

Class Project Group 2

Ahmad Aljawish [100987855], Michael Carinci [100826903],

Patrik Prenga [100990129], Marco Pasqua [100991681]

CSCI 3060U Course Project - Winter 2026

Phase #3 - Failure Tables

Login Test Table:

File Name	Test Intent	Expected Behavior	Error in Code	Actions Taken
Login_02	Invalid login mode.	The system should reject non-standard/non-admin strings and remain logged out.	Missing check for valid mode types	Added Validators.validate_mode check in login_transaction.
Login_03	Standard login with missing name	A standard login attempt with an empty line for the username should trigger an error.	No length check for standard users	Updated login_transaction to reject empty strings via Validators.
Login_06	Entering a username over 20 characters	Entering a username longer than 20 characters to test the Validators logic.	Name validation was too permissive	Enforced strict len(name) <= 20 check in Validators.

Create Transaction Test Table:

Test Name	Test Intent	Expected Behavior	Error in Code	Action Taken
Create_04	Attempting to create a duplicate user	The system should reject the name "User1" with an error and write no transaction record.	Missing uniqueness check against AccountManager.	Added a name-existence check in create_transaction.
Create_05	Creating a user with a name that is too long	The system should reject the name via validator and write no transaction record.	create_transaction bypassed the Validators class.	Implemented Validators.validate_name check before processing.
Create_06	Creating a user with a zero / invalid balance	The system should reject the \$0.00 amount and write no transaction record.	Inconsistent logic between validator and requirement.	Enforced strict validate_amount check in the creation flow.

Withdrawal Test Table:

Test Name	Test Intent	Expected Behavior	Error in Code	Action Taken
Withdrawal _02	Standard user \$500 limit.	The system should reject \$600.00 withdrawal for standard users.	Missing conditional check for session.mode and amount limit.	Added if mode == 'standard' and amount > 500 logic.
Withdrawal _03	Insufficient funds check.	The system should reject withdrawal exceeding the account balance.	withdrawal_transaction was not comparing input to account.balance	Implemented balance verification against AccountManager data.
Withdrawal _04	Admin bypass of \$500 limit.	The system should allow an admin to withdraw \$1000.00.	Limit check was either global or entirely missing.	Ensured limit only applies to standard mode users.
Withdrawal _05	Invalid account number.	The system should reject withdrawal for a non-existent account 99999.	No lookup was performed against the loaded accounts list.	Added existence check in self.account_manager.accounts_by_number.

Transaction Test Table:

Test Name	Test Intent	Expected Behavior	Error in Code	Action Taken
Transfer_02	Check sender balance.	The system should reject the transfer because Jane Smith only has \$50.00.	transfer_transaction was not comparing the amount to the sender's balance.	Added a balance check against the AccountManager record for the source account.
Transfer_03	Validate destination existence.	The system should reject transfer to non-existent account 88888.	The code only verified the source account and ignored the destination validity.	Added a lookup for the destination account number in the accounts dictionary.
Transfer_04	Validate account status.	The system should reject transfer to account 00003 because it is Disabled (D).	The system was only checking if the account existed, not if it was Active (A).	Added a check to ensure account.status == 'A' for both source and destination.
Transfer_05	Enforce a \$1000.00 limit.	Standard users should be blocked from transferring more than \$1000.00.	No session-mode limit was implemented for the transfer function.	Integrated a check: if mode == 'standard' and amount > 1000.00.

Transfer_06	Prevent standard users from transferring from accounts they don't own.	The system should reject the transfer because John Doe does not own account 00002.	transfer_transaction did not verify if the from_account belonged to the current user in standard mode.	Added a check: if mode == 'standard' and from_acc.holder != session.user.
--------------------	--	--	--	---

Pay Bill Test Table:

Test Name	Test Intent	Expected Behavior	Error in Code	Action Taken
Paybill_01	Standard success for John Doe.	The system accepts the login and processes the \$50.00 payment.	Invalid transaction code error previously appeared because the name was partially left in the buffer.	Used readline().strip() to ensure the full name "John Doe" is consumed during login.
Paybill_02	Validate company code.	The system should reject the payment because "XY" is not a valid utility company.	The code originally lacked a validation check for the utility company input.	Added a check against valid_companies = ["EC", "CQ", "TV"].
Paybill_03	Validate balance for Jane Smith.	The system rejects payment due to insufficient funds without crashing.	AttributeError: 'Session' object has no attribute 'user' occurred during ownership comparison.	Updated session.py to include the user attribute and store the username during login.

Paybill_04	Verify account ownership.	John Doe (standard) is blocked from paying bills via Jane Smith's account.	Program crashed at line 300 when trying to access acc.holder before confirming account existence.	Added an early return after the account existence check to prevent logic from reaching the crash point.
Paybill_05	Validate account existence.	The system rejects payment for non-existent account number 88888.	The system was not verifying if the account number existed in the master file.	Added a lookup check in self.account_manager.accounts_by_number.
Paybill_06	Enforce a \$1000 limit.	Standard users should be blocked from paying a bill over \$1000.00.	No logic was present to distinguish limits between standard and admin modes.	Integrated a conditional check: if self.session.mode == "standard" and amount > 1000.00.

Deposit Test Table:

Test Name	Test Intent	Expected Behavior	Error in Code	Action Taken
Deposit_01	Standard success for John Doe.	System records the \$200.00 deposit into account 00001.	Program crashed at line 354 during the ownership check despite the user being logged in.	Synchronized naming by updating session.py to use self.user instead of current_user.
Deposit_02	Validate account existence.	The system should reject deposits into non-existent account labelled 88888.	The code would crash if it attempted to access attributes of a non-existent account object.	Ensured the if account_num not in ... block uses an early return to stop execution.
Deposit_03	Standard user deposit limit.	John Doe (Standard) should be blocked from depositing \$1000.01.	The \$1000 limit was not being enforced based on the session mode.	Integrated a conditional check if self.session.mode == "standard" and amount > 1000.00.
Deposit_04	Admin limit bypass.	Admin should be allowed to deposit \$5000.00 successfully.	A global limit would have incorrectly blocked this administrative transaction.	Applied the deposit limit check only to sessions where mode == "standard".

Deposit_05	Deposit into disabled account.	The system should reject deposit into Alex Lee's account (00003/Status D).	The initial logic did not check the account status (A vs D) before recording.	Added a check for acc.status != 'A' to prevent transactions on disabled accounts.
Deposit_06	Unauthorized standard deposit.	John Doe (standard) should be blocked from depositing into Jane Smith's account.	Standard users were previously allowed to deposit into any account number provided.	Added ownership verification: if self.session.mode == "standard" and acc.holder != self.session.user.

Disable Tests:

Test Name	Test Intent	Expected Behavior	Error in Code	Action Taken
Disable_03	Validate account existence.	System rejects disabling for non-existent account 88888.	The code recorded the transaction without checking the master account list.	Added a lookup check in accounts_by_number with an early return.
Disable_04	Verify name/number match.	Reject if name "Jane Smith" doesn't match account 00001 owner.	Code accepted the request because it didn't compare input name to the record.	Added an if acc.holder != name validation block before recording.
Disable_06	Status validation.	Reject if the account (Alex Lee) is already marked 'D'.	The system allowed redundant "disable" commands on already inactive accounts.	Added a check for acc.status == 'A' to ensure only active accounts are disabled.

Delete Tests:

Test Name	Test Intent	Expected Behavior	Error in Code	Action Taken
Delete_03	Validate account existence.	System rejects deletion for non-existent account 88888.	The code recorded the transaction without checking if the account was in the master file.	Added an existence check with an early return to stop the transaction record.
Delete_04	Verify name/number match.	Reject if name "Jane Smith" doesn't match account 00001 owner.	Code accepted the delete request because it didn't compare the input name to acc.holder.	Added validation: if acc.holder != name before calling the transaction manager.
Delete_06	Status validation.	Reject deletion if the account is already marked as 'D' (Alex Lee).	The system allowed redundant deletions on accounts that were not active.	Added check for acc.status == 'A' to ensure only active accounts are deleted.

Changeplan Tests:

Test Name	Test Intent	Expected Behavior	Error in Code	Action Taken
ChangePlan_03	Validate account existence.	Reject change for non-existent account 88888.	Code lacked a lookup in accounts_by_number, allowing "88888" to be recorded.	Added an existence check with an early return in changeplan_transaction.
ChangePlan_04	Verify name/number match.	Reject if name "Jane Smith" doesn't match account 00001 owner.	Code recorded the transaction without verifying if the input name matched the record.	Added validation: if acc.holder != name before recording the transaction.
ChangePlan_06	Status validation.	Reject change for disabled account 00003 (Alex Lee).	The system ignored the status attribute, allowing changes to inactive accounts.	Added check for acc.status == 'A' before allowing the plan change.