Actors

- **Admin:** the coordinator of the system;
- **Customer:** A person logged in as an owner of a **BankAccount** in the system;

Glossary

- **Bank:** a Bank that has name, logo, e.t.c;
- **BankAccount:** A combination of the **Bank**, **account-number** and **account-name** of an individual;
- **Beneficiary:** for a particular **Customer,** it is an individual with a **BankAccount** that that **Customer** can transfer money to;
- **VisibleBeneficiary:** for a **Customer**, it is a **Beneficiary** of that **Customer** visible to that **Customer;**

  HiddenBeneficiary: a **Beneficiary** of a **Customer** that is not visible to that **Customer;**
- **CustomerKillSwitch:** any condition that when met, triggers the system to deactivate the **Customer's** account**;**
- **AAT:** Accumulative Amount Transferred; that total amount that has been transferred successfully by **Customer** since he/she began making transfers;
- **Transaction:** A detailed representation of money-transfer to or from the **Customer;**
- **TransactionPeer:** for a particular **Customer,** it is a **BankAccount** that can act as receipient when the sender is that **Customer**, or the sender when the receipient is that **Customer;**
- **TransactionRemarks:** A brief description, by the sender, of the purpose of the **Transaction;**
- **TransactionHistory:** the list of all **Transaction** a particular **Customer** is involved in;

Use Cases: Admin’s Perspective

A1: Admin logs in;

A2: Admin registers a new **Customer;**

A3: Admin registers a new **TransactionPeer** for a **Customer;**

A4: Admin adds a new **Transaction** to the **TransactionHistory** of a **Customer**;

A5: Admin registers a new **Beneficiary** for a **Customer;**

A6: Admin deactivates a **Customer;**

A7: Admin reactivates a deactivated **Customer;**

A8: Admin logs out;

Use Cases Details: Admin’s Perspective

A2: Admin registers a new Customer

Pre-conditions

- **Admin** is logged in as an **Admin.**

Steps

1. **Admin** selects **initializer**: this takes admin to the registeration form;
2. **Admin** fills registeration form:
   I. **Admin** inputs **Customer** biodata(first name, last name, gender and date of birth) ;
      - first name;
      - last name;
      - middle name;
      - gender; and
      - date of birth;

   II. **Admin** inputs **Customer** account-details:
      - Account number; and
      - Account type(savings or current);

   III. **Admin** inputs **Customer** login-details:
      - email address; and
      - password;

   IV. **Admin** inputs **CustomerKillSwitches:**
      - number of transfers;
      - Accumulative amount transferred(**AAT)**; and
      - Amount transferred at once; then
3. **Admin** submits the registeration form.

Post-conditions

- The **Customer** account has been created; and
- The **Customer’s** account balance is **$0.00**

Abnormal Paths

- If any part of the form is inappropriately filled, **Admin** is notified and prompted to try again.

Non-functional Requirements

- The **initializer** is a ‘Add new customer’ button.

A3: Admin registers a new TransactionPeer For a Customer

Pre-conditions

- **Admin** is logged in as **Admin;**
- The concerned **BankAccount** has not been registered as a **TransactionPeer.**

Steps

1. **Admin** selects the **initializer**: this triggers the presentation of a form to the **Admin;**
2. **Admin** fills the form:
    1. **Admin** inputs the bio-data of the **TransactionPeer**:
        1. First name;
        2. Last name; and
        3. Middle name;
    2. **Admin** inputs the bank-details of the **TransactionPeer:**
        1. **Bank** name; and
        2. **BankAccount** account-number;
    3. **Admin** chooses the **Customer** the **TransactionPeer** is to belong to;
3. **Admin** submits the form;

Post-conditions

- The **TransactionPeer** has been registerred;

Abnormal Paths

- If any part of the form is inappropriately filled, **Admin** is notified and prompted to try again.

Non-functional Requirements

- The **initializer** is a ‘Add new Transaction Peer’ button;
- The bank name is chosen from a list (available in the form) of porpular **USA** banks;
- The digits starting the account-number is auto-filled as soon as the bank-name is selected. This is to make sure the account-number rhymes with the bank, as **Customer** may be aware of discrepancies in that aspect;

A4: Admin adds a new Transaction to the TransactionHistory of a Customer

Pre-conditions

- **Admin** is logged in as **Admin;**
- **Admin** has selected the concerned **Customer;**

Steps

1. **Admin** selects the **initializer:** this opens a form to the **Admin;**
2. **Admin** fills the form:
   I. **Admin** chooses the transaction-type (credit or debit);
   II. **Admin** inputs the transaction amount;
   III. **Admin** inputs the **TransactionRemarks;**
   IV. **Admin** inputs **Transaction** meta-data:
      1. Transaction Date;
   V. **Admin** chooses the **TransactionPeer** from the list of all the **Customer's TransactionPeers;**
3. **Admin** submits the form;

Post-conditions

- A **TransactionRefNumber** has been automatically generated for the **Transaction;**
- **The Transaction** has been created;
- The **Transaction** has been added to the **TransactionHistory** of the concerned **Customer;**
- The **Customer's** new account balance is the difference between the sum of all credit **Transactions** and the sum of all debit **Transactions** in that **Cusomter's TransactionHistory;**
- If the **Transaction** is a debit and the **TransactionPeer** is not a **Beneficiary** of the concerned **Customer**, that **TransactionPeer** is added to the list of **VisibleBeneficiaries** of the **Customer;**

Abnormal Paths

- If the transaction-type is debit and the **Customer's** current account-balance is lesser than the transaction amount, the form has been inappropriately filled;
- If any part of the form is inappropriately filled, **Admin** is notified and prompted to try again.

Non-functional Requirements

- The **initializer** is a 'Record new transaction' button;

A5: Admin registers a new Beneficiary for a Customer

Pre-conditions

- **Admin** is logged in as **Admin;**
- **Admin** has registered the **Beneficiary's BankAccount** as a **TransactionPeer** of the concerned **Customer;**
- **Admin** has selected the concerned **Customer;**

Steps

1. **Admin** selects the **initializer**: this triggers the presentaion of the form to **Admin;**
2. **Admin** fills the form;
    1. **Admin** chooses the **Beneficiary's BankAccount** from the list of the **Customer's** available **TransactionPeers**; and
    2. **Admin** chooses the visibility of the **Beneficiary;**
3. **Admin** submits the form;

Post-conditions

- The concerened **Customer** can now make a transfer to the **BankAccount** of the **Beneficiary**

Abnormal Paths:

- If any part of the form is inappropriately filled, **Admin** is notified and prompted to try again.

Non-functional Requirements

- The **TransactionPeers** available to **Admin**, in the form, are the **TransactionPeers** of the concerned **Customer** yet to be **Beneficiaries** of that **Customer;**

Use Cases: Customer's Perspective

C1: Customer logs in;

C2: Customer views the account (includes **C3**, includes **C6**);

C3: Customer views **Customer's BankAccount** details**;**

C5: Customer transfers money to one of **Customer's Beneficiaries;**

C6: Customer views **Customer's TransactionHistory;**

C7: Customer selects a particular **Transaction** from **Customer's TransactionHistory** (extends **C6**);

C8: Customer logs out;

Use Cases Details: Customer's Perspective

C5: Customer Transfers Money to One of Customer's Beneficiaries

Pre-conditions

- **Customer** is logged in as **Customer;**

Steps

1. **Customer** selects **initializer:** this triggers the presentation of a form to **Customer;**
2. **Customer** fills form:
    1. **Customer** inputs **Beneficiary** details: this may be done in two different mutually exclusive ways:
        I. **Customer** chooses **Beneficiary** from the list of **Customer's VisibleBeneficiaries**; or
        II. **Customer** inputs **Beneficiary's BankAccount** details, manually:
            1. **Customer** inputs **BankAccount** account-number;
            2. **Customer** chooses the **BankAccount Bank** from a list of porpular **USA Banks**;
            3. The system autmatically fetches and inputs the account-name that corresponds to the combination of inputted account-number and **Bank**;
    2. **Customer** inputs **TransferDetails:**
        I. transfer-amount: the amount of money to be transferred to the **Beneficiary;** and
        II. **TransactionRemarks**
    3. **Customer** inputs password;
3. **Customer** submits the form;

Post-conditions

- The money-transfer has been registerred as a debit **Transaction** of **Customer;**
- The **Transaction** has been added to the **TransactionHistory** of **Customer**;
- **Customer's** new account balance is the difference between the sum of all credit **Transactions** and the sum of all debit **Transactions** in **Cusomter's TransactionHistory;**

Abnormal Paths

- While filling the **Beneficiay** details, if the combination of the inputted account-number and inputted **Bank** does not match the **BankAccount** of any **Beneficiary** of **Customer,** the system informs **Customer** that inputted **Beneficiary** details cannot be trusted, and prompts **Customer** to re-input the **BankAccount** details;
- If any part of the form is inappropriately filled, **Customer** is notified and prompted to try again.

Non-functional Requirements

- The **initializer** is a ‘Make a Transfer’ Link in the NavigationBar of the system;