Day 5 CC 4: 💻 React Coding Challenge: Advanced Routing & State Management

📝 Problem Statement

Build a **Task Management Application** with the following functionalities:

1. **Authentication System** (Login & Protected Routes).
2. **Task Dashboard** (List, Add, Delete Tasks).
3. **Lazy Loading & Code Splitting** (Optimize Performance).
4. **Redux Middleware & State Management** (Using Redux Thunk).

⏳ **Time Limit: 120 Minutes**

📌 User Stories

✅ **As a user**, I should be able to log in and access the dashboard.
✅ **As a user**, I should be redirected to the login page if I'm not authenticated.
✅ **As a user**, I should be able to navigate between Home, Dashboard, and Profile pages.
✅ **As a user**, I should be able to add and delete tasks using the Redux store.
✅ **As a user**, I should experience smooth page transitions when navigating.
✅ **As a user**, I should benefit from lazy loading to improve app performance.

💡 Hints

- Use **React Router** for navigation.
- Implement **Protected Routes** to restrict access based on authentication.
- Use **Redux for State Management** (Tasks should be managed using Redux).
- Implement **Lazy Loading** with React's `lazy` and `Suspense` for optimized performance.
- Use `localStorage` to persist user authentication state.

📂 Suggested File Structure

bash
CopyEdit

```
/task-manager
 ├── /src
 │   ├── /components
 │   │   ├── Navbar.js
 │   │   ├── TaskList.js
 │   │   ├── TaskItem.js
 │   ├── /pages
 │   │   ├── Home.js
 │   │   ├── Dashboard.js
 │   │   ├── Login.js
 │   │   ├── Profile.js
 │   ├── /redux
 │   │   ├── store.js
 │   │   ├── actions.js
 │   │   ├── reducers.js
 │   ├── App.js
 │   ├── index.js
 ├── package.json
 ├── README.md
```

🎯 Expected Outcome

✅ Users can **log in**, navigate between **pages**, and manage **tasks**.
✅ Users cannot access **Dashboard/Profile** unless authenticated.
✅ App **loads pages lazily**, improving performance.
✅ **Redux** handles **state management** for tasks.

📌 Bonus Tasks (Optional Enhancements)

🔹 **Enhance UI with Tailwind CSS or Bootstrap.**
🔹 **Add Toast Notifications** for success/error messages.
🔹 **Connect Backend API (Express & MongoDB)** for real-world use.
