Huge

laboratoriodetesting.com

Automation Testing Strategy

Building a Resilient E-commerce QA Foundation

07.10.25

Agenda.

Summary.

Goal: Build a Robust Automation Foundation
Context: Sprint 6 of 18 (1 QA, 3 Devs)

Goal.

Build a robust automation foundation

Context.

Sprint 6 of 18 (1 QA, 3 Devs)

Strategic Focus.

Revenue Protection: Checkout & Payment Reliability

User Acquisition: Registration & Login Dependability

Customer Experience: Essential E-commerce Journeys

Scalability: Future-proof, Modular Framework

Context.

With payment expansion right around the corner, and homepage redesign later, automation must be flexible and proactive, not reactive.

Current Phase: Sprint 6 (Project ~33% Complete)

Known Issues:
Client-reported Registration bugs
Errors during Checkout process

Upcoming Changes (High Impact):
Sprint 7–9: **New Payment Methods**
Sprint 15–18: **Homepage UX Redesign**

Tech Stack: API-driven Backend, Integrated Frontend

Testing Scope & Priorities.

Priorities are based on risk, business value, and upcoming changes. We focus first on the areas most likely to break and most visible to users.

High Priority. Must Cover ASAP

User Registration

User Authentication

E-commerce Core Flow: Product selection, cart operations, checkout

Medium Priority.

Order Management

Product Search/Filter

Profile Management

Basic Error Handling

Low Priority (Post-MVP)

Performance Testing

Extensive Cross-browser Compatibility

Mobile Responsiveness

Automation Approach.

Cypress was chosen for its speed and reliability. We pair it with a Page Object Model for maintainability, and fixtures to control test data across sprint phases.

Framework: Cypress *Why:* Real-time debugging, API support, auto-waiting, speed.

Pattern: Page Object Model (POM) *Benefits:* Maintainability, separation of concerns, scalability.

Critical Scenarios (Week 1 Priority):

- **Complete Purchase Journey:** *Captures baseline before Sprint 7 payment overhaul.*
- **User Registration:** *Directly addresses known client-reported issues.*
- **Authentication & Session Management:** *Critical for user flow integrity and access control.*

Implementation Strategy.

Week 1 Foundation & Baseline:

Cypress setup, repo scaffolding, env access.

Full automation of CURRENT checkout journey.

Reason: Payment logic changes next sprint; capture now.

Week 2 Stable Core Coverage:

Automate Registration flow (addresses known issues).

Automate Authentication & Session tests.

Low risk of breaking changes during S7–9 = stable test ROI.

Weeks 3–6 Adaptive Evolution:

Monitor payment features; update tests incrementally as new methods roll out.

Optimize framework for Sprint 15 Homepage redesign (resilient selectors, responsiveness).

The strategy matches the roadmap.
We start with stability, then evolve with the product changes.

Risk Mitigation & Technical Foundations.

This plan reduces risk by testing before change and coding stable selectors, modular functions, and communication with devs.

Sprint-Driven Risks & Mitigation

Payment Expansion: Week 1 baseline + flexible test modules.

Homepage Redesign: Resilient locators (data-testid, aria-labels) + POM isolation.

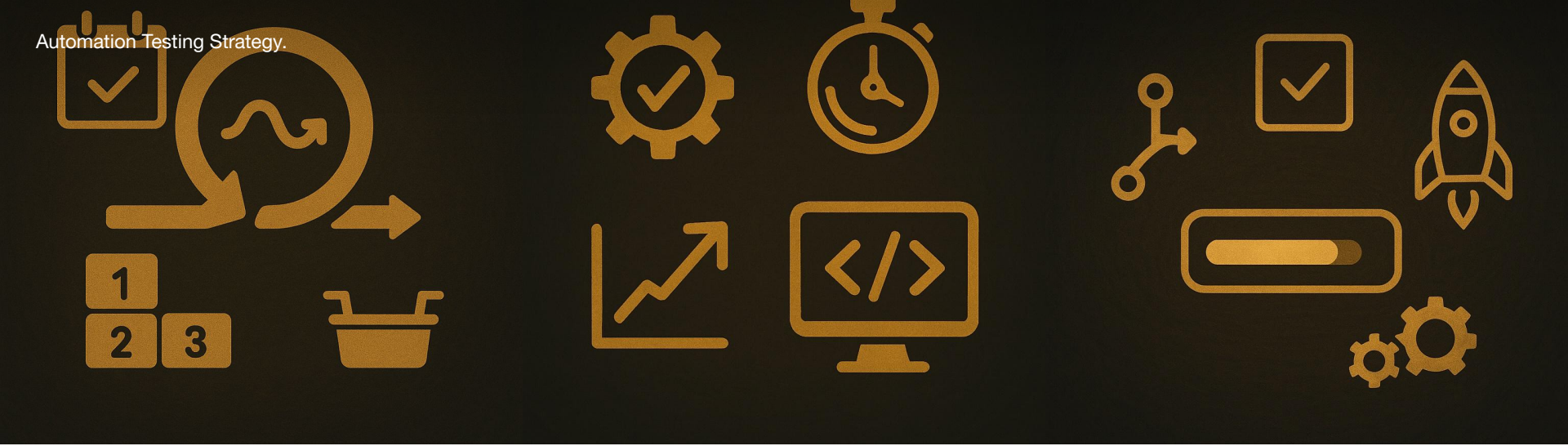
Key Technical Foundations

POM Modularity: Reusable components across flows.

API Testing: Validate frontend-backend interactions, failures, timeouts.

Sprint-Aware Test Data: Use fixtures to simulate different sprint states.

Reporting: Mochawesome Reports + CI/CD (Test Health Dashboard).



Sprint-Aligned Outcomes:

Week 1: Baseline checkout tests **completed.**

Week 2: Stable features **fully tested.**

Ongoing: Tests adapt **within 1 sprint** of feature changes.

Performance Metrics:

Test suite execution time: **<10 mins.**

Feedback loop: **<15 mins** post-commit.

Conclusion:

Change-resilient & **sprint-aware** strategy.
Focuses on **critical business flows early.**

Designed to **adapt quickly** to new features.
A **strategic asset**, not a liability.

This strategy is lean, and aligned with the sprint timeline. It's a forward-looking plan that safeguards both users and the roadmap.

Huge

laboratoriodetesting.com

Framework Execution: Results & Insights

Building a Resilient E-commerce QA Foundation

07.10.25

01 Test Results & Key Metrics
02 Analysis of Detected Bugs
03 Essential Configurations
04 CI/CD, Advanced Reporting & Limitations
05 Conclusions & Next Steps

Agenda.

Test Results & Key Metrics.

76%
Overall Success Rate (22/25 Tests)

7
Bugs Detected

3m29s
Total Execution Time

Results by Test Module:

Authentication Flow: 100% Success (7/7 tests)
User access working perfectly - STABLE MODULE.

E-Commerce Purchase Flow: 50% Success (4/8 tests)
Impact: CRITICAL - Multiple revenue-blocking bugs detected.

Registration Flow: 80% Success (8/10 tests)
Impact: HIGH - User acquisition mostly functional, UX issues identified.

Analysis of Detected Bugs.

The framework automatically identified 4 significant bugs, demonstrating its ability to find issues with high business impact.

CRITICAL.

SweetAlert Error: Blocks ALL purchases (0% conversion). *Location: Checkout popup.*

Missing CVV Validation: Payment failures (empty CVVs pass). *Location: Checkout CVV field.*

#aquatic Section Missing: Product category inaccessible. *Location: #aquatic section.*

HIGH.

Name Validation Flaw: Data integrity risk ("John123" in names). *Location: Name fields (checkout/reg).*

MEDIUM.

Email Validation UX: User confusion/abandonment (invalid emails allowed). *Location: Reg. email field.*

(Security) XSS Risk: Potential XSS via script in name fields. *Location: Name fields.*

LOW.

Form Button State: Minor UX confusion (premature button enable). *Location: Reg. form button.*

Essential Configurations.

cypress.config.js

Defines: Cypress's global behavior.

Key Settings: `baseUrl`, `viewport`, `defaultCommandTimeout`, `video`, `screenshotOnRunFailure`, `Mochawesome` integration.

Practical Application: Consistent environment, balanced timeouts, automatic evidence collection.

.gitignore

Defines: Excludes unnecessary/sensitive files from version control.

Excludes: `node_modules/`, `cypress/screenshots/`, `cypress/videos/`, `Mochawesome-results/`, `Mochawesome-report/`, `.env`, `cypress.env.json`.

Practical Application: Keeps repository clean, improves security, optimizes performance.

package.json

Defines: Commands for test execution & management.

Examples: `cy:open`, `cy:run:critical`, `test:all`, `Mochawesome:generate`, `Mochawesome:open`, `clean`.

Practical Application: Simplifies running tests, generating reports, and managing test artifacts.

Dynamic Test Data

Purpose: Generates unique data (e.g., emails) using timestamps & random strings.

Benefits: More robust and reliable tests, prevents collision, enables parallel testing, eliminates manual data cleanup.

CI/CD, Advanced Reporting & Limitations.

CI/CD Integration with GitHub Actions

Automation Trigger: Every Push / Pull Request to `main`, `develop`.

Workflow: Code Checkout -> Setup Node.js -> Run Cypress Tests (Chrome, Firefox) -> Upload Artifacts.

Benefits: Fast feedback, early regression detection, automatic cross-browser validation, evidence preservation.

Advanced Reporting with Mochawesome Reports

Features: Interactive HTML reports, test classification, automatic visual evidence (screenshots, videos), detailed steps.

Benefits: Clear stakeholder communication, easy failure analysis, living documentation.

LIMITATION: Mobile Testing Scope

Impact: Limited mobile coverage (desktop viewport only).

Reason: Mobile-specific UI/patterns not optimized; cross-platform tool integration challenges. (Browserstack)

Conclusions & Next Steps.

Framework Proven

Invaluable tool for **early detection of critical bugs** (revenue, UX).

Investment justified by **preventing costly production failures.**

Modular & Scalable architecture (POM, custom commands).

Fast, actionable feedback via CI/CD & Mochawesome.

Key Recommendations

IMMEDIATE FIX:
Critical Bug (Payment flow).

STRENGTHEN SECURITY:
Input Validations.

OPTIMIZE UX:
Investigate empty product sections.

EXPAND COVERAGE:
More scenarios & functionalities.

FUTURE:
Integrate Performance & Accessibility Tests.

Overall: Solid foundation for a continuous quality strategy, a strategic asset for the business.

Huge

Thank you.