|  |
| --- |
| BỘ GIÁO DỤC VÀ ĐÀO TẠO  TRƯỜNG ĐẠI HỌC NGUYỄN TẤT THÀNH  KHOA CÔNG NGHỆ THÔNG TIN  Description: C:\Documents and Settings\Administrator\Desktop\logo dai hoc_khong nen.png  BÁO CÁO THỰC TẬP TỐT NGHIỆP  TRANG WEB QUẢN LÝ  CHUNG CƯ TẠI MỸ  Giảng viên hướng dẫn: BÙI TIẾN ĐỨC  Sinh viên thực hiện: NGUYỄN LÊ QUỐC BẢO  MSSV: 2100004053  Chuyên ngành: Trí Tuệ Nhân Tạo  Đơn vị thực tập: Công ty TNHH Công nghệ số Việt Trí Đạo  Khóa: 2021  Tp.HCM, tháng 12 năm 2024 |
| BỘ GIÁO DỤC VÀ ĐÀO TẠO  TRƯỜNG ĐẠI HỌC NGUYỄN TẤT THÀNH  KHOA CÔNG NGHỆ THÔNG TIN  Description: C:\Documents and Settings\Administrator\Desktop\logo dai hoc_khong nen.png  BÁO CÁO THỰC TẬP TỐT NGHIỆP  TRANG WEB QUẢN LÝ  CHUNG CƯ TẠI MỸ  Giảng viên hướng dẫn: BÙI TIẾN ĐỨC  Sinh viên thực hiện: NGUYỄN LÊ QUỐC BẢO  MSSV: 2100004053  Chuyên ngành: Trí tuệ nhân tạo  Đơn vị thực tập: Công ty TNHH Công nghệ số Việt Trí Đạo  Khóa: 2021  Tp.HCM, tháng 12 năm 2024 |

# LỜI CẢM ƠN

Trong suốt thời gian học tập và làm bài tiểu luận này, nhóm chúng em đã nhận được sự hướng dẫn, giúp đỡ quý báu của các thầy cô, các anh chị và các bạn. Với quý trọng và biết ơn sâu sắc, chúng em xin được bày tỏ lới cảm ơn chân thành tới:

Ban giám hiệu, Phòng đào tạo khoa CNTT trường đại học Nguyễn Tất Thành đã tạo mọi điều kiện thuận lợi giúp đỡ chúng em trong quá trình học tập và hoàn thành báo cáo.

Xin gửi lời cảm ơn tới những người bạn trong nhóm 2 của kì thực tập, các anh chị em trong nhóm lập trình đã giúp đỡ chúng em trong những lúc khó khăn.

Xin chân thành cảm ơn bố mẹ đã luôn ở bên cạnh động viên con trong việc học tập và làm việc để hoàn thành bài tiểu luận.

Cuối cùng chúng em xin được gửi lời cảm ơn đến thầy Bùi Tiến Đức– giảng viên bộ môn “Thực Tập Tốt Nghiệp“ trong khoa Công nghệ thông tin vì đã luôn dành sự quan tâm, hướng dẫn và giúp đỡ tận tình cho chúng em bên cạnh đó còn giúp đỡ các chúng em tìm công ty thực tập để hoàn thành báo cáo này.

# LỜI MỞ ĐẦU

Trong bối cảnh đô thị hóa nhanh chóng, nhu cầu quản lý và vận hành các tòa nhà chung cư ngày càng trở nên phức tạp và đòi hỏi tính chuyên nghiệp cao. Các phương pháp quản lý truyền thống không còn đáp ứng được tốc độ phát triển và yêu cầu hiện đại hóa. Việc áp dụng công nghệ thông tin vào quản lý chung cư không chỉ giúp nâng cao hiệu quả hoạt động mà còn tối ưu hóa trải nghiệm của cư dân và ban quản lý.

Dự án **Web Quản Lý Chung Cư** được xây dựng nhằm cung cấp một giải pháp toàn diện, hiện đại, và dễ sử dụng. Hệ thống tập trung vào việc tự động hóa quy trình quản lý, hỗ trợ việc lưu trữ thông tin, quản lý cư dân, quản lý tài sản, xử lý phản hồi và yêu cầu bảo trì. Ngoài ra, trang web còn tạo cầu nối giữa cư dân và ban quản lý, giúp tăng cường sự minh bạch và tiện lợi trong giao tiếp.

Dự án không chỉ là một sản phẩm công nghệ mà còn là một bước tiến trong việc hiện đại hóa ngành quản lý chung cư, với mục tiêu mang lại giá trị thiết thực cho cả ban quản lý và cư dân.

# NHẬN XÉT CỦA GIẢNG VIÊN HƯỚNG DẪN

*TPHCM, Ngày …… tháng …… năm 2017*

Giáo viên hướng dẫn

(Ký tên, đóng dấu)

# NHẬN XÉT CỦA CƠ QUAN THỰC TẬP

*TPHCM, Ngày …… tháng …… năm 2017*

Xác nhận của đơn vị thực tập

(Ký tên, đóng dấu)

# MỤC LỤC

[LỜI CẢM ƠN 3](#_Toc185855748)

[LỜI MỞ ĐẦU 4](#_Toc185855749)

[NHẬN XÉT CỦA GIẢNG VIÊN HƯỚNG DẪN 5](#_Toc185855750)

[NHẬN XÉT CỦA CƠ QUAN THỰC TẬP 6](#_Toc185855751)

[MỤC LỤC 7](#_Toc185855752)

[GIỚI THIỆU ĐỀ TÀI NGHIÊN CỨU VÀ CÔNG TY THỰC TẬP 1](#_Toc185855753)

[1 Đề Tài Nghiên Cứu Và Lĩnh Vực 1](#_Toc185855754)

[1.1 Khái Quát Về Dự Án Và Kiến Trúc Tổng Quát 1](#_Toc185855755)

[1.2 Tại Sao Lại Chọn Dự Án Về Web 1](#_Toc185855756)

[1.3 Cảm Nhận Và Mong Muốn Khi Nhận Dự Án 2](#_Toc185855757)

[2 Công Ty Thực Tập 2](#_Toc185855758)

[2.1 Giới Thiệu Về Công Ty 2](#_Toc185855759)

[2.2 Quy Định Về Quá Trình Thực Tập Và Hình Thức Làm Việc 2](#_Toc185855760)

[2.3 Yêu Cầu Về Sản Phẩm 3](#_Toc185855761)

[2.4 Quy Mô Dự Án 3](#_Toc185855762)

[2.5 Giới Thiệu Thành Viên Và Cơ Cấu Tổ Chức 4](#_Toc185855763)

[2.6 Quy Định Về Dự Án 4](#_Toc185855764)

[Yêu Cầu Về Năng Lực Thành Viên 5](#_Toc185855765)

[Ví Trí Trong Dự Án 5](#_Toc185855766)

[CƠ SỞ LÝ LUẬN, LÝ THUYẾT VỀ DỰ ÁN 5](#_Toc185855767)

[1 Mô Tả Dự Án 5](#_Toc185855768)

[2 Các Vấn Đề Của Dự Án 6](#_Toc185855769)

[3 Cơ Sở Lý Thuyết Và Kiến Thức Cần Có 6](#_Toc185855770)

[3.1 Các kiến Thức Cơ Bản nhất Về Web 6](#_Toc185855771)

[3.2 Fontend, Backend, Ba, Tester 12](#_Toc185855772)

[3.3 React.js, Kiến Thức Nền Tảng, Nâng Cao 14](#_Toc185855773)

[3.4 Spring boot là gì? Kiến Thức Nền Tảng 16](#_Toc185855774)

[3.5 Kiến trúc Microservice và Monolithic ở Backend 18](#_Toc185855775)

[3.6 Kiến trúc Monolithic và Micro ở Fontend 19](#_Toc185855776)

[3.7 Kiến trúc Shared Database và Database Per Service 20](#_Toc185855777)

[TRIỂN KHAI XÂY DỰNG DỰ ÁN 22](#_Toc185855778)

[1 Tổng Thời Gian Làm Việc Và Phân Bố 22](#_Toc185855779)

[2 Thống Nhất Về Thư Viện Và Công Cụ Hổ Trợ 26](#_Toc185855780)

[3 Viết User Story 27](#_Toc185855781)

[4 Thiết Kế Giao Diện Dự Án 27](#_Toc185855782)

[5 Thiết Kế Cơ Sở Dữ Liệu Dự Án 48](#_Toc185855783)

[6 Quá Trình Sửa Chửa, Cải Thiện Giao Diện, Cơ Sở Dữ lIệu 57](#_Toc185855784)

[7 Viết Test Case 60](#_Toc185855785)

[8 Phân Chia Công Việc Và Lập Nhóm 60](#_Toc185855786)

[9 Công Việc Của Cá Nhân 61](#_Toc185855787)

[10 Khởi Tạo Backend với Spring Boot 62](#_Toc185855788)

[11 Tổng Các Api Và Chức Năng Với Mỗi Api 63](#_Toc185855789)

[12 Khởi Tạo Fontend Với Vite React.js 65](#_Toc185855790)

[13 Thời Gian Hoàn Thành Và Kiểm Tra Lỗi 71](#_Toc185855791)

[14 Gộp Mã Nguồn Của Các Nhóm 72](#_Toc185855792)

[15 Hoàn Thiện Dự Án Và Lưu Trữ 72](#_Toc185855793)

[KẾT QUẢ VÀ ĐÁNH GIÁ 72](#_Toc185855794)

[1 Kết Quả Mong Muốn 72](#_Toc185855795)

[2 Kết Quả Thực Tế 72](#_Toc185855796)

[3 Đánh Giá Về Công Việc 72](#_Toc185855797)

[4 Đánh Giá Về Nhóm 73](#_Toc185855798)

[KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN 74](#_Toc185855799)

[1 Kết Luận 74](#_Toc185855800)

[2 Các Phương Hướng Để Cải Thiện Dự Án 75](#_Toc185855801)

# DANH MỤC ẢNH

[Hình II‑1 Ví dụ cho không có và có css khi sử dụng các thẻ h1 và p 7](#_Toc185856164)

[Hình II‑2 Độ ưu tiên của css theo thứ tự từ dưới lên 8](#_Toc185856165)

[Hình II‑3 Độ ưu tiên css theo class 8](#_Toc185856166)

[Hình II‑4 Độ ưu tiên của css theo id 8](#_Toc185856167)

[Hình II‑5 Độ ưu tiên của css theo cách viết trực tiếp 9](#_Toc185856168)

[Hình II‑6 Độ ưu tiên css theo sự rõ ràng khi trỏ 9](#_Toc185856169)

[Hình II‑7 Độ ưu tiên css theo sự rõ ràng nhưng lại trỏ vào tag thay vì class 10](#_Toc185856170)

[Hình II‑8 Ví dụ cho bố trí code trong scss và được built ra css 10](#_Toc185856171)

[Hình III‑1 mail các công việc trong ngày đầu tiên 21](#_Toc185856172)

[Hình III‑2 mail từ ngày 2 đến ngày 6 của công ty 22](#_Toc185856173)

[Hình III‑3 mail công việc ngày 7, ngày 8 của công ty 22](#_Toc185856174)

[Hình III‑4 mail công việc ngày 9 - ngày 16 của công ty 23](#_Toc185856175)

[Hình III‑5 mail công việc ngày 17 - ngày 22 của công ty 24](#_Toc185856176)

[Hình III‑6 Sơ đồ chức năng tramg web 26](#_Toc185856177)

[Hình III‑7 Screen 1-5 27](#_Toc185856178)

[Hình III‑8 Screen 6-7 27](#_Toc185856179)

[Hình III‑9 Giao diện thông báo 29](#_Toc185856180)

[Hình III‑10 Giao diện khiếu nại 29](#_Toc185856181)

[Hình III‑11 Giao diện thanh toán 30](#_Toc185856182)

[Hình III‑12 Giao diện quản lý tài khoản 31](#_Toc185856183)

[Hình III‑13 Phần Payment 32](#_Toc185856184)

[Hình III‑14 Thêm income và gửi thông báo 33](#_Toc185856185)

[Hình III‑15 Giao diện outcome 33](#_Toc185856186)

[Hình III‑16 Giao diện thông tin người thuê 34](#_Toc185856187)

[Hình III‑17 Giao diện thông tin thanh toán và lịch sử thanh toán 35](#_Toc185856188)

[Hình III‑18 Giao diện quản lý nhân sự 36](#_Toc185856189)

[Hình III‑19 Giao diện thông tin tòa nhà 37](#_Toc185856190)

[Hình III‑20 Giao diện bảo trì thiết bị 38](#_Toc185856191)

[Hình III‑21 Giao diện quản lý yêu cầu sửa chữa 38](#_Toc185856192)

[Hình III‑22 Giao diện quản lý danh sách góp ý của người thuê 39](#_Toc185856193)

[Hình III‑23 Giao diện hợp đồng cho thuê 40](#_Toc185856194)

[Hình III‑24 Giao diện vi phạm hợp đồng 41](#_Toc185856195)

[Hình III‑25 Giao diện quản lý hợp đồng dịch vụ 41](#_Toc185856196)

[Hình III‑26 Giao diện hợp đồng tiện ích 42](#_Toc185856197)

[Hình III‑27 Giao diện hợp đồng nhà cung cấp 43](#_Toc185856198)

[Hình III‑28 Giao diện lịch sử thanh toán 43](#_Toc185856199)

[Hình III‑29 Giao diện hợp đồng nhân viên 44](#_Toc185856200)

[Hình III‑30 Giao diện của nhân viên kỹ thuật 45](#_Toc185856201)

[Hình III‑31 Giao diện của nhân viên xử lý khiếu nại 46](#_Toc185856202)

[Hình III‑32 bảng user\_, staff\_, role\_ trong cơ sở dữ liệu 47](#_Toc185856203)

[Hình III‑33 bảng contract\_lease\_, bảng utility\_contract\_, employment\_contract\_, contract\_service\_ trong cơ sở dữ liệu 49](#_Toc185856204)

[Hình III‑34 bảng imcome\_, outcome\_, type\_income\_, type\_outcome\_, repair\_history\_ trong cơ sở dữ liệu 50](#_Toc185856205)

[Hình III‑35 các bảng apartment\_, suppliers\_, service\_, equipment\_ và type\_equipment\_ trong cơ sở dữ liệu 51](#_Toc185856206)

[Hình III‑36 bảng rental\_request\_, schedule\_a\_home\_inspection\_ trong cơ sở dữ liệu 52](#_Toc185856207)

[Hình III‑37 bảng pyment\_ và payment\_method\_ trong cơ sở dữ liệu 53](#_Toc185856208)

[Hình III‑38 bảng technical\_maintenance\_, technical\_repair\_, maintenance\_repair\_ trong cơ sở dữ liệu 54](#_Toc185856209)

[Hình III‑39 bảng complaints\_ và notification\_staff\_ trong cơ sơ dữ liệu 55](#_Toc185856210)

[Hình III‑40 ví dụ cho màn hình 1 trên figma tương ứng với chức năng 1 trong user story 56](#_Toc185856211)

[Hình III‑41 phân công công việc sau buổi họp Sprint plan của leader(nhóm trưởng) 59](#_Toc185856212)

[Hình III‑42 cài đặt thư viện trong dapedency 60](#_Toc185856213)

[Hình III‑43 file entity và reposity của các bảng sẽ dùng trong dự án 61](#_Toc185856214)

[Hình III‑44 ví dụ cho một api trả về 62](#_Toc185856215)

[Hình III‑45 trang quản lý người dùng và thanh toán với chức năng tìm kiếm 63](#_Toc185856216)

[Hình III‑46 chức năng xuất thành file excel của trang quản lý người dùng và thanh toán 64](#_Toc185856217)

[Hình III‑47 giao diện tạo, xóa, cập nhật thông tin người dùng 64](#_Toc185856218)

[Hình III‑48 giao diện xóa, sửa thông tin thanh toán 65](#_Toc185856219)

[Hình III‑49 giao diện quản lý nhân viên với chức năng tìm kiếm 65](#_Toc185856220)

[Hình III‑50 giao diện chức năng thêm, xóa, cập nhật nhân viên 66](#_Toc185856221)

[Hình III‑51 trang quản lý phản hồi người dùng với chức năng tìm kiếm 67](#_Toc185856222)

[Hình III‑52 trang chi tiết phản hồi với chức năng phân công nhân viên và cập nhật trạng thái phản hồi 68](#_Toc185856223)

# GIỚI THIỆU ĐỀ TÀI NGHIÊN CỨU VÀ CÔNG TY THỰC TẬP

## Đề Tài Nghiên Cứu Và Lĩnh Vực

### Khái Quát Về Dự Án Và Kiến Trúc Tổng Quát

Đây là một dự án về web, cụ thể là một web site quản lý chung cứ với các chức năng chính xoay quanh 3 hướng chính. Ngưới dùng thấp nhất là User chính là các công dân của chung cư và cấp tiếp theo là nhân viên trong chung cư bao gồm hỗ trợ sữa chửa bảo trì thiết bị, quản lý giấy tờ, phản hồi,…. Cuối cùng là Admin cũng chính là quyền hạn cao nhất là người điều hành cao nhất của chung cư.

Dự án được xây dựng là một web api với kiến trúc micro-service.

### Tại Sao Lại Chọn Dự Án Về Web

Một phần là do ý thích tuy nhiên phần lớn là do quy định của công ty, do đợt thực tập được công ty tổ chức dưới hình thức bố trí dự án không lựa chọn nên phải bắt buộc làm theo chương trình định ra của công ty.

Phần nữa cũng là do muốn một dự án gì đó ở mức khó và thử thách bản thân với các thư viện lạ hoàn toàn thay gì lặp đi lặp lại với các mô bản như dự đoán này kìa (Lstm, Rnn, Tranfromer, Resnet, Vgg) hay nhận diện (Yolo,…), hoặt chat bot trả lời câu hỏi, máy dịch và các dạng đề tài thuộc học máy hay học sâu.

Không phải những đề tài trên quá dễ ngược lại nó rất khó nếu đi sâu ví dụ ở mức độ dùng một mô hình nào đó để làm đề tài thì quá dễ tuy nhiên nếu nghiên cứu sâu về vấn đề, cải thiện mô hình đó, viết lại mô hình, các đánh giá và ý tưởng mới với đề tài đó, thì nó đang là bật thang hơi cao với tôi nên. Với kiến thức hiện tại nếu làm về chuyên sâu về việc nghiên cứu các tối ưu và cải tiến, tạo ra mô hình mới thì tôi không quá tự tin, còn làm với các mô hình sẵn có thì phí thời gian nên thay vì vậy trực tiếp chuyển sang một lĩnh vực khác và học một nền tản của lĩnh vực đó luôn sẽ là lựa chọn của tôi.

### Cảm Nhận Và Mong Muốn Khi Nhận Dự Án

Đây là một dự án với quy mô khá lớn và tổ chức rõ ràng chỉnh chu nên với một sinh viên lúc đầu có hơi bất ngờ, lúc đăng kí dự án và được nhận thì tôi hoàn toan vẫn chưa có bất kì kiến thức gì về Spring Boot java và React.js tức là để hoàn thành đợt thực tập tôi phải tự học cả hai thư viện khá lớn trong trước 2 tuần vì sau 2 tuần theo lịch chúng tôi sẽ bắt tay vào code dự án, việc này làm tôi khá lo lắng tuy nhiên cũng ở mức khá vì tôi rất tự tin với khả năng tự học của bản thân.

## Công Ty Thực Tập

### Giới Thiệu Về Công Ty

Tên đầy đủ là "Công ty TNHH Công Nghệ Số Việt Trí Đạo". Một đơn vị thành viên của công ty cổ phần công nghệ CNPT. Là công ty sẽ chịu trách nhiệm thay mặt công ty CNPT tổ chức đợt thực tập này.

Công ty có 3 lĩnh vực chính: Đào tạo nhân lực CNTT, hợp tác làm phần mềm cùng các doanh nghiệp IT, tư vấn quản trị cho các doanh nghiệp.IT.

### Quy Định Về Quá Trình Thực Tập Và Hình Thức Làm Việc

Chương trình thực tập của công ty được tỏ chức dưới hình thức online và làm một dự án do công ty đề ra, mỗi ngày công ty sẽ gửi mail với các công việc cụ thể phải hoàn thành. Và tất nhiên đợt thực tập chỉ là một khóa tổ chức của công ty với một dự án ảo tuy nhiên công ty sẽ hỗ trợ đánh giá sửa chữa các sản phẩm và công việc trong lúc làm việc.

Nghe có hơi thất vọng khi làm một dự án ảo lại còn online tuy nhiên nghĩ tới việc bây giờ không kinh nghiệm, không biết gì mà đi xin việc lại được làm cả một dự án thật thì có chút khó khăn. Nên việc tham gia các công ty tổ chức các đợt thực tập và hỗ trợ sinh viên lấy kinh nghiệm cũng hợp lý. Dù làm online và dự án ảo tuy nhiên sau quá trình thực tập tôi có thể khẳn định nó không hề dễ dàng một chút nào. Dựa trên cách thức làm việc thực tế nên cả quá trình thực tập cực kì chuyên nghiệp và rõ ràng.

Các quy định được đề ra trong kì thực tập : Giờ giấc làm việc từ 8 giờ sáng tới 4 giờ chiều. Mỗi ngày phải tổ chức họp online và ghi lại biên bản, mỗi các nhân phải viết một bản word Breakdown ghi lại quá trình làm việc của bản thân sau mỗi ngày. Trong quá trình làm việc mentor hay phía công ty sẽ làm việc với nhóm trưởng(leader) và nhóm phó để yêu cầu xem một vài thứ như user story bản thông tin về chi tiết các chức năng trên web, figma quản lý và hiển thị demo giao diện, test case thông tin về các chức năng cần test của phía các bạn tester. Cùng với quy định tổ chức lưu trữ dự án.

### Yêu Cầu Về Sản Phẩm

Do đợt thực tấp tổ chức trong vòng một tháng và từ tuần 2 mới bắt đầu code nên việc xây dựng hệ thống và thống nhất lại từ các project riêng lẻ là rất khó chính vì vậy từ đầu công ty chỉ đề ra hạn mức tối thiểu là các chức năng đều hoàn thành, các ngôn ngữ giống nhau có thể gộp code lại được, bên fontend React.js có thể gộp lại tất cả các trang. Còn kết quả tốt nhất tất nhiên là hoàn thành toàn bộ với kiến trúc micro service rồi.

### Quy Mô Dự Án

Như từ đầu đã nói qua dự án không hề nhỏ ít nhất là với tôi, web site có tổng cộng hơn 70 chức năng. Và cũng tầm 70 trang do sai lệch phía Ba lúc vẽ giao diện trên figma. Do nhiều chức năng và việc đây là một dự án rất chỉnh chu nên dĩ nhiên cơ sở dữ liệu khá phức tạp và nhiều thông tin.

### Giới Thiệu Thành Viên Và Cơ Cấu Tổ Chức

Dự án có khoản 22 thành viên cả nhóm trưởng và nhóm phó. Và tất cả thành viên đều là sinh viên thực tập bao gồm nhiều vị trí như Ba, Fontend, Backend, Tester.

Ba chịu trách nhiện phân tích nghiệp vụ là chính và vẽ ra bản thảo giao diện ban đầu

Fontend chịu trách nhiệm code React.js là chính.

BackEnd thì chủ yếu là code backend tức viết api, thiết lập, cấu hình, xây dựng hệ thống cơ sở dữ liệu.

Tetster thì chịu trách nhiệm tets các chức năng của Fontend và Backend.

### Quy Định Về Dự Án

Các quy tắt chung về dự án bao gồm viết toàn bộ bằng tiếng anh bao gồm code, biên bản họp, nhật kí bản thân,….

Phải code theo chuẩn, tôi không nhớ tên nhưng cơ bản là quy tắc đặc tên biến, tên hàm, ghi chú, tên class…

Và phải dùng cơ sở dữ liệu là Sql sever bản 2019

Đây là một web api tức là các Backend phải viết api document theo chuẩn và tương tự bên Fontend cũng phải dựa theo đó để làm

Tất cả các công việc của thành viên trừ khi nhận trực tiếp từ mail thì đều nhận phân công từ leader thông qua việc họp daily meeting vào mỗi buổi sáng lúc 8:00.

Hẳng ngày lead – trưởng nhóm phải phân công và làm phân công nhiệm vụ cho thành viên vào mỗi buổi sáng trên github theo mô hình Agile

### Yêu Cầu Về Năng Lực Thành Viên

May mắn cho tôi là đợt thực tập mở ra với mục đích là hỗ trợ sinh viên là chính nên không yêu cầu kinh nghiệm, nhưng khi vào giai đoạn cuối nó là một thảm họa, việc chênh lệch trình độ, ngơ ngát của các thành viên và nhiều vấn đề khác phát sinh nên đợt thực tập thật sự rất vất vã.

### Ví Trí Trong Dự Án

Ban đầu vị trí được phân công cho tôi là Backend tuy nhiên tới giai đoạn cuối do thiếu xót thành viên nên tôi phải lấn sang làm cả Fontend và trở thành Fullstack.

# CƠ SỞ LÝ LUẬN, LÝ THUYẾT VỀ DỰ ÁN

## Mô Tả Dự Án

Chúng ta sẽ cùng mô tả lại tổng quan của dự án, là một dự án về web cụ thể là một website quản lý cho một chung cư ở Hoa Kỳ. Dự án sẽ triển khai tầm 70 trang (page) với 70 chức năng bao gồm những chức năng của công dân, nhân viên và chủ sỡ hữu. Dự án được triển khai trên nền kiến trúc Microservice với sự độc lập của nhiều ngôn ngữ ở phía Backend và bên phía Fontend thì vẫn là kiến trúc Monolithic với một thư viện thống nhất là React.js.

Tương ứng với các ngôn ngữ chúng ta sẽ dùng các thư viện sau. Với java thì sẽ dùng Spring Boot, python thì là Django, Js sẽ là Nodejs, phía Fontend chỉ là React.js, và cơ sở dữ liệu là kiến trúc Shared Database với một Database duy nhất cho toàn bộ hệ thống dự án.

Các lựa chọn về kiến trúc là do phía công ty đề ra nên các nhận viên thực tập sẽ không có quyến thay đổi các kiến trúc này mà bắt buột phải tự học nếu không biết.

## Các Vấn Đề Của Dự Án

Quy mô nếu nói theo lý thuyết thì chắc là hệ thống ở mức trung là dành cho doanh nghiệp vừa nhỏ, nhưng đối với tôi thì đây hoàn toàn là một dự án lớn và quy mô của dự án có thể gây khó khăn.

Các sinh viên trong đợt thực tập có rất nhiều người vẫn chưa học thư viện mà mình đăng kí và sự khác nhau giữa các chức vụ Ba, Fe, Be, Tester khiến cho việc chênh lệch trình độ chắc chắn sẽ xảy ra và gây ảnh hưởng khá lớn.

Và việc các bạn sinh viên hoàn toàn không quen biết có thể gây khó khăn và một vài bất đồng trong quá trình làm việc nhóm.

Học thư viện là một chuyện tuy nhiên biết và viết theo chuẩn của kiến trúc Microservice lại là một chuyện khác nó phải được thống nhất từ phân chia thiết lập các dịch vụ (service) từ lúc đầu đồng thời tiến hành phân công và bố trí hợp lý chặt chẽ để đảm bảo các chức năng có thể kết nối mượt mà với nhau.

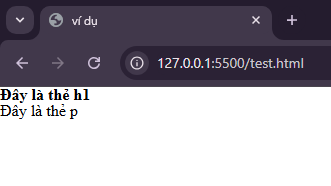
## Cơ Sở Lý Thuyết Và Kiến Thức Cần Có

### Các kiến Thức Cơ Bản nhất Về Web

#### Html

HTML, viết tắt của HyperText Markup Language, là một ngôn ngữ đánh dấu được thiết kế để tổ chức và sắp xếp dữ liệu trên các trang web. Về cơ sở lý thuyết, HTML hoạt động như một hệ thống các thẻ (tags), trong đó mỗi thẻ đại diện cho một loại dữ liệu hoặc thành phần cụ thể, chẳng hạn như đoạn văn bản, hình ảnh, liên kết, hoặc tiêu đề. Thông qua việc sử dụng các thẻ này, HTML giúp định nghĩa cấu trúc và trật tự của nội dung trên trang web, tạo ra một bố cục dễ hiểu cho cả con người và máy tính. Ngôn ngữ này không chỉ cung cấp cách để tổ chức dữ liệu một cách logic, mà còn cho phép tích hợp các công nghệ khác như CSS và JavaScript để bổ sung tính năng và phong cách hiển thị. Nhờ HTML, dữ liệu không còn là những chuỗi thông tin rời rạc mà trở thành một hệ thống sắp xếp có cấu trúc rõ ràng, dễ dàng truy cập và tương tác trên môi trường Internet.

Nói sơ qua cách thức hoạt động của các thẻ thì mỗi thẻ thường áp dụng riêng biệt cho một kiểu dữ liệu tuy nhiên việc dùng các thẻ đúng với các chức năng thường không mấy xảy ra, do khi thiết kế với css thì việc làm mới lại (reset) css là điều thường thấy, sau khi làm mới thì các thẻ của HTML hầu như giống nhau. Ví dụ thẻ h1 và thẻ p lúc làm mới sẽ giữ cùng một kích thước là 16 pixel theo bộ css reset của meyerweb việt làm mới là để cho quá trình code giao diện ở mức hoàn hảo và giống bản thiết kế nhất có thể. Như vậy có nghĩa là trang web được phép sử dụng các thẻ vô tội vạ vì google có một chuẩn cho web và chuẩn này sẽ ảnh hưởng đến việc web có thể lên top tìm kiếm hay không ví dụ như thẻ h1 sẽ xác định tiêu đề của trang web và bổ trí mỗi trang web đều có tiêu đề sẽ giúp trang web dễ xác định nội dung và dễ dàng nằm trong top tìm kiếm, tuy nhiên đây chỉ là phàn nhỏ còn việc làm sao tối ưu và chuẩn hóa code cho để lên được top của google thì có cả một chuyên ngành riêng biệt nên tôi sẽ không trình bài sâu.

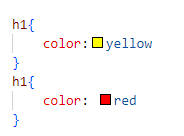
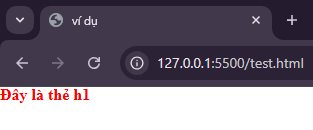


Hình II‑1 Ví dụ cho không có và có css khi sử dụng các thẻ h1 và p

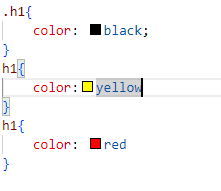
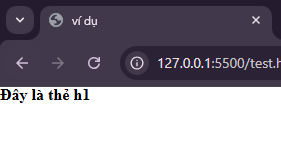
#### Css

CSS, viết tắt của Cascading Style Sheets, là một ngôn ngữ được sử dụng trong lập trình frontend để định dạng và trình bày nội dung trên các trang web. Về cơ sở lý thuyết, CSS hoạt động như một lớp trang trí, giúp điều chỉnh giao diện của các thành phần HTML mà không làm thay đổi cấu trúc dữ liệu của chúng. CSS cho phép lập trình viên kiểm soát các yếu tố về màu sắc, kích thước, font chữ, khoảng cách, bố cục, và nhiều khía cạnh khác liên quan đến hiển thị.

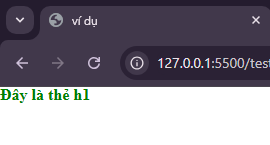
Một đặc điểm nổi bật của CSS là khả năng cascade, tức là các quy tắc định dạng được áp dụng theo thứ tự ưu tiên. Điều này có nghĩa là nếu có nhiều quy tắc định dạng cùng ảnh hưởng đến một phần tử, trình duyệt sẽ chọn quy tắc phù hợp nhất dựa trên mức độ ưu tiên như thứ tự khai báo, độ đặc hiệu (specificity), và tính kế thừa, tóm lại là Css sẽ được tải từ dưới lên nên mã nào được viết phía dưới sẽ được ưu tiên, và việt trỏ trực tiếp vào class sẽ được ưu tiên hơn là trỏ vào thẻ, và id là mức ưu tiên cao nhất.

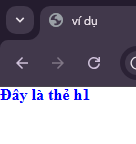
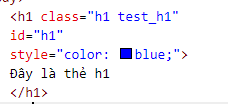
Hình II‑2 Độ ưu tiên của css theo thứ tự từ dưới lên

Hình II‑3 Độ ưu tiên css theo class

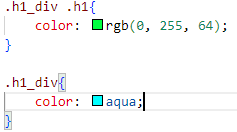
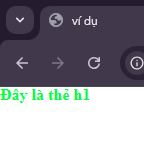


Hình II‑4 Độ ưu tiên của css theo id

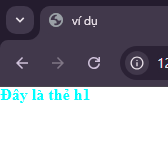
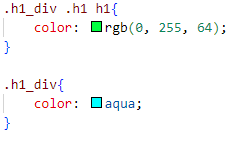


Hình II‑5 Độ ưu tiên của css theo cách viết trực tiếp

Và độ ưu tiên còn phụ thuộc vào mức độ rõ ràng khi trỏ trong css tức là khi trỏ vào class cha sẽ không được ưu tiên bằng trỏ vào class con tức là việc trỏ càng chi tiết thì ưu tiên càng cao và độ ưu tiên này tính theo class tức là nếu trỏ rõ ràng nhưng lại trỏ vào tag thay vì class thì cũng sẽ không được ưu tiên. Tức là nên trỏ tới class thay vì là thẻ trừ khi thẻ đó không có class.

Hình II‑6 Độ ưu tiên css theo sự rõ ràng khi trỏ



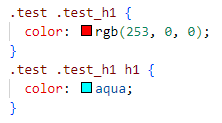
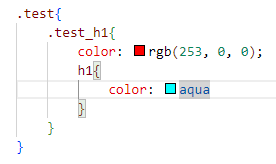
Hình II‑7 Độ ưu tiên css theo sự rõ ràng nhưng lại trỏ vào tag thay vì class

CSS còn hỗ trợ các khái niệm như Responsive Design, cho phép giao diện web tự động điều chỉnh trên nhiều thiết bị với kích thước màn hình khác nhau, nhờ vào các kỹ thuật như Media Queries. Ngoài ra, CSS có thể được viết trực tiếp trong thẻ HTML (inline CSS), trong một phần tử <style> (internal CSS), hoặc trong một tệp tin riêng biệt (external CSS), giúp tách biệt nội dung và giao diện, tăng tính tổ chức và tái sử dụng mã nguồn, và ngoài những trường hợp đặc biệt thì việc sử dụng theo cách viết trực tiếp trong mã HTML là rất hiếm.

Trong lập trình frontend, CSS đóng vai trò quan trọng, biến các trang web từ một bộ khung cơ bản thành những giao diện trực quan, hấp dẫn và thân thiện với người dùng.

#### Scss (thư viện Sass)

Là một thư viện hỗ trợ viết css, cơ bản là ta sẽ viết code css trên file scss thay vì file css và sẽ được hỗ trợ rất nhiều từ việc có thể lòng các css vào nhau khiến cho việc đọc hiểu trở nên dễ dàng hơn.



Hình II‑8 Ví dụ cho bố trí code trong scss và được built ra css

#### Js

JavaScript là một ngôn ngữ lập trình phổ biến và mạnh mẽ, chủ yếu được sử dụng để phát triển các ứng dụng web. Được tạo ra bởi Netscape vào năm 1995, JavaScript là ngôn ngữ chính để xây dựng các tính năng động và tương tác trên trang web, hay còn gọi là một ngôn ngữ xử lý sự kiện.

JavaScript có thể được sử dụng để theo dõi các sự kiện của người dùng như nhấp chuột, chuyển chuột, nhập văn bản, v.v, hay có thể thay đổi nội dung của trang web mà không cần phải tải lại trang (ví dụ: sử dụng AJAX để gửi và nhận dữ liệu mà không thực hiện trang mới). Sử dụng các API web để truy xuất dữ liệu từ các dịch vụ bên ngoài, như Google Maps, Facebook hoặc Twitter. Xử lý các loại dữ liệu như văn bản, số và mảng để tạo dữ liệu phân tích hoặc toán toán được phép.

Và hiện tại là một ngôn ngữ đứng thứ 2 thế giới sau sự tất yếu của Python với sự bùng nổ của AI và là một ngôn ngữ hỗ trợ cực mạnh cho lập trình web với các thư viện nổi tiếng và được sử dụng rộng rãi như Nodejs, React.js, Angular, Vue.js, …

Ngoài ra còn có các thư viện hỗ trợ Fontend app như : React Native, Vue Native, NativeScript, ….

Nói tóm lại thì Js một ngôn ngữ cực trâu bò trong tất cả mọi thứ có liên quan tới Fontend web hay app, từ đơn thuần như xử lý các aniamtion, tương tác đơn giản trong giai đoạn thiết kế hay cực chuyên sâu với các thư viện mạnh mẽ.

### Fontend, Backend, Ba, Tester

#### Fontend

Một frontend Developer (lập trình viên frontend) chịu trách nhiệm xây dựng phần giao diện người dùng và trải nghiệm người dùng của một ứng dụng hoặc trang web. Công việc của họ chủ yếu liên quan đến việc hiện thực hóa những gì người dùng nhìn thấy và tương tác với trong ứng dụng.

Một Fe Dev phải làm các công việc sau :

Thiết kế bố trí trang web địa phương hoặc ứng dụng: Giao diện người dùng của lập trình viên phải tạo bố cục hợp lý và dễ sử dụng. Điều này bao gồm việc sử dụng hệ thống lưới, bố cục không hợp lý và tạo giao diện người dùng các phần tử như menu, nút, bảng điều khiển, v.v.

Tạo các phần tử tương tác: Các phần tử như nút nhấn, biểu mẫu điền thông tin, thanh cuộn (thanh cuộn) và các phần tương tác khác cần được xây dựng để người dùng có thể thao tác ứng dụng một cách dễ dàng.

Đảm bảo rằng cấu trúc HTML được mã hóa rõ ràng và có nghĩa để tối ưu hóa SEO (Tối ưu hóa công cụ tìm kiếm) và truy cập quyền truy cập của người dùng.

Sử dụng CSS hoặc các công cụ như hoạt ảnh CSS để tạo hiệu ứng trực tiếp như chuyển động mượt mà khi di chuột qua, thay đổi màu sắc khi nhấp chuột, v.v.

Sử dụng một thư viện Fontend như React để xây dựng và hoàn thiện web site ở phía tương tác với người dùng sau cho hoàn hảo nhất, do đây là một web api nên bao gồm cả việt sử dụng các thư viện để tương tác với phía Backend để tác động tới dữ liệu như sửa xóa tạo thông tin.

#### Backend

Đảm bảo trách nhiệm xây dựng và duy trì các phần logic và quy trình xử lý dữ liệu của ứng dụng, đảm bảo mọi hoạt động cho máy chủ diễn đàn được chia sẻ và hiệu quả. Họ thiết kế hệ thống để nhận và xử lý các yêu cầu từ người dùng, quản lý luồng dữ liệu và thực hiện phép toán hoặc xử lý tạp chí mà người dùng không nhìn thấy. Công việc của một nhà phát triển phụ trợ không chỉ bao gồm việc tạo các thuật toán để xử lý dữ liệu mà phải đảm bảo hệ thống luôn hoạt động ổn định, an toàn và có thể mở rộng khi cần thiết.

Backend cần quản lý việc lưu trữ và truy xuất dữ liệu một cách hiệu quả, đồng thời tối ưu hóa các quy trình xử lý sao cho phù hợp với yêu cầu về hiệu suất và tài nguyên. Việc xử lý bảo mật cũng là một phần không thể thiếu trong công việc của backend Developer, từ việc bảo vệ dữ liệu người dùng đến xác thực quyền truy cập và ngăn chặn các mối đe dọa từ bên ngoài. Khi ứng dụng có yêu cầu mở rộng, backend Developer cũng sẽ chịu trách nhiệm xây dựng các giải pháp để đảm bảo hệ thống có thể đáp ứng được lượng người dùng và dữ liệu ngày càng lớn mà không làm giảm hiệu suất.

Ngoài ra, một backend Developer còn phải hợp tác chặt chẽ với các đội frontend để đảm bảo sự đồng bộ giữa dữ liệu và giao diện, giúp cho người dùng có trải nghiệm mượt mà và đầy đủ tính năng. Công việc của họ đòi hỏi sự hiểu biết sâu rộng về các công nghệ máy chủ, các phương pháp lập trình và cách tối ưu hóa các quy trình để đáp ứng nhu cầu của hệ thống và người dùng bao gồm thống nhất và code các chức năng trả về một api theo đúng với Api Document (một file có thể là work, exel chứa đầy đủ các thông tin mà một api phải trả về, dịnh dạng các api theo các chức năng và quy định rõ ràng rành mạch khi lỗi, khi truy xuất không thành công api sẽ trả về như thế náo, ...)

### React.js, Kiến Thức Nền Tảng, Nâng Cao

#### Giới thiệu về React.js

React.js là một thư viện JavaScript mã nguồn mở do Facebook phát triển vào năm 2013, được thiết kế để xây dựng giao diện người dùng (UI) một cách nhanh chóng, hiệu quả và linh hoạt. React ra đời để giải quyết vấn đề hiệu suất và khả năng mở rộng của các ứng dụng web lớn, bằng cách sử dụng công nghệ Virtual DOM, cho phép cập nhật giao diện nhanh hơn bằng cách chỉ thay đổi những phần cần thiết thay vì làm mới toàn bộ trang. React áp dụng kiến trúc Component-based, cho phép chia nhỏ ứng dụng thành các phần tái sử dụng được, giúp việc phát triển và bảo trì dễ dàng hơn. JSX, một cú pháp mở rộng của JavaScript, giúp kết hợp HTML với logic JavaScript một cách mượt mà và dễ hiểu. So với Angular, React linh hoạt hơn nhưng yêu cầu lập trình viên tự lựa chọn các công cụ bổ trợ, trong khi Vue.js dễ tiếp cận hơn cho người mới bắt đầu nhưng không phổ biến và đa dạng như React. React được đánh giá cao nhờ hiệu suất, cộng đồng mạnh mẽ và khả năng tích hợp với các công cụ hiện đại, tuy nhiên, việc học tập và áp dụng có thể phức tạp với người mới.

Tóm lại thì nó là một thư việ hỗ trợ viết mã của Fontend, hỗ trợ cho các Dev về tổ chức, Router (điều trang, chuyển trang), phân quyền phía người dùng (client), tối ưu hóa các đoạn mã HTML bằng cách viết chúng thành các Component và sử dụng lại, giống việc tạo ra một giao diện cart(thẻ) sản phẩm và khi nào cần thì gọi ra và tùy chỉnh nhanh gọn với props, giống bên lập trình hướng đối tượng, react xem Component như một đối tượng và có thể tùy chỉnh lúc gọi ra bằng cách truyền thông số vào, nó như cơ chế hàm(function hay methoth) trong lập trình truyền thống.

#### Vite

Vite là một công cụ xây dựng (build tool) hiện đại, nhanh và tối ưu được thiết kế để cải thiện hiệu suất trong việc phát triển và xây dựng ứng dụng web, đặc biệt là với các framework như React, Vue, và Svelte. Tăng tốc quá trình phát triển ứng dụng là công dụng chính của nó, việt sử dụng vite khiến cho quá trình cài đặt React và các thư viện con trở nên dễ dàng và không gặp các lỗi vặt, nếu dùng React thuần thì khi cài đặt các thư viện con hổ trợ như axios dể truy xuất api và .. sẽ rất dễ lỗi do chênh lệch phiên bản, …..

#### React Hooks, Context Api, Router

React cung cấp các công cụ mạnh mẽ như React Hooks để giúp quản lý trạng thái và xử lý các hiệu ứng phụ trong ứng dụng. useState được dùng để khai báo và quản lý trạng thái cục bộ trong các functional component, trong khi useEffect cho phép xử lý các tác vụ bất đồng bộ như gọi API hoặc lắng nghe sự kiện. Đối với các trường hợp trạng thái phức tạp hơn, useReducer cung cấp cách quản lý logic trạng thái tương tự Redux nhưng đơn giản hơn. Ngoài ra, bạn có thể tạo custom hooks để tái sử dụng các logic phức tạp giữa các component. Với các ứng dụng lớn, việc quản lý trạng thái toàn cục trở nên cần thiết và React hỗ trợ Context API để truyền dữ liệu xuyên suốt ứng dụng mà không cần thông qua props. Nếu cần một giải pháp mạnh mẽ và mở rộng hơn, Redux là lựa chọn phổ biến với khả năng quản lý trạng thái phức tạp và theo dõi các thay đổi dễ dàng. Bên cạnh đó, React Router hỗ trợ điều hướng giữa các trang, giúp xây dựng ứng dụng nhiều màn hình. Nó cung cấp các công cụ để cấu hình route cơ bản, route động và quản lý tham số URL, giúp việc điều hướng trong ứng dụng React trở nên linh hoạt và tối ưu.

#### Api và Deloy

Trong React, API tương tác là một phần quan trọng để lấy hoặc gửi dữ liệu. Fetch API là công cụ tích hợp sẵn trong JavaScript, giúp thực hiện các HTTP yêu cầu như GET, POST, PUT hoặc DELETE. Nó dễ sử dụng nhưng cần lặp lại thao tác để xử lý các lỗi hoặc cấu hình phức tạp. Axios là một thư viện bên ngoài mạnh mẽ hơn Tìm nạp, cung cấp cú pháp ngắn gọn và tính năng mở rộng như tự động chuyển đổi JSON, hỗ trợ adobe request và cấu hình tiêu đề dễ dàng hơn. Cả hai đều được sử dụng để kết nối ứng dụng React với API RESTful.

Netlify và Vercel là nền tảng phổ biến để phát triển ứng dụng React nhanh chóng và dễ dàng. Chúng hỗ trợ tích hợp GitHub, GitLab hoặc Bitbucket, cho phép phát triển tự động ngay khi có thay đổi nguồn mã hóa. Netlify nổi bật với các tính năng như quản lý miền, gửi biểu mẫu và các chức năng không có máy chủ. Vercel, ngược lại, tối ưu cho các ứng dụng sử dụng React hoặc Next.js, cung cấp khả năng xử lý mạnh mẽ Kết xuất phía máy chủ (SSR). Cả hai đều giúp đưa ra ứng dụng React từ môi trường phát triển lên môi trường thực tế mà không yêu cầu cấu hình máy chủ phức

### Spring boot là gì? Kiến Thức Nền Tảng

#### Giới thiệu về Spring Boot

Spring Boot là một framework Java mã nguồn mở, được xây dựng trên Spring Framework, nhằm đơn giản hóa quá trình phát triển và triển khai ứng dụng. Nó ra đời năm 2014 bởi Pivotal để giải quyết sự phức tạp trong việc cấu hình và tích hợp Spring truyền thống. Với các tính năng như tự động cấu hình, tích hợp thư viện dễ dàng, và máy chủ nhúng, Spring Boot giúp lập trình viên tập trung vào logic kinh doanh mà không cần lo lắng về cấu hình chi tiết. Spring Boot thường được sử dụng trong phát triển microservices, hệ thống quản lý dữ liệu lớn, và các ứng dụng doanh nghiệp do tính linh hoạt và hiệu quả cao.

Spring Boot là một phần mở rộng của Spring Framework, được thiết kế để làm đơn giản hóa quá trình cấu hình và triển khai các ứng dụng Spring. Trong khi Spring Framework cung cấp các tính năng mạnh mẽ cho việc phát triển ứng dụng, Spring Boot tự động hóa nhiều phần cấu hình phức tạp của Spring, giúp giảm thiểu mã và cấu hình cần thiết. Nó hỗ trợ nhanh chóng tạo ứng dụng với các cấu hình mặc định hợp lý, đặc biệt là cho các ứng dụng microservices

#### Annotation, Dependency Management

@SpringBootApplication: Là chú thích chính trong Spring Boot, bao gồm các tính năng của @Configuration, @EnableAutoConfiguration và @ComponentScan. Nó giúp Spring Boot tự động cấu hình ứng dụng và quét các thành phần lớp trong gói.

@RestController: Được sử dụng để đánh dấu bộ điều khiển API của các lớp, kết hợp với @ResponseBody để trả về dữ liệu JSON từ các phương thức, ngoài ra còn @ResquestParam, @ResquestBody, @PathVariable để nhận dữ liệu khi dược Request.

@Service: Được sử dụng để đánh dấu một lớp chứa logic nghiệp vụ, giúp Spring quản lý các service Bean trong container của nó.

@Entity được áp dụng cho các lớp Java để đánh dấu chúng là một entity, tức là các đối tượng sẽ được ánh xạ với bảng trong cơ sở dữ liệu. Mỗi instance của lớp này sẽ tương ứng với một dòng dữ liệu trong bảng. Trong khi đó, @Repository là annotation được sử dụng cho các lớp có nhiệm vụ truy cập và quản lý dữ liệu trong cơ sở dữ liệu. Các lớp được đánh dấu với @Repository thường kế thừa từ JpaRepository hoặc CrudRepository, giúp cung cấp các phương thức CRUD cơ bản mà không cần phải viết mã truy vấn SQL thủ công. Cả hai annotation này đều giúp đơn giản hóa việc tương tác với cơ sở dữ liệu trong các ứng dụng Spring Boot.

Spring Boot sử dụng Spring Boot Starter để quản lý các phụ thuộc. Các "starter" là các bộ sưu tập của các thư viện thường dùng trong ứng dụng, giúp đơn giản hóa việc thêm các thư viện vào dự án. Ví dụ, spring-boot-starter-web cung cấp các thư viện cần thiết để phát triển ứng dụng web, bao gồm Spring MVC, Tomcat, và các thư viện liên quan đến REST. Các starter giúp giảm bớt sự phức tạp trong việc cấu hình và duy trì các thư viện trong dự án.

### Kiến trúc Microservice và Monolithic ở Backend

Kiến trúc Monolithic là mô hình phát triển ứng dụng trong đó tất cả các thành phần của hệ thống (giao diện người dùng, logic nghiệp vụ, cơ sở dữ liệu) được gói gọn trong một ứng dụng duy nhất. Trong kiến ​​trúc này, các tính năng và chức năng của ứng dụng được tích hợp chặt chẽ với nhau ở một nguồn mã nguồn và phát triển duy nhất.

Ưu điểm: Dễ dàng phát triển và khai thác các ứng dụng nhỏ và trung bình. Đơn giản hóa công việc quản lý nguồn và cấu hình. Giao tiếp giữa các thành phần trong ứng dụng nhanh chóng và hiệu quả.

Nhược điểm: Khó bảo trì và mở rộng khi ứng dụng trở nên lớn hơn. Việc cập nhật hoặc thay đổi một phần hệ thống có thể ảnh hưởng đến toàn bộ ứng dụng. Khó khăn khi ứng dụng cần hỗ trợ nhiều môi trường hoặc công nghệ khác nhau.

Kiến trúc Microservice chia sẻ một ứng dụng cho nhiều dịch vụ nhỏ độc lập, mỗi dịch vụ thực hiện một chức năng cụ thể và giao tiếp với các dịch vụ khác thông qua API (thường là HTTP hoặc nhắn tin). Mỗi dịch vụ có thể được phát triển, phát triển và mở rộng độc lập, sử dụng công nghệ và cơ sở dữ liệu riêng biệt nếu cần.

Ưu điểm: Dễ dàng mở rộng và bảo trì vì mỗi dịch vụ đều có thể được phát triển và phát triển độc lập. Khả năng sử dụng các công nghệ khác nhau cho các dịch vụ khác nhau. Dễ dàng phát triển việc khai báo và kiểm tra các dịch vụ độc lập mà không ảnh hưởng đến tổng thể hệ thống. Tính linh hoạt cao, dễ dàng phát triển theo từng phần, phù hợp với các ứng dụng lớn và phức tạp.

Nhược điểm: Phúc tạp trong việc phát triển và quản lý, đặc biệt khi có số lượng dịch vụ lớn. Cần có cơ chế để quản lý giao dịch tiếp theo giữa các dịch vụ và đồng bộ hóa dữ liệu. Đòi hỏi hạ tầng mạnh mẽ để giám sát, bảo mật và đảm bảo tính sẵn sàng của các dịch vụ.

### Kiến trúc Monolithic và Micro ở Fontend

Trong frontend, kiến ​​trúc Monolithic và Microfrontend cũng phản ánh các sự khác biệt về cách tổ chức và phát triển ứng dụng, tương tự như trong backend, nhưng chúng trung tập vào cách chia nhỏ và quản lý các phần giao diện người dùng.

Trong kiến ​​trúc giao diện người dùng nguyên khối, toàn bộ giao diện người dùng của ứng dụng được xây dựng và phát triển như một ứng dụng duy nhất. Tất cả các phần tử người dùng giao diện người dùng (UI), chức năng và ứng dụng logic đều được kết hợp trong một nguồn mã duy nhất và được phát triển trên một máy chủ. Đây là cách tiếp cận truyền thông tiếp theo, trong đó toàn bộ ứng dụng là một khối duy nhất.

Ưu điểm: Dễ dàng quản lý và phát triển khi ứng dụng nhỏ hoặc ở giai đoạn đầu. Các giao diện được tích hợp chặt chẽ và có thể chia sẻ nguồn mã hóa một cách dễ dàng. Quản lý và phát triển một dự án duy nhất, đơn giản hơn.

Nhược điểm: Khó mở rộng khi ứng dụng trở nên lớn hơn vì sự phát triển của các tính năng mới có thể ảnh hưởng đến toàn bộ ứng dụng. Các thay đổi hoặc cập nhật thường xuyên có thể yêu cầu toàn bộ ứng dụng cấu trúc tái sinh. Mã nguồn có thể trở nên phức tạp và khó bảo trì khi có nhiều người phát triển hoặc tăng cường năng lực.

Microfrontend là một phương pháp phát triển trong đó người dùng được chia nhỏ thành các phần độc lập, mỗi phần có thể được phát triển, phát triển khai và bảo trì riêng biệt. Mỗi "giao diện người dùng vi mô" có thể là một ứng dụng nhỏ, có thể tương tác với các phần khác của API thông tin ứng dụng hoặc giao diện giao diện giữa các phần (có thể là giao diện hoặc trạng thái). Các phần này có thể sử dụng các công nghệ khác nhau nhưng được tích hợp lại thành một ứng dụng thông tin qua một hệ thống điều phối.

Ưu điểm: Mỗi phần của người dùng có thể được phát triển và phát triển độc lập, giúp giảm sự phụ thuộc giữa các nhóm phát triển. Có thể sử dụng các công nghệ khác nhau cho từng phần ứng dụng, linh hoạt hơn trong việc phát triển. Dễ dàng mở rộng và bảo trì vì các phần nhỏ đã được cập nhật mà không ảnh hưởng đến toàn bộ ứng dụng.

Nhược điểm: Cần một tầng kiến ​​trúc phức tạp để hợp nhất các phần lại với nhau. Quản lý và đồng bộ các microfrontend có thể trở nên phức tạp. Hỗ trợ khai thác và bảo trì miễn phí khi các phần ứng dụng cần giao tiếp với nhau.

### Kiến trúc Shared Database và Database Per Service

Trong Cơ sở dữ liệu dùng chung có cấu trúc, tất cả các dịch vụ trong hệ thống đều chia sẻ một cơ sở dữ liệu chung. Các dịch vụ này đều truy cập vào cùng một cơ sở dữ liệu và hoạt động với các bảng và bảng của cơ sở dữ liệu. Mỗi dịch vụ trong hệ thống có thể thực hiện đọc/ghi các thao tác vào cơ sở dữ liệu chung này.

Ưu điểm: Quản lý dữ liệu đơn giản: Việc chia sẻ cơ sở dữ liệu giúp tránh trùng lặp dữ liệu và đảm bảo tính nhất quán của dữ liệu giữa các dịch vụ. Tính đơn giản trong phát triển khai báo: Quản lý và duy trì một cơ sở dữ liệu duy nhất một cách dễ dàng hơn vì không cần phải đồng nhiều cơ sở dữ liệu khác nhau. Dễ dàng thực hiện giao dịch: Vì tất cả các dịch vụ đều sử dụng cơ sở dữ liệu chung nên các giao dịch giữa các dịch vụ có thể dễ dàng thực hiện hơn.

Nhược điểm: Mở rộng khó khăn: Khi tăng số lượng dịch vụ, chia sẻ một cơ sở dữ liệu duy nhất có thể gây tắc nghẽn cổ chai và giảm hệ thống hiệu suất. Khó khăn trong việc bảo trì: Sự thay đổi trong cơ sở dữ liệu cấu trúc có thể ảnh hưởng đến toàn bộ hệ thống, gây khó khăn cho việc duy trì và phát triển các dịch vụ. Rủi ro khi xảy ra lỗi: Nếu cơ sở dữ liệu gặp sự cố, tất cả các dịch vụ đều có thể bị ảnh hưởng, dẫn đến sự cố toàn hệ thống.

Trong cấu trúc cơ sở dữ liệu trên mỗi dịch vụ, mỗi dịch vụ trong hệ thống đều có cơ sở dữ liệu riêng biệt và độc lập. Các dịch vụ này có thể sử dụng các loại cơ sở dữ liệu khác nhau, tùy thuộc vào yêu cầu và mục đích sử dụng của từng dịch vụ.

Ưu điểm: Tính độc lập cao: Mỗi dịch vụ có thể phát triển và phát triển độc lập mà không ảnh hưởng đến các dịch vụ khác, do đó giảm thiểu sự phụ thuộc vào cơ sở dữ liệu chung. Dễ dàng mở rộng: Mỗi dịch vụ có thể được tối ưu hóa và mở rộng riêng biệt, phù hợp với nhu cầu của dịch vụ đó mà không ảnh hưởng đến các dịch vụ khác. Lựa chọn cơ sở dữ liệu phù hợp: Mỗi dịch vụ có thể chọn loại cơ sở dữ liệu phù hợp với yêu cầu của mình (SQL, NoSQL, v.v.), giúp hiệu suất hóa tối ưu.

Nhược điểm: Khó khăn trong công việc duy trì tính chất nhất quán: Dữ liệu của các dịch vụ có thể được phân tích, công việc duy trì tính chất nhất quán và đồng bộ hóa giữa các cơ sở dữ liệu là một công thức lớn. Tăng cường quản lý chi phí: Cần phải quản lý và duy trì nhiều cơ sở dữ liệu khác nhau, dẫn đến chi phí cao và phức tạp hơn trong việc phát triển khai thác, sao lưu và phục hồi dữ liệu. Thử thách trong việc thực hiện phân tích giao dịch: Việc thực hiện giao dịch liên dịch vụ trở nên khó khăn hơn khi mỗi dịch vụ có cơ sở dữ liệu riêng biệt.

# TRIỂN KHAI XÂY DỰNG DỰ ÁN

## Tổng Thời Gian Làm Việc Và Phân Bố

Thời gian thực tế triển khai dự án khoản hơn 1 tháng bỏ đi 2 ngày cuối tuần, tổng thời gian làm việc là 22 ngày mỗi ngày 8 tiếng.

Ở những ngày đầu tiên sẽ bao gồm những công việc chủ yếu làm làm quen với cách thức tổ chức của công việc, chuẩn bị các bước cơ bản cho công việc phía sau và ở 2 tuần đầu chủ đạo công việc là của Ba nên các Dev và tester cũng không hoạt động nhiều, bắt đầu ở khoảng ngày 10 thì mới chình thức bắt đầu code lúc này thì nhóm Dev mới bắt đầu chính thức làm việc còn những ngày trước đó thì công việc tương đối nhẹ nhàn và chủ yếu thời gian là dành cho việc học và học các kiến trúc mới. Và tới ngày 17 các Dev sẽ không còn code gì nữa việc tiếp theo chỉ là fixbug và testing. Tới ngày 22 thì sẽ nộp demo bao gồm báo cáo thống kế công việc hoàn thành, file testing. Sau đây sẽ là chi tiết công việc phân công theo từng ngày phía công ty.

Ngày 1 : Tham gia cuộc họp KICK-OFF xác nhận tham gia thự tập và phân nhóm, sau khi biết được bản thân thuộc nhóm nào tiến hành tham gia hợp meeting và làm quen sau đó lựa chọn nhóm trưởng(leader) và nhóm phó. Sau đó là thông báo về việc kiến trúc tổng quan của dự án các quy định, và xác định các yêu cầu về kết quả của dự án.



Hình III‑1 mail các công việc trong ngày đầu tiên

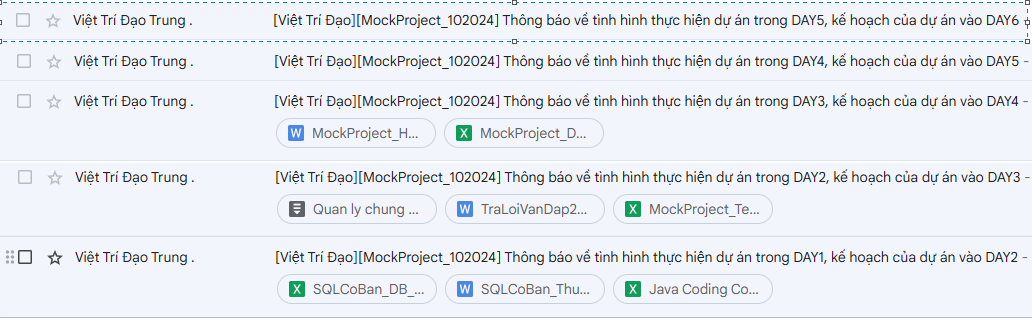
Ngày 2 – ngày 6: các công việc sẽ bao gồm:

Học vẽ giao diện trên figma, học nếu chưa biết ôn lại nếu biết về Sql sever 2019, các Dev thống nhất với nhau về thư viện.

Lựa chọn và thống nhất giao diện sau đó nhóm Dev bắt đầu code thử giao diện đăng nhập – đăng kí, đội tester sẽ bắt đầu viết testcase dựa trên giao diện, Ba sẽ bắt đầu nghiên cứu về nghiệp vụ của phần mềm.

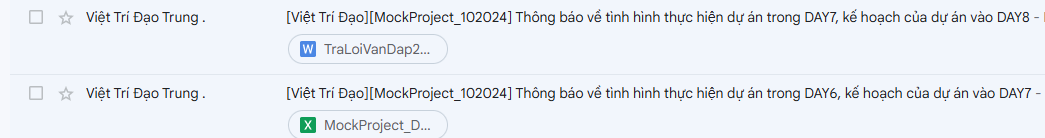
Ba bắt đầu phát thảo toàn bộ màn hình giao diện trên nháp, Dev sau khi thông nhất thư viện ở ngày 2 thì bây giờ menter sẽ xem xét và hỗ trợ sửa đổi nếu cần. Ngoài ra đội Dev phải bắt đầu học cách sử dụng Github để push, merge code lại thử, Tester cũng học cách quản lý lỗi trên github.

Ba triển khai các giao diện lên figma, và đưa menter xem xét sửa chửa, các Dev sẽ thử làm một màn hình bất kì và có Responsive.



Hình III‑2 mail từ ngày 2 đến ngày 6 của công ty

Ngày 7 – ngày 8 : Bắt đầu thiết kế cơ sở dữ liệu và đưa cho menter xem xét, phải sửa chửa nếu chưa tốt, sau đó bắt đầu triển khai lên Sql sever với mỗi bảng gồm 20 dòng dữ liệu mẫu.



Hình III‑3 mail công việc ngày 7, ngày 8 của công ty

Ngày 9 – ngày 16:

Tổ chức một buổi Sprint plan với nội dung sau : Ba viết User story theo các giao diện đã đưa ra trước đó trên figma, quy tắc đặc tên, đội Dev thì tùy theo lựa chọn là web api hay web không api mà tiến hành công việc nếu có api thì phải thống nhất viết Api Document còn nếu không api thì bắt đầu triển khai code, leader cũng phải viết một file phân công và chia công việc.

Ba chỉnh sửa lại User story, Dev bắt đầu code các chức năng mình phụ trách theo file phân công từ leader, Tester viết Testcase theo User story.

leader viết báo cáo tiến độ lần 1, Dev tiếp tục code, Tester tiếp tục viết Testcase.

Phía công ty thông báo thêm chức năng, toàn bộ thành viên họp về phương án xử lý, sửa chửa, thêm trong cơ sở dữ liệu, giao diện, cho các chức năng mới.

Dev tiếp tục code, Tester tiếp tục viết Testcase và sửa chữa code hay Testcase theo phương án hôm qua đã bàn nếu có.

Dev tiếp tục code, Tester chuẩn bị data cho sắp tới, leader báo cáo tiến độ lần 2.

Dev tiếp tục code và Tester học cách kiểm tra api.

Tester học cách kiểm tra Api Ba viết tài liệu SRS



Hình III‑4 mail công việc ngày 9 - ngày 16 của công ty

Ngày 17 – ngày 22 : từ ngày 17 các Dev ngừng code.

Dev tiến hành hợp code và fixbug nếu có, Tester bắt đầu kiểm tra dò lỗi và thông báo cho Dev.

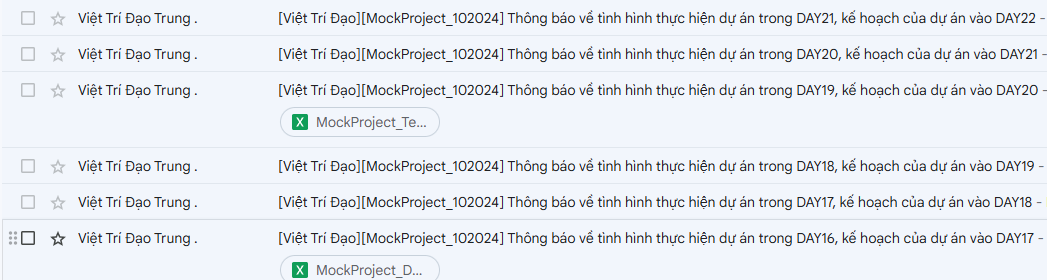
Như ngày 17

Như ngày 17 nhưng cuối ngày thì Dev và Tester dừng lại không động vào dự án nữa dù còn lỗi hay không.

Tester viết Test report, các thành viên còn lại làm slide báo cáo cuối cùng cho dự án, báo cáo gồm khoản 7 trang với 2 phiên bản tiếng việt và tiếng anh.

Các thành viện tổ chức họp Sprint review quay lại toàn bộ và nộp cho menter. Nội dung cuộc hợp bao gồm review qua toàn bộ các công việc mà mỗi thành viên đã làm.

Họp cuộc hợp Sprint Retro với nội dung review và phản hồi lại về các thành viên trong toàn bộ quá trình thực tập và thông báo kết thúc đợt thực tập.



Hình III‑5 mail công việc ngày 17 - ngày 22 của công ty

## Thống Nhất Về Thư Viện Và Công Cụ Hổ Trợ

Những công việc theo từng ngày với toàn bộ thành viên tôi đã trình bài vô cùng rõ ràng phía trên phần 1 nên từ phần này tôi sẽ bắt đầu xoay quanh những công việc mà cá nhân tôi làm.

Ban đầu tôi là Backend dev Java Spring Boot nên chỉ thống nhất thư viện và công cụ với bên Java Backend và không quan tâm tới các nhóm khác như React js, Django, Nodejs, và đến tận giữa dự án do các Fontend code nghỉ quá nhiều nên bắt buộc tôi phải lấn sang trợ giúp và code cả React.js nên lúc đó tôi phai theo cấu hình đã dược bàn trước. Và do cuối cùng tôi sẽ làm cả 2 nên trong phần này tôi sẽ ghi cả 2 phần.

Cụ thể thì phía Java sẽ thống nhất thư viện như sau: Jdk 21, phiên bản Spring Boot là 3.2.10, IDE là intelij, project Maven, các dependencies : Spring web, lombook, Jpa, Mssql driver.

Phía React.js : React.js 18, css bootstrap, java scrip không dùng vite dùng create-react-app để tạo prooject.

## Viết User Story

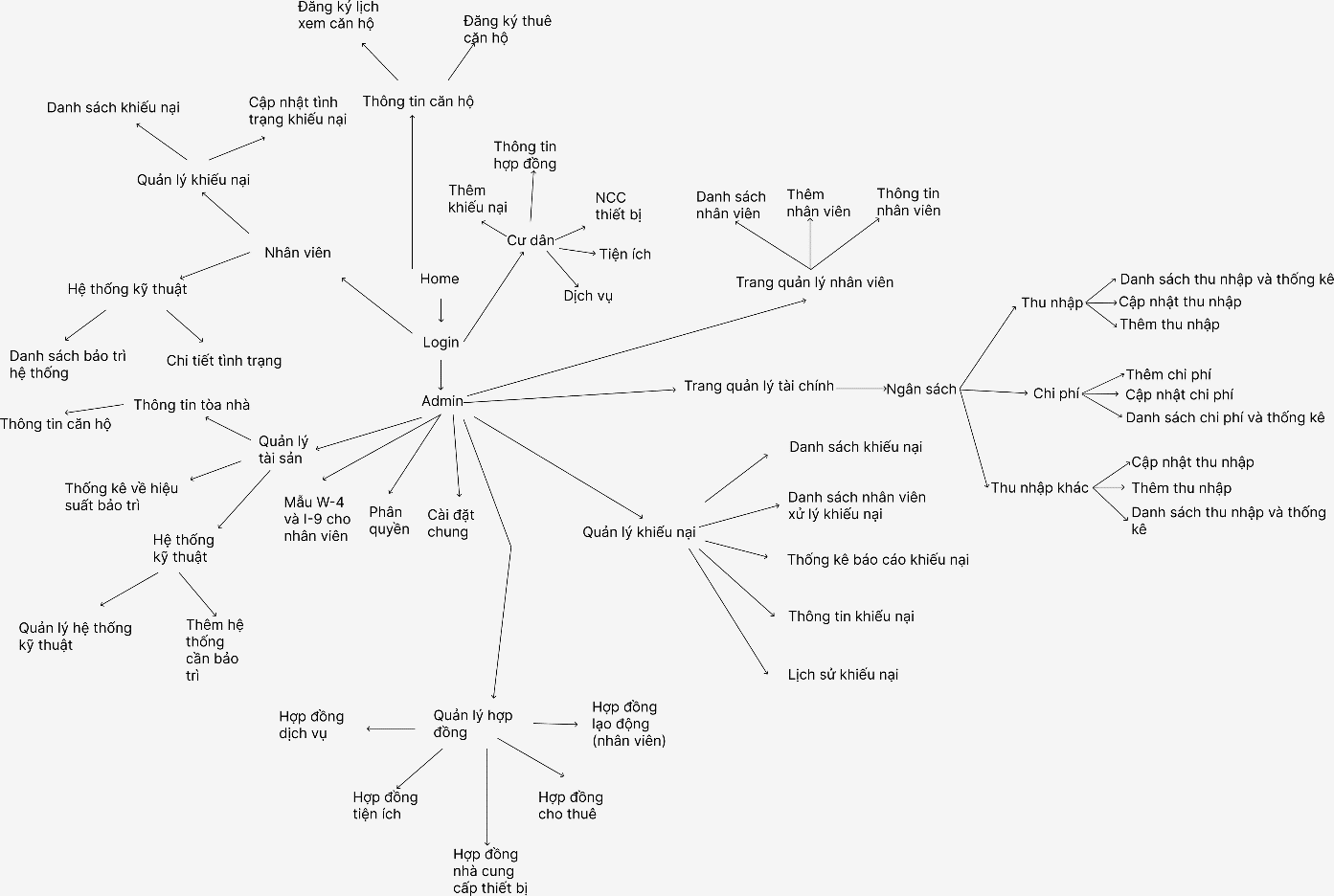
Như đã nói qua ở các phần trên đây là file thống nhất vả tổng hợp toàn bộ các chức năng, trang giao diện.

Phần này là của đội Ba nên tôi sẽ không nói chuyên sâu chỉ nói về phần mình đã đóng góp trong phàn này. Do lúc đầu đội Ba làm khá vội nên bị thiếu một vài màn hình bên chức năng của Admin cụ thể là trang thêm, sửa, xóa người dùng, trang xóa và sửa thanh toán. Và một vài sai lệch nhỏ trong trang như hiển thị bị dư thông tin, thiếu thông tin, các trang sửa dữ liệu bị thừa thông tin. Do lúc làm cơ sở dữ liệu mới phát hiện là các trang chưa tối ưu nên phần này đã được leader thông qua trong cuộc hợp là Dev được quyền tùy chỉnh lại nếu sai lệch nhỏ, và nếu sai lệch lớn thì báo lại Ba và tìm cách khắc phục sau.

## Thiết Kế Giao Diện Dự Án

Tuần đầu tiên nhóm sẽ học về sử dụng Figma để bắt đầu tự thiết kế một giao diện web cơ bản cho dự án.

Sơ đồ đường đi của trang web ban đầu được thiết kế:



Hình III‑6 Sơ đồ chức năng tramg web

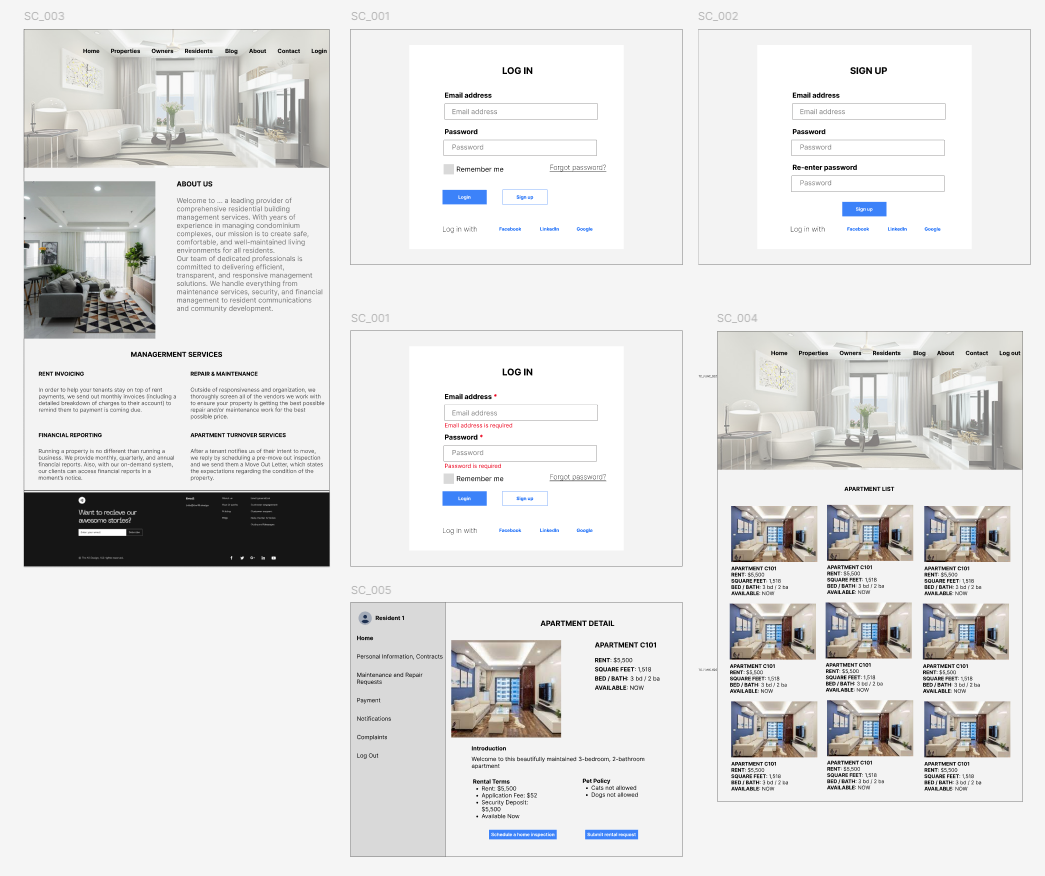
Các phần giao diện được thiết kế:

Phần giao diện người truy cập vào web

Phần giao diện đăng nhập và trang chủ lúc truy cập vào web và xem thông tin của căn hộ.

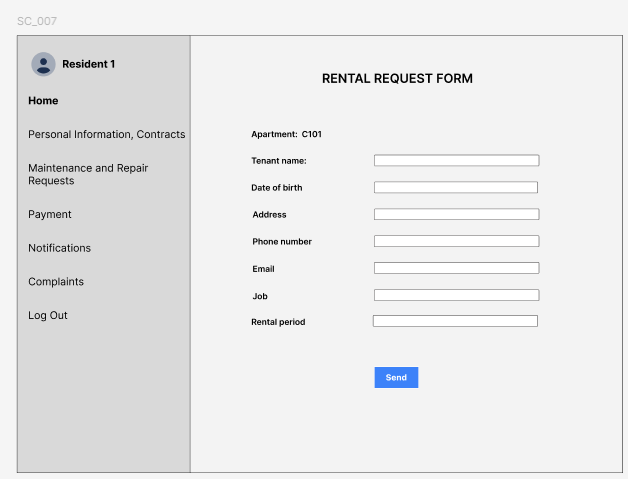
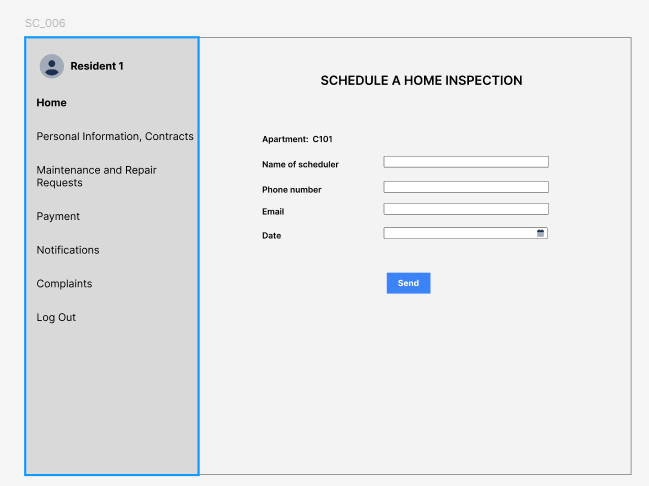
Nếu là người đang có ý định thuê căn hộ, sau khi xem chi tiết căn hộ có thể thuê căn hộ người truy cập web có thể đặt lịch hẹn xem nhà hoặc điền thông tin vào biểu mẫu đăng ký thuê căn hộ.

Giao diện đăng nhập dành cho Admin, nhân viên, người thuê căn hộ.



Hình III‑7 Screen 1-5

Trang cho khách xem đăng ký lịch xem căn hộ và thuê nhà.



Hình III‑8 Screen 6-7

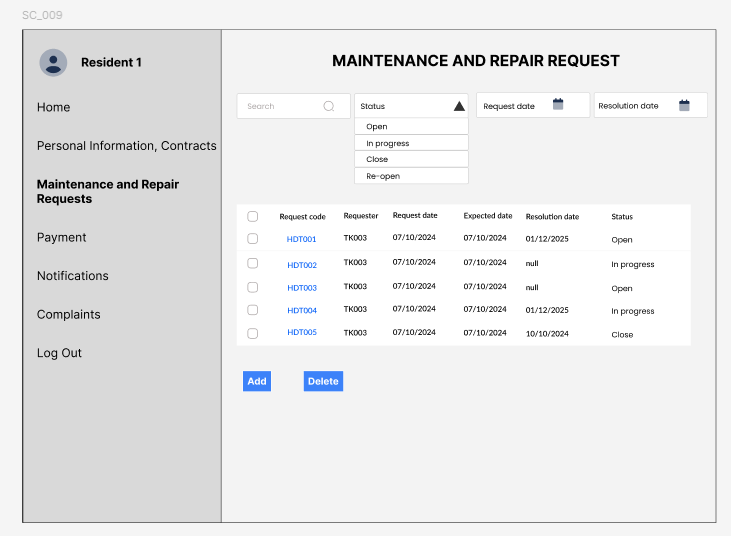
Phần giao diện cho người đã thuê căn hộ (resident). Gồm những trang:

Thông tin người thuê nhà.

Người thuê sẽ xem được thông tin của mình. Có thể sửa được thông tin trong trường hợp người nhập dữ liệu vào điền sai hoặc muốn thay đổi thông tin.

Trang gửi yêu cầu sửa chữa, bảo trì thiết bị.

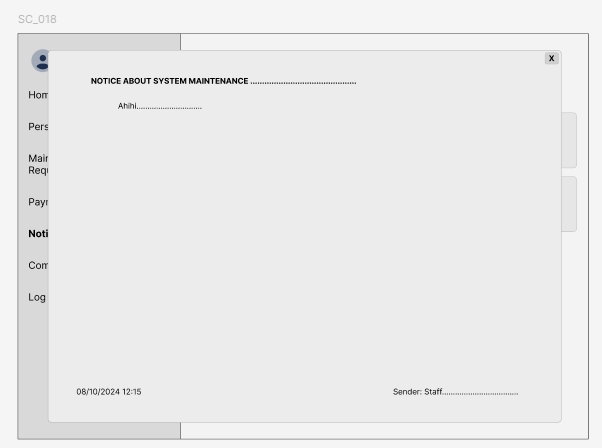
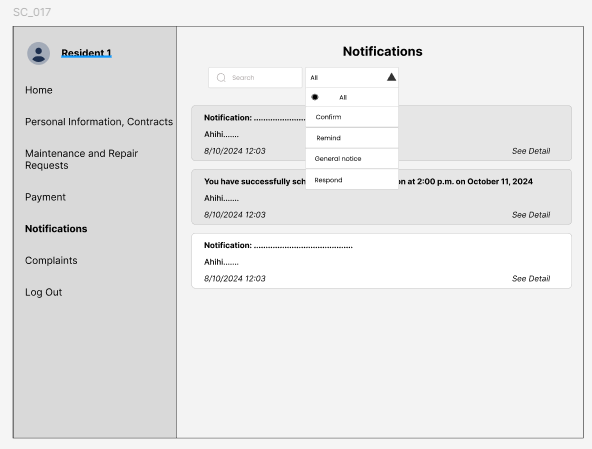
Người thuê sẽ gửi yêu cầu sửa chữa, bảo trì thiết bị tới cho admin. Có thể xem được lịch sử gửi yêu cầu và trạng thái của yêu cầu là đã được sửa chữa xong hay chưa hay đang trong quá trình sửa chữa.



Hình III. 4 Giao diện cho người thuê

Trang nhận thông báo từ chủ nhà.

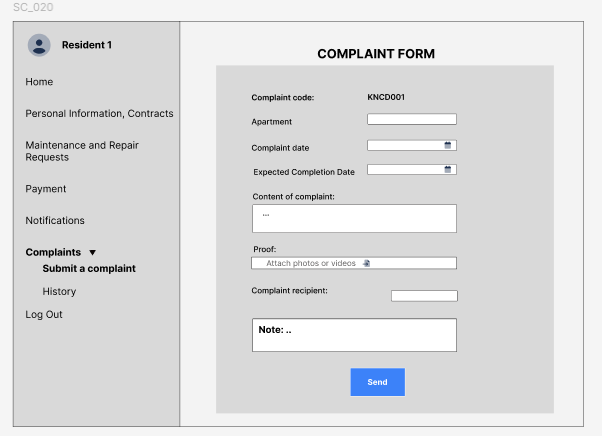
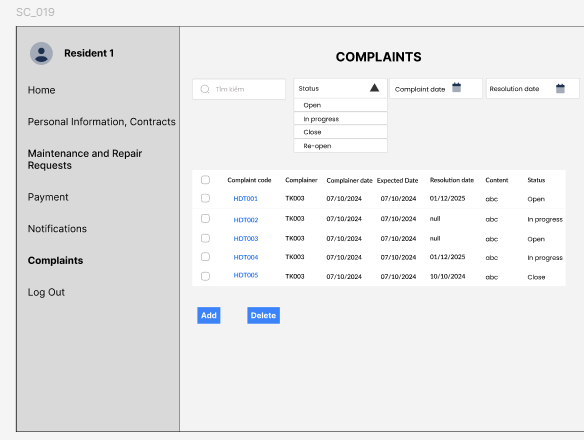
Thông báo của chủ nhà gửi tới người thuê nhà, người thuê nhà có thể trực tiếp vào đây xem thông báo. Gồm danh sách các thông báo, người dùng có thể bấm vào phần mô tả để xem chi tiết thông báo.



Hình III‑9 Giao diện thông báo

Trang góp ý, phản hồi.

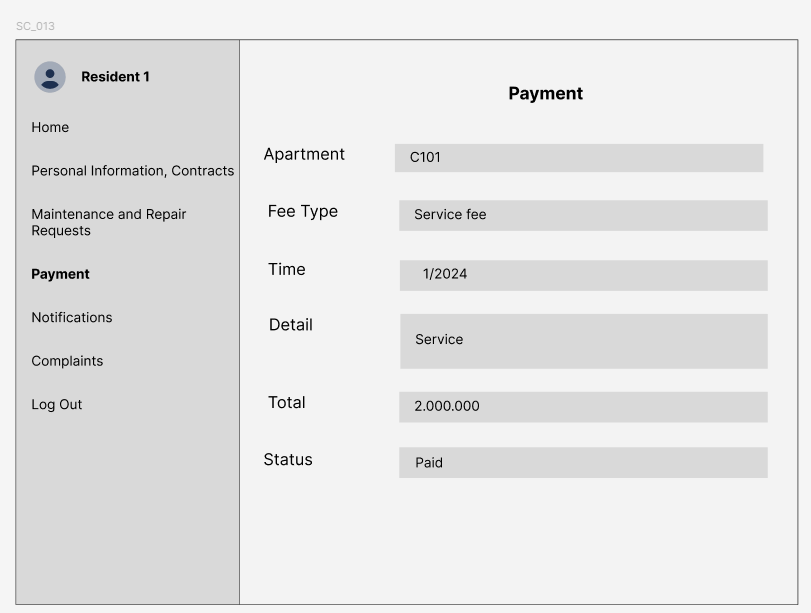
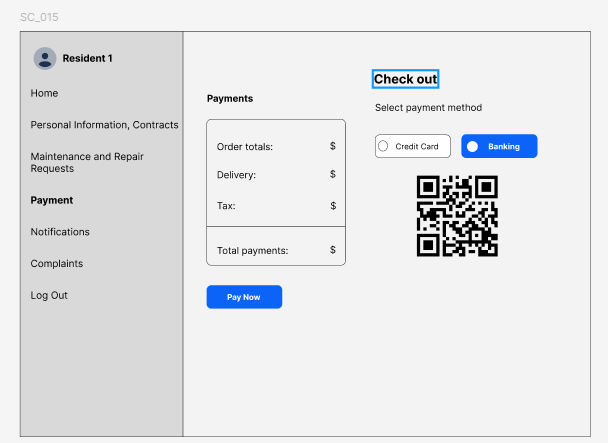
Người thuê nhà muốn gửi góp ý hoặc khiếu nại gì thì vào trang này để tạo khiếu nại.



Hình III‑10 Giao diện khiếu nại

Trang thanh toán:

Người thuê nhà sẽ vào phần giao diện này để kiểm tra thanh toán và công nợ, gồm lịch sử thanh toán của cá nhân người dân, sẽ có cả thông báo mục nào người thuê nhà chưa thanh toán. Còn phải có phần giao diện để xem chi tiết cái hóa đơn thanh toán đó. Thanh toán online gồm 2 loại là quét mã QR thanh toán hoặc chuyển khoản bằng thẻ tín dụng.



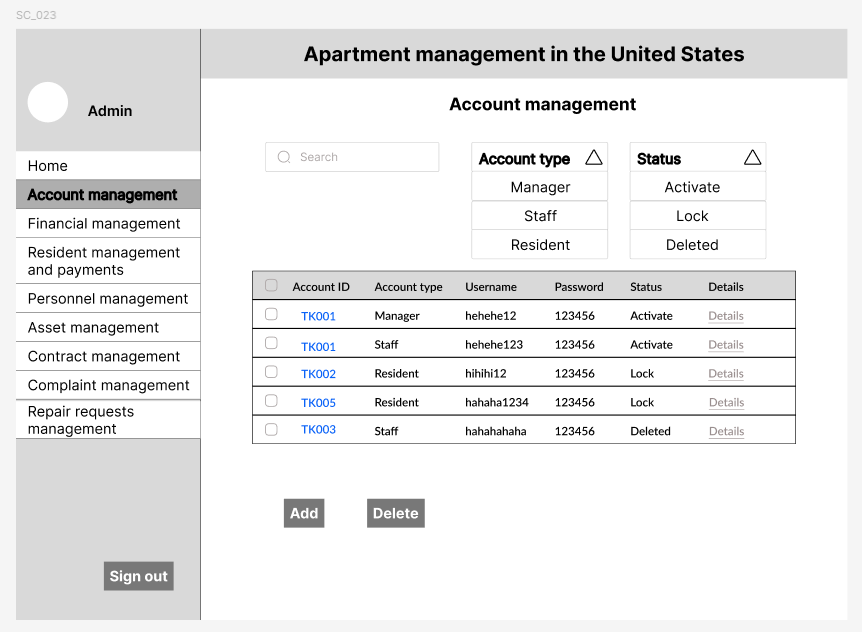
Hình III‑11 Giao diện thanh toán

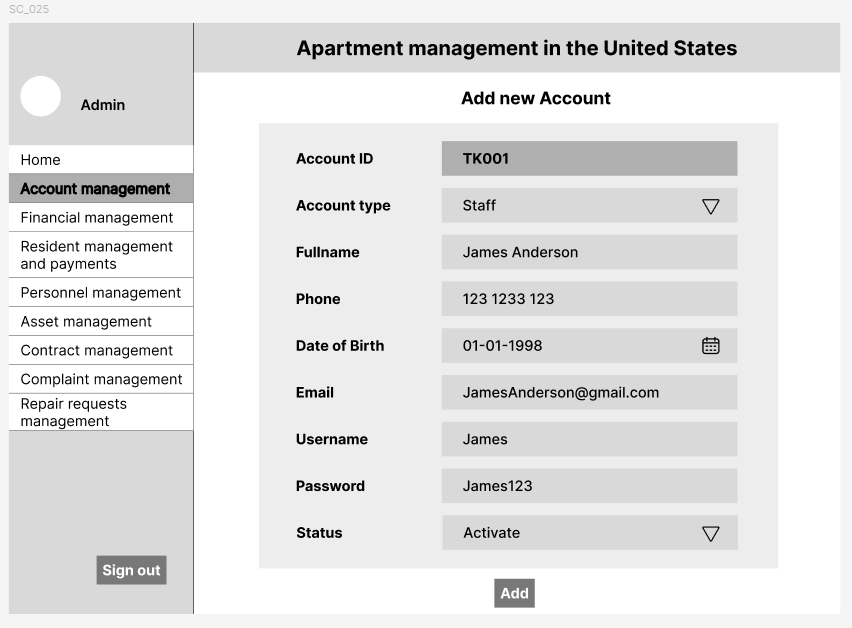
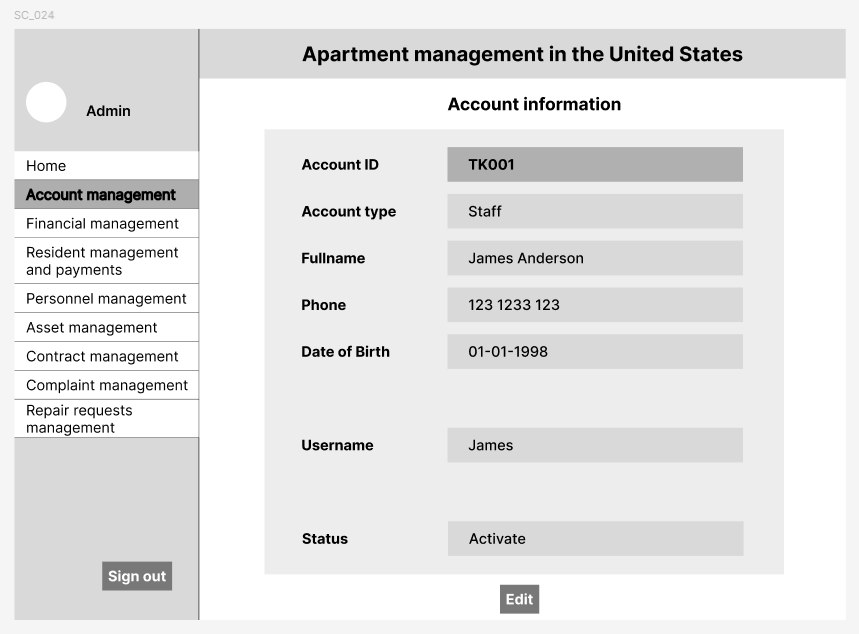
Phần giao diện dành cho Admin – Người quản lý chung cư

Đầu tiên là phần quản lý tài khoản

Bao gồm quản lý tài khoản của người thuê nhà, tài khoản của các quản trị viên, tài khoản của nhân viên.

Admin có quyền chỉnh sửa, thêm, xóa sửa tài khoản. Có thể xem chi tiết tài khoản mô tả thông tin chủ tài khoản.





Hình III‑12 Giao diện quản lý tài khoản

Phần quản lý tài chính.

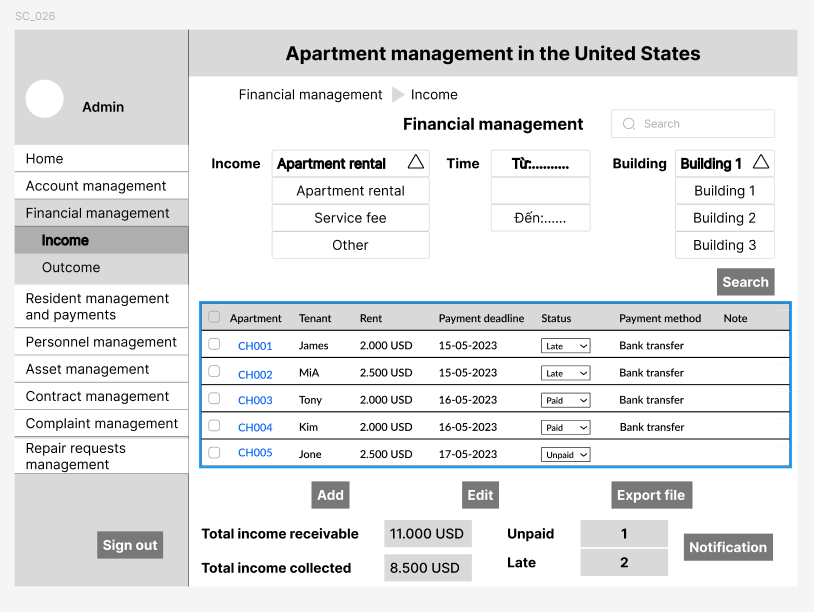
Đầu tiên là phần quản lý nguồn tiền vào:

Người quản trị sẽ vào trang này để xem thống kê các khoản tiền người dân đóng vào. Bao gồm cả tiền đóng trực tiếp hoặc online. Dữ liệu sẽ do admin nhập vào chứ không được tự động hóa.

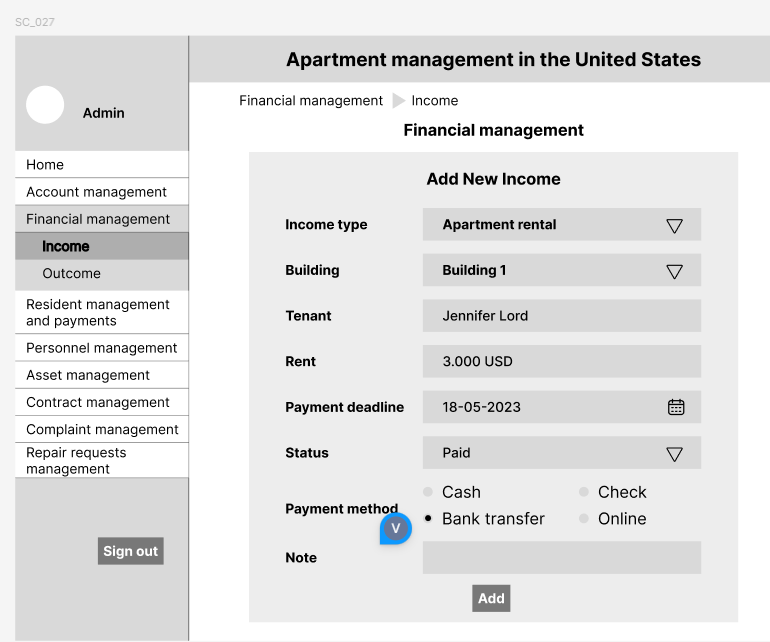
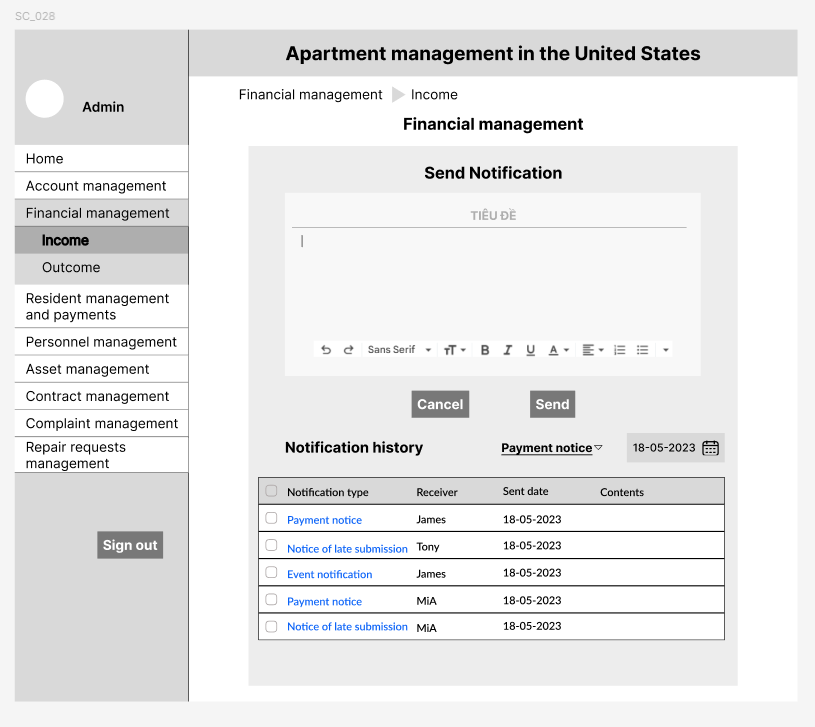
Sẽ có phần thống kê tổng tiền sẽ thu được trong tháng này và số tiền đã thu được. Thống kê được số lượng loại thanh toán chưa thu tiền hoặc trễ hạn.

Giao diện có các tính năng lọc theo loại thanh toán, tìm theo tên, thêm sửa xóa, xuất file excel.

Ngoài ra còn có phần gửi thông báo cho những người chưa thanh toán đủ tiền, giao diện bao gồm cả lịch sử những lần gửi thông báo trước đó. Trạng thái của mục thanh toán có thể thay đổi được là đã thanh toán, chưa thanh toán, hay thanh toán trễ ( sau deadline)



Hình III‑13 Phần Payment

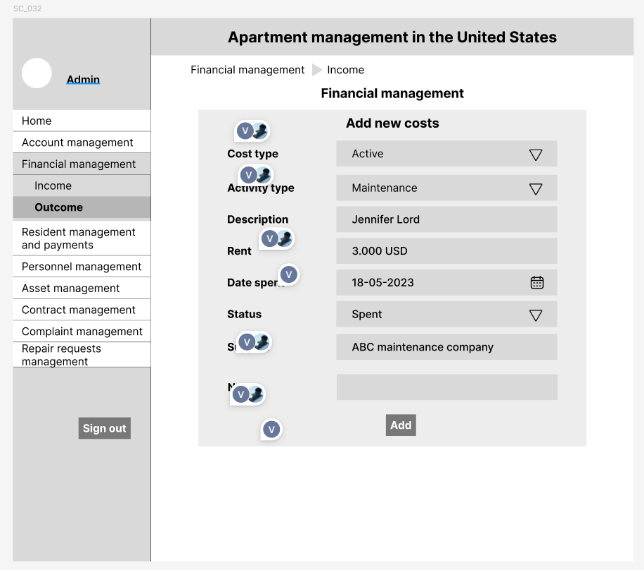
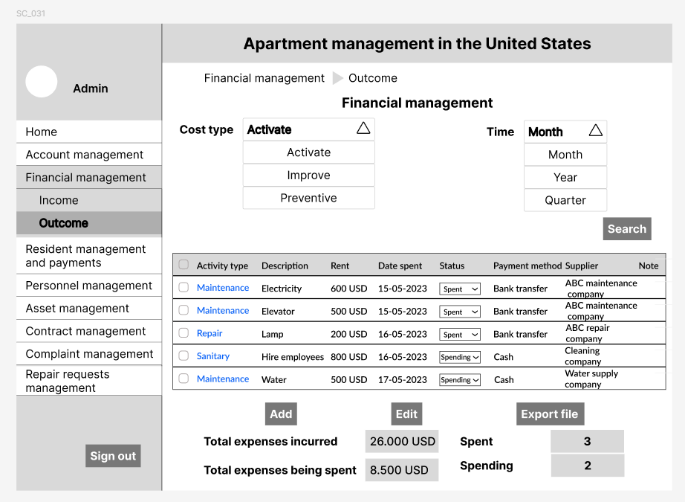
 

Hình III‑14 Thêm income và gửi thông báo

Bên cạnh đó còn có phần quản lý dòng tiền ra như việc sử dụng tiền để trả cho các hoạt động sửa chữa bảo trì thiết bị chung.

Tương tự như phần giao diện Income thì Outcome cũng có những chức năng thêm, xóa, sửa, xuất file excel. Giao diện của trang có chức năng tìm kiếm, lọc theo loại thanh toán, khoảng thời gian.

Có phần thống kê tổng chi phí phát sinh và tổng chi phí đã chi.

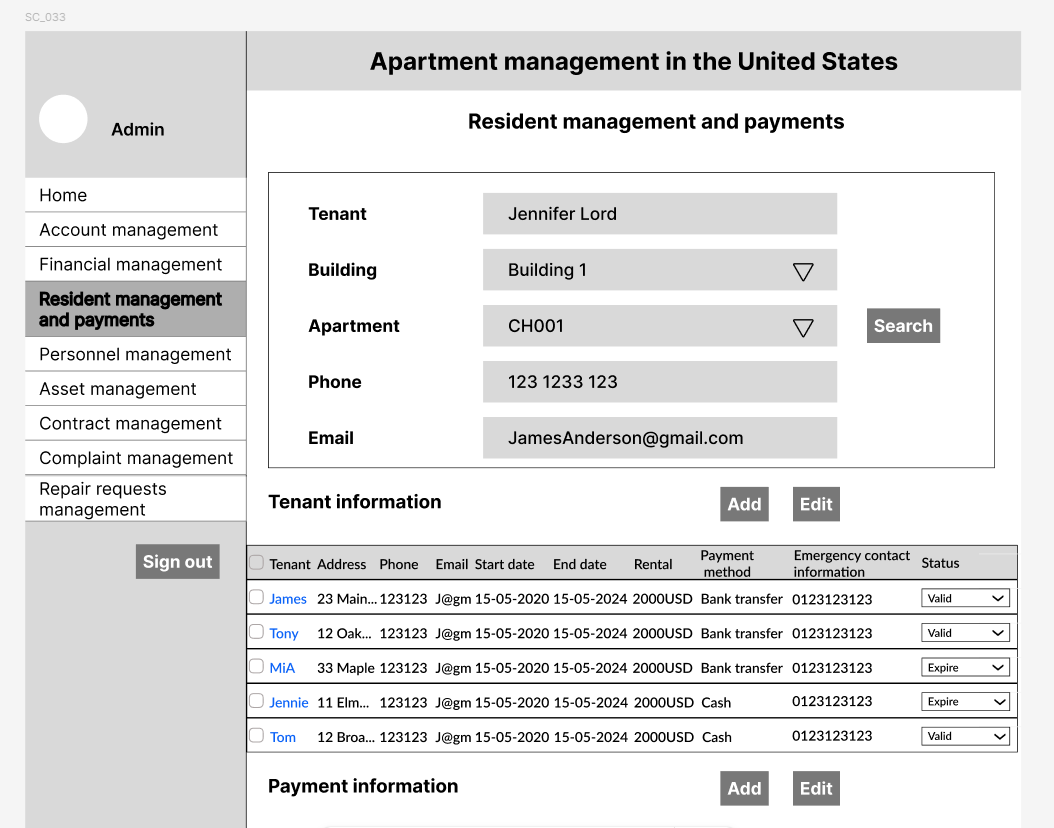


Hình III‑15 Giao diện outcome

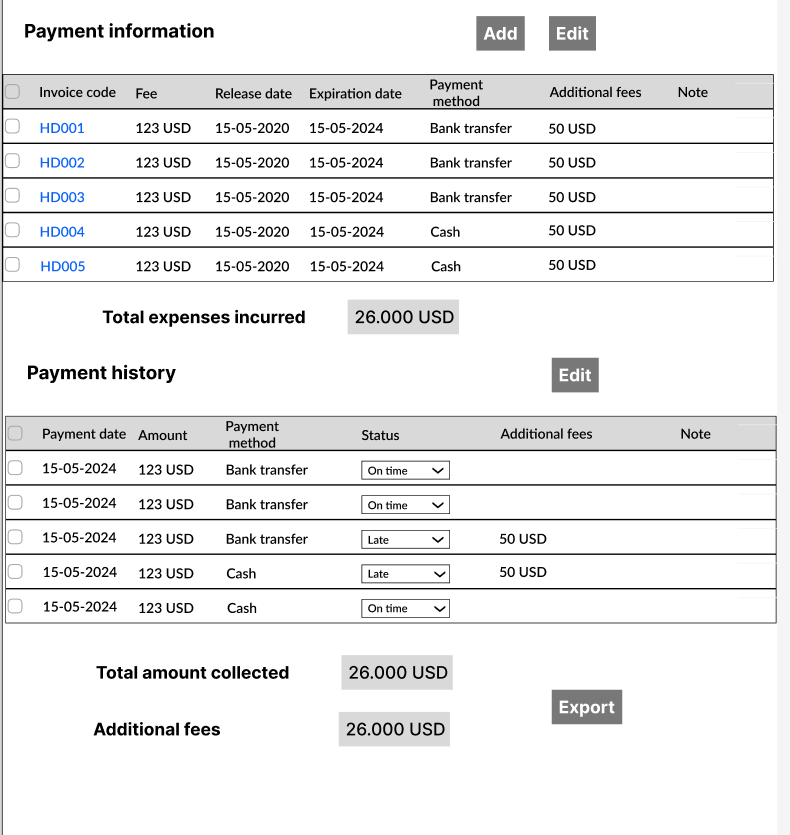
Phần quản lý dân cư và thanh toán

Gồm danh sách những dân cư đang thuê căn hộ, danh sách lịch sử thanh toán, thông tin thanh toán

Trang giao diện có những chức như tìm kiếm bằng cách lọc ra danh sách theo những thông tin nhập vào và có cả chức năng xuất ra danh sách người thuê căn hộ, lịch sử thanh toán thành file excel. Ngoài ra còn các chức năng thêm, xóa, sửa.



Hình III‑16 Giao diện thông tin người thuê

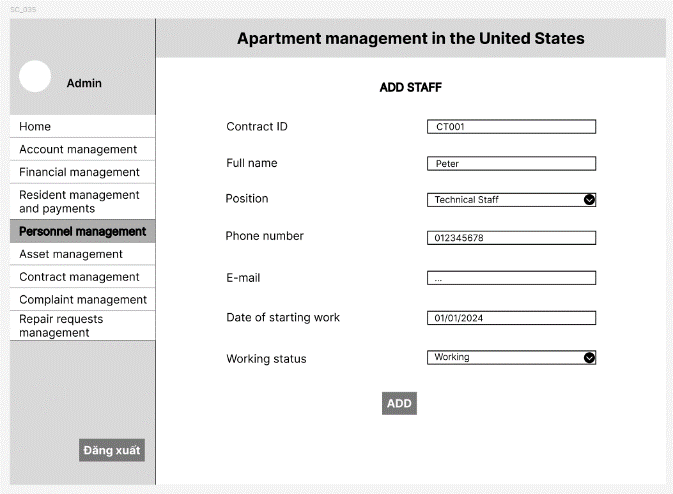
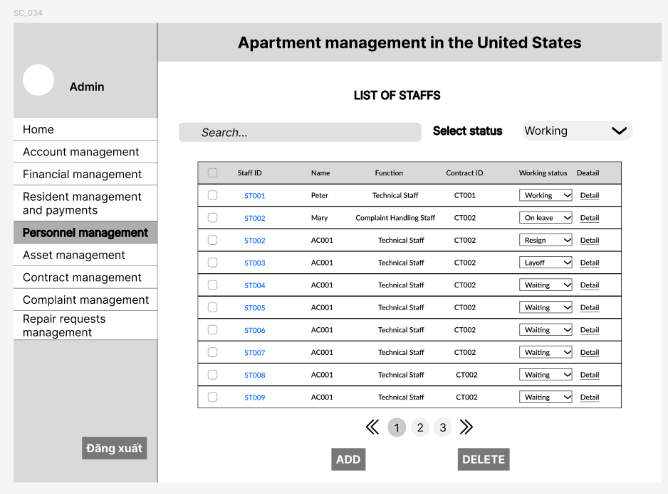


Hình III‑17 Giao diện thông tin thanh toán và lịch sử thanh toán

Phần quản lý nhân lực – (quản lý nhân viên)

Trang giao diện là danh sách các nhân viên, gồm các thông tin và vai trò của nhân viên, trạng thái còn làm việc hay không. Những nhân viên nào không còn làm việc thì có thể cân nhắc xóa khỏi danh sách. Trang web gồm những chức năng thêm sửa xóa.

Có thể xem chi tiết thông tin của riêng nhân viên.

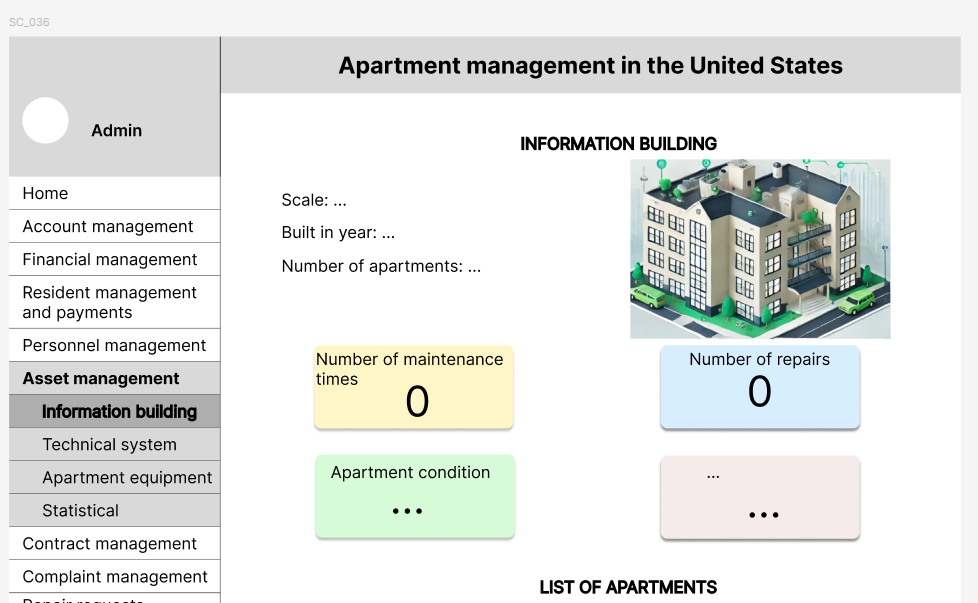


Hình III‑18 Giao diện quản lý nhân sự

Phần quản lý tài sản gồm:

Thông tin tòa nhà: Gồm thông tin của tòa nhà, các thông số bảo trì sửa chữa tòa nhà, các căn hộ bên trong tòa nhà.

Giao diện bao gồm cả danh sách những căn hộ của tòa nhà, Vì đây là dữ liệu cố định nên không có chức năng thêm, sửa, xóa.

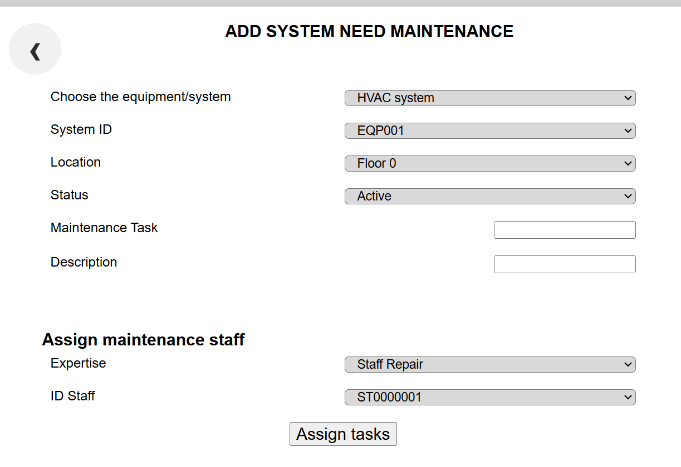
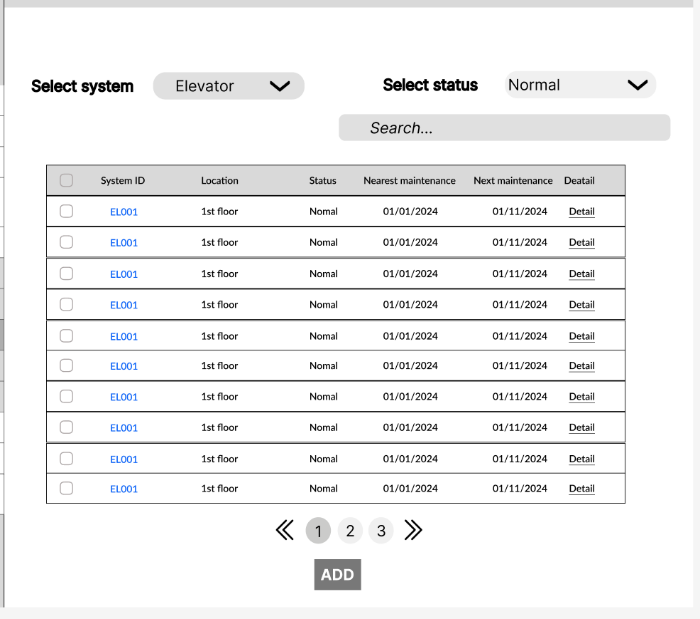


Hình III‑19 Giao diện thông tin tòa nhà

Quản lý hệ thống kỹ thuật: Là phần giao diện quản lý danh sách bảo trì các trang thiết bị chung của tòa nhà, bao gồm các thông tin: Tên thiết bị, Vị trí ( cụ thể ở tầng nào ), trạng thái ( gồm 3 trạng thái là complete – đã bảo trì theo định kì, pending – Chưa giải quyết yêu cầu bảo trì, in process – đang trong quá trình được sửa chữa, bảo trì. Có các chức năng tìm theo tên của thiết bị, lọc ra các thiết bị nào theo loại, các thiết bị đang trong trạng thái nào.

Giao diện mô tả chi tiết thông tin bảo trì.

Giao diện thêm thiết bị cần bảo trì vào danh sách. Khi thêm thiết bị cần sửa chữa sẽ chọn nhân viên nào phải thực hiện công việc đó.



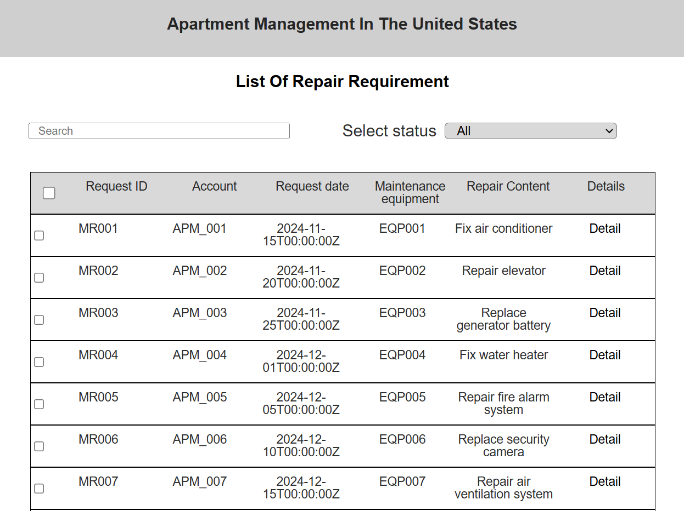
Hình III‑20 Giao diện bảo trì thiết bị

Phần quản lý danh sách yêu cầu sửa chữa thiết bị

Phần giao diện hiển thị danh sách yêu cầu sửa chữa thiết bị riêng của căn hộ được người thuê gửi tới sẽ hiển thị trong này cho admin xem và quản lý.

Giao diện gồm chức năng tìm kiếm trong danh sách yêu cầu, tìm theo nội dung sửa chữa. Không có chức năng thêm vì yêu cầu sửa chữa mới sẽ được thêm vào từ người dân gửi lên.

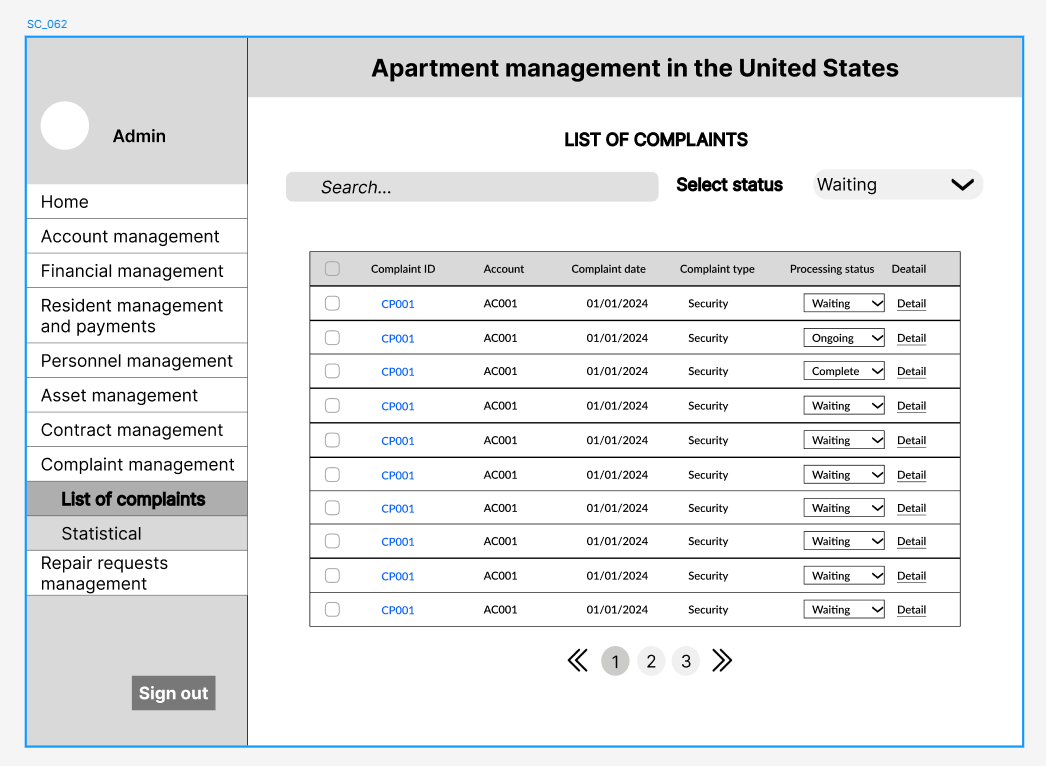
Giao diện chi tiết yêu cầu sửa chữa.

Hình III‑21 Giao diện quản lý yêu cầu sửa chữa

Phần giao diện quản lý danh sách góp ý của người thuê căn hộ:

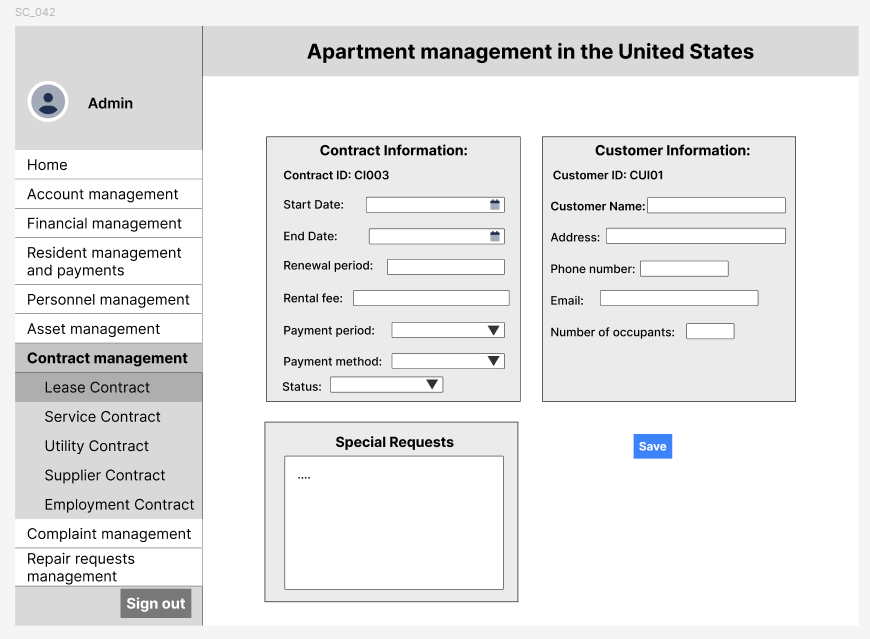
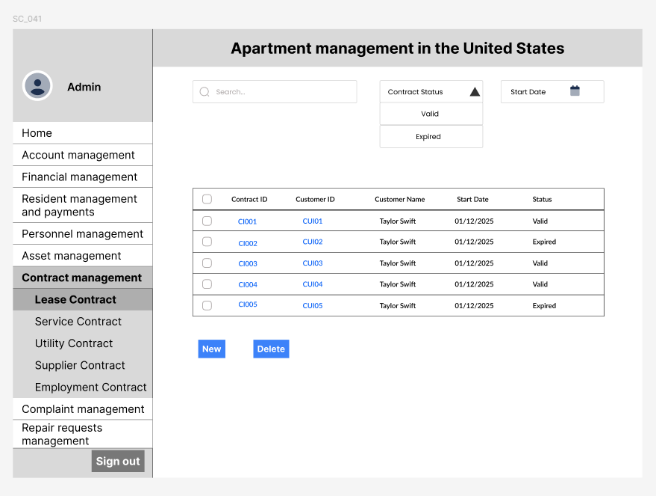
Hiển thị danh sách các góp ý của người thuê.

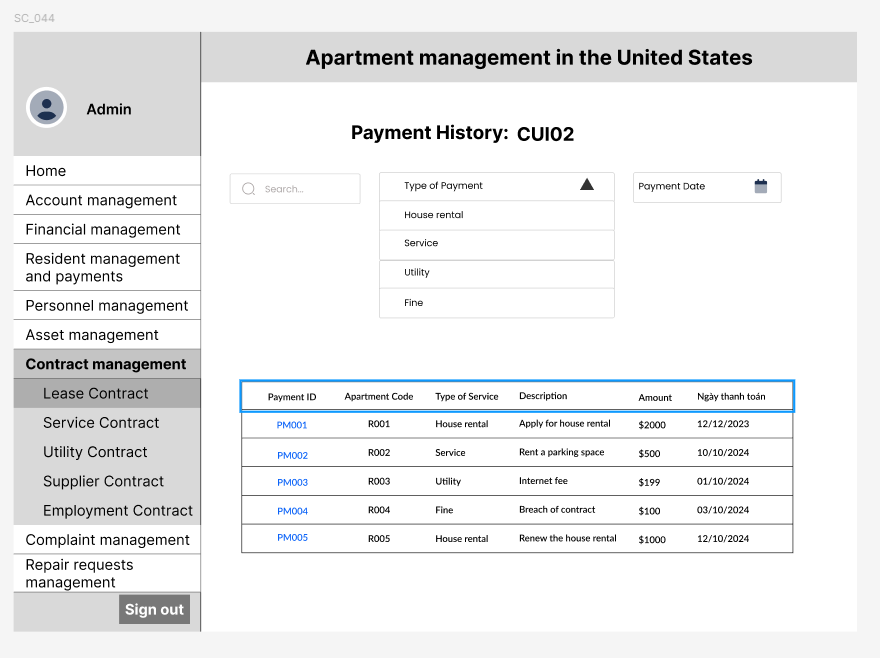
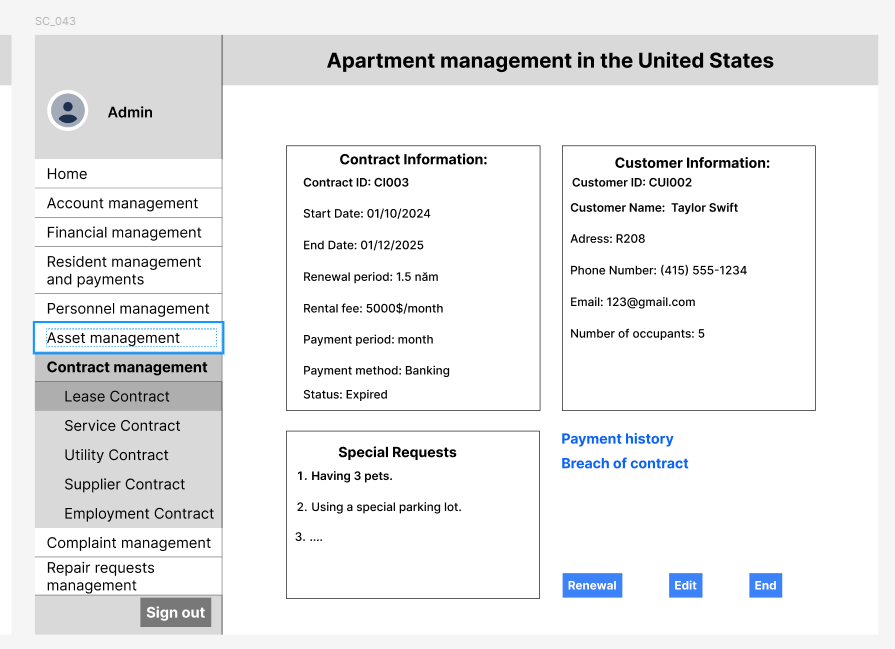


Hình III‑22 Giao diện quản lý danh sách góp ý của người thuê

Phần giao diện quản lý hợp đồng gồm:

Quản lý hợp đồng cho thuê gồm những chức năng: Hiển thị danh sách người thuê căn hộ, tạo hợp đồng người thuê mới, xóa hợp đồng thuê đã hết hạn (không thuê căn hộ nữa), xem thông tin chi tiết người thuê, lịch sử thanh toán, đóng tiền của người thuê, còn có phần phi phạm hợp đồng.





Hình III‑23 Giao diện hợp đồng cho thuê

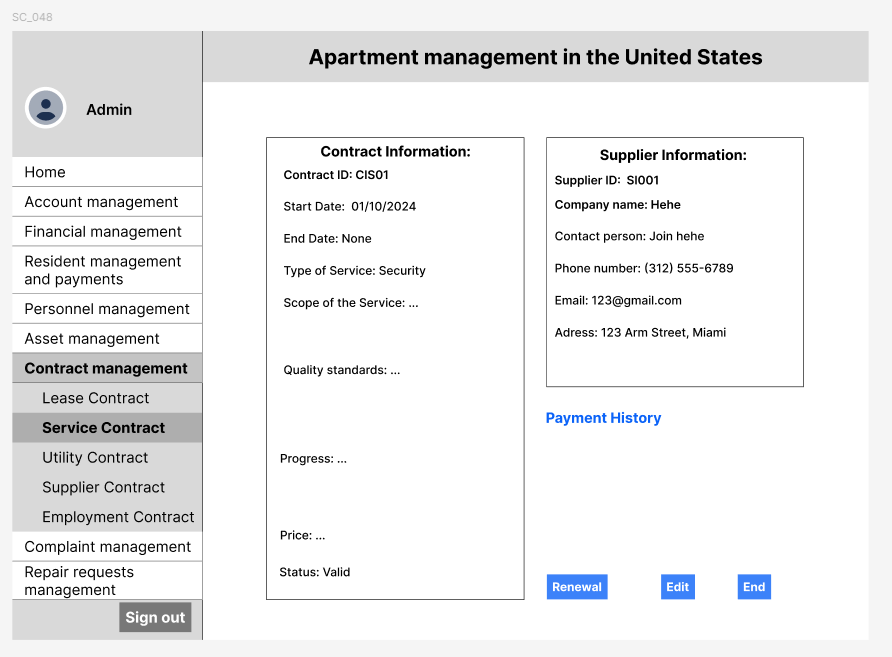
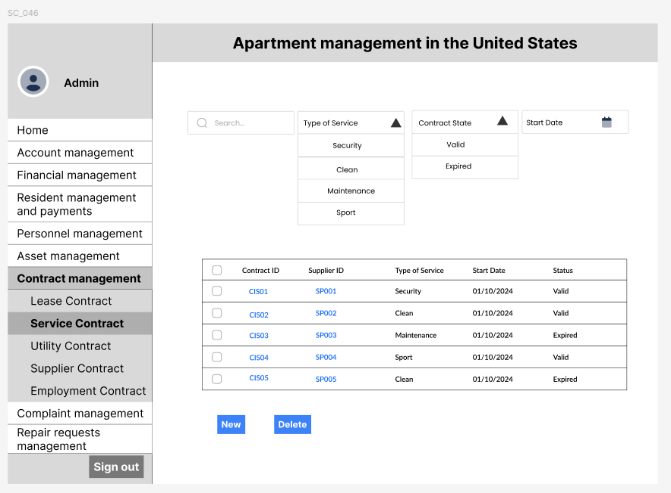


Hình III‑24 Giao diện vi phạm hợp đồng

Quản lý hợp đồng dịch vụ:

Hiển thị danh sách các loại dịch vụ, nhà cung cấp dịch vụ, trạng thái,…

Có chức năng thêm loại dịch vụ mới. Có thêm giao diện lịch sử thanh toán cho các loại dịch vụ đó, chi tiết thông tin của loại dịch vụ.

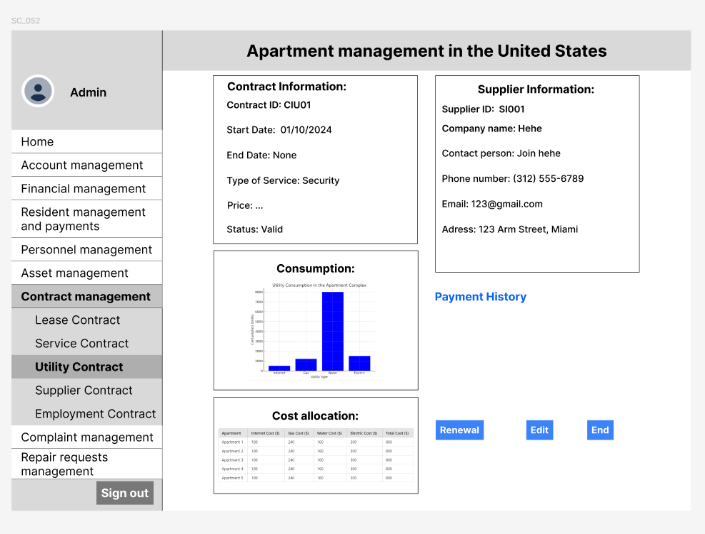
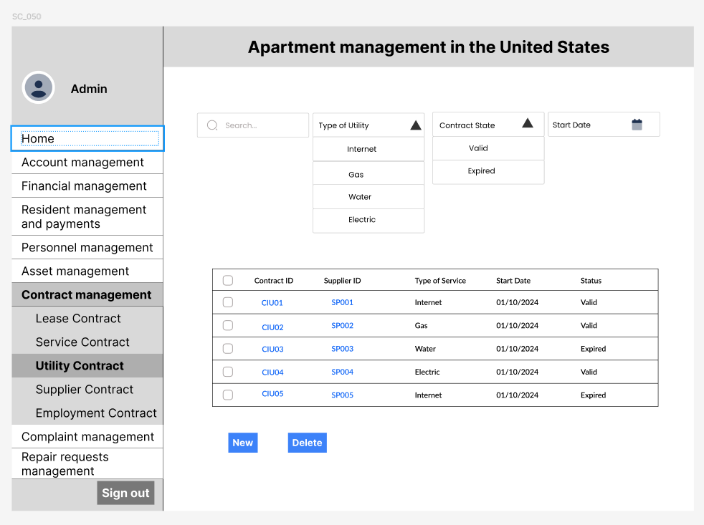


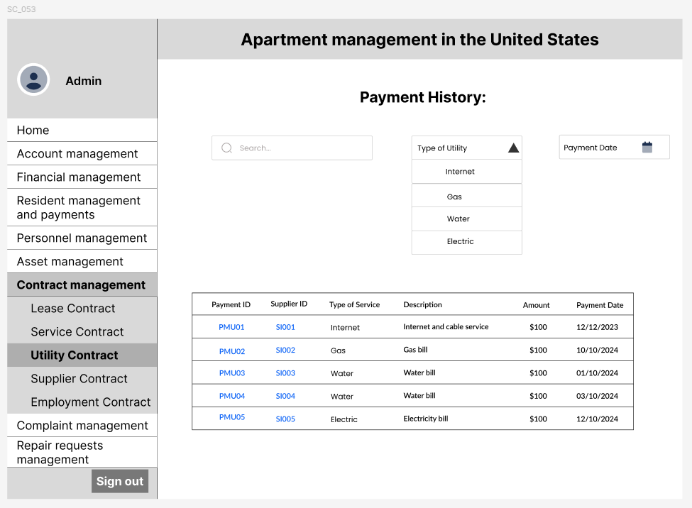
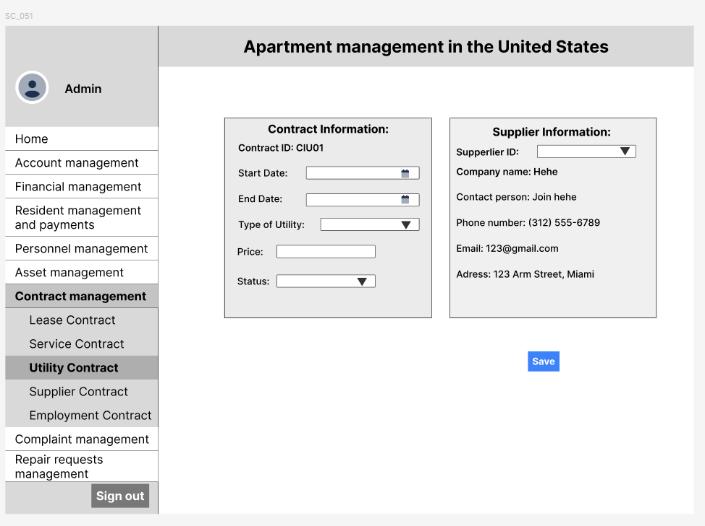
Hình III‑25 Giao diện quản lý hợp đồng dịch vụ

Quản lý hợp đồng tiện ích:

Hiển thị các loại tiện ích như ga, điện, mạng Internet, nước, tên nhà cung cấp, trạng thái, ngày bắt đầu hợp đồng tiện ích,…

Chức năng thêm loại tiện ích mới. Có thêm giao diện lịch sử thanh toán tiện ích, thông tin chi tiết của loại tiện ích.

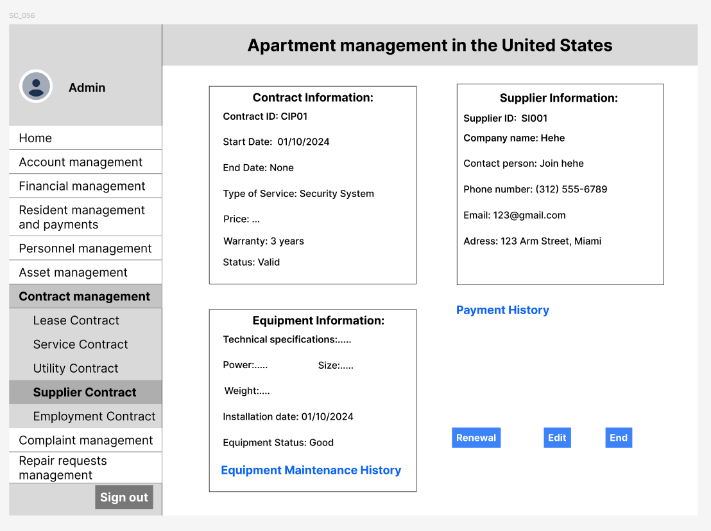
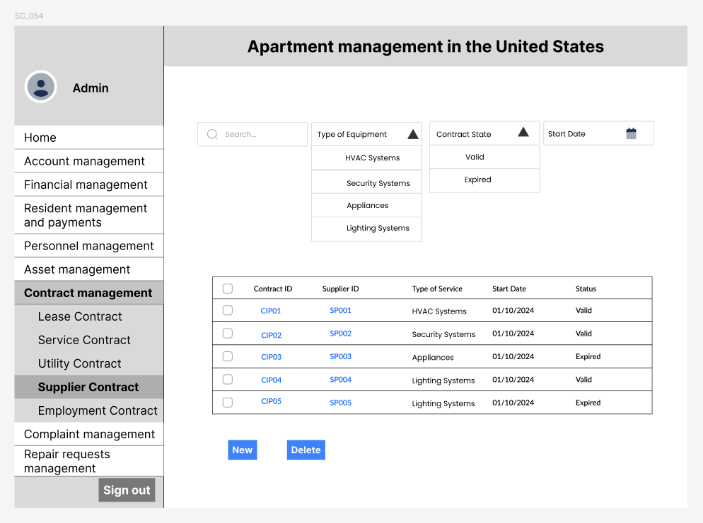




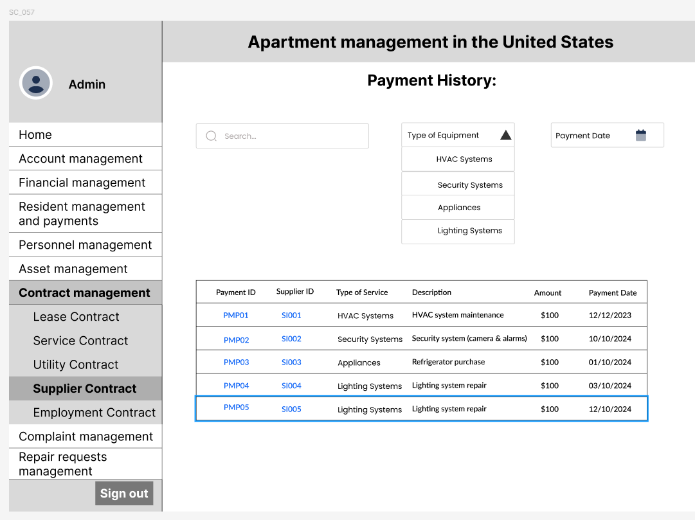
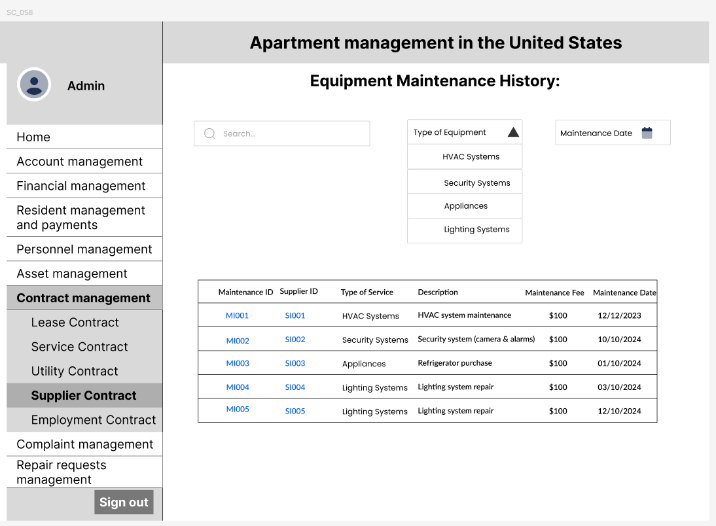
Hình III‑26 Giao diện hợp đồng tiện ích

Quản lý hợp đồng với nhà cung cấp các thiết bị:

Tương tự với các giao diện quản lý hợp đồng khác, xem thông tin các thiết bị được cung cấp-mua bởi hãng nào. Phần lịch sử thanh toán bao gồm các loại chi phí sửa chữa, phí bảo trì định kỳ theo hợp đồng, phí mua thêm thiết bị mới,…



Hình III‑27 Giao diện hợp đồng nhà cung cấp

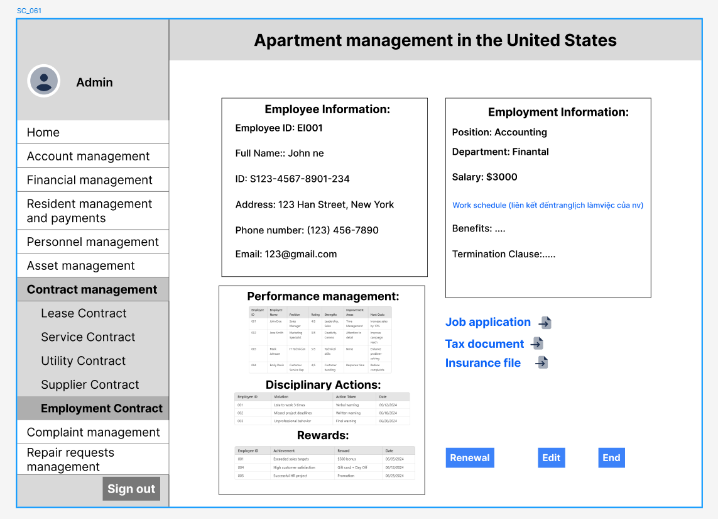
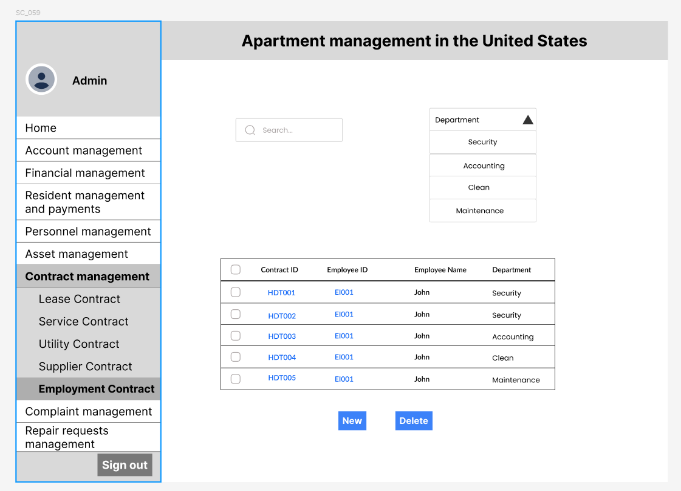
 

Hình III‑28 Giao diện lịch sử thanh toán

Quản lý hợp đồng nhân viên

Giao diện hiển thị danh sách các nhân viên đang ký hợp đồng làm việc, có chức năng lọc theo chức năng của nhân viên, thêm, sửa thông tin nhân viên, xóa thông tin nhân viên đã nghỉ việc.

Chi tiết hợp đồng với nhân viên hiển thị thông tin chi tiết, mức lương, đãi ngộ,…

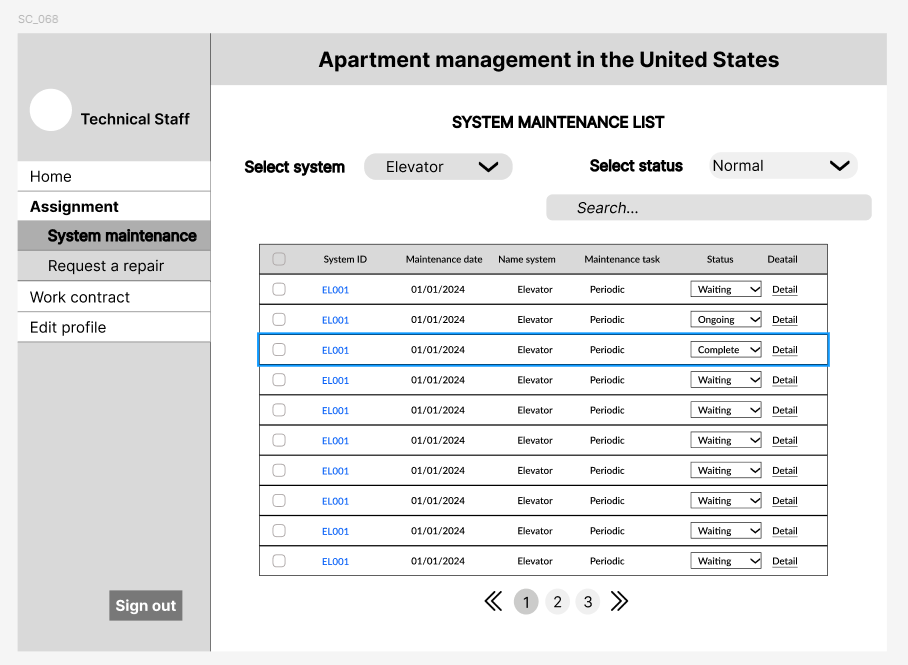


Hình III‑29 Giao diện hợp đồng nhân viên

Phần giao diện dành cho nhân viên.

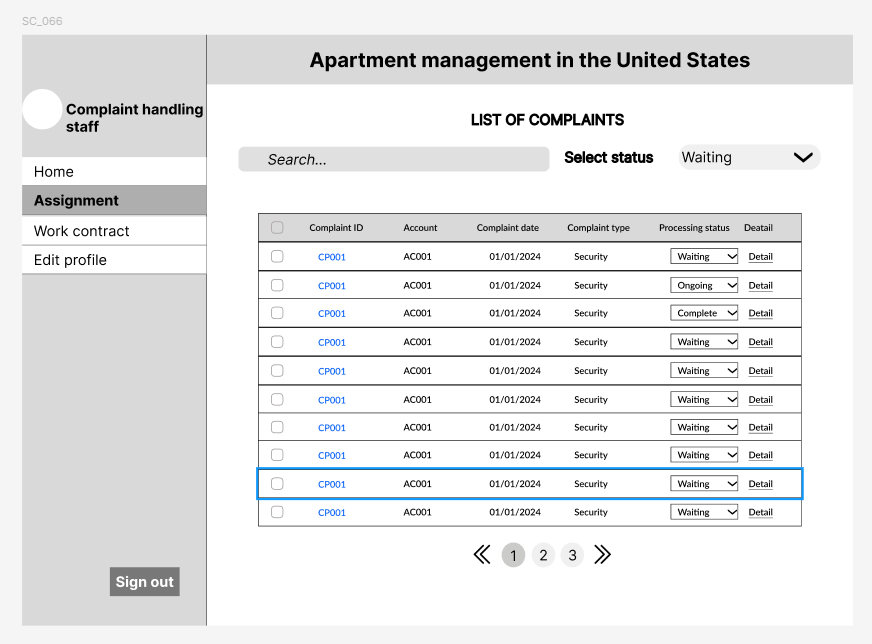
Phần giao diện cho nhân viên sẽ tùy theo chức năng của nhân viên đó mà có giao diện khác nhau.

Nhân viên kỹ thuật: Phần giao diện cho nhân viên kỹ thuật sẽ là danh sách các thiết bị đang được yêu cầu sửa chữa, bảo trì



Hình III‑30 Giao diện của nhân viên kỹ thuật

Đối với nhân viên xử lý khiếu nại phần giao diện sẽ là danh sách các khiếu nại của người thuê, nhân viên sẽ căn cứ vào danh sách và trạng thái của khiếu nại để giải quyết cho người thuê căn hộ



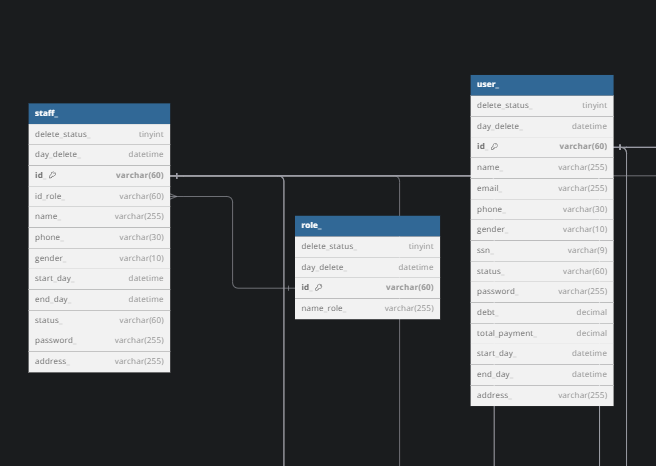
Hình III‑31 Giao diện của nhân viên xử lý khiếu nại

## Thiết Kế Cơ Sở Dữ Liệu Dự Án

Cơ sở dữ liệu của dự án được tôi cùng leader trực tiếp thiết kết và xây dựng, đến lúc nhập dữ liệu mẫu thì các thành viên khác mới tham gia. Do lúc thiết kế và xây dựng chỉ có 2 người nên việc sai lệch và không tối ưu đã xảy ra dẫn đến trong suốt quá trình làm cơ sở dữ liệu phải cập nhật tận 3, 4 lần mới hoàn tất dẫn đến các Dev phải sửa đổi liên tục gây mất thời gian toàn đội. Các cơ sở dữ liệu lúc chưa hoàn thiện đều bị xóa nên ở phần này ta sẽ đi vào phân tích và nói về cơ sở dữ liệu lúc đã hoàn thiện nhất. Và quy tắc đặt tên của cơ sở dữ liệu theo nhóm thống nhất sẽ là nối nhau bởi dấu \_ và có dấu \_ ở cuối tên bản tương tự với tên của cột(collum).

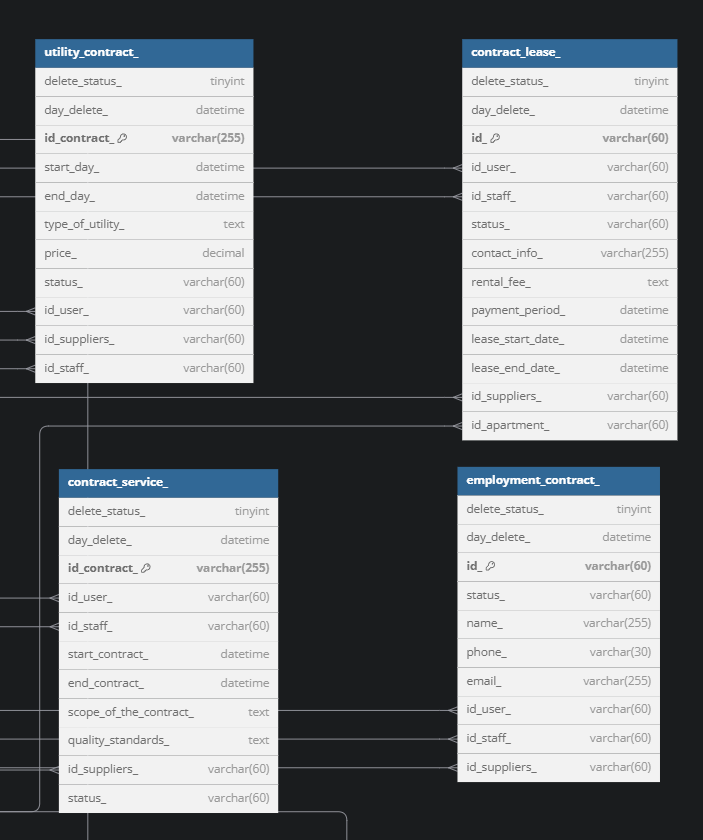
Cơ sở dữ liệu với 29 bảng tổng cộng trải dài trên chức năng của cả 3 phân quyền chính là công dân chugn cư(user), nhân viên trong chung cư(staff) và quản lý cao nhất của chung cư(admin). Việc đi qua 29 bản và mô tả từng bản sẽ rất mất thời gian nên ta sẽ đi qua theo chức năng của các bảng:

Các bảng xác định danh tính và phân quyền : bảng user\_, bảng staff\_ bảng role\_, các bảng này lưu trữ chủ yếu là thông tin của người dùng, nhân viên, xác chứa các thông tin được dùng để xác định danh tính cho một người dùng, các thông tin cơ bản như quyền hạn, tên, tuổi, công nợ các thông tin cá nhân của người dùng, nhân viên, chủ sở hữu, bảng user\_ sẽ quản lý các thông tin của công dân, và công dân thuộc không có bất kì quyền hạng nào trong quản lý hệ thống nên sẽ không liên kết với bảng role\_, còn bảng staff\_ thì chỉ lưu các thông tin có liên quan tới nhân viên, quản lý nên sẽ có liên kết với bảng role\_, bảng role\_ sẽ chứa các thông tin liên quan tới chức vụ của nhân viên.



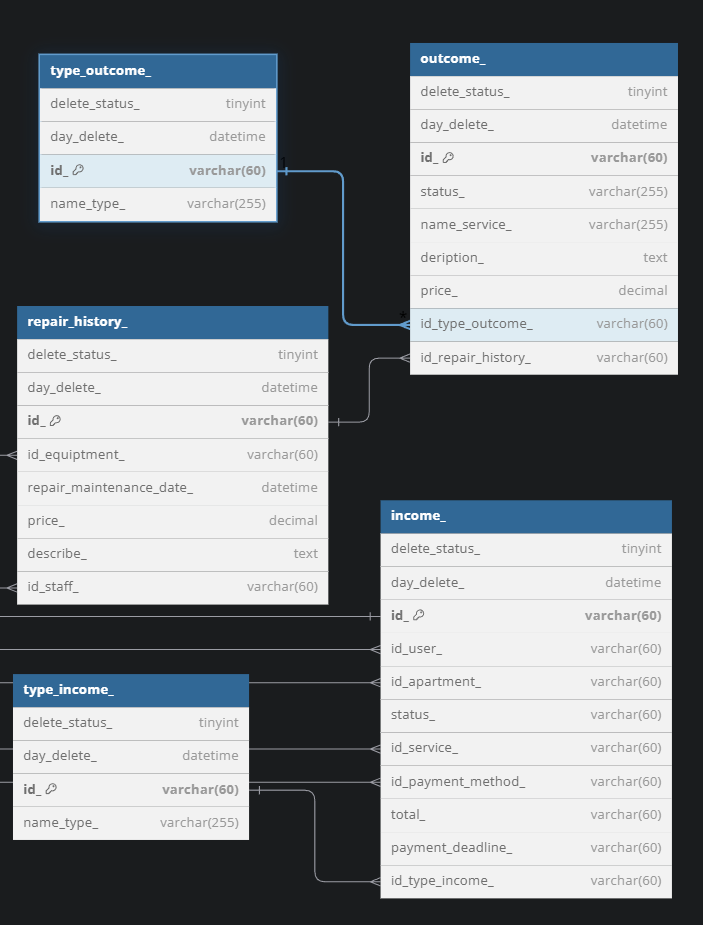
Hình III‑32 bảng user\_, staff\_, role\_ trong cơ sở dữ liệu

Các bảng liên quan đến hợp đồng và quyền lợi của công dân : bảng contract\_lease\_, bảng utility\_contract\_, bảng employment\_contract\_, bảng contract\_service\_ chứa thông tin về các hợp đồng của người dùng khi thuê nhà, các thông tin về hợp đổng của chung cư với bên cung cấp thiết bị, hợp đồng của công ty với bên dịch vụ(như dọn phòng, nấu ăn, …), tiện ích(phòng gym, hồ bơi, …). Ngoài việc được lưu trữ trên cơ sở dữ liệu thì bên quản lý chung cư sẽ còn giữ cả các bản hợp đồng giấy để đối chiếu khi có vấn đề. Và các nhân viên về quan lý lợp đồng hay chủ sỡ hữu mới có quyền truy xuất sửa chữa hay xóa các bảng này.



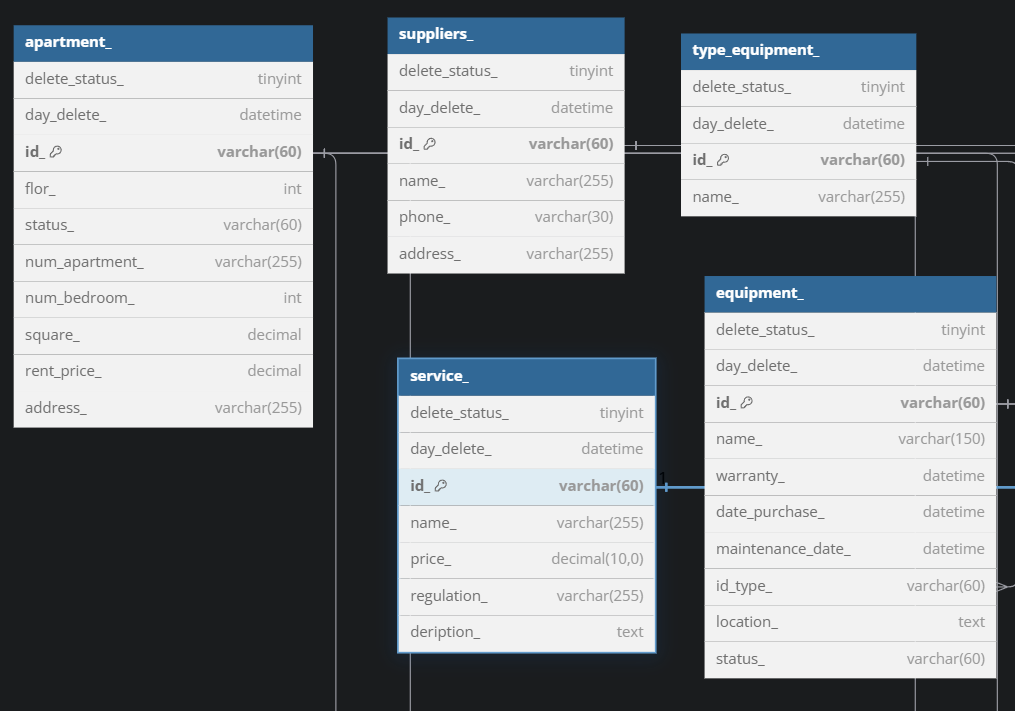
Hình III‑33 bảng contract\_lease\_, bảng utility\_contract\_, employment\_contract\_, contract\_service\_ trong cơ sở dữ liệu

Các bảng quản lý bên thu nhập, chi phí : bao gồm bảng outcome\_, income\_, type\_income\_, type\_outcome\_, và bảng repair\_history\_, bản income\_ chịu trách nhiệm cho các thông tin về thu nhập có liên kết với các bản thanh toán phía người dùng như pyment\_, service\_, pyment\_method\_ để xác minh, và bản type\_income\_ để xác định xem đầu vào của thu nhập đến từ đâu. Bảng outcome\_ quản lý thông tin cho chi phí mất mát có liên kết với bảng repair\_history\_ để xác định chi phí mất mát khi sửa chữa và bảo trì thiết bị tương tự với type\_outcome\_ xác định xem chi phí đầu ra phát sinh ra do đâu.



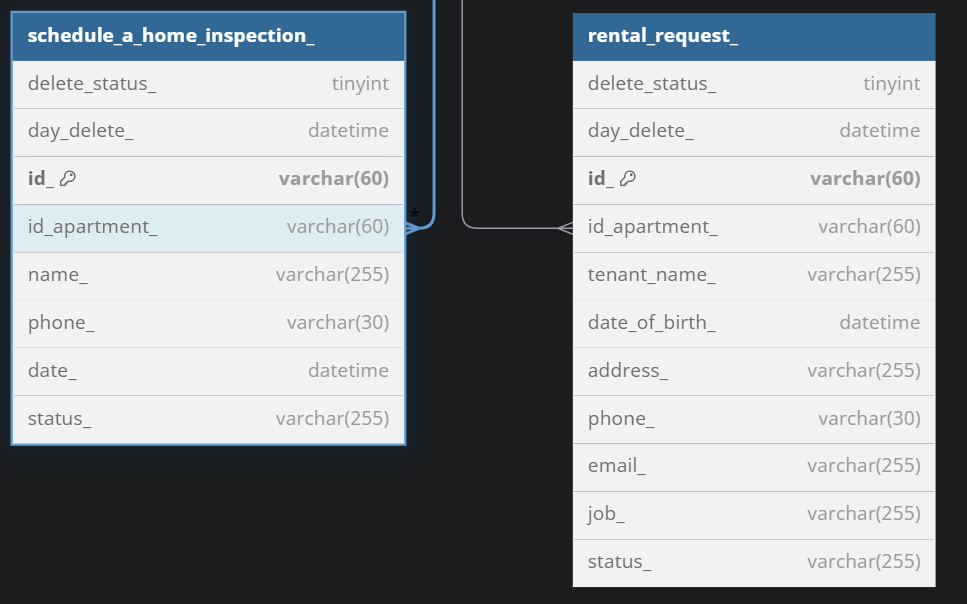
Hình III‑34 bảng imcome\_, outcome\_, type\_income\_, type\_outcome\_, repair\_history\_ trong cơ sở dữ liệu

Các bảng quản lý thông tin chung cư , căng hộ, nhà cung cấp, thiết bị và dịch vụ : bao gồm bảng apartment\_ chứa thông tin của tất cả các căng hộ trong chung cư như địa chỉ, số phòng, giá tiền khi thuê. Bảng suppliers\_ chứa thông tin của nhà cung cấp. Bảng service\_ chứa các thông tin của các dịch vụ riêng, chung của chung cư. Bảng equipment\_ và type\_equipment\_ chứa thông tin của các thiết bị trong chung cư như các thiết bị chung như than máy, đèn hành lan, …. và cả các thiết bị mà chung cư phụ trách trong mỗi căng hộ như máy lạnh, máy sưới, máy nước nóng, ….



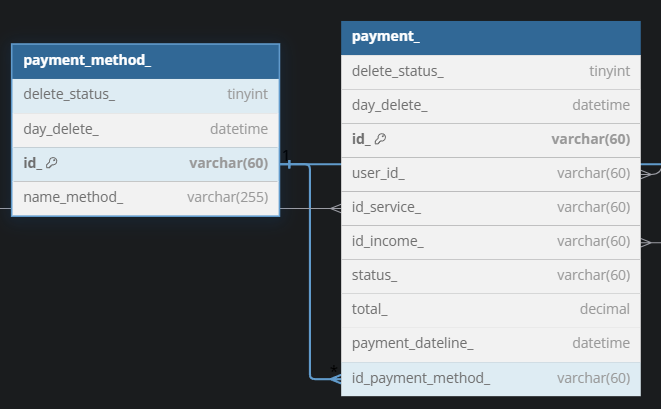
Hình III‑35 các bảng apartment\_, suppliers\_, service\_, equipment\_ và type\_equipment\_ trong cơ sở dữ liệu

Các bảng quản lý thông tin cho việc gửi yêu cầu xem nhà, thuê nhà : bảng rental\_request\_ và schedule\_a\_home\_inspection\_ với thông tin lưu trữ là thời gian, tên, sdt của khách hàng muốn xem nhà, thuê nhà.



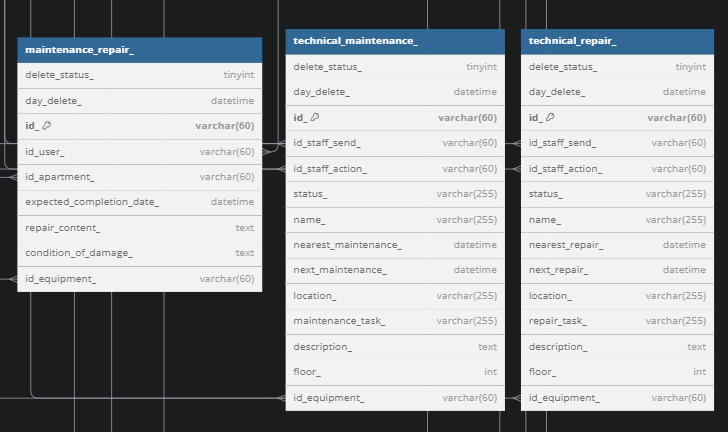
Hình III‑36 bảng rental\_request\_, schedule\_a\_home\_inspection\_ trong cơ sở dữ liệu

Bảng lưu trữ thông tin thanh toán của người dùng : bảng payment\_ và payment\_method\_ chứa thông tin thanh toán của người dùng như đóng tiền thuê, đóng viền dịch vụ, và phương thức thanh toán như chuyển khoản, đóng trên web, đóng tiền mặt.



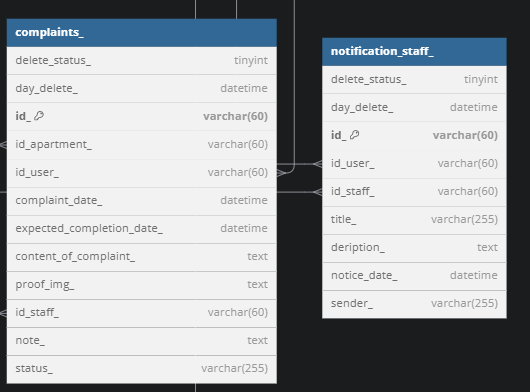
Hình III‑37 bảng pyment\_ và payment\_method\_ trong cơ sở dữ liệu

Các bảng quản lý thông tin cho việc bảo trì và sửa chữa : bao gồm bảng maintenance\_repair\_, là bản lưu thông tin của người dân khi yêu cầu sửa chửa, yêu cầu bảo trì, và bảng technical\_repair\_,technical\_maintenance\_ lưu thông tin của các nhân viên khi nhận được yêu cầu tức bảng maintenance\_repair\_ được tạo, và 2 bảng dó là yêu cầu của nhân viên quản lý gửi tới các nhân viên phụ trách mảng sửa chữa và bảo trì để bắt đầu giải quyết.



Hình III‑38 bảng technical\_maintenance\_, technical\_repair\_, maintenance\_repair\_ trong cơ sở dữ liệu

Các bản quản lý góp ý phản hồi của công dân và thông báo của nhân viên cho công dân, bảng complaints\_ chứa thông tin về phản hồi hay góp ý của công dân, còn bảng notification\_staff\_ chứa thông tin của các thông báo mà nhân viên gửi cho các công dân trong căng hộ.

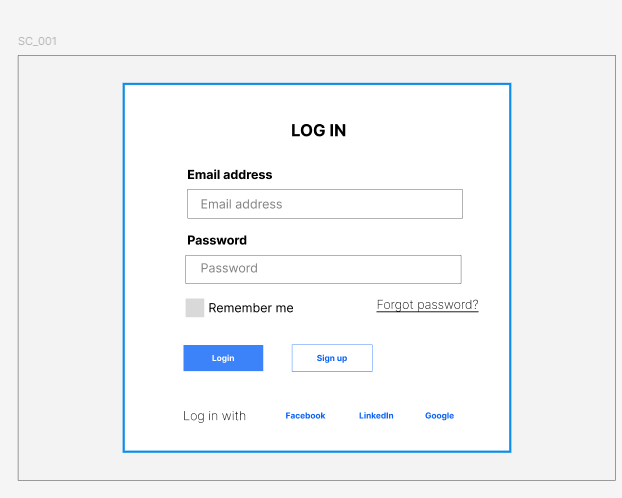
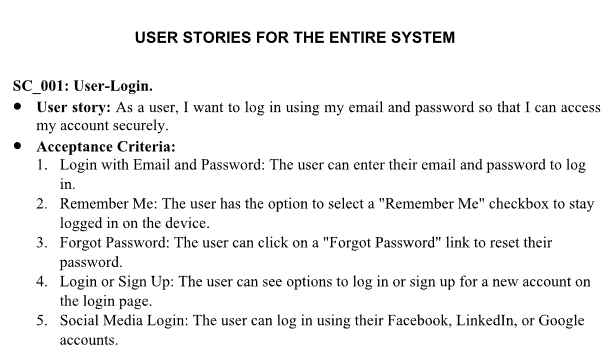


Hình III‑39 bảng complaints\_ và notification\_staff\_ trong cơ sơ dữ liệu

## Quá Trình Sửa Chửa, Cải Thiện Giao Diện, Cơ Sở Dữ lIệu

Ở phần trên ta đã nói về giao diện và cơ sở dữ liệu tuy nhiên đó chỉ là phần giao diện và cơ sở dữ liệu sau chỉnh sửa, ở trước đó khi thiết kế cơ sở dữ liệu và giao diện thì nhóm Ba cùng tôi, nhóm trưởng và vài thành viên hỗ trợ đã phải sữa chửa và chắc lọc rất nhiều lần. Ở phần này chúng ta sẽ nói qua về những lần sửa chữa của giao diện và cơ sở dữ liệu.

Đầu tiên là về giao diện, giao diện được xây dựng hoàn toàn trên figma nên những gì thay đổi ở figma thì đều tính là một lần sửa đổi, lần một chính là thay đổi theo đúng các chức năng trên User story với kiến trúc mỗi một chức năng trên User story thì sẽ tương ứng với một màn hình trên figma với cách đặc tên là SC\_001 cho chức năng và màn hình 1.

Hình III‑40 ví dụ cho màn hình 1 trên figma tương ứng với chức năng 1 trong user story

Lần 2 thay đổi giao diện là khi bắt đầu code, lúc code thì nhóm Dev thấy có rất nhiều vấn đề phía giao diện cùng cơ sỡ dữ liệu bị thiếu thông tin, thừa thông tin giống như ở trang thanh toán của người dùng có chức năng là gửi thông báo cho nhân viên hay admin tuy nhiên lúc làm cơ sở dữ liệu lại bỏ qua phần đó, và một phần cũng do khi thiết kế ra giao diện ban đầu và User story thì nhóm Dev không thật sự tham giá góp ý quá nhiều dẫn đến việc nếu làm theo 100% bên giao diện thì cơ sở dữ liệu bị dư thừa quá nhiều, và các liên kết của các bảng trở nên quá rắc rối. Do vậy nên phía cơ sở dữ liệu sẽ có những phần sai lệch so với giao diện lúc này thì nhóm Dev sẽ theo phương án đề ra lúc hợp là code theo cơ sở dữ liệu còn giao diện thì thông báo cho đội Ba để sửa đổi lúc cần thiết tuy nhiên đây là giai đoạn giữa – cuối dự án các bạn phía Ba không quá quan tâm đến dự án và không để ý sửa lại giao diện.

Còn bên cơ sở dữ liệu thì chỉ có 2 thành viên làm chính là tôi và nhóm trưởng, có một vài thành viên giúp tuy nhiên chỉ hỗ trợ ở mức cho ý kiến là nhiều chứ không trực tiếp tham gia xây dựng, lý do cho việc chỉ có vài thành viên tham gia xây dựng là do một phần đội Ba, Tester,Fontend thì không thông thạo và hiểu về cơ sở dữ liệu nên không tham gia xây dựng còn đội Backend thì có người thông thạo người thì vừa học nên nếu như làm theo phương án đề ra lúc đầu là mọi người trong nhóm Backend cùng tạo bảng trên file trang tính của google và bàn luận, sửa chửa của nhau hoàn toàn không khả thi, minh chứng là việc file trang tính lúc đầu cực kì hỗn loạn, bảng thì thiếu thông tin, bảng thì không có liên kết với các bảng khác, theo tiêu chí dùng kiến trúc Shared database thì một hệ thống cơ sở dữ liệu phải liên kết chặc chẽ với nhay nên việc các bảng vô cùng rời rạc là không chập nhận được lúc này thì nhóm trưởng mới quyết định dùng phương án giải quyết là mọi người sẽ không tham gia nữa mà do tôi cùng nhóm trưởng thiết kế chính và các bạn cứng cơ sở dữ liệu sẽ góp ý trong quá trình xây dựng. Và chỉ vì có 2 người tham gia xây dựng thiết kế nên phải sửa khá nhiều lần.

Lần đầu là file trang tính ban đầu do tất cả thiết kế chung ở lần sửa chữa đâu tiên là bỏ hoàn toàn cơ sỡ dữ liệu cũ và thiết kế một hệ cơ sở mới hoàn toàn.

Lần 2 là sửa chữa theo ý của menter sau khi review, cụ thể là tổng thể cơ sở dữ liệu bị thiếu thông tin lịch sử của các thiết bị bảo trì, các chi phí đầu ra, đầu vào chưa hợp lý và cả phần công nợ, và lịch sử thanh toán của người dùng (công dân chung cư).

Lần 3 là lúc thống nhất tên cho cơ sở dữ liệu, trong lúc thiết kế thì việc đặc tên bị không đồng bộ nên phải sửa lại theo một quy chuẩn cho cả hệ thống.

Lần 4 là lúc menter nhắc nhở là khi thiết kế hệ cơ sở thì các chức năng như xóa sẽ không thực sự xóa mà chỉ tiến hành thay đổi trạng thái cho dòng dữ liệu đó, giống như cơ chế thùng rác trên pc, đó là xóa lần 1 thì không thật sự xóa đi mà chỉ thai đổi và xóa thật khi có tương tác từ người dùng hay hết hạn thời gian. Việc này cần phải thêm vào mỗi bản 2 cột(collum) là trạng thái xóa và thời gian xóa.

Sau đó thì cơ sở dữ liệu còn phát sinh từ 4-5 lần sửa chửa nhỏ nhưng không đáng kể như thay kiểu dữ liệu, thêm một cột nào đó trong bảng,….

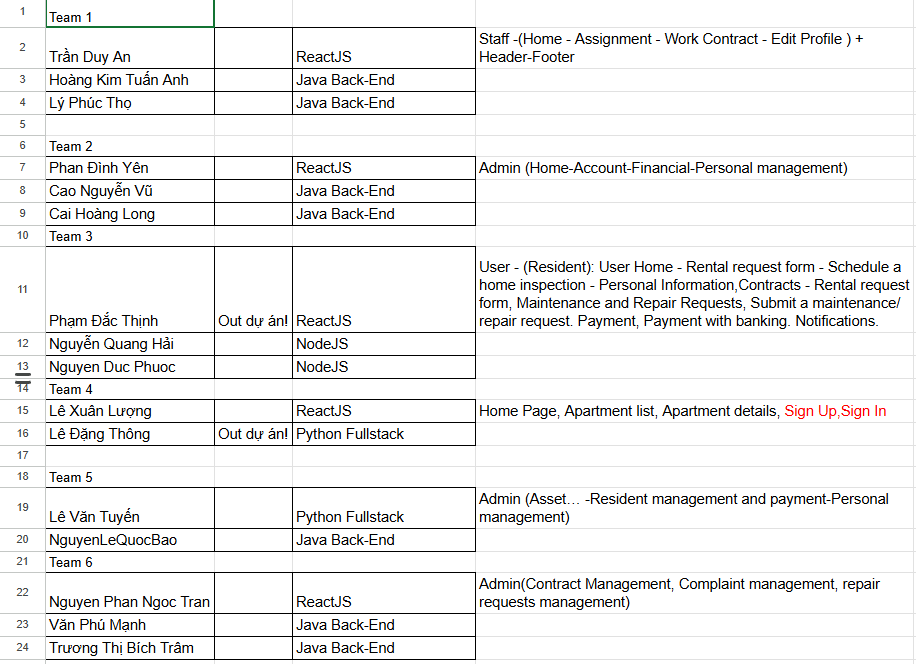
## Viết Test Case

Test Case giúp phát hiện lỗi, xác minh tính toàn vẹn của API, đảm bảo rằng các yêu cầu kinh doanh được đáp ứng, và cải thiện chất lượng phần mềm.

Đây là công việc chính của Tester và nhóm Dev chỉ tham gia góp ý chứ không có quyền tham gia trược tiếp làm việc nên ở đây tôi chỉ giới thiệu chứ không mô tả nhiều về phần này.

## Phân Chia Công Việc Và Lập Nhóm

Như đã nói trước đó thì sau cuộc họp Sprint plan với mục đích chính là phân chia công việc cho các Dev và chia nhóm nhỏ để tiền làm việc. Theo như thống nhất từ cuộc họp thì nhóm trưởng sẽ toàn quyền quyết định phân công nên cuối cùng nhóm đã được chia thành 6 nhóm với phân công như sau.



Hình III‑41 phân công công việc sau buổi họp Sprint plan của leader(nhóm trưởng)

Tôi thuộc nhóm 5 cùng nhóm với nhóm trưởng, cơ chế phân công sẽ là mỗi nhóm từ 1-2 Backend và 1 Fontend.

## Công Việc Của Cá Nhân

Lúc đầu tôi đã nói do bị thiếu hụt nhân lực vì các bạn thực tập nghĩ ngang nên tôi phải code cả Fontend và Backend, và trong nhóm nhỏ gồm tôi và nhóm trưởng thì nhóm trưởng cũng code Fullstack nên tôi vfa nhóm trưởng quyết định chia theo giao diện mỗi người sẽ chọn một nửa giao diện và code cả giao diện và chức năng. Cụ thể tôi sẽ code các trang SC\_033, SC\_034, SC\_035, SC\_062, SC\_063 với :

SC033 : quản lý thanh toán và người dùng của admin bao gồm thêm xóa sửa thông tin người dùng và cập nhật trạng thái cho thanh toán của người dùng.

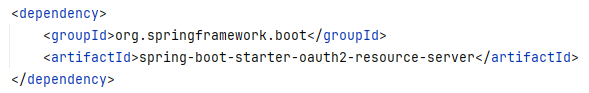
SC034 - SC035: Quản lý nhân viên của admin bao gồm thêm xóa sửa thông tin nhân viên.

SC064 – SC065: quản lý phản hồi của người dùng bao gồm xóa, cập nhật trạng thái và giao nhiệm vụ cho nhân viên để giải quyết vấn đề của người dùng.

## Khởi Tạo Backend với Spring Boot

Với các phiên bản như đã thống nhất ở mục 2, tôi tạo một project java theo đúng các phiên bản yêu cầu và bắt đầu code api theo các chức năng được yêu cầu.

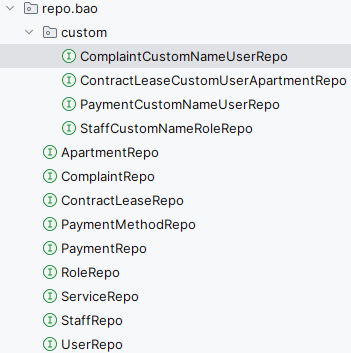
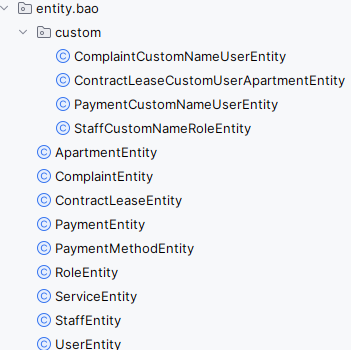
Cài đặc file bảo mật : với Spring Boot nếu muốn truy xuất api từ các thư viện React.js lúc đang ở local thì ta phải cài đặt cors, vì khi truy xuất api từ một localhost thì cơ chế bảo mật của Spring Boot sẽ nhận định là một truy xuất không an toàn. Ngoài ra còn phải tùy chỉnh lại bảo mật vì chức năng mà tôi code là một chức năng của admin tức là người dùng thông thường và nhân viên sẽ không có quyền truy cập, và Spring đã hỗ trợ tất cả các cơ chế phân quyền này trong thư viện oauth2 nên chỉ việc cài thư viện vào project và tùy chỉnh cài đặt.



Hình III‑42 cài đặt thư viện trong dapedency

Cơ chế của bảo mật Spring sau khi tùy chỉnh và cài đặt thì nó sẽ xem phần Header khi được Request xem có gửi một token qua không và kiểm tra token có đúng định dạng hay không và sau khi giải mã token nó kiểm tra xem phân quyền của token này là gì có được phép truy cập vào chức năng này hay không và ngăng chặn nếu không thỏa.

Sau khi cài đặt phân quyền và bảo mật thì tôi viết các file liên kết đến các cơ sở dữ liệu mà mình cần sử dụng. Sau đây là các file entity và reposity để kết nối với các bảng tương ứng trên cơ sở dữ liệu :



Hình III‑43 file entity và reposity của các bảng sẽ dùng trong dự án

## Tổng Các Api Và Chức Năng Với Mỗi Api

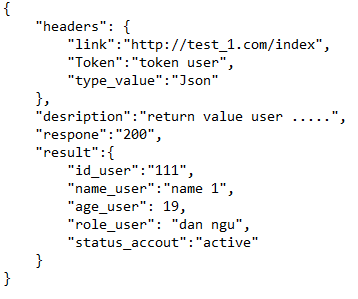
Ở phần này khi làm việc tới api thì đáng ra phải có api document, ban đầu lúc xác định kiến trúc dự án là api hay là web không api thì nhóm theo vote và lựa chọn không api tuy nhiên đến lúc làm thì lại bất ngờ đổi lại thành api nên không kịp thời gian và vốn nhóm đã chia ra mỗi nhóm nhỏ đều có cả Backend và Fontend nên chỉ quy định cấu trúc tổng quát mà một api sẽ phải có chứ không viết api document rõ ràng cho từng chức năng. Và trong khi code thì cấu trúc tổng quan của api thì là theo mẫu còn cấu trúc bên trong api thức là phần nội dung thông tin trả về thì tự Backend và Fontend quyết định.

Cấu trúc tổng quát của api phải bao gồm 3 thông tin sau :

Respone : trả về 200 nếu không lỗi và 400 nếu lỗi

Deription : thông báo sơ lược nội dung trả về

Result : nội dung trả về là một Object tùy vào chức năng



Hình III‑44 ví dụ cho một api trả về

Tổng cộng sẽ có 13 api bao gồm tất cả các chức năng trong giao diện yêu cầu, với phân công tôi sẽ tạo 3 service chính trong project tương đương với 3 chức năng chính là : quản lý phản hồi, quản lý nhân viên và quản lý thanh toán- người dùng.

Quản lý thanh toán – người dùng : bao gồm các chức năng tìm kiếm theo tên, số điện thoại, email, số phòng của căng hộ, chức năng sửa lại thông tin người dùng, sửa lại thanh toán của người dùng, xóa người dùng, xóa thanh toán của người dùng và tạo người dùng.

Quản lý phản hồi : các chức năng tìm kiếm phản hồi người dùng theo tên, căng hộ, sửa lại phản hồi hay lấy ra thông tin chi tiết của phản hồi.,

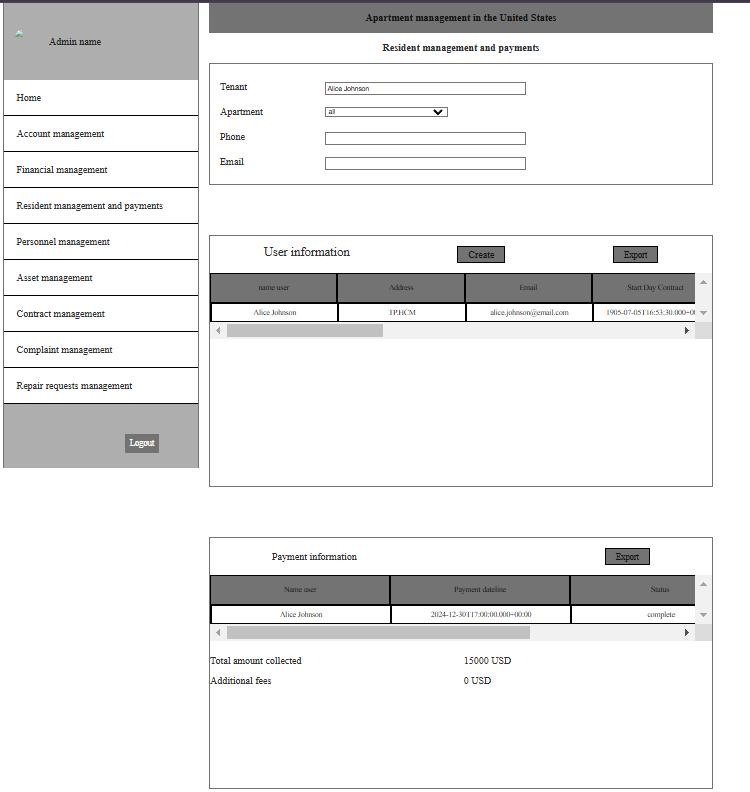
Quản lý nhân viên : với các chức năng tìm kiếm nhân viên theo tên, và trạng thái, sửa chửa thông tin của nhân viên, xóa nhân viên, thêm nhân viên.

## Khởi Tạo Fontend Với Vite React.js

Do lúc đầu không nằm trong đội Fontend của Reactjs nên tôi không biết về thông tin của các phiên bản và cách thức khởi tạo cũng như không có project root của đội Reactjs và lúc tôi làm về React.js thì đã gần cuối dự án nên không ai rảnh rỗi để giúp đỡ nên phải tự tạo một project ngoài thay vì viết trong file root của nhóm, việc này làm cho đến cuối phần code của tôi rất khó hợp lại với mọi người trong nhóm, và do mới học React.js không lâu nên cách tổ chức code cũng không tối ưu gây khó khăn cho việc hợp nhất code.

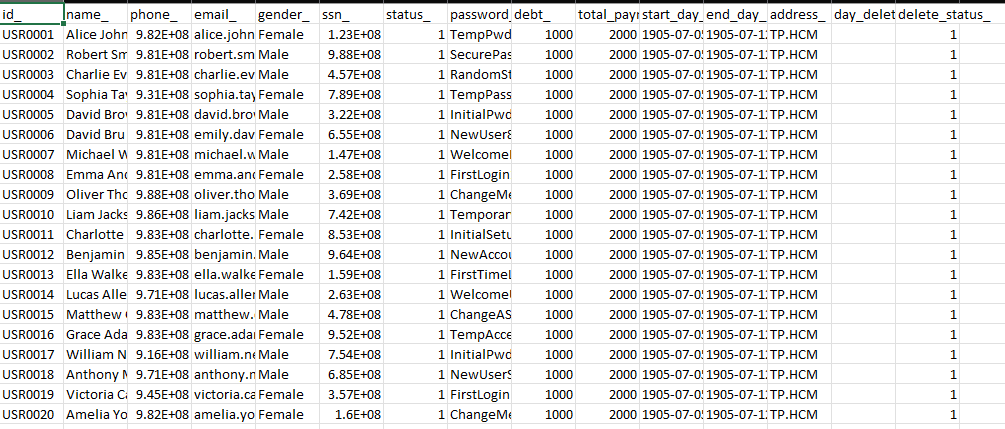
Tổng cộng tôi sẽ code 4 trang, thực tế thì sẽ hơn 4 trang tuy nhiên do có những trang có thể tóm gọn lại thành một podup nhỏ như cập nhật hay xóa, tạo tôi sẽ tổng hợp vào một trang cho gọn gàng hơn, trước khi quyết định tôi đã họp và có được sự đồng ý của nhóm Ba với nội dung tối ưu phần giao diện.

Trang đầu tiên là trang quản lý thanh toán và người dùng, có chức năng tìm kiếm theo tên, số điện thoại, email, số phòng, trang sẽ hiển thị thông tin của người dùng và các giao dịch của người dùng đó,



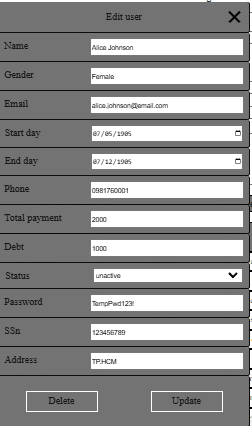
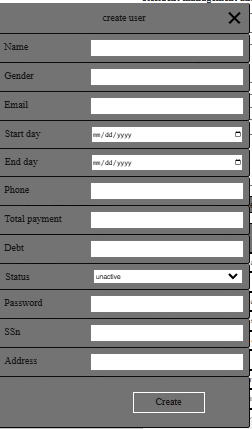
Hình III‑45 trang quản lý người dùng và thanh toán với chức năng tìm kiếm

Ngoài ra còn có chức năng xuất thông tin thành file excel :



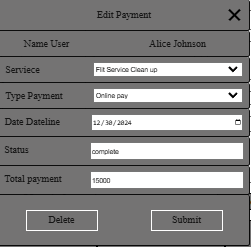
Hình III‑46 chức năng xuất thành file excel của trang quản lý người dùng và thanh toán

Và đương nhiên trang sẽ có tạo, xóa, sửa người dùng:



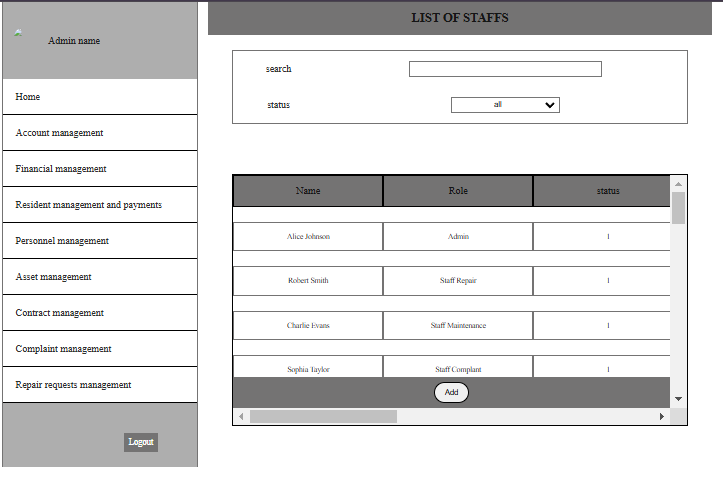
Hình III‑47 giao diện tạo, xóa, cập nhật thông tin người dùng

Sửa và xóa thông tin thanh toán :

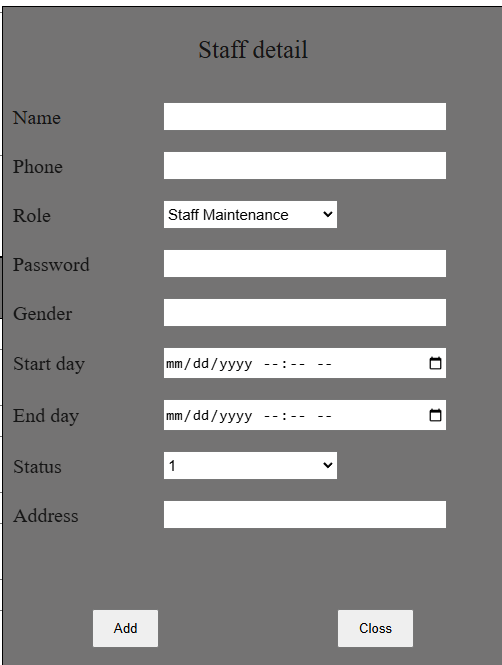
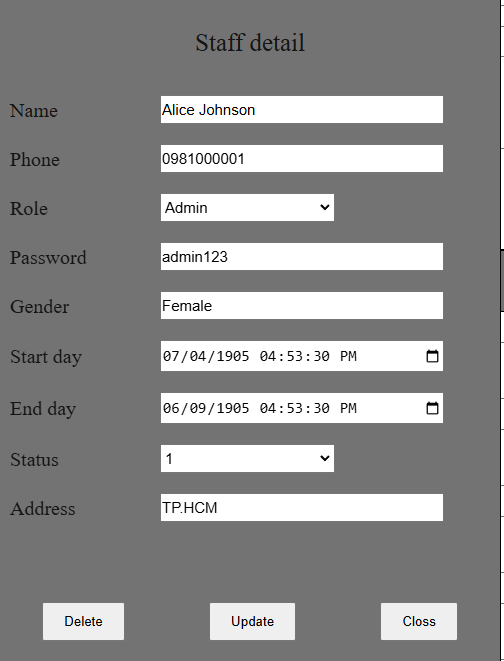


Hình III‑48 giao diện xóa, sửa thông tin thanh toán

Tiếp theo là trang quản lý nhân viên, trang này sẽ có các chức năng cơ bản như thêm xóa sửa các thông tin của nhân viên :

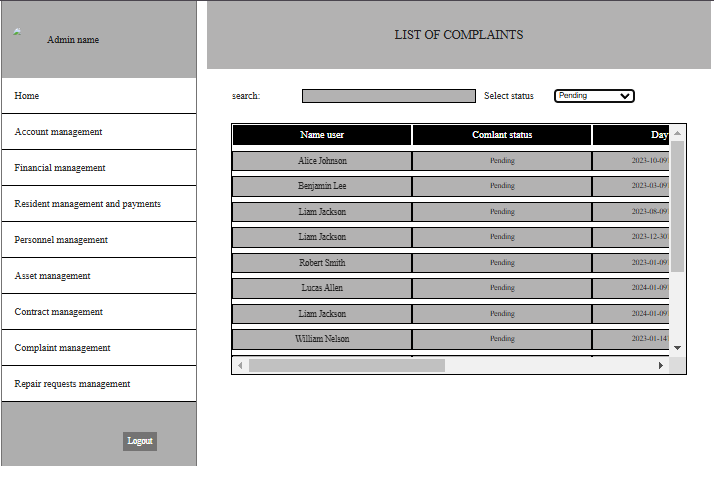


Hình III‑49 giao diện quản lý nhân viên với chức năng tìm kiếm

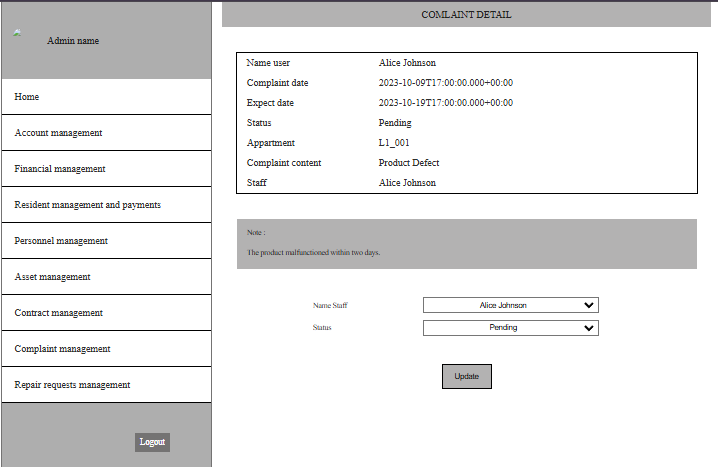
 

Hình III‑50 giao diện chức năng thêm, xóa, cập nhật nhân viên

Cuối cùng là giao diện quản lý phản hồi và trang chi tiết góp ý của người dùng, trang có các chức năng cơ bản như tìm kiếm, cập nhật trạng tháy tiến độ phản hồi người dùng.



Hình III‑51 trang quản lý phản hồi người dùng với chức năng tìm kiếm



Hình III‑52 trang chi tiết phản hồi với chức năng phân công nhân viên và cập nhật trạng thái phản hồi

## Thời Gian Hoàn Thành Và Kiểm Tra Lỗi

Theo như lịch trình thì tại ngày 19 các Dev sẽ ngưng code và làm việc khác tuy nhiên tôi phải làm cả React.js thêm nên phải lố sang tận ngày 21 thì mới hoàn thành tất cả các giao diện. và trong giai đoạn cuối các Tester nghỉ việc gần hết và chỉ còn lại một Tester khiến cho việc kiểm tra trở nên không thuận lợi, chỉ có một Tester nên không thể kiểm lỗi toàn bộ các trang được, nhận thấy vấn đề của nhóm nên tôi tiến hành tự kiểm tra lỗi bằng cách dùng thẳng các chức năng trên trang web, do tôi code cả Fontend và Backend nên việc này khá đơn giản, dựa theo file Testcase Tester đã viết trước đó toàn bộ các chức năng của phần tôi làm đều hoạt động tốt và không hề thiếu bất kì một chức năng hay bị một lỗi gì cả.

## Gộp Mã Nguồn Của Các Nhóm

Đây là phần mà nhóm đã làm không tốt, do các nhóm nhỏ bị trể tiến độ nên quá trình gộp code bị chậm đi rất nhiều, cuối cùng may mắn là các nhóm Backend đã hợp code thành công tuy nhiên các Fontend thì không gộp được code và thậm chí bỏ dỡ rất nhiều giao diện.

## Hoàn Thiện Dự Án Và Lưu Trữ

Đến ngày 22 dự án chính thức kết thúc, toàn bộ code và các thông tin như file User story, Testcase, Sprint Retro được đăng tải lên drive chung , github để lưu trữ và dự án chính thức đóng lại dù còn khá nhiều những vấn đề không hoàn thành.

# KẾT QUẢ VÀ ĐÁNH GIÁ

## Kết Quả Mong Muốn

Với nhóm là hoàn thành tối thiểu là phải xong 80% các chức năng cả Backend, Fontend và Tester. Với cá nhân thì hoàn thành 100% các chức năng của bản phân được phân công.

## Kết Quả Thực Tế

Với nhóm, chỉ hoàn thành gần như 100% phía Backend, tuy nhiên bên Fontend thì chưa tới 50%, kết quả khá tệ, với bản thân thì đã đạt được đúng kết quả mong muốn là 100% công việc, ngoài ra còn tham gia đóng góp làm cơ sở dữ liệu với nhóm trưởng và thiết kế giao diện với bên Ba.

## Đánh Giá Về Công Việc

Ở giai đoạn đầu thì việc không có chuẩn bị với phải vừa học vừa làm dẫn đến không kịp tiến độ và có phần chậm tiến độ, tuy nhiên sau khi quen dần với cách thức tổ chức công việc và học gần xong các thư viện thì dự án trở nên vô cùng đơn giản và dễ dàng, tuy nhiên việc làm việc nhóm khá khó chịu, nhất là về ý thức làm việc của các cá nhân do nhóm đông và hầu hết đều là sinh viên, và không quen biết nên lúc hợp không mấy sôi nổi chỉ có số ít tham gia bàn luận làm tiến độ làm việc bị chậm trễ

## Đánh Giá Về Nhóm

Các điểm sáng và tích cực :

Nhóm BA Phân tích nghiệp vụ đầy đủ, kĩ lưỡng và chặt chẽ. Thiết kế giao diện figma ban đầu rất tốt.

Nhóm Tester Viết Testcase và Restreport rất đầy đủ và chi tiết. Tuy không tets được hết màn hình nhưng thông báo lỗi chức năng của màn hình được test cho team DEV rất chi tiết.

Nhóm DEV Tuy có nhiều hạn chế nhưng cuối cùng đã hoàn thành được hầu hết các chức năng.

Các điểm tối, nên rút kinh nghiệm :

Nhóm trưởng : do ít kinh nghiệm nên việc phân công và bố trí công việc của bạn còn khá lóng ngóng và không đồng đều, các buổi họp cũng không thực sự hiệu quả chỉ có vài người tương tác không có bàn bạc gì nhiều.

Về các thành viên : do là đợt thực tập phải làm nhóm lớn nên việc kinh nghiệm và kiến thức không đồng điều chắc chắn sẽ xảy ra gây khó khăn trong việc phát triển dự án một cách nhanh chóng. Ngoài ra các thành viên khá thụ động trong việc đóng góp ý kiến, có vài thành viên out nhóm giữa chừng và tỏ ra không nghiêm túc nên dự án tiến triển khá chậm.

Vấn đề về BA – figma : các BA ban đầu cực kì tích cực trong việc làm giao diện tuy nhiên tới khoản giai đoạn cuối dự án thì các bạn dường như bị thụ động đi vì khi có dev đóng góp thì cũng không thấy phản hồi và sửa chữa lại giao diện và dev phải tự sửa lại giao diện.

Vấn đề về Tester : Do các bạn tester về cuối có out gần hết nên chỉ còn khoản một bạn test khiến cho việc test các chức năng và giao diện bị trì trệ và chỉ test chưa tới 50% tổng các chức năng và giao diện.

Vấn đề về Dev Backend và fontend : do ban đầu lúc quy định chuẩn về framework nhưng lại bỏ qua cách thức đặt tên class và cấu trúc bố trí của project nên việc gộp code khá mất thời gian. Ngoài ra do có bạn thì vừa học vừa làm nên có thể sẽ code không được chuẩn chỉ tuyên nhiên vấn đề này thực tế không ảnh hưởng nhiều nên có thể bỏ qua.

Trên hết là ý thức làm việc nhóm của các bạn không cao lắm vì đến cuối có vài bạn khi được phân công không nói gì đến cuối mới out dự án và chưa làm gì của phần mình hết, có những bạn làm không kịp cũng không thấy trao đổi đến cuối mới thông báo là không kịp nên thiếu khá nhiều chức năng. Vài bạn không update code lên github, nhiều bạn đến ngày ngưng code rồi vẫn không thấy code trên github khiến việc gộp code chậm và vô cùng mất thời gian

# KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

## Kết Luận

Tổng quan thì dự án còn chưa hoàn thành và thiếu xót rất nhiều. Tuy nhiên với cá nhân tôi thì học được rất nhiều thứ sau đợt thực tập như cách các ví trí làm việc với nhau, cách thức tổ chức một dự án như thế nào, các phần nào nên làm trước, lúc các Ba thiết kế và phân tích nghiệp vụ ta phải tham gia cố vấn để không bị sai lệch lúc thiết kế cơ sở dữ liệu, ngoài ra thì vấn đề xác định các kiến trúc, cách thức làm việc như quy định về đặc tên biến tên hàm tên class, tạo một project root thật sự rất quan trọng, ngoài ra còn sự quan trọng của buổi họp, đưa ra các quyết định phải khách quan tổng hợp từ nhiều cách nhìn sẽ ít bị lỗi, và tối ưu hơn.

## Các Phương Hướng Để Cải Thiện Dự Án

Nên đẩy nhanh tiến độ bắt đầu code dự án, phân tích kĩ hơn change requirement, Gộp code hiệu quả hơn, việc up code lên github cần các thành viên triển khai nhanh chóng hơn.

Cần chú tâm, chủ động hơn trong việc làm việc nhóm, có khó khăn hay gặp vấn đề gì thì cần nhắn vào nhóm để nhóm cùng bàn bạc.

Nên sôi nổi hơn trong cuộc họp, bàn bạc kĩ hơn trong họp, quy định những bố trí source code, quy ước đặt tên, tạo một project khuôn mẫu kỹ hơn cho mỗi một ngôn ngữ lập trình. Nếu có vấn đề gì trong lúc code phải phản ánh ngay, để nhóm cùng xử lý

Vì nhóm mình triển khai code chậm tiến độ hơn các nhóm khác do việc bầu chọn còn thiếu quyết đoán, chậm chạp hơn nhóm khác. Bên cạnh đó việc phân công cho các thành viên tự chọn là không phù hợp vì không thực sự hiệu quả và lề mề, nên để nhóm trưởng và nhóm phó trực tiếp chỉ định phân công luôn để không mất thời gian trong việc phân công.