**A.T.M Application**

**Group 9**

**Design Document**

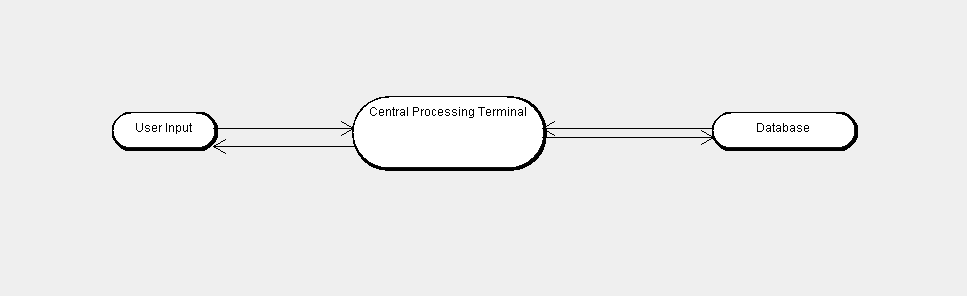
**1. Introduction**

The purpose of this document is to establish a base architecture and design for the A.T.M. OS application. This document contains the base flow and design of the application and will allow those looking at the requirements to have a base understanding of how the application is supposed to function and the general setup of it. This will also help the developers have something to reference back to to ensure that the application lines up with the design described in this document.

**2. Architecture**

**2.1 Introduction**

The high level architecture of the application will be a repository model. This was chosen because the entirety of the processing occurs in a main terminal, and the rest simply contains or sends data to be processed. The first portion contains the user interface, which will be achieved through Java GUI libraries to generate a display for the users to interact with. The second is the main processing terminal, consisting of several Java classes, that takes all the input from the user and allows them to access the functions of the Java classes. Then we will have a data transport layer which implements the Java Database Connection, JDBC, to ensure a secure connection the the database. Lastly, the database, coded in MySQL, contains the bank information, funds, exchange rates, and other information that the processing terminal uses.



**2.2.1**

The user input of the application consists of the display and methods to input information, such as the bank card information and pin number. This is the only point that the user interacts with directly, after the user information has been entered and a positive identification has been achieved the user can access the functions of the system without accessing the information directly..

**2.2.2 Central Processing Terminal**

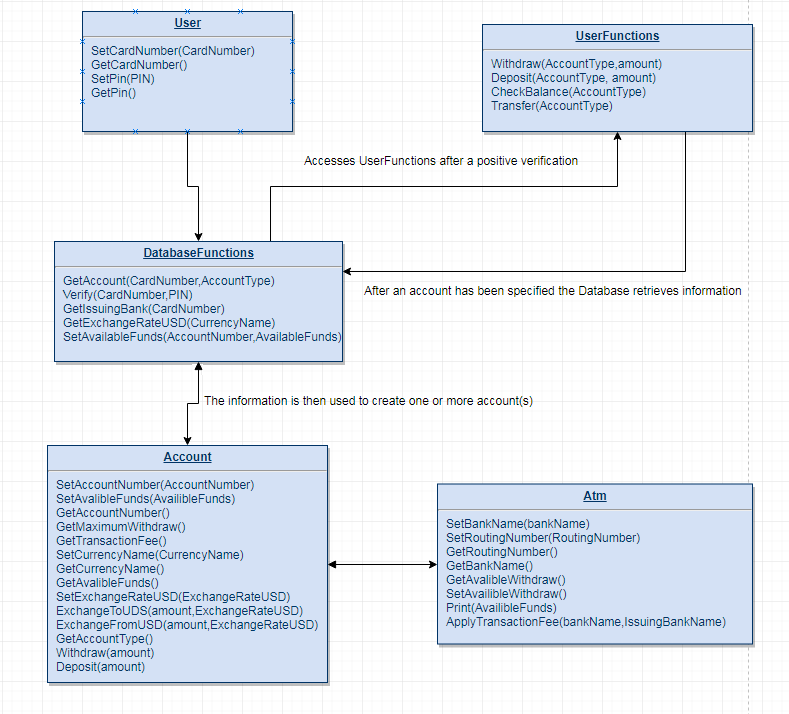
All of the application processing happens in the terminal. This uses several Java classes when processing the user input, and a JDBC object when interacting with the database. The terminal executes all the commands and subsequent data gathering needed to complete these actions.

**2.2.3 Database**

The information that is used to verify credentials and store funds is stored in the database. The processing terminal can also change the values in the database based on the user requests, such as subtracting the funds value based on how much is withdrawn. The database also holds currency exchange rates to allow for currencies to be exchanged.

**3.1 Class Representation**

**3.1.1 Class Diagram**

****

**3.2 Class Components**

**3.2.1 User**

This user consists of a 16 digit bank card number and personal identification number, PIN, that is entered through the controlling terminal to verify its presence in the bank account database and if it is not, the user receives an error on the user display

**3.2.2 Database Functions**The database functions are used to first verify the user and if the bank card number and PIN are held within the database then the user is directed to the user functions. The GetAccount() function of the class are to retrieve the account information and construct one or more account classes. The next function that will be used would be the SetAvailableFunds() this would be used to update the available balance of a given account. If the funds in the account are requested to be changed to a different currency, then GetExchangeRateUSD() is called.

**3.2.3 Account**

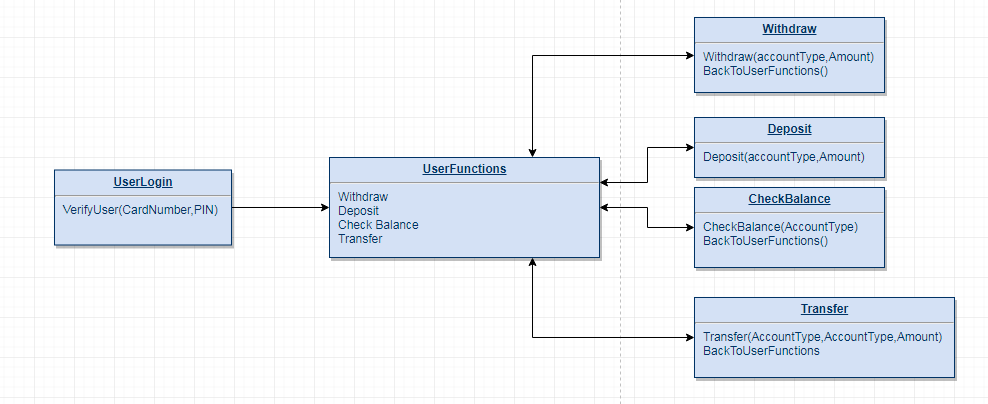
The account class controls all aspects of the account once it has been created, the functions that are important to the user are the withdraw and deposit functions. These will be used for the withdraw, deposit, and transfer user functions. Withdraw will remove funds from the account, deposit will add funds to the account, and transfer will withdraw from one account and add to another account. At the end of the withdraw and deposit functions will be a function call to update the available funds in each account. The other function that will be available to check the balance is the CheckAvailableFunds() function. The rest of the uses are for the ATM functions.

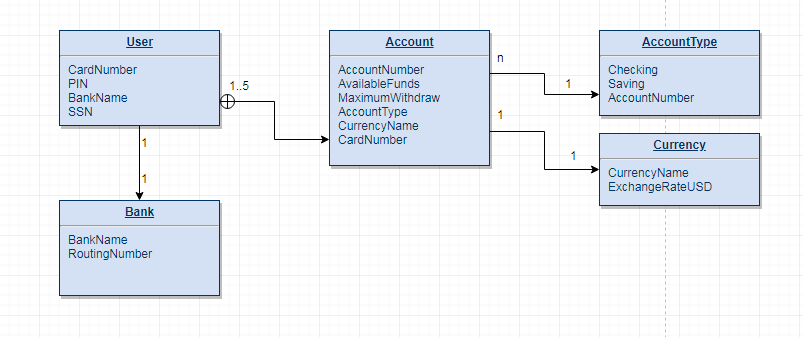
**3.2.4 ATM Functions**

This focuses on the state of the ATM by keeping track of how much money is contained within the machine at the time of the transaction. In addition to this it holds the information of the bank that owns the machine and will apply a transaction fee if the machine’s bank differs from that of the user. In the startup event then the first thing that appears will be that of the ATM functions so that the host bank and available funds can be entered.

**3.2.5 Java ATM Controlling Terminal**

The Controlling terminal consists of several GUIs, which are coded in Java, that changes based on which action the user decides and will allow the user to the functions of the class that shares the same name. This allows the user to be able to access their accounts while limiting the access to the user.



**3.2.6 Bank Database**

Coded in MySQL, this database contains the account information, such as the card number and associated pin, the bank information, the amount of funds in the account, and the type of currency associated with the funds. This allows for a maximum of five cards being linked to the same joint account. This model has the user’s social security number and the bank’s routing number as primary keys. The foreign keys in this design are the card number, bank name, currency name, and account number.