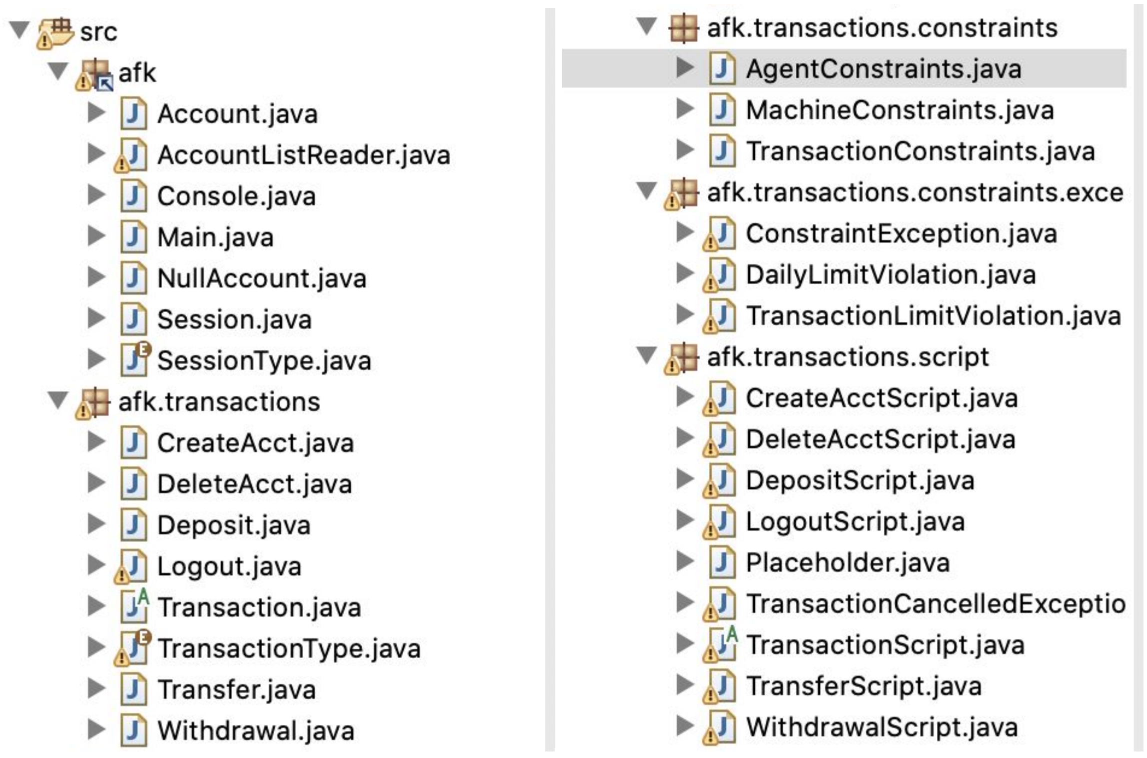
Design Document

Group 17
Ziping Li, Shihao Lu, Ryan Kerr

All the java files of the prototype are under the src folder. The files in afk package contain the general codes for the application such as opening the console, creating sessions and file I/O. There is a afk.transaction package that specifically handles all the transactions including deposit, withdraw, transfer, create and delete accounts. The scripts, constraints and exceptions of the transactions are put under different packages.



```
▼ src
  ▼ afk
    ▶ Account.java
    ▶ AccountListReader.java
    ▶ Console.java
    ▶ Main.java
    ▶ NullAccount.java
    ▶ Session.java
    ▶ SessionType.java
  ▼ afk.transactions
    ▶ CreateAcct.java
    ▶ DeleteAcct.java
    ▶ Deposit.java
    ▶ Logout.java
    ▶ Transaction.java
    ▶ TransactionType.java
    ▶ Transfer.java
    ▶ Withdrawal.java
```

```
▼ afk.transactions.constraints
  ▶ AgentConstraints.java
  ▶ MachineConstraints.java
  ▶ TransactionConstraints.java
▼ afk.transactions.constraints.exce
  ▶ ConstraintException.java
  ▶ DailyLimitViolation.java
  ▶ TransactionLimitViolation.java
▼ afk.transactions.script
  ▶ CreateAcctScript.java
  ▶ DeleteAcctScript.java
  ▶ DepositScript.java
  ▶ LogoutScript.java
  ▶ Placeholder.java
  ▶ TransactionCancelledExceptio
  ▶ TransactionScript.java
  ▶ TransferScript.java
  ▶ WithdrawalScript.java
```

General

Main Class

Main is the entry point into the application, where the application starts and exits.

Method

main	Initializes the console and runs the application in a loop
Constructor	Prompts the user to login, reads the accounts list file, starts the session, and then writes the transaction summary once the session is complete. Handle exit command from the user
writeFile	Append the summary string of transactions into the transaction summary file

Console Class

Serves as a reusable wrapper around System.in to read input from the user.

Method

readString	Reads a single line of input, returns the empty string on failure
readAccount	Reads a 7-digit number, throws exceptions if malformed
readAmount	Reads a positive integer that may have formatting (allow commas and spaces)

Account Data Class

Represents an account. Has a number and a record of monetary transactions to ensure daily transaction limits aren't violated.

Method

Constructor	Ensures the given account number is valid
getNumber toString	Returns the account number as a string
addTransaction	Adds the monetary amount to the account's own transaction log
getTransactionAmount	Returns the amount logged for a given transaction type
compareTo equal	Allows for comparing of Accounts

Children

NullAccount	Represents the special account #0000000 to be used as a placeholder in unused transaction fields

Session Class

The Session class provides the interface for users to make transactions after they have logged in. A session is provided with a SessionType and a valid accounts list.

Method

run	The session prompts the user for a command and performs a lookup of the command to get the appropriate TransactionType. If the transaction type is allowed by the session's type, the session executes the TransactionScript and stores the result in the transaction log. This repeats until the END_OF_SESSION transaction is executed when the user types "logout".

SessionType Enum

Defines the available session types, their names, and links their **TransactionConstraint**s.

Method

getName	Returns the name of the session type
getConstraints	Returns the **TransactionConstraints** imposed by this session type

Transaction

Transaction Class

Represents a transaction between two accounts.

Method

Constructor	Creates a new transaction with the fields **TransactionType**, source account, destination account, amount, and optional account name
getAmount	Returns the amount stored in transaction
getSourceAccount getDestinationAccount	Returns the relevant account objects stored in transaction
getType	Returns the enumerated **TransactionType**
toString	Returns the transaction in the format specified for the transaction summary file

Children

Logout	End the session and return transaction record with EOS message
Deposit	Deal with limitation of amount, and the messages for deposit
Withdrawal	Deal with limitation of amount, and the messages for withdraw
Transfer	Deal with limitation of amount, and the messages for transfer
CreateAcct	Add NEW message into the list of transaction
DeleteAcct	Add DEL message into the list of transaction

TransactionType Enum

The TransactionType enum serves as the /usr/bin of this program. It defines all the available transaction types available, each with their own summary code, invocation command, and builder script.

Method

stringToEnum	Performs a lookup and returns the enum corresponding to a given command. Returns null if there is no such command.
getShortCode	Returns the 3 character transaction code
getCommand	Returns the command used to call the transaction script
getScript	Returns a **TransactionScript** corresponding to this transaction type

TransactionScript<TransactionType> Abstract Class

Transaction scripts are responsible for interacting with the user in order to get all the required user inputs which make up a given Transaction.

Method

getAccount	Prompts the user for an account number. Does not return until the user provides a valid account number which corresponds to an existing account. The user can cancel this prompt by typing in “0000000” (The null account)
getAmount	Prompts the user for a transaction amount. Does not return until the user provides a valid amount. The user can cancel this prompt by giving the value of 0 cents.
execute	Function implemented by specific transaction scripts. These scripts define the dialogue that the user will interact with and provide input. The execute function will return a valid transaction at the end of the dialog. If the user cancels the dialog, the execution function will throw a TransactionCancelled exception for the session to handle.

Children

LogoutScript	Does nothing
DepositScript	Get the account and amount, create deposit transaction, handle limitation messages
WithdrawalScript	Get the account and amount, create withdraw transaction, handle limitation messages
TransferScript	Get the account and amount, create transfer transaction, handle limitation messages

CreateAcctScript	Deal with the input from user to return the new account number and account name, and also check if the account name and account number is valid.
DeleteAcctScript	Deal with the input from to check the provided account number and return the delete account method.

TransactionScript<TransactionType> Abstract Class

Transaction scripts are responsible for interacting with the user in order to get all the required user inputs which make up a given Transaction.

Method

getAccount	Prompts the user for an account number. Does not return until the user provides a valid account number which corresponds to an existing account. The user can cancel this prompt by typing in “0000000” (The null account)
getAmount	Prompts the user for a transaction amount. Does not return until the user provides a valid amount. The user can cancel this prompt by giving the value of 0 cents.
execute	Function implemented by specific transaction scripts. These scripts define the dialogue that the user will interact with and provide input. The execute function will return a valid transaction at the end of the dialog. If the user cancels the dialog, the execution function will throw a TransactionCancelled exception for the session to handle.

TransactionConstraints Class

Transaction constraints define the per-transaction, daily limits, and allowed transaction types.

Method

(Protected) addAllowedTransaction Type	This function is used by child classes to specify which transactions are permitted
(Protected) setPerTransactionLimit	Sets the per-transaction limit for a given transaction type

(Protected) setDailyLimit	Sets the daily limit for a given transaction type
isAllowedTransaction	Checks whether a given transaction type is permitted by this constraint
getPerTransactionLimit	Returns the per-transaction limit for a given transaction type. Returns 0 if there are no limits for the type.
getDailyLimit	Returns the daily limit for a given transaction type. Returns 0 if there are no limits for the type.
getAllowedTransactions	Returns all the transactions that are permitted by this constraint

Children

AgentConstraints	Allows all the transaction types and sets no daily limits on accounts.
MachineConstraints	Only allows Logout, Withdrawal, Deposit, and Transfer. Sets all limits accordingly