# 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 |
| 04/04/2025 | 1.06 | Revision: Clarified Distinctions Between User Account and Financial Account | Harven Dhanota |
| 04/04/2025 | 1.07 | Revision: Added Another ATM Module Use Case | Harven Dhanota |
| 04/05/2025 | 1.08 | Revision: Added Additional ATM Module and Teller Module Use Cases | Harven Dhanota |
| 04/07/2025 | 1.09 | Revision: Modified Use Cases 2.2.6, 2.2.7, and 2.2.12; Deleted “Check Deposit” Use Case from ATM Module | Harven Dhanota |
| 04/07/2025 | 1.10 | Revision: Added references for design docs, added ATM & Teller use cases, added stubs for more central module sue cases | Jaishnoor Kaur |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

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)

1.4. FLOW/UI PROTOTYPE……………………………………………………………………………………..4

2. Use Cases [5](#__RefHeading___Toc19440724)

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

2.2. Teller Module Use Cases [8](#__RefHeading___Toc19440726)

2.3. Central Server Module Use Cases 16

# Diagrams

## Use Case Diagrams

TBD

## Class Diagrams

UML\_Class\_Diagrams.png

## Sequence Diagram

TBD

## Flow/UI Prototype

GUI\_or\_Flow\_Prototype/ATM.png

GUI\_or\_Flow\_Prototype/Teller.png

# Use Cases

## ATM Module Use Cases

2.1.1. Customer Logging-In to ATM

Relevant Requirement(s) from SRS:

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

Postcondition(s): the customer has access to their financial 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 user account is currently being accessed

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

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

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

(2) if a user 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 financial account information

Exceptions:

(1) the central server does not receive any credentials

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

(3) the customer’s user account status is not set to “currently in access”

Related Use Case(s):

2.1.2. Deposit into ATM

Relevant Requirement(s) from SRS:

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

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

Basic Flow:

(1) the customer selects a financial account

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

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

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

(5) if the amount 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 deposited by that customer, so that they cannot exceed the daily limit

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

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

Alternate Flows:

[none]

Exceptions:

(1) the central server does not properly update the customer’s financial 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):

2.1.3. Withdrawing from ATM

Relevant Requirement(s) from SRS:

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

Postcondition(s): the customer’s financial 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 a financial account

(2) the customer selects to withdraw from that financial 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 financial 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 financial account balance

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

Alternate Flows:

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

Exceptions:

(1) the central server does not properly update the customer’s financial 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):

2.1.4. Checking Balance in Financial Accounts

Relevant Requirement(s) from SRS:

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

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

Basic Flow:

(1) the customer selects a financial account

(2) the customer selects to view the 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):

2.1.5. Customer Logging-Out of ATM

Relevant Requirement(s) from SRS:

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

Postcondition(s): the customer is logged out of their user 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 user account as no longer being in access

(4) the central server sends a confirmation back to the ATM Module to notify the customer 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 user account is marked as not being in access

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

Related Use Case(s):

2.1.6. Automatic Log-Out from ATM

Relevant Requirement(s) from SRS:

Precondition(s): the ATM Module is online and connected to the central server, the customer is logged-in to their user account, and there is no activity or input from the customer for 60 seconds

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

Basic Flow:

(1) there is no input or activity from the customer for 60 seconds

(2) the ATM Module prompts the customer to confirm that they are still present at the ATM, and a visible 10-second timer is shown to the user

(3) if the customer does not confirm that they are still present before the 10-second timer runs out, then the ATM Module sends a request to the central server to automatically log out the user

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

(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 confirm that they are still present at the ATM before the 10-second timer runs out, then nothing is sent to the central server and the automatic log-out process is canceled altogether

Exceptions:

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

(2) the customer’s user account status is not set to “no longer in access”

Related Use Case(s):

2.1.7. View Transaction History

Relevant Requirement(s) from SRS:

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

Postcondition(s): the customer can view all transactions that have occurred on one of their financial accounts

Basic Flow:

(1) the customer selects a financial account

(2) the customer selects to view the transaction history

(3) a request is sent to the central server to get the transaction history associated with the financial account that the customer selected

(4) the transaction history is sent back to the ATM Module from the central server

(5) the transaction history associated with the financial account selected is shown to the customer

Alternate Flows:

[none]

Exceptions:

(1) the central server does not receive the request to access financial account’s transaction history

(2) the transaction history is not properly sent back to the ATM Module

Related Use Case(s):

2.1.8 Customer Select a Specific Financial Account

Relevant Requirement(s) from SRS:

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

Postcondition(s): the customer can view all options for selected financial account

Basic Flow:

(1) the customer selects a financial account

(2) the ATM Module sends a ACCESSBAREQ message to central module

(3) central module verifies that financial account is not currently in access

(4) central module sends ACCESSBAREQGRANTED message and updates in\_access to true

(5) ATM Module displays options (withdraw, deposit, view transaction history, view balance) to user

Alternate Flows:

1. if financial account is already in access, central module sends ERROR message to ATM

Exceptions:

(1) the central server does not receive the request to access financial account

(2) messages time out in ATM module

Related Use Case(s):

2.1.9 Customer Exits Current Financial Account

Relevant Requirement(s) from SRS:

Precondition(s): the customer is logged-in to their user account and is currently accessing the financial account

Postcondition(s): the customer is returned to view that lists all their financial accounts

Basic Flow:

(1) the customer clicks exit, triggering GUI event listener

(2) the ATM Module sends a EXITBAREQ message to central module

(3) the central module updates in\_access to true and sends EXITBAREQGRANTED message to ATM module

(5) GUI closes window and ATM Module displays returns to previous screen

Alternate Flows:

Exceptions:

(1) the central server does not receive the request to exit financial account

(2) messages time out in ATM module

Related Use Case(s):

## Teller Module Use Cases

2.2.1. Teller Logging-In to Teller Module

Relevant Requirement(s) from SRS:

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 user account 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 message to the Teller Module, rather than sending all customer data

Exceptions:

(1) the central server does not receive any credentials

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

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

Related Use Case(s):

2.2.2. Deposit by Teller

Relevant Requirement(s) from SRS:

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

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

Basic Flow:

(1) the teller selects a customer’s user account

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

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

(4) the teller selects one of the customer’s financial accounts

(5) the teller selects to deposit into that financial account

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

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

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

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

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

(11) the teller exits the customer’s user account

Alternate Flows:

(1) if a user 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 Teller Module, rather than sending the customer’s financial account information

Exceptions:

(1) the central server does not properly update the customer’s financial 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):

2.2.3. Withdrawing by Teller

Relevant Requirement(s) from SRS:

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

Postcondition(s): the customer’s financial 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 user account

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

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

(4) the teller selects one of the customer’s financial accounts

(5) the teller selects to withdraw cash from that financial account

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

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

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

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

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

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

(12) the teller exits the customer’s user account

Alternate Flows:

(1) if a user 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 Teller Module, rather than sending the customer’s financial account information

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

Exceptions:

(1) the central server does not properly update the customer’s financial 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):

2.2.4. Teller Logging-Out of Teller Module

Relevant Requirement(s) from SRS:

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’s user account, then the user 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):

2.2.5. Teller Creates New User Account for a Customer

Relevant Requirement(s) from SRS:

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 a user account

Basic Flow:

(1) the teller initiates a user 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 user account

(4) the customer themselves inputs a password for their user 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 a user account with the provided name and phone number

(7) if no user account exists, then the central server creates a new user 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 user account was successfully created

Alternate Flows:

(1) if a user 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 a user account already exists, rather than creating a new user 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 user account before creating a new user account, assuming the customer does have an existing user account

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

Related Use Case(s):

2.2.6. Adding Additional Users to a Financial Account

Relevant Requirement(s) from SRS:

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 financial account and the customer that wants to be added to that financial account

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

Basic Flow:

(1) the teller selects a customer’s user account

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

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

(4) the teller selects one of the customer’s financial accounts

(5) the teller selects to add an additional user to that financial account

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

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

(8) if the user being added has a user account, then the financial account they are being added to is added to their user account

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

(10) the teller exits the customer’s user account

Alternate Flows:

(1) if the user being added does not have a user account, then the central server sends a message back to the Teller Module to notify the teller that the user being added does not have a user account, and the process of adding a user to the financial account is canceled altogether

Exceptions:

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

(2) the central server does not properly add the new user to the financial account

(3) the confirmation message is not properly sent back to the Teller Module after the user is added to the financial account

Related Use Case(s):

2.2.7. Removing Additional Users from a Financial Account

Relevant Requirement(s) from SRS:

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 financial account

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

Basic Flow:

(1) the teller selects a customer’s user account

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

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

(4) the teller selects the customer’s financial account

(4) the teller selects to remove a user from that financial account

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

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

(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 financial account

(10) the teller exits the customer’s user 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 financial account

(2) the confirmation message is not properly sent back to the Teller Module after the is removed from the financial account

(3) the central server does not properly remove the user from the financial account

Related Use Case(s):

2.2.8. Blocking User Accounts

Relevant Requirement(s) from SRS:

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 user account to be blocked

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

Basic Flow:

(1) the teller selects a customer’s user account

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

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

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

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

(6) if any users are logged-in to the user 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 user account was successfully blocked

Alternate Flows:

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

Exceptions:

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

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

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

Related Use Case(s):

2.2.9. Unblocking Accounts

Relevant Requirement(s) from SRS:

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 user account to be unblocked

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

Basic Flow:

(1) the teller selects a customer’s user account

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

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

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

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

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

Alternate Flows:

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

Exceptions:

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

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

Related Use Case(s):

2.2.10. View Transaction History

Relevant Requirement(s) from SRS:

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

Postcondition(s): the customer can view all transactions that have occurred on one of their financial accounts

Basic Flow:

(1) the teller selects a customer’s user account

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

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

(4) the teller selects one of the customer’s financial accounts

(5) the teller selects to view the transaction history of that financial account

(6) a request is sent to the central server to get the transaction history associated with the financial account that the teller selected

(7) the transaction history is sent back to the Teller Module from the central server

(8) the transaction history associated with the financial account selected is shown to the teller

Alternate Flows:

[none]

Exceptions:

(1) the central server does not receive the request to access financial account’s transaction history

(2) the transaction history is not properly sent back to the Teller Module

Related Use Case(s):

2.2.11. Update User Account Information

Relevant Requirement(s) from SRS:

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

Postcondition(s): the customer’s user account information is updated

Basic Flow:

(1) the teller selects a customer’s user account

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

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

(4) the teller selects to edit/modify the customer’s user account information

(5) the teller updates the user account information, such as full name, mailing address, phone number, etc.

(6) once the teller has finished updating the user account information, the Teller Module sends the information to the central server

(7) the central server updates the user account information

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

Alternate Flows:

[none]

Exceptions:

(1) the central server does not properly update the user account information

(2) the central server does not send the confirmation message back to the Teller Module after the user account information is updated

Related Use Case(s):

2.2.12. Teller Deletes Financial Account for a Customer

Relevant Requirement(s) from SRS:

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

Postcondition(s): the customer now has one less financial account

Basic Flow:

(1) the teller selects a customer’s user account

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

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

(4) the teller selects one of the customer’s financial accounts

(5) the teller selects to delete that financial account

(6) the Teller Module prompts the teller to confirm that they want to delete the user account

(7) if the teller confirms that they want to delete the user account, then the request is sent to the central server

(8) if the financial account has a balance of zero, then the central server deletes the files associated with that financial account

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

Alternate Flows:

(1) if the teller does not confirm that they want to delete the financial account, then nothing is sent to the central server and the process of deleting a financial account is canceled altogether

(3) if the financial account does not have a balance of zero, then the central server sends a message notifying the teller that the financial account has a non-zero balance, and that they must withdraw that money before the financial account can be deleted

Exceptions:

(1) the central server does not properly delete the financial account files

(2) the central server does not verify that the financial account has a balance of zero before deleting the files

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

Related Use Case(s):

2.2.13 Teller Selects a Specific Financial Account

Relevant Requirement(s) from SRS:

Precondition(s): the teller is logged-in to a user account

Postcondition(s): the teller can view all teller options for selected financial account

Basic Flow:

(1) the teller selects a financial account

(2) the Teller Module sends a ACCESSBAREQ message to central module

(3) central module verifies that financial account is not currently in access

(4) central module sends ACCESSBAREQGRANTED message and updates in\_access to true

(5) Teller Module displays options (withdraw, deposit, view transaction history, view balance, add/remove user, view users) to teller

Alternate Flows:

1. if financial account is already in access, central module sends ERROR message to Teller module

Exceptions:

(1) the central server does not receive the request to access financial account

(2) messages time out

Related Use Case(s):

2.2.14 Teller Exits Current Financial Account

Relevant Requirement(s) from SRS:

Precondition(s): the teller is currently accessing the financial account

Postcondition(s): the teller is returned to view that lists all of a user’s financial accounts

Basic Flow:

(1) the teller clicks exit, triggering GUI event listener

(2) the ATellerTM Module sends a EXITBAREQ message to central module

(3) the central module updates in\_access to true and sends EXITBAREQGRANTED message to Teller module

(5) GUI closes window and Teller Module displays returns to previous screen

Alternate Flows:

Exceptions:

(1) the central server does not receive the request to exit financial account

(2) messages time out

Related Use Case(s):

2.2.15 Teller Selects a Specific User Account

Relevant Requirement(s) from SRS:

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

Postcondition(s): the teller can view all options for suer account

Basic Flow:

(1) the teller selects a financial account by clicking a user account or manually entering user details and clicking select

(2) the Teller Module sends a ACCESSCAREQ message to central module

(3) central module verifies that user account is not currently in access

(4) central module sends ACCESSCAREQGRANTED message and updates in\_access to true

(5) teller Module displays options (update suer information, create new bank account, select financial account)

Alternate Flows:

1. if user account is already in access, central module sends ERROR message to Teller

Exceptions:

(1) the central server does not receive the request

(2) messages time out

Related Use Case(s):

2.2.16 Teller Exits Current User Account

Relevant Requirement(s) from SRS:

Precondition(s): the teller is currently accessing the user account

Postcondition(s): the telelr is returned to view that lists all user accounts

Basic Flow:

(1) the telelr clicks exit, triggering GUI event listener

(2) the Teller Module sends a EXITCAREQ message to central module

(3) the central module updates in\_access to true and sends EXITCAREQGRANTED message to Teller module

(5) GUI closes window and Teller Module displays returns to previous screen

Alternate Flows:

Exceptions:

(1) the central server does not receive the request to exit user account

(2) messages time out in ATM module

Related Use Case(s):

2.2.17 View Account Level History

Relevant Requirement(s) from SRS:

Precondition(s): the teller is currently logged-in

Postcondition(s): the teller can view account level history

Basic Flow:

(1) the teller selects option to view account level history, triggering GUI event listener

(2) the Teller Module sends a VIEWACCLVLHISTREQ message to central module

(3) the central module sends VIEWACCLVLHISTREQGRANTED message to Teller module and account level history as payload

(5) Teller Module displays account level history

Alternate Flows:

Exceptions:

(1) the central server does not receive the request

(2) messages time out

(3) history data is corrupted

Related Use Case(s):

## Central Server Module Use Cases

2.3.1. Record Transaction History: Deposits

Relevant Requirement(s) from SRS:

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 user account’s transaction history

Basic Flow:

(1) a deposit is made into a customer’s financial 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 associated with the financial account

Alternate Flows:

[none]

Exceptions:

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

Related Use Case(s):

2.3.2. Record Transaction History: Withdrawals

Relevant Requirement(s) from SRS:

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 user account’s transaction history

Basic Flow:

(1) a withdrawal is made from a customer’s financial 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 associated with the financial account

Alternate Flows:

[none]

Exceptions:

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

Related Use Case(s):

2.3.3. Record History: Account Creation

Relevant Requirement(s) from SRS:

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

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

Basic Flow:

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

(2) the user account’s 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 user account was created, into a text file associated with that user account

Alternate Flows:

[none]

Exceptions:

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

Related Use Case(s):

2.3.4. Record History: Account Blocking

Relevant Requirement(s) from SRS:

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

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

Basic Flow:

(1) a customer’s user 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 user account to mark it as blocked, meaning that no other users can access that user account from an ATM Module or a Teller Module until the user 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 associated with the user account

Alternate Flows:

[none]

Exceptions:

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

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

Related Use Case(s):

2.3.5. Record History: Account Unblocking

Relevant Requirement(s) from SRS:

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

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

Basic Flow:

(1) a customer’s user 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 user 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 associated with the user account

Alternate Flows:

[none]

Exceptions:

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

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

Related Use Case(s):

2.3.6. Monthly Savings Account Interest Added

Relevant Requirement(s) from SRS:

Precondition(s):

Postcondition(s):

Basic Flow:

(1)

Alternate Flows:

(1)

Exceptions:

(1)

Related Use Case(s):

2.3.7. Daily ATM Reset

Relevant Requirement(s) from SRS:

Precondition(s):

Postcondition(s):

Basic Flow:

(1)

Alternate Flows:

(1)

Exceptions:

(1)

Related Use Case(s):