

# **MEDES**

**A Thesis Submitted  
In Partial Fulfillment of the Requirements  
for the degree of**

## **MASTER OF COMPUTER APPLICATIONS**

**by**

**Arun Kumar**

**(Enrollment No. 200029014005712)**

**Vishal Sahu**

**(Enrollment No. 200029014005819)**

**Gunjit Dalal**

**(Enrollment No. 200029014005733)**

**Under the Supervision of  
Mr. R.N.Panda & Mr. Ankit Verma  
KIET Group of Institutions, Ghaziabad**



**to the**

**FACULTY OF .....**

**DR. APJ ABDUL KALAM TECHNICAL UNIVERSITY  
LUCKNOW**

**(Formerly Uttar Pradesh Technical University, Lucknow)**

**June, 2022**

## **DECLARATION**

I hereby declare that the work presented in this report entitled "MEDES", was carried out by me. I have not submitted the matter embodied in this report for the award of any other degree or diploma of any other University or Institute. I have given due credit to the original authors/sources for all the words, ideas, diagrams, graphics, computer programs, experiments, results, that are not my original contribution. I have used quotation marks to identify verbatim sentences and given credit to the original authors/sources. I affirm that no portion of my work is plagiarized, and the experiments and results reported in the report are not manipulated. In the event of a complaint of plagiarism and the manipulation of the experiments and results, I shall be fully responsible and answerable.

Name :

Enroll. No. :

Field :

(Candidate Signature)

## CERTIFICATE

Certified that **Arun Kumar (2000290140027)**, **Vishal Sahu (2000290140134)**, **Gunjit Dalal (2000290140048)**, have carried out the project work having “**MEDES**” for Master of Computer Applications from Dr. A.P.J. Abdul Kalam Technical University (AKTU) (formerly UPTU), Technical University, Lucknow under my supervision. The project report embodies original work, and studies are carried out by the student himself / herself and the contents of the project report do not form the basis for the award of any other degree to the candidate or to anybody else from this or any other University/Institution.

Signature

Mr. R.N.Panda  
Associate Professor  
KIET Group of Institution Ghaziabad

Mr. Ankit Kumar  
Assistant Professor  
KIET Group of Institution Ghaziabad

Date:

## **ABSTRACT**

Our Project “MEDES” is an Online book an Appointment Application The proposed project is a smart appointment booking system that provides patients or any user an easy way of booking a doctor’s appointment online. This is a application that overcomes the issue of managing and booking appointments according to user’s choice or demands. The task sometimes becomes very tedious for the compounder or doctor himself in manually allotting appointments for the users as per their availability. Hence this project offers an effective solution where users can view various booking slots available and select the preferred date and time. The already booked space will be marked yellow and will not be available for anyone else for the specified time. This system also allows users to cancel their booking anytime. The application uses Flutter as a front-end and Firebase database as the backend.

## ACKNOWLEDGEMENTS

Success in life is never attained single handedly. My deepest gratitude goes to my thesis supervisor, **Mr. R.N. Panda and Mr. Ankit Verma** for his guidance, help and encouragement throughout my research work. Their enlightening ideas, comments, and suggestions.

Words are not enough to express my gratitude to Dr. Ajay Kumar Shrivastava, Professor and Head, Department of Computer Applications, for his insightful comments and administrative help at various occasions.

Fortunately, I have many understanding friends, who have helped me a lot on many critical conditions.

Finally, my sincere thanks go to my family members and all those who have directly and indirectly provided me moral support and other kind of help. Without their support, completion of this work would not have been possible in time. They keep my life filled with enjoyment and happiness.

**Arun Kumar**  
**Vishal Sahu**  
**Gunjit Dalal**

## LIST OF CONTENTS

	Page No.
Declaration	ii
Certificate	iii
Abstract	iv
Acknowledgement	v
List of Tables	vi
<b>Chapter 1 : Introduction</b>	<b>2-3</b>
1.1 Project description	2
1.2 Project Scope	3
1.3 Hardware / Software used in Project	3
<b>Chapter 2 : Feasibility Study</b>	<b>4-7</b>
2.1 Technical feasibility	4
2.2 Operational Feasibility	6
2.3 Economical Feasibility	7
<b>Chapter 3 : Database Design</b>	<b>8-12</b>
3.1 Database Tables	8
3.2 Flow Chart	9
3.3 Use Case Diagram	10
3.4 Sequence Diagram	11
3.6 Collaborative Diagram	12
<b>Chapter 4 : Form Design</b>	<b>13-16</b>
4.1 Input / Output Form (Screenshot)	
<b>Chapter 5 : Coding</b>	<b>17-68</b>
<b>Chapter 6 : Testing</b>	<b>69</b>
6.1 Test Case for Home	
6.2 Test Case for Article	
6.3 Test Case for Appointment Page	
6.4 Test Case for Profile Page	
Bibliography	70

(Signature of the candidate)

Name: .....

Enrollment No. : .....

# **CHAPTER 1**

## **INTRODUCTION**

Medes has created tech-enabled processes that ensure that the customer's entire journey from booking to report delivery is seamless and simplified. We make sure that the sample collection and testing process through our proprietary technological developments like the tamper-proof sample collection kits (SmartPrik) to pre-barcoded vials to Live temperature monitored sample transportation boxes (CoolSure) is quick, painless, and of high quality. To ensure the accuracy of reports, Medes has deployed an advanced tech-enabled quality control process. Through a mix of specialized training, remote supervision, and modern age devices, Medes has standardized the diagnostics process and makes sure that customers get accurate reports every time..

The user can use it only after registering/creating the account on our platform. We provide a vast range of medical tests that can be done at your doorstep and others from the nearest test centre.

### **1.1 PROJECT DESCRIPTION**

The proposed project is a smart appointment booking system that provides patients or any user an easy way of booking a doctor's appointment online. This is a web based application that overcomes the issue of managing and booking appointments according to user's choice or demands. The task sometimes becomes very tedious for the compounder or doctor himself in manually allotting appointments for the users as per their availability. Hence this project offers an effective solution where users can view various booking slots available and select the preferred date and time. The already booked space will be marked yellow and will not be available for anyone else for the specified time. This system also allows users to cancel their booking anytime.

The application uses flutter as a front-end and firebase database as the backend. You can also book an Appointment for a medical test. This project is very user friendly and android application, In which user Signup in this application and book their appointment according available slots.

Medes has created tech-enabled processes that ensure that the customer's entire journey from booking to report delivery is seamless and simplified. We make sure that the sample collection and testing process through our proprietary technological developments like the tamper-proof sample collection kits (SmartPrik) to pre-barcoded vials to Live temperature monitored sample transportation boxes (CoolSure) is quick, painless, and of high quality. To ensure the accuracy of reports, Healthians has deployed an advanced tech-enabled quality control process. Through a mix of specialized training, remote supervision, and modern age devices, Healthians has standardized the diagnostics process and makes sure that customers get accurate reports every time.

The user can use it only after registering/creating the account on our platform. We provide a vast range of medical tests that can be done at your doorstep and others from

the nearest test centre.

Medical tests can help detect a condition, determine a diagnosis, plan treatment, check to see if treatment is working, or monitor the condition over time. A doctor may order these tests as part of a routine checkup, to check for certain diseases and disorders, or to monitor your health.

## **1.2 PROJECT SCOPE**

I guess it depends on how busy they are, if the patients are coming naturally, then they won't do anything, but if they need more footprint then could use one of the lead gen companies

In India majority of the appointments are still booked through phone call or walk-ins but things are changing since we have quite a few companies focusing on fixing this problem and investors have pumped in enough money in this segment (doctor discovery).

As more and more people are starting to use smartphones these days, gathering the user's attention and making them reach the hospital has been made simple with the doctor booking app solution. The digital interface made the appointment booking simple and effective. This reduces the waiting time on the front of the hospital for checkups.

## **1.3 HARDWARE / SOFTWARE USED IN PROJECT :SYSTEM REQUIREMENTS**

The cool thing about using Flutter for creating cross-platform native mobile apps is the fact that you can build those on almost any OS.

Here are some System Requirements for Android Studio which is needed for running an Android simulator.

Windows:

- Microsoft® Windows® 7/8/10 (64-bit)
- 4 GB RAM minimum, 8 GB RAM recommended
- 2 GB of available disk space minimum,
- 4 GB Recommended (500 MB for IDE + 1.5 GB for Android SDK and emulator system image)
- 1280 x 800 minimum screen resolution

### **Software used in project**

- VS CODE



## **CHAPTER 2**

### **2.1 FEASIBILITY STUDY**

A feasibility study assesses the operational, technical and economic merits of the proposed project. The feasibility study is intended to be a preliminary review of the facts to see if it is worthy of proceeding to the analysis phase. From the systems analyst perspective, the feasibility analysis is the primary tool for recommending whether to proceed to the next phase or to discontinue the project.

The feasibility study is a management-oriented activity. The objective of a feasibility study is to find out if an information system project can be done and to suggest possible alternative solutions.

Projects are initiated for two broad reasons:

1. Problems that lend themselves to systems solutions
2. Opportunities for improving through: (a) upgrading systems (b) altering systems (c) installing new systems

A feasibility study should provide management with enough information to decide:

- Whether the project can be done
- Whether the final product will benefit its intended users and organization
- What are the alternatives among which a solution will be chosen
- Is there a preferred alternative

## 2.2 TECHNICAL FEASIBILITY

A large part of determining resources has to do with assessing technical feasibility. It considers the technical requirements of the proposed project. The technical requirements are then compared to the technical capability of the organization. The systems project is considered technically feasible if the internal technical capability is sufficient to support the project requirements.

The analyst must find out whether current technical resources can be upgraded or added to in a manner that fulfills the request under consideration. This is where the expertise of system analysts is beneficial, since using their own experience and their contact with vendors they will be able to answer the question of technical feasibility. The essential questions that help in testing the operational feasibility of a system include the following:

Is the project feasible within the limits of current technology? Does the technology exist at all?

Is it available within given resource constraints? Is it a practical proposition?

Manpower- programmers, testers & debuggers Software and hardware

Are the current technical resources sufficient for the new system?

Can they be upgraded to provide to provide the level of technology necessary for the new system?

Do we possess the necessary technical expertise, and is the schedule reasonable? Can the technology be easily applied to current problems?

Does the technology have the capacity to handle the solution?

## **2.3 OPERATIONAL FEASIBILITY**

Operational feasibility is dependent on human resources available for the project and involves projecting whether the system will be used if it is developed and implemented. Operational feasibility is a measure of how well a proposed system solves the problems, and takes advantage of the opportunities identified during scope definition and how it satisfies the requirements identified in the requirements analysis phase of system development.

Operational feasibility reviews the willingness of the organization to support the proposed system. This is probably the most difficult of the feasibilities to gauge. In order to determine this feasibility, it is important to understand the management commitment to the proposed project. If the request was initiated by management, it is likely that there is management support and the system will be accepted and used. However, it is also important that the employee base will be accepting of the change. The essential questions that help in testing the operational feasibility of a system include the following:

- Does current mode of operation provide adequate throughput and response time?
- Does current mode provide end users and managers with timely, pertinent, accurate and useful formatted information?
- Does current mode of operation provide cost-effective information services to the business?
- Could there be a reduction in cost and or an increase in benefits?
- Does current mode of operation offer effective controls to protect against fraud and to guarantee accuracy and security of data and information?
- Does current mode of operation make maximum use of available resources, including people, time, and flow of forms?
- Does current mode of operation provide reliable services
- Are the services flexible and expandable?
- Are the current work practices and procedures adequate to support the new system?
- If the system is developed, will it be used?
- Manpower problems
- Labour objections

## 2.4 ECONOMICAL FEASIBILITY

Economic analysis could also be referred to as cost/benefit analysis. It is the most frequently used method for evaluating the effectiveness of a new system. In economic analysis the procedure is to determine the benefits and savings that are expected from a candidate system and compare them with costs. If benefits outweigh costs, then the decision is made to design and implement the system. An entrepreneur must accurately weigh the cost versus benefits before taking an action. Possible questions raised in economic analysis are:

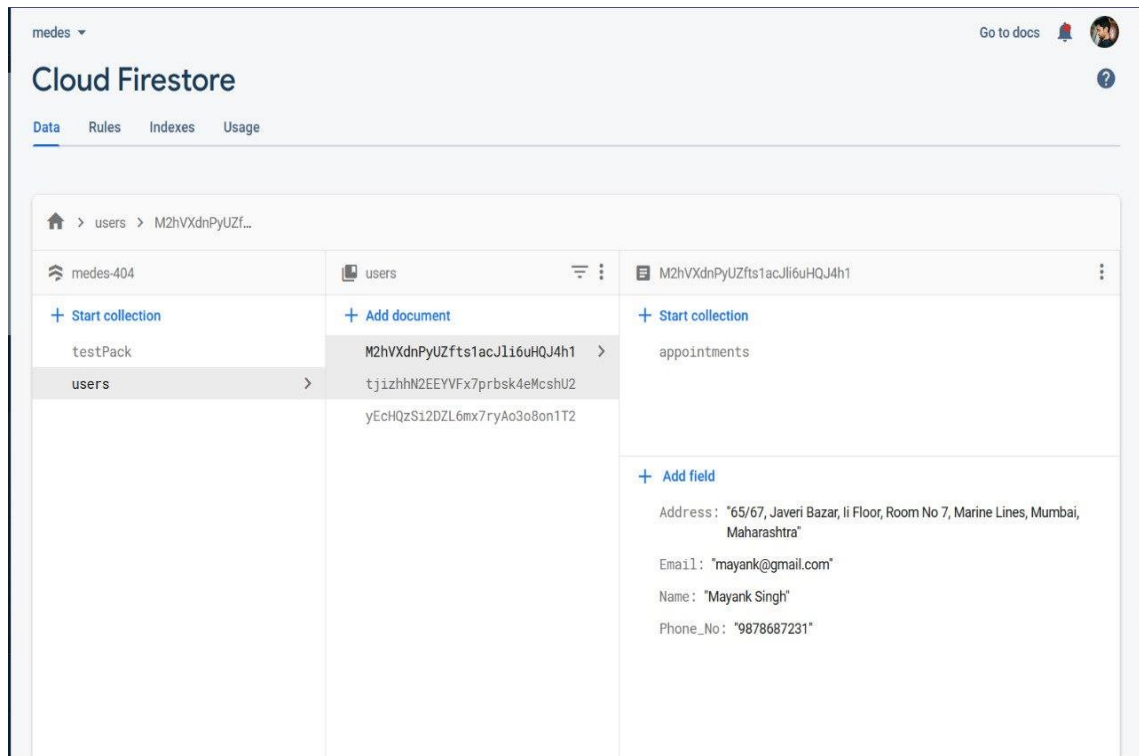
- Is the system cost effective?
- Do benefits outweigh costs?
- The cost of doing full system study
- The cost of business employee time
- Estimated cost of hardware
- Estimated cost of software/software development
- Is the project possible, given the resource constraints?
- What are the savings that will result from the system?
- Cost of employees' time for study
- Cost of packaged software/software development
- Selection among alternative financing arrangements (rent/lease/purchase)

The concerned business must be able to see the value of the investment it is pondering before committing to an entire system study. If short-term costs are not overshadowed by long-term gains or produce no immediate reduction in operating costs, then the system is not economically feasible, and the project should not proceed any further. If the expected benefits equal or exceed costs, the system can be judged to be economically feasible. Economic analysis is used for evaluating the effectiveness of the proposed system. The economical feasibility will review the expected costs to see if they are in-line with the projected budget or if the project has an acceptable return on investment. At this point, the projected costs will only be a rough estimate. The exact costs are not required to determine economic feasibility. It is only required to determine if it is feasible that the project costs will fall within the target budget or return on investment. A rough estimate of the project schedule is required to determine if it would be feasible to complete the systems project within a required timeframe. The required timeframe would need to be set by the organization.

## CHAPTER 3

### 3. DATABASE DESIGN

#### 3.1 Database Tables



Store and sync data with our NoSQL cloud database. Data is synced across all clients in real-time, and remains available when your app goes offline. The Firebase Realtime Database is a cloud-hosted database. Data is stored as JSON and synchronized in real-time to every connected client. When you build cross-platform apps with our Apple platforms, Android, and JavaScript SDKs, all of your clients share one Realtime Database instance and automatically receive updates with the newest data.

Data Base Tables Component:

- 1.Users
- 2.Add Document
- 3.Appointment
- 4.Address,Email,Name,Phone No.

### 3.2 FLOW CHART

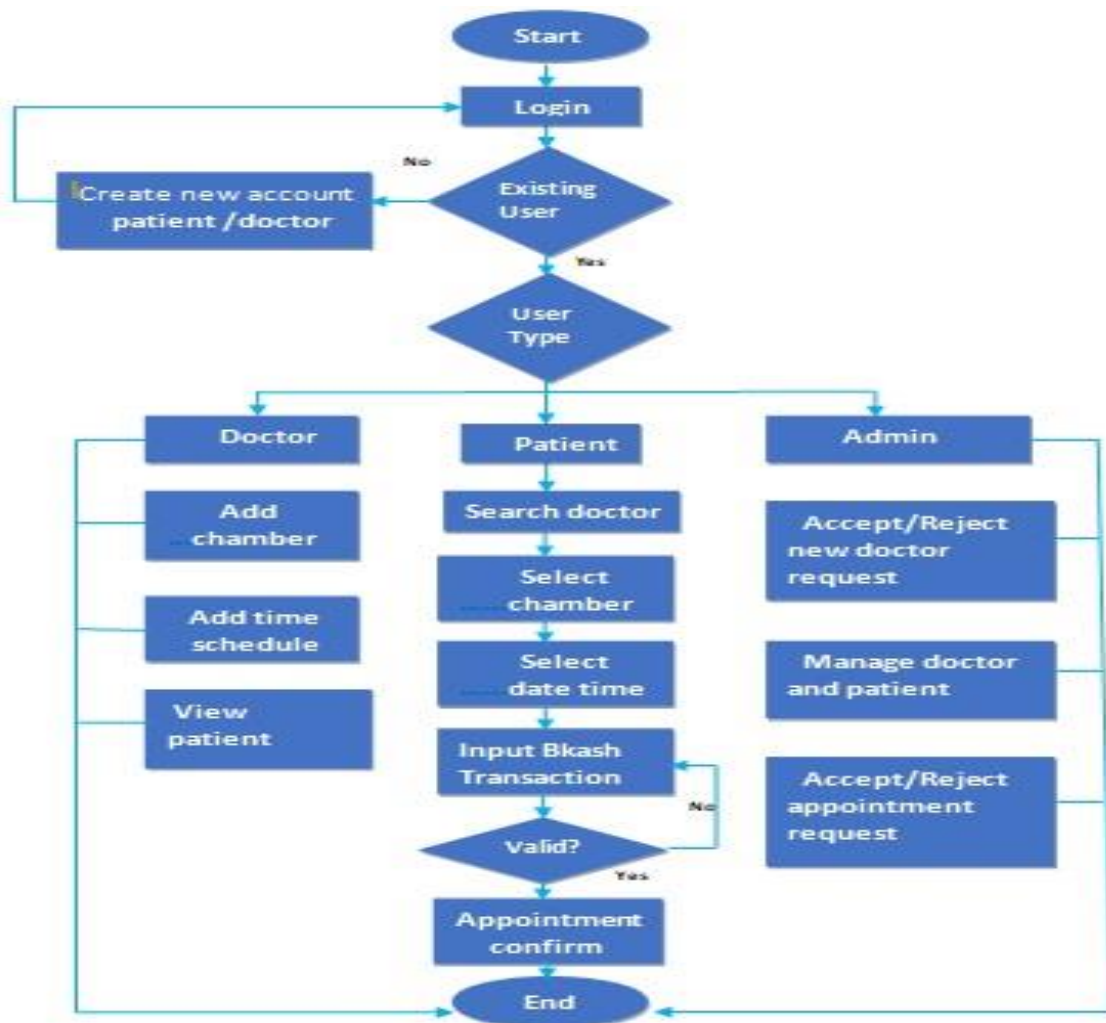


Figure 1: Flow Chart

### 3.3 USE CASE DIAGRAM

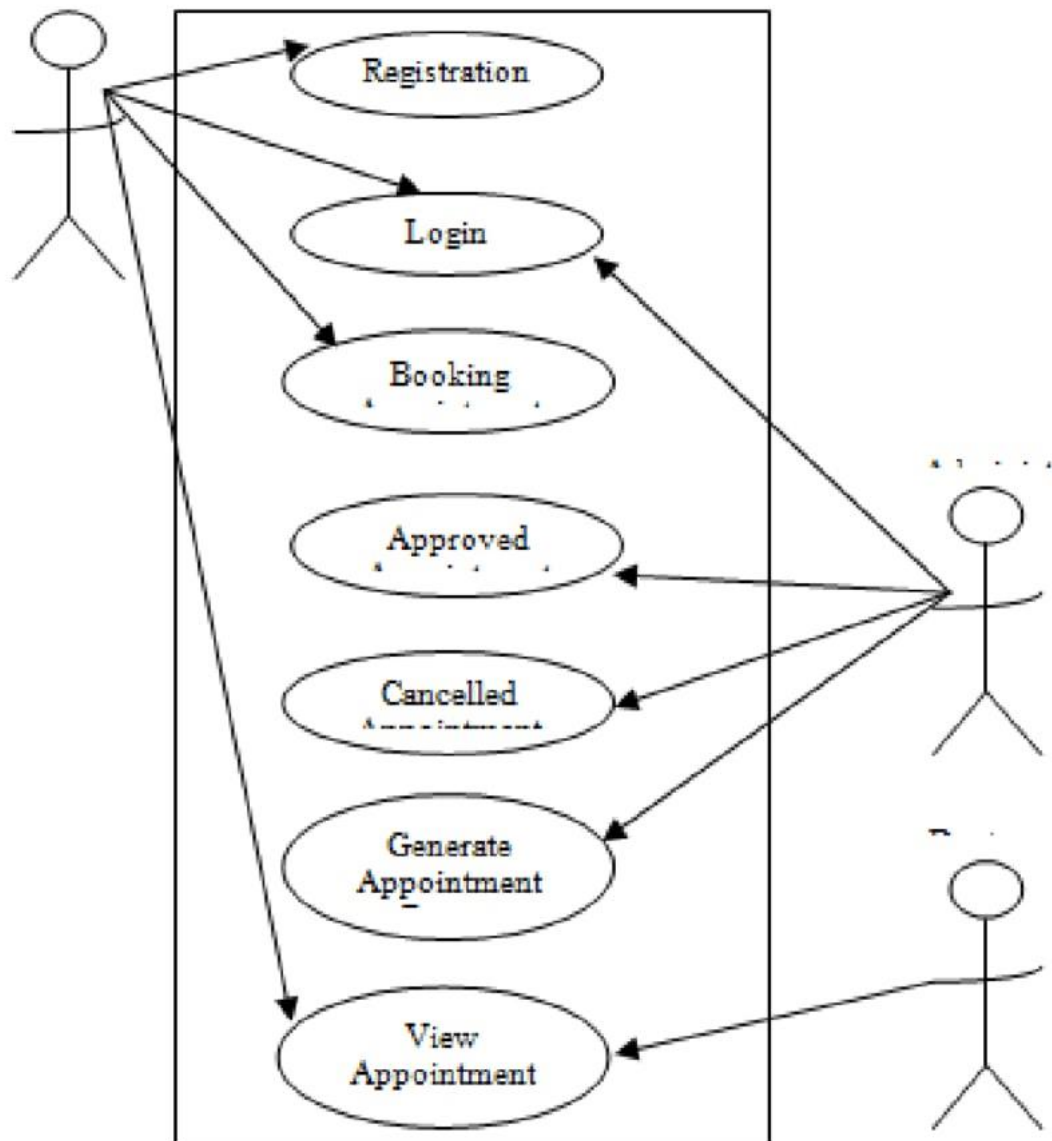
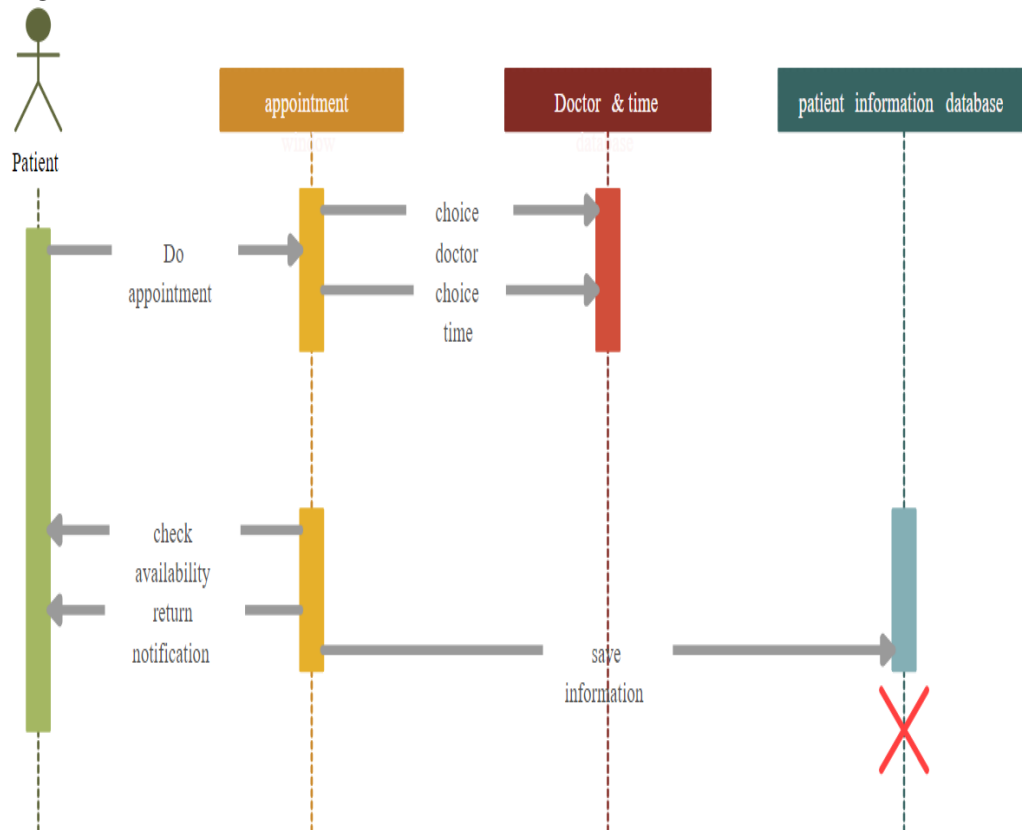


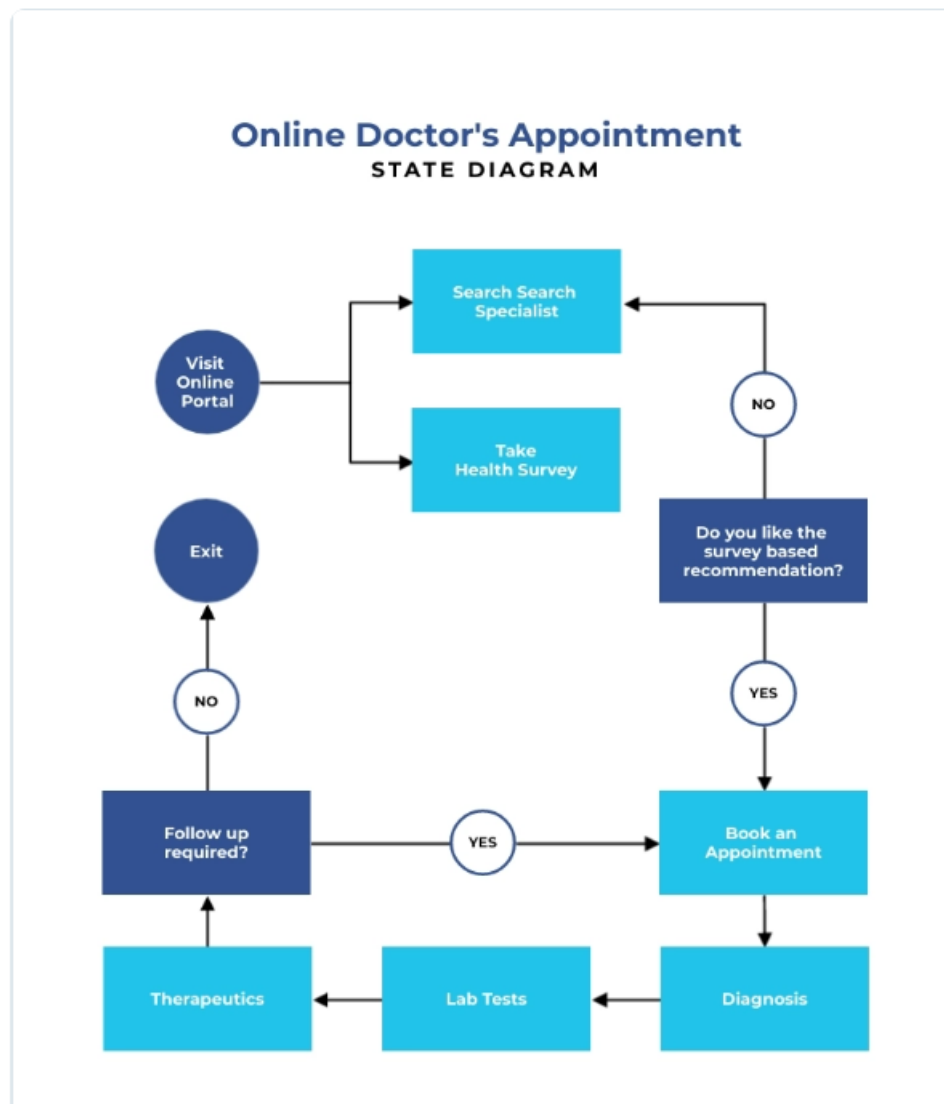
Figure 2: Use-Case Diagram of Online Shopping Website

### 3.4 SEQUENCE DIAGRAM





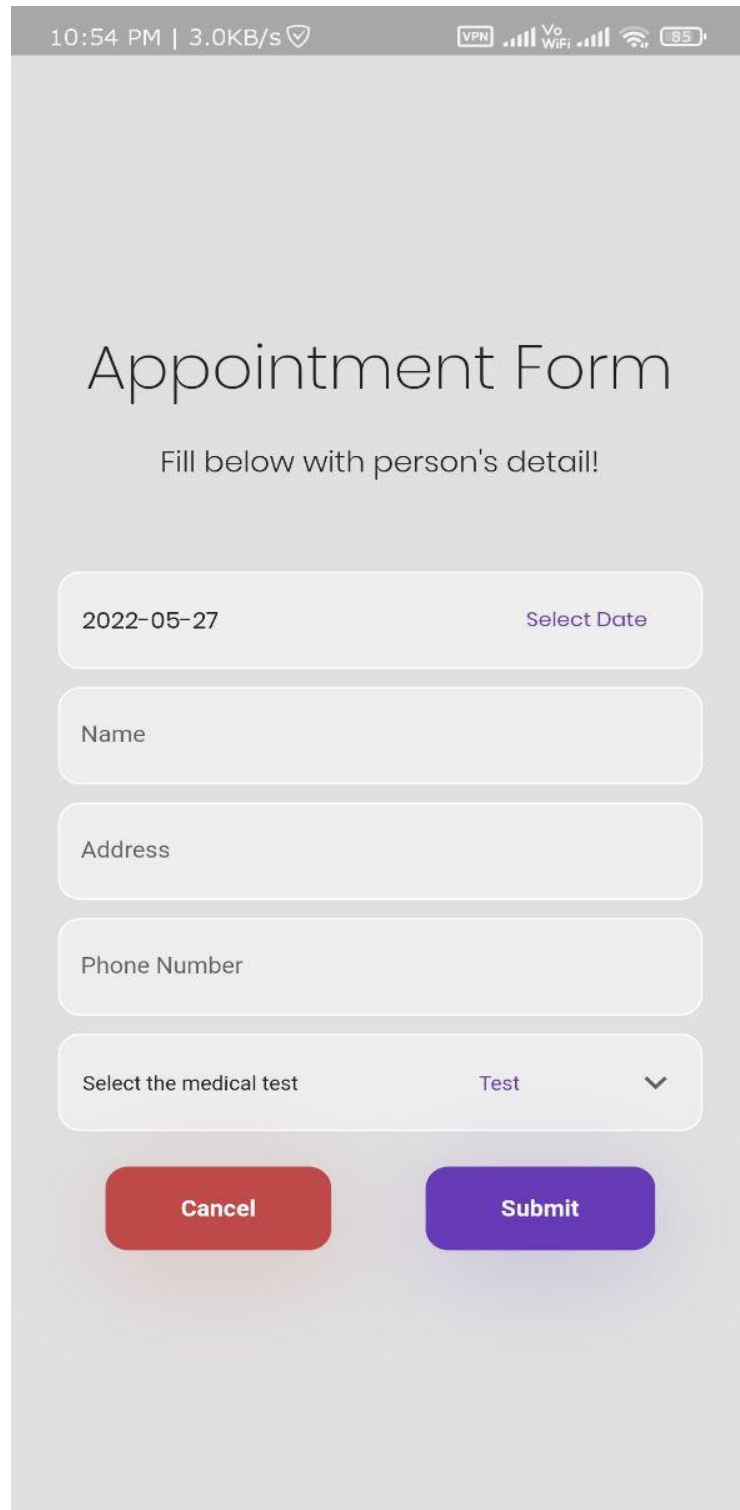
### 3.5 COLLABORATIVE DIAGRAM



## CHAPTER 4

### 4 FORM DESIGN

#### 4.1 INPUT / OUTPUT FORM (SCREENSHOT)



The screenshot shows a mobile application interface for an appointment form. At the top, the status bar displays the time 10:54 PM, data speed 3.0KB/s, VPN status, cellular signal, VoWiFi, Wi-Fi, and a battery level of 85%. The form has a light gray background with rounded rectangular input fields. The title 'Appointment Form' is centered in a large, dark gray font, followed by the instruction 'Fill below with person's detail!'. The form contains five input fields: a date field with '2022-05-27' and a 'Select Date' link, a text field for 'Name', a text field for 'Address', a text field for 'Phone Number', and a dropdown menu for 'Select the medical test' with 'Test' selected and a downward arrow. At the bottom, there are two buttons: a red 'Cancel' button and a purple 'Submit' button.

10:54 PM | 3.0KB/s

VPN VoWiFi 85%

# Appointment Form

Fill below with person's detail!

2022-05-27 [Select Date](#)

Name

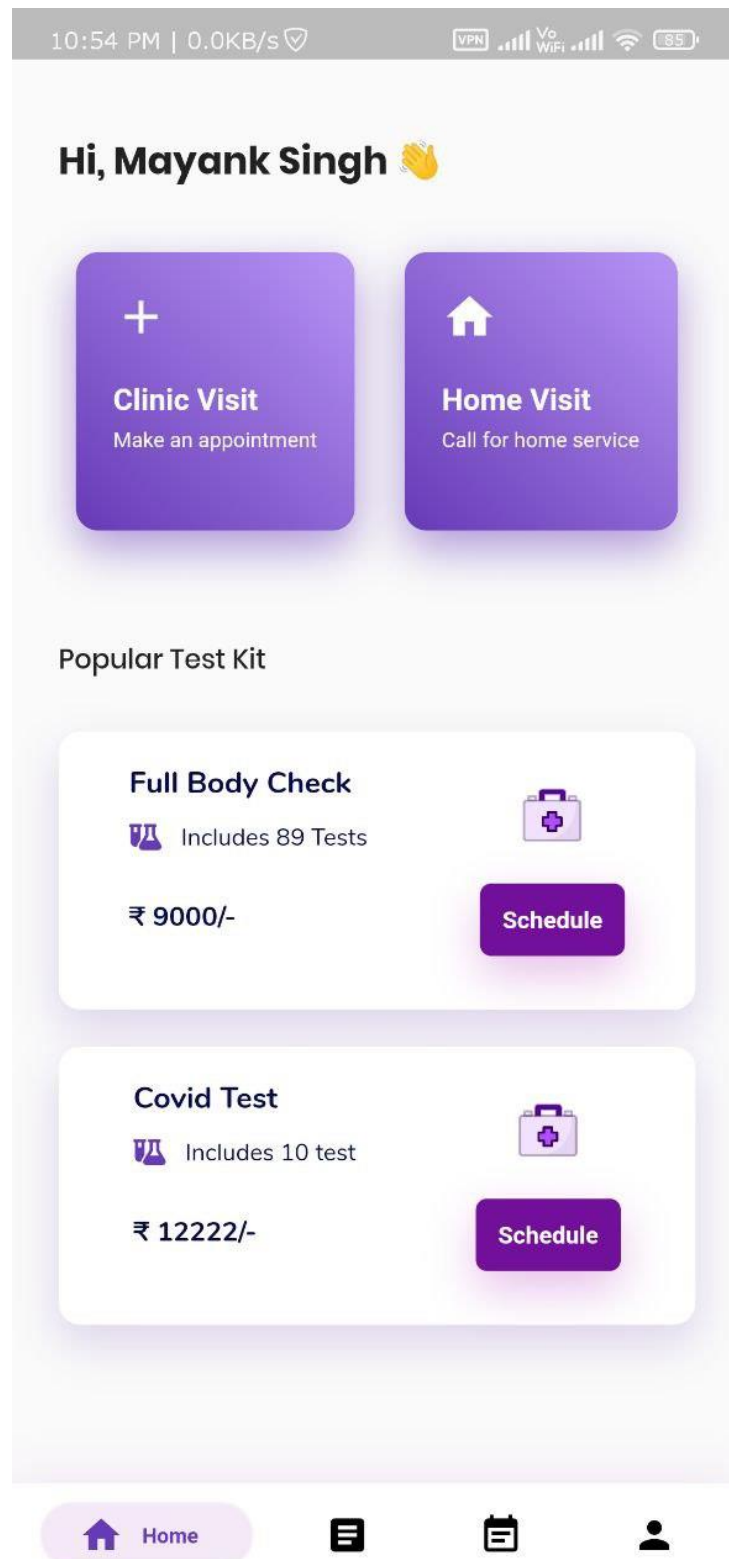
Address

Phone Number

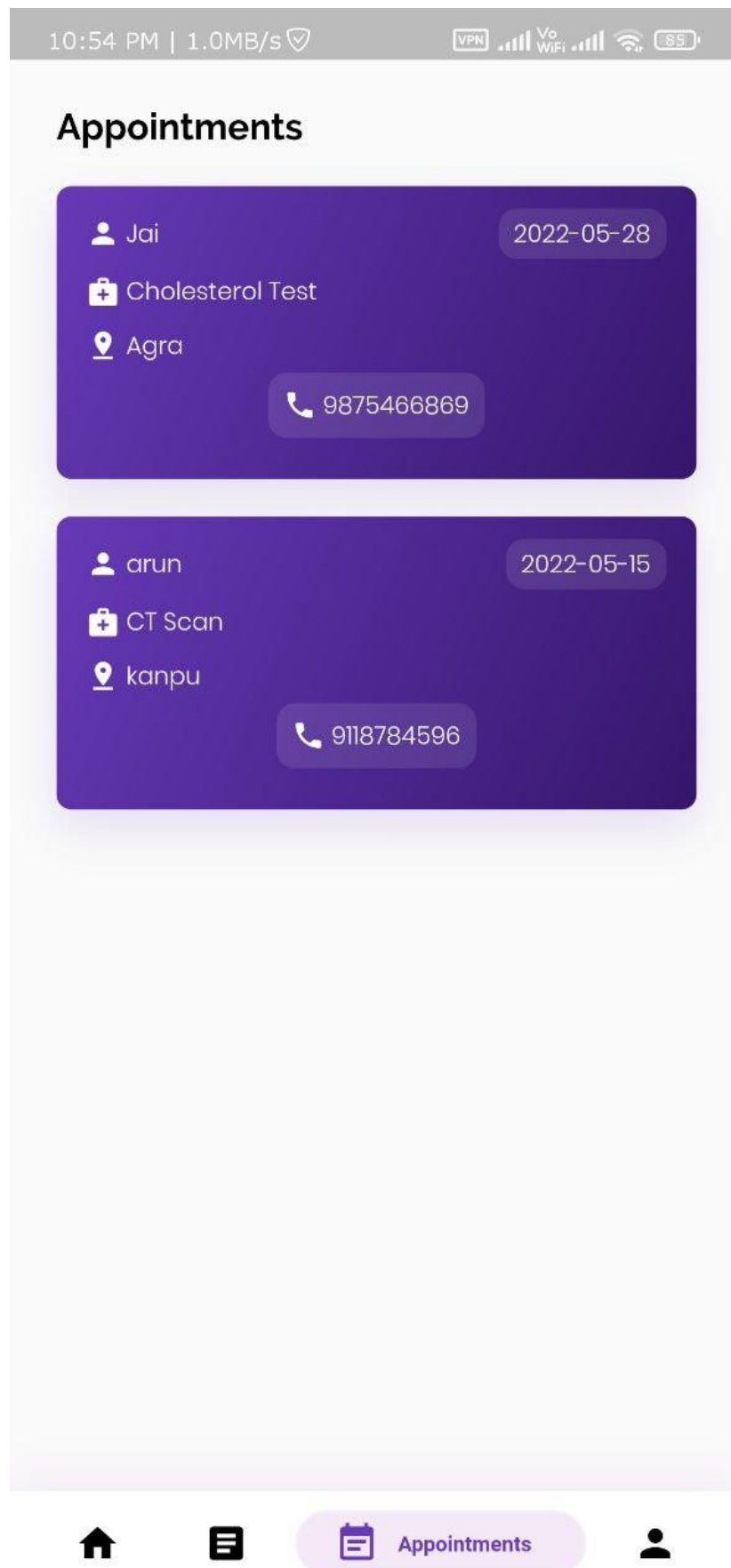
Select the medical test [Test](#) ▼

[Cancel](#) [Submit](#)

## HOMEPAGE



## APPOINTMENT PAGE





## CHAPTER 5

### 1 CODING:

#### appointment\_form.dart

```
import 'package:cloud_firestore/cloud_firestore.dart';
import 'package:firebase_auth/firebase_auth.dart';
import 'package:flutter/material.dart';
import 'package:google_fonts/google_fonts.dart';

class AppointmentForm extends StatefulWidget {
  const AppointmentForm({Key? key}) : super(key: key);

  @override
  State<AppointmentForm> createState() => _AppointmentFormState();
}

class _AppointmentFormState extends State<AppointmentForm> {
  final _nameController = TextEditingController();
  final _myLocController = TextEditingController();
  final _contactNumberController = TextEditingController();
  DateTime currentDate = DateTime.now();

  Future<void> _selectDate(BuildContext context) async {
    final DateTime? pickedDate = await showDatePicker(
      context: context,
      initialDate: currentDate,
      firstDate: DateTime.now().subtract(
        const Duration(days: 0),
      ),
      lastDate: DateTime(2030));
    if (pickedDate != null && pickedDate != currentDate) {
      setState(() {
        currentDate = pickedDate;
      });
    }
  }

  String dropdownValue = 'Test';

  Future createAppointment() async {
    User? user = FirebaseAuth.instance.currentUser;
    await FirebaseFirestore.instance
      .collection('users')
      .doc(user!.uid)
      .collection('appointments')
      .add({
        'name': _nameController.text.trim(),
        'address': _myLocController.text.trim(),
        'test-type': dropdownValue.toString().trim(),
        'contact_info': _contactNumberController.text.trim(),
      });
  }
}
```

```

      'date': currentDate.toString().substring(0, 10),
    });
    ScaffoldMessenger.of(context)
      .showSnackBar(const SnackBar(content: Text("Form submitted successfully.")));
    Navigator.pop(context);
    dispose();
  }

  @override
  void dispose() {
    _nameController.dispose();
    _myLocController.dispose();
    _contactNumberController.dispose();
    super.dispose();
  }

  final _formKey = GlobalKey<FormState>();
  @override
  Widget build(BuildContext context) {
    return Scaffold(
      body: Scaffold(
        backgroundColor: Colors.grey[300],
        body: SafeArea(
          child: Center(
            child: SingleChildScrollView(
              child: Form(
                key: _formKey,
                child: Column(
                  mainAxisAlignment: MainAxisAlignment.center,
                  children: [
                    Text(
                      'Appointment Form',
                      style: GoogleFonts.poppins(
                        fontSize: 42, fontWeight: FontWeight.w200),
                    ),
                    const SizedBox(height: 15),
                    Text(
                      'Fill below with person\'s detail!',
                      style: GoogleFonts.poppins(
                        fontSize: 20, fontWeight: FontWeight.w300),
                    ),

                    const SizedBox(height: 50),
                    Padding(
                      padding: const EdgeInsets.symmetric(horizontal: 25),
                      child: Container(
                        height: 55,
                        decoration: BoxDecoration(
                          borderRadius: BorderRadius.circular(12),
                          border: Border.all(color: Colors.white),

```

```

        color: Colors.grey[200],
      ),
      child: Row(
        mainAxisAlignment: MainAxisAlignment.spaceAround,
        children: [
          Text(
            currentDate.toString().substring(0, 10),
            style: GoogleFonts.poppins(fontSize: 16),
          ),
          const SizedBox(
            width: 100,
          ),
          MaterialButton(
            textColor: Colors.deepPurple,
            elevation: 0,
            onPressed: () => _selectDate(context),
            child: Text(
              'Select Date',
              style: GoogleFonts.poppins(fontSize: 14),
            ),
          ),
        ],
      ),
    ),
  ),
  const SizedBox(
    height: 10,
  ),
  //Name Field
  Padding(
    padding: const EdgeInsets.symmetric(
      horizontal: 25,
    ),
    child: TextFormField(
      validator: (value) {
        if (value == null || value.isEmpty) {
          return 'Please fill your name';
        }
        return null;
      },
      keyboardType: TextInputType.name,
      controller: _nameController,
      decoration: InputDecoration(
        enabledBorder: OutlineInputBorder(
          borderSide: const BorderSide(color: Colors.white),
          borderRadius: BorderRadius.circular(12),
        ),
        focusedBorder: OutlineInputBorder(
          borderSide:
            const BorderSide(color: Colors.deepPurple),

```



```

        borderRadius: BorderRadius.circular(12),
      ),
      fillColor: Colors.grey[200],
      filled: true,
      hintText: 'Name',
      border: InputBorder.none,
    ),
  ),
),
const SizedBox(
  height: 10,
),

//Address Field
Padding(
  padding: const EdgeInsets.symmetric(
    horizontal: 25,
  ),
  child: TextFormField(
    validator: (value) {
      if (value == null || value.isEmpty) {
        return 'Please fill address';
      }
      return null;
    },
    keyboardType: TextInputType.streetAddress,
    controller: _myLocController,
    decoration: InputDecoration(
      enabledBorder: OutlineInputBorder(
        borderSide: const BorderSide(color: Colors.white),
        borderRadius: BorderRadius.circular(12),
      ),
      focusedBorder: OutlineInputBorder(
        borderSide:
          const BorderSide(color: Colors.deepPurple),
        borderRadius: BorderRadius.circular(12),
      ),
      fillColor: Colors.grey[200],
      filled: true,
      hintText: 'Address',
      border: InputBorder.none,
    ),
  ),
),
const SizedBox(
  height: 10,
),

//Contact Number
Padding(

```

```

padding: const EdgeInsets.symmetric(
  horizontal: 25,
),
child: TextFormField(
  validator: (value) {
    if (value == null ||
        value.isEmpty ||
        value.length != 10) {
      return 'Please fill proper contact number';
    }
    return null;
  },
  keyboardType: TextInputType.number,
  controller: _contactNumberController,
  decoration: InputDecoration(
    enabledBorder: OutlineInputBorder(
      borderSide: const BorderSide(color: Colors.white),
      borderRadius: BorderRadius.circular(12),
    ),
    focusedBorder: OutlineInputBorder(
      borderSide:
        const BorderSide(color: Colors.deepPurple),
      borderRadius: BorderRadius.circular(12),
    ),
    fillColor: Colors.grey[200],
    filled: true,
    hintText: 'Phone Number',
    border: InputBorder.none,
  ),
),
const SizedBox(
  height: 10,
),
//Email Field

//password field
Padding(
  padding: const EdgeInsets.symmetric(
    horizontal: 25,
  ),
  child: Container(
    height: 55,
    decoration: BoxDecoration(
      borderRadius: BorderRadius.circular(12),
      border: Border.all(color: Colors.white),
      color: Colors.grey[200],
    ),
  ),
  child: Padding(
    padding: const EdgeInsets.symmetric(horizontal: 12),

```

```

child: Row(
  mainAxisAlignment: MainAxisAlignment.spaceBetween,
  children: [
    const Text('Select the medical test'),
    DropdownButton<String>(
      value: dropdownValue,
      icon: const Icon(Icons.expand_more),
      elevation: 16,
      style:
        const TextStyle(color: Colors.deepPurple),
      underline: Container(
        height: 0,
      ),
      onChanged: (String? newValue) {
        setState() {
          dropdownValue = newValue!;
        };
      },
      items: <String>[
        'Test',
        'Cholesterol Test',
        'CT Scan',
        'DNA Test'
      ].map<DropdownMenuItem<String>>(
        (String value) {
          return DropdownMenuItem<String>(
            value: value,
            child: Text(value),
          );
        }).toList(),
    ),
  ],
),
),
)),
const SizedBox(
  height: 20,
),
Row(
  mainAxisAlignment: MainAxisAlignment.spaceEvenly,
  children: [
    GestureDetector(
      onTap: () {
        Navigator.pop(context);
      },
      child: Container(
        padding: const EdgeInsets.symmetric(
          horizontal: 40, vertical: 16),
        decoration: BoxDecoration(
          color: const Color.fromARGB(255, 192, 73, 73),

```



```

        ),
      ),
    ),
  ],
),
]),
),
),
),
));
}
}

```

### **appointments\_list**

```

import 'package:cloud_firestore/cloud_firestore.dart';
import 'package:firebase_auth/firebase_auth.dart';
import 'package:flutter/material.dart';
import 'package:google_fonts/google_fonts.dart';
import 'package:medes/model/appointments_data.dart';

class AppointmentList extends StatefulWidget {
  const AppointmentList({Key? key}) : super(key: key);

  @override
  State<AppointmentList> createState() => _AppointmentListState();
}

class _AppointmentListState extends State<AppointmentList> {
  User? user = FirebaseAuth.instance.currentUser;
  List<String> docIDs = [];
  Future getDocIds() async {
    await FirebaseFirestore.instance
      .collection('users')
      .doc(user!.uid)
      .collection('appointments')
      .get()
      // ignore: avoid_function_literals_in_foreach_calls
      .then((snapshot) => snapshot.docs.forEach((element) {
        docIDs.add(element.reference.id);
      }));
  }

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      body: SafeArea(
        child: Column(

```

```

crossAxisAlignment: CrossAxisAlignment.start,
children: [
  Padding(
    padding: const EdgeInsets.only(
      left: 25, right: 25, bottom: 10, top: 25),
    child: Text(
      "Appointments",
      style: GoogleFonts.raleway(
        fontSize: 24,
        color: Colors.black,
        fontWeight: FontWeight.bold),
    ),
  ),
  Expanded(
    child: FutureBuilder(
      future: getDocIds(),
      builder: (context, snapshot) {
        return ListView.builder(
          itemCount: docIDs.length,
          itemBuilder: (context, index) {
            return AppointmentData(documentId: docIDs[index]);
          },
        );
      },
    ),
  ),
],
),
);
}
}

```

#### **bottom\_nav\_bar:**

```

import 'package:firebase_auth/firebase_auth.dart';
import 'package:flutter/material.dart';
import 'package:google_nav_bar/google_nav_bar.dart';
import 'package:medes/pages/appointments_list.dart';
import 'package:medes/pages/home_page.dart';
import 'package:medes/pages/news_article_page.dart';
import 'package:medes/pages/profile_page.dart';

class BottomNavBar extends StatefulWidget {
  const BottomNavBar({Key? key}) : super(key: key);

  @override
  State<BottomNavBar> createState() => _BottomNavBarState();
}

class _BottomNavBarState extends State<BottomNavBar> {

```

```

int _selectedIndex = 0;
final pages = const [
  HomePage(),
  NewsArticlePage(),
  AppointmentList(),
  ProfilePage(),
];
final user = FirebaseAuth.instance.currentUser!;
@override
Widget build(BuildContext context) {
  return Scaffold(
    bottomNavigationBar: Container(
      decoration: BoxDecoration(
        color: Colors.white,
        boxShadow: [
          BoxShadow(
            color: Colors.purple[200]!.withOpacity(0.2),
            blurRadius: 25.0,
          ),
        ],
      ),
    child: Padding(
      padding: const EdgeInsets.symmetric(horizontal: 15, vertical: 10),
      child: GNav(
        gap: 10,
        curve: Curves.fastOutSlowIn,

        color: Colors.black, // unselected icon color
        activeColor: Colors.deepPurple,
        iconSize: 24,
        tabBackgroundColor: Colors.purple.withOpacity(0.1),
        tabActiveBorder: Border.all(color: Colors.grey[100]!, width: 1),

        padding: const EdgeInsets.symmetric(horizontal: 20, vertical: 5),
        selectedIndex: _selectedIndex,
        onTabChange: (index) {
          setState() {
            _selectedIndex = index;
          };
        },
        tabs: const [
          GButton(
            icon: Icons.home,
            text: 'Home',
          ),
          GButton(
            icon: Icons.article,
            text: 'Articles',
          ),
          GButton(

```

```

        icon: Icons.event_note,
        text: 'Appointments',
      ),
      GButton(
        icon: Icons.person,
        text: 'Profile',
      )
    ],
  ),
),
),
body: pages[_selectedIndex],
);
}
}

```

### forgot\_pw\_page

```

// ignore_for_file: prefer_const_constructors

import 'package:firebase_auth/firebase_auth.dart';
import 'package:flutter/material.dart';

class ForgotPasswordPage extends StatefulWidget {
  const ForgotPasswordPage({Key? key}) : super(key: key);

  @override
  State<ForgotPasswordPage> createState() => _ForgotPasswordPageState();
}

class _ForgotPasswordPageState extends State<ForgotPasswordPage> {
  final TextEditingController _emailController = TextEditingController();

  @override
  void dispose() {
    _emailController.dispose();
    super.dispose();
  }

  Future passwordReset() async {
    try {
      await FirebaseAuth.instance
        .sendPasswordResetEmail(email: _emailController.text.trim());
      ScaffoldMessenger.of(context).showSnackBar(SnackBar(
        content: Text('Password reset link sent! check your email'),
      ));
    } on FirebaseException catch (e) {
      ScaffoldMessenger.of(context).showSnackBar(SnackBar(
        content: Text('${e.message}'),
      ));
    }
  }
}

```



```

}

@override
Widget build(BuildContext context) {
  return Scaffold(
    appBar: AppBar(
      backgroundColor: Colors.deepPurple[200],
      elevation: 0,
    ),
    body: Column(
      mainAxisAlignment: MainAxisAlignment.center,
      children: [
        const Padding(
          padding: EdgeInsets.symmetric(horizontal: 25),
          child: Text(
            'Enter your email and we will send you a password reset link.',
            textAlign: TextAlign.center,
            style: TextStyle(
              fontSize: 20,
            ),
          ),
        ),
        SizedBox(
          height: 20,
        ),
        Padding(
          padding: const EdgeInsets.symmetric(
            horizontal: 25,
          ),
          child: TextField(
            controller: _emailController,
            decoration: InputDecoration(
              hintText: 'Email ID',
              filled: true,
              fillColor: Colors.grey[200],
              enabledBorder: OutlineInputBorder(
                borderSide: BorderSide(color: Colors.white),
                borderRadius: BorderRadius.circular(12),
              ),
              focusedBorder: OutlineInputBorder(
                borderRadius: BorderRadius.circular(12),
                borderSide: BorderSide(color: Color(0xFF673AB7)),
              ),
            ),
          ),
        ),
        SizedBox(
          height: 10,
        ),
        MaterialButton(

```

```

        child: Text('Reset Password'),
        color: Colors.deepPurple[200],
        onPressed: passwordReset,
      ),
    ],
  ),
);
}
}

```

### home\_appointment\_form:

```

import 'package:cloud_firestore/cloud_firestore.dart';
import 'package:firebase_auth/firebase_auth.dart';
import 'package:flutter/material.dart';
import 'package:google_fonts/google_fonts.dart';

class HomeAppointment extends StatefulWidget {
  const HomeAppointment({Key? key}) : super(key: key);

  @override
  State<HomeAppointment> createState() => _HomeAppointmentState();
}

class _HomeAppointmentState extends State<HomeAppointment> {
  final _nameController = TextEditingController();
  final _myLocController = TextEditingController();
  final _contactNumberController = TextEditingController();
  DateTime currentDate = DateTime.now();

  Future<void> _selectDate(BuildContext context) async {
    final DateTime? pickedDate = await showDatePicker(
      context: context,
      initialDate: currentDate,
      firstDate: DateTime.now().subtract(
        const Duration(days: 0),
      ),
      lastDate: DateTime(2030));
    if (pickedDate != null && pickedDate != currentDate) {
      setState(() {
        currentDate = pickedDate;
      });
    }
  }

  String dropdownValue = 'Test';

  Future createAppointment() async {
    User? user = FirebaseAuth.instance.currentUser;
    await FirebaseFirestore.instance
      .collection('users')

```

```

        .doc(user!.uid)
        .collection('appointments')
        .add({
          'name': _nameController.text.trim(),
          'address': _myLocController.text.trim(),
          'test-type': dropdownValue.toString().trim(),
          'contact_info': _contactNumberController.text.trim(),
          'date': currentDate.toString().substring(0, 10),
        });
ScaffoldMessenger.of(context).showSnackBar(
  const SnackBar(content: Text("Form submitted successfully.")));
Navigator.pop(context);
dispose();
}

@override
void dispose() {
  _nameController.dispose();
  _myLocController.dispose();
  _contactNumberController.dispose();
  super.dispose();
}

final _formKey = GlobalKey<FormState>();
@override
Widget build(BuildContext context) {
  return Scaffold(
    body: Scaffold(
      backgroundColor: Colors.grey[300],
      body: SafeArea(
        child: Center(
          child: SingleChildScrollView(
            child: Form(
              key: _formKey,
              child: Column(
                mainAxisAlignment: MainAxisAlignment.center,
                children: [
                  Text(
                    'Home Appointment',
                    style: GoogleFonts.poppins(
                      fontSize: 42, fontWeight: FontWeight.w200),
                  ),
                  const SizedBox(height: 15),
                  Text(
                    'Fill below with person\'s detail!',
                    style: GoogleFonts.poppins(
                      fontSize: 20, fontWeight: FontWeight.w300),
                  ),
                  const SizedBox(height: 50),

```

```

Padding(
  padding: const EdgeInsets.symmetric(horizontal: 25),
  child: Container(
    height: 55,
    decoration: BoxDecoration(
      borderRadius: BorderRadius.circular(12),
      border: Border.all(color: Colors.white),
      color: Colors.grey[200],
    ),
    child: Row(
      mainAxisAlignment: MainAxisAlignment.spaceAround,
      children: [
        Text(
          currentDate.toString().substring(0, 10),
          style: GoogleFonts.poppins(fontSize: 16),
        ),
        const SizedBox(
          width: 100,
        ),
        MaterialButton(
          textColor: Colors.deepPurple,
          elevation: 0,
          onPressed: () => _selectDate(context),
          child: Text(
            'Select Date',
            style: GoogleFonts.poppins(fontSize: 14),
          ),
        ),
      ],
    ),
  ),
  const SizedBox(
    height: 10,
  ),
  //Name Field
  Padding(
    padding: const EdgeInsets.symmetric(
      horizontal: 25,
    ),
    child: TextFormField(
      validator: (value) {
        if (value == null || value.isEmpty) {
          return 'Please fill your name';
        }
        return null;
      },
      keyboardType: TextInputType.name,
      controller: _nameController,
      decoration: InputDecoration(

```

```

        enabledBorder: OutlineInputBorder(
          borderSide: const BorderSide(color: Colors.white),
          borderRadius: BorderRadius.circular(12),
        ),
        focusedBorder: OutlineInputBorder(
          borderSide:
            const BorderSide(color: Colors.deepPurple),
          borderRadius: BorderRadius.circular(12),
        ),
        fillColor: Colors.grey[200],
        filled: true,
        hintText: 'Name',
        border: InputBorder.none,
      ),
    ),
  ),
  const SizedBox(
    height: 10,
  ),

  //Address Field
  Padding(
    padding: const EdgeInsets.symmetric(
      horizontal: 25,
    ),
    child: TextFormField(
      validator: (value) {
        if (value == null || value.isEmpty) {
          return 'Please fill address';
        }
        return null;
      },
      keyboardType: TextInputType.streetAddress,
      controller: _myLocController,
      decoration: InputDecoration(
        enabledBorder: OutlineInputBorder(
          borderSide: const BorderSide(color: Colors.white),
          borderRadius: BorderRadius.circular(12),
        ),
        focusedBorder: OutlineInputBorder(
          borderSide:
            const BorderSide(color: Colors.deepPurple),
          borderRadius: BorderRadius.circular(12),
        ),
        fillColor: Colors.grey[200],
        filled: true,
        hintText: 'Address',
        border: InputBorder.none,
      ),
    ),
  ),

```

```

    ),
    const SizedBox(
      height: 10,
    ),

    //Contact Number
    Padding(
      padding: const EdgeInsets.symmetric(
        horizontal: 25,
      ),
      child: TextFormField(
        validator: (value) {
          if (value == null ||
            value.isEmpty ||
            value.length != 10) {
            return 'Please fill proper contact number';
          }
          return null;
        },
        keyboardType: TextInputType.number,
        controller: _contactNumberController,
        decoration: InputDecoration(
          enabledBorder: OutlineInputBorder(
            borderSide: const BorderSide(color: Colors.white),
            borderRadius: BorderRadius.circular(12),
          ),
          focusedBorder: OutlineInputBorder(
            borderSide:
              const BorderSide(color: Colors.deepPurple),
            borderRadius: BorderRadius.circular(12),
          ),
          fillColor: Colors.grey[200],
          filled: true,
          hintText: 'Phone Number',
          border: InputBorder.none,
        ),
      ),
    ),
    const SizedBox(
      height: 10,
    ),
    //Email Field

    //password field
    Padding(
      padding: const EdgeInsets.symmetric(
        horizontal: 25,
      ),
      child: Container(
        height: 55,

```

```

decoration: BoxDecoration(
  borderRadius: BorderRadius.circular(12),
  border: Border.all(color: Colors.white),
  color: Colors.grey[200],
),
child: Padding(
  padding: const EdgeInsets.symmetric(horizontal: 12),
  child: Row(
    mainAxisAlignment: MainAxisAlignment.spaceBetween,
    children: [
      const Text('Select the medical test'),
      DropdownButton<String>(
        value: dropdownValue,
        icon: const Icon(Icons.expand_more),
        elevation: 16,
        style:
          const TextStyle(color: Colors.deepPurple),
        underline: Container(
          height: 0,
        ),
        onChanged: (String? newValue) {
          setState() {
            dropdownValue = newValue!;
          };
        },
        items: <String>[
          'Test',
          'Cholesterol Test',
          'CT Scan',
          'DNA Test'
        ].map<DropdownMenuItem<String>>(<
          (String value) {
            return DropdownMenuItem<String>(
              value: value,
              child: Text(value),
            );
          }).toList(),
      ),
    ],
  ),
),
)),
const SizedBox(
  height: 20,
),
Row(
  mainAxisAlignment: MainAxisAlignment.spaceEvenly,
  children: [
    GestureDetector(
      onTap: () {

```

```

Navigator.pop(context);
},
child: Container(
  padding: const EdgeInsets.symmetric(
    horizontal: 40, vertical: 16),
  decoration: BoxDecoration(
    color: const Color.fromARGB(255, 192, 73, 73),
    borderRadius: BorderRadius.circular(12),
    boxShadow: [
      BoxShadow(
        offset: const Offset(0, 10),
        blurRadius: 50,
        color: const Color.fromARGB(255, 192, 73, 73)
          .withOpacity(0.23),
      ),
    ],
  ),
child: const Center(
  child: Text(
    'Cancel',
    style: TextStyle(
      color: Colors.white,
      fontWeight: FontWeight.bold,
      fontSize: 16,
    ),
  ),
),
),
GestureDetector(
  onTap: () {
    if (_formKey.currentState!.validate()) {
      createAppointment();
    }
  },
child: Container(
  padding: const EdgeInsets.symmetric(
    horizontal: 40, vertical: 16),
  decoration: BoxDecoration(
    color: Colors.deepPurple,
    borderRadius: BorderRadius.circular(12),
    boxShadow: [
      BoxShadow(
        offset: const Offset(0, 10),
        blurRadius: 50,
        color: Colors.deepPurple.withOpacity(0.23),
      ),
    ],
  ),
child: const Center(

```





```

setState() {
  _username = name;
});
} catch (e) {
  _username = 'invalid';
}
}

final List testPacks = [];
Future getTestId() async {
  await FirebaseFirestore.instance
    .collection("testPack")
    .get()
    // ignore: avoid_function_literals_in_foreach_calls
    .then((snapshot) => snapshot.docs.forEach((element) {
      testPacks.add(element.reference.id);
    }));
}

@override
void initState() {
  getTestId();
  getUserData();
  super.initState();
}

@override
Widget build(BuildContext context) {
  Size size = MediaQuery.of(context).size;
  return Padding(
    padding: const EdgeInsets.only(top: 25, left: 25, right: 25),
    child: Column(
      children: [
        SizedBox(
          height: size.height / 2.7,
          child: Column(
            crossAxisAlignment: CrossAxisAlignment.start,
            children: [
              SizedBox(
                height: size.height / 20,
              ),
              Text(
                "Hi, $_username 🐼",
                style: GoogleFonts.poppins(
                  fontSize: 25,
                  color: Colors.black87,
                  fontWeight: FontWeight.bold),
              ),
              SizedBox(
                height: size.height / 25,

```

```

),
Row(
  mainAxisAlignment: MainAxisAlignment.spaceEvenly,
  children: [
    InkWell(
      onTap: () {
        Navigator.push(
          context,
          MaterialPageRoute(
            builder: (context) {
              return AppointmentForm();
            },
          ),
        );
      },
      child: Ink(
        padding: EdgeInsets.all(20),
        decoration: BoxDecoration(
          boxShadow: [
            BoxShadow(
              offset: const Offset(0, 10),
              blurRadius: 20,
              color: Colors.deepPurple.withOpacity(0.5),
            ),
          ],
          gradient: LinearGradient(
            begin: Alignment.topRight,
            end: Alignment.bottomLeft,
            colors: const [
              Color.fromARGB(255, 184, 149, 246),
              Colors.deepPurple,
            ],
          ),
          borderRadius: BorderRadius.circular(12)),
        height: 150,
        child: Column(
          crossAxisAlignment: CrossAxisAlignment.start,
          children: const [
            Icon(
              Icons.add,
              color: Colors.white,
              size: 30,
            ),
            SizedBox(
              height: 20,
            ),
            Text(
              "Clinic Visit",
              style: TextStyle(
                color: Colors.white,

```

```

        fontSize: 20,
        fontWeight: FontWeight.bold),
    ),
    SizedBox(
      height: 5,
    ),
    Text("Make an appointment",
      style: TextStyle(
        color: Colors.white,
      )),
    ],
  ),
),
SizedBox(
  width: 8,
),
InkWell(
  onTap: () {
    Navigator.push(
      context,
      MaterialPageRoute(
        builder: (context) {
          return HomeAppointment();
        },
      ),
    );
  },
  child: Ink(
    padding: EdgeInsets.all(20),
    decoration: BoxDecoration(
      boxShadow: [
        BoxShadow(
          offset: const Offset(0, 10),
          blurRadius: 20,
          color: Colors.deepPurple.withOpacity(0.5),
        ),
      ],
      color: Colors.blue,
      gradient: LinearGradient(
        begin: Alignment.topRight,
        end: Alignment.bottomLeft,
        colors: const [
          Color.fromARGB(255, 184, 149, 246),
          Colors.deepPurple,
        ],
      ),
      borderRadius: BorderRadius.circular(12)),
    height: 150,
    child: Column(
      crossAxisAlignment: CrossAxisAlignment.start,

```



```

    )),
  ),
],
),
);
}
}

```

login\_page:

```

// ignore_for_file: prefer_const_constructors, prefer_const_literals_to_create_immutables
import 'package:firebase_auth/firebase_auth.dart';
import 'package:flutter/material.dart';
import 'package:google_fonts/google_fonts.dart';
import 'package:medes/pages/forgot_pw_page.dart';

class LoginPage extends StatefulWidget {
  final VoidCallback showRegisterPage;

  const LoginPage({Key? key, required this.showRegisterPage}) : super(key: key);

  @override
  State<LoginPage> createState() => _LoginPageState();
}

class _LoginPageState extends State<LoginPage> {
  bool _isHidden = true;

  final _emailController = TextEditingController();
  final _passwordController = TextEditingController();

  Future signIn() async {
    showDialog(
      context: context,
      builder: (context) {
        return Center(
          child: CircularProgressIndicator(
            color: Colors.deepPurple,
          ));
      });
    try {
      await FirebaseAuth.instance.signInWithEmailAndPassword(
        email: _emailController.text.trim(),
        password: _passwordController.text.trim(),
      );
    } on FirebaseAuthException catch (e) {
      ScaffoldMessenger.of(context).showSnackBar(SnackBar(
        content: Text('${e.message}'),
      ));
    }
  }
}

```

```

    }
    Navigator.of(context, rootNavigator: true).pop();
  }

  void _togglePasswordView() {
    setState(() {
      _isHidden = !_isHidden;
    });
  }
}

@override
void dispose() {
  _emailController.dispose();
  _passwordController.dispose();
  super.dispose();
}

final _formKey = GlobalKey<FormState>();
@override
Widget build(BuildContext context) {
  return Scaffold(
    backgroundColor: Colors.grey[300],
    body: SafeArea(
      child: Center(
        child: SingleChildScrollView(
          child: Form(
            key: _formKey,
            child: Column(
              mainAxisAlignment: MainAxisAlignment.center,
              children: [
                Image.asset(
                  'assets/logo/logo.png',
                  height: 100,
                ),
                SizedBox(
                  height: 50,
                ),
                Text(
                  'Hello Again!',
                  style: GoogleFonts.bebasNeue(fontSize: 54),
                ),
                SizedBox(height: 10),
                Text(
                  'Welcome back, you\'ve been missed!',
                  style: TextStyle(
                    fontSize: 20,
                  ),
                ),
                SizedBox(height: 50),

```









```
}  
}
```

```

news_article_page:
// ignore_for_file: unused_local_variable

import 'package:flutter/material.dart';
import 'package:google_fonts/google_fonts.dart';
import 'package:medes/model/article_model.dart';
import 'package:medes/services/health_articles.dart';
import 'package:medes/views/article_view_page.dart';

class NewsArticlePage extends StatefulWidget {
  const NewsArticlePage({Key? key}) : super(key: key);

  @override
  State<NewsArticlePage> createState() => _NewsArticlePageState();
}

class _NewsArticlePageState extends State<NewsArticlePage> {
  List<HealthApiModel>? articlesList;
  bool isLoading = true;
  @override
  void initState() {
    getArticles().then((value) => {
      setState() {
        if (value.isNotEmpty) {
          articlesList = value;
          isLoading = false;
        } else {}
      }
    });
    super.initState();
  }

  @override
  Widget build(BuildContext context) {
    final Size size = MediaQuery.of(context).size;
    return SafeArea(
      child: Scaffold(
        appBar: AppBar(
          iconTheme: const IconThemeData(
            color: Colors.deepPurple, //change your color here
          ),
          backgroundColor: Colors.white,
          centerTitle: true,
          title: const Text(
            'Trending Articles',
            style: TextStyle(
              color: Colors.black,

```

```

        fontSize: 20,
      ),
    ),
    elevation: 0,
  ),
  body: Column(
    mainAxisAlignment: MainAxisAlignment.max,
    mainAxisAlignment: MainAxisAlignment.center,
    children: [
      isLoading
        ? const Center(
            child: CircularProgressIndicator(),
          )
        : Expanded(
            child: ListView.builder(
              itemCount: articlesList!.length,
              itemBuilder: (context, index) {
                return listItems(size, articlesList![index]);
              },
            ),
          ),
    ],
  ),
);
}

```

```

Widget listItems(Size size, HealthApiModel model) {
  return Padding(
    padding: const EdgeInsets.symmetric(vertical: 12.5),
    child: GestureDetector(
      onTap: () => Navigator.of(context).push(
        MaterialPageRoute(
          builder: (_) => ArticleReadPage(
            model: model,
          ),
        ),
      ),
    child: Container(
      height: size.height / 3,
      width: size.width / 1.15,
      decoration: BoxDecoration(
        color: Colors.white,
        boxShadow: [
          BoxShadow(
            offset: const Offset(0, 10),

```

```

        borderRadius: 15,
        color: Colors.deepPurple.withOpacity(0.15),
      ),
    ],
  ),
  child: Padding(
    padding: const EdgeInsets.symmetric(horizontal: 25, vertical: 14),
    child: Column(
      mainAxisAlignment: MainAxisAlignment.spaceBetween,
      children: [
        Container(
          height: 160,
          width: double.infinity,
          decoration: BoxDecoration(
            borderRadius: BorderRadius.circular(12),
            color: Colors.white,
          ),
        ),
        child: model.imageUrl != ""
          ? ClipRRect(
              borderRadius: BorderRadius.circular(12),
              child: Image.network(
                model.imageUrl,
                fit: BoxFit.fitWidth,
              ),
            )
          : ClipRRect(
              borderRadius: BorderRadius.circular(12),
              child: Image.network(
                'https://eagle-sensors.com/wp-content/uploads/unavailable-image.jpg',
                fit: BoxFit.fitWidth,
              ),
            ),
      ],
    ),
  ),
  model.title != ""
    ? Text(
        model.title,
        style: GoogleFonts.poppins(
          fontSize: 16,
          fontWeight: FontWeight.bold,
        ),
      )
    : const Text("Title is unavailable"),
  Row(
    mainAxisAlignment: MainAxisAlignment.start,
    children: [
      model.author != ""

```

```

        ? Text(
          '- ' + model.author,
          style: GoogleFonts.poppins(
            fontSize: 16,
            fontStyle: FontStyle.italic,
          ),
        )
        : const Text('Author is unavailable'),
      ],
    ),
  ],
),
),
),
),
);
}
}

```

profile\_page:

```

import 'package:cloud_firestore/cloud_firestore.dart';
import 'package:firebase_auth/firebase_auth.dart';
import 'package:flutter/material.dart';
import 'package:google_fonts/google_fonts.dart';

class ProfilePage extends StatefulWidget {
  const ProfilePage({Key? key}) : super(key: key);

  @override
  State<ProfilePage> createState() => _ProfilePageState();
}

class _ProfilePageState extends State<ProfilePage> {
  User? user = FirebaseAuth.instance.currentUser;
  String _username = "";
  String _userEmail = "";
  String _userAddress = "";
  String _userMobileNo = "";
  final userCollection = FirebaseFirestore.instance.collection('users');
  Future getUserData() async {
    try {
      DocumentSnapshot ds = await userCollection.doc(user!.uid).get();

      String name = ds.get('Name');
      String address = ds.get('Address');
      String email = ds.get('Email');
      String phoneNo = ds.get('Phone_No');
    }
  }
}

```

```

setState() {
  _username = name;
  _userAddress = address;
  _userEmail = email;
  _userMobileNo = phoneNo;
});
} catch (e) {
  _username = 'invalid';
}
}

@override
void initState() {
  getUserData();
  super.initState();
}

@override
Widget build(BuildContext context) {
  TextEditingController nameController = TextEditingController()
    ..text = _username;
  TextEditingController emailController = TextEditingController()
    ..text = _userEmail;
  TextEditingController addressController = TextEditingController()
    ..text = _userAddress;
  TextEditingController phoneController = TextEditingController()
    ..text = _userMobileNo;
  return Scaffold(
    body: SingleChildScrollView(
      child: Column(
        children: [
          Container(
            width: double.infinity,
            height: 250,
            decoration: const BoxDecoration(
              gradient: LinearGradient(
                begin: Alignment.topCenter,
                end: Alignment.bottomCenter,
                colors: [
                  Color.fromARGB(255, 152, 95, 250),
                  Colors.deepPurple,
                ],
              ),
            child: Center(
              child: Padding(
                padding: const EdgeInsets.only(top: 12),

```



```

child: Text(
  nameController.text,
  style:
    GoogleFonts.raleway(fontSize: 35, color: Colors.white),
),
),
),
),
Padding(
  padding: const EdgeInsets.symmetric(horizontal: 25),
  child: Column(
    mainAxisAlignment: MainAxisAlignment.start,
    children: [
      const SizedBox(
        height: 60,
      ),
      Row(
        children: [
          Padding(
            padding: const EdgeInsets.all(4),
            child: Text(
              'My Email :',
              style: GoogleFonts.poppins(
                fontSize: 16,
                color: Colors.deepPurple[500],
              ),
            ),
          ),
        ],
      ),
      TextField(
        keyboardType: TextInputType.name,
        controller: emailController,
        readOnly: true,
        decoration: InputDecoration(
          enabledBorder: OutlineInputBorder(
            borderSide: const BorderSide(color: Colors.white),
            borderRadius: BorderRadius.circular(12),
          ),
          fillColor: Colors.grey[200],
          filled: true,
          border: InputBorder.none,
        ),
        style: GoogleFonts.poppins(
          fontSize: 18,
        ),
      ),
    ],
  ),
),

```

```

    ),
    const SizedBox(
      height: 14,
    ),
    Row(
      children: [
        Padding(
          padding: const EdgeInsets.all(4),
          child: Text(
            'My Address :',
            style: GoogleFonts.poppins(
              fontSize: 16,
              color: Colors.deepPurple[500],
            ),
          ),
        ),
      ],
    ),
    TextField(
      style: GoogleFonts.poppins(
        fontSize: 18,
      ),
      maxLines: null,
      keyboardType: TextInputType.name,
      controller: addressController,
      readOnly: true,
      decoration: InputDecoration(
        enabledBorder: OutlineInputBorder(
          borderSide: const BorderSide(color: Colors.white),
          borderRadius: BorderRadius.circular(12),
        ),
        fillColor: Colors.grey[200],
        filled: true,
        border: InputBorder.none,
      ),
    ),
    const SizedBox(
      height: 14,
    ),
    Row(
      children: [
        Padding(
          padding: const EdgeInsets.all(4),
          child: Text(
            'My Contact Number :',
            style: GoogleFonts.poppins(

```

```

        color: Colors.deepPurple[500],
        fontSize: 16,
      ),
    ),
  ],
),
TextField(
  style: GoogleFonts.poppins(
    fontSize: 18,
  ),
  keyboardType: TextInputType.name,
  controller: phoneController,
  readOnly: true,
  decoration: InputDecoration(
    enabledBorder: OutlineInputBorder(
      borderSide: const BorderSide(color: Colors.white),
      borderRadius: BorderRadius.circular(12),
    ),
    fillColor: Colors.grey[200],
    filled: true,
    border: InputBorder.none,
  ),
),
const SizedBox(
  height: 10,
),
Padding(
  padding: const EdgeInsets.all(4),
  child: Text(
    '*To change user details, email us at users@medes.com',
    style: GoogleFonts.poppins(
      color: Colors.grey[500],
      fontSize: 14,
    ),
  ),
),
const SizedBox(
  height: 30,
),
Padding(
  padding: const EdgeInsets.symmetric(
    horizontal: 100,
  ),
  child: GestureDetector(
    onTap: () {

```



```

@override
State<RegisterPage> createState() => _RegisterPageState();
}

class _RegisterPageState extends State<RegisterPage> {
  final _emailController = TextEditingController();
  final _passwordController = TextEditingController();
  final _confirmPasswordController = TextEditingController();
  final _nameController = TextEditingController();
  final _contactNumberController = TextEditingController();
  final _addressController = TextEditingController();

  @override
  void dispose() {
    _emailController.dispose();
    _passwordController.dispose();
    _confirmPasswordController.dispose();
    _nameController.dispose();
    _addressController.dispose();
    _contactNumberController.dispose();
    super.dispose();
  }

  Future signUp() async {
    showDialog(
      context: context,
      builder: (context) {
        return const Center(
          child: CircularProgressIndicator(
            color: Colors.deepPurple,
          ));
      });
    //user register
    if (passwordConfirmed()) {
      await FirebaseAuth.instance.createUserWithEmailAndPassword(
        email: _emailController.text.trim(),
        password: _passwordController.text.trim());
    }
    User? user = FirebaseAuth.instance.currentUser;
    //user details
    addUserDetails(
      user!.uid,
      _nameController.text.trim(),
      _emailController.text.trim(),
      _addressController.text.trim(),

```

```

        _contactNumberController.text.trim(),
    );
    Navigator.of(context).pop();
}

User? user = FirebaseAuth.instance.currentUser;
Future addUserDetails(String userId, String name, String email,
    String address, String phone) async {
  await FirebaseFirestore.instance.collection('users').doc(userId).set({
    'Name': name,
    'Email': email,
    'Address': address,
    'Phone_No': phone,
  });
}

bool passwordConfirmed() {
  if (_passwordController.text.trim() ==
    _confirmPasswordController.text.trim()) {
    return true;
  } else {
    return false;
  }
}

final _formKey = GlobalKey<FormState>();
@override
Widget build(BuildContext context) {
  return Scaffold(
    backgroundColor: Colors.grey[300],
    body: SafeArea(
      child: Center(
        child: SingleChildScrollView(
          child: Form(
            key: _formKey,
            child: Column(
              mainAxisAlignment: MainAxisAlignment.center,
              children: [
                Text(
                  'Hello There!',
                  style: GoogleFonts.bebasNeue(fontSize: 54),
                ),
                const SizedBox(height: 10),
                const Text(
                  'Register below with your details!',
                  style: TextStyle(

```

```

        fontSize: 20,
      ),
    ),
    const SizedBox(height: 50),
    //Name Field
    Padding(
      padding: const EdgeInsets.symmetric(
        horizontal: 25,
      ),
      child: TextFormField(
        validator: (value) {
          if (value == null || value.isEmpty) {
            return 'Please fill your name';
          }
          return null;
        },
        keyboardType: TextInputType.name,
        controller: _nameController,
        decoration: InputDecoration(
          enabledBorder: OutlineInputBorder(
            borderSide: const BorderSide(color: Colors.white),
            borderRadius: BorderRadius.circular(12),
          ),
          focusedBorder: OutlineInputBorder(
            borderSide:
              const BorderSide(color: Colors.deepPurple),
            borderRadius: BorderRadius.circular(12),
          ),
          fillColor: Colors.grey[200],
          filled: true,
          hintText: 'Name',
          border: InputBorder.none,
        ),
      ),
    ),
    const SizedBox(
      height: 10,
    ),

    //Address Field
    Padding(
      padding: const EdgeInsets.symmetric(
        horizontal: 25,
      ),
      child: TextFormField(
        validator: (value) {

```

```

        if (value == null || value.isEmpty) {
          return 'Please fill your address';
        }
        return null;
      },
      keyboardType: TextInputType.streetAddress,
      controller: _addressController,
      decoration: InputDecoration(
        enabledBorder: OutlineInputBorder(
          borderSide: const BorderSide(color: Colors.white),
          borderRadius: BorderRadius.circular(12),
        ),
        focusedBorder: OutlineInputBorder(
          borderSide:
            const BorderSide(color: Colors.deepPurple),
          borderRadius: BorderRadius.circular(12),
        ),
        fillColor: Colors.grey[200],
        filled: true,
        hintText: 'Address',
        border: InputBorder.none,
      ),
    ),
  ),
  const SizedBox(
    height: 10,
  ),

  //Contact Number
  Padding(
    padding: const EdgeInsets.symmetric(
      horizontal: 25,
    ),
    child: TextFormField(
      validator: (value) {
        if (value == null || value.isEmpty) {
          return 'Please fill your contact number';
        }
        return null;
      },
      keyboardType: TextInputType.number,
      controller: _contactNumberController,
      decoration: InputDecoration(
        enabledBorder: OutlineInputBorder(
          borderSide: const BorderSide(color: Colors.white),
          borderRadius: BorderRadius.circular(12),

```



```

    ),
    focusedBorder: OutlineInputBorder(
      borderSide:
        const BorderSide(color: Colors.deepPurple),
      borderRadius: BorderRadius.circular(12),
    ),
    fillColor: Colors.grey[200],
    filled: true,
    hintText: 'Phone Number',
    border: InputBorder.none,
  ),
),
),
const SizedBox(
  height: 10,
),
//Email Field
Padding(
  padding: const EdgeInsets.symmetric(
    horizontal: 25,
  ),
  child: TextFormField(
    validator: (value) {
      if (value == null || value.isEmpty) {
        return 'Please fill your email ID';
      }
      return null;
    },
    keyboardType: TextInputType.emailAddress,
    controller: _emailController,
    decoration: InputDecoration(
      enabledBorder: OutlineInputBorder(
        borderSide: const BorderSide(color: Colors.white),
        borderRadius: BorderRadius.circular(12),
      ),
      focusedBorder: OutlineInputBorder(
        borderSide:
          const BorderSide(color: Colors.deepPurple),
        borderRadius: BorderRadius.circular(12),
      ),
      fillColor: Colors.grey[200],
      filled: true,
      hintText: 'Email ID',
      border: InputBorder.none,
    ),
  ),
),

```

```

    ),
    const SizedBox(
      height: 10,
    ),

    //password field
    Padding(
      padding: const EdgeInsets.symmetric(
        horizontal: 25,
      ),
      child: TextFormField(
        validator: (value) {
          if (value == null || value.isEmpty) {
            return 'Please fill your password';
          }
          return null;
        },
        controller: _passwordController,
        obscureText: true,
        decoration: InputDecoration(
          enabledBorder: OutlineInputBorder(
            borderSide: const BorderSide(color: Colors.white),
            borderRadius: BorderRadius.circular(12),
          ),
          focusedBorder: OutlineInputBorder(
            borderSide:
              const BorderSide(color: Colors.deepPurple),
            borderRadius: BorderRadius.circular(12),
          ),
          fillColor: Colors.grey[200],
          filled: true,
          hintText: 'Password',
          border: InputBorder.none,
        ),
      ),
    ),
    const SizedBox(
      height: 10,
    ),
    Padding(
      padding: const EdgeInsets.symmetric(
        horizontal: 25,
      ),
      child: TextFormField(
        validator: (value) {
          if (value == null || value.isEmpty) {

```

```

        return 'Please repeat same password';
    }
    return null;
},
controller: _confirmPasswordController,
obscureText: true,
decoration: InputDecoration(
  enabledBorder: OutlineInputBorder(
    borderSide: const BorderSide(color: Colors.white),
    borderRadius: BorderRadius.circular(12),
  ),
  focusedBorder: OutlineInputBorder(
    borderSide:
      const BorderSide(color: Colors.deepPurple),
    borderRadius: BorderRadius.circular(12),
  ),
  fillColor: Colors.grey[200],
  filled: true,
  hintText: 'Confirm Password',
  border: InputBorder.none,
),
),
),
const SizedBox(
  height: 10,
),
//Login button
Padding(
  padding: const EdgeInsets.symmetric(
    horizontal: 25,
  ),
  child: GestureDetector(
    onTap: () {
      if (_formKey.currentState!.validate()) {
        signUp();
      }
    },
    child: Container(
      padding: const EdgeInsets.all(20),
      decoration: BoxDecoration(
        color: Colors.deepPurple,
        borderRadius: BorderRadius.circular(12),
      ),
      child: const Center(
        child: Text(
          'Sign Up',

```

```

        style: TextStyle(
          color: Colors.white,
          fontWeight: FontWeight.bold,
          fontSize: 18,
        ),
      ),
    ),
  ),
),
const SizedBox(
  height: 25,
),
Row(
  mainAxisAlignment: MainAxisAlignment.center,
  children: [
    const Text(
      'Already a member!',
      style: TextStyle(fontWeight: FontWeight.bold),
    ),
    GestureDetector(
      onTap: widget.showLoginPage,
      child: const Text(
        'Login Now',
        style: TextStyle(
          color: Colors.blue,
          fontWeight: FontWeight.bold),
      ),
    ),
  ],
),
]),
),
),
),
);
}
}

```

article\_view\_page:

```

import 'package:flutter/material.dart';
import 'package:google_fonts/google_fonts.dart';
import 'package:medes/model/article_model.dart';

class ArticleReadPage extends StatelessWidget {
  final HealthApiModel model;

```

```

const ArticleReadPage({required this.model, Key? key}) : super(key: key);

@override
Widget build(BuildContext context) {
  return SafeArea(
    child: Scaffold(
      body: SingleChildScrollView(
        child: Padding(
          padding: const EdgeInsets.symmetric(horizontal: 25),
          child: Column(
            crossAxisAlignment: CrossAxisAlignment.start,
            children: [
              Padding(
                padding: const EdgeInsets.only(
                  top: 50,
                ),
                child: Text(
                  model.title,
                  style: GoogleFonts.poppins(
                    fontSize: 24, fontWeight: FontWeight.w500),
                ),
              ),
              const SizedBox(
                height: 36,
              ),
              Row(
                mainAxisAlignment: MainAxisAlignment.spaceBetween,
                children: [
                  Text(
                    '- ' + model.author,
                    style: GoogleFonts.poppins(
                      fontSize: 16,
                      fontWeight: FontWeight.w500,
                      color: Colors.deepPurple,
                    ),
                  ),
                  model.publishedAt != ""
                    ? Text(
                        model.publishedAt.substring(0, 10),
                      )
                    : const Text(""),
                ],
              ),
              const SizedBox(
                height: 20,
              ),
            ],
          ),
        ),
      ),
    ),
  );
}

```

```

ClipRRect(
  child: Image.network(
    model.imageUrl,
    fit: BoxFit.fitWidth,
  ),
),
const SizedBox(
  height: 20,
),
Text(
  model.description,
  style: GoogleFonts.poppins(
    fontSize: 16,
    fontWeight: FontWeight.w300,
  ),
),
const SizedBox(
  height: 30,
),
const Divider(
  color: Colors.deepPurple,
  height: 150,
  thickness: 1,
  indent: 100,
  endIndent: 100,
),
],
),
),
),
),
);
}
}

```

#### test\_pack\_info:

```

import 'package:cloud_firestore/cloud_firestore.dart';
import 'package:flutter/material.dart';
import 'package:flutter_svg/svg.dart';
import 'package:font_awesome_flutter/font_awesome_flutter.dart';
import 'package:google_fonts/google_fonts.dart';

// ignore: must_be_immutable
class TestInfoCard extends StatefulWidget {
  String testId;
  TestInfoCard({Key? key, required this.testId}) : super(key: key);

```

```

@override
State<TestInfoCard> createState() => _TestInfoCardState();
}

class _TestInfoCardState extends State<TestInfoCard> {
  @override
  Widget build(BuildContext context) {
    CollectionReference testInfo =
      FirebaseFirestore.instance.collection('testPack');
    return FutureBuilder<DocumentSnapshot>(
      future: testInfo.doc(widget.testId).get(),
      builder: (context, snapshot) {
        if (snapshot.connectionState == ConnectionState.done) {
          Map<String, dynamic> data =
            snapshot.data!.data() as Map<String, dynamic>;
          return Padding(
            padding: const EdgeInsets.only(bottom: 20),
            child: Container(
              decoration: BoxDecoration(
                color: Colors.white,
                boxShadow: [
                  BoxShadow(
                    offset: const Offset(0, 10),
                    blurRadius: 20,
                    color: Colors.deepPurple.withOpacity(0.2),
                  ),
                ],
                borderRadius: BorderRadius.circular(12)),
              height: 150,
              child: Padding(
                padding: const EdgeInsets.all(8.0),
                child: Row(
                  mainAxisAlignment: MainAxisAlignment.spaceAround,
                  children: [
                    Column(
                      mainAxisAlignment: MainAxisAlignment.spaceEvenly,
                      crossAxisAlignment: CrossAxisAlignment.start,
                      children: [
                        Text(
                          data['name'],
                          style: GoogleFonts.nunito(
                            fontSize: 20,
                            color: const Color(0xff050a4e),
                            fontWeight: FontWeight.w700),
                        ),

```

```

Row(
  children: [
    const FaIcon(
      FontAwesomeIcons.flaskVial,
      color: Colors.deepPurple,
      size: 14,
    ),
    const SizedBox(
      width: 10,
    ),
    Text(
      data['description'],
      style: GoogleFonts.nunito(
        fontSize: 16,
        color: const Color(0xff050a4e),
        fontWeight: FontWeight.w400),
    ),
  ],
),
const SizedBox(
  height: 5,
),
Text(
  '₹ ${data['price'].toString()}/-',
  style: GoogleFonts.nunito(
    fontSize: 18,
    color: const Color(0xff050a4e),
    fontWeight: FontWeight.w700),
),
const SizedBox(
  height: 14,
),
],
),
Column(
  mainAxisAlignment: MainAxisAlignment.spaceEvenly,
  children: [
    Image.asset(
      'assets/icons/medical-test.png',
      scale: 08,
    ),
    GestureDetector(
      onTap: () {},
      child: Container(
        padding: const EdgeInsets.all(12),
        decoration: BoxDecoration(

```





## **CHAPTER 6**

### **TEST CASES**

#### **6.1 TEST CASES FOR HOME PAGE:**

- Verify that home page is displayed after login or not.
- Verify that Username is displayed on homepage or not.
- Verify that featured health test are present on home page or not.
- Verify the home page of application on different devices.
- Verify the alignment on the home page.
- Verify that health tests displayed on home page are clickable or not.

#### **6.2 TEST CASES FOR HEALTH ARTICLE PAGE:**

- Verify that the images of articles are displayed correctly or not.
- Verify that the date of publication is displayed or not.
- Verify that product author are mentioned or not.

#### **6.3 TEST CASES FOR APPOINTMENT FORM FUNCTIONALITY:**

- Verify that the name field accepts only alphabets.
- Verify that the address field accepts alphabets numbers or symbols.
- Verify that the appointments created are visible on appointment list page.
- Verify that sorting options should be present on search results page.

#### **6.4 TEST CASES FOR PROFILE PAGE:**

- Verify that the user details are visible or not.
- Verify that there is logout button or not.

## CHAPTER 7

### BIBLIOGRAPHY

<https://flutter.dev/>

<https://console.firebase.google.com/>

<https://www.youtube.com/c/flutterdev>

<https://firebase.flutter.dev/docs/overview/>

<https://pub.dev/>

<https://stackoverflow.com/>

<https://www.youtube.com/c/FlutterMapp>

<https://www.youtube.com/channel/UC0FD2apauvegCcsvqIBceLA>