**BỘ GIÁO DỤC VÀ ĐÀO TẠO**

**TRƯỜNG ĐẠI HỌC MỞ TP. HỒ CHÍ MINH**

**KHOA CÔNG NGHỆ THÔNG TIN**

**ĐỒ ÁN NGÀNH**

WEBSITE GIA SƯ



Ngành : Khoa Học Máy Tính

Giảng viên hướng dẫn: Thầy Nguyễn Chí Thanh

Sinh viên thực hiện :

* Nguyễn Minh Tuệ - 1751010175
* Nguyễn Cao Tấn - 1751010131

Lớp: TH73

TP.HCM 2020

**BỘ GIÁO DỤC VÀ ĐÀO TẠO**

**TRƯỜNG ĐẠI HỌC MỞ THÀNH PHỐ HỒ CHÍ MINH**



**WEB GIA SƯ**

**Đồ Án Nghành**

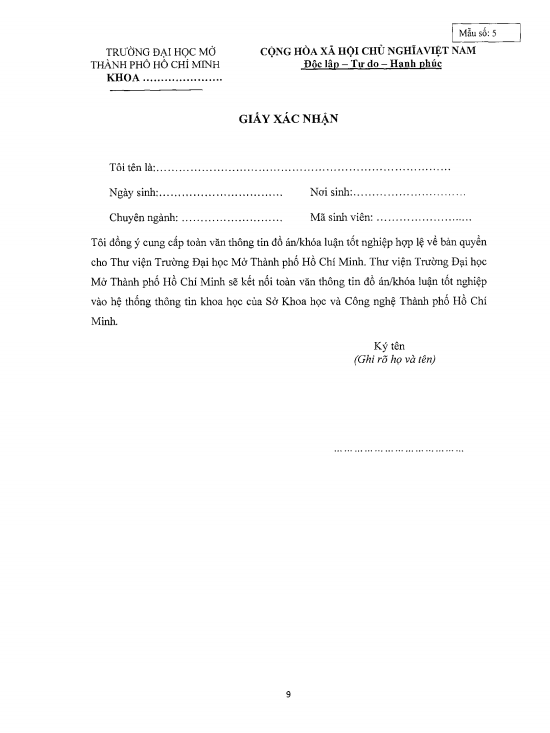
**NGÀNH: Khoa Học Máy Tính**

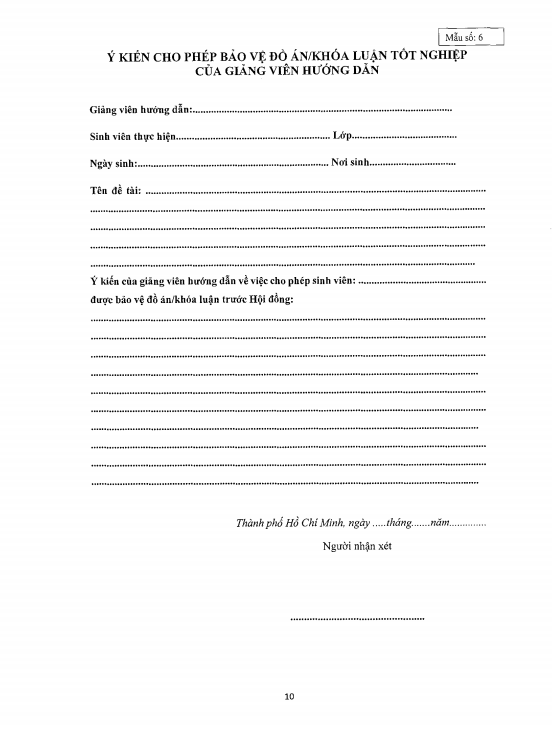
* Nguyễn Minh Tuệ - 1751010175
* Nguyễn Cao Tấn - 1751010131

**Giảng viên hướng dẫn: Thầy Nguyễn Chí Thanh**

**Lớp TH03**

**TP. HỒ CHÍ MINH, 2020**





[DANH MỤC HÌNH VẼ 13](#_Toc55569676)

[DANH MỤC BẢNG 14](#_Toc55569677)

[DANH MỤC TỪ VIẾT TẮT 15](#_Toc55569678)

[TÓM TẮT ĐỒ ÁN 1](#_Toc55569679)

[MỞ ĐẦU 2](#_Toc55569680)

[Chương 1.TỔNG QUAN VỀ INTERNET, WORLD WIDE WEB, HTML 4](#_Toc55569681)

[1.1 Khái niệm cơ bản về Internet 4](#_Toc55569682)

[1.1.1. Giới thiệu Chung 4](#_Toc55569683)

[1.1.2. Phân Loại 4](#_Toc55569684)

[1.2 World Wide Web 5](#_Toc55569685)

[1.2.1 Các Khái Niệm Cơ bản Về World Wide Web 5](#_Toc55569686)

[1.2.2 Cách tạo một trang Web 5](#_Toc55569687)

[1.2.3 Phân Loại Web 6](#_Toc55569688)

[1.2.4 Web Server 8](#_Toc55569689)

[1.2.5 Trình duyệt Web (Web Client hoặc Web Browser) 9](#_Toc55569690)

[1.3 HTML 9](#_Toc55569691)

[1.3.1 Giới thiệu về HTML 9](#_Toc55569692)

[1.3.2 Cấu trúc Trang 9](#_Toc55569693)

[1.3.3 Các thẻ HTML cơ bản 10](#_Toc55569694)

[Chương 2.TÌM HIỂU VỀ NGÔN NGỮ NODEJS 11](#_Toc55569695)

[2.1 Khái Niệm về NodeJs 11](#_Toc55569696)

[2.2 Tại sao chúng ta nên dùng NodeJS 11](#_Toc55569697)

[2.3 Sơ đồ hoạt động của NodeJS 12](#_Toc55569698)

[2.4 Các kiểu dữ liệu 13](#_Toc55569699)

[2.5 Biến và Hằng 15](#_Toc55569700)

[2.5.1 Biến 15](#_Toc55569701)

[2.5.2 Hằng 15](#_Toc55569702)

[2.6 Các đặc tính của NodeJS 16](#_Toc55569703)

[2.7 NPM 17](#_Toc55569704)

[2.8 Toán tử trong NodeJs 19](#_Toc55569705)

[2.8.1 Toán tử gán 19](#_Toc55569706)

[2.8.2 Toán tử số học 19](#_Toc55569707)

[2.8.3 Toán tử so sánh 20](#_Toc55569708)

[2.8.4 Toán tử logic 21](#_Toc55569709)

[2.8.5 Toán tử kết hợp 21](#_Toc55569711)

[2.9 Hàm trong NodeJS 22](#_Toc55569712)

[2.10 Câu lệnh rẽ nhánh 23](#_Toc55569713)

[2.10.1 Câu Lệnh If-ELSE 23](#_Toc55569714)

[2.10.2 Câu lệnh lồng nhau 23](#_Toc55569715)

[2.10.3 Switch-Case 24](#_Toc55569716)

[2.11 Đối tượng trong NodeJs (object) 25](#_Toc55569717)

[2.11.2 Thuộc tính và phương thức trong đối tượng 25](#_Toc55569718)

[2.11.4 Phương thức 26](#_Toc55569719)

[2.11.5 Gọi thuộc tính và phương thức 27](#_Toc55569720)

[2.11.6 Đối tượng trong đối tượng 27](#_Toc55569721)

[2.11.7 Mảng trong đối tượng 28](#_Toc55569722)

[2.12 Khởi tạo Server trong NodeJs với HTTP module 29](#_Toc55569723)

[2.12.1 Giới thiệu về HTTP module 29](#_Toc55569724)

[2.12.2 Khởi tạo Server với HTTP module 29](#_Toc55569725)

[2.12.3 Các phương thức và thuộc tính được dùng trong resquest và response 30](#_Toc55569726)

[2.13 Đọc ghi file bằng module fs trong Node.js 31](#_Toc55569727)

[2.13.1 Giới thiệu về module fs 31](#_Toc55569728)

[2.13.2 Đọc files với module fs 31](#_Toc55569729)

[2.13.3 Ghi file 32](#_Toc55569730)

[2.14 Module URL 32](#_Toc55569731)

[2.14.1 Khai báo sử dụng Module URL 33](#_Toc55569732)

[2.15 Upload file trong NodeJs 33](#_Toc55569733)

[2.15.1 Tổng quan 33](#_Toc55569734)

[2.15.2 cài đặt module formidable 33](#_Toc55569735)

[2.15.3 code upload 33](#_Toc55569736)

[2.16 Sự kiện trong NodeJs 34](#_Toc55569737)

[2.16.1 Event loop 34](#_Toc55569738)

[2.17 Tạo client request trong NodeJs 35](#_Toc55569739)

[2.17.1 Cài đặt 35](#_Toc55569740)

[2.17.2 các phương thức mà module request hỗ trợ: 36](#_Toc55569741)

[Chương 3.CƠ SỞ DỮ LIỆU MONGODB 38](#_Toc55569742)

[3.1 Giới thiệu về MongoDB 38](#_Toc55569743)

[3.1.1 Lợi thế của Mongodb 38](#_Toc55569744)

[3.1.2 đặc điểm của Mongodb 38](#_Toc55569745)

[3.2 Mục đích sử dụng cơ sở dữ liệu Mongodb 38](#_Toc55569746)

[3.3 Thuật ngữ trong Mongodb 39](#_Toc55569747)

[3.4 Kiểu dữ liệu trong Mongodb 40](#_Toc55569748)

[3.5 Một số câu lệnh truy vấn cơ bản của Mongodb 41](#_Toc55569749)

[3.6 Truy vấn có điều kiện trong MongoDB 42](#_Toc55569750)

[3.7 MongoDB Replication 43](#_Toc55569751)

[3.8 Cài đặt mongodb 43](#_Toc55569752)

[3.9 Kết nối NodeJs với mongodb 43](#_Toc55569753)

[3.10 Đóng kết nối NodeJs đến Mongodb 44](#_Toc55569754)

[3.11 Sử dụng Mongodb 44](#_Toc55569755)

[Chương 4.PHÂN TÍCH THIẾT KẾ HỆ THỐNG 45](#_Toc55569756)

[4.1 Tác nhân và use case hệ thống 45](#_Toc55569757)

[4.1.1 Các tác nhân (Actor) 45](#_Toc55569758)

[4.2 Các use case 45](#_Toc55569759)

[4.2.1 Đăng nhập: 45](#_Toc55569760)

[4.2.2 Đăng ký: 45](#_Toc55569761)

[4.2.3 Đăng tin tuyển dụng: 45](#_Toc55569762)

[4.2.4 Tạo hồ sơ gia sư: 45](#_Toc55569763)

[4.2.5 Nạp tiền vào tài khoản 45](#_Toc55569764)

[4.2.6 Tra cứu thông tin 46](#_Toc55569765)

[4.2.7 Hỗ trợ khách hàng 46](#_Toc55569766)

[4.2.8 Thay đổi quy định 46](#_Toc55569767)

[4.2.9 Lập báo cáo 46](#_Toc55569768)

[4.2.10 Use Case 46](#_Toc55569769)

[4.2.11 Active Diagram 46](#_Toc55569770)

[4.2.12 Sequence Diagram 46](#_Toc55569771)

[4.2.13 Class Diagram 46](#_Toc55569772)

[4.2.14 Mô Hình Database 46](#_Toc55569773)

[PHỤ LỤC 48](#_Toc55569774)

[TÀI LIỆU THAM KHẢO 49](#_Toc55569775)

Contents

**LỜI CẢM ƠN**

Điều đầu tiên em xin phép được gửi lời cảm ơn đến Thầy Nguyễn Chí Thanh Giảng viên môn học đã nhiệt tình giúp đỡ, đóng góp ý kiến tận tình. Mặc dù thầy rất bận rộn nhưng luôn dành thời gian giúp đỡ cũng như chỉ bảo cho em trong khoảng thời gian làm đồ án.

Em cũng xin chân thành cảm ơn các thầy cô giáo trong Khoa Công Nghệ Thông Tin, văn phòng công tác sinh viên nói chung, Nghành Khoa học máy tính nói riêng. Đã dạy và tạo điều kiện giúp đỡ em có kiến thức vững vàng về các môn đại cương cũng như các môn chuyên nghành để chúng em có thể thực hiện bài đồ án thành công tốt đẹp.

Cuối cùng, em xin chân thành gửi lời cảm ơn đến gia đình và bạn bè, luôn tạo điều kiện thuận lợi, giúp đỡ, quan tâm, động viên em trong suốt quá trình hoàn thành bài đồ án.

………..Ngày….Tháng….Năm……..

**NHẬN XÉT CỦA GIÁO VIÊN HƯỚNG DẪN**

|  |  |  |  |
| --- | --- | --- | --- |
| **TT** | **Hình** | **Tên hình** | **Trang** |
| 1 | *Hình 1.1* | sơ đồ hoạt động của Web Động. | 3 |
| 2 | *Hình 1.2* | sơ đồ hoạt động của Web tĩnh. | 3 |
| 3 | *Hình 1.3* | sơ đồ hoạt động của Web Server. | 4 |
| 4 | *Hình 2.1* | Sơ đồ hoạt động của NodeJs. |  |
| 5 | *Hình 2.2* | Ảnh minh họa khởi tạo một dự án. |  |
| 6 | *Hình 2.3* | Ảnh minh họa cách khởi tạo Package. |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

# DANH MỤC HÌNH VẼ

# DANH MỤC BẢNG

[Bảng 2.1: Bảng toán tử số học. 3](#_Toc367742567)

[Bảng 2.2: Bảng toán tử so sánh. 4](#_Toc367742568)

[Bảng 2.3: bảng toán tử logic 4](#_Toc367742568)

[Bảng 2.4: bảng phương thức module request. 3](#_Toc367742567)

[Bảng 3.1: bảng thuật ngữ trong Mongodb 4](#_Toc367742568)

[Bảng 3.2: bảng kiểu dữ liệu trong MongoDB 4](#_Toc367742568)

[Bảng 3.3: bảng cú pháp truy vấn có điều kiện trong MongoDB 3](#_Toc367742567)

# DANH MỤC TỪ VIẾT TẮT

|  |  |  |
| --- | --- | --- |
| **TT** | **Từ viết tắt** | **Ý nghĩa** |
| 1 | CNTT | Công nghệ thông tin |
| 2 | NCKH | Nghiên cứu khoa học |
| 3 | DAMH | Đồ án môn học |
| 4 | SDT | Số điện thoại |
| 5 | CMND | Chứng minh nhân dân |
| 6 | CSDL | Cơ sở dữ liệu |

# TÓM TẮT ĐỒ ÁN

Với mức độ xây dựng phức tạp và quy mô ứng dụng rộng rãi, cho nên vấn đề về thời gian để xây dựng “Website Gia sư” chỉ tạm thời dừng lại ở mức tìm hiểu về ngôn ngữ lập trình NodeJS, và ứng dụng vào thực tế thực ng hiệp trang web thuê gia sư này.

Website có mục đích nhằm cung cấp cho các khác hàng (học sin, sinh viên) các thông tin chính xác về môn học giảng dạy của các gia sư (Tên, môn học giảng dạy, lời giới thiệu về bản thân) và các thức thuê gia sư. Các thông tin của gia sư sẽ được cập nhật thường xuyên và nhanh chóng nhất có thể. Vì vậy có thể tạo điều kiện cho học sinh, sinh viên tiết kiệm được nhiều thời gian tìm người dạy kèm, giá cả theo ý muốn đồng thời kèm theo các thông tin mà khách hàng thực sự cần.

Vì trong thời gian thực hiện còn một số hạn chế cho nên website thuê gia sư chỉ dừng lại ở các chức năng cơ bản như đăng nhập, đăng ký, tạo lớp, tạo hồ sơ gia sư, tra cứu thông tin . Trong tương lai, hệ thống website sẽ tiếp tục phát triển hơn nữa và mở rộng phạm vi chức năng để hoàn thiện được một website chuyên nghiệp.

# MỞ ĐẦU

Hiện nay, việc ứng dụng công nghệ thông tin vào các lĩnh vực của xã hội và việc tin học hóa được xem là một yếu tố quan trọng hàng đầu của các tổ chức, các công ty, nhà nước, nó đóng vai trò cốt yếu trong việc tạo ra những bước đột phá trong công nghệ.

Song song với việc phát triển không ngừng nghỉ về khoa học máy tính, kỹ thuật máy tính và mạng truyền thông, thương mại điện tử. đã được chinh phục bởi những công nghệ có đặc tính kỹ thuật tiên tiến, và những công nghệ vượt trội có mặt trên thế giởi hiện nay. Ngay tại thời điểm này, internet đã trở thành một sản phẩm có giá trị đặc biệt mà không có đối thủ thay thế và không thể phủ nhận được tầm quan trọng của nó. Internet là nền tảng cốt yếu quan trọng nhất để truyền tải, trao đổi thông tin trên phạm vi toàn cầu.

Ngày nay, chúng ta sẽ nhận thấy mọi việc sẽ dễ dàng hơn khi chúng ta có Internet, mọi việc liên quan đến thông tin tài nguyên sẽ được truyền tải nhanh chóng. Chỉ cần bạn sở hữu một thiết bị thông minh như Laptop hoặc điện thoại có kết nối internet và kèm theo đó là một dòng dữ liệu theo chính nhu cầu người sử dụng muốn, thì lập tức mọi thứ có liên quan đến dữ liệu người dùng sẽ xuất hiện ngay trước mắt chỉ trong vài giây, có đủ các thông tin hình ảnh, âm thanh và các tài nguyên khác mà người sử dụng cần.

Nhờ có Internet mà chúng ta có thể tiện lợi thực hiện được nhiều công việc với tốc độ tối đa và chi phí thấp hơn so với các hoạt động không có internet. Chính vì vậy đã thúc đẩy mạnh sự phát triển thịnh vượng về thương mại điện tử trên khắp thế giới, giúp tăng nhanh năng xuất chất lượng cuộc sống con người.

Giở đây, thương mại điện tử đã khẳng định vị thế và vai trò của mình trong hoạt động sản xuất, kinh doanh thúc đẩy sự phát triển trong lĩnh vực công nghệ. Đối với doanh nghiệp tư nhân và nhà nước, lớn nhỏ, việc quảng bá marketing và giới thiệu sản phẩm đến khách hàng là một vấn đề cần thiết cho doanh nghiệp. vậy việc quảng bá cho doanh nghiệp trước tiên là xây dựng một website để quảng bá tất cả các sản phẩm của doanh nghiệp.

Theo nhu cầu thiết yếu của thị trường cũng như học tập và phát triển hơn về công nghệ này, nhóm em đã quyết định thực hiện đồ án với chủ đề là “xây dựng website gia sư” cho người đang có nhu cầu tuyển gia sư.

Người sử dụng khi truy cập website sẽ được chia làm 2 dạng, khách hàng thuê gia sư và gia sư. Khách hàng truy cập vào website để tìm thông tin gia sư phù hợp. gia sư truy cập website để tìm khách hàng phù hợp. tất cả các thông tin gia sư sẽ được chia sẽ cho khách hàng khi cần.

Với sự chỉ bảo và hướng dẫn tận tình của thầy Nguyễn Chí Thanh em đã thực hiện hoàn thành bài báo cáo đồ án này một cách hoàn thiện. chúng em đã cố gắng rất nhiều để tìm hiểu, phân tích hệ thống và thiết kế nhưng không thể không tránh khỏi những thiếu sót và lỗi sai. Chúng em rất mong quý thầy quý cô cho chúng em lời góp ý qua đó em có thể sửa những lỗi sai và hoàn thành bài báo cáo tốt hơn nữa. chúng em xin chân thành cảm ơn quý thầy cô.

Sinh viên thực hiện: Nguyễn Cao Tấn

Nguyễn Minh Tuệ

Lớp: TH73

……..Ngày…..Tháng…..Năm

# TỔNG QUAN VỀ INTERNET, WORLD WIDE WEB, HTML

* 1. Khái niệm cơ bản về Internet
     1. Giới thiệu Chung

Internet hay Mạng là một hệ thống kết nối thông tin toàn cầu có thể được truy nhập công đồng gồm các mạng máy tính được liên kết lại với nhau. Hệ thống này sẽ truyền thông tin theo kiểu nối chuyển gói dữ liệu (packet switching) và dựa trên một giao thức đó là liên mạng đã được chuẩn hóa (giao thức IP).

Hệ thống này bao gồm rất nhiều mạng máy tính nhỏ hơn của các doanh nghiệp, của các nhà, viện nghiên cứu và các trường đại học trên thế giới, của người dùng riêng và các chính phủ trên toàn cầu, được liên kết bằng một loạt các công nghệ như mạng điện tử, không dây và mạng quang. Internet đem lại cho chúng ta hàng loạt các tài nguyên và dịch vụ thông tin, chẳng hạn như các tài liệu và ứng dụng siêu văn bản được liên kết với nhau bằng World Wide Web, điện thoại, thư điện tử, và chia sẻ file, vân vân.

Hệ thống internet được thành lập từ những năm 80 bắt nguồn từ mạng APARRNET của bộ quốc phòng hoa kỳ.

Khi kết nối mạng internet chúng ta có thể tìm được rất nhiều tài nguyên để nghiên cứu, học tập, trao đổi thông tin, đặt mua, bán hàng một cách rất dễ dàng bằng một vài dòng lệnh và một lần click chuột.

Từng máy tính cá nhân trên Internet được gọi là một host. Mỗi host có chức năng và khả năng như nhau về truyền thông đến máy khác trên Internet. Một số host khác được nối đến mạng bằng đường dẫn Dial-up tạm thời, còn số khác được nối bằng đường dẫn mạng thật (như Tokenring, Ethernet...).

* + 1. Phân Loại

Tất cả máy tính trên mạng được chia làm 2 nhóm đó chính là Client và Server.

- Máy Client: là tất cả Máy khách trên mạng và chứa các chương trình của Client

- Máy Server: Máy phục vụ hay được gọi là Máy chủ. Chứa các tài nguyên (tập tin, tài liệu...) và chương trình Server dùng chung cho tất cả máy khách. Server sẽ luôn ở trạng thái chờ để yêu cầu và đáp ứng yêu cầu của Client.

- Internet Server: Là tất cả server cung cấp dịch vụ Internet như FPT Server, Web Server, Mail Server…

Các dịch vụ phổ biến được dùng trên Internet

- – Tổ chức, khai thác nguồn thông tin trên web (WWW – World Wide Web): người dùng, khách hàng có thể dễ dàng truy cập để xem nội dung các trang web khi máy tính được kết nối với Internet

- Dịch vụ ELECTRONIC MAIL (được viết tắt là Email và được gọi là thư điện tử) đây là dịch vụ trao đổi một số thông tin trên Internet thông qua các hộp thư điện tử, được sử dụng rộng rãi và rất phổ biến, tiện lợi, nhanh chóng, với chi phí thấp.

- Dịch vụ FPT

Để trao đổi và truyền thông với tất cả các máy tính khác, hầu như mọi máy tính trên Internet phải được hỗ trợ giao thức chung đó là TCP/IP (Transmission Control Protocol/Internet Protocol - là giao thức điều khiển truyền, gửi dữ liệu và giao thức Internet), đồng thời một giao thức đồng bộ, cho phép truyền thông tin, thông điệp từ nhiều nguồn khác nhau và đến nhiều đích khác nhau. Môt ví dụ nhỏ, ta có thể vừa lấy thư từ trong hộp thư và đồng thời truy cập đến trang Web. TCP mang lại sự đảm bảo tính an toàn tương đối về dữ liệu, IP là giao thức mà nó có thể chi phối phương thức vận chuyển các nguồn dữ liệu trên Internet.

* 1. World Wide Web
     1. Các Khái Niệm Cơ bản Về World Wide Web

Như đã được đề cập ở trên, “www” chính là tên viết tắt của World Wide Web và được gọi tắt là Web, nơi đây ta có thể được hiểu là một không gian thông tin toàn cầu. Tại đây, mọi người có thể sử dụng để viết, đọc, chia sẻ thông tin thông qua mạng Internet.

Không gian này được xác định nhờ vào đường dẫn URL được gán và liên kết với nhau bởi các siêu liên kết khi truy cập thông qua Internet. World Wide Web còn là thuật ngữ có nhiệm vụ đề cập tới tất cả các trang HTML kết nối với nhau, có thể truy cập thông qua Internet.

Web là một dịch vụ được xem là phổ biến nhất hiện nay trên Internet, 85% các giao dịch trên Internet ước lượng thuộc về WWW. Ngày nay số lượng trang Web trên thế giới đã đạt tới con số rất lớn. WWW còn cho phép bạn truy xuất thông tin hình ảnh, văn bản, âm thanh, video trên toàn thế giới. Thông qua các Website, các công ty có thể giảm được thiểu tối đa chi phí in ấn và phân phát tài liệu cho khách hàng ở nhiều nơi.

* + 1. Cách tạo một trang Web

Chúng ta có rất nhiều cách để tạo một trang Web.

* Chúng ta có thể tạo một trang Web ngay trên bất kì chương trình xử lý văn bản nào.
* Ngoài ra còn có thể thiết kế Web bằng các phần mềm chuyên nghiệp như: DreamWeaver, Visual studio code, Nescape Editor, FrontPage.
* Phầm mềm DreamWeaver và Visual Studio code là 2 loại phần mêm chuyên nghiệp sẽ giúp chúng ta thiết kế một trang web nhanh và dễ dàng hơn rất nhiều vì phần lớn các thư viện và mã lệnh được cung cấp sẵn trong code của chúng ta nên khi thao tác chúng ta sẽ thực hiện một cách rất dễ dàng.
* Tạo Web bằng cách viết mã nguồn bởi trình soạn thảo văn bản như: NodePad, WorldPad. Đây là những chương trình soạn thảo trong văn bản có sẵn trong Window.
* Ngoài ra còn có thể thiết kế bằng cách dùng Web Wizard và những công cụ của Word 1997 hoặc World 2000 để thiết kế.

Nhằm xây dững một trang web có tính thương mại và ứng dụng cao được gọi là hoàn chỉnh. Thì chúng ta phải kết hợp giữa những phương thức sau: Server Script và Client Script (kịch bản trên máy chủ và kịch bản trên máy khách) cộng thêm một trong những trình xử lý cơ sở dữ liệu sau: MS Access, SQL Server, MySQL và Oracle.

Ngoài ra khi chúng ta muốn triển khai ứng dụng web trên Internet, ngoài các điều kiện về cấu hình phần cứng, thì chúng ta cần có trình chủ Web hay được gọi là Web Server.

* + 1. Phân Loại Web
       1. Web Động

Và ở đây Web động được hiểu là gì?

* Web Động là một tập hợp các dữ liệu số hóa được tổ chức thành các cơ sở dữ liệu, các dữ liệu được số hóa được gọi ra trình diễn trên các trang web dưới dạng văn bản, âm thanh, hình ảnh. Web còn có thêm các phần truy xuất dữ liệu và xử lý thông tin còn website tĩnh thì không.
* Khác với web tĩnh, web động luôn luôn có thông tin mới do các thông tin này sẽ được cập nhật bởi các phần mềm quản trị website do các công ty thiết kế website cung cấp. Các thông tin mới này sẽ được lưu vào cơ sở dữ liệu (Database) của website và đưa ra sử dụng dựa theo yêu cầu của người dùng.
* Về cơ bản nội dung của trang Web động như một trang Web tĩnh, ngoài ra còn có thể thao tác với CSDL (cơ sở dữ liệu) để đáp ứng nhu cầu phức tập của một trang Web. Sau khi nhận được yêu cầu từ Web Client, chẳng hạn như một truy vấn từ một CSDL đặt trên Server, ứng dụng Internet Server sẽ truy vấn CSDL này, và tạo một trang HTML có chứa kết quả truy vấn rồi sau đó gửi trả cho người dùng.

|  |
| --- |
| URL |
| Form |

CGI

Yêu Cầu Dữ liệu trả về

Kết Nối Dữ liệu trả về

CSDL

Hình 1.1: sơ đồ hoạt động của Web Động.

* + - 1. Web Tĩnh

Hoạt Động của trang Web tĩnh được hoạt động theo mô hình sau:

Browser gửi yêu cầu

Browser

Server trả tài liệu

Hình 1.2 : sơ đồ hoạt động của Web tĩnh.

Là trang web sử dụng hoàn toàn ngôn ngữ HTML, sau khi trang HTML được tải từ máy chủ xuống, trình duyệt sẽ có nhiệm vụ biên dịch mã và hiển thị nội dung của trang web, và người dùng dường như không thể tương tác với trang web.

Ưu Điểm :

* Tốc độ truy cập nhanh vì đó là những file HTML
* Chi phí đầu tư giá rẻ vì không phải chi trả nhiều tiền cho coder
* Giao diện có thể design theo một hướng đột phá mới lạ
* Thân thiện với môi trường computer vì có thể đặt tên file tùy ý

Nhược Điểm :

* Rất khó để quản lý nội dung.
* Và khó để nâng cấp hoặc bảo trì.
  + 1. Web Server

Webserver: là một máy tính có kết nối vào Internet và run các phần mềm được thiết kế. Webserver đóng vai trò một chương trình xử lí các nhiệm vụ xác định, xử lí tổ hợp dữ liệu, như tìm trang thích hợp, kiểm tra dữ liệu hợp lệ.... Webserver còn là nơi lưu trữ cơ sở dữ liệu (Database), là phần mềm đảm nhiệm vai trò Server cung cấp dịch vụ Web.

Webserver hỗ trợ các công nghệ nổi tiến khác nhau:

* Apache HTTP Server, Nginx, IIS (Internet Information Services), lighttpd….

Cách thức hoạt động của một Web Server:

Web Server

Web Browser

Web Server trả Web Browser

về dịch vụ Web gửi Yêu Cầu

Hình 1.3: sơ đồ hoạt động của Web Server.

* + 1. Trình duyệt Web (Web Client hoặc Web Browser)

Trình duyệt Web là được xem công cụ truy xuất dữ liệu trên mạng, là phần mềm giao diện trực tiếp với người sử dụng. Nhiệm vụ của Web Browser là nhận các yêu cầu của người sử dụng, gửi tất cả các yêu cầu đó qua mạng tới các Web Server và nhận các dữ liệu cần thiết từ Server để hiển thị lên màn hình. Để vận hành và sử dụng dịch vụ WWW, Client hay người dùng cần có 1 chương trình duyệt Web, kết nối vào Internet thông qua một ISP. Các trình duyệt thông dụng hiện nay là: Microsoft Internet Explorer(IE), Mozilla Firefox và Google Chrome… Trong đó Internet Explorer là một trình duyệt chuẩn cho phép trình bày nội dung được Web server cung cấp, được phép đăng kí tới bất kì Website nào trên Internet, hỗ trợ trình bày trang Web.

* 1. HTML
     1. Giới thiệu về HTML

HTML là chữ viết tắt của HyperText Markup Language được World Wide Web Consortium đặt ra. Một trang web là sự kết hợp giữa các văn bản kèm theo là các thẻ HTML. Các tập tin HTML đều có đuôi là .html hoặc .htm.

HTML giúp định dạng trong văn bảng nhờ các thẻ. Các thể HTML có thể liên kết từ hoặc cụm từ với các tài liệu khác trên Internet. Hầu hết cái thể HTML đầu có thẻ đóng và thẻ mở. thẻ đóng được kí hiệu kèm dấu “/”

Ví dụ: <head></head>

Ngôn ngữ HTML được qui định cú pháp không phân biết chữ hoa và chữ thường. và không được chứa các khoảng trắng trong định nghĩa thẻ.

Ví dụ: có thể khai báo

<html></HTML>

* + 1. Cấu trúc Trang

Một trang HTML có cấu trúc trang khá đơn giản và được quy định như sau:

<html>

<head>

<title></title>

</head>

<body>

(các loại thẻ HTML và nội dụng hiển thị)

</body>

<footer></footer>

</html>

* + 1. Các thẻ HTML cơ bản

1. **Thẻ <title>...</title>:** Tạo tiêu đề trang trên thanh tiêu đề, đây là thẻ bắt buộc và nằm trong thẻ <head>.
2. **Thẻ <head>...</head>:** Tạo đầu mục trang
3. **Thẻ <body>...</body>:** Tất cả các thông tin khai báo có trong thẻ <body> đều có thể được xuất hiện trên trang Web. Những thông tin này được nhìn thấy trên trang Web.
4. **Các thẻ định dạng khác**. Thẻ <p>…</p>: Tạo một đoạn hoặc một câu mới. Thẻ <font>... </font>: Thay đổi kích cỡ, phông chữ và màu kí tự…
5. **Thẻ định dạng bảng:** <table>…</table> Đây là loại thẻ định dạng một bảng trên trang Web. khi khai báo thẻ này, bạn có thể phải khai báo các thẻ cột <td> và thẻ hàng <tr> cùng với các thuộc tính đi kèm của nó.
6. **Thẻ hình ảnh <img>:** Cho phép bạn được chèn hình ảnh vào một trang Web. Thẻ này là loại thẻ thuộc loại không có thẻ đóng.
7. **Thẻ liên kết** <a>... </a>: Là thẻ dùng để liên kết các trang Web hoặc liên kết đến địa chỉ Internet.
8. **Các thẻ Input**: Thẻ Input cho phép người dùng nhập dữ liệu hay chỉ thị thực thi một hành động, thẻ Input bao gồm các loại thẻ như: text, password, submit, button, reset, checkbox, radio, image.
9. **Thẻ Textarea:** < Textarea>.... < \Textarea>: Thẻ Textarea cho phép client (người dùng) nhập liệu văn bản. loại thẻ này không thể giới hạn chiều dài lớn nhất trên một trang Web.
10. **Thẻ Select:** Loại thẻ Select cho phép người được dùng chọn phần tử trong các tập phương thức đã định nghĩa trước. thẻ Select còn cho phép người dùng (Client) chọn một phần tử nào đó trong danh sách phần tử thì thẻ Select sẽ giống như combobox. thẻ Select còn cho phép người dùng (Client) chọn nhiều phần tử cùng một lần trong danh sách phần tử, thẻ Select đó là dạng listbox.
11. **Thẻ Form:** Khi bạn muốn submit dữ liệu người dùng nhập từ Website phía người dùng (Client) đến phía máy chủ (Server), có hai cách để làm điều nàu ứng với hai phương thức POST và GET trong thẻ form. Trong một Website còn có thể có nhiều (thẻ Form) khác nhau, nhưng các loại thẻ Form này không được phép lồng nhau, mỗi thẻ form được khai báo hành động (action) chỉ đến một trang khác.

# TÌM HIỂU VỀ NGÔN NGỮ NODEJS

* 1. Khái Niệm về NodeJs

- Nodejs là một nền tảng độc lập có mã nguồn mở được phát triển và xây dựng ở trên Javascript Runtime của chrome. NodeJS được viết bởi Ryan Dahl vào năm 2009 dựa trên Javascript engine V8 của Google Qua đó chúng ta có thể xây dựng được các ứng dụng mạng nhanh chóng và rất dễ dàng mở rộng.

- về bản chất NodeJS còn được xem là JavaScript. Trước khi NodeJS ra đời JavaScript chủ yếu được sử dụng để nhúng vào HTML để sử dụng trên các website để người dùng có thể tương tác với trình duyệt. Cho đến hiện nay Javascript đang có một số cải tiến rất đáng kể. trước đó javascript rất thỉnh thoảng được sử dụng đến để tạo một hiệu ứng cho trang Web. Nhưng giờ đây Javascript đã trở thành ngôn ngữ chính và được sử dụng nhiều như các ngôn ngữ khác: C, Ruby, PHP, vân vân.

- NodeJS cho phép chúng ta sử dụng JavaScript trên Server và nó còn cho phép chúng ta viết JavaScript bên ngoài trình duyệt.

- NodeJS Khác với hầu hết các chương trình Javascript, Nodejs không chạy trên một trình duyệt nhưng lại chạy trên Server. Nodejs sử dụng nhiều kỹ thuật chi tiết của CommonJS. NodeJS cung cấp một môi trường REPL cho kiểm thử tương tác.

- Tất nhiên Nodejs thừa hưởng hầu như toàn bộ sức mạnh vốn có của Javascript và đã cho chúng ta thấy một cách nhìn toàn diện một cách mới mẻ về việc phát triển web động bằng NodeJS.

* 1. Tại sao chúng ta nên dùng NodeJS

Để thiết kế một website động chúng ta có thể sử dụng rất nhiều ngôn ngữ khác nhau để viết nên một website hoàn chỉnh, mặc dù tính chất, tính năng và dùng các cầu hình khác nhau nhưng tất cả đều đưa ra một kết quả giống nhau. Có khá nhiều ngôn ngữ phổ biến hiện nay như: PHP, Java, Perl, ASP.NET…. và còn rất nhiều ngôn ngữ khác. vậy tại sao chúng ta nên chọn NodeJS để làm ngôn ngữ chính để thực hiện những website này. Và sau đây là những điểm mạnh mà NodeJS đem lại cho các coder một môi trường làm việc thuận lợi, tiết kiệm thời gian và có hiệu quả cao.

* Các ứng dụng của NodeJS được viết hầu hết bằng Javascript, và ngôn ngữ này chính là ngôn ngữ khá thông dụng cho mọi Coder. Trích nguyên văn câu nói của Ryan Dahl người đã tạo ra NodeJS “Javascript có những đặc tính mà làm cho nó rất khác biệt so với các ngôn ngữ lập trình động, và cụ thể là javascript không có khái niệm về đa luồng, tất cả là đơn luồng và hướng sự kiện.”
* NodeJs chạy đa nền tảng từ phía Server, sử dụng kiến trúc hướng sự kiện Event – Driven và cơ chế non-Blocking I/O làm cho NodeJS rất nhẹ và có tính hiệu quả.
* Chúng ta có thể chạy ứng dụng Nodejs ở bất kỳ đâu trên máy Mac, Window, Linux, hơn nữa cộng đồng người sử dụng Nodejs rất lớn và nguồn tài nguyên là hoàn toàn miễn phí. NodeJs hiện đang là một ngôn ngữ thật sự hot trong nghành CNTT, các package đều hoàn toàn free: <https://www.npmjs.com/>
* Các ứng dụng của NodeJS đáp ứng rất tốt về thời gian thực và chạy đa nền tảng, đa thiết bị.

đây là những điểm mạnh mà NodeJS mang lại cho người dùng. Và đó là gì do người dùng nên sử dụng NodeJS để tạo ra những WebSite hoàn thiện.

* 1. Sơ đồ hoạt động của NodeJS

Ngoài ra Node.js cung cấp một môi trường phía máy chủ mà từ trước tới nay Javascript không làm được, ngoài ra NodeJs còn cho phép chúng ta cũng sử dụng ngôn ngữ JavaScript để tạo các trang web. Nói chung nó thay thế các ngôn ngữ máy chủ như Java EE, PHP, v.v…

Client Server

1. Máy Khách gửi yêu cầu tạo một trang web từ máy chủ

2. môi trường nodejs chạy javascript để tạo html (hoặc để thực hiện các hoạt động khác)

3. máy chủ gửi trang html tạo cho máy khách

4. mã hóa javascript vẫn có thể được chạy bởi trình duyệt của khách hàng để sửa đổi trang html

Hình 2.1: Sơ đồ hoạt động của NodeJs.

Khi người dùng truy cập vào website ở máy khách sẽ gửi yêu cầu về máy chủ, máy chủ sẽ xử lý chúng theo hướng dẫn đã được mã hóa sẵn. ở server môi trường NodeJS sẽ chạy Javascript để tạo ra HTML, sau đó máy chủ gửi một dữ liệu thích hợp (trang HTML đã tạo) cho máy khách. Khi đó máy client vẫn có thể sửa đổi trang vì mã hóa Javascript vẫn có thể được chạy bởi chính trình duyệt của client.

* 1. Các kiểu dữ liệu

Vì được viết dựa trên JavaScript nên rất nhiều tính năng trong NodeJS được kế thừa từ ngôn ngữ JavaScript. chúng ta sẽ tìm hiểu kiểu dữ liệu, cú pháp, các vòng lặp và cấu trúc điều khiển trong NodeJS.

Gồm các kiểu dữ liệu:

* Kiểu Undefined:

Typeof instance == “undefined”

(Một biến chưa được gán giá trị có giá trị undefined.)

Ví dụ:

var x;(tạo một biến nhưng không gán giá trị)

console.log("x's value is", x)

* Kiểu Boolean:

Typeof instance == “boolean”

(Kiểu boolean mang hai giá trị logic là: true, và false.)

Ví dụ:

if (điều kiện boolean) {

console.log(" trả về giá trị là true");

} else {

console.log("trả về giá trị là false");

}

* Kiểu số:

Typeof instance == “number”

(Không có kiểu số nguyên. Ngoài việc chứa giá trị là dấu phẩy động, kiểu số mang ba giá trị biểu tượng: +Infinity, -Infinity, and NaN (not-a-number).)

* Kiểu chuỗi:

Typeof instance == “string”

(Tạo một chuỗi con của chuỗi ban đầu bằng cách ghép từng ký tự hoặc dùng String.substr().)

* Kiểu Biglnt:

Typeof instance == “bigint”

(đại diện cho các giá trị số nguyên với độ chính xác (precision) bất kỳ. BigInt cho phép ta có thể lưu và tính toán trên các số nguyên lớn mà nó có thể lớn hơn cả giới hạn an toàn của kiểu Number. Số bigint không thể chung số với số Number để tính toán. Khi đó, lỗi TypeErrror sẽ xảy ra.)

* Kiểu Symbol:

Typeof instance == “symbol”

Mỗi Symbol là một giá trị đơn nhất và có thể được dùng như một khóa của một Object và bất biến.

* Kiểu Mảng một chiều:

Array.Length (chiều dài của Array(mảng) trong constructor có giá trị là 1)

Array.prototype(được bổ sung các thuộc tính cho hầu hết tất cả các đối tượng Array(mảng).)

Array.from(): (Tạo một Mảng mới từ đối tượng giống iterable hoặc mảng.)

Array.isArray(): (Trả về giá trị true nếu một biến là mảng, và false nếu không phải là mảng.)

Array.of(): (Tạo một Array(mảng) mới với bất kể số hoặc loại của các đối số và một số lượng các đối số.)

Ví dụ:

var msgArray = [];

msgArray [0] = "Hello";

msgArray [99] = 'World';

if (msgArray.length === 100) {

console.log('Chiều dài là 100');

}

* Mảng 2 chiều:

Ví dụ:

const allClasses = [

['nguyen v a', 'nguyen v b'],

['nguyen v c', 'nguyen v d']

];

console.log(allClasses[0][0]); //print 'nguyen v a'

console.log(allClasses[1][1]); //print 'nguyen v d'

* 1. Biến và Hằng
     1. Biến

Tên biến và tên hằng chỉ được bắt đầu bằng dấu “\_” và chữ và không được bắt đầu bằng số, không được chứa các kí tự đặc biệt.

Trong NodeJS ta có 2 cách để khai báo biến và giá trị. một là dùng từ khóa Var để khai báo, hai là không cần dùng từ khóa.

var a = "helloworld";

// khai bao bien a su dung tu khoa var

b = "xinchaothegioi";

console.log("bien a=" + a);

console.log("bien b=" + b);

kết quả là:

bien a= helloworld

bien b=xinchaothegioi.

* + 1. Hằng

Hằng trong NodeJs sẽ được khai báo cùng từ khóa const.

Trong nodejs thì khi khai báo hằng và biến thì hoàn toàn sẽ không cần phải khai báo kèm theo kiểu dữ liệu của chúng mà ngôn ngữ nó tự nhận diện kiểu cho chúng ta.

Ví dụ cụ thể:

const x = "welcome ";

khai báo hằng x cùng kiểu dữ liệu là string

const y = 10000;

khai báo hằng y cùng kiểu dữ liệu là số nguyên(int)

const z = 10.6;

khai báo hằng z cùng kiểu dữ liệu số thực(float)

const d = true;

khai báo hằng d cùng kiểu dữ liệu logic(boolean)

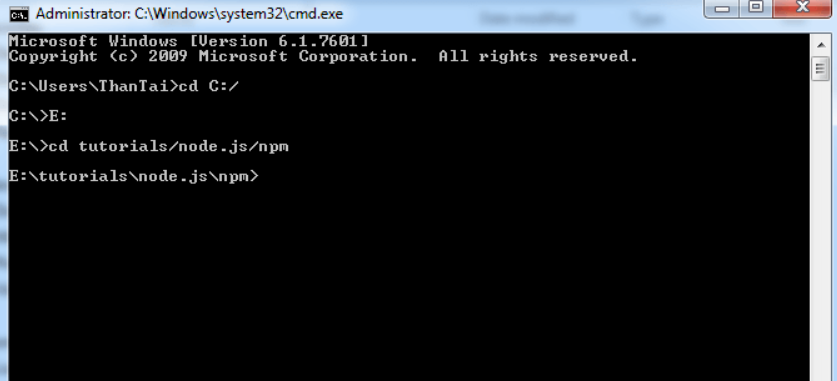
* 1. Các đặc tính của NodeJS
* Không đồng bộ: Tất cả các API của NodeJS đều không đồng bộ (none-blocking), NodeJs chủ yếu dựa vào nền của NodeJS Server và chờ Server trả dữ liệu về. Việc di chuyển máy chủ đến các API tiếp theo sau khi gọi và cơ chế thông báo các sự kiện của NodeJS giúp máy chủ để có được một phản ứng từ các cuộc gọi API trước
* Chạy rất nhanh: NodeJS được xây dựng và hình thành dựa vào nền tảng V8 Javascript Engine, cho nên việc thực thi chương trình rất nhanh.
* Đơn luồng nhưng khả năng mở rộng cao: NodeJS 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 có khả năng mở rộng như trái ngược với hầu hết 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 đến số lượng lớn hơn nhiều so với yêu cầu máy chủ truyền thống như Apache HTTP Server.
* Không đệm: NodeJS không đệm bất kì kiểu 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.
  1. NPM

Npm là một chương trình quản lý thư viện, source của NodeJs được tích hợp sẵn trong NodeJs. khi cài đặt node.js thì cũng đồng nghĩa với việc đã cài npm. Npm giống như composer trong PHP, NPM sử dụng command line để quản lý (download, update, delete) các package cho node.js.

* Để khởi tạo một dự án bằng nodeJS với npm chúng ta sử dụng lệnh

npm init

* Đầu tiên đến thư mục cần khởi tạo dự án.



Hình 2.2: Ảnh minh họa khởi tạo một dự án.

* Sau đó sử dụng lệnh (npm init) để bắt đầu khởi tạo project



Hình 2.3: Ảnh minh họa cách khởi tạo Package.

* Sau khi chạy xong các bạn sẽ thấy một file package.json vừa được sinh ra. File này sẽ lưu trữ hầu hết các thông tin của dự án.

Xây dựng package là một điều quan trọng trong NodeJS mà chúng ta không thể bỏ qua. Điều này cũng tương tự như Ruby-Gems: đây là tập hợp hàm có sẵn và có thể sử dụng được, thành phần tái sử dụng, và tập hợp các cài đặt dễ dàng thông qua kho lưu trữ trực tuyến.

Một số các Modul npm phổ biến:

* ExpressJS là Sinatra-inspired web framework rất phát triển của NodeJS, chứa hầu hết các ứng dụng chuẩn của NodeJS hiện nay.
* Connect: là phần mở rộng của HTTP server framework cho Node.js, cung cấp “plugins” được xem là trung gian, phục vụ như một nền tảng cơ sở cho Express.
* Socket.io và sockJS được xem là hai thành phần Server-side websockets components nổi tiếng nhất cho đến bây giờ.
* Jade: Một trong những engines mẫu, có cảm hứng từ HAML, còn là một phần mặc định trong Express.js
* mongo and mongojs: MongoDB hàm bao để cung cấp các API cho cơ sở dữ liệu đối tượng trong MongoDB NodeJS
* redis: là thư viện Redis client.
* Underscore: Thư viện tiện ích phổ biến nhất trong JavaScript, package được sử dụng với NodeJS.
  1. Toán tử trong NodeJs
     1. Toán tử gán

Toán tử gán là lấy giá trị của toán hạng bên phải gán nó vào toán hạng bên trái.

Ví dụ:

var name="HelloWorld";

* + 1. Toán tử số học

Toán tử số học là 1 dạng phép tính giản đơn cộng, trừ, nhân, chia trong số học. Ngoài ra còn có phép chia lấy dư (%) sử dụng để lấy ra đơn vị dư của 1 phép toán.

|  |  |  |  |
| --- | --- | --- | --- |
| Toán Tử | Lý giải | Ví dụ | Kết quả |
| + | Cộng hai số hạng | 26 + 7 | 33 |
| - | Trừ hai số hạng | 26 - 7 | 19 |
| \* | Nhân hai số hạng | 26 \* 7 | 182 |
| / | Chia hai số hạng | 26 / 7 | 3 |
| % | Trả về số dư | 26 % 7 | 5 |

Bảng 2.1: Bảng toán tử số học.

* + 1. Toán tử so sánh

Là toán tử được sử dụng nhằm thực hiện các phép toán so sánh giữa hai số hạng

|  |  |  |  |
| --- | --- | --- | --- |
| Phép toán | Tên | Giải Thích | Ví dụ |
| == | Bằng | Hai số hàng bằng nhau | A==5 |
| != | Không bằng | Hai số hạng không bằng nhau | A!=5 |
| === | Đồng nhất | Hai số bằng nhau và cùng kiểu | A===5 |
| > | Lớn hơn | Vế trái lớn hơn vế phải | A>5 |
| >= | Lớn hơn hoặc bằng | Vế trái lớn hơn hoặc bằng vế phải | A>=5 |
| < | Nhỏ hơn | Vế trái nhỏ hơn vế phải | A<5 |
| <= | Nhỏ hơn hoặc bằng | Vế trái nhỏ hơn hoặc bằng vế phải | A<=5 |

Bảng 2.2: Bảng toán tử so sánh.

* + 1. Toán tử logic

Toán tử logic chính là tổ hợp các giá trị boolean.

Ví dụ: bài toán tử dấu && trả giá trị true nếu toán tử ở vế trái và toán tử ở vế phải là true.

TRUE && TRUE

//kết quả là true

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Toán tử | Tên | Trả về True nếu | Ví dụ | Kết quả |
| || | Or | Vế trái hoặc vế phải là True | True||False | True |
| Or | Or | Vế trái hoặc vế phải là True | True||False | True |
| Xor | Xor | Vế trái hoặc vế phải là True nhưng không phải cả hai. | True||False | False |
| && | And | Vế trái và vế phải là True | True && False | False |
| And | And | Vế trái và vế phải là True | True && False | False |
| ! | Not | Không phải true | !True | False |

### Bảng 2.3: bảng toán tử logic

### Toán tử kết hợp

* Tăng

var a=10;

var a++;

trả về a=11

* Giảm

var a=10;

var a--;

trả về a=9

* Cộng

var a=10;

var b=15;

a+=b;

trả về a=25

* Trừ

var a=20;

var b=15;

a-=b;

trả về a=5

* Nhân

var a=10;

var b=15;

a\*=b;

trả về a=150

* Chia

var a=30;

var b=15;

a/=b;

trả về a=2

* 1. Hàm trong NodeJS

Dùng từ khóa function để khai báo hàm trong NodeJs

Ví dụ:

function hello()

{

consoloe.log('hello world');

}

ở đây chúng ta đã khai báo một hàm có tên là hello và để sử dụng chúng ta cần phải gọi lại tên hàm là hello kèm với cặp dấu ().

Ví dụ:

hello();

Sử dụng hàm có tham số:

Ví dụ:

function hello(name)

{

console.log("chao " + name);

}

khởi tạo hàm

hello('tuan');

gọi hàm hello với tham số truyền vào là tuan

hello('nam');

gọi hàm hello với tham số truyền vào là nam

Kết quả:

chao tuan

chao nam

* 1. Câu lệnh rẽ nhánh
     1. Câu Lệnh If-ELSE

If-else là điều kiện được sử dụng khá nhiều trong hầu hết các ngôn ngữ lập trình.

Ví dụ:

var diem = 7.5;

if (diem < 5) {

console.log("yeu");

} else

if (diem >= 5 && diem < 7) {

console.log("TB");

} else

if (diem >= 7 && diem < 8) {

console.log("kha");

} else if(diem >8 && diem<=9)

{

Console.log(“gioi”)

}

{

console.log("XuatXac");

}

Kết quả: kha

* + 1. Câu lệnh lồng nhau

Ví dụ:

var diem = 6.5;

if (diem < 5) {

console.log("yeu");

} else

if (diem >= 5 && diem < 7) {

console.log("TB");

} else

if (diem >= 7 && diem < 8) {

console.log("kha");

} else if(diem >8 && diem<=9)

{

Console.log(“gioi”)

}

{

console.log("XuatXac");

}

Kết quả: TB

* + 1. Switch-Case

Cấu trúc của Switch-Case như sau:

Switch (bien)

{

Case giá trị 1:

//hanh dong;

Break;

Case giá trị 2:

//hanh dong;

Break;

Case giá trị N:

//hanh dong;

Break;

Default:

//hanh dong;

Break;

}

Ví dụ:

var now = 5;

switch (now)

{

Case1:

case1:

break;

case 2:

console.log("Thứ 2");

break;

case 3:

console.log("Thứ 3");

break;

case 4:

console.log("Thứ 4");

break;

case 5:

console.log("Thứ 5");

break;

case 6:

console.log("Thứ 6");

break;

case 7:

console.log("Thứ 7");

break;

case 8:

console.log("Chủ nhật");

break;

default:

console.log("Sai");

break;

}

Kết quả: Thứ 5

* 1. Đối tượng trong NodeJs (object)

Trong nodejs đối tượng là một khái niệm trừu tượng thể hiện cho một đối tượng cụ thể, và có thể tự tạo một đối tượng theo ý của mình dựa vào yêu cầu của ứng dụng.

Ví dụ:

Chúng ta tạo ra một đối tượng chuyên xử lý vấn đề về sinh viên thì ta sẽ đặt tên cho nó là Student.

* + 1. Sử dụng từ khóa new Object (lưu ý chữ “O” trong Object phải ghi hoa)

Ví dụ: var Students = new Object();

* Ngoài ra ta cũng có thể dùng từ khóa {}.

Ví dụ: var Students = {};

* + 1. Thuộc tính và phương thức trong đối tượng
    2. Thuộc tính
* Thuộc tính là tất cả những đặc điểm (có thể hiểulà biến) cần lưu trữ trong đối tượng.

Ví dụ: với đối tượng Students ta cần có các thuộc tính sau:

Name: tên

Age: tuổi

Class: lớp

Pointavg: điểm TB

ví dụ về khai báo:

cách 1: khai báo new Object().

var Students = new Object();

Students.name = 'Nguyễn Cao Tấn';

Students.age = 18;

Students.class = 'Sinh viên';

Students.pointAvg = 7.0;

Cách 2: khai báo nhóm

var Students = {name: 'Nguyễn Cao Tấn', age: 18, class: 'Sinh viên', pointAvg: 7.0};

Cách 3: thêm thuộc tính vào sau.

var Students = {};

Students.name = 'Nguyễn Cao Tấn;

Students.age = 18;

Students.class = 'Sinh viên';

Students.pointAvg = 7.0;

* + 1. Phương thức

Phương thức trong NodeJs cũng có các tính chất như các ngôn ngữ khác, có thể hiểu và gọi là hàm.

Ví dụ: với Object Students ở phần thuộc tính ta sẽ tạo thêm phương thức addstudents() bằng 3 cách sau:

Cách 1: Khởi tạo xong khai báo với new Object().

var Students = new Object();

Students.AddStudents = function (ten, tuoi, lop, DiemTb) {

this.name = ten;

this.age = tuoi;

this.class = lop;

this.pointAvg = Diemtb;

}

Cách 2: Khai báo ngay trong khi khởi tạo.

var Students = {

AddStudents: function (ten, tuoi, lop, DiemTb)

{

this.name = ten;

this.age = tuoi;

this.class = lop;

this.pointAvg = Diemtb;

}

};

Cách 3: Khởi tạo xong rồi khai báo ({}).

var Students = {};

Students.AddStudents = function ((ten, tuoi, lop, DiemTb) {

this.name = ten;

this.age = tuoi;

this.class = lop;

this.pointAvg = Diemtb;

}

* + 1. Gọi thuộc tính và phương thức
       1. gọi thuộc tính

Cú pháp:

doituong.thuoctinh;

hoặc

doituong['thuoctinh'];

Ví dụ:

Students.name;

Hoặc

Students['name'];

* + - 1. gọi phương thức

Cú pháp:

doituong.phuongthuc;

ví dụ:

Students.AddStudents('Tấn', 20, 'ABC', 2.5);

* + 1. Đối tượng trong đối tượng

Cú pháp:

doituong = {

doituong1: {

doituong2: {

//code

}

}

}

Ví dụ:

var Person = {

name: {

firstName: "Cao",

lastName: "Tấn"

},

birthday: {

day: 26,

month: 7,

year: 1999

}

}

* + 1. Mảng trong đối tượng

Cú pháp:

arr = [doituong1, doituong2];

hoặc

arr = [];

arr[0] = doituong;

arr[1] = doituong1;

arr[..] = doituong...;

Ví dụ:

var Person = {

name: {

firstName: "Cao",

lastName: "Tấn"

},

birthday: {

day: 26,

month: 07,

year: 1999

}

};

var arr = [Person];

console.log(arr[0].name.firstName);

* 1. Khởi tạo Server trong NodeJs với HTTP module
     1. Giới thiệu về HTTP module

HTTP là một module được tích hợp sẵn vào trong NodeJs, module này có nhiệm vụ khởi tạo một cổng kết nối HTTP server trả về client.

* + 1. Khởi tạo Server với HTTP module

Để sử dụng được bất kỳ module nào trong NodeJs thì chúng ta cần phải require module đó. Và để có thể require một module trong ngôn ngữ NodeJs. sử dụng từ khóa “require”.

Ví dụ: khởi tạo server chạy port 3000

var http = require('http');

http.createServer().listen(3000);

trong phương thức createServer sẽ chứa một phương thức ẩn danh có 2 tham số truyền vào.

Tham số thứ nhất: là biến chứa đựng các thông số có liên quan đến request mà người dùng (client) gửi lên.

Tham số thứ hai là biến chứa đựng các thông số có liên quan đến response mà máy chủ Server muốn trả về client.

Ví dụ:

tạo một tệp tin index.js trong thư mục node-tutorial

(khai báo sử dụng module HTTP)

var http = require('http');

(Khởi tạo server chạy cổng 3000)

http.createServer(function (request, response) {

response.write('hello world! toidicode.com');

response.end();

}).listen(3000);

Sau đó run file index.js và mở trình duyệt truy cập link sau:

<http://localhost:3000>

* + 1. Các phương thức và thuộc tính được dùng trong resquest và response
       1. Writehead

Hàm writehead() là hàm thiết lập kiểu dữ liệu mà máy chủ server muốn trả về.

Ví dụ: thiết lập server trả về là một trang HTML

(khai báo sử dụng module HTTP)

var http = require('http');

(Khởi tạo server chạy cổng 3000)

http.createServer(function (request, response) {

response.writeHead(200, {“Content-Type': 'text/html”});

response.write('Hello world');

response.end();

}).listen(3000);

* + - 1. write

Hàm write() thiết lập nội dung mà server muốn trả về trình duyệt, nội dung này có thể là text có thể là HTML code.

Ví dụ: Trả về một trang HTML có thẻ H1 chứa dòng chữ hello world.

(khai báo module HTTP)

var http = require('http');

(Khởi tạo server chạy cổng 3000)

http.createServer(function (request, response) {

response.writeHead(200, {'Content-Type': 'text/html'});

response.write('<html>');

response.write('<head>');

response.write('<title>Hello world </title>');

response.write('</head>');

response.write('<body> <h1>Hello world<h1> </body>');

response.write('</html>');

response.end();

}).listen(3000);

* + - 1. URL

URL chứa các paramter trong URL mà client gửi lên server.

Ví dụ:

(khai báo sử dụng module HTTP)

var http = require('http');

(Khởi tạo server chạy cổng 3000)

http.createServer(function(request, response)

{

var params = request.url;

response.write(param);

response.end();

}).listen(3000);

Nếu các tên miền hiện ra phía sau domain thì sẽ là url

Ví dụ: <http://localhost:3000/WebDoAn> thì url sẽ là WebDoAn

* 1. Đọc ghi file bằng module fs trong Node.js
     1. Giới thiệu về module fs

Module fs (viết tắt của file system) là module được tích hợp sẵn trong NodeJs có chức năng xử lý file, thư mục trong NodeJs

Khi sử dụng module chúng ta phải require() nó. Để require module fs sử dụng cú pháp:

Require(‘fs’);

Để xem phương thức trong module ta dùng lệnh console.log().

Ví dụ:

var fs = require(‘fs’);

console.log(fs);

* + 1. Đọc files với module fs

Để đọc file ta sử dụng phương thức readFile() cú pháp là:

readFile(‘pathFile’, [option], function(err, data) {

//

});

pathFile: đường dẫn File cần đọc

option: có thể bỏ trống, có thể chứa các chuỗi hoặc mảng

err: biến lỗi

data: dữ liệu đọc được từ file.

Các bước thực hiện:

Bước đầu tiên sẽ tạo ra một file code.html

Bước 2: tạo một file server.js cùng cấp với file code.html để khởi tạo server và xử lý

Bước 3: chạy file server là nhận lại kết quả.

* + 1. Ghi file

Dùng phương thức WriteFile() để ghi nội dung vào một file.

Cú pháp:

wireFile(‘filespath’, 'noidung' , options, callback);

trong đó:

filepath: đường dẫn file cần ghi.

content: nội dung muốn ghi vào file.

option: chuỗi hoặc mảng, và có thể để trống.

callback: xử lý sau khi ghi file.

Ví dụ: ghi nội dung vào file write.html

var fs = require(‘fs’);

var content = ‘ghi vào file writer.html’;

fs.writeFile('writer.html', content, ‘utf8’, function (err) {

(Kiểm tra)

if (err)

throw err;

else

console.log('Ghi file thanh cong!');

});

* 1. Module URL

Module url dùng để xử lý và phân tích chuỗi URL qua đó biết được thông số URL

* + 1. Khai báo sử dụng Module URL

Cú pháp:

require('url');

các thuộc tính của module giống như đối tượng location trong javascript.

Ví dụ:

var url = require('url');

var website = "http://webgiasu?a=5";

var parse = url.parse(website, true);

console.log('auth: ' + parse.auth);

console.log('hash: ' + parse.hash);

console.log('host: ' + parse.host);

console.log('hostname: ' + parse.hostname);

console.log('href: ' + parse.href);

console.log('path: ' + parse.path);

console.log('pathname: ' + parse.pathname);

console.log('port: ' + parse.port);

console.log('protocol: ' + parse.protocol);

console.log('query: ' + parse.query.a);

console.log('search: ' + parse.search);

console.log('slashes: ' + parse.slashes);

* 1. Upload file trong NodeJs
     1. Tổng quan

Chức năng này được thực hiện khi tích hợp các module sau

* Module http
* Module fs
* Module formidable (module phân tích dữ liệu từ form gửi lên)
  + 1. cài đặt module formidable

cú pháp:

npm installl -s formidable

* + 1. code upload
* viết code khởi tạo server (chạy port 3000)
* tạo file chứa code html form (đặt ở file index.html cùng cấp với server.js)
* dùng module fs để render file view đó ra trên server.
* chạy server và nhận kết quả.
* trên file server.js chúng ta sẽ tiếp tục require module formidable
* xuất kết quả khi chạy server.

Ví dụ:

var https = require('http');

var fs = require('fs');

var formidables = require('formidable');

http.createServer(function (req, res) {

if (req.url === '/upload' && req.method.toLowerCase() === 'post') {

var form = new formidables.IncomingForm();

form.uploadDir = "uploads/";

form.parse(req, function (err, fields, file) {

var path = file.files.path;

var newpath = form.uploadDir + file.files.name;

fs.rename(paths, newpaths, function (err) {

if (err) throw err;

res.end('Upload Thanh cong!');

});

});

return;

}

res.writeHead('200', {'Content-Type': 'text/html'});

fs.readFile('index.html', 'utf8', function (err, data) {

if (err) throw err;

res.end(data);

})

}).listen(3000);

* 1. Sự kiện trong NodeJs
     1. Event loop

NodeJs được thực thi nhanh hơn rất nhiều so với ngôn ngữ khác là nhờ có rất nhiều sự kiện ở trong NodeJs, đó là các yếu tố giúp cho Nodejs cải thiện được tốc độ mà ngôn ngữ khác không có được.

- Thêm sự kiện: để sự dụng được sự kiện trong NodeJs bắt buộc phải requite module event và khởi tạo eventEimitter

Cú pháp:

Require module event

var event = require('events');

var eventEmitter = new event.EventEmitter();

và muốn thêm sự kiện dùng phương thức on theo cú pháp sau:

on(eventName, handleEvent);

trong đó:

eventName: tên sự kiện muốn thêm

handleEvent: hàm xử lý khi eventName được gọi

* Thực thi sự kiện: dùng phương thức emit

Cú pháp:

emit(evenName[,...args]);

trong đó:

eventNam: sự kiện muốn thực thi

args: là tham số truyền vào dữ liệu.

Ví dụ:

var event = require('events');

var eventEmitter = new event.EventEmitter();

var connectionHandle = function (data) {

console.log('connection successful' + data);

}

eventEmitter.on('connection', connectionHandle);

thực thi sự kiện

eventEmitter.emit('connection', new Date());

* 1. Tạo client request trong NodeJs
     1. Cài đặt

module request là một module phát triển dựa trên module http của NodeJs

-cú pháp:

npm install request

* require vào dự án sau khi cài đặt xong.

let requests = require('requests');

gửi request trong module request sử dụng cú pháp sau:

request(url, function (err, res, body));

-khi đó:

request: biến ánh xạ module request.

err: chứa lỗi nếu có.

res: chưa thông tin server trả về.

body: chưa body mà server trả về.

* + 1. các phương thức mà module request hỗ trợ:

|  |  |
| --- | --- |
| Method | Mô tả |
| Request.get() | Gửi request với phương thức GET. |
| Request.post() | Gửi request với phương thức POST. |
| Request.put() | Gửi request với phương thức PUT . |
| Request.patch() | Gửi request với phương thức PATCH. |
| Request.del() | Gửi request với phương thức DEL. |
| Request.head() | Gửi request với phương thức HEAD. |
| Request.options() | Gửi request với phương thức OPTIONS. |

Bảng 2.4: bảng phương thức module request.

Ví dụ:

let requests = require('requests');

request.post('https://webgiasu', function (err, res, body) {

nếu có lỗi

if (err)

throw err;

in ra header

console.log(res);

in ra body nhận được

console.log(body);

})

# CƠ SỞ DỮ LIỆU MONGODB

* 1. Giới thiệu về MongoDB

Mongodb là một chương trình cơ sở dữ liệu có mã nguồn mở được thiết kế theo hướng đối tượng, mongodb được viết bằng C++ trong đó các bảng được cấu trúc một cách cực kì linh hoạt, cho phép dữ liệu lưu trên bảng không cần phải tuân theo một quy định cấu trúc nào. Chính vì vậy mà mongodb được sử dụng để lưu trữ các dữ liệu có cấu trúc phức tạp, đa dạng và không cố định.

Ngoài ra mongodb còn là cơ sở dữ liệu đa nền tảng, có hiệu suất cao, tính khả dụng cao và mở rộng rất dễ dàng.

* + 1. Lợi thế của Mongodb
* Cấu trúc đối tượng rõ ràng.
* Không có Jojn phức tạp.
* Khả năng truy vấn sâu, mongodb hỗ trợ truy vấn động trên các Document bởi sử dụng ngôn ngữ truy vấn, nhưng rất mạnh mẽ như SQL.
* Ít schema: Collection giữ các Document khác nhau, dữ liệu không đồng nhất, không bị ràng buộc.
* Mongodb dễ dàng mở rộng hơn.
* Giúp truy cập dữ liệu nhanh hơn, sử dụng bộ nhớ nội tại để lưu giữ công việc.
  + 1. đặc điểm của Mongodb
* định hướng của Document: dữ liệu được lưu trong các tài liệu JSON.
* Có thể lập chỉ mục trên bất kì thuộc tính nào.
* Truy vấn của mongodb đa dạng hơn
* Tốc độ cập nhật nhanh hơn.
  1. Mục đích sử dụng cơ sở dữ liệu Mongodb

Cơ sở dữ liệu mongodb gồm các chức nặng như: lưu trữ (storage), truy cập(accessibility), phân tán dữ liệu(distribution)

* Lưu trữ: Mongodb lưu trữ dữ liệu định hướng document một cách linh hoạt, được định dạng JSON, điều này có nghĩa các trường có thể thay đổi từ document này sang document khác và cấu trúc dữ liệu cũng được thay đổi theo thời gian, cập nhật nhanh chóng, document model ảnh xạ đến các object trong code giúp dễ dàng thao tác với dữ liệu.
* Truy cập: truy vấn đặc biệt, đa dạng, lập chỉ mục và tổng hợp theo thời gian thực, còn cung cấp các phương thức mạnh mẽ giúp phân tích dữ liệu và truy cập.
* Phân tán dữ liệu: mongodb là một cơ sở dữ liệu phân tán, tính sẵn sàng cao, kèm theo là nhân rộng tích hợp theo chiều ngang và phân tán theo vị trí địa lý, rất dễ dàng xây dựng và sử dụng. tự dộng sharding, protect end-to-end và chuyển đổi dự phòng, được quản lý tự động hóa, giám sát và sao lưu và hoàn toàn miễn phí
  1. Thuật ngữ trong Mongodb

|  |  |
| --- | --- |
| Thuật Ngữ | Diễn giải |
| \_id | Trường bắt buộc phải có trong document, đại diện cho giá trị duy nhất, được hiểu khóa chính trong document. |
| Collection | Nhóm của nhiều document trong mongodb, là một bảng tương ứng trong csdl relational database management system, nằm trong một csdl duy nhất, không định nghĩa các cột, hàng hoặc kiểu dữ liệu trước |
| Cursor | là một con trỏ đến tập kết quả của một truy vấn, máy khách có thể lặp qua con trỏ để lấy kết quả. |
| Database | Chưa các collection, chứa các bảng, có tập tin riêng lưu trên bộ nhở vật lý, máy chủ có thể chứa nhiều database |
| Document | Là một bản ghi thuộc một collection, bao gồm trường tên và giá trị. |
| Field | Là cặp name-value trong document, một document có thể có hoặc không có trường. các trường giống các cột csdl quan hệ. |
| JSON | Viết tắt của JavaScript Object Notation, dữ liệu có cấu trúc và hỗ trợ rất nhiều ngôn ngữ lập trình. |
| Index | Cấu trúc dữ liệu đặc biệt, quét một phần nhỏ các tập dữ liệu dễ dàng, hỗ trợ phân tích hiệu quả các truy vấn, nếu không có chỉ mục, mongodb sẽ quét tất cả các documents của collection để tìm document phù hợp với câu truy vấn. |

Bảng 3.1: bảng thuật ngữ trong Mongodb

* 1. Kiểu dữ liệu trong Mongodb

|  |  |  |
| --- | --- | --- |
| Kiểu dữ liệu | Number | Alias |
| Double | 1 | “double” |
| String | 2 | “string” |
| Object | 3 | “object” |
| Array | 4 | “array” |
| Binary data | 5 | “binData” |
| Underfined | 6 | “underfined” |
| Objectld | 7 | “objectld” |
| Boolean | 8 | “bool” |
| Date | 9 | “date” |
| Null | 10 | “null” |
| Regular expression | 11 | “regex” |
| Dbpointer | 12 | “dbpointer” |
| Javascript | 13 | “javascript” |
| Symbol | 14 | “symbol” |
| JavaScript(with scope) | 15 | “javascriptWithScope” |
| 32-bit integer | 16 | “int” |
| Timestamp | 17 | “timestamp” |
| 64-bit integer | 18 | “long” |
| Decimal 128 | 19 | “decimal” |
| Min key | -1 | “minkey” |
| Max key | 127 | “maxkey” |

Bảng 3.2: bảng kiểu dữ liệu trong MongoDB

* Chuỗi (string): Đây là kiểu dữ liệu được sử dụng phổ biến nhất để lưu giữ dữ liệu. Chuỗi trong MongoDB phải được định dạng là UTF-8 hợp lệ.
* Số nguyên (integer): Kiểu dữ liệu này được sử dụng để lưu một giá trị số. Số nguyên được quy định là 32 bit hoặc 64 bit phụ thuộc vào máy chủ Server.
* Boolean: Kiểu dữ liệu này được dùng với mục đích lưu giữ một giá trị Boolean (true/false).
* Double: Kiểu dữ liệu này được dùng với mục đích lưu các giá trị số thực dấu chấm động.
* Min/ Max keys: Kiểu dữ liệu này được dùng với mục đích so sánh một giá trị với các phần tử BSON thấp nhất và cao nhất.
* Mảng (array): Kiểu dữ liệu này được sử dụng để lưu giữ các mảng hoặc danh sách hoặc nhiều giá trị vào trong một key.
* Timestamp: Đánh dấu thời điểm nào đó mà một Document được sửa đổi hoặc được thêm vào.
* Object: Kiểu dữ liệu này được dùng với mục đích cho các Document được nhúng vào.
* Null: Kiểu dữ liệu này được dùng với mục đích lưu một giá trị Null.
* Symbol: Kiểu dữ liệu này được sử dụng giống như một chuỗi
* Date: Loạt dữ liệu này được dùng để lưu giữ date và time hiện tại trong định dạng UNIX time.
* Object ID: Kiểu dữ liệu này được dùng với mục đích lưu giữ ID của Document.
* Binary data: Kiểu dữ liệu này được dùng với mục đích lưu giữ dữ liệu nhị phân.
* Code: Kiểu dữ liệu này được dùng với mục đích lưu giữ JavaScrip code vào trong Document.
* Regular expression: Kiểu dữ liệu này được dùng với mục đích lưu giữ Regular Expresion.
  1. Một số câu lệnh truy vấn cơ bản của Mongodb

• CREATE TABLE (Tạo bảng):

db.createCollection('t\_name'); (t\_name: là tên bảng)

• ADD TABLE (chèn bảng):

db.t\_name.insert({c1\_ten: c1\_giatri, c2\_ten: c2\_type,…});

• CREATE DATABASE (Tạo Database):

use db\_name;

• DELETE DATABASE (Xóa Database):

db.dropDatabase();

• DELETE TABLE (Xóa Bảng):

db.t\_name.drop();

• UPDATE (Cập nhật):

db.t\_name.update({ \_id: id\_giatri }, { $set: { c\_name: c\_new\_giatri } });

• DELETE (Xóa):

db.t\_name.remove({\_id: id\_value});

• DELETE ALL (Xóa tất cả):

db.t\_name.deleteMany();

• FIND ALL (Tìm tất cả):

db.t\_name.find();

• FIND (tìm theo trường):

db.t\_name.find({c\_name: c\_value});

* 1. Truy vấn có điều kiện trong MongoDB

Có thể thêm điều kiện vào trong hàm find() với cú pháp:

db.collectionName.find(condition)

Liệt kê cú pháp của mệnh đề điều kiện:

|  |  |
| --- | --- |
| Phép Toán | Cú pháp |
| Bằng (equality) | {key: value} |
| Nhỏ hơn (less than) | {key: {$lt: value}} |
| Nhỏ hơn bằng (less than equals) | {key: {$lte: value}} |
| Lớn hơn (greater than) | {key: {$gt: value}} |
| Lớn hơn bằng (greater than equals) | {key: {$gte: value}} |
| Khác (not equals) | {key: {$ne: value}} |
| Thuộc (in) | {key: {$in: [gtri1, gtri2,..]}} |
| Không thuộc vào(Not in) | {key: {$nin: [gtri1, gtri2,..]}} |

Bảng 3.3: bảng cú pháp truy vấn có điều kiện trong MongoDB

* 1. MongoDB Replication

Replication là dữ liệu được đồng bộ hóa trên nhiều server. Nó sẽ cung cấp rất nhiều tính năng và tăng tính sẵn sàng của data với nhiều bản sao dữ liệu trên nhiều server DB, replication bảo về DB khỏi việc mất dữ liệu trên một server đơn lẻ. replication sẽ cho phép bạn khôi phục dữ liệu từ các ỗ cững lỗi và việc service lỗi và hay gặp vấn đề. Các dữ liệu có thể sao chép rất nhiều và và có thể chỉnh một bản cho vấn đề khôi phục, reportho

Vậy tại sao lại cần REPLICATION:

* Điều đầu tiên là dữ liệu data rất an toàn.
* Có tính năng luôn luôn sẵn sàng cho dữ liệu.
* Có thể khôi phục dữ liệu khỏi lỗi.
* Không bị downtime khi xảy ra maintenance.
* Tăng khả năng đọc và xử lý dữ liệu được cải thiện.
  1. Cài đặt mongodb
* Để có thể cài đặt Mongodb ta sử dụng lệnh:

Npm install mongodb

* Để sự dụng module mongodb ta phải require nó vào file chúng ta sử dụng (chỉ sử dụng module MongoClient)

Cú pháp:

Var mongoClient = require(mongodb).MongoClient;

* 1. Kết nối NodeJs với mongodb

Muốn khởi tạo kết nối từ NodeJs đến mongodb chúng ta sẽ sử dụng phương thức connect trong đối tượng MongoClient

Cú pháp:

mongoClient.connect(url, function (error, dbs) {

});

Khi đó:

* url là đường dẫn kết nối đến mongodb
* error: biến chứa lỗi nếu có.
* db: là object chứ data kết nối.

ví dụ:

var mongoClients = require('mongodb').MongoClient;

mongoClients.connect('mongodb://127.0.0.1:3000', function (err, db) {

(neu ket noi khong thanh cong thi in ra loi)

if (err) throw err;

console.log('Ket noi thanh cong');

});

* 1. Đóng kết nối NodeJs đến Mongodb

để đóng kết nối bằng NodeJs ta dùng cú pháp sau:

db.close();

trong đó, db là object chưa data kết nối.

ví dụ:

var mongoClient = require('mongodb').MongoClient;

mongoClient.connect('mongodb://127.0.0.1:3000', function (err, db) {

(neu ket noi khong thanh cong thi in ra loi)

if (err) throw err;

console.log('Ket noi thanh cong');

db.close();

console.log('close thanh cong');

});

* 1. Sử dụng Mongodb

Các bước thực thi:

1. Hiện thị tất cả các databases

Cú pháp: show dbs

1. Tạo database

Sử dụng cú pháp: use 'database\_name'

1. Tạo một collection và insert dữ liệu vào collection

Cú pháp: db.collection\_name.insert({'key\_name': 'value'})

1. Get tất cả object của một collection

Cú pháp: db.collection\_name.find({})

1. Drop một collection

Cú pháp: db.collection\_name.drop()

# PHÂN TÍCH THIẾT KẾ HỆ THỐNG

## Tác nhân và use case hệ thống

### Các tác nhân (Actor)

* Khách hàng (Client)
* Người quản trị (Admin).
* Nhân viên (Staff)

## Các use case

### Đăng nhập:

* Mô tả: Người dùng đăng nhập vào hệ thống có thể thay đổi mật khẩu, cập nhật thông tin cá nhân, đăng ký tài khoản mới và các chức năng trong menu chính được người quản trị phân quyền. Các t

### Đăng ký:

* Mô tả: Người dùng đăng ký tài khoản để nhận lớp hoặc đăng thông tin tuyển dụng gia sư.

### Đăng tin tuyển dụng:

* Mô tả: Người dùng điền thông tin cần tuyển dụng (ngày, giờ dạy, học phí, môn,…). Sau khi đăng tin các gia sư sẽ tiến hành thương lượng và đề nghị nhận lớp. Sau khi nhà tuyển dụng chấp nhận sẽ trừ phí hoa hồng của gia sư (theo quy định) và thông tin tuyển dụng sẽ ẩn đi. Khi người tuyển dụng hoặc gia sư hủy lớp tùy theo quy định sẽ được hoàn tiền lại tài khoảng.

### Tạo hồ sơ gia sư:

* Mô tả: Người dùng muốn nhận lớp sẽ phải điền đầy đủ thông tin( địa chỉ, email, sdt, hình đại diện, số CMND,… ). Sau khi xác thực qua tin nhắn hoặc email sẽ được quyền nhận lớp.

### Nạp tiền vào tài khoản

* Mô tả: Người dùng có thể nạp tiền vào tài khoản trang web để sử dụng 1 số chức năng. Phía người tuyển dụng có thể sử dụng tiền nạp vào để đẩy trang web mình lên đầu trong 1 khoảng thời gian. Phía gia sư mỗi lần đề nghị nhận 1 lớp sẽ mất 1 khoảng phí nhỏ và sau khi nhà tuyển dụng đồng ý gia sư sẽ mất 1 khoảng tiền hoa hồng. Phí có thể xem phần quy định.

### Tra cứu thông tin

* Mô tả: Người dùng có thể tra cứu thông tin về việc làm, hoặc các gia sư để xem chất lượng giảng dạy.

### Hỗ trợ khách hàng

* Mô tả: Người dùng khi muốn hỗ trợ 1 vấn đề sẽ gửi mail về địa chỉ [GaSuSupport@gmail.com](mailto:GaSuSupport@gmail.com) để nhận được sự hỗ trợ từ nhân viên tư vấn và hỗ trợ. ( hủy lớp, hoàn tiền,…)

### Thay đổi quy định

* Mô tả: Người quản trị (admin) có thể thay đổi các quy định trên trang web (mức phí, số lớp có thể nhận,…).

### Lập báo cáo

* Mô tả: Người quản trị có thể xem và xuất báo cáo doanh thu, số lượng người dùng,…

## Use Case

## Active Diagram

## Sequence Diagram

## Class Diagram

## Mô Hình Database

# PHỤ LỤC

# TÀI LIỆU THAM KHẢO

Theo chuẩn IEEE