**ĐẠI HỌC DUY TÂN**

**TRƯỜNG KHMT**

**KHOA CÔNG NGHỆ THÔNG TIN**

**--------------🙖🙐✰🙖🙐-------------**

****

**Tên đề tài:**

**XÂY DỰNG VAXBOT – WEBSITE ĐẶT LỊCH**

**TIÊM CHỦNG TỰ ĐỘNG NHẮC NHỞ VÀ**

**CHATBOX AI TƯ VẤN**

**(PRODUCT BACKLOG)**

**GVHD:** Th.S Hồ Lê Viết Nin

**Thành Viên**

Nguyễn Thị Lưu Ly - 27201240742

Đoàn Hải Trân - 27207940630

Lê Đại Ánh - 27211242562

Cao Thanh Đức - 27211200374

Nguyễn Đức Ninh - 27211501635

**Đà Nẵng, Tháng 03 năm 2025**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **THÔNG TIN DỰ ÁN** | | | | |
| **Dự án viết tắt** | **VAXBOT** | | | |
| **Tên dự án** | Xây dựng **VaxBot** – Website đặt lịch tiêm chủng tự động nhắc nhở và chatbot AI tư vấn. | | | |
| **Thời gian bắt đầu** | 17/03/2025 | **Thời gian kết thúc** | 17/5/2025 | |
| **Lead Institution** | Khoa Công Nghệ Thông Tin, Đại Học Duy Tân | | | |
| **Giáo viên hướng dẫn** | Hồ Lê Viết Nin  Email: hlvnin88@gmail.com  Phone: 0905455246 | | | |
| **Chủ sở hữu dự án & Chi tiết liên hệ** | Hồ Lê Viết Nin  Email: hlvnin88@gmail.com  Phone: 0905455246 | | | |
| **Đối tác** | Duy Tan University | | | |
| **Quản lý dự án & Scrum Master** | Nguyễn Thị Lưu Ly | lykiri0505@gmail.com | | 0977585728 |
| **Thành viên nhóm** | Đoàn Hải Trân | doanhaitran64@gmail.com | | 0934998645 |
| Cao Thanh Đức | caothanhduc.bdqb@gmail.com | | 0374018578 |
| Lê Đại Ánh | ledaianh172003@gmail.com | | 0335728612 |
| Nguyễn Đức Ninh | nguyenducninh210303@gmail.com | | 0799830773 |

|  |  |  |  |
| --- | --- | --- | --- |
| **TÊN TÀI LIỆU** | | | |
| **Tiêu đề tài liệu** | Product Backlog | | |
| **Tác giả** | Nguyễn Thị Lưu Ly | | |
| **Chức năng** | Scrum master | | |
| **Ngày** | 26/03/2025 | **Tên tệp:** | 01\_[KLTN - 52] Product Backlog |
| **URL** |  | | |
| **Access** | Khoa CNTT | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **LỊCH SỬ BẢN SỬA** | | | |
| **Version** | **Person** | **Date** | **Description** |
| 1.0 | Nguyễn Thị Lưu Ly | 25/03/2025 | Tạo tài liệu |
| 1.1 | Nguyễn Thị Lưu Ly | 26/03/2025 | Chỉnh sửa tài liệu |

|  |  |  |  |
| --- | --- | --- | --- |
| **PHÊ DUYỆT TÀI LIỆU** | | | |
| **Người hướng dẫn** | | Hồ Lê Viết Nin | **Chữ ký:** |  |
| **Ngày:** | 26/03/2025 |
| **Chủ sở hữu** | | Hồ Lê Viết Nin | **Chữ ký:** |  |
| **Ngày:** | 26/03/2025 |
| **Scrum master** | | Nguyễn Thị Lưu Ly | **Chữ ký:** |  |
| **Ngày:** | 26/03/2025 |
| **Thành viên** | | Đoàn Hải Trân | **Chữ ký:** |  |
| **Ngày:** | 26/03/2025 |
| Cao Thanh Đức | **Chữ ký:** |  |
| **Ngày:** | 26/03/2025 |
| Lê Đại Ánh | **Chữ ký:** |  |
| **Ngày:** | 26/03/2025 |
| Nguyễn Đức Ninh | **Chữ ký:** |  |
| **Ngày:** | 26/03/2025 |

**MỤC LỤC**

[1. GIỚI THIỆU 1](#_Toc193786079)

[1.1. Mục đích 1](#_Toc193786080)

[1.2. Phạm vi 1](#_Toc193786081)

[1.3. Tham khảo 1](#_Toc193786082)

[1.4. Biểu đồ UseCase tổng quát 2](#_Toc193786083)

[2. PRODUCT BACKLOG 2](#_Toc193786084)

[2.1. Danh sách các Product Backlog trong Hệ thống 2](#_Toc193786085)

[2.1.1. PB01 – Đăng ký 6](#_Toc193786086)

[2.1.2. PB02 – Đăng nhập 7](#_Toc193786087)

[2.1.3. PB03 – Đăng xuất 8](#_Toc193786088)

[2.1.4. PB04 – Quên mật khẩu 8](#_Toc193786089)

[2.1.5. PB05 – Tìm kiếm 9](#_Toc193786090)

[2.1.6. PB06 – Quản lý thông tin cá nhân 10](#_Toc193786091)

[2.1.7. PB07 – Xem thông tin trung tâm 11](#_Toc193786092)

[2.1.8. PB08 – Xem thông tin dịch vụ 12](#_Toc193786093)

[2.1.9. PB09 – Xem thông tin vaccine 12](#_Toc193786094)

[2.1.10. PB10 – Đặt vaccine và hẹn lịch tiêm 13](#_Toc193786095)

[2.1.11. PB11 – Thanh toán online 14](#_Toc193786096)

[2.1.12. PB12 – Liên hệ tư vấn chatbot AI 15](#_Toc193786097)

[2.1.13. PB13 – Quản lý người dùng 16](#_Toc193786098)

[2.1.14. PB14 – Quản lý loại vaccine 18](#_Toc193786099)

[2.1.15. PB15 – Quản lý vaccine 20](#_Toc193786100)

[2.1.16. PB16 – Quản lý danh sách lịch hẹn 21](#_Toc193786101)

[2.1.17. PB17 – Quản lý lịch sử tiêm chủng 23](#_Toc193786102)

[2.1.18. PB18 – Quản lý đơn đặt hàng 24](#_Toc193786103)

[2.1.19. PB19 – Quản lý thanh toán 25](#_Toc193786104)

[2.1.20. PB20 – Quản lý nhà cung cấp 26](#_Toc193786105)

[2.1.21. PB21 – Quản lý nhà sản xuất 27](#_Toc193786106)

[2.1.22. PB22 – Tư vấn khách hàng 29](#_Toc193786107)

[2.1.23. PB23 – Nhắc hẹn tự động 30](#_Toc193786108)

[2.1.24. PB24 – Báo cáo thống kê 30](#_Toc193786109)

[2.1.25. PB25 - Đọc blog tin tức: 31](#_Toc193786110)

[2.1.26. PB26 - Quản lý blog 32](#_Toc193786111)

[2.2. Phân tích về Sprint Backlog 33](#_Toc193786112)

[3. HẠN CHẾ 34](#_Toc193786113)

[4. TÓM LƯỢC 34](#_Toc193786114)

1. **GIỚI THIỆU**
   1. **Mục đích**

Mục đích của tài liệu này:

* Dựa vào tài liệu User Story, nêu rõ các chức năng mà người dùng yêu cầu sẽ thực hiện như thế nào, sử dụng công nghệ gì, độ ưu tiên ra làm sao,…
* Đưa ra các đề xuất dự án, kế hoạch hành động dự án, kiến trúc, giải pháp thực hiện, bao gồm cả về quy hoạch, phát triển, thực hiện và giám sát các dự án.
* Ngoài ra, xác định nhu cầu kinh doanh, các vấn đề hoặc tình huống liên quan đến các dự án khởi tạo, xây dựng và các rủi ro khi triển khai dự án.
  1. **Phạm vi**

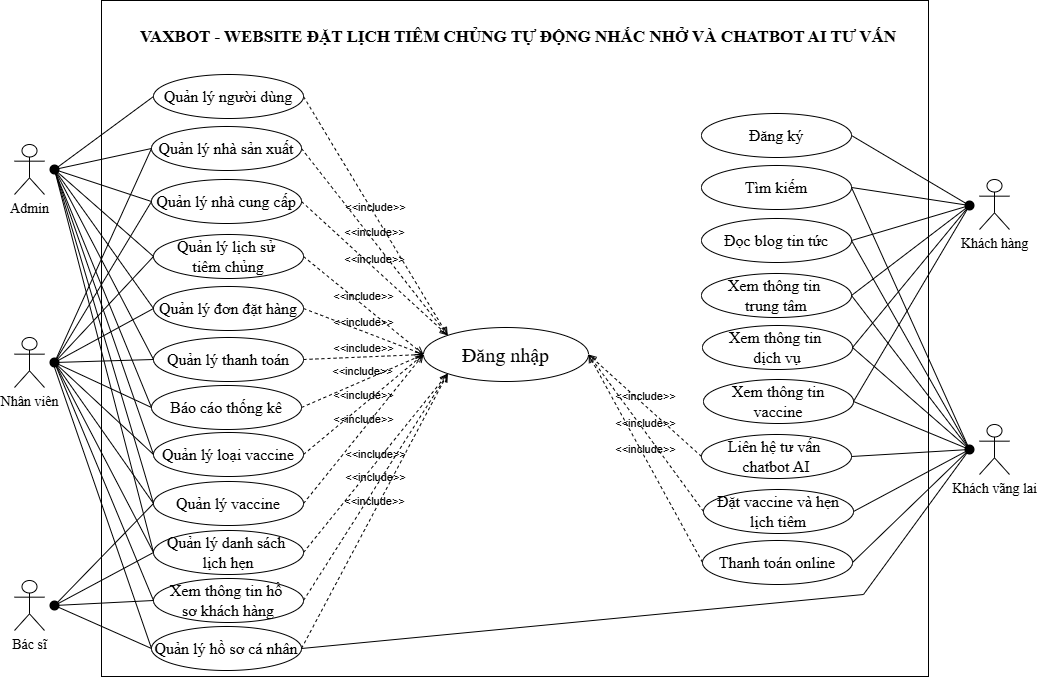
Cung cấp cho những lập trình viên:

* Một cái nhìn tổng quan về hệ thống và cái nhìn chi tiết các yêu cầu
* Các bước thực hiện của từng chức năng.
* Tùy vào độ ưu tiên của các chức năng để xác định thứ tự thực hiện.
  1. **Tham khảo**

*Bảng 1.1. Danh mục tài liệu tham khảo*

|  |  |  |
| --- | --- | --- |
| No | Document Information | References |
| 1 | Scrum Process | <https://www.mountaingoatsoftware.com/agile/scrum>  <https://www.scrum.org/resources/what-scrum-module> |
| 3 | Tool and Technical | <https://www.postgresql.org/docs/> |
| <https://www.react.dev/> |
| <https://www.typescriptlang.org/docs/> |
| <https://docs.nestjs.com/> |
| <https://tailwindcss.com/docs/space> |
| 4 | References | User Story Document  Sample Product Backlog Document |

* 1. **Biểu đồ UseCase tổng quát**



*Hình 1.1: Biểu đồ Use Case tổng quát*

1. **PRODUCT BACKLOG**
   1. **Danh sách các Product Backlog trong Hệ thống**

*Bảng 2.1: Đặc tả tổng quát các Product Backlog*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **ID** | **Chủ đề** | **Tác nhân** | **Mong muốn** | **Nên** | **Độ ưu tiên** |
| PB01 | Đăng nhập | Khách Hàng/ Admin/Nhân Viên/ Bác sĩ | Đăng nhập vào hệ thống | Người dùng có thể đăng nhập vào hệ thống | 2 |
| PB02 | Đăng ký | Khách vãng lai | Thêm tài khoản khoản vào hệ thống | Khách hàng có thể đăng kí tài khoản | 2 |
| PB03 | Đăng xuất | Khách hàng/ Admin/ Nhân viên/ Bác sĩ | Đăng xuất khỏi hệ thống | Người dùng có thể đăng xuất khỏi phiên truy cập của mình. | 2 |
| PB04 | Quên mật khẩu | Khách hàng/ Admin/ Nhân viên/ Bác sĩ | Lấy lại mật khẩu để truy cập vào hệ thống | Người dùng có thể lấy lại mật khẩu cho tài khoản của mình | 3 |
| PB05 | Tìm kiếm | Khách hàng/ Khách vãng lai | Tìm kiếm thông tin | Người dùng có thể tìm kiếm thông tin về vaccine, các bài viết về sức khỏe, y tế,… | 2 |
| PB06 | Quản lý thông tin cá nhân | Khách hàng/ Admin/ Nhân viên/ Bác sĩ | Quản lý các thông tin cá nhân của mình | Người dùng có thể xem/cập nhật thông tin cá nhân của mình | 3 |
| PB07 | Xem thông tin trung tâm | Khách hàng/ Khách vãng lai | Xem thông tin của trunng tâm | Khách hàng có thể xem các thông tin liên quan, công khai của trung tâm. | 1 |
| PB08 | Xem thông tin dịch vụ | Khách hàng/ Khách vãng lai | Xem thông tin các dịch vụ của trung tâm | Khách hàng có thể xem các dịch vụ hiện có của trung tâm | 1 |
| PB09 | Xem thông tin vaccine | Khách hàng/ Khách vãng lai | Xem thông tin của từng vaccine | Khách hàng có thể xem bảng giá, các thông tin liên quan của vaccine đó | 1 |
| PB10 | Đặt vaccine và hẹn lịch tiêm | Khách hàng | Đặt và hẹn trước lịch tiêm các mũi vaccine | Khách hàng có thể đặt trước mũi vaccine và hẹn lịch ngày tiêm. | 1 |
| PB11 | Thanh toán online | Khách hàng | Thanh toán online bằng ví điện tử | Khách hàng có thể thực hiện thanh toán online bằng ví momo. | 3 |
| PB12 | Liên hệ tư vấn chatbot AI | Khách hàng/ Khách vãng lai | Liên hệ nhắn tin tư vấn với chatbot AI | Khách hàng có thể liên hệ chatbot AI để nhận được sự tư vấn nhanh chóng, tự động. | 1 |
| PB13 | Quản lý người dùng | Admin | Quản lý các account người dùng hiện có trong hệ thống. | Admin có thể xem/sửa/xóa/phân quyền với từng vai trò/chức vụ người dùng. | 2 |
| PB14 | Quản lý loại vaccine | Admin/Nhân viên | Quản lý các loại vaccine (phòng bệnh gì) | Người dùng có thể thêm/sửa/xóa các loại vaccine (phòng bệnh gì) | 1 |
| PB15 | Quản lý vaccine | Admin/Nhân viên | Quản lý chi tiết từng mũi vaccine | Người dùng có thể xem/thêm/cập nhật/ xóa thông tin từng mũi vaccine. | 1 |
| PB16 | Quản lý danh sách lịch hẹn | Admin/Nhân viên | Quản lý danh sách lịch hẹn đã được đặt trước | Người dùng có thể theo dõi, quản lý danh sách các lịch hẹn mà khách hàng đặt trước. | 1 |
| PB17 | Quản lý lịch sử tiêm chủng | Admin/Nhân viên | Quản lý lịch sử các mũi tiêm của trung tâm | Người dùng có thể theo dõi, quản lý lịch sử các mũi vaccine được tiêm cho khách hàng ở trung tâm trong thời gian chỉ định. | 2 |
| PB18 | Quản lý đơn đặt hàng | Admin/Nhân viên | Quản lý các đơn đặt hàng (vaccine) của khách hàng | Người dùng có thể theo dõi, thao tác đối với các đơn đặt trước của khách hàng. | 1 |
| PB19 | Quản lý thanh toán | Admin/Nhân viên | Quản lý các hóa đơn thanh toán của khách hàng | Người dùng có thể theo dõi/quản lý các hóa đơn thanh toán của khách hàng. | 1 |
| PB20 | Quản lý nhà cung cấp | Admin/Nhân viên | Quản lý thông tin nhà cung cấp vaccine | Người dùng có thể xem/thêm/cập nhật/xóa thông tin nhà cung cấp vaccine | 1 |
| PB21 | Quản lý nhà sản xuất | Admin/Nhân viên | Quản lý thông tin nhà sản xuất vaccine | Người dùng có thể xem/thêm/cập nhật/xóa thông tin nhà sản xuất vaccine | 1 |
| PB22 | Tư vấn khách hàng | Admin/Nhân viên | Chat tư vấn với khách hàng | Nhân viên chat tư vấn với khách hàng khi AI không hỗ trợ được. | 3 |
| PB23 | Nhắc hẹn tự động |  | Nhắc hẹn tự động qua email | Hệ thống sẽ tự động gửi mail nhắc nhở khách hàng khi sắp đến lịch tiêm. | 1 |
| PB24 | Báo cáo thống kê | Admin/Nhân viên | Thống kê, báo cáo kinh doanh của trung tâm | Người dùng có thể theo dõi, quản lý, xuất báo cáo doanh thu. | 3 |
| PB25 | Đọc blog tin tức | Khách hàng/ Khách vãng lai | Đọc các bài đăng tin tức. | Người dùng có thể đọc các bài blog tin tức trên hệ thống. | 2 |
| PB26 | Quản lý blog | Admin/Nhân viên | Quản lý các bài đăng tin tức | Người dùng có thể theo dõi và quản lý các bài blog tin tức trên hệ thống | 2 |

*Bảng 2.2: Bảng độ ưu tiên*

|  |  |  |
| --- | --- | --- |
| **Ký hiệu** | **Độ ưu tiên** | **Giải thích** |
| 1 | Cao | Cần thiết để sản phẩm hoạt động. |
| 2 | Trung bình | Quan trọng nhưng có thể làm sau khi hoàn thành các chức năng cần thiết. |
| 3 | Thấp | Có thể làm sau khi hoàn thành các chức năng cần thiết và quan trọng. |

* + 1. ***PB01 – Đăng ký***

|  |  |
| --- | --- |
| **ID** | **PB01** |
| **Tiêu đề** | Đăng ký |
| **Mô tả** | Khách vãng lai có thể đăng ký tài khoản truy cập Website |
| **Các bước thực hiện** | 1. Thiết kế giao diện đăng ký tài khoản.  2. Xử lý logic kiểm tra email đăng ký và gọi API đăng ký tài khoản.  3. Kiểm duyệt thông tin vừa đăng ký.  4. Hiển thị thống báo và chuyển sang giao diện đăng nhập nếu thành công hoặc thông báo nếu đăng ký thất bại. |
| **Điều kiện trước** | 1. Khách vãng lai vào giao diện trang chủ, nhấn chọn nút “Đăng ký”.  2. Khi giao diện đăng ký đổ ra, tiếp tục nhập các thông tin yêu cầu vào form đăng ký.  3. Sau đó, nhấn nút “Tạo tài khoản” để gửi yêu cầu đăng ký tài khoản mới.  4. Hệ thống tiến hành kiểm tra email, mật khẩu.  - Nếu email và mật khẩu hợp lệ, hệ thống hiển thị thông báo “Đăng ký thành công” và sẽ chuyển tới giao diện đăng nhập.  - Nếu không hợp lệ, hệ thống sẽ hiển thị thông báo “Email đã tồn tại” hoặc “Mật khẩu chưa đúng định dạng”. |
| **Điều kiện sau** | Giao diện được tải thành công và khách vãng lai có thể thấy các nút tùy chọn chức năng. |
| **Công nghệ sử dụng** | - Frontend: React.js (giao diện đăng ký, gọi API)  - Backend: Node.js (xử lý lưu trữ cơ sở dữ liệu)  - Database: PostgreSQL(lưu trữ thông tin tài khoản)  - API: Nestjs và Swagger |
| **Độ phức tạp** | Cao |

* + 1. ***PB02 – Đăng nhập***

|  |  |
| --- | --- |
| **ID** | **PB02** |
| **Tiêu đề** | Đăng nhập |
| **Mô tả** | Khách hàng có thể đăng nhập để truy cập Website |
| **Các bước thực hiện** | 1. Thiết kế giao diện đăng nhập.  2. Xử lý logic kiểm tra thông tin đăng nhập và gọi API đăng nhập.  3. Đối chiếu thông tin đăng nhập.  4. Hiển thị thông báo và chuyển sang giao diện trang chủ nếu thành công và thông báo nếu thất bại. |
| **Điều kiện trước** | 1. Khách hàng vào trang chủ và nhấn chọn nút “Đăng nhập”.  2. Giao diện đăng nhập đổ ra, nhập các thông tin đăng nhập.  3. Nhấn chọn nút “Đăng nhập”.  4. Hệ thống tiến hành kiểm tra, đối chiếu thông tin tài khoản.  - Nếu hợp lệ, hệ thống hiển thị thông báo đăng nhập thành công và chuyển sang giao diện trang chủ.  - Nếu không hợp lệ, hệ thống hiển thị thông báo “Sai tên đăng nhập hoặc mật khẩu” hoặc “Tài khoản không tồn tại”. |
| **Điều kiện sau** | Tài khoản đã được đăng ký thành công. |
| **Công nghệ sử dụng** | - Frontend: React.js (giao diện nhập, gọi API)  - Backend: Node.js (xử lý đối chiếu cơ sở dữ liệu)  - Database: PostgreSQL(lưu trữ thông tin tài khoản)  - API: Nestjs và Swagger |
| **Độ phức tạp** | Trung bình |

* + 1. ***PB03 – Đăng xuất***

|  |  |
| --- | --- |
| **ID** | **PB03** |
| **Tiêu đề** | Đăng xuất |
| **Mô tả** | Khách hàng có thể đăng xuất khỏi phiên đăng nhập của mình. |
| **Các bước thực hiện** | 1. Thiết kế nút đăng xuất.  2. Hệ thống ghi nhận và thoát khỏi phiên đăng nhập |
| **Điều kiện trước** | 1. Người dùng đang ở trong phiên đăng nhập, và nhấn chọn nút “Đăng xuất”.  2. Hệ thống xử lý yêu cầu đăng xuất và thoát khỏi phiên đăng nhập và chuyển về trang chủ. |
| **Điều kiện sau** | Đã đăng nhập vào hệ thống |
| **Công nghệ sử dụng** | - Frontend: React.js  - Backend: Node.js |
| **Độ phức tạp** | Thấp |

* + 1. ***PB04 – Quên mật khẩu***

|  |  |
| --- | --- |
| **ID** | **PB04** |
| **Tiêu đề** | Quên mật khẩu |
| **Mô tả** | Người dùng quên mật khẩu và muốn lấy lại hoặc thay đổi mật khẩu của mình. |
| **Các bước thực hiện** | 1. Thiết kế giao diện quên mật khẩu.  2. Xử lý logic thay đổi mật khẩu và gọi API.  3. Hệ thống ghi nhận yêu cầu và kiểm tra.  4. Hiển thị thông báo và chuyển sang giao diện đăng nhập nếu thành công hoặc thông báo nếu thất bại. |
| **Điều kiện trước** | 1. Ở giao diện đăng nhập, người dùng nhán chọn “Quên mật khẩu”.  2. Giao diện mật khẩu hiện ra, người dùng nhập email và xác nhận email.  3. Sau khi hệ thông nhận được xác nhận, người dùng có thể nhập mật khẩu mới và nhấn chọn “Đổi mật khẩu” để gửi yêu cầu tới hệ thống.  4. Hệ thống ghi nhận và tiến hành xử lý.  - Nếu hợp lệ, hiển thị thông báo “Đã đổi mật khâu thành công”.  - Nếu không hợp lệ, hiển thị thông báo “Mật khẩu chưa đúng định dạng” hoặc “Xác nhận mật khẩu chưa đúng”. |
| **Điều kiện sau** | Người dùng xác thực email thành công |
| **Công nghệ sử dụng** | - Frontend: React.js (giao diện đổi mật khẩu, gọi API)  - Backend: Node.js (xử lý thay đổi mật khẩu)  - Database: PostgreSQL (lưu trữ dữ liệu tài khoản)  - API: Nestjs và Swagger |
| **Độ phức tạp** | Cao |

* + 1. ***PB05 – Tìm kiếm***

|  |  |
| --- | --- |
| **ID** | **PB05** |
| **Tiêu đề** | Tìm kiếm |
| **Mô tả** | Người dùng có thể tìm kiếm các thông tin, tin tức trên website |
| **Các bước thực hiện** | 1. Thiết kế giao diện ô tìm kiếm.  2. Xử lý logic kiểm tra từ khóa và gọi API tìm kiếm.  3. Lọc kết quả tìm kiếm theo từ khóa.  4. Hiển thị kết quả tìm kiếm hoặc thông báo nếu không có kết quả phù hợp. |
| **Điều kiện trước** | 1. Người dùng vào website và nhập từ khóa vào ô tìm kiếm.  2. Người dùng nhấn nút “Tìm kiếm” hoặc “Enter” để gửi yêu cầu đến hệ thống.  3. Hệ thống ghi nhận và kiểm tra từ khóa:  - Nếu có từ khóa trong thông tin, tin tức hệ thống hiển thị thông tin liên quan phù hợp.  - Nếu không có thông tin khớp, hệ thống hiển thị thông báo “Thông tin tìm kiếm không tồn tại”. |
| **Điều kiện sau** | Trang chủ đã được tải thành công và người dùng có thể thấy ô tìm kiếm. |
| **Công nghệ sử dụng** | - Frontend: React.js (giao diện tìm kiếm, gọi API)  - Backend: Node.js (xử lý tìm kiếm, truy vấn cơ sở dữ liệu)  - Database: PostgreSQL (lưu trữ thông tin)  - API: Nestjs và Swagger |
| **Độ phức tạp** | Cao |

* + 1. ***PB06 – Quản lý thông tin cá nhân***

|  |  |
| --- | --- |
| **ID** | **PB06** |
| **Tiêu đề** | Quản lý thông tin cá nhân |
| **Mô tả** | Người dùng có thể xem/cập nhật thông tin cá nhân của mình. |
| **Các bước thực hiện** | 1. Thiết kế giao diện thông tin cá nhân.  2. Xử lý logic và gọi API cập nhật thông tin cá nhân,  3. Hệ thống cập nhật thông tin của tài khoản.  4. Cập nhật thông tin và hiển thị thông báo nếu thành công hoặc không cập nhật và hiển thi thông báo nếu thất bại. |
| **Điều kiện trước** | 1. Người dùng vào trang “Thông tin cá nhân”.  2. Người dùng có thể thực hiện chỉnh sửa các thông tin sau:  - Họ tên\*  - Email\*  - Ngày sinh  - Giới tính  - Số điện thoại\*  - Địa chỉ  - Ảnh đại diện  - Tiểu sử bênh án  3. Người dùng nhấn chọn “Lưu” để gửi yêu cầu đến hệ thống.  4. Hệ thống kiểm tra dữ liệu và xử lý:  - Nếu hợp lệ, hiển thị thông báo “Cập nhật thành công” và hiển thị dữ liệu đã được cập nhật.  - Nếu không hợp lệ, hiển thị thông báo “Dữ liệu không hợp lệ. Vui lòng kiểm tra lại”. |
| **Điều kiện sau** | Trang thông tin cá nhân được tải thành công và người dùng có thể thực hiện cập nhật. |
| **Công nghệ sử dụng** | - Frontend: React.js (giao diện cập nhật và gọi API)  - Backend: Node.js (xử lý cập nhật dữ liệu)  - Database: PostgreSQL (lưu trữ dữ liệu)  - API: Nestjs và Swagger |
| **Độ phức tạp** | Trung bình |

* + 1. ***PB07 – Xem thông tin trung tâm***

|  |  |
| --- | --- |
| **ID** | **PB07** |
| **Tiêu đề** | Xem thông tin trung tâm |
| **Mô tả** | Khách hàng có thể xem các thông tin về trung tâm được đăng tải công khai trên website. |
| **Các bước thực hiện** | 1. Thiết kế giao diện thông tin trung tâm.  2. Hiển thị dữ liệu thông tin của trung tâm |
| **Điều kiện trước** | 1. Khách hàng truy cập vào trang chủ và nhấn chọn danh mục thông tin.  2. Hệ thống chuyển sang trang thông tin về trung tâm |
| **Điều kiện sau** | Không có |
| **Công nghệ sử dụng** | - Frontend: React.js (giao diện thông tin) |
| **Độ phức tạp** | Thấp |

* + 1. ***PB08 – Xem thông tin dịch vụ***

|  |  |
| --- | --- |
| **ID** | **PB08** |
| **Tiêu đề** | Xem thông tin dịch vụ |
| **Mô tả** | Khách hàng có thể xem thông tin các dịch vụ hiện có của trung tâm. |
| **Các bước thực hiện** | 1. Thiết kế giao diện thông tin dịch vụ.  2. Hiển thị dữ liệu thông tin dịch vụ mà trung tâm đang có. |
| **Điều kiện trước** | 1. Khách hàng truy cập vào trang chủ và nhấn chọn danh mục dịch vụ  2. Hệ thống chuyển sang trang thông tin về dịch vụ. |
| **Điều kiện sau** | Không có |
| **Công nghệ sử dụng** | - Frontend: React.js (giao diện dịch vụ) |
| **Độ phức tạp** | Thấp |

* + 1. ***PB09 – Xem thông tin vaccine***

|  |  |
| --- | --- |
| **ID** | **PB09** |
| **Tiêu đề** | Xem thông tin vaccine |
| **Mô tả** | Khách hàng có thể xem thông tin về loại vaccine đang được trung tâm lưu hành. |
| **Các bước thực hiện** | 1. Thiết kế giao diện Vaccine  2. Hiển thị thông tin các loại vaccine trung tâm đang có. |
| **Điều kiện trước** | 1. Người dùng truy cập vào trang chủ và nhấn chọn mục vaccine  2. Hệ thống chuyển sang trang thông tin về vaccine.  3, Người dùng có thể xem danh sách các loại vaccine hoặc nhấn chọn vào xem chi tiết các vaccine đó |
| **Điều kiện sau** | Không có |
| **Công nghệ sử dụng** | - Frontend: React.js (giao diện vaccine)  - Database: PostgreSQL (lưu trữ thông tin vaccine) |
| **Độ phức tạp** | Trung bình |

* + 1. ***PB10 – Đặt vaccine và hẹn lịch tiêm***

|  |  |
| --- | --- |
| **ID** | **PB10** |
| **Tiêu đề** | Đặt vaccine và hẹn lịch tiêm |
| **Mô tả** | Khách hàng có thể đặt trước loại vaccine không có sẵn và hẹn trước lịch tiêm phòng. |
| **Các bước thực hiện** | 1. Thiết kế giao diện đặt mua vaccine.  2. Xử lý logic kiểm tra thông tin đơn hàng và gọi API đặt lịch tiêm chủng.  3. Xử lý đơn đặt hàng theo yêu cầu.  4. Thêm đơn đặt hàng vào lịch hẹn và hiển thị thông báo nếu thành công hoặc hiển thị thông báo nếu thất bại. |
| **Điều kiện trước** | 1. Khách hàng vào trang đặt vaccine.  2. Khách hàng tiến hành quy trình đặt vaccine:  2.1. Chọn loại vaccine muốn đặt mua. Nhấn chọn “Đăng ký mũi tiêm”.  2.2. Điền các thông tin cá nhân người tiêm bao gồm:  - Họ tên\*  - Số điện thoại\*  - Ngày sinh\*  - Giới tính\*  - Email\*  - Địa chỉ\*  - Tiểu sử bênh án (nếu có)  - Chọn ngày muốn tiêm\*  2.3. Nhấn chọn “Thanh toán”.  2.4. Khách hàng xác nhận lại thông tin và tiến hành thanh toán (qua momo)  2.5. Chờ xác nhận và liên hệ từ trung tâm.  3. Hệ thống sẽ ghi nhận và xử lý đơn đặt vaccine:  - Nếu thành công, hệ thống hiển thị thông báo “Đăng ký mũi tiêm thành công” và thêm vào lịch hẹn tiêm.  - Nếu thất bại, hiển thị thông báo “Giao dịch thất bại, hãy liên hệ chúng tôi qua tin nhắn hoặc số hotline để được hướng dẫn”. |
| **Điều kiện sau** | Giao diện đặt vaccine được tải thành công và khách hàng có thể thấy thủ tục đặt vaccine. |
| **Công nghệ sử dụng** | - Frontend: React.js (giao diện đặt mua vaccine và gọi API)  - Backend: Node.js (xử lý đơn đặt hàng)  - Database: PostgreSQL (lưu trữ giữ liệu đơn hàng)  - API: Nestjs và Swagger |
| **Độ phức tạp** | Cao |

* + 1. ***PB11 – Thanh toán online***

|  |  |
| --- | --- |
| **ID** | **PB11** |
| **Tiêu đề** | Thanh toán online |
| **Mô tả** | Khách hàng có thể thanh toán online khi đặt mua vaccine |
| **Các bước thực hiện** | 1. Thiết kế giao diện thanh toán.  2. Xử lý kiểm tra tổng giá trị đơn hàng và gọi API thanh toán momo.  3. Hiển thị giao diện thanh toán momo.  4. Hiển thị hóa đơn thanh toán nếu thành công và thông báo nếu thất bại |
| **Điều kiện trước** | 1. Khách hàng sau khi điền thông tin tiêm chủng và nhấn thanh toán, sẽ chuyển sang trang thanh toán ví momo.  2. Khách hàng thực hiện thanh toán đơn hàng qua ví momo.  3. Hệ thống xử lý yêu cầu thanh toán:  - Hiển thị hóa đơn điện tử khi thanh toán thành công.  - Hiển thị thông báo “Giao dịch thất bại” khi thanh toán không thành công. |
| **Điều kiện sau** | Giao diện thanh toán momo được tải thành công. |
| **Công nghệ sử dụng** | - Frontend: React.js  - Backend: Node.js  - Database: PostgreSQL  - API: Nestjs và Swagger |
| **Độ phức tạp** | Cao |

* + 1. ***PB12 – Liên hệ tư vấn chatbot AI***

|  |  |
| --- | --- |
| **ID** | **PB12** |
| **Tiêu đề** | Liên hệ tư vấn chatbot AI |
| **Mô tả** | Khách hàng có thể liên tư vấn các dịch vụ, thông tin về tiêm chủng thông qua chatbot AI. |
| **Các bước thực hiện** | 1. Thiết kế giao diện chatbot AI  2. Phân tích và trả lời câu hỏi từ khách hàng.  3. Hiển thị câu trả lời, thông tin về câu hỏi hoặc lời xin lỗi khi không tìm thấy thông tin. |
| **Điều kiện trước** | 1. Khách hàng truy cập vào chatbot tư vấn.  2. Khách hàng tiến hành đặt câu hỏi và nhấn nút gửi hoặc enter để gửi câu hỏi đến AI.  3. Gemini AI sẽ tiến hành phân tích câu hỏi và gửi câu trả lời đến khách hàng.  Nếu không trả lời được câu hỏi của khách hàng, AI sẽ gửi lời xin lỗi “Xin lỗi, hiện chúng tôi không thể trả lời câu hỏi này của quý khách!”. |
| **Điều kiện sau** | Giao diện chatbot AI được tải thành công và người dùng có thể nhập liệu câu hỏi. |
| **Công nghệ sử dụng** | - Frontend: React.js  - Backend: Node.js  - Database: PostgreSQL  - API: Nestjs và Swagger |
| **Độ phức tạp** | Cao |

* + 1. ***PB13 – Quản lý người dùng***

|  |  |
| --- | --- |
| **ID** | **PB13** |
| **Tiêu đề** | Quản lý người dùng |
| **Mô tả** | Admin, Nhân viên có thể quản lý các thông tin của người dùng có tài khoản truy cập hệ thống. |
| **Các bước thực hiện** | 1. Thiết kế giao diện quản lý người dùng.  2. Xử lý logic và gọi các API quản lý người dùng.  3. Hiển thị dữ liệu người dùng và các tùy chọn chức năng thực hiện ở màn hình này. |
| **Điều kiện trước** | 1. Admin vào trang quản lý người dùng.  2. Người dùng sẽ thấy và theo dõi được danh sách người dùng hiện có trên hệ thống.  3. Admin có thể thực hiện các chức năng thêm/cập nhật/xóa/ phân quyền người dùng.  3.1. [Thêm]  - Admin nhấn chọn nút “Tạo mới”.  - Điền các thông tin về người dùng: tên\*, email\*, địa chỉ, số điện thoại\*, mật khẩu\*, chức vụ, ngày sinh, giới tính, ảnh đại diện.  - Nhấn chọn “Lưu” để gửi yêu cầu tới hệ thống. Hệ thống sẽ kiểm tra và hiển thị thông báo “Tạo mới thành công” và thêm thông tin người dùng vào danh sách. Hoặc hiển thị thông báo “Lỗi! Tạo mới thất bại” nếu thông tin không hợp lệ.  - Nhấn chọn “Thoát” nếu muốn thoát khỏi tác vụ đang thực hiện.  3.2. [Cập nhật]  - Admin nhấn chọn nút “Cập nhật”.  - Cập nhật các thông tin cần thiết.  - Nhấn chọn “Lưu” để gửi yêu cầu tới hệ thống. Hệ thống sẽ kiểm tra và hiển thị thông báo “Cập nhật thành công” và cập nhật lại thông tin người dùng ở danh sách. Hoặc hiển thị thông báo “Lỗi! Cập nhật thất bại” nếu thông tin không hợp lệ.  - Nhấn chọn “Thoát” nếu muốn thoát khỏi tác vụ đang thực hiện.  3.3. [Xóa]  - Admin nhấn chọn nút “Xóa”.  - Admin xác nhận xóa để gửi yêu cầu lên hệ thống. Hệ thống xóa người dùng khỏi danh sách và hiển thị thông báo “Xóa thành công” hoặc thông báo “Lỗi! Xóa thất bại”.  - Nhấn chọn “Thoát” nếu muốn thoát khỏi tác vụ đang thực hiện.  3.4. [Phân quyền]  - Admin nhấn chọn chức năng “Phân quyền”.  - Admin thực hiện cấp các quyền cho người dùng đó.  - Nhấn “Lưu” để gửi yêu cầu đến hệ thống. Hệ thống xử lý và hiển thị thông báo “Cập nhật thành công” hoặc “Lỗi, cập nhật thất bại”.  - Nhấn chọn “Thoát” nếu muốn thoát khỏi tác vụ đang thực hiện. |
| **Điều kiện sau** | Giao diện quản lý người dùng tải thành công. |
| **Công nghệ sử dụng** | - Frontend: React.js  - Backend: Node.js  - Database: PostgreSQL  - API: Nestjs và Swagger |
| **Độ phức tạp** | Cao |

* + 1. ***PB14 – Quản lý loại vaccine***

|  |  |
| --- | --- |
| **ID** | **PB14** |
| **Tiêu đề** | Quản lý loại vaccine |
| **Mô tả** | Người dùng có thể quản lý các loại vaccine (chữa bênh gì) |
| **Các bước thực hiện** | 1. Thiết kế giao diện quản lý loại vaccine  2. Xử lý logic và gọi các API quản lý loại vaccine  3. Hiển thị dữ liệu các loại vaccine và các tùy chọn chức năng thực hiện ở màn hình này. |
| **Điều kiện trước** | 1. Người dùng vào trang quản lý loại vaccine.  2. Người dùng sẽ thấy và theo dõi được danh sách các loại vaccine hiện có.  3. Người dùng có thể thực hiện các chức năng thêm/cập nhật/xóa các loại vaccine.  3.1. [Thêm]  - Người dùng nhấn chọn nút “Tạo mới”.  - Điền các thông tin về loại: Tên phòng bệnh\*, thông tin về bệnh, ghi chú  - Nhấn chọn “Lưu” để gửi yêu cầu tới hệ thống. Hệ thống sẽ kiểm tra và hiển thị thông báo “Tạo mới thành công” và thêm thông tin loại vaccine vào danh sách. Hoặc hiển thị thông báo “Lỗi! Tạo mới thất bại” nếu thông tin không hợp lệ.  - Nhấn chọn “Thoát” nếu muốn thoát khỏi tác vụ đang thực hiện.  3.2. [Cập nhật]  -Người dùng nhấn chọn nút “Cập nhật”.  - Cập nhật các thông tin cần thiết.  - Nhấn chọn “Lưu” để gửi yêu cầu tới hệ thống. Hệ thống sẽ kiểm tra và hiển thị thông báo “Cập nhật thành công” và cập nhật lại thông tin loại vaccine ở danh sách. Hoặc hiển thị thông báo “Lỗi! Cập nhật thất bại” nếu thông tin không hợp lệ.  - Nhấn chọn “Thoát” nếu muốn thoát khỏi tác vụ đang thực hiện.  3.3. [Xóa]  - Người dùng nhấn chọn nút “Xóa”.  - Người dùng xác nhận xóa để gửi yêu cầu lên hệ thống. Hệ thống xóa loại vaccine đó khỏi danh sách và hiển thị thông báo “Xóa thành công” hoặc thông báo “Lỗi! Xóa thất bại”.  - Nhấn chọn “Thoát” nếu muốn thoát khỏi tác vụ đang thực hiện. |
| **Điều kiện sau** | Giao diện quản lý loại vaccine được tải thành công |
| **Công nghệ sử dụng** | - Frontend: React.js  - Backend: Node.js  - Database: PostgreSQL  - API: Nestjs và Swagger |
| **Độ phức tạp** | Cao |

* + 1. ***PB15 – Quản lý vaccine***

|  |  |
| --- | --- |
| **ID** | **PB15** |
| **Tiêu đề** | Quản lý vaccine |
| **Mô tả** | Người dùng có thể quản lý chi tiết các vaccine hiện được sử dụng ở trung tâm. |
| **Các bước thực hiện** | 1. Thiết kế giao diện quản lý vaccine  2. Xử lý logic và gọi các API quản lý vaccine  3. Hiển thị dữ liệu các vaccine và các tùy chọn chức năng thực hiện ở màn hình này. |
| **Điều kiện trước** | 1. Người dùng vào trang quản lý vaccine.  2. Người dùng sẽ thấy và theo dõi được danh sách các vaccine hiện có.  3. Người dùng có thể thực hiện các chức năng thêm/cập nhật/xóa các vaccine.  3.1. [Thêm]  - Người dùng nhấn chọn nút “Tạo mới”.  - Điền các thông tin về loại: Tên vaccine\*, mô tả, giá\*, quốc gia sản xuất\*, Thuộc loại vaccine\*, số lượng\*, ngày hết hạn\*, tình trạng\*.  - Nhấn chọn “Lưu” để gửi yêu cầu tới hệ thống. Hệ thống sẽ kiểm tra và hiển thị thông báo “Tạo mới thành công” và thêm thông tin vaccine mới vào danh sách. Hoặc hiển thị thông báo “Lỗi! Tạo mới thất bại” nếu thông tin không hợp lệ.  - Nhấn chọn “Thoát” nếu muốn thoát khỏi tác vụ đang thực hiện.  3.2. [Cập nhật]  -Người dùng nhấn chọn nút “Cập nhật”.  - Cập nhật các thông tin cần thiết.  - Nhấn chọn “Lưu” để gửi yêu cầu tới hệ thống. Hệ thống sẽ kiểm tra và hiển thị thông báo “Cập nhật thành công” và cập nhật lại thông tin vaccine đó ở danh sách. Hoặc hiển thị thông báo “Lỗi! Cập nhật thất bại” nếu thông tin không hợp lệ.  - Nhấn chọn “Thoát” nếu muốn thoát khỏi tác vụ đang thực hiện.  3.3. [Xóa]  - Người dùng nhấn chọn nút “Xóa”.  - Người dùng xác nhận xóa để gửi yêu cầu lên hệ thống. Hệ thống xóa vaccine đó khỏi danh sách và hiển thị thông báo “Xóa thành công” hoặc thông báo “Lỗi! Xóa thất bại”.  - Nhấn chọn “Thoát” nếu muốn thoát khỏi tác vụ đang thực hiện. |
| **Điều kiện sau** | Giao diện quản lý vaccine được tải thành công. |
| **Công nghệ sử dụng** | - Frontend: React.js  - Backend: Node.js  - Database: PostgreSQL  - API: Nestjs và Swagger |
| **Độ phức tạp** | Cao |

* + 1. ***PB16 – Quản lý danh sách lịch hẹn***

|  |  |
| --- | --- |
| **ID** | **PB16** |
| **Tiêu đề** | Quản lý danh sách lịch hẹn |
| **Mô tả** | Người dùng có thể quản lý lịch hẹn tiêm phòng của khách hàng |
| **Các bước thực hiện** | 1. Thiết kế giao diện quản lý danh sách lịch hẹn.  2. Xử lý logic và gọi các API danh sách lịch hẹn.  3. Hiển thị dữ liệu các danh sách lịch hẹn và các tùy chọn chức năng thực hiện ở màn hình này. |
| **Điều kiện trước** | 1. Người dùng vào trang quản lý danh sách lịch hẹn.  2. Người dùng sẽ thấy và theo dõi được danh sách lịch hẹn tiêm chủng của khách hàng.  3. Người dùng có thể thực hiện các chức năng thêm/cập nhật/xóa danh sách lịch hẹn.  3.1. [Thêm]  - Người dùng nhấn chọn nút “Tạo mới”.  - Điền các thông tin về loại: Họ tên, số điện thoại, liều vaccine, trạng thái, ngày hẹn, ghi chú.  - Nhấn chọn “Lưu” để gửi yêu cầu tới hệ thống. Hệ thống sẽ kiểm tra và hiển thị thông báo “Tạo mới thành công” và thêm lịch hẹn mới vào danh sách. Hoặc hiển thị thông báo “Lỗi! Tạo mới thất bại” nếu thông tin không hợp lệ.  - Nhấn chọn “Thoát” nếu muốn thoát khỏi tác vụ đang thực hiện.  3.2. [Cập nhật]  -Người dùng nhấn chọn nút “Cập nhật”.  - Cập nhật các thông tin cần thiết.  - Nhấn chọn “Lưu” để gửi yêu cầu tới hệ thống. Hệ thống sẽ kiểm tra và hiển thị thông báo “Cập nhật thành công” và cập nhật lại lịch hẹn của khách hàng ở danh sách. Hoặc hiển thị thông báo “Lỗi! Cập nhật thất bại” nếu thông tin không hợp lệ.  - Nhấn chọn “Thoát” nếu muốn thoát khỏi tác vụ đang thực hiện.  3.3. [Xóa]  - Người dùng nhấn chọn nút “Xóa”.  - Người dùng xác nhận xóa để gửi yêu cầu lên hệ thống. Hệ thống xóa lịch hẹn đó khỏi danh sách và hiển thị thông báo “Xóa thành công” hoặc thông báo “Lỗi! Xóa thất bại”.  - Nhấn chọn “Thoát” nếu muốn thoát khỏi tác vụ đang thực hiện. |
| **Điều kiện sau** | Giao diện quản lý danh sách lịch hẹn được tải thành công. |
| **Công nghệ sử dụng** | - Frontend: React.js  - Backend: Node.js  - Database: PostgreSQL  - API: Nestjs và Swagger |
| **Độ phức tạp** | Cao |

* + 1. ***PB17 – Quản lý lịch sử tiêm chủng***

|  |  |
| --- | --- |
| **ID** | **PB17** |
| **Tiêu đề** | Quản lý lịch sử tiêm chủng |
| **Mô tả** | Người dùng có thể theo dõi và quản lý lịch sử tiêm chủng của trung tâm. |
| **Các bước thực hiện** | 1. Thiết kế giao diện quản lý lịch sử tiêm chủng.  2. Xử lý logic và gọi API lấy danh sách giao dịch.  3. Hiển thị dữ liệu danh sách các mũi tiêm thành công và xử lý thực hiện các tùy chọn chức năng ở màn hình này. |
| **Điều kiện trước** | 1. Người dùng vào trang quản lý lịch sử tiêm chủng.  2. Người dùng sẽ thấy và theo dõi được danh sách các mũi tiêm đã hoàn thành.  3. Người dùng có thể thực hiện các tác vụ:  - In hóa đơn.  - Tải xuống file pdf hóa đơn.  - Xuất file excel danh sách lịch sử tiêm chủng. |
| **Điều kiện sau** | Giao diện quản lý danh sách tiêm chủng tải thành công và hiện ra đầy dủ danh sách và các tác vụ. |
| **Công nghệ sử dụng** | - Frontend: React.js  - Backend: Node.js  - Database: PostgreSQL  - API: Nestjs và Swagger |
| **Độ phức tạp** | Trung bình. |

* + 1. ***PB18 – Quản lý đơn đặt hàng***

|  |  |
| --- | --- |
| **ID** | **PB18** |
| **Tiêu đề** | Quản lý đơn đặt hàng |
| **Mô tả** | Người dùng có thể quản lý các đơn đặt mua vaccine. |
| **Các bước thực hiện** | 1. Thiết kế giao diện quản lý đơn đặt hàng.  2. Xử lý logic và gọi các API danh sách đơn đặt hàng.  3. Hiển thị dữ liệu các danh sách đơn đặt hàng và các tùy chọn chức năng thực hiện ở màn hình này. |
| **Điều kiện trước** | 1. Người dùng vào trang quản lý đơn đặt hàng.  2. Người dùng sẽ thấy và theo dõi được danh sách đơn đặt mua vaccine của khách hàng.  3. Người dùng có thể thực hiện các chức năng thêm/cập nhật/xóa danh sách lịch hẹn.  3.1. [Thêm]  - Người dùng nhấn chọn nút “Tạo mới”.  - Điền các thông tin về loại: Họ tên, số điện thoại, email, liều vaccine, trạng thái, ngày hẹn, ghi chú.  - Nhấn chọn “Lưu” để gửi yêu cầu tới hệ thống. Hệ thống sẽ kiểm tra và hiển thị thông báo “Tạo mới thành công” và thêm đơn đặt hàng mới vào danh sách. Hoặc hiển thị thông báo “Lỗi! Tạo mới thất bại” nếu thông tin không hợp lệ.  - Nhấn chọn “Thoát” nếu muốn thoát khỏi tác vụ đang thực hiện.  3.2. [Cập nhật]  - Người dùng nhấn chọn nút “Cập nhật”.  - Cập nhật các thông tin cần thiết.  - Nhấn chọn “Lưu” để gửi yêu cầu tới hệ thống. Hệ thống sẽ kiểm tra và hiển thị thông báo “Cập nhật thành công” và cập nhật lại đơn hàng của khách hàng ở danh sách. Hoặc hiển thị thông báo “Lỗi! Cập nhật thất bại” nếu thông tin không hợp lệ.  - Nhấn chọn “Thoát” nếu muốn thoát khỏi tác vụ đang thực hiện.  3.3. [Xóa]  - Người dùng nhấn chọn nút “Xóa”.  - Người dùng xác nhận xóa để gửi yêu cầu lên hệ thống. Hệ thống xóa đơn hàng đó khỏi danh sách và hiển thị thông báo “Xóa thành công” hoặc thông báo “Lỗi! Xóa thất bại”.  - Nhấn chọn “Thoát” nếu muốn thoát khỏi tác vụ đang thực hiện. |
| **Điều kiện sau** | Giao diện quản lý đơn đặt hàng được tải thành công. |
| **Công nghệ sử dụng** | - Frontend: React.js  - Backend: Node.js  - Database: PostgreSQL  - API: Nestjs và Swagger |
| **Độ phức tạp** | Cao |

* + 1. ***PB19 – Quản lý thanh toán***

|  |  |
| --- | --- |
| **ID** | **PB19** |
| **Tiêu đề** | Quản lý thanh toán |
| **Mô tả** | Người dùng có thể theo dõi, quàn lý hóa đơn thanh toán. |
| **Các bước thực hiện** | 1. Thiết kế giao diện quản lý thanh toán.  2. Xử lý logic và gọi các API thanh toán momo.  3. Hiển thị dữ liệu các danh sách thanh toán và các tùy chọn chức năng thực hiện ở màn hình này. |
| **Điều kiện trước** | 1. Người dùng vào trang quản lý thanh toán.  2. Người dùng sẽ thấy và theo dõi được danh sách hóa đơn thanh toán của khách hàng.  3. Người dùng có thể thực hiện các chức năng:  - In hóa đơn.  - Tải hóa đơn  - Xuất file excel danh sách hóa đơn thanh toán. |
| **Điều kiện sau** | Giao diện quản lý thanh toán được tải thành công. |
| **Công nghệ sử dụng** | - Frontend: React.js  - Backend: Node.js  - Database: PostgreSQL  - API: Nestjs và Swagger |
| **Độ phức tạp** | Cao |

* + 1. ***PB20 – Quản lý nhà cung cấp***

|  |  |
| --- | --- |
| **ID** | **PB20** |
| **Tiêu đề** | Quản lý nhà cung cấp |
| **Mô tả** | Admin, nhân viên có thể quản lý thông tin nhà cung cấp. |
| **Các bước thực hiện** | 1. Thiết kế giao diện quản lý nhà cung cấp.  2. Xử lý logic và gọi các API nhà cung cấp.  3. Hiển thị dữ liệu nhà cung cấp và các tùy chọn chức năng thực hiện ở màn hình này. |
| **Điều kiện trước** | 1. Người dùng vào trang quản lý nhà cung cấp.  2. Người dùng sẽ thấy và theo dõi được danh sách nhà cung cấp vaccine hiện có.  3. Người dùng có thể thực hiện các chức năng thêm/cập nhật/xóa nhà cung cấp.  3.1. [Thêm]  - Người dùng nhấn chọn nút “Tạo mới”.  - Điền các thông tin về nhà cung cấp: tên, địa chỉ, email, số điện thoại, ghi chú,…  - Nhấn chọn “Lưu” để gửi yêu cầu tới hệ thống. Hệ thống sẽ kiểm tra và hiển thị thông báo “Tạo mới thành công” và thêm thông tin nhà cung cấp vào danh sách. Hoặc hiển thị thông báo “Lỗi! Tạo mới thất bại” nếu thông tin không hợp lệ.  - Nhấn chọn “Thoát” nếu muốn thoát khỏi tác vụ đang thực hiện.  3.2. [Cập nhật]  -Người dùng nhấn chọn nút “Cập nhật”.  - Cập nhật các thông tin cần thiết.  - Nhấn chọn “Lưu” để gửi yêu cầu tới hệ thống. Hệ thống sẽ kiểm tra và hiển thị thông báo “Cập nhật thành công” và cập nhật lại thông tin nhà cung cấp ở danh sách. Hoặc hiển thị thông báo “Lỗi! Cập nhật thất bại” nếu thông tin không hợp lệ.  - Nhấn chọn “Thoát” nếu muốn thoát khỏi tác vụ đang thực hiện.  3.3. [Xóa]  - Người dùng nhấn chọn nút “Xóa”.  - Người dùng xác nhận xóa để gửi yêu cầu lên hệ thống. Hệ thống xóa nhà cung cấp khỏi danh sách và hiển thị thông báo “Xóa thành công” hoặc thông báo “Lỗi! Xóa thất bại”.  - Nhấn chọn “Thoát” nếu muốn thoát khỏi tác vụ đang thực hiện. |
| **Điều kiện sau** | Giao diện quản lý nhà cung cấp được tải thành công |
| **Công nghệ sử dụng** | - Frontend: React.js  - Backend: Node.js  - Database: PostgreSQL  - API: Nestjs và Swagger |
| **Độ phức tạp** | Cao |

* + 1. ***PB21 – Quản lý nhà sản xuất***

|  |  |
| --- | --- |
| **ID** | **PB21** |
| **Tiêu đề** | Quản lý nhà sản xuất |
| **Mô tả** | Admin, nhân viên có thể quản lý thông tin nhà sản xuất |
| **Các bước thực hiện** | 1. Thiết kế giao diện quản lý nhà sản xuất.  2. Xử lý logic và gọi các API nhà sản xuất  3. Hiển thị dữ liệu nhà sản xuất và các tùy chọn chức năng thực hiện ở màn hình này. |
| **Điều kiện trước** | 1. Người dùng vào trang quản lý nhà sản xuất.  2. Người dùng sẽ thấy và theo dõi được danh sách nhà sản xuất vaccine hiện có.  3. Người dùng có thể thực hiện các chức năng thêm/cập nhật/xóa nhà sản xuất.  3.1. [Thêm]  - Người dùng nhấn chọn nút “Tạo mới”.  - Điền các thông tin về nhà sản xuất: tên, quốc gia,…  - Nhấn chọn “Lưu” để gửi yêu cầu tới hệ thống. Hệ thống sẽ kiểm tra và hiển thị thông báo “Tạo mới thành công” và thêm thông tin nhà sản xuất vào danh sách. Hoặc hiển thị thông báo “Lỗi! Tạo mới thất bại” nếu thông tin không hợp lệ.  - Nhấn chọn “Thoát” nếu muốn thoát khỏi tác vụ đang thực hiện.  3.2. [Cập nhật]  -Người dùng nhấn chọn nút “Cập nhật”.  - Cập nhật các thông tin cần thiết.  - Nhấn chọn “Lưu” để gửi yêu cầu tới hệ thống. Hệ thống sẽ kiểm tra và hiển thị thông báo “Cập nhật thành công” và cập nhật lại thông tin nhà sản xuất ở danh sách. Hoặc hiển thị thông báo “Lỗi! Cập nhật thất bại” nếu thông tin không hợp lệ.  - Nhấn chọn “Thoát” nếu muốn thoát khỏi tác vụ đang thực hiện.  3.3. [Xóa]  - Người dùng nhấn chọn nút “Xóa”.  - Người dùng xác nhận xóa để gửi yêu cầu lên hệ thống. Hệ thống xóa nhà sản xuất khỏi danh sách và hiển thị thông báo “Xóa thành công” hoặc thông báo “Lỗi! Xóa thất bại”.  - Nhấn chọn “Thoát” nếu muốn thoát khỏi tác vụ đang thực hiện. |
| **Điều kiện sau** | Trang quản lý nhà sản xuất được tải thành công |
| **Công nghệ sử dụng** | - Frontend: React.js  - Backend: Node.js  - Database: PostgreSQL  - API: Nestjs và Swagger |
| **Độ phức tạp** | Cao |

* + 1. ***PB22 – Tư vấn khách hàng***

|  |  |
| --- | --- |
| **ID** | **PB22** |
| **Tiêu đề** | Tư vấn khách hàng |
| **Mô tả** | Nhân viên có thể tư vấn khách hàng khi chatbot AI không thể hỗ trợ. |
| **Các bước thực hiện** | 1. Thiết kế giao diện chatbot.  2. Nhận câu hỏi từ phía khách hàng và thông báo đến nhân viên  3. Gửi câu trả lời của nhân viên tới khách hàng. |
| **Điều kiện trước** | 1. Khách hàng truy cập vào chatbot tư vấn.  2. Khách hàng tiến hành đặt câu hỏi và nhấn nút gửi hoặc enter để gửi câu hỏi đến nhân viên.  3. Nhân viên sau khi nhận được thông báo có tin nhắn, sẽ tiến hành trả lời hỗ trợ khách hàng. |
| **Điều kiện sau** | Giao diện chatbot được tải thành công và người dùng có thể thực hiện nhắn tin. |
| **Công nghệ sử dụng** | - Frontend: React.js  - Backend: Node.js  - Database: PostgreSQL  - API: Nestjs và Swagger |
| **Độ phức tạp** | Cao |

* + 1. ***PB23 – Nhắc hẹn tự động***

|  |  |
| --- | --- |
| **ID** | **PB23** |
| **Tiêu đề** | Nhắc hẹn tự động |
| **Mô tả** | Khách hàng đã có lịch hẹn sẽ nhận được lời nhắc hẹn qua email khi sắp đến ngày hẹn tiêm chủng. |
| **Các bước thực hiện** | 1. Xử lý lời nhắc hẹn tự động.  2. Xử lý gửi lời nhắc hẹn tự động đến email khách hàng có lịch hẹn tiêm chung. |
| **Điều kiện trước** | Khách hàng phải có lịch hẹn tiêm chủng. |
| **Điều kiện sau** | Không có |
| **Công nghệ sử dụng** | - Frontend: React.js  - Backend: Node.js  - Database: PostgreSQL  - API: Nestjs và Swagger |
| **Độ phức tạp** | Cao |

* + 1. ***PB24 – Báo cáo thống kê***

|  |  |
| --- | --- |
| **ID** | **PB24** |
| **Tiêu đề** | Báo cáo thống kê |
| **Mô tả** | Admin, Nhân viên có thể theo dõi doanh thu của trung tâm. |
| **Các bước thực hiện** | 1. Thiết kế giao diện theo dõi doanh thu.  2. Xử lý logic và đổ dữ liệu về báo cáo.  3. Hiển thị số liệu báo cáo về doanh thu, số mũi vaccine tiêm ra, số khách hàng đặt lịch hẹn,…  4. Xuất được dữ liệu báo cáo ra file Excel. |
| **Điều kiện trước** | 1. Người dùng truy cập vào trang báo cáo tổng quan hệ thống.  2. Người dùng có thể theo dõi số liệu sau:  - Doanh thu theo tháng.  - Số mũi vaccine được tiêm ra theo tháng.  - Số khách hàng đặt lịch tiêm.  - Biểu đồ cột về số mũi vaccine tiêm ra trong năm.  - Bảng xếp hạng vaccine đang được tiêm nhiều nhất.  3. Người chọn “Xuất Excel” để xuất báo cáo ra file Excel.  4. Hệ thống xử lý yêu cầu và thực hiện yêu cầu xuất báo cáo. |
| **Điều kiện sau** | Giao diện tổng quan tải thành công và người dùng thấy được các số liệu hiển thị trên giao diện. |
| **Công nghệ sử dụng** | - Frontend: React.js  - Backend: Node.js  - Database: PostgreSQL  - API: Nestjs và Swagger |
| **Độ phức tạp** | Cao |

* + 1. ***PB25 - Đọc blog tin tức:***

|  |  |
| --- | --- |
| **ID** | **PB25** |
| **Tiêu đề** | Đọc blog tin tức |
| **Mô tả** | Khách hàng có thể đọc các blog tin tức về vaccine, y tế |
| **Các bước thực hiện** | 1. Thiết kế giao diện đọc blog tin tức.  2. Xử lý logic và lấy dữ liệu hiển thị.  3. Hiển thị dữ liệu ra màn hình.1 |
| **Điều kiện trước** | 1. Khách hàng truy cập vào mục “Tin tức”.  2. Sau khi giao diện hiển thị danh sách các bài viết, tin tức. Người dùng có thể click chọn vào tiêu đề để đọc. |
| **Điều kiện sau** | Giao diện blog tin tức được tải thành công |
| **Công nghệ sử dụng** | - Frontend: React.js  - Backend: Node.js  - Database: PostgreSQL  - API: Nestjs và Swagger |
| **Độ phức tạp** | Cao |

* + 1. ***PB26 - Quản lý blog***

|  |  |
| --- | --- |
| **ID** | **PB26** |
| **Tiêu đề** | Quản lý loại blog |
| **Mô tả** | Người dùng có thể quản lý các blog tin tức. |
| **Các bước thực hiện** | 1. Thiết kế giao diện quản lý blog  2. Xử lý logic và gọi các API quản lý blog  3. Hiển thị dữ liệu các bài viết và các tùy chọn chức năng thực hiện ở màn hình này. |
| **Điều kiện trước** | 1. Người dùng vào trang quản lý blog.  2. Người dùng sẽ thấy và theo dõi được danh sách các bài viết đã được tạo.  3. Người dùng có thể thực hiện các chức năng thêm/cập nhật/xóa các bài viết.  3.1. [Thêm]  - Người dùng nhấn chọn nút “Tạo mới”.  - Sau đó tiến hành soạn thảo bài viết  - Nhấn chọn “Lưu” để gửi yêu cầu tới hệ thống. Hệ thống sẽ kiểm tra và hiển thị thông báo “Tạo mới thành công” và thêm bài viết vào danh sách. Hoặc hiển thị thông báo “Lỗi! Tạo mới thất bại” nếu thông tin không hợp lệ.  - Nhấn chọn “Thoát” nếu muốn thoát khỏi tác vụ đang thực hiện.  3.2. [Cập nhật]  -Người dùng nhấn chọn nút “Cập nhật”.  - Cập nhật các thông tin cần thiết.  - Nhấn chọn “Lưu” để gửi yêu cầu tới hệ thống. Hệ thống sẽ kiểm tra và hiển thị thông báo “Cập nhật thành công” và cập nhật lại bài viết ở danh sách. Hoặc hiển thị thông báo “Lỗi! Cập nhật thất bại” nếu thông tin không hợp lệ.  - Nhấn chọn “Thoát” nếu muốn thoát khỏi tác vụ đang thực hiện.  3.3. [Xóa]  - Người dùng nhấn chọn nút “Xóa”.  - Người dùng xác nhận xóa để gửi yêu cầu lên hệ thống. Hệ thống xóa bài viết đó khỏi danh sách và hiển thị thông báo “Xóa thành công” hoặc thông báo “Lỗi! Xóa thất bại”.  - Nhấn chọn “Thoát” nếu muốn thoát khỏi tác vụ đang thực hiện. |
| **Điều kiện sau** | Giao diện quản lý blog được tải thành công |
| **Công nghệ sử dụng** | - Frontend: React.js  - Backend: Node.js  - Database: PostgreSQL  - API: Nestjs và Swagger |
| **Độ phức tạp** | Cao |

* 1. **Phân tích về Sprint Backlog**

|  |  |  |  |
| --- | --- | --- | --- |
| Sprint | Function | Started Date | Due Date |
| Sprint 1 | PB07, PB08, PB09, PB10, PB12, PB13, PB14, PB15, PB16, PB18, PB19, PB20, PB21, PB23, PB01, PB02, PB03 | 29/03/2025 | 20/04/2025 |
| Sprint 2 | PB04, PB05, PB06, PB17, PB24, PB25, PB26, PB11, PB22 | 21/04/2025 | 10/05/2025 |

1. **HẠN CHẾ**

|  |  |
| --- | --- |
| Hạn chế | Tình trạng |
| Thời gian | Thời gian để hoàn thành dự án là 924 giờ. |
| Nhân lực | Toàn bộ nhóm, tổng cộng 5 người làm việc cùng nhau để hoàn thành dự án. |
| Các yêu cầu | Dựa trên các yêu cầu của Product owner. |

1. **TÓM LƯỢC**

|  |  |  |
| --- | --- | --- |
| Name | Mô tả | Role |
| Product Owner | Người đưa ra yêu cầu. | Cung cấp thông tin để phát triển hệ thống. Ra quyết định nghiệm thu và triển khai dự án, làm unit test. |
| Scrum Master | Đây là bên liên quan lãnh đạo, quản lý Nhóm phát triển hệ thống. | Kiểm soát, quản lý, giám sát, đảm bảo dự án hoàn thành đúng thời hạn, ngân sách, đúng kế hoạch và yêu cầu. |
| Requirement Analyzer | Đây là một bên liên quan làm việc với các Nhà phân tích để truyền đạt chính xác các yêu cầu hoặc nhu cầu thành các yêu cầu được sử dụng cho thiết kế. | Chỉ định chi tiết của một hoặc nhiều phần chức năng của hệ thống bằng cách mô tả một hoặc các khía cạnh của yêu cầu.  Điều này sẽ bao gồm chức năng và phi chức năng. |
| Software Architect | Đây là một bên liên quan nắm vai trò chính, dẫn dắt sự phát triển hệ thống. | Chịu trách nhiệm về kiến trúc phần mềm, bao gồm các quyết định kỹ thuật quan trọng hạn chế thiết kế tổng thể cho dự án. |
| Coder | Đây là bộ phận lập trình phần mềm. | Thực hiện dự án. |