HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG

KHOA CÔNG NGHỆ THÔNG TIN 1

HỌC PHẦN: KIẾN TRÚC VÀ THIẾT KẾ PHẦN MỀM

ASSIGNMENT 1

Book Store Web System

Giảng viên:	**Trần Đình Quế**
Sinh viên:	**Nguyễn Tùng Dương**
Mã sinh viên:	**B22DCVT114**
Lớp:	**E22CNPM01**

Hà Nội 2025

Giới thiệu đề tài

1. Tổng quan:

- Dự án xây dựng hệ thống Book Store Web System nhằm mục đích tìm hiểu và so sánh ba kiến trúc phần mềm phổ biến: Monolithic, Clean Architecture và Microservices. Hệ thống được xây dựng bằng ngôn ngữ Python (Framework Django) và cơ sở dữ liệu MySQL.

2. Các thực thể chính:

- Hệ thống quản lý các đối tượng dữ liệu sau:
    - Customer: Quản lý thông tin người dùng (id, name, email, password).
    - Book: Quản lý danh mục sách (id, title, author, price, stock).
    - Cart: Quản lý giỏ hàng của khách (id, customer_id, created_at).
    - CartItem: Chi tiết sản phẩm trong giỏ (id, cart_id, book_id, quantity).

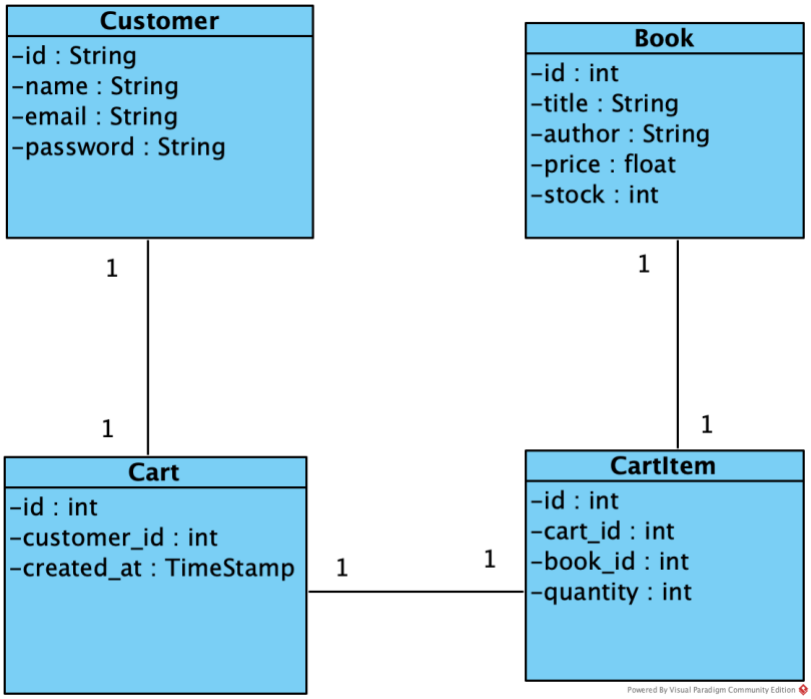
3. Yêu cầu chức năng:

- Hệ thống được xây dựng với các chức năng chính gồm:
    - Đăng ký và Đăng nhập khách hàng.
    - Xem danh sách sách (Catalog).
    - Thêm sách vào giỏ hàng (Kèm kiểm tra tồn kho).
    - Xem chi tiết giỏ hàng.

Phân tích & Thiết kế hệ thống

1. Class Diagram

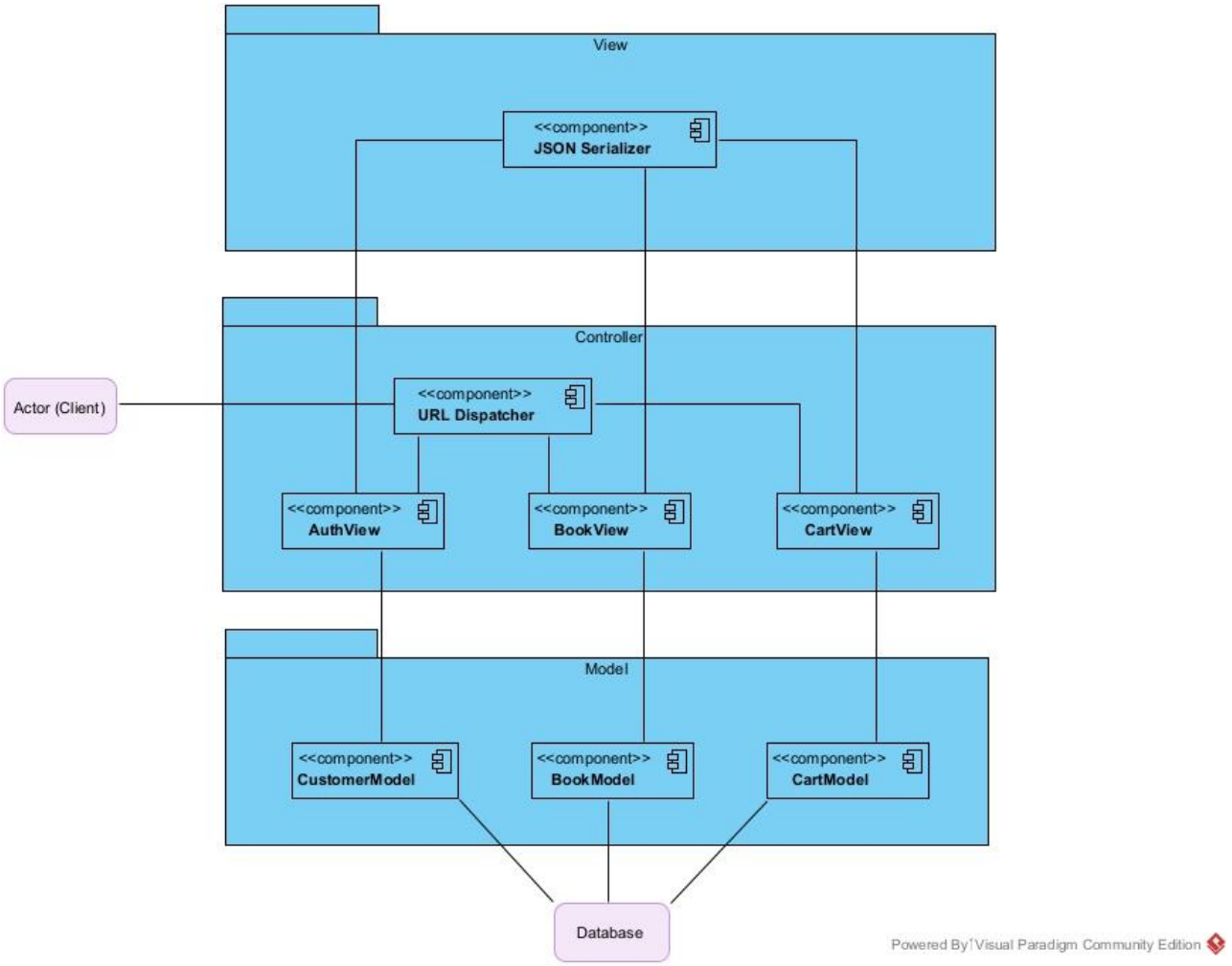
Biểu đồ lớp (Class Diagram) mô hình hóa cấu trúc dữ liệu tĩnh của hệ thống Book Store, bao gồm các thực thể chính và mối quan hệ giữa chúng:



- Customer: Lưu trữ thông tin khách hàng với các thuộc tính định danh (id), thông tin cá nhân (name, email) và bảo mật (password).
- Book: Đại diện cho sản phẩm sách trong kho, bao gồm thông tin chi tiết (title, author), giá bán (price) và số lượng tồn kho (stock).
- Cart: Đại diện cho giỏ hàng của một khách hàng cụ thể. Quan hệ giữa Customer và Cart là 1-1 (Mỗi khách hàng sở hữu một giỏ hàng).
- CartItem: Chi tiết các mục trong giỏ hàng.
    - Quan hệ giữa Cart và CartItem là 1-n (Một giỏ hàng chứa nhiều mục sách).
    - Quan hệ giữa Book và CartItem là quan hệ tham chiếu để xác định cuốn sách nào đang được chọn mua và số lượng là bao nhiêu.

2. MVC Layer Diagram (Monolithic):

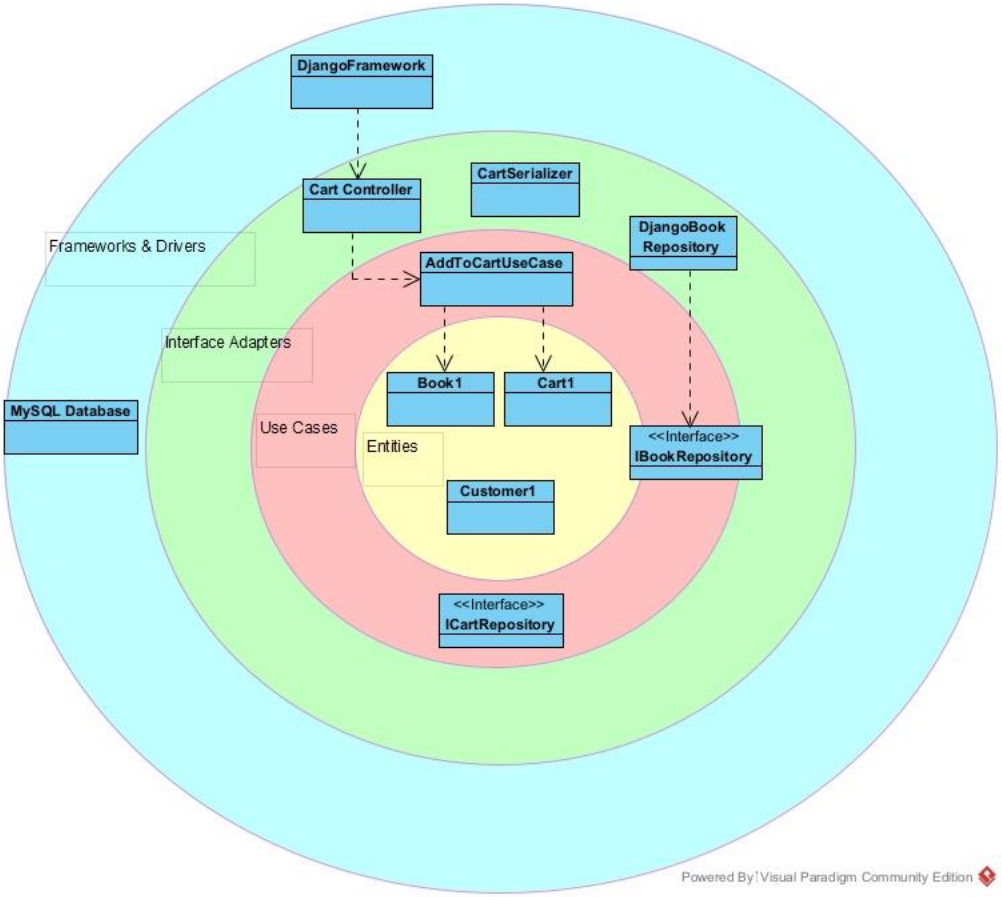
Trong phiên bản Monolithic, hệ thống được thiết kế theo mô hình MVC (Model - View - Controller) truyền thống của Django, tập trung toàn bộ mã nguồn trong một Project duy nhất.



- View Layer (Presentation):
  - Bao gồm các thành phần Serializer (chuyển đổi dữ liệu sang JSON) để phản hồi lại Client.
  - Client gửi request thông qua Actor.
- Controller Layer (Logic):
  - URL Dispatcher: Đóng vai trò điều hướng request đến đúng View xử lý.
  - Các Views (AuthView, BookView, CartView): Chứa logic xử lý nghiệp vụ, tiếp nhận input từ người dùng và gọi xuống tầng Model.
- Model Layer (Data Access):
  - Các Models (CustomerModel, BookModel, CartModel): Đại diện cho cấu trúc dữ liệu, tương tác trực tiếp với Database chung của hệ thống.
- Luồng dữ liệu: Request đi từ Client -> URL Dispatcher -> View (xử lý logic) -> Model (truy vấn DB) -> Serializer -> Response về Client.

3. Clean Architecture Diagram:

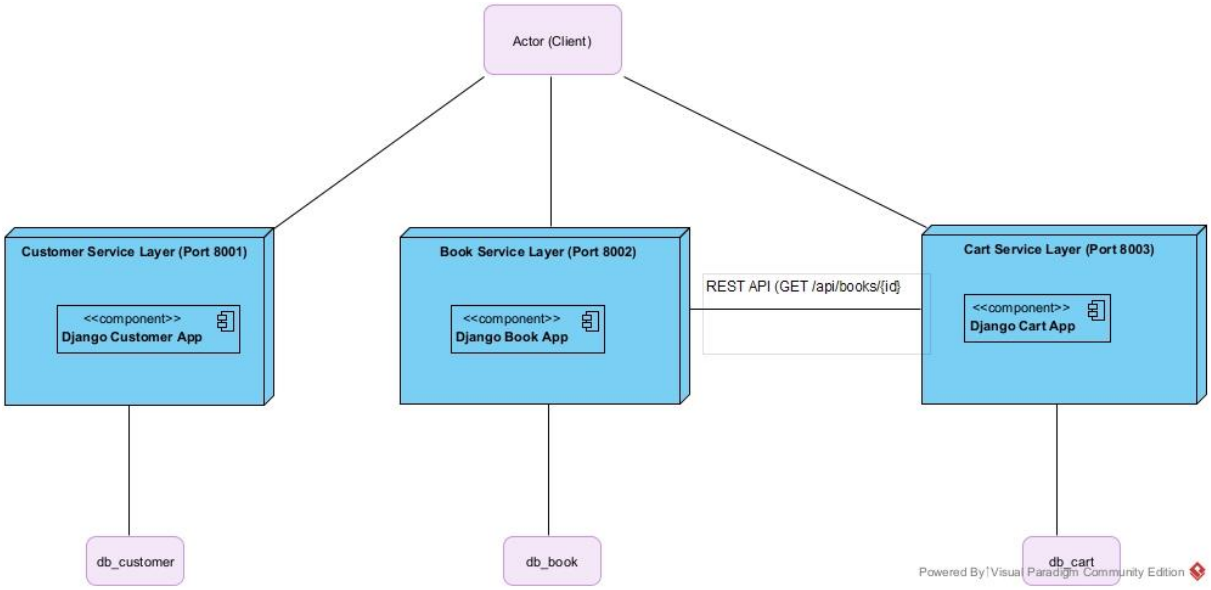
Phiên bản này tuân thủ nghiêm ngặt Quy tắc phụ thuộc (Dependency Rule), chia hệ thống thành các vòng tròn đồng tâm, với nguyên tắc: Các lớp bên trong không được biết gì về các lớp bên ngoài.



- Entities (Vòng tròn vàng - Tâm):
    - Chứa các Enterprise Business Rules (Book, Cart, Customer). Đây là các đối tượng Python thuần, hoàn toàn độc lập, không phụ thuộc vào Django hay Database.
- Use Cases (Vòng tròn hồng):
    - Chứa Application Business Rules (ví dụ: AddToCartUseCase). Tại đây xử lý các luồng nghiệp vụ cụ thể như kiểm tra tồn kho trước khi thêm vào giỏ.
- Interface Adapters (Vòng tròn xanh lá):
    - Cầu nối giữa Framework và Use Cases. Bao gồm CartController, CartSerializer và các Repository Implementation (DjangoBookRepository).
    - Sử dụng cơ chế Dependency Injection thông qua các Interface (IBookRepository, ICartRepository) để đảo ngược sự phụ thuộc.
- Frameworks & Drivers (Vòng tròn xanh dương ngoài cùng): Chứa các chi tiết kỹ thuật như DjangoFramework và MySQL Database.

4. Microservices Architecture Diagram:

Hệ thống được phân rã thành 3 dịch vụ độc lập, mỗi dịch vụ là một Django Project riêng biệt chạy trên một cổng (Port) và Database riêng:



- Customer Service (Port 8001):
    - Chịu trách nhiệm quản lý người dùng và xác thực.
    - Sử dụng database riêng: db_customer.
- Book Service (Port 8002):
    - Quản lý danh mục sách và tồn kho.
    - Cung cấp REST API (ví dụ: GET /api/books/{id}) để các service khác tra cứu thông tin.
    - Sử dụng database riêng: db_book.
- Cart Service (Port 8003):
    - Quản lý giỏ hàng của khách.
    - Sử dụng database riêng: db_cart.
- Cơ chế giao tiếp (Communication):
    - Sử dụng giao tiếp đồng bộ (Synchronous) qua HTTP/REST API.
    - Khi người dùng thêm sách vào giỏ (Cart Service), hệ thống sẽ thực hiện một HTTP Request sang Book Service để kiểm tra giá và tồn kho thực tế trước khi lưu vào database của Cart.