Project: PRAW

Members: Omar Alhinnawi, Timothy Lee

Introduction

For the sake of making everyone's lives easier, we put all our testing work into folders titled as "*pull_request_x*". There exist five of these inside the current master branch in the group module. These contain the entirety of our testing and will be the things we refer to prove our testing methods.

For this project, there was already thorough unit testing, so our focus was mainly geared towards the automated testing, with that being said, some unit testing was done for the sake of completeness/formality.

Unit Testing

- **"Use the search function to search for the top subreddits/posts for prespecified subreddits and visually seeing if the results match what shows up on the subreddit's page."**
    - Refer to pull_request_3/read_post_testing.py lines 47-51 there is a function titled "testAutomatedReadSubmission", this function searches and determines if posts exist inside a subreddit (originally had print statements for us to visually confirm but forgot to re-write that in after pushing).

- **"Pre-selecting posts and retrieving comment trees from these posts."**
    - Refer to pull_request_3/read_post_testing.py lines 33-46, there is a function titled "testReadSubmission", this function is what was used for our pre-selected posts and printing/assertions.
- **"Making manual post and seeing if the module can find it"**
    - Refer to pull_request_1/submission_unit_test.py, lines 50-76 are a series of manual submissions which we posted to our subreddit. For seeing if our module could find them, refer to: Refer to pull_request_3/read_post_testing.py lines 47-51 (originally had print statements for us to visually confirm but forgot to re-write that in after pushing).
- **"Attempt to authorize without proper credentials/with proper credentials."**

    - With proper credentials: Refer to pull_request_1/submission_unit_test.py, inside there is a function title testLogin
    - Without proper credentials: Refer to pull_request_5/invalid_login_test.py, in this file, we essentially make an attempt to login without the proper client credentials which will result in a failed login exception.

Automated Testing

- **“we will automatically generate posts with comments containing set texts into a private subreddits”**
    - We were able to test this portion of our proposal by generating random titles and bodies and using the praw functions to post a random amount of posts from (1 to 10) to a private subreddit.
        - The way we generated random titles and bodies were by concatenating words from an online word dictionary.
        - Link to code: https://github.com/csun-comp587-s20/praw/blob/master/pull_request_2/automated_submission_testing.py
            - The `def randomTitleGenerator(self):` and `def randomBodyGenerator(self):` are methods used to concat a random amount of words from the online dictionary to form a string (title, body).
            - `def testMultiplePostings(self):` is the main method to test multiple postings.

- **“we will use the search functionality of the module to search for our previously written posts and comments”**
    - Refer to pull_request_5/broad_search_testing.py, lines 31-65.
    - In there we used a function titled BroadSearch to test the search for our subreddit.
- **Property Test: 2 request per second limit:**
    - Refer to pull_request_3/test_rate_limiting.py, lines 33-44, in here, we test to see if the module can request below the 2 requests per second limit.
- **Submission/Comments Properties Testing**
    - Refer to the pull_request_4 folder, this contains all our tests for the basic functions/properties of submissions/comments.

Lessons Learned

- Going into a bigger project and testing specific components instead of just picking random parts is almost always the best strategy. Good planning early on is worth its weight in gold during testing, especially with bigger projects. For our testing, it didn't feel misguided because we knew what we were planning to do very specifically from the start.
- We probably should've come up with a more cohesive plan for better validation, our assertions are hardcoded (for the most part) which just makes the system harder to refactor because that means we have to write new tests from scratch instead of taking what wrote before and altering it to fit the situations.
- Overall I found that testing (unit, automated) are two important tools to ensure that your code is in the right shape for production. Testing a library such as PRAW (python reddit api wrapper), was a great way to see how testing comes in handy. Tests can find bugs and other deficiencies that you may not find from not testing your code.