ALGORITHM FOR PHASE END PROJECT AUTOMATE AN ECOMMERCE WEB APPLICATION

Algorithm: Capture Screenshot

1. Start the WebDriver and create a constructor for the BasePage class to initialize the WebDriver using the DriverManager class.
2. Implement the captureScreenshot method in the BasePage class, taking the name of the screenshot as input.
3. Typecast the WebDriver instance to TakesScreenshot to enable capturing screenshots.
4. Capture the screenshot using the getScreenshotAs(OutputType.FILE) method, obtaining a source file containing the screenshot.
5. Create a destination path with a unique name for the screenshot, using the screenshotCounter to generate unique names.
6. Attempt to create the directories specified in the destination path using Files.createDirectories(destination.getParent()).
7. Copy the source file to the destination path using Files.copy(source.toPath(), destination).
8. If successful, print a message indicating that the screenshot is captured and saved.
9. If any error occurs during the process, catch the IOException and print an error message.

Algorithm: Search for a Product

1. In the FlipkartHomePage class, create a constructor to initialize the WebDriver using the DriverManager class.
2. Implement the getHomepageLoadTime method to get the homepage loading time by navigating to the Flipkart homepage using driver.get(baseUrl).
3. Execute JavaScript to calculate the load time using window.performance.timing.
4. Implement the searchForProduct method to search for a product:

- ❖ Close any login popup if displayed using driver.findElement(searchInput).sendKeys(Keys.chord(Keys.ESCAPE)).
- ❖ Enter the search text using driver.findElement(searchInput).sendKeys(productName).
- ❖ Construct the XPath for the specified product using the provided searchOptions string.
- ❖ Select the product from the search options using driver.findElement(By.xpath(searchOptionsProductXpathString)).click().
- ❖ Click the search button using driver.findElement(searchButton).click().

Algorithm: Search Results Page

1. In the SearchResultsPage class, create a constructor to initialize the WebDriver using the DriverManager class.
2. Implement the isSearchResultsDisplayed method to check if search results are displayed by checking if the product results list is empty using driver.findElements(productResults).isEmpty().
3. Implement the getSearchResults method to get the list of search results using driver.findElements(productResults).
4. Implement the scrollToElement method to scroll to a specific element on the page using JavaScript executor.
5. Implement the isImageDisplayed method to check if the product image is displayed in the search results using JavaScript executor.
6. Implement the hasScrollFeature method to check if the page has a scroll feature using JavaScript executor.
7. Implement the scrollToBottom method to scroll to the bottom of the page using JavaScript executor.
8. Implement methods to get the number of visible products, visible images, and the total number of images using driver.findElements(productResults) and findElements(imageSelector).
9. Implement the isAtBottomOfPage method to check if the cursor is at the bottom of the page using JavaScript executor.

Algorithm: Create Excel File

1. Create a class CreateExcelFile with the main method.
2. Initialize the workbook using new XSSFWorkbook().
3. Create a new sheet named "Sheet1" using workbook.createSheet("Sheet1").
4. Add headers "Name", "Age", and "Occupation" to the header row.
5. Create the data array with the sample data.
6. Iterate over the data array and create rows for each entry. For each row, create cells for "Name", "Age", and "Occupation" and populate them with the corresponding data.
7. Save the workbook to a file using workbook.write(fileOut).
8. Catch any IOException that might occur and print the stack trace.

Algorithm: DriverManager

1. Create a class DriverManager with static methods.
2. Implement the initializeDriver method to set up the Chrome driver with desired capabilities using WebDriverManager.chromedriver().capabilities(options).create().
3. Implement the getDriver method to return the initialized driver instance or initialize it if null.
4. Implement the quitDriver method to quit the driver if it is not null.

Algorithm: FileUtils

1. Create a class FileUtils with a static method readTestData.
2. The method should take the file path and sheet name as inputs.
3. Use Apache POI to read the data from the Excel file and store it in a 2D array.
4. Return the 2D array as the test data.