System Architecture Documentation – Financial System with Virtual Cards

1. Introduction

This document describes the system architecture for a financial platform that provides virtual cards to users. The system is designed to handle user authentication, KYC verification, card provisioning through third-party APIs, fraud detection via rules engine, payment processing, and notifications.

2. System Overview

The financial system follows a client-server model using a request–response architecture. Clients (mobile or web apps) send HTTPS requests to the backend server, which processes the requests, communicates with external APIs, and returns JSON responses.

3. Core System Modules

1. **Authentication Module:** Handles user login, registration, and session management securely using JWT.
2. **User Management:** Stores and manages user profiles, permissions, and account details.
3. **KYC Module:** Verifies user identity and documents through third-party KYC providers.
4. **Virtual Card Service:** Requests and manages virtual cards from third-party APIs such as Visa or Stripe.
5. **Payment Processor:** Processes payments, handles transaction logs, and integrates with payment gateways.
6. **Rules Engine:** Detects and flags suspicious transactions for fraud prevention.
7. **Notification Service:** Sends SMS, push, or email notifications for important events like payments or KYC updates.

4. Request–Response Flow

In this architecture, clients send HTTPS requests to the API Gateway or Load Balancer. The requests are routed to appropriate backend services. The backend processes the request, interacts with databases and third-party services, and sends back a JSON response.

5. Technology Stack

1. Frontend: React (Web), Flutter (Mobile)
2. Backend: Node.js (Express) or Python (FastAPI)
3. Database: PostgreSQL
4. Cache: Redis
5. Authentication: JWT / OAuth2
6. Hosting: AWS Cloud or Docker containers
7. 3rd Party Integrations: Stripe / Visa / Twilio

6. Security Considerations

Security is a central aspect of the system. All data is transmitted over HTTPS. Sensitive user data such as passwords and card details are encrypted at rest. Role-based access control ensures users only access permitted resources.

7. Conclusion

This architecture provides a modular, secure, and scalable framework for financial systems offering virtual cards. Each module is designed for flexibility, easy maintenance, and integration with third-party services.