VISVESVARAYA TECHNOLOGICAL UNIVERSITY

JNANA SANGAMA, BELAGAVI – 590 018

An Internship Project Report
on

Fitway

Submitted in partial fulfillment of the requirements for the VII Semester of degree of **Bachelor of Engineering in Information Science and Engineering** of Visvesvaraya Technological University, Belagavi

By

Sanjana Prakash

1RN18IS094

Under the Guidance of

Dr. Suresh L
Professor & Head
Department of ISE

ESTD:2001
An Institute with a Difference

Department of Information Science and Engineering
RNS Institute of Technology

Dr. Vishnuvaradhan Road, Rajarajeshwari Nagar post,
Channasandra, Bengaluru-560098

2021-2022

RNS INSTITUTE OF TECHNOLOGY

Dr. Vishnuvaradhan Road, Rajarajeshwari Nagar post,
Channasandra, Bengaluru - 560098

DEPARTMENT OF INFORMATION SCIENCE AND ENGINEERING

CERTIFICATE

Certified that the Internship work entitled ***Fitway*** has been successfully completed by **Sanjana Prakash (1RN18IS094)** bonafide student of **RNS Institute of Technology, Bengaluru** in partial fulfillment of the requirements of 7th semester for the award of degree in **Bachelor of Engineering in Information Science and Engineering** of **Visvesvaraya Technological University, Belagavi** during academic year **2021-2022.** The internship report has been approved as it satisfies the academic requirements in respect of internship work for the said degree.

Dr. Suresh L Internship Guide Associate Professor Department of ISE	**Dr. Suresh L** Professor and HoD Department of ISE RNSIT	**Dr. M K Venkatesha** Principal RNSIT

External Viva

Name of the Examiners	**Signature with Date**
1. _____	**1.** _____
2. _____	**2.** _____

DECLARATION

I, **Sanjana Prakash [USN: 1RN18IS094]** student of VII Semester BE, in Information Science and Engineering, RNS Institute of Technology hereby declare that the Internship work entitled ***Fitway*** has been carried out by us and submitted in partial fulfillment of the requirements for the *VII Semester degree of* ***Bachelor of Engineering in Information Science and Engineering*** *of Visvesvaraya Technological University, Belagavi* during academic year 2021-2022.

Place : Bengaluru

Date : 10/01/2021

Sanjana Prakash

(1RN18IS094)

ABSTRACT

Over recent years the world has seen a spike in the download and usage of fitness and health apps. In 2014 fitness app usage grew at a substantial rate, being up there as the most used category of application for that year. Since then it has maintained its user base and continues to grow, with the inclusion of wearables like google fit, Fitbit and HealthKit. This is the dawn of a new era, an era where people look more to their mobiles or their fitness watches to check on their health, rather than the traditional method of going and seeing a doctor. These apps provide a great avenue for those who are interested on tracking their fitness levels runners, cyclists, and gym goers alike. Everything can be tracked nowadays, even the standard iPhone comes with a health app built in, with a range of features.

There are several highly successful fitness apps on the market, some of these include mapmyrun, mapmyride, and the FitBit app. Both mapmyrun and mapmyride are made by the same group and are simple route trackers for running and cycling. Once your journey is finished the app then gives detailed statistics of the route and calories burned. The FitBit app is one that I use, it works in tandem with a wearable FitBit watch, which tracks your steps and monitors your heart rate.

ACKNOWLEDGMENT

At the very onset I would like to place our gratefulness to all those people who helped me in making the Internship a successful one.

Coming up, this internship to be a success was not easy. Apart from the sheer effort, the enlightenment of the very experienced teachers also plays a paramount role because it is they who guided me in the right direction.

First, I would like to thank the **Management of RNS Institute of Technology** for providing such a healthy environment for the successful completion of internship work.

In this regard, I express sincere gratitude to my beloved Principal **Dr. M K Venkatesha,** for providing us all the facilities.

I am extremely grateful to my own and beloved Professor and Head of Department of Information science and Engineering, **Dr. Suresh L**, for having accepted to patronize me in the right direction with all her wisdom.

I place my heartfelt thanks to **Dr. Suresh L** Professor and HOD, Department of Information Science and Engineering for having guided internship and all the staff members of the department of Information Science and Engineering for always helping.

I thank **Mr. Akshay D R, ENMAZ**, for providing the opportunity to be a part of the Internship program and having guided me to complete the same successfully.

I also thank our internship coordinator **Dr. R Rajkumar,** Associate Professor, Department of Information Science and Engineering. I would thank my friends for having supported me with all their strength and might. Last but not the least, I thank my parents for supporting and encouraging me throughout. I have made an honest effort in this assignment.

Sanjana Prakash
1RN18IS094

TABLE OF CONTENTS

LIST OF FIGURES

ABBREVIATION

UI	User Interface
APK	Android Application Packet
SDK	Software Development Kit
JSON	JavaScript Object Notation

Chapter 1

INTRODUCTION

1.1 Introduction to Flutter

Flutter is Google's Mobile SDK to build native iOS and Android, Desktop (Windows, Linux, macOS), Web apps from a single codebase. When building applications with Flutter everything towards Widgets – the blocks with which the flutter apps are built. They are structural elements that ship with a bunch of material design-specific functionalities and new widgets can be composed out of existing ones too. The process of composing widgets together is called composition. The User Interface of the app is composed of many simple widgets, each of them handling one particular job. That is the reason why Flutter developers tend to think of their flutter app as a tree of widgets.

1.1.1 History

Flutter launched as a project called Sky which at the beginning worked only on Android. Flutter's goal is enabling developers to compile for every platform using its own graphic layer rendered by the Skia engine. Here's a brief presentation of Flutter's relatively short history.

Flutter is a free and open-source mobile UI framework created by Google and released in May 2017. In a few words, this allows you to create a native mobile application with only one code. It means that you can use one programming language and one codebase to create two different apps (IOS and Android).

The first version of Flutter was known by the codename "Sky" and ran on the Android operating system. It was unveiled at the 2015 Dart developer summit[6] with the stated intent of being able to render consistently at 120 frames per second.[7] During the keynote of Google Developer Days in Shanghai in September 2018, Google announced Flutter Release Preview 2, which is the last big release before Flutter 1.0. On December 4th of that year, Flutter 1.0 was released at the Flutter Live event, denoting the first "stable" version of the Framework. On December 11, 2019, Flutter 1.12 was released at the Flutter Interactive event.[8]

On May 6, 2020, the Dart software development kit (SDK) in version 2.8 and the Flutter in version 1.17.0 were released, where support was added to the Metal API, improving performance on iOS devices (approximately 50%), new Material widgets, and new network tracking.

On March 3, 2021, Google released Flutter 2 during an online Flutter Engage event. This major update brought official support for web-based applications with new CanvasKit renderer and web specific widgets, early-access desktop application support for Windows, macOS, and Linux and improved Add-to-App APIs.[9] This release included sound null-safety, which caused many breaking changes and issues with many external packages, but the Flutter team included instructions to mitigate these changes as well.

On September 8th, 2021, the Dart SDK in version 2.14 and Flutter version 2.5 were released by Google. The update brought improvements to the Android Full-Screen mode and the latest version of Google's Material Design called Material You. Dart received two new updates, the newest lint conditions have been standardized and preset as the default conditions as well Dart for Apple Silicon is now stable.

1.1.2 Framework-Architecture

The major components of Flutter include:

- Dart platform
- Flutter engine
- Foundation library
- Design-specific widgets
- Flutter Development Tools (DevTools)

Dart platform

Flutter apps are written in the Dart language and make use of many of the language's more advanced features.

On Windows, macOS, and Linux[11] Flutter runs in the Dart virtual machine, which features a just-in-time execution engine. While writing and debugging an app, Flutter uses Just In Time compilation, allowing for "hot reload", with which modifications to source files can be injected into a running application. Flutter extends this with support for stateful hot reload, where in most cases changes to source code are reflected immediately in the running app without requiring a restart or any loss of state.

For better performance, release versions of Flutter apps targeting Android and iOS are compiled with ahead-of-time (AOT) compilation.

Flutter engine

Flutter's engine, written primarily in C++, provides low-level rendering support using Google's Skia graphics library. Additionally, it interfaces with platform-specific SDKs such as those provided by Android and iOS.[10] The Flutter Engine is a portable runtime for hosting Flutter applications. It implements Flutter's core libraries, including animation and graphics, file and network I/O, accessibility support, plugin architecture, and a Dart runtime and compile toolchain. Most developers interact with Flutter via the Flutter Framework, which provides a reactive framework and a set of platform, layout, and foundation widgets.

Foundation library

The Foundation library, written in Dart, provides basic classes and functions that are used to construct applications using Flutter, such as APIs to communicate with the engine.

Design-specific widgets

The Flutter framework contains two sets of widgets that conform to specific design languages: Material Design widgets implement Google's design language of the same name, and Cupertino widgets implement Apple's iOS Human interface guidelines.

Chapter 2

LITERATURE SURVEY

2.1 Fitway: A Fitness Application

This is the dawn of a new era, an era where people look more to their mobiles or their fitness watches to check on their health, rather than the traditional method of going and seeing a doctor. These apps provide a great avenue for those who are interested on tracking their fitness levels runners, cyclists, and gym goers alike. Everything can be tracked nowadays, even the standard iPhone comes with a health app built in, with a range of features.

There are several highly successful fitness apps on the market, some of these include mapmyrun, mapmyride, and the FitBit app. Both mapmyrun and mapmyride are made by the same group and are simple route trackers for running and cycling. Once your journey is finished the app then gives detailed statistics of the route and calories burned. The FitBit app is one that I use, it works in tandem with a wearable FitBit watch, which tracks your steps and monitors your heart rate.

2.1.1 Introduction

The purpose of my application is to develop an application that is valuable to gym goers and people who exercise in general who would like to track their workouts and accomplish their fitness goals. The graphical user interface of the app should look appealing to the user so as to entice them. The app should provide a pleasant experience and a provide a feeling of accomplishment after being used to encourage recurrent usage.

It should be highly accessible regardless of the user's familiarity with applications. Whether the user is a novice or is experienced, the app will be good for both. The key to this app is simplicity and this app will provide a few features popular in this market, through a simple and straight to the point application. The app will also provide the user with a fun experience.

The popular features I have aimed to include in this application are;

1. A stopwatch – A simple stopwatch for interval training when working out. A workout planner -A simple workout planner which allows you to add and delete workout routines.

2. Catalogue – A catalogue of workouts for different body parts depending on what the user wants to target on.

2.1.2 Related work

2.1.2.1 Fitness application

Brad Millington (2014) in his paper “Smartphone Apps and the Mobile Privatization of Health and Fitness” conducts extensive research on the well-known smartphone fitness apps. It points out how the apps help users to associate with the rest of the world. It also concludes that the apps place great emphasis on activity tracking to promote fitness. Juliana Chen, Janet E Cade and Margaret Allman-Farinelli (2015) in their paper “The Most Popular Smartphone Apps for Weight Loss: A Quality Assessment” analyses the quality of top 200-rated weight-loss apps available for smartphone users. Those apps available in market were less than standard quality and Behaviours Change Technique incorporation was also limited. Steven S. Coughlin, Mary Whitehead, Joyce Q. Sheats, Jeff Mastromonico, and Selina Smith (2016) in the paper “A Review of Smartphone Applications for Promoting Physical Activity” focuses on analysing the fitness apps to determine whether they help in tracking physical activity and promoting health. The study reveals that respondents of different ages prefer smartphone apps for their physical activity as it favourably help in coaching and motivating them. Lynn Katherine Herrmann and Jinsook Kim (2017) in their paper “The fitness of apps: a theory-based examination of mobile fitness app usage over 5 months” focused on the effectiveness of fitness apps by examining three fitness apps for a period of 5 months. The apps were examined based on the theory of planned behaviour (TPB) which was done by a survey and measured by t-test, sign test, fisher's exact tests. They found that the intensity of usage decreased over time as the participants were not comfortable in using the app. They concluded that the app should focus more on usefulness and ease of use in order to increase the adherence and effectiveness of apps. Maria D. Molina, and S. Shyam Sundar (2020) in the paper “Can Mobile Apps Motivate Fitness Tracking: A Study of Technological Affordances and Workout Behaviours” tries to examine whether the fitness apps drives the user to maintain workout regime. The study examined 682 profiles for analysing and disclosing the use of fitness apps. The study includes a content analysis for analysing the pivotal qualities which helps in retaining the users in a long run.

Chapter 3

ANALYSIS

3.1 Hardware and Software Requirements

The Hardware requirements are very minimal and the program can be run on most of the machines.

Processor	:	Pentium 4 Processor
Processor Speed	:	2.4 GHz
RAM	:	2 GB
Storage Space	:	40 GB

The software requirements are very minimal and the program can be run on the machines with these requirements satisfied:

Editor	:	Visual Studio Code
Operating System	:	Windows/Mac OS
IDE	:	VS Code
Backend Tool	:	JSON

3.2 Tools/ Languages/ Platform

Various tool used in making this project is given below:

Editor/IDE	:	Visual Studio Code
Operating System	:	Windows/Mac OS
Languages	:	Dart
Backend Tool	:	JSON

3.3 Functional Requirements

Flutter

Flutter is Google's Mobile SDK to build native iOS and Android apps from a single codebase. When building applications with Flutter everything towards Widgets – the blocks with which the flutter apps are built. The User Interface of the app is composed of many simple widgets, each of them handling one particular job. That is the reason why Flutter developers tend to think of their flutter app as a tree of widgets.

Compared to its contemporary technologies like React Native, Kotlin, and Java, Flutter is much better in regard to having a Single Codebase for Android and iOS, Reusable UI and Business Logic, high compatibility, performance, and productivity.

Dart

Dart is an open-source general-purpose programming language developed by Google. It supports application development in both client and server-side. But it is widely used for the development of android apps, iOS apps, IoT(Internet of Things), and web applications using the Flutter Framework.

Syntactically, Dart bears a strong resemblance to Java, C, and JavaScript. It is a dynamic object-oriented language with closure and lexical scope. The Dart language was released in 2011 but came into popularity after 2015 with Dart 2.0.

JSON

JSON (JavaScript Object Notation) is an open standard file format and data interchange format that uses human-readable text to store and transmit data objects consisting of attribute–value pairs and arrays (or other serializable values). It is a common data format with diverse uses in electronic data interchange, including that of web applications with servers.

JSON is a language-independent data format. It was derived from JavaScript, but many modern programming languages include code to generate and parse JSON-format data. JSON filenames use the extension .json.

Douglas Crockford originally specified the JSON format in the early 2000s. He and Chip Morningstar sent the first JSON message in April 2001.

Chapter 4

SYSTEM DESIGN

4.1 Home page widget-tree:

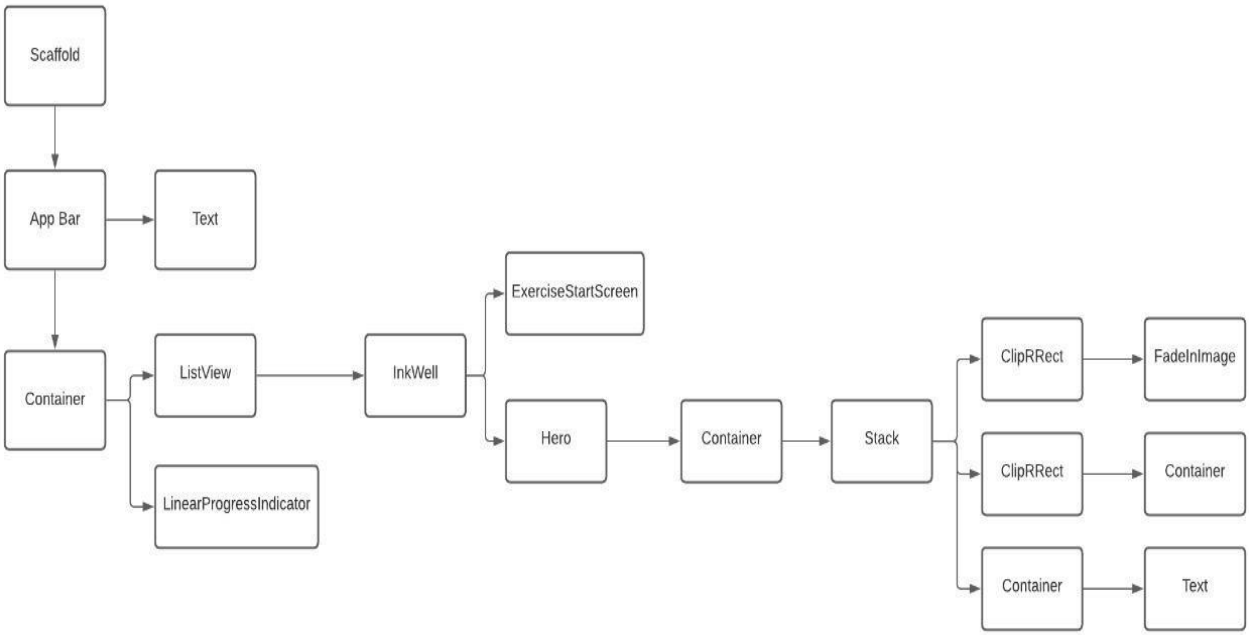

Figure 4.1

4.2 Onboarding Screen widget-tree:

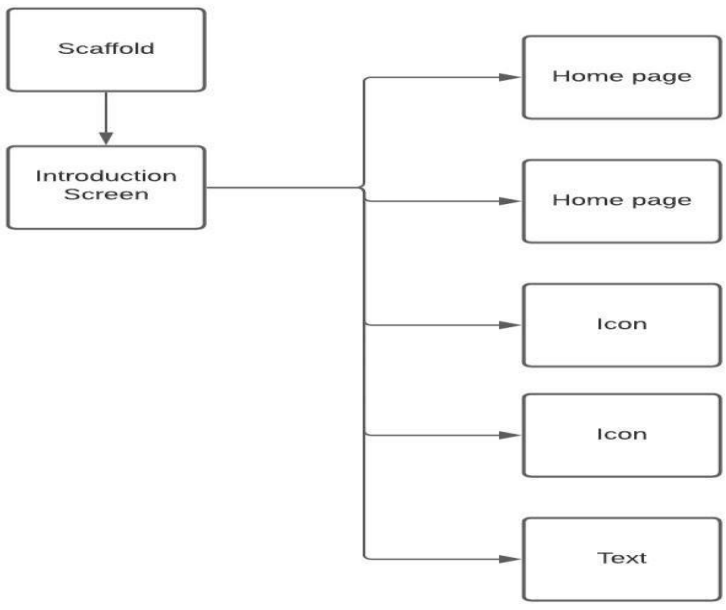

Figure 4.2

4.3 Exercise Start Screen widget-tree:

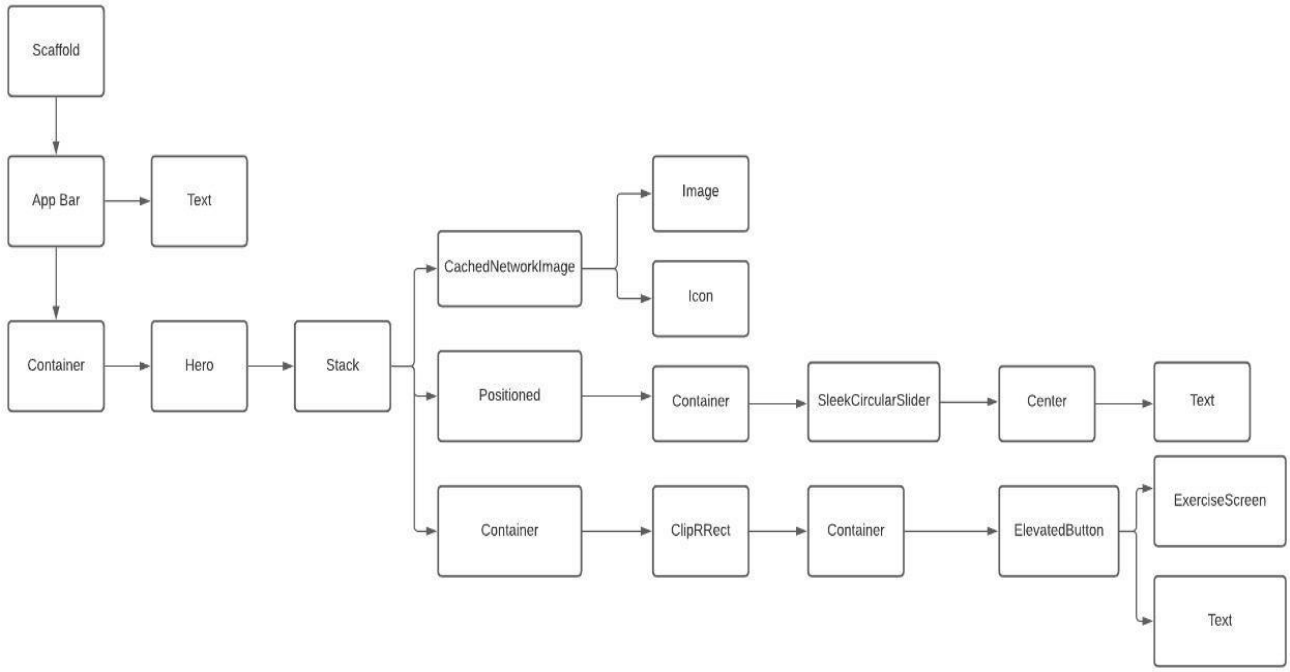

Figure 4.3

4.4 Exercise Screen widget-tree:

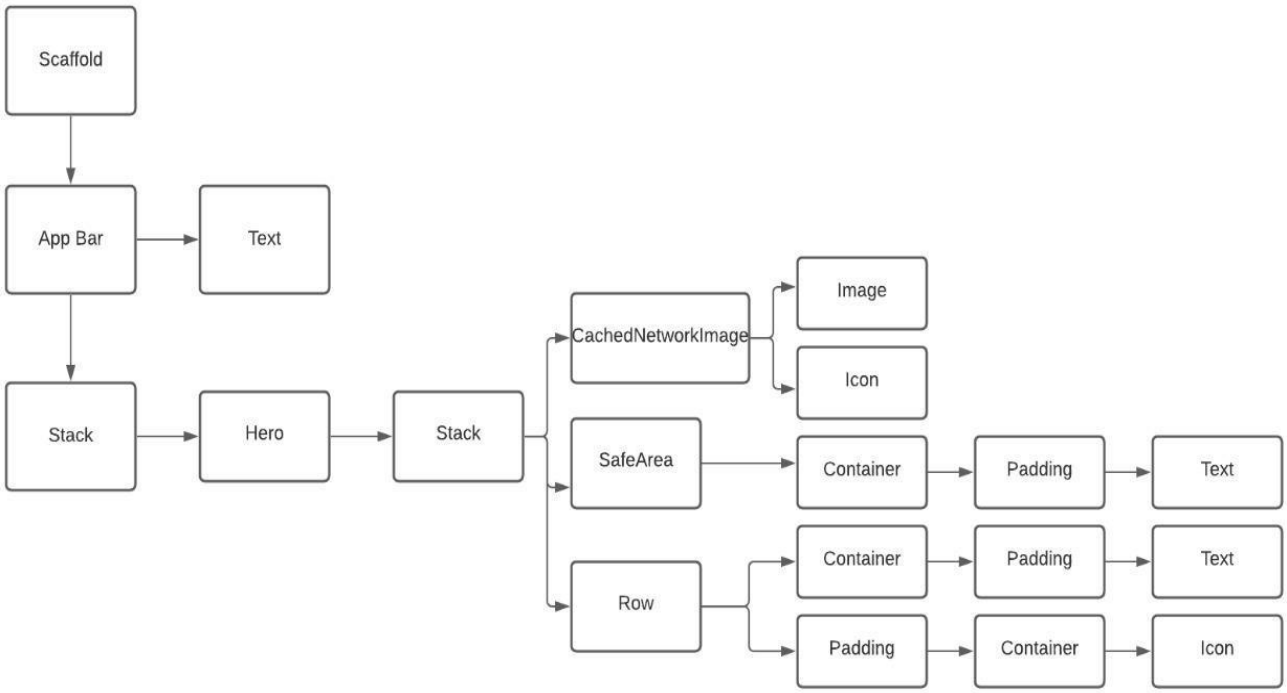

Figure 4.4

4.5 Schema design:

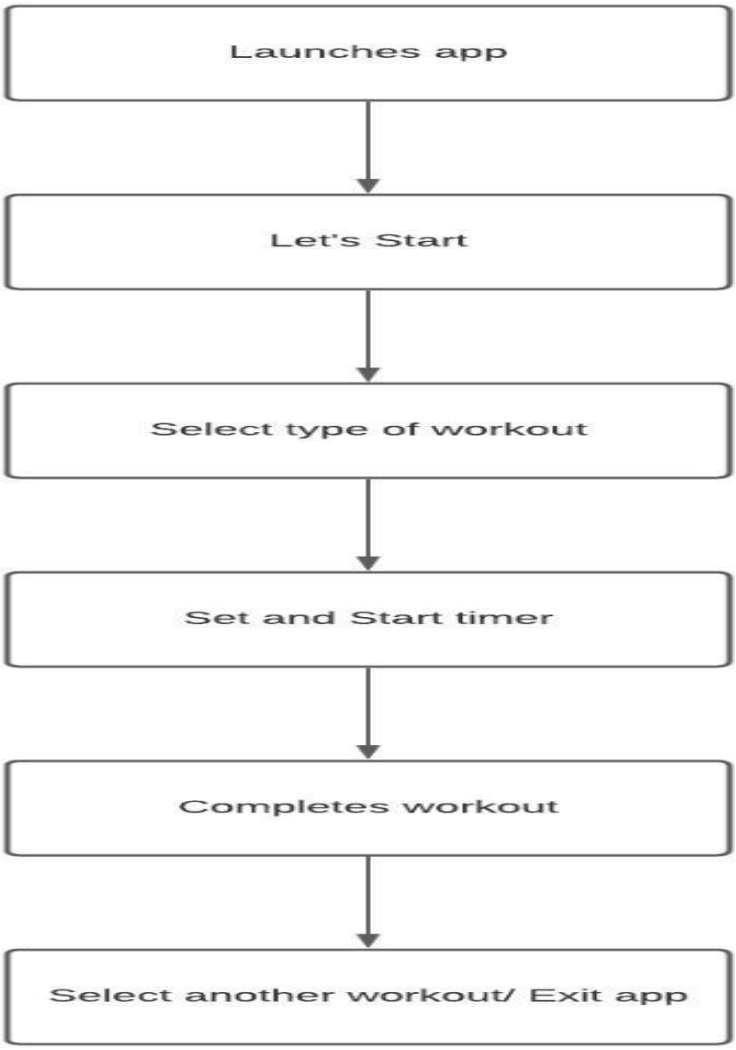

Figure 4.5

Chapter 5

IMPLEMENTATION DETAILS

5.1 main.dart

```dart
import 'package:fitness_app/screens/onboarding_screen.dart';
import 'package:flutter/material.dart';

void main() {
  runApp(const MyApp());
}

class MyApp extends StatelessWidget {
  const MyApp({Key? key}) : super(key: key);

  // This widget is the root of your application.
  @override
  Widget build(BuildContext context) {
    return const MaterialApp(
      debugShowCheckedModeBanner: false,
      title: 'Flutter Demo',
      home: OnboardingScreen(),
    );
  }
}

class MyHomePage extends StatefulWidget {
  const MyHomePage({Key? key, required this.title}) : super(key: key);


  final String title;

  @override
  State<MyHomePage> createState() => _MyHomePageState();
}

class _MyHomePageState extends State<MyHomePage> {
  int _counter = 0;

  void _incrementCounter() {
    setState(() {

      _counter++;
    });
  }
```

```
  @override
  Widget build(BuildContext context) {

    return Scaffold(
      appBar: AppBar(

        title: Text(widget.title),
      ),
      body: Center(

        child: Column(

          mainAxisAlignment: MainAxisAlignment.center,
          children: <Widget>[
            const Text(
              'You have pushed the button this many times:',
            ),
            Text(
              '$_counter',
              style: Theme.of(context).textTheme.headline4,
            ),
          ],
        ),
      ),
      floatingActionButton: FloatingActionButton(
        onPressed: _incrementCounter,
        tooltip: 'Increment',
        child: const Icon(Icons.add),
      ),
    );
  }
}
```

5.2 homepage.dart

```
import 'dart:convert';

import 'package:fitness_app/screens/exercise_hub.dart';
import 'package:fitness_app/screens/exercise_start_screen.dart';
import 'package:flutter/material.dart';
import 'package:http/http.dart' as http;


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

```
  @override
  _HomePageState createState() => _HomePageState();
}

class _HomePageState extends State<HomePage> {

  // get data from Json
  Uri apiURL = Uri.parse('https://raw.githubusercontent.com/codeifitech/fitness-app/master/exercises.json');
  var decodedjson;

  @override
  void initState() {

    getExercises();
    super.initState();
  }

  // decode json data
  void getExercises() async{
    var response = await http.get(apiURL);
    var body = response.body;
    decodedjson = jsonDecode(body);

    setState(() {
      decodedjson = jsonDecode(body);
    });

  }


  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(backgroundColor:  const Color(0xff192a56), title: const Text('Stay Fit'),centerTitle: true,),

      body: Container(
        child: decodedjson != null ? ListView.builder(
          itemBuilder: (context, index){
            return InkWell(
              onTap: (){
                Navigator.push(
                  context,
                  MaterialPageRoute(builder: (context) =>
                  ExerciseStartScreen(exercises:
                  ExerciseHub.fromJson(decodedjson).exercises[index],)
```

```
                ));
            },
            child: Hero(
              tag: ExerciseHub.fromJson(decodedjson).exercises[index].id,
              child: Container(margin: const EdgeInsets.all(10.0),
              decoration: BoxDecoration(
                borderRadius: BorderRadius.circular(16)
              ),
                child:

              Stack(
                children: [
                  ClipRRect(
                    borderRadius: BorderRadius.circular(16),
                    child :

                    // Fadein image before getting the thumbnail
                    FadeInImage(
                      placeholder:  const AssetImage('assets/placeholder.jpg'),

image:NetworkImage(ExerciseHub.fromJson(decodedjson).exercises[index].thumbnail),
                      height:  MediaQuery.of(context).size.height *0.25,
                      width: MediaQuery.of(context).size.width ,
                      fit: BoxFit.cover,
                    ),
                  ),

                  // Show Gradient
                  ClipRRect(
                    borderRadius: BorderRadius.circular(16),
                    child: Container(
                      width: MediaQuery.of(context).size.width,
                      height:  MediaQuery.of(context).size.height *0.25,
                      decoration: const BoxDecoration(
                        gradient: LinearGradient(colors:
[Color(0xFF000000),Color(0x00000000) ],
                        begin: Alignment.bottomCenter,
                        end: Alignment.center
                        ),
                      ),
                    ),
                  ),

                  // Show exercises title
                  Container(
                    height: MediaQuery.of(context).size.height *0.25,
                    alignment: Alignment.bottomLeft,
                    padding: const EdgeInsets.all(10.0),
```

```
                    child: Text(
                      ExerciseHub.fromJson(decodedjson).exercises[index].title,
                      style: const TextStyle(
                        color: Colors.white,
                        fontSize: 18.0

                      ),
                    ),
                    )

                ],
              )
              ,),
            ),
          );
        },
        itemCount:ExerciseHub.fromJson(decodedjson).exercises.length
        ) :

        // If data is yet to be fetched
        const LinearProgressIndicator(),
        ),
    );
  }
}
```

5.3 onboarding_screen.dart

```
import 'package:fitness_app/screens/homepage_screen.dart';
import 'package:flutter/gestures.dart';
import 'package:flutter/material.dart';
import 'package:introduction_screen/introduction_screen.dart';

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

  @override
  _OnboardingScreenState createState() => _OnboardingScreenState();
}

// A list of three screens shown as a brief introduction

var pages = [
  PageViewModel(
  title: "Stay fit with our tips!",
  body: "get in shape and follow the tutorials, make sure to complete your timer",
```

```
  image: Center(
    child: Image.asset("assets/screen1nobg.png", height: 250.0),
  ),
  decoration: const PageDecoration(
    pageColor: Color(0xff192a56),
    titleTextStyle: TextStyle(color: Colors.white, fontSize: 24.0),
    bodyTextStyle: TextStyle(color: Colors.white, fontSize: 18.0)

  )
),

PageViewModel(
  title: "Exercise Now!",
  body: "Need motivation? We got you covered! Complete your time to get your
applause from the audience",
  image: Center(
        child: Image.asset("assets/screen2nobg.png", height: 200.0),
  ),
  decoration: const PageDecoration(
    pageColor: Color(0xff192a56),
    titleTextStyle: TextStyle(color: Colors.white, fontSize: 24.0),
    bodyTextStyle: TextStyle(color: Colors.white, fontSize: 18.0)

  )
),

PageViewModel(
  title: "Let's start",
  body: "What are you waiting for? Start exercise today",
  image: Center(
       child: Image.asset("assets/screen3nobg.png", height: 200.0),
  ),
  decoration: const PageDecoration(
    pageColor: Color(0xff192a56),
    titleTextStyle: TextStyle(color: Colors.white, fontSize: 24.0),
    bodyTextStyle: TextStyle(color: Colors.white, fontSize: 18.0)

  )
)
];

class _OnboardingScreenState extends State<OnboardingScreen> {
  @override
  Widget build(BuildContext context) {
    return Scaffold(

      body: IntroductionScreen(
        globalBackgroundColor: const Color(0xff192a56),
```

```
  pages: pages,
  onDone: () {
    Navigator.pushReplacement(
      context,
      MaterialPageRoute(builder: (context) => HomePage()));
    // When done button is press
  },
  onSkip: () {
    Navigator.pushReplacement(
      context,
      MaterialPageRoute(builder: (context) => HomePage()));
    // You can also override onSkip callback
  },
  showSkipButton: true,
  skip: const Icon(Icons.skip_next, color: Colors.white,),
  next: const Icon(Icons.arrow_right, color: Colors.white,),
  done: const Text("Done", style: TextStyle(fontWeight: FontWeight.w600, color:
Colors.white,)),
  dotsDecorator: DotsDecorator(
    size: const Size.square(10.0),
    activeSize: const Size(20.0, 10.0),
    activeColor: Colors.amber,
    color: Colors.white,
    spacing: const EdgeInsets.symmetric(horizontal: 3.0),
    activeShape: RoundedRectangleBorder(
      borderRadius: BorderRadius.circular(25.0)
    )
  ),
),


    );
  }
}
```

5.4 exercise_screen.dart

```
import 'dart:async';
import 'package:audioplayers/audioplayers.dart';
import 'package:cached_network_image/cached_network_image.dart';
import 'package:fitness_app/screens/exercise_hub.dart';
import 'package:flutter/cupertino.dart';
import 'package:flutter/material.dart';
import 'package:flutter/painting.dart';
import 'package:flutter/rendering.dart';
```

```
class ExerciseScreen extends StatefulWidget {
  final Exercises exercises;
  final int seconds;


  ExerciseScreen({required this.exercises, required this.seconds});


  @override
  _ExerciseScreenState createState() => _ExerciseScreenState();
}

class _ExerciseScreenState extends State<ExerciseScreen> {

  bool _iscomplete=false;
  int _elapsedSeconds=0;

  late Timer timer;

  AudioPlayer audioPlayer = AudioPlayer();
  AudioCache audioCache =  AudioCache();

  // dispose timer when screen is changed
  @override
  void dispose(){
    timer.cancel();
    super.dispose();
  }

  //play audio when exercise is completed
  void audioplay() {
    audioCache.play("applause.wav");
  }

  @override
  void initState(){

    // Increase timer after every second
    timer = Timer.periodic(Duration(seconds: 1), (t) {
      if (t.tick == widget.seconds){
        t.cancel();
        setState(() {
          _iscomplete = true;
          audioplay();
        });
      }

      setState(() {
```

```
        _elapsedSeconds = t.tick;
      });

    });

    super.initState();
  }


  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(backgroundColor:  const Color(0xff192a56), title: const
Text('Stay Fit'),centerTitle: true,),

    //  Display exercise gif and elapsed time
      body: Stack(
        children: [
          Container(
            height: MediaQuery.of(context).size.height ,
            width: MediaQuery.of(context).size.width ,
            child:

            // Create Hero Animation
            Hero(
              tag: widget.exercises.id,
              child: Stack(
                children: [
                  CachedNetworkImage(
                    imageUrl: widget.exercises.gif,
                    placeholder: (context,url) =>  Image(
                      image:  const AssetImage('assets/placeholder.jpg'),
                      height: MediaQuery.of(context).size.height,
                      width: MediaQuery.of(context).size.width ,
                      fit: BoxFit.fitWidth,


                    ),
                    errorWidget: (context, url, error) => Icon(Icons.error),
                    height: MediaQuery.of(context).size.height ,
                      width: MediaQuery.of(context).size.width,
                      fit: BoxFit.fitWidth,
                  ),

                  // If exercise is not completed, show the elapsed and remaining
time
                  _iscomplete!= true ?
```

```
                SafeArea(child:
                Container(
                  alignment: Alignment.topCenter,
                  child: Padding(
                    padding: const EdgeInsets.all(10.0),
                    child: Text("$_elapsedSeconds / ${widget.seconds} Seconds",
style: const TextStyle(
                      fontSize: 20.0,
                      color: Color(0xff5E3C6B),
                      fontFamily: 'Times Roman',
                      fontWeight: FontWeight.bold
                    ),
                    ),
                  ),
                  )
                  ) :

                  // If exercise is completed, show check
                  Row(
                    mainAxisAlignment: MainAxisAlignment.center,
                    children: [
                       Container(
                  alignment: Alignment.topCenter,
                  child: Padding(
                    padding: const EdgeInsets.all(10.0),
                    child: Text("$_elapsedSeconds / ${widget.seconds} Seconds",
style: const TextStyle(
                      fontSize: 20.0,
                      color: Color(0xff5E3C6B),
                      fontFamily: 'Times Roman',
                      fontWeight: FontWeight.bold


                    ),
                    ),
                  ),
                  ),
                      Padding(
                        padding: const EdgeInsets.all(10.0),
                        child: Container(
                          alignment: Alignment.topCenter,
                          child: const Icon(Icons.check_box_sharp
                          , color: Color(0xff5E3C6B),)),
                      )
                    ],
                  )
              ]
            ),
```

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

5.5 exercise_start_screen.dart

```
import 'package:cached_network_image/cached_network_image.dart';
import 'package:fitness_app/screens/exercise_hub.dart';
import 'package:fitness_app/screens/exercise_screen.dart';
import 'package:flutter/cupertino.dart';
import 'package:flutter/material.dart';
import 'package:flutter/painting.dart';
import 'package:flutter/rendering.dart';
import 'package:flutter/widgets.dart';
import 'package:sleek_circular_slider/sleek_circular_slider.dart';


class ExerciseStartScreen extends StatefulWidget {
  final Exercises exercises;


  ExerciseStartScreen({required this.exercises});

  @override
  _ExerciseStartScreenState createState() => _ExerciseStartScreenState();
}

class _ExerciseStartScreenState extends State<ExerciseStartScreen> {
  int seconds = 30;

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(backgroundColor:  const Color(0xff192a56), title: const
Text('Stay Fit'),centerTitle: true,),


      body:

      // To prvide custom height and width
      Container(
        height: MediaQuery.of(context).size.height ,
        width: MediaQuery.of(context).size.width ,
```

```
        child: Hero(
          tag: widget.exercises.id,
          child: Stack(
            children: [

              // Show placeholder till the image is fetched
              CachedNetworkImage(
                            imageUrl: widget.exercises.thumbnail,
                            placeholder: (context,url) =>  Image(
                              image:  const AssetImage('assets/placeholder.jpg'),
                              height: MediaQuery.of(context).size.height,
                              width: MediaQuery.of(context).size.width ,
                              fit: BoxFit.fitWidth,


                            ),
                            errorWidget: (context, url, error) =>
Icon(Icons.error),
                            height: MediaQuery.of(context).size.height ,
                              width: MediaQuery.of(context).size.width,
                              fit: BoxFit.fitWidth,

                            ),

                        // To show timer at the bottom of the screen
                        Positioned.fill(
                          bottom: 30,
                          left: 0,
                          right: 0,

                          child:
                        Container(

                          alignment: Alignment.bottomCenter,

                          child: SleekCircularSlider(
                            appearance: const
CircularSliderAppearance(animationEnabled: true)
                            ,
                            initialValue: 30,
                            max: 60,
                            min: 2,

                            // Customize inner widget
                            innerWidget: (v){

                             return Center(
```

```
                        child: Text("${v.toInt()} Sec", style: const
TextStyle(
                          fontFamily: "Times New Roman",
                          color: Color(0xff5E3C6B),
                          fontSize: 25.0,

                        )),
                      );

                    },
                    onChange: (double v){
                      seconds = v.toInt();

                    },
                  ),
                  )
                  ),

                  // Start Button
                  Container(
                    alignment: Alignment.bottomCenter,

                    child: ClipRRect(
                      borderRadius: BorderRadius.circular(25),
                      child: Container(
                        height: 50,
                        width: 120,

                        child: ElevatedButton(onPressed: (){
                          Navigator.push(
                          context,
                          MaterialPageRoute(builder: (context) =>
                           ExerciseScreen(exercises:
                           widget.exercises,
                           seconds: seconds)
                            ));
                        },
                        style: ElevatedButton.styleFrom(primary: const
Color(0xff5B54FA), ),

                        child: const Text("Start", style: TextStyle(
                          color: Colors.white,


                        ),

                        ),
```

```
                        ),


                          ),

                        ),

                      )

          ],
        )

        /* FadeInImage(
          image: NetworkImage(widget.exercises.thumbnail),
          placeholder: AssetImage("assets/placeholder.jpg"),
          height: MediaQuery.of(context).size.height * 0.25,
          width: MediaQuery.of(context).size.width ,


        ), */
        ),


      ),
    );
  }
}
```

Chapter 6

TESTING

6.1 Introduction

Testing is a process of executing a program with the interest of finding an error. A good test is one that has high probability of finding the yet undiscovered error. Testing should systematically uncover different classes of errors in a minimum amount of time with a minimum number of efforts. Two classes of inputs are provided to test the process

A software configuration that includes a software requirement specification, a design specification and source code.

A software configuration that includes a test plan and procedure, any testing tool and test cases and their expected results.

6.2 Levels of Testing

6.2.1 Unit Testing

Unit testing is a level of software testing where individual units/ components of a software are tested. The purpose is to validate that each unit of the software performs as designed. A unit is the smallest testable part of any software. It usually has one or a few inputs and usually a single output.

Unit testing is commonly automated, but may still be performed manually. The objective in unit testing is to isolate a unit and validate its correctness. A manual approachto unit testing may employ a step-by-step instructional document. The unit testing is the process of testing the part of the program to verify whether the program is working corrector not. In this part the main intention is to check the each and every input which we are inserting to our file. Here the validation concepts are used to check whether the program istaking the inputs in the correct format ornot.

Unit testing may reduce uncertainty in the units themselves and can be used in a bottom-up testing style approach. By testing the parts of a program first and then testing the sum of its parts, integration testing becomes much easier. Unit test cases embody characteristics that are critical to the success of the unit.

6.2.2 Integration Testing

Integration testing is also taken as integration and testing this is the major testing process where the units are combined and tested. Its main objective is to verify whether the major parts of the program is

working fine or not. This testing can be done by choosing the optionsin the program and by giving suitable inputs.

6.2.3 System Testing

System testing is defined as testing of a complete and fully integrated software product. This testing falls in black-box testing wherein knowledge of the inner design of the code isnot a pre-requisite and is done by the testing team. System testing is done after integrationtesting is complete. System testing should test functional and non-functional requirementsof the software.

6.2.4 Validation Testing

In this, requirements established as part of software requirements analysis are validated against the software that has been constructed. Validation testing provides final assurance that software meets all functional, behavioral and performance requirements. Validation can be defined in many ways but a simple definition is that validation succeeds when software Function in a manner that can be reasonably by the customer.

1. Validation test criteria
2. Configuration review
3. Alpha and Beta testing (conducted by end user)

6.2.5 Output Testing

After preparing test data, the system under study is tested using the test data. While testing the system using test data, errors are again uncovered and corrected by using above testing and corrections are also noted for future use.

6.2.6 User Acceptance Testing

User acceptance testing is a type of testing performed by the end user or the client to verify/accept the software application to the production environment.

User Acceptance Testing is done in the final phase of testing.

Chapter 7

DISCUSSION OF RESULTS

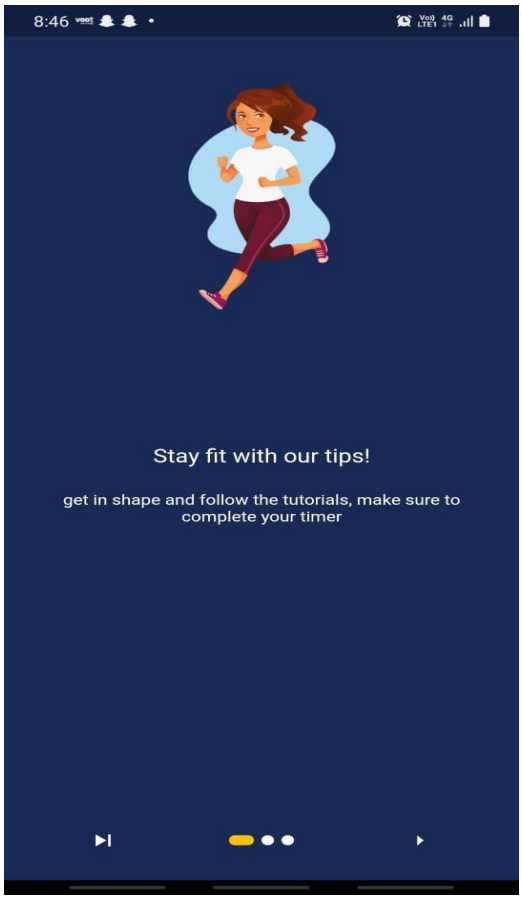

Figure 7.1

This screen onboards the customers into the app while painting a general image of what to expect in the main screen. It also displays a few quotes in three different slides and screens to give encouragement in the motivational sense to exercise and stay fit.

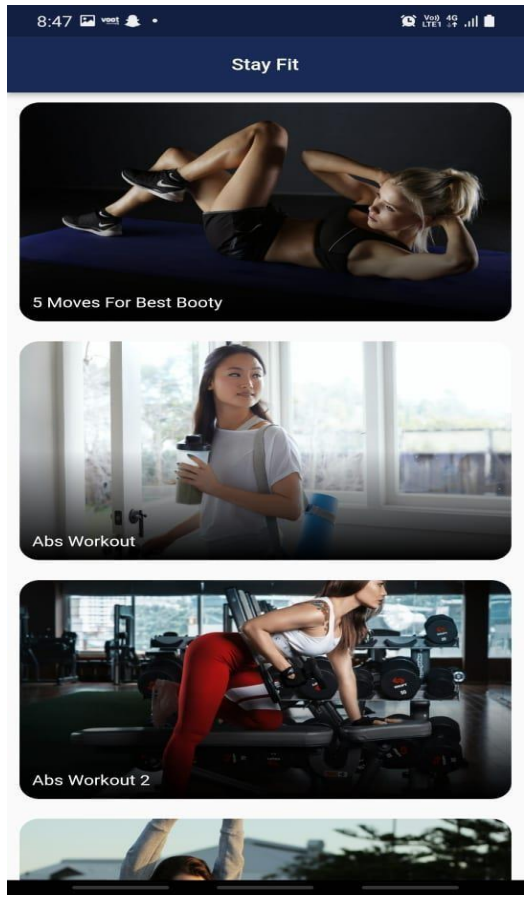

Figure 7.2

The On-Boarding screen leads next to the home page which contains a catalogue of exercises targeting specific body parts depending on the user wants to target for that particular workout session.

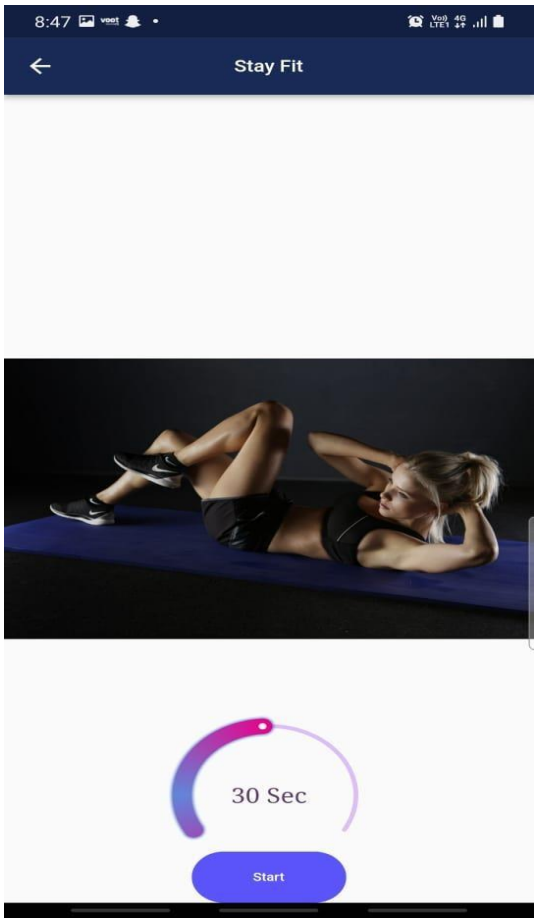

Figure 7.3

This page allows us to set the timer for the particular workout routine and start the exercise.

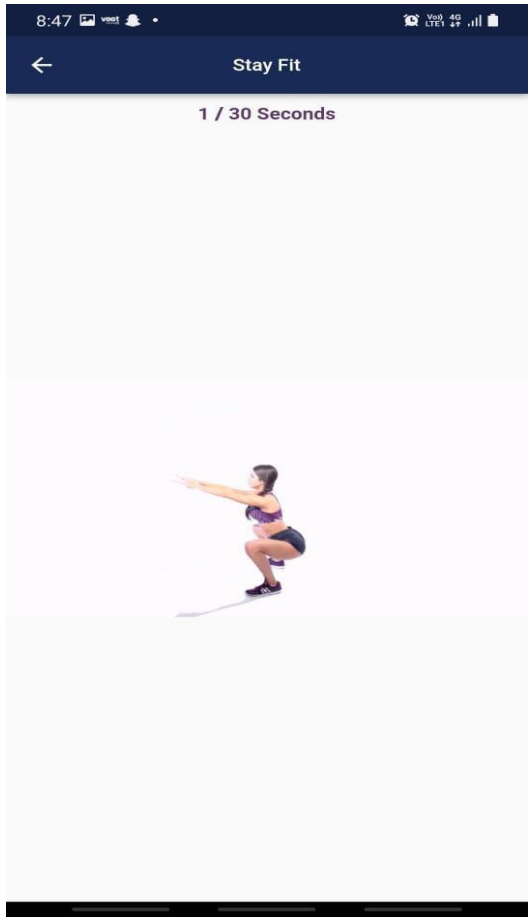

Figure 7.4

This screen shows a gif of the complete exercise in proper form so that the user can follow along with it and each exercise is performed for the time that was set by the user in the previous screen. Once the whole workout is completed, there's a "cheers from the audience" indication, which serves as an appreciation note and also indicates that the workout is complete.

Chapter 8

CONCLUSION AND FUTURE REFERENCES

8.1 Conclusion

This report gives an insight into the background as to why I chose to develop this application. I believed that there was a problem with apps of this type and that if done a certain way this could be solved. The idea and my motivations have been stated clearly. The requirements are described in detail and the technologies used have also been outlined in this document. I have touched on the Systems design to explain it further. I believe that I achieved most of what I have set out to do when this idea was conceived. However, there are multiple different areas which could be improved and lots of room for expansion.

8.2 Future Reference

In the future, this application could be greatly improved and expanded to include new features. The tracker and step counter can be improved. A calorie counter could be added to allow users keep track of their daily intake and pursue their weight loss goals. This would mean populating a database with vast amounts of food and nutritional data and allowing the user to enter food eaten after every meal. Once entered the app takes the number of calories from their daily allowance. An exercise instruction manual which advises users of exercises and how to do them.

Chapter 9

REFERENCES

1. **https://rubygarage.org**
2. **www.hackernoon.com**
3. **www.ncbi.nlm.nih.gov**
4. **www.ispo.com**
5. **www.github.com**
6. **www.stackoverflow.com**
7. Adria Muntaner-Mas, Antonio Martinez-Nicolas, Carl J. Lavie, Steven N. Blair, Robert Ross, Ross Arena, and Francisco B. Ortega (2019). A Systematic Review of Fitness Apps and Their Potential Clinical and Sports Utility for Objective and Remote Assessment of Cardiorespiratory Fitness. Sports Medicine 2019, 49(4), 587-600. doi:10.1007/s40279-019-01084-y