**TASK 2**

**Tournament Management System**

**1-Planning Phase:**

**–Define Objectives and Scopes:**

The objectives and goals of this tournament and many and many and now we will discuss some of main goals. So let’s start:

1- provide and effective way for participate in different events and competitions by providing apply as individual or as a team.

2- also we tracked the result of each individual event and calculate points.

3- college admin have full control to all website and he can make another user to be admin with them and add new event or update existing one and so on..

4- after any actions the website will update data automatically to ensure flexibility of the website.

**–Define Systems Requirements:**

After Thinking a lot and lot .. the software application should have the following features:

**1- Participate Rules:**

- can choose how to apply on competition as a individual or team

- Team member is maximum 5 people .

- can create his own team and share team code with his friends to join to team .

- can enter team code to enter to an existing team.

**2- Competition Rules:**

1- participate can apply only once on certain competition

2- Allow 20 participate to join to the same competition

3-

**3- Event Rules:**

1- Determine the event type (acadimic, entertainment and so on ..)

2- tracking the event result and update the Ranking of Team or individual based on those results.

3- each competition should have their own events which should be at least 5 event per single competitions

**4- Ranking :**

1- provide a table in home page for individual ranking to ensure compete between participate.

2- provide another table for teams ranking and display it in home page also.

**2-Designing Phase:**

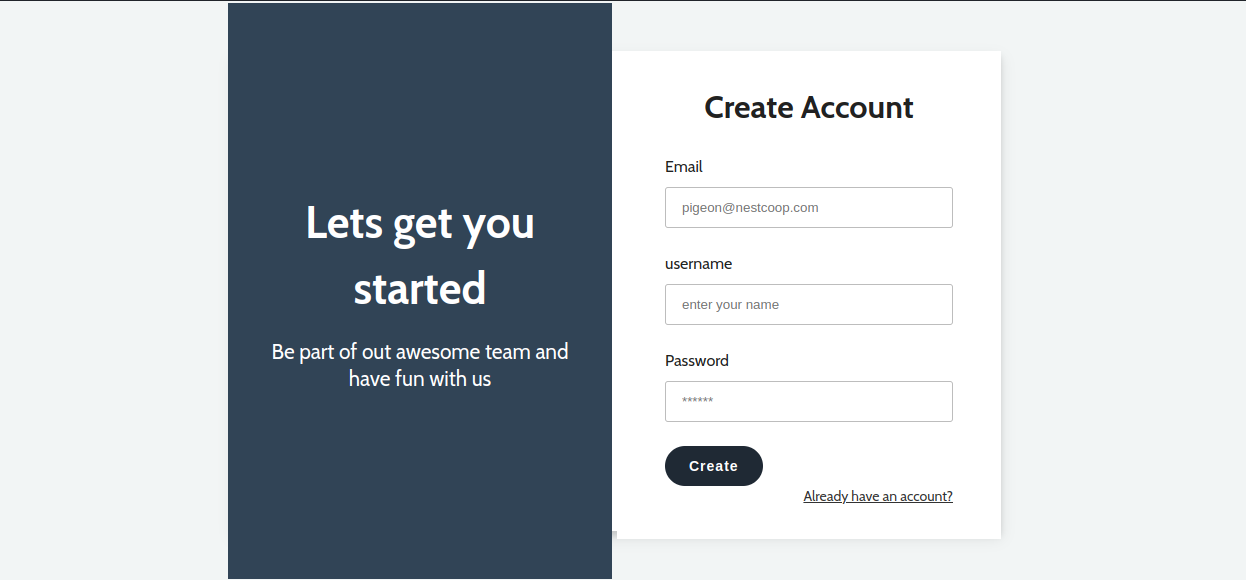
Designing phase is very important phase in SDLC because we ensure that the happiness of user is high. But user interface not only the main part in designing phase because there is also Digrams and database design which. So let’s discuss those in details:

**–User Interface:**

Now we will discuss the user interface in different parts of our system:

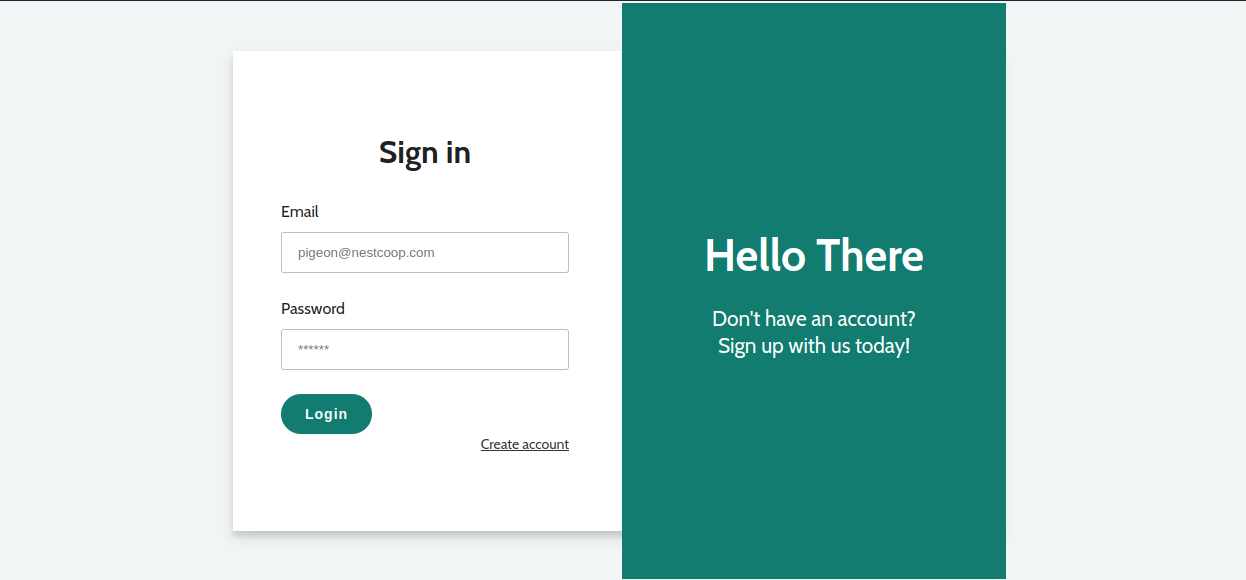
**1- Register Page:**

This page contain field that user will enter his data in to register to the system like email and username and password



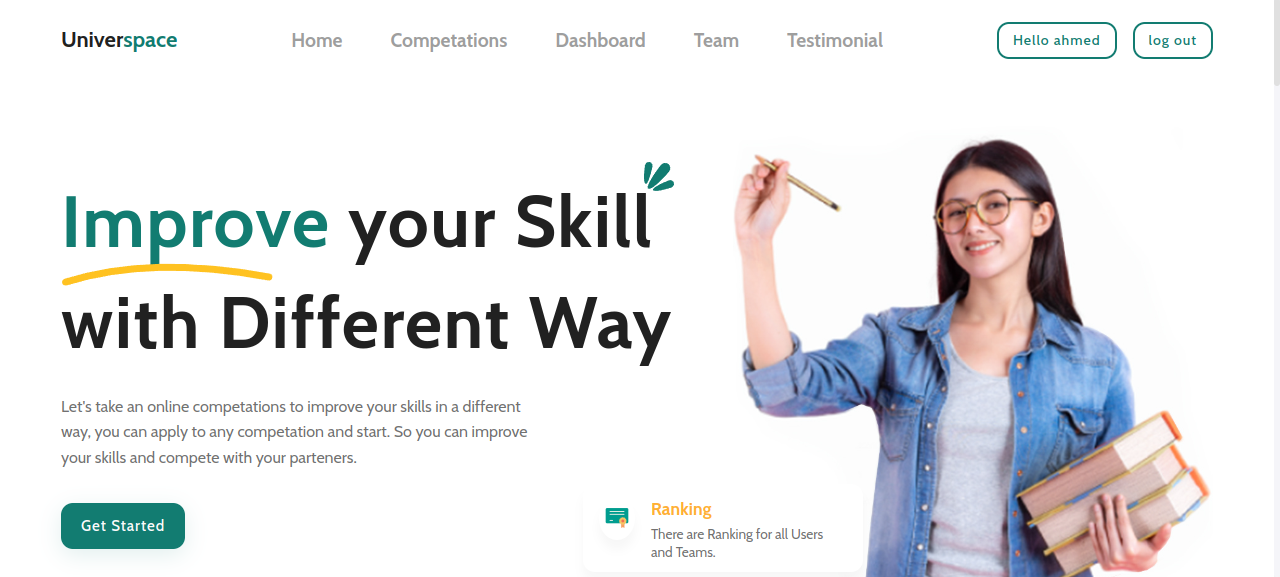
**2- Login Page:**

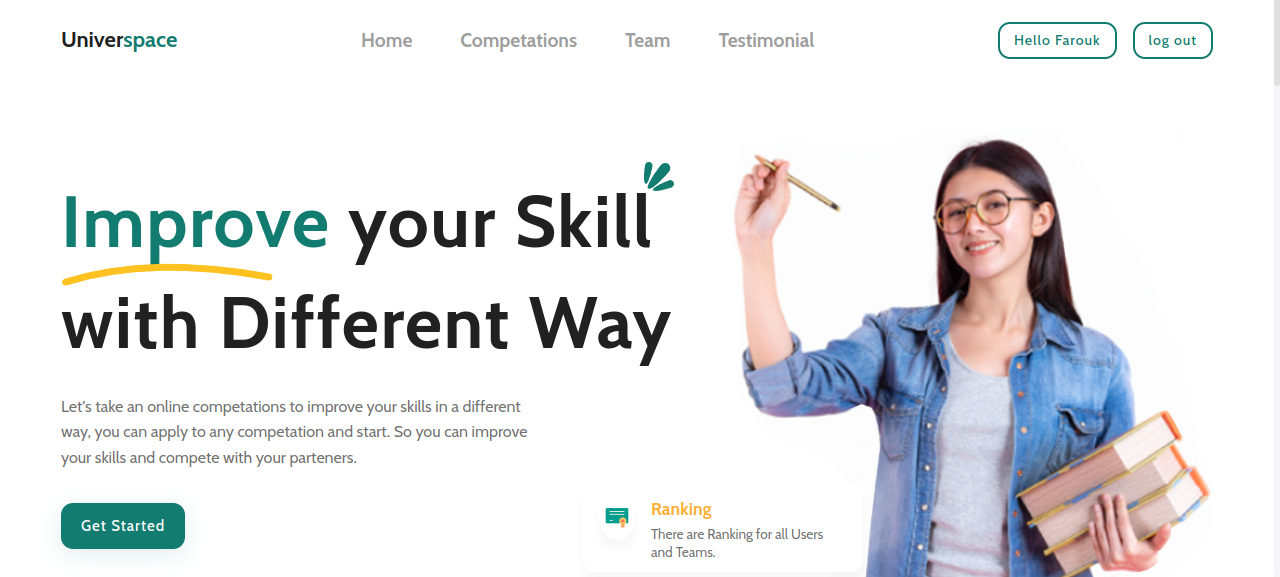
This page contain fields that user will enter his email and password to login to system after validate user data.

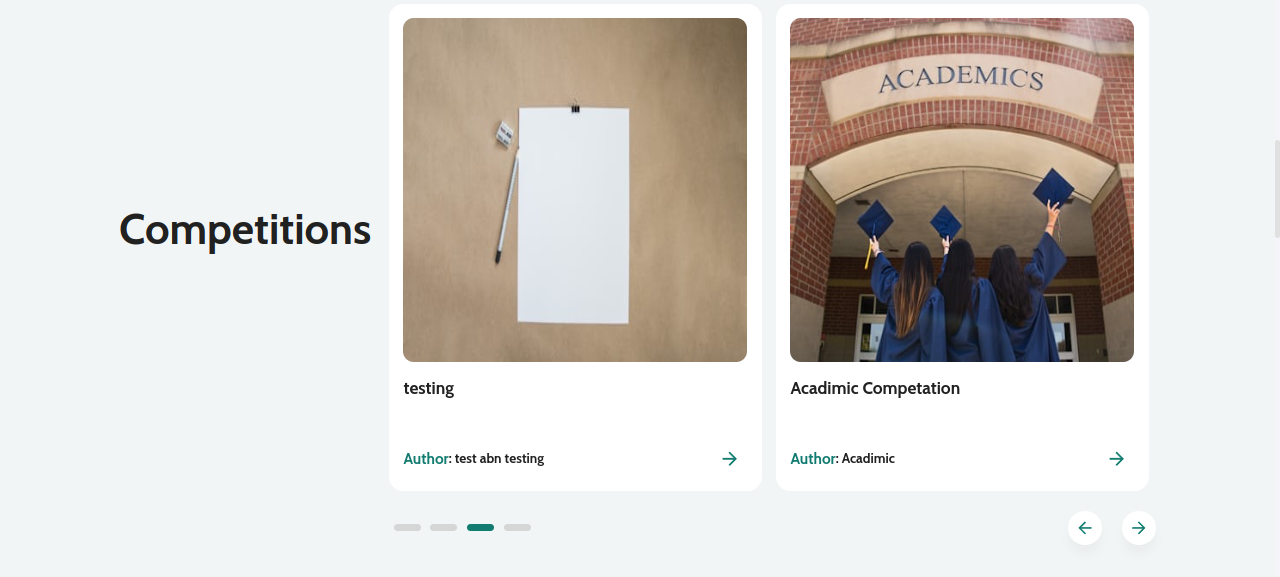


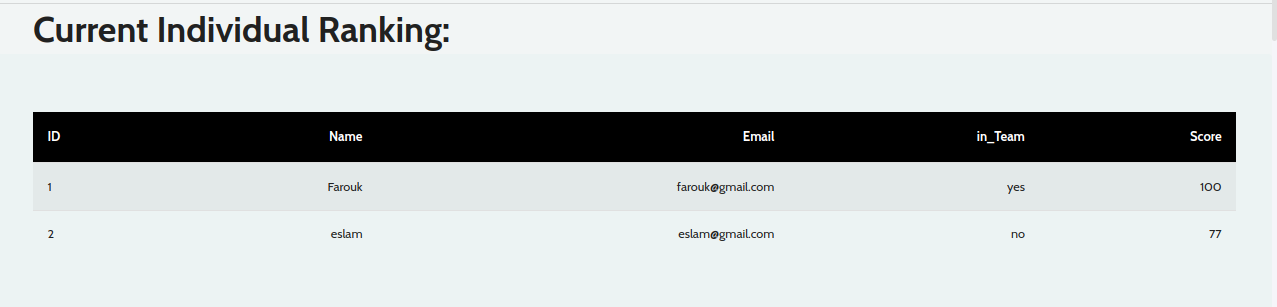
**3- Home Page:**

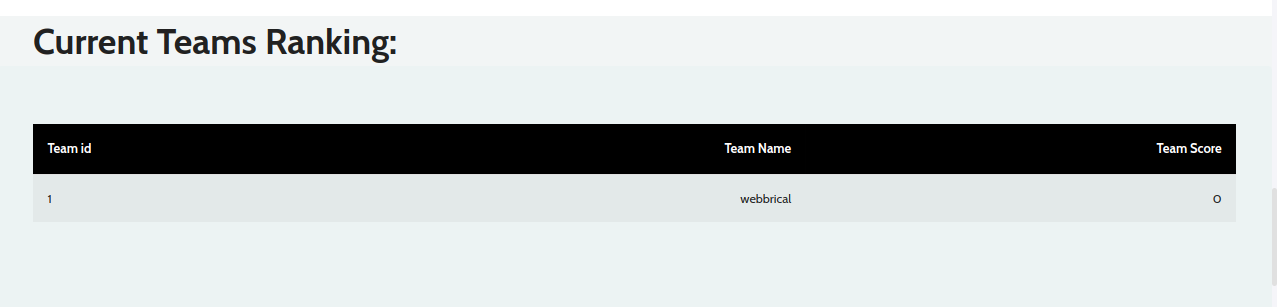
This is the main page of the system and it shows the dashboard link if user is admin and also shows the existing Competitions and current ranking for each individual and team and some of statics testimonial..

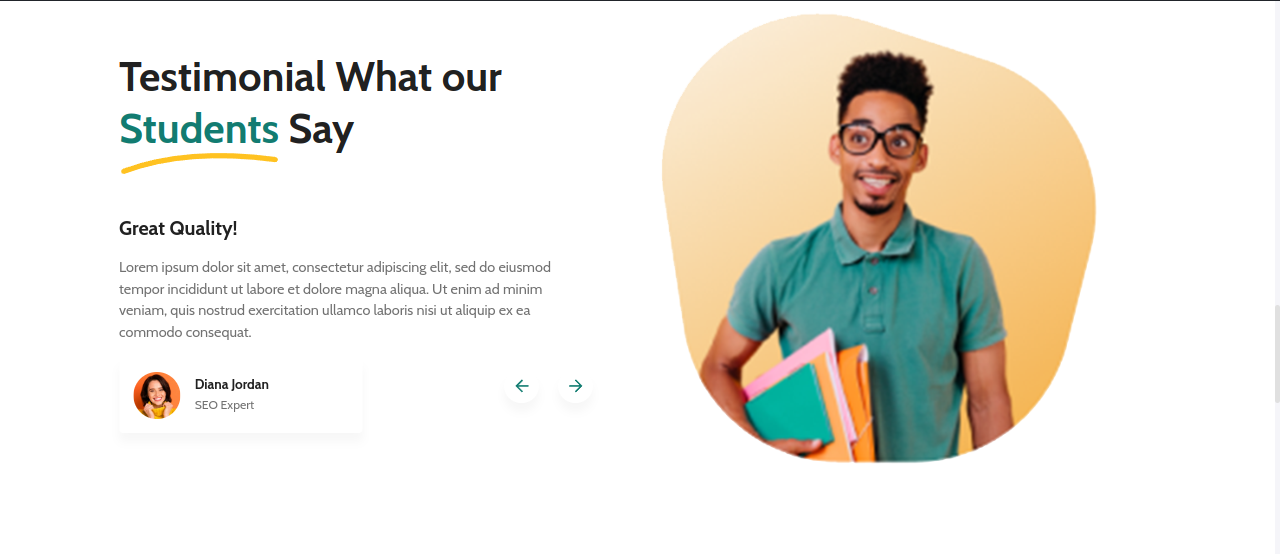
–shows Dashboard link if user is Admin:

– Hide Dashboard if it’s not Admin (also it prevented in authorization ):

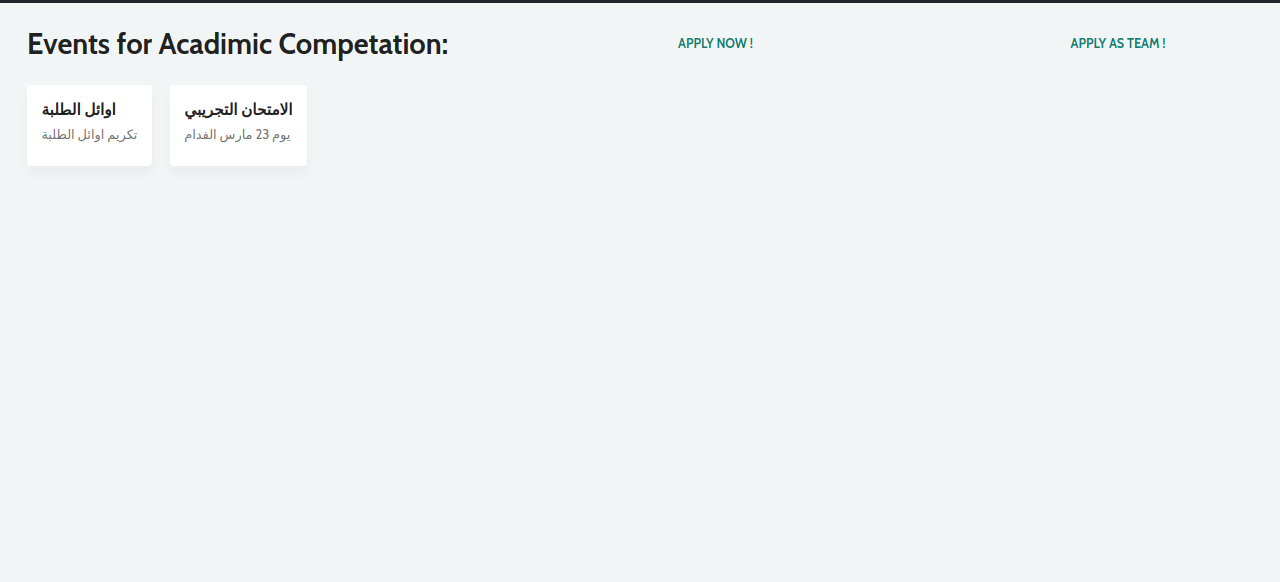
– Show Current Competitions with beautiful slider:

– show current individual ranking: 

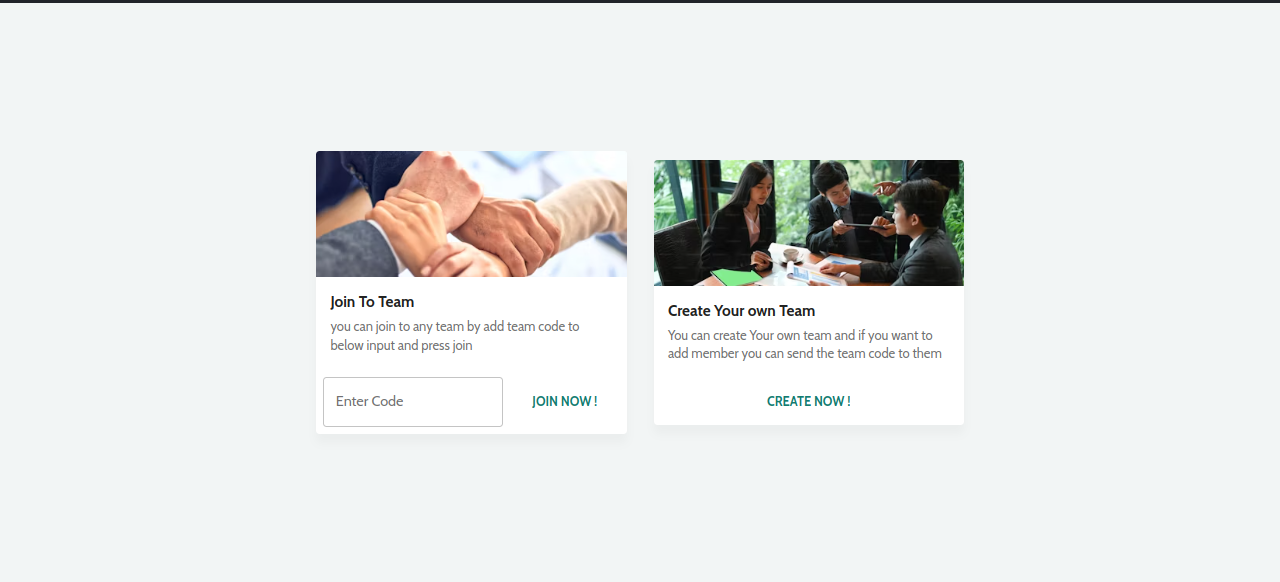
– show current team ranking:

– show testimonials: 

**4- Competition Page:**

Show a single competition with it’s events and buttons to make user apply on it as individual or as team:

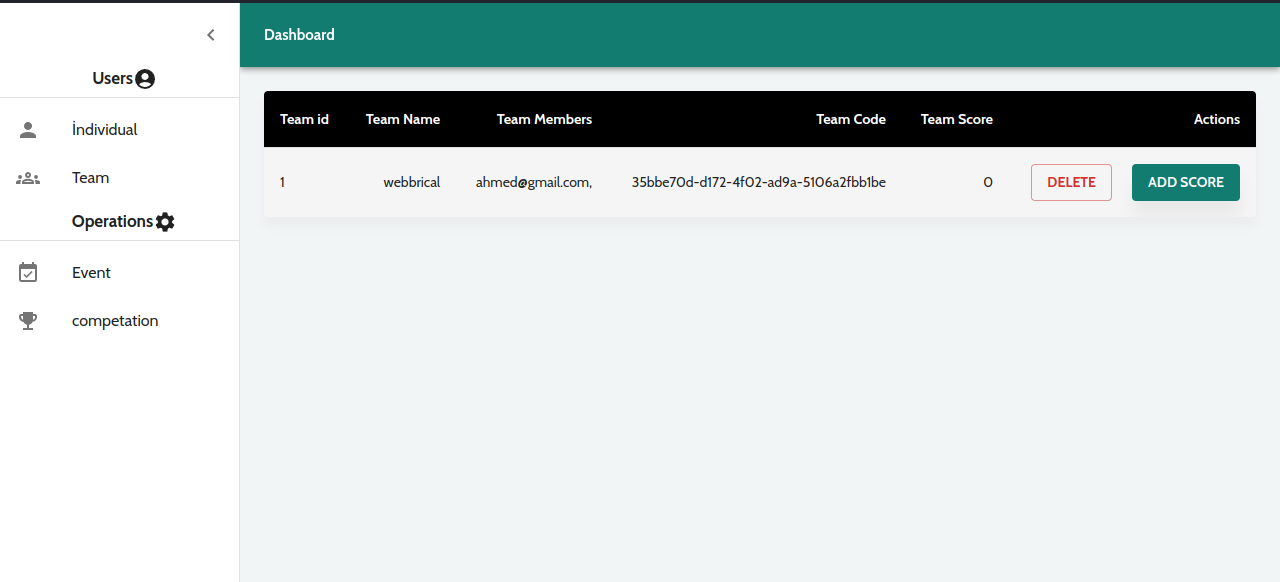
**4- Team Page:**

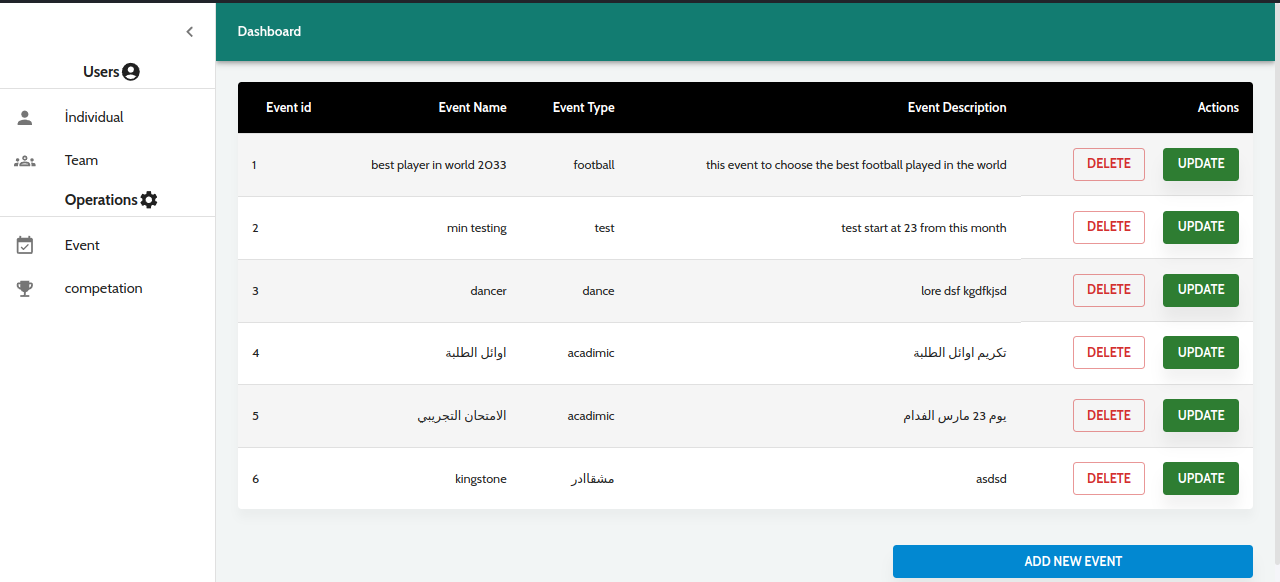
Allow to user to join to an existing team by entering team code or create his own team:

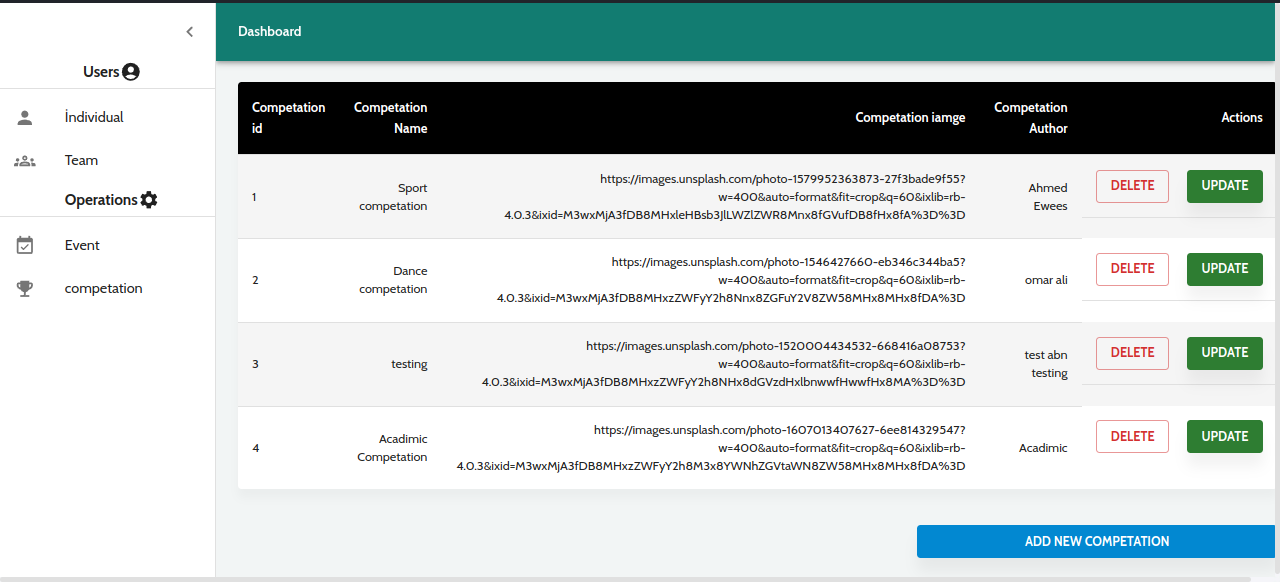
**5- Dashboard:**

This page allow to admin to control whole system by providing many features like:

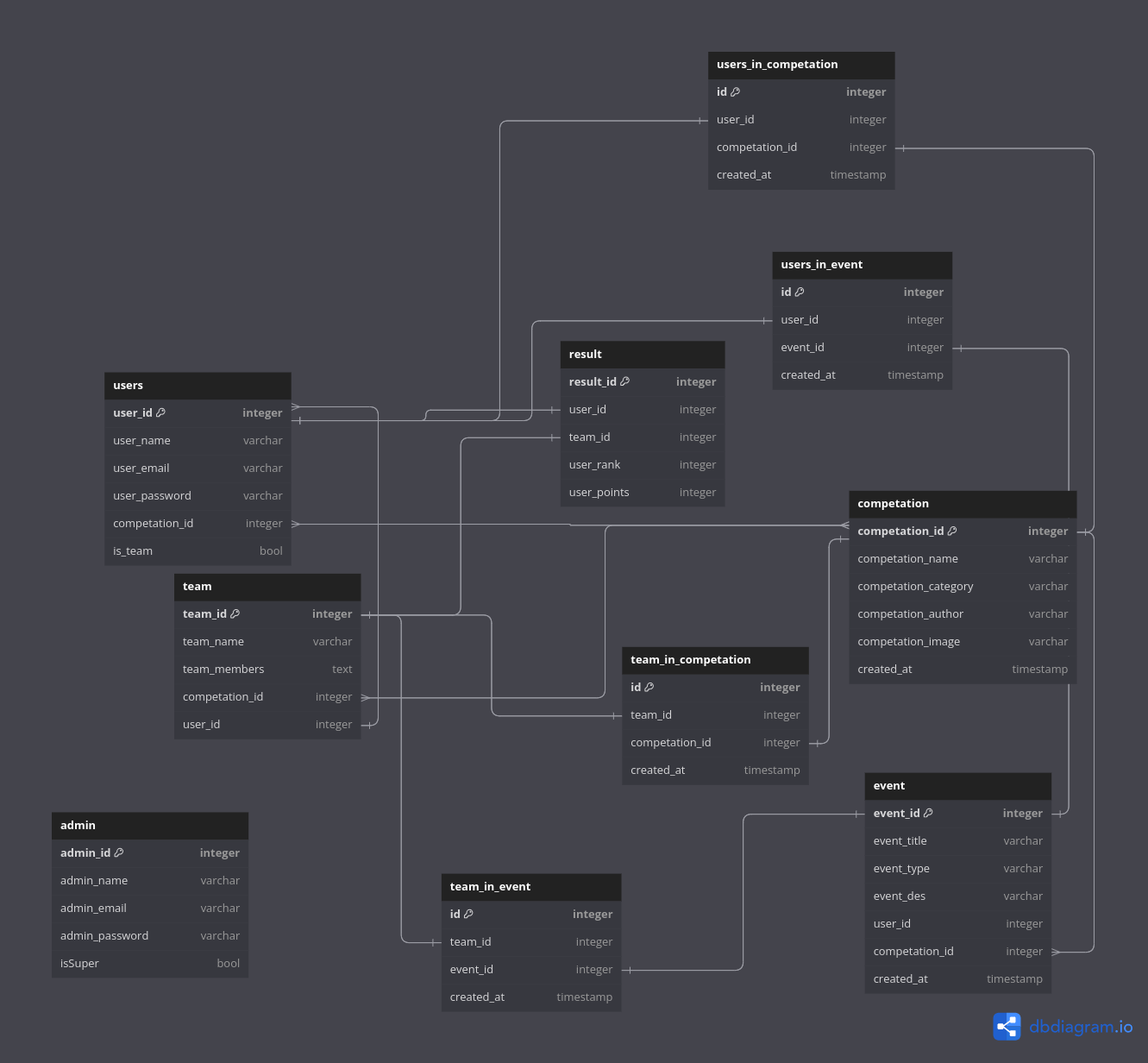
1- have a full access to user and he can delete user or make it admin or adding score to user to ranking up:

2- have a full access on teams and he can delete team or add score to team for ranking up in rank table:

3- can add event or delete and event or update event : 

4- can add a new competition or delete or update a competition:

**– Diagrams:**

Now we will discuss the database design with diagrams as following:

The above image shows the relation between tables. Let’s talk more about this image:

**1- Competition table:**

This table contains information about competition like id and name and author of it and image of competition

**2- Event Table:**

This table contains information about event like id and title and description and type of event and have a competition\_id which is a foreign key that refer to competition table to specify in which competition this event in.

**3- Team Table:**

In this table there is a information about all teams in system like team\_id and name and members of the team and team code that users can use it to apply in team and leader\_id which refer to the person that created this team and he only have access to team code to ensure security and also contain the score of the team.

**4-Team\_in\_competation Table:**

This table contains information about teams that already enrolled in competitions by adding the team\_id and the id of competition that the team enrolled in.

**5-Team\_in\_Event Table:**

This table contains information about teams that already enrolled in event by adding the team\_id and the id of event that the team enrolled in.

**6- Users Table:**

This table contain information about user like id and name and email and password and if is admin or not and if is superAdmin or not and if he is enrolled in team before or not and his current score.

**7-user\_in\_competition Table:**

This table contained information about the users that already applied in competition like user\_id and comp\_id

**8-user\_in\_event Table:**

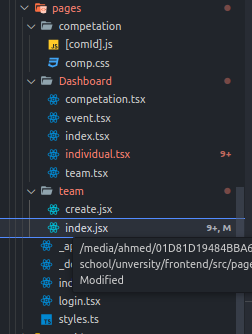
This table contained information about the users that already applied in event like user\_id and event\_id.

**3-Development Phase:**

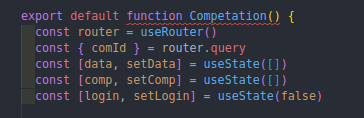
In this phase , the technologies that i use is node js and mySQl in back end and use next js to handle frontend part i used next js with typescript to handle unexpected errors in my code and ensure speed of the website. And i use node js and mysql with MVC design pattern to deal with backend and initialise my endpoints and ensure flexibility of website.

Even Laravel is known for its ability to handle data with high efficiently and offer ready architecture, but it is slow that makes user unhappy, when using the website. Now let’s dive into our code and explain some parts of it . so let’s start:

**1- Handle Routes Effectively in Next Js:**

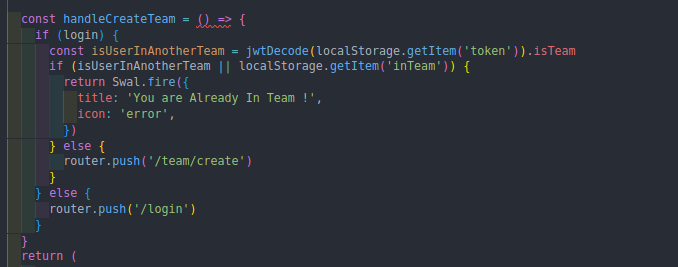


Next js provide a beautiful way to handle routes by adding the files in pages directory and it will handle route automatically based on file name. But if you want to make something more complex like nested routed or passing a params.. Well.. here is the magic solution. Next js deal with nested route by creating a directory for parent route and index.tsx will be the default for this route and any file in this directory will be sub route under parent. And if you want to take for example event id from url you can use the [] when you naming file.tsx and use useRouter hook and destructuring to deal with it.



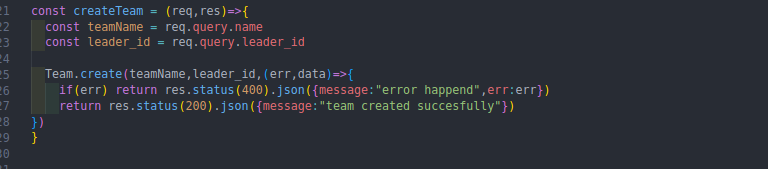
But how next js or react really handle routes behind the scene? Well it’s something called VDOM and VDOM is simply compy of real Dom that react or next uses to compare between them and update changes only which make webPage more flexible and fast as well.

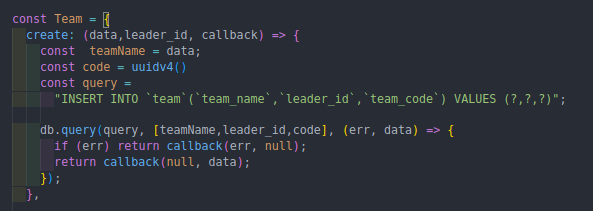
**2- Validation while create Team:**



Here when user click on create team button we will check first if user is logged in otherwise redirect him to login page but if he logged in we will check the JWT to check if user applied in another team before or not if not we will move him to create page :



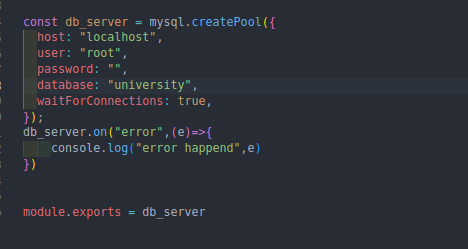
After redirect him to create page we will receive the team data and send it to server via axios and change the user state to be inTeam. 

Teamcontroller that will execute when we called axios and then move to the model Team after defining teamName And Leader\_id to send it via create functions as parameters 

**3- Backend Endpoints:**

We will show some of backend endpoints to be in picture with axios requests 

4- init connection with MYSQL database:



By defining host , user,password,database and handle errors and export it to use with models we now init the connections with our database.

**5- Handle Competition Creation form Admin Dashboard:**

Here i used sweetalert to handle create competition because it’s really beautiful and effective in handling errors and easy to maintenance and use by defining HTMLINPUTELEMENT and validate it before sending it to server was really nice and challengeable.

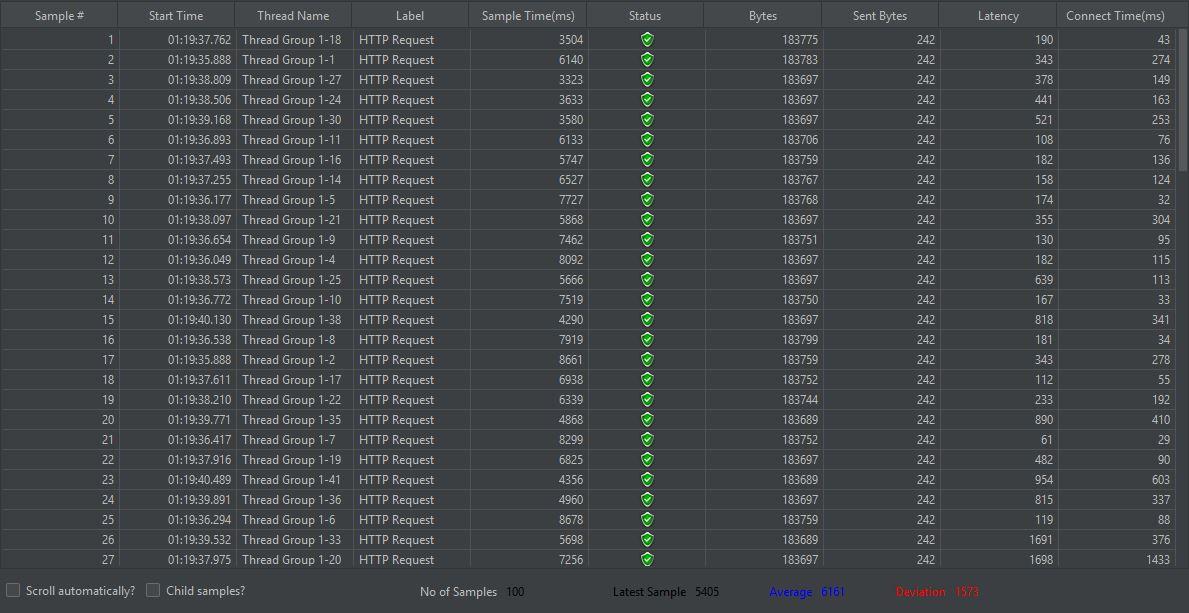
**6- using sweetAlert2 and react toastify for showing messages:**



**4-Testing Phase:**

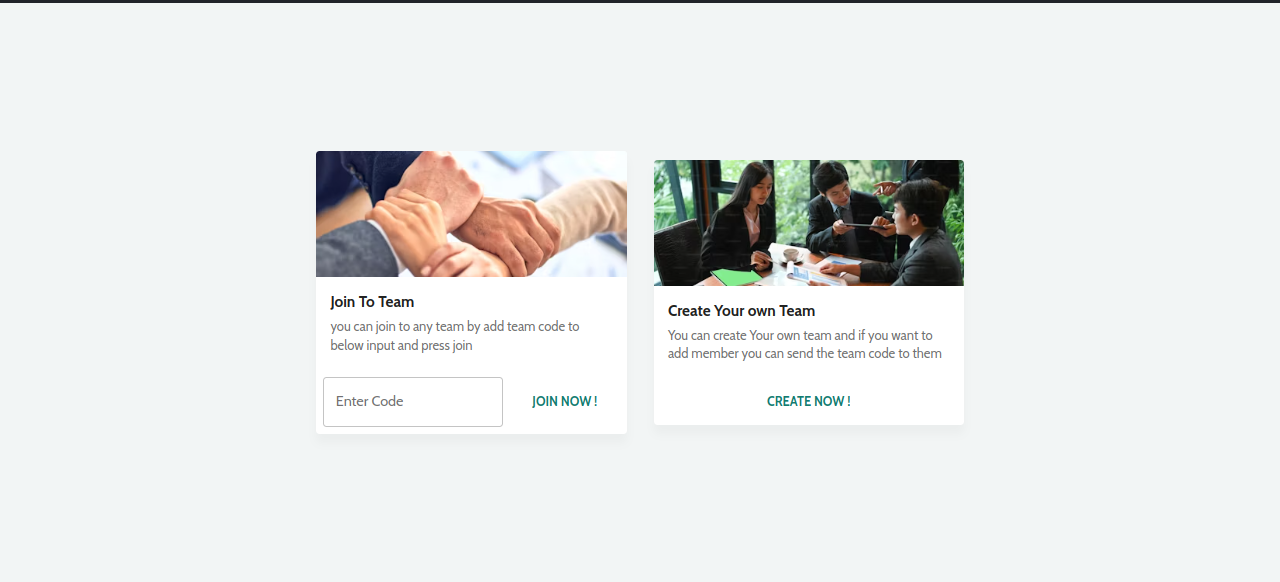
The testing phase in software development is a critical stage where the functionality, performance, and quality of a product are evaluated. It involves executing various tests to identify bugs, errors, and potential issues within the software. This phase encompasses a range of testing types including unit testing, integration testing, system testing, and acceptance testing. Test cases are designed based on requirements and specifications to validate the software's behaviour against expected outcomes. Through rigorous testing, developers ensure that the software meets quality standards, is stable, and functions as intended. Successful testing mitigates risks, enhances user experience, and contributes to the overall reliability of the software product. And now we identify the test case and what is expected output and the final result:

| Identify Test case | Expected Output | Final result |
| --- | --- | --- |
|  |  |  |
| Registration Page | Expected to created a email by entering email name and password | Test case: passed |
| Login Page | Expected to login to system after entering email and password and check if those data are in server | Test case: passed |
| Home page | Should home page render all competitions with their events | Test case: passed |
| Go to dashboard | Should return the dashboard page with its functions if only user is admin or superAdmin otherwise redirect him to home page with sweetAlert you are not authorized | Test case: passed |
| Add competition | This page should add competition with competition name and its type | Test case: passed |
| Add event | This page should add event with event name and its category | Test case: passed |
| create team | This page should create team and generate the team code | Test case: passed |
| Add score | This page should add score to a given team | Test case: passed |
| Show events rank | This page should show all users rank | Test case: passed |
| Add A New Admin | Should add a new admin | Test case: passed |
| Delete an Admin | Super Admin can delete any normal admin | Test case: passed |

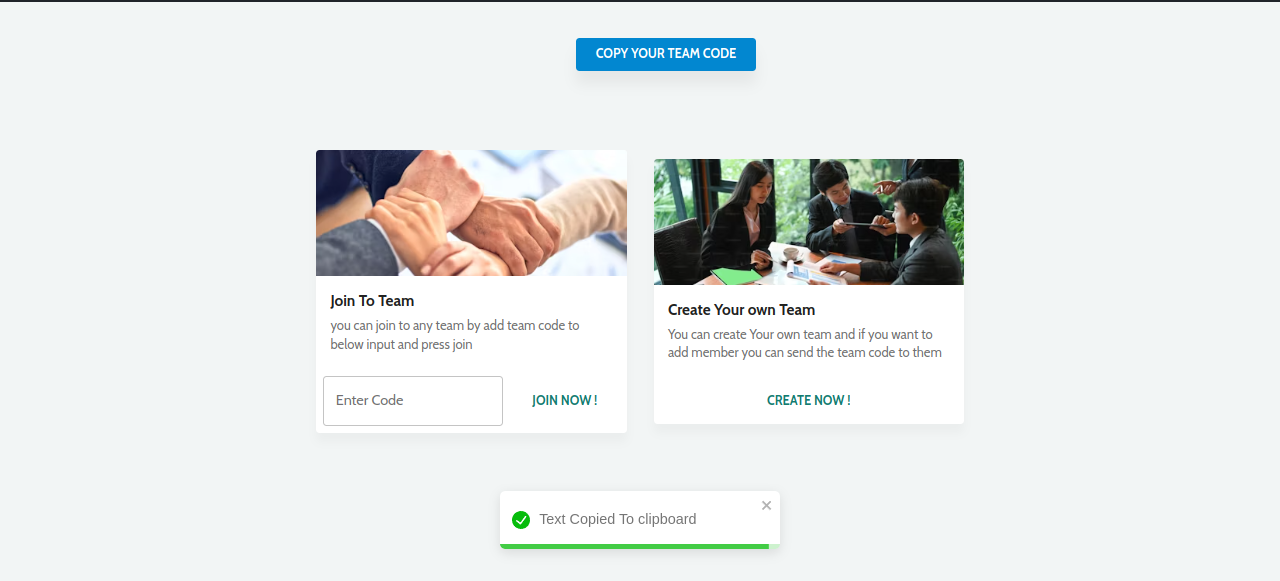
Adding jemeter testing to ensure request handling:

**5-FeedBack Phase:**

I received feedback from my partners and friends to make teamleader have access to copy team code after clicking on copy button:

Before: 

After :



**6-Deployment Phase:**To make the website accessible to all users, the website should be deployed to a certain server and has it domain name. I preferred using hostinger to get a server of a domain name. This website offered excellent servers and domain names with low expense.

**7-Maintenance Phase:**

During the maintenance phase in software development, it's like giving your favorite tool a tune-up. We're constantly updating, fixing bugs, and adding cool new features to keep the software running smoothly and meeting your needs. Think of it as giving your software buddy a little TLC to make sure it stays reliable and easy to use as things change.

**Evaluation:**

When we were designing the tournament, we really focused on getting the structure and user experience just right. We decided to use React for the frontend because we wanted a UI that's super responsive and fun to interact with. Going with a component-based architecture was a no-brainer because it helps us scale things up smoothly and keeps everything easy to maintain. And we made sure that whether you're entering solo or with a team, and no matter if it's a sports or academic event, it all fits perfectly with what the college needs.We opted for Node.js and Express on the backend to build a strong and reliable server, making sure it communicates flawlessly with the frontend. Security was a big concern, so we beefed it up by using bcrypt to encrypt sensitive user info like passwords. And to keep things safe and scalable, we went with JWT for authentication. Mongoose came in handy for managing our database, giving us the flexibility we needed to handle all our tournament data smoothly.

**Conclusion:**

I have put alot of effort to do this project. adding the option for single-event entry is my goal, and ironing out any remaining issues. We're determined to reach 100% completion because we're dedicated to providing a top-notch systemj that perfectly meets the needs of our college's event management.