Technical Specification: Web Application Security & Quality Standards

Audience: Engineering, DevOps, QA, and Security Teams

---

1. Executive Summary

This document establishes the mandatory baseline requirements for all production-grade web applications deployed. The purpose of these standards is to minimize security risks (specifically OWASP Top 10 vulnerabilities), ensure code maintainability, and guarantee system reliability.

2. Terminology & Compliance

- **MUST / MANDATORY:** The item is an absolute requirement.
- **SHOULD / RECOMMENDED:** Valid reasons may exist to ignore this item, but the implications must be understood and documented.

---

3. Security Standards (SEC)

SEC-01: Identity & Access Management (IAM)

- **SEC-01.1:** Session tokens **MUST** utilize `HttpOnly`, `Secure`, and `SameSite=Strict` attributes to mitigate Cross-Site Scripting (XSS) and Cross-Site Request Forgery (CSRF).
- **SEC-01.2:** Passwords **MUST NOT** be stored in plain text. Use strong hashing algorithms (e.g., Argon2id or Bcrypt with a work factor > 10).

SEC-02: Data Protection

- **SEC-02.1:** All data in transit **MUST** be encrypted via TLS 1.3.
- **SEC-02.2:** Sensitive data at rest (PII, financial data) **MUST** be encrypted using AES-256 (or equivalent industry standard).
- **SEC-02.3:** Secrets (API keys, DB credentials) **MUST NOT** be committed to version control. They must be injected via a Secrets Manager (e.g., Vault, AWS Secrets Manager) at runtime.

SEC-03: Input & Output Handling

- **SEC-03.1:** All user input **MUST** be treated as untrusted. Input validation should occur on both the Client and Server sides.
- **SEC-03.2:** SQL queries **MUST** use Parameterized Queries (Prepared Statements) to prevent SQL Injection. String concatenation for queries is strictly prohibited.
- **SEC-03.3:** Content Security Policy (CSP) headers **SHOULD** be implemented to restrict resources the browser is allowed to load.

---

4. Code Quality Standards (QUAL)

QUAL-01: Maintainability & Style

- **QUAL-01.1:** All code **MUST** pass a static analysis check (Linting) before merging.
  - *Standard:* ESLint (JavaScript/TS), Pylint (Python), or equivalent.
- **QUAL-01.2:** Cyclomatic Complexity **SHOULD** be kept below 10 per function to ensure readability.
- **QUAL-01.3:** Components **MUST** adhere to the "Single Responsibility Principle."

QUAL-02: Testing & Coverage

- **QUAL-02.1:** The codebase **MUST** maintain a minimum Unit Test coverage of **80%**.
- **QUAL-02.2:** Critical business paths **MUST** be covered by Integration Tests.
- **QUAL-02.3:** A "Smoke Test" suite **MUST** run successfully against the production build artifact before deployment.

---

5. Operations & Reliability (OPS)

OPS-01: Monitoring & Logging

- **OPS-01.1:** Applications **MUST** implement centralized structure logging (e.g., JSON format).
- **OPS-01.2:** Logs **MUST NOT** contain PII, PCI data, or authentication tokens.
- **OPS-01.3:** Health check endpoints (`/healthz`, `/readyz`) **MUST** be exposed for load balancer and orchestrator monitoring.

OPS-02: Performance

- **OPS-02.1:** API endpoints **SHOULD** respond within 200ms (p95) for standard read operations.
- **OPS-02.2:** Rate Limiting **MUST** be implemented on public APIs to prevent Abuse/DoS attacks.

---

6. Review & Enforcement

Non-compliance with **MANDATORY** items requires a written exception approved by the CTO or Head of Security.

Role	Responsibility
Developer	Adhere to standards; run local linters/tests.
Tech Lead	Enforce standards during Code Review.
DevOps	Maintain CI/CD pipelines that automatically reject non-compliant builds.