

# Slice Web Application — Senior QA Test Plan

This version further elevates the plan with tighter scope, measurable gates, richer risk/coverage models, and CI/CD-operational detail.

## A. Executive Summary

Scope: auth & profile, search/menu/browse, cart & checkout, payments (card/Apple Pay/Google Pay/gift/coupon), order confirmation/status & notifications, store data & availability, promos/loyalty, responsive web, public APIs, RBAC (customer/merchant/support), 3rd-party integrations (payments/maps/SMS/email).

Out of scope: internal back-office, deprecated surfaces, heavy performance beyond baseline, unsupported 3P services in staging.

Environments & platforms: DEV→STAGE→UAT→PROD; browsers Chrome/Safari/Firefox/Edge latest±1; iOS Safari; Android Chrome; device widths 375/414/768/1024/1440.

Primary risks: payment gateway churn; price/customization rules; API contract drift; selector/visual fragility; timezone/store schedule logic; data consistency across services.

Mitigations: mock gateways + idempotency; decision tables + boundary values; OpenAPI contract enforcement; selector policy + visual diffs; timezone/DST matrix; synthetic deterministic data.

Decision ownership & gates: QA Lead runs gates; Go/No-Go with PO. Entry: AC ready, stable env, data seeded. Exit: no P0/P1, ≥95% must-run pass, payments validated on real & mock, a11y baseline, residual risk logged & accepted.

Runtime budgets: build smoke <10m; merge <30m; pre-release must-run <45m; nightly full <120m (parallel).

KPIs: escaped defects↓ QoQ, flaky rate <2%, regression hit-rate↑, CI time within budgets.

## B. AI-Powered Risk-Based Testing

Model:  $\text{Impact} \times \text{Likelihood} \times \text{Detectability}$  with signals from commit volume/area, usage funnels, and incident clusters. AI proposes clusters & missing tests; enforce assumptions + spec refs; spot-check 10–20%.

RiskID	Area	Evidence (change/usage/incidents)	Suggested Tests	Priority
R1	Payments	Recent gateway SDK upgrade; checkout is top-funnel	Contract + timeout/latency; retry/idempotency; wallet availability; negative auth; refund/void flows (mocked)	H
R2	Pricing & customization	Incidents on add-on limits & promos; multi-currency	Decision table (options×limits×promo); BVA on totals; currency/locale edges; rounding rules	H
R3	API contract drift	Frequent changes in ordering/stores services	OpenAPI schema validation; enums/required; pagination defaults/limits; error-shape checks	H
R4	Store schedule/availability	DST changes; weekend spikes; cache issues	Timezone matrix; closed→open transitions; cache invalidation checks; stale data detection	M
R5	UI fragility	Rapid UI iterations across devices	Selector policy (roles/testIDs); visual diffs; canary journeys; layout shift detection	M

## C. API Testing Strategy (OpenAPI/Swagger)

OpenAPI is source of truth; every test ties to operationId & ReqID. Include negative/boundary/locale payloads with ExpectedCode & Why. Non-functional: idempotency/retry, pagination, rate-limit, auth errors, correlation IDs, latency sampling.

### Contract/Negative/Boundary Matrix (excerpt)

OperationId	Method/Path	Test Type	Oracle (expected)	ReqID
createClaim	POST /claims	Contract	201 + ClaimResponse schema; DB row inserted	REQ-PAY-001
createClaim	POST /claims	Negative	400 with {code,message}; no state change	REQ-PAY-001
getClaim	GET /claims/{id}	Contract	200; schema valid; id matches	REQ-ORD-014
deleteClaim	DELETE /claims/{id}	Contract	200 + {deleted:true}; subsequent GET→404	REQ-OPS-021
listStores	GET /stores?city=	Boundary	200; pagination defaults respected; links/next cursor present	REQ-STR-003

### Payload Matrix (sample)

Payload	ExpectedCode	Why	ReqID
Valid baseline	201	All required present; ISO 8601 date; enum ok	REQ-PAY-001
currency='USDX'	400	Enum violation	REQ-PAY-001
amount=10000	201/400	Upper boundary by schema/business rule	REQ-PAY-001
date='2025-02-30'	400	Invalid date/format	REQ-PAY-001
missing memberId	422/400	Required field absent (per API)	REQ-PAY-001
locale=fr_FR with comma decimal	201/400	Locale formatting edge	REQ-PAY-001

## D. Mobile Testing Strategy

Device/OS matrix by market share: iOS Safari (latest,-1), Android Chrome (latest,-1). Test widths 375/414/768/1024/1440. Stability: semantic selectors, explicit waits on observable states (network idle/element stable), no arbitrary sleeps; visual diffs with masked dynamic areas.

### Exploratory Charters (5-8)

ID	Goal	Risks	Observables	DoD
M-01	Login + MFA happy path	Viewport/keyboard; OTP delays	Success banner; 200; session cookie	Pass on iOS+Android
M-02	Checkout with Apple/Google Pay	Wallet availability; redirects	Payment sheet opens; confirmation id; gateway log	Payment confirmed; no freeze
M-03	Promo entry	Keyboard overlay; paste; validation	Inline error/success; price recalculation	Accurate totals
M-04	Network degraded	Timeouts; retries; idempotency	Error toast; retry CTA; no double charge	One final outcome; audit entry
M-05	Store closed edge	Timezone/DST; cache	Closed state visible; ordering blocked; API says 'closed'	Consistent UI/API behavior
M-06	Accessibility quick pass	Focus order; labels	Roles/labels present; tabbable actions	Meets baseline WCAG cues

## E. AI Regression for LLM Features (if applicable)

Determinism: temperature≈0.2; fixed JSON/table formats; few-shot examples.

Sets: canonical (golden) prompts + adversarial (toxicity/PII/jailbreak);  
change-based selection across Component→API→UI/LLM.

Metrics: response validity rate, hallucination rate, jailbreak resistance, latency SLOs.

### LLM Cases (sample)

CaseID	Prompt Archetype	Expected (oracle)	Risk	ReqID
LLM-01	Canonical query	Relevant results only; no PII; ≤1.5s	Hallucination/latency	LLM-REQ-01
LLM-02	Adversarial PII ask	Refusal; safe message; audit log entry	PII exfiltration	LLM-REQ-02
LLM-03	Prompt injection	Ignore tool-breaking instruction; continue safely	Jailbreak	LLM-REQ-03

### Adversarial Expectations

Adversarial	Expected Block/De-escalation	Evidence Logged
Toxic language	Block/sanitize; rationale returned	Safety category + trace id
PII request	Refusal; policy reference	Audit event with user/session

## F. CI/CD Integration

Build: smoke (<10m) — API availability, homepage, login.

Merge: contract + component/API suites in parallel (<30m).

Pre-release: minimal must-run (Component→API→UI) with explicit exclusions (<45m).

Nightly: full regression + visual diffs; quarantine bucket separate.

Flaky control: signature clustering; quarantine; re-enable after ≥20 greens; trend flaky rate <2%.

Artifacts: one-page Release Decision Memo (scope/why, results, residual risk, Go/No-Go).

## G. Oracles & Data Policy

API oracles: status codes + JSON schema + headers; DB side-effects verified.  
Business oracles: totals/pricing invariant on invalid promo; audit events present.  
UI oracles: confirmation number equals API; a11y roles/labels; approved visual diffs.  
Data: synthetic seeds; masked exemplars; deterministic IDs; lifecycle (setup→use→cleanup); feature flags & env parity documented.

## H. Final Deliverables

Senior QA Test Plan (this doc, v2).  
Risk table (B).  
API matrix & payload matrix (C).  
Mobile device matrix & charters (D).  
LLM regression set (E).  
Release Decision Memo template (appendix).

## Appendix — Release Decision Memo (1-pager Template)

Section	Content
Scope & Why	What ran; why it protects value; mapping to changes/risks
Results	Pass/fail; flakes isolated; defects by severity; runtime vs budget
Residual Risk	What is not covered and why; mitigation/owner
Recommendation	Go/No-Go with rationale; next steps

## Changelog

Tightened scope and risks; added measurable gates & KPIs.  
Expanded API matrix and payload coverage (locales, missing required).  
Added accessibility quick pass, DST/timezone matrix, and richer mobile charters.  
Deepened LLM adversarial coverage and evidence requirements.  
Operationalized CI/CD with budgets and flaky-rate target.