Fitness App: React Native

A PROJECT REPORT

Submitted By

Varun Nikte (160770107247)
Arjun Gharti Chhetri (160770107253)
Jay Soni (160770107221)
Dhaval Deshkar (160770107041)
Shrenik Mehta (160770107079)

In fulfillment for the award of the degree of

BACHELOR OF ENGINEERING

In

Computer Engineering

SILVER OAK COLLEGE OF ENGINEERING AND TECHNOLOGY

OPP. BHAGWAT VIDYAPITH, NEAR GOTA CROSS ROAD,

AHMEDABAD- 382481

Silver Oak College of Engineering & Technology

Computer Engineering
2019-2020

CERTIFICATE

Date:

This is to certify that the project entitled "**Fitness App Through React Native"** has been carried out by "Varun Nikte (160770107247)", "Arjun Chhetri (160770107253)", "Jay Soni (160770107221)", "Dhaval Deshkar (160770107041)", and "Shrenik Mehta (160770107079)" under my guidance in fulfillment of the Degree of Bachelor of Engineering in **Computer Engineering**–8th Semester of Gujarat Technological University, Ahmedabad during the academic year 2019-2020.

Name of Guide

(Computer Engineering)

Head of Department

(Computer Engineering)

Candidate's Declaration

We hereby declare that project report titled "FITNESS APP THROUGH REACT NATIVE" submitted towards the completion of project in 8th semester of B.E Computer in Silver Oak College of Engineering & Technology, Ahmedabad is an authenticate record of our work carried out.

We further declare that to the best of our knowledge the report of Computer Engineering 8th semester.

Candidate's signature	**:**
First candidate's name	**: Varun Nikte**
Branch	**: CE**
Enrollment Number	**: 160770107247**
Candidate's signature	**:**
Second candidate's name	**: Arjun Chhetri**
Branch	**: CE**
Enrollment Number	**: 160770107253**
Candidate's signature	**:**
Third candidate's name	**: Jay Soni**
Branch	**: CE**
Enrollment Number	**: 160770107221**

Candidate's signature	**:**
Fourth candidate's name	**: Dhaval Deshkar**
Branch	**: CE**
Enrollment Number	**: 160770107041**
Candidate's signature	**:**
Fifth candidate's name	**: Shrenik Mehta**
Branch	**: CE**
Enrollment Number	**: 160770107079**

Submitted to:

Silver Oak College of Engineering & Technology, Ahmedabad

Affiliated to:

Gujarat Technological University

Abstract

- A daily fitness routine helps in to develop your overall health and the positive effects become very apparent when you invest time to exercise.

- An active walk for 30-35 plus minute and some basic resistance training is all you need to get the ball rolling. This will get the blood circulating, burns those annoying calories, and improves immunity and physical strength… who doesn't need a raised immune system? The basic take away?

- **Why is it Important to Be Physically Fit?**

  People who are physically fit are also healthier, are able to maintain their most optimum weight, and are also not prone to cardiac and other health problems.

  In order to maintain a relaxed state of mind, a person should be physically active. A person who is fit both physically and mentally is strong enough to face the ups and downs of life, and is not affected by drastic changes if they take place.

  By becoming more active you can increase your body's fitness levels and also avoid health problems like diabetes and high blood pressure from developing. Exercise is also good for your joints and makes your body stronger overall.

ACKNOWLEDGEMENT

We are heartily thankful to our supervisor, Prof. Darshil. Shah, whose encouragement, supervision and support from the preliminary to the concluding level enabled me to develop an understanding of the subject. At the end, we offer our regards and blessings to all of those who supported us in any respect during the completion of the project and to our college for providing a resources and materials.

INDEX

LIST OF FIGURES

INDEX

LIST OF TABLES

INDEX

INDEX

INDEX

INDEX

CHAPTER: 1

INTRODUCTION

CHAPTER 1

INTRODUCTION

1.1 Project Summary

- This is a Total Fitness Mobile based Application. Here we provide you fitness guidelines for all the health-conscious fans.
- Fit n Fab offers you a big amount of info that will help you to reach your Fitness goals.
- This App can work Without an Internet Connection.
- Fit n Fab has many exercises, workouts, Yoga, Meditation, Diet and BMI Calculator also for training at your gym or at home.

1.1.1Purpose:

- A daily fitness routine helps in to develop your overall health and the positive effects become very apparent when you invest time to exercise
- People who are physically fit are also healthier, are able to maintain their most optimum weight, and are also not prone to cardiac and other health problems.
- In order to maintain a relaxed state of mind, a person should be physically active. A person who is fit both physically and mentally is strong enough to face the ups and downs of life, and is not affected by drastic changes if they take place.

1.2 Scope:

The scopes are mention below:

- This App can be made available on various platforms like the Google Play Store and Apple IOS Store

- This is a Total Fitness Mobile based Application. Here we provide you fitness guidelines for all the health-conscious fans.

- Fit n Fab offers you a big amount of info that will help you to reach your Fitness goals.

- This App can work Without an Internet Connection.

- Fit n Fab has many exercises, workouts, Yoga, Meditation, Diet and BMI Calculator also for training at your gym or at home.

1.3 Objective:

The main objective of this project is to achieve physical fitness and improve the level of health of users by helping and motivating them to doing sports exercise and eating healthy food.

1.3.1 Specific Objective

This project will provide the following service as specific objective:

- To develop the attention in the health fitness and nutrition.
- To motivate the interest of health and physical fitness and nutrition.
- To manage the healthy life system in health fitness and nutrition.
- To facilitate the connection between nutrition and health, fitness and fashion in one application to not distract the user more than one application.

1.4 Technology and Literature Review:

- HTML
- JavaScript
- React Native

HTML

- HTML an initialize of Hyper Text Markup Language for web pages.
- It provides a means to describe the structure of text-based information in document by denoting text as headings, paragraphs, lists and so on and to supplement that text with interactive forms, embedded images and other objects.

JavaScript

- JavaScript supports the development of both client and server components of web-based applications.
- On the client side, it can be used to write programs that are executed by a web browser within the context of the web page.
- On the server side, it can be used to write web server programs that can be process information submitted by a web browser and then update the web browser display accordingly.

React Native

- React Native lets you build mobile apps using only JavaScript. It uses the same design as React, letting you compose a rich mobile UI from declarative components.
- With React Native, you don't build a mobile web app, an HTML5 app, or a hybrid app; you build a real mobile app that's indistinguishable from an app built using Objective-C or Java. React Native uses the same fundamental UI building blocks as regular iOS and Android apps.
- You just put those building blocks together using JavaScript and React.

React Native Features

- **React** − This is a Framework for building web and mobile apps using JS
- **Native** − You can use native components controlled by JavaScript.
- **Platforms** − React Native supports IOS and Android platform.

React Native Advantages

- **JavaScript** − You can use the existing JavaScript knowledge to build native mobile apps.
- **Code sharing** − You can share most of your code on different platforms.
- **Community** − The community around React and React Native is large, and you will be able to find any answer you need.

React Native Limitations

- **Native Components** − If you want to create native functionality which isnot created yet, you will need to write some platform specific code.

CHAPTER: 2

SOFTWARE PROJECT MANAGEMENT

CHAPTER 2

SOFTWARE PROJECT MANAGEMENT

2.1 Project planning and scheduling

2.1.1 Project Planning

- Project planning is part of project management, which relates to the use of schedules such as Gantt charts to plan and subsequently report progress within the project environment.
- Initially, the project scope is defined and the appropriate methods for completing the project are determined. Following this step, the durations for the various tasks necessary to complete the work are listed and grouped into a work breakdown structure.
- Project planning is often used to organize different areas of a project, including project plans, workloads and the management of teams and individuals.

2.1.2 Project Scheduling

- Project Scheduling is the culmination of a planning activity that is primary component of software project management.
- When combined with estimation methods and risk analysis, scheduling, establishes a road map for the project management.
- Scheduling begins with the process composition. The characteristics of the project are used to adapt an appropriate task set for the work to be done.
- The task network is used to compute the critical project path, a time line chart and a variety of project information.

2.2 Project Development Approach

The activities we followed for this project is listed below:

- Planning the work or objectives
- Analysis & Design of objectives
- Assessing and controlling risk

- Allocation of resources
- Organizing the work
- Database Designing
- Form Design

The Process Paradigm we used for our project is Incremental Model.

The Incremental Software Process Model

- The Incremental Model combines elements of the linear sequential model with the iterative philosophy of prototyping. The incremental model applies linear sequences in a staged fashion as calendar time progresses.
- Each linear sequence produces a deliverable "increment" of the software. For example, word processing software developed using the incremental paradigm might deliver basic file management, editing and document production functions in the first increment; more sophisticated editing and document production capabilities in the second increment; spelling and grammar checking in the third increment; and advanced page layout capability in the fourth increment.
- It should be noted that the process flow for any increment can incorporate the prototyping paradigm.
- When an incremental model is used, the first increment is often a core product. That is, basic requirements are addressed, but many supplementary features remain undelivered.
- The core product is used by the customer. As a result of use and/or evaluation, a plan is developed for the next increment. The plan addresses the modification of the core product to better meet the needs of the customer and the delivery of additional features and functionality.
- This process is repeated following the delivery of each increment, until the complete product is produced.
- The Incremental process model, like prototyping and other evolution approaches, is iterative in nature.
- But unlike prototyping, the incremental model focuses on the delivery of an operational product with each increment.

- Early increments are stripped down versions of the final product, but they do provide capability that serves the user and also provide a platform for evaluation by the user.

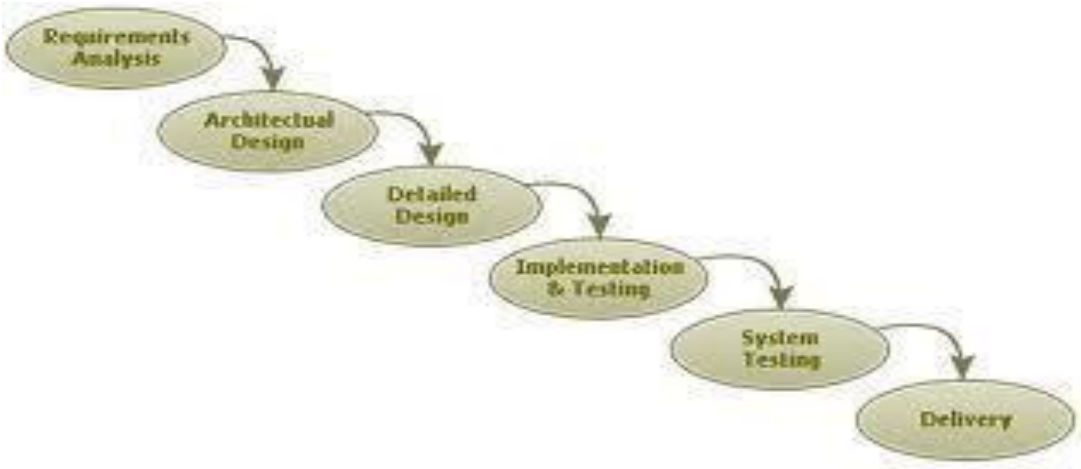
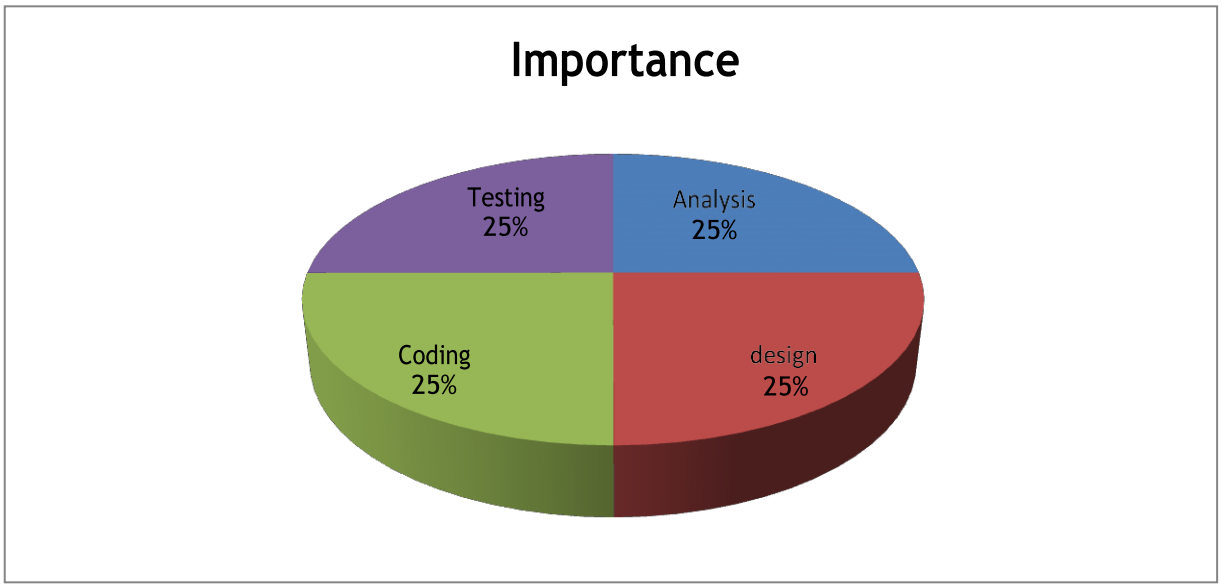


Figure 2.1 Incremental Model

2.3 Project Plan

	From Date	**To Date**
1.Preliminary Investigation	1/7/2019	1/8/2019
2.Requirment Analysis	3/8/2019	3/9/2019
3.Designing	5/9/2019	31/10/2019
4.Implementation	5/11/2020	28/2/2020
5.Testing	01/3/2020	01/4/2020

[Table 2. 1 Project Plan]



[Figure 2.2 Pie Chart]

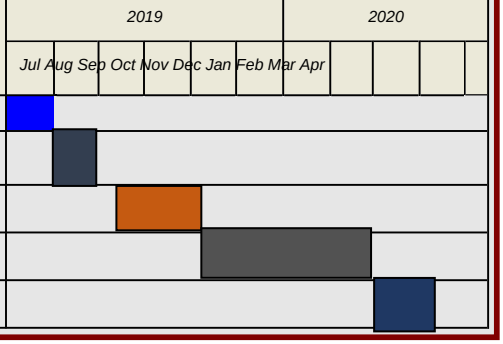
2.3.1 Milestone and Deliverables

- In this project, we went through Module Wise Completion. First, we did analysis of first module; we went through all the requirements for first module that is Admin Module.
- By this analysis we decided field of all the tables of Admin Module. Then we started Database Design.
- After competing it we started with the design of all forms of this module. Then we did coding and finally validations and testing of forms that we made.
- After completing the first module we started the same procedure for the Member Module.
- After that we completed User Module. Between this we did settings forms e.g. Change theme, change background color of the main screen etc.
- There was continuously interaction with the client that was very beneficial for us.
- When we completed whole project, we started testing the whole project for final verification.
- Then we started documentation of our project. Finally, we completed the project with client's satisfaction.
- A milestone is an end point of software process activity.

Software Process Activity	**Milestones**
Project Plan	Project Schedule
Requirement Collection	User requirements
Data Flow Analysis	System Flow
Design 1. Database Design 2. User Interface Design 3. System Design	System Design Document
Implementation 1. Code For giving security 2. Code for reports	Access Reports Reports
Testing	Setting validation and error message

2.3.2 Gantt Chart

ID	Task Name	Start	Finish	Duration
1	Preliminary Investigation	01/07/2019	01/08/2019	4 w
2	Requirement Analysis	03/08/2019	03/09/2019	4 w
3	Design	05/09/2019	31/10/2019	7.2w
4	Implementation	05/11/2019	28/02/2020	14 w
5	Testing	0103/2020	01/04/2020	4.4w



2.3.3 Cost Estimation:

The COCOMO Model

- Like all estimation models for software, the COCOMO models require sizing information.
- Three different sizing options are available as part of the model hierarchy: object points, function points, and lines of source code.
- Like function points, the object point is indirect software that is computed using counts of the number of

(1) Screens (at the user interface),

(2) Reports,

(3) Components likely to be required to build the application.

- Once complexity is determined, the number of screens, reports, and components are weighted according to Table above.
- The object point count is then determined by multiplying the original number of object instances by the weighting factor in table above and summing to obtain a total object point count.
- When component-based development or general software reuse is to be applied, the percent of reuse (%reuse) is estimated and the object point count is adjusted:
- **NOP = (object points) X [(100 - %reuse) / 100].**

- Where NOP is defined as new object points. To derive an estimate of effort based on the computed NOP value, a "productivity rate" must be derived.
- **PROD=NOP / person-month**
- For different levels of developer experience and development environment maturity. Once the productivity rate has been determined, an estimate of project effort can be derived as Estimated effort = NOP/PROD.
- There are three types of software project:
    - Organic project
    - Semi-deteched project
    - Embedded project

Cost required to develop project=effort*rs/month

- **Effort Estimation (E):**

  In Organic=2.4 $(KLOC)^{1.05}$ PM

  In semidetached=3.0$(KLOC)^{1.12}$ PM

  In Embedded=3.6$(KLOC)^{1.20}$PM

- **Duration Estimation (D):**

  In Organic=2.5$(effort)^{0.38}$months

  In semidetached=2.5$(effort)^{0.35}$ months

  In Embedded=2.5$((effort)^{0.32}$months

- **Person Estimation:**

  P=E/D

  KLOC=Kilo Line of Code

In Or Project,

Modules	Page	Estimated line of code
Admin	Sign Up	
	Login	
	Index	
	Procedure	
	Prediction	
	By Cities	
	By Salary	
	By Years	
	Prediction	
	Contact Us	
	Total	

[Table 2. 3 Cost Estimation]

- Total line of code=•
  KLOC=

We are using Organic Project Type,

Effort Estimation (E):

=

=

Duration Estimation (D):

=

=

=

Advantages of COCOMO:

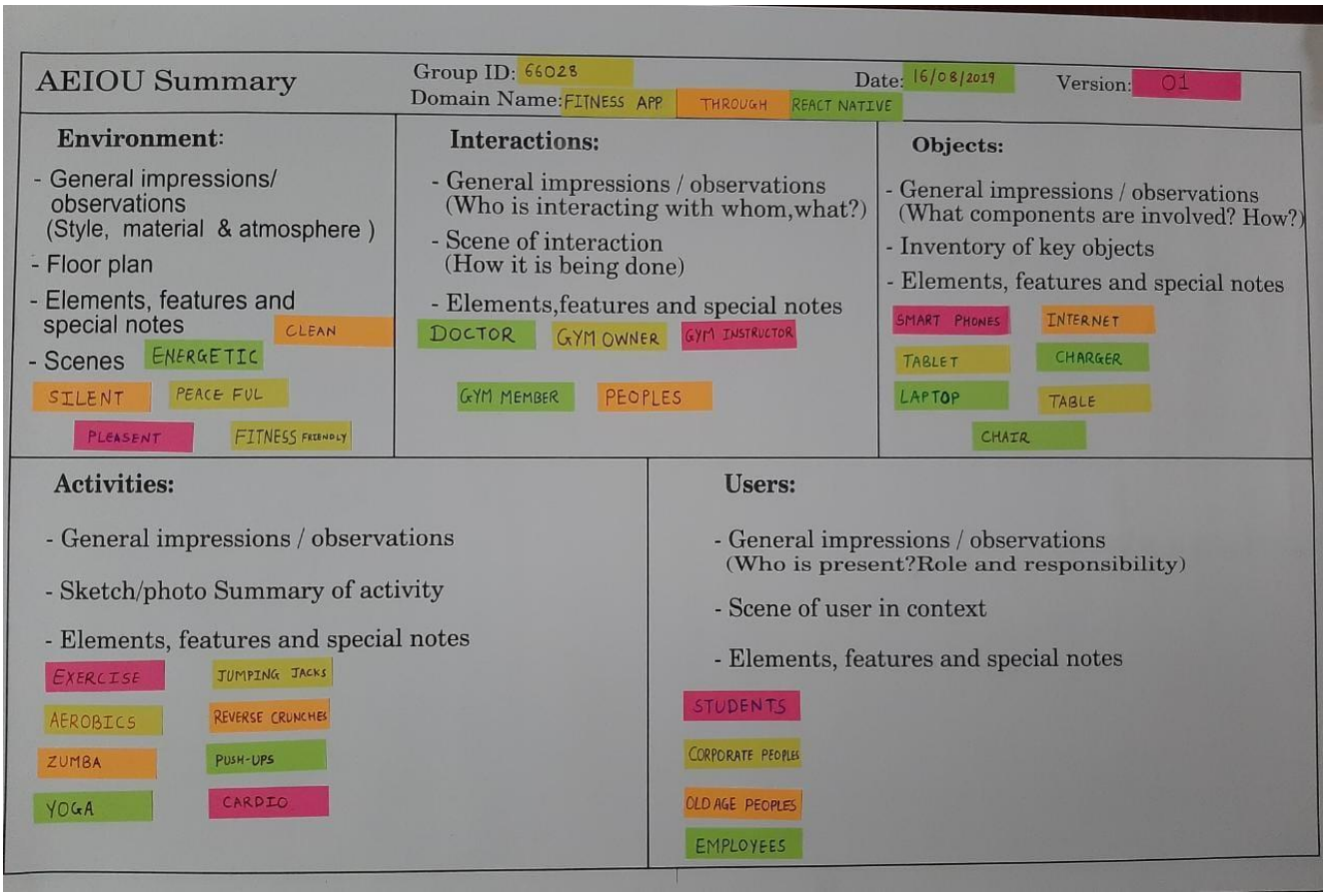
- COCOMO is factual and easy to interpret.
- One can clearly understand how it works.
- Accounts for various factors that affect cost of the project.
- Works on historical data and hence is **more** predictable and accurate.

Disadvantages:

- COCOMO model ignores requirements and all documentation.
- It ignores **customer skills**, cooperation, knowledge and other parameters.
- It oversimplifies the impact of safety/security aspects.
- It ignores hardware issues
- It ignores personnel turnover levels
- It is dependent on the amount of time spent in each phase

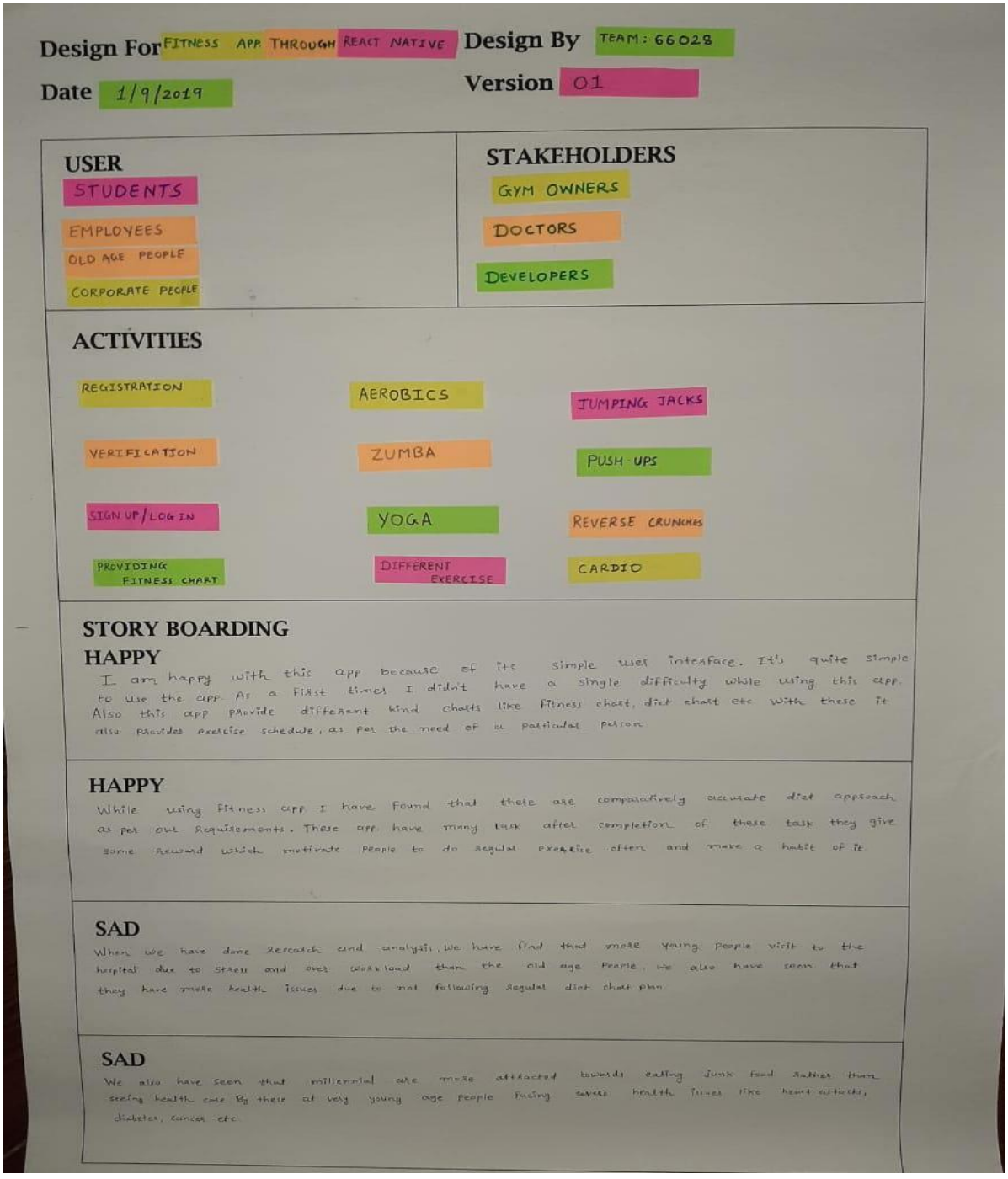
2.4 CANVAS

2.4.1 AEIOU CANVAS



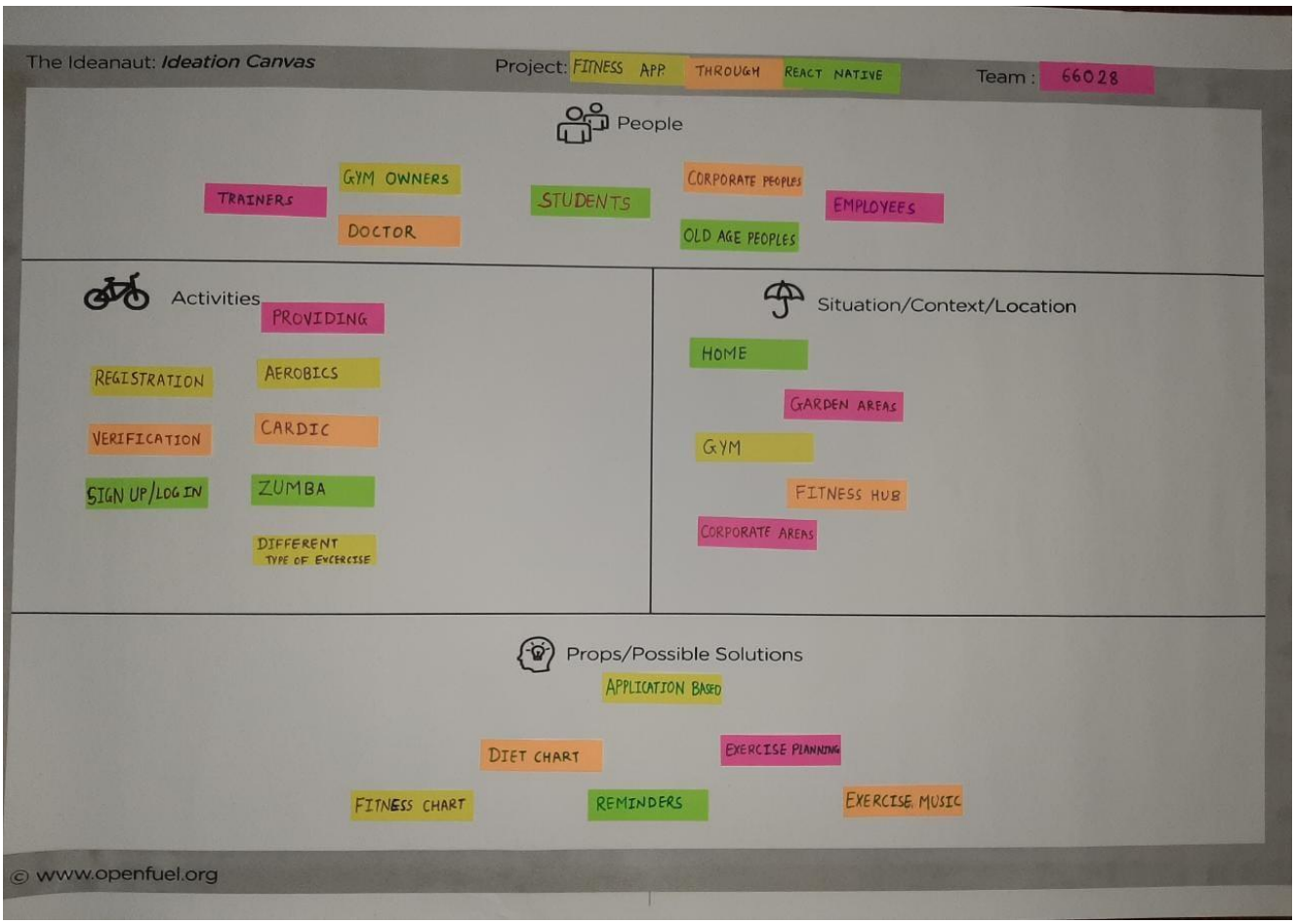
[Fig 2.4.1 AEIOU CANVAS]

2.4.2 EMPATHY CANVAS



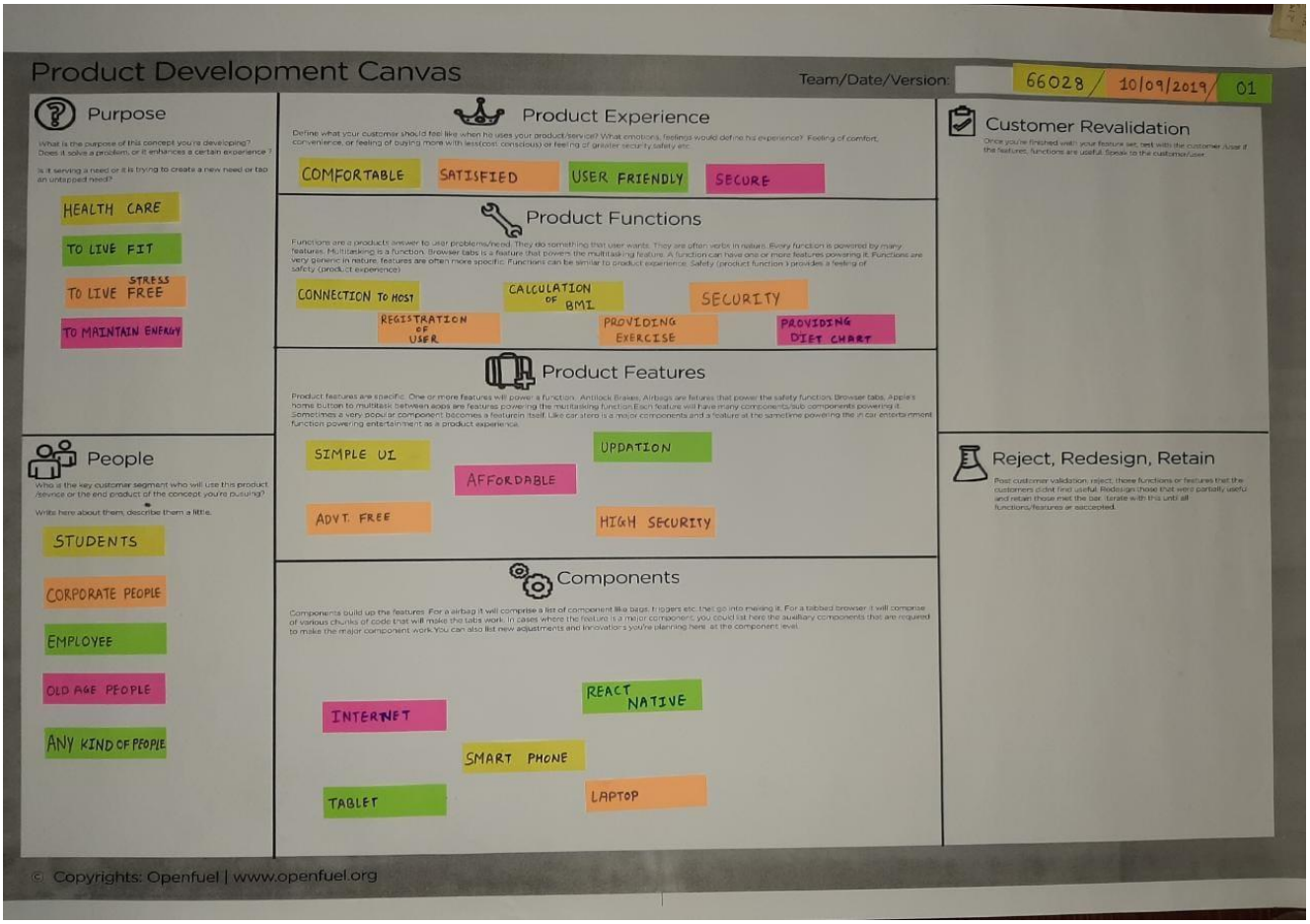
[Fig 2.4.2 EMPATHY CANVAS]

2.4.3 IDEATION CANVAS



[Fig 2.4.3 IDEATION CANVAS]

2.4.4 PRODUCT & DEVELOPMENT CANVAS



[Fig 2.4.4 PRODUCT & DEVELOPMENT CANVAS]

CHAPTER 3
SYSTEM REQUIREMENT STUDY

CHAPTER 3

SYSTEM REQUIREMENT STUDY

3.1 SYSTEMREQIREMENT STUDY

3.1.1 User Characteristic:

- It describes the type of users which deals with the applications. Basically, this application has two types of users as given below:
    1. Administrator
    2. User

1. Administrator

- Responsibility of the administrator is to manage the application database and update the data in database regularly. For e.g. manage the data of user verification.

2. User

- Once the application being tasted and being developed the after the customer will use this application. User is the main reason for which this application is being built. The end user can easily interact with the application and with its features.

3.2 Hardware and Software Characteristics:

Hardware Requirements:

I. Minimum 2.27Ghz processor
II. RAM: 2 GB
III. Android Version : above Lollipop(5.0)

Software Requirements:

I. React-Native Framework
II. Java-script Properties
III. Internet connection (not compulsory)

Additional requirement for user:

I. A working platform

II. Proper details

Hardware requirement for Data Science:

I. Minimum 2.0Ghz processor

II. Minimum 2GB RAM

III. 100GB free space in HDD

Software Requirement for Data Science:

I. React-Native Framework

II. Java-script Properties

III. Android Studio

3.3 Constraints:

3.3.1 Regulatory Policies:

- Regulatory policies or mandates, limits the discretion of individual and agencies, or otherwise compel certain types of behavior.
- These policies are generally thought to be the best applied when good behavior can easily be defined and bad behavior can be easily regulated and punished through fine and sanctions.

3.3.2 Hardware Limitation:

- The smooth functionality of the application depends on the speed of hardware and their specification.
- It is always advisable to be updated as far as hardware is concerned. The hardware limitation occurs if the user is still using very low MHz processor and RAM or less than 128mb.
- This will generally reduce the portability and also the use will waste a lot of useful time and resources.

CHAPTER 4
SYSTEM ANALYSIS

CHAPTER 4
SYSTEM ANALYSIS

4.1 Study of Current System

- This is not any fitness application that requires network connection all the time for its functioning, once the data are entered while it is connected in the network this application can be opened or run in offline mode.

4.2 Requirement of this System

- We are developing fitness application with the help of react-native frame and java-script properties. By using this application one's can get all the requirement for his/her exercise, diet plan etc. According to his/her weight, age and entered data.
- Currently this type of application is not available in the widely developed world. We are developing our application on this creative concept.

4.2.1 Functional Requirements:

A. User

All users of the application shall have the ability to create account which is used to stored data and tie user action to a user alias. User registration and login shall be mandatory.

B. Create Account

The system provide the user with an easy to use GUI to facilitate their creating an account. The system shall ask for email address and password. The system shall notify user if incorrect character are used in email and password field. The system should notify the user if their email address is already registered. The system should notify the user if any required filed in left empty. The system should not allow user to create weak and unsecure password. The system should prevent the user from completing registration if the terms and services has not been agreed to.

C. Login

The system should provide a user friendly GUI to allow the user to login when the application launches. The system should prompt the user for their email address and password.

D. View Existing Program

The system should provide intuitive and user friendly navigation to allow user to locate current list exercise available in application. The system should display full details of an exercise routine once one is selected by the user. The system should

allow quick and easy navigation between different routines in the list. The system should be able the display the diet plan to the user submitted details. The system should allow quick and easy navigation between different routines in the list.

E. Save a Program to Personal List

The system should provide a button to save a program from the public list of user submitted programs to their own personal lists.

F. Rate a Program

The system should provide a button to rate. Program can be rated up or down based on the level of success the user has with them.

G. Edit Personal Program

The system should provide an intuitive and user friendly UI to allow the user to view and manage their personal list of workout programs.

H. Send Message

The system shall provide an interface for sending message between users. Message should be sent in real time and have no delays.

4.2.2 Non Functional Requirements:

- ### Portability

The application will be written using react-native framework using JavaScript properties, the application will run natively in any android device whose version is above lollipop (5.0). This includes the device like any android mobile, tablet etc.

- ### Reliability

The system should be extremely reliable and have an approximate up time of 99.99%. In the event of a crash or any other error, the system should inform the user of any problem and gracefully terminate.

- ### Ease of Use

The application should be user friendly and intuitive to use. GUIs should make their function clear and navigation around the application should be straight forward. Users should be comfortable using the application after 20 minutes of use.

- ### Speed

The application should open and be ready to use within 10 seconds of being selected. The UI should be quick and smooth with no delay between button presses and screen reaction. All database reads/writes should take no longer than 5

seconds. If the database encounters any error, a user friendly waring should be displayed to the user.

- ## Size

  The size of the software in relation to storage media should be no larger than 250mb.

- ## Privacy

  All data retained by the system will be stored in accordance with the Data Protection Act 1988 and the Data Protection (Amendment) Act 2003.

- ## System Stress Management

  The system will initially handle up to 100 simultaneous users. It can handle only 100 simultaneous reads/writes at any one time on the basic plan.

User Requirements:

A. Product Perspective

The application will be written in Android Studio using the react-native framework using javas-script properties. The application will run in android platform. It will provide users the ability to fid or submit workout routine, diet plan inside the application. Users will be able to register an account and login using their details to take part in online community and gain access to user submitted content. This application can be used in offline mode once all the required information of the user in entered.

B. Product Function

- Users will be able to register to the application with their personal details. Initially will be via email and password but future changes could allow for Google and Facebook account login facility.
- Registered user will be able to sign in and access the workout routine and diet plan from the application.
- Registered user can be able to choose their required workout routine and diet plan according to the entered data.
- Registered users will be able to be connected with the online community and ask for help whenever required.
- Registered users of the application will be able to participate in group discussions within the application or opt to communicate one on one with another single user. This allows users to discuss things as part of a group and meet new people and also allows them to discuss things privately or ask the author of a program a question should they need clarification on

any aspect of their program.

4.3 User Characteristics

i. Unregistered User

Unregistered users will not be able to gain access to the application or use any of its features. The whole focus of the application is on user submitted content and social interaction, none of this possible without an account alias to tier a user to. Additionally, the service offered by the application will be behind a ToS (Terms of Service) agreement, this model was chosen as users will be following programs submitted by users completely at their own risk. No liability is accepted by the developer of the application or any parties in connection with the developer.

ii. Registered User

Registered users will gain full, non admin access to the application. Once a user registers their details and agrees to the ToS they will be able to view workout routines, rate workout routines, save routines to their personal list for edits, chat with other users and use the gyms nearby feature and save diet plan according to their submitted details.

iii. Administrator

The administrator will maintain the data in the application behind the scenes. The administrator will deal with managing user accounts if any bans are levied against a user. They will also manage workout program data based on user feedback. If a program receives enough negative feedback it will need to be removed from the application by the administrator. The administrator will also carry out day to day housekeeping within the application and ensure the quality of the application content.

4.4 Feasibility Study

- Feasibility is the measure of how beneficial the development of information system will be to an organization.
- The feasibility analysis is categorized under four different types
    (1) Operational Feasibility
    (2) Technical Feasibility
    (3) Schedule Feasibility
    (4) Economic Feasibility

✓ Operational Feasibility

The system is to be developed for any user who want to use it. We want our system user friendly and easy to use. The administration also may be non-

terminal, so the user interface will be designed in such a way that it gets comfortable for non- terminal person to operate easily.

- ✓ **Technical Feasibility**

  It is a partially measurement of specific technical solution and the availability of technical resort and expertise. The analyst must find out whether the current technical resources, which are available in the system is capable of handling the job. If not, then the analyst with the help of developer should confirm whether the technology is available and capable or not.

- ✓ **Schedule Feasibility**

  Schedule feasibility corresponds to whether sufficient time is available to complete the project.

- ✓ **Economic Feasibility**

  Economic feasibility is a measure of cost effective of a project or solution. For declaring that the system is economically feasible, the benefit from the project should exceed or at least to the equal to the cost of development.

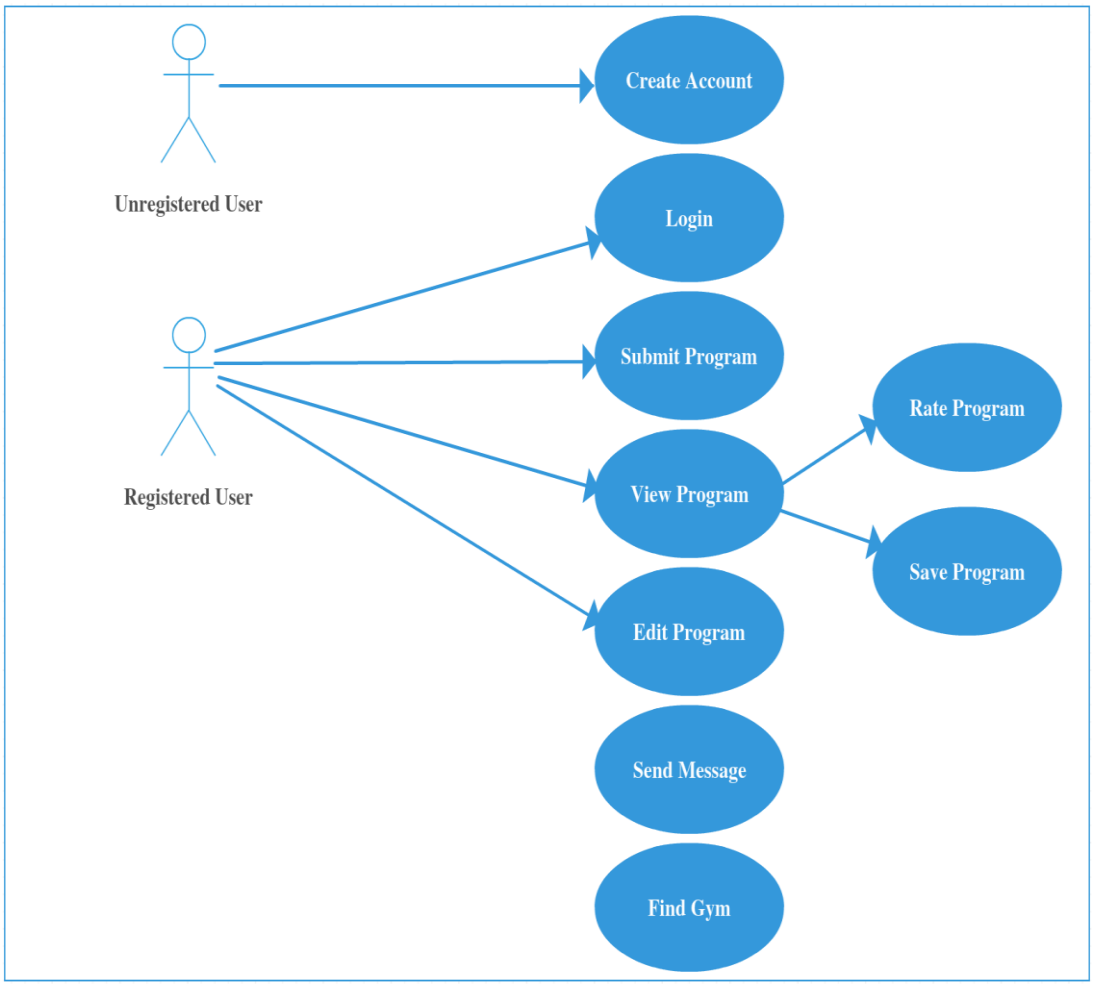
4.5 Requirement Validation:

- Requirement validation examines this specification to ensure that all the system requirement have been stated unambiguously.
- These inconsistent, error have been detected and corrected and the work product confirm to the standard.
- Source of the requirement are identified, final statement of requirement has been examined by original source.
- Requirement related to main requirement are found.
- All source if requirement are converted to get maximum requirement.
- Requirement are clearly stated and are not misinterpreted.
- All the methods of finding requirement is applied.

4.6 Function of the System:

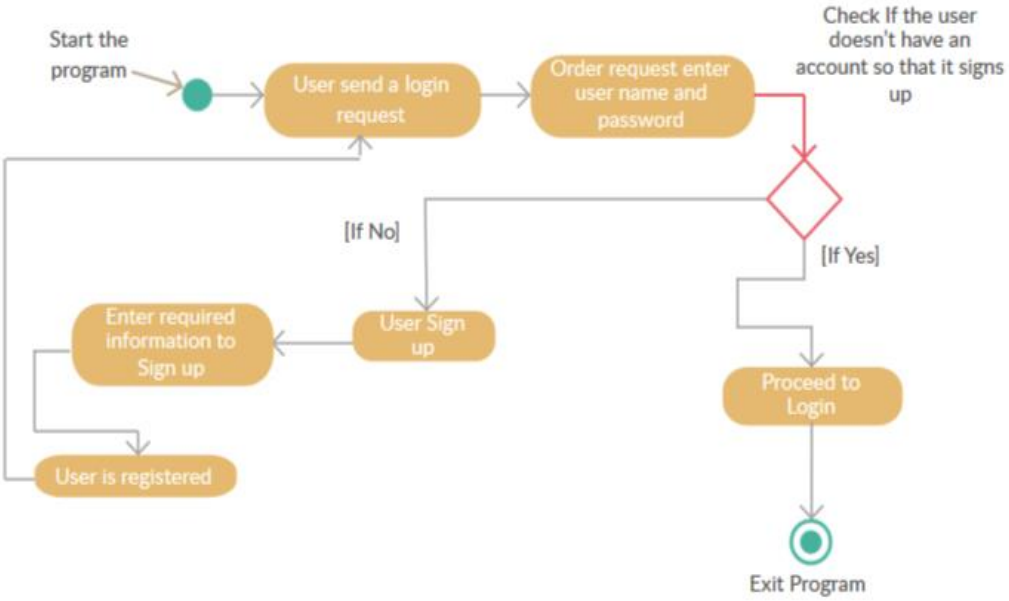
4.6.1 Use-Case:

In software and systems engineering, a **use case** is a list of steps, typically defining interactions between actor and a system, to achieve a goal. The actor can be a human, an external system, or time. In systems engineering, use cases are used at a higher level than within software engineering, often representing missions or stakeholder goals. The detailed requirements may then be captured in Systems Modelling Language or as contractual statements. As an important requirement technique, use cases have been widely used in modern software engineering over the last two decades. Use case driven development is a key characteristic of process models and frameworks. With its iterative and evolutionary nature, use case is also a good fit for agile development.



[Fig 4.6.1 Use-Case Diagram]

4.6.2 Activity Diagram



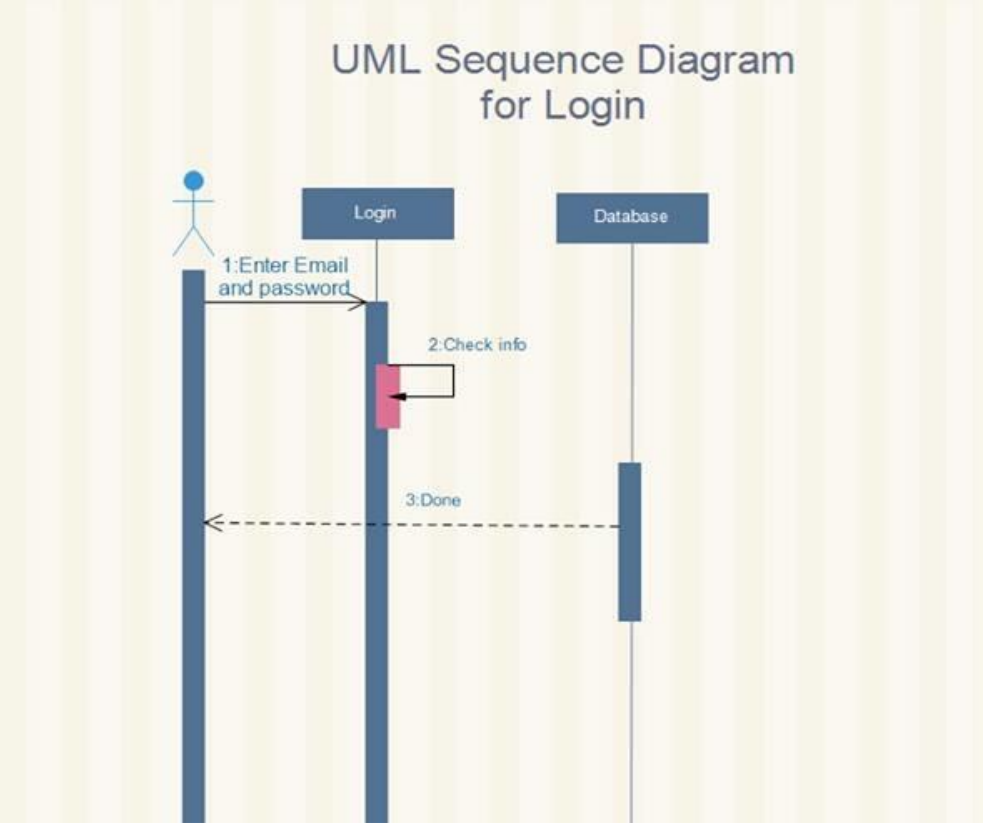
[Fig 4.6.2 Activity Diagram]

4.6.3 Sequence Diagram:

The well-known Message Sequence Chart technique has been incorporated into the Unified Modelling Language (UML) diagram under the name of **Sequence Diagram**. A sequence diagram shows, as parallel vertical lines, different processes or objects that live simultaneously, and, as horizontal arrows, the messages exchanged between them, in the order in which they occur. This allows the specification of simple runtime scenarios in a graphical manner.

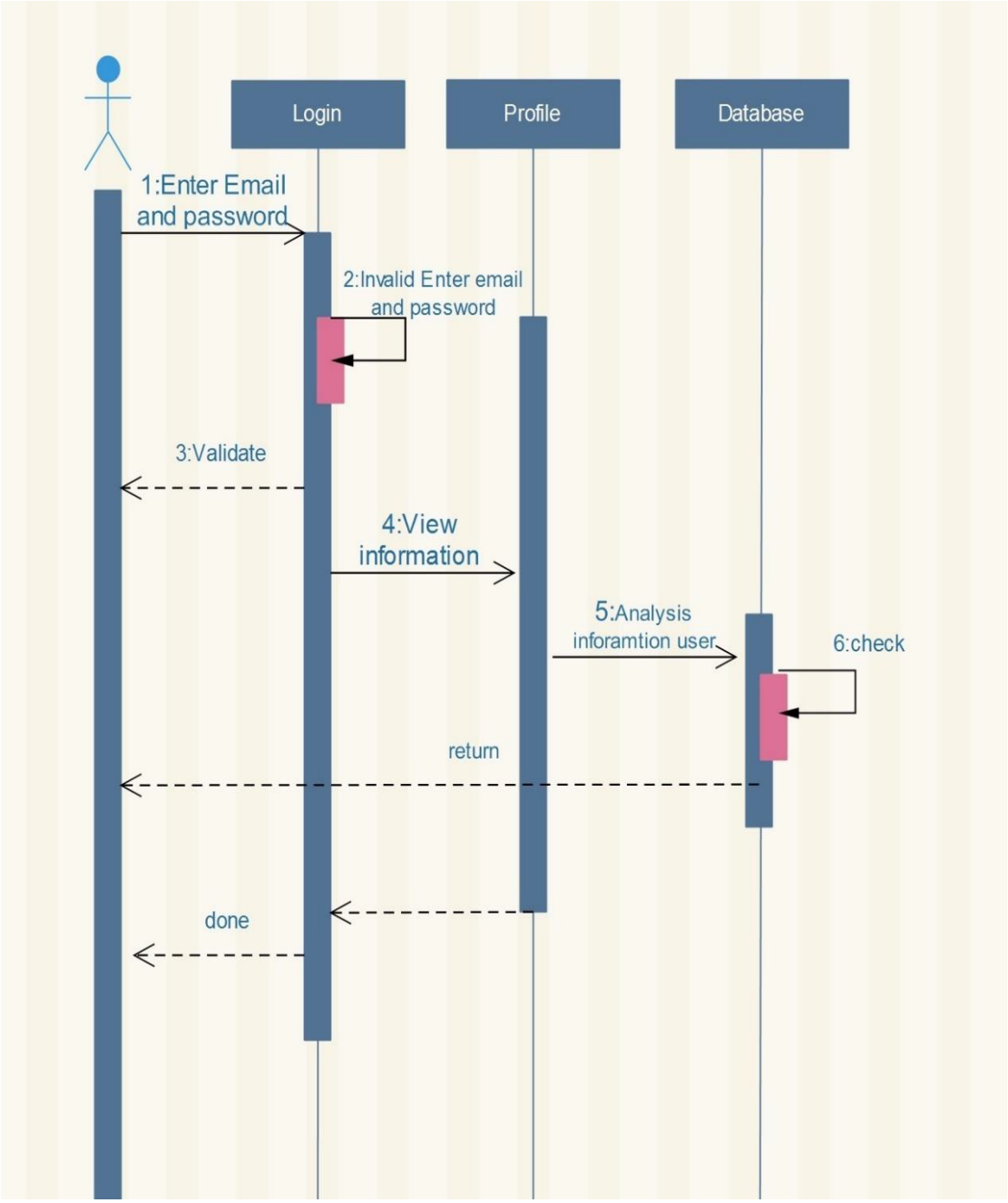
The well-known Message Sequence Chart technique has been incorporated into the Unified Modelling Language (UML) diagram under the name of **Sequence Diagram**. A sequence diagram shows, as parallel vertical lines, different processes or objects that live simultaneously, and, as horizontal arrows, the messages exchanged between them, in the order in which they occur. This allows the specification of simple runtime scenarios in a graphical manner.

1. Login



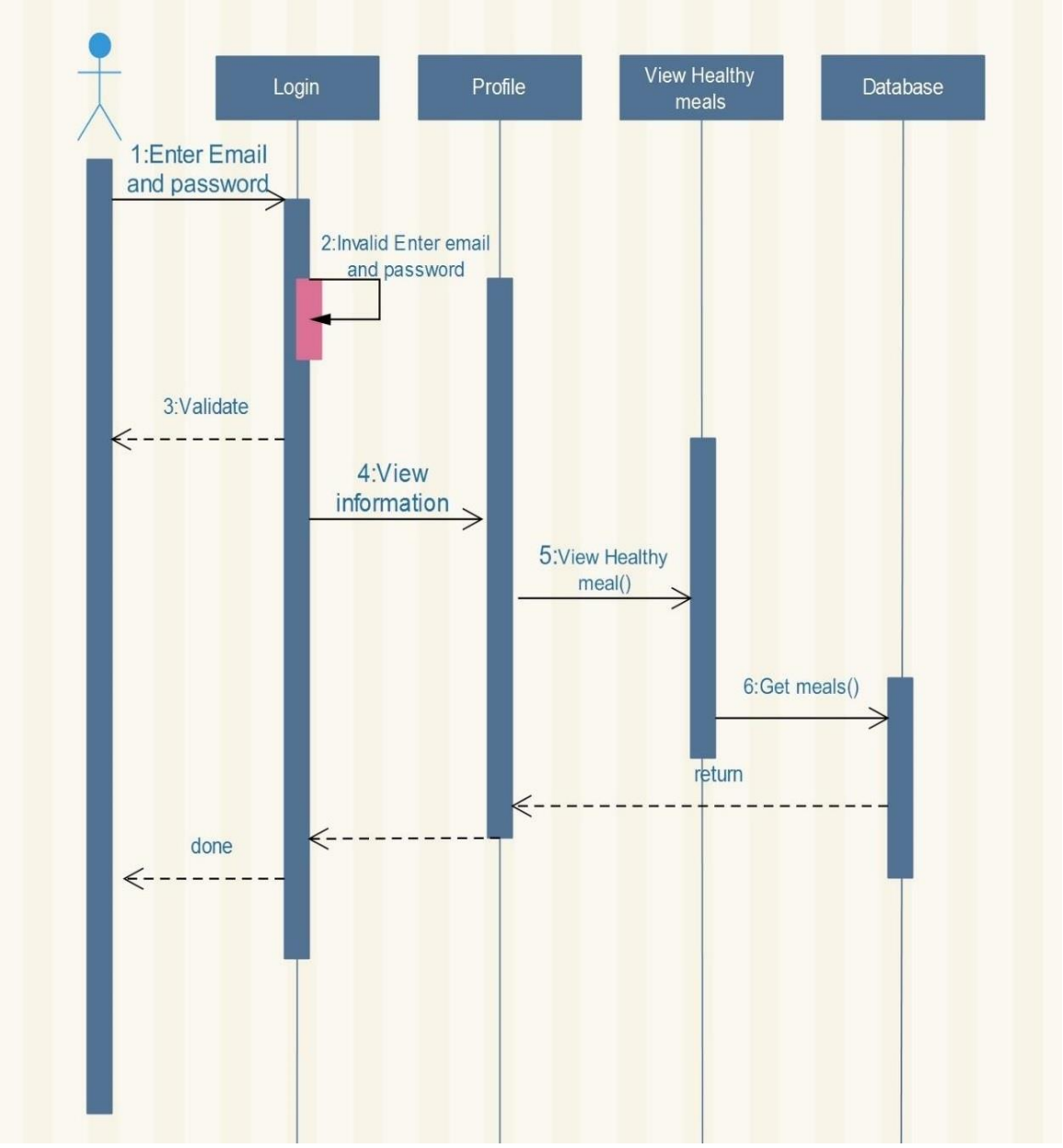
[Fig 4.6.3(1) Login]

2. Analysis information



[Fig 4.6.3(2) Analysis Information]

3. Healthy Meal

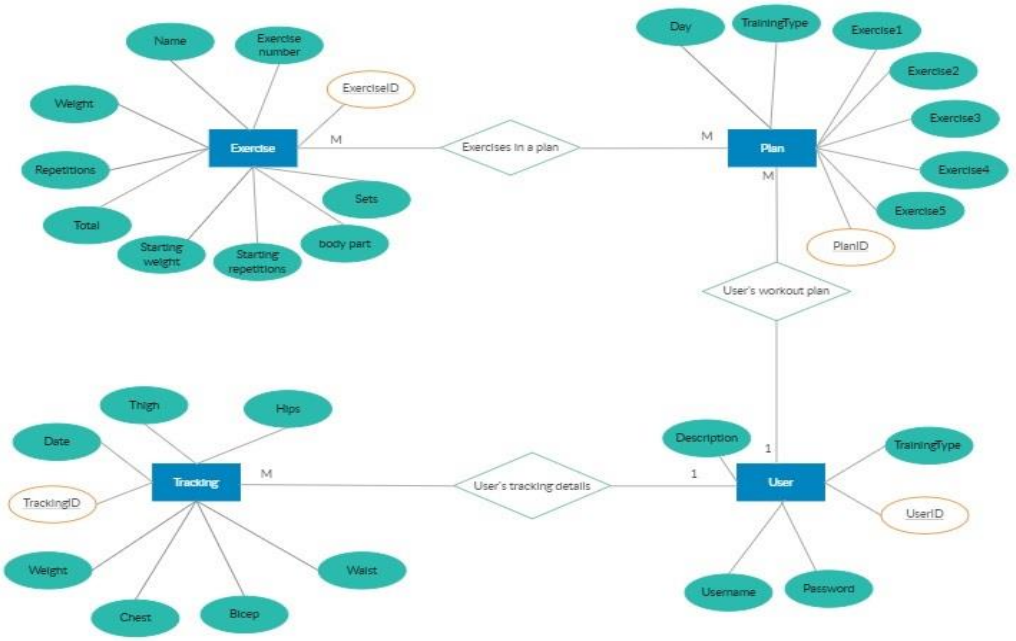


[Fig 4.6.3(c) Healthy Meal]

4.7 Data Modelling

4.7.1 ER Diagram

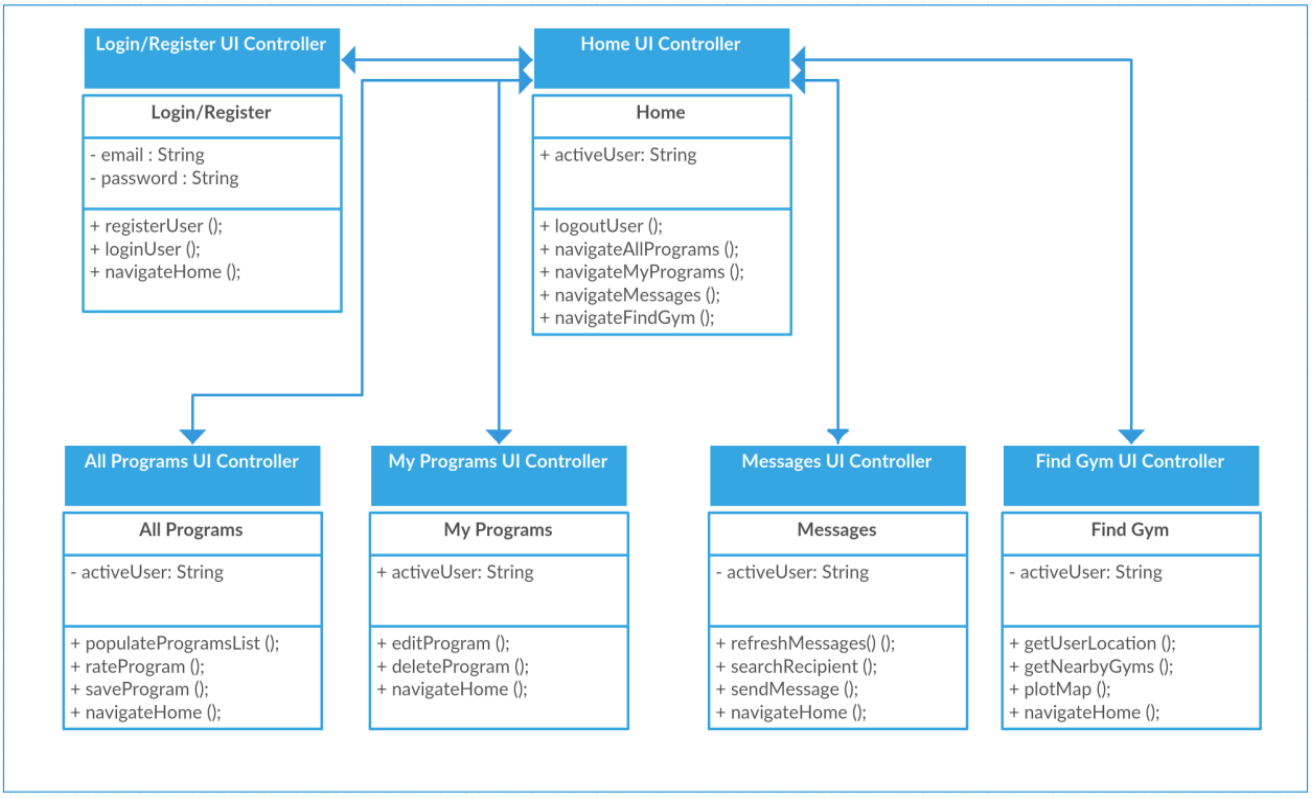
In software engineering, an **entity–relationship model** (**ER model**) is a data model for describing the data or information aspects of a business domain or its process requirements, in an abstract way that lends itself to ultimately being implemented in a database such as a relational database. The main components of ER models are entities and the relationships that can exist among them, and databases. An entity-relationship model is a systematic way of describing and defining a business process. The process is modelled as components (entities) that are linked with each other by relationships that express the dependencies and requirements between them, such as: one building may be divided into zero or more apartments, but one apartment can only be located in one building. Entities may have various properties (attributes) that characterize them. Diagrams created to represent these entities, attributes, and relationships graphically are called entity–relationship diagrams.



[Fig 4.7.1 ER Diagram]

4.7.2 Class Diagram

In software engineering, a **class diagram** in the Unified Modelling Language (UML) is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among objects. The class diagram is the main building block of object oriented modelling. It is used both for general conceptual modelling of the systematic of the application, and for detailed modelling translating the models into programming code. Class diagrams can also be used for data modelling. The classes in a class diagram represent both the main objects, interactions in the application and the classes to be programmed.



[Fig 4.7.2 Class Diagram]

CHAPTER 5
SYSTEM DESIGN

CHAPTER 5

SYSTEM DESIGN

Systems design is the process of defining the architecture, components, modules, interfaces, and data for a system to satisfy specified requirements. The System Design Description report provides summary or detailed information about a system design represented by a model. Systems design is therefore the process of defining and developing systems to satisfy specified requirements of the user.

5.1 SYSTEM REQUIREMENT

Since we have mobile application, we need different hardware and software requirement for each.

- **Operational Requirement**

Software requirement which will be used during the development process:

1. Operating System : any computer OS and can be operated in any android mobile whose version is above lollipop(5.0)
2. Android Operating System
3. React-Native Frame with JavaScript properties.

- **Hardware Requirement**

Hardware requirement which will be used during the development process:

1. Mobile with Android operating system

- **Development Requirement**

Tools which will be required while developing application:

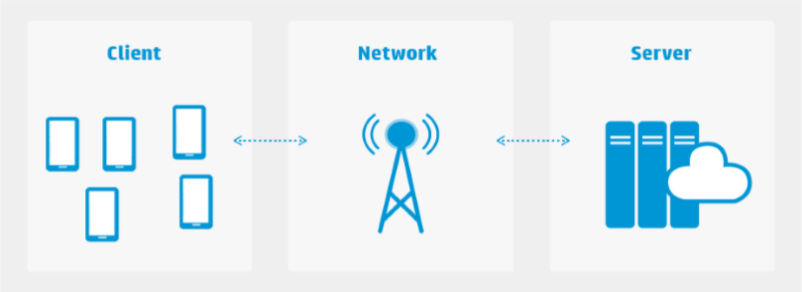
1. Android Studio
2. VS Code
3. Node JS

5.2 Technology Choices.

At first the client will have to install an Android application so they can access to Share Experience app, for the mobile application android provides a variety of pre-build UI components such as structured layout objects and UI controls that allow you to build the graphical user interface for your app. Android also provides other UI modules for special interfaces such as dialogs, notifications, and menus. Also we will use XML to build an interactive interface for the users, with a compatible version for the mobile users too.

5.3 System Architecture

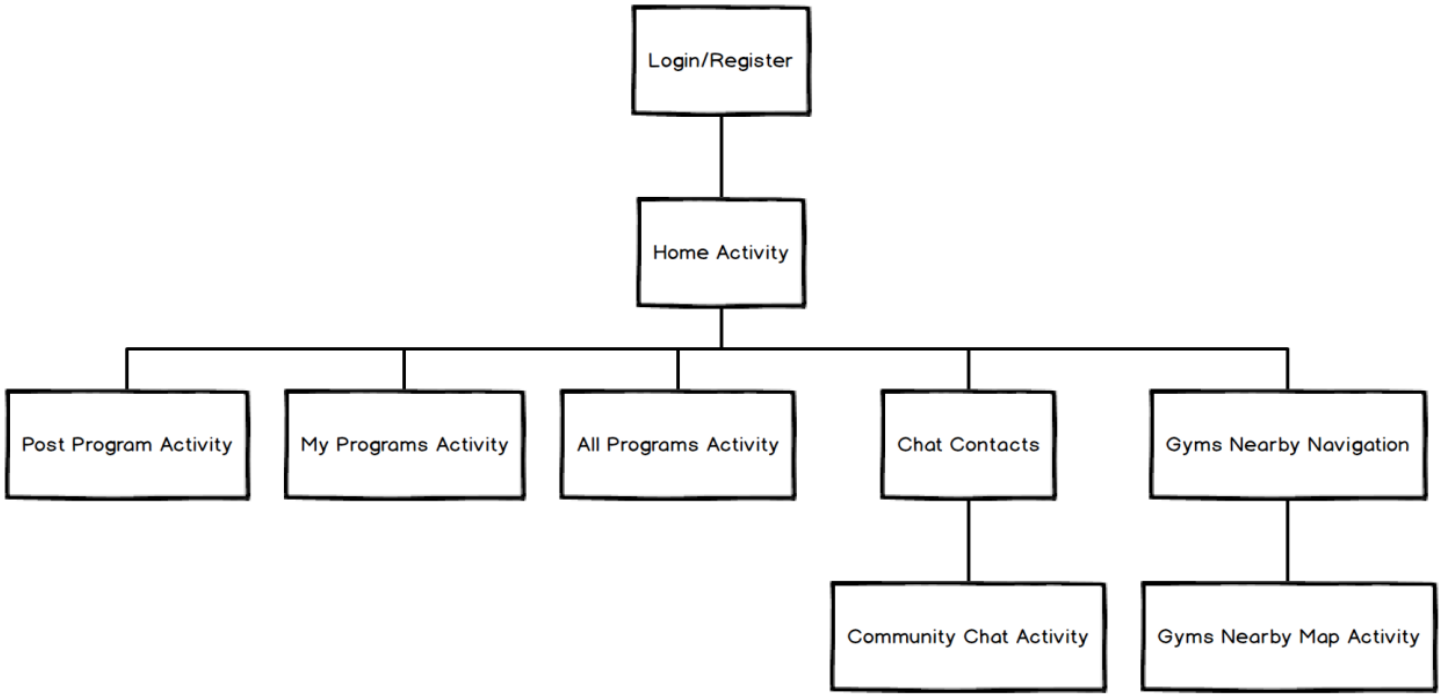
A system architecture is the conceptual model that defines the structure, behaviour, and more views of a system. An architecture description is a formal description and representation of a system, organized in a way that supports reasoning about the structures and behaviours of the system.
A system architecture can comprise of system components, the externally visible properties of those components, and the relationships (e.g. the behavior) between them. It can provide a plan from which products can be procured, and systems developed, that will work together to implement the overall system. We use three tire architecture for mobile development, we choose this architecture cause easy to modify without affecting other modules, fast communication and Performance will be good in three tier architecture

Description	The basic idea is that the core of the application would reside in the network layer. The interface to the network and server layers would stay constant, the client layer consists of things like, on Android activities. The server layer would abstract features of the device like database access, network access etc. for the network and client layers. It would not only be a wrapper on the device specific API but it could also simplify access to the feature so the network layer can concentrate on the application logic.
Design Diagram	
Components	Client Sever- user need register first in order to access the data stored.
Relationship	The client and server are connected over network The client make request The server provides the service
Pros	Once the user is registered he/she can use the application in offline mode. They can save the list of workout in their playlist if they like.
Cons	Its cons is that all files are in one location, which means that if server goes down or system crashes, we will lose everything simultaneously.

5.4 Deployment Diagram

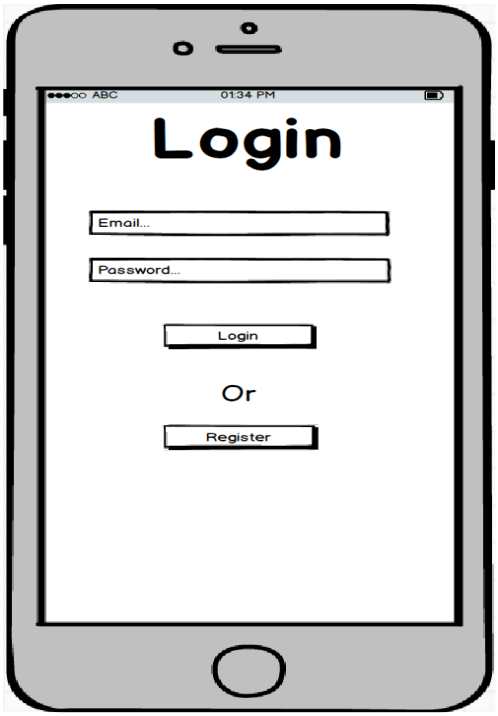
Deployment diagram would show what hardware components ("nodes") exist, what software components ("artifacts") run on each node, and how the different pieces are connected Device nodes are physically computing resources with processing memory and services to execute software, such as typical computer or mobile phones.

5.4.1 Hierarchical Diagram



[Fig 5.4.1 Hierarchical Diagram]

5.4.2 Login or Register



[Fig 5.4.2 Login or Register]

CHAPTER 6 IMPLEMENATION PLANNING AND DETAILS

CHAPTER 6
IMPLEMENTATION PLANNIG AND DETAILS

6.1. Implementation Environment:

6.1.1 Developing for the Android Platform

In developing our Android application, we utilized the Eclipse integrated development environment bundled with the Android Development Tools plugin for approximately three quarters of the development time. About halfway through our project, Google released Android Studio, an IDE created specifically for designing Android applications.

6.1.2 The Development Process

The development process consisted of weekly meetings, in which the entire group met to come up with goals for the next week and review the progress of current programming tasks that were assigned to each team member. Our goals were often driven by the obstacles we had faced during the previous week and were prioritized by how severely they impacted further development of the app.

6.1.3 Development Overview

At the beginning, much of the work was individual, as we were all learning the syntax and architecture of Android development. As time progressed, we split into two main teams; one that worked with the development of the app itself and one that worked with the development of the character that would personify the information collected by the app. Later this was broken down further into more individual tasks within each team, which typically exemplified each team member's individual strengths.

6.1.4 Design & Coding

Relatively early in the development process, we came to the decision to separate each data into individual tabs within the app. After researching several different methods for implementing this type of multi-page app design, we came to the decision to use the View Pager layout manager, an Android class that is "most often used in conjunction with Fragment, which is a convenient way to supply and manage the lifecycle of each page. In addition, the tabs share a single Android activity, which is the Android class that handles all user interaction with the app. The main activity of the application manages the tab layout and tells the application which tab view to display when a user selects a certain tab. The purpose of using fragments is to eliminate the need to create a new activity every time the user switches between tabs, thus increasing both the temporal and spatial efficiency of the application. Structurally, each fragment is comprised of two components. The first is an XML file that defines the visual layout of the fragment. The second is the logic of the fragment, which contains various functions defining certain actions to take at different times. Each time a certain fragment is switched to or away from, these functions are called in a particular order, all of which make up what is called the fragment's "lifecycle".

Different modules specified in the design document are coded in the Coding phase according to the module specification. The main goal of the coding phase is to code from the design document prepared after the design phase through a high-level language and then to unit test this code. A coding standard gives a uniform appearance to the codes written by different engineers. It improves readability, and maintainability of the code and it reduces complexity also. It helps in code reuse and helps to detect error easily. It promotes sound programming practices and increases efficiency of the programmers. Good software development organizations want their programmers to maintain to some well-defined and standard style of coding called coding standards. They usually make their own coding standards and guidelines depending on what suits their organization best and based on the types of software they develop. It is very important for the programmers to maintain the coding standards otherwise the code will be rejected during code review.

```
import { AppRegistry } from 'react-native';
import App from './App';
import dasbord from './page/dasbord'

import YogaList from './page/YogaList';
import yogadetails from './page/yogadetails';
import Suryanamaskar from './page/Suryanamaskar';
import weightlossyoga from './page/weightlossyoga';
import prePregnancyyoga from './page/prePregnancyyoga';
import postPregnancyyoga from './page/postPregnancyyoga';


import WorkOutList from './page/WorkOutList';
import basicworkout from './page/basicworkout';
import wormupworkout from './page/wormupworkout';
import weightlossWorkout from './page/weightlossWorkout';
import bicepworkout from './page/bicepworkout';

import DaietList from './page/DaietList';
import daietdetails from './page/daietdetails';
import weightlossDiet from './page/weightlossDiet';
import HealthyShakes from './page/HealthyShakes';


import meditation from './page/meditation';
import bmi from './page/bmi';
import trainer from './page/trainer';


AppRegistry.registerComponent('FitnFab', () => App);
```

[Fig 6.1.4 (I) Module Import]

```
package android.app;

import ...

public class Application extends ContextWrapper implements ComponentCallbacks2 {
    public Application() {
        super((Context)null);
        throw new RuntimeException("Stub!");
    }

    public void onCreate() { throw new RuntimeException("Stub!"); }

    public void onTerminate() { throw new RuntimeException("Stub!"); }

    public void onConfigurationChanged(Configuration newConfig) {
        throw new RuntimeException("Stub!");
    }

    public void onLowMemory() { throw new RuntimeException("Stub!"); }

    public void onTrimMemory(int level) { throw new RuntimeException("Stub!"); }

    public void registerComponentCallbacks(ComponentCallbacks callback) {
        throw new RuntimeException("Stub!");
    }

    public void unregisterComponentCallbacks(ComponentCallbacks callback) {
        throw new RuntimeException("Stub!");
    }
```

[Fig 6.1.4(II) Application Class]

```
package android.app.admin;

import android.os.Parcel;
import android.os.Parcelable;
import android.os.Parcelable.Creator;
import java.net.InetAddress;

public final class ConnectEvent extends NetworkEvent implements Parcelable {
    public static final Creator<ConnectEvent> CREATOR = null;

    ConnectEvent(Parcel in) { throw new RuntimeException("Stub!"); }

    public InetAddress getInetAddress() { throw new RuntimeException("Stub!"); }

    public int getPort() { throw new RuntimeException("Stub!"); }

    public String toString() { throw new RuntimeException("Stub!"); }

    public int describeContents() { throw new RuntimeException("Stub!"); }

    public void writeToParcel(Parcel out, int flags) { throw new RuntimeException("Stub!"); }
}
```

[Fig 6.1.4(III) Event Connect]

6.2 Hardware & Software Requirement

- Minimum 2.27Ghz processor
- RAM : Minimum 2 GB
- Android Version: Above Jelly Bean (4.2.2)
- React Native Framework
- JavaScript properties
- Internet Connection(Not compulsory)

6.2.1 Hardware Requirement for Data Science

- Minimum 2.0Ghz Processor
- Minimum 4GB RAM
- 100GB free space in Hdd

6.2.1 Software Requirement for Data Science

- React Native Framework
- JavaScript properties
- Android Studio

6.3 Program/Modules Specification

- A precise statement of the effects that a software module is required to achieve.
- It can be employed both by the implementer of the module, since it gives a definitive statement of the requirements that are imposed on the module, and by users of the module, since it gives a precise statement of what the module provides.
- A good module specification makes no commitment as to how the module's effects are achieved.
- A variety of techniques have been developed for module specification.
- A *functional specification* identifies the operations that the module makes available and provides an individual specification for each operation, typically in the form of an input-output specification describing the mapping that the operation provides from a set of input values to a set of output values.
- In the typical case where a module has local data, a simple functional specification will need to refer to this local data when specifying each individual operation.

6.4 Security Features

- Our application supports the user login in features but we haven't included the user login features.
- We stand to serve and provide privacy to user this is the reason why login is not compulsory.
- We use the latest version on Android Studio Plugins and React Native feature which provides the updates security.
- React Native provides the security like:
    a. Securing app to server connection
    b. Securing local data
    c. Advance integrity check
- Android Studio provides the security like:
    a. The Android Application Sandbox, which isolates your app data and code execution from other apps.
    b. An application framework with robust implementations of common security functionality such as cryptography, permissions, and secure IPC.
    c. An encrypted file system that can be enabled to protect data on lost or stolen devices.
    d. User-granted permissions to restrict access to system features and user data.
    e. Application-defined permissions to control application data on a per-app basis.

CHAPTER 7
TESTING

CHAPTER 7
TESTING

7.1 Testing

Initial testing was done on an Android simulator which ran on our computers. This aided in development because we were able to immediately load changes and view how they affected the app. This also eliminated the immediate need for a physical Android device and we were able to begin developing and testing our app immediately.

7.2 Testing Result

Once our application was completed d and polished, we sent it to 10-15 Android phone users who assisted us in testing. Before sending them the application, we sent them a pre-testing survey that assessed general information about the user. Once all users had completed the initial survey, we sent them the Android Application Package (APK) used for installing application on their phones. In addition, we sent them another survey, used for assessing any bugs that might be present within the app, as well as gathering data regarding the effectiveness of the application's sleep and step tracking algorithms. We received 14 responses from the pre-testing survey and 10 responses from the post-testing survey. Please see Appendix for response data from the pre-testing survey, and for response data from the post-testing survey.

CHAPTER 8
SCREEN SHOTS AND USER MANUAL

CHAPTER 8

SCREEN SHOTS AND USER MANUAL

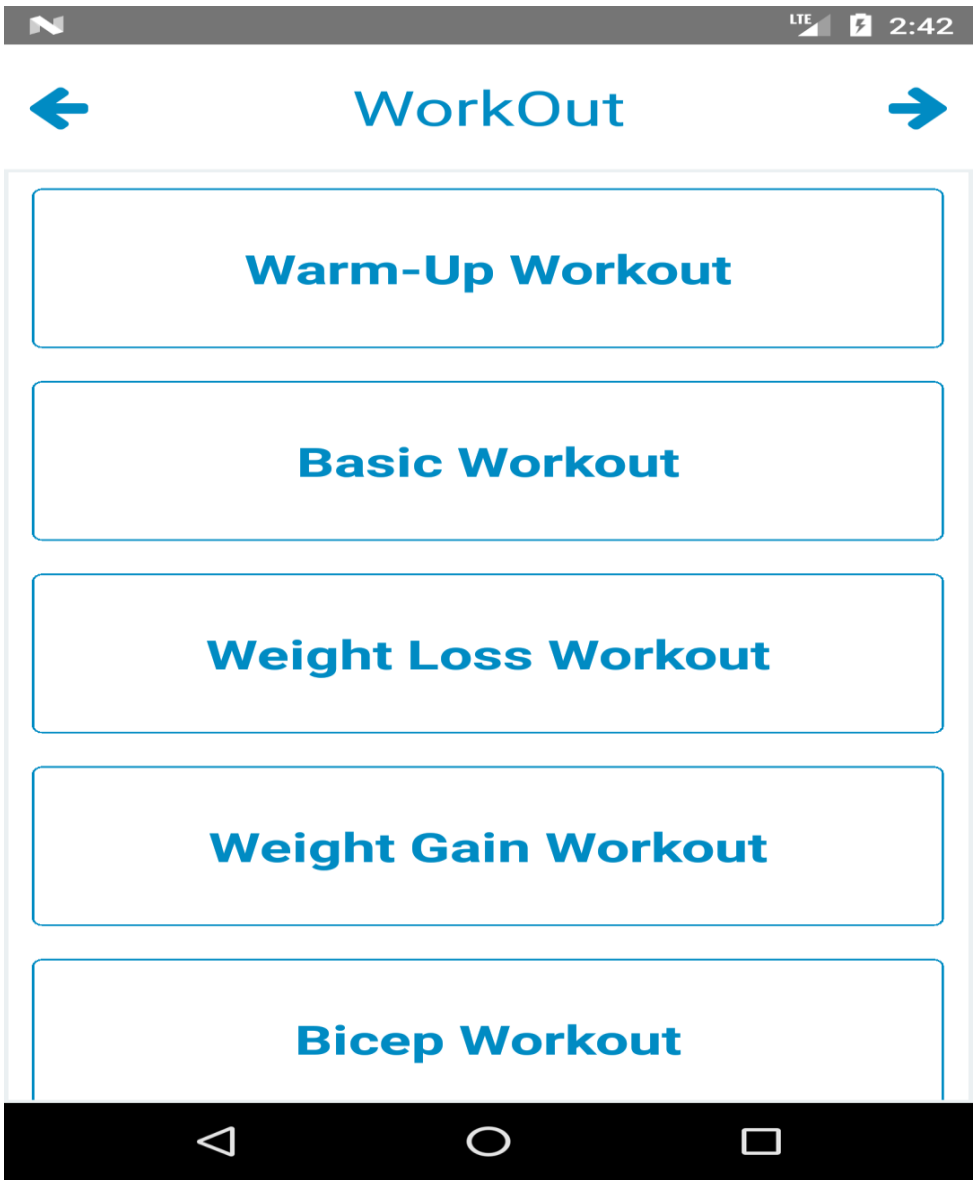
8.1 Front end

[Fig 8.1(I) Home1]

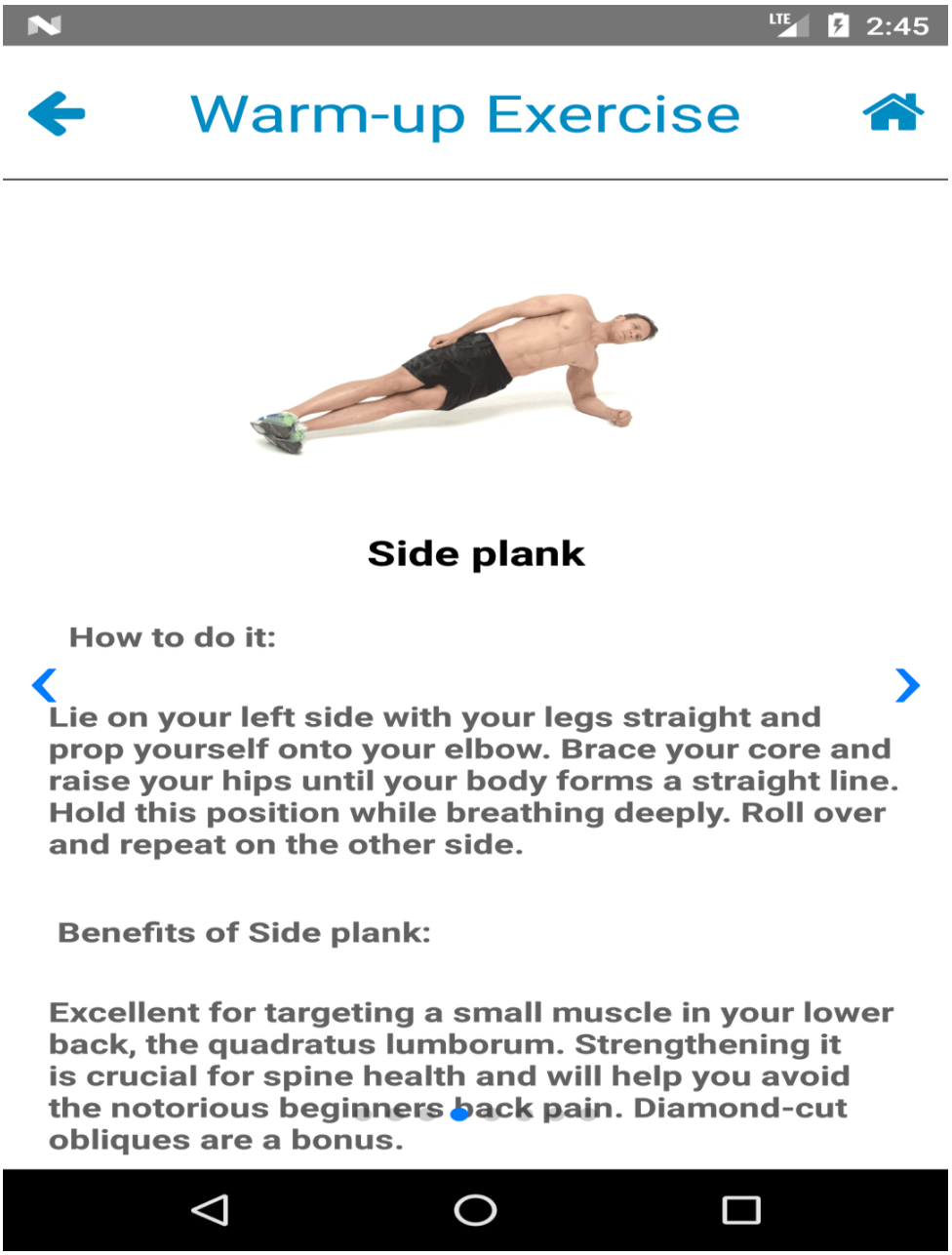
[Fig 8.1(II) Home2]

8.2 Modules

8.2.1 Workout

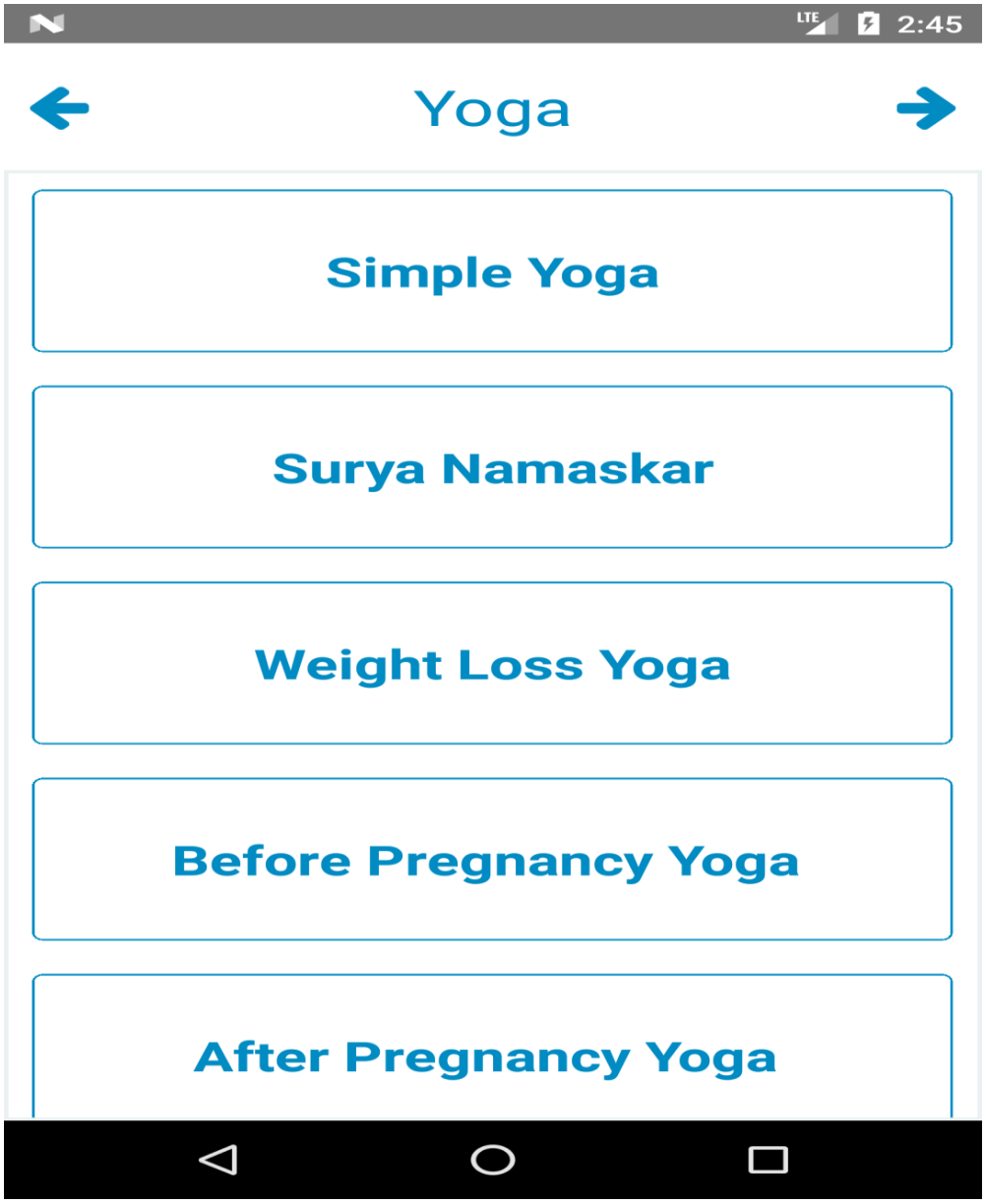


[Fig 8.2.1 (I) Workout]

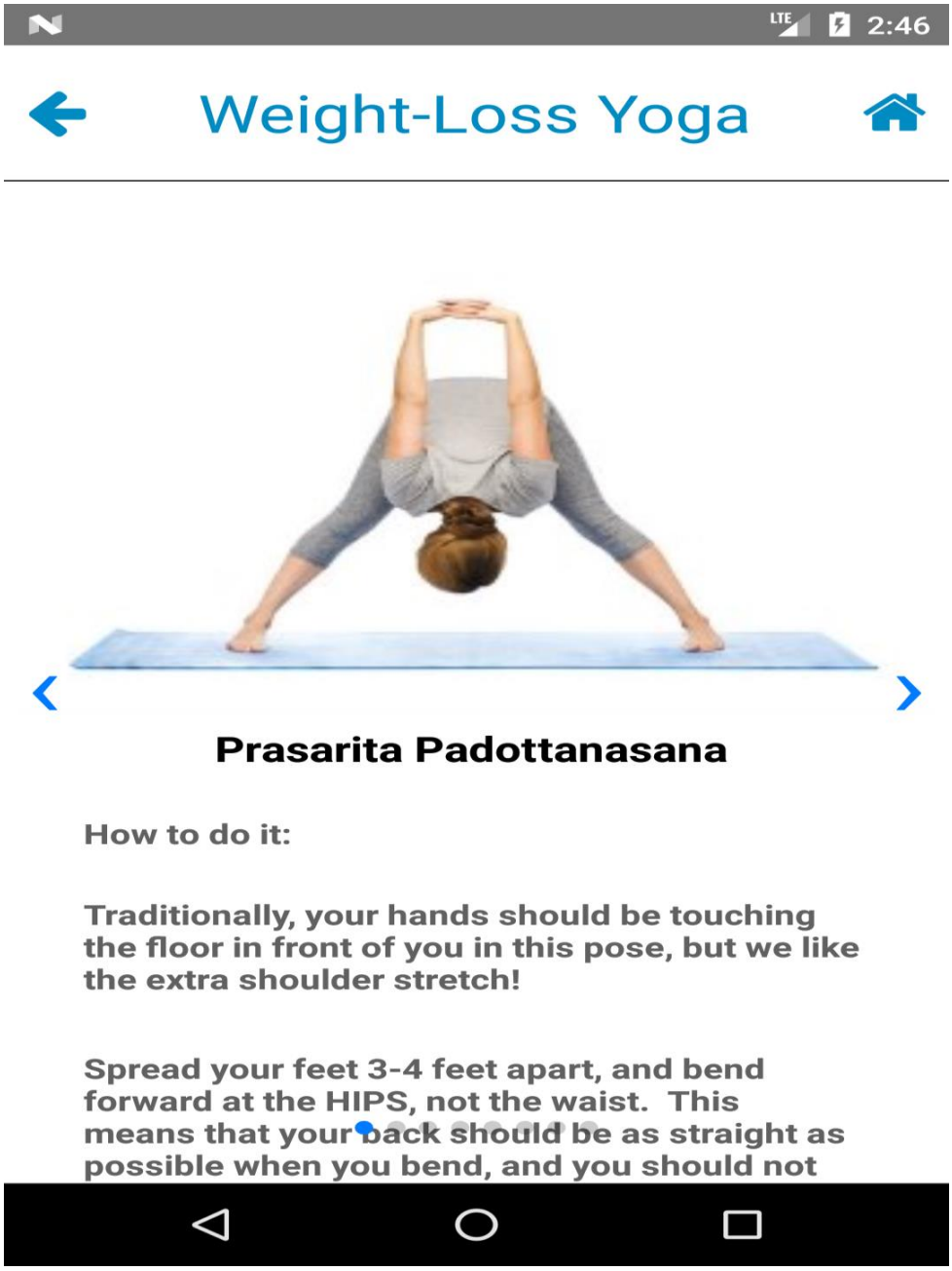


[Fig 8.2.1 (II) Workout1]

8.2.2 Yoga



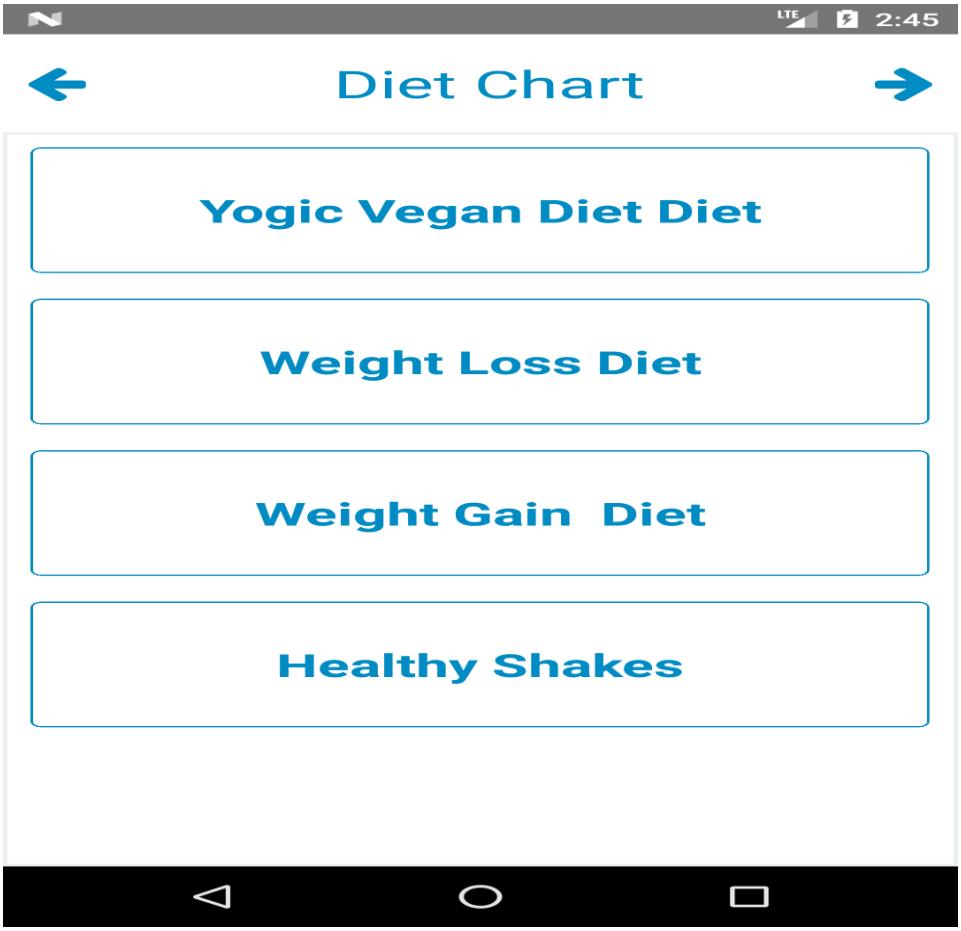
[Fig 8.2.2(I) Yoga]



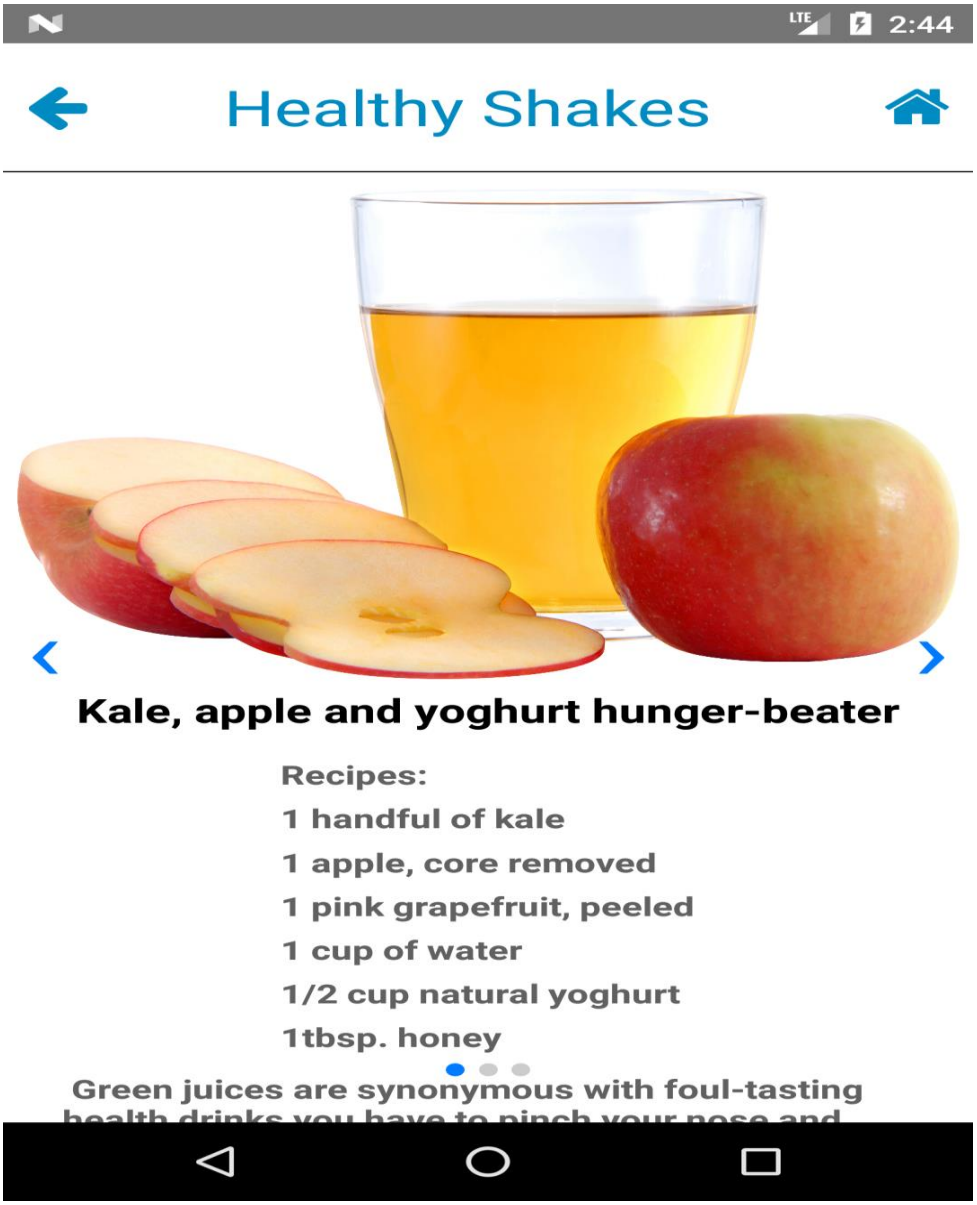
[Fig 8.2.2(II) Yoga1]

[Fig 8.2.2(III) Yoga2]

8.2.3 Diet

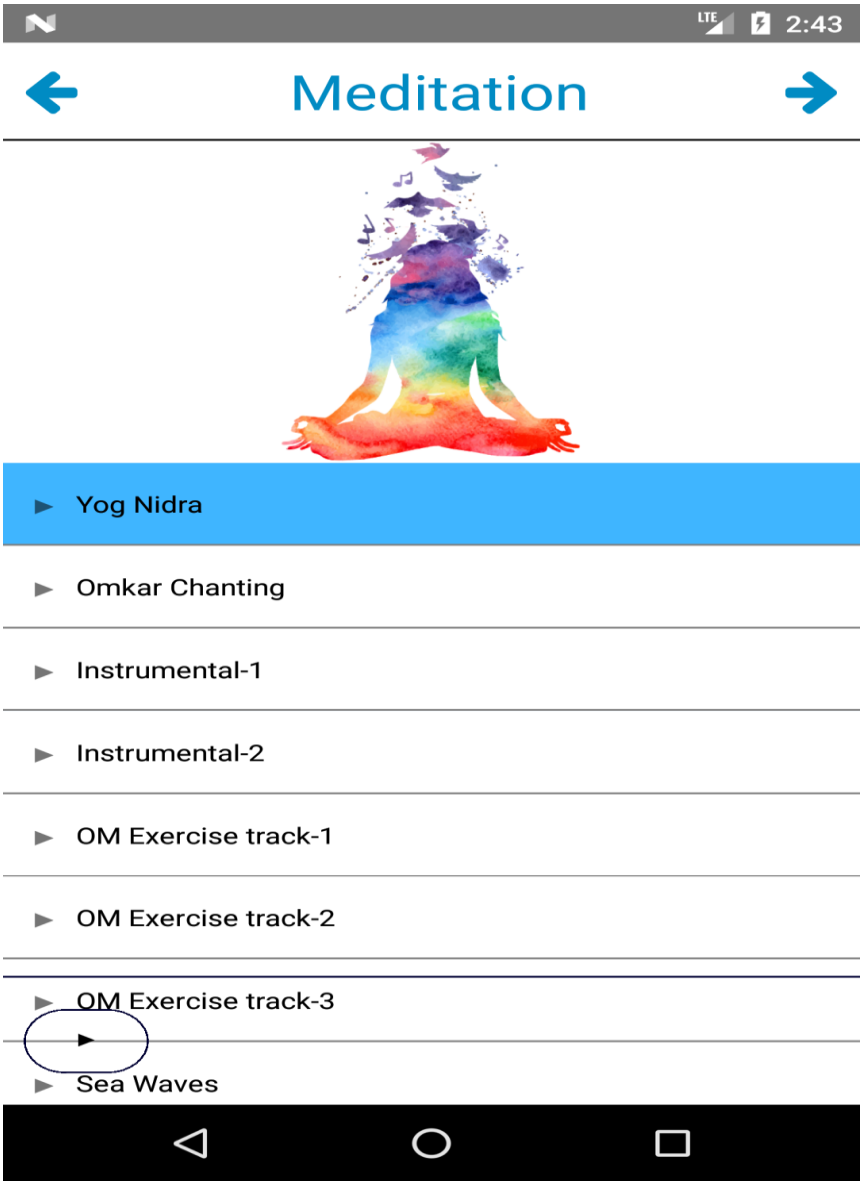


[Fig 8.2.3(I) Diet]



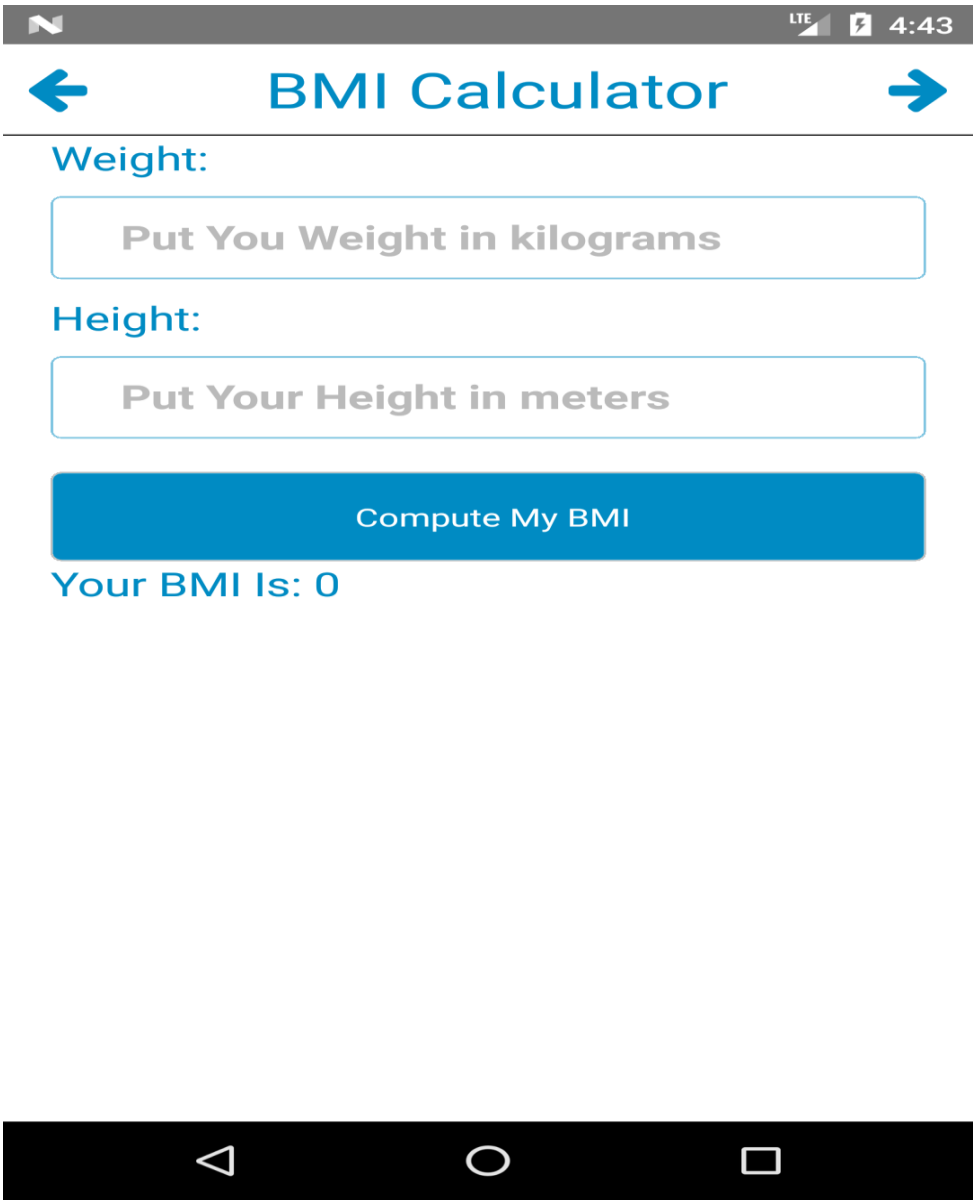
[Fig 8.2.3(II) Diet1]

8.2.4 Meditation



[Fig 8.2.1 Meditation]

8.2.5 BMI



[Fig 8.2.5 BMI]

8.2.6 Index Code

```
import { AppRegistry } from 'react-native';
import App from './App';
import dasbord from './page/dasbord'

import YogaList from './page/YogaList';
import yogadetails from './page/yogadetails';
import Suryanamaskar from './page/Suryanamaskar';
import weightlossyoga from './page/weightlossyoga';
import prePregnancyyoga from './page/prePregnancyyoga';
import postPregnancyyoga from './page/postPregnancyyoga';


import WorkOutList from './page/WorkOutList';
import basicworkout from './page/basicworkout';
import wormupworkout from './page/wormupworkout';
import weightlossWorkout from './page/weightlossWorkout';
import bicepworkout from './page/bicepworkout';

import DaietList from './page/DaietList';
import daietdetails from './page/daietdetails';
import weightlossDiet from './page/weightlossDiet';
import HealthyShakes from './page/HealthyShakes';


import meditation from './page/meditation';
import bmi from './page/bmi';
import trainer from './page/trainer';


AppRegistry.registerComponent('FitnFab', () => App);
```

[Fig 8.2.6 Index.js]

CHAPTER 9
CONCLUSION

CHAPTER 9
CONCLUSION

9 Conclusion

9.1 Accomplishments

While it was a challenge to develop, our project team successfully created a prototype wellness application for the Android Platform capable of calculating BMI, providing diet plans, yoga suggestion and workout plans. While the full scope of the initial app design was not realized, all of the core functionality has been successfully implemented.

9.2 Future Work

While we feel that we successfully implemented the desired functionalities, there are still several elements that could be added to improve the effectiveness of application. First of all we will be providing the user login facility and along with that capability of tracking, recording and displaying data relevant to a user's sleep, activity, and mood habits.