1. Register user acceptance tests

- For the field to submit properly the field must include: full name (first and last), password, username, and email with an optional field of date of birth
- Multiple tests can be performed by excluding one or more for each field, in semi-random combinations making sure that most common combinations will spit out an error properly
    - Test data will be proper data for the fields that are filled in for every test to ensure it is not the data that is giving errors
- Another test case can be done by excluding the email field but entering every other field correctly
    - Every other field will be input with working data other than date of birth which will be left blank to ensure the field is truly optional
- Another test case should be done to ensure uniqueness, as the fields username and email have unique constraints upon them within the database definition. So, it should be ensured that an error is given properly when those fields are not unique in the form
    - Two separate tests will be performed, one with username lacking uniqueness and every other piece of data being proper and one with email lacking uniqueness and every other piece of data being proper. This is to ensure it is the lack of uniqueness that will output the errors and not the other data
- All of these tests are also to ensure that proper errors are given when errors are encountered
- All tests will be performed on both the cloud and on localhost, localhost first to make error fixing easier
- All tests will be performed by users outside of the project and those within the project to ensure there is a range of knowledge pertaining to the site

2. Login user acceptance tests

- For the field to submit properly there must be email and password that match together and are found within the database
- One test is having a proper email and password but they do not match each other to ensure the proper error is given for when emails and passwords are not matched properly behind the scenes
    - The test will be a proper email and a proper password but that password does not match the email, this should give an error and block login. Another test will doing the vice versa for insurance purposes though the test will be largely the same
- Another test will be a proper email, but an improper password and the vice versa
    - The first test will be an email present within the database, but a password that does not exist, this test should give an error and block login as it should find the email but not the password. The either test will have an

improper email, but a proper password, this should give the same error as the first test and should still block login
    - Another test is having neither a proper password or a proper email to ensure the correct error is given
        - The data will be some random email and password that is not present within the database to ensure that improper data gives an error and blocks login. This test is essentially testing a random input will not work
    - Another test is leaving both blank to ensure that if the form is empty nothing will submit and nothing will happen along with the proper error given
        - This test will purposely have no data and nothing will be given, the html should block this, but if it somehow gets through this should give the proper error and block login
    - Another test is putting proper data in both fields to ensure that it will submit properly if there are no errors
        - This test's data will give an email and password that match and that both exist within the database. When the data is submitted it should login properly and render the home page
    - Another test is leaving one or the other field blank to ensure the form will also not submit and the proper error is given
    - All of these tests are also to ensure that proper errors are given when errors are encountered
    - All tests will be performed on both the cloud and on localhost, localhost first to make error fixing easier
    - All tests will be performed by users outside of the project and those within the project to ensure there is a range of knowledge pertaining to the site

3. Search events via preferences user acceptance tests

- The proper way this page should work is the user puts in between 1 and 3 preferences, then the data is searched against the preference data present within every event instance using a like sql search
- One test will be inserting no data to ensure that the form will not submit
    - There will be no data for this test it will just be a test to ensure that the form will not submit when no preference data is given. The html should block this but it is better to be safe than sorry
- Another test will be inserting 1, 2, and 3 proper preferences into the preference data fields
    - There will be one set of data for this test with 3 improper preference datas, then the test will be inserting 1, then 2, then all 3 datas to ensure that first: the form works for any number of preference datas, and second: the form will still give the same error regardless of the number of improper test datas given and will still work even if there are 3 improper test datas given

- Another test will be inserting one or more improper preference datas into the forms but at least 1 proper preference data into the form to ensure the computer will still give some matching events
    - This test will have two data sets: one with 1 proper 2 improper, another with 2 proper 1 improper. There is none with all 3 as proper or improper because those are reserved for other tests. This is to ensure that the matching events will still be given even if there are some preference datas inserted that yield no results, so even if one preference data is the only one to yield any results it should still output those matching from that one result
- Another test will be all improper preference data to ensure the proper error message is given when no matching events are found. Inserting 1, 2 and 3 improper preference datas to ensure all 3 ways the form can go work the same and properly
    - The test data will be 3 preference datas that have no matches within the database, the expected output should not be a blank events page, but rather an events page with some message along the lines of “no events found for those preferences” and should give the user a way to go back to the search to try again with different preferences. This test is also to ensure nothing is bricked if no results are found
- The last test will be inserting 1, 2, and 3 proper preference datas into the forms to ensure that the site will give events regardless of how many proper preference data(s) are given
    - The test data will be 3 proper preference datas that exist within the database and will all yield at least one event result when used as a search term. The test will consist of inserting 1, then 2, then all 3 proper preference datas to ensure that events will be output properly for all 3 preference datas. 1 preference should output the events matching that 1, 2 preferences should output events matching those two, and 3 should output events matching 3. Ensures that it does not freak out when there is one or more preference datas
- All of these tests are also to ensure that proper errors are given when errors are encountered
- All tests will be performed on both the cloud and on localhost, localhost first to make error fixing easier
- All tests will be performed by users outside of the project and those within the project to ensure there is a range of knowledge pertaining to the site