1. **Software Test Plane=STP**

1.1 Web App- "Sauce Demo"

Version: 1.0

Created: 11/14/2025

Last Updated: 11/14/2025

Prepared by: Maya Ronen

2. Introduction

"Sauce Demo" is a sample web application for practicing QA testing. It simulates a full purchase flow: Login, browse products, add items to the cart and complete checkout.

3. Team members:

This QA assignment includes only the QA tester- Maya Ronen.

4. Support documents:

There is no SRS or design documents attached. (In the real-world links to SRS and designs would be provided here).

5. Test environment

5.1 Browser/Devices

- Web, Chrome (Most used browser according to gs.statcounter.com).
- iPhone 13 + Galaxy S25 (popular mobile devices. Due to time constraints focus on main UI tests).

5.2 Testing environment

- This exercise is conducted on production- public demo site; however, in the real world, tests are first run in a Staging/QA environment before being released to production.

6. Entry/Exit Criteria

6.1 Entry Criteria

- SRS and Design documentation**.**
- A stable build that is ready for testing**.**
- Full access to data, including permissions for testing and log investigation.

6.2 Release Criteria

- 0 Critical bugs
- Maximum 3 Medium bugs
- Maximum 7 Low bugs

(In the real world, this may be a decision made by the project manager and could even be documented in the SRS).

7. In Scope:

7.1 Login page

7.2 Product page

7.3 Cart page (before billing page)

7.4 Checkout page

8. Test Strategy

This task has a 24-hour deadline.

(In a real-world scenario, the project would likely use an agile approach with weekly iterations, where the requirements for each iteration are delivered to the team and tested at the end of the week.)

8.1 Test Types

The test strategy consists of a series of different tests- mainly Functional, UI, Error Handling (including edge cases) and Compatibility testing to ensure seamless user flow across the system.

8.1.1 Exploratory:

Series of tests that are done on the whole application without any test scripts and documentation.

8.1.2 Performance (lightweight) Testing:

Tests that simulate traffic to identify performance issues, such as ensuring reasonable page loading times.

8.1.3 Security:

Tests that will prevent unauthorized access (of users) and protect data.

8.1.4 Regression:

The purpose of these tests is to confirm that the system continues to behave as expected after changes or updates have been made, ensuring that these modifications have not introduced new bugs or broken previously working functionalities.

8.1.5 Accessibility:

The purpose of these tests is to ensures that the interface can be used by everyone, including those relying on screen readers or keyboard-only navigation. For this task, I will perform a manual accessibility check.

8.2 Risks

Potential risks include security weaknesses on the login page and during the checkout process if payment handling is unsecured. Another risk is the time constraint, which requires prioritizing the most essential functionalities first.

8.3 Observability/ Quality Metrics:

8.3.1 Percentage of pass/fail tests.

8.3.2 Root cause and log analysis of issues.

8.3.3 Number of open bugs out of the total bugs reported.

8.4 Data Strategy:

8.4.1 Creating users for testing, for example using postman. (In this exercise, new users cannot be created, so existing exercise users must be used).

8.4.2 Creating products for purchase (in this exercise, new products cannot be created or added, so the existing catalog must be used).

8.4.3 Different valid payment methods for the checkout process. (in this exercise, there is no option to enter payment information, so the data is whatever already exists in the system).

8.5 Release Gate:

8.5.1 Test documentation, clear test results, and if necessary, assessment of potential risks.
8.5.2 Approval from the product manager/ technical manager.
8.5.3 A rollback plan that is prepared and approved in case of a critical failure.
8.5.4 No critical failures.

8.6 Validation and bug Management

Bugs that are found during the testing will be categorized according to the bug-reporting tool and be prioritized accordingly.

8.7 Test tools:

	Tool	Comments
Bug Tracking	Jira	Report bugs
Agile	Jira	Design sprint milestones
Debugging	DevTools, Kibana, Postman	Analyzing root causes, API, logs