

Team 29 Bank Project Design Document

Team Members: Shilpen Patel, Justin DiEmmanuele, George Padavick, and Matthew Gilgo

Table of Contents:

1. Introduction
2. Requirements Specification
3. Database Design
4. User Interface Design
5. Object Design

Introduction

The purpose of this assignment was to design, test, and implement an application for a client who wanted to generate income from a virtual Bank. The client referred to as the Bank Manager in the project specification outlined a set of services they wished to offer to customers. These services would be catered to different subsets of customers and were fairly vague in description therefore, certain assumptions were made to ultimately deliver a product for the client. Based on the customers input initially a set of detailed requirements were defined to work towards incorporating into the product. Once requirements were identified and understood the task of designing the product to satisfy the requirements was completed in three parts: Database design, User Interface Design, and Object Oriented Design. The process has been outlined in further detail in the following sections.

Requirements Specification

Before going through the process of design an appropriate Database, User Interface, and Object Design a detailed understanding of the customer requirements was generated. The requirements were generated from the provided assignment specification and through conference call exchanges with the customer. Based on the customers (who will be referred to as the Bank Manager) input the following requirements were generated for the program:

- Generic Requirements:
 - The Bank Manager does not want to interface with customers in person therefore all functionalities of the bank must be implemented virtually
 - The program shall implement a fully digital Bank with certain functionalities detailed below
 - The goal of the bank is to make money by charging fees to customers who are using the banks services
 - Customers and Managers should be able to easily navigate the functionalities of the Bank through an easy to use Graphical User Interface (GUI)
 - The user of the application should be create a user account with a username and password
 - There will be two types of users interfacing with the bank: a Customer and Manager that will each have secure login credentials.
- Customer Requirements:
 - Must be able to create/delete checking, savings, trading, and loan accounts
 - Must be able to manipulate each type of account
 - Checking accounts can be deposited into and withdrawn from
- Manager Requirements:
 - Must be able see who the customers are
 - Must be able to see what transactions the customers have made
 - Must be able to maintain and manipulate the stock market
- Account Requirements:

- Checking account should operate as a generic cash account where a customer is able to withdraw and deposit (transfer) money between each account
- Savings accounts can be deposited into and withdrawn from and high balance accounts accumulate interest
- Trading accounts can be used to buy and sell stock, see current stock portfolio, and see unrealized gains and losses
- Loan accounts must be charged interest
- Stock Market Requirements:
 - The Bank Manager would like the bank to offer brokerage services to customers who have accounts that satisfy certain metrics. Specifically, the customers who have more than \$5000 in their account can open a brokerage account.
 - Manager accounts should be able to update the stock market stocks list and prices for customers
 - Customers with brokerage accounts should be able to trade stocks in the stock market and see realized and unrealized gains
- Persistence Requirement
 - if bank service goes down, their information/money should not go away

From the above requirements outlined above there were three main components needed to be designed and outlined before any implementation could begin. The first component needed was the back-end database design to satisfy the persistence requirements the Bank Manager has requested. The second outline needed was the design and wireframing of the Graphical User Interface based that outlines the user experience with the Bank services. The final and main component of this application was the object-oriented design that implemented all the logic to connect the front-end GUI with the back-end Database.

Database Design

After digesting the requirements an initial design of the back-end database needed to capture the appropriate information for this application. There were five schemas (tables) outlined in the database design that was needed for the application. The tables are as follows:

users		accounts		transactions		stocks		stocks_owned	
PK	user_id	PK	account_id	PK	transaction_id	PK	stock_id	PK	stock_instance_owned_id
	user_type		user_id		transaction_type		stock_ticker		account_id
	username		account_type		transaction_amount		stock_price		stock_id
	password		balance		timestamp				cash_balance
			currency_name		account_id				stock_buy_price
			currency_symbol						num_shares

The general idea behind this database design was to have a users table that captures each unique user in the bank with the type of user (Customer or Manager) along with the security credentials. The users once logged in can access the accounts schema through their unique user_id and access their accounts if they

are a customer or access all the users if the user is a manager. The transactions table was needed to capture all the transactions happening in the bank and create a paper-trail of all the transactions in the bank. The stocks and stocks_owned tables are used to store information needed about the stock market, since the stock market is not technically part of the bank they were separated from the other bank tables. Using this structure proved to be extremely useful in implementing the actual application as queries proved to be powerful tools in getting information from the database and coercing it into a form that is easily digestible.

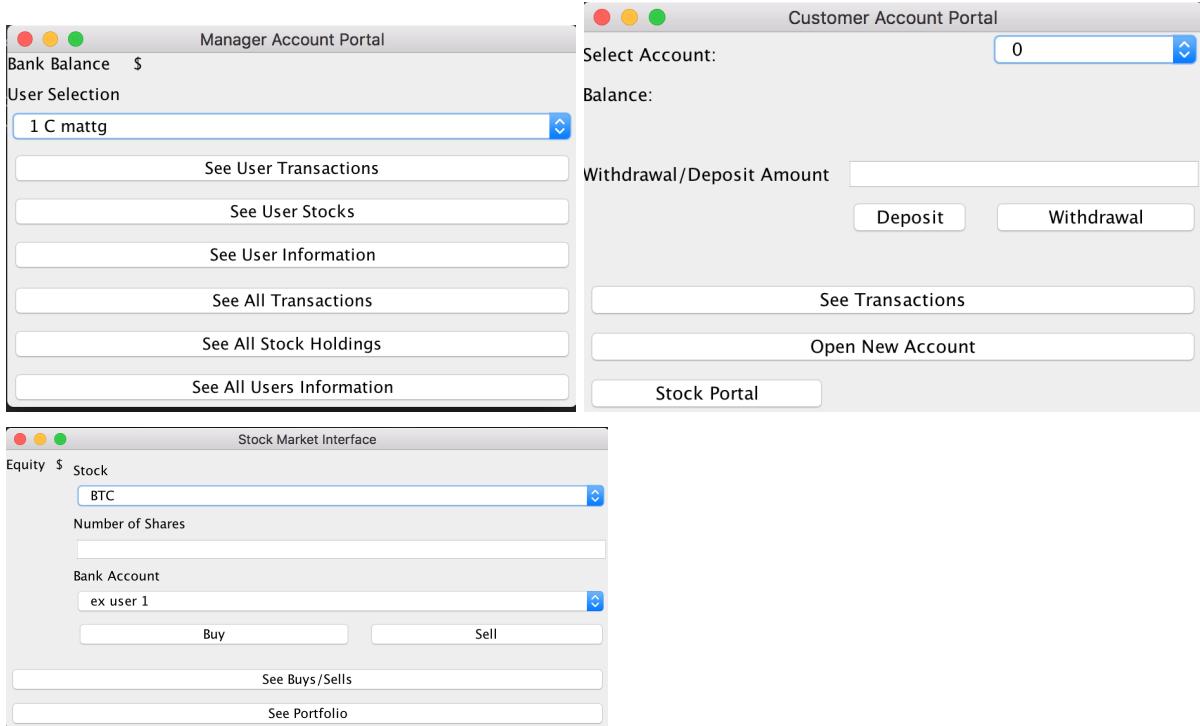
GUI Design

One of the key requirements from the Bank Manager was to allow the user to easily interface with the Bank so that the services offered by the bank could be easily accessed and the information for each type of user could be easily seen. Therefore, before beginning the actual implementation of the application and GUI's that would be needed powerpoint was used to generate initial renderings of screens the user would see and map out the flow of user experience. Specifically, the requirements for each user types were translated into a flow chart of renderings that outlined what the ultimate GUI would look like and feel like to make the ease of use requirement was satisfied appropriately. Initial renderings are shown below:

The wireframe consists of three rectangular panels. The left panel contains fields for 'Username' and 'Password', and buttons for 'Login' and 'Create Account'. The middle panel displays a summary of stocks owned: 'Stocks owned: Stocks Listed here and below. Show Gains (unrealized and realized)'. It includes 'Buy Stock' and 'Sell Stock' buttons. The right panel lists account types with their respective balance boxes: 'Checking: \$\$\$\$ Listed here', 'Savings: \$\$\$\$ Listed here', and 'Any other accounts listed similarly...'. A 'Enter Stock Account' button is also present.

These rough wireframes were then implemented and flushed out upon complete implementation of the Bank application. The actual application GUI screens are shown below:

The screenshot shows the 'Account Creation Portal' window. On the left, there are fields for 'Username' and 'Password', and buttons for 'Create Manager' and 'Create User'. On the right, there are dropdown menus for 'Account Currency' (set to 'USD'), 'Account Type' (set to 'C'), and 'Deposit Amount'. Below these are fields for 'Account Fee' (\$1.0) and 'Account to charge' (set to '0'). At the bottom right is a large 'Create Account' button.



Manager View of all Accounts:

user_id	user_type	username	password	account_id	account_type	balance	currency_name
1	C	mattg	almostdone	1	CK	1000.0	
1	C	mattg	almostdone	2	Checking	1500.0	USD
1	C	mattg	almostdone	3	Saving	500.0	USD
1	C	mattg	almostdone	4	Loan	100000.0	USD
1	C	mattg	almostdone	5	Loan	100000.0	USD
3	C	shilpentestuser	shilpentestuser	6	Saving	15.0	BTC
3	C	shilpentestuser	shilpentestuser	7	Loan	5.0	ETH

View of all Transactions:

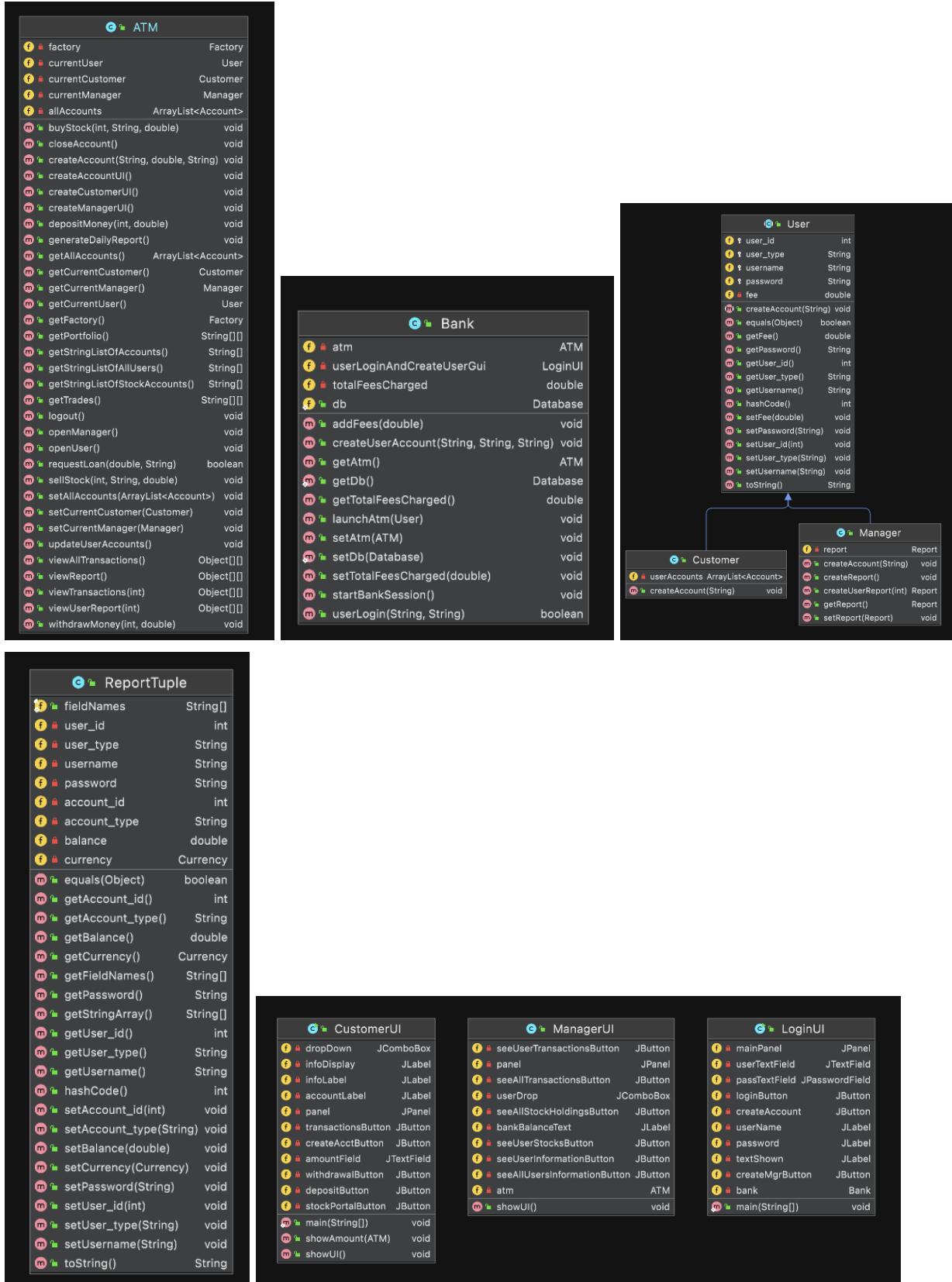
transaction_id	transaction_type	transaction_amount	transaction_time	account_id
1	W	1000.0	1969-12-31 19:00:00.0	1
4	deposit	0.0	1969-12-31 19:00:00.0	1

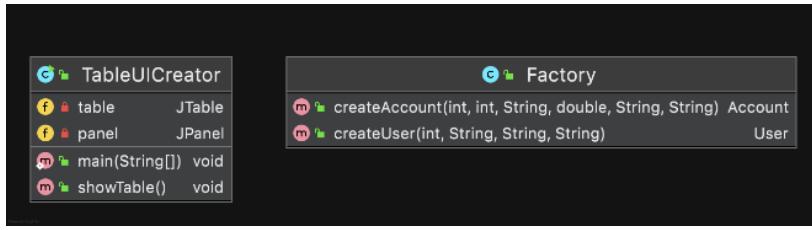
Object Design

The third component that needed to be designed and implemented to complete the application was the object-oriented design needed to be outlined in detail. As a team the OO-design was done by starting with a clear understanding of the requirements which have been outlined in the requirements specification section of this design document. Once the requirements were understood a detailed UML was created to capture all the necessary objects and connections needed for the logic of the application including the connection between the front-end GUI and the back-end database. In general, the use of design patterns and object-oriented design principles were also thought out during the process of generating the UML diagram. The initial UML had four main components in it, the first was a Bank object that was connected to an ATM object and the appropriate user types (customer/manager) that would be instantiated once a user logged into their account. The second group of objects was an abstract account object with the various account types as children objects. The third major component was based on the transactions that a

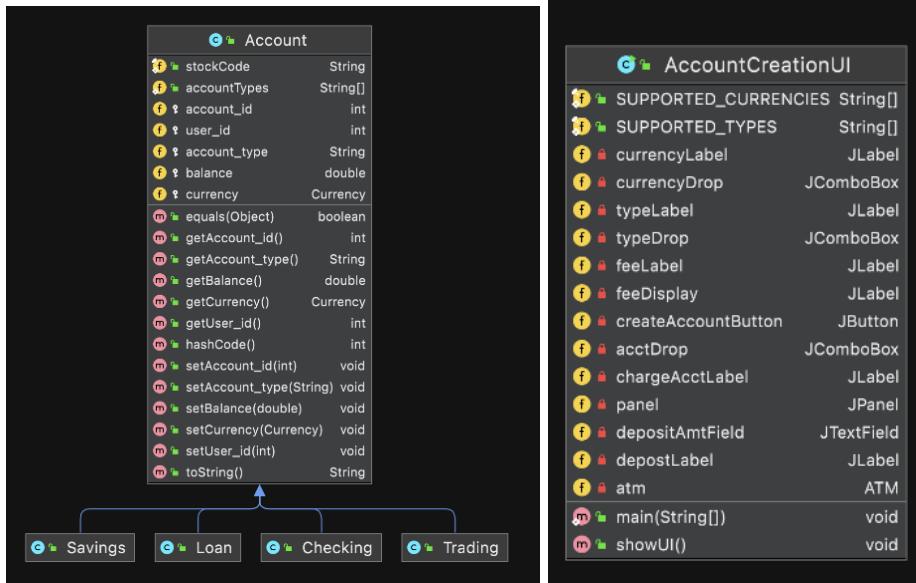
user would be able to enact as part of the services suite the bank offered. The transactions class acted as a parent class for all types of transactions to stem from in the application. The fourth major component was the stock market object which captured the external to the bank stock market that the Bank Manager requested certain customers would have access to trade. The UML components are shown below:

Component 1: Bank, ATM, and User

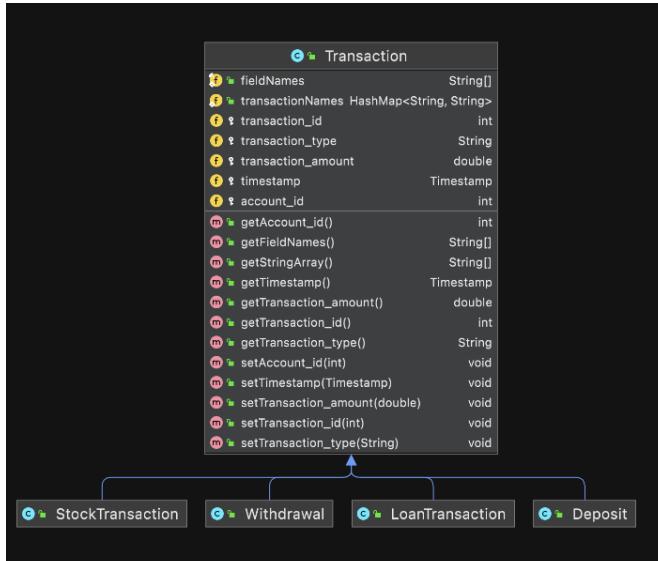




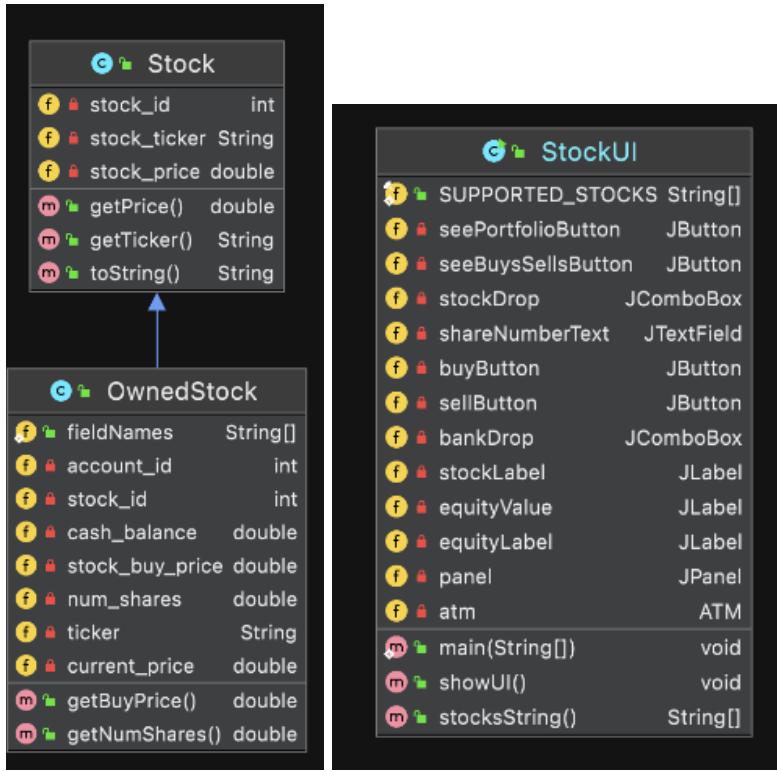
Component 2: Account/Account Types



Component 3: Transactions



Component 4: Stock Market



The complete UML is provided in the Appendix.

Conclusions

Overall, the end-to-end design and implementation of this Bank application that was requested by the Bank Manager proved to be a challenging tasks. Due to significant time constraints some of the requested functionalities were not implemented. Specifically, due to poor database design there is no way to capture the amount of money the Bank Manager has made from customers. Furthermore, the application does not have all of the requested constraints on accounts to differentiate which accounts have access to certain services offered by the Bank. Given fewer time constraints and a slight addition to the database all of the functionalities could be easily implemented especially due to the rigorous flexible structure of the database design, GUI design, and OO-Design the application can be easily extended to fulfill all the requested requirements.

Appendix

