**E\_commerce website**

**Part 1\_ Test planning:**

**Test Strategy Document**

**Objectives of Testing**

The objective is to ensure the quality and functionality of the e-commerce website through systematic testing.

**Scope of Testing**

Testing will cover all levels (unit, integration, system, acceptance) and types (functional, usability, performance) of testing.

**Testing Levels**

* Unit Testing: Validate individual components.
* Integration Testing: Verify the interaction between components.
* System Testing: Evaluate the system as a whole.
* Acceptance Testing: Ensure the system meets business requirements.

**Testing Types**

* Functional Testing: Validate functionality.
* Usability Testing: Assess user interface and experience.
* Performance Testing: Check system response and scalability.

**Entry and Exit Criteria**

* Entry Criteria: Code is ready for testing, environment is set up.
* Exit Criteria: All test cases executed, critical defects fixed.

**Test Environment and Tools**

* Web browsers: Chrome, Firefox.
* Automation framework: Selenium.
* CI/CD: Optional.

**Risk Analysis**

* Risks include changes in requirements, browser compatibility issues, and data security concerns.

**Test Plan**

**Test Deliverables**

* Test cases (functional, usability, performance).
* Automated test scripts.
* Test execution reports.

**Test Schedule**

* Start Date: [07/01/23]
* End Date: [09/01/23]

**Test Resources**

* Testing Team: [Prashant Singh, A, B, X, Y, Z]
* Test Environment: [Selenium]
* Test Data: [Specify Test Data]

**Test Data and Environment Setup**

* Test Data: Used realistic test data.
* Test Environment Setup: Ensured the website is deployed on the test server.

**Test Execution and Reporting**

* Execution: Run automated tests and manual tests if needed.
* Reporting: Document test results and any defects found.

**Part 2: Test Case Design**

**User Registration**

1. **Positive Test Case:**

* Navigate to the registration page.
* Fill in valid details.
* Verify successful registration.

1. **Negative Test Case: Invalid Email:**

* Navigate to the registration page.
* Enter an invalid email.
* Verify that an error message is displayed.

1. **Negative Test Case: Password Mismatch:**

* Navigate to the registration page.
* Enter valid details but with mismatched passwords.
* Verify that an error message is displayed.

**Edge and Boundary Test Cases**

1. **Edge Case: Maximum Product Quantity:**

* Add the maximum number of products to the cart.
* Verify that the cart handles the maximum quantity correctly.

1. **Edge Case: Empty Cart:**

* Remove all items from the cart.
* Verify that the cart is empty.

**Part 3: Test Automation**

**Test Automation Framework:**

**For this e\_commerce site, I'll use Selenium with Python. The choice is based on Selenium's popularity, cross-browser support, and ease of use.**

**Automated Test Scripts**

**Selenium with Python**

**Installation:**

1. Install Python: [<https://www.python.org/downloads/>]
2. Install Selenium: pip install selenium
3. Download a WebDriver (ChromeDriver) and ensure it's in your system's PATH.

**Automated Test Scripts:**

1. Create a Python file (e.g., test\_registration.py).

**test\_registration.py**

from selenium import webdriver  
import time

**initialize the web driver**

driver = webdriver.Chrome()

**navigate to the registration page**

driver.get("<https://your-e-commerce-website.com/register>")

**Positive test case for user registration**

def test\_positive\_registration():  
# locate registration form elements and fill in valid details  
driver.find\_element\_by\_id("username").send\_keys("testuser")  
driver.find\_element\_by\_id("email").send\_keys("[testuser@example.com](mailto:testuser@example.com)")  
driver.find\_element\_by\_id("password").send\_keys("testpassword")  
driver.find\_element\_by\_id("confirm\_password").send\_keys("testpassword")  
  
# submit the registration form  
driver.find\_element\_by\_id("register\_button").click()  
  
# wait for registration process to complete  
time.sleep(2)  
  
# assert that registration was successful  
assert "Welcome" in driver.page\_source

**Negative test case for user registration (invalid email)**

def test\_negative\_registration\_invalid\_email():  
# locate registration form elements and fill in invalid email  
driver.find\_element\_by\_id("username").send\_keys("testuser")  
driver.find\_element\_by\_id("email").send\_keys("invalid\_email")  
driver.find\_element\_by\_id("password").send\_keys("testpassword")  
driver.find\_element\_by\_id("confirm\_password").send\_keys("testpassword")  
  
# submit the registration form  
driver.find\_element\_by\_id("register\_button").click()  
  
# assert that there is an error message for invalid email  
assert "Invalid email" in driver.page\_source

**close the browser window**

driver.quit()