

Phase #3 - Front End Requirements Testing

Group: Bushrat Zahan, Menhdi Patel, Sevara Omonova, Nabiha Shah

How To Run UNIX Script

Navigate to directory named tests/scripts

```
bash run_tests.sh
```

```
bash check_tests.sh
```

```
=====
paybill_invalid_company: PASS
=====
paybill_negative_balance: PASS
=====
paybill_over_limit: PASS
=====
paybill_valid: PASS
=====
standard_privileged_reject: PASS
=====
standard_wrong_account: PASS
=====
transaction_after_logout: PASS
=====
transaction_before_login: PASS
=====
transaction_file_format: PASS
=====
transfer_negative_balance: PASS
=====
transfer_nonexistent_account: PASS
=====
transfer_over_limit: PASS
=====
transfer_valid: PASS
=====
whitespace_only_input: PASS
=====
withdrawal_disabled_account: PASS
=====
withdrawal_negative_balance: PASS
=====
withdrawal_over_limit: PASS
=====
withdrawal_valid: PASS
=====
===== TEST SUMMARY =====
Total PASS: 41
Total FAIL: 0
○ sevaraomonova@Sevaras-MacBook-Air scripts %
```

Failure Table

Test Name	Intention (what its testing)	Failed Output (how the output was wrong)	Error (what the code error was)	Fix Applied (how the program/test was changed to fix it)
admin_privileged_allowed	Verify after a successful admin login, privileged transactions are allowed and recorded correctly into output and transaction files.	The create transaction did not match the expected terminal output and transaction file. The generated output differed from the expected results.	Test environment configuration issue, the admin_privileged_allowed.txt referenced a different current accounts file than the one used to generate the expected outputs, resulting in inconsistent behavior during the admin session.	Updated the admin_privileged_allowed.txt to reference the correct current accounts file and regenerated the expected output files to ensure consistency between execution and validation.
admin_name_prompt	To verify that the admin is prompted for the account holder's name and account number before processing a transaction.	The program displayed "Invalid command" and "Invalid amount" in response to valid input instead of prompting for the required fields in the correct order.	The admin session prompts following a transaction request were not properly implemented, causing incorrect input validation.	Updated the TransactionProcessor to ensure all admin sessions correctly prompt first for the Account Holder Name and then for the Account Number, in accordance with the specified instructions.
deposit_valid	To verify that a valid deposit to a valid account for the logged-in user is accepted and recorded properly.	The deposit transaction was rejected and the output did not match the expected results.	Test data error, the account number provided in the input file did not correspond to the specified account holder in the accounts	Corrected the account number in deposit_valid.txt to match the specified account holder in the accounts file, and regenerated expected files.

			file, causing validation to fail.	
create_name_too_long_reject	To verify that account creation is rejected when the account holder's name exceeds 20 characters.	The system allowed an account to be created even though the entered name exceeded the 20-character limit.	A validation check for the length of the account holder's name was not implemented.	Added a conditional statement in the process_create function within the TransactionProcessor file to display an error and reject the transaction if the name length exceeds 20 characters.
end_of_file_handling	To ensure that the END_OF_FILE marker is handled correctly and that only valid account names listed before it can successfully log in.	The system allowed a login attempt using a name that appeared after the END_OF_FILE marker in the current accounts file.	There was no validation to confirm that the entered account holder name exists in the account records during login.	Updated the handle_login function in front_end_app to validate that the entered username exists in the account manager and to display an error message if the account is not found.
Incorrect_input	To verify that invalid inputs (e.g., unrecognized commands, non-numeric values, or negative numbers where not permitted) are properly rejected by the system.	The system did not allow login to proceed even when a valid session type and account holder name were provided. As a result, no commands could be executed because the user was not successfully logged in.	A nested if statement responsible for validating the username in standard mode was incorrectly indented, preventing proper execution of the login validation logic.	Corrected the indentation in the handle_login function within front_end_app so that the username validation check is properly nested inside the standard session condition.
login_invalid_session_type	To verify that the system rejects an invalid session type during login and handles the	After entering invalid session type, subsequent inputs were treated as top-level	Test design mismatch, the test assumed the login process would continue	Updated the test input to restart the login command after an invalid session type and adjusted the expected output to reflect the

	incorrect login input gracefully.	commands, resulting in multiple "Unknown command" errors and outputs that did not match the expected behavior.	prompting for a valid session type, while the implemented design returns to the main command loop after detecting invalid session input.	implemented login control flow behavior.
login_standard_with_name_not_in_system	Verify that standard login is rejected when the user name does not exist, and the program continues safely without creating session output.	Test originally expected an .atf transaction file, but no transaction file was produced because the user never successfully logged in or started a session.	Test design error, the test assumed an .atf file would be generated even though there is no successful session and no logout writing.	Removed the expected .atf file for this test so the test now only validates the terminal output.