# Banking System

Software Design Specification

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Revision** | **Description** | **Author** |
| 04/02/2025 | 1.00 | Initial Version | Harven Dhanota |
| 04/02/2025 | 1.01 | Revision: Added Additional ATM Module and Teller Module Use Cases | Harven Dhanota |
| 04/03/2025 | 1.02 | Revision: Added Teller Module Use Case | Harven Dhanota |
| 04/03/2025 | 1.03 | Revision: Added More Teller Module Use Cases | Harven Dhanota |
| 04/03/2025 | 1.04 | Revision: Added Even More Teller Module Use Cases | Harven Dhanota |
| 04/04/2025 | 1.05 | Revision: Added Central Server Use Cases | Harven Dhanota |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Table of Contents

1. Diagrams [4](#__RefHeading___Toc19440719)

1.1. Use Case Diagram [4](#__RefHeading___Toc19440720)

1.2. Class Diagram [4](#__RefHeading___Toc19440721)

1.3. Sequence Diagram [4](#__RefHeading___Toc19440722)

2. Use Cases [5](#__RefHeading___Toc19440724)

2.1. ATM Module Use Cases [5](#__RefHeading___Toc19440725)

2.2. Teller Module Use Cases [5](#__RefHeading___Toc19440726)

2.3. Central Server Module Use Cases [5](#__RefHeading___Toc19440727)

# Diagrams

## Use Case Diagrams

TBD

## Class Diagrams

TBD

## Sequence Diagram

TBD

# Use Cases

## ATM Module Use Cases

2.1.1. Client Logging-In to ATM

Precondition(s): the ATM Module is online and connected to the central server

Postcondition(s): the customer has access to their bank account(s)

Basic Flow:

(1) the customer initiates a log-in request

(2) the ATM Module requests the user for their full name, phone number, and password

(3) the customer enters their name, phone number, and password

(4) the ATM Module sends the customer’s credentials to the central server

(5) the central server validates the customer’s credentials

(6) the central server checks if the customer’s accounts are currently being accessed

(7) the central server sends the customer’s account information back to the ATM Module

(8) the central server marks the account as being currently in access

(9) the ATM Module displays the customer’s account information to the customer

Alternate Flows:

(1) if the customer enters invalid credentials, the central server sends a failure message to the ATM Module, rather than sending the customer’s account information

(2) if an account is already being accessed and is attempted to be accessed again from another ATM Module or from a Teller Module, the central server sends a failure message to the ATM Module, rather than sending the customer’s account information

Exceptions:

(1) the central server does not receive any credentials

(2) the customer’s account information is not properly sent back to the ATM Module after the central server validates the customer’s credentials

(3) the account status is not set to “currently in access”

Related Use Case(s): 2.1.2, 2.1.3, 2.1.4, 2.1.5, 2.1.6

2.1.2. Cash Deposit into ATM

Precondition(s): the ATM Module is online and connected to the central server, and the customer is logged-in to their account

Postcondition(s): the customer’s account balance has increased by the amount deposited

Basic Flow:

(1) the customer selects an account

(2) the customer selects to deposit cash into that account

(3) the ATM Module prompts the customer to input how much cash they will deposit

(4) the ATM Module verifies that the cash to be deposited does not exceed the $4,000 daily limit for ATM cash deposits

(5) if the amount of cash to be deposited is valid, the ATM Module prompts the customer to insert their cash into the ATM

(6) the central server updates the amount of cash deposited by that customer, so that they cannot exceed the daily limit

(7) the central server updates the customer’s account balance

(8) the central server records the transaction into the account’s transaction history

Alternate Flows:

[none]

Exceptions:

(1) the central server does not properly update the customer’s account balance

(2) the central server does not properly update the amount of cash that the customer can deposit before they exceed their daily limit

(3) the central server does not properly record the transaction into the transaction history

Related Use Case(s): 2.1.1, 2.1.3, 2.3.1

2.1.3. Check Deposit into ATM

Precondition(s): the ATM Module is online and connected to the central server, and the customer is logged-in to their account

Postcondition(s): the customer’s account balance has increased by the amount deposited

Basic Flow:

(1) the customer selects an account

(2) the customer selects to deposit a check into that account

(3) the ATM Module prompts the user to insert their check into the ATM

(4) the ATM Module verifies that the amount on the check does not exceed the $10,000 daily limit for ATM check deposits

(5) if the amount on the check is valid, the central server updates the amount deposited in checks by that customer, so that they cannot exceed the daily limit

(6) the central server updates the customer’s account balance

(7) the central server records the transaction into the account’s transaction history

Alternate Flows:

[none]

Exceptions:

(1) the central server does not properly update the customer’s account balance

(2) the central server does not properly update the amount deposited in checks by that customer before they exceed their daily limit

(3) the central server does not properly record the transaction into the transaction history

Related Use Case(s): 2.1.1, 2.1.2, 2.3.1

2.1.4. Withdrawing Cash from ATM

Precondition(s): the ATM Module is online and connected to the central server, and the customer is logged-in to their account

Postcondition(s): the customer’s account balance has decreased by the amount withdrawn, and the customer now has cash equal to the amount withdrawn

Basic Flow:

(1) the customer selects an account

(2) the customer selects to withdraw cash from that account

(3) the ATM Module prompts the user to input the amount they want to withdraw

(4) the ATM Module verifies that the ATM has enough cash within its reserves to perform the withdrawal

(5) if the ATM has enough cash, the ATM Module sends the withdrawal request to the central server

(6) the central server temporarily updates the customer’s account balance

(7) the ATM gives the amount of cash requested to the customer

(8) the ATM Module sends a confirmation to the central server, acknowledging that the cash was successfully withdrawn

(9) the central server permanently updates the customer’s account balance

(10) the central server records the transaction into the account’s transaction history

Alternate Flows:

(1) if the customer tries to withdraw an amount greater than their current account balance, the central server sends a failure message to the ATM Module, rather than temporarily updating the customer’s account balance

(2) if the ATM does not have enough cash in its reserves to perform the withdrawal, the ATM Module displays a failure message, rather than send a request to the central server

Exceptions:

(1) the central server does not properly update the customer’s account balance

(2) the central server does not properly record the transaction into the transaction history

(3) the central server does not receive the withdrawal request from the ATM Module

(4) the ATM Module does not properly verify that there is enough cash in the ATM to perform the withdrawal

Related Use Case(s): 2.1.1, 2.3.2

2.1.5. Checking Account Balance

Precondition(s): the ATM Module is online and connected to the central server, and the customer is logged-in to their account

Postcondition(s): the customer can view their account balance

Basic Flow:

(1) the customer selects an account

(2) the customer selects to view their account balance

(3) the ATM sends a request to the central server

(4) the central server sends the requested account balance back to the ATM Module

Alternate Flows:

[none]

Exceptions:

(1) the central server does not receive the request from the ATM Module

Related Use Case(s): 2.1.1

2.1.6. Client Logging-Out of ATM

Precondition(s): the ATM Module is online and connected to the central server, and the customer is logged-in to their account

Postcondition(s): the customer is logged out of their account

Basic Flow:

(1) the customer initiates a log-out request

(2) the ATM Module prompts the customer to confirm that they want to log out of the ATM

(3) if the customer confirms the log-out request, then the ATM Module sends the request to the central server to log the user out

(3) the central server marks the account as no longer being in access

(4) the central server sends a confirmation back to the ATM Module to notify the user that they are no longer logged in

(5) the ATM Module returns to the log-in page, ready for the next customer to log in

Alternate Flows:

(1) if the customer does not confirm that they want to log out, then the log-out request is canceled and nothing is sent to the central server

Exceptions:

(1) the central server does not receive the log-out request

(2) the confirmation message is not properly sent back to the ATM Module after the account is marked as not being in access

(3) the account status is not set to “no longer in access”

Related Use Case(s): 2.1.1

## Teller Module Use Cases

2.2.1. Teller Logging-In to Teller Module

Precondition(s): the Teller Module is online and connected to the central server

Postcondition(s): the bank employee has access to the Teller Module

Basic Flow:

(1) the bank employee initiates a log-in request

(2) the Teller Module requests the user for their bank-issued Employee ID and their password

(3) the bank employee enters their Employee ID and password

(4) the Teller Module sends the employee’s credentials to the central server

(5) the central server validates the employee’s credentials

(6) if the credentials are correct, the central server sets the teller’s status to logged-in

(7) the central server sends all customer data back to the Teller Module where the log-in request was initiated

Alternate Flows:

(1) if the bank employee enters invalid credentials, the central server sends a failure to the Teller Module, rather than sending all customer data

Exceptions:

(1) the central server does not receive any credentials

(2) the customer data is not properly sent back to the Teller Module after the central server validates the employee’s credentials

(3) the teller’s status is not set to “currently logged in”

Related Use Case(s): 2.2.2, 2.2.3, 2.2.4

2.2.2. Deposit by Teller

Precondition(s): the Teller Module is online and connected to the central server, and the teller has verified the customer

Postcondition(s): the customer’s account balance has increased by the amount deposited

Basic Flow:

(1) the teller selects a customer’s account

(2) the central server checks if the customer’s account is currently being accessed

(3) the central server sends the customer’s account information back to the Teller Module

(4) the teller selects to deposit either cash or a check into that account

(5) the teller takes the cash or check from the customer

(6) the Teller Module prompts the teller to input the amount being deposited

(7) the Teller Module sends the deposit request to the central server

(8) the central server updates the customer’s account balance, increasing it by the amount deposited

(9) the central server records the transaction into the account’s transaction history

(10) the teller exits the customer’s account

Alternate Flows:

(1) if an account is already being accessed and is attempted to be accessed again from another Teller Module or from an ATM Module, the central server sends a failure message to the ATM Module, rather than sending the customer’s account information

Exceptions:

(1) the central server does not properly update the customer’s account balance

(2) the central server does not receive the deposit request from the Teller Module

(3) the central server does not properly record the transaction into the transaction history

Related Use Case(s): 2.2.1, 2.3.1

2.2.3. Withdrawing by Teller

Precondition(s): the Teller Module is online and connected to the central server, and the teller has verified the customer

Postcondition(s): the customer’s account balance has decreased by the amount withdrawn, and the customer now has cash equal to the amount withdrawn

Basic Flow:

(1) the teller selects a customer’s account

(2) the central server checks if the customer’s account is currently being accessed

(3) the central server sends the customer’s account information back to the Teller Module

(4) the teller selects to withdraw cash from that account

(5) the Teller Module prompts the teller to input the amount being withdrawn

(6) the Teller Module sends the withdrawal request to the central server

(7) the central server temporarily updates the customer’s account balance, and waits for the teller to confirm that they have given the cash to the customer

(8) once the cash has been given to the customer, the teller sends a confirmation message (via the Teller Module) to the central server

(9) the central server permanently updates the customer’s account balance

(10) the central server records the transaction into the account’s transaction history

(11) the teller exits the customer’s account

Alternate Flows:

(1) if an account is already being accessed and is attempted to be accessed again from another Teller Module or from an ATM Module, the central server sends a failure message to the ATM Module, rather than sending the customer’s account information

(2) if the teller tries to withdraw an amount greater than the customer’s current account balance, the central server sends a failure message to the Teller Module, rather than temporarily updating the customer’s account balance

Exceptions:

(1) the central server does not properly update the customer’s account balance

(2) the central server does not properly record the transaction into the transaction history

(3) the central server does not receive the withdrawal request from the Teller Module

Related Use Case(s): 2.2.1, 2.3.2

2.2.4. Teller Logging-Out of Teller Module

Precondition(s): the Teller Module is online and connected to the central server, and the teller is logged-in to the Teller Module

Postcondition(s): the teller is logged-out of the Teller Module

Basic Flow:

(1) the teller initiates a log-out request

(2) the Teller Module prompts the teller to confirm that they want to log out of the module

(3) if the teller confirms the log-out request, then the Teller Module sends the request to the central server to log the teller out

(4) if the teller is logged in to any customer accounts, then the customer account will automatically be logged out of before the teller is logged out of the Teller Module

(5) the central server sets the teller’s status to logged out

(6) the central server sends a confirmation back to the Teller Module to notify the teller that they are no longer logged in

(7) the Teller Module returns to the log-in page, ready for the next teller to log in

Alternate Flows:

(1) if the teller does not confirm that they want to log out, then the log-out request is canceled and nothing is sent to the central server

Exceptions:

(1) the central server does not receive the log-out request

(2) the confirmation message is not properly sent back to the Teller Module after the teller’s status is set to logged out

(3) the teller’s status is not set to “logged out”

Related Use Case(s): 2.2.1

2.2.5. Teller Creates New Customer Account

Precondition(s): the Teller Module is online and connected to the central server, and the teller is logged-in to the Teller Module

Postcondition(s): the customer now has an account with the bank

Basic Flow:

(1) the teller initiates an account creation request

(2) the Teller Module prompts the teller to input the full name and phone number of the customer

(3) the Teller Module prompts the teller for a password for the customer’s account

(4) the customer themselves inputs a password for their account

(5) the Teller Module sends the customer’s name, phone number, and password to the central server

(6) the central server checks if there already exists an account with the provided name and phone number

(7) if no account exists, then the central server creates a new bank account with the provided credentials

(8) the central server sends a confirmation message back to the Teller Module to notify the teller that a new account was successfully created

Alternate Flows:

(1) if an account already exists with the provided name and phone number, then the central server sends a message back to the Teller Module to notify the teller than an account already exists, rather than creating a new bank account with the provided credentials

Exceptions:

(1) the central server does not properly receive the customer’s credentials

(2) the central server does not find a customer’s existing account before creating a new account, assuming the customer does have an existing account

(3) the confirmation message is not properly sent back to the Teller Module after the new account is created

Related Use Case(s): 2.2.1, 2.2.6

2.2.6. Adding Additional Users to Customer Accounts

Precondition(s): the Teller Module is online and connected to the central server, and the teller has verified the identities of both the customer who owns the account and the customer that wants to be added to that account

Postcondition(s): an additional user is added to the account, and that user now has full access to the account they were added to

Basic Flow:

(1) the teller selects a customer’s account

(2) the central server checks if the customer’s account is currently being accessed

(3) the central server sends the customer’s account information back to the Teller Module

(4) the teller selects to add an additional user to the account

(5) the Teller Module prompts the teller to input the full name and phone number of the user that wants to be added to the account

(6) the Teller Module prompts the teller for a password for the user being added to the account

(7) the user themselves inputs their own password for accessing the account

(8) the Teller Module sends the credentials to the central server

(9) the central server adds these credentials to the account that the user wants to be added to

(10) the central server sends a confirmation message back to the Teller Module to notify the teller that the user was successfully added to the account

(11) the teller exits the customer’s account

Alternate Flows:

[none]

Exceptions:

(1) the central server does not properly receive the credentials of the user that wants to be added to the account

(2) the central server does not properly add the user’s credentials to the customer’s account

(3) the confirmation message is not properly sent back to the Teller Module after the user’s credentials are added to the account

Related Use Case(s): 2.2.1, 2.2.5, 2.2.7

2.2.7. Removing Additional Users from Customer Accounts

Precondition(s): the Teller Module is online and connected to the central server, and the teller has verified the identity of the customer who owns the account

Postcondition(s): an additional user is removed from the account, and that user can no longer access the account they were removed from

Basic Flow:

(1) the teller selects a customer’s account

(2) the central server checks if the customer’s account is currently being accessed

(3) the central server sends the customer’s account information back to the Teller Module

(4) the teller selects to remove an additional user from the account

(5) the Teller Module prompts the teller to select the name of the user that they want to remove from the account

(6) the Teller Module prompts the teller to confirm that they want to remove the selected user from the account

(7) if the teller confirms they want the user removed, then the Teller Module sends a request to the central server

(8) the central server removes the user from the account

(9) the central server sends a confirmation message back to the Teller Module to notify the teller that the user was successfully removed from the account

(10) the teller exits the customer’s account

Alternate Flows:

(1) if the teller does not confirm that they want the user removed, then nothing is sent to the central server and the process of removing a user is canceled altogether

Exceptions:

(1) the central server does not properly receive the name of the user that needs to be removed from the account

(2) the confirmation message is not properly sent back to the Teller Module after the user’s credentials are removed from the account

(3) the central server does not properly remove the user’s credentials from the account

Related Use Case(s): 2.2.1, 2.2.6

2.2.8. Blocking Accounts

Precondition(s): the Teller Module is online and connected to the central server, and the teller has verified the identity of the customer who owns the account to be blocked

Postcondition(s): the customer’s account is blocked, and it can no longer be accessed from any ATM Module

Basic Flow:

(1) the teller selects a customer’s account

(2) the teller selects to block access to the account

(3) the Teller Module prompts the teller to confirm that they want to block the account

(4) if the teller confirms that they want to block the account, then a block request is sent to the central server

(5) the central server marks the account as being blocked from access

(6) if any users are logged-in to the account when it is blocked, they will be automatically logged-out with no confirmation prompt

(7) the central server sends a confirmation message back to the Teller Module to notify the teller that the account was successfully blocked

Alternate Flows:

(1) if the teller does not confirm that they want to block the account, then nothing is sent to the central server and the process of blocking an account is canceled altogether

Exceptions:

(1) the central server does not properly block the customer’s account

(2) the central server does not automatically log-out of the customer’s account after it has been blocked

(3) the confirmation message is not properly sent back to the Teller Module after the customer’s account has been blocked

Related Use Case(s): 2.2.1, 2.2.9

2.2.9. Unblocking Accounts

Precondition(s): the Teller Module is online and connected to the central server, and the teller has verified the identity of the customer who owns the account to be unblocked

Postcondition(s): the customer’s account is unblocked, and it can now be accessed from any ATM Module

Basic Flow:

(1) the teller selects a customer’s account

(2) the teller selects to unblock access to the account

(3) the Teller Module prompts the teller to confirm that they want to unblock the account

(4) if the teller confirms that they want to unblock the account, then an unblock request is sent to the central server

(5) the central server marks the account as being accessible, effectively unblocking access to the account

(6) the central server sends a confirmation message back to the Teller Module to notify the teller that the account was successfully unblocked

Alternate Flows:

(1) if the teller does not confirm that they want to unblock the account, then nothing is sent to the central server and the process of unblocking an account is canceled altogether

Exceptions:

(1) the central server does not properly unblock the customer’s account

(2) the confirmation message is not properly sent back to the Teller Module after the customer’s account has been unblocked

Related Use Case(s): 2.2.1, 2.2.8

## Central Server Module Use Cases

2.3.1. Record Transaction History: Deposits

Precondition(s): a deposit is made by a customer, either directly through an ATM or indirectly through a teller

Postcondition(s): the deposit is recorded in the account’s transaction history

Basic Flow:

(1) a deposit is made into a customer’s account

(2) the deposit request is sent either by the ATM Module or the Teller Module to the central server

(3) the central server records the date and time that the deposit was made, the amount deposited, whether the deposit was cash or check, and whether the deposit was performed through an ATM or by a teller, into a comma-separated text file that is stored in the central server

Alternate Flows:

[none]

Exceptions:

(1) the data is not properly recorded in the comma-separated text file

Related Use Case(s): 2.1.2, 2.1.3, 2.2.2

2.3.2. Record Transaction History: Withdrawals

Precondition(s): a withdrawal is made by a customer, either directly through an ATM or indirectly through a teller

Postcondition(s): the deposit is recorded in the account’s transaction history

Basic Flow:

(1) a withdrawal is made from a customer’s account

(2) the withdrawal request is sent either by the ATM Module or the Teller Module to the central server

(3) the central server records the date and time that the withdrawal was made, the amount withdrawn, and whether the withdrawal was performed through an ATM or by a teller, into a comma-separated text file that is stored in the central server

Alternate Flows:

[none]

Exceptions:

(1) the data is not properly recorded in the comma-separated text file

Related Use Case(s): 2.1.4, 2.2.3

2.3.3. Record History: Account Creation

Precondition(s): a new account is created by a teller

Postcondition(s): the account’s creation is recorded in the account’s history

Basic Flow:

(1) a new account is made by a teller for a customer

(2) the account credentials are sent to the central server from the Teller Module

(3) the central server records the credentials, including the customer’s full name, phone number, and password, along with the date and time that the account was created, into a text file that is stored in the central server

Alternate Flows:

[none]

Exceptions:

(1) the data is not properly recorded in the text-file

Related Use Case(s): 2.2.5

2.3.4. Record History: Account Blocking

Precondition(s): an account is blocked by a teller, with the customer’s approval

Postcondition(s): the block is recorded in the account’s history

Basic Flow:

(1) a customer’s account is blocked by a teller

(2) the block request is sent to the central server from the Teller Module

(3) the central server sets a flag on the account to mark it as blocked, meaning that no users can access that account from an ATM Module or a Teller Module until the account is unblocked

(4) the central server records the date and time that the block was performed, along with the ID of the teller that performed the block, into a text file

Alternate Flows:

[none]

Exceptions:

(1) the data is not properly recorded in the text file

(2) the account is not properly flagged as being blocked

Related Use Case(s): 2.2.8

2.3.5. Record History: Account Unblocking

Precondition(s): an account is unblocked by a teller, with the customer’s approval

Postcondition(s): the unblock is recorded in the account’s history

Basic Flow:

(1) a customer’s account is unblocked by a teller

(2) the unblock request is sent to the central server from the Teller Module

(3) the central server removes the flag on the account that marks it as being blocked

(4) the central server records the date and time that the unblock was performed, along with the ID of the teller that performed the unblock, into a text file

Alternate Flows:

[none]

Exceptions:

(1) the data is not properly recorded in the text file

(2) the account is not properly set to being unblocked

Related Use Case(s): 2.2.9

2.3.6.

Precondition(s):

Postcondition(s):

Basic Flow:

(1)

Alternate Flows:

(1)

Exceptions:

(1)

Related Use Case(s):