**Appendix 1: Assignment submission cover sheet**

**ASSIGNMENT SUBMISSION COVER SHEET**

| | |
|---|---|
| **Student Id:** | cse24-138 |
| **Student names:** | Shutu Tanakamunashe |
| **Student email**: | cse24-138@thuto.bac.ac.bw |
| **Cohort**: | 2025 |
| **Assignment title**: | OOAD |
| **Date of submission**: | 11/16/2015 |
| **Programme of Study**: | CSE |
| **Year of Study**: | Year 2 |

**Intellectual property statement**

By checking the box below, I certify that this assignment is my own work and is free from plagiarism. I understand that the assignment may be checked for plagiarism by electronic or other means and may be transferred and stored in a database for the purposes of data-matching to help detect plagiarism. The assignment has not previously been submitted for assessment in any other unit or to any other institution. **I**

**have read and understood the Botswana Accountancy College plagiarism guidelines policy.**

✓
☐ Agree **Signature**…… 11/16/2025
**Date**

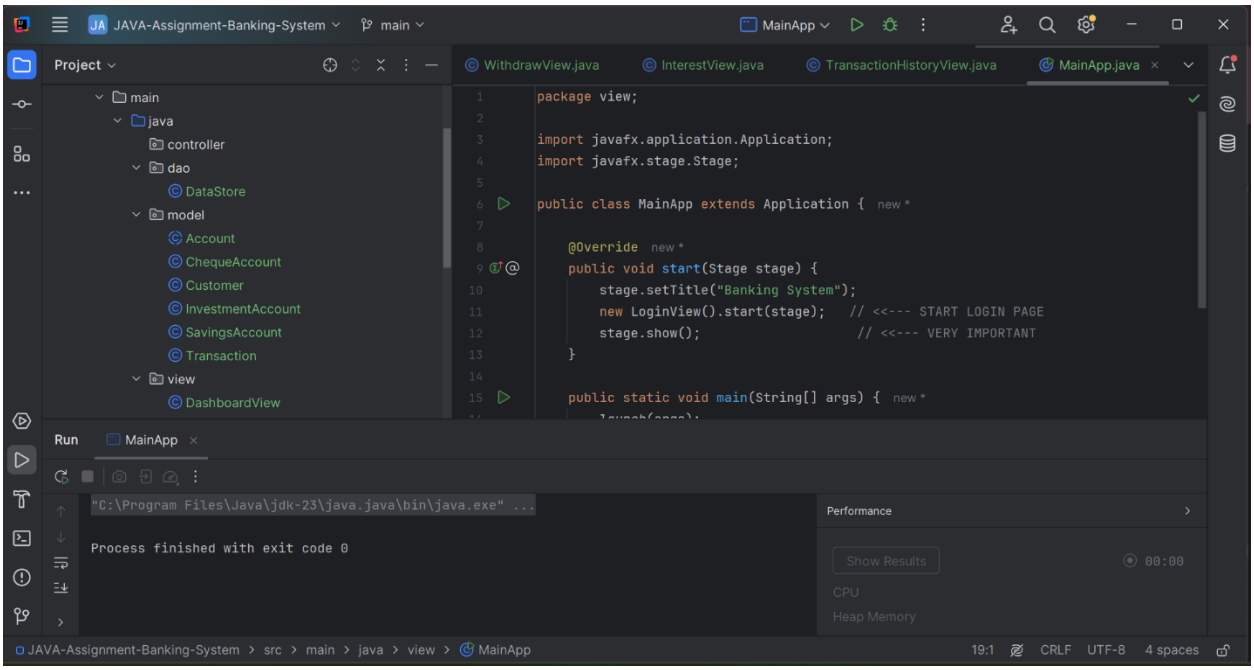
# Module Integration & Testing

### 8.1 Module Integration

All system components were integrated to work together as one complete application. The integration followed the MVC structure required by the assignment:

- **View Layer (GUI):** JavaFX screens such as LoginView, RegistrationView, DashboardView, DepositView, WithdrawView, ViewAccountsView, InterestView, and TransactionHistoryView.
- **Controller Layer:** Controllers handle input validation and trigger business logic.
- **Model Layer:** Core classes such as Customer, Account, SavingsAccount, InvestmentAccount, ChequeAccount, and Transaction.
- **Data Layer:** DataStore acts as the in-memory database storing customers, accounts, and transactions.

Overall, the system flow is:
**GUI → Controller → Model → DataStore → Model → Controller → GUI**

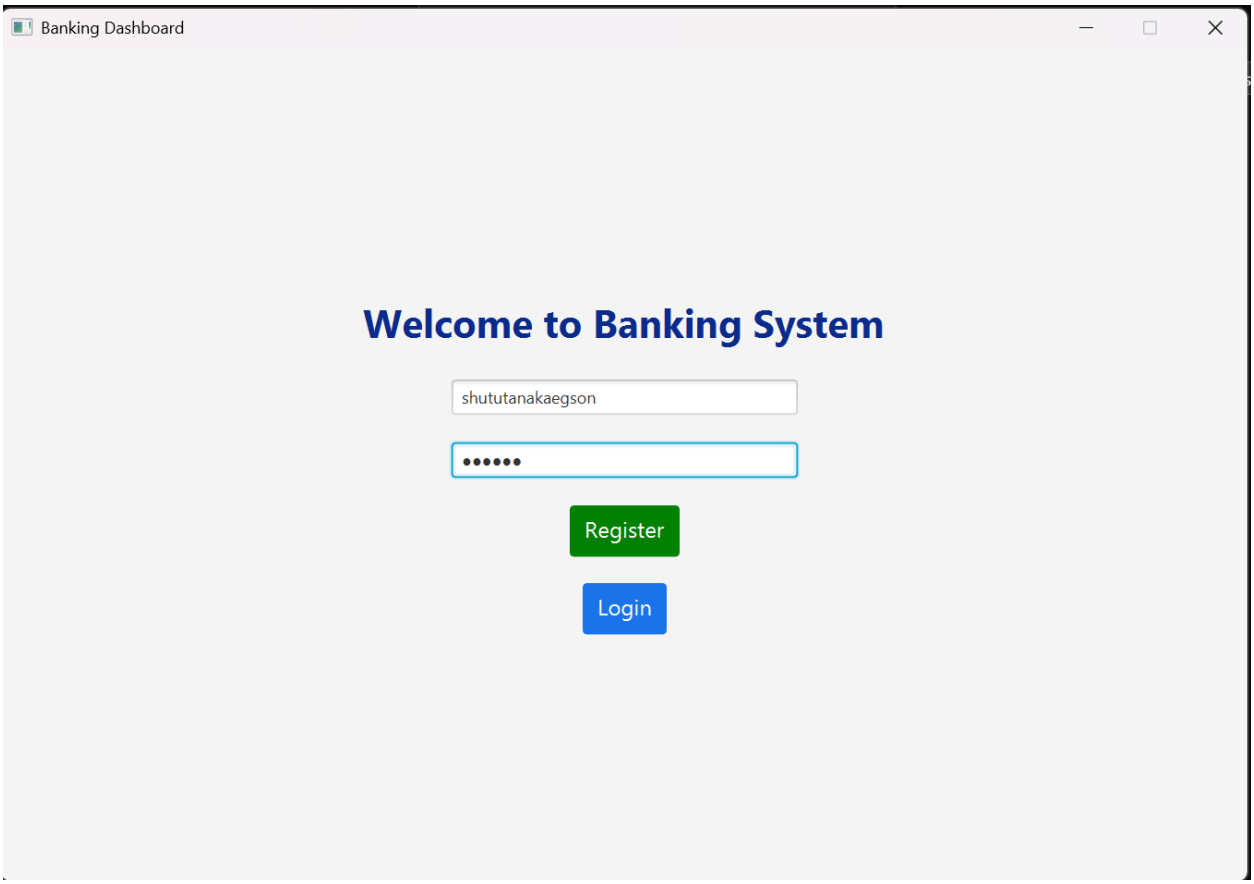



*Screenshot of my project folder structure (controller, model, view, DataStore)*

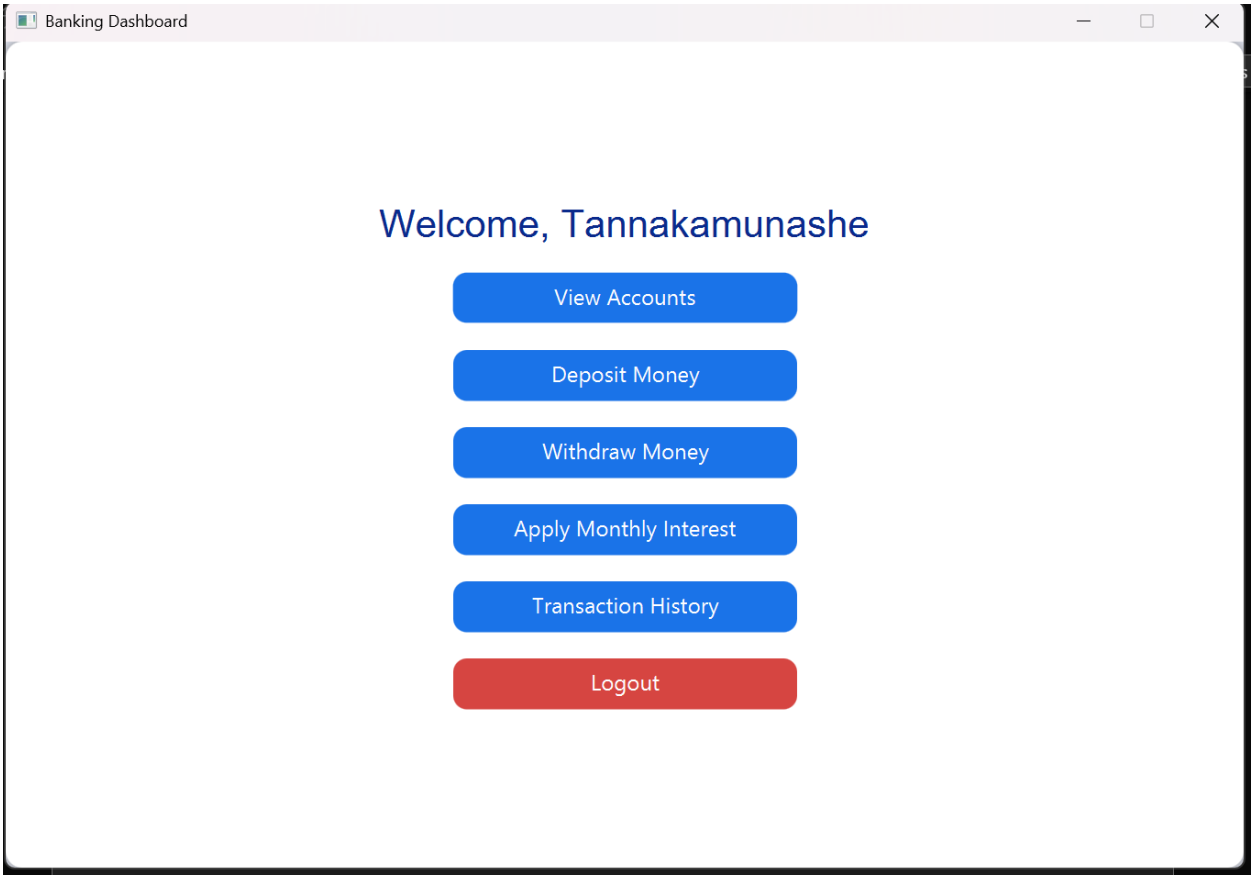
### 8.2 Integration Testing

Integration testing was performed to ensure that all modules interact correctly as a complete system. Below are the main test cases executed.

**Test 1 — User Login**

- **Steps:** Launch system → Enter username & password → Press Login
- **Expected:** System verifies credentials and loads Dashboard.
- **Actual:** Dashboard successfully displays the logged-in user's name.
- 

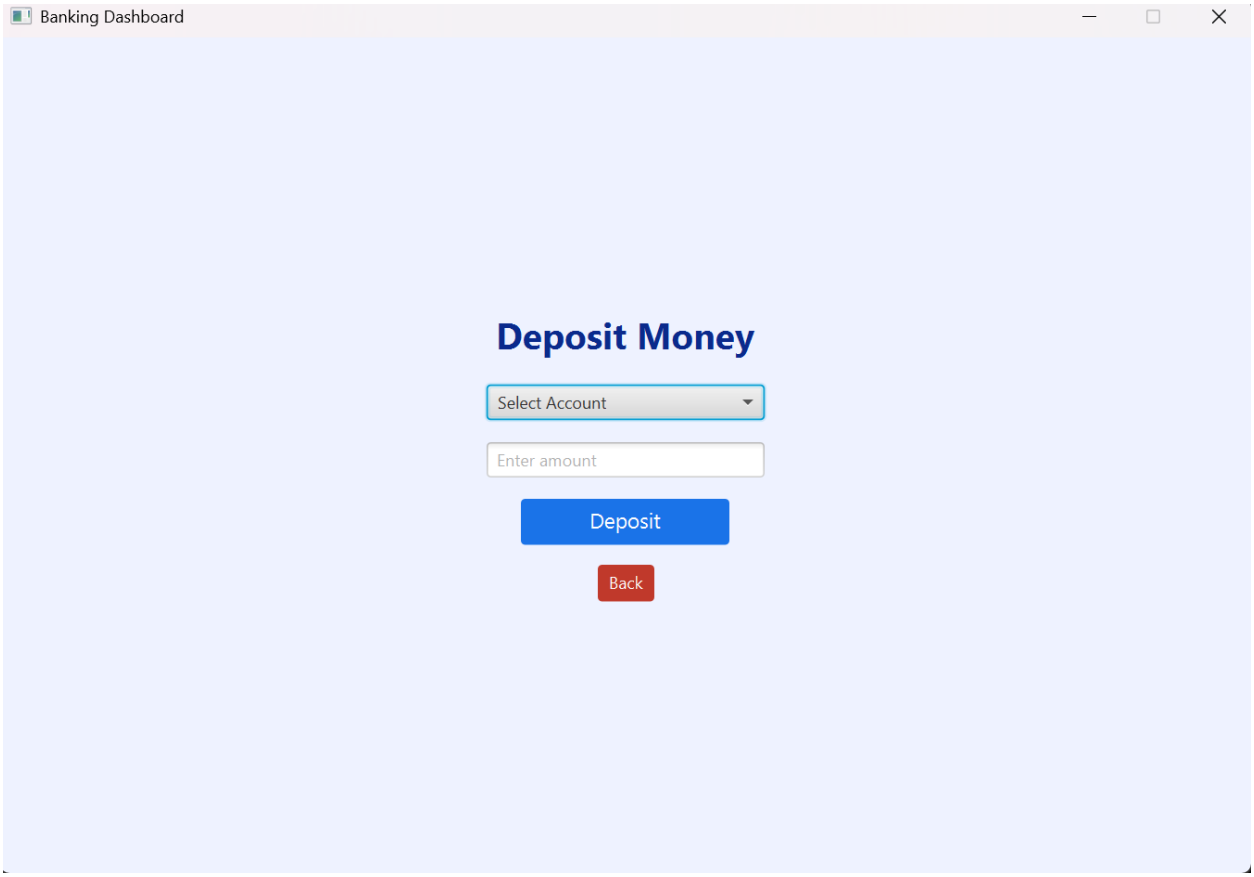

*Screenshot of Login Page*



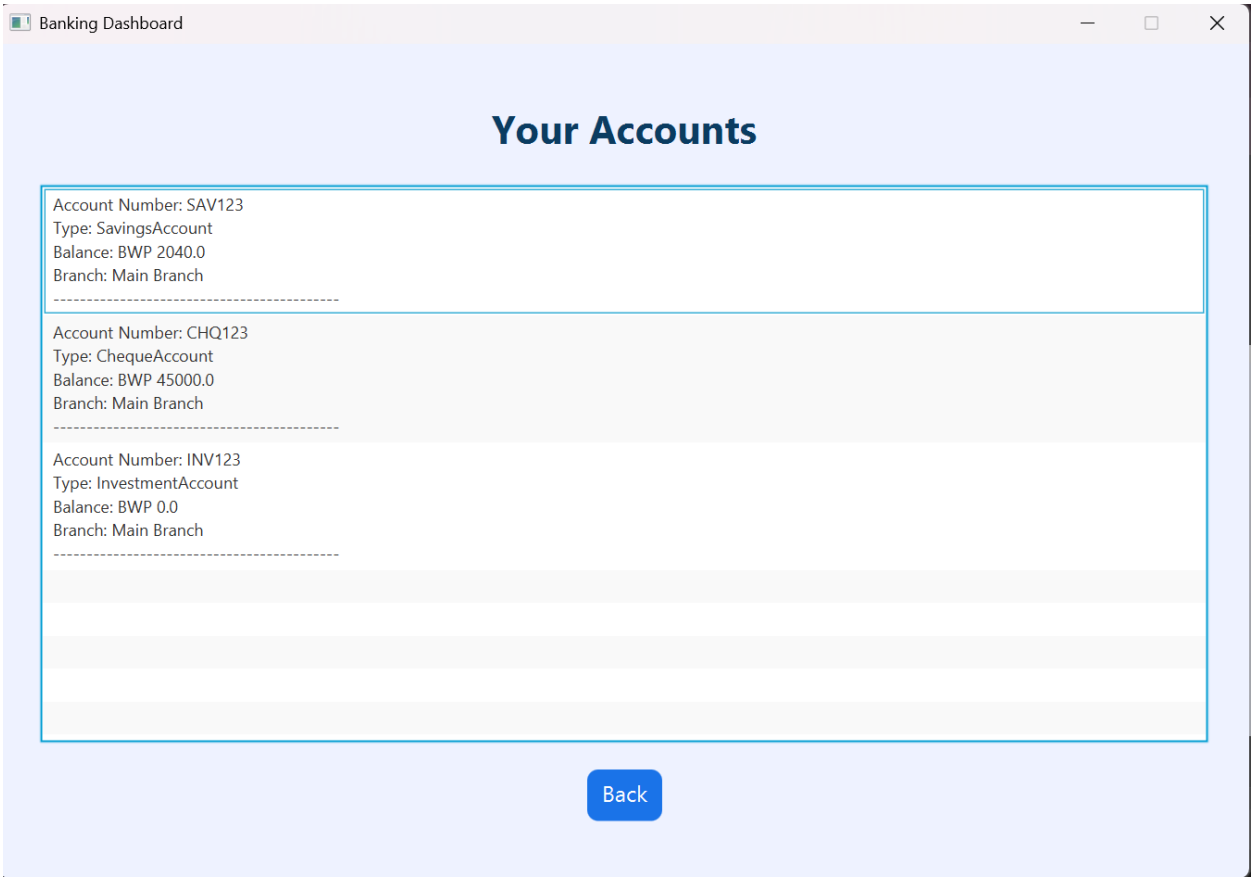

*Screenshot of Dashboard after Login*

**Test 2 — Deposit Money**

- **Steps:** Dashboard → “Deposit Money” → Choose account → Enter amount → Confirm
- **Expected:** Balance increases and transaction is recorded.
- **Actual:** Deposit works, updated balance shows, transaction saved in DataStore.



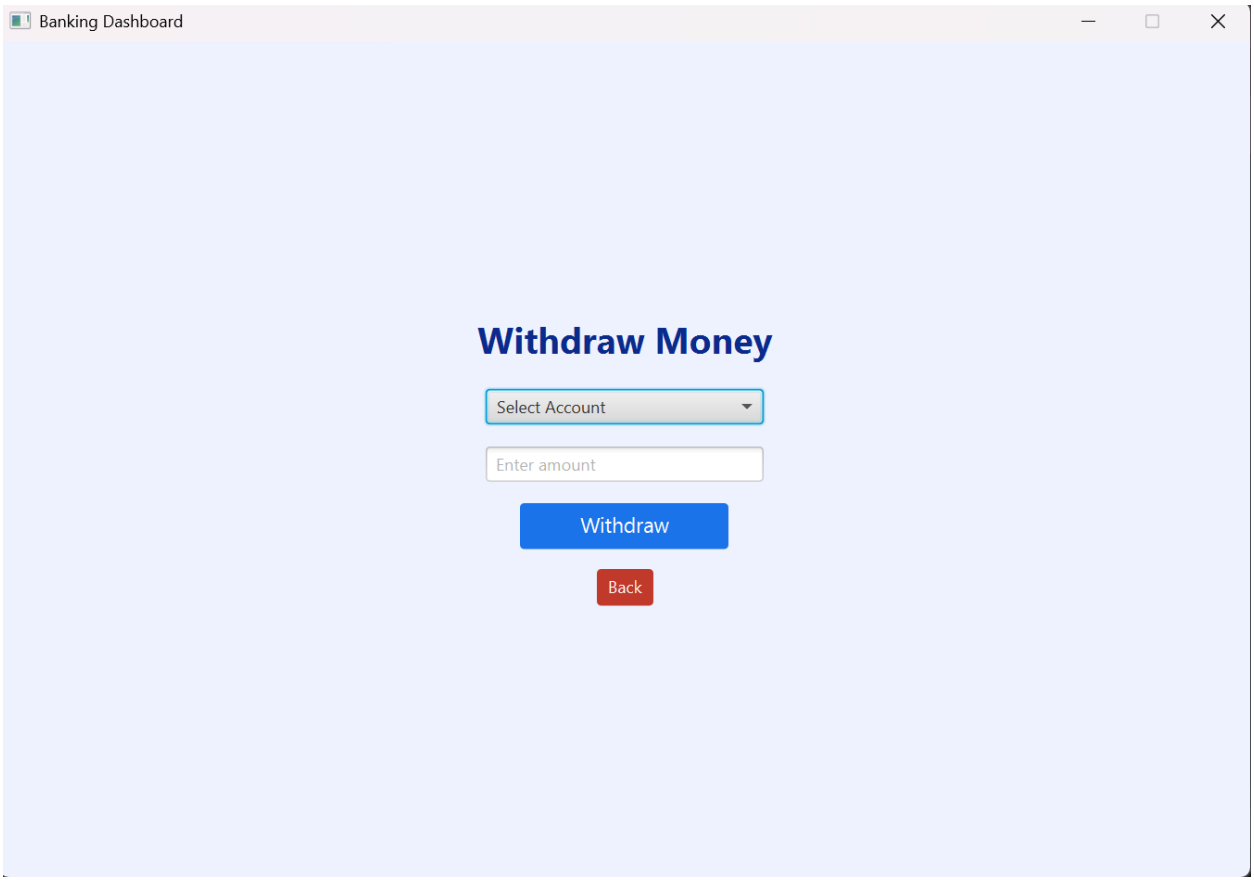

*Screenshot of Deposit Screen*




*Screenshot showing balance updated*

**Test 3 — Withdraw Money**

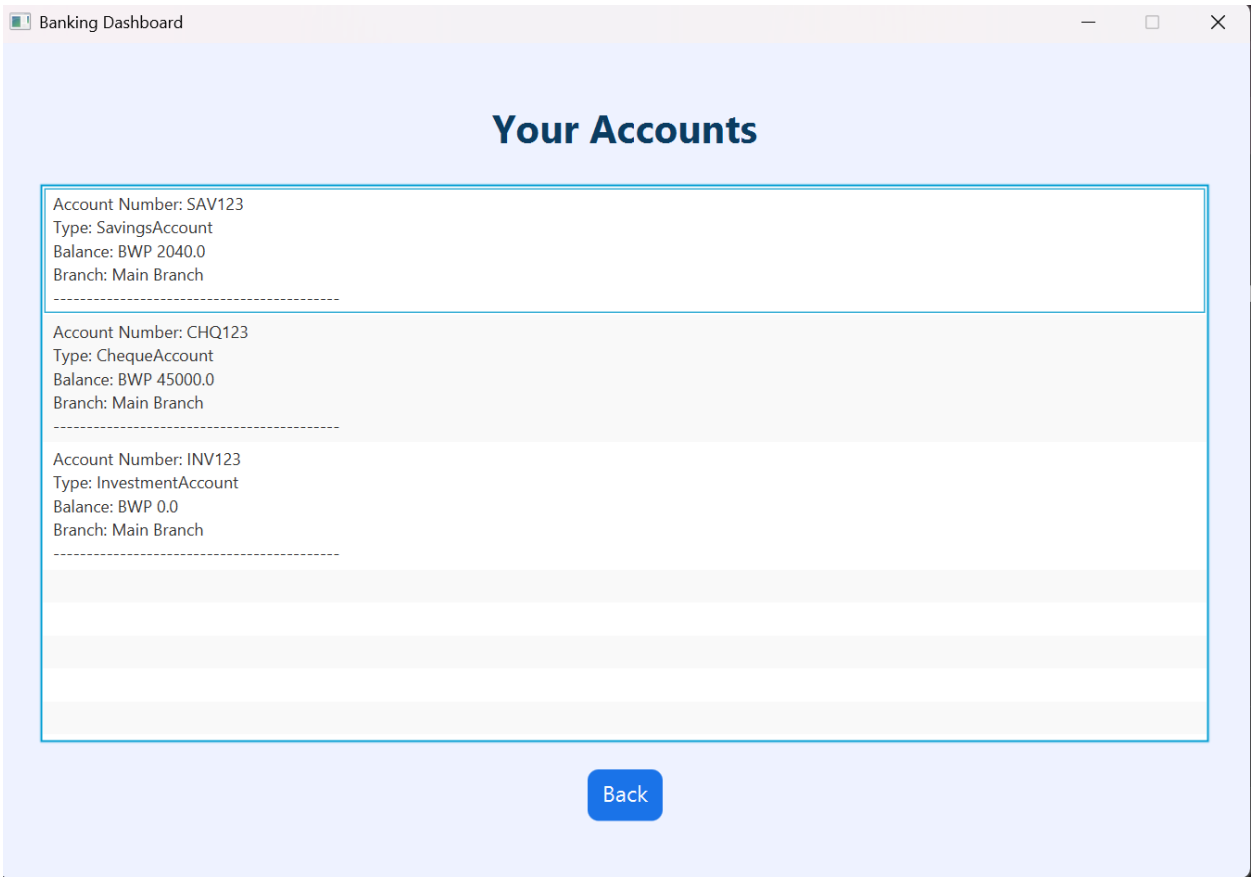
- **Expected:** Withdrawal reduces balance, logs transaction, errors for insufficient funds.
- **Actual:** Works correctly with correct validation.




*Screenshot of Withdraw Screen*

**Test 4 — View Accounts**

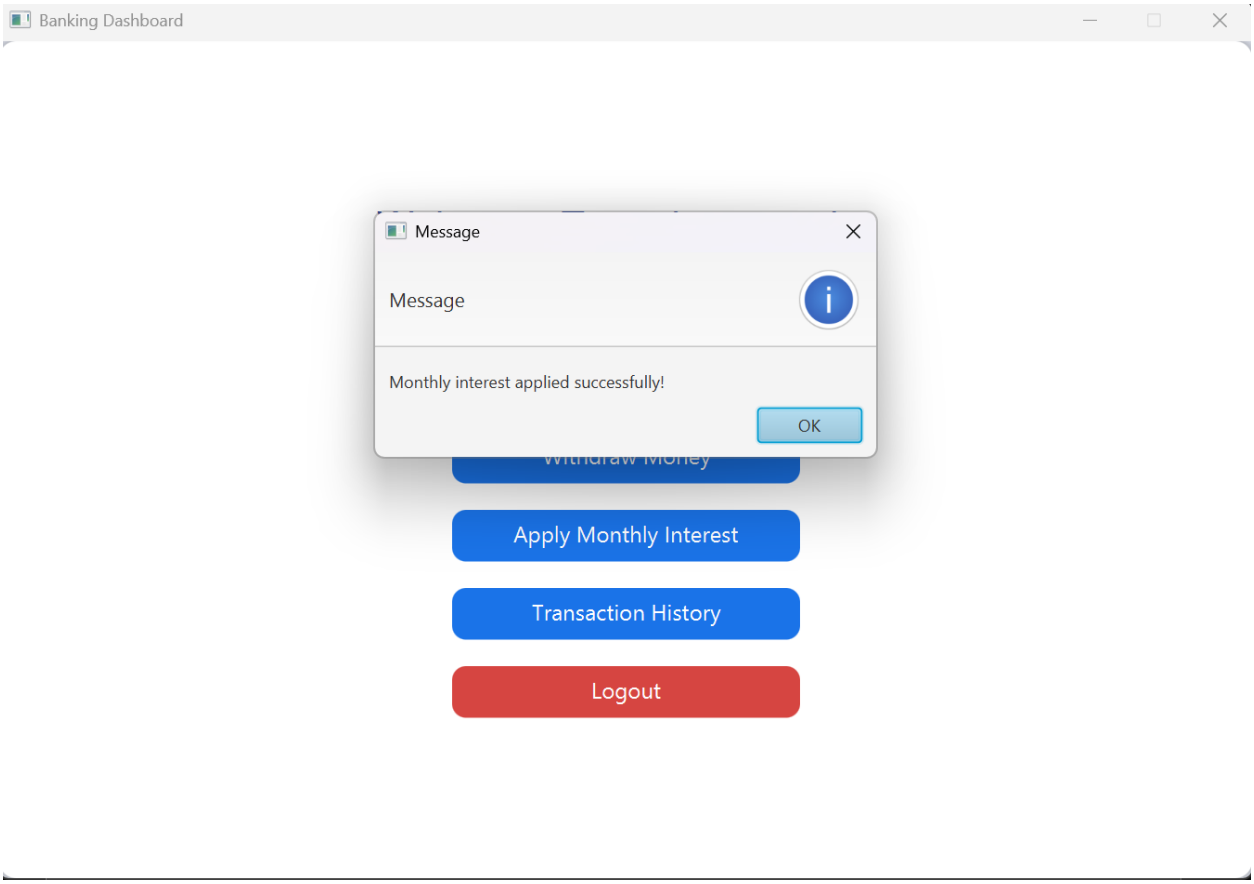
- **Expected:** All accounts and updated balances load from DataStore.
- **Actual:** Accounts appear correctly.




*Screenshot of View Accounts screen*

**Test 5 — Apply Monthly Interest**

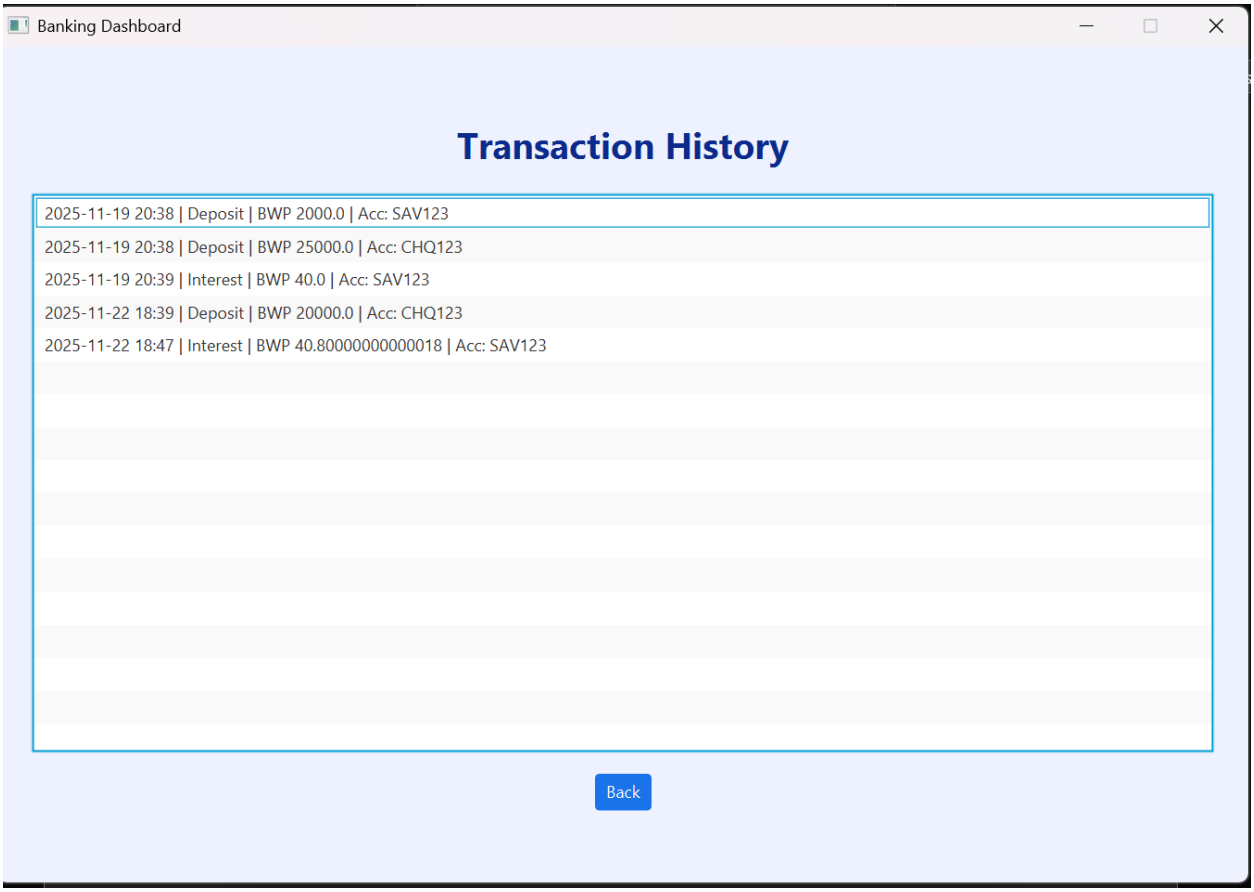
- **Expected:** SavingsAccount and InvestmentAccount apply correct monthly interest.
- **Actual:** Interest is calculated and updated correctly.




*Screenshot of Interest Application screen*

**Test 6 — Transaction History**

- **Expected:** All deposits and withdrawals appear correctly.
- **Actual:** Transaction list matches all performed operations.




*Screenshot of Transaction History screen*

### 8.3 Integration Test Summary

All modules of the application work together as expected. The GUI communicates correctly with controllers, controllers link correctly to model logic, and the DataStore maintains updated customer, account, and transaction information.
The system passed all integration tests without errors.