# **📌 High-Level Architecture – InsureConnect**

### **Goals of HLD**

* Show **all major components** of the system.
* Demonstrate **interaction between users, system, and insurers**.
* Highlight **distributed design, scalability, and extensibility**.

## **1. Actors**

* **Customer** – web/mobile app
* **Agent** – internal dashboard
* **Admin** – system management
* **Insurer API** – external systems

## **2. Major Components**

1. **API Gateway / Django REST Framework**
   * Single entry point for all clients (mobile, web).
   * Handles JWT authentication.
2. **User Module**
   * Customer registration, login, profile management.
   * Role-based access control (customer/agent/admin).
3. **Insurance Modules**
   * **Motor Insurance Module**
   * **Health Insurance Module**
   * **Life Insurance Module** Each module handles:
   * User input validation
   * Quote request processing
   * Purchase & policy issuance
4. **Quote Engine**
   * Aggregates requests from all insurer adapters.
   * Normalizes insurer responses.
   * Returns a **comparable quote list**.
5. **Insurer Integration Layer**
   * Adapter pattern → one adapter per insurer.
   * Handles retries, failure fallback.
6. **Policy Management**
   * Store purchased policy metadata in DB.
   * Generate PDF documents.
   * Store expiry date for renewal reminders.
7. **Notification Service**
   * Async tasks using **Celery + Redis**.
   * Sends email/SMS for purchase confirmation & renewal.
8. **Admin & Analytics**
   * View sales, commissions, system health.
   * Monitor insurer API failures & retries.

## **3. Data Flow (Simplified)**

[Customer/Agent]

↓

API Gateway

↓

[Insurance Modules] → [Quote Engine] → [Insurer Adapters] → [Insurer APIs]

↓

[Policy Management] → [PDF Generation]

↓

[Notification Service] → [Email/SMS]

## **4. Distributed / Scalable Considerations**

* **Celery Workers** → for heavy tasks (PDF, notifications).
* **Redis** → cache insurer quotes for faster responses.
* **Load Balancer + API Gateway** → horizontal scaling.
* **Database** → PostgreSQL (structured), can use read replicas.
* **Insurer API Adapters** → fault-tolerant, retries, circuit breaker pattern.