Automation Test Plan for Selenium, Cucumber, Java, and Maven

# 1. Objective

Define the scope, objectives, and approach for automating tests using Selenium, Cucumber, Java, and Maven. Ensure robust and maintainable test scripts that align with the overall testing strategy.

# 2. Scope

In-Scope: Automated testing of web application features including functional, regression, and integration tests.  
Out-of-Scope: Performance testing, security testing, and non-web-based application tests.

# 3. Test Environment

Browsers: Chrome, Firefox, Edge  
Operating Systems: Windows, macOS, Linux  
Test Frameworks: Selenium WebDriver, Cucumber, JUnit/TestNG  
Build Tool: Maven  
CI/CD Tool: Jenkins (or any other CI tool used)  
Version Control: GitHub/GitLab/Bitbucket for code repository management.

# 4. Test Strategy

Automation Framework:  
- Page Object Model (POM): Structure to manage web page elements and actions.  
- Behavior-Driven Development (BDD) with Cucumber: Writing tests in Gherkin language.  
Test Cases:  
- Prioritize test cases based on risk and business impact.  
- Identify reusable test components.  
Data Management:  
- Use external files (CSV, Excel, or JSON) or a database for test data.  
- Data-driven testing to cover different input scenarios.  
Test Execution:  
- Tests to be executed on various browsers and operating systems.  
- Integrate with CI/CD pipeline for automated test execution on code commits.

# 5. Tools and Technologies

Selenium WebDriver: Automating browser actions.  
Cucumber: BDD tool to define test cases in a human-readable format.  
Java: Programming language for writing test scripts.  
Maven: Build automation tool for managing dependencies and executing tests.  
Jenkins: For CI/CD integration.  
Git: Version control system for managing code repository.