

Công Nghệ Phần Mềm Chuyên Sâu - SE214.Q11

Đề tài 5: Layered / Domain-Driven Design + Extreme  
Programming (XP) — Hệ thống  
Quản lý Học tập (LMS)

# Báo cáo Sprint 1

09.10.2025

23521224 Trương Hoàng Phúc

23521736 Bùi Văn Tùng

23520657 Vũ Quốc Huy

23520466 Tạ Hoàng Hiệp

23520682 Đỗ Đình Khang

23520448 Nguyễn Văn Hào

23520557 Dương Quốc Hùng

## Contents

<b>1. Kế hoạch thực hiện</b>	<b>1</b>
1.1. Sprint 1 – Project Setup & Role Initialization	1
1.2. Sprint 2 – Authentication & Course Browsing	1
1.3. Sprint 3 – Enrollment & Learning Flow	1
1.4. Sprint 4 – Instructor Tools & Course Management	1
1.5. Sprint 5 – Reports, Certificates & Monetization	2
<b>2. Product Backlog</b>	<b>2</b>
2.1. UC-01 – Login / Logout (Authentication)	2
2.2. UC-02 – Search Course	3
2.3. UC-03 – View Course Details	3
2.4. UC-04 – Enroll in Course	4
2.5. UC-05 – Learn Course / View Lesson	4
2.6. UC-06 – Submit Assignment	5
2.7. UC-07 – Grade Assignment	6
2.8. UC-08 – Apply / Approve Instructor	6
2.9. UC-09 – Notification (System Notifications)	7
2.10. UC-10 – Messaging / Q&A (Course Messages)	8
2.11. UC-11 – Create Course (Instructor Creates Course)	8
2.12. UC-12 – Modify Course (Edit Course)	9
2.13. UC-13 – View Reports / Analytics	9
2.14. UC-14 – Issue Certificate	10
2.15. UC-15 – Rate & Review Course	10
2.16. UC-16 – Payment / Checkout	11
2.17. UC-17 – Manage Users & Roles (Admin User Management)	12
2.18. UC-18 – Audit Log / System Logs	12
<b>3. Tổ chức nhóm</b>	<b>13</b>
<b>4. Công cụ</b>	<b>13</b>
<b>5. Hình ảnh</b>	<b>14</b>

## 1. Kế hoạch thực hiện

Mục này bao gồm kế hoạch thực hiện theo từng Sprint

### 1.1. Sprint 1 – Project Setup & Role Initialization

- **Mục tiêu:** Khởi tạo hệ thống, thiết lập môi trường, cơ sở dữ liệu, và chia vai trò.
- **Các công việc:**
  - Dụng kiến trúc dự án: Setup repo, CI/CD, môi trường backend/frontend
  - Tạo cơ sở dữ liệu ban đầu: Schema cho User, Role, Course
  - Thiết kế và seed dữ liệu Role: Admin, Instructor, Learner
  - Thiết lập mô hình xác thực cơ bản (chuẩn bị cho UC-01)
  - Phân quyền truy cập ban đầu: Cấu hình middleware, RBAC
  - Cấu hình hệ thống log ban đầu (chuẩn bị cho UC-18)
- **Kết quả:** Dự án có cấu trúc hoàn chỉnh, database hoạt động, có thể đăng nhập cơ bản bằng tài khoản mẫu.

### 1.2. Sprint 2 – Authentication & Course Browsing

**Mục tiêu:** Cho phép người dùng đăng nhập và tìm kiếm khóa học.

- **Use Cases:**
  - UC-01: Login / Logout
  - UC-02: Search Course
  - UC-03: View Course Details
  - UC-18: Audit Log / System Logs

**Kết quả:** Người dùng có thể đăng nhập và xem khóa học; hệ thống ghi log hành động.

### 1.3. Sprint 3 – Enrollment & Learning Flow

- **Mục tiêu:** Hoàn thiện luồng học của học viên.
- **Use Cases:**
  - UC-04: Enroll in Course
  - UC-05: Learn Course / View Lesson
  - UC-06: Submit Assignment
  - UC-09: Notification
- **Kết quả:** Học viên có thể học, nộp bài, và nhận thông báo tự động.

### 1.4. Sprint 4 – Instructor Tools & Course Management

- **Mục tiêu:** Giảng viên có thể tạo, chỉnh sửa, và quản lý khóa học.
- **Use Cases:**

- UC-07: Grade Assignment
- UC-08: Apply / Approve Instructor
- UC-11: Create Course
- UC-12: Modify Course
- UC-17: Manage Users & Roles
- **Kết quả:** Hệ thống hỗ trợ quy trình tạo và quản lý khóa học cho giảng viên, admin.

### 1.5. Sprint 5 – Reports, Certificates & Monetization

- **Mục tiêu:** Bổ sung chức năng nâng cao, thương mại hóa và phản hồi người dùng.
- **Use Cases:**
  - UC-10: Messaging / Q&A
  - UC-13: View Reports / Analytics
  - UC-14: Issue Certificate
  - UC-15: Rate & Review Course
  - UC-16: Payment / Checkout
- **Kết quả:** Hệ thống hoàn thiện với chứng chỉ, đánh giá, thanh toán và báo cáo phân tích.

## 2. Product Backlog

Mục này gồm các use case của hệ thống.

### 2.1. UC-01 – Login / Logout (Authentication)

- **Actor:** Learner / Instructor / Admin (Owner / Staff / Customer)
- **Description:** Log in and log out of the system to access features based on user roles.
- **Preconditions:** The user already has a registered and verified account in the system.
- **Postconditions:** Successful login → session/token created; Logout → session terminated; Invalid credentials → error message displayed.
- **Priority:** High
- **Frequency of Use:** High
- **Normal Course of Events:**
  1. User enters credentials (email/password)
  2. System validates credentials
  3. On success → create session/token and redirect to dashboard
  4. User logs out → system terminates session
- **Alternative Courses:**

1. User forgets password → selects “Forgot password” → system sends reset email
  2. Login via OAuth (Google/Facebook) if enabled
- **Exceptions:** Account locked, email not verified, too many failed attempts → show corresponding error
  - **Includes:** Password reset, optional 2FA verification
  - **Extends:** N/A
  - **Special Requirements:** Password must not be displayed on screen; HTTPS encryption; login attempts rate-limited
  - **Assumptions:** Email service is available for verification and password reset
  - **Notes and Issues:** All login/logout actions are logged for auditing

## 2.2. UC-02 — Search Course

- **Actor:** Learner / Guest
- **Description:** Search and filter courses by keyword, topic, instructor, level, or price (free/paid).
- **Preconditions:** Published courses exist in the system.
- **Postconditions:** Displays a list of matching results; user can open course detail page.
- **Priority:** High
- **Frequency of Use:** High
- **Normal Course of Events:**
  1. User opens the search page
  2. Enters keywords or selects filters
  3. The system displays paginated and sorted results
  4. User clicks a result to view course details
- **Alternative Courses:**
  1. No results found → show suggestions or related courses
- **Exceptions:** Server/database error → display error message
- **Includes:** View Course Details (UC-03)
- **Extends:** N/A
- **Special Requirements:** Unicode search supported; response time < 1s for normal page size
- **Assumptions:** Courses are properly indexed for fast searching
- **Notes and Issues:** Consider caching for common queries

## 2.3. UC-03 — View Course Details

- **Actor:** Learner / Guest / Instructor
- **Description:** Display detailed information about a course: description, syllabus, instructor, reviews, and assignments.
- **Preconditions:** The course is published (or in draft if viewed by the instructor).

- **Postconditions:** User can choose to enroll or bookmark the course.
- **Priority:** High
- **Frequency of Use:** High
- **Normal Course of Events:**
  1. User opens the course detail page
  2. System displays title, description, syllabus, preview video, rating, price, and “Enroll” button
- **Alternative Courses:**
  1. If course is unpublished and viewer isn’t the instructor → show 404 or forbidden message
- **Exceptions:** Media fails to load → show placeholder
- **Includes:** Search Course (UC-02), Enroll Course (UC-04)
- **Extends:** N/A
- **Special Requirements:** Supports preview videos; responsive design; lazy-load for reviews
- **Assumptions:** Course media and reviews are available via CDN
- **Notes and Issues:** Display enrollment limits if applicable

## 2.4. UC-04 — Enroll in Course

- **Actor:** Learner
- **Description:** Enroll in a course to gain access to learning materials.
- **Preconditions:** Learner is logged in; course is open for enrollment.
- **Postconditions:** Enrollment record created; learner gains access; if paid, payment transaction is created.
- **Priority:** High
- **Frequency of Use:** High
- **Normal Course of Events:**
  1. Learner clicks “Enroll”
  2. If course is free → enrollment created immediately
  3. If paid → redirect to checkout/payment
  4. On payment success → create enrollment and send confirmation email
- **Alternative Courses:**
  1. Payment fails → display error and remain on checkout page
- **Exceptions:** Course full or closed → show “Enrollment unavailable”
- **Includes:** Payment workflow
- **Extends:** N/A
- **Special Requirements:** Store timestamp; prevent duplicate enrollment
- **Assumptions:** Payment gateway active; enrollment limits handled
- **Notes and Issues:** Should support coupons/discounts

## 2.5. UC-05 — Learn Course / View Lesson

- **Actor:** Learner

- **Description:** Access and view lessons (videos, slides, files), mark them as completed.
- **Preconditions:** Learner is enrolled in the course; lesson is published.
- **Postconditions:** Learner's progress is updated and saved.
- **Priority:** High
- **Frequency of Use:** High
- **Normal Course of Events:**
  1. Learner opens a module/lesson
  2. System displays the content (video, text, file)
  3. Learner completes the lesson → marks as "Complete"
  4. System updates progress %
- **Alternative Courses:**
  1. Lesson includes assignment → redirects to Submit Assignment (UC-06)
- **Exceptions:** Media not available → show error with support option
- **Includes:** Progress tracking, material download
- **Extends:** N/A
- **Special Requirements:** Auto-resume video position; offline access if allowed
- **Assumptions:** Stable streaming and storage
- **Notes and Issues:** Log viewing time for engagement analytics

## 2.6. UC-06 – Submit Assignment

- **Actor:** Learner
- **Description:** Submit assignment files or text for instructor evaluation.
- **Preconditions:** Learner is enrolled and the assignment is active.
- **Postconditions:** Submission record created (status = Submitted); Instructor receives notification.
- **Priority:** High
- **Frequency of Use:** Medium
- **Normal Course of Events:**
  1. Learner opens assignment page
  2. Uploads file or enters text → clicks Submit
  3. System validates type and size → saves file, creates submission record, timestamped
  4. Notification sent to instructor
- **Alternative Courses:**
  1. Submission after deadline → accepted but marked Late
  2. Resubmission allowed before deadline
- **Exceptions:** Invalid file type or size → error message; storage failure → rollback submission
- **Includes:** File upload service, Notification (UC-09)
- **Extends:** N/A

- **Special Requirements:** File size limit (e.g., 50MB), allowed formats (PDF/DOCX/ZIP), optional virus scan
- **Assumptions:** File storage service (e.g., S3/CDN) available
- **Notes and Issues:** Keep submission versions for audit; allow resubmission if permitted

## 2.7. UC-07 – Grade Assignment

- **Actor:** Instructor
- **Description:** Evaluate submitted assignments and assign grades with feedback.
- **Preconditions:** At least one submitted assignment exists; instructor has permission for the course.
- **Postconditions:** Submission updated with grade and feedback; learner notified; grade affects progress.
- **Priority:** High
- **Frequency of Use:** Medium
- **Normal Course of Events:**
  1. Instructor views submission list
  2. Opens one submission → reviews file → enters score and feedback → saves
  3. System updates status to “Graded” and notifies learner
- **Alternative Courses:**
  1. Instructor requests resubmission → status “Needs Resubmit”
- **Exceptions:** File corrupted or missing → request new submission
- **Includes:** Notification (UC-09), Audit log (UC-18)
- **Extends:** N/A
- **Special Requirements:** Track who graded and when; store grade change history
- **Assumptions:** Instructor has access to submission files
- **Notes and Issues:** Support bulk grading or CSV import if needed

## 2.8. UC-08 – Apply / Approve Instructor

- **Actor:** Learner (Applicant) / Admin
- **Description:** A learner applies to become an instructor; admin reviews and approves or rejects the application.
- **Preconditions:** Applicant is logged in and has a complete profile; Admin has permission to approve.
- **Postconditions:** Approved → role updated to Instructor; email notification sent; Rejected → applicant notified with reason.
- **Priority:** Medium
- **Frequency of Use:** Low to Medium
- **Normal Course of Events:**



1. Applicant opens the “Apply for Instructor” form
  2. Fills in information, uploads portfolio, and submits
  3. System sets status = Pending and notifies Admin
  4. Admin reviews and either Approves or Rejects
  5. System updates status and sends email notification
- **Alternative Courses:**
    1. Admin requests more info → status “Needs More Info”
  - **Exceptions:** Applicant doesn’t meet requirements → rejected automatically
  - **Includes:** Notification, Audit log (UC-18)
  - **Extends:** N/A
  - **Special Requirements:** Store portfolio; form validation; email templates for notifications
  - **Assumptions:** Manual review process by admin; SLA for approval
  - **Notes and Issues:** Consider automating approval if criteria become standardized

## 2.9. UC-09 — Notification (System Notifications)

- **Actor:** System (background), Learner, Instructor, Admin
- **Description:** Send in-app and/or email notifications for events such as enrollment, grading, new messages, certificate issuance.
- **Preconditions:** User has an account; user notification preferences (opt-in/opt-out) are set (default ON).
- **Postconditions:** Notification record created and stored; user receives notification in UI and/or by email according to settings.
- **Priority:** Medium
- **Frequency of Use:** High
- **Normal Course of Events:**
  1. An event occurs (e.g., a submission is graded)
  2. The system creates a notification record
  3. The notification appears in the notification center; an email is sent if configured
- **Alternative Courses:**
  1. If user disabled email notifications → only show in-app
- **Exceptions:** Email service outage → retry and log failure
- **Includes:** Audit log (UC-18)
- **Extends:** N/A
- **Special Requirements:** Email templates, rate-limiting, localization
- **Assumptions:** SMTP/notification services are available and operational
- **Notes and Issues:** Support batching for multiple notifications to the same user

## 2.10. UC-10 — Messaging / Q&A (Course Messages)

- **Actor:** Learner, Instructor
- **Description:** Exchange direct messages (1:1) or threaded Q&A within the course context (questions and replies).
- **Preconditions:** Sender is enrolled in the course or is the course instructor.
- **Postconditions:** Messages persisted; thread updated; recipients receive notifications.
- **Priority:** Low → Medium
- **Frequency of Use:** Medium
- **Normal Course of Events:**
  1. User opens or creates a thread
  2. Types a message and sends it
  3. Server saves the message and notifies the recipient(s)
- **Alternative Courses:**
  1. Attach a file to the message (subject to allowed types and size)
- **Exceptions:** Message too long or invalid format → reject with error
- **Includes:** Notification (UC-09)
- **Extends:** N/A
- **Special Requirements:** Input sanitization (prevent XSS), pagination, message search
- **Assumptions:** Real-time or near-real-time delivery (via WebSocket/long-polling or polling) is implemented as needed
- **Notes and Issues:** Can be extended to forum-style public Q&A

## 2.11. UC-11 — Create Course (Instructor Creates Course)

- **Actor:** Instructor
- **Description:** Create a new course with title, description, modules, lessons, assignments, pricing, and media.
- **Preconditions:** User has the Instructor role; profile verification if required by policy.
- **Postconditions:** Course draft saved; instructor may publish the course to allow enrollments.
- **Priority:** High
- **Frequency of Use:** Medium
- **Normal Course of Events:**
  1. Instructor opens “Create Course”
  2. Enters metadata, builds modules & lessons, uploads media, defines assignments/pricing
  3. Saves as Draft or Publishes the course
- **Alternative Courses:**
  1. Save draft to complete later
- **Exceptions:** Media upload failure → display error and retry

- **Includes:** Media upload service; Course Preview (UC-03)
- **Extends:** N/A
- **Special Requirements:** WYSIWYG editor for descriptions; media validation; autosave drafts
- **Assumptions:** Storage/CDN service is available
- **Notes and Issues:** Consider support for versioning and course cloning

## 2.12. UC-12 — Modify Course (Edit Course)

- **Actor:** Instructor
- **Description:** Edit an existing course's content: syllabus, lessons, price, and publish status.
- **Preconditions:** Instructor owns the course or has edit permission.
- **Postconditions:** Course updated; if published, changes are reflected live; change history recorded.
- **Priority:** Medium
- **Frequency of Use:** Medium
- **Normal Course of Events:**
  1. Instructor opens the course and clicks Edit
  2. Makes changes and clicks Save/Publish
  3. System updates the course and records an audit trail
- **Alternative Courses:**
  1. Rollback to a previous version if version history exists
- **Exceptions:** Concurrent edit conflict → notify user and offer merge or overwrite options
- **Includes:** Audit log (UC-18)
- **Extends:** N/A
- **Special Requirements:** Draft vs. Published workflow; preview before publish
- **Assumptions:** Locking/versioning mechanisms exist
- **Notes and Issues:** Consider granular permissions for co-instructors

## 2.13. UC-13 — View Reports / Analytics

- **Actor:** Instructor, Admin
- **Description:** View reports and analytics: enrollments, completion rates, revenue, engagement, top lessons.
- **Preconditions:** Activity data exists; user has permission (instructors limited to their courses).
- **Postconditions:** Dashboard displayed; data may be exported as CSV/PDF for decision making.
- **Priority:** Medium
- **Frequency of Use:** Medium
- **Normal Course of Events:**

1. User opens Reports
  2. Selects filters (date range, course)
  3. System displays charts, tables, and KPIs
- **Alternative Courses:**
    1. Export report data to CSV/PDF
  - **Exceptions:** Data unavailable or incomplete → show notice
  - **Includes:** Audit log (UC-18) for sensitive data queries
  - **Extends:** N/A
  - **Special Requirements:** Performance for large datasets; role-based access control for sensitive metrics
  - **Assumptions:** ETL/analytics pipelines provide aggregated data
  - **Notes and Issues:** Optionally support scheduled reports/email digests

#### 2.14. UC-14 — Issue Certificate

- **Actor:** System (automated), Instructor, Admin
- **Description:** Automatically or manually issue certificates when a learner meets course completion/pass conditions.
- **Preconditions:** Issuance rules defined (completion threshold, passing grade, or manual approval).
- **Postconditions:** Certificate PDF or URL generated with a unique ID; learner can download and verify it.
- **Priority:** Medium
- **Frequency of Use:** Low → Medium
- **Normal Course of Events:**
  1. Learner meets issuance conditions
  2. System generates a certificate from a template with prefilled data
  3. Certificate is stored and made available for download; notification sent
- **Alternative Courses:**
  1. Manual issuance by instructor/admin
- **Exceptions:** Missing template data → flag for manual review
- **Includes:** Notification (UC-09)
- **Extends:** N/A
- **Special Requirements:** Unique verification URL, tamper-evident signature, PDF generation
- **Assumptions:** Template and learner data (name, date) are accurate
- **Notes and Issues:** Support sharing (e.g., LinkedIn) and a verification API

#### 2.15. UC-15 — Rate & Review Course

- **Actor:** Learner
- **Description:** After enrolling or completing a course, learners can rate (stars) and write reviews for the course.

- **Preconditions:** Learner has enrolled; optionally only allowed after completion.
- **Postconditions:** Review saved; the course's average rating is updated and displayed.
- **Priority:** Medium
- **Frequency of Use:** Medium
- **Normal Course of Events:**
  1. Learner selects to rate the course, enters stars and text, and submits
  2. System validates (e.g., one review per enrollment) and saves the review, updating aggregate rating
- **Alternative Courses:**
  1. Learner edits or deletes a review within the allowed policy window
- **Exceptions:** Review content violates policy → flag for moderation
- **Includes:** Optional notification to the instructor
- **Extends:** N/A
- **Special Requirements:** Moderation queue, spam/fraud detection
- **Assumptions:** Users act in good faith and review policy exists
- **Notes and Issues:** Consider incentives for leaving reviews

## 2.16. UC-16 – Payment / Checkout

- **Actor:** Learner, Payment Gateway, Admin
- **Description:** Process payments when enrolling in paid courses: cart/checkout, payment processing, invoice generation.
- **Preconditions:** Course is marked as paid; learner has billing information.
- **Postconditions:** On payment success → enrollment and invoice created; on failure → transaction rolled back.
- **Priority:** High (for paid-course platforms)
- **Frequency of Use:** Low → Medium
- **Normal Course of Events:**
  1. Learner proceeds to checkout
  2. Selects payment method and submits
  3. System calls the payment gateway; on success, creates enrollment, invoice, and sends notifications
- **Alternative Courses:**
  1. Apply coupon/discount
  2. Use wallet/balance or stored payment token
- **Exceptions:** Payment declined → display reason and offer retry options
- **Includes:** Notification (UC-09), Audit log (UC-18)
- **Extends:** N/A
- **Special Requirements:** PCI-DSS compliance, idempotency for payment requests, secure handling of payment tokens
- **Assumptions:** Payment provider supports required currencies and methods

- **Notes and Issues:** Implement webhooks to handle asynchronous payment updates reliably

### 2.17. UC-17 — Manage Users & Roles (Admin User Management)

- **Actor:** Admin
- **Description:** Admin creates/edits/deletes users, assigns roles (Instructor, Admin), and suspends accounts.
- **Preconditions:** Admin is authenticated and has sufficient privileges.
- **Postconditions:** User records updated; changes logged for audit.
- **Priority:** High
- **Frequency of Use:** Low
- **Normal Course of Events:**
  1. Admin searches for a user and edits roles/status, then saves
  2. System applies changes and records the action in the audit log
- **Alternative Courses:**
  1. Bulk operations via CSV import/export
- **Exceptions:** Attempt to change super-admin role → blocked
- **Includes:** Audit log (UC-18), Notification if role changed
- **Extends:** N/A
- **Special Requirements:** RBAC enforcement, strong validation, secure admin UI
- **Assumptions:** Admin UI and role model are implemented
- **Notes and Issues:** Support impersonation for support purposes (with strict auditing)

### 2.18. UC-18 — Audit Log / System Logs

- **Actor:** System, Admin
- **Description:** Record important events: login/logout, role changes, grade changes, publish/unpublish course, payments, etc.
- **Preconditions:** Logging service is configured and operational.
- **Postconditions:** Logs stored and queryable/exportable by authorized admins.
- **Priority:** Medium
- **Frequency of Use:** High (continuous logging)
- **Normal Course of Events:**
  1. An event occurs → system logs with timestamp, user, action, and details
  2. Admins can search, filter, and export logs
- **Alternative Courses:**
  1. Archive/rotate logs according to retention policies
- **Exceptions:** Log storage full → alert and degrade gracefully

- **Includes:** Events from other use cases
- **Extends:** N/A
- **Special Requirements:** Tamper-evident storage, retention and archival policies, strict access control for logs
- **Assumptions:** Log storage and retention strategy are defined
- **Notes and Issues:** Use logs for compliance, debugging, and forensic investigations

### 3. Tổ chức nhóm

- Nhóm đối ngoại: chuẩn bị tài liệu và báo cáo, phân tích, giao tiếp với khách hàng, quản lý, đảm bảo tiến độ của sản phẩm
  1. Trương Hoàng Phúc: Product Owner, phụ trách làm trưởng nhóm, chia công việc, và deploy sản phẩm
  2. Tạ Hoàng Hiệp, Đỗ Đình Khang: Scrum Master, làm tài liệu, phân tích, thiết kế Figma
- Nhóm đối nội: tập trung phát triển phần mềm
  1. Front-End Team (FET): Vũ Quốc Huy, Nguyễn Văn Hào — phát triển giao diện
  2. Back-End Team (BET): Bùi Văn Tùng, Dương Quốc Hưng — phát triển hệ thống
- Chia việc đầy đủ cho tất cả thành viên trên Project repo: <https://github.com/users/Huangphoux/projects/2>
- Phổ biến về việc cần nộp Pair Programming Log
- Hướng dẫn FET về công việc cần làm
- Deploy Front-End: <https://uit-limousine.netlify.app/>

### 4. Công cụ

Zalo, Discord

## 5. Hình ảnh





