**Solution Approach & Design Document**

**Objective**

The objective of this project is to create a transaction service that handles deposit, withdrawal, and transfer operations for accounts. The service communicates with an account service to fetch and update account details.

**Technologies Used**

* Spring Boot
* Feign Client for RESTful communication
* PostgreSQL for database
* Eureka Server for service discovery

**Service Architecture**

The transaction service interacts with the account service using a Feign Client.

Both services are registered with a Eureka Server for service discovery.

**Data Models**

**Transaction Model**

public class Transaction {

private Long id;

private Long accountId;

private String transactionType;

private double amount;

private LocalDateTime timestamp;

private Long destinationAccountId;

}

**Account Model**

public class Account {

private Long id;

private double balance;

}

**TransactionService Class**

**Methods**

deposit(Transaction transaction): Deposits an amount into an account.

withdraw(Transaction transaction): Withdraws an amount from an account.

transfer(Transaction transaction): Transfers an amount from one account toanother.

getTransactionHistory(Long accountId): Fetches transaction history for an account.

**AccountService Class**

**Methods**

getAccountById(Long accountId): Fetches an account based accountId

updateAccount(Account account): Updates the account with current transaction.

**AccountFeignClient Interface**

@FeignClient(name = "account-service")

public interface AccountFeignClient {

@GetMapping("/api/accounts/{id}")

Account getAccountById(@PathVariable Long id);

@PutMapping("/api/accounts/update")

void updateAccount(@RequestBody Account account);

}

**Error Handling**

InsufficientBalanceException: Thrown when there's insufficient balance for withdrawal or transfer.

AccountNotFoundException: Thrown when an account is not found.

**Testing**

Tested the entire flow from depositing, withdrawing, transferring,and fetching transaction history using Postman.

**Deployment**

Deployed the transaction service and account service on separate instances.

Register both services with the Eureka Server.

**Steps to Run My Application**

Start Eureka Server.

Start Account Service.

Start Transaction Service.