VIT BHOPAL UNIVERSITY,
KOTHRI KALAN, SEHORE MADHYA PRADESH - 466114

December, 2024

CSE3005
Software Engineering

REPORT (GROUP 24)

Online Banking System

Submitted To:
Dr. Sasmita Padhy

Team Members:

Akarsh Jain (22BET10008)

Tarandeep Singh Juneja (22BET10026)

Pravin Kumar (22BET10027)

Shreyansh Soni (22BET10037)

Nihira Pandey (22BET10043)

Dikshant Shubhankar (22BET10048)

in partial fulfillment for the award of the degree of

BACHELOR OF TECHNOLOGY
in
COMPUTER SCIENCE AND ENGINEERING (EDUCATION TECHNOLOGY)

SCHOOL OF COMPUTING SCIENCE AND ENGINEERING

INDEX

1.Objective

Objectives of the Online Banking System:

- **Enhance Accessibility:** Provide banking services 24/7 to users across devices and locations.
- **Improve Security:** Implement advanced security protocols to protect user data and transactions.
- **Increase Efficiency:** Reduce manual intervention and automate repetitive tasks.
- **Boost User Engagement:** Design a user-centric interface for seamless navigation and operations.
- **Support Financial Inclusion:** Reach underbanked populations through simplified processes and mobile banking solutions.

1.1 Study of the Problem

Online banking has transformed the traditional banking experience, enabling customers to perform financial transactions and access services through digital platforms. However, challenges such as security vulnerabilities, limited accessibility for less tech-savvy users, and integration complexities with existing banking systems persist.

Problem Analysis:

- **Customer Convenience:** Traditional banking methods often involve long wait times, limited hours, and physical presence. Customers need a more flexible solution.

- **Security Threats:** Online platforms face risks such as phishing, malware attacks, and data breaches.

- **System Scalability:** As the customer base grows, the system needs to handle high transaction volumes efficiently.

- **User Experience:** An intuitive interface is crucial for ensuring adoption across diverse user groups.

Key Challenges:

1. Ensuring robust cybersecurity measures.
2. Providing uninterrupted and scalable service.
3. Educating users about safe online practices.
4. Seamlessly integrating with existing financial systems.

1.2 Project Scope

The Online Banking System aims to deliver a comprehensive, secure, and user-friendly platform for banking services. This system will empower customers to manage accounts, transfer funds, apply for loans, and access banking services remotely.

Features:

1. **Account Management:** View balances, transaction history, and account details.
2. **Fund Transfer:** Perform intra- and inter-bank transactions.

3. **Bill Payments:** Pay utility bills, credit card dues, and more.
4. **Loan Management:** Apply for loans and track repayment schedules.
5. **Customer Support:** Chatbots and live support for customer queries.
6. **Security Measures:** Two-factor authentication, encryption, and fraud detection mechanisms.

Beneficiaries:

1. Individual customers seeking convenience.
2. Businesses require streamlined banking solutions.
3. Banks looking to enhance operational efficiency and customer satisfaction.

1.3 Purpose

The purpose of the Online Banking System is to create a secure, efficient, and user-friendly platform that empowers customers by providing easy access to essential banking services. It seeks to eliminate the barriers of traditional banking, such as limited working hours and the need for physical presence, while ensuring the highest level of convenience and trust. By leveraging cutting-edge technology, the system aims to redefine the way customers interact with their banks and manage their financial lives.

Key Aspects of Purpose:

1. Simplify financial transactions for users.
2. Reduce operational overhead for banks.

3. Minimize security risks through advanced encryption and fraud detection.
4. Foster financial literacy and inclusion through accessible digital tools.

1.4 Vision

The vision of the project is to establish a benchmark for digital banking solutions by creating a platform that not only meets current market demands but also anticipates future trends. This vision is rooted in the belief that technology can bridge the gap between banks and their customers, providing seamless services while addressing security and accessibility challenges. By aligning with evolving technological advancements, the system aims to ensure scalability, adaptability, and long-term customer satisfaction.

Vision Highlights:

1. Be the leader in secure and innovative digital banking solutions.
2. Create an inclusive platform accessible to all demographics.
3. Continuously improve the system to incorporate AI, machine learning, and big data analytics.
4. Adapt to global banking trends and compliance standards.

2. SRS Document

Online Banking System

2.1. Introduction

The Online Banking System (OBS) provides a platform for users to perform financial transactions, access account details, and manage their finances anytime and anywhere. The system integrates advanced security features, user-friendly interfaces, and robust functionality to meet the diverse needs of individual and business users.

2.1.1 Purpose

The purpose of this document is to outline the functional and nonfunctional requirements for the Online Banking System. It will serve as a guide for the development, deployment, and maintenance phases of the project, ensuring all stakeholders have a shared understanding of the system's capabilities and limitations. This document also aims to provide detailed technical specifications and use cases to facilitate the development process.

2.1.2 Scope

The Online Banking System aims to provide 24/7 access to banking services through web and mobile platforms. Users can manage their accounts, transfer funds, pay bills, and apply for loans securely. Administrators will have tools to monitor and manage the system effectively. The system also aims to provide financial literacy resources and promote secure online banking practices among users.

Key features include:

- Secure user authentication.
- Real-time transaction processing.
- Comprehensive account management.
- Admin tools for system monitoring and management.
- Notifications and alerts for transactions and account activities.
- Multi-language support to cater to diverse user bases.

2.1.3 Definitions, Acronyms, and Abbreviations

- **OBS:** Online Banking System.
- **UI:** User Interface.
- **API:** Application Programming Interface.
- **2FA:** Two-Factor Authentication.
- **SSL:** Secure Socket Layer.
- **OTP:** One-Time Password.
- **UAT:** User Acceptance Testing.

2.2.Overall Description

2.2.1 Product Perspective

The Online Banking System is a web-based application integrated with mobile platforms. It utilizes cloud infrastructure for scalability and high availability. The system integrates with external APIs for payment processing and government compliance. It also includes a robust backend for administrators to monitor and optimize system operations.

2.2.2 Product Functions

1. **User Account Management:** View balances, update profile details, and manage linked accounts.
2. **Transaction Processing:** Fund transfers, bill payments, scheduled transactions, and international remittances.

3. **Loan Management:** Loan applications, approvals, and repayment tracking with notifications.
4. **Customer Support:** Chatbots, live agent support, and FAQs for troubleshooting.
5. **Admin Panel:** Monitor system activity, generate analytics reports, and oversee security protocols.
6. **Notifications:** SMS, email, and app-based alerts for transaction updates and system announcements.

2.2.3 User Classes and Characteristics

1. **Individual Users:** Access personal banking services with ease, including savings and checking accounts.
2. **Business Users:** Perform high-value transactions, manage payroll, and access corporate financial features.
3. **Administrators:** Manage system performance, security, and user accounts, with tools for detailed analytics and monitoring.

2.2.4 Assumptions and Dependencies

- Users must have access to the internet and a compatible device.
- The system will comply with regional banking regulations and standards.
- Third-party APIs for payment gateways and verification services will remain operational and secure.
- Regular maintenance and updates will be performed to ensure system reliability and security.

2.3. Functional Requirements

2.3.1 User Registration

- Users can register using an email ID and phone number.
- System sends an OTP for verification.

- Registration process includes password setup, security question selection, and user agreement acceptance.

2.3.2 User Authentication

- Login via username/password and 2FA.
- Remember me option for trusted devices.
- Account lockout after multiple failed login attempts.
- Option for biometric login (e.g., fingerprint or facial recognition).

2.3.3 Product Catalog

- Display various banking services like loans, deposits, and insurance.
- Dynamic updates based on user preferences and eligibility criteria.

2.3.4 Shopping Cart

- Allows users to select and compare banking products.
- Save and revisit options for later.

2.3.5 Order Processing and Payment

- Securely handle payments for selected products or services.
- Provide multiple payment methods such as credit cards, UPI, and net banking.

2.3.6 Admin Panel

- Manage user accounts and oversee transactions.

- Generate reports, monitor logs, and analyze system usage.
- Control access permissions and manage user roles.

2.4. Non-Functional Requirements

2.4.1 Security Requirements

- SSL encryption for data transmission.
- Regular vulnerability assessments and penetration testing.
- Compliance with GDPR, PCI DSS, and other regional security standards.

2.4.2 Performance Requirements

- Handle 1000 concurrent users with less than 2-second response time.
- Optimized for high-speed internet as well as moderate-speed connections.

2.4.3 Availability Requirements

- 99.9% uptime with automated failover mechanisms.
- Real-time monitoring and alert systems for system administrators.

2.4.4 Maintainability Requirements

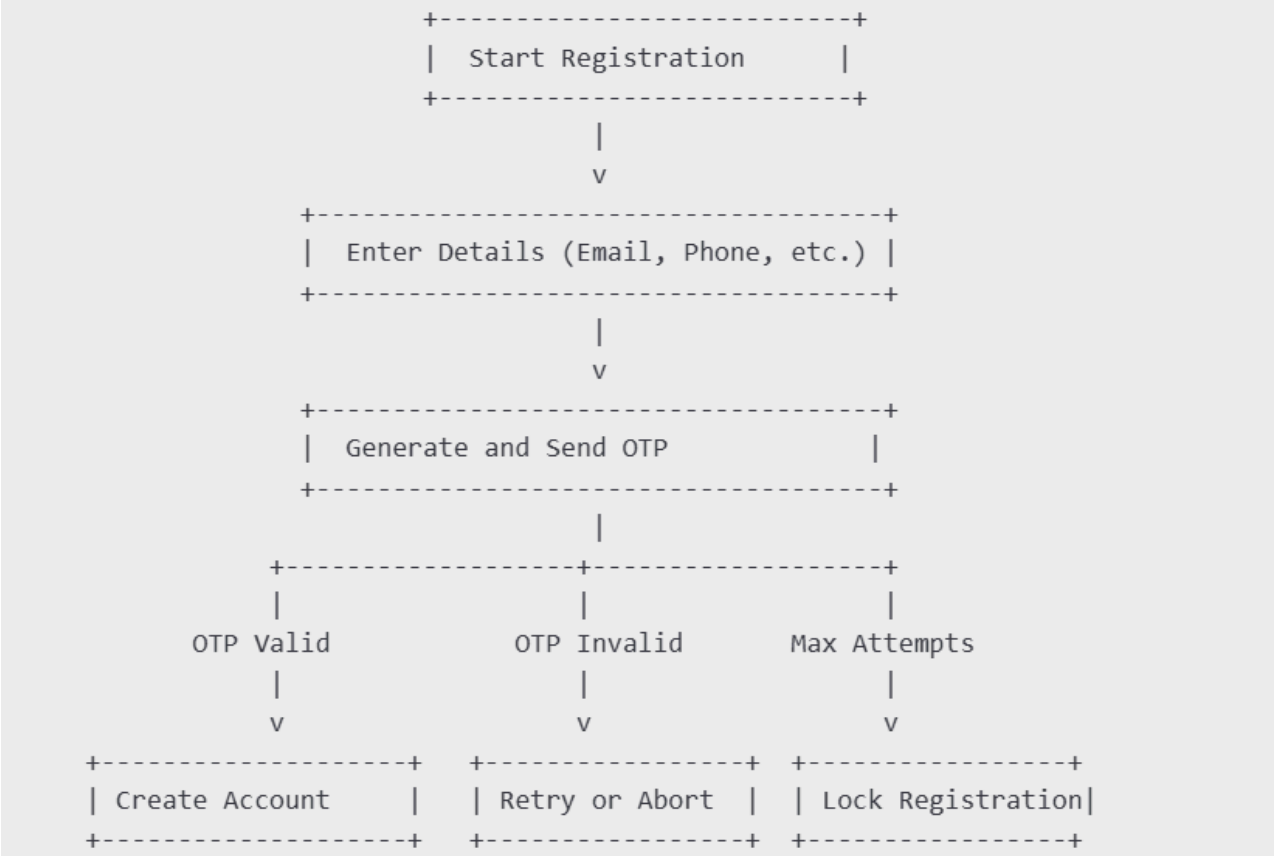
- Modular design to support updates and scalability.
- Detailed documentation for developers and administrators.

2.4.5 Usability Requirements

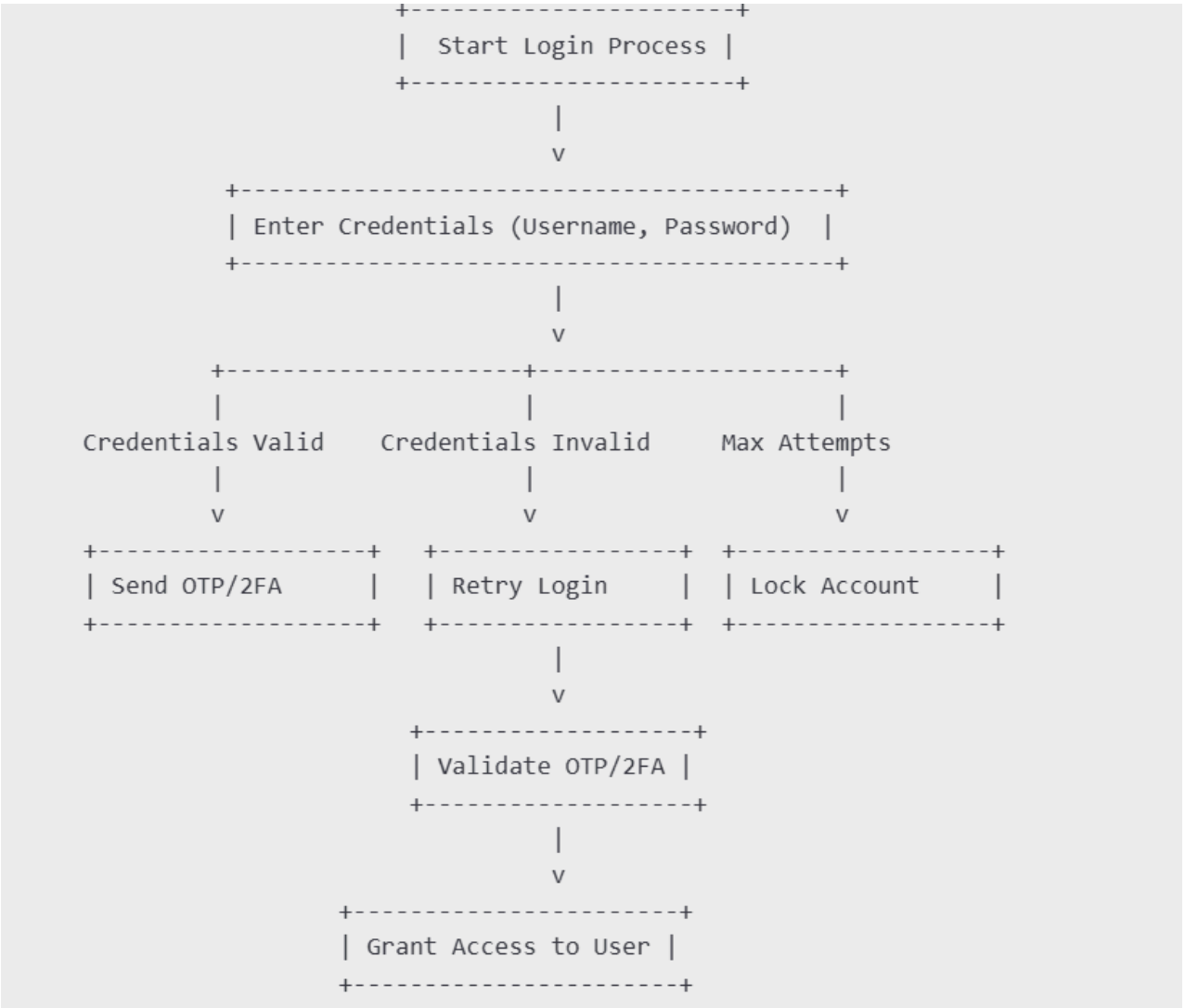
- Simple and intuitive UI for all user classes.
- Accessibility features such as screen readers and high-contrast modes.

5. Decision Tree

For the user registration:-



For the user authentication:-



6. Decision Table

Condition	Payment method available?	Cart Not Empty?	Payment Succesful?	Order Status	Action
User selects a product	Yes	Yes	Yes	Order Confirmed	Process Payment, Uodate Inventory
User selects a product	Yes	Yes	NO	Payment Failed	Show Error: “Payment

					Failed”
User proceeds without products	N/A	No	N/A	Cart Empty	Show Error: “Cart is Empty”
Payment method unavailable	No	Yes	N/A	Payment Unavailable	Show Error: "Payment Not Available"

2.7. External Interface Requirements

2.7.1 User Interfaces

Web interface with responsive design for various screen sizes.

Mobile app for Android and iOS with offline notification support.

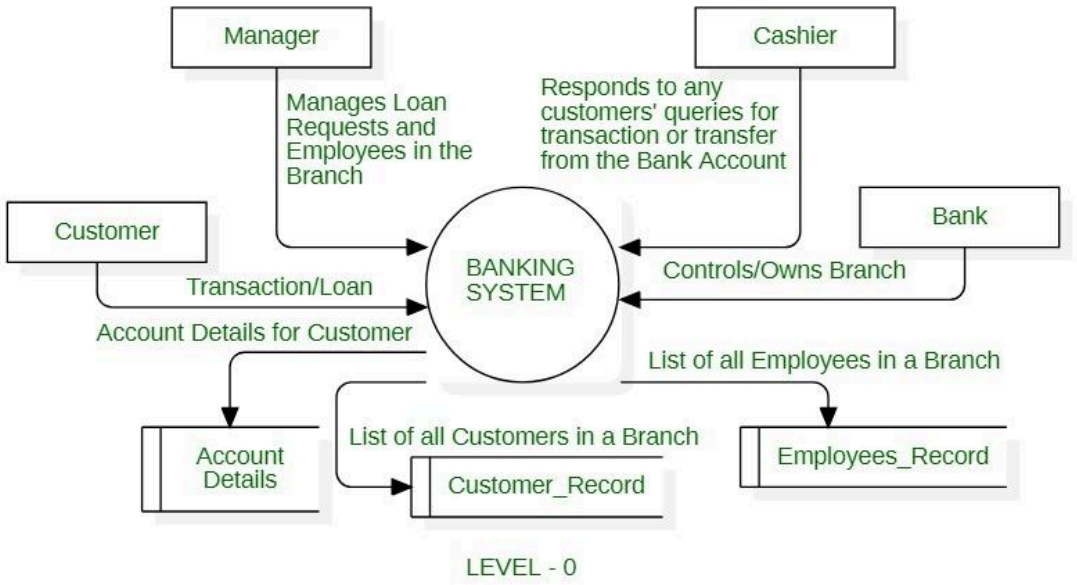
2.7.2 Hardware Interfaces

Supports standard desktops, laptops, and mobile devices with compatible operating systems.Integration with biometric hardware such as fingerprint scanners.

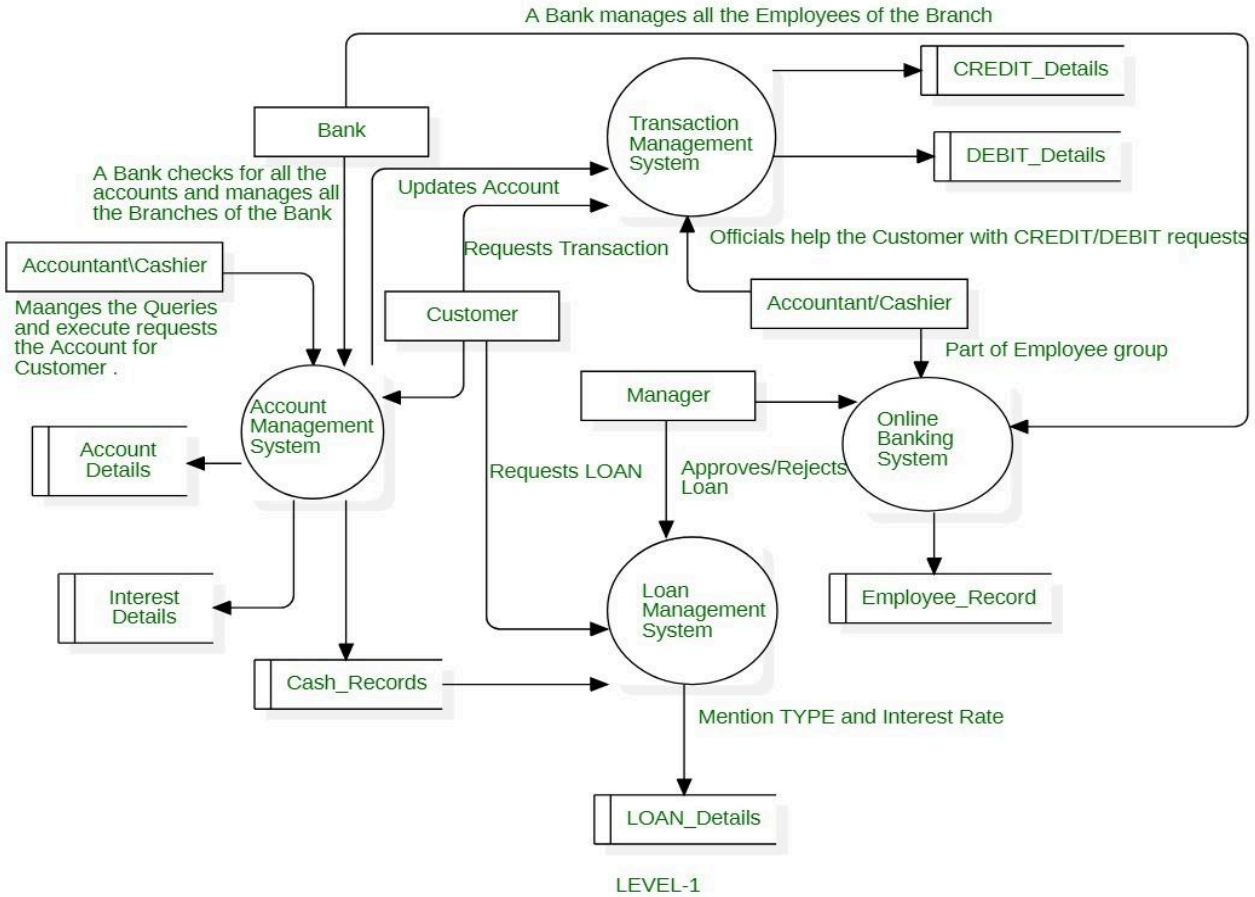
2.7.3 Software Interfaces

Integration with third-party APIs for payments and identity verification.Database servers for storing user data securely.Compatibility with cloud services for enhanced scalability and backup solutions.

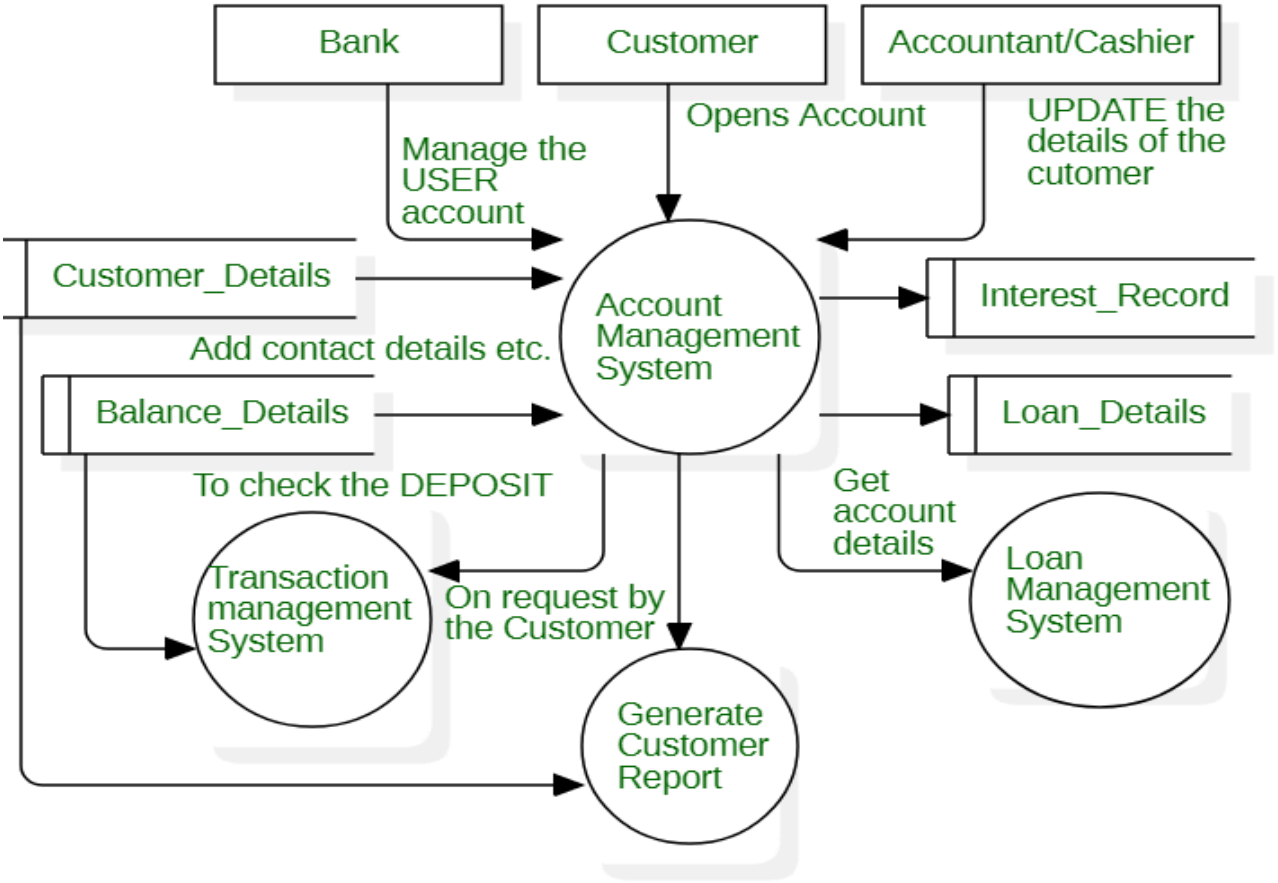
3.1 DFD Diagrams with description



LEVEL - 0



LEVEL-1

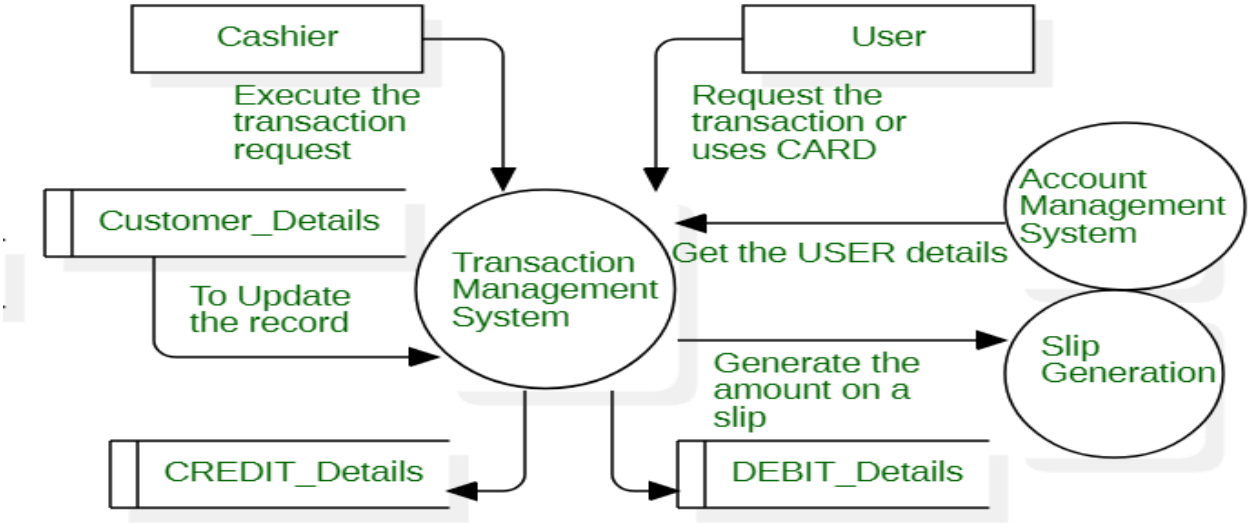


1. Account Management System

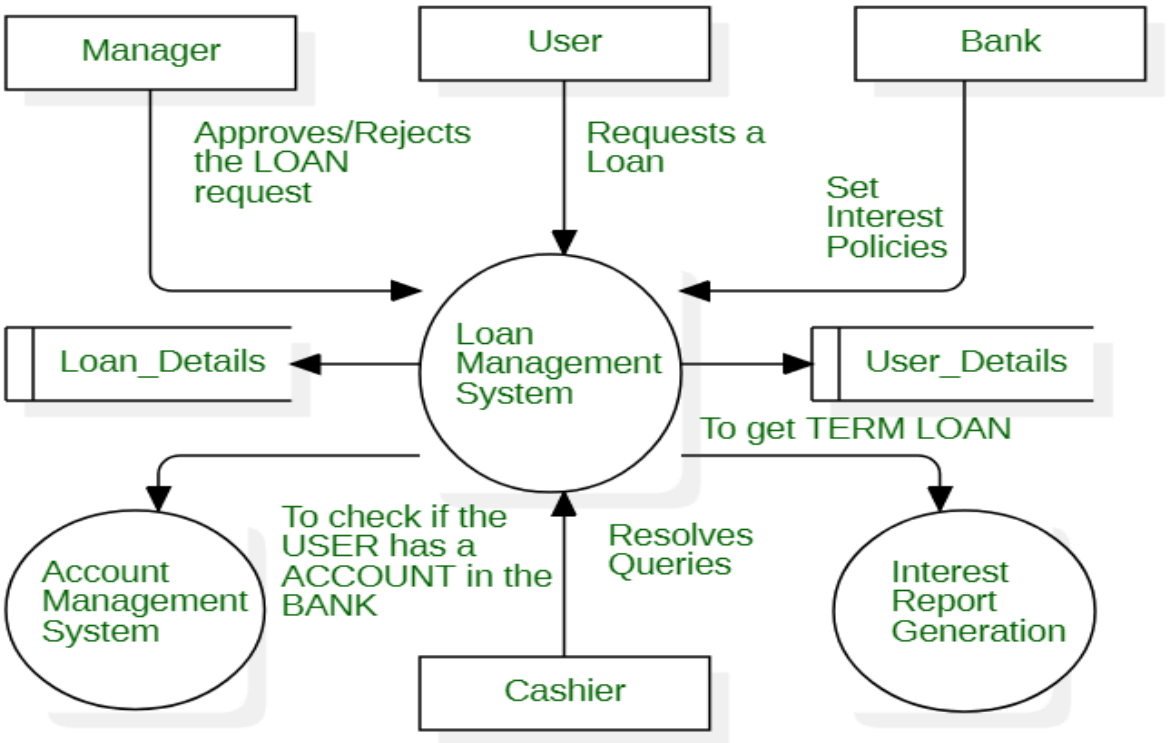
Level - 2

Account Management System –

In this Customer can access all the services offered by the Bank by adding his details. As the Customer avails any services as a transaction or a loan, then the required data flows to Transaction Management System or the Loan Management System respectively. Any Customer can print the activity status of the account which fetches information from all the available databases using the Generate Customer Report System



2. Transaction Management System

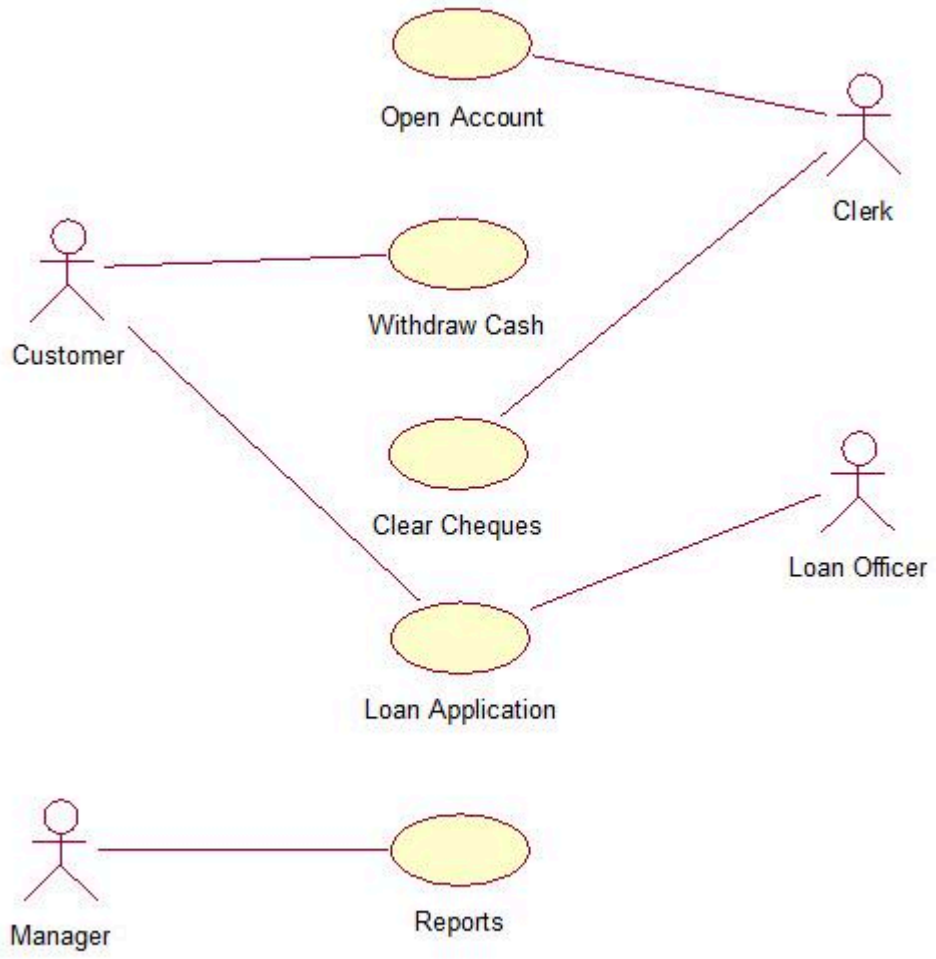


3. Loan Management System

Level - 2

4. UML Diagrams with description

4.1. Use-Case Diagram:



Actors:

- Customer: Represents the end-user who interacts with the system.
- Clerk: Represents a bank employee who handles routine transactions.

- Loan Officer: Represents a bank employee who processes loan applications.
- Manager: Represents a bank manager who oversees operations and generates reports.

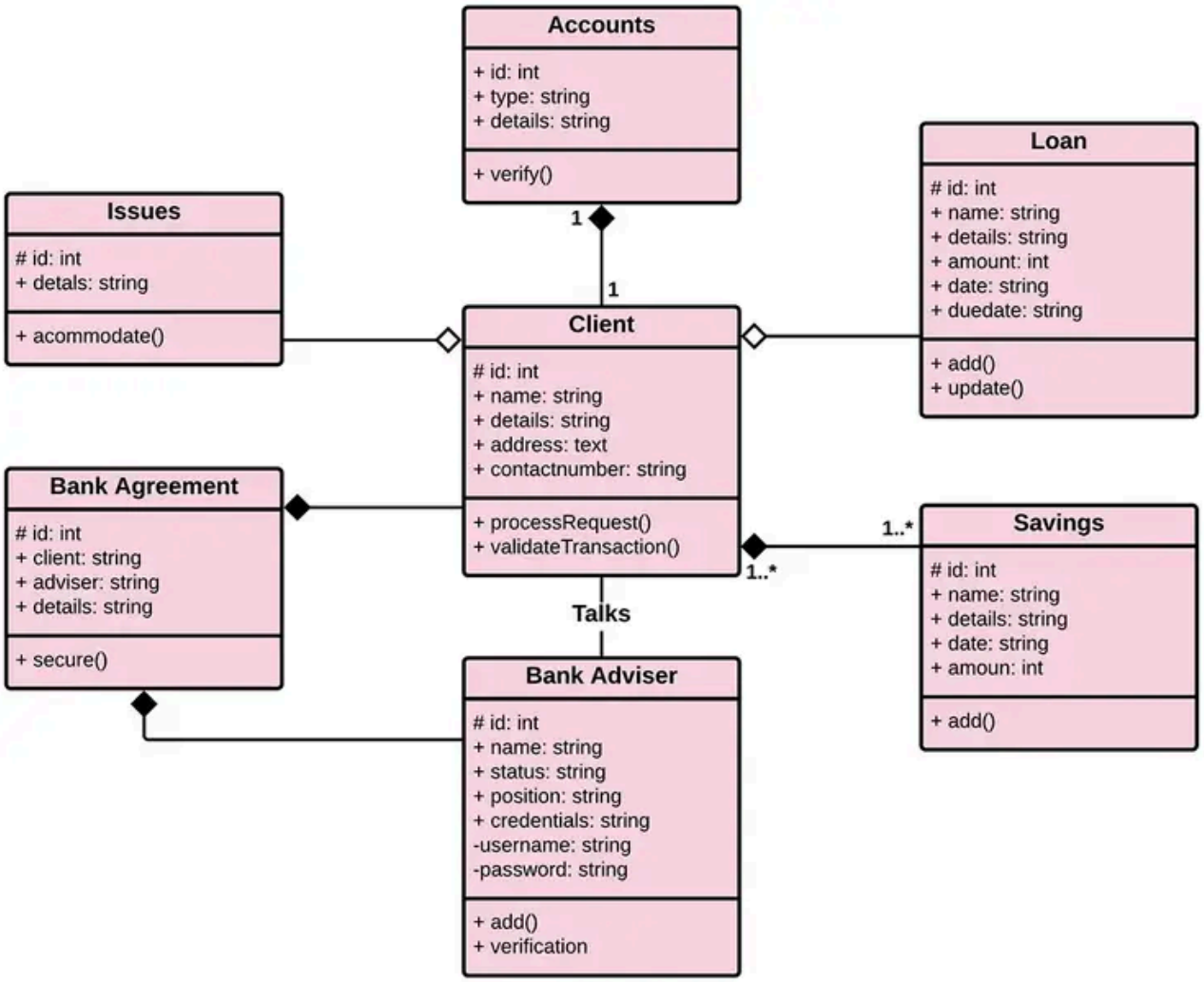
Use Cases:

- Open Account: The process of creating a new bank account.
- Withdraw Cash: The process of withdrawing cash from an account.
- Clear Cheques: The process of clearing cheques.
- Loan Application: The process of applying for a loan.
- Reports: The process of generating various reports, such as financial reports or customer reports.

Relationships:

- The diagram shows how different actors interact with the system to perform specific tasks. For example, a customer can withdraw cash or apply for a loan, while a clerk can open an account or clear cheques.
- The relationships between actors and use cases are represented by lines connecting them.

4.2. Class Diagram:



Classes:

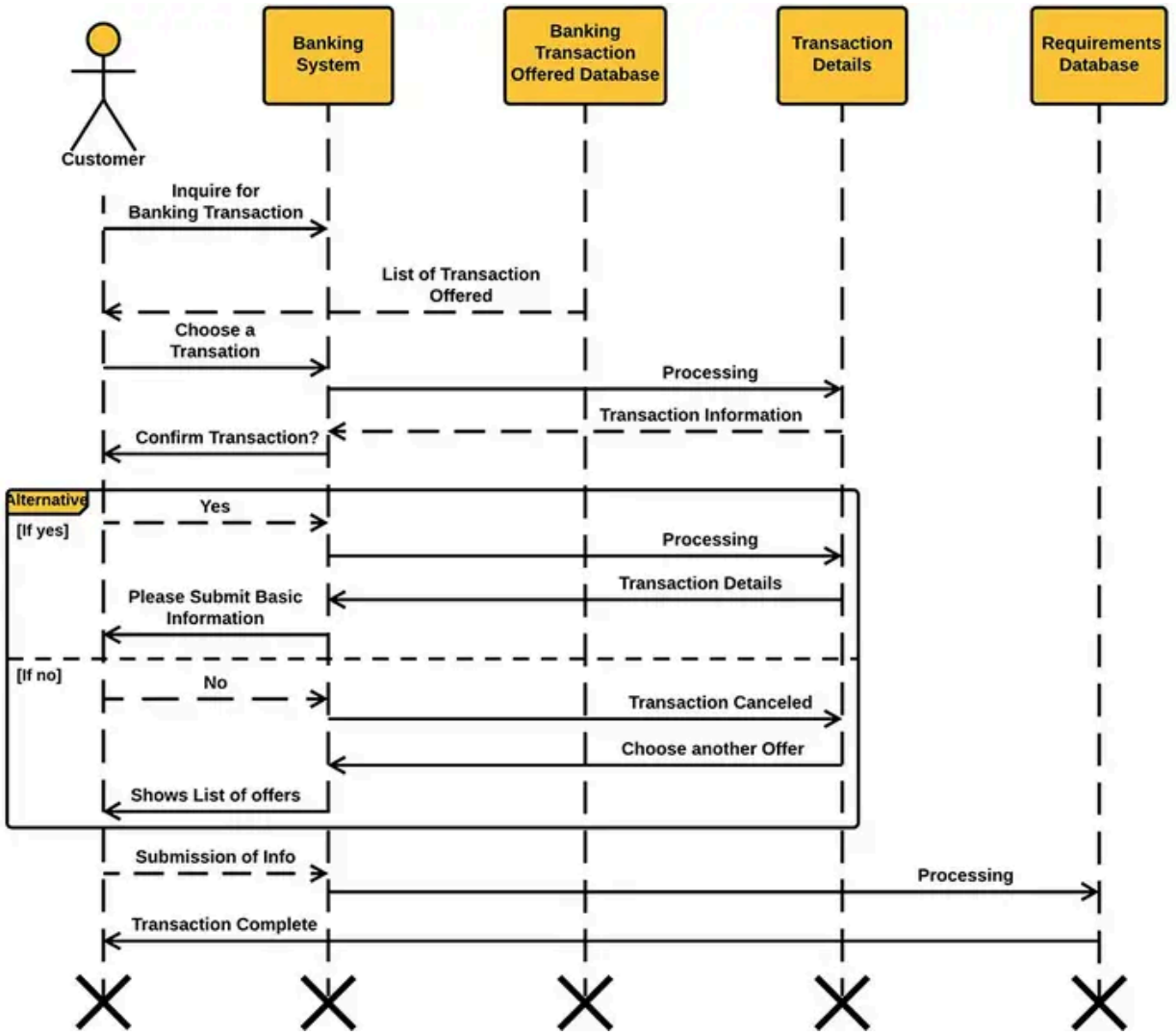
- **Accounts:** Represents bank accounts with attributes like ID, type, and details. It has a method `verify()` to verify account details.
- **Loan:** Represents loans with attributes like ID, name, details, amount, date, and due date. It has methods `add()` and `update()` to manage loan information.
- **Client:** Represents bank clients with attributes like ID, name, details, address, contact number, and methods to process requests and validate transactions.

- **Bank Agreement:** Represents bank agreements with attributes like ID, client, advisor, details, and methods to secure the agreement.
- **Savings:** Represents savings accounts with attributes like ID, name, details, date, and amount. It has a method `add()` to add savings.
- **Bank Adviser:** Represents bank advisors with attributes like ID, name, status, position, credentials, username, password, and methods to add and verify.

Relationships:

- **Association:** A line connecting two classes, indicating a relationship between them. For example, a Client can have one or more Accounts and Loans.
- **Aggregation:** A diamond-shaped line connecting two classes, indicating a "has-a" relationship. For example, a Bank Agreement has a Client.
- **Inheritance:** A triangle-shaped line connecting two classes, indicating an "is-a" relationship. For example, Savings is a type of Account.

4.3. Sequence Diagram:



Actors:

- Customer: Represents the end-user who interacts with the system.

Objects:

- Banking System: Represents the core system that handles transactions.
- Banking Transaction Offered Database: Represents the database storing information about available transactions.

- Transaction Details: Represents the specific details of a transaction.
- Requirements Database: Represents the database storing requirements for different transactions.

Interactions:

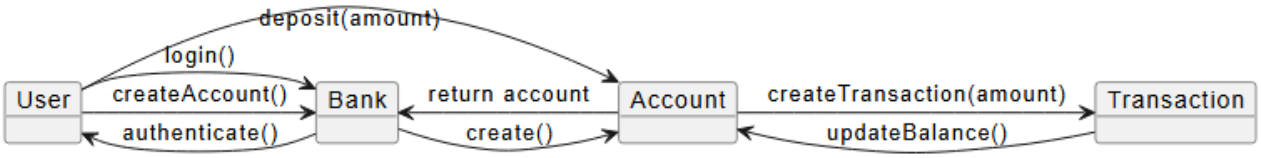
1. Customer Initiates Transaction: The customer inquires about banking transactions.
2. System Presents Options: The system displays a list of available transactions.
3. Customer Chooses Transaction: The customer selects a specific transaction.
4. System Confirms: The system asks the customer to confirm the transaction.
5. Customer Confirms or Cancels: The customer either confirms or cancels the transaction.
    - If confirmed: The customer is prompted to provide basic information.
    - If canceled: The system displays a list of other available transactions.
6. System Processes Transaction: The system processes the transaction based on the provided information.
7. Transaction Completed: The system informs the customer that the transaction is complete.

Key Points:

- The diagram shows the sequential flow of messages between the customer and the system.
- The dashed lines represent asynchronous messages, while solid lines represent synchronous messages.

- The diagram uses a combination of vertical lines and arrows to show the sequence of interactions.
- The alternative flow for transaction cancellation is represented using a dashed box.

4.4. Collaboration Diagram:

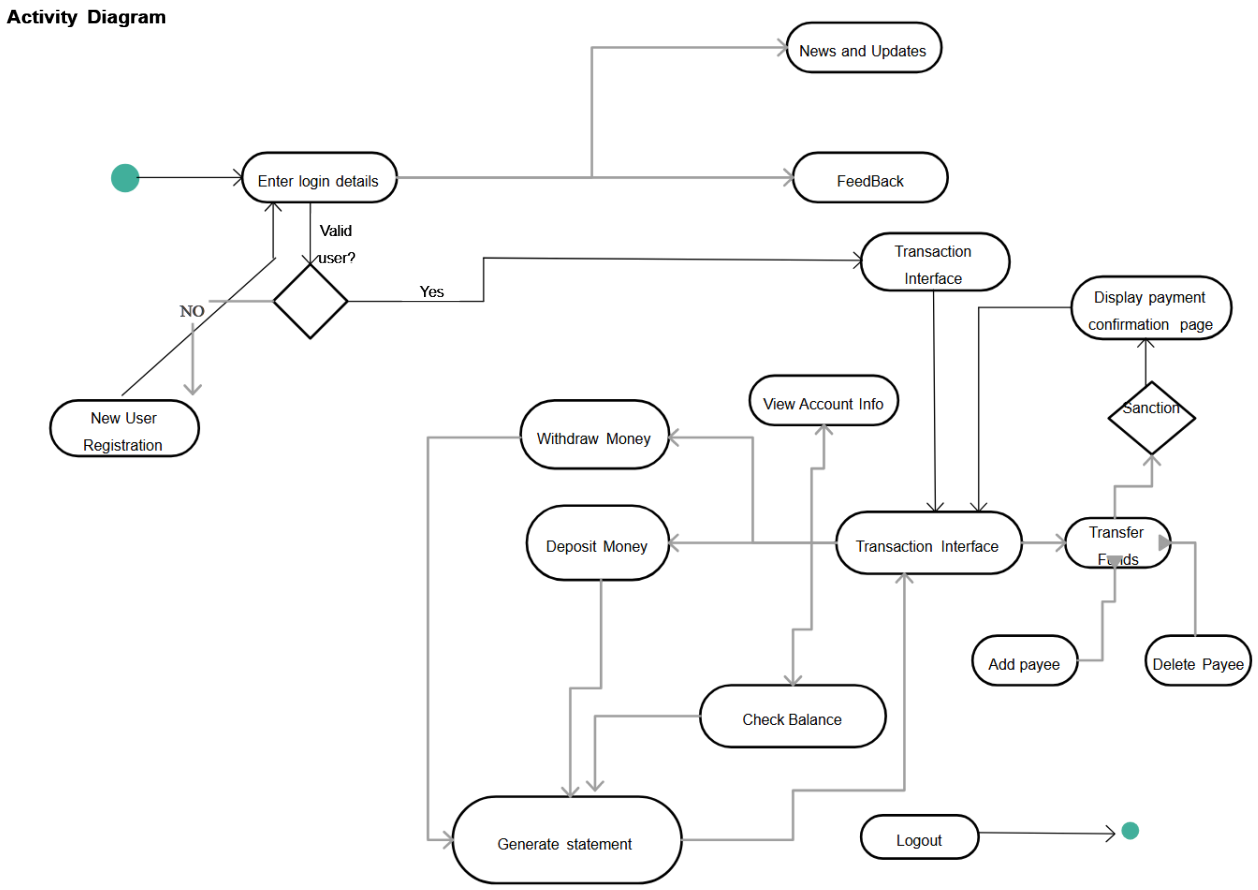


The key elements and their relationships are as follows:

- **User:**
  - Performs the login() operation to authenticate.
  - Calls the createAccount() operation to create a new account.
- **Bank:**
  - Receives the login() request from the User and performs the authenticate() operation.
  - Creates a new account when the User calls the createAccount() operation.
  - Returns the account information to the User.
- **Account:**
  - Is created by the Bank when the User calls the createAccount() operation.
  - Allows the User to perform createTransaction(amount) operations to update the account balance.
  - Provides the updateBalance() operation to update the account balance.

- **Transaction:**
  - Is created by the User through the createTransaction(amount) operation on the Account.

4.5. <u>Activity Diagram:</u>



Activities:

- **Enter login details:** The initial step where the user enters their credentials.
- **Valid User?** A decision point to check if the entered credentials are valid.

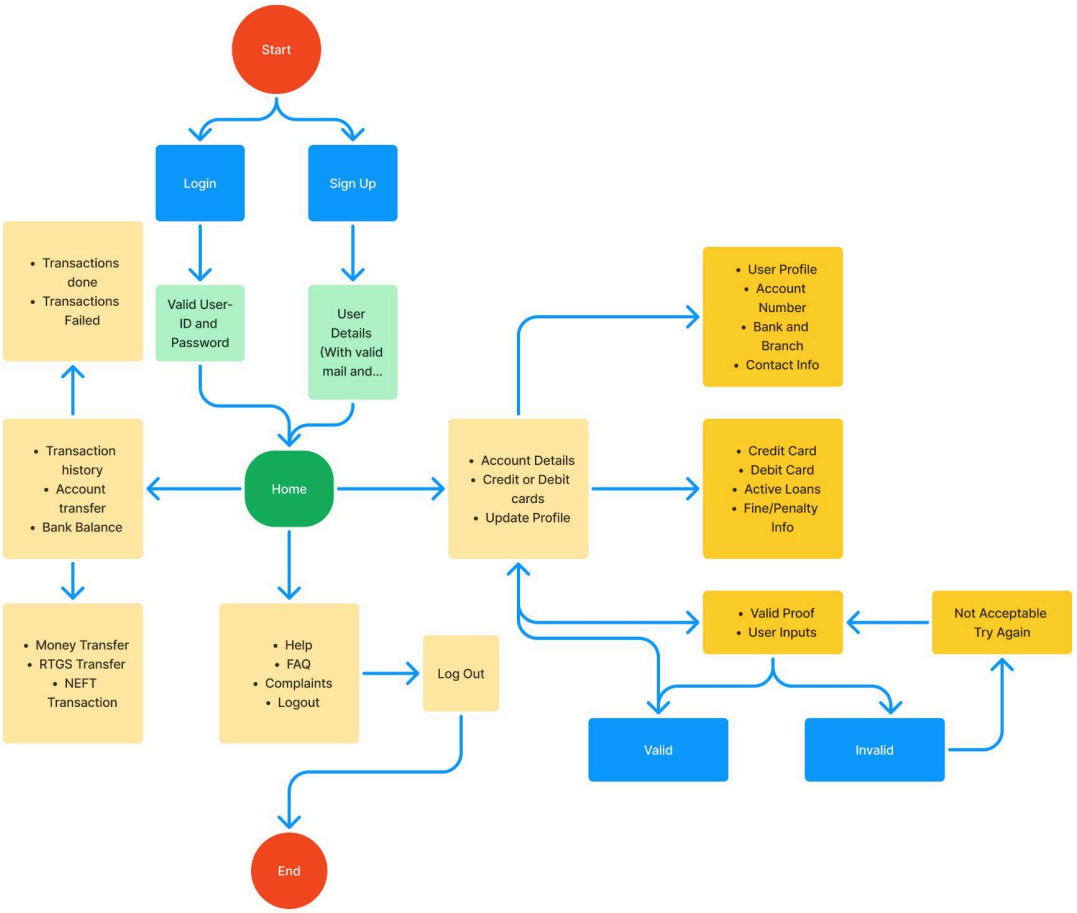
- **New User Registration:** If the user is new, they are directed to the registration process.
- **Transaction Interface:** The main interface for various banking transactions.
- **Withdraw Money:** The user can withdraw money from their account.
- **Deposit Money:** The user can deposit money into their account.
- **Transfer Funds:** The user can transfer funds between accounts.
- **Check Balance:** The user can view their account balance.
- **Generate Statement:** The user can generate a statement of their transactions.
- **View Account Info:** The user can view their account information.
- **Add Payee:** The user can add a new payee for future transactions.
- **Delete Payee:** The user can delete an existing payee.
- **Sanction:** The system may perform a security check or sanction a transaction.
- **Display Payment Confirmation Page:** After a successful transaction, the system displays a confirmation page.
- **Logout:** The user logs out of the system.
- **News and Updates:** The system may display news and updates to the user.
- **Feedback:** The user can provide feedback on the system.

Control Flow:

- **Arrows:** The arrows indicate the flow of control between activities.
- **Decision Nodes:** Diamond-shaped nodes represent decision points where the flow can branch based on a condition (e.g., "Valid User?").
- **Initial and Final Nodes:** The rounded rectangles at the beginning and end of the diagram represent the start and end points of the flow.

5 Front-End Design

5.1 Flow Diagram:



5.2 Figma link :

https://www.figma.com/design/yQfn4JZzB53Srfl6rT50sJ/Online-Banking-System?node-id=0-1&p=f&t=xgKWdaEKv4PxTcuA-0

6. Database Design

Database design serves as the backbone of an Online Banking System, ensuring smooth functionality, secure data handling, and efficient operations. A robust database design is critical for managing sensitive customer and transaction data, ensuring compliance with industry standards, and delivering a seamless banking experience to users.

6.1 Objectives of Database Design

The database design for the Online Banking System aims to achieve the following objectives:

1. **Data Accuracy**: Ensure that customer and transaction records are stored correctly and consistently.
2. **Performance**: Support fast data retrieval and updates, even under high transaction loads.
3. **Scalability**: Provide flexibility for future enhancements and increased user volumes.
4. **Security**: Protect sensitive financial data through encryption and access control.
5. **Reliability**: Maintain data integrity and prevent loss or corruption during operations.

6.2 Key Entities and Attributes

The database design includes the following primary entities:

1. Customer

Represents the users of the banking system.

- **Attributes:**
  - `CustomerID` (Primary Key): Unique identifier for each customer.
  - `FirstName` and `LastName`: Customer's full name.
  - `Email`: Customer's email address.
  - `PhoneNumber`: Contact number.
  - `Address`: Residential address.
  - `Username`: Unique username for login.
  - `Password`: Encrypted password.

2. Account

Tracks financial accounts belonging to customers.

- **Attributes:**
  - `AccountID` (Primary Key): Unique identifier for each account.
  - `CustomerID` (Foreign Key): Links to the `Customer` table.
  - `AccountType`: Type of account (e.g., Savings, Current).
  - `Balance`: Current account balance.
  - `BranchCode`: Identifies the branch managing the account.

3. Transaction

Stores details of all financial transactions.

- **Attributes:**
  - `TransactionID` (Primary Key): Unique identifier for each transaction.
  - `AccountID` (Foreign Key): Links to the `Account` table.
  - `TransactionType`: Type of transaction (Deposit, Withdrawal, Transfer).
  - `Amount`: Transaction amount.
  - `Date`: Transaction date.
  - `Time`: Transaction time.
  - `Remarks`: Additional details about the transaction.

4. Loan

Represents loan-related information.

- **Attributes:**
  - `LoanID` (Primary Key): Unique identifier for each loan.
  - `CustomerID` (Foreign Key): Links to the `Customer` table.
  - `LoanType`: Type of loan (e.g., Personal, Home).
  - `LoanAmount`: Total loan amount.
  - `InterestRate`: Interest rate applied.
  - `RepaymentSchedule`: Details of repayments.

5. ServiceRequest

Handles customer service-related requests.

- **Attributes:**

- `RequestID` (Primary Key): Unique identifier for each request.
- `CustomerID` (Foreign Key): Links to the `Customer` table.
- `RequestType`: Type of service requested.
- `Status`: Current status of the request (Pending, Completed).
- `RequestDate`: Date of request submission.

6. LoginSecurity

Manages secure login sessions and tracks login activity.

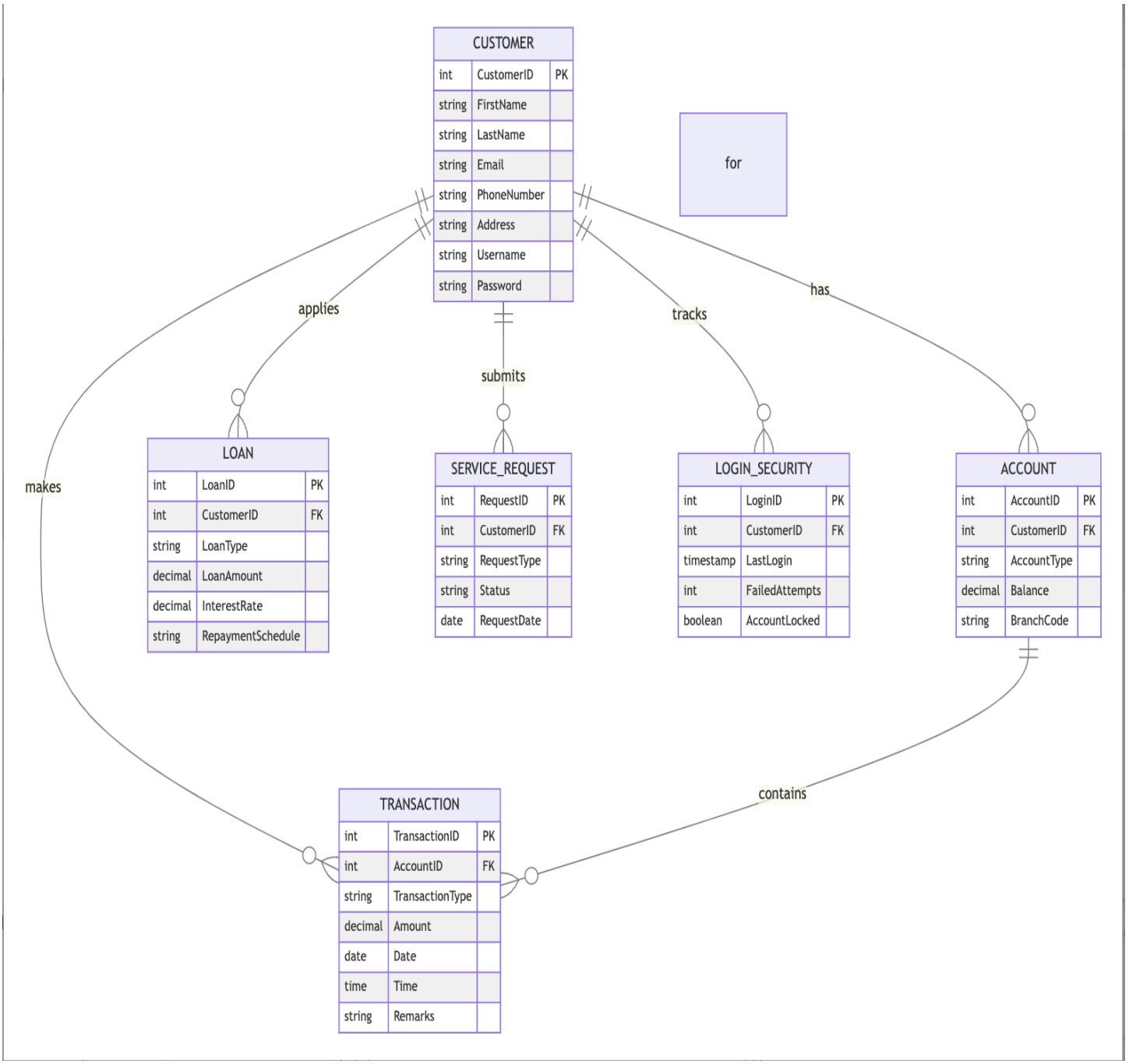
- **Attributes:**
  - `LoginID` (Primary Key): Unique identifier for each login entry.
  - `CustomerID` (Foreign Key): Links to the `Customer` table.
  - `LastLogin`: Timestamp of the last successful login.
  - `FailedAttempts`: Tracks unsuccessful login attempts.
  - `AccountLocked`: Indicates whether the account is locked.

6.3 ER Diagram

The ER Diagram visually represents the relationships between the primary entities of the Online Banking System. It includes:

1. **Relationships** between `Customer`, `Account`, and `Transaction` entities.

2. **Foreign Keys** connecting tables to maintain referential integrity.



6.4 SQL Schema with Example Data

Customer Table

Structure:

```
CREATE TABLE Customers (
    CustomerID INT AUTO_INCREMENT PRIMARY KEY,
    FirstName VARCHAR(50),
    LastName VARCHAR(50),
    Email VARCHAR(100) UNIQUE,
    PhoneNumber VARCHAR(15) UNIQUE,
    Address TEXT,
    Username VARCHAR(50) UNIQUE,
    Password VARCHAR(255)
);
```

Sample Data:

```
INSERT INTO Customers (FirstName, LastName, Email,
PhoneNumber, Address, Username, Password)
VALUES ('John', 'Doe', 'john.doe@example.com', '9876543210', '123
Elm Street', 'johndoe', 'hashed_password');
```

CustomerID	FirstName	LastName	Email	PhoneNumber	Address	Username	Password
1	John	Doe	john.doe@example.com	9876543210	123 Elm Street	johndoe	hashed_password

Account Table

Structure:

```
CREATE TABLE Accounts (
    AccountID INT AUTO_INCREMENT PRIMARY KEY,
    CustomerID INT NOT NULL,
    AccountType ENUM('Savings', 'Current', 'Fixed Deposit'),
    Balance DECIMAL(15,2) NOT NULL,
```

```
    BranchCode VARCHAR(10),
    FOREIGN KEY (CustomerID) REFERENCES
Customers(CustomerID)
);
```

Sample Data:

```
INSERT INTO Accounts (CustomerID, AccountType, Balance,
BranchCode)
VALUES (1, 'Savings', 15000.00, 'BR123');
```

Output:

AccountID	CustomerID	AccountType	Balance	BranchCode
1	1	Savings	15000.00	BR123

Transaction Table

Structure:

```
CREATE TABLE Transactions (
    TransactionID INT AUTO_INCREMENT PRIMARY KEY,
    AccountID INT NOT NULL,
    TransactionType ENUM('Deposit', 'Withdrawal', 'Transfer'),
    Amount DECIMAL(15,2) NOT NULL,
    Date DATE NOT NULL,
    Time TIME NOT NULL,
    Remarks TEXT,
```

```
    FOREIGN KEY (AccountID) REFERENCES
Accounts(AccountID)
);
```

Sample Data:

```
INSERT INTO Transactions (AccountID, TransactionType, Amount,
Date, Time, Remarks)
VALUES (1, 'Deposit', 5000.00, '2024-12-19', '10:30:00', 'Salary
deposit');
```

Output:

TransactionID	AccountID	TransactionType	Amount	Date	Time	Remarks
1	1	Deposit	5000.00	2024-12-19	10:30:00	Salary deposit

6.5 Conclusion

The database design of the Online Banking System is a comprehensive and secure framework that ensures efficient data management and supports critical banking operations. By adhering to best practices in normalization, indexing, and access control, the design guarantees optimal performance and scalability. The inclusion of key entities such as `Customer`, `Account`, and `Transaction` ensures all essential banking processes are covered, while relationships between these entities facilitate seamless integration.

Security remains a top priority, with features like encrypted passwords and login tracking safeguarding customer data. Additionally, the design's modular structure allows for easy expansion and adaptability to future banking requirements, such as mobile banking integration or advanced analytics.

In conclusion, this database design not only meets the current functional requirements of an Online Banking System but also provides a robust foundation for long-term growth and innovation.

7. Test Document

7.1 Objective

Explain the purpose of testing the Online Banking System, such as ensuring functionality, security, and reliability. For example:

- To validate that the system performs all operations (login, transactions, etc.) correctly.
- To ensure secure handling of sensitive information like account details.

7.2 Test Plan

Describe how testing is planned, including the following:

- **Testing Types**: Functional, Security, Usability, and Performance testing.
- **Test Environment**: Mention the system environment (e.g., Windows 10, macOS, browsers like Chrome or Firefox).
- **Tools Used**: Examples include Selenium for automated testing and JMeter for performance testing.

7.3 Test Cases

Provide a table or list of test cases. Each test case should include:

- **Test Case ID**: Unique identifier (e.g., TC001).
- **Test Case Description**: What is being tested (e.g., "Verify login functionality").

- **Steps to Execute**: Actions the tester must perform.

Test Case ID	Test Case	Description	Steps	Expected Result	Status
TC001	User Login	Verify login with valid credentials.	1. Open the login page. 2. Enter valid username and password. 3. Click on "Login".	User should be successfully logged in.	Pass/Fail
TC002	Invalid Login Attempt	Check error handling for invalid inputs.	1. Enter incorrect username or password. 2. Click on "Login".	Error message "Invalid credentials" should appear.	Pass/Fail

TC003	Fund Transfer	Verify fund transfer functionality.	1. Login. 2. Navigate to "Transfer Funds". 3. Enter recipient details and amount. 4. Confirm transfer.	Transaction completes successfully with a message.	Pass/Fail
TC004	View Account Balance	Verify account balance display.	1. Login. 2. Navigate to "Account Balance".	Correct account balance is displayed.	Pass/Fail
TC005	Logout Functionality	Verify system logout works properly.	1. Login. 2. Click on "Logout".	User is redirected to the login page.	Pass/Fail

- **Expected Result**: The desired outcome (e.g., "User successfully log’s in)

7.4 Test Summary

Summarize the testing outcomes, including:

- Total test cases executed.
- Number of passed/failed test cases.

- Observations or issues identified.
  For example:
- **Total Test Cases**: 15
- **Passed**: 14
- **Failed**: 1
- **Pending**: 0

7.5 Conclusion

Conclude by stating whether the Online Banking System is ready for deployment or needs further improvements.
For example:

- "The system has passed most of the test cases successfully, indicating readiness for deployment. The identified issue will be resolved in the next iteration."

Contributions

Akarsh Jain : Objective and SRS Document

Tarandeep Singh Juneja : Database Design

Pravin Kumar : Frontend Design

Shreyansh Soni : Test Document

Nihira Pandey : UML Diagrams and description

Dikshant Shubhankar : Data Flow Diagrams(DFD)