**ĐẠI HỌC ĐÀ NẴNG**

**TRƯỜNG ĐẠI HỌC SƯ PHẠM**

**KHOA TIN HỌC**

****

**BÁO CÁO PHÂN TÍCH VÀ THIẾT KẾ HƯỚNG ĐỐI TƯỢNG**

**ĐỀ TÀI:**

**PHÂN TÍCH VÀ THIẾT KẾ**

**HỆ THỐNG ĐẶT LỊCH KHÁM SỨC KHỎE**

**TẠI PHÒNG KHÁM TOÀN MỸ**

|  |  |
| --- | --- |
| **Nhóm 2** | |
| Sinh viên thực hiện: | 1. Phạm Thị Như Thủy\_19CNTTD |
| 2. Trần Thị Thảo Nguyên\_19CNTTD |
| 3. Trần Hữu Thọ\_19CNTTD |
| 4. Nguyễn Văn Đạt\_19CNTTD |
| 5. Phan Công Tài\_17CNTT2 |
| Giảng viên hướng dẫn | Thầy Nguyễn Văn Vương |

Đà Nẵng – 01/2023

MỤC LỤC

[MỤC LỤC 2](#_Toc123934464)

[DANH MỤC CÁC HÌNH 5](#_Toc123934465)

[MỞ ĐẦU 6](#_Toc123934466)

[CHƯƠNG I: CƠ SỞ LÝ THUYẾT 8](#_Toc123934467)

[1. TÌM HIỂU CÁC NGÔN NGỮ 8](#_Toc123934468)

[1.1. Ngôn ngữ UML 8](#_Toc123934469)

[1.1.1. Khái niệm 8](#_Toc123934470)

[1.1.2. Mô hình 9](#_Toc123934471)

[1.1.2.1. Phần tử trong UML 9](#_Toc123934472)

[1.1.2.2. Các quan hệ trong UML 12](#_Toc123934473)

[1.1.2.3. Kiểu dữ liệu 13](#_Toc123934474)

[1.1.2.4. Biểu đồ UML 13](#_Toc123934475)

[1.2. Ngôn Ngữ HTML 14](#_Toc123934476)

[1.2.1. Khái niệm HTML 14](#_Toc123934477)

[1.2.2. Cấu trúc của HTML 15](#_Toc123934478)

[1.3. Ngôn ngữ CSS 17](#_Toc123934479)

[1.3.1. Khái niệm CSS 17](#_Toc123934480)

[1.3.2. Cú pháp của CSS 18](#_Toc123934481)

[1.4. Mô hình MVC 19](#_Toc123934482)

[1.5. Mô hình MVVM (Model – View – View - Model) 20](#_Toc123934483)

[2. KIẾN TRÚC HỆ THỐNG C4 Model 21](#_Toc123934484)

[2.1. C4 model 21](#_Toc123934485)

[2.2. Ký hiệu 25](#_Toc123934486)

[3. MỘT SỐ THƯ VIỆN, FRAMEWORK HỖ TRỢ 26](#_Toc123934487)

[3.1. Thư viện Javascript 26](#_Toc123934488)

[3.1.1. Khái niệm Javascript 26](#_Toc123934489)

[3.1.2. Cách hoạt động của Javascript 27](#_Toc123934490)

[3.1.3. Cấu trúc của Javascript 27](#_Toc123934491)

[3.1.4. Javascript dùng làm gì 29](#_Toc123934492)

[3.1.5. Lợi ích khi dùng Javascript 29](#_Toc123934493)

[3.1.6. Ưu điểm và nhược điểm của Javascript 30](#_Toc123934494)

[3.2. Thư viện Jquery 31](#_Toc123934495)

[3.2.1. Khái niệm Jquery 31](#_Toc123934496)

[3.2.2. Các tính năng của Jquery 31](#_Toc123934497)

[3.2.3. Cài đặt Jquery vào Website 32](#_Toc123934498)

[3.2.4. Cú pháp 32](#_Toc123934499)

[3.3. Thư viện ReactJs 33](#_Toc123934500)

[3.3.1. Giới thiệu về ReactJs 33](#_Toc123934501)

[3.3.2. Ưu và nhược điểm 34](#_Toc123934502)

[3.4. NodeJs 35](#_Toc123934503)

[4. API 36](#_Toc123934504)

[5. CƠ SỞ DỮ LIỆU MYSQL 37](#_Toc123934505)

[5.1. Khái niệm cơ sở dữ liệu mysql 37](#_Toc123934506)

[5.2. Cơ chế hoạt động của MySql 38](#_Toc123934507)

[5.3. Ưu điểm và nhược điểm của Mysql 38](#_Toc123934508)

[6. CÔNG CỤ HỖ TRỢ LẬP TRÌNH 40](#_Toc123934509)

[6.1. Visual Studio Code 40](#_Toc123934510)

[6.1.1. Visual Studio Code là gì? 40](#_Toc123934511)

[6.1.2. Ưu điểm của Visual Studio Code 40](#_Toc123934512)

[6.1.3. Cài đặt Visual Studio Code 40](#_Toc123934513)

[6.2. Visual Studio Current 43](#_Toc123934514)

[6.2.1. Visual Studio Current là gì 43](#_Toc123934515)

[6.2.2. Ưu điểm khi sử dụng Visual Studio Current 43](#_Toc123934516)

[6.2.3. Hướng dẫn cài đặt 43](#_Toc123934517)

[CHƯƠNG II: PHÂN TÍCH THIẾT KẾ HỆ THỐNG WEBSITE 47](#_Toc123934518)

[1. MÔ TẢ BÀI TOÁN, KHẢO SÁT, ĐƯA RA YÊU CẦU 47](#_Toc123934519)

[1.1. Mô tả bài toán 47](#_Toc123934520)

[1.2. Khảo sát yêu cầu 47](#_Toc123934521)

[1.3. Yêu cầu bài toán 51](#_Toc123934522)

[1.3.1. Đối với bác sĩ 51](#_Toc123934523)

[1.3.2. Đối với người dùng của trang web 51](#_Toc123934524)

[1.3.3. Đối với người quản trị trang web (admin) 52](#_Toc123934525)

[2. PHÂN TÍCH HỆ THỐNG 52](#_Toc123934526)

[2.1. Mô tả hệ thống 52](#_Toc123934527)

[2.2. Kết quả Phân tích hệ thống 53](#_Toc123934528)

[3. THIẾT KẾ HỆ THỐNG( SỬ DỤNG C4 MODEL) 54](#_Toc123934529)

[3.1. Context diagram 54](#_Toc123934530)

[3.2. Container diagram 55](#_Toc123934531)

[3.3. Component diagram 56](#_Toc123934532)

[CHƯƠNG III: XÂY DỰNG HỆ THỐNG WEBSITE 64](#_Toc123934533)

[1. GIAO DIỆN HIỂN THỊ 64](#_Toc123934534)

[1.1. Giao diện trang chủ 64](#_Toc123934535)

[1.2. Giao diện thông tin bác sĩ 64](#_Toc123934536)

[1.3. Giao diện quản lý kế hoạch khám chữa bệnh của bác sĩ 66](#_Toc123934537)

[1.4. Giao diện thông tin đặt lịch khám bệnh 66](#_Toc123934538)

[1.5. Giao diện Login 67](#_Toc123934539)

[2. GIAO DIỆN QUẢN LÝ 67](#_Toc123934540)

[2.1. Giao diện quản lý thông tin người dùng 67](#_Toc123934541)

[2.2. Giao diện tạo mới thông tin bác sĩ 68](#_Toc123934542)

[2.3. Giao diện cập nhật thông tin tài khoản bác sĩ 68](#_Toc123934543)

[2.4. Giao diện chỉnh sửa thông tin tài khoản người dùng 69](#_Toc123934544)

[2.5. Giao diện xóa thông tin tài khoản người dùng 69](#_Toc123934545)

[KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN 70](#_Toc123934546)

[TÀI LIỆU THAM KHẢO 71](#_Toc123934547)

DANH MỤC CÁC HÌNH

[Hình 1.1: HTML 15](#_Toc123934366)

[Hình 1.2: CSS 18](#_Toc123934367)

[Hình 1.3: Mô hình MVC 20](#_Toc123934368)

[Hình 1.4: Mô hình MVVM 21](#_Toc123934369)

[Hình 1.5: C4 Model 25](#_Toc123934370)

[Hình 1.6: Javascript 27](#_Toc123934371)

[Hình 1.7: Jquery 31](#_Toc123934372)

[Hình 1.8: MySQL 38](#_Toc123934373)

[Hình 1.9: Visual Studio Code 40](#_Toc123934374)

[Hình 2.1: Chức năng của hệ thống 53](#_Toc123934380)

[Hình 2.2: Mô hình ER 53](#_Toc123934381)

[Hình 2.3:[System Context] Booking System 54](#_Toc123934382)

[Hình 2.4: [Container] Booking System 55](#_Toc123934383)

[Hình 2.5: [Component] Booking system - Web Application 56](#_Toc123934384)

[Hình 2.6: [Conponent] Booking system - SinglePageApplication 57](#_Toc123934385)

[Hình 2.7: [Conponent] Booking system – API Application 57](#_Toc123934386)

[Hình 2.8: [Conponent] Booking system – API Application (#SignIn) 58](#_Toc123934387)

[Hình 2.9: [Conponent] Booking system – API Application (#DoctorsInformation) 59](#_Toc123934388)

[Hình 2.10: [Conponent] Booking system – API Application (#ManagementAccount) 60](#_Toc123934389)

[Hình 2.11: [Conponent] Booking system – API Application (#UpdateDoctorInformation) 61](#_Toc123934390)

[Hình 2.12: [Conponent] Booking system – API Application (#AddAccount) 62](#_Toc123934391)

[Hình 2.13: [Conponent] Booking system – API Application (#Booking) 63](#_Toc123934392)

[Hình 3.1: Giao diện trang chủ 64](#_Toc123934397)

[Hình 3.2: Giao diện thông tin bác sĩ 65](#_Toc123934398)

[Hình 3.3: Giao diện quản lý kế hoạch khám chữa bệnh của bác sĩ 66](#_Toc123934399)

[Hình 3.4: Giao diện thông tin đặt lịch khám bệnh 66](#_Toc123934400)

[Hình 3.5: Giao diện Login 67](#_Toc123934401)

[Hình 3.6: Giao diện quản lý thông tin bác sĩ 67](#_Toc123934402)

[Hình 3.7: Giao diện tạo mới thông tin bác sĩ 68](#_Toc123934403)

[Hình 3.8: Giao diện chỉnh sửa thông tin tài khoản người dùng 68](#_Toc123934404)

[Hình 3.9: Giao diện chỉnh sửa thông tin tài khoản người dùng 69](#_Toc123934405)

[Hình 3.10: Giao diện xóa thông tin tài khoản người dùng 69](#_Toc123934406)

MỞ ĐẦU

**1. Lý do chọn đề tài**

Với sự phát triển nhảy vọt của công nghệ thông tin hiện nay, Internet ngày càng giữ vai trò quan trọng trong các lĩnh vực khoa học kĩ thuật và đời sống. Dĩ nhiên các bạn đã được nghe nói nhiều về Internet, nói một cách đơn giản, Internet là một tập hợp máy tính nối kết với nhau, là một mạng máy tính toàn cầu mà bất kì ai cũng có thể kết nối bằng máy PC của họ. Với mạng Internet, tin học thật sự tạo nên một cuộc cách mạng trao đổi thông tin trong mọi lĩnh vực văn hóa, xã hội, chính trị, kinh tế...

Trong thời đại ngày nay, thời đại mà “người người làm Web, nhà nhà làm Web” thì việc có một Website để quảng bá công ty hay một Website cá nhân không còn là điều gì xa xỉ nữa. Thông qua Website khách hàng có thể lựa chọn những sản phẩm mà mình cần một cách nhanh chóng và hiệu quả.

**2. Mục tiêu của đề tài**

Xem thông tin chi tiết về thông tin bác sĩ, lịch khám

Đặt lịch khi đã tìm kiếm được bác sĩ, thời gian phù hợp

Website hỗ trợ khách hàng đăng ký, đăng nhập, đặt lịch, cập nhật thông tin về những thời gian một cách nhanh nhất, đánh giá cũng như bình luận dịch vụ đã sử dụng, ... đem lại sự hài lòng cao nhất từ khách hàng

**3. Đối tượng và phạm vi thực hiện của đề tài**

- Đối tượng: Web đặt lịch khám sức khỏe.

- Phạm vi: UML,HTML, CSS, JAVASCRIPT, NODEJS, REACTJS, BOOSTRAP, …

**4. Phương pháp thực hiện**

Tiến hành thu thập và phân tích những thông tin, tài liệu liên quan đến đề tài để hình thành những ý tưởng tổng quan đưa ra những mục đích cần đạt được từ website, biết được đối tượng hướng đến là ai. Sử dụng những thông tin hiện có như thế nào

Xác định được yêu cầu phân tích thiết kế hệ thống chương trình để đưa ra cho phù hợp

Xây dựng chương trình theo mô hình yêu cầu đã đề ra

Triển khai chương trình và đánh giá kết quả đạt được

**5. Bố cục của đề tài**

Đề tài được phân thành 3 chương với cấu trúc như sau:

**Chương 1: Cơ sở lý thuyết**

**Chương 2: Phân tích và thiết kế hệ thống website**

**Chương 3: Xây dựng hệ thống website**

CHƯƠNG I: CƠ SỞ LÝ THUYẾT

* 1. TÌM HIỂU CÁC NGÔN NGỮ
     1. Ngôn ngữ UML

UML là ngôn ngữ mô hình hóa, trước hết nó là mô tả ký pháp thống nhất, ngữ nghĩa và các định nghĩa về metamodel (mô tả định nghĩa chính ngôn ngữ mô hình hóa), nó không mô tả về phương pháp phát triển. UML được sử dụng để hiển thị, đặc tả, xây dựng và làm tài liệu các vật phẩm (artifacts) của phân tích quá trình xây dựng hệ thống phần mềm theo hướng đối tượng. UML được sử dụng cho mọi tiến trình phát triển phần mềm, xuyên suốt vòng đời phát triển và độc lập với các công nghệ cài đặt hệ thống.

* + - 1. Khái niệm

Ngôn ngữ mô hình hóa thống nhất (tiếng Anh: Unified Modeling Language, viết tắt thành UML) là một ngôn ngữ mô hình gồm các ký hiệu đồ họa mà các phương pháp hướng đối tượng sử dụng để thiết kế các hệ thống thông tin một cách nhanh chóng.

Cách xây dựng các mô hình trong UML phù hợp mô tả các hệ thống thông tin cả về cấu trúc cũng như hoạt động. Cách tiếp cận theo mô hình của UML giúp ích rất nhiều cho những người thiết kế và thực hiện hệ thống thông tin cũng như những người sử dụng nó; tạo nên một cái nhìn bao quát và đầy đủ về hệ thống thông tin dự định xây dựng. Cách nhìn bao quát này giúp nắm bắt trọn vẹn các yêu cầu của người dùng; phục vụ từ giai đoạn phân tích đến việc thiết kế, thẩm định và kiểm tra sản phẩm ứng dụng công nghệ thông tin. Các mô hình hướng đối tượng được lập cũng là cơ sở cho việc ứng dụng các chương trình tự động sinh mã trong các ngôn ngữ lập trình hướng đối tượng, chẳng hạn như ngôn ngữ C++, Java,... Phương pháp mô hình này rất hữu dụng trong lập trình hướng đối tượng. Các mô hình được sử dụng bao gồm Mô hình đối tượng (mô hình tĩnh) và Mô hình động.

* + - 1. Mô hình
      2. Phần tử trong UML

Các khối hình thành mô hình UML gồm ba loại như sau: phần từ, quan hệ và biểu đồ. Phần tử là trừu tượng căn bản trong mô hình, các quan hệ gắn các phần tử này lại với nhau; còn biểu đò nhóm tập hợp các phần tử. T

rong UML có bốn loại phần tử mô hình, đó là cấu trúc, hành vi, nhóm và chú thích. Các phần tử này là các khối để xây dựng hướng đối tượng cơ bản của UML.

Phần từ cấu trúc là các danh từ trong mô hình UML. Chúng là bộ phận tĩnh của mô hình để biểu diễn các thành phần khái niệm hay vật lý. Có bảy loại phẩn tử cấu trúc như mô tả sau đây:

* **Lớp**. Lớp mô tả tập các đối tượng cùng chung thuộc tinh, thao tác, quan hệ và ngữ nghĩa. Một lớp cài đặt một hay nhiều ghép nối.Biểu diễn đồ họa lớp bằng hình chữ nhật, thông thường chú có tên, thuộc tính và thao tác.
* **Giao diện**. Giao diện là tập hợp các thao tác làm dịch vụ của lớp hay thành phần. Giao diện mô tả hành vi thấy được từ ngoài của thành phần. Giao diện biểu diễn toàn bộ hay một phần hành vi của lớp. Giao diện định nghĩa tập đặc tả thao tác chứ không định nghĩa cài đặt của chúng.Giao diện thường không đứng một mình mà được gắn vào lớp hay thành phần thực hiện giao diện.

Table

Description automatically generated Diagram

Description automatically generated

* **Giao diện Phần tử cộng tác**. Phần tử cộng tác là mô tả ngữ cảnh của tương tác. Ký pháp đồ họa của nó là hình elíp với đường vẽ nét đứt, kèm theo tên. Phần tử cộng tác thể hiện một giải pháp thi hành bên trong hệ thống, bao gồm các lớp, quan hệ và tương tác giữa chúng để đạt được một chức năng mong đợi của UC.

 A picture containing lamp, necklet

Description automatically generated

* **Trường hợp sử dụng (Use case).** Mô tả chình tự các hành động mà hệ thống sẽ thực hiện để đạt được một kết quả cho tác nhân nào đó. Tác nhân là những gì bên ngoài tương tác với hệ thống. Tập hợp các UC của hệ thống sẽ hình thành các trường hợp mà hệ thống được sử dụng. Sử dụng UC để cấu trúc các phần tử có tính hành vi trong mô hình. Nó được hiện thực hóa bởi phần tử cộng tác như vừa mô tả trên.
* **Lớp tích cực (active class).** Lớp tích cực là lớp có đối tượng làm chủ một hay nhiều lớp tiến trình hay luồng. Lớp tích cực được xem như lớp thông thường nhưng đối tượng của nó biểu diễn các thành phần có hành vi đang tương tranh với các thành phần khác. Ký pháp đồ họa của nó tương tự lớp thông thường nhưng biên chữ nhật được tô đậm. Thông thường chúng cũng có tên, thuộc tính và thao tác.
* **Thành phần**. Thành phần biểu diễn vật lý mã nguồn, các tệp nhị phân trong quá trình phát triển hệ thống.
* **Nút (node).** Nút là thể hiện thành phần vật lý, tồn tại khi chương trình chạy và biểu diễn các tài nguyên tính toán. Có thể đặt tập các thành phần trên nút chuyển từ nút này sang nút khác. Nút có thể là máy tính, thiết bị phần cứng.

Phần tử hành vi là bộ phận động của mô hình UML. Chúng là các động từ của mô hình, biểu diễn hành vi theo thời gian và không gian. Có hai loại chính là tương tác và trạng thái.

* **Tương tác.** Tương tác là hành vi bao gồm tập các thông điệp trao đổi giữa các đối tượng trong ngữ cảnh cụ thể để thực hiện mục đích cụ thể. Hành vi của nhóm đối tượng hay của mỗi thao tác có thể được chỉ ra bằng tương tác. Biểu diễ đồ họa của thông điệp được thể hiện bao gồm mũi tên và tên thao tác của nó.

A picture containing diagram

Description automatically generated A close-up of a logo

Description automatically generated with low confidence

* **Máy trạng thái**. Máy trạng thái là hành vi chỉ ra trật tự các trạng thái mà đối tượng hay tương tác sẽ đi qua để đáp ứng sự kiện. Hành vi của lớp hay cộng tác của lớp có thể được xác định bằng máy trạng thái. Máy trạng thái kích hoạt nhiều phần tử, bao gồm trạng thái, chuyển tiếp (từ trạng thái này sang trạng thái khác), sự kiện và các hoạt động (đáp ứng sự kiện). Ký pháp đồ họa của trạng thái, nó chứa tên và trạng thái con nếu có.

Phần tử nhóm là bộ phận tổ chức của mô hình UML. Chỉ có một phần tử thuộc nhóm này có tên là gói (package). Gói là cơ chế đa năng để tổ chức các phần tử vào nhóm. Các phần tử cấu trúc, hành vi và ngay cả phần tử nhóm có thể cho vào gói. Không giống thành phần (component), phần tử nhóm hoàn toàn là khái niệm, có nghĩa rằng chúng chỉ tồn tại vào thời điểm phát triển hệ thống chứ không tồn tại vào thời gian chạy chương trình. Ký pháp đồ họa của nhóm, gói giúp ta quan sát hệ thống ở mức tổng quan hơn.

* **Chú thích (annotaitonal)** Phần tử chú thích là bộ phận chú giải của mô hình UML. Đó là lời giải thích áp dụng để mô tả các phần tử khác trong mô hình. Phần tử chú thích được gọi là ghi chú (note).

Diagram

Description automatically generated

* + - 1. Các quan hệ trong UML

Có bốn loại quan hệ trong UML, bao gồm quan hệ phụ thuộc, kết hợp, khai quát và hiện thực hóa. Chúng là cơ sở đễ xây dựng mọi quan hệ trong UML.

* **Phụ thuộc (dependency).** Phục thuộc là quan hệ ngữ nghĩa hai phần tử trong đó thay đổi phần tử độc lập sẽ tác động đến ngữ nghĩa của phần tử phục thuộc.

Shape, rectangle

Description automatically generated

* **Kết hợp (association).** Kết hợp là quan hệ cấu trúc để mô tả tập liên kết (một liên kết là kết nối giữa các đối tượng). Khi đối tượng của lớp này gửi/nhận thông điệp đến/từ đối tượng của lớp kia thì ta gọi chúng là có quan hệ kết hợp. Ký pháp đồ họa của kết hợp chúng có thể chứa tên nhiệm vụ và tính nhiều (multiplicity). Tụ hợp (aggregation) là dạng đặc biệt của kết hợp, nó biểu diễn quan hệ cấu trúc giữa toàn thể và bộ phận. Một dạng dặc biệt của tập hợp là quan hệ hợp thành (composition), trong đó nếu như đối tượng toàn thể bị hủy bỏ thì các đối tượng bộ phận của nó cũng bị hủy bỏ theo.
* **Khái quát hóa (generalization).** Khái quát hóa là quan hệ đặc biệt hóa/ khái quát hóa mà trong đó đối tượng cụ thể sẽ kế thừa các thuộc tính và phương pháp của đối tượng tổng quát.



* **Hiện thực hóa (realization).** Hiện thực hóa là quan hệ ngữ nghĩa giữa giao diện và lớp (hay thành phần) hiện thực lớp; giữa UC và hợp tác hiện thực UC.

Shape, rectangle

Description automatically generated

* + - 1. Kiểu dữ liệu

Kiểu dữ liệu không phải là phần tử mô hình trong UML. Kiểu dữ liệu cơ sở là kiểu dữ liệu không có cấu trúc. UML có các kiểu dữ liệu sau:

* Boolean: là kiểu đếm với hai giá trị True và False
* Biểu thức (Expression): là xâu ký tự có cú pháp
* Tính nhiều (Multiplicity): là tập không rỗng của các số nguyên dương và ký tự \* (để biểu thị tính nhiều vô hạn).
* Tên (Name): là xâu ký tự cho khả năng đặc tả phẩn tử.
* Số nguyên (Integer): là kiểu cơ bản và là phần tử của tập vô hạn các sô nguyên âm và dương.
* Xâu (String): là trật tự cảu các ký tự, được sử dụng là tên.
* Thời gian (Time): xâu ký tự biểu dirn giá trị tuyệt đối hay khoảng tương tương đối.
* Không lý giải (Uninterpreted): là ‘cái gì đó’ mà ý nghĩa của nó phụ thuộc và lĩnh vực
  + - 1. Biểu đồ UML

UML sử dụng một hệ thống ký hiệu thống nhất biểu diễn các Phần tử mô hình (model elements). Tập hợp các phần tử mô hình tạo thành các Sơ đồ UML (UML diagrams). Có các loại sơ đồ UML chủ yếu sau:

* Sơ đồ lớp (Class Diagram)
* Sơ đồ đối tượng (Object Diagram)
* Sơ đồ tình huống sử dụng (Use Cases Diagram)
* Sơ đồ trình tự (Sequence Diagram)
* Sơ đồ cộng tác (Collaboration Diagram hay là Composite Structure Diagram)
* Sơ đồ trạng thái (State Machine Diagram)
* Sơ đồ thành phần (Component Diagram)
* Sơ đồ hoạt động (Activity Diagram)
* Sơ đồ triển khai (Deployment Diagram)
* Sơ đồ gói (Package Diagram)
* Sơ đồ liên lạc (Communication Diagram)
* Sơ đồ tương tác (Interaction Overview Diagram - UML 2.0)
  + Sơ đồ phối hợp thời gian (Timing Diagram - UML 2.0)

* + 1. Ngôn Ngữ HTML
       1. Khái niệm HTML

- HTML thực chất là từ viết tắt của "HyperText Markup Language", nghĩa là ngôn ngữ đánh dấu siêu văn bản. Trong đó, Hypertext (siêu văn bản) là một đoạn text bất kỳ nhưng có chứa link đến một nguồn thông tin khác (như một đoạn văn bản, một địa chỉ website, hình ảnh, âm thanh…).

- Ngôn ngữ đánh dấu này sẽ là cầu nối của sự giao tiếp giữa người dùng và trình duyệt, giúp trình duyệt hiểu được cách thức hiển thị trang web.

- Thẻ HTML được định nghĩa bằng một cặp từ khóa nằm giữa dấu < và dấu >, báo cho trình duyệt biết cách thức hiển thị đoạn kí tự bên trong thẻ HTML đó.



Hình 1.: HTML

* + - 1. Cấu trúc của HTML

- Cấu trúc của HTML rất đơn giản và logic, với bố cục từ trên xuống dưới, từ trái qua phải, bao gồm 2 phần chính là HEAD và BODY.

- Các website viết bằng HTML đều tuân theo cấu trúc cơ bản như sau:

* Mọi trang HTML đều phải khai báo DOCTYPE (định nghĩa chuẩn văn bản) ngay từ dòng đầu tiên.
* Thẻ <html> cho trình duyệt biết mở đầu và kết thúc của trang HTML.
* Thẻ <head> chứa tiêu đề và các thông tin khai báo, các thông tin ẩn khác.
* Thẻ <body> sẽ hiển thị nội dung của website. Đây là phần thông tin mà người dùng sẽ nhìn thấy khi trình duyệt đọc các mã HTML.
* Mọi ký tự nằm giữa dấu <!- và -> sẽ được xem là thẻ comment và bị trình duyệt bỏ qua, không xử lý và không hiển thị.

- Các tổng hợp các tag HTML:

* Đề mục: Có 6 loại đề mục tất cả với mức độ nhấn mạnh giảm dần từ <h1> cho đến <h6>.

    <h1>Tiêu đề bài viết</h1>

    <h2>Phần 1</h2>

    <h3>Chương 1.1</h3>

    <h4>Tiêu mục 1.1.1</h4>

    <h5>Đề mục 1.1.1.1</h5>

    <h6>Đề mục con 1.1.1.1.1</h6>

* Định dạng text:

    <b>Tô đậm</b>

    <i>In nghiêng</i>

    <strong>Tô đậm theo chuẩn web ngữ nghĩa</strong>

    <em>In nghiêm theo chuẩn web ngữ nghĩa</em>

* Phân đoạn:

    <!-- Comment: thẻ p dùng để đánh dấu đoạn văn,

        tách riêng với những đoạn text bình thường -->

    <p>Đoạn văn bản</p>

    <!-- Comment: br là thẻ đơn, nên không cần thẻ đóng -->

    Xuống dòng với <br>

* Danh sách:

<!-- Comment: Danh sách có sắp xếp thứ tự -->

<ol>

    <li>Phần tử 1</li>

    <li>Phần tử 2</li>

</ol>

<!-- Comment: Danh sách không sắp xếp thứ tự -->

<ul>

    <li>Phần tử 1</li>

    <li>Phần tử 2</li>

</ul>

* Liên kết:

<a href="địa chỉ liên kết">Đây là một liên kết</a>

<a href="địa chỉ liên kết" target="\_blank">Liên kết này sẽ mở ở cửa sổ</a>

* Ảnh:

<img src="đường dẫn đến ảnh" alt="chú thích cho ảnh">

* + 1. Ngôn ngữ CSS
       1. Khái niệm CSS

- CSS là từ viết tắt của Cascade Style Sheet, giúp trình duyệt hiểu được các thiết lập định dạng và bố cục cho trang web. CSS cho phép bạn điều khiển thiết kế của nhiều thành phần HTML chỉ với duy nhất 1 vùng chọn. Điều này giúp giảm tiết kiệm thời gian thiết kế và chỉnh sửa, khi bạn có thể tách biệt được cấu trúc HTML và định dạng CSS.

- CSS cho phép bạn đưa các thông tin định nghĩa thẻ thông qua nhiều con đường khác nhau. Style có thể được quy định ở trong một thẻ HTML, trong một trang web hoặc trong một file CSS bên ngoài.



Hình 1.: CSS

* + - 1. Cú pháp của CSS

- Cú pháp cơ bản của CSS bao gồm 3 phần là vùng chọn (selector), thuộc tính (property) và giá trị (value).

    VUNG-CHON {

        Thuộc-tính-1: giá-trị-1;

        Thuộc-tính-2: giá-trị-2;

    }

* Vùng chọn (selector): Là cách xác định các thẻ HTML dựa trên cấu trúc phân cấp. Vùng chọn có thể được tạo thành dựa trên nhiều yếu tố như định danh (id), tên lớp (class), quan hệ cha - con …
* Thuộc tính (property): Là yếu tố mà bạn muốn thay đổi ở các thẻ HTML thuộc vùng chọn.
* Giá trị (value): Mỗi thuộc tính sẽ yêu cầu một giá trị khác nhau. Đó có thể là một từ khóa định sẵn (none, block), một tên màu hay mã màu (black, white, #000, #FFFFFF) hay một giá trị kích thước tính bằng px, em, rem, %.

- Tổng hợp các vùng chọn:

* #ID: Dấu # được sử dụng để chọn một thẻ HTML có định danh (ID) cụ thể. Chính vì vậy, bản chất của ID là không thể tái sử dụng. Hãy chắc chắn rằng việc gán ID cho thẻ HTML và chọn thẻ này bằng ID này là cách duy nhất bạn có thể định dạng cho nó.
* .Class: Dấu "." đại diện cho tên lớp. Nhiều thẻ HTML có thể có cùng tên lớp. Chính vì vậy, sự khác nhau giữa ID và Class là ID chỉ cho phép chọn một phần tử, trong khi Class thì cho phép bạn chọn nhiều thẻ HTML cùng lúc.
* A.Class: Vùng chọn này giúp xác định tất cả các thẻ HTML thuộc cùng một lớp.
* A B: Vùng chọn này xác định tất cả các thẻ B nằm bên trong thẻ A, không kể đến phân cấp, ngôi thứ như con, …
* A > B: Vùng chọn này còn được gọi là vùng chọn con (child). Tuy nhiên, vùng chọn này chỉ xác định tất cả các thẻ B là con trực tiếp của thẻ A, chứ không nằm bên trong thẻ nào khác.
  + 1. Mô hình MVC

MVC (viết tắt của Model-View-Controller) là một mẫu kiến trúc phần mềm để tạo lập giao diện người dùng trên máy tính. MVC chia một ứng dụng thành ba phần tương tác được với nhau để tách biệt giữa cách thức mà thông tin được xử lý nội hàm và phần thông tin được trình bày và tiếp nhận từ phía người dùng.

Khi sử dụng đúng cách, mẫu MVC giúp cho người phát triển phần mềm cô lập các nguyên tắc nghiệp vụ và giao diện người dùng một cách rõ ràng hơn. Phần mềm phát triển theo mẫu MVC tạo nhiều thuận lợi cho việc bảo trì vì các nguyên tắc nghề nghiệp và giao diện ít liên quan với nhau.

Trong mẫu Model-View-Controller, mô hình (model) tượng trưng cho dữ liệu của chương trình phần mềm. Tầm nhìn hay khung nhìn (view) bao gồm các thành phần của giao diện người dùng. Bộ kiểm tra hay bộ điều chỉnh (controller) quản lý sự trao đổi giữa dữ liệu và các nguyên tắc nghề nghiệp trong các thao tác liên quan đến mô hình.

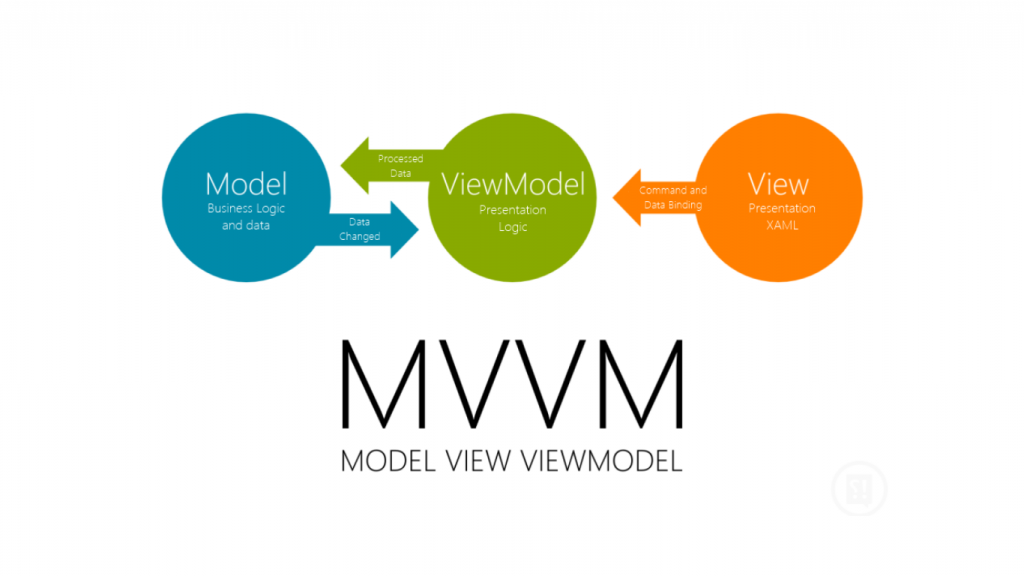


Hình 1.: Mô hình MVC

* + 1. Mô hình MVVM (Model – View – View - Model)

Là một mẫu kiến ​​trúc phần mềm tạo điều kiện cho việc tách biệt sự phát triển của giao diện người dùng đồ họa ( chế độ xem ) - có thể là thông qua ngôn ngữ đánh dấu hoặc mã GUI - khỏi sự phát triển của logic nghiệp vụ hoặc back- logic kết thúc ( mô hình ) để chế độ xem không phụ thuộc vào bất kỳ nền tảng mô hình cụ thể nào. Các viewmodel của MVVM là một chuyển đổi giá trị,nghĩa viewmodel có trách nhiệm tiếp xúc (chuyển đổi) các đối tượng dữ liệutừ mô hình theo cách mà các đối tượng được quản lý và trình bày dễ dàng. Về mặt này, view model là mô hình hơn là view và xử lý hầu hết nếu không muốn nói là tất cả logic hiển thị của view. Chế độ xem có thể triển khai một mẫu dàn xếp, tổ chức quyền truy cập vào logic back-end xung quanh tập hợp các trường hợp sử dụng được hỗ trợ bởi chế độ xem.

Model – view – viewmodel còn được gọi là model – view – binder , đặc biệt là trong các triển khai không liên quan đến nền tảng .NET.



Hình 1.: Mô hình MVVM

* 1. KIẾN TRÚC HỆ THỐNG C4 Model
     1. C4 model

C4 là viết tắt của **ngữ cảnh** , **bộ chứa** , **thành phần** và **mã** — một tập hợp các sơ đồ phân cấp mà bạn có thể sử dụng để mô tả kiến ​​trúc phần mềm của mình ở các mức thu phóng khác nhau, mỗi mức hữu ích cho các đối tượng khác nhau.

Mô hình C4 là một phương pháp mô tả kiến trúc phần mềm một cách tinh gọn nhất. Nó bao gồm một tập hợp 4 sơ đồ mô tả **cấu trúc tĩnh** của một hệ thống phần mềm.

Hệ thống phân cấp của sơ đồ C4 cung cấp các mức trừu tượng khác nhau, mỗi mức có liên quan đến một đối tượng khác nhau.

**Level 1: System context diagram**

Sơ đồ ngữ cảnh hệ thống là điểm khởi đầu tốt để lập sơ đồ và ghi lại hệ thống phần mềm, cho phép bạn lùi lại và xem bức tranh toàn cảnh. Vẽ một sơ đồ thể hiện hệ thống của bạn dưới dạng một hộp ở giữa, xung quanh là những người dùng của nó và các hệ thống khác mà nó tương tác.

Chi tiết không quan trọng ở đây vì đây là chế độ xem thu nhỏ của bạn hiển thị bức tranh toàn cảnh về toàn cảnh hệ thống. Nên tập trung vào con người (diễn viên, vai trò, nhân cách, v.v.) và hệ thống phần mềm hơn là công nghệ, giao thức và các chi tiết cấp thấp khác. Đó là loại sơ đồ mà bạn có thể hiển thị cho những người không có kỹ thuật.

* **Phạm vi** : Một hệ thống phần mềm duy nhất.
* **Các yếu tố chính** : Hệ thống phần mềm trong phạm vi.  
  **Các yếu tố hỗ trợ** : Con người (ví dụ: người dùng, diễn viên, vai trò hoặc cá nhân) và hệ thống phần mềm (phụ thuộc bên ngoài) được kết nối trực tiếp với hệ thống phần mềm trong phạm vi. Thông thường, các hệ thống phần mềm khác này nằm ngoài phạm vi hoặc ranh giới của hệ thống phần mềm của riêng bạn và bạn không có trách nhiệm hoặc quyền sở hữu đối với chúng.
* **Đối tượng dự kiến** : Tất cả mọi người, cả dân kỹ thuật và phi kỹ thuật, trong và ngoài nhóm phát triển phần mềm.
* **Được đề xuất cho hầu hết các đội** : Có.

**Level 2: Container diagram**

Sau khi bạn hiểu cách hệ thống của mình phù hợp với môi trường CNTT tổng thể, bước tiếp theo thực sự hữu ích là phóng to ranh giới hệ thống bằng sơ đồ Vùng chứa. "Vùng chứa" giống như ứng dụng web phía máy chủ, ứng dụng một trang, ứng dụng dành cho máy tính để bàn, ứng dụng dành cho thiết bị di động, lược đồ cơ sở dữ liệu, hệ thống tệp, v.v. Về cơ bản, vùng chứa là một đơn vị có thể chạy/triển khai riêng biệt (ví dụ: một không gian xử lý riêng biệt ) thực thi mã hoặc lưu trữ dữ liệu.

Sơ đồ bộ chứa hiển thị hình dạng cấp cao của kiến ​​trúc phần mềm và cách phân bổ trách nhiệm trên đó. Nó cũng cho thấy các lựa chọn công nghệ chính và cách các vùng chứa giao tiếp với nhau. Đó là một sơ đồ đơn giản, tập trung vào công nghệ cấp cao, hữu ích cho các nhà phát triển phần mềm cũng như nhân viên hỗ trợ/vận hành.

* **Phạm vi** : Một hệ thống phần mềm duy nhất.
* **Các phần tử chính** : Các bộ chứa trong hệ thống phần mềm trong phạm vi.  
  **Yếu tố hỗ trợ** : Con người và hệ thống phần mềm kết nối trực tiếp với container.
* **Đối tượng dự định** : Những người kỹ thuật trong và ngoài nhóm phát triển phần mềm; bao gồm kiến ​​trúc sư phần mềm, nhà phát triển và nhân viên vận hành/hỗ trợ.
* **Được đề xuất cho hầu hết các đội** : Có.
* **Lưu ý** : Sơ đồ này không nói gì về các kịch bản triển khai, phân cụm, sao chép, chuyển đổi dự phòng, v.v.

**Level 3: Component diagram**

Tiếp theo, bạn có thể phóng to và phân tách từng vùng chứa hơn nữa để xác định các khối xây dựng cấu trúc chính và các tương tác của chúng.

Biểu đồ Thành phần cho biết cách một vùng chứa được tạo thành từ một số "thành phần", mỗi thành phần đó là gì, trách nhiệm của chúng và chi tiết công nghệ/triển khai.

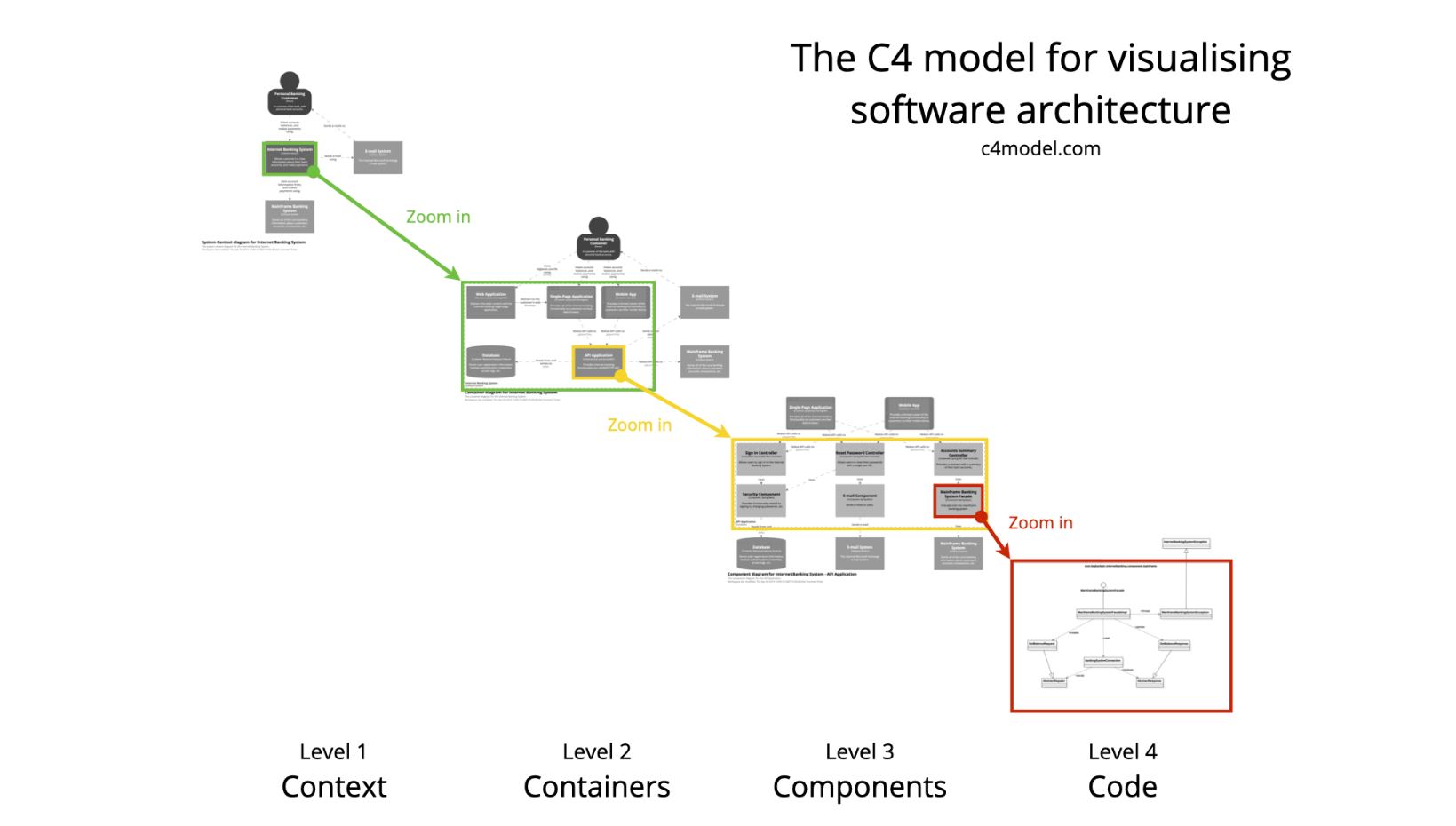
* **Phạm vi** : Một container duy nhất.
* **Các phần tử chính** : Các thành phần bên trong vùng chứa trong phạm vi.  
  **Các yếu tố hỗ trợ** : Các thùng chứa (trong phạm vi hệ thống phần mềm) cộng với con người và hệ thống phần mềm được kết nối trực tiếp với các thành phần.
* **Đối tượng dự định** : Kiến trúc sư và nhà phát triển phần mềm.
* **Được đề xuất cho hầu hết các nhóm** : Không, chỉ tạo sơ đồ thành phần nếu bạn cảm thấy chúng tăng thêm giá trị và xem xét việc tự động hóa quá trình tạo của chúng để có tài liệu tồn tại lâu dài.

**Level 4: Code**

Cuối cùng, bạn có thể phóng to từng thành phần để hiển thị cách nó được triển khai dưới dạng mã; sử dụng sơ đồ lớp UML, sơ đồ mối quan hệ thực thể hoặc tương tự.

Đây là mức độ chi tiết tùy chọn và thường có sẵn theo yêu cầu từ các công cụ như IDE. Lý tưởng nhất là sơ đồ này sẽ được tạo tự động bằng cách sử dụng công cụ (ví dụ: công cụ lập mô hình IDE hoặc UML) và bạn nên xem xét chỉ hiển thị những thuộc tính và phương thức cho phép bạn kể câu chuyện mà bạn muốn kể. Mức độ chi tiết này không được khuyến nghị cho bất cứ điều gì ngoại trừ các thành phần quan trọng nhất hoặc phức tạp nhất.

* **Phạm vi** : Một thành phần duy nhất.
* **Các phần tử chính** : Các phần tử mã (ví dụ: lớp, giao diện, đối tượng, hàm, bảng cơ sở dữ liệu, v.v.) trong thành phần trong phạm vi.
* **Đối tượng dự định** : Kiến trúc sư và nhà phát triển phần mềm.
* **Được đề xuất cho hầu hết các nhóm** : Không, đối với tài liệu tồn tại lâu dài, hầu hết các IDE có thể tạo mức độ chi tiết này theo yêu cầu.



Hình 1.: C4 Model

* + 1. Ký hiệu

Mô hình C4 không quy định bất kỳ ký hiệu cụ thể nào. Một ký hiệu đơn giản hoạt động tốt trên bảng trắng, giấy, ghi chú dán, thẻ mục lục và nhiều công cụ vẽ sơ đồ như sau.

Sau đó, bạn có thể sử dụng màu sắc và hình dạng để bổ sung cho sơ đồ, để thêm thông tin bổ sung hoặc đơn giản là làm cho sơ đồ trở nên đẹp mắt hơn về mặt thẩm mỹ.

Mặc dù mô hình C4 là cách tiếp cận ưu tiên trừu tượng hóa và ký hiệu độc lập, bạn vẫn cần đảm bảo rằng ký hiệu sơ đồ của bạn có ý nghĩa và các sơ đồ đó có thể hiểu được. Và đây là một số khuyến nghị liên quan đến ký hiệu.

**Sơ đồ**

* Mọi sơ đồ phải có tiêu đề mô tả loại sơ đồ và phạm vi (ví dụ: "Sơ đồ ngữ cảnh hệ thống cho hệ thống phần mềm của tôi").
* Mỗi sơ đồ nên có một khóa/chú giải giải thích ký hiệu đang được sử dụng (ví dụ: hình dạng, màu sắc, kiểu đường viền, loại đường, đầu mũi tên, v.v.).
* Các từ viết tắt và từ viết tắt (kinh doanh/miền hoặc công nghệ) phải dễ hiểu đối với tất cả khán giả hoặc được giải thích trong khóa/chú giải sơ đồ.

**Yếu tố**

* Loại của mọi phần tử phải được chỉ định rõ ràng (ví dụ: Người, Hệ thống phần mềm, Bộ chứa hoặc Thành phần).
* Mỗi yếu tố nên có một mô tả ngắn, để cung cấp một cái nhìn "trong nháy mắt" về các trách nhiệm chính.
* Mỗi vùng chứa và thành phần phải có một công nghệ được chỉ định rõ ràng.

**Các mối quan hệ**

* Mỗi dòng nên đại diện cho một mối quan hệ đơn hướng.
* Mỗi dòng phải được dán nhãn, nhãn nhất quán với hướng và mục đích của mối quan hệ (ví dụ: phụ thuộc hoặc luồng dữ liệu). Cố gắng ghi nhãn càng cụ thể càng tốt, tốt nhất là tránh những từ đơn lẻ như "Công dụng".
* Mối quan hệ giữa các vùng chứa (thường là những mối quan hệ này đại diện cho giao tiếp giữa các quá trình) nên có một công nghệ/giao thức được dán nhãn rõ ràng.
  1. MỘT SỐ THƯ VIỆN, FRAMEWORK HỖ TRỢ
     1. Thư viện Javascript
        1. Khái niệm Javascript

- JavaScript là một ngôn ngữ lập trình website, được tích hợp và nhúng trong HTML giúp website sống động hơn. JavaScript cho phép kiểm soát các hành vi của trang web tốt hơn so với khi chỉ sử dụng mỗi HTML. Thế mạnh của Javascript là nó tương thích với nhiều thiết bị kết nối khác nhau.

- JavaScript là một trong các ngôn ngữ lập trình đa nền tảng được hỗ trợ hầu như trên tất cả các trình duyệt như Firefox, Chrome, … thậm chí các trình duyệt trên thiết bị di động.



Hình 1.: Javascript

* + - 1. Cách hoạt động của Javascript

JavaScript thường được nhúng trực tiếp vào một trang web hoặc được tham chiếu qua file .js riêng. Nó là ngôn ngữ lập trình từ phía khách hàng, tức là script được tải về máy của người dùng đang truy cập và được xử lý tại đó, thay vì xử lý trên máy chủ rồi mới đưa kết quả tới khách hàng.

* + - 1. Cấu trúc của Javascript

- Cặp thẻ mở và thẻ đóng:

<script>

    alert('Hello World!')

</script>

- Đặt thẻ script: Có ba cách đặt thẻ script thường được sử dụng dưới đây:

* Internal – viết trong file html hiện tại

Thông thường, bạn có thể viết những đoạn mã Javascript trên phần head. Tuy nhiên, đó cũng không phải là điều kiện bắt buộc. Vì vậy bạn có thể đặt ở bất kỳ đâu tùy thích miễn là được bao lại bằng thẻ script.

>Ví dụ: Đặt trong thẻ head

<html>

<head>

    <title></title>

    <script>

        alert("Hello World!");

    </script>

</head>

<body>

</body>

</html>

* + External – viết ra một file js khác rồi import vào

Bạn có thể viết những đoạn mã Javascript ở một file có phần mở rộng là .js, sau đó dùng thẻ script để import vào. Lúc này bên trong file demo.js bạn không đặt thẻ scirpt. Vì nhờ vào đuôi .js, trình duyệt sẽ tự nhận đây là file chứa mã Javascript.

>Ví dụ:

<script src="demo.js"></script>

* + Inline – viết trực tiếp trong thẻ HTML:

Với Inline, bạn sẽ viết những đoạn mã Javascript trực tiếp trong thẻ HTML.

>Ví dụ: Viết dưới dạng inline vì đoạn mã alert(1) được đặt trong sự kiện onclick của thẻ button.

<input type="button" onclick="alert(1)" value="Click" />

* + - 1. Javascript dùng làm gì

- JavaScript có thể được ứng dụng trong rất nhiều lĩnh vực:

* Lập trình website.
* Xây dựng ứng dụng cho website máy chủ.
* Ứng dụng di động, app, trò chơi

- Khi tải một trang web, trình duyệt phân tích cú pháp HTML và tạo ra một loại dữ liệu gọi là DOM từ nội dung. DOM thể hiện chế độ xem trực tiếp của trang web với mã JavaScript. Đoạn mã này thực hiện cập nhật cho DOM và được trình bày ngay lập tức cho người dùng.

- Trình duyệt cũng ghi nhận các sự kiện giao diện người dùng như: di chuyển chuột, nhấp chuột, v.v. Sau đó, tùy theo phản hồi của người dùng, đoạn mã sẽ thực hiện công việc được lập trình tương ứng. Sử dụng tất cả các tiện ích này, bạn có thể xây dựng các ứng dụng nhỏ để phục vụ cho một vài mục đích được cho phép.

* + - 1. Lợi ích khi dùng Javascript

- Là một ngôn ngữ lập trình phổ biến sử dụng trên 92% nền tảng website hiện nay, JavaScript đã thể hiện vai trò quan trọng với lĩnh vực này. Tầm quan trọng của nó thể hiện qua các thao tác, công dụng như:

* Triển khai tập lệnh phía máy khách: Nhờ Javascript, các lập trình viên có thể dễ dàng viết tập lệnh phía máy khách, tích hợp các tập lệnh một cách liền mạch vào HTML, cho phép website tương tác, trả lời người dùng ngay lập tức và tạo ra giao diện hiển thị phong phú hơn.
* Viết mã phía máy chủ: Các lập trình viên có thể viết mã phía máy chủ bằng JavaScript.
* Đơn giản hóa phát triển ứng dụng web phức tạp: Javascript cho phép các nhà phát triển đơn giản hóa thành phần của ứng dụng, qua đó đơn giản hóa việc phát triển các ứng dụng web phức tạp.
* Thiết kế web responsive: JavaScript cho phép thiết kế web responsive – tối ưu trên cả máy tính và thiết bị di động chỉ với một bộ mã.
  + - 1. Ưu điểm và nhược điểm của Javascript

- Ưu điểm:JavaScript có rất nhiều ưu điểm vượt trội hơn so với các đối thủ khác, chẳng hạn như:

* JavaScript là ngôn ngữ lập trình dễ học.
* Lỗi của JavaScript dễ phát hiện hơn và vì vậy dễ sửa hơn.
* JavaScript hoạt động trên nhiều trình duyệt, nền tảng.
* JavaScript giúp website tương tác tốt hơn với khách truy cập.
* JavaScript nhanh hơn và nhẹ hơn các ngôn ngữ lập trình khác.

- Nhược điểm:

+ Không ngôn ngữ lập trình nào là hoàn hảo và không có khuyết điểm. Bởi JavaScript là ngôn ngữ lập trình phổ biến, được sử dụng rộng rãi trên nền tảng website, vì vậy nó cũng là đối tượng tiềm năng của một lượng lớn hacker, scammer – những kẻ chuyên tìm kiếm lỗ hổng và các lỗi bảo mật và thâm nhập, đánh cắp dữ liệu nội bộ.

+ Một số khuyết điểm của JavaScript có thể kể đến như:

* Dễ bị khai thác.
* Có thể được dùng để thực thi mã độc trên máy tính của người dùng.
* Có thể bị triển khai khác nhau tùy từng thiết bị dẫn đến việc không đồng nhất.
  + 1. Thư viện Jquery
       1. Khái niệm Jquery

JQuery là một thư viện JavaScript nhỏ gọn, chạy nhanh. jQuery giải quyết các vấn đề tương tác với HTML, bắt sự kiện, hiệu ứng động ... trở lên rất đơn giản. Mọi tính năng của jQuery thực ra nó bao bọc các lệnh, hàm của JavaScript nên để hiểu jQuery bạn cần hiểu về HTML cơ bản cũng như CSS cơ bản trước.



Hình 1.: Jquery

* + - 1. Các tính năng của Jquery

- Các tính năng chính jQuery cung cấp gồm:

* Tương tác với HTML/DOM
* Tương tác với CSS
* Bắt và xử lý sự kiện HTML
* Các hiệu ứng và chuyển động trong HTML
* AJAX (Asynchronous JavaScript and XML)
* JSON parsing
* Các tiện ích xây dựng sẵn
  + - 1. Cài đặt Jquery vào Website

- Để sử dụng jQuery trong trang HTML, bạn cần đảm bảo trang HTML của mình load chính xác thư viện jQuery bằng cách sử dụng thẻ <script> ở phần <head> của HTML.

<html>

    <head>

        <script src="file-jquery.js"></script>

    </head>

    <body>

        ....

    </body>

</html>

* Cách 1: tải về file js thư viện tại jQuery download (nên chọn bản đã nén có chữ .min.js), sau đó tích hợp vào trang bằng thẻ <script>.

<script src="/public/jquery-3.1.1.min.js"></script>

* Cách 2: thay vì phải tải về file thư viện, bạn dùng luôn từ các CDN (Content Delivery Network) giúp cho trang tải nhanh hơn. Dùng CDN Google: Lấy link phiên bản jQuery tại jQuery Google, sau đó tích hợp vào Website.

<script

src="https://ajax.googleapis.com/ajax/libs/jquery/3.2.1/jquery.min.js">

</script>

* + - 1. Cú pháp

- Bạn có thể sử dụng các hàm của thư viện jQuery ở bất kỳ đâu trên trang, tuy nhiên cách tốt để thực hành là cần đảm bảo các mã sử dụng jQuery do bạn viết chỉ chạy sau khi toàn bộ trang HTML được tải đầy đủ. Để làm điều này thì bạn cần bắt sự kiện ready của đối tượng document.

$(document).ready(function () {

  // Mã JavaScript / jQuery ở đây

});

Hoặc viết gọn lại

$(function () {

  // Mã JavaScript / jQuery ở đây

});

- Ký tự $ được dùng để truy cập jQuery.

- Khi sử dụng jQuery, bạn cần chọn các phần tử(query) sau đó thực hiện các hành động trên chúng, điều này được thể hiện bằng cú pháp sử dụng jQuery:

$("selector").action();

* $ là ký hiệu cho biết bạn truy cập jQuery
* (selector) phần tìm phần tử HTML, theo cách chọn phần tử như đã biết trong CSS .

\* Ví dụ chọn tất cả các phần tử p là $('p')

\* Ví dụ chọn phần tử có id là examp là $('#examp')

\* Ví dụ chọn các phần tử có class là examclass là $('.examclass')

* action() là các hành động trên phần tử đã chọn, là các hàm mà jQuery cung cấp.
  + 1. Thư viện ReactJs
       1. Giới thiệu về ReactJs

React là một thư viện JavaScript do Facebook phát triển, giúp hiển thị giao diện ứng dụng (web, mobile, 3D…) một cách tối ưu nhất có thể, với tư duy [thiết kế giao diện theo component](https://codeschool.vn/portal/kien-truc-web-app-dua-tren-component/).

[Package “react”](https://www.npmjs.com/package/react) cung cấp thư viện bao gồm các hàm và class để tạo ra component và quản lý state của từng component. Bất kỳ nền tảng nào chạy được JavaScript đều có thể sử dụng package “react”.

Package “[react-dom](https://www.npmjs.com/package/react-dom)” là thư viện bổ sung cho “react”, nó thao tác với DOM để hiển thị (render) các component một cách tối ưu nhất có thể, dựa vào thuật toán DOM ảo (virtual DOM, thường gọi là vDOM). Khi các component xảy ra thay đổi, tất cả những thay đổi ấy sẽ được render lên vDOM, sau đó cây vDOM sẽ được so sánh với cây DOM thật của trình duyệt, đảm bảo chỉ những node nào thật sự khác biệt mới được cập nhật lên DOM thật. Thuật toán này giúp cho giao diện phức tạp vẫn giữ được sự mượt mà, tránh giật lag khi có nhiều component được cập nhật.

Package “[react-native](https://www.npmjs.com/package/react-native)” cũng là thư viện bổ sung cho “react” nhưng dùng để phát triển giao diện cho ứng dụng di động. Trong khi “react-dom” gắn kết React core với trình duyệt để hiển thị ứng dụng web, “react-native” lại giúp gắn kết giữa core với nền tảng di động bên dưới (Android, iOS…), giúp lập trình viên chỉ cần thao tác với JavaScript, những chi tiết liên quan đến native API sẽ được xử lý giùm, nhờ đó cùng một mã nguồn React Native có thể build ra được app cho các nền tảng khác nhau (Android, iOS v.v.)

Rất dễ sử dụng, vì nó được viết bằng html, css, javascript nên bạn chỉ cần có kiến thức cơ bản về 3 cái đó là có thể sử dụng bootstrap tốt. Tính năng Responsive hữu ích: Bootstrap xây dựng sẵn reponsive css trên các thiết bị mobiles, tablets, và desktops

Tương thích với trình duyệt: Nó tương thích với tất cả các trình duyệt (Chrome, Firefox, Internet Explorer, Safari, and Opera).

* + - 1. Ưu và nhược điểm

Mỗi thư viện thành phần React UI khác nhau đều có ưu và nhược điểm. Nhưng nói chung, sử dụng bất kỳ thư viện nào cũng có nhiều lợi ích sẽ giúp bạn khi làm việc trên dự án React của mình:

* Thân thiện với người mới bắt đầu .
* Tạo mẫu nhanh hơn.
* Tiết kiệm thời gian.
* Các thành phần có thể nhận biết được bởi người dùng.
* Các thành phần có thể tùy chỉnh.
* Khả năng tương thích đã được chứng minh trên các thiết bị .
* Có thể truy cập theo mặc định Hầu hết các thư viện thành phần React UI phổ biến đều có các tính năng trợ năng được tích hợp sẵn hoặc thậm chí tuân thủ đầy đủ WCAG hoặc các tiêu chuẩn và phương pháp hay nhất khác.
* Nguồn gốc từ cộng đồng. Thư viện thành phần giao diện người dùng thường có cộng đồng của họ tập trung xung quanh GitHub.

Mặc dù có nhiều ưu điểm như vậy, ngay cả những thư viện thành phần React UI tốt nhất cũng có một số nhược điểm

* Tùy chỉnh các thành phần có thể khó khăn.
* Thiết kế web tương tự với các trang khác .
* Hỗ trợ dựa vào cộng đồng. Hầu hết các thư viện React UI không cung cấp hỗ trợ chính thức mà thay vào đó hướng dẫn người dùng của họ đến Stack Overflow, GitHub, Discord hoặc các kênh tương tự khác. Với các thư viện ít phổ biến hơn, cộng đồng nhỏ hơn và việc nhận trợ giúp có thể phức tạp hơn.
  + 1. NodeJs

**NodeJS** là một môi trường runtime chạy [JavaScript](https://vietnix.vn/javascript-la-gi/) đa nền tảng và có mã nguồn mở, được sử dụng để chạy các ứng dụng web bên ngoài trình duyệt của client. Nền tảng này được phát triển bởi Ryan Dahl vào năm 2009, được xem là một giải pháp hoàn hảo cho các ứng dụng sử dụng nhiều dữ liệu nhờ vào mô hình hướng sự kiện (event-driven) không đồng bộ.

**3.4.1 Ưu nhược điểm**

**\* Ưu điểm**

* IO hướng sự kiện không đồng bộ, cho phép xử lý nhiều yêu cầu đồng thời.
* Sử dụng JavaScript – một ngôn ngữ lập trình dễ học.
* Chia sẻ cùng code ở cả phía client và server.
* [NPM](https://vietnix.vn/npm-la-gi/)(Node Package Manager) và module Node đang ngày càng phát triển mạnh mẽ.
* Cộng đồng hỗ trợ tích cực.
* Cho phép stream các file có kích thước lớn.

**\* Nhược điểm**

* Không có khả năng mở rộng, vì vậy không thể tận dụng lợi thế mô hình đa lõi trong các phần cứng cấp server hiện nay.
* Khó thao tác với cơ sử dữ liệu quan hệ.
* Mỗi callback sẽ đi kèm với rất nhiều callback lồng nhau khác.
* Cần có kiến thức tốt về JavaScript.
* Không phù hợp với các tác vụ đòi hỏi nhiều CPU.
  1. API

API là cụm viết tắt của Giao diện chương trình ứng dụng. Trong ngữ cảnh API, từ “Ứng dụng” đề cập đến mọi phần mềm có chức năng riêng biệt. Giao diện có thể được xem là một hợp đồng dịch vụ giữa 2 ứng dụng. Hợp đồng này xác định cách thức hai ứng dụng giao tiếp với nhau thông qua các yêu cầu và phản hồi. Tài liệu API của các ứng dụng này chứa thông tin về cách nhà phát triển xây dựng cấu trúc cho những yêu cầu và phản hồi đó.

Kiến trúc API thường được giải thích dưới dạng máy chủ và máy khách. Ứng dụng gửi yêu cầu được gọi là máy khách, còn ứng dụng gửi phản hồi được gọi là máy chủ. Như vậy, trong ví dụ về thời tiết, cơ sở dữ liệu của cơ quan thời tiết là máy chủ còn ứng dụng di động là máy khách.

* 1. CƠ SỞ DỮ LIỆU MYSQL
     1. Khái niệm cơ sở dữ liệu mysql

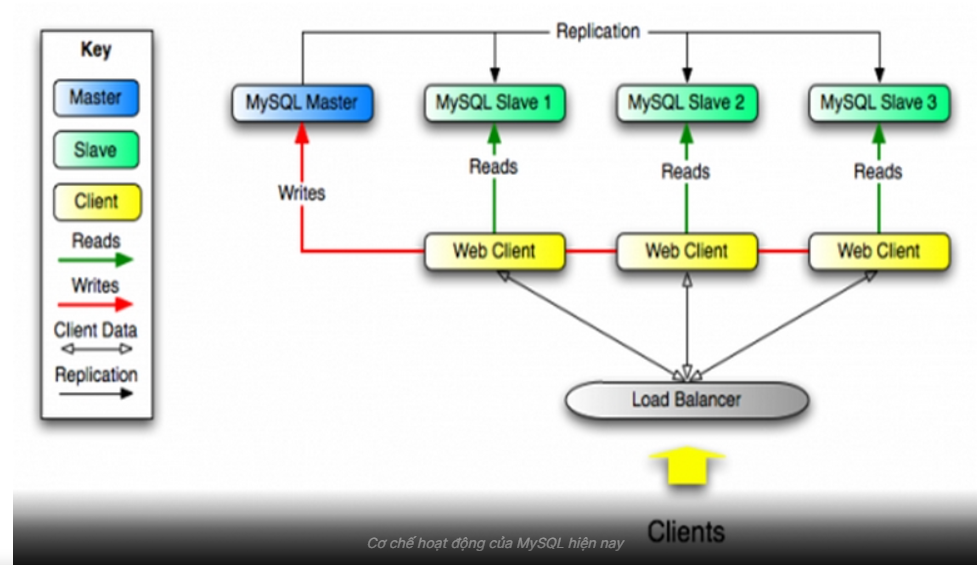
**MySQL là gì?** MySQL là 1 hệ thống quản trị về cơ sở dữ liệu với mã nguồn mở (được gọi tắt là RDBMS) và đang hoạt động theo mô hình dạng client-server. Đối với RDBMS – Relational Database Management System thì MySQL đã được tích hợp apache và PHP.

Được phát hành chính thức từ thập niên 90s, MySQL hiện đang quản lý dữ liệu qua những cơ sở dữ liệu, với mỗi một cơ sở dữ liệu hoàn toàn có thể có rất nhiều những bản quan hệ có chứa dữ liệu. Ngoài ra, MySQL cũng có cùng 1 cách thức truy xuất cũng như mã lệnh tương tự cùng với ngôn ngữ SQL.

* + 1. Cơ chế hoạt động của MySql

Với những thông tin chi tiết trên về khái niệm MySQL, vậy cơ chế hoạt động của **MySQL là gì**? Theo đó cách thức vận hành chính tại MySQL môi trường hiện tại là:

* MySQL đang tạo ra bảng để có thể lưu trữ dữ liệu và định nghĩa về sự liên quan giữa những bảng đó
* Client sẽ trực tiếp gửi yêu cầu SQL bằng 1 lệnh đặc biệt có trên MySQL.
* Ứng dụng tại server sẽ tiến hành phản hồi thông tin cũng như trả về những kết quả trên máy client.



Hình 1.: MySQL

* + 1. Ưu điểm và nhược điểm của Mysql

**\* Ưu điểm MySQL**

* **Nhanh chóng:** Nhờ vào việc đưa ra một số những tiêu chuẩn và cho phép MySQL làm việc hiệu quả cũng như tiết kiệm chi phí, giúp gia tăng tốc độ thực thi.
* **Mạnh mẽ và khả năng mở rộng:** MySQL hoàn toàn có thể xử lý số lượng lớn dữ liệu và đặc biệt hơn thế nữa thì nó còn có thể mở rộng nếu như cần thiết.
* **Đa tính năng:** Ưu điểm MySQL là gì? MySQL hiện đang hỗ trợ nhiều những chức năng SQL rất được mong chờ từ 1 hệ quản trị CSDL quan hệ cả gián tiếp cũng như trực tiếp.
* **Độ bảo mật cao:** MySQL là gì? Hiện tại nó đang rất thích hợp cho những ứng dụng truy cập CSDL thông qua internet khi sở hữu rất nhiều những tính năng về bảo mật và thậm chí là đang ở cấp cao.
* **Dễ dàng sử dụng:** MySQL đang là cơ sở dữ liệu dễ sử dụng, ổn định, tốc độ cao và hoạt động trên rất nhiều những hệ điều hành đang cung cấp 1 hệ thống lớn những hàm tiện ích rất mạnh.

**\* Nhược điểm MySQL là gì?**

* **Dung lượng hạn chế:** Trong trường hợp nếu như số lượng bản ghi của bạn đang lớn dần lên thì khi đó quá trình truy xuất dữ liệu sẽ diễn ra vô cùng khó khăn. Như vậy cần phải áp dụng rất nhiều những biện pháp khác nhau để có thể gia tăng được tốc độ truy xuất những dữ liệu ví dụ như tạo cache MySQL hoặc chia tải database ra nhiều server.
* **Độ tin cậy:** Nhược điểm MySQL là gì? Theo đó cách thức nhận chức năng cụ thể đang được xử lý cùng với MySQL (ví dụ như kiểm toán, những giao dịch, tài liệu tham khảo,…) khiến cho nó trở nên kém tin cậy hơn một số những hệ quản trị về cơ sở dữ liệu có quan hệ khác.
* **Giới hạn:** MySQL là gì? Theo thiết kế thì MySQL không có ý định thực hiện toàn bộ và nó đang đi kèm cùng với những hạn chế liên quan tới chức năng mà một số ứng dụng có thể cần tới. Vậy hiện tại điểm khác biệt giữa MySQL Server và MySQL là gì?
  1. CÔNG CỤ HỖ TRỢ LẬP TRÌNH
     1. Visual Studio Code
        1. Visual Studio Code là gì?

- Visual Studio Code (VS Code) là một trình soạn thảo mã nguồn mở gọn nhẹ nhưng có khả năng vận hành mạnh mẽ trên 3 nền tảng là Windows, Linux và macOS được phát triển bởi Microsoft. Nó hỗ trợ cho JavaScript, Node.js và TypeScript, cũng như cung cấp một hệ sinh thái mở rộng vô cùng phong phú cho nhiều ngôn ngữ lập trình khác.



Hình 1.: Visual Studio Code

* + - 1. Ưu điểm của Visual Studio Code
* Cài đặt nhanh – chỉ trong 1 phút là xong.
* Khởi động chương trình khi viết code chỉ sau vài giây.
* Thao tác file và thư mục trên trình soạn thảo code rất dễ dàng.
* Giao diện đơn giản, dễ sử dụng.
* Số lượng các tiện ích mở rộng phong phú dành cho nhiều ngôn ngữ khác nhau.
* Được tạo ra bởi tập đoàn Microsoft và nó hoàn toàn miễn phí.
  + - 1. Cài đặt Visual Studio Code

Bước 1: Tải phần mềm VS Code

- Link: <https://code.visualstudio.com/download>

- Bạn truy cập vào link trên và chọn hệ điều hành thích hợp để download

Graphical user interface, website

Description automatically generated

Bước 2: Nhấn vào file vừa tải để cài đặt

- File vừa tải xuống thường nằm ở trong thư mục download của máy tính. Hoặc bạn theo dõi ở góc trái trình duyệt.

- Sau đó chọn I accept the agreement và nhấn vào Next để cài đặt như hình vẽ bên dưới

Graphical user interface, text, application, Word

Description automatically generated

- Tiếp tục nhấn next cho đến khi hoàn thành

Graphical user interface, application, Word

Description automatically generated

Bước 3: Khởi động phần mềm

- Sau khi cài đặt hoàn thành bạn click vào biểu tượng phần mềm để vào giao diện chính.

A screenshot of a computer

Description automatically generated

Vậy là chúng ta đã cài đặt thành công phần mềm Visual Studio Code rất nhanh chóng. Bây giờ bạn hãy vào menu File chọn Add Folder to Workspace... để bắt đầu code dự án.

* + 1. Visual Studio Current
       1. Visual Studio Current là gì

Visual studio là một trong những công cụ hỗ trợ lập trình website rất nổi tiếng nhất hiện nay của Mcrosoft và chưa có một phần mềm nào có thể thay thế được nó. Visual Studio được viết bằng 2 ngôn ngữ đó chính là C# và VB+. Đây là 2 ngôn ngữ lập trình giúp người dùng có thể lập trình được hệ thống một các dễ dàng và nhanh chóng nhất thông qua Visual Studio.

* + - 1. Ưu điểm khi sử dụng Visual Studio Current

Visual Studio hỗ trợ lập trình trên nhiều nền tảng ngôn ngữ khác nhau từ C/C++, C#, cho đến F#, Visual Basic, HTML, CSS, JavaScript.

Visual Studio giúp hỗ trợ khả năng gỡ rối (Debug) hiệu quả Visual Studio sở hữu giao diện thân thiện, dễ dàng sử dụng cho người mới bắt đầu, cho tích hợp nhiều ứng dụng khác như ứng dụng hỗ trợ quản lý và viết mã nhanh

* + - 1. Hướng dẫn cài đặt

1. Tải Visual Studio 2019 tại <https://visualstudio.microsoft.com/download>
2. Chọn bản Community
3. Click vào file vừa tải về và chờ Installer

Graphical user interface, text, application

Description automatically generated

Sau khi xong ta có được giao diện. chọn các tính năng để thực hiện dự án

Graphical user interface, text, application

Description automatically generated

Sau khi chọn xong thì click Install

Graphical user interface, application

Description automatically generated

Sau khi cài xong thì màn hình sẽ tự động xuất hiện. chúng ta có thể đăng nhập hoặc không

Graphical user interface, text, application, email

Description automatically generated

Và đây là màn hình khởi động của chương trình

Graphical user interface, application

Description automatically generated

CHƯƠNG II: PHÂN TÍCH THIẾT KẾ HỆ THỐNG WEBSITE

* 1. MÔ TẢ BÀI TOÁN, KHẢO SÁT, ĐƯA RA YÊU CẦU
     1. Mô tả bài toán

Tên bài toán: Xây dựng web đặt lịch khám sức khỏe.  
Mỗi ngày Phòng khám Toàn Mỹ tiếp đón và khám chữa bệnh cho nhiều bệnh nhân ở tất cả các chuyên khoa. Do đó phòng khám cũng đang phải đối mặt với tình trạng bệnh nhân phải xếp hàng chờ khám và đứng từ góc độ bệnh nhân thì việc này cũng gây ra không ít những phiền toái và lãng phí không đáng có.

Chính vì thế, Phòng khám Toàn Mỹ cho ra đời website đặt lịch khám chữa bệnh Booking Care nhằm mang đến giải pháp thuận tiện nhất cho nhu cầu khám, chữa bệnh, nâng cao tính chủ động và tiết kiệm được thời gian đi khám cho bệnh nhân.

Website booking Care giống như một kênh tương tác giữa bệnh viện và khách hàng. Thông qua Booking Care, bệnh nhân có thể đăng ký khám và đặt lịch hẹn khám mà không cần phải trực tiếp đến bệnh viện xếp hàng lấy số.

Booking Care còn tăng sự kết nối giữa bác sĩ và người bệnh, giúp bệnh nhân chủ động nắm được tình trạng sức khỏe bản thân thông qua việc kiểm tra lịch sử khám bệnh, hỏi đáp vấn đề bệnh lý đang gặp phải và sẽ được bác sĩ giải đáp ngay một cách nhanh chóng và chính xác nhất.

* + 1. Khảo sát yêu cầu

Khảo sát yêu cầu bằng phương pháp phỏng vấn

* **Chuẩn bị**
* Danh sách tham dự: Bác sĩ
* Địa điểm: Tại Phòng khám Toàn Mỹ ở Núi Thành Quảng Nam
* **Câu hỏi**

**\* Câu hỏi phỏng vấn bác sĩ**

Có bao nhiêu vai trò trong dự án khám chữa bệnh của phòng khám:

Bạn muốn dự án gồm các chức năng nào, có độ ưu tiên cho chức năng nào không?

Mục tiêu của dự án này là gì?

Bạn có yêu cầu gì về giao diện của hệ thống không, hay các sản phẩm tương tự

- Bạn có yêu cầu gì về logo hay ngôn ngữ của hệ thống hay không?

- Dự án này phát hành trên nền tảng gì website hay app?

- Thời gian phát hành của dự án là lúc nào :

- Kế hoạch chi tiết cho dự án?

- Chi phí của dự án?

- Giờ làm việc của bệnh viện như thế nào?

- Quản lý bệnh nhân theo danh mục hay không, thông tin bệnh nhân gồm gì?

- Có lưu trữ số lượng lớn dữ liệu hay không?

- Thông tin có lưu trữ dài hạn không?

- Có tra cứu thông tin dịch vụ, bác sĩ trong thời gian ngắn không?

* **Cách thức tiến hành**
* 2 người tham gia hỏi và ghi chép.
* Hỏi: Trần Thị Thảo Nguyên. Ghi chép: Phạm Thị Như Thủy.
* **Theo dõi sau phỏng vấn**

Có theo dõi sau phỏng vấn.

* **Lập báo cáo sau phỏng vấn**

**\*\*\* Báo cáo phỏng vấn \*\*\***

* **Địa điểm:** Tại Phòng khám Toàn Mỹ ở Núi Thành Quảng Nam
* **Người thực hiện:**
* Hỏi: Trần Thị Thảo Nguyên
* Ghi chép: Phạm Thị Như Thủy
* **Những người được phỏng vấn:**
* Bác sĩ
* **Ngày phỏng vấn:** ngày 10/11/2022
* **Mục tiêu:** thu nhập thông tin, xây dựng được mô hình hệ thống, tạo sưu liệu để xây dựng sưu liệu đặc tả.
* **Tóm tắt nội dung:**

**- Có bao nhiêu vai trò trong dự án(role)?**

-> Gồm 3 vai trò: người quản trị, bác sĩ, bệnh nhân.

**- Mục tiêu của dự án là gì?**

--> Giúp bệnh nhân có thể đặt lịch khám trực tuyến mà không cần đến bệnh viện để chờ trong thời gian dài.

- **Dự án gồm những chức năng gì, có độ ưu tiên cho các chức năng không?**

--> Đăng nhập, đăng kí, quên mật khẩu,… các chức năng không thể thiếu của một hệ thống. Xác thực thông tin đăng nhập qua email, số điện thoại nhằm kiểm tra tính tồn tại của dữ liệu. Xem thông tin các phòng khám, đánh giá, bình luận từ những người đã sử dụng dịch vụ,…

**- Giờ làm việc của phòng khám như thế nào?**

-> Ngày làm việc gồm 03 ca: mỗi ca làm việc 8 giờ. Ngày làm việc gồm ca: một ca làm việc 8 giờ theo giờ hành chính và một ca làm việc 16 giờ hoặc mỗ ca làm việc 12 giờ

**- Quản lý bệnh nhân theo danh mục hay không, thông tin bệnh nhân gồm gì?**

-> Quản lý bệnh nhân theo ngày khám, khoa khám, bác sĩ điều trị,...

**- Có lưu trữ số lượng lớn dữ liệu hay không?**

-> Có lưu trữ số lượng lớn dữ liệu.

**- Hồ sơ bệnh nhân có lưu trữ dài hạn không?**

-> Hồ sơ bệnh nhân nội trú, ngoại trú được lưu trữ ít nhất 10 năm. Hồ sơ bệnh nhân tai nạn lao động, tai nạn sinh hoạt được lưu trữ ít nhất 15 năm. Hồ sơ bệnh nhân đối với người tâm thần, người bệnh tử vong được lưu trữ ít nhất 20 năm.

**- Có tra cứu thông tin dịch vụ, bác sĩ trong thời gian ngắn không?**

-> Bệnh nhân được phép tra cứu thông tin dịch vụ, bác sĩ trong thời gian ngắn.

**Bạn có yêu cầu gì về giao diện của hệ thống không, hay các sản phẩm tương tự**

**->** Giao diện thân thiện với người dùng, dễ sử dụng,..

**- Bạn có yêu cầu gì về logo hay ngôn ngữ của hệ thống hay không?**

**-->** logo theo thiết kế mà phòng khám yêu cầu , ngôn ngữ tiếng việt dễ hiểu,..

**-Dự án này phát hành trên nền tảng gì website hay app?**

--> nền tảng website thích hợp sử dụng trên điện thoại, máy tính,..

-**Thời gian phát hành của dự án là lúc nào:**

-> 3 tháng kể từ ngày bắt đầu

**- Kế hoạch chi tiết cho dự án?**

-> không có yêu cầu, tuỳ thuộc vào đội ngũ hoàn thành dự án

**- Chi phí của dự án?**

-> 300.000.000 VNĐ.

**\* Kết quả thu được gồm các vấn đề**

**1. User**

**2. Business**

**3. Functions**

**4. Information DB**

**5. Local/ Internet**

**6. Time Release**

**7. Budget**

**8. Same product**

**9. Product Management**

**10. UI/ logo language**

**11. Plan**

**12. Cost**

* + 1. Yêu cầu bài toán
       1. Đối với bác sĩ
* Xem thông tin bác sĩ
* Xem lịch khám
* Xem quản lí kế hoạch khám chữa bệnh
* Xem thông tin đặt lịch khám chữa bệnh
  + - 1. Đối với người dùng của trang web
* Được đăng ký tài khoản, đăng nhập trước khi đặt lịch.
* Được xem các thông tin về bác sĩ,lịch khám, các đánh giá và bình luận của những người dùng khác và các thông tin liên quan đến trang web.
* Được quyền đặt lịch khám .
* Được quyền đánh giá và bình luận khi sau khi đã sử dụng dịch vụ.
* Thanh toán 1 phần xem như đặt cọc và thanh toán hoàn toàn
* Được xem thông tin hóa đơn sau khi đặt lịch khám.
  + - 1. Đối với người quản trị trang web (admin)
* Được đăng nhập để vào trang quản trị .
* Được quản lý người dùng, sửa quyền của người dùng, xóa người dùng.
* Được quản lý bác sĩ, sửa quyền của bác sĩ, xóa bác sĩ .
* Được quản lý lịch khám , thêm lịch khám mới, xóa lịch khám không phù hợp.
* Được quản lý đơn đặt lịch, xem thông tin lịch khám đã đặt, xóa lịch khám.
  1. PHÂN TÍCH HỆ THỐNG
     1. Mô tả hệ thống

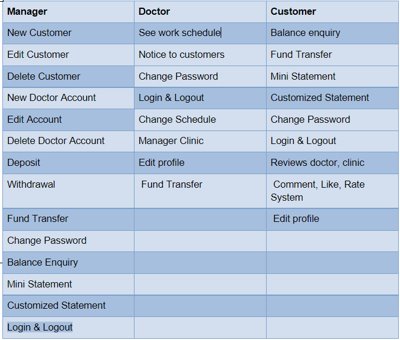
Hệ thống đặt lịch khám sức khỏe sẽ có 1 website người dùng và 1 website quản lí

Website người dùng mọi người có thể truy cập như 1 khách hàng xem các thông tin về bác sĩ, lịch khám, lịch sử đặt lịch .Sử dụng công nghệ ReactJS, NodeJS và lưu trữ trong MySQL , khi đặt lịch hệ thống sẽ gửi thông báo email tới người sử dụng xác nhận đặt lịch, sử dụng API.

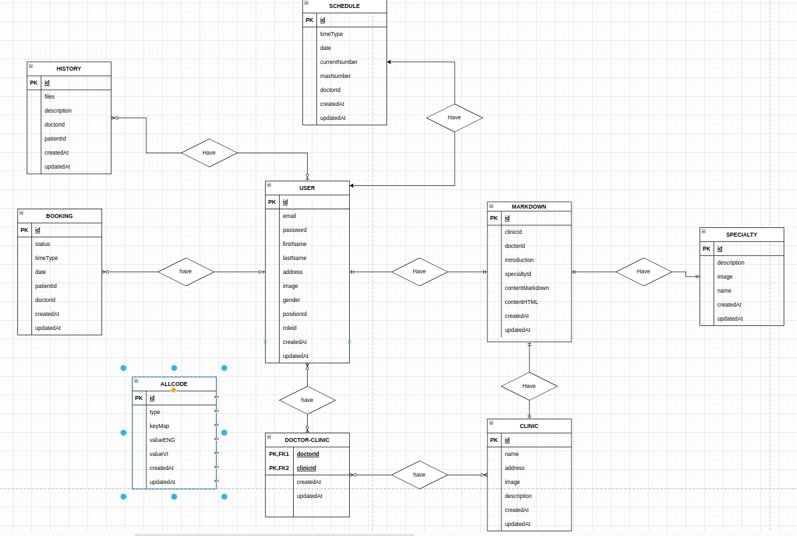
Website quản lí cho bác sĩ và Admin truy cập , bác sĩ xem thông tin bác sĩ xem lịch khám quản lí kế hoạch khám chữa bệnh ,xem thông tin đặt lịch khám chữa bệnh. Admin quản lí tất cả thông tin tài khoản, thông tin lịch khám, đơn đặt lịch cũng như lịch sử đặt lịch của khách hàng. Sử dụng công nghệ ReactJS, NodeJS và lưu trữ trong MySQL, sử dụng API.

* + 1. Kết quả Phân tích hệ thống

Kết quả của quá trình phân tích và thiết kế là các chức năng của hệ thống, các ràng buộc và tài liệu đặc tả

1. 

Hình 2.: Chức năng của hệ thống



Hình 2.: Mô hình ER

* 1. THIẾT KẾ HỆ THỐNG( SỬ DỤNG C4 MODEL)
     1. Context diagram

Diagram

Description automatically generated

Hình 2.:[System Context] Booking System

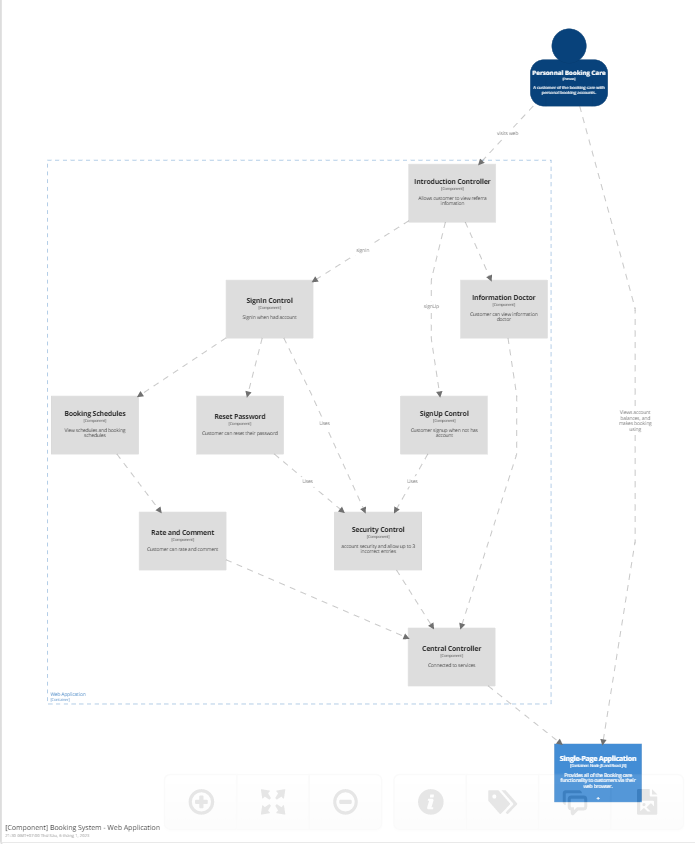
* + 1. Container diagram

Diagram

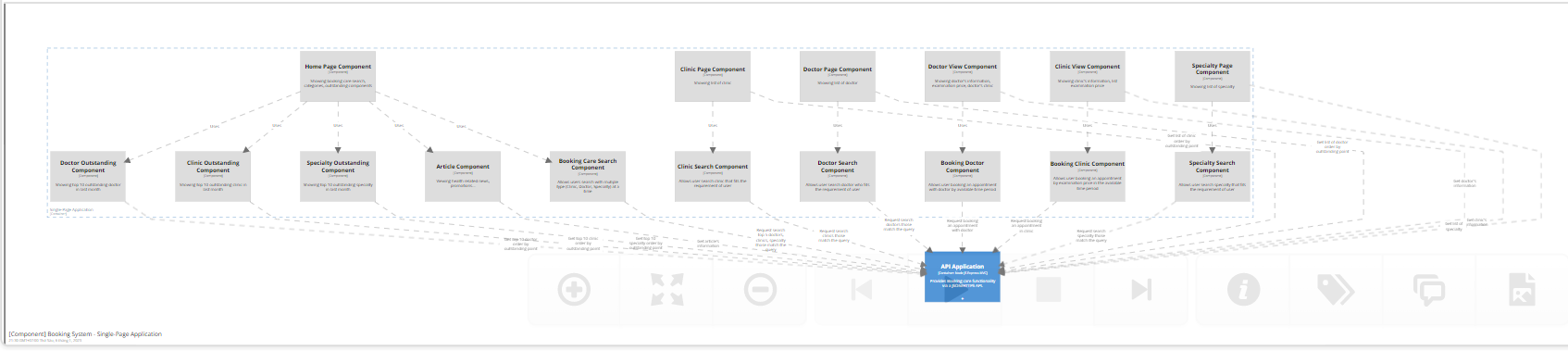
Description automatically generated with medium confidence

Hình 2.: [Container] Booking System

* + 1. Component diagram

****

Hình 2.: [Component] Booking system - Web Application

****

Hình 2.: [Conponent] Booking system - SinglePageApplication

**A picture containing graphical user interface

Description automatically generated**

Hình 2.: [Conponent] Booking system – API Application

Graphical user interface, diagram, application

Description automatically generated

Hình 2.: [Conponent] Booking system – API Application (#SignIn)

Diagram

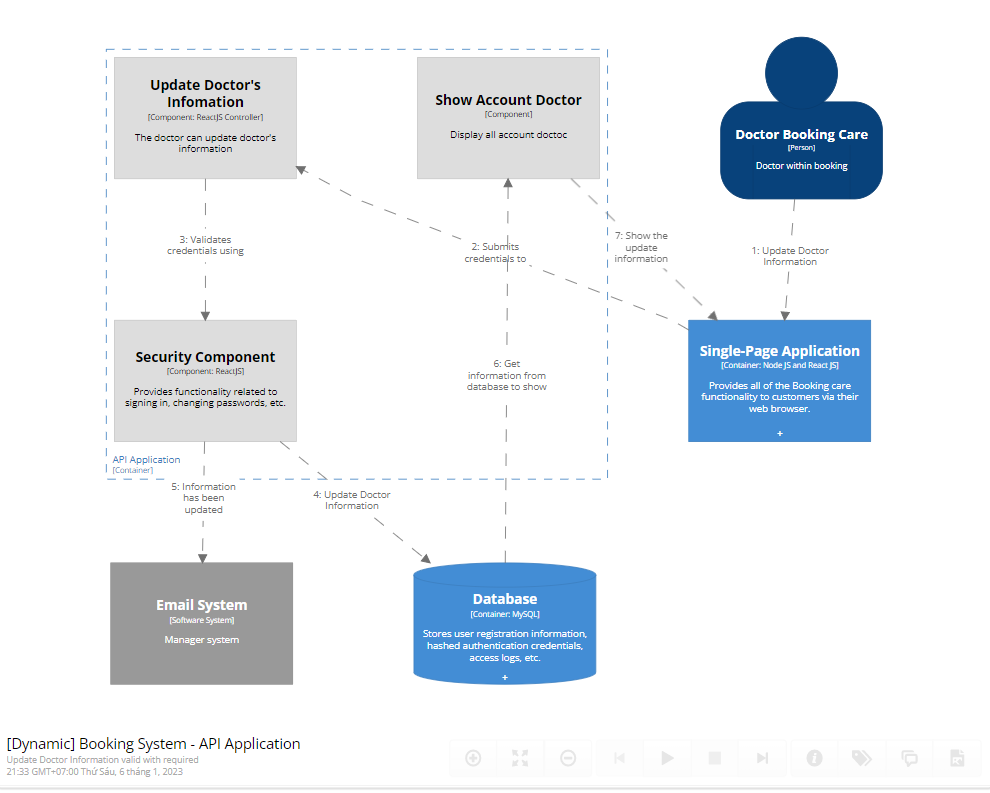
Description automatically generated

Hình 2.: [Conponent] Booking system – API Application (#DoctorsInformation)

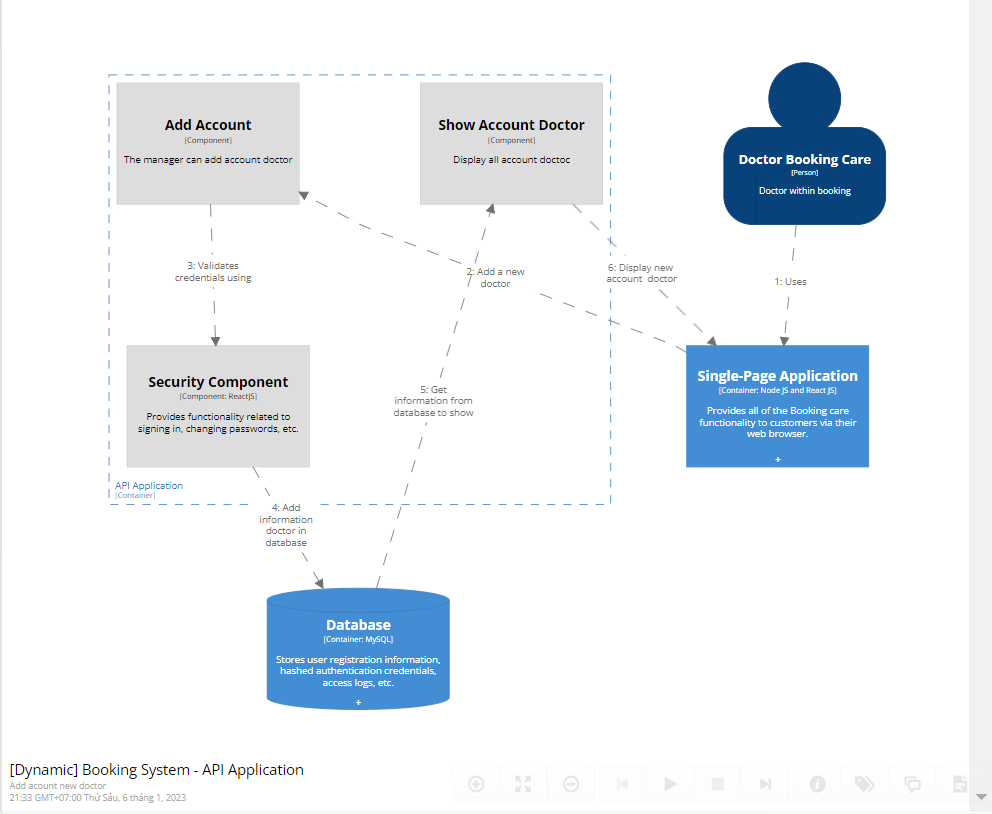
Diagram

Description automatically generated

Hình 2.: [Conponent] Booking system – API Application (#ManagementAccount)



Hình 2.: [Conponent] Booking system – API Application (#UpdateDoctorInformation)



Hình 2.: [Conponent] Booking system – API Application (#AddAccount)

Diagram

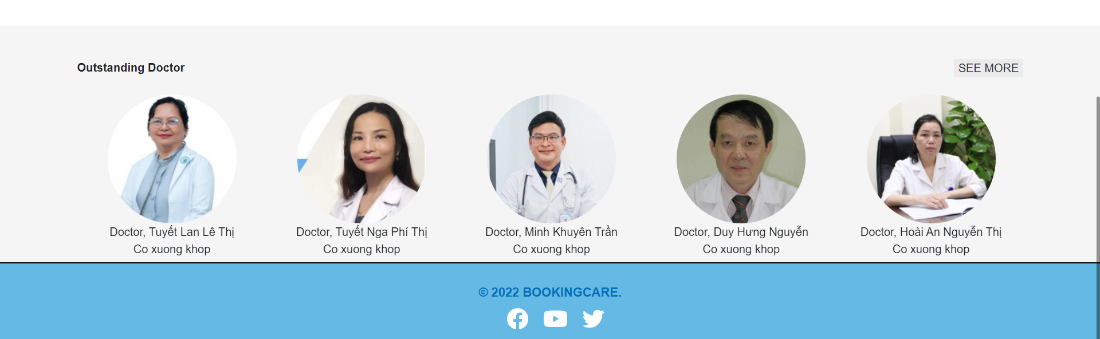
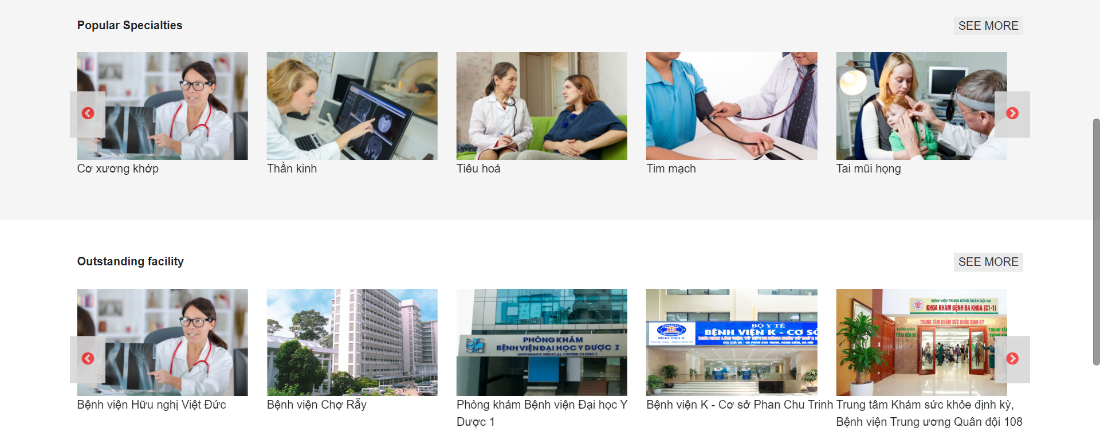
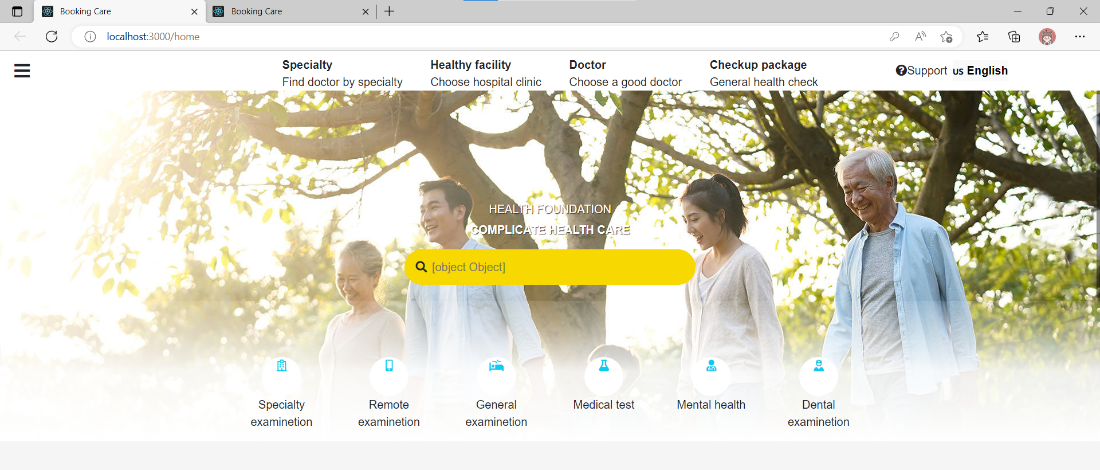
Description automatically generated

Hình 2.: [Conponent] Booking system – API Application (#Booking)

CHƯƠNG III: XÂY DỰNG HỆ THỐNG WEBSITE

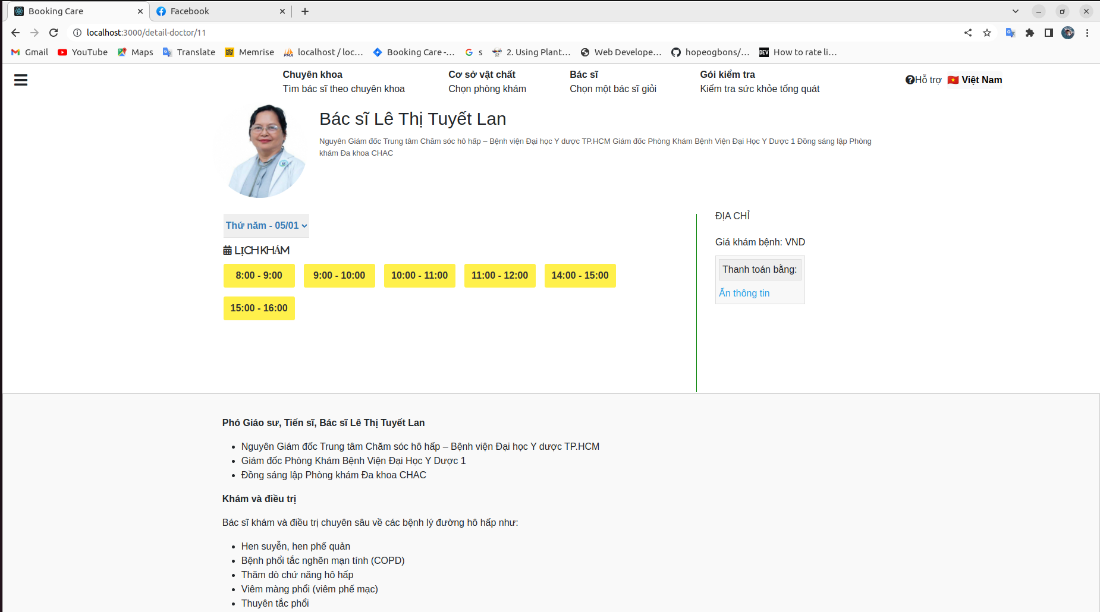
* 1. GIAO DIỆN HIỂN THỊ
     1. Giao diện trang chủ

- Trang chủ: Hiển thị thông tin website, danh sách menu, các chức năng dành cho người sử dụng.



Hình 3.: Giao diện trang chủ

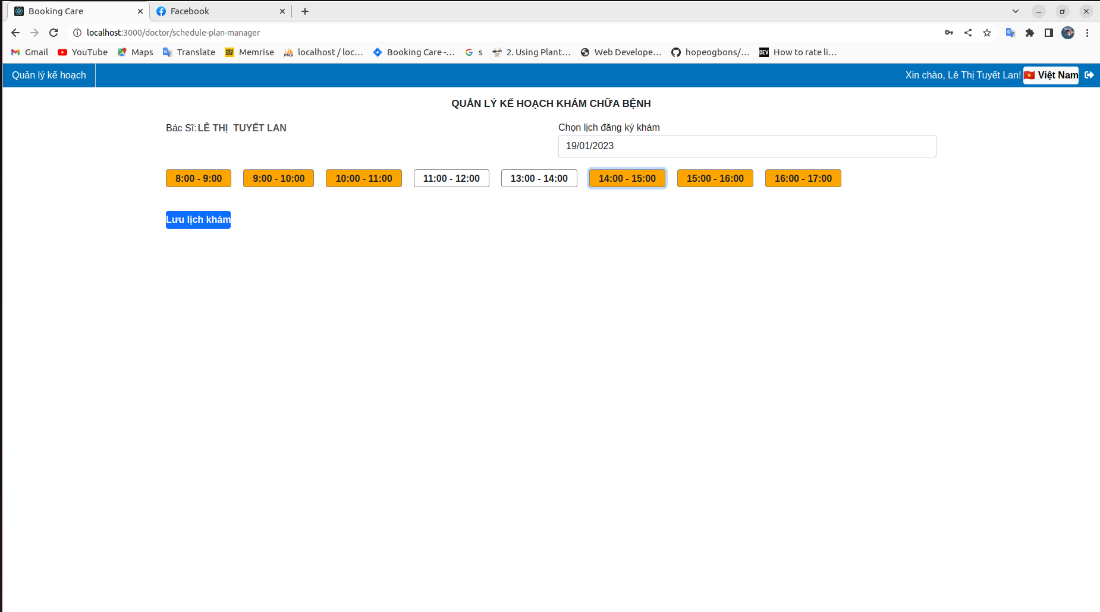
* + 1. Giao diện thông tin bác sĩ

1. - Chức năng thông tin bác sĩ: người dùng có thể xem thông tin của bác sĩ.
2. 

Hình 3.: Giao diện thông tin bác sĩ

* + 1. Giao diện quản lý kế hoạch khám chữa bệnh của bác sĩ

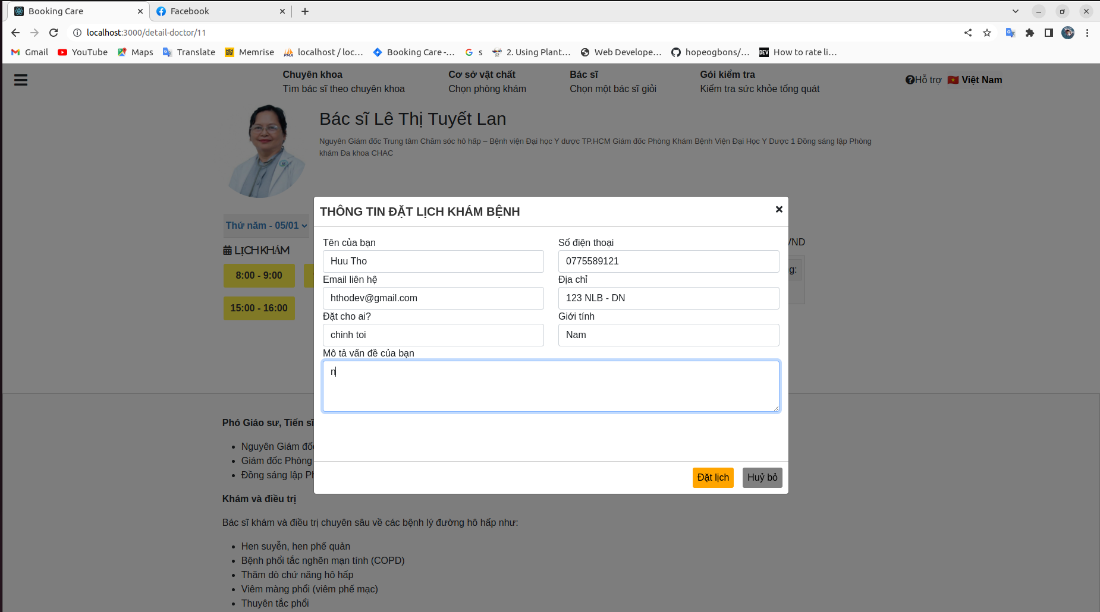
- Chức năng quản lý kế hoạch khám chữa bệnh của bác sĩ: Người dùng có thể xem kế hoạch khám chữa bệnh của bác sĩ.



Hình 3.: Giao diện quản lý kế hoạch khám chữa bệnh của bác sĩ

* + 1. Giao diện thông tin đặt lịch khám bệnh

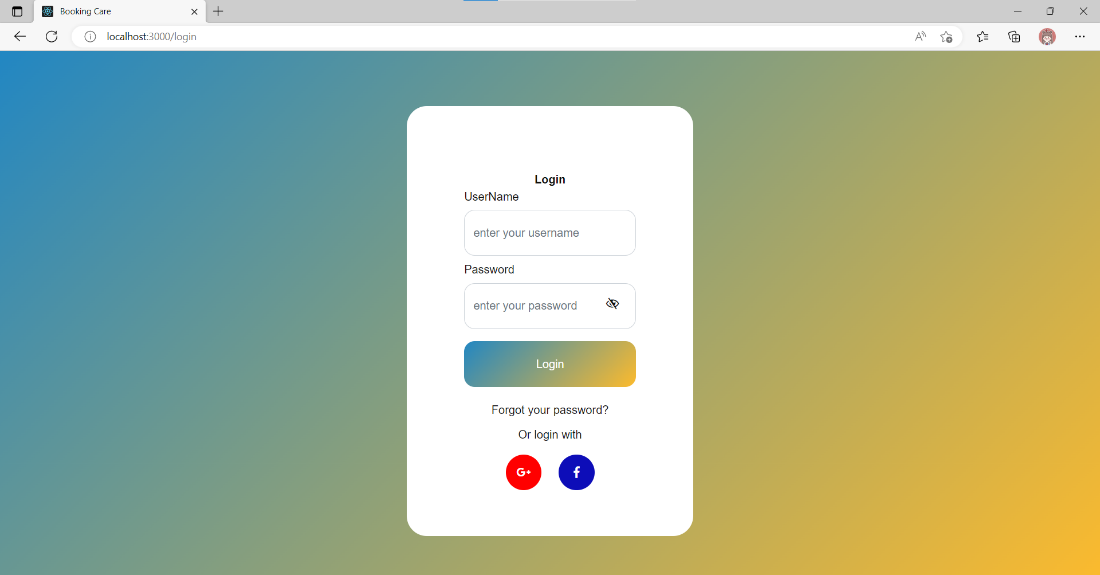
- Chức năng thông tin đặt lịch khám bệnh: Người dùng có thể đặt lịch khám bệnh.



Hình 3.: Giao diện thông tin đặt lịch khám bệnh

* + 1. Giao diện Login

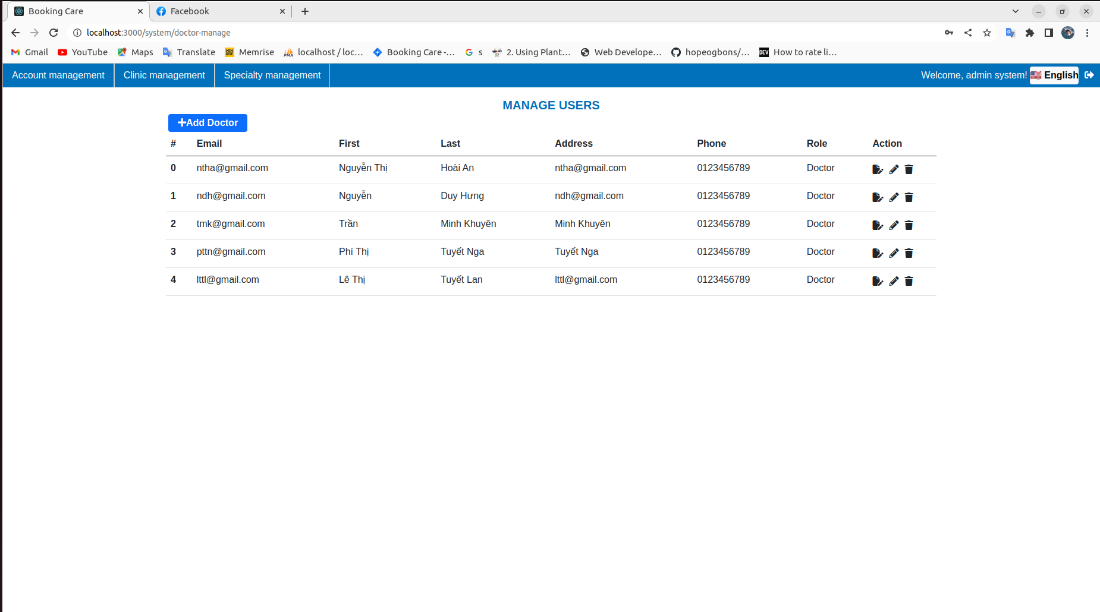
- Chức năng Login: người dùng điền thông tin đã đăng ký để đăng nhập vào website.



Hình 3.: Giao diện Login

* 1. GIAO DIỆN QUẢN LÝ
     1. Giao diện quản lý thông tin người dùng

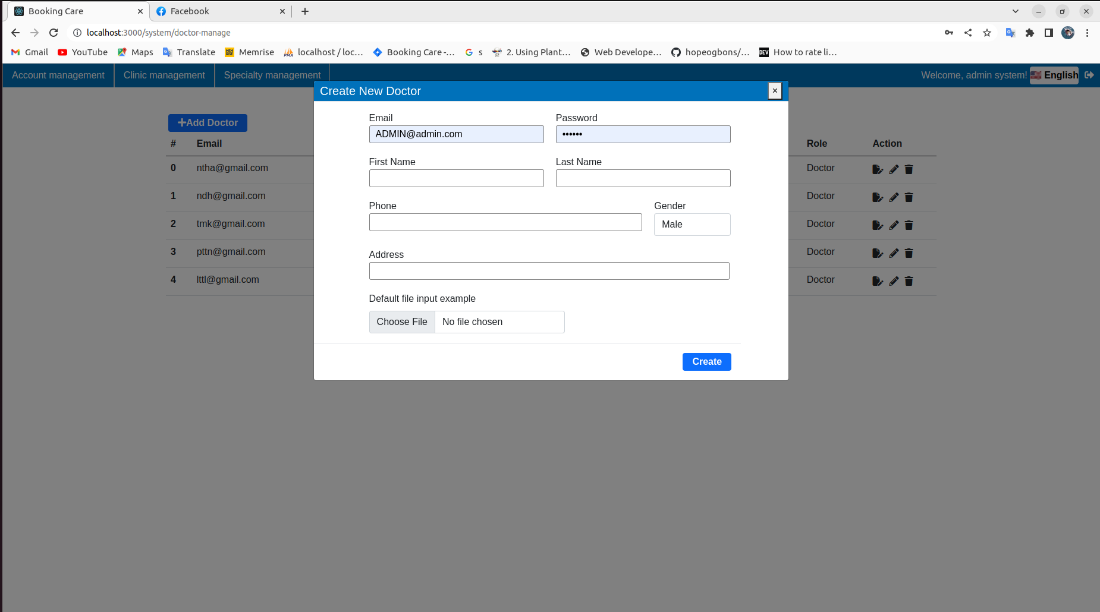
- Chức năng quản lý thông tin người dùng: Người quản trị xem được thông tin người dùng.



Hình 3.: Giao diện quản lý thông tin bác sĩ

* + 1. Giao diện tạo mới thông tin bác sĩ

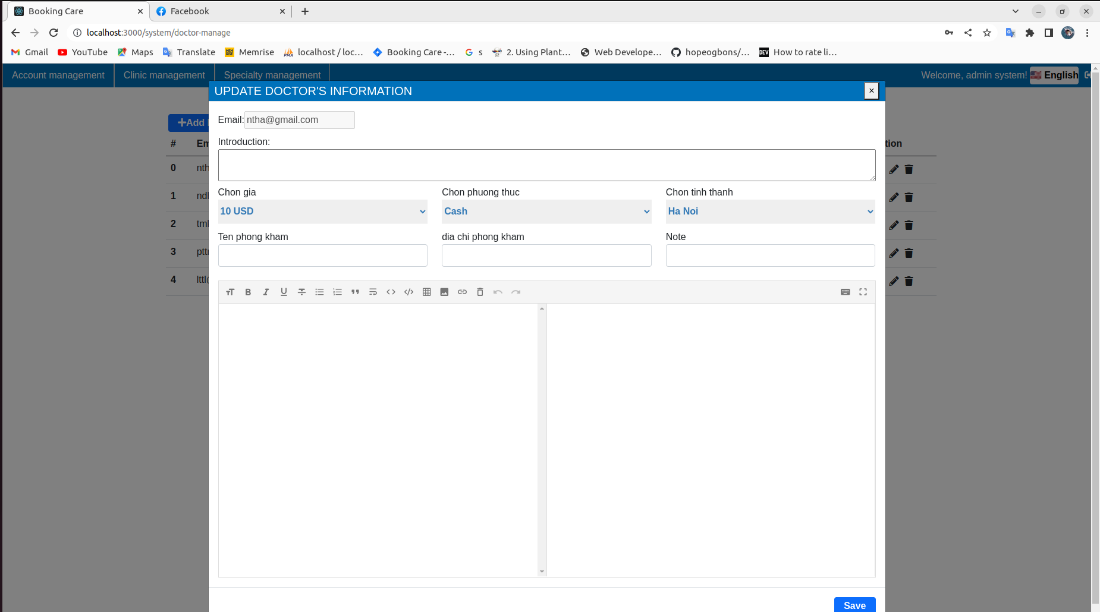
- Chức năng tạo mới thông tin bác sĩ: Người quản trị được tạo mới thông tin bác sĩ.



Hình 3.: Giao diện tạo mới thông tin bác sĩ

* + 1. Giao diện cập nhật thông tin tài khoản bác sĩ

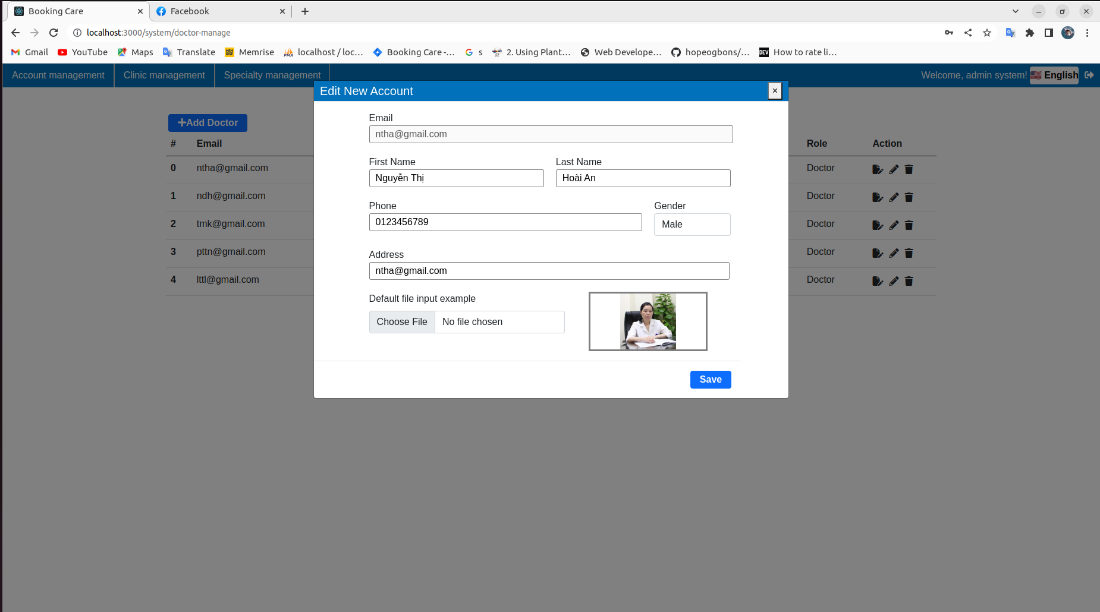
- Chức năng cập nhật thông tin tài khoản bác sĩ: Người quản trị được cập nhật thông tin tài khoản bác sĩ



Hình 3.: Giao diện chỉnh sửa thông tin tài khoản người dùng

* + 1. Giao diện chỉnh sửa thông tin tài khoản người dùng

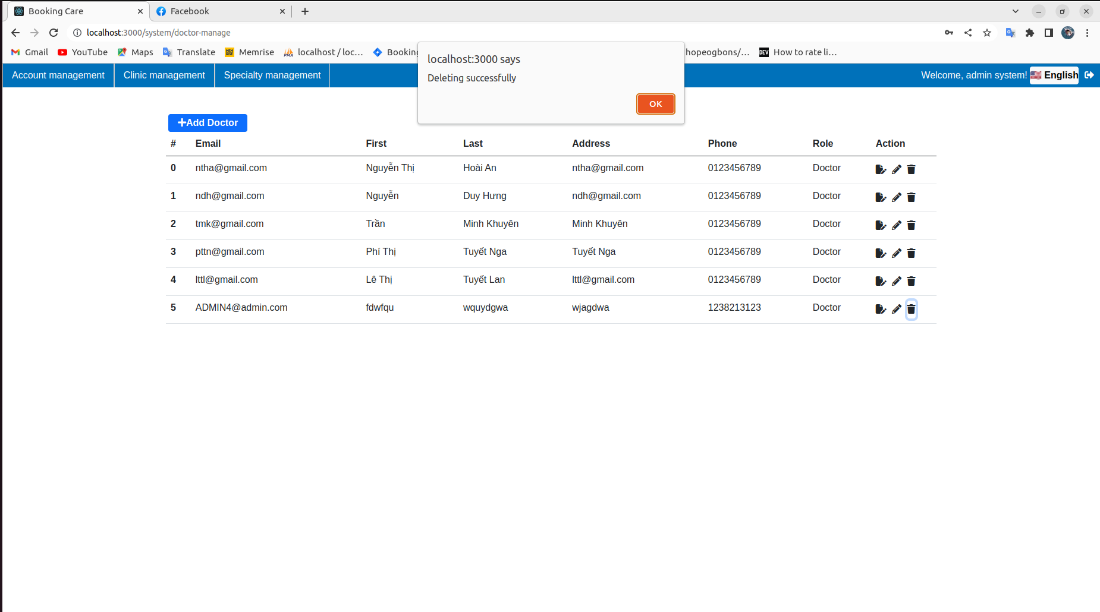
- Chức năng chỉnh sửa thông tin tài khoản bác sĩ: Người quản trị được chỉnh sửa thông tin tài khoản người dùng



Hình 3.: Giao diện chỉnh sửa thông tin tài khoản người dùng

* + 1. Giao diện xóa thông tin tài khoản người dùng

- Chức năng xóa thông tin tài khoản người dùng: Người quản trị được xóa thông tin tài khoản người dùng



Hình 3.: Giao diện xóa thông tin tài khoản người dùng

KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

* **Kết luận**
* **Những công việc đã làm được**
* Nắm được các kiến thức về thiết kế và xây dựng website đặt lịch khám sức khỏe bằng HTML, CSS, ReactJS, NodeJS, MySQL …
* Xây dựng được đa phần các chức năng cần thiết cho một trang web.
* Thiết kế website thân thiện, đẹp mắt cho người sử dụng, khách hàng dễ dàng lựa chọn được những sản phẩm theo yêu cầu.
* Xuất được đơn hàng đã mua khi khách hàng thực hiện giao dịch.
* **Hạn chế**
* Vẫn còn một số hạn chế, sai sót khi phân tích các chức năng của hệ thống, tuy nhiên nhóm đã cố gắng để hoàn thiện.
* Vẫn còn một số hạn chế, sai sót khi phân tích, thiết kế các biểu mẫu của hệ thống, tuy nhiên nhóm đã cố gắng để hoàn thiện.
* Vẫn còn gặp một số khó khăn khi đưa ra các giải pháp cho hệ thống, tuy nhiên nhóm đã cố gắng để hoàn thiện.
* Vẫn còn gặp một số khó khăn khi hoạt động nhóm, tuy nhiên nhóm đã cố gắng để hoàn thiện.
* **Hướng phát triển**

Xây dựng thêm các chức năng còn thiếu:

- Xây dựng giao diện đẹp mắt hơn, sinh động hơn, cơ sở dữ liệu nhất quán hơn.

- Tích hợp hệ thống thanh toán online vào dự án.

- Xây dựng chức năng phân loại người dùng nhằm đem lại quyền lợi tốt nhất cho người dùng.

TÀI LIỆU THAM KHẢO

[1] Bài giảng trên lớp môn Phân tích và thiết kế hướng đối tương

[2]

<https://www.vuemastery.com/pdf/Vue-Essentials-CheatSheet.pdf?fbclid=IwAR0oEGoihOJ2IVjlDoLot5SHSxOdLdImKYS7LnDgyNc4GNxJJ9KC4_kukM4>

[3] <https://vuejs.org/v2/guide/?fbclid=IwAR2xB1OuV7jxaZS7bVur77oiwuOQZpjZBTVf-_xgOuJ9KfF3Jmzt69OEcdw>