

SEMESTER PROJECT REPORT

This is a report for the website we created for our web engineering course project.

Presented to
Sir Qaiser Riaz

Presented by
Hafsa Imran
Hamna Naveed
Muhammad Danish
Muhammad Taha Naveed





TABLE OF CONTENTS

Introduction 3

About the website 4

Frontend Details 5

Backend Details 6

Database Details 7

Website Layout 8

User Requirements13

Conclusion 14



Introduction



In this Web Engineering course, we learned how to develop static and dynamic websites using different languages, tools, frameworks, and technologies including HTML/CSS, Bootstrap, JavaScript, server-side programming with PHP and Laravel, and development of single page web applications using MERN stack. After learning all these skills, we were asked to create a website of our own using the aforementioned skills. This report describes in detail what skill we chose, why we chose it and what our website looks like.

The website we have engineered this semester is basically a social networking website. It is a chat and networking platform that helps to manage, grow and connect communities through messaging, content and discovery. The name that we chose for our website is "CHATTER".

Chatter is a chat website intended for developers which is made in react, so that users get real-time notifications and chat messages without having to refresh or render a new page. The motivation behind the development of Chatter is to make community messaging, collaboration and discovery as smooth and simple as possible. You can easily create, organize and grow your communities, inviting others to join just in one click.

The website is made using the MERN framework. We found this framework very interesting to work with and therefore chose a chat website as a great project to best showcase the power of React with its real-time notifications.

ABOUT THE WEBSITE

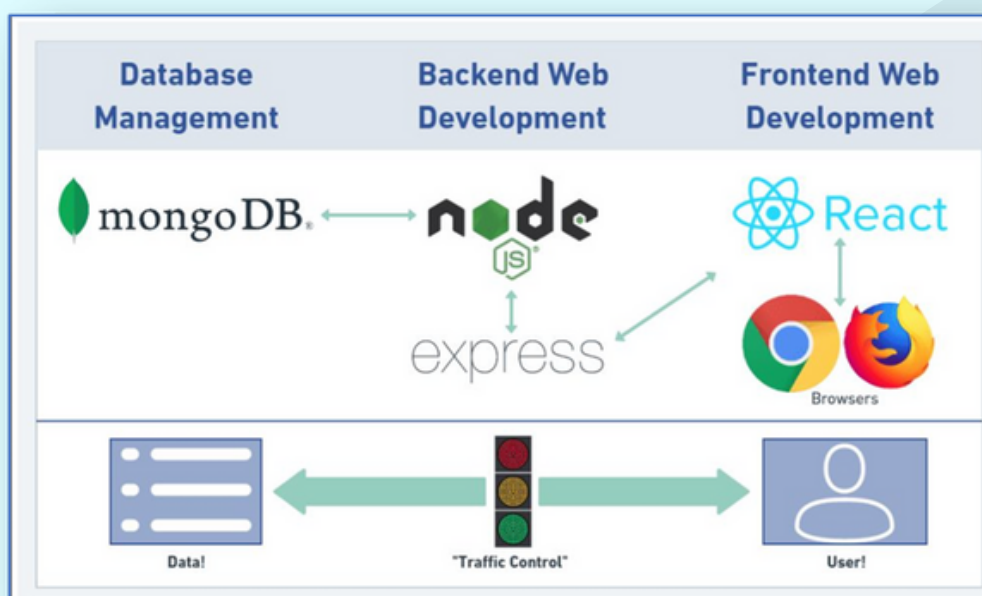
As mentioned earlier, Chatter has been developed using the MERN framework. The reason behind this selection is that MERN stack technologies provide a flexible data model with a dynamic schema. These features come up with powerful GUI and command-line tools. Finally, developers find that the MERN stack technology is faster, better, and more customized to build applications.

The way this website works is that it allows users to communicate in two-way persistent chats with one or multiple participants.

It allows communities, groups, or teams to contribute in a shared workspace where messages and digital content on a specific topic are shared. Team members can be added by a team administrator or owner. It allows admins and teachers to set up groups for classes, professional learning communities (PLCs), staff members, and everyone.

Architecture

The architecture used is the MVC (Model-View-Controller) Architecture. It emphasizes a separation between the software's business logic and display. This separation of concerns provides for a better division of labor and improved maintenance.



Frontend Details

REACT

Creation of reusable UI components: React allows developers to create reusable components that can be composed to build complex UI.

Client side-routing: Client doesn't have to reload the full webpage to get a new page each time a user makes a new request. Instead, React intercepts the request and only fetches and changes the sections that need changing without having to trigger a full page reload.

REACT COMPONENTS AND WEB PAGES

<FullLayout />

<Starter />

<About />

<Alerts />

<Badges />

<Buttons />

<Cards />

<Grid />

<Forms />

<Rooms />

<Tables />

<Chat />

<Login />

<Signup />



Backend Details

EXPRESS and NODE

- EXPRESS SERVER

Express is a popular web application framework for Node.js that is often used in React projects. It provides a simple and flexible way to create web servers and handle HTTP requests and responses.

- **Handle back-end logic:** The server can be used to handle backend logic, such as connecting to a database or processing form submissions.
- **Handle routing:** The server can be used to handle routing for the application, mapping incoming requests to the appropriate handler function.
- **Enable server-side rendering:** The server can be used to render the React application on the server and send the fully-rendered HTML to the client, which can improve the performance

- SOCKET IO

Socket.IO is a library that enables real-time, bidirectional communication between clients and servers. It is often used in React projects to enable real-time communication between the client (usually a web browser) and the server.

- CORS

CORS (Cross-Origin Resource Sharing) is a security feature implemented by web browsers that prevents a web page from making requests to a different domain than the one that served the web page.

The CORS library is often used in React projects to enable the server to send the appropriate CORS headers in its responses. This allows the React application to make HTTP requests to the server from a different domain or port than the one the server is running on.

Database Details

MongoDB

MongoDB is a popular NoSQL database that is often used in React projects to store and manage data.

Flexible and scalable database

Simple and intuitive API

Integrates well with Node.js

Strong community support

API Calls

GET

Get a user: /api/users/userId

POST

Register User: /api/users/register

Login User: /api/users/login

Create Room: /api/rooms/create

DELETE

Delete user: /api/users/delete

PUT

Join room: /api/rooms/join/roomId/userID

Leave room: /api/rooms/

Documents

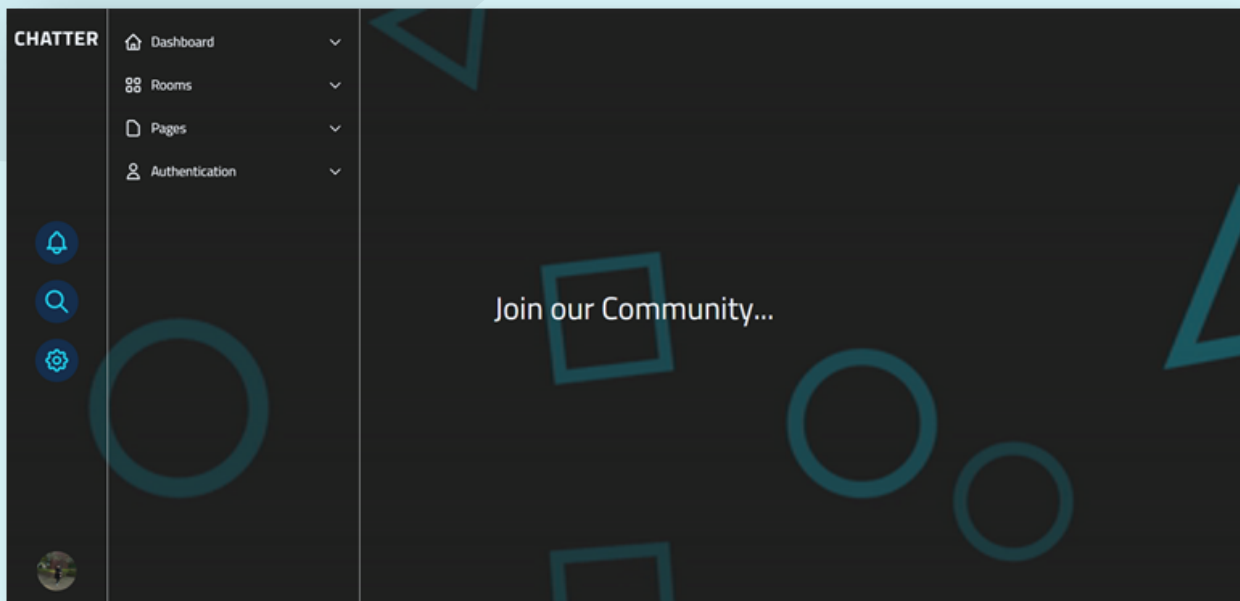
- User
- Messages
- Room



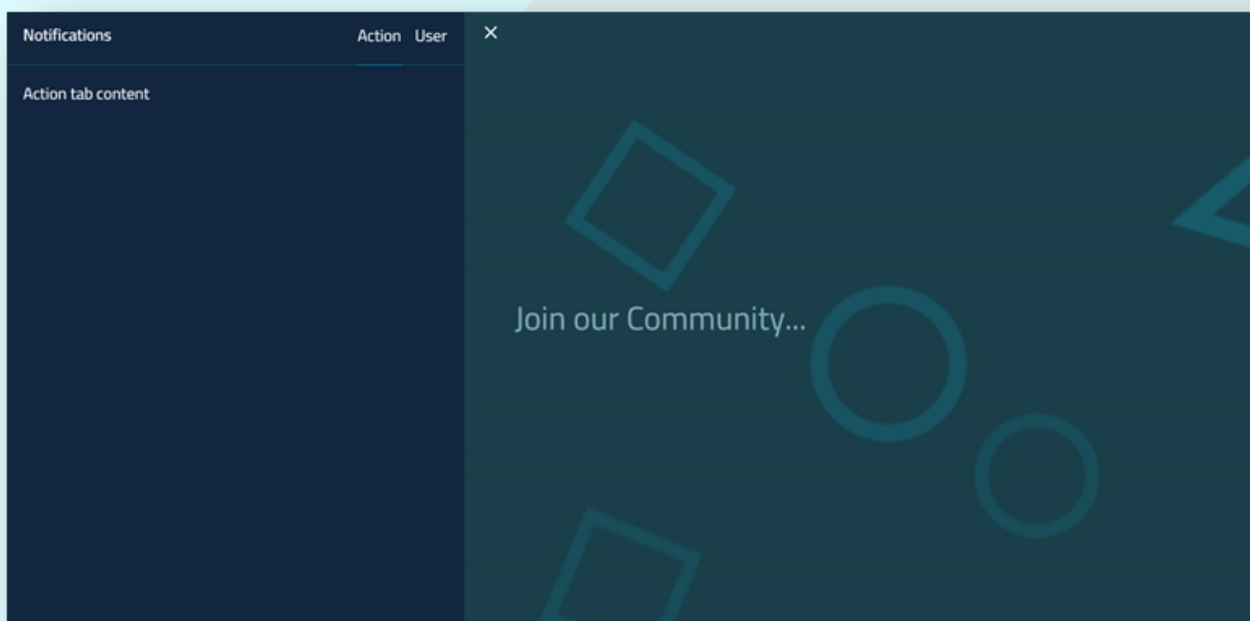
Website Layout

The website consists of a two column navbar on the left side with an attractive black and green dynamic background. The navbar guides the user to the different pages and components of the website.

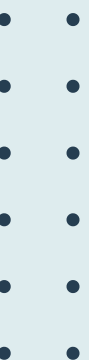
- Landing Page:

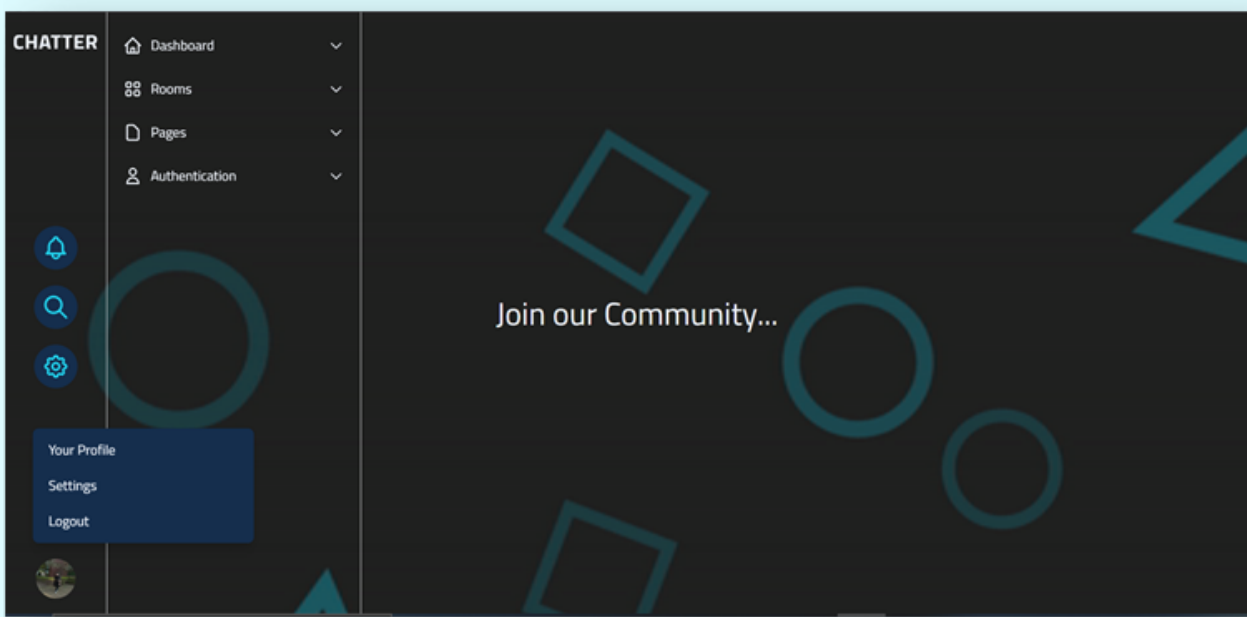
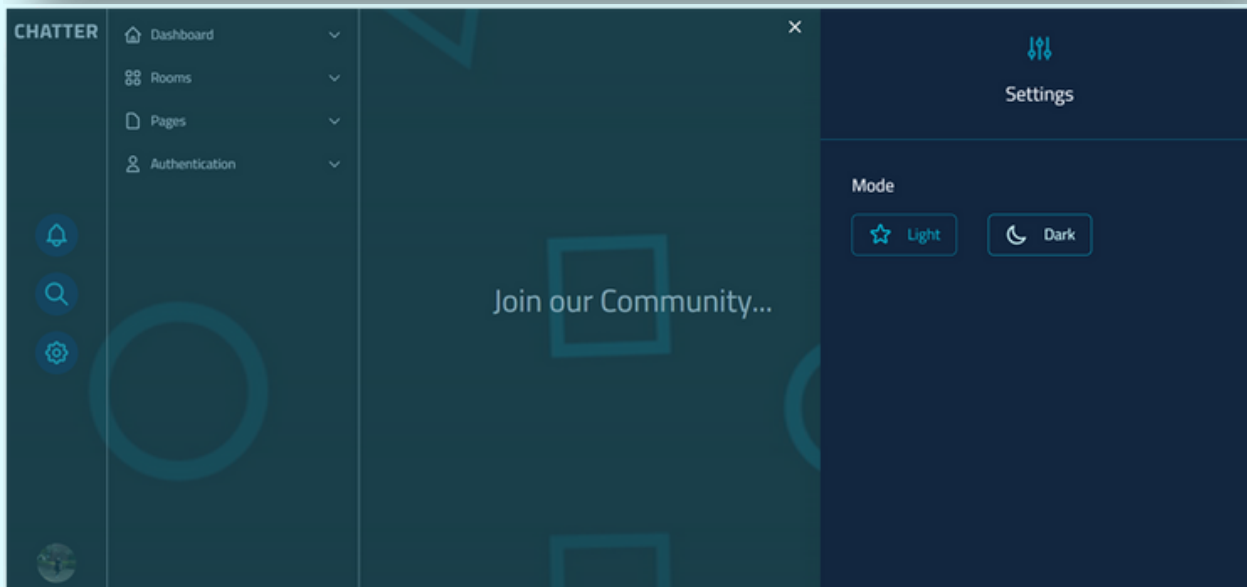
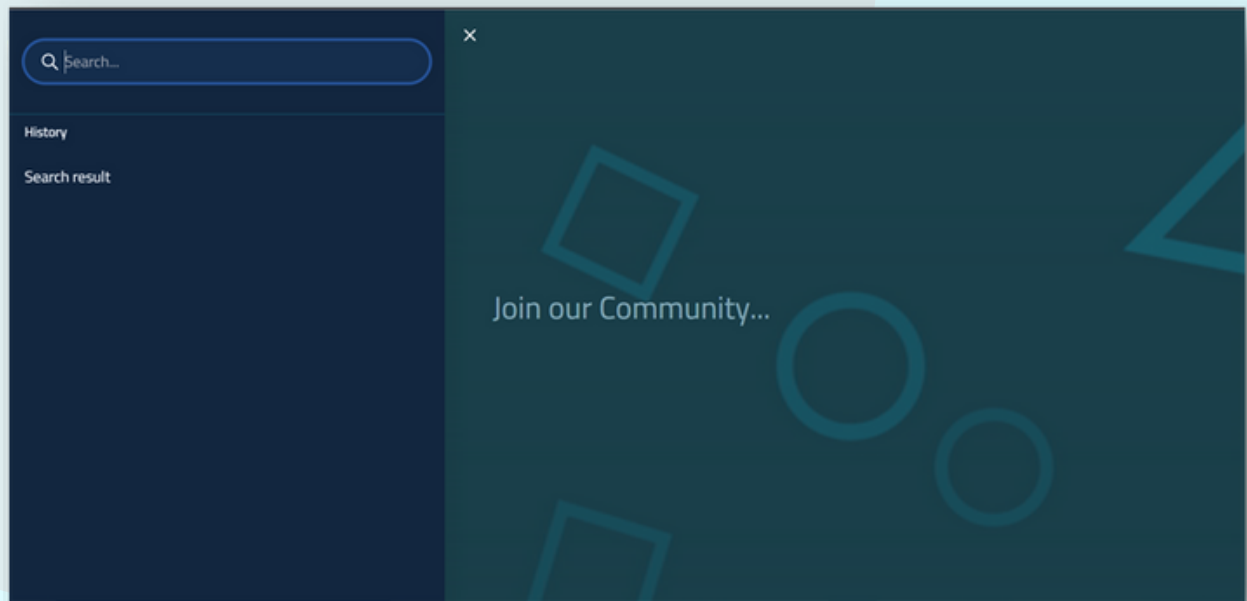


The dynamic landing page refers to the different dynamic pages.

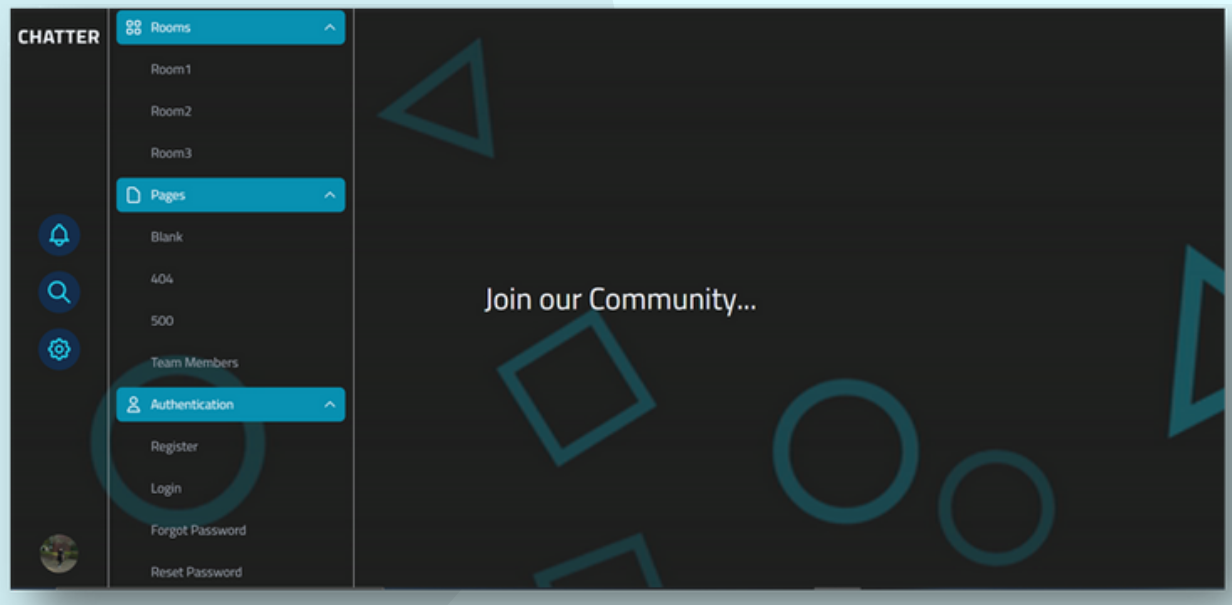


The mini column displays the name of the website at top and provides an attractive display of notification, search and setting panel option to the user.





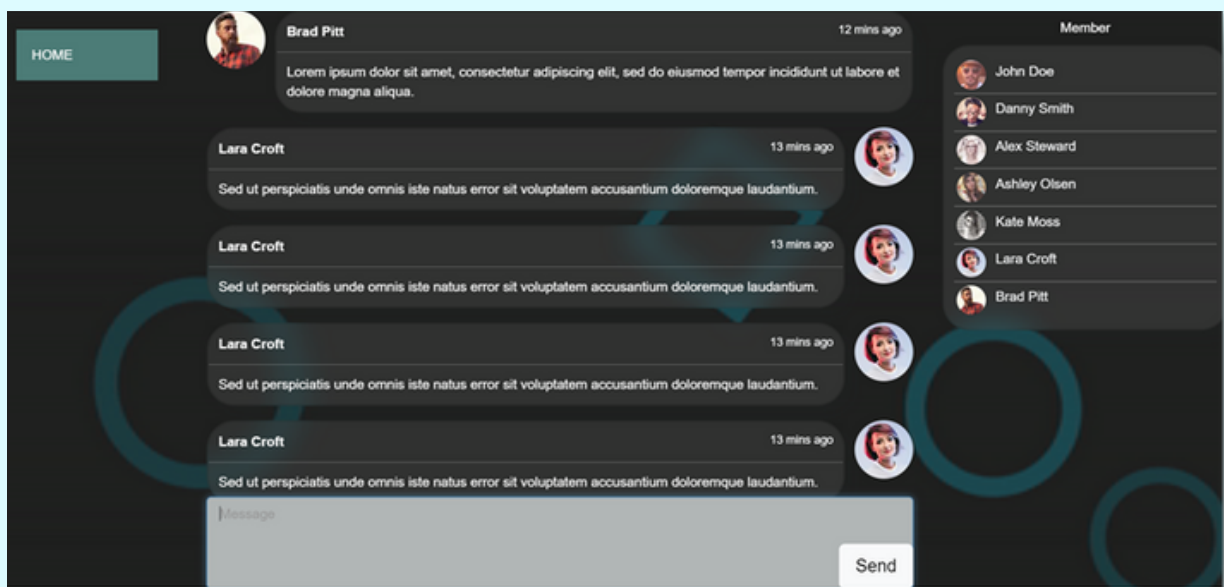
At the bottom of the mini column, there is a small display of user, further giving the options of profile, setting and logout.



The second column provides the options of chat rooms, about us, login related options. The dynamic landing page refers to the different dynamic pages.

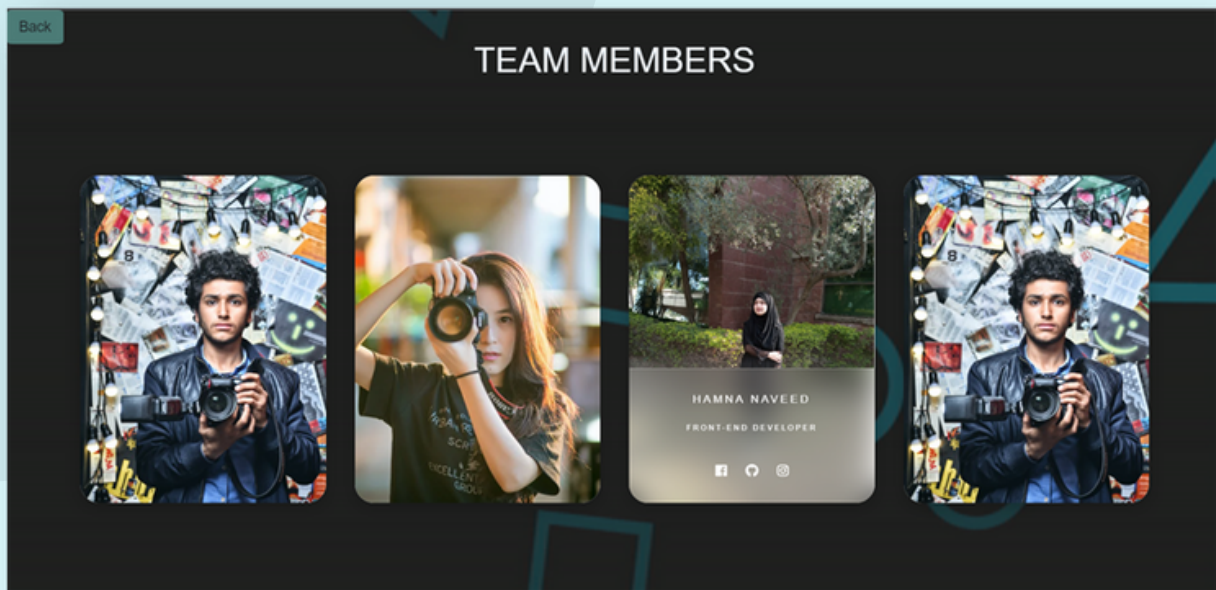
- **CHAT ROOM:**

The chat room consist of two main sections. One is to display the chat area and the other, is to display the members of the room. This dynamic page allows auto scroll to end of chat and to text area, a sticky text area and a sticky members display on scroll.



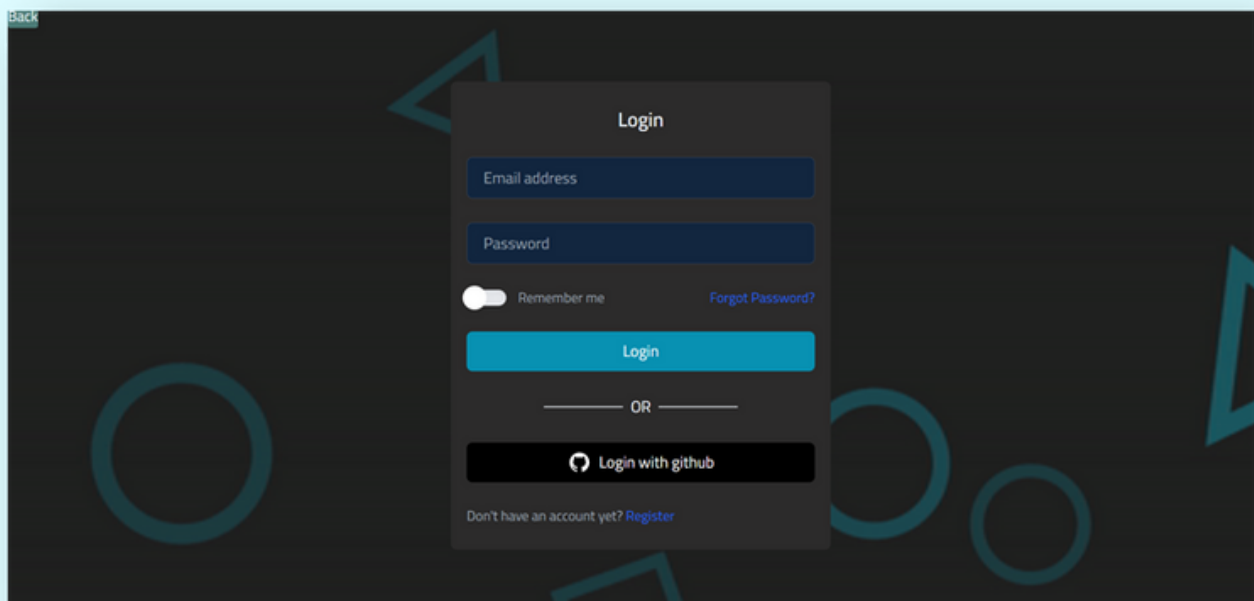
- **ABOUT US:**

This page consists of four cards that adjust them dynamically according to the screen size. The card displays the image of the team member and on hover displays the info about that member and social links.



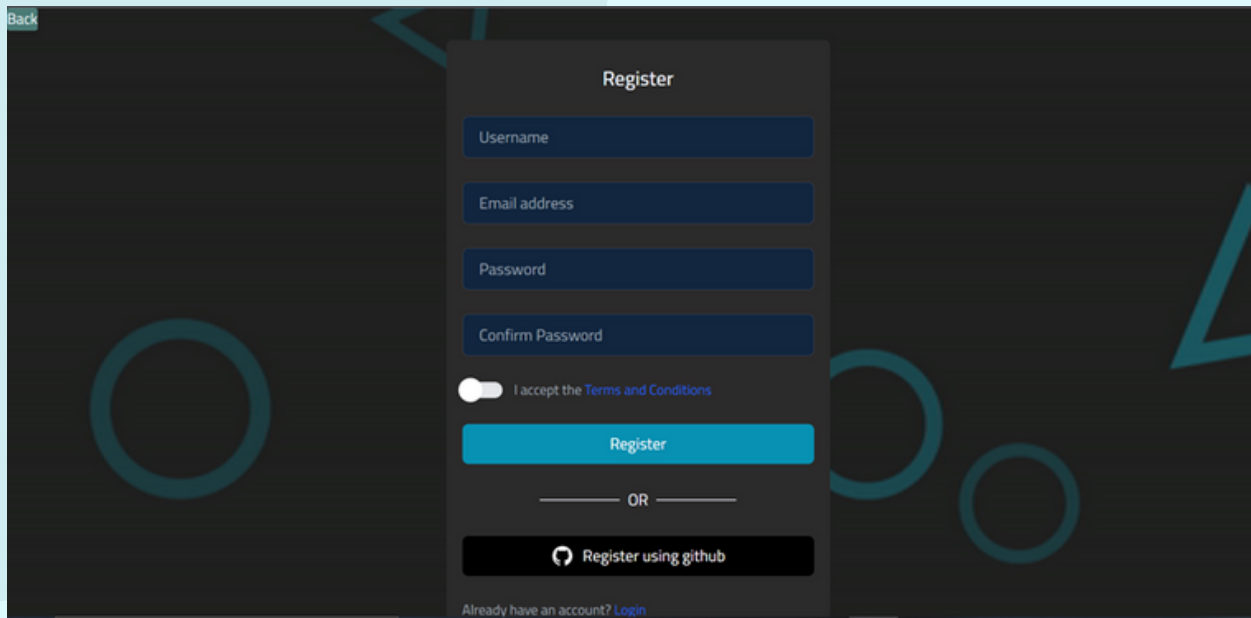
- **LOGIN:**

This page provides an attractive login form that provides the option to login with credentials or GitHub account. It also provides the link to forget password and register page.



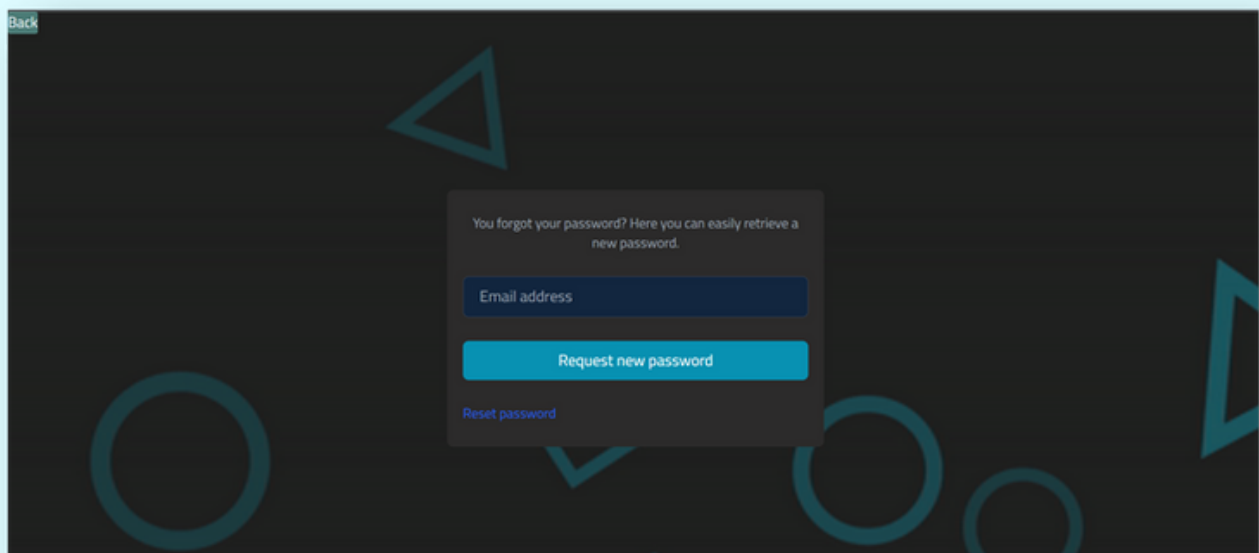
- **REGISTER:**

This dynamic page provides a user form that helps to register with credentials or GitHub account.

A screenshot of a web application's 'Register' form. The form is centered on a dark background with abstract teal geometric shapes. It features a 'Back' link in the top left corner. The form fields include 'Username', 'Email address', 'Password', and 'Confirm Password', each with a dark blue input box. Below these is a toggle switch for 'I accept the Terms and Conditions'. A prominent blue 'Register' button is positioned below the toggle. Underneath the button is an 'OR' separator. Below the separator is a dark button with a GitHub logo and the text 'Register using github'. At the bottom of the form, there is a link that says 'Already have an account? Login'.

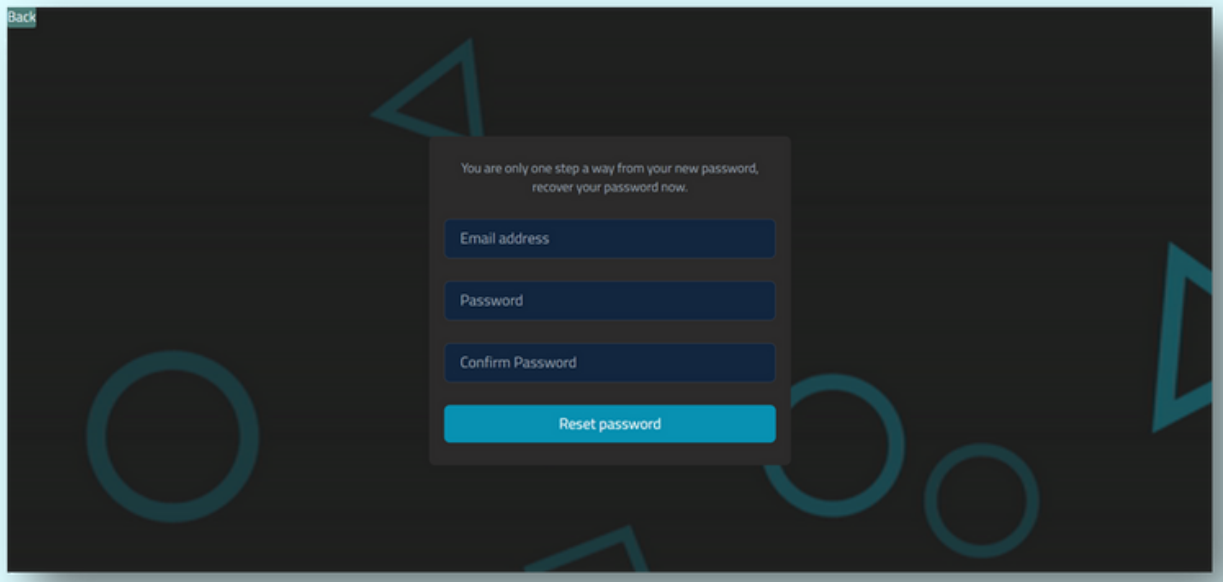
- **FORGET PASSWORD:**

This dynamic page provides a user form that helps to retrieve the forgot password.

A screenshot of a web application's 'Forget Password' form. The form is centered on a dark background with abstract teal geometric shapes. It features a 'Back' link in the top left corner. The form contains a text area with the message: 'You forgot your password? Here you can easily retrieve a new password.' Below this is an 'Email address' input field. A blue 'Request new password' button is located below the input field. At the bottom of the form, there is a link that says 'Reset password'.

- **RESET PASSWORD:**

This dynamic page provides a user form that helps to change the password.



USER REQUIREMENTS

The following are the functional and non functional requirements that the user would want and need.

- Live notification
- Sticky room members display on scroll
- Autofocus to text area
- Sticky text area on scroll
- Creation of room
- Adding participants to room
- Login with GitHub account
- Change room setting



CONCLUSION

- Usage of MERN stack offered us hands-on experience of websites that use real-time notifications
- Use of a No-SQL database like MongoDB offered us how unstructured data is better for scaling.
- Express together with NodeJS offers reliable back-end connectivity and easy to implement logic.

