TestVector - QA Signal Audit Sample Output

Disclaimer: This is a sample deliverable created to show the structure of a QA Signal Audit. It does not represent a real client system.

Severity	Meaning
High	Could allow release-critical failure through
Medium	Creates friction, false confidence, or CI noise
Low	Improvement opportunity

Product:
Fictional SaaS scheduling platform

Audit Scope:
- Signup/login flow
- Subscription checkout
- Admin user management
- Notification delivery
- CSV export
- CI regression suite

Key Findings:

1. High-risk workflow lacks backend verification
Severity: High
Area: Subscription checkout
Finding: The UI test confirms that the checkout success page appears, but no test verifies subscription state in the database or billing-service response.
Risk: A user may see a successful payment screen while backend entitlement creation fails.
Recommended fix: Add API-level verification for billing response and DB-level validation for subscription status.

2. Regression suite has duplicated UI setup
Severity: Medium
Area: Login/session setup
Finding: 42 tests repeat full login through the UI.
Risk: The suite spends unnecessary time testing the same login path and increases flakiness.
Recommended fix: Keep 2–3 direct login coverage tests, then use authenticated browser state for unrelated workflows.

3. CSV export is manually checked
Severity: High

Area: Reporting/export
Finding: CSV export is validated manually before release.
Risk: Column order, missing rows, formatting changes, and encoding issues may reach production.
Recommended fix: Add file-level automated validation for headers, required columns, row counts, and sample transformed values.

4. CI failures are not trusted
Severity: Medium
Area: CI signal
Finding: Intermittent failures are commonly rerun instead of investigated.
Risk: Real failures may be dismissed as flaky tests.
Recommended fix: Track flaky-test rate, quarantine unstable tests, and separate product failures from automation failures.

Recommended 30-Day Plan:

Week 1:
Map release-critical workflows and remove duplicated UI login setup.

Week 2:
Add API and database checks for checkout and account state.

Week 3:
Add CSV export validation and CI failure categorization.

Week 4:
Stabilize flaky tests and define release-gate smoke coverage.

Expected Outcome:
- Faster smoke suite
- Better backend coverage
- Cleaner CI signal
- Reduced manual export checking
- Higher confidence before release