**ELECTA - Test Strategy & Plan (v1.2)**

**Document Version:** 1.2 **Date:** July 29, 2025 **Status:** Approved

**1.0 Introduction**

This document outlines the comprehensive Quality Assurance (QA) and testing strategy for the ELECTA platform. The plan covers all stages of the Software Development Life Cycle (SDLC), from initial planning through to post-launch monitoring, ensuring a high-quality, reliable, and secure application. This strategy is designed to be integrated directly into our development sprints.

**2.0 Planning Phase**

* **Goal:** To align our testing strategy with the project's core objectives and risks.
* **Activities for ELECTA:**
  + **Requirements Analysis:** The full suite of approved project documents (ELECTA\_01 through ELECTA\_14) will be used as the single source of truth for all functional and non-functional requirements.
  + **Test Strategy Definition:** Our strategy is risk-based, prioritizing the principles of Robust Security, High Performance, and Robustness & Reliability as defined in the Project Brief.
  + **Risk Assessment:** The primary risks identified for focused testing are the Manual NID Review Bottleneck, the security of the Admin Panel and Public Q&A systems, and the data integrity of the new Charter Compliance Module.

**3.0 Design Phase**

* **Goal:** To prepare detailed test cases and the necessary environment before development begins.
* **Activities for ELECTA:**
  + **Test Case Design:** Detailed test cases will be designed for all 16 approved User Stories. This includes positive ("happy path") scenarios and negative scenarios (e.g., uploading incorrect file types for NID, exceeding rate limits).
  + **Test Data Preparation:** A sanitized and anonymized dataset will be prepared, including mock Citizen profiles, Candidate data, Party information, and sample image files for NID verification testing. This will also include mock data for Charter compliance assessments.
  + **Test Environment Planning:** We will maintain three distinct environments: a local Development environment, a cloud-based QA/Staging environment for formal testing, and the final Production environment.

**4.0 Development Phase (Shift Left Testing)**

* **Goal:** To find and fix bugs as early as possible while code is being written.
* **Activities for ELECTA:**
  + **Unit Testing:** Developers are responsible for writing unit tests for all backend logic (using pytest) and Android components (using JUnit/Espresso).
  + **Static Code Analysis:** Our CI/CD pipeline will automatically run linting and static analysis on all code commits to maintain quality.
  + **Smoke Testing:** A basic, automated smoke test suite (TEST-1) will be built during Sprint 0 to verify that the application builds and core services are responsive after every deployment.

**5.0 Integration & System Testing Phase**

* **Goal:** To verify that the Android App and Backend API work together seamlessly and that the system as a whole meets requirements.
* **Activities for ELECTA:**
  + **Integration Testing:** We will conduct thorough API testing (using Postman/Newman) to validate the contracts between the Kotlin Android app and the FastAPI backend.
  + **System Testing:** We will test the complete, end-to-end user journeys defined in our user stories, such as a full citizen registration and a complete Q&A lifecycle.
  + **Regression Testing:** An automated regression suite will be built and expanded with each sprint. It will run automatically as part of our GitHub Actions CI/CD pipeline to ensure new features do not break existing functionality.

**6.0 User Acceptance Testing (UAT) Phase**

* **Goal:** To receive formal confirmation from stakeholders that the platform meets their needs before launch.
* **Activities for ELECTA:**
  + **Internal UAT:** A pilot group of internal stakeholders (including Data Editors) will be given access to the QA/Staging environment to test the Admin Panel workflows for user verification, content moderation, and Charter compliance assessment.
  + **External Beta Testing:** A small, trusted group of external users will be invited to a closed beta program to test the Citizen App and provide real-world feedback on usability and functionality.

**7.0 Deployment & Commercialization Phase**

* **Goal:** To ensure a stable launch and to monitor the platform's health in production.
* **Activities for ELECTA:**
  + **Performance & Load Testing:** Before launch, we will use a tool like JMeter or Locust to simulate the "Peak Submissions," "Burst Risk," and "Stress-Test" scenarios. This will include testing the performance of the Charter compliance public-facing APIs.
  + **Security Testing:** A third-party security firm will be engaged to conduct penetration testing against the live application, focusing on the vulnerabilities outlined for protection in our API Security Policy. This will include testing the evidence upload functionality and RBAC for the new module.
  + **Monitoring & Observability:** We will utilize the monitoring stack (Prometheus, Grafana, Sentry) to track application health, performance, and errors in real-time.

**8.0 Module-Specific Test Plans**

**8.1 Charter Compliance Module Test Plan**

* **Goal:** To ensure the Charter Compliance Module is accurate, secure, performant, and correctly reflects the specified business logic.
* **Unit Tests:**
  + The compliance badge calculation logic will be unit-tested with various mock data sets to ensure it correctly assigns Fully Compliant, Partially Compliant, and Non-Compliant badges based on the defined rules.
* **Integration Tests:**
  + Test the API contracts for all new CRUD endpoints under /api/v1/admin/charter/....
  + Validate that a PartyComplianceRecord cannot be created or updated without a valid foreign key to the ComplianceEvidence table.
  + Test the data-caching mechanism to ensure that updates to compliance records correctly invalidate the Redis cache.
* **System & End-to-End (E2E) Tests:**
  + **Core Workflow:** An automated E2E test will simulate a Data Editor logging in, creating a Charter Clause, uploading an evidence document, creating a full PartyComplianceRecord, and then, in a separate flow, verifying that the correct badge and data appear on the public-facing API for that party.
  + **Filtering:** Test the clause\_group filtering on the public API to ensure it returns the correct subset of data.
* **Security & Access Control Tests:**
  + Verify that API requests to create, update, or delete Charter compliance data are rejected if the user's JWT does not contain the DATA\_EDITOR or SUPER\_ADMIN role.
  + Verify that users with read-only roles (e.g., MODERATOR) cannot write data using these endpoints.
  + Test the evidence file upload mechanism for common vulnerabilities (e.g., path traversal, unrestricted file types).
* **Performance Tests:**
  + Execute a dedicated load test against the public /parties/{id}/compliance-summary endpoint to ensure it meets performance targets under heavy load, validating the effectiveness of the Redis caching strategy.