|  |  |  |
| --- | --- | --- |
| HỌC VIỆN NÔNG NGHIỆP VIỆT NAM  **KHOA CÔNG NGHỆ THÔNG TIN** |  | CỘNG HOÀ XÃ HỘI CHỦ NGHĨA VIỆT NAM  **Độc lập - Tự do - Hạnh phúc** |



**ĐỀ CƯƠNG KHÓA LUẬN TỐT NGHIỆP**

**ĐỀ TÀI:**

***“Tìm hiểu mô hình kiến trúc Microservice và xây dựng ứng dụng”***

**Giảng viên hướng dẫn chính: Ths. Trần Trung Hiếu**

**Họ và Tên sinh viên: Đào Văn Tâm**

**Mã sinh viên: 601329**

**Lớp: K60THA**

**Hà Nội 2020**

**Phần 1: Mở đầu**

* 1. . Đặt vấn đề
  2. . Mục đích và yêu cầu
     1. Mục đích
     2. Yêu cầu

**Phần 2: Tổng quan tình hình nghiên cứu trong và ngoài nước**

2.1. Tình hình nghiên cứu trong nước

2.2. Tình hình nghiên cứu ngoài nước

2.3. Nêu tên Đề tài và tính thời sự, tầm quan trọng của Đề tài

**Phần 3: Nội dung và phương pháp nghiên cứu**

3.1. Địa điểm và thời gian nghiên cứu (thực tập)

3.2. Nội dung nghiên cứu

3.3. Phương pháp nghiên cứu: Trình bày đầy đủ phương pháp nghiên cứu cần áp dụng để đạt được những nội dung chủ yếu của đề tài:

**Phần 4: Kế hoạch và dự kiến kết quả đạt được**

Nêu kế hoạch thực hiện đề tài theo thời gian là tuần và dự kiến kết quả chính sẽ đạt được

**Phần 5: Tài liệu tham khảo**

Chi tiết các tài liệu tham khảo (dự kiến) có liên quan trong quá trình xây dựng đề cương

**Phần 1: Mở đầu**

* 1. **Đặt vấn đề**

Trong thời đại công nghệ hiện nay, có thể thấy công nghệ thông tin đang giữ một vai trò quan trọng đối với sự phát triển của loài người nói chung và sự phát triển kinh tế, chính trị, xã hội của một đất nước nói riêng. Với những bước tiến như vũ bão ở những tập kỷ cuối thế kỷ XX, công nghệ thông tin đã tạo nên một diện mạo mới – kỉ nguyên công nghệ thông tin. Nghành công nghệ thông tin rất đa dạng được chia thành nhiều lĩnh vực, trong đó phát triển ứng dụng web là một trong những linh vực có vai trò khá quan trọng và được con người hướng tới nhiều nhất. Từ những website tĩnh đơn sơ đến các website sinh động đẹp mắt như hiện nay, công nghệ phát triển ứng dụng web đã có những bước tiến vượt bậc. Cho đến hiện tại, lĩnh vực này đã vô cùng quen thuộc trong đời sống con người và sẽ là nghành nghề thu hút một nguồn nhân lực vô cùng lớn trong tương lai.

Trong quá khứ cũng như đến thời điểm hiện tại có 2 loại mô hình phổ biến hiện nay mà các doanh nghiệp cũng như các công ty trong và ngoài nước đã và đang phát triển đó là Monolithic và Microservice. Nhiều năm trước cũng như thời điểm hiện tại dường như mô hình Monolithic vẫn được đang áp dụng, tiếp cận dễ dàng.



Mặc dù mô hình này có cấu trúc module hợp lý, nhưng cách triển khai ứng dụng lại tùy thuộc vào ngôn ngữ lập trình hay thư viện framework, cơ sở dữ liệu mà mình sử dụng. Các thành phần business logic (services), đối tượng cho từng nghiệp vụ (objects) và các sự kiện (events) đều được bọc trong lõi ứng dụng. Ứng dụng viết kiểu này cực kì phổ biến. Chúng hoàn toàn dễ phát triển, đặc biệt cùng với sự phát triển của IDE và những công cụ khác tập trung vào xây dựng những ứng dụng dựa trên template sẵn có. Tuy cách tiếp cận kiến trúc một khối này dễ dàng nhưng lại bộc lộ nhiều khuyết điểm về lâu dài. Khi ứng dụng có số lượng người dùng tăng lên, yêu cầu thêm các tính năng mới, dữ liệu tăng lên, yêu cầu giao tiếp với các hệ thống bên ngoài khác nhiều hơn. Điều đó dẫn đến ứng dụng sẽ lớn hơn và việc bảo trì cũng như mở rộng sẽ khó khăn hơn. Một vấn đề khác nữa đó là do các module đều chạy trong một quá trình và hợp nhất với nhau, một lỗi trong bất kỳ module nào xảy ra cũng có khả năng làm lỗi cả một hệ thống.

Vậy nên để khắc phục những nhược điểm tồn tại trên mô hình kiến trúc Monolithic thì mô hình kiến trúc Microservice đã ra đời. Kiến trúc Microservice là một cách tiếp cận về kiến trúc, trong đó một ứng dụng duy nhất được tạo bởi nhiều thành phần (hay các dịch vụ) nhỏ hơn có thể triển khai độc lập. Kiến trúc Microservice hiện đang ngày càng phổ biến, ít nhất là đối với các nhà điều hành, những người đứng đầu dự án cũng như các nhà phát triển. Chính vì vậy, các lập trình viên càng có thêm cơ hội để tìm hiểu một cách dễ dàng và nhanh chóng hơn.

* 1. **Mục đích và yêu cầu**
     1. **Mục đích**
* Nghiên cứu mô hình kiến trúc Microservice và các công cụ, thư viện hỗ trợ như: Java, Spring Boot, Zuul, Eureka, Ribbon
* Ứng dụng mô hình kiến trúc Microservice để xây dựng ứng dụng đăng tin thuê nhà
  + 1. **Yêu cầu**
* Biết cách sử dụng ngôn ngữ lập trình Java, Spring framework và các công cụ hỗ trợ để thiết kế ứng dụng
* Xây dựng thành công ứng dụng đăng tin thuê nhà
* Yêu cầu chức năng:

+) Quản trị viên: Quản lý các bài đăng, duyệt bài, quản lý các thành viên chủ nhà

+) Chủ nhà: Đăng bài viết, sửa bài viết, xóa bài viết

+) Khách hàng: Tìm kiếm, xem thông tin bài đăng

**Phần 2: Tổng quan tình hình nghiên cứu trong và ngoài nước**

**2.1 Tình hình nghiên cứu trong nước**

Ở Việt Nam, mặc dù công nghệ thông tin phát triển và muộn hơn so với nhiều nước trên thế giới nhưng việc nắm bắt công nghệ của Việt Nam lại không hề thua kém. Các công ty, doanh nghiệp Việt Nam đang học hỏi và áp dụng theo các công nghệ mới mà các nước phát triển áp dụng. Trong đó mô hình kiến trúc Microservice đang là một trong các mô hình được các doanh nghiệp áp dụng để triển khai hệ thống. Hướng tới những bước chuyển đổi công nghệ tại thị trường trong nước, nhiều doanh nghiệp hướng tới chia nhỏ hệ thống nguyên khối thành kiến trúc Microservice, nhằm cung cấp khả năng mở rộng lớn hơn và thời gian nhanh hơn, đảm bảo khả năng tái sử dụng cao. Do đó, kiến trúc Microservice là sự lựa chọn ưu việt cho các dự án phức tạp.

**2.2 Tình hình nghiên cứu ngoài nước**

Hiện nay trên thế giới có nhiều tập đoàn như Amazon, eBay, Netflix đã giải quyết vấn đề ứng dụng bằng kiến trúc Microservice (nhiều dịch vụ nhỏ). Ý tưởng là chia nhỏ ứng dụng lớn ra thành các dịch vụ nhỏ để kết nối với nhau.

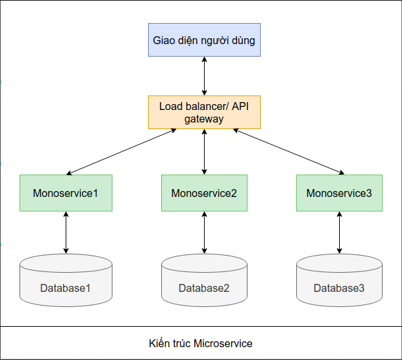
**2.3 Nêu tên đề tài, tính thời sự và tầm quan trọng của đề tài**

Microservices là kiến trúc dịch vụ siêu nhỏ “Microservice Architecture” phát triển nhanh chóng trong nhiều năm gần đây nhằm mô tả cách thiết kế phần mềm ứng dụng mà các dịch vụ có thể triển khai một cách độc lập. Mặc dù không có định nghĩa rõ ràng về kiểu kiến trúc này, ta vẫn có thể kể đến rất nhiều đặc tính chung của các tổ chức, phạm vi nghiệp vụ, tính chất quản lý phân tán cũng như dữ liệu khi nói đến microservices.

Nhiều tập đoàn như Amazon, eBay, Netflix đã giải quyết vấn đề ứng dụng một khối bằng kiến trúc Microservice (nhiều dịch vụ nhỏ). Mỗi dịch vụ nhỏ thực hiện một tập các chức năng chuyên biệt. Mỗi dịch vụ là một ứng dụng nhỏ có kiến trúc đa diện lõi là bussiness logic kết nối ra các adapter khác nhau. Khi vận hành, mỗi dịch vụ nhỏ được chạy trong một máy ảo (vitural machine) hoặc Docker container (ảo hóa tầng ứng dụng).

Khi áp dụng mô hình kiến trúc Microservice, mỗi vùng chức năng giờ được thực thi bởi một service nhỏ. Ứng dụng web cũng có thể chia nhỏ hơn chuyên cho từng đối tượng người dùng.

Mô hình kiến trúc Microservice ảnh hướng lớn đến quan hệ ứng dụng và cơ sở dữ liệu. Thay vì dùng chung một cơ sở dữ liệu giữa các service, mỗi service sẽ có cơ sở dữ liệu riêng.



Những đặc điểm của Microservice:

* Các service trong một hệ thống phần lớn được tách rời. Vì vậy, toàn bộ ứng dụng có thể dễ dàng được xây dựng, thay đổi và thu nhỏ.
* Các thành phần độc lập có thể dễ dàng thay đổi và nâng cấp.
* Mỗi service trong kiến trúc Microservice rất đơn giản và tập trung vào một nhiệm vụ chức năng duy nhất.
* Các lập trình viên hoặc những người mới tham gia vào một nhóm hay các nhóm có thể làm việc đọc lập với nhau trong quá trình phát triển.
* Mỗi service quản lý một cơ sở dữ liệu riêng, nằm trên một server riêng, tách biệt hoàn toàn với nhau.

Ưu điểm của Microservice:

* Giảm thiểu sự gia tăng phức tạp khi gặp hệ thống lớn.
* Dễ nâng cấp, mở rộng.
* Do tách biệt nên nếu một service bị lỗi, toàn bộ hệ thống vẫn hoạt động bình thường
* Các ngôn ngữ và công nghệ khác nhau có thể sử dụng để xây dựng các service khác nhau của cùng một ứng dụng.

Hạn chế của Microservice:

* Các module giao tiếp qua mạng có thể tốc độ không cao bằng mô hình kiến trúc một khối (Monolithic). Ngoài ra, mỗi module phải tự giải quyết các vấn đề về bảo mật, transaction, lỗi kết nối,…
* Mỗi service sử dụng database riêng, việc đảm bảo tính đồng nhất trong dữ liệu trở nên phức tạp.
* Sử dụng nhiều service nên việc theo dõi, quản lý các service phức tạp hơn.
* Cần đội ngũ có khả năng và trình độ để phân tách module, setup workflow.

Hiện nay hình thức cho thuê nhà vẫn đang phổ biến trong đời sống nhằm đáp ứng nhu cầu cho người dân. Thế nhưng quy trình để tìm và thuê nhà mà thỏa mãn về nhu cầu cũng như đáp ứng tiêu chí của người thuê sẽ gặp một số khó khăn nhất định như:

* Không biết được giá cả, chi phí thuê như nào, cao hay thấp và giá đó so với mặt bằng chung chênh ra sao.
* Không có ảnh chi tiết để người thuê biết rõ thông tin hơn.
* Khó khăn trong việc tìm những nhà có địa điểm thuận tiện cho việc đi lại.
* Khó khăn trong việc cần thông tin liên hệ với chủ.

Qua những bất cập trên em đã quyết định thực hiện đề tài “Xây dựng ứng dụng đăng tin thuê nhà”. Nhằm giúp mọi người hay bất kỳ ai có thể nắm bắt thông tin rõ hơn, khách hàng có thể truy cập ứng dụng để xem thông tin, giá cả phù hợp với nhu cầu. Dưới sự hướng dẫn của Ths. Trần Trung Hiếu, em mong muốn sẽ làm ra một ứng dụng trực tuyến để tối ưu nhu cầu cho khách hàng.

**Phần 3: Nội dung và phương pháp nghiên cứu**

**3.1 Địa điểm và thời gian nghiên cứu**

- Đề tài được thực hiện tại bộ môn Công nghệ phần mềm, trực thuộc Khoa Công nghệ thông tin

- Địa điểm thực tập tại: Công ty TNHH Rabiloo

- Thời gian thực hiện: Từ ngày 21/09/2020 đến ngày 29/11/2020

**3.2 Nội dung nghiên cứu**

Nghiên cứu lý thuyết và các công cụ, công nghệ và thư viện hỗ trợ như: HTML, CSS, Javascript, Bootstrap, Java, Spring Boot, Zuul, Eureka, Ribbon, MySQL Server, SQL Server

**3.2.1 HTML**

a) HTML là gì ?

- HTML là chữ viết tắt của cụm từ HyperText Markup Language, có nghĩa là “ngôn ngữ đánh dấu siêu văn bản”.

- HTML dùng để mô tả cấu trúc, các thuộc tính liên quan đến cách hiển thị của 1 đoạn văn bản nào đấy ra trình duyệt.

- Các phần tử HTML là các khối xây dựng nên trang web.

- Các phần tử HTML được biểu diễn bằng thẻ.

- Trình duyệt không hiển thị các thẻ HTML, nhưng sử dụng chúng để hiển thị nội dung của trang.

b) Bố cục cơ bản của HTML trên một trang web

- Thông thường, bố cụ của một trang web được chia làm 2 phần chính trong cặp thẻ <html></html>

<!DOCTYPE html>

<html>

<head>

<title>Page Title</title>

</head>

<body>

<h1>This is a Heading</h1>

<p>This is a paragraph.</p>

</body>

</html>

Trong đó:

<!DOCTYPE html>: Dùng để xác định tài liệu này thuộc dạng html cho trình duyệt biết.

<html> </html>: Là phần tử gốc, nằm ngoài cùng mỗi trang web, có nhiệm vụ bao toàn bộ nội dung của trang web

<head> </head>: Phần mở đầu. Có nhiệm vụ khai báo tất cả thông tin của trang web

<title> </title>: Thẻ tiêu đề: Dùng để khai báo tiêu đề cho mỗi trang web. Tiêu đề này sẽ hiển thị ở thanh tab trên cùng của trình duyệt.

<body> </body>: Phần nội dung. Chứa tất cả nội dung sẽ hiển thị ra trên trình duyệt.

<h1> </h1> và <p> </p> : Các thẻ định dạng văn bản

c) Thẻ HTML

- Là các phần tử được bao quanh bởi các cặp ngoặc nhọn:

<tagname> Nội dung ở đây</tagname>

- Các thẻ trong HTML thường theo cặp

Ví dụ: <p> </p>

- Đầu tiên là thẻ bắt đầu (opening tag). Thẻ tiếp theo là thẻ kết thúc (closing tag)

- Thẻ kết thúc được viết giống như thẻ bắt đầu, chỉ khác thêm dấu gạch chéo (/) được chèn trước tên thẻ.

d) Trình duyệt web

- Có nhiều trình duyệt web hiện đang được sử dụng phổ biến trên như Chrome, IE, Firefox, Safari,… Mục đích của chúng là đọc các tài liệu HTML và hiển thị nội dung của chúng.

- Trình duyệt không hiển thị các thẻ HTML nhưng sử dụng nó để xác định cách hiển thị nội dung tài liệu.

e) Thẻ <!DOCTYPE>

- Thẻ khai báo <!DOCTYPE> dùng để xác định định dạng của tài liệu. Và giúp các trình duyệt hiển thị các trang web một cách chính xác.

- Thẻ <!DOCTYPE> chỉ xuất hiện một lần, ở đầu trang trước bất kỳ thẻ HTML nào.

- Thẻ không phân biệt chữ hoa và chữ thường. Nên <!DOCTYPE> hay <!doctype> đều giống nhau.

- Khai báo cho định dạng html là:  <!DOCTYPE html>.

**3.2.2 CSS**

- Cascading Style Sheets, thường được viết tắt là CSS, là một ngôn ngữ định kiểu được sử dụng để định dạng – trình bày một tài liệu HTML hoặc XML (bao gồm các ngôn ngữ XML khác nhau như SVG hoặc XHTML). CSS mô tả quy cách các phần tử được cấu trúc phải được hiển thị trên màn hình, trên giấy, trong lời nói hoặc trên các phương tiện khác.

**3.2.3 Javascript**

a) Khái niệm: JavaScrip là ngôn ngữ lập trình HTML và Web. JavaScript là một ngôn ngữ kịch bản dựa trên đối tượng nhằm phát triển các ứng dụng Internet chạy trên phía client và phía server. Javascript rất hữu ích trong việc xây dựng các hệ thống HTML có thể tương tác với người dùng.

b) Tại sao phải dùng Javascript ?

- Javascript là một trong 3 ngôn ngữ được tất cả nhà phát triển web sử dụng

- Ưu điểm của Javascript:

+) Sự tương tác Server ít hơn.

+) Phản hồi thông tin ngay lập tức tới khách truy cập.

+) Khả năng tương tác với người dùng tăng lên.

+) Giao diện phong phú hơn.

- Nhược điểm của Javascript:

+) Client-side JavaScript không cho phép đọc hoặc ghi các tập tin vì lý do bảo mật

+) Không được sử dụng cho các ứng dụng mạng, vì không hỗ trợ.

+) Không hỗ trợ xử lý đa luồng hay đa xử lý

**3.2.4 Bootstrap**

- Bootstrap là một framework cho phép thiết kế web động nhanh hơn và dễ dàng hơn. Bootstrap là bao gồm các HTML templates, CSS templates và Javascript tạo ra những cái cơ bản có sẵn như: typography, forms, buttons, tables, navigation, modals, image carousels và nhiều thứ khác. Trong bootstrap có thêm các plugin Javascript trong nó. Giúp cho việc thiết kế web động của bạn dễ dàng hơn và nhanh chóng hơn.

**3.2.5 Java**

a) Tổng quan về ngôn ngữ Java

- Java là một trong những ngôn ngữ lập trình hướng đối tượng. Nó được sử dụng phát triển phần mềm, trang web, game hay ứng dụng trên các thiết bị di động.

- Java được tạo ra với tiêu chí “Viết một lần, thực thi khắp nơi”. Bởi Java là một ngôn ngữ lập trình có khả năng biên dịch và khả năng thông dịch. Chương trình nguồn viết bằng ngôn ngữ Java có đuôi (\*.java) đầu tiên được biên dịch thành tập tin có đuôi (\*.class) và sau đó sẽ được trình thông dịch thông dịch thành mã máy. Các Java class file có thể được dùng ở bất kỳ flatform nào

- Ưu điểm: Phương pháp này giúp các đoạn mã viết bằng Java có thể chạy được trên nhiều nền tảng khác nhau. Với điều kiện máy ảo Java (JVM) có hỗ trợ chạy trên nền tảng này

- Nhược điểm: Quá trình chạy các đoạn mã Java chậm hơn các ngôn ngữ biên dịch khác

b) Các dạng công nghệ Java

- Desktop applications – J2SE

+) Phiên bản chuẩn – Java 2 Standard Edition. J2SE hỗ trợ viết các ứng dụng đơn, ứng dụng client-server

+) Java Applications: ứng dụng Java thông thường trên desktop

+) Java Applets: ứng dụng nhúng hoạt động trong trình duyệt web

- Server applications – J2EE

+) Nền tảng Java 2, phiên bản doanh nghiệp – Java 2 Enterprise Edition. Hỗ trợ phát triển các ứng dụng thương mại

+) Chạy trên máy chủ lớn với sức mạnh xử lý và dung lượng bộ nhớ lớn, hỗ trợ gắn liền với servlet, jsp và XML.

- Mobile application – J2ME

+) Phiên bản thu nhỏ - Java 2 Micro Edition

+) Hỗ trợ viết các ứng dụng trên các thiết bị di động, không dây, thiết bị nhúng,…

c) Các dạng ứng dụng của Java

Dùng Java ta có thể viết các dạng ứng dụng sau:

* Ứng dụng độc lập (console application)
* Ứng dụng Applets
* Ứng dụng giao diện (GUI application)
* Ứng dụng Web (Servlet và Jsp)
* Ứng dụng nhúng (embedded applcation)
* Ứng dụng cơ sở dữ liệu (database application)
* Games

**3.2.6 Spring Boot**

a) Spring Boot là gì ?

- Spring Boot là một module của Spring Framework, cung cấp tính năng RAD (Rapid Application Development) – Phát triển ứng dụng nhanh.

- Spring Boot được dùng để tạo các ứng dụng độc lập dựa trên Spring

- Với Spring Boot các cấu hình được giảm thiểu tối đa. Spring Boot hỗ trợ các bộ chứa nhúng (embedded containers) điều này cho phép các ứng dụng web có thể chạy độc lập mà không cần phải khai lên các Web Server.

b) Ưu điểm của Spring Boot

- Có các tính năng của Spring Framework

- Tạo ứng dụng độc lập, có thể chạy java -jar (cho cả java web)

- Nhúng trực tiếp các ứng dụng server (Tomcat, Jetty,…) do đó không cần phải triển khai file WAR

- Cấu hình ít, tự động cấu hình bất kì khi nào có thể -> giảm thời gian viết code, tăng năng suất

- Cung cấp nhiều plugin

- Chuẩn cho mô hình kiến trúc Microservice (Cloud support, giảm việc setup, config, các thư viện hỗ trợ,…)

**3.2.7 Zuul**

- Là địa chỉ truy xuất duy nhất để gọi vào các microservice

- Zuul cũng sử dụng Ribbon để gọi tới các service

- Điểm đặc biệt của Zuul đó là ZuulFilter: ZuulFilter sẽ xử lý các request theo cơ chế pipeline

+) PRE: xử lý trước khi routing đến service

+) ROUTING: thay đổi routing

+) POST: xử lý sau khi service trả về kết quả

+) ERROR: xử lý khi có lỗi xảy ra

**3.2.8 Eureka server**

- Eureka server là nơi cho phép các dịch vụ Microservice tự đăng ký mình vào danh sách các service hoạt động lúc khởi chạy

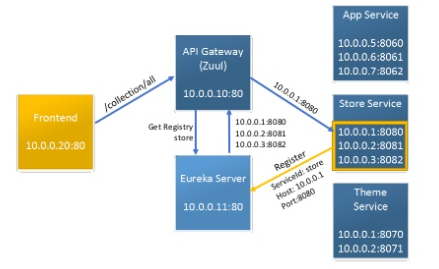
- Server này sẽ chịu trách nhiệm lưu trữ và cung cấp thông tin của các service trong một hệ thống Microservice

- Khi một service đăng ký thông tin với server, nó cung cấp các thông tin như: host, port, trạng thái của service. Do đó, server này có thể dễ dàng cung cấp thông tin về các service khi chúng được gọi tới từ các service khác

- Máy chủ Eureka khi chạy sẽ chạy mặc định trên cổng 8761

**3.2.9 Ribbon**

Ribbon là một thư viện của Netflix chủ yếu sẽ cung cấp các thuật toán cân bằng tải phía client.



Như ví dụ trên ta thấy:

- Đầu tiên các thông tin cần đăng ký lên Eureka server gồm: host, port, serviceId. Khi có một request gửi vào sẽ đi qua cổng gateway (zuul) và sẽ gửi đến Eureka server để tìm service tương ứng với request (Store Service) đang chạy ở đâu. Sau đó Eureka server sẽ trả về danh sách các instance đang chạy tại Store Service (10.0.0.1.8080 | 10.0.0.2.8081 | 10.0.0.3.8082). Nhiệm vụ tiếp theo phía Zuul sẽ lựa chọn ra một địa chỉ để chạy, thì việc lựa chọn đó chính là nhiệm vụ của Ribbon. Ribbon là thư viện của Netflix, nó sẽ nắm giữ cả ba địa chỉ khi Eureka server trả về (10.0.0.1.8080 | 10.0.0.2.8081 | 10.0.0.3.8082) và ribbon sẽ có các giải thuật để lựa chọn địa chỉ nào là địa chỉ tiếp theo và sẽ trả về

**3.2.10 MySQL Server**

- MySQL là một hệ thống quản trị cơ sở dữ liệu mã nguồn mở (gọi tắt là RDBMS) hoạt động theo mô hình client-server. Với RDBMS là viết tắt của Relational Database Management System. MySQL quản lý dữ liệu thông qua các cơ sở dữ liệu. Mỗi cơ sở dữ liệu có thể có nhiều bảng quan hệ chứa dữ liệu

- MySQL hiện nay có hai phiên bản miễn phí (MySQL Community Server) và có phí (Enterprise Server)

- MySQL là một hệ quản trị cơ sở dữ liệu rất phổ biến hiện nay

- Ưu điểm của MySQL:

+) Dễ sử dụng: MySQL là cơ sở dữ liệu tốc độ cao, ổn định, dễ sử dụng và hoạt động trên nhiều hệ điều hành cung cấp một hệ thống lớn các hàm tiện ích rất mạnh.

+) Độ bảo mật cao: MySQL rất thích hợp cho các ứng dụng có truy cập CSDL trên Internet khi sở hữu nhiều nhiều tính năng bảo mật thậm chí là ở cấp cao.

+) Đa tính năng: MySQL hỗ trợ rất nhiều chức năng SQL được mong chờ từ một hệ quản trị cơ sở dữ liệu quan hệ cả trực tiếp lẫn gián tiếp.

+) Khả năng mở rộng và mạnh mẽ: MySQL có thể xử lý rất nhiều dữ liệu và hơn thế nữa nó có thể được mở rộng nếu cần thiết.

+) Nhanh chóng: Việc đưa ra một số tiêu chuẩn cho phép MySQL để làm việc rất hiệu quả và tiết kiệm chi phí, do đó nó làm tăng tốc độ thực thi.

- Nhược điểm của MySQL:

+) Giới hạn: Theo thiết kế, MySQL không có ý định làm tất cả và nó đi kèm với các hạn chế về chức năng mà một vào ứng dụng có thể cần.

+) Độ tin cậy: Cách các chức năng cụ thể được xử lý với MySQL (ví dụ tài liệu tham khảo, các giao dịch, kiểm toán,…) làm cho nó kém tin cậy hơn so với một số hệ quản trị cơ sở dữ liệu quan hệ khác.

+) Dung lượng hạn chế: Nếu số bản ghi của bạn lớn dần lên thì việc truy xuất dữ liệu của bạn là khá khó khăn, khi đó chúng ta sẽ phải áp dụng nhiều biện pháp để tăng tốc độ truy xuất dữ liệu như là chia tải database này ra nhiều server, hoặc tạo cache MySQL.

**3.2.11 SQL Server**

- SQL Server là một hệ quản trị cơ sở dữ liệu quan hệ (Relational Database Management System (RDBMS) ) sử dụng câu lệnh SQL (Transact-SQL) để trao đổi dữ liệu giữa máy Client và máy cài SQL Server. Một RDBMS bao gồm databases, database engine và các ứng dụng dùng để quản lý dữ liệu và các bộ phận khác nhau trong RDBMS.

- SQL Server được tối ưu để có thể chạy trên môi trường cơ sở dữ liệu rất lớn (Very Large Database Environment) lên đến Tera-Byte và có thể phục vụ cùng lúc cho hàng ngàn user. SQL Server có thể kết hợp “ăn ý” với các server khác như Microsoft Internet Information Server (IIS), E-Commerce Server, Proxy Server….

- Một vài ấn bản SQL Server:

+) Enterprise : chứa tất cả cá đặc điểm nổi bật của SQL Server, bao gồm nhân bộ máy cơ sở dữ liệu và các dịch vụ đi kèm cùng với các công cụ cho tạo và quản lý phân cụm SQL Server. Nó có thể quản lý các CSDL lớn tới 524 petabytes và đánh địa chỉ 12 terabytes bộ nhớ và hỗ trợ tới 640 bộ vi xử lý(các core của cpu)

+) Standard : Rất thích hợp cho các công ty vừa và nhỏ vì giá thành rẻ hơn nhiều so với Enterprise Edition, nhưng lại bị giới hạn một số chức năng cao cấp (advanced features) khác, edition này có thể chạy tốt trên hệ thống lên đến 4 CPU và 2 GB RAM.

+) Developer : Có đầy đủ các tính năng của Enterprise Edition nhưng được chế tạo đặc biệt như giới hạn số lượng người kết nối vào Server cùng một lúc…. Ðây là phiên bản sử dụng cho phát triển và kiểm tra ứng dụng. Phiên bản này phù hợp cho các cá nhân, tổ chức xây dựng và kiểm tra ứng dụng

+) Workgroup: ấn bản SQL Server Workgroup bao gồm chức năng lõi cơ sở dữ liệu nhưng không có các dịch vụ đi kèm. Chú ý phiên bản này không còn tồn tại ở SQL Server 2012.

+) Express : SQL Server Express dễ sử dụng và quản trị cơ sở dữ liệu đơn giản. Được tích hợp với Microsoft Visual Studio, nên dễ dàng để phát triển các ứng dụng dữ liệu, an toàn trong lưu trữ, và nhanh chóng triển khai. SQL Server Express là phiên bản miễn phí, không giới hạn về số cơ ở dữ liệu hoặc người sử dụng, nhưng nó chỉ dùng cho 1 bộ vi xử lý với 1 GB bộ nhớ và 10 GB file cơ sở dữ liệu. SQL Server Express là lựa chọn tốt cho những người dùng chỉ cần một phiên bản SQL Server 2005 nhỏ gọn, dùng trên máy chủ có cấu hình thấp, những nhà phát triển ứng dụng không chuyên hay những người yêu thích xây dựng các ứng dụng nhỏ.

**3.3 Phương pháp nghiên cứu**

- Tham khảo giáo trình, tài liệu liên quan đến nội dung đề tài

- Tìm hiểu, nghiên cứu lý thuyết về các kỹ thuật thiết kế và xây dựng website

- Phân tích, thiết kế đối tượng và cơ sở dữ liệu quan hệ

- Xây dựng chương trình theo mô hình tiến hóa và kiểm thử

- Thu thập ý kiến người sử dụng để bảo trì và cập nhật

**Phần 4: Kế hoạch và dự kiến kết quả đạt được**

**4.1. Kế hoạch thực hiện đề tài**

|  |  |  |  |
| --- | --- | --- | --- |
| **STT** | **Công việc** | **Kết quả dự kiến** | **Thời gian** |
| 1 | Khảo sát thu thập thông tin | Bản đặc tả yêu cầu | 1 tuần |
| 2 | Viết đề cương, lập kế hoạch | Bản đề cương | 1 tuần |
| 3 | Khảo sát, nghiên cứu thực tế tại công ty | Báo cáo các thông tin đã tìm hiểu được | 4 tuần |
| 4 | Phân tích thiết kế hệ thống, xây dựng CSDL | Bản phân tích thiết kế hệ thống, CSDL | 1 tuần |
| 5 | Xây dựng một số chức năng của ứng dụng | Viết ứng dụng | 6-7 tuần |
| 6 | Kiểm thử chức năng | Hoàn thiện ứng dụng | 1 tuần |
| 7 | Viết báo cáo | Báo cáo | 1 tuần |

**4.2. Dự kiến kết quả đạt được**

Xây dựng ứng dụng đăng tin thuê nhà để đáp ứng nhu cầu của mọi người

**Phần 5: Tài liệu tham khảo**

W3schools, <https://www.w3schools.com/>

Microservice, <https://www.nginx.com/blog/introduction-to-microservices/> ,

<https://www.edureka.co/blog/what-is-microservices/> ,

<https://viblo.asia/p/gioi-thieu-ve-kien-truc-microservices-4P8566O35Y3>

**GIẢNG VIÊN HƯỚNG DẪN SINH VIÊN**

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