**ACCOUNT MANAGEMENT SYSTEM**

NAME: KUMAR SIDDHANT (ks1306)  
DATE : 12/11/2017

PROJECT URL : <https://github.com/Sid231/FinalProject>

The account management system is a system which imitates how the stock and bank account system work collectively in real world. In this system the user is allowed to view, buy and sell stocks and side by side view his account details, bank balance and the transaction history of his activity. The user can also wish to deposit and withdraw cash from the account management system and can check the remaining amount in his account and can also check the transaction history of his activities.

Below is a complete explanation of how this system works.

**Account\_Siddhant.h**

In this file, we create the base class called **Account** which is responsible to hold the data related to the account of the user.

Variables defined in the class:  
**cashBalance** : This variable holds the current balance in the user’s bank account

Methods defined in the class:  
**Account()** : This is the constructor of the account class.  
**~Account()** : This is the destructor of the account class.  
**getCashBalance()** : This method returns the current balance of the user’s account.  
**setCashBalance()** : This method sets the cash balance from a source.

**Account\_Siddhant.cpp**

In this file, we implement the methods that are defined in the **Account\_Siddhant.h**.

Methods implementations:  
**Account ::Account()** : Set the balance.  
**Account ::~Account()** : Not used as it is not required considering the motives of the project.

**accountNode\_Siddhant.h**

In this file, we create a class named **accountNode** whose objects form the linkedlist nodes.

Variables defined in the class:  
**company** : This variable holds the name of the company of the stock purchased/sold.  
**numberOfShares** : This variable holds the number of stocks.  
**amountPerShareForSorting**: This variable holds the price per stock for sorting purpose.  
**amountPerShare**: This variable holds the price per stock.  
**currentPortfolioNodeVal**: This variable holds the gross value of the stocks.  
**date** : This variable holds the date.  
**\*prev** : This pointer points to the previous node of the current object node.  
**\*next** : This pointer points to the next node of the current object node.

Methods defined in the class:  
**accountNode()** : This is the constructor of the class.

**accountNode\_Siddhant.cpp**

In this file, we implement the methods that are defined in the **accountNode\_Siddhant.h**.

Methods implementations:  
**Account ::accountNode()** :Initialize the **prev** and **next** pointers as NULL.

**BankAccount\_Siddhant.h**

In this file we create a class called **BankAccount** which inherits the **Account** class. This class is responsible for handling all the account operations which are related to the banking system like cash deposit, cash withdrawal and in the end give the history of all the transactions.

Variables defined in the class:  
**depositAmount**: This variable holds the amount deposited to the account  
**withdrawalAmount**: This variable holds the amount withdrawn from the account

Method Implementations:  
**BankAccount()** : This is the constructor of the class  
**~BankAccount()** : This is the destructor of the class.  
**setBalance()** : This method sets the cash balance by getting the value from its source  
**getBalance()** : This method returns the cash balance at the places where it is needed  
**viewBalance():** This method displays the current balance of the account.  
**depositCashAmount()**: This method runs the algorithm after some amount is deposited into the account.  
**withdrawCashAmount()**: This method runs the algorithm after some amount has been withdrawn from the account.  
**printHistory()**: This method prints the history of all the banking transactions.

**BankAccount\_Siddhant.cpp**

In this file we implement the methods defined in the **BankAccount\_Siddhant.h**. Below is a brief description of what is done in each method implementations of this file.

Method Implementations:  
**BankAccount::BankAccount()** : This method is the constructor of the class and is used to initialize the **cashBalance**, **depositAmount** and the **withdrawalAmount** variables.  
**BankAccount::~BankAccount() :** This method is not used as it is not required considering the motives of the project.  
**BankAccount::setBalance():** This method gets the latest value from the file **cashBalance.txt** and sets the variable **cashBalance** of the parent class **Account**.  
**BankAccount::getBalance()**: This method gets the cash balance value stored in the variable **cashBalance** of the parent class **Account**.  
**BankAccount::depositCashAmount()**: This method runs the algorithm to update and store the data in the account system after a new amount enters into the account system. The timestamp of the transaction is recorded, cash balance is updated in the cashBalance.txt file and the transaction is recorded in the bank\_transaction\_history.txt file.  
**BankAccount::withdrawCashAmount()**: This method runs the algorithm to update and store the data in the account system after a new amount is withdrawn from the account system. The timestamp of the transaction is recorded, cash balance is updated in the cashBalance.txt file and the transaction is recorded in the bank\_transaction\_history.txt file.  
**BankAccount::printHistory()**: This method prints the recorded transaction in the bank\_transaction\_history.txt file in the console.

**StockAccount\_Siddhant.h**

In this file we create a class called **StockAccount** which inherits the **Account** class. This class is responsible for handling all the operations related to stocks and shares like buying shares, selling shares, viewing shares, etc.

Variables defined in the class:  
**\*previousPointer** : This pointer is used for LinkedList traversal operations  
**\*tailPointer**: This pointer points to the last node of the LinkedList  
**\*headPointer**: This pointer points to the first node of the LinkedList.  
**company**: This variable holds the company name of the stock.  
**amount**: This variable holds the price of the stock per share  
**date**: this variable holds the date of the transaction  
**portValue**: This variable holds the total amount of the portfolio  
**portValue\_array**: This variable holds the portValue recorded after each transaction.  
**timestamp\_array**: This variable holds the time recorded after each transaction.  
**sizeOfPortValueArray**: This variable holds the size of the portValue\_array.  
**sizeOfList**: This variable holds the size of the linkedlist.  
**stockDataMap<string,double>:** This variable holds the stock data in the format key = **company** and value = **amount  
portfolioDataMap<string,int>:** This variable holds the portfolio data in the format key = **company** and value = number of stocks

StockAccount\_Siddhant.cpp