Names:
Ahmad Aljawish [100987855], Michael Carinci [100826903],
Patrik Prenga [100990129], Marco Pasqua [100991681]

CSCI 3060U Course Project - Winter 2025
Phase #1 - Front End Requirements

Part 3 – Front End Test Plan

1. Purpose

This document describes how the Front End requirement tests for the Banking System are organized, executed, and managed. The purpose of this test plan is to ensure that all required Front End behaviors are exercised using complete test session input streams and that outputs are recorded and compared for correctness.

2. Directory Organization

All Phase 1 materials are contained inside the top-level directory:

phase1FrontEnd/

The directory structure is as follows:

```
phase1FrontEnd/
 ├── docs/
 ├── fixtures/
 │   └── current_accounts.txt
 ├── tests/
 │   ├── Login Transactions/
 │   │   ├── inputs/
 │   │   └── expected_outputs/
 │   ├── Create Transactions/
 │   │   ├── inputs/
 │   │   └── expected_outputs/
 │   ├── Deposit Transactions/
 │   │   ├── inputs/
 │   │   └── expected_outputs/
 │   ├── Withdrawal Transactions/
 │   │   ├── inputs/
 │   │   └── expected_outputs/
 │   ├── Transfer Transactions/
 │   │   ├── inputs/
```

```
│   │   └── expected_outputs/
│   ├── Changeplan Transactions/
│   │   ├── inputs/
│   │   └── expected_outputs/
│   ├── Disable Transactions/
│   │   ├── inputs/
│   │   └── expected_outputs/
│   └── Delete Transactions/
│   ├── inputs/
│   └── expected_outputs/

│   └── Paybill Transactions/
│   ├── inputs/
│   └── expected_outputs/

│   └── Logout Transactions/
│   ├── inputs/
│   └── expected_outputs/
```

3. Test Case Organization

Each transaction type has its own directory under the tests folder.

Inside each transaction directory:

- The inputs folder contains all test input files (*.in.txt)
- The expected_outputs folder contains all expected output files (*.expected.transactions.txt)

Each input file has a corresponding expected output file with the same test name and number.

Example:

```
Create_01.in.txt
Create_01.expected.transactions.txt
```

This naming convention ensures a clear one-to-one mapping between inputs and expected outputs.

4. Test Execution Plan

Each test case is executed by redirecting the contents of an input file into the Banking System Front End program.

General command format:

```
banking < inputfile > actual_output.txt
```

Example:

```
banking < Create_01.in.txt > Create_01.actual.transactions.txt
```

5. Output Storage and Organization

Actual output files generated by test execution are stored in the same transaction directory as the test case.

Actual output naming format:

```
<TestName>.actual.transactions.txt
```

Example:

```
Create_01.actual.transactions.txt
```

6. Output Comparison

Actual output files are compared against the corresponding expected output files using a file comparison tool such as diff.

Example:

```
diff Create_01.expected.transactions.txt
Create_01.actual.transactions.txt
```

If no differences are reported, the test is considered passed. If differences are reported, the test is considered failed.

7. Batch Test Execution

Shell scripts may be created to automate running all tests within a transaction directory.

Example script:

```
for f in tests/Create\ Transactions/inputs/*.in.txt
 do
 name=$(basename "$f" .in.txt)
 banking < "$f" > tests/Create\
Transactions/$name.actual.transactions.txt
 done
```

Similar scripts can be created for each transaction type.

8. Reporting

Test results are determined by reviewing diff outputs or script logs. Any failing test cases and observed differences are recorded.

9. Future Regression Testing

This test suite can be reused in later phases to ensure that new changes do not break existing Front End functionality.