Automation Testing Project Report

Project Name

Automation Framework for Testing Croma E-Commerce Website

Application / Module Name

Croma – E-Commerce Web Application

---

1. Project Overview

This project involves automating the testing of the **Croma e-commerce website** using **Selenium WebDriver**, **TestNG**, and **Maven**. The automation framework is designed to validate key business functionalities such as product search, filtering, sorting, add-to-cart operations, checkout flow, and user authentication modules.
By automating regression scenarios, the framework significantly minimizes manual testing efforts, improves execution accuracy, and ensures application stability across releases.

---

2. Scope of Automation

Modules Automated

- Product Search
- Brand-based Filtering
- Sorting Products by Discount
- Price Validation and Calculation
- Add to Cart Functionality
- Checkout Process
- User Login and Registration

Out of Scope

- Payment Gateway Validation
- CAPTCHA / OTP Authentication
- Mobile Responsiveness Testing

---

3. Tools & Technologies

- **Programming Language:** Java (JDK 17)
- **Automation Tool:** Selenium WebDriver
- **Testing Framework:** TestNG
- **Build Tool:** Maven

- **Design Pattern:** Page Object Model (POM)
- **Reporting:** Extent Reports
- **Data Handling:** Apache POI
- **Logging:** Log4j2
- **IDE:** Eclipse / IntelliJ IDEA
- **Browsers:** Chrome, Firefox, Edge
- **Version Control:** GitHub
- **CI/CD (Optional):** Jenkins

---

4. Framework Architecture

The automation framework is built using the **Page Object Model (POM)** approach to improve code reusability and maintainability. Common utilities are implemented for browser management, test data handling, reporting, and synchronization.

Project Structure

- src/main/java – Page classes and utility classes
- src/test/java – Test case implementations
- testdata – Excel files for test data
- reports – HTML reports and failure screenshots
- pom.xml – Maven dependency configuration

---

5. Test Scenarios & Test Cases

Module	Test Case ID	Description	Status
Search	TC_01	Verify product search for “Refrigerator”	Passed
Filter	TC_02	Apply brand filters (Samsung, LG, Whirlpool)	Passed
Sorting	TC_03	Sort products by discount (High → Low)	Passed
Price	TC_04	Calculate average price of top 10 discounted products	Passed
Cart	TC_05	Add product to cart	Passed
Checkout	TC_06	Validate checkout flow	Passed
Login	TC_07	Verify user login	Passed
Register	TC_08	Verify new user registration	Passed

---

6. Execution & Results

Execution Summary

- **Total Test Cases:** 40
- **Passed:** 38
- **Failed:** 2
- **Skipped:** 0

Reports Generated

- Extent HTML Report
- TestNG Execution Report
- Execution Logs
- Screenshots for failures

Tests were executed using Maven command mvn test across multiple browsers.

---

7. Defects / Issues Identified

Defect ID	Description	Status
DEF-301	Brand filter not applied for multiple selections	Fixed
DEF-302	Checkout delay in Edge browser	Open
DEF-303	Minor UI alignment issue	Fixed

---

8. Challenges & Solutions

- **Dynamic Web Elements:** Handled using robust XPath strategies and explicit waits
- **Synchronization Issues:** Implemented FluentWait and explicit waits
- **Browser Driver Management:** Automated using WebDriverManager
- **Test Data Handling:** Implemented Excel-based data handling using Apache POI
- **Reporting:** Integrated Extent Reports with screenshots for failed cases

---

9. ROI / Benefits of Automation

- Reduced manual testing effort by approximately **70%**
- Improved accuracy and reliability of test execution
- Faster regression testing cycles
- Reusable and maintainable automation code

- Enhanced visibility through detailed HTML reports

---

10. Conclusion

The Croma Selenium automation framework effectively validates major e-commerce workflows with high reliability. Its modular, scalable design supports continuous testing and ensures long-term maintainability, making it suitable for enterprise-level regression testing.

---

11. Future Enhancements

- CI/CD integration using Jenkins
- Mobile automation using Appium
- API testing with Rest Assured
- Performance testing using JMeter
- AI-based smart element locators
