Product Name: Middleground
Date Modified: 6/1/21

System and Unit Test Report

System Tests

Sprint 1:

- As developers, we want to retrieve and store articles
    - NA
- As a user, I want to have an engaging UI where I can see a variety of media outlets
    - NA

Sprint 2:

- As a user, I want to access and use a secure website.
- As a user, I want to easily view multiple articles from various media outlets simultaneously.
    - Scenario: User is on the `mg.com/`
        - Users will see rows of carousel components with each element being an article with a link to its specific article page.
        - Each carousel component has a title indicating the news publisher it came from.
- As a user, I want to be able to see a home banner with modern design
    - Scenario: User is on the `mg.com/`
        - On the top part of the page, users can see a news article with a description next to it with the picture of the article.
        - If article is not loaded, user will see a black box with a `loading…` label

Product Name: Middleground
Date Modified: 6/1/21

Sprint 3:

- As a user I want to be able to navigate to and read articles
- As a user, I want to be able to share ideas about an article with others
    - Scenario: User is on the `mg.com/{news_publisher}/{article_id}`
        - All instances:
            - Any users will be able to see comments that have been logged from their respective users in a comment board at the bottom of the article page.
        - NoUser Logged In
            - Users will not be able to create/edit/delete.
            - When a user is prompted to type it will redirect them to `mg.com/login`
        - User Logged In
            - Users will be type into the textfield any combination of characters of up to 160 characters.
            - Once typed in they will see their comment with their username associated with it posted.
            - They can have the ability to edit which will prompt the same text in the textfield and upon submission replace the old comment with a new one.
            - Users can reply to comments of themselves and by others which will be shown by an expandable list of replies.
            - They can delete the comment.
                - If they delete it, any replies to the comment will also be deleted.
- As a user I want to be able to access the website
    - Scenario `mg.com/register`:
        - Users will register an account that is available based on the current database when they type into the fields `username` and `password`.
            - `Username = <some_username>`
            - `Password = <P@ssword> (could be anything except empty)`
            - Succeeds if no `<some_username>`
        - Upon a successful registration they will be redirected to `login.`
        - Upon an invalid registration they will be prompted a red flag warning to the user that registration wasn't successful.
            - `Username = <empty>`
            - `User is already registered`
    - Scenario: User on `mg.com/login`:
        - Users will login to an account that is registered in the database when they type into the fields `username` and `password`.

    - Upon a successful login they will be redirected to `/`.
    - Upon an invalid login they will be prompted a red flag warning to the user that login wasn't successful.

<u>Sprint 4</u>:

- As a user I want to be able to use a friendly and intuitive interface
  - NA
- As a user I want to be able to see articles related to the ones I am interested in
  - Scenario: User is on the `mg.com/{news_publisher}/{article_id_1}`
    - Users can see all other articles pertaining to the respective news publisher.
    - Upon clicking one of the articles, users will be redirected to their respective article view page, or `mg.com/{news_publisher}/{article_id_2}`
- As a user I want to be able to access the website online (anytime, anywhere)
  - Scenario: Users on any browser connected to the internet can access a public url from any other site.
    - User can access any article page with id=`article_id and publisher=news_publisher directly` by visiting `mg.com/{news_publisher}/{article_id}`

Product Name: Middleground
Date Modified: 6/1/21

Product Name: Middleground
Date Modified: 6/1/21

Unit Tests

- news.py: the endpoint “/news” takes in a series of query parameters, each corresponding to a column to be matched in the article table. Each query parameter can be a single value or a list. There is also a special parameter “partition_by” that tells the endpoint to divide the results into sublists by the specified column. Test Equivalent classes:
  - list parameters: *{'source': ["cnn", 'bbc-news'], 'title': ['a', 'the'], 'publishedAt': [oneMonthBefore, today]}*
  - single value parameter: *{'source': "cnn", 'title': 'a', 'publishedAt': today, 'limit_articles': 5}*
  - list parameters with more than one value in it: *{'source': ["cnn"], 'title': ['a'], 'publishedAt': [today]}*
  - no parameter: *{}*
  - partition_by plus a list parameter: *{'partition_by': 'source', 'publishedAt': [oneMonthBefore, today]}*

- test_auth.py : Endpoint with the prefix “/auth” has two primary endpoints for user authentication. Register takes two fields of username and password in json format. Set up class is 2 default post data. Each test has permutations of inputs to test each potential output and breakage of the login and register endpoints and verifies the proper database state with its own database instance.
  - Test Equivalence Classes
    - Test Parameters:
      - post_data1 = *{“username”:”Barry Allen”, “password” : “flash123”}*
      - post_data2 = *{“username”:”Clark Kent”, “password” : “Superman!2@#”}*
  - The register endpoint sends 201 success codes that are registered with valid username and password with no other instance of a user in the database with the same username. Shared passwords are allowed.
  - Login endpoint sends 201 success codes on a successful login and 401 on invalid usage which are modified passwords to be not the original values or blank usernames in the post data

- test_comments.py : The endpoints /post accepts a series of query parameters including the username, articleID, and the contents of the comment to post, while /edit accepts the comment ID and new contents of the comment, and /delete accepts only the commentID to delete.

Product Name: Middleground
Date Modified: 6/1/21

- Post Test Equivalence Classes
  - Invalid user: {'username': "BADUSER", 'articleID': '1', 'userComment': "Comment 1"}
  - Invalid articleID: {'username': "GOODUSER", 'articleID': '5555', 'userComment': "Comment 1"}
  - Valid articleID and user: {'username': "GOODUSER", 'articleID': '1', 'userComment': "Comment 1"}
- Delete Test Equivalence Classes
  - Valid articleID : {'commentID': '1'}
  - Invalid articleID: {'commentID': '2'}
- Edit Test Equivalence Classes
  - Valid articleID: {'commentID': '1', 'content': 'Comment 1'}
  - Invalid articleID : {'commentID': '2', 'content': 'Comment 1'}
- For all of the above tests, the database has one user GOODUSER, one article with ID 1, and one comment with ID 1. Passing in invalid parameters results in the response “Unsuccessful. Check Exceptions” with an appropriate exception description. Valid parameters result in the message “comment successful”