Task:2

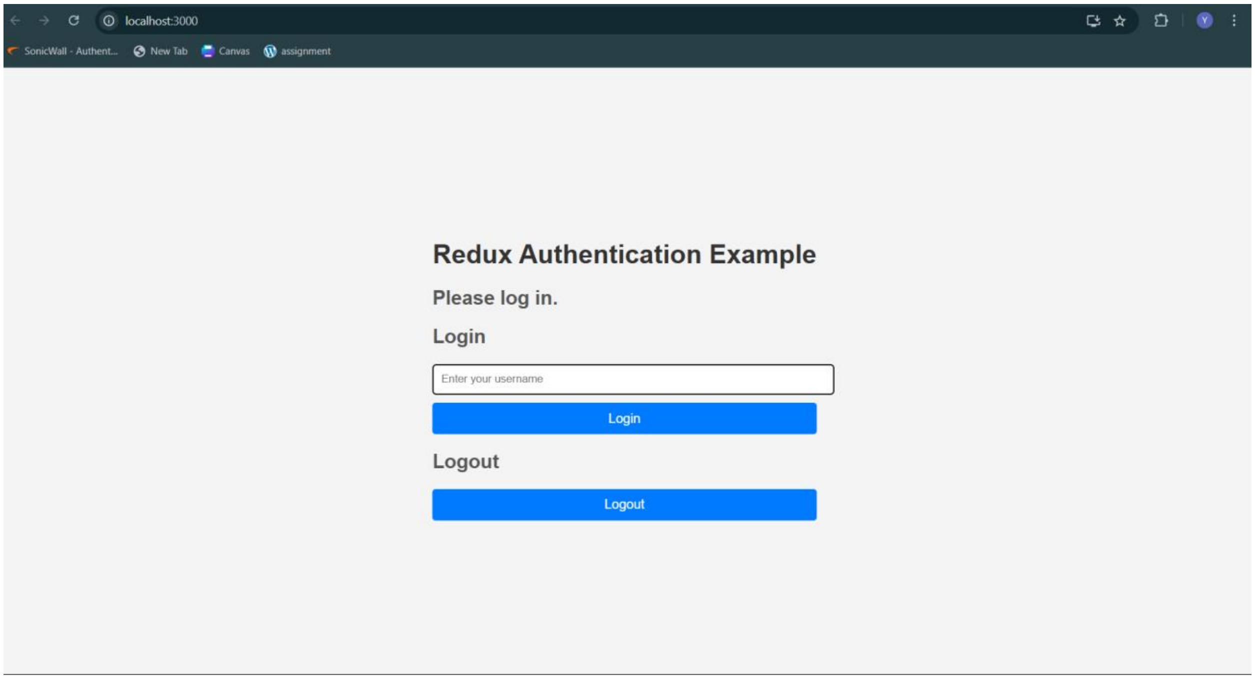
Task Name: Redux Auth Routing

Project Name: REDUX-AUTH-EXAMPLE

1. Task Description

The task is to develop a **Redux store** to manage user authentication states within a React application. This includes handling actions for login, logout, and maintaining the authentication token in the store. The goal is to create a secure, centralized state management system for user sessions, so any part of the app can easily access the user's authentication status and associated data. It should also manage error states and loading during the authentication process.

2. Task Output Screenshot

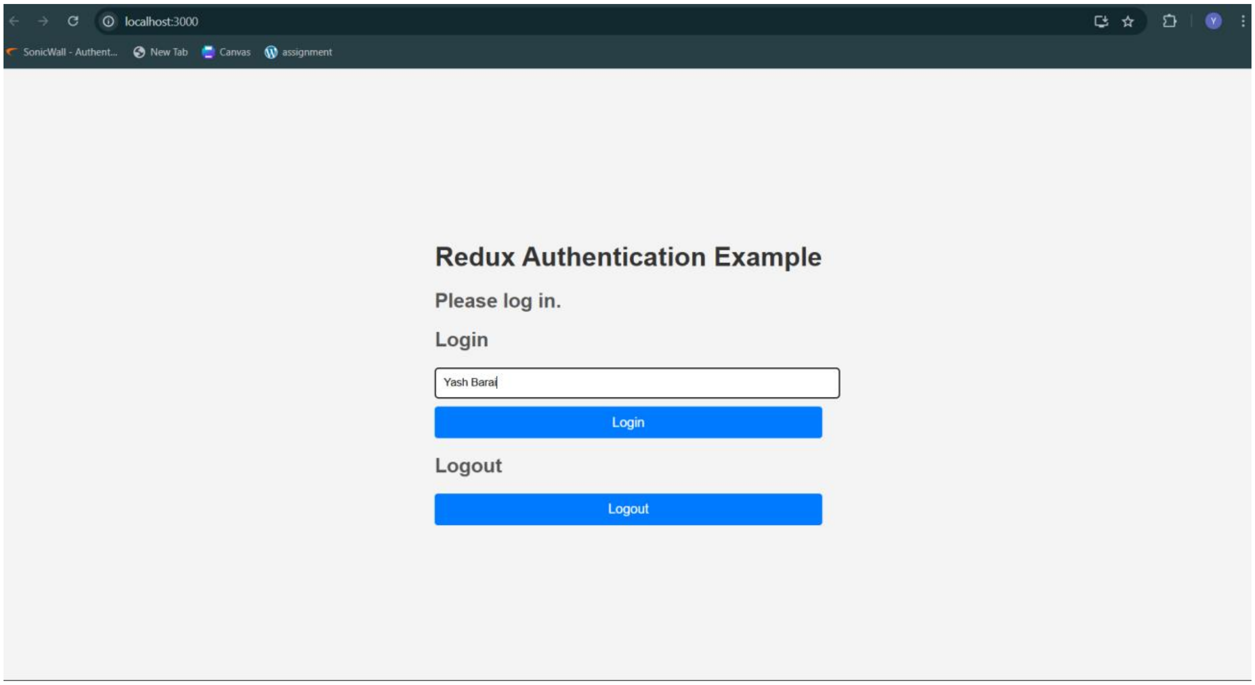


Redux Authentication Example

Please log in.

Login

Logout



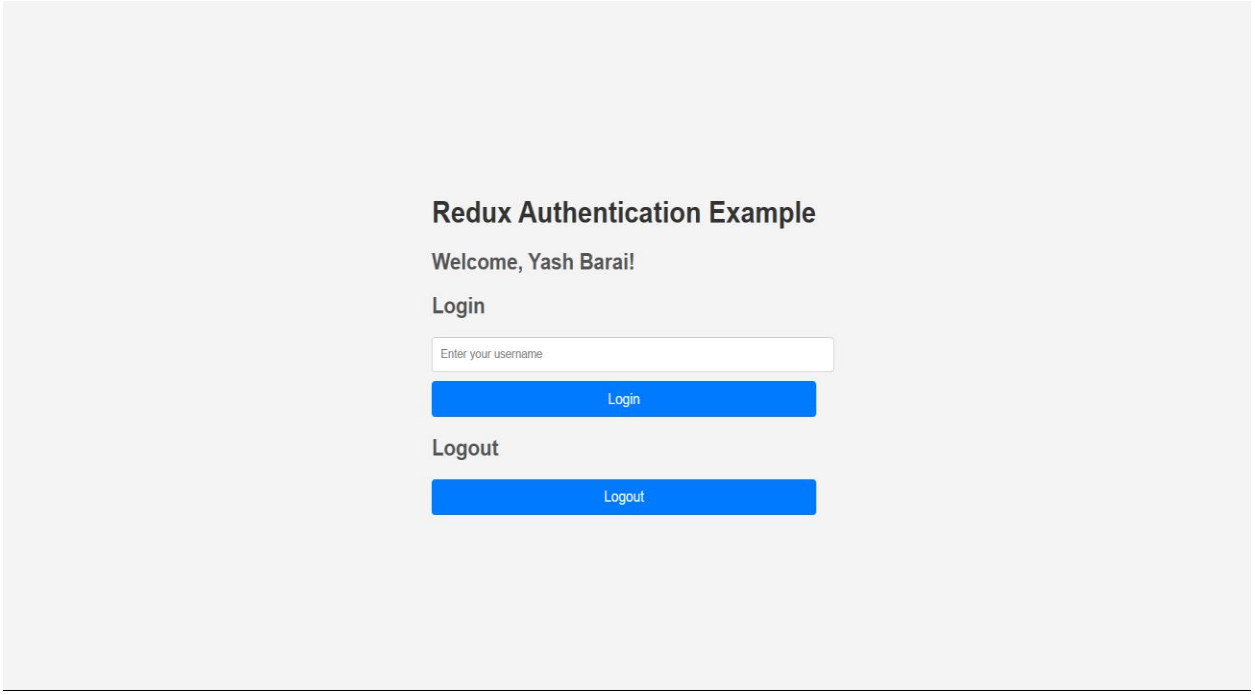
Redux Authentication Example

Please log in.

Login

Yash Barai

Logout



Redux Authentication Example

Welcome, Yash Barai!

Login

Enter your username

Logout

3. Widget/Algorithm Used In Task

- **Redux Store**: To hold the global authentication state (e.g., user data, token, login status).
- **Actions**: For triggering login, logout, and session persistence (e.g., LOGIN_SUCCESS, LOGOUT).
- **Reducers**: To update the store based on dispatched actions.
- **Middleware (Redux Thunk)**: For handling asynchronous actions like making API requests during login.
- **Selectors**: For extracting authentication-related state, like isAuthenticated or userToken.

4. Algorithm explanation

- **Set up Redux store** with initial state including is Authenticated, user, and authToken.
- **Create actions** for LOGIN_REQUEST, LOGIN_SUCCESS, LOGIN_FAILURE, and LOGOUT.
- **Define reducer functions** to handle these actions and update the state (e.g., setting isAuthenticated to true upon login success, clearing the state on logout).
- Use **Redux Thunk** middleware to make async API calls for login (sending credentials to a backend, receiving and storing the authentication token).
- On successful login, dispatch LOGIN_SUCCESS and save the token in local storage or session storage.
- Implement selectors to access isAuthenticated, authToken, and user data from the store in any component that needs it.
- Ensure the token is automatically cleared on logout, and handle edge cases like expired sessions or invalid tokens with error actions and states.

5. How to start:

Install Node Module in frontend and backend folder: Go to terminal "npm install" How to Start: Go to terminal "npm run server" OR "npm start".