



PROJECT NAME : PROJECT MANAGEMENT THROUGH AGENTIC AI (SE-04)

GROUP MEMBERS :

SYED MEERAN KAHN (22FA-049-SE)

WOOSQA SOHAIL (22FA-042-SE)

SIDQIAH JESSICA JOHN (22FA-030-SE)

SYEDA YUSRA ARSHAD (22FA-032-SE)

COURSE NAME : SOFTWARE QUALITY ENGINEERING COURSE

CODE : CSE-303

ASSIGNMENT 02: BLACK BOX TESTING USING SELENIUM COURSE

INSTRUCTOR : TALHA AHMED

1. Introduction

This report presents the results of black-box automated UI testing performed on the Agile Project Management Tool Frontend using Selenium WebDriver and pytest.

The primary objective of this testing activity was to verify application availability, page accessibility, navigation flow, and basic UI rendering without inspecting internal implementation details such as component logic, React state, or DOM-specific identifiers.

All tests were executed from an end-user perspective, ensuring compliance with black-box testing principles as required in Software Quality Engineering (SQE).

2. Test Environment

Component	Details
Programming Language	Python 3.13.3
Automation Tool	Selenium WebDriver
Test Framework	pytest 9.0.2
Reporting Tool	pytest-html 4.1.1
Browser	Google Chrome
Platform	Windows 11 (10.0.26200-SP0)
Execution Type	Black-Box UI Testing

3. Test Scope

The test suite focused on UI availability and navigation validation across key application routes.

3.1 Application Availability

- Application startup verification
- Base URL accessibility
- Browser-page rendering confirmation

3.2 Navigation & Routing

- Landing page access

- Dashboard page access
- Backlog page access
- Settings page access
- Story Review page access

Login and registration route reachability

3.3 UI Rendering Validation

- Presence of page body after load
- Page source availability
- Page title existence
- Page reload stability

3.4 Navigation Workflow

- Sequential navigation across multiple pages
- URL consistency during navigation
- Browser refresh behavior

4. Test Execution Summary

Metric	Count
Total Test Cases	16
Passed	16
Failed	0
Skipped	0
Errors	0
Execution Time	~6.5 seconds

All test cases executed successfully, indicating stable page availability and navigation behavior across the tested modules.

5. Test Case Results Analysis

5.1 Observed Results

- All core application routes loaded successfully
- No runtime UI crashes or blocking errors observed
- Navigation between pages remained consistent
- Pages rendered correctly after browser refresh
- No dependency on fragile UI selectors

5.2 Key Observations

- Tests intentionally avoided data-testid, XPath, or internal selectors
- Validation relied solely on URL reachability and visible page presence
- This approach increased test stability and reduced false failures

6. Test Code Structure Review

The automated test suite demonstrates the following strengths:

1. Proper use of explicit waits (WebDriverWait)
2. Clear modular separation of test files:
 - test_misc.py
 - test_dashboard.py
 - test_backlog.py
 - test_routes.py
 - test_navigation_smoke.py
 - test_page_titles.py
 - test_story_review.py
3. Strict black-box testing approach
4. Lightweight and maintainable test design
5. Reproducible execution with HTML reporting support

7. Risk & Impact Assessment

Area	Risk Level
Core Navigation	Low
Page Accessibility	Low
Routing & URL Handling	Low
UI Rendering Stability	Low

The absence of failures indicates low operational risk for the tested UI features under current conditions.

8. Recommendations

1. Extend tests to include authenticated workflows after login
2. Add negative routing tests (invalid URLs / 404 handling)
3. Capture screenshots on failure for diagnostic purposes
4. Introduce performance thresholds for page load times
5. Maintain black-box discipline to avoid brittle tests

9. Conclusion

The Selenium-based black-box UI testing of the Agile Project Management Tool Frontend was fully successful, achieving a 100% test pass rate across 16 automated test cases.

The results confirm that the application demonstrates strong UI availability, stable navigation, and reliable page rendering.

The testing approach aligns with SQE best practices and provides credible evidence of frontend stability without relying on internal implementation details.

10. Appendix – Test Execution Evidence

The automated test execution report generated using pytest-html is attached as formal evidence.

The report provides visual confirmation of:

- Total test cases executed
- Passed test results
- Execution environment details
- Individual test durations and outcomes

This report serves as official execution evidence for the black-box UI testing conducted on the Agile Project Management Tool.

Appendix A: Selenium Test Execution Report

Environment

Python	3.13.3
Platform	Windows-11-10.0.26200-SFP0
Packages	<ul style="list-style-type: none">• pytest-8.0.2• pluggy-1.6.0
Plugins	<ul style="list-style-type: none">• html-4.1.1• metadata-3.1.1

Summary

16 tests took 00:00:06

(Un)check the boxes to filter the results:

☐ 0 Failed ☒ 16 Passed ☐ 0 Skipped ☐ 0 Expected failures ☐ 0 Unexpected passes ☐ 0 Errors ☐ 0 History

Show all details Hide all details

Result	Test	Duration 	Links
Passed	test/testSmoke.py::test_app_is_running	110 ms	
Passed	test/testRoutes.py::test_routes_are_accessible[]	113 ms	
Passed	test/testRoutes.py::test_routes_are_accessible[dashboard]	116 ms	
Passed	test/testRoutes.py::test_routes_are_accessible[backlog]	117 ms	
Passed	test/testRoutes.py::test_routes_are_accessible[story-review]	117 ms	
Passed	test/testRoutes.py::test_routes_are_accessible[settings]	124 ms	
Passed	test/testMisc.py::test_dashboard_page_accessible	127 ms	
Passed	test/testRoutes.py::test_routes_are_accessible[login]	138 ms	
Passed	test/testMisc.py::test_settings_page_accessible	136 ms	
Passed	test/testMisc.py::test_landing_page_loads	140 ms	
Passed	test/testRoutes.py::test_routes_are_accessible[register]	156 ms	
Passed	test/testDashboard.py::test_dashboard_loads	316 ms	
Passed	test/testNavigationSmoke.py::test_user_can_navigate_multiple_pages	363 ms	
Passed	test/testPageFlow.py::test_pages_have_flow	475 ms	