BAN CƠ YẾU CHÍNH PHỦ
HỌC VIỆN KỸ THUẬT MẬT MÃ

BÁO CÁO MÔN HỌC

THƯƠNG MẠI ĐIỆN TỬ

Ngành: Công nghệ thông tin

Sinh viên thực hiện:

Lê Trọng Thiên - CT07N0151

Hồ Kỳ Hiếu – CT07N0118

Giảng viên hướng dẫn

Ths. Trần Đắc Tốt

TP. Hồ Chí Minh - 2025

LỜI CẢM ƠN

Em xin gửi lời cảm ơn chân thành tới giảng viên hướng dẫn đã tận tình chỉ bảo, hỗ trợ và tạo điều kiện thuận lợi để em hoàn thành đồ án này. Sự hướng dẫn, động viên và góp ý của thầy/cô là nguồn động lực lớn giúp em vượt qua khó khăn trong quá trình nghiên cứu và phát triển hệ thống.

CAM KẾT

Em cam kết toàn bộ hệ thống được xây dựng trong đồ án là sản phẩm tự nghiên cứu, thiết kế và phát triển, không sao chép mã nguồn từ các dự án khác. Các thành phần mã nguồn mở (opensource) sử dụng đều tuân thủ giấy phép hợp lệ, được tích hợp đúng quy định và có trích dẫn rõ ràng trong tài liệu tham khảo. Nếu phát hiện vi phạm, em xin hoàn toàn chịu trách nhiệm trước nhà trường và pháp luật.

TÓM TẮT NỘI DUNG

Đề tài tập trung xây dựng hệ thống thương mại điện tử đa dịch vụ, vận hành theo mô hình microservices, bao gồm các phân hệ: quản lý người dùng, sản phẩm, giao dịch, ví điện tử và thông báo. Hệ thống hướng tới việc cung cấp nền tảng mua bán trực tuyến an toàn, hiệu quả, hỗ trợ nhiều loại sản phẩm và phương thức thanh toán hiện đại.

Các mục tiêu chính của đồ án:

- Thiết kế và triển khai kiến trúc microservices đảm bảo khả năng mở rộng, bảo trì.

- Xây dựng các chức năng nghiệp vụ: đăng ký, đăng nhập, quản lý sản phẩm, đặt hàng, thanh toán, gửi thông báo.

- ảm bảo tính bảo mật, xác thực người dùng, bảo vệ dữ liệuĐ giao dịch.

- Tích hợp các công nghệ hiện đại: Spring Boot, Docker, MySQL,MongoDb, Kafka, JWT, WebSocket, React, Tailwinds.

Phạm vi áp dụng: hệ thống dành cho các doanh nghiệp vừa và nhỏ, cá nhân kinh doanh mong muốn triển khai nền tảng thương mại điện tử riêng, hỗ trợ đa dạng sản phẩm số.

Kết quả đạt được: Đã xây dựng thành công hệ thống gồm các service độc lập, giao tiếp qua API và message broker, đáp ứng các yêu cầu nghiệp vụ cơ bản của thương mại điện tử, đảm bảo tính bảo mật, hiệu năng và khả năng mở rộng. Hệ thống đã được kiểm thử chức năng, sẵn sàng triển khai thực tế và phát triển thêm các tính năng nâng cao trong tương lai.

MỤC LỤC

DANH MỤC TỪ VIẾT TẮT & THUẬT NGỮ

• UI (User Interface): Giao diện người dùng

• UX (User Experience): Trải nghiệm người dùng

• API (Application Programming Interface): Giao diện lập trình ứng dụng, chuẩn giao tiếp giữa các service

• JWT (JSON Web Token): Chuẩn token xác thực, truyền thông tin người dùng giữa các service

• CDN (Content Delivery Network): Mạng phân phối nội dung, tăng tốc truy cập tài nguyên tĩnh

• KYC (Know Your Customer): Quy trình xác thực định danh người dùng

• 2FA (Two-Factor Authentication): Xác thực hai lớp

• OTP (One-Time Password): Mã xác thực một lần

• ORM (Object-Relational Mapping): Kỹ thuật ánh xạ đối tượng sang cơ sở dữ liệu

• CRUD (Create, Read, Update, Delete): Các thao tác cơ bản với dữ liệu

• RESTful API: Chuẩn thiết kế API sử dụng HTTP, hỗ trợ các phương thức GET, POST, PUT, DELETE

• Kafka: Hệ thống message broker, truyền sự kiện phi đồng bộ giữa các service

• Redis: Hệ thống lưu trữ cache, tăng tốc truy vấn, hỗ trợ bảo mật

• Docker: Công nghệ đóng gói, triển khai ứng dụng dạng container

• JWKS (JSON Web Key Set): Chuẩn cung cấp public key xác thực chữ ký số cho JWT

• SIEM (Security Information and Event Management): Hệ thống giám sát, cảnh báo bảo mật

• CI/CD (Continuous Integration/Continuous Deployment): Quy trình tích hợp và triển khai liên tục

• BPMN (Business Process Model and Notation): Chuẩn mô hình hóa quy trình nghiệp vụ

• Microservices: Kiến trúc dịch vụ nhỏ, độc lập, giao tiếp qua API

CHƯƠNG 1: GIỚI THIỆU

1.1 Lý do chọn đề tài

Trong bối cảnh chuyển đổi số diễn ra mạnh mẽ trên toàn cầu, các sản phẩm kỹ thuật số như key game, thẻ nạp, gift card, tài khoản dịch vụ trực tuyến, phần mềm bản quyền ngày càng trở nên phổ biến và có giá trị cao. Tuy nhiên, các nền tảng giao dịch hiện tại chủ yếu tập trung vào sản phẩm vật lý, chưa đáp ứng tốt nhu cầu giao dịch trực tiếp các sản phẩm số giữa người mua và người bán. Việc thiếu cơ chế trung gian đảm bảo, quy trình xác thực, bảo mật và hỗ trợ khiếu nại khiến người dùng gặp nhiều rủi ro khi giao dịch online. Đề tài "Xây dựng hệ thống thương mại điện tử Digital P2P" ra đời nhằm giải quyết các vấn đề trên, tạo ra một marketplace chuyên biệt cho sản phẩm số, đảm bảo an toàn, minh bạch, tối ưu hóa trải nghiệm cho cả người mua, người bán và quản trị viên.

1.2 Mục tiêu nghiên cứu và phát triển hệ thống

Mục tiêu tổng quát của đồ án là xây dựng một hệ thống thương mại điện tử vận hành theo mô hình peer-to-peer (P2P), tập trung vào giao dịch các sản phẩm số. Các mục tiêu cụ thể gồm:

• Thiết kế kiến trúc microservices, đảm bảo tính module hóa, dễ mở rộng, bảo trì.

• Xây dựng các chức năng nghiệp vụ: quản lý gian hàng, đăng bán sản phẩm, quản lý tồn kho, xử lý đơn hàng, quản lý ví điện tử, gửi thông báo, hỗ trợ khiếu nại, đánh giá người bán.

• Đảm bảo các yêu cầu phi chức năng: bảo mật giao dịch, xác thực người dùng, hiệu năng hệ thống, khả năng mở rộng, tính minh bạch và giám sát hoạt động.

• Tích hợp các công nghệ hiện đại: Spring Boot, Docker, MySQL, Kafka, JWT, WebSocket, React...

• Cung cấp tài liệu đặc tả đầy đủ về yêu cầu chức năng, phi chức năng, các ràng buộc thiết kế và các yếu tố cần thiết cho quá trình phát triển, triển khai hệ thống.

1.3 Phạm vi áp dụng

Phạm vi của hệ thống Digital P2P bao gồm:

• Đối tượng sử dụng: các doanh nghiệp vừa và nhỏ, cá nhân kinh doanh mong muốn triển khai nền tảng thương mại điện tử cho sản phẩm số; các bên liên quan như kiến trúc sư hệ thống, lập trình viên, nhà thiết kế giao diện, quản trị viên.

• Loại sản phẩm: các sản phẩm kỹ thuật số (key game, thẻ nạp, gift card, tài khoản dịch vụ trực tuyến, phần mềm bản quyền, dịch vụ số).

• Chức năng: quản lý gian hàng, đăng bán sản phẩm, quản lý tồn kho, xử lý đơn hàng, quản lý ví điện tử, gửi thông báo, hỗ trợ khiếu nại, đánh giá người bán, giám sát hoạt động hệ thống.

• Phạm vi tài liệu: mô tả đầy đủ các yêu cầu chức năng, phi chức năng, các ràng buộc thiết kế, các yếu tố kỹ thuật cần thiết cho phát triển và triển khai hệ thống.

1.4 Phương pháp nghiên cứu

Phương pháp nghiên cứu của đồ án bao gồm:

• Tổng hợp, phân tích các tài liệu chuyên ngành về thương mại điện tử, mô hình P2P, kiến trúc microservices.

• Khảo sát thực tế các nền tảng marketplace trong và ngoài nước, đánh giá ưu nhược điểm, xác định bài toán cần giải quyết.

• Phân tích nghiệp vụ chi tiết, xây dựng mô hình quy trình nghiệp vụ (BPMN), xác định các tác nhân và luồng sự kiện chính.

• Thiết kế kiến trúc hệ thống, mô hình dữ liệu, giao diện người dùng, API.

• Xây dựng, kiểm thử từng phân hệ (user, product, transaction, wallet, notify), tích hợp toàn hệ thống.

• Đánh giá, kiểm thử chức năng, hiệu năng, bảo mật, khả năng mở rộng của hệ thống.

CHƯƠNG 2: CƠ SỞ LÝ THUYẾT & CÔNG NGHỆ

2.1 Kiến thức nền tảng về thương mại điện tử

Thương mại điện tử (E-commerce) là mô hình kinh doanh dựa trên nền tảng số, cho phép các bên thực hiện giao dịch mua bán sản phẩm/dịch vụ qua Internet. Các mô hình phổ biến gồm:

• B2C (Business to Customer): doanh nghiệp bán hàng cho khách hàng cá nhân.

• B2B (Business to Business): doanh nghiệp giao dịch với doanh nghiệp khác.

• C2C (Customer to Customer): khách hàng giao dịch trực tiếp với nhau qua nền tảng trung gian.

Hệ thống Digital P2P thuộc mô hình C2C, tập trung vào sản phẩm số, tối ưu hóa quy trình giao dịch, đảm bảo an toàn, minh bạch cho các bên tham gia.

2.2 Các khái niệm nghiệp vụ

• Gian hàng (Store): không gian riêng cho người bán đăng sản phẩm, quản lý tồn kho, xử lý đơn hàng.

• Sản phẩm số (Digital Goods): key game, thẻ nạp, gift card, tài khoản dịch vụ, phần mềm bản quyền, dịch vụ số.

• Đơn hàng (Order): quá trình đặt mua sản phẩm, xác nhận, thanh toán, giao nhận sản phẩm số.

• Ví điện tử (E-wallet): quản lý số dư, rút tiền, nạp tiền, thanh toán giao dịch.

• Thanh toán đảm bảo (Escrow): cơ chế giữ tiền tạm thời, chỉ chuyển cho người bán khi giao dịch hoàn tất.

• Thông báo (Notification): gửi thông tin realtime tới người dùng qua WebSocket.

• Khiếu nại, đánh giá: hỗ trợ xử lý tranh chấp, phản hồi chất lượng giao dịch.

2.3 Các công nghệ sử dụng

• Spring Boot 3.5.x: framework backend, hỗ trợ phát triển microservices, RESTful API.

• Java 21: ngôn ngữ lập trình chính, đảm bảo hiệu năng và bảo mật.

• Spring Security, OAuth2 Authorization Server: xác thực, phân quyền, bảo vệ tài nguyên.

• Spring Data JPA (user, transaction, wallet): quản lý dữ liệu với MySQL.

• Spring Data MongoDB (product): quản lý dữ liệu phi cấu trúc với MongoDB.

• Kafka: message broker, giao tiếp giữa các service, xử lý sự kiện phi đồng bộ.

• WebSocket (notify): gửi thông báo realtime tới người dùng.

• Lombok: giảm boilerplate code, tăng hiệu quả phát triển.

• OpenAPI/Swagger: tài liệu hóa API, hỗ trợ kiểm thử và tích hợp.

• Docker: đóng gói, triển khai các service độc lập, dễ dàng mở rộng.

2.4 Công cụ hỗ trợ phát triển

• Maven: quản lý dependencies, build project.

• Git: quản lý mã nguồn, phối hợp nhóm.

• Postman: kiểm thử API, mô phỏng request/response.

• VSCode, IntelliJ IDEA: môi trường phát triển, hỗ trợ debug, refactor.

• Figma: thiết kế giao diện, wireframe/mockup.

CHƯƠNG 3: PHÂN TÍCH HỆ THỐNG

3.1 Khảo sát nghiệp vụ

Hoạt động kinh doanh sản phẩm số trên hệ thống Digital P2P được tổ chức thành các quy trình nghiệp vụ rõ ràng, đảm bảo an toàn, minh bạch và tối ưu hóa trải nghiệm người dùng. Các chức năng chính được triển khai qua các controller của từng service:

• Đăng ký/đăng nhập, xác thực người dùng (AuthController): quản lý tài khoản, bảo mật, hỗ trợ xác thực hai lớp, đổi mật khẩu, quên mật khẩu.

• Quản lý thông tin người dùng, duyệt hồ sơ KYC, quản lý trạng thái tài khoản (AdminController).

• Quản lý hồ sơ và đánh giá người bán, nộp đơn đăng ký seller, chấm điểm seller (SellerController).

• Quản lý sản phẩm, danh mục, biến thể sản phẩm, upload hình ảnh (ProductController, CategoryController, ProductVariantController).

• Đặt hàng, xem chi tiết đơn hàng, xác nhận giao dịch, upload bằng chứng giao dịch (OrderController).

• Xử lý khiếu nại, mở tranh chấp, cập nhật trạng thái tranh chấp (OrderDisputeController).

• Quản lý thanh toán, kiểm tra trạng thái, xử lý giao dịch qua nhiều phương thức (PaymentController).

Quy trình nghiệp vụ tổng quát:

1. Người dùng đăng ký/đăng nhập, xác thực tài khoản.

2. Người bán nộp hồ sơ KYC, đăng ký seller, tạo gian hàng, đăng bán sản phẩm.

3. Người mua tìm kiếm, lựa chọn sản phẩm, đặt hàng, thanh toán.

4. Hệ thống xác thực giao dịch, giữ tiền tạm thời (escrow), chuyển sản phẩm số cho người mua.

5. Sau khi giao dịch hoàn tất, tiền được chuyển cho người bán, người mua có thể đánh giá, khiếu nại nếu cần.

6. Quản trị viên giám sát, xử lý tranh chấp, duyệt hồ sơ, đảm bảo minh bạch hệ thống.

3.2 Mô hình nghiệp vụ (BPMN)

Mô hình BPMN tổng quan các luồng nghiệp vụ chính của hệ thống:

• Đăng ký/đăng nhập → Xác thực tài khoản → Nộp hồ sơ KYC → Đăng ký seller → Quản lý gian hàng → Đăng bán sản phẩm → Đặt hàng → Thanh toán → Xác thực giao dịch → Giao nhận sản phẩm số → Đánh giá/hoàn tất giao dịch.

• Quản lý sản phẩm: thêm/sửa/xóa sản phẩm, quản lý danh mục, biến thể, upload hình ảnh.

• Quản lý ví điện tử: nạp tiền, rút tiền, kiểm tra số dư, thanh toán đơn hàng, kiểm tra trạng thái giao dịch.

• Xử lý khiếu nại/tranh chấp: mở tranh chấp, cập nhật trạng thái, upload bằng chứng, giải quyết tranh chấp giữa các bên.

• Quản trị viên: duyệt hồ sơ, giám sát hoạt động, xử lý tranh chấp, quản lý khách hàng.

3.3 Các tác nhân

• Người bán (Seller): đăng ký tài khoản, nộp hồ sơ KYC, đăng ký seller, tạo gian hàng, đăng bán sản phẩm, quản lý tồn kho, nhận tiền sau giao dịch, phản hồi đánh giá.

• Người mua (Buyer): đăng ký/đăng nhập, tìm kiếm sản phẩm, đặt hàng, thanh toán, nhận sản phẩm số, đánh giá người bán, mở tranh chấp khi cần.

• Quản trị viên (Admin): duyệt hồ sơ KYC, quản lý tài khoản, giám sát hoạt động, quản lý sản phẩm, xử lý tranh chấp, quản lý khách hàng, đảm bảo minh bạch hệ thống.

3.4 Use Case tổng quan

Hệ thống Digital P2P được tổ chức thành các nhóm Use Case theo từng phân hệ chức năng, mỗi Use Case được mã hóa và mô tả rõ ràng:

3.4.1 Phân hệ Người dùng (End-User / Buyer)

UC01-010: Đăng ký tài khoản

UC01-020: Đăng nhập hệ thống

UC01-030: Quản lý mật khẩu (Đổi, Quên, Reset)

UC01-040: Quản lý hồ sơ cá nhân (Xem, cập nhật, avatar, địa chỉ thanh toán)

UC01-050: Quản lý tuỳ chọn cá nhân (User Preferences, Settings)

UC01-060: Quản lý bảo mật (2FA, OTP, đăng xuất phiên, refresh token)

3.4.2 Phân hệ Sản phẩm (Catalog / Seller)

UC02-010: Xem danh sách sản phẩm

UC02-020: Xem chi tiết sản phẩm

UC02-030: Thêm sản phẩm mới (Seller)

UC02-040: Cập nhật sản phẩm (Seller)

UC02-050: Xoá sản phẩm (Seller)

UC02-060: Quản lý hình ảnh sản phẩm

UC02-070: Quản lý biến thể sản phẩm (thêm, sửa, xoá, liệt kê)

UC02-080: Quản lý tồn kho (đặt giữ, huỷ giữ, trừ kho khi thanh toán)

UC02-090: Quản lý danh mục sản phẩm (Admin)

3.4.3 Phân hệ Giỏ hàng & Đặt hàng (Buyer / Seller)

UC03-010: Tạo đơn hàng

UC03-020: Xem đơn hàng (chi tiết, lịch sử)

UC03-030: Huỷ đơn hàng

UC03-040: Theo dõi trạng thái đơn hàng

UC03-050: Xử lý đơn hàng (Seller xác nhận, fulfilment)

UC03-060: Upload bằng chứng giao hàng

UC03-070: Quản lý tranh chấp & hoàn tiền

3.4.4 Phân hệ Thanh toán & Ví điện tử

UC04-010: Tạo ví điện tử (tự động khi user đăng ký)

UC04-020: Xem số dư, lịch sử giao dịch

UC04-030: Nạp tiền / Thanh toán bằng ví

UC04-040: Thanh toán qua cổng VNPAY

UC04-050: Theo dõi trạng thái thanh toán

UC04-060: Rút tiền về tài khoản ngân hàng

UC04-070: Quản lý tài khoản ngân hàng liên kết

3.4.5 Phân hệ Thông báo (Notification)

UC05-010: Nhận email xác nhận đăng ký

UC05-020: Nhận email reset mật khẩu

UC05-030: Nhận OTP 2FA

UC05-040: Nhận cảnh báo bảo mật (login từ thiết bị mới)

UC05-050: Nhận URL thanh toán real-time qua WebSocket

UC05-060: Nhận thông báo đơn hàng/thanh toán real-time

3.4.6 Phân hệ Quản trị (Admin)

UC06-010: Quản lý người dùng (duyệt, khoá, xoá, reset)

UC06-020: Quản lý sản phẩm (giám sát, duyệt)

UC06-030: Giám sát giao dịch & hệ thống

3.4.7 Phân hệ KYC & Người bán

UC07-010: Gửi thông tin định danh (KYC)

UC07-020: Đăng ký trở thành người bán (Seller Application)

UC07-030: Quản lý trạng thái duyệt hồ sơ KYC (Admin)

CHƯƠNG 4: THIẾT KẾ HỆ THỐNG

4.1 Thiết kế kiến trúc

Hệ thống Digital P2P được xây dựng theo mô hình kiến trúc microservices, mỗi phân hệ nghiệp vụ được triển khai thành một service độc lập, giao tiếp qua API và message broker (Kafka). Các service chính gồm:

• User Service: Quản lý người dùng, xác thực, phân quyền, quản lý hồ sơ, KYC, seller, bảo mật (2FA, OTP), preferences.

• Product Service: Quản lý sản phẩm, danh mục, biến thể, tồn kho, hình ảnh, cung cấp API cho buyer/seller và admin.

• Transaction Service: Quản lý đơn hàng, trạng thái đơn hàng, xử lý giao dịch, upload bằng chứng, quản lý tranh chấp, hoàn tiền.

• Wallet Service: Quản lý ví điện tử, số dư, lịch sử giao dịch, nạp/rút tiền, liên kết ngân hàng, tích hợp cổng thanh toán VNPAY.

• Notify Service: Gửi thông báo realtime qua WebSocket, email xác nhận, OTP, cảnh báo bảo mật, cập nhật trạng thái đơn hàng/thanh toán.

Các service được triển khai độc lập, đóng gói bằng Docker, dễ dàng mở rộng, bảo trì và triển khai trên nhiều môi trường (local, cloud, container orchestration). Giao tiếp giữa các service sử dụng RESTful API cho các tác vụ đồng bộ và Kafka cho các sự kiện phi đồng bộ (event-driven).

Kiến trúc tổng thể đảm bảo:

• Tách biệt nghiệp vụ, giảm phụ thuộc giữa các module.

• Khả năng mở rộng theo chiều ngang (scale-out) từng service.

• Đảm bảo bảo mật, xác thực, phân quyền ở từng lớp.

• Dễ dàng tích hợp các công nghệ mới, nâng cấp từng service mà không ảnh hưởng toàn hệ thống.

• Hỗ trợ giám sát, logging, tracing, quản lý trạng thái giao dịch và xử lý sự kiện realtime.

Sơ đồ kiến trúc tổng quan:

• Frontend (Web/App) → API Gateway → Các service (User, Product, Transaction, Wallet, Notify)

• Các service kết nối tới database riêng (MySQL/MongoDB), sử dụng Kafka để truyền sự kiện (order, payment, notification, KYC, ...)

• Notify Service sử dụng WebSocket để gửi thông báo realtime tới client.

4.2 Thiết kế CSDL

Hệ thống Digital P2P sử dụng mô hình cơ sở dữ liệu phân tán, mỗi service quản lý CSDL riêng biệt, đảm bảo tính module hóa, bảo mật và hiệu năng. Các service sử dụng MySQL (user, transaction, wallet) và MongoDB (product) với các bảng/collection chính như sau:

4.2.1. User Service (MySQL)

• user_auth: Quản lý thông tin xác thực, trạng thái, bảo mật, 2FA, KYC.

• user_inf: Hồ sơ cá nhân, thông tin hiển thị, trạng thái seller, avatar, địa chỉ.

• roles, user_roles: Quản lý phân quyền, mapping user-role.

• preferences: Tuỳ chọn cá nhân, thông báo, currency, privacy.

• audit_logs, login_history, password_history, device_manager: Lịch sử hoạt động, bảo mật, quản lý thiết bị.

• seller_ratings: Đánh giá seller, liên kết buyer, transaction.

• user_verifications, delete_kyc_requests: Quản lý hồ sơ KYC, trạng thái xác minh, tài liệu định danh.

• billing_address: Quản lý địa chỉ thanh toán.

• seller_applications: Quản lý hồ sơ đăng ký seller, trạng thái duyệt.

4.2.2. Product Service (MongoDB)

• Product: Thông tin sản phẩm số, seller, giá, tồn kho, trạng thái, mô tả.

• Category: Danh mục sản phẩm, phân loại, mapping sản phẩm.

• ProductVariant: Biến thể sản phẩm (loại, giá, tồn kho riêng).

• ProductImage: Quản lý hình ảnh sản phẩm, liên kết sản phẩm.

• ProductReview: Đánh giá sản phẩm, liên kết buyer, sản phẩm.

• AuditLog: Lịch sử thay đổi sản phẩm, tracking seller/admin.

• SendMessageError: Lưu lỗi gửi message (Kafka, notification).

4.2.3. Transaction Service (MySQL)

• orders: Quản lý đơn hàng, buyer, seller, trạng thái, tổng tiền, currency.

• order_items: Sản phẩm trong đơn hàng, biến thể, số lượng, giá.

• order_proofs: Bằng chứng giao hàng, hình ảnh/video/text, seller upload.

• order_refunds: Quản lý hoàn tiền, trạng thái, lý do, thời gian xử lý.

• order_disputes: Quản lý tranh chấp, loại vấn đề, mô tả, trạng thái, thời gian xử lý.

• order_logs: Lịch sử trạng thái đơn hàng, tracking thay đổi, người thực hiện.

4.2.4. Wallet Service (MySQL)

• wallets: Quản lý ví điện tử, số dư, currency, trạng thái, user mapping.

• wallet_transactions: Lịch sử giao dịch ví, loại giao dịch, số tiền, trạng thái, idempotency.

• wallet_reservations: Quản lý số tiền giữ tạm thời (escrow), liên kết đơn hàng.

• payments: Quản lý giao dịch thanh toán, provider, trạng thái, liên kết order/payment.

• payment_attempts: Lưu chi tiết từng lần thực hiện thanh toán, callback provider.

• withdrawal_requests: Quản lý yêu cầu rút tiền, trạng thái, liên kết ngân hàng.

• bank_accounts: Quản lý tài khoản ngân hàng liên kết, mã hóa thông tin.

• idempotency_store: Quản lý idempotency cho các giao dịch quan trọng.

4.2.5. Notify Service

• Lưu trạng thái gửi thông báo, lỗi gửi, tracking event (Kafka, WebSocket, email).

Các bảng/collection được thiết kế với khóa chính UUID, đảm bảo tính duy nhất, hỗ trợ mở rộng. Các mối quan hệ giữa bảng được quản lý qua khóa ngoại, đảm bảo toàn vẹn dữ

liệu, hỗ trợ truy vấn hiệu quả. Các bảng lịch sử, log, audit giúp giám sát, truy vết hoạt động hệ thống, phục vụ bảo mật và compliance.

CHƯƠNG 5: XÂY DỰNG & TRIỂN KHAI

5.1 Công cụ, ngôn ngữ và framework sử dụng

- Ngôn ngữ lập trình: Java 21 (backend), JavaScript/TypeScript (frontend)

- Framework backend: Spring Boot 3.5.x, Spring Security, Spring Data JPA, Spring Data MongoDB, OAuth2 Authorization Server

- Framework frontend: ReactJS, Tailwind CSS

- Message broker: Apache Kafka

- Realtime: WebSocket

- Database: MySQL (user, transaction, wallet), MongoDB (product)

- Công cụ CI/CD: Docker, Maven, Git

- Thiết kế giao diện: Figma

- Kiểm thử API: Postman

- IDE: VSCode, IntelliJ IDEA

5.2 Quá trình xây dựng hệ thống

- Phân tích yêu cầu, thiết kế kiến trúc tổng thể, xác định các phân hệ nghiệp vụ.

- Xây dựng từng service độc lập theo mô hình microservices, đảm bảo tách biệt nghiệp vụ và dữ liệu.

- Thiết kế và triển khai các API RESTful cho từng service, tích hợp xác thực OAuth2, bảo mật JWT.

- Tích hợp Kafka cho các sự kiện phi đồng bộ (order, payment, notification, KYC).

- Xây dựng giao diện người dùng với ReactJS, kết nối tới API Gateway.

- Kiểm thử chức năng từng service bằng Postman, kiểm thử tích hợp toàn hệ thống.

- Đóng gói các service bằng Docker, chuẩn bị file cấu hình cho môi trường triển khai.

5.3 Mô tả triển khai hệ thống

- Môi trường phát triển: Sử dụng Docker Compose để khởi tạo các service, database, Kafka.

- Môi trường triển khai: VPS , sử dụng Docker để triển khai từng service độc lập, cấu hình domain, SSL/TLS cho API Gateway.

- Quản lý cấu hình môi trường qua file application.properties, yaml.

- Tích hợp giám sát, logging, tracing cho từng service.

- Hệ thống hỗ trợ mở rộng dễ dàng.

5.4 Demo hình ảnh (bổ sung sau)

- Hình ảnh giao diện chính (trang chủ, đăng nhập, quản lý sản phẩm, đặt hàng, ví điện tử, thông báo realtime).

- Sơ đồ luồng mua hàng, trạng thái đơn hàng, ví dụ về thông báo realtime.

- (Lưu ý: Hình ảnh sẽ được bổ sung khi hoàn thiện giao diện và kiểm thử thực tế.)

CHƯƠNG 6: KIỂM THỬ HỆ THỐNG

6.1 Chiến lược kiểm thử

- Kiểm thử chức năng từng service thông qua API sử dụng Postman.

- Tập trung kiểm thử các luồng nghiệp vụ chính: đăng ký, đăng nhập, quản lý sản phẩm, đặt hàng, thanh toán, ví điện tử, thông báo realtime.

- Kiểm thử tích hợp giữa các service qua API Gateway và Kafka.

- Kiểm thử bảo mật: xác thực OAuth2, kiểm tra phân quyền, kiểm thử các endpoint nhạy cảm.

- Kiểm thử dữ liệu: kiểm tra tính toàn vẹn dữ liệu, các trường hợp lỗi, trạng thái giao dịch.

6.2 Các ca kiểm thử chính

- Đăng ký tài khoản mới (User Service)
- Đăng nhập, xác thực 2FA, đổi mật khẩu
- Quản lý hồ sơ cá nhân, cập nhật thông tin
- Đăng ký seller, duyệt hồ sơ KYC
- Thêm, sửa, xoá sản phẩm (Product Service)
- Đặt hàng, kiểm tra trạng thái đơn hàng (Transaction Service)
- Upload bằng chứng giao hàng, mở tranh chấp, hoàn tiền
- Tạo ví điện tử, kiểm tra số dư, nạp/rút tiền (Wallet Service)
- Thanh toán qua ví, qua cổng VNPAY
- Nhận thông báo realtime qua WebSocket (Notify Service)

6.3 Kết quả kiểm thử

- Các chức năng chính đều hoạt động đúng theo đặc tả nghiệp vụ.
- Các API trả về kết quả đúng, trạng thái HTTP chuẩn, thông báo lỗi rõ ràng.
- Kiểm thử tích hợp: các service giao tiếp ổn định qua API Gateway và Kafka, dữ liệu đồng bộ giữa các phân hệ.
- Kiểm thử bảo mật: xác thực OAuth2, JWT, phân quyền user/admin/seller đảm bảo đúng logic.
- Một số trường hợp lỗi (input sai, trạng thái bất thường) được xử lý đúng, trả về thông báo hợp lệ.

6.4 Đánh giá mức độ hoàn thiện

- Hệ thống đã đáp ứng đầy đủ các yêu cầu chức năng cơ bản của thương mại điện tử số.
- Các luồng nghiệp vụ chính được kiểm thử đầy đủ bằng Postman, đảm bảo tính ổn định và bảo mật.

- Hệ thống sẵn sàng triển khai thực tế, có thể mở rộng thêm kiểm thử tự động, kiểm thử hiệu năng và bảo mật nâng cao trong tương lai.

CHƯƠNG 7: BẢO MẬT & HIỆU NĂNG

7.1 Các giải pháp bảo mật

- Hệ thống áp dụng kiến trúc bảo mật đa lớp, đảm bảo an toàn cho dữ liệu, giao dịch và người dùng. Các giải pháp chính gồm:

7.1.1 Xác thực và phân quyền

• Sử dụng OAuth2 Authorization Server cho xác thực người dùng, cấp phát access token dạng JWT.

• Các endpoint được bảo vệ bằng annotation @PreAuthorize, phân quyền rõ ràng theo vai trò (ROLE_USER, ROLE_SELLER, ROLE_ADMIN).

• Cấu hình SecurityFilterChain (xem mẫu Security.java) kiểm soát truy cập từng API, phân biệt endpoint public/private, mapping quyền cho từng HTTP method và URL.

• Các endpoint nhạy cảm (quản lý sản phẩm, seller, admin) chỉ cho phép truy cập với token hợp lệ và quyền phù hợp.

7.1.2 Quản lý token và session

• Sử dụng JWT cho xác thực stateless, lưu thông tin user, role, thời gian hiệu lực, xác thực chữ ký số RSA.

• Token được sinh ra bởi JwtTokenFactory, xác thực và giải mã bởi JwtUtils, JwtTokenConfig, RsaKeyConfig.

• Hệ thống sử dụng JWKS (JSON Web Key Set) để cung cấp public key xác thực chữ ký token cho các service qua Auth Service. Các service (product, transaction, wallet...) sẽ tự động lấy public key từ endpoint JWKS của Auth để giải mã và xác thực token, đảm bảo tính linh hoạt và bảo mật khi thay đổi key.

• Token được kiểm tra ở mỗi request, các trường hợp hết hạn, sai chữ ký, bị thu hồi đều bị từ chối truy cập.

• Refresh token, logout, revoke token được quản lý qua AuthService, đảm bảo user có thể đăng xuất, đổi thiết bị an toàn.

7.1.3 Bảo vệ dữ liệu và chống tấn công

• Cấu hình CORS kiểm soát domain truy cập, chỉ cho phép các domain hợp lệ (wezd.io.vn, admin, product).

• CSRF được disable do sử dụng JWT và stateless session.

• SessionCreationPolicy.STATELESS đảm bảo không lưu session trên server, giảm nguy cơ tấn công session hijacking.

• Các thông tin nhạy cảm (mật khẩu, private key) được mã hóa, không lưu plaintext trong database.

• Quản lý thiết bị đăng nhập, OTP, 2FA, xác thực KYC, kiểm tra hành vi bất thường qua DeviceManager, OtpModel, WhiteList, lưu cache trên Redis.

• Redis được sử dụng để lưu cache OTP, device, whitelist, tăng tốc xác thực, giảm truy vấn DB, hỗ trợ chống brute-force, kiểm soát số lần gửi OTP, thiết bị lạ.

7.1.4 Logging, audit, giám sát

• Mọi hoạt động đăng nhập, thay đổi mật khẩu, xác thực, gửi OTP, truy cập endpoint đều được ghi log, lưu audit trail.

• Các bảng audit_logs, login_history, password_history giúp truy vết, phát hiện hành vi bất thường, phục vụ compliance.

7.1.5 Bảo vệ API và hệ thống

• Các API public chỉ cho phép GET, các API thay đổi dữ liệu yêu cầu xác thực và phân quyền rõ ràng.

• Swagger UI, OpenAPI chỉ mở cho môi trường phát triển, bị hạn chế trên production.

• Các service sử dụng Docker, cấu hình bảo mật môi trường, không expose port không cần thiết.

• Thường xuyên kiểm thử bảo mật bằng Postman, kiểm tra các trường hợp lỗi, input bất thường, tấn công injection.

7.1.6 Các giải pháp nâng cao

• Hỗ trợ xác thực đa lớp (2FA, OTP), quản lý thiết bị, kiểm tra whitelist IP, chống spam OTP.

• Sử dụng Redis để lưu trạng thái OTP, thiết bị, whitelist, tăng tốc xác thực, giảm tải DB.

• Tích hợp giám sát, cảnh báo bảo mật, phát hiện truy cập bất thường, gửi cảnh báo realtime qua Notify Service.

7.2 Tối ưu hiệu năng

- Hệ thống được thiết kế và triển khai với nhiều giải pháp tối ưu hiệu năng ở cả tầng backend, database và giao tiếp giữa các service:

7.2.1 Tối ưu truy vấn và xử lý dữ liệu

• Sử dụng Spring Data JPA và MongoDB với các chỉ mục (index) phù hợp, tối ưu hóa các truy vấn phức tạp.

• Áp dụng phân trang (pagination), giới hạn kết quả trả về cho các API danh sách lớn (sản phẩm, đơn hàng, lịch sử giao dịch).

• Sử dụng cache Redis cho các dữ liệu truy cập thường xuyên (OTP, device, whitelist), giảm tải truy vấn trực tiếp tới database.

• Các thao tác ghi/lưu dữ liệu lớn (upload hình ảnh, log, audit) được xử lý bất đồng bộ, giảm thời gian phản hồi API.

7.2.2 Tối ưu giao tiếp giữa các service

• Sử dụng Kafka làm message broker cho các sự kiện phi đồng bộ (order, payment, notification), giảm độ trễ và tăng throughput hệ thống.

• Các service giao tiếp qua API Gateway, sử dụng connection pool, timeout hợp lý, giảm nguy cơ nghẽn mạng.

7.2.3 Tối ưu tầng triển khai

• Đóng gói các service bằng Docker, tối ưu cấu hình JVM, memory, thread pool cho từng service.

• Sử dụng health check, auto-restart, giám sát trạng thái service để đảm bảo uptime cao.

• Tích hợp logging, tracing, monitoring (Prometheus, Grafana) để theo dõi hiệu năng, phát hiện sớm bottleneck.

7.2.4 Tối ưu frontend

• Sử dụng ReactJS, Tailwind CSS, tối ưu bundle, lazy loading, nén ảnh, giảm thời gian tải trang.

• Sử dụng CDN cho các tài nguyên tĩnh, tăng tốc độ truy cập từ nhiều khu vực địa lý.

7.3 Khả năng mở rộng

- Hệ thống được xây dựng theo kiến trúc microservices, đảm bảo khả năng mở rộng linh hoạt cả về nghiệp vụ lẫn hạ tầng:

7.3.1 Scale-out từng service

• Mỗi service (user, product, transaction, wallet, notify) có thể mở rộng độc lập theo chiều ngang (scale-out), tăng số lượng instance khi tải lớn.

• Sử dụng Docker, Kubernetes (hoặc các nền tảng container orchestration) để tự động scale, cân bằng tải (load balancing).

7.3.2 Mở rộng nghiệp vụ

• Dễ dàng bổ sung thêm service mới (ví dụ: recommendation, chatbot, analytics) mà không ảnh hưởng tới các service hiện tại.

• Các API, message broker, database được thiết kế module hóa, hỗ trợ tích hợp các tính năng mới.

7.3.3 Mở rộng dữ liệu và giao dịch

• Database được phân tách theo từng service, hỗ trợ sharding, replication, backup tự động.

• Kafka hỗ trợ mở rộng số lượng topic, partition, tăng khả năng xử lý sự kiện lớn.

7.3.4 Mở rộng giám sát và bảo mật

• Hệ thống giám sát, logging, tracing có thể mở rộng theo số lượng service, hỗ trợ cảnh báo realtime.

• Các giải pháp bảo mật (JWT, JWKS, Redis, audit) được thiết kế module hóa, dễ nâng cấp khi quy mô hệ thống tăng.

CHƯƠNG 8: KẾT LUẬN & HƯỚNG PHÁT TRIỂN

8.1 Kết quả đạt được

- Đồ án đã xây dựng thành công hệ thống thương mại điện tử Digital P2P theo mô hình microservices, đáp ứng đầy đủ các yêu cầu nghiệp vụ và phi chức năng đặt ra ban đầu.

- Hệ thống gồm các service độc lập: quản lý người dùng, sản phẩm, giao dịch, ví điện tử, thông báo, đảm bảo tính module hóa, bảo mật, hiệu năng và khả năng mở rộng.

- Các chức năng chính như đăng ký, đăng nhập, quản lý sản phẩm, đặt hàng, thanh toán, gửi thông báo realtime đều hoạt động ổn định, được kiểm thử kỹ càng bằng Postman.

- Kiến trúc bảo mật đa lớp, xác thực OAuth2, JWT, phân quyền, quản lý session, cache Redis, logging/audit giúp bảo vệ dữ liệu và giao dịch an toàn.

- Hệ thống đã sẵn sàng triển khai thực tế, hỗ trợ mở rộng nghiệp vụ, tích hợp các công nghệ mới.

8.2 Hạn chế của hệ thống

- Chưa triển khai kiểm thử tự động (unit test, integration test) và kiểm thử hiệu năng chuyên sâu.

- Giao diện người dùng mới ở mức cơ bản, chưa tối ưu trải nghiệm UI/UX cho đa nền tảng.

- Chưa tích hợp các tính năng nâng cao như AI recommendation, chatbot hỗ trợ, đa kênh bán hàng (Omni-channel).

- Hệ thống giám sát, cảnh báo bảo mật mới ở mức cơ bản, cần nâng cấp để đáp ứng quy mô lớn.

- Một số quy trình nghiệp vụ (refund, dispute, KYC) cần hoàn thiện thêm logic xử lý phức tạp.

8.3 Hướng phát triển tương lai

- Bổ sung kiểm thử tự động, kiểm thử hiệu năng, kiểm thử bảo mật nâng cao.

- Nâng cấp giao diện người dùng, tối ưu UI/UX, hỗ trợ đa nền tảng (mobile, tablet).

- Tích hợp AI recommendation, chatbot hỗ trợ khách hàng, hệ thống phân tích dữ liệu giao dịch.

- Phát triển đa kênh bán hàng (Omni-channel), tích hợp ví điện tử, cổng thanh toán mới.

- Mở rộng hệ thống giám sát, cảnh báo bảo mật, tích hợp các giải pháp SIEM, cảnh báo realtime.

- Hoàn thiện các quy trình nghiệp vụ phức tạp, tối ưu logic xử lý refund, dispute, KYC.