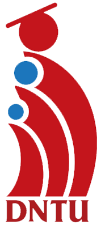
**BỘ GIÁO DỤC VÀ ĐÀO TẠO**



**TRƯỜNG ĐẠI HỌC CÔNG NGHỆ ĐỒNG NAI**

**VÕ THỊ KIỀU OANH**

**LÊ ĐỨC MINH**

**WEBSITE ĐẶT LỊCH KHÁM CHỮA BỆNH CHO PHÒNG KHÁM**

**KHÓA LUẬN TỐT NGHIỆP**

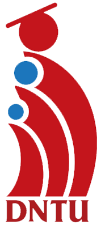
Chuyên ngành: CÔNG NGHỆ THÔNG TIN

Khóa học: 2019 – 2023

**Người hướng dẫn: ThS. Trần Văn Ninh**

**ĐỒNG NAI – 2023**

**BỘ GIÁO DỤC VÀ ĐÀO TẠO**



**TRƯỜNG ĐẠI HỌC CÔNG NGHỆ ĐỒNG NAI**

**VÕ THỊ KIỀU OANH**

**LÊ ĐỨC MINH**

**WEBSITE ĐẶT LỊCH KHÁM CHỮA BỆNH CHO PHÒNG KHÁM**

**KHÓA LUẬN TỐT NGHIỆP**

Chuyên ngành: CÔNG NGHỆ THÔNG TIN

Khóa học: 2019 – 2023

**Người hướng dẫn: ThS. Trần Văn Ninh**

**ĐỒNG NAI – 2023**

TRƯỜNG ĐẠI HỌC CÔNG NGHỆ ĐỒNG NAI **CỘNG HÒA XÃ HỘI CHỦ NGHĨA VIỆT NAM**

**KHOA CÔNG NGHỆ Độc lập – Tự do – Hạnh phúc**

**NHIỆM VỤ KHÓA LUẬN TỐT NGHIỆP**

Sinh viên thực hiện: VÕ THỊ KIỀU OANH , LÊ ĐỨC MINH

- Chuyên ngành: CÔNG NGHỆ THÔNG TIN

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| STT | MSSV | Họ | Tên | Lớp | Khóa học | Ghi chú |
| 1. | 151901497 | Võ Thị Kiều | Oanh | 19DTH4 | 2019-2023 |  |
| 2 | 151900413 | Lê Đức | Minh | 19DTH4 | 2019 - 2023 |  |

1. **Tên đề tài:**

.................................................................................................................................

.................................................................................................................................

1. **Cấu trúc của khóa luận tốt nghiệp**

Chương 1. Tổng quan về đề tài

- Giới thiệu đề tài

- Lý do chọn đề tài

- Nội dung đề tài

- Mục tiêu đề tài

- Phạm vi đề tài

- Phương pháp nghiên cứu và hướng giải quyết đề tài

- Kết quả dự kiến

Chương 2. Cơ sở lý thuyết

- Các lý thuyết liên quan

- Giới thiệu web, chức năng của web

- Tìm hiểu cách thức hoạt động các chức năng

- Nền tảng ngôn ngữ phát triển web

Chương 3. Nội dung và phương pháp nghiên cứu

- Các công cụ phát triển

- Phân tích thiết kế hệ thống

- Thiết kế giao diện

- Thuật toán thực hiện các chức năng ứng dụng

Chương 4. Kết quả, nhận xét và kết luận

- Kết quả đạt được

- Nhận xét

- Kết luận

1. **Ngày giao nhiệm vụ khóa luận:** …/…/20…
2. **Ngày hoàn thành nhiệm vụ:** …/…/20…
3. **Họ tên người hướng dẫn:** ThS/TS Nguyễn Văn Ninh

Nội dung và yêu cầu khóa luận tốt nghiệp đã được thông qua bộ môn và khoa.

*Đồng nai, ngày … tháng … năm 20…*

NGƯỜI HƯỚNG DẪN TRƯỞNG KHOA

*(Ký và ghi rõ họ tên) (Ký và ghi rõ họ tên)*

TỔ TRƯỞNG BỘ MÔN PHẦN GHI KẾT QUẢ KLTN

*(Ký và ghi rõ họ tên) Ngày bảo vệ:*

*Điểm tổng kết:*

**LỜI CAM ĐOAN**

Tôi xin cam đoan đây là công trình nghiên cứu riêng tôi. Các số liệu và kết quả được ghi trong luận văn là trung thực. Các thông tin trích dẫn trong luận văn đã được xác minh rõ nguồn gốc

**LỜI CẢM ƠN**

Trong suốt thời gian học tập và rèn luyện tại Trường Đại học Công Nghệ Đồng Nai cho đến nay, em đã nhận được rất nhiều sự quan tâm, giúp đỡ của quý Thầy Cô và bạn bè. Với lòng biết ơn sâu sắc và chân thành nhất, em xin gửi đến quý Thầy Cô ở Khoa Công Nghệ – Trường Đại học Công Nghệ Đồng Nai đã cùng với tri thức và tâm huyết của mình để truyền đạt vốn kiến thức quý báu cho em trong suốt thời gian học tập tại trường.

Với đề tài “**Website đạt lịch khám bệnh cho phòng khám**” cho môn học Khoá luận tốt nghiệp . Em xin chân thành cảm ơn Th.s Trần Văn Ninh đã tận tâm hướng dẫn những thiếu sót và khó khăn khi thực hiện đề tài, chỉ bảo kinh nghiệm quý báo trong suốt quá trình tìm hiểu, nghiên cứu đề tài.

Em cũng cảm ơn các thành viên trong lớp đã có những ý kiến đóng góp, bổ sung, cũng như động viên chúng em hoàn thành tốt đề tài.

Nhóm thực hiện đề tài đã cố gắng hoàn thiện đồ án, với kiến thức, cũng như kinh nghiệm còn hạn chế của một học viên, bài báo cáo này không thể tránh được những thiếu sót. Em rất mong nhận được sự chỉ bảo, đóng góp ý kiến của các quý thầy cô để tôi có điều kiện bổ sung, nâng cao ý thức của bản thân, phục vụ tốt hơn công tác thực tế sau này.

Em xin chân thành cảm ơn !

BỘ GIÁO DỤC VÀ ĐÀO TẠO **CỘNG HÒA XÃ HỘI CHỦ NGHĨA VIỆT NAM**

**TRƯỜNG ĐẠI HỌC CÔNG NGHỆ ĐỒNG NAI Độc lập – Tự do – Hạnh phúc**

*Đồng nai, ngày … tháng … năm 20…*

**NHẬN XÉT CỦA GIẢNG VIÊN HƯỚNG DẪN**

**Họ tên sinh viên:**

**Về đề tài**:

**Họ và tên giảng viên nhẫn xét:**

**Học hàm, học vị:**

**Đơn vị:**

1. **Về tính cấp thiết của đề tài**

1. **Về nội dung**
   1. **Những ưu điểm**

* 1. **Những nhược điểm**

*.*

1. **Về thái độ, tinh thần thực hiện việc trong quá trình thực hiện khóa luận tốt nghiệp**

1. **Kết luận**

**NGƯỜI HƯỚNG DẪN**

*(Ký và ghi rõ họ tên)*

**CHƯƠNG 1: TỔNG QUAN VỀ ĐỀ TÀI**

**1.1 GIỚI THIỆU VỀ ĐỀ TÀI**

Trong thời đại công nghệ hiện đại, việc đặt lịch khám cho phòng khám trở nên dễ dàng hơn bao giờ hết nhờ sự hỗ trợ của các ứng dụng và công cụ trực tuyến. Đề tài này sẽ tập trung vào việc giới thiệu về quá trình đặt lịch khám cho phòng khám và những lợi ích mà nó mang lại cho cả bệnh nhân và các cơ sở y tế.

Với sự phát triển của công nghệ, việc gọi điện thoại hoặc đến trực tiếp phòng khám để đặt lịch không còn là lựa chọn duy nhất. Thay vào đó, người ta có thể sử dụng các ứng dụng di động hoặc website để thuận tiện và linh hoạt hơn trong việc chọn ngày giờ khám, chọn bác sĩ và xem thông tin về phí dịch vụ.

Đối với bệnh nhân, việc có thể tự do chọn ngày giờ khám theo ý muốn mà không cần phải đi lại hay gọi điện là một tiện ích rất lớn. Ngoài ra, thông tin chi tiết về các bác sĩ và các loại dịch vụ được cung cấp từ các ứng dụng và website giúp bệnh nhân có thể tìm hiểu trước về phòng khám và lựa chọn phù hợp với nhu cầu của mình.

Đối với các cơ sở y tế, việc áp dụng hệ thống đặt lịch trực tuyến giúp quản lý lịch khám hiệu quả hơn, giảm thiểu sự nhầm lẫn trong việc ghi nhận thông tin khách hàng. Ngoài ra, việc tổ chức đặt lịch trực tuyến còn giúp nâng cao trải nghiệm của bệnh nhân và xây dựng lòng tin từ phía khách hàng.

Qua đề tài này, chúng ta sẽ đi vào chi tiết về quá trình đặt lịch khám cho phòng khám và những ưu điểm mà công nghệ mang lại. Hi vọng rằng thông qua việc áp dụng các ứng dụng và công cụ đặt lịch trực tuyến, ta có thể nâng cao hiệu quả trong việc tiếp cận dịch vụ y tế và mang lại sự thuận tiện cho mọi người.

**1.2 LÝ DO CHỌN ĐỀ TÀI**

Việc đặt lịch khám cho phòng khám là một đề tài hết sức quan trọng và hữu ích trong việc nghiên cứu và thực hiện đồ án. Lý do chọn đề tài này là vì nó liên quan trực tiếp đến sự tiện lợi và hiệu quả của việc đi khám bệnh. Trong xã hội ngày càng phát triển, việc đặt lịch khám qua các phòng khám trở nên cần thiết để giúp người dân tiết kiệm thời gian, công sức và tăng tính chuyên nghiệp trong dịch vụ y tế.  
Đề tài này cũng mang lại những lợi ích rõ ràng cho các phòng khám. Việc có một hệ thống đặt lịch thông minh giúp tổ chức công việc một cách hiệu quả, giảm thiểu sai sót trong việc xếp lịch và thuận tiện cho bác sĩ và nhân viên y tế.  
Ngoài ra, viết về đề tài này còn giúp cho sinh viên có cơ hội nắm vững kiến thức về quản lí cuộc sống hàng ngày thông qua ứng dụng công nghệ vào y tế. Đồ án này không chỉ mang tính ứng dụng cao mà còn giúp sinh viên phát triển kỹ năng tư duy logic, quản lí thời gian và tìm hiểu về quy trình làm việc trong phòng khám.  
Với những lý do trên, đề tài đặt lịch khám cho phòng khám là một sự lựa chọn hợp lý và mang lại nhiều giá trị cho người thực hiện đồ án.

**1.3 NỘI DUNG ĐỀ TÀI**

Đồ án này tập trung vào việc xây dựng nội dung liên quan đến việc đặt lịch khám cho phòng khám. Nội dung sẽ giúp người dùng hiểu rõ quy trình và các bước cần thiết để có thể đặt lịch một cách thuận tiện nhất. Các thông tin về các loại hình dịch vụ y tế, thông tin về bác sĩ và chuyên gia trong ngành y tế cũng được cung cấp để giúp người dùng có được cái nhìn tổng quan về phòng khám.

Với những nội dung chi tiết và chính xác, đồ án này hy vọng sẽ mang lại cho người dùng những thông tin hữu ích và giúp họ tự tin trong việc tìm kiếm và đặt lịch khám cho phòng khám.

**1.4 MỤC TIÊU ĐỀ TÀI**

Gồm 3 mục tiêu chính

Củng cố kiến thức, rèn luyên và mở rộng khả năng phân tích hệ thống thông tin và lập trình.

Xây dựng website đặt lịch có thể áp dụng vào thực tế

Hoàn thành khoá luận tốt nghiệp và thực hành nghiên cứu và lập trình Node js, Express js, Angular

**1.5 PHẠM VI ĐỀ TÀI**

Sử dụng Visual Studio Code và MongoDb

Xây dựng phần mềm ứng dụng trên Node js liên kết với mongo để quản lý website

**1.6 PHƯƠNG PHÁP NGHIÊN CỨU VÀ HƯỚNG GIẢI QUYẾT ĐỀ TÀI**

Phương pháp nghiên cứu :

Thu thập thông tin từ các trang mạng và xin số liệu từ các website của các phòng khám

Tham khảo các tài liệu hướng dẫn lập trình trên mạng, trên youtube và các tài liệu liên quan

Trao đổi ý kiến từ các giảng viên, các người có hoạt động cùng đề tài

Tìm hiểu và nghiên cứu lý thuyết dựa trên sự hướng dẫn của giảng viên và tài liệu sưu tầm được

Kế hoạch làm việc theo dự tính

Bước 1: Tìm hiểu hoạt động đặt lịch khám của các phòng khám

Bước 2: Tìm hiểu về Node js, Express js và Angular và kết nối database

Bước 3: Phân tích thiết kế hệ thông website

Bước 4: Xây dựng website đặt lịch khám bệnh cho phòng khám

Bước 5: Chỉnh sửa thêm giao diện (hoặc API) (nếu có), kiểm tra và sửa các lỗi chương trình

**1.7 KẾT QUẢ DỰ KIẾN**

Phân tích được hệ thống hoạt dộng của website, đồng thời kiểm soát hoạt động và cách quản lý website.

Hiểu được cách sử dụng MongoDB, Expressjs, Angular và Nodejs

Biết được cách thức hoạt động từ đó xây dựng được website

Hoàn thành thiết kế và xây dựng chức năng, giao diện của website:

+Phân quyền quản lý của admin  
+Thêm các thông tin liên quan và lưu xuống database  
+Thực hiện đặt lịch khám bệnh cho bệnh nhân

**CHƯƠNG 2: CƠ SỞ LÝ THUYẾT**

**2.1 WEBSITE VÀ CÁCH THỨC HOẠT ĐỘNG**

**2.1.1 Giới thiệu web**

**Website đặt lịch khám bệnh cho phòng khám** là nơi bạn có thể dễ dàng đặt lịch khám bệnh tại các phòng .Trang web này được thiết kế và phát triển nhằm giúp người dùng tiết kiệm thời gian và nỗ lực khi muốn hẹn lịch khám bệnh. Thay vì phải gọi điện hoặc đến trực tiếp phòng khám để đặt lịch, người dùng chỉ cần mở trang web, chọn thời gian khám mong muốn và nhập thông tin cá nhân, sau đó chỉ cần một vài thao tác đơn giản là xong!

Trang web này không chỉ mang lại sự thuận tiện cho người dùng mà còn giúp các phòng khám quản lý lịch làm việc hiệu quả hơn. Bạn có thể yên tâm rằng thông tin cá nhân của bạn sẽ được bảo mật hoàn toàn và chỉ được sử dụng cho mục đích xác nhận và thông báo lịch hẹn.  
**2.1.2 Lịch sử phát triển**

Ban đầu, việc đặt lịch khám bệnh vẫn được thực hiện thông qua cuộc gọi điện thoại hoặc trực tiếp tới phòng khám. Đây là một quá trình tốn thời gian và không tiện lợi cho cả bác sĩ và bệnh nhân.

Tuy nhiên, với sự phát triển của internet và công nghệ thông tin, các website đặt lịch khám bệnh đã xuất hiện. Những người sáng tạo trong ngành y học đã nhận ra tiềm năng của việc áp dụng công nghệ để cải thiện quy trình này.   
Nhờ vào các ứng dụng di động và website, bất kỳ ai có kết nối internet cũng có thể dễ dàng đặt lịch hẹn với các phòng khám y tế.  
**2.1.3 Ý nghĩa của việc sử dụng Website**

Việc đặt lịch khám bệnh trực tuyến giúp bệnh nhân tiết kiệm thời gian và công sức. Thay vì phải gọi điện hay đến trực tiếp phòng khám để xếp hàng chờ lấy số, bằng cách sử dụng website đặt lịch, họ có thể dễ dàng chọn ngày và giờ hẹn mà không cần di chuyển. Điều này không chỉ giúp tiết kiệm thời gian mà còn giảm thiểu sự phiền toái cho bệnh nhân.

Đối với các phòng khám y tế, việc có một website đặt lịch mang lại nhiều lợi ích. Trước tiên, nó giúp tổ chức quản lý cuộc hẹn một cách hiệu quả. Nhân viên không cần phải ngồi điện thoại liên tục để lấy thông tin và xếp lịch cho bệnh nhân mà có thể dễ dàng theo dõi và cập nhật thông tin trên hệ thống.

Thứ hai, việc sử dụng website đặt lịch giúp phòng khám tăng cường quảng bá và tiếp cận khách hàng tiềm năng. Bằng cách hiển thị thông tin về các dịch vụ, bác sĩ chuyên khoa và các đánh giá từ bệnh nhân đã điều trị thành công, website đặt lịch khám bệnh giúp xây dựng lòng tin và thu hút người dùng mới.

Tóm lại, việc sử dụng website đặt lịch khám bệnh cho phòng khám mang lại ý nghĩa to lớn cho cả bệnh nhân và các phòng khám y tế. Nó không chỉ mang lại tiện ích trong việc đặt hẹn một cách thuận tiện mà còn tạo ra hiệu quả trong quản lý cuộc hẹn và tăng cường quảng bá thương hiệu.  
**2.1.4 Giới thiệu về hệ thống**

Website được lập trình dựa trên việc kết hợp Nodejs, Expressjs, Angular và MongoDB, có giao diện gọn gàng và dễ dàng sử dụng

Cung cấp cho người sử dụng một tuỳ chọn để lưu trữ và quản lý các thông tin cơ bản về phòng khám,… và tiếp nhận các lịch đặt của bệnh nhân (chức năng của phòng khám), và cho phép lựa chọn đặt lịch khám bệnh (chức năng của bệnh nhân)

**2.1.5 Cấu tạo chức năng của Web**

**2.1.6 Các phần mềm tương tự hiện nay**

**2.2 NỀN TẢNG LẬP TRÌNH (MEAR STACK VÀ VISUAL STUDIO CODE)**

2.2.1 Tổng quan về MEAN STACK

Giới thiệu

MEAN stack là một công nghệ phát triển ứng dụng web đa nền tảng (full-stack) được xây dựng bằng cách kết hợp các công nghệ sau đây: MongoDB, Express.js, Angular và Node.js. Các thành phần này cùng nhau tạo nên một cấu trúc hoàn chỉnh để phát triển ứng dụng web từ phía máy chủ (server-side) đến phía người dùng (client-side).

Đặc điểm:

Tính linh hoạt và khả năng mở rộng: MEAN Stack cho phép linh hoạt trong việc phát triển và mở rộng các ứng dụng web. Sử dụng các công nghệ và framework phổ biến, bạn có thể tận dụng sức mạnh của MongoDB, Express.js, Angular và Node.js để xây dựng ứng dụng linh hoạt và dễ mở rộng.

Tích hợp dễ dàng: Các công nghệ trong MEAN Stack được thiết kế để tích hợp tốt với nhau. Dữ liệu có thể được lưu trữ trong MongoDB và truy vấn bằng sử dụng Express.js API, trong khi Angular được sử dụng để xây dựng giao diện người dùng phía máy khách. Node.js đóng vai trò là môi trường chạy cho toàn bộ ứng dụng.

Xây dựng ứng dụng đa nền tảng: Với MEAN Stack, bạn có thể xây dựng ứng dụng web đa nền tảng có thể chạy trên cả desktop và di động. Angular cung cấp khả năng xây dựng giao diện người dùng phản hồi (responsive UI), trong khi Node.js cho phép ứng dụng chạy trên máy chủ phía máy khách.

Cộng đồng phát triển và hỗ trợ: MEAN Stack có một cộng đồng phát triển rộng lớn và hỗ trợ từ các nhà phát triển trên toàn thế giới. Các tài liệu, ví dụ và thư viện được cung cấp để giúp nhà phát triển tận dụng tối đa các công nghệ trong MEAN Stack.

Mục tiêu phát triển

Tích hợp đầy đủ: MEAN stack cung cấp một cấu trúc phát triển hoàn chỉnh từ phía máy chủ đến phía người dùng, giúp đơn giản hóa việc phát triển và triển khai ứng dụng web. Sự linh hoạt: MEAN stack cho phép phát triển ứng dụng web đa nền tảng, cho phép nền tảng di động và máy tính để bàn sử dụng cùng một mã nguồn. Hiệu suất cao: Sử dụng Node.js và Express.js, MEAN stack cho phép xây dựng các ứng dụng web có hiệu suất cao, có khả năng xử lý đồng thời hàng ngàn người dùng. Tích hợp dữ liệu linh hoạt: MongoDB là một hệ quản trị cơ sở dữ liệu phi quan hệ (NoSQL) linh hoạt, cho phép lưu trữ dữ liệu dưới dạng tài liệu JSON. Điều này giúp đơn giản hóa việc thay đổi cấu trúc dữ liệu và mở rộng ứng dụng. Phát triển nhanh chóng: Với các công nghệ và framework như Angular, Express.js và Node.js, MEAN stack giúp phát triển ứng dụng web nhanh chóng và dễ dàng. Các công cụ phát triển và tích hợp sẵn trong MEAN stack giúp tiết kiệm thời gian và công sức trong quá trình phát triển. Cộng đồng mạnh mẽ: MEAN stack có một cộng đồng phát triển rộng lớn, với nhiều tài liệu, tài nguyên và các dự án mã nguồn mở hỗ trợ. Điều này giúp người phát triển dễ dàng tìm kiếm giải pháp và hỗ trợ trong quá trình phát triển ứng dụng. Tóm lại, MEAN stack là một công nghệ phát triển ứng dụng web đa nền tảng mạnh mẽ và linh hoạt. Với sự kết hợp của MongoDB, Express.js, Angular và Node.js, MEAN stack giúp người phát triển xây dựng và triển khai các ứng dụng web hiệu quả, linh hoạt và có hiệu suất cao.

2.2.2 Nodejs

Giới thiệu:

-NodeJs là một runtime environment (môi trường chạy mã), được sử dụng ở dạng đa nền tảng và mã nguồn mở. NodeJs được xây dựng và phát triển từ năm 2009 trên nền tảng Chrome's V8 JavaScript engine. Nó cho phép bạn phát triển ứng dụng mạnh mẽ và hiệu quả cho web và các dịch vụ máy chủ.

-Với Node.js, bạn có thể viết mã JavaScript để xây dựng các ứng dụng máy chủ, xử lý các tác vụ đồng thời và xây dựng API phía máy chủ. Node.js được thiết kế để xử lý hàng nghìn kết nối đồng thời mà không gây ra sự chậm trễ cho người dùng cuối. Điều này làm cho nó trở thành một lựa chọn phổ biến cho việc xây dựng các ứng dụng thời gian thực, ứng dụng trò chơi và ứng dụng web có khả năng mở rộng cao.

-Một trong những điểm mạnh của Node.js là hệ sinh thái package manager NPM (Node Package Manager). NPM là một kho chứa các thư viện JavaScript có sẵn mà bạn có thể sử dụng trong dự án của mình. Nó cho phép bạn tìm và cài đặt các gói phần mềm từ kho NPM, giúp tiết kiệm thời gian và nỗ lực trong việc phát triển.

-Node.js cũng hỗ trợ xây dựng ứng dụng web thời gian thực thông qua framework Express.js. Express.js là một framework phát triển nhanh và mạnh mẽ cho Node.js, giúp bạn xây dựng các API và ứng dụng web nhanh chóng và dễ dàng. Nó cung cấp các tính năng mạnh mẽ như routing, middleware, và quản lý session để phát triển ứng dụng web linh hoạt và dễ bảo trì.

-Một trong những lợi ích khác của Node.js là khả năng chia sẻ mã giữa phía máy khách và phía máy chủ. Bạn có thể sử dụng cùng một mã JavaScript để xây dựng logic phía máy khách và phía máy chủ, giúp tiết kiệm thời gian và công sức phát triển.

-Với sự phát triển nhanh chóng và một cộng đồng mạnh mẽ, Node.js trở thành một công cụ mạnh mẽ trong việc phát triển ứng dụng web hiện đại. Điều này giúp các nhà phát triển tận dụng tiềm năng của JavaScript cả ở phía máy khách và phía máy chủ, đồng thời xây dựng các ứng dụng hiệu quả và linh hoạt.

Đặc điểm:

-Node.js có một số đặc điểm quan trọng mà làm nổi bật nó trong việc phát triển ứng dụng máy chủ và web. Dưới đây là một số đặc điểm quan trọng của Node.js:

+Hệ sinh thái phi đồng bộ: Node.js hoạt động dựa trên mô hình phi đồng bộ (asynchronous) và sự kiện (event-driven). Điều này cho phép Node.js xử lý các tác vụ I/O (Input/Output) mà không chặn trở lại (non-blocking), giúp tối ưu hóa hiệu suất của ứng dụng. Thay vì đợi các hoạt động I/O hoàn thành trước khi tiếp tục, Node.js tiếp tục xử lý các tác vụ khác và gọi lại (callback) khi hoạt động I/O hoàn thành.

+Hiệu năng cao: Nhờ kiến ​​trúc phi đồng bộ và non-blocking, Node.js đạt được hiệu năng cao khi xử lý các yêu cầu đồng thời. Nó có thể xử lý hàng nghìn kết nối đồng thời với tài nguyên máy tính tương đối ít. Điều này làm cho Node.js trở thành một lựa chọn phổ biến cho việc xây dựng các ứng dụng thời gian thực và ứng dụng có khả năng mở rộng cao.

+Mã chung giữa máy chủ và máy khách: Một trong những lợi ích lớn của Node.js là khả năng chia sẻ mã giữa phía máy khách (client-side) và phía máy chủ (server-side). Với Node.js, bạn có thể sử dụng cùng một mã JavaScript để xây dựng logic phía máy khách và phía máy chủ. Điều này giảm thiểu sự trùng lặp và giúp tiết kiệm thời gian và công sức phát triển.

+Hỗ trợ mạnh mẽ từ NPM: Node.js có hệ sinh thái package manager NPM (Node Package Manager) với hàng ngàn thư viện JavaScript có sẵn. NPM cung cấp một kho chứa các gói phần mềm để bạn có thể sử dụng trong dự án của mình. Điều này giúp bạn tiết kiệm thời gian và công sức trong việc phát triển, vì bạn có thể sử dụng các gói đã được phát triển sẵn để thực hiện các chức năng cần thiết.

+Xử lý đa luồng thông qua các tiến trình con: Mặc dù Node.js chạy trên một luồng duy nhất, nhưng nó có khả năng xử lý đa luồng thông qua sử dụng các tiến trình con (child processes). Bằng cách tạo các tiến trình con, Node.js có thể xử lý các tác vụ đa luồng mà không chặn hoạt động của ứng dụng.

+Module hóa: Node.js khuyến khích module hóa trong việc xây dựng ứng dụng. Bạn có thể chia ứng dụng thành các mô-đun nhỏ và sử dụng lại chúng trong các dự án khác nhau. Điều này giúp tăng tính tái sử dụng, dễ bảo trì và quản lý mã nguồn.

+Cộng đồng mạnh mẽ: Node.js có một cộng đồng phát triển mạnh mẽ và sôi nổi. Cộng đồng này cung cấp rất nhiều tài nguyên, hướng dẫn và thư viện phong phú để hỗ trợ trong việc phát triển ứng dụng Node.js.

Tóm lại, Node.js có các đặc điểm quan trọng như kiến ​​trúc phi đồng bộ, hiệu năng cao, khả năng chia sẻ mã giữa máy chủ và máy khách, hỗ trợ mạnh mẽ từ NPM, xử lý đa luồng thông qua các tiến trình con, mô-đun hóa và cộng đồng mạnh mẽ. Điều này làm cho Node.js trở thành một công cụ mạnh mẽ và phổ biến trong việc phát triển ứng dụng máy chủ và web.

Mục tiêu phát triển:

-Mục tiêu phát triển của NodeJs bao gồm:

+Tăng cường hiệu suất và khả năng mở rộng: Node.js luôn tập trung vào việc cải thiện hiệu suất và khả năng mở rộng của nền tảng. Điều này đảm bảo rằng Node.js có thể xử lý một lượng lớn yêu cầu đồng thời mà không gây ra sự chậm trễ hoặc gián đoạn cho người dùng cuối.

+Mở rộng khả năng sử dụng: Node.js đang cố gắng mở rộng phạm vi sử dụng của nó từ việc phát triển ứng dụng web đến các lĩnh vực khác như Internet of Things (IoT), thời gian thực, trò chơi, và ứng dụng di động. Mục tiêu là tạo ra một nền tảng đa dụng có thể đáp ứng nhu cầu đa dạng của các ngành công nghiệp.

+Tăng tính bảo mật: Node.js đang liên tục nâng cao tính bảo mật của mình để đối phó với các mối đe dọa và lỗ hổng bảo mật. Cộng đồng Node.js và nhóm phát triển chính thức luôn làm việc để cung cấp các cập nhật và sửa lỗi an ninh để đảm bảo rằng các ứng dụng Node.js được phát triển và triển khai an toàn.

+Đa nền tảng: Node.js đang phát triển để hỗ trợ đa nền tảng, cho phép phát triển ứng dụng Node.js trên nhiều hệ điều hành như Windows, macOS và Linux. Điều này tạo điều kiện thuận lợi cho việc phát triển và triển khai trên các môi trường khác nhau.

+Hỗ trợ công cụ và thư viện phong phú: Node.js tiếp tục tăng cường hệ sinh thái công cụ và thư viện của mình. NPM (Node Package Manager) là một phần quan trọng trong việc hỗ trợ phát triển ứng dụng Node.js bằng cách cung cấp hàng ngàn gói phần mềm được phát triển sẵn. Node.js cũng cung cấp các công cụ phát triển, framework và thư viện như Express.js, Socket.IO, Mocha và nhiều hơn nữa để giúp nhà phát triển xây dựng ứng dụng nhanh chóng và hiệu quả.

+Đồng hành với xu hướng công nghệ: Node.js luôn điều chỉnh và đồng hành với xu hướng công nghệ mới nhất. Nó liên tục cập nhật và tích hợp các tính năng mới để hỗ trợ các công nghệ mới như Machine Learning, Serverless Computing, Blockchain và nhiều hơn nữa. Mục tiêu là để Node.js tiếp tục là một nền tảng linh hoạt và đáp ứng được những thách thức công nghệ mới trong tương lai.

Tổng quan, mục tiêu phát triển của Node.js là tăng cường hiệu suất, khả năng mở rộng, tính bảo mật, hỗ trợ đa nền tảng, cung cấp công cụ và thư viện phong phú, đồng hành với xu hướng công nghệ mới và phục vụ một loạt các lĩnh vực ứng dụng khác nhau.

Lịch sử:

-Node.js có một lịch sử phát triển thú vị, bắt đầu từ năm 2009 và đã trở thành một trong những công nghệ phát triển ứng dụng phía máy chủ phổ biến nhất. Dưới đây là một cái nhìn tổng quan về lịch sử của Node.js:

+Năm 2009: Node.js được tạo ra bởi Ryan Dahl, một nhà phát triển phần mềm người Mỹ. Ý tưởng ban đầu của Dahl là tạo ra một môi trường chạy mã JavaScript ở phía máy chủ dựa trên Chrome's V8 JavaScript engine. Node.js được phát triển nhằm giải quyết nhược điểm của các máy chủ truyền thống dựa trên luồng, như khả năng mở rộng kém và khả năng xử lý đồng thời thấp.

+Năm 2009-2011: Node.js thu hút sự quan tâm của cộng đồng phát triển và trở thành một dự án mã nguồn mở. Dahl đã công bố phiên bản đầu tiên của Node.js (v0.1.14) và cùng với cộng đồng người dùng, tiếp tục phát triển nền tảng này.

+Năm 2011: Node Package Manager (NPM) ra đời, cung cấp một cách tiện lợi để quản lý và chia sẻ các gói phần mềm JavaScript. NPM đã đóng góp rất lớn vào việc mở rộng hệ sinh thái và sự phát triển của Node.js.

+Năm 2012: Node.js Foundation được thành lập, nhằm quản lý và hướng dẫn sự phát triển của Node.js. Điều này đồng nghĩa với việc Node.js được hỗ trợ bởi một tổ chức phi lợi nhuận và có một quá trình quản lý và phát triển chặt chẽ hơn.

+Năm 2015: Node.js v4.0.0 được phát hành, đánh dấu bước đột phá quan trọng với việc hỗ trợ ES6 (ECMAScript 2015) - phiên bản tiêu chuẩn mới nhất của JavaScript. Điều này đẩy mạnh tính năng và khả năng sử dụng của Node.js.

+Năm 2018: Node.js v10.0.0 ra mắt, đưa vào nền tảng nhiều cải tiến, cải thiện hiệu suất và bảo mật. Phiên bản này cũng đánh dấu việc hỗ trợ dài hạn (LTS - Long Term Support) của Node.js, đảm bảo sự ổn định và bảo mật cho các ứng dụng lâu dài.

+Từ năm 2018 đến nay: Node.js tiếp tục phát triển và cải tiến, với việc ra mắt các phiên bản mới và tích hợp các tính năng tiên tiến. Node.js cũng trở thành một công nghệ phổ biến cho việc xây dựng các ứng dụng web thời gian thực, ứng dụng di động, và cả ứng dụng IoT.

2.2.3 Expressjs

Giới thiệu:

-Express.js là một framework phát triển ứng dụng web phía máy chủ (server-side) dựa trên Node.js. Nó được thiết kế để xây dựng các ứng dụng web nhanh chóng, đơn giản và linh hoạt. Với Express.js, bạn có thể xây dựng các API và ứng dụng web một cách dễ dàng và hiệu quả.

-Express.js cung cấp một tập hợp các tính năng mạnh mẽ như routing, middleware, và quản lý session. Điều này giúp bạn xử lý các yêu cầu HTTP một cách linh hoạt và tiện lợi. Bằng cách xác định các tuyến đường (routes) và các xử lý (handlers) tương ứng, bạn có thể điều hướng yêu cầu đến các phần tử khác nhau trong ứng dụng của mình. Điều này cho phép bạn xây dựng các ứng dụng web có cấu trúc dễ bảo trì và mở rộng.

-Express.js cũng hỗ trợ middleware, là các chức năng trung gian được thực thi trước khi yêu cầu đến các xử lý cuối cùng. Điều này cho phép bạn thực hiện các tác vụ như xác thực người dùng, xử lý lỗi, ghi nhật ký (logging), và nhiều hơn nữa. Bạn có thể tùy chỉnh các middleware có sẵn hoặc viết middleware tùy chỉnh của riêng bạn để xử lý các yêu cầu theo nhu cầu cụ thể của ứng dụng.

-Express.js cũng có khả năng tích hợp với các công cụ và thư viện khác của cộng đồng Node.js. Bạn có thể sử dụng NPM (Node Package Manager) để tìm và cài đặt các gói phần mềm có sẵn cho Express.js, giúp bạn tiết kiệm thời gian và công sức trong việc phát triển. Ngoài ra, Express.js cũng có khả năng tích hợp với các ORM (Object-Relational Mapping), template engines, và các công cụ khác để giúp bạn xây dựng ứng dụng mạnh mẽ và linh hoạt.

Với cú pháp đơn giản và tài liệu phong phú, Express.js là một lựa chọn phổ biến cho việc phát triển ứng dụng web phía máy chủ. Nó cung cấp một cách tiếp cận tối giản và linh hoạt để xây dựng các ứng dụng web hiệu quả và dễ bảo trì.

Đặc điểm:

-Express.js có một số đặc điểm quan trọng giúp nó trở thành một framework phát triển ứng dụng web phía máy chủ phổ biến. Dưới đây là một số đặc điểm chính của Express.js:

+Hỗ trợ routing linh hoạt: Express.js cung cấp một cách tiếp cận đơn giản và mạnh mẽ để định nghĩa và quản lý các tuyến đường (routes) trong ứng dụng web. Bằng cách kết hợp các URL với các xử lý tương ứng, bạn có thể xác định cách yêu cầu từ người dùng sẽ được xử lý và trả về phản hồi tương ứng.

+Middleware mạnh mẽ: Express.js sử dụng middleware để thực hiện các chức năng trung gian giữa yêu cầu và xử lý cuối cùng. Điều này cho phép bạn thực hiện các tác vụ như xác thực, ghi nhật ký, xử lý lỗi, và nhiều hơn nữa. Middleware giúp tách biệt logic xử lý và cung cấp khả năng mở rộng linh hoạt cho ứng dụng.

+Quản lý session và cookie: Express.js hỗ trợ quản lý session và cookie một cách dễ dàng. Bạn có thể lưu trữ thông tin người dùng trong session và sử dụng cookie để duy trì trạng thái của người dùng qua các yêu cầu khác nhau. Điều này giúp xây dựng các ứng dụng có khả năng xác thực và theo dõi người dùng.

+Template engine linh hoạt: Express.js cho phép bạn sử dụng các template engine như EJS, Pug, Handlebars để tạo và hiển thị giao diện người dùng. Điều này giúp tách biệt logic xử lý và giao diện, và giúp bạn dễ dàng tạo các trang HTML động và đáp ứng.

+Hỗ trợ API phía máy chủ: Express.js là một lựa chọn phổ biến cho việc xây dựng các API phía máy chủ. Nó hỗ trợ các phương thức HTTP như GET, POST, PUT, DELETE và cho phép bạn xây dựng các API một cách dễ dàng và linh hoạt. Bạn có thể truy cập các tham số yêu cầu, kiểm tra và xử lý dữ liệu đầu vào, và trả về các phản hồi tương ứng với các yêu cầu API.

+Hỗ trợ middleware và thư viện mạnh mẽ: Express.js được tích hợp với NPM (Node Package Manager), cho phép bạn sử dụng các middleware và thư viện có sẵn từ cộng đồng Node.js. Có hàng ngàn gói phần mềm được phát triển sẵn để giúp bạn xây dựng các tính năng phức tạp như xác thực người dùng, gửi email, xử lý hình ảnh, và nhiều hơn nữa.

+Dễ học và dễ sử dụng: Express.js có cú pháp đơn giản và tài liệu phong phú, làm cho việc học và sử dụng framework này trở nên dễ dàng. Bạn có thể nhanh chóng bắt đầu xây dựng các ứng dụng web phía máy chủ với Express.js mà không cần mất nhiều thời gian.

Tóm lại, Express.js có các đặc điểm như routing linh hoạt, middleware mạnh mẽ, quản lý session và cookie, template engine linh hoạt, hỗ trợ API phía máy chủ, hỗ trợ middleware và thư viện phong phú, cùng với tính dễ học và sử dụng. Điều này giúp Express.js trở thành một framework phát triển ứng dụng web phía máy chủ phổ biến và được ưa chuộng.

Mục tiêu phát triển:

-Mục tiêu phát triển của Express.js bao gồm:

+Đơn giản và dễ sử dụng: Một trong những mục tiêu chính của Express.js là cung cấp một cách tiếp cận đơn giản và dễ sử dụng để phát triển ứng dụng web phía máy chủ. Express.js cho phép nhà phát triển tập trung vào việc xây dựng ứng dụng mà không phải vướng mắc trong cú pháp phức tạp hoặc quy tắc khó hiểu.

+Linh hoạt và mở rộng: Express.js cung cấp sự linh hoạt cho nhà phát triển trong việc xây dựng ứng dụng web. Bạn có thể tuỳ chỉnh các tuyến đường (routes) và middleware theo nhu cầu cụ thể của ứng dụng của mình. Điều này giúp tạo ra các ứng dụng linh hoạt và dễ mở rộng, có khả năng thích ứng với nhu cầu phát triển và thay đổi của dự án.

+Hỗ trợ API phía máy chủ: Express.js được thiết kế để hỗ trợ việc xây dựng các API phía máy chủ một cách dễ dàng. Mục tiêu là cung cấp một cú pháp và các công cụ hữu ích để xây dựng và quản lý các API hiệu quả, từ việc xác thực người dùng đến xử lý yêu cầu và trả về phản hồi tương ứng.

+Tích hợp dễ dàng: Express.js tạo điều kiện thuận lợi cho việc tích hợp với các công cụ và thư viện khác của cộng đồng Node.js. Bạn có thể sử dụng các gói phần mềm có sẵn từ NPM (Node Package Manager) để mở rộng khả năng và tính năng của ứng dụng Express.js. Điều này giúp tiết kiệm thời gian và công sức trong việc phát triển ứng dụng.

+Hiệu suất cao: Express.js chú trọng đến hiệu suất của ứng dụng web. Nó cung cấp cơ chế xử lý đồng thời (concurrency) và hỗ trợ sự kiện (event-driven) để đảm bảo ứng dụng hoạt động mượt mà và phản hồi nhanh chóng đối với các yêu cầu từ người dùng.

+Hướng tới cộng đồng: Express.js đang phát triển và tiếp tục lắng nghe ý kiến và đóng góp từ cộng đồng người dùng. Mục tiêu là xây dựng một framework được cập nhật, bảo trì và nâng cấp liên tục để đáp ứng các yêu cầu và xu hướng phát triển mới nhất trong lĩnh vực phát triển ứng dụng web.

Tóm lại, mục tiêu phát triển của Express.js là cung cấp một cách tiếp cận đơn giản và linh hoạt cho việc phát triển ứng dụng web phía máy chủ. Đồng thời, nó hướng đến việc tối ưu hiệu suất, tích hợp dễ dàng, hỗ trợ phát triển API phía máy chủ và đáp ứng nhu cầu của cộng đồng người dùng.

Lịch sử:

-Express.js có một lịch sử phát triển đáng chú ý, bắt đầu từ năm 2010 và đã trở thành một trong những framework phát triển ứng dụng web phía máy chủ phổ biến nhất. Dưới đây là một cái nhìn tổng quan về lịch sử của Express.js:

+Năm 2010: Express.js được tạo ra bởi TJ Holowaychuk, một nhà phát triển phần mềm người Anh. Ý tưởng ban đầu của Holowaychuk là tạo ra một framework nhẹ, đơn giản và linh hoạt để xây dựng các ứng dụng web phía máy chủ. Express.js được phát triển dựa trên mô hình middleware và là một phần mở rộng của framework Connect.

+Năm 2010-2014: Express.js thu hút sự quan tâm và sự ủng hộ từ cộng đồng phát triển, trở thành một dự án mã nguồn mở phổ biến. Holowaychuk và cộng đồng người dùng đã tiếp tục phát triển và cải tiến Express.js thông qua việc thêm tính năng mới và sửa lỗi.

+Năm 2014: TJ Holowaychuk thông báo rằng ông sẽ rút lui khỏi việc phát triển Express.js và chuyển giao dự án cho cộng đồng để tiếp tục phát triển và bảo trì. Điều này đồng nghĩa với việc Express.js được hỗ trợ bởi một nhóm phát triển chính thức và cộng đồng người dùng.

+Từ năm 2014 đến nay: Express.js tiếp tục phát triển và trở thành một trong những framework phát triển ứng dụng web phía máy chủ phổ biến và được ưa chuộng. Các phiên bản mới được phát hành thường xuyên, đưa vào nền tảng các tính năng và cải tiến. Express.js cũng tích hợp với các công cụ và thư viện khác của cộng đồng Node.js, giúp nhà phát triển xây dựng ứng dụng phức tạp một cách dễ dàng.

+Năm 2020: Express.js đạt cột mốc quan trọng khi phiên bản 5.0.0-alpha.7 được phát hành. Phiên bản này đánh dấu một bước tiến lớn với việc hỗ trợ HTTP/2 và các cải tiến hiệu suất khác.

Express.js đã trải qua một hành trình phát triển đáng kể và đã trở thành một framework phát triển ứng dụng web phía máy chủ rất mạnh mẽ và linh hoạt. Sự tiếp tục của Express.js trong việc cải tiến và phát triển hứa hẹn mang lại nhiều tính năng mới và tiện ích hơn cho việc xây dựng ứng dụng web phía máy chủ trong tương lai.

2.2.4 Angular

Giới thiệu:

-Angular là một framework phát triển ứng dụng web phía máy khách (client-side) mã nguồn mở, được phát triển bởi Google.

-Angular giúp xây dựng các ứng dụng web độc lập với nền tảng, linh hoạt và dễ bảo trì. Với Angular, bạn có thể tạo ra các ứng dụng đáng tin cậy và phức tạp một cách dễ dàng, từ các ứng dụng doanh nghiệp lớn đến các ứng dụng di động đáp ứng.

-Với cộng đồng phát triển rộng lớn và sự hỗ trợ từ Google, Angular đảm bảo rằng bạn có tài liệu phong phú và sự cải tiến liên tục. Angular cũng hướng đến việc tối ưu hiệu suất và hỗ trợ đa nền tảng, cho phép bạn xây dựng các ứng dụng web đúng thời gian thực và di động.

Đặc điểm:

-Một số đặc điểm quan trọng của Angular bao gồm:

+TypeScript: Angular được viết bằng TypeScript, một ngôn ngữ lập trình phục vụ cho phát triển ứng dụng web phía máy khách. TypeScript là một phiên bản mở rộng của JavaScript, cung cấp các tính năng như kiểu dữ liệu tĩnh, hỗ trợ cho OOP (Object-Oriented Programming) và các tính năng nâng cao khác, giúp tăng tính rõ ràng và bảo trì của mã nguồn.

+Kiến trúc gói: Angular sử dụng kiến trúc gói (component-based architecture) để xây dựng ứng dụng. Kiến trúc này cho phép bạn chia ứng dụng thành các thành phần độc lập như thành phần (component), dịch vụ (service) và định tham số (directive). Điều này giúp tách biệt logic và giao diện, giảm sự phức tạp và tăng khả năng mở rộng.

+Two-way Data Binding: Angular cung cấp two-way data binding, cho phép dữ liệu trong model và giao diện (view) được đồng bộ hóa tự động. Khi dữ liệu trong model thay đổi, giao diện sẽ được cập nhật và ngược lại. Điều này giúp tiết kiệm thời gian và công sức trong việc quản lý dữ liệu và tạo ra trải nghiệm tương tác tốt hơn cho người dùng.

+Dependency Injection: Angular hỗ trợ Dependency Injection (DI), cho phép bạn quản lý các phụ thuộc (dependencies) trong ứng dụng một cách dễ dàng và linh hoạt. DI giúp tách biệt logic và cung cấp khả năng kiểm thử tốt hơn và tái sử dụng mã nguồn.

+Các tính năng nâng cao: Angular cung cấp nhiều tính năng mạnh mẽ khác như routing, animation, forms, validation, HTTP module và nhiều hơn nữa. Các tính năng này giúp bạn xây dựng các ứng dụng đa dạng và đáp ứng được nhu cầu phức tạp của dự án.

Với sự mạnh mẽ và tính toàn diện của nó, Angular là một lựa chọn phổ biến cho việc phát triển ứng dụng web phía máy khách trong MEAN stack. Nó cung cấp một cách tiếp cận hiện đại và mạnh mẽ để xây dựng các ứng dụng web phức tạp và đáng tin cậy.

Mục tiêu phát triển:

-Mục tiêu phát triển của Angular bao gồm:

+Hiệu suất và tải trọng: Một trong những mục tiêu chính của Angular là đảm bảo hiệu suất cao và tải trọng thấp cho các ứng dụng web. Angular tập trung vào việc tối ưu hóa và cải thiện hiệu suất của ứng dụng, từ quá trình biên dịch (compilation) đến việc tải trang và xử lý dữ liệu. Điều này giúp cung cấp trải nghiệm tốt hơn cho người dùng và tăng khả năng mở rộng của ứng dụng.

+Tính linh hoạt và mở rộng: Angular được thiết kế để cung cấp tính linh hoạt và khả năng mở rộng cao cho việc phát triển ứng dụng. Kiến trúc gói (component-based architecture) của Angular giúp tách biệt logic và giao diện, tạo điều kiện thuận lợi cho việc tái sử dụng và mở rộng các thành phần trong ứng dụng. Đồng thời, Angular hỗ trợ Dependency Injection (DI) để quản lý các phụ thuộc và giảm sự phức tạp của mã nguồn.

+Bảo trì và sửa lỗi dễ dàng: Angular đặt mục tiêu để tạo ra các ứng dụng dễ bảo trì và sửa lỗi. Bằng cách tách biệt logic và giao diện, Angular giúp rõ ràng hóa mã nguồn và giảm khả năng xảy ra lỗi. Kiến trúc gói và Dependency Injection cũng cung cấp khả năng kiểm thử tốt hơn và giảm thời gian sửa lỗi.

+Hỗ trợ đa nền tảng: Angular đặt mục tiêu hỗ trợ đa nền tảng, cho phép bạn xây dựng ứng dụng web một cách dễ dàng trên nhiều nền tảng khác nhau, bao gồm desktop, di động và web. Bằng cách sử dụng công cụ như Angular Universal, bạn có thể tạo ra các ứng dụng có khả năng SEO tốt và có thể chạy trên server-side.

+Cộng đồng phát triển và hỗ trợ: Angular tập trung vào việc xây dựng một cộng đồng phát triển mạnh mẽ và hỗ trợ. Cộng đồng này cung cấp tài liệu phong phú, hướng dẫn, ví dụ và thư viện bổ sung để giúp nhà phát triển nâng cao kỹ năng và giải quyết các vấn đề phát triển trong quá trình sử dụng Angular.

+Đúng thời gian thực và ứng dụng di động: Angular đặt mục tiêu phát triển các ứng dụng đúng thời gian thực (real-time) và ứng dụng di động. Với các tính năng như Observables, Web Workers và Service Workers, Angular cung cấp cơ sở để xây dựng các ứng dụng real-time và mobile-friendly.

Tóm lại, mục tiêu phát triển của Angular là tạo ra các ứng dụng web hiệu suất cao, linh hoạt và dễ bảo trì. Angular đặt mục tiêu hỗ trợ đa nền tảng, tạo điều kiện thuận lợi cho phát triển và mở rộng ứng dụng. Bằng cách tạo ra một cộng đồng phát triển và hỗ trợ mạnh mẽ, Angular giúp nhà phát triển nâng cao kỹ năng và tận dụng tối đa khả năng của framework.

Lịch sử:

-Dưới đây là một cái nhìn tổng quan về lịch sử của Angular:

+Năm 2010: AngularJS, phiên bản đầu tiên của Angular, được phát triển bởi Misko Hevery và Adam Abrons tại Google. Ban đầu, AngularJS được tạo ra để giúp xây dựng các ứng dụng web động một cách dễ dàng bằng cách mở rộng HTML và cung cấp các tính năng như two-way data binding và dependency injection.

+Năm 2012: AngularJS 1.0 được phát hành công khai và nhanh chóng thu hút sự quan tâm và sự ủng hộ từ cộng đồng phát triển. AngularJS trở thành một trong những framework phát triển ứng dụng web phía máy khách phổ biến nhất.

+Năm 2014: AngularJS 2.0 được công bố. Đây là một phiên bản đáng chú ý với nhiều thay đổi quan trọng. AngularJS 2.0 đã hoàn toàn viết lại nguyên bộ mã nguồn và mang lại một kiến trúc hoàn toàn mới, gọi là Angular.

+Năm 2016: Angular 2.0 được phát hành chính thức với một kiến trúc gói (component-based architecture) mạnh mẽ. Angular 2.0 đã đem lại nhiều cải tiến về hiệu suất, khả năng mở rộng và tính linh hoạt so với phiên bản trước.

+Năm 2017: Angular 4.0 được phát hành. Phiên bản này tập trung vào việc cải thiện hiệu suất và kích thước của ứng dụng, cũng như đưa ra nhiều tính năng mới và cải tiến.

+Năm 2018: Angular 6.0 được phát hành với nhiều tính năng mới như Angular Elements (cho phép bạn tạo các thành phần Angular có thể tái sử dụng trong các framework khác), Angular CLI và cải tiến hiệu suất.

+Năm 2019: Angular 8.0 được phát hành với các tính năng như Ivy Renderer (một bộ đoạn mã render mới giúp tăng hiệu suất ứng dụng), ngModule lười biếng (lazy loading) và nhiều cải tiến khác.

+Năm 2020: Angular 9.0 được phát hành, đồng thời đánh dấu sự thay đổi về quy ước đặt tên phiên bản của Angular. Các tính năng mới bao gồm Ivy Renderer mặc định, cải tiến hiệu suất và quản lý tài nguyên tốt hơn.

+Năm 2021: Angular 12.0 được phát hành. Phiên bản này tập trung vào cải tiến hiệu suất, cải thiện trải nghiệm phát triển và cung cấp các tính năng mới như strict mode, cập nhật Ivy, và các công cụ hỗ trợ phát triển khác.

Angular đã trải qua một quá trình phát triển và cải tiến liên tục để đáp ứng các yêu cầu và xu hướng phát triển mới nhất trong lĩnh vực phát triển ứng dụng web. Với mỗi phiên bản, Angular đã mang lại những cải tiến và tính năng mới để giúp nhà phát triển xây dựng các ứng dụng web hiệu suất cao, linh hoạt và dễ bảo trì.

2.2.5 MongoDB

Giới thiệu:

MongoDB là một hệ quản trị cơ sở dữ liệu phi quan hệ (NoSQL) mã nguồn mở, được phát triển bởi MongoDB Inc. MongoDB sử dụng mô hình lưu trữ tài liệu linh hoạt, cho phép lưu trữ và truy vấn dữ liệu một cách dễ dàng và hiệu quả.

MongoDB được xây dựng dựa trên kiến trúc client-server và lưu trữ dữ liệu dưới dạng JSON-like document (BSON). Với MongoDB, bạn có thể lưu trữ các tài liệu (documents) dưới dạng cặp key-value, giúp đơn giản hóa quá trình lưu trữ và truy vấn dữ liệu. Điều này giúp MongoDB trở thành một lựa chọn phổ biến cho các ứng dụng web và di động, cũng như các dự án phát triển nhanh.

Đặc điểm:

-Đặc điểm của MongoDB bao gồm:

+Phân tán và mở rộng: MongoDB được thiết kế để hoạt động trong môi trường phân tán và có khả năng mở rộng. Bạn có thể dễ dàng mở rộng hệ thống MongoDB bằng cách thêm các node mới vào cluster để tăng khả năng xử lý và dung lượng lưu trữ.

+Linh hoạt và dễ sử dụng: MongoDB cho phép bạn lưu trữ các tài liệu với cấu trúc động, không cần tuân thủ một schema cố định. Điều này giúp linh hoạt trong việc thay đổi cấu trúc dữ liệu và đáp ứng nhanh chóng với yêu cầu thay đổi. MongoDB cũng cung cấp ngôn ngữ truy vấn mạnh mẽ và dễ hiểu (MongoDB Query Language) để truy vấn dữ liệu.

+Tích hợp và khả năng mở rộng: MongoDB tích hợp tốt với các ngôn ngữ lập trình phổ biến như JavaScript, Python, Java và Node.js. Nó cũng hỗ trợ các tính năng mở rộng như replica set và sharding, cho phép bạn tăng tính sẵn sàng và hiệu suất của hệ thống.

+Hỗ trợ cho các tác vụ phức tạp: MongoDB cung cấp các tính năng và công cụ hỗ trợ cho các tác vụ phức tạp như indexing, aggregation framework, full-text search và map-reduce. Điều này giúp bạn truy xuất và phân tích dữ liệu một cách hiệu quả và linh hoạt.

+Hệ sinh thái phong phú: MongoDB có một hệ sinh thái phong phú với nhiều công cụ và thư viện hữu ích để hỗ trợ phát triển và quản lý cơ sở dữ liệu. Ví dụ, MongoDB Atlas cung cấp một dịch vụ quản lý cơ sở dữ liệu đám mây, trong khi MongoDB Compass cung cấp một giao diện đồ họa dễ sử dụng để trực quan hóa và quản lý dữ liệu.

Tóm lại, MongoDB là một hệ quản trị cơ sở dữ liệu phi quan hệ linh hoạt và mở rộng. Với tính linh hoạt, khả năng mở rộng và các tính năng mạnh mẽ, MongoDB là một lựa chọn phổ biến cho việc lưu trữ và truy vấn dữ liệu cho các ứng dụng web và di động.

Mục tiêu phát triển:

-Mục tiêu phát triển của MongoDB bao gồm:

+Đơn giản và dễ sử dụng: MongoDB đặt mục tiêu tạo ra một hệ quản trị cơ sở dữ liệu dễ sử dụng, giúp nhà phát triển nhanh chóng làm quen và triển khai các ứng dụng. Giao diện và cú pháp dễ hiểu của MongoDB giúp giảm thời gian và công sức trong việc phát triển và quản lý cơ sở dữ liệu.

+Linh hoạt và mở rộng: MongoDB tập trung vào tính linh hoạt và khả năng mở rộng. Với cấu trúc tài liệu động và không yêu cầu một schema cố định, MongoDB cho phép bạn linh hoạt thay đổi cấu trúc dữ liệu mà không cần phải thay đổi schema. Điều này giúp giải quyết các yêu cầu thay đổi và phát triển ứng dụng một cách linh hoạt. Ngoài ra, MongoDB hỗ trợ khả năng mở rộng dễ dàng bằng cách thêm node vào cluster để tăng hiệu suất và dung lượng lưu trữ.

+Hiệu suất cao: MongoDB tập trung vào việc cung cấp hiệu suất cao cho các ứng dụng có khối lượng lớn dữ liệu và tải trọng cao. Với tính năng như indexing, caching và cơ chế truy vấn thông minh, MongoDB cung cấp tốc độ truy vấn và xử lý dữ liệu nhanh chóng. Điều này giúp tăng khả năng đáp ứng của ứng dụng và cung cấp trải nghiệm tốt hơn cho người dùng.

+Độ tin cậy và sẵn sàng cao: MongoDB đặt mục tiêu cung cấp độ tin cậy và sẵn sàng cao cho cơ sở dữ liệu của bạn. Tính năng replica set của MongoDB cho phép sao chép dữ liệu và xác định các bản sao dự phòng, đảm bảo rằng dữ liệu không bị mất đi trong trường hợp một node gặp sự cố. Ngoài ra, các tính năng khác như auto-sharding và failover cung cấp khả năng chịu lỗi và sẵn sàng liên tục của hệ thống.

+Hỗ trợ toàn diện: MongoDB đặt mục tiêu cung cấp một hệ sinh thái phong phú với các công cụ và tài liệu hữu ích để hỗ trợ phát triển và quản lý cơ sở dữ liệu. Có sự hỗ trợ từ MongoDB Atlas (dịch vụ quản lý cơ sở dữ liệu đám mây), MongoDB Compass (giao diện đồ họa), và cộng đồng người dùng rộng lớn, nhà phát triển có tài liệu và tài nguyên phong phú để xây dựng và duy trì các ứng dụng MongoDB.

Tóm lại, mục tiêu phát triển của MongoDB là cung cấp một hệ quản trị cơ sở dữ liệu đơn giản, linh hoạt và hiệu suất cao. MongoDB tập trung vào tính linh hoạt, khả năng mở rộng, hiệu suất cao và sẵn sàng của hệ thống. Với hỗ trợ toàn diện và sự tiếp tục phát triển từ cộng đồng, MongoDB đáp ứng nhu cầu của các ứng dụng web và di động đòi hỏi hiệu suất và linh hoạt.

Lịch sử:

-Dưới đây là một cái nhìn tổng quan về lịch sử của MongoDB:

+Năm 2007: MongoDB được sáng lập bởi Dwight Merriman và Eliot Horowitz tại một công ty khởi nghiệp tên là 10gen (tên sau này đã được đổi thành MongoDB Inc.). Merriman và Horowitz nhận thấy rằng hệ quản trị cơ sở dữ liệu quan hệ không đáp ứng đầy đủ nhu cầu của các ứng dụng web đang phát triển, vì vậy họ quyết định tạo ra một giải pháp mới.

+Năm 2009: Phiên bản đầu tiên của MongoDB (MongoDB 1.0) được phát hành công khai. MongoDB sử dụng mô hình lưu trữ tài liệu (document-oriented) và cung cấp một cú pháp truy vấn linh hoạt và mạnh mẽ dựa trên JSON-like document (BSON).

+Năm 2010: MongoDB đạt được sự chú ý lớn từ cộng đồng phát triển và người dùng. Sự phổ biến của MongoDB tăng lên khi các công ty công nghệ hàng đầu như Foursquare, SourceForge và The Guardian bắt đầu sử dụng nó cho các ứng dụng của mình.

+Năm 2013: Phiên bản MongoDB 2.4 được phát hành, đem lại nhiều tính năng mới và cải tiến quan trọng. Một trong những tính năng đáng chú ý là tệp nhị phân lớn (GridFS) cho phép lưu trữ và truy xuất các tệp tin lớn trong MongoDB.

+Năm 2014: MongoDB 2.6 được phát hành, đồng thời đánh dấu sự thay đổi trong cấu trúc lưu trữ và quản lý của MongoDB. Phiên bản này giới thiệu một bộ điều khiển WiredTiger mới, cung cấp khả năng nén dữ liệu, xử lý đồng thời và hiệu suất cao hơn.

+Năm 2015: MongoDB 3.0 được phát hành, mang đến nhiều tính năng mới và cải tiến. Phiên bản này giới thiệu một phiên bản mới của bộ điều khiển WiredTiger, index text, tính năng quản lý tài nguyên tốt hơn và tính năng quản lý quyền truy cập.

+Năm 2017: Phiên bản MongoDB 3.4 được phát hành với nhiều tính năng mới như bộ điều khiển vị trí (geospatial), tìm kiếm toàn văn bản (text search) và tích hợp với Apache Kafka.

+Năm 2018: MongoDB 4.0 được phát hành, đem lại nhiều cải tiến hiệu suất, bao gồm việc cải thiện khả năng mở rộng và tăng cường tính năng truy vấn. Phiên bản này giới thiệu tính năng multi-document ACID (Atomicity, Consistency, Isolation, Durability) và transactions.

+Năm 2019: MongoDB 4.2 được phát hành với các tính năng như distributed transactions, on-demand materialized views và tính năng truy vấn tương tự SQL.

+Năm 2020: MongoDB 4.4 được phát hành, đem lại nhiều tính năng mạnh mẽ như realm-sync, encrypted field-level access control và nhiều cải tiến về hiệu suất và bảo mật.

-MongoDB tiếp tục phát triển và cải tiến theo thời gian để đáp ứng các yêu cầu ngày càng phức tạp của các ứng dụng web và di động. Với sự hỗ trợ từ cộng đồng phát triển và sự tiếp tục của MongoDB Inc., MongoDB trở thành một trong những hệ quản trị cơ sở dữ liệu phi quan hệ hàng đầu được sử dụng rộng rãi trên toàn thế giới.

2.2.6 Công cụ lập trình VISUAL STUDIO CODE

CHƯƠNG 3 NỘI DUNG VÀ PHƯƠNG PHÁP NGHIÊN CỨU

3.1 CÔNG CỤ PHÁT TRIỂN

3.2 PHÂN TÍCH THIẾT KẾ HỆ THỐNG

3.2.1 Mô hình tổng quan hệ thống

Hệ thống bao gồm các chức năng như:  
+ Admin: đăng nhập(đăng ký, đăng xuất, đổi mật khẩu), quản lý thông tin bác sĩ trong phòng khám, quản lý thông tin lịch khám, quả lý thông tin người dùng

+ Người dùng: đăng nhập (đăng ký, đăng xuất, đổi mật khẩu),đăng ký lịch khám chửa bệnh, xem thông tin tổng quan về phòng khám ( giới thiệu, chuyên khoa, liên hệ,…)

3.2.2 Phân tích và thiết kế hệ thống

Các tác vụ của hệ thống:

Hệ thống quản trị:

Quản lý thông tin người dùng

Quản lý lịch khám chữa bệnh

Quản lý t hông tin bác sĩ

Hệ thống đặt lịch:

Hiển thị pop-up đặt lịch khám

Hiển thị thông tin ngày giờ đăng ký khám chữa bệnh

Các tác nhân hệ thống:

Theo thông tin mô tả có các tác nhân hệ thống chính như sau:

Tác nhân Quản lý: người thực hiện chức năng quản lý hệ thống chính , bao gồm người dùng, bác sĩ và lịch khám

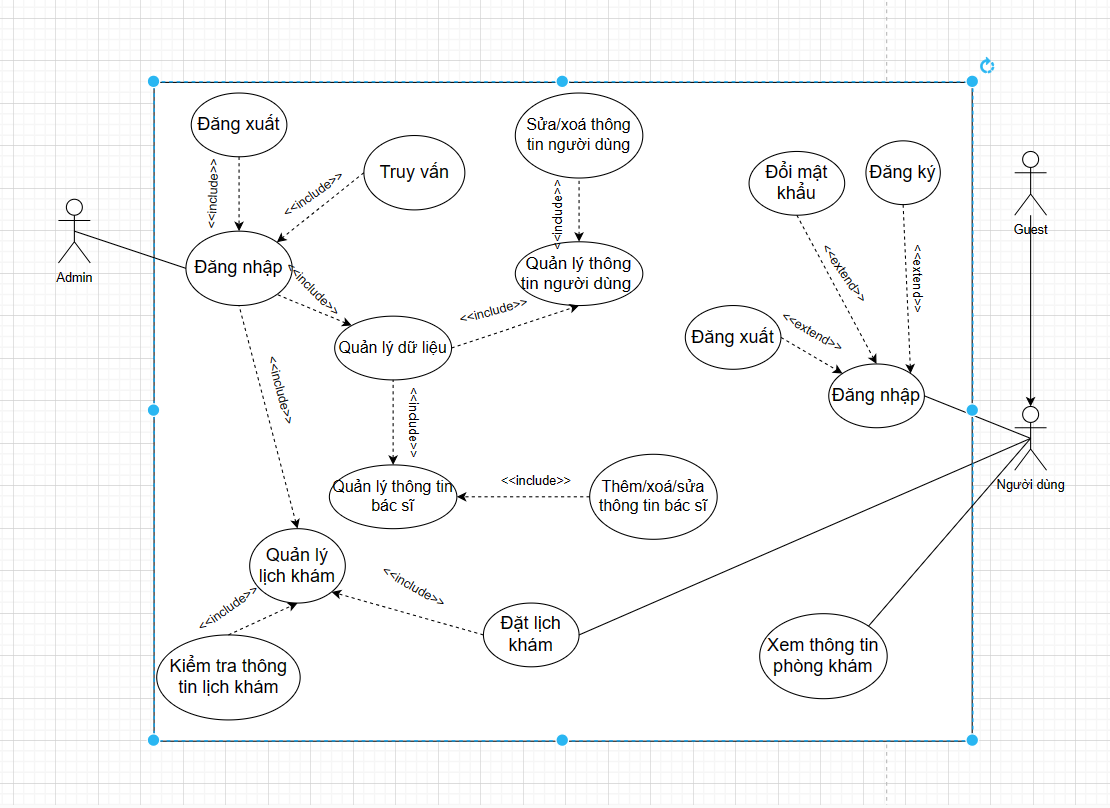
Tác nhân Người dùng: lịch khám, xem thông tin

Bảng chức năng của quản lý

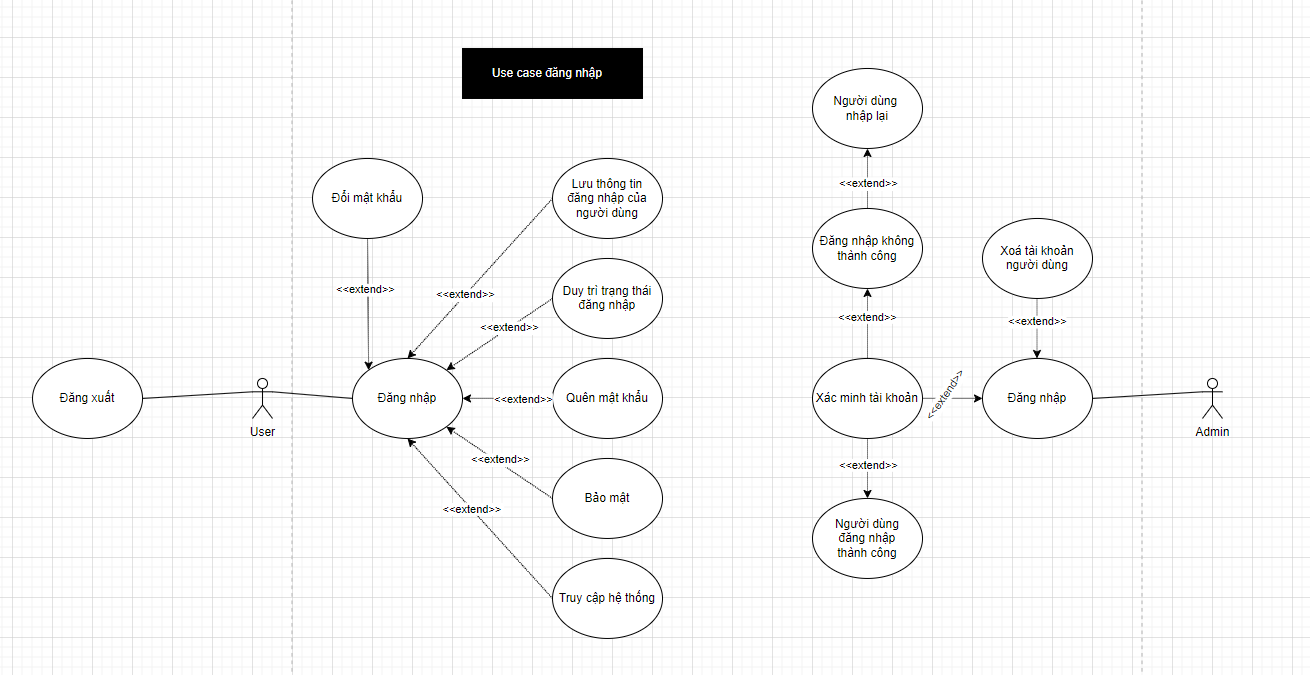
Bảng chức năng của nhân viên

3.2.3 Biểu đồ usecase

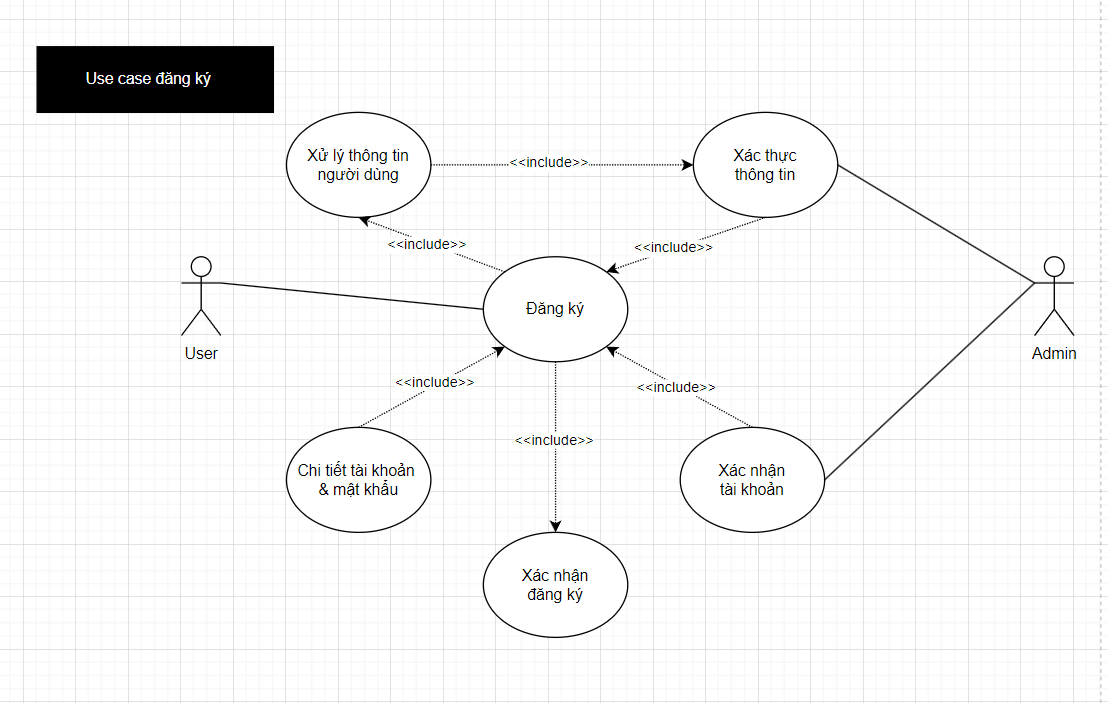
- Biểu đồ use case tổng quát :



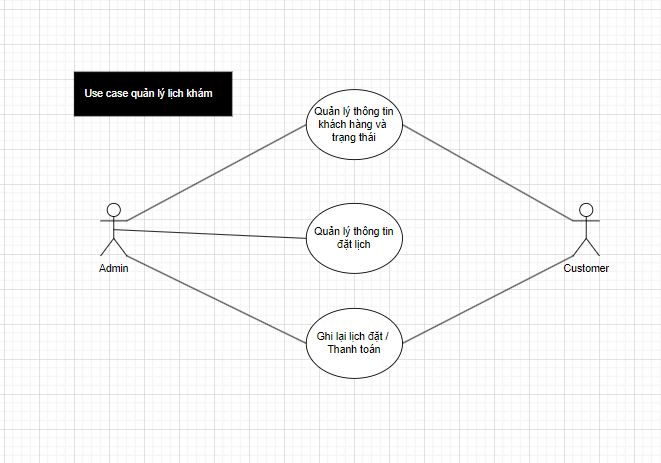
- Biểu đồ usecase đăng nhập/đăng xuất:



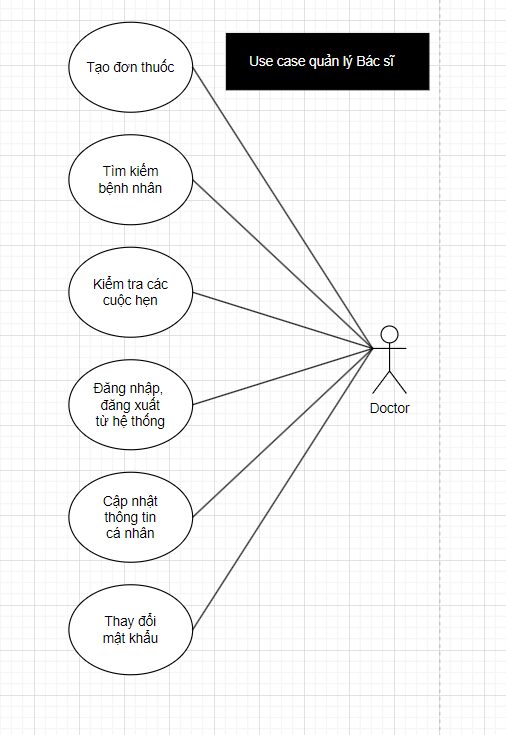
- Biểu đồ usecase đăng ký:



- Biểu đồ usecase quản lý lịch khám:



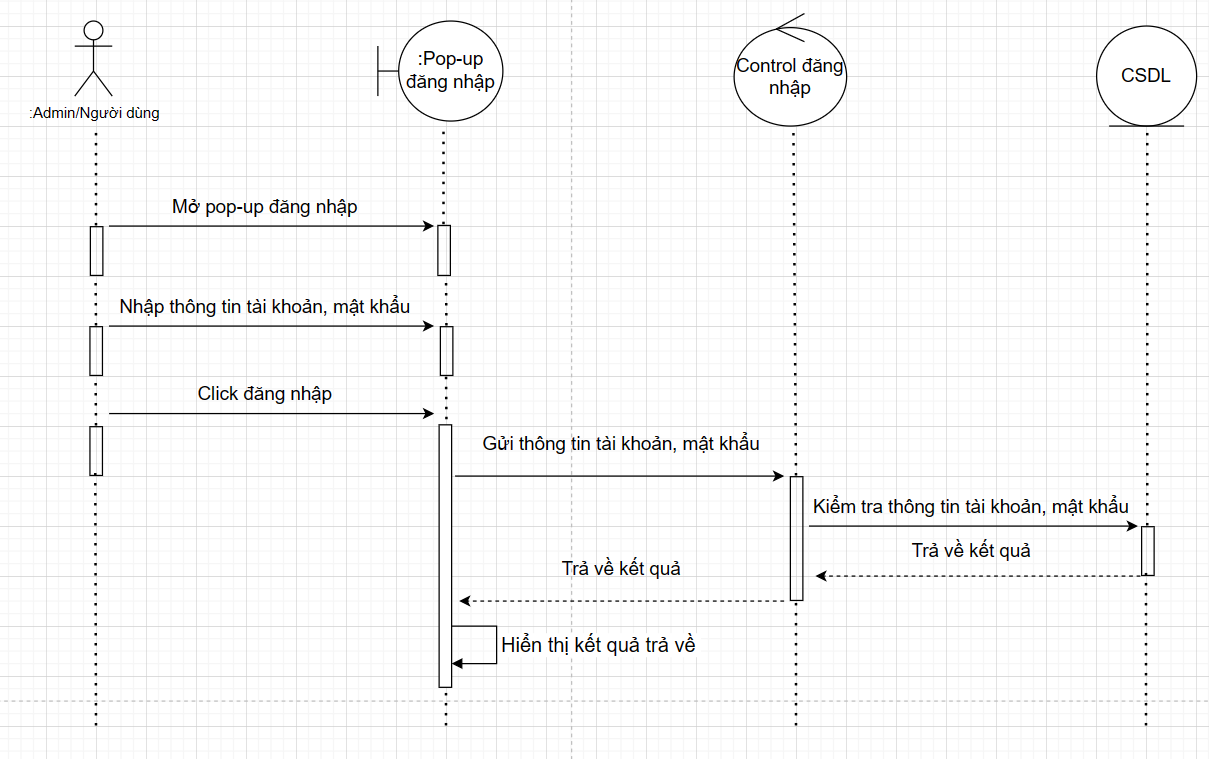
- Biểu đồ quản lý bác sĩ:



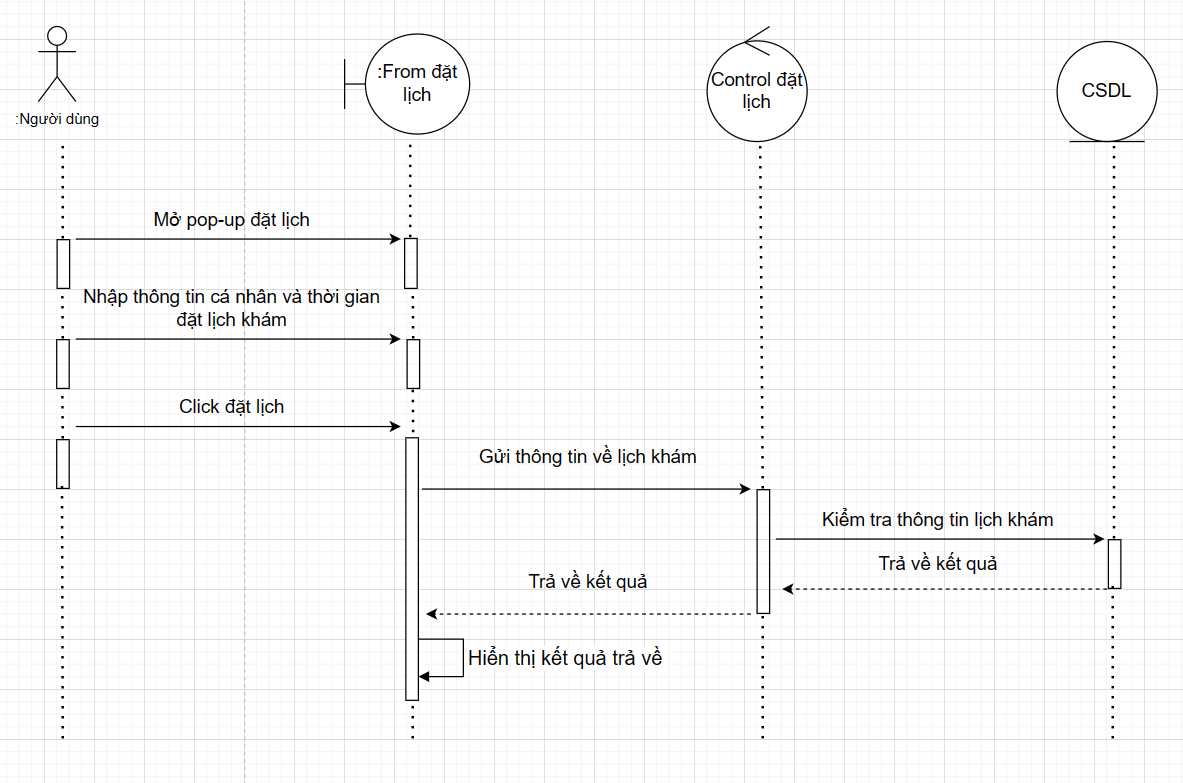
- Biểu đồ quản lý thông tin người dùng

3.2.3 Biểu đồ tuần tụ

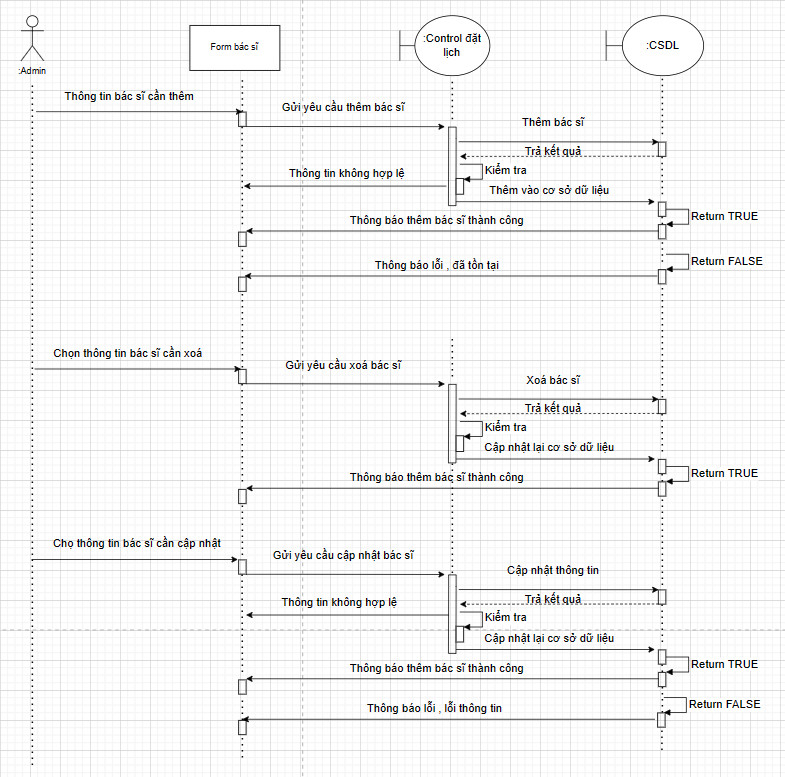
-Biểu đồ tuần tự đăng nhập



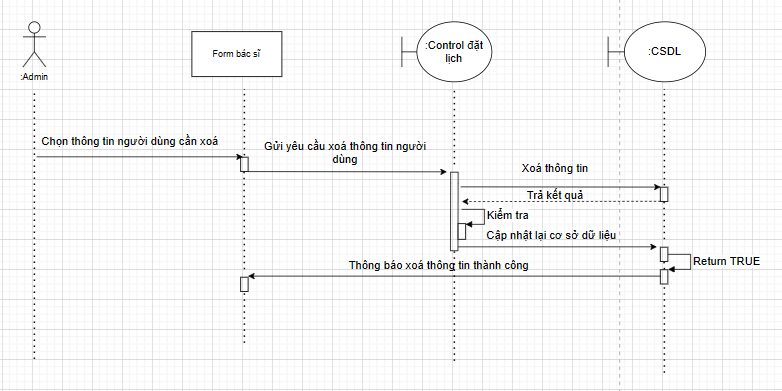
- Biểu đồ tuần tự đặt lịch khám

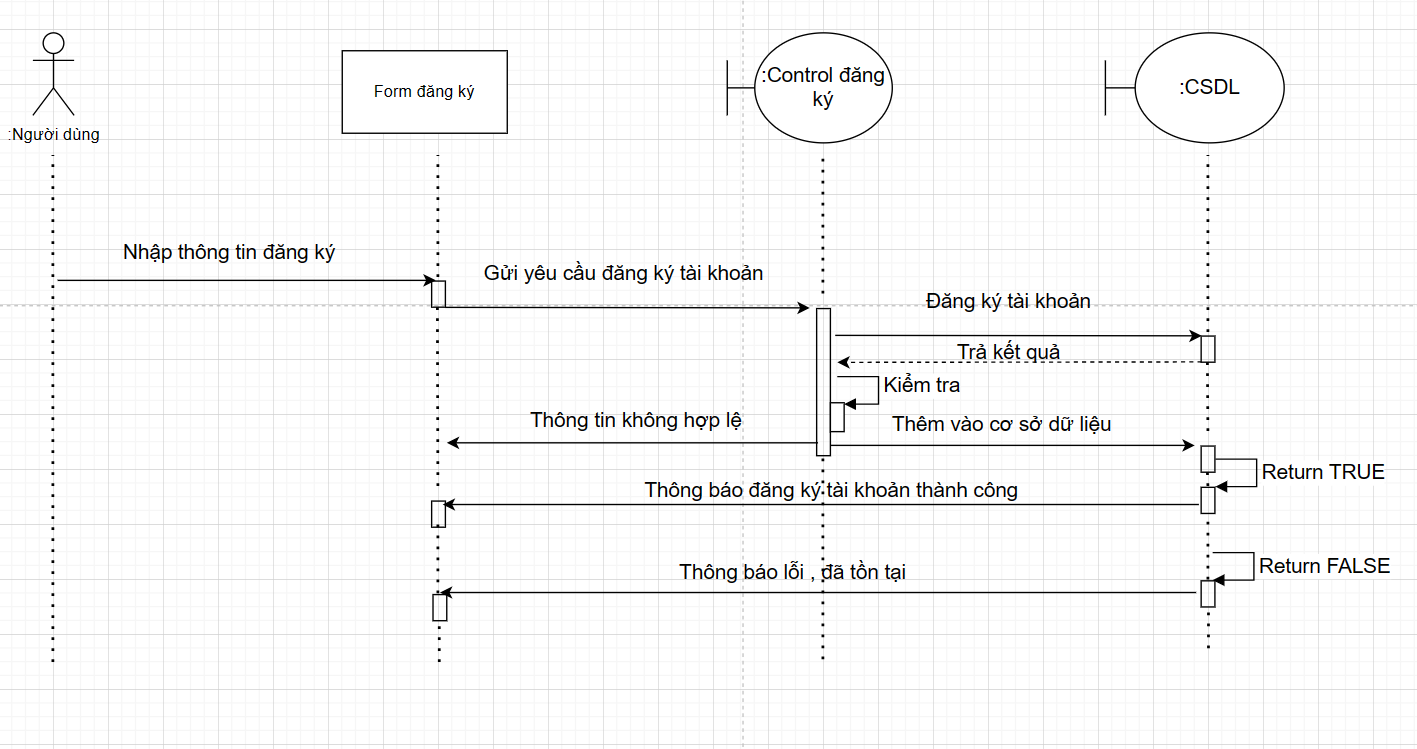


- Biểu đồ quản lý thông tin bác sĩ



- Biểu đồ tuần tự thông tin người dùng





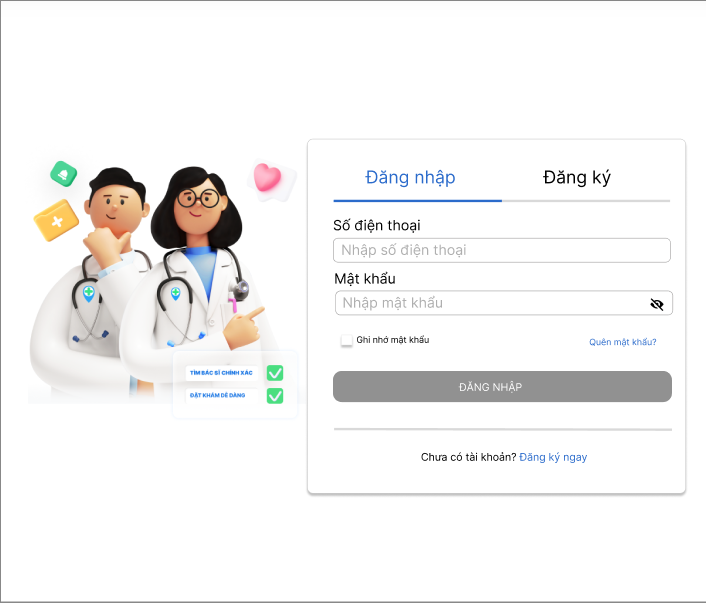
3.3 THIẾT KẾ GIAO DIỆN

Giao diện website đáp ứng được nhu cầu chức năng đặt lịch khám bệnh của người dùng. Giúp phòng khám quản lý được thông tin về các người dùng có nhu cầu đặt trước thời gian khám chữa bệnh, các thông tin về phòng khám như bác sĩ, lịch hẹn,..

Xem thông tin giao diện tại link figma: [KHOÁ LUẬN TỐT NGHIỆP](https://www.figma.com/file/CH8csfTGsA9x8Fh1ozvtrn/%C4%90%E1%BB%92-%C3%81N-T%E1%BB%90T-NGHI%E1%BB%86P?type=design&mode=design)

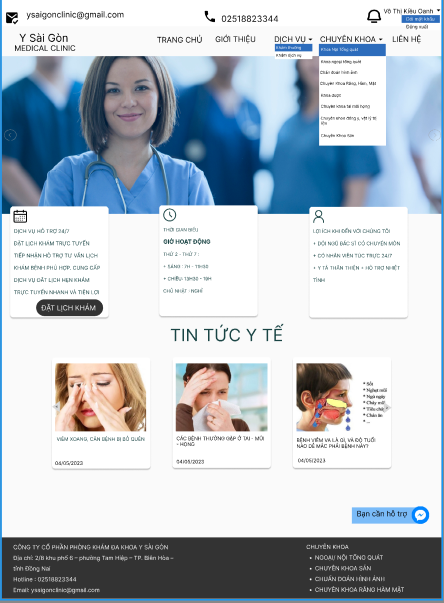
Giao diện các chức năng

-Tác vụ đăng nhập:



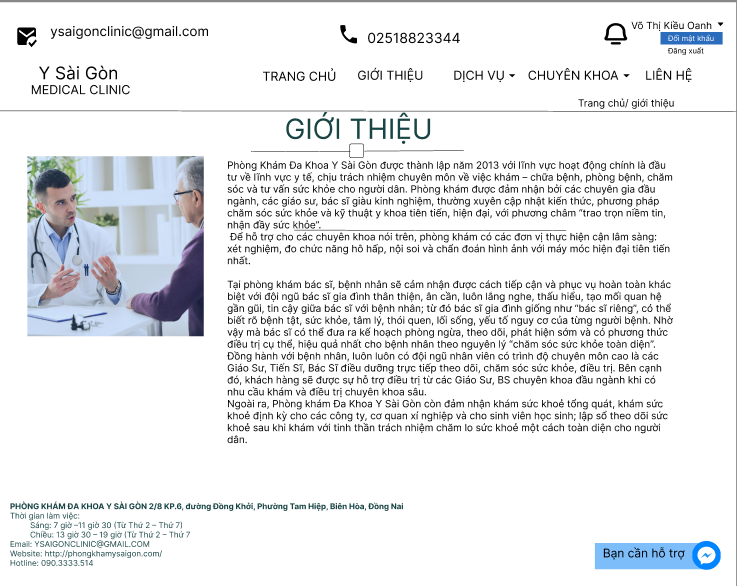
Website bao gồm người quản trị và bệnh nhân , mỗi người đều có tài khoản riêng biệt và mật khẩu riêng khác nhau.

-Trang chủ của website:



Trang chủ bao gồm các thông tin, hình ảnh mô tả sơ lược địa chỉ cũng như tin tức y tế về phòng khám.Có hỗ trợ trực tuyến thông qua tin nhắn

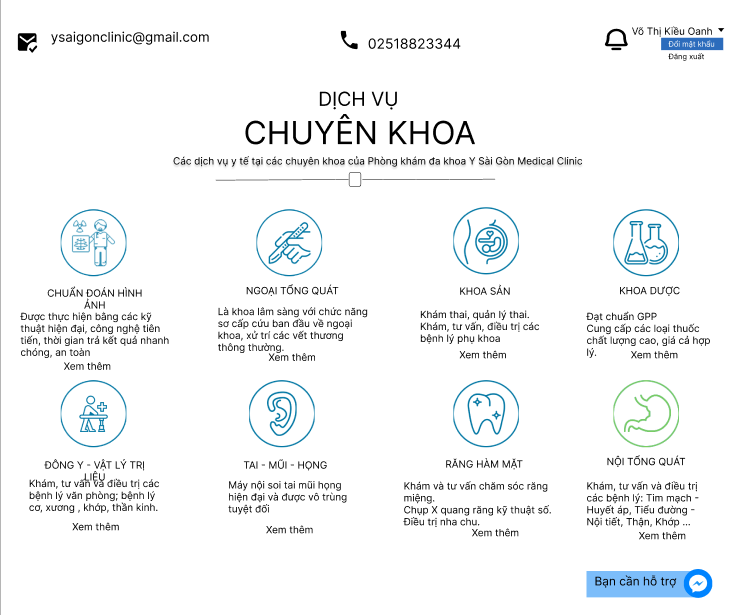
-Trang giới thiệu:



Trong trang này người dùng có thể thấy các thông tin cụ thể liên quan đến phòng khám cũng như địa chỉ, giờ làm việc và thông tin mạng xã hội

-Trang chuyên khoa:

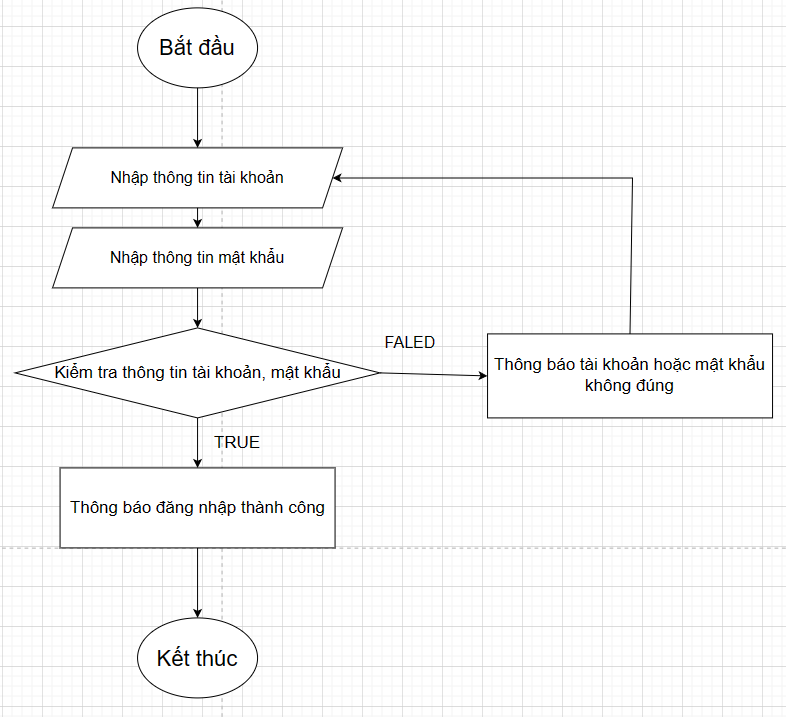
Mô tả tất cả các dịch vụ chuyên khoa đang có tại phòng khám

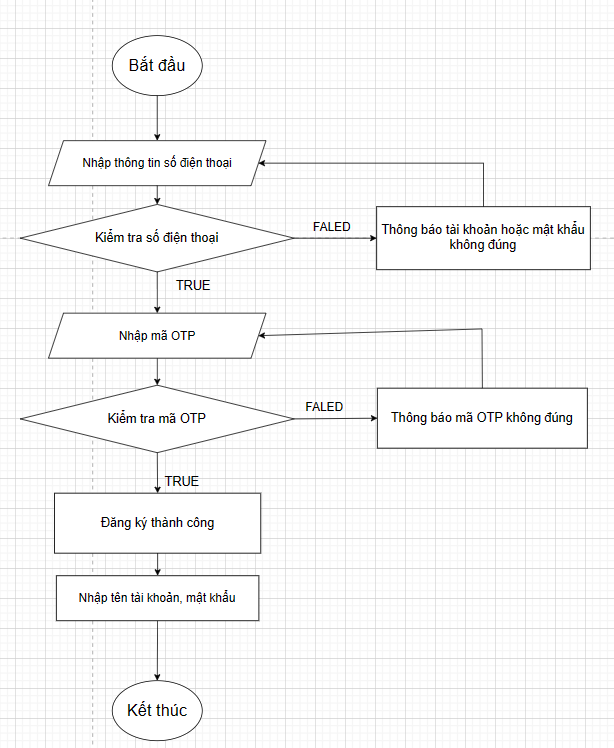


3.4 THUẬT TOÁN THỰC HIỆN CÁC CHỨC NĂNG ỨNG DỤNG

3.4.1 Lưu đồ thuật toán thực hiện các chức năng chính

Lưu đồ thuật toán đăng nhập



Lưu đồ thuật toán đăng ký   


CHƯƠNG 4 KẾT QUẢ, NHẬN XÉT VÀ KẾT LUẬN