Bài 1 :

* Giao dịch bán hàng : TPS
* Phân tích xu hướng kinh doanh : DSS
* Bảng cho CEO : EIS

Bài 2 :

* Dự án A : waterfall
* Dự án B : Agile
* Dự án C : Spiral

Bài 3 :

|  |  |
| --- | --- |
| Thành phần | Mô tả |
| Con người | Khách hàng  Restaurant  Shipper  Admin |
| Dữ liệu | User, restaurant , shipper, menu ,food,payment |
| Quy trình | -Đặt món : web || mobile  -Moblie : ứng dụng -> đăng nhập -> chọn món -> cửa hàng -> đặt món -> thanh toán -> đợi hoàn thành đơn -> xác nhận |
| Phần mềm | App , web |
| Phần cứng | Điện thoại , máy tính, server,  Client |

Bài 4 :

| Giai đoạn | Việc cần làm trong dự án “Ứng dụng điểm danh” |
| --- | --- |
| 1. Planning (Lập kế hoạch) | - Xác định mục tiêu dự án: tạo ứng dụng giúp giảng viên điểm danh nhanh, chính xác. - Xác định phạm vi, chi phí, thời gian, nhân sự và rủi ro. - Lập kế hoạch tổng thể: công nghệ dùng (mobile/web), thời hạn từng giai đoạn. |
| 2. Analysis (Phân tích yêu cầu) | - Thu thập và phân tích yêu cầu người dùng (giảng viên, sinh viên, phòng đào tạo). - Xác định các chức năng chính: đăng nhập, quản lý lớp, điểm danh bằng mã QR, thống kê điểm danh. - Mô tả luồng dữ liệu và các đối tượng liên quan. |
| 3. Design (Thiết kế hệ thống) | - Thiết kế giao diện người dùng (UI/UX): màn hình đăng nhập, danh sách lớp, bảng điểm danh. - Thiết kế kiến trúc hệ thống, cơ sở dữ liệu (ERD), các API backend. - Đảm bảo thiết kế dễ mở rộng và bảo mật dữ liệu sinh viên. |
| 4. Implementation (Triển khai/lập trình) | - Tiến hành lập trình frontend (ứng dụng mobile/web) và backend (API, database). - Tạo các chức năng chính: đăng nhập, quét mã QR, cập nhật trạng thái điểm danh, báo cáo thống kê. - Tích hợp với hệ thống trường học (nếu có). |
| 5. Testing (Kiểm thử) | - Thực hiện kiểm thử chức năng, hiệu năng, bảo mật. - Kiểm tra độ chính xác khi quét QR, tốc độ xử lý, khả năng đăng nhập nhiều người dùng. - Sửa lỗi (bug) và tối ưu hiệu suất ứng dụng. |
| 6. Deployment & Maintenance (Triển khai và bảo trì) | - Triển khai ứng dụng cho giảng viên và sinh viên sử dụng (đưa lên server hoặc store). - Theo dõi hoạt động thực tế, tiếp nhận phản hồi. - Cập nhật, sửa lỗi, nâng cấp tính năng (ví dụ: thêm chức năng thống kê theo tháng, thông báo vắng học). |

Bài 5 :

|  |  |
| --- | --- |
| Giai đoạn | Việc cần làm |
| Planning | - Xác định mục tiêu: Tự động hóa việc điểm danh sinh viên bằng QR code, giảm gian lận, tiết kiệm thời gian. - Xác định phạm vi dự án: Ứng dụng cho giảng viên và sinh viên, có hệ thống quản trị cho phòng đào tạo. |
| Requirement analysis | |  | | --- | | - Phỏng vấn và khảo sát giảng viên, sinh viên, phòng đào tạo để hiểu rõ nhu cầu. - Xác định chức năng chính:  • Giảng viên: tạo buổi học, tạo QR code, xem danh sách sinh viên đã điểm danh.  • Sinh viên: đăng nhập, quét mã QR, xem lịch sử điểm danh.  • Phòng đào tạo: xem thống kê, báo cáo tỷ lệ chuyên cần theo lớp/môn. - Mô tả use case, biểu đồ luồng dữ liệu (DFD), và Use Case Diagram UML. - Đầu ra: tài liệu *SRS (Software Requirement Specification)*. | |
| System design | - Thiết kế kiến trúc tổng thể: gồm 3 phần – Ứng dụng mobile (client), Server API, và CSDL. - Thiết kế cơ sở dữ liệu: bảng *Users*, *Classes*, *Sessions*, *Attendances*. - Thiết kế giao diện người dùng (UI/UX): màn hình đăng nhập, danh sách lớp, quét QR, thống kê. - Thiết kế UML chi tiết:  • *Use Case Diagram* (phân tích chức năng).  • *Class Diagram* (thiết kế các lớp chính: GiảngVien, SinhVien, BuoiHoc, DiemDanh).  • *Sequence Diagram* (mô tả luồng điểm danh: giảng viên → sinh viên → server). |

Bài 6 :

|  |  |
| --- | --- |
| Tình huống | Sơ đồ tương ứng |
| A | Use case diagram |
| B | Class diagram |
| C | Acticity diagram |
| D | Deployment diagram |
| E | Sequence diagram |

Bài 7:

|  |  |  |
| --- | --- | --- |
| Giai đoạn | Mục tiêu chính | Kết quả chính |
| planning | Xác định phạm vi, ngân sách, stakeholders, rủi ro | Project charter, scope statement, roadmap, team RACI |
| Analysis | Thu thập yêu cầu, phân tích nghiệp vụ | SRS (Requirements), use cases, data model, user journeys |
| Design | |  | | --- | |  |  |  | | --- | | Thiết kế kiến trúc và UI/UX, DB schema | | High-level architecture, ERD, wireframes, API spec |
| Implementation | Lập trình các module, tích hợp | Source code, CI config, Dockerfile, migrations |
| testing | Kiểm tra chức năng, bảo mật, hiệu năng | Test plan, test cases, test reports, bug backlog |
| Deployment & Maintenance | Triển khai, vận hành, bảo trì | Release notes, runbook, monitoring dashboards |

Bài 8 :

1. Xác định các tác nhân chính và chức năng tương ứng

| Tác nhân (Actor) | Vai trò / mô tả | Chức năng tương ứng (Use cases) |
| --- | --- | --- |
| Học viên (Student) | Người đăng ký và học các khóa học | - Đăng ký tài khoản, đăng nhập - Xem danh sách khóa học - Đăng ký khóa học - Truy cập bài học (video, tài liệu, quiz) - Làm bài kiểm tra, xem điểm - Gửi phản hồi cho giảng viên |
| Giảng viên (Instructor) | Người tạo và quản lý nội dung học tập | - Đăng nhập hệ thống - Tạo / chỉnh sửa khóa học, bài giảng, tài liệu - Quản lý danh sách học viên trong khóa - Chấm điểm, nhập điểm, nhận phản hồi - Thống kê tiến độ học viên |
| Quản trị viên (Admin) | Người giám sát toàn hệ thống | - Quản lý người dùng (thêm/sửa/xóa, phân quyền) - Quản lý khóa học và giảng viên - Theo dõi báo cáo: doanh thu, số lượng học viên, khóa học, điểm số trung bình - Sao lưu dữ liệu, bảo mật, cấu hình hệ thống |
| Hệ thống thanh toán (Payment Gateway) *(tác nhân ngoài)* | Cổng thanh toán điện tử | - Xử lý giao dịch khi học viên thanh toán khóa học - Gửi phản hồi xác nhận thanh toán thành công hoặc thất bại |
| Hệ thống thông báo (Notification Service) *(tác nhân ngoài)* | Gửi email / SMS nhắc lịch học, thông báo điểm | - Gửi thông báo khi có khóa học mới, điểm số, hoặc nhắc hạn nộp bài |

2. Phân loại hệ thống thông tin phù hợp

| Loại hệ thống thông tin | Vai trò trong nền tảng học trực tuyến |
| --- | --- |
| TPS (Transaction Processing System) | Xử lý các giao dịch hàng ngày như đăng ký tài khoản, đăng ký khóa học, thanh toán, chấm điểm. |
| MIS (Management Information System) | Hỗ trợ Admin và giảng viên theo dõi tình hình học tập, thống kê số lượng học viên, khóa học, doanh thu, kết quả học tập. |
| DSS (Decision Support System) | Hỗ trợ ban quản lý ra quyết định |

3. Đề xuất mô hình phát triển phần mềm phù hợp nhất

Mô hình đề xuất: Agile

Lý do chọn:

1. Yêu cầu thay đổi liên tục: Khóa học, phương pháp học, giao diện hay tính năng (quiz, video call, chứng chỉ…) có thể phải cập nhật thường xuyên. Agile linh hoạt hơn Waterfall.
2. Phù hợp phát triển theo từng giai đoạn (Sprint):
   * Sprint 1: Đăng ký học viên, đăng nhập
   * Sprint 2: Quản lý khóa học
   * Sprint 3: Tích hợp thanh toán
   * Sprint 4: Báo cáo & thống kê
3. Có sự tham gia liên tục của người dùng thật: Giảng viên và học viên có thể phản hồi sớm → cải thiện nhanh.
4. Giảm rủi ro: Mỗi sprint đều có bản chạy được, dễ kiểm soát lỗi và hiệu năng

4. Ba sơ đồ UML sẽ sử dụng trong thiết kế hệ thống

| Tên sơ đồ UML | Mục đích sử dụng | Ví dụ ứng dụng trong dự án |
| --- | --- | --- |
| Use Case Diagram | Mô tả các tác nhân và chức năng tương tác với hệ thống | Biểu diễn học viên, giảng viên, admin cùng các chức năng chính: đăng ký, quản lý, báo cáo... |
| Class Diagram | Thể hiện cấu trúc hệ thống ở mức đối tượng (lớp, thuộc tính, quan hệ) | Các lớp: User, Course, Lesson, Enrollment, Grade, Payment... |
| Sequence Diagram | Mô tả luồng tương tác theo thời gian giữa các đối tượng | Luồng “Học viên đăng ký khóa học” → hệ thống → thanh toán → xác nhận → thông báo |

Bài 9:

1) Các tác nhân chính (Actors) và chức năng tương ứng

Khách hàng (Customer)

* Tạo đơn hàng (từ/to; kích thước, trọng lượng, giá trị, option bảo hiểm)
* Thanh toán / chọn phương thức thanh toán
* Theo dõi trạng thái đơn hàng (real-time tracking)
* Hủy hoặc chỉnh sửa đơn (trong phạm vi cho phép)
* Gửi khiếu nại / phản hồi / đánh giá dịch vụ

Nhân viên vận chuyển (Driver / Courier)

* Nhận danh sách đơn được giao trong ca
* Cập nhật trạng thái giao (đang giao, đã lấy hàng, giao thành công, giao không thành/hoãn)
* Chụp ảnh xác nhận giao hàng (POD), thu/ghi biên lai COD
* Báo cáo sự cố (đường cấm, khách vắng, thất lạc)

Nhân viên điều phối (Dispatcher / Operations)

* Tạo/ phân tuyến đơn, gán chuyến cho tài xế
* Tối ưu lộ trình (gộp đơn, sắp xếp tour)
* Giám sát trạng thái đội xe theo thời gian thực
* Quản lý tồn kho tạm/ kho trung chuyển (nếu có)

Quản lý / Supervisor

* Giám sát hiệu suất giao hàng theo khu vực, nhân viên
* Xem và phê duyệt khiếu nại, xử lý hoàn tiền
* Quản lý ca, tài xế, phương tiện

CEO / Ban lãnh đạo (Executive / EIS user)

* Xem dashboard tổng quan: số đơn, doanh thu, tỉ lệ giao đúng hẹn, khu vực nghẽn
* So sánh KPI theo thời gian / vùng / kênh
* Nhận cảnh báo vùng tồn đọng hoặc rủi ro lớn

Hệ thống bên ngoài (External systems)

* Cổng thanh toán (Payment Gateway)
* Bản đồ / Routing API (Google, Mapbox, OpenRoute)
* Hệ thống SMS/Email (Notification)
* Hệ thống ERP/CRM (tùy tích hợp)

2) Phân loại từng chức năng vào loại hệ thống thông tin

Giải thích ngắn: TPS = Transaction Processing System; MIS = Management Information System; DSS = Decision Support System; EIS = Executive Information System.

TPS (xử lý giao dịch trực tiếp)

* Tạo đơn hàng, cập nhật trạng thái (driver updates), thanh toán, ghi nhận POD/COD.
* Tính năng: logging giao dịch, xử lý thanh toán, tạo biên nhận.

MIS (thông tin quản lý)

* Báo cáo hàng ngày/tuần về số đơn, tỉ lệ thành công, doanh thu theo vùng.
* Bảng điều khiển cho quản lý/operations: danh sách tồn, đơn trễ, hiệu suất tài xế.

DSS (hỗ trợ ra quyết định)

* Công cụ phân tích tuyến tối ưu, mô phỏng tác động khi thêm nhân lực/xe.
* Phân tích nguyên nhân tồn đọng, gợi ý phân bổ nguồn lực.
* Forecasting demand (dự báo khối lượng theo ngày/ vùng) và tối ưu lịch ca.

EIS (hệ thống thông tin điều hành / điều hành cấp cao)

* Dashboard cho CEO: KPI tổng quan, heatmap vùng hoạt động, tỉ lệ giao đúng hẹn, trend so sánh quý/ năm.
* Cảnh báo chiến lược (ví dụ: vùng có tỉ lệ hoàn trả cao > threshold).

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

Agile (Scrum) kết hợp DevOps / Continuous Delivery

Lý do:

1. Tính liên tục thay đổi yêu cầu: Logistic cần tích hợp API bên thứ ba (maps, thanh toán), thay đổi pháp lý, điều chỉnh quy trình vận hành — Agile cho phép thay đổi và ưu tiên backlog liên tục.
2. Phát hành nhanh & lặp: Cần ra mắt sớm các chức năng cốt lõi (tạo đơn, tracking, cập nhật trạng thái) rồi mở rộng (optimization, dashboards). Scrum với sprint 1–3 tuần phù hợp.
3. Tính vận hành và độ tin cậy cao: DevOps (CI/CD, Infrastructure as Code, observability) giúp deploy an toàn, rollback nhanh, monitoring — rất cần cho hệ thống vận hành 24/7.
4. Tính tích hợp & test tự động: Tự động hóa test (unit/integration/E2E), pipeline deploy staging → canary/blue-green để giảm downtime.
5. Iterative for DSS/EIS: Dashboard phân tích phức tạp có thể làm dần theo mô-đun, validate với stakeholders.

Kiến trúc kỹ thuật gợi ý:

* Microservices (module: orders, tracking, routing, billing, notifications, analytics) hoặc modular monolith nếu team nhỏ.
* Event-driven (message queue: Kafka/RabbitMQ) để xử lý trạng thái, analytics, và đảm bảo loose coupling (ví dụ driver update → event → update TPS + feed vào analytics).
* API gateway + mobile/web clients; database chuyên biệt cho OLTP (orders) và data warehouse cho OLAP/BI (analytics).

4) Đề xuất 4 sơ đồ UML dùng trong phân tích & thiết kế (và mục đích ngắn)

1. Use Case Diagram

* Mục đích: Mô tả các tác nhân chính và các chức năng (use cases) hệ thống hỗ trợ.
* Ứng dụng: Hiển thị Customer, Driver, Dispatcher, Manager, CEO và các use case: CreateOrder, TrackOrder, UpdateStatus, AssignRoute, GenerateReport, ViewExecutiveDashboard.

2. Class Diagram

* Mục đích: Mô tả các lớp/ thực thể dữ liệu và quan hệ giữa chúng (attributes, methods, associations).
* Ứng dụng: Lớp ví dụ: Order, Customer, Address, Shipment, Driver, Vehicle, Route, Payment, POD — thể hiện thuộc tính, khóa chính/ngoại, quan hệ 1-n, n-n.

3. Sequence Diagram

* Mục đích: Mô tả luồng tương tác theo thời gian giữa các đối tượng/hệ thống cho một scenario cụ thể.
* Ứng dụng: Ví dụ luồng “Khách tạo đơn → Hệ thống → Tạo tour → Gán driver → Driver cập nhật giao thành công → Hệ thống gửi confirmation + cập nhật analytics”.

4. Activity Diagram (hoặc Process Flow)

* Mục đích: Mô tả quy trình nghiệp vụ/luồng công việc phức tạp có nhiều nhánh quyết định (decision) và trạng thái.
* Ứng dụng: Quy trình xử lý đơn: CreateOrder → ValidatePayment? → AssignDispatch → Enroute → DeliveryAttempt? → Delivered / Return to Hub. Rất hữu ích để thiết kế logic xử lý lỗi và SLA.

5) Gợi ý deliverables & lộ trình ngắn (để đưa vào repo / nộp bài)

* README.md mô tả mục tiêu, stakeholders, kiến trúc tổng quát.
* docs/ chứa: UseCase.md, ClassDiagram.png, SequenceDiagram.png, ActivityDiagram.png, SRS.md (tối thiểu).
* arch/ chứa sơ đồ kiến trúc high-level (microservices/event-driven) và đề xuất stack (DB, message broker, map API, payment).
* Sprint đề xuất:
  + Sprint 1 (MVP): Tạo đơn, quản lý đơn cơ bản, driver cập nhật trạng thái, tracking đơn.
  + Sprint 2: Thanh toán + POD + notifications.
  + Sprint 3: Dispatch tools + route assignment.
  + Sprint 4: MIS dashboards + báo cáo cơ bản.
  + Sprint 5: DSS features (route optimization, forecasting) + EIS dashboards.