# Software User Manual for automation for Automation and Manual testing Project

## 1. Table of Contents



## 1. Introduction

### 1.1 Overview

This document provides user manuals for two distinct testing projects:

1. **Manual Testing Project**: Focuses on executing test cases manually to validate the core functionalities of the Universal Service Portal. Manual testing is designed to check how the system behaves from an end-user's perspective, ensuring the user interface, workflows, and business logic meet expected outcomes.
2. **Automation Testing Project**: Covers the automated testing process for SmartShop . This project aims to automate critical functionalities, including user registration, login processes, landing page navigation, and wishlist management, ensuring a seamless shopping experience for customers.The project uses automation tools like Selenium and TestNG to execute repetitive tests, such as regression tests, efficiently. Automated tests ensure the system's functionality remains stable across different builds and releases.

Each project has its own unique processes and tools. This manual is split into two main parts: one dedicated to **Manual Testing** and the other to **Automation Testing**.

### 1.2 Scope

This manual provides separate instructions for setting up, executing, and reporting on manual and automated tests. It is divided into two sections:

- **Universal service portal (Manual Testing)**: Includes test case creation through manual processes. This section is ideal for testers focusing on end-user interactions and edge cases that need human observation.
- **SmartShop (Automation Testing)**: Focuses on setting up and executing test scripts using automation tools. The section also covers maintaining automation scripts, running test suites, and generating reports in testing and cucumber design .

The scope of each project includes:

**Universal serve portal (Manual Testing):**

1. **Product Filtering Feature**:
    - Implement a product filtering system that allows customers to sort products by categories (Games, Shoes, Toys).
    - Provide detailed instructions for users on how to apply filters effectively to find products that match their interests.
    - Include user interface guidelines and expectations for displaying filtered results.

2. **Online Passport Application**:
    - Develop a feature for citizens to apply for a new passport online, eliminating the need to visit the passport office in person.
    - Outline the step-by-step process for users to fill out the application, upload required documents, and submit their requests.
    - Detail the verification process and expected time frames for application processing.

3. **Online Appointment Scheduling**:
    - Create a system for patients to schedule appointments with their doctors online.
    - Describe how users can view available time slots, select their preferred appointment times, and receive confirmations.
    - Include information on rescheduling or canceling appointments as needed.

4. **User Roles and Permissions**:
    - Define the different user roles (customers, citizens, patients) and their access levels within the portal.

**SmartShop project (Automation Testing)**:

1. **User Registration and Login**:

- Automate the testing of user registration and login processes, ensuring that users can create accounts, log in successfully, and recover passwords when necessary.
- Include tests for validation messages for input errors, such as invalid confirmation password formats

2. **Landing Page Testing**:
   - Automate the verification of the landing page's functionality, including loading times, navigation links, and the display of featured products.
   - Test various scenarios, such as different screen resolutions and device types, to ensure the page is responsive and user-friendly.

3. **Wishlist Functionality**:
   - Automate the tests for the wishlist feature, ensuring users can add, remove, and view items in their wishlist.
   - Verify that the wishlist persists across user sessions and correctly reflects items added by the user

4. **Shopping Cart Operations**:
   - Automate tests for adding, updating, and removing items from the shopping cart.
   - Validate the correct calculation of totals, including taxes and discounts, during the checkout process.

5. **Reporting and Documentation**:
   - Generate reports for automated test executions by allure report, highlighting pass/fail rates, defect logs, and test coverage.
   - Document test cases, scripts, and configurations for future reference and maintenance.

**Benefits of the Project Scope**

- **Improved Efficiency**: Automation will significantly reduce the time required for regression testing, enabling faster release cycles.
- **Consistency**: Automated tests ensure consistent execution and accuracy, reducing human errors.
- **Enhanced Coverage**: The automation suite can cover a wide range of test cases that may be time-consuming to perform manually.
- **Increased Confidence**: Regular automated testing increases confidence in the software's stability and functionality.

**1.3 Audience**

This manual is intended for two types of users:

- **Manual Testing Team**: QA engineers who execute test cases manually, log defects, and verify UI and workflow functionality.

- **Automation Testing Team**: Engineers who are responsible for maintaining, running, and interpreting the results of automated test scripts.

## 2. System Requirements

**2.1 System Requirements for Universal service portal project :**

**Hardware Requirements**:

- **Computer**: A PC or laptop with a minimum of:
  - Processor: Dual-core or better (e.g., Intel i5 or AMD equivalent).
  - RAM: At least 8 GB (16 GB recommended for multitasking).
  - Storage: Sufficient free disk space (at least 100 GB) for test data, logs, and tools.

**Software Requirements**:

- **Operating System**: Windows 10
- **Web Browser**: Latest versions of popular browsers Chrome for testing compatibility.
- **Test Management Tools**: Access to tools like Jira for managing test cases and defect tracking.
- **Document Tools**: Software like Microsoft Office for documentation test plans
- **Communication Tools**: Microsoft Teams for team collaboration.

**Network Requirements**:

- Stable internet connection for accessing the e-commerce platform and online resources.

**2.3 System Requirements for SmartShop project :**

**Hardware Requirements**:

- **Computer**: A PC or laptop with a minimum of:
  - Processor: Quad-core or better (e.g., Intel i7 or AMD equivalent).
  - RAM: At least 16 GB (32 GB recommended for running multiple tests simultaneously).
  - Storage: Sufficient free disk space (at least 200 GB) for storing test scripts, logs, and tools.

**Software Requirements**:

- **Operating System**: Windows 10 .
- **Automation Testing Frameworks**: Installation of frameworks like Selenium, TestNG or Cucumber, depending on your automation strategy.
- **Programming Environment**: IDEs like Eclipse , IntelliJ IDEA, or Visual Studio Code for writing and managing test scripts Java.
- **Web Browsers**: Latest versions of browsers  Chrome with respective WebDriver installed.

**Network Requirements**:

- Stable internet connection for accessing cloud-based services or repositories.

**Access Requirements**:

- User accounts with necessary permissions to execute automated tests on the Luma website.

**Dependencies**:

- Third-party libraries and tools (e.g., Allure for reporting, Maven for build management) may need to be installed and configured.
- Any specific browser extensions required for automation (e.g., for handling CAPTCHA or pop-ups).

## 3. Installation and Setup Instructions

**Installation Steps for Universal service portal (Manual testing)** :

- **Install a Web Browser**: Download and install your preferred web browser.
- **Set Up Test Management Tool**: Create an account in jira and project for Universal service portal.
- **Create Test Plan**: Document the testing scope, objectives, and schedule.
- **Document Test Cases**: Write test cases in the chosen test management tool.
- **Environment Setup**: Access the Universal service portal application and ensure necessary login credentials.

### Automation Testing Setup for SmartShop

 **Installation Steps**:

- **Install Java JDK**: Download and configure JDK.
- **Install Maven**: (Optional) Download and configure Maven.
- **Set Up IDE**: Install and configure IntelliJ IDEA or Eclipse.
- **Create a New Maven Project**: Add required dependencies in pom.xml.
- **Set Up Selenium WebDriver**: Download the appropriate WebDriver (ChromeDriver).
- **Create Automation Framework**: Organize project structure and implement test classes using TestNG and Selenium.

 **Example Test Class**: Implement a simple automation test using Selenium WebDriver.

 **Running Automation Tests**: Execute tests from the IDE or via Maven with mvn test.

## 4. Automation test execution

***1. Prepare Your Automation Environment***

- **Verify the Setup**:
  - Ensure the automation framework (Selenium, TestNG) is properly set up in your IDE (IntelliJ IDEA, Eclipse).
  - Ensure all necessary dependencies are added to your project ( Selenium WebDriver, TestNG, Apache POI for data-driven tests).
- **Set Up the WebDriver**:
  - Ensure you have the correct WebDriver for your browser (ChromeDriver for Google Chrome).
  - Set the path for the WebDriver in your test scripts.
- **Access the Application**:
  - Make sure the SmartShop application is accessible in the environment (staging or development) where you will run your tests.

***2. Execute Automation Tests***

1. **Open Your IDE**:
   - Launch your IDE (IntelliJ IDEA or Eclipse) where your automation test scripts are located.
2. **Select the Test Class/ Suite**:
   - Choose the test class or suite you want to execute. You can run individual test cases or the entire suite based on your needs.
3. **Monitor Test Execution**:
   - Watch the console output in your IDE to monitor the test execution process. Look for any errors or failures during execution.

***3. Review Test Results***

- **Check Execution Reports**:
    - After the tests run, check the output report generated by your testing framework (TestNG).
    - Look for the number of tests executed, passed, failed, and skipped.
- **Review Logs and Screenshots**:
    - Review any logs or screenshots generated during the test execution, especially for failed tests. This information can help diagnose issues.

***4. Report Defects***

- **Log Defects**:
    - Use your bug tracking tool (allure reports) to log defects with relevant details:
        - Title and description of the defect
        - Expected
        - Screenshots or logs

## *5. Test Cases and Scenarios*

***Test Scenarios Universal service portal***

**Key Test Scenarios for Universal service portal for customer :**

1. No categories selected
2. Handling No Products in a Category
3. Verify Behavior When No Categories Are Selected
4. Select Multiple Categories from the List
5. Visible Selected Filters
6. Behavior Resulting from complex interactions
7. Behavior resulting from multiaple interactions
8. Display Product Categories
9. Select a Single Category from the List
10. Clear Category Selection
11. Dynamic Product Count Updates
12. Filtering Functionality on Desktop Devices
13. Filtering Functionality on Mobile Devices
14. Verify interaction with other filters such as Price Rang

**Key Test Scenarios for Universal service portal for citizen :**

1. Enter Valid Personal Information
2. Form Submission During Network Failure
3. Invalid Reference Number for Tracking Application
4. Save Progress Beyond 48 Hours
5. Save Progress and Continue Later
6. Invalid Personal Information Validation
7. Real-Time Form Validation for Missing/Invalid Fields
8. Invalid Document Upload Validation

9. Upload Valid Documents
10. Submit Completed Application
11. Receive Confirmation Email with Reference Number
12. Track Application Status with Reference Number
13. Check Application Status - Pending
14. Check Application Status - In Progress
15. Check Application Status – Ready
16. Pay Application Fee Securely
17. Invalid Payment Details
18. Access Passport Application Form

**Key Test Scenarios for Universal service portal for patient :**

1. Incorrect Phone Number
2. Invalid Date Selection
3. Overlapping Appointments
4. Different time zone booking
5. Multiple booking within 24 Hours
6. Successful Appointment Booking
7. Appointment Rescheduled or cancellation
8. Doctor Unavailability
9. Confirmation Email
10. Confirmation SMS
11. Appointment details
12. Notify patient and doctor of appointment change
13. No cancellation after 24 Hours
14. Error in booking an appointment

***Test Scenarios SmarShop***

1. **User Registration**:
    - Verify successful registration with valid details.
    - Verify successful registration with invalid confirmation password.
    - Check the same value of password in required fields
    - Check error messages for missing required fields.
2. **User Login**:
    - Verify login functionality with valid credentials.
3. **Shopping Cart**:
    - Verify that the cart total updates correctly.
4. **Checkout Process**:
    - Ensure smooth navigation through the checkout process.
    - Validate entry of shipping and payment information.
    - Confirm order confirmation is displayed after payment.
5. **Wish List**:

- Test adding products to the wish list.
- Verify adding products within guest user
- Validate viewing and managing wish list items.

## 6. Generate Report

- **Automation Test Execution Report** :
  - Compile a report summarizing the execution results:
    - DataProvider : 6 of tests executed .  5 passed / 1 failed
    - TestNG : 9 of tests executed . 9 passed
    - Cucumber : 9 of tests executed . 9 passed
- **Share Results**:
  - Share the test execution report with stakeholders, including project managers and development teams.

## 7.Best Practices

1. Maintain Clear Test Case Documentation
2. Follow the POM Pattern
3. Check elements locators
4. Use Allure Annotations
5. Implement Data_Driven Testing (DataProvider Testing)
6. Run tests in isolation

## 8. FAQ

This section includes frequently asked questions this can serve as a useful resource for team members, stakeholders, or anyone involved in the testing process..

**FAQs for Automation Testing**

**1. What is Automation Testing?**

- **Answer**: Automation testing involves using specialized tools and scripts to execute tests on software applications, significantly speeding up the testing process and reducing human error.

2. **What are the benefits of Automation Testing?**

- **Answer**: Benefits include increased test coverage, faster execution, repeatability, improved accuracy, and the ability to run tests unattended during off-hours.

3. **When should I use Automation Testing?**

- **Answer**: Automation is ideal for regression tests, performance testing, load testing, and tests that need to be run repeatedly across different versions of the application.

**4. Which tools are commonly used for Automation Testing?**

- **Answer**: Popular automation tools include Selenium, TestNG, Cucumber and Allure for reporting.

**5. How do I choose which tests to automate?**

- **Answer**: Focus on high-frequency tests, tests that require extensive data input, tests that are prone to human error, and tests that need to be run across multiple environments.

**6. What is a Data-Driven Testing approach?**

- **Answer**: Data-Driven Testing involves executing the same test with multiple sets of input data, allowing for more extensive coverage and less redundancy in test scripts.

**7. Can automation testing be done on all types of applications?**

- **Answer**: Automation testing can be applied to web applications, mobile applications, and desktop applications, but the choice of tools may vary based on the application type.

**8. How do I generate reports in Automation Testing?**

- **Answer**: Tools like Allure, and TestNG provide built-in reporting features. You can configure these tools to generate detailed reports on test execution results.

**FAQs for Manual Testing**

**1. What is Manual Testing?**

- **Answer**: Manual testing is the process of manually executing test cases without automation tools to identify defects in software applications.

**2. What are the key benefits of Manual Testing?**

- **Answer**: Manual testing allows for exploratory testing, human observation, and immediate feedback on the user experience. It is useful for usability testing and one-off tests.

**3. When should I use Manual Testing?**

- **Answer**: Manual testing is best used for exploratory testing, usability testing, and when the application undergoes significant changes that require a human touch.

**4. How do I create effective test cases?**

- **Answer**: Ensure test cases are clear, concise, and cover both positive and negative scenarios. Include preconditions, steps to execute, expected results, and postconditions.

**5. What tools can assist with Manual Testing?**

- **Answer**: Tools such as JIRA can help manage test cases, track defects, and document test execution results.

**6. What is Exploratory Testing?**

- **Answer**: Exploratory testing is an unscripted testing approach where testers explore the application to identify defects based on their experience and intuition.

**7. What are some common challenges in Manual Testing?**

- **Answer**: Common challenges include time consumption, human error, inconsistent test execution, and difficulties in managing extensive test cases.

## 9. Conclusion

**For Manual Testing:**

- Manual testing allows for valuable exploratory testing and human observation that automated tests may overlook. By consistently updating test cases, soliciting stakeholder feedback, and fostering team training, manual testing processes can remain effective and relevant in a dynamic environment.

**For Automation Testing:**

 **TestNG**: Utilizing TestNG provides a robust framework that supports various testing methodologies, including unit, functional, and integration testing. Its features such as annotations, parallel execution, and comprehensive reporting empower testers to execute and manage test cases efficiently.
 **Page Object Model (POM)**: Implementing the POM design pattern promotes better organization and reusability of code. By encapsulating the page-specific methods and elements, POM simplifies test maintenance and improves readability. This separation of concerns makes it easier to manage changes in the UI, as updates to the page structure can be made in one place without affecting the test logic.
 **Cucumber**: Cucumber enables behavior-driven development (BDD), allowing stakeholders to write test scenarios in plain language. This fosters collaboration between technical and non-technical team members, ensuring that requirements are clearly understood and met. The ability to automate these scenarios enhances validation and promotes confidence in the software's functionality.
 **DataProvider**: The DataProvider feature in TestNG facilitates data-driven testing, allowing the same test to run with multiple data sets. This increases test coverage and efficiency while reducing redundancy. It enables testers to validate the application against various inputs, ensuring robustness in real-world scenarios.

## 10. Appendix

**A. Glossary of Terms :**

- **TestNG**: A testing framework inspired by JUnit, designed for test configuration and execution.
- **Page Object Model (POM)**: A design pattern that abstracts the UI elements and interactions of web pages into classes, promoting maintainability and reusability.
- **Cucumber**: A tool for behavior-driven development (BDD) that allows writing test scenarios in plain language (Gherkin syntax).
- **DataProvider**: A TestNG feature that allows parameterization of tests, enabling multiple sets of data to be used in a single test method.
- **Automation Testing**: The use of automated tools and scripts to execute test cases on a software application.

## 11. Contact Support

This section provides contact details for technical support, including email addresses, phone numbers, or links to help centers.

For any further assistance, contact our support team:

- Email: ayah51053@gmail.com
- Phone: +2 (01152607711)