**1. Angular Unit Testing - Football Tournament Management**

**Angular - Jasmine - Unit Testing**

**Instructions to install and run front-end and back-end in online IDE :**

* You should install project dependencies, To install project dependencies and run angular front-end server, Click **Run ->** **Install**.  
  (If installation not ends with running angular server or brings error, click **Run -> Install** again or give command to run angular - **npm start**in terminal)
* To run the back-end server, click **Run ->** **Run** (or give command to run node - **npm start**in terminal, don't modify the backend).
* For viewing your application, click **Run -> Open Preview**(you can view your application in new tab)

**Football Tournament Management**

Football Tournament Management application developed in **Angular**as frontend and **NodeJS**as backend. This application used to register the teams participating in the tournament, remove/update the team and player details, select and remove the players for team11 and map the teams for the tournament leader board.  The functionalities of the application are shared below.

**Now, you need to write unit test cases to cover all the statements of the application**.

**Instructions to run test cases :**

* Start writing the test cases. Write the test cases in the spec files that can be found at **src/components/\*** , **src/guards/\* and src/services/\* .**
* After writing the test cases, you can test your project by clicking **Run > Test**or using**npm test**command in the terminal
* You can view the**test report**corresponding to the written test cases in **'test-results.xml'.**
* You can view the **coverage report** corresponding to the written test cases in **'./coverage/footBallTournament/clover.xml'.**
* Once you are done with writing unit test cases and fixing quality issues, click **Submit** for evaluation.

**Note:**

1. **Any syntax errors or invalid statements in your test cases will stop generating the test reports.**
2. **This application contains some dummy test cases, please don't remove that because it needs for validation.**
3. **Making any changes in the read-only files will result in a zero score.**

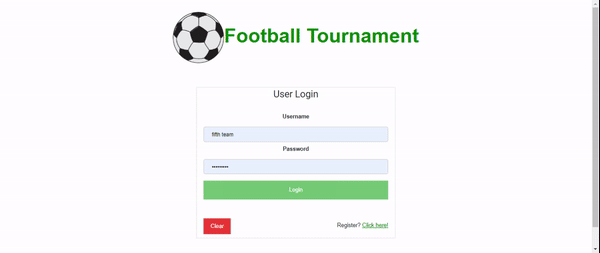
**FUNCTIONALITY:**

* In this application, There will be two users **admin** and **team**. A team can manage its own team after a successful login where as ADMIN can manage all the teams.
* The condition for the playing eleven is that in the **playing eleven** there must only **one goal keeper**, **atleast one mid-fielder**, **atleast one forwarder** and**atleast one defender.**
* The **leaderboard** option which in this case is called mappings model  is used to map teams for semi finals and final and declare winner.
* This leaderboard option is only updated by an admin user and there must be at least 4 teams to manage leaderboard. Team can just view the leaderboard.

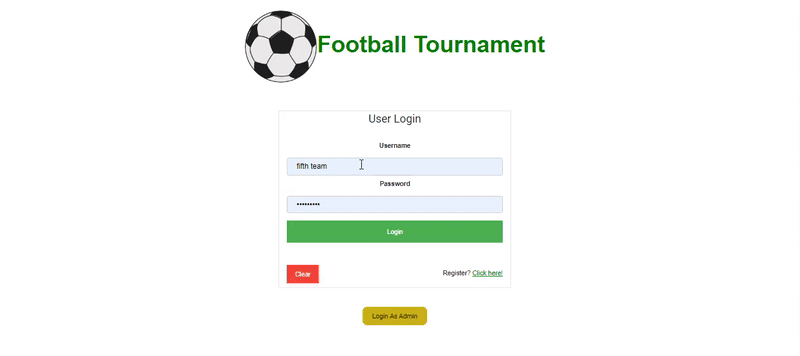
**Note : There is a defaultDB.js file in backend which contains default data set to store in database for admin and user login.(nodejs/src/mongoose/db/defaultDB.js)**

**Components**

**AdminLogin Component**

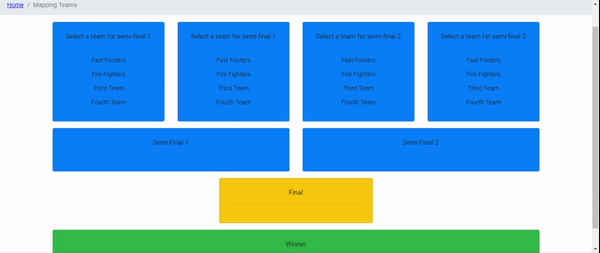
****

**Login Component**

****

**Mapping Component**

**Display to admin**

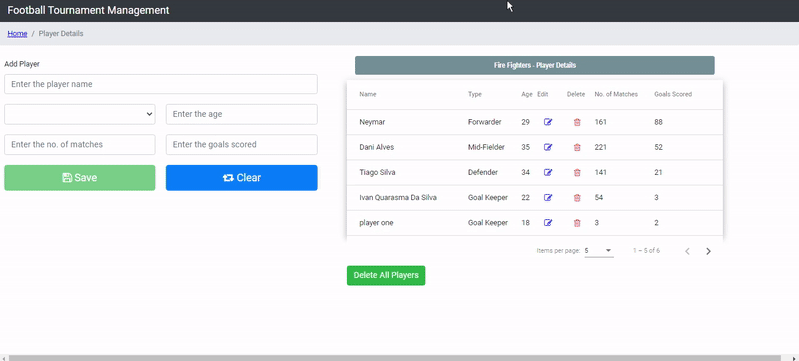


**Display to user**

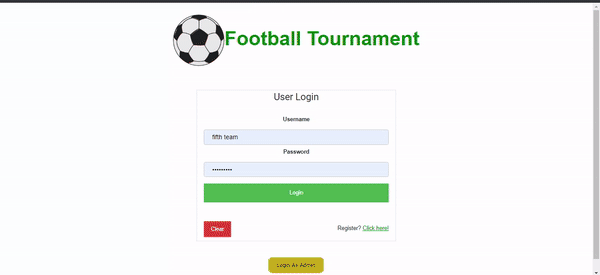


**Players Component**

**Display for both admin and user:**

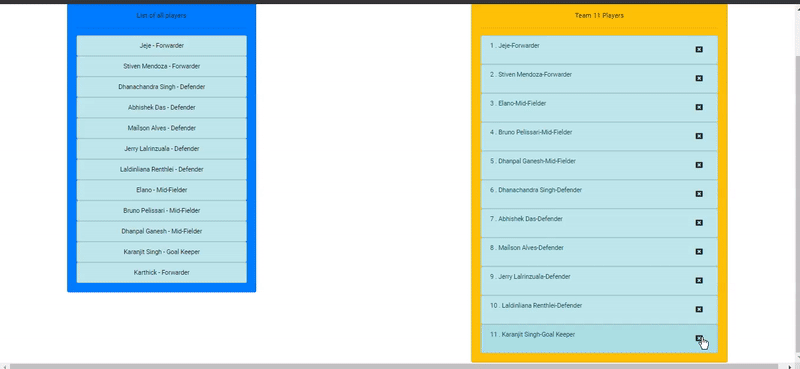
****

**Registration Component**

****

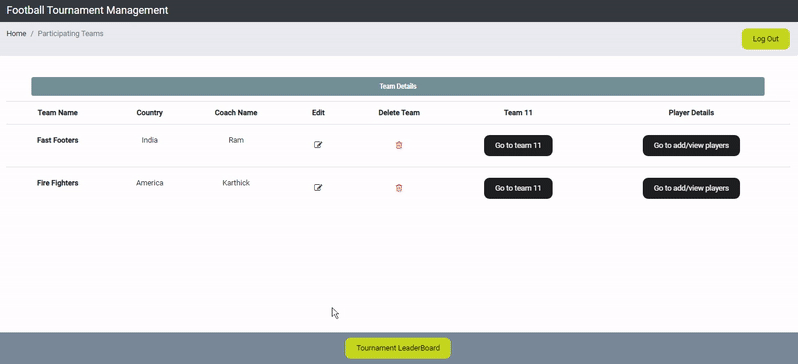
**Team11 Component**

**Display to both admin and user:**

****

**Viewdetails Component**

**Display to admin:**

****

**Display to user:**

****