Testing

Product/Team Name: Amnis
Date: December 2nd, 2018

Google Login/Logout:

Equivalence classes:

- Test_Login_With_Valid_Account: Tests that you can log in with a valid Gmail account which gets authenticated by Google. Tests that you stay logged in after switching through pages. Expected behavior: successful login, user state maintained.
- Test_Invalid_User_Login: Test returns a pop up error if you press back or don't fill out the sign in form before you try to view lecture. Expected behavior: redirected to main landing page.
- Test_Existing_User_Information_Retrieval: Tests that the database will create a new account for brand new user, and retrieves information (name/student/professor) for older users. Expected behavior: information is retrieved, user is logged in with previous details. Otherwise a new account is created.
- Test_Logout: Tests that you can log out of the site by clicking the logout buttons. Makes sure to give you the option to login after you log out. Expected behavior: user is successfully logged out, redirected to main landing page.

User Authorization Between Pages:

Equivalence classes:

- Test_Student_User_Authorization: Tests that the student is able to login successfully and can navigate through other pages only when logged in (i.e. cannot view lecture page until signed in). Expected behavior: Error modal appears when trying to access lecture pages and not logged in.
- Test_Professor_User_Authorization: Tests that a Professor is also able to login successfully. Makes sure the user has to login before viewing lectures. Expected behavior: Error modal appears when trying to access pages and not logged in.

Questions to discussion board:

Equivalence classes:

- Test_Post_Database_Question: Tests a single input to check if question by the user is added to the database. If the length of the question is 0 then nothing is posted to the database, but if the length is greater than 0 then the question will be posted to the database. Expected behavior: Question posted on length greater than 0, questions updated in database.

- Test_Get_Database_Question: Creates a discussion board with all the questions from the database. Test whether all the database questions show on the website. The question with any length will display in the discussion board. Expected behavior: Send a GET request to the database, and discussion board component is updated with questions.

- Test_Sort_Database_Questions: Discussion board displays questions with the highest score on top and lowest score on the bottom. Test that if a user updates the score in the discussion board than discussion board will sort the questions. Expected behavior: Questions are sorted automatically from initial GET request, ordered by score.

Upvoting questions on discussion board:

Equivalence classes:

- Test_Put_Database_Upvote: Tests a single input to check when you upvote a question, the upvote count is incremented by one in the database. Expected behavior: If question's upvote count is incremented by one in the database, then the test passes.
- Test_Put_Frontend_Upvote: Tests a single input to check when you a upvote a question, the question's upvote count is incremented by one on the front end, without using a get command from the database. Expected behavior: If the score is incremented by one on the page's local copy of questions, then the test passes.

Generating tags:

Equivalence classes:

- Test_Words_With_Count: Tests that the words are recorded in real time with each word's count being updated by occurrence. Expected behavior: Words with higher counts are collected and sent to the database. The main lecture viewing page is then updated with the relevant tags.

- Test_Tag_Click: Tests that the tag will take you to a useful page for reference (such as Wikipedia) on click. Expected behavior: If the given tag has a link associated with it then the user can see the reference opened in a new browser tab.

Routing between Pages:

Equivalence classes:

- Test_Page_Link: Tests that the View Lectures button on the navbar takes you to the main lectures page when logged in. If the user is a professor, they can also see a form for adding a new lecture. Expected behavior: User is redirected to lecture page if logged in, and is able to view lectures (and add a new lecture if professor).

- Test_Amnis_Click: Tests that the Amnis logo button on the navbar takes you to the login page, and when clicked while already logged in, the button logs you out. Expected behavior: User is redirected to main landing page regardless of page (tested from both lecture page and individual stream page).

Lecture page:

Equivalence classes:

- Test_Post_Database_Lecture: Tests to see if professor is able to use a form to create a new lecture and post to the database. Expected behavior: Database is updated and lectures are shown in the user interface.

- Test_Get_Database_Lectures: Creates a discussion board with all the questions from the database. Test whether all the database questions show on the website. The question with any length will display in the discussion board. Expected behavior: Send a GET request to the database, and lecture page component is updated with current/previous lectures.