**Create high level Test Plan for "Rent a car" functionality of https://www.lyft.com/rider/rentals.**

Introduction: The purpose of this test plan is to define the testing approach for the "Rent a car" functionality of <https://www.lyft.com/rider/rentals>. The testing will focus on verifying the functionality and usability of the system, including the ability to reserve, pick up, and return a rental car.

Scope: The scope of this testing will cover the "Rent a car" functionality of <https://www.lyft.com/rider/rentals>, including the reservation process, pickup process, and return process. The testing will be performed manually, and will focus on functional, usability, and compatibility testing. The testing will not cover security, performance, or load testing.

Test Approach: The testing will be performed manually and will focus on the following areas:

* Reservation Process: Verify that users can reserve a car online and that the reservation process is user-friendly and error-free.
* Pickup Process: Verify that users can easily pick up the reserved car at the designated location.
* Return Process: Verify that users can easily return the rented car at the designated location and that the return process is error-free.
* Usability: Verify that the user interface is easy to use, intuitive, and user-friendly.
* Compatibility: Verify that the system is compatible with popular web browsers, operating systems, and devices.

Test Environment: The testing will be performed on the production environment of <https://www.lyft.com/rider/rentals>. The following tools will be used for testing:

* Web Browsers: Google Chrome, Mozilla Firefox, Microsoft Edge
* Operating Systems: Windows 10, macOS Mojave, iOS 14, Android 10a
* Devices: Desktop/laptop, smartphone, and tablet

Test Cases: The following test cases will be executed during testing:

1. Reservation Process:

* Verify that users can select the pickup and return date and time.
* Verify that users can select the car type.
* Verify that users can view the total cost of the rental.
* Verify that users can confirm the reservation.

1. Pickup Process:

* Verify that users can easily find the pickup location.
* Verify that users can easily pick up the reserved car.
* Verify that the condition of the car is acceptable.

1. Return Process:

* Verify that users can easily find the return location.
* Verify that users can easily return the rented car.
* Verify that the return process is error-free.

1. Usability:

* Verify that the user interface is easy to use and intuitive.
* Verify that the text on the website is easy to read.
* Verify that the website is responsive and adjusts to different screen sizes.

1. Compatibility:

* Verify that the website works correctly on different web browsers.
* Verify that the website works correctly on different operating systems.
* Verify that the website works correctly on different devices.

Conclusion: This test plan outlines the approach and test cases for testing the "Rent a car" functionality of <https://www.lyft.com/rider/rentals>. By executing these tests, we can ensure that the system is functional, user-friendly, and compatible with different platforms and devices.

**Describe in details at least 2 the most critical scenarios (1 positive and 1 negative) to test “Rent a car” form functionality of https://www.lyft.com/rider/rentals.**

Two critical scenarios for testing the "Rent a car" form functionality of <https://www.lyft.com/rider/rentals>, one positive and one negative:

Scenario 1: Positive Test Scenario - Successful Car Reservation

Objective: To verify that a user can successfully reserve a car through the "Rent a car" form functionality.

Preconditions:

* The user has a valid Lyft account and is logged in.
* The user has access to a compatible device and internet connection.

Test Steps:

1. Navigate to the "Rent a car" page of <https://www.lyft.com/rider/rentals>.
2. Select the pickup location and date/time and the return location and date/time.
3. Select the type of car that you would like to rent.
4. Verify that the total rental cost is displayed accurately.
5. Enter your payment information and confirm the reservation.
6. Verify that the confirmation page is displayed and includes the details of your reservation.

Expected Result:

* The user is able to successfully reserve a car and receives a confirmation of the reservation.

Scenario 2: Negative Test Scenario - Invalid Payment Information

Objective: To verify that the "Rent a car" form functionality properly handles invalid payment information.

Preconditions:

* The user has a valid Lyft account and is logged in.
* The user has access to a compatible device and internet connection.

Test Steps:

1. Navigate to the "Rent a car" page of <https://www.lyft.com/rider/rentals>.
2. Select the pickup location and date/time and the return location and date/time.
3. Select the type of car that you would like to rent.
4. Verify that the total rental cost is displayed accurately.
5. Enter invalid payment information (e.g., an expired credit card).
6. Confirm the reservation.

Expected Result:

* The system should not allow the user to complete the reservation with invalid payment information.
* The user should receive an error message indicating that the payment information is invalid and prompted to enter valid payment information before confirming the reservation.

By testing these two critical scenarios, we can ensure that the "Rent a car" form functionality of <https://www.lyft.com/rider/rentals> is properly functioning and able to handle both successful reservations and invalid payment information.

**Investigate and find defects on the web-site** [**http://automationpractice.com**](http://automationpractice.com/)**. Create detailed description for all defects found and share the report**

To investigate and find defects on the website [http://automationpractice.com](http://automationpractice.com/), you can follow these steps:

1. Plan the testing approach: Create a test plan that outlines the testing objectives, test scope, test scenarios, and test tools. Identify the areas of the website that need to be tested, such as the user interface, functionality, usability, and performance.
2. Conduct functional testing: Test the functionality of the website by performing various actions such as logging in, registering, searching, adding products to the cart, checking out, and placing orders. Verify that the website performs these functions as expected and does not produce any errors or unexpected behaviors.
3. Conduct usability testing: Test the usability of the website by evaluating the user interface, navigation, design, and layout. Verify that the website is easy to use, intuitive, and aesthetically pleasing.
4. Conduct compatibility testing: Test the compatibility of the website by checking its functionality and usability on different web browsers, devices, and operating systems.
5. Conduct performance testing: Test the performance of the website by evaluating its speed, responsiveness, and scalability. Verify that the website responds quickly, does not crash or freeze, and can handle a large number of users and transactions.
6. Report defects: Create a detailed report of all the defects found during testing. Include the following information in the defect report:

* Description: Provide a brief description of the defect, including its symptoms and impact on the website.
* Steps to reproduce: Describe the steps required to reproduce the defect.
* Expected result: State what the expected outcome should be.
* Actual result: State what the actual outcome was.
* Severity: Rate the severity of the defect on a scale of 1 to 5, with 1 being the lowest severity and 5 being the highest.
* Priority: Assign a priority to the defect based on its severity and impact on the website.

By following these steps, you can conduct thorough testing of the website and create a detailed report of all the defects found. The defect report can then be used by the website developers to fix the issues and improve the functionality and usability of the website.