# Electro – Final Project

**Project Client:** *Title here*

**Project Folder:** *Link here*

**Project Mentor:** *Name here*

## Description

**What we will go through:**

* The main outcome of this Final Project is to go through what we have learned so far from both OOP and Selenium.
* Let’s break down key subjects that we will test with this project:



**The steps:**

1. **First, let’s navigate to the homepage of our Project site:**

* <https://electro.madrasthemes.com/>

A screenshot of a website

Description automatically generated

1. **Next, from the secondary menu choose the following:**

* Trending Styles

A screenshot of a computer

Description automatically generated

1. **On this step, we need to add the following product to Wishlist:**

* Purple Solo 2 Wireless

**A screenshot of a website

Description automatically generated**

1. **After we have added the product to Wishlist, we need to *navigate to our Wishlist page*:**

**A screenshot of a computer

Description automatically generated**

1. **Once we opened our Wishlist, we need to confirm product was added successfully:**

A screenshot of a computer

Description automatically generated

1. **From the Wishlist, we need to go back to Homepage:**

A screenshot of a website

Description automatically generated

1. **With this step, we will navigate to Contact page:**

**A map with a location pin

Description automatically generated**

1. **On this page, we will fill in message details and send message:**

**A screenshot of a computer

Description automatically generated**

1. **Final step would be to verify message was successfully sent:**

**A screenshot of a map

Description automatically generated**

**NOTES:**

* **Follow Page Object Model design for the project.**
* **Following POM, you will implement the inheritance concept from OOP.**
* **Use TestNG for annotations and test scenarios.**
* **Use Assertions for confirmation.**
* **Think of generic implementation of classes and methods.**
* **Avoid usage of the Thread.sleep method.**

Level 1: Beginner

Beginner project requirements

**In this beginner-level project, the focus is on getting familiar with basic Selenium functionalities, Page Object Model (POM) design, and implementing simple test scenarios. As a student, you will be expected to:**

**Navigate to the Homepage:**

* Open the project site (https://electro.madrasthemes.com/) using Selenium WebDriver.
* Verify that the homepage is successfully loaded.

**Access Trending Styles:**

* Navigate to the Trending Styles section from the secondary menu.

**Add Product to Wishlist:**

* Select the "Purple Solo 2 Wireless" product.
* Add the selected product to the Wishlist.

**Navigate to Wishlist:**

* Move to the Wishlist page.
* Confirm Product Addition:
* Verify that the "Purple Solo 2 Wireless" product is successfully added to the Wishlist.

**Return to Homepage from Wishlist:**

* Go back to the homepage from the Wishlist page.
* Navigate to Contact Page:
* Open the Contact page.

**Send Message:**

* Fill in the message details on the Contact page.
* Trigger the message sending functionality.

**Verify Message Sent:**

* Confirm that the message was successfully sent.

**Implement Page Object Model (POM):**

* Organize your code following the Page Object Model design.

**Implement Inheritance:**

* Utilize inheritance concepts from Object-Oriented Programming (OOP) in your test scenarios.

**Use TestNG for Annotations and Test Scenarios:**

* Structure your test cases using TestNG annotations.
* Ensure that your test scenarios are well organized.

Level 2: Intermediate

Beginner project requirements + Intermediate project requirements

**Use Assertions for Confirmation:**

* Implement assertions to confirm expected outcomes in your test cases.

**Think of Generic Implementation:**

* Develop generic classes and methods for better maintainability.

**Create a test suite.**

Level 3: Advanced

Beginner project requirements + Intermediate project requirements + Advanced project requirements

**Avoid Usage of Thread.sleep:**

* Demonstrate the use of implicit or explicit waits instead of the Thread.sleep method for handling synchronization issues.

**Handle Dynamic Elements:**

* Address scenarios where elements on the page are dynamic.

### Project Goals & Objectives:

**Selenium Basics:**

* Understand the fundamentals of Selenium WebDriver.
* Demonstrate the ability to navigate to a web page using Selenium.

**Page Object Model (POM) Design:**

* Grasp the concept of Page Object Model (POM) and its advantages in test automation.
* Implement POM to organize and structure the code effectively.

**Object-Oriented Programming (OOP) Principles:**

* Apply inheritance concepts from OOP in the context of test automation.
* Design classes and methods following OOP principles for maintainability.

**TestNG Annotations and Scenarios:**

* Familiarize yourself with TestNG annotations and their significance.
* Structure test scenarios using TestNG to ensure orderly execution.

**Assertions for Confirmation:**

* Understand the importance of assertions in validating expected outcomes.
* Implement assertions to confirm the success or failure of test cases.

**Generic Implementation:**

* Design and implement generic classes and methods to enhance code reusability.

**Handling Dynamic Elements:**

* Learn strategies for handling dynamic elements on web pages.

**Explicit and Implicit Waits:**

* Differentiate between explicit and implicit waits and apply them appropriately to handle synchronization issues.

**Parameterization of Test Data:**

* Implement parameterization techniques for test data to enhance flexibility.

**Project Structure and Organization:**

* Organize the project structure effectively for easy maintenance and scalability.

### Project Step-by-Step Procedural Information:

**Step 1: Set Up Your Project Environment**

* Create a new Java project in your preferred IDE.
* Set up a Maven or Gradle project for dependency management.
* Add Selenium WebDriver and TestNG dependencies to your project.

**Step 2: Create Base Classes**

* Implement a base class to initialize the WebDriver and set up common configurations.
* Create a base Page Object class for shared methods and elements across pages.

**Step 3: Navigate to Homepage**

* Write a test case to open the project site (https://electro.madrasthemes.com/).
* Use WebDriver methods to verify that the homepage is successfully loaded.

**Step 4: Access Trending Styles**

* Implement a method to navigate to the Trending Styles section from the secondary menu.
* Write a test case to verify the correct navigation and the presence of Trending Styles.

**Step 5: Add Product to Wishlist**

* Create a Page Object for the product page.
* Implement a method to add the "Purple Solo 2 Wireless" product to the Wishlist.
* Write a test case to confirm the successful addition to the Wishlist.

**Step 6: Navigate to Wishlist**

* Create a method to navigate to the Wishlist page.
* Write a test case to ensure the correct navigation to the Wishlist.

**Step 7: Confirm Product Addition in Wishlist**

* Implement a method to confirm that the product is present in the Wishlist.
* Write a test case to verify the successful addition of the product in the Wishlist.

**Step 8: Return to Homepage from Wishlist**

* Implement a method to go back to the homepage from the Wishlist page.
* Write a test case to confirm the correct return to the homepage.

**Step 9: Navigate to Contact Page**

* Create a method to navigate to the Contact page.
* Write a test case to ensure successful navigation to the Contact page.

**Step 10: Fill in Message Details and Send Message**

* Implement methods to fill in the message details on the Contact page.
* Create a method to trigger the message sending functionality.
* Write a test case to confirm the successful sending of the message.

**Step 11: Verify Message Sent**

* Implement a method to verify that the message was successfully sent.
* Write a test case to confirm the successful message delivery.

**Step 12: Implement Page Object Model (POM)**

* Organize your project structure following the Page Object Model design.
* Create separate classes for each page with related methods and elements.

**Step 13: Implement Inheritance from OOP**

* Utilize inheritance concepts from Object-Oriented Programming in your Page Object classes.
* Ensure that common functionalities are implemented in base classes.

**Step 14: Use TestNG for Annotations and Test Scenarios**

* Utilize TestNG annotations such as @Test, @BeforeMethod, and @AfterMethod.
* Structure your test scenarios in an orderly manner using TestNG.

**Step 15: Use Assertions for Confirmation**

* Implement assertions to verify expected outcomes in your test cases.
* Use TestNG assertions or Java assertions as appropriate.

**Step 16: Think of Generic Implementation**

* Design generic methods and classes for improved code reusability.
* Consider parameterization for methods where applicable.

**Step 17: Avoid Usage of Thread.sleep**

* Implement explicit or implicit waits to handle synchronization issues.
* Refactor any existing code that uses Thread.sleep for better efficiency.

### Project Deliverables

**Coding Environment:**

* Write the automation scripts in Java using IntelliJ IDEA.
* Utilize Selenium for web automation and TestNG for test orchestration.

**Project Export:**

* Export the entire project from IntelliJ IDEA, including source code, configurations, and dependencies.
* The project must be zipped.

**Hosting on Drive:**

* Upload the exported project to a cloud storage platform (e.g., Google Drive).
* Ensure that the uploaded project includes all necessary files and folders.

**Sharing Link on Brainster Learn Platform:**

* Share the link on the Brainster Learn platform for project submission.

Is the project presented in front of a client: NO

## Evaluation system

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| ***Criteria\*\**** | Excellent  10p | Proficient  8p | Good  6p | Fair  4p | Poor  2p |
| Usage of Page Object Model | *Successfully implements Page Object Model for all mentioned steps with clear structure and organization.* | *Implements Page Object Model for most mentioned steps with minor inconsistencies or deviations.* | *Implements Page Object Model for some mentioned steps but lacks consistency or clear structure.* | *Partially implements Page Object Model with significant inconsistencies or deviations.* | *Does not implement Page Object Model.* |
| Navigation to Homepage, Trending Styles, and Wishlist | *Successfully navigates to the homepage, Trending Styles page, adds the product to Wishlist, and navigates to the Wishlist page without errors.* | *Navigates to the required pages and adds the product to Wishlist with minor issues or inefficiencies.* | *Navigates to the required pages and adds the product to Wishlist, but with noticeable delays or errors.* | *Attempts to navigate and add the product to Wishlist, but with significant issues or incomplete execution.* | *Fails to navigate or add the product to Wishlist.* |
| Confirmation of Product Addition and Navigation to Contact Page | *Successfully confirms product addition on the Wishlist page and navigates to the Contact page without errors.* | *Confirms product addition on the Wishlist page and navigates to the Contact page, but with minor issues or incomplete execution.* | *Confirms product addition on the Wishlist page and navigates to the Contact page, but with noticeable delays or errors.* | *Attempts to confirm product addition on the Wishlist page and navigate to the Contact page, but with significant issues or incomplete execution.* | *Fails to confirm product addition on the Wishlist page or navigate to the Contact page.* |
| Avoiding usage of Thread.sleep method | *Does not use the Thread.sleep method in any part of the test suite.* | *Minimally uses the Thread.sleep method, only in exceptional cases where alternatives are not feasible.* | *Uses Thread.sleep method in several instances without exploring alternative solutions.* | *Frequently relies on the Thread.sleep method instead of exploring alternative solutions.* | *Heavily relies on the Thread.sleep method throughout the test suite.* |

**Total Points: 40**

## Deadline

7 days after its presentation, at 23:59 (end of the day).

## Assessment Rules

* Fair Assessment: ethical considerations
  + Assessors should ensure that the assessments are conducted in a fair and ethical manner, respecting the principles of academic integrity and honesty.
* Reliability and Validity: enabling consistency
  + Assessments should be consistent and reliable, meaning that they yield consistent results when applied repeatedly to the same task or performance.
  + Assessments must accurately gauge the knowledge, skills, or abilities they are intended to evaluate, ensuring their validity as indicators.
* Feedback: as a method for continuous improvement
  + Assessors should offer constructive feedback that identifies strengths and areas for improvement. The Feedback should be specific and actionable, it should include thought provoking guides and should challenge the student to become better at a specific task.
* Late Submission Policy: assessing assignments after their deadline
  + Students should be allowed a grace period of 3 days (72 hours) to make a late submission on any assignment, with the notice that they will be deducted 20% from the total possible points.
* Plagiarism Policy: assessing assignments with matching solutions
  + In the event of suspected plagiarism, the assessor is required to promptly collaborate with the Student Experience Coordinator/Team as the initial step. Together, they will draft a notice to remind students of the strict prohibition against plagiarism, with potential repercussions for recurrent violations. The following actions will be considered in cases of repeated plagiarism:
    - If a submitted solution exhibits substantial similarity, exceeding 60%, with another student's work (individually or within a group), the respective challenge/project will incur a 50% reduction from the maximum points attainable.
    - In cases where a solution is identified as more than 90% identical to another student's work (individually or within a group), the project/challenge in question will receive a score of 0 points.
  + The use of generative AI is encouraged as a learning tool in our educational programs; however, students must engage with the material and contribute with original thought. Reliance on AI for complete content generation is strictly prohibited and will result in point deductions, official warning, or other academic penalties.
  + Upon completion of the assessment process for each challenge/project, the assessor is tasked with selecting the most complete, optimal, and creative solution and to showcase it by publishing it on the platform together with the assessment results.
* Timeliness: timeframe for delivering results and feedback to students
  + Feedback on challenges should be provided within 7 days after the deadline has passed.