Tap W, Dickler M, Baselga J, Taylor BS, Chakravarty D, Gao J, Schultz N, Solit DB, Berger MF, Hyman DM, *"Oncologist use and perception of large panel next-generation tumor sequencing.",* Ann Oncology, 2017 Sep 1

[4] Cinnamon S. Bloss, Dilip V. Jeste, and Nicholas J. Schork, *"Genomics for Disease Treatment and Prevention,"* Psychiatric Clinics of North America, pp. 147–166. , March 2011

[5] Guy Edwards, Tim Wilsdon Anthony Barron, *"The benefits of personalised to patients, society and healthcare systems,"* Charles River Associates, London, Evidence-based analysis for PM, July 2018.

[6] (2018, September) https://www.ncbi.nlm.nih.gov. [Online]. https://www.ncbi.nlm.nih.gov

[7] (2018, September) https://www.ddbj.nig.ac.jp/index-e.html. [Online]. https://www.ddbj.nig.ac.jp/index-e.html

[8] (2018, September) https://www.ensembl.org/index.html. [Online]. https://www.ensembl.org/index.html

[9] (2018, September) https://www.statista.com/statistics. [Online]. https://www.statista.com/statistics/266136/global-market-share-held-by-smartphone-operating-systems/

[10] (2018, January) http://androiddeveloper.galileo.edu. [Online]. http://androiddeveloper.galileo.edu/2017/03/31/android-an-open-source-code-platform/

[11] (2018, June) https://blog.trigent.com. [Online]. https://blog.trigent.com/five-of-the-most-popular-databases-for-mobile-apps/

[12] Karen Taylor, *"How digital technology is transforming health and social care,"* Deloitte Centre for Health Solutions, London, 2015.

[13] (2018, July) https://www.cancer.gov. [Online]. https://www.cancer.gov/publications/dictionaries/cancer-terms/def/genomic-profiling

[14] (2018, September) http://www.justscience.in. [Online]. http://www.justscience.in/articles/pros-cons-individual-genome-profiling/2017/12/09

[15] Karin Jegalian, "Genetics The Future of Medicine," *National Human Genome Research Institute*, pp. 1-16, 2003.

[16] KÄÄRIÄINEN, Helena, et al, "*Genetics in an isolated population like Finland: a different basis for genomic medicine?*," Journal of community genetics, pp. 319-326, 2017

[17] B. Singh, and M. Kaur, *"IT applications in healthcare.,"* Mumbai, Maharashtra, India, February 26 - 27, pp.170-172, 2010

[18] WORLD HEALTH ORGANIZATION, et al., "*From innovation to implementation—eHealth in the WHO European region.",* World Health Organization: Geneva, Switzerland, 2016.

[19] (2018, September) https://www.statista.com. [Online]. https://www.statista.com/statistics/387867/value-of-worldwide-digital-health-market-forecast-by-segment/

[20] Samyuktha, G. Geethakumari, and C. S. N. Prasad Challa, *"Patient data viewer: an Android application for healthcare.,"* India Conference (INDICON), 2011 Annual IEEE. IEEE, pp. 1-4, 2011

[21] Thiery Phommavongsay, Aries E. Aisporna, Tao Huan, Duane Rinehart, Jose Rafael Montenegro Burke, *"Smartphone Analytics: Expanding the Lab into the Cloud," Analytical Chemistry*, pp. 16, August 2016.

[22] (2018, January) https://play.google.com/store [Online]. https://play.google.com/store/apps/details?id=com.human.genom

[23] (2018, January) https://play.google.com/store [Online]. https://play.google.com/store/apps/details?id=per.koszut.beata.biology.genes

[24] (2018, January) https://play.google.com/store [Online]. https://play.google.com/store/apps/details?id=com.do_apps.catalog_130

[25] (2018, January) https://www.unixmen.com [Online]. https://www.unixmen.com/the-history-of-android/

[26] Kirandeep and Anu Garg, *"Implementing Security on Android Application,"* The International Journal Of Engineering And Science (IJES), vol. 2, pp. 56-59, March 2013.

[27] Frank Sposaro and Gary Tyson, *"iFall: An android application for fall monitoring and response",* 31st Annual International Conference of the IEEE Engineering in Medicine and Biology Society, pp. 6119–6122, 2009.

[28] Frank Sposaro, Justin Danielson and Gary Tyson, *"iWander: An Android application for dementia patients.,"* Engineering in Medicine and Biology Society (EMBC), 2010 annual international conference of the IEEE. IEEE,, pp. 3875-3878, 2010

[29] (2018, January) https://www.androidauthority.com. [Online]. https://www.androidauthority.com/develop-android-apps-languages-learn-391008/
[30] (2018, January) https://www.javatpoint.com. [Online]. https://www.javatpoint.com/dalvik-virtual-machine
[31] (2018, January) http://www.tutorialride.com. [Online]. http://www.tutorialride.com/android/architecture-of-android.htm
[32] (2018, January) https://developer.android.com [Online]. https://developer.android.com/guide/components/fundamentals
[33] (2018, January) https://developer.android.com [Online]. http://developer.android.com/guide/topics/fundamentals.html
[34] (2018, January) https://developer.android.com [Online]. https://developer.android.com/guide/topics/manifest/manifest-intro
[35] (2018, May) https://developer.android.com [Online]. https://developer.android.com/guide/topics/resources/providing-resources
[36] (2018, May) https://www.techopedia.com [Online]. https://www.techopedia.com/definition/4220/android-sdk
[37] (2018, January) https://developer.android.com [Online]. https://developer.android.com/studio/build/
[38] (2018, January) http://www.tech-soigne.com/ [Online]. http://www.tech-soigne.com/sqlite-database-use-in-android-app-tutorial/
[39] (2018, January) https://developer.android.com/ [Online]. https://developer.android.com/reference/android/database/sqlite/SQLiteOpenHelper
[40] (2018, January) https://www.sqlitetutorial.net/. [Online]. http://www.sqlitetutorial.net/sqlite-data-types/
[41] (2018, January) https://howtofirebase.com/. [Online]. https://howtofirebase.com/what-is-firebase-fcb8614ba442
[42] (2018, January) https://www.raizlabs.com/. [Online]. https://www.raizlabs.com/dev/2016/12/firebase-case-study/
[43] (2018, January) https://firebase.google.com/ [Online]. https://firebase.google.com/docs/database/
[44] (2018, February) https://developers.googleblog.com/. [Online]. https://developers.googleblog.com/2015/11/add-backend-logic-to-real-time-data.html
[45]]Nguyen, Phi-Vu, Chandra Shekhar Verma, and Samuel Ken-En Gan, *"DNAApp: a mobile application for sequencing data analysis,"* Oxford University Press, pp. 3270-3271, August 2014.
[46] (2018, February) https://firebase.google.com/. [Online]. https://firebase.google.com/docs/database/security/securing-data
[47] (2018, January) http://www.tech-soigne.com/. [Online]. http://www.tech-soigne.com/sqlite-database-use-in-android-app-tutorial/
[48] (2018, January) https://www.ncbi.nlm.nih.gov/. [Online]. https://www.ncbi.nlm.nih.gov/gene/?term=human
[49] (2018, September) https://en.wikipedia.org/. [Online]. https://en.wikipedia.org/wiki/National_Center_for_Biotechnology_Information