Bank Management System Documentation

Version 3.0

May 26th, 2025

Table of Contents

Executive Summary

Project Overview

This project involves the development of a comprehensive bank management software designed to enhance the efficiency of banking operations and improve customer experience. The product serves a wide range of users, including customers, bank staff, and administrators.

The software is intended to support critical banking processes such as account management, transaction handling and loan approvals. By integrating with existing infrastructure and providing robust security measures, the system seeks to deliver reliable banking services.

Product/Service Description

Product Context

This bank management software integrates seamlessly with a variety of related systems, forming part of a broader ecosystem of financial products and services.

Relationships to Other Systems:

1. ATMs: Facilitate real-time balance inquiries and cash withdrawals.
2. Payment Processors: Facilitate debit/credit card transactions at POS (Points-of-Sale)
3. Wire Transfer Network System: Facilitate secure transfers to external bank accounts.

User Characteristics

Internal Users:

- ***System Administrators***
  - Responsible for managing employee profiles and role assignments
  - Moderate technical expertise
  - Require access control
- ***Bank Tellers, Loan Officers***
  - Responsible for handling customer requests such as account creation, transactions, and loan approvals
  - Low technical expertise
  - Need for intuitive UI and quick access to transactions
- ***Bank Managers, Fraud Analysts, Auditors***
  - Responsible for reviewing customer, fraud detection, and audit analytics.
  - Low to moderate technical expertise
  - Require dashboards, audit logs, and exportable reports

External Users:

- ***Customers***
  - Use online banking to manage personal or business accounts, perform transactions, apply for loans, and access customer services
  - Basic computer skills
  - Need for intuitive UI

Assumptions

- Customers and employees will have stable internet access to access the system.
- Basic familiarity with banking operations is assumed for all users.
- The bank will keep the system updated with evolving regulatory requirements.
- The bank will provide secure hosting, firewalls, and encryption measures.
- A secure, centralized database is available.
- Data backup systems are reliable to ensure continuity during outages.

Constraints and Dependencies

- The system must operate alongside existing legacy banking systems until full migration is complete.
- **User Management** module must be completed before access to any other banking functionalities.
- **Funds Management**, **Audit & Reporting**, and **Fraud Detection** modules must be implemented before account-related operations can be processed.

Requirements

Functional Requirements

Req#	Requirement	Business Process	Priority
BR_UM_01	System administrator shall be able to maintain system profiles for bank employees.	User Management	1
BR_UM_02	Bank employees / customers shall be able to log into the system.	User Management	1
BR_UM_03	Customers shall be able to create an online account.	User Management	2
BR_FM_01	Bank managers shall be able to view funds information about their assigned financial units (branches, ATMs).	Funds Management	1
BR_FM_02	Bank managers shall be able to allocate funds to their assigned financial units.	Funds Management	1
BR_AM_01	Bank tellers shall be able to open a new bank account for a customer.	Account Management	1
BR_AM_02	Bank tellers shall be able to close an existing bank account upon customer request.	Account Management	1
BR_AM_03	Bank tellers shall be able to modify account details.	Account Management	1
BR_AM_04	Customers shall be able to access their bank account details and transaction history.	Account Management	2
BR_TM_01	Customers shall be able to deposit money into their accounts.	Transaction Management	1
BR_TM_02	Customers shall be able to withdraw money.	Transaction Management	1
BR_TM_02A	Customers shall be able to withdraw money at a bank branch.	Transaction Management	1
BR_TM_02B	Customers shall be able to withdraw money at an ATM.	Transaction Management	2
BR_TM_03	Customers shall be able to make transfers.	Transaction Management	1
BR_TM_03A	Customers shall be able to make	Transaction Management	1

Functional Requirements

Req#	Requirement	Business Process	Priority
	transfers to accounts within the same bank.		
BR_TM_03B	Customers shall be able to make transfers to external bank accounts via wire transfer.	Transaction Management	1
BR_TM_03C	Customers shall be able to set up automated recurring transfers.	Transaction Management	1
BR_TM_04	Customers shall be able to make card payments.	Transaction Management	1
BR_LM_01	Customers shall be able to apply for and be granted loans.	Loan Management	1
BR_LM_02	Customers shall be able to repay their loans.	Loan Management	1
BR_CM_01	Customers shall be able to apply for cards for their bank account online.	Card Management	1
BR_CM_02	Customers shall be able to request checkbooks for their checking accounts.	Card Management	1
BR_CM_03	Customers shall be able to set spending limits on their cards.	Card Management	2
BR_FD_01	Fraud analysts shall be able to review and take action on suspicious banking activities.	Fraud Detection	1
BR_FD_02	Customers shall be able to report fraudulent activities related to their account.	Fraud Detection	1
BR_CS_01	Customers shall be able to access real-time information about ATMs and branches.	Customer Services	3
BR_CS_02	Customers shall be able to schedule in-branch appointments online.	Customer Services	3
BR_CS_03	Customers shall be able to submit feedback and complaints about their banking experience online.	Customer Services	3
BR_CS_04	Customers shall be able to request official account-related documents.	Customer Services	1

Functional Requirements

Req#	Requirement	Business Process	Priority
BR_AR_01	Auditors shall be able to view audit trails to ensure compliance with banking regulations.	Audit & Reporting	1
BR_AR_02	Bank managers shall be able to access customer analytics.	Audit & Reporting	2
BR_AR_03	Bank managers shall be able to access fraud detection analytics.	Audit & Reporting	2

Non-Functional Requirements

1. Usability Requirements

1.1. The system shall have a user-friendly interface.
1.2. The system shall support multilingual interfaces (Albanian, English).
1.3. The user documentation and help shall be comprehensive and easy to understand.

2. Performance Requirements

2.1. The system shall support at least 50,000 active online banking users at any given time.
2.2. The system shall support up to 300 concurrent bank employees.
2.3. 95% of transactions shall be processed within 1 second.
2.4. 99.9% of transactions shall be processed within 3 seconds, even under peak load.
2.5. The system shall sustain peak loads of 10 times the normal transaction volume without downtime or performance degradation.

3. Availability Requirements

3.1. Critical banking operations (e.g., transactions, account management) shall be available 24/7, v during scheduled maintenance windows.
3.2. Mobile and online banking services must be available to customers in all regions with internet access.
3.3. Downtime exceeding 15 minutes during business hours shall trigger an automatic failover to a backup server.
3.4. Scheduled maintenance shall be conducted between 12 AM - 4 AM to minimize user impact.
3.5. The maximum number of failures shall not exceed 1 per 10,000 transactions.
3.6. Automated failover mechanisms must be in place to switch to backup infrastructure in case of failures.
3.7. The system shall have an MTBF (Mean Time Between Failures) of at least 10,000 hours.

4. Security Requirements

4.1. The system shall maintain logs of all user actions for at least 5 years.
4.2. Only authorized modules shall have access to sensitive banking functions.
4.3. Logs shall be immutable and protected from unauthorized modifications.
4.4. All failed login attempts shall trigger alerts.
4.5. Database consistency checks shall be conducted daily and any anomalies shall trigger alerts.
4.6. All sensitive customer and transaction data must be encrypted both at rest and in transit using industry-standard encryption algorithms.

5. Organizational Requirements

5.1. All software must be developed following bank-approved coding and security guidelines.
5.2. The system shall support biometric authentication (fingerprint) and multi-factor authentication.
5.3. The system shall be accessible on Windows and macOS for employees.
5.4. The system shall have role-based access control (RBAC) to restrict functions based on user roles.

6. External Requirements

6.1. The system must adhere to Basel III and Anti-Money Laundering (AML) regulations to prevent fraud and financial crimes.
6.2. The system must comply with the General Data Protection Regulation (GDPR) to protect customer data.
6.3. The system must comply with the Payment Card Industry Data Security Standard (PCI-DSS) to secure transactions and prevent credit card fraud.
6.4. The system must support secure API integration with payment processors (Visa, MasterCard, PayPal).
6.5. Customers must be able to access the system via web, mobile apps (Android, iOS), and ATMs.

User Scenarios/Use Cases

User Management

UC Name	**UC_UM_01** Maintain Employee Profiles
Summary	The system administrator maintains system profiles for bank employees.
Dependency	None
Actors	Administrator
Preconditions	Administrator has access to the system.

Description of the Main Sequence	1. System administrator navigates to the “Manage Employees” section.
	2. System retrieves and displays a list of current employees.
	3. Administrator selects an action: create, modify or delete an employee profile.
	4. If creating a profile:
	  a. Administrator enters the following information: full name, employee ID, date of birth, SSN, phone number, email, address, workplace, and employment start date.
	  b. Administrator assigns a role: teller, manager, loan officer, auditor, fraud analyst.
	  c. System saves the profile and grants the permissions based on the role.
	5. If modifying a profile:
	  a. Administrator selects an existing employee profile.
	  b. Administrator provides the updated personal details.
	  c. System saves the updated information.
	6. If deleting a profile:
	  a. Administrator selects an employee profile to delete.
	  b. System removes the profile from active employees.
	7. System confirms the action to the administrator.

Description of the Alternative Sequence	**Step 4a, 5b:** If required fields are missing, the system prompts the administrator to complete all mandatory fields.
Non-Functional Requirements	Employee data must be securely stored. System must retrieve employee details within 3 seconds.
Postconditions	Employee profile is successfully created, updated or deleted. Administrator receives confirmation of changes.

UC Name	**UC_UM_02** Login
Summary	Bank employees / Customers log into the system.
Dependency	None
Actors	Bank Employee / Customer
Preconditions	Bank employee / Customer must have an active profile in the system.
Description of the Main Sequence	1. Employee / Customer navigates to the login page. 2. Employee / Customer enters password and username. 3. System verifies credentials. 4. System prompts for multi-factor authentication. 5. Employee / Customer provides the required authentication. 6. System validates authentication. 7. If authentication is successful, access is granted.
Description of the Alternative Sequence	**Step 3:** If credentials are incorrect or incomplete, the system displays an error message: “Invalid Credentials”. **Step 6:** If authentication fails, the system prompts the employee to retry. After four failed login attempts, the system temporarily locks the account.
Non-Functional Requirements	System shall support biometric authentication (fingerprint) and multi-factor authentication. System must log all authentication attempts for audit purposes.
Postconditions	Employee / Customer gains system access.

UC Name	**UC_UM_03** Create Online Account
Summary	Customers create their online banking accounts.
Dependency	None
Actors	Customer
Preconditions	Customer has an existing bank account.
Description of the Main Sequence	1. Customer navigates to the “Create Account” page 2. Customer provides full name, SSN, email address, and password. 3. System validates the information provided against customer information associated with the bank account. 4. System sends a verification email or SMS with a verification link. 5. Customer verifies the account by clicking the verification link.
Description of the Alternative Sequence	**Step 2:** If required fields are missing, the system prompts the customer to complete all fields. **Step 3:** If information provided by the customer is not valid, the system displays an error message: “Invalid credentials.” **Step 4:** If the verification is not completed within 10 minutes, the verification link expires and the customer must retry creating the account.
Non-Functional Requirements	System must securely encrypt and store customer credentials.
Postconditions	Customer account is successfully created.

Funds Management

UC Name	**UC_FM_01** View Funds
Summary	Bank managers view data about funds in their assigned bank units (branches/ATMs).
Dependency	None
Actors	Bank Manager
Preconditions	Bank manager is logged in. Funds data is available.

Description of the Main Sequence	1. Bank manager navigates to the “Funds” section.
	2. Bank manager selects a specific branch/ATM from the available list.
	3. System retrieves and displays unit details and transaction data belonging to the unit.
	4. Bank managers may choose to filter data according to the following categories:
	  a. Incoming/Outgoing funds
	  b. Deposits/Withdrawals/Transfers
	  c. Time period
	5. System retrieves and displays current financial information, including:
	  a. Deposits
	  b. Withdrawals
	  c. Transfers
	  d. Current liquidity

Description of the Alternative Sequence	**Step 5d:** System displays an alert if current liquidity is below a predefined threshold.
Non-Functional Requirements	The system should retrieve fund details within 3 seconds. Secure encryption should be used for data transmission. The interface should be user-friendly and provide filter options for better navigation.
Postconditions	Bank manager views fund details about selected units and is alerted of shortfalls.

UC Name	**UC_FM_02** Allocate Funds
Summary	Bank managers transfer operational funds to bank units (branches/ATMs).
Dependency	Include *UC_FM_01* (View Funds)
Actors	Bank Manager
Preconditions	Bank manager is logged in.
Description of the Main Sequence	1. Include *UC_FM_01* (View Funds) 2. Bank manager selects the “Allocate Funds” option. 3. Bank manager selects the unit for the fund allocation. 4. Bank manager enters the amount of funds to be allocated. 5. System checks if the entered amount is valid. 6. Bank manager confirms fund transfer. 7. System updates the bank unit’s current liquidity. 8. System displays confirmation and generates an allocation information document available for download/print. 9. Bank manager provides their signature.
Description of the Alternative Sequence	**Step 5:** If validation fails, the system displays an error message: “Invalid transfer amount.”
Non-Functional Requirements	The system should complete the fund allocation within 3 seconds. Secure encryption should be applied to all financial transactions. The system should log all actions for auditing purposes.
Postconditions	Funds are allocated to the selected bank unit. Confirmation of the allocation is issued to the bank manager.

Account Management

UC Name	**UC_AM_01** Open Bank Account
Summary	Bank tellers open a new account by selecting account type, currency, and providing customer details.
Dependency	None
Actors	Bank Teller, Customer
Preconditions	Bank teller is logged in. Customer details are available.
Description of the Main Sequence	1. Customer requests a new account at a branch. 2. Bank teller selects "Open New Account" option. 3. Bank teller selects account type (savings, checking, fixed deposit, business), currency (Lek, Euro, USD), interest rate, and maintenance fees. 4. Bank teller enters customer details (full name, SSN, birthdate, phone, email, address). 5. System validates the information. 6. If validation is successful, the system creates a new bank account with a unique account number and associates it with the customer. 7. System confirms account creation and displays account details. 8. System generates an account opening document for download/print. 9. Bank teller and customer provide their signature. 10. System periodically applies interest/maintenance fees, which are reflected in the account's balance.
Description of the Alternative Sequence	**Step 4:** If the customer does not already exist, the customer is saved in the system. **Step 5:** If validation fails (missing/invalid data), system displays an error message: "Invalid data."
Non-Functional Requirements	The system should process account creation within 3 seconds. Secure data encryption should be applied to customer information. The system should provide a user-friendly interface for data entry. The system should log all actions for auditing purposes.
Postconditions	A new bank account is created and linked to the customer. Confirmation of the account opening is provided to the bank teller. Interest is applied periodically and balance is updated.

UC Name	**UC_AM_02** Close Bank Account
Summary	Bank tellers close existing accounts upon customer request.
Dependency	Include *UC_TM_02A* (Withdraw Money at Branch)
Actors	Bank Teller, Customer
Preconditions	Account that is being processed for closure already exists. Account balance is settled (no pending transactions or negative balance).
Description of the Main Sequence	1. Customer requests account closure. 2. Bank teller navigates to the "Account Closure" section. 3. Bank teller searches for the customer’s account using the account number. 4. System retrieves and displays account details. 5. To clear outstanding balances, include *UC_TM_02A* (Withdraw Money at Branch). 6. Bank teller confirms closure alongside justification for closure. 7. System updates account status to “Closed”. 8. System checks for linked cards. 9. System generates a closure confirmation document for download/print. 10. Bank teller and customer provide their signature.
Description of the Alternative Sequence	**Step 3:** If the account is not found, the system displays an error: "No matching account found." **Step 8:** If the account has linked cards, deactivate linked cards.
Non-Functional Requirements	The system should process account closure within 3 seconds. Secure encryption should be used for customer and transaction data. The system should log all actions for auditing purposes.
Postconditions	Account is successfully closed. Account balance is withdrawn. Linked cards are deactivated.

UC Name	**UC_AM_03** Modify Bank Account
Summary	Bank teller modifies account details, such as owner's personal information, account currency, and applicable interest rates.
Dependency	None
Actors	Bank Teller, Customer
Preconditions	Customer has an active bank account.

Description of the Main Sequence	1. Customer requests account modification.
	2. Bank teller navigates to the "Account Management" section.
	3. Bank teller searches for the customer's account by entering account details.
	4. System retrieves and displays current account details.
	5. Bank teller enters updated information, such as:
	  a. Account status
	  b. Account currency
	  c. Applicable interest rate
	  d. Applicable maintenance fees
	6. System validates the modifications.
	7. System displays a confirmation message to the bank teller.
	8. System sends notification to the customer concerning the changes.

Description of the Alternative Sequence	**Step 3:** If the account is not found, the system displays an error: "No matching account found." **Step 5c:** If currency conversion is requested, bank teller must provide a conversion rate. **Step 6:** If validation fails (missing/invalid data), system displays an error message: "Invalid data".
Non-Functional Requirements	Updates should be processed within 3 seconds. System must log all modifications for audit purposes.
Postconditions	Account details are updated. Customer is notified of the changes.

UC Name	**UC_AM_04** View Bank Account Information
Summary	Customers access their bank account details and transaction history.
Dependency	None
Actors	Customer
Preconditions	Customer has an active bank account. Customer is logged into their online account. Account information is available.

Description of the Main Sequence	1. Customer navigates to the "Accounts" section.
	2. System retrieves and displays a list of the customer's accounts.
	3. Customer selects an account to view details.
	4. System displays the following account details:
	  a. Account number
	  b. Account type and currency
	  c. Balance
	  d. Interest rate
	5. System retrieves and displays chronologically all transactions associated with the account.
	6. Customer can filter transactions by:
	  a. Date range
	  b. Type (deposits, withdrawals, transfers, application of interest/maintenance fees, recurring transfers)
	  c. Amount
	7. System retrieves and displays details of filtered transactions.

Description of the Alternative Sequence	**Step 7:** If no transactions are found after application of filters, the system displays: "No transactions found for this criteria."
Non-Functional Requirements	System should retrieve account information within 3 seconds. Transaction history should be retrievable for at least the past 12 months. Secure encryption should protect customer and account data.
Postconditions	The customer successfully views account details and transaction history.

Transaction Management

UC Name	**UC_TM_01** Deposit Money
Summary	Customers deposit money into their accounts upon bank teller approval.
Dependency	None
Actors	Bank Teller, Customer
Preconditions	Bank teller is logged in. Customer has an active bank account. Customer possesses cash or check for deposit.
Description of the Main Sequence	1. Customer requests a deposit at a branch. 2. Bank teller navigates to the “Deposits” section. 3. Bank teller searches for the customer’s account by entering account details. 4. System retrieves and displays account details. 5. Bank teller enters deposit details (amount, deposit type: cash/check). 6. System validates the input. 7. Bank teller confirms the deposit. 8. System updates account balance and transaction history after approval. 9. System generates a deposit confirmation receipt for download/print. 10. Bank teller and customer provide their signature.
Description of the Alternative Sequence	**Step 4:** If the account is not found, the system displays an error: "No matching account found. Please verify the details and try again." **Step 6:** If validation fails (e.g., missing/incorrect information), the system prompts the bank teller for correction.
Non-Functional Requirements	Deposits should be processed within 3 seconds. Cash and check authenticity should be verified according to bank policies. Secure encryption should be used for transaction data. System should log all actions for audit purposes.
Postconditions	The deposited amount is reflected in the customer’s account and transaction history. A deposit receipt is generated.

UC Name	**UC_TM_02** Withdraw Money
Summary	Customers withdraw money from their bank accounts.
Dependency	None
Actors	Customer
Preconditions	Customer has an active bank account eligible for withdrawals.
Description of the Main Sequence	See *UC_TM_02A*, *UC_TM_02B*
Description of the Alternative Sequence	See *UC_TM_02A*, *UC_TM_02B*
Non-Functional Requirements	Withdrawals should be processed within 3 seconds. Secure authentication must be used before withdrawal. System should log all actions for audit purposes.
Postconditions	The withdrawn amount is reflected in the customer’s account and transaction history. Customer receives cash and a withdrawal receipt.

UC Name	**UC_TM_02A** Withdraw Money (Branch)
Summary	Customers withdraw money from their bank accounts at a physical bank branch with the help of a bank teller, subject to balance availability and withdrawal amount limits, with applicable penalties for certain account types.
Dependency	Inherit *UC_TM_02* (Withdraw Money) Include *UC_UM_02* (Login)
Actors	Customer (inherited), Bank Teller
Preconditions	Bank teller is logged in.
Description of the Main Sequence	1. Customer requests a withdrawal at a branch. 2. Bank teller navigates to the “Withdrawals” section. 3. Bank teller searches for the customer’s account by entering account details. 4. System retrieves and displays account details. 5. Bank teller enters the withdrawal amount. 6. System checks if the account has sufficient funds and if the withdrawal amount exceeds daily withdrawal limits. 7. Bank teller confirms the withdrawal. 8. System updates the account balance and transaction history. 9. System generates a withdrawal receipt for download/print. 10. Bank teller and customer provide their signature.
Description of the Alternative Sequence	**Step 4:** If the account is not found, the system displays an error: "No matching account found. Please verify the details and try again." **Step 7:** If account does not have sufficient funds, the system displays an error message: “Insufficient balance for withdrawal.”, **Step 7:** If requested amount exceeds limits, the system displays an error message: “Withdrawal amount exceeds daily withdrawal limit.”
Non-Functional Requirements	Inherit *UC_TM_02* (Withdraw Money)
Postconditions	Inherit *UC_TM_02* (Withdraw Money)

UC Name	**UC_TM_02B** Withdraw Money (ATM)
Summary	Customers make limited withdrawals from their bank accounts using an ATM.
Dependency	Inherit *UC_TM_02* (Withdraw Money)
Actors	Customer (inherited), ATM
Preconditions	Customer has an active card. ATM is operational and has sufficient cash.

Description of the Main Sequence	1. Customer inserts a card into the ATM machine.
	2. ATM reads card number and checks whether card exists in the system.
	3. ATM prompts for the security code.
	4. Customer provides security code.
	5. System verifies:
	  a. PIN
	  b. Card expiration date
	  c. Card status (active/frozen).
	6. System displays current account balance.
	7. Customer selects the “Withdraw” option.
	8. ATM displays preset withdrawal amount options.
	9. Customer selects withdrawal amount.
	10. System checks if the account has sufficient funds or if the withdrawal amount exceeds daily withdrawal limits.
	11. If funds are available, ATM dispenses cash.
	12. ATM generates a withdrawal receipt.
	13. System updates the customer’s account balance and transaction history.
	14. ATM returns the card.

Description of the Alternative Sequence	**Step 2:** If card is not found in the system, the system displays an error message: “Card not valid.” and ATM ejects the card. **Step 5a:** If the PIN is incorrect, the system displays an error message: “Incorrect PIN. Please try again.” After four failed attempts, the ATM locks the card and the system updates its status to “Frozen”. **Step 5b:** If the card is past its expiration date, the system displays an error message: “Card expired. Please visit a branch to obtain a new card.” and ATM ejects the card. **Step 5c:** If the card is frozen or invalid, the system displays an error message: “Card not valid.” and ATM ejects the card. **Step 10:** If account does not have sufficient funds, the system displays an error message: “Insufficient balance for withdrawal.”, **Step 10:** If requested amount exceeds limits, the system displays an error message: “Withdrawal amount exceeds daily withdrawal limit.”
Non-Functional Requirements	Inherit *UC_TM_02* (Withdraw Money) PIN authentication must be encrypted and secure.
Postconditions	Inherit *UC_TM_02* (Withdraw Money)

UC Name	**UC_TM_03** Make Transfers
Summary	Customers transfer funds.
Dependency	None
Actors	Customer
Preconditions	Customer has an active bank account. Customer is logged into their online account.
Description of the Main Sequence	1. Customer navigates to the “Transfer Funds” section.
Description of the Alternative Sequence	No alternative sequence
Non-Functional Requirements	Transfers should be processed within 3 seconds. Secure authentication must be used before transfer. Transaction logs should be maintained for audit purposes.
Postconditions	Transfer is successfully registered. Both accounts reflect updated balances and transaction histories. Confirmation is provided to the customer.

UC Name	**UC_TM_03A** Make Internal Transfers
Summary	Customers transfer funds between accounts within the same bank.
Dependency	Inherit *UC_TM_03* (Make Transfers)
Actors	Customer (inherited)
Preconditions	Inherit *UC_TM_03* (Make Transfers)
Description of the Main Sequence	1. Inherit *UC_TM_03* (Make Transfers) 2. Customer selects the “Internal Transfer” option. 3. Customer selects the account for the transaction. 4. Customer enters recipient account details (name, account number). 5. System checks the validity of the recipient account. 6. Customer enters the transfer amount. 7. System checks if sufficient funds are available. 8. Customer confirms the transfer. 9. System updates both sender and recipient’s balance and transaction history. 10. System generates a transaction receipt for download.
Description of the Alternative Sequence	**Step 5:** If the recipient account details are incorrect, the system displays an error message: “Invalid recipient account.” **Step 7:** If the transfer amount exceeds available funds, the system displays an error message: “Insufficient balance for transfer.”
Non-Functional Requirements	Inherit *UC_TM_03* (Make Transfers)
Postconditions	Inherit *UC_TM_03* (Make Transfers)

UC Name	**UC_TM_03B** Make External Transfers
Summary	Customers transfer money to external bank accounts via wire transfer.
Dependency	Inherit *UC_TM_03* (Make Transfers)
Actors	Customer (inherited), Wire Transfer Network System
Preconditions	Inherit *UC_TM_03* (Make Transfers)
Description of the Main Sequence	1. Inherit *UC_TM_03* (Make Transfers) 2. Customer selects the “External Transfer” option. 3. Customer selects the account for the transaction. 4. Customer enters recipient account details (name, account number). 5. Wire Transfer Network System verifies the account. 6. Customer enters the transfer amount. 7. System checks if sufficient funds are available. 8. Customer confirms the transfer. 9. System applies external transfer fees. 10. Wire Transfer Network System receives the transaction for processing. 11. System updates the customer’s balance and transaction history. 12. System generates a transaction receipt for download.
Description of the Alternative Sequence	**Step 4:** If recipient account details are not validated by the wire transfer system, the system displays an error message: “Invalid recipient account.” **Step 7:** If the transfer amount exceeds available funds, the system displays an error message: “Insufficient balance for transfer.”
Non-Functional Requirements	Inherit *UC_TM_03* (Make Transfers)
Postconditions	Inherit *UC_TM_03* (Make Transfers)

UC Name	**UC_TM_03C** Set Up Recurring Transfers
Summary	Customers can set up automated recurring transfers.
Dependency	Inherit *UC_TM_03* (Make Transfers)
Actors	Customer (inherited)
Preconditions	Inherit *UC_TM_03* (Make Transfers)

Description of the Main Sequence	1. Inherit *UC_TM_03* (Make Transfers)
	2. Customer selects the “Set Up Recurring Payments” option.
	3. Customer enters recipient account details (name, account number).
	4. System checks the validity of the recipient account.
	5. Customer specifies payment frequency (daily, weekly, monthly, etc.), start date, end date, and transfer amount.
	6. Customer confirms the recurring payment.
	7. System generates a confirmation message and saves the recurring transaction.
	8. System checks the following before processing a scheduled transfer:
	  a. if customer has cancelled
	  b. if end date has passed
	  c. if sufficient funds are available
	9. System updates the account balance after every payment.

Description of the Alternative Sequence	**Step 4:** If the recipient account details are incorrect, the system displays an error message: “Invalid recipient account.” **Step 8a,b:** If the customer cancels or end date has passed, system stops future payments. **Step 8c:** If the account has insufficient funds at the scheduled time, the system sends a notification: “Scheduled transfer failed due to insufficient funds.
Non-Functional Requirements	Inherit *UC_TM_03* (Make Transfers) Notifications should be sent 24 hours before the scheduled transfer.
Postconditions	Inherit *UC_TM_03* (Make Transfers) Customer is notified before every payment.

UC Name	**UC_TM_04** Make Card Payments
Summary	Customers make payments using their debit or credit card for various services or purchases.
Dependency	None
Actors	Customer, Payment Processor
Preconditions	Customer has an active bank card. POS (Point of Sale) integrates with a payment processor that is recognized by the system.
Description of the Main Sequence	1. Customer initiates a card payment at a POS (online website, payment terminal). 2. Customer scans the card on the payment processor. 3. Payment processor verifies card details and status with the system. 4. Customer selects the payment amount. 5. Payment processor queries the system for sufficient balance or card limits. 6. Customer confirms payment. 7. Upon successful validation, system saves the payment and confirms it to the customer. 8. System updates the customer's balance and transaction history. 9. System generates a payment receipt.
Description of the Alternative Sequence	**Step 2:** If card is invalid, system displays an error message: "Invalid card." **Step 3:** If card is not active, system displays an error message: "Inactive card." **Step 5:** If payment exceeds available funds, the system responds with an error message: "Insufficient balance for payment." **Step 5:** If payment exceeds card spending limit, the system responds with an error message: "Payment amount exceeds card limits."
Non-Functional Requirements	Payment transaction should complete within 3 seconds. System must comply with the Payment Card Industry Data Security Standard (PCI-DSS) System logs all transaction actions for audit purposes.
Postconditions	Account balance and transaction history is updated. A payment receipt is generated.

Loan Management

UC Name	**UC_LM_01** Apply for Loans
Summary	Customers apply for various types of loans, which are reviewed by loan officers.
Dependency	None
Actors	Customer, Loan Officer
Preconditions	Bank teller is logged in. Customer is logged into their online account.
Description of the Main Sequence	1. Customer navigates to the “Loan Application” section. 2. Customer selects one of the following loan types: personal, student, auto, home, payday, business. 3. Customer chooses loan terms, including loan amount and monthly installment amount. 4. Customer provides supporting personal and financial documents. 5. System validates the information. 6. Customer submits the application. 7. System saves the loan application with a status of “Submitted”. 8. Loan officer navigates to the "Loan Applications" section. 9. System retrieves and displays the list of submitted applications. 10. Loan officer selects an application to review. 11. Loan officer approves or rejects the loan application. 12. System updates application status to “Approved”. 13. Loan officer provides details for the new loan, including interest rate and loan start date. 14. System generates a loan confirmation document. 15. Loan officer and customer provide their signature. 16. System generates an overview of the loan repayment plan, including due dates, amounts, and remaining balance.
Description of the Alternative Sequence	**Step 5:** If validation fails (e.g., missing/incorrect information), the system prompts the customer for correction. **Step 11:** If the application is rejected, system updates loan status to “Rejected” and notifies the customer.
Non-Functional Requirements	Loan applications should be processed within 3 seconds. Loan applications should be reviewed within 48 hours. The system must securely store personal and financial information.
Postconditions	Loan is either stored in the system with a status of “Applied”, “Approved”, or “Rejected”. Customer is notified of changes to the loan application status. Customer receives loan details and repayment information.

UC Name	**UC_LM_02** Repay Loans
Summary	Customers repay their loans while the system tracks repayment schedules, outstanding balances, and penalties.
Dependency	None
Actors	Customer
Preconditions	Customer must have an active loan. Customer is logged in. Loan details are available.
Description of the Main Sequence	1. Customer navigates to the “Loan Repayment” section. 2. System displays loan details, including payment schedule, outstanding balance and amount due. 3. Customer selects “New Loan Payment”. 4. System checks if sufficient funds are available. 5. Customer confirms payment. 6. System processes the payment and updates the outstanding balance. 7. System generates a loan repayment receipt. 8. System updates the repayment schedule.
Description of the Alternative Sequence	**Step 4:** If the payment amount exceeds available funds, the system displays an error message: “Insufficient funds for loan repayment.” **Step 6:** If a payment is overdue, the system calculates and applies penalties as per bank policy. **Step 6:** If the loan has been fully repaid, system updates its status to “Repaid”.
Non-Functional Requirements	Customers should receive notifications for due payments at least 3 days in advance. Secure authentication must be used. Transaction logs should be maintained for audit purposes.
Postconditions	Loan balance is updated. Customers receive a confirmation of their payment. If overdue, penalties are applied.

Card & Check Management

UC Name	**UC_CM_01** Apply for Cards
Summary	Allows customers to apply for a debit or credit card online.
Dependency	None
Actors	Customer, Bank Teller
Preconditions	Customer has an active bank account. Customer is logged in. Bank teller is logged in.
Description of the Main Sequence	1. Customer selects “Apply for a New Card”. 2. Customer selects from available card types: debit or credit. 3. Customer selects the account for the card. 4. Customer selects a delivery method (pickup from branch, mail delivery). 5. Customer submits the application. 6. System saves the application. 7. Bank teller navigates to the "Card Applications" section. 8. System retrieves and displays the list of submitted applications. 9. Bank teller selects an application to review. 10. Bank teller approves or rejects the application. 11. If approved, the system generates a unique card number and security code. 12. System sends a notification to the customer once the card is issued.
Description of the Alternative Sequence	**Step 10:** If the bank employee rejects the application, the system notifies the customer.
Non-Functional Requirements	Card application should be processed within 3 seconds. Processing time for approval should be within 1 business day. Customer data must be securely stored and encrypted.
Postconditions	Approved customers receive a notification about their card issuance. Rejected customers are informed of the decision.

UC Name	**UC_CM_02** Request Checkbooks
Summary	Customers can request checkbooks for their checking accounts.
Dependency	None
Actors	Customer, Bank Teller
Preconditions	Customer has a bank account eligible for check transactions. Customer is logged in. Bank teller is logged in.
Description of the Main Sequence	1. Customer navigates to the "Request Checkbook" section. 2. Customer selects the bank account for the checkbook. 3. Customer selects the number of checks/pages needed. 4. Customer selects a delivery method (pickup from branch, mail delivery). 5. System validates the information. 6. Customer submits the application. 7. Bank teller navigates to the "Application" section. 8. System retrieves and displays the list of submitted applications. 9. Bank teller selects an application to review. 10. Bank teller approves or rejects the application. 11. If approved, the system generates unique serial numbers for the checks. 12. System sends a notification to the customer once the checkbook is issued.
Description of the Alternative Sequence	**Step 5:** If validation fails (e.g., missing/incorrect information), the system prompts the customer for correction. **Step 10:** If the bank employee rejects the application, the system notifies the customer.
Non-Functional Requirements	Checkbook applications should be processed within 3 seconds. Check requests should be reviewed within 2 business days. Customer data must be securely stored and encrypted.
Postconditions	A checkbook is issued to the customer. Rejected customers are informed of the decision.

UC Name	**UC_CM_03** Set Card Limits
Summary	Customers set spending limits on their cards for better financial control.
Dependency	None
Actors	Customer
Preconditions	Customer has an active bank card. Customer is logged into the online account. Card details are available.
Description of the Main Sequence	1. Customer navigates to the “Manage Cards” section. 2. System retrieves and displays information on all cards associated with the customer. 3. Customer selects a card. 4. Customer selects “Set Spending Limits”. 5. Customer enters and confirms limit preferences (e.g., per transaction, per day). 6. System validates customer input. 7. System updates card limits and displays confirmation.
Description of the Alternative Sequence	**Step 6:** If validation fails (e.g., incorrect number format, negative input), the system prompts for correction.
Non-Functional Requirements	Spending limits should take effect immediately.
Postconditions	Card limits are updated. Card transactions are restricted based on the set limits.

Fraud Detection

UC Name	**UC_FD_01** Handle Fraudulent Activities
Summary	Fraud analysts review and take action on suspicious transactions or account access attempts.
Dependency	None
Actors	Fraud Analyst
Preconditions	Fraud analyst is logged in. Fraud data is available.
Description of the Main Sequence	1. Fraud analyst navigates to the "Flagged Activities" section. 2. System retrieves and displays both system-flagged and customer-reported activities. 3. Fraud analyst selects a flagged activity for review. 4. System displays the following details: a. Customer account information b. Activity type and details (transaction, account access) c. Attached documents 5. If the activity reported is not suspicious, fraud analyst activates the account again. 6. System confirms the action to the fraud analyst and the customer.
Description of the Alternative Sequence	**Step 5:** If the activity reported is suspicious, fraud analyst updates account status to Frozen.
Non-Functional Requirements	Fraud alerts must be reviewed within 1 hour of detection. Sensitive customer data must be securely encrypted.
Postconditions	Suspicious activity is either approved or blocked. Customer is notified of fraud-handling actions.

UC Name	**UC_FD_02** Report Fraudulent Activities
Summary	Customers report fraudulent activities related to their account.
Dependency	Include *UC_FD_01* (Handle Fraudulent Activities)
Actors	Customer, Fraud Analyst
Preconditions	Customer has an active bank account. Customer is logged into their online account.
Description of the Main Sequence	1. Customer navigates to the "Report Fraud" section. 2. Customer selects the type of fraud: a. Unauthorized transaction b. Suspicious login attempt c. Other 3. Customer provides details (e.g., transaction ID, date, amount). 4. Customer uploads supporting documents (optional). 5. System validates the information. 6. Customer submits the fraud report. 7. System generates a case reference number and saves the report. 8. System temporarily suspends account access/activities. 9. Include *UC_FD_01* (Handle Fraudulent Activities).
Description of the Alternative Sequence	**Step 5:** If required fields are missing, system prompts the customer to complete the information.
Non-Functional Requirements	Fraud report submission should be processed within 3 seconds. Sensitive customer data must be securely encrypted.
Postconditions	Fraud report is successfully logged and handled.

Customer Services

UC Name	**UC_CS_01** Access Bank-Related Information
Summary	Customers are able to access real-time information about ATMs and branches.
Dependency	None
Actors	Customer
Preconditions	Bank information is available. Customer is logged into their online account.
Description of the Main Sequence	1. Customer navigates to the “Bank Information” section. 2. System retrieves and displays a list of available ATMs and branches. 3. Customer can filter bank units based on their location or type (ATM/physical branch). 4. System displays a filtered list of available ATMs and branches. 5. Customer selects a specific branch or ATM from the list. 6. System displays the details of the selected branch/ATM, including: a. Working hours b. Contact details c. Available services d. Temporary closures or disruptions.
Description of the Alternative Sequence	**Step 4:** If no records match the selected filters, the system displays: "No bank units found for the given criteria."
Non-Functional Requirements	System should retrieve and display information in under 5 seconds for optimal user experience. System should provide an easy-to-navigate interface.
Postconditions	Customer views up-to-date ATM/branch information.

UC Name	**UC_CS_02** Schedule Appointments
Summary	Customers are able to schedule in-branch appointments for different banking services online.
Dependency	None
Actors	Customer, Bank Teller
Preconditions	Customer is logged in. Bank teller is logged in.
Description of the Main Sequence	1. Customer navigates to the “Schedule Appointment” section. 2. Customer selects the type of service (loan consultation, account opening, etc.) and a branch. 3. System displays available time slots at the selected branch. 4. Customer selects a convenient time. 5. Customer confirms appointment. 6. System updates appointment calendar. 7. System sends an appointment confirmation notification to the customer. 8. Bank teller navigates to the “Appointments” section. 9. System retrieves and displays the updated appointment calendar.
Description of the Alternative Sequence	**Step 3:** If no slots are available, the system displays: "No available time slots at the selected branch. Please try another option."
Non-Functional Requirement	System should display available appointment slots within 3 seconds. Appointment booking should be completed within 2 minutes. Appointment data must be securely stored and retrieved.
Postconditions	Appointment is successfully scheduled, confirmed, and reflected in the system.

UC Name	**UC_CS_03** Submit Feedback
Summary	Customers can submit feedback and complaints about their banking experience online.
Dependency	None
Actors	Customer, Bank Teller
Preconditions	Customer is logged in. Bank teller is logged in.
Description of the Main Sequence	1. Customer navigates to the “Feedback” section. 2. Customer selects the type of feedback/complaint (service, transaction issue, etc.). 3. Customer provides details of the complaint/feedback. 4. Customer submits complaint/feedback. 5. System generates a unique tracking number for the complaint and confirms the submission with the customer. 6. Bank teller navigates to the “Feedback” section. 7. System retrieves and displays submitted feedback. 8. Bank teller reviews the complaint. 9. Bank teller responds to the complaint with a resolution. 10. System updates status of feedback. 11. System sends update notifications to the customer regarding resolution of the complaint.
Description of the Alternative Sequence	No alternative sequence
Non-Functional Requirements	Feedback submission should be processed within 3 seconds. Complaints and feedback must be securely stored. Complaints must be addressed within 5 business days.
Postconditions	Complaint is successfully submitted. Customers are notified of updates on the status of their complaints.

UC Name	**UC_CS_04** Request Official Documents
Summary	Customers can request various account-related documents and be notified when requested documents are available.
Dependency	None
Actors	Customer, Bank Teller
Preconditions	Customer must have an active account with the bank for which they can request documents. Customer is logged in. Bank teller is logged in.

Description of the Main Sequence	1. Customer navigates to the “Request Documents” section.
	2. Customer selects the type of document they wish to request (e.g. account verification, loan payoff).
	3. Customer selects the delivery method:
	  a. Download
	  b. Physical copy (requires selection of pickup branch)
	4. System generates a request for the document and notifies the customer that the request is being processed.
	5. Bank teller navigates to the “Document Requests” section.
	6. System retrieves and displays document requests.
	7. Bank teller reviews the document request.
	8. Bank teller prepares and signs the requested document.
	9. Customer is notified to download or collect the physical copy of the document.

Description of the Alternative Sequence	**Step 7:** If the document request is denied by the bank employee, the system will notify the customer that their request was denied.
Non-Functional Requirements	Online document requests should be processed within 1 day. Physical document requests should be processed within 2-3 business days. Secure encryption must protect sensitive customer data.
Postconditions	Customer’s document request is successfully submitted. Customer is notified once the document is ready.

Audit & Reporting

UC Name	**UC_AR_01** View Audit Trails
Summary	Auditors access and review audit trails to ensure compliance with banking regulations.
Dependency	None
Actors	Auditor
Preconditions	Log trails are available. Auditor is logged in.

Description of the Main Sequence	1. Auditor navigates to the "Audit Trails" section.
	2. System retrieves and displays logs chronologically, including:
	  a. Login attempts
	  b. Account modifications
	  c. Financial transactions
	3. Auditor can filter records by:
	  a. Date range
	  b. User
	  c. Type (from previous points)
	4. System displays a filtered list of records.
	5. Auditor downloads audit log documents for compliance purposes.

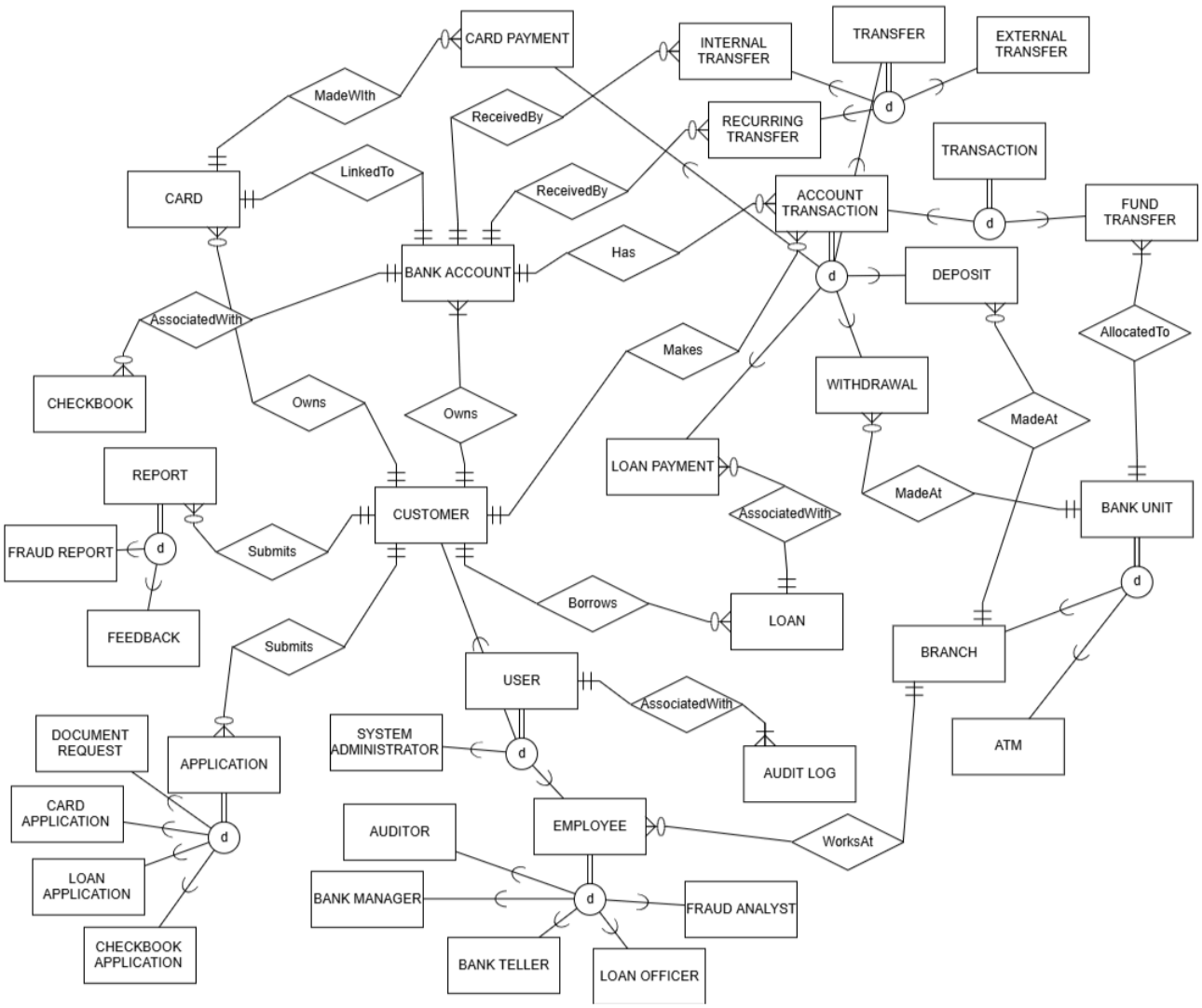
Description of the Alternative Sequence	**Step 3:** If no records match the selected filters, the system displays: "No audit logs found for the given criteria."
Non-Functional Requirements	Audit logs must be stored securely and encrypted. The system must ensure logs cannot be tampered with. System should support retrieval of logs for at least 7 years.
Postconditions	Auditor successfully views, filters, and downloads audit logs.

UC Name	**UC_AR_02** View Customer Analytics
Summary	Bank managers access customer analytics to gain insight into customer behavior and banking trends.
Dependency	None
Actors	Bank Manager
Preconditions	Customer analytics is available. Bank manager is logged in.
Description of the Main Sequence	1. Manager navigates to the "Customer Analytics" section. 2. Manager selects specific filters such as: a. Date range b. Customer demographics (age, location) c. Account type d. Account activities (savings, spending, loaning, transfers) 3. System retrieves and displays customer analytics in graphs and tables. 4. Manager downloads customer analytics reports for strategic decision-making.
Description of the Alternative Sequence	**Step 3:** If no records match the selected filters, the system displays: "No data found for the given criteria."
Non-Functional Requirements	Customer data should be retrieved within 3 seconds. Data visualization should be interactive and user-friendly. Customer data must be anonymized to protect privacy.
Postconditions	Manager successfully views and exports customer analytics.

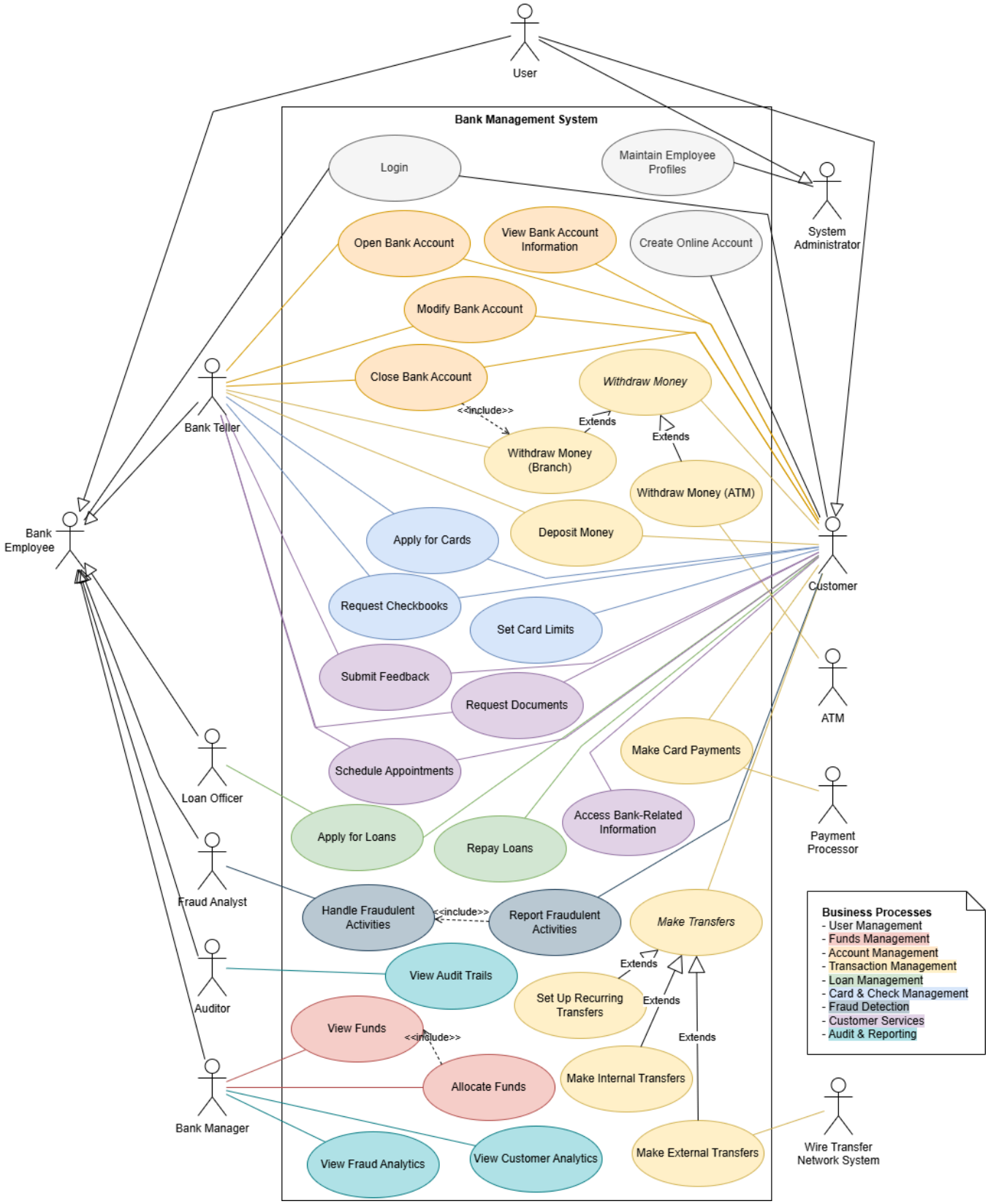
UC Name	**UC_AR_03** View Fraud Detection Analytics
Summary	Bank managers access fraud detection analytics to extract fraud insights.
Dependency	None
Actors	Bank Manager
Preconditions	Fraud detection analytics is available. Bank manager is logged in.
Description of the Main Sequence	1. Manager navigates to the “Fraud Analytics” section. 2. Manager selects analysis filters such as: a. Date range b. Geographical location c. Customer demographics d. Account and transaction types 3. System retrieves and displays fraud detection data in graphs and tables. 4. Manager downloads fraud data reports.
Description of the Alternative Sequence	**Step 3:** If no records match the selected filters, the system displays: "No data found for the given criteria."
Non-Functional Requirements	Fraud detection data should be retrieved within 3 seconds. Data visualization should be interactive and user-friendly. Customer data must be anonymized to protect privacy.
Postconditions	Manager successfully views and exports fraud analytics.

Diagrams

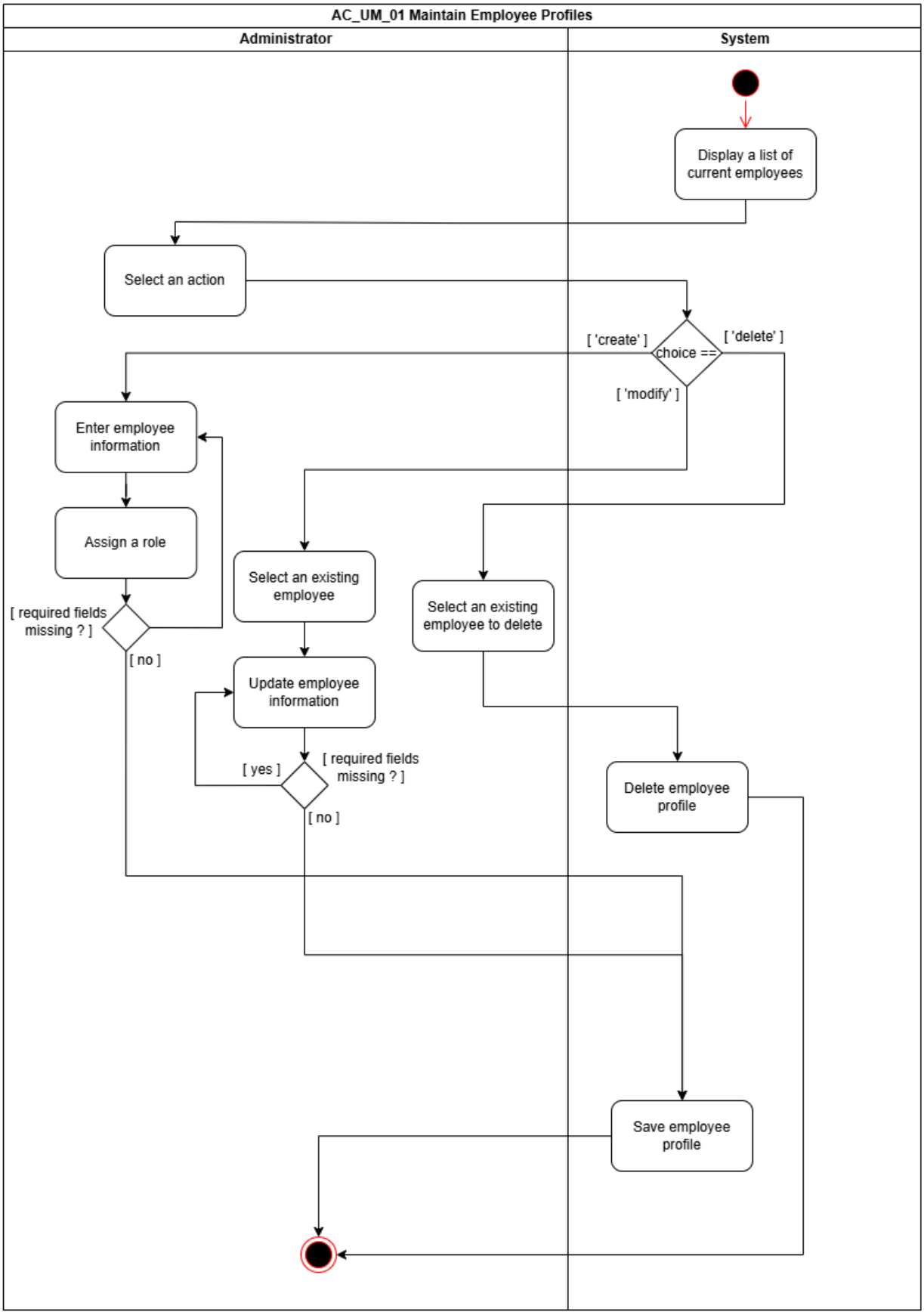
ER Diagram

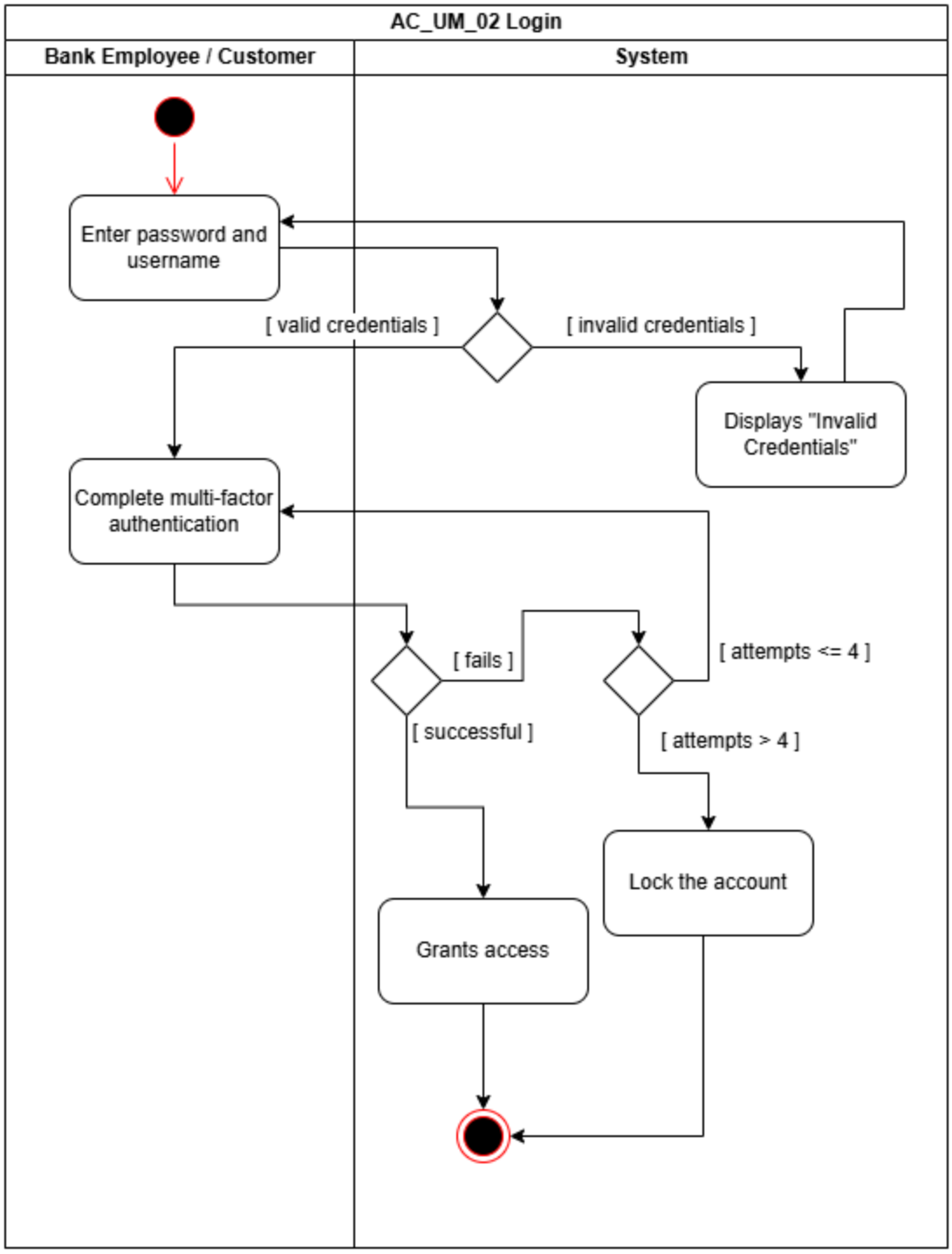


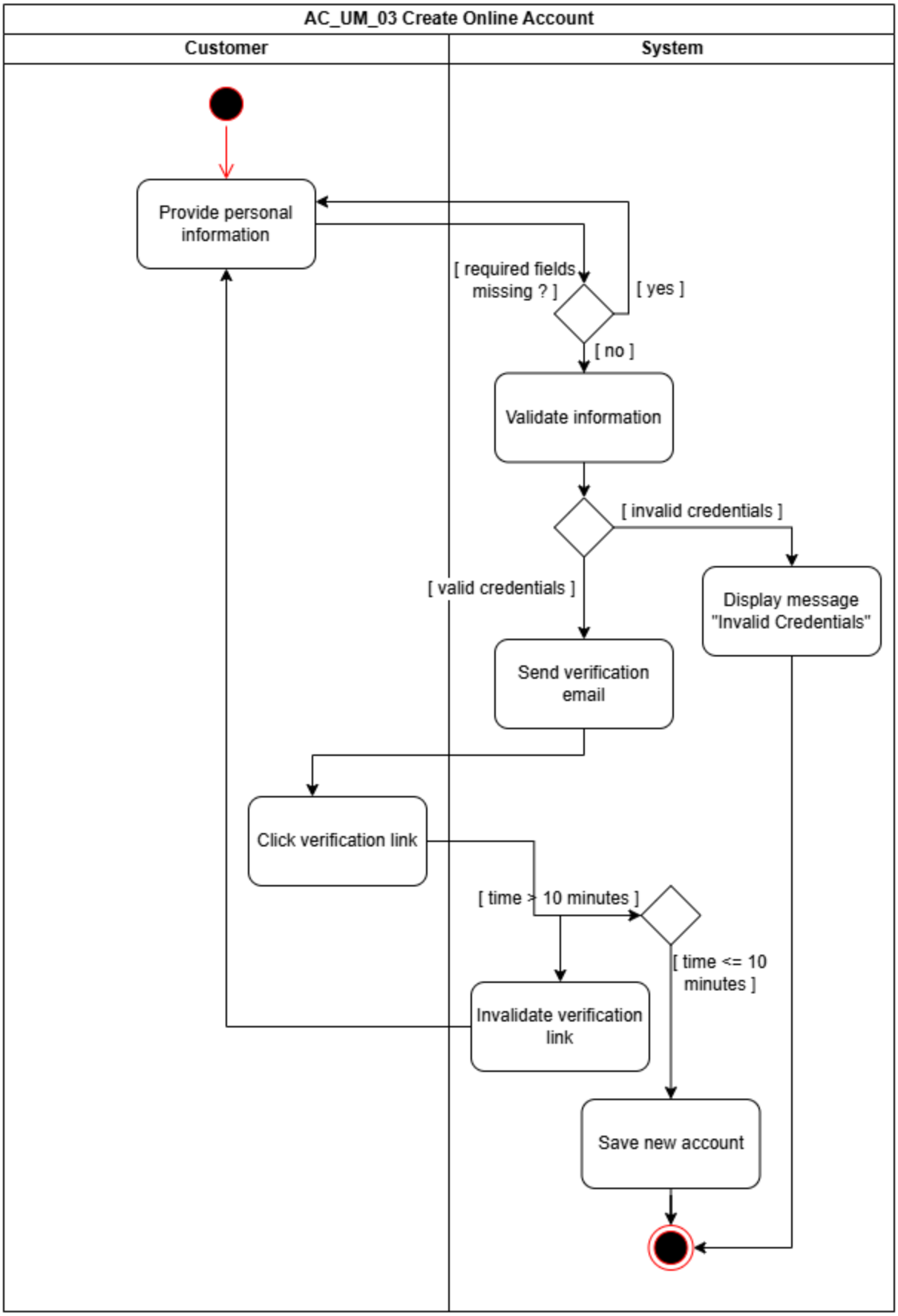
Use Case Diagram

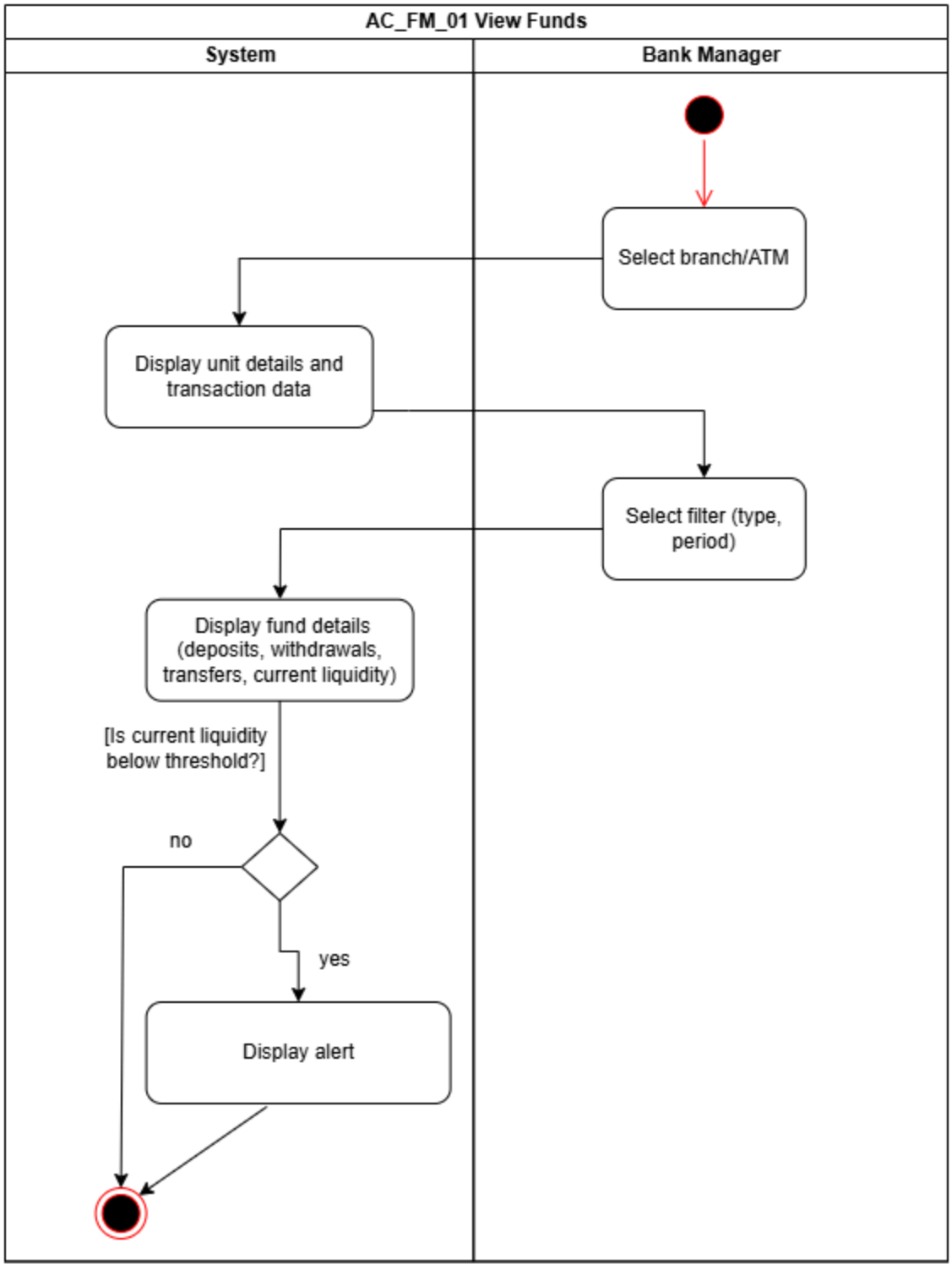


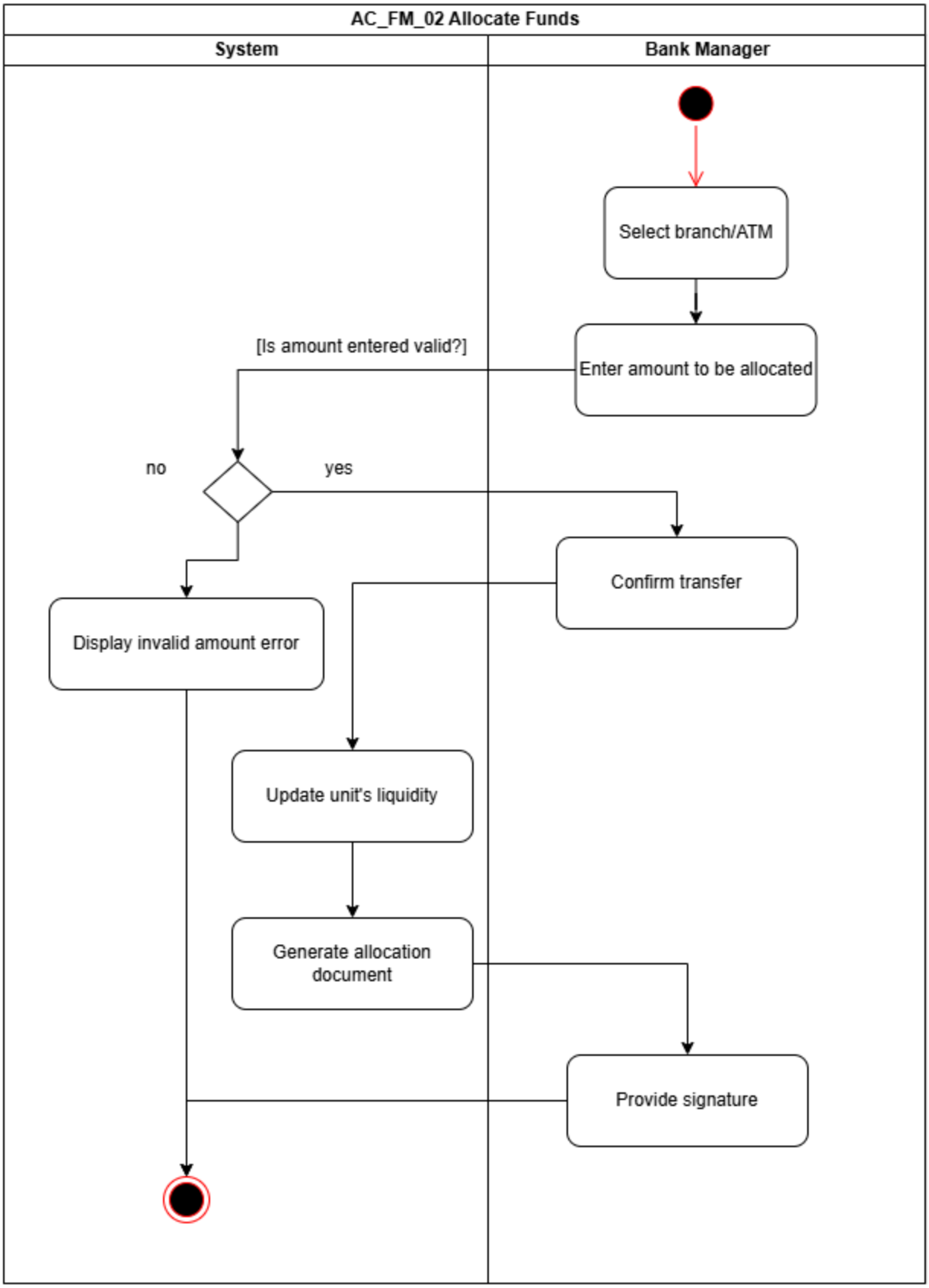
Activity Diagrams

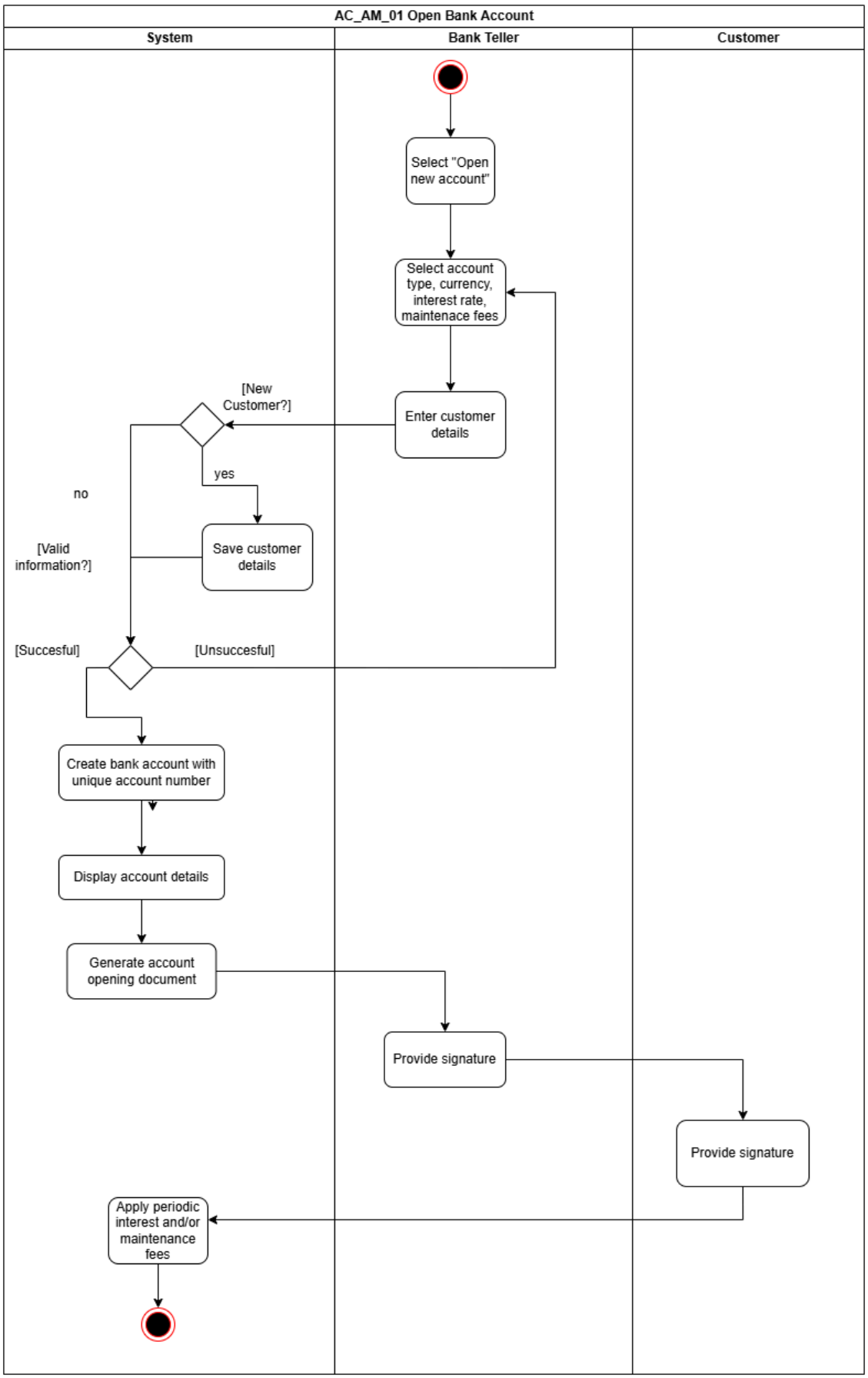


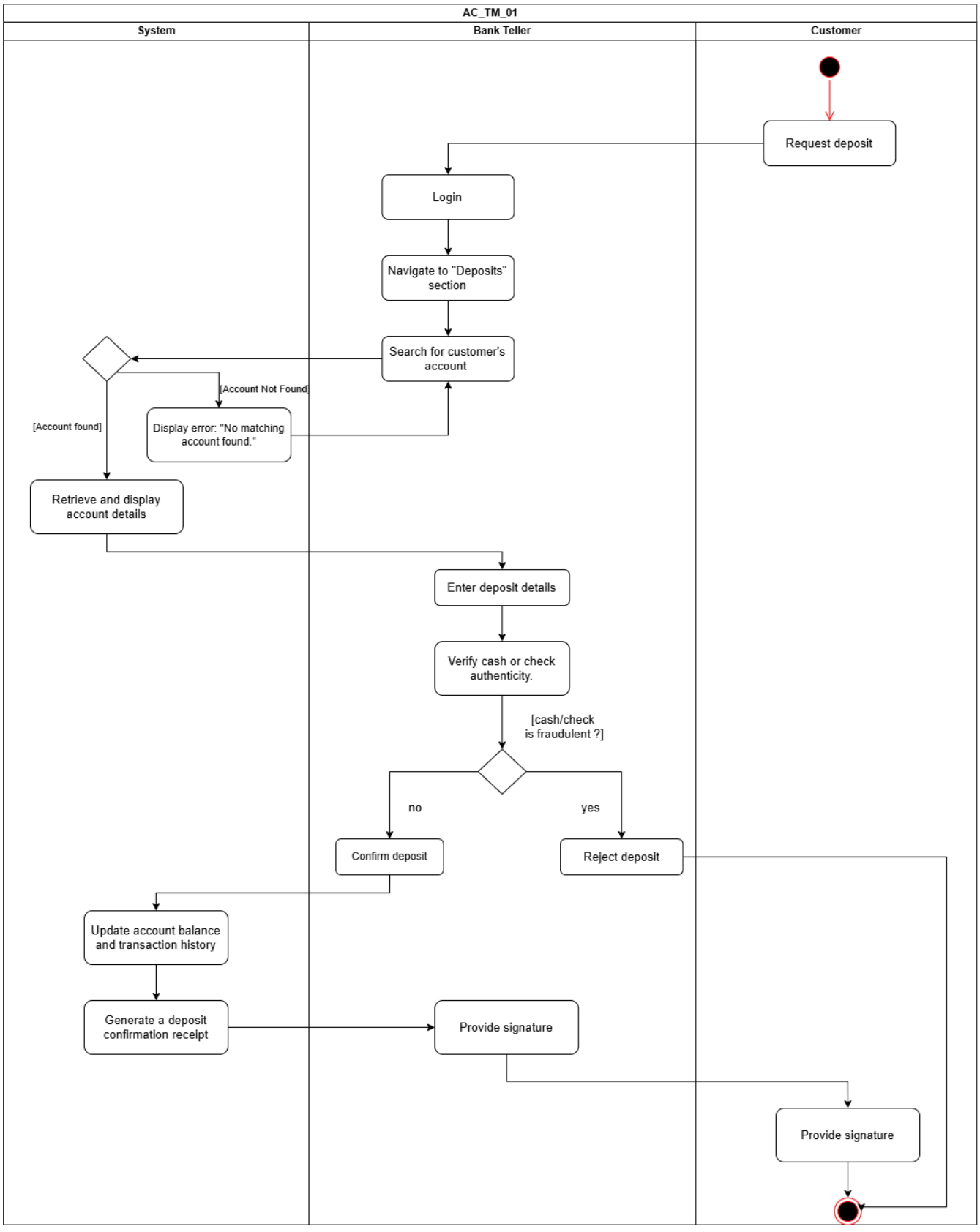

AC_UM_02 Login
Bank Employee / Customer
System
Enter password and username
[valid credentials]
[invalid credentials]
Displays "Invalid Credentials"
Complete multi-factor authentication
[fails]
[attempts <= 4]
[successful]
[attempts > 4]
Lock the account
Grants access

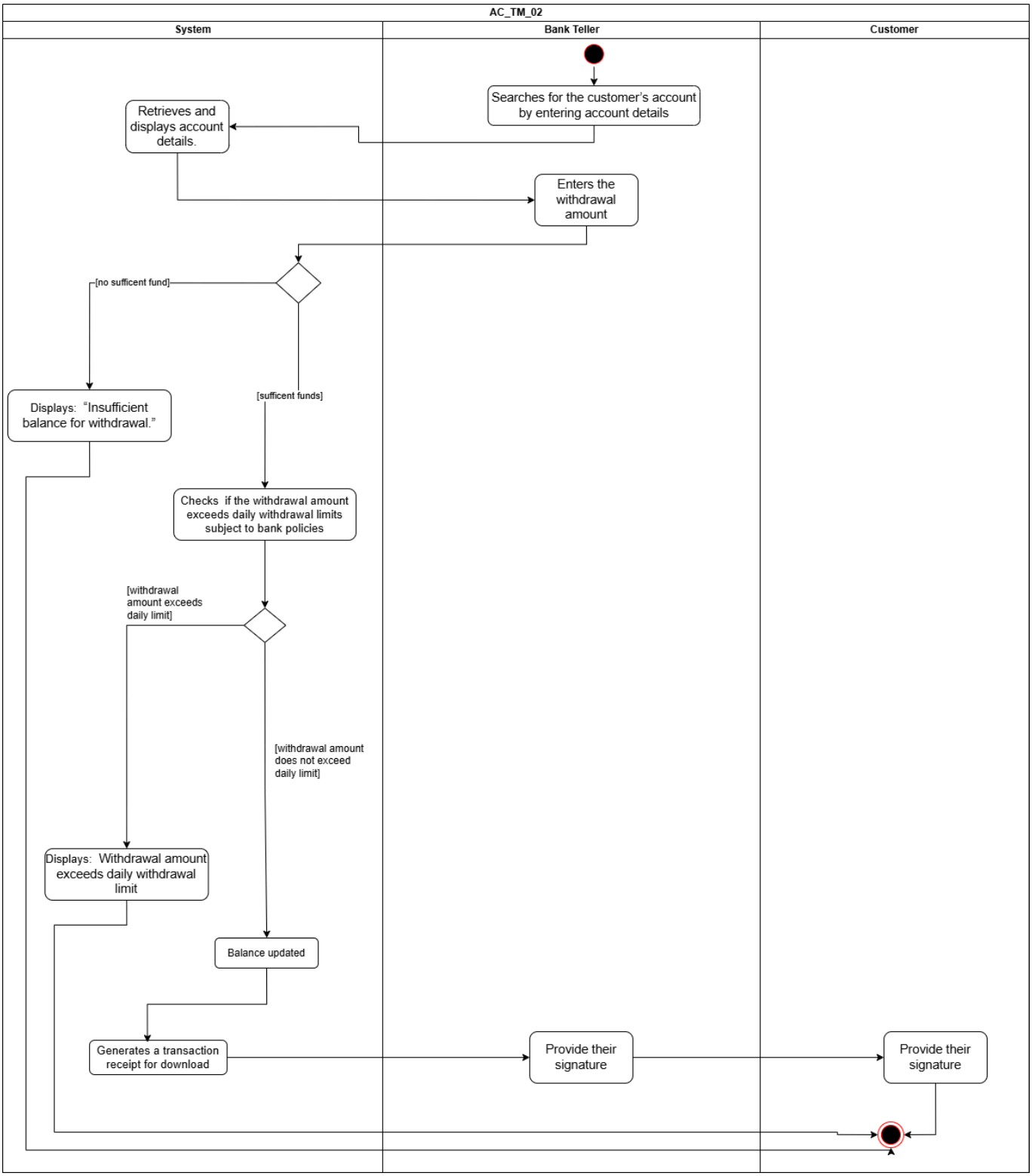

AC_UM_03 Create Online Account
Customer
System
Provide personal information
[required fields missing ?]
[yes]
[no]
Validate information
[invalid credentials]
[valid credentials]
Display message "Invalid Credentials"
Send verification email
Click verification link
[time > 10 minutes]
[time <= 10 minutes]
Invalidate verification link
Save new account

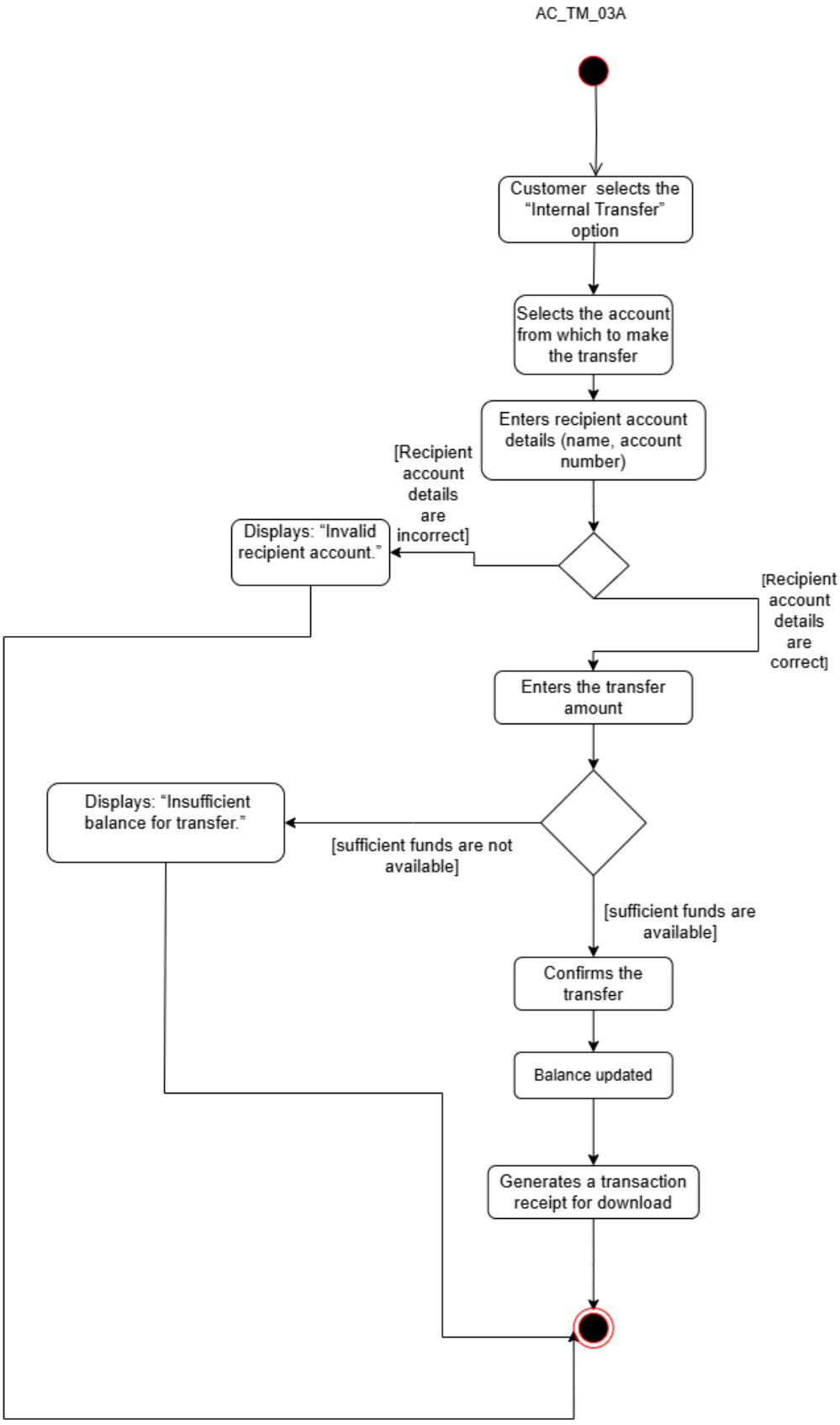

AC_FM_01 View Funds
System
Bank Manager
Select branch/ATM
Display unit details and transaction data
Select filter (type, period)
Display fund details (deposits, withdrawals, transfers, current liquidity)
[Is current liquidity below threshold?]
no
yes
Display alert

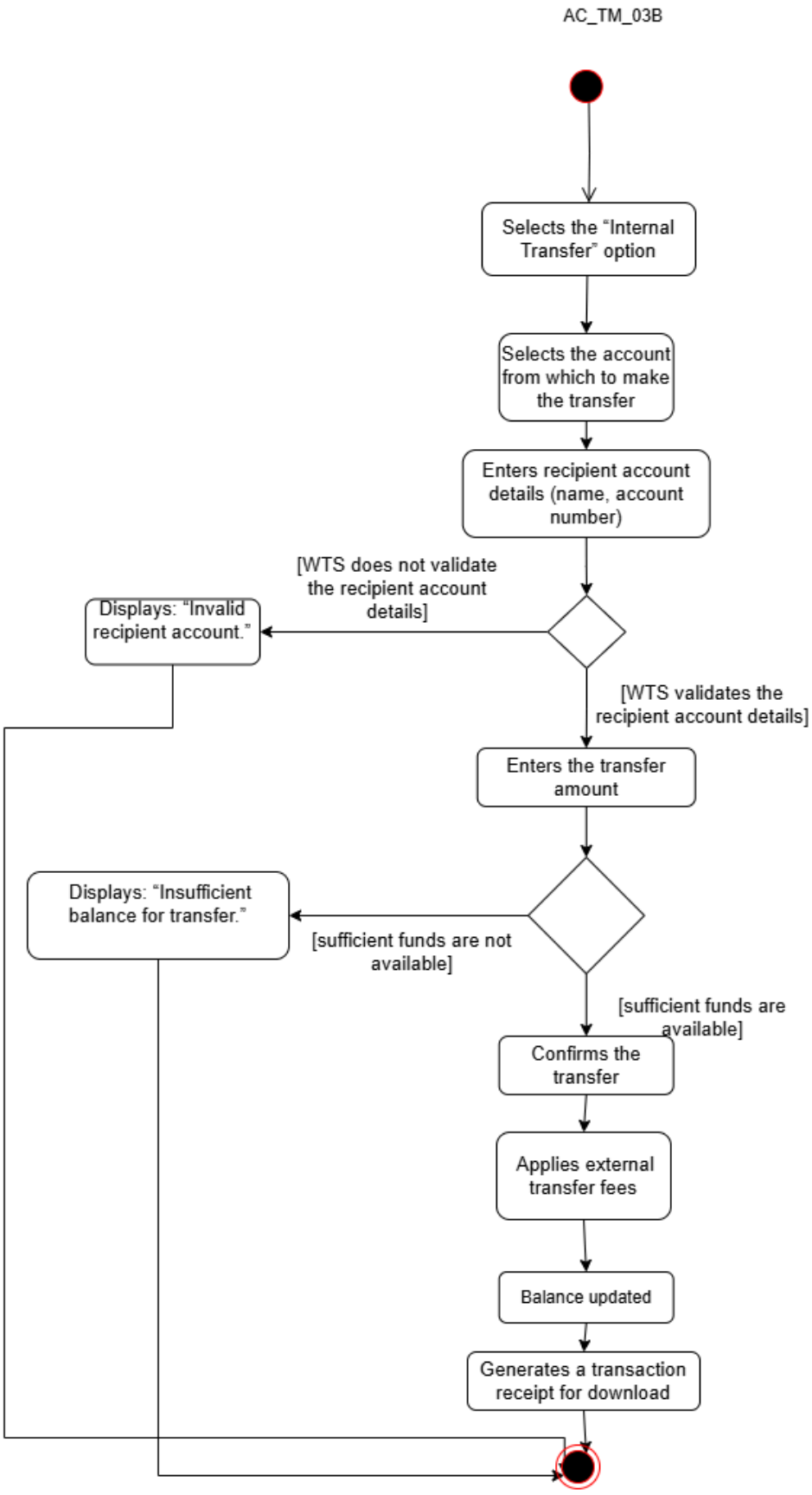

AC_FM_02 Allocate Funds
System
Bank Manager
Select branch/ATM
Enter amount to be allocated
[Is amount entered valid?]
no
yes
Confirm transfer
Display invalid amount error
Update unit's liquidity
Generate allocation document
Provide signature

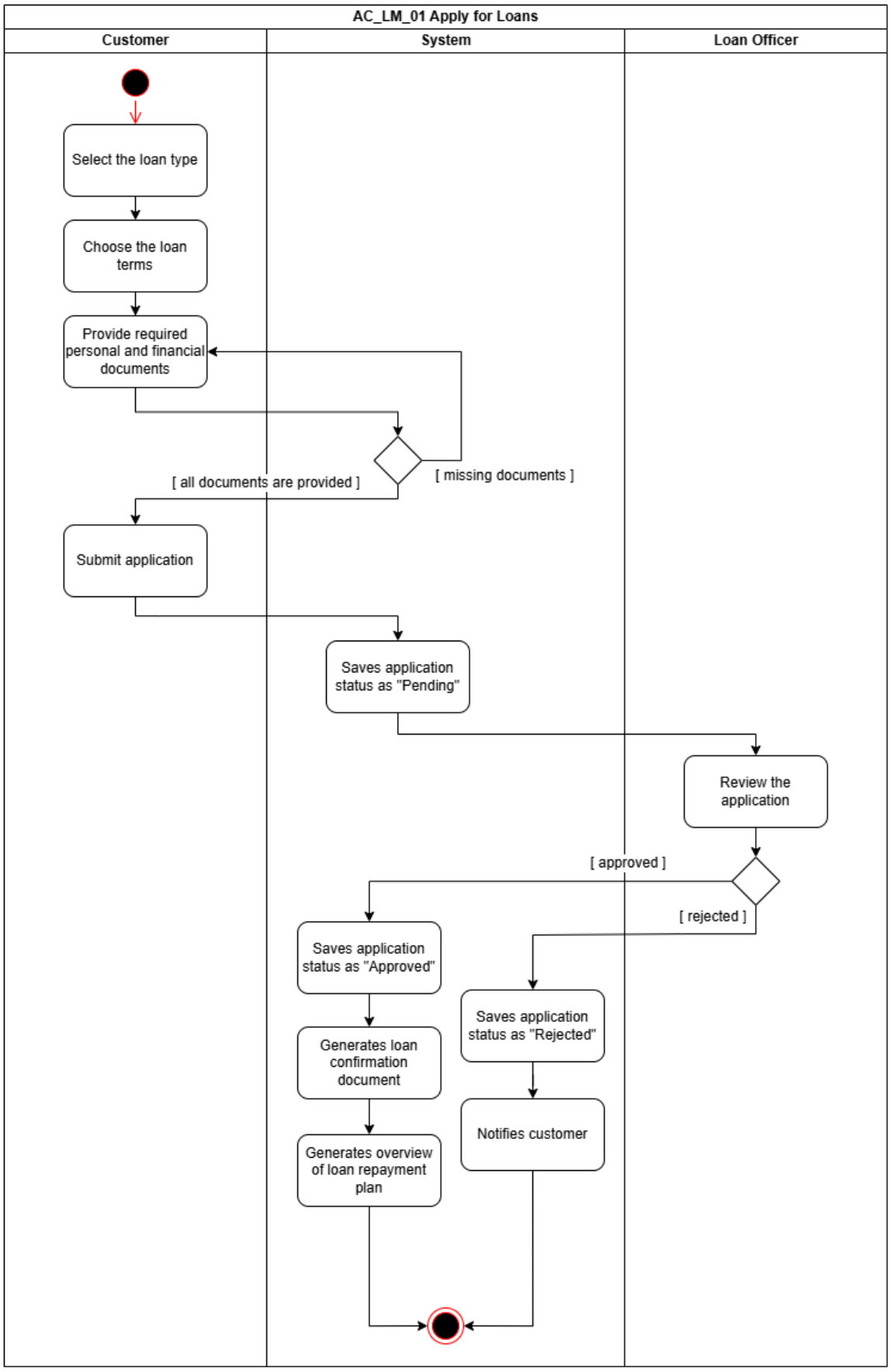

AC_AM_01 Open Bank Account
System
Bank Teller
Customer
Select "Open new account"
Select account type, currency, interest rate, maintenace fees
Enter customer details
[New Customer?]
yes
no
Save customer details
[Valid information?]
[Succesful]
[Unsuccesful]
Create bank account with unique account number
Display account details
Generate account opening document
Provide signature
Provide signature
Apply periodic interest and/or maintenance fees

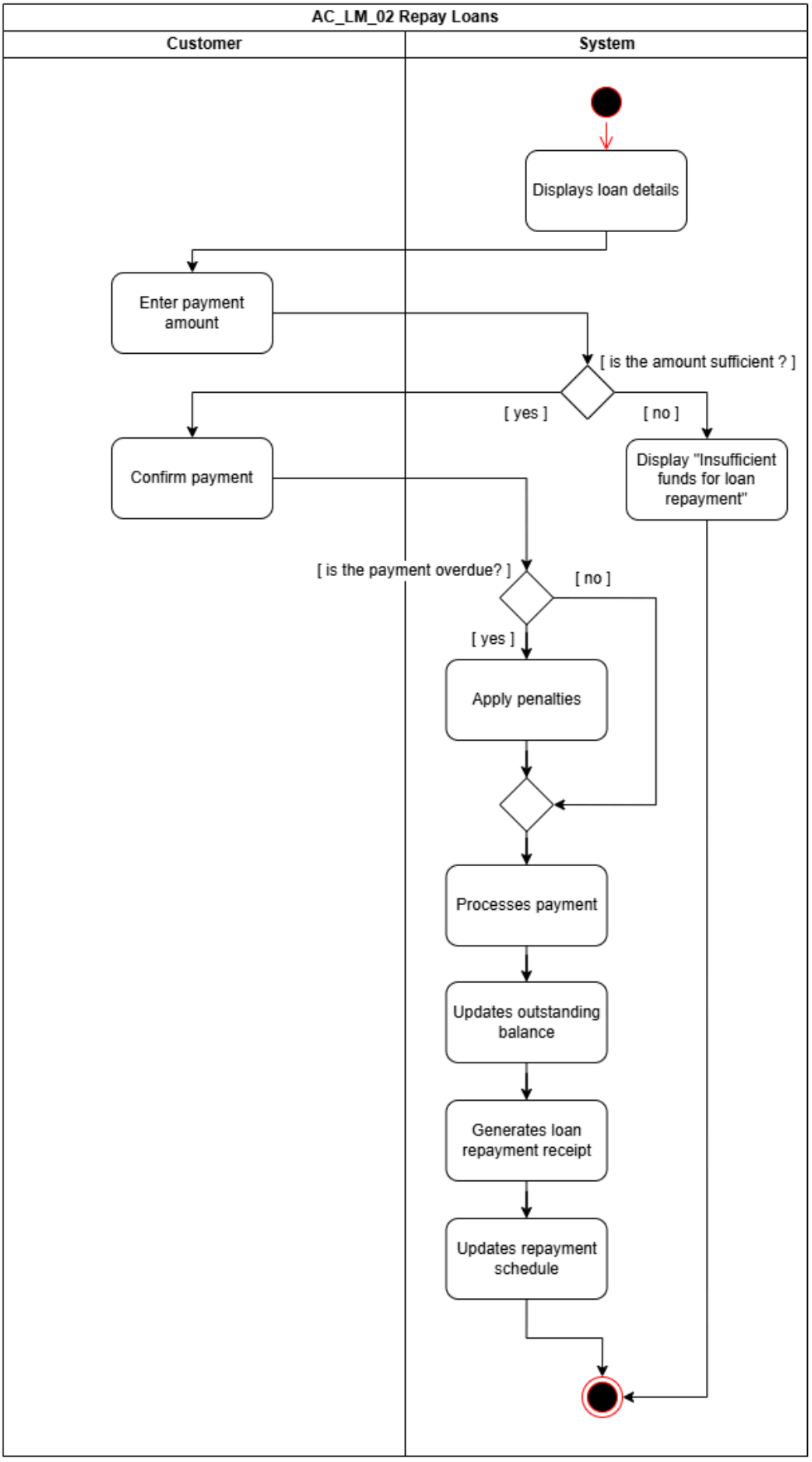

AC_TM_01
System
Bank Teller
Customer
Request deposit
Login
Navigate to "Deposits" section
Search for customer's account
[Account Not Found]
Display error: "No matching account found."
[Account found]
Retrieve and display account details
Enter deposit details
Verify cash or check authenticity.
[cash/check is fraudulent ?]
no
yes
Confirm deposit
Reject deposit
Update account balance and transaction history
Generate a deposit confirmation receipt
Provide signature
Provide signature

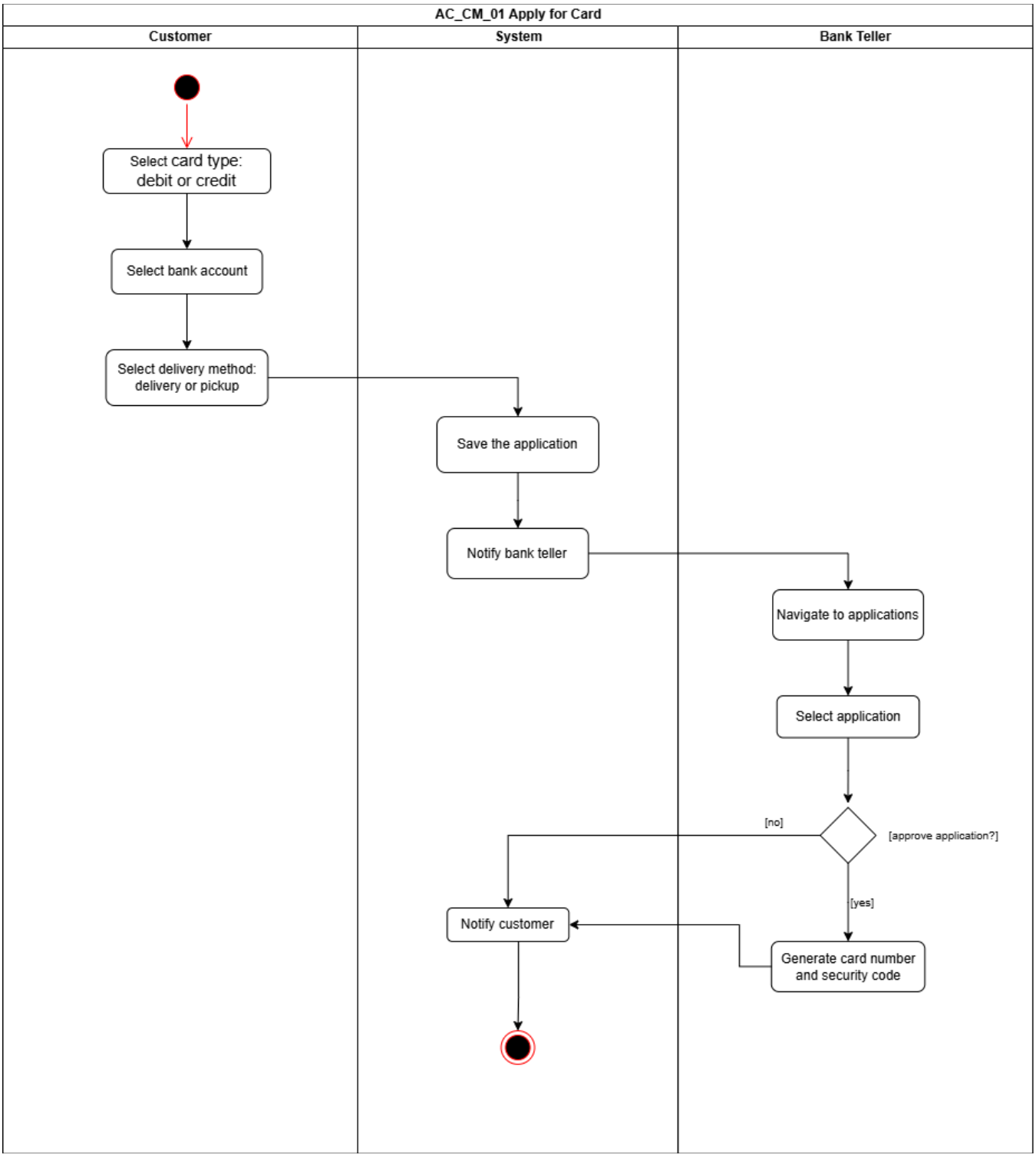

AC_TM_02
System
Bank Teller
Customer
Searches for the customer's account by entering account details
Retrieves and displays account details.
Enters the withdrawal amount
[no sufficent fund]
[sufficent funds]
Displays: "Insufficient balance for withdrawal."
Checks if the withdrawal amount exceeds daily withdrawal limits subject to bank policies
[withdrawal amount exceeds daily limit]
[withdrawal amount does not exceed daily limit]
Displays: Withdrawal amount exceeds daily withdrawal limit
Balance updated
Generates a transaction receipt for download
Provide their signature
Provide their signature

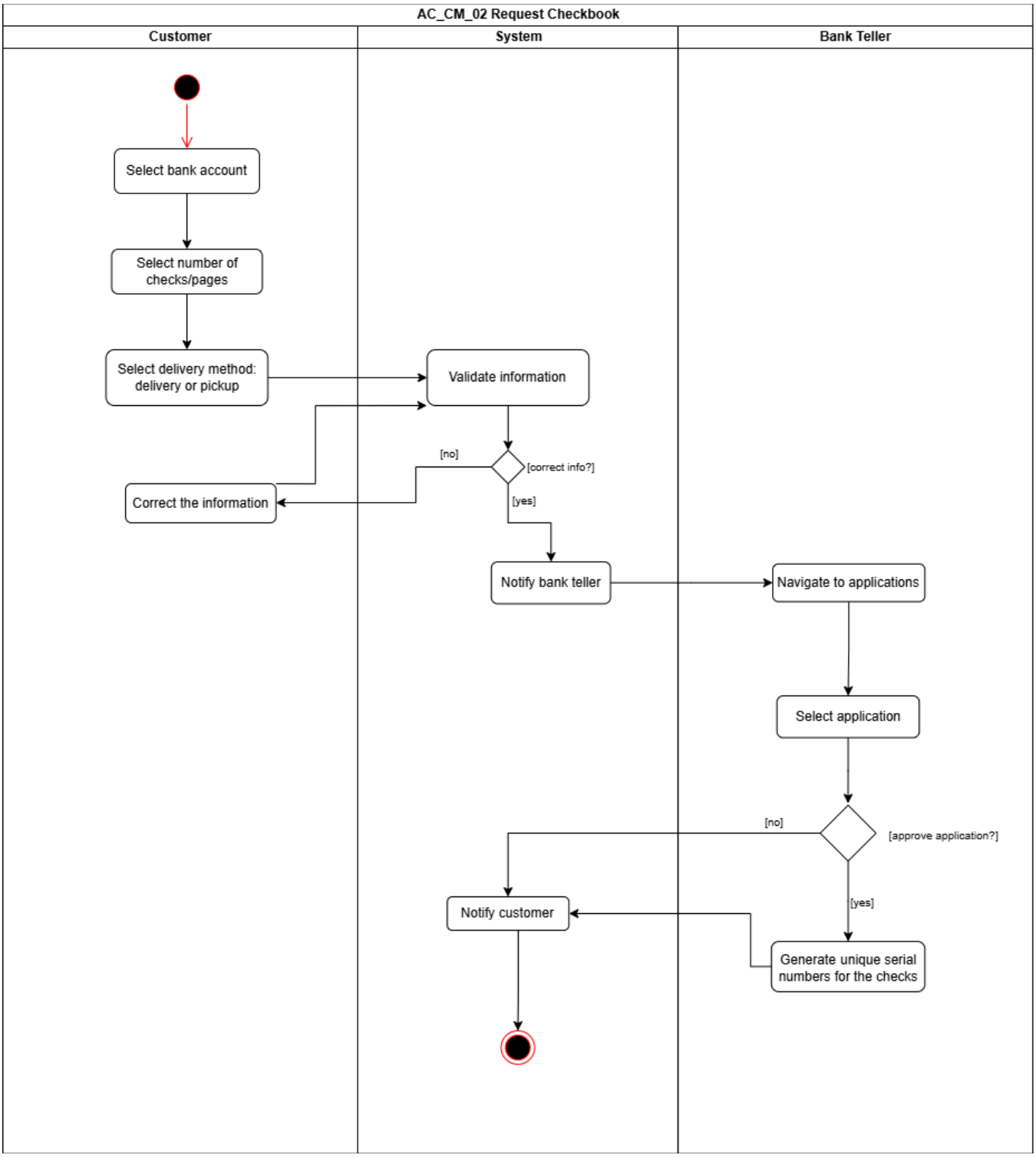

AC_TM_03A
Customer selects the "Internal Transfer" option
Selects the account from which to make the transfer
Enters recipient account details (name, account number)
[Recipient account details are incorrect]
Displays: "Invalid recipient account."
[Recipient account details are correct]
Enters the transfer amount
Displays: "Insufficient balance for transfer."
[sufficient funds are not available]
[sufficient funds are available]
Confirms the transfer
Balance updated
Generates a transaction receipt for download

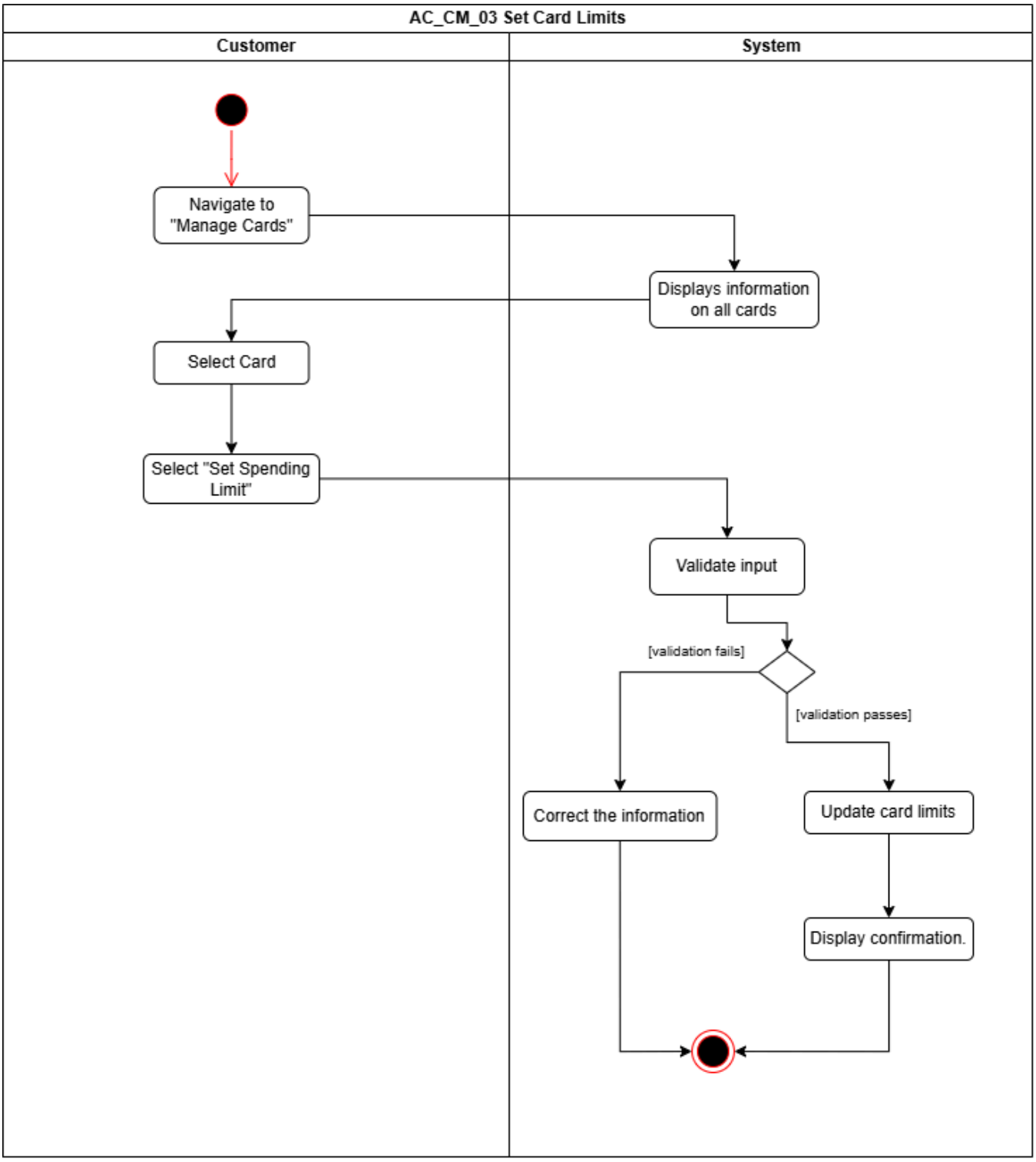

AC_TM_03B
Selects the “Internal Transfer” option
Selects the account from which to make the transfer
Enters recipient account details (name, account number)
[WTS does not validate the recipient account details]
Displays: “Invalid recipient account.”
[WTS validates the recipient account details]
Enters the transfer amount
Displays: “Insufficient balance for transfer.”
[sufficient funds are not available]
[sufficient funds are available]
Confirms the transfer
Applies external transfer fees
Balance updated
Generates a transaction receipt for download

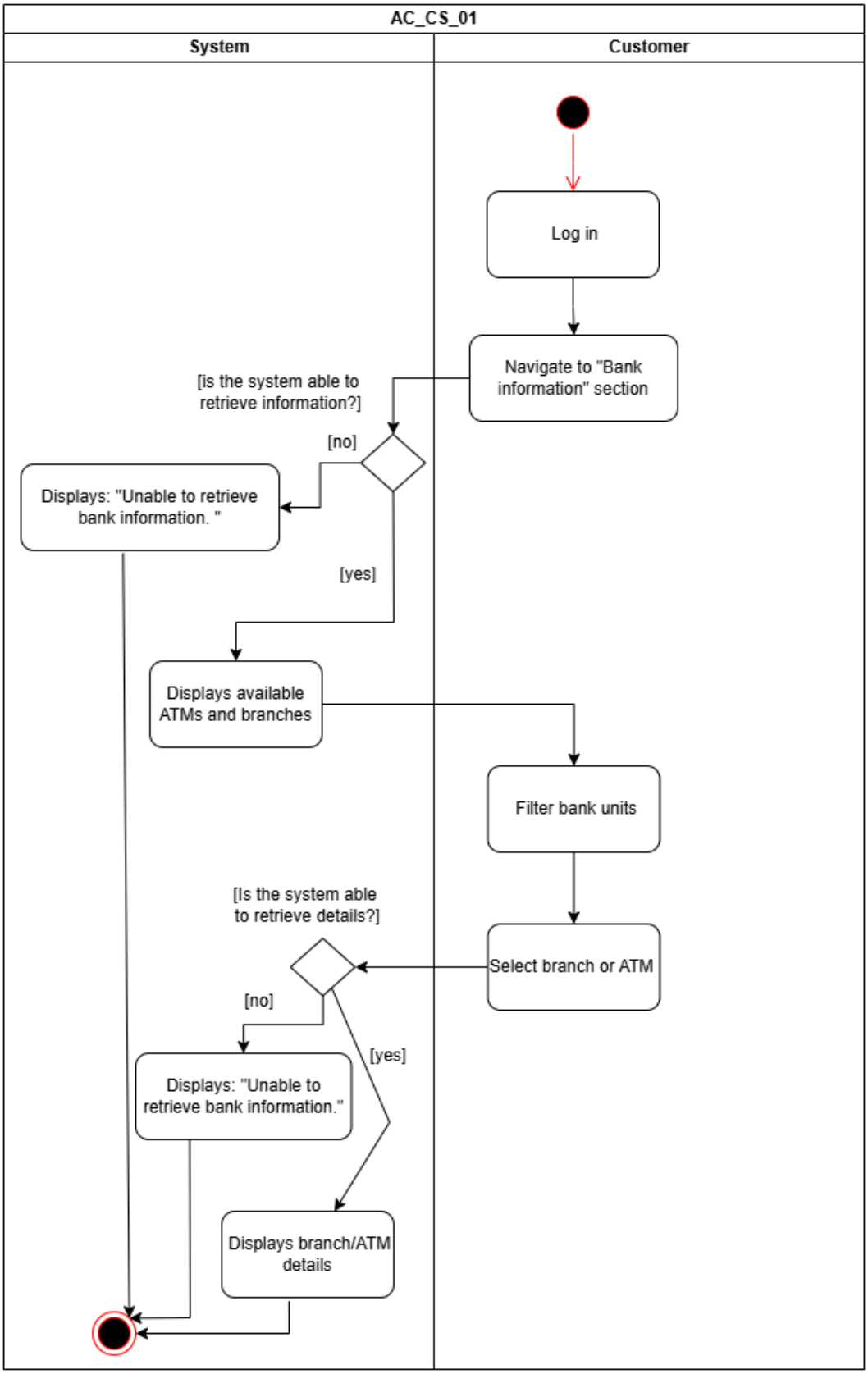

AC_LM_01 Apply for Loans
Customer
System
Loan Officer
Select the loan type
Choose the loan terms
Provide required personal and financial documents
[missing documents]
[all documents are provided]
Submit application
Saves application status as "Pending"
Review the application
[approved]
[rejected]
Saves application status as "Approved"
Generates loan confirmation document
Generates overview of loan repayment plan
Saves application status as "Rejected"
Notifies customer

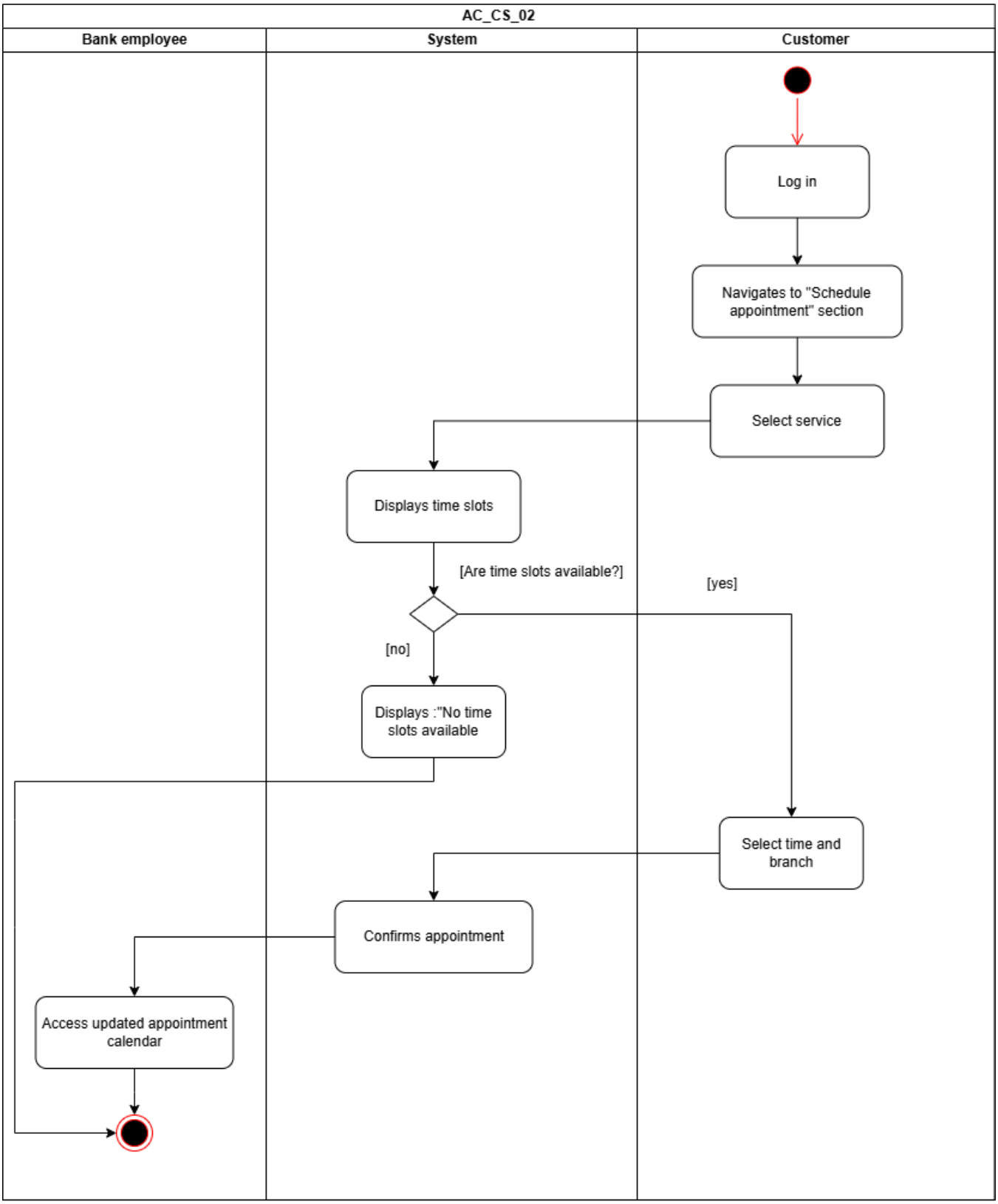

AC_LM_02 Repay Loans
Customer
System
Displays loan details
Enter payment amount
[is the amount sufficient ?]
[yes]
[no]
Confirm payment
Display "Insufficient funds for loan repayment"
[is the payment overdue?]
[no]
[yes]
Apply penalties
Processes payment
Updates outstanding balance
Generates loan repayment receipt
Updates repayment schedule

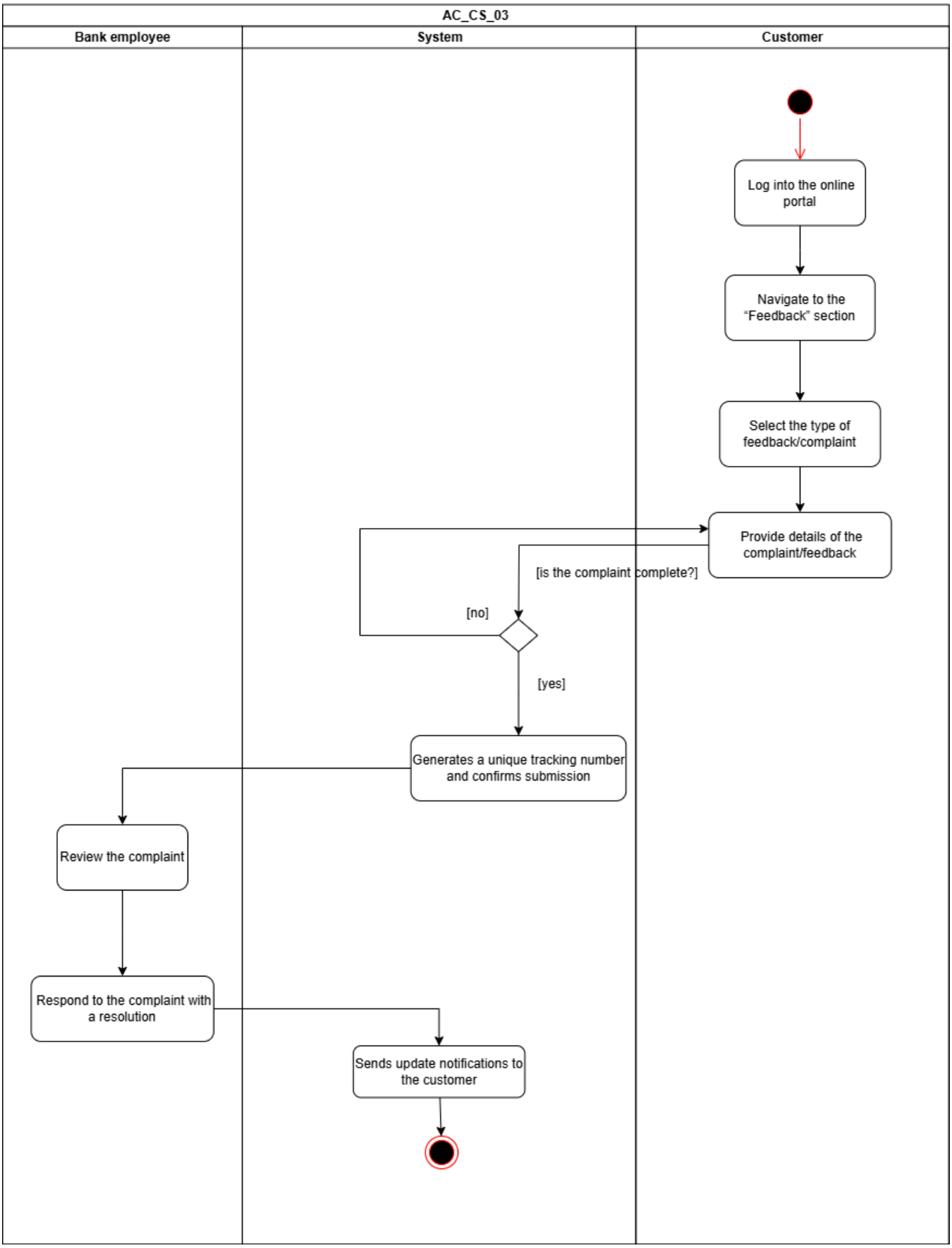

AC_CM_01 Apply for Card
Customer
System
Bank Teller
Select card type: debit or credit
Select bank account
Select delivery method: delivery or pickup
Save the application
Notify bank teller
Navigate to applications
Select application
[no]
[approve application?]
[yes]
Notify customer
Generate card number and security code

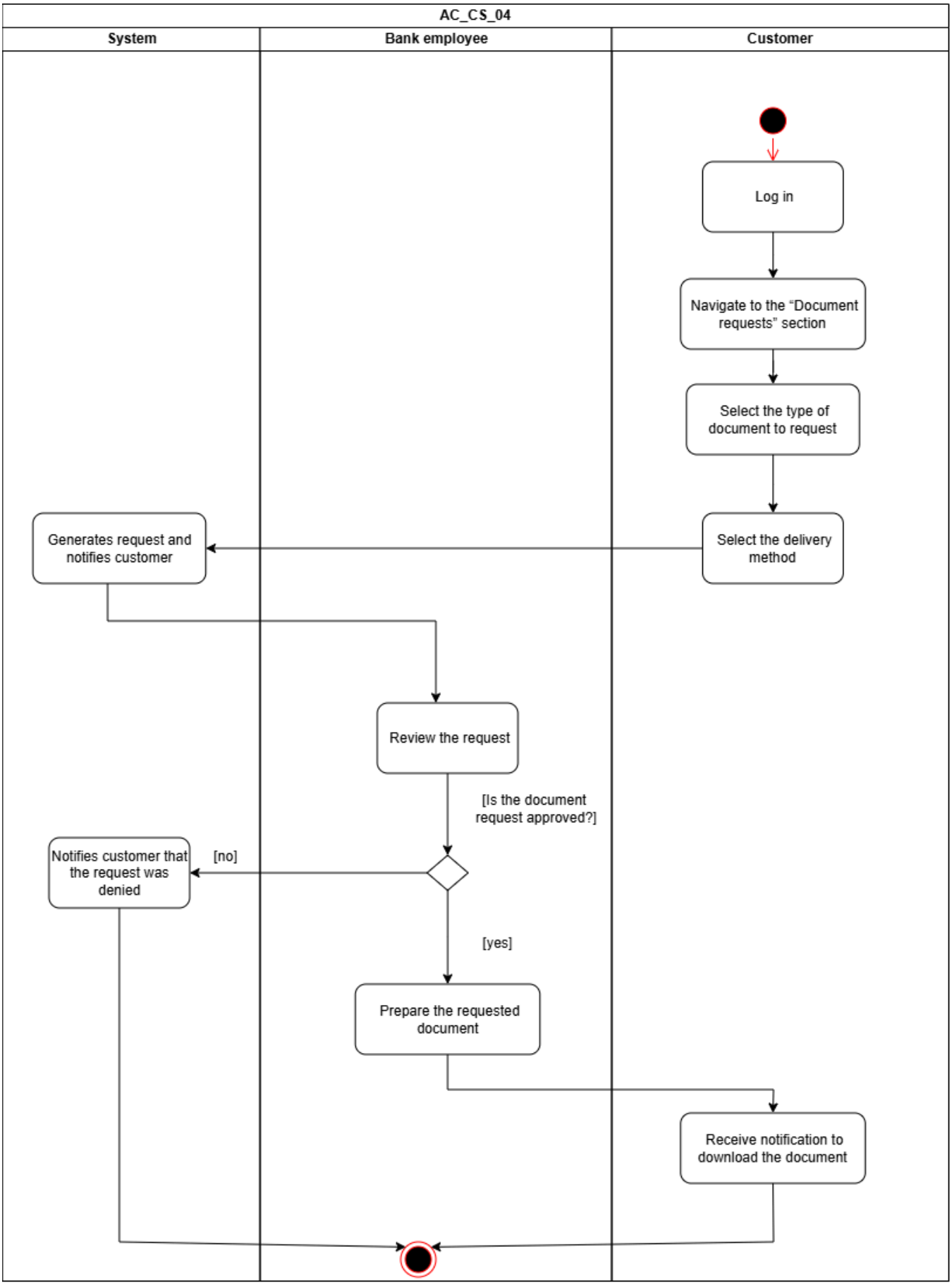

AC_CM_02 Request Checkbook
Customer
System
Bank Teller
Select bank account
Select number of checks/pages
Select delivery method: delivery or pickup
Validate information
[no]
[correct info?]
[yes]
Correct the information
Notify bank teller
Navigate to applications
Select application
[no]
[approve application?]
[yes]
Notify customer
Generate unique serial numbers for the checks

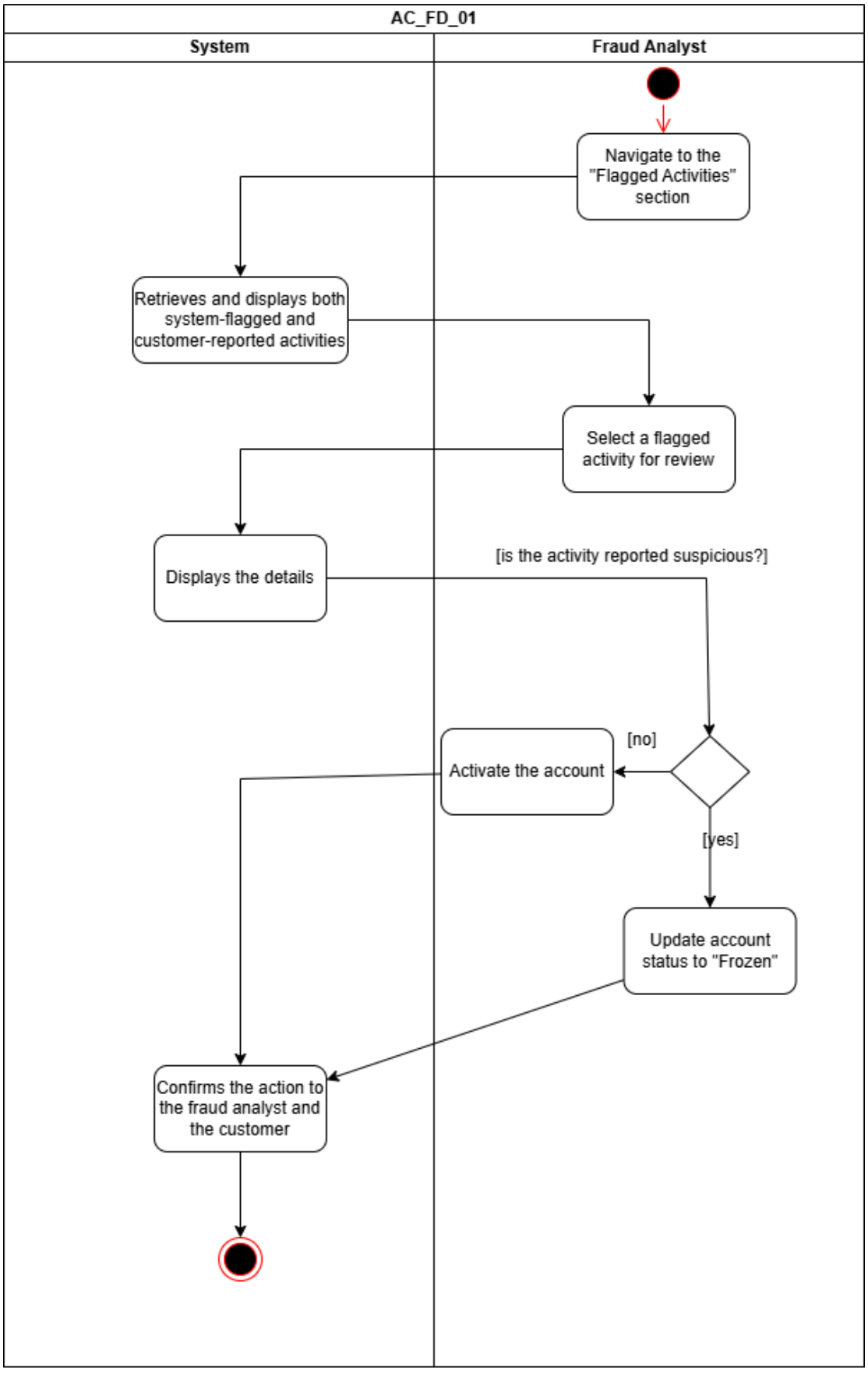

AC_CM_03 Set Card Limits
Customer
System
Navigate to "Manage Cards"
Displays information on all cards
Select Card
Select "Set Spending Limit"
Validate input
[validation fails]
[validation passes]
Correct the information
Update card limits
Display confirmation.

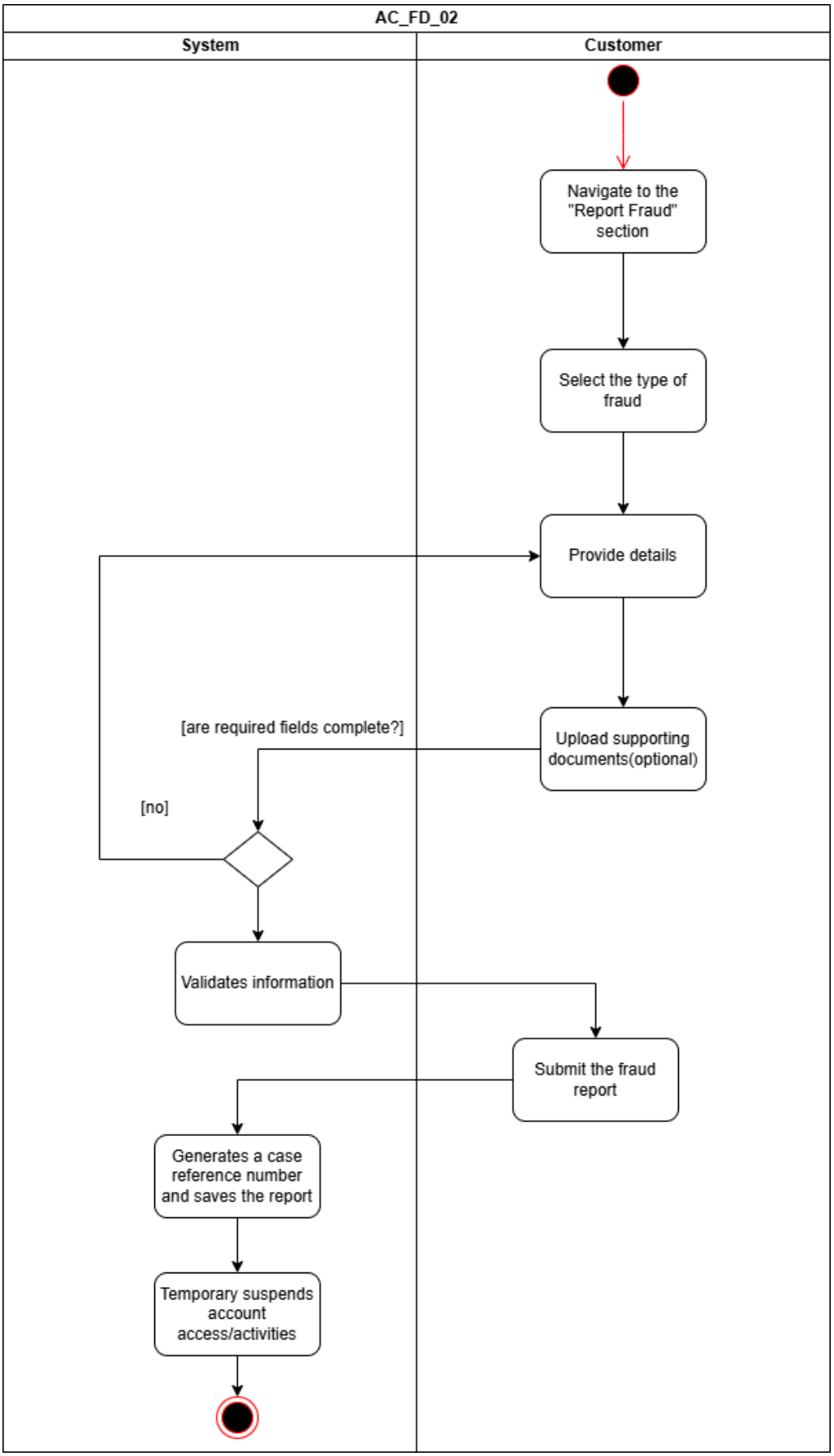

AC_CS_01
System
Customer
Log in
Navigate to "Bank information" section
[is the system able to retrieve information?]
[no]
Displays: "Unable to retrieve bank information. "
[yes]
Displays available ATMs and branches
Filter bank units
[Is the system able to retrieve details?]
Select branch or ATM
[no]
[yes]
Displays: "Unable to retrieve bank information."
Displays branch/ATM details


AC_CS_02
Bank employee
System
Customer
Log in
Navigates to "Schedule appointment" section
Select service
Displays time slots
[Are time slots available?]
[yes]
[no]
Displays :"No time slots available
Select time and branch
Confirms appointment
Access updated appointment calendar

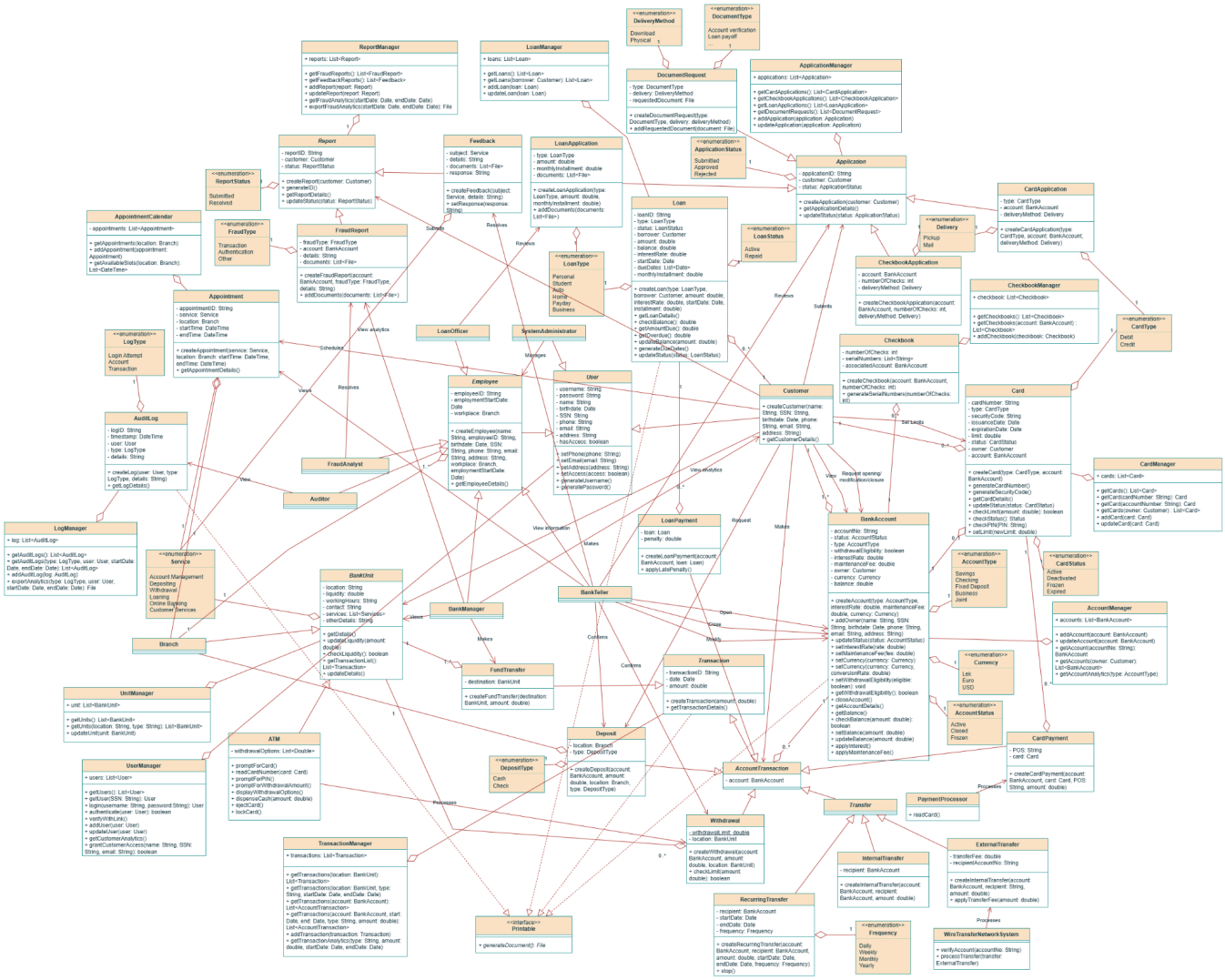

AC_CS_03
Bank employee
System
Customer
Log into the online portal
Navigate to the "Feedback" section
Select the type of feedback/complaint
Provide details of the complaint/feedback
[is the complaint complete?]
[no]
[yes]
Generates a unique tracking number and confirms submission
Review the complaint
Respond to the complaint with a resolution
Sends update notifications to the customer


AC_CS_04
System
Bank employee
Customer
Log in
Navigate to the "Document requests" section
Select the type of document to request
Select the delivery method
Generates request and notifies customer
Review the request
[Is the document request approved?]
[no]
Notifies customer that the request was denied
[yes]
Prepare the requested document
Receive notification to download the document

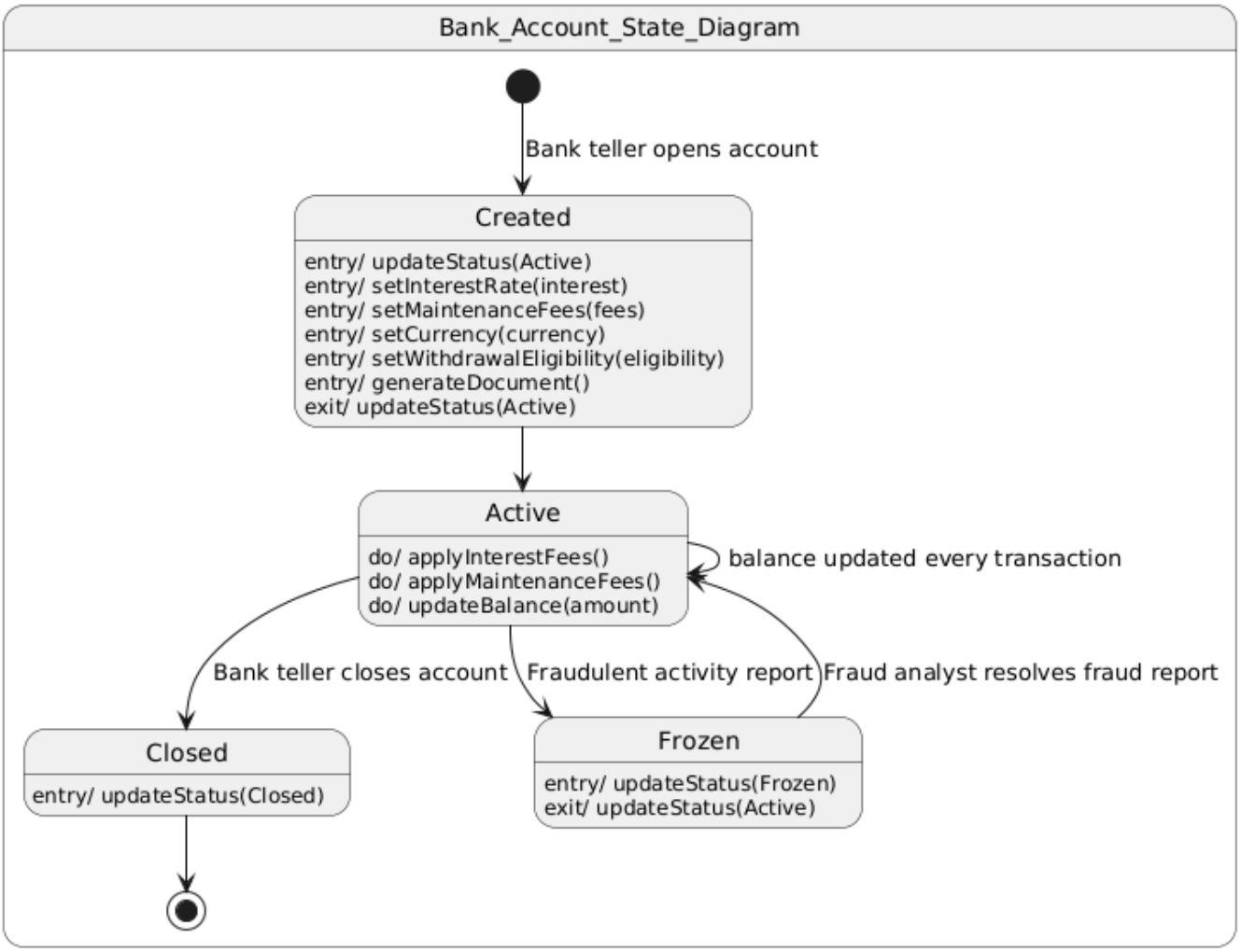

AC_FD_01
System
Fraud Analyst
Navigate to the "Flagged Activities" section
Retrieves and displays both system-flagged and customer-reported activities
Select a flagged activity for review
Displays the details
[is the activity reported suspicious?]
[no]
Activate the account
[yes]
Update account status to "Frozen"
Confirms the action to the fraud analyst and the customer

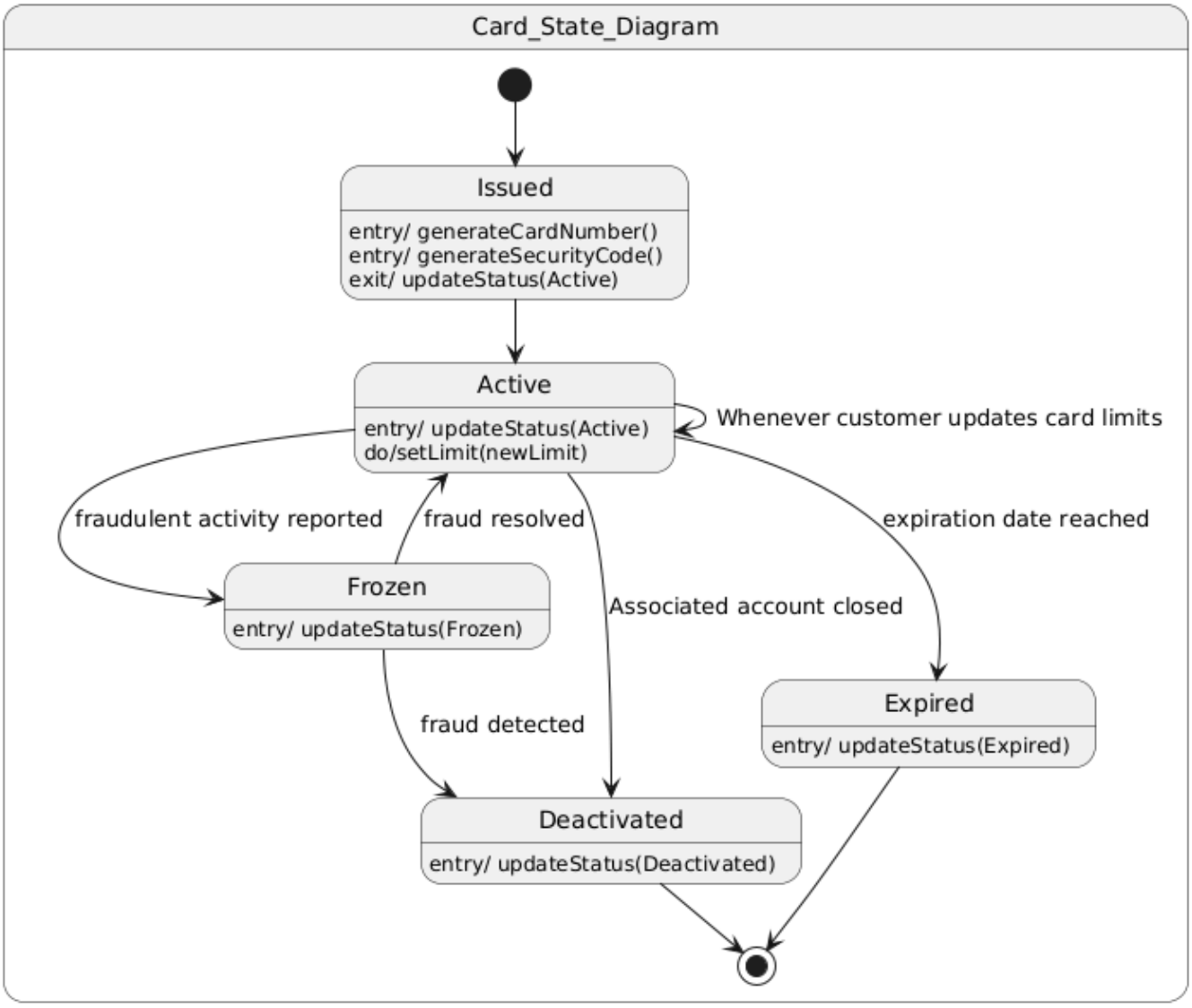

AC_FD_02
System
Customer
Navigate to the "Report Fraud" section
Select the type of fraud
Provide details
Upload supporting documents(optional)
[are required fields complete?]
[no]
Validates information
Submit the fraud report
Generates a case reference number and saves the report
Temporary suspends account access/activities

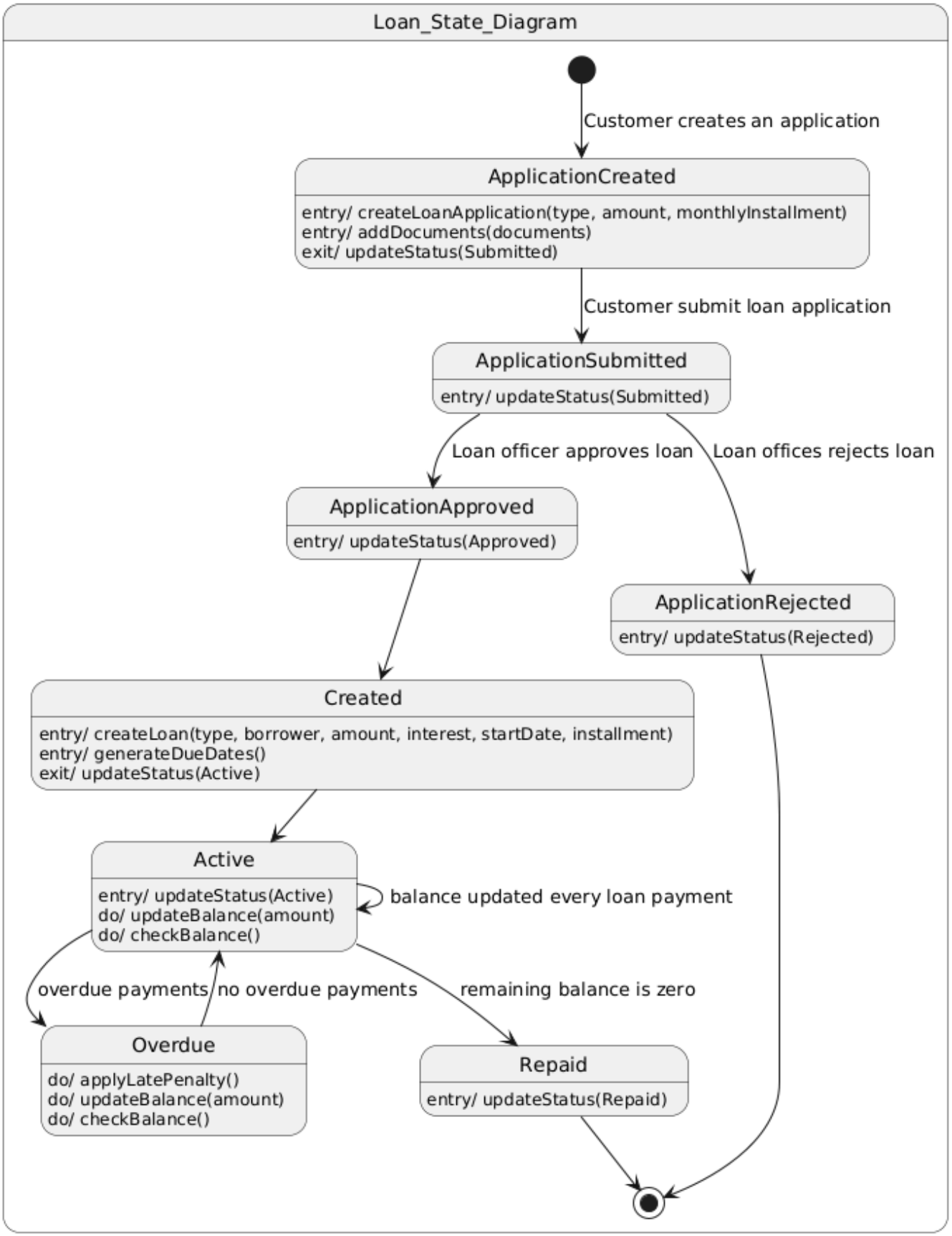
Class Diagram



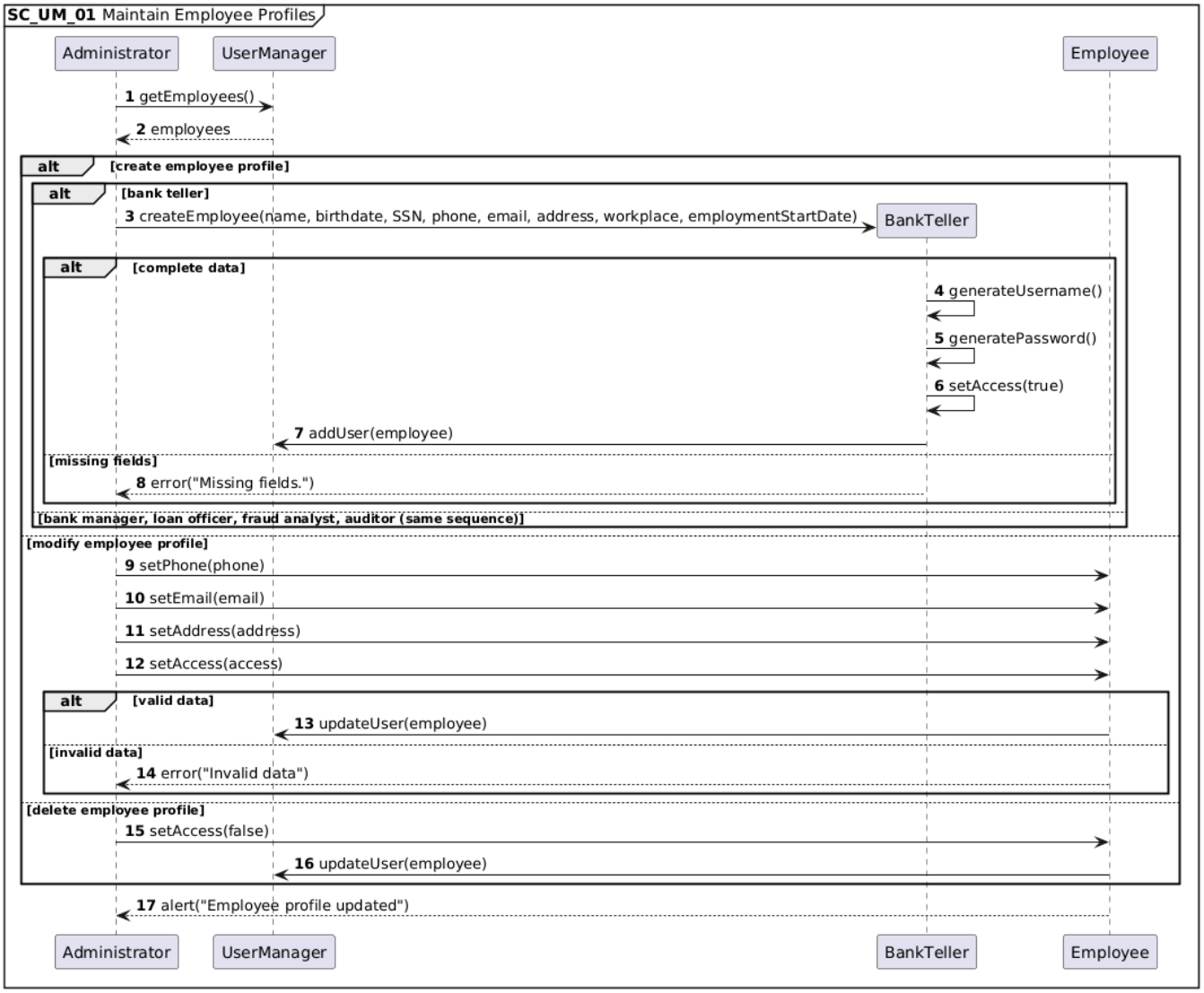
State Diagrams

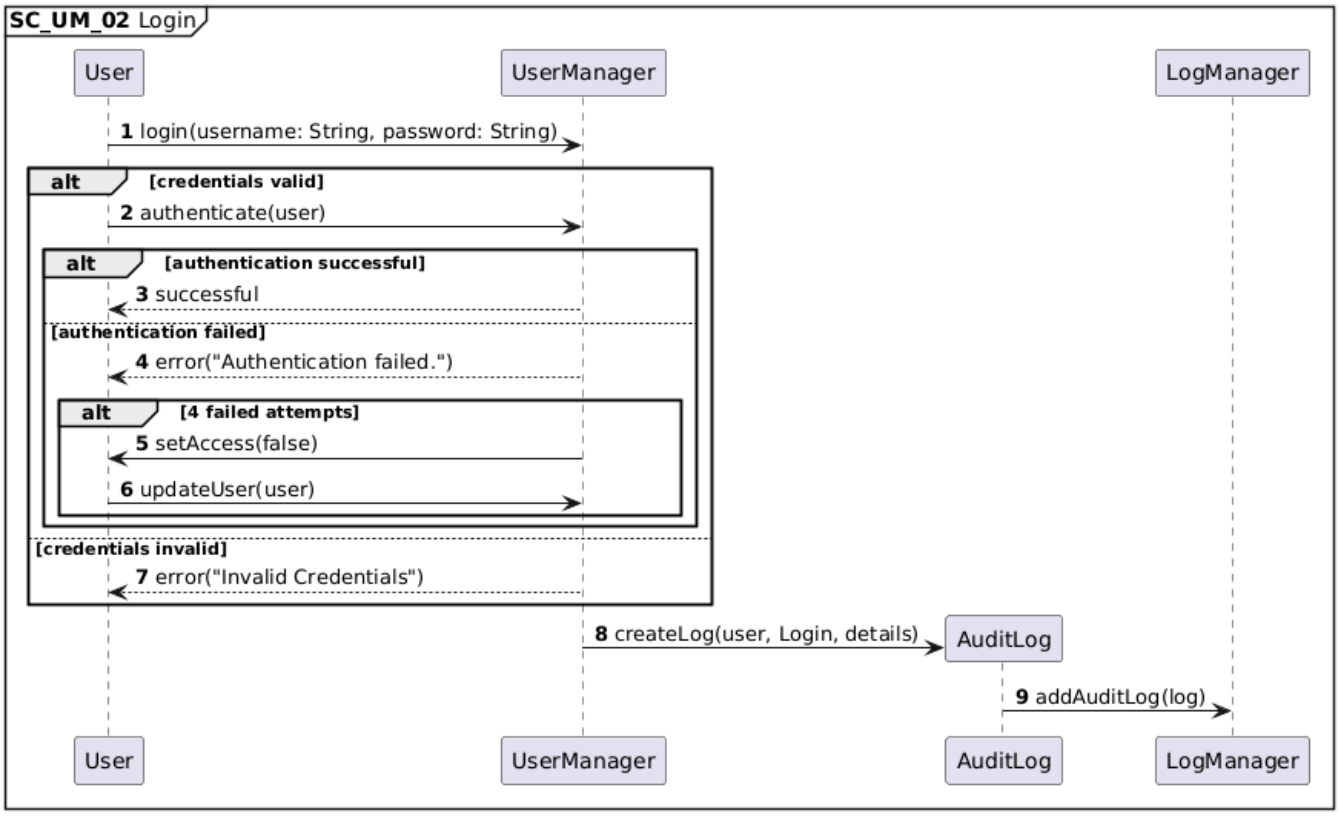


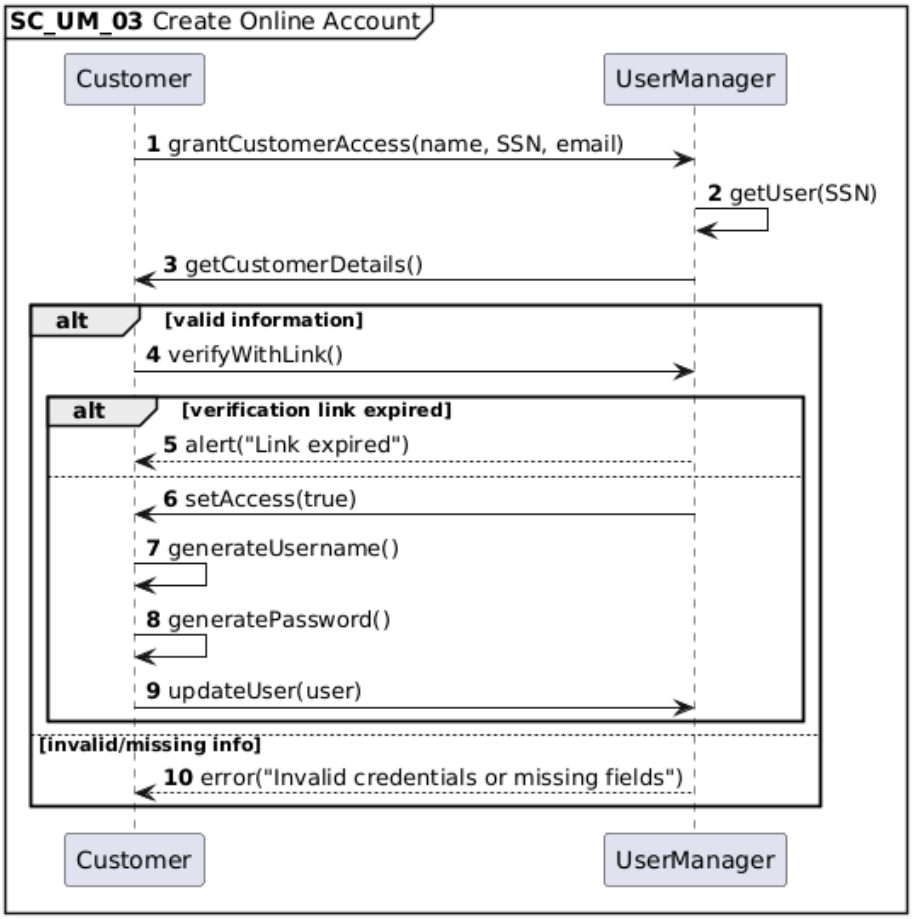

Card_State_Diagram
Issued
entry/ generateCardNumber()
entry/ generateSecurityCode()
exit/ updateStatus(Active)
Active
entry/ updateStatus(Active)
do/setLimit(newLimit)
Whenever customer updates card limits
fraudulent activity reported
fraud resolved
expiration date reached
Frozen
entry/ updateStatus(Frozen)
Associated account closed
fraud detected
Expired
entry/ updateStatus(Expired)
Deactivated
entry/ updateStatus(Deactivated)

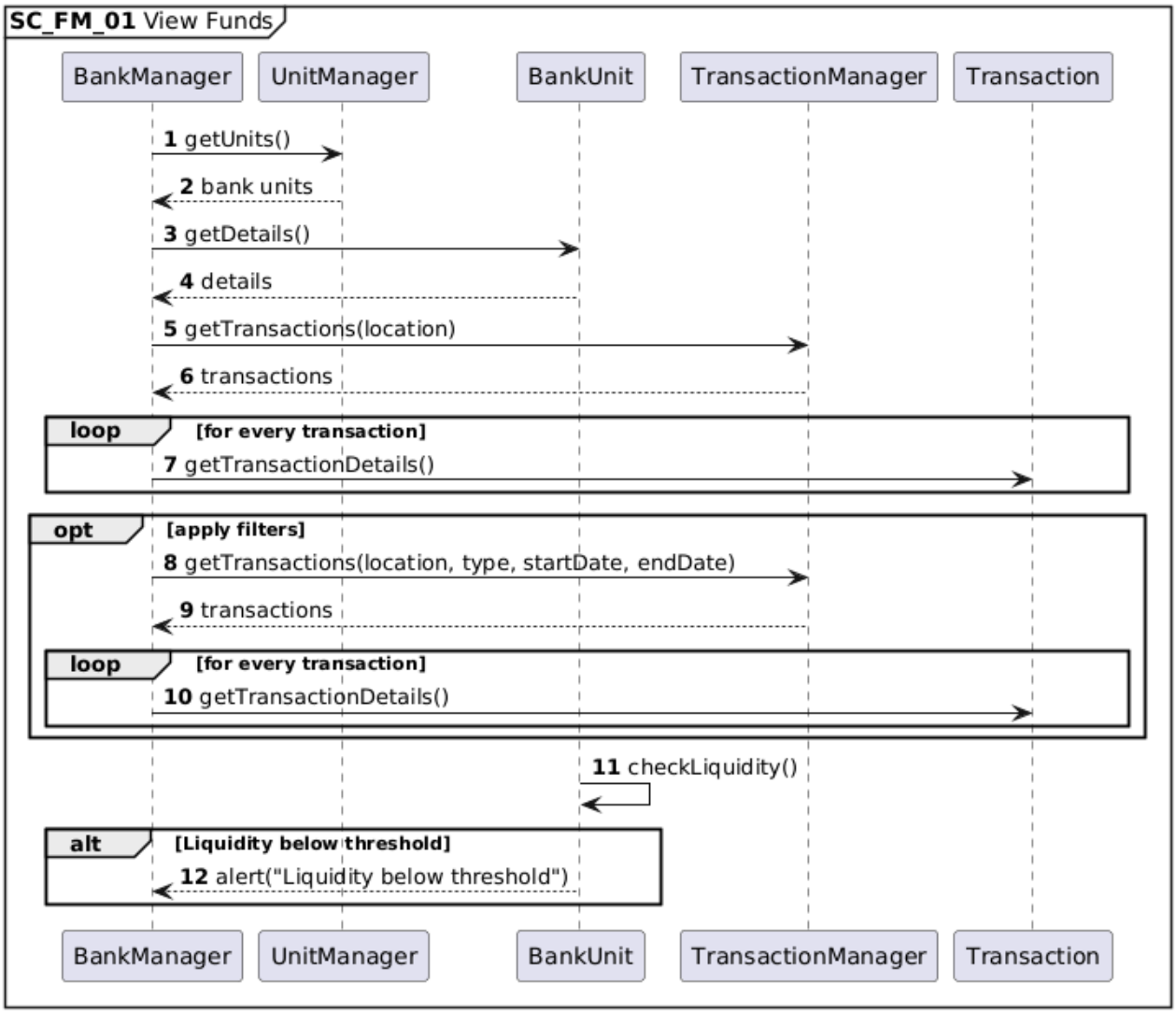

Loan_State_Diagram
Customer creates an application
ApplicationCreated
entry/ createLoanApplication(type, amount, monthlyInstallment)
entry/ addDocuments(documents)
exit/ updateStatus(Submitted)
Customer submit loan application
ApplicationSubmitted
entry/ updateStatus(Submitted)
Loan officer approves loan
Loan offices rejects loan
ApplicationApproved
entry/ updateStatus(Approved)
ApplicationRejected
entry/ updateStatus(Rejected)
Created
entry/ createLoan(type, borrower, amount, interest, startDate, installment)
entry/ generateDueDates()
exit/ updateStatus(Active)
Active
entry/ updateStatus(Active)
do/ updateBalance(amount)
do/ checkBalance()
balance updated every loan payment
overdue payments
no overdue payments
remaining balance is zero
Overdue
do/ applyLatePenalty()
do/ updateBalance(amount)
do/ checkBalance()
Repaid
entry/ updateStatus(Repaid)

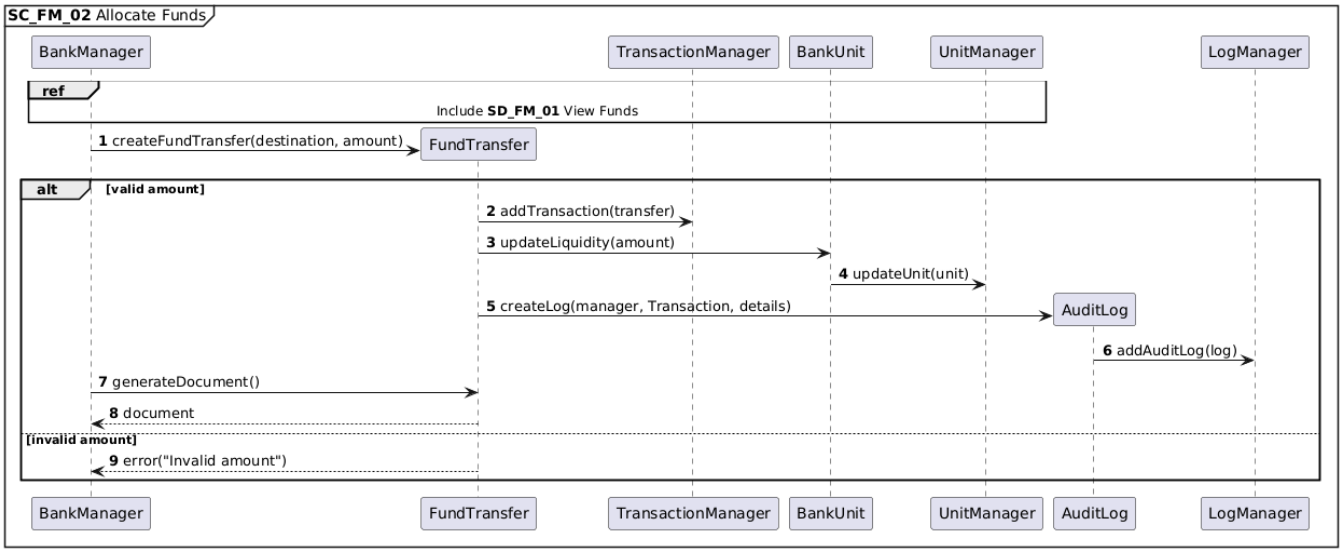
Sequence Diagrams

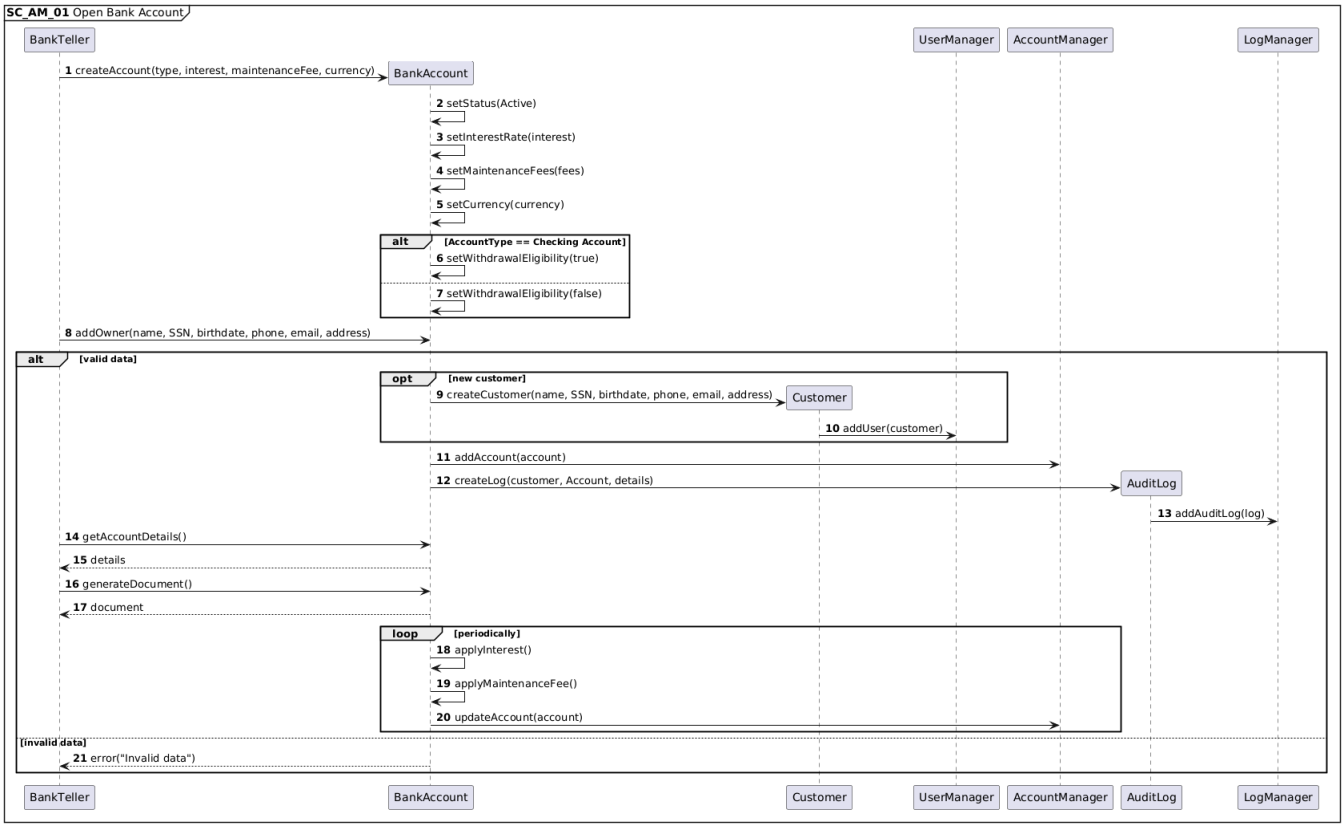


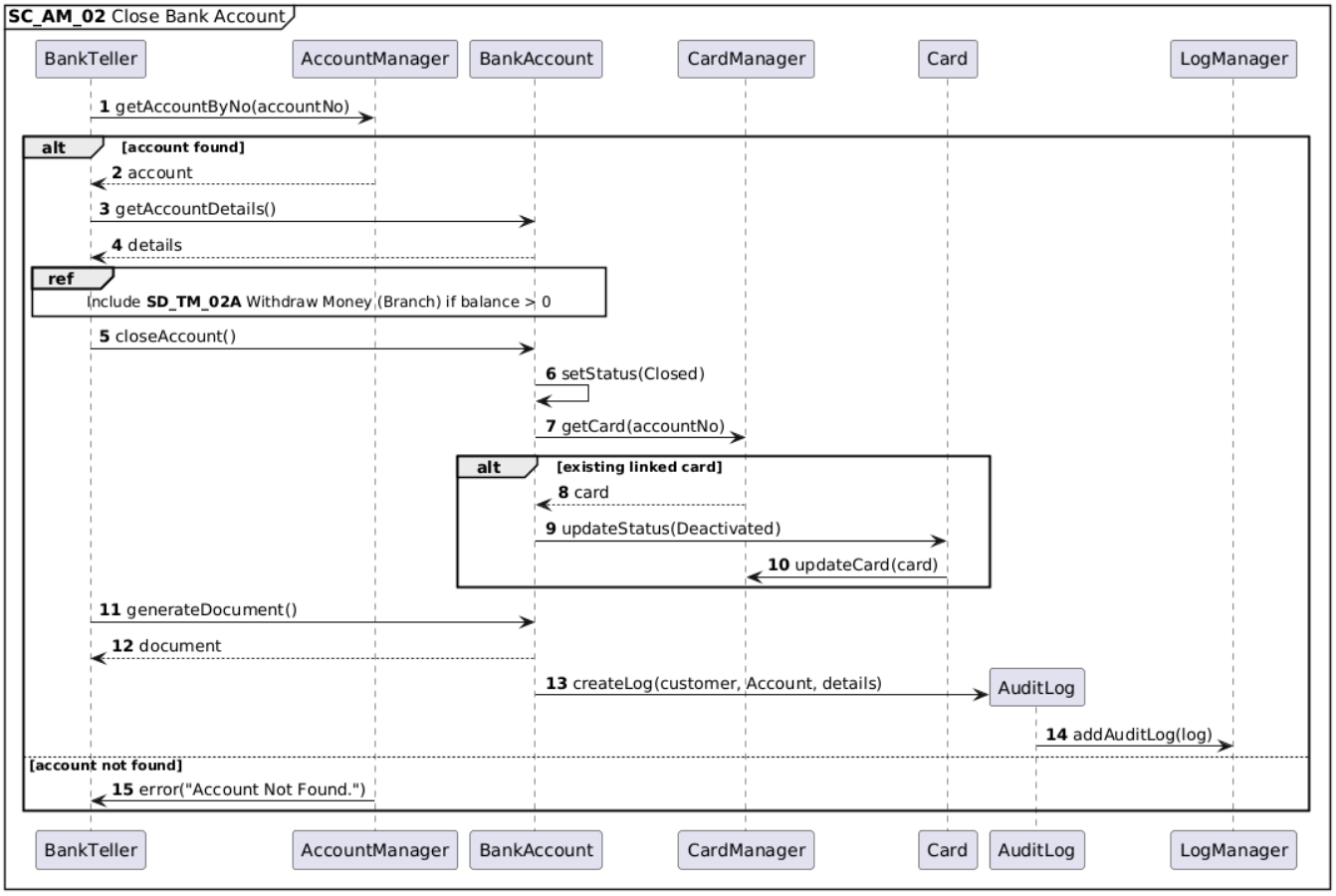

SC_UM_02 Login
User
UserManager
LogManager
1 login(username: String, password: String)
alt [credentials valid]
2 authenticate(user)
alt [authentication successful]
3 successful
[authentication failed]
4 error("Authentication failed.")
alt [4 failed attempts]
5 setAccess(false)
6 updateUser(user)
[credentials invalid]
7 error("Invalid Credentials")
8 createLog(user, Login, details)
AuditLog
9 addAuditLog(log)
User
UserManager
AuditLog
LogManager

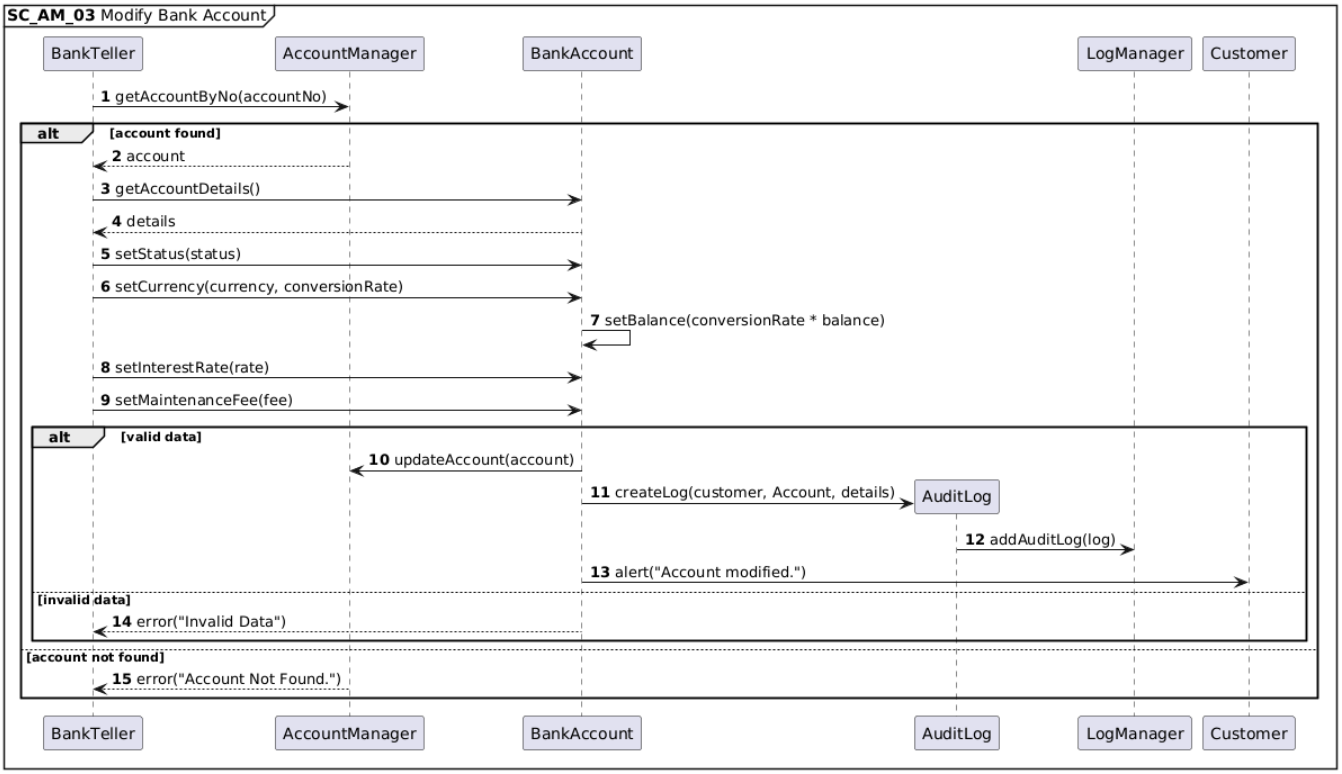

SC_UM_03 Create Online Account
Customer
UserManager
1 grantCustomerAccess(name, SSN, email)
2 getUser(SSN)
3 getCustomerDetails()
alt [valid information]
4 verifyWithLink()
alt [verification link expired]
5 alert("Link expired")
6 setAccess(true)
7 generateUsername()
8 generatePassword()
9 updateUser(user)
[invalid/missing info]
10 error("Invalid credentials or missing fields")
Customer
UserManager

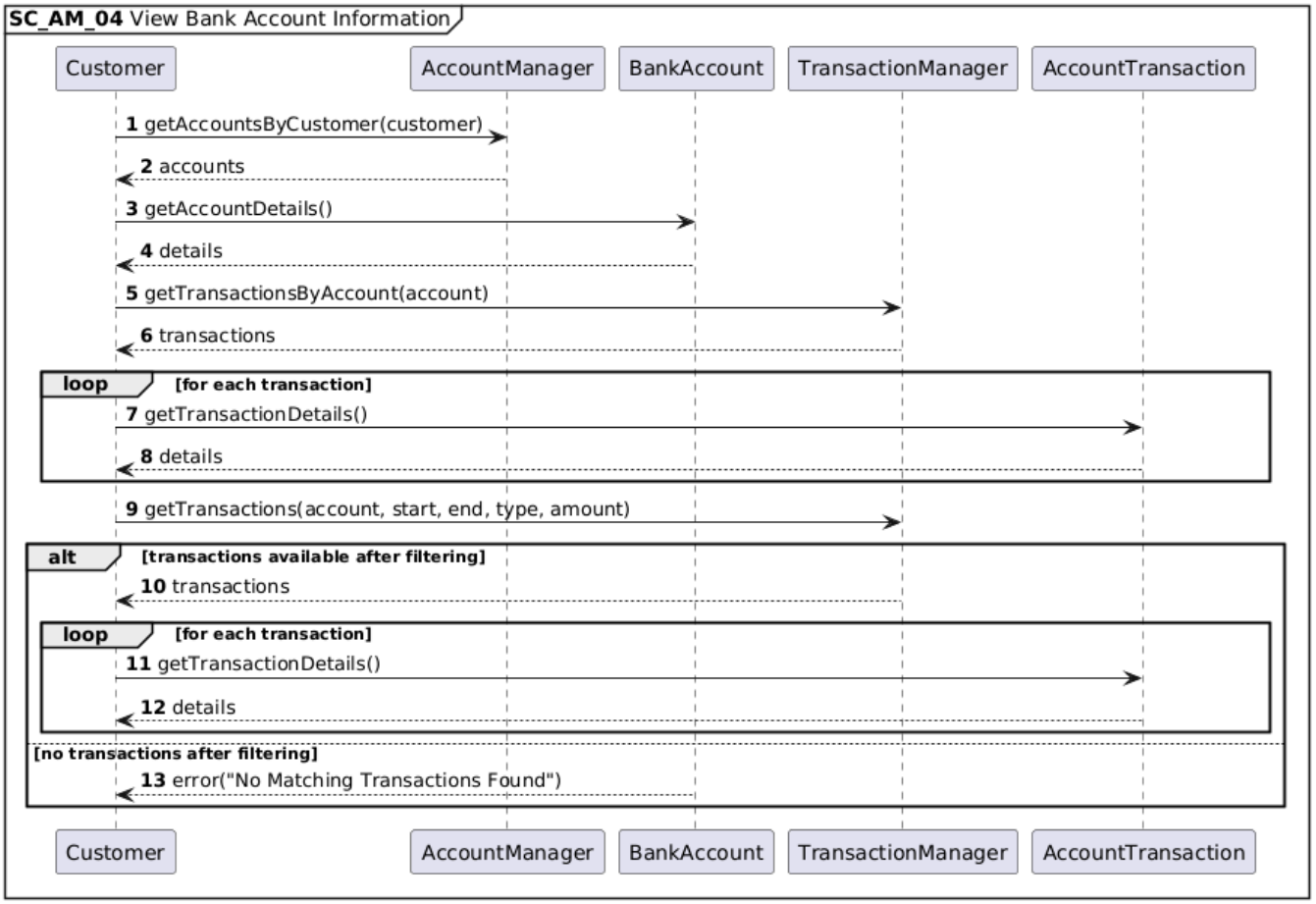

SC_FM_01 View Funds
BankManager
UnitManager
BankUnit
TransactionManager
Transaction
1 getUnits()
2 bank units
3 getDetails()
4 details
5 getTransactions(location)
6 transactions
loop
[for every transaction]
7 getTransactionDetails()
opt
[apply filters]
8 getTransactions(location, type, startDate, endDate)
9 transactions
loop
[for every transaction]
10 getTransactionDetails()
11 checkLiquidity()
alt
[Liquidity below threshold]
12 alert("Liquidity below threshold")
BankManager
UnitManager
BankUnit
TransactionManager
Transaction

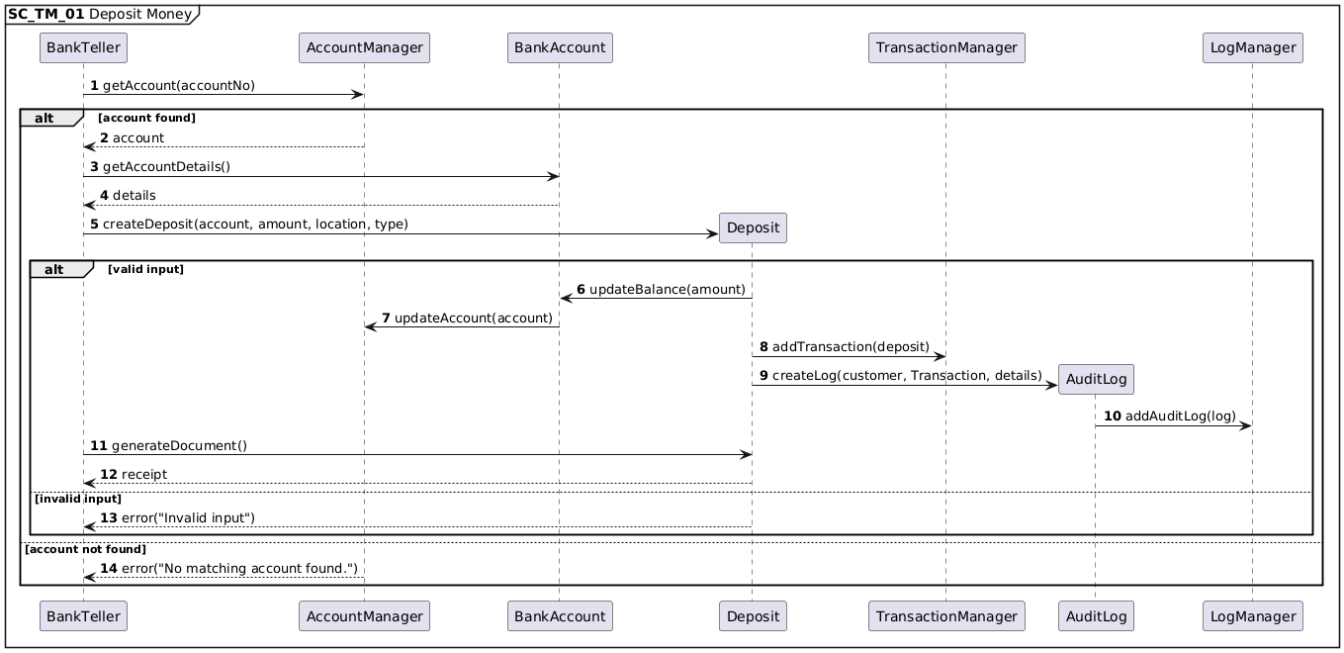

SC_FM_02 Allocate Funds
BankManager
TransactionManager
BankUnit
UnitManager
LogManager
ref
Include SD_FM_01 View Funds
1 createFundTransfer(destination, amount)
FundTransfer
alt
[valid amount]
2 addTransaction(transfer)
3 updateLiquidity(amount)
4 updateUnit(unit)
5 createLog(manager, Transaction, details)
AuditLog
6 addAuditLog(log)
7 generateDocument()
8 document
[invalid amount]
9 error("Invalid amount")
BankManager
FundTransfer
TransactionManager
BankUnit
UnitManager
AuditLog
LogManager

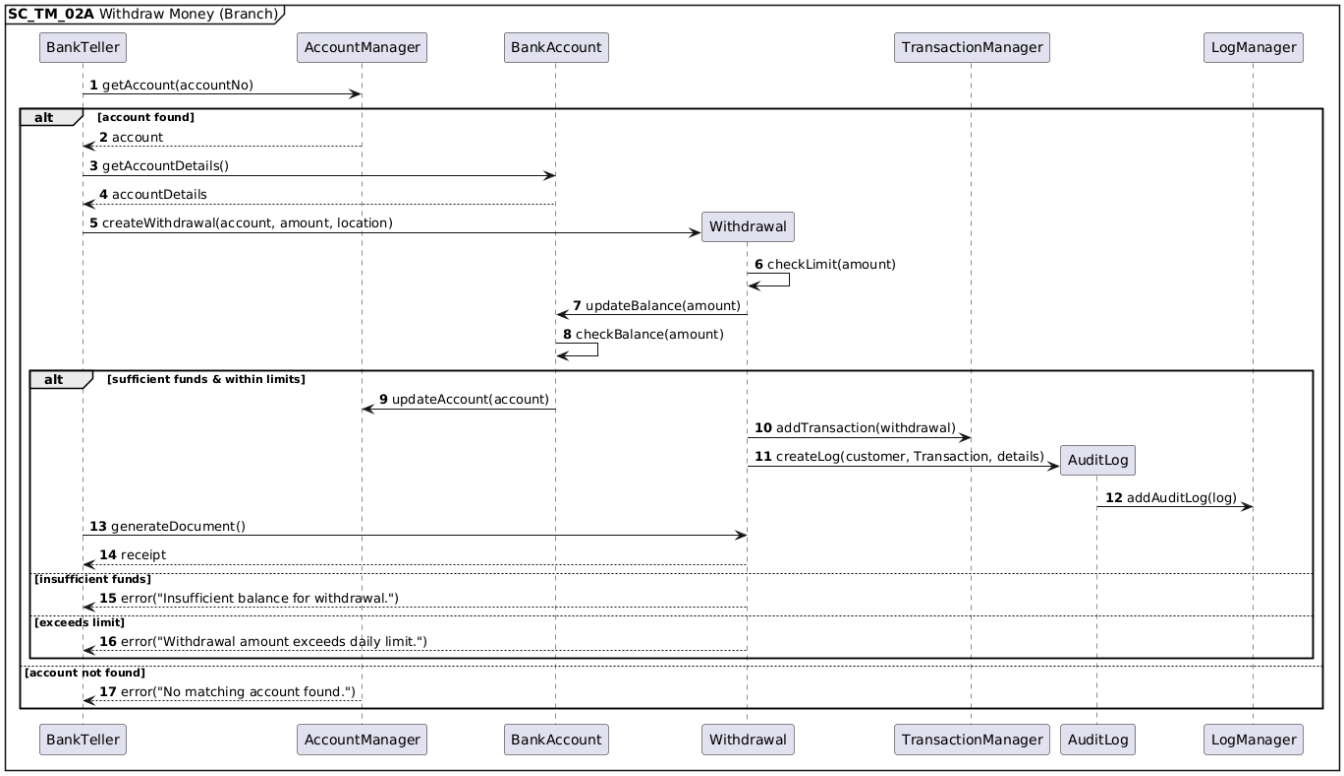

SC_AM_01 Open Bank Account
BankTeller
UserManager
AccountManager
LogManager
1 createAccount(type, interest, maintenanceFee, currency)
BankAccount
2 setStatus(Active)
3 setInterestRate(interest)
4 setMaintenanceFees(fees)
5 setCurrency(currency)
alt [AccountType == Checking Account]
6 setWithdrawalEligibility(true)
7 setWithdrawalEligibility(false)
8 addOwner(name, SSN, birthdate, phone, email, address)
alt [valid data]
opt [new customer]
9 createCustomer(name, SSN, birthdate, phone, email, address)
Customer
10 addUser(customer)
11 addAccount(account)
12 createLog(customer, Account, details)
AuditLog
13 addAuditLog(log)
14 getAccountDetails()
15 details
16 generateDocument()
17 document
loop [periodically]
18 applyInterest()
19 applyMaintenanceFee()
20 updateAccount(account)
[invalid data]
21 error("Invalid data")

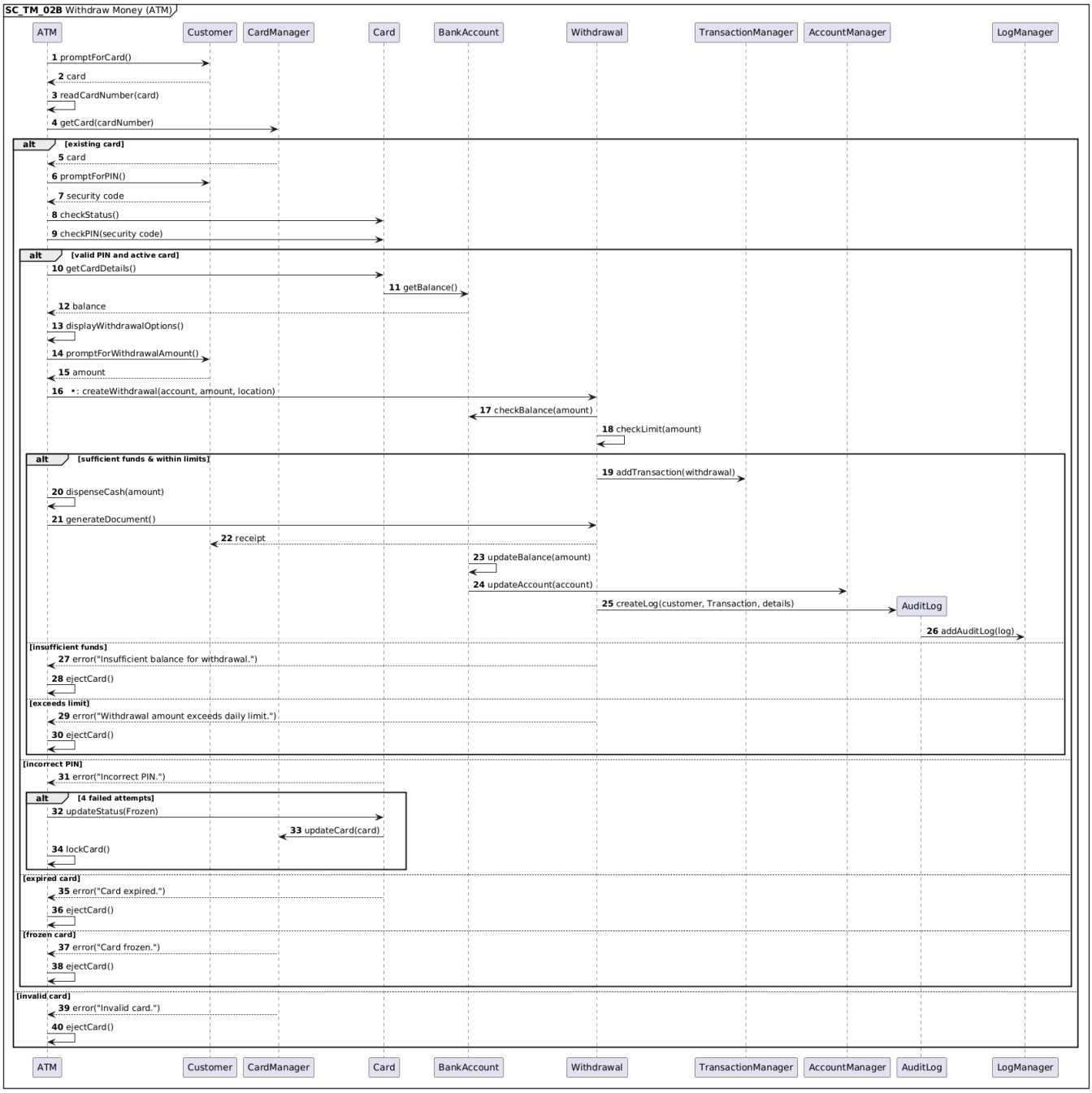

SC_AM_02 Close Bank Account
BankTeller
AccountManager
BankAccount
CardManager
Card
LogManager
1 getAccountByNo(accountNo)
alt [account found]
2 account
3 getAccountDetails()
4 details
ref
include SD_TM_02A Withdraw Money (Branch) if balance > 0
5 closeAccount()
6 setStatus(Closed)
7 getCard(accountNo)
alt [existing linked card]
8 card
9 updateStatus(Deactivated)
10 updateCard(card)
11 generateDocument()
12 document
13 createLog(customer, Account, details)
AuditLog
14 addAuditLog(log)
[account not found]
15 error("Account Not Found.")

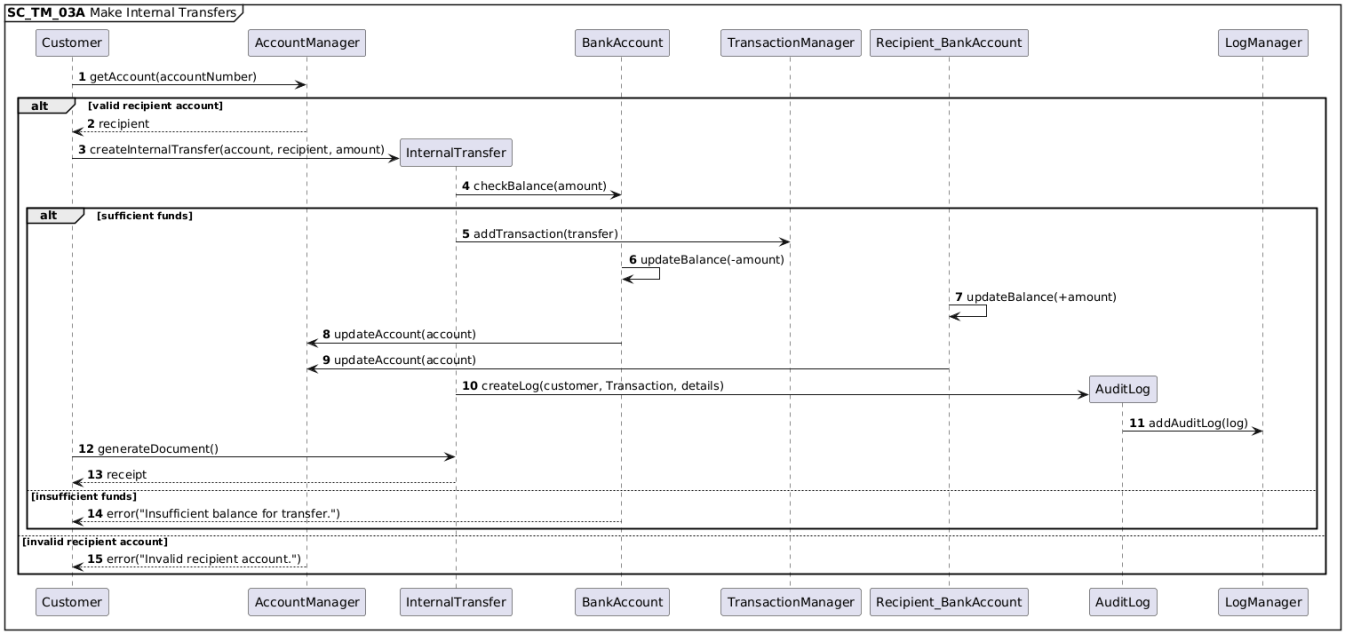

SC_AM_03 Modify Bank Account
BankTeller
AccountManager
BankAccount
LogManager
Customer
1 getAccountByNo(accountNo)
alt
[account found]
2 account
3 getAccountDetails()
4 details
5 setStatus(status)
6 setCurrency(currency, conversionRate)
7 setBalance(conversionRate * balance)
8 setInterestRate(rate)
9 setMaintenanceFee(fee)
alt
[valid data]
10 updateAccount(account)
11 createLog(customer, Account, details)
AuditLog
12 addAuditLog(log)
13 alert("Account modified.")
[invalid data]
14 error("Invalid Data")
[account not found]
15 error("Account Not Found.")
BankTeller
AccountManager
BankAccount
AuditLog
LogManager
Customer

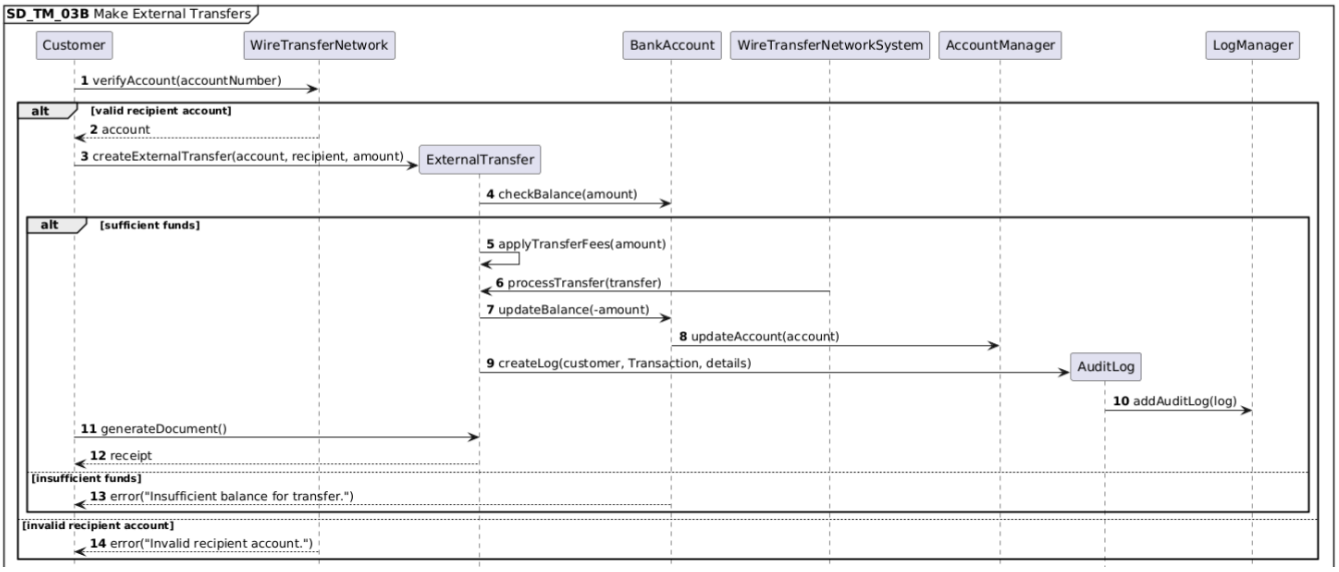

SC_AM_04 View Bank Account Information
Customer
AccountManager
BankAccount
TransactionManager
AccountTransaction
1 getAccountsByCustomer(customer)
2 accounts
3 getAccountDetails()
4 details
5 getTransactionsByAccount(account)
6 transactions
loop
[for each transaction]
7 getTransactionDetails()
8 details
9 getTransactions(account, start, end, type, amount)
alt
[transactions available after filtering]
10 transactions
loop
[for each transaction]
11 getTransactionDetails()
12 details
[no transactions after filtering]
13 error("No Matching Transactions Found")
Customer
AccountManager
BankAccount
TransactionManager
AccountTransaction

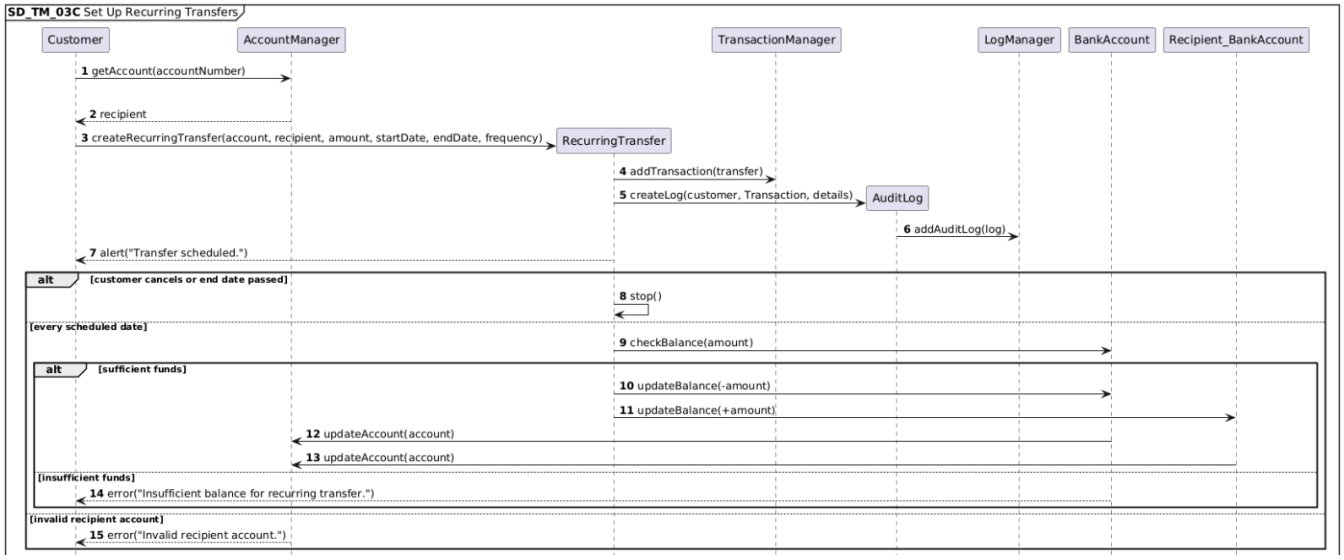

SC_TM_01 Deposit Money
BankTeller
AccountManager
BankAccount
TransactionManager
LogManager
1 getAccount(accountNo)
alt
[account found]
2 account
3 getAccountDetails()
4 details
5 createDeposit(account, amount, location, type)
Deposit
alt
[valid input]
6 updateBalance(amount)
7 updateAccount(account)
8 addTransaction(deposit)
9 createLog(customer, Transaction, details)
AuditLog
10 addAuditLog(log)
11 generateDocument()
12 receipt
[invalid input]
13 error("Invalid input")
[account not found]
14 error("No matching account found.")
BankTeller
AccountManager
BankAccount
Deposit
TransactionManager
AuditLog
LogManager

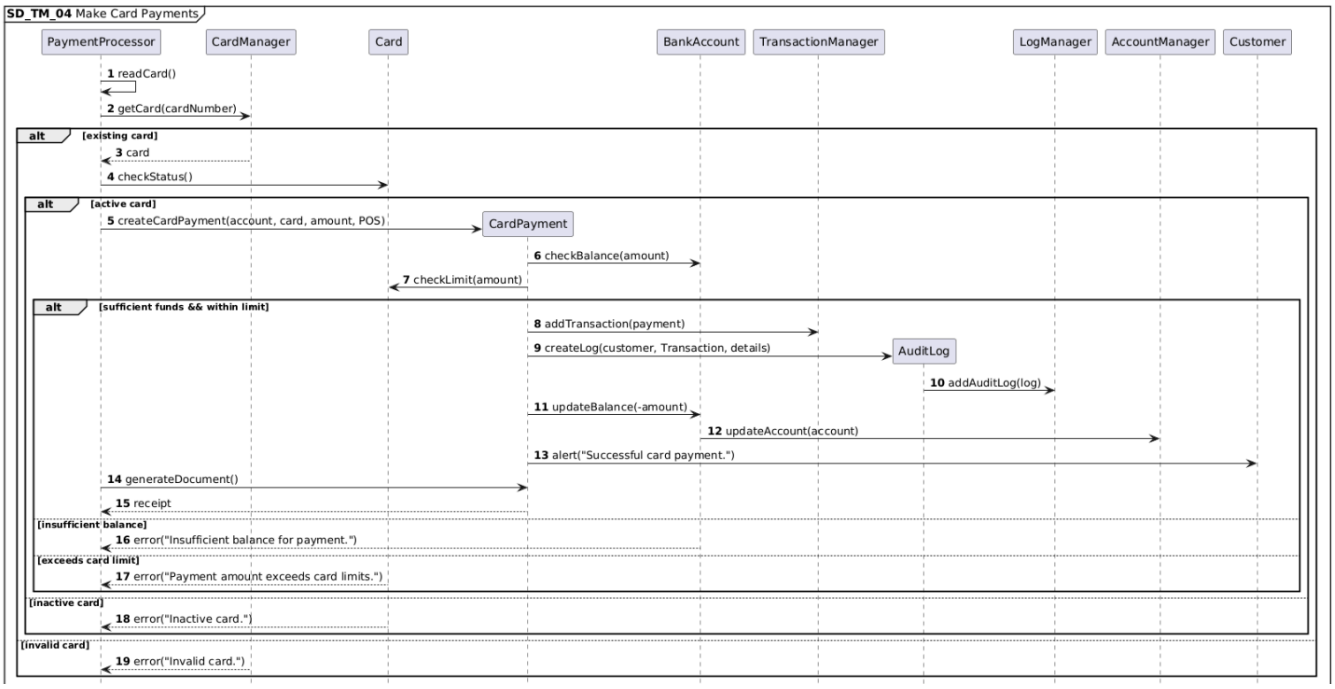

SC_TM_02A Withdraw Money (Branch)
BankTeller
AccountManager
BankAccount
TransactionManager
LogManager
1 getAccount(accountNo)
alt
[account found]
2 account
3 getAccountDetails()
4 accountDetails
5 createWithdrawal(account, amount, location)
Withdrawal
6 checkLimit(amount)
7 updateBalance(amount)
8 checkBalance(amount)
alt
[sufficient funds & within limits]
9 updateAccount(account)
10 addTransaction(withdrawal)
11 createLog(customer, Transaction, details)
AuditLog
12 addAuditLog(log)
13 generateDocument()
14 receipt
[insufficient funds]
15 error("Insufficient balance for withdrawal.")
[exceeds limit]
16 error("Withdrawal amount exceeds daily limit.")
[account not found]
17 error("No matching account found.")
BankTeller
AccountManager
BankAccount
Withdrawal
TransactionManager
AuditLog
LogManager

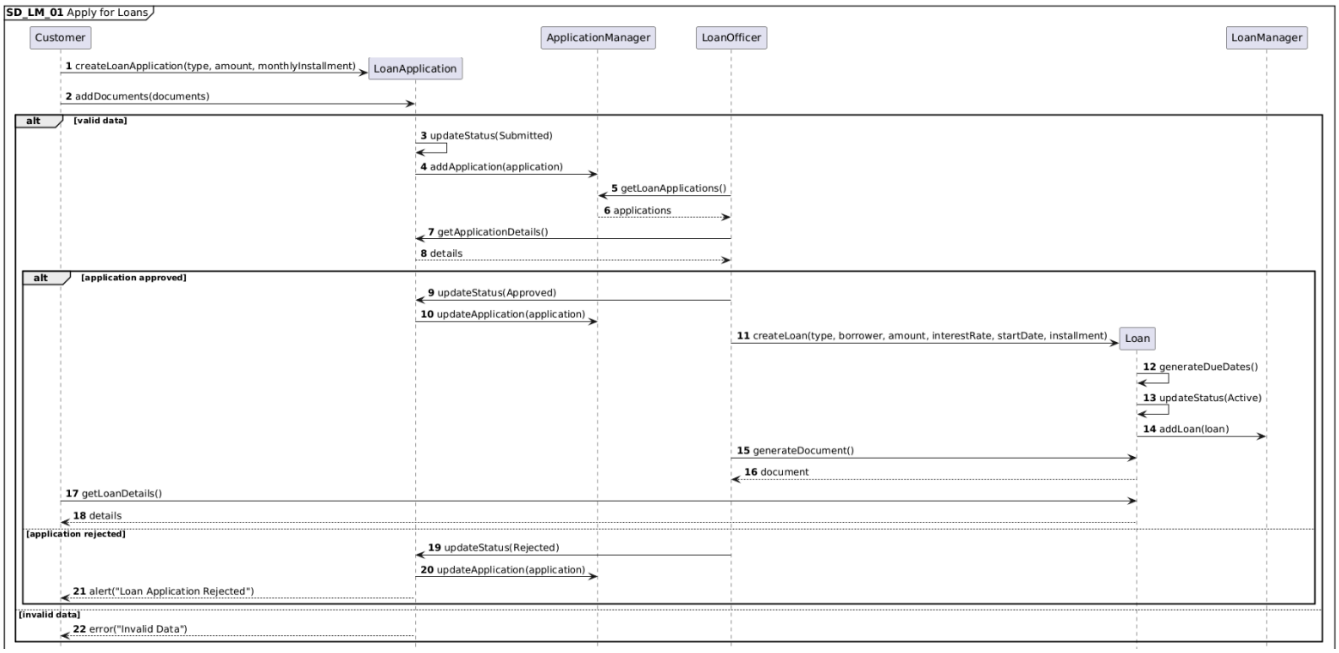

SC_TM_02B Withdraw Money (ATM)
ATM
Customer
CardManager
Card
BankAccount
Withdrawal
TransactionManager
AccountManager
AuditLog
LogManager
1 promptForCard()
2 card
3 readCardNumber(card)
4 getCard(cardNumber)
alt [existing card]
5 card
6 promptForPIN()
7 security code
8 checkStatus()
9 checkPIN(security code)
alt [valid PIN and active card]
10 getCardDetails()
11 getBalance()
12 balance
13 displayWithdrawalOptions()
14 promptForWithdrawalAmount()
15 amount
16 *: createWithdrawal(account, amount, location)
17 checkBalance(amount)
18 checkLimit(amount)
alt [sufficient funds & within limits]
19 addTransaction(withdrawal)
20 dispenseCash(amount)
21 generateDocument()
22 receipt
23 updateBalance(amount)
24 updateAccount(account)
25 createLog(customer, Transaction, details)
26 addAuditLog(log)
[insufficient funds]
27 error("Insufficient balance for withdrawal.")
28 ejectCard()
[exceeds limit]
29 error("Withdrawal amount exceeds daily limit.")
30 ejectCard()
[incorrect PIN]
31 error("Incorrect PIN.")
alt [4 failed attempts]
32 updateStatus(Frozen)
33 updateCard(card)
34 lockCard()
[expired card]
35 error("Card expired.")
36 ejectCard()
[frozen card]
37 error("Card frozen.")
38 ejectCard()
[invalid card]
39 error("Invalid card.")
40 ejectCard()

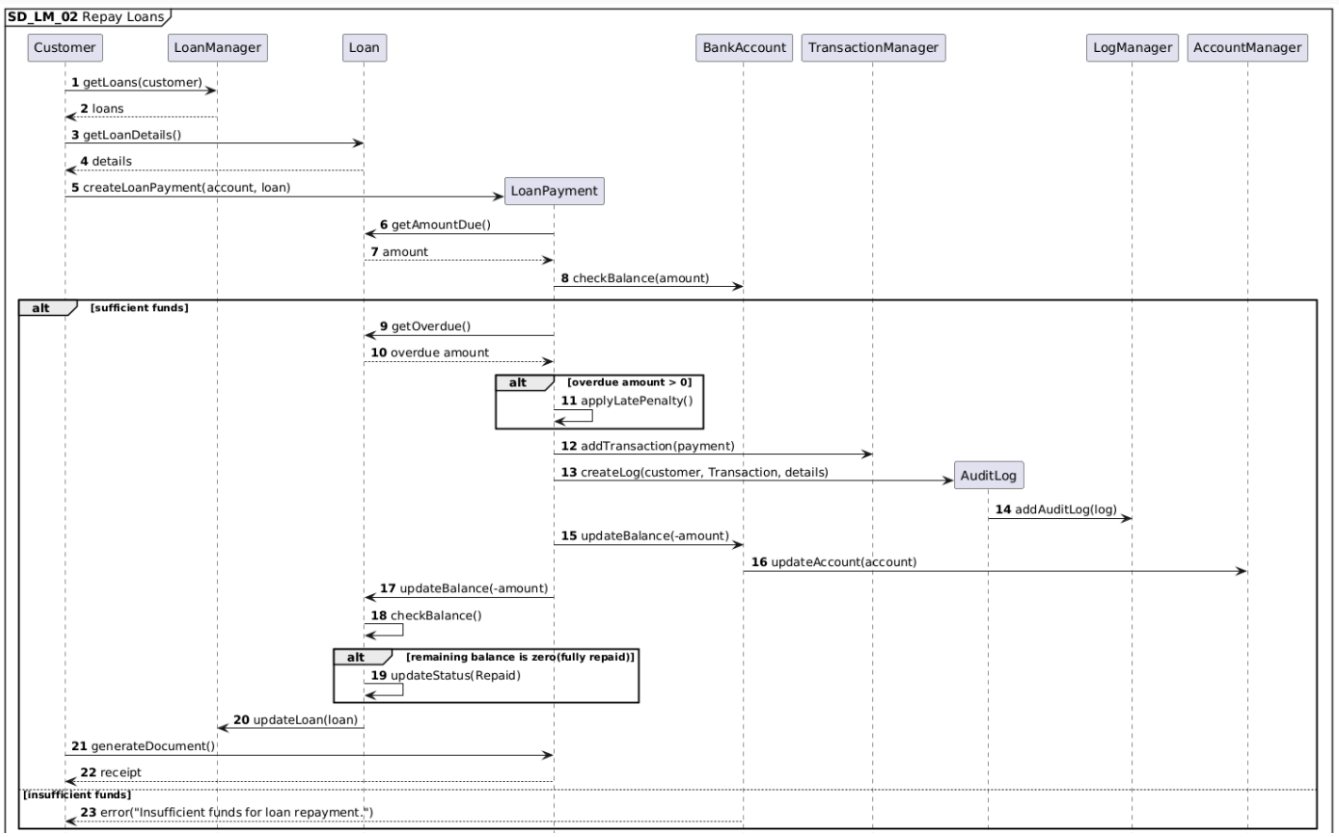

SC_TM_03A Make Internal Transfers
Customer
AccountManager
BankAccount
TransactionManager
Recipient_BankAccount
LogManager
1 getAccount(accountNumber)
alt [valid recipient account]
2 recipient
3 createInternalTransfer(account, recipient, amount)
InternalTransfer
4 checkBalance(amount)
alt [sufficient funds]
5 addTransaction(transfer)
6 updateBalance(-amount)
7 updateBalance(+amount)
8 updateAccount(account)
9 updateAccount(account)
10 createLog(customer, Transaction, details)
AuditLog
11 addAuditLog(log)
12 generateDocument()
13 receipt
[insufficient funds]
14 error("Insufficient balance for transfer.")
[invalid recipient account]
15 error("Invalid recipient account.")

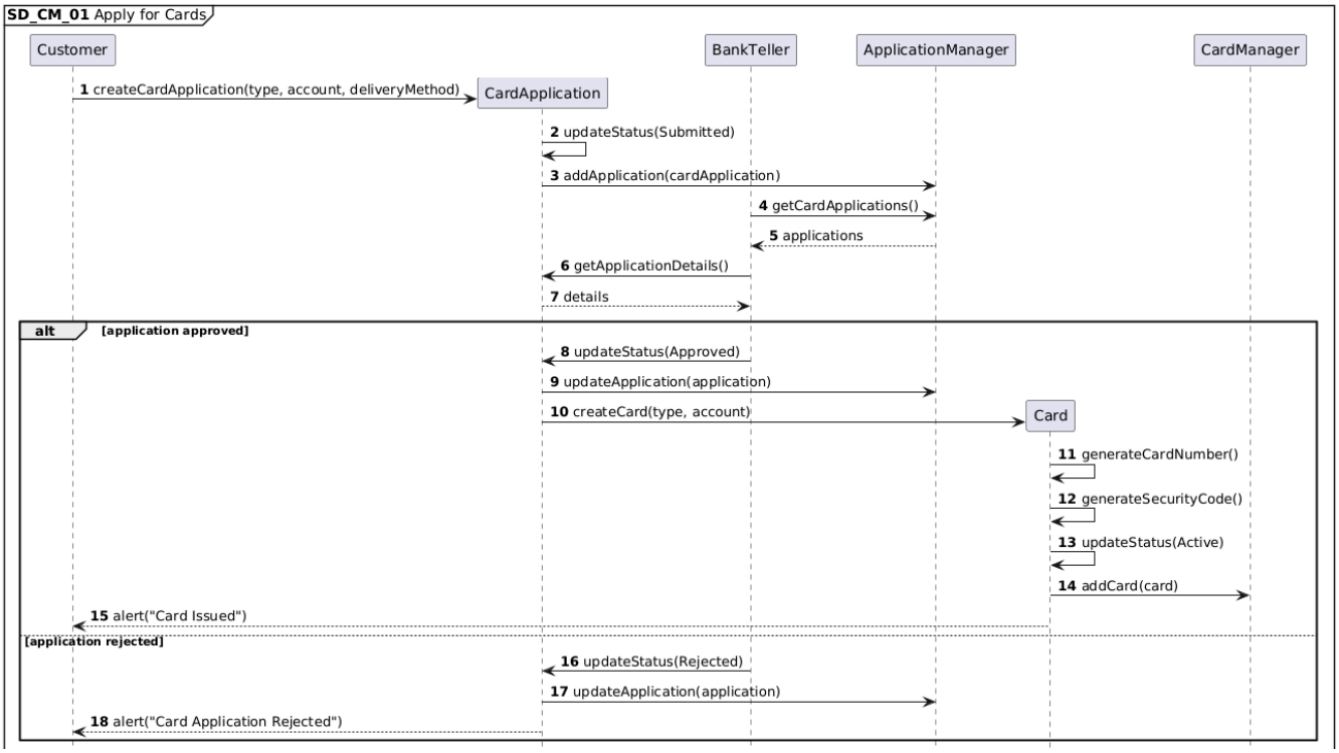

SD_TM_03B Make External Transfers
Customer
WireTransferNetwork
BankAccount
WireTransferNetworkSystem
AccountManager
LogManager
1 verifyAccount(accountNumber)
alt [valid recipient account]
2 account
3 createExternalTransfer(account, recipient, amount)
ExternalTransfer
4 checkBalance(amount)
alt [sufficient funds]
5 applyTransferFees(amount)
6 processTransfer(transfer)
7 updateBalance(-amount)
8 updateAccount(account)
9 createLog(customer, Transaction, details)
AuditLog
10 addAuditLog(log)
11 generateDocument()
12 receipt
[insufficient funds]
13 error("Insufficient balance for transfer.")
[invalid recipient account]
14 error("Invalid recipient account.")


SD_TM_03C Set Up Recurring Transfers
Customer
AccountManager
TransactionManager
LogManager
BankAccount
Recipient_BankAccount
1 getAccount(accountNumber)
2 recipient
3 createRecurringTransfer(account, recipient, amount, startDate, endDate, frequency)
RecurringTransfer
4 addTransaction(transfer)
5 createLog(customer, Transaction, details)
AuditLog
6 addAuditLog(log)
7 alert("Transfer scheduled.")
alt [customer cancels or end date passed]
8 stop()
[every scheduled date]
9 checkBalance(amount)
alt [sufficient funds]
10 updateBalance(-amount)
11 updateBalance(+amount)
12 updateAccount(account)
13 updateAccount(account)
[insufficient funds]
14 error("Insufficient balance for recurring transfer.")
[invalid recipient account]
15 error("Invalid recipient account.")

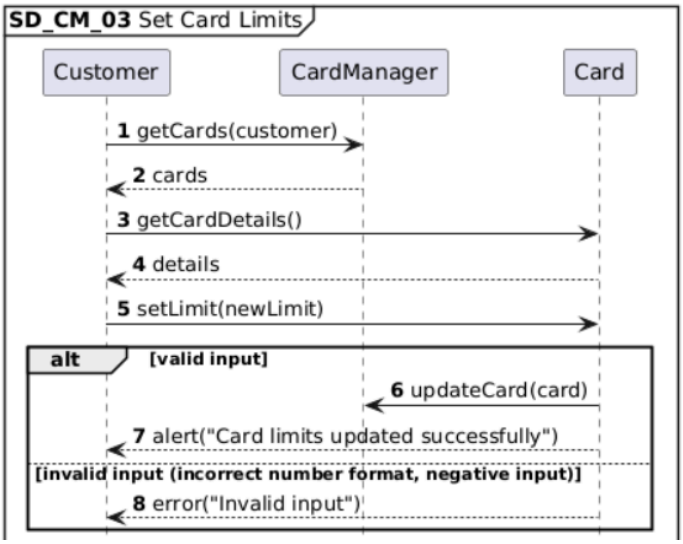

SD_TM_04 Make Card Payments
PaymentProcessor
CardManager
Card
BankAccount
TransactionManager
LogManager
AccountManager
Customer
1 readCard()
2 getCard(cardNumber)
alt [existing card]
3 card
4 checkStatus()
alt [active card]
5 createCardPayment(account, card, amount, POS)
CardPayment
6 checkBalance(amount)
7 checkLimit(amount)
alt [sufficient funds && within limit]
8 addTransaction(payment)
9 createLog(customer, Transaction, details)
AuditLog
10 addAuditLog(log)
11 updateBalance(-amount)
12 updateAccount(account)
13 alert("Successful card payment.")
14 generateDocument()
15 receipt
[insufficient balance]
16 error("Insufficient balance for payment.")
[exceeds card limit]
17 error("Payment amount exceeds card limits.")
[inactive card]
18 error("Inactive card.")
[invalid card]
19 error("Invalid card.")

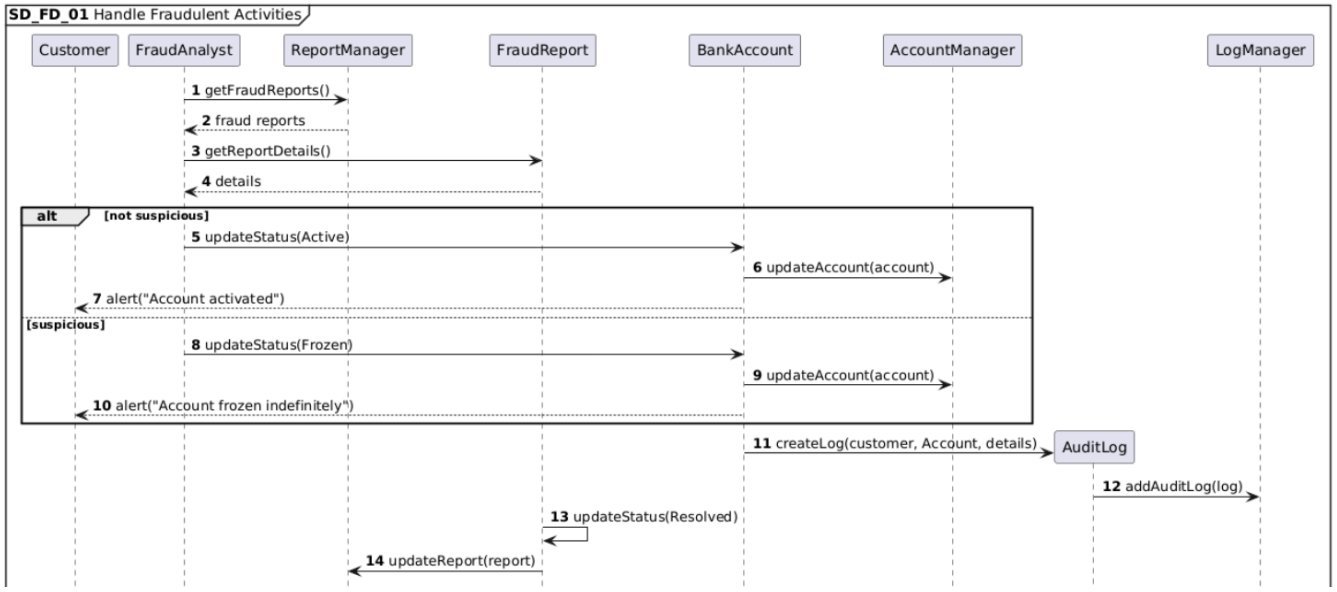

SD_LM_01 Apply for Loans
Customer
LoanApplication
ApplicationManager
LoanOfficer
LoanManager
1 createLoanApplication(type, amount, monthlyInstallment)
2 addDocuments(documents)
alt [valid data]
3 updateStatus(Submitted)
4 addApplication(application)
5 getLoanApplications()
6 applications
7 getApplicationDetails()
8 details
alt [application approved]
9 updateStatus(Approved)
10 updateApplication(application)
11 createLoan(type, borrower, amount, interestRate, startDate, installment)
Loan
12 generateDueDates()
13 updateStatus(Active)
14 addLoan(loan)
15 generateDocument()
16 document
17 getLoanDetails()
18 details
[application rejected]
19 updateStatus(Rejected)
20 updateApplication(application)
21 alert("Loan Application Rejected")
[invalid data]
22 error("Invalid Data")

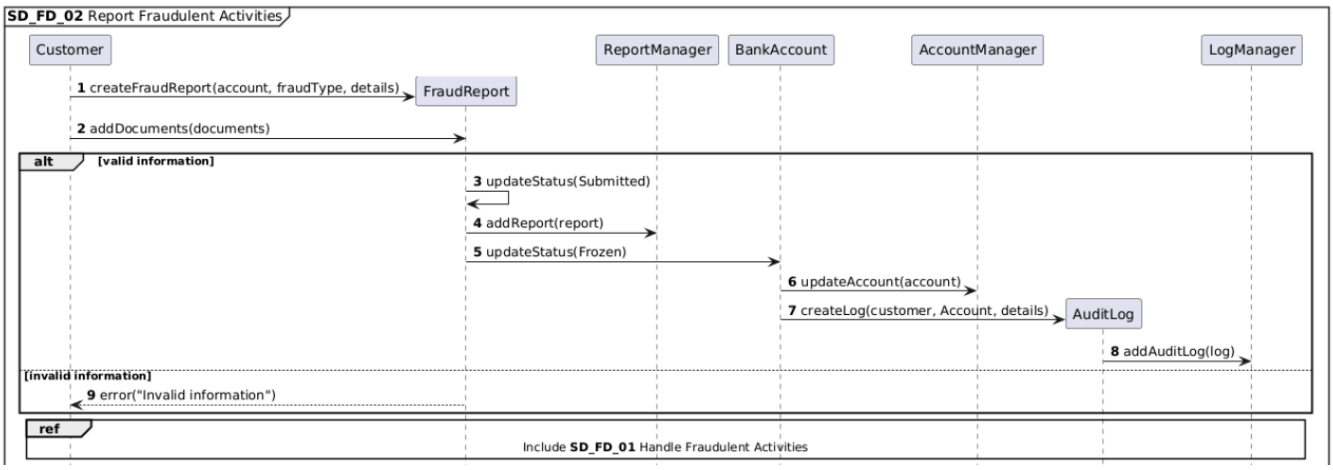

SD_LM_02 Repay Loans
Customer
LoanManager
Loan
BankAccount
TransactionManager
LogManager
AccountManager
1 getLoans(customer)
2 loans
3 getLoanDetails()
4 details
5 createLoanPayment(account, loan)
LoanPayment
6 getAmountDue()
7 amount
8 checkBalance(amount)
alt [sufficient funds]
9 getOverdue()
10 overdue amount
alt [overdue amount > 0]
11 applyLatePenalty()
12 addTransaction(payment)
13 createLog(customer, Transaction, details)
AuditLog
14 addAuditLog(log)
15 updateBalance(-amount)
16 updateAccount(account)
17 updateBalance(-amount)
18 checkBalance()
alt [remaining balance is zero(fully repaid)]
19 updateStatus(Repaid)
20 updateLoan(loan)
21 generateDocument()
22 receipt
[insufficient funds]
23 error("Insufficient funds for loan repayment.")

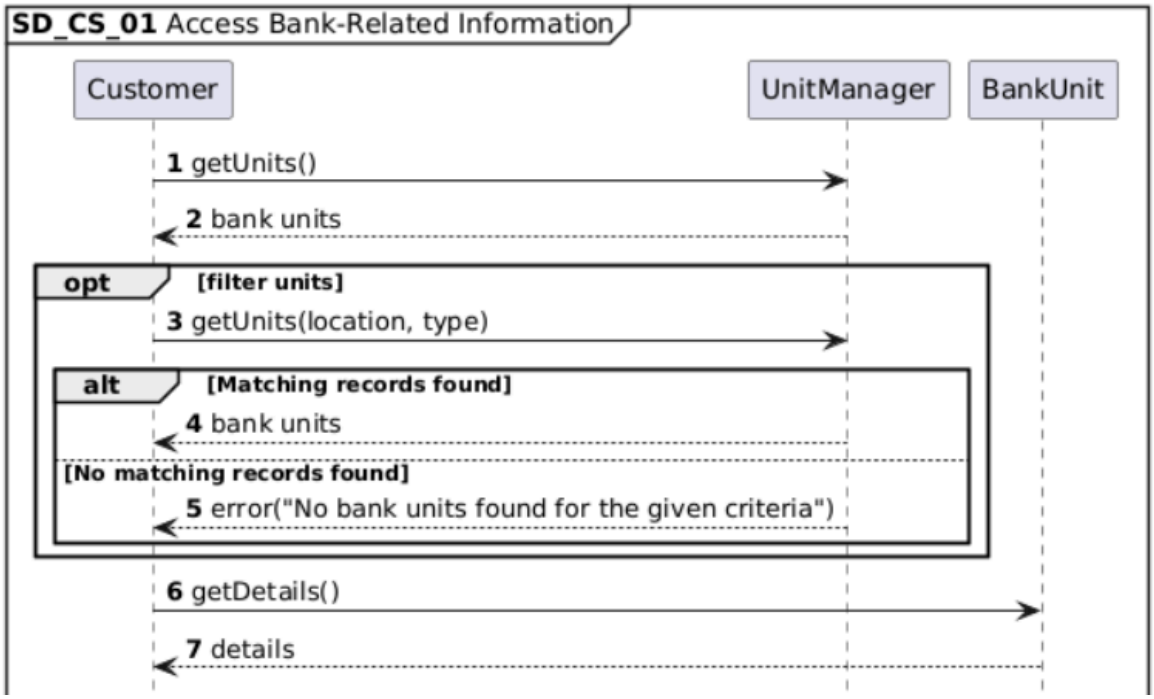

SD_CM_01 Apply for Cards
Customer
BankTeller
ApplicationManager
CardManager
1 createCardApplication(type, account, deliveryMethod)
CardApplication
2 updateStatus(Submitted)
3 addApplication(cardApplication)
4 getCardApplications()
5 applications
6 getApplicationDetails()
7 details
alt [application approved]
8 updateStatus(Approved)
9 updateApplication(application)
10 createCard(type, account)
Card
11 generateCardNumber()
12 generateSecurityCode()
13 updateStatus(Active)
14 addCard(card)
15 alert("Card Issued")
[application rejected]
16 updateStatus(Rejected)
17 updateApplication(application)
18 alert("Card Application Rejected")

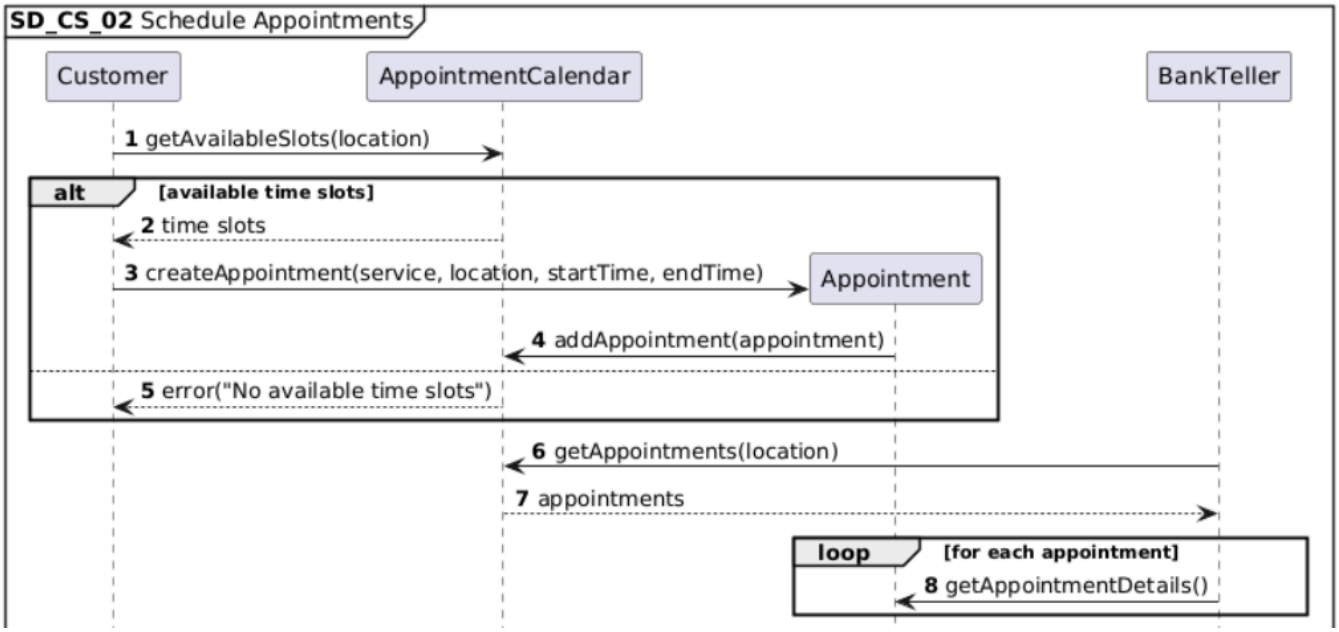

SD_CM_02 Request Checkbooks
Customer
ApplicationManager
BankTeller
CheckbookManager
1 createCheckbookApplication(account, numberOfChecks, deliveryMethod)
CheckbookApplication
alt [valid data]
2 updateStatus(Submitted)
3 addApplication(checkbookApplication)
4 getCheckbookApplications()
5 applications
6 getApplicationDetails()
7 details
alt [approve application]
8 updateStatus(Approved)
9 updateApplication(application)
10 createCheckbook(account, numberOfChecks)
Checkbook
11 generateSerialNumbers(numberOfChecks)
12 addCheckbook(checkbook)
13 alert("Checkbook Issued")
[reject application]
14 updateStatus(Rejected)
15 updateApplication(application)
16 alert("Checkbook Application Rejected")
[invalid data]
17 error("Invalid data")

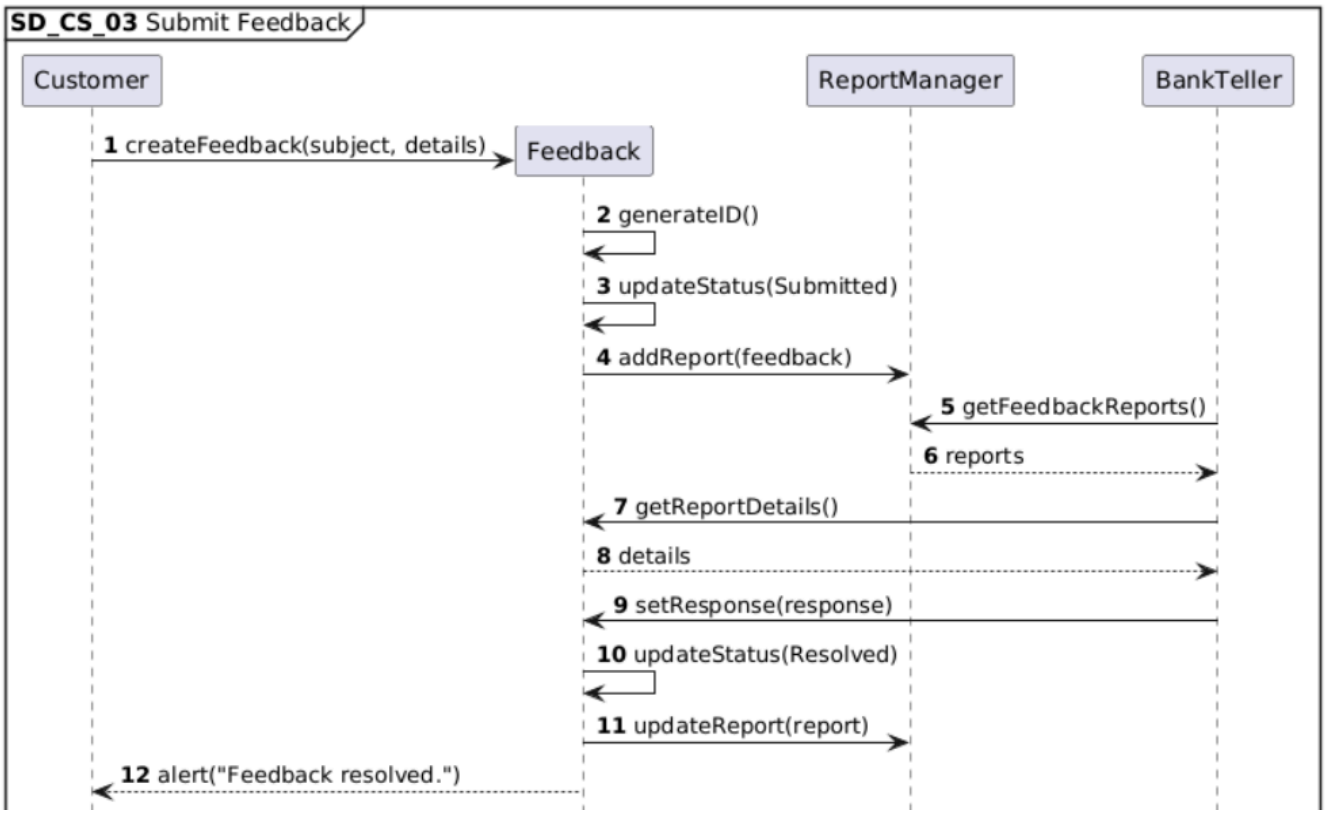

SD_CM_03 Set Card Limits
Customer
CardManager
Card
1 getCards(customer)
2 cards
3 getCardDetails()
4 details
5 setLimit(newLimit)
alt [valid input]
6 updateCard(card)
7 alert("Card limits updated successfully")
[invalid input (incorrect number format, negative input)]
8 error("Invalid input")

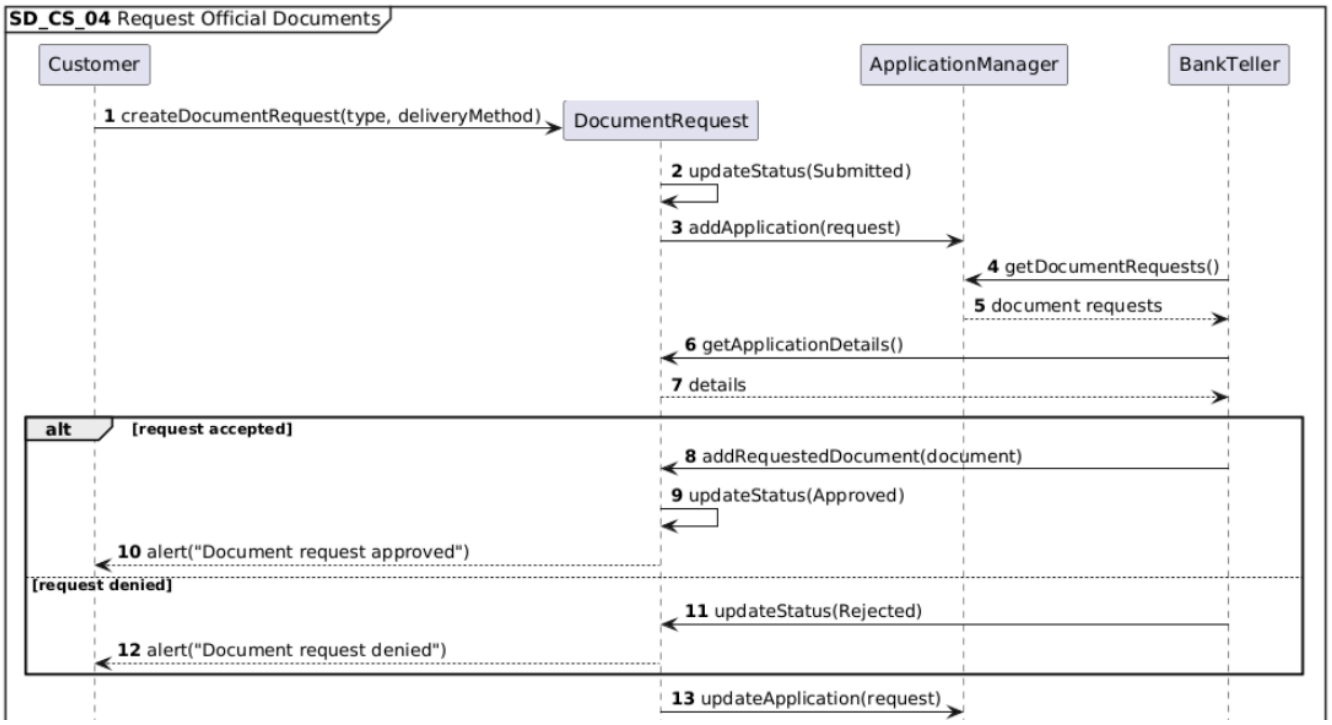

SD_FD_01 Handle Fraudulent Activities
Customer
FraudAnalyst
ReportManager
FraudReport
BankAccount
AccountManager
LogManager
1 getFraudReports()
2 fraud reports
3 getReportDetails()
4 details
alt [not suspicious]
5 updateStatus(Active)
6 updateAccount(account)
7 alert("Account activated")
[suspicious]
8 updateStatus(Frozen)
9 updateAccount(account)
10 alert("Account frozen indefinitely")
11 createLog(customer, Account, details)
AuditLog
12 addAuditLog(log)
13 updateStatus(Resolved)
14 updateReport(report)

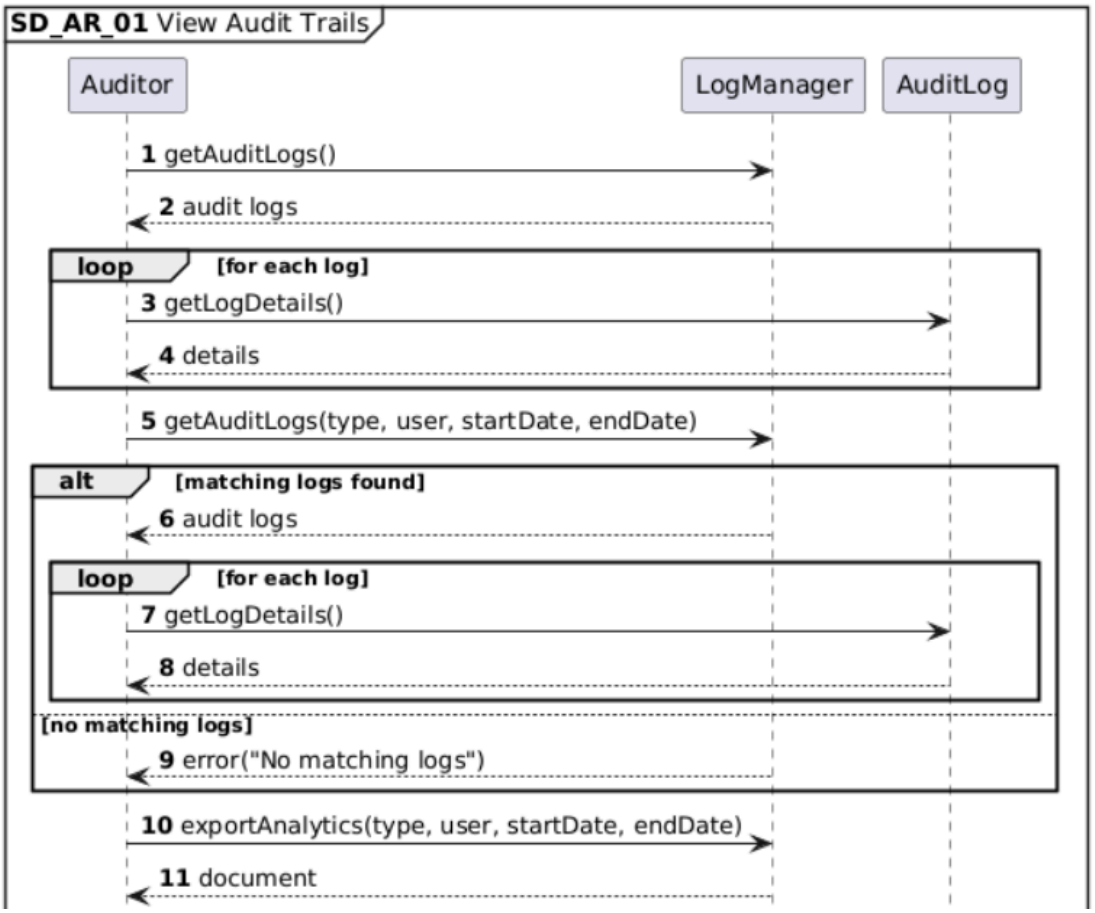

SD_FD_02 Report Fraudulent Activities
Customer
FraudReport
ReportManager
BankAccount
AccountManager
AuditLog
LogManager
1 createFraudReport(account, fraudType, details)
2 addDocuments(documents)
alt
[valid information]
3 updateStatus(Submitted)
4 addReport(report)
5 updateStatus(Frozen)
6 updateAccount(account)
7 createLog(customer, Account, details)
8 addAuditLog(log)
[invalid information]
9 error("Invalid information")
ref
Include SD_FD_01 Handle Fraudulent Activities

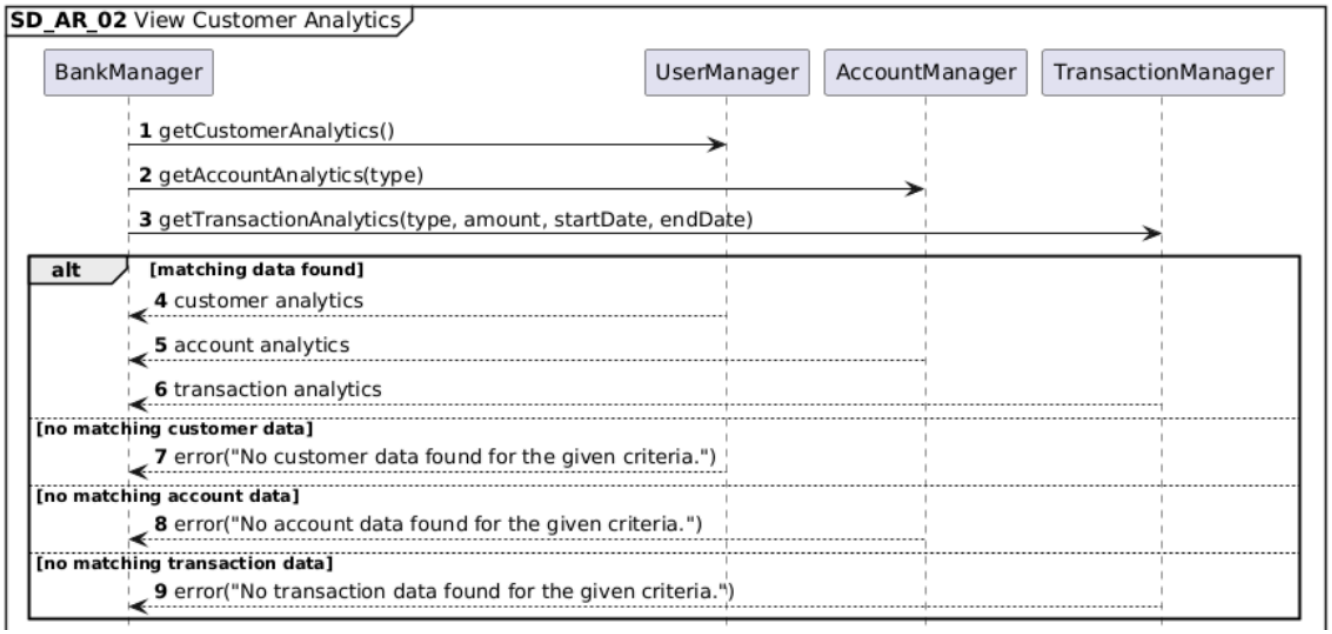

SD_CS_01 Access Bank-Related Information
Customer
UnitManager
BankUnit
1 getUnits()
2 bank units
opt
[filter units]
3 getUnits(location, type)
alt
[Matching records found]
4 bank units
[No matching records found]
5 error("No bank units found for the given criteria")
6 getDetails()
7 details

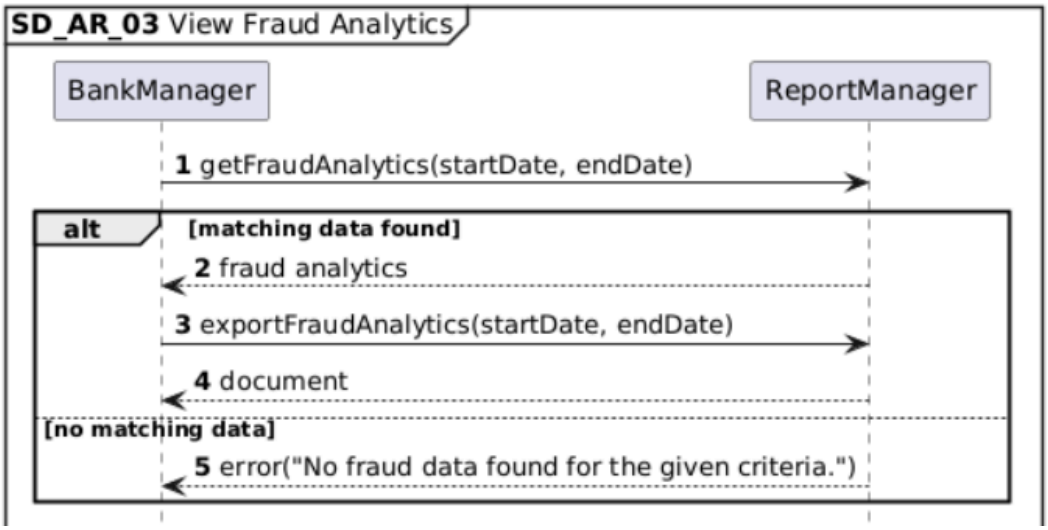

SD_CS_02 Schedule Appointments
Customer
AppointmentCalendar
BankTeller
1 getAvailableSlots(location)
alt
[available time slots]
2 time slots
3 createAppointment(service, location, startTime, endTime)
Appointment
4 addAppointment(appointment)
5 error("No available time slots")
6 getAppointments(location)
7 appointments
loop
[for each appointment]
8 getAppointmentDetails()


SD_CS_03 Submit Feedback
Customer
ReportManager
BankTeller
1 createFeedback(subject, details)
Feedback
2 generateID()
3 updateStatus(Submitted)
4 addReport(feedback)
5 getFeedbackReports()
6 reports
7 getReportDetails()
8 details
9 setResponse(response)
10 updateStatus(Resolved)
11 updateReport(report)
12 alert("Feedback resolved.")

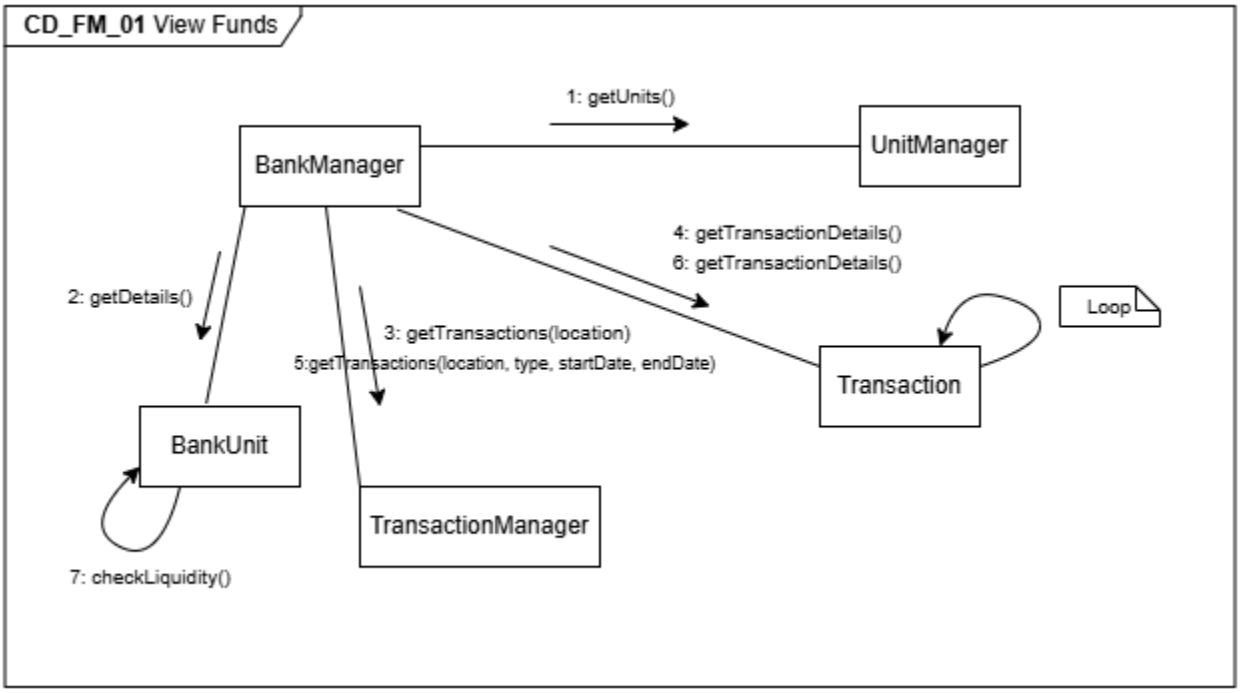

SD_CS_04 Request Official Documents
Customer
DocumentRequest
ApplicationManager
BankTeller
1 createDocumentRequest(type, deliveryMethod)
2 updateStatus(Submitted)
3 addApplication(request)
4 getDocumentRequests()
5 document requests
6 getApplicationDetails()
7 details
alt
[request accepted]
8 addRequestedDocument(document)
9 updateStatus(Approved)
10 alert("Document request approved")
[request denied]
11 updateStatus(Rejected)
12 alert("Document request denied")
13 updateApplication(request)

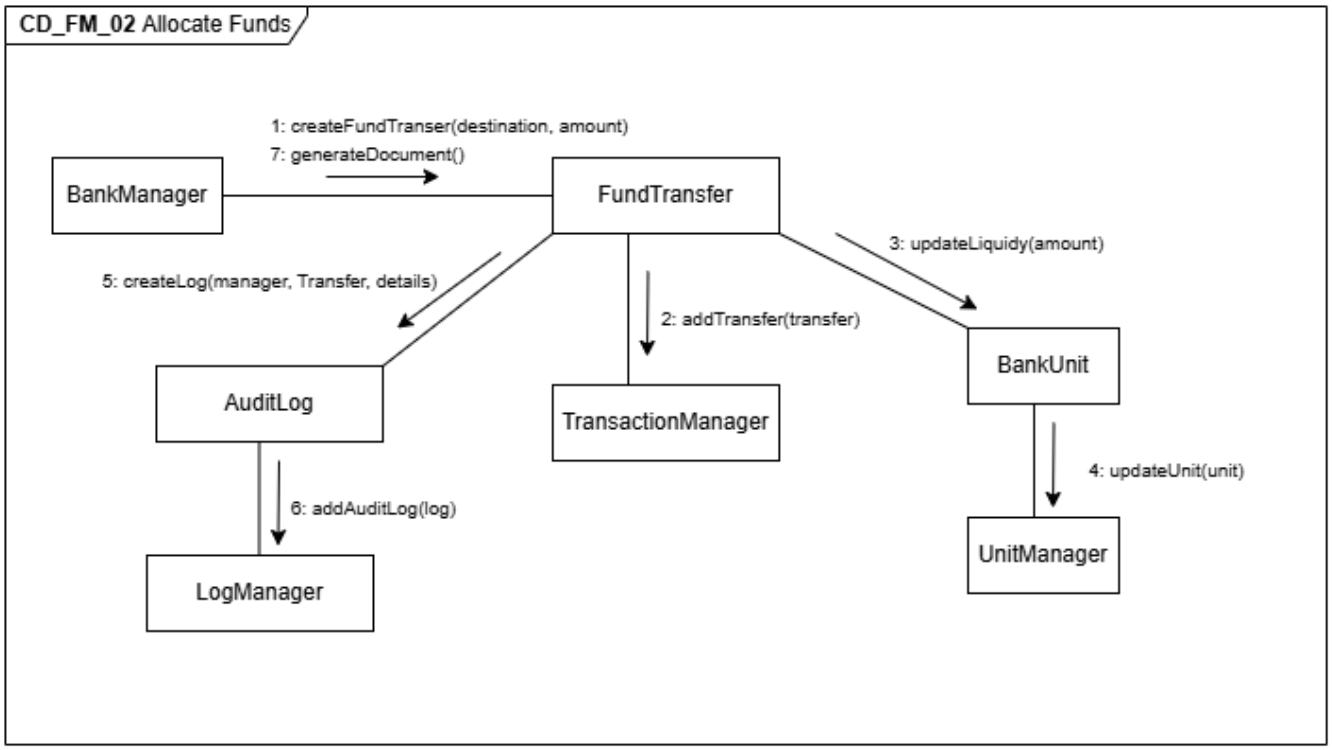

SD_AR_01 View Audit Trails
Auditor
LogManager
AuditLog
1 getAuditLogs()
2 audit logs
loop
[for each log]
3 getLogDetails()
4 details
5 getAuditLogs(type, user, startDate, endDate)
alt
[matching logs found]
6 audit logs
loop
[for each log]
7 getLogDetails()
8 details
[no matching logs]
9 error("No matching logs")
10 exportAnalytics(type, user, startDate, endDate)
11 document

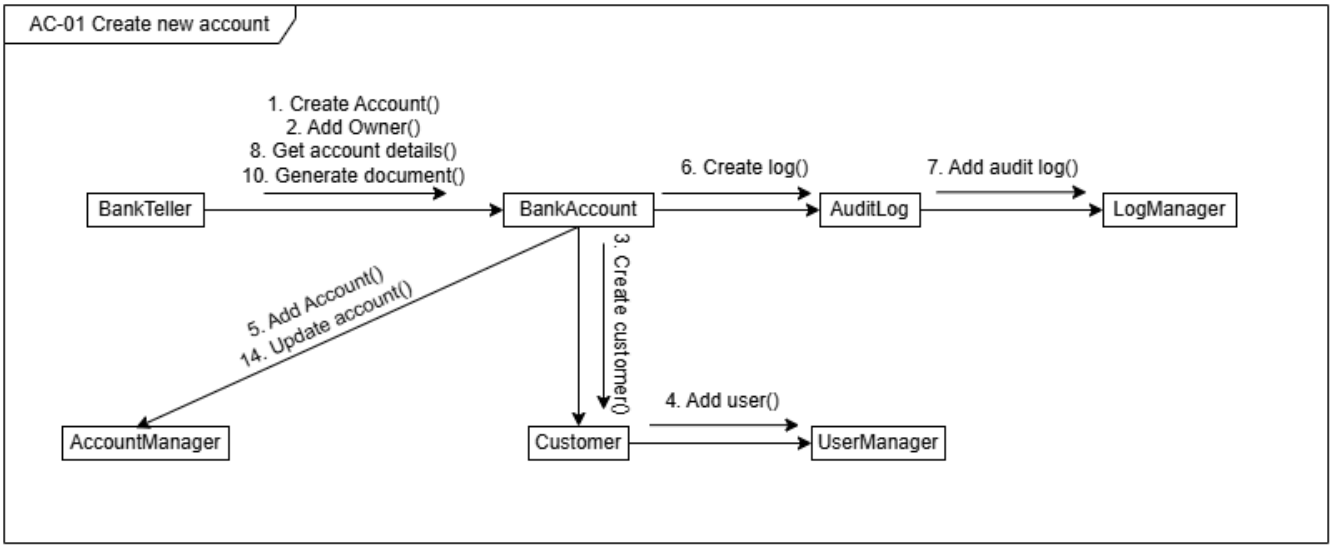

SD_AR_02 View Customer Analytics
BankManager
UserManager
AccountManager
TransactionManager
1 getCustomerAnalytics()
2 getAccountAnalytics(type)
3 getTransactionAnalytics(type, amount, startDate, endDate)
alt
[matching data found]
4 customer analytics
5 account analytics
6 transaction analytics
[no matching customer data]
7 error("No customer data found for the given criteria.")
[no matching account data]
8 error("No account data found for the given criteria.")
[no matching transaction data]
9 error("No transaction data found for the given criteria.")

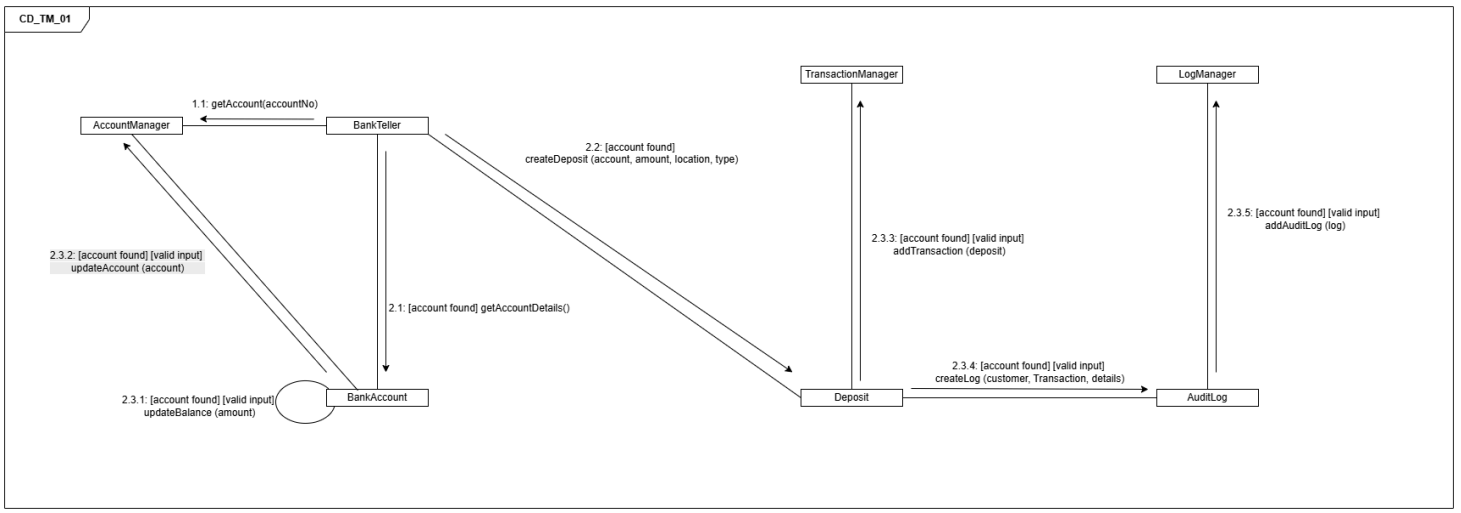

SD_AR_03 View Fraud Analytics
BankManager
ReportManager
1 getFraudAnalytics(startDate, endDate)
alt
[matching data found]
2 fraud analytics
3 exportFraudAnalytics(startDate, endDate)
4 document
[no matching data]
5 error("No fraud data found for the given criteria.")

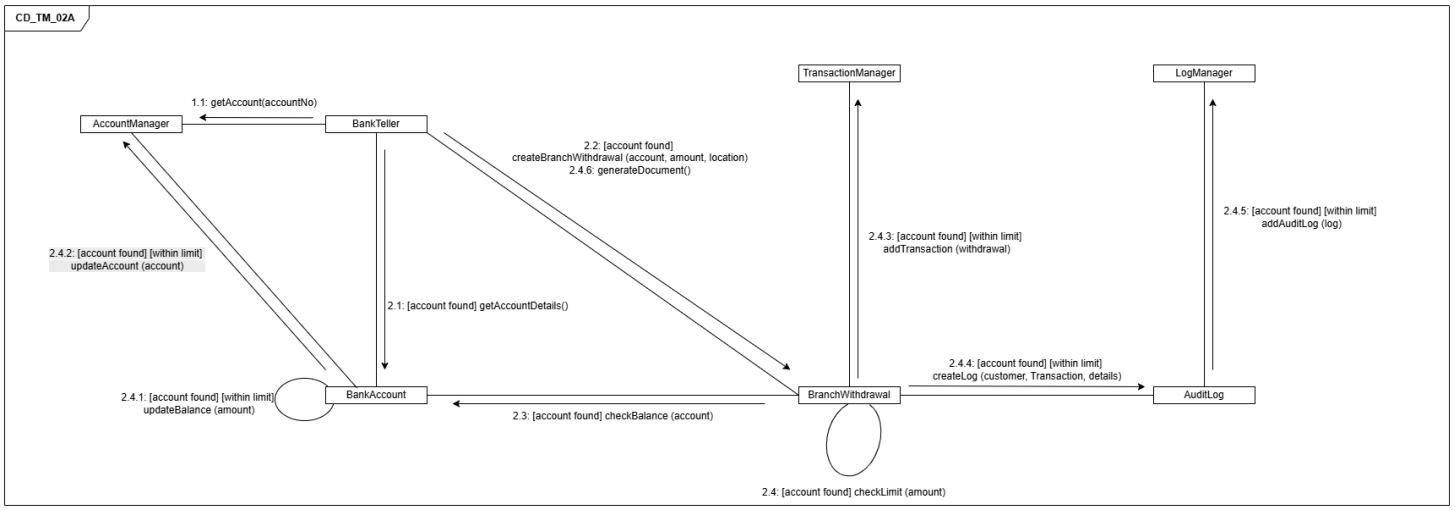
Collaboration Diagrams

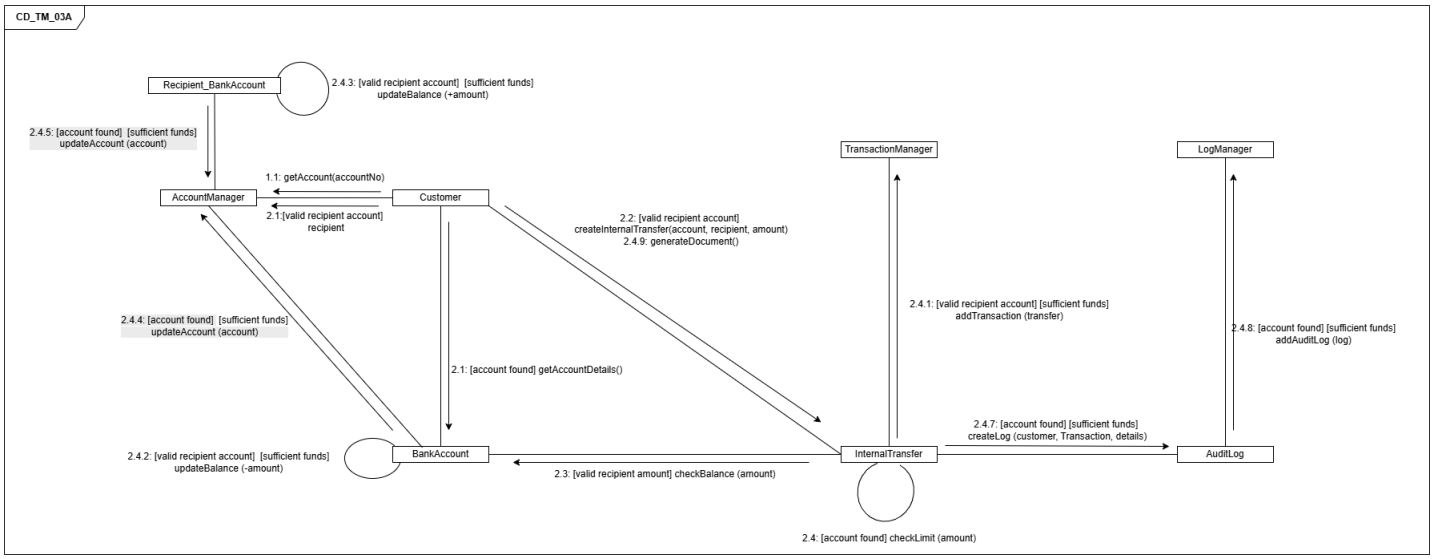


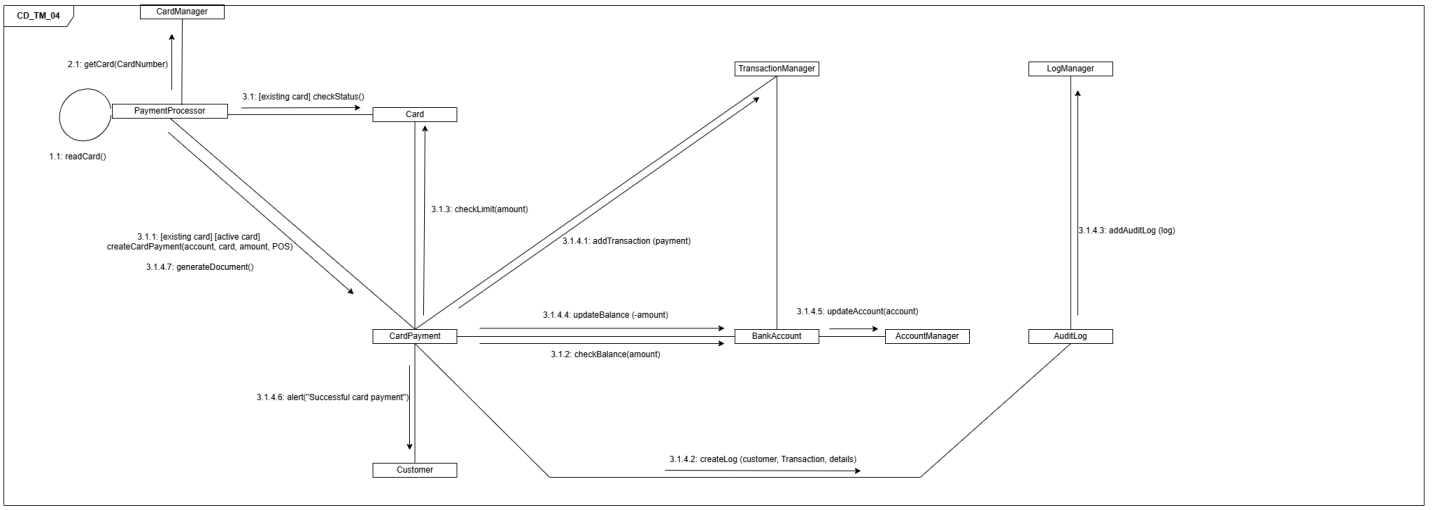


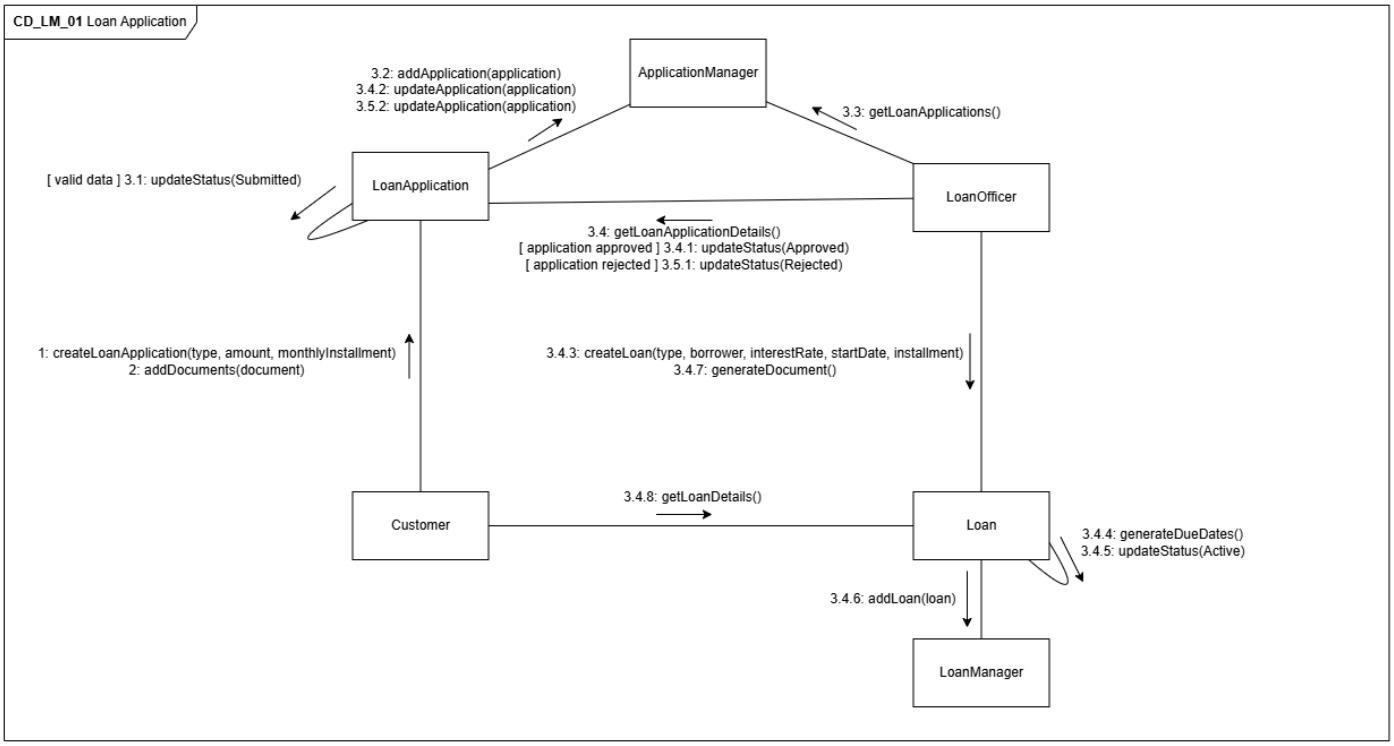

AC-01 Create new account
1. Create Account()
2. Add Owner()
8. Get account details()
10. Generate document()
6. Create log()
7. Add audit log()
BankTeller
BankAccount
AuditLog
LogManager
3. Create customer()
5. Add Account()
14. Update account()
4. Add user()
AccountManager
Customer
UserManager

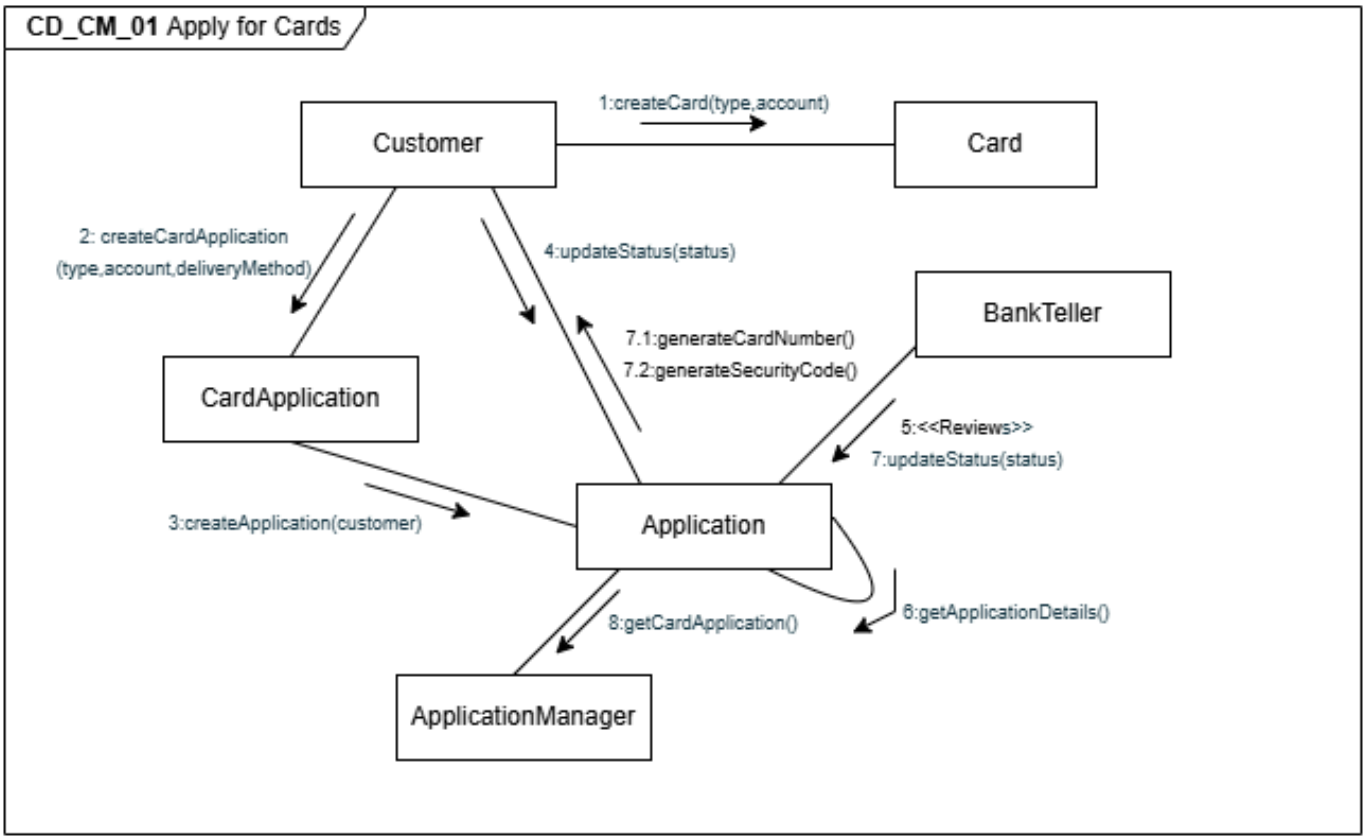

CD_TM_01
TransactionManager
LogManager
1.1: getAccount(accountNo)
AccountManager
BankTeller
2.2: [account found]
createDeposit (account, amount, location, type)
2.3.5: [account found] [valid input]
addAuditLog (log)
2.3.3: [account found] [valid input]
addTransaction (deposit)
2.3.2: [account found] [valid input]
updateAccount (account)
2.1: [account found] getAccountDetails()
2.3.4: [account found] [valid input]
createLog (customer, Transaction, details)
2.3.1: [account found] [valid input]
updateBalance (amount)
BankAccount
Deposit
AuditLog

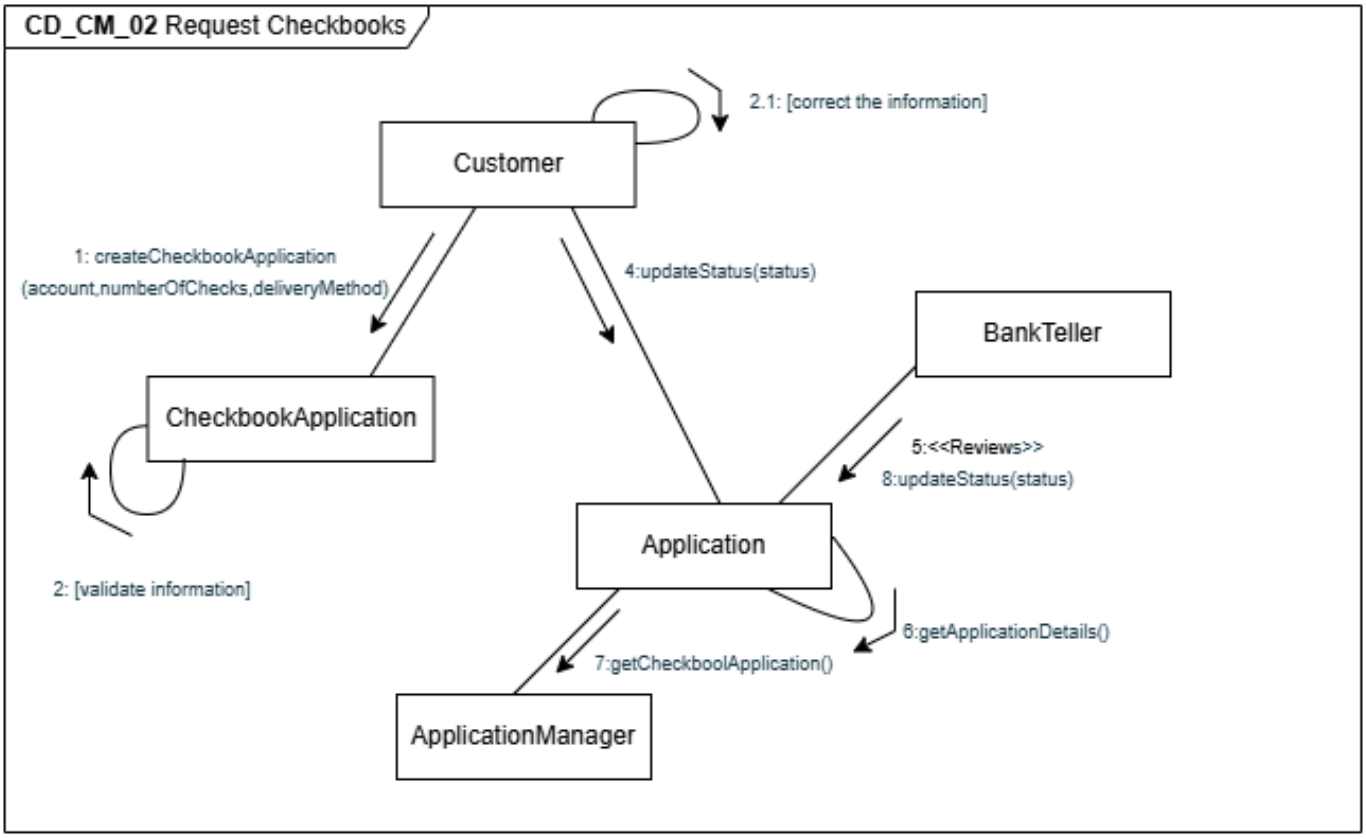

CD_TM_02A
TransactionManager
LogManager
1.1: getAccount(accountNo)
AccountManager
BankTeller
2.2: [account found]
createBranchWithdrawal (account, amount, location)
2.4.6: generateDocument()
2.4.5: [account found] [within limit]
addAuditLog (log)
2.4.3: [account found] [within limit]
addTransaction (withdrawal)
2.4.2: [account found] [within limit]
updateAccount (account)
2.1: [account found] getAccountDetails()
2.4.4: [account found] [within limit]
createLog (customer, Transaction, details)
2.4.1: [account found] [within limit]
updateBalance (amount)
BankAccount
BranchWithdrawal
AuditLog
2.3: [account found] checkBalance (account)
2.4: [account found] checkLimit (amount)

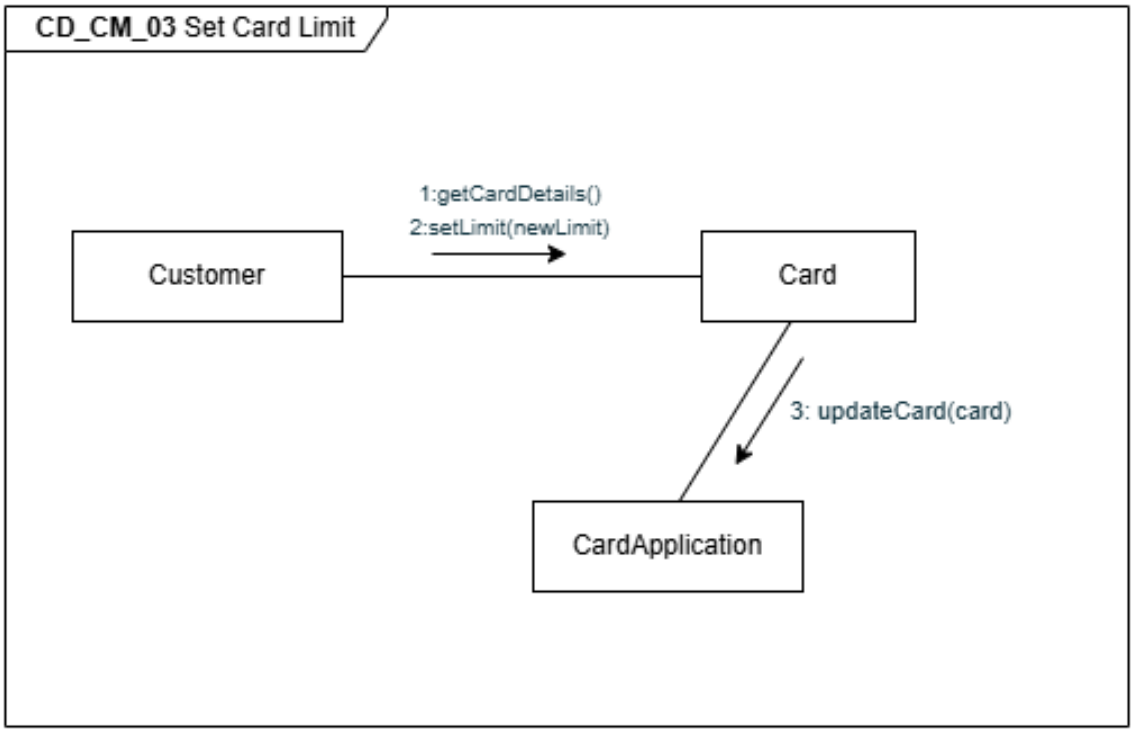

CD_TM_03A
Recipient_BankAccount
2.4.3: [valid recipient account] [sufficient funds]
updateBalance (+amount)
2.4.5: [account found] [sufficient funds]
updateAccount (account)
TransactionManager
LogManager
1.1: getAccount(accountNo)
AccountManager
Customer
2.1:[valid recipient account]
recipient
2.2: [valid recipient account]
createInternalTransfer(account, recipient, amount)
2.4.9: generateDocument()
2.4.1: [valid recipient account] [sufficient funds]
addTransaction (transfer)
2.4.8: [account found] [sufficient funds]
addAuditLog (log)
2.4.4: [account found] [sufficient funds]
updateAccount (account)
2.1: [account found] getAccountDetails()
2.4.7: [account found] [sufficient funds]
createLog (customer, Transaction, details)
2.4.2: [valid recipient account] [sufficient funds]
updateBalance (-amount)
BankAccount
InternalTransfer
AuditLog
2.3: [valid recipient amount] checkBalance (amount)
2.4: [account found] checkLimit (amount)

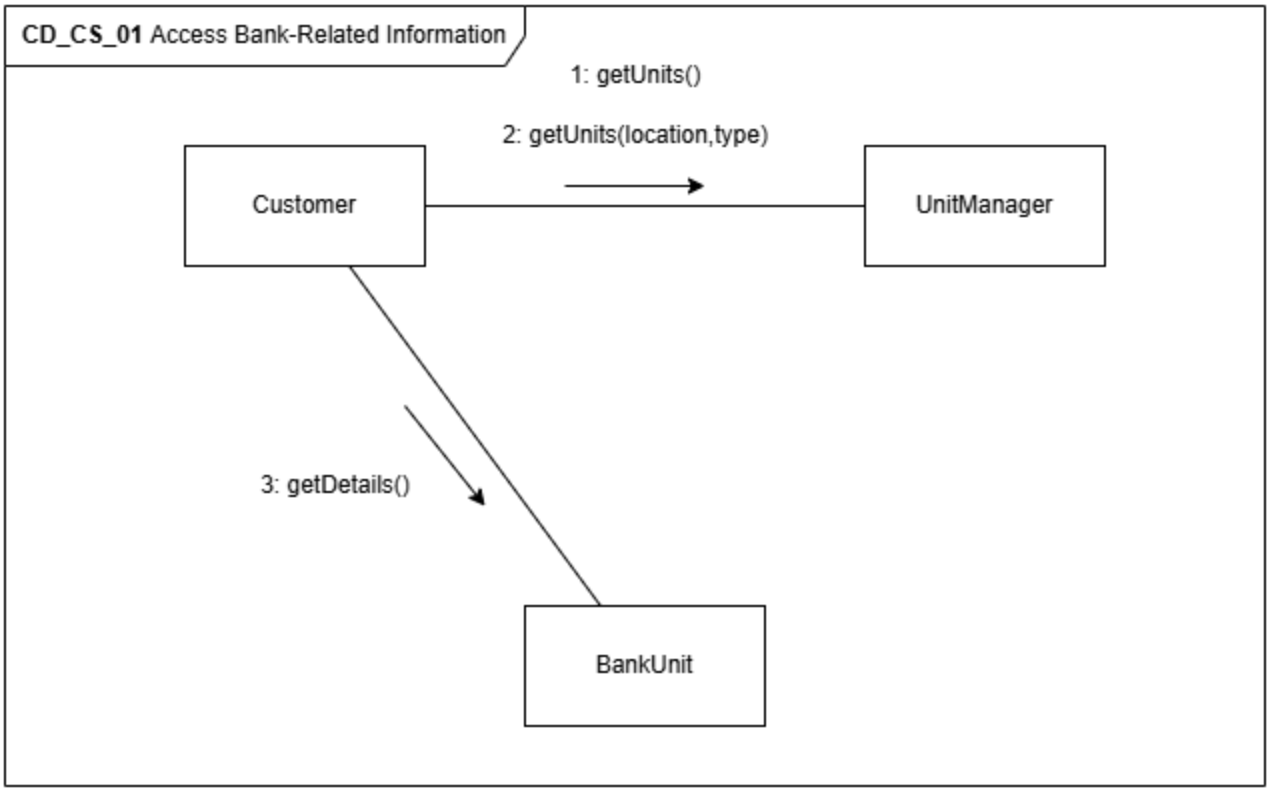

CD_TM_04
CardManager
2.1: getCard(CardNumber)
TransactionManager
LogManager
3.1: [existing card] checkStatus()
PaymentProcessor
Card
1.1: readCard()
3.1.3: checkLimit(amount)
3.1.1: [existing card] [active card]
createCardPayment(account, card, amount, POS)
3.1.4.1: addTransaction (payment)
3.1.4.3: addAuditLog (log)
3.1.4.7: generateDocument()
3.1.4.4: updateBalance (-amount)
3.1.4.5: updateAccount(account)
CardPayment
BankAccount
AccountManager
AuditLog
3.1.2: checkBalance(amount)
3.1.4.6: alert("Successful card payment")
Customer
3.1.4.2: createLog (customer, Transaction, details)

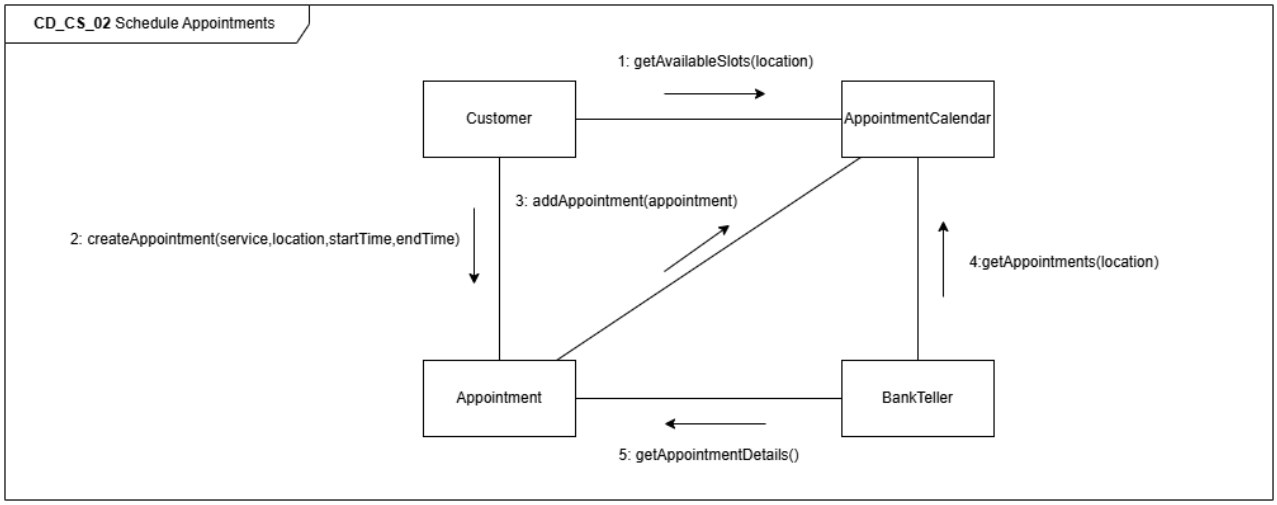

CD_LM_01 Loan Application
ApplicationManager
3.2: addApplication(application)
3.4.2: updateApplication(application)
3.5.2: updateApplication(application)
3.3: getLoanApplications()
[valid data] 3.1: updateStatus(Submitted)
LoanApplication
LoanOfficer
3.4: getLoanApplicationDetails()
[application approved] 3.4.1: updateStatus(Approved)
[application rejected] 3.5.1: updateStatus(Rejected)
1: createLoanApplication(type, amount, monthlyInstallment)
2: addDocuments(document)
3.4.3: createLoan(type, borrower, interestRate, startDate, installment)
3.4.7: generateDocument()
Customer
3.4.8: getLoanDetails()
Loan
3.4.4: generateDueDates()
3.4.5: updateStatus(Active)
3.4.6: addLoan(loan)
LoanManager

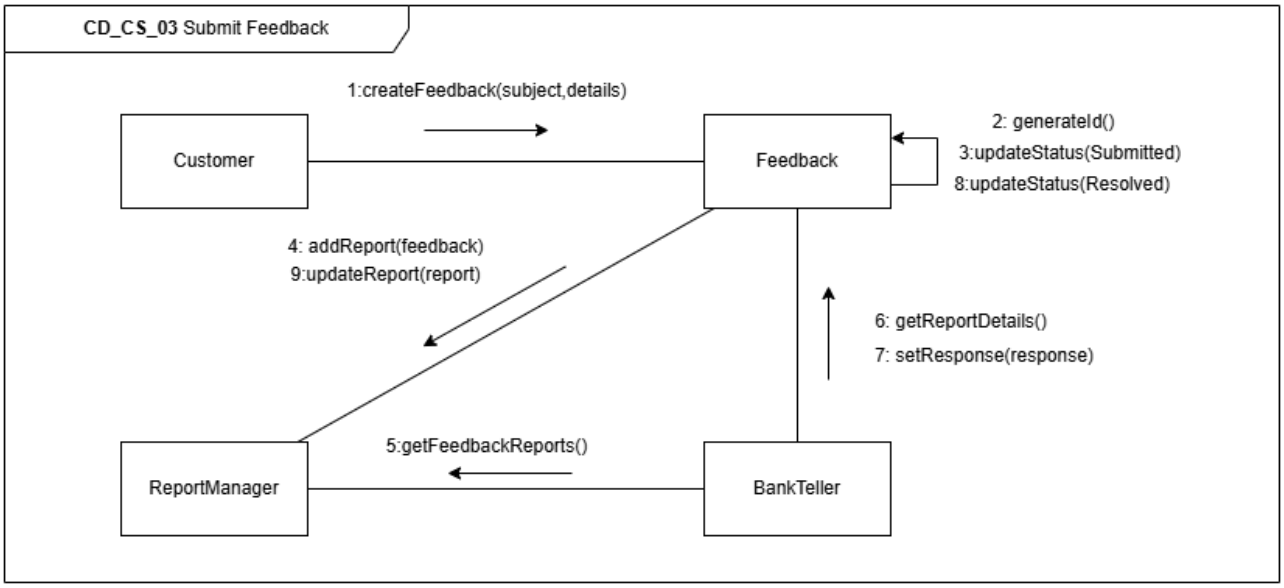

CD_CM_01 Apply for Cards
Customer
1:createCard(type,account)
Card
2: createCardApplication
(type,account,deliveryMethod)
4:updateStatus(status)
BankTeller
7.1:generateCardNumber()
7.2:generateSecurityCode()
CardApplication
5:<<Reviews>>
7:updateStatus(status)
3:createApplication(customer)
Application
8:getCardApplication()
6:getApplicationDetails()
ApplicationManager

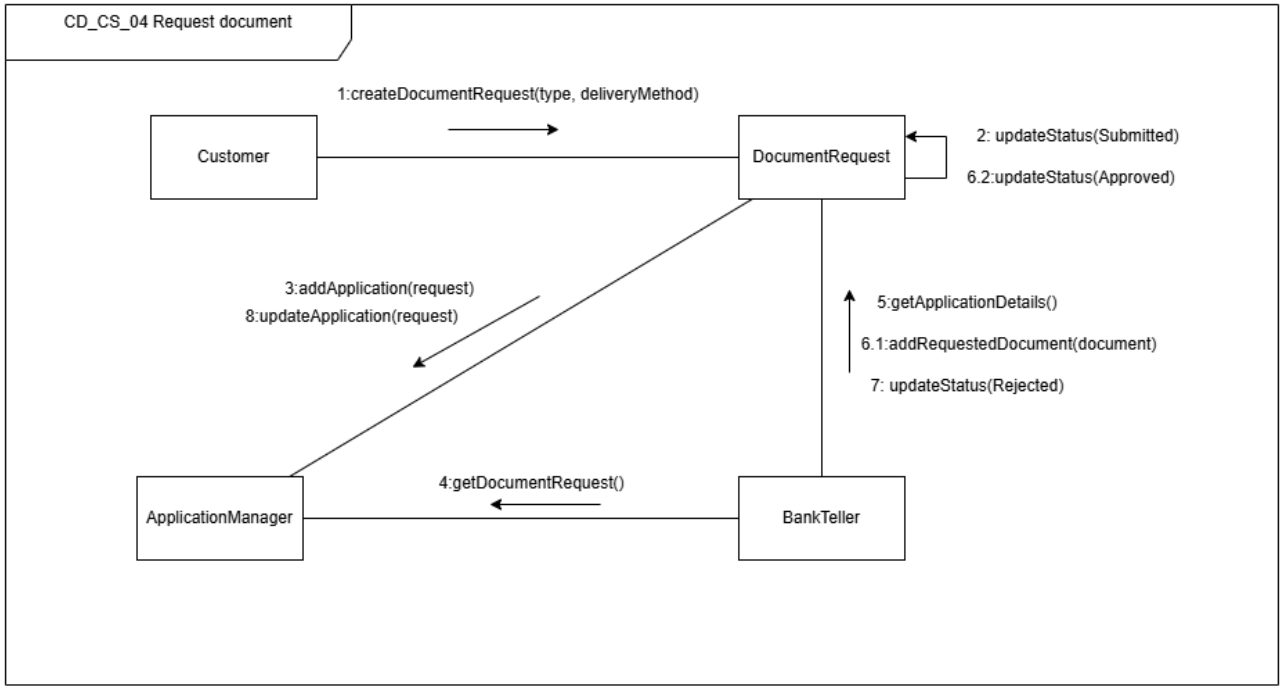

CD_CM_02 Request Checkbooks
2.1: [correct the information]
Customer
1: createCheckbookApplication
(account,numberOfChecks,deliveryMethod)
4:updateStatus(status)
BankTeller
CheckbookApplication
5:<<Reviews>>
8:updateStatus(status)
Application
2: [validate information]
6:getApplicationDetails()
7:getCheckboolApplication()
ApplicationManager


CD_CM_03 Set Card Limit
1:getCardDetails()
2:setLimit(newLimit)
Customer
Card
3: updateCard(card)
CardApplication


CD_CS_01 Access Bank-Related Information
1: getUnits()
2: getUnits(location,type)
Customer
UnitManager
3: getDetails()
BankUnit


CD_CS_02 Schedule Appointments
1: getAvailableSlots(location)
Customer
AppointmentCalendar
3: addAppointment(appointment)
2: createAppointment(service,location,startTime,endTime)
4:getAppointments(location)
Appointment
BankTeller
5: getAppointmentDetails()

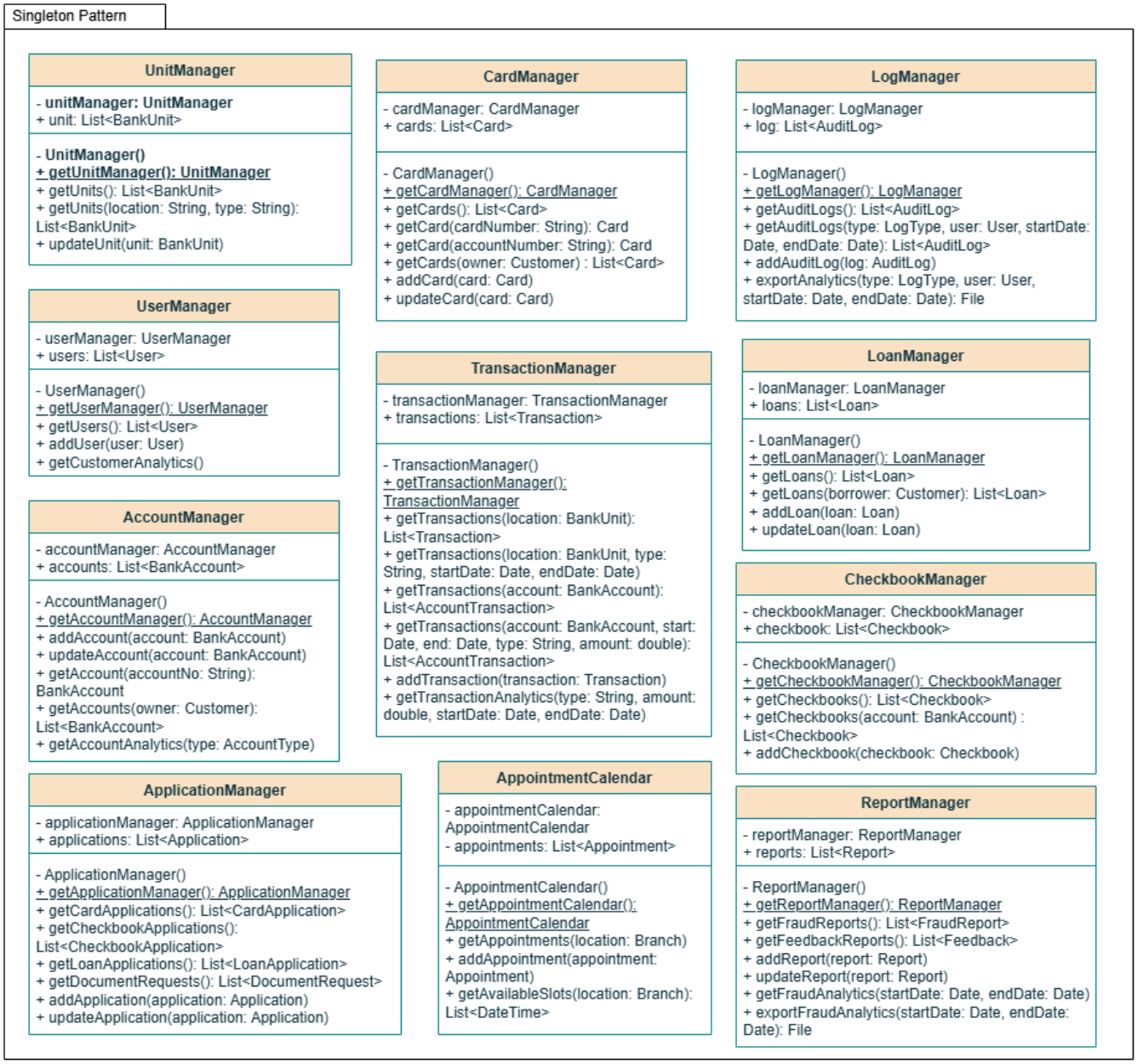

CD_CS_03 Submit Feedback
1:createFeedback(subject,details)
Customer
Feedback
2: generateId()
3:updateStatus(Submitted)
8:updateStatus(Resolved)
4: addReport(feedback)
9:updateReport(report)
6: getReportDetails()
7: setResponse(response)
5:getFeedbackReports()
ReportManager
BankTeller


CD_CS_04 Request document
1:createDocumentRequest(type, deliveryMethod)
Customer
DocumentRequest
2: updateStatus(Submitted)
6.2:updateStatus(Approved)
3:addApplication(request)
8:updateApplication(request)
5:getApplicationDetails()
6.1:addRequestedDocument(document)
7: updateStatus(Rejected)
4:getDocumentRequest()
ApplicationManager
BankTeller

Design Patterns

Singleton Pattern

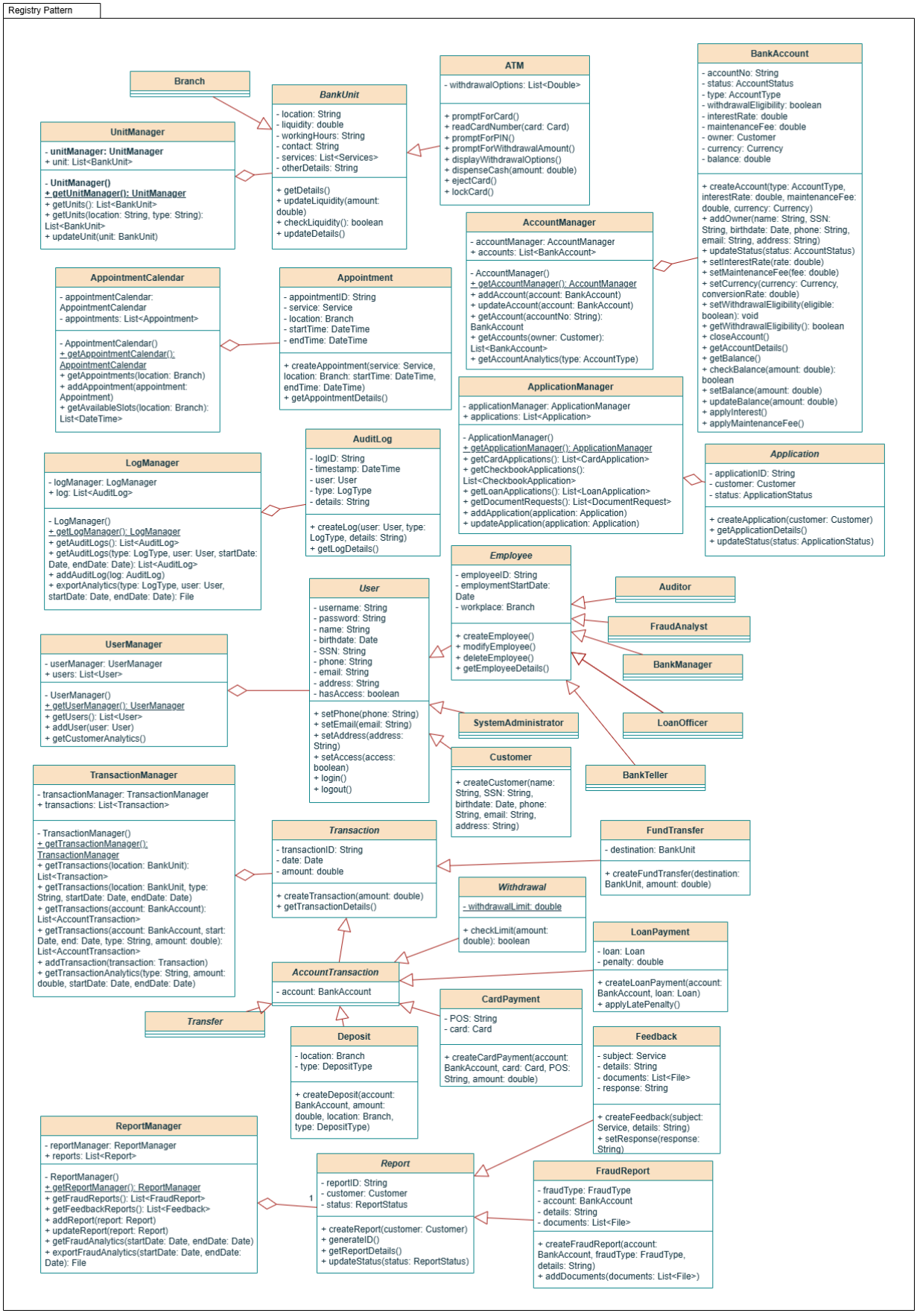
We used this pattern for class managers, since only one instance of each is needed to control the corresponding classes and client classes have a global access point to them.



Registry Pattern

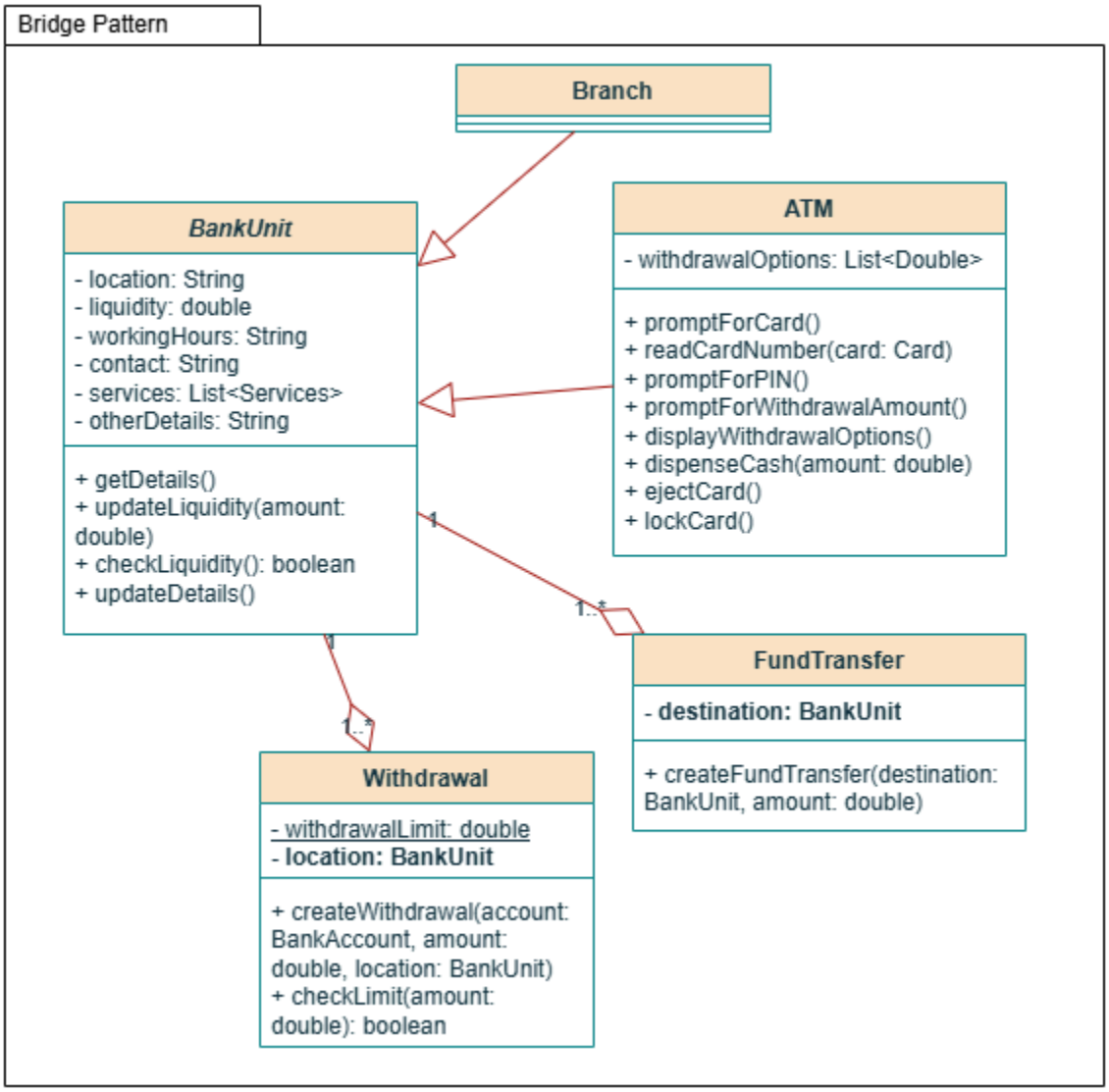
We applied the registry pattern to the manager classes, coupled with the singleton pattern, to manage the collections of objects from a single access point and to decouple object creation from their usage. For example, the *TransactionManager* class contains methods for managing objects of the *Transaction* class and all of its subclasses, such as adding a transaction to the collection or getting all transactions that fit a certain criteria.

Registry Pattern



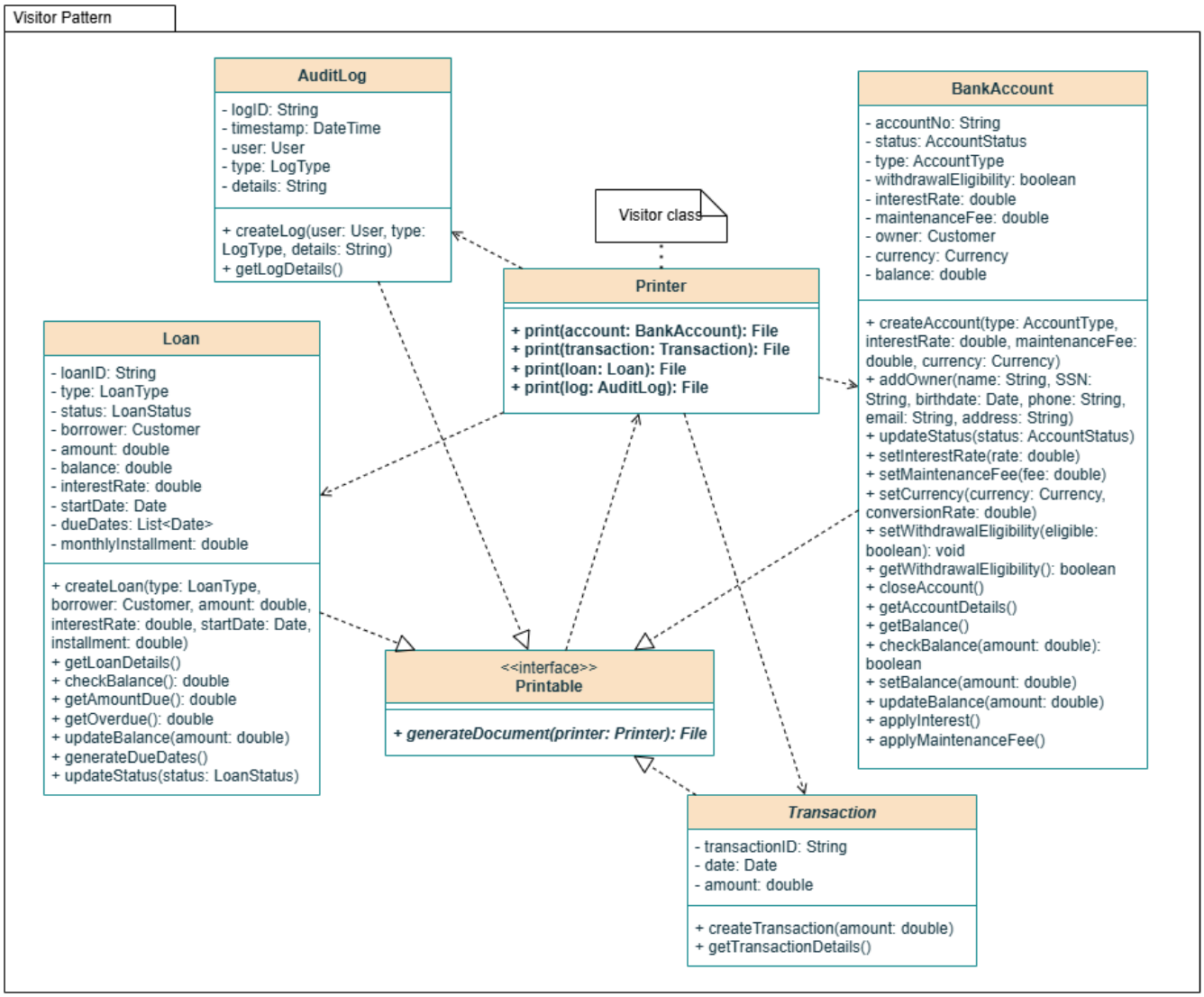
Bridge Pattern

To reduce the number of classes from inheritance, we used composition instead of inheritance. For example, instead of there being two separate concrete classes *BranchFundTransfer* and *ATMFundTransfer*, a single *FundTransfer* class was used, containing a field of the abstract type *BankUnit*, from which the *Branch* and *ATM* classes inherit. The same reasoning can be used to apply the bridge pattern to concrete classes *BranchWithdrawal* and *ATMWithdrawal*.



Visitor Pattern

Since the document generation behavior applies to several unrelated classes, it can be extracted into a separate visitor class called *Printer*, while the original classes implement the *Printable* interface with the generateDocument() method that accepts a *Printer* object. The *Printer* class contains methods to print documents that accept objects of the original concrete classes, so that they can use relevant data from the objects.


Visitor Pattern
AuditLog
- logID: String
- timestamp: DateTime
- user: User
- type: LogType
- details: String
+ createLog(user: User, type: LogType, details: String)
+ getLogDetails()
Visitor class
Printer
+ print(account: BankAccount): File
+ print(transaction: Transaction): File
+ print(loan: Loan): File
+ print(log: AuditLog): File
BankAccount
- accountNo: String
- status: AccountStatus
- type: AccountType
- withdrawalEligibility: boolean
- interestRate: double
- maintenanceFee: double
- owner: Customer
- currency: Currency
- balance: double
+ createAccount(type: AccountType, interestRate: double, maintenanceFee: double, currency: Currency)
+ addOwner(name: String, SSN: String, birthdate: Date, phone: String, email: String, address: String)
+ updateStatus(status: AccountStatus)
+ setInterestRate(rate: double)
+ setMaintenanceFee(fee: double)
+ setCurrency(currency: Currency, conversionRate: double)
+ setWithdrawalEligibility(eligible: boolean): void
+ getWithdrawalEligibility(): boolean
+ closeAccount()
+ getAccountDetails()
+ getBalance()
+ checkBalance(amount: double): boolean
+ setBalance(amount: double)
+ updateBalance(amount: double)
+ applyInterest()
+ applyMaintenanceFee()
Loan
- loanID: String
- type: LoanType
- status: LoanStatus
- borrower: Customer
- amount: double
- balance: double
- interestRate: double
- startDate: Date
- dueDates: List<Date>
- monthlyInstallment: double
+ createLoan(type: LoanType, borrower: Customer, amount: double, interestRate: double, startDate: Date, installment: double)
+ getLoanDetails()
+ checkBalance(): double
+ getAmountDue(): double
+ getOverdue(): double
+ updateBalance(amount: double)
+ generateDueDates()
+ updateStatus(status: LoanStatus)
<<interface>>
Printable
+ generateDocument(printer: Printer): File
Transaction
- transactionID: String
- date: Date
- amount: double
+ createTransaction(amount: double)
+ getTransactionDetails()