BRAC UNIVERSITY

Inspiring Excellence

CSE470: Software Engineering

Section: 5

Group: 3

SPRINT 3

Sadat Mahmud	22301301
Rana Mustafa	21101060
Atoshi Samadder	21301706 *(Scrum Master)*
Nabil Nashit	21201060

Software: VisiNexus – Connecting insights through vision

Requirements

1. Real-time demos for computer vision applications such as object detection, emotion recognition, etc.
2. Interactive tools for learning and understanding computer vision applications
3. Documentation and tutorials
4. API access and integration guide
5. User Management
6. Virtual coins for purchasing accessibility

Framework Setup

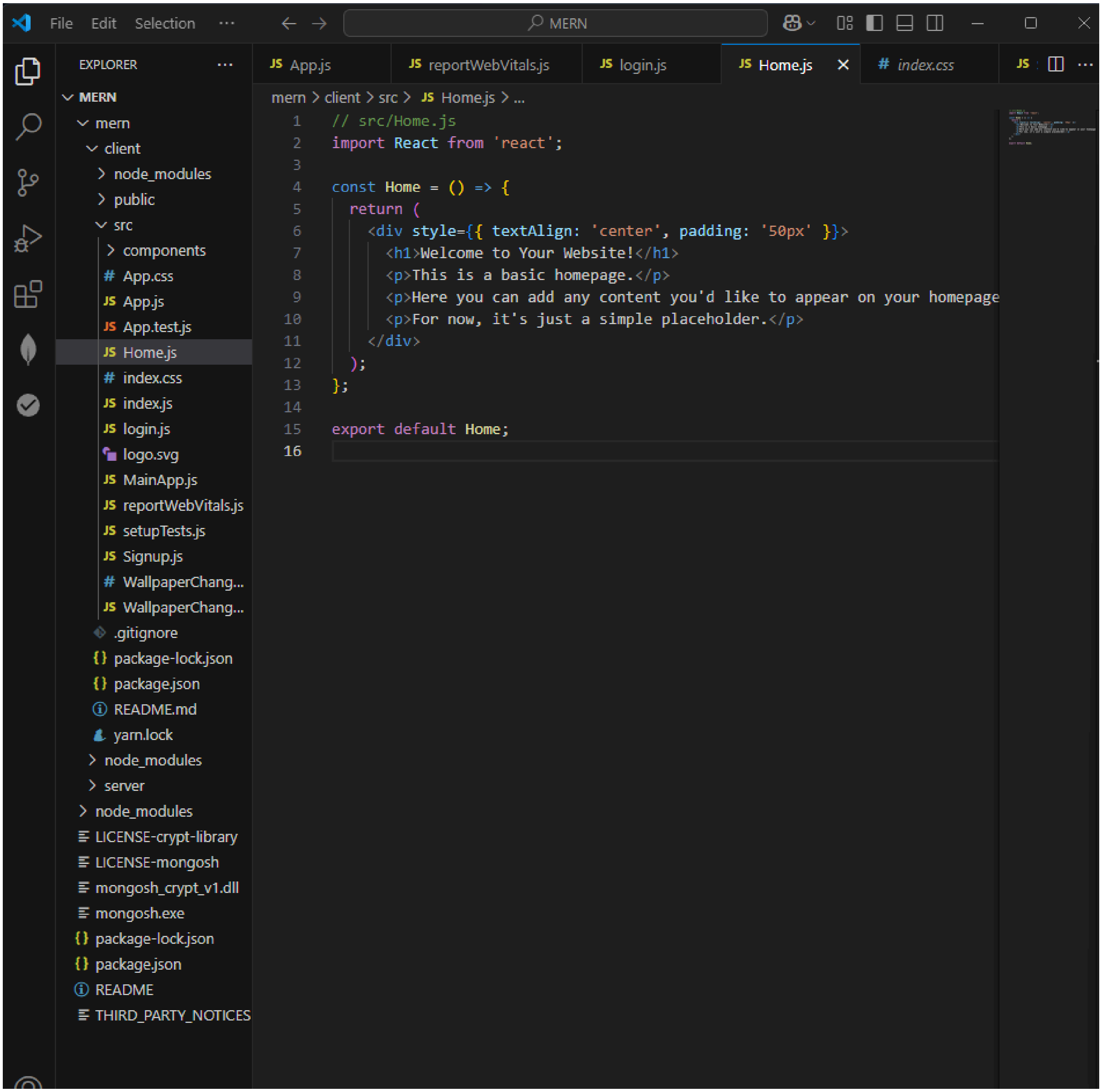
Frontend	React.js
Backend	Express.js
Computer Vision Models	OpenCV and TensorFlow
Database	MongoDB
Hosting	GoDaddy

Workload Distribution

Sadat Mahmud	Project Manager: Oversee environmental setup, and proper deployment
Rana Mustafa	Computer Vision Specialist: Optimize Computer Vision Models
Atoshi Samadder	Frontend Developer: Develop the user interface
Nabil Nashit	Backend Developer: Build the server-side logic and API endpoints

1. Initial Setup of the MERN Stack

- MongoDB: The database for storing your app's data, using the free cluster0 for our database.
- Express.js: The backend framework running on Node.js to handle HTTP requests and responses.
- React.js: The frontend JavaScript library for building the user interface we will use for our projects.
- Node.js: A JavaScript runtime environment used for running the server-side code.



```
// src/Home.js
import React from 'react';

const Home = () => {
  return (
    <div style={{ textAlign: 'center', padding: '50px' }}>
      <h1>Welcome to Your Website!</h1>
      <p>This is a basic homepage.</p>
      <p>Here you can add any content you'd like to appear on your homepage
      <p>For now, it's just a simple placeholder.</p>
    </div>
  );
};

export default Home;

```

2. Installing React :

Using the terminal of VS Code

1.First creating the npm directory manually or by this terminal cmd :

```
npx create-react-app client
```

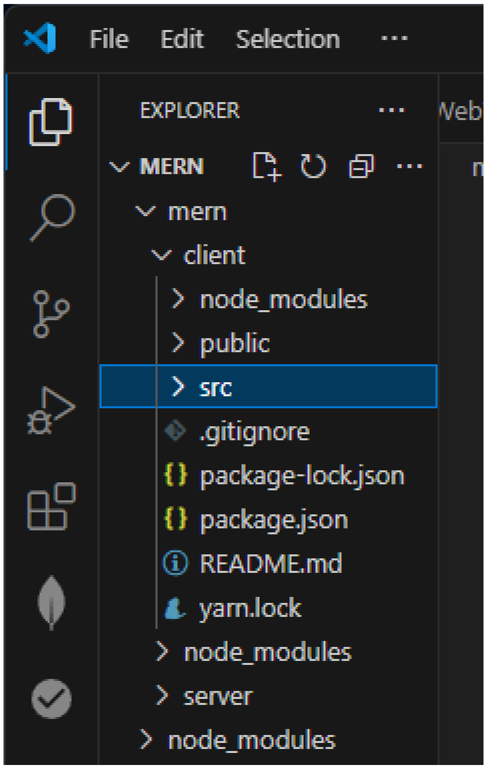
2. Initialize the project and installing dependency:

```
npm init -y

npm install react react-dom react-scripts cra-template
```

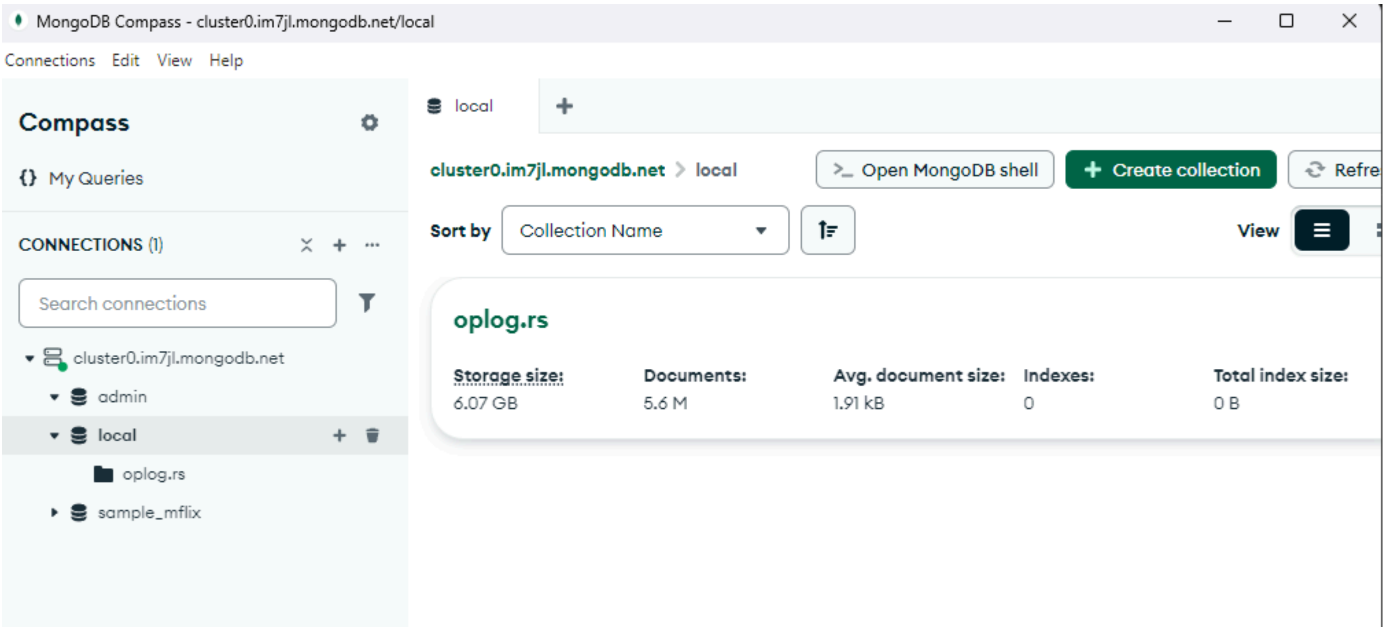
3. Updating Package.json to handle API request running on port 5050

```
"proxy": "http://localhost:3000"
```

3. Project Structure : [Frontend]



4. MongoDB Setup and Connection: No updates



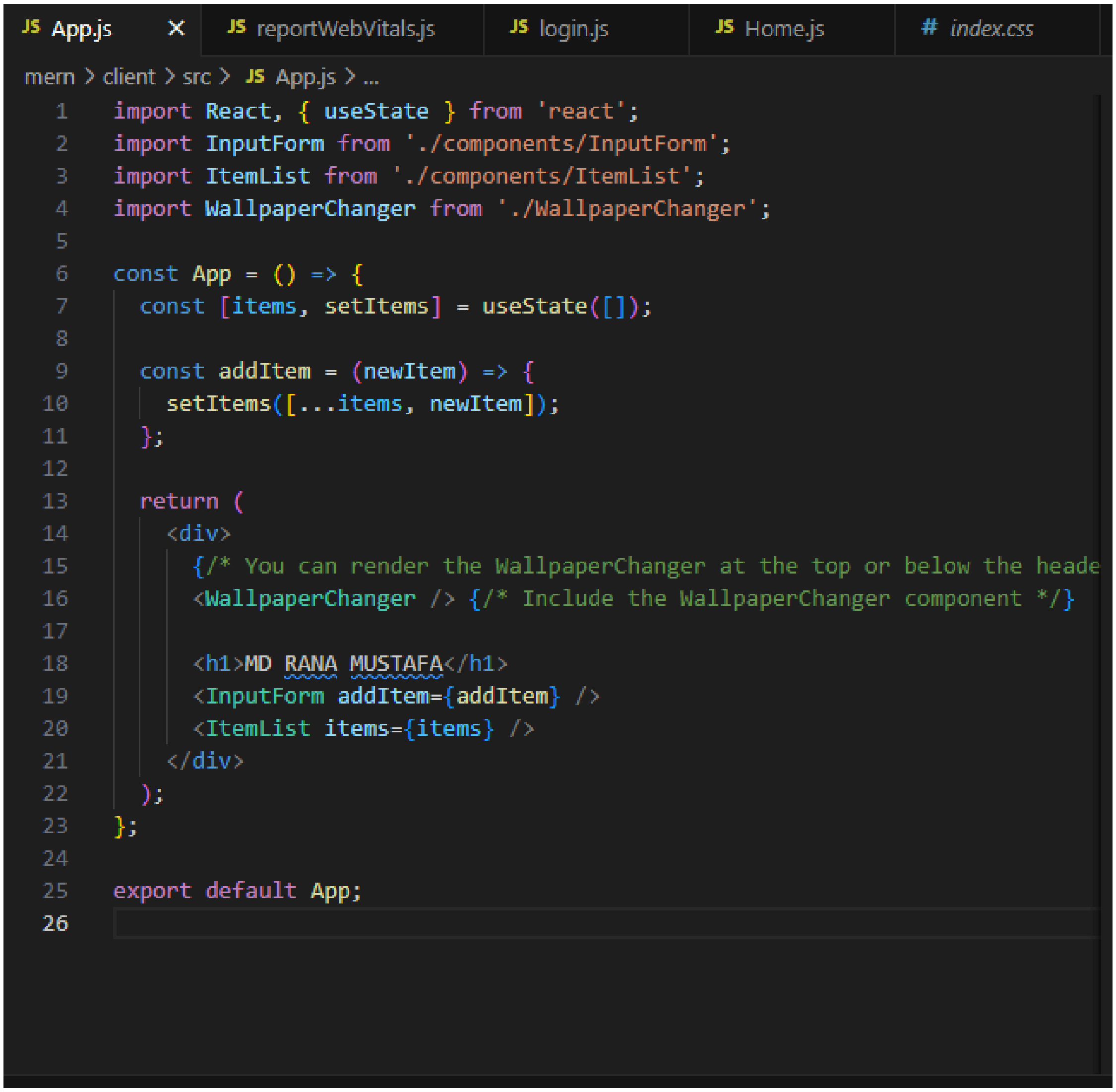
oplog.rs

Storage size:	Documents:	Avg. document size:	Indexes:	Total index size:
6.07 GB	5.6 M	1.91 kB	0	0 B

5. Frontend layout and styling set up : [Frontend]

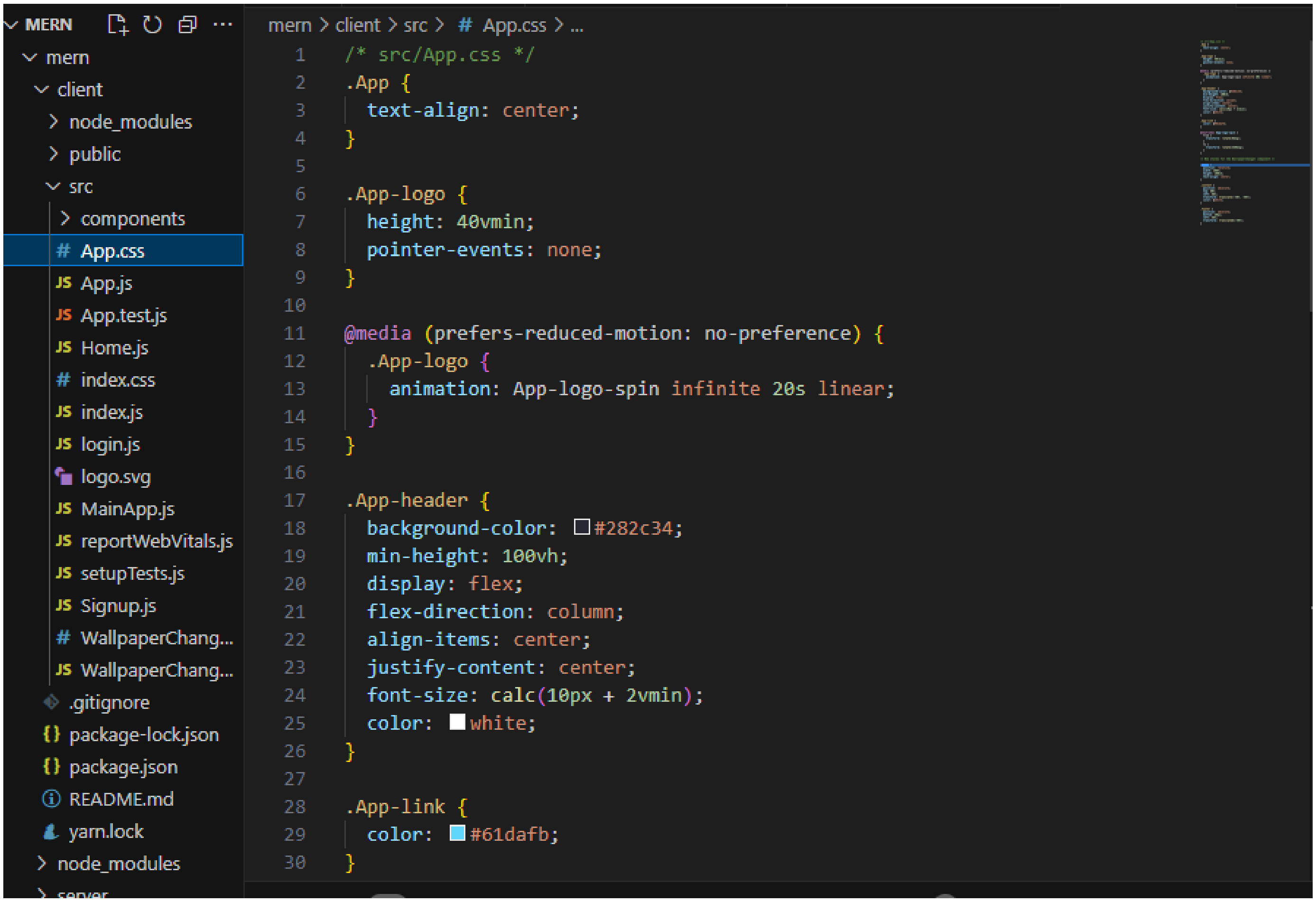
□ Core Files

- App.js
  - □ The main component that acts as the entry point for my application logic.
  - □ Typically, contains the app-wide layout, routing, and general structure.



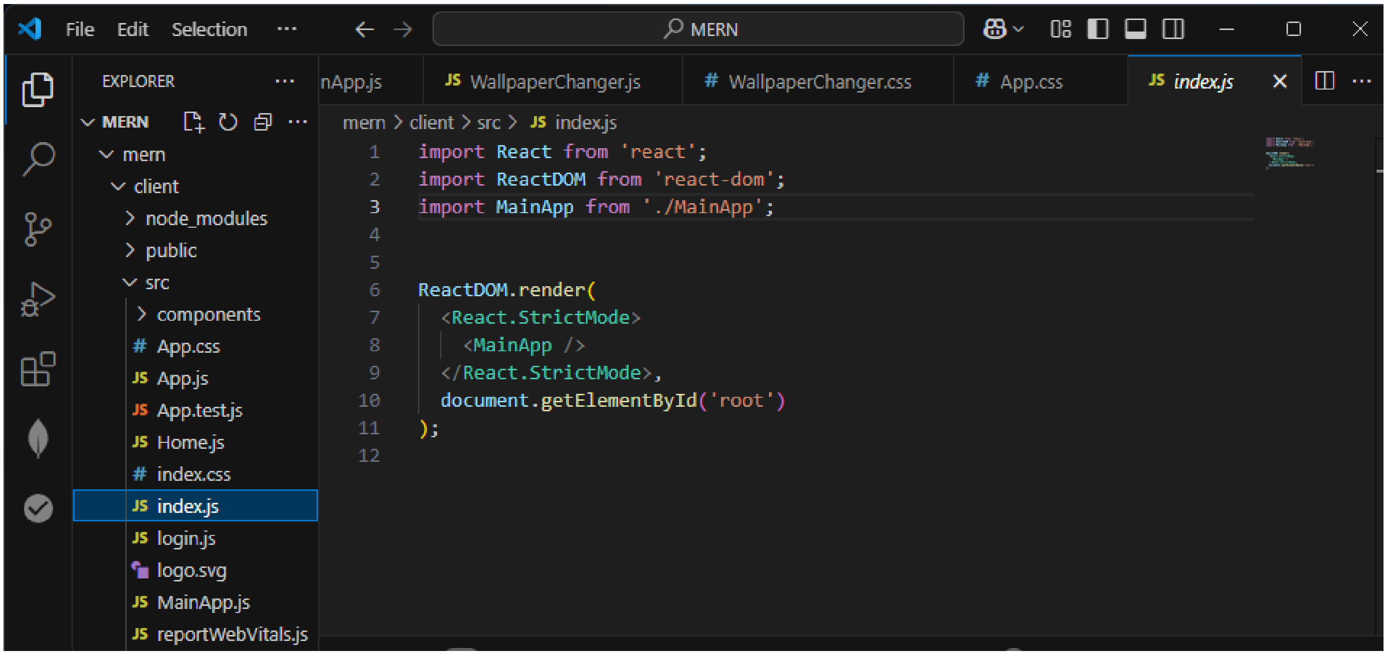
```
import React, { useState } from 'react';
import InputForm from './components/InputForm';
import ItemList from './components/ItemList';
import WallpaperChanger from './WallpaperChanger';

const App = () => {
  const [items, setItems] = useState([]);

  const addItem = (newItem) => {
    setItems([...items, newItem]);
  };

  return (
    <div>
      {/* You can render the WallpaperChanger at the top or below the heade
      <WallpaperChanger /> {/* Include the WallpaperChanger component */}

      <h1>MD RANA MUSTAFA</h1>
      <InputForm addItem={addItem} />
      <ItemList items={items} />
    </div>
  );
};

export default App;

```

- App.css
  - □ The primary CSS file for styling your App.js component.
  - □ Contains styles specific to global app design.



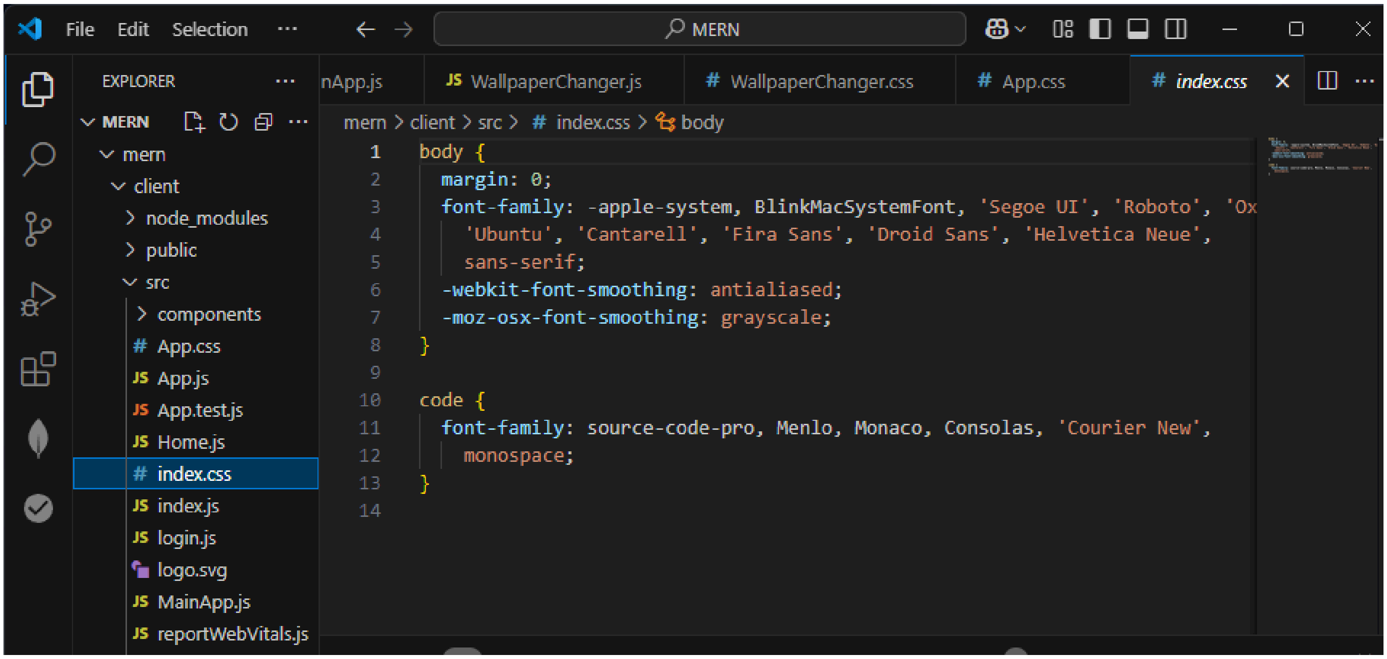
```css
/* src/App.css */
.App {
  text-align: center;
}

.App-logo {
  height: 40vmin;
  pointer-events: none;
}

@media (prefers-reduced-motion: no-preference) {
  .App-logo {
    animation: App-logo-spin infinite 20s linear;
  }
}

.App-header {
  background-color: #282c34;
  min-height: 100vh;
  display: flex;
  flex-direction: column;
  align-items: center;
  justify-content: center;
  font-size: calc(10px + 2vmin);
  color: white;
}

.App-link {
  color: #61dafb;
}
```

- Index.js
  - □ The entry point of your React application.
  - □ Responsible for rendering the root React component (App) to the DOM.



```javascript
import React from 'react';
import ReactDOM from 'react-dom';
import MainApp from './MainApp';


ReactDOM.render(
  <React.StrictMode>
    <MainApp />
  </React.StrictMode>,
  document.getElementById('root')
);
```

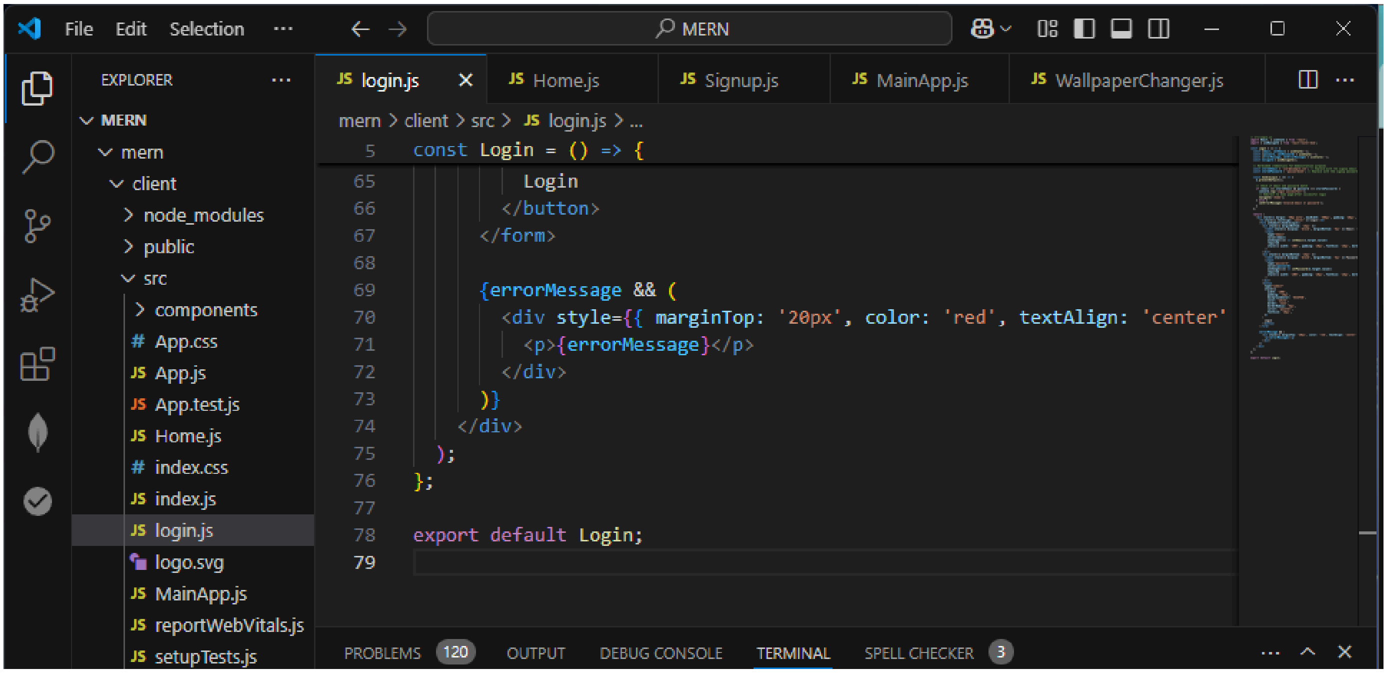
- ndex.css
  - □ Global CSS file applied across the app.



```css
body {
  margin: 0;
  font-family: -apple-system, BlinkMacSystemFont, 'Segoe UI', 'Roboto', 'Ox
    'Ubuntu', 'Cantarell', 'Fira Sans', 'Droid Sans', 'Helvetica Neue',
    sans-serif;
  -webkit-font-smoothing: antialiased;
  -moz-osx-font-smoothing: grayscale;
}

code {
  font-family: source-code-pro, Menlo, Monaco, Consolas, 'Courier New',
    monospace;
}
```

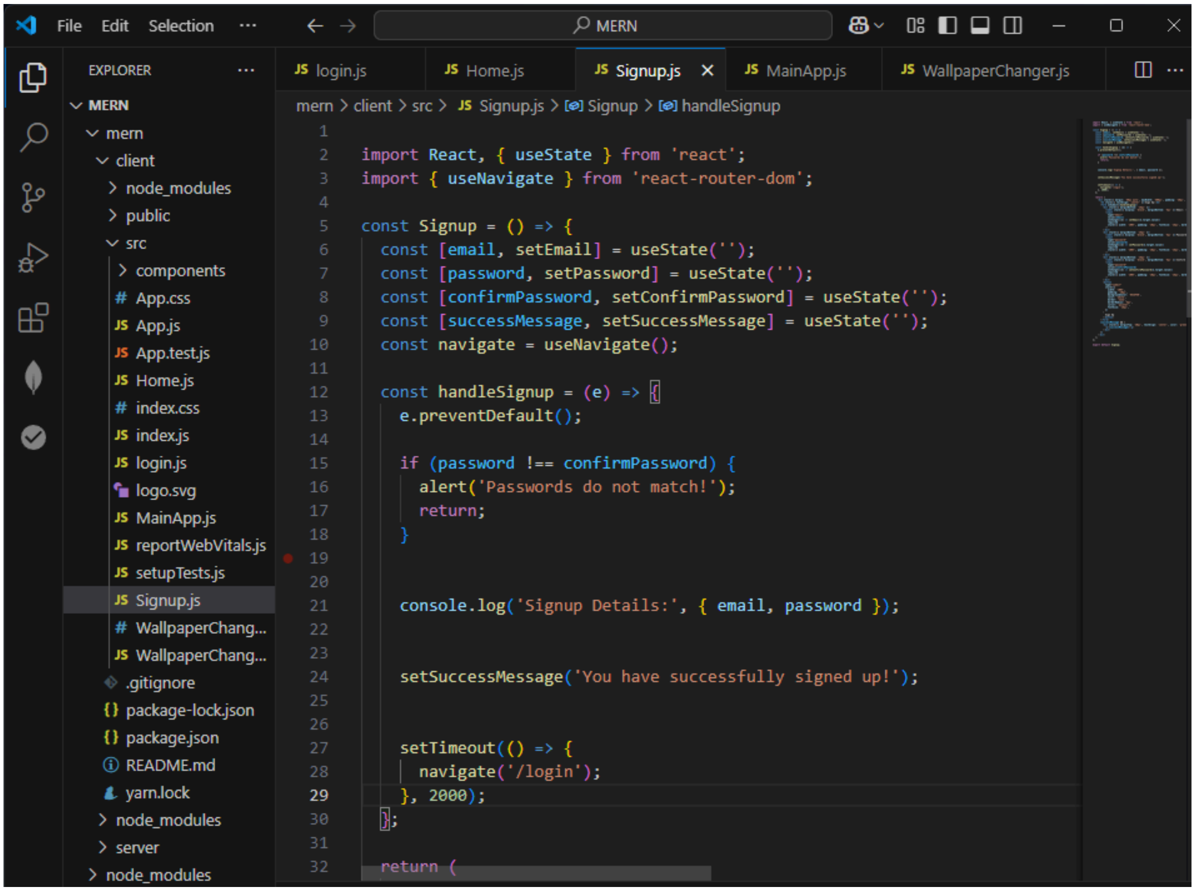
6.Authentication Files:

- Login.js
    - □ Component for the login form.
    - □ Contains fields for user email and password along with the login logic.



```
const Login = () => {
     Login
    </button>
   </form>

   {errorMessage && (
    <div style={{ marginTop: '20px', color: 'red', textAlign: 'center'
     <p>{errorMessage}</p>
    </div>
   )}
  </div>
 );
};

export default Login;
```

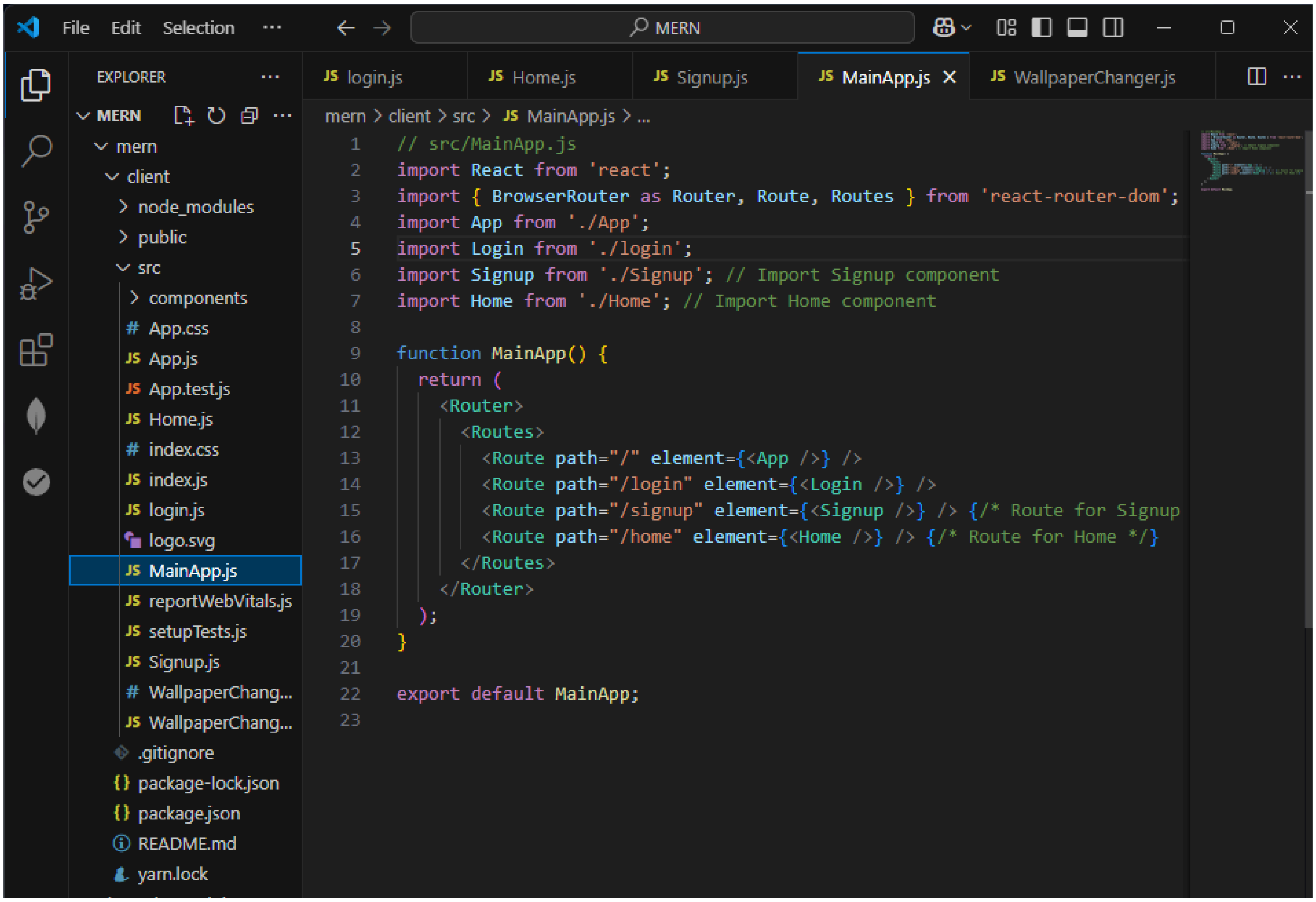
- Signup.js
    - ☐ Component for the signup form.
    - ☐ Allows users to create an account, with logic to handle account creation.



```javascript

import React, { useState } from 'react';
import { useNavigate } from 'react-router-dom';

const Signup = () => {
  const [email, setEmail] = useState('');
  const [password, setPassword] = useState('');
  const [confirmPassword, setConfirmPassword] = useState('');
  const [successMessage, setSuccessMessage] = useState('');
  const navigate = useNavigate();

  const handleSignup = (e) => {
    e.preventDefault();

    if (password !== confirmPassword) {
      alert('Passwords do not match!');
      return;
    }


    console.log('Signup Details:', { email, password });


    setSuccessMessage('You have successfully signed up!');


    setTimeout(() => {
      navigate('/login');
    }, 2000);
  };

  return (
```

7.Routing and Layout Files:

1. MainApp.js
   - ○ Handles routing between different pages (Login, Signup, App, etc.).
   - ○ Wraps your app with BrowserRouter for navigation.

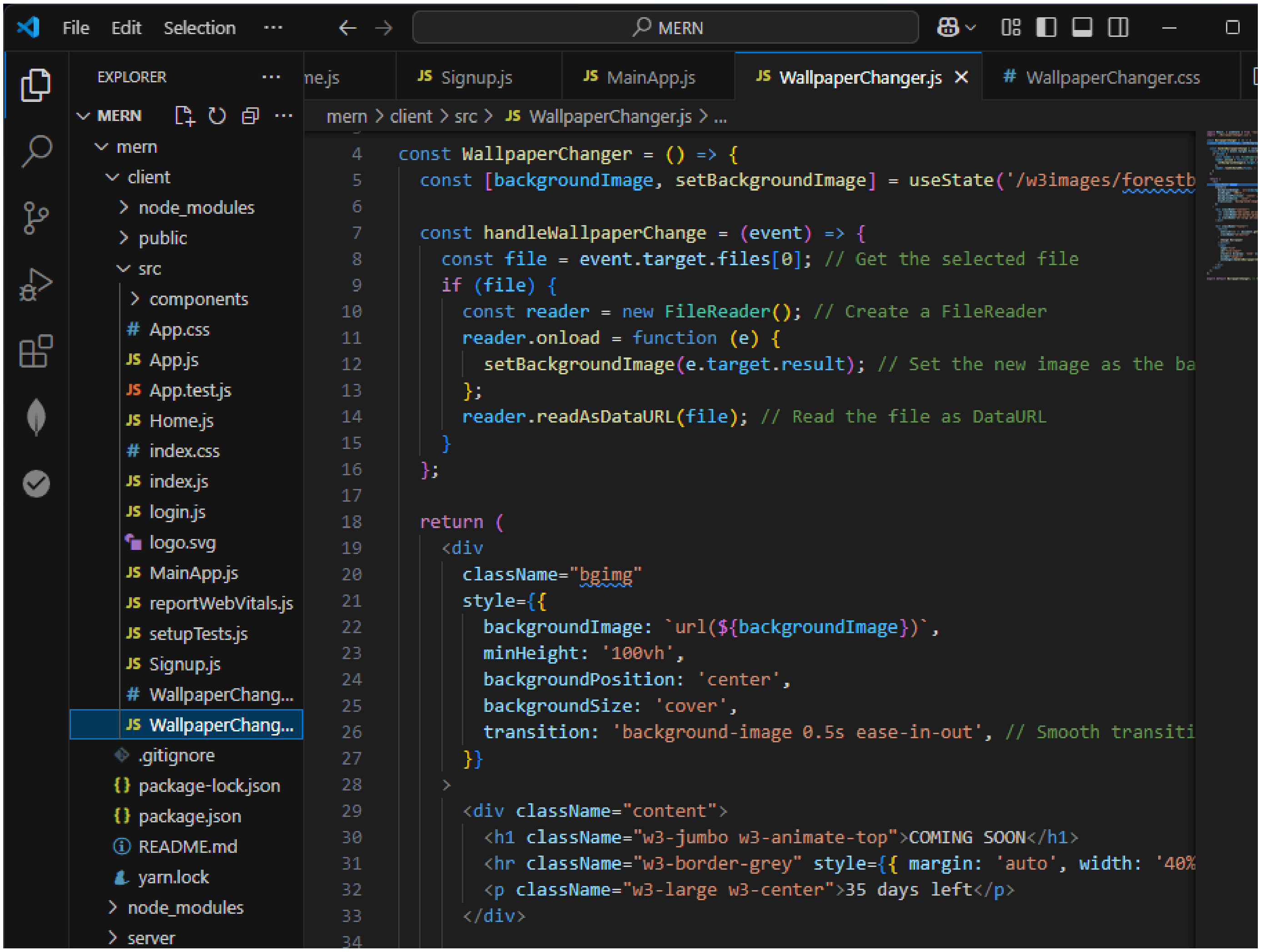


```
// src/MainApp.js
import React from 'react';
import { BrowserRouter as Router, Route, Routes } from 'react-router-dom';
import App from './App';
import Login from './login';
import Signup from './Signup'; // Import Signup component
import Home from './Home'; // Import Home component

function MainApp() {
  return (
    <Router>
      <Routes>
        <Route path="/" element={<App />} />
        <Route path="/login" element={<Login />} />
        <Route path="/signup" element={<Signup />} /> {/* Route for Signup
        <Route path="/home" element={<Home />} /> {/* Route for Home */}
      </Routes>
    </Router>
  );
}

export default MainApp;
```

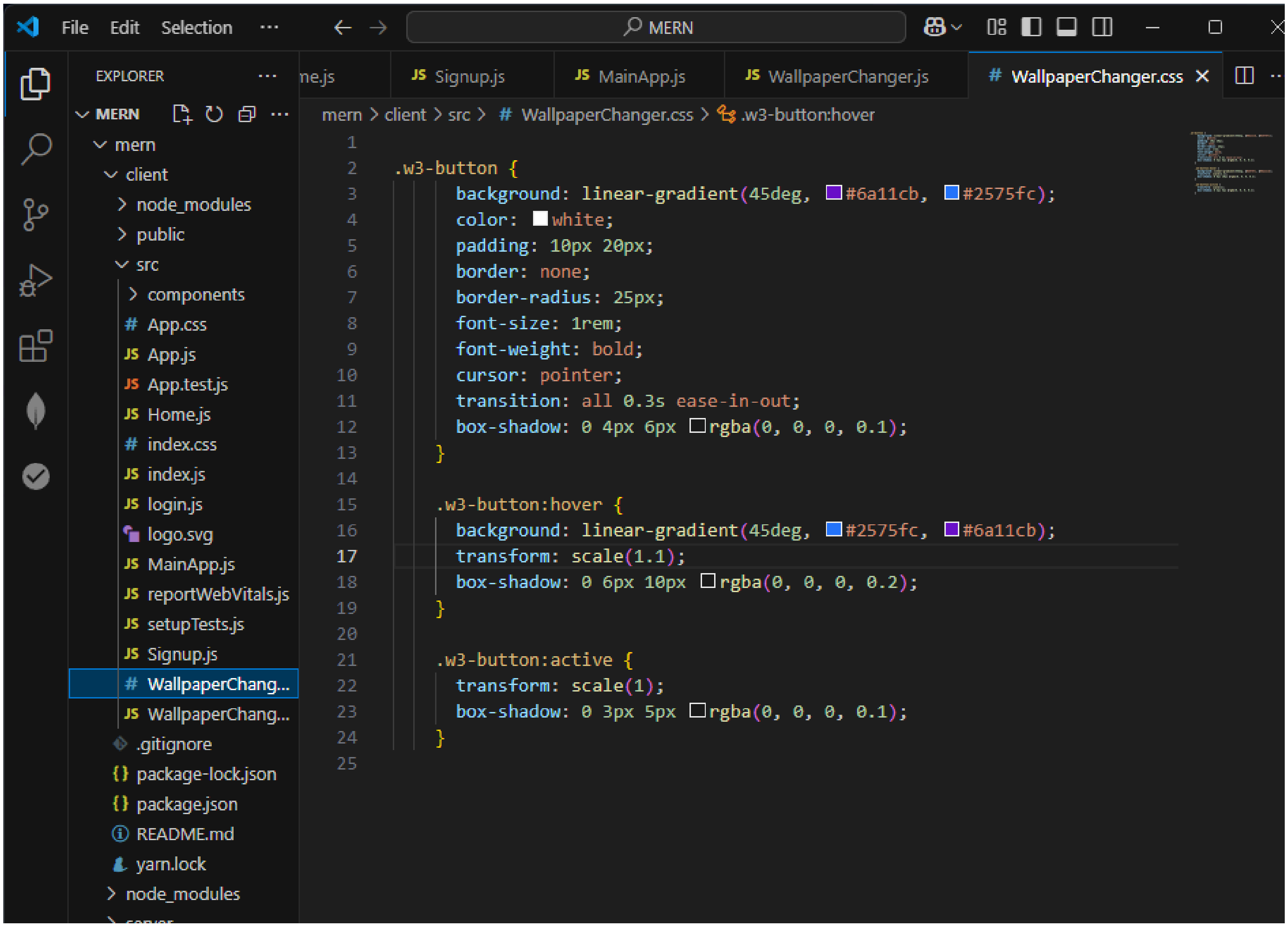
8. **Wallpaper Feature Files**

1. WallpaperChanger.js
   - used to update my wallpaper according to my choice
2. WallpaperChanger.css
   - For styling the WallpaperChanger component.

From W3school I have imported this background code to try in the frontend



```
const WallpaperChanger = () => {
  const [backgroundImage, setBackgroundImage] = useState('/w3images/forestb

  const handleWallpaperChange = (event) => {
    const file = event.target.files[0]; // Get the selected file
    if (file) {
      const reader = new FileReader(); // Create a FileReader
      reader.onload = function (e) {
        setBackgroundImage(e.target.result); // Set the new image as the ba
      };
      reader.readAsDataURL(file); // Read the file as DataURL
    }
  };

  return (
    <div
      className="bgimg"
      style={{
        backgroundImage: `url(${backgroundImage})`,
        minHeight: '100vh',
        backgroundPosition: 'center',
        backgroundSize: 'cover',
        transition: 'background-image 0.5s ease-in-out', // Smooth transiti
      }}
    >
      <div className="content">
        <h1 className="w3-jumbo w3-animate-top">COMING SOON</h1>
        <hr className="w3-border-grey" style={{ margin: 'auto', width: '40%
        <p className="w3-large w3-center">35 days left</p>
      </div>
```

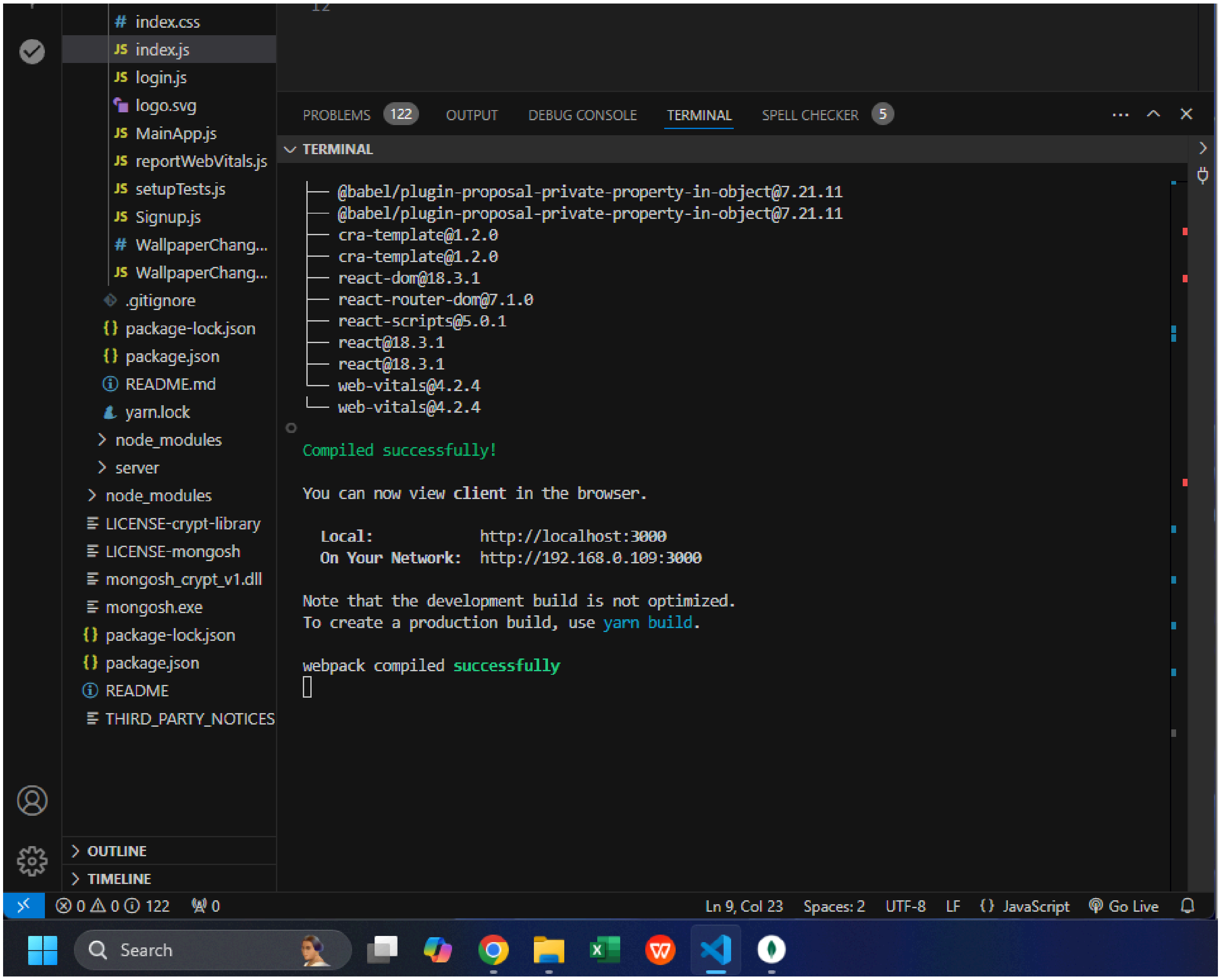


```css

.w3-button {
      background: linear-gradient(45deg, #6a11cb, #2575fc);
      color: white;
      padding: 10px 20px;
      border: none;
      border-radius: 25px;
      font-size: 1rem;
      font-weight: bold;
      cursor: pointer;
      transition: all 0.3s ease-in-out;
      box-shadow: 0 4px 6px rgba(0, 0, 0, 0.1);
    }

    .w3-button:hover {
      background: linear-gradient(45deg, #2575fc, #6a11cb);
      transform: scale(1.1);
      box-shadow: 0 6px 10px rgba(0, 0, 0, 0.2);
    }

    .w3-button:active {
      transform: scale(1);
      box-shadow: 0 3px 5px rgba(0, 0, 0, 0.1);
    }

```

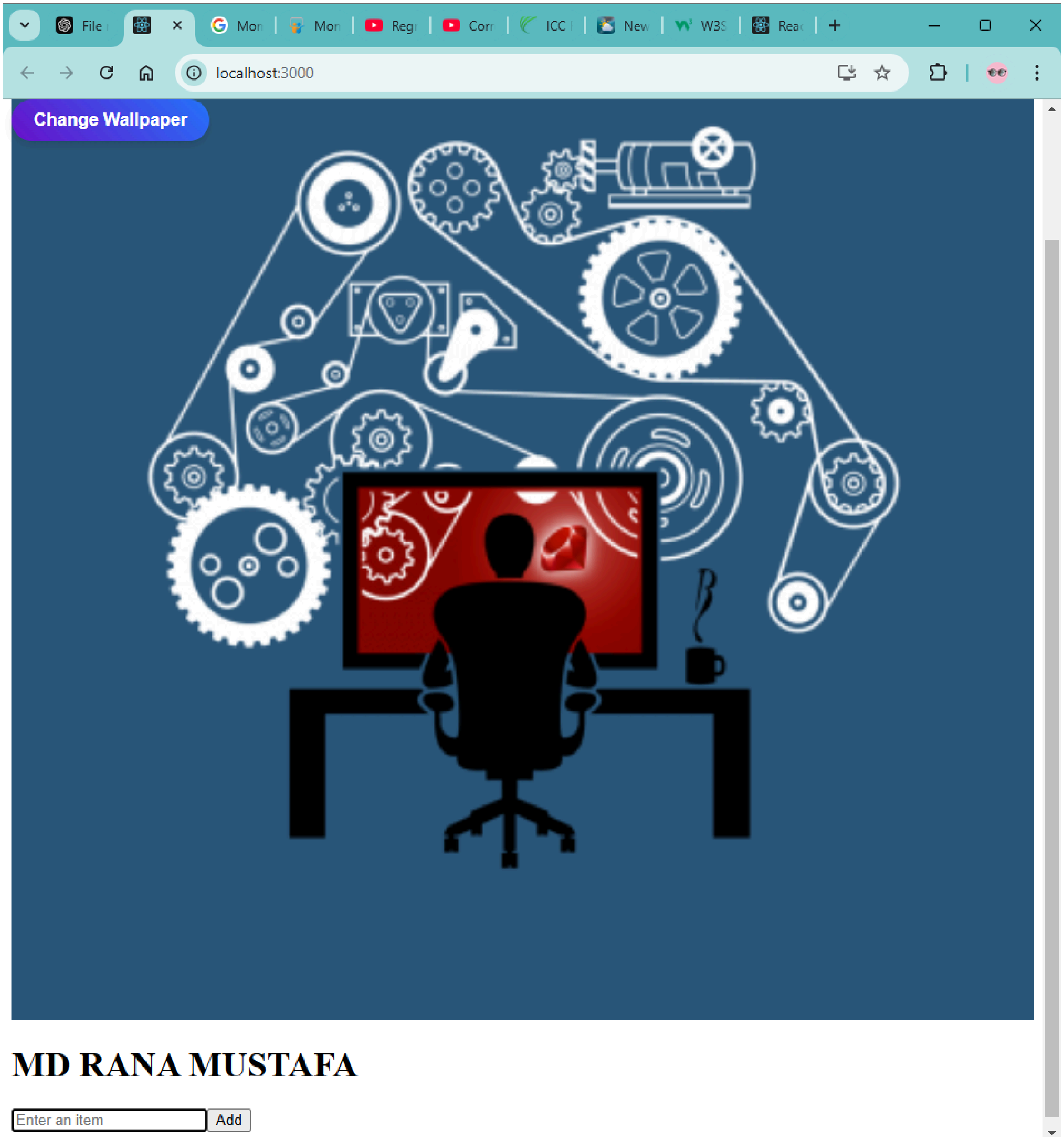
Running the server:

By using the cmd “npm start “ i have launched the frontend ,below is the screenshots of my login ,signup, home page . A lot to work on this, a primary version of my project(frontend).

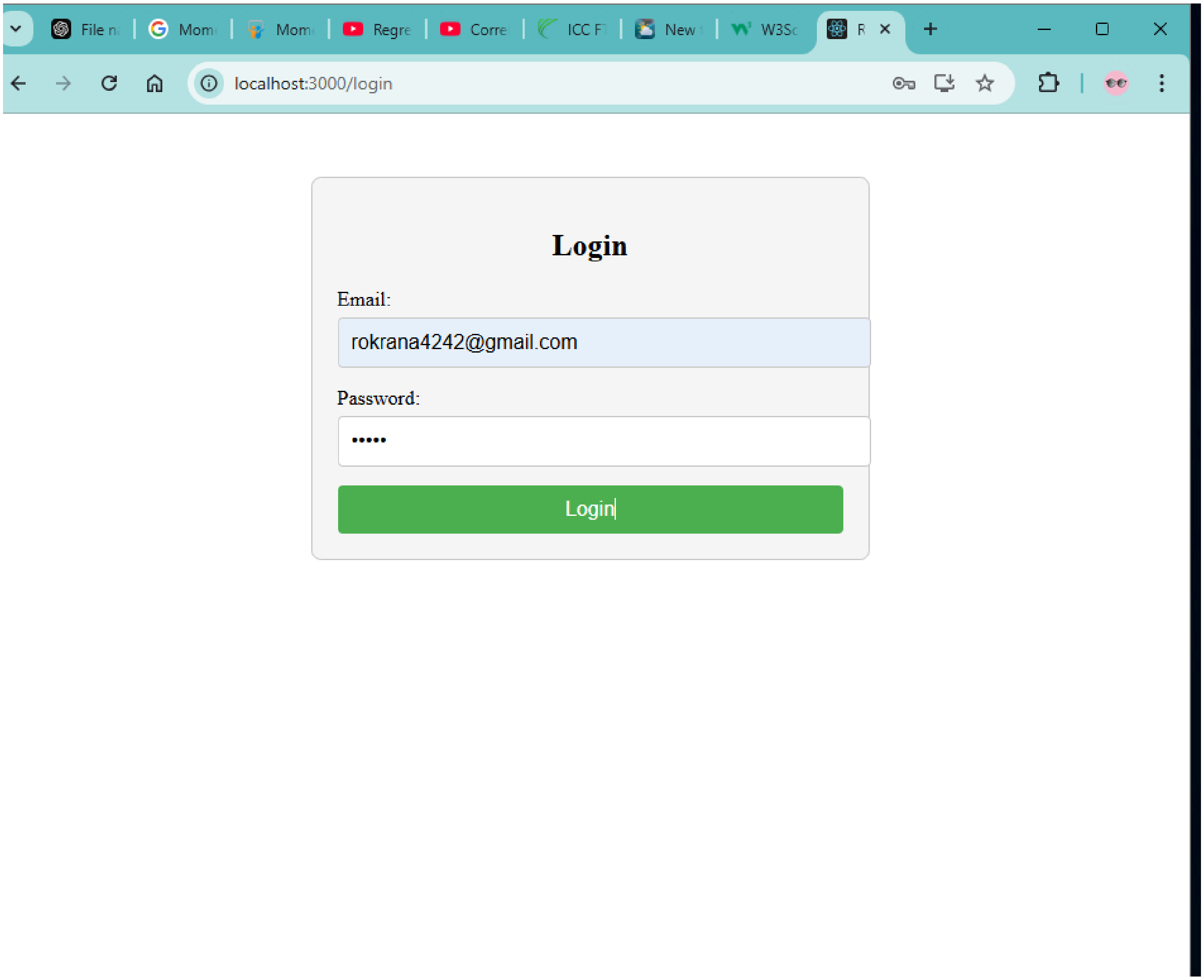
web-page : http://localhost:3000



```
├── @babel/plugin-proposal-private-property-in-object@7.21.11
├── @babel/plugin-proposal-private-property-in-object@7.21.11
├── cra-template@1.2.0
├── cra-template@1.2.0
├── react-dom@18.3.1
├── react-router-dom@7.1.0
├── react-scripts@5.0.1
├── react@18.3.1
├── react@18.3.1
├── web-vitals@4.2.4
└── web-vitals@4.2.4

Compiled successfully!

You can now view client in the browser.

  Local:            http://localhost:3000
  On Your Network:  http://192.168.0.109:3000

Note that the development build is not optimized.
To create a production build, use yarn build.

webpack compiled successfully
```



MD RANA MUSTAFA



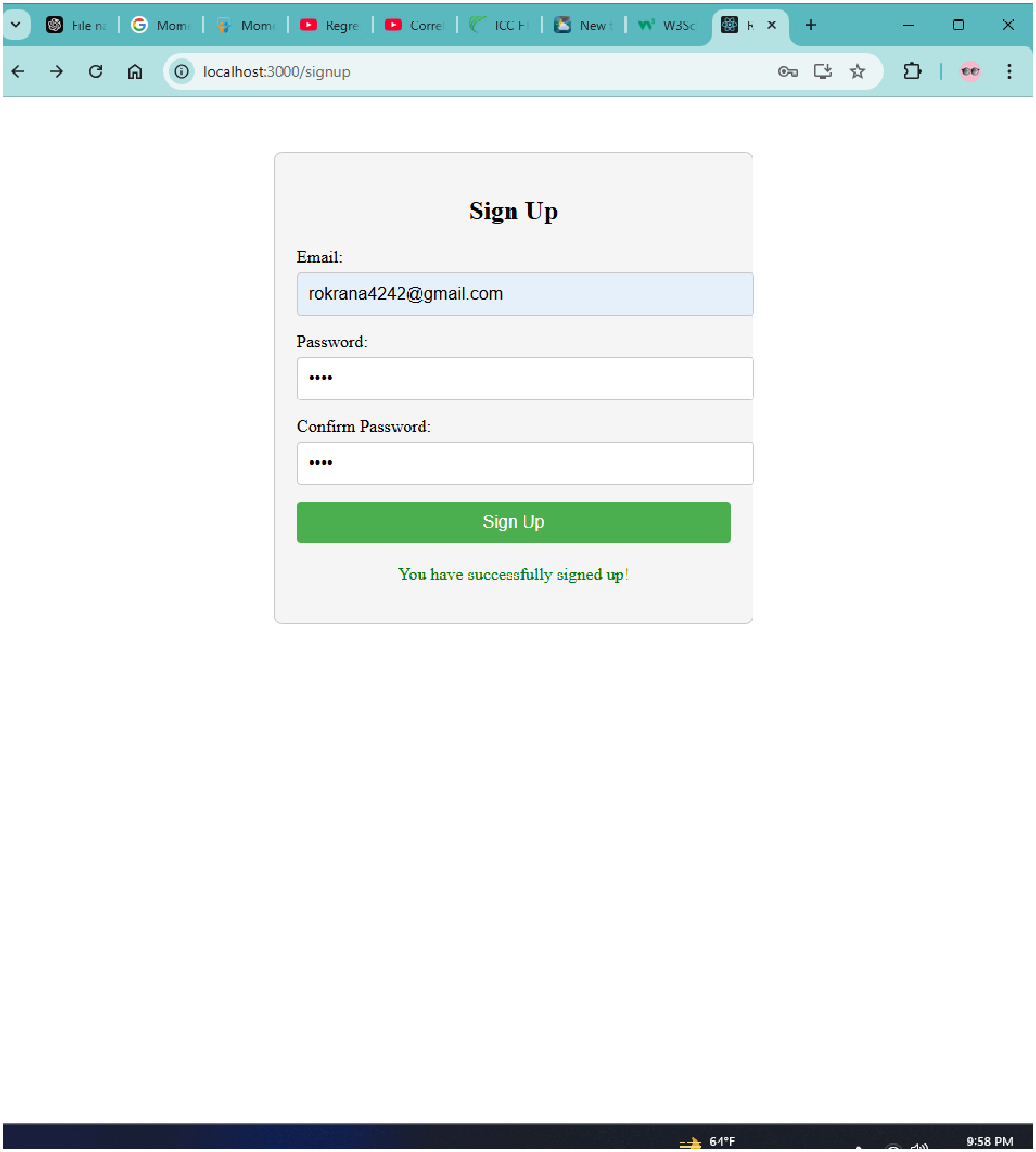
Login

Email:

rokrana4242@gmail.com

Password:

•••••



localhost:3000/signup

Sign Up

Email:

rokrana4242@gmail.com

Password:

••••

Confirm Password:

••••

You have successfully signed up!