

DigiDoc: A smart approach for telemedicine

BY

Sayma Fariha Sharna

ID: 181-15-11168

Kazi Shihabur Rahman

ID: 181-15-11148

AND

Shanto Chandra Bhowmick

ID: 181-15-11146

This Report Presented in Partial Fulfillment of the Requirements for the Degree of
Bachelor of Science in Computer Science and Engineering

Supervised By

Md. Jueal Mia

Sr. Lecturer

Department of CSE

Daffodil International University

Co-Supervised By

Md. Zahid Hasan

Assistant Professor

Department of CSE

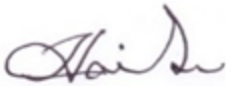
Daffodil International University



APPROVAL

This Project/internship titled “**Digi-Doc: A smart approach for telemedicine**” submitted by Sayma Fariha Sharna, ID No: 181-15-11168, Kazi Shihabur Rahman, ID No: 181-15-11148, Shanto Chandra Bhowmick, ID No: 181-15-11146 to the Department of Computer Science and Engineering, Daffodil International University has been accepted as satisfactory for the partial fulfillment of the requirements for the degree of B.Sc. in Computer Science and Engineering and approved as to its style and contents. The presentation has been held on 05 January,2022.

BOARD OF EXAMINERS



Chairman


Dr. Sheak Rashed Haider Noori (SRH)

Associate Professor and Associate Head

Department of Computer Science and Engineering

Faculty of Science & Information Technology

Daffodil International University



Internal Examiner

Md. Tarek Habib (MTH)

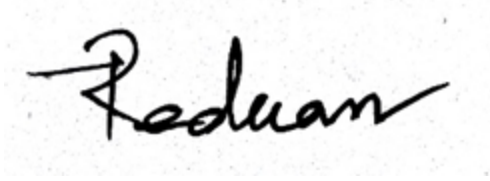
Assistant Professor

Department of Computer Science and Engineering

Faculty of Science & Information Technology

Daffodil International University

Internal Examiner



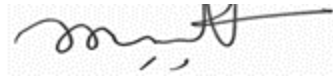
Md. Reduanul Haque (MRH)

Assistant Professor

Department of Computer Science and Engineering

Faculty of Science & Information Technology

Daffodil International University



External Examiner

Dr. Mohammad Shorif Uddin

Professor

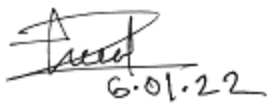
Department of Computer Science and Engineering

Jahangirnagar University

DECLARATION

We hereby declare that, this project has been done by us under the supervision of **Md. Jueal Mia, Sr. Lecturer, Department of CSE** Daffodil International University. We also declare that neither this project nor any part of this project has been submitted elsewhere for award of any degree or diploma.

Supervised by:



6.01.22

Md. Jueal Mia

Sr. Lecturer

Department of CSE

Daffodil International University

Co-Supervised by:



Md. Zahid Hasan

Assistant Professor

Department of CSE

Daffodil International University

Submitted by:



Sayma Fariha Sharna

ID: -181-15-11168

Department of CSE
Daffodil International University

Shihabur

Kazi Shihabur Rahman

ID: -181-15-11148

Department of CSE

Daffodil International University

Shanto

Shanto Chandra Bhowmick

ID: 181-15-11146

Department of CSE

Daffodil International University

ACKNOWLEDGEMENT

First we express our heartiest thanks and gratefulness to almighty God for His divine blessing makes us possible to complete the final year project/internship successfully.

We owe a huge debt of gratitude to **Md. Jueal Mia, Senior Lecturer, Department of CSE,** Daffodil International University, Dhaka. To complete this project, our supervisor must have extensive knowledge and a strong interest in the subject of "Android Application Development." His never-ending patience, intellectual direction, constant encouragement, persistent and vigorous supervision, constructive criticism, helpful suggestions, and reading many inferior drafts and revising them at every level allowed this project to be completed.

We would like to express our heartiest gratitude to **Professor Dr. Touhid Bhuiyan** and Head, Department of CSE, for his kind help to finish our project and also to other faculty members and the staff of CSE department of Daffodil International University.

We would like to thank our entire course mate in Daffodil International University, who took part in this discussion while completing the course work.

Finally, we must acknowledge with due respect the constant support and patients of our parents

Table of Content

APPROVAL	I
DECLARATION	III
ACKNOWLEDGEMENT	V
ABSTRACT	VI
LIST OF FIGURES	XI
Chapter 1:	9
Introduction:	9
1.1 Background of the Project:	9
1.2 Objective Of Project:	9
1.3 Aim of The Project:	9
1.4 Existing System:	10
1.5 Problem Statement:	10
1.6 Research Methodology:	10
1.7 Proposed System:	11
Registration & Log In:	11
1.8 Conclusion:	11
Chapter 2	12
Requirement Analysis	12
2.1 Introduction :	12
2.2 Requirements analysis	12
2.3 Conclusion	13
Chapter 3	13
Literature review	13
3.1 Introduction	13
3.2 Literature review	13
3.3 Research Methodology	14
3.4 Conclusion	15
Chapter 4	16
System Analysis and Design	16

4.1 Introduction	16
4.2 System Analysis	16
4.3 System Design	17
4.3.1 Block Diagram	17
4.3.2 Flow chart	18
4.3.2.1 Patients Module Flow-Chart	18
4.3.2.2 Doctors Module Flow-Chart	19
4.4 Conclusion	19
Chapter 5	20
System Evolution/Development	20
5.1 Introduction	20
5.2 Front end development	20
5.2.1 User interface Design	20
5.2.1.1 Patients Registration:	21
5.2.1.2 Doctors Registration:	22
5.2.1.3 Authentication	22
5.2.1.4 Dashboard Design:	22
XML Code:	22
Fragment:	23
Doctor Category Activity:	24
Fragment:	24
5.3 Back End development	25
5.3.1 Database Design	25
5.4 Conclusion	26
Chapter 6	26
Testing and Implementation	26
6.1 Introduction	26
6.2 Testing	26
6.3 Implementation	27
6.4 Result and Discussion	27
6.5 Conclusion	27
Chapter 7	28
Critical Appraisal	28
7.1 Introduction	28
7.2 SWOT Analysis	28
7.2.1 Strength	28
7.2.2 Weakness	28
7.2.3 Opportunity	29

7.2.4 Thread	29
7.3 Conclusion	29
Chapter 8	30
Conclusion	30
8.1 Conclusion	30
8.2 Further Suggested Work :	30
Reference	31
Appendix	31
Appendix -A Backend Code	31
Doctor Registration:	31
Patients Registration:	39
Doctor's Department List Activity :	44
Book Appointment System:	49
Confirm Appointment Activity:	56
Call Option:	59
Appendix -B Database Design:	60
Appendix -C GUI Design:	62

ABSTRACT

During the pandemic COVID-19, everything becomes so difficult for people. At this time, the most needed services are health care services. It is also becoming very difficult to get. People can not travel from one place to another for medical treatment. And medical care has become one of the most dangerous places. The spreading system of COVID-19 become more powerful in medical care. So the health care unit can not give their services perfectly. On the other hand, people who are older can not go to health care because of this pandemic. Older people become more sick in this COVID-19 time. At this point, an online base all telecommunication system is the one option. In this era, almost 98% of people have an android phone or smartphone. And this is a huge chance to create an e system. Phone base health care system. It will reduce the contact of person to person which means reducing the spreading of COVID-19. Telemedicine can create a good impact in this situation. There is one more plus point that telemedicine can create, it can reduce the time. People in the large city face traffic jam problems which kill valuable time. Telemedicine can reduce it. Phone base telemedicine can create huge impact on medical field

Chapter 1:

Introduction:

In the pandemic time, our medical system became very weak. The physical medical care system has become more dangerous. This system increases the 90% possibility to get infected and also spreads of covid 19. In this modern era, we need a modern solution. Nowadays people have smartphones. 98% of

people have phones in our country. And 80% of people are smartphone users. If we count 80% as 100%, in this 100% of the people 80% people use android phones. We can build an android based health care system. So that, we can reduce the spread and also people can get their medical care. By this, we can also reduce the traveling time. And those patients who are not able to go to the doctor's chamber can get the treatment very easily. We can build a modern and affordable health care system with this application.

1.1 Background of the Project:

In our country, we still use an offline medical care system. People need to go to the hospital or medical care system to make an appointment with the doctor. And in this covid situation it has become a dangerous process. Govt. declare lockdown to protect the people of the country and also decrease the spreading. In this case, people can not go to the doctors office and medical care. People are suffering because of insufficient medical services. In these circumstances, we analyze some problems and try to make a solution to get rid of the problem.

1.2 Objective Of Project:

We have set some objectives to achieve our aims. This will make our way much easier. Objective is pointed below.

- **Centralized Healthcare System:** There are many doctors out there. So we try to create a centralised medicare system, where patients can find their needed doctors.
- **Reduce time:** People from any place can take treatment by this app. They don't have to go directly to doctors or hospitals which will save their time.
- **Patient safety:** In the event of an emergency, it will provide prompt service to the patients. The patient learns everything there is to know about their medical condition.
- **Advance Telemedicine:** By this app , we want to make a benchmark of advanced telimedication. This app will have an advanced appointment system,payment system, online base medical care system.

1.3 Aim of The Project:

We are expecting to provide this service all over the country. Hopefully, it will work very well.

- A centralised medical care system.
- Provide healthcare services to patients by a doctor.
- Saving time and work pressure
- Reduce the paperwork system
- Making a swift and easy way for both doctors and patients

1.4 Existing System:

According to study, the majority of notable doctors reside in large cities or central cities. Doctors do not want to travel to a low-income city. There are several diagnostic facilities and hospitals, but just a few well-known doctors who can treat patients. In this pandemic, most of the doctors remain busy in the

hospitals to treat the COVID patients. And for this, the rest of the people with other health problems can't get treatment timely.

1.5 Problem Statement:

When it comes to the challenges faced by Bangladeshi healthcare personnel during the COVID-19 pandemic, concerns raised by poor administration cannot be overlooked. During the COVID-19 pandemic, doctors are having a rough time at work. Although social isolation is the most effective method of preventing the virus from spreading, it is difficult to implement for healthcare workers who must come into close contact with COVID-19 patients, placing them at risk of infection. Many people can't get to the doctor due to transportation issues, and doctors can't come to them. It can also be seen that Patients receive healthcare from private medicals in 77.3 percent of cases. As a result, a substantial portion of the population suffered, and many died as a result of a lack of competent medical care.

1.6 Research Methodology:

- **Introduction:**

Choosing a methodology is critical to the success of any project. It should be appropriate and relevant to the project's circumstances. Methodologies assist in the establishment of a project team as well as the understanding of the project's business requirements in connection to the end-requirements. This section gives a high-level summary of the proposed project's strategy.

- **Data Collection Objective and Area:**

Concern has conducted telephone interviews with households to learn more about the impact of the Covid-19 pandemic and government actions on people's health, livelihoods, and coping mechanisms. To track how this is changing over time, households will be interviewed every two weeks or so.

- **Data Collection:**

Data collection methods come in a wide range of shapes and sizes. There are two types of approaches that can be used. One is the primary data collection system, while the other is the Second Stage Information Collection system.

- Primary data is information obtained directly from sources such as doctor hassles, checks, surveys, and scheduled interviews with workers and stakeholders. Primary data is split into qualitative and quantitative orders. Quantitative data is used to determine commercial rates, but qualitative data is used to assess attitudes, passions, and connections to the product or outcome.
- Information received from sources other than the original stoner is referred to as secondary data. It can be found on the internet, in general literature, or in books. Secondary data is less important than primary data and is easier to gather. For analysis, both systems are still required.

I completed both data collection strategies to complete the system so that it may be built even better.

- **Data Analysis:**

The data from the first and second stages of information collection are compared. After you've gathered all of the data and compared and analyzed it, you may move on to the next step. We discovered some consequences, as one could expect from a poor country in the third world. Bangladesh's health system is reliant on roughly 100,000 registered doctors, and if these few doctors, in comparison to the population, are unable to deliver healthcare, the country's health system will suffer. That's why most people can't get treatment timely during COVID situation. This is the outcome of our little investigation. Due to restrictions, we are unable to conduct the research on a large scale.

1.7 Proposed System:

We are proposing an android base application which will provide advanced digital telemedicine services to the patients. First and foremost, the system is a mobile application native to android platform but it's also exchangeable to cross pulpit software as well. The intercommunication of each user with the application depends on who the end user is. By this application doctors and patients can communicate and patients can get their services.

Registration & Log In:

1.8 Conclusion:

We can envisage a possible outcome if this project is in the field. People are able to schedule appointments with their doctors. Patients who are unable to get to the hospital can still receive medical care. This also allows users to save thousands of hours of time. It will be possible to establish an ideal healthcare system.

Chapter 2

Requirement Analysis

2.1 Introduction :

In this situation, going to the hospital or doctor's chamber for taking medical services is very dangerous for people. It will increase the probability of covid 19 spreading and also create a high risk. In the lockdown situation people can not go to the doctors chamber. If we think from another perspective, we

waste a lot of time traveling to the doctor and getting appointments. It also kills people's valuable time. Some people do not have that good condition to go to the doctors' chamber for medical care.

2.2 Requirements analysis

To create a manual process to an online base process, we need to do some analysis and field research to understand the problem better and find problem's deep. To understand the problem we set up some online based research, and field research. We set up some questions for the doctor and also the patients. Through the survey, we find some big problems with this offline system. So, we noted the problem and tried to solve this offline base problem by an online based system.

We survey on 8 hospital from all over Bangladesh

- Evercare Hospital Dhaka
- Square Hospital Dhaka
- Rajshahi Medical College Hospital
- Asgar Ali Hospital
- Lab-aid Cardiac And Specialized Hospital
- Islami Bank Medical College Hospital Rajshahi
- Sylhet Shishu Clinic & General Hospital
- Apollo Hospital Dhaka

Following the Required system we have gathered from the survey.

Online Base Appointment: Most of the healthcare systems do not have an online base treatment system. For this people have to suffer, patients have to waste their valuable time. It is also difficult for those patients who can not go to the doctors chamber. In this case, an online base appointment and telemedicine system can reduce the problem and make patients live earlier.

Online Base Billing System: In our country, people have to go to the health care to get the medical services and also have to pay the bill offline. In this covid situation, the transaction of paper money can be very dangerous for people. But an online base billing system can be a good effective solution. By this we can reduce the covid spreading.

2.3 Conclusion

Based on the information we have received from the survey, these requirements are more important in the healthcare system. Managing all health care systems in a manual system is both time-consuming and dangerous. So even though we are not able to fulfill all the requirements mentioned above, we have given solutions to some of the major problems in this application such as Online base appointment, online base billing system, medical system. This will reduce the work of the manual activities as well as the patients will be much benefited.

Chapter 3

Literature review

3.1 Introduction

The literature review indicates the summary of the necessary information based on a particular topic on a scholarly paper. It ensures the complete understanding of a topic. It helps to identify similar work within the selected area, potential problems and which are needed for successful project development. So it plays an important role to develop a project.

3.2 Literature review

Nowadays many mobile applications are being developed to perform different types of academic activities. Which is used by many educational institutions worldwide. Some of the great contributions to this topic are mentioned below:

Umme Sayma Busra, Mohammad Zahidur Rahman: Proposed Mobile base telecommunication system. They concentrated on developing an Internet-based telemedicine architecture (ECG) to connect different hospitals and mobile medical professionals who work in rural health clinics.

Mafruha Alam, MPH, corresponding author Cathy Banwell, PhD,¹ Anna Olsen, PhD,¹ and Kamalini Lokuge, PhD¹. Telemedicine solutions based on mobile phones have been proposed for rural regions. And this is for folks who live in remote areas. Primary therapy will be used to diagnose them.

3.3 Research Methodology

According to the survey from the problem statement subchapter we have solved only Patients and doctor menial appointments and treatment systems. As we said this software is an online based application that means mobile data or internet connection must be established on the

device. The actual actors of our project are doctors and patients. The services of the system established by the participation of both sides. The services are: Online appointment system, Video calling system, Doctor category and doctors list.

Here doctors can share their services via video call. And patients have access to the doctors category and list. Where patients can find doctor information and also book appointments. Both can give and take services via application. Important factor is for getting all kinds of services users must be authenticated successfully.

Use Case 1: Login

Primary Actor: Patients and Doctor

Precondition: Must have registered Phone & Password in the system

Main Success Scenario:

1. The actor places the Phone Number in the Phone Number section
2. The actor places the password id in the password section
3. Must select the user type clicking on radio button
4. Press the Login button

Exception Scenario:

1. The Phone Number must be registered on the system
2. Provided Phone Number must be verified by the user
3. Incorrect password, password must be more than 6 characters

Use Case 2: Register

Primary Actor: Patients and Doctor

Precondition: Must have registered Phone Number

Main Success Scenario:

1. The actor must provide a clear profile picture
2. The actor must provide the academic information
3. The actor places the Phone Number in the Phone Number section
4. The actor places the password id in the password section
5. Must select the user type clicking on radio button

6. Press the Register button

Exception Scenario:

1. The Phone Number must be registered on the system
2. Provided Phone Number must be verified by the user
3. Incorrect password, password must be more than 6 characters
4. Must give the profile picture
5. Correct user type must be checked
6. Text can't be empty

For implementing the system UI, we have used XML (Extensible Markup Language) where we have made a combination of RecyclerView and Cardview where android material design is present. On the backend section we have used JAVA language because of object oriented. To store the information and contents we have used Firebase realtime features which is called "Realtime Database".

Actually, the backend of our project is divided into three parts, one server, one database and most importantly one application. If you book an air ticket in a travel agency, you usually open the ticket booking website and interact with the frontend because the frontend contains useful navigation to navigate a user. When the data user needs is inputted, the application saves it to a database created on a server. Therefore, in our back-end system we create a database to store the data that the server receives through the users. We used JAVA for our application development.

3.4 Conclusion

After analyzing all the things I have acquired a vast knowledge. It also helped to understand the issues regarding the problem domain and how to resolve the issues. Moreover it also helped me to understand the professional activities. It enhances my ability to find out the requirements of a solution.

Chapter 4

System Analysis and Design

4.1 Introduction

Systems Analysis and Design (SAD) is a term for describing system methodologies for developing quality automation systems. It combines information technology. It is a step-by-step procedure that comprises planning, analysis, design, deployment, and maintenance. It includes the development process as well as the system's future maintenance work. The basic SAD methodology is the waterfall model which is very easy to implement following the model steps. hence the focus is on programming.

It has some objectives. These are:

- Most importantly, it focuses on systems where the other minor systems may have clashing destinations and it also enables the intellect of critical structures.
- System analysis assists with accomplishing similarity and solidarity of the other systems.
- This kind of analysis gives a fortunate position of intelligence and contrasts the other systems sizes, capacities and complete systems.

4.2 System Analysis

It is a method of gathering and evaluating data, recognizing a problem, and dissecting an automated system into its constituent parts.

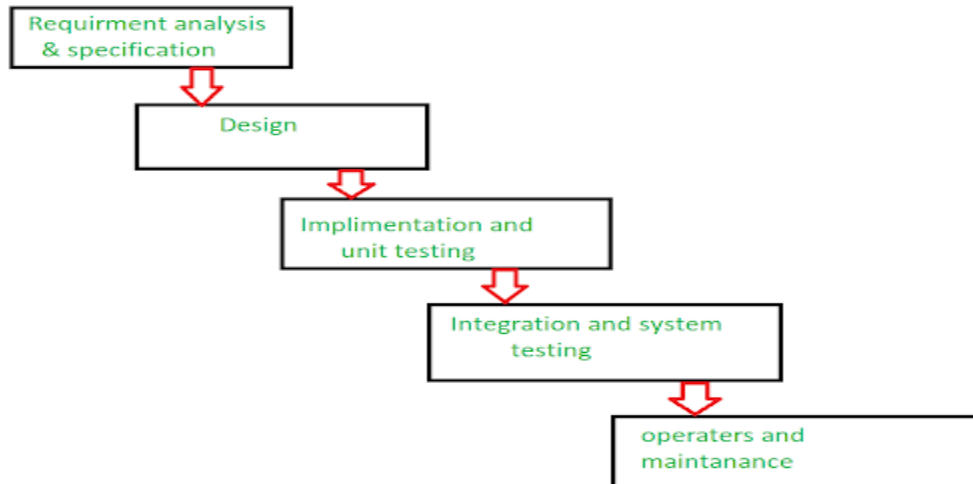
This kind of analysis is conducted for the purpose of studying a manual problem or its parts in order to identify its necessities or objectives. It is a problem-solving fetch that improves the method and ensures that all other components of the automation system work perfectly to accomplish their purpose.

The software development and life cycle are the most significant aspects of system analysis (SDLC). The Software Development Life Cycle (SDLC) is a method for producing high-quality software at the lowest feasible cost in the shortest amount of time. This life cycle provides an organized sequence of stages that assist an organization in producing well-tested software fast. SDLC may be described using a variety of models. However, we shall employ the Waterfall Model in our project.

WATERFALL MODEL is a follow-up or sequential model. This paradigm splits the creation of software into many phases. Because there is no overlap between the phases, each one must be finished before moving on to the next. During the life cycle phase, each phase is meant to accomplish certain jobs or activities. This model was introduced in 1970 by Winston Royce[4].

The primary advantage of this strategy is that each step of development must be finished before moving on to the next. Project, on the other side, is fully reliant on the project team, with limited customer involvement.

However, the issue or disadvantage in this paradigm is Developers and testers spend a lot of time on documentation. Small adjustments or flaws in the final program that occur throughout the deployment of this project may generate a slew of issues. Despite the fact that this model has several flaws, we have picked it. Because needs do not change regularly, applications are not as complex and large, requirements are clear, the environment is appropriate, and technology and tools are the most significant factors.



4.3 System Design

Systems design is that the procedure of defining components of a system like modules, architecture, elements and their user interfaces and knowledge for a system supports the required requirements. This is the method of defining, developing and designing methods which satisfy the actual needs and requirements of an application.

4.3.1 Block Diagram

When we represent the principal parts or functions of a system by blocks connected by lines is called a block diagram. This is a diagram of a system which is used to show the relationship of the blocks. It contains the operating of the system, what are its intakes or inputs according to the given inputs, what are the outputs and the most important thing is how the information, findings or materials flow through it. The block diagram of our application is presenting below-

This academic activity application has a client server architecture. All the information of the users is kept in the Real-time database server. Now the question is what a real-time database server is. This is a database system which uses a special way to handle the data and the special way is Real-time processing.

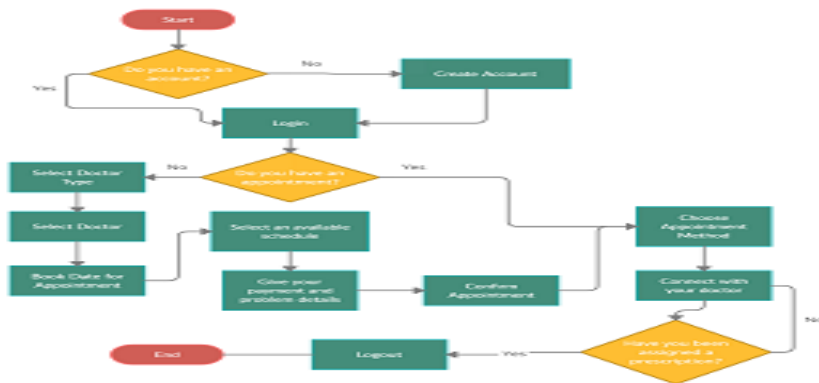
In this case if the state is constantly changing, there is no effect on database access. Anyway, this information can be accessed by users but before that they must install the application on their smartphone. Each client will observe a different UI (User Interface) depending on client type. That means, each category of user will have a different user interface on the basis of the authorization given upon him.

4.3.2 Flow chart

First of all, a proposed system needs a blueprint which helps to guide a programmer through the progression of the development. This blueprint is called flowchart. It's an excellent way to communicate the details of a system to others. Without drawing process flow development is impossible.

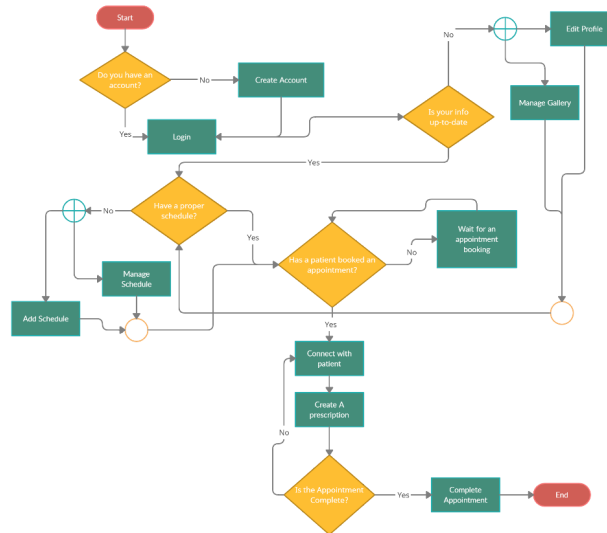
There is a different flowchart in our system depending on different modules. Each module has different conditions, process or flow. But most of the processes in different modules are almost the same according to logic. Here we are described two main module of our system-

4.3.2.1 Patients Module Flow-Chart



4.3.2.2 Doctors Module Flow-Chart

←



4.4 Conclusion

The whole chapter has been devoted to the system analysis and design that will be utilized to construct this application. This app's system study entails a thorough examination of a system in order to discover areas for improvement and, if required, create upgrades. When a system analysis is done correctly, it ensures that the appropriate route is taken in terms of applications and helps to decrease mistakes, which reduces future IT requirements for problem-solving. If the above process is followed correctly, it will not only save the company money in the short term, but it will also ensure that the correct application path is taken the first time, that growth and business charge considerations have been taken into account to accommodate future plans, and that errors are kept to a minimum, reducing the need for future IT overhauls, which is less expensive, efficient, and flexible. Proper system design, such as block diagrams and flowcharts, enables for improved administration by allowing the software to be changed to meet any business changes, resulting in a completely controlled end product. If updates or upgrades are needed, the requirement to rewrite the entire software will be eliminated, which is usually expensive.

Chapter 5

System Evolution/Development

5.1 Introduction

The process of defining, creating, or implementing a system based on system analysis and design is known as system evolution or system development. Because it is the general structure of how a system is produced or maintained, it is an important structure in the field of software development. Any information system's success is dependent on each phase of its development.

5.2 Front end development

Android Studio, Visual Studio, and Droid Script are just a few of the IDEs available for designing Android apps. We utilized Android Studio in our project. Because one of the greatest IDEs for Android programming is Android Studio. Google created this development tool, which has been praised by mobile app developers all around the world. The advantages of utilizing Android Studio as an Android development environment are numerous. Front-end and back-end development may be done in a fluid environment using this software.

5.2.1 User interface Design

XML was used to develop our project's interface. We're talking about the Extension Markup Language when we say xml. It's a markup language for data descriptions that's comparable to HTML. Humans and robots alike can readily read Xml. On Android, we use xml to create our layouts since it is a lightweight language that keeps our layouts light. In fact, the entire concept of xml is defined by the hierarchy of View and View Group objects.

A ViewGroup is a hidden container for arranging child views. These child views are additional widgets that are used to create various UI components. The user interface varies depending on the user. To showcase this application, we have two different sorts of interfaces. Because their jobs are distinct, the views of a patient's activity and a doctor's activity are completely different.

When a user registers for the service in this application, he must choose the kind of registration. This category will be used to categorize the user types. Overall, the user interface is fairly intuitive.

5.2.1.1 Patients Registration:

2:08 6.00 KBPS Wi-Fi 91%

Patient Signup

FullName

Email

Password

ConfirmPassword

Phone no

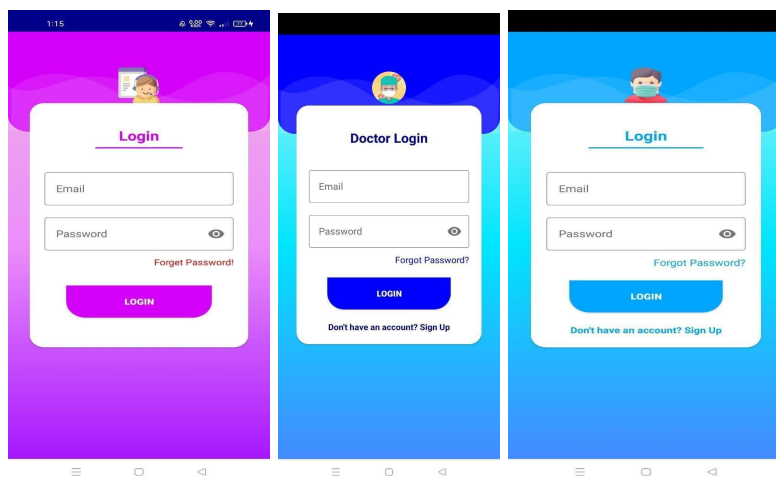
Photo



5.2.1.2 Doctors Registration:

The screenshot shows a mobile application interface for doctor registration. At the top, the title "Doctor Signup" is displayed in a bold, dark red font. Below the title, there is a list of input fields, each with a light gray border and a light gray background. The fields are labeled as follows: "Enter Name", "Email ID", "Password", "Confirm Password", "Phone No", "Basic Fees", "Experience", "Start Hour", "End Hour", and "Address". At the bottom of the form, there is a light gray button with the text "Select category first". The entire form is set against a light yellow background. The status bar at the top shows the time as 2:09 and various system icons.

5.2.1.3 Authentication



5.2.1.4 Dashboard Design:

XML Code:


```

<?xml version="1.0" encoding="utf-8"?>
<androidx.drawerlayout.widget.DrawerLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/drawer_layout"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:fitsSystemWindows="true"
    tools:openDrawer="start">

    <include
        layout="@layout/app_bar_doctor_home"
        android:layout_width="match_parent"
        android:layout_height="match_parent" />

    <com.google.android.material.navigation.NavigationView
        android:id="@+id/nav_view"
        android:layout_width="wrap_content"
        android:layout_height="match_parent"
        android:layout_gravity="start"
        android:fitsSystemWindows="true"
        app:headerLayout="@layout/nav_header_doctor_home"
        app:menu="@menu/activity_doctor_home_drawer" />

</androidx.drawerlayout.widget.DrawerLayout>

```

Fragment:

<pre> <?xml version="1.0" encoding="utf-8"?> <LinearLayout xmlns:android="http://schemas.android.com/apk/res/android" xmlns:app="http://schemas.android.com/apk/res-auto" android:id="@+id/nav_name" android:layout_width="match_parent" android:layout_height="@dimen/nav_header_height" android:background="@drawable/side_nav_bar" android:gravity="center" android:orientation="vertical" android:theme="@style/ThemeOverlay.AppCompat.Dark"> <LinearLayout android:layout_width="match_parent" android:layout_height="wrap_content" android:orientation="horizontal"> <androidx.cardview.widget.CardView android:id="@+id/myCardView" android:layout_marginLeft="10dp" android:layout_width="100dp" android:layout_height="100dp" app:cardCornerRadius="50dp" app:cardBackgroundColor="#00E5FF" android:layout_gravity="center"> <ImageView android:id="@+id/imageView_nav" android:layout_width="match_parent" android:layout_height="match_parent" android:scaleType="centerCrop" android:layout_gravity="center" /> </androidx.cardview.widget.CardView> </pre>	<pre> <LinearLayout android:layout_width="match_parent" android:layout_height="match_parent" android:orientation="vertical"> <TextView android:id="@+id/name_nav" android:layout_width="match_parent" android:layout_height="wrap_content" android:gravity="right" android:paddingRight="10dp" /> <TextView android:layout_marginTop="2dp" android:id="@+id/phone_nav" android:gravity="right" android:layout_width="match_parent" android:layout_height="wrap_content" android:textSize="13sp" android:textColor="#000000"/> <TextView android:gravity="right" android:id="@+id/mail_nav" android:layout_width="match_parent" android:layout_height="wrap_content" android:textColor="#000000" android:paddingRight="10dp" android:layout_marginTop="10dp" /> </LinearLayout> </LinearLayout> </LinearLayout> </pre>
--	---

Doctor Category Activity:

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
xmlns:android="http://schemas.android.com/apk/res/android"
xmlns:app="http://schemas.android.com/apk/res-auto"
xmlns:tools="http://schemas.android.com/tools"
android:layout_width="match_parent"
android:layout_height="match_parent"
android:orientation="vertical"
tools:context=".DoctorDepartmentListActivity">

<GridView
    android:id="@+id/gridView"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:layout_margin="10dp"
    android:horizontalSpacing="10dp"
    android:numColumns="3"
    android:verticalSpacing="10dp" />

</LinearLayout>
```

Fragment:

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.cardview.widget.CardView
xmlns:android="http://schemas.android.com/apk/res/android"
app:cardBackgroundColor="#ffffff"
android:layout_margin="5dp"
android:id="@+id/cardview1"
android:layout_width="match_parent"
android:layout_marginTop="10dp"
android:layout_marginLeft="15dp"
android:layout_marginRight="15dp"
app:cardElevation="10dp"
app:cardCornerRadius="10dp"
android:layout_height="wrap_content"
xmlns:app="http://schemas.android.com/apk/res-auto">

<LinearLayout
    android:orientation="horizontal"
    android:weightSum="2"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    >

<LinearLayout
    android:layout_width="1"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:orientation="horizontal">

<ImageView
    android:id="@+id/inv111"
    android:layout_width="0dp"
    android:layout_height="180dp"
    android:layout_margin="10dp"
    android:layout_weight="0.8"
    android:src="@drawable/abc" />
</LinearLayout>

<LinearLayout
    android:layout_width="1"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:orientation="vertical">

<TextView
    android:layout_weight="1"
    android:layout_width="match_parent"
    android:orientation="vertical">

<TextView
    android:textColor="#9C27B0"
    android:layout_marginTop="15dp"
    android:layout_marginLeft="10dp"
    android:textSize="20sp"
    android:text=""

    android:id="@+id/tv111"
    android:layout_width="match_parent"
    android:layout_height="wrap_content" />

<TextView
    android:textColor="@color/logo_dark"
    android:layout_marginTop="5dp"
    android:layout_marginLeft="10dp"
    android:textSize="18sp"
    android:text=""
    android:id="@+id/tv222"
    android:layout_width="match_parent"
    android:layout_height="wrap_content" />

<TextView
    android:textColor="@color/logo_dark"
    android:layout_marginTop="5dp"
    android:layout_marginLeft="10dp"
    android:textSize="12sp"
    android:text=""
    android:id="@+id/tv333"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content" />

<TextView
    android:textColor="@color/logo_dark"
    android:layout_marginTop="5dp"
    android:layout_marginLeft="10dp"
    android:textSize="12sp"
    android:text=""
    android:id="@+id/tv444"
    android:layout_width="match_parent"
    android:layout_height="wrap_content" />

<TextView
    android:textColor="@color/logo_dark"
    android:layout_marginTop="5dp"
    android:layout_marginLeft="10dp"
    android:textSize="12sp"
    android:text=""
    android:id="@+id/tv555"
    android:layout_width="match_parent"
    android:layout_height="wrap_content" />

<Button
    android:layout_marginTop="15dp"
    android:textColor="@color/white"
    android:layout_width="150dp"
    android:layout_height="match_parent"
    android:text="View details"
    android:textStyle="bold"
    android:background="@drawable/roundedbutton"
    android:id="@+id/bt1"
    android:layout_marginBottom="20dp"/>
</LinearLayout>
</LinearLayout>
</androidx.cardview.widget.CardView>
```

5.3 Back End development

For the back-end portion of our project, we employed the JAVA programming language. Java is a programming language that focuses on objects. It is a class-based general-purpose programming language that works on any platform. It necessitates as little dependencies as possible, which is rather impressive. The run-time code executes as compiled code within the Java Virtual Machine (JVM). Generally, The Android SDK and the JAVA programming language are used to develop Android applications (Software Development Kit). It is the most popular and commonly used programming language for Android App Development.

The major goal of the Android expansion application was to build a platform application environment that could operate on any device. It's incredibly simple to turn your software into a dynamic system. Because Java has this type of standard, it was picked for Android development.

5.3.1 Database Design

A database is a logically ordered collection of organized data stored in a computer system electronically. The security of databases and the safeguarding of personal information are both strictly controlled. The core of an automation system is a secure database. To store data in our app, we use a Firebase database behind a room database.

Firebase is a backend software system. It's a piece of software that lets users interact with a database in a relational format. It's a Google platform for creating interactive mobile apps and online applications. As a result, we may classify it as a Google backend application that delivers excellent services such as app crash resolution, analytics reporting, and tracking. A database in Firebase is kept in a single file, which sets it apart from other database engines.

The fundamental goal of adopting Firebase is to make the app as real-time as possible. Because Firebase is the original platform through which we may obtain real-time read and write services. Because the database table is stored in a hierarchical manner, accessing the main key is as simple as contacting the root data. It does, however, have several distinct components, such as authentication, storage, cloud firestore, and real-time database.

We've also included a report issue option leveraging the Firebase advantage. Any user who encounters a systemic problem or illegal bugs can report it to the developers. He must provide the subject's name and details for this. His ID will be sent with the report systematically with actual time and date inside the technique. The developer may identify the individual and solve the problem by utilizing the ID. We can simply describe the specifics in the following database snapshot.

These are used to handle the data very nicely. In this project we are using the following features:

- 1) Realtime storage
- 2) Realtime read and write data
- 3) Authentication system
- 4) Account Verification
- 5) Push notification

For further development we can use app messaging service from this firebase.

5.4 Conclusion

We learned about front-end design in this chapter, and we built back-end development based on the front-end design. For creating user interfaces, xml (Extension Markup Language) is used, which is a human language rather than a computer language. It's also easy to read and comprehend, even for beginners, and it's not that complicated to code. We utilize java for backend development in order to create the front-end components as user listeners. Java is a popular language because of its amazing capabilities and performance. The first and most significant benefit of utilizing Java for app development is that it supports OOPS (Object Oriented Programming) ideas and is more capable due to its scalability, extensibility, and flexibility.

Chapter 6

Testing and Implementation

6.1 Introduction

First and foremost, implementation refers to the process of putting a plan into action. The project plan is put into effect during the implementation phase. On the other hand, project testing after implementation is a critical step in ensuring the project's success.

6.2 Testing

Software testing is a technique for determining if the actual software product meets the expected criteria and ensuring that it is defect-free. In contrast to real requirements, software testing's goal is to find mistakes, gaps, and missing requirements.

In our project, the testing outcome is largely reliant on the database's proper transaction. As previously stated, students might use the question as the basis for their queries. As a result, anytime people search or post a query, the data in the database should be adjusted using SQL commands. Simultaneously, a powerful authentication system is installed. As a result, the testing outcome is nearly entirely dependent on the authentication system. During testing, all exceptions should work regardless of the mistake. All toasts and error messages must be shown flawlessly.

There are several aspects of our project that must be executed flawlessly in order for it to succeed. As an example, the authentication system, the fragment presented, the button listener's flawless listening, the intent from one activity to the next, and so on. The most crucial aspect is reliable internet access. Our system will not be able to access it without an Internet connection. The permission was obtained from the Android Studio manifest file. App messaging, quiz participation, and viewing previous questions all require internet access. Because we've already said that we're employing a live server for our real-time database. Users cannot access the database if their internet connection is down. As a consequence, they will not receive timely notification at the appropriate moment.

6.3 Implementation

We'll talk about how to put the testing results into action in this section. Incorrect Login Credentials, for example. It might, for example, be due to the administrator panel being accessed with the incorrect login account and password. If your login credentials have recently changed, you will be unable to access your database using the previous credentials. As a result, the first thing you should do is double-check that the login credentials you're using are correct. To create an account in our project, users must give some basic information as well as some unique information. An exception notice will appear if somebody refuses to supply the correct information or provides incorrect information. After completing the sign-up process, you will be asked to enter your username and password during the sign-in process. If someone provides false legitimate information, an exception notice will appear. If a user forgets their password, they can use the forget password option. The user must provide his or her email address, and an email will be sent to that address at the same time.

6.4 Result and Discussion

The project's outcome is determined by the results of all testing and the implementation phase. All of these occurrences, including question searching by proper query, preceding question with solution, and donation, are recorded directly in the database. Particularly in the case of a contribution system, the user must first register in, then supply the relevant data of the question in the contribution fragment, before browsing the question from his device's hard drive. After uploading to the database, it becomes visible to all users, allowing them to search for data or get information from the database. This is the application's real algorithm.

If the UI is straightforward and good, all of the tasks should be completed more clearly. The user interface in this project is highly user friendly, and there is also a user guide where the user can acquire all of the information about the system. There is also a link to the authority's contact action, which users may use to contact them if they have any problems using the program.

We already know about the room database from the last chapter, which uses convenience annotations to save repetitious and error-prone boilerplate code. This is a fantastic chance for the project's development. It eliminates the need for a developer to write a lot of it in order to establish and manage databases. SQL queries can also be validated at compile time. All of the notices are mentioned in another fragment class. Over CustomAdapter, all of the notifications will be shown in a listview.

Adapter is an Android component that connects a UI component to a data source and allows us to populate data into the UI component. It keeps the data and sends it to an Adapter view, which may then take the data from the adapter view and display it in other views like ListView, GridView, and so on. However, we can easily utilize ArrayAdapter in this case. Our major goal was to make the app as dynamic as possible, which is why we utilized CustomAdapter.

6.5 Conclusion

Overall, a number of essential project-related subjects have been clarified in this chapter. From the first subpoint, we're talking about functional and real-world software testing, and how it affects actual success. Functional bugs can usually be fixed, but real-world bugs cannot, which is an uncommon exception.

Mistakes can sometimes obscure further errors. That is why thorough testing is critical. The implementation process should be done after the module, according to the testing protocol. It was mentioned in the second point of this chapter. The real outcome of the implementation is shown in the attached screenshot. The smooth operating component is also discussed here, which brings an application up to par.

Chapter 7

Critical Appraisal

7.1 Introduction

The practice of thoroughly and methodically reviewing research to determine its trustworthiness, as well as its value and applicability in a specific context, is known as critical appraisal. In this chapter, a thorough study of this application is offered. Every project has its own set of constraints, flaws, strengths, and opportunities. Even so, there are a lot of threads to get to the application's target. In fact, this style of evaluation helps us to reduce information overload by removing unnecessary or weak research and assessing the study's value and clinical application. This type of analysis may be beneficial in recognizing our strengths and weaknesses, as well as identifying the Opportunities and, as a result, the Threats we face.

7.2 SWOT Analysis

SWOT Analysis is a basic yet powerful framework for analyzing our company's strengths, weaknesses, opportunities, and threats. It allows us to make the most of our resources, satisfy our needs, lower our risks, and increase our chances of success. SWOT Analysis can help us make the most of what we have to help the project succeed. We may lower our chances of failing by discovering what we're lacking and removing any threats that would otherwise catch us off guard. Even better, we might begin to develop a strategy that sets us apart from our competitors, allowing us to compete successfully in our market.

7.2.1 Strength

The system of the project is much more advanced than the competitor of the market. We analyze the market competitor application then we try to implement the process. This application will be ad free. Users can use this application without any hassle. The most strongest part of the system is the video calling system. We have added a video calling system via a third party application. By this this patient can get their medical services from home. Existing applications on the market do not have this feature at this time.

7.2.2 Weakness

Now it is time to consider our project weakness. We are using Firebase realtime database for our project data storage. Currently we are using a free version so we have limitations. In our country, firebase servers have many issues. Sometimes the data transfer rate is so slow. It can be a great issue in future.

7.2.3 Opportunity

Opportunities are windows of opportunity or opportunities for something good to happen, but we must seize them. Essentially, this project has the potential to propel the Bangladeshi educational system forward.

Now govt. Took many initiatives to create a platform for telemedicine. There will be huge opportunities for this project. Even A2I investing for project development in the telemedicine sector. This can be a huge opportunity .

7.2.4 Thread

Threats include anything that can gloomily affect our system from the outside, such as supply chain problems, shifts in market requirements, or a shortage of recruits. It's vital to expect threats and to take steps against them before we become a victim of them and our growth stalls.

In our country, most of the people are not experts in use of the internet or any application. If we count it in a percentage, it will be at least 55% people. This can be a great threat for this project.

Another thread is market competition. There are many giant IT companies who have enough capacity to adapt any new idea in a day and implement it in a week. That will be a great challenge for us in the copyright sector.

Another issue is a sluggish internet connection. True, there are many areas in this nation where an adequate internet connection is unavailable. Even in that isolated region, the internet is difficult to come by. Though we will be vigilant in using the least amount of bandwidth possible to run the program, it will be a significant problem for us because a weak signal might impair the application's flavor.

7.3 Conclusion

After all, we may claim that this SWOT analysis involves identifying a project's primary strengths. It gives you a sneak peak into the possibilities that lie ahead. We may utilize this to create critical growth plans that are centered on our limitations and strengths. A strong brand image, a large amount of working capital, a high consumer standing, and even robust distribution networks are examples of these qualities. A company's strength is essentially whatever advantage it has over its true competitors. Companies should, however, break down the strengths of their competitors as well, since this provides a more accurate picture of how a company will fare in the marketplace.

Furthermore, it encourages the creation of backup or optional plans, as well as emergency plans. We can address the flaws in the near future, and at the same time, we should consider competing with other tech giant platforms that will bolster our vulnerable areas, ward off attacks, and seize every opportunity.

In fact, the entire SWOT analysis approach highlights our assets and provides inspiration and motivation to continue our marketing tactics in the face of adversity.

Chapter 8

Conclusion

8.1 Conclusion

In this technology dependent world, since the advent of android the world of mobile application has witnessed a huge change. The major reasons behind that are android being an open-source OS. Though in every sector android apps are available, in our country it's actually absent in the healthcare sector. This automation system can undoubtedly change the outlook of traditional systems.

Health care system can not stop due to any situation. So. This project can ensure the health care system properly in the pandemic time. By this project, people do not need to go to the hospital or doctors' chambers. They can take their services from the home. In near future, we will develop this virtual healthcare system as more flexible by bringing incredible features.

Face to Face interaction is the best way in health care or prescription. But in this pandemic, this can be a friend of a patient. It will also save time and money for patients. Because this application can be minimum cost application services.

8.2 Further Suggested Work :

We have found some lacking in the current system. Now we are using the realtime database of firebase. In the future we will use a dedicated server for the project. Doctor and patients appointment reports have limitations after setting up the dedicated server that can store a large amount of data of appointment.

Now, in the project, we are using a third party video calling system. In the future we will integrate our own video calling system. On the other hand, this system is an android Operating system based application IOS user will be deprived of. So, in the near future we will develop this system into responsive web software. Moreover, for ensuring different mobile devices and variety of users we will develop this system. We will also work on the interface to make it more user friendly.

Reference

- [1] WHO [2019] Online available:
https://www.who.int/medical_devices/innovation/videoconferencing_telemedicine.pdf (May 21, 2019)
- [2] Mobile phone based telemedicine service for rural Bangladesh: ECG 29 December 2014
<https://ieeexplore.ieee.org/document/6997381>
- [3] Patients' and Doctors' Perceptions of a Mobile Phone–Based Consultation Service for Maternal, Neonatal, and Infant Health Care in Bangladesh: A Mixed-Methods Study
<https://www.ncbi.nlm.nih.gov/pmc/articles/PMC6658262/>
- [4] Software Testing Help, "What Is SDLC Waterfall Model?," 2021. [Online]. Available: <https://www.softwaretestinghelp.com/what-is-sdlc-waterfall-model/>. [Accessed 10 August 2021].

Appendix

Appendix -A Backend Code

This part is an extended part of chapter 5.3: Back-end development. Here we have shown the actual java code of a specific task.

Doctor Registration:

```
package com.example.digi_doctor;
```

```
import android.app.ProgressDialog;  
import  
android.content.ContentResolver;  
import android.content.Intent;  
import android.graphics.Bitmap;  
import android.net.Uri;  
import android.os.Bundle;  
import android.provider.MediaStore;  
import android.view.View;  
import android.webkit.MimeTypeMap;
```

```
import  
android.widget.AdapterView;  
import  
android.widget.AdapterView  
;  
import  
android.widget.EditText;  
import  
android.widget.ImageView;  
import  
android.widget.Spinner;  
import android.widget.Toast;  
  
import  
androidx.annotation.NonNull  
;  
import  
androidx.appcompat.app.App  
CompatActivity;
```

```
rence;  
import  
com.google.firebase.d  
atabase.FirebaseData  
base;  
import  
com.google.firebase.s  
torage.FirebaseStorag  
e;  
import  
com.google.firebase.s  
torage.OnProgressList  
ener;  
import  
com.google.firebase.s  
torage.StorageRefere  
nce;  
import  
com.google.firebase.s  
torage.UploadTask;
```

```
import  
java.io.IOException;  
import  
java.util.ArrayList;  
  
public class  
DoctorSignupActivity  
extends  
AppCompatActivity {  
    Uri FileUri;  
    StorageReference  
storageReference;  
    DatabaseReference  
databaseReference;
```

```

DatabaseReference
booking;
int
Image_Request_Code
=7;
ProgressDialog
progressDialog;

EditText et1, et2,
et3, et4, et5, et6, et7,
et9, et10, et_address;

ImageView imv1;
ArrayList<String> al;

```

```
Spinner sp1;
```

```

ArrayAdapter<String> ad;
String drcategory = "";
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_doctor_signup);

```

```

        storageReference =
        FirebaseStorage.getInstance().getReference("doctorimages");
        databaseReference =
        FirebaseDatabase.getInstance().getReference("DoctorDetails");
        progressDialog = new ProgressDialog(DoctorSignupActivity.this);
        et1 = findViewById(R.id.et1);
        et2 = findViewById(R.id.et2);
        et3 = findViewById(R.id.et3);
        et4 = findViewById(R.id.et4);
        et5 = findViewById(R.id.et5);
        et6 = findViewById(R.id.et6);
        et7 = findViewById(R.id.et7);
        et9 = findViewById(R.id.et9);
        et10 = findViewById(R.id.et10);
        et_address = findViewById(R.id.et_address);

```

```

        al = new ArrayList<>();
        al.add("Select category first");
        al.add("Allergists");
        al.add("Dermatologists");
        al.add("Infectious Disease Doctor");
        al.add("Ophthalmologists");
        al.add("Gynecologists");
        al.add("Cardiologists");
        al.add("Endocrinologists");
        al.add("Gastroenterologists");
        al.add("Nephrologists");
        al.add("Urologists");
        al.add("Pulmonologists");
        al.add("Neurologists");
        al.add("Ent");

        spi = findViewById(R.id.spi);
        imv1 = findViewById(R.id.imv1);
        ad = new ArrayAdapter<>(this,
R.layout.support_simple_spinner_dropdown_item, al);
        spi.setAdapter(ad);
        spi.setOnItemClickListener(new AdapterView.OnItemClickListener()

```

```

{
    @Override
    public void onItemClick(AdapterView<?> parent, View view, int
position, long id) {
        Toast.makeText(DoctorSignupActivity.this, "" +
al.get(position).toString(), Toast.LENGTH_SHORT).show();
        drcategory = al.get(position).toString();
    }
}

```

```

@Override
public void onNothingSelected(AdapterView<?> parent) {

}

});

}

//Choose Photo From Gallery
public void go2(View view)
{
    Intent intent = new Intent();
    intent.setType("image/*");
    intent.setAction(Intent.ACTION_GET_CONTENT);
    startActivityForResult(Intent.createChooser(intent, "Select Image"),
Image Request Code);
}

```

```

// signup
public void go3(View view) {

    if (et1.getText().toString().trim().length() == 0) {
        et1.setError("FullName is Required");
        et1.requestFocus();
    } else if (et2.getText().toString().trim().length() == 0) {
        et2.setError("Email is Required");
        et2.requestFocus();
    }
    else if (et3.getText().toString().trim().length() == 0) {
        et3.setError("Password is Required");
        et3.requestFocus();
    }
    else if
(!et4.getText().toString().trim().equals(et3.getText().toString().trim())) {
        et4.setError("Confirm password Not match");
        et4.requestFocus();
    }
    else if (et5.getText().toString().trim().length() == 0) {
        et5.setError("Phone no is required");
        et5.requestFocus();
    }
}
}

```

```

else if (et6.getText().toString().trim().length() == 0) {
    et6.setError("Required");
    et6.requestFocus();

} else if (et7.getText().toString().trim().length() == 0) {
    et7.setError("Required");
    et7.requestFocus();

} else if (et9.getText().toString().trim().length() == 0) {
    et9.setError("Required");
    et9.requestFocus();

} else if (et10.getText().toString().trim().length() == 0) {
    et10.setError("Required");
    et10.requestFocus();

} else if (et_address.getText().toString().trim().length() == 0) {
    et_address.setError("Required");
    et_address.requestFocus();

}

else{
    UploadImage();
}

```

```

@Override
protected void onActivityResult(int requestCode, int resultCode, Intent
data) {

    super.onActivityResult(requestCode, resultCode, data);

    if (requestCode == Image Request Code && resultCode == RESULT_OK &&
data != null && data.getData() != null) {

        FilePathUri = data.getData();

        try {
            Bitmap bitmap =
MediaStore.Images.Media.getBitmap(getContentResolver(), FilePathUri);
            imv1.setImageBitmap(bitmap);
        }
    }
}

```

```

        catch (IOException e) {

            e.printStackTrace();
        }
    }

    public String GetFileExtension(Uri uri) {

        ContentResolver contentResolver = getContentResolver();
        MimeTypeMap mimeTypeMap = MimeTypeMap.getSingleton();
        return
mimeTypeMap.getExtensionFromMimeType (contentResolver.getType

    }

    public void UploadImage () {

        if (FilePathUri != null) {

            progressDialog.setTitle("Image is Uploading...");
            progressDialog.show();
            StorageReference storageReference2 =
storageReference.child(System.currentTimeMillis() + "." +

```

```

GetFileExtension(FilePathUri));
            storageReference2.putFile(FilePathUri)
                .addOnSuccessListener(new
OnSuccessListener<UploadTask.TaskSnapshot>() {
                @Override
                public void onSuccess(UploadTask.TaskSnapshot
taskSnapshot) {

                    final String
Fullname, Email, Password, PhoneNo, BasicFees, Experience, Service, StartHour, EndHour
, Address;

                    Fullname = et1.getText().toString().trim();
                    Email = et2.getText().toString().trim();
                    Password = et3.getText().toString().trim();
                    PhoneNo = et5.getText().toString().trim();
                    BasicFees = et6.getText().toString().trim();
                    Experience = et7.getText().toString().trim();
                    Service="service";
                    StartHour = et9.getText().toString().trim();
                    EndHour = et10.getText().toString().trim();
                    Address = et_address.getText().toString().trim();
                    progressDialog.dismiss();

```

```

        Toast.makeText(DoctorSignupActivity.this, "Signup
Success", Toast.LENGTH_SHORT).show();
        @SuppressWarnings("VisibleForTests")
        Task<Uri> uriTask =
taskSnapshot.getStorage().getDownloadUrl();
        while (!uriTask.isSuccessful());
        Uri downloadUrl = uriTask.getResult();
        String key = databaseReference.push().getKey();
        Doctor details obj = new Doctor details(Fullname,
Email, Password, category, PhoneNo, BasicFees, Experience, Service,
StartHour, EndHour, downloadUrl.toString(), key, "pending", Address);
        databaseReference.child(key).setValue(obj);
        FirebaseDatabase firebaseDatabase =
FirebaseDatabase.getInstance();
        booking =
firebaseDatabase.getReference("BookedSlots");
        booking.setValue(key);

```

```

        finish();
    }
    }).addOnFailureListener(new OnFailureListener() {
        @Override
        public void onFailure(@NonNull Exception e) {
            progressDialog.dismiss();
            Toast.makeText(DoctorSignupActivity.this, "Failed
"+e.getMessage(), Toast.LENGTH_SHORT).show();
        }
    }).addOnProgressListener(new
OnProgressListener<UploadTask.TaskSnapshot>() {
        @Override
        public void onProgress(UploadTask.TaskSnapshot taskSnapshot) {
            double progres time =
(100.0*taskSnapshot.getBytesTransferred()/taskSnapshot.getTotalByteCount());
            progressDialog.setMessage("Uploaded "+(int)progres time+"
%");
        }
    });
    }
    else {
        Toast.makeText(DoctorSignupActivity.this, "Please Select Image or
Add Image Name", Toast.LENGTH_LONG).show();
    }
}

```


Patients Registration:

```
package com.example.digi.doctor;

import androidx.appcompat.app.AppCompatActivity;

import android.app.ProgressDialog;
import android.content.ContentResolver;

import android.content.Intent;

import android.graphics.Bitmap;
import android.net.Uri;
import android.os.Bundle;

import android.provider.MediaStore;
import android.view.View;
import android.webkit.MimeTypeMap;
import android.widget.Button;
import android.widget.EditText;
import android.widget.ImageView;
import android.widget.TextView;
import android.widget.Toast;
```

```

import com.google.android.gms.tasks.OnSuccessListener;

import com.google.android.gms.tasks.Task;
import com.google.firebase.database.DatabaseReference;
import com.google.firebase.database.FirebaseDatabase;
import com.google.firebase.storage.FirebaseStorage;
import com.google.firebase.storage.StorageReference;
import com.google.firebase.storage.UploadTask;

import java.io.IOException;

public class PatientSignupActivity extends AppCompatActivity {
    TextView tv1,tv2;
    Button Bt1,Bt2,Bt3;
    EditText et1,et2,et3,et4,et5;
    ImageView imv1;
    Uri FilePathUri;

    StorageReference storageReference;
    DatabaseReference databaseReference;
    int Image Request Code = 7;
    ProgressDialog progressDialog ;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_patient_signup);

        storageReference = FirebaseStorage.getInstance().getReferenceFromUrl("gs://myapp-1234567890.firebaseio.com/images");
        databaseReference =
        FirebaseDatabase.getInstance().getReference();
        progressDialog = new ProgressDialog(this);

        tv1= findViewById(R.id.tv1);
        tv2=findViewById(R.id.tv2);
        et1=findViewById(R.id.et1);
        et2=findViewById(R.id.et2);
        et3=findViewById(R.id.et3);
        et4=findViewById(R.id.et4);
        et5=findViewById(R.id.et5);
        Bt1=findViewById(R.id.Bt1);
        Bt2=findViewById(R.id.Bt2);
        Bt3=findViewById(R.id.Bt3);
        imv1=findViewById(R.id.imv1);
    }
}

```

```

//for Gallery
public void go2(View view)
{
    Intent intent = new Intent();
    intent.setType("image/*");
    intent.setAction(Intent.ACTION_GET_CONTENT);
    startActivityForResult(Intent.createChooser(intent, "Select Image"),
Image_Request_Code);
}
@Override
protected void onActivityResult(int requestCode, int resultCode, Intent
data) {

    super.onActivityResult(requestCode, resultCode, data);

    if (requestCode == Image_Request_Code && resultCode == RESULT_OK &&
data != null && data.getData() != null) {

        FileUri = data.getData();

        try {
            Bitmap bitmap =
MediaStore.Images.Media.getBitmap(getContentResolver(), FileUri);
            imageView.setImageBitmap(bitmap);
        }
        catch (IOException e) {

```

```

        e.printStackTrace();
    }
}

public String GetFileExtension(Uri uri) {
    ContentResolver contentResolver = getContentResolver();
    MimeTypeMap mimeTypeMap = MimeTypeMap.getSingleton();
    return
mimeTypeMap.getExtensionFromMimeType (contentResolver.getType (uri) ) ;
}

public void go3(View view) {

    if (et1.getText().toString().trim().length() == 0) {
        et1.setError("FullName is Required");

        et1.requestFocus();
    } else if (et2.getText().toString().trim().length() == 0) {
        et2.setError("Email is Required");
        et2.requestFocus();
    }
    if (et3.getText().toString().trim().length() == 0) {
        et3.setError("Password is Required");
        et3.requestFocus();
    } else if (et5.getText().toString().trim().length() == 0) {
        et5.setError("Phone no is required");
        et5.requestFocus();
    }

    else {
        UploadImage();
    }
}
}

```

```

public void UploadImage () {

    if (FilePathUri != null) {
        progressDialog.setTitle("Image is Uploading...");
        progressDialog.show();
        StorageReference storageReference2 =
storageReference.child(System.currentTimeMillis() + "." +
GetFileExtension(FilePathUri));

```

```

        storageReference2.putFile(FilePathUri)
            .addOnSuccessListener(new
OnSuccessListener<UploadTask.TaskSnapshot>() {
            @Override
            public void onSuccess(UploadTask.TaskSnapshot
taskSnapshot) {

                final String Fullname, Email, Password, PhoneNo;
                Fullname = et1.getText().toString();
                Email = et2.getText().toString();
                Password = et3.getText().toString();
                PhoneNo = et5.getText().toString();
                progressDialog.dismiss();
                Toast.makeText(getApplicationContext(), "Image
Uploaded Successfully ", Toast.LENGTH_LONG).show();
                @SuppressWarnings("VisibleForTests")
                Task<Uri> uriTask =
taskSnapshot.getStorage().getDownloadUrl();
                while (!uriTask.isSuccessful());
                Uri downloadUrl = uriTask.getResult();
                String key = databaseReference.push().getKey();
                PatientDetails obj = new PatientDetails(Fullname,
Email, Password, PhoneNo, downloadUrl.toString(), key);
                databaseReference.child(key).setValue(obj);
                finish();

```

```

            }
        });
    }
    else {

        Toast.makeText(PatientSignupActivity.this, "Please Select Image or
Add Image Name", Toast.LENGTH_LONG).show();

    }
}
}
}

```

Doctor's Department List Activity :

```
package com.example.digi_doctor;

import androidx.appcompat.app.AppCompatActivity;

import android.app.ProgressDialog;
import android.content.ContentResolver;

import android.content.Intent;

import android.graphics.Bitmap;
import android.net.Uri;
import android.os.Bundle;

import android.provider.MediaStore;
import android.view.View;
import android.webkit.MimeTypeMap;
import android.widget.Button;
import android.widget.EditText;
import android.widget.ImageView;
import android.widget.TextView;
import android.widget.Toast;
```

```

import com.google.android.gms.tasks.OnSuccessListener;

import com.google.android.gms.tasks.Task;
import com.google.firebase.database.DatabaseReference;
import com.google.firebase.database.FirebaseDatabase;
import com.google.firebase.storage.FirebaseStorage;
import com.google.firebase.storage.StorageReference;
import com.google.firebase.storage.UploadTask;

import java.io.IOException;

public class PatientSignupActivity extends AppCompatActivity {
    TextView tv1,tv2;
    Button Bt1,Bt2,Bt3;
    EditText et1,et2,et3,et4,et5;
    ImageView imv1;
    Uri FilePathUri;
    StorageReference storageReference;
    DatabaseReference databaseReference;
    int Image Request Code = 7;
    ProgressDialog progressDialog ;

```

```

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

    storageReference = FirebaseStorage.getInstance().getReference("Patient
images");
    databaseReference =
FirebaseDatabase.getInstance().getReference("PatientDetails");
    progressDialog = new ProgressDialog(PatientSignupActivity.this);

    tv1= findViewById(R.id.tv1);
    tv2=findViewById(R.id.tv2);
    et1=findViewById(R.id.et1);
    et2=findViewById(R.id.et2);
    et3=findViewById(R.id.et3);
    et4=findViewById(R.id.et4);
    et5=findViewById(R.id.et5);
    Bt1=findViewById(R.id.Bt1);
    Bt2=findViewById(R.id.Bt2);
    Bt3=findViewById(R.id.Bt3);
    imv1=findViewById(R.id.imv1);
}

```

```

//for Gallery
public void go2(View view)
{
    Intent intent = new Intent();
    intent.setType("image/*");
    intent.setAction(Intent.ACTION_GET_CONTENT);
    startActivityForResult(Intent.createChooser(intent, "Select Image"),
Image Request Code);
}
@Override
protected void onActivityResult(int requestCode, int resultCode, Intent
data) {

    super.onActivityResult(requestCode, resultCode, data);

    if (requestCode == Image Request Code && resultCode == RESULT_OK &&
data != null && data.getData() != null) {

        FilePathUri = data.getData();

        try {
            Bitmap bitmap =
MediaStore.Images.Media.getBitmap(getContentResolver(), FilePathUri);
            imv1.setImageBitmap(bitmap);
        }
}
}

```



```

        catch (IOException e) {
            e.printStackTrace();
        }
    }
}

public String GetFileExtension(Uri uri) {
    ContentResolver contentResolver = getContentResolver();
    MimeTypeMap mimeTypeMap = MimeTypeMap.getSingleton();
    return
mimeTypeMap.getExtensionFromMimeType (contentResolver.getType (uri)) ;
}

public void go3(View view) {
    if (et1.getText().toString().trim().length() == 0) {

```

```

        et1.setError("FullName is Required");
        et1.requestFocus();
    } else if (et2.getText().toString().trim().length() == 0) {
        et2.setError("Email is Required");
        et2.requestFocus();
    }
    if (et3.getText().toString().trim().length() == 0) {
        et3.setError("Password is Required");
        et3.requestFocus();
    } else if (et5.getText().toString().trim().length() == 0) {
        et5.setError("Phone no is required");
        et5.requestFocus();
    }
}

else {
    UploadImage();
}
}
}

```

```

public void UploadImage() {
    if (FilePathUri != null) {
        progressDialog.setTitle("Image is Uploading...");
        progressDialog.show();
    }
}

```

```

StorageReference storageReference2 =
storageReference.child(System.currentTimeMillis() + "." +
GetFileExtension(FilePathUri));
storageReference2.putFile(FilePathUri)
.addOnSuccessListener(new
OnSuccessListener<UploadTask.TaskSnapshot>() {
@Override
public void onSuccess(UploadTask.TaskSnapshot
taskSnapshot) {

final String Fullname, Email, Password, PhoneNo;
Fullname = et1.getText().toString();
Email = et2.getText().toString();
Password = et3.getText().toString();
PhoneNo = et5.getText().toString();
progressDialog.dismiss();
Toast.makeText(getApplicationContext(), "Image
Uploaded Successfully ", Toast.LENGTH_LONG).show();
@SuppressWarnings("VisibleForTests")
Task<Uri> uriTask =
taskSnapshot.getStorage().getDownloadUrl();
while (!uriTask.isSuccessful()) ;
Uri downloadUrl = uriTask.getResult();
String key = databaseReference.push().getKey();
PatientDetails obj = new PatientDetails(Fullname,
Email, Password, PhoneNo, downloadUrl.toString(), key);
databaseReference.child(key).setValue(obj);

```

```

finish();
}
});
}
else {

Toast.makeText(PatientSignupActivity.this, "Please Select Image or
Add Image Name", Toast.LENGTH_LONG).show();

}
}
}

```

Book Appointment System:

```
package com.example.digi_doctor;

import androidx.annotation.NonNull;
import androidx.appcompat.app.AppCompatActivity;

import android.annotation.SuppressLint;
import android.app.DatePickerDialog;
import android.content.Intent;
import android.graphics.Color;
import android.os.Bundle;
import android.util.Log;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.AdapterView;
import android.widget.AdapterView.OnItemClickListener;
import android.widget.BaseAdapter;
import android.widget.Button;
import android.widget.DatePicker;
import android.widget.GridView;
import android.widget.TextView;
import android.widget.Toast;
```

```
import com.google.firebase.database.DataSnapshot;
import com.google.firebase.database.DatabaseError;
import com.google.firebase.database.DatabaseReference;
import com.google.firebase.database.FirebaseDatabase;
import com.google.firebase.database.ValueEventListener;
```

```
import java.text.SimpleDateFormat;
import java.util.ArrayList;
import java.util.Calendar;
import java.util.HashSet;
import java.util.Locale;
import java.util.Set;

public class BookAppointmentActivity extends AppCompatActivity {
    ArrayList<String> al;
    //Spinner spl;

    ArrayAdapter<String> ad;
    BookAppointmentActivity.myadapter2 myad;
    ArrayList<slotdetails> all;
    String day = " ";
```

```

String did;
String finder;

//
TextView showdate;
ArrayList<String> mylist;
TextView daySelected;
TextView stayAlert;
String selecteddate;
ArrayList<leaveDetails> leavelist;
Set<String> set;
DatabaseReference leaveref;
DatabaseReference mainref;
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_book_appointment);
    showdate=findViewById(R.id.showDate);
    // Toast.makeText(this, "RECEIVED>>>", Toast.LENGTH_SHORT).show();
    stayAlert=findViewById(R.id.showAlert);
    daySelected=findViewById(R.id.daySelected);

```

```

        ArrayList<> al;
        mylist=new ArrayList<>();

        // getting the doctor id from ONlickBookingAppointment
        Intent in = getIntent();
        did = in.getStringExtra("d key");
        // Toast.makeText(this, "---$--->" + did, Toast.LENGTH_SHORT).show();

        FirebaseDatabase firebaseDatabase= FirebaseDatabase.getInstance();
        mainref = firebaseDatabase.getReference("DoctorSlots").child(did);
        // FirebaseDatabase FirebaseDatabase= FirebaseDatabase.getInstance();
        leaveref = firebaseDatabase.getReference("LeaveDetail");

        GridView lvsat = findViewById(R.id.search);
        myad=new BookAppointmentActivity.mvadapter2();
        lvsat.setAdapter(myad);

```

```

        al = new ArrayList<>();
        al.add("Select day of the week");
        al.add("Monday");

```

```

        al.add("Tuesday");
        al.add("Wednesday");
        al.add("Thursday");
        al.add("Friday");
        al.add("Saturday");
        al.add("Sunday");

        // spl = findViewById(R.id.spl);
        // ad = new ArrayAdapter<>(this,
R.layout.support_simple_spinner_dropdown_item, al);

```

```

        /* spl.setAdapter(ad);
        spl.setOnItemClickListener(new AdapterView.OnItemClickListener()
        {
            @Override
            public void onItemClick(AdapterView<?> parent, View view, int
            position, long id) {
                day = al.get(position); // get day name

                mainref.orderByChild("day").equalTo(day).addValueEventListener(new
                ValueEventListener() {
                    @Override
                    public void onDataChange(@NonNull DataSnapshot
                    dataSnapshot) {
                        Log.d("datasnap", dataSnapshot.toString());

                        all.clear();
                        for (DataSnapshot sin : dataSnapshot.getChildren())
                        {
                            slotdetails bj = sin.getValue(slotdetails.class);
                            all.add(bj);
                        }
                        myad.notifyDataSetChanged();
                    }
                });
            }
        });
    }
}

```

```

        @Override
        public void onCancelled(@NonNull DatabaseError
        databaseError) {
        }
    }
}

```

```

@Override
public void onNothingSelected(AdapterView<?> parent) {

}

})*

// creating a new function to get name

mylist.add("Select day of the week");
mylist.add("Sunday");
mylist.add("Monday");
mylist.add("Tuesday");
mylist.add("Wednesday");
mylist.add("Thursday");
mylist.add("Friday");
mylist.add("Saturday");

final Calendar myCalendar= Calendar.getInstance();

final DatePickerDialog.OnDateSetListener date= new
DatePickerDialog.OnDateSetListener() {
@Override
public void onDateSet(DatePicker view, int year, int month, int

```

```

dayOfMonth) {
myCalendar.set(android.icu.util.Calendar.YEAR, year);
myCalendar.set(android.icu.util.Calendar.MONTH, month);
myCalendar.set(android.icu.util.Calendar.DAY_OF_MONTH, dayOfMonth);

Label();
}

@Override
private void Label() {

```

```

String myFormat = "dd-MM-yyyy";
SimpleDateFormat sdf = new SimpleDateFormat(myFormat,
Locale.US);

int dayOfWeek = myCalendar.get(Calendar.DAY_OF_WEEK);
day=mylist.get(dayOfWeek); // getting the day with date;
daySelected.setText(day);
//Storing the date in variable
selecteddate=sdf.format(myCalendar.getTime());
showdate.setText(selecteddate);

stayAlert.setTextColor(android.R.color.white);
// check for leave

// checking for the existing slots
getbookedslots();

// //
// //

mainref.orderByChild("day").equalTo(day).addValueEventListener(new

```

```

@Override
public void onDataChange(@NonNull DataSnapshot
dataSnapshot) {

    Log.d("datasnap", dataSnapshot.toString());

    all.clear();
    stayAlert.setText("");
    for (DataSnapshot sin : dataSnapshot.getChildren()) {
        slotdetails bj = sin.getValue(slotdetails.class);
        if (set.contains(bj.getSlotstart()))
        {
            // skip the value if the there is booked slot
        }
        else {
            all.add(bj);
        }
    }

    if (all.size() == 0) // if there is no booking schedule
then
    { stayAlert.setText("No Schedule for Today");
      // stayAlert.setTextColor(Color.RED);
    }
}

```

```

    Toast.makeText(BookAppointmentActivity.this, "Yeh works
--", Toast.LENGTH_SHORT).show();
    myad.notifyDataSetChanged();

}

@Override
public void onCancelled(@NonNull DatabaseError
databaseError) {
    Toast.makeText(BookAppointmentActivity.this, "Check For
Internet Connection", Toast.LENGTH_SHORT).show();
}

Log.d("MYMSG", "going to run");
getleavefunction();
/* if (leavelist.size() == 0) {
    Toast.makeText(BookAppointmentActivity.this, "HelloWorld",
Toast.LENGTH_SHORT).show();
}
else
{
}
stayAlert.setTextColor(Color.RED);
*/ stayAlert.setTextColor(Color.RED);
}
}

```

```

        showdate.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                //datePickerDialog.getDatePicker().setMinDate(System.currentTimeMillis()-1000);
                DatePickerDialog datePickerDialog=new
                DatePickerDialog(BookAppointmentActivity.this,
                date,myCalender.get(android.icu.util.Calendar.YEAR),
                myCalender.get(android.icu.util.Calendar.MONTH),
                myCalender.get(android.icu.util.Calendar.DAY_OF_MONTH));
                datePickerDialog.getDatePicker().setMinDate(System.currentTimeMillis()-1000);
                datePickerDialog.show();
            }
        });
    }
}

```

```

// // //
private void getbookedslots() {
    Log.d("MYMSG", "I am here");
    set= new HashSet<>();
    //String finderkey=did+"-"+selecteddate; // making the key for
checking
    FirebaseDatabase firebaseDatabase= FirebaseDatabase.getInstance();
    DatabaseReference booked =
    firebaseDatabase.getReference("BookedSlots");
    DatabaseReference bookeddoctor =booked.child(did);
    Log.d("MYMSG", "start");
    bookeddoctor.orderByChild("date").equalTo(selecteddate).addValueEventListener(
    new ValueEventListener() {
        @Override
        public void onDataChange(@NonNull DataSnapshot dataSnapshot) {
            set.clear();
            for (DataSnapshot sin : dataSnapshot.getChildren()) {
                Log.d("MYMSG", "searching Successfully");
                Log.d("MYMSG", sin+"");
                BookedSlots bj = sin.getValue(BookedSlots.class);
                Log.d("MYMSG", "Object created Successfully");
            }
            String start=sin.getStart();
            set.add(start);
        }
    });
}
}

```

```

@Override
public void onCancelled(@NonNull DatabaseError databaseError) {
}
});
}

public void getleavefunction() {
    leavelist=new ArrayList<>();
    finder=did+"-"+selecteddate;
    Log.d("MYMSG", finder);
    leaveref.addListenerForSingleValueEvent(new ValueEventListener() {
        @Override
    }
}
}

```



```

public void onDataChange(@NonNull DataSnapshot dataSnapshot) {
    leavelist.clear();
    Log.d("MYMSG", "Entering");

    for (DataSnapshot sin : dataSnapshot.getChildren()) {
        leaveDetails bj = sin.getValue(leaveDetails.class);
        Log.d("MYMSG", sin+" "+sin.getKey());
        if (bj.getIdwithdate().equals(finder)) {
            leavelist.add(bj);
            Log.d("MYMSG", "leave object added Successfully");
            Log.d("MYMSG", "Runned Successfully");
            stayAlert.setText(" Sorry, Doctor is on Leave Today ");
            //stayAlert.setTextColor(Color.RED);
        }
        Log.d("MYMSG", leavelist.size()+"");
    }
    Log.d("MYMSG", "Exit");
}

@Override
public void onCancelled(@NonNull DatabaseError databaseError) {
}
}

```

```

}
});

}

//INNER CLASS
class myadapter2 extends BaseAdapter
{
    @Override
    public int getCount() {
        return all.size();
    }

    @Override
    public Object getItem(int i) {
        return all.get(i);
    }

    @Override
    public long getItemId(int i) {
        return i*10;
    }
}
}

```

```

@Override
public View getView(int position, View convertView, ViewGroup parent) {
    LayoutInflater inflater = LayoutInflater.from(parent.getContext());
    convertView = inflater.inflate(R.layout.patienttimeslotdesign, parent, false);

    final slotdetails obj = all.get(position);
    TextView tv1, tv2;
    Button B1;

    tv1 = convertView.findViewById(R.id.tv1);
    tv2 = convertView.findViewById(R.id.tv2);
    B1 = convertView.findViewById(R.id.B1);

    tv1.setText("From : "+obj.slotstart);
    tv2.setText("To : "+obj.slotend);
    B1.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
            Intent intent = new Intent(getApplicationContext(), ConfirmAppointmentActivity.class);
            intent.putExtra("Start", obj.slotstart);

```

```

            intent.putExtra("End", obj.slotend);
            intent.putExtra("DoctorId", obj.did);
            intent.putExtra("Day", obj.day);
            intent.putExtra("date", selecteddate);

            startActivity(intent);
        }
    });
    return convertView;
}
}
}
}

```

Confirm Appointment Activity:

```
package com.example.digi.doctor;

import androidx.annotation.NonNull;
import androidx.appcompat.app.AppCompatActivity;

import android.content.Intent;
import android.content.SharedPreferences;
```

```
import android.os.Bundle;
import android.util.Log;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.TextView;
import android.widget.Toast;

import com.google.firebase.database.DataSnapshot;
import com.google.firebase.database.DatabaseError;
import com.google.firebase.database.DatabaseReference;
import com.google.firebase.database.FirebaseDatabase;
import com.google.firebase.database.ValueEventListener;
import com.razorpay.Checkout;

import org.json.JSONObject;

import java.util.Calendar;
```

```
public class ConfirmAppointmentActivity extends AppCompatActivity {
    EditText et2;
    EditText et1;
    Button Btl;
    String p_key, p_email, Start, End, DoctorId, Day;
    String doctorname = "", Doctorphn = "", category = "";
    String patientName = "", Patientno = "";

    TextView tv222, tv333, tv444, tv555, tv666, tv777, tv888, tv999, tv1000;
    DatabaseReference doctorref, patientref, BookingRef, booking, bookingdocref;
    String problem;
    //String date;
    String mydate;
    int year = 1;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
```

```
        setContentView(R.layout.activity_confirm_appointment);
        //
        final Calendar myCalendar = Calendar.getInstance();
        et1 = (EditText) findViewById(R.id.et1);
        SharedPreferences
        sharedPreferences = getSharedPreferences("Patient", MODE_PRIVATE);
        p_key = sharedPreferences.getString("Patient Key", "");
        p_email = sharedPreferences.getString("UserName", "");

        // getting values from intent
        Intent incomingIntent = getIntent();
        Start = incomingIntent.getStringExtra("Start");
        End = incomingIntent.getStringExtra("End");
        DoctorId = incomingIntent.getStringExtra("DoctorId");
        Day = incomingIntent.getStringExtra("Day");
        mydate = incomingIntent.getStringExtra("date"); ///////////////

        et1.setText(mydate); /////
```

```
        //final DatePickerDialog.OnDateSetListener date = new
        DatePickerDialog.OnDateSetListener() {
            @Override
            public void onDateSet(DatePicker view, int year, int month, int
            dayOfMonth) {
                myCalendar.set(Calendar.YEAR, year);
                myCalendar.set(Calendar.MONTH, month);
                myCalendar.set(Calendar.DAY_OF_MONTH, dayOfMonth);
                Label();
            }
        }
```

```

private void label1()
{
    String myFormat = "dd-MM-yyyy";
    SimpleDateFormat sdf = new SimpleDateFormat(myFormat,
    Locale.US);
    et1.setText(sdf.format(myCalendar.getTime()));
}
et1.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v)
    {
        DatePickerDialog(ConfirmAppointmentActivity.this, date, myCalendar.get(Calendar.
        YEAR), myCalendar.get(Calendar.MONTH), myCalendar.get(Calendar.DAY_OF_MONTH)).sh
        ow();
    }
}
}

```

```

et2=findViewById(R.id.et2);
bt1=findViewById(R.id.bt1);
tv222=findViewById(R.id.tv222);
tv333=findViewById(R.id.tv333);
tv444=findViewById(R.id.tv444);
tv555=findViewById(R.id.tv555);
tv666=findViewById(R.id.tv666);
tv777=findViewById(R.id.tv777);
tv888=findViewById(R.id.tv888);
tv999=findViewById(R.id.tv999);
tv1000=findViewById(R.id.tv1000);

// tv222.setText(doctorname);
// tv333.setText(Doctorphn);
// tv444.setText(patientName);
//tv555.setText(PatientAge);
//tv666.setText(category);
//tv777.setText(Day);
//tv999.setText(Start);
//tv1000.setText(End);

```

```

FirebaseDatabase firebaseDatabase = FirebaseDatabase.getInstance();
doctorref = firebaseDatabase.getReference("DoctorDetails");
patientref = firebaseDatabase.getReference("PatientDetails");
BookingRef = firebaseDatabase.getReference("BookingDetails");

```

```

doctorref.child(DocId).addListenerForSingleValueEvent(new
 ValueEventListener() {
    @Override
    public void onDataChange(@NonNull DataSnapshot dataSnapshot)
    {
        Log.d("DataSnapshot", dataSnapshot.toString());
        Doctor details obj
        dataSnapshot.getValue(Doctor details.class);
        Doctorphn = obj.getPhoneNo();
        doctorname = obj.getFullName();
        category = obj.getCategory();

        tv222.setText(doctorname);
        tv333.setText(Doctorphn);
        tv666.setText(category);
        tv777.setText(Day);

        fees=Integer.parseInt(obj.getBasicFees());
        tv888.setText("Fees is "+obj.getBasicFees()+" Rupees");
        tv999.setText(Start);
        tv1000.setText(End);
    }
}
}

```

```

@Override
public void onCancelled(@NonNull DatabaseError databaseError) {
}
}

patientref.child(p_key).addListenerForSingleValueEvent(new
 ValueEventListener() {
@Override
public void onDataChange(@NonNull DataSnapshot dataSnapshot) {
PatientDetails obj =
dataSnapshot.getValue(PatientDetails.class);
Patientno = obj.getPhoneno();
patientName = obj.getFullName();

tv11.setText(patientName);
tv55.setText(patientno);
}
}
}

```

```

@Override

```

```

public void onCancelled(@NonNull DatabaseError databaseError) {
}
}

//
}

public void Proceed(View view) {
problem = et2.getText().toString();
// Date = et1.getText().toString();
//startPayment();

onPaymentSuccess("success");
}
}

```

```

public void startPayment() {
//
You need to pass current activity in order to let Razorpay create
CheckoutActivity
+
final ConfirmAppointmentActivity activity = this;
final Checkout co = new Checkout();
try {
fee=fee*100;
//basefee=basefee*100;

JSONObject options = new JSONObject();
options.put("name", "Doctor Consultation");
options.put("description", "Appointment Charge");
// You can omit the image option to fetch the image from dashboard
options.put("image",
"http://s3.amazonaws.com/osp-public/images/fee.png");
options.put("currency", "INR");
options.put("amount", fee);

JSONObject prefill = new JSONObject();
prefill.put("email", "");

```

```

prefill.put("contact", "");

options.put("prefill", prefill);

co.open(activity, options);
} catch (Exception e) {
Log.d("MYMSG", e.getMessage());
Toast.makeText(activity, "Error: " + e.getMessage(),
Toast.LENGTH_SHORT)
.show();
e.printStackTrace();
}
}
}

```

```

public void onPaymentSuccess(String s) {
    Toast.makeText(this, "Success " + s, Toast.LENGTH_SHORT).show();
    String bookingkey = BookingRef.push().getKey();
}

```

```

Booking obj = new Booking(p key, patientName, Patientno, Day, Start
, End, DoctorId,
    doctorname, category, Doctorphn, problem, "booked", mydate
, bookingkey);
BookingRef.child(bookingkey).setValue(obj);

BookedSlots mob=new BookedSlots(mydate, Start, End, mydate+"-"+Start);
// Log.d("MYMSG", "object created Successfully");

FirebaseDatabase firebaseDatabase = FirebaseDatabase.getInstance();
booking = firebaseDatabase.getReference("BookedSlots");
bookingdoctor=booking.child(DoctorId);

String mykey = bookingdoctor.push().getKey();
// Log.d("MYMSG", "Key Saved Successfully");
bookingdoctor.child(mykey).setValue(mob);
//Log.d("MYMSG", "Data Saved Successfully");

//Toast.makeText(this, "Slot Booked Successfully",
Toast.LENGTH_SHORT).show();
finish();
}
}

```

```

public void onPaymentError(int i, String s) {
    Toast.makeText(this, "Fails " + s, Toast.LENGTH_SHORT).show();
}
}

```

Call Option:

```

package com.example.digi doctor;

import androidx.appcompat.app.AppCompatActivity;
import androidx.cardview.widget.CardView;

```

```

import android.content.Intent;
import android.content.pm.PackageManager;
import android.net.Uri;
import android.os.Bundle;
import android.view.View;
import android.widget.Toast;

public class Call options extends AppCompatActivity {
    CardView userlogin;
    String phone = "";
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_call_options);
        Intent in = getIntent();
        phone = in.getStringExtra("mobile");
        // Toast.makeText(this, ""+phone, Toast.LENGTH_SHORT).show();
        userlogin = findViewById(R.id.userlogin);
    }
}

```

```
userLogin.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        //whatsapp
        String st = phone.trim();
        if(!st.contains("+91")){
            st = "+91"+st;
        }
    }
});
```

```
String text = "Welcome to Doctor Consultations";
String contact = st.trim(); //use country code with your phone number
String url =
"https://api.whatsapp.com/send?text="+text+"&phone="+contact;
try {
    PackageManager pm = getApplication().getPackageManager();
    pm.getPackageInfo("com.whatsapp",
PackageManager.GET_ACTIVITIES);
    Intent i = new Intent(Intent.ACTION_VIEW);
    i.setData(Uri.parse(url));
    startActivity(i);
}
```

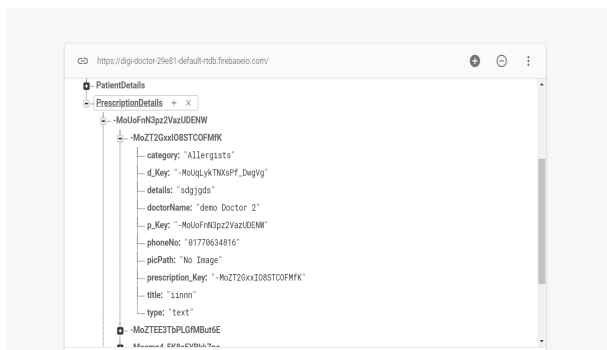
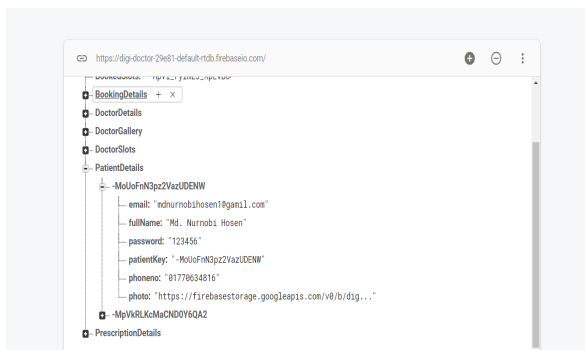
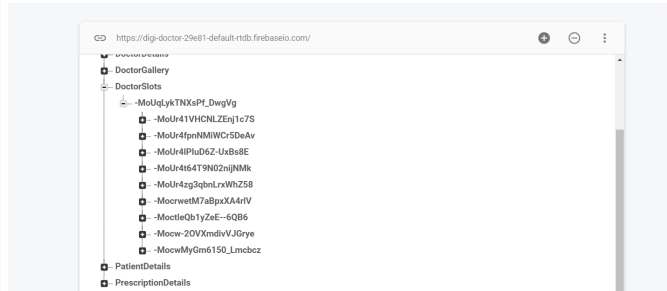
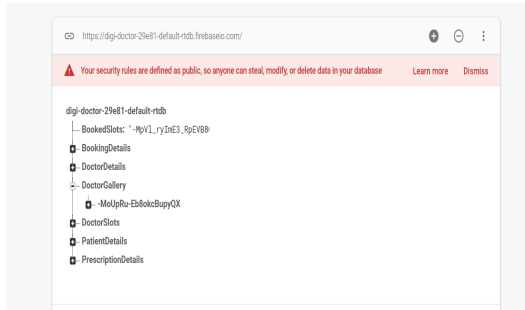
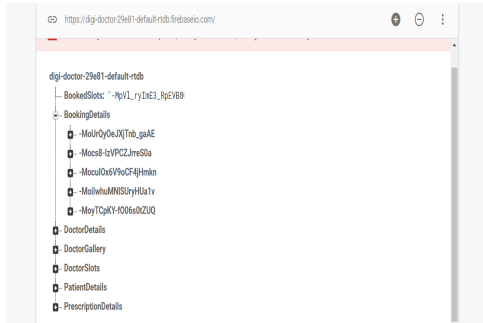
```
    } catch (PackageManager.NameNotFoundException e) {
        Toast.makeText(CallOptions.this, "Whatsapp app not
installed in your phone", Toast.LENGTH_SHORT).show();
        e.printStackTrace();
    }
}
});
}
}
}
}
}
}

public void btn_call(View view) {
    //dialer
    Uri u = Uri.parse("tel:" + phone);
    // Create the intent and set the data for the
    // intent as the phone number
    Intent i = new Intent(Intent.ACTION_DIAL, u);
    try {
        // Launch the Phone app's dialer with a phone
        // number to dial a call
        startActivity(i);
    } catch (SecurityException s) {
    }
}
```

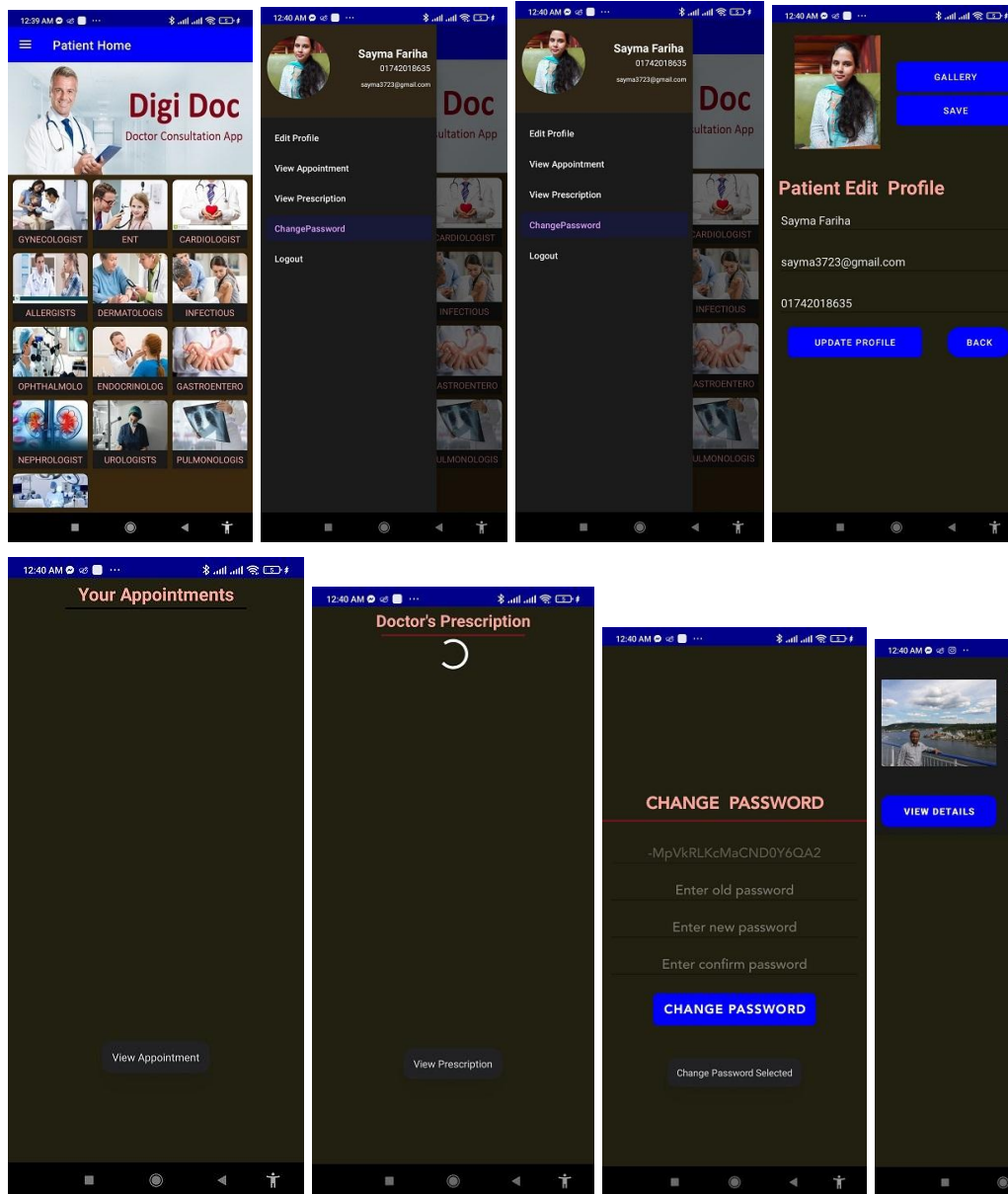
```
    }
    // show() method display the toast with
    // exception message
    Toast.makeText(this, "An error occurred", Toast.LENGTH_LONG)
        .show();
}
}
}
```

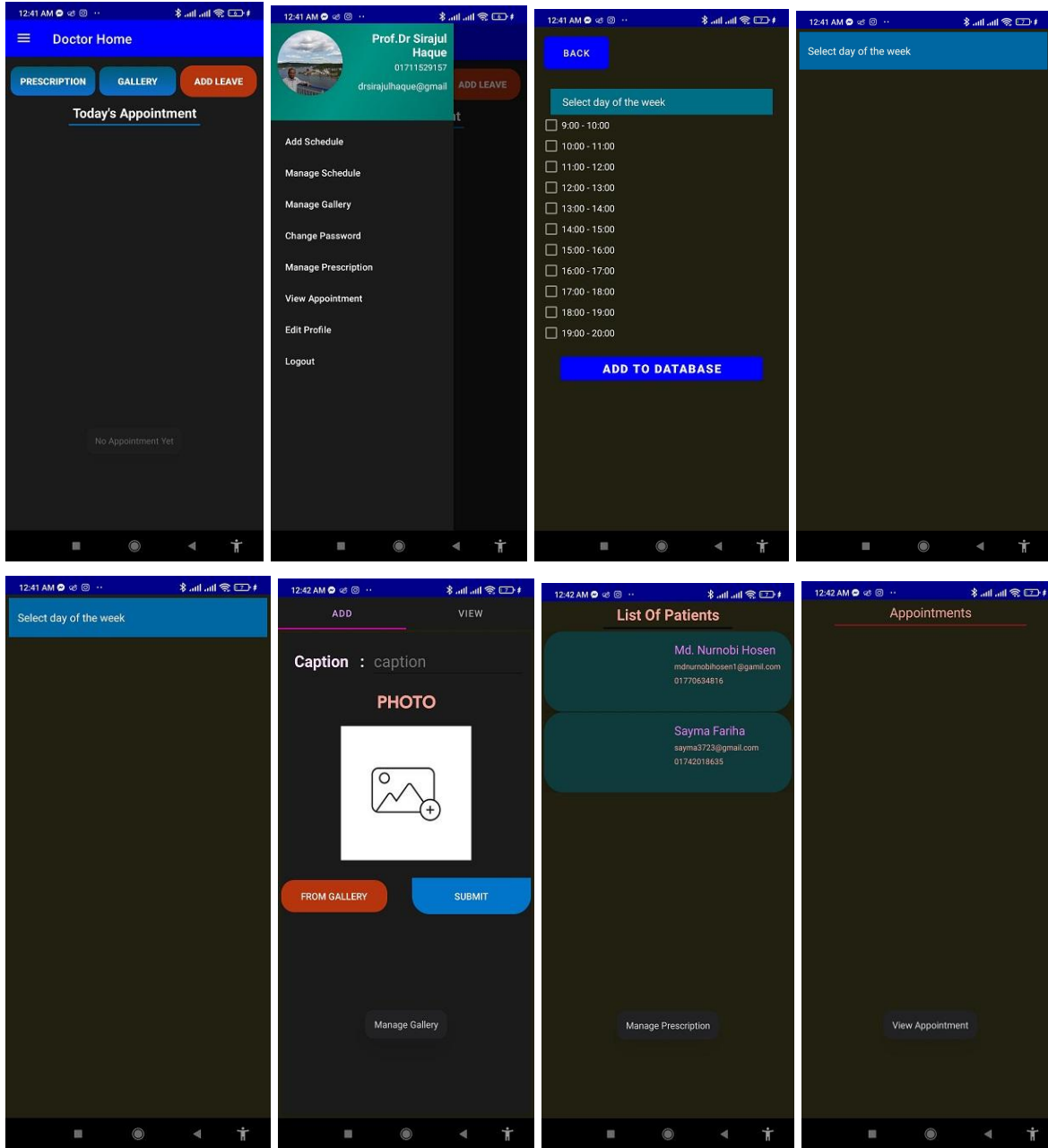
Appendix -B Database Design:

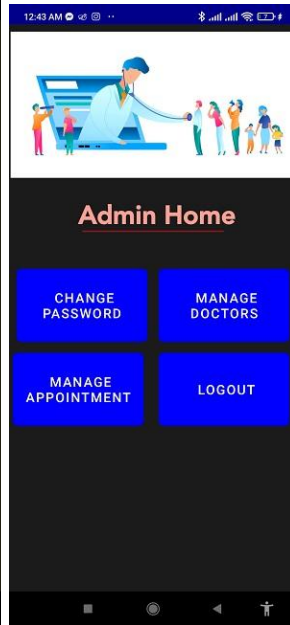
This section is the extended part of Chapter 5.3.1: Database Design. Here we have shown a multiple table of the firebase which is Realtime. There are only three figures which are Real Time databases, Patients and Doctors Data.



Appendix -C GUI Design:







TurnitinReport

Digi Doc

ORIGINALITY REPORT

15% SIMILARITY INDEX	8% INTERNET SOURCES	1% PUBLICATIONS	14% STUDENT PAPERS
--------------------------------	-------------------------------	---------------------------	------------------------------

PRIMARY SOURCES

1	dspace.daffodilvarsity.edu.bd:8080 Internet Source	4%
2	Submitted to Daffodil International University Student Paper	2%
3	Submitted to University of Greenwich Student Paper	1%
4	Submitted to Huddersfield New College Student Paper	1%
5	Submitted to University of Wales Institute, Cardiff Student Paper	1%
6	taxguru.in Internet Source	1%
7	Submitted to Angel Group Student Paper	1%
8	Submitted to Multimedia University Student Paper	<1%
9	Submitted to The British College Student Paper	<1%

10	Submitted to Harrisburg University of Science and Technology Student Paper	<1 %
11	Submitted to SP Jain School of Global Management Student Paper	<1 %
12	Submitted to Westcliff University Student Paper	<1 %
13	cnnb.daffodilvarsity.edu.bd Internet Source	<1 %
14	Submitted to British University in Egypt Student Paper	<1 %
15	Submitted to Heriot-Watt University Student Paper	<1 %
16	Submitted to Middlesex University Student Paper	<1 %
17	Submitted to St. Mary's College Twickenham Student Paper	<1 %
18	Submitted to University of Central Lancashire Student Paper	<1 %
19	www.ukessays.com Internet Source	<1 %
20	Submitted to Western International College (WINC London) Student Paper	<1 %

21 Submitted to Universiti Teknologi MARA <1 %
Student Paper

22 bmcmedinformdecismak.biomedcentral.com <1 %
Internet Source

23 www.gfmer.ch <1 %
Internet Source

Exclude quotes Off

Exclude matches Off

Exclude bibliography Off