

1. INTRODUCTION

‘BeatMe’ is a website & Android Application that enables users to create tournaments, challenges/Matches. This Application can be used by universities or sport authorities to conduct sporting event and letting all the interested parties know the result and updates in real time. After successfully logging via email/password or through google authenticator, one can create tournaments with a suitable tournament name. enter names of the teams, and the fixtures for those teams will be generated automatically. Users can also create matches where you need to enter the name of the competitors, time etc then the user can update the scores of the teams as well as pause/play the match anytime. Users can even watch matches, fixtures, and challenges. All the data regarding the tournament and matches are updated in real time and available for all the users to watch. Only the creator of the tournament/matches can update the data in them.

We are using Firebase’s Fire store database which provides real time database updating and Firebase Authentication which makes our application completely secure.

The Android Application is written in java and material designs are used, Web Application is build on node.js and React is used for the frontend.

2. RATIONALE & SCOPE

2.1. SCOPE

- It will help to create and organize tournaments in an easy way.
- One can challenge others to compete.
- It will help in easily managing the scores for the teams.
- To fill the gap between traditional and modern approaches for tournament creation.

2.2. PROBLEM

- Gambling
- Match Fixing
- Anyone can change Fixtures
- Hard work needed for fixture creation
- Fixture may be biased

2.3. SOLUTION

- No need to wait for any tournament when you can challenge your friend.
- No biased fixtures
- Easy tournament creation

3. EXISTING SYSTEM

3.1. INTRODUCTION

There are many Mobile Applications and websites which provide us with ways to make Fixtures, tournaments, and challenges, as they help different organizations to create tournaments.

3.2. EXISTING SOFTWARE

- **Challonge**

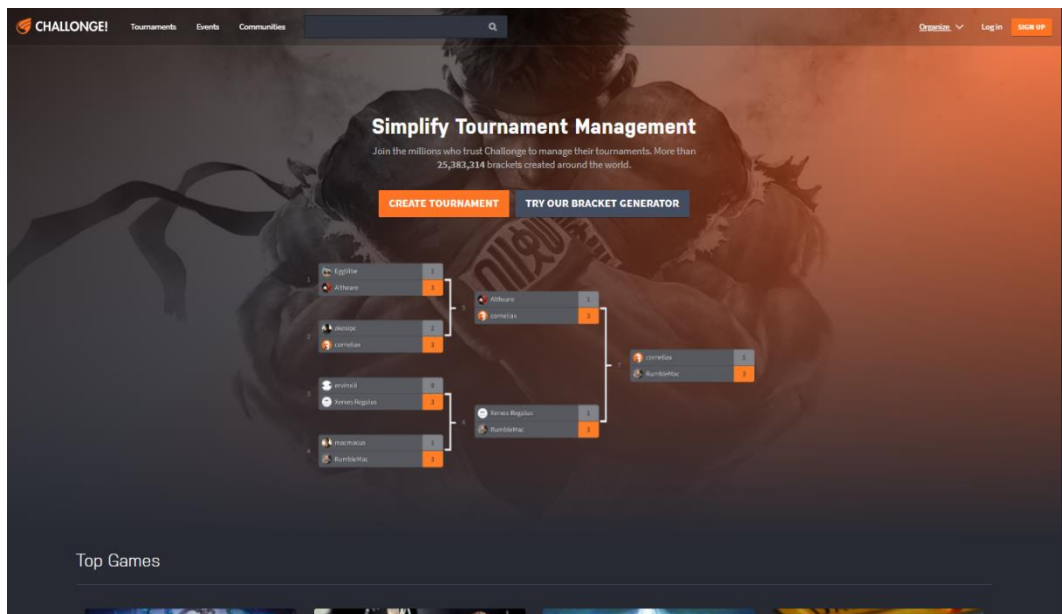


Figure 1 Challonge

<https://challonge.com/>

This is a website where one can create fixtures by giving the team name and they also update the score besides the team name.

keepthescore
[Try it!](#) [About](#)

Most active scoreboards

These are the most active scoreboards of the last 7 days. Click on a scoreboard to take a look!

Note you won't be able to add scores because you'll see the public (read-only) version.

	Sales (created 30 Dec, 2020 / updated 5 days ago).
	LM Cup 2021 (created 31 Mar, 2021 / updated 4 days ago).
	North Division Standings (created 18 Mar, 2021 / updated 12 hours ago).
4	412 Term 2 (created 17 Mar, 2021 / updated Yesterday).
5	Sonday System Points (created 04 Mar, 2021 / updated 4 minutes ago).
6	The Yalta Conference Scoreboard (created 29 Mar, 2021 / updated 23 hours ago).
7	Classroom Points (created 01 Sep, 2020 / updated 49 minutes ago).
8	0000000000000000000000000000.JBSFDDGDHVMVJEJERRHFTCWRYTHFDETYUREU5Y (created 11 Apr, 2021 / updated 2 days ago).
9	P2 தமிழ் வகுப்பு (4000 Points) (created 23 Mar, 2021 / updated 11 hours ago).
10	P4 தமிழ் வகுப்பு (4000 Points) (created 23 Mar, 2021 / updated 10 hours ago).

Figure 2 keep the score

<https://keepthescore.co>

KeepTheScore is a web-based live scoring system. Start recording points for up to 150 teams on your own ranking. This website is ideal for any situation in which you need to keep track of your scores and share them with others. It's perfect for college use, school sports, sales events, and any other situation that needs ease of use and enjoyment.

- **Fixture maker**

Fixture Maker is a program that assists you in organizing leagues when participating in console or PC sporting competitions for your mates. In a team or tournament, you can create as many teams as you want.

https://play.google.com/store/apps/details?id=com.kuarkdijital.footballschedular&hl=en_IN&gl=US



Figure 3 Fixture maker

The screenshot shows the 'Fixture Maker' app interface. At the top, there's a green header with a home icon, the title 'Fixture Maker', and a share icon. Below the header, the text 'My League' is displayed. The fixtures are organized into two sections: '1. Week' and '2. Week'. Each section contains three rows of fixtures, each with two teams and two empty boxes for scores. The teams listed are B. Munchen, R. Madrid, Ac Milan, Juventus, M. United, and Barcelona. At the bottom, there's a green navigation bar with four icons: a calendar, a list, two people, and a clock.

Teams	P	W	D	L	F	A	GD	Pts
1. Juventus	1	1	0	0	3	2	1	3
2. R. Madrid	1	0	1	0	2	2	0	1
3. M. United	1	0	1	0	2	2	0	1
4. Ac Milan	0	0	0	0	0	0	0	0
5. Barcelona	0	0	0	0	0	0	0	0
6. B. Munchen	1	0	0	1	2	3	-1	0

Figure 4 Fixture Maker Tournament

3.3. WHAT'S NEW IN THE SYSTEM TO BE DEVELOPED:

In these Websites and applications, they all have a single module, and we combine all the above ideas into a single project. We have tried to come up with a solution which is highly scalable readily available and easy to use all you need is a internet connection.

Tournament: Only one or two website are have the tournament feature but all the other website or mobile applications are having only tournament module.

Challenges: In this module also website doesn't have the features the challenges but in our project both the website and mobile application are having challenges module.

Real-time: In our project we use real-time database to show the changes to user what creator want to show him in real-time like points and change in the fixture and tournament are visible in real-time to the user.

Time based: All the challenges are of time based user has to ser the time as per the challenge is required.

4. PROBLEM ANALYSIS

4.1. PRODUCT DEFINITION

The BeatMe website and application is designed for everyone and it is free to use and mainly focused on events and tournaments organised by the Student Organization in the university. As other websites and applications do not have all the modules together.

4.2. FEASIBILITY ANALYSIS:

The Feasibility of the project is analysed during this part and the business proposal is place forth with a general arrangement for the project and a few value estimates. During system analysis, the feasibility study of the planned system is to be meted out. This is to make sure that the planned system is not a burden to the corporate. For feasibility analysis, some understanding of the major requirements for the system is essential.

Economic Feasibility:

This study is meted out to see the economic impact can wear the system can the organization. The amount of funds that the corporate will pour into the analysis and development of the system is proscribed. The expenditures must be justified. Thus, the developed system moreover inside the budget, and this was achieved as a result of most of the technologies used are free on the market. We feel our projects economical cost is minimal which makes it easy to be supported.

Technical Feasibility:

This study is meted out to see the technical feasibleness, that is, the technical requirements of the system. Any system developed should not have a high demand on the market technical resources. This will cause high demands to be placed on the shopper. The developed system should have a modest demand, as solely stripped-down or null changes for implementing this technique. All the technologies used in our project are open source and readily available and there is a huge community supporting them.

Operational Feasibility:

The aspect of the study is to check the level of acceptance of the system by the user, this includes the process of training the user to use the system efficiently. The user must not feel threatened by the system, instead must accept it as a necessity. The level of acceptance by the users solely depends on the methods that are employed to educate the user about the system and to make him familiar with it. This level of confidence must be raised so that able to make some constructive criticism, which is welcomed, as the final user of the system.

5. SOFTWARE REQUIREMENT ANALYSIS

5.1. INTRODUCTION

While making this Application we used different software and technology, for developing the project we used Android studio, Visual studio Code and Firebase. For the website, React.js is used with the aid of Visual studio Code and for the backend, firebase is used. For the mobile application, Java programming is used with the help of Android studio and for the backend, we again used Firebase.

5.2. GENERAL DISCRPTION

Android Studio

Based on JetBrains' IntelliJ IDEA software and developed specifically for Android production, Android Studio is the official integrated development environment (IDE) for Google's Android operating system. In 2020, it will be available for use on Windows, macOS, and Linux operating systems, as well as as a subscription-based service. It takes the place of the Eclipse Android Development Tools (E-ADT) as the main IDE for developing native Android apps.

Visual Studio Code

Microsoft's Visual Studio Code is a freeware source-code editor for Windows, Linux, and macOS. Debugging, syntax highlighting, intelligent code completion, snippets, code refactoring, and embedded Git are among the features. Users can customize the theme, keyboard shortcuts, and settings, as well as install plugins that incorporate new features.

Why is Visual Studio Code used in this project?

Visual Studio code is used to create the React Application so we can host locally and see the visually in real time what we are changing in the code.

Firebase

Google Firebase is a platform for developing smartphone and mobile applications. It started out as a stand-alone business in 2011. Google acquired the application in 2014, and it is now their flagship platform for software growth.

Why is firebase used in this project?

Firebase is used to create mainly for real time databases and used for authentication.

React.js

React (also known as React.js or ReactJS) is an open-source front-end JavaScript library for designing user interfaces or UI components. It is maintained by Facebook and a community of independent developers and enterprises. React may be used to create single-page or mobile applications. However, since React is only concerned with state management and rendering the state to the DOM, building React implementations typically necessarily requires the use of external libraries for routing and client-side features.

Why is React used in this Project?

React is mainly used to create the frontend of the website and all the skeleton design of the Website.

CSS

CSS, or Cascading Style Sheets, is a basic programming language designed to make the task of rendering web pages presentable easier. CSS is in charge of a web page's appearance and feel. CSS allows you to change the colour of the text, the font type, the distance between lines, how columns are sized and laid out, what background images or colors are used, interface styles, display changes with various platforms and screen sizes, and a host of other effects.

Why is CSS used in this project?

CSS is used in this project to design the website to look better.

Java Programming language

Java is a class-based, object-oriented programming language with a low number of implementation dependencies. It is a general-purpose programming language designed to allow application developers to write once and run anywhere (WORA), which means that compiled Java code can run on all platforms that support Java without requiring recompilation. Java programs are usually compiled to bytecode, and will run on any Java virtual machine (JVM), independent of underlying device architecture. Java's syntax is close to that of C and C++, but it has less low-level facilities than any of them. The Java runtime has complex functionality (such as reflection and runtime code modification) that standard compiled languages can not. According to GitHub, as of 2019, Java was one of the most common programming languages in use, especially for client-server web applications, with an estimated 9 million developers.

Why is Java used?

Java is used to write the backend code for the android application.

XML

Extensible Markup Language (XML) is a markup language that specifies a set of rules for encoding documents in a machine-readable and human-readable format. XML is defined by the World Wide Web Consortium's XML 1.0 Specification from 1998 and many other similar specifications, all of which are free open standards. XML is used to create the frontend view which a user interacts with.

5.3. SPECIFIC REQUIREMENTS

HARDWARE REQUIREMENTS FOR PRESENT PROJECT:

PROCESSOR: Intel dual Core, i5

RAM: 8GB

HARD DISK: 60 GB

SOFTWARE REQUIREMENTS FOR PRESENT PROJECT:

OPERATING SYSTEM: Windows 7/ 8/10

SOFTWARE: Android Studio, Visual Studio Code

FRONT END: CSS, React.js, Java

DATABASE: Firebase

6. DESIGN

6.1. SYSTEM DESIGN

Introduction to UML:

UML DESIGN: The Unified Modelling Language (UML) is a standard language for specifying, visualizing, constructing, and documenting the software system and its components. It is a graphical language, which provides a vocabulary and set of semantics and rules. The UML focuses on the conceptual and physical representation of the system. It captures the decisions and understandings about systems that must be constructed. It is used to understand, design, configure, maintain, and control information about the systems.

The UML is a language for:

- Visualizing
- Specifying
- Constructing
- Documenting

Visualizing: Through UML we see or visualize an existing system and ultimately, we visualize how the system is going to be after implementation. Unless we think, we cannot implement. UML helps to visualize, how the components of the system communicate and interact with each other.

Specifying: Specifying means building, models that are precise, unambiguous, and complete UML addresses the specification of all the important analysis design, implementation decisions that must be made in developing and deploying a software system.

Constructing: UML models can be directly connected to a variety of programming languages through mapping a model from UML to a programming language like JAVA or C++ or VB. Forward Engineering and Reverse Engineering are possible through UML.

Documenting: The Deliverables of a project apart from coding are some Artifacts, which are critical in controlling, measuring, and communicating about a system during its developing requirements, architecture, design, source code, project plans, tests, prototypes, releases, etc...

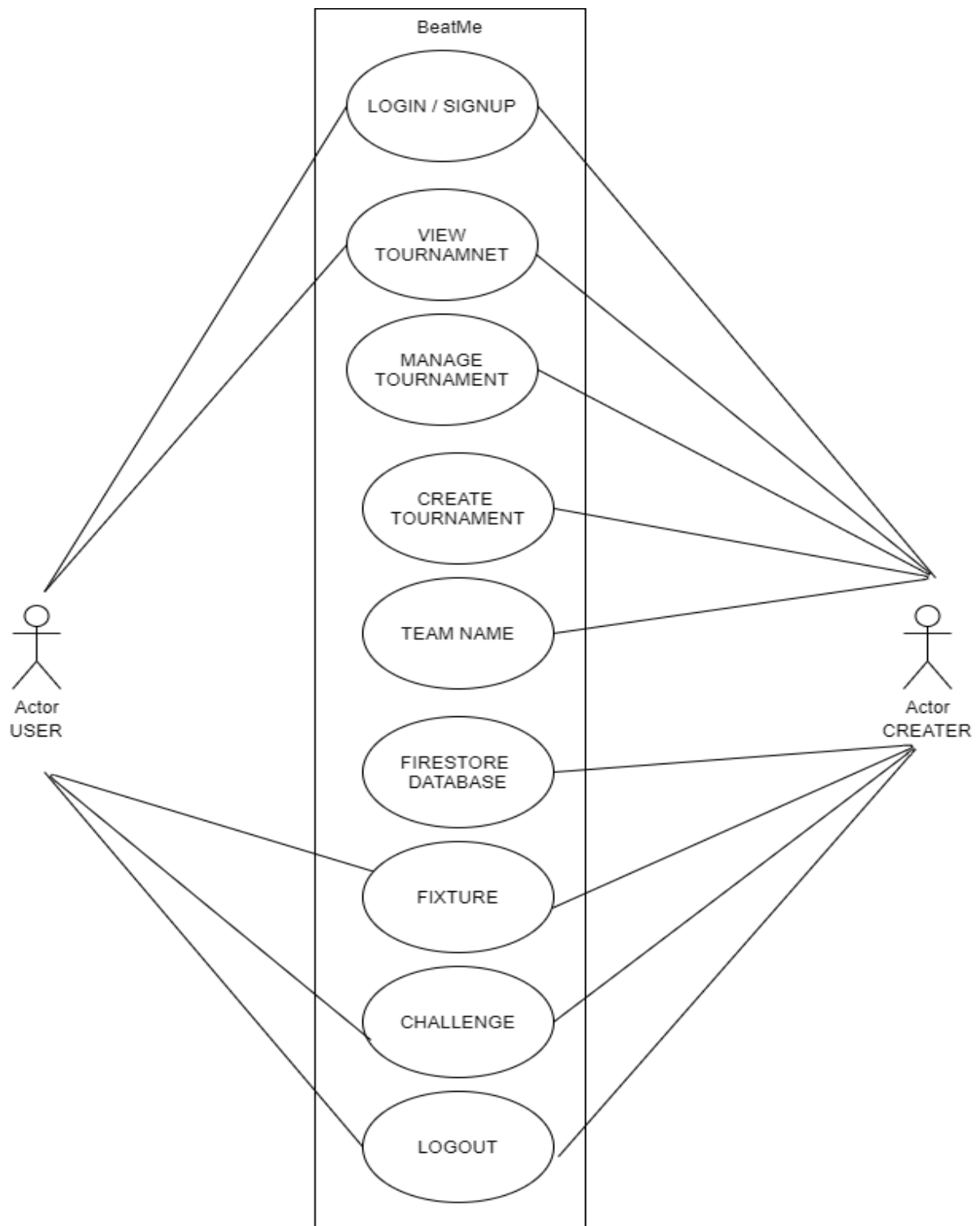
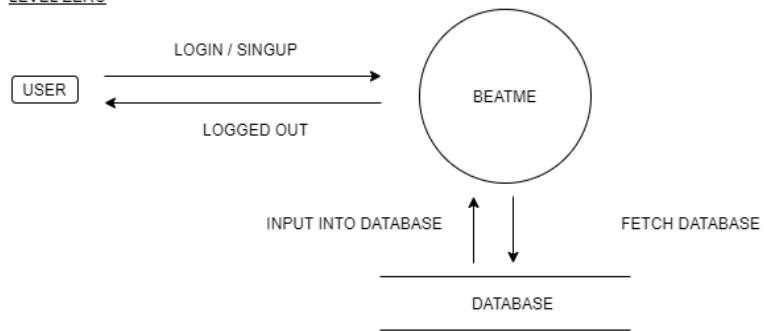


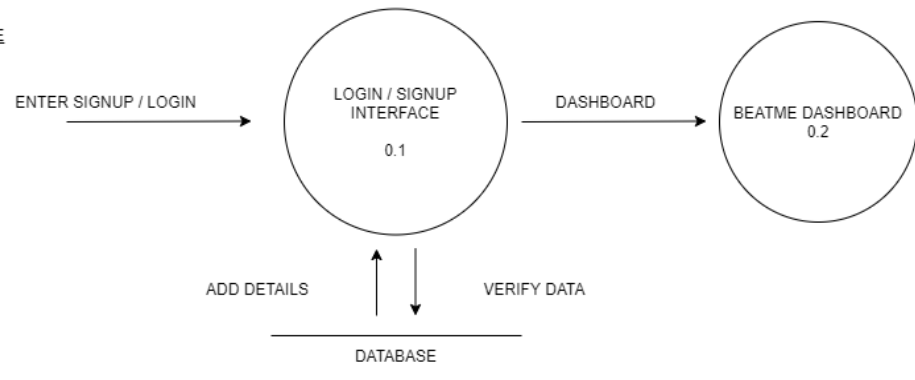
Figure 5 User case diagram

6.2. DATA FLOW DIAGRAM

LEVEL ZERO



LEVEL ONE



LEVEL TWO

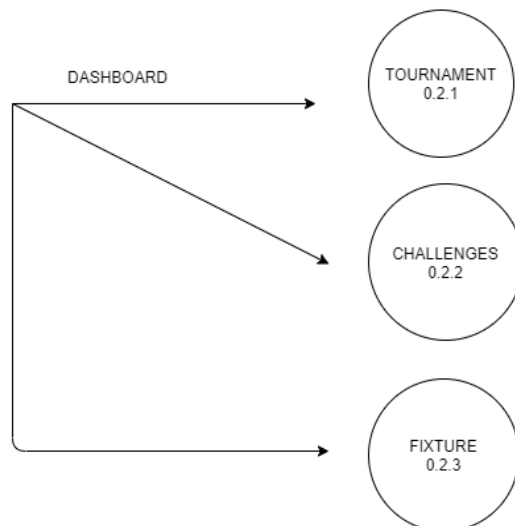


Figure 6 Data flow Diagram Level one and two

LEVEL THREE

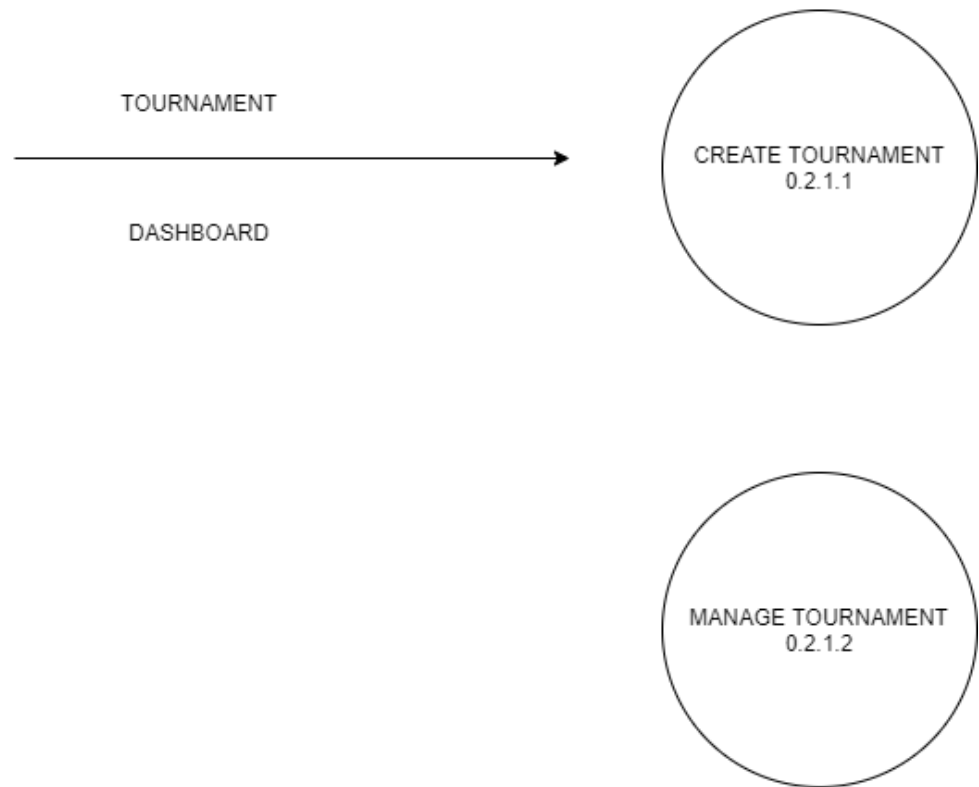


Figure 7 Data flow Diagram Level three

7. TESTING

7.1. FUNCTIONAL TESTING

Functionality Testing of a website is a process which includes several Testing parameters like User interface, APIs, database Testing, Security testing and basic website functionality. We have done manual testing in different functionality of the website.

Functionality testing is centred on the following items:

Valid Input: identify cases of valid input must be accepted.

Invalid Input: Identify cases of invalid input must be rejected.

Output: identify cases of application outputs must be Integrity.

Example:

Login and signup page:

Input 1: Null or left at least one field empty and press the login button.

Expected output: Please fill in the required fields.

Output: Please fill in the required fields.

Input 2: Invalid username or password and press the login button.

Expected output: Enter valid username or password.

Output: Enter valid username or password.

Input 3: Correct combination of username & password and a press login button.

Expected output: Display home page.

Output: Display home page.

API

we have use google firebase API which give us a real time database 24*7 Availability.

User Interface testing:

Application: Test requests are sent correctly to the Firebase Fire store and an Output at the client side displayed correctly.

Database server: All the queries sent to the firebase database give expected results.

Database testing:

Firebase Database is one critical components of any web Application or any Mobile application.

Test if any error is shown while executing queries.

- Data Integrity is maintained while creating, Updating, or deleting data in database.
- Test if any error is shown while executing queries.
- As we are using a real time database response time in minimum.
- data retrieved from the firestone database is shown accurately in your web application and android Application.

Security Testing:

Security testing is vital for this website that all the data are stored in the Google Firebase like Email, password etc.

- Unauthorized access to secure pages should not be permitted.

7.2. TESTING THE PROJECT

All the test cases mentioned above successfully pass some defect still encountered.

Testing Strategy and approach:

All the testing is performed manually, and functionality tests are written in detail in above.

Test Objective:

1. All field entries must work properly.
2. Pages must be activated from the identified link.
3. The entry screen, log-in and responses must not be delayed.

Features to be Tested:

1. Verify that all the entities are in the correct format.
2. No duplicate entries should be allowed.
3. All the links should take the user to the correct page.

8. IMPLEMENTATION

8.1. IMPLEMENTATION OF THE PROJECT

1. Prepare the infrastructure.

Many components are tried and tested for the best use case in this project. Individual components are tried separated and combined into a single project. This strategy involved hardware capability, software, and mode of communication, etc. In our capstone project we used different packages different components to finalize the project, we use firebase for the real time data entry and populate it to the item fields.

2. Coordinate with the organizations involved in implementation.

we communicate with our team numbers and work as a community. however, we try to solve every problem without the help of external organizations. we try every bit of components with more than three times to make it operational, we were bound to check the capability of the infrastructure as we are using free version of the firebase as it can expand automatically according to the need of website and application. we divide our group into different teams so we can work for specific role assigned to them.

3. Implement training.

As we were not aware of all the technologies used in the project, we used to google and search YouTube to learn new things specially about firebase and used StackOverFlow to debug most of bugs we encountered.

4. Database Design.

We are using Firebase Fire store as our data base which is NoSQL database where data is stored in documents which are stored inside collections. The database we designed has taken into consideration the cost of fetching and storing data on cloud and is accordingly optimized.

5. Perform final verification in production.

We have tested the all the components and try to clear all error so all the components can work properly. First, we have checked all the frontend side like all the hooks are

working properly, all the hyperlinks are linked to correct path. Secondly, we have tested all the database side implementation, like storing the data and fetching it, to make sure all the data are properly showing to it correct path.

6. Implement new processes and procedures.

We have constantly tried to implement new features and remove any bug we discover later.

7. Monitor the solution.

We are monitoring Fire store analytics to make sure there is no error and overuse.

9. USER MANUAL

9.1. USER MANUAL OF WEBSITE:

1. When a user wishes to visit a website, he must first log in by entering his username and password and then clicking on the "login" tab.

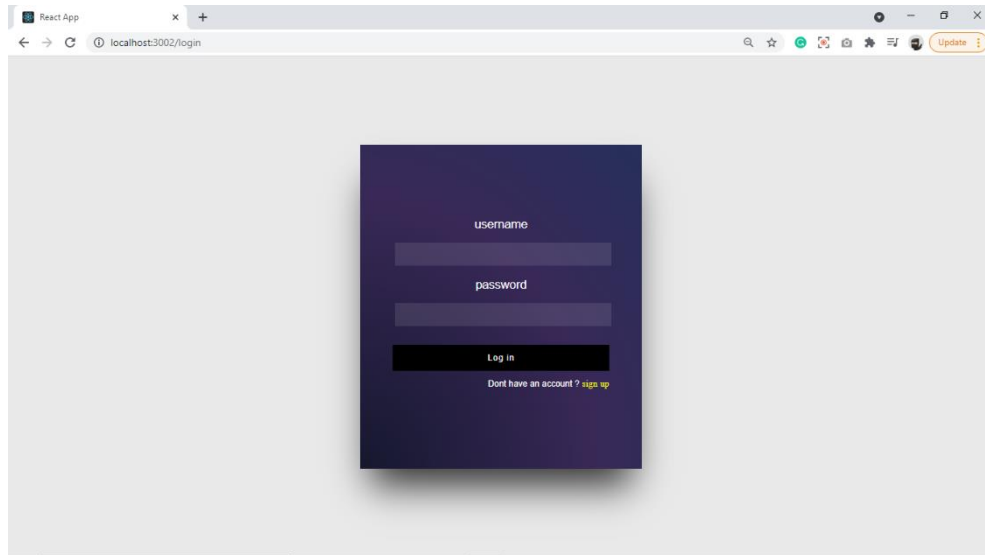


Figure 8 Login Page

2. If the user doesn't have an account, then the user has to create a new account by clicking on "sign up". On clicking "sign up", the following page opens.

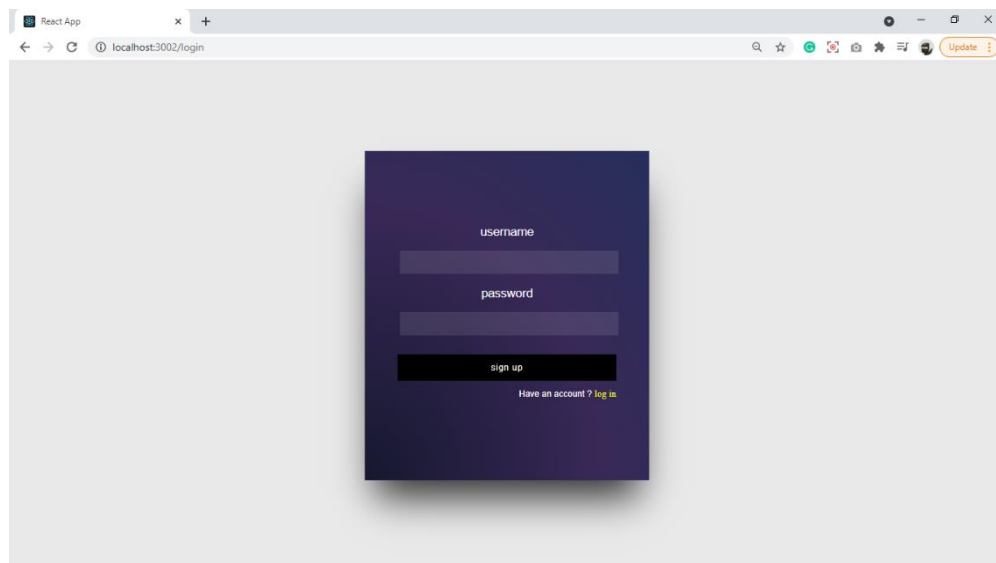


Figure 9 Signup page

3. On logging in, the user sees the “home” page. This is the homepage of the website.

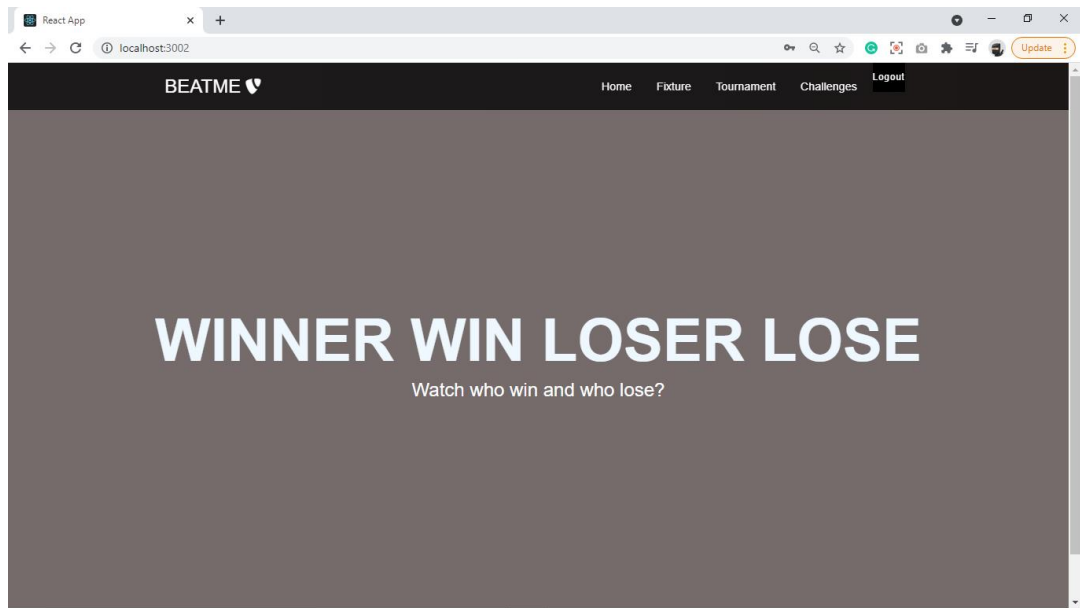


Figure 10 Home page

4. When we click in the “challenges” button, user can set the time of the challenge and then user has to enter the team’s name in each field and submit it.

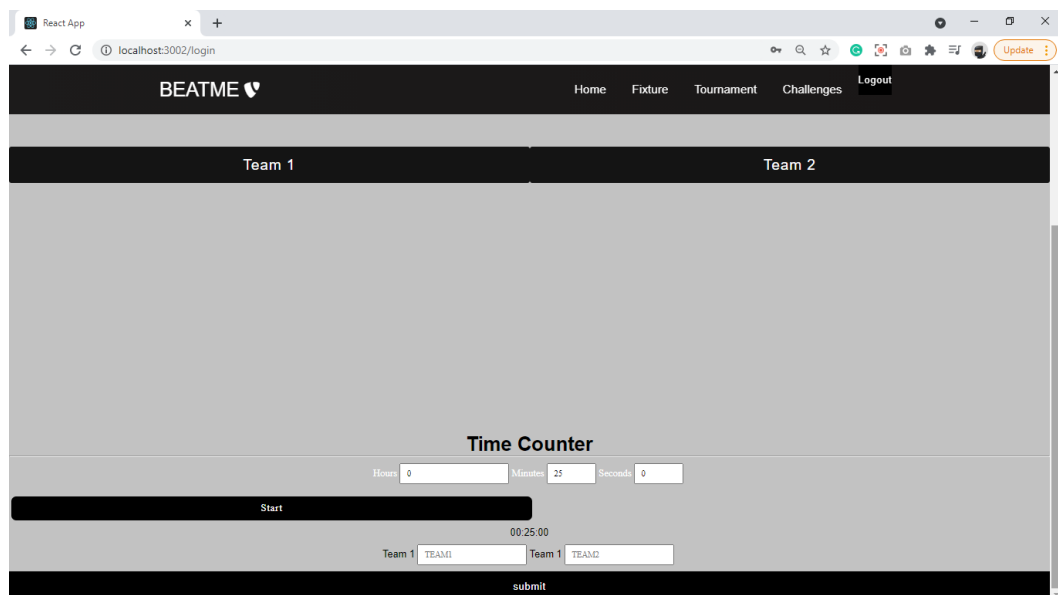


Figure 11 Challenge page

5. Click on any team to give the points, to increase the point click on + button to decrease the point click on – button.

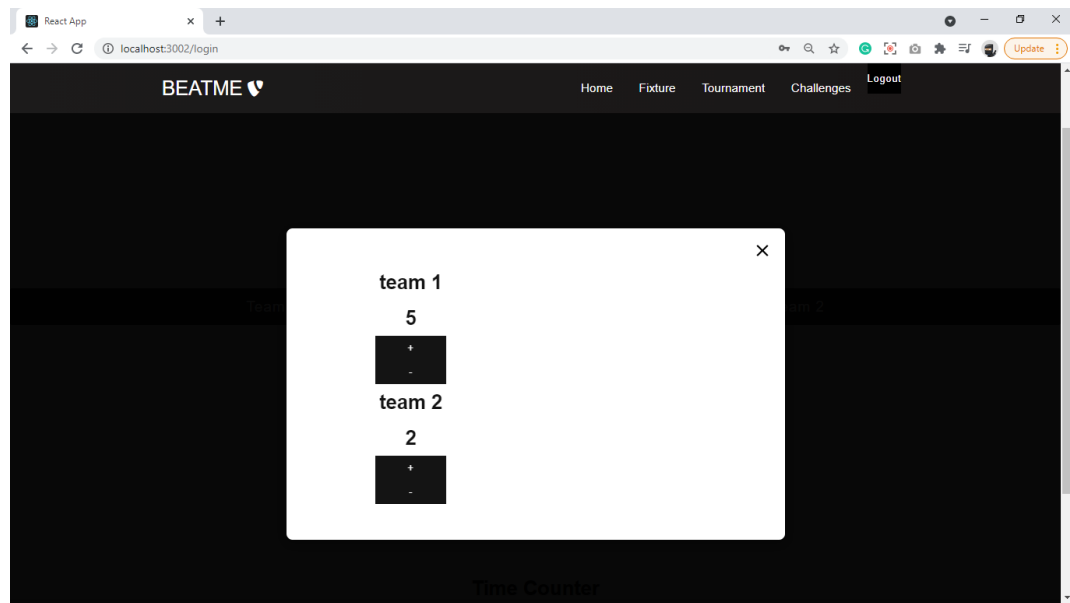


Figure 12 Challenge between both the team

6. Click on the Tournament button to create the tournament, write the Tournament name and click on next.

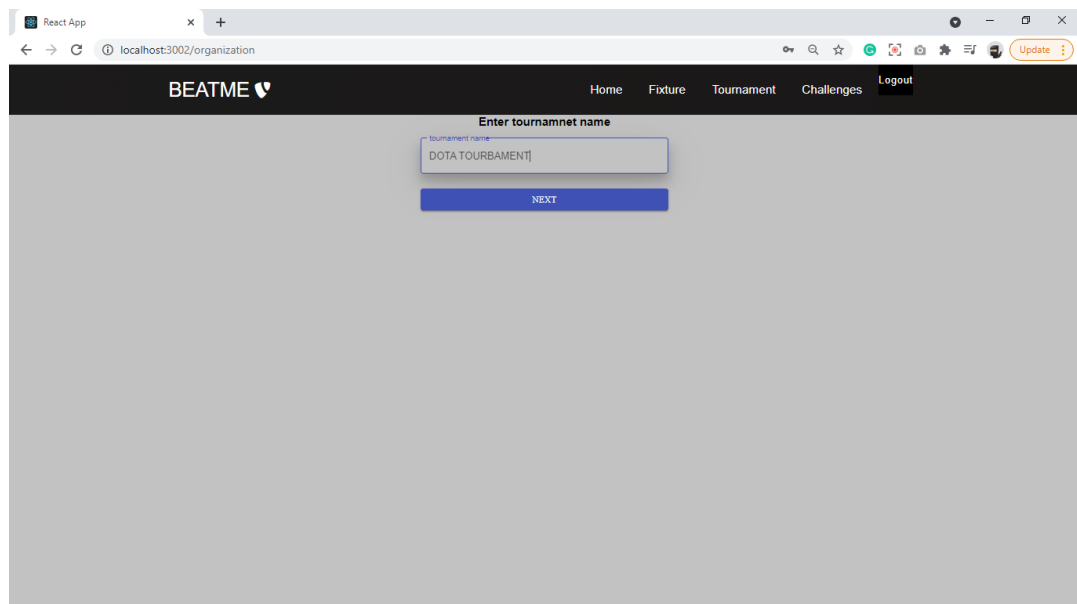


Figure 13 Creating Tournament Name

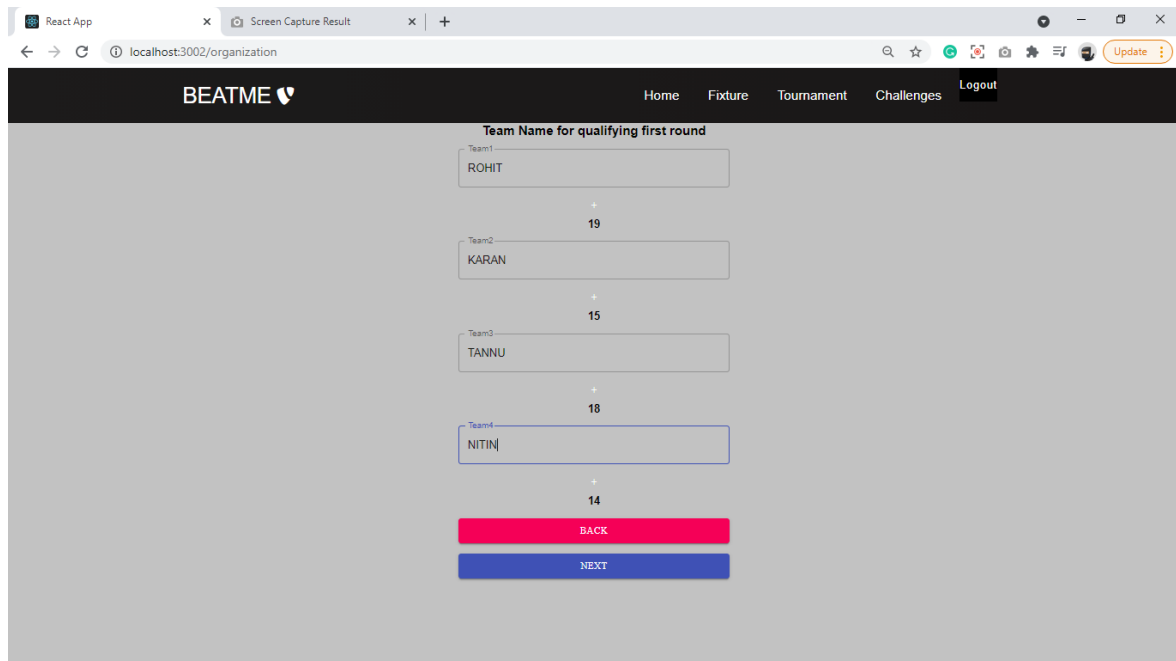
7. Enter the Team names of each team and give the points according to the performance and click on next button.

The screenshot shows the 'BEATME' app interface. At the top, there is a navigation bar with the app name and a heart icon, and links for Home, Fixture, Tournament, Challenges, and Logout. The main content area is titled 'Team Name' and displays a list of 8 teams. Each team entry consists of a label (Team1 to Team8), a text input field containing the team name, and a point value. The teams and their points are: Team1 (ROHIT, 8), Team2 (RAHUL, 4), Team3 (KARAN, 9), Team4 (FAZX, 0), Team5 (TANNU, 9), Team6 (ALICE, 6), Team7 (REX, 2), and Team8 (NITIN, 7). At the bottom of the screen, there are two buttons: a red 'BACK' button and a blue 'NEXT' button.

Team	Name	Points
Team1	ROHIT	8
Team2	RAHUL	4
Team3	KARAN	9
Team4	FAZX	0
Team5	TANNU	9
Team6	ALICE	6
Team7	REX	2
Team8	NITIN	7

Figure 14 Naming teams and giving points

8. Every teams will be challenged with one team then it will be only populated the winning team click on next to see the winning team.



BEATME

Home Fixture Tournament Challenges Logout

Team Name for qualifying first round

Team1
ROHIT
+
19

Team2
KARAN
+
15

Team3
TANNU
+
18

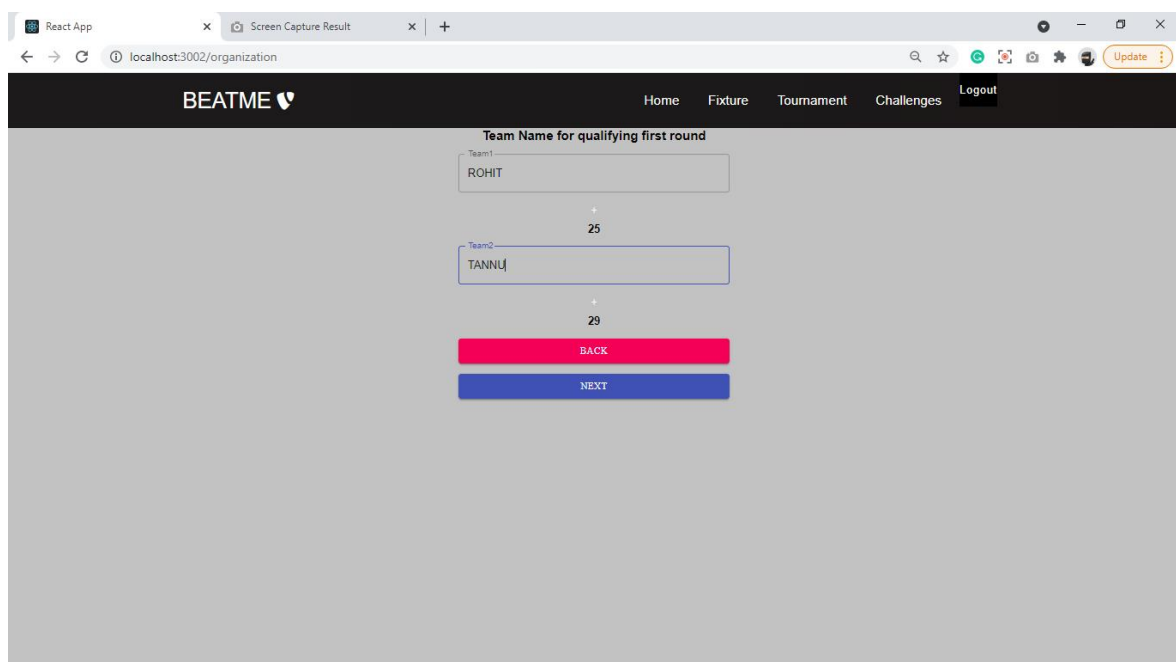
Team4
NITIN
+
14

BACK

NEXT

Figure 15 Entering points of qualified teams

9. Again, every team will be challenged with each other then it will be populated the winning team in next round, click on next button to see the winning team.



BEATME

Home Fixture Tournament Challenges Logout

Team Name for qualifying first round

Team1
ROHIT
+
25

Team2
TANNU
+
29

BACK

NEXT

Figure 16 Semifinal Round

10. Winning team will be displayed click on next.

The screenshot shows a web browser window with the URL `localhost:3002/organization`. The application header is dark with the logo 'BEATME' and a heart icon. Navigation links include 'Home', 'Fixture', 'Tournament', 'Challenges', and 'Logout'. The main content area has a light gray background. A form titled 'Team Name for qualifying first round' is centered. It features a text input field with the value 'TANNU' and a small 'winner' label above it. Below the input field are two buttons: a red 'BACK' button and a blue 'NEXT' button.

Figure 17 Winner

11. One review page will be displayed then click on submit.

The screenshot shows the same web browser window. The 'Review' form is displayed, containing two sections. The first section, titled 'tournamentName', has three input fields with the values 'one', 'two', and 'Submit'. The second section, titled 'winner', has a text input field with the value 'winner, tannu' and a small 'winner' label above it. Below these sections is a large blue 'SUBMIT' button.

Figure 18 Review the points

12. Fixture will be displayed of ongoing tournament.

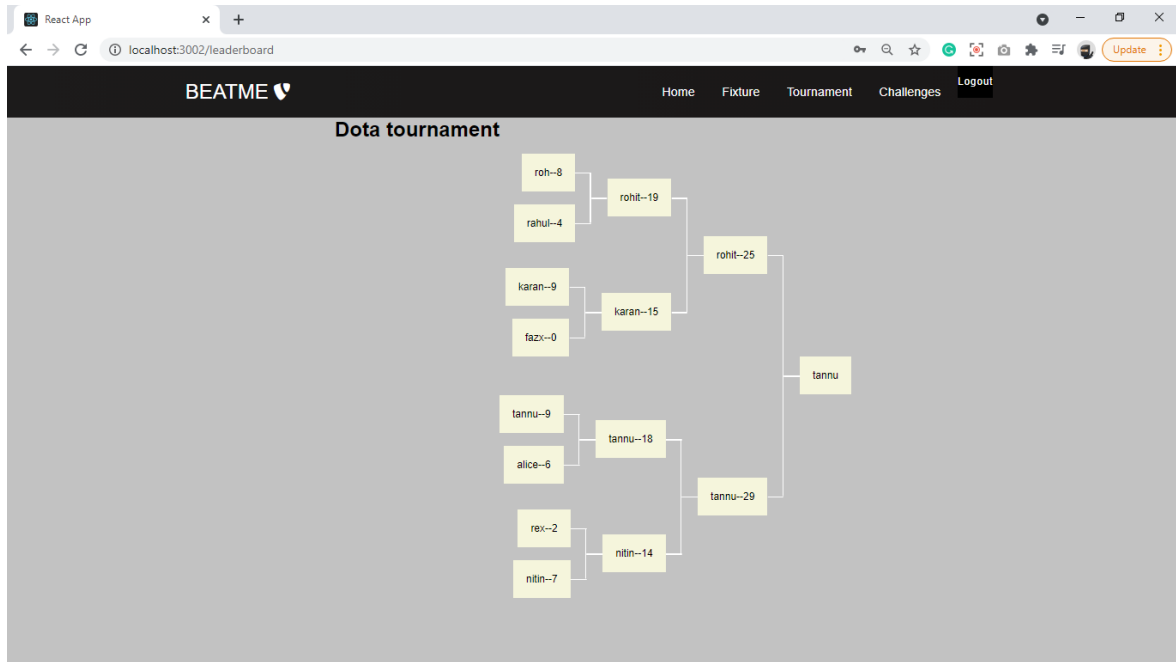


Figure 19 Fixture

13. The match of the finalized teams will be displayed on “Board”. This is the page for viewing “Board”.

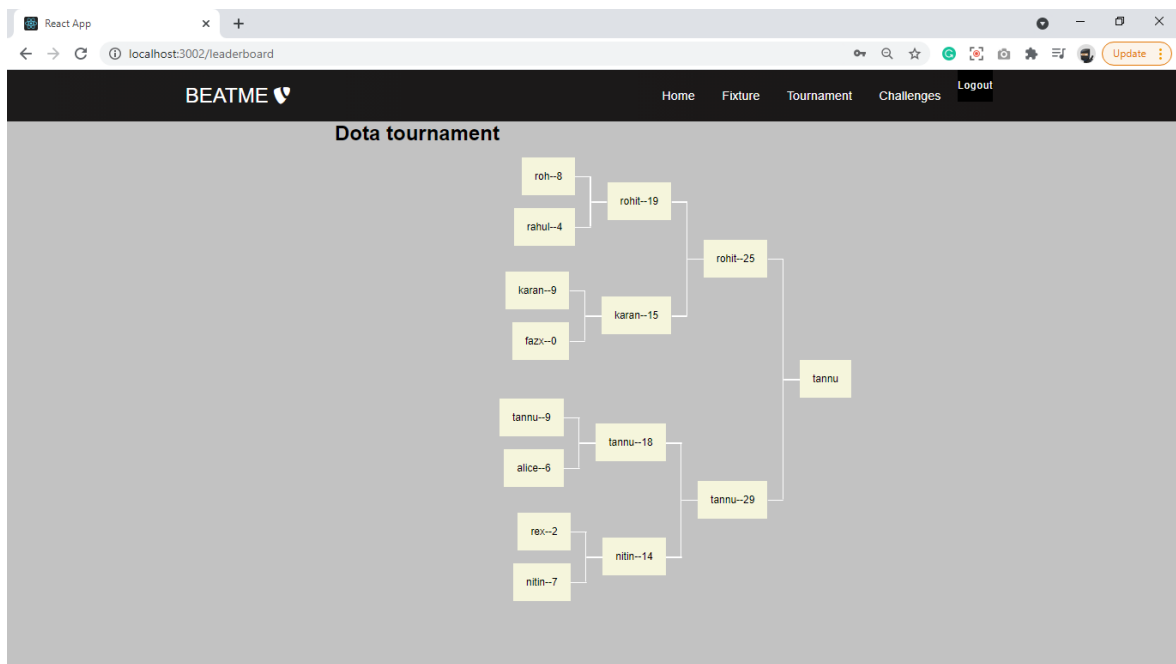


Figure 20 Fixture Page

9.2. USER MANUAL FOR ANDROID APP:

1. Open the App “BeatMe”. If you already have an account then, Enter your ‘Email’ and your ‘Password’ and Login to the App. You can also Login through your Gmail account.

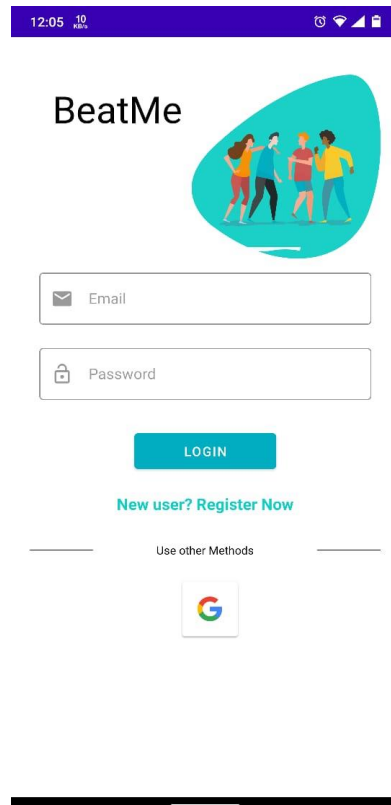
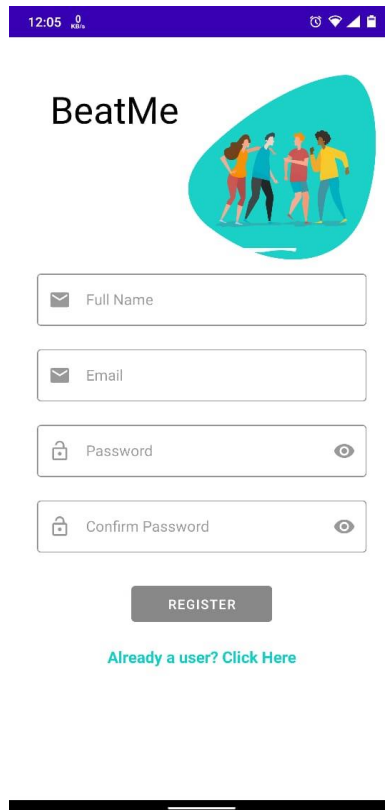


Figure 21 Login Page

2. If you don't have an existing account, 'Click on' the "Register Now". The following page opens. Fill your details and 'click' on "Register".



The screenshot shows a mobile application interface for 'BeatMe'. At the top is a purple status bar with the time '12:05' and various icons. Below the status bar, the app's name 'BeatMe' is displayed in a large, bold, black font. To the right of the name is a circular logo with a teal background, featuring a stylized illustration of four people in various colors (red, blue, yellow, and green) walking or running. Below the logo, there are four input fields stacked vertically. The first field is labeled 'Full Name' with an envelope icon on the left. The second field is labeled 'Email' with an envelope icon on the left. The third field is labeled 'Password' with a lock icon on the left and an eye icon on the right. The fourth field is labeled 'Confirm Password' with a lock icon on the left and an eye icon on the right. Below these fields is a dark grey button with the word 'REGISTER' in white capital letters. Underneath the button is a link that says 'Already a user? Click Here' in teal text. At the bottom of the screen, there is a thick black horizontal bar.

Figure 22 Signup page

3. After you Login, the App home page opens. You can see your account details.

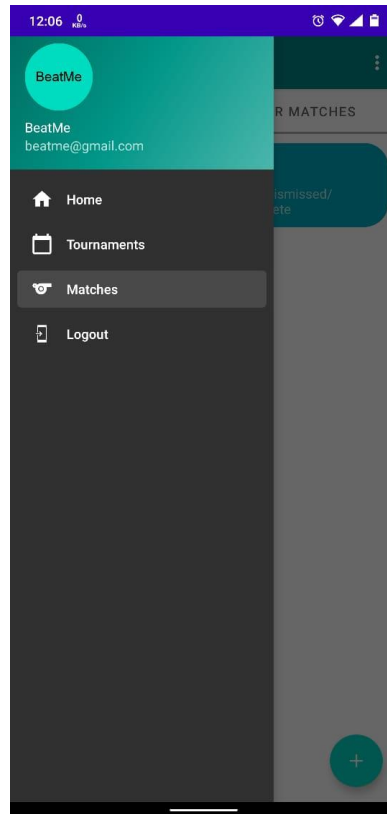


Figure 23 Navigation Bar

4. On opening matches, you can view “All Matches” or “Your Matches” being held . Accordingly a list is displayed.

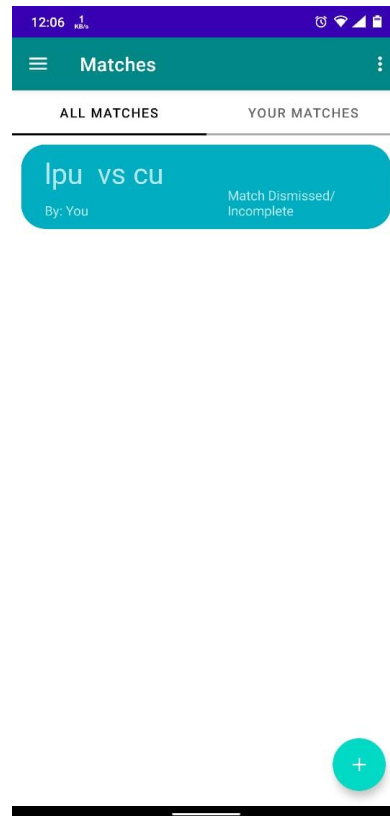


Figure 24 Challenges

5. If you click on “+” which is at the bottom right of the screen, the following page opens. “Add” a new match here along with the new team names and “set Match duration” for each match.

The screenshot shows a mobile application interface for adding a match. At the top, there is a dark green header bar with a hamburger menu icon on the left, the word "Matches" in the center, and a vertical ellipsis icon on the right. Below the header, there are two tabs: "ALL MATCHES" and "YOUR MATCHES". The main content area is a dark grey overlay with a lighter grey form. The form contains two text input fields labeled "Team1" and "Team2". The "Team1" field has the text "Ipu" entered, and the "Team2" field has the text "cu" entered. Below these fields is a section titled "Select Match Duration" which contains a grid of time options: 23, 00, 00, 01, 01, and 02. At the bottom of the form is a white button with the text "ADD". In the bottom right corner of the screen, there is a green circular button with a white plus sign.

Figure 25 Editing the challenges.

6. If you click on any match on the list, the following page opens. This page displays the timer which you can 'play' or 'pause' according to the match. Scores can also be updated here by clicking on the scores given.



Figure 26 Matching

10. SOURCE CODE

10.1. WEB APPLICATION

Complete Source code are available at the mentioned GitHub Repositories:

<https://github.com/Rohitsamad/automatic-guacamole.git>

Login page functionality

App.js

```
import React, {useState, useEffect} from 'react';
import './App.css';
import Login from './components/Auth/Login';
import fire from './components/fire';
import Cap from './Cap';

const App = () => {
  const [user, setUser] = useState("");
  const [email, setEmail] = useState("");
  const [password, setPassword] = useState("");
  const [emailError, setEmailError] = useState("");
  const [passwordError, setPasswordError] = useState("");
  const [hasAccount, setHasAccount] = useState(false);

  const clearInputs = () => {
    setEmail('');
    setPassword('');
  }

  const clearError = () => {
    setEmailError('');
    setPasswordError('');
  }

  const handleLogin = () => {
    clearError();
    fire
      .auth()
      .signInWithEmailAndPassword(email, password)
      .catch((err) => {
        switch (err.code) {
          case "auth/invalid-email":
          case "auth/user-disabled":
```

```

        case "auth/user-not-found":
            setEmailError(err.message);
            break;
        case "auth/wrong-password":
            setPasswordError(err.message);
            break;
    }
});
};

const handlesignup = () => {
    clearError();
    fire
        .auth()
        .createUserWithEmailAndPassword(email, password)
        .catch((err) => {
            switch (err.code) {
                case "auth/email-already-exist":
                case "auth/invalid":
                    setEmailError(err.message);
                    break;
                case "auth/weakpassword":
                    setPasswordError(err.message);
                    break;
            }
        });
};

const handleLogout = () => {
    fire.auth().signOut();
};

const authListener = () => {
    fire.auth().onAuthStateChanged((user) => {
        if(user){
            clearInputs();
            setUser(user);
        }else{
            setUser("");
        }
    });
};

useEffect(() => {
    authListener();
}, []);

return (
    <div className="App">

```

```

    {user ? (
      <Cap handleLogout={handleLogout} />
    ) : (
      <Login
        email={email}
        setEmail={setEmail}
        password={password}
        setPassword={setPassword}
        handleLogin={handleLogin}
        handlesignup={handlesignup}
        hasAccount={hasAccount}
        setHasAccount={setHasAccount}
        emailError={emailError}
        passwordError={passwordError}
      />
    )}
  </div>
);
}

export default App;

```

Login page / signup page

Login.js

```
import React from 'react';

const Login = (props) => {

  const {email, setEmail, password, setPassword, handleLogin, handlesignup, hasAccount, setHasAccount, emailError, passwordError } = props;

  return(
    <section className="login">
      <div className="loginContainer">
        <label>username</label>
        <input
          type="text"
          autoFocus
          required
          value={email}
          onChange={(e) => setEmail(e.target.value)}
        />
        <p className="errorMsg">{emailError}</p>
        <label>password</label>
        <input
          type="password"
          required
          value={password}
          onChange={(e) => setPassword(e.target.value)}
        />
        <p className="errorMsg">{passwordError}</p>
        <div className="btnContainer">
          {hasAccount ? (
            <>
              <button onClick={handleLogin}>Log in</button>
            </>
            Dont have an account ?
            <span onClick={() => setHasAccount(!hasAccount)}>
              >sign up</span>
            </p>
          ) : (
            <>
              <button onClick={handlesignup}>sign up</button>
            </p>
            Have an account ?
            <span onClick={() => setHasAccount(!hasAccount)}>
              >log in</span>
            </p>
          )
        }
      </div>
    </section>
  );
}
```

```
        </>
      })
    </div>
  </div>
</section>
)
}

export default Login;
```


All path ways of login page

Cap.js

```
import React from 'react';
import { BrowserRouter as Router, Switch, Route } from 'react-router-dom';
import './App.css';
import Navbar from './components/Navbar';
import Home from './components/pages/Home';
import LeaderBoard from './components/pages/LeaderBoard';
import Organization from './components/pages/Organization';
import Login from './components/pages/Login';
import Signup from './components/pages/Signup';

function Cap({handleLogout}) {
  return (
    <>
      <Router>
        <Navbar handleLogout={handleLogout} />
        <Switch>
          <Route path="/" exact component={Home} />
          <Route path="/leaderboard" component={LeaderBoard} />
          <Route path="/organization" component={Organization} />
          <Route path="/login" component={Login} />
          <Route path="/signup" component={Signup} />
        </Switch>
      </Router>
    </>
  );
}

export default Cap;
```

Navigation bar

Navbar.js

```
import React, { useState } from 'react';
import { Link } from 'react-router-dom';

import './Navbar.css';

function Navbar({handleLogout}) {
  const [click, setClick] = useState(false);

  const handleClick = () => setClick(!click);
  const closeMobileMenu = () => setClick(false);

  return (
    <>
      <nav className="navbar">
        <div className="navbar-container">
          <Link to="/" className="navbar-
logo" onClick={closeMobileMenu}>
            BEATME <i className='fab fa-typo3' />
          </Link>
          <div className='menu-icon' onClick={handleClick}>
            <i className={click ? 'fas fa-times' : 'fas fa-bars'} />
          </div>
          <ul className={click ? 'nav-menu active' : 'nav-menu'}>
            <li className='nav-item'>
              <Link to="/" className='nav-
links' onClick={closeMobileMenu}>
                Home
              </Link>
            </li>
            <li className='nav-item'>
              <Link to='/leaderboard' className='nav-
links' onClick={closeMobileMenu}>
                Fixture
              </Link>
            </li>
            <li className='nav-item'>
              <Link to='/organization' className='nav-
links' onClick={closeMobileMenu}>
                Tournament
            </li>
          </ul>
        </div>
      </nav>
    </>
  )
}
```

```

        </Link>
      </li>
      <li className='nav-item'>
        <Link to='/login' className='nav-
links' onClick={closeMobileMenu}>
          Challenges
        </Link>
      </li>

      <li className='nav-item'>
        <button onClick={handleLogout} >Logout</button>
      </li>
    </ul>

  </div>
</nav>
</>
)
}

export default Navbar;

```

Fixture

Match.js

```
import React, { useEffect, useState } from 'react';
import '../Apps.css';
import fire from '../fire';

function Match() {
  const [teams, setTeam] = useState([]);

  const ref = fire.firestore().collection("teams");
  console.log(ref);

  function getteam() {
    ref.onSnapshot((querySnapshot) => {
      const items = [];
      querySnapshot.forEach((doc) => {
        items.push(doc.data());
      });
      setTeam(items);
    });
  }

  useEffect(() => {
    getteam();
  }, []);

  return (
    <>
    {teams.map(teams => (

      <div className="wrapper" key={teams.id}>
        <h1>Dota tournament</h1>
      <div className="item">
        <div className="item-parent">
          <p>tannu</p>
        </div>
        <div className="item-childrens">
          <div className="item-child">
            <div className="item">
              <div className="item-parent">
                <p>rohit--25</p>
              </div>
              <div className="item-childrens">
                <div className="item-child">
```

```

    <div className="item">
      <div className="item-parent">
        <p>rohit--19</p>
      </div>
      <div className="item-childrens">
        <div className="item-child">
          <p>{teams.team1}--8</p>
        </div>
        <div className="item-child">
          <p>{teams.team2}--4</p>
        </div>
      </div>
    </div>
  </div>
</div>

<div className="item-child">
  <div className="item">
    <div className="item-parent">
      <p>karan--15</p>
    </div>
    <div className="item-childrens">
      <div className="item-child">
        <p>{teams.team3}--9</p>
      </div>
      <div className="item-child">
        <p>{teams.team4}--0</p>
      </div>
    </div>
  </div>
</div>
</div>

</div>
</div>
</div>
<div className="item-child">
  <div className="item">
    <div className="item-parent">
      <p>tannu--29</p>
    </div>
    <div className="item-childrens">
      <div className="item-child">
        <div className="item">
          <div className="item-parent">
            <p>tannu--18</p>
          </div>
          <div className="item-childrens">

```

```

        <div className="item-child">
          <p>{teams.team5}--9</p>
        </div>
        <div className="item-child">
          <p>{teams.team6}--6</p>
        </div>
      </div>
    </div>
  </div>

  <div className="item-child">
    <div className="item">
      <div className="item-parent">
        <p>nitin--14</p>
      </div>
      <div className="item-childrens">
        <div className="item-child">
          <p>{teams.team7}--2</p>
        </div>
        <div className="item-child">
          <p>{teams.team8}--7</p>
        </div>
      </div>
    </div>
  </div>
</div>
  )))} </>
  );
}

export default Match;

```

Functionality of the tournament pages

Multisteped.js

```
import React from 'react'
import { useForm, useStep } from "react-hooks-helper";
import { One } from './stepForm/One';
import { Review } from './stepForm/Review';
import { Semifinal } from './stepForm/Semifinal';
import { Submit } from './stepForm/Submit';
import { Tournamentname } from './stepForm/Tournamentname';
import { Two } from './stepForm/Two';
import { Winner } from './stepForm/Winner';

const defaultData = {
  tournamentname: "",
  team1: "",
  team2: "",
  team3: "",
  team4: "",
  team5: "",
  team6: "",
  team7: "",
  team8: "",
  team11: "",
  team12: "",
  team13: "",
  team14: "",
  team21: "",
  team22: "",
  winner: ""
};

const steps = [
  { id: 'tournamentname' },
  { id: 'one' },
  { id: 'two' },
  { id: 'semifinal' },
  { id: 'winner' },
  { id: 'review' },
  { id: 'fixture' },
];

export const Multisteped = () => {
  const [formData, setForm] = useForm(defaultData);
  const { step, navigation } = useStep({
    steps,
    initialStep: 0,
  });
};
```

```

const props = { formData, setForm, navigation }

switch(step.id) {
  case 'tournamentname':
    return <Tournamentname { ...props } />;
  case 'one':
    return <One { ...props } />;
  case 'two':
    return <Two { ...props } />;
  case 'semifinal':
    return <Semifinal { ...props } />;
  case 'winner':
    return <Winner { ...props } />;
  case 'review':
    return <Review { ...props } />;
  case 'fixture':
    return <Submit { ...props } />;
}

console.log(step);

return (
  <div>
    <h1>hello</h1>
  </div>
);
};

export default Multisteped;

```


Functionality of timer

Timer.js

```
import React, { useEffect, useState } from 'react';
import './timer.css';

export const Timer = () => {
  const [count, setCount] = useState(100);
  const [second, setSecond] = useState(0);
  const [minute, setMinute] = useState(25);
  const [hours, setHours] = useState(0);
  const [disable, setDisable] = useState(false);
  const [percentage, setPercentage] = useState(50);

  useEffect(
    ()=>{
      let intervalId;
      if(count === 0)
      {
        setDisable(false);
        setMinute(25);
      }
      if(count>0 && disable){
        intervalId=setInterval(()=>{

          const x=count-1;
          const sec=x%60;
          const min=Math.floor(x/60);
          const hour=Math.floor(x/3600);
          setCount(x);
          setMinute(min);
          setHours(hour);
          setSecond(sec);
        },1000);
      }
      return ()=>clearInterval(intervalId)

    },[count,minute,second,hours,disable]

  );

  function handleHours(event){
    setHours(event.target.value);
  }
  function handleMinutes(event){
    setMinute((event.target.value)%60);
  }
}
```

```

function handleSeconds(event){
    setSecond((event.target.value)%60);
}
function handleClick(){
    setDisable(!disable);
    if(!disable)
    {
        const x=second+(minute*60)+(hours*60*60);
        setPercentage(x);
        setCount(x);}
}

return(
    <>
    <div>
        <h1>Time Counter</h1>
        <hr/>
        <div className={disable?'Show':'Hide'}>
            <span>Hours<input type="number" value={hours} onChange={handleHours} min="0"/></span>
            <span>Minutes<input type="number" value={minute} onChange={handleMinutes} min="0" max="59"/></span>
            <span>Seconds<input type="number" value={second} onChange={handleSeconds} min="0" max="59"/></span>
        </div>
        <span><button type="submit" class="x" onClick={handleClick}>
{disable?'Stop':'Start'}</button></span>

        <p>
            {hours.toString().padStart(2,'0')}
            :{minute.toString().padStart(2,'0')}
            :{second.toString().padStart(2,'0')}
        </p>

        <div className="ProgressBar">
            <div className="fill" style={{width:`${(count/percentage)*100}%`}}></div>
        </div>

    </div>
    </>
);
};

export default Timer;

```

Functionality of Challenge

Creation.js

```
import React, { useState } from 'react';
import fire from '../fire';

const Creation = () => {
  const [teamchallenge, setTeamchallenge] = useState("");
  const [teamchallenge1, setTeamchallenge1] = useState("");

  const handleSubmit1 = (e) => {
    e.preventDefault();

    fire.firestore().collection('challenge').add({
      teamchallenge: teamchallenge,
      teamchallenge1: teamchallenge1,
    })
    .then(() => {
      alert('Message has been submitted');
    })
    .catch((error) => {
      alert(error.message);
    });

    setTeamchallenge("");
    setTeamchallenge1("");
  };

  return (
    <form className="" onSubmit={handleSubmit1}>
      <label>Team 1</label>
      <input placeholder="team1" value={teamchallenge} onChange={(e) => setTeamchallenge(e.target.value)} />
      <label>Team 1</label>
      <input placeholder="team2" value={teamchallenge1} onChange={(e) => setTeamchallenge1(e.target.value)} />
      <button type="submit">submit</button>
    </form>
  );
};

export default Creation;
```

10.2. ANDROID APPLICATION:

<https://github.com/nitinpawar1999/BeatMe>

Login & Register Activity

Activity_login.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    android:focusableInTouchMode="true"
    android:gravity="center"
    android:background="@color/white"
    android:layout_gravity="center">

    <com.google.android.material.progressindicator.LinearProgressIndicator
        android:id="@+id/loginProgress"
        android:layout_width="match_parent"
        android:layout_height="5dp"
        app:indicatorColor="@color/teal_700"
        android:indeterminate="true"
        android:visibility="invisible"/>

    <ScrollView
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:layout_alignParentTop="true"
        android:layout_centerInParent="true"
        android:paddingTop="20dp"
        android:layout_alignParentStart="true">
        <LinearLayout
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:orientation="vertical"
            android:padding="20dp">
            <RelativeLayout
                android:layout_width="match_parent"
                android:layout_height="wrap_content">

                <TextView
                    android:layout_width="wrap_content"
                    android:layout_height="wrap_content"
                    android:text="@string/beatme"
                    android:textColor="@color/black"
                    android:layout_margin="10dp"
                    android:textSize="40sp"
                    android:layout_toStartOf="@+id/image"/>

                <ImageView
                    android:id="@+id/image"
                    android:layout_width="wrap_content"
                    android:layout_height="wrap_content"
                    android:src="@drawable/ic_register_hero"
                    android:layout_alignParentTop="true"
                    android:layout_alignParentEnd="true"
                    android:contentDescription="@string/image" />
            </RelativeLayout>
        </LinearLayout>
    </ScrollView>
</RelativeLayout>
```

```

</RelativeLayout>

<com.google.android.material.textfield.TextInputLayout
    android:id="@+id/emailField"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    app:startIconDrawable="@drawable/ic_baseline_email_24"
    android:hint="@string/email"
    android:layout_margin="10dp"
    app:boxStrokeColor="#66000000"
    app:hintTextColor="#66000000"
    android:textColorHint="#66000000"
    app:startIconTint="#66000000"

style="@style/Widget.MaterialComponents.TextInputLayout.OutlinedBox"
    >

    <com.google.android.material.textfield.TextInputEditText
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:textColor="@color/black"
    />

</com.google.android.material.textfield.TextInputLayout>

<com.google.android.material.textfield.TextInputLayout
    android:id="@+id/passwordField"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_margin="10dp"
    app:startIconDrawable="@drawable/ic_baseline_lock_open_24"
    app:endIconMode="password_toggle"
    app:boxStrokeColor="#66000000"
    app:hintTextColor="#66000000"
    android:textColorHint="#66000000"
    app:startIconTint="#66000000"
    android:hint="@string/password"

style="@style/Widget.MaterialComponents.TextInputLayout.OutlinedBox"
    >

    <com.google.android.material.textfield.TextInputEditText
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:textColor="@color/black"
        android:inputType="textPassword"/>

</com.google.android.material.textfield.TextInputLayout>

<Button
    android:id="@+id/loginButton"
    android:layout_width="150dp"
    android:layout_height="50dp"
    android:text="@string/login"
    android:layout_margin="20dp"
    android:layout_gravity="center"
    android:backgroundTint="#00ADC1"

```

```

        android:textColor="@color/white"
    />

    <TextView
        android:id="@+id/registerPage"
        android:layout_height="wrap_content"
        android:layout_width="wrap_content"
        android:textAlignment="center"
        android:textStyle="bold"
        android:textColor="@color/themeColor"
        android:textSize="17sp"
        android:text="@string/new_user_register_now"
        android:layout_gravity="center"/>

    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:weightSum="12"
        android:gravity="center"
        android:layout_marginTop="30dp">

        <View
            android:layout_width="0dp"
            android:layout_height="1dp"
            android:background="@color/colorPrimaryDark"
            android:layout_weight="3"/>

        <TextView
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="@string/use_other_methods"
            android:layout_weight="6"
            android:textSize="12sp"
            android:textAlignment="center"
            android:textColor="@color/black"
            android:gravity="center_horizontal" />

        <View
            android:layout_width="0dp"
            android:layout_height="1dp"
            android:background="@color/colorPrimaryDark"
            android:layout_weight="3"/>

    </LinearLayout>

    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:gravity="center"
        android:layout_marginTop="20dp">
        <ImageView
            android:id="@+id/googleLogin"
            android:layout_width="70dp"
            android:layout_height="70dp"
            android:src="@drawable/btn_google_light_normal_xxxhdpi"
            android:layout_marginStart="10dp"
            android:clickable="true"
            android:onClick="signInWithGoogle"

```

```
        android:contentDescription="@string/googlelogin"
        android:focusable="true" />
    </LinearLayout>

    </LinearLayout>
</ScrollView>

</RelativeLayout>
```

activity_register.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    android:focusableInTouchMode="true"
    android:gravity="center"
    android:background="@color/white"
    android:layout_gravity="center">

    <com.google.android.material.progressindicator.LinearProgressIndicator
        android:id="@+id/registerProgress"
        android:layout_width="match_parent"
        android:layout_height="5dp"
        android:indeterminate="true"
        app:indicatorColor="@color/teal_700"
        android:visibility="invisible"/>

    <ScrollView
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:layout_alignParentTop="true"
        android:layout_centerInParent="true"
        android:paddingTop="20dp"
        android:layout_alignParentStart="true">

        <LinearLayout
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:orientation="vertical"
            android:padding="20dp">

            <RelativeLayout
                android:layout_width="match_parent"
                android:layout_height="wrap_content">

                <TextView
                    android:layout_width="wrap_content"
                    android:layout_height="wrap_content"
                    android:text="@string/beatme"
                    android:textColor="@color/black"
                    android:layout_margin="10dp"
                    android:textSize="40sp"
                    android:layout_toStartOf="@+id/image"/>

                <ImageView
                    android:id="@+id/image"
                    android:layout_width="wrap_content"
                    android:layout_height="wrap_content"
                    android:src="@drawable/ic_register_hero"
                    android:layout_alignParentTop="true"
                    android:layout_alignParentEnd="true"
                    android:contentDescription="@string/image" />

            </RelativeLayout>

            <com.google.android.material.textfield.TextInputLayout
                android:id="@+id/name"
                android:layout_width="match_parent"
                android:layout_height="wrap_content"
                app:startIconDrawable="@drawable/ic_baseline_email_24"
                android:hint="@string/full_name">
```



```

        android:layout_margin="10dp"
        app:boxStrokeColor="#66000000"
        app:hintTextColor="#66000000"
        android:textColorHint="#66000000"
        app:startIconTint="#66000000"

style="@style/Widget.MaterialComponents.TextInputLayout.OutlinedBox"
>

        <com.google.android.material.textfield.TextInputEditText
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:textColor="@color/black"
        />

    </com.google.android.material.textfield.TextInputLayout>

    <com.google.android.material.textfield.TextInputLayout
        android:id="@+id/registerEmail"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        app:startIconDrawable="@drawable/ic_baseline_email_24"
        android:hint="@string/email"
        android:layout_margin="10dp"
        app:boxStrokeColor="#66000000"
        app:hintTextColor="#66000000"
        android:textColorHint="#66000000"
        app:startIconTint="#66000000"

style="@style/Widget.MaterialComponents.TextInputLayout.OutlinedBox"
>

        <com.google.android.material.textfield.TextInputEditText
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:textColor="@color/black"
        />

    </com.google.android.material.textfield.TextInputLayout>

    <com.google.android.material.textfield.TextInputLayout
        android:id="@+id/registerPassword"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_margin="10dp"
        app:startIconDrawable="@drawable/ic_baseline_lock_open_24"
        app:endIconMode="password_toggle"
        app:boxStrokeColor="#66000000"
        app:hintTextColor="#66000000"
        android:textColorHint="#66000000"
        app:endIconTint="#66000000"
        app:startIconTint="#66000000"
        android:hint="@string/password"

style="@style/Widget.MaterialComponents.TextInputLayout.OutlinedBox"
>

        <com.google.android.material.textfield.TextInputEditText

```

```

        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:textColor="@color/black"
        android:inputType="textPassword"/>

</com.google.android.material.textfield.TextInputLayout>

<com.google.android.material.textfield.TextInputLayout
    android:id="@+id/confirmPassword"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_margin="10dp"
    app:startIconDrawable="@drawable/ic_baseline_lock_open_24"
    app:endIconMode="password_toggle"
    app:boxStrokeColor="#66000000"
    app:hintTextColor="#66000000"
    android:textColorHint="#66000000"
    app:startIconTint="#66000000"
    android:hint="@string/confirm_password"
    app:endIconTint="#66000000"

style="@style/Widget.MaterialComponents.TextInputLayout.OutlinedBox"
    >

    <com.google.android.material.textfield.TextInputEditText
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:textColor="@color/black"
        android:inputType="textPassword"/>

</com.google.android.material.textfield.TextInputLayout>

<Button
    android:id="@+id/registerBtn"
    android:layout_width="150dp"
    android:layout_height="50dp"
    android:text="@string/register"
    android:layout_margin="20dp"
    android:layout_gravity="center"
    android:backgroundTint="#00ADC1"

    android:textColor="@color/white"
/>

<TextView
    android:id="@+id/loginPage"
    android:layout_height="wrap_content"
    android:layout_width="wrap_content"
    android:textAlignment="center"
    android:textStyle="bold"
    android:textColor="@color/themeColor"
    android:textSize="17sp"
    android:text="@string/already_a_user_click_here"
    android:layout_gravity="center"/>

</LinearLayout>
</ScrollView>

```

```
</RelativeLayout>
```

LoginActivity.java

```
package com.example.beatme.authentication;

import android.app.Activity;
import android.content.Intent;
import android.os.Bundle;
import android.text.Editable;
import android.text.TextWatcher;
import android.util.Log;
import android.view.View;
import android.view.inputmethod.InputMethodManager;
import android.widget.Button;
import android.widget.TextView;
import android.widget.Toast;

import androidx.appcompat.app.AppCompatActivity;

import com.example.beatme.LandingActivity;
import com.example.beatme.R;
import com.google.android.gms.auth.api.signin.GoogleSignIn;
import com.google.android.gms.auth.api.signin.GoogleSignInAccount;
import com.google.android.gms.auth.api.signin.GoogleSignInClient;
import com.google.android.gms.auth.api.signin.GoogleSignInOptions;
import com.google.android.gms.common.api.ApiException;
import com.google.android.gms.tasks.Task;
import com.google.android.material.progressindicator.LinearProgressIndicator;
import com.google.android.material.textfield.TextInputLayout;
import com.google.firebase.auth.AuthCredential;
import com.google.firebase.auth.FirebaseAuth;
import com.google.firebase.auth.FirebaseUser;
import com.google.firebase.auth.GoogleAuthProvider;

import java.util.Objects;

public class LoginActivity extends AppCompatActivity {

    private static final int RC_SIGN_IN = 9000;
    private FirebaseAuth mAuth;
    private static final String TAG = "loginActivity";
    String emailString;
    String passwordString;
    LinearProgressIndicator progressIndicator;
    GoogleSignInClient mGoogleSignInClient;
    @Override
    protected void onStart() {
        super.onStart();
        FirebaseUser currentUser = mAuth.getCurrentUser();
        if(currentUser != null){
            Toast.makeText(this, "Already Signed in, Happy to see you again"
            +currentUser.getDisplayName(), Toast.LENGTH_SHORT).show();
            updateUser(currentUser);
        }
    }

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
```

```

        setContentView(R.layout.activity_login);
        // Configure Google Sign In
        GoogleSignInOptions gso = new
GoogleSignInOptions.Builder(GoogleSignInOptions.DEFAULT_SIGN_IN)
            .requestIdToken(getString(R.string.default_web_client_id))
            .requestEmail()
            .build();

        mGoogleSignInClient = GoogleSignIn.getClient(this, gso);

        mAuth = FirebaseAuth.getInstance();
        progressIndicator = findViewById(R.id.loginProgress);
        TextInputLayout email = findViewById(R.id.emailField);
        TextInputLayout password = findViewById(R.id.passwordField);
        TextView registerPage = findViewById(R.id.registerPage);
        registerPage.setOnClickListener(v -> {
            Intent intent = new Intent(this, RegisterActivity.class);
            startActivity(intent);
        });

        Objects.requireNonNull(email.getEditText()).addTextChangedListener(new
TextWatcher() {
            @Override
            public void beforeTextChanged(CharSequence s, int start, int count,
int after) {
            }

            @Override
            public void onTextChanged(CharSequence s, int start, int before,
int count) {
                emailString = s.toString();
            }

            @Override
            public void afterTextChanged(Editable s) {
            }
        });

        Objects.requireNonNull(password.getEditText()).addTextChangedListener(new
TextWatcher() {
            @Override
            public void beforeTextChanged(CharSequence s, int start, int count,
int after) {
            }

            @Override
            public void onTextChanged(CharSequence s, int start, int before,
int count) {
                passwordString = s.toString();
            }

            @Override
            public void afterTextChanged(Editable s) {
            }
        });

        Button login = findViewById(R.id.loginButton);

```

```

        login.setOnClickListener(v -> {
            hideKeyboard(LoginActivity.this);
            if(emailString == null || passwordString == null ||
emailString.isEmpty() || passwordString.isEmpty()){
                Toast.makeText(LoginActivity.this, "Please enter
Credentials", Toast.LENGTH_SHORT).show();
            }else{
                login(emailString, passwordString);
            }
        });
    }

    @Override
    public void onActivityResult(int requestCode, int resultCode, Intent data)
    {
        super.onActivityResult(requestCode, resultCode, data);

        // Result returned from launching the Intent from
        GoogleSignInApi.getSignInIntent(...);
        if (requestCode == RC_SIGN_IN) {
            Task<GoogleSignInAccount> task =
            GoogleSignIn.getSignedInAccountFromIntent(data);
            try {
                // Google Sign In was successful, authenticate with Firebase
                GoogleSignInAccount account =
task.getResult(ApiException.class);
                assert account != null;
                Log.d(TAG, "firebaseAuthWithGoogle:" + account.getId());
                firebaseAuthWithGoogle(account.getIdToken());
            } catch (ApiException e) {
                // Google Sign In failed, update UI appropriately
                Log.w(TAG, "Google sign in failed", e);
            }
        }
    }

    private void firebaseAuthWithGoogle(String idToken) {
        AuthCredential credential = GoogleAuthProvider.getCredential(idToken,
null);
        mAuth.signInWithCredential(credential)
            .addOnCompleteListener(this, task -> {
                if (task.isSuccessful()) {
                    // Sign in success, update UI with the signed-in user's
information
                    Log.d(TAG, "signInWithCredential:success");
                    FirebaseUser user = mAuth.getCurrentUser();
                    updateUI(user);
                } else {
                    // If sign in fails, display a message to the user.
                    Log.w(TAG, "signInWithCredential:failure",
task.getException());
                    updateUI(null);
                }
            });
    }
}

```

```

public void updateUI(FirebaseUser user){
    if(user != null) {
        Intent intent = new Intent(this, LandingActivity.class);
        startActivity(intent);
    }
}

public void login(String emailString, String passwordString) {
    progressIndicator.setVisibility(View.VISIBLE);
    mAuth.signInWithEmailAndPassword(emailString, passwordString)
        .addOnCompleteListener(this, task -> {
            if (task.isSuccessful()) {
                Log.d(TAG, "signInWithEmail:success");
                FirebaseUser user = mAuth.getCurrentUser();
                progressIndicator.setVisibility(View.INVISIBLE);
                updateUI(user);
            } else {
                Log.d(TAG, "Signin Failed", task.getException());
                progressIndicator.setVisibility(View.INVISIBLE);
                Toast.makeText(LoginActivity.this, "Authentication
failed. "+ Objects.requireNonNull(task.getException()).getMessage(),
                    Toast.LENGTH_SHORT).show();
                updateUI(null);
            }
        });
}

public void signInWithGoogle(View view) {
    Intent signInIntent = mGoogleSignInClient.getSignInIntent();
    startActivityForResult(signInIntent, RC_SIGN_IN);
}

public static void hideKeyboard(Activity activity) {
    InputMethodManager imm = (InputMethodManager)
activity.getSystemService(Activity.INPUT_METHOD_SERVICE);
    //Find the currently focused view, so we can grab the correct window
token from it.
    View view = activity.getCurrentFocus();
    //If no view currently has focus, create a new one, just so we can grab
a window token from it
    if (view == null) {
        view = new View(activity);
    }
    imm.hideSoftInputFromWindow(view.getWindowToken(), 0);
}
}

```

RegisterActivity.java

```
package com.example.beatme.authentication;

import android.app.Activity;
import android.graphics.Color;
import android.os.Bundle;
import android.text.Editable;
import android.text.TextWatcher;
import android.util.Log;
import android.util.Patterns;
import android.view.View;
import android.view.inputmethod.InputMethodManager;
import android.widget.Button;
import android.widget.TextView;
import android.widget.Toast;

import androidx.annotation.Nullable;
import androidx.appcompat.app.AppCompatActivity;

import com.example.beatme.R;
import com.google.android.material.progressindicator.LinearProgressIndicator;
import com.google.android.material.textfield.TextInputLayout;
import com.google.firebase.auth.FirebaseAuth;
import com.google.firebase.auth.FirebaseUser;
import com.google.firebase.auth.UserProfileChangeRequest;

import java.util.Objects;

public class RegisterActivity extends AppCompatActivity {

    private static final String TAG = "RegisterActivity";
    private FirebaseAuth mAuth;
    private boolean emailValidate = false;
    private boolean passwordValidate = false;
    private boolean confirmPasswordValidate = false;

    @Override
    protected void onCreate(@Nullable Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_register);

        mAuth = FirebaseAuth.getInstance();
        TextInputLayout nameEditText = findViewById(R.id.name);
        TextInputLayout email = findViewById(R.id.registerEmail);
        TextInputLayout password = findViewById(R.id.registerPassword);
        TextInputLayout confirmPassword = findViewById(R.id.confirmPassword);
        Button registerBtn = findViewById(R.id.registerBtn);
        TextView loginPage = findViewById(R.id.loginPage);
        LinearProgressIndicator progressBar =
findViewById(R.id.registerProgress);

        registerBtn.setEnabled(false);
        registerBtn.setBackgroundColor(-7829368);

        Objects.requireNonNull(email.getEditText()).addTextChangedListener(new
TextWatcher() {
            @Override
```



```

        public void beforeTextChanged(CharSequence s, int start, int count,
int after) {

        }

        @Override
        public void onTextChanged(CharSequence s, int start, int before,
int count) {
            if(Patterns.EMAIL_ADDRESS.matcher(s).matches() &&
s.length()>0){
                emailValidate = true;
                email.setErrorEnabled(false);
            }else{
                emailValidate=false;
                email.setError("Invalid Email!");
            }
            if(emailValidate && passwordValidate &&
confirmPasswordValidate) {
                registerBtn.setEnabled(true);
                registerBtn.setBackgroundColor(Color.rgb(0,121,107));
            }
        }

        @Override
        public void afterTextChanged(Editable s) {

        }
    });

Objects.requireNonNull(password.getEditText()).addTextChangedListener(new
TextWatcher() {
    @Override
    public void beforeTextChanged(CharSequence s, int start, int count,
int after) {

    }

    @Override
    public void onTextChanged(CharSequence s, int start, int before,
int count) {
        if(s.length() > 7){
            passwordValidate=true;
            password.setErrorEnabled(false);
        }else{
            passwordValidate=false;
            password.setError("Password must be greater than 8
Character!");
        }
        if(emailValidate && passwordValidate &&
confirmPasswordValidate) {
            registerBtn.setEnabled(true);
            registerBtn.setBackgroundColor(Color.rgb(0,121,107));
        }
    }

    @Override
    public void afterTextChanged(Editable s) {

```

```

    });

Objects.requireNonNull(confirmPassword.getEditText()).addTextChangedListener(new TextWatcher() {
    @Override
    public void beforeTextChanged(CharSequence s, int start, int count,
int after) {

    }

    @Override
    public void onTextChanged(CharSequence s, int start, int before,
int count) {

if(!password.getEditText().getText().toString().equals(s.toString())){
    confirmPassword.setError("Password Doesn't Match");
    confirmPasswordValidate = false;
} else{
    confirmPassword.setErrorEnabled(false);
    confirmPasswordValidate = true;
}

        if(emailValidate && passwordValidate &&
confirmPasswordValidate) {
            registerBtn.setEnabled(true);
            registerBtn.setBackgroundColor(Color.rgb(0,121,107));
        }

    @Override
    public void afterTextChanged(Editable s) {

    }

});

registerBtn.setOnClickListener(v -> {
    progressBar.setVisibility(View.VISIBLE);
    hideKeyboard(RegisterActivity.this);
    String name =
Objects.requireNonNull(nameEditText.getEditText().getText().toString());
    if(name.isEmpty()) name = "default_name";
    String emailString = email.getEditText().getText().toString();
    String passwordString =
password.getEditText().getText().toString();

    String finalName = name;
    mAuth.createUserWithEmailAndPassword(emailString,
passwordString).addOnCompleteListener(this, task -> {
        if(task.isSuccessful()) {
            Log.d(TAG, "createUserWithEmail:success");
            FirebaseUser user = mAuth.getCurrentUser();
            UserProfileChangeRequest profileChangeRequest = new
UserProfileChangeRequest
                .Builder().setDisplayName(finalName).build();
            assert user != null;

```

```

user.updateProfile(profileChangeRequest).addOnCompleteListener(task1 -> {
    if(task1.isSuccessful()) {
        Log.d(TAG, "User name added.");
        Toast.makeText(RegisterActivity.this, "User
Registration Success...Redirecting to Login Page",
            Toast.LENGTH_SHORT).show();
        mAuth.signOut();
        onBackPressed();
    }
});

}else {
    // If sign in fails, display a message to the user.
    progressBar.setVisibility(View.INVISIBLE);
    Log.w(TAG, "createUserWithEmail:failure",
task.getException());
    Toast.makeText(RegisterActivity.this, "User Registration
failed.\n"+ Objects.requireNonNull(task.getException()).getMessage(),
        Toast.LENGTH_SHORT).show();
    }
});
loginPage.setOnClickListener((View v) -> onBackPressed());
}

public static void hideKeyboard(Activity activity) {
    InputMethodManager imm = (InputMethodManager)
activity.getSystemService(Activity.INPUT_METHOD_SERVICE);
    //Find the currently focused view, so we can grab the correct window
token from it.
    View view = activity.getCurrentFocus();
    //If no view currently has focus, create a new one, just so we can grab
a window token from it
    if (view == null) {
        view = new View(activity);
    }
    imm.hideSoftInputFromWindow(view.getWindowToken(), 0);
}
}

```

Match/ Challenge Activity:

fragment_matches.xml :

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:background="@color/white"
    tools:context=".ui.matches.MatchesFragment">

    <com.google.android.material.tabs.TabLayout
        android:id="@+id/matchesTabLayout"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        style="@style/Widget.MaterialComponents.TabLayout.Colored"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintBottom_toTopOf="@id/matches_swipe_layout"
        >
        <com.google.android.material.tabs.TabItem
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="@string/all_matches"
            />

        <com.google.android.material.tabs.TabItem
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="@string/your_matches"
            />
    </com.google.android.material.tabs.TabLayout>

    <TableRow
        android:background="#52000000"
        android:layout_height="2dp"
        android:layout_width="match_parent"
        app:layout_constraintTop_toBottomOf="@id/matchesTabLayout"
        app:layout_constraintBottom_toTopOf="@id/matches_load_progress"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        />

    <com.google.android.material.progressindicator.LinearProgressIndicator
        android:id="@+id/matches_load_progress"
        android:layout_width="match_parent"
        android:layout_height="5dp"
        app:indicatorColor="@color/teal_700"
        android:indeterminate="true"
        android:visibility="invisible"
        app:layout_constraintTop_toBottomOf="@id/matchesTabLayout"
        app:layout_constraintBottom_toTopOf="@id/matches_swipe_layout"/>

    <androidx.swiperefreshlayout.widget.SwipeRefreshLayout
        android:id="@+id/matches_swipe_layout"
        android:layout_width="match_parent"
```

```

        android:layout_height="0dp"
        app:layout_constraintTop_toBottomOf="@id/matchesTabLayout"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintBottom_toBottomOf="parent">
        <androidx.recyclerview.widget.RecyclerView
            android:id="@+id/matches_recycler_view"
            android:layout_width="match_parent"
            android:layout_height="0dp"
            app:layout_constraintTop_toBottomOf="@+id/matchesTabLayout"
            app:layout_constraintBottom_toBottomOf="parent"
            app:layout_constraintEnd_toEndOf="parent"
            app:layout_constraintStart_toStartOf="parent"
            android:scrollbarAlwaysDrawVerticalTrack="true"
            android:scrollbars="vertical"
            />

    </androidx.swiperefreshlayout.widget.SwipeRefreshLayout>

    <com.google.android.material.floatingactionbutton.FloatingActionButton
        android:id="@+id/matches_floating_action_button"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_gravity="bottom|end"
        android:layout_margin="16dp"
        android:contentDescription="@string/add_new_match"
        app:srcCompat="@drawable/ic_baseline_add_24"
        app:layout_constraintRight_toRightOf="parent"
        app:layout_constraintBottom_toBottomOf="parent"/>

</androidx.constraintlayout.widget.ConstraintLayout >

```

fragment_match_view.xml

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    xmlns:app="http://schemas.android.com/apk/res-auto">
    <Button
        android:id="@+id/match_view_close"
        android:layout_width="40dp"
        android:layout_height="wrap_content"
        android:text="@string/x"
        android:textColor="@color/white"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintBottom_toTopOf="@id/match_view_label"
        android:backgroundTint="@color/black"
        android:layout_margin="10dp"
    />
    <TextView
        android:id="@+id/match_view_label"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="@string/team_a_vs_team_b"
        android:textSize="30sp"
        app:layout_constrainedWidth="true"
        android:textAlignment="center"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintBottom_toTopOf="@id/match_view_timer"
        android:layout_marginTop="60dp"
    />
    <TextView
        android:id="@+id/match_view_timer"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        app:layout_constrainedWidth="true"
        android:text="@string/_00_00_00"
        android:textSize="35sp"
        android:gravity="center_horizontal"
        app:layout_constraintTop_toBottomOf="@id/match_view_label"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintBottom_toTopOf="@id/match_view_horizontal_line"
        android:layout_marginTop="20dp"
        app:layout_constraintEnd_toEndOf="parent"/>
    <LinearLayout
        android:id="@+id/match_view_linear_Layout1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        app:layout_constraintTop_toBottomOf="@id/match_view_timer"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintEnd_toStartOf="@id/match_view_horizontal_line"
        android:orientation="vertical"
        android:gravity="center"
        android:layout_margin="50dp">
        <TextView
```

```

        android:id="@+id/match_view_team1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="@string/team_a"
        android:textSize="30sp"/>
    <TextView
        android:id="@+id/match_view_score_team1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="@string/_0"
        android:layout_marginTop="20dp"
        android:textSize="30sp"
    />
</LinearLayout>

<View
    android:id="@+id/match_view_horizontal_line"
    android:layout_width="2dp"
    android:layout_height="200dp"
    android:background="@color/black"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintTop_toBottomOf="@id/match_view_timer"
    android:layout_marginTop="20dp"
/>

<LinearLayout
    android:id="@+id/match_view_linear_Layout2"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    app:layout_constraintTop_toBottomOf="@id/match_view_timer"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintStart_toEndOf="@id/match_view_horizontal_line"
    android:orientation="vertical"
    android:gravity="center"
    android:layout_margin="50dp">
    <TextView
        android:id="@+id/match_view_team2"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="@string/team_b"
        android:textSize="30sp"/>
    <TextView
        android:id="@+id/match_view_score_team2"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="@string/_0"
        android:layout_marginTop="20dp"
        android:textSize="30sp"
    />
</LinearLayout>

<TextView
    android:id="@+id/match_view_by"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_marginStart="10dp"
    app:layout_constraintTop_toBottomOf="@id/match_view_horizontal_line"
    app:layout_constraintStart_toStartOf="parent"

```

```
        app:layout_constraintEnd_toEndOf="parent"
        android:layout_marginTop="20dp"
        android:text="@string/by_user"
        android:textSize="20sp"
        android:layout_margin="20dp"/>
    <TextView
        android:id="@+id/match_view_status"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_marginStart="10dp"
        app:layout_constraintTop_toBottomOf="@id/match_view_by"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        android:layout_marginTop="20dp"
        android:text="@string/status"
        android:textSize="20sp"
        android:layout_margin="20dp"/>

</androidx.constraintlayout.widget.ConstraintLayout>
```


MatchesFragment.java

```
package com.example.beatme.ui.matches;

import android.os.Bundle;
import android.util.Log;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.Toast;

import androidx.annotation.NonNull;
import androidx.fragment.app.Fragment;
import androidx.recyclerview.widget.LinearLayoutManager;
import androidx.recyclerview.widget.RecyclerView;
import androidx.swiperefreshlayout.widget.SwipeRefreshLayout;

import com.example.beatme.R;
import com.google.android.material.floatingactionbutton.FloatingActionButton;
import com.google.android.material.progressindicator.LinearProgressIndicator;
import com.google.android.material.tabs.TabLayout;
import com.google.firebase.auth.FirebaseAuth;
import com.google.firebase.auth.FirebaseUser;
import com.google.firebase.firestore.FirebaseFirestore;
import com.google.firebase.firestore.QueryDocumentSnapshot;

import org.jetbrains.annotations.NotNull;

import java.util.ArrayList;
import java.util.List;
import java.util.Objects;

public class MatchesFragment extends Fragment implements
SwipeRefreshLayout.OnRefreshListener {

    private static final String TAG = "MatchFragment";
    View root;
    MatchAdapter matchAdapter;
    RecyclerView matchRecyclerView;
    SwipeRefreshLayout swipeRefreshLayout;
    FirebaseUser user;
    FirebaseFirestore db;
    LinearProgressIndicator linearProgressIndicator;
    int currentTab = 0;

    public View onCreateView(@NonNull LayoutInflater inflater, ViewGroup
container, Bundle savedInstanceState) {

        db = FirebaseFirestore.getInstance();
        user = FirebaseAuth.getInstance().getCurrentUser();
        root = inflater.inflate(R.layout.fragment_matches, container, false);
        FloatingActionButton matchesFb =
root.findViewById(R.id.matches_floating_action_button);
        TabLayout matchesTL = root.findViewById(R.id.matchesTabLayout);
        linearProgressIndicator =
root.findViewById(R.id.matches_load_progress);
        matchRecyclerView = root.findViewById(R.id.matches_recycler_view);
        swipeRefreshLayout = root.findViewById(R.id.matches_swipe_layout);
```

```

        swipeRefreshLayout.setOnRefreshListener(this);

swipeRefreshLayout.setColorSchemeColors(getResources().getColor(android.R.color
.black),
        getResources().getColor(android.R.color.white),
        getResources().getColor(android.R.color.darker_gray),
        getResources().getColor(android.R.color.white));
getAllMatches();

matchesTL.addTabSelectedListener(new
TabLayout.OnTabSelectedListener() {

    @Override
    public void onTabSelected(TabLayout.Tab tab) {
        if (tab.getPosition() == 0) {
            currentTab = 0;
            getAllMatches();
        }
        if (tab.getPosition() == 1) {
            currentTab = 1;
            getCurrentUsersMatches();
        }
    }

    @Override
    public void onTabUnselected(TabLayout.Tab tab) {
    }

    @Override
    public void onTabReselected(TabLayout.Tab tab) {
        if (tab.getPosition() == 0) {
            currentTab = 0;
            getAllMatches();
        }
        if (tab.getPosition() == 1) {
            currentTab = 1;
            getCurrentUsersMatches();
        }
    }
});

matchesFb.setOnClickListener(v -> {
    AddMatchFragment addMatchFragment = new AddMatchFragment();
    addMatchFragment.show(getParentFragmentManager(),
"dialogAddMatch");
});

return root;
}

@Override
public void onRefresh() {
    if (currentTab == 0) getAllMatches();
    if (currentTab == 1) getCurrentUsersMatches();
    swipeRefreshLayout.setRefreshing(false);
}

public void getAllMatches() {

```

```

        linearProgressIndicator.setVisibility(View.VISIBLE);
        List<Match> list = new ArrayList<>();
        if (user != null) {
            db.collection("matches").get().addOnCompleteListener(task -> {
                if (task.isSuccessful()) {
                    for (QueryDocumentSnapshot documentSnapshot :
Objects.requireNonNull(task.getResult())) {

                        String match_uid = documentSnapshot.getId();
                        String userString;
                        if (Objects.equals(user.getEmail(),
documentSnapshot.get("user"))) {
                            userString = "You";
                        } else {
                            userString = (String) documentSnapshot.get("user");
                        }
                        String team1 = (String) documentSnapshot.get("team1");
                        String team2 = (String) documentSnapshot.get("team2");
                        String status = (String)
documentSnapshot.get("status");
                        String duration = (String)
documentSnapshot.get("duration");

                        Match match = new Match(match_uid, userString, team1,
team2, status, duration);
                        list.add(match);
                    }
                    matchAdapter = new MatchAdapter(list, currentTab,
getChildFragmentManager());
                    initializeRecyclerView(matchRecyclerView, matchAdapter);

                } else {
                    Log.d(TAG, "Error getting data", task.getException());
                    Toast.makeText(getContext(), "Something went wrong",
Toast.LENGTH_SHORT).show();
                }
                linearProgressIndicator.setVisibility(View.INVISIBLE);
            });
        }

        public void getCurrentUsersMatches() {
            linearProgressIndicator.setVisibility(View.VISIBLE);
            List<Match> list = new ArrayList<>();
            if (user != null) {
                db.collection("matches").whereEqualTo("user",
user.getEmail()).get().addOnCompleteListener(task -> {
                    if (task.isSuccessful()) {
                        for (QueryDocumentSnapshot documentSnapshot :
Objects.requireNonNull(task.getResult())) {
                            String match_uid = documentSnapshot.getId();
                            String userString;
                            if (Objects.equals(user.getEmail(),
documentSnapshot.get("user"))) {
                                userString = "You";
                            } else {
                                userString = (String) documentSnapshot.get("user");
                            }
                            String team1 = (String) documentSnapshot.get("team1");

```

```

        String team2 = (String) documentSnapshot.get("team2");
        String status = (String)
documentSnapshot.get("status");
        String duration = (String)
documentSnapshot.get("duration");

        Match match = new Match(match_uid, userString, team1,
team2, status, duration);
        list.add(match);
    }

    matchAdapter = new MatchAdapter(list, currentTab,
getChildFragmentManager());
    initializeRecyclerView(matchRecyclerView, matchAdapter);

    } else {
        Log.d(TAG, "Error getting data", task.getException());
        Toast.makeText(getContext(), "Something went wrong",
Toast.LENGTH_SHORT).show();
    }
    linearProgressIndicator.setVisibility(View.INVISIBLE);
});
    }
}

    public void initializeRecyclerView(@NotNull RecyclerView recyclerView,
MatchAdapter adapter) {
        recyclerView.setLayoutManager(new LinearLayoutManager(getActivity()));
        recyclerView.setAdapter(adapter);
    }
}

```

These Are few source codes snaps for the complete source code visit:

Website: <https://github.com/Rohitsamad/automatic-guacamole.git>

Android: <https://github.com/nitinpawar1999/BeatMe>

11. SYSTEM SNAP

Login Page

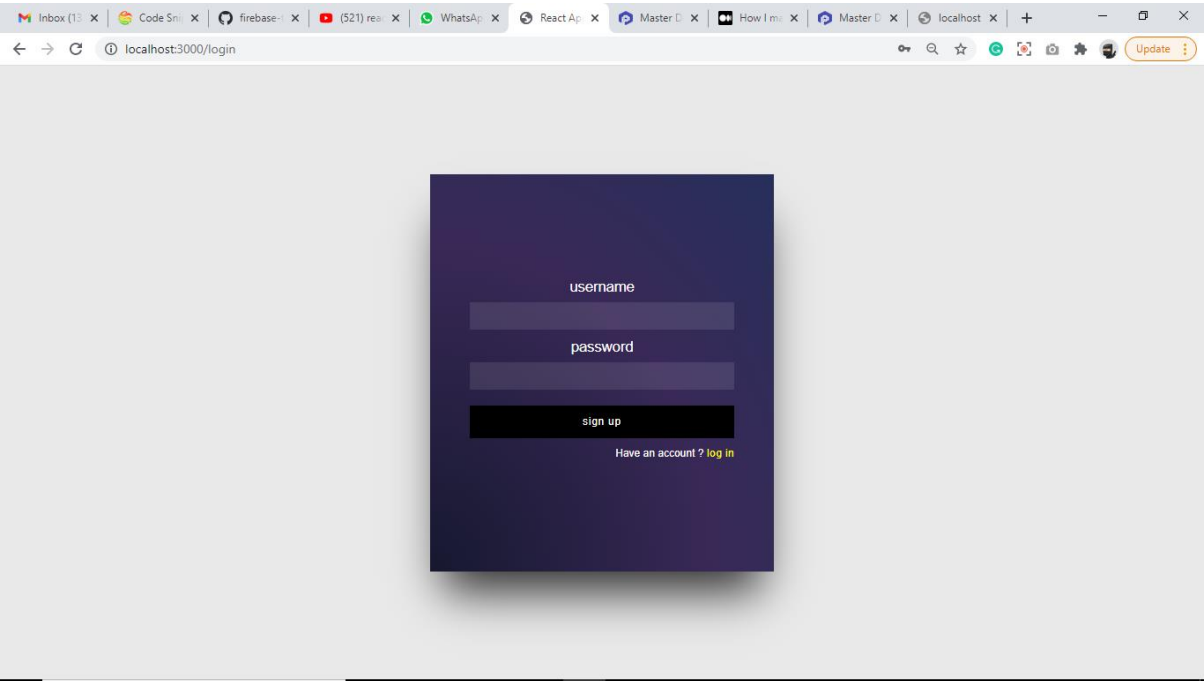


Figure 27 Login Page

signup

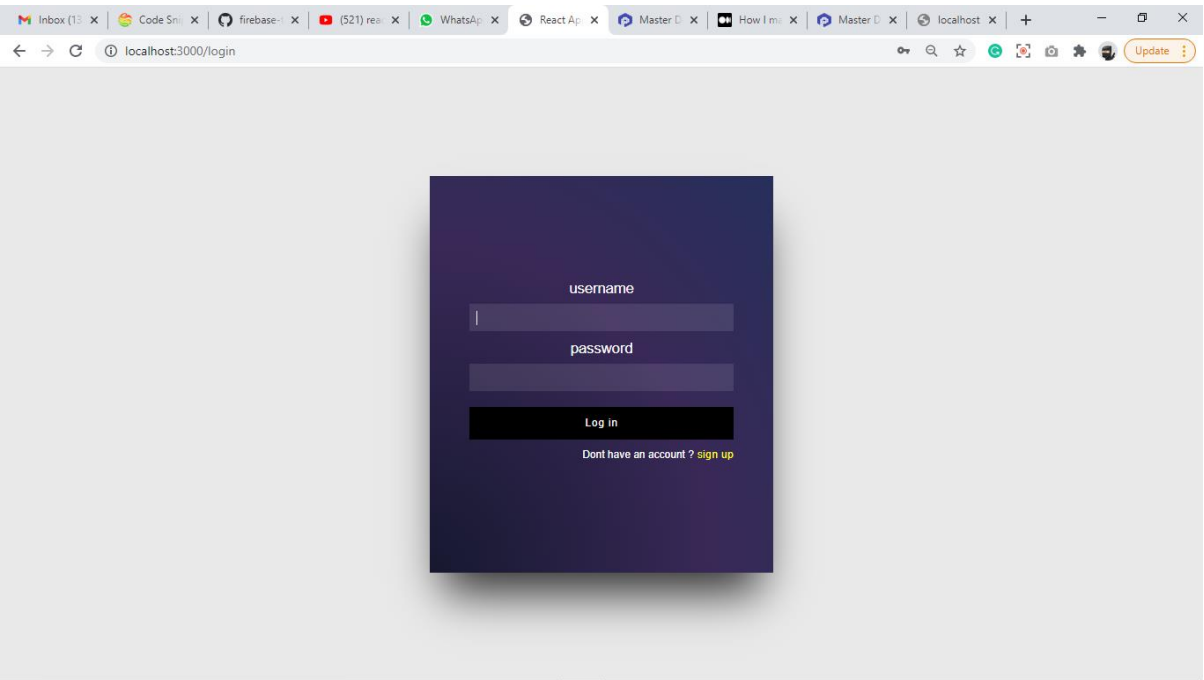


Figure 28 Signup page

home page

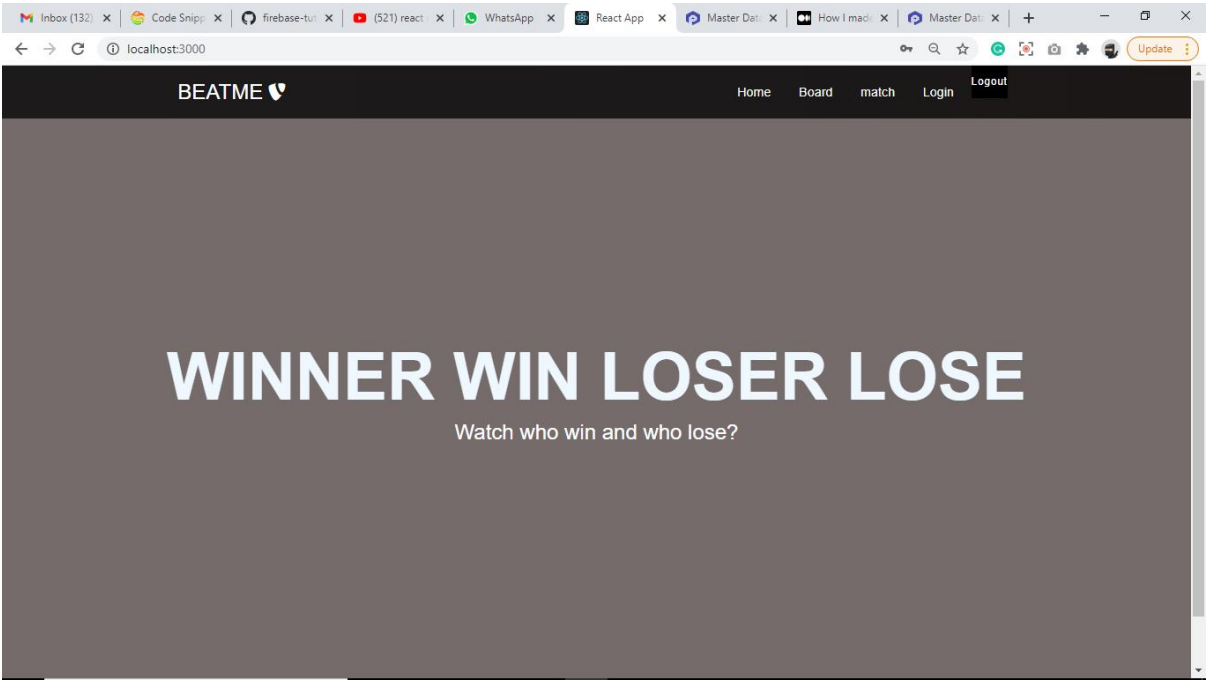


Figure 29 Home Page

Fixture page

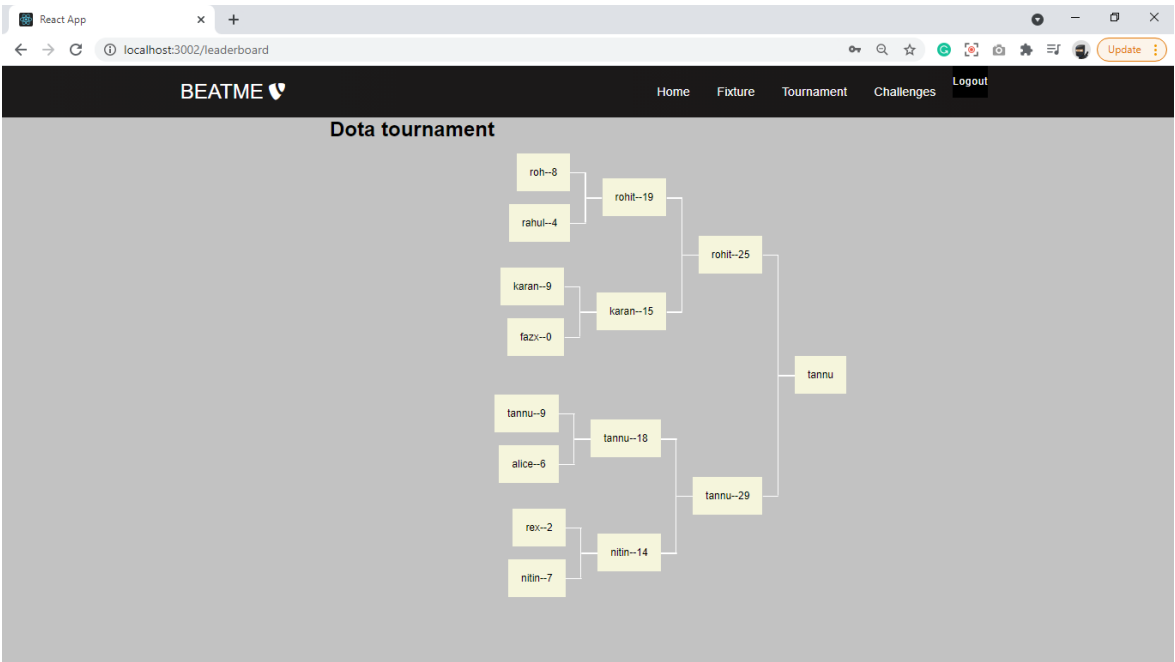


Figure 30 Fixture Page

Adding Tournament name

The screenshot shows a web browser window with the URL `localhost:3002/organization`. The application header includes the logo 'BEATME' and navigation links: Home, Fixture, Tournament, Challenges, and Logout. The main content area is titled 'Enter tournament name' and contains a text input field with the placeholder 'tournament name' and the text 'DOTA TOURBAMENT'. Below the input field is a blue 'NEXT' button.

Figure 31 Tournament creation

Adding teams name and giving points

The screenshot shows the 'Team Name' form in the BEATME application. It lists eight teams with their names and points:

Team	Name	Points
Team1	ROHIT	8
Team2	RAHUL	4
Team3	KARAN	9
Team4	FAZX	0
Team5	TANNU	9
Team6	ALICE	6
Team7	REX	2
Team8	NITIN	7

At the bottom of the form are two buttons: a red 'BACK' button and a blue 'NEXT' button.

Figure 32 Adding Team names and points

Giving points for winner

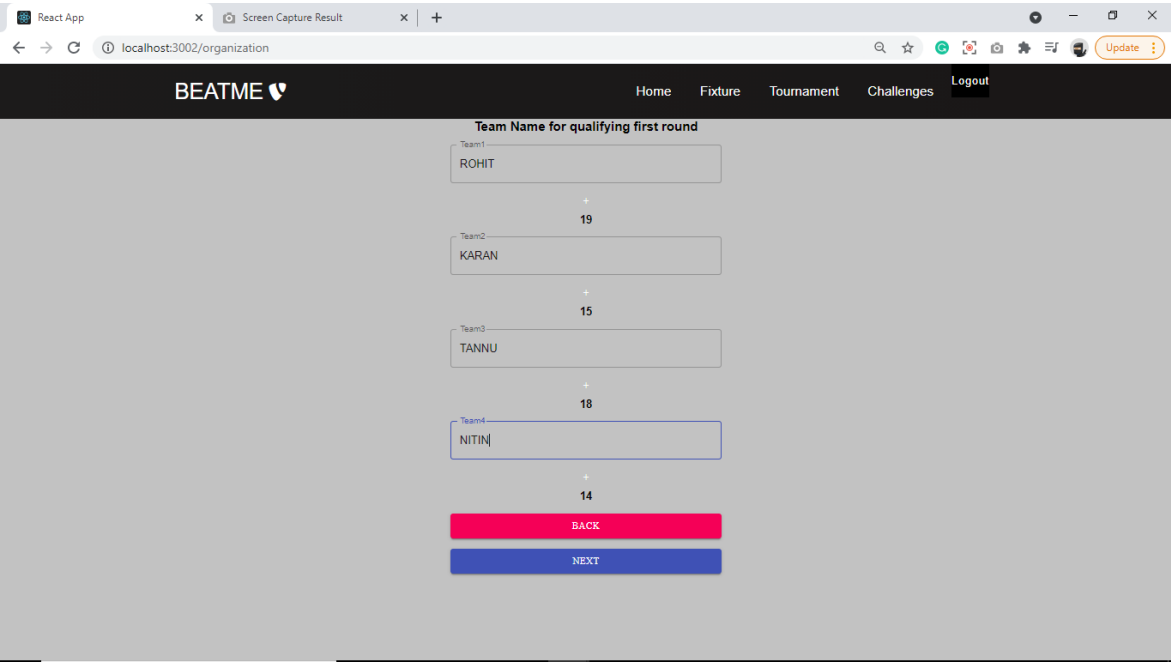


Figure 33 Giving points to qualifier

Giving points for winner

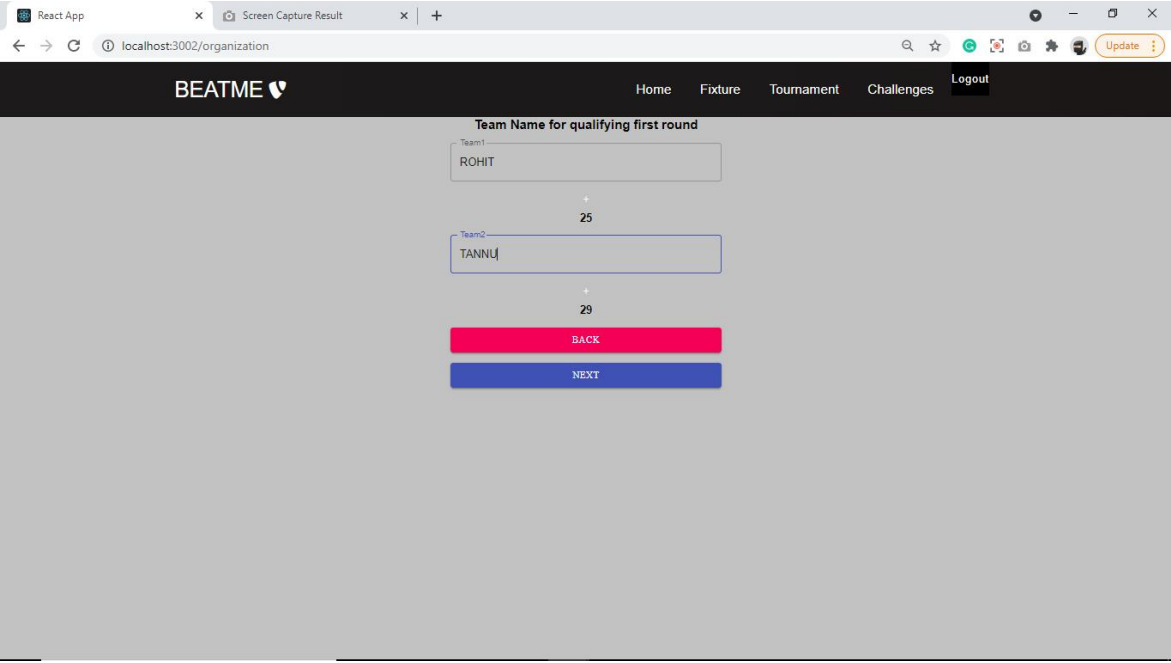


Figure 34 Giving points to semifinal

Winner

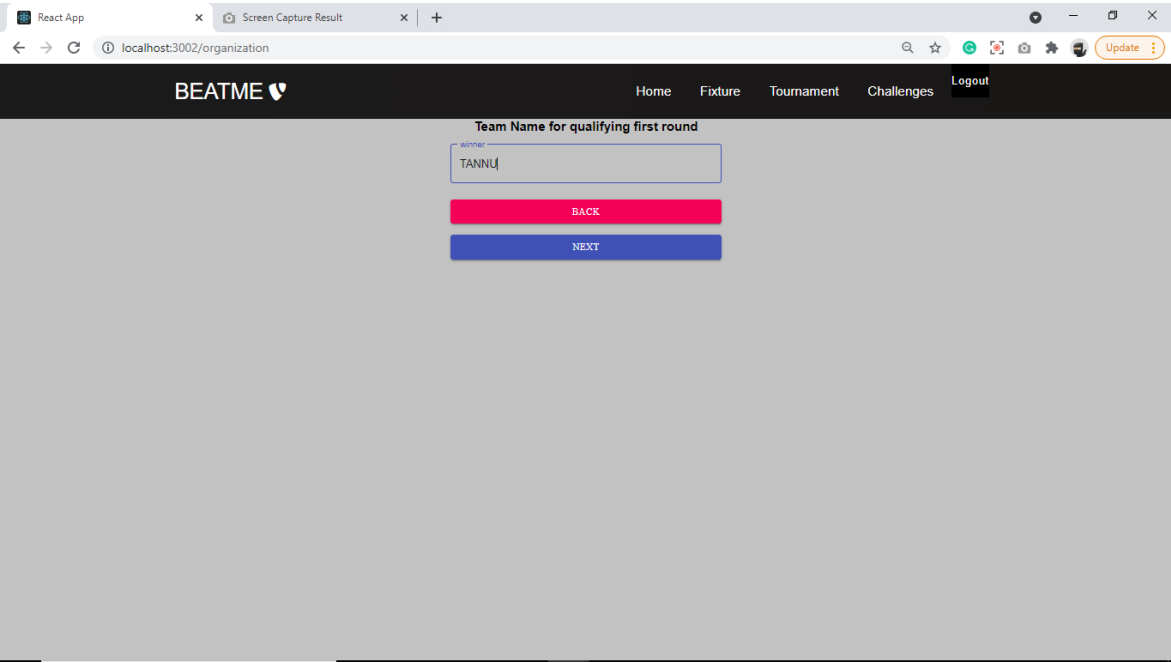


Figure 35 Winner

Review

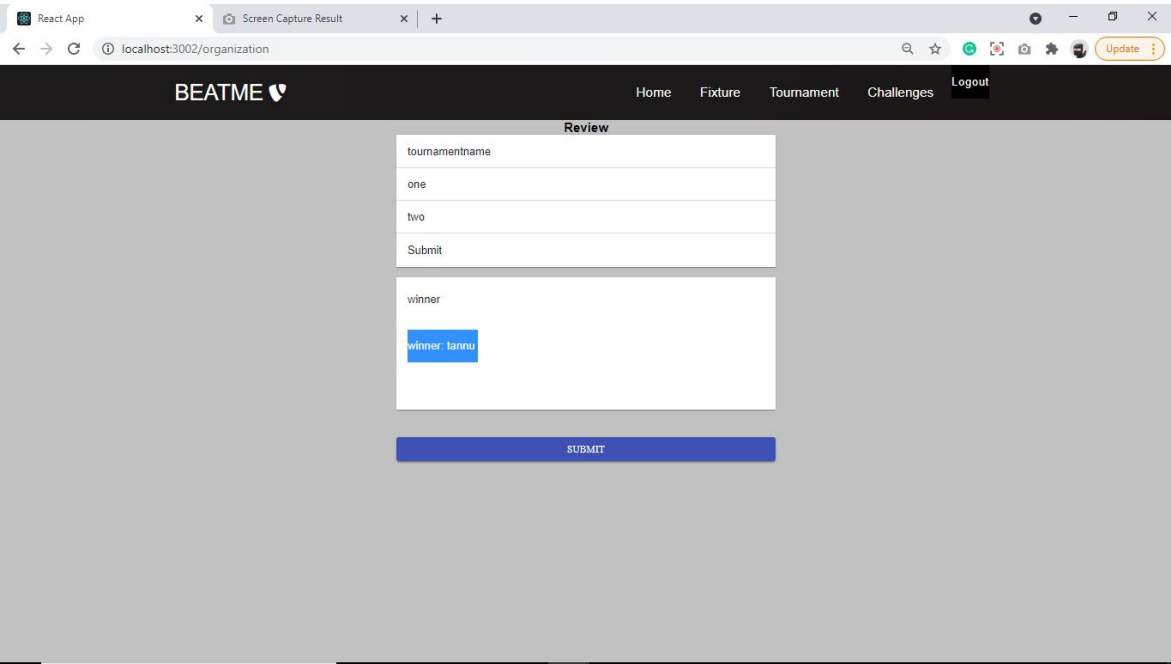


Figure 36 Review

Fixture

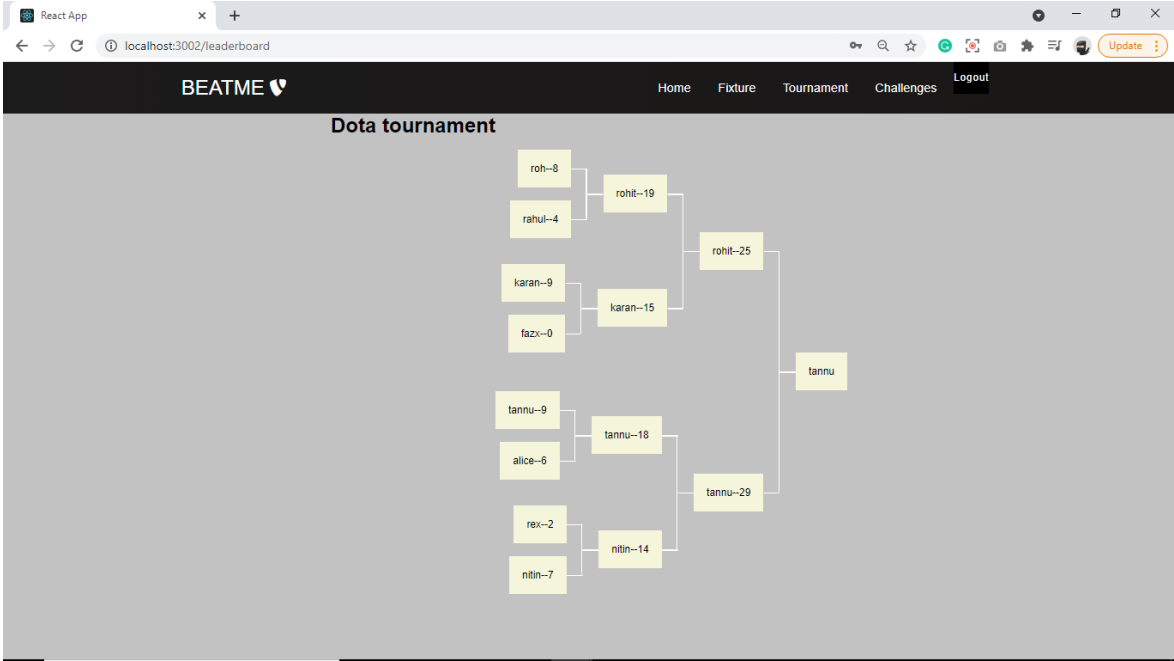
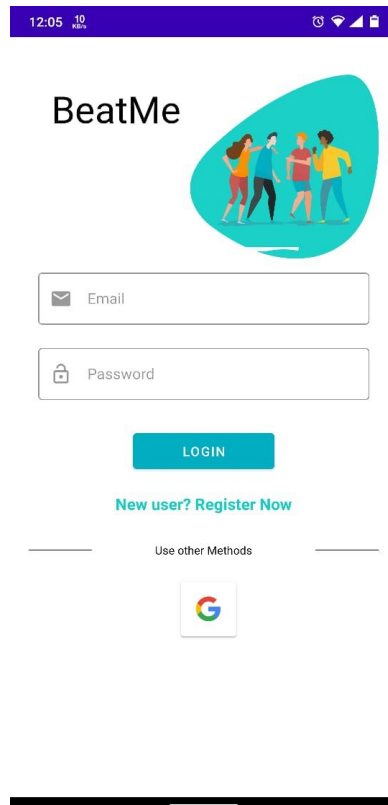


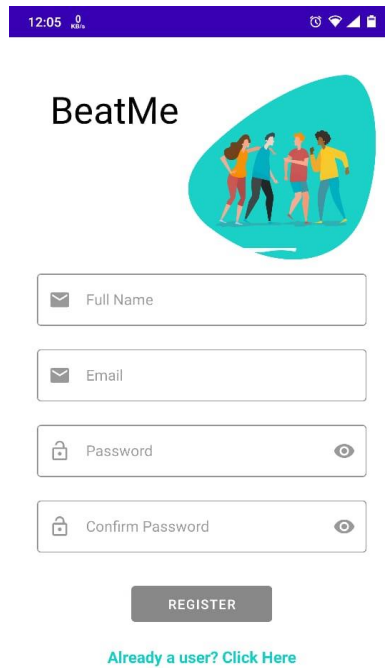
Figure 37 Fixture

Login




The image shows a mobile application login screen for 'BeatMe'. At the top is a purple status bar with the time '12:05', battery level '10%', and icons for alarm, Wi-Fi, and cellular signal. Below the status bar, the app name 'BeatMe' is displayed in a large, black, sans-serif font. To the right of the text is a teal-colored oval graphic containing a white illustration of four people (two men and two women) in a dynamic, dancing or running pose. Below the app name and graphic are two input fields: the first is labeled 'Email' with an envelope icon, and the second is labeled 'Password' with a lock icon. A teal button with the word 'LOGIN' in white capital letters is positioned below the password field. Underneath the button is a link that reads 'New user? Register Now' in teal text. Below this link is a horizontal line with the text 'Use other Methods' centered between two short horizontal dashes. Underneath this line is a square button featuring the Google 'G' logo. At the very bottom of the screen is a solid black horizontal bar.

Figure 38 Login



12:05 0 MB/s

BeatMe



Full Name

Email

Password

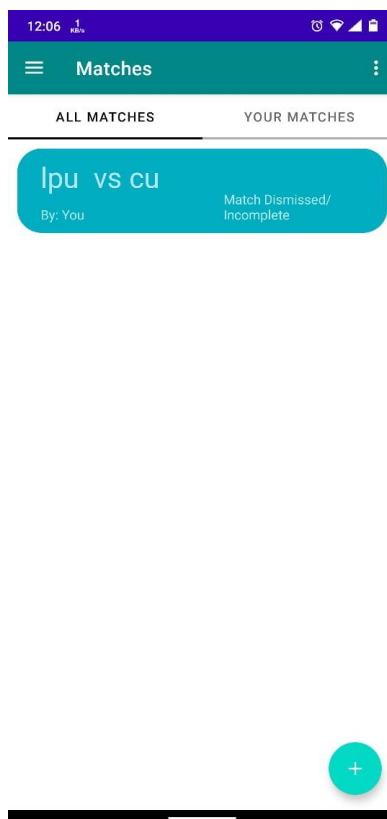
Confirm Password

REGISTER

[Already a user? Click Here](#)

The registration screen features a purple header with the time 12:05 and 0 MB/s. Below the 'BeatMe' title and a running illustration, there are four input fields: 'Full Name', 'Email', 'Password', and 'Confirm Password'. Each field has an icon (envelope for name/email, padlock for password) and a visibility toggle (eye icon). A grey 'REGISTER' button is positioned below the fields, followed by a teal link 'Already a user? Click Here'. A black horizontal bar is at the bottom.

Figure 40 Registration



12:06 1 MB/s

Matches

ALL MATCHES YOUR MATCHES

lpu vs cu

By: You Match Dismissed/Incomplete

+

The matches screen has a purple header with the time 12:06 and 1 MB/s. A teal header bar contains a menu icon, the title 'Matches', and a settings icon. Below this, two tabs 'ALL MATCHES' and 'YOUR MATCHES' are visible. A teal card displays 'lpu vs cu' with 'By: You' and 'Match Dismissed/Incomplete' text. A teal circular button with a white plus sign is at the bottom right. A black horizontal bar is at the bottom.

Figure 39 Challenges

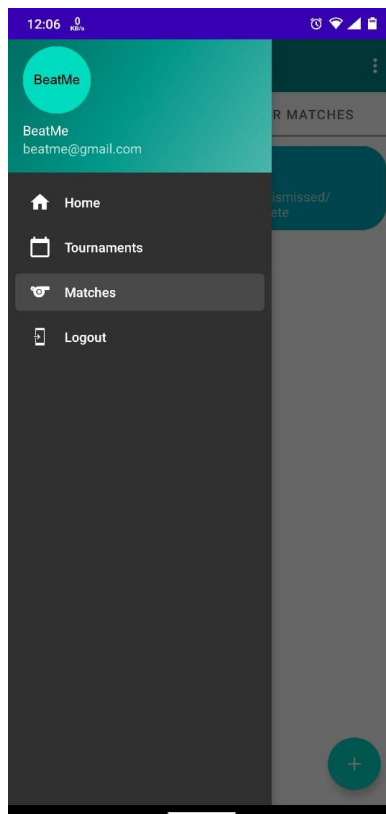


Figure 41 Navition bar

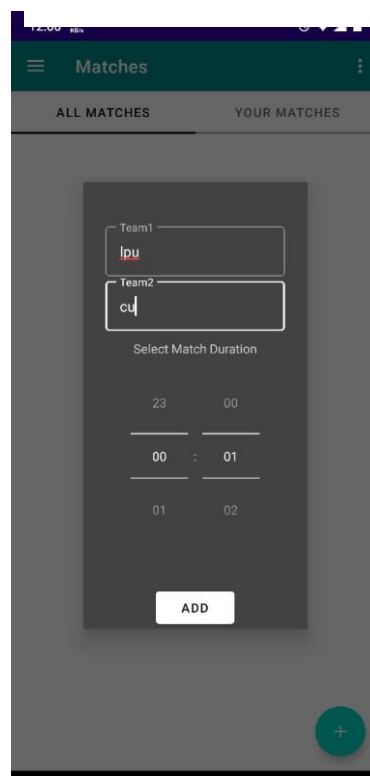


Figure 42 names and time



Figure 43 Match started

12. BIBLIOGRAPHY

- <https://firebase.google.com/docs>
- <https://developer.android.com/docs>
- <https://reactnative.dev/docs/getting-started>
- <https://docs.npmjs.com/getting-started>
- <https://www.youtube.com/>
- <https://stackoverflow.com/>
- <https://reactjs.org/docs/getting-started.html>
- <https://styled-components.com/docs>
- <https://react-spring.io/>
- <https://react-icons.github.io/react-icons/>
- <https://material-ui.com/>
- <https://github.com/revelcw/react-hooks-helper>
- <https://reactrouter.com/web/guides/quick-start>