Name:	Aman Singh Negi
Section:	622 A
UID:	23BCS11672

Event Booking Portal

1. Core Functionality:-

- Create React App or Vite:

```
npx create-react-app event-booking-portal
# or
npm create vite@latest event-booking-portal
```

- Install Dependencies:

```
npm install react-router-dom axios
```

- Folder Structure:

```
src/
components/
   EventCard.jsx
   Navbar.jsx
   BookingForm.jsx
   Dashboard.jsx
 pages/
   Home.jsx
   EventDetails.jsx
   Login.jsx
   Signup.jsx
```

```
  UserDashboard.jsx
AdminDashboard.jsx
context/
  AuthContext.jsx
BookingContext.jsx
data/
  events.json   # mock events data
App.jsx
index.jsx
```

2. Core Features Implementation:-

A. Events List & Event Details

- **Home page (Home.jsx)**: Fetch events from events.json or mock API.
- **EventCard Component**: Display title, date, available seats.
- **EventDetails Page (EventDetails.jsx)**:
  - Show full description, location, date/time, available seats.
  - Booking button (enabled only if seats available).

```
Jsx..
const [events, setEvents] = useState([]);
useEffect(() => {
  fetch("/data/events.json")
    .then(res => res.json())
    .then(data => setEvents(data));
}, []);
```

B. Authentication

- Use **Context API** for global auth state.
- Signup/Login:

- Store user info in **localStorage** for persistence.
- Example structure:

Json..

```
{ "username": "john", "password": "1234", "bookings": [] }
```

3. Routing & State Management:-

- React Router Setup:

  Jsx..

```
<BrowserRouter>
  <Navbar />
  <Routes>
    <Route path="/" element={<Home />} />
    <Route path="/event/:id" element={<EventDetails />} />
    <Route path="/login" element={<Login />} />
    <Route path="/signup" element={<Signup />} />
    <Route                                       path="/dashboard"
element={<PrivateRoute><UserDashboard /></PrivateRoute>} />
    <Route path="/admin" element={<PrivateRoute><AdminDashboard
/></PrivateRoute>} />
  </Routes>
</BrowserRouter>
```

4. UI/UX & Responsiveness:-

- Use **CSS Flex/Grid** or **TailwindCSS** for responsive design.
- Components should be reusable: `EventCard`, `BookingForm`.
- Show **loading** and **error messages** during API calls.
- Keep design clean and intuitive.

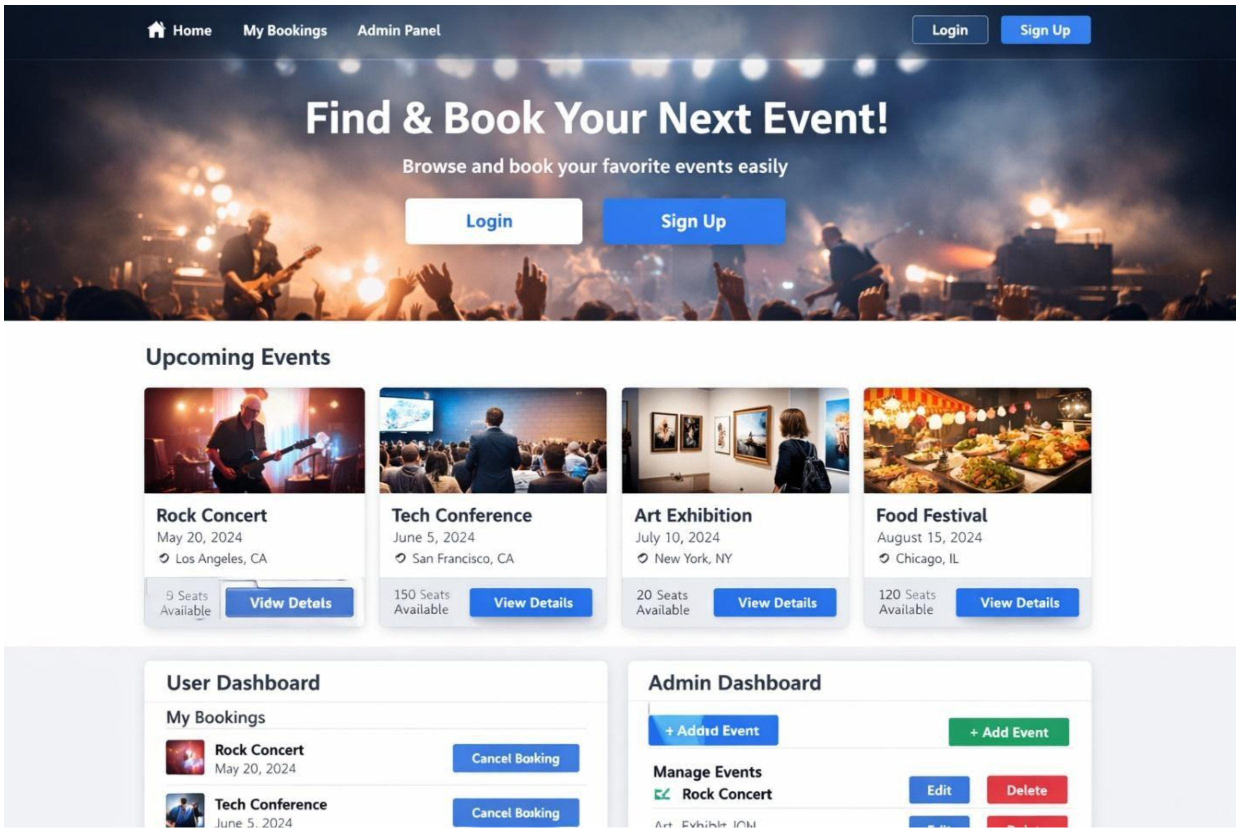
5. Code Quality:-

- Keep folder structure modular.

- Use reusable components (`EventCard`, `Button`, `FormInput`).
- Comment where necessary (logic-heavy parts like booking flow or auth).
- DRY principle: avoid repeating fetch logic or form handling.

6. Extra Features:-

- **Email confirmation mock**: after booking, `console.log("Email sent to user@example.com")`.
- **Pagination/Search**: on `Home` page, filter events by name or date.


Home
My Bookings
Admin Panel
Login
Sign Up
Find & Book Your Next Event!
Browse and book your favorite events easily
Login
Sign Up
Upcoming Events
Rock Concert
May 20, 2024
Los Angeles, CA
9 Seats Available
Vidw Detals
Tech Conference
June 5, 2024
San Francisco, CA
150 Seats Available
View Details
Art Exhibition
July 10, 2024
New York, NY
20 Seats Available
View Details
Food Festival
August 15, 2024
Chicago, IL
120 Seats Available
View Details
User Dashboard
My Bookings
Rock Concert
May 20, 2024
Cancel Boking
Tech Conference
June 5, 2024
Cancel Boking
Admin Dashboard
+ Addad Event
+ Add Event
Manage Events
Rock Concert
Edit
Delete