ATM Banking Interface

Seth McCrudden and Jimmy Butcher

Abstract

This project proposal is about the requirements and the design used for planning an ATM styled banking interface that allows bank account holders to access their account(s) using a system that provides security and allows common banking transactions along with certain added features all made available through a simple and clearly presented visual screen interface that is easy to read and to understand by the user. This provides account holders a way to do business at when it is most convenient to their busy schedule.

1. Introduction

The ATM banking interface is designed to give users a secure, fast, convenient, and easy way to access their bank account(s) at any given time using a unique password to help insure only an authorized user can conduct transactions. The planning of this system will have security and protection of the account holders' monetary assets and privacy as a top priority.

The screen interface gives the user the ability to do withdrawals, deposits, transfers from one account to another, and it provides a fast and easy way to check and print the available balance for any account that the user has access to.

This interface will provide an easy-to-understand visual screen presentation with the goal of giving the user an easy-to-understand presentation to provide a positive experience for conducting banking transactions.

1.1. Background

The system will be implemented as a Windows Presentation Foundation (WPF) (.NET Framework) console application using the C# programming language. The design of the banking interface will be based upon the appearance and functionality of an ATM (Automated Teller Machine) screen user interface (UI). For security the system will use a user provided password. This project choice is based upon the real-world functionality and practical application of such a project.

1.2. Impacts

A well thought out banking interface system can offer a means of saving users (bank account holders) time and provides added convenience in accessing their cash assets and performing banking transactions.

1.3. Challenges

How to best implement allowing users to change their password if needed and also allowing multiple users to access the same account.

2. Scope

The product will have an attractive, easy to understand appearance that provides a simple and straightforward screen clearly showing the available transaction options. The system will have a way to set user passwords, set up accounts, close accounts, make deposits, withdrawals, transfer between accounts, adjust the account balance according to any given transaction as well as the ability to check and print the balance of any account. It will also print a receipt for any transaction.

2.1. Requirements

The banking interface will need to have an aesthetically appealing screen interface that is simple, easy to understand, and provides the following capabilities and features: The ability to open an account, close an account, make a deposit, make a withdrawal, transfer between accounts, adjust each account balance per user transaction, check available account balance, and the ability to set a secure password.

Use Case ID	Use Case Name	Primary Actor	Complexity	Priority
1	Withdraw amount from bank account	Account holder	Med	1
2	Deposit amount into account	Account holder	Med	1
3	Transfer amount from one account to another	Account holder	Med	1

TABLE 1. SAMPLE USE CASE TABLE

2.1.1. Functional.

- System allows user to set up user account with user name and password
- System prompts user for user name and password
- System to verify the the user name and password for secure login
- System to present any available accounts to the user
- System allows user to open an account
- System allows user to close an account
- System allows user to deposit to an account
- System allows user to withdraw from an account
- System allows user to transfer between accounts
- System adjusts account balance per transaction
- System prints receipt showing any and all transactions
- System prints available account balance
- System allows user to cancel transaction at any time before completion
- System allows user to log out of user account

2.1.2. Non-Functional.

- Security – user password mush be encrypted
- System has key pad and/or text boxes for data entry
- Screen interface must be easy to read and understand

2.2. Use Cases

Use Case Number: 1
Use Case Name: Withdraw from account
Description: A user logs into their user account, withdraws an amount from their bank account, views a receipt printed for the transaction, and logs out of their user account.

You will then go on to (minimally) discuss a basic flow for the process:

1) User logs in to user account with user name and password.
2) User select option to withdraw from an account
3) User selects account to withdraw from.
4) User enters amount to withdraw.
5) User confirms transaction.
6) User selects to view receipt showing transaction and updated account balance.
7) User selects to log out of user account.

Termination Outcome: The user now has withdrawn the selected amount from their account.

Use Case Number: 2
Use Case Name: Deposit to account
Description: A user logs into their user account, deposits an amount into their bank account, views a receipt printed for the transaction, and logs out of their user account.

1) User logs in to user account with user name and password.
2) User select option to deposit to an account
3) User selects account to deposit to.
4) User enters amount to deposit.
5) User confirms transaction.
6) User selects to view receipt showing transaction and updated account balance.
7) User selects to log out of user account.

Termination Outcome: The user now has deposited the selected amount to their account.

Use Case Number: 3
Use Case Name: Transfer from one account to another

Description: A user logs into their user account, transfers an amount from bank account to another account, views a receipt printed for the transaction, and logs out of their user account.

1) User logs in to user account with user name and password.
2) User select option to transfer from one account to another account.
3) User selects account to transfer from.
4) User selects account to transfer to.
5) User enters amount to transfer.
6) User confirms transaction.
7) User selects to view receipt showing transaction and updated account balance.
8) User selects to log out of user account.

Termination Outcome: The user now has transferred the selected amount from one account to another account.

2.3. Interface Mockups

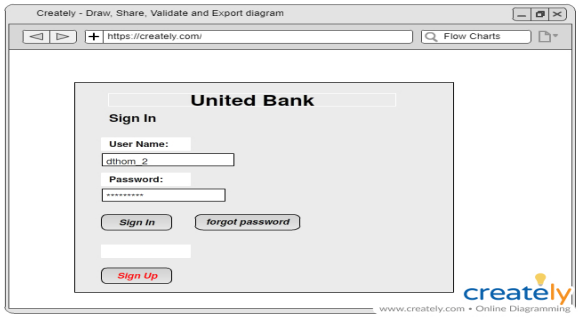

Figure 1. Bank Interface Login

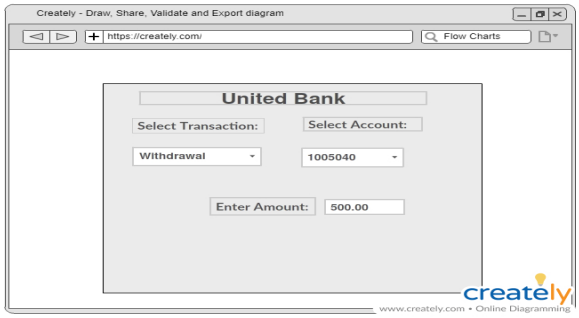

Figure 2. Bank Interface Withdrawal

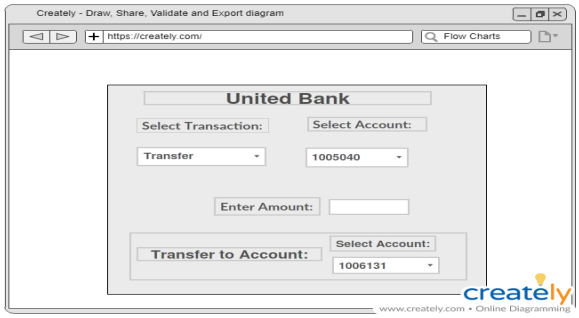

Figure 3. Bank Interface Transfer

3. Project Timeline

Go back to your notes and look up a typical project development life cycle for the Waterfall approach. How will you follow this life cycle over the remainder of this semester? This will usually involve a chart showing your proposed timeline, with specific milestones plotted out. Make sure you have deliverable dates from the course schedule listed, with a plan to meet them (NOTE: these are generally optimistic deadlines).

4. Project Structure

At first, this will be a little empty (it will need to be filled in by the time you turn in your final report). This is your chance to discuss all of your design decisions (consider this the README's big brother).

4.1. UML Outline

Show the full structure of your program. Make sure to keep on updating this section as your project evolves (you often start out with one plan, but end up modifying things as you move along). As a note, while Dia fails miserably at generating pdfs (probably my fault), I have had much success with png files. Make sure to wrap your images in a `figure` environment, and to reference with the `ref` command. For example, see Figure **??**.

4.2. Design Patterns Used

Make sure to actually use at least 2 design patterns from this class. This is not normally part of such documentation, but largely just specific to this class – I want to see you use the patterns!

5. Results

This section will start out a little vague, but it should grow as your project evolves. With each deliverable you hand in, give me a final summary of where your project stands. By the end, this should be a reflective section discussing how many of your original goals you managed to attain/how many desired use cases you implemented/how many extra features you added.

5.1. Future Work

Where are you going next with your project? For early deliverables, what are your next steps? (HINT: you will typically want to look back at your timeline and evaluate: did you meet your expected goals? Are you ahead of schedule? Did you decide to shift gears and implement a new feature?) By the end, what do you plan on doing with this project? Will you try to sell it? Set it on fire? Link to it on your resume and forget it exists?