Bài 01

* Giao dịch bán hang (tps)
* Phân tích xu hướng kinh doanh (dss)
* Bảng tổng quan hiệu suất hàng tháng dành cho CEO (eis)

Bài 02

| Dự án | Mô tả ngắn |
| --- | --- |
| A | Phần mềm quản lý điểm cho trường cấp 2, yêu cầu rõ ràng |
| B | Ứng dụng mobile đặt lịch khám bệnh, yêu cầu linh hoạt |
| C | Hệ thống ngân hàng điện tử có bảo mật và độ phức tạp cao |

A Waterfall (Vì yêu cầu đã rõ ràng)

B Agile (Vì yêu cầu lich hoạt)

C Spiral (Vì yêu cầu lich hoạt bảo mật và độ phức tạp cao)

Bài 03

|  |  |
| --- | --- |
| **Thành phần** | **Mô tả** |
| Con người | Người đặt hàng, người bán hàng, người giao hàng, người quản trị viên |
| Dữ liệu | Thông tin người dùng, thông tin món, thông tin đơn, thông tin người giao hàng, đánh giá |
| Quy trình | Đăng nhập, đặt món, giao hàng, thanh toán, đánh giá |
| Phần mềm | Ứng dụng điện thoại, websibe |
| Phần cứng | Điện thoại, máy tính, server, thiết bị mạng |

Bài 04

|  |  |
| --- | --- |
| **Giai đoạn** | **Việc cần làm trong dự án "Ứng dụng điểm danh"** |
| Planning | Mục tiêu: Giúp giảng viên điểm danh nhanh và chính xác |
| Analysis | Thu thập yêu cầu từ giảng viên, sinh viên, phòng đào tạo |
| Design | Thiết kế giao diện người dùng , thiết kế cơ sở dữ liệu và kiến trúc hệ thống |
| Implementation | Tiến hành viết mã cho ứng dụng: giao diện sinh viên – giảng viên, chức năng điểm danh, kết nối cơ sở dữ liệu, API xử lý dữ liệu. |
| Testing | Kiểm tra từng chức năng: đăng nhập, điểm danh, lưu dữ liệu, hiển thị báo cáo. Thử nghiệm trên nhiều thiết bị khác nhau để đảm bảo hoạt động ổn định và không lỗi. |
| Deployment & Maintenance | Phát hành cho người dùng. Theo dõi, sửa lỗi phát sinh, cập nhật tính năng mới |

Bài 05

1. Planning

- Lấy yêu cầu của phí nhà trường nhà trường

- Nghiên cứu tính khả thi

2. Requirement Analysis

- Thu thập yêu cầu từ giảng viên, sinh viên, phòng đào tạo

- Phân tích yêu cầu

- Vẽ Use Case

- Xác địch yêu cầu chức năng và phi chức năng

3. System Design

- Thiết kế cơ sở dữ liệu

+ Cấu trúc dữ liệu

+ Mối quan hệ

- Thiết kế giao diện

+ Giao diện thuận tiện, đơn giản, dễ hiểu

- Thiết kế kiến trúc hệ thống

Bài 6

|  |  |
| --- | --- |
| **Tình huống** | **Sơ đồ UML phù hợp** |
| A. Mô tả chức năng người dùng có thể thực hiện trên ứng dụng học tiếng Anh | Use Case Diagram (Sơ đồ ca sử dụng) |
| B. Mô tả lớp NguoiDung, KhoaHoc, BaiHoc và quan hệ giữa chúng | Class Diagram (Sơ đồ lớp) |
| C. Mô tả luồng học viên bắt đầu → vào học → làm bài → hoàn thành | Activity Diagram (Sơ đồ hoạt động) |
| D. Mô tả cách hệ thống triển khai trên các máy chủ, thiết bị | Deployment Diagram (Sơ đồ triển khai) |
| E. Mô tả thứ tự tương tác giữa học viên và hệ thống khi nộp bài | Sequence Diagram (Sơ đồ trình tự) |

Bài 7

|  |  |
| --- | --- |
| **Giai đoạn** | **Việc cần làm trong dự án " hệ thống hỗ trợ đăng ký tiêm chủng online "** |
| Planning | - Làm việc với trung tâm y tế để hiểu nhu cầu  - Xác định mục tiêu: hỗ trợ người dân đăng ký tiêm online  - Lên kế hoạch thời gian, nhân sự, ngân sách và công nghệ  - Đánh giá rủi ro, quy định pháp lý, yêu cầu bảo mật |
| Analysis | - Phỏng vấn, khảo sát nhu cầu của người dân, nhân viên y tế  - Xác định chức năng: đăng ký tiêm, xác nhận lịch, quản lý người tiêm/chưa tiêm  - Xác định yêu cầu phi chức năng: tốc độ, bảo mật, tính ổn định  - Vẽ sơ đồ Use Case và mô tả dữ liệu cần thu thập |
| Design | - Thiết kế kiến trúc client–server, cơ sở dữ liệu trung tâm  - Xây dựng sơ đồ lớp (Class Diagram), trình tự (Sequence Diagram)  - Thiết kế giao diện người dùng: form đăng ký, trang quản lý, bảng thống kê  - Xác định quy trình xử lý và API kết nối giữa các thành phần |
| Implementation | - Phát triển website/ứng dụng đăng ký tiêm  - Xây dựng các module: đăng ký, xác nhận, quản lý lịch, thống kê  - Tích hợp xác thực người dùng và gửi thông báo (SMS, Email)  - Viết mã backend và frontend, kết nối cơ sở dữ liệu |
| Testing | - Thực hiện kiểm thử chức năng (đăng ký, xác nhận, thống kê)  - Kiểm thử hiệu năng khi nhiều người truy cập cùng lúc  - Kiểm thử bảo mật dữ liệu người dùng  - Sửa lỗi, tối ưu tốc độ xử lý |
| Deployment & Maintenance | - Triển khai hệ thống lên server thật - Hướng dẫn nhân viên y tế sử dụng  - Theo dõi hoạt động, sửa lỗi phát sinh, nâng cấp định kỳ  - Thu thập phản hồi người dùng để cải tiến |

Bài 8

1 Các tác nhân chính và chức năng tương ứng

* Học viên (Student)
  + Đăng ký tài khoản / đăng ký khóa học
  + Tham gia bài học (video/zoom), nộp bài tập, làm bài kiểm tra, xem điểm, theo dõi tiến độ
  + Trao đổi với giảng viên (chat/forum), đánh giá khóa học
* Giảng viên (Instructor)
  + Tạo/ quản lý khóa học, bài học, tài liệu
  + Chấm điểm, tạo bài kiểm tra/quiz, lên lịch buổi live
  + Trả lời câu hỏi, feedback, quản lý lớp (danh sách học viên)
* Admin (Quản trị hệ thống / Trung tâm)
  + Quản lý người dùng (phân quyền), quản lý khóa học, phê duyệt nội dung
  + Xem báo cáo doanh thu, báo cáo tiến độ học tập, thống kê KPI
  + Cấu hình hệ thống, xử lý sự cố, sao lưu dữ liệu
* Hệ thống phụ trợ (Payment gateway / Notification service / LMS engine)
  + Xử lý thanh toán, gửi email/SMS thông báo, stream video/recording

2 Phân loại hệ thống thông tin phù hợp

* TPS (Transaction Processing System) — xử lý giao dịch cơ bản: đăng ký khóa, thanh toán, nộp bài, ghi điểm. (Cốt lõi, cần tin cậy và nhanh)
* MIS (Management Information System) — tổng hợp báo cáo cho Admin: doanh thu, số học viên theo khóa, tỉ lệ hoàn thành.
* DSS (Decision Support System) — hỗ trợ quyết định quản lý: gợi ý mở khóa mới dựa trên xu hướng, phân bổ giảng viên theo tải. (tùy có thêm phân tích dữ liệu)
* LMS (Learning Management System) — dạng hệ thống chuyên biệt (tập hợp TPS + nội dung học + tương tác + reporting).

Tổng kết: nền tảng này kết hợp TPS + MIS, có thể mở rộng thêm DSS nếu triển khai phân tích dữ liệu/AI.

3 Đề xuất mô hình phát triển phần mềm (và lý do)

Agile — Scrum (kết hợp CI/CD & DevOps)

* Lý do:
  + Yêu cầu thay đổi nhanh (nội dung khóa, feedback giảng viên) => Agile hỗ trợ iteratives và phản hồi sớm.
  + Có thể ra mắt MVP sớm (đăng ký + học + chấm điểm cơ bản) rồi bổ sung tính năng (thông báo, báo cáo nâng cao).
  + Scrum giúp phân chia công việc theo sprint, dễ quản lý priorites (ví dụ: bảo mật, thanh toán là ưu tiên cao).
  + Kết hợp CI/CD & DevOps đảm bảo deploy nhanh, rollback an toàn, tự động test.
* Khi nào dùng khác: Nếu dự án rất nhỏ/đơn giản có thể dùng Waterfall; nhưng với hệ thống học trực tuyến thường cần Agile.

4 Ba sơ đồ UML nên dùng khi đi sâu thiết kế (và vì sao)

1. Use Case Diagram — mô tả chức năng hệ thống theo tác nhân (phù hợp cho yêu cầu nghiệp vụ, dễ trình bày cho stakeholder).
2. Class Diagram — mô tả mô hình dữ liệu: User, Course, Lesson, Enrollment, Submission, Grade… (cốt lõi cho thiết kế CSDL & backend).
3. Sequence Diagram (hoặc Activity Diagram) — mô tả luồng tương tác cho các quy trình quan trọng như đăng ký khóa 🡪 thanh toán 🡪 kích hoạt truy cập hoặc nộp bài 🡪 chấm điểm 🡪 thông báo. Sequence minh họa thứ tự message giữa client, server, payment gateway.

Bài 9

**1) Các tác nhân (Actors) chính & chức năng của họ**

* **Khách hàng (Customer)**
  + Tạo đơn hàng (địa chỉ, hàng hóa, thời gian yêu cầu)
  + Thanh toán / chọn phương thức thanh toán
  + Theo dõi trạng thái giao hàng (tracking)
  + Yêu cầu hỗ trợ / khiếu nại / đánh giá dịch vụ
* **Nhân viên vận chuyển (Driver / Courier)**
  + Nhận đơn được phân công
  + Cập nhật trạng thái giao: “đang lấy hàng”, “đang giao”, “giao thành công”, “giao thất bại”
  + Ghi chú lỗi/ghi nhận hình ảnh (proof-of-delivery)
* **Điều phối/Dispatcher (Operations staff)**
  + Phân luồng, gán đơn cho tài xế theo khu vực/khả năng
  + Giám sát trạng thái thời gian thực, tái phân công khi cần
* **Nhân viên kho (Warehouse staff)**
  + Xác nhận nhập/xuất hàng, đóng gói, cập nhật tồn kho
* **Quản lý (Manager / Regional Manager)**
  + Xem báo cáo hiệu suất vận chuyển (time-to-deliver, succes-rate)
  + Phân tích khu vực tồn đọng, đưa ra quyết định vận hành
* **CEO / Ban giám đốc (Executive)**
  + Xem dashboard tổng quan: số giao dịch, vùng hoạt động, tỷ lệ giao đúng hẹn, KPI cao cấp
* **Hệ thống phụ trợ / bên thứ ba** (Payment gateway, Map/Route service, SMS/Email provider) — không phải “người” nhưng là tác nhân hệ thống cần tích hợp.

**2) Ánh xạ chức năng → loại hệ thống thông tin (TPS, MIS, DSS, EIS)**

|  |  |  |
| --- | --- | --- |
|  |  |  |
| **Chức năng / Use case** | **Loại hệ thống** | **Lý do** |
| Tạo đơn hàng, thanh toán, cập nhật trạng thái giao (PO, delivery updates) | **TPS (Transaction Processing System)** | Ghi nhận giao dịch thời gian thực; cần tính chính xác, tính sẵn sàng, xử lý hàng loạt. |
| Theo dõi tồn kho, xác nhận nhập/xuất | **TPS** | Giao dịch kho và cập nhật trạng thái tồn kho. |
| Báo cáo hiệu suất hàng ngày/tuần cho quản lý (số đơn, tỉ lệ giao thành công, thời gian trung bình) | **MIS (Management Information System)** | Tổng hợp dữ liệu TPS thành báo cáo định kỳ, phục vụ vận hành. |
| Phân tích khu vực tồn đọng, đề xuất lịch trình / tuyến tối ưu, mô phỏng tải | **DSS (Decision Support System)** | Cung cấp phân tích, mô phỏng và gợi ý cho các quyết định vận hành (route optimization, resource allocation). |
| Dashboard CEO — KPI tổng quan, heatmap vùng hoạt động, alert chiến lược | **EIS (Executive Information System)** | Trình bày thông tin chiến lược, tóm tắt cao cấp, hỗ trợ quyết định điều hành. |
| Cảnh báo thời gian thực (đơn trễ, sự cố lớn) | **Kết hợp TPS + MIS/EIS** | Cần dữ liệu giao dịch thời thực nhưng được đẩy vào hệ thống thông báo/điều hành. |

**Ghi chú:** hệ thống thực tế sẽ là **hệ sinh thái**: TPS (cốt lõi) → dữ liệu được tổng hợp vào MIS → DSS dùng dữ liệu để hỗ trợ quyết định → EIS cung cấp cái nhìn điều hành cho CEO.

**3) Mô hình phát triển phần mềm đề xuất (và lý do)**

**Agile (Scrum) kết hợp DevOps + kiến trúc Microservices (Event-driven)**

**Lý do chính:**

* **Thay đổi yêu cầu cao:** logistic thường phải tích hợp nhiều dịch vụ bên ngoài (nhà vận chuyển thứ ba, cổng thanh toán, dịch vụ bản đồ). Agile cho phép lặp nhanh, ưu tiên tích hợp quan trọng trước (MVP).
* **Phần mềm phức tạp, module tách rời:** Microservices giúp tách TPS, routing engine, billing, tracking, reporting thành các dịch vụ độc lập — dễ mở rộng và deploy riêng.
* **Yêu cầu vận hành 24/7 & triển khai liên tục:** DevOps + CI/CD cho phép deploy nhanh, rollback, và giám sát (monitoring/alerting).
* **Event-driven phù hợp cho logistic:** trạng thái đơn thay đổi (events) được stream tới các consumers: dashboard, MIS, notification service — giảm coupling, tăng khả năng mở rộng.
* **Khả năng mở rộng theo khu vực:** microservices + autoscaling giúp xử lý peak load (ví dụ: sale, mùa cao điểm).

**Kết luận:** mô hình Agile+DevOps + microservices là phù hợp nhất cho hệ thống logistics cần tính mở rộng, tích hợp nhiều service, và đổi mới nhanh.

**4) Bốn sơ đồ UML đề xuất & mục đích sử dụng**

1. **Use Case Diagram**

**Mục đích:** Giới thiệu nhanh các actors và chức năng chính (tạo đơn, theo dõi, cập nhật trạng thái, quản lý báo cáo).  
**Khi dùng:** Giai đoạn Requirement Analysis để xác định scope và trao đổi với stakeholders.

1. **Class Diagram (Sơ đồ lớp / Data Model)**

**Mục đích:** Mô tả các thực thể chính (Order, Customer, Shipment, Driver, Warehouse, Inventory, Route, ExceptionReport) và quan hệ giữa chúng.  
**Khi dùng:** Thiết kế CSDL và định nghĩa model cho backend (ORM, API contract).

1. **Sequence Diagram (Ví dụ: quy trình “Tạo đơn → Phân tuyến → Cập nhật trạng thái → Thông báo”)**

**Mục đích:** Minh họa thứ tự thông điệp giữa client, order service, dispatch service, driver mobile, notification service, tracking DB.  
**Khi dùng:** Thiết kế luồng nghiệp vụ quan trọng, làm rõ tương tác và timing giữa các service, hữu ích cho lập trình viên backend/frontend.

1. **Deployment Diagram (hoặc Component/Activity Diagram — mình khuyên Deployment)**

**Mục đích:** Mô tả cách triển khai hệ thống trên hạ tầng: load balancer, API gateway, service nodes, database cluster, message broker (Kafka/RabbitMQ), CDN, external providers.  
**Khi dùng:** Thiết kế hạ tầng, thảo luận với đội DevOps/IT về scaling, HA, backup, network.

**Gợi ý thêm:** Khi cần tối ưu nghiệp vụ như “tối ưu tuyến” thì **Activity Diagram** hoặc **State Machine Diagram** cho Shipment cũng rất hữu ích để mô tả các trạng thái và chuyển đổi (Pending → InPickup → InTransit → Delivered → Exception).