3. User Acceptance Testing

Definition

https://rapidapi.com/tipsters/api/priceline-com-provider

This is the last phase of the software testing process. During UAT, actual users test the software to make sure it can handle the required tasks in real-world scenarios, according to the specifications.

Below are some sample test cases for UAT testing:

- User should be able to login with correct credentials.
- User authentication fails when the user provides invalid credentials.
- The form provides the user with specific feedback about the error.

Acceptance Criteria:

Acceptance criteria refers to a set of predefined requirements that must be met in order to mark a user story as complete. Below is an example :

- A user cannot submit a form without completing all of the mandatory fields. Mandatory fields include:
    - Name
    - Email Address
    - Password
    - Confirm Password
- Information from the form is stored in the `user` table in the `football_db` database.

To-Do: Create UAT plans for at least 3 features

REMEMBER

You have to execute this test plan in week 4 of your project. So it would be ideal if your test plan is well thought through as it will expedite the testing process.

1. Create 1 document per team, within the milestones folder in the project directory, that describes how, at least, 3 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.

UAT Plans:

Feature 1: Search Form
Test Cases:

1. Searching for one way flights from LAX to NYC returns the correct list of flights
   Acceptance criteria:
   - The user must input the exact airport codes NYC and LAX to get the information
   - The user must input valid dates for departure and return
   - The user must input a type of flight (ECO, BUS…etc)

   Test Data: Departure: LAX Departure Date: 2/2/2024 Arrival Location: NYC
   Test Environment: On the website
   Test Results: A list of flights is returned to the user corresponding to the flight information

2. Searching for a date prior to the current date returns an error message with an invalid date.
   Acceptance criteria:
   - The user will input any previous date from the current day of testing.
   - The user will be given a message which states: “Invalid date.”
   - The user will not be shown any search results.

   Test Data: A date and time before the current date
   Test Environment: Website
   Test Results: By selecting a date and time that has already passed in relation to the current date will return an error message to the user expressing an invalid input.

3. Searching for a location returns a list of hotels at that location.
   Acceptance criteria:
   - The user must click the hotel option in the drop down
   - The user must input a valid location in the world.

   Test Data: The test data that will be Denver
   Test Environment: Website

Test Results: by providing the location Denver, a list of hotels in or near Denver is provided.

Feature 2: Adding to Cart and checking out

Test Cases:

1. Adding an item to the cart updates the cart with whatever information was in the item.
   Acceptance criteria:
   - The user must choose the specific flight/car/hotel they would like to add to their cartItem
   - The cart item is added when user clicks on “add” button

   Test Data: The items chosen and added to the cart
   Test Environment: test on the front end and the inspect console.
   Test Results: The newly added item should be reflected in the item array each time.

2. Adding the same item to the cart updates the quantity in the cart for the given item.
   Acceptance criteria:
   - User will select an item that has already been added to the cart.
   - Doing so will add the item to the cart

   Test Data: Add an item previously added to the cart
   Test Environment: On website
   Test Results: Result should update quantity of items in cart

3. Deleting a cart item removes the cart item successfully.
   Acceptance criteria:
   - The added item should be deleted from the current cart list

   Test Data: the cart array at the time of testing
   Test Environment: the website and the inspect console
   Test Results: successfully remove any items from the cart and the both the front and the database is reflected of the delete operation.

4. Users can not add additional items to cart that cause time conflicts, unless they are adding duplicate items.
   Acceptance criteria:
   - The user will attempt to add an item which causes a time conflict with an item already within the cart, and then the user will be prompted with a message: “Can not add to cart because due to time conflicts.”
   - The user must not be able to add any items which cause time conflicts and which are not duplicate items, and must be prompted with a warning message.
   - The user will add two of the same item to their cart, and the user must be allowed to add duplicate items.

   Test Data: The user will attempt to add an item which causes a time conflict with an item already within the cart. The user will attempt to add two of the same item to their cart.
   Test Environment: Website.
   Test Results: When attempting to add a new item with a time conflict, the user will be

prompted with the message: “Can not add to cart because due to time conflicts.” and will not be allowed to add the item to their cart. When attempting to add a duplicate item, the user will be allowed to add the item to their cart.

Feature 3: Cart and Planner working together.

Test Cases:

1. User’s cart is reflected in the user’s planner.
   Acceptance criteria:
   - The user’s planner will have an ordered list of all the items which the user has “purchased.”
   - The user’s planner will not include any items within the cart which the user has not yet “purchased.”
   - The user’s planner will be made up of cards which are ordered by date.

   Test Data: Data that was submitted in the cart
   Test Environment: Website: Portfolio tab
   Test Results: User can see a list of all purchased items ordered by date of the item in their planner tab.

2. User is redirected to the planner after checking out.
   Acceptance criteria:
   - The user will add items to their cart, and the user will click the “Checkout” option once finished shopping.
   - The user will be shown their planner after clicking “Checkout” from their cart.
   - The user’s planner will show only items that have been purchased by the user
   - The user’s planner will *not* show items which are in the cart but have not been purchased.

   Test Data: The user should add any one of each item, and then checkout from their cart. The user should add another item to the cart after checking out, and then return to their planner without “purchasing” the final item in their cart.
   Test Environment: Website.
   Test Results: User is redirected to the user’s planner page, which shows an ordered list of all of their purchased items, and does not show any items which have not been purchased yet.

3. Duplicate items have multiple cards/are displayed multiple times.
   Acceptance criteria:
   - The user will add duplicate items to their cart, and the user will click the “Checkout” option once finished shopping.
   - The user will be shown their planner after clicking “Checkout” from their cart.
   - The user’s planner will show the duplicate items twice, or equivalent to the quantity purchased.

   Test Data: The user should add any number of the same item, and then checkout from their cart. The user should be shown their planner after checkout and multiple cards of that item should appear.

Test Environment: Website.
Test Results: User is redirected to the user's planner page, which shows an ordered list of all of their purchased items, and shows any duplicate items displayed multiple times equivalent to the times purchased.