Andrew Guerra

Joshua Jaime

Professor Onwuka

Compsci 2

5/12/2025

ATM FINAL REPORT

Our inspiration for this project came from Automated Teller Machines (ATM) as they allow their users to deposit and withdraw money. We also took inspiration from bank apps operations where you can access your Checking Account, Savings Account and Credit Account. Another thing that inspired us was the use of username and password because most apps now require the user to create their own personalized username and password so we wanted to learn how to create this for our own model. With these inspirations we created our own model using C++ programming language using OPP fundamentals such as classes, encapsulation, inheritance, and polymorphism. Our model uses 3 main classes: BankAccount, User and Bank. The Bank class manages user interactions, transactions and the menu. Bank also has a run function that contains a while loop which allows the system to run and repeatedly display the menu. Each user has a Checking Account, Savings Account and Credit Account, that each derive from the base class BankAccount. We actually encountered a problem with the bank account

class where we had created a separate deposit() function for the Credit Account function. When we had done that the program would not run correctly for creditAccount so we decided to use the same inherited deposit function from BankAccount we just needed to override it. The purpose of overriding the deposit function because its original purpose in the base class was to add an amount to the balance however in the creditAccount class its purpose is for the amount to reduce the credit owed. We also had a problem where when the system needed to call the CreditAccount constructor it would rather call the BankAccount constructor which doesn't align with the CreditAccount logic. That is when we decided to use virtual which overrides the BankAccount constructor when needed which allowed the CreditAccount constructor to run properly. These problems helped us get insights on destructors and why they are also important for inheritance. Another problem we encountered was aligning our code which wasn't a big hassle just required a few modifications. In addition, the log in and log out feature provided us insight on how user authentication works. A key point was Joshua's method of using User* currentUser pointer which tracks the session state to ensure that deposit and withdrawal actions are only accessible when the user is logged in. We got great insights on how to organize our code and properly utilize important techniques that are needed for c++.