

**CRICKETARENA**

**MOHAMMAD HUMAYUN**

**[64086]**

**TALHA PATNI**

**[64085]**

**UNDER THE SUPERVISION OF**

**MUHAMMAD UMAIR QUARISHI**

**2023**

**COLLEGE OF COMPUTING & INFORMATION SCIENCES PAF-KIET CAMPUS, KARACHI**

**PAKISTAN**

**CERTIFICATE**

THIS PROJECT **“CRICKET ARENA”** PRESENTED BY **MOHAMMAD HUMAYUN (64086), TALHA PATNI (64085)** UNDER THE

DIRECTION OF THEIR PROJECT ADVISOR AND APPROVED BY THE PROJECT EXAMINATION COMMITTEE, HAS BEEN PRESENTED TO AND ACCEPTED BY THE COCIS, IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE BACHELOR’S DEGREE OF COMPUTER SCIENCE.

**MUHAMMAD UMAIR QUARISHI**



**(PROJECT ADVISOR)**

**USMAN KHAN**

**(FYP COMMITTEE HEAD)**





**DR.KHALID KHAN**

**(DIRECTOR COCIS)**

# ABSTRACT

Many young boys are facing ground booking problems for playing cricket and organized tournaments to show their talent. However, many of the boys want their scorecard to be noticed and can show people their consistency via social media apps so our mobile application purpose is to help these young boys to access all their preparations for playing cricket or organize tournaments by sitting at their homes by using their mobile phones. You just have to login and add basic information then you can use this app freely.

# ACKNOWLEDGEMENT

***In the name of Allah, the most Gracious and the Most Merciful.***

***Peace and blessing of Allah be upon Prophet Muhammad*** ﷺ

First, praise of Allah, for giving us this opportunity, the strength and the patience to complete our FYP finally, after the challenges and difficulties. We would like to thank our supervisor***\_\_\_\_\_\_\_M.Umair Qureshi\_\_\_\_\_***for his guidance, motivation and most his significant contribution in this

project, expert ***\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_*** and

***\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_*** for giving us the opportunity to work on this project. We would also like to thank our parents for financial and moral support and our friends who have helped and motivated us throughout.

May Allah reward them all abundantly. Ameen

# DEDICATION

This report is dedicated to PAF-KIET University, our Teacher, our Supervisor, our Parents, our fellow colleagues and the hard-working students of PAF-KIET, with a hope that they will succeed in every aspect of their Academic Career and this project may help them in any aspect of their life.

# TABLEOFCONTENTS

**ABSTRACT**..................................................................................................................... i

**ACKNOWLEDGEMENT**............................................................................................. ii

**DEDICATION**............................................................................................................... iii

**TABLE OF CONTENTS**.............................................................................................. iv

**LIST OF FIGURES**.......................................................................................................vi

**LIST OF TABLES**........................................................................................................vii

**CHAPTER 1**....................................................................................................................1

1. Introduction........................................................................................................... 1
   1. Motivations 1
   2. Problem Statement 1
   3. Objectives and Contributions 1
   4. Project Scope 2
   5. Organization of the Report 2

**CHAPTER 2**....................................................................................................................3

1. Literature Review/Process Review........................................................................3
   1. Introduction 3
   2. Literature Review 3
   3. Functional and Non Functional Requirements 4
   4. Project Significance 5
   5. Software Platform 5
   6. Scalability 5
   7. Services 5

**CHAPTER 3**....................................................................................................................6

1. Projects diagrams...................................................................................................6
   1. Use Case Diagrams 6
   2. Activity diagram (diagram correction) 8
   3. System Architecture 11
   4. ER Diagram 13
   5. Used Technologies 14

**Chapter NO 4**................................................................................................................ 15

1. Project Planning...................................................................................................15
   1. Project Timeline Summary 15
   2. Project Timeline Details 15
   3. Black-box Testing 15
   4. Test Cases 17

**CHAPTER NO 5**...........................................................................................................20

1. GUI of Android Application................................................................................20
   1. Login Interface 20
   2. Sign up Interface 21
   3. User Panel 22
   4. Ground Panel 23
   5. Admin Panel…………………………………………………………………23

**CHAPTER 6**..................................................................................................................24

1. Conclusion and Future Work............................................................................... 24
   1. Limitation 24
   2. Conclusion 25
   3. Future Works 25

References...................................................................................................................... 25

Appendix........................................................................................................................ 26

1. Coding – Login File.........................................................................................26
2. Gantt chart........................................................................................................39

# LISTOFFIGURES

|  |  |  |  |
| --- | --- | --- | --- |
| **S.NO** | **Figure No.** | **Figure Name** | **Page No.** |
| **1** | **Figure 1.4** | **Project Scope** | **2** |
| **2** | **Figure 3.1** | **Use Case Diagram - Android Application** | **6** |
| **3** | **Figure 3.2** | **Activity Diagram** | **8** |
| **4** | **Figure 3.3** | **System Architecture** | **11** |
| **5** | **Figure 3.4** | **ER Diagram** | **13** |
| **6** | **Figure 4.3** | **Project Timeline** | **15** |
| **7** | **Figure 5.1.1** | **Login Interface** | **20** |
| **8** | **Figure 5.1.2** | **Sign Up Interface** | **21** |
| **9** | **Figure 5.1.3** | **User Panel** | **22** |
| **10** | **Figure 5.1.5** | **Ground Panel** | **23** |
| **11** | **Figure 5.1.8** | **Admin Panel** | **25** |
| **12** | **B.** | **Gantt Chart** | **41** |

# CHAPTER1

## 1. Introduction

Cricket Arena is an app for the local passionate cricketers of the world. It allows you to score your local cricket match. Cricket Arena also allows you to add your own team, search players, search teams, arrange a match with local teams, find tournaments around you, book grounds of your own choice and much more. Cricket Arena is your new Cricket Network!

Made in Pakistan by the passionate cricketers for the passionate cricketers of the world!

.

**1.1. Motivations**

Our motivation is to fulfill the requirements of youth to make themselves bigger in the world of cricket and be a pride of Pakistan because our youth didn’t get many chances to show their talent & Passion which may lead to stress and mental problems. So just to ensure a good life we are making a Cricket app to achieve goals for our users.

**1.2. Problem Statement**

Ground owners are struggling to make their ground famous and successful in terms of booking and earning money.

Some have reduced their staff, but it still doesn’t come close to the revenue they’d made in person. In a recent survey by cricket researchers stated that only 31% of people who belonged to a ground and played cricket have returned and many more plan to cancel their dreams and change their occupations.

So, we just tried to solve this problem as we all are so concerned with our dreams & passion.

**1.3. Objectives and Contributions**

The objectives of this project are:

* A way to connect with users throughout the day and keep them updated with prices of grounds and new tournaments.
* Improve existing problems.
* Deliver content virtually without any expense.
* We all can continue staying motivated to book ground on my own at home.
* Can create a scorecard of their current match and prove it.
* Helps everyone to make their country proud in a world of sports.
  1. **Project Scope**

To begin the design process, we wanted to define the problem better. We initially created proto- personas and then created user need statements.

Many users can use at a single time Avoid wasting fuel.

Time saving

Can book and organize tournaments at home.

Can book ground at home.

Increased flexibility in scheduling, cost.

No need to face traffic congestion

* 1. **Organization of the Report**

This report is composed of six chapters. Chapter one includes introduction of project, motivation of project, objectives and contributions for the project, project scope, organization of report. Chapter 2 is about literature review/process review which includes introduction, literature review, functional and non-functional requirements, project significance, scalability, software platform and services .Chapter 3 is the introduction of project diagrams which includes use case diagrams, entity system block diagram and circuit diagram. Chapter 4 is about project planning which includes project timeline summary, testing and test cases. Chapter 5 includes the GUI of the project. Chapter 6 includes conclusion, limitations, future work, references and the coding which is done for the application.

# CHAPTER2

## 2. Literature Review/Process Review

**2.1. Introduction**

After researching we come to the conclusion that mobile apps these days are fairly in demand. We decided to go with Flutter because it is very user friendly. This also seemed like a well-placed opportunity to become adept in the dart Language hence we picked this as it will turn out to be a good way to learn it. Also learning about cloud functions, and data Fetching made me realize how data will be handled on the backend so I decided to go with Google Firebase.

**2.2. Literature Review**

Cricket ground booking apps have gained significant attention in recent years, revolutionizing the way cricket enthusiasts reserve and manage their ground bookings. This literature review aims to explore the existing research and trends related to cricket ground booking apps, highlighting their benefits, challenges, and potential areas for improvement.

* User Adoption and Satisfaction
* Technological Features and Integration
* Challenges and Limitations
* Social and Community Integration
* Business Models and Revenue Generation
* User Experience and Interface Design

**Functional and Non-Functional Requirements**

***2.2.1. Functional Requirements***

List of functional requirements:

* User registration
* User’s image
* Ground registration
* Ground’s Image
* Scorecard
* Player Searching
* Live Score
* Ground Booking

***2.2.2. Non-Functional Requirements***

Non-functional requirements are requirements that simplify the design for the user. However, if they don’t finish, it doesn’t mean the project can’t work.

* Performance
* Capacity
* Maintainability
* Security
* Scalability
* Reliability ● Usability
* Regularity
  1. **Project Significance**

Cricket app development, like any other type of app development, can be difficult. It is, nonetheless, effective and worthwhile if you ensure that the cricket app of your choice includes the best features that will be of value to users in the booking of a ground, Tournament and scoring purpose. To do so, make sure you can provide people with the best of your features that are tailored to their specific needs. It’s more likely that your cricket app will stand out and succeed if you have the sincere intention of assisting users in achieving their sports-based goals through various features.

* 1. **Software Platform**

The best available software mentioned below was used to extract, store and perform the required applications.

* For development: Flutter for mobile app development and Visual Studio Code for Dart have been used.
* For testing: Multiple phones have been used.
  1. **Scalability**

As of now it’s only for the people of KIET north nazimabad campus but as it evolves, we can add more features to it and make it customizable, products will be added with each update to the application. The project solves general problems of people of every age group which makes it more scalable.

* 1. **Services**

This report will be providing the following services to the users regarding this project

* Mobile app development in Flutter ● Google Firebase for Data handling usage ● Dart for flutter.

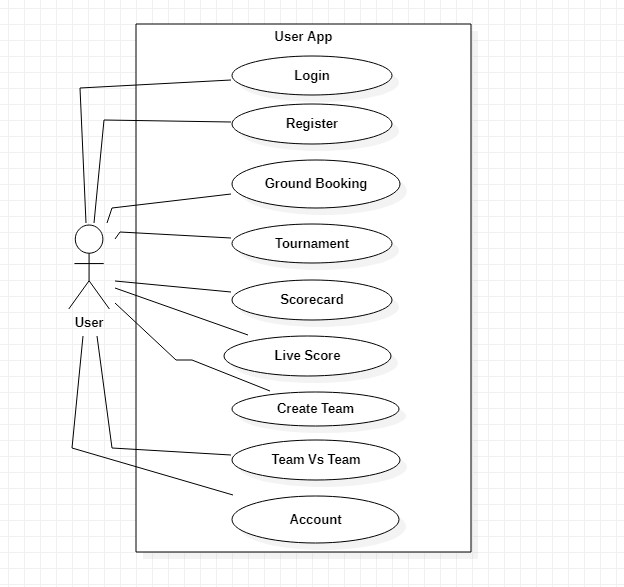
# CHAPTER3

## 3. Projects diagrams

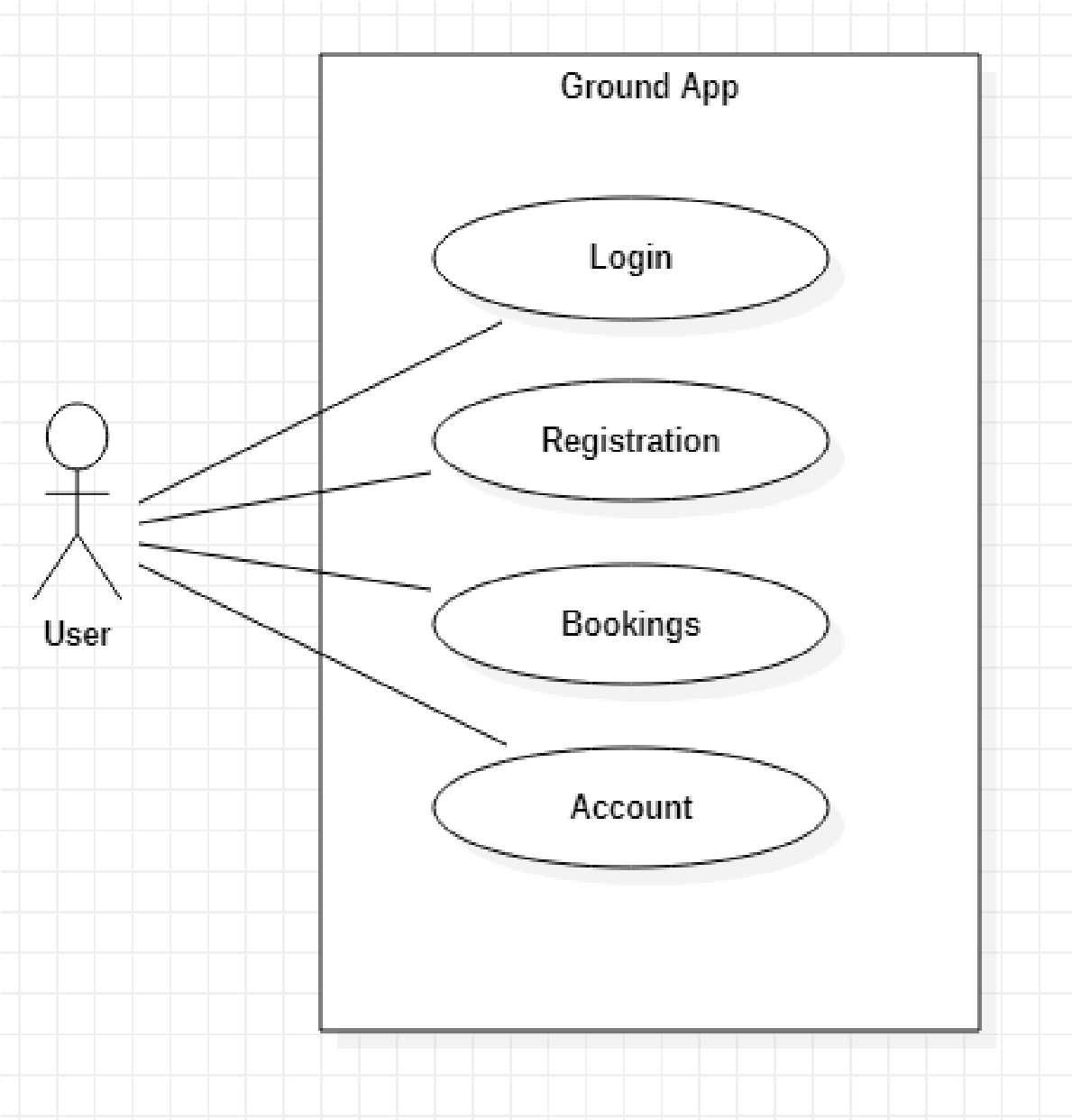
Based on the above literature review and project scope here are some diagrams, which illustrates that what will be our project or the system is capable to reach the desired results.

**3.1. Use Case Diagrams**

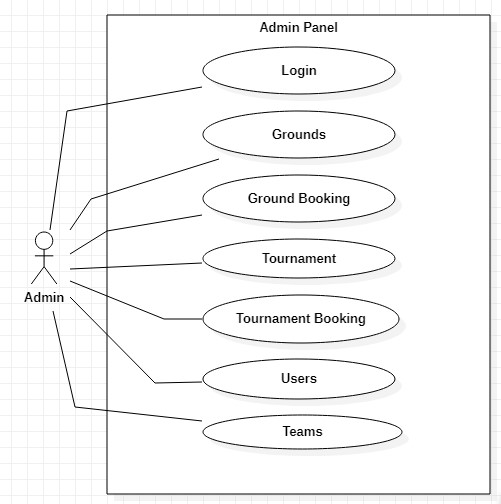
**3.1.1 User App Use Case Diagram**



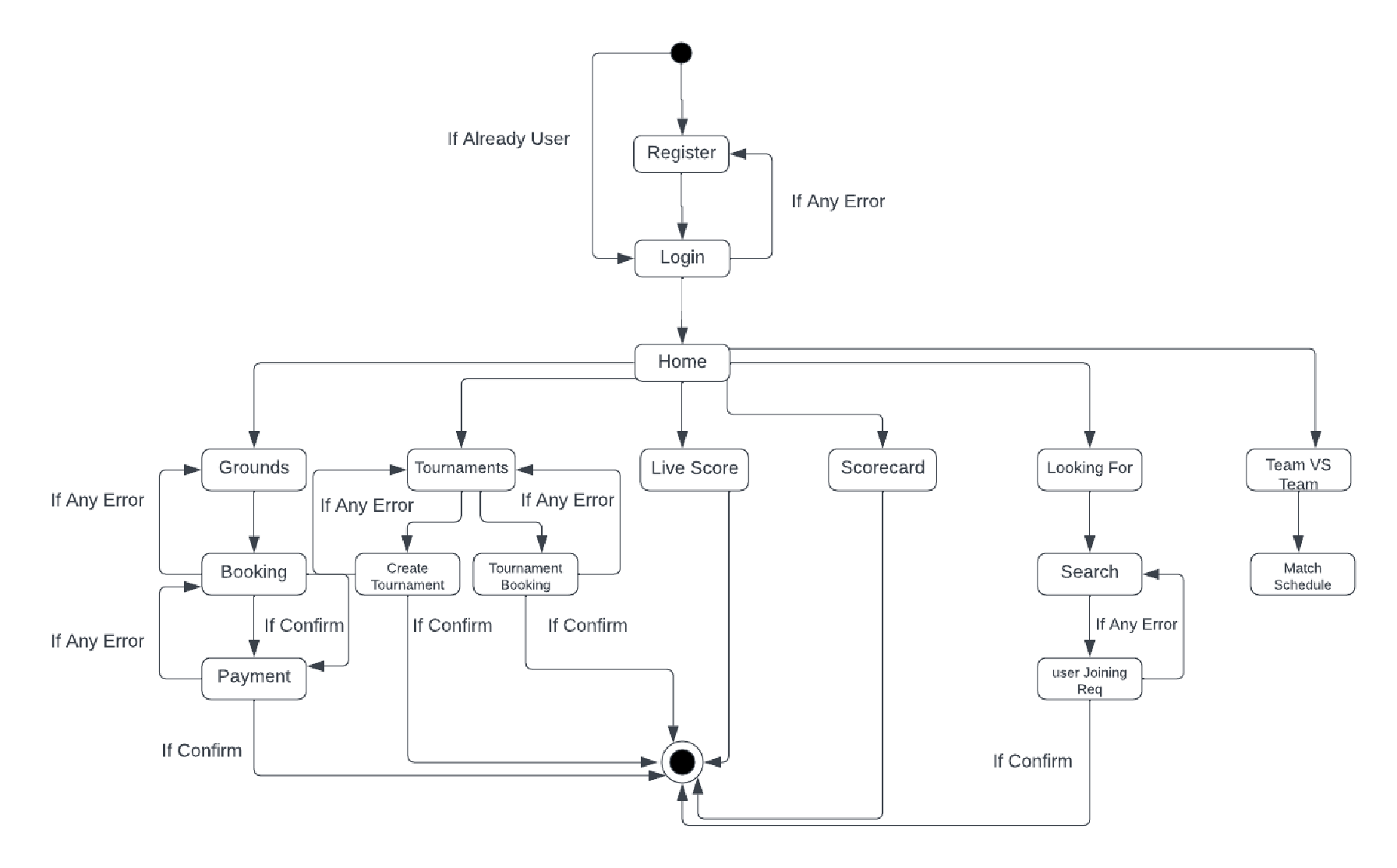
**3.1.2 Ground App use case Diagram**



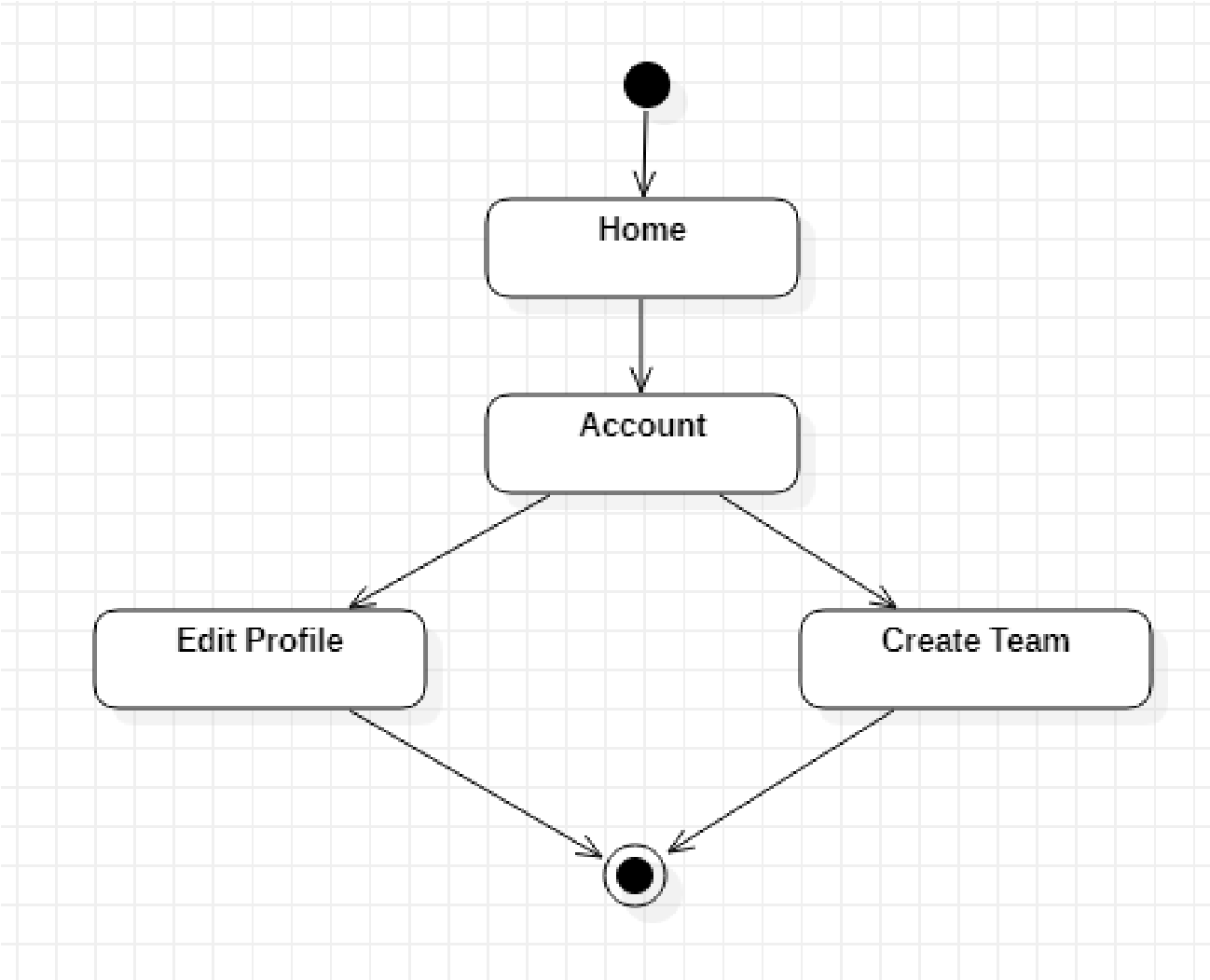
**3.1.3 Admin Dashboard use case Diagram**



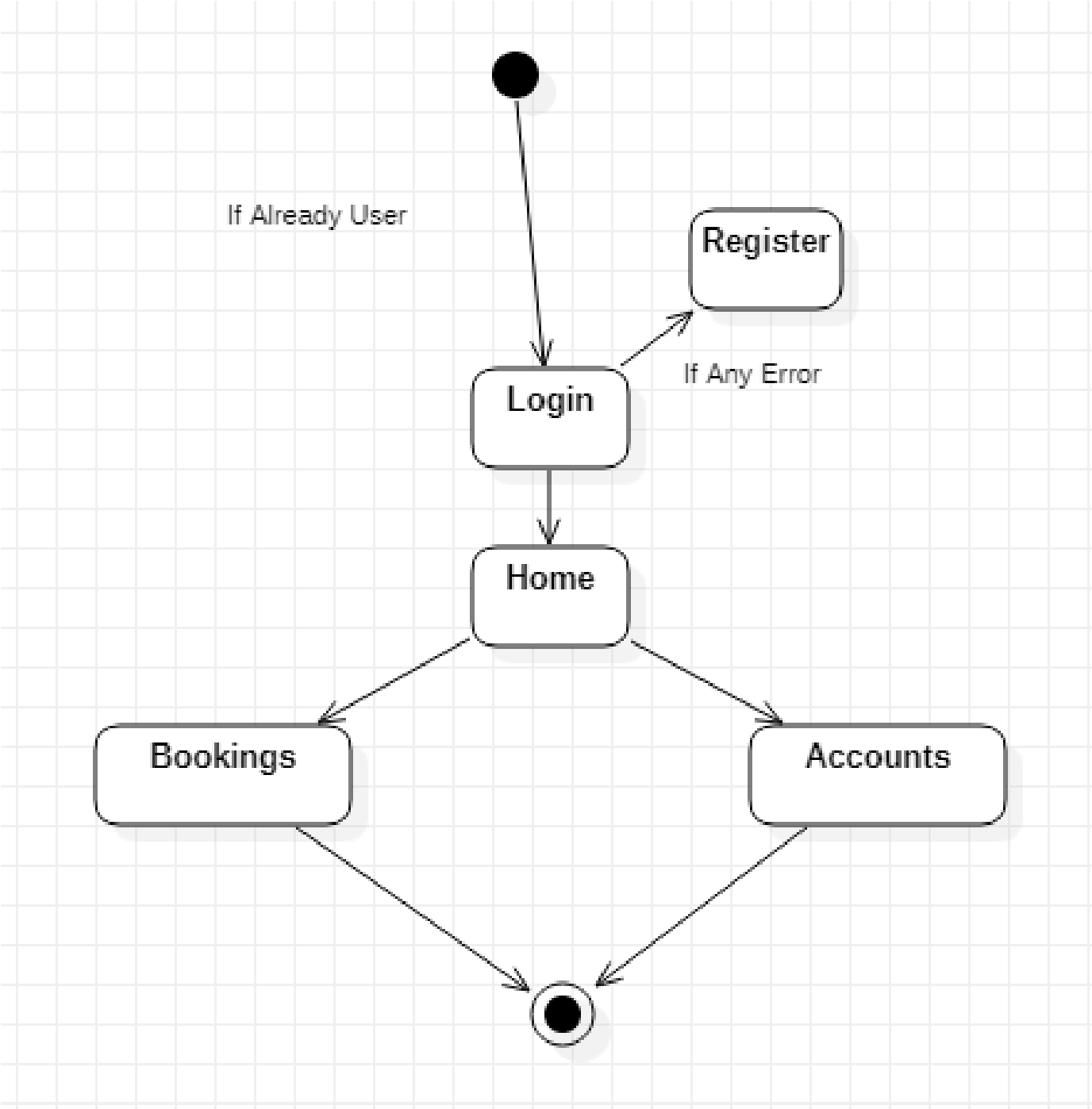
**3.2. Activity diagram (diagram correction) 3.2.1 User app Activity Diagram**



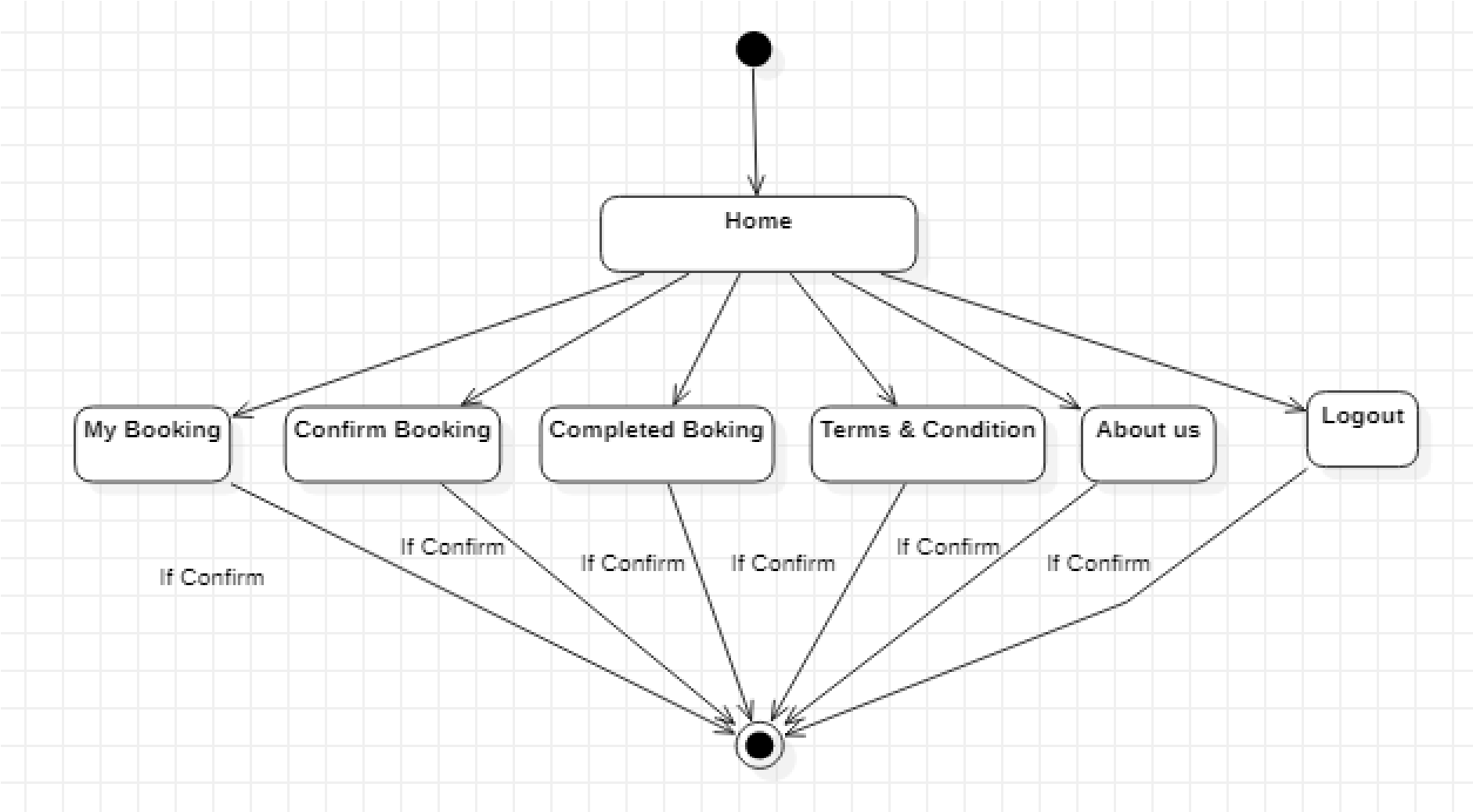
**3.2.2 User App Account Activity Diagram**



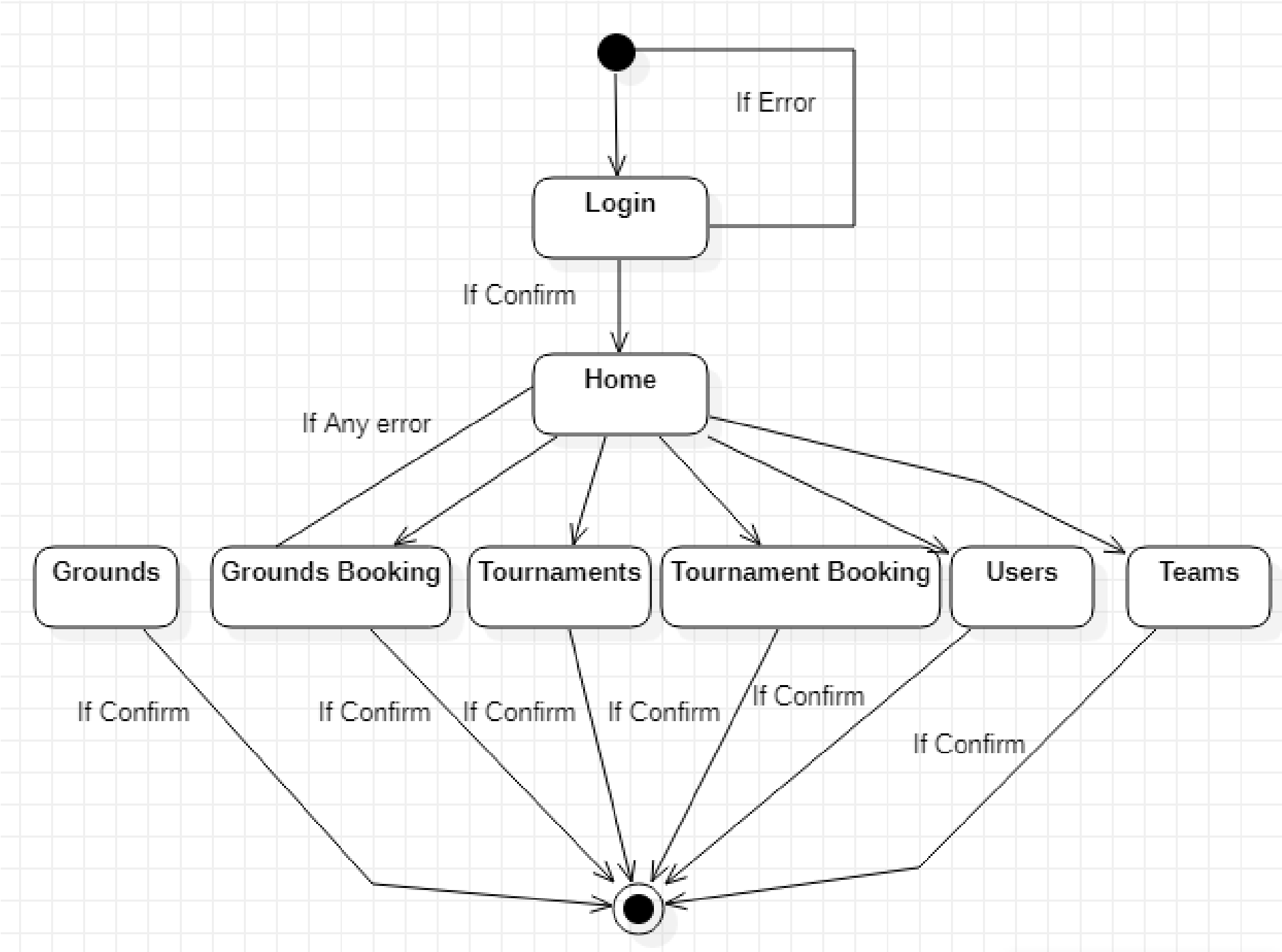
**3.2.3 Ground App Activity Diagram**



**3.2.4 Ground App Drawer Diagram**

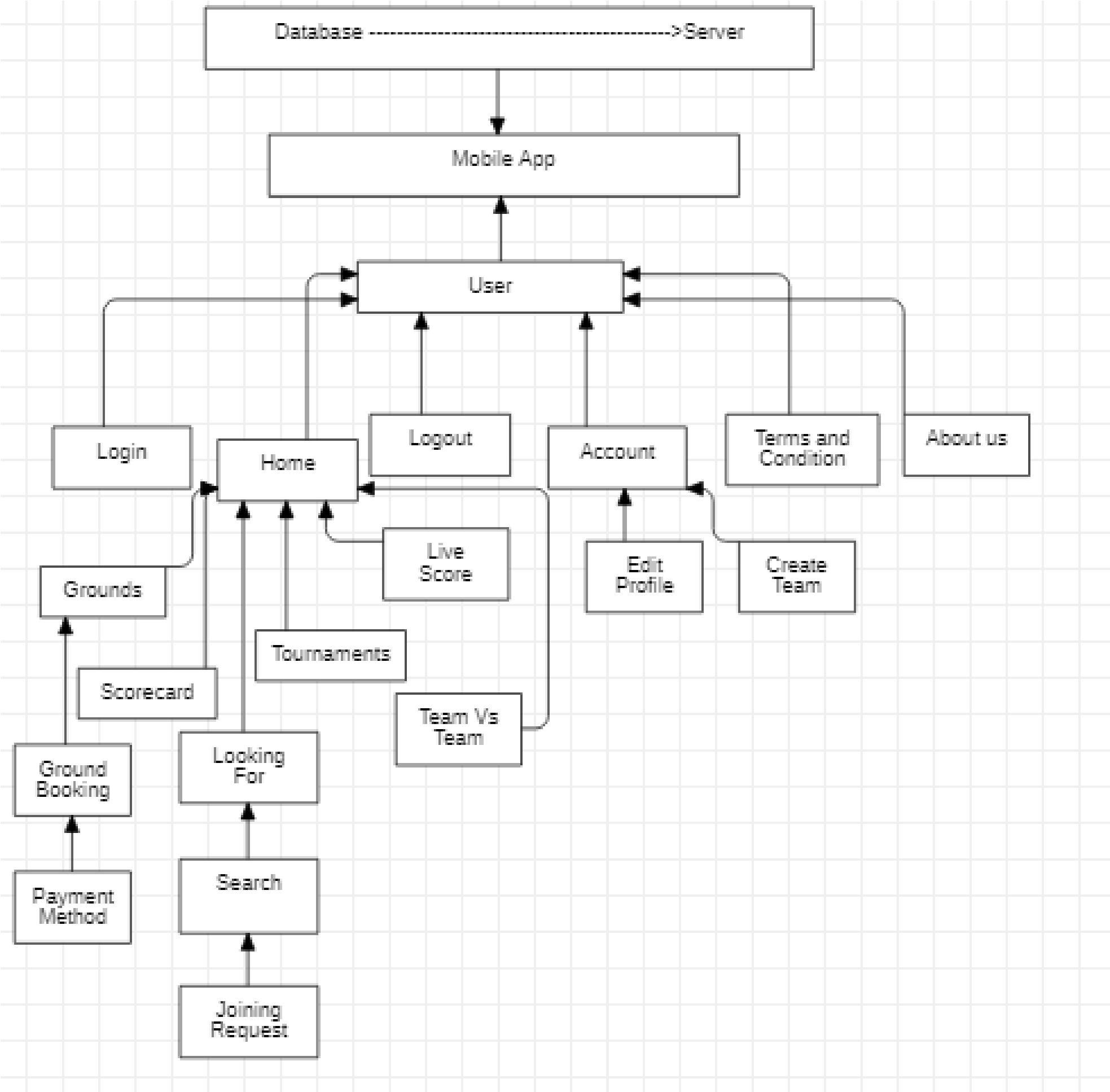


**3.2.5 Admin Dashboard Activity Diagram**

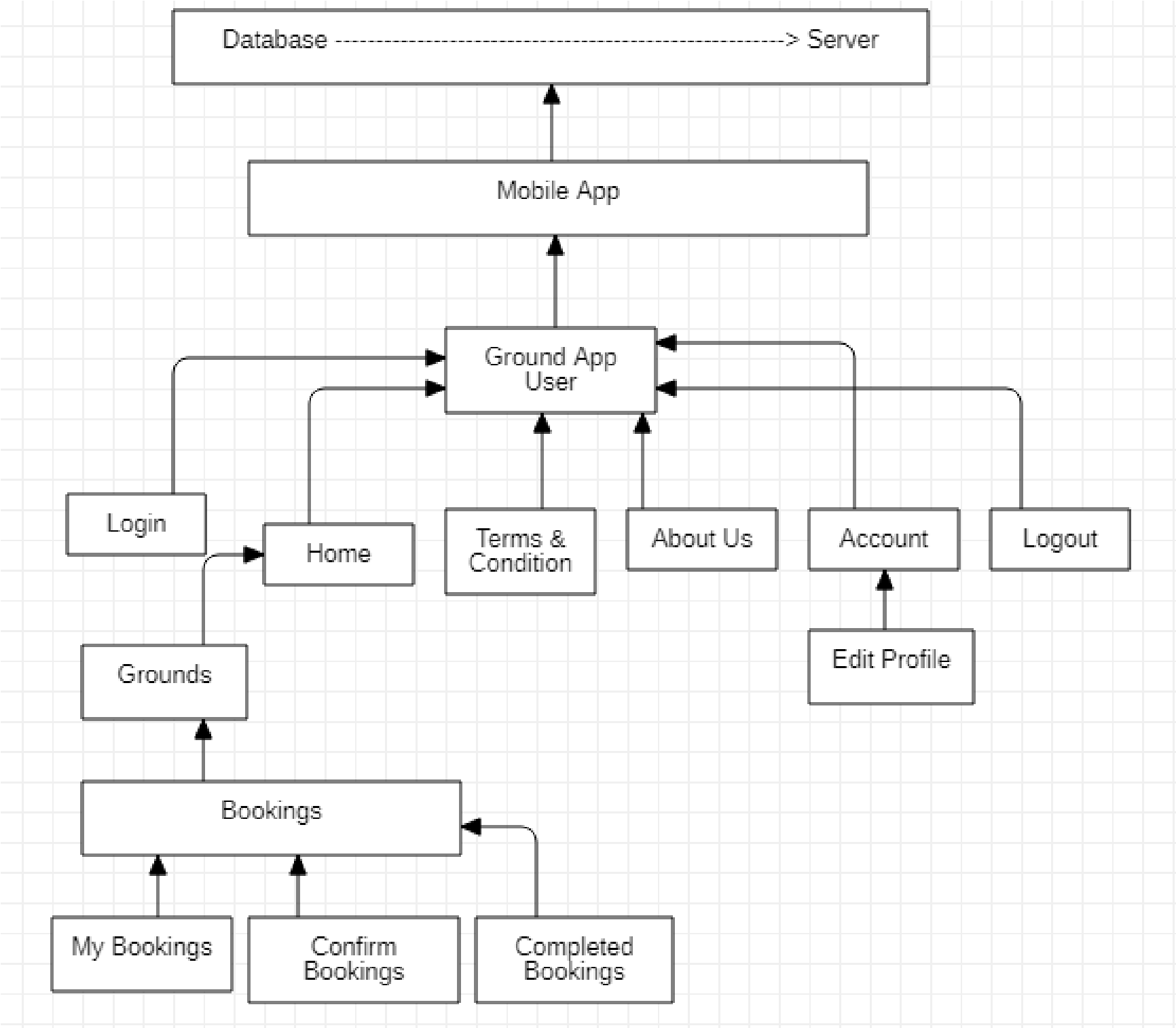


**3.3. System Architecture**

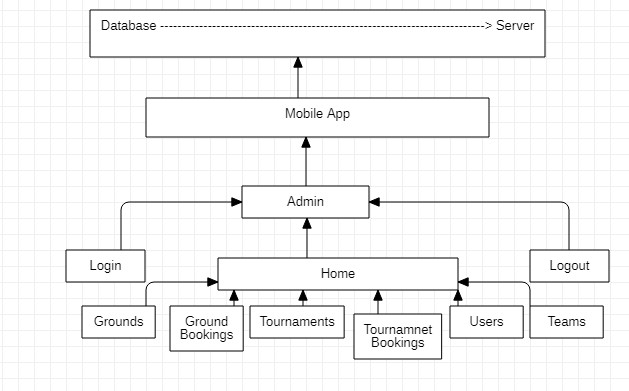
**3.3.1 User App System Architecture**

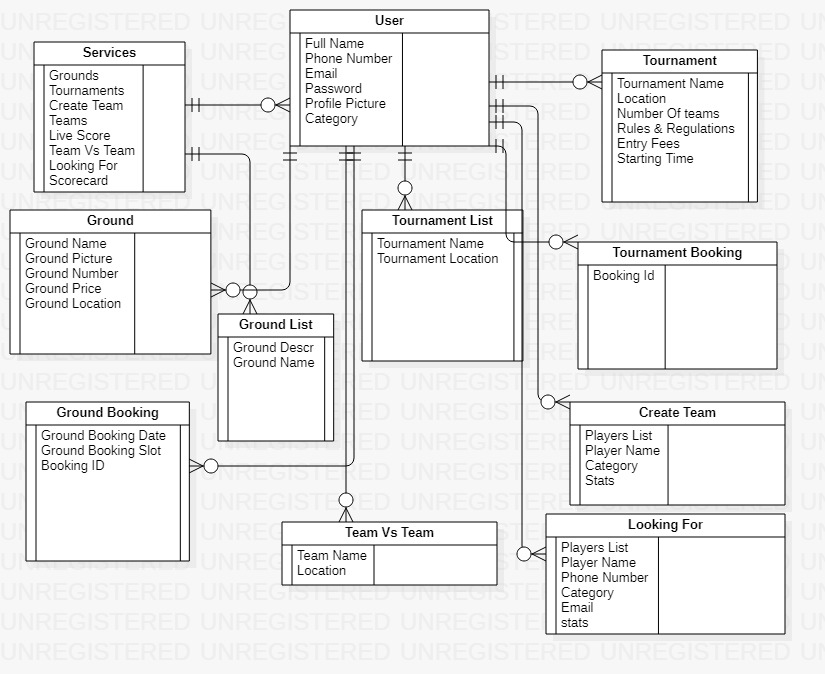


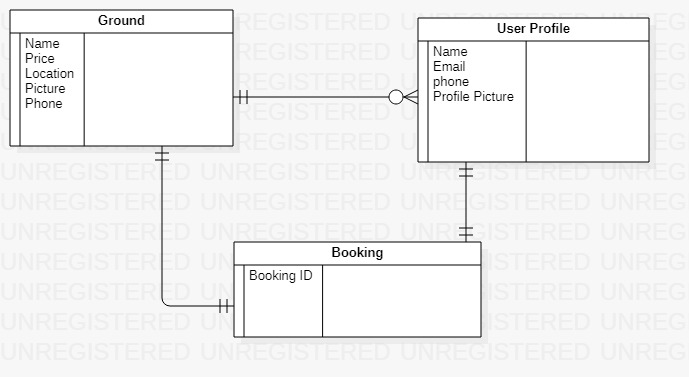
**3.3.2 Ground App System Architecture**

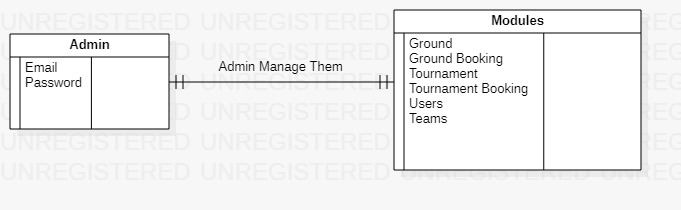


**3.3.3 Admin Dashboard System Architecture**



**3.4. ER Diagram**  
**3.4.1 User App:**  


**3.4.2 Ground App:**  


**3.4.3 Admin Panel:**  


**3.5. Used Technologies**

**Flutter:**

Flutter is an open source GUI developing software Released in 2017 which is created by Google and used to create cross-platform applications from a single codebase for iOS, Windows, Linux, Android, Google Fuchsia, Mac, web and web platform. Using a single codebase and programming language, Flutter allows developers to construct iOS and

Android mobile apps. This feature simplifies and accelerates the development of iOS and Android apps.

**Dart:**

It is a programming-based language which is developed by Google, and it first appeared on 10-October-2011 which is designed for client development such as for the mobile and web apps.

**Visual Studio:**

A feature-rich tool that covers several elements of software development is known as an integrated development environment (IDE). The Visual Studio IDE is a creative starting pad for editing, debugging, and building code, as well as publishing an app. Visual Studio contains compilers, code completion tools, graphic designers, and many other features to aid the software development process, in addition to the conventional editor and debugger that most IDEs include.

**Firebase:**

Firebase is actually a Backend-as-a-Service developed by Google Firebase is actually a server ,API , real time data storage it has authentication , file storage, analytics and the list goes on. All these services are hosted in Google cloud services with very less effort from the back-end developer. Traditional app development involves writing 100 lines of back end code but using firebase developers just have to query the database which makes our life very easy.

# CHAPTERNO4

## ● 4.1 Project Planning

Some time recently beginning any venture, it is vital to check which assignments got to be done and appropriately distribute assignment to gather members in extend improvement project arranging is an organized administration prepare that centered on the assignment required to complete on time it defines the parts of the group working on the extend this avoids the roadblocks and obstacles to completing the project on given time. Our application is Cricket Arena, which is partitioned into different parts.

* **4.2. Project Timeline Summary**

We have made sprint and excess of our system which contain all the task

* **4.3. Project Timeline Details**



* **4.4. Black-box Testing**

Black Box Testing, also known as Behavioral Testing, is a software testing method in which the internal structure/ design/ implementation of the item being tested is not known to the tester. These tests can be functional or non-functional, though usually functional. Testing, either functional or non-functional, without reference to the internal structure of the component or system.

This technique is the procedure to derive and/or select test cases based on an analysis of the specification, either functional or non-functional, of a component or system without reference to its internal structure.

Black Box testing method is applicable to the following levels of software testing:

* Integration Testing
* System Testing
* Acceptance Testing
* ***4.5. System Testing***

It could be a level of software testing strategy in which the complete project is tested coordinates with its environment. The reason for this testing level is to decide in case the project is working concurring to user prerequisites. Framework testing is really a total projecttesting coordinates with its environment the process of the testing is to ensure that the framework is working appropriately in its environment because a computer program could be a little component of a huge computer environment so it is essential to guarantee that the project works accurately with its environment,equipment and computer program components.

* ***4.6. Integration Testing***

This testing may be a level of computer program testing procedure in which distinctive modules or parts of the project are tested as a bunch. Integration testing is for when we combine distinctive components / modules of the project with each other we need to form beyond any doubt that the integration has no flaws in it. It is used to uncover the flaws and defects interfacing and within the intelligence between the components.

* ***4.7. Unit Testing***

Unit testing may be a computer program testing level in which as it were a little portion or component of the framework is tested. Unit is the smallest portion of the program and it is testable. The purpose of this level is to guarantee each component is working as outlined for the most part performed by the developers.

* ***4.8. User Acceptance Testing***

This testing is the final portion of a program test it is additionally called beta testing.This level is the foremost critical because the genuine user tests the computer program to create beyond any doubt, if the program is working accurately within the genuine world scenarios. In addition, in case everything is working fine and the program is working according to the prerequisites, the program is at that point distributed appropriately within the advertisement.

* **4.9. Test Cases.**

**TEST CASE # 1**

**Test Case Title:** Verify Registration

|  |  |
| --- | --- |
| Preconditions | Registration interface should be visible to the user |
| Actions | Filling out the fields and entering emails address and password |
| Expected Results | The user can’t proceed without filling out the all field and successfully register |
| Tested By | Mohammad Humayun Khalid |
| Result | Pass |

**TEST CASE # 2**

**Test Case Title:** Verify Login functionality

|  |  |
| --- | --- |
| Preconditions | Textboxes are should be clear |
| Actions | Enter valid email address and password |
| Expected Results | successfully login |
| Tested By | Mohammad Humayun Khalid |
| Result | Pass |

**TEST CASE # 3**

**Test Case Title:** Verify Login functionality

|  |  |
| --- | --- |
| Preconditions | Textboxes are should be clear |
| Actions | Enter in-valid email address |
| Expected Results | successfully login |
| Tested By | Mohammad Humayun Khalid |
| Result | Fail |

**TEST CASE # 4**

**Test Case Title:** Verify Login functionality

|  |  |
| --- | --- |
| Preconditions | Textboxes are should be clear |
| Actions | Enter wrong password |
| Expected Results | successfully login |
| Tested By | Mohammad Humayun Khalid |
| Result | Fail |

**TEST CASE # 5**

**Test Case Title:** Moving towards Main Menu

|  |  |
| --- | --- |
| Preconditions | User must be login on this app |
| Actions | After login navigates into Main screen |
| Expected Results | Home screen appears |
| Tested By | Talha Patni |
| Result | Pass |

**TEST CASE # 6**

**Test Case Title:** Grounds shown in list

|  |  |
| --- | --- |
| Preconditions | First Ground must be registered to app |
| Actions | Users press the specific ground option, they will see the list of grounds. |
| Expected Results | After selecting the specific ground then they can see the ground profile. |

|  |  |
| --- | --- |
| Tested By | Talha Patni |
| Result | Pass |

**TEST CASE # 7**

**Test Case Title:** Tournament list should be shown

|  |  |
| --- | --- |
| Preconditions | Tournament should be crated |
| Actions | User press specific tournament button to get list |
| Expected Results | Tournament List Appears |
| Tested By | Talha Patni |
| Result | Pass |

**TEST CASE # 8**

**Test Case Title:** Challenge to other team

|  |  |
| --- | --- |
| Preconditions | First team should be created |
| Actions | User send request to other team |
| Expected Results | Team up match should be schedule |
| Tested By | Talha Patni |
| Result | Pass |

**TEST CASE # 9**

**Test Case Title:** Searching for a team members

|  |  |
| --- | --- |
| Preconditions | User must be registered to the app |
| Actions | Have to search via looking for module |
| Expected Results | Player found |
| Tested By | Talha Patni |
| Result | Pass |

**TEST CASE # 10**

**Test Case Title:** Match Updates should be shown

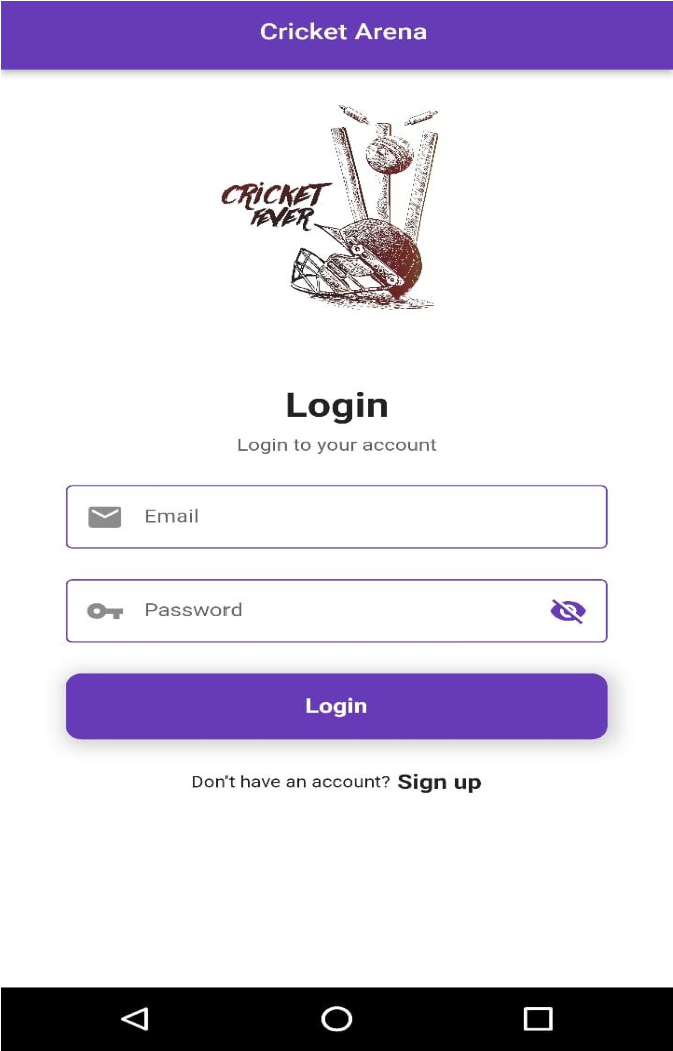
|  |  |
| --- | --- |
| Preconditions | Match have been starts |
| Actions | user have to navigate to live score options |
| Expected Results | Live Score Updates shown |
| Tested By | Talha Patni |
| Result | Pass |

## CHAPTERNO5

### **● 5.1 GUI of Android Application**

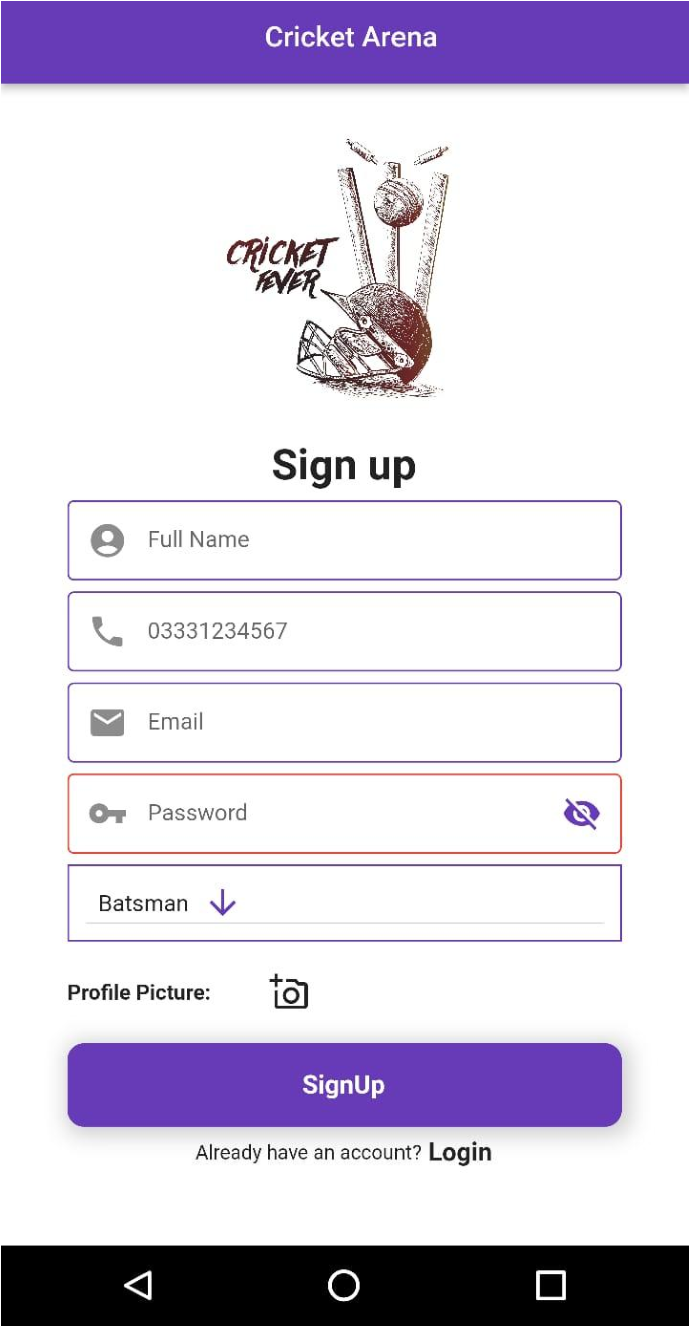
* **5.1.1. Login Interface**

This is the user side Login interface where user provide their credentials to login to the application. It contains an email address text box and password text box which user have entered in creating their account.

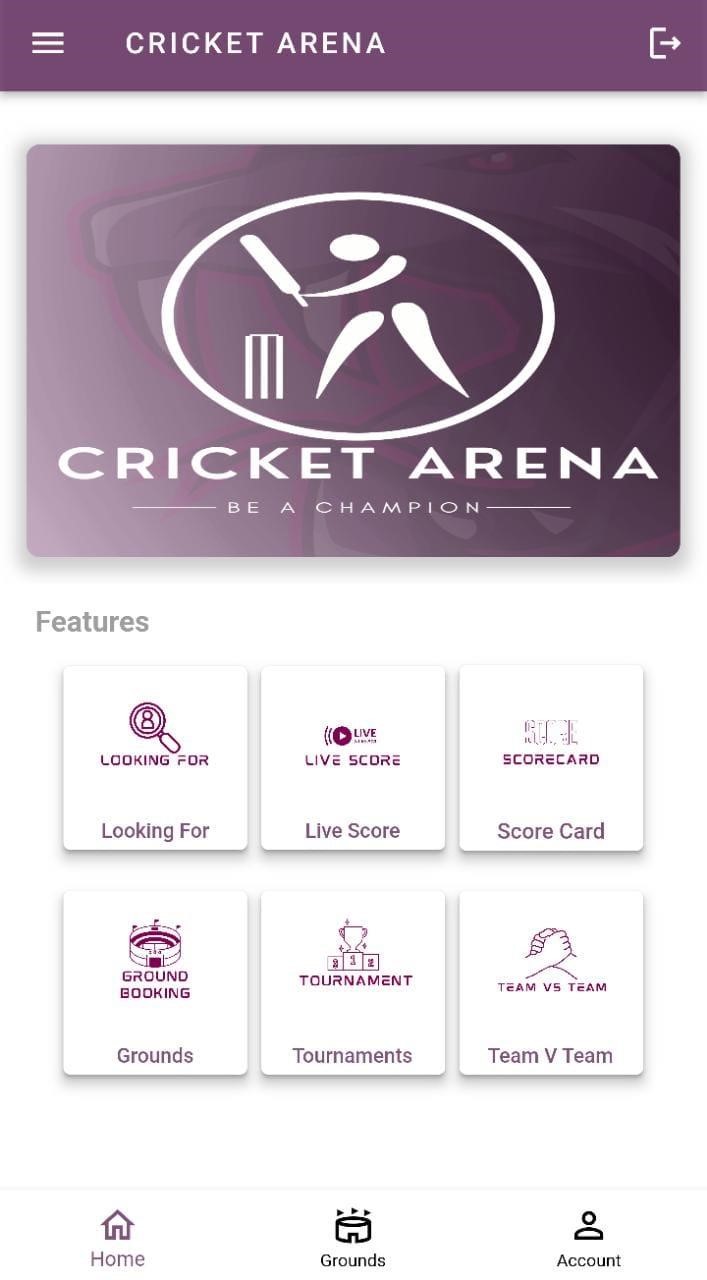


**5.1.2. Sign up Interface**

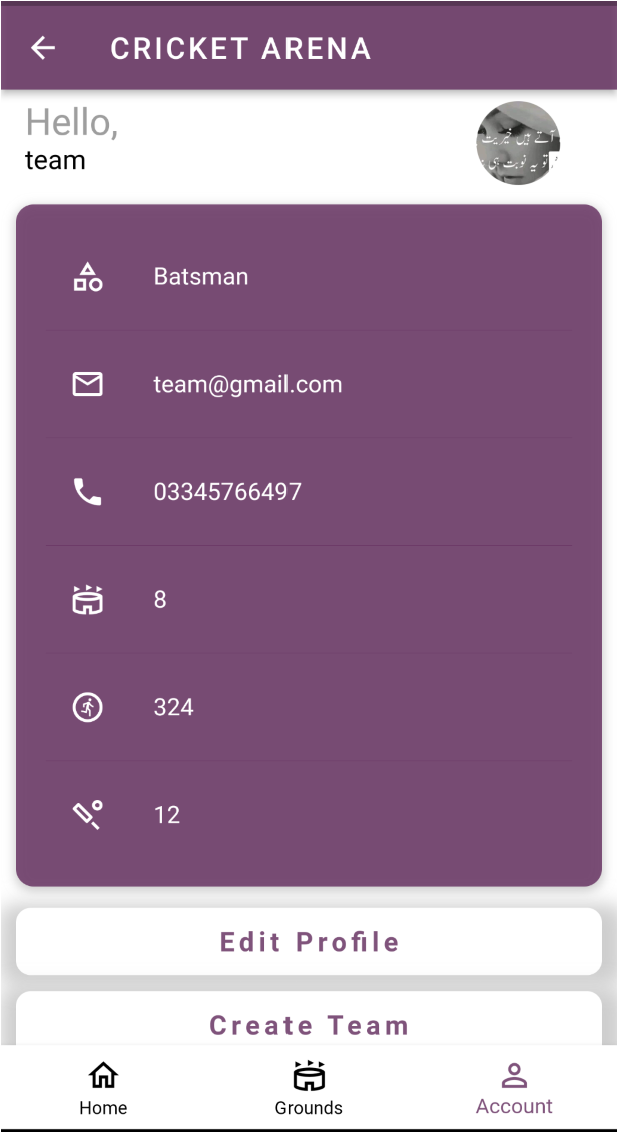
This is the user side Sign up interface where user provide their credentials to create an account. It contains full name, phone number, email address, password and Category text boxes and a profile picture box to upload a profile pic and all these information is mandatory for user which they must enter for the creation of their account

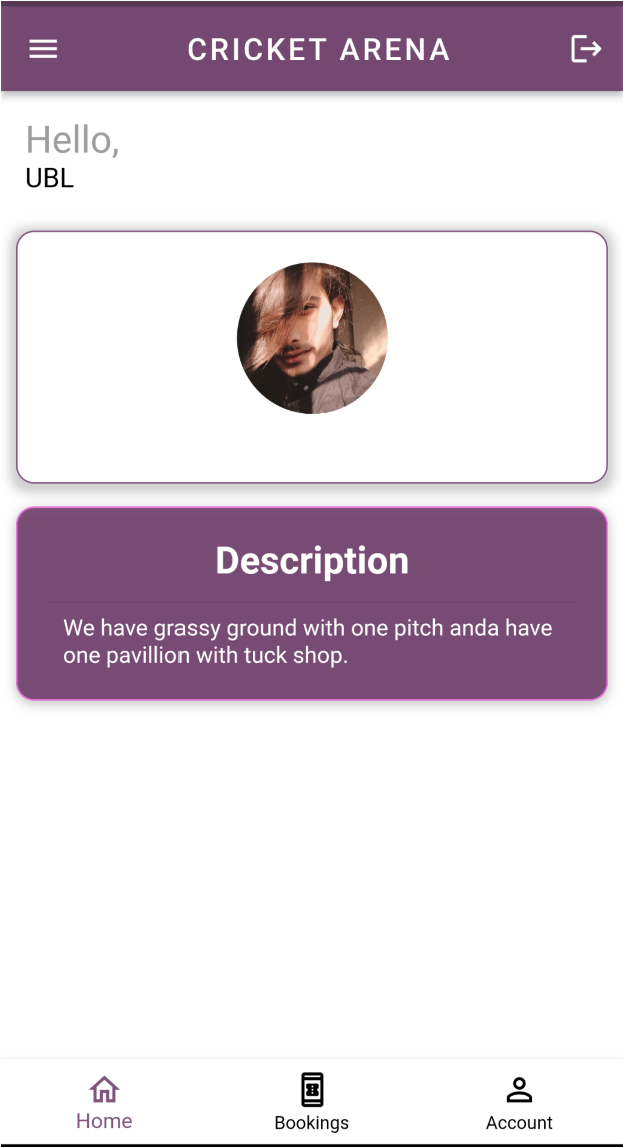


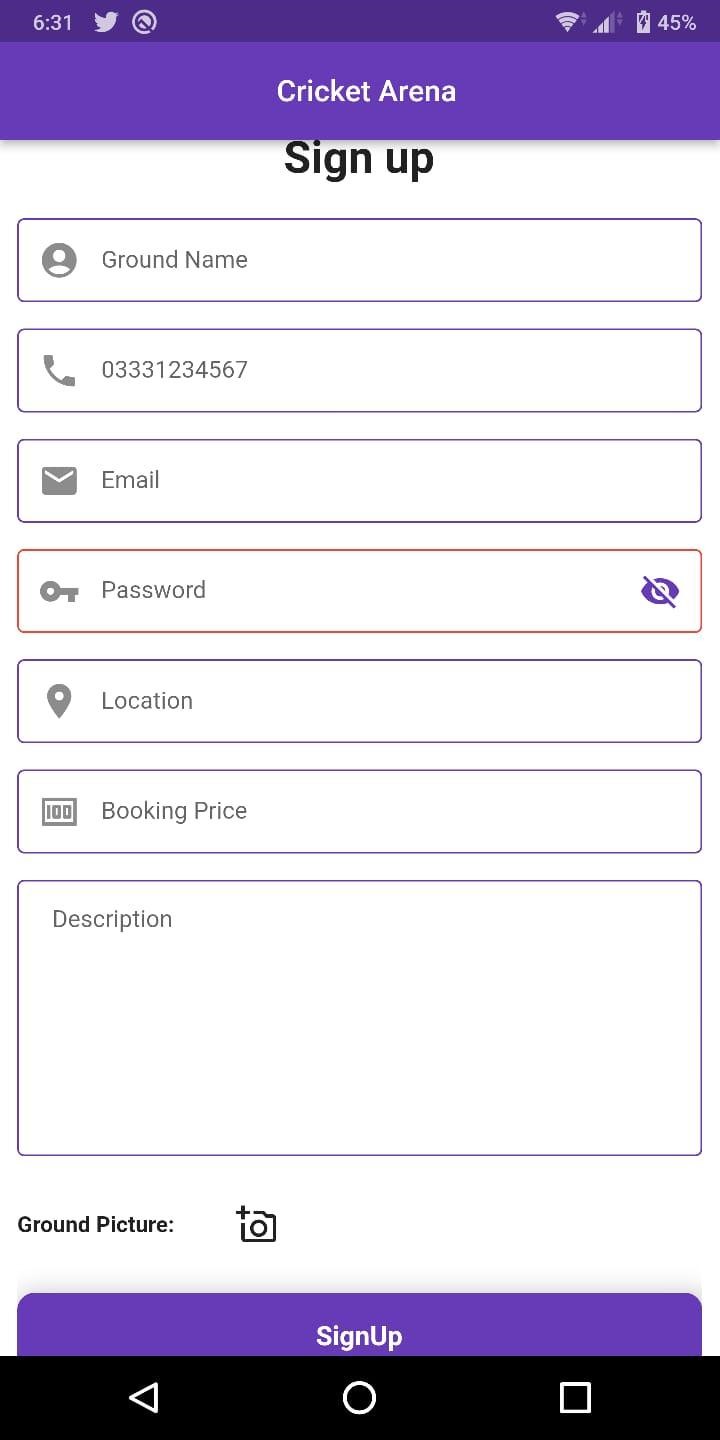
**5.1.3. User Dashboard**



* **5.1.4. User Account**

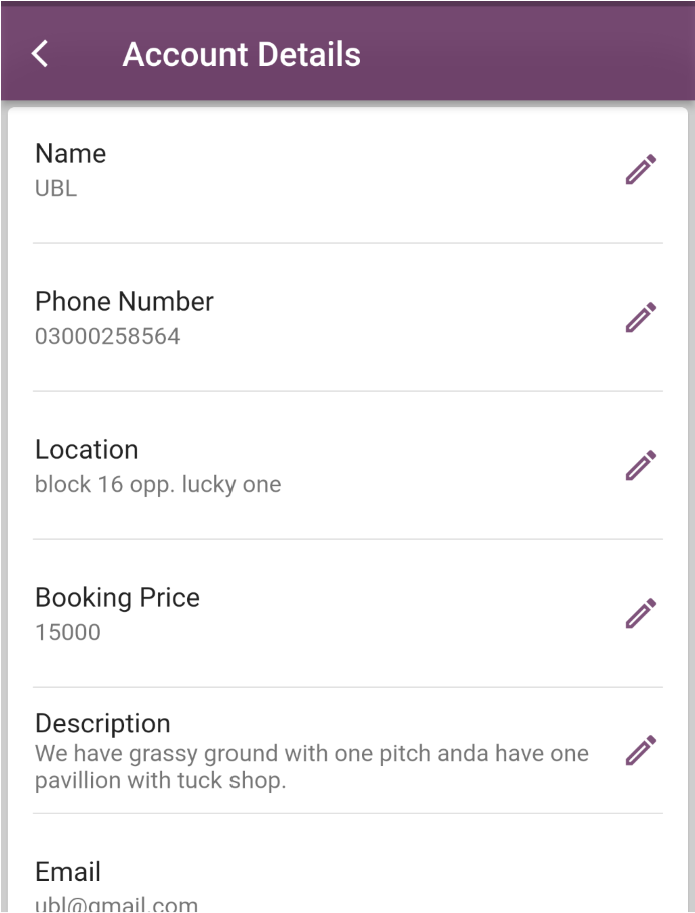


**5.1.5. Ground App Signup**

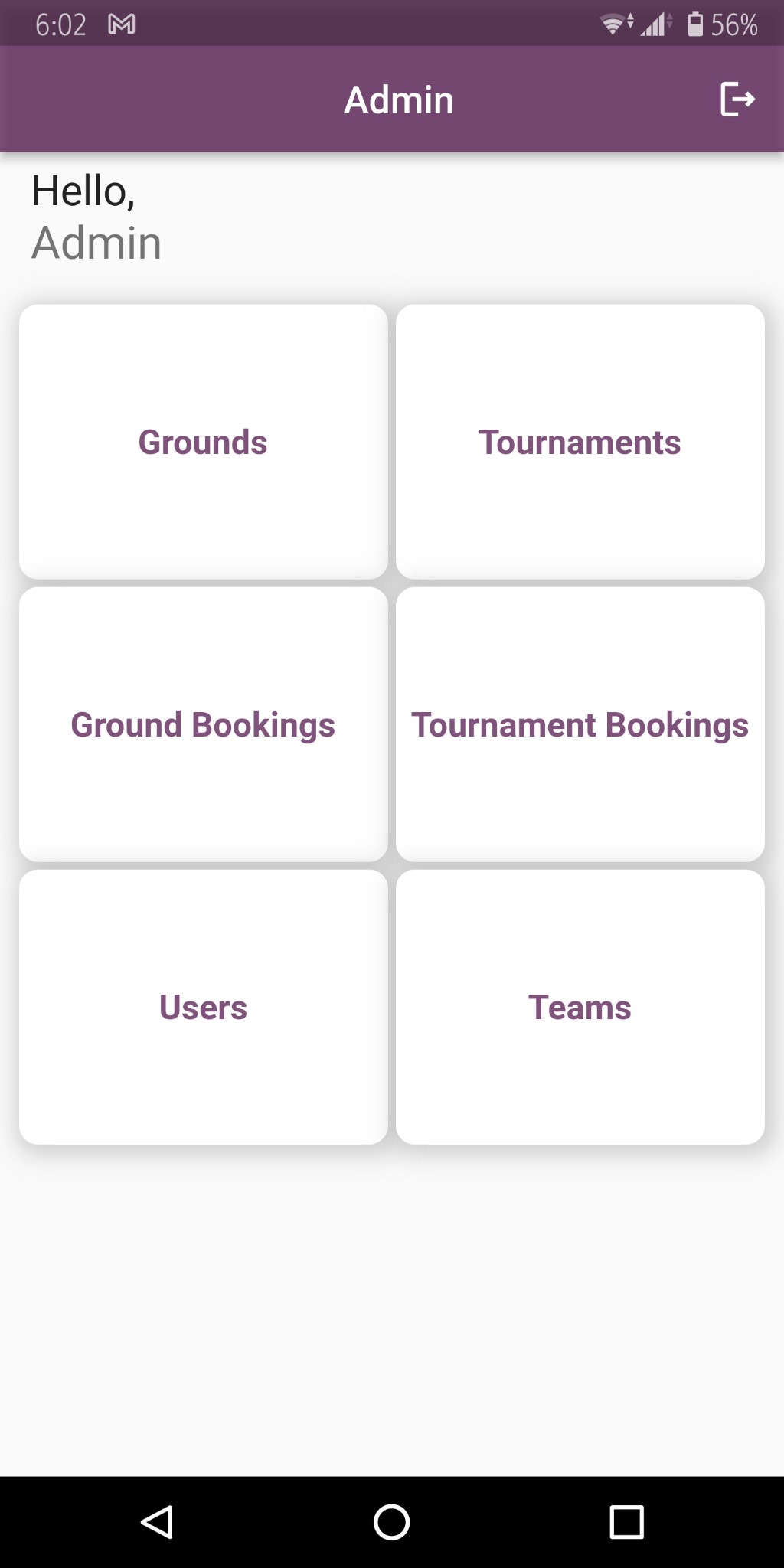


* **5.1.6. Ground App Home**

**5.1.7. Ground App Account**



* **5.1.8. Admin Panel**



## CHAPTER6

### **● 6.1. Conclusion and Future Work**

* **6.1.1. Limitation**

While a cricket ground booking app offers many benefits, it also has some limitations that should be considered:

* **Limited Availability:** The app's effectiveness is dependent on the number of cricket grounds that are registered within its database. If there is a limited number of grounds listed, users may have fewer options and face difficulties finding available grounds, especially during peak times or in certain locations.
* **Geographical Constraints:** The app's usability may be limited to specific regions or countries where it is supported. If the app is not available in certain areas, it may restrict users from accessing its features and services.
* **Ground Information Accuracy:** The app relies on the accuracy and completeness of the information provided by the cricket ground owners or managers. If the information, such as availability, pricing, or amenities, is not up-to-date or incorrect, it may lead to confusion and inconvenience for users.
* **External Factors:** The availability of cricket grounds can be influenced by external factors such as weather conditions, maintenance activities, or unforeseen events. The app may not always reflect real-time changes or cancellations, leading to potential booking conflicts or disappointments for users.
* **Limited Customization:** Users may have specific requirements or preferences for their cricket ground bookings, such as specific time slots, amenities, or customized services. The app may not always provide the flexibility to accommodate these individual preferences, leading to limitations in meeting specific user needs.
* **Payment Processing:** While many cricket ground booking apps incorporate online payment options, there may be limitations or challenges related to payment processing. This could include issues with payment gateways, security concerns, or limited payment options, which may affect the user experience.
* **User Connectivity:** The app's functionality depends on a stable internet connection. If users are in areas with poor network coverage or face connectivity issues, they may struggle to access the app's features, make bookings, or receive updates in real-time.
* **User Adoption:** The success of the app relies on user adoption and engagement. If the target audience, such as local cricket players or organizers, is not aware of the app or reluctant to embrace technology for booking purposes, it may limit the app's usage and impact.
* **6.1.2. Conclusion**

The cricket arena app is a convenient and efficient way for cricket enthusiasts to book and manage their cricket ground reservations. By leveraging modern technology, the app streamlines the process of finding and reserving cricket grounds, eliminating the need for traditional manual booking methods. The app offers several key benefits to its users. Firstly, it provides a comprehensive database of cricket grounds, allowing users to easily search for available grounds based on their location, preferred date, and other criteria. This saves users valuable time and effort in finding suitable grounds for their Cricket activities.

* **6.1.3. Future Works**
* Fantasy League Addition.
* Cricket Products Online Store .
* Enhance User Interface.
* Live Streaming.

### REFERENCES

1. <https://pub.dev/>
2. <https://docs.flutter.dev/deployment/android>

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

4.[https://www.google.com/search?q=stack+overflow&oq=stacko&aqs=chrome.1.69i57j sourceid](https://www.google.com/search?q=stack+overflow&oq=stacko&aqs=chrome.1.69i57j)=chrome&ie=UTF-8

5. <https://cricketdata.org/>

6.<https://github.com/topics/cricket-data>

7. <https://github.com/topics/flutter-calendar>

### APPENDIX

#### A. Coding – Login File

import 'package:firebase\_auth/firebase\_auth.dart'; import 'package:flutter/material.dart'; import 'package:fluttertoast/fluttertoast.dart'; import 'package:shared\_preferences/shared\_preferences.dart'; import 'package:user\_app/Screens/Auth/signUp.dart'; import 'package:user\_app/Screens/adminDashboard.dart'; import 'package:user\_app/Screens/home.dart';

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

@override

\_LoginScreenState createState() => \_LoginScreenState();

}

class \_LoginScreenState extends State<LoginScreen> {

// for password visible or not late bool \_passwordVisible;

@override void initState() {

\_passwordVisible = false;

}

// for password visible or not

// editing controller final TextEditingController emailController = new TextEditingController(); final TextEditingController passwordController = new TextEditingController();

// firebase final \_auth = FirebaseAuth.instance;

// string for displaying the error Message

String? errorMessage;

@override

Widget build(BuildContext context) {

//email field final emailField = Padding( padding: EdgeInsets.symmetric(horizontal: 40), child: TextFormField(

autofocus: false, controller: emailController, keyboardType: TextInputType.emailAddress, validator: (value) { if (value!.isEmpty) { return ("Please Enter Your Email"); }

// reg expression for email validation if (!RegExp("^[a-zA-Z0-9+\_.-]+@[a-zA-Z0-9.-]+.[a-z]")

.hasMatch(value)) { return ("Please Enter a valid email");

} return null;

},

onSaved: (value) { emailController.text = value!;

}, textInputAction: TextInputAction.next, decoration: InputDecoration( enabledBorder:

OutlineInputBorder(borderSide: BorderSide(color: Color.fromARGB(214, 102,

50, 98))), focusedBorder:

OutlineInputBorder(borderSide: BorderSide(color: Color.fromARGB(214, 102,

50, 98))), prefixIcon: Icon(Icons.mail), contentPadding: EdgeInsets.fromLTRB(20, 15, 20, 15), hintText: "Email", border: OutlineInputBorder( borderRadius: BorderRadius.circular(10),

),

),

),

);

//password field final passwordField = Padding( padding: EdgeInsets.symmetric(horizontal: 40), child: TextFormField( autofocus: false, controller: passwordController, obscureText: !\_passwordVisible, validator: (value) {

RegExp regex = new RegExp(r'^.{6,}$'); if (value!.isEmpty) { return ("Password is required for login");

}

if (!regex.hasMatch(value)) { return ("Enter Valid Password(Min. 6 Character)");

}

},

onSaved: (value) { passwordController.text = value!;

}, textInputAction: TextInputAction.done, decoration: InputDecoration( enabledBorder:

OutlineInputBorder(borderSide: BorderSide(color: Color.fromARGB(214, 102,

50, 98))), prefixIcon: Icon(Icons.vpn\_key), contentPadding: EdgeInsets.fromLTRB(20, 15, 20, 15), hintText: "Password", suffixIcon: IconButton( icon: Icon(

// Based on passwordVisible state choose the icon

\_passwordVisible ? Icons.visibility : Icons.visibility\_off, color: Theme.of(this.context).primaryColor,

),

onPressed: () {

// Update the state i.e. toogle the state of passwordVisible variable setState(() {

\_passwordVisible = !\_passwordVisible;

});

},

),

border: OutlineInputBorder( borderRadius: BorderRadius.circular(10),

),

),

),

);

final loginButton = Padding( padding: EdgeInsets.symmetric(horizontal: 40), child: Container( decoration:

BoxDecoration(borderRadius: BorderRadius.circular(10), boxShadow: [

BoxShadow(

color: Colors.grey.withOpacity(0.5), spreadRadius: 3, blurRadius: 10, offset: Offset(2, 2),

) ]),

child: MaterialButton(

onPressed: () { signIn(); }, minWidth: double.infinity, color: Color.fromARGB(214, 102, 50, 98), height: 50, elevation: 0, shape: RoundedRectangleBorder( borderRadius: BorderRadius.circular(10),

),

child: Text( "Login", style: TextStyle( color: Colors.white, fontSize: 18, fontWeight: FontWeight.w600),

),

),

),

);

return Scaffold( resizeToAvoidBottomInset: false, backgroundColor: Colors.white, appBar: AppBar( backgroundColor: Color.fromARGB(214, 102, 50, 98), leading: Icon( Icons.arrow\_back, color: Color.fromARGB(0, 102, 50, 98),

),

title: Center(child: Text('Cricket Arena')), actions: [

Icon(Icons.arrow\_back\_ios\_new\_sharp, color: Color.fromARGB(0, 102, 50, 98)),

Icon(Icons.arrow\_back\_ios\_new\_sharp, color: Color.fromARGB(0, 102, 50, 98)),

],

),

body: SingleChildScrollView( child: Container( height: MediaQuery.of(context).size.height, width: double.infinity, color: Colors.white, child: Column( mainAxisAlignment: MainAxisAlignment.start, children: [

Expanded( child: Column( mainAxisAlignment: MainAxisAlignment.start, children: [

Column( children: <Widget>[Container( // padding: EdgeInsets.only(top: 10), height: 250, decoration: const BoxDecoration( image: DecorationImage( image: AssetImage("assets/calogo4.png"), fit: BoxFit.fitHeight),

),

),

SizedBox( height: MediaQuery.of(context).size.height\*0.02,

),

Text( "Login", style: TextStyle( fontSize: 30, fontWeight: FontWeight.bold),

),

SizedBox( height: MediaQuery.of(context).size.height\*0.01,

),

Text(

"Login to your account",

style:

TextStyle(fontSize: 15, color: Colors.grey[700]),

)

],

),

SizedBox( height: MediaQuery.of(context).size.height\*0.03,

),

emailField,

SizedBox( height: MediaQuery.of(context).size.height\*0.03,

),

passwordField,

SizedBox( height: MediaQuery.of(context).size.height\*0.03,

),

loginButton,

SizedBox( height: MediaQuery.of(context).size.height\*0.03,

),

Row( mainAxisAlignment: MainAxisAlignment.center, children: <Widget>[

Text("Don't have an account?"),

InkWell( child: Text(

" Sign up",

style: TextStyle( fontWeight: FontWeight.w600, fontSize: 18,

),

),

onTap: () {

Navigator.push( context,

MaterialPageRoute( builder: (context) =>

RegistrationScreen()));

},

)

],

),

],

),

),

],

),

),

),

);

}

// login function

Future signIn() async { try { if (emailController.text == 'admin@gmail.com' && passwordController.text == '123456789') { await \_auth.signInWithEmailAndPassword( email: emailController.text, password: passwordController.text);

Navigator.of(context).pushReplacement(

MaterialPageRoute(builder: (context) => AdminDashboard()));

} else { await \_auth.signInWithEmailAndPassword( email: emailController.text, password: passwordController.text); SharedPreferences prefs = await SharedPreferences.getInstance(); prefs.setString("email", emailController.text); prefs.setString("pass", passwordController.text);

Fluttertoast.showToast(msg: "Login Successful");

Navigator.of(context).pushReplacement(

MaterialPageRoute(builder: (context) => Home()));}

} on FirebaseAuthException catch (e) { switch (e.code) { case "invalid-email":

errorMessage = "Your email address appears to be malformed.";

break; case "wrong-password":

errorMessage = "Your password is wrong."; break; case "user-not-found":

errorMessage = "User with this email doesn't exist."; break; case "user-disabled":

errorMessage = "User with this email has been disabled."; break; case "too-many-requests":

errorMessage = "Too many requests"; break; case "operation-not-allowed":

errorMessage = "Signing in with Email and Password is not enabled."; break; default:

errorMessage = "An undefined Error happened.";

}

Fluttertoast.showToast(msg: errorMessage!); print(e.code);

}

}

}

#### B. Gantt chart

