Overview

This document represents an overview of the design for the bank application. The application has multiple components and composition of components which make it up. This application stores accounts of customers in a binary tree. Each account has eight funds. Together, they are stored as a vector of funds. Each fund stores its own transaction history, amount of available funds in the account, and type of fund specified as string. The account also has a customer which is a separate component that contains the first and last name of the customer as a string. The process starts with reading a series of transactions from a file, followed by and storing each transaction to a queue as a string. Once all of the transactions are read and stored, it parses and executes each transaction from the queue until there are no more transactions in the queue. There are six types of transactions, and each transaction will have a specific flow.

1. Open: Initially checks if the account exists in a binary tree. If it does not exist, it dynamically creates an account and inserts to binary tree.
2. Deposit: Check if the account exists. If it does not exist, it displays an error message. Otherwise, it finds the appropriate fund within an account and adjusts the amount. Also, it ensures the correct amount is entered.
3. Withdraw: Check if the account exists. If it does not exist, it displays an error message. Otherwise, find the appropriate fund and check if the fund has sufficient funds to cover the withdrawal. If not, it will check if the entire account has enough funds. If there are sufficient funds, it will adjust all of the funds effected in the withdrawal. If there are not sufficient funds, then it will display an error message.
4. Transfer: Initially checks if the account exists in a binary tree. If does not exist, it displays an error message. Otherwise, if the transfer amount is correct (less or equal to account balance), it transfers and makes necessary adjustments to each fund effected. Otherwise, it shows an error.
5. Display history of all transactions: Initially checks if the account exists in the binary tree. If does not exist, it displays an error message. Otherwise, it will iterate through the vector, and it will display the history of each fund.
6. Display history of find transactions: Initially checks if the account exists in the binary tree. If does not exist, it displays an error message. Otherwise, it will find the given fund and it will display its history.

For more detail refer to below figures. Figure 1 shows flow of the program, Figure 2 is class diagram and Figure 3 is visuals of key data structures.

Class Interaction

There are five classes: Bank, BSTree, Account, Customer, and Fund. The Bank class is the main driver that reads requests from the file, parses each transaction, and sends the request to BSTree. Requests can be to insert a new account to BSTree or retrieve an account to adjust funds (deposit, transfer, withdraw, or see fund or account history). If insert request is issued, Bank class dynamically creates an Account which consists of Customer and Fund classes and passes to BSTree class to allocated inside the BSTree container. If it fails insertion, Bank class bear the responsibility to clean up necessary memory allocation. If retrieve request is issued, Bank class retrieves Account and requests Account class to perform transactions like deposit, withdraw, transfer or display fund or account history.

Figure 1: Program Flow

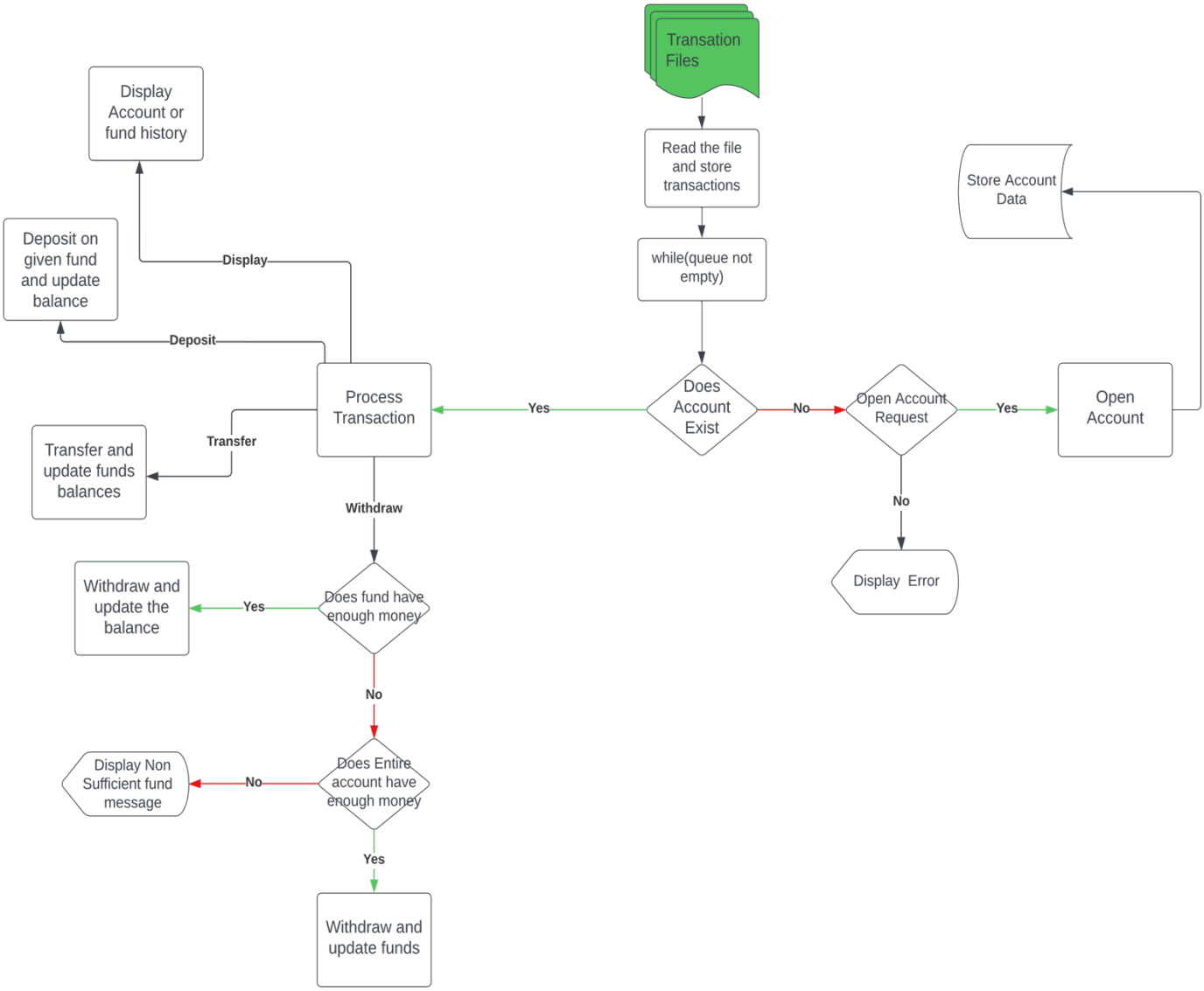

Figure 2: Class Diagram