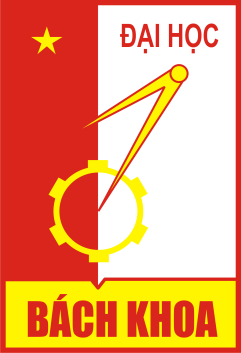
**TRƯỜNG ĐẠI HỌC BÁCH KHOA HÀ NỘI**

**VIỆN CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG**



**BÁO CÁO BÀI TIỂU LUẬN**

**Công Nghệ Web**

**Đề tài: Tìm Hiểu Về NodeJS Và MongoDB**

|  |  |
| --- | --- |
| **Giảng viên phụ trách:** | TS: Đỗ Bá Lâm |
| **Sinh viên thực hiện:** | Trần Trọng Bình - 20155142 |
|  |

***MỤC LỤC***

1. **TỔNG QUAN ĐỀ TÀI**
2. **TỔNG QUAN VỀ NODEJS VÀ MONGODB**
3. **TRIỂN KHAI DỰ ÁN THỰC TẾ**
4. **KÊT LUẬN**

LỜI NÓI ĐẦU

Lướt Facebook xuất hiện khá nhiều chủ đề về nên học gì vào năm 2018 và các câu trả lời chủ yếu là NodeJS, AngularJS, Phalcon FW, … Trong số đó em khá ấn tượng với NodeJS – một nền tảng rất mới mẻ. Nó là dự án được xem nhiều thứ 2 trên GitHub.org, có khá nhiều theo dõi trên một nhóm của nó trên Google.com và có trên 15.000 modules được công bố trên NMP. Tất cả những điều đó cho thấy rằng nó hứa hẹn là một nền tảng hấp dẫn có thể thay thế được các nền tảng truyền thống như Apache, PHP, Python …

Bên cạnh quá trình tìm hiểu về nodejs thì em bắt gặp được một hệ quản trị có dạng NoSQL, đó là MongoDB – một mã nguồn mở, là một tập tài liệu dùng cơ chế NoSQL để truy vấn, được viết bởi C++ nên có khả năng tính toán với tốc độ cao chứ không giống như các hệ quản trị CSDL hiện nay.

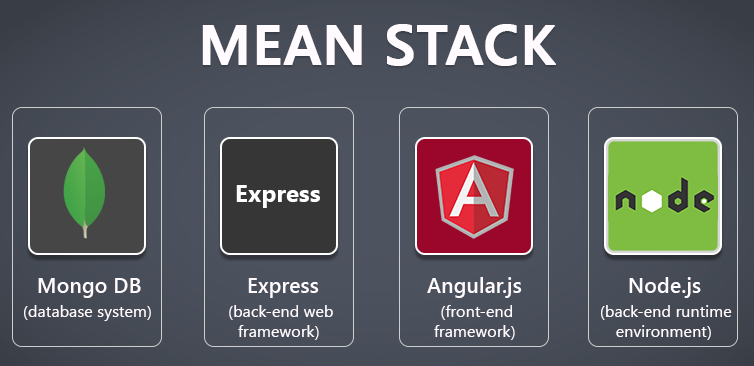
1. **TỔNG QUAN ĐỀ TÀI**
2. **Tên đề tài**

Tìm hiểu về NodeJS và MongoDB

1. **Ý tưởng đề tài**

Như ở phần mở đầu đã trình bày thì NodeJs đang là một nền tảng mới và thú vị cho việc phát triển các ứng dụng web, ứng dụng Server với Javascript – nền tảng nhập xuất bất đồng bộ, mạnh mẽ cho việc xây dựng một ứng dụng thời gian thực. Do vậy em đã đi tìm hiểu về nó và bắt tay vào làm một dự án thực tế đó là website bán quần áo đề kết hợp giữa NodeJS và MongoDB đề nắm được những thứ cơ bản về cặp đôi này.

1. **TỔNG QUAN VỀ NODEJS VÀ MONGODB**
2. **Khái niệm MEAN Stack**



Để tạo một ứng dụng hoàn chỉnh như một trang web động thật sự thì cần phải có sự kết hợp với các ngôn ngữ phía server và ngôn ngữ thao tác cơ sở dữ liệu. Và MEAN Stack là một sự kết hợp như thế.

MEAN Stack bao gồm: M (MongoDB), E (ExpressJS), A (AngularJS), N (NodeJS).

MEAN Stack là giải pháp hoàn hảo để xây dựng một mô hình SPA hoàn chỉnh nhất.

1. **NodeJS**

NodeJS là một nền tảng được xây dựng trên “V8 Javascript engine” được viết bằng c++ và Javasccript. Nền tảng này được phát triển bởi Ryan Lienhart Dahl vào năm 2009.

Tại thời điểm này, nó được rất nhiều nhà phát triển ứng dụng ưa chuộng và còn là một nền tảng rất mới mẻ. Nó là dự án được xem nhiều thứ 2 trên GitHub.org, có khá nhiều theo dõi trên một nhóm của nó trên Google.com và có trên 15.000 modules được công bố trên NMP(khái niệm này sẽ được giới thiệu ở phần sau). Tất cả những điều trên cho thấy rằng, NodeJS đang là một nền tảng mới và thú vị cho việc phát triển các ứng dụng web, ứng dụng Server. NodeJS cũng cho thấy rằng nó hứa hẹn là một nền tảng hấp dẫn có thể thay thế được các nền tảng truyền thống như Apache, PHP, Python . . .

NodeJS là một nền tảng cho việc viết ứng dụng Javascript phía server, không giống như Javascript chúng ta thường viết trên trình duyệt. Với ngôn ngữ Javascript và nền tảng nhập xuất bất đồng bộ, nó là một nền tảng mạnh mẽ để phát triển các ứng dụng thời gian thực.

NodeJS có thể chạy trên nhiều nền tảng hệ điều hành khác nhau từ WIndow cho tới Linux, OS X nên đó cũng là một lợi thế. NodeJS cung cấp các thư viện phong phú ở dạng Javascript Module khác nhau giúp đơn giản hóa việc lập trình và giảm thời gian ở mức thấp nhất.

Khi nói đến NodeJS thì phải nghĩ tới vấn đề Realtime.

Realtime ở đây chính là xử lý giao tiếp từ client tới máy chủ theo thời gian thực. Giống như khi bạn lướt Facebook thì mỗi khi comment hay like một topic nào đó thì ngay lập tức chủ topic và những người đã comment trên đó sẽ nhận được thông báo là có người comment.

Nếu nghĩ Facebook đang sử dụng [Ajax](https://freetuts.net/hoc-php/hoc-ajax) thì đó là điều sai, bởi vì các request gửi lên với số lượng tính theo tấn nên nếu dùng Ajax thì Server chết ngay lập tức.

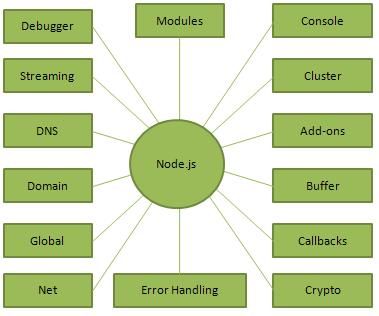
1. **Các đặc tính của NodeJS**



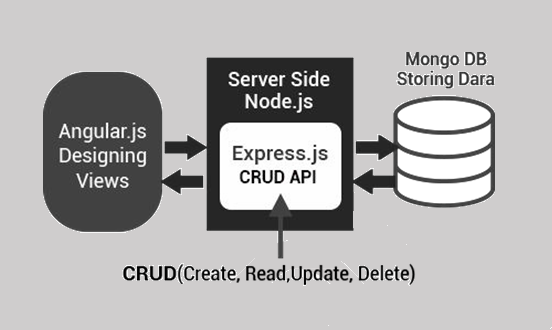
Đặc tính quan trọng nhất của NodeJS là Realtime, Bên cạnh đó còn khá nhiều đặc tính hay khác:

* **Không đồng bộ**: Tất cả các API của NodeJS đều không đồng bộ (none-blocking), nó chủ yếu dựa trên nền của NodeJS Server và chờ đợi Server trả dữ liệu về. Việc di chuyển máy chủ đến các API tiếp theo sau khi gọi và cơ chế thông báo các sự kiện của Node.js giúp máy chủ để có được một phản ứng từ các cuộc gọi API trước (Realtime).
* **Chạy rất nhanh**: NodeJ được xây dựng dựa vào nền tảng V8 Javascript Engine nên việc thực thi chương trình rất nhanh.
* **Đơn luồng nhưng khả năng mở rộng cao**: Node.js sử dụng một mô hình luồng duy nhất với sự kiện lặp. cơ chế tổ chức sự kiện giúp các máy chủ để đáp ứng một cách không ngăn chặn và làm cho máy chủ cao khả năng mở rộng như trái ngược với các máy chủ truyền thống mà tạo đề hạn chế để xử lý yêu cầu. Node.js sử dụng một chương trình đơn luồng và các chương trình tương tự có thể cung cấp dịch vụ cho một số lượng lớn hơn nhiều so với yêu cầu máy chủ truyền thống như Apache HTTP Server.
* **Không đệm**: NodeJS không đệm bất kì một dữ liệu nào và các ứng dụng này chủ yếu là đầu ra dữ liệu.
* **Có giấy phép**: NodeJS đã được cấp giấy phép bởi [MIT License](https://raw.githubusercontent.com/joyent/node/v0.12.0/LICENSE).

**Các thành phần quan trọng trong NodeJS:**



1. **ExpressJS**

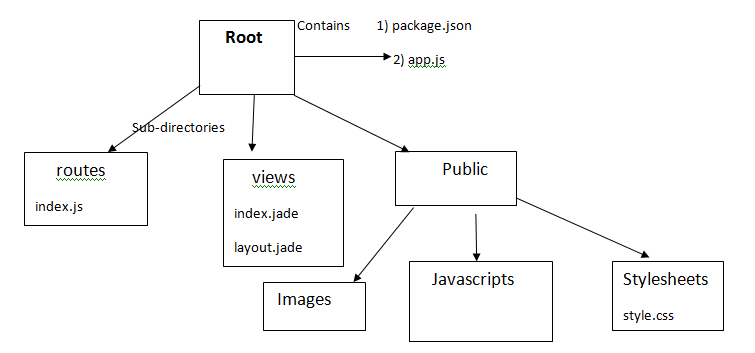


Express là một web application framework cho NodeJS, cung cấp các tính năng mạnh mẽ cho việc xây dựng một ứng dụng web đúng nghĩa hoặc lai.

ExpressJS là framework phổ biến và được sử dụng rộng rãi nhất của NodeJS, được xây dựng trên cấu trúc ngữ pháp của Sinatra. Ý tưởng đằng sau ExpressJS là đưa đến một framework nhẹ, dễ dàng tiếp cận để phát triển các ứng dụng web từ nhỏ đến lớn hay hybrid.

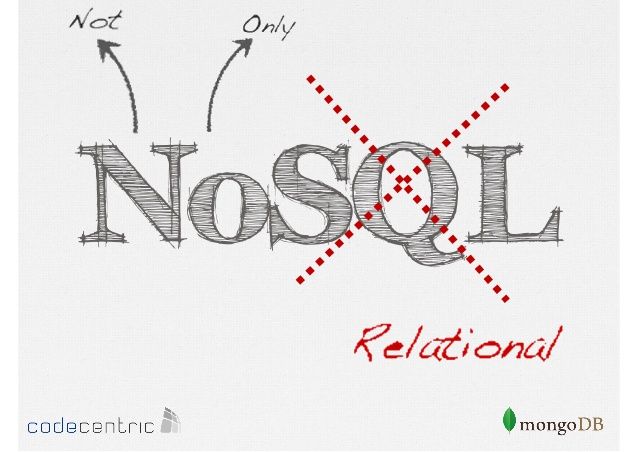
Express cũng có thể sử dụng để xây dựng một API mạnh mẽ và thân thiện với người dùng, vì nó cung cấp rất nhiều tiện ích HTTP và middleware cho việc kết nối.

**Cấu trúc của ExpressJS:**



* app.js chứa các thông tin về cấu hình, khai báo, các định nghĩa,... để ứng dụng chạy được.
* package.json chứa các package cho ứng dụng chạy.
* Folder routes: chứa các route có trong ứng dụng
* Folder view: chứa view/template cho ứng dụng
* Folder public chứa các file css, js, images,...cho ứng dụng

1. **NoSQL**



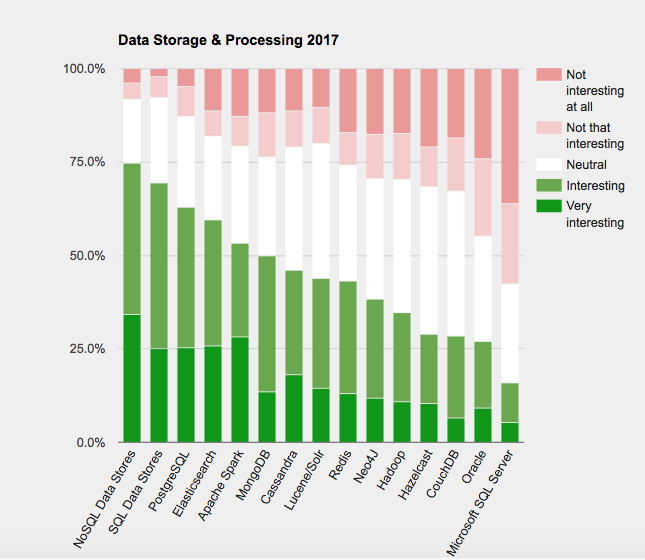
Khi làm việc với database, chúng ta đã quá quen với SQLServer, MySQL, PostgreSQL, Oracle ... Điểm chung của những database này là sử dụng ngôn ngữ SQL để truy vấn dữ liệu. Nhưng có 1 dạng database khác với những đặc tính khác biệt được gọi chung dưới cái tên là NoSQL.

NoSQL là 1 dạng CSDL mã nguồn mở không sử dụng T-SQL để truy vấn thông tin.

NoSQL được phát triển trên Javascript Framework với kiểu dữ liệu là JSON và dạng dữ liệu theo kiểu key và value (1 đặc trưng về dữ liệu trong JSON). NoSQL ra đời như là 1 mảnh vá cho những khuyết điểm và thiếu xót cũng như hạn chế của mô hình dữ liệu quan hệ RDBMS-hệ thống quản lý CSDL về tốc độ, tính năng, khả năng mở rộng, memory cache,...

Với NoSQL có thể mở rộng dữ liệu mà không lo tới những việc như tạo khóa ngoại, khóa chính, kiểm tra ràng buộc, .v.v...Vì NoSQL không hạn chế việc mở rộng dữ liệu nên tồn tại nhiều nhược điểm như: sự phục thuộc của từng bản ghi, tính nhất quán, toàn vẹn dữ liệu,....nhưng chúng ta có thể chấp nhận những nhược điểm đó để khiến ứng dụng cải thiện hiệu suất cao hơn khi giải quyết những bài toán lớn về hệ thống thông tin, phân tán hay lưu trữ dữ liệu.

**Thống kê tỉ lệ sử dụng của NoSQL:**



1. **MongoDB**



[MongoDB](http://www.mongodb.org/) là 1 hệ thống CSDL mã nguồn mở được phát triển và hỗ trợ bởi 10gen, là CSDL NoSQL hàng đầu được hàng triệu người sử dụng.

Thay vì lưu trữ dữ liệu dưới dạng bảng và các tuple như trong các CSDL quan hệ thì nó lưu trữ dữ liệu dưới dạng JSON (trong MongoDB được gọi là dạng BSON vì nó lưu trữ dưới dạng binary từ 1 JSON document)

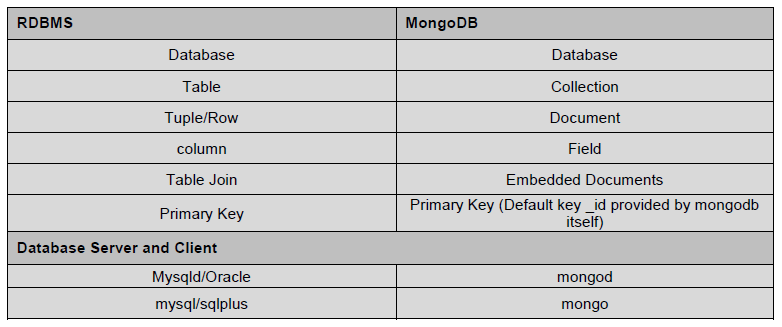
Mục tiêu chính của Mongo là giữ lại các thuộc tính thân thiện của SQL. Do đó các câu truy vấn khá giống với SQL nên MongoDB khá thích hợp cho các lập trình viên đã quen với ngôn ngữ truy vấn SQL. MongoDB có một khối lượng tính năng lớn và hiệu năng cao. Với các loại dữ liệu phong phú, nhiều truy vấn và việc giảm thời gian phát triển trong việc mô hình hóa các đối tượng.

MongoDB được sử dụng tốt nhất với nhu cầu cần truy vấn động, cần tốc độ nhanh cho một cơ sở dữ liệu lớn vì MongoDB ngoài tốc độ đọc nhanh ra thì tốc độ ghi của nó rất nhanh. MongoDB hỗ trợ việc tìm theo trường, khoảng kết quả tìm và tìm theo cú pháp. Các truy vấn có thể trả về các trường được qui định trong văn bản và cũng có thể bao gồm các hàm Javascript mà người dùng chưa định nghĩa. Cũng giống như các cơ sở dữ liệu quan hệ, bất cứ một trường nào trong MongoDB đều được đánh chỉ mục.

**Các thuật ngữ sử dụng trong MongoDB:**

* **\_id** – Là trường bắt buộc có trong mỗi document. Trường \_id đại diện cho một giá trị duy nhất trong document MongoDB. Trường \_id cũng có thể được hiểu là khóa chính trong document. Nếu bạn thêm mới một document thì MongoDB sẽ tự động sinh ra một \_id đại diện cho document đó và là duy nhất trong cơ sở dữ liệu MongoDB.
* **Collection** – Là nhóm của nhiều document trong MongoDB. Collection có thể được hiểu là một bảng tương ứng trong cơ sở dữ liệu RDBMS (Relational Database Management System). Collection nằm trong một cơ sở dữ liệu duy nhất. Các collection không phải định nghĩa các cột, các hàng hay kiểu dữ liệu trước.
* **Cursor** – Đây là một con trỏ đến tập kết quả của một truy vấn. Máy khách có thể lặp qua một con trỏ để lấy kết quả.
* **Database** – Nơi chứa các Collection, giống với cơ sở dữ liệu RDMS chúng chứa các bảng. Mỗi Database có một tập tin riêng lưu trữ trên bộ nhớ vật lý. Một mấy chủ MongoDB có thể chứa nhiều Database.
* **Document** – Một bản ghi thuộc một Collection thì được gọi là một Document. Các Document lần lượt bao gồm các trường tên và giá trị.
* **Field** – Là một cặp name – value trong một document. Một document có thể có không hoặc nhiều trường. Các trường giống các cột ở cơ sở dữ liệu quan hệ.
* **JSON** – Viết tắt của JavaScript Object Notation. Con người có thể đọc được ở định dạng văn bản đơn giản thể hiện cho các dữ liệu có cấu trúc. Hiện tại JSON đang hỗ trợ rất nhiều ngôn ngữ lập trình.
* **Index** – Là những cấu trúc dữ liệu đặc biệt, dùng để chứa một phần nhỏ của các tập dữ liệu một cách dễ dàng để quét. Chỉ số lưu trữ giá trị của một fields cụ thể hoặc thiết lập các fields, sắp xếp theo giá trị của các fields này. Index hỗ trợ độ phân tích một cách hiệu quả các truy vấn. Nếu không có chỉ mục, MongoDB sẽ phải quét tất cả các documents của collection để chọn ra những document phù hợp với câu truy vấn. Quá trình quét này là không hiệu quả và yêu cầu MongoDB để xử lý một khối lượng lớn dữ liệu.

### **So sánh giữa RDBMS và MongoDB**



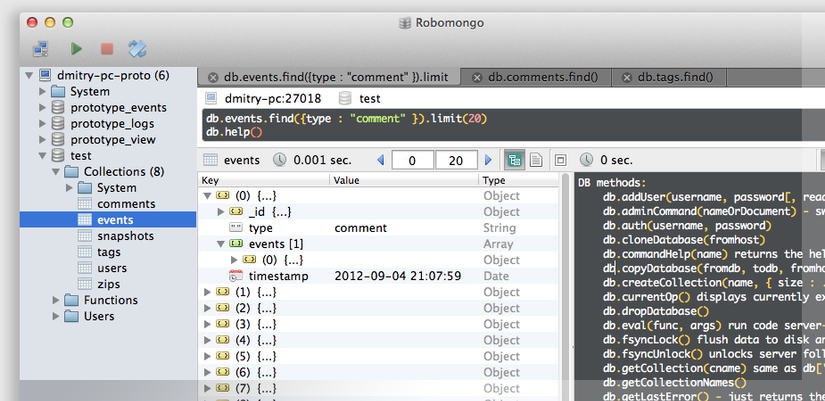
**Sử dụng MongoDB trong trường hợp:**

* Nếu website của bạn có tính chất INSERT cao Bởi vì mặc định MongoDB có sẵn cơ chế ghi với tốc độ cao và an toàn.Website của bạn ở dạng thời gian thực nhiều, nghĩa là nhiều người thao tác với ứng dung. Nếu trong quá trình load bị lỗi tại một điểm nào đó thì nó sẽ bỏ qua phần đó nên sẽ an toàn.
* Website bạn có nhiều dữ liệu quá Giả sử web bạn có đến 10 triệu records thì đó là cơn ác mộng với MYSQL. Bởi vì MongoDB có khả năng tìm kiến thông tin liên quan cũng khá nhanh nên trường hợp này nên dùng nó.
* Máy chủ không có hệ quản trị CSDL Trường hợp này thường bạn sẽ sử dụng SQLITE hoặc là MongoDB.

**Không nên sử dụng trong các trường hợp:**

* Các ứng dụng cần sử dụng nhiều transaction (như ngân hàng) do Mongodb không có cơ chế transaction (giao dịch) để phục vụ cho các ứng dụng ngân hàng
* Các ứng dụng cần SQL (sử dụng joins).

**Công cụ quản trị MongoDB:** RoboMongo



**Một số cân nhắc khi thiết kế schema trong MongoDB:**

* Thiết kế schema dựa trên yêu cầu của user
* Kết hợp các object vào một document nếu bạn sử dụng chúng với nhau. Nếu không thì tách riêng hẳn nó ra (nhưng cần phải chú ý là sau này nó sẽ không phải joins).
* Duplicate dữ liệu (nhưng có giới hạn) vì không gian đĩa là quá rẻ so với tốc độ xử lý.
* joins khi write, không joins khi read.
* Tối ưu schema cho hầu hết các case sử dụng thông thường
* Tập hợp các nhóm phức tạp vào schema

1. **TRIỂN KHAI DỰ ÁN THỰC TẾ**
2. **Tổng quan dự án**

Thiết kế website bán quần áo sử dụng ExpressJS và MongoDB

1. **Tìm hiểu nghiệp vụ trong hệ thống**

**Các đối tượng tham gia vào hệ thống:**

Hệ thống website gồm các đối tượng sau:

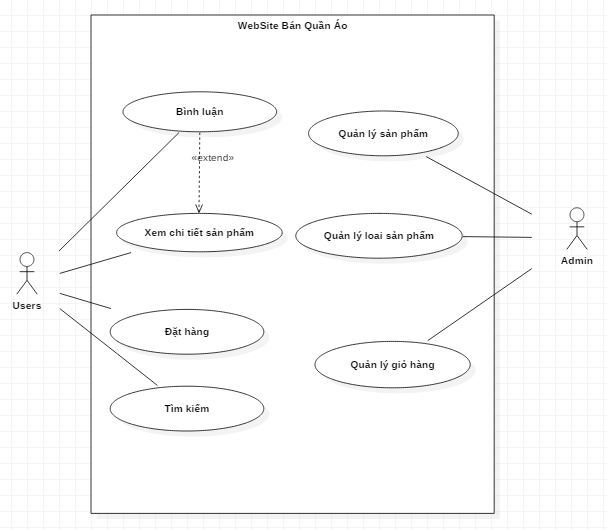
* Admin: Điều hành mọi hoạt động kỹ thuật trong hệ thống, cấu hình hệ thống và thiết lập các kết nối trong mạng người dùng. Cũng như quản lý toàn bộ nội dung của website.
* Người dùng: là những người đã truy cập vào hệ thống nói chung, ghé thăm trang web

**Tổng quan về hệ thống:**

Hệ thống gồm các chức năng cơ bản của một website bán hàng bao gồm:

* Xem sản phẩm, bình luận, đặt hàng, tìm kiếm cho người dùng
* Đăng nhập và quản lý trang web cho quản trị viên

1. **Biểu đồ Use - Case**



Biểu đồ Use- Case

1. **Mô tả Data Models**

* **Collection Giỏ hàng:**

**Schema({**

**name : String,**

**email : String,**

**sdt : String,**

**msg : String,**

**cart : Object,**

**st : Number**

**}**

* **Collection Thể loại:**

**Schema({**

**name: String,**

**nameKhongDau: String**

**}**

* **Collection Sản phẩm:**

**Schema({**

**name : String,**

**nameKhongDau : String,**

**img : String,**

**cateId : String,**

**des : String,**

**price : Number,**

**st : Number**

**}**

* **Collection Users**

**Schema({**

**fullname : String,**

**img : String,**

**email : String,**

**password : String,**

**}**

1. **KẾT LUẬN**

**Kết quả đạt được:**

- Trong quá trình thực hiện đề tài, em đã nghiên cứu một số lĩnh vực như sau:

* Tìm hiểu được được kiến thức cơ bản về NodeJS và MongoDB
* Hiểu và hoàn thiện quy trình thiết kế một hệ thống website bằng ExpressJS và MongoDB.

**Hạn chế :**

* Chưa nắm vững được một số kỹ thuật cũng như kiến thức về NodeJS và MongoDB

.

**Định hướng :**

* Tìm hiểu sâu thêm về NodeJS và MongoDB

Em rất mong nhận được thêm sự giúp đỡ và nhận xét của thầy để em có thể hoàn thiện hơn những ký năng cũng như kinh nghiệm.

Em xin chân thành cảm ơn thầy!