

**BỘ GIÁO DỤC VÀ ĐÀO TẠO**

**TRƯỜNG ĐẠI HỌC SƯ PHẠM KỸ THUẬT TP. HỒ CHÍ MINH**

**KHOA ĐÀO TẠO CHẤT LƯỢNG CAO**

**🕯✡🕮🕮✡🕯**



**KHÓA LUẬN TỐT NGHIỆP**

**XÂY DỰNG WEBSITE BÁN SÁCH**

**TRÊN NỀN TẢNG MEAN STACK**

**GVHD: ThS. LÊ VĨNH THỊNH**

**TRƯƠNG THÁI TUẤN MSSV : 16110249**

**NGUYỄN ĐỨC HẬU MSSV : 16110069**

**KHÓA 2016**

**NGÀNH CÔNG NGHỆ THÔNG TIN**

Hồ Chí Minh , Ngày 15 tháng 03 năm 2020

|  |  |
| --- | --- |
|  | **CỘNG HÒA XÃ HỘI CHỦ NGHĨA VIỆT NAM**  **Độc lập – Tự do – Hạnh phúc**  **\*\*\*\*\*\*\*** |

Tp. Hồ Chí Minh, ngày 09 tháng 12 năm 2020

NHIỆM VỤ KHÓA LUẬN TỐT NGHIỆP

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Họ và tên sinh viên: Nguyễn Đức Hậu  Họ và tên sinh viên: Trương Thái Tuấn | | | MSSV: 16110069  MSSV: 16110249 | |
| Ngành: Công Nghệ Thông Tin | | | Lớp: 16110ST4 | |
| Giảng viên hướng dẫn: ThS Lê Vĩnh Thịnh | | | ĐT: 0938 252 222 | |
| Ngày nhận đề tài: 24/02/2020 | | | Ngày nộp đề tài: | |
| 1. Tên đề tài: Xây dựng website bán và quản lý sách sử dụng MEAN STACK | | | | |
| 2. Các số liệu, tài liệu ban đầu: Không có | | |  | |
| 3. Nội dung thực hiện đề tài:  - Tìm hiểu về MEAN STACK  - Xây dựng website demo | | |  | |
| 4. Sản phẩm: Source code website bán sách | | |  | |
| TRƯỞNG NGÀNH  Nguyễn Đăng Quang | | | GIẢNG VIÊN HƯỚNG DẪN  Lê Vĩnh Thịnh | |
|  | **CỘNG HÒA XÃ HỘI CHỦ NGHĨA VIỆT NAM**  **Độc lập – Tự do – Hạnh phúc**  **\*\*\*\*\*\*\*** | |

**PHIẾU NHẬN XÉT CỦA GIÁO VIÊN HƯỚNG DẪN**

Họ và tên Sinh viên 1: Trương Thái Tuấn.........................MSSV 1: 16110249  
Họ và tên Sinh viên 2: Nguyễn Đức Hậu...........................MSSV 2: 16110069  
Ngành: Công nghệ Thông tin  
Tên đề tài: XÂY DỰNG WEBSITE BÁN SÁCH TRÊN NỀN TẢNG MEAN STACK.  
Họ và tên Giáo viên hướng dẫn: ………...ThS. LÊ VĨNH THỊNH...............................   
**NHẬN XÉT**1. Về nội dung đề tài và khối lượng thực hiện:  
................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................

2. Ưu điểm:  
................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................

3. Khuyết điểm:  
............................................................................................................................................................................................................................................................................................................................................................................................................................................................

4. Đề nghị cho bảo vệ hay không ?

5. Đánh giá loại:

6. Điểm:

Tp*. Hồ Chí Minh, ngày tháng năm 2020*  
Giáo viên hướng dẫn  
*(Ký & ghi rõ họ tên)*

|  |  |
| --- | --- |
|  | **CỘNG HÒA XÃ HỘI CHỦ NGHĨA VIỆT NAM**  **Độc lập – Tự do – Hạnh phúc**  **\*\*\*\*\*\*\*** |

**PHIẾU NHẬN XÉT CỦA GIÁO VIÊN PHẢN BIỆN**

Họ và tên Sinh viên 1: Trương Thái Tuấn.........................MSSV 1: 16110249  
Họ và tên Sinh viên 2: Nguyễn Đức Hậu...........................MSSV 2: 16110069  
Ngành: Công nghệ Thông tin  
Tên đề tài: XÂY DỰNG WEBSITE BÁN SÁCH TRÊN NỀN TẢNG MEAN STACK.  
Họ và tên Giáo viên phản biện: ……….....................................................................................   
**NHẬN XÉT**1. Về nội dung đề tài và khối lượng thực hiện:  
................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................

2. Ưu điểm:  
................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................

3. Khuyết điểm:  
............................................................................................................................................................................................................................................................................................................................................................................................................................................................

4. Đề nghị cho bảo vệ hay không ?

5. Đánh giá loại:

6. Điểm:

Tp*. Hồ Chí Minh, ngày tháng năm 2020*  
Giáo viên hướng dẫn  
*(Ký & ghi rõ họ tên)*

# **LỜI CẢM ƠN**

Lời đầu tiên, cho phép nhóm em được gửi lời cảm ơn chân thành đến thầy Lê Vĩnh Thịnh, người đã phụ trách hướng dẫn nhóm chúng em trong việc thực hiện và hoàn thành bài báo cáo một cách tốt và hiệu quả nhất. Những sự chỉ dạy, nhật xét, đóng góp ý kiến từ thầy đã góp phần rất quan trọng trong việc giúp chúng em hoàn hành đề tài một cách hoàn thiện nhất.

Cảm ơn các thầy cô trong khoa Công Nghệ Thông Tin đã luôn nhiệt tình, cống hiến, tận tình giải đáp các thắc mắc của chúng em. Bên cạnh đó, chúng em xin được cảm ơn các bạn cùng khóa, các anh chị khóa trên đã không ngần ngại chia sẻ các thông tin, kinh nghiệm bổ ích giúp chúng em hoàn thiện hơn đề tài của mình.

Báo cáo được thực hiện trong một khoảng thời gian có hạn, cùng với những hạn chế về mặt kiến thức nên việc sai sót là điều không thể tránh khỏi trong quá trình thực hiện. Chúng em kính mong nhận được những lời nhận xét, ý kiến đóng góp quý báo từ quý Thầy Cô để chúng em có thêm kinh nghiệm và hoàn thành tốt hơn những đề tài sau này.

***Chúng em xin chân thành cảm ơn !***

# **PHẦN 1 : MỞ ĐẦU**

**1.1. TÍNH CẤP THIẾT CỦA ĐỀ TÀI**

Sách là nguồn tri thức vô giá, là bàn đạp để phát triển nhân loại. Còn thương mại điện tử tại VN hiện nay đang ngày càng trở nên phổ biến - người người, nhà nhà ai cũng có smart phone, một click là có hàng. Kết hợp sách với thương mại điện tử, với mục đích giúp việc mua sách trở nên đơn giản hơn nữa, để những quyển sách quý sách hay dễ dàng đến tay người đọc và cũng vì đây là một đề tài thực tiễn, nhóm tụi em thử sức xây dựng một trang web bán sách online.

**1.2. MỤC ĐÍCH CỦA ĐỀ TÀI**

Đề tài “KHÓA LUẬN TỐT NGHIỆP XÂY DỰNG WEBSITE BÁN SÁCH TRÊN NỀN TẢNG MEAN STACK” đặt ra 2 vấn đề trọng tâm về mục tiêu mà sinh viên nghiên cứu cần thực hiện được bao gồm:

- Vấn đề đầu tiên đưa ra về yếu tố lý thuyết, sinh viên thực hiện cần có kết quả nghiên cứu chuyên môn và hiểu được tổng quan và khả năng ứng dụng của từng thành phần trong mô hình MEAN nói chung. Với từng thành phần của MEAN Stack hiểu xác định được các đặc trưng về nguyên tắc hoạt động, thành phần hoặc mô hình. Thư viện được xây dựng dựa trên ngôn ngữ lập trình Javascript đạt được sự phổ biến đến lập trình viên như: NodeJS – xây dựng Server, Angular – xây dựng ứng dụng phía người dùng, MongoDB – cơ sở dữ liệu, ExpressJS – middleware hỗ trợ xây dựng Server. Song song với việc hiểu về 4 thành phần tiêu biểu của MEAN Stack sinh viên cũng cần phải có thêm tìm hiểu về các thư viện, mã nguồn module giúp ích trong quá trình xây dựng ứng dụng Web thực tế.

- Vấn đề thứ hai của đề tài đặt ra về ứng dụng mô hình MEAN Stack vào xây dựng một sản phẩm cụ thể, ở đây là “Website bán sách”. Sản phẩm hoàn thành phải là một ứng dụng Web được xây dựng có đầy đủ các thành phần của mô hình MEAN Stack và đáp ứng thêm các yêu cầu về ứng dụng mua, lựa chọn sách yêu thích, tracking sản phẩm trong quá trình vận chuyển và một số tính năng khác có thể áp dụng trong thực tế cũng như khả năng nâng cấp hoặc tiếp tục phát triển tính năng trong tương lai.

**1.3. ĐỐI TƯỢNG VÀ PHẠM VI NGHIÊN CỨU**

Đề tài được thực hiện nghiên cứu xoay quanh hai nhóm đối tượng trọng tâm bao gồm: nhóm đối tượng về kiến thức công nghệ và nhóm đối tượng có nhu cầu mua, sử dụng các tựa sách mà họ yêu thích.

Trong đó nhóm đối tượng về kiến thức công nghệ bao gồm 4 đối tượng nghiên cứu bắt buộc là: Mã nguồn mở NodeJS, framework Angular, cơ sở dữ liệu MongoDB, thư viện ExpressJS. Kết hợp thêm trong nhóm đối tượng nghiên cứu về kiến thức công nghệ là một số thư viện Javascript có giá trị ứng dụng như JWT (Json Web Token), charts cũng cần được nghiên cứu cho các ứng dụng về doanh thu cũng như mức độ ưa thích của sản phẩm.

Nhóm đối tượng có nhu cầu mua, sử dụng các tựa sách mà họ yêu thích bao gồm kiến thức về tên sách, thể loại, tựa seri sách cũng như các đánh giá về trải nghiệm người dùng trên một trang web bán sách cụ thể.

Phạm vi nghiên cứu đề ra ở mức độ tổng quát, người thực hiện đề tài hiểu được các kiến thức chung của nội dung nghiên cứu và áp dụng được từng nội dung kiến thức vào sản phẩm thực tế, không đặt nặng các lý thuyết nặng nề không mang tính ứng dụng.

**1.4. KHẢO SÁT HIỆN TRẠNG**

Với sự cạnh tranh của các trang website bán sách qua mạng ngày càng lớn, các trang web thay đổi về chức năng và cả giao diện để đáp ứng nhu cầu mua sách của người dùng. Nên các website liên quan đến sách hiện nay phải đáp ứng được về cả giao diện đẹp, dễ tìm được sách mà người dùng cần và chức năng dễ sử dụng, tiện lợi.

Nhiều tính năng được thêm vào để đem lại cho người dùng những trải nghiệm mới như hình thức quay số trúng mã giảm giá hoặc các phần thưởng có giá trị, các khuyến mãi được diễn ra liên tục hàng ngày để lôi cuốn người dùng đến với website nhiều hơn.

Ngoài hình thức mua qua trả tiền mặt thì nay các website còn tích hợp với thanh toán bằng Paypal nhằm đưa sản phẩm đến gần với người dùng hơn.

# **PHẦN 2: PHẦN NỘI DUNG**

## **CHƯƠNG 1: MÔ HÌNH MEAN STACK**

* 1. **Tổng quan**
     1. ***Khái niệm về MEAN Stack***

Khái niệm MEAN stack được nhắc đến lần đầu tiên bởi Valeri Karpov - một lập trình viên làm việc cho MongoDB - vào năm 2013.

MEAN là một mô hình phát triển website trên nền tảng JavaScript để xây dựng các trang web động. Mô hình này cung cấp cho lập trình viên các thành phần đã đóng gói sẵn để phát triển những website được nhanh chóng.

Một lợi ích lớn từ MEAN Stack là nó được xây dựng dựa trên một ngôn ngữ lập trình duy nhất - JavaScript, mà vẫn đáp ứng được hoạt động của đầy đủ thành phần trong mô hình, điều này khiến nó hoạt động hiệu quả và dễ tiếp cận trong quá trình phát triển website.

* + 1. ***Các thành phần trong MEAN stack***

MEAN đại điện cho MongoDB (M), ExpressJS (E), Angular (A) và NodeJS(N).

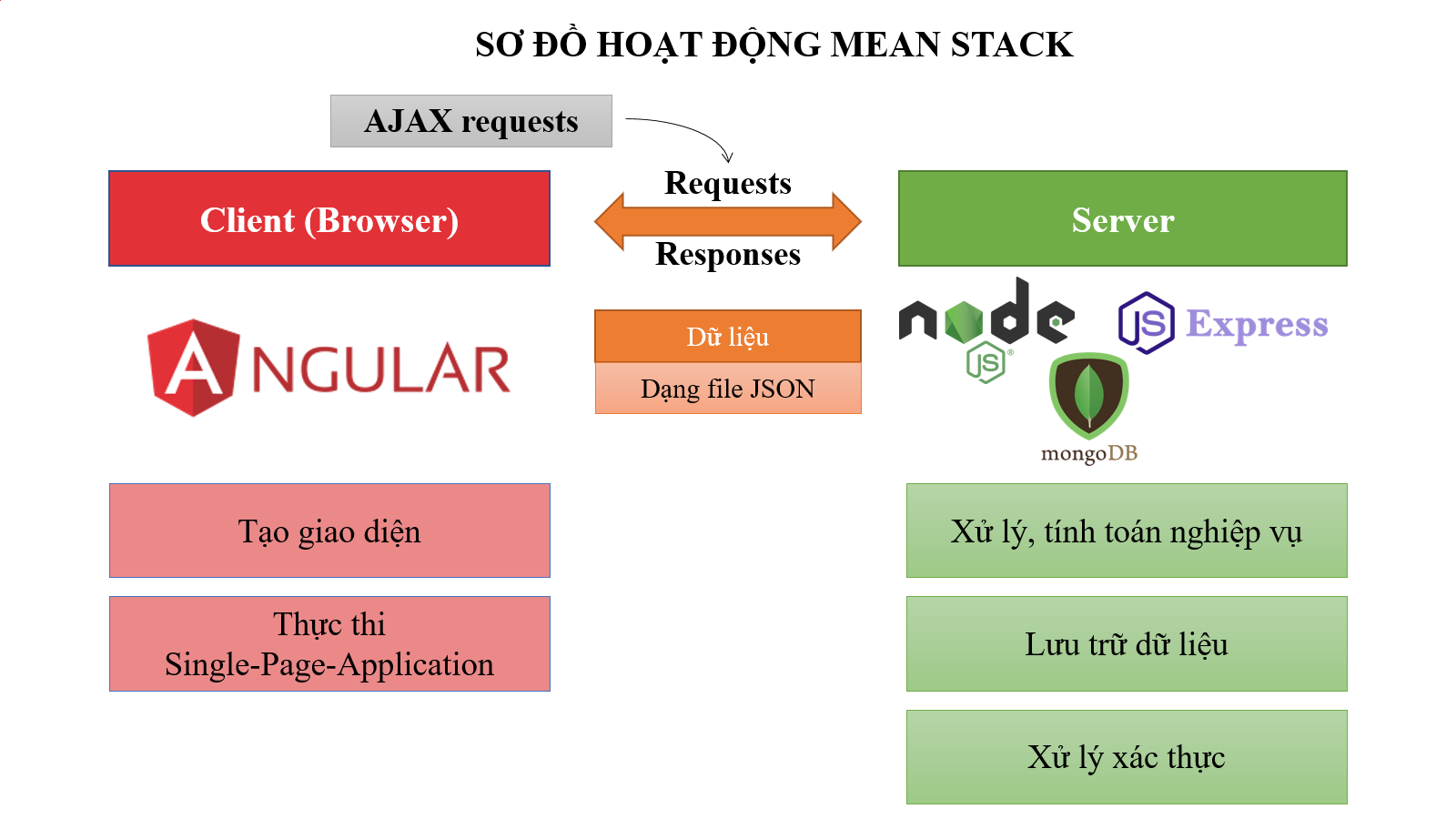
Angular là một framework xây dựng dựa trên JavaScript chạy trên browser của người dùng cho phép lập trình viên xây dựng Single-Page-Applications (SPA). Nhiệm vụ của Angular là xử lý và xuất ra màn hình giao diện website cùng với những dữ liệu động chèn trên đó, hoạt động này được Angular thực hiện theo cơ chế react - tức là liên tục cập nhật giao diện website dựa vào những thay đổi mới nhất (có thể là thay đổi từ phía người dùng hay server). Nhiệm vụ thứ hai của Angular là xử lý dữ liệu người dùng nhập vào các form như kiểm tra tính hợp lệ của dữ liệu đầu vào và kết nối với phần backend để gửi dữ liệu đó về phía server. Angular đem lại cho người dùng trải nghiệm sử dụng website như sử dụng một ứng dụng di động vì nó không cần reload lại trang web khi di chuyển giữa các site trên trang web đó, các thao tác được đáp ứng nhanh và đó là trải nghiệm mà người dùng có khi sử dụng ứng dụng di động.

MongoDB là một cơ sở dữ liệu có dạng NoSQL. Đặc điểm của nó là không có ràng buộc quan hệ, phân tán, mã nguồn mở, có khả năng lưu trữ, xử lý một lượng rất nhỏ cho tới lượng dữ liệu cực lớn. Trong mô hình MEAN stack, MongoDB đóng vai trò là cơ sở dữ liệu cho website và có khả năng kết nối dễ dàng với Node và Express.

NodeJS là một môi trường cho phép chạy JavaScript ở phía server mà không cần trình duyệt. Nó có nhiệm vụ tiếp nhận những Request và gửi trả những Response nếu có, cụ thể trong mô hình MEAN stack thì những Request này sẽ đến từ Angular ở phía client hay browser của người dùng. NodeJS cũng có nhiệm vụ xử lý những tính toán logic ở phía server, xử lý vấn đề bảo mật như xác thực, phân quyền… hay khả năng thao tác với các tập tin và giao tiếp với cơ sở dữ liệu MongoDB.

ExpressJS là một framework của NodeJS, nó cũng cấp thêm những chức năng giúp cho quá trình làm việc bằng NodeJS được dễ dàng hơn, cụ thể có thể là giúp dễ dàng hơn khi xử lý những Request tới những địa chỉ khác nhau từ phía người dùng. Chung quy lại, ExpressJS giúp cho công việc xử lý ở phía backend trở nên dễ dàng và nhanh chóng hơn.

* + 1. ***Cách thức MEAN stack hoạt động***



**Hình 1. Sơ đồ hoạt động Mean Stack**

Tại mỗi client, Angular thực hiện xử lý và tạo ra giao diện người dùng, Angular cũng thực thi single-page-application trên mỗi client.

Tại server, NodeJS và ExpressJS có nhiệm vụ xử lý logic, nghiệp vụ và xác thực. MongoDB ở server có nhiệm vụ tổ chức và lưu trữ dữ liệu.

Client và server sẽ giao tiếp bằng cách gửi các Request và response, đây là các Request và Response gửi ngầm bằng công nghệ AJAX giúp người dùng không cần reload lại trang web. Trong các Request và Response này, dữ liệu được tổ chức thành dạng JSON - là một dạng dữ liệu được tổ chức ngắn gọn và có dễ đọc hiểu.

* + 1. ***Lợi thế của MEAN stack***

MEAN stack sử dụng một ngôn ngữ chung là JavaScript cho cả client-side và server-side, giúp cho quá trình phát triển trang web trở nên dễ tiếp cận.

NodeJS có khả năng chạy trên cả Linux, Windows và OS X nên nó không phụ thuộc vào hệ điều hành. Cơ chế non-blocking I/O giúp server có thể đáp ứng hàng ngàn connection đồng thời.

MEAN stack sử dụng JSON là kiểu dữ liệu trao đổi giữa tất cả các thành phần nên không cần thêm thư viện để chuyển đổi dữ liệu trong quá trình tương tác giữa phía client và phía server.

MongoDB có khả năng quản lý một lượng dữ liệu lớn, có thể thêm thuộc tính mới vào các bảng mà không cần cập nhật lại toàn bộ bảng.

Các công nghệ sử dụng trong MEAN stack đều là mã nguồn mở và miễn phí. Cộng đồng sử dụng rộng lớn thuận tiện cho việc học hỏi kinh nghiệm hay giải đáp thắc mắc.

## **CHƯƠNG 2: NODEJS**

### **Tổng quan**

NodeJS là một framework môi trường hỗ trợ dịch và thực thi chương trình được viết bằng JavaScript mà không cần dùng tới trình duyệt.

NodeJS ra đời nhờ những lập trình viên đầu tiên của JavaScript mở rộng nó từ một ngôn ngữ chỉ có thể thực thi trên trình duyệt thành ngôn ngữ có thể thực thi độc lập trên máy tính.

NodeJS cũng giống với trình duyệt hiện nay, được chạy dựa trên V8 JavaScript engine - công cụ giúp trình duyệt dịch và thực thi JavaScript. Ngoài ra, NodeJS cung cấp một số thư viện không có sẵn trong V8 engine

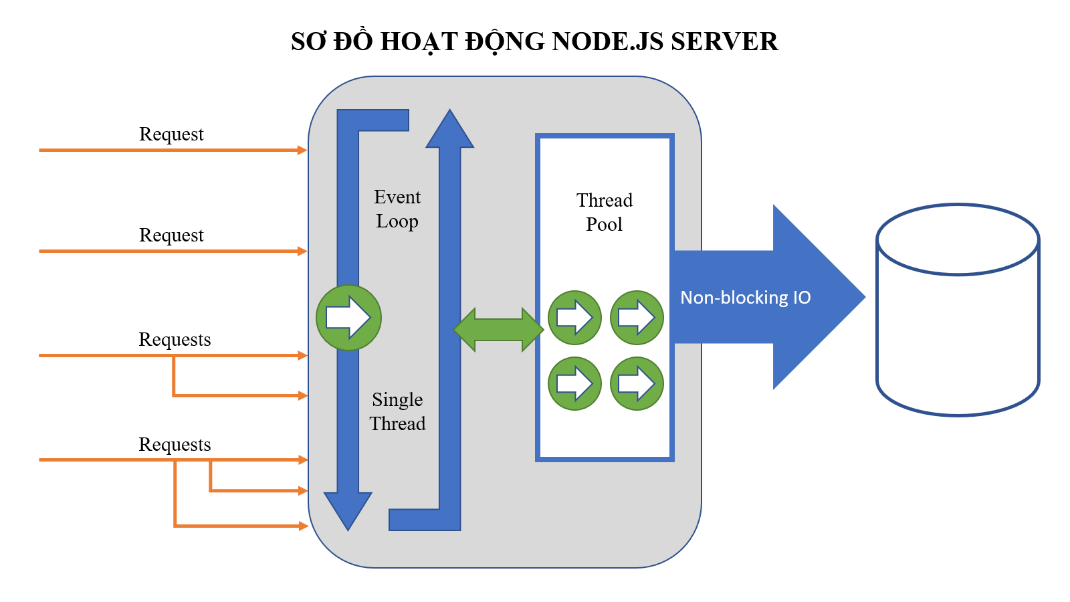
* Một số khái niệm tiêu biểu trong NodeJS
* **Event**: mỗi một thao tác, yêu cầu (request) đến server đều được NodeJS xem là một sự kiện (event)
* **Event Queue**: hàng đợi sự kiện là nơi tất cả các sự kiện (event) đã đi vào sẽ lần lượt được xử lý theo cơ chế FIFO (First In First Out), tức là các sự kiện đến server trước sẽ được NodeJS xử lý trước, các sự kiện đến server sau sẽ được xử lý sau.
* **Callbacks**: là các thao tác, các xử lý, các hàm sẽ được thực hiện sau khi đã thực thi hoàn thành một hàm định nghĩa callback. Việc sử dụng callback giúp cho lập trình viên quản lý các vấn đề về.
* **Message**: nội dung thông báo tương ứng với từng callback.
* **Blocking I/O**: là cơ chế được ứng dụng trong phần lớn các ngôn ngữ server-side, cơ chế này quản lý các request đến server và lần lượt cho thực hiện yêu cầu của request tiếp theo khi request đang xử lý đã hoàn thành. Từ đây dễ thấy được rằng nếu các yêu cầu xử lý của request đến sớm tiêu tốn nhiều thời gian thì các request đến sau bắt buộc phải đợi toàn bộ thời gian xử lý này. Với NodeJS việc quản lý các request yêu cầu được thay thế bằng cơ chế xử lý **Non-Blocking I/O** để giải quyết vấn đề thời gian chờ nêu trên.
* **Thread Pool**: là một chương trình viết bằng ngôn ngữ C++ được tích hợp bên trong NodeJS, chương trình này cho phép xử lý đa luồng. Đây chính là môi trường xử lý chính cho tất cả tác vụ được server NodeJS xử lý và các tác vụ này cũng được xử lý trên các luồng khác nhau.
* **Event Loop**: là một vòng lặp vô tận, nó chuyển các yêu cầu đã nhận sang Thread Pool, đồng thời mỗi một yêu cầu cũng được tự động đăng ký một Callback, khi yêu cầu được xử lý xong hàm Callback tương ứng sẽ được thực thi.

### **Nguyên tắc hoạt động**

Nguyên tắc hoạt động quan trọng nhất tạo nên khả năng xử lý với tốc độ cao của NodeJS đến từ việc xử lý **Non-Blocking**. Quá trình xử lý này cho phép NodeJS nhận và xử lý đồng thời nhiều tác vụ thông qua duy nhất một luồng xử lý (Single thread) thay vì tạo nhiều luồng xử lý tương ứng với các request ở các ngôn ngữ Server truyền thống đã có. Các tác vụ được nhận vào luồng xử lý nếu không được quy định tuần tự thực hiện (phải hoàn thành công việc A trước rồi sau đó bắt đầu thực hiện công việc công việc B) sẽ được cho phép truy cập tài nguyên song song lẫn nhau mà không bị khóa lại (Blocking) đợi tác vụ trước đó. Từ đó mà những tác vụ diễn ra không liên quan đến nhau có thể xử lý song song độc lập với nhau, rút ngắn lượng lớn thời gian xử lý cho server NodeJS

Kết hợp với xử lý quá trình xử lý Non-Blocking là **Callbacks** và **Event Loop** giúp cho NodeJS quản lý tất cả các tác vụ được được thực thi. Cụ thể khi một tác vụ được xử lý Non-Blocking hoàn thành sẽ gửi về 1 Callback với ý nghĩa thông báo với hệ thống rằng tác vụ đã thực hiện xong hoàn toàn. Trước đó khi các tác vụ này được định nghĩa thì callback của từng tác vụ cũng đã được đưa vào Event Table để chờ các sự kiện (Event) cho phép thực thi. Khi Event mong chờ của tác vụ được diễn ra cũng là lúc các message của tác vụ đang chờ trong Event Table được di chuyển sang Event Queue chờ thực hiện theo nguyên tắc “đến trước thực hiện trước” (First In First Out - FIFO). Song song với quá trình này là quá trình Event Loop thực hiện một vòng lặp vô tận kiểm tra có message nào đang đợi trong Event Queue hay không và Thread Pool có đang thực hiện tác vụ nào hay không, nếu đồng thời có message đang chờ và Thread Pool có thể tiếp nhận tác vụ Event Loop sẽ đưa phần tử đầu tiên của Queue vào Thread Pool xử lý và gửi về callback khi tất cả các message quy định callback tương ứng đã hoàn thành. Nếu không còn message nào trong Event Queue vòng lặp Event Loop vẫn luôn luôn được chạy để kiểm tra và xử lý ngay khi có tác vụ chờ.

Tổng thể với sự kết hợp giữa các đặc trưng tiêu biểu: Non-Blocking I/O, Callbacks, Event Loop một server NodeJS xử lý bất đồng bộ (asynchronous) các đồng bộ các tác vụ được yêu cầu và trả về kết quả ngay khi các tác vụ được hoàn thành thông qua callback. Còn quá trình xử lý và phản hồi callback này được xử lý với sự điều phối, quản lý của vòng lặp vô tận xử lý sự kiện Event Loop lấy từng message của quy định callback thực hiện với tài nguyên hệ thống và phản hồi callback khi tất cả message quy định callback tương ứng đã hoàn thành.



**Hình 2. Sơ đồ hoạt động NodeJs**

### **Ưu điểm**

NodeJS cung cấp khả năng mở rộng cao. NodeJS được nhận xét là dễ mở rộng chương trình cả về chiều rộng lẫn chiều sâu.

Vì JavaScript là một trong những ngôn ngữ lập trình phổ biến nhất. Những lập trình viên đã biết JavaScript có thể chuyển sang NodeJS mà không gặp nhiều khó khăn.

Sử dụng NodeJS, lập trình viên sẽ chỉ cần sử dụng một ngôn ngữ lập trình duy nhất là JavaScript cho cả phần front-end và back-end.

Nhờ có tốc độ thông dịch và thực thi nhanh, NodeJS có thêm lợi thế để được chọn làm ngôn ngữ phát triển server-side hiệu quả.

Cộng đồng sử dụng NodeJS lớn mạnh cũng giúp các lập trình viên mới tiếp xúc với NodeJS dễ hòa nhập, học hỏi và tháo gỡ khó khăn.

Có được sự hỗ trợ từ các công cụ thông dụng tích hợp sẵn trong NodeJS. Lập trình viên có thể dễ dàng kiểm tra mã nguồn đã chạy đúng hay chưa với Jasmin trong việc tạo các unit test. Tương tự khi cài đặt những thư viện mới phục vụ cho dự án, Node package manager (npm) sẽ là công cụ hỗ trợ đắc lực.

### **Khuyết điểm**

Ứng dụng tiêu tốn tài nguyên do được viết trên nền tảng C++ và Javascript nên quá trình khởi chạy NodeJS phải thông qua một quá trình biên dịch. Việc ngôn ngữ lập trình Javascript thân thiện với người dùng cũng làm cho hệ thống phải tiêu tốn nhiều hơn các tài nguyên để biên dịch về ngôn ngữ máy.

## **CHƯƠNG 3: EXPRESSJS**

### **Tổng quan**

Express là một framework nhỏ nhưng linh động được xây dựng trên nền tảng NodeJS, framework này cung cấp các chức năng mạnh mẽ để phát triển các ứng dụng web và mobile.

Express được sinh ra để làm đơn giản hóa quá trình cấu hình định tuyến, giao tiếp giữa client với server cho lập trình viên. Hiện tại framework này đã có thêm nhiều gói trung gian (middleware packages) hỗ trợ đơn giản hóa công việc của lập trình viên như: body-parser, cors, timeout, ...

Express đang là một trong những framework phổ biến nhất trong lập trình các ứng dụng server-side và được áp dụng trong nhiều framework sử dụng Javascript khác điển hình có MEAN Stack, MERN Stack, ...

### **Công dụng**

Cấu hình/lập trình các xử lý cho các HTTP request ở tất cả các URL khác nhau.

Kết hợp với các công cụ kết xuất giao diện (“View” rendering) để phản hồi dữ liệu thông qua việc chèn dữ liệu vào các mẫu (templates)

Cấu hình chung toàn bộ ứng dụng website như một cổng (port) kết nối và quy định các thông tin phản hồi (response) tương ứng.

Thêm các thông tin bổ sung vào request, response đang xử lý thông qua các trung gian (middleware) bất cứ khi nào mà request vẫn còn nằm trong giai đoạn xử lý.

## **CHƯƠNG 4: ANGULAR**

### **Tổng quan về Angular**

Angular là một Javascript framework xây dựng ứng dụng web mã nguồn mở (open source) hoạt động ở phía người dùng (client-side) sử dụng HTML, CSS và ngôn ngữ lập trình Typescript.

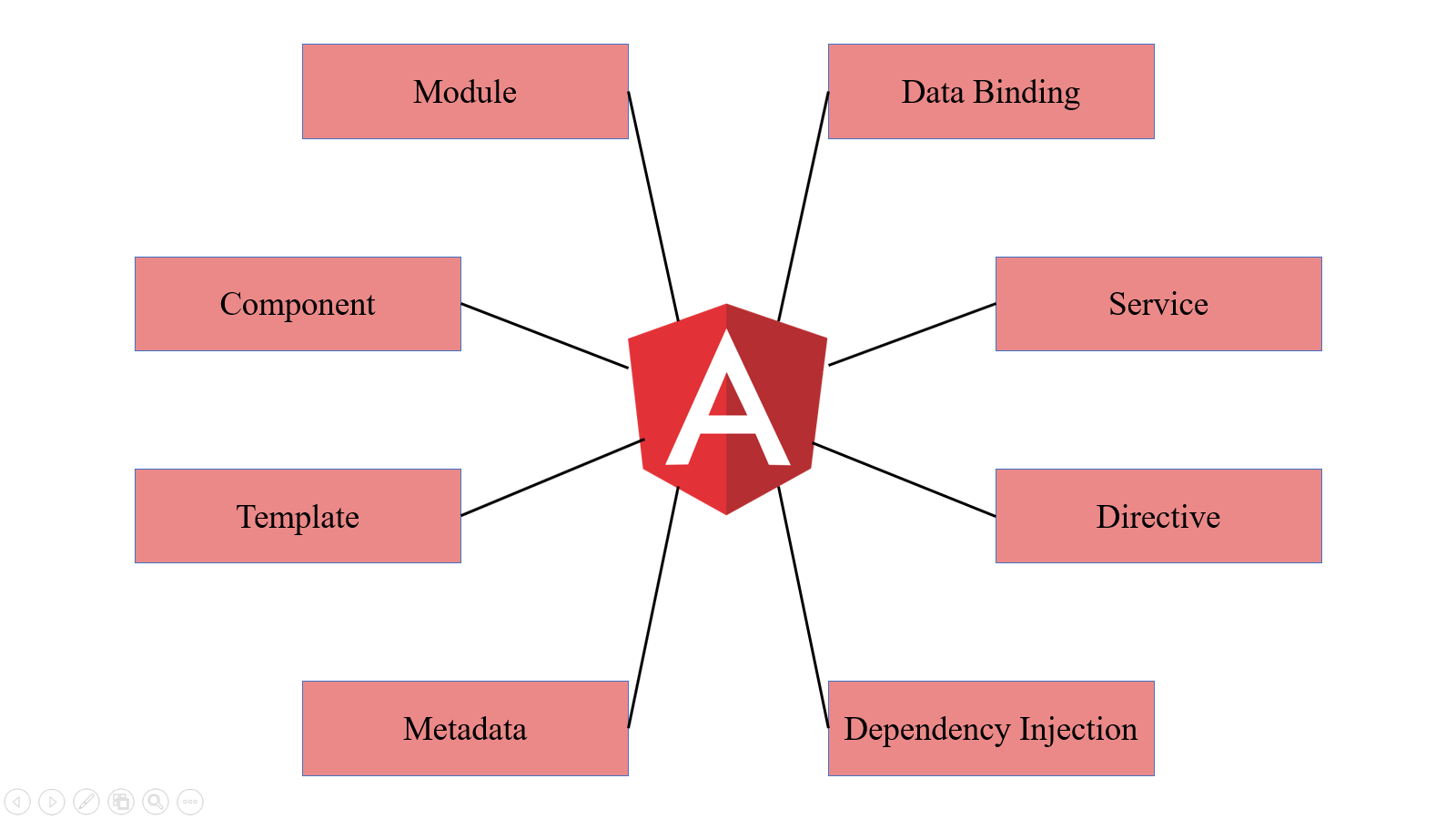
Được phát triển từ đầu năm 2009 và lần đầu ra mắt chính thức vào năm 2012 với tên gọi AngularJS hay Angular 1, framework này nhanh chóng thu hút được sự chú ý từ nhiều lập trình viên với cơ chế phân chia module, binding data và hoạt động trên một trang giao diện (single page application). Tại thời điểm này AngularJS bao gồm HTML, CSS và ngôn ngữ lập trình Javascript

Đồng thời với sự phát triển và ưu thế được lựa chọn bởi nhiều lập trình viên, Angular cũng đã được các nhà phát triển bên thứ ba nổi tiếng trên thế giới, cũng như các lập trình viên xây dựng thư viện hỗ trợ riêng cho framework này đặc biệt là các thư viện nổi tiếng trong xây dựng giao diện ứng dụng website như Bootstrap, Font Awesome, AmChart, …

Đặc trưng của Angular tiêu biểu so với các framework đã có tại thời điểm ra mắt là Angular cung cấp nền tảng để lập trình viên tạo ra ứng dụng web trên một trang (single page application). Điều này đồng nghĩa với việc giảm thiểu tối đa số lần tải lại trang và rút ngắn thời gian đợi của người dùng trong quá trình sử dụng

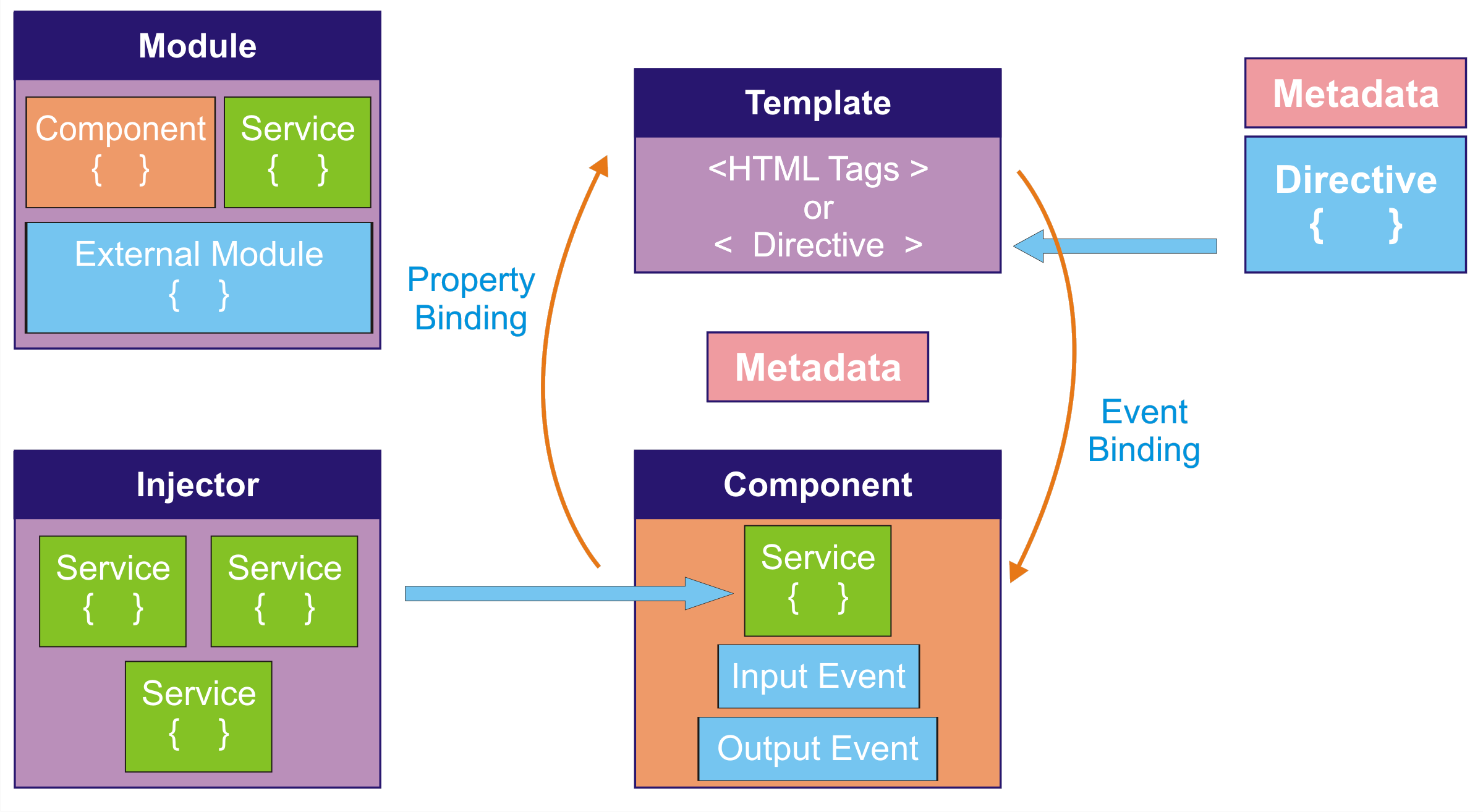
### **Mô hình**

Angular hoạt động dựa trên 8 thành phần cơ bản là Module, Component, Template, Metadata, Data binding, Service, Directive, Dependency injection



**Hình 3. Các thành phần cơ bản trong angular**

Hoạt động giữa các thành phần trong angular sẽ được mô tả như dưới đây



**Hình 4. Cách hoạt động giữa các thành phần trong Angular**

Các thành phần như component, service, directive, template,... sẽ được khai báo và đóng gói trong các Module. Trong các module, các component, service, directive bản chất đều là các class bình thường trong TypeScript, để giúp angular hiểu và nhận biết đâu và component, đâu là directive, angular cung cấp sẵn các decorator (một mẫu design pattern) để gắn vào các class như @Component để nhận biết component, @Directive để nhận biết directive… Trong các decorator này, người dùng có thể cung cấp các metadata để khai báo tên (selector) thành phần đó, khai báo component sẽ gắn với giao diện template html và css là gì. Với service thì không có decorator nhận diện và nó giống như một class TypeScript thông thường. Như vậy, angular đã nhận diện được các thành phần trong nó.

Trong quá trình hoạt động, template sẽ được dùng để khởi tạo giao diện. Nhờ các directive và data binding, template sẽ trải qua giai đoạn xử lý, sửa đổi để tạo ra giao diện hiển thị thật sự trên màn hình, các sự kiện từ DOM (Document Object Model đại diện cho trang giao diện) sẽ liên tục được theo dõi và qua đó, component và directive sẽ có các phản hồi tương ứng. Từ các phản hồi của directive và component, đối tượng đại diện cho giao diện là DOM có thể nhận được những thay đổi và những thay đổi đó sẽ tự động được trình duyệt cập nhật và hiển thị trên màn hình. Các service sẽ đóng vai trò là thành phần hỗ trợ xử lý, có thể sử dụng lại trong nhiều component nhờ dependency injection.

Angular được biết đến là có nhiệm vụ thực thi Single Page Application và giao diện hiển thị trên màn hình của người dùng cũng chỉ có một trang html duy nhất. Nhưng mỗi component lại có phần template đính kèm và sẽ nằm tách rời ở nhiều file, angular sẽ cần gom tất cả phần template đang sử dụng, hợp lại thành một template thống nhất để hiển thị trên màn hình.

Trong một ứng dụng angular có thể có rất nhiều module, mỗi module lại có nhiều component, service…. Trong tất cả sẽ có một module là root module. Module cũng là một class TypeScript và được gắn decorator @NgModule, bên trong nó là các metadata. Với root module thì sẽ có một metadata đặc biệt là “bootstrap”, nó chứa mảng các component sẽ được chọn để khởi động ban đầu. Thông thường các ứng dụng angular sẽ có sẵn một component được chọn để khởi động là AppComponent ngay từ khi tạo project. Khi ứng dụng angular mới khởi động, trang giao diện sẽ là nội dung từ trang index.html với styles.css.

## **CHƯƠNG 5: MONGODB**

### **Tổng quan**

MongoDB hệ quản trị cơ sở dữ liệu mã nguồn mở (open-source) thuộc họ cơ sở dữ liệu phi quan hệ (NoSQL).

MongoDB được phát triển từ năm 2007 bởi một tổ chức tên là MongoDB Inc có trụ sở tại New York. Tại thời điểm đó MongoDB được phát triển như là một nền tảng dịch vụ (Platform as a services - PAAS). Đến năm 2009 MongoDB đã được chính thức giới thiệu trên thị trường như một máy chủ cơ sở dữ liệu mã nguồn mở (open-source database server) được duy trì và phát triển bởi tổ chức MongoDB Inc. Tại thời điểm đó đã có nhiều tổ chức lớn và vừa như SourceForge, Foursquare, eBay sử dụng MongoDB để phát triển cơ sở dữ liệu của họ.

MongoDB được xây dựng theo hướng là một cơ sở dữ liệu hướng đối tượng vì vậy nó không có các ràng buộc toàn vẹn dữ liệu như ở các Cơ sở dữ liệu quan hệ mà cho phép lưu trữ dữ liệu linh hoạt (stores data in flexible) có dữ liệu khác nhau trong cùng một Collection (tương đương với Table ở SQL). Do chính đặc thù này mà MongoDB Có thể được sử dụng để lưu các dữ liệu đa dạng và phức tạp mà không bị hạn chế khi có thay đổi về cấu trúc lưu dữ liệu.

MongoDB là một cơ sở dữ liệu phân tán vì vậy nó sẽ có những đặc trưng của cơ sở dữ liệu phân tán như: tính sẵn sàng cao, khả năng mở rộng theo chiều ngang,....

### **Đặc trưng**

MongoDB là cơ sở dữ liệu No-SQL loại Document Database với nhiều yếu tố được bổ sung để nâng cao hiệu suất sử dụng như Index, Schema,....

MongoDB được lập trình bằng ngôn ngữ C++ và hỗ trợ tốt với các Javascript Framework.

Các khái niệm tiêu biểu khi sử dụng MongoDB

* Document - tài liệu: là đơn vị cơ bản của dữ liệu trong MongoDB, một document tương đương với một dòng trong bảng của SQL. Mỗi một document đều có cấu trúc là một đoạn chuỗi json độc lập.
* Collection: là tập hợp các document có các thuộc tính tương tự nhau. Collection tương đương với một bảng (Table) trong SQL. Điểm khác nhau rõ rệt nhất ở đây chính là việc bảng trong SQL có cấu trúc được quy định sẵn và dữ liệu vào luôn phải tuân theo cấu trúc đó, còn Collection có thể tiếp nhận tự do mọi document có các thuộc tính khác nhau vào mà không hề phát sinh lỗi.
* Database: là các cơ sở dữ liệu độc lập lẫn nhau của MongoDB, một Database của MongoDB bao gồm một hay nhiều Collection và các cài đặt, phân quyền kèm theo Database đó.
* \_id: là một khóa định danh đặc biệt được quy định tự động bởi MongoDB khi thêm một Document vào Collection. \_id của bất cứ Document nào đều luôn là duy nhất trong Collection chứa document đó.
* Index - chỉ mục: là một cấu trúc dữ liệu thu thập thông tin về các trường (field) trong Document của Collection từ đó mà nó giúp cho các câu truy vấn (query) đến các trường này tăng đáng kể tốc độ truy vấn. Tuy nhiên việc tạo thêm các Index cũng làm tăng đáng kể dung lượng bộ nhớ tiêu hao cho Database MongoDB. Trường \_id là trường chỉ mục luôn luôn được khởi tạo mặc định bởi MongoDB khi thêm mới một Document.
* Query - truy vấn: cũng giống như tất cả cơ sở dữ liệu khác, MongoDB hỗ trợ truy vấn dữ liệu và nhà phát triển của cơ sở dữ liệu này cũng đã cung cấp số lượng lớn các câu lệnh bổ trợ giúp đơn giản hóa việc truy vấn dữ liệu hơn rất nhiều lần so với các ngôn ngữ truy vấn dữ liệu truyền thống mà vẫn đảm bảo được tốc độ truy vấn nhanh.

### **Ưu điểm**

* Lược đồ dữ liệu linh hoạt, do là cơ sở dữ liệu No-SQL tổ chức dữ liệu theo kiểu Document Database nên có thể tùy ý lưu các Document với các thuộc tính và kích thước Document khác nhau và cùng một Collection của MongoDB
* Tốc độ truy vấn dữ liệu nhanh, có thể tăng tốc độ truy vấn dữ liệu ở các trường (field) thông thường bằng cách khởi tạo Index cho trường tương ứng đó.
* Khả năng mở rộng cơ sở dữ liệu lớn.
* Phù hợp cho xu hướng Big Data trên toàn thế giới cũng như dễ dàng được lựa chọn với các ứng dụng thời gian thực (real-time)
* Được hỗ trợ bởi nhiều thư viện Javascript và có thể ứng dụng cho nhiều nền tảng, ngôn ngữ khác nhau.

### **Nhược điểm**

Là cơ sở dữ liệu không có cấu trúc nên không còn các ràng buộc khóa ngoại, khóa chính như cơ sở dữ liệu quan hệ. Từ đó các vấn đề về toàn vẹn dữ liệu phải được chuyển đến cho Server ứng dụng xử lý thay. Các vấn đề về sai dữ liệu đầu vào thường không được MongoDB cảnh báo do không có ràng buộc nào quy định.

Tiêu tốn nhiều tài nguyên trong quá trình xử lý cũng như tiêu tốn bộ nhớ cho việc khởi tạo thêm các Index tăng tốc độ truy vấn.

1.1.2. Yêu cầu chức năng

1.1.2.1. Đôi với khách hàng

* View ,Search , Filter sách
* Chức năng giỏ hàng
* Lưu lại lịch sử mua hàng của khách
* Giới thiệu những sách cùng thể loại/tác giả với sách khách đã mua (Recomment System)
* Khi có sách mới về hoặc có Chương trình khuyến mãi, gửi email cho khách *(Nếu khách đồng ý)*
* Comment, review, đánh giá sách
* Sau khi khách xác nhận mua hàng, gửi 1 mail xác nhận đơn hàng đến khách
* Có nút Favorite - Khách hàng muốn mua nhưng giá cao quá *( Khách hàng sẽ nhận được thông báo khi những quyển sách này giảm giá )*
* *Nhận được mã giảm giá sau mỗi lần mua hang với mệnh giá xxx.xxxđ*
* *Sử dụng mã giảm giá mỗi lần thanh toán*
* *Đổi mật khẩu*

1.1.2.2. Đối với chủ cửa hàng

* Đổi mật khẩu
* Xem thông tin khách hàng , danh mục sản phẩm , lịch sử mua hàng
* Tạo sự kiện , event
* Doashboard quản lý hàng tháng
* Tracking theo dõi hoạt động khách hàng
* Nhận được thông báo khi khách đăng thắc mắc, phàn nàn và phải trả lời
* Nhận được thông báo khi khách rate sách thấp (1, 2 sao), xác định được khách nào và liên hệ để hỏi lý do

## **1.2. Mô tả quá trình làm**

### 1.2.1. Kế hoạch thực hiện

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Bảng phân công công việc | | | | | |
| STT | Mô tả công việc | Ngày bắt đầu | Ngày kết thúc | Ngày bắt đầu | Ngày kết thúc |
| (Dự kiến) | (Dự Kiến) | (Thực Tế) | (Thực Tế) |
| 1 | Lên ý tưởng, viết mô tả đồ án; tìm hiểu, lựa chọn và chuẩn bị môi trường làm việc | 24/02/2020 | 24/02/2020 |  |  |
| 2 | Thiết kế giao diện BookStore | 24/02/2020 | 27/02/2020 |  |  |
| 3 | Thiết kế Database | 28/02/2020 | 02/03/2020 |  |  |
|  |  |  |  |  |  |
| 4 | Tìm hiểu về Facebook/Google Api, payment gate | 03/03/2020 | 03/03/2020 |  |  |
| 5 | Code chức năng trang đăng nhập, đăng ký | 03/03/2020 | 04/03/2020 |  |  |
| 6 | Code chức năng danh mục sản phẩm, thông tin sản phẩm ,seri | 05/03/2020 | 07/03/2020 |  |  |
| 7 | Code chức năng giỏ hàng , thanh toán , sendmail thanh toán | 08/03/2020 | 10/03/2020 |  |  |
| 8 | Code chức năng trang lịch sử mua hàng | 11/03/2020 | 11/03/2020 |  |  |
| 9 | Code chức năng Search,Filter | 12/03/2020 | 12/03/2020 |  |  |
| 10 | Code chức năng đánh giá và rating sản phẩm | 13/03/2020 | 14/03/2020 |  |  |
| 11 | Code chức năng Admin ,  Code chức năng thêm danh mục sản phẩm, tác giả, NXB,…. | 15/03/2020 | 17/03/2020 |  |  |
| 12 | Code chức năng yêu thích sản phẩm | 18/03/2020 | 19/03/2020 |  |  |
| 13 | Code chức năng tạo event ưu đãi sản phẩm khách hàng | 20/03/2020 | 24/03/2020 |  |  |
| 14 | Code chức năng Recommend system cho sản phẩm | 25/03/2020 | 28/03/2020 |  |  |
| 15 | Tạo DoashBoard quản lý hàng tháng | 29/03/2020 | 31/03/2020 |  |  |
| 16 | Code chức năng tracking theo dõi hoạt động của người dùng | 01/04/2020 | 05/04/2020 |  |  |
| 17 | Bổ sung , sửa đổi và fixbug | 06/04/2020 | 09/04/2020 |  |  |
| 18 | Viết báo cáo | 10/04/2020 | 15/04/2020 |  |  |

**Bảng 1**: Bảng phân công công việc