Milestone 5

Team number: 104-3

Team name: Orange

Team Members: Konlan Rondini, Nicole Costello, Dylan Sain, Zhongzhi Zhang, Liyang Ru

Process for Testing:

When we are done with each feature, we will use various test cases inside the HTML and JS. Each test case will have one thing that is incorrect about it, and if we are able to correctly throw the error and tell the user, the test case is successful. We will also have a test case with everything valid in order to make sure that our program works as intended. If in the case that we have a test case that doesn't do what was intended, we will have to look at what is does do instead and use that information to fix the code.

Feature 1: Login Page

Important functionality of this feature:

- User not allowed to continue without correct username and password combination.
    - A user must input a username and password that are both inside the database, and are connected to each other. You cannot login using one user's username and another user's password
    - Example tests:
        - Example SQL query to validate login: (username = valid, password = valid) valid login
            - If the user inputs a valid username and password, they will be allowed to continue to edit their account.
        - Example SQL query to validate login: (username = valid, password = invalid) invalid login
            - If the user inputs a valid username, but password not found in the SQL database, they will be asked to re-enter a valid one.
        - Example SQL query to validate login: (username = invalid, password = valid) invalid login
            - If the user inputs a username not found in the SQL database, they will be asked to re-enter a valid one.
- User cannot sign up for account using a username that is already taken.
    - Test Case:
        - If the user enters a username which is found in the SQL database, they will be told that their selected username is already taken and will be prompted to enter a new one.
            - SQL query to compare entered username with usernames in database
                - Suppose "Krondini" already exists in database.
                    - Valid username: Krondini1
                    - Invalid username: Krondini

        - If the user enters a username which is not already in the SQL database, they will be allowed to create their account.
- User must sign up using a valid email.
    - If the user enters an email which has the form of ___@___.__ they will be allowed to continue with that email address (accomplished via regex)
    - If the user enters an email which does not follow the above form, they will be asked to re-enter their email address.
    - Test cases:
        - Valid email: someone@email.com
        - Invalid email: someone@someonelse@email.com
        - Invalid email: someoneemail.com
        - Invalid email: @email.com
        - Invalid email: someone@email
- User must fill in ALL information when signing up. (First name, last name, username, password, email)
    - If there are any areas left blank when the user clicks "Submit" they will be told to fill in the missing information
    - If all areas are properly filled, then the user will be allowed to continue and create their account.
    - Test Cases:
        - Example SQL query to validate login: (username = valid, password =)
        - Example SQL query to validate login: (username = , password = valid)

- When signing up the password and username must inside the length limit.
    - If the password is too short or the user name is too long the page will refuse the registration
    - Test Cases:
        - Example SQL query to validate login: (username = valid, password = aaaaaaaaaaaaaaaaaaaaaaaaaaaa)
        - Example SQL query to validate login: (username = aaaaaaaaaaaaaaaaaaaaaaaaaaaa, password = valid)

Feature 2: Net Worth Entry Page

Important functionality of this feature:

- User input is of correct data type (bool, int, string).
    - Test cases:
        - SQL addition of data: (name = house, value = 5, type = asset, category = property)
            - valid addition
        - SQL addition of data: (name = house, value = A, type = asset, category = property)

            - invalid addition (no letters)
        - SQL addition of data: (name = house, value = -5, type = liability, category = property)
            - invalid addition (no negative)
- No empty fields when an asset or liability is added by user (Name, Type, Value, Category)
    - Test cases:
        - SQL addition of data: (name = house, value = 5, type = asset, category = property)
            - valid addition
        - SQL addition of data: (name = house, value = , type = asset, category = property)
            - invalid addition (value is empty)
        - SQL addition of data: (name = house, value = 5 ,type = , category = property)
            - invalid addition (type not specified)
        - SQL addition of data: (name = , value = 5, type = liability, category = property)
            - invalid addition (name is empty)
        - SQL addition of data: (name = house, value = 5, type = liability, category =)
            - invalid addition (category not specified)
- Double entries (i.e. 2 houses should be disallowed)
    - Test cases:
        - Have the user enter name = "Toyota Prius", category = "Vehicle" two times
            - If working correctly, querying the database will reveal there are 2 of these combinations and the user will be disallowed from making entry
        - SQL addition of data: (name = house, value = 5, type = liability, category = property)
            - invalid addition (two 'house' already exists in category of property)
- Access page from profile page via a button
    - Test cases:
        - User (on home profile page) presses "Add Net Worth" button. If working, the user should be redirected to the net worth page.
- Is all of the current data visible
    - Test cases:
        - User navigates to Net Worth page
            - If the user has entered items previously, they should see their running totals for assets, liabilities, and net worth at the bottom of the page in a table. Otherwise, they should see these values as '0' or blank.
        - User adds entries
            - When a user adds assets and liabilities, these changes should be reflected in the table at the bottom of the page that contains total assets, liabilities, and net worth.

Feature 3: Edit Page:

Important functionality of this feature:

- Ability to delete/modify entries

    - Test Cases:
        - If the user changes the value of an item to “0”, the item will be removed from the table
            - Ex: “Name: ‘Toyota Prius’, Type: ‘Vehicle’, Value: ‘0’”
                - The above example will be removed from the database
        - User changes their house value from $100,000 to $110,000
                - This change should be reflected in the database.
- Changes stay when you revisit change (are they correctly being made in the database)
    - Test Case:
        - If the user makes any edits to their table of entries, then leaves the page. The changes should remain upon reopening the page
            - Test Case:
            - Ex: “Name: ‘Toyota Prius’, Type: ‘Vehicle’, Value: ‘5000’”
                - If the user adds the above item, leaves the page, then reopens the page, the item should remain in their table of entries
- Access the page from the user profile page via a button on the profile page
    - Test Case:
        - Click “Edit”
            - When the user is on the home profile page, they should be able to click a button to navigate to the edit page.
- All of the data is visible
    - Test Case:
        - Navigate to Edit page
            - If working properly, navigating to this page should cause our program to query our SQL database and display all of the assets and liabilities that have been added by the user (including their descriptions/category).
            - If working properly, the new data table of assets and liabilities should be the value that they inputted
- Website is readable
    - Test Case:
        - Do all HTML pages load properly on a user’s machine?
            - If there are any attributes overlapping when a page is opened, then the page did not opened properly