# Bank Credit System - System Design & Flowchart (Text-Based)

## System Overview

Design a Bank Credit System:

a. User applies for a credit card.

b. User logs in and accesses a dashboard.

c. Customer makes payments and transactions.

d. User checks and pays outstanding balances.

## Clarifying Questions

i. Business users? B2C or B2B?

ii. Preferred Cloud Provider?

iii. Local or Global Usage?

iv. Peak Load Expectations?

v. Compliance/Security Requirements?

vi. Recovery Time Objective (RTO)?

## Key Non-Functional Requirements

i. Scalability: Use AWS Auto Scaling.

ii. Availability: Use Multi-AZ deployments.

iii. Resilience & Durability: Regular backups and snapshots.

iv. Security: AWS WAF, DDoS protection, API Gateway, IAM.

v. Low Latency: Edge locations, optimized APIs.

vi. Monitoring: AWS CloudWatch.

## Architecture Design (Microservices)

- Presentation Layer (React/Angular for web, React Native for mobile)

- Application Layer (Node.js/Spring Boot microservices, AWS Lambda)

- Database Layer:

* RDS (MySQL) for transactional data

* DynamoDB/MongoDB for audit logs, session data

**Microservices Examples**

- User Service

- Account Service

- Transaction Service

- Payment Gateway Service

- Loan Service

- Notification Service

(API Gateway manages access and routing)

**Testing**

- Integration Testing: Validate inter-service communication.

- Deployment: Use Docker and AWS Elastic Beanstalk for simplicity.

**VPC Design**

- Non-default VPC with CIDR allocation.

- Public Subnets: Web servers with Internet Gateway.

- Private Subnets: App servers, DBs, behind NAT Gateway.

- Multi-AZ deployment for resilience.

**DNS Setup via Route 53**

1. Create a hosted zone for bank.com.

2. Deploy containers behind an ALB (EKS).

3. Get ALB DNS name.

4. In Route 53, create A record (Alias) to ALB DNS name.