



BSCS-F22-029

03-134192-044 KUNWAR MUZZAMMIL WAHEED

03-134192-071 TAIMOOR KHAN

Visualization of Health Data Through a Virtual Avatar

In partial accomplishment of the degree requirements

Bachelor of Science in Computer Science

Supervisor: Dr. Ghulam Mustafa

Department of Computer Sciences
Bahria University, Lahore Campus

June 2023

Certificate



We accept the work included in the named report
“Visualization of Health Data Through a Virtual Avatar”
written by

KUNWAR MUZZAMMIL WAHEED

TAIMOOR KHAN

as verification of the required standard for partial fulfilment of the degree
Bachelor of Science in Computer Science.

Approved by:

Supervisor:

Dr. Ghulam Mustafa

(Signature)

June 20, 2023

DECLARATION

We hereby declare that this project report is based on our original work except for citations and quotations which have been duly acknowledged. We also declare that it has not been previously and concurrently submitted for any other degree or award at Bahria University or other institutions.

Enrolment	Name	Signature
03-134192-044	KUNWAR MUZZAMMIL WAHEED	
03-134192-071	TAIMOOR KHAN	

Date: June 20, 2023

Specially dedicated to
my beloved grandmother, mother and father
(KUNWAR MUZZAMMIL WAHEED)
my beloved grandmother, mother and father
(TAIMOOR KHAN)

ACKNOWLEDGEMENTS

We would like to thank everyone who had contributed to the successful completion of this project. We would like to express our gratitude to our research supervisor, Dr Ghulam Mustafa for his invaluable advice, guidance, and his enormous patience throughout the development of the research.

In addition, we would also like to express our gratitude to our loving parent and friends who had helped and given us encouragement.

**KUNWAR MUZZAMMIL WAHEED
TAIMOOR KHAN**

Visualization of Health Data Through a Virtual Avatar

ABSTRACT

The proposed is a mobile application that revolutionizes the visualization of health data using Virtual Avatars. This innovative application will allow users to interactively visualize their health data in real time and witness their visual representation on their smartphones. By creating a digital representation of the user, linked to their device, the application will provide an immersive and engaging experience.

With the rise of wearable devices and smartphones, health data collection has become more accessible than ever. Traditional healthcare applications often overwhelm users with numerical data, leading to confusion and frustration.

In contrast, our application leverages the power of virtual avatars to present health data in an interactive and engaging manner. By creating a visual representation of the user's health, the application simplifies complex measurements and provides a more intuitive understanding of one's well-being.

The project follows the Feature-Driven Development (FDD) methodology and utilizes Flutter as the Software Development Kit (SDK). FDD provides a structured approach to software development, ensuring efficient collaboration and timely delivery of features. Flutter, with its cross-platform capabilities, enables us to develop the application for both Android and iOS platforms in future, reaching a wider audience.

Through this project, we address the growing need for user-friendly healthcare data visualization. By combining technology, visual representation, and real-time data, our application empowers users to take control of their health in an engaging and accessible way. We envision a future where individuals can easily interpret and act upon their health data, leading to improved well-being and informed decision-making.

TABLE OF CONTENTS

DECLARATION	ii
ACKNOWLEDGEMENTS	iv
ABSTRACT	v
TABLE OF CONTENTS	vi
LIST OF TABLES	ix
LIST OF FIGURES	x
LIST OF SYMBOLS / ABBREVIATIONS	xi

CHAPTERS

1	INTRODUCTION	1
	1.1 Background	1
	1.2 Problem Statement	1
	1.3 Aims and Objectives	2
	1.4 Scope of Project	2
2	SOFTWARE REQUIREMENTS SPECIFICATIONS	4
	2.1 Overall Description	4
	2.2 User Classes and Characteristics	4
	2.3 Operating Environment	5
	2.4 Development Environment	5
	2.5 Assumptions and Dependencies	5
	2.6 Other Non-Functional Requirements	6
	2.6.1 Performance Requirements	6
	2.6.2 Security Requirements	6

2.7	Software Quality Attributes	6
2.7.1	Maintainability	6
2.7.2	Reusability	7
2.8	Software Requirements Chart	7
3	DESIGN AND METHODOLOGY	8
3.1	Diagrams	8
3.1.1	Use case Diagram	9
3.1.2	Sequence Diagram	14
3.1.3	Entity Relationship Diagram	16
3.1.4	Activity Diagram	17
3.1.5	Class Diagram	18
3.1.6	Collaboration Diagram	19
4	IMPLEMENTATION	20
4.1	Avatar	20
4.2	Tools and Technologies	20
4.2.1	Flutter	20
4.2.2	Android Studio	21
4.2.3	Blender	21
5	RESULTS AND DISCUSSIONS (or USER MANUAL)	22
5.1	Splash Screen	22
5.2	Sign Up Page	23
5.3	Log In Page	24
5.4	Avatar	25
5.4.1	Idle Position	26
5.4.2	Wiping Sweat	27
5.4.3	Walking	28
5.4.4	Sit Down	29
5.4.5	Stand Up	30
5.5	Health Profile	31

6	CONCLUSION AND RECOMMENDATIONS	32
6.1	Conclusion	32
6.2	Limitations	32
6.3	Recommendations & Future Use	33
	REFERENCES	34

LIST OF TABLES

TABLE	TITLE	PAGE
Table 2.1:	Development Environment	5
Table 2.2:	Software Requirements Chart	7
Table 3.1:	Create Profile-U1	10
Table 3.2:	Sign In-U1	11
Table 3.3:	Homepage-U3	12
Table 3.4:	Avatar-U4	13

LIST OF FIGURES

FIGURE	TITLE	PAGE
Figure 1:	System Use Case	9
Figure 2:	Sequence Diagram Sign up	14
Figure 3:	Sequence Diagram Log in	15
Figure 4:	ERD16	
Figure 5:	Activity Diagram	17
Figure 6:	Class Diagram	18
Figure 7:	Collaboration Diagram	19
Figure 8:	Splash screen displaying the logo and branding elements.	22
Figure 9:	Featuring input fields for user registration and account creation.	23
Figure 10:	Login page with username and password fields	24
Figure 11:	Animated avatar in a relaxed and idle position	25
Figure 12:	Animated avatar in a relaxed and idle position	26
Figure 13:	Animated avatar in a sweating position	27
Figure 14:	Animated avatar in a walking animation.	28
Figure 15:	Animated avatar in a sitting down animation	29
Figure 16:	Animated avatar in a standing up animation	30
Figure 17:	User Health Profile	31

LIST OF SYMBOLS / ABBREVIATIONS

AFL	Flutter
FB	Firebase
DT	Dart
SRS	Software Requirement Specification
ERD	Entity Relationship Diagram
AR	Architectural Diagram
UML	Unified Modelling Language
DM	Domain Model
SD	Sequence Diagram
FDD	Feature-Driven Development
APK	Android Application Package
iOS	iPhone Operating Object
API	Application Program Interface
DHA	Digital Health Avatar
UI	User Interface

CHAPTER 1

INTRODUCTION

1.1 Background

Proposed mobile application provides a unique and innovative way to visualize health data. Users are able to track their health in real time through a virtual avatar on their mobile devices. The avatar indicates when the user is getting tired, prompting them to take a break and rest. Additionally, the avatar dynamically responds to changes in the user's health, providing a clear and easy-to-understand visualization of their health data. With the Health Avatar app, users can easily monitor their health and take proactive steps towards maintaining their well-being.

1.2 Problem Statement

The problem statement of our project is that existing health apps often fail to help users understand their health data due to their overwhelming interface and presentation of numerical data that can be difficult for users to understand. This results in users often deleting health apps approximately within 8 days of installation. The main challenge is to create an intelligent and interactive health app that visualizes health data in a more user-friendly manner.

To address this challenge, the proposed solution is to create an app called Health Avatar that uses interactive animations and a smart, friendly avatar to help

users understand their health data. The avatar will be used to help users understand their health conditions in a more intuitive way and the app will show user if their health condition is not healthy based on the collected data.

Overall, the aim is to create a health app that is more interactive, intuitive, and engaging, and that can help users better understand their health data.

1.3 Aims and Objectives

The objectives of the thesis are shown as following:

- i) To visualize the health data with the help of a virtual avatar.
- ii) To implement a mobile-based application to show health data in an interactive format.
- iii) To integrate the application with real-time health data.

1.4 Scope of Project

Proposed project solves the problem of scattered health and wellness data by creating a virtual avatar that visualizes all healthcare data available to the user. This avatar acts as a digital representation of the user.

It also improves the user's understanding of their health data. The visualizations displayed by the avatar are designed to be user-friendly, making it easier for the user to understand their health information. By doing so, the user is more informed about their health status and take necessary steps to maintain good health.

The data used by your application comes from wearables and smartphones. This means that users can use the app with devices they already own and do not need to purchase additional hardware.

Overall, the project's scope is to provide an easy-to-use, intuitive interface that allows users to track their health and wellness data. The application's virtual avatar serves as an interactive and friendly companion that keeps the user informed about their health status, making it easier for them to maintain good health.

CHAPTER 2

SOFTWARE REQUIREMENTS SPECIFICATIONS

2.1 Overall Description

The proposed project is a Mobile based application therefore front-end is developed in Android Studio. The final application is an Android-based application. Our development approach is feature-driven development in which first we built an overall model then built a feature list and then we planned, designed, and developed by feature.

2.2 User Classes and Characteristics

The application acts as your personal data manager thus features only a single user i.e., the person who installs the app for their own use.

User

- The user can perform the following activities.
- Sign-in
- View avatar

2.3 Operating Environment

The required environment for the Digital Health Avatar is as follows.

- Android Smartphone Phone
- Android 10 or above
- Stable Internet Connection

2.4 Development Environment

Table 2.1: Development Environment

Name	Description
Operating system Environment	Windows
Support Device	Android-based Smart Phone
Language	Dart
Framework	Flutter
Tools and Technology	<ul style="list-style-type: none"> • Nvidia GPU with at least 8GB • RAM 16 GB • SSD 256 GB • Android Studio • VS code
Database	Fire Base

2.5 Assumptions and Dependencies

Following are the assumptions and dependencies:

- i. Assume the phone is at least Android 10 or better.
- ii. Assume the user grants permission to the app.
- iii. Assume the user's phone is Android based.

2.6 Other Non-Functional Requirements

The non-functional requirements of the proposed project are given below:

2.6.1 Performance Requirements

The user requirement is for the Android mobile application to have a smaller APK size so that it can work on a wider range of mobile devices, including those with lower storage capacity.

2.6.2 Security Requirements

To protect your health data, the login system is secure Software Quality Attributes. The system includes basic database security features such as login and password authentication.

2.7 Software Quality Attributes

Software quality attributes are as following:

2.7.1 Maintainability

The developer maintains the mobile application data and all the services.

2.7.2 Reusability

Provided mobile application has reusability functions.

2.8 Software Requirements Chart

Table 2.2: Software Requirements Chart

ID	Priority	Type	Description
DHC - R1	High	Functional	Login
DHC - R2	High	Functional	Register new user
DHC - R3	High	Functional	Read Heartbeat
DHC - R4	High	Functional	Count Steps
DHC - R5	Medium	Functional	Display Data
DHC - R6	High	Functional	Stress
DHC - R7	High	Functional	Avatar to visualize above Functionalities

CHAPTER 3

DESIGN AND METHODOLOGY

3.1 Diagrams

This chapter provides an overview of the Digital Health Avatar design. The overall view of the system is provided by the system architectural design. Developers and clients are able to examine and check the design plan of the project. This Chapter includes the following objects.

- Use case diagrams.
- Sequence Diagrams.
- Entity relationship diagram
- Activity Diagram
- Class Diagram
- Domain Diagram

3.1.1 Use case Diagram

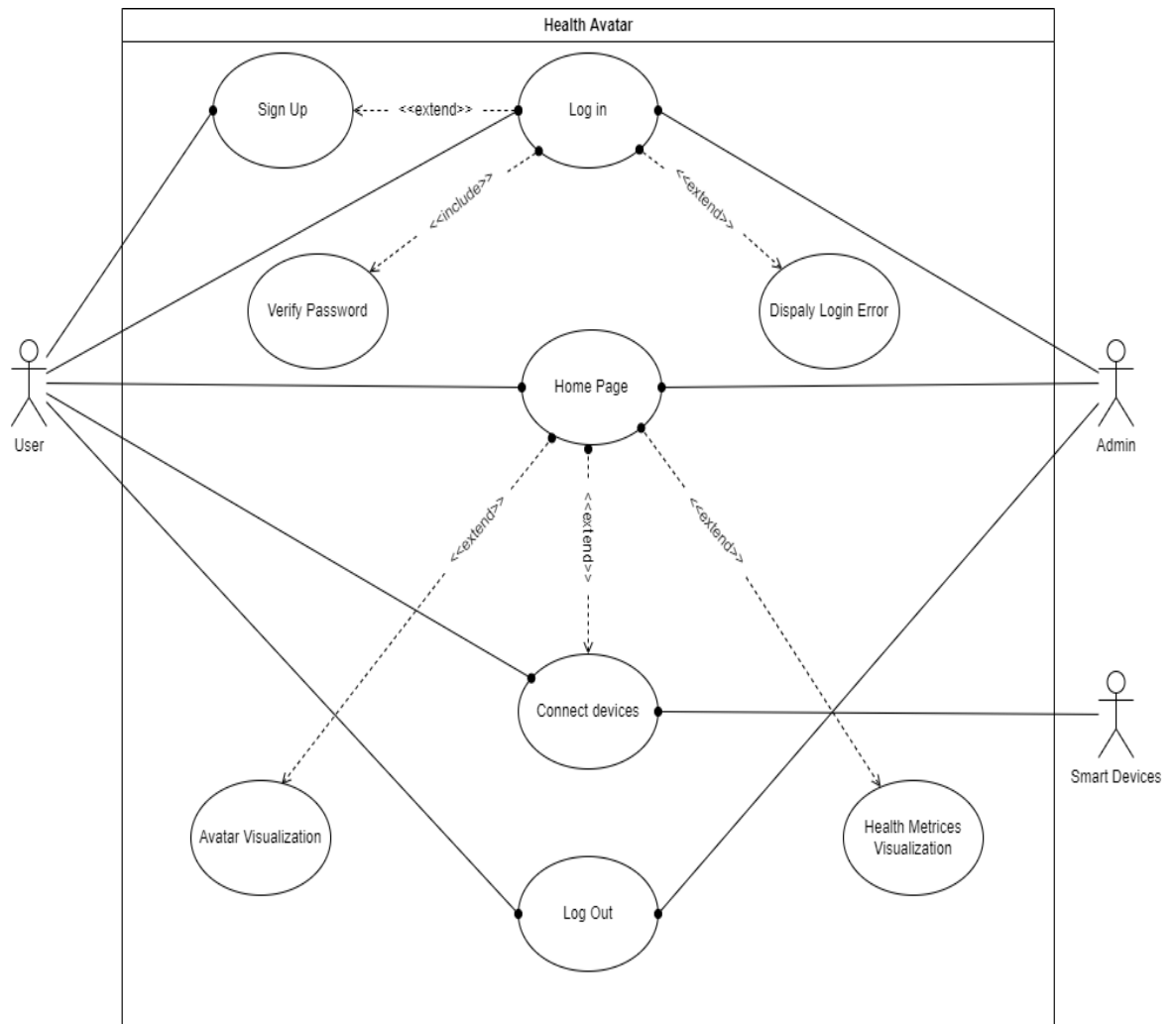


Figure 3.1: System Use Case

Fig 1 shows a high-level view of a system's functionality from a user's perspective, identifying the actors, their interactions with the system, and the use cases that meet their needs.

3.1.1.1 User Create Profile (U1)

Table 3.1: Create Profile-U1

	Name	Create Profile
1.	Use Case ID	U1
2.	Objective	Users will always log in with their username and password
3.	Priority	High
4.	Initiating Actor	Will be User
5.	Goal	To Create a new profile
6.	Pre-Conditions	User is viewing the login screen of the system
7.	Post Conditions	System displays the home screen
8.	Flow of Events	<ol style="list-style-type: none"> 1. User enter the personal information and press submit button 2. The system checks to see whether a user with the same ID already exists. 3. The system generates a new page containing the user's data and shows it as the user's home page
8.1.	Basic Flow	After success creates profile user go to login page U3
9.	Flow of Events for Extension (Alternate Scenario)	No alternative flow must sign up to proceed further
10.	Use Case	No other use case use

3.1.1.2 User Sign-In (U2)

Table 3.2: Sign In-U1

	Name	Sign In
1.	Use Case ID	U2
2.	Objective	The user will sign in with the credentials
3.	Priority	High
4.	Source	Data Base
5.	Actors	User
6.	Flow of Events	<ol style="list-style-type: none"> 1. Open Application 2. Enter sign in 3. Enter Username and Password 4. Click on Sign-in Button
6.1.	Basic Flow	After successful sign in user will go to U3
6.2.	Alternate Flow	No alternate flow, user must sign in to proceed further
6.3.	Exception Flow	Invalid Username Invalid Password
7.	Includes	U1
8.	Preconditions	Must sign up
9.	Postconditions	Taken to Home page
10.	Notes/Issues	If the User will sign in with the right credentials no problem will occur

3.1.1.3 Homepage (U3)

Table 3.3: Homepage-U3

	Name	Homepage
1.	Use Case ID	U3
2.	Objective	In this use case, user can view the main area of our app and access features from there
3.	Priority	High
4.	Source	User
5.	Actors	User
6.	Flow of Events	Sign in to and view application home screen with all the available features
6.1.	Basic Flow	After successful sign in Administrator can go to any of succeeding flows
6.2.	Alternate Flow	No alternate flow
6.3.	Exception Flow	No exception flow
7.	Includes	No other use case includes
8.	Preconditions	User must be signed in to perform U3
9.	Postconditions	Health status can be checked
10.	Notes/Issues	Provide Accurate data

3.1.1.4 Avatar (U4)

Table 3.4: Avatar-U4

	Name	Avatar
1.	Use Case ID	U4
2.	Objective	Show condition of user by actions
3.	Priority	High
4.	Source	Database
5.	Actors	Animated avatar
6.	Flow of Events	Login
6.1.	Basic Flow	None
6.2.	Alternate Flow	No alternate flow
6.3.	Exception Flow	Accurate data should be entered
7.	Includes	None
8.	Preconditions	None
9.	Postconditions	No postcondition
10.	Notes/Issues	None

3.1.2 Sequence Diagram

3.1.2.1 Sign Up

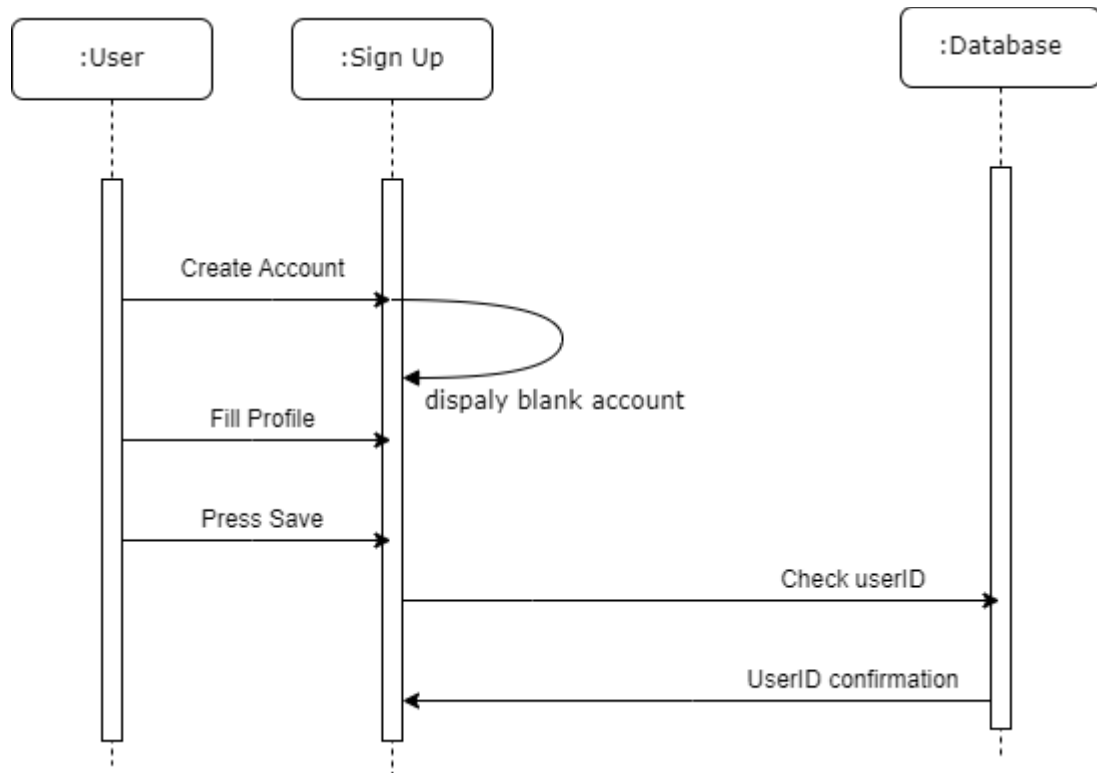


Figure 3.2: Sequence Diagram Sign up

Fig 2 shows an illustration of the flow of interactions between the user and the system during the sign-up process. It highlights the steps involved, such as inputting user details, validating the information, and creating a new account.

3.1.2.2 Log In

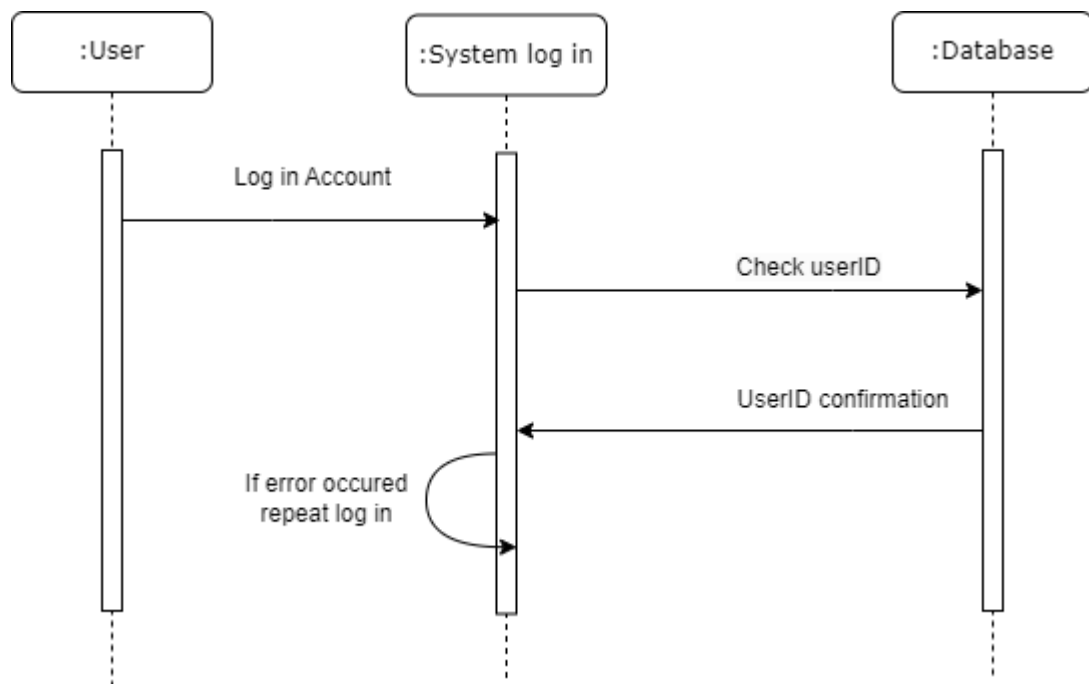


Figure 3.3: Sequence Diagram Log in

Fig 3 shows a capture of the sequence of interactions between the user and the system while logging in. It visualizes the steps involved, including providing login credentials, verifying the information, and granting access to the user account.

3.1.3 Entity Relationship Diagram

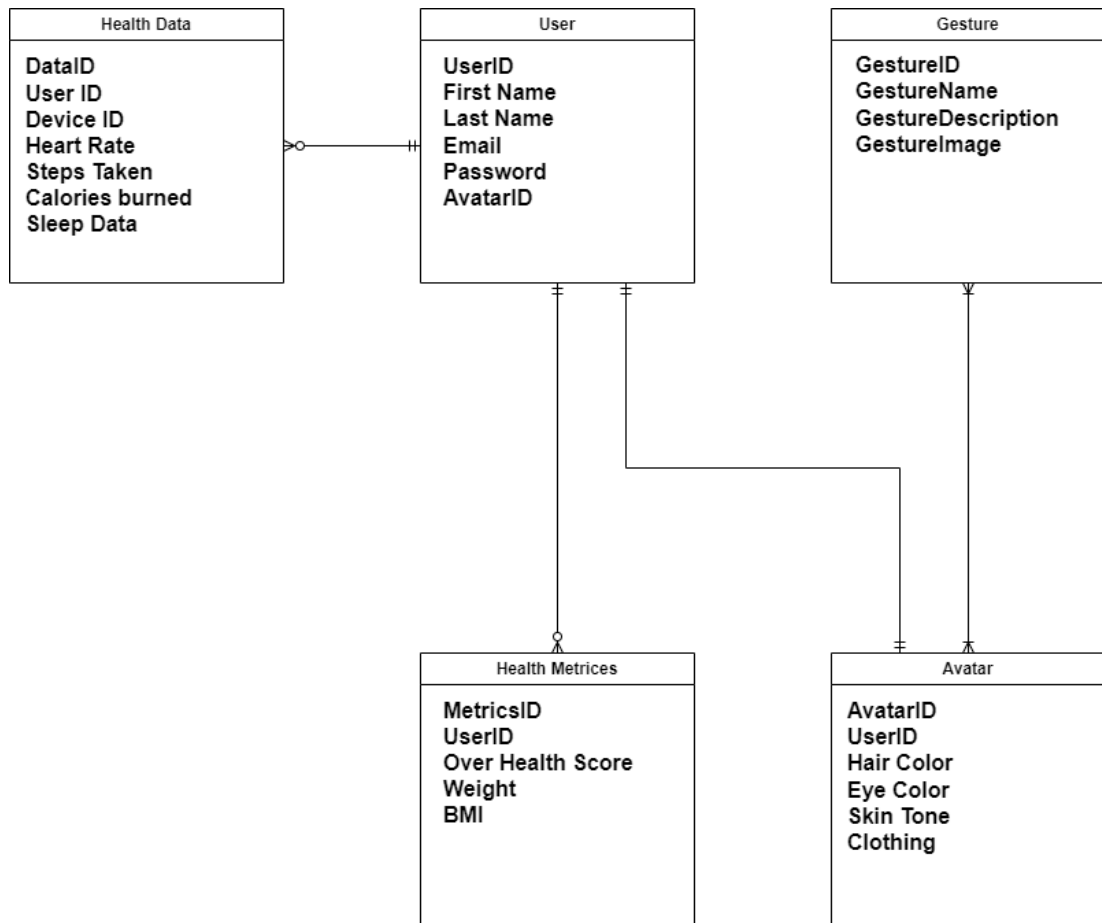


Figure 3.4: ERD

Fig 4 tells a visual representation of the entities, attributes, relationships, cardinality, and constraints in a system or domain's data model.

3.1.4 Activity Diagram

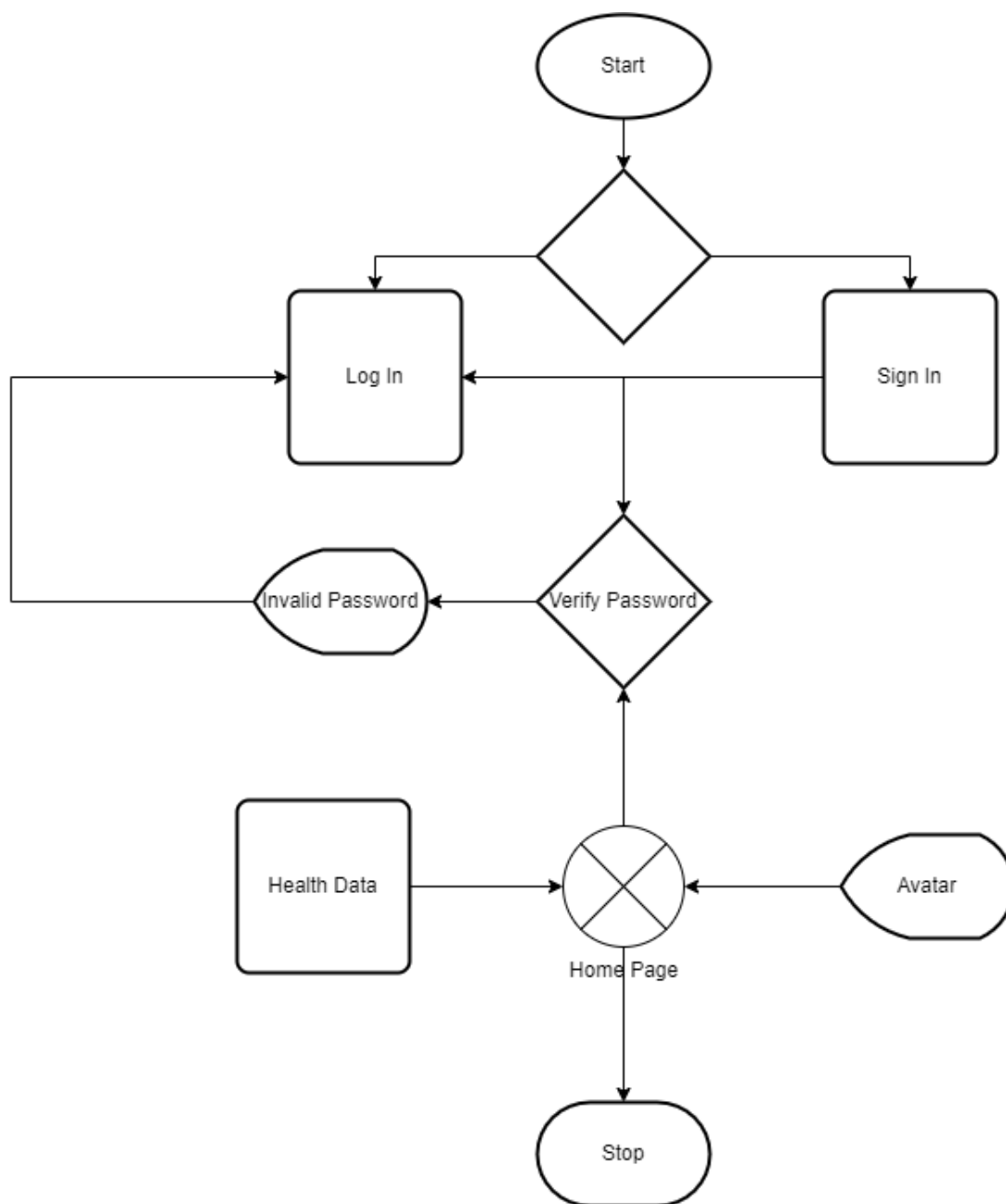


Figure 3.5: Activity Diagram

Fig 5 shows us the illustrates the flow of activities, control flow, decision points, synchronization, and actors involved in a system or organizational workflow or process flow.

3.1.5 Class Diagram

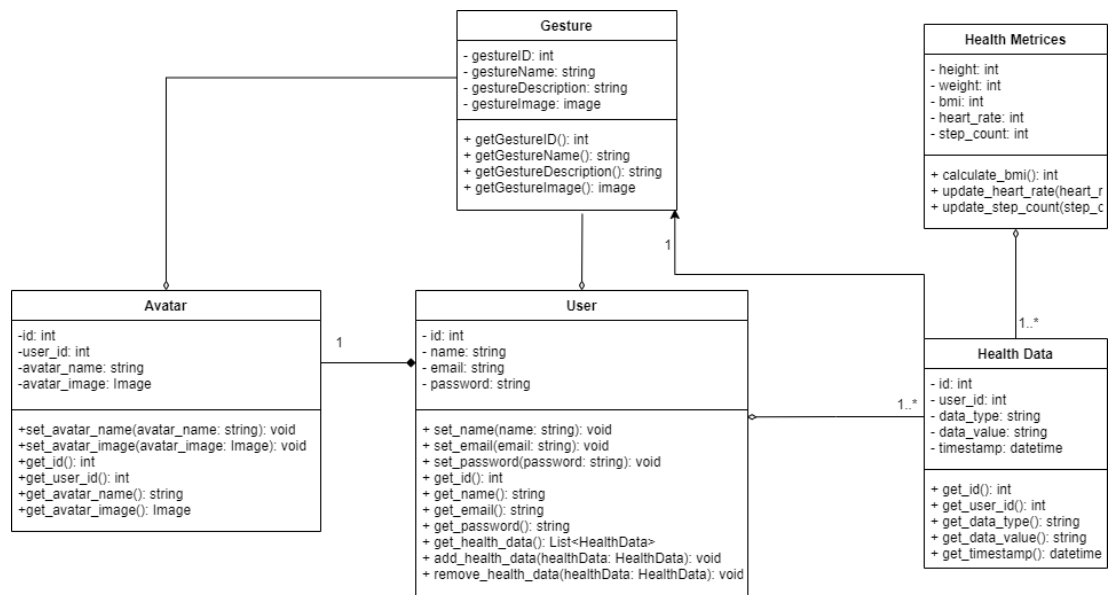


Figure 3.6: Class Diagram

Fig 6 shows class diagram of a project provides an overview of the classes, their attributes, methods, and inheritance with each other. It describes the structure of the system and helps in understanding the behavior of the system.

3.1.6 Collaboration Diagram

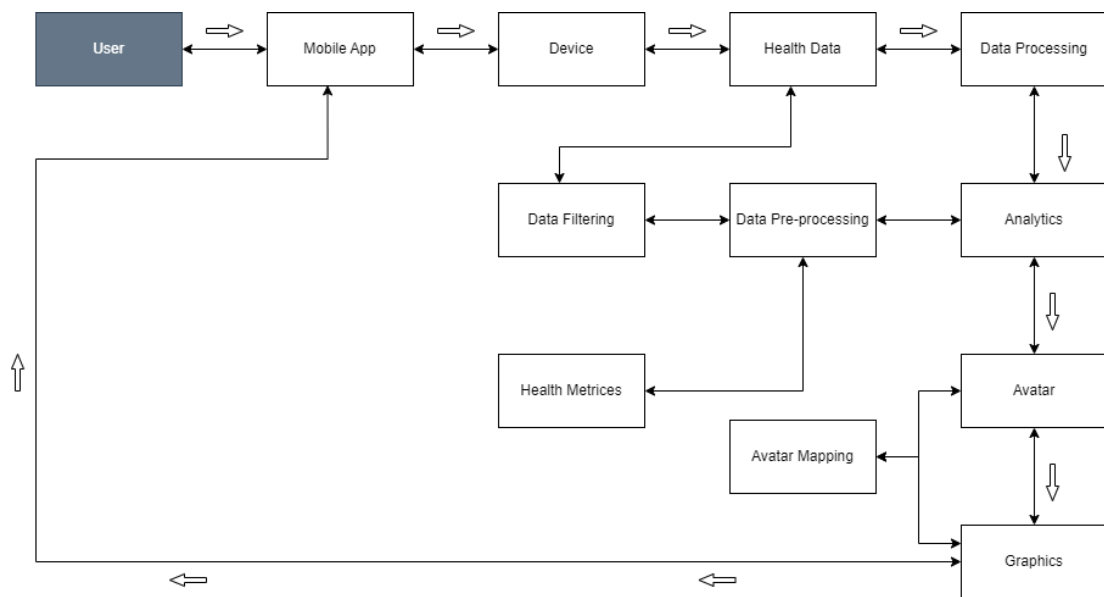


Figure 3.7: Collaboration Diagram

Fig 7 shows illustration of the interactions and relationships between various objects or components in a system. It visualizes the communication and collaboration between different entities, showcasing the flow of information and functionality within the system.

CHAPTER 4

IMPLEMENTATION

4.1 Avatar

We have used animated Avatars which makes the application more user-friendly and easier to understand.

4.2 Tools and Technologies

The tool and technologies used in this application are under following:

4.2.1 Flutter

Flutter is an open-source software created by Google to develop beautiful and fast-moving UIs. It is used to create cross platform apps from a single code base for Android, iOS, Linux, macOS, Windows, and web. It utilizes Dart as its base language to write code. Flutter allows you to create fast and Dynamic UIs.

4.2.2 Android Studio

Android Studio is a unified development environment for creating Android apps. Flutter is used for the front end, while Dart is used for the back end, which includes the train model, camera sensors, avatars, and outcomes.

4.2.3 Blender

Blender is a free and open-source 3D modeling and animation program that is used to create animations, visual effects, video games, and more. It provides several capabilities, including 3D modeling, texturing, lighting, animation, simulation, rendering, compositing, and video editing. It is accessible for Windows, Mac, and Linux operating systems, and it has a big and supportive user and development community.

CHAPTER 5

RESULTS AND DISCUSSIONS (or USER MANUAL)

5.1 Splash Screen

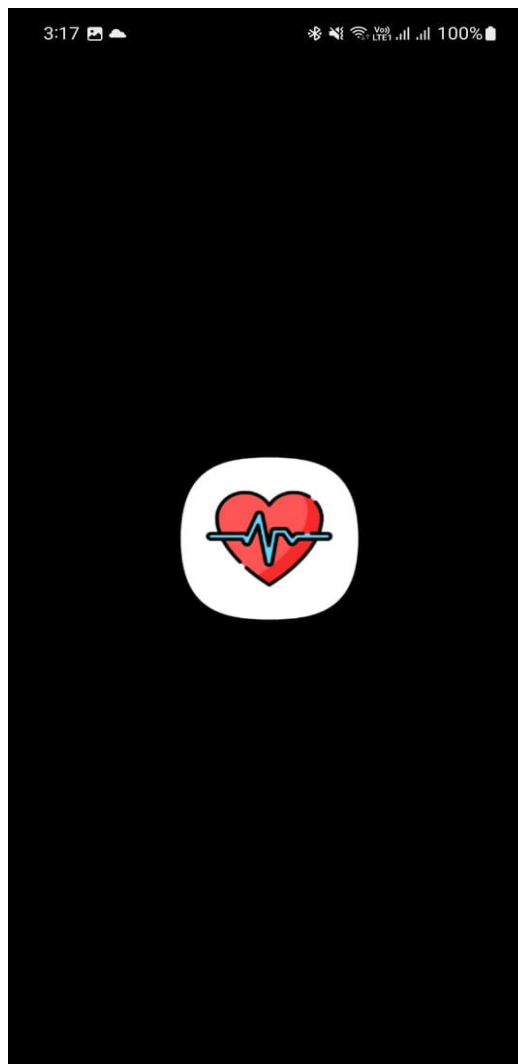


Figure 5.1: Splash screen displaying the logo and branding elements.

5.2 Sign Up Page

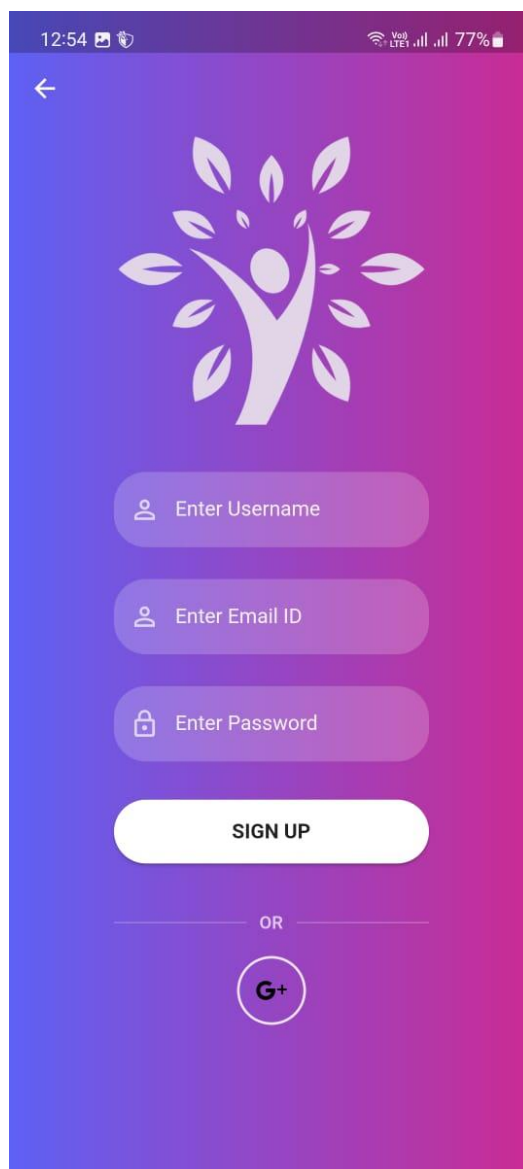


Figure 5.2: Featuring input fields for user registration and account creation.

The sign-up screen allows you to establish a new account by entering the required information. To begin, input your chosen username, which should be distinct and distinctive to represent your profile. After that, provide a valid email address that will be used for correspondence and account verification. Finally, to safeguard the security of your account, create a strong password. Once you've completed all the needed forms, click the "Sign Up" button to finish the process and successfully establish your account.

5.3 Log In Page

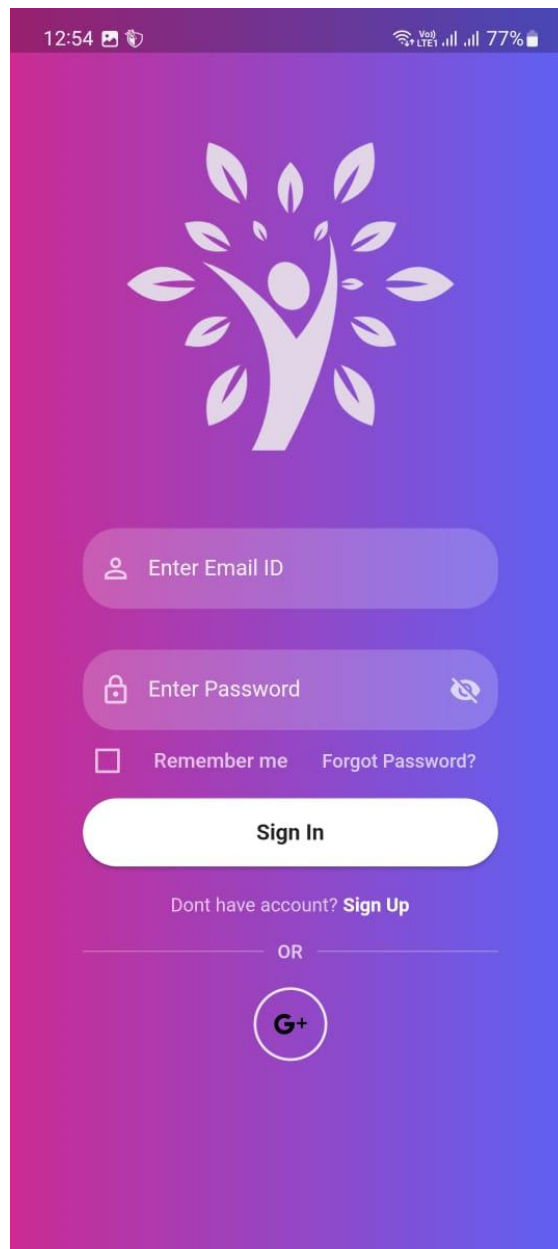


Figure 5.3: Login page with username and password fields

The login screen allows you to access your existing account. Enter your username, which should be the same as the one you used at registration, to log in. Then enter your password while assuring its accuracy and security. If you do not yet have an account, you can sign up by clicking on the "Don't have an account? Sign Up" option at the bottom of the page. After entering your login information, click the "Log In" button to proceed and access your account.

5.4 Avatar

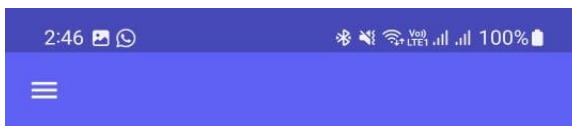


Figure 5.4: Animated avatar in a relaxed and idle position

Avatar depicting the user's virtual health will appear on the screen. The avatar will seem inactive, suggesting that the user's real-time health data is within typical limits. The avatar will move in a pleasant and peaceful manner, reflecting the user's general well-being. This visual depiction is a reassuring signal that the user's health is steady and that no immediate action or attention is necessary.

5.4.1 Idle Position



Figure 5.5: Animated avatar in a relaxed and idle position

Avatar depicting that the user's real-time health data is within typical limits. The avatar is in a pleasant and peaceful manner, reflecting the user's general well-being. This visual depiction is a reassuring signal that the user's health is steady and that no immediate action or attention is necessary.

5.4.2 Wiping Sweat



Figure 5.6: Animated avatar in a sweating position

The avatar is sweating, suggesting that the user's real-time health data has recognised a condition that causes excessive perspiration. The avatar's motions will become livelier, indicating exertion or discomfort. It urges the user to take relevant steps, such as monitoring vital signs or obtaining medical assistance as needed.

5.4.3 Walking

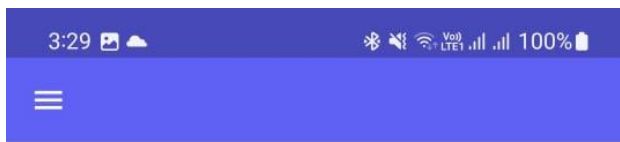


Figure 5.7: Animated avatar in a walking animation.

The avatar is depicted walking, showing that the user's real-time health data shows that steps are increasing. The purpose of this graphic depiction is to encourage the user to keep an active lifestyle and to emphasise the need of regular physical activity for general health and well-being.

5.4.4 Sit Down

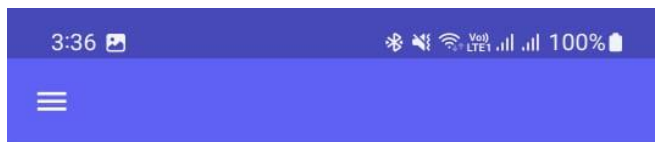


Figure 5.8: Animated avatar in a sitting down animation

The avatar will be portrayed sitting down, suggesting that it is exhausted. This image represents the user's current health statistics, implying that the user has strained oneself or participated in strenuous physical activity. The fatigued avatar acts as a visual reminder to the user to relax, take a break, and prioritise self-care.

5.4.5 Stand Up



Figure 5.9: Animated avatar in a standing up animation

The avatar will be seen standing up this time, reflecting an active and energetic state. The programme encourages the user to be active, exercise, and live a healthy lifestyle by presenting the avatar standing up.

5.5 Health Profile

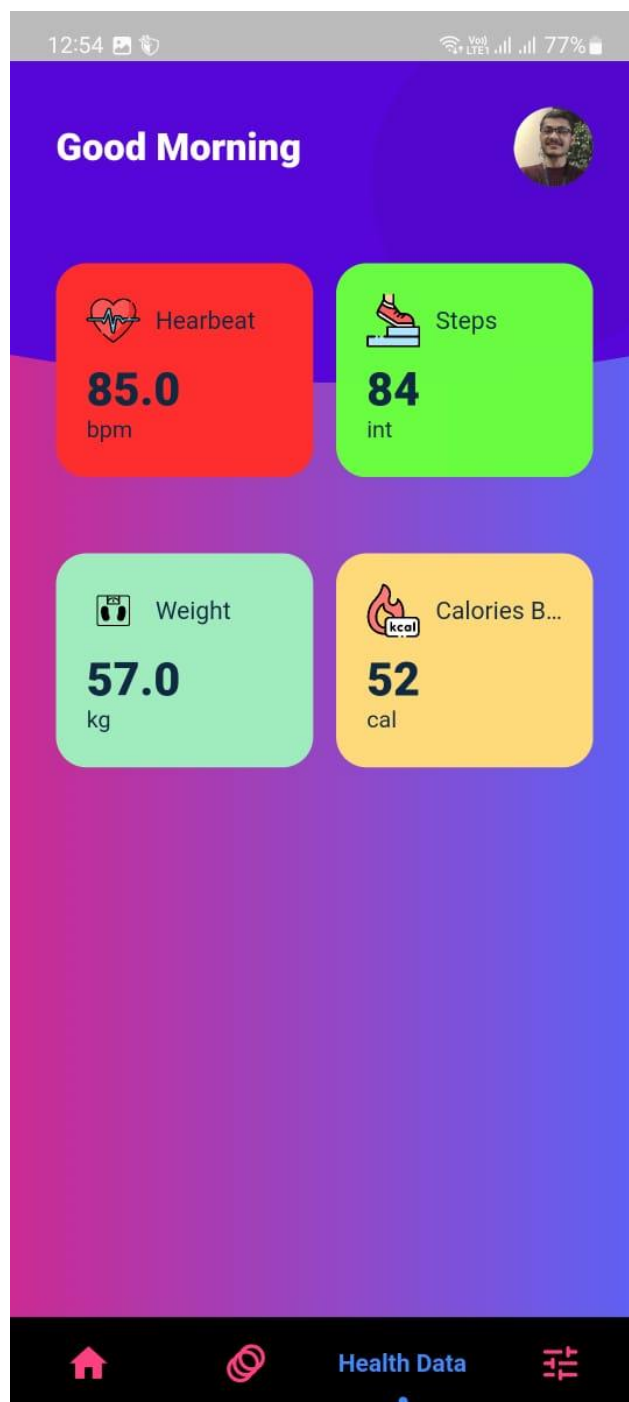


Figure 5.10: User Health Profile

The real-time health data panel displays a complete summary of the user's important health parameters. It shows vital information including heart rate, step count, weight, and calories burnt. Users may check their cardiovascular health, measure their physical activity, manage their weight goals, and keep track of their energy.

CHAPTER 6

CONCLUSION AND RECOMMENDATIONS

6.1 Conclusion

This pandemic has been an eye-opener for everyone. Now more than ever it has become crucial to maintain a healthy immune system so that your body can be prepared for the worst. Digital health avatar application which allows the user to better understand their health data by creating dynamic visualizations of that data. This application helps the user in regulating and maintaining a healthy lifestyle. The Application features dynamic graphics and avatars to help users get a better understanding of their health data.

6.2 Limitations

This app can only be used on Android smartphones as it was developed on Android Studio and this tool are not light weighted and sometimes on low-configuration devices start crashing and mostly then when you are using the app multiple times at the same time. 3D Model Viewers don't give us many options to control this animation, limited customization, limited GLTF, and OBJ file formats.

6.3 Recommendations & Future Use

This application covers a drop in the ocean that is the potential a mobile phone possesses in the field of medicine. Nowadays the technology has been pushed back and limitation barriers on everything is available on the palm of your hand. The features we are currently utilizing can be improved upon and added upon to create a much more functional and dynamic application. In future, we can add multiple modules and integrate many features that we can integrate soon. We will also work towards improving the animation and graphics system to create a much more interactive and customizable UI.

REFERENCES

- [1] 'Futureshaper: Shikha's journey to the "patient twin"'. <https://www.siemens-healthineers.com/perspectives/futureshaper-patient-twin> (accessed Jan. 03, 2023).
- [2] 'FitBit Fitness Tracker Data'. <https://www.kaggle.com/datasets/arashnic/fitbit> (accessed Jan. 03, 2023).
- [3] 'Health Platform API', Android Developers. <https://developer.android.com/training/wearables/healthservices/health-platform> (accessed Jan. 03, 2023).
- [4] L. Team, '7 Things You Need to Know About Feature Driven Development', Lvivity, Jun. 12, 2020. <https://lvivity.com/7-things-about-feature-driven-development> (accessed Jan. 04, 2023)
- [5] "Meet your virtual avatar. the future of personalized healthcare." <https://news.itu.intimeet-your-virtual-avatar-the-future-of-personalized-healthcare/> (accessed Oct. 16, 2022).
- [6] "The virtual human: digital avatars that are advancing healthcare - Atos." <https://atos.net/en/blog/the-virtual-human-digital-avatars-that-are-advancing-healthcare> (accessed Oct. 16, 2022).
- [7] "ISO - One step ahead with mobile apps." https://www.iso.org/news/isofocus_141-6.html (accessed Sep. 09, 2022).
- [8] "ISO - ISO/TS 82304-2:2021 - Health software - Part 2: Health and wellness Apps - Quality and reliability." <https://www.iso.org/standard/78182.html> (Accessed Sep. 09.2022).
- [9] E. Parimbelli et al., "A review of AI and Data Science support for cancer management," *Artif Intell. Med*, vol. 117, Jul. 2021. doi: 10.1016/J.ARTMED.2021A02111. (Accessed Sep. 09.2022).

APPENDIX A: Code

Main.dart

```
import 'package:firebase_auth/firebase_auth.dart';
import 'package:firebase_core/firebase_core.dart';
import 'package:flutter/material.dart';
import 'package:fyp/screens/home_screen.dart';
import 'package:fyp/screens/LogIn_Screen.dart';

void main() async{
  WidgetsFlutterBinding.ensureInitialized();
  await Firebase.initializeApp();
  runApp(const MyApp());
}

class MyApp extends StatefulWidget {
  const MyApp({super.key});

  @override
  State<MyApp> createState() => _MyAppState();
}

class _MyAppState extends State<MyApp> {
  User? user;

  @override
  void initState() {
    super.initState();
    user = FirebaseAuth.instance.currentUser;
  }
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      debugShowCheckedModeBanner: false,
      home: user != null ? const HomeScreen() : const SignInScreen(),
    );
  }
}
```


Animation Control:

```

import 'package:firebase_auth/firebase_auth.dart';
import 'package:flutter/material.dart';
import 'package:fyp/screens/GoogleFit_API.dart';
import 'package:fyp/screens/LogIn_Screen.dart';
import 'dart:async';
import 'package:model_viewer_plus/model_viewer_plus.dart';
import 'package:health/health.dart';

class HomeScreen extends StatefulWidget {
  const HomeScreen({super.key});

  @override
  State<HomeScreen> createState() => _HomeScreenState();
}

enum AppState {
  DATA_NOT_FETCHED,
  NO_DATA,
  STEPS_READY,
}

class _HomeScreenState extends State<HomeScreen> {
  int _count = 0;
  String _currentAnimation = "";
  final user = FirebaseAuth.instance.currentUser!;
  List<HealthDataPoint> _healthDataList = [];
  AppState _state = AppState.DATA_NOT_FETCHED;
  int _nofSteps = 0;
  int tempsteps = 0;
  HealthFactory health = HealthFactory();

  Future fetchStepData() async {
    int? steps;
    final now = DateTime.now();
    final midnight = DateTime(now.year, now.month, now.day);
    bool requested = await
health.requestAuthorization([HealthDataType.STEPS]);
    if (requested) {
      try {
        steps = await health.getTotalStepsInInterval(midnight, now);
      } catch (error) {
        print("Caught exception in getTotalStepsInInterval: $error");
      }
      print("Total number of steps: $steps");
      setState() {
        _nofSteps = (steps == null) ? 0 : steps;
        _state = (steps == null) ? AppState.NO_DATA :
AppState.STEPS_READY;
      });
    }
  }
}

```

```

else {
    print("Authorization not granted - error in authorization");
    setState(() => _state = AppState.DATA_NOT_FETCHED);
}
}

@override
void initState() {
    super.initState();
    _currentAnimation = randomAnimation();
    Timer.periodic(const Duration(seconds: 10), (timer) {
        setState() {
            _currentAnimation = randomAnimation();
            _count += 1;
            key:
            ValueKey<int>(_count);
            print('Animation changed: $_currentAnimation');
        });
    });
}

int flag = 0;
int randomIndex = 0;
String randomAnimation() {
    fetchStepData();
    if (randomIndex >= 5) {
        randomIndex = 1;
    }
    randomIndex++;
    if (_nofSteps > 10000) {
        randomIndex = 4;
    }
    switch (randomIndex) {
        case 1:
            return 'StandToSit';
        case 2:
            return 'SitToStand';
        case 3:
            return 'BreathingIdle';
        case 4:
            return 'Sweating';
        case 5:
            return 'walking';
        default:
            return 'BreathingIdle';
    }
}
}

```

```

Widget build(BuildContext context) {
  return Scaffold(
    appBar: AppBar(
      backgroundColor: Color.fromRGBO(94, 97, 244, 1),
    ),
    drawer: Drawer(
      child: Container(
        color: Color.fromRGBO(94, 97, 244, 1),
        child: ListView(
          children: [
            DrawerHeader(
              child: Icon(
                Icons.home,
                size: 35,
              ),
            ),
            ListTile(
              leading: Icon(
                Icons.home,
                size: 30,
              ),
              title: Text(
                'Sign In',
                style: TextStyle(fontSize: 18),
              ),
              onTap: () {
                Navigator.of(context).push(
                  MaterialPageRoute(builder: (context) => SignInScreen()));
              },
            ),
            ListTile(
              leading: Icon(
                Icons.home,
                size: 30,
              ),
              title: Text(
                'Log out',
                style: TextStyle(fontSize: 18),
              ),
              onTap: () {
                FirebaseAuth.instance.signOut();
                Navigator.of(context).push(
                  MaterialPageRoute(builder: (context) => SignInScreen()));
              },
            ),
          ],
        ),
      ),
    ),
  );
}

```

```

ListTile(
  leading: Icon(
    Icons.home,
    size: 30,
  ),
  title: Text(
    'Step Count',
    style: TextStyle(fontSize: 18),
  ),
  onTap: () {
    Navigator.of(context).push(MaterialPageRoute(
      builder: (context) => HealthDataScreen()));
  },
),
],
),
),
),
body: Center(
  child: Container(
    color: Colors.white,
    child: AnimatedSwitcher(
      duration: const Duration(milliseconds: 2750),
      child: ModelViewer(
        key: ValueKey<int>(_count),
        autoPlay: true,
        src: 'assets/leanordAnimations.glb',
        animationName: _currentAnimation,
      ),
    ),
  ),
);
}
}

```

User Health Profile:

```

import 'package:flutter/material.dart';
import 'package:fyp/resuable_widgets/custom_clipper.dart';
import 'package:fyp/resuable_widgets/card_main.dart';
import 'package:fyp/utils/const.dart';
import 'package:health/health.dart';
import 'dart:async';

class dScreen extends StatefulWidget {
  const dScreen({super.key});

  @override
  State<dScreen> createState() => _dScreenState();
}

enum AppState {
  DATA_NOT_FETCHED,
  FETCHING_DATA,
  DATA_READY,
  NO_DATA,
  AUTH_NOT_GRANTED,
  STEPS_READY,
}

class _dScreenState extends State<dScreen> {
  List<HealthDataPoint> _healthDataList = [];
  AppState _state = AppState.DATA_NOT_FETCHED;
  int _nofSteps = 10;
  HealthFactory health = HealthFactory();

  Future fetchData() async {
    setState(() => _state = AppState.FETCHING_DATA);

    final types = [
      HealthDataType.STEPS,
      HealthDataType.WEIGHT,
      HealthDataType.HEIGHT,
      HealthDataType.ACTIVE_ENERGY_BURNED,
      HealthDataType.HEART_RATE,
    ];
    final now = DateTime.now();
    final yesterday = now.subtract(Duration(hours: 24));
    _healthDataList.clear();
    List<HealthDataPoint> healthData =
      await health.getHealthDataFromTypes(yesterday, now, types);
    _healthDataList.addAll(
      (healthData.length < 100) ? healthData : healthData.sublist(0, 100));
    // _healthDataList.forEach((x) => print(x));
    setState() {

```

```

        _state = _healthDataList.isEmpty ? AppState.NO_DATA :
AppState.DATA_READY;
    });
    //_content();
}

Future fetchStepData() async {
    int? steps;
    final now = DateTime.now();
    final midnight = DateTime(now.year, now.month, now.day);
    steps = await health.getTotalStepsInInterval(midnight, now);
    print('Total number of steps: $steps');
    setState(() {
        _nofSteps = (steps == null) ? 0 : steps;
        _state = (steps == null) ? AppState.NO_DATA :
AppState.STEPS_READY;
    });
}

@override
void initState() {
    super.initState();
    Timer.periodic(const Duration(seconds: 6), (timer) {
        setState(() {
            fetchData();
            fetchStepData();
            Timer(Duration(seconds: 3), () {
                print('1 here-----1');
                sortdata();
                print('2 here-----1');
                print(stps);
                print(hrtrate);
                print(cales);
                print(wgt);
                print('3 here-----1');
            });
        });
    });
}

List stps = [];
List hrtrate = [];
List cales = [];
List hgt = [];
List wgt = [];
String replacer = "";
double n2 = 0.0; //heartbeat
double n3 = 0.0; //calories
double nn3 = 0.0;
int calnum = 0;

```

```

int n4 = 0;
double n5 = 0; //weight
void sortdata() {
    stps.clear();
    hrtrate.clear();
    cales.clear();
    hgt.clear();
    wgt.clear();
    for (int i = 0; i < _healthDataList.length; i++) {
        if (_healthDataList[i].typeString == 'STEPS') {
            stps.add(_healthDataList[i].value);
        } else if (_healthDataList[i].typeString == 'HEART_RATE') {
            hrtrate.add(_healthDataList[i].value);
        } else if (_healthDataList[i].typeString ==
'ACTIVE_ENERGY_BURNED') {
            cales.add(_healthDataList[i].value);
        } else if (_healthDataList[i].typeString == 'HEIGHT') {
            hgt.add(_healthDataList[i].value);
        } else if (_healthDataList[i].typeString == 'WEIGHT') {
            wgt.add(_healthDataList[i].value);
        }
    }
}

n4 = _nofSteps;
replacer = hrtrate[0].toString();
n2 = double.parse(replacer);

replacer = cales[0].toString();
n3 = double.parse(replacer);

replacer = wgt[0].toString();
n5 = double.parse(replacer);

for (int i = 0; i < cales.length; i++) {
    replacer = cales[i].toString();
    nn3 = double.parse(replacer);
    n3 += nn3;
}
calnum = n3.toInt();
// String n1 = "";
// double n2 = 0.0;
// double n3 = 0.0; //calories
// double n5 = 0;
// n1 = hrtrate[0].toString();
// n5 = double.parse(n1);
// for (int i = 0; i < cales.length; i++) {
//     n1 = cales[i].toString();
//     n2 = double.parse(n1);
//     n3 += n2;
// }

```

```

    print(n2);
    print(calnum);
    print(n4);
    print(n5);
  }

  //heartrate
  // $wgt
  // $ _nofSteps

  // Widget _contentDataReady() {
  //   sortdata();
  //   // print(stps);
  //   // print(hrtrate);
  //   // print(cales);
  //   // print(wgt);
  //   return Text("");
  // }
  // Widget _stepsFetched() {
  //   return Text("Total number of steps: $ _nofSteps");
  // }
  // Widget _content() {
  //   if (_state == AppState.DATA_READY)
  //     return _contentDataReady();
  //   else if (_state == AppState.STEPS_READY)
  //     return _stepsFetched();
  //   else
  //     return Text('No data');
  // }

  @override
  Widget build(BuildContext context) {
    double statusBarHeight = MediaQuery.of(context).padding.top;
    return Scaffold(
      backgroundColor: Constants.backgroundColor,
      body: Stack(
        children: <Widget>[
          ClipPath(
            clipper: MyCustomClipper(clipType: ClipType.bottom),
            child: Container(
              color: Color.fromARGB(255, 87, 6, 217),
              height: Constants.headerHeight + statusBarHeight,
            ),
          ),
          Positioned(
            right: -45,
            top: -30,
            child: ClipOval(
              child: Container(
                color: Colors.black.withOpacity(0.05),

```



```

        height: 220,
        width: 220,
      ),
    ),
  ),

// BODY
Padding(
  padding: EdgeInsets.all(Constants.paddingSide),
  child: ListView(
    children: <Widget>[
      // Header - Greetings and Avatar
      Row(
        children: <Widget>[
          Expanded(
            child: Text(
              "Good Morning,\nMuzzammil",
              style: TextStyle(
                fontSize: 25,
                fontWeight: FontWeight.w900,
                color: Colors.white),
            ),
          ),
          CircleAvatar(
            radius: 26.0,
            backgroundImage: AssetImage(
              'assets/icons/muzzamildaku.jpeg'))
//assets\icons\muzzamildaku.jpeg
        ],
      ),

      SizedBox(height: 50),

      // Main Cards - Heartbeat and Blood Pressure
      Container(
        height: 140,
        child: ListView(
          scrollDirection: Axis.horizontal,
          children: <Widget>[
            CardMain(
              image: AssetImage('assets/icons/heartbeaticon.png'),
              title: "Hearbeat",
              value: '$n2',
              unit: "bpm",
              color: Color.fromARGB(255, 255, 46, 46)),
            CardMain(
              image: AssetImage(
                'assets/icons/step.png'), //assets\icons\step.png
              title: "Steps",
              value: "$n4",

```

```

        unit: "int",
        color: Color.fromRGBO(105, 255, 64, 0.992)),
    ],
  ),
),
SizedBox(
  height: 50, // <-- SEE HERE
),
Container(
  height: 140,
  child: ListView(
    scrollDirection: Axis.horizontal,
    children: <Widget>[
      CardMain(
        image: AssetImage('assets/icons/weight.png'),
        title: "Weight",
        value: "$n5",
        unit: "kg",
        color: Constants.lightGreen,
      ),
      CardMain(
        image: AssetImage('assets/icons/calories.png'),
        title: "Calories Burn",
        value: "$calnum",
        unit: "cal",
        color: Constants.lightYellow)
    ],
  ),
),
),
),
),
),
),
);
}
}

```

health avatar

ORIGINALITY REPORT

9%

SIMILARITY INDEX

2%

INTERNET SOURCES

1%

PUBLICATIONS

7%

STUDENT PAPERS

PRIMARY SOURCES

1	Submitted to Higher Education Commission Pakistan Student Paper	4%
2	Submitted to Midlands State University Student Paper	1%
3	Submitted to Dhofar University Student Paper	1%
4	www.scaler.com Internet Source	1%
5	Submitted to Edge Hill University Student Paper	<1%
6	Submitted to University of Hertfordshire Student Paper	<1%
7	businessdocbox.com Internet Source	<1%
8	theses.fi Internet Source	<1%
9	flutterawesome.com Internet Source	<1%

10

"Human-Computer Interaction – INTERACT 2011", Springer Science and Business Media LLC, 2011

Publication

<1 %

11

dspace.unza.zm

Internet Source

<1 %

12

Kyu C. Cho, Yeong-Tae Song. "Layered Design of CORBA Audio/Video Streaming Service in a Distributed Java ORB Middleware Multimedia Platform", Sixth International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing and First ACIS International Workshop on Self-Assembling Wireless Networks (SNPD/SAWN'05), 2005

Publication

<1 %

Exclude quotes Off

Exclude matches Off

Exclude bibliography On