GENERAL UAT PLAN:

- **Register API Route:**
  - Positive unit test case (1):
    - The user should be able to valid credentials and create a user account within the DB
    - The test case will simulate a POST request to /register with data similar to { username: 'testUser1', password1: 'abc2app*', password2: 'abc2app*' }
    - The test data is representative of valid user credentials (fulfilling minimum username/pwd lengths, etc)
    - The test environment will be localhost
    - The UA testers will consist of members of the team who are knowledgeable about the user acceptance criteria (UAC)
    - We expect an HTTP status of 201 and an HTML body.
  - Negative unit test case (1):
    - The user can't register an account with a username that matches an existing user
    - The test case will simulate a POST request to /register with data similar to { username: 'testUser1', password1: 'abc2app*', password2: 'abc2app*' }. Assuming this test case occurs after the positive test case, 'testUser1' will already exist in the database and we expect an error.
    - The test data is representative of duplicated user credentials because we explicitly create this user in a prior test case
    - The test environment will be localhost
    - The UA testers will consist of members of the team who are knowledgeable about the UAC
    - We expect an HTTP status of 400 and an HTML body
  - Negative unit test case (2):
    - The user can't register an account with a username that is too short (minimum of 5 chars)
    - The test case will simulate a POST request to /register with data similar to { username: 'abc', password1: 'abc2app*', password2: 'abc2app*' }. The username is 3 characters long which is too short.
    - The test data is representative of inadequate user credentials because the username is too short
    - The test environment will be localhost
    - The UA testers will consist of members of the team who are knowledgeable about the UAC
    - We expect an HTTP status of 400 and an HTML body
  - Negative unit test case (3):
    - The user can't register an account with a password that is too short (minimum of 5 chars)
    - The test case will simulate a POST request to /register with data similar to { username: 'testUser2', password1: 'abc', password2: 'abc' }.

        - The test data is representative of inadequate user credentials because the password is too short
        - The test environment will be localhost
        - The UA testers will consist of members of the team who are knowledgeable about the UAC
        - We expect an HTTP status of 400 and an HTML body
- **Login API Route:**
    - Positive unit test case (1):
        - The user should be able to login with credentials corresponding to an existing account in the DB
        - The test case will simulate a POST request to /login with data similar to { username: 'testUser1', password: 'abc2app*' }.
        - This test data is representative of an existing user account because we explicitly create an account with these exact credentials in a previous test case (Register API, positive test case #1)
        - The test environment will be localhost
        - The UA testers will consist of members of the team who are knowledgeable about the UAC
        - We expect an HTTP status of 200 and a redirect to /profile
    - Negative unit test case (1):
        - The user should not be able to login with credentials corresponding to a nonexistent user
        - The test case will simulate a POST request to /login with data similar to { username: 'abc', password: 'abc' }.
        - This test data is representative of a nonexistent account because we clear the database before testing and no other test explicitly creates a user with these credentials.
        - The test environment will be localhost
        - The UA testers will consist of members of the team who are knowledgeable about the UAC
        - We expect an HTTP status of 400 and an HTML body
- **Profile API Route:**
    - Positive unit test case (1):
        - The user should be able to change their password in their profile page.
        - The test case will simulate a POST request to /profile with data similar to { oldPwd: 'abc2app*', newPwd1: 'abc3app*', newPwd2: 'abc3app*' }.
        - This test data is representative of real-world cases because this password corresponds to an existing user created explicitly in a previous test. We will need to ensure the "user" is logged in prior to performing this test.
        - The test environment will be localhost
        - The UA testers will consist of members of the team who are knowledgeable about the UAC
        - We expect an HTTP status of 200 and an HTML body to indicate successful modification of resource

    - Negative unit test case (1):
        - The password fails to meet the requirements for a password.
        - The test case will simulate a POST request to /profile with data similar to { oldPwd: 'abc3app*', newPwd1: 'abc4app*', newPwd2: 'abc5app*' }.
        - This test data is representative of real-world cases because the old password corresponds to an existing user created explicitly in a previous test, but the new password and its confirmation do not match.
        - The test environment will be localhost
        - The UA testers will consist of members of the team who are knowledgeable about the UAC
        - We expect an HTTP response of 400 and an HTML body
- **Question API Route:**
    - Positive unit test case (1):
        - The user should be able to post a new question to a particular class.
        - The test case will simulate a POST request to /question with data similar to { title: “random title”, content: “random content”, class: “random_class_name” }
        - This test data is representative of a typical use case because question content is variable and we only need to ensure that the class exists in the database prior to submitting the request
        - The test environment will be localhost
        - The UA testers will consist of members of the team who are knowledgeable about the UAC
        - We expect an HTTP response of 201 with an HTML body because the route should have directed the user to a new page or rendered a new page.
    - Negative unit test case (1):
        - The user should not be able to post a question with empty content to any class
        - The test case will simulate a POST request to /question with data similar to { title: “”, content: “”, class: “random_class_name” }
        - This data is representative of a typical error because content (or more likely title) may be empty.
        - The test environment will be localhost
        - The UA testers will consist of members of the team who are knowledgeable about the UAC
        - We expect an HTTP response of 400 with an HTML body conveying the error to the user in some user-friendly manner
- **Home API Route:**
    - Positive unit test case (1):
        - The user should be able to view the home page and interact with it
        - The test case will be a GET request to the /home route
        - This test is representative of a typical use-case because users will be visiting our home page often
        - The test environment will be localhost
        - The UA testers will consist of members of the team who are knowledgeable about the UAC

        - We expect an HTTP response of 200 with an HTML body (i.e. the home page has been sent to the user successfully)
    - Negative unit test case (1):
        - The user should not be able to create a new class with an invalid format
        - The test case will be a POST request to /home route with data similar to { class_name: “abc” }
        - This test is representative of a typical use-case because class names may be poorly formatted which hinders accessibility and the ability of other users to find this class on the site
        - The test environment will be localhost
        - The UA testers will consist of members of the team who are knowledgeable about the UAC
        - We expect an HTTP response of 400 with an HTML body to indicate unsuccessful creation of the new class.