February 22, 2016

Achal Parikh, stewart gibson, andrew lau

team 4

design document

front-end prototype

Table of Contents

[Introduction 1](#_Toc443937418)

[Overall structure 2](#_Toc443937419)

[UML class diagram: 2](#_Toc443937420)

[Description: 3](#_Toc443937421)

[Class and Method intentions: 4](#_Toc443937422)

[Input and output files 7](#_Toc443937423)

[Current bank account file: 7](#_Toc443937424)

[Sample Daily Transaction File: 8](#_Toc443937425)

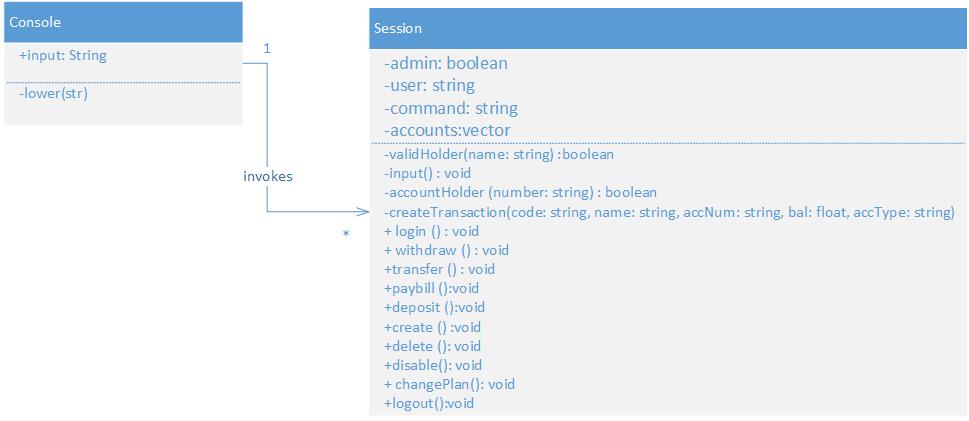
# 

# Introduction

This document is designed to be a reference for any individual who wishes to understand or implement the structure of the front end prototype of the bank transaction system created by team 4. This document describes the overall structure of the system and provides a structural class diagram to show the relationships between the classes. Additionally, this document contains a description of each method and class and the specific purpose they serve in within the system. The document also includes the input file of existing accounts in the system and example of an output file consisting of the daily transactions.

# Overall structure

## UML class diagram:



## Description:

The front end of the system consists of two classes.

First class is called console whose attributes include a string and is used to prompt the user of initial information. The class is then used to call the session class that handles all the features of the system. The console class also consists of one method (lower()) that turns all input characters into lower case characters before they are passed to the session class

The second class is called session class which handles all features of the system during an active session. The features include login, withdraw, transfer, pay bill, deposit, create, delete, disable, change plan, and logout. The class is also takes input from the input file called the current accounts file and stores the accounts in a vector called accounts to keep track of the accounts and to check for account validity in addition to checking balances to perform the transactions and generate a daily transaction file used by the back-end of the system.

Each method in the session class contains its own local variables to perform the transactions.

The console class is also associated with the session class with a multiplicity of 1 to many meaning that a single console can invoke multiple sessions during one day.

# Class and Method intentions:

|  |  |  |
| --- | --- | --- |
| Type | Name | Description |
| **Class** | Console | This class reads in a string until a valid login type is entered at which point it invokes the session class. |
| Method | Lower(input) | The lower method takes in the input from the console class and converts it into lower case letters |

|  |  |  |
| --- | --- | --- |
| Type | Name | Description |
| **Class** | Session | Creates a new session after every login and deletes a session after a logout operation.  This class is also responsible for supplying the user with all features of the system.  Additionally, this class is responsible for creating daily transaction file which will be used by the back end. |
| Method | validHolder(name) | This method is responsible for verifying that a name entered into the system is valid in the current accounts file. |
| Method | Input() | This method reads in a command to perform transaction |
| Method | accountholder(number) | This method is responsible for verifying that an account number entered into the system is valid in the current accounts file |
| Method | createTransaction (code, name, accNum, bal, accType) | This method is responsible for generating output for the bank account transaction file.  It creates the output by taking in the transaction code, account holder name, account number, account balance and account type. |
| Method | Login() | This method is used to login into the appropriate account type (standard or admin) and information after appropriate for the type is prompted |
| Method | Withdraw() | This method is responsible for withdrawing a certain amount less than or equal to 500.00 from an account and also for application of appropriate fees according to the plan if the transaction occurs in standard mode |
| Method | Transfer () | This method is responsible for transferring amount from one account to another and for application of fees if it is used in standard mode.  Also this method generates an error if the money being transferred will exceed the maximum balance in the account it is being transferred too. |
| Method | Paybill() | This method is responsible for withdrawing a certain amount less than or equal to 2000.00 from an account and sending it to an appropriate company and also apply appropriate transaction fees according to the plan if the transaction occurs in standard mode |
| Method | Deposit() | This method is responsible for depositing amount in a certain account valid for the user during an active session |
| Method | Create() | This method is an admin exclusive feature to create new accounts in the system |
| Method | Delete() | This method is an admin exclusive feature to delete an account in the system |
| Method | Disable() | This method is an admin exclusive feature to disable an account. This means that the account will exist in the system but no transactions can occur from that account |
| Method | Changeplan() | This is an admin exclusive feature to change the plan of the account from a student to a non-student or from a non-student to a student |
| Method | Logout() | This method is responsible for ending the active session |

# Input and output files

## Current bank account file:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Account number | Name | Active/disabled | Balance | Plan |
| 00001 | Achal Parikh | A | 01001.01 | Student |
| 00002 | Andrew Law | A | 20000.00 | Student |
| 00003 | Jack | A | 30000.00 | Non-student |
| 00004 | Bruce Banner | A | 40000.00 | Non-student |
| 00005 | Hawk eye | A | 50000.00 | Non-student |
| 00006 | Black Widow | A | 00001.00 | Student |
| 00007 | Steve Rogers | D | 05000.00 | Non-student |
| 00008 | Tony Stark | A | 99999.99 | Non-student |
| 00009 | Thor Odinson | A | 00500.00 | Non-student |
| 00010 | Student | A | 00500.00 | Student |
| 00011 | John Doe | A | 00500.05 | Non-student |
| 00012 | John Doe | D | 00500.05 | Non-student |
| 00000 | End of file | A | 00000.00 | Non-student |

## Sample Daily Transaction File:

Each transaction is made up of three rows where:

The first row indicates a successful login.

The second row consists of data regarding the transaction.

The third row indicates a successful logout.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 10 | | | | |
| 01 | BLACK WIDOW | 00006 | 00025.00 | S |
| 00 | | | | |
| 10 | | | | |
| 03 | ACHAL PARIKH | 00001 | 020000.00 | A |
| 00 | | | | |
| 10 | | | | |
| 02 | STUDENT | 000011 | 00500.00 | A |
| 00 | | | | |