## **Headless Browser Testing using Selenium.**

## **By**

## **Biyyapu Tripura Amulya**

## **Batch Code - 2023-11124**

# **Enrolment Number-EBGNTR0224135036**

**INDEX**

* Acknowledgement
* Introduction
* Objectives
* Existing System
* Tools and Technologies
* Performing the Program
* Screenshots
* Conclusion

# **ACKNOWLEDGEMENT**

I would like to express my sincere gratitude to all those who have contributed to the successful completion of this **Headless Browser using Selenium** testing project.

First and foremost, I am deeply thankful to **Mr. Santhosh Kumar**, whose guidance and support were invaluable throughout the project. Their expertise and insights provided the direction needed to navigate challenges and achieve our objectives.

I extend my appreciation to my peers and colleagues, whose dedication and collaboration were instrumental in conducting thorough tests and ensuring the reliability of our automation scripts.

Special thanks are to **EDUBRIDGE** who provided technical assistance and valuable feedback that significantly enhanced the quality of our testing framework.

This project would not have been possible without the collective effort and commitment of everyone involved. Thank you all for your contributions.

Amulya.

# **INTRODUCTION**

**What is Selenium?**

Selenium is an open-source, automated testing tool used to test web applications across various browsers like Firefox, Chrome, Opera, and Safari, and these tests can be coded in several programming languages like Java, Python, Perl, PHP, and Ruby.

**What is headless browser testing in Selenium?**

It actually is what it sounds like. Headless testing is when you run a UI-based browser test without showing the browser UI. It's running a test or running a script against a browser, but without the browser, UI starts up in the background.

**What is TestNG?**

**TestNG** is an automation testing framework where NG stands for “Next Generation”. TestNG is inspired by JUnitwhich uses the annotations (@). TestNG overcomes the limitations of JUnit and is designed to make end to end testing easy. Using TestNG, you can generate a proper report, and you can easily come to know how many test cases are passed, failed, and skipped. You can execute the failed test cases separately.

# **OBJECTIVES**

This project aims to automate the testing of the drag-and-drop and login functionality of a web application using Selenium WebDriver with Java. The primary goals are:

* **Drag and Drop functionality in Selenium:**

This action is performed using a mouse when a user moves (drags) a web element from one location and then places (drops) it at another point. This is a common action used in Windows Explorer when moving a file from one folder to another.

* **Login Functionality in Selenium:**

The login functionality is a critical component of most web applications, serving as the gateway to access secure user-specific features and data. Ensuring the reliability, security, and efficiency of the login process is essential for maintaining user trust and protecting sensitive information. Manual testing of login functionality can be time-consuming and prone to human error, which is why automating this process is highly beneficial.

**EXISTING SYSTEM**

In the existing system, the drag and drop, login functionality of the web application is tested manually. This process involves testers manually as below

* **Drag and drop functionality** 🡪 Involves testers to manually perform left click on the source element, hold the click to hold the source element, moves to the location of the target element and then releases the mouse click.
* **Login Functionality** 🡪 Involves testers to manually entering various sets of user credentials into the login form to verify if the login process works correctly. Manual testing can be time-consuming, prone to errors, and is not efficient for validating multiple test scenarios. Automating this process can significantly improve testing efficiency and accuracy.

# **TOOLS & TECHNOLOGIES**

* Selenium WebDriver
* Java
* TestNG
* Maven
* IDE (e.g., IntelliJ IDEA, Eclipse)

**PERFORMING THE PROGRAM**

**Step 1:**Create a Maven Project

**Step 2:**Add Dependencies

Open the pom.xml file in the project and add dependencies for TestNG and Selenium WebDriver.

**Step 3:**Set up a Java Project

If you are not using Maven you can set up a Java project by creating a new Java project and adding TestNG and Selenium WebDriver libraries to your IDE

**Step 4:**Create a Test Group

Now you can start creating a test group. You can organize your tests as **‘@Test’, ‘@BeforeTest’, ‘@AfterTest’**etc. Publish with TestNG annotations. and write Selenium WebDriver code in this process to run your tests.

**Step 5:** Write TestNG XML Configuration

Create a TestNG XML configuration file **(‘testng.xml’)**where you can define your test suites, test classes, and any parameters.

**Step 6:**Run Tests

**Step7:**View Results

After you complete the test, you can view the test results in the console output or in the HTML report generated by TestNG.

# **SCREENSHOTS**

Please find the screenshots below.

* Drag And Drop and Login functionality.

A screenshot of a computer program

Description automatically generated

A screen shot of a computer

Description automatically generated

# **CONCLUSION**

Headless browser testing with Selenium offers a powerful, efficient approach to test automation, especially suited for specific scenarios like performance testing. Headless browser testing reduces human effort and makes the work faster by eliminating graphical user interface (GUI) rendering.

**THANK YOU!!!**