**HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG**

**KHOA ĐA PHƯƠNG TIỆN**

**----------------------------------**



**ĐỒ ÁN**

**TỐT NGHIỆP ĐẠI HỌC**

**Đề tài: “XÂY DỰNG HỆ THỐNG QUẢN LÝ PHÒNG THÍ NGHIỆM THỰC HÀNH TẠI HỌC VIỆN CÔNG NGHỆ BCVT.”**

**Giảng viên hướng dẫn: THS.HÀ ĐÌNH DŨNG**

**Sinh viên thực hiện: LÊ VĂN TÙNG**

**Lớp: D20PTDPT**

**Khóa: 2020 - 2025**

**Hệ: ĐẠI HỌC CHÍNH QUY**

**HÀ NỘI – 2024**

**LỜI CẢM ƠN**

Trong khoảng thời gian làm đồ án tốt nghiệp, em đã nhận được nhiều sự giúp đỡ, đóng góp ý kiến và sự dẫn dắt chỉ bảo nhiệt tình của thầy cô, gia đình và bạn bè. Em xin gửi lời cảm ơn chân thành đến giảng viên hướng dẫn – Thạc sĩ Hà Đình Dũng người đã tận tình hướng dẫn, chỉ bảo em trong suốt quá trình làm đồ án. Cũng xin cảm ơn đến gia đình, bạn bè những người luôn ở bên cạnh đã luôn tạo điều kiện, quan tâm, giúp đỡ, động viên em trong suốt quá trình học tập và hoàn thành đồ án tốt nghiệp. Cuối cùng xin cảm ơn chính bản thân đã nỗ lực hết sức mình để hoàn thành đồ án một cách tốt nhất. Với điều kiện về thời gian cũng như lượng kiến thức còn đang hạn chế, đồ án này không thể tránh được những thiếu sót. Em rất mong nhận được sự chỉ bảo, đóng góp ý kiến của các thầy cô để em có điều kiện bổ sung, nâng cao ý thức của mình, phục vụ tốt hơn công việc thực tế sau này.

Em xin chân thành cảm ơn!

Hà Nội, ngày tháng năm 2024

Sinh viên thực hiện

**MỤC LỤC**

[TÓM TẮT ĐỒ ÁN viii](#_Toc185885680)

[DANH MỤC HÌNH VẼ ix](#_Toc185885681)

[DANH MỤC CÁC BẢNG BIỂU, LƯU ĐỒ xi](#_Toc185885682)

[DANH MỤC CÁC TỪ VIẾT TẮT xii](#_Toc185885683)

[PHẦN MỞ ĐẦU xiii](#_Toc185885684)

[CHƯƠNG 1: GIỚI THIỆU ĐỀ TÀI 1](#_Toc185885685)

[1.1. Đặt vấn đề 1](#_Toc185885686)

[1.2. Mục tiêu và phạm vi của đề tài 1](#_Toc185885687)

[1.3. Định hướng giải pháp 2](#_Toc185885688)

[1.4. Yêu cầu hệ thống 3](#_Toc185885689)

[CHƯƠNG 2: TỔNG QUAN 5](#_Toc185885690)

[2.1. Khảo sát hiện trạng 5](#_Toc185885691)

[2.1.1. Labster 6](#_Toc185885692)

[2.1.2. LMS của LabWorks 6](#_Toc185885693)

[2.1.3. Simplelab 7](#_Toc185885694)

[2.2. Tổng quan chức năng. 8](#_Toc185885695)

[2.2.1. Biểu đồ usecase tổng quát 8](#_Toc185885696)

[2.2.2. Biểu đồ usecase phân rã quản lý tài khoản 9](#_Toc185885697)

[2.2.3. Biểu đồ usecase phân rã quản lý phòng học 10](#_Toc185885698)

[2.2.4. Biểu đồ usecase phân rã quản lý thiết bị 11](#_Toc185885699)

[2.2.5. Biểu đồ usecase phân rã quản lý đặt lịch 12](#_Toc185885700)

[2.2.6. Biểu đồ usecase phân rã quản lý vai trò 13](#_Toc185885701)

[2.2.7. Biểu đồ usecase phân rã quản lý báo cáo 14](#_Toc185885702)

[2.2.8. Biểu đồ usecase phân rã quản lý loại thiết bị 15](#_Toc185885703)

[2.2.9. Biểu đồ usecase phân rã quản lý quyền truy cập 16](#_Toc185885704)

[2.3. Đặc tả chức năng 16](#_Toc185885705)

[2.3.1. Đăng nhập 16](#_Toc185885706)

[2.3.2. Đăng ký 17](#_Toc185885707)

[2.3.3. Thay đổi thông tin tài khoản 18](#_Toc185885708)

[2.3.4. Xóa tài khoản 19](#_Toc185885709)

[2.3.5. Tạo phòng 20](#_Toc185885710)

[2.3.6. Thay đổi thông tin phòng 21](#_Toc185885711)

[2.3.7. Xóa phòng 22](#_Toc185885712)

[2.3.8. Tạo thiết bị 23](#_Toc185885713)

[2.3.9. Thay đổi thông tin thiết bị 24](#_Toc185885714)

[2.3.10. Xóa thiết bị 25](#_Toc185885715)

[2.3.11. Tạo vai trò 25](#_Toc185885716)

[2.3.12. Thay đổi thông tin vai trò 26](#_Toc185885717)

[2.3.13. Xóa vai trò 27](#_Toc185885718)

[2.3.14. Tạo báo cáo 28](#_Toc185885719)

[2.3.15. Thay đổi thông tin báo cáo 29](#_Toc185885720)

[2.3.16. Xóa báo cáo 30](#_Toc185885721)

[2.3.17. Tạo loại thiết bị 31](#_Toc185885722)

[2.3.18. Thay đổi thông tin loại thiết bị 32](#_Toc185885723)

[2.3.19. Xóa loại thiết bị 33](#_Toc185885724)

[2.3.20. Tạo quyền truy cập 34](#_Toc185885725)

[2.3.21. Thay đổi thông tin quyền truy cập 34](#_Toc185885726)

[2.3.22. Xóa quyền truy cập 36](#_Toc185885727)

[2.3.23. Tạo lịch học 36](#_Toc185885728)

[2.3.24. Thay đổi thông tin lịch học 37](#_Toc185885729)

[2.3.25. Tạo lịch học bằng file excel 38](#_Toc185885730)

[2.3.26. Tạo tài khoản bằng file excel 39](#_Toc185885731)

[2.3.27. Thông báo lịch học đến email của giảng viên 40](#_Toc185885732)

[2.4. Yêu cầu phi chức năng 41](#_Toc185885733)

[2.4.1. Hiệu suất 41](#_Toc185885734)

[2.4.2. Độ tin cậy 41](#_Toc185885735)

[2.4.3. Tính dễ dùng 41](#_Toc185885736)

[2.4.4. Tính dễ bảo trì 41](#_Toc185885737)

[2.4.5. Yêu cầu về kỹ thuật 41](#_Toc185885738)

[CHƯƠNG 3: CÔNG NGHỆ SỬ DỤNG. 43](#_Toc185885739)

[3.1. Framework front-end: NextJS 43](#_Toc185885740)

[3.1.1. Giới thiệu về NextJs 43](#_Toc185885741)

[3.1.2. Các khái niệm về NextJs 43](#_Toc185885742)

[3.1.2.1. File-base Routing 43](#_Toc185885743)

[3.1.2.2. Các phương thức kết xuất 44](#_Toc185885744)

[3.1.2.3. API Routes 44](#_Toc185885745)

[3.1.2.4. Tối ưu hóa hình ảnh 45](#_Toc185885746)

[3.1.2.5. Hỗ trợ CSS linh hoạt 45](#_Toc185885747)

[3.1.2.6. Middleware 45](#_Toc185885748)

[3.1.2.7. Built-in TypeScript Support 45](#_Toc185885749)

[3.1.2.8. Tích hợp mạnh mẽ với các công nghệ hiện đại 46](#_Toc185885750)

[3.1.3. Lý do sử dụng 46](#_Toc185885751)

[3.1.3.1. Hiệu suất vượt trội 46](#_Toc185885752)

[3.1.3.2. Tối ưu SEO 46](#_Toc185885753)

[3.1.3.3. Khả năng mở rộng và đa dạng 47](#_Toc185885754)

[3.1.3.4. Trải nghiệm phát triển mượt mà 47](#_Toc185885755)

[3.1.3.5. Cộng đồng lớn và tài liệu phong phú 47](#_Toc185885756)

[3.2. Framework back-end: Gin 48](#_Toc185885757)

[3.2.1. Giới thiệu về Gin 48](#_Toc185885758)

[3.2.2. Lý do sử dụng 48](#_Toc185885759)

[3.2.2.1. Hiệu suất vượt trội 48](#_Toc185885760)

[3.2.2.2. API thân thiện và dễ sử dụng 49](#_Toc185885761)

[3.2.2.3. Middleware linh hoạt 49](#_Toc185885762)

[3.2.2.4. Tích hợp mạnh mẽ với các công cụ Go 49](#_Toc185885763)

[3.2.2.5. Xử lý dữ liệu mạnh mẽ 50](#_Toc185885764)

[3.2.2.6. Cộng đồng và tài liệu phong phú 50](#_Toc185885765)

[3.3. Redux 51](#_Toc185885766)

[3.3.1. Giới thiệu về Redux 51](#_Toc185885767)

[3.3.2. Thành phần chính và cách hoạt động 51](#_Toc185885768)

[3.3.2.1. Store 51](#_Toc185885769)

[3.3.2.2. Action 52](#_Toc185885770)

[3.3.2.3. Reducer 52](#_Toc185885771)

[3.3.2.4. Dispatch 52](#_Toc185885772)

[3.3.2.5. Selector 53](#_Toc185885773)

[3.3.2.6. Middleware 53](#_Toc185885774)

[3.4. Websocket 54](#_Toc185885775)

[3.4.1. Giới thiệu về Websocket 54](#_Toc185885776)

[3.4.2.Các thành phần chính của WebSocket: 55](#_Toc185885777)

[3.4.2.1.Client (Máy khách): 55](#_Toc185885778)

[3.4.2.2. Server (Máy chủ): 55](#_Toc185885779)

[3.4.2.3. Protocol (Giao thức): 55](#_Toc185885780)

[3.5. Docker 57](#_Toc185885781)

[3.5.1. Giới thiệu về docker 57](#_Toc185885782)

[3.5.2. Lý do sử dụng 57](#_Toc185885783)

[3.5.2.1. Tính nhất quán trong môi trường làm việc 57](#_Toc185885784)

[3.5.2.2. Dễ dàng thiết lập và triển khai cơ sở dữ liệu 57](#_Toc185885785)

[3.5.2.3. Quản lý phụ thuộc dễ dàng 58](#_Toc185885786)

[3.5.2.4. Dễ dàng mở rộng và tái sử dụng 58](#_Toc185885787)

[3.5.2.5. Tiết kiệm tài nguyên 58](#_Toc185885788)

[3.5.2.6. Cộng đồng lớn và nhiều tài nguyên 58](#_Toc185885789)

[CHƯƠNG 4: THIẾT KẾT, TRIỂN KHAI VÀ ĐÁNH GIÁ HỆ THỐNG 59](#_Toc185885790)

[4.1. Thiết kế kiến trúc 59](#_Toc185885791)

[4.1.1. Lựa chọn kiến trúc phần mềm 59](#_Toc185885792)

[4.1.2. Thiết kế tổng quan 60](#_Toc185885793)

[4.1.2.1. Thiết kế tổng quan phía backend 60](#_Toc185885794)

[4.1.2.2. Thiết kế tổng quan phía frontend 61](#_Toc185885795)

[4.2. Thiết kế chi tiết 61](#_Toc185885796)

[4.2.1. Thiết kế giao diện 61](#_Toc185885797)

[4.2.1.1. Giao diện chung 62](#_Toc185885798)

[4.2.1.2. Giao diện nhắn tin 62](#_Toc185885799)

[4.2.2. Thiết kế lớp 64](#_Toc185885800)

[4.2.2.1. Chức năng quản lý tài khoản 64](#_Toc185885801)

[4.2.2.2. Chức năng thông báo lịch học đến email của giảng viên 65](#_Toc185885802)

[4.2.2.3. Import file 66](#_Toc185885803)

[4.2.2.4. Tạo lịch học 67](#_Toc185885804)

[4.2.3. Thiết kế cơ sở dữ liệu 68](#_Toc185885805)

[4.3. Xây dựng ứng dụng 69](#_Toc185885806)

[4.3.1. Thư viện và công cụ sử dụng 69](#_Toc185885807)

[4.3.2. Kết quả đạt được 69](#_Toc185885808)

[4.3.3. Minh họa các chức năng chính 70](#_Toc185885809)

[4.4. Kiểm thử 71](#_Toc185885810)

[4.5. Triển khai 73](#_Toc185885811)

[CHƯƠNG 5: KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN 74](#_Toc185885812)

[5.1. Kết luận 74](#_Toc185885813)

[5.2. Hướng phát triển 74](#_Toc185885814)

[PHỤ LỤC I 75](#_Toc185885815)

[TÀI LIỆU THAM KHẢO 76](#_Toc185885816)

# TÓM TẮT ĐỒ ÁN

Chương 1: Giới thiệu về đề tài, chương này sẽ đưa ra thực trạng, mục tiêu phạm vi của đề tài, định hướng giải pháp và yêu cầu của hệ thống

Chương 2: Trình bày về khảo sát và phân tích yêu cầu của hệ thống, đưa ra những thông số thực tế về các hệ thống tương đương sau đó phân tích và đưa ra những yêu cầu chức năng và phi chức năng của hệ thống đặt lịch, quản lý. Kết quả của chương này sẽ là một bản phân tích chi tiết về các yêu cầu cần thiết để phát triển hệ thống.

Trong chương 3: Giới thiệu những công nghệ sử dụng để triển khai hệ thống như NextJs, Gin,...Mỗi công nghệ sẽ được mô tả về cách thức hoạt động, ưu điểm và hạn chế. Chương này sẽ cung cấp một cái nhìn tổng quan về kiến trúc và công nghệ của hệ thống.

Chương 4: Mô tả chi tiết quá trình phân tích hệ thống, từ việc thiết kế cơ sở dữ liệu, xây dựng các module chức năng, đến việc tích hợp các công nghệ đã chọn.

Chương 5: Đưa ra kết luận tổng quát về những gì đã đạt được qua quá trình nghiên cứu và phát triển hệ thống

# DANH MỤC HÌNH VẼ

**Chương 2** Trang

[Hình 2 - 1: Labster 6](#_Toc185885609)

[Hình 2 - 2: LabWorks 6](#_Toc185885610)

[Hình 2 - 3: Simplelab 7](#_Toc185885611)

[Hình 2 - 4: Biểu đồ usecase tổng quát 8](#_Toc185885612)

[Hình 2 - 5: Biểu đồ usecase quản lý tài khoản 9](#_Toc185885613)

[Hình 2 - 6: Biểu đồ usecase quản lý phòng 10](#_Toc185885614)

[Hình 2 - 7: Biểu đồ usecase quản lý thiết bị 11](#_Toc185885615)

[Hình 2 - 8: Biểu đồ usecase quản lý đặt lịch 12](#_Toc185885616)

[Hình 2 - 9: Biểu đồ usecase quản lý vai trò 13](#_Toc185885617)

[Hình 2 - 10: Biểu đồ usecase quản lý báo cáo 14](#_Toc185885618)

[Hình 2 - 11: Biểu đồ usecase quản lý loại thiết bị 15](#_Toc185885619)

[Hình 2 - 12: Biểu đồ usecase quản lý quyền truy cập 16](#_Toc185885620)

**Chương 3**

[Hình 3 - 1: NextJs 43](#_Toc185885625)

[Hình 3 - 2: Gin 48](#_Toc185885626)

[Hình 3 - 3: Redux 51](#_Toc185885627)

[Hình 3 - 4: Cách hoạt động của Redux 54](#_Toc185885628)

[Hình 3 - 5: Websocket 54](#_Toc185885629)

[Hình 3 - 6: Cách hoạt động của Websocket 56](#_Toc185885630)

[Hình 3 - 7: Docker 57](#_Toc185885631)

**Chương 4**

[Hình 4 - 1: Clean architecture 60](#_Toc185885648)

[Hình 4 - 2: Thiết kế tổng quan backend 60](#_Toc185885649)

[Hình 4 - 3: Thiết kế tổng quan phía frontend 61](#_Toc185885650)

[Hình 4 - 4: Giao diện chung 62](#_Toc185885651)

[Hình 4 - 5: Giao diện nhắn tin 63](#_Toc185885652)

[Hình 4 - 6: Biểu đồ trình tự quản lý tài khoản 64](#_Toc185885653)

[Hình 4 - 7: Lớp thiết kế chức năng quản lý tài khoản 65](#_Toc185885654)

[Hình 4 - 8: Biểu đồ trình tự chức năng gửi thông báo lịch học đến email của giảng viên 65](#_Toc185885655)

[Hình 4 - 9: Lớp thiết kế chức năng thông báo 66](#_Toc185885656)

[Hình 4 - 10: Biểu đồ tuần tự chức năng import file 66](#_Toc185885657)

[Hình 4 - 11: Lớp thiết kế chức năng import file 67](#_Toc185885658)

[Hình 4 - 13: Sequence diagram chức năng đặt lịch học 67](#_Toc185885659)

[Hình 4 - 14: Lớp thiết kế chức năng đặt lịch 68](#_Toc185885660)

[Hình 4 - 15: Thiết kế cơ sở dữ liệu 68](#_Toc185885661)

[Hình 4 - 16: Giao diện cập nhật thông tin 70](#_Toc185885662)

[Hình 4 - 17: Giao diện thống kê 70](#_Toc185885663)

[Hình 4 - 18: Giao diện đặt lịch 71](#_Toc185885664)

[Hình 4 - 19: Giao diện quản lý lịch học 71](#_Toc185885665)

[Hình 4 - 20: Giao diện nhắn tin 71](#_Toc185885666)

# DANH MỤC CÁC BẢNG BIỂU, LƯU ĐỒ

**Chương 2**

[Bảng 2 - 1: Bảng đặc tả usecase đăng nhập 16](#_Toc184556121)

[Bảng 2 - 2: Bảng đặc tả usecase tạo tài khoản 17](#_Toc184556122)

[Bảng 2 - 3: Bảng đặc tả usecase thay đổi thông tin tài khoản 18](#_Toc184556123)

[Bảng 2 - 4: Bảng đặc tả usecase xóa tài khoản 19](#_Toc184556124)

[Bảng 2 - 5: Bảng đặc tả usecase tạo phòng 20](#_Toc184556125)

[Bảng 2 - 6: Bảng đặc tả usecase thay đổi thông tin phòng 21](#_Toc184556126)

[Bảng 2 - 7: Bảng đặc tả usecase xóa phòng 22](#_Toc184556127)

[Bảng 2 - 8: Bảng đặc tả usecase tạo thiết bị 23](#_Toc184556128)

[Bảng 2 - 9: Bảng đặc tả usecase thay đổi thông tin thiết bị 24](#_Toc184556129)

[Bảng 2 - 10: Bảng đặc tả usecase xóa thiết bị 25](#_Toc184556130)

[Bảng 2 - 11: Bảng đặc tả usecase tạo vai trò 25](#_Toc184556131)

[Bảng 2 - 12: Bảng đặc tả usecase thay đổi thông tin vai trò 26](#_Toc184556132)

[Bảng 2 - 13: Bảng đặc tả usecase xóa vai trò 27](#_Toc184556133)

[Bảng 2 - 14: Bảng đặc tả usecase tạo báo cáo 28](#_Toc184556134)

[Bảng 2 - 15: Bảng đặc tả usecase thay đổi thông tin báo cáo 29](#_Toc184556135)

[Bảng 2 - 16: Bảng đặc tả usecase xóa báo cáo 30](#_Toc184556136)

[Bảng 2 - 17: Bảng đặc tả usecase tạo loại thiết bị 31](#_Toc184556137)

[Bảng 2 - 18: Bảng đặc tả usecase thay đổi thông tin loại thiết bị 32](#_Toc184556138)

[Bảng 2 - 19: Bảng đặc tả usecase xóa loại thiết bị 33](#_Toc184556139)

[Bảng 2 - 20: Bảng đặc tả usecase tạo quyền truy cập 34](#_Toc184556140)

[Bảng 2 - 21: Bảng đặc tả usecase thay đổi thông tin quyền truy cập 34](#_Toc184556141)

[Bảng 2 - 22: Bảng đặc tả usecase xóa quyền truy cập 36](#_Toc184556142)

[Bảng 2 - 23: Bảng đặc tả usecase tạo lịch học 36](#_Toc184556143)

[Bảng 2 - 24: Bảng đặc tả usecase thay đổi thông tin lịch học 37](#_Toc184556144)

**Chương 4**

[Bảng 4 - 1: Thư viện và công cụ sử dụng 69](#_Toc184556145)

[Bảng 4 - 2: Bảng testcase 71](#_Toc184556146)

[Bảng 4 - 3: Triển khai 73](#_Toc184556147)

# DANH MỤC CÁC TỪ VIẾT TẮT

|  |  |  |
| --- | --- | --- |
| STT | Ký hiệu | Ý nghĩa |
| 1 | UI | User interface  (Giao diện người dùng) |
| 2 | SEO | Search Engine Optimization  (Tối ưu hóa tìm kiếm) |
| 3 | API Resful | Representational State Transfer API |
| 4 | JSON | JavaScript Object Notation |
| 5 | LMS | Laboratory Management System |
| 6 | TCP |  |

# PHẦN MỞ ĐẦU

Trong bối cảnh giáo dục ngày càng đóng vai trò quan trọng trong sự phát triển của xã hội, các cơ sở đào tạo không ngừng nâng cao chất lượng giảng dạy và quản lý nhằm đáp ứng nhu cầu của sinh viên và giảng viên. Đặc biệt, tại Học viện Công nghệ Bưu chính Viễn thông (PTIT), các phòng thí nghiệm thực hành đóng vai trò then chốt trong việc hỗ trợ học tập và nghiên cứu. Tuy nhiên, thực trạng quản lý phòng thí nghiệm hiện nay còn tồn tại nhiều vấn đề như khó khăn trong việc quản lý tài nguyên, sắp xếp lịch sử dụng, và theo dõi trạng thái hoạt động của các phòng thí nghiệm. Những hạn chế này không chỉ gây ra lãng phí tài nguyên mà còn làm giảm hiệu quả học tập và nghiên cứu.

Với quy mô đào tạo ngày càng mở rộng, số lượng sinh viên và giảng viên tăng lên theo từng năm, nhu cầu quản lý hiệu quả phòng thí nghiệm trở nên cấp thiết hơn bao giờ hết. Các phương pháp quản lý truyền thống không còn đáp ứng được yêu cầu về tính đồng bộ, minh bạch và khả năng tối ưu hóa quy trình. Việc áp dụng một hệ thống quản lý hiện đại, tự động và toàn diện sẽ giúp giải quyết những vấn đề này, từ đó nâng cao chất lượng học tập và nghiên cứu tại PTIT.

Xuất phát từ nhu cầu thực tiễn này, đề tài “Hệ thống quản lý phòng thí nghiệm thực hành” đã được lựa chọn với mục tiêu xây dựng một giải pháp quản lý tiên tiến, đáp ứng nhu cầu hiện đại hóa của nhà trường. Hệ thống không chỉ hỗ trợ tối ưu hóa quy trình quản lý mà còn mang lại sự thuận tiện và minh bạch cho người dùng, bao gồm cả giảng viên lẫn sinh viên.

Đề tài không chỉ mang ý nghĩa thực tiễn trong việc nâng cao chất lượng quản lý tại PTIT, mà còn góp phần vào việc thúc đẩy chuyển đổi số trong giáo dục, tạo ra một mô hình quản lý hiệu quả và bền vững cho các cơ sở đào tạo trong tương lai.

# 

# CHƯƠNG 1: GIỚI THIỆU ĐỀ TÀI

## 1.1. Đặt vấn đề

Trong thời đại chuyển đổi số đang diễn ra mạnh mẽ, các cơ sở giáo dục cần không ngừng đổi mới để nâng cao chất lượng quản lý và đào tạo. Tại Học viện Công nghệ Bưu chính Viễn thông (PTIT), việc quản lý phòng thí nghiệm thực hành đóng vai trò then chốt, giúp sinh viên và giảng viên tiếp cận hiệu quả các thiết bị và tài nguyên phục vụ học tập, nghiên cứu. Tuy nhiên, để đáp ứng nhu cầu ngày càng cao về tính hiệu quả, minh bạch và hiện đại, việc xây dựng một hệ thống quản lý tự động, đồng bộ và thông minh là hết sức cần thiết.

Việc phát triển một hệ thống quản lý phòng thí nghiệm thực hành tích hợp toàn diện không chỉ giúp cải thiện hiệu quả quản lý tại Học viện Công nghệ Bưu chính Viễn thông (PTIT) mà còn nâng cao trải nghiệm học tập và nghiên cứu của sinh viên, giảng viên. Hệ thống này không chỉ đáp ứng nhu cầu hiện đại hóa quản lý giáo dục mà còn hỗ trợ PTIT cạnh tranh và phát triển bền vững trong bối cảnh chuyển đổi số mạnh mẽ và yêu cầu ngày càng cao từ người học cũng như xã hội.

## 1.2. Mục tiêu và phạm vi của đề tài

Hiện nay, tại Học viện Công nghệ Bưu chính Viễn thông (PTIT), việc quản lý phòng thí nghiệm thực hành đóng vai trò quan trọng trong hoạt động đào tạo và nghiên cứu. Tuy nhiên, các phương pháp quản lý hiện tại vẫn còn mang tính thủ công, phân tán và thiếu tính đồng bộ, dẫn đến nhiều khó khăn trong việc quản lý tài nguyên, lên lịch sử dụng phòng học, theo dõi trạng thái thiết bị và hỗ trợ các hoạt động học tập.

Một số giải pháp quản lý hiện có như sử dụng bảng tính Excel hoặc các phần mềm độc lập không đáp ứng được nhu cầu quản lý toàn diện và tích hợp của học viện. Những hạn chế này gây ra sự thiếu linh hoạt, mất nhiều thời gian xử lý và giảm hiệu quả vận hành. Việc thiếu cơ chế thống kê, báo cáo và giao tiếp nội bộ cũng ảnh hưởng không nhỏ đến hiệu quả công việc và trải nghiệm người dùng.

Để giải quyết những vấn đề trên, đề tài "Hệ thống quản lý phòng thí nghiệm thực hành tại PTIT" được xây dựng với mục tiêu phát triển một nền tảng quản lý tích hợp và toàn diện. Hệ thống sẽ bao gồm các chức năng chính như quản lý phòng, thiết bị, lịch trình sử dụng, người dùng, phân quyền, và báo cáo. Đồng thời, hệ thống tích hợp tính năng nhắn tin thời gian thực, cơ chế thông báo tự động và cung cấp giao diện trực quan, thân thiện để nâng cao trải nghiệm người dùng.

Mục tiêu là xây dựng hệ thống quản lý phòng thí nghiệm thực hành. Các mục tiêu cụ thể của đề tài bao gồm:

* Xây dựng nền tảng quản lý phòng thí nghiệm tích hợp: Cho phép quản lý thông tin chi tiết về phòng học, trạng thái thiết bị và lịch sử sử dụng.
* Phát triển công cụ hỗ trợ lịch trình: Đảm bảo việc lên lịch, quản lý và đăng ký sử dụng phòng học hiệu quả.
* Hỗ trợ phân quyền và quản lý vai trò: Tích hợp quyền truy cập phù hợp cho từng đối tượng người dùng như quản trị viên, giảng viên, nhân viên trực ban và giám đốc.
* Cung cấp chức năng báo cáo và thống kê: Đưa ra các phân tích chi tiết và trực quan về tình trạng phòng, thiết bị, và hoạt động sử dụng.
* Tích hợp nhắn tin thời gian thực: Tạo kênh giao tiếp nhanh chóng, hỗ trợ trao đổi thông tin giữa các vai trò trong hệ thống.

## 1.3. Định hướng giải pháp

Để giải quyết các vấn đề trong việc quản lý phòng thí nghiệm thực hành tại Học viện Công nghệ Bưu chính Viễn thông (PTIT), hệ thống được thiết kế dưới dạng một ứng dụng web tích hợp, sử dụng các công nghệ hiện đại nhằm đảm bảo tính hiệu quả, linh hoạt và mở rộng trong quá trình phát triển.

Các công nghệ và phương pháp được lựa chọn bao gồm:

* **Next.js**: Next.js được sử dụng để xây dựng giao diện người dùng (UI) nhờ khả năng hỗ trợ kết xuất phía máy chủ (Server-Side Rendering - SSR), giúp tăng tốc độ tải trang và tối ưu hóa SEO. Ngoài ra, Next.js cung cấp các tính năng tích hợp sẵn như phân trang, điều hướng động và tạo các thành phần giao diện tái sử dụng, giúp tăng tính linh hoạt và hiệu quả trong phát triển giao diện.
* **Gin Framework**: Gin là một framework nhẹ và hiệu quả dành cho Go (Golang), được sử dụng để phát triển máy chủ backend. Gin hỗ trợ tạo các API RESTful với hiệu năng cao, giúp xử lý nhanh chóng các yêu cầu từ phía người dùng và tích hợp dễ dàng với các thành phần khác của hệ thống.
* **WebSocket**: Tính năng nhắn tin thời gian thực giữa các vai trò trong hệ thống, chẳng hạn như quản trị viên, giảng viên, nhân viên trực ban và giám đốc, được triển khai bằng WebSocket. Điều này giúp cải thiện khả năng giao tiếp, hỗ trợ tức thì và nâng cao trải nghiệm người dùng.
* **PostgreSQL**: PostgreSQL được sử dụng làm hệ quản trị cơ sở dữ liệu chính, nhờ khả năng quản lý cơ sở dữ liệu quan hệ mạnh mẽ và hỗ trợ các tính năng nâng cao như JSON, các truy vấn phức tạp và đảm bảo tính toàn vẹn dữ liệu. Điều này giúp hệ thống dễ dàng xử lý các yêu cầu truy xuất dữ liệu nhanh chóng và chính xác.

# CHƯƠNG 2: TỔNG QUAN

Trong chương này, chúng ta sẽ nghiên cứu quá trình khảo sát và phân tích yêu cầu đối với hệ thống quản lý phòng thực hành. Chương này sẽ bắt đầu bằng việc phân tích thực trạng quản lý phòng thực hành hiện tại và nhu cầu tối ưu hóa quy trình làm việc trong các cơ sở giáo dục. Tiếp theo, các yêu cầu của hệ thống sẽ được mô tả chi tiết thông qua biểu đồ Use Case, đảm bảo mô hình hóa chính xác các chức năng cốt lõi và các tương tác giữa người dùng với hệ thống.

## 2.1. Khảo sát hiện trạng

Trong bối cảnh giáo dục ngày càng chú trọng vào thực hành và nghiên cứu, phòng thí nghiệm thực hành đóng vai trò quan trọng trong việc nâng cao chất lượng đào tạo. Tuy nhiên, hiện trạng quản lý phòng thí nghiệm thực hành tại nhiều trường đại học, đặc biệt ở Việt Nam, vẫn còn gặp nhiều bất cập và hạn chế.

Hiện nay, phòng thí nghiệm thực hành tại học viện vẫn được quản lý theo phương thức thủ công. Hồ sơ sử dụng phòng, danh sách thiết bị, hay lịch đăng ký thường được lưu trữ dưới dạng giấy tờ hoặc các file bảng tính như Excel. Điều này dẫn đến việc xử lý thông tin chậm trễ, dễ xảy ra sai sót trong việc cập nhật dữ liệu, cũng như khó khăn trong việc tra cứu lịch sử sử dụng. Ví dụ, các lớp học hoặc nhóm nghiên cứu có thể vô tình đặt lịch trùng nhau vì thiếu sự đồng bộ hóa dữ liệu, gây ra xung đột và ảnh hưởng đến hiệu quả sử dụng phòng.

Ngoài ra, việc quản lý tài sản và thiết bị cũng gặp nhiều thách thức. Tình trạng thiết bị thường chỉ được kiểm tra định kỳ hoặc khi xảy ra sự cố. Điều này khiến các thiết bị hỏng hóc không được sửa chữa hoặc thay thế kịp thời, làm gián đoạn quá trình học tập và nghiên cứu của sinh viên và giảng viên. Hơn nữa, việc kiểm kê tài sản thường xuyên đòi hỏi nhiều thời gian và công sức, dẫn đến việc quản lý tài nguyên chưa thực sự hiệu quả.

Khả năng giao tiếp và liên lạc giữa các bên liên quan trong phòng thí nghiệm cũng là một vấn đề. Giảng viên, và nhân viên trực ban thường gặp khó khăn trong việc liên hệ khi phát sinh các vấn đề khẩn cấp, chẳng hạn như cần hỗ trợ kỹ thuật hoặc giải quyết xung đột lịch. Hiện trạng này khiến quy trình làm việc trở nên kém linh hoạt và thiếu tính chủ động.

Mặc dù học viện đã phát triển các hệ thống quản lý nội bộ để giải quyết các vấn đề này, nhưng những hệ thống này thường có hạn chế về tính mở rộng, thiếu khả năng đáp ứng đầy đủ các yêu cầu đặc thù, và không tích hợp tốt với các công nghệ hiện đại.

Trên thị trường cũng đã có những hệ thống, ứng dụng lớn như:

### 2.1.1. Labster

A blue text on a white background

Description automatically generated

Hình 2 - 1: Labster

Labster là một nền tảng học tập trực tuyến chuyên cung cấp các mô phỏng thí nghiệm ảo cho sinh viên. Mặc dù Labster không phải là hệ thống quản lý phòng thí nghiệm truyền thống, nhưng nó cung cấp một môi trường học tập và thực hành thí nghiệm trong một không gian ảo.

Ưu điểm: Labster cho phép sinh viên thực hiện các thí nghiệm trong môi trường ảo, giúp giảm thiểu chi phí và thời gian thực hiện thí nghiệm trong phòng thí nghiệm thực tế. Nền tảng này đặc biệt hữu ích trong các môn học khoa học, sinh học, hóa học, và vật lý.

Nhược điểm: Không hỗ trợ trực tiếp quản lý phòng thí nghiệm vật lý và thiết bị thực tế.

### 2.1.2. LMS của LabWorks

A black background with white letters

Description automatically generated

Hình 2 - 2: LabWorks

LabWorks là một hệ thống quản lý phòng thí nghiệm chuyên nghiệp được thiết kế để quản lý tất cả các hoạt động trong phòng thí nghiệm, bao gồm quản lý thiết bị, lịch sử sử dụng, bảo trì thiết bị, và quản lý người dùng.

Ưu điểm: Hệ thống này hỗ trợ người quản lý theo dõi và kiểm soát tài nguyên phòng thí nghiệm dễ dàng, giảm thiểu sự cố và tăng hiệu quả sử dụng tài sản. LabWorks cũng tích hợp các tính năng như lập lịch sử dụng phòng và thiết bị, giúp tránh xung đột và tối ưu hóa lịch trình.

Nhược điểm: Phần mềm có chi phí triển khai khá cao và cần một đội ngũ quản lý để duy trì hệ thống.

### 2.1.3. Simplelab

A blue and black text

Description automatically generated

Hình 2 - 3: Simplelab

SimpleLab là một phần mềm quản lý phòng thí nghiệm dành cho các trường học và tổ chức giáo dục. Nó cung cấp các tính năng như quản lý tài nguyên, theo dõi lịch sử sử dụng, bảo trì thiết bị, và tạo báo cáo cho các thí nghiệm.

Ưu điểm: Hệ thống này thân thiện với người dùng và dễ sử dụng, giúp các giảng viên và sinh viên dễ dàng quản lý và sử dụng tài nguyên phòng thí nghiệm. SimpleLab cũng hỗ trợ các tính năng nhắc nhở về việc bảo trì thiết bị.

Nhược điểm: Phần mềm này thiếu các tính năng nâng cao như tích hợp với các thiết bị tự động hoặc phần mềm quản lý dữ liệu thí nghiệm phức tạp.

Dựa trên những khảo sát thực tế, hệ thống quản lý phòng thí nghiệm thực hành sẽ hướng đến việc phát triển một ứng dụng hỗ trợ người dùng đặt lịch sử dụng phòng thí nghiệm, và quản lý các tài nguyên thiết bị trong phòng. Ứng dụng này không chỉ cung cấp khả năng quản lý tài nguyên một cách hiệu quả mà còn tích hợp các tiện ích nhằm nâng cao trải nghiệm của người dùng, như báo cáo tình trạng thiết bị, và hỗ trợ giao tiếp giữa giảng viên, và nhân viên phòng thí nghiệm.

## 2.2. Định hướng và yêu cầu ứng dụng

Hệ thống cần đảm bảo cung cấp các chức năng chính phục vụ nhu cầu quản lý và vận hành phòng thực hành một cách toàn diện. Trước hết, hệ thống phải hỗ trợ quản lý người dùng và vai trò, cho phép phân quyền linh hoạt để đảm bảo rằng từng vai trò như quản trị viên, giảng viên, nhân viên trực ban, và giám đốc chỉ truy cập được các chức năng phù hợp. Người dùng có thể đăng nhập và thực hiện các tác vụ dựa trên quyền hạn của mình.

Ngoài ra, hệ thống cần có khả năng quản lý chi tiết thông tin về phòng thực hành. Tính năng này bao gồm việc thêm, sửa, xóa, và theo dõi tình trạng sử dụng của từng phòng, đồng thời cung cấp thông tin về các thiết bị có trong phòng. Hệ thống cũng phải tích hợp khả năng hiển thị trạng thái thời gian thực của phòng, chẳng hạn như phòng đang sử dụng, phòng trống, hay phòng cần bảo trì.

Để đảm bảo các thiết bị trong phòng được quản lý chặt chẽ, hệ thống cần hỗ trợ lưu trữ thông tin về tình trạng, ngày bảo trì, và lịch sử sửa chữa của từng thiết bị.

Về quản lý lịch trình, hệ thống cần hỗ trợ người dùng tạo, sửa đổi, và theo dõi lịch sử dụng phòng thực hành một cách dễ dàng. Giảng viên và quản trị viên có thể đặt lịch, quản lý lịch. Điều này giúp việc sắp xếp và theo dõi lịch trình trở nên thuận tiện và hiệu quả hơn.

Một phần không thể thiếu là tính năng báo cáo và thống kê, giúp quản trị viên và giám đốc theo dõi hiệu quả sử dụng tài nguyên. Hệ thống cần tạo được các báo cáo chi tiết về lịch sử sử dụng phòng, tình trạng thiết bị, và các chỉ số quan trọng để hỗ trợ ra quyết định chiến lược.

Hơn nữa, hệ thống cần cung cấp khả năng nhắn tin thời gian thực thông qua WebSocket. Điều này cho phép người dùng dễ dàng trao đổi thông tin, nâng cao hiệu quả giao tiếp và hỗ trợ công việc nhanh chóng.

Bên cạnh các chức năng chính, hệ thống cũng cần đáp ứng các yêu cầu phi chức năng. Hiệu năng của hệ thống phải đảm bảo khả năng xử lý đồng thời nhiều yêu cầu mà không làm giảm tốc độ phản hồi. Đồng thời, hệ thống cần được thiết kế linh hoạt để dễ dàng mở rộng khi số lượng người dùng tăng lên.

Yêu cầu bảo mật cũng rất quan trọng, bao gồm việc xác thực người dùng bằng JWT, mã hóa dữ liệu nhạy cảm, và kiểm soát quyền truy cập chặt chẽ. Tính khả dụng của hệ thống cần được đảm bảo với thời gian hoạt động ổn định và cơ chế sao lưu dữ liệu để khắc phục sự cố nhanh chóng.

Ngoài ra, giao diện của hệ thống phải thân thiện, dễ sử dụng, và tương thích với nhiều loại thiết bị, bao gồm cả trình duyệt web và thiết bị di động.

## 2.3. Tổng quan chức năng.

### 2.3.1. Biểu đồ usecase tổng quát

A diagram of a diagram

Description automatically generated

Hình 2 - 4: Biểu đồ usecase tổng quát

Hệ thống quản lý phòng thí nghiệm thực hành gồm có các tác nhân chính tham gia: quản trị viên (admin), trực ban (trực ban), giảng viên (giangvien), giám đốc học viện (giamdoc) cụ thể:

* Admin: Là người có quyền cao nhất trong hệ thống, Admin chịu trách nhiệm toàn bộ các hoạt động quản lý như: tài khoản người dùng, phòng học, thiết bị, lịch đặt phòng, vai trò, quyền truy cập, lịch làm việc, báo cáo, và danh mục thiết bị. Mọi tác vụ quan trọng liên quan đến việc cấu hình hoặc điều chỉnh dữ liệu trong hệ thống đều thuộc quyền của Admin.
* Giám đốc học viện: Giám đốc có quyền truy cập đầy đủ vào hệ thống, chủ yếu để theo dõi và giám sát tình hình hoạt động. Vai trò này tập trung vào việc xem xét thông tin hơn là can thiệp trực tiếp vào các quy trình quản lý chi tiết.
* Trực ban: Với vai trò hỗ trợ vận hành hàng ngày, trực ban có thể truy cập danh sách phòng học, thiết bị, người dùng và báo cáo để đảm bảo hoạt động suôn sẻ. Họ được phép điều chỉnh các lịch đặt phòng nhằm đáp ứng nhu cầu thực tế. Các hoạt động thiết lập và thay đổi thông tin quan trọng như phòng học, thiết bị hoặc tài khoản người dùng sẽ được Admin đảm nhiệm.
* Giảng viên: Giảng viên được cung cấp quyền quản lý lịch làm việc và báo cáo liên quan về tình trạng phòng học hoặc thiết bị khi phát hiện các vấn đề như hỏng hóc, hư hại, đảm bảo tính linh hoạt và chủ động trong việc tổ chức các hoạt động thực hành. Những tác vụ quản lý tổng quát khác sẽ nằm ngoài quyền hạn của vai trò này.

### 2.3.2. Biểu đồ usecase phân rã quản lý tài khoản

A diagram of a person's diagram

Description automatically generated

Hình 2 - 5: Biểu đồ usecase quản lý tài khoản

Admin:

* Có quyền tạo mới tài khoản và chỉnh sửa thông tin liên quan như họ tên, email, số điện thoại, ảnh đại diện, tiểu sử, và các thông tin liên kết mạng xã hội.
* Được phép xóa tài khoản khi cần thiết, đảm bảo tính quản lý và bảo mật hệ thống.

Admin và Giám đốc học viện:

* Được phép tìm kiếm tài khoản bằng họ tên để dễ dàng quản lý thông tin người dùng.
* Có thể liệt kê danh sách tài khoản toàn hệ thống để tiện theo dõi và thực hiện các thao tác quản trị.

### 2.3.3. Biểu đồ usecase phân rã quản lý phòng học

A diagram of a diagram

Description automatically generated

Hình 2 - 6: Biểu đồ usecase quản lý phòng

Admin:

* Có quyền tạo mới, chỉnh sửa thông tin và xóa phòng học để đảm bảo dữ liệu luôn chính xác và phù hợp với nhu cầu sử dụng.
* Những thao tác này bao gồm cập nhật thông tin chi tiết phòng như tên, trạng thái phòng

Admin và Giám đốc học viện:

* Có thể liệt kê danh sách tất cả các phòng học hiện có trong hệ thống, hỗ trợ theo dõi và đánh giá hiệu quả sử dụng.
* Được quyền tìm kiếm phòng học dựa trên tên phòng để thuận tiện trong việc tra cứu và quản lý.
* Có khả năng thống kê phòng học dựa trên số lượng báo cáo liên quan và tần suất sử dụng, cung cấp thông tin hữu ích để đưa ra các quyết định quản trị phù hợp.

### 2.3.4. Biểu đồ usecase phân rã quản lý thiết bị

A diagram of a diagram

Description automatically generated

Hình 2 - 7: Biểu đồ usecase quản lý thiết bị

Admin:

* Có quyền thêm mới thiết bị vào hệ thống, chỉnh sửa thông tin thiết bị, hoặc xóa thiết bị khi không còn sử dụng, đảm bảo cơ sở dữ liệu luôn được cập nhật và phù hợp với thực tế.
* Những thao tác này giúp quản lý thông tin thiết bị một cách chi tiết, bao gồm tên thiết bị, trạng thái, loại thiết bị, và phòng học mà thiết bị được sử dụng.

Admin và Giám đốc học viện:

* Có thể xem danh sách toàn bộ thiết bị hiện có trong hệ thống, hỗ trợ việc theo dõi và đánh giá tình hình thiết bị của từng phòng học.
* Được quyền tìm kiếm thiết bị dựa trên phòng học để dễ dàng xác định các tài nguyên liên quan.
* Có khả năng thống kê thiết bị theo trạng thái như hoạt động, đang sửa chữa, hoặc không sử dụng, giúp quản lý tài nguyên hiệu quả hơn.

### 2.3.5. Biểu đồ usecase phân rã quản lý đặt lịch

A diagram of a diagram

Description automatically generated

Hình 2 - 8: Biểu đồ usecase quản lý đặt lịch

Admin, Trực ban, Giảng viên:

* Có quyền tạo mới, chỉnh sửa hoặc hủy lịch học trong hệ thống.
* Những thao tác này cho phép linh hoạt điều chỉnh kế hoạch học tập để phù hợp với nhu cầu thực tế của từng phòng học và giảng viên.

Admin, Trực ban, Giảng viên, Giám đốc học viện:

* Có thể tìm kiếm lịch học dựa trên thông tin người giảng dạy hoặc phòng học, giúp nhanh chóng xác định thông tin liên quan đến các buổi học.
* Được phép liệt kê danh sách các lịch học hiện có, phục vụ cho mục đích giám sát, thống kê, hoặc quản lý chung về việc sử dụng tài nguyên phòng học và giảng viên.

### 2.3.6. Biểu đồ usecase phân rã quản lý vai trò

A diagram of a person with text

Description automatically generated

Hình 2 - 9: Biểu đồ usecase quản lý vai trò

Admin:

* Có quyền thiết lập vai trò mới, chỉnh sửa thông tin vai trò hiện có, hoặc loại bỏ vai trò không còn cần thiết.
* Việc này cho phép Admin định nghĩa và quản lý quyền hạn của từng vai trò trong hệ thống một cách hiệu quả, đảm bảo tuân thủ theo các yêu cầu hoạt động của tổ chức.

Admin và Giám đốc học viện:

* Có thể tìm kiếm vai trò thông qua tên vai trò, giúp nhanh chóng xác định và kiểm tra thông tin liên quan đến các quyền hạn được thiết lập cho từng vai trò.

### 2.3.7. Biểu đồ usecase phân rã quản lý báo cáo

A diagram of a network

Description automatically generated

Hình 2 - 10: Biểu đồ usecase quản lý báo cáo

Admin, Trực ban, Giảng viên:

* Có thể tạo mới, chỉnh sửa hoặc xóa các báo cáo liên quan đến tình trạng phòng học, thiết bị hỏng hóc, hoặc các vấn đề phát sinh trong quá trình sử dụng.
* Đồng thời, họ có thể tìm kiếm báo cáo dựa trên phòng học để nhanh chóng xác định và xử lý các vấn đề được báo cáo.
* Hệ thống cũng hỗ trợ hiển thị danh sách báo cáo giúp quản lý thuận tiện và theo dõi chi tiết từng vấn đề.

Admin, Trực ban, Giám đốc học viện:

* Có quyền thống kê báo cáo theo từng phòng học để nắm bắt tần suất các sự cố xảy ra. Thống kê này giúp đưa ra các quyết định cải thiện cơ sở vật chất hoặc điều chỉnh lịch trình sử dụng phòng hợp lý.

### 2.3.8. Biểu đồ usecase phân rã quản lý loại thiết bị

A diagram of a person with text

Description automatically generated

Hình 2 - 11: Biểu đồ usecase quản lý loại thiết bị

Admin:

* Được phép tạo mới, chỉnh sửa, hoặc xóa các loại thiết bị trong hệ thống. Điều này giúp admin duy trì và cập nhật danh sách các loại thiết bị phù hợp với nhu cầu sử dụng thực tế tại các phòng thí nghiệm.

Admin và Giám đốc học viện:

* Có thể tìm kiếm loại thiết bị dựa trên tên, cho phép truy xuất thông tin nhanh chóng và hỗ trợ việc quản lý, lập kế hoạch hoặc ra quyết định liên quan đến việc mua sắm và sử dụng thiết bị.

### 2.3.9. Biểu đồ usecase phân rã quản lý quyền truy cập

A diagram of a person's relationship

Description automatically generated

Hình 2 - 12: Biểu đồ usecase quản lý quyền truy cập

Admin:

* Có thể tạo mới, chỉnh sửa và xóa quyền truy cập trong hệ thống. Chức năng này giúp admin kiểm soát chi tiết các mức quyền của người dùng, đảm bảo tính bảo mật và quản lý hệ thống một cách hiệu quả.

Admin và Giám đốc học viện:

* Được phép tìm kiếm quyền truy cập dựa trên tên, giúp dễ dàng xác định và kiểm tra các quyền hiện có trong hệ thống để hỗ trợ công tác giám sát và quản lý.

## 2.4. Đặc tả chức năng

### 2.4.1. Đăng nhập

Bảng 2 - 1: Bảng đặc tả usecase đăng nhập

|  |  |  |  |
| --- | --- | --- | --- |
| Mã usecase | UC001 | Tên usecase | Đăng nhập |
| Tác nhân | admin, trucban, giamdoc, giangvien | | |
| Mô tả | Đăng nhập tài khoản với vai trò là trực ban, giảng viên, giám đốc | | |
| Tiền điều kiện | Tác nhân đã có tài khoản trong hệ thống | | |
| Luồng sự kiện chính | STT | Thực hiện | Hành động |
| 1 | Hệ thống | Truy cập vào hệ thống |
| 2 | Hệ thống | Hiển thị form đăng nhập bao gồm input username, password, button đăng nhập |
| 3 | admin, giamdoc, trucban, giangvien | Nhập thông tin username, password |
| 4 | admin, giamdoc, trucban, giangvien | Nhấn đăng nhập |
| 5 | Hệ thống | Kiểm tra thông tin |
| 6 | Hệ thống | Chuyển hướng sang màn hình trang chủ |
| Luồng sự kiện thay thế | STT | Thực hiện | Hành động |
| 3a | Hệ thống | Thông tin không đúng yêu cầu |
| 5a | Hệ thống | Thông báo thông tin không tồn tại trong hệ thống |
| Hậu điều kiện | Không | | |

### 2.4.2. Đăng ký

Bảng 2 - 2: Bảng đặc tả usecase tạo tài khoản

|  |  |  |  |
| --- | --- | --- | --- |
| Mã usecase | UC002 | Tên usecase | Tạo tài khoản |
| Tác nhân | Admin | | |
| Mô tả | Tạo tài khoản với vai trò là trực ban, giảng viên, giám đốc | | |
| Tiền điều kiện | Admin phải đăng nhập thành công vào hệ thống | | |
| Luồng sự kiện chính | STT | Thực hiện | Hành động |
| 1 | Admin | Nhấn tài khoản |
| 2 | Hệ thống | Hiển thị danh sách tài khoản đã tồn tại trong hệ thống, nút tạo tài khoản |
| 3 | Admin | Nhấn nút tạo tài khoản |
| 4 | Hệ thống | Hiển thị form nhập thông tin tài khoản |
| 5 | Admin | Chọn vai trò |
| 6 | Admin | Nhấn tạo tài khoản |
| 7 | Hệ thống | Kiểm tra thông tin |
| 8 | Hệ thống | Thông báo tạo tài khoản thành công và chuyển hướng sang màn hình hiển thị danh sách tài khoản |
| Luồng sự kiện thay thế | STT | Thực hiện | Hành động |
| 7a | Hệ thống | Thông tin không đúng yêu cầu |
| 8a | Hệ thống | Thông báo thông tin đã tồn tại trong hệ thống |
| Hậu điều kiện | Không | | |

### 2.4.3. Thay đổi thông tin tài khoản

Bảng 2 - 3: Bảng đặc tả usecase thay đổi thông tin tài khoản

|  |  |  |  |
| --- | --- | --- | --- |
| Mã usecase | UC003 | Tên usecase | Thay đổi thông tin tài khoản |
| Tác nhân | Admin | | |
| Mô tả | Thay đổi thông tin tài khoản | | |
| Tiền điều kiện | Admin phải đăng nhập thành công vào hệ thống | | |
| Luồng sự kiện chính | STT | Thực hiện | Hành động |
| 1 | Admin | Nhấn tài khoản |
| 2 | Hệ thống | Hiển thị danh sách tài khoản đã tồn tại trong hệ thống, cột action bao gồm sửa và xóa ở mỗi tài khoản, button tạo tài khoản |
| 3 | Admin | Nhấn sửa ở cột action của tài khoản cần sửa |
| 4 | Hệ thống | Hiển thị form  thông tin tài khoản |
| 5 | Admin | Sửa đổi thông tin tài khoản |
| 6 | Admin | Nhấn sửa tài khoản |
| 7 | Hệ thống | Kiểm tra thông tin |
| 8 | Hệ thống | Thông báo sửa thông tin tài khoản thành công và chuyển hướng sang màn hình hiển thị danh sách tài khoản |
| Luồng sự kiện thay thế | STT | Thực hiện | Hành động |
| 7a | Hệ thống | Thông tin không đúng yêu cầu |
| 8a | Hệ thống | Thông báo thông tin đã tồn tại trong hệ thống |
| Hậu điều kiện | Không | | |

### 2.4.4. Xóa tài khoản

Bảng 2 - 4: Bảng đặc tả usecase xóa tài khoản

|  |  |  |  |
| --- | --- | --- | --- |
| Mã usecase | UC004 | Tên usecase | Xóa tài khoản |
| Tác nhân | Admin | | |
| Mô tả | Xóa tài khoản | | |
| Tiền điều kiện | Admin phải đăng nhập thành công vào hệ thống | | |
| Luồng sự kiện chính | STT | Thực hiện | Hành động |
| 1 | Admin | Nhấn tài khoản |
| 2 | Hệ thống | Hiển thị danh sách tài khoản đã tồn tại trong hệ thống, cột action bao gồm sửa và xóa ở mỗi tài khoản, button tạo tài khoản |
| 3 | Admin | Nhấn xóa ở cột action của tài khoản cần sửa |
| 4 | Hệ thống | Thực hiện xóa tài khoản |
| 5 | Hệ thống | Thông báo xóa tài khoản thành công và chuyển hướng sang màn hình hiển thị danh sách tài khoản |
| Luồng sự kiện thay thế | Không | | |
| Hậu điều kiện | Không | | |

### 2.4.5. Tạo phòng

Bảng 2 - 5: Bảng đặc tả usecase tạo phòng

|  |  |  |  |
| --- | --- | --- | --- |
| Mã usecase | UC005 | Tên usecase | Tạo phòng |
| Tác nhân | Admin | | |
| Mô tả | Tạo phòng học | | |
| Tiền điều kiện | Admin phải đăng nhập thành công vào hệ thống | | |
| Luồng sự kiện chính | STT | Thực hiện | Hành động |
| 1 | Admin | Nhấn phòng học |
| 2 | Hệ thống | Hiển thị danh sách phòng đã tồn tại trong hệ thống, nút tạo phòng |
| 3 | Admin | Nhấn nút tạo phòng |
| 4 | Hệ thống | Hiển thị form nhập thông tin phòng |
| 5 | Admin | Nhấn tạo phòng |
| 6 | Hệ thống | Kiểm tra thông tin |
| 7 | Hệ thống | Thông báo tạo phòng thành công và chuyển hướng sang màn hình hiển thị danh sách phòng |
| Luồng sự kiện thay thế | STT | Thực hiện | Hành động |
| 6a | Hệ thống | Thông tin không đúng yêu cầu |
| Hậu điều kiện | Không | | |

### 2.4.6. Thay đổi thông tin phòng

Bảng 2 - 6: Bảng đặc tả usecase thay đổi thông tin phòng

|  |  |  |  |
| --- | --- | --- | --- |
| Mã usecase | UC006 | Tên usecase | Thay đổi thông tin phòng |
| Tác nhân | Admin | | |
| Mô tả | Thay đổi thông tin phòng | | |
| Tiền điều kiện | Admin phải đăng nhập thành công vào hệ thống | | |
| Luồng sự kiện chính | STT | Thực hiện | Hành động |
| 1 | Admin | Nhấn phòng học |
| 2 | Hệ thống | Hiển thị danh sách phòng đã tồn tại trong hệ thống, cột action bao gồm sửa và xóa ở mỗi tài khoản, button tạo phòng |
| 3 | Admin | Nhấn sửa ở cột action của phòng cần sửa |
| 4 | Hệ thống | Hiển thị form  thông tin phòng |
| 5 | Admin | Sửa đổi thông tin phòng |
| 6 | Admin | Nhấn sửa phòng |
| 7 | Hệ thống | Kiểm tra thông tin |
| 8 | Hệ thống | Thông báo sửa thông tin phòng thành công và chuyển hướng sang màn hình hiển thị danh sách phòng |
| Luồng sự kiện thay thế | STT | Thực hiện | Hành động |
| 7a | Hệ thống | Thông tin không đúng yêu cầu |
| 8a | Hệ thống | Thông báo thông tin đã tồn tại trong hệ thống |
| Hậu điều kiện | Không | | |

### 2.4.7. Xóa phòng

Bảng 2 - 7: Bảng đặc tả usecase xóa phòng

|  |  |  |  |
| --- | --- | --- | --- |
| Mã usecase | UC007 | Tên usecase | Xóa phòng |
| Tác nhân | Admin | | |
| Mô tả | Xóa phòng | | |
| Tiền điều kiện | Admin phải đăng nhập thành công vào hệ thống | | |
| Luồng sự kiện chính | STT | Thực hiện | Hành động |
| 1 | Admin | Nhấn phòng học |
| 2 | Hệ thống | Hiển thị danh sách phòng đã tồn tại trong hệ thống, cột action bao gồm sửa và xóa ở mỗi phòng, button tạo phòng |
| 3 | Admin | Nhấn xóa ở cột action của phòng cần xóa |
| 4 | Hệ thống | Thực hiện xóa phòng |
| 5 | Hệ thống | Thông báo xóa phòng thành công và chuyển hướng sang màn hình hiển thị danh sách phòng |
| Luồng sự kiện thay thế | Không | | |
| Hậu điều kiện | Không | | |

### 2.4.8. Tạo thiết bị

Bảng 2 - 8: Bảng đặc tả usecase tạo thiết bị

|  |  |  |  |
| --- | --- | --- | --- |
| Mã usecase | UC008 | Tên usecase | Tạo thiết bị |
| Tác nhân | Admin | | |
| Mô tả | Tạo thiết bị | | |
| Tiền điều kiện | Admin phải đăng nhập thành công vào hệ thống | | |
| Luồng sự kiện chính | STT | Thực hiện | Hành động |
| 1 | Admin | Nhấn thiết bị |
| 2 | Hệ thống | Hiển thị danh sách thiết bị đã tồn tại trong hệ thống, nút tạo thiết bị |
| 3 | Admin | Nhấn nút tạo thiết bị |
| 4 | Hệ thống | Hiển thị form nhập thông tin thiết bị |
| 5 | Admin | Nhấn tạo thiết bị |
| 6 | Hệ thống | Kiểm tra thông tin |
| 7 | Hệ thống | Thông báo tạo thiết bị thành công và chuyển hướng sang màn hình hiển thị danh sách thiết bị |
| Luồng sự kiện thay thế | STT | Thực hiện | Hành động |
| 6a | Hệ thống | Thông tin không đúng yêu cầu |
| Hậu điều kiện | Không | | |

### 2.4.9. Thay đổi thông tin thiết bị

Bảng 2 - 9: Bảng đặc tả usecase thay đổi thông tin thiết bị

|  |  |  |  |
| --- | --- | --- | --- |
| Mã usecase | UC009 | Tên usecase | Thay đổi thông tin thiết bị |
| Tác nhân | Admin | | |
| Mô tả | Thay đổi thông tin thiết bị | | |
| Tiền điều kiện | Admin phải đăng nhập thành công vào hệ thống | | |
| Luồng sự kiện chính | STT | Thực hiện | Hành động |
| 1 | Admin | Nhấn thiết bị |
| 2 | Hệ thống | Hiển thị danh sách thiết bị đã tồn tại trong hệ thống, cột action bao gồm sửa và xóa ở mỗi thiết bị, button tạo thiết bị |
| 3 | Admin | Nhấn sửa ở cột action của thiết bị cần sửa |
| 4 | Hệ thống | Hiển thị form  thông tin thiết bị |
| 5 | Admin | Sửa đổi thông tin thiết bị |
| 6 | Admin | Nhấn sửa thiết bị |
| 7 | Hệ thống | Kiểm tra thông tin |
| 8 | Hệ thống | Thông báo sửa thông tin thiết bị thành công và chuyển hướng sang màn hình hiển thị danh sách thiết bị |
| Luồng sự kiện thay thế | STT | Thực hiện | Hành động |
| 7a | Hệ thống | Thông tin không đúng yêu cầu |
| Hậu điều kiện | Không | | |

### 2.4.10. Xóa thiết bị

Bảng 2 - 10: Bảng đặc tả usecase xóa thiết bị

|  |  |  |  |
| --- | --- | --- | --- |
| Mã usecase | UC010 | Tên usecase | Xóa thiết bị |
| Tác nhân | Admin | | |
| Mô tả | Xóa thiết bị | | |
| Tiền điều kiện | Admin phải đăng nhập thành công vào hệ thống | | |
| Luồng sự kiện chính | STT | Thực hiện | Hành động |
| 1 | Admin | Nhấn thiết bị |
| 2 | Hệ thống | Hiển thị danh sách thiết bị đã tồn tại trong hệ thống, cột action bao gồm sửa và xóa ở mỗi thiết bị, button tạo thiết bị |
| 3 | Admin | Nhấn xóa ở cột action của thiết bị cần xóa |
| 4 | Hệ thống | Thực hiện xóa thiết bị |
| 5 | Hệ thống | Thông báo xóa thiết bị thành công và chuyển hướng sang màn hình hiển thị danh sách thiết bị |
| Luồng sự kiện thay thế | Không | | |
| Hậu điều kiện | Không | | |

### 2.4.11. Tạo vai trò

Bảng 2 - 11: Bảng đặc tả usecase tạo vai trò

|  |  |  |  |
| --- | --- | --- | --- |
| Mã usecase | UC011 | Tên usecase | Tạo vai trò |
| Tác nhân | Admin | | |
| Mô tả | Tạo vai trò | | |
| Tiền điều kiện | Admin phải đăng nhập thành công vào hệ thống | | |
| Luồng sự kiện chính | STT | Thực hiện | Hành động |
| 1 | Admin | Nhấn vai trò |
| 2 | Hệ thống | Hiển thị danh sách vai trò đã tồn tại trong hệ thống, nút tạo vai trò |
| 3 | Admin | Nhấn nút tạo vai trò |
| 4 | Hệ thống | Hiển thị form nhập thông tin vai trò |
| 5 | Admin | Nhấn tạo vai trò |
| 6 | Hệ thống | Kiểm tra thông tin |
| 7 | Hệ thống | Thông báo tạo vai trò thành công và chuyển hướng sang màn hình hiển thị danh sách vai trò |
| Luồng sự kiện thay thế | STT | Thực hiện | Hành động |
| 6a | Hệ thống | Thông tin không đúng yêu cầu |
| Hậu điều kiện | Không | | |

### 2.4.12. Thay đổi thông tin vai trò

Bảng 2 - 12: Bảng đặc tả usecase thay đổi thông tin vai trò

|  |  |  |  |
| --- | --- | --- | --- |
| Mã usecase | UC012 | Tên usecase | Thay đổi thông tin vai trò |
| Tác nhân | Admin | | |
| Mô tả | Thay đổi thông tin vai trò | | |
| Tiền điều kiện | Admin phải đăng nhập thành công vào hệ thống | | |
| Luồng sự kiện chính | STT | Thực hiện | Hành động |
| 1 | Admin | Nhấn vai trò |
| 2 | Hệ thống | Hiển thị danh sách vai trò đã tồn tại trong hệ thống, cột action bao gồm sửa và xóa ở mỗi vai trò, button tạo vai trò |
| 3 | Admin | Nhấn sửa ở cột action của vai trò cần sửa |
| 4 | Hệ thống | Hiển thị form  thông tin vai trò |
| 5 | Admin | Sửa đổi thông tin vai trò |
| 6 | Admin | Nhấn sửa vai trò |
| 7 | Hệ thống | Kiểm tra thông tin |
| 8 | Hệ thống | Thông báo sửa thông tin vai trò thành công và chuyển hướng sang màn hình hiển thị danh sách vai trò |
| Luồng sự kiện thay thế | STT | Thực hiện | Hành động |
| 7a | Hệ thống | Thông tin không đúng yêu cầu |
| 8a | Hệ thống | Thông báo thông tin đã tồn tại trong hệ thống |
| Hậu điều kiện | Không | | |

### 2.4.13. Xóa vai trò

Bảng 2 - 13: Bảng đặc tả usecase xóa vai trò

|  |  |  |  |
| --- | --- | --- | --- |
| Mã usecase | UC013 | Tên usecase | Xóa vai trò |
| Tác nhân | Admin | | |
| Mô tả | Xóa vai trò | | |
| Tiền điều kiện | Admin phải đăng nhập thành công vào hệ thống | | |
| Luồng sự kiện chính | STT | Thực hiện | Hành động |
| 1 | Admin | Nhấn vai trò |
| 2 | Hệ thống | Hiển thị danh sách vai trò đã tồn tại trong hệ thống, cột action bao gồm sửa và xóa ở mỗi vai trò, button tạo vai trò |
| 3 | Admin | Nhấn xóa ở cột action của vai trò cần xóa |
| 4 | Hệ thống | Thực hiện xóa vai trò |
| 5 | Hệ thống | Thông báo xóa vai trò thành công và chuyển hướng sang màn hình hiển thị danh sách vai trò |
| Luồng sự kiện thay thế | Không | | |
| Hậu điều kiện | Không | | |

### 2.4.14. Tạo báo cáo

Bảng 2 - 14: Bảng đặc tả usecase tạo báo cáo

|  |  |  |  |
| --- | --- | --- | --- |
| Mã usecase | UC014 | Tên usecase | Tạo báo cáo |
| Tác nhân | Admin, giangvien | | |
| Mô tả | Tạo báo cáo | | |
| Tiền điều kiện | Admin, giangvien phải đăng nhập thành công vào hệ thống | | |
| Luồng sự kiện chính | STT | Thực hiện | Hành động |
| 1 | Admin | Nhấn báo cáo |
| 2 | Hệ thống | Hiển thị danh sách báo cáo đã tồn tại trong hệ thống, nút tạo báo cáo |
| 3 | Admin | Nhấn nút tạo báo cáo |
| 4 | Hệ thống | Hiển thị form nhập thông tin báo cáo |
| 5 | Admin | Nhấn tạo báo cáo |
| 6 | Hệ thống | Kiểm tra thông tin |
| 7 | Hệ thống | Thông báo tạo báo cáo thành công và chuyển hướng sang màn hình hiển thị danh sách báo cáo |
| Luồng sự kiện thay thế | Không | | |
| Hậu điều kiện | Không | | |

### 2.4.15. Thay đổi thông tin báo cáo

Bảng 2 - 15: Bảng đặc tả usecase thay đổi thông tin báo cáo

|  |  |  |  |
| --- | --- | --- | --- |
| Mã usecase | UC015 | Tên usecase | Thay đổi thông tin báo cáo |
| Tác nhân | Admin, giangvien | | |
| Mô tả | Thay đổi thông tin báo cáo | | |
| Tiền điều kiện | Admin phải đăng nhập thành công vào hệ thống | | |
| Luồng sự kiện chính | STT | Thực hiện | Hành động |
| 1 | Admin | Nhấn báo cáo |
| 2 | Hệ thống | Hiển thị danh sách báo cáo đã tồn tại trong hệ thống, cột action bao gồm sửa và xóa ở mỗi báo cáo, button tạo báo cáo |
| 3 | Admin | Nhấn sửa ở cột action của báo cáo cần sửa |
| 4 | Hệ thống | Hiển thị form  thông tin báo cáo |
| 5 | Admin | Sửa đổi thông tin báo cáo |
| 6 | Admin | Nhấn sửa báo cáo |
| 7 | Hệ thống | Kiểm tra thông tin |
| 8 | Hệ thống | Thông báo sửa thông tin báo cáo thành công và chuyển hướng sang màn hình hiển thị danh sách báo cáo |
| Luồng sự kiện thay thế | STT | Thực hiện | Hành động |
| Không | | |
| Hậu điều kiện | Không | | |

### 2.4.16. Xóa báo cáo

Bảng 2 - 16: Bảng đặc tả usecase xóa báo cáo

|  |  |  |  |
| --- | --- | --- | --- |
| Mã usecase | UC016 | Tên usecase | Xóa báo cáo |
| Tác nhân | Admin, giangvien | | |
| Mô tả | Xóa báo cáo | | |
| Tiền điều kiện | Admin phải đăng nhập thành công vào hệ thống | | |
| Luồng sự kiện chính | STT | Thực hiện | Hành động |
| 1 | Admin | Nhấn báo cáo |
| 2 | Hệ thống | Hiển thị danh sách báo cáo đã tồn tại trong hệ thống, cột action bao gồm sửa và xóa ở mỗi báo cáo, button tạo báo cáo |
| 3 | Admin | Nhấn xóa ở cột action của báo cáo cần xóa |
| 4 | Hệ thống | Thực hiện xóa báo cáo |
| 5 | Hệ thống | Thông báo xóa báo cáo thành công và chuyển hướng sang màn hình hiển thị danh sách báo cáo |
| Luồng sự kiện thay thế | Không | | |
| Hậu điều kiện | Không | | |

### 2.4.17. Tạo loại thiết bị

Bảng 2 - 17: Bảng đặc tả usecase tạo loại thiết bị

|  |  |  |  |
| --- | --- | --- | --- |
| Mã usecase | UC017 | Tên usecase | Tạo loại thiết bị |
| Tác nhân | Admin | | |
| Mô tả | Tạo loại thiết bị | | |
| Tiền điều kiện | Admin phải đăng nhập thành công vào hệ thống | | |
| Luồng sự kiện chính | STT | Thực hiện | Hành động |
| 1 | Admin | Nhấn loại thiết bị |
| 2 | Hệ thống | Hiển thị danh sách loại thiết bị đã tồn tại trong hệ thống, nút tạo loại thiết bị |
| 3 | Admin | Nhấn nút tạo loại thiết bị |
| 4 | Hệ thống | Hiển thị form nhập thông tin loại thiết bị |
| 5 | Admin | Nhấn tạo loại thiết bị |
| 6 | Hệ thống | Kiểm tra thông tin |
| 7 | Hệ thống | Thông báo tạo loại thiết bị thành công và chuyển hướng sang màn hình hiển thị danh sách loại thiết bị |
| Luồng sự kiện thay thế | STT | Thực hiện | Hành động |
| 6a | Hệ thống | Thông tin không đúng yêu cầu |
| Hậu điều kiện | Không | | |

### 2.4.18. Thay đổi thông tin loại thiết bị

Bảng 2 - 18: Bảng đặc tả usecase thay đổi thông tin loại thiết bị

|  |  |  |  |
| --- | --- | --- | --- |
| Mã usecase | UC018 | Tên usecase | Thay đổi thông tin loại thiết bị |
| Tác nhân | Admin | | |
| Mô tả | Thay đổi thông tin loại thiết bị | | |
| Tiền điều kiện | Admin phải đăng nhập thành công vào hệ thống | | |
| Luồng sự kiện chính | STT | Thực hiện | Hành động |
| 1 | Admin | Nhấn loại thiết bị |
| 2 | Hệ thống | Hiển thị danh sách loại thiết bị đã tồn tại trong hệ thống, cột action bao gồm sửa và xóa ở mỗi loại thiết bị, button tạo loại thiết bị |
| 3 | Admin | Nhấn sửa ở cột action của loại thiết bị cần sửa |
| 4 | Hệ thống | Hiển thị form  thông tin loại thiết bị |
| 5 | Admin | Sửa đổi thông tin loại thiết bị |
| 6 | Admin | Nhấn sửa loại thiết bị |
| 7 | Hệ thống | Kiểm tra thông tin |
| 8 | Hệ thống | Thông báo sửa thông tin loại thiết bị thành công và chuyển hướng sang màn hình hiển thị danh sách loại thiết bị |
| Luồng sự kiện thay thế | STT | Thực hiện | Hành động |
| 7a | Hệ thống | Thông tin không đúng yêu cầu |
| 8a | Hệ thống | Thông báo thông tin đã tồn tại trong hệ thống |
| Hậu điều kiện | Không | | |

### 2.4.19. Xóa loại thiết bị

Bảng 2 - 19: Bảng đặc tả usecase xóa loại thiết bị

|  |  |  |  |
| --- | --- | --- | --- |
| Mã usecase | UC019 | Tên usecase | Xóa loại thiết bị |
| Tác nhân | Admin | | |
| Mô tả | Xóa loại thiết bị | | |
| Tiền điều kiện | Admin phải đăng nhập thành công vào hệ thống | | |
| Luồng sự kiện chính | STT | Thực hiện | Hành động |
| 1 | Admin | Nhấn loại thiết bị |
| 2 | Hệ thống | Hiển thị danh sách loại thiết bị đã tồn tại trong hệ thống, cột action bao gồm sửa và xóa ở mỗi loại thiết bị, button tạo loại thiết bị |
| 3 | Admin | Nhấn xóa ở cột action của loại thiết bị cần xóa |
| 4 | Hệ thống | Thực hiện xóa loại thiết bị |
| 5 | Hệ thống | Thông báo xóa loại thiết bị thành công và chuyển hướng sang màn hình hiển thị danh sách loại thiết bị |
| Luồng sự kiện thay thế | Không | | |
| Hậu điều kiện | Không | | |

### 2.4.20. Tạo quyền truy cập

Bảng 2 - 20: Bảng đặc tả usecase tạo quyền truy cập

|  |  |  |  |
| --- | --- | --- | --- |
| Mã usecase | UC020 | Tên usecase | Tạo quyền truy cập |
| Tác nhân | Admin | | |
| Mô tả | Tạo quyền truy cập | | |
| Tiền điều kiện | Admin phải đăng nhập thành công vào hệ thống | | |
| Luồng sự kiện chính | STT | Thực hiện | Hành động |
| 1 | Admin | Nhấn quyền truy cập |
| 2 | Hệ thống | Hiển thị danh sách quyền truy cập đã tồn tại trong hệ thống, nút tạo quyền truy cập |
| 3 | Admin | Nhấn nút tạo quyền truy cập |
| 4 | Hệ thống | Hiển thị form nhập thông tin quyền truy cập |
| 5 | Admin | Nhấn tạo quyền truy cập |
| 6 | Hệ thống | Kiểm tra thông tin |
| 7 | Hệ thống | Thông báo tạo quyền truy cập thành công và chuyển hướng sang màn hình hiển thị danh sách quyền truy cập |
| Luồng sự kiện thay thế | STT | Thực hiện | Hành động |
| 6a | Hệ thống | Thông tin không đúng yêu cầu |
| Hậu điều kiện | Không | | |

### 2.4.21. Thay đổi thông tin quyền truy cập

Bảng 2 - 21: Bảng đặc tả usecase thay đổi thông tin quyền truy cập

|  |  |  |  |
| --- | --- | --- | --- |
| Mã usecase | UC021 | Tên usecase | Thay đổi thông tin quyền truy cập |
| Tác nhân | Admin | | |
| Mô tả | Thay đổi thông tin quyền truy cập | | |
| Tiền điều kiện | Admin phải đăng nhập thành công vào hệ thống | | |
| Luồng sự kiện chính | STT | Thực hiện | Hành động |
| 1 | Admin | Nhấn quyền truy cập |
| 2 | Hệ thống | Hiển thị danh sách quyền truy cập đã tồn tại trong hệ thống, cột action bao gồm sửa và xóa ở mỗi quyền truy cập, button tạo quyền truy cập |
| 3 | Admin | Nhấn sửa ở cột action của quyền truy cập cần sửa |
| 4 | Hệ thống | Hiển thị form  thông tin quyền truy cập |
| 5 | Admin | Sửa đổi thông tin quyền truy cập |
| 6 | Admin | Nhấn sửa quyền truy cập |
| 7 | Hệ thống | Kiểm tra thông tin |
| 8 | Hệ thống | Thông báo sửa thông tin quyền truy cập thành công và chuyển hướng sang màn hình hiển thị danh sách quyền truy cập |
| Luồng sự kiện thay thế | STT | Thực hiện | Hành động |
| 7a | Hệ thống | Thông tin không đúng yêu cầu |
| 8a | Hệ thống | Thông báo thông tin đã tồn tại trong hệ thống |
| Hậu điều kiện | Không | | |

### 2.4.22. Xóa quyền truy cập

Bảng 2 - 22: Bảng đặc tả usecase xóa quyền truy cập

|  |  |  |  |
| --- | --- | --- | --- |
| Mã usecase | UC022 | Tên usecase | Xóa quyền truy cập |
| Tác nhân | Admin | | |
| Mô tả | Xóa quyền truy cập | | |
| Tiền điều kiện | Admin phải đăng nhập thành công vào hệ thống | | |
| Luồng sự kiện chính | STT | Thực hiện | Hành động |
| 1 | Admin | Nhấn quyền truy cập |
| 2 | Hệ thống | Hiển thị danh sách quyền truy cập đã tồn tại trong hệ thống, cột action bao gồm sửa và xóa ở mỗi quyền truy cập, button tạo quyền truy cập |
| 3 | Admin | Nhấn xóa ở cột action của quyền truy cập cần xóa |
| 4 | Hệ thống | Thực hiện xóa quyền truy cập |
| 5 | Hệ thống | Thông báo xóa quyền truy cập thành công và chuyển hướng sang màn hình hiển thị danh sách quyền truy cập |
| Luồng sự kiện thay thế | Không | | |
| Hậu điều kiện | Không | | |

### 2.4.23. Tạo lịch học

Bảng 2 - 23: Bảng đặc tả usecase tạo lịch học

|  |  |  |  |
| --- | --- | --- | --- |
| Mã usecase | UC023 | Tên usecase | Tạo lịch học |
| Tác nhân | Admin, trucban, giangvien | | |
| Mô tả | Tạo lịch làm việc | | |
| Tiền điều kiện | Admin phải đăng nhập thành công vào hệ thống | | |
| Luồng sự kiện chính | STT | Thực hiện | Hành động |
| 1 | Admin | Nhấn lịch học |
| 2 | Hệ thống | Hiển thị bảng thời gian lịch học, input tìm kiếm, button chọn phòng |
| 3 | Admin | Chọn phòng, sau đó nhấn vào 1 ngày bất kỳ trên bảng thời gian |
| 4 | Hệ thống | Hiển thị form nhập thông tin lịch học |
| 5 | Admin | Nhập thông tin lịch học và nhấn tạo lịch học |
| 6 | Hệ thống | Kiểm tra thông tin |
| 7 | Hệ thống | Thông báo tạo lịch học thành công và chuyển hướng sang màn hình hiển thị danh sách bảng thời gian lịch học |
| Luồng sự kiện thay thế | STT | Thực hiện | Hành động |
| 6a | Hệ thống | Thông tin không đúng yêu cầu |
| Hậu điều kiện | Không | | |

### 2.4.24. Thay đổi thông tin lịch học

Bảng 2 - 24: Bảng đặc tả usecase thay đổi thông tin lịch học

|  |  |  |  |
| --- | --- | --- | --- |
| Mã usecase | UC024 | Tên usecase | Thay đổi thông tin lịch học |
| Tác nhân | Admin | | |
| Mô tả | Thay đổi thông tin lịch học | | |
| Tiền điều kiện | Admin phải đăng nhập thành công vào hệ thống | | |
| Luồng sự kiện chính | STT | Thực hiện | Hành động |
| 1 | Admin | Nhấn lịch học |
| 2 | Hệ thống | Hiển thị bảng thời gian lịch làm việc, input tìm kiếm, button chọn phòng |
| 3 | Admin | Nhấn chọn phòng và chọn lịch làm việc cần sửa |
| 4 | Hệ thống | Hiển thị form thông tin lịch học |
| 5 | Admin | Sửa đổi thông tin lịch học |
| 6 | Admin | Nhấn sửa lịch học |
| 7 | Hệ thống | Kiểm tra thông tin |
| 8 | Hệ thống | Thông báo sửa lịch học thành công và chuyển hướng sang màn hình hiển thị danh sách lịch học |
| Luồng sự kiện thay thế | STT | Thực hiện | Hành động |
| 7a | Hệ thống | Thông tin không đúng yêu cầu |
| 8a | Hệ thống | Thông báo thông tin đã tồn tại trong hệ thống |
| Hậu điều kiện | Không | | |

### 2.4.25. Tạo lịch học bằng file excel

Bảng 2 - 25: Bảng đặc tả usecase thay đổi thông tin lịch học

|  |  |  |  |
| --- | --- | --- | --- |
| Mã usecase | UC025 | Tên usecase | Tạo lịch học bằng file excel |
| Tác nhân | Admin | | |
| Mô tả | Tạo nhiều lịch học bằng cách sử dụng excel | | |
| Tiền điều kiện | Admin phải đăng nhập thành công vào hệ thống | | |
| Luồng sự kiện chính | STT | Thực hiện | Hành động |
| 1 | Admin | Nhấn lịch học |
| 2 | Hệ thống | Hiển thị bảng thời gian lịch làm việc, button tạo lịch học, input import file |
| 3 | Admin | Nhấn import file |
| 4 | Hệ thống | Hiển thị file đã tồn tại trong thiết bị |
| 5 | Admin | Chọn file excel |
| 6 | Admin | Nhấn Save |
| 7 | Hệ thống | Kiểm tra thông tin |
| 8 | Hệ thống | Thông báo tạo lịch học thành công và chuyển hướng sang màn hình hiển thị danh sách lịch học |
| Luồng sự kiện thay thế | STT | Thực hiện | Hành động |
| 7a | Hệ thống | Thông tin không đúng yêu cầu |
| Hậu điều kiện | Không | | |

### 2.4.26. Tạo tài khoản bằng file excel

Bảng 2 - 26: Bảng đặc tả usecase thay đổi thông tin lịch học

|  |  |  |  |
| --- | --- | --- | --- |
| Mã usecase | UC026 | Tên usecase | Tạo tài khoản bằng file excel |
| Tác nhân | Admin | | |
| Mô tả | Tạo nhiều tài khoản bằng cách sử dụng excel | | |
| Tiền điều kiện | Admin phải đăng nhập thành công vào hệ thống | | |
| Luồng sự kiện chính | STT | Thực hiện | Hành động |
| 1 | Admin | Nhấn tài khoản |
| 2 | Hệ thống | Hiển thị danh sách tài khoản, button tạo tài khoản, input import file |
| 3 | Admin | Nhấn import file |
| 4 | Hệ thống | Hiển thị file đã tồn tại trong thiết bị |
| 5 | Admin | Chọn file excel |
| 6 | Admin | Nhấn Save |
| 7 | Hệ thống | Kiểm tra thông tin |
| 8 | Hệ thống | Thông báo tạo tài khoản thành công và chuyển hướng sang màn hình hiển thị danh sách tài khoản |
| Luồng sự kiện thay thế | STT | Thực hiện | Hành động |
| 7a | Hệ thống | Thông tin không đúng yêu cầu |
| Hậu điều kiện | Không | | |

### 2.4.27. Thông báo lịch học đến email của giảng viên

Bảng 2 - 27: Bảng đặc tả usecase thay đổi thông tin lịch học

|  |  |  |  |
| --- | --- | --- | --- |
| Mã usecase | UC027 | Tên usecase | Thông báo lịch học |
| Tác nhân | Admin | | |
| Mô tả | Thông báo lịch học đến email của giảng viên | | |
| Tiền điều kiện | Lịch học đã tồn tại trong hệ thống | | |
| Luồng sự kiện chính | STT | Thực hiện | Hành động |
| 1 | Hệ thống | Tự động chạy chương trình để lấy ra tất cả lịch học có trong hệ thống |
| 2 | Hệ thống | Tìm kiếm lịch học có thời gian bắt đầu cách thời điểm hiện tại là 2 tiếng |
| 3 | Hệ thống | Tìm kiếm email giảng viên phụ trách lịch học |
| 4 | Hệ thống | Gửi thông báo qua email của giảng viên |
| Luồng sự kiện thay thế | Không | | |
| Hậu điều kiện | Không | | |

## 2.5. Yêu cầu phi chức năng

Các yêu cầu phi chức năng cho một trang web xác định các tiêu chí mô tả cách thức một trang web, trang web phải hoạt động, đảm bảo rằng trang web đáp ứng các thuộc tính chất lượng nhất định và sự mong đợi của người dùng. Dưới đây là một số yêu cầu phi chức năng cho ứng dụng này:

### 2.5.1. Hiệu suất

* Thời gian phản hồi: Hệ thống phải có khả năng phản hồi nhanh chóng với thời gian tải trang dưới 3 giây cho tất cả các chức năng.
* Khả năng xử lý: Hệ thống cần hỗ trợ ít nhất 1000 người dùng đồng thời mà không ảnh hưởng đến hiệu năng.

### 2.5.2. Độ tin cậy

* Hệ thống phải đạt mức khả dụng ít nhất 99.9, với thời gian ngừng hoạt động tối đa không quá 8.76 giờ mỗi năm.
* Khôi phục sau sự cố: Hệ thống phải có khả năng khôi phục sau sự cố trong vòng 1 giờ.

### 2.5.3. Tính dễ dùng

* Giao diện người dùng thân thiện: Giao diện phải trực quan và dễ sử dụng, phù hợp với người dùng không có nhiều kinh nghiệm về công nghệ

### 2.5.4. Tính dễ bảo trì

* Mã nguồn: Mã nguồn phải được viết rõ ràng, có cấu trúc tốt, dễ hiểu và dễ bảo trì
* Tài liệu: Mã nguồn, kiến trúc và chức năng của trang web được ghi lại đầy đủ để dễ bảo trì và phát triển trong tương lai.

### 2.5.5. Yêu cầu về kỹ thuật

* Cơ sở dữ liệu: Hệ thống sử dụng PostgreSQL làm nền tảng quản lý dữ liệu. PostgreSQL được chọn vì tính ổn định, khả năng mở rộng cao và hỗ trợ các thao tác phức tạp trên dữ liệu. Việc triển khai PostgreSQL được thực hiện thông qua Docker, đảm bảo môi trường phát triển đồng nhất và dễ dàng di chuyển.
* Công nghệ sử dụng:

i, Frontend: Giao diện người dùng được xây dựng bằng Next.js

ii, Backend: API phía server được phát triển bằng Gin Framework

iii, Websocket: Công nghệ WebSocket được tích hợp để hỗ trợ các tính năng nhắn tin thời gian thực

# CHƯƠNG 3: CÔNG NGHỆ SỬ DỤNG.

## 3.1. Framework front-end: NextJS



Hình 3 - 1: NextJs

### 3.1.1. Giới thiệu về NextJs

Next.js là một framework hàng đầu trong lĩnh vực phát triển web hiện đại, được xây dựng trên nền tảng React nhằm tối ưu hóa việc tạo ra các ứng dụng web nhanh chóng, mạnh mẽ và thân thiện với người dùng. Được phát triển bởi Vercel, Next.js ra mắt vào năm 2016 và nhanh chóng trở thành một lựa chọn phổ biến của các lập trình viên nhờ khả năng hỗ trợ đa dạng các phương thức kết xuất, từ Server-Side Rendering (SSR), Static Site Generation (SSG), cho đến Incremental Static Regeneration (ISR).

Framework này không chỉ đơn thuần là công cụ hỗ trợ phát triển giao diện mà còn cung cấp các tính năng mạnh mẽ để tích hợp backend ngay trong cùng một dự án. Next.js giúp thu hẹp khoảng cách giữa frontend và backend, giảm thiểu sự phức tạp trong việc triển khai ứng dụng. Đặc biệt, với khả năng tối ưu hóa hiệu suất và trải nghiệm người dùng, Next.js hiện được nhiều tập đoàn công nghệ lớn như Netflix, Uber, và Hulu tin dùng.

### 3.1.2. Các khái niệm về NextJs

#### 3.1.2.1. File-base Routing

Next.js sử dụng hệ thống định tuyến tự động dựa trên cấu trúc thư mục trong các thư mục pages hoặc app. Hệ thống này giúp loại bỏ nhu cầu cấu hình định tuyến thủ công phức tạp, thay vào đó mỗi tệp trong thư mục tương ứng trực tiếp với một đường dẫn URL. Chẳng hạn, tệp pages/index.js sẽ tương ứng với URL /, còn pages/blog.js sẽ tương ứng với URL /blog. Đối với các thư mục lồng nhau, đường dẫn cũng được xây dựng dựa trên cấu trúc thư mục, ví dụ: pages/blog/posts.js sẽ tương ứng với /blog/posts.

Ngoài ra, Next.js hỗ trợ dynamic routing (định tuyến động) thông qua việc sử dụng các tên tệp động, chẳng hạn như [id].js. Điều này giúp ứng dụng dễ dàng tạo các URL động như /blog/123 mà không cần phải định nghĩa trước tất cả các đường dẫn. Hệ thống này không chỉ đơn giản hóa việc quản lý định tuyến mà còn giúp code dễ đọc, dễ bảo trì.

#### 3.1.2.2. Các phương thức kết xuất

Static Site Generation (SSG): Đây là phương thức mà các trang được tạo ra dưới dạng nội dung tĩnh tại thời điểm build ứng dụng. Điều này có nghĩa là toàn bộ HTML của trang được dựng trước và có thể được phân phát ngay lập tức từ server hoặc CDN mà không cần xử lý thêm. SSG giúp tối ưu hóa tốc độ tải trang, cải thiện SEO và giảm tải cho server. Phương pháp này đặc biệt phù hợp với các trang có nội dung ít thay đổi, chẳng hạn như trang giới thiệu sản phẩm hoặc blog.

Server-Side Rendering (SSR): Trái ngược với SSG, SSR cho phép nội dung của trang được dựng lại ở phía server mỗi lần có yêu cầu từ người dùng. Điều này đảm bảo dữ liệu luôn được cập nhật theo thời gian thực, phù hợp với các trang cần hiển thị thông tin động như bảng điều khiển, trang thương mại điện tử, hoặc các ứng dụng quản lý. Dù hiệu năng không cao bằng SSG, SSR vẫn đảm bảo sự linh hoạt trong việc hiển thị dữ liệu.

Incremental Static Regeneration (ISR): ISR kết hợp ưu điểm của cả SSG và SSR, cho phép các trang tĩnh được cập nhật định kỳ mà không cần phải build lại toàn bộ ứng dụng. Với ISR, có thể đặt thời gian làm mới (revalidate) cho từng trang, giúp ứng dụng vừa duy trì hiệu suất cao, vừa hiển thị dữ liệu mới một cách linh hoạt. Phương pháp này thích hợp cho các trang tin tức hoặc blog có tần suất cập nhật dữ liệu không quá cao.

#### 3.1.2.3. API Routes

Next.js cung cấp khả năng xây dựng API ngay trong dự án thông qua thư mục pages/api. Mỗi tệp trong thư mục này tương ứng với một endpoint API, giúp tạo các chức năng backend như xử lý form, xác thực người dùng, hoặc lấy dữ liệu từ cơ sở dữ liệu mà không cần sử dụng thêm framework hoặc server bên ngoài.

API Routes giúp tích hợp backend và frontend trong cùng một dự án, đơn giản hóa quá trình phát triển và triển khai ứng dụng full-stack. Ví dụ: một tệp pages/api/hello.js có thể xử lý một endpoint /api/hello, nơi viết code để trả về phản hồi JSON hoặc thực hiện các logic server khác.

#### 3.1.2.4. Tối ưu hóa hình ảnh

Next.js đi kèm với một công cụ mạnh mẽ để tối ưu hóa hình ảnh thông qua thành phần next/image. Công cụ này hỗ trợ nén ảnh tự động, lazy loading (tải ảnh khi cần thiết), và cung cấp các kích thước ảnh phù hợp dựa trên thiết bị người dùng. Điều này không chỉ giúp giảm thời gian tải trang mà còn tăng hiệu suất ứng dụng và nâng cao trải nghiệm người dùng.

Ngoài ra, tính năng này còn hỗ trợ định dạng hình ảnh hiện đại như WebP, giúp giảm dung lượng file ảnh mà không ảnh hưởng đến chất lượng. Khả năng tối ưu hóa hình ảnh của Next.js đặc biệt quan trọng trong các ứng dụng hiển thị nhiều hình ảnh như các trang thương mại điện tử hoặc portfolio cá nhân.

#### 3.1.2.5. Hỗ trợ CSS linh hoạt

Next.js cho phép lập trình viên sử dụng nhiều công cụ và phương pháp để quản lý giao diện. Framework hỗ trợ CSS Modules, giúp viết CSS có phạm vi riêng biệt cho từng thành phần mà không sợ xung đột với các lớp CSS khác. Ngoài ra, Next.js còn tương thích với các thư viện như Styled-components và Emotion, giúp áp dụng các kỹ thuật CSS-in-JS hiện đại.

Đối với các dự án lớn hoặc cần hiệu suất cao, có thể tích hợp Tailwind CSS để tận dụng các lớp tiện ích (utility-first CSS), giúp tăng tốc độ phát triển giao diện mà vẫn giữ được tính tùy chỉnh cao.

#### 3.1.2.6. Middleware

Middleware trong Next.js là các đoạn code chạy giữa giai đoạn yêu cầu từ phía người dùng (request) và phản hồi từ server (response). Middleware được sử dụng để xử lý các logic phức tạp như xác thực người dùng, quản lý quyền truy cập, hoặc kiểm tra các điều kiện trước khi cho phép truy cập vào một trang cụ thể.

Tính năng này đặc biệt hữu ích trong các ứng dụng cần bảo mật cao hoặc có luồng xử lý phức tạp, giúp tổ chức code một cách rõ ràng và dễ mở rộng.

#### 3.1.2.7. Built-in TypeScript Support

Next.js hỗ trợ TypeScript ngay từ đầu mà không cần cấu hình phức tạp. Điều này đảm bảo có thể sử dụng TypeScript để viết mã nguồn với kiểu dữ liệu rõ ràng, giúp giảm thiểu lỗi runtime và cải thiện khả năng bảo trì. Hơn nữa, TypeScript tích hợp sẵn trong Next.js giúp việc chuyển đổi từ JavaScript sang TypeScript trở nên dễ dàng hơn bao giờ hết.

#### 3.1.2.8. Tích hợp mạnh mẽ với các công nghệ hiện đại

Next.js dễ dàng tích hợp với nhiều dịch vụ đám mây như AWS, Firebase, và Vercel (dịch vụ chính thức hỗ trợ Next.js). Framework này cũng hỗ trợ triển khai các công cụ DevOps như Docker, Kubernetes, giúp triển khai ứng dụng một cách nhanh chóng và hiệu quả. Khả năng kết nối với các cơ sở dữ liệu như MongoDB, PostgreSQL, hay Prisma cũng là một điểm mạnh, làm cho Next.js trở thành một lựa chọn tuyệt vời cho các dự án full-stack.

### 3.1.3. Lý do sử dụng

#### 3.1.3.1. Hiệu suất vượt trội

Next.js nổi tiếng với hiệu suất vượt trội, nhờ sự kết hợp linh hoạt của các phương thức kết xuất như **Static Site Generation (SSG)**, **Server-Side Rendering (SSR)** và **Incremental Static Regeneration (ISR)**. Các trang được tối ưu hóa để tải nhanh chóng thông qua nội dung tĩnh hoặc được tạo động khi cần thiết. Điều này không chỉ giúp giảm thời gian tải trang mà còn tăng khả năng tương tác của người dùng, đặc biệt quan trọng trong các ứng dụng thương mại điện tử hoặc dịch vụ trực tuyến.

Ngoài ra, Next.js còn hỗ trợ các tính năng như **lazy loading** (tải nội dung khi cần thiết), tối ưu hóa hình ảnh tự động và nén các tệp JavaScript/CSS, đảm bảo rằng các tài nguyên được phân phối hiệu quả. Kết quả là ứng dụng không chỉ chạy mượt trên các thiết bị cao cấp mà còn đảm bảo trải nghiệm tốt trên các thiết bị cấu hình thấp hoặc kết nối internet chậm.

#### 3.1.3.2. Tối ưu SEO

Tối ưu hóa công cụ tìm kiếm (SEO) là một trong những ưu điểm nổi bật nhất của Next.js. Framework này cung cấp khả năng kết xuất **HTML đầy đủ** ngay từ server thông qua SSR và SSG. Điều này giúp các trang web được trình thu thập thông tin của Google và các công cụ tìm kiếm khác lập chỉ mục một cách chính xác và hiệu quả hơn so với các ứng dụng đơn trang truyền thống (SPA), nơi dữ liệu thường được tải qua JavaScript sau khi trang đã tải xong.

Bên cạnh đó, Next.js còn hỗ trợ **dynamic meta tags** (thẻ meta động), cho phép dễ dàng tùy chỉnh tiêu đề, mô tả và các thuộc tính khác cho từng trang, cải thiện xếp hạng tìm kiếm của trang web. Điều này đặc biệt hữu ích cho các trang thương mại điện tử, blog, hoặc bất kỳ ứng dụng nào yêu cầu tối ưu hóa SEO để tăng lưu lượng truy cập.

#### 3.1.3.3. Khả năng mở rộng và đa dạng

Next.js là một framework linh hoạt, phù hợp với nhiều loại dự án khác nhau, từ các ứng dụng nhỏ như trang portfolio cá nhân đến các hệ thống lớn và phức tạp như nền tảng thương mại điện tử hoặc mạng xã hội. Framework này hỗ trợ tích hợp dễ dàng với các dịch vụ backend thông qua API Routes, cơ sở dữ liệu như MongoDB hoặc PostgreSQL, và các công cụ như GraphQL.

Hơn nữa, Next.js được thiết kế để hoạt động tốt trong môi trường **microservices** hoặc các hệ thống phân tán, nhờ khả năng triển khai độc lập từng phần của ứng dụng. Có thể mở rộng ứng dụng mà không lo ngại về hiệu suất hoặc cấu trúc hệ thống, điều này đặc biệt quan trọng đối với các doanh nghiệp đang phát triển.

#### 3.1.3.4. Trải nghiệm phát triển mượt mà

Next.js mang đến một môi trường phát triển thân thiện, giúp lập trình viên tiết kiệm thời gian và tập trung vào việc xây dựng tính năng. Các tính năng như **hot-reload** (tải lại trang ngay lập tức khi có thay đổi trong mã nguồn) và hỗ trợ **TypeScript** giúp quá trình phát triển trở nên nhanh chóng và chính xác hơn.

Ngoài ra, Next.js đi kèm với các công cụ tích hợp mạnh mẽ như ESLint và Prettier, giúp duy trì chất lượng mã nguồn và giảm thiểu lỗi cú pháp hoặc logic. Việc hỗ trợ các thư viện UI như Tailwind CSS, Styled-components, và Emotion cũng giúp lập trình viên tùy chỉnh giao diện một cách dễ dàng và sáng tạo.

Đặc biệt, Next.js còn hỗ trợ tích hợp với **Vercel**, nền tảng triển khai chính thức của framework, cho phép triển khai ứng dụng chỉ với một vài bước đơn giản. Điều này giúp giảm bớt các công việc DevOps phức tạp, giúp các nhóm phát triển tập trung hơn vào sản phẩm.

#### 3.1.3.5. Cộng đồng lớn và tài liệu phong phú

Next.js được duy trì bởi Vercel, một công ty nổi tiếng trong lĩnh vực phát triển phần mềm, cùng với sự hỗ trợ của một cộng đồng lập trình viên đông đảo. Framework này có một hệ sinh thái mạnh mẽ và tài liệu chính thức được biên soạn kỹ lưỡng, bao gồm các ví dụ, hướng dẫn chi tiết và câu trả lời cho các câu hỏi thường gặp.

Cộng đồng Next.js trên các nền tảng như GitHub, Reddit, và Stack Overflow rất năng động, giúp dễ dàng tìm kiếm sự hỗ trợ hoặc giải pháp cho các vấn đề gặp phải trong quá trình phát triển. Hơn nữa, các hội nghị, bài viết blog và video hướng dẫn từ cộng đồng cũng cung cấp nguồn cảm hứng và kiến thức để áp dụng Next.js vào các dự án thực tế.

Sự hỗ trợ từ cộng đồng không chỉ giúp Next.js trở thành một framework dễ học, dễ sử dụng mà còn đảm bảo luôn có thể tìm thấy tài nguyên phù hợp để giải quyết mọi thách thức trong quá trình phát triển.

## 3.2. Framework back-end: Gin



Hình 3 - 2: Gin

### 3.2.1. Giới thiệu về Gin

Gin là một web framework mạnh mẽ và hiệu quả dành cho Go (Golang), được thiết kế để xây dựng các API và dịch vụ web tốc độ cao. Ra mắt lần đầu vào năm 2014, Gin nhanh chóng trở thành một trong những lựa chọn phổ biến nhất trong cộng đồng Go nhờ vào sự đơn giản và hiệu suất vượt trội.

Điểm nổi bật của Gin chính là middleware linh hoạt, cho phép lập trình viên mở rộng các tính năng hoặc tùy chỉnh logic dễ dàng. Gin dựa trên thư viện httprouter giúp xử lý routing nhanh chóng, đồng thời hỗ trợ xử lý JSON, form binding, validation, logging, và cả lỗi (errors) một cách hiệu quả.

### 3.2.2. Lý do sử dụng

#### 3.2.2.1. Hiệu suất vượt trội

Gin là một framework Go được tối ưu hóa để xử lý lượng lớn yêu cầu với độ trễ thấp, nhờ vào việc sử dụng thư viện **httprouter**. Thư viện này xử lý định tuyến cực kỳ nhanh chóng, giúp Gin có thể phục vụ hàng triệu yêu cầu mỗi giây mà không gây ảnh hưởng đến hiệu suất. Với khả năng xử lý song song và tối ưu hóa tài nguyên, Gin đặc biệt phù hợp cho các ứng dụng yêu cầu khả năng mở rộng cao, chẳng hạn như các API REST hoặc các dịch vụ microservices.

Khi sử dụng Gin, các ứng dụng có thể tận dụng được những lợi thế về tốc độ phản hồi và giảm thiểu gánh nặng tài nguyên, đặc biệt là trong các hệ thống phải đáp ứng nhiều yêu cầu đồng thời. Điều này giúp ứng dụng luôn hoạt động ổn định và hiệu quả ngay cả khi có tải cao, đồng thời đảm bảo trải nghiệm người dùng không bị gián đoạn.

#### 3.2.2.2. API thân thiện và dễ sử dụng

Gin cung cấp một API rất rõ ràng và dễ sử dụng, với cú pháp ngắn gọn nhưng đầy đủ tính năng. Điều này giúp các lập trình viên, dù mới bắt đầu làm quen với Go hay đã có kinh nghiệm, có thể nhanh chóng nắm bắt và sử dụng Gin để phát triển ứng dụng.

Cấu trúc của Gin rất đơn giản và dễ hiểu, chỉ với vài dòng mã, lập trình viên có thể nhanh chóng xây dựng một REST API cơ bản. Việc giảm thiểu sự phức tạp trong cú pháp giúp giảm thời gian học hỏi và tăng tốc độ phát triển dự án, từ đó tạo ra một quy trình phát triển hiệu quả hơn. Nhờ vào tính dễ sử dụng này, Gin trở thành một lựa chọn tuyệt vời cho các dự án yêu cầu thời gian phát triển ngắn mà vẫn đảm bảo chất lượng.

#### 3.2.2.3. Middleware linh hoạt

Một trong những lý do quan trọng khiến Gin trở nên phổ biến chính là khả năng sử dụng và tùy chỉnh middleware linh hoạt. Middleware trong Gin cho phép lập trình viên can thiệp vào quá trình xử lý yêu cầu HTTP, từ đó quản lý các chức năng như xác thực người dùng, log, kiểm soát truy cập và nhiều thứ khác.

Ví dụ, có thể sử dụng middleware để kiểm tra tính hợp lệ của yêu cầu hoặc để xác thực người dùng trước khi yêu cầu tiếp tục được xử lý. Điều này giúp tăng cường bảo mật và khả năng quản lý của ứng dụng. Middleware trong Gin có thể được áp dụng toàn cục cho toàn bộ ứng dụng hoặc chỉ áp dụng cho một số route nhất định, mang lại sự linh hoạt tối đa trong việc quản lý các chức năng của ứng dụng.

#### 3.2.2.4. Tích hợp mạnh mẽ với các công cụ Go

Gin được thiết kế để tích hợp mượt mà với các thư viện và công cụ phổ biến trong hệ sinh thái Go. Một trong những ví dụ điển hình là việc tích hợp dễ dàng với **GORM**, một thư viện ORM phổ biến để kết nối và quản lý cơ sở dữ liệu. Sự kết hợp giữa Gin và GORM giúp lập trình viên dễ dàng xây dựng các ứng dụng web có khả năng tương tác mạnh mẽ với cơ sở dữ liệu mà không phải lo lắng về việc xử lý các vấn đề phức tạp liên quan đến kết nối cơ sở dữ liệu.

Ngoài ra, Gin còn hỗ trợ tích hợp với **Swagger** để tạo tài liệu API tự động, giúp lập trình viên có thể nhanh chóng cung cấp thông tin chi tiết về các API mà họ xây dựng. Việc này giúp quá trình phát triển trở nên thuận tiện hơn, nhất là khi làm việc trong một nhóm lớn hoặc khi cần chia sẻ thông tin API với các đội phát triển khác.

#### 3.2.2.5. Xử lý dữ liệu mạnh mẽ

Gin cung cấp khả năng xử lý dữ liệu từ các request rất mạnh mẽ thông qua các tính năng như **binding** và **validation**. Việc binding giúp dễ dàng trích xuất dữ liệu từ các nguồn khác nhau như JSON body, query string, hoặc form data. Điều này giúp giảm thiểu việc viết mã phức tạp và cho phép lập trình viên dễ dàng lấy và sử lý dữ liệu từ người dùng.

Hơn nữa, Gin hỗ trợ tính năng **validation**, cho phép lập trình viên kiểm tra tính hợp lệ của dữ liệu trước khi sử dụng. Điều này giúp giảm thiểu lỗi lập trình, bảo vệ ứng dụng khỏi các đầu vào không hợp lệ và tăng cường tính bảo mật của hệ thống. Việc xử lý dữ liệu được thực hiện một cách rõ ràng và dễ dàng, giúp lập trình viên tập trung vào các nhiệm vụ quan trọng khác trong dự án.

#### 3.2.2.6. Cộng đồng và tài liệu phong phú

Gin không chỉ là một framework mạnh mẽ mà còn được hỗ trợ bởi một cộng đồng lớn và rất nhiệt tình. Cộng đồng này chia sẻ các tài nguyên học tập, bài viết, và giải pháp cho các vấn đề mà lập trình viên gặp phải trong quá trình phát triển ứng dụng.

Tài liệu chính thức của Gin rất chi tiết và dễ hiểu, với các ví dụ trực quan và đầy đủ hướng dẫn, giúp người mới bắt đầu cũng có thể nhanh chóng nắm bắt được cách sử dụng framework này. Ngoài tài liệu chính thức, cộng đồng của Gin còn đóng góp nhiều bài viết blog, video hướng dẫn và hội thảo trực tuyến, tạo ra một kho tài nguyên phong phú cho lập trình viên. Nhờ vào sự hỗ trợ của cộng đồng và tài liệu phong phú, Gin trở thành một framework rất dễ tiếp cận và học hỏi, giúp lập trình viên nhanh chóng triển khai các dự án thực tế.

## 3.3. Redux



Hình 3 - 3: Redux

### 3.3.1. Giới thiệu về Redux

Redux là một thư viện quản lý trạng thái ứng dụng phổ biến trong phát triển web hiện đại, đặc biệt khi kết hợp với React. Được phát triển bởi Dan Abramov và Andrew Clark, Redux tuân theo nguyên tắc quản lý trạng thái dựa trên một kho lưu trữ duy nhất (store) và các hành động (actions) để cập nhật trạng thái. Mục tiêu của Redux là cung cấp một cách tiếp cận có tổ chức, dễ dự đoán và duy trì trong việc quản lý dữ liệu ứng dụng.

### 3.3.2. Thành phần chính và cách hoạt động

#### 3.3.2.1. Store

Store trong Redux là thành phần quản lý trạng thái duy nhất của ứng dụng. Nó có vai trò lưu trữ toàn bộ trạng thái (state) của ứng dụng, giúp đảm bảo rằng mọi thay đổi trạng thái đều được kiểm soát một cách chặt chẽ. Redux sử dụng **single source of truth** (nguồn sự thật duy nhất), có nghĩa là toàn bộ dữ liệu của ứng dụng được lưu trữ trong một đối tượng duy nhất. Điều này giúp dễ dàng theo dõi, quản lý và duy trì tính nhất quán của dữ liệu trên toàn bộ ứng dụng.

Store không chỉ lưu trữ trạng thái mà còn cung cấp các phương thức như getState() để lấy trạng thái hiện tại và dispatch() để gửi các action nhằm thay đổi trạng thái. Chính nhờ vào cách quản lý trạng thái tập trung này mà Redux giúp giảm thiểu sự phức tạp trong việc đồng bộ hóa và quản lý trạng thái khi ứng dụng phát triển lớn hơn và có nhiều thành phần hơn. Việc tất cả các phần của ứng dụng truy cập vào cùng một nguồn dữ liệu giúp tránh được tình trạng dữ liệu bị phân mảnh, làm cho ứng dụng dễ bảo trì và mở rộng hơn.

#### 3.3.2.2. Action

Action trong Redux là các đối tượng mô tả một sự kiện hoặc một yêu cầu để thay đổi trạng thái. Mỗi action phải có ít nhất một trường type, trường này xác định loại hành động mà ứng dụng cần thực hiện, ví dụ như "ADD\_ITEM" hoặc "REMOVE\_ITEM". Ngoài trường type, action có thể mang thêm các dữ liệu bổ sung (payload) để cung cấp thêm thông tin cần thiết cho việc cập nhật trạng thái. Dữ liệu này có thể là bất kỳ loại dữ liệu nào, chẳng hạn như một đối tượng, mảng, hoặc số.

Actions có thể được gọi từ các thành phần giao diện người dùng (UI) khi người dùng tương tác, hoặc có thể được kích hoạt từ các sự kiện khác trong ứng dụng, chẳng hạn như phản hồi từ API. Mặc dù actions không trực tiếp thay đổi trạng thái, nhưng chúng gửi yêu cầu đến store để thực hiện các thay đổi đó thông qua reducers. Action giúp xác định những thay đổi cần thực hiện đối với trạng thái của ứng dụng và truyền tải các thông tin cần thiết để thực hiện chúng.

#### 3.3.2.3. Reducer

Reducer trong Redux là một hàm thuần túy (pure function) dùng để xử lý các action và trả về trạng thái mới của ứng dụng. Reducer nhận vào hai đối số: **state** (trạng thái hiện tại) và **action** (hành động cần thực hiện). Tùy vào loại action được gửi đến, reducer sẽ thực hiện thay đổi trạng thái và trả về trạng thái mới. Một điều quan trọng cần nhớ là reducer không bao giờ thay đổi trực tiếp trạng thái hiện tại mà sẽ tạo ra một bản sao mới của trạng thái để giữ tính bất biến (immutability).

Reducers trong Redux phải giữ cho logic đơn giản và dễ hiểu. Mỗi reducer thường sẽ xử lý một phần nhỏ của trạng thái, chẳng hạn như danh sách các sản phẩm trong giỏ hàng, hoặc thông tin người dùng đang đăng nhập. Nếu ứng dụng có nhiều phần trạng thái khác nhau, các reducer có thể được kết hợp lại thành một reducer tổng hợp thông qua hàm combineReducers(). Reducers giúp đảm bảo rằng mọi thay đổi trạng thái trong ứng dụng đều được kiểm soát và dễ dàng tái tạo.

#### 3.3.2.4. Dispatch

Dispatch là phương thức được sử dụng để gửi một action tới store. Khi một action được dispatch, store sẽ xử lý hành động đó bằng cách chuyển giao cho các reducer. Các reducer sau đó sẽ xác định cách thay đổi trạng thái dựa trên action mà chúng nhận được. Quá trình này giúp đảm bảo rằng mọi thay đổi trạng thái đều được thực hiện thông qua các bước rõ ràng và có thể kiểm soát.

Dispatch là một phương thức quan trọng trong việc kích hoạt các thay đổi trạng thái trong ứng dụng. Nó cho phép các thành phần giao diện người dùng hoặc các sự kiện khác trong ứng dụng gửi các action để yêu cầu thay đổi trạng thái. Dispatch có thể được sử dụng trong các sự kiện người dùng, chẳng hạn như nhấp chuột hoặc nhập liệu, hoặc trong các tác vụ không đồng bộ, như gọi API.

#### 3.3.2.5. Selector

Selector là các hàm dùng để truy xuất dữ liệu từ store một cách hiệu quả. Thay vì trực tiếp truy cập trạng thái trong store, các thành phần của ứng dụng sẽ sử dụng selectors để lấy thông tin mà chúng cần. Điều này giúp mã nguồn trở nên dễ đọc và dễ bảo trì hơn, vì selectors có thể được tối ưu hóa để chỉ trả về các phần dữ liệu cần thiết cho mỗi thành phần.

Selectors có thể giúp giảm thiểu việc tính toán lại dữ liệu không cần thiết. Ví dụ, có thể sử dụng reselect (một thư viện phổ biến) để tạo các selector memoized, tức là các selector sẽ chỉ tính toán lại giá trị khi dữ liệu đầu vào thay đổi. Điều này giúp nâng cao hiệu suất, đặc biệt là trong những ứng dụng lớn với nhiều dữ liệu và yêu cầu render lại thường xuyên.

#### 3.3.2.6. Middleware

Middleware trong Redux là một công cụ rất hữu ích để mở rộng và cải thiện khả năng xử lý của Redux. Các middleware có thể can thiệp vào quá trình dispatch của action và thực hiện các tác vụ bổ sung trước hoặc sau khi action được gửi đến reducer. Ví dụ, middleware có thể được sử dụng để xử lý các tác vụ không đồng bộ (asynchronous tasks), chẳng hạn như gọi API và cập nhật trạng thái sau khi nhận được phản hồi.

Một ví dụ điển hình của middleware trong Redux là **redux-thunk**, cho phép dispatch các action không đồng bộ, chẳng hạn như gửi yêu cầu HTTP. Các middleware khác như **redux-logger** có thể dùng để log các action và trạng thái, giúp lập trình viên theo dõi những thay đổi đang xảy ra trong ứng dụng. Middleware giúp tối ưu hóa và mở rộng khả năng của Redux, làm cho nó trở thành một công cụ mạnh mẽ cho việc quản lý trạng thái trong các ứng dụng phức tạp.

Cách hoạt động

1. Người dùng thực hiện một thao tác trong UI (giao diện người dùng).
2. Một action được tạo ra và gửi tới store thông qua dispatch.
3. Store chuyển action tới reducer phù hợp.
4. Reducer cập nhật trạng thái dựa trên action nhận được và trả về trạng thái mới.
5. Store thông báo UI về sự thay đổi trạng thái, từ đó giao diện được cập nhật.

Hình ảnh sau minh họa quy trình hoạt động của Redux:

A diagram of a software development process

Description automatically generated

Hình 3 - 4: Cách hoạt động của Redux

## 3.4. Websocket



Hình 3 - 5: Websocket

### 3.4.1. Giới thiệu về Websocket

WebSocket là một giao thức truyền thông cung cấp kênh giao tiếp hai chiều (bi-directional) trên một kết nối TCP duy nhất. Được chuẩn hóa bởi IETF (Internet Engineering Task Force) trong RFC 6455, WebSocket được thiết kế để thay thế các phương pháp truyền thống như HTTP polling hoặc long polling, giúp tăng hiệu suất và giảm độ trễ trong các ứng dụng thời gian thực.

### 3.4.2.Các thành phần chính của WebSocket:

#### 3.4.2.1.Client (Máy khách):

Client trong kiến trúc WebSocket thường là trình duyệt web hoặc một ứng dụng gửi yêu cầu kết nối đến máy chủ qua giao thức WebSocket. Client có nhiệm vụ khởi tạo kết nối và duy trì kết nối WebSocket đến máy chủ. Sau khi kết nối được thiết lập, client có thể gửi và nhận dữ liệu từ máy chủ mà không cần phải thiết lập lại kết nối mỗi lần, giúp cải thiện hiệu suất và tiết kiệm băng thông.

Trong trường hợp của trình duyệt, client có thể sử dụng API WebSocket tích hợp sẵn để mở kết nối đến máy chủ. Khi kết nối thành công, client có thể gửi các tin nhắn qua kết nối WebSocket đã mở và nhận các tin nhắn từ máy chủ trong thời gian thực, mà không cần phải gửi lại yêu cầu HTTP mỗi lần.

#### 3.4.2.2. Server (Máy chủ):

Máy chủ WebSocket có nhiệm vụ tiếp nhận các yêu cầu kết nối từ client, xử lý chúng và duy trì kết nối liên tục với client trong suốt thời gian giao tiếp. Sau khi nhận yêu cầu kết nối từ client, máy chủ thực hiện quá trình "handshake" để thiết lập một kết nối WebSocket. Khi kết nối đã được thiết lập, máy chủ sẽ quản lý các thông điệp được gửi từ client đến server và ngược lại. Một trong những tính năng đặc biệt của WebSocket là khả năng giao tiếp hai chiều, cho phép server chủ động gửi dữ liệu tới client bất cứ lúc nào mà không cần chờ yêu cầu từ client.

Máy chủ WebSocket cũng có thể quản lý nhiều kết nối đồng thời với các client khác nhau, lưu trữ và xử lý các dữ liệu liên quan đến các kết nối này. Ngoài ra, máy chủ có thể thực hiện các tác vụ như phân phối tin nhắn cho các client khác nhau, xử lý các kết nối không còn hoạt động, và bảo mật các kết nối để ngăn chặn truy cập trái phép.

#### 3.4.2.3. Protocol (Giao thức):

Sử dụng giao thức TCP và một handshake thông qua HTTP để thiết lập kết nối.

Sau khi kết nối được mở, giao thức chuyển sang chế độ truyền dữ liệu hai chiều.

Cách hoạt động của WebSocket:

* 1. Handshake:Kết nối bắt đầu bằng một yêu cầu HTTP từ client với một tiêu đề đặc biệt (Upgrade: websocket) để chuyển đổi từ HTTP sang WebSocket.
  2. Kết nối mở: Nếu handshake thành công, kết nối WebSocket được thiết lập. Server và client có thể gửi dữ liệu qua lại mà không cần thiết lập lại kết nối.
  3. Trao đổi dữ liệu:Dữ liệu được truyền theo định dạng khung (frame), với các khung chứa dữ liệu (payload) cần thiết.
  4. Giao tiếp là thời gian thực và hai chiều, không yêu cầu client phải khởi tạo yêu cầu để nhận dữ liệu.
  5. Kết nối đóng:Bất kỳ bên nào (client hoặc server) đều có thể đóng kết nối khi cần thiết bằng cách gửi một khung đóng (close frame).

Lợi ích:

1. Độ trễ thấp: Phù hợp với các ứng dụng yêu cầu thời gian thực như chat, game online, hoặc giao dịch tài chính.
2. Hiệu suất cao: Giảm tải cho máy chủ so với các phương pháp truyền thống.
3. Đơn giản: Dễ dàng triển khai và mở rộng.

Hình ảnh dưới đây minh họa cách WebSocket hoạt động:

Client gửi yêu cầu handshake đến server.

Server phản hồi và kết nối WebSocket được thiết lập.

Cả hai bên trao đổi dữ liệu trong thời gian thực.

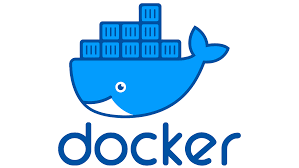
Kết nối được đóng khi không cần thiết.

A diagram of a computer network

Description automatically generated

Hình 3 - 6: Cách hoạt động của Websocket

## 3.5. Docker



Hình 3 - 7: Docker

### 3.5.1. Giới thiệu về docker

Docker là một nền tảng mã nguồn mở cho phép tạo, triển khai và chạy các ứng dụng bên trong container. Container là các môi trường nhẹ, độc lập, bao gồm đầy đủ các thư viện, phụ thuộc, và cấu hình cần thiết để chạy ứng dụng, giúp ứng dụng chạy nhất quán trên mọi môi trường (máy local, staging, production...).

### 3.5.2. Lý do sử dụng

#### 3.5.2.1. Tính nhất quán trong môi trường làm việc

Docker đảm bảo rằng ứng dụng sẽ hoạt động giống nhau trên mọi máy tính, bất kể hệ điều hành hay cấu hình phần cứng. Điều này giúp loại bỏ các sự cố liên quan đến sự khác biệt giữa môi trường phát triển và môi trường sản xuất, ví dụ như tình trạng "works on my machine but not on the server". Docker cho phép đóng gói ứng dụng và tất cả các phụ thuộc của nó vào trong một container, giúp ứng dụng chạy chính xác như vậy trên mọi hệ thống. Do đó, không cần phải lo lắng về sự khác biệt giữa môi trường phát triển, môi trường thử nghiệm và môi trường sản xuất nữa

#### 3.5.2.2. Dễ dàng thiết lập và triển khai cơ sở dữ liệu

Docker cung cấp cách dễ dàng để thiết lập và triển khai các dịch vụ cơ sở dữ liệu như MySQL, PostgreSQL, Redis, và nhiều dịch vụ khác mà không cần phải thực hiện các bước cấu hình phức tạp. Thay vì phải cài đặt từng cơ sở dữ liệu riêng biệt, chỉ cần chạy các Docker Images sẵn có, giúp tiết kiệm thời gian và giảm thiểu sai sót. Điều này cũng cho phép dễ dàng chạy nhiều phiên bản của cùng một cơ sở dữ liệu trên các container khác nhau mà không xảy ra bất kỳ xung đột nào giữa chúng. Docker giúp việc triển khai cơ sở dữ liệu trở nên nhanh chóng và thuận tiện, đặc biệt trong môi trường phát triển và thử nghiệm.

#### 3.5.2.3. Quản lý phụ thuộc dễ dàng

Với Docker, có thể định nghĩa tất cả các phụ thuộc của dự án trong tệp cấu hình docker-compose.yml. Các phụ thuộc này có thể là cơ sở dữ liệu, Redis, Kafka, hoặc bất kỳ dịch vụ nào mà ứng dụng cần. Điều này giúp đơn giản hóa việc thiết lập môi trường làm việc cho các thành viên trong nhóm, khi họ chỉ cần chạy một lệnh để tự động thiết lập tất cả các dịch vụ cần thiết mà không cần phải cài đặt và cấu hình thủ công từng phần. Việc này giúp tiết kiệm thời gian và đảm bảo môi trường làm việc của tất cả các thành viên trong nhóm là nhất quán.

#### 3.5.2.4. Dễ dàng mở rộng và tái sử dụng

Docker containers rất linh hoạt và dễ dàng mở rộng (scale) khi cần thiết. Có thể dễ dàng nhân bản các containers để phục vụ các yêu cầu tăng trưởng của ứng dụng mà không làm gián đoạn các hoạt động khác. Docker cũng cho phép lưu trữ và chia sẻ Docker Images, giúp tái sử dụng chúng trong các dự án khác. Có thể lưu trữ các Docker Images trong Docker Hub hoặc các kho lưu trữ Docker khác để sử dụng lại trong tương lai mà không phải mất công tạo lại từ đầu

#### 3.5.2.5. Tiết kiệm tài nguyên

Docker containers rất nhẹ và hiệu quả về mặt tài nguyên. Vì Docker containers chia sẻ nhân (kernel) của hệ điều hành thay vì chạy một hệ điều hành riêng biệt như các máy ảo (VMs), chúng sử dụng ít tài nguyên hơn và khởi động nhanh hơn. Docker chỉ chứa những gì cần thiết cho ứng dụng, không phải toàn bộ hệ điều hành, giúp giảm thiểu tài nguyên sử dụng và chi phí vận hành. Điều này rất hữu ích khi cần chạy nhiều ứng dụng trên cùng một máy chủ mà không gây ra tình trạng lãng phí tài nguyên.

#### 3.5.2.6. Cộng đồng lớn và nhiều tài nguyên

Docker có một cộng đồng rất lớn và tài nguyên phong phú, bao gồm Docker Hub, một kho lưu trữ chứa hàng ngàn Docker Images sẵn có cho các dịch vụ phổ biến như PostgreSQL, MongoDB, Redis, và nhiều công cụ khác. Điều này giúp dễ dàng tìm thấy và sử dụng các Docker Images cần thiết mà không cần phải tự tạo lại từ đầu. Với sự hỗ trợ từ cộng đồng và tài liệu phong phú, việc triển khai và quản lý Docker trở nên dễ dàng và hiệu quả hơn bao giờ hết.

# CHƯƠNG 4: THIẾT KẾT, TRIỂN KHAI VÀ ĐÁNH GIÁ HỆ THỐNG

## 4.1. Thiết kế kiến trúc

### 4.1.1. Lựa chọn kiến trúc phần mềm

**Clean Architecture** là một cách tiếp cận trong thiết kế phần mềm, được tạo ra với mục tiêu giúp các hệ thống dễ dàng bảo trì, mở rộng và kiểm thử. Ý tưởng chính của nó là tách biệt rõ ràng các thành phần khác nhau trong ứng dụng để giảm sự phụ thuộc giữa chúng. Điều này không chỉ giúp cải thiện khả năng tái sử dụng mã mà còn làm tăng tính ổn định khi hệ thống phát triển hoặc thay đổi.

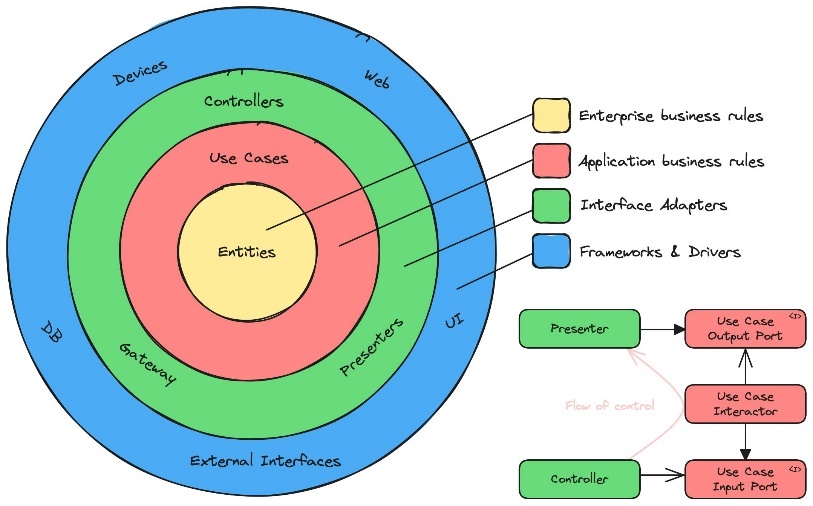
Cốt lõi của Clean Architecture nằm ở việc tổ chức mã nguồn theo các vòng tròn đồng tâm, với mỗi vòng đại diện cho một tầng chức năng riêng biệt. Vòng trong cùng là **Entities**, nơi chứa các quy tắc và logic nghiệp vụ cốt lõi. Đây là phần không bị ảnh hưởng bởi các thay đổi từ bên ngoài, như giao diện hay cơ sở dữ liệu. Ví dụ, nếu có một ứng dụng quản lý người dùng, tầng này sẽ định nghĩa các đối tượng như "User" và các quy tắc cơ bản liên quan.

Bao quanh tầng Entities là tầng **Use Cases**, nơi tập trung xử lý các tác vụ cụ thể mà hệ thống thực hiện, chẳng hạn như "Đăng ký người dùng mới" hoặc "Đặt hàng". Đây là nơi logic ứng dụng được thực thi, sử dụng các Entities để hoàn thành công việc mà người dùng yêu cầu.

Tiếp theo là tầng **Interface Adapters**, chịu trách nhiệm chuyển đổi dữ liệu giữa các tầng. Ví dụ, nếu dữ liệu từ cơ sở dữ liệu có định dạng khác với dữ liệu mà ứng dụng cần, tầng này sẽ thực hiện việc chuyển đổi để đảm bảo mọi thứ hoạt động trơn tru.

Cuối cùng, vòng ngoài cùng là tầng **Frameworks and Drivers**, nơi chứa các thành phần liên quan đến cơ sở hạ tầng như cơ sở dữ liệu, API, hoặc framework. Đây là tầng dễ thay đổi nhất mà không ảnh hưởng đến các tầng bên trong. Nếu cần chuyển từ MySQL sang MongoDB hoặc thay đổi framework, chỉ cần chỉnh sửa ở tầng này mà không phải động đến logic nghiệp vụ.

Điều đặc biệt trong Clean Architecture là mọi sự phụ thuộc phải hướng vào bên trong, về phía các tầng trung tâm. Điều này nghĩa là các tầng bên trong không biết gì về các tầng bên ngoài.



Hình 4 - 1: Clean architecture

### 4.1.2. Thiết kế tổng quan

#### 4.1.2.1. Thiết kế tổng quan phía backend

A blue screen with many folders

Description automatically generated

Hình 4 - 2: Thiết kế tổng quan backend

Thiết kế tổng quan các gói ở phía backend được mô tả như sau:

• Configs package: Chứa các cấu hình như môi trường, cấu hình kết nối cơ sở dữ liệu. • Controllers package: Xử lý các request từ phía client và trả về response sau khi xử lý

• Dtos package: Chứa các đối tượng truyền dữ liệu để định nghĩa cấu trúc dữ liệu cho request/response, thực hiện nhiệm vụ xác thực.

• Middleware package: Chứa các lớp hoặc hàm được sử dụng để xử lý các request/response trước khi đến Controller.

• Interface package: Chứa các interface để định nghĩa các hàm được sử dụng trong ứng dụng.

• Model package: Chứa các model mô tả cấu trúc của dữ liệu lưu trữ trong cơ sở dữ liệu.

• Repository package: Chứa các lớp hoặc hàm chịu trách nhiệm tương tác với cơ sở dữ liệu.

• Util package: Chứa các hàm hoặc module tiện ích dùng chung trong toàn bộ ứng dụng

• Routes package: Định nghĩa và quản lý các đường dẫn (route) trong ứng dụng

• Services package: Nhận dữ liệu từ controller và thực hiện các thao tác với repository hoặc các service khác.

#### 4.1.2.2. Thiết kế tổng quan phía frontend

A diagram of a computer system

Description automatically generated

Hình 4 - 3: Thiết kế tổng quan phía frontend

Thiết kế tổng quan các gói ở phía frontend (hình 4.3) được mô tả như sau:

• Component package: Chứa các component dùng chung.

• App package: Là phần cốt lõi của ứng dụng, chứa các file khởi tạo ứng dụng và logic điều hướng (routing).

• Lib package: Chứa các thư viện và module tự xây dựng được dùng lại trong nhiều nơi trong dự án.

• Public package: Chứa các file tĩnh (static files) có thể truy cập công khai trong ứng dụng

• Util package: Chứa các file liên quan đến quản lý state thông qua Redux, bao gồm actions và reducers

## 4.2. Thiết kế chi tiết

### 4.2.1. Thiết kế giao diện

Hệ thống hướng tới việc tạo ra giao diện được thiết kế để tương thích tốt với các laptop phổ thông, đảm bảo rằng giao diện hệ thống sẽ hiển thị đẹp và rõ ràng trên

hầu hết các thiết bị mà người dùng sử dụng.

Để đạt được sự thống nhất và chuẩn hóa trong thiết kế giao diện, các nguyên tắc

đã được áp dụng:

• Các nút và điều khiển được thiết kế với kích thước phù hợp để dễ dàng nhấn

và tương tác trên bàn phím và touchpad của laptop. Màu sắc và biên viền được

sử dụng để làm nổi bật các nút chức năng và các điều khiển, các vùng thông

tin quan trọng được làm sáng màu giúp người dùng nhận diện dễ dàng.

• Thông điệp phản hồi được hiển thị ở góc màn hình với kích thước đủ lớn để

người dùng có thể nhận thấy và đọc thông tin trả về từ hệ thống một cách dễ

dàng.

• Sử dụng màu sắc hài hòa và tương phản tốt giữa nền và văn bản để đảm bảo

độ đọc rõ ràng và thuận tiện.

#### 4.2.1.1. Giao diện chung

**Hình 4.4** mô tả giao diện chung cho các vai trò với phần trên cùng chứa logo, avatar. Cột điều hướng ở phía bên trái hiển thị các tùy chọn quản lý lịch học và các chức năng khác, trong khi đó cột ở bên phải chứa các thông tin chi tiết, nội dung chính của hệ thống.

A white rectangular object with black lines

Description automatically generated

Hình 4 - 4: Giao diện chung

#### 4.2.1.2. Giao diện nhắn tin

**Hình 4.5** minh họa giao diện nhắn tin với bố cục sau: Phần tin nhắn được chia làm 2 phần, bên trái là danh sách các cuộc trò chuyện, bên phải là nội dung cuộc trò chuyện

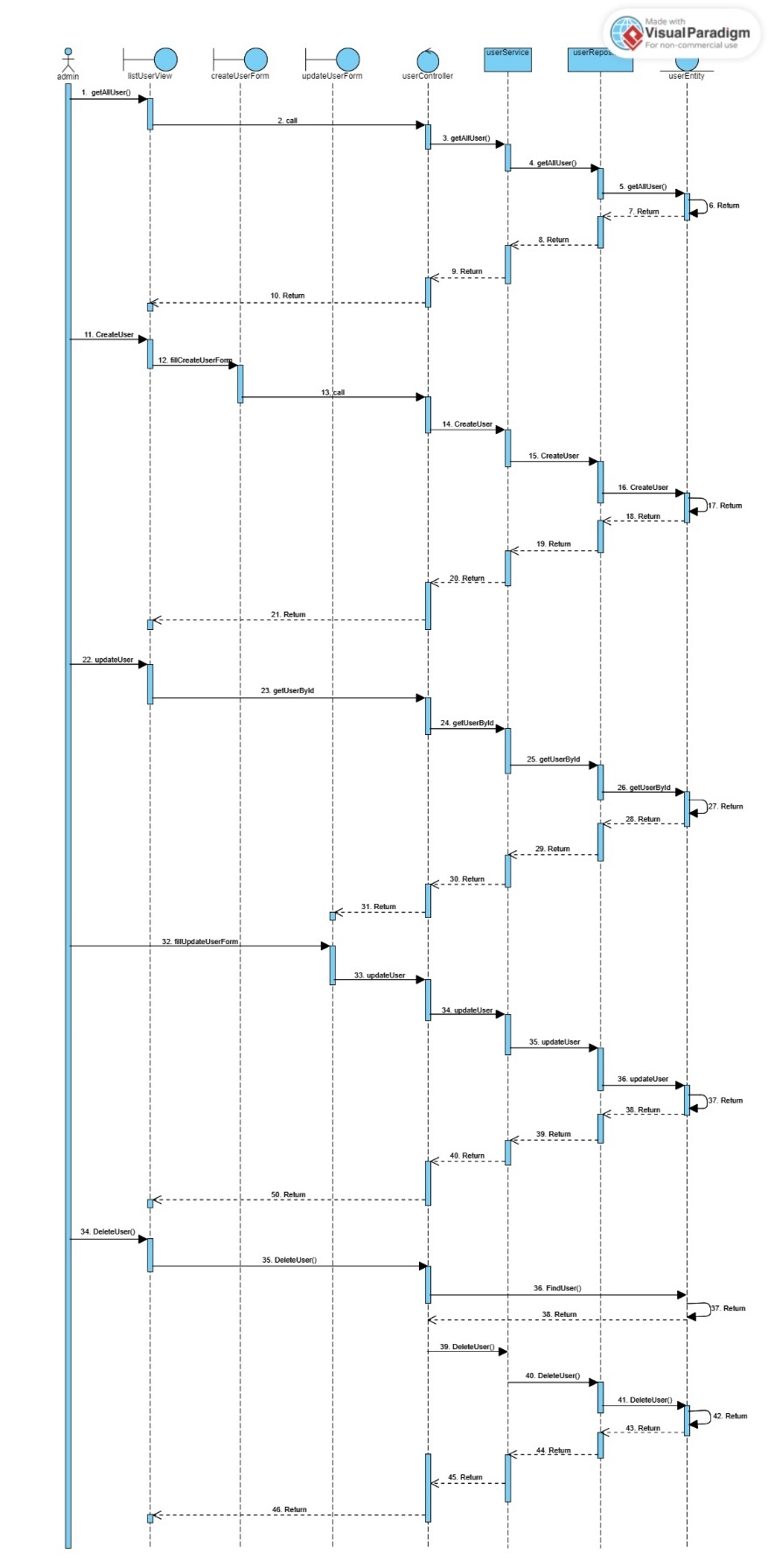
A white rectangular object with black lines

Description automatically generated

Hình 4 - 5: Giao diện nhắn tin

### 4.2.2. Thiết kế lớp

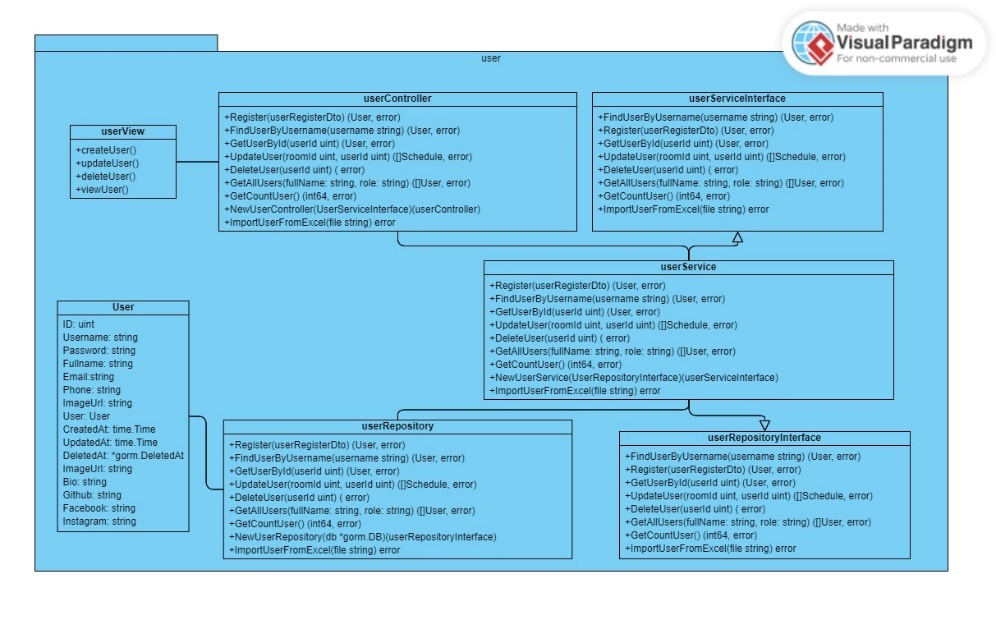
#### 4.2.2.1. Chức năng quản lý tài khoản



Hình 4 - 6: Biểu đồ trình tự quản lý tài khoản

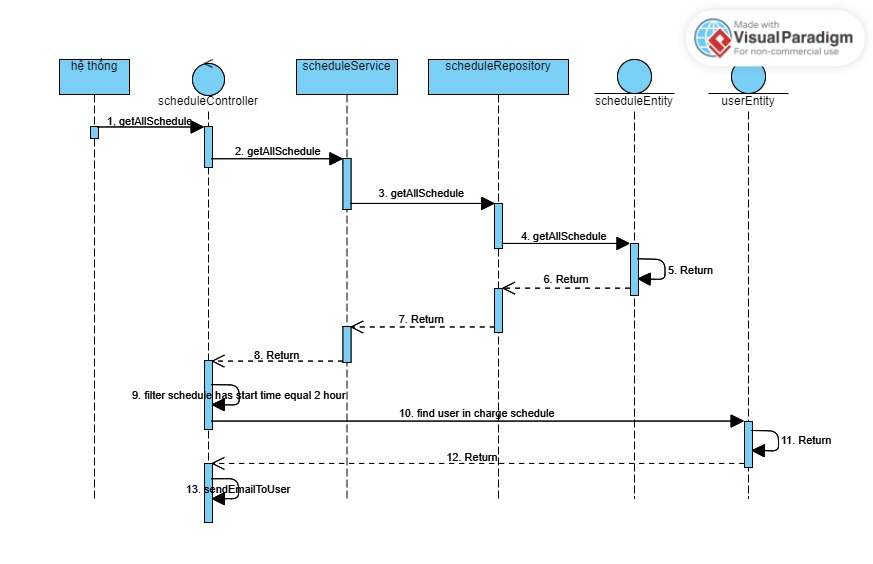
**Hình 4 – 6** minh họa biểu đồ trình tự quản lý tài khoản với các chức năng: xem danh sách, thêm mới, chỉnh sửa, và xóa tài khoản. Biểu đồ thể hiện luồng tương tác giữa người dùng, hệ thống, và cơ sở dữ liệu để xử lý yêu cầu.

**Hình 4 – 7** thể hiện sơ đồ lớp thiết kế chức năng quản lý tài khoản



Hình 4 - 7: Lớp thiết kế chức năng quản lý tài khoản

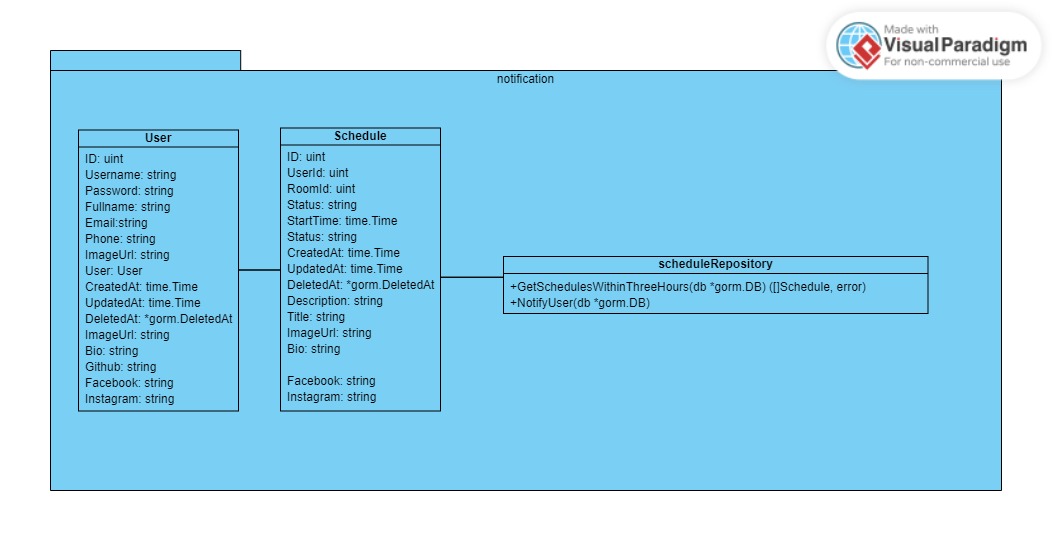
#### 4.2.2.2. Chức năng thông báo lịch học đến email của giảng viên



Hình 4 - 8: Biểu đồ trình tự chức năng gửi thông báo lịch học đến email của giảng viên

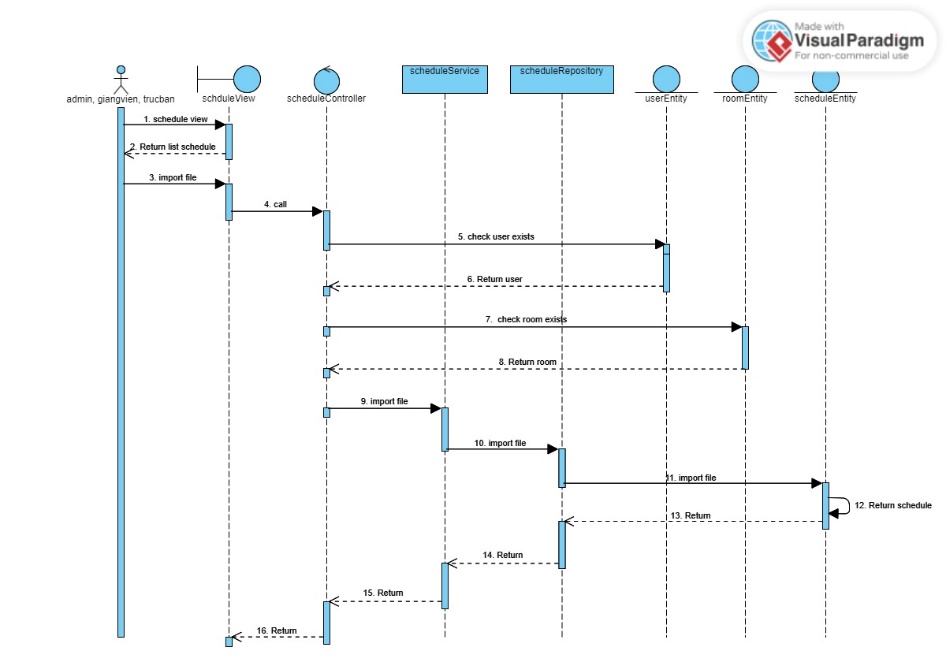
**Hình 4 – 8** mô tả biểu đồ trình tự chức năng gửi thông báo lịch học đến email của giảng viên, mỗi khi chương trình được khởi động, mỗi phút hệ thống sẽ tự động lọc tất cả các lịch học đã tồn tại trong hệ thống, nếu lịch học có thời gian bắt đầu còn lại 2 tiếng thì hệ thống sẽ gửi mail thông báo cho giảng viên phụ trách lịch học đó

**Hình 4 – 9** thể hiện lớp thiết kế chức năng thông báo



Hình 4 - 9: Lớp thiết kế chức năng thông báo

#### 4.2.2.3. Import file



Hình 4 - 10: Biểu đồ tuần tự chức năng import file

**Hình 4 – 10** mô tả chức năng import file để tạo lịch học, admin, giảng viên, trực ban có thể sử dụng chức năng này, người dùng có thể tạo nhiều lịch học đồng thời bằng cách sử dụng chức năng import file excel

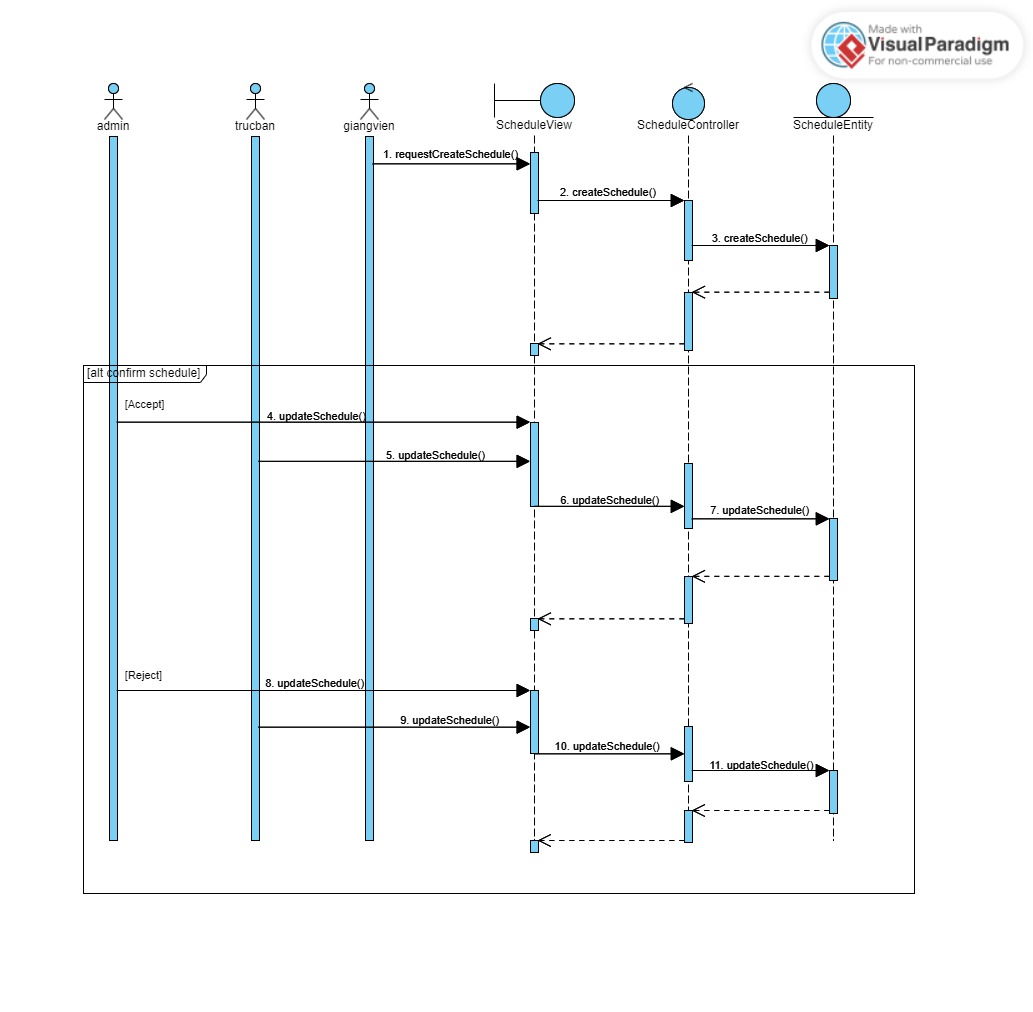
**Hình 4 – 11** thể hiện lớp thiết kế chức năng import file

A blue screen with text on it

Description automatically generated

Hình 4 - 11: Lớp thiết kế chức năng import file

#### 4.2.2.4. Tạo lịch học



Hình 4 - 12: Sequence diagram chức năng đặt lịch học

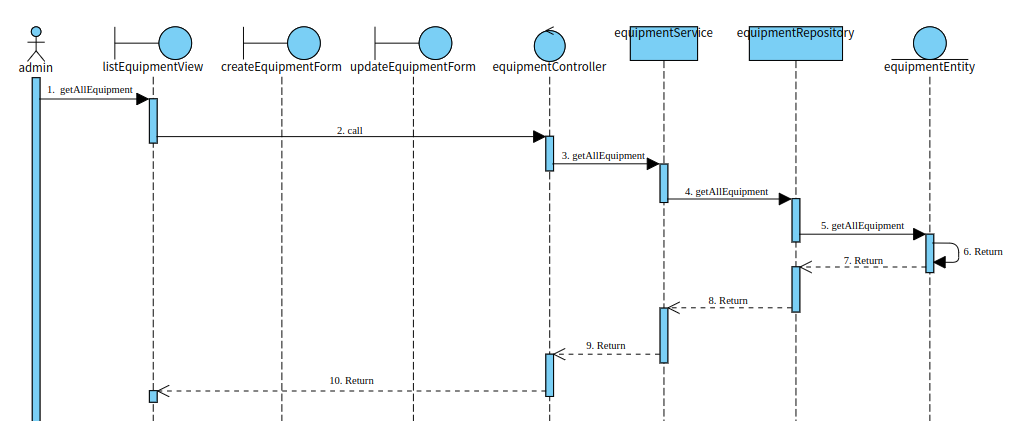
**Hình 4 - 13** trình bày biểu đồ tuần tự đặt lịch học. Sau khi giảng viên gửi yêu cầu đặt lịch học, admin hoặc trực ban có thể từ chối hoặc chấp nhận lịch học của giảng viên

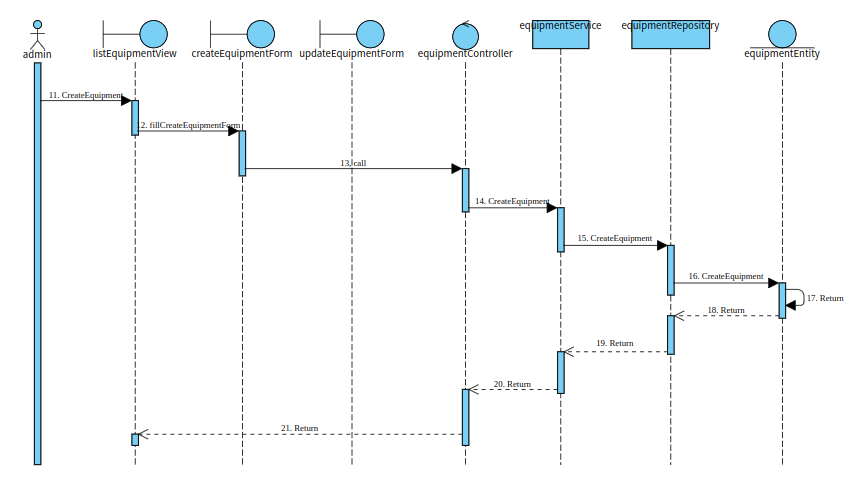
A blue screen with text on it

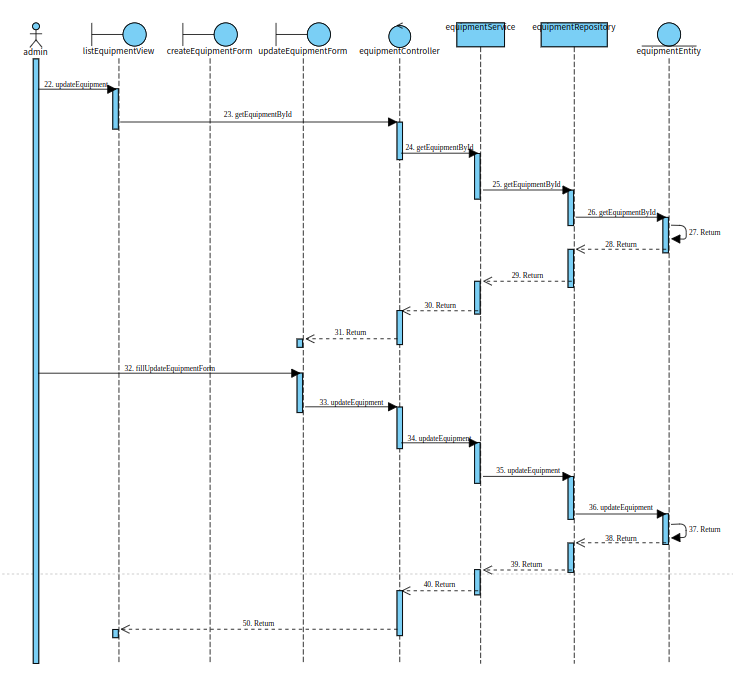
Description automatically generated

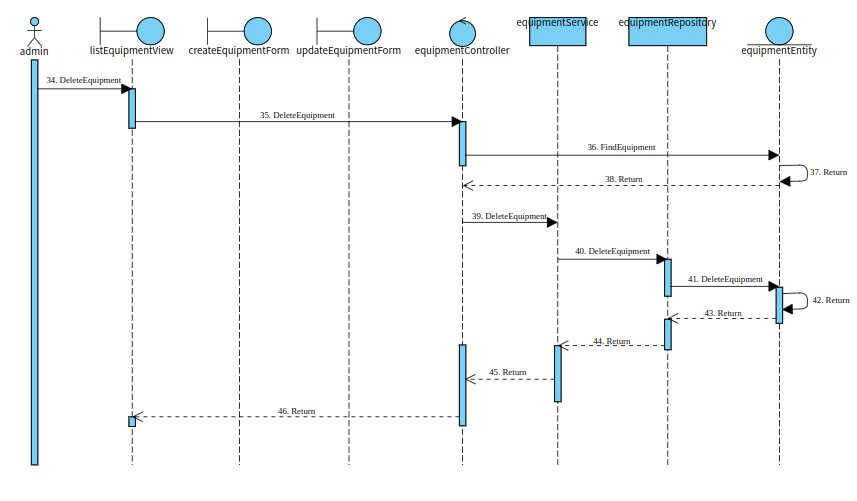
Hình 4 - 13: Lớp thiết kế chức năng đặt lịch

### **4.2.2.5. Chức năng quản lý thiết bị**



****





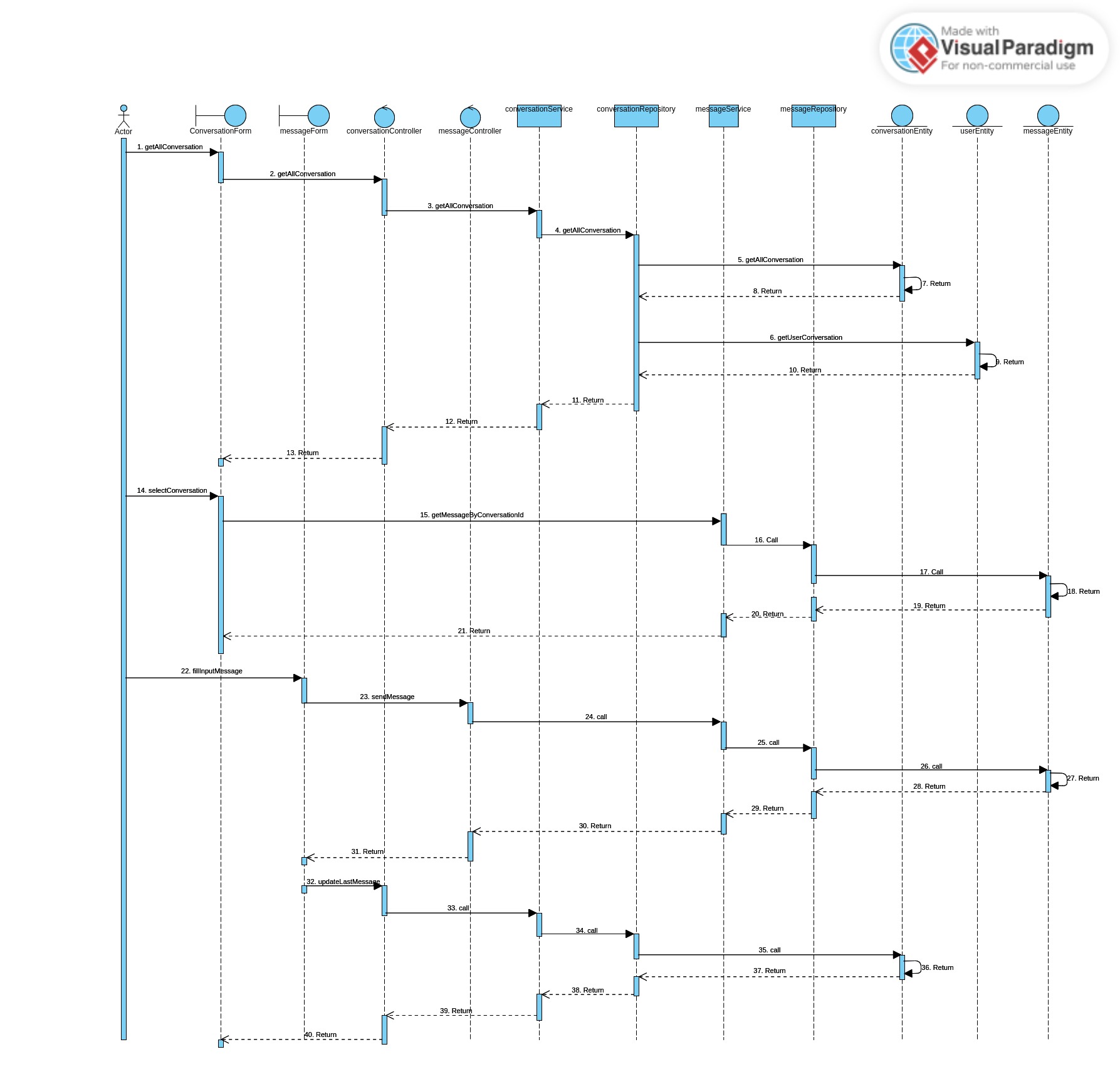
**4.2.2.6. Chức năng quản lý phòng**

**4.2.2.7. Chức năng quản lý quyền truy cập**

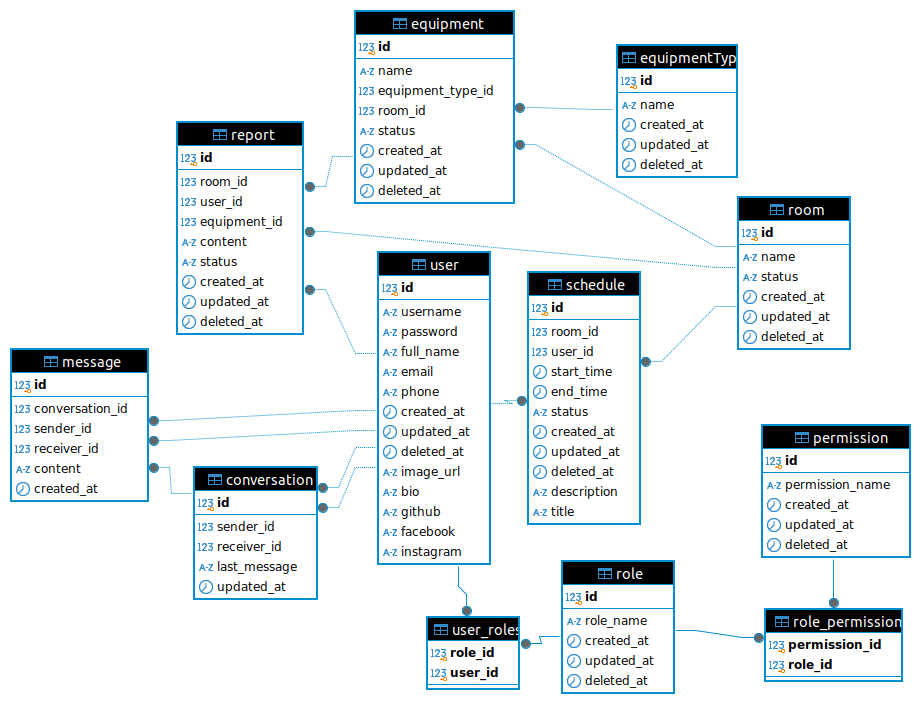
**4.2.2.8. Chức năng quản lý báo cáo**

**4.2.2.9. Chức năng quản lý vai trò**

**4.2.2.10. Chức năng gửi tin nhắn**



### 4.2.3. Thiết kế cơ sở dữ liệu



Hình 4 - 14: Thiết kế cơ sở dữ liệu

Ý nghĩa của các bảng

- Message: Thông tin các tin nhắn

- Report: Thông tin các báo cáo

- Conversation: Thông tin các cuộc trò chuyện

- Equipment: Thông tin các thiết bị

- User: Thông tin người dung

- User\_role: Liên kết giữa bảng user và role

- Role: Thông tin các vai trò

- Schedule: Thông tin các lịch học

- EquipmentType: Thông tin các loại thiết bị

- Room: Thông tin các phòng

- Permission: Thông tin quyền

- Role\_permission: Liên kết giữa bảng role và permission

## 4.3. Xây dựng ứng dụng

### 4.3.1. Thư viện và công cụ sử dụng

Bảng 4 - 1: Thư viện và công cụ sử dụng

|  |  |  |
| --- | --- | --- |
| Mục đích | Công cụ | Địa chỉ URL |
| IDE lập trình | Visual Studio Code 64 bit | https://code.visualstudio.com/ |
| Kiểm thử | Postman | https://www.postman.com/ |
| Cơ sở dữ liệu | PostgreSQL | https://www.postgresql.org/ |
| Ứng dụng vẽ sơ đồ | Visual paradigm | https://www.visual-paradigm.com/ |
| Thư viện UI | Shadcn | https://ui.shadcn.com/ |
| Thư viện CSS | TailwindCSS | https://tailwindcss.com/ |
| Framework backend | Gin | https://gin-gonic.com/ |
| Framework frontend | NextJs | https://nextjs.org/ |

### 4.3.2. Kết quả đạt được

Hệ thống quản lý phòng thí nghiệm thực hành đã đáp ứng được yêu cầu cơ bản của người dùng và việc quản lý, đặt lịch học. Người dùng có thể: (i) đặt lịch học, (ii) nhắn tin thời gian thực, (iii) tạo báo cáo. Hệ thống có các chức năng: (i) quản lý lịch đặt, (ii) quản lý tài khoản, (iii) quản lý thiết bị, (iv) xem thống kê, (v) quản lý tài khoản, (vi) thông báo lịch học, (vii) quản lý phòng, (viii) quản lý quyên truy cập.

### 4.3.3. Minh họa các chức năng chính

A screenshot of a computer

Description automatically generated

Hình 4 - 15: Giao diện cập nhật thông tin

A screenshot of a computer

Description automatically generated

A screenshot of a graph

Description automatically generated

Hình 4 - 16: Giao diện thống kê

A screenshot of a calendar

Description automatically generated

Hình 4 - 17: Giao diện đặt lịch

A screenshot of a computer

Description automatically generated

Hình 4 - 18: Giao diện quản lý lịch học

A screenshot of a computer

Description automatically generated

Hình 4 - 19: Giao diện nhắn tin

### 4.4. Kiểm thử

Bảng 4 - 2: Bảng testcase

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| ID | Trường hợp kiểm thử | Quy trình kiểm thử | Kết quả mong muốn | Kết quả thực hiện | Kết luận |
| 1 | Tạo lịch học | 1. Vào màn Calendar  2. Chọn phòng học  3. Chọn thời gian | Hiển thị form nhập thông tin lịch học | Hiển thị thông báo thành công và tạo được lịch học | PASS |
| 2 | Cập nhật trạng thái lịch học | 1. Vào màn Schedule  2. Chọn lịch học | Hiển thị thông tin lịch học | Hiển thị thông báo cập nhật lịch học thành công | PASS |
| 3 | Lọc lịch học theo phòng | 1. Vào màn Calendar  2. Chọn phòng học | Hiển thị lịch học | Hiển thị lịch học | PASS |
| 4 | Thay đổi thông tin cá nhân | 1. Vào màn profile  2. Chọn edit | Hiển thị thông tin cá nhân | Hiển thị thông báo cập nhật thông tin cá nhân thành công | PASS |
| 5 | Tạo báo cáo | 1. Vào màn Report  2. Chọn tạo báo cáo | Hiển thị form nhập thông tin | Hiển thị thông báo tạo báo cáo thành công | PASS |
| 6 | Tạo nhiều tài khoản bằng cách import file excel | 1. Vào màn tài khoản  2. Chọn import  3. Chọn file excel đã tồn tại trong máy | Hiển thị thông báo import thành công, hiển thị danh sách tài khoản | Hiển thị thông báo import thành công, hiển thị danh sách tài khoản | PASS |
| 7 | Tạo nhiều lịch học bằng cách import file excel | 1. Chọn lịch học  2. Chọn import  3. Chọn file excel đã tồn tại trong máy | Hiển thị thông báo import thành công, hiển thị danh sách lịch học | Hiển thị thông báo import thành công, hiển thị danh sách lịch học | PASS |
| 8 | Gửi thông báo qua email cho giảng viên khi thời gian bắt đầu môn học còn 2 tiếng | 1. Vào màn Calendar  2. Chọn phòng học  3. Chọn thời gian | Hiển thị form nhập thông tin lịch học, thông báo về email nếu thời gian bắt đầu lịch học còn 2 tiếng | Hiển thị form nhập thông tin lịch học, thông báo về email nếu thời gian bắt đầu lịch học còn 2 tiếng | PASS |

## 4.5. Triển khai

Ứng dụng đang được thực thi cục bộ trên môi trường localhost, với các tham số máy chủ thử nghiệm được cho ở bảng 4.9

Bảng 4 - 3: Triển khai

|  |  |
| --- | --- |
| Thông số kỹ thuật | Chi tiết |
| RAM | 16Gb |
| Hệ điều hành | Window 10 |
| Dung lượng phần cứng | SSD 256Gb |
| CPU | Intel Core i5 9300H |

# CHƯƠNG 5: KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

## 5.1. Kết luận

Trong suốt quá trình thực hiện ĐATN, em đã phát triển một hệ thống quản lý phòng thí nghiệm thực hành, mang lại sự tiện lợi cho cả giảng viên, trực ban, giám đốc học viện. Hệ thống này cho phép giảng viên đặt lịch học trực tuyến, quản lý thông tin cá nhân. Đối với học viện, hệ thống cung cấp các công cụ để quản lý lịch học, quản lý phòng học, quản lý tài khoản, quản lý thiết bị, quản lý báo cáo, xem thống kê các phòng, thiết bị, báo cáo, tần suất sử dụng. Ngoài ra còn mở rộng thêm tính năng nhắn tin giúp việc trao đổi thông tin thuận tiện hơn. Trong suốt quá trình thực hiện ĐATN, mặc dù đã có những cố gắng nhất có thể nhưng kinh nghiệm thiếu sót và việc hoàn thiện một hệ thống hoàn chỉnh, toàn diện đáp ứng nhiều yêu cầu cần bỏ ra nhiều công sức hơn. Hệ thống còn một số hạn chế như chưa phát triển các chức năng liên quan đến thông báo lịch học,... Từ quá trình thực hiện ĐATN, em đã rút ra được những bài học quý báu bao gồm việc lập kế hoạch chi tiết, quản lý thời gian hiệu quả và sự cần thiết của việc kiểm tra và đánh giá hệ thống một cách liên tục.

## 5.2. Hướng phát triển

Để triển khai sản phẩm trên thực tế và nâng cao tính ứng dụng, hệ thống cần được cải thiện và bổ sung các chức năng mới để đáp ứng các nhu cầu và mở rộng phạm vi dịch vụ. Đầu tiên, phải phát triển thêm nhiều chức năng ở giao diện quản lý cho quản trị viên để họ có thể dễ dàng quản lý nhiều thông tin hơn, và các hoạt động khác trong hệ thống. Ngoài ra, hệ thống cần cải tiến tính năng phản hồi và đánh giá từ người dùng để thu thập ý kiến và cải thiện chất lượng dịch vụ. Cuối cùng, để mở rộng phạm vi dịch vụ, cần cung cấp các tính năng gọi thoại, tạo lớp học, cho phép sinh viên có thể đăng nhập vào hệ thống. Đây là những yếu tố quan trọng giúp đáp ứng đầy đủ nhu cầu trong lĩnh vực quản lý phòng thí nghiệm thực hành

# PHỤ LỤC I

**Mã nguồn chương trình**

Backend: <https://github.com/TuqL3/Graduate_be>

Frontend: https://github.com/TuqL3/Graduate\_FE

# TÀI LIỆU THAM KHẢO

1. Learn Nextjs. [Online]. Available: <https://nextjs.org/learn>
2. Gin. [Online]. Available : <https://gin-gonic.com/>
3. NextJs Image Upload To Cloudinary. [Online]. Available: <https://cloudinary.com/blog/uploading-images-videos-next-js-server-side-multer-xata>
4. Setup Gin Websocket. [Online]. Available: <https://pkg.go.dev/github.com/gorilla/websocket>
5. Setup Nextjs Websocket. [Online]. Available: <https://socket.io/how-to/use-with-nextjs>