
Requirements:

1. Create 1 document per team that describes how at least three features within your finished product will be tested.
2. The test plans should include specific test cases (user acceptance test cases) that describe the data and the user activity that will be executed in order to verify proper functionality of the feature.
3. The test plans should include a description of the test data that will be used to test the feature.
4. The test plans should include a description of the test environment that will be used to test the feature.
5. The test plans should include a description of the test results that will be used to test the feature.
6. The test plan should include information about the user acceptance testers. Please note that the deployed application can only be tested on the CU Boulder campus. So make sure to select your testers accordingly

---

1) Login page

   Test Environment: QA environment
   UA Testers: CU students looking to find a roommate on campus

   a) Features
      i) Authentication
         (1) Test cases
            (a) Logging in with a non exist user
               (i) Expected Behavior:There should be a pop up saying “user does not exist” and will stay on the login page to re enter an existing user.
            (b) Logging in with an incorrect password
               (i) Expected Behavior: There should be a pop up saying “ Incorrect password”
            (c) Logging in with correct username and password:
               (i) Expected behavior: When logging in with the correct information the user should be redirected to the home page
      ii) Buttons / Links
         (1) Register Button
            (a) Expected Behavior: Redirects to the Registration page

(2) Login Button
(a) Expected Behavior: Redirects to the Homepage if the test cases are met

2) Carousel cards
    a) Test cases:
        i) The site should display a potential roommate's name and information on the screen when initially opening the page. There should be a button at the bottom of the screen which you select when you are interested in matching with the displayed user.
        ii) If the user selects the the match button, then they should get a notification: "Added *Username*", and the account they selected should get a message in their inbox indicating that a user matched with them.
        iii) If the user swipes left, the previous account they viewed should be displayed.
        iv) If a user swipes right, a new account should be available for them to use.
        v) If the user selected a dorm, all of the accounts displayed should also be assigned to that dorm.
        vi) If the user has not selected a dorm, all of the accounts displayed should not be assigned a dorm.
        vii) The accounts displayed should match the user's preferences.
        viii) If the user has swiped through all of the current potential accounts, a message will be displayed "End of current selection".
    b) Test environment
        i) Local QA docker image with dummy data for database and admin account access in the db.

3) User search
    a) Test environment
        i) Local QA docker image with dummy data for database and admin account access in the db
    b) UA testers
        i) CU students who may or may not actually be looking for a roommate.
    c) Search bar
        i) Should appear at the top of each page in the nav bar
            (1) Should follow our styling guidelines
        ii) Shouldn't be broken by updating html, static at the top of the page.
    d) Text input
        i) Numbers, special characters, very long inputs, as well as other unicode that could break the text input. Should also test against common sql injection vulnerabilities.

(1) Server should sanitize special inputs and return an error, or not process the data at all when error-prone inputs are entered. As all searches should be names, special characters should not return.

ii) Empty input
    (1) Should return an error for empty input

iii) Regular inputs with results
    (1) Should post the search API and return to the results page with a list of users that were found.

iv) Regular inputs without results
    (1) Should post the search API and return to the results page with an error stating the search returned no results.