Unit testing of the login and logout functions

# Login:

The login function prompts the user to enter a username and a password, then checks the information against a json of users. If the information matches, the user is returned. If the user exists but the password is incorrect, the function displays an error message and returns None. If the user doesn’t exist, the user is asked if he would like to register. If not, the function returns None. If yes, the user is prompted to input a password. If the password has at least one capital letter, one numeral, one special character and is of length 8 or more, a new user is added to the json and the function returns the user object (username and wallet: 0). If not, the function returns None.

Given that the login function receives no parameters, but only direct user input from the terminal, the “input” for our tests will consist of a list of keyboard inputs, all of type string. We will then test the function on the inputs’ values, not types, and on the different decisions a user can make.

In order to test the function, 3 stubs were also created: write\_to\_file\_stub, check\_password\_stub\_incorrect, check\_password\_stub\_correct.

Given that the input domain is not consistent, depending on the choices the user makes, we have tested each possible decision in the decision tree. These can be the Equivalence Classes:

1. Login existing user with correct password
2. Login existing user with incorrect password
3. Login new user without registering
4. Login new user without registering
5. Register new user with valid password
6. Register new user with invalid password

We also tested an extra function we added: check\_password. It checks the password entered by the user based on 4 criteria: the password has at least one capital letter, one numeral, one special character and is of length 8 or more. Therefore, the equivalence classes:

1. Check password with no capital letter
2. Check password with no numeral
3. Check password with length < 8
4. Check password with no special character
5. Check valid password

For each EC, we have tested 1 valid input. There is to note that EC 1-4 for check\_password have valid inputs, but are asserted to fail because of given conditions for the validity of the password.

All test cases data can be found in Login\_Logout\_Testing\_Document. It may be redundant to write them here because they are all customized for each action an user can take.

Logout:

The logout function receives a parameter of type Cart. It then checks if the cart has any items. If not, it returns True. If it does, it prints the items and asks for confirmation on the Logout

The logout function receives one parameter: a Cart instance. Given this fact, we were able to test invalid inputs on one side, and valid inputs with different scenarios on the other side

Therefore, the equivalence classes:

1. Logout function test with a string input
2. Logout function test with an integer input
3. Logout function test with a float input
4. Logout function test with a list input
5. Logout function test with an empty input
6. Logout when the cart is empty
7. Logout with a non-empty cart and no confirmation
8. Logout with a non-empty cart and confirmation

For EC 1-4, the function should throw an AttributeError

For EC 5, the function throws a TypeError

For the rest of the EC, we test 1 valid input. All test cases data can be found in Login\_Logout\_Testing\_Document.