Testing Policy

Table of Contents

Overview

- Backend (Rust): cargo-nextest was used for unit and integration tests. Nextest was chosen because it runs tests in parallel and is significantly faster than normal cargo test.
- End-to-End (System Tests): Playwright is used for E2E tests since it provides reliable cross-browser automation (Chromium, Firefox, WebKit), built-in parallelism, auto-waiting to reduce flakiness, and strong debugging capabilities.
- Performance Testing: K6 is used for load/performance testing. K6 was chosen because of its scripting flexibility, CI/CD integration, and modern developer workflow.
- CI/CD & Automation: GitHub Actions is used for automated pipelines. All tests run on every pull request. Builds fail if tests fail or coverage drops below the required threshold.

Justification:

- Cargo-nextest chosen for speed + reliability in Rust projects.
- Playwright was chosen for E2E because it runs across multiple browsers.
- K6 chosen for modern, scriptable load testing.
- GitHub Actions used as it is first-class with GitHub repos

Aims

The testing policy ensures the following:

- The backend API behaves as expected and handles edge cases.
- The frontend UI renders correctly and responds to user actions.
- Integration tests confirm backend + frontend communication.
- System/E2E tests validate full workflows in staging-like environments.
- Non-functional tests verify performance (K6), accessibility, and security.
- When failures occur, the system fails safely and recovers gracefully.

Implementation

- Test-Driven Development (TDD-ish): As features are built, tests are written alongside code. Making sure negative, edge and positive cases are covered for each test to ensure no unintended behaviour.
- CI/CD:
    - Every PR triggers GitHub Actions workflows.
    - Workflows run cargo nextest, rust linting and playwright
    - Merge blocked if tests fail
- Reviews: At least one reviewer must approve all PRs. For merges into main, two reviewers are required.
- Mocking/Isolation: External APIs are mocked when appropriate to ensure tests are deterministic.

Examples & References

- Workflow file: Workflow.yml
- GitHub Actions: Actions
- Example of Unit Marker
- Example of Integration Assignments
- Example of E2E Assignments
- Example PR