Test Plan — LMS (Implement & Deploy Phase II) — Final

Document control

Version: 1.0

Date: 2025-08-28

Author: QA Team / Sok Kim Thanh

# A. Test Plan Identifier & Overview

**Test Plan ID: TP-LMS-2025-08-28-v1**

Project: LMS (Learning Management System) — Implement & Deploy Phase II

Scope summary: Comprehensive test plan for functional, integration, migration, deployment, and non-functional testing aligned with SDD, Implement\_And\_Deploy\_Phase\_II\_LMS.docx, and deployment checklist.

Objectives: verify feature correctness, deployment safety (migrations/backups/rollback), NFRs (performance, security, scalability), and observability (logging/metrics/tracing).

# B. Reference Documents

* SDD (Software Design Document) — architecture, NFRs, DDL, Outbox pattern, backup/DR requirements.
* Implement\_And\_Deploy\_Phase\_II\_LMS.docx — deployment runbook, env vars, snippets.
* Deployment\_Checklist\_LMS\_mini.docx — pre/post deploy checklist.
* TestPlan\_LMS\_mini.docx — original test cases (retained and extended).

# C. Test Items & Features to be Tested

**Test Items (modules & major components):**

* Authentication & Authorization (ASP.NET Identity, JWT, refresh tokens, RBAC, cookie vs bearer flows).
* Course Management, Module, Lesson, Enrollment flows.
* Quiz, QuizAttempt, scoring, timers, concurrency.
* FileAssets: upload (<=100MB), virus-scan, SHA256 hashing, signed URL retrieval.
* Outbox, Background Workers, Broker integration, DLQ handling.
* Database: migrations, seeders, backup & restore.
* Infrastructure: Blob Storage, SMTP, Message Broker, KeyVault/Managed Identity.
* Logging, Health Checks, Metrics (Prometheus), Tracing (OpenTelemetry/Jaeger).
* CI/CD pipeline: PR → build/test → staging deploy → migration-runner → smoke tests → prod deploy → post-deploy monitoring.

**Features NOT to be tested in this phase (unless requested):**

* Third-party integrations outside test scope (external LMS sync) unless test environment available.
* Long-term archival/analytics pipelines beyond acceptance criteria.

# D. Test Strategy & Approach

**Overview:**

Testing approach follows shift-left and ops-aware strategy covering unit, integration, API contract, E2E functional, migration/drill, smoke, performance, security (SAST/DAST), and chaos/resilience tests. Automation is prioritized for repeatability and CI integration.

**Levels of testing:**

1. Unit Testing: xUnit with Moq; target >= 80% coverage for domain services.
2. Integration Testing: dockerized SQL Server (or dedicated staging DB), testcontainers; verify EF Core mappings, filtered indexes, transactions.
3. API Contract Testing: contract tests (Postman/Newman or Pact) validating status codes, error envelopes, headers, and schema.
4. E2E / Functional Testing: Playwright (Blazor UI) + API-level E2E for main user flows.
5. Smoke Tests: automated scripts run post-deploy (pipeline) against staging and production (after manual approval).
6. Performance Testing: k6 scenarios executed in staging with production-like data.
7. Security Testing: SAST at PR, DAST on staging, and periodic pentest for auth & file upload.
8. Resilience/Chaos: simulate DB/broker outages, Outbox backlog scenarios, and verify recovery and alerts.

# E. Test Environment & Configuration

**Environments required:**

* Local Developer: unit & quick integration tests (in-memory or local DB).
* CI: containerized test runners for unit/integration suites.
* Staging: production-like infra (SQL Server/Azure SQL, Blob Storage, Broker, KeyVault, SMTP test relay, Prometheus/Grafana, Jaeger, ELK/Seq).
* Production: final smoke tests after manual approval.

**Mandatory environment configurations & environment variables:**

* ASPNETCORE\_ENVIRONMENT
* ConnectionStrings\_\_DefaultConnection
* BlobStorage\_\_ConnectionString
* SMTP\_\_Host, SMTP\_\_Port, SMTP\_\_User, SMTP\_\_Password
* Jwt\_\_Issuer, Jwt\_\_Key, Jwt\_\_ExpireMinutes
* DataProtection\_\_Keys (for multi-instance data protection)
* Storage\_\_SASKey (signed URL production key)
* Admin\_\_InitialEmail (for seeded admin)
* KeyVault\_\_Uri & ManagedIdentity settings (for production secrets)

Observability endpoints should be configured to send metrics/logs/traces to staging Prometheus/Grafana, ELK/Seq, and Jaeger. Health endpoints: /health (readiness/liveness) must be reachable.

# F. Test Deliverables

* This Test Plan document.
* Traceability Matrix (requirement → test case).
* Test cases and test scripts (Postman collections, Playwright suites, k6 scripts).
* Automated test run reports (unit, integration, E2E).
* Performance & load test reports and graphs.
* Security scan reports (SAST/DAST) and penetration test summary.
* Migration & backup drill reports.
* Deployment smoke-test results and sign-off checklist.

# G. Roles & Responsibilities

* QA Lead: owns test plan, coordinates execution, approves exit criteria.
* QA Engineers: author & automate test cases, execute suites, report defects.
* Developers: fix defects, provide migration scripts, assist with test infra.
* DevOps: provision infra, pipelines, backups, KeyVault, monitoring.
* Product Owner / Architect: accept sign-off and prioritize defects.

# H. Schedule & Milestones

**Sample timeline (adjust per project calendar):**

1. Week 0: Test planning & infra setup.
2. Week 1-2: Unit & integration test expansion; write API contract tests.
3. Week 3: E2E smoke automation & k6 baseline scripts.
4. Week 4: Run performance tests & security scans; execute migration & backup drills in staging.
5. Week 5: Regression, finalize test reports, pre-prod smoke and sign-off.

# I. Pass/Fail Criteria, Suspension & Resumption

**Pass Criteria (summary):**

* All Severity-1 (blocker) defects resolved or have approved mitigation before production deploy.
* All critical functional test cases pass (Enroll, Auth, Submit Quiz, File Upload).
* Performance NFRs met (P95 latencies within SLOs).
* Security scan shows no critical/high findings unresolved.
* Migration & backup drills successful and meet RPO/RTO.

Fail Criteria: any blocker defects, performance regressions beyond agreed thresholds, failed backup/restore drill, or critical security findings.

**Suspension & Resumption rules:**

- Suspend execution if environment is unstable (DB down, monitoring unavailable) or if a blocker defect prevents further meaningful testing.

- Resume after root cause fixed and environment validated; prior test artifacts to be re-run or re-validated as needed.

# J. Risk Management & Contingency Plans

**Key risks and mitigations:**

* Migration failure / data loss — Mitigation: mandatory full backup, transaction log backups, dry-run in staging, pre-approval, rollback scripts, and restore drill.
* Outbox backlog or broker outage — Mitigation: scaling workers, rate-limiting, alerting when backlog > 1000, DLQ handling, and operational playbook.
* Performance regressions — Mitigation: baseline tests, capacity planning, auto-scaling rules.
* Security vulnerabilities — Mitigation: SAST in PR, DAST on staging, emergency patching and incident response.
* Secrets exposure — Mitigation: use KeyVault/Managed Identity, rotate secrets every 90 days.

# K. Test Tasks & Activities

1. Prepare test environments and datasets (seed roles, admin user, sample courses).
2. Implement and run unit tests and measure coverage.
3. Implement integration tests for EF Core & repositories.
4. Develop Postman collections & Playwright suites for critical flows.
5. Implement k6 scripts for load scenarios and run in staging.
6. Run SAST/DAST scans and triage findings.
7. Execute migration-runner job in staging and run backup/restore drill.
8. Execute smoke-tests in CI pipeline post-deploy (staging & prod as appropriate).
9. Conduct resilience tests (DB failover, broker downtime) and verify alerts.
10. Prepare final test report and sign-off documentation.

# L. Test Cases — Functional (selected, with steps)

TC\_SMK\_001 — Smoke: Login (post-deploy)

1. Precondition: Service running; /health returns OK; test user exists.
2. Steps:
3. 1. POST /api/v1/auth/login with valid credentials.
4. 2. Verify response 200 and accessToken returned.
5. 3. Use token to call GET /api/v1/users/me and expect 200 + correct user data.
6. Expected: Login succeeds, token valid, user profile returned.

TC\_SMK\_002 — Smoke: Create Course

1. Precondition: Authenticated as Instructor.
2. Steps:
3. 1. POST /api/v1/courses with minimal valid payload.
4. 2. Verify response 201 and Location header.
5. 3. GET the course by ID and assert fields.
6. Expected: Course created and retrievable.

TC\_ENR\_001 — Enrollment positive

1. Precondition: student user exists and course is open.
2. Steps:
3. 1. POST /api/v1/enrollments with studentId & courseId and Idempotency-Key.
4. 2. Verify response 201 and Enrollment record in DB.
5. 3. Verify Outbox table has entry for enrollment event.
6. Expected: Enrollment persisted, Outbox entry present.

TC\_QUIZ\_001 — Start & Submit Quiz

1. Precondition: student enrolled and quiz exists.
2. Steps:
3. 1. POST /api/v1/quizzes/{quizId}/attempts to start.
4. 2. Verify attempt created and timer started.
5. 3. POST submission payload to /api/v1/quizzes/{quizId}/attempts/{attemptId}/submit.
6. 4. Verify grading, score persisted and result returned.
7. Expected: Grading correct and persisted.

TC\_FILE\_001 — File Upload boundary & validation

1. Precondition: authenticated user.
2. Steps:
3. 1. POST /api/v1/files with a valid file ≤100MB.
4. 2. Verify response 201 with signed URL and metadata contains SHA256.
5. 3. Upload a 101MB file and expect 413/ERR\_FILE\_TOO\_LARGE.
6. 4. Upload a file with invalid MIME; expect 400 and audit log entry.
7. Expected: file accepted within limits with hash and virus-scan; rejected otherwise.

# M. Test Cases — Migration, Backup & Rollback

1. TC\_MIG\_001 — Staging migration-runner (non-breaking)
2. Precondition: staging DB backed up.
3. Steps: 1) Run migration-runner job with new migration. 2) Verify migration applied (schema change present). 3) Run smoke tests to ensure functionality.
4. Expected: migration applies cleanly and services operate normally.
5. TC\_MIG\_002 — Failing migration & rollback/drill
6. Precondition: staging DB backed up and snapshot taken.
7. Steps: 1) Apply intentionally failing migration or simulate schema conflict. 2) If migration fails, execute rollback plan or restore from backup. 3) Time the restore and verify data consistency.
8. Expected: rollback or restore completes within RTO and data is consistent.
9. TC\_BCK\_001 — Backup & point-in-time restore drill
10. Precondition: database accumulating transactions.
11. Steps: 1) Trigger full backup and continuous transaction log backups. 2) Simulate data loss and perform point-in-time restore to desired timestamp. 3) Validate data integrity and app behavior.
12. Expected: Restore completes within RTO and data matches pre-loss state within RPO.

# N. Test Cases — Outbox & Background Workers

1. TC\_OUT\_001 — Outbox write & processing
2. Precondition: Outbox table empty.
3. Steps: 1) Perform operation that triggers domain event (e.g., enroll). 2) Verify Outbox entry in DB in same transaction. 3) Start worker and observe event published to broker.
4. Expected: Outbox entry created and published by worker.
5. TC\_OUT\_002 — Broker outage & DLQ behavior
6. Precondition: broker intentionally stopped.
7. Steps: 1) Trigger multiple events so Outbox backlog increases. 2) Verify worker retries with exponential backoff. 3) After RetryCount exhausted, verify messages moved to Dead-Letter Queue.
8. Expected: Messages retried and eventually moved to DLQ; alerts fired when backlog threshold exceeded.
9. TC\_OUT\_003 — Outbox backlog alerting
10. Steps: 1) Create synthetic backlog above configured threshold (e.g., >1000). 2) Verify alert in Grafana/alertmanager and paging behavior.
11. Expected: Alert triggered and notifications sent to on-call.

# O. Test Cases — Non-Functional (Performance, Security, Reliability)

**Performance & Load Tests:**

1. TC\_PERF\_001 — 1000 concurrent enrolls (k6)
2. Precondition: staging environment with scaled DB and workers.
3. Steps: 1) Run k6 script simulating 1000 concurrent enrolls. 2) Collect P95, error rates, DB CPU, and Outbox backlog.
4. Expected: P95 < 1s, error rate < 1%, Outbox backlog manageable.
5. TC\_PERF\_002 — 500 concurrent quiz submissions (k6)
6. Expected: acceptable latency and no data correctness issues.

**Security Tests:**

1. TC\_SEC\_001 — SAST at PR: ensure no critical/high issues.
2. TC\_SEC\_002 — DAST on staging: run OWASP ZAP scan against staging and verify no exploitable issues in auth/file upload paths.

**Reliability & DR:**

1. TC\_REL\_001 — DB failover drill (geo-replica)
2. TC\_REL\_002 — Full restore drill and validation

# P. Entry & Exit Criteria

**Entry Criteria:**

* Build passes unit tests and static analysis.
* Migrations authored and reviewed.
* Staging environment provisioned and seeded.
* Monitoring dashboards provisioned.

**Exit Criteria:**

* All critical test cases passed.
* No open Severity-1 defects.
* Performance & security acceptance criteria met or have mitigation plan documented.
* Smoke tests after production deploy are green and monitoring stable for agreed window.

# Q. Test Reporting & Metrics

* Daily test execution reports during active test phase.
* Consolidated test report with pass/fail, coverage, defect severity, and remediation status.
* Performance graphs (latency percentiles), resource utilization, and trend analysis.
* Security scan reports & remediation status.

# R. Approvals

**Sign-off required from:**

* QA Lead
* Engineering Lead / Architect
* DevOps Lead
* Product Owner

# S. Appendices & Automation Artifacts

Appendix A: Traceability Matrix (separate file).

Appendix B: Postman collections & Newman scripts for API contract tests.

Appendix C: Playwright suites for smoke & regression tests.

Appendix D: k6 scripts for load tests (1000 enrolls, 500 quiz submissions).

Appendix E: Sample migration-runner job template & pipeline snippets.

Appendix F: Contact & escalation list (QA, DevOps, On-call).