## Team #6: Saul Gonzalez, Gian Hernandez, Bruno Valencia

## November 12, 2024

CS 3331 – Advanced Object-Oriented Programming – Fall 2024

## Instructor: Dr. Bhanukiran Gurijala

## Project Part 2 Lab Report

This work was done individually/as a team and completely on my/our own. I/we did not share, reproduce, or alter any part of this assignment for any purpose. I/we did not share code, upload this assignment online in any form, or view/received/modified code written from anyone else. All deliverables were produced entirely on my/our own. This assignment is part of an academic course at The University of Texas at El Paso and a grade will be assigned for the work I/we produced.

(NOTE: Please write in complete sentences (paragraph form). Check grammar, punctuation, and ensure your writing is clear. Write enough to make sure you clearly explain each part of the question.)

# **Program Explanation**

In this part of the project, we were tasked to handle more users and implement a functionality for the bank manager. We also refactored our code to make sure that each class was easily understood and could work without having too much in the main class. Since we already had a foundation for the program, we were able to build from that. First, we started with modifying the UML Use Case and Class diagrams to make sure we had all the classes needed in this assignment, then we moved on to refactoring the main class and moved a lot of the bank methods into new classes to clean up the code. We created new classes for more implementations to include a class that will create users from the console and add them to the csv.

# **What did I learn?**

This part of the project helped us understand the importance of refactoring and how we can use interfaces to as a blueprint for other classes that may use the methods inside of that interface class. It was a challenge trying to implement newer classes from the original code but once we started cleaning up, it was easier to understand how we could implement them. There are still areas that could be improved, one of which would be to minimize the amount of code we have written by condensing if statement and for loops. The assignment was lengthier this time so it took most of the time that was assigned to finish what we could.

# **Solution Design**

What did you do in this program?

What was your approach to solving this problem?

What data structures did you use? Why?

What assumptions, if any, did you make?

In this program we implemented a interface class called bankOperations, this class acts like a control where the checkings, savings, and credit classes have the same basic functions. But they can still do their own thing. Another implementation in this program was the adding users through the console, here we prompt the user to input some information that will then add the to the CSV. This was all in the UserRegistration class. We added a few methods to assign a random ID that is 4 digits and does not repeat. Also, each new user will have a starting balance of 15$ and depending on what they put as their credit score, it will assign them a random credit start balance (Based off of the parameters in the pdf) All of which gets saved in the CSV and can be accessed from the menu when prompting an existing customer. We included some test cases and a test unit suite that will handle all the test cases for each class.

# **Testing**

How did you test the program?

Did you use black-box, white-box testing, or both? Why?

Did you test the solution enough? How can the testing practices be improved?

What are the test cases I used?

Did you break the program and use that to improve it?

We had various methods of testing, to include testing some of the methods with Junit testing, we had the program test for deposit, withdraw, etc. by assigning a starting value and seeing that the methods worked as should. We also tested by running the program and making sure the output values from creating a user created the user. Also making sure that when a user pays someone, it updates the balance on both the sender and receiver.

# **Test results**

Describe the results of your tests.

Include any console outputs showing your results.

Include any text document results of your tests.

Testing adding a user:

# **Code Review**

Explain how you conducted a review of your code. Describe how you checked each part of the code review checklist.

(Note: Turn in all source code, output results (if applicable), reports, and all other required material specified by the assignment). Save this lab report at Team#\_PA# – Do not turn in report with notes.)