**Full Stack Development with MERN**

**Frontend Development Report**

|  |  |
| --- | --- |
| Date | 18-07-2024 |
| Team ID | SWTID1720167264c |
| Project Name | Cab Booking App |
| Maximum Marks | 10 |

**Project Title: Cab Booking App**

Date: 18-07-2024

Prepared by: NANDAM SAKETHRAM

**Objective**

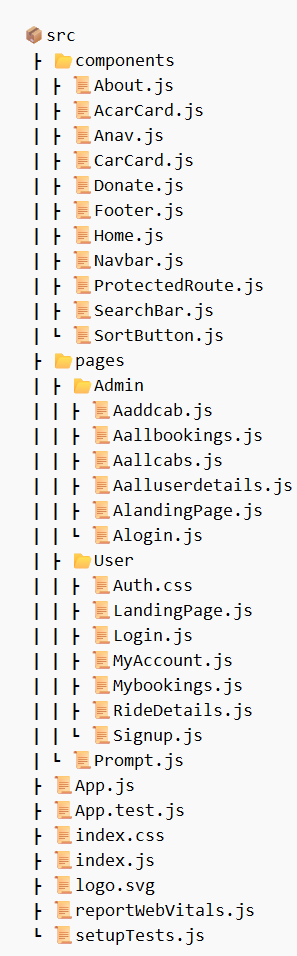
The objective of this report is to document the frontend development progress and key aspects of the user interface implementation for the Cab Booking App project.

**Technologies Used**

* **Frontend Framework:** React.js
* **UI Framework/Libraries:** Bootstrap ,Bootstrap icons
* **API Libraries:** Axios

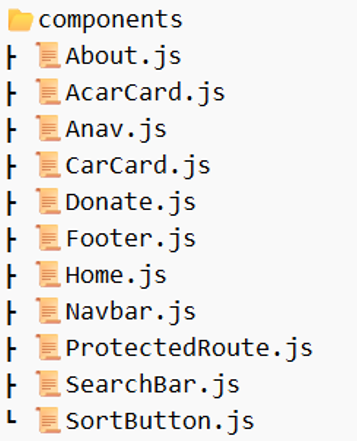
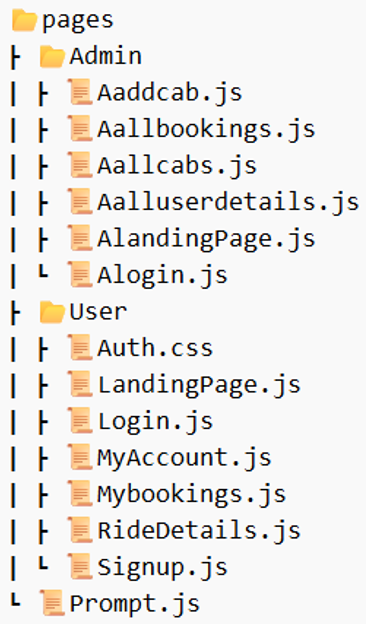
**Project Structure**

Give Screenshot of Frontend Project with explanation

****

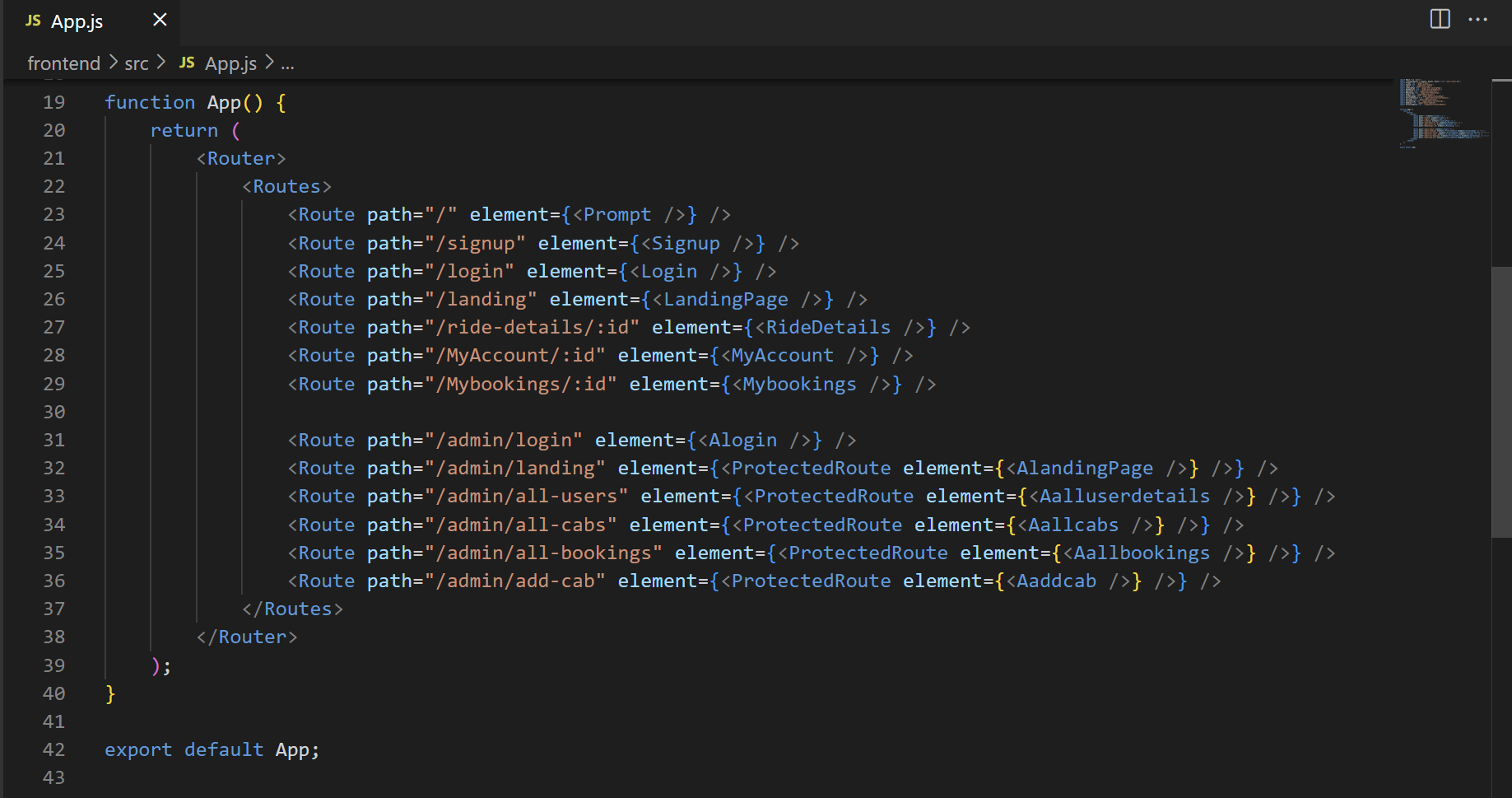
The src folder consists of 2 folders : components and pages , and other default files of react.

**Key Components**

1. **App.js**
   * Responsible for routing and main application layout.
   * The App.js file consists of the Routes (Normal Routes, that anyone can visit and Protected Routes, that only authorised can visit).
2. **/components**
   * Contains reusable UI components used across the application.
   * The ‘components’ folder consists of .js files which are essentially react components that are used in various places of the Application.
   * 
3. **/pages**
   * Includes different pages for Web App.
   * The ‘pages’ folder consists of 2 sub-folders : User and Admin , and a file prompt.js, which asks whether you are a user or an admin.
   * The ‘User’ sub-folder consists of the pages that the user can visit .
   * Similarly, the ‘Admin’ sub-folder is for the Admin.
   * 

**Routing**

Routing is managed using React Router. Here are the main routes:

* 

**Integration with Backend**

The frontend communicates with the backend APIs hoste on localhost:8000 . Key endpoints include:

* **GET /api/data** - Retrieves data for display.
* **POST /api/user/login** - Handles user authentication.

// GET requests

**• GET /getrides** - Retrieves all ride details.

• GET /cabs - Retrieves all cab details.

• GET /getAllUsers - Retrieves all user details.

• GET /getUser/:id - Retrieves details of a specific user.

• GET /getrides/:id - Retrieves details of a specific ride.

• GET /cab/:id - Retrieves details of a specific cab.

// POST requests

• POST /cabs - Adds a new cab with the provided form data.

• POST /adminlogin - Handles admin authentication.

• POST /login - Handles user authentication.

• POST /rides - Adds a new ride with the provided data.

• POST /register - Registers a new user with the provided details.

// PUT requests

• PUT /rides/:id - Updates a specific ride with the provided form data.

• PUT /userEdit/:id - Updates the name of a specific user.

• PUT /userEdit/:id - Updates the email of a specific user.

• PUT /userEdit/:id - Updates the password of a specific user.

// DELETE requests

• DELETE /deleteride/:id - Deletes a specific ride by its ID.

• DELETE /deleteCar/:id - Deletes a specific cab by its ID.

• DELETE /deleteUser/:id - Deletes a specific user by its ID.

**User Interface (UI) Design**

* The UI design follows a clean and modern aesthetic with a focus on usability and responsiveness.
* The design includes a navbar with tabs for navigation, a user-friendly form layout, and visually appealing components such as cards for displaying rides and cabs.
* Implemented using react and bootstrap.
* 