CHAT-APP

(Using Node JS)

Project report (CA3) submitted in fulfilment of the requirements for the Degree of

BACHELOR OF TECHNOLOGY

in

COMPUTER SCIENCE AND ENGINEERING

By

TATIKONDA SAI MURAHARI

11913079

SUBJECT

INT222 – ADVANCE WEB DEVELOPMENT

School of Computer Science and Engineering

Lovely Professional University

Phagwara, Punjab (India) November

2022

CHAPTER 1

INTRODUCTION

The chatting application has huge impact on day to day life. There are numerous chatting application available in this world. Each application has different additional features varying from other applications. These application organizations compete with each other and add some competing features during each release. They have reached people much and have an impact on people's life. People find a better application from an available internet application which they feel much reliable and secure. Some of the available chatting applications that are available in these days are Whatsapp, Facebook, Instagram, Hike, etc...The above mentioned applications have billion users all over the world. Those companies are one of the top companies in the world. They have higher revenue per year and have many employees for their organizations developing additional features to compete with other organizations during their each release. These applications have different features and follows different ways to ensure security of their user data. Today a data theft is the major crime and most people are involved in it. There are many cases being filed these days about personal data loss. So the organizations have to ensure the security from data loss by the third party data crisis. The basic chatting system should involve both sending and receiving processes simultaneously. In this application both sending and receiving messages simultaneously happens through MERN concept.

Chatting app **allows you to communicate with your customers through chat**. It enables you to send and receive messages. Chatting apps make it easier, simpler and faster to connect with everyone and it is also easy to use. There are many types of chatting apps and every one has its own format, design and functions. The "Online Chat Application" has been developed to override the problems prevailing in the practicing manual system. This software is supported to eliminate and in some cases reduce the hardship faced by this existing system. Moreover this system is designed for the particular need of the company to carry out operations in a smooth and effective manner. Chat applications allow you to stay connected with other people who may be using the application even on the other side of the world. In customer service, such applications are one of the most important communication channels.

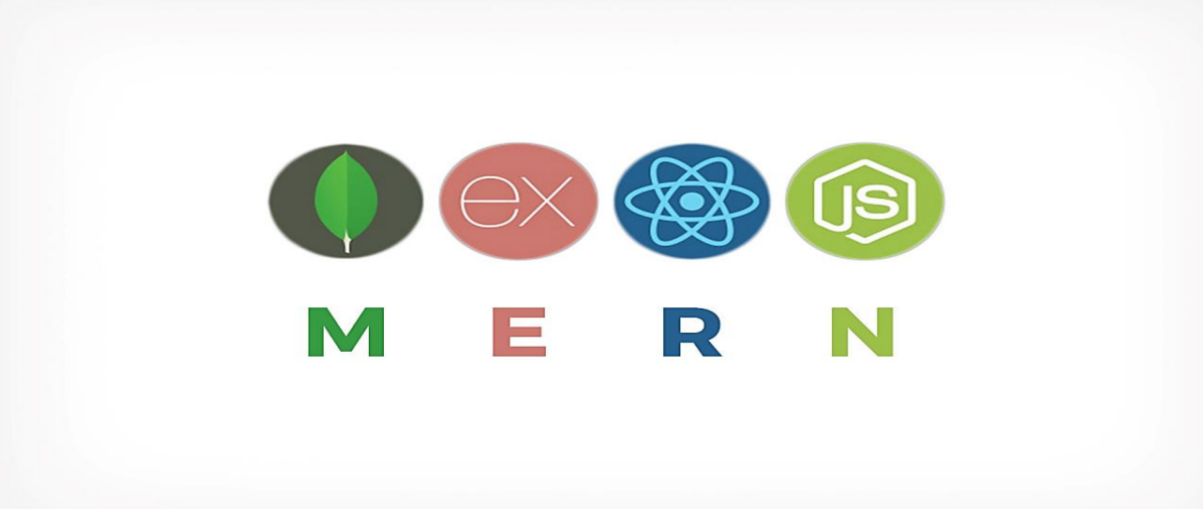
The application is reduced as much as possible to avoid errors while entering the data. It also provides error message while entering invalid data. No formal knowledge is needed for the user to use this system. Thus, by this all it proves it is user-friendly. Online chat-application as

described above can lead to error free, secure, reliable and fast management system.

The purpose of Online Chat Application is to automate the exisiting manual system by the help of computerized equipment and full-fledged computer software fulfilling their requirements, so that their valuable data/information can be stored for longer period with easy accessing and manipulation of the same. The required software and hardware are easily available and easy to work with. Teleconferencing or Chatting is a method of using technology to bring people and ideas "together" despite of the geographical barriers. The technology has be available for years but the acceptance it was quit recent. Our project is an example of a chat server. It is made up of 2 applications the client side and the server side.

MERN Stack:

Today Developers around the world are making efforts to enhance user experience of using application as well as to enhance the developer's workflow of designing applications to deliver projects and rollout change requests under strict timeline. Stacks can be used to build web applications in the shortest span of time. The stacks used in web development are basically the response of software engineers to current demands. They have essentially adopted pre-existing frameworks (including JavaScript) to make their lives easier. While there are many, MEAN and MERN are just two of the popular stacks that have evolved out of JavaScript. Both stacks are made up of open source components and offer an end-to-end framework for building comprehensive web apps that enable browsers to connect with databases. The common theme between the two is JavaScript and this is also the key benefit of using either stack. One can basically avoid any syntax errors or any confusion by just coding in one programming language, JavaScript. Another advantage of building web projects with MERN is the fact that one can benefit from its enhanced flexibility. In order to understand MERN stack, we need to understand the four components that make up the MERN stack(fig.1), namely – MongoDB, Express.js, React and Node.js.



1.1 What is MONGODB?

- MongoDB is a cross-platform document oriented NoSQL database used for high volume data storage that provides high performance, high availability and easy scalability.
- MongoDB stores data in flexible, JSON-like documents, meaning fields can vary from document to document and data structure can be changed over time. The document model maps to the object in the application code, making data easy to work with.
- The data model available within MongoDB allow users to represent hierarchical relationships, to store arrays, and other more complex structues more easily.
- MongoDB works on concept of collection and documents. Each database contains collections which in turn contains documents. Each document can have varying number of fields. The size and content of each document can also be different from each other.

1.1.1 Key components of MongoDB Architecture

1. **_id:** This is a 24-digit unique identifier field required in every MongoDB document in the collection. The _id field is like the document's primary key. If the user creates a new document without an _id field, MongoDB will automatically create the field.
2. **Collection:** Collection is a group of MongoDB documents. It is the equivalent of an RDBMS table. A collection exists within a single database. Collections do not enforce a schema. Typically, all documents in a collection are of similar or related purpose.
3. **Document:** - A document is a set of key-value pairs. Documents have dynamic schema. Dynamic schema means that documents in the same collection do not need to have the same set of fields or structure, and common fields in a collection's

documents may hold different types of data.

4. **Database:** Database is a physical container for collections. Each database gets its own set of files on the file system. A single MongoDB server typically has multiple databases.
5. **Field:** A name-value pair in a document. A document has zero or more fields. Fields are analogous to columns in relational databases.

1.2 What is Express JS?

- Express is a minimal and flexible Node.js web application framework that provides a robust set of features for web and mobile applications. It is an open source framework developed and maintained by the Node.js foundation.
- Express provides us the tools that are required to build our app, be it single-page, multi-page or hybrid web applications. It is flexible as there are numerous modules available on npm (Node Package Manager), which can be directly plugged into Express.
- Unlike its competitors like Rails and Django, which have an opinionated way of building applications, Express has no "best way" to do something. It is very flexible and pluggable.
- Pug (earlier known as Jade) is a terse language for writing HTML templates. It produces HTML, supports dynamic code and code reusability (DRY). It is one of the most popular template languages used with Express.
- Express can be thought of as a layer built on the top of the Node.js that helps manage a server and routes. It allows users to setup middleware to respond to HTTP Requests and defines a routing table which is used to perform different actions based on HTTP method and URL.
- Express allows to dynamically render HTML Pages based on passing arguments to templates.
- Express is asynchronous and single threaded and performs I/O operations quickly

1.2.1 Why use Express?

- Ultra fast I/0
- Asynchronous and single threaded
- MVC like structure
- Robust API makes routing easy.

1.3 What is Node JS?

- Node.js is a very powerful JavaScript-based platform built on Google Chrome's JavaScript V8 Engine. It is used to develop I/O intensive web applications like video

streaming sites, single page applications, and other web applications. Node.js is open source, completely free, and used by thousands of developers around the world.

- Node.js is a server-side platform built on Google Chrome's JavaScript Engine (V8 Engine). Node.js was developed by Ryan Dahl in 2009.
- Node.js applications are written in JavaScript and can be run within the Node.js runtime on OS X, Microsoft Windows, and Linux.
- Node.js also provides a rich library of various JavaScript modules which simplifies the development of web applications using Node.js to a great extent.

1.3.1 Features of Node JS?

1. **Extremely fast:** Node.js is built on Google Chrome's V8 JavaScript Engine, so its library is very fast in code execution.
2. **I/O is Asynchronous and Event Driven:** All APIs of Node.js library are asynchronous i.e. non-blocking. So, a Node.js based server never waits for an API to return data. The server moves to the next API after calling it and a notification mechanism of Events of Node.js helps the server to get a response from the previous API call. It is also a reason that it is very fast.
3. **Single threaded:** Node.js follows a single threaded model with event looping.
4. **Highly Scalable:** Node.js is highly scalable because event mechanism helps the server to respond in a non-blocking way.
5. **No buffering:** Node.js cuts down the overall processing time while uploading audio and video files. Node.js applications never buffer any data. These applications simply output the data in chunks.
6. **Open source:** Node.js has an open source community which has produced many excellent modules to add additional capabilities to Node.js application.

1.4 What is React ?

1.4.1 Virtual-DOM:

Virtual-DOM is a JavaScript object, each object contains all the information needed to create a DOM, when the data changes it will calculate the change between the object and the real tree, which will help optimize re-render DOM tree. It can be assumed that is a virtual model can handle client data.

1.4.2 Component: React is built around components, not templates like other frameworks.

A component can be created by the create Class function of the React object, the starting point when accessing this library**.** ReactJS creates HTML tags unlike we normally write but uses Component to wrap HTML tags into stratified objects to render. Among React Components, render function is the most important. It is a function that handles the generation of HTML tags as well as a demonstration of the ability to process via Virtual- DOM. Any changes of data at any time will be processed and updated immediately by Virtual- DOM.

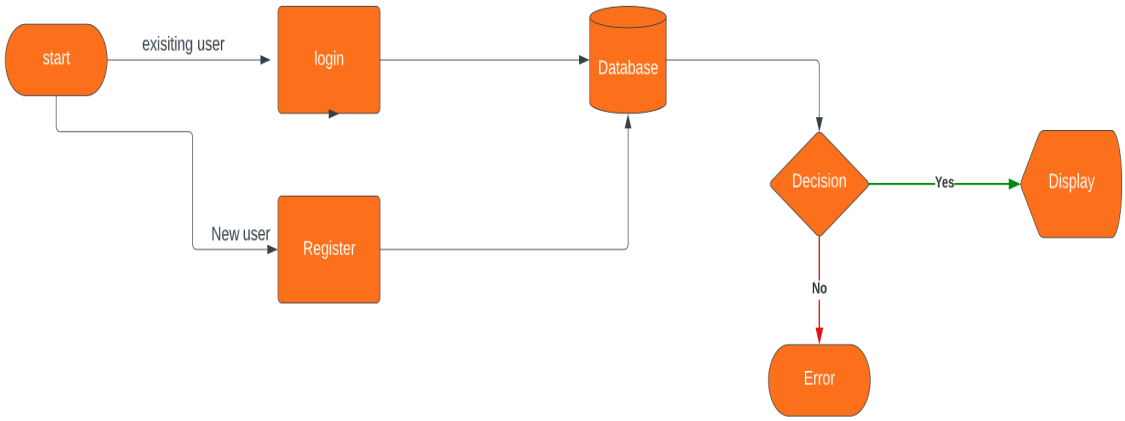
1.4.3 Props and State

Props: are not controlled by Component, actually stands for Properties

The title = “Title” line creates a name attribute with the value "Title". It looks like a function call. It is true that props are passed to the component in the same way that an argument is passed to a function. A component may also have a default props, so if the component does not pass any props, it will still be set.

State: private data is controlled by Component. Like props, sate also holds information about the component. However, the type of information and how to handle it varies. State works differently than Props. The state is a component of the component, while props are passed in from the outside into the component. It should be noted that we should not update the state directly using this. State but always use setState to update. Update the state of the objects. Use setState to renders one component and all its children. This is great, because you don't have to worry about writing event handlers like any other language.

Process of Chat Application



Future Scope:

- Video call/ Audio Call
- Emoji/ GIF
- Groups
- Adding Profile pic

CHAPTER 2

TECHNOLOGIES USED:

- REACT JS (Front END)
- MongoDB (Database NoSql)
- Node JS (Client Side)

Dependencies in React JS:

- Axios (to communicate with backend)
- React-icons (for Icons)
- React-Router-Dom (Routing)
- React-toastify (Alerts)

Dependencies in Node JS:

- Express
- Mongoose
- Nodemon
- Socket.io
- Bcrypt
- Cors
- Dotenv

CHAPTER 3

MODULES:

- **Login/Sign in Module**: This Modules is used to Login/Sign In to the chat application.
- **Signup/Register:** This Modules is used to Signup/Register to create a new account to login to chat application.
- **Home:** This Module is used to access home page where user can do chat or communicate with their friends / colleagues.

CHAPTER 4

4.1 SNAPSHOTS:

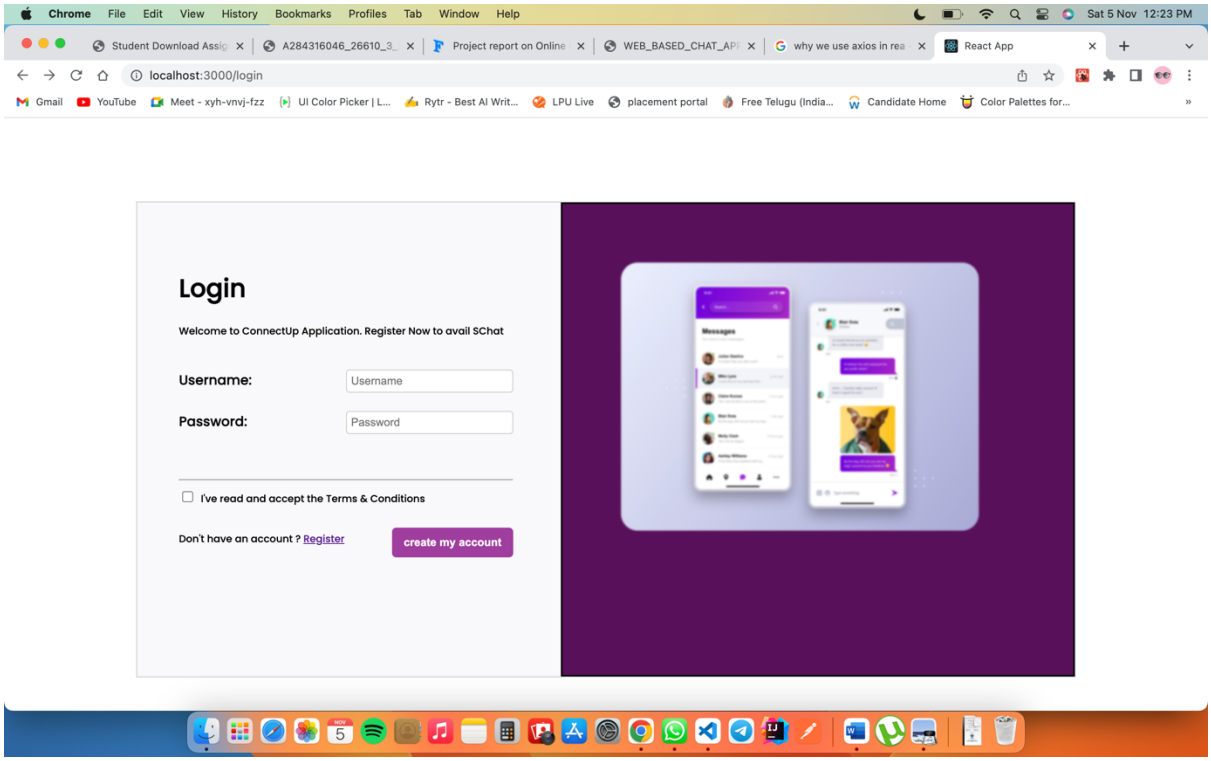


Figure 4.1.1 Login Page

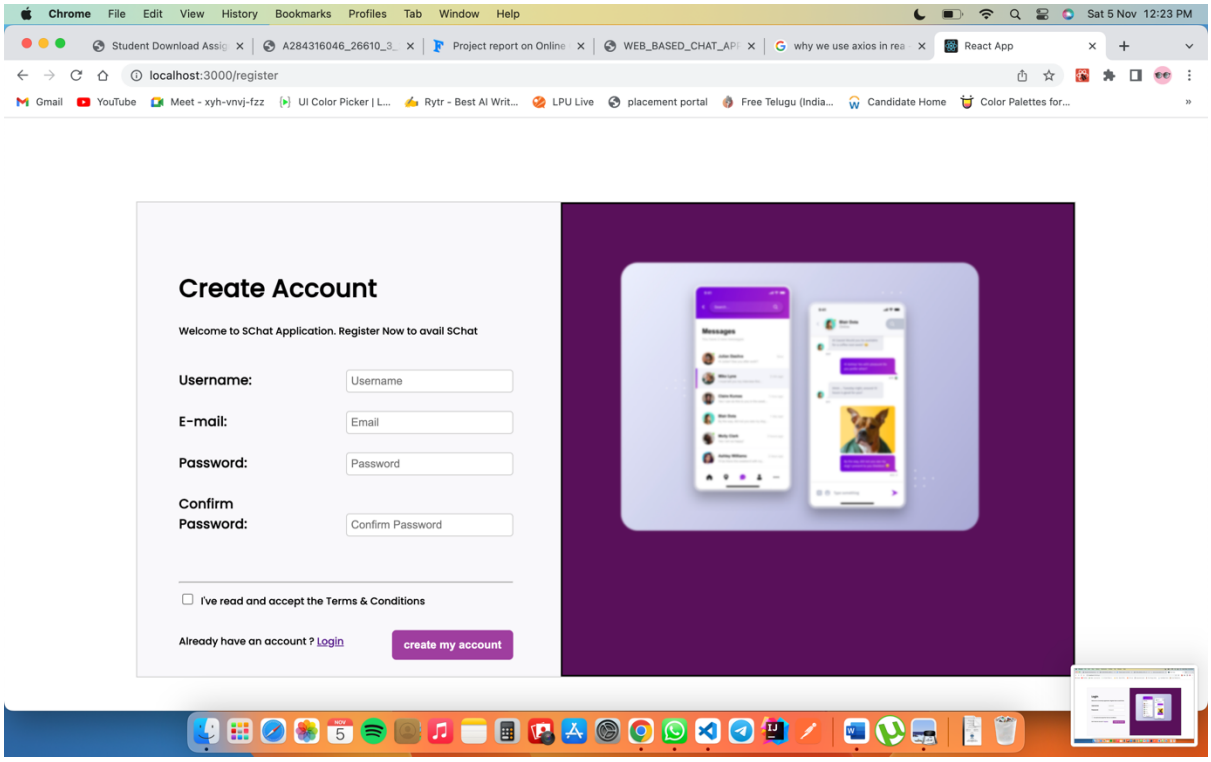


Figure 4.1.2 Register Page

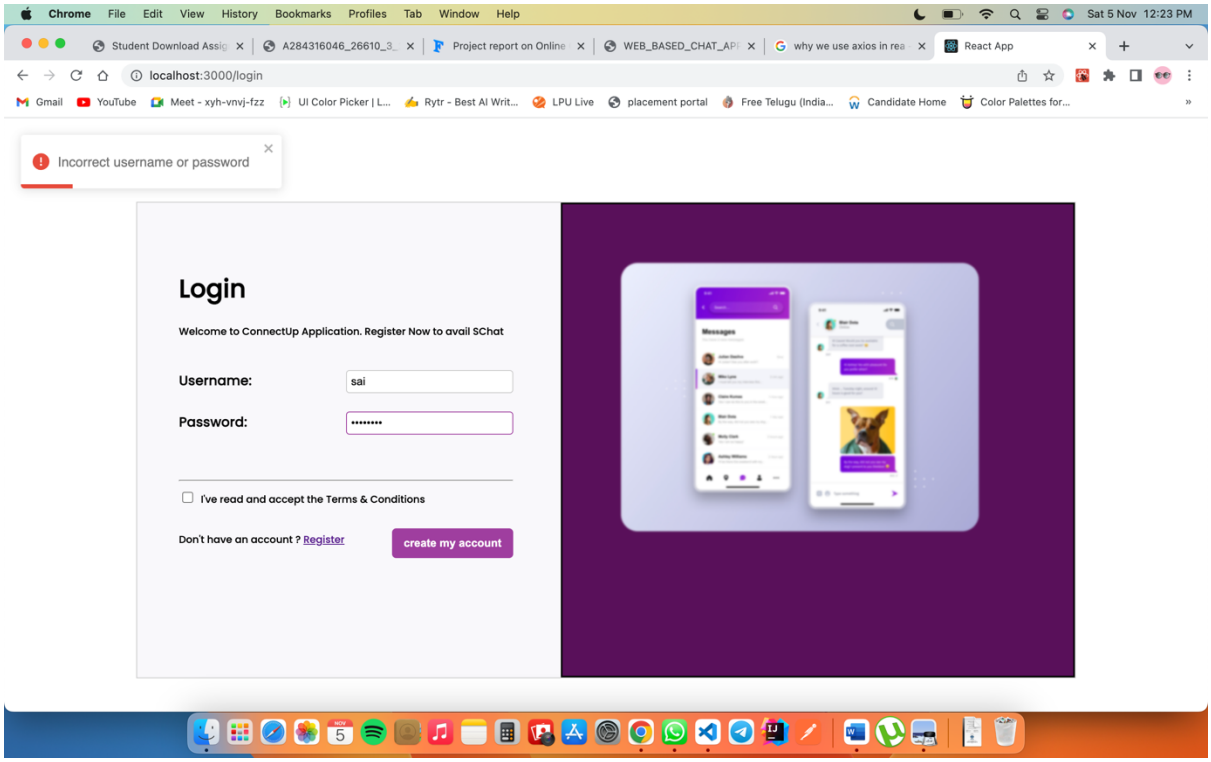


Figure 4.1.3 Login Alerts

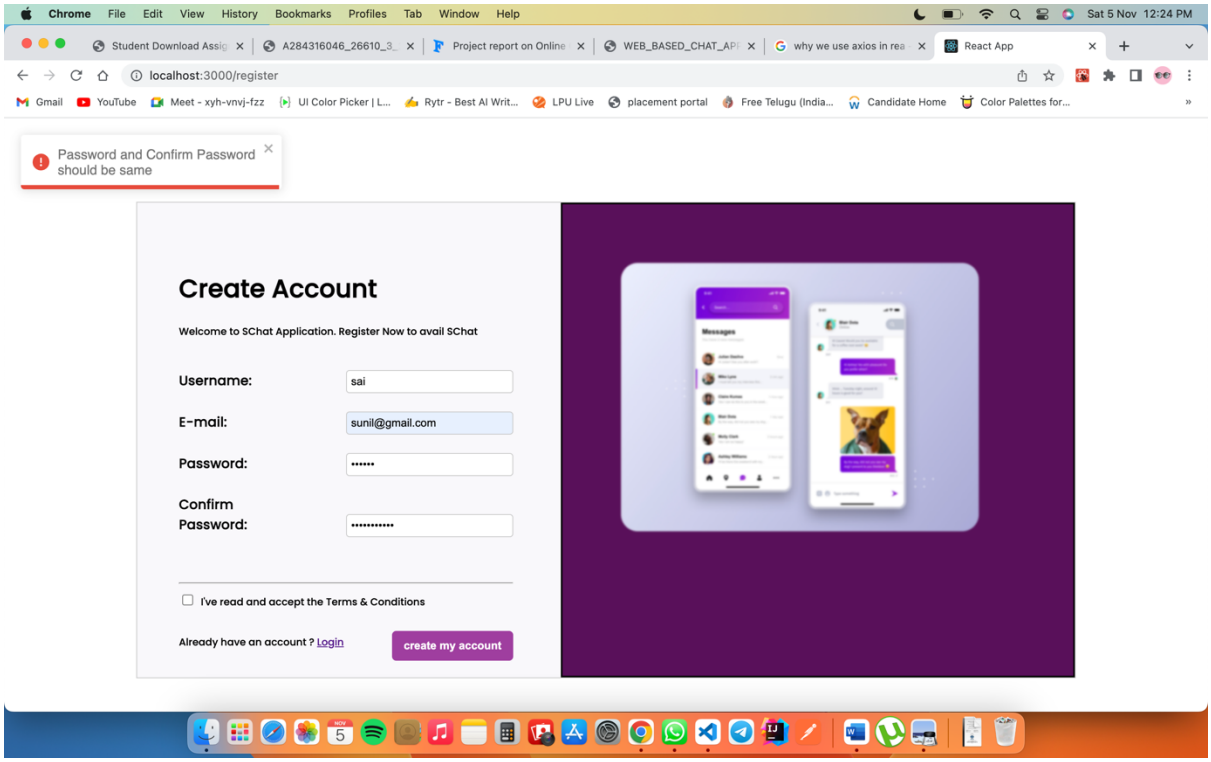


Figure 4.1.4 Register Alerts

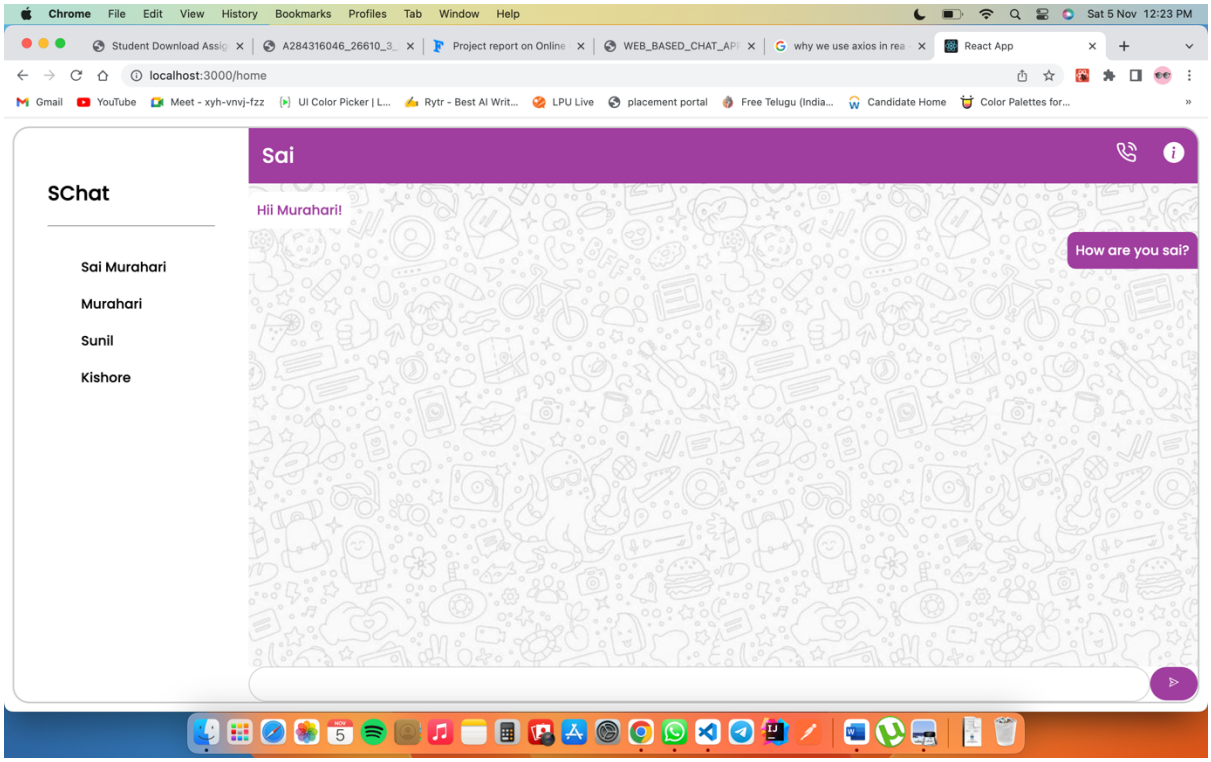


Figure 4.1.5 Home page (CHAT-APP

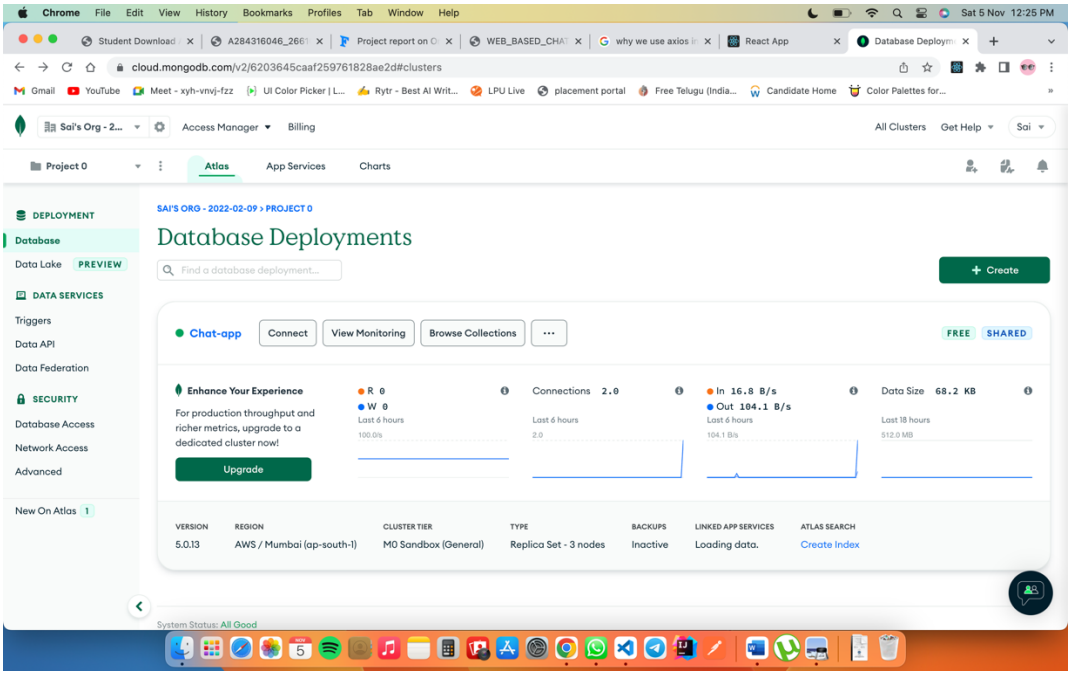


Figure 4.1.6 MongoDB Cluster

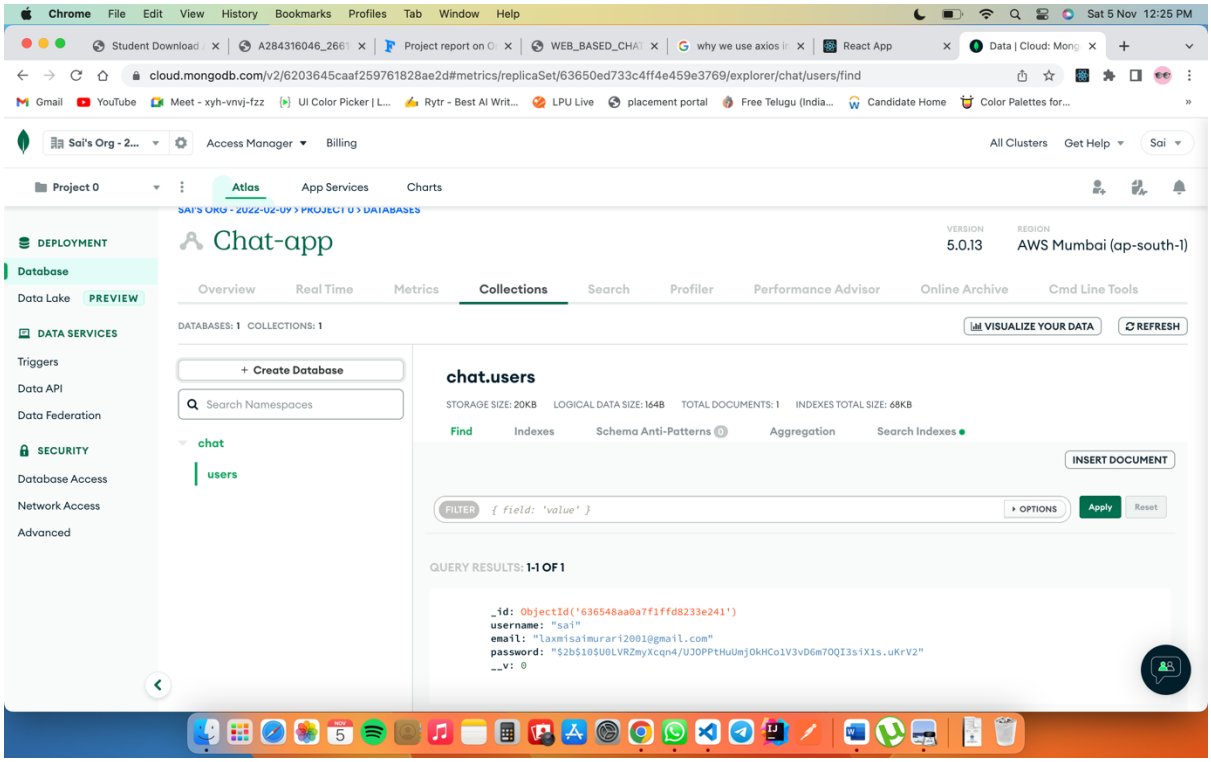


Figure 4.1.7 MongoDB Database Collections and Data

4.2 Folder Structure

1. Client Side:

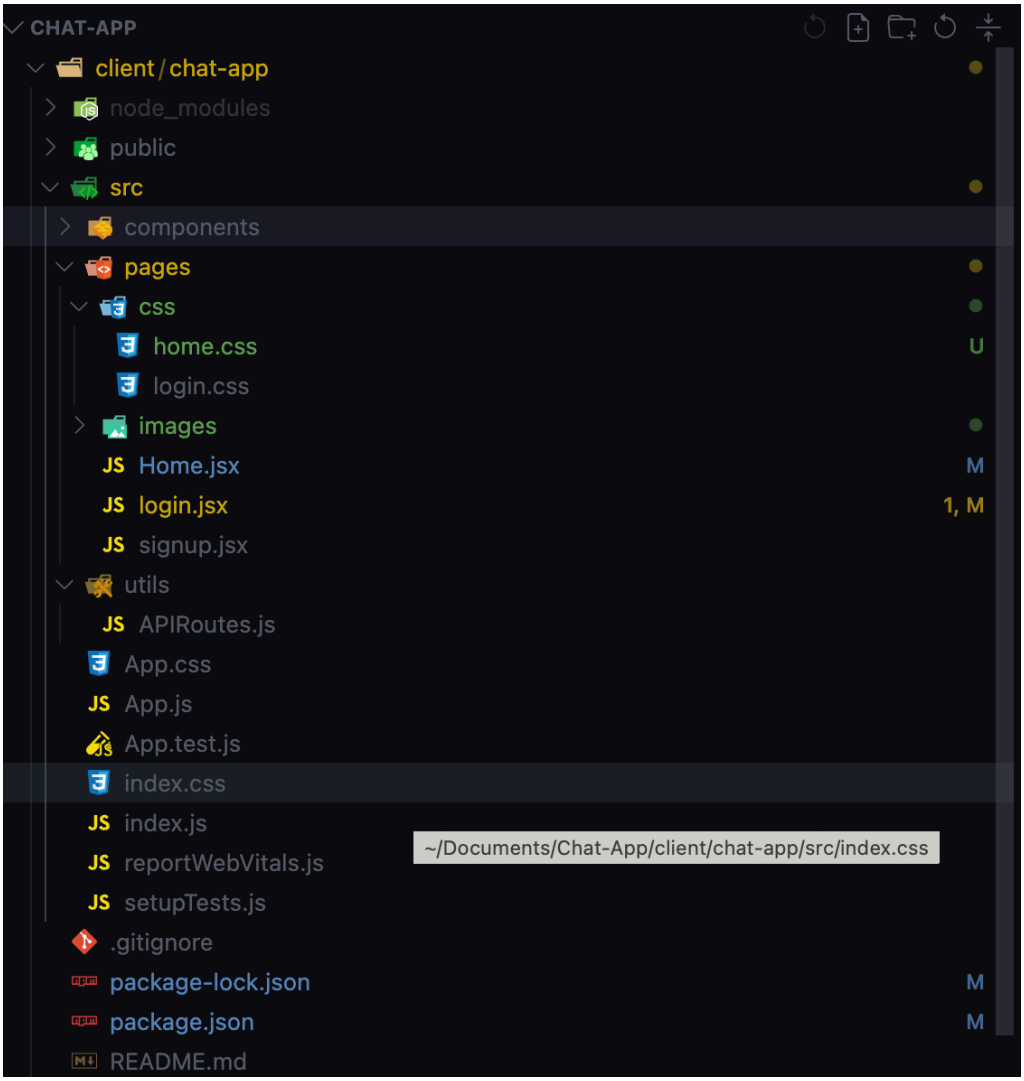


Figure 4.2.1 React Folder structure

All the main Folders are present at the src folder

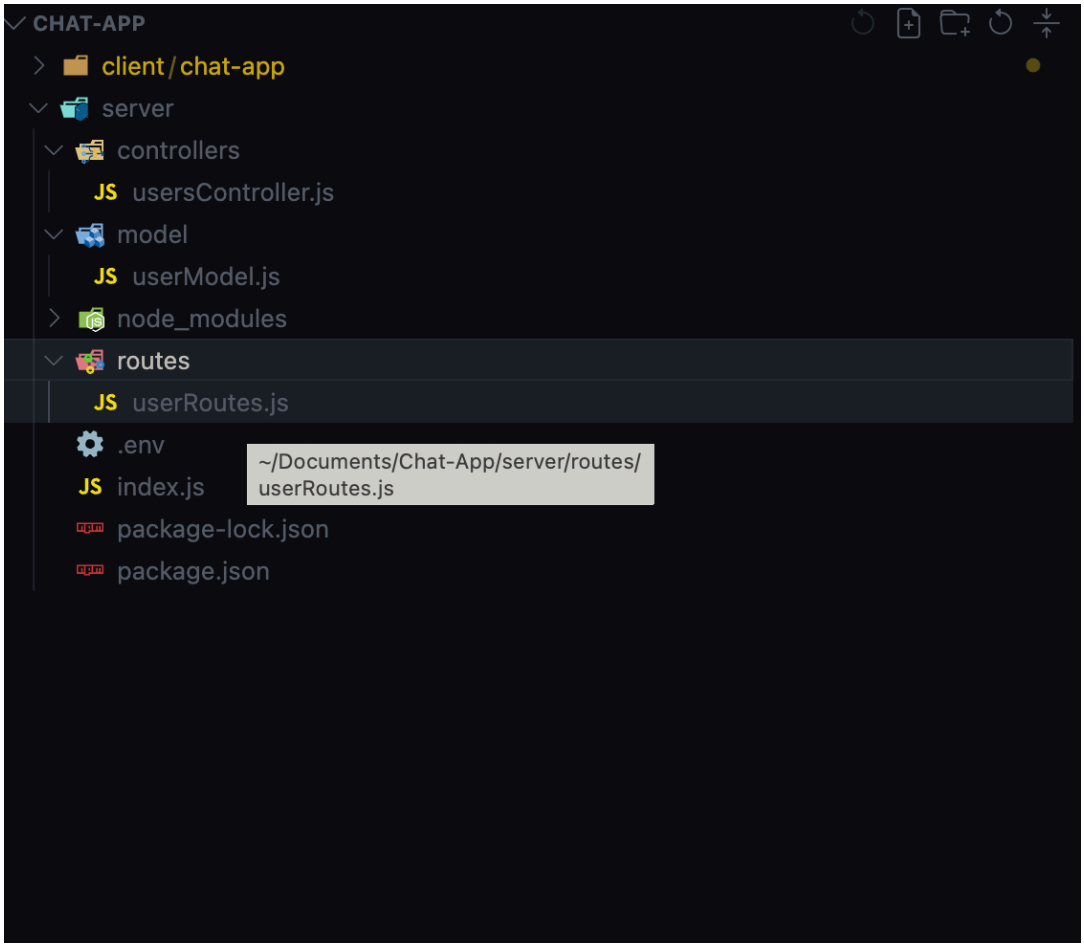
Pages: folder is for the pages like login, register & Home Page.

CSS: All the Styling Pages present at css folder for login/Register and Home page

Utils: API routes file is to communicate with the backend/ server of Node js

Front End server is running on **PORT 3000**

2. Server Side:



Controller: For making validations like if user already present or if any user enters wrong credentials.

Models: For creating schema or collections in MongoDB.

Routes: Routing the components like login register

Backend server is running on **PORT : 4000**

CHAPTER 5

Github - Link : https://github.com/saimurahari/Chat-App

CHAPTER 6

REFERENCES

1. **MongoDB:** https://docs.mongodb.com/ecosystem/drivers/ , https://www.guru99.com/what-is-mongodb.html.
2. **ExpressJS:** https://expressjs.com/en/guide/routing.html, https://www.javatpoint.com/expressjs-tutorial.
3. **Npm:** https://www.npmjs.com/
4. **ReactJS:** https://reactjs.org/docs/getting-started.html, https://www.javatpoint.com/reactjs-tutorial, https://www.tutorialspoint.com/reactjs/reactjs_overview.htm.
5. **NodeJS**: https://nodejs.org/en/docs/, https://www.javatpoint.com/nodejs-tutorial.
6. **Socket.io** : https://socket.io/