Architectural Design: Medical Insurance System

Applying the 3-Tier Client-Server Architectural Style to ensure Security, Scalability, and Performance.

Group 11:

Joseph Kariuki SCT212-0147/2022
Clinton Maina SCT212-0591/2022
Mwema Jonah SCT212-0165/2022

Giovanni Opiyo SCT212-0199/2022
Daniel Mwangi SCT212-0575/2022
Mercy Gitonga SCT212-0550/2022

01. System Analysis

Defining the actors and core functional modules for a robust insurance ecosystem.

Key System Stakeholders

Medical Providers

Hospitals and clinics responsible for verifying patient eligibility and submitting claims for services rendered.

Insurance Agents

Front-line staff tasked with creating policies, onboarding new members, and managing renewals.

Claims Adjudicators

Internal specialists who review, approve, or reject claims based on predefined business rules.

System Administrators

Security-focused users managing access, configurations, and maintaining the overall system health.

Core Functional Modules

- **Member Management:** Enrollment, history, and dependent handling.
- **Policy Administration:** Product definitions and premium renewals.
- **Claims Processing:** Adjudication rules engine and payment flows.
- **Provider Network:** Managing hospital contracts and credentials.

02. Architectural Design

Proposing the 3-Tier Model to decouple the user interface from critical data assets.

The 3-Tier Client-Server Model

Presentation

Hybrid Client (Web/Desktop) rendering the UI. Performs basic validation to reduce server round-trips.

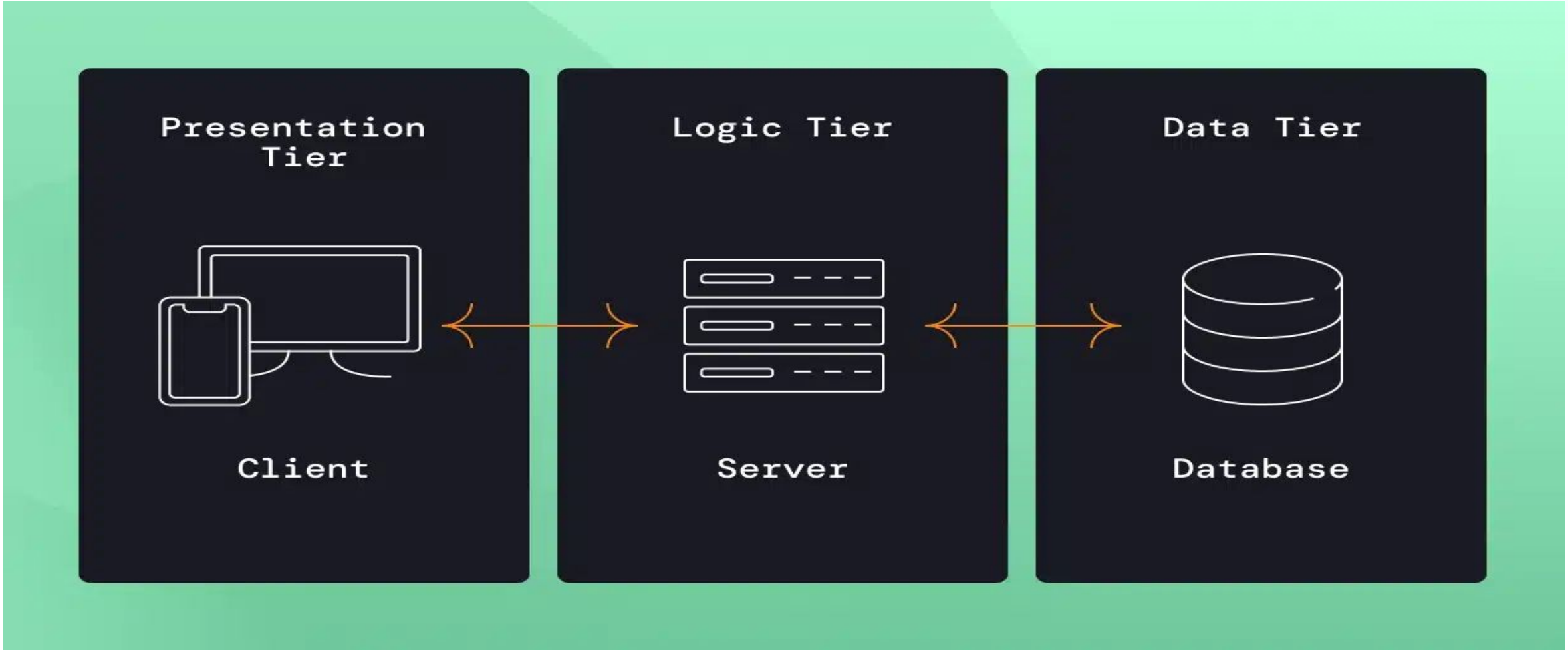
Application

The "Logic Server." Hosts business rules, claims engines, and acts as the secure gatekeeper.

Data

Persistent storage using RDBMS (Oracle/PostgreSQL) ensuring ACID compliance for financial integrity.

The 3-Tier Client-Server Model



Security: Attack Surface Reduction

In a 3-tier model, direct database access is impossible for the client. Even if a client machine is compromised, the attacker must bypass the Application Tier's validation and logic.

Centralized logging facilitates HIPAA compliance by tracking every request (Who, What, When) at the server level.

Performance Optimization

ACID

Compliance Standard

- **Connection Pooling:** Eliminates overhead by maintaining open DB connections.
- **Caching (Redis):** Frequently accessed drug formularies and hospital lists are stored in memory.
- **Offloading Logic:** Complex premium risk assessments are executed on powerful servers.

Scalability Strategy

Handling growth

- **Vertical Scaling:** Upgrading Database Server hardware (CPU/RAM) to handle larger datasets.
- **Horizontal Scaling:** Adding multiple Application Servers behind a Load Balancer to handle concurrent load.

Since the **Application Tier is stateless**, we can easily spin up multiple instances of the logic server to handle thousands of concurrent insurance agents without data loss.

Data Design & Relationships

Entity Relationship	Type	Logic & Justification
Member ---- Policy	1 : N	A single member can hold multiple policies over time.
Policy ---- Claims	1 : N	A claim is strictly linked to a valid, active policy.
Claim ---- ClaimLines	1 : N	Hospital visits include multiple billable items (Pharmacy, Lab).
Schema Model	3NF	Normalization ensures consistency and reduces data redundancy.

Summary of Architectural Trade-offs

Integrity

ACID-compliant RDBMS is used over NoSQL because financial payouts cannot accept "eventual consistency."

Maintenance

Thin Client (Web) approach ensures updates are deployed once to the server rather than 100s of hospital PCs.

Availability

Active-Passive Failover ensures 24/7 processing if the primary Application Server fails.

Image Sources

https://vfunction.com/wp-content/uploads/2024/05/blog-3-tier-application.webp

Source: vfunction.com