

STAT 11

21.03.2023

Version 1.0

| Name of the member | Enrollment no. | Mobile no. |
|---------------------------|-----------------------|-------------------|
| Amandeep Singh | 21411005 | 8860368851 |
| Akhil Punia | 21114008 | 8295489973 |
| Manashree Kalode | 21114057 | 8080930624 |
| Nishita Singh | 21114068 | 8826468735 |
| Raiwat Bapat | 21114078 | 7666191528 |
| Subhajit Biswas | 21114100 | 7432080915 |

SUMMARY

This is the Design Document for STAT11, a website that aims to create a platform for digitizing scoring and analytics for small-scale cricket matches. The website will be designed to allow scorers to update scores in real-time while viewers can view live scores and analysis of the match.

Our website STAT11 will have a user-friendly interface enabling scorers to update scores, track player performances, and provide detailed match statistics. Viewers will be able to access the website from a wide range of devices, making it accessible to a larger audience. The website will be built using modern web technologies, including Django Rest Framework, Reactjs and Redux toolkit. It will also utilize a backend database MySQL to store match data and user database.

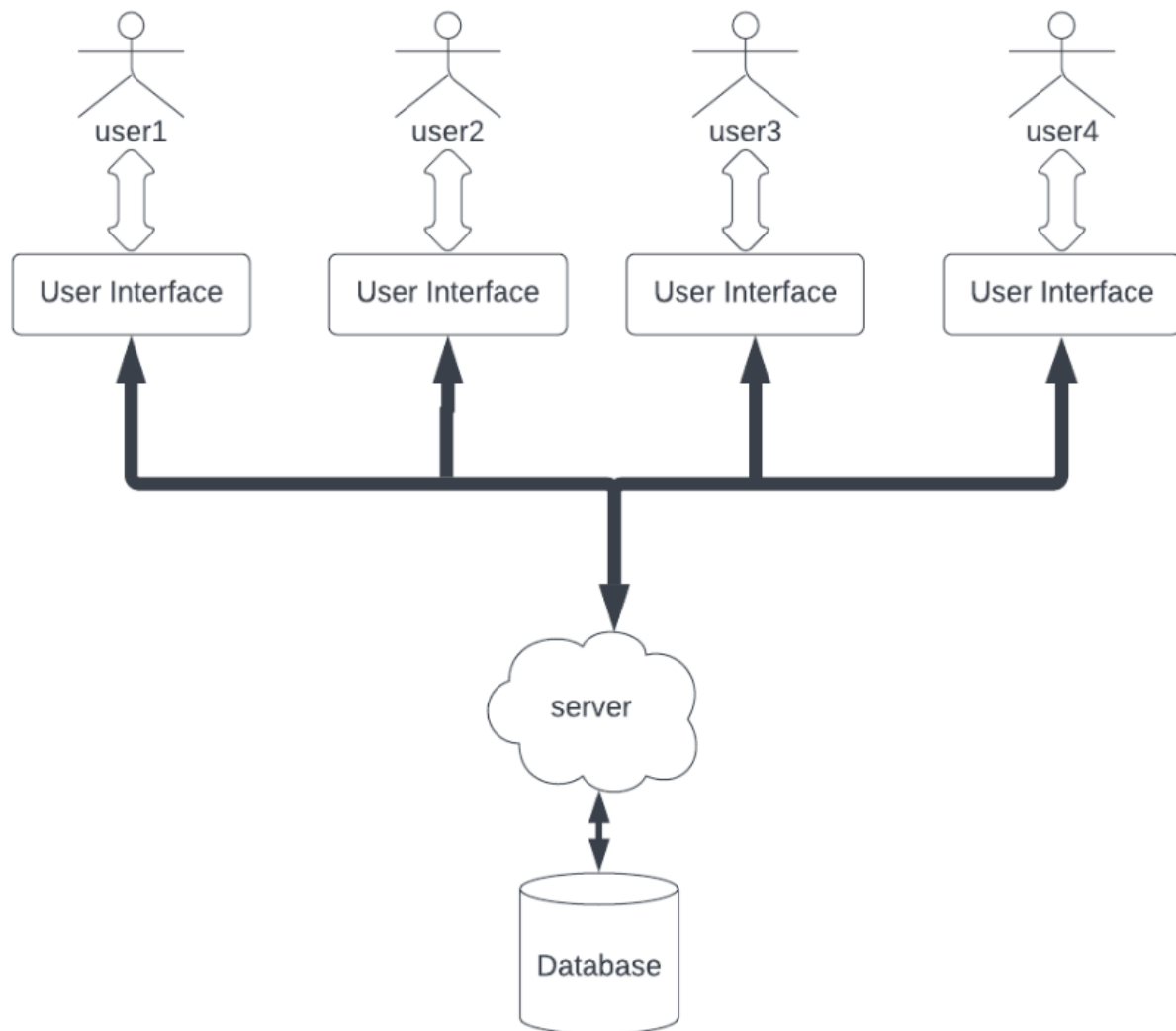
The website will have various features that include live scoring, player statistics, and match analysis. It will also have a dashboard that will display real-time information about ongoing matches, such as the current score and match statistics as well as analytics of past matches that were updated during the matches.

The project will follow an agile development methodology, which will allow for iterative development and continuous improvement. The team will use regular sprint reviews and retrospectives to ensure that the website is meeting the requirements of the stakeholders.

Finally, the project will be tested using various testing methods, including unit testing, integration testing, and acceptance testing. The website will be deployed to a production server once it passes all testing phases.

HIGH LEVEL DESIGN

SYSTEM ARCHITECTURE - MVC



MVC is a design pattern used in software engineering that separates an application into three interconnected components: the Model, the View, and the Controller. This separation allows for the application to be more modular and easier to maintain and scale.

Here is a brief explanation of each component:

1. **Model:** This component represents the data and the business logic of the application. It contains the application's data and the rules for manipulating that data.

2. **View:** This component represents the user interface of the application. It is responsible for presenting the data from the Model to the user and for receiving input from the user.
3. **Controller:** This component acts as an intermediary between the Model and the View. It receives input from the user via the View, manipulates the data in the Model, and updates the View with the new data.

The advantages of using MVC architecture include:

1. **Separation of concerns:** MVC separates the application into three distinct components, allowing for each component to be developed independently. This separation of concerns leads to better modularity and easier maintenance.
2. **Reusability:** Since each component of the MVC architecture is modular, it can be reused in other parts of the application or in other applications altogether. This leads to faster development times and reduced costs.
3. **Testability:** The separation of concerns also makes it easier to test each component of the application independently. This leads to more comprehensive and reliable testing.
4. **Scalability:** The modular nature of MVC makes it easier to scale the application as the user base grows. Each component can be scaled independently, allowing for better performance and reliability.
5. **Flexibility:** Because each component of the MVC architecture is independent, it is possible to change one component without affecting the others. This allows for more flexibility in the development process and easier updates to the application.

Although the Model-View-Controller (MVC) architecture has many advantages, there are also some potential disadvantages that should be considered. Here are a few:

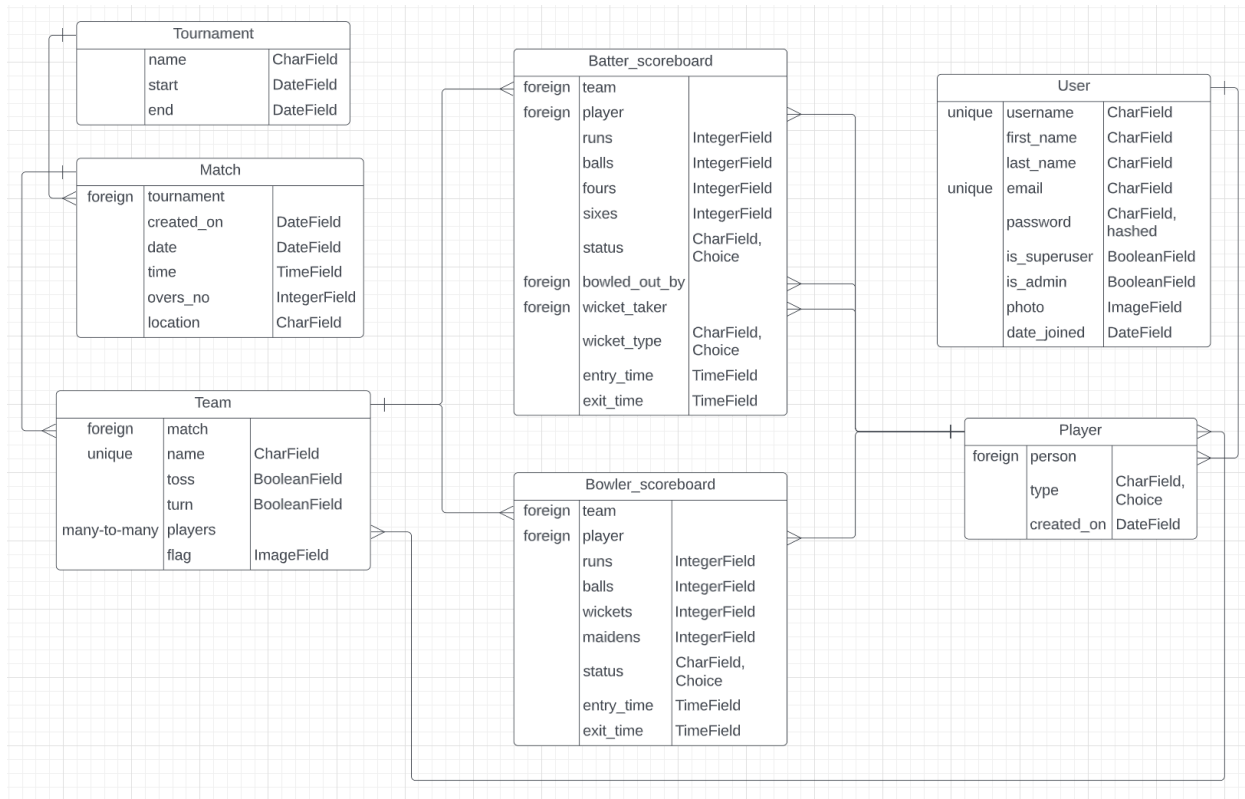
1. **Complexity:** Implementing MVC can be more complex than other design patterns. This is especially true for smaller applications where the additional complexity may not be necessary.
2. **Overhead:** The separation of concerns in MVC can result in additional overhead in terms of code and performance. This can be a concern for applications that require high performance or have limited resources.

3. **Learning Curve:** MVC can have a steeper learning curve than other design patterns, especially for developers who are not familiar with it. This can lead to longer development times and increased costs.
4. **Tight Coupling:** While MVC separates the application into distinct components, it can also result in tight coupling between those components. This can make it more difficult to modify or replace a component without affecting the others.
5. **Over-Engineering:** MVC can lead to over-engineering, where the design becomes overly complex or the separation of concerns is taken too far. This can result in a design that is difficult to maintain or modify.
6. **Potential for Duplication:** MVC can potentially result in duplication of code, especially if the separation of concerns is not well-defined. This can make the codebase harder to manage and lead to bugs and errors.

It's important to weigh both the advantages and disadvantages of MVC, and consider whether it's the right design pattern for your specific application and development team.

DATABASE MODELING

Entity Relationship Diagram (ERD)



User

Attributes

1. Username(String) - Unique username for each user
2. First_name(String) - First name of user
3. Last_name(String) - Last name of user
4. Email(String) - Unique email address for the user.

5. Password(String) - Password string used while authenticating users registered in the application, will be stored via SHA256 hashing.
6. Is_superuser(Boolean) - A boolean telling whether the user has superuser rights or not.
7. Is_staff(Boolean) - A boolean telling whether the user has admin rights or not.
8. Photo(Image) - String representing user's avatar
9. Date_joined(Date) - Date when the account was created.

Player

Attributes

1. Person(Foreign_Key) - A foreign key representing the user object the player is connected to
2. Type(String) - String containing the player's type (batter, bowler, all_rounder)
3. Created_on(Date) - Contains the date this player was created

Tournament

Attributes

1. Name(String) - Name of the tournament
2. Start(Date) - The start date of the tournament
3. End(Date) - The end date of the tournament

Match

Attributes

1. Tournament(Foreign_Key) - The foreign key representing the tournament given match is part of. It may be null in case the match is an independent match.
2. Created_on(Date) - Contains the date when the match was created.
3. Date(Date) - The date scheduled for the match.
4. Time(Time) - The time match is scheduled at
5. Overs_no(Integer) - Number of overs in the match
6. Location(String) - The location of the match.

Team

Attributes

1. Match(Foreign_Key) - The foreign key representing the match given team is participating in.
2. Name(String) - The name of the team
3. Toss(Boolean) - A boolean telling if the team won the toss
4. Turn(Boolean) - A boolean telling if its the team's turn to bat
5. Players(Many_To_Many) - Players list which are part of the team
6. Flag(Image) - Image representing the team's flag

Batter_scoreboard

Attributes

1. Team(Foreign_Key) - The foreign key representing the team given batter is the part of

2. Player(Foreign_Key) - The foreign key representing the player object whose scores are been recorded in this table
3. Runs(Integer) - Integer representing the runs scored by the batter
4. Balls(Integer) - The balls played by the batter
5. Fours(Integer) - Number of fours hit
6. Sixes(Integer) - Number of sixes hit
7. Status(String) - Contains the status i.e. yet_to_bat, batting, out, idle of the batter
8. Bowled_out_by(Foreign_Key) - The foreign key to the bowler who took the wicket
9. Wicket_taker(Foreign_Key) - Foreign key to the player who took the wicket
10. Wicket_type(String) - The type of wicket
11. Entry_time(Time) - The time batter entered the arena
12. Exit_time(Time) - The time batter exit the arena

Bowler_scoreboard

Attributes

1. Team(Foreign_Key) - The foreign key representing the team given bowler is the part of
2. Player(Foreign_Key) - The foreign key representing the player object whose data is been recorded in this table
3. Runs(Integer) - The runs conceded by the bowler
4. Balls(Integer) - Number of balls bowled

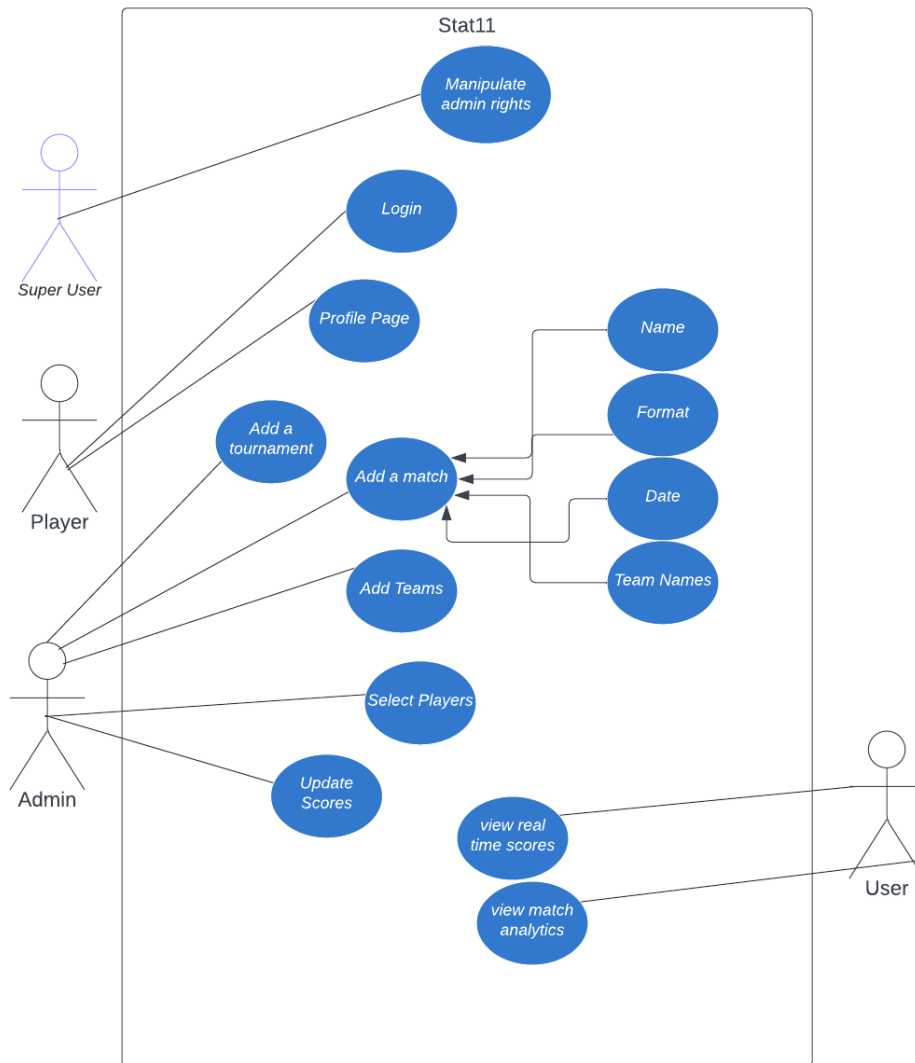
5. Wickets(Integer) - Number of wickets taken
6. Maidens(Integer) - Number of overs bowled by the bowler with 0 runs scored by batter
7. Status(String) - Contains the status i.e. idle, bowling of the bowler

Function

CRUD (Create Retrieve Update Delete) operations are performed on every database table. Users based on his/her access rights will be able to request for above operations on the database.

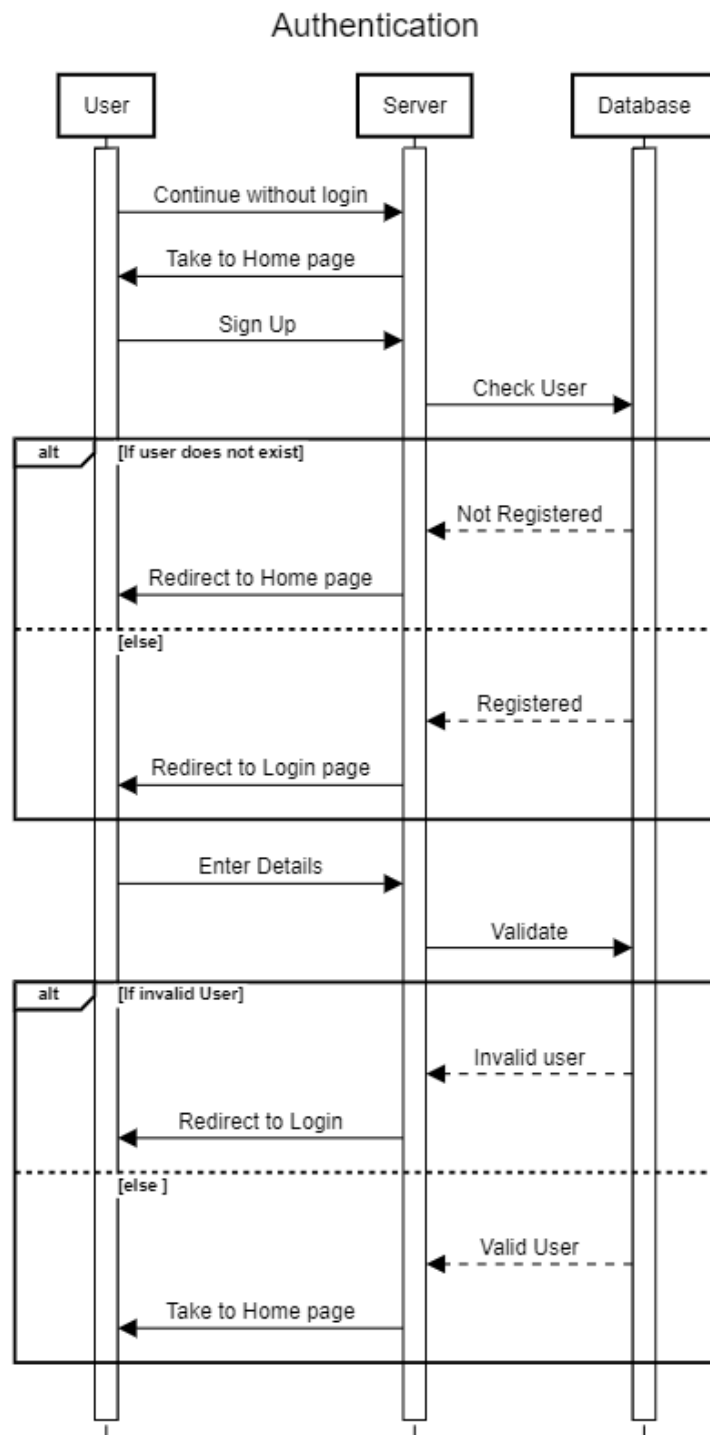
DESIGN OF FUNCTIONAL UNITS

Use case representation

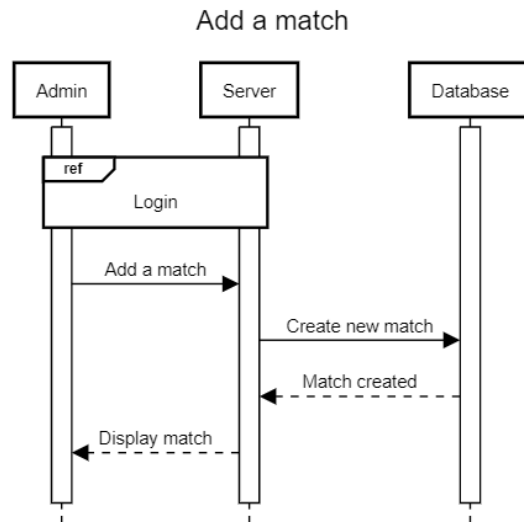


Sequence Diagrams:

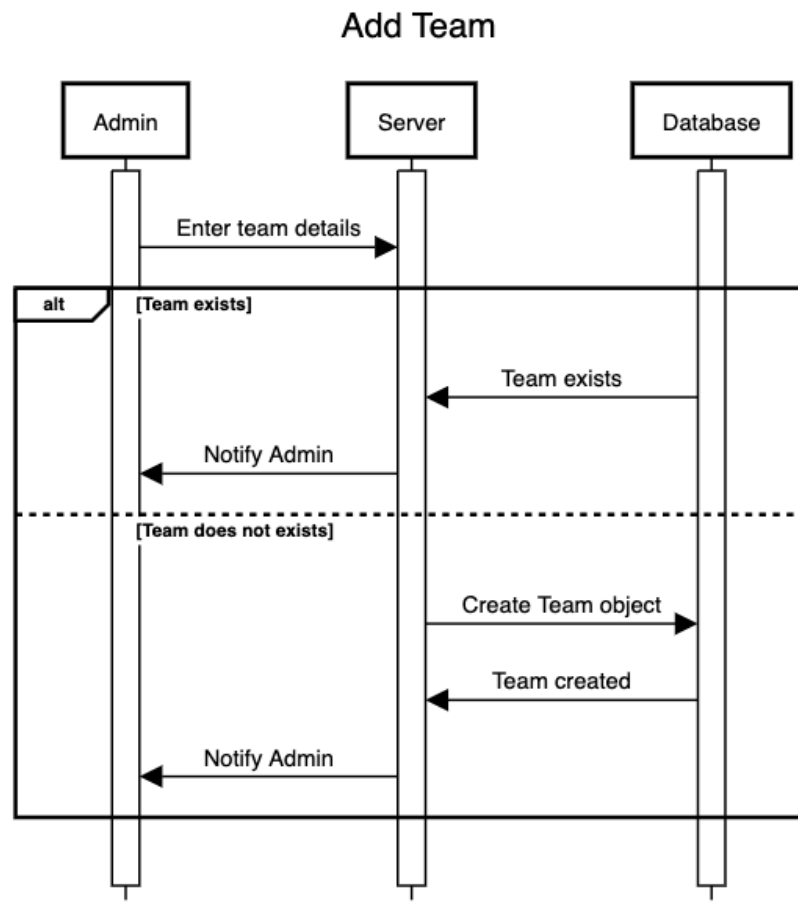
1. Authentication



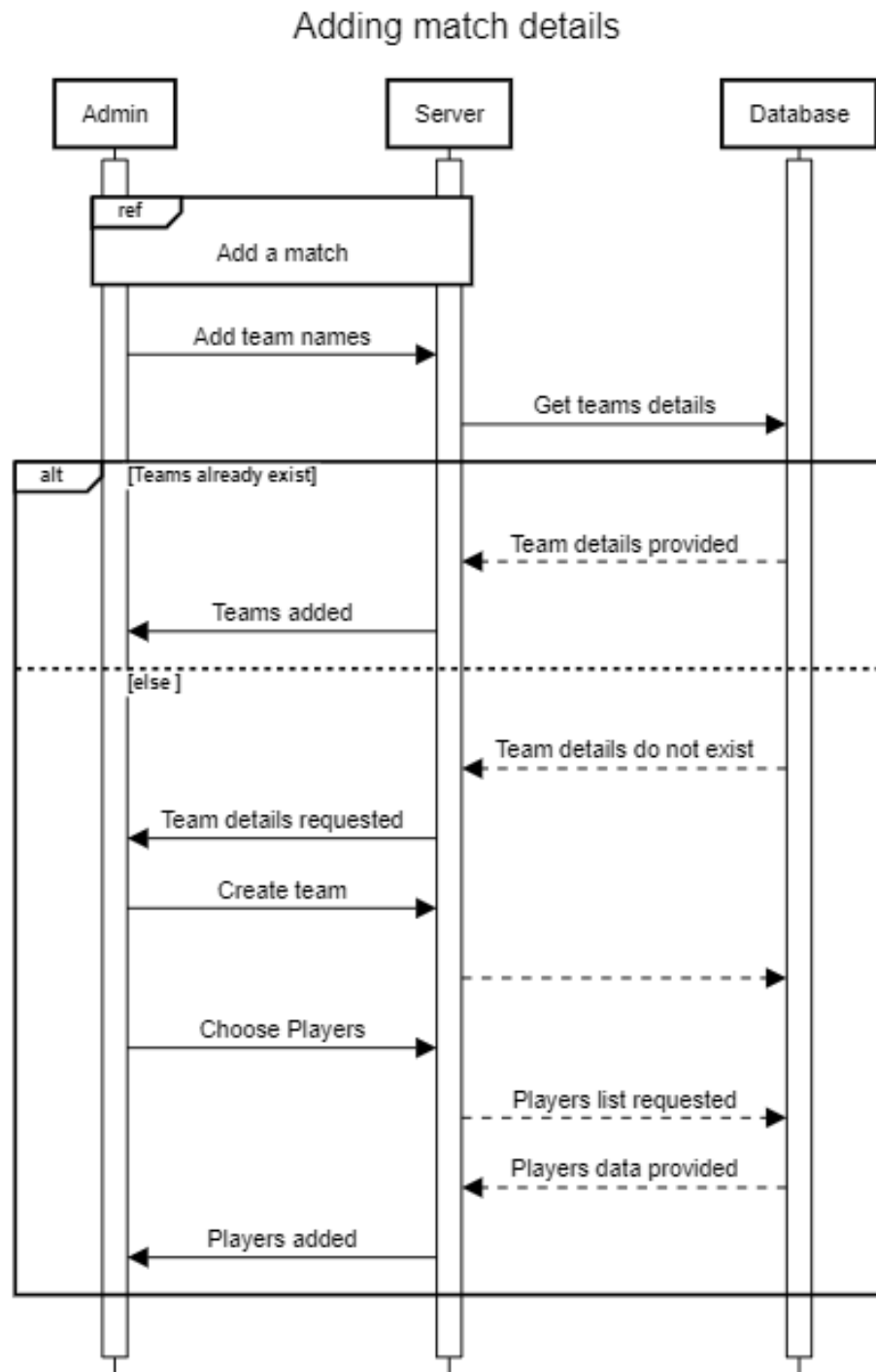
2. Add a match



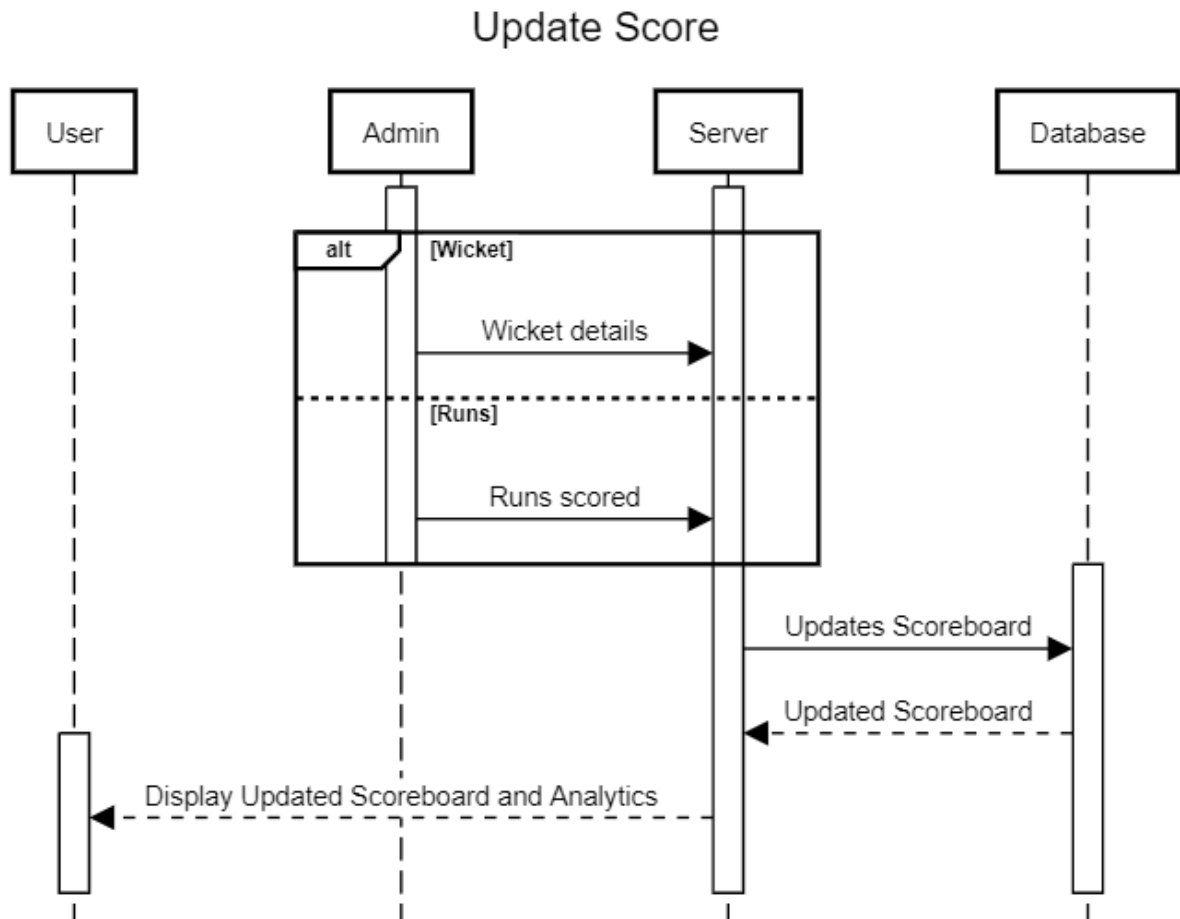
3. Add Team



4. Adding match details



5. Update Scores

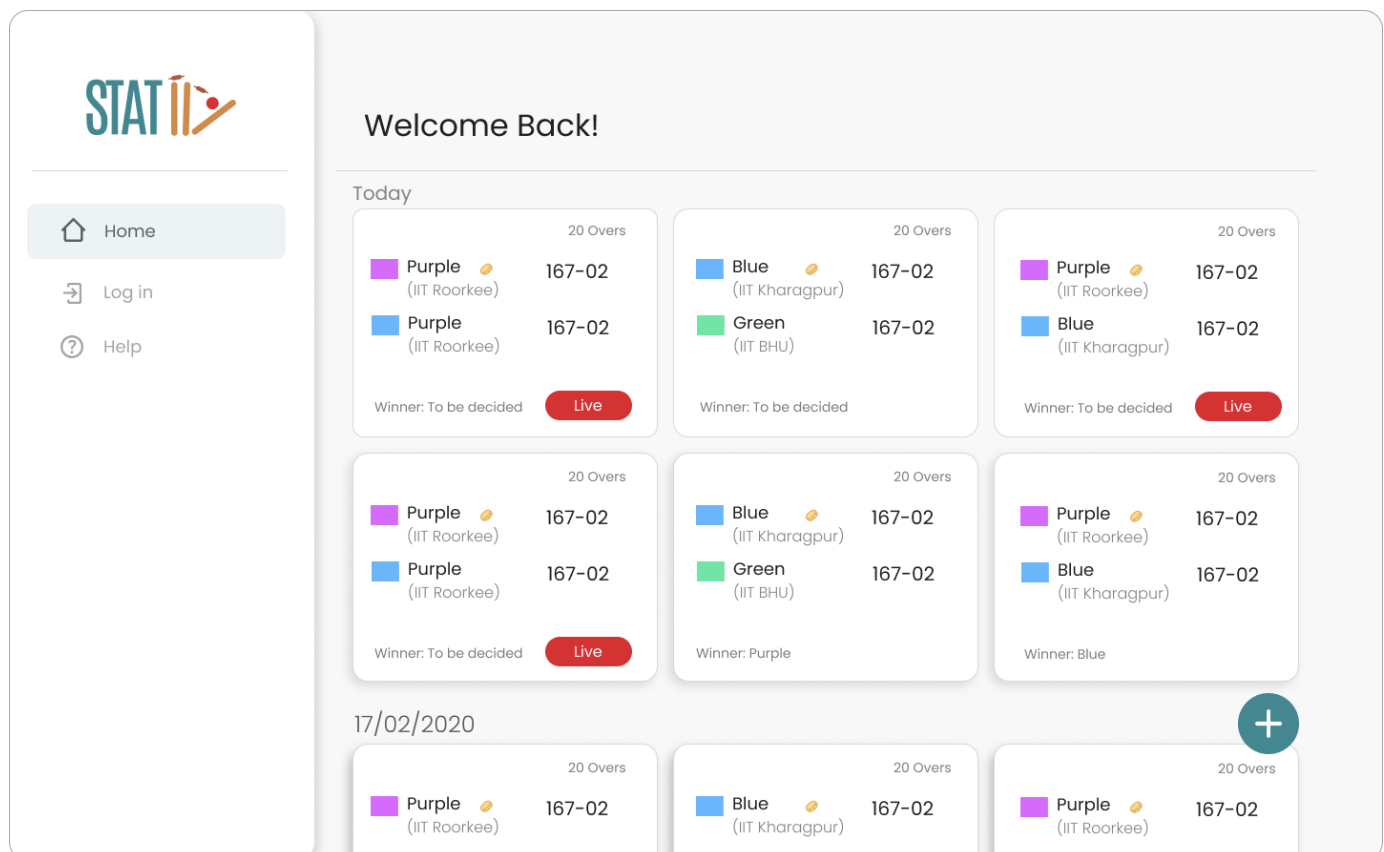


Low-Level Design

User Interface

1. Home page

- This is the landing page for everyone. Players and admins will have to navigate to the login page to sign in.
- It will show the following information about all the recent matches.
 - Names of the two teams
 - The number of overs
 - The Winner of the toss
 - The Winner of the match (if the match is over)
- It will have a side panel for navigation to various pages like the login page, tournaments page and a help page.
- If an admin is logged in he/she will also be able to view the add a match button.



2. Login Page

- This page is for superadmins, admins and players to log in.
- Users will have to enter their email address and their password.
- On clicking on the login button, we check the entered email address and password with our database and redirect to the home page if the fields are correct.

Welcome back,

[Forgot Password?](#)

Submit

[Continue without login](#)

3. Login Page (Forgot password)

- This page is for superadmins, admins and players to log in.
- Users will have to enter their username and email address.
- On clicking on the submit button, we check the entered email address and username with our database and send an email to the user if the fields are correct.

User Name

Email-ID

We will mail you a link to reset password

Submit

[Continue without login](#)

4. Sign up Page

- This page is only for the players and admins.
- Apart from email-id and password, we will ask whether the player is a batter, bowler or an all rounder.
- On clicking on sign up, the data of the player is added to our database, and the player is redirected to our home page.

User Name

E-Mail ID

Password

Confirm Password

Role

☐ Batter

☐ Bowler

☐ Scorer

Sign Up

[Continue without login](#)


5. Individual Match Page

- On clicking on any match on the home page, this page will show up.
- This page will have multiple subpages to suit every type of audience

The subpages are as follows :

1. **Live** : The default page will have the current score and the details of the current over i.e the batters, bowler, runs made in that over.

For admins this page will also show various options to update the scores corresponding to each ball.



Live

Scoreboard

Analytics

Teams

Highlights

Purple Team v/s Orange Team

Location, City, State Match Type

Toss: Purple Team

Purple Team 107(14.3)

CRR 45.6

| Batters | Team | Runs | Balls | 4s | 6s | S.R. |
|------------------|--------|------|-------|----|----|------|
| 1. Raiwat Bapat | Purple | | | | | |
| 2. Nishita Singh | Purple | | | | | |

| Bowler | Team | Overs | Maidens | Runs | Wickets | Eco. |
|-----------------|------|-------|---------|------|---------|------|
| 1. Raiwat Bapat | Blue | | | | | |

Runs scored

1

2

3

Bye

No ball

Wicket

4

5

6

Wide

Leg bye

Add

Mode of Dismissal

☐ Bowled

☐ Caught

☐ LBW (Leg Before Wicket)

☐ Runout

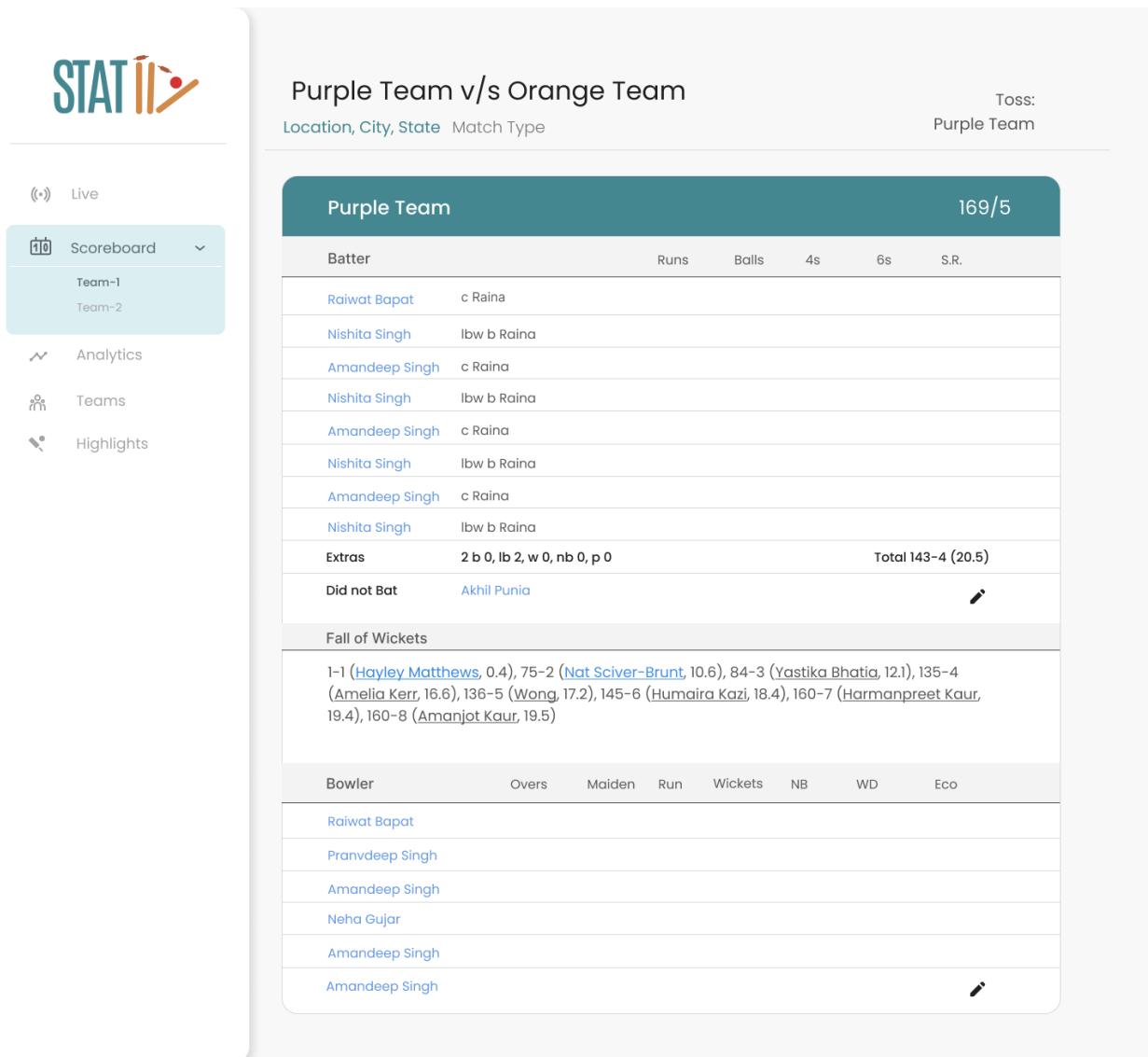
☐ Stumped

☐ Hit Wicket

Wicket Keeper

Update

2. **Scoreboard** : This shows the runs made by each player along with the number of fours and sixes, the fall of wickets, the number of wickets taken by each bowler. It will also show the strike rate as well as the economy rate for all the players.



3. **Analytics** : It shows the details of the match in a visual and easy to understand manner. We are showing 4 charts,
- Run worms (Runs vs Overs)
 - Run rate (Runs per over for all the overs)
 - Top-5 Batters (Distribution of the runs amongst the top 5 batters)
 - Runs v/s Overs bar-graph

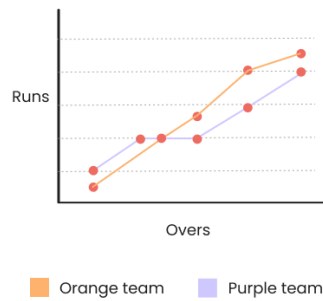
Purple Team v/s Orange Team

Location, City, State Match Type

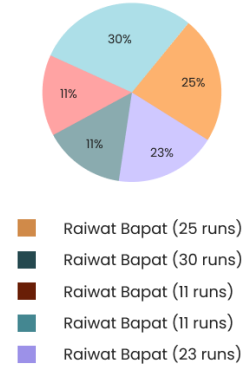
Toss:
Purple Team

Purple Team

Runs v/s Overs



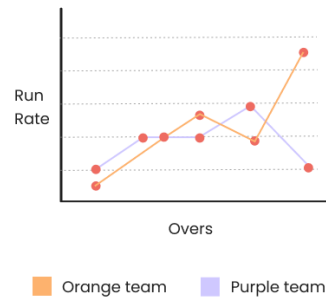
Team Purple



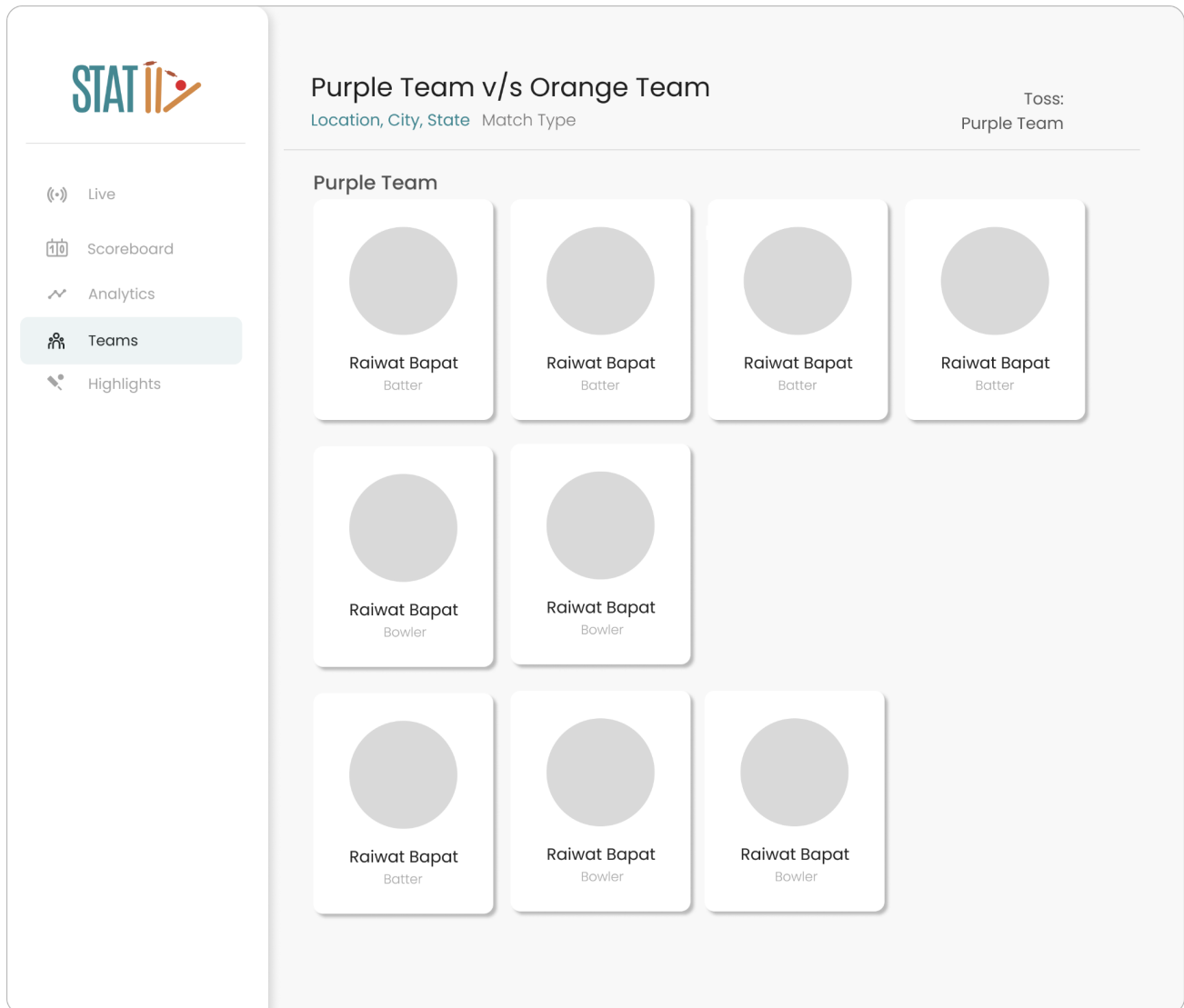
Runs per over




Run Rate



- Teams :** It shows the names of all the players for both the playing teams, along with their positions on the team.



5. **Highlights** : It shows the names and scores of the top 3 best performing batters, and bowlers. It also shows the top 3 players overall, using the formula used for calculating the MVP.



- Live
- Scoreboard
- Analytics
- Teams
- Highlights

Purple Team v/s Orange Team

Location, City, State Match Type

Toss: Purple Team

| MVP | Team | Points |
|-------------------|--------|--------|
| 1. Raiwat Bapat | Purple | 4.573 |
| 2. Nishita Singh | Purple | 4.573 |
| 3. Amandeep Singh | Orange | 4.573 |

| Best Batters | Team | Runs | Balls | 4s | 6s | S.R. |
|-------------------|--------|------|-------|----|----|------|
| 1. Raiwat Bapat | Purple | | | | | |
| 2. Nishita Singh | Purple | | | | | |
| 3. Amandeep Singh | Blue | | | | | |

| Best Bowlers | Team | Overs | Maidens | Runs | Wickets | Eco. |
|-------------------|--------|-------|---------|------|---------|------|
| 1. Raiwat Bapat | Purple | | | | | |
| 2. Nishita Singh | Purple | | | | | |
| 3. Amandeep Singh | Blue | | | | | |

5. Add a Match Page

Only accessible to the admins through an add match button on the home screen.

The admin has to enter the names of the two teams and then add the players from a dropdown list.



Home

Log in

Help

New Cricket match

Team 1

Raiwat Bapat

Nishita Singh

Sannidhya Singh

+ Add a player

Team 2

Raiwat Bapat

Nishita Singh

Sannidhya Singh

+ Add a player

Players

Search

Raiwat Bapat

Amandeep Singh

Nishita Singh

Amandeep Singh

Nishita Singh

Amandeep Singh

Nishita Singh

No. of overs

Submit

Development Environment

1. A laptop will be required to carry out the development process. System requirements of the laptop are as follows,
 - a. Computer with at least 4GB ram and a decent CPU
 - b. The operating system can be any of windows 10, windows 11 , or linux distros like Ubuntu 22.04 and above, Manjaro,etc. However the windows developers are required to install either git bash or wsl2.
2. ReactJS , MySQL and Django Rest Framework must be installed and set up on the system.
3. Front-end code of the application will be written in ReactJS using the Redux toolkit. And the backend will be written in Django Rest Framework (using python). Therefore, code editors like VSCode, Atom, Sublime, Notepad++, etc. can be used to edit the code. However, it is recommended to use VSCode as it offers many advantages like syntax highlighting, code formatting,error detection, automatic code completion with the help of Extensions.
4. Git and Github are used for source code management. So the system must have the latest version of git installed on the system.

Tech Stack used

- [ReactJS](#)
 - ReactJS is a popular JavaScript library for building user interfaces with reusable UI components.
 - ReactJS uses a virtual DOM, which improves performance by minimizing the number of updates to the actual DOM.
 - ReactJS has a large and active community of developers, which means there are many third-party libraries and resources available.

- Redux Toolkit

- Redux Toolkit will be used for state management in the application
- Redux Toolkit is a set of utilities and best practices that simplify the process of writing Redux code.
- Redux Toolkit provides a standard way to write and configure a Redux store, which reduces boilerplate code and makes the codebase more maintainable.
- Redux Toolkit includes a powerful data fetching library, which makes it easy to handle asynchronous data loading in Redux applications.

- Django Rest Framework(DRF)

- Django Rest Framework is a powerful framework for building RESTful APIs in Python.
- Django Rest Framework includes built-in support for authentication, serialization, and pagination, which makes it easy to build scalable and secure APIs.
- Django Rest Framework is highly customizable and supports a wide range of HTTP methods, making it a versatile tool for building APIs.

- MySQL Database

- MySQL is a popular open-source relational database management system that uses SQL.
- MySQL is fast, reliable, and scalable, making it a popular choice for web applications that require database functionality.
- MySQL has a large and active community of developers, which means there are many third-party libraries and resources available.