System and Unit Test Report
Social Stock Analyzer
03/04/2022

1. **<u>Backend Testing</u>**
   Library: PyTest
   Type: Automatic(Run every time system is built)

   Unit Tests:
   1. User Authentication Module -
      - Sprint 1: As a user, I want to be able to create an account and log in and out of it so I can maintain a portfolio of the stocks of my choosing without having to create a new one each time.
      - Sprint 2: As a user, I should be able to change my personal information including my password, email address, and name.

      a. test_add_user
         i. Description: verifies that a user can register successfully given valid information.
         ii. Input fields: first name, last name, valid email, username, password, repeat password
         iii. Expected system output: "Account creation successful" with HTTP code 201

      b. test_add_user_weak_password
         i. Description: verifies that the system gives an error when a user tries to sign up using a weak password
         ii. Input fields: first name, last name, valid email, username, weak password, repeat weak password
         iii. Expected system output: "Password not strong enough" with HTTP code 400

      c. test_add_user_invalid_email
         i. Description: verifies that the system gives an error when user tries to sign up using an invalid email(i.e with invalid domain or format)
         ii. Input fields: first name, last name, invalid email, username, weak password, repeat weak password
         iii. Expected system output: "Invalid email." with HTTP code 400

      d. test_add_user_missing_field
         i. Description: verifies that the system gives an error when user tries to sign up without entering one or more required fields
         ii. Input fields: first name, last name, email, blank username, weak password, repeat weak password

iii. Expected system output: "Username is required." with HTTP code 400

e. test_add_user_already_registered
   i. Description: verifies that the system gives an error when user tries to sign up using an email that's already registered
   ii. Input fields: first name, last name, registered email, blank username, weak password, repeat weak password
   iii. Expected system output: "User is already registered." with HTTP code 400
f. Test_authenticate_user
   i. Description: verifies that a user can successfully login with valid credentials
   ii. Input fields: username, password
   iii. Expected system output: <json token>
g. test_authenticate_user_bad_password
   i. Description: verifies that the system gives an error when the user tries logging in with an invalid password
   ii. Input fields: username, invalid password
   iii. Expected system output: "Invalid password" with HTTP code 401
h. test_authenticate_user_bad_username
   i. Description: verifies that the system gives an error when the user tries logging in with an invalid username
   ii. Input fields: invalid username, password
   iii. Expected system output: "Invalid username" with HTTP code 401
i. Test_logout
   i. Description: verifies that a user can log out successfully when in a valid session.
   ii. Input fields: N/A
   iii. Expected system output: HTTP code 200
j. Test_change_password
   i. Description: verifies that a user can change their password successfully when a new valid password is provided
   ii. Input fields: current password, new password, repeat new password
   iii. Expected system output: HTTP code 200
k. Test_change_password_incorrect_password
   i. Description: verifies that the system gives an error when the current password provided by the user is invalid
   ii. Input fields: invalid current password, new password, repeated new password

        iii. Expected system output: HTTP code 401
    l. test_change_password_mismatched
        i. Description: verifies that the system gives an error when the new password is entered incorrectly in the repeat password field
        ii. Input fields: current password, new password, incorrect repeated new password
        iii. Expected system output: HTTP code 400
    m. test_change_password_weak
        i. Description: verifies that the system gives an error when the new password entered by the user is weak
        ii. Input fields: current password, weak new password, weak repeated new password
        iii. Expected system output: HTTP code 400
    n. test_change_account_details
        i. Description: verifies that the user account information is changed successfully
        ii. Input fields: first name, last name, username
        iii. Expected system output: HTTP code 200
    o. Test_delete_user_account
        i. Description: verifies that the user can successfully delete their account.
        ii. Input fields: N/A
        iii. Expected system output: HTTP code 200

2. Social Interaction Module -
    - Sprint 3: As a user I should be able to like and comment on a stock to voice my preferences/predictions so that I can engage with other users.
    - Sprint 3: As a user, when I search up a stock I should be able to add it to my portfolio and see the social engagement on that stock.(5)
    a. test_add_like
        i. Description: verifies that the user can like a stock
        ii. Input fields: stock ticker
        iii. Expected system output: HTTP code 200
    b. test_add_like_like_twice
        i. Description: verifies that the system gives an error when a duplicate like is entered for a user
        ii. Input fields: stock ticker
        iii. Expected system output: HTTP code 500
    c. test_remove_like

    i. Description: verifies that the user can remove their like from a particular stock
    ii. Input fields: stock ticker
    iii. Expected system output: HTTP code 200
d. test_liked
    i. Description: verifies that the system returns the number of likes on a particular stock when requested.
    ii. Input fields: stock ticker
    iii. Expected system output: HTTP code 200 and number of likes
e. test_user_likes
    i. Description: verifies that the system returns the stocks liked by the user when requested.
    ii. Input fields: N/A
    iii. Expected system output: HTTP code 200 and list of liked stocks
f. test_total_likes
    i. Description: verifies that the system returns the total likes on a particular stock when requested.
    ii. Input fields: stock ticker
    iii. Expected system output: HTTP code 200 and number of likes
g. test_all_likes
    i. Description: verifies that the system returns the users that liked a particular stock when requested.
    ii. Input fields: stock ticker
    iii. Expected system output: HTTP code 200 and list of users
h. test_add_comment
    i. Description: verifies that the user can comment on a stock
    ii. Input fields: stock ticker, comment body
    iii. Expected system output: HTTP code 200
i. test_user_comments
    i. Description: verifies that the system returns all the comments on a particular stock made by the user.
    ii. Input fields: stock ticker
    iii. Expected system output: HTTP code 200
j. test_fetch_latest_comments
    i. Description: verifies that the system returns the top 5 recent comments on a particular stock
    ii. Input fields: stock ticker
    iii. Expected system output: HTTP code 200
k. test_all_comments

i. Description: verifies that the system returns all the comments on a particular stock
ii. Input fields: stock ticker
iii. Expected system output: HTTP code 200

3. Portfolio Interaction Module
   - Sprint 2: As a user I should be able to add a searched stock to my portfolio/watch list so that I can monitor the progress of my investments. (9)
   - Sprint 3: As a user, I should be able to add and remove stocks from my portfolio as I choose so I can maintain the portfolio of my choice.
   - Sprint 3: As a user, when I search up a stock I should be able to add it to my portfolio and see the social engagement on that stock.(5)

   a. Test_add_stock
      i. Description: verifies that the system is able to add a new stock to the user's portfolio.
      ii. Input Fields: stock ticker
      iii. Expected system output: HTTP code 200
   b. test_remove_stock
      i. Description: verifies that the system is able to remove a stock from the user's portfolio.
      ii. Input Fields: stock ticker
      iii. Expected system output: HTTP code 200
   c. Test_remove_nonexistent_stock
      i. Description: verifies if the system can handle when it is asked to remove a stock that doesn't exist.
      ii. Input Fields: stock ticker
      iii. Expected system output: HTTP code 404
   d. Test_buy_stock
      i. Description: verifies that the system is able to buy a stock, given the user already owns some shares of the stock.
      ii. Input Fields: stock ticker, amount invested, number of shares
      iii. Expected system output: HTTP code 200
   e. Test_buy_negative_stock
      i. Description: verifies that the system is able to react to a negative amount of stocks or shares being bought.
      ii. Input Fields: stock ticker, amount invested, number of shares
      iii. Expected system output: HTTP code 400
   f. Test_sell_stock
      i. Description: verifies that the system is able to sell a stock from the user's portfolio.

ii. Input fields: stock ticker, amount invested, number of shares
iii. Expected system output: HTTP code 200

4. Twitter module
   - Sprint 4: As a user, I want to see the popularity of a stock on widely used apps like Twitter.
   a. Test_getTweet
      i. Description: verifies if twitter code returns a dataframe
      ii. Input fields: query- stock name
      iii. Expected system output: HTTP code 200

2. <u>Front End Testing</u>
Type: Manual

a. Sprint 1: As a user, I want to be able to create an account and log in and out of it so I can maintain a portfolio of the stocks of my choosing without having to create a new one each time.
   i. Scenario
      1. Start application using npm start
      2. Click the "Create New Account" button, page will direct to registration page
         a. username = <unitTest.user>
         b. first name = <unitTest>
         c. last name = <user>
         d. password = <Uniuse@12345>
         e. verify password = <Uniuse@12345>
      3. Click the "Register" button and the page should be directed to the login page.
         a. username = <unitTest.user>
         b. password = <Uniuse@12345>
      4. Click "Log In" button, should be directed to the home page with the dashboard displayed
      5. Click the profile icon button on the top right corner and a menu will drop down.
      6. Click "Log Out" and the page should be directed to the login page.

b. Sprint 1: As a user, I want a home page where I can access all the applications' features in one place.
c. Sprint 2: As a user, I would like an intuitive flow between the different functionalities with easy access to each separate feature.

d. Sprint 4: As a user, I want to interact and navigate throughout the dashboard without having difficulty.
    i. Scenario
        1. Start application using npm start
        2. Login page will appear in the browser
            a. username = <john.doe>
            b. password = <John@12345>
            c. Click “Log In” button, should be directed to the home page with the dashboard displayed
        3. Click the “Portfolio” button in the side drawer and the portfolio component should appear including a graph, top performer, and list of owned stocks.
        4. Click the Market Filter button in the side drawer and an empty table with the “Filter Options” button should appear.
        5. Click the “Dashboard” button in the side drawer and the initially displayed dashboard with related news and portfolio information should appear.
        6. Click the profile icon button and a menu should appear
        7. Click the “Profile” button on the menu and the user’s likes should be displayed.
        8. Click the profile icon button icon and click the “Settings” button on the menu and all the user information should be displayed with an option to edit and change password.
        9. Click the profile icon button icon and click the “Log Out” button on the menu and the page should direct to the login page.
e. Sprint 2: As a user, I should be able to look up a stock and view its price trend and basic financial information so that I can make a decision about investment.
f. Sprint 3: As a user, when I search up a stock I should be able to add it to my portfolio and see the social engagement on that stock.
    i. Scenario
        1. Start application using npm start
        2. Login page will appear in browser
            a. username = <john.doe>
            b. password = <John@12345>
            c. Click “Log In” button, should be directed to the home page with the dashboard displayed
        3. Click the left side of the search field on the top bar of the home page.
        4. Enter “AAPL” into the field.
        5. Click the search icon button on the left.

6. A chart should be displayed showing “AAPL”’s progress over the last six years that can be zoomed in on.
7. The related news, tweets, as well as likes and comments for “AAPL” should be displayed.

g. Sprint 2: As a user, I should be able to change my personal information including my password, username, and name. (5)
    i. Scenario
        1. Start application using npm start
        2. Log in page will appear in browser
            a. username = <anna.g>
            b. password = <Anna@12345>
            c. Click “Log In” button, which should be directed to the home page with the dashboard displayed
        3. Click the profile menu icon in the top right corner and a menu will appear.
        4. Click “Settings” on that menu and account details should appear.
        5. Click edit
            a. First name =<Annaa>
            b. Last name =<Georgee>
            c. Username = <anna.g>
        6. Click “Submit”
        7. Account settings should display new details.
        8. Click “Change Password” and a dialog will appear prompting password details.
            a. Current password = <Anna@12345>
            b. New password = <Anna@123456>
            c. Repeat new password = <Anna@123456>
        9. Click submit, password will be updated.
h. Sprint 2: As a user, I should be able to add a searched stock to my portfolio/watch list If I want so that I can monitor the progress.
i. Sprint 3: As a user, when I search up a stock I should be able to add it to my portfolio and see the social engagement on that stock.
    i. Scenario
        1. Start application using npm start
        2. Login page will appear in browser
            a. username = <anna.g>
            b. password = <Anna@12345>
            c. Click “Log In” button, which should be directed to the home page with the dashboard displayed

3. Click the left side of the search field on the top bar of the home page.
4. Enter “AAPL” into the field.
5. Click the search icon button on the left.
6. A page displaying “AAPL”’s details should appear
7. Click on the “+” icon button and a dialog should appear.
    a. Stock Name = <AAPL>
    b. Amount Invested = <300>
    c. Shares = <2>
8. Click the “Add” button.
9. Click the “Portfolio” button in the left side drawer and the table should include AAPL with the correct amount invested and shares.

j. Sprint 3: As a user, I should be able to add and remove stocks from my portfolio as I choose so I can maintain the portfolio of my choice.
    i. Scenario
        1. Start application using npm start
        2. Login page will appear in browser
            a. username = <anna.g>
            b. password = <Anna@12345>
            c. Click “Log In” button, should be directed to the home page with the dashboard displayed
        3. Click the “Portfolio” button on the left side drawer.
        4. Click the “+” icon button above the table and a dialog should appear.
            a. Stock Name = <FB>
            b. Amount Invested = <400>
            c. Shares = <3>
        5. Click the “Add” button.
        6. The dialog will disappear and the table should include FB with the correct amount invested and shares.
k. Sprint 3: As a user, I should be able to like and comment on stock to voice my preferences/predictions so that I can engage with other users.
    i. Scenario
        1. Start application using npm start
        2. Login page will appear in browser
            a. username = <anna.g>
            b. password = <Anna@12345>
            c. Click “Log In” button, should be directed to the home page with the dashboard displayed

3. Click the left side of the search field on the top bar of the home page.
4. Enter “GOOGL” into the field.
5. Click the search icon button on the left.
6. A page displaying “GOOGL”’s details should appear
7. Click on the empty heart icon button and it should instead display a filled heart icon button, you have liked this stock.
8. Click on the commend icon button and the focus will shift to the text field below the comments
    a. Comment = <cool!>
    b. Click “Submit’
9. The comment should appear on top of all the comments.

l. Sprint 3: As a user, I should be able to go to the stock viewer for stocks in my portfolio or likes section so that I can conveniently see the details of my preferred stocks. (3)
    i. Scenario
        1. Start application using npm start
        2. Login page will appear in browser
            a. username = <anna.g>
            b. password = <Anna@12345>
            c. Click “Log In” button, should be directed to the home page with the dashboard displayed
        3. Click the “Portfolio” button on the left side drawer and the portfolio details and table should be displayed.
        4. Click on the “TSLA” ticker in the table and stock viewer page for “TSLA” should be displayed.
        5. Click on the profile icon in the top right corner and a menu will appear, click the “Profile” button.
        6. Liked stocks should be displayed under “My Likes”
        7. Click on “TSLA” and stock viewer page for “TSLA” should be displayed.

m. Sprint 3: As a user I should be able to access the application over the internet so that I can access it on multiple devices without needing to download it.
    i. Scenario
        1. Access the application using this link,
        2. Login page will appear in browser
            a. username = <anna.g>
            b. password = <Anna@12345>
            c. Click “Log In” button, should be directed to the home page with the dashboard displayed

n. Sprint 3: As a user, I should be able to see stock news on the dashboard so that I can be up to date on current events related to my stocks and the stock market.(3)
    i. Scenario
        1. Start application using npm start
        2. Login page will appear in browser
            a. username = <anna.g>
            b. password = <Anna@12345>
            c. Click “Log In” button, should be directed to the home page with the dashboard displayed
        3. On the dashboard, news items should be displayed relating to stock tickers.
        4. Upon clicking one of the articles, it should direct you to that link in a new tab.
o. Sprint 3: As a user, I should be able to get stocks that match certain criteria.
    i. Scenario
        1. Start application using npm start
        2. Login page will appear in browser
            a. username = <anna.g>
            b. password = <Anna@12345>
            c. Click “Log In” button, which should be directed to the home page with the dashboard displayed
        3. Click on the “Market Filters” button in the left drawer”
        4. Click on the “Filter Options” Button
            a. Set “Market Cap” to “50M-”
            b. Set “Industry” to “Consumer Electronics”
            c. Click “Find Recommendations”
        5. The table should now display stocks that meet that criteria.
p. Sprint 4: As a user, I want to see the popularity of a stock on widely used apps like Twitter.
    i. Scenario:
        1. Start application using npm start
        2. Login page will appear in browser
            a. username = <anna.g>
            b. password = <Anna@12345>
            c. Click “Log In” button, should be directed to the home page with the dashboard displayed
        3. Click the left side of the search field on the top bar of the home page.
        4. Enter “AAPL” into the field.
        5. Click the search icon button on the left.

6. A page displaying “AAPL”’s details should appear
7. Click on the “Twitter” tab on the far right of the page, tweets relating to the ticker should be displayed.