Phase 5 Testing Document

CSCI 3060U

Dr. Michael Miljanovic

Levi Willms, Daniel Hinbest, Syed Rizvi, Raje Singh

No other component of the project needs to be compiled or run to test the backend, just make sure python3 and pytest are installed on your system. With any of the methods below, make sure you are running the commands from the root directory `(csci3060/`, or if you renamed the directory where the Makefile is)..

Recommended Method

To run the backend tests, use `make test_backend`. Running this command runs all the tests on the backend. If you wish to test only one module and not the entire suite, you can also do `make test_backend [module_name]` where `module_name` is the name of the module you want to test (e.g. `make test_backend available_games` runs the tests for available_games only). This assumes `pytest` and `python3` are installed.

Alternate

You can also just run `pytest` if it is on your PATH, and if it is not do `python3 -m pytest`. **Only do this if you wish to pass flags to pytest. Example:** with `pytest-cov` installed, passing the --cov flag will generate a table with statement coverage information.

Phase 5 Statement Coverage

Assume usernames and any string values have a minimum of 5 characters.

File	Test Method	Backend Method	Test Results	Fixes Required
user.py	test_get_username	get_username	PASS	
user.py	test_get_user_type	get_user_type	PASS	

File	Test Method	Backend Method	Test Results	Fixes Required
user.py	test_get_credit	get_credit	PASS	
user.py	test_add_credit	add_credit	PASS	
user.py	test_remove_credit	remove_credit	PASS	
user.py	test_set_username	set_username	PASS	
user.py	test_set_user_type	set_user_type	PASS	
user.py	test_to_string	to_string	PASS	
transaction.py	test_handle_create_user_does_not_exist	handle_create	FAIL	Mistake in asserting a string during test that did not match with expected string. Both now match.
transaction.py	test_handle_create_user_already_exists	handle_create	PASS	
game.py	test_get_game	get_game	PASS	
game.py	test_get_seller	get_seller	PASS	
game.py	test_get_price	get_price	PASS	
game.py	test_set_name	set_name	PASS	
game.py	test_set_seller	set_seller	PASS	
game.py	test_set_price	set_price	PASS	

File	Test Method	Backend Method	Test Results	Fixes Required
game.py	test_to_string	to_string	PASS	
available_games.py	test_write	write	FAIL	Test was not recording the correct data. After providing valid data, the formatting in the test was incorrect. The write function was also passing one too many digits int the format. Fixed after this was lowered. PASSED
available_games.py	test_read	read	FAIL	Logic for string separation was not correct, so fixed that in the test_available_games_file. The test itself was looking for incorrect game name formats, so adjusting that allows the test to pass. PASSED.
transaction.py	test_handle_delete	handle_delete	FAIL	The log file could not be found because of how the file's path was declared. Fixed by changing the current working directory to that of the test file. PASSED
transaction.py	test_handle_buy	handle_delete	FAIL	Was trying to access get_available_cr

File	Test Method	Backend Method	Test Results	Fixes Required
				edit() when the correct function was get_credit(). Changed to correct method call. PASSED
file_utils.py	test_change_to_file_dir_existing	change_to_file_dir	PASS	
file_utils.py	test_change_to_file_dir_non_existing	change_to_file_dir	PASS	
collection.py	test_init	init	PASS	
collection.py	test_get_game_name	get_game_name	PASS	
collection.py	test_get_owner	get_owner	PASS	
collection.py	test_set_game_name	set_game_name	PASS	
collection.py	test_set_owner	set_owner	PASS	
collection.py	test_to_string	to_string	PASS	
test_user_accounts.py	test_read	read	PASS	
test_user_accounts.py	test_write	write	PASS	
test_user_accounts.py	test_read_failure	read	PASS	
test_user_accounts.py	test_write_failure	write	PASS	
test_game_collection	test_read	read	PASS	
test_game_collection	test_write	write	PASS	
test_game_collection	test_read_failure	read	PASS	

File	Test Method	Backend Method	Test Results	Fixes Required
test_game_collection	test_write_failure	write	PASS	
test_main.py	test_fileParser	fileParser	FAIL	AttributeError: 'str' object has no attribute 'write' File parser was not receiving a file from the testing suite but the string of the filename. The test was changed to pass the file object to the file parser function. Test now PASSED.
test_main.py	test_main	main	FAIL	Main was searching for an existing file. Update was made to create the file if it does not exist.
test_main.py	test_main	main	FAIL	Could not find log file path. Changed to correct one in the test file. PASSED
test_daily_transaction.py	test_read	read	FAIL	Assertion error on file length
test_daily_transaction.py	test_write	write	FAIL	Value error on file name

Phase 5 Path Coverage

The handle_delete() transaction in the transaction.py file was chosen for path coverage. This is because it has loops, if statements, and multiple dependencies that make a optimal demonstration for displaying path coverage.

transaction.py	test_handle_delete_user_has_games	handle_delete	FAIL	fixed issue with referencing delete transaction methods directly, modified code to call the handle_transaction method that then navigates to the handle_delete method. PASSED.
transaction.py	test_handle_delete_user_does_not_exist	handle_delete	FAIL	Fixed issue where file was not being read correctly in tests. File was already opened and did not need to be reopened in read mode. PASSED.
transaction.py	test_handle_delete_user_has_game_collection	handle_delete	FAIL	Fixed issue where file was not being read correctly in tests. File was already opened and did not need to be reopened in read mode. PASSED.
transaction.py	test_handle_delete_user_has_games	handle_delete	FAIL	Fixed issue where file was not being read

				correctly in tests. File was already opened and did not need to be reopened in read mode. PASSED.
transaction.py	test_handle_delete_user_exists	handle_delete	FAIL	Issue with reading underscores in username. Removed underscores from test data. Also fixed issue with referencing delete transaction methods directly, modified code to call the handle_transaction method that then navigates to the handle_delete method. PASSED.
transaction.py	test_handle_delete_user_has_no_games	handle_delete	FAIL	fixed issue with referencing delete transaction methods directly, modified code to call the handle_transaction method that then navigates to the handle_delete method. PASSED.
transaction.py	test_handle_delete_user_exists_games_in_one_map_only	handle_delete	PASS	

transaction.py	test_handle_delete_nonexistent_user_with_no_games_or_collection	handle_delete	PASS	
transaction.py	test_handle_delete_nonexistent_user_with_games_no_collection	handle_delete	FAIL	Assertions were checking for values in, when they should have been checking that the values were not in the map. PASSED
transaction.py	test_handle_delete_nonexistent_user_with_collection_no_games	handle_delete	FAIL	Assertions were checking for values in, when they should have been checking that the values were not in the map. PASSED
transaction.py	test_handle_delete_nonexistent_user_with_games_and_collection	handle_delete	FAIL	Assertions were checking for values in, when they should have been checking that the values were not in the map. PASSED
transaction.py	test_handle_sell_game_already_exists	handle_sell	PASS	
transaction.py	test_handle_sell_new_game	handle_sell	PASS	
transaction.py	test_handle_sell_invalid_price	handle_sell	PASS	
transaction.py	test_handle_sell_empty_game_name	handle_sell	PASS	
transaction.py	test_handle_buy	handle_buy	PASS	

	_game_does_not_exist			
transaction.py	test_handle_buy_buyer_does_not_exist	handle_buy	PASS	
transaction.py	test_handle_buy_seller_does_not_exist	handle_buy	PASS	
transaction.py	test_handle_buy_not_enough_credit	handle_buy	PASS	
transaction.py	test_handle_buy_successful	handle_buy	PASS	