**Redux Auth Routing**

1. **Task Description**

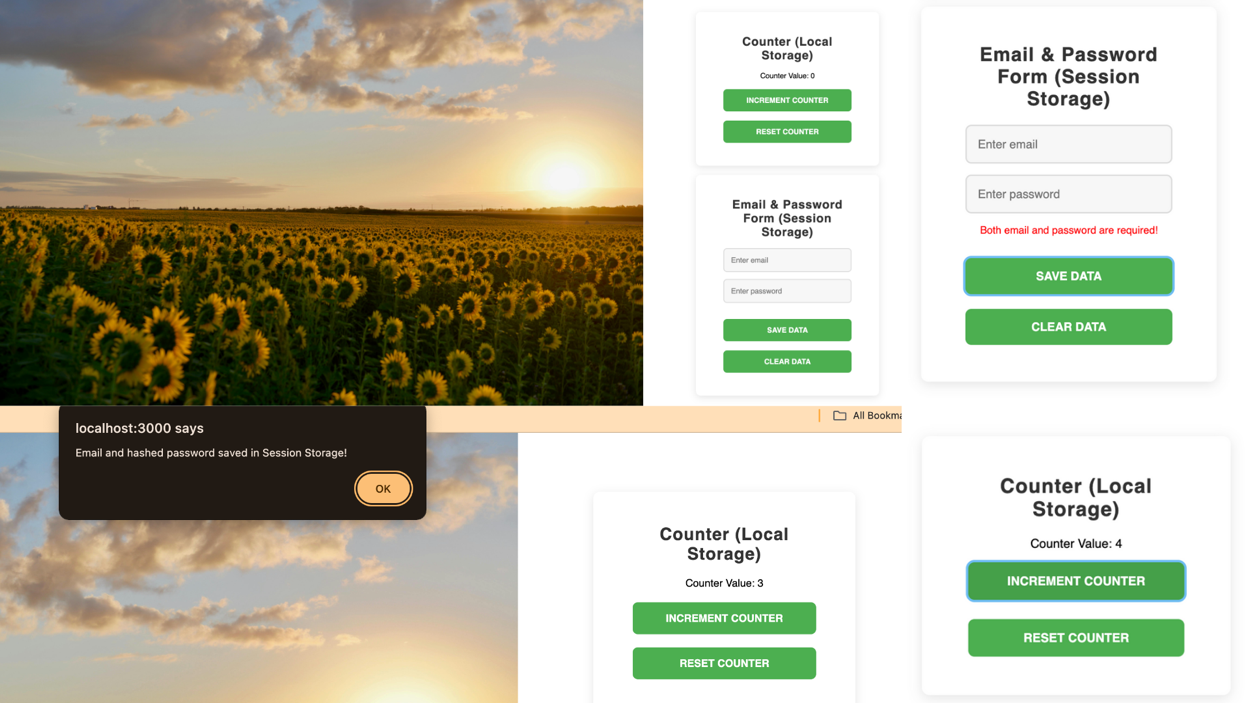
The task involves creating a React component that implements local storage and session storage functionalities. The goal is to build two features:

1. **Counter with Local Storage**: A counter that persists its value across tab closures and reopenings using localStorage. The counter value should remain the same even after the browser tab is closed and reopened.
2. **Email and Password Form with Session Storage**: A form containing email and password fields. The entered values will be saved in sessionStorage. However, once the browser tab is closed, these values should be cleared from sessionStorage.

**The implementation uses React, and the components are:**

* CounterComponent: Manages the counter value with localStorage.
* SessionFormComponent: Manages email and password data with sessionStorage.

1. **Task Output Screenshot**



1. **Widget/Algorithm Used In Task**

**1. Widgets Used:**

* Input Fields (Email & Password):
* input tags are used for taking user input. The email field is type "email", and the password field is type "password".
* Buttons:
* Two buttons are used: one for saving the email and password, and the other for clearing the session storage.

**2. Local Storage:**

* Used to persist the counter's value across browser sessions. The value is stored whenever the counter is updated and retrieved when the component is loaded again.

**3. Session Storage:**

* Used to store the email and password values only for the current browser session. Once the session ends (tab is closed), the data is automatically cleared.

**4. Algorithms/Functionalities:**

* Counter Logic (Local Storage):
* The counter's state is initialized from localStorage if a value exists. The counter value is updated and saved to localStorage whenever the counter changes (incremented). A reset button clears the counter and removes it from localStorage.
* Email and Password Form (Session Storage):
* The form captures user inputs for email and password. The email is stored in sessionStorage, and the password is hashed using SHA-256 before saving in sessionStorage. The form also includes validation to ensure that both fields are filled before saving the data.