Ben Fagan
July, 11, 2025
CIS25
Prof. Brooks

Part 1: Project Overview

My project focuses on a very simple banking simulation. Its architecture is divided into roughly four parts: a class that stores client data, a class that stores financial data for the bank, a class that stores information derived from both classes for import, export, and search applications, and a number of helper functions that mostly validate user input. Client facing options include deposit, withdrawal, check balance, & check transaction history. Bank employee options include searching client accounts by name or account number and printing a report that compiles the values of all transactions across all clients.

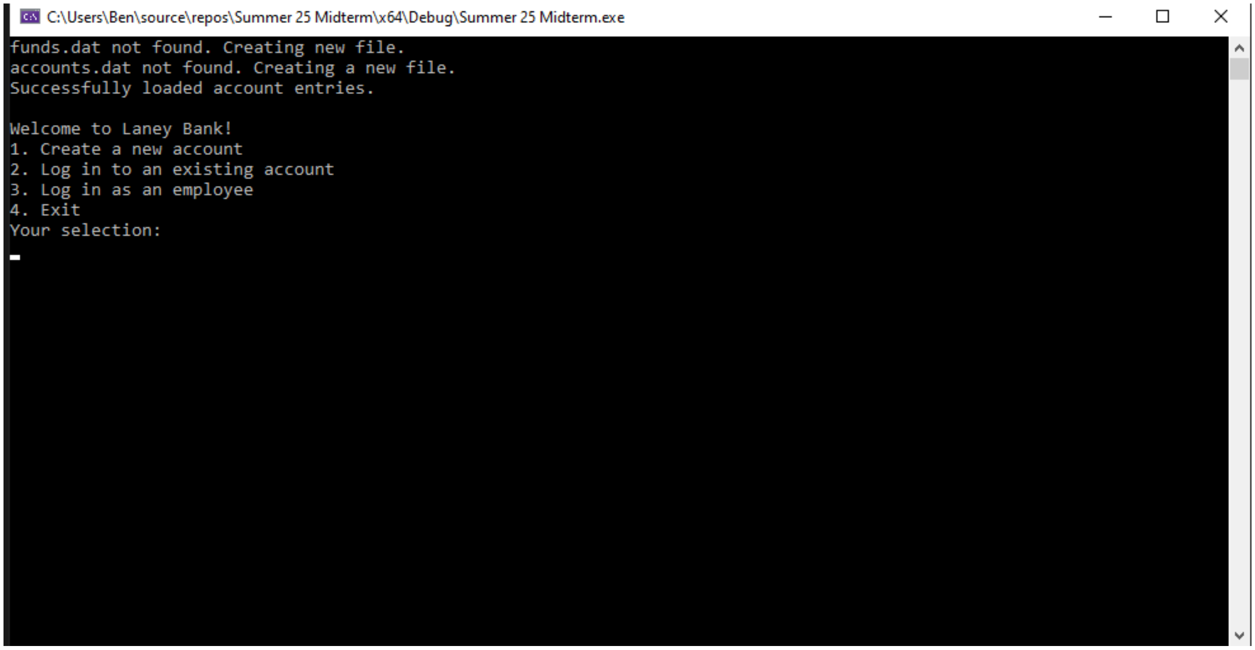
Part 2: Concepts Used & Inclusion Checklist

Required concepts:

- **File types**: This program exports to & imports from two binary files and contains serialization functions to assist.
    - See: Storage.cpp: saveAccount, saveBank, loadAccount, and loadBank. (lines 131-268).
    - See also: BankFunds.cpp: serialize, deserialize (lines 92-123).
    - See also: Accounts.cpp: serialize, deserialize (lines 206-303).
- **Pointers**: Pointers are used extensively to pass information to and from functions, and also employs a pointer to direct account access with user log in.
    - See: Validators.cpp: userLogIn (lines 111-151).
- **Arrays:** While this program mostly uses vectors to store data, an array is used to store individual client transaction information.
    - See: Account.h (lines 33-43).
    - See also: Account.cpp: logTransactions (lines 168-194).
- **Search:** This program employs two range based for loops to iterate over vectors containing client information and compares each relevant index to user input.
    - See: Storage.cpp: searchAccounts (lines 69-123).
- **Strings:** Strings are used extensively in this program, both for user input & to present information to the user.
    - See: Account.h: private member variables (lines 19-43).
- **File I/O:** This program writes files to and reads files from binary in the .dat format. It also initializes those files with default information if they do not exist when the program begins.
    - See: Storage.cpp: saveAccount, saveBank, loadAccount, and loadBank. (lines 131-268).
    - See also: BankFunds.cpp: serialize, deserialize (lines 92-123).
    - See also: Accounts.cpp: serialize, deserialize (lines 206-303).
- **At least two classes with meaningful interaction:** This program employs three classes. Two act as a framework for different data (user and the bank), while the third acts as a container for the other two, which makes file storage and retrieval easier.
    - See: Storage.h private member variables (lines 15-20).

Part 3: Screenshots

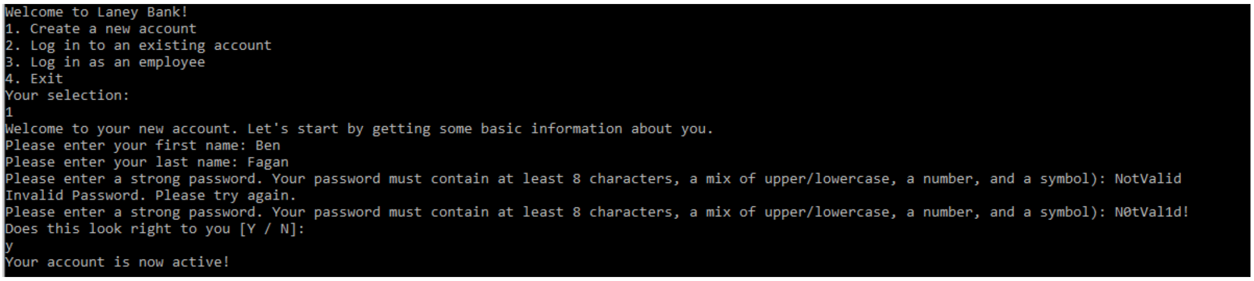
1. Entry menu with notifications for new .dat files.



C:\Users\Ben\source\repos\Summer 25 Midterm\x64\Debug\Summer 25 Midterm.exe

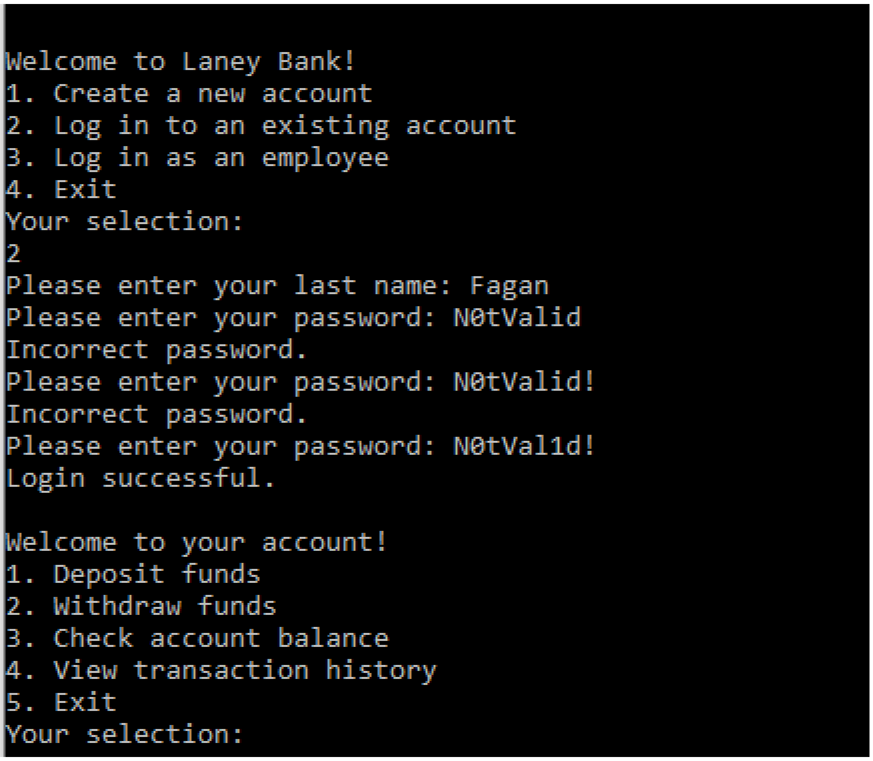
```
funds.dat not found. Creating new file.
accounts.dat not found. Creating a new file.
Successfully loaded account entries.

Welcome to Laney Bank!
1. Create a new account
2. Log in to an existing account
3. Log in as an employee
4. Exit
Your selection:
```

2. New Account Creation



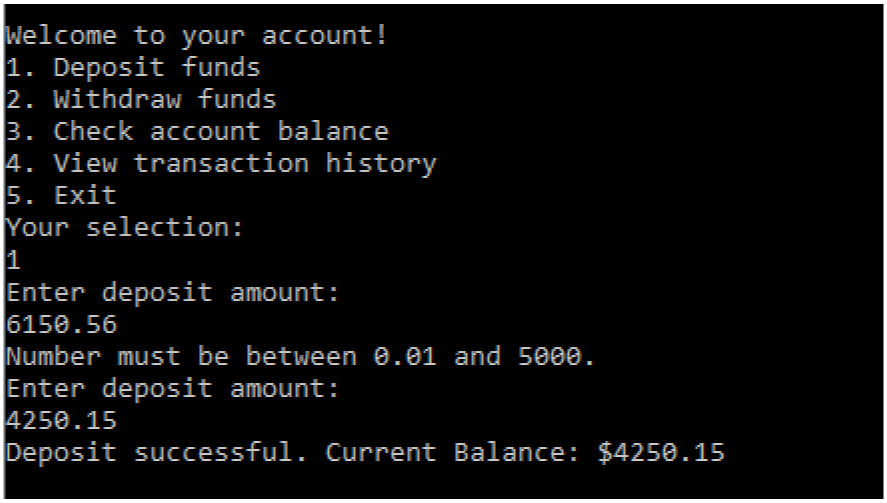
```
Welcome to Laney Bank!
1. Create a new account
2. Log in to an existing account
3. Log in as an employee
4. Exit
Your selection:
1
Welcome to your new account. Let's start by getting some basic information about you.
Please enter your first name: Ben
Please enter your last name: Fagan
Please enter a strong password. Your password must contain at least 8 characters, a mix of upper/lowercase, a number, and a symbol): NotValid
Invalid Password. Please try again.
Please enter a strong password. Your password must contain at least 8 characters, a mix of upper/lowercase, a number, and a symbol): N0tVal1d!
Does this look right to you [Y / N]:
y
Your account is now active!
```

3. Client Log In



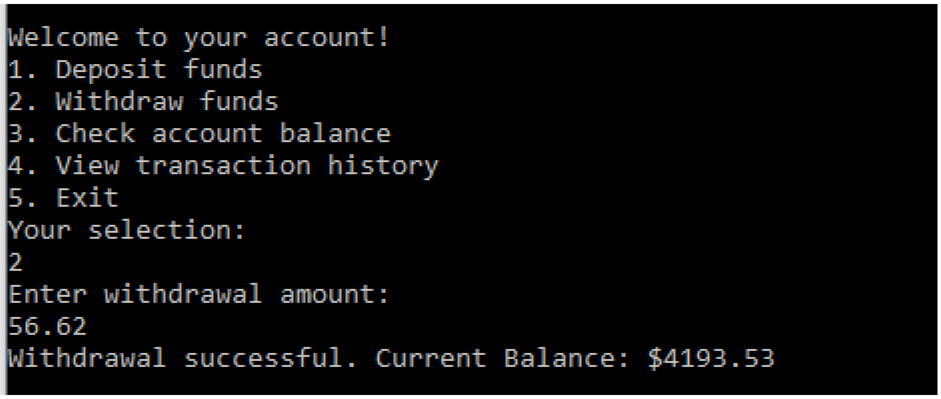
```
Welcome to Laney Bank!
1. Create a new account
2. Log in to an existing account
3. Log in as an employee
4. Exit
Your selection:
2
Please enter your last name: Fagan
Please enter your password: N0tValid
Incorrect password.
Please enter your password: N0tValid!
Incorrect password.
Please enter your password: N0tVal1d!
Login successful.

Welcome to your account!
1. Deposit funds
2. Withdraw funds
3. Check account balance
4. View transaction history
5. Exit
Your selection:
```

4. Deposit



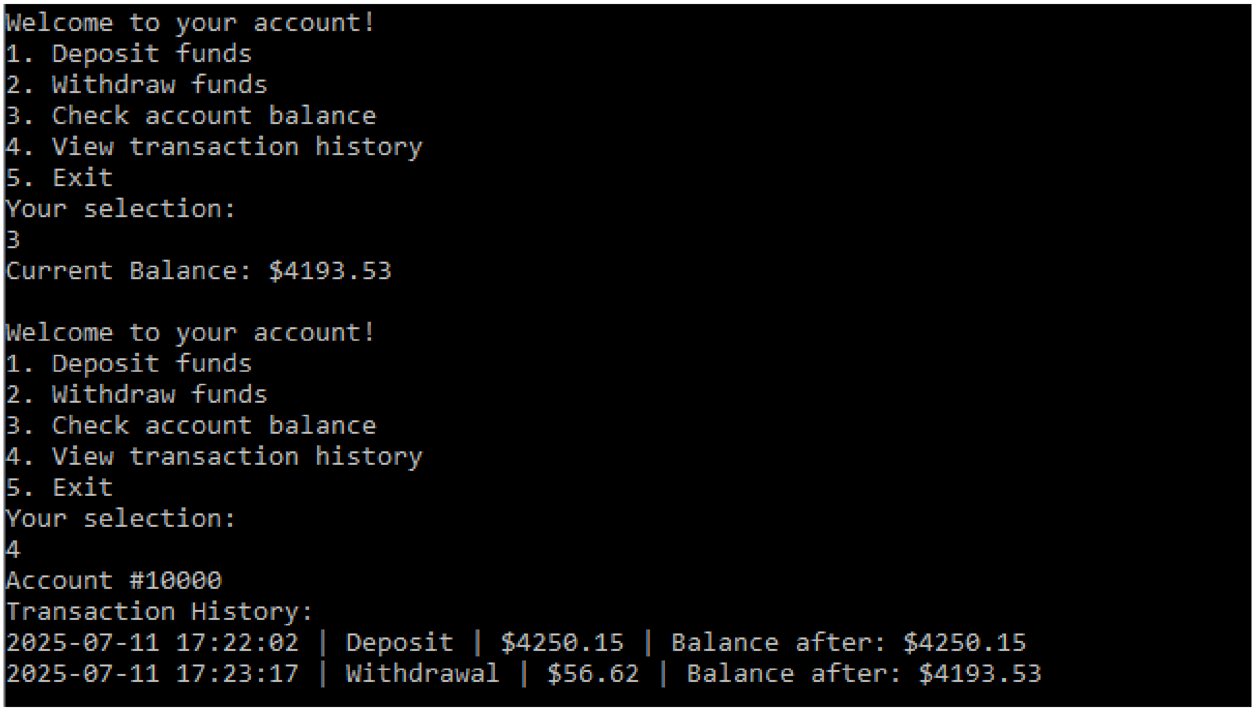
```
Welcome to your account!
1. Deposit funds
2. Withdraw funds
3. Check account balance
4. View transaction history
5. Exit
Your selection:
1
Enter deposit amount:
6150.56
Number must be between 0.01 and 5000.
Enter deposit amount:
4250.15
Deposit successful. Current Balance: $4250.15
```

5. Withdrawal



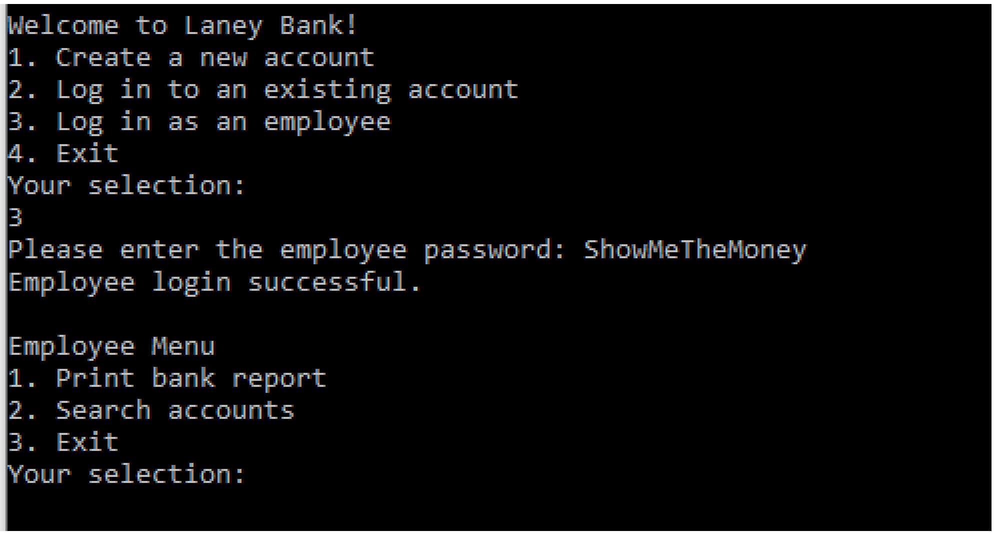
```
Welcome to your account!
1. Deposit funds
2. Withdraw funds
3. Check account balance
4. View transaction history
5. Exit
Your selection:
2
Enter withdrawal amount:
56.62
Withdrawal successful. Current Balance: $4193.53
```

6. Check balance & transaction history



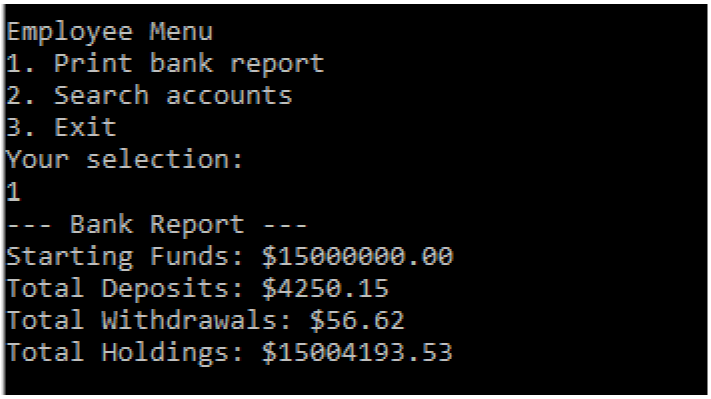
```
Welcome to your account!
1. Deposit funds
2. Withdraw funds
3. Check account balance
4. View transaction history
5. Exit
Your selection:
3
Current Balance: $4193.53

Welcome to your account!
1. Deposit funds
2. Withdraw funds
3. Check account balance
4. View transaction history
5. Exit
Your selection:
4
Account #10000
Transaction History:
2025-07-11 17:22:02 | Deposit | $4250.15 | Balance after: $4250.15
2025-07-11 17:23:17 | Withdrawal | $56.62 | Balance after: $4193.53
```

7. Employee Log In



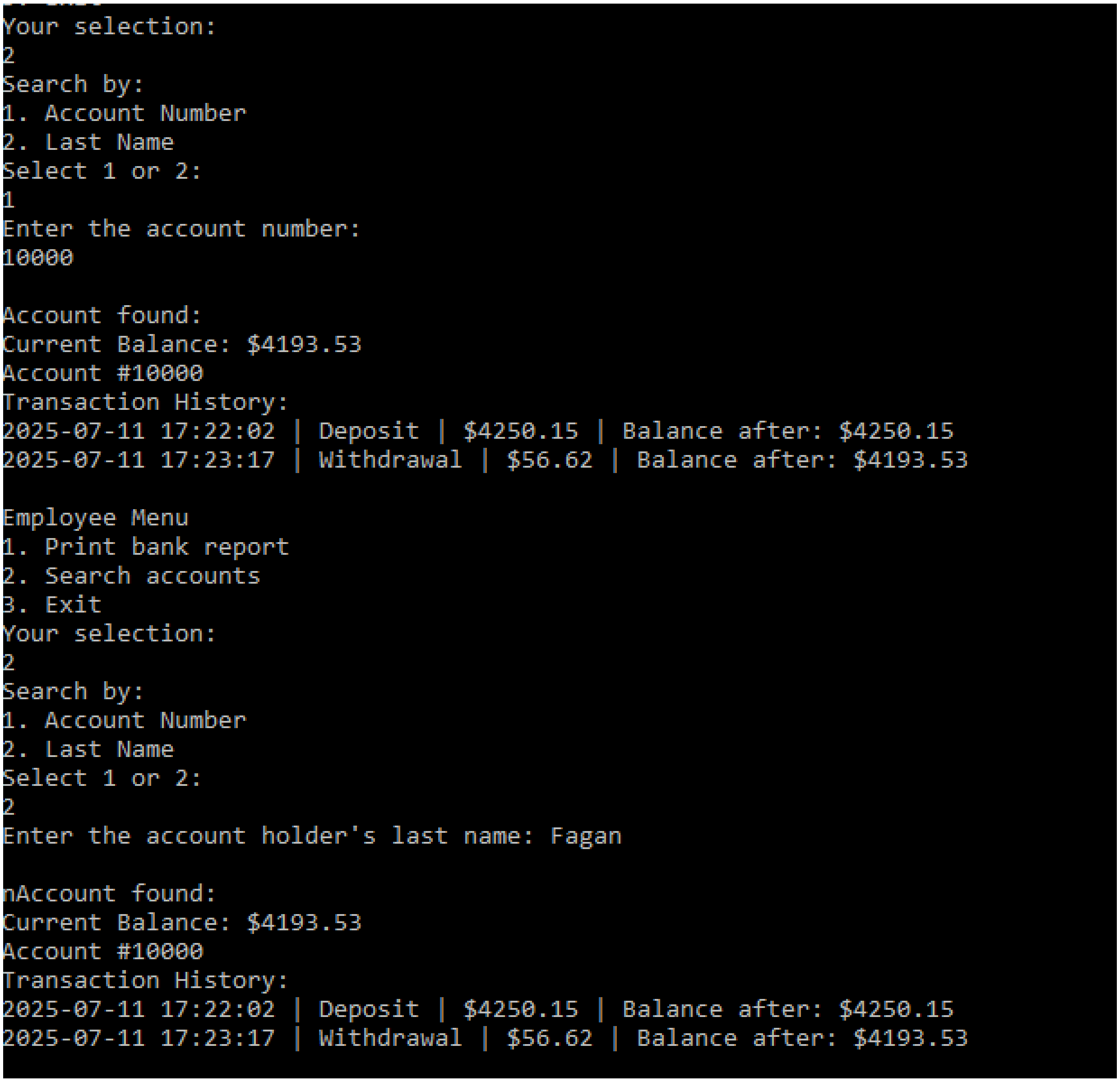
```
Welcome to Laney Bank!
1. Create a new account
2. Log in to an existing account
3. Log in as an employee
4. Exit
Your selection:
3
Please enter the employee password: ShowMeTheMoney
Employee login successful.

Employee Menu
1. Print bank report
2. Search accounts
3. Exit
Your selection:
```

8. Bank Report



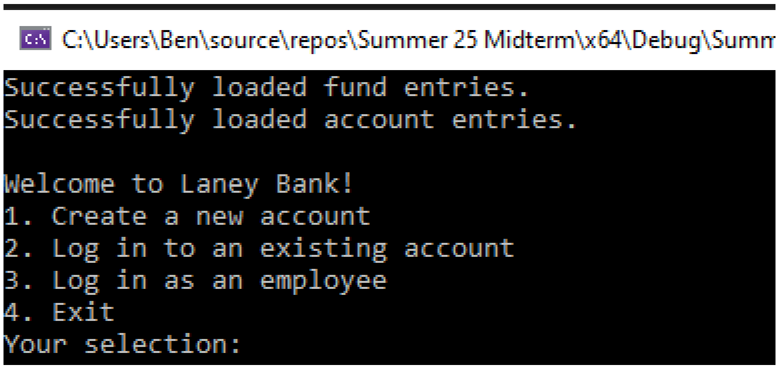
```
Employee Menu
1. Print bank report
2. Search accounts
3. Exit
Your selection:
1
--- Bank Report ---
Starting Funds: $15000000.00
Total Deposits: $4250.15
Total Withdrawals: $56.62
Total Holdings: $15004193.53
```

9. Employee Searches



```
Your selection:
2
Search by:
1. Account Number
2. Last Name
Select 1 or 2:
1
Enter the account number:
10000

Account found:
Current Balance: $4193.53
Account #10000
Transaction History:
2025-07-11 17:22:02 | Deposit | $4250.15 | Balance after: $4250.15
2025-07-11 17:23:17 | Withdrawal | $56.62 | Balance after: $4193.53

Employee Menu
1. Print bank report
2. Search accounts
3. Exit
Your selection:
2
Search by:
1. Account Number
2. Last Name
Select 1 or 2:
2
Enter the account holder's last name: Fagan

nAccount found:
Current Balance: $4193.53
Account #10000
Transaction History:
2025-07-11 17:22:02 | Deposit | $4250.15 | Balance after: $4250.15
2025-07-11 17:23:17 | Withdrawal | $56.62 | Balance after: $4193.53
```

10. Successful Program Reload



```
Successfully loaded fund entries.
Successfully loaded account entries.

Welcome to Laney Bank!
1. Create a new account
2. Log in to an existing account
3. Log in as an employee
4. Exit
Your selection:
```

Part 4: Reflection

This project provided a number of challenges that deepened my understanding of programming. The first is that I chose to challenge myself by writing and reading from binary .dat files. That meant that I had to learn about and implement serialization functions for both of the files that I used. The second challenge involved properly dividing classes into different .h & .cpp files. While I am familiar with how each of those things work, I did not realize that best practices were to have each class in a separate one. This led to refactoring my entire project when it was mostly complete. Third and finally, I wanted to focus on data integrity in a way that I haven't in prior projects. What this meant was paying much more attention to which variables to mark as const and to think about why I was doing that. Unlike the other challenges, there were fewer good resources on Stack Overflow or even ChatGPT. It ended up being a lot of trial and error, but I definitely have a greater appreciation for how and when to use const.