Title: Online Bank Management System

- **Subtitle:** EX 8 : OO design – Package, Component and deployment models
- **Name:** Karan Sehgal
- **Registration No:** 22BCE3939
- **Team No: 24**
- **Course/Subject:** Software Engineering Lab (BCSE301P)
- **Instructor's Name:** Dr. Mehfooza M

Date of Submission: 05/03/25

1. Package Model

Overview

The Online Bank Management System is structured into distinct packages to modularize functionalities such as user authentication, transaction processing, loan management, and notifications. Each package contains relevant classes that handle specific banking operations.

Packages and Their Classes

1. **User Management Package**
    - `User` - Represents customers and employees.
    - `Customer` - Stores customer-specific details (e.g., name, email, account information).
    - `Employee` - Handles employee details and tasks.
    - `Admin` - Manages administrative actions (e.g., user account control).
    - `Authentication` - Manages user login, authentication, and access control.
2. **Account Management Package**
    - `Account` - Stores account details (e.g., account type, balance, status).
    - `SavingsAccount` - Inherits from `Account` for savings accounts.
    - `CurrentAccount` - Inherits from `Account` for current accounts.
    - `TransactionHistory` - Stores transaction logs for auditing.
3. **Transaction Processing Package**
    - `Transaction` - Manages all banking transactions.
    - `FundTransfer` - Handles online fund transfers between accounts.
    - `BillPayment` - Manages bill payments through the banking system.
    - `ExternalPaymentGateway` - Connects to third-party payment services.
4. **Loan Management Package**
    - `LoanApplication` - Handles customer loan applications.
    - `LoanApproval` - Processes loan approvals and rejections.

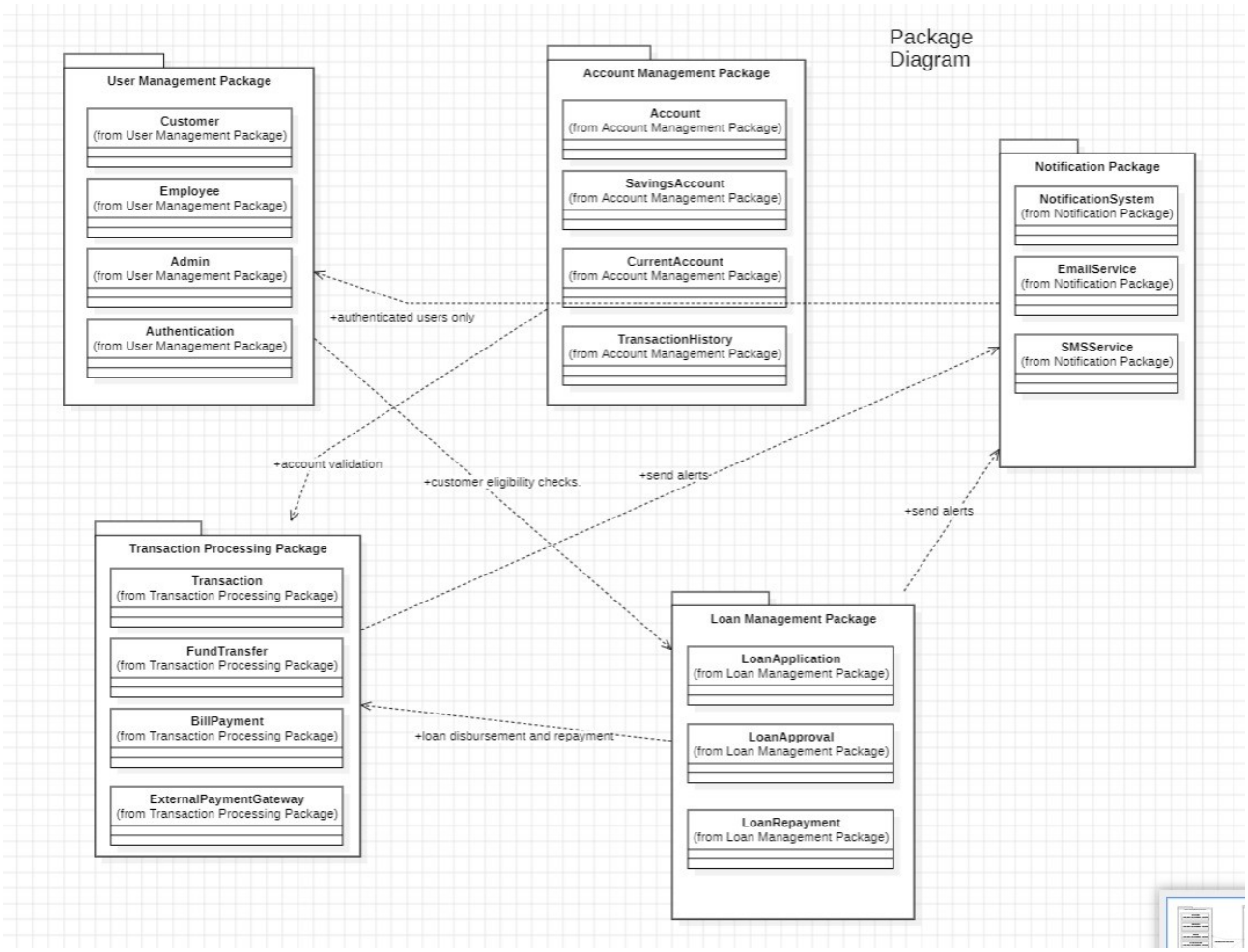
- `LoanRepayment` - Manages EMI payments and schedules.
- `InterestCalculator` - Computes interest and repayment details.

5. **Notification Package**
    - `NotificationSystem` - Sends alerts and messages to customers.
    - `EmailService` - Sends email notifications.
    - `SMSService` - Sends SMS alerts.

Dependencies

- The `Transaction Processing` package depends on `Account Management` for account validation.
- The `Loan Management` package depends on `User Management` for customer eligibility checks.
- The `Notification` package interacts with `Transaction Processing` and `Loan Management` to send alerts.
- The `Authentication` package controls access to all other packages.
- The `Transaction Processing` package depends on the `ExternalPaymentGateway` for online transactions.
- The `Loan Management` package depends on `Transaction Processing` for loan disbursement and repayment.
- The `Admin` class in `User Management` depends on `Account Management` and `Loan Management` for monitoring transactions and loan approvals.
- The `Notification Package` depends on `Authentication` to ensure notifications are sent to authenticated users only.

Package Diagram:



2. Component Model

Overview

The Component Model represents the modular structure of the Online Bank Management System, showing interactions between different components through defined interfaces.

Components and Their Interfaces

1. **User Authentication Service**
    - Interfaces:
        - `registerUser()` - Registers a new customer.
        - `loginUser()` - Authenticates a customer.
        - `updateProfile()` - Allows users to modify account details.
        - `validateUser()` - Verifies login credentials.
    - **Motive:** Handles user authentication and profile updates.
2. **Transaction Processing System**
    - Interfaces:
        - `processTransaction()` - Handles fund transfers and payments.
        - `validateAccount()` - Ensures sufficient balance before transactions.
        - `transferFunds()` - Transfers money between accounts.
    - **Motive:** Ensures secure and accurate financial transactions.
3. **Loan Management System**
    - Interfaces:
        - `applyLoan()` - Submits a new loan request.
        - `approveLoan()` - Approves or rejects a loan.
        - `calculateEMI()` - Computes repayment details.
    - **Motive:** Manages loan applications, approvals, and repayments.
4. **Notification System**
    - Interfaces:

        - `sendNotification()` - Sends general notifications.
        - `sendEmail()` - Sends email alerts.
        - `sendSMS()` - Sends SMS notifications.
    - **Motive:** Keeps customers informed about transactions and account updates.

5. **Database Service**
    - Interfaces:
        - `storeData()` - Saves customer, account, and transaction details.
        - `fetchData()` - Retrieves stored data.
        - `updateData()` - Updates records in the database.
    - **Motive:** Ensures secure and efficient data storage.

6. **Admin Management System**
    - Interfaces:
        - `manageUsers()` - Blocks or unblocks users.
        - `manageAccounts()` - Freezes or deactivates accounts.
    - **Motive:** Allows administrative control over banking operations.

Dependencies

- `User Authentication` interacts with `Transaction Processing` and `Loan Management` for secure transactions.
- `Transaction Processing` and `Loan Management` rely on `Database Service` for data storage.
- `Notification System` works with `Transaction Processing` and `Loan Management` to send alerts.
- `Admin Management System` depends on `Transaction Processing` and `Loan Management` to monitor transactions and loan approvals.

Mappings Between Interfaces

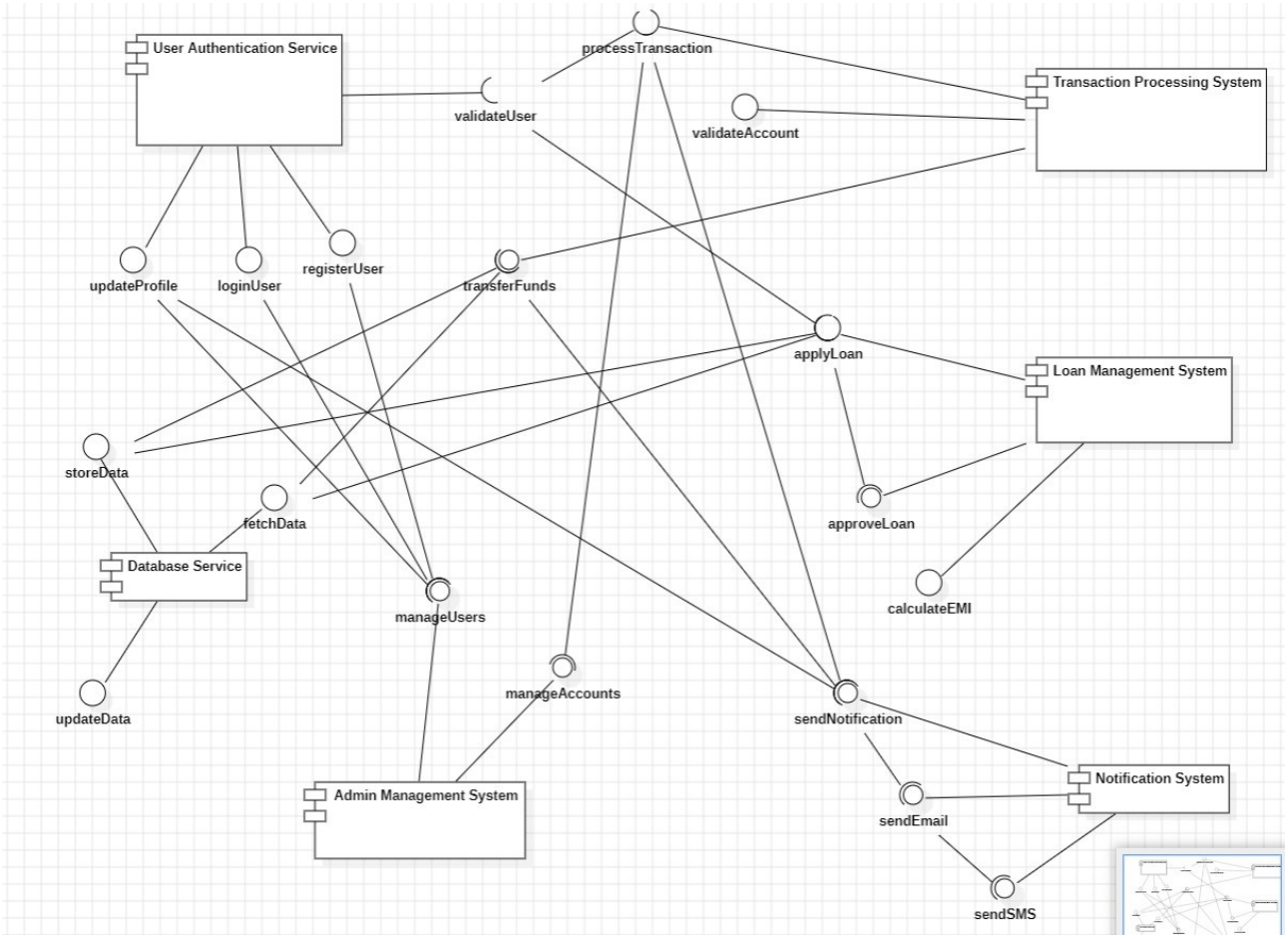
Component	Interface	Interacting Component(s)	Purpose of Interaction
User Authentication Service	`validateUser()`	Transaction Processing System, Loan Management System	Ensures only authenticated users can initiate transactions or apply for loans.
User Authentication Service	`registerUser()`	Database Service	Stores user registration details securely.
User Authentication Service	`updateProfile()`	Database Service	Updates user account details in the database.
Transaction Processing System	`processTransaction()`	Database Service, Notification System	Validates transactions, updates account balances, and notifies users.
Transaction Processing System	`validateAccount()`	User Authentication Service, Database Service	Ensures the user is authenticated and has sufficient balance for transactions.
Transaction Processing System	`transferFunds()`	Database Service, Notification System	Transfers money between accounts and alerts users upon successful transfer.
Loan Management System	`applyLoan()`	Database Service, User Authentication Service	Verifies user details and stores loan applications.
Loan Management System	`approveLoan()`	Admin Management System, Notification System	Sends loan approval/rejection updates to users and administrators.
Loan Management System	`calculateEMI()`	Database Service	Retrieves loan amount and computes EMI details.
Notification System	`sendNotification()`	Transaction Processing System, Loan Management System	Notifies users about transactions and loan updates.
Notification System	`sendEmail()`	Admin Management	Sends email alerts for system-related

Component	Interface	Interacting Component(s)	Purpose of Interaction
		System, Transaction Processing System	actions.
Notification System	`sendSMS()`	Admin Management System, Transaction Processing System	Sends SMS alerts for transactions and critical updates.
Database Service	`storeData()`	User Authentication, Transactions, Loans	Stores various system-related data securely.
Database Service	`fetchData()`	All major components	Provides stored data for business logic processing.
Database Service	`updateData()`	All major components	Updates account, loan, or transaction details based on system actions.
Admin Management System	`manageUsers()`	Database Service, Notification System	Manages user access, blocking, and security.
Admin Management System	`manageAccounts()`	Database Service, Transaction Processing System	Freezes or deactivates accounts as required.

Interaction Flow Example

1. **User logs in** → `validateUser()` (User Authentication → Transaction Processing)
2. **User initiates a transaction** → `processTransaction()` (Transaction Processing → Database Service)
3. **Funds are transferred** → `transferFunds()` (Transaction Processing → Database & Notification)
4. **User receives a confirmation email & SMS** → `sendNotification()`, `sendEmail()`, `sendSMS()` (Notification System)
5. **Admin monitors user activities** → `manageUsers()`, `manageAccounts()` (Admin Management → Database & Notifications)

Component Diagram :



3. Deployment Model

Overview

The Deployment Model illustrates the physical architecture of the Online Bank Management System, showing how different components are deployed across multiple nodes.

Deployment Components

1. **Client System (Web & Mobile Apps)**
    - Hosts: Web Browser, Mobile App
    - Interacts with: User Authentication Service, Transaction Processing System
2. **Load Balancer**
    - Distributes requests to web servers to enhance performance.
3. **Web Server**
    - Hosts: User interface components.
    - Communicates with: Application Server.
4. **Application Server**
    - Hosts: Business Logic (Authentication, Transactions, Loans, Admin Management, Notifications)
    - Communicates with: Database Server, External APIs (e.g., Payment Gateway, SMS Service)
5. **Database Server**
    - Hosts: Database Service
    - Stores: User, Account, Transaction, and Loan Data
    - Implements replication for high availability.
6. **External Services**
    - Includes: Payment Gateway, Email Server, SMS Gateway
    - Communicates with: Application Server for processing payments and sending notifications

Deployment Interactions

- **Clients** (Web/Mobile Apps) send requests to the **Application Server** via the **Load Balancer**.
- **Application Server** processes business logic and interacts with the **Database Server** for data storage and retrieval.
- **Application Server** connects to **External Services** for payments and notifications.
- **Admin System** is hosted on the **Application Server** and manages users and accounts.

Deployment Diagram :