NAVEEN AUTOMATION LABS
YouTube: Naveen AutomationLabs

Playwright Test Runner

VS

Cucumber + Playwright

Complete Feature Comparison Guide

Document Type:	Technical Comparison Guide
Created By:	Naveen Automation Labs
Target Audience:	QA Engineers, Test Automation Engineers
YouTube Channel:	Naveen AutomationLabs
Date:	December 2025
Version:	1.0

Executive Summary

This comprehensive guide compares two popular approaches for test automation with Playwright: using Playwright's native test runner (@playwright/test) versus integrating Playwright with Cucumber (@cucumber/cucumber) for Behavior-Driven Development (BDD).

Playwright Test Runner Strengths:	Cucumber + Playwright Strengths:
• 90% feature score (44/49 points)	• 59% feature score (29/49 points)
• Out-of-the-box features ready	• Gherkin/BDD syntax for collaboration
• Superior debugging tools (UI mode)	• Living documentation
• Automatic screenshots/videos/traces	• Business-readable test scenarios
• Built-in fixtures and test isolation	• Better for cross-functional teams
• Faster setup (5 minutes)	• Scenario Outline for data-driven tests

Conclusion: Choose Playwright Test Runner for technical teams seeking maximum features with minimal setup. Choose Cucumber + Playwright when BDD process and natural language test documentation are organizational requirements, understanding you'll need to manually implement many features that come out-of-the-box with Playwright Test.

Comprehensive Feature Comparison

1. Core Playwright Features

Feature	Playwright Test	Cucumber + PW	Winner
Test Runner	@playwright/test	@cucumber/cucumber	-
Config File	playwright.config.ts	cucumber.js	-
Test Syntax	TypeScript/JavaScript	Gherkin (Given/When/Then)	Depends
Auto-waiting	■ Yes	■ Yes	■ Equal
Smart Locators	■ Yes	■ Yes	■ Equal
Web-first Assertions	■ Yes	■ Yes	■ Equal
Cross-browser Support	■ Yes	■ Yes	■ Equal
Mobile Emulation	■ Yes	■ Yes	■ Equal
Network Interception	■ Yes	■ Yes	■ Equal

2. Test Runner Features

Feature	Playwright Test	Cucumber + PW	Winner
Built-in Fixtures	■ Auto-inject page/context	■■ Manual via World	■ PW
Test Isolation	■ Automatic	■■ Manual setup	■ PW
Multi-browser Config	■ Projects in one file	■■ Env vars/scripts	■ PW
Parallel Execution	■ Built-in workers	■ Built-in parallel	■ Equal
Test Sharding	■ Built-in	■■ Manual	■ PW

3. Retry & Error Handling

Feature	Playwright Test	Cucumber + PW	Winner
Test Retries	■ Smart per-test retries	■■ Scenario retries only	■ PW
Retry on Failure	■ Yes	■ Yes (with tag filter)	■ Equal

Timeout Config	■ Global + per-test	■■ Manual	■ PW

4. Debugging & Reporting

Feature	Playwright Test	Cucumber + PW	Winner
Screenshots on Fail	■ Automatic	■■ Manual After hook	■ PW
Video Recording	■ Auto (on-fail/always)	■■ Manual context config	■ PW
Trace Viewer	■ Built-in integration	■■ Manual	■ PW
HTML Reporter	■ Playwright Reporter	■■ Cucumber HTML/JSON	■ PW
Allure Reporter	■ Via plugin	■ Via plugin	■ Equal
JUnit/JSON Reports	■ Built-in	■ Built-in	■ Equal

5. Test Organization

Feature	Playwright Test	Cucumber + PW	Winner
Test Annotations	■ @skip, @slow, @fixme	■ Use Cucumber tags	■ PW
Test Grouping	■ describe() blocks	■■ Feature files	■ Equal
Tags/Labels	■ @tag in describe	■ @tag in Gherkin	■ Equal
Test Metadata	■ test.info()	■■ Manual attachment	■ PW

6. Test Execution

Feature	Playwright Test	Cucumber + PW	Winner
Run Specific Tests	■ --grep, file path	■ --tags, file path	■ Equal
Headed Mode	■ --headed	■■ Manual launch	■ PW
Debug Mode	■ --debug (Inspector)	■■ Manual debugging	■ PW
UI Mode	■ --ui (interactive)	■ Not available	■ PW
Watch Mode	■ --watch	■ Not available	■ PW

7. Code Reusability & BDD

Feature	Playwright Test	Cucumber + PW	Winner
Page Object Model	■ Easy to implement	■ Easy to implement	■ Equal
Custom Fixtures	■ Built-in system	■■ World + hooks	■ PW
Before/After Hooks	■ beforeEach, afterEach	■ Before, After	■ Equal
Shared State	■ Via fixtures	■ Via World	■ Equal
Gherkin Syntax	■ No	■ Yes	■ Cucumber
Living Documentation	■■ Via comments/docs	■ Feature files	■ Cucumber
Non-tech Readability	■ Code-based	■ Natural language	■ Cucumber
Business Analyst Friendly	■ No	■ Yes	■ Cucumber
Scenario Outline	■ Use loops	■ Built-in	■ Cucumber

8. Performance & Integrations

Feature	Playwright Test	Cucumber + PW	Winner
Setup Complexity	■ Simple (npx pw test)	■■ More config needed	■ PW
Test Execution Speed	■ Very fast	■■ Slight overhead	■ PW
Learning Curve	■■ Medium	■■ Medium	■ Equal
CI/CD Integration	■ Easy	■ Easy	■ Equal
VS Code Extension	■ Official	■ Cucumber ext	■ Equal
Test Generation	■ Codegen tool	■ Not available	■ PW
Package Dependencies	■ Minimal	■■ More packages	■ PW

Score Summary by Category

Category	Playwright Test	Cucumber + PW	Winner
Core Playwright Features	6/6 ■	6/6 ■	Equal
Test Runner Features	5/5 ■	2/5 ■■	Playwright
Retry & Error Handling	3/3 ■	1/3 ■■	Playwright
Debugging & Reporting	6/6 ■	2/6 ■■	Playwright
Test Organization	4/4 ■	2/4 ■■	Playwright
Test Execution	5/5 ■	2/5 ■■	Playwright
Code Reusability	4/4 ■	3/4 ■■	Mostly Equal
BDD & Collaboration	0/5 ■	5/5 ■	Cucumber
Performance & Speed	3/3 ■	1/3 ■■	Playwright
Integrations	3/3 ■	2/3 ■■	Mostly Equal
Package Management	2/2 ■	0/2 ■■	Playwright
TOTAL SCORE	**44/49 (90%)**	**29/49 (59%)**	**Playwright**

Decision Matrix

Choose Playwright Test Runner If:

■	You want fastest setup (5 minutes)
■	Team is technical (QA engineers, developers)
■	You need maximum out-of-the-box features
■	You want best debugging experience (UI mode, trace viewer)
■	Performance is critical
■	You prefer code-first approach
■	You want automatic screenshots/videos/traces
■	Test isolation and fixtures are important

Choose Cucumber + Playwright If:

■	Non-technical stakeholders need to read/write tests
■	BDD process is important to your organization
■	You need living documentation in natural language
■	Business analysts are involved in test creation
■	You're willing to implement features manually
■	Gherkin syntax is a requirement
■	You want Scenario Outline for data-driven tests
■	Cross-functional collaboration is priority

Effort Required to Match Playwright Test Features

The following table shows the approximate effort required to manually implement Playwright Test Runner features in a Cucumber + Playwright setup:

Feature	Lines of Code	Time Required	Complexity
Fixtures (World setup)	~50 lines	2-3 hours	■■ Medium
Screenshots on failure	~20 lines	30 minutes	■ Easy
Video recording	~15 lines	30 minutes	■ Easy
Multiple browser configs	~30 lines	1 hour	■■ Medium
Trace viewer integration	~40 lines	2-3 hours	■■■ Hard
Retries configuration	~10 lines	15 minutes	■ Easy
Custom reporters	~100 lines	4-5 hours	■■■ Hard
TOTAL EFFORT	**~265 lines**	**10-14 hours**	**Medium-Hard**

Final Verdict by Use Case

Aspect	Winner	Reason
Technical Teams	■ Playwright Test	Better DX, more features
Cross-functional Teams	■ Cucumber	Better collaboration
Speed of Setup	■ Playwright Test	5 min vs 2 hours
Maintainability	■ Playwright Test	Less boilerplate code
Documentation	■ Cucumber	Living Gherkin docs
Debugging Experience	■ Playwright Test	UI mode, trace viewer
Test Execution Speed	■ Playwright Test	Less overhead
Business Readability	■ Cucumber	Natural language
Learning Curve	■ Equal	Both require training
Long-term Maintenance	■ Playwright Test	Less custom code

Conclusion

The choice between Playwright Test Runner and Cucumber + Playwright is not about which is "better" in absolute terms, but rather which is more appropriate for your specific context.

Playwright Test Runner excels in technical environments where development speed, comprehensive debugging tools, and maximum out-of-the-box features are priorities. With a 90% feature score, it provides everything needed for effective test automation with minimal setup.

Cucumber + Playwright shines in organizations where collaboration between technical and non-technical team members is crucial, where living documentation is valued, and where the BDD process itself brings value beyond just test execution. While it requires more initial setup and ongoing maintenance, it serves a different purpose - facilitating communication and shared understanding.

For most technical QA teams, **Playwright Test Runner is the recommended starting point**. Teams can always introduce Cucumber later if organizational needs change. However, if BDD and Gherkin are organizational requirements from the start, be prepared to invest the additional effort to implement features that come automatically with Playwright Test.

Key Takeaway: Both are powerful approaches. Choose based on your team's composition, organizational culture, and project requirements - not based solely on technical capabilities.

NAVEEN AUTOMATION LABS
YouTube Channel: Naveen AutomationLabs
Website: www.naveenautomationlabs.com
Email: mail@naveenautomationlabs.com

This comprehensive comparison guide helps QA engineers and test automation professionals make informed decisions about test automation framework choices.

Document Version: 1.0 | Last Updated: December 2025 | © Naveen Automation Labs