Quathera

Test Runner Guide

How to run tests, monitor execution, and handle self-healing.

Version 1.0 | 2025

Table of Contents

1. Starting a Test Run

Prerequisites before running tests:

- Desktop Agent is running and shows 'Online'
- Form Discovery has been completed
- Form Mapping has been completed
- Test Scenarios have been generated

Steps to Start:

1. Go to 'Run Tests' in the sidebar
2. Select the project
3. Choose forms/scenarios to run (or select all)
4. Select target environment
5. Choose browser
6. Click 'Start Test Run'

What Happens Next:

The Desktop Agent receives the test run command and launches a browser. AI begins executing the test scenarios automatically.

2. Selecting What to Run

You can run tests at different levels of granularity:

Run All:

Execute all scenarios for all forms in the project. Best for full regression testing.

Run by Form:

Select specific forms to test. Useful when you've made changes to specific features.

Run by Scenario Type:

- CRUD tests only
- Negative tests only
- Workflow tests only

Run by Priority:

- High priority only (smoke tests)
- High + Medium priority
- All priorities

Run Failed Only:

Re-run only scenarios that failed in the previous run.

3. Environment Selection

Choose which environment to run tests against:

QA Environment:

- Default for most testing
- Safe to create/modify/delete test data
- Full test coverage recommended

Staging Environment:

- Pre-production validation
- Run before releases
- May need separate test data

Production Environment:

- Use with caution!
- Typically smoke tests only
- Read-only tests preferred
- Consider data cleanup requirements

Important: AI uses the Test Site credentials configured for the selected environment. Make sure they're valid and up-to-date.

4. Browser Options

Quathera supports multiple browsers:

Chrome:

- Default browser
- Best compatibility
- Fastest execution

Firefox:

- Good for cross-browser testing
- Some sites render differently

Edge:

- Windows enterprise environments
- Chromium-based, similar to Chrome

Electron:

- For Electron desktop applications
- Requires application path configuration

Multi-Browser Runs:

You can select multiple browsers to run the same tests across all of them sequentially.

5. Monitoring Test Execution

While tests are running, you can monitor progress:

On the Desktop Agent:

- Watch the browser executing steps
- See real-time log output
- View current scenario and step

On the Web App:

- Progress bar showing completion %
- Count of passed/failed/pending scenarios
- List of completed scenarios with status
- Real-time updates as tests complete

Status Indicators:

- ■ Green: Test passed
- ■ Red: Test failed
- ■ Yellow: Test passed with self-healing
- ■ Gray: Pending/not yet run

6. Self-Healing in Action

When UI changes break element locators, AI automatically heals them:

What Triggers Self-Healing:

- Element not found at expected location
- Element ID or class changed
- Element moved to different position
- Parent element structure changed

How AI Heals:

1. AI detects element not found
2. AI analyzes the page visually and structurally
3. AI finds the element using alternative locators
4. AI updates the locator for future runs
5. Test continues without interruption

Self-Healing Log:

All healing actions are logged with:

- Original locator that failed
- New locator that was found
- Confidence score of the match
- Screenshot showing the healed element

7. Handling Errors

When tests encounter errors, AI handles them intelligently:

Locator Changed (Self-Healed):

Test continues after healing. Marked as passed with warning.

Locator Not Found (Cannot Heal):

If AI cannot find the element, test fails with screenshot and details.

Validation Error:

Expected value doesn't match actual. Bug is created.

Network Error:

AI captures HTTP error details (4xx, 5xx responses) and continues or fails based on severity.

Page State Lost:

If session expires or navigation fails, AI attempts to recover by re-logging in. If recovery fails, test is marked as error.

Timeout:

If page or element takes too long, AI waits up to configured timeout, then fails with timeout error.

8. Stopping and Pausing Tests

You can control test execution:

Pause:

Temporarily stop execution. Tests can be resumed from where they paused.

- Click 'Pause' button in web app or agent
- Current scenario completes, then pauses
- Click 'Resume' to continue

Stop:

Completely stop the test run. Cannot be resumed.

- Click 'Stop' button
- Current scenario is terminated
- Results are saved for completed scenarios

Stop on Failure:

Configure tests to stop immediately when any scenario fails. Useful for debugging.

9. Viewing Test Logs

Logs are available in two places:

Desktop Agent Logs:

Real-time logs showing:

- Each step being executed
- Field values being entered
- Buttons being clicked
- Verification results
- Self-healing actions
- Errors and warnings

Web App Logs:

After test completes, detailed logs including:

- Full step-by-step execution log
- Screenshots at key points
- Screenshots on every failure
- Actual vs expected comparisons
- Timing information
- Self-healing details

Both views show the same information - use Agent logs for real-time monitoring, Web App logs for review and sharing.

© 2025 Quathera. All rights reserved.