**InsurAI – Corporate Policy Automation and Intelligence System**

**Problem Statement**

Managing corporate insurance policies is often complex, manual, and error-prone. Traditional systems require heavy human intervention for policy creation, claims settlement, compliance checks, and reporting. This leads to delays, inconsistencies, and higher operational costs.  
**InsurAI aims to automate and streamline these processes using AI and modern web technologies, improving efficiency, accuracy, and customer satisfaction.**

**1. 📌 Project Overview**

**InsurAI (Insurance AI Automation & Verification System)** is a **web-based platform** designed to:

* Automate **corporate policy management**.
* Provide **AI-powered verification** and **expert validation**.
* Deliver a **modern, user-friendly interface** with **dark/light mode support**.

The platform allows:

* Users to **register, login, and access dashboards**.
* Viewing and managing **policies**.
* Verifying policies against **industry standards**.
* Visualizing real-time updates and automation status.

**2. 🎯 Core Features (MVP Scope)**

* **Authentication**
  + User registration & login.
  + Role-based access (User, Admin).
* **Dashboard**
  + Displays policy stats (active, pending, verified).
  + Personalized after login (⚠️ No dummy stats for guests).
* **Policies Management**
  + Create, edit, view, and delete corporate policies.
  + Integration with AI/ML verification (future enhancement).
* **Verification System**
  + AI + Expert validation workflows.
  + Fraud detection (future).
* **User Experience**
  + Responsive UI (mobile + desktop).
  + Light/Dark mode toggle.
  + Animations (Framer Motion).

**3. 🛠️ Tech Stack**

**Frontend**

* **React.js** (SPA architecture).
* **React Router v6** (page navigation).
* **Tailwind CSS** (utility-first styling).
* **Framer Motion** (animations).

**Backend (recommended options)**

* **Option A: Node.js (Express.js) + REST APIs**.
* **Option B: Python (FastAPI/Django) if AI-first backend is preferred**.

**Database (recommended)**

* **MySQL** (structured corporate policy data, widely used).
* Alternative: **PostgreSQL** (better for scalability & JSON policies).

**Future Integrations**

* **AI models (NLP, ML)** for policy analysis.
* **Cloud deployment**: AWS / Azure / GCP.
* **CI/CD pipelines** for scaling.

**4. 🎨 UI/UX Design Guidelines**

* **Theme**
  + Light: Soft Indigo + Blue gradients.
  + Dark: Gray + Indigo gradients.
  + Smooth transitions with transition-colors duration-500.
* **Typography**
  + font-extrabold for headers.
  + font-medium for navigation.
* **Animations**
  + Page transitions: framer-motion.
  + Hover effects: Tailwind + Motion.
* **Responsiveness**
  + Flexbox & Grid system.
  + Mobile-first design.

**5. 🔐 Security & Auth (Planned)**

* JWT-based authentication.
* Encrypted password storage (bcrypt).
* Role-based access control.
* Session management & logout.

**6. 📅 Roadmap**

| **Phase** | **Features** | **Status** |
| --- | --- | --- |
| **Phase 1** | React setup, routing, Tailwind UI | ✅ Done |
| **Phase 2** | Home, Navbar, Footer, Theme Toggle | ✅ Done |
| **Phase 3** | Dashboard, Policies, Verify pages | ⚡ In progress |
| **Phase 4** | Auth (Login/Register + JWT backend) | ⏳ Planned |
| **Phase 5** | Database integration (MySQL/Node) | ⏳ Planned |
| **Phase 6** | AI verification module | ⏳ Planned |
| **Phase 7** | Deployment (Netlify/Vercel + backend cloud) | ⏳ Planned |