Application: app.vwo.com – VWO A/B Testing & Conversion Optimization Platform
Prepared By: Sravanthi Veshala
Date: May 2025

---

1. Objective

To define a comprehensive, scalable, and risk-optimized test strategy that ensures the quality, performance, and reliability of app.vwo.com, supporting business-critical experimentation and conversion goals while enabling seamless integration and real-time data validation.

2. Scope

✅ In-Scope

- User Authentication (Login, MFA, SSO)

- Campaign Creation (A/B, Split URL, Multivariate)

- Visual Editor & Code Editor

- Audience Targeting & Segmentation

- Goal Setup, Tracking & Reporting

- Role-Based Access & Permissions

- Integrations (GA4, Mixpanel, HubSpot, etc.)

- API Interactions (Internal & External)

- Analytics & Experiment Result Data

❌ Out-of-Scope

- Marketing website (vwo.com)

- Backend Infrastructure (e.g., CDN, hosting) unless affecting app functionality

3. Quality Focus Areas

Area	Risk	Priority
Campaign Launch & Data Accuracy	High	Highest
Report Visualization & Filters	High	High
Editor Functionality (WYSIWYG + Code)	High	High
User Roles & Permissions	Medium	Medium
Third-party Integrations	Medium–High	High
Localization (if supported)	Low	Low

4. Testing Types & Approach

🔁 **Functional Testing**

- Modular test cases for campaign flow, goals, integrations

- Positive & negative scenario coverage

- CRUD operations (Campaigns, Goals, Segments)

🔌 **API Testing**

- Validate REST endpoints (Auth, Campaigns, Goals, Users)

- Contract testing with tools like **Pact**

- Schema validation and edge-case assertions

🎯 **UI Testing**

- Usability checks, form validations, modals & dynamic elements

- Responsive behavior on all supported viewports

🔁 **Regression Testing**

- Prioritized by usage frequency and bug history

- Automated for critical paths (CI-triggered)

- Manual regression for niche/low-change features

🌐 **Cross-Browser & Device Testing**

- Coverage: Chrome, Firefox, Safari, Edge (latest 2 versions)

- Devices: Windows, macOS, Android, iOS

- Use BrowserStack for automation grid

📊 **Performance Testing**

- Load and stress tests for:

  - Report generation

  - Editor rendering

  - Concurrent campaign publishing

- Tools: **k6, JMeter**

🔐 **Security Testing**

- OWASP Top 10 checklist (XSS, CSRF, IDOR, etc.)

- Role escalation checks

- Session timeout, token handling

♿ Accessibility Testing

- WCAG 2.1 AA compliance (color contrast, keyboard nav, ARIA roles)
- Tools: **axe-core**, **Pa11y**, **Lighthouse**

🎨 Visual Regression Testing

- Baseline snapshots of key pages (Editor, Campaign Setup)
- Tools: **Percy**, **Chromatic**

⚡ Resiliency & Chaos Testing

- Simulate downstream service failure (e.g., GA outage)
- Check retry logic, error handling, and campaign recovery

5. Environments

Environment	Purpose
Dev	Developer validation & unit testing
Staging	QA functional, API, and regression testing
Pre-Prod	UAT and performance testing
Production	Sanity testing & synthetic monitoring only

Test Data Management

- Automated test data factories
- Isolated data per suite/user
- Version-controlled mock datasets for regression

6. Automation Strategy

Area	Tool
UI	Playwright (fast, headless, cross-browser)
API	Postman + Newman, REST Assured (Java)
CI/CD	GitHub Actions with quality gates
Visual	Percy / Chromatic
Load	k6, JMeter

Automation Standards

- Page Object Model + TypeScript (for UI)
- API tests integrated in CI pipeline
- Flaky test auto-flagging with retry limits
- Daily/nightly test runs with reports

7. Entry/Exit Criteria

Entry

- Feature available in staging with acceptance criteria
- APIs available and documented
- Test data plan finalized

Exit

- 100% critical tests passed
- ≤1 major issue open with documented workaround
- Regression suite passes (automated + manual)
- Test summary published and signed off

8. Defect Management

Severity	SLA to Fix	Example
Critical	24 hrs	Data loss, login failure
Major	48 hrs	Broken campaign creation
Minor	Next Sprint	Cosmetic or UX inconsistency
Trivial	Backlog	Typos, minor spacing

Tracking via **JIRA**, linked to test cases (TestRail).

9. Reporting & Metrics

Weekly QA Dashboard

- Build Health: Pass/Fail rates
- Bug Trend by Module

- Flaky Test Detection
- Regression Stability Score
- MTTR (Mean Time to Resolve)

KPIs

- Test Case Effectiveness > 85%
- Automation Coverage (Critical Path) ≥ 85%
- Defect Leakage ≤ 2%
- Test Execution Time (Full Regression) < 2 hours
- Reopen Rate < 5%

10. Continuous Improvement

- Bi-weekly test case reviews with Product/Dev
- Monthly test suite refactoring
- Exploratory test sessions with cross-functional teams
- Post-release QA retrospectives
- Shadow real users periodically to refine test scenarios