MINI PROJECT

(2020-21)

"MEDICAL APP"

Project Report

Institute of Engineering & Technology

Submitted By -

Bhaskar Kumar (191500217)

Ganesh Pandey (191500285)

Ritu Kushwah (191500665)

Sneha Gautam (191500814)

Under the Supervision Of

Mr. Kunal Goyal

Technical Trainer

Department of Computer Engineering & Applications

Department of Computer Engineering and Applications

GLA University, 17 km. Stone NH#2, Mathura-Delhi Road,

Chaumuha, Mathura – 281406 U.P (India)

Declaration

I/we hereby declare that the work which is being presented in the Bachelor of technology. Project **"Medical App"**, in partial fulfilment of the requirements for the award of the ***Bachelor of Technology*** in Computer Science and Engineering and submitted to the Department of Computer Engineering and Applications of GLA University, Mathura, is an authentic record of my/our own work carried under the supervision of **Mr. Kunal Goyal, Technical Trainer, Dept. of CEA, GLA University.**

The contents of this project report, in full or in parts, have not been submitted to any other Institute or University for the award of any degree.

Sign: Bhaskar
Name of Candidate: Bhaskar Kumar
University Roll No.:191500217

Sign: Ritu
Name of Candidate: Ritu kushwah
University Roll No.:191500665

Sign: Ganesh
Name of Candidate: Ganesh Pandey
University Roll No.:191500285

Sign: Sneha
Name of Candidate: Sneha Gautam
University Roll No:191500814

Department of Computer Engineering and Applications

GLA University, 17 km. Stone NH#2, Mathura-Delhi Road,

Chaumuha, Mathura – 281406 U.P (India)

Certificate

This is to certify that the project entitled "Medical App", carried out in Mini Project – I Lab, is a Bonafide work by Bhaskar Kumar, Ganesh Pandey, Ritu Kushwah and Sneha Gautam and is submitted in partial fulfilment of the requirements for the award of the degree Bachelor of Technology (Computer Science & Engineering).

Signature of Supervisor:

Name of Supervisor: Mr. Kunal Goyal

Date:

Department of Computer Engineering and Applications
GLA University, 17 km. Stone NH#2, Mathura-Delhi Road,
Chaumuha, Mathura – 281406 U.P (India)

ACKNOWLEDGEMENT

Presenting the ascribed project paper report in this very simple and official form, we would like to place my deep gratitude to GLA University for providing us the instructor Mr Kunal Goyal, our technical trainer and supervisor.

He has been helping us since Day 1 in this project. He provided us with the roadmap, the basic guidelines explaining on how to work on the project. He has been conducting regular meeting to check the progress of the project and providing us with the resources related to the project. Without his help, we wouldn't have been able to complete this project.

And at last, but not the least we would like to thank our dear parents for helping us to grab this opportunity to get trained and also my colleagues who helped me find resources during the training.

Thanking You

Sign: *Bhaskar Kumar*
Name of Candidate: Bhaskar Kumar
University Roll No.:191500217

Sign: *Ritu Kushwah*
Name of Candidate: Ritu Kushwah
University Roll No.:191500665

Sign: *Ganesh Pandey*
Name of Candidate: Ganesh Pandey

Sign: *Sneha Gautam*
Name of Candidate: Sneha Gautam

University Roll No.:191500285

ABSTRACT

In this project, we are creating an android application, basically a Medical App which we have named MEDHELP. The purpose of the project entitled as "MEDICAL APP" is to computerize the Front Office Management of Medical Service to develop software which is user friendly simple, fast, and cost – effective. It deals with the collection of patient's information, diagnosis details, etc. Traditionally, it was done manually. The main function of the system is register and store patient details and doctor details and retrieve these details as and when required, and also to manipulate these details meaningfully System input contains patient details, diagnosis details, while system output is to get these details on to the screen. The Medical App can be entered using a username and password. It is accessible either by a doctor or patients. Only they can add data into the database. The data can be retrieved easily. The data are well protected for personal use and makes the data processing very fast.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER-1

INTRODUCTION

1.1 CONTEXT

This Android Application “Medhelp” has been submitted in partial fulfilment of the requirements for the award of the degree of Bachelor of Technology in Computer Science and Engineering at GLA University, Mathura supervised by Mr. Kunal Goyal This project has been completed approximately three months and has been executed in modules, meetings have been organised to check the progress of the work and for instructions and guidelines.

1.2 MOTIVATION

In the recent years, we have realized the importance of virtual learning and how important it is for us to have our resources online. The medical app allows you to attract new customers – the kind of customers who love technological innovations and adapt quickly to changing tools.

Medical mobile apps allow you to build a closer relationship with your customers by being able to proactively respond to their needs. Additionally, enabling medical services online is something that is now considered to be of great value.

Healthcare personel has less work – they can focus on other tasks by automating certain processes. New technologies allow reclaiming resources that are spent on repetitive, tedious tasks.

A medical app is a great way to get data about your customers in order to offer them even better, tailored products and services.

Consumers who use Medical mobile apps tend to be more loyal to the brand/product/service by being able to use a convenient, trustworthy tool.

Better corporate image – with the help of medical apps, you can easily let the world know that a particular brand cares about staying up to date with current trends.

Medical mobile apps save a lot of people who find it difficult to use traditional forms of healthcare. People with disabilities, limited mobility can easily access a doctor or appropriate specialist.

1.3 OBJECTIVE

The main objective of this application is to create a Medical App named “Medhelp” which

1) Define medical services

2) Recording information about the Patients that come.

3) Generating bills.

4) Recording information related to patients.

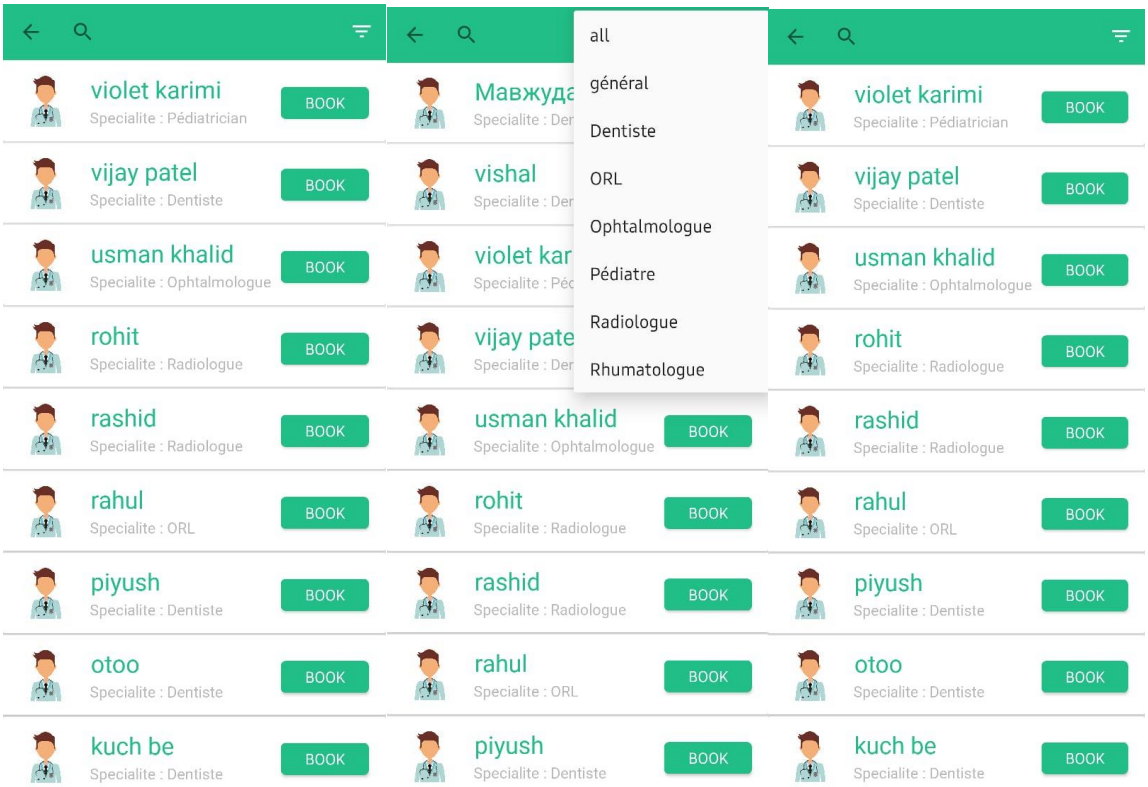
5) Keeping record of the services provided to children/patients.

6) Keeping information about various diseases and medicines available to cure them.

1.4 EXISTING SYSTEM

In the present scenario, we are dealing with the manual searching of doctors from thousands of other doctors as per the disease that a person suffers. With the help of this application, we are able to find a doctor where we can easily book the appointment with the help of search bar. As this idea as already implemented here are the some snap how our application will look.

As soon as the user enters the app, there will be landing page containing the name of the app and then there will be a login /signup page. Initially there will be search bar as shown in the image below. Then on the basis of this the app will fetch the results and the doctors will be displayed as shown in the screenshot. In this we will book an appointment and consult with doctor as this will be helpful to easily book their appointment.



(a) **(b)** **(c)**

Figure-1: Existing System

CHAPTER -2 SOFTWARE REQUIREMENT ANALYSIS

2.1 IMPACT OF MOBILE HEALTH APP ON DAILY LIFE

The difference between healthcare and other industries is that with healthcare, mobile technology can literally mean the difference between life and death. For example, a patient presents to the emergency room with signs of a stroke. Mobile health technology can provide an attending physician with access to a stroke specialist, enabling swift and accurate intervention. Or think of a mobile app that can tell an emergency room physician everything they need to know about a patient (e.g., previous surgeries, allergies, and more) when that person is unconscious. Again, life or death.

Mobile health technology can also bridge geographic barriers to connect a patient with cancer to a world-renowned oncologist located five hours away. It can help a physician monitor a patient's blood glucose daily without requiring in-office visits. It can empower a patient to lose weight and normalize their blood pressure or reverse their diabetes. The possibilities are truly endless.

The impact of mobile technology in healthcare can be felt throughout a patient's care journey. It starts with prevention. Consumer-facing mobile apps provide education and promote health awareness. Mobile health technology also helps providers diagnose and manage treatment even from a remote location. Finally, technology in healthcare puts patients in the driver's seat, enabling them to collect data and self-manage their health. Each of these steps in the journey are ripe with opportunities to integrate mobile technology to improve outcomes and reduce costs.

There are several reasons why mobile devices are transforming healthcare:

1. Healthcare consumerism

As in all industries, consumers want information and other conveniences (e.g., consumer reviews, online bill pay, and the ability to book appointments online) at their fingertips.

2. Shift to value-based care

Healthcare providers are looking for ways to drive down costs and make services more affordable. Under fee-for-service payment models, there are no financial incentives for providers to use this technology because outcomes and cost don't dictate payment. Now, with value-based payments, that's changing.

2.2 PROBLEM STATEMENT

Health systems in India are grappling with the effects of existing communicable and noncommunicable means to cater to the needs of the people. Lack of adequate progress on these underlying social determinants of health has been acknowledged as a glaring failure of public health.

Our main motto to take up this problem is to cater to each and every need of the patient as well as monitor his/her health from the time he/she first visits the hospital till he/she totally recovers.

Additional attention and adequate support to ensure faster recovery is not possible as there is no proper communication channel/link between the doctor and the patient after the check-up.

Due to insufficient data obtained in the form of feedback the government is unable to do the necessary improvements required in this field.

According to the statistics, feedback from merely 5-10% of the patients is collected monthly from any of the government aided medical centres.

The course of medication prescribed to patients is not monitored.

The staff is not monitored in terms of their behaviour and supportiveness towards the needs of the patients.

2.3 HARDWARE AND SOFTWARE REQUIREMENTS

Hardware Requirement

- Processor: intel i5
- Operating System: Any Operating System

- RAM: 8 GB (or higher)
- Hard disk: 256GB

Software Requirement

- Software used: Android Studio
- Language used: JAVA, XML
- Database: Firebase
- User Interface Design: Android Application

2.4 MODULES AND FUNCTIONALITIES

- **Splash Screen**: The first screen with which the user interacts will be this screen containing the logo and the app name. This will disappear within 5 seconds after the app is displayed.

- **Login Page**: This page is for those users who have already registered themselves on the app and have a username and a password. There is also a way on this page for the new users to register themselves which will take them to the registration page.

- **Registration Page:** This is page is solely designed for the new users of the app who are willing to register themselves. This page takes input of the various details of the user and stores it in the database, later helping the user to login into the account with credentials they have provided.

- **Google Sign Up:** This page comes into picture when one of the users wants to sign up for the app and he/she might want to sign up then there is an option for them that he/she will login through their registered google account and once they sign up their data is automatically saved in the history.

- **Search:** This is the most important part of the application that provides interactivity within the app as it connects the various activities together like it is a side bar on which the profile, the dashboard section, the FAQ section, the About page of the page are linked and on clicking on each you can visit the pages.

- **Dashboard Page:** This is the page displayed for every user after entering the app successfully. It contains the search bar where the user can search the doctor according to the wish as well as some of the disease lists are suggested so that user can easily search for their doctor as per their need.

- **Appointment Description Page**: After searching there are a number of doctors that appear, when the user selects the appointment page the page will be displayed that will contain all the details of the doctors i.e., the doctor's name, their specification and a short summary of the doctors and the link to book the appointment.

- **Medical Folder:** It contains the information of all the medicines and saves data. In this we can search for any medicines.

- **Profile**: This page will contain all the user details that the user entered while creating the account on the app. The user can update and make changes to all this information as desired.

- **FAQ Pages:** This page contains some of the questions that might arise in the mind of the users while using the app and to answer those, these answers are pre-written.

- **Logout page**: Then is this last panel for the users to sign out from the account. As soon as the users sign out, they are brought back to the login page.

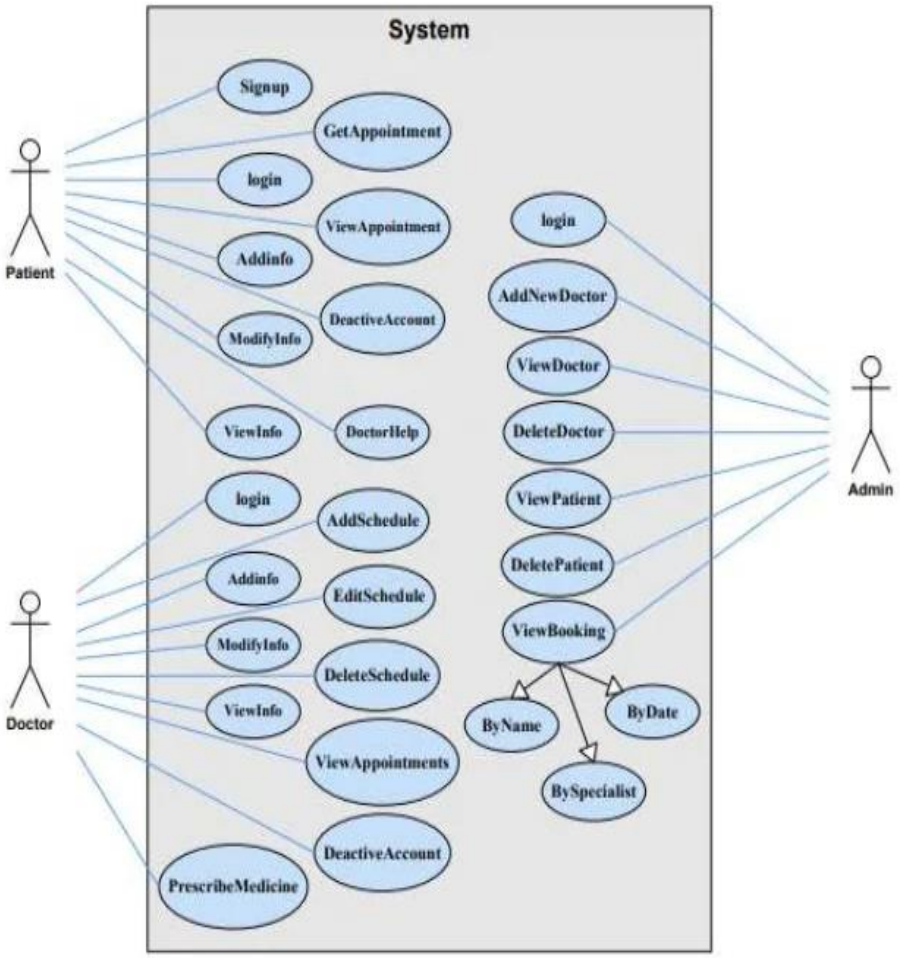
2.5 MEDHELP ON ANDROID APPLICATION

The establishment and improvement of doctor-patient interaction system is a very important requirement, especially now when the mobile communication technology is developing rapidly. The advantages of mobile web can be made full use of to make up the time and distance gap between doctors and patients and to provide fast and adequate medical services. Through the connection between mobile terminals and specific service, both doctors and patients are able to obtain required data to achieve a better interaction. Android is a Linux based open-source operating system which is mainly used in portal devices with excellent performance thus making its market share growing. The platform, Web services and database technology are all gradually maturing, so that we can develop a doctor- patient interaction system on Android platform to meet the needs of the patient and provide doctors more efficient and convenient means of communication with patients. This app is containing information of all the doctor and is very useful for patients.

CHAPTER- 3

SOFTWARE DESIGN

3.1 USE-CASE DIAGRAM:



Use case diagram of the system

So, the above diagram represents the point of view of the patient, the doctors, and the admin and the arrows to each module show the interactivity of the person.

The Patient will first be required to create up a new account so will interact with the “sign-up module” and fill up all the details that will be stored in the database. The next patient will land into the dashboard where there will be “search bar” where the patient can enter the name of doctor or a disease for that he/she must need a doctor. The list of doctors related to the search will appear on the screen and the patient will interact with “select the doctor” module. Then the patient can interact with the “book appointment” module to book the appointment.

For the registered patient, the user will be having the credentials to login and will interact with the “login module” and then the user will enter into the dashboard where there will be “search bar” where the user can enter the disease for what they need a doctor. The list of doctors will appear on the screen and the registered patient will interact with “select the doctor” module. Then the patient can interact with the “book appointment” module to book the appointment.

For the admin he can connect with each and every module mentioned in the use case diagram. Apart from the modules mentioned in use case diagram there are modules like profile, sign out, FAQ and about us section that every registered user can access.

3.2 DATA FOW DIAGRAM

The doctor list details contain the doctor's name, the doctor specification that in which fields he/she have their career, and a short review about the doctor and the link to book the appointment. .

As soon as the user encounters with the login module, we check the credentials of the user if the credentials are correct as per the database, we proceed to the dashboard else if wrong we then it will not accept that and the process is not continue for further things to go. From the dashboard module, we can interact with the profile module, the FAQ Section, The sign out section. On searching the doctor, from the dash board module, we encounter the doctor book list activity, checking the booking details activity and the published date activity.

The DFD for the Medhelp is shown below:

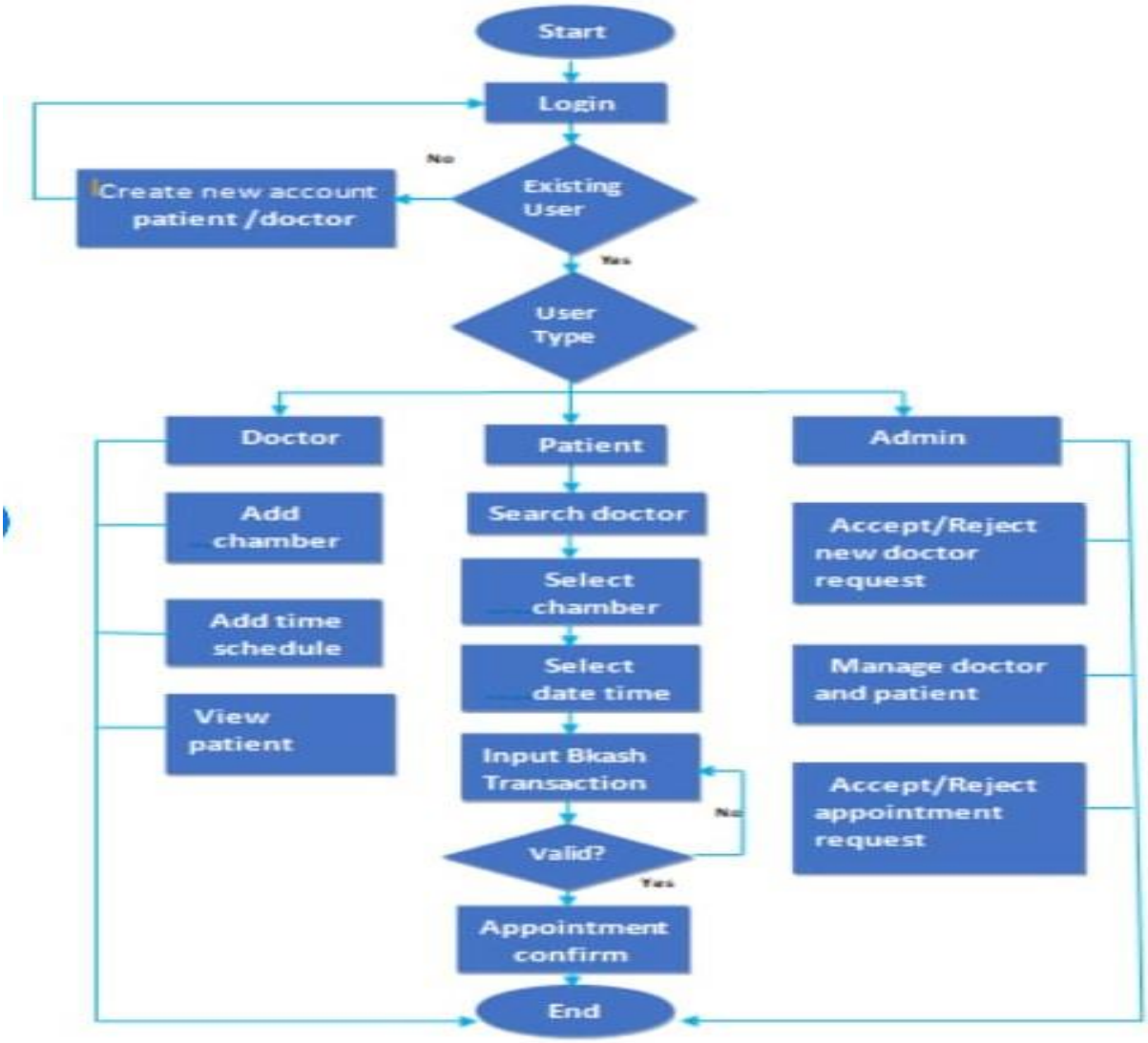


Figure-3: Data Flow Diagram

3.3 SEQUENCE DIAGRAM

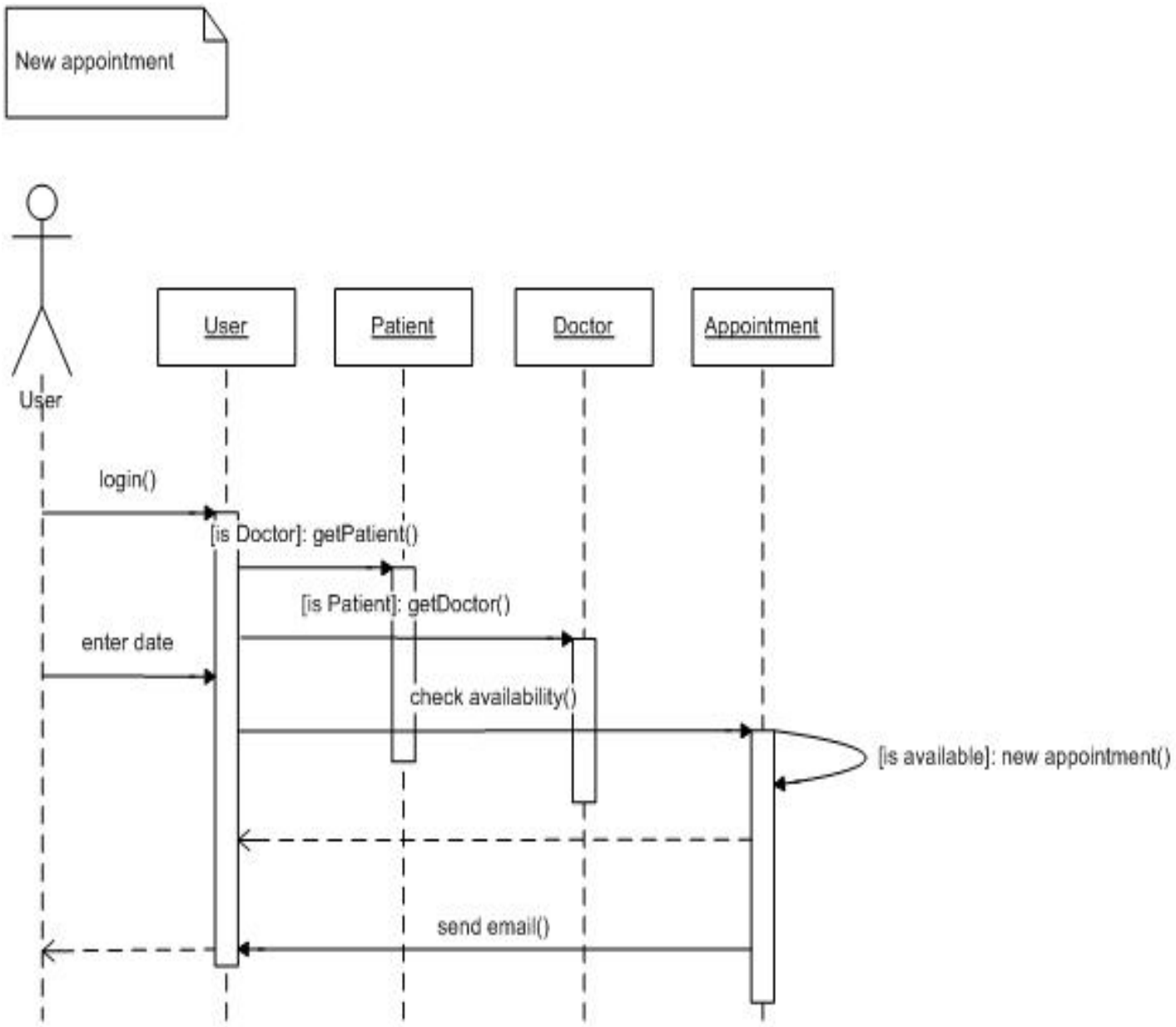


Figure-4: Sequence Diagram

CHAPTER-4 TECHNOLOGY USED

4.1 ANDROID

Android is a Linux-based operating system designed primarily for touch screen devices such as smart phone tablets and computers. Released in 2008, is now owned by Google. So android is a operating system like Windows, Ubuntu and Mac OS and a lot number of devices use Android these days like mobile phones, watches, laptop and television. So, we also created an android application "Medhelp". Play Store is a market place for all the Android Apps. So, we need to know what basically an android app is. An Android app is software running on a Android Platform. So, this can be concluded that like all the software it is a combination of Backend and Frontend. Backend to design the logical parts of the app, for the functionality whereas Front End to develop the User Interface. And to implement the various parts of the android app, we require a number of tools and technologies which will come into picture. But first it would be great to see the three different types of Android Apps: -

- **Native Apps:** An executable program coded in the machine language of the hardware platform it is running in. **Native applications** are compiled into the machine language of that CPU. For example, **Windows** and Mac executable **apps** are in x86 machine language, while **mobile apps** are ARM based. Native apps are the most common. They're coded in a specific language like Swift for **iOS** or Java for Android. A popular example is WhatsApp.

- **Web Apps:** are accessed via the internet browser and will adapt to whichever device you're viewing them on. They are not native to a particular system, and don't need to be downloaded or installed. Due to their responsive nature, they do indeed look and function a lot like mobile apps — and this is where the confusion arises.

- **Hybrid Apps:** Hybrid apps are deployed in a native container that uses a mobile Web View object. When the app is used, this object displays web content thanks to the use of web technologies (CSS, JavaScript, HTML, HTML5). It is in fact displaying web pages from a desktop website that are adapted to a Web View display. The web content can either be displayed as soon as the app is opened or for certain parts of the app only i.e., for the purchase funnel. In order to access a

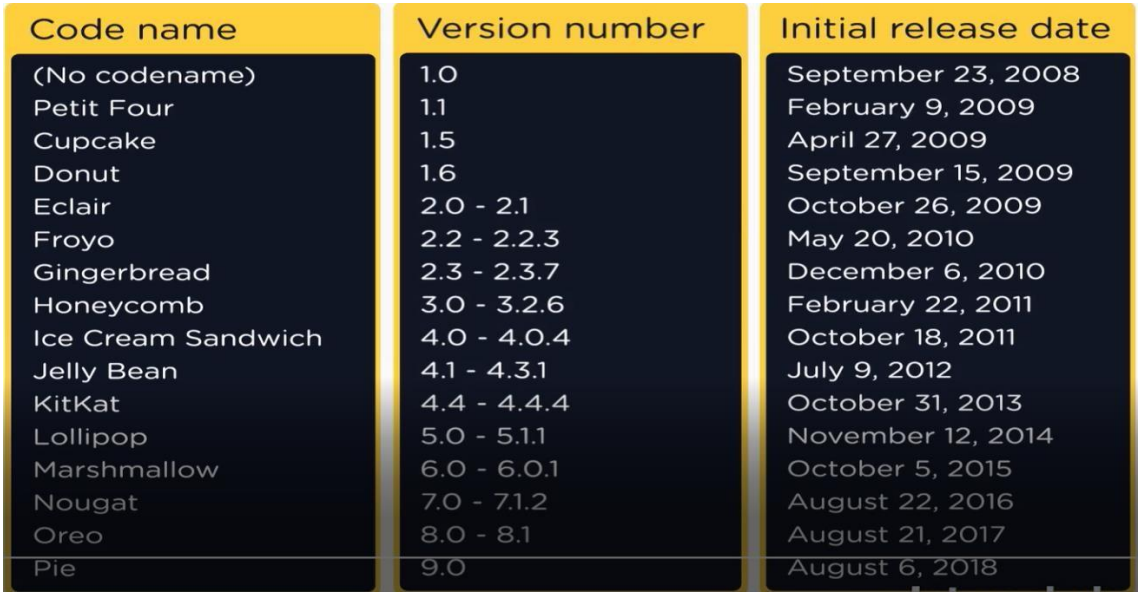
device's hardware features (accelerometer, camera, contacts…) for which the native apps are installed, it is possible to include native elements of each platform's user interfaces (iOS, Android): native code will be used to access the specific features in order to create a seamless user experience. Hybrid apps can also rely on platforms that offer JavaScript

APIs if those functionalities are called within a Web View

4.2 VERSION OF ANDROID

Each year Android releases a new version with better features, better security and better User Interface experience and a new symbol. Here is the table of list of versions

Figure-5: Android KitKat



Code name	Version number	Initial release date
(No codename)	1.0	September 23, 2008
Petit Four	1.1	February 9, 2009
Cupcake	1.5	April 27, 2009
Donut	1.6	September 15, 2009
Eclair	2.0 - 2.1	October 26, 2009
Froyo	2.2 - 2.2.3	May 20, 2010
Gingerbread	2.3 - 2.3.7	December 6, 2010
Honeycomb	3.0 - 3.2.6	February 22, 2011
Ice Cream Sandwich	4.0 - 4.0.4	October 18, 2011
Jelly Bean	4.1 - 4.3.1	July 9, 2012
KitKat	4.4 - 4.4.4	October 31, 2013
Lollipop	5.0 - 5.1.1	November 12, 2014
Marshmallow	6.0 - 6.0.1	October 5, 2015
Nougat	7.0 - 7.1.2	August 22, 2016
Oreo	8.0 - 8.1	August 21, 2017
Pie	9.0	August 6, 2018

Table -1: Versions of Android

4.3 TOOLS AND LANGUAGES

Tools used to build the Android App are: -

- **Android Studio**: Android Studio is an environment that help us create and edit Android applications. It is the official IDE for Android App Development. It has intelliJ's powerful code editor and developer tools and various features that enhance productivity while developing apps.

- **Software Development Kit (SDK)**: Android Studio requires a collection of libraries and data therefore SDK is mandatory.

Languages used in building an Android Application are classified as per the Front End and Back End. For designing the Front End of an application, we have used JAVA,XML and for designing the Back End we have used Kotlin.

- **XML:** XML is the extensible Markup Language. It is the met language which allows users to define their own customized markup language especially in order to display documents

on Internet. It is the language that contains tags that store information. And the tags can be used to present data on the screen.

- **JAVA:** Java is a programming language and a platform. Java is a high level, robust, object-oriented and secure programming language. Java was developed by Sun Microsystems (which is now the subsidiary of Oracle) in the year 1995. James Gosling is known as the father of Java. Before Java, its name was Oak. Since Oak was already a registered company, so James Gosling and his team changed the name from Oak to Java. Platform: Any hardware or software environment in which a program runs, is known as a platform. Since Java has a runtime environment (JRE) and API, it is called a platform.

- **Kotlin:** Kotlin is statically typed programming language based on Java Virtual Machine. Kotlin is the fundamental language of Android since 2017 as declared by Google Developers of Android also prefer to use Java for the backend but Kotlin has a upper-hand due to many features like Java has a length syntax and hence sometimes the code is also redundant. To remove the boiler Plate code, Kotlin is preferred. Kotlin is cross platform, general purpose programming language with type inference. It can interoperate fully with Java but type inference allows its syntax to be more concise.

4.4 BASIC TERMINOLOGY

- **Layout:** Layout is the parent of view. It arranges all the views in a proper manner on the screen.

- **Activity**: An activity can be referred as your device's screen which you see. User can place UI elements in any order in the created window of user's choice.

- **View**: A view is an UI which occupies rectangular area on the screen to draw and handle user events.

- **Emulator**: An emulator is an Android virtual device through which you can select the target Android version or platform to run and test your developed application.

- **Manifest file**: Manifest file acts as a metadata for every application. This file contains all the essential information about the application like app icon, app name, launcher activity, and required permissions etc.

- **API:** Short for Application Programming Interface. APIs are functioning those developers can call on to access specific features by calling upon programs, code, and services that others have written. For example, if a developer wants to draw a button on the screen, she can insert a small bit of code that says "draw this kind of button, with this color and size and style, at this location" instead of dozens of lines of code that tells the graphics processor, in detail, exactly how to draw a button. If the application wants your location, it can use the location API to "get the device's location" and let Google's code handle the rest, instead of requiring the developer to build an entire location service from scratch just for her own app. There are thousands of APIs in Android, covering everything from drawing interface elements, to the cameras, to location access, to accessing storage, to 3D graphics (see: OpenGL ES) and much more.

- **Intent:** Intents are an essential part of the Android ecosystem. They are used to express an action to be performed. Intents allow you to interact with components from the same applications as well as with components contributed by other applications. It can be classified into implicit and explicit intents.

- **Implicit intent:** It does not name a specific component, but instead declare a general action to perform, which allows a component from another app to handle it.

- **Explicit Intent:** It specifies the component to start by name. You'll typically use an explicit intent to start a component in your own app, because you know the class name of the activity or service you want to start.

- **APK**: Short for "Android application package." The extension used in Android app installation files (e.g., app.APK). Similar in nature to an EXE file on Windows.

- **SDK:** Short for "Software Development Kit." As it pertains to Android, the SDK is a set of tools such as code libraries, a debugger, and a handset emulator that can be run on Windows, Mac, or Linux to facilitate the creation of Android apps by developers. While the SDK is generally intended for use by developers, end users can install the software on their home computer to execute ADB and Fast boot commands.

- **Action Bar**: The action bar is an important design element, usually at the top of each screen in an app that provides a consistent familiar look between Android apps. It is used to provide better user interaction and experience by supporting easy navigation through tabs and drop-down lists.

- **Navigation bar**: Android Navigation Drawer is a sliding left menu that is used to display the important links in the application. Navigation drawer makes it easy to navigate to and fro between those links. It's not visible by default and it needs to opened either by sliding from left or clicking its icon in the Action Bar.

- **Fragment**: A Fragment represents a behaviour or a portion of user interface in a Fragment Activity. You can combine multiple fragments in a single activity to build a multi-pane UI and reuse a fragment in multiple activities.

- **Firebase** is a Backend-as-a-Service (Baas). It provides developers with a variety of tools and services to help them develop quality apps, grow their user base, and earn profit. It is built on Google's infrastructure. Firebase is categorized as a NoSQL database program, which stores data in JSON-like documents. Firebase has three core services: a real-time database, user authentication and hosting. With the Firebase iOS SDK, you can use these services to create apps without writing any server code.

JSON stands for JavaScript Object Notation. It is an independent data exchange format and is the best alternative for XML. JSON is used for data interchange (posting and retrieving) from the server. Hence knowing the syntax and its usability is important.

JSON is the best alternative for XML and its more readable by human

CHAPTER -5

IMPLEMENTATION AND USER INTERFACE

Creating an app concept design with screen sketches and functional flow diagrams is the best way to communicate your vision to the mobile app developer. Making the concept clear to the developer is probably the most important factor in successful mobile app development. Yet it is one of the most common problems or obstacles in a mobile app development outsourcing project.

No matter what the marketing and profit goals are or if you are outsourcing an app for your personal use, you need to fully design and document the app concept if you expect a programmer to make your vision a reality. Developers are not mind readers and even descriptions given during conversations can be very fleeting or interpreted differently. Fully documenting your concept, therefore, leaves little to chance. The two most important things to do are: A- make a comprehensive description of how the app works and what it does (functionality) and B- create a comprehensive description of what the user sees and does (look and feel).

5.1 Implementation of the Medhelp:

Implementation of Medhelp is taken place in various phases. Firstly, we build the login interface then search i.e., make fragment for each of the doctor list using the search bar view and the make various layout for the supporting features. And finally, we parse the Jason object to get the data in the required format and then display the result.

5.1.1 Step to be followed to develop the app:

1.Firstly, we create login phase which comprises of various phases that are mentioned below:

- Login Page: allows user to login into the app if the user is existing one
- Register Page: If the user is new to our app, then firstly, he/she have to register themselves on the app.
- Google sign up: allows user to login through their google id and password directly.
- For authenticating the user, we have used cloud firebase authentication.

2.Now, we are going to create search drawer for that purpose we have used following functionality of android:

- Fragments(SupportFramentManager)
- Menu – items
- Drawer header
- Hamburger icon
- ActionBarDrawableToggle (help to create search bar)

3. Creating fragment for each of the menu item. Our Menu items are:

- Dashboard
- Profile
- search
- FAQ
- Appointment
- Sign-Out

4.Now we have created various activities like Doctor List, patient details and many more.

1. . In this step we connect our app with the FIREBASE using GOOGLE CLOUD FIREBASE.
2. Now we add data (that we have received from Google FIREBASE) to the profile description activity.
3. After that there is a medical folder activity section in which we save the data of the patient

Flow Chart for the User is given below:

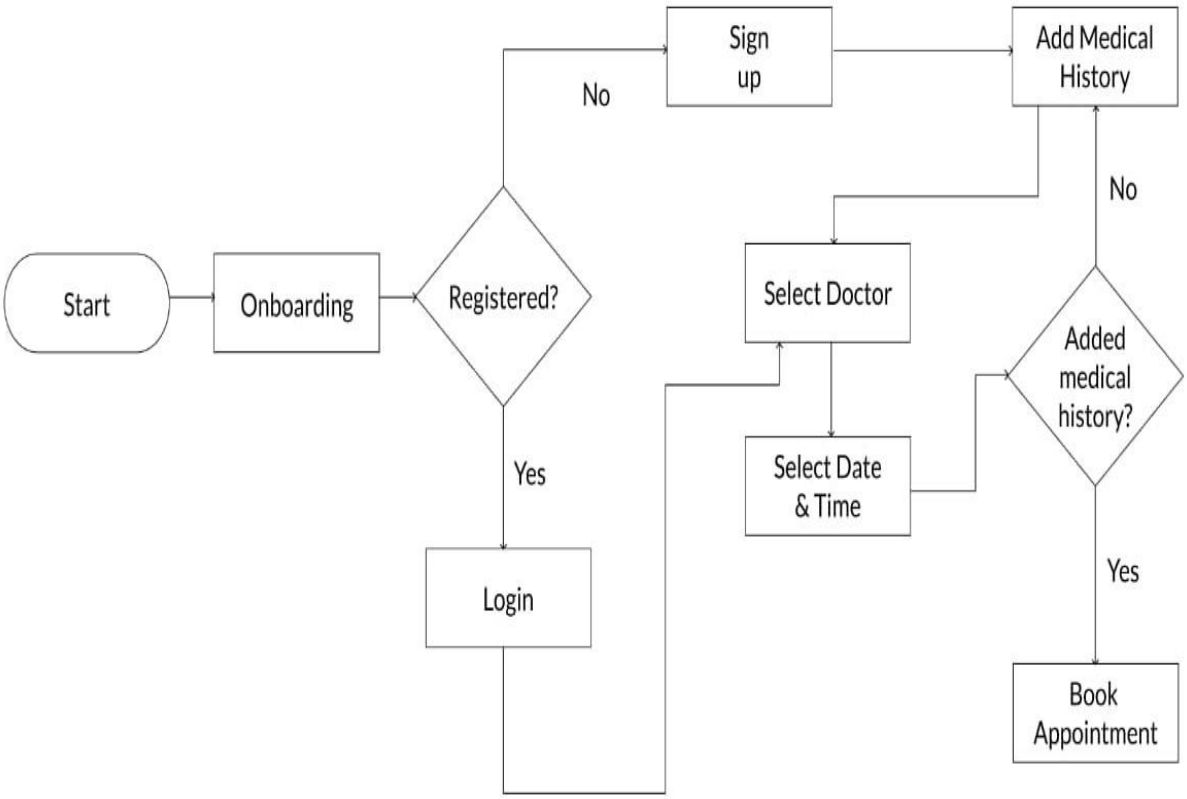


Figure-6: Flow Chart for User (PATIENT SIDE)

5.1.2 Step to be followed by the user

1. Firstly, we have the Login activity which consists of following steps
    - Register: for new User
    - Login: For existing as well as new user
    - Google sign up: To sign in direct through google.
2. We authenticate and store the user information from the Firebase authentication.
3. After that, we made a Drawer layout of our Medhelp app which includes various functionality
    - Profile Fragment: To check the profile and update the database.
    - Dashboard Fragment: Show the book on the genre basis and it is open by default. .
    - FAQ's Fragment: It comprises all the frequently asked questions.
    - Profile: Information about the patient(user)

- Sign-out Fragment: Remove/logout you from the app.

4. In Dashboard fragment we include the search bar in which the user can search for the doctor of their interest.
5. After that list of doctors according to your search will appear
6. Select the doctor according to your choice.
7. Then the booking option of the book will appear. It comprises of the following things:
    - Book as a consultation
    - Book as a hospitalization
    - Select Date
    - Select time
    - confirm
    - preview
8. Now your appointment is book.

5.2 User Interface

- **Sign up Page**

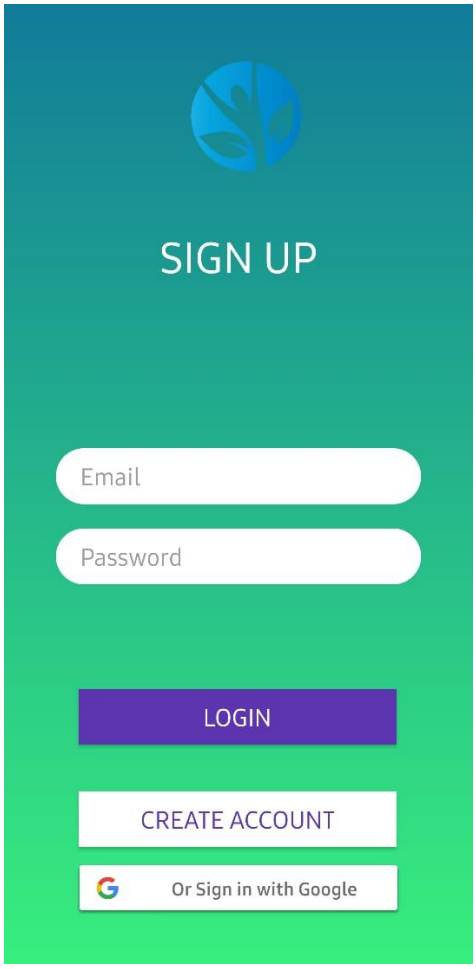


Figure-8: sign up Page

- **Create account**

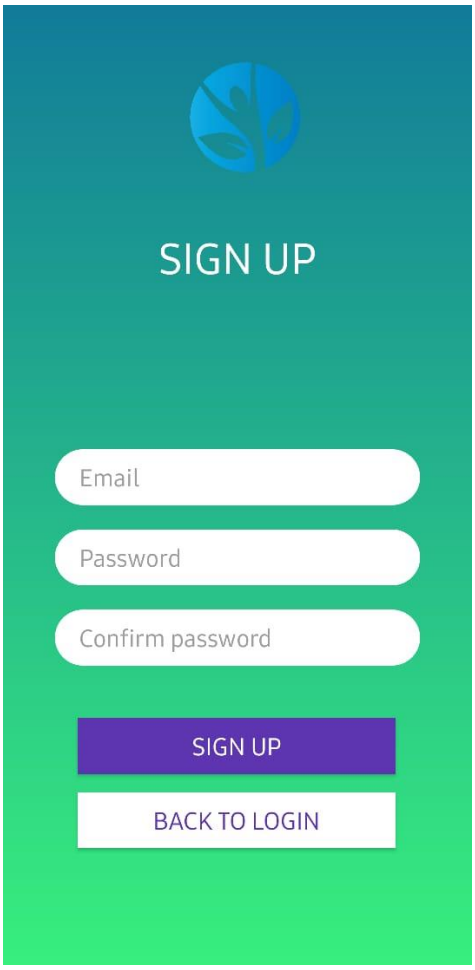


Figure-9: create account

- **Dashboard**

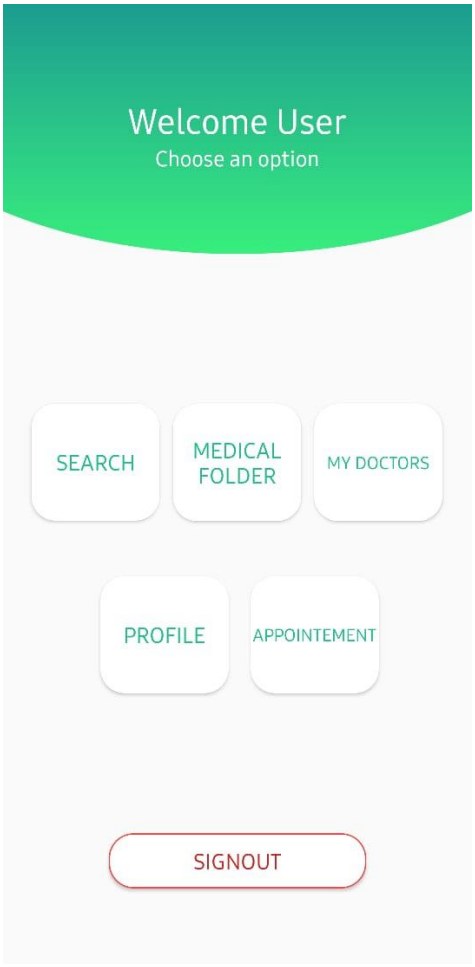


Figure-10: Dashboard

- **Search Drawer**

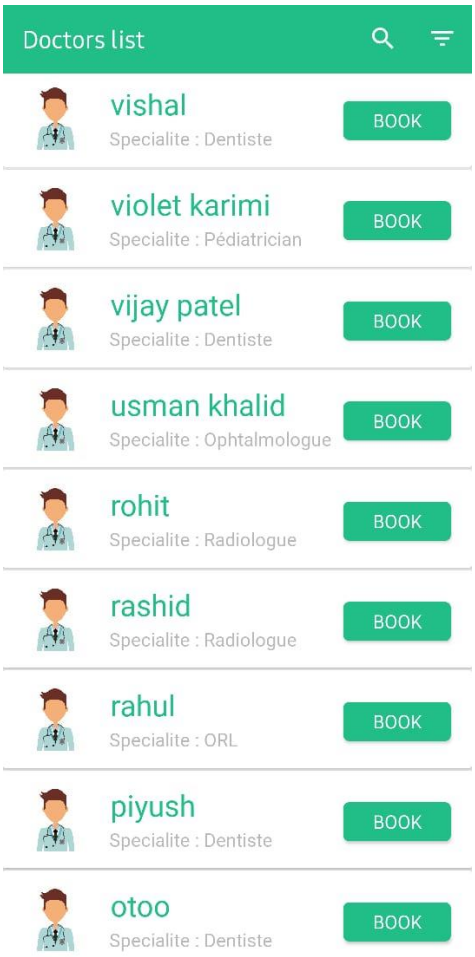


Figure-11: Search Drawer

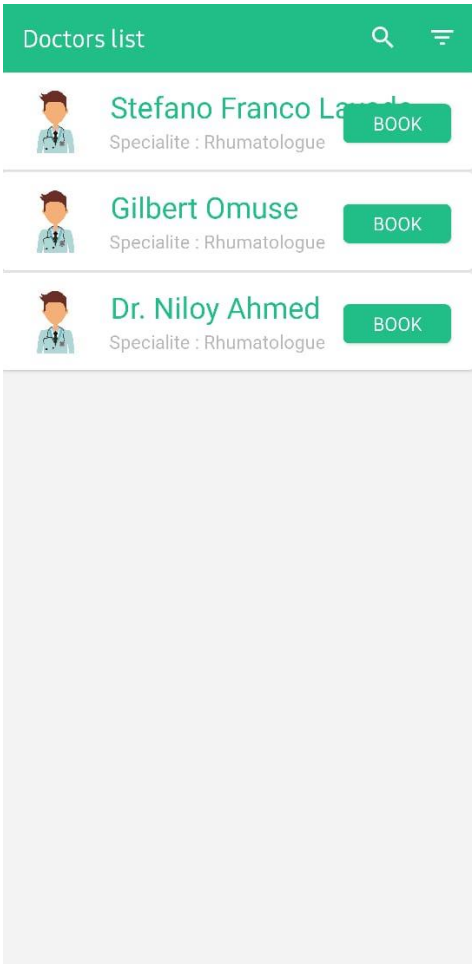


Figure-12 list of doctors as per search

- **Profile**

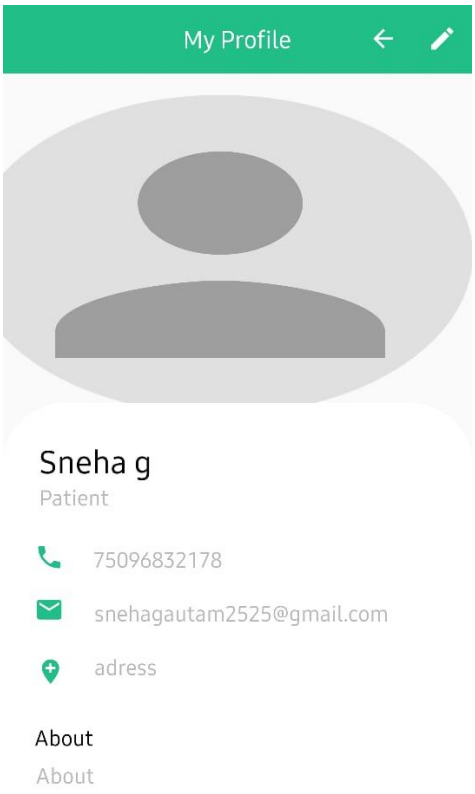


Figure-13: Profile page

- **Appointment Page**

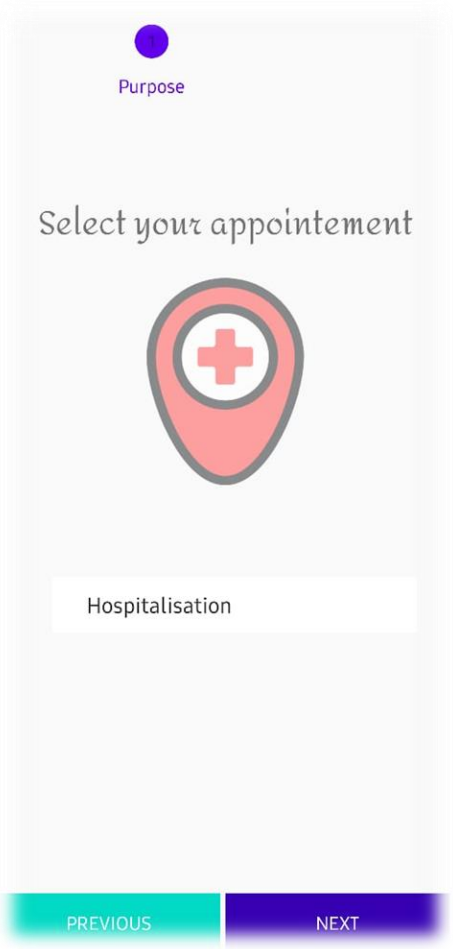
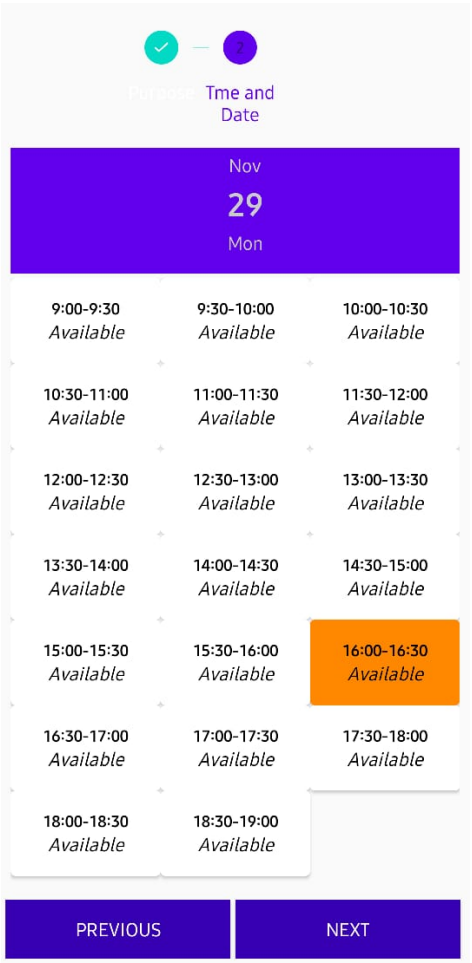
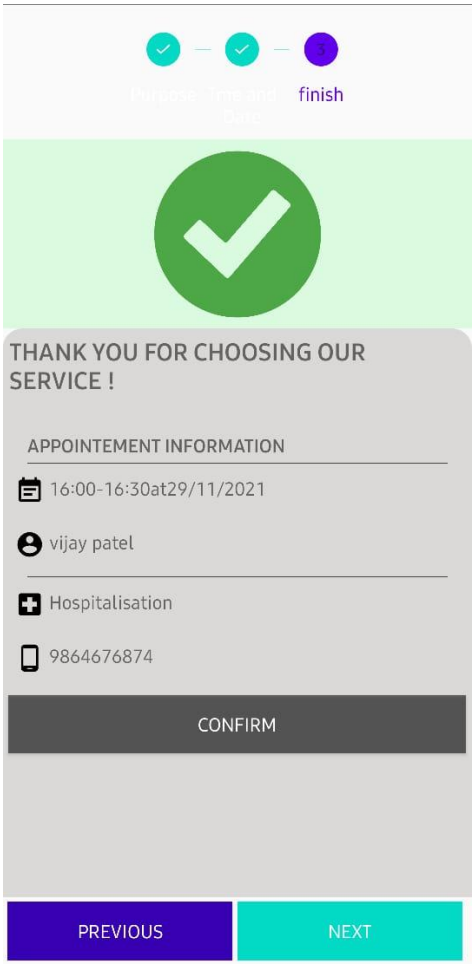


Figure-15: Appointment page



Appointment page(1)



Appointment page(2)

- **Booking Appointment details**

CHAPTER - 6

TESTING

Once source code has been generated, software must be tested to uncover as many errors as possible before delivery. It is very important to work the system successfully and achieve high quality of software. Testing includes designing a series of test cases that have a high likelihood of finding errors by applying software-testing techniques.

System testing makes logical assumptions that if all the parts of the system are correct, the goal will be successfully achieved. The system should be checked logically. Validations and cross checks should be there. Avoid duplications of record that cause redundancy of data.

In other Words, Testing is the process of evaluating a system or its component(s) with the intent to find whether it satisfies the specified requirements or not. It is executing a system in order to identify any gaps, errors, or missing requirements in contrary to the actual requirements.

The Android framework includes an integrated testing framework that helps you test all aspects of your application and the SDK tools include tools for setting up and running test applications. Whether you are working in Eclipse with ADT or working from the command line, the SDK tools help you set up and run your tests within an emulator or the device you are targeting.

There are different types of testing some of them are listed below:

6.1 Installation Testing:

There are two types of apps on an Android device i.e., pre-installed applications and the applications which are installed later by the user.

For both of the above, installation testing is carried out by our teammates. It is ensuring smooth installation of the application without ending up in errors, partial installation etc.

6.2 Unit Testing

It focuses on smallest unit of software design. In this we test an individual unit or groups of inter related units. It is often done by programmer by using sample input and observing its corresponding outputs. In this testing technique we are primarily focuses on

- Loop methods and function is working fine or not. Misunderstood or incorrect Arithmetic precedence
- Incorrect Initialization

Unit Testing of the app:

Test cases	**Description**	**Expected Outcome**	**Result**

1	Start Page – Launch Screen	Should display splash screen with animated text	Pass
2	Register Screen	Should display register activity where you need to fill the required details	Pass
3	Login Screen	Should display login screen And ask for your credentials.	Pass

4	Google sign up	Should login through google	Pass
5	FAQ	Should display the Frequently asked question	Pass

6	View Favorites	Should display the Favourite activity	Pass

7	Profile	Should display the information that you have entered	Pass
8	View Doctor	View Home page	Pass

9	Search bar	Should give the details of the doctor's that you have searched	Pass
10	View book appointment page	Should show the information of the appointment book	Pass
11	Add to favourites	Should add the selected book into the favourites	Pass
12	Clear Favourites	Should remove the selected book from the favourites	Pass

13	Preview	Should display the preview of the doctor	Pass
14	Logout	Sign out you from the app	Pass

Table 1: Unit Testing of Medhelp

6.3 User Testing

User testing is the process through which the interface and functions of a website, app, product, or service are tested by real users who perform specific tasks in realistic conditions. The purpose of this process is to evaluate the usability of that website or app and to decide whether the product is ready to be launched for real users.

This app was tested by our team mates and friends who are using different mobile phones (and having different android version) also tested on different emulator to check its performance and it seems to be working fine and users of this app are satisfied with the facilities and performance of the app and like the way how the app is worked.

6.4 Performance Testing

In this type of testing, we have checked the performances of our application under some peculiar conditions are checked. Those conditions include:

- Low memory in the device.
- The battery in extremely at a low level.

- Poor/Bad network reception.

Performance is basically tested from 2 ends, application end, and the application server end. Our app is also performing well in this phase of testing as well. And we are getting positive feedback from user of our app.

6.5 Compability Testing

This application was tested and used on different devices like LG G3, Google Nexus 4. The application worked fine and is stable. The application worked fine in portrait mode and there isn't any problem with compatibility.

On all types of testing (that we have performed above) our performing well on our app i.e. Medhelp.

CHAPTER -7

CONCLUSION

So, we come up to the conclusion that, this app will help normal users to access limited doctor's information and take appointments. The administrator can login using the in-built id and password and access all information, delete and update them. Administrator can view appointments too. Common users who don't know the password can only get access to limited and relevant information on any doctor. From this project we learnt a lot on team work and it has also enhanced us knowledge of the programming methods since, we learnt a lot along the way. This application serves the basic purpose of accessing doctor's information, booking appointments and modifying information.

FUTURE SCOPE

In a nutshell, it can be summarized that the future scope of the project circles around maintaining information regarding We can add printer in future. We can give more advance software for Doctor Appointment System including more facilities. We will host the platform on online servers to make it accessible worldwide

• Integrate multiple load balancers to distribute the loads of the system

• Create the master and slave database structure to reduce the overload of the database queries

Implement the backup mechanism for taking backup of codebase and database on regular basis on different servers

The above-mentioned points are the enhancements which can be done to increase the applicability and usage of this project. Here we can maintain the records of doctor and Appointment. Also, as it can be seen that now-a-days the players are versatile, i.e., so there is a scope for introducing a method to maintain the Doctor Appointment System Enhancements can be done to maintain all the Doctor, Appointment. Patient Booking, Doctor Schedule.

We have left all the options open so that if there is any other future requirement in the system by the user for the enhancement of the system then it is possible to implement them in the last we would like to thanks all the persons involved in the development of the system directly or indirectly. We hope that the project will serve its purpose for which it is develop there by undefining success of process.

LIMITATIONS OF MEDHELP

Although We have put my best efforts to make the software flexible, easy to operate but limitations cannot be ruled out even by me. Though the software presents a broad range of options to its users some intricate options could not be covered into it partly because of logistic and partly due to lack of sophistication. Paucity of time was also major constraint; thus, it was not possible to make the software fool proof and dynamic...

Lack of time also compelled me to ignore some part such as storing old result of the candidate etc. Considerable efforts have made the software easy to operate even for the people not related to the field of computers but it is acknowledged that a layman may find it a bit problematic at the

first instance. The user is provided help at each step for his convenience in working with the software.

REFERENCES

1. Database Programming with JDBC
2. http://www.jdbc-tutorial.com
3. Java and Software Design Concepts by Apress
4. https://docs.oracle.com/javase/tutorial/
5. https://www.tutonalspoint.com/java/
6. http://www.javatpoint.com/java-tutorial/
7. . http://www.JSP.net!
8. http://www.tutorialspoint.com/mysql
9. httpd.apache.org/docs/2.0/misc./tutorials.html
10. Android developer Guide: https://developer.android.com/
11. For rectifying the error: https://stackoverflow.com/
12. Google for problem solving