|  |  |
| --- | --- |
|  | **Define Regression testing and its purpose.**  Regression Testing: The process of re-running previously conducted tests on the software to ensure that recent changes haven't adversely affected existing functionalities.  Purpose: To confirm that modifications or enhancements to the software have not introduced new defects or caused regression in existing features. |
|  | **List the levels of testing.**   * Unit Testing * Integration Testing * System Testing * Acceptance Testing |
|  | **Mention the purpose of Unit Testing.**  Purpose: To test individual units or components of the software in isolation to ensure they function correctly according to specifications and requirements. |
|  | **Differentiate Test Case and Test Scenario.**  Test Case: A detailed set of conditions or variables under which a tester will determine whether a system under test satisfies requirements or works correctly.  Test Scenario: A hypothetical story used to help a person think through a complex problem or system. |
|  | **Understand the purpose of DSR in testing.**  DSR (Defect Severity Rating): It's a measure of the impact a defect has on the functionality of the software. It helps prioritize which defects should be fixed first based on their severity. |
|  | **Define Test case prioritization.**  Test Case Prioritization: The process of ordering test cases based on factors such as business impact, risk, and dependencies to ensure that the most critical functionalities are tested first. |
|  | **Write down the types of X-path locators.**   * Absolute XPath * Relative XPath |
|  | **Compare and contrast driver.quit() and driver.close().**  driver.quit(): Closes all browser windows and terminates the WebDriver session, releasing the associated process.  driver.close(): Closes the current browser window but keeps the WebDriver session open for further interactions. |
|  | **Illustrate the fluent wait with a code.**  FluentWait wait = new FluentWait(driver)  .withTimeout(Duration.ofSeconds(30)) // Maximum wait time  .pollingEvery(Duration.ofSeconds(5)) // Frequency of checking  .ignoring(Exception.class); // Exception to ignore  wait.until(ExpectedConditions.alertIsPresent());  driver.switchTo().alert().accept(); |
|  | **Illustrate the link text locator with an example.**  WebDriver driver = new ChromeDriver();  driver.get(baseUrl);  driver.findElement(By.linkText("click here")).click(); |
|  | **List two levels of software testing.**  Two levels of software testing are:  Unit Testing: This is the lowest level of testing, where individual units or components of a software application are tested independently. Unit testing focuses on verifying that each unit of the software performs as designed. It is typically performed by developers during the development phase.  Ex : A developer writes unit tests to verify the functionality of the addToCart() method in the shopping cart module. The unit test ensures that when an item is added to the cart, it reflects the correct quantity and total price.  Integration Testing: Integration testing involves testing the interaction between different units or components of a software application. It verifies that the integrated units work together correctly as a group. Integration testing ensures that the software modules interact seamlessly and that data is passed correctly between them.  Ex :  An integration test is written to simulate adding an item to the cart and then proceeding to checkout. This test verifies that when a user adds items to the cart, the inventory is updated accordingly, and the payment process is initiated seamlessly. |
|  | **Name two types of software testing.**  There are two main categories of software testing: functional and non-functional testing.   * **Functional testing** focuses on whether the software works as intended according to the requirements and specifications. This type of testing ensures that all the features of the software are working correctly and that the software behaves as expected by the users. Here are some common types of functional testing:   + Unit testing: This type of testing focuses on individual units of code, such as functions or classes.   + Integration testing: This type of testing verifies that different modules of the software work together correctly.   + System testing: This type of testing tests the entire software system as a whole.   + Acceptance testing: This type of testing is performed by the end users of the software to ensure that it meets their needs. * **Non-functional testing** focuses on how well the software performs. This type of testing ensures that the software is usable, reliable, secure, and performant. Here are some common types of non-functional testing:   1. Usability testing: This type of testing evaluates how easy and user-friendly the software is to use.   2. Performance testing: This type of testing measures the speed, scalability, and stability of the software.   3. Security testing: This type of testing identifies and mitigates security vulnerabilities in the software. |
|  | **Identify two components of test scenario design.**  Two components of test scenario design are:  Functional Test Scenarios: Functional test scenarios outline specific functionalities or features of the software application that need to be tested. These scenarios describe the actions that users would perform in the application and the expected outcomes. Functional test scenarios ensure that the software behaves as expected and meets the requirements specified by stakeholders.  Non-functional Test Scenarios: Non-functional test scenarios focus on aspects of the software application that are not directly related to its specific functionalities but are crucial for its overall performance and user experience. These scenarios cover attributes such as performance, reliability, scalability, security, usability, and compatibility. Non-functional test scenarios help evaluate the application's behavior under different conditions and ensure that it meets quality standards. |
|  | **Mention two elements of test case design.**  Here are two key elements of test case design:   1. **Test Steps:** This element outlines the specific actions the tester needs to take to execute the test. It should be a clear, step-by-step guide that ensures consistency and repeatability when running the test.expand\_more 2. **Expected Results:** This element defines what the tester anticipates to happen after following the test steps. It describes the desired outcome of the test and provides a basis for comparison with the actual results obtained during test execution. By having clear expected results, testers can efficiently determine if the software behaves as intended. |
|  | **Design a test- case which should have two functional and non-functional of a e-commerce application.**   | **Test Case: Purchase Process** | **Functional Test Scenarios** | **Preconditions: User is logged in to the e-commerce website.** | **Test Steps** | **Expected Results: The selected product should be successfully added to the cart, and the cart icon should display the updated item count.** | | --- | --- | --- | --- | --- | | 1. Add Product to Cart | Preconditions: User is logged in to the e-commerce website. | Test Steps: 1. Navigate to the product page. 2. Click on the "Add to Cart" button. 3. Verify that the product is added to the cart. | Expected Results: The selected product should be successfully added to the cart, and the cart icon should display the updated item count. |  | | 2. Checkout Process | Preconditions: User has one or more items in the shopping cart. | Test Steps: 1. Navigate to the shopping cart page. 2. Click on the "Checkout" button. 3. Enter the shipping address and payment details. 4. Click on the "Place Order" button. | Expected Results: The user should be able to complete the checkout process successfully, and an order confirmation page should be displayed with the order details. |  | |  |  |  |  |  | | Non-Functional Test Scenarios | 1. Performance Testing - Page Load Time | Preconditions: User accesses the e-commerce website. | Test Steps: 1. Measure the page load time of the home page, product pages, and checkout process. 2. Simulate multiple concurrent users accessing the website. | Expected Results: The page load time should be within acceptable limits, and the website should be able to handle concurrent user traffic without significant degradation in performance. | |  | 2. Security Testing - Data Encryption | Preconditions: User submits sensitive information during the checkout process. | Test Steps: 1. Inspect the network traffic to verify that data transmission is encrypted using HTTPS. 2. Attempt to intercept and decode any transmitted data. | Expected Results: All sensitive data, such as login credentials and payment information, should be encrypted during transmission, and no unauthorized access to sensitive information should be possible. | |
|  | **Understand the purpose of RTM in testing.**   * **Ensures Complete Test Coverage:** RTM guarantees all software requirements are covered by designed test cases. * **Simplifies Defect Tracking:** RTM helps trace defects back to the specific requirement they impact. * **Demonstrates Compliance:** RTM serves as a record showing developed software aligns with initial requirements. |
|  | **Difference between driver.get() and driver.navigate().**   * Use driver.get() for simple navigation to a URL where you don't need browser history or session state management. * Use driver.navigate().to() when you need to navigate within the browser history or want to maintain session state (cookies) across page loads. This can be particularly important for testing scenarios simulating user behavior that involves logins or interacting with multiple pages within a single session. |
|  | **Illustrate with example how you will create and close the Edge browser instance.**  public class EdgeExample {  @Test  public void method1() {  // Set the system property for the Edge driver path (replace with your actual path)  WebDriverManager.chromedriver().setup();  // Create a new Edge driver instance  WebDriver driver = new EdgeDriver();  // Open a URL (replace with your desired URL)  driver.get("https://www.google.com");  // Perform your test actions here (replace with your test logic)  // ...  // Close the Edge browser instance  driver.quit();  } |
|  | **Difference between Implicit and explicit wait.**  Difference between Implicit and Explicit wait |
|  | **Write down code for handling dropdown using select.**  WebElement dropdownElement = driver.findElement(By.id("dropdown"));  // Create a Select object and pass the dropdown element to it  Select dropdown = new Select(dropdownElement);  // Select an option by visible text  dropdown.selectByVisibleText("Option 1");  // Select an option by value  dropdown.selectByValue("value1");  // Select an option by index (starting from 0)  dropdown.selectByIndex(2);  // Close the browser  driver.quit(); |

**PART – B**

1.Elaborate on the various levels of testing in software development, starting from the granular examination of individual components in unit testing, progressing to the validation of integrated modules in integration testing, followed by comprehensive system testing to assess overall functionality and performance, and culminating in acceptance testing to ensure alignment with business requirements and readiness for deployment.

• **Unit Testing**

**• Integration Testing**

**• System Testing**

* **Acceptance Testing**

2.Discuss the various types of software testing methodologies, providing a brief explanation of each along with a suitable example for illustration.

**Waterfall Model, Agile Model, V Model**

3. Discuss the role of software testing in ensuring the reliability and functionality of software systems. Explain the various levels of testing and their significance in detecting defects and improving software quality.

**Test Manager, Test Lead, QA, QC, Validatio n, Verification,**

**Unit Testing**

**• Integration Testing**

**• System Testing**

**• Acceptance Testing**

4. How does testing contribute to the Software Development Life Cycle (SDLC)? Provide a comprehensive analysis of the testing process and its significance throughout the SDLC stages.

**Plan, Analysis, Design, Code&Test, Deployment, Maintenance**

5. Formulate a Functional Requirements document, articulate a corresponding Test Scenario (at least 4), and design a TestCase document(minimum 4 testcases for each Test scenario) specifically tailored for the Facebook application, highlighting the cognitive processes involved in each step

6. Develop test cases for the user registration and login system of the website. For the following test scenarios write the appropriate test cases:

a. Include test cases for validating user input during registration (e.g., email format, password strength).

b. Define test cases to verify successful registration, including email verification if applicable.

c. Outline test cases to validate the login process, including correct and incorrect credentials scenarios.

d. Consider edge cases such as concurrent user registrations or login attempts with expired credentials.

7. Design test cases for the checkout process of the e-commerce section on the website. For the following test scenarios write the appropriate test cases:

a. Include test cases to verify adding items to the cart, updating quantities, and removing items.

b. Define test cases for different payment methods (e.g., credit card, PayPal) and their integration with the checkout process.

c. Outline test cases to validate order confirmation, including order summary, shipping details, and payment confirmation. d. Consider scenarios such as abandoned carts, payment failures, and order cancellations.

8. Design 3 test scenarios for Search flight, Airport transfers and 6 test cases for the website https://www.redbus.com/ . Use the Proper template.

a)Define positive and negative test cases

b)Define test case prioritization

c)Define test case optimization

9. Develop six test cases for specific functionalities of the Flipkart.com website, covering areas such as adding items to the cart, checking order status, and managing user profiles. Use a proper test case template and include positive and negative test cases.

10. Discuss the testing strategy for the DemoBlaze website, covering positive and negative test cases, test scenarios for features like Product Search and Order Placement, and specific test cases for functionalities such as Adding Products to Cart, Viewing Product Details, and Managing User Accounts.

Please provide 6 test scenarios and 4 test cases for each scenario in your response

11. Discuss the challenges and considerations involved in testing a complex e-commerce website like ixigo.com, including factors such as multiple functionalities, user interactions, third-party integrations, and scalability. Additionally, propose 6 test scenarios or test cases that address specific challenges or areas of concern identified in the ixigo.com website testing context. Provide a rationale for each proposed test scenario or test case.

12. Identify and discuss three major challenges faced in testing complex e-commerce websites such as ixigo.com. Based on these challenges, propose 6 test scenarios or test cases that address specific areas of concern in testing ixigo.com. Provide a rationale for each proposed test scenario or test case.

**13.** Apply your understanding of Selenium WebDriver locators by illustrating different types of locators and providing suitable examples for each type. Analyze the effectiveness of using appropriate locators based on the characteristics of web elements and evaluate their impact on test automation reliability.

**14.** Illustrate the synchronization methods in selenium with a suitable code and explain their functionalities.

**15.** Discuss Selenium WebDriver to demonstrate effective strategies for handling alerts, iframes, and calendar elements in test automation. Identify challenges faced in dealing with these elements and provide detailed examples and Java code snippets showcasing your approach to interact with them, manage synchronization, and ensure the reliability of test automation scripts.

**16.** Illustrate the different methods of testing different form elements with suitable examples for each elements.

**17.** Write a Java Selenium program to implement the following steps.

1. Open the Page:  https://j2store.net/free/
2. Get the URLof the Page
3. Validate the URL of the Page is “https://j2store.net/free/”
4. Click on the Clothing link.
5. Get the URLof the Page
6. Validate the URL of the Page is https://j2store.net/free/index.php/shop?filter\_catid=11

**18.** Demonstrate how to locate and interact with web elements using WebDriver methods like findElement, findElements, click, sendKeys, getText, functions., with examples.

19. Analyze the advanced techniques of keyboard and mouse handling, JavaScript executor usage, and web table handling in Selenium WebDriver automation. Discuss the importance, challenges, strategies, and implementation approaches for each of these elements. Provide detailed examples, code snippets, and practical scenarios demonstrating proficiency in using these features for effective test automation.

**20.** Discuss common types of web pop-ups, alerts, and frames encountered in web applications and their impact on Selenium WebDriver testing