Test Plan Report: SauceDemo Login Automation

1. Objective

This test plan outlines the approach for automating the login functionality of the SauceDemo web application. The goal is to verify the behavior of the system under various user login scenarios using Selenium WebDriver, and to produce detailed reports and logs for test traceability.

2. Scope

In Scope:

- Automating login attempts for multiple user accounts using valid credentials.
- Validation of login success or failure based on URL and error messages.
- Screenshot capture for both successful and failed logins.
- Logging of test results and exceptions.
- Generation of Excel and HTML reports.

Out of Scope:

- Testing other functional areas like product listing, cart, checkout.
- Cross-browser and mobile testing.
- Performance or load testing.

3. Test Environment

OS: Windows 10/11

Browser: Google Chrome (latest version)

Programming Language: Python 3.12

Automation Tool: Selenium WebDriver

Libraries Used: pandas, openpyxl, logging

IDE: Visual Studio Code

4. Test Data

Test credentials are maintained in users.xlsx:

- standard_user
- locked_out_user

- problem_user
- performance_glitch_user
- error_user
- visual_user

All users share the password: secret_sauce.

5. Test Cases

TC01: Login with valid standard user -> Redirect to inventory page

TC02: Login with locked_out_user -> Error message shown, login blocked

TC03: Logout after successful login -> User returned to login page

TC04: Screenshot capture after each login attempt -> Screenshot saved

TC05: Log login result -> Logged with timestamp

TC06: Generate Excel report -> test_result.xlsx

TC07: Generate HTML report -> test_report.html

6. Execution Strategy

Read test data from users.xlsx. Open the SauceDemo login page in Chrome. Iterate through each username and attempt login. Validate success via current URL. Log result, take screenshot, and generate reports.

7. Exit Criteria

All test cases executed at least once. All login outcomes are logged and screenshots captured. Reports are generated without errors. No unhandled exceptions during execution.

8. Risks and Mitigation

- Site layout or element ID changes -> Use stable locators
- Incomplete or malformed Excel data -> Validate headers
- Browser-driver mismatch -> Match versions
- Delays in page load -> Use WebDriverWait

9. Deliverables

main.py

users.xlsx

test_result.xlsx

test_report.html

logs/test_log.txt

screenshots/

10. Sample Log Entry

```
2025-07-03 08:55:04 - INFO - Login successful for: standard_user
2025-07-03 08:55:10 - ERROR - Login failed for: locked_out_user - Reason: Login failed (locked out)
2025-07-03 08:55:13 - INFO - Login successful for: problem_user
```

11. Conclusion

This test plan ensures that all critical login scenarios are validated using automation. The reports and logs generated provide full traceability, and the framework can be extended to support additional test modules in the future.