Test Automation Strategy Template

A strategy template for defining automation goals, scope, tool choices, ownership, environments, data, reporting, and maintenance rules.

Purpose

Use this template before starting or resetting an automation program so the team aligns on value, risk, and sustainable execution.

1. Strategy Summary

- Business goals for automation are stated in measurable terms such as faster feedback, safer releases, or reduced manual repetition.
- Primary users of automation results are identified, including testers, developers, product owners, managers, and release leads.
- In-scope and out-of-scope applications, platforms, APIs, browsers, and workflows are documented.
- Automation success criteria are agreed before tool implementation begins.

2. Coverage Model

- Coverage is split across unit, integration, API, UI, visual, accessibility, performance, and smoke layers as appropriate.
- High-value regression paths are prioritized over large volumes of low-signal scripts.
- Manual, exploratory, and automated testing responsibilities are clearly separated.
- Critical journeys, high-risk integrations, and defect-prone areas receive explicit automation focus.

3. Tooling and Architecture

- Tool choices are justified by product architecture, team skill, CI/CD needs, reporting, and maintenance cost.
- Coding standards, naming conventions, folder structure, and reusable helpers are documented.
- Selector strategy, test data setup, cleanup, waits, retries, and environment configuration are standardized.
- The framework supports readable failures, useful logs, screenshots, traces, or other diagnostics.

4. Ownership and Workflow

- Teams know who writes, reviews, approves, fixes, and retires automated tests.
- Automation review is part of normal pull request or change-management workflow.
- Flaky tests are tracked, triaged, and fixed rather than ignored indefinitely.
- A change-impact process decides when automation must be updated for product changes.

5. Metrics and Maintenance

- Metrics include pass rate, failure reason, execution time, flaky rate, defect detection, and maintenance effort.
- Reports distinguish product failures, environment failures, data failures, and script failures.
- Old, duplicate, low-value, or unstable tests are retired through a regular audit process.
- The strategy is reviewed after major product, team, tool, or release-process changes.