**PHASE-WISE PROJECT DEVELOPMENT PLAN**

**🚀 PHASE 1: PROJECT INITIAL SETUP**

**✅ Step 1: Create the React + TypeScript + Tailwind Project**

bash

npm create vite@latest autozone-dashboard -- --template react-ts

cd autozone-dashboard

npm install

npm install axios react-router-dom zustand clsx

npm install -D tailwindcss postcss autoprefixer

npx tailwindcss init -p

**✅ Step 2: Setup Tailwind CSS**

Update tailwind.config.js:

content: ["./index.html", "./src/\*\*/\*.{js,ts,jsx,tsx}"]

In src/index.css:

@tailwind base;

@tailwind components;

@tailwind utilities;

**🗂️ PHASE 2: FILE STRUCTURE & ROUTING**

**✅ Step 3: Create Folder Structure**

less

CopyEdit

src/

├── assets/

├── components/

│ ├── common/ // Layout, Navbar, Sidebar

│ └── dashboard/ // Cards, Tables

├── pages/

│ ├── auth/ // Login, Signup

│ ├── admin/

│ ├── user/

│ └── company/

├── routes/ // ProtectedRoutes

├── services/ // API functions

├── store/ // Zustand store

├── utils/ // Constants, formatters

├── App.tsx

└── main.tsx

**✅ Step 4: Setup Routing (React Router)**

In App.tsx:

tsx

import { BrowserRouter, Routes, Route } from 'react-router-dom'

import Login from './pages/auth/Login'

import UserDashboard from './pages/user/UserDashboard'

import AdminDashboard from './pages/admin/AdminDashboard'

import CompanyDashboard from './pages/company/CompanyDashboard'

function App() {

return (

<BrowserRouter>

<Routes>

<Route path="/" element={<Login />} />

<Route path="/user" element={<UserDashboard />} />

<Route path="/admin" element={<AdminDashboard />} />

<Route path="/company" element={<CompanyDashboard />} />

</Routes>

</BrowserRouter>

)

}

export default App

**🔐 PHASE 3: MOCK AUTHENTICATION & ROLE-BASED ROUTING**

**✅ Step 5: Create Login Page**

In pages/auth/Login.tsx, build a login form:

* Mock dropdown to choose role: admin, user, company
* Save role in Zustand store (or React context)

**✅ Step 6: Role-Based Route Protection**

Create ProtectedRoute.tsx in routes/:

tsx

CopyEdit

import { Navigate } from "react-router-dom";

import { useAuthStore } from "../store/auth";

const ProtectedRoute = ({ children, role }: { children: JSX.Element; role: string }) => {

const { userRole } = useAuthStore();

return userRole === role ? children : <Navigate to="/" />;

};

export default ProtectedRoute;

Use it like:

<Route path="/admin" element={<ProtectedRoute role="admin"><AdminDashboard /></ProtectedRoute>} />

**🧾 PHASE 4: BACKEND MOCK DATA (JSON Server)**

**✅ Step 7: Create db.json**

Use the db.json from your document (includes cars, users, parts, services, bookings)

Run server:

json-server --watch db.json --port 3001

**✅ Step 8: Create Axios Services**

In services/carService.ts:

import axios from "axios";

const API = "http://localhost:3001";

export const fetchCars = async () => {

const res = await axios.get(`${API}/cars`);

return res.data;

};

Do similar for services, parts, bookings, users.

**📦 PHASE 5: DASHBOARDS & CRUD OPERATIONS**

**✅ Step 9: User Dashboard**

* List of cars
* View service options
* Book a service (form + POST request)
* Track status

**✅ Step 10: Company Dashboard**

* Add/Edit/Delete: Cars, Spare Parts, Services
* Show service requests

**✅ Step 11: Admin Dashboard**

* View all users, companies, and bookings
* Approve/Reject requests
* Add/Delete entities

**🎨 PHASE 6: UI & POLISHING**

**✅ Step 12: UI Enhancements**

* Use Tailwind CSS cards, tables
* Add loading states, error handling
* Make it mobile responsive

**✅ Step 13: Extras**

* Search & filters
* Charts (use Recharts or Chart.js)
* PDF invoice download
* Dark mode (optional)

**✅ RESULT: YOU WILL MASTER**

* React + TypeScript
* React Router & Zustand
* API integration (CRUD)
* Form handling & validation
* Real dashboard logic (role-based)
* Frontend architecture & reusable code
* UI skills with Tailwind/MUI
* Interview-ready confidence

Would you like me to:

* Generate the Login.tsx and Zustand store?
* Create one full working dashboard page (like Car Listing)?

Let’s go step-by-step together 💪