Automation Assignment: Saucedemo.com

# Objective

Create automated tests for the Saucedemo.com website using Selenium, Junit, and Java. Use Selenium Grid to run tests on Firefox and Chrome on your local machine. The email and password should be passed as data, and tests should be run based on tags.

Website: [https://www.saucedemo.com](https://www.saucedemo.com/)

#### **Description:**

This project automates the testing of key e-commerce functionalities, such as:

* Adding items to the cart
* Proceeding to checkout
* Completing the checkout process
* Verifying the confirmation message

#### **Automation Testing-Design Pattern:**

The ***Page Object Model (POM)*** is a design pattern commonly used in automation testing to enhance maintainability, readability, and reusability of test scripts. It separates the test scripts from the application's user interface (UI) by creating an abstraction layer for the web pages in the application. Each web page is represented as a class, and the elements on that page are defined as variables within that class. Actions performed on the elements are implemented as methods in the same class.

### ***Structure of POM:***

1. **Page Classes**: Define locators and methods for specific pages.
2. **Test Classes**: Contain the test logic and call methods from page classes.
3. **Utilities**: Additional reusable code (e.g., utilities for reporting, wait conditions).

### ***Prerequisites***

1. **Java Development Kit (JDK)**: Ensure JDK 8 or later is installed.
2. **Maven**: Install Maven for dependency management.
3. **WebDriver**: Download and configure the ChromeDriver.
4. **IDE**: Use IntelliJ IDEA.
5. **Browser**: Google Chrome (latest version)
6. **Technology stack**: Selenium, Jnit,Java

#### **Test Coverage**

The following test cases have been automated:

1. Add items to the cart and proceed to checkout.
2. Complete the checkout process with valid user details.
3. Verify the purchase confirmation message.

#### **Challenges Faced**

1. **Dynamic Locators**: Adjusted locators to align with changes in the webpage structure.
2. **Wait Times**: Implemented explicit waits to handle slow-loading elements.
3. **Data Management**: Centralized configuration in the config.properties file for flexibility.