CROSS-PLATFORM MOBILE APPLICATION DEVELOPMENT - 502071

SEMESTER 1 – ACADEMIC YEAR 2025 – 2026

Lecturer: Mai Van Manh

FINAL PROJECT

E-learning Management App

For content highlighted in **purple**, students must adhere strictly to the descriptions provided. Any deviation, modification, or addition of features will result in the submission being disqualified from evaluation.

I. OVERVIEW

The final project requires each team to develop a cross-platform E-learning application using Flutter, inspired by the design and interface organization of **Google Classroom** but tailored to the course scope. The application operates in a single context: the system supports only two roles—Instructor (a single user, also acting as the content administrator, with the mandatory and fixed account credentials admin/admin) and Student (multiple users). No additional roles are permitted. All academic content, courses, and data must pertain exclusively to the Faculty of Information Technology (e.g., programming, databases, AI). Any irrelevant content (e.g., cooking, sports, arts) will result in a score of 0 and the submission will not be graded.

The application must fully reflect the operational lifecycle of a digital classroom: the instructor defines semesters, creates courses, assigns groups, and manages students; the instructor publishes and distributes learning resources (announcements, assignments, quizzes, materials); the system tracks learning activities to generate statistics, reports, and CSV exports for end-of-semester evaluations; students receive notifications, track progress, participate in discussions, and submit work by deadlines. Beyond functional correctness, the submission must demonstrate practical deployability: it must include an Android APK (arm64), Windows executable (64-bit) or macOS application, and a publicly accessible web version.

Phát triển ứng dụng di động đa nền tảng - 502071

Học kỳ 1 – Năm học 2025 – 2026

Giảng viên: Mai Văn Mạnh

DỰ ÁN CUỐI KỲ

Ứng dụng quản lý học trực tuyến

Đối với nội dung được tô màu tím, sinh viên phải tuân thủ nghiêm ngặt các mô tả được cung cấp. Mọi sự lệch, chỉnh sửa hoặc bổ sung các tính năng sẽ khiến bài nộp bị loại khỏi quá trình đánh giá.

N

Dự án cuối kỳ yêu cầu mỗi nhóm phát triển một ứng dụng E-learning đa nền tảng bằng Flutter, lấy cảm hứng từ thiết kế và cách tổ chức giao diện của Google Classroom nhưng được điều chỉnh để phù hợp với phạm vi khóa học. Ứng dụng hoạt động trong một ngữ cảnh duy nhất: hệ thống chỉ hỗ trợ hai vai trò—Giảng viên (một người dùng duy nhất, đồng thời đóng vai quản trị nội dung, với thông tin đăng nhập bắt buộc và cố định là admin/admin) và Học sinh (nhiều người dùng). Không được phép có vai trò bổ sung. Tất cả nội dung học thuật, khóa học và dữ liệu phải thuộc riêng Khoa Công nghệ Thông tin (ví dụ: lập trình, cơ sở dữ liệu, AI). Bất kỳ nội dung không liên quan nào (ví dụ: nấu ăn, thể thao, nghệ thuật) sẽ bị cho 0 điểm và bài nộp sẽ không được chấm. Ứng dụng phải phản ánh đầy đủ vòng đời vận hành của một lớp học kỹ thuật số: giảng viên xác định học kỳ, tạo khóa học, phân công nhóm và quản lý sinh viên; giảng viên xuất bản và phân phối tài nguyên học tập (thông báo, bài tập, câu hỏi trắc nghiệm, tài liệu); hệ thống theo dõi hoạt động học tập để tạo ra thống kê, báo cáo và xuất CSV cho đánh giá cuối kỳ; sinh viên nhận thông báo, theo dõi tiến độ, tham gia thảo luận và nộp bài đúng hạn. Ngoài tính đúng chức năng, bài nộp phải chứng tỏ khả năng triển khai thực tế: nó phải bao gồm một APK Android (arm64), trình thực thi Windows (64-bit) hoặc ứng dụng macOS, và một phiên bản web công khai có thể truy cập được.

During grading, the web application must operate smoothly, with teams responsible for resolving any cold start issues with free backend services—proactively prepare wake-up scripts for the grading session.

II. PROJECT REQUIREMENTS

1. Interface and User Experience Design

The application should organize its interface into three distinct layers:

- Homepage with Role-Based Context:
  - For students, the homepage displays enrolled courses as cards, each featuring a cover image, course name, instructor name, and other relevant details.
  - For the instructor, the homepage transforms into a dashboard summarizing key metrics for the current semester: number of courses, groups, and students managed; number of assignments and quizzes; and progress charts for quick insights.
  - Both roles have a semester switcher in a convenient interface location (student-defined). By default, the system loads the current (latest) semester; users can switch to past semesters for reference. For students, past semesters are read-only, prohibiting assignment submissions or quiz attempts.
- Course Space with Three Tabs:
  Each course is organized into three tabs:
  - Stream: Displays recent announcements with short comment threads for quick interaction.
  - Classwork: Centralizes assignments, quizzes, and materials, organized systematically with search and sorting capabilities for large datasets.
  - People: Lists groups and students enrolled in the course.
  - Students can view all three tabs but cannot message other students directly; valid interaction channels include course/group forums and private messaging with the instructor.
- User Profile with Avatar: Both instructor and students have a Profile page to view and edit basic information (e.g., avatar, supplementary fields), but display names cannot be modified. Usernames must be real names (e.g., no “user1”, “user2”).

2. Semester–Course–Group–Student Model

The instructor is responsible for CRUD operations on core entities:

- Semester: Requires only a code and name, encompassing multiple courses.
- Course: Includes a course code, name, number of sessions (10 or 15), and belongs to a single semester.
- Group: Belongs to a single course in a specific semester; within a course, each student can belong to only one group (e.g., Web Programming & Applications course has Group 1, Group 2, Group 3).

Trong quá trình chấm điểm, ứng dụng web phải hoạt động trơn tru, với các nhóm chịu trách nhiệm giải quyết bất kỳ vấn đề khởi động lạnh nào với các dịch vụ backend miễn phí — chủ động chuẩn bị các tập lệnh đánh thức cho phiên chấm điểm.

II. YÊU CẦU DỰ ÁN

1. Thiết kế Giao diện và Trải nghiệm Người dùng

Ứng dụng nên sắp xếp giao diện thành ba lớp riêng biệt:

- Trang chủ với ngữ cảnh dựa trên vai trò:

  - Đối với sinh viên, trang chủ hiển thị các khóa học đã tham gia dưới dạng thẻ, mỗi thẻ có hình bìa, tên khóa học, tên giảng viên và các chi tiết liên quan khác.

  - Đối với giảng viên, trang chủ biến thành một bảng điều khiển tóm tắt các chỉ số chính cho học kỳ hiện tại: số lượng khóa học, nhóm và sinh viên được quản lý; số lượng bài tập và câu đố; và các biểu đồ tiến độ để có cái nhìn nhanh.

  - Cả hai vai trò đều có một công cụ chuyển đổi học kỳ ở một vị trí giao diện thuận tiện (theo định nghĩa của sinh viên). Theo mặc định, hệ thống sẽ tải học kỳ hiện tại (mới nhất); người dùng có thể chuyển sang các học kỳ trước để tham khảo. Đối với sinh viên, các học kỳ trước ở chế độ chỉ đọc, không cho phép nộp bài tập hoặc làm bài kiểm tra.

- Không gian Khóa học với ba tab:

  Mỗi khóa học được tổ chức thành ba tab:

  - Luồng: Hiển thị các thông báo gần đây với các chuỗi bình luận ngắn để tương tác nhanh.

  Bài tập ở lớp: Tập trung các bài tập, bài kiểm tra và tài liệu, được tổ chức một cách có hệ thống với khả năng tìm kiếm và sắp xếp cho tập dữ liệu lớn.

  - Người tham gia: Liệt kê các nhóm và sinh viên đã đăng ký tham gia khóa học.

  Học viên có thể xem tất cả ba tab nhưng không thể nhắn tin cho các học viên khác trực tiếp; các kênh tương tác hợp lệ bao gồm diễn đàn khóa học và diễn đàn nhóm, cũng như nhắn tin riêng với giảng viên.

- Hồ sơ người dùng có ảnh đại diện: Cả giảng viên và sinh viên đều có trang Hồ sơ để xem và chỉnh sửa thông tin cơ bản (ví dụ: ảnh đại diện, các trường bổ sung), nhưng tên hiển thị không thể chỉnh sửa. Tên người dùng phải là tên thật (ví dụ: không được dùng “user1”, “user2”).

2. Mô hình Học kỳ–Khóa học–Nhóm–Sinh viên

Người hướng dẫn chịu trách nhiệm về các thao tác CRUD trên các thực thể lõi:

is possibility of star (*) formatting. My final answer:

- Khóa học: Bao gồm mã khóa học, tên, số buổi học (10 hoặc 15), và thuộc về một học kỳ duy nhất.
- Nhóm: Thuộc về một khóa học duy nhất ở một học kỳ cụ thể; trong một khóa học, mỗi sinh viên chỉ có thể thuộc về một nhóm duy nhất (ví dụ, khóa học Web Programming& Applications có Nhóm 1, Nhóm 2, Nhóm 3).

- Student: Managed independently; the instructor creates student accounts first, then assigns them to appropriate groups. For example, in a new semester, a list of 50 students may include 40 existing accounts and 10 new ones; the system must support this process seamlessly and efficiently.

To enhance operational efficiency, all bulk import features must support CSV uploads with validation and preview. For instance, when uploading a 50-student CSV, the system must not reject the entire batch due to duplicates; instead, it should display a preview listing all 50 entries with their status ("already exists," "will be added"), allowing the user to proceed with importing only the 20 new students while skipping the 30 existing ones. A post-import results screen must show the status of each item. This preview rule applies similarly to semesters, courses, groups, and student-group assignments.

3. Distribution of Learning Content

The instructor creates and publishes four types of content, each with distinct behaviors, scopes, and tracking:

- **Announcement**: Includes a title, rich-text content, and optional file attachments. When published, the instructor selects the scope (one, multiple, or all groups in a course). Both instructor and students can comment briefly under announcements in a simplified "social media" style. The instructor can track who has viewed the announcement and, if applicable, who has downloaded attached files.
- **Assignment**: Includes a title, description, and multiple file/image attachments. The instructor sets the start date, deadline, whether late submissions are allowed (with a specified late deadline), maximum submission attempts, file format, and size limits. Assignments are scoped to groups, similar to announcements. The instructor's interface must provide real-time tracking: who has submitted, who hasn't, late submissions, multiple attempts (e.g., 2nd or 3rd submission), and current grades. All tracking tables must support filtering, searching, and sorting by name, group, time, or status, with CSV export for individual assignments or all assignments in a course/semester for final evaluations.
- **Quiz**: Built on a course-specific question bank reusable across semesters. Each question has multiple choices, one correct answer, and a difficulty label (easy, medium, hard). When creating a quiz, the instructor configures the time window (open/close), number of attempts, duration, and can randomly select a structure (e.g., x easy, y medium, z hard questions). Post-release, the system must show who has completed the quiz, who hasn't, scores, submission times, and support CSV export for individual quizzes or all quizzes in a course/semester.
- **Material**: Includes a title, description, and one or more files/links. Unlike announcements/assignments, materials are automatically visible to all students in a course, with no group scoping. The instructor can track who has viewed or downloaded materials.

4. Interaction Rules, Forums, Messaging, and Notifications

- **Forum**: Allows topic creation per course, with all enrolled students having equal discussion rights, threaded replies, file attachments, and search functionality.

• Sinh viên: được quản lý độc lập; giảng viên tạo tài khoản sinh viên trước, sau đó phân chúng vào các nhóm phù hợp. Ví dụ, trong một học kỳ mới, một danh sách 50 sinh viên có thể bao gồm 40 tài khoản hiện có và 10 tài khoản mới; hệ thống phải hỗ trợ quy trình này một cách liền mạch và hiệu quả.

Để nâng cao hiệu quả vận hành, tất cả các tính năng nhập khẩu hàng loạt phải hỗ trợ tải lên CSV kèm xác thực và xem trước. Ví dụ, khi tải lên một CSV chứa 50 sinh viên, hệ thống không được từ chối toàn bộ lô dữ liệu do trùng lặp; thay vào đó, nó nên hiển thị một bản xem trước liệt kê 50 mục kèm trạng thái của chúng (“đã tồn tại,” “sẽ được thêm”), cho phép người dùng tiếp tục nhập chỉ 20 sinh viên mới trong khi bỏ qua 30 sinh viên đã tồn tại. Một màn hình kết quả sau khi nhập xong phải hiển thị trạng thái của từng mục. Quy tắc xem trước này cũng áp dụng tương tự cho học kỳ, khóa học, nhóm và phân công sinh viên–nhóm.

3. Phân phối nội dung học tập

Người hướng dẫn tạo và xuất bản bốn loại nội dung, mỗi loại có các hành vi, phạm vi và việc theo dõi khác nhau:

• Thông báo: Bao gồm tiêu đề, nội dung văn bản được định dạng phong phú, và các tệp đính kèm tùy chọn. Khi được công bố, giảng viên chọn phạm vi (một nhóm, nhiều nhóm, hoặc tất cả các nhóm trong khóa học). Cả giảng viên và sinh viên đều có thể bình luận ngắn dưới các thông báo theo phong cách “mạng xã hội” đơn giản. Giảng viên có thể theo dõi ai đã xem thông báo và, nếu có, ai đã tải xuống các tệp đính kèm.

• Bài tập: Bao gồm tiêu đề, mô tả và nhiều tệp đính kèm là hình ảnh. Người hướng dẫn thiết lập ngày bắt đầu, hạn chót, có cho phép nộp bài muộn (với hạn chót nộp muộn được chỉ định), số lần nộp tối đa, định dạng tệp và giới hạn kích thước. Bài tập được giới hạn theo nhóm, tương tự như các thông báo. Giao diện của giảng viên phải cung cấp theo dõi thời gian thực: ai đã nộp, ai chưa nộp, nộp muộn, nhiều lần nộp (ví dụ lần nộp thứ 2 hoặc 3), và điểm hiện tại. Tất cả bảng theo dõi phải hỗ trợ lọc, tìm kiếm và sắp xếp theo tên, nhóm, thời gian hoặc trạng thái, với xuất CSV cho từng bài tập hoặc toàn bộ bài tập trong một khóa học hoặc một học kỳ cho các đánh giá cuối kỳ.

• Bài kiểm tra: Được xây dựng trên một ngân hàng câu hỏi dành riêng cho môn học có thể tái sử dụng qua các học kỳ. Mỗi câu hỏi có nhiều lựa chọn, một đáp án đúng, và nhãn độ khó (dễ, trung bình, khó). Khi tạo một bài kiểm tra, giảng viên cấu hình khoảng thời gian mở/đóng, số lượt làm, thời lượng, và có thể chọn ngẫu nhiên một cấu trúc (ví dụ: x câu dễ, y câu trung bình, z câu khó). Sau khi phát hành, hệ thống phải hiển thị ai đã hoàn thành bài kiểm tra, ai chưa, điểm số, thời gian nộp bài, và hỗ trợ xuất CSV cho từng bài kiểm tra hoặc cho tất cả các bài kiểm tra trong một khóa học/học kỳ.

• Tài liệu: Bao gồm tiêu đề, mô tả và một hoặc nhiều tệp/liên kết. Khác với thông báo/bài tập, tài liệu được tự động hiển thị cho tất cả sinh viên trong một khóa học, không giới hạn theo nhóm. Giảng viên có thể theo dõi ai đã xem hoặc tải xuống tài liệu.

4. Quy tắc Tương tác, Diễn đàn, Nhắn tin và Thông báo

• Diễn đàn: Cho phép tạo chủ đề cho từng khóa học, với tất cả sinh viên đã tham gia có quyền thảo luận bình đẳng, trả lời theo chuỗi, đính kèm tệp, và chức năng tìm kiếm.

- **Private Messaging**: Exists only between students and the instructor; direct student-to-student messaging is prohibited—students must use course/group forums for peer interaction.
- **Notifications**: The system implements in-app notifications for students with clear read/unread status; instructors do not require in-app notifications. Email notifications (for students only) must be sent for key events, such as new announcements, approaching assignment/quiz deadlines, important feedback, or confirmation of assignment/quiz submissions.

5. Search, Sorting, and Performance Optimization

Due to potentially large datasets, list screens (courses, groups, students, assignments, quizzes, materials, submissions) must support keyword search, filtering (e.g., by group, status, or time), and sorting (e.g., by name, deadline, score, or update date). The application should implement appropriate caching (e.g., caching "category" data and recent query results) to reduce API calls, minimize response times, and maintain a smooth experience under unstable network conditions. When switching semesters or refreshing data, cache synchronization must ensure no data inconsistencies.

6. Student Privileges, Personal Dashboard, and Action Restrictions

Beyond accessing resources, students must have a personal dashboard displaying learning progress: submitted, pending, or late assignments; completed quizzes with scores (if available); and a chart/timeline of upcoming deadlines. For past semesters, all actions (submitting assignments, taking quizzes, editing) are disabled, allowing only read-only access for reference.

7. Deployment

For the backend of the application, you have the freedom to decide how to implement it. However, for the front end (Flutter), you need to provide the following: an APK version for Android (arm64), a 64-bit version for Windows/macOS, and deploy the web version to an online hosting service (e.g., Firebase Hosting or GitHub Pages). Afterward, you should provide the URL of the website in your submission. The web deployment section, if successful, will earn you 0.5 points. Although the Windows/macOS and APK versions do not contribute to the score, they are still mandatory. If you only submit the source code without providing any working deployment versions, or if they are faulty and cannot run, your submission will not be graded.

8. Other requirements

- UI/UX: The web app must have a clear, user-friendly design with intuitive navigation. Focus on user experience, quick load times, and easy interaction with elements.
- Responsive Design: The web app should be responsive, adapting seamlessly to different devices and screen sizes. Use frameworks like Bootstrap or CSS Grid to ensure compatibility.

- **Private Messaging**: Exists only between students and the instructor; direct student-to-student messaging is prohibited—students must use course/group forums for peer interaction.
- **Notifications**: The system implements in-app notifications for students with clear read/unread status; instructors do not require in-app notifications. Email notifications (for students only) must be sent for key events, such as new announcements, approaching assignment/quiz deadlines, important feedback, or confirmation of assignment/quiz submissions.

5. Search, Sorting, and Performance Optimization

Due to potentially large datasets, list screens (courses, groups, students, assignments, quizzes, materials, submissions) must support keyword search, filtering (e.g., by group, status, or time), and sorting (e.g., by name, deadline, score, or update date). The application should implement appropriate caching (e.g., caching "category" data and recent query results) to reduce API calls, minimize response times, and maintain a smooth experience under unstable network conditions. When switching semesters or refreshing data, cache synchronization must ensure no data inconsistencies.

6. Student Privileges, Personal Dashboard, and Action Restrictions

Beyond accessing resources, students must have a personal dashboard displaying learning progress: submitted, pending, or late assignments; completed quizzes with scores (if available); and a chart/timeline of upcoming deadlines. For past semesters, all actions (submitting assignments, taking quizzes, editing) are disabled, allowing only read-only access for reference.

7. Deployment

For the backend of the application, you have the freedom to decide how to implement it. However, for the front end (Flutter), you need to provide the following: an APK version for Android (arm64), a 64-bit version for Windows/macOS, and deploy the web version to an online hosting service (e.g., Firebase Hosting or GitHub Pages). Afterward, you should provide the URL of the website in your submission. The web deployment section, if successful, will earn you 0.5 points. Although the Windows/macOS and APK versions do not contribute to the score, they are still mandatory. If you only submit the source code without providing any working deployment versions, or if they are faulty and cannot run, your submission will not be graded.

8. Other requirements

- UI/UX: The web app must have a clear, user-friendly design with intuitive navigation. Focus on user experience, quick load times, and easy interaction with elements.
- Responsive Design: The web app should be responsive, adapting seamlessly to different devices and screen sizes. Use frameworks like Bootstrap or CSS Grid to ensure compatibility.

- Team Collaboration: Team members must work together using version control (e.g., Git), dividing tasks and ensuring smooth integration of contributions. Regular communication is essential. You are required to demonstrate the group work process by capturing screenshots of team members' contributions using the **GitHub Insights** feature. The evidence must clearly show that the project has been ongoing for at least one month from its start date, with each member making a minimum of two commits per week. Teamwork is **mandatory**, so if the team has only one member or lacks evidence of collaboration, **0.5** points will be deducted from the total score.
- Offline Capability: The application must support an offline mode to ensure functionality when the user has no internet connection. At a minimum, students should be able to view previously accessed course materials, announcements, and their personal dashboard (including submitted assignments, completed quizzes with scores, and upcoming deadlines). The instructor should be able to view previously accessed course data, student lists, and tracking metrics (e.g., who has viewed materials or submitted assignments). To achieve this, the application must integrate both an online database and an offline database (e.g., SQLite, Hive). The offline database should maintain a synchronized copy of critical data from the online database, enabling faster access or serving as a fallback when the internet is unavailable.

9. Bonus features

To encourage technical depth, teams may implement the following extensions, with descriptions and evidence placed in a bonus/ directory. Each well-executed feature earns 0.25–0.5 points, with a maximum bonus of 4 features:

- Self-built backend (without using Firebase or similar services).
- Microservices architecture with message queues (e.g., RabbitMQ/Redis) and/or Kubernetes for deployment.
- AI chatbot for learning support.
- AI-driven question and answer generation for quizzes from instructor materials, with parameterized difficulty and minimal validity checks.
- Other AI features that enhance learning or teaching (with quantitative/qualitative evidence and clear descriptions, prominently showcased in the final demo video; final bonus points are at the instructor's discretion).
- Publishing the application on Google Play, Apple App Store, or Microsoft Store.

How to determine bonus points (0.25 vs. 0.5): For example, for the AI chatbot, simply calling OpenAI/Gemini APIs without customization, contextualization, or specialized testing for learning support earns only 0.25 points. To achieve 0.5 points, the chatbot must be designed and validated for the specific context (e.g., using RAG or fine-tuning the model for learning support, with evidence via specific examples or benchmarks).

If your team successfully implemented any of these bonus features, you should clearly state it in the self-evaluation form, the README file, the product introduction video along with specific, convincing evidence. This ensures that your contributions are recognized and properly communicated to reviewers.

• Hợp tác nhóm: Thành viên trong nhóm phải làm việc cùng nhau bằng cách sử dụng kiểm soát phiên bản (ví dụ Git), phân chia nhiệm vụ và đảm bảo tích hợp đóng góp một cách suôn sẻ. Việc giao tiếp thường xuyên là rất cần thiết. Bạn được yêu cầu thể hiện quá trình làm việc nhóm bằng cách chụp ảnh màn hình đóng góp của các thành viên trong nhóm bằng tính năng GitHub Insights. Bằng chứng phải cho thấy rõ dự án đã được tiến hành ít nhất một tháng kể từ ~~ngày bắt đầu,~~ với mỗi thành viên thực hiện tối thiểu hai lần commit mỗi tuần. Làm việc nhóm là bắt buộc, nên nếu đội ngũ chỉ có một thành viên hoặc thiếu bằng chứng cho sự hợp tác, 0,5 điểm sẽ bị trừ khỏi tổng điểm.

• Khả năng làm việc ngoại tuyến: Ứng dụng phải hỗ trợ chế độ ngoại tuyến để đảm bảo chức năng khi người dùng không có kết nối Internet. Ít nhất, sinh viên nên có thể xem các tài liệu khóa học đã truy cập trước đó, các thông báo và bảng điều khiển cá nhân của họ (bao gồm bài tập đã nộp, các bài kiểm tra đã hoàn thành với điểm số và các hạn nộp sắp tới). Giảng viên nên có thể xem dữ liệu khóa học đã truy cập trước đó, danh sách sinh viên và các chỉ số theo dõi (ví dụ: ai đã xem tài liệu hoặc đã nộp bài tập). Để đạt được điều này, ứng dụng phải tích hợp cả cơ sở dữ liệu trực tuyến và cơ sở dữ liệu ngoại tuyến (ví dụ: SQLite, Hive). Cơ sở dữ liệu ngoại tuyến nên duy trì một bản sao đồng bộ của dữ liệu quan trọng từ cơ sở dữ liệu trực tuyến, cho phép truy cập nhanh hơn hoặc làm phương án dự phòng khi kết nối Internet không có.

9. Các tính năng bổ sung

Để thúc đẩy chiều sâu kỹ thuật, các đội có thể triển khai các phần mở rộng sau đây, với mô tả và bằng chứng được đặt trong thư mục bonus/. Mỗi tính năng được thực thi tốt sẽ được cộng 0,25–0,5 điểm, với tối đa 4 tính năng được thưởng:

- Backend tự xây dựng (không sử dụng Firebase hoặc các dịch vụ tương tự).
- Kiến trúc microservices với hàng đợi tin nhắn (ví dụ: RabbitMQ/Redis) và/hoặc Kubernetes để triển khai.
- Trợ lý trò chuyện AI cho hỗ trợ học tập.
- Tạo câu hỏi và đáp án cho các bài kiểm tra từ tài liệu của giảng viên, được AI hỗ trợ, với độ khó được tham số hóa và kiểm tra tính hợp lệ tối thiểu.
- Các tính năng AI khác nhằm nâng cao việc học tập hoặc giảng dạy (kèm theo bằng chứng định lượng/định tính và mô tả rõ ràng, được trình diễn nổi bật trong video demo cuối cùng; điểm thưởng cuối cùng do giảng viên quyết định).
- Đăng tải ứng dụng trên Google Play, Apple App Store hoặc Microsoft Store.

Cách xác định điểm thưởng (0.25 so với 0.5): Ví dụ, đối với chatbot AI, chỉ cần gọi API OpenAI/Gemini mà không tùy chỉnh, ngữ cảnh hóa hoặc kiểm tra chuyên sâu cho hỗ trợ học tập sẽ chỉ được 0.25 điểm. Để đạt 0.5 điểm, chatbot phải được thiết kế và xác thực cho ngữ cảnh cụ thể (ví dụ, sử dụng RAG hoặc tinh chỉnh mô hình cho hỗ trợ học tập, với bằng chứng thông qua các ví dụ cụ thể hoặc các bài kiểm tra chuẩn).

Nếu đội ngũ của bạn đã triển khai thành công bất kỳ tính năng thưởng nào này, bạn nên nêu rõ điều đó trong mẫu tự đánh giá, tệp README, video giới thiệu sản phẩm, cùng với các bằng chứng cụ thể và thuyết phục. Điều này đảm bảo đóng góp của bạn được công nhận và thông báo đúng tới người đánh giá.

10. Notes

The description provided above is intended as a general guideline and cannot be considered as a detailed step-by-step implementation for each feature, specifying what is right or wrong. However, during the grading process, the features must be developed based on these descriptions, particularly those highlighted in a distinct color. These features must be implemented to a relatively high standard in order to qualify for maximum points. Teams should actively refer to related applications and apply their daily user experience with such applications to the task. For example:

- **When displaying a list of courses (or other resources)**:
  - **Poor approach**: Only display the information available in the Course table of the database, even showing the createdAt field in a raw machine-like format (e.g., 2025-08-16T13:45:30.000Z).
  - **Better approach**: Display not only the information from the Course table with a user-friendly format (e.g., 15/08/2025), but also include related information that enhances usability (e.g., the number of groups belonging to the course, the total number of students enrolled in the course).
- **When displaying a list of courses (or other resources)**:
  - **Poor approach**: Always show a loading screen while fetching data from the API, even when the user revisits the same screen. This forces repeated waits and unnecessary API calls.
  - **Better approach**: Use a user-friendly loading indicator (e.g., skeleton screen) combined with local caching, so data only needs to load once and is refreshed only when updated. This reduces waiting time and API usage.
- **When adding a new student**:
  - **Poor approach**: Only allow adding through a form or uploading a CSV in a simplistic way. If the CSV happens to be in the correct format, it uploads successfully; if not, the outcome is uncertain. After a successful upload, the system merely displays a generic message such as “Upload completed.”
  - **Better approach**: Support adding students via both the form and CSV upload. On the CSV upload interface, provide clear instructions so users know the correct column order and whether headers are required. Once a file is selected, show a preview of the parsed results: highlight missing or incorrectly formatted fields, and list which students already exist and which do not. Allow the user to proceed only if the file is valid. Upon confirming the upload, create accounts only for students who do not already exist, skip duplicates, and display a summary screen so the user clearly understands the import results.
- **When attaching image files**:
  - **Poor approach**: After selecting, the system only indicates that one image has been chosen, without letting the user verify whether the correct file was selected.

10. Ghi chú

Mô tả được cung cấp ở trên được xem như một hướng dẫn chung và không thể được coi là một triển khai chi tiết theo từng bước cho mỗi tính năng, xác định cái đúng hay sai. Tuy nhiên, trong quá trình chấm điểm, các tính năng phải được phát triển dựa trên các mô tả này, đặc biệt là những tính năng được làm nổi bật bằng một màu khác biệt. Những tính năng này phải được triển khai ở mức độ tương đối cao để đủ điều kiện nhận điểm tối đa. Các đội nên tích cực tham khảo các ứng dụng liên quan và áp dụng kinh nghiệm người dùng hàng ngày của họ với các ứng dụng đó vào nhiệm vụ. Ví dụ:

nguyên khác):

- o Cách tiếp cận kém: Chỉ hiển thị thông tin có sẵn trong bảng Course của cơ sở dữ liệu, thậm chí hiển thị trường createdAt ở định dạng giống rawmachine (ví dụ: 2025-08-16T13:45:30.000Z).
- o Cách tiếp cận tốt hơn: Hiển thị không chỉ thông tin từ bảng Course ở định dạng thân thiện với người dùng (ví dụ: 15/08/2025), mà còn bao gồm các thông tin liên quan nhằm tăng cường khả năng sử dụng (ví dụ: số nhóm thuộc khóa học, tổng số sinh viên đã đăng ký khóa học).

• Khi hiển thị một danh sách các khóa học (hoặc các nguồn lực khác):

- o Cách tiếp cận kém: Luôn hiển thị màn hình chờ trong khi tải dữ liệu từ API, ngay cả khi người dùng quay lại cùng một màn hình. Điều này khiến người dùng phải chờ đợi lặp đi lặp lại và thực hiện các cuộc gọi API không cần thiết.
- o Cách tiếp cận tốt hơn: Sử dụng một chỉ báo tải dễ sử dụng (ví dụ: màn hình khung xương) kết hợp với bộ nhớ đệm cục bộ, để dữ liệu chỉ cần tải một lần và được làm mới chỉ khi có cập nhật. Điềũ làm giảm thời gian chờ và việc sử dụng API.

• Khi thêm một học sinh mới:

- o Phương pháp kém: Chỉ cho phép thêm thông qua một biểu mẫu hoặc tải lên một tệp CSV theo cách đơn giản. Nếu tệp CSV tình cờ ở đúng định dạng, nó tải lên thành công; nếu không, kết quả sẽ không chắc chắn. Sau khi tải lên thành công, hệ thống chỉ hiển thị một thông báo chung như “Đã tải lên xong.”
- o Cách tiếp cận tốt hơn: Hỗ trợ thêm sinh viên thông qua cả mẫu đơn và tải lên CSV. Trên giao diện tải lên CSV, cung cấp hướng dẫn rõ ràng để người dùng biết thứ tự các cột đúng và liệu tiêu đề có bắt buộc hay không. Khi một tệp được chọn, hiển thị trước kết quả phân tích: làm nổi bật các trường bị thiếu hoặc định dạng sai, và liệt kê những sinh viên đã tồn tại và những sinh viên chưa tồn tại. Cho phép người dùng tiếp tục chỉ khi tệp hợp lệ. Sau khi xác nhận tải lên, tạo tài khoản chỉ cho những sinh viên chưa tồn tại, bỏ qua trùng lặp, và hiển thị một màn hình tổng kết để người dùng hiểu rõ kết quả nhập khẩu.

• Khi đính kèm các tập tin hình ảnh:

- o Cách tiếp cận kém: Sau khi chọn, hệ thống chỉ cho biết đã chọn một hình ảnh, mà không cho phép người dùng xác nhận xem tệp đúng đã được chọn hay chưa.

- **Better approach**: For formats that can be previewed (e.g., images, text files, etc.), allow the user to view them directly in the app by clicking on the file, instead of forcing them to download it first. Furthermore, if multiple images are included, enable swipe navigation to quickly move between files, rather than requiring the user to close one image and click another to view it.

- **Other examples**: students may look up and study further by themselves.

II. RUBRIK

Category	Points	25% (Low Achievement)	25%–75% (Moderate Achievement)	75%–100% (High Achievement)
Semester, Course, Group, and Student Management	**2.0**	Basic CRUD operations are implemented for only 1–2 entities (e.g., semester or course), with significant errors (e.g., unable to add students or groups). CSV upload is either missing or non-functional, failing to import data correctly. Student management lacks independent account creation, and group assignment is broken or causes crashes. User interface for these operations is confusing or incomplete, making it unusable for practical purposes.	CRUD operations are implemented for most entities (semester, course, group, student), but with noticeable limitations or errors (e.g., CSV import processes data incorrectly or lacks preview functionality). Student management supports independent account creation, but group assignment may have minor issues (e.g., duplicate entries or slow processing). CSV upload works partially, with preview or import functionality incomplete, causing occasional errors. Interface is functional but may have minor usability issues, such as unclear error messages or slow response times.	Full CRUD operations are seamlessly implemented for all entities (semester, course, group, student) with no errors. CSV upload includes robust preview and smart import functionality, correctly handling duplicates and providing clear feedback on import status. Student management allows independent account creation and efficient group assignment, with intuitive workflows. Interface is user-friendly, with clear navigation, error handling, and fast response times, ensuring practical usability for both instructors and students.
Content Delivery	**2.0**	Only 1–2 content types (e.g., announcement or assignment) are partially functional, with critical errors (e.g., unable to attach files or scope content to groups). Question bank for quizzes is either missing or unusable, with no search or difficulty tagging. View/download tracking is non-existent, and content delivery fails to meet basic requirements (e.g., materials not accessible to students). Interface for content management is broken or severely limited, preventing practical use.	3–4 content types (announcement, assignment, quiz, material) are implemented, but with missing secondary features (e.g., no view/download tracking, limited question bank search, or partial group scoping). File attachments work but may have errors (e.g., size limits not enforced). Question bank supports basic tagging but lacks advanced features like cross-semester reuse. Interface is functional but has minor issues, such as slow loading of content or inconsistent display of attachments.	All 4 content types (announcement, assignment, quiz, material) are fully functional, with seamless group scoping, file attachments, and view/download tracking. Question bank is robust, supporting search, difficulty tagging, and cross-semester reuse. Content delivery is reliable, with clear options for instructors to set deadlines, file formats, and submission limits. Interface is intuitive, with fast loading, clear organization, and no errors, ensuring a smooth experience for all users.

o Cách tiếp cận tốt hơn: Đối với các định dạng có thể xem trước (ví dụ: hình ảnh, tệp văn bản, v.v.), cho phép người dùng xem trực tiếp chúng trong ứng dụng bằng cách nhấp vào tệp, thay vì bắt buộc họ phải tải xuống trước. Hơn nữa, nếu có nhiều hình ảnh được đưa vào, hãy cho phép điều hướng bằng vuốt để nhanh chóng di chuyển giữa các tệp, thay vì yêu cầu người dùng đóng một hình ảnh và nhấp vào hình ảnh khác để xem.

- Các ví dụ khác: học sinh có thể tự tra cứu và tự học thêm.

II. TIÊU ĐỀ

Danh mục		25% (Thành tích thấp)	25%–75% (Thành tích ở mức trung bình)	75%–100% (Thành tích cao)
Học kỳ, Khóa học, Nhóm, và Quản lý sinh viên	2.0	Các thao tác CRUD cơ bản được triển khai cho chỉ 1–2 thực thể (ví dụ: học kỳ hoặc khóa học), với những lỗi nghiêm trọng (ví dụ: không thể thêm sinh viên hoặc nhóm). Việc tải lên CSV bị thiếu hoặc không hoạt động, khiến việc nhập dữ liệu không chính xác. Quản lý sinh viên thiếu khả năng tạo tài khoản độc lập, và phân công nhóm bị hỏng hoặc gây treo. Giao diện người dùng cho các thao tác này gây khó hiểu hoặc thiếu sót, khiến nó không thể sử dụng được cho mục đích thực tế.	Các thao tác CRUD được triển khai cho hầu hết các thực thể (học kỳ, khóa học, nhóm, sinh viên), nhưng có những hạn chế hoặc lỗi đáng chú ý (ví dụ, quá trình nhập CSV xử lý dữ liệu không chính xác hoặc thiếu chức năng xem trước). Quản lý sinh viên hỗ trợ tạo tài khoản độc lập, nhưng việc phân công nhóm có thể gặp phải một số vấn đề nhỏ (ví dụ, trùng lặp mục hoặc xử lý chậm). Việc tải lên CSV hoạt động ở mức độ một phần, với chức năng xem trước hoặc nhập khẩu chưa hoàn chỉnh, gây ra một số lỗi thỉnh thoảng. Giao diện hoạt động nhưng có thể gặp các vấn đề về khả năng sử dụng ở mức nhỏ, chẳng hạn như thông báo lỗi không rõ ràng hoặc thời gian phản hồi chậm.	Toàn bộ các thao tác CRUD đầy đủ được triển khai liền mạch cho tất cả các thực thể (học kỳ, khóa học, nhóm, sinh viên) mà không gặp lỗi. Việc tải lên CSV bao gồm xem trước mạnh mẽ và chức năng nhập dữ liệu thông minh, xử lý trùng lặp một cách chính xác và cung cấp phản hồi rõ ràng về trạng thái nhập. Quản lý sinh viên cho phép tạo tài khoản riêng biệt và phân nhóm hiệu quả, với các quy trình làm việc trực quan. Giao diện người dùng thân thiện, với điều hướng rõ ràng, xử lý lỗi và thời gian phản hồi nhanh, đảm bảo tính khả dụng thực tế cho cả giảng viên và sinh viên.
Phân phối nội dung	2.0	Chỉ 1–2 loại nội dung (ví dụ: thông báo hoặc bài tập) có chức năng một phần, kèm các lỗi nghiêm trọng (ví dụ: không thể đính kèm tệp hoặc gán nội dung cho các nhóm). Ngân hàng câu hỏi cho các bài kiểm tra bị thiếu hoặc không thể sử dụng, không có chức năng tìm kiếm hoặc gắn thẻ theo mức độ khó. Việc theo dõi lượt xem/tải xuống không tồn tại, và việc phân phối nội dung không đáp ứng các yêu cầu cơ bản (ví dụ: tài liệu không thể tiếp cận được với sinh viên). Giao diện quản lý nội dung bị hỏng hoặc bị giới hạn nghiêm trọng, khiến cho việc sử dụng thực tế bị hạn chế.	(assignment, quiz, material) được triển khai, nhưng thiếu các tính năng phụ (ví dụ: không có theo dõi lượt xem/tải xuống, tìm kiếm trong ngân hàng câu hỏi bị giới hạn, hoặc phạm vi nhóm một phần). Tệp đính kèm hoạt động nhưng có thể gặp lỗi (ví dụ: giới hạn kích thước không được áp dụng). Ngân hàng câu hỏi hỗ trợ gắn thẻ cơ bản nhưng thiếu các tính năng nâng cao như tái sử dụng qua các học kỳ. Giao diện hoạt động nhưng có vài vấn đề nhỏ, chẳng hạn tải nội dung chậm hoặc hiển thị tệp đính kèm không nhất quán.	Cả 4 loại nội dung (thông báo, bài tập, quiz, tài liệu) đều hoạt động đầy đủ, với phạm vi nhóm liền mạch, tệp đính kèm và theo dõi lượt xem/tải xuống. Ngân hàng câu hỏi mạnh mẽ, hỗ trợ tìm kiếm, gắn thẻ độ khó và tái sử dụng giữa các học kỳ. Việc phân phối nội dung đáng tin cậy, với các tùy chọn rõ ràng cho giảng viên thiết lập thời hạn, định dạng tệp và giới hạn nộp bài. Giao diện trực quan, với tải nhanh, tổ chức rõ ràng và không có lỗi, đảm bảo trải nghiệm mượt mà cho mọi người dùng.

Interaction and Notifications	**2.0**	Interaction features (forum, private chat) or notifications (in-app/email) are largely non-functional or have critical errors (e.g., students can chat with each other, violating rules, or email notifications fail to send). Forum lacks threading or file attachment capabilities, and in-app notifications are either missing or do not display read/unread status. System is unusable for communication, with frequent crashes or incorrect behavior.	Forum and private chat (student-instructor only) are implemented but lack key features (e.g., forum search is missing, or email notifications are inconsistent for events like deadlines). In-app notifications work but may have issues, such as missing read/unread status or delayed updates. Interaction features are functional but have minor errors, such as slow loading of forum threads or occasional notification failures. Interface is usable but may cause confusion due to inconsistent behavior.	Forum and private chat (student-instructor only) are fully functional, with threaded discussions, file attachments, and search capabilities in the forum. In-app notifications are reliable, with clear read/unread status and real-time updates. Email notifications are sent consistently for key events (e.g., new announcements, deadlines, submission confirmations) with no failures. Interface is intuitive, with fast and error-free interaction, ensuring effective communication for all users.
Reports and Analytics	**2.0**	Dashboard is either missing or severely limited, showing no meaningful metrics for instructors or students (e.g., no progress tracking or assignment status). CSV export is non-functional or produces unusable files (e.g., incorrect data or formatting). Search/filter/sort features are absent, and progress charts are either missing or broken, making analytics unusable for practical purposes.	Dashboard is implemented but lacks some metrics (e.g., incomplete progress tracking or missing charts for instructors). CSV export works but has minor errors (e.g., missing columns or incorrect data for some entries). Search/filter/sort features are present but slow or incomplete (e.g., limited filtering options). Progress charts are basic and may have display issues, but the system is partially usable for analytics.	Comprehensive dashboard for both instructors and students, displaying all required metrics (e.g., assignment/quiz status, progress tracking) with intuitive progress charts. CSV export is accurate, supporting individual assignments/quizzes and course/semester summaries. Search/filter/sort functions are fast and comprehensive, with no errors. Analytics interface is user-friendly, visually clear, and fully supports end-of-semester reporting.
Teamwork (GitHub Insights)	**0.5**	No GitHub Insights provided, or commits are limited to 1–2 members with minimal activity (e.g., fewer than 2 commits total). Commits are trivial (e.g., formatting changes) or lack descriptions, showing no evidence of task distribution or collaborative effort. Alternative formats (e.g., text files, screenshots) are used instead of GitHub Insights.	GitHub Insights is provided, but commits are uneven across members (e.g., some members have <2 commits/week) or lack clear task distribution. Commit messages are vague or inconsistent, making it difficult to assess individual contributions. Team collaboration is evident but lacks consistency or organization over the required one-month period.	GitHub Insights is fully provided, showing consistent activity (≥2 commits/week/member) over at least one month. Commits have clear, descriptive messages reflecting specific tasks (e.g., "implemented CSV import," "fixed UI for course page"). Task distribution is well-organized, with balanced contributions from all members, demonstrating strong teamwork.
Web Deployment	**0.5**	Web version is inaccessible, crashes frequently, or fails to support core functionalities (e.g., cannot view courses, submit assignments, or access	Web version is accessible but has minor issues, such as slow loading, UI glitches, or occasional errors in core functionalities (e.g., some features work inconsistently). Cold start issues	Web version is publicly accessible, runs smoothly without cold start issues, and fully supports all core functionalities (e.g., course management, content delivery, notifications). A script or clear

Tương tác và Thông báo	2.0	Các tính năng tương tác (diễn đàn, trò chuyện riêng) hoặc thông báo (trong ứng dụng/email) hầu như không hoạt động hoặc gặp lỗi nghiêm trọng (ví dụ: học sinh có thể trò chuyện với nhau, vi phạm quy tắc, hoặc email thông báo không gửi được). Diễn đàn thiếu khả năng phân luồng cuộc thảo luận hoặc đính kèm tệp, và thông báo trong ứng dụng (in-app) hoặc không có hoặc không hiển thị trạng thái đã đọc/chưa đọc. Hệ thống không thể sử dụng được cho việc liên lạc, với các lần treo máy thường xuyên hoặc hành vi không chính xác.	Diễn đàn và trò chuyện riêng (chỉ dành cho sinh viên và giảng viên) đã được triển khai nhưng thiếu các tính năng cốt lõi (ví dụ, tính năng tìm kiếm trên diễn đàn bị thiếu, hoặc email thông báo không nhất quán đối với các sự kiện như hạn chót). Thông báo trong ứng dụng hoạt động nhưng có thể gặp sự cố, chẳng hạn như thiếu trạng thái đã đọc/chưa đọc. Trạng thái hoặc cập nhật bị chậm. Các tính năng tương tác hoạt động nhưng có lỗi nhỏ, ví dụ như tải chậm các chủ đề diễn đàn hoặc một số thông báo thỉnh thoảng bị lỗi. Giao diện có thể dùng được nhưng có thể gây nhầm lẫn do hành vi không nhất quán.	Diễn đàn và trò chuyện riêng tư (chỉ dành cho sinh viên và giảng viên) hoạt động đầy đủ, với các cuộc thảo luận có chủ đề theo luồng, tệp đính kèm và khả năng tìm kiếm trong diễn đàn. Thông báo trong ứng dụng đáng tin cậy, với trạng thái đã đọc/chưa đọc rõ ràng và cập nhật thời gian thực. Thông báo qua email được gửi đều đặn cho các sự kiện quan trọng (ví dụ: thông báo mới, thời hạn, xác nhận nộp bài) mà không gặp lỗi. Giao diện trực quan, với tương tác nhanh và không gặp lỗi, đảm bảo giao tiếp hiệu quả cho mọi người dùng.
Báo cáo và Phân tích	2.0	Dashboard bị thiếu hoặc bị giới hạn nghiêm trọng, không hiển thị các chỉ số có ý nghĩa cho giảng viên hoặc sinh viên (ví dụ: không theo dõi tiến độ hoặc trạng thái bài tập). Việc xuất CSV không hoạt động hoặc tạo ra các tệp không dùng được (ví dụ: dữ liệu hoặc định dạng sai). Các tính năng tìm kiếm, lọc và sắp xếp đều thiếu, và các biểu đồ tiến độ cũng bị thiếu hoặc bị hỏng, khiến phân tích trở nên không thể sử dụng cho mục đích thực tế.	Bảng điều khiển đã được triển khai nhưng vẫn thiếu một số chỉ số (ví dụ: theo dõi tiến độ chưa đầy đủ hoặc thiếu biểu đồ cho giảng viên). Việc xuất CSV hoạt động nhưng có một số lỗi nhỏ (ví dụ: thiếu cột hoặc dữ liệu không chính xác cho một số mục). Các tính năng tìm kiếm, lọc và sắp xếp có mặt nhưng chậm hoặc chưa đầy đủ (ví dụ: tùy chọn lọc bị giới hạn). Biểu đồ tiến độ ở mức cơ bản và có thể gặp sự cố hiển thị, nhưng hệ thống vẫn có thể sử dụng được một phần cho phân tích.	Bảng điều khiển tổng hợp cho cả giảng viên và sinh viên, hiển thị đầy đủ các chỉ số yêu cầu (ví dụ: trạng thái bài tập/kiểm tra, theo dõi tiến độ) với các biểu đồ tiến độ trực quan. Việc xuất CSV chính xác, hỗ trợ các bài tập/kiểm tra cá nhân và tổng kết khóa học/học kỳ. Các chức năng tìm kiếm/lọc/sắp xếp nhanh và toàn diện, không có lỗi. Giao diện phân tích số liệu thân thiện với người dùng, trực quan và hoàn toàn hỗ trợ báo cáo cuối học kỳ.
Làm việc nhóm (GitHubInsights)	0.5	Không có GitHub Insights được cung cấp, hoặc các lần commit bị giới hạn cho 1–2 thành viên với hoạt động tối thiểu (ví dụ, tổng số commit ít hơn 2). Các lần commit rất đơn sơ (ví dụ, thay đổi định dạng) hoặc thiếu mô tả, cho thấy không có bằng chứng về phân công công việc hoặc nỗ lực hợp tác. Các định dạng thay thế (ví dụ, tệp văn bản, ảnh chụp màn hình) được sử dụng thay cho GitHub Insights.	GitHub Insights được cung cấp, nhưng số lượng commit giữa các thành viên không đồng đều (ví dụ, một số thành viên có ít hơn 2 commit mỗi tuần) hoặc thiếu phân công nhiệm vụ rõ ràng. Các thông điệp commit mơ hồ hoặc không nhất quán, khiến việc đánh giá đóng góp của từng cá nhân trở nên khó khăn. Sự hợp tác nhóm được thể hiện nhưng thiếu tính nhất quán hoặc tổ chức trong khoảng thời gian một tháng được yêu cầu.	GitHub Insights được cung cấp đầy đủ, cho thấy hoạt động nhất quán (≥2 lần commit mỗi tuần cho mỗi thành viên) trong ít nhất một tháng. Các commit có tin nhắn rõ ràng, mô tả phản ánh các nhiệm vụ cụ thể (ví dụ: "triển khai nhập CSV," "sửa giao diện người dùng cho trang khóa học"). Phân bổ công việc được tổ chức tốt, với sự đóng góp cân bằng từ tất cả các thành viên, cho thấy tinh thần làm việc nhóm mạnh mẽ.
Triển khai Web	0,5	Phiên bản web không thể truy cập được, thường xuyên gặp sự cố, hoặc không hỗ trợ các chức năng cốt lõi (ví dụ: không thể xem các khóa học, nộp bài tập, hoặc truy cập	Phiên bản web có thể truy cập được nhưng gặp phải một số vấn đề nhỏ, chẳng hạn như tải chậm, lỗi giao diện người dùng, hoặc các lỗi thỉnh thoảng trong các chức năng cốt lõi (ví dụ: một số tính năng hoạt động không nhất quán). Các vấn đề khởi động lần đầu.	Phiên bản web có thể truy cập công khai, chạy trơn tru mà không gặp vấn đề khởi động lạnh, và đầy đủ hỗ trợ tất cả các chức năng cốt lõi (ví dụ: quản lý khóa học, phân phối nội dung, thông báo). A script or clear

		dashboards). No script or instructions provided for cold start mitigation, making the system unusable during grading. Deployment is effectively non-existent or broken.	require manual intervention, and the deployment may not fully support all required features. Documentation for deployment is incomplete or unclear.	instructions for service wake-up is provided, ensuring immediate access during grading. Deployment is reliable and user-friendly.
User Interface (UI) Scored based on the teacher's perception; only above-average work earns points (a basic interface gets no points)	**0.5**	UI lacks critical components (e.g., no homepage, missing course tabs, or no profile page) or is not responsive (e.g., unusable on mobile or desktop). Design is inconsistent, with broken layouts, missing images, or unreadable text, making the application visually unusable.	UI includes most required components (homepage with role-based views, 3-tab course structure, profile page) but has minor display issues (e.g., misaligned elements, poor scaling on tablets). Responsive design is partially implemented, with some devices showing suboptimal visuals. Design is functional but lacks polish or consistency.	UI is visually appealing, with all required components (homepage with role-based views, 3-tab course structure, profile page) implemented correctly. Fully responsive across mobile, tablet, and desktop, with consistent design, clear layouts, and no display errors. Visuals enhance usability and align with project requirements.
User Experience (UX) Scored based on the teacher's perception; only above-average work earns points (a basic interface gets no points)	**0.5**	UX is poor, with confusing navigation, frequent crashes, or critical errors (e.g., incorrect data display, broken semester switching). Search/filter/sort features are non-functional or cause errors. Users cannot complete basic tasks without significant frustration, rendering the system impractical.	UX is functional but has noticeable issues, such as slow search/filter/sort, clunky semester switching, or minor cache synchronization errors. Users can complete tasks, but the experience is inconsistent or inconvenient (e.g., delayed responses or unclear error messages). Basic usability is achieved but not optimized.	UX is smooth and intuitive, with fast, accurate search/filter/sort and seamless semester switching. Cache synchronization is reliable, preventing data mismatches. All interactions are user-friendly, with clear feedback and no errors, ensuring an efficient and pleasant experience for both instructors and students.

III. OUTPUT REQUIREMENTS

- Required submission components include:
  - The "**source**" folder: Contains the entire source code of the Flutter application, web server, and API if applicable, along with relevant database files. Ensure that this source code can be compiled and run on the teacher's computer. The project needs to be "cleaned" to remove unnecessary content before submission and to reduce the size of the compressed file.
  - The "**bin**" folder: This folder should contain executable files for at least two platforms, including Android (APK) and Windows (EXE). Teachers will use these files for installation on their real/virtual machines for assessment. If necessary, teachers can choose to build the source code from scratch.
  - The demo video, titled "**demo.mp4**" must include the participation of all team members to introduce the group's product. The video should provide a brief overview of the technologies and architecture of the web application. Following this, the team must sequentially demonstrate each feature developed. Any criteria not demonstrated in the video will be considered as not implemented, even if the

		các bảng điều khiển). Không có tập lệnh hoặc hướng dẫn được cung cấp để giảm thiểu khởi động lạnh, khiến hệ thống không thể sử dụng trong quá trình chấm điểm. Việc triển khai gần như không tồn tại hoặc bị hỏng.	Yêu cầu sự can thiệp thủ công, và việc triển khai có thể không hỗ trợ đầy đủ tất cả các tính năng cần thiết. Tài liệu triển khai không đầy đủ hoặc không rõ.	Hướng dẫn đánh thức dịch vụ được cung cấp, đảm bảo truy cập ngay trong quá trình chấm điểm. Việc triển khai đáng tin cậy và thân thiện với người dùng.
Giao diện người dùng (UI) Điểm được dựa trên nhận thức của giáo viên; chỉ những công việc ở mức trên trung bình mới được điểm (một giao diện cơ bản sẽ không được điểm).	0,5	Giao diện người dùng thiếu các thành phần quan trọng (ví dụ: không có trang chủ, các tab khóa học bị thiếu, hoặc trang hồ sơ không có) hoặc không đáp ứng được (ví dụ: không thể sử dụng trên điện thoại di động hoặc máy tính để bàn). Thiết kế không nhất quán, với các bố cục bị hỏng, hình ảnh bị thiếu, hoặc văn bản khó đọc, khiến cho ứng dụng không thể sử dụng được về mặt trực quan.	Giao diện người dùng bao gồm hầu hết các thành phần cần thiết (trang chủ với các chế độ xem dựa trên vai trò, cấu trúc 3 tab cho khóa học, trang hồ sơ) nhưng có một số vấn đề hiển thị nhỏ (ví dụ: các yếu tố căn chỉnh sai lệch, tỷ lệ phóng/thu nhỏ không tối ưu trên máy tính bảng). Thiết kế đáp ứng được triển khai một phần, với một số thiết bị cho thấy hình ảnh không tối ưu. Thiết kế hoạt động nhưng thiếu sự trau chuốt hoặc nhất quán.	Giao diện người dùng trông bắt mắt, với đầy đủ các thành phần yêu cầu (trang chủ với các chế độ xem dựa trên vai trò, cấu trúc khóa học gồm 3 tab, trang hồ sơ) được triển khai đúng. Hoàn toàn đáp ứng trên di động, máy tính bảng và máy tính để bàn, với thiết kế nhất quán, bố cục rõ ràng và không có lỗi hiển thị. Các yếu tố trực quan tăng cường khả năng sử dụng và phù hợp với yêu cầu của dự án.
Trải nghiệm người dùng (UX) Điểm được chấm dựa trên nhận thức của giáo viên; chỉ những công việc ở mức trên trung bình mới được cộng điểm (một giao diện cơ bản sẽ không được cộng điểm).	0,5	Trải nghiệm người dùng (UX) kém, với điều hướng khó hiểu, gặp sự cố thường xuyên hoặc lỗi nghiêm trọng (ví dụ: hiển thị dữ liệu không chính xác, chuyển học kỳ bị hỏng). Các tính năng tìm kiếm/lọc/sắp xếp không hoạt động hoặc gây ra lỗi. Người dùng không thể hoàn thành các nhiệm vụ cơ bản mà không gặp phải sự thất vọng đáng kể, khiến hệ thống trở nên không khả thi.	UX hoạt động ở mức chức năng nhưng có những vấn đề đáng chú ý, chẳng hạn như tìm kiếm/lọc/sắp xếp chậm, việc chuyển đổi kỳ học rườm rà, hoặc các lỗi đồng bộ hóa bộ nhớ đệm nhỏ. Người dùng có thể hoàn thành các tác vụ, nhưng trải nghiệm không nhất quán hoặc bất tiện (ví dụ: phản hồi chậm hoặc thông báo lỗi không rõ). Khả năng sử dụng cơ bản đã đạt được nhưng chưa tối ưu.	UX mượt mà và trực quan, với tìm kiếm/lọc/sắp xếp nhanh và chuyển đổi giữa các học kỳ một cách liền mạch. Việc đồng bộ bộ nhớ đệm đáng tin cậy, ngăn ngừa sự không khớp dữ liệu. Tất cả các tương tác đều thân thiện với người dùng, có phản hồi rõ ràng và không gặp lỗi, đảm bảo trải nghiệm hiệu quả và dễ chịu cho cả giảng viên và sinh viên.

- Các thành phần cần nộp bao gồm:

  o Thư mục "source": Chứa toàn bộ mã nguồn của ứng dụng Flutter, máy chủ web và API nếu có, cùng với các tệp cơ sở dữ liệu liên quan. Đảm bảo mã nguồn này có thể được biên dịch và chạy trên máy tính của giáo viên. Dự án cần được "làm sạch" để loại bỏ nội dung không cần thiết trước khi nộp và để giảm kích thước của tệp nén.

  o Thư mục 'bin': Thư mục này nên chứa các tệp thực thi cho ít nhất hai nền tảng, bao gồm Android (APK) và Windows (EXE). Giáo viên sẽ sử dụng các tệp này để cài đặt trên các máy thật/ảo của họ để đánh giá. Nếu cần thiết, giáo viên có thể chọn xây dựng từ mã nguồn từ đầu.

  Video demo có tiêu đề “demo.mp4” phải bao gồm sự tham gia của tất cả các thành viên trong nhóm để giới thiệu sản phẩm của nhóm. Video phải cung cấp một cái nhìn tổng quan ngắn gọn về công nghệ và kiến trúc của ứng dụng web. Sau đó, đội ngũ phải lần lượt trình diễn từng tính năng được phát triển. Bất kỳ tiêu chí nào không được trình diễn trong video sẽ được coi là chưa được triển khai, ngay cả khi

group claims otherwise in their self-assessment form. The video must have a minimum resolution of 1080p, clear audio, and be easy to understand. If the video file is too large, the team should upload it to **YouTube** and include the link in the submission.

- The "**git**" folder must contain screenshots taken only from **GitHub Insights** that demonstrate collaboration between team members on the GitHub or GitLab repository, while any other forms of evidence (such as .txt files, terminal git log captures, or photos of command-line history) are strictly **not** acceptable. The submitted screenshots should provide clear proof of effective teamwork with balanced workload distribution, early and regular commits, and proactive collaboration among all members. In addition, the evidence must confirm that the project has lasted for at least one month, during which each member has consistently maintained a minimum of two commits per week.
- **Readme.txt** file: Provide all necessary information for the evaluation process, such as project building and running instructions, URL + server login information (if applicable), and usernames/passwords for accounts with pre-loaded data for assessment. Include any relevant notes on building, running, and using the application for teachers to reproduce the project. If your team implements some optional features (which get extra points), it should be clearly stated in the readme file.
- **The "Bonus"** folder**:** The bonus folder should include a description of the extra features your team implemented for additional points, along with evidence. Organize the information clearly, concisely, and convincingly.
- **Rubrik.docx:** This file lists the 36 required features for the project. Teams should self-assess their completion level in this file. The instructor will provide this file at the submission time. The file will also include the public URL to the web application and any required username/password for login.

- Organize all the above contents into a folder named **id1_fullname1_id2_fullname2**, then compress this folder in ZIP format with the same name, e.g., id1_fullname1_id2_fullname2.zip. A team representative should submit this file on the online learning system as instructed by the course instructor.
- Teachers do not accept sumissions via email, only elearning is accepted.

IV. IMPORTANT NOTES

- The Final Project must be implemented using a **Flutter** project using the **Dart** programming language. Groups are allowed to use any libraries within the framework of a Flutter project.
- Teams may use online storage services such as Firebase or equivalent services, or set up their own web server.
- If your group submits a project that is not relevant, it will not be graded, and the entire group will receive a score of 0. For example, if your group obtains the source code of another email client application for some reason, and only a few features are related to the description in this requirement, while the majority are completely unrelated, the project will be given 0 point.
- The Essay is entirely independent of the Final Project. Therefore, all team members must participate in both the Essay and the Final Project. The Essay will be assessed by the lab instructor, while the Final Project will be assessed by the theoretical instructor.

Nhóm tuyên bố ngược lại trong phiếu tự đánh giá của họ. Video phải có độ phân giải tối thiểu 1080p, âm thanh rõ ràng và dễ hiểu. Nếu tệp video quá lớn, nhóm nên tải lên YouTube và đính kèm liên kết trong bài nộp.

Thư mục 'git' phải chứa ảnh chụp màn hình được lấy từ GitHub Insights, chỉ thể hiện sự hợp tác giữa các thành viên trên kho lưu trữ GitHub hoặc GitLab, trong khi mọi hình thức chứng cứ khác (ví dụ như các tệp .txt, bản ghi log git ở terminal, hoặc ảnh lịch sử dòng lệnh) thì hoàn toàn không được chấp nhận. Các ảnh chụp màn hình được nộp phải cung cấp bằng chứng rõ ràng về làm việc nhóm hiệu quả với phân bổ khối lượng công việc cân bằng, các lần commit sớm và đều đặn, và sự hợp tác tích cực giữa tất cả các thành viên. Thêm vào đó, bằng chứng phải xác nhận rằng dự án đã kéo dài ít nhất một tháng, trong đó mỗi thành viên liên tục duy trì tối thiểu hai lần commit mỗi tuần.

tệp Readme.txt: Cung cấp mọi thông tin cần thiết cho quá trình đánh giá, như hướng dẫn xây dựng và chạy dự án, URL và thông tin đăng nhập máy chủ (nếu có), và tên người dùng/mật khẩu cho các tài khoản có dữ liệu được nạp sẵn để đánh giá. Bao gồm bất kỳ ghi chú liên quan nào về việc xây dựng, chạy và sử dụng ứng dụng để giáo viên có thể tái hiện dự án. Nếu đội của bạn triển khai một số tính năng tùy chọn (nhận thêm điểm), điều đó nên được nêu rõ trong tệp Readme.txt.

Thư mục “Bonus”: Thư mục này nên bao gồm mô tả các tính năng bổ sung mà đội ngũ của bạn đã triển khai để được thêm điểm, kèm theo bằng chứng. Hãy sắp xếp thông tin một cách rõ ràng, ngắn gọn và thuyết phục.

o Rubrik.docx: Tệp này liệt kê 36 tính năng bắt buộc cho dự án. Các đội nên tự đánh giá mức độ hoàn thành của mình trong tệp này. Giảng viên sẽ cung cấp tệp này tại thời điểm nộp bài. Tệp sẽ bao gồm cả URL công khai của ứng dụng web và bất kỳ tên đăng nhập và mật khẩu cần thiết để đăng nhập.

Hãy sắp xếp toàn bộ nội dung ở trên vào một thư mục có tên id1_fullname1_id2_fullname2, sau đó nén thư mục này ở định dạng ZIP với cùng tên, ví dụ: id1_fullname1_id2_fullname2.zip. Đại diện của nhóm nên nộp tập tin này lên hệ thống học tập trực tuyến theo đúng hướng dẫn của giảng viên khóa học.

• Giáo viên không nhận bài nộp qua email, chỉ có e-learning được chấp nhận.

G

• Dự án cuối kỳ phải được triển khai bằng một dự án Flutter sử dụng ngôn ngữ lập trình Dart. Các nhóm được phép sử dụng bất kỳ thư viện nào trong khuôn khổ của một dự án Flutter.

• Các đội có thể sử dụng các dịch vụ lưu trữ trực tuyến như Firebase hoặc các dịch vụ tương đương, hoặc tự thiết lập máy chủ web của riêng họ.

• Nếu nhóm của bạn nộp một dự án không liên quan, nó sẽ không được chấm điểm, và toàn bộ nhóm sẽ nhận được điểm 0. Ví dụ, nếu nhóm bạn có được mã nguồn của một ứng dụng khách email khác vì một lý do nào đó, và chỉ có một vài tính năng liên quan đến mô tả trong yêu cầu này, trong khi phần lớn hoàn toàn không liên quan, dự án sẽ được cho 0 điểm.

• Bài luận hoàn toàn độc lập với Dự án cuối kỳ. Do đó, tất cả các thành viên trong nhóm phải tham gia ở cả Bài luận và Dự án cuối kỳ. Bài luận sẽ được đánh giá bởi giảng viên phụ trách phòng thí nghiệm, trong khi Dự án cuối kỳ sẽ được đánh giá bởi giảng viên lý thuyết.

- Groups are prohibited from sharing code with each other, obtaining source code from the internet, and must take responsibility for protecting their group's source code. Groups with similar source code (verified by specialized software) or code found online, even if only in part, will receive a score of 0 for all members, regardless of which group shared or received the code.
- Failure to submit source code will result in the entire team receiving a score of 0.
- If the team does not fill out and submit the **rubric.docx** file, the submission will not be graded.
- Failure to deploy the web version to public hosting or submit the apk/exe file will result in the entire team receiving a score of 0.
- In case the submission is missing or there are no specific instructions on how to run the project, and the lecturer has tried his best but still cannot run the project on his computer, leading to no way to check the results, the whole group will also receive 0 points.
- Deductions will also apply in the following situations:
    - Late submission: 1-day late deducts 1 point. Submissions late by 1 second to less than 1 day are considered 1 day late.
    - Complex project configuration without specific instructions for instructors to compile and run the program: Deduction of 2 points.
    - Submit the entire project without performing a clean to remove unnecessary files: 0.5 point.
    - Failure to provide necessary grading information, such as missing usernames/passwords, incorrect file naming, or not submitting required content: 1.0 point.

• Các nhóm bị cấm chia sẻ mã nguồn với nhau; không được lấy mã nguồn từ Internet; và phải có trách nhiệm bảo vệ mã nguồn của nhóm mình. Các nhóm có mã nguồn tương tự (được xác nhận bằng phần mềm chuyên dụng) hoặc mã nguồn tìm thấy trên mạng, ngay cả khi chỉ là một phần, sẽ bị 0 điểm cho toàn bộ các thành viên, bất kể nhóm nào đã chia sẻ hay nhận mã nguồn.

• Việc không nộp mã nguồn sẽ khiến cả đội nhận được điểm 0.

• Nếu nhóm không điền đầy đủ và nộp tệp rubric.docx, bài nộp sẽ không được chấm điểm.

• Việc triển khai phiên bản web lên hosting công khai hoặc nộp tệp apk/exe sẽ khiến toàn đội nhận được 0 điểm.

• Trong trường hợp bài nộp bị thiếu hoặc không có hướng dẫn cụ thể nào về cách chạy dự án, và giảng viên đã cố gắng hết sức nhưng vẫn không thể chạy dự án trên máy tính của giảng viên, dẫn đến không có cách nào để kiểm tra kết quả, cả nhóm cũng sẽ nhận được 0 điểm.

• Các khoản khấu trừ cũng sẽ được áp dụng trong các tình huống sau:

- Nộp bài muộn: trễ 1 ngày sẽ bị trừ 1 điểm. Các bài nộp muộn từ 1 giây đến dưới 1 ngày được coi là trễ 1 ngày.
- Cấu hình dự án phức tạp mà không có hướng dẫn cụ thể cho giảng viên biên dịch và chạy chương trình: Trừ 2 điểm.
- Nộp toàn bộ dự án mà không thực hiện dọn dẹp để loại bỏ các tệp không cần thiết: 0,5 điểm.
- Không cung cấp thông tin chấm điểm cần thiết, chẳng hạn như thiếu tên đăng nhập hoặc mật khẩu, đặt tên tệp không đúng, hoặc không nộp nội dung bắt buộc: 1,0 điểm.