**BỘ GIÁO DỤC VÀ ĐÀO TẠO**

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

**KHOA CÔNG NGHỆ THÔNG TIN**

--------------------

****

**TIỂU LUẬN CHUYÊN NGÀNH**

**TÌM HIỂU SPRING BOOT**

**VÀ ỨNG DỤNG XÂY DỰNG WEBSITE HỌC IELTS**

**NHÓM SINH VIÊN THỰC HIỆN:**

**ĐOÀN LÊ BÌNH AN – MSSV: 18110244**

**NGUYỄN VĂN PHÁP – MSSV: 18110333**

**GIÁO VIÊN HƯỚNG DẪN:**

**TS.NGUYỄN THÀNH SƠ****N**

**BỘ GIÁO DỤC VÀ ĐÀO TẠO**

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

**KHOA CÔNG NGHỆ THÔNG TIN**

--------------------

****

**TIỂU LUẬN CHUYÊN NGÀNH**

**TÌM HIỂU SPRING BOOT**

**VÀ XÂY DỰNG WEBSITE MINH HỌA**

**NHÓM SINH VIÊN THỰC HIỆN:**

**ĐOÀN LÊ BÌNH AN – MSSV: 18110244**

**NGUYỄN VĂN PHÁP – MSSV: 18110333**

**GIÁO VIÊN HƯỚNG DẪN:**

**TS.NGUYỄN THÀNH SƠN**

**ĐH SƯ PHẠM KỸ THUẬT TP.HCM XÃ HỘI CHỦ NGHĨA VIỆT NAM**

**KHOA CNTT Độc lập – Tự do – Hạnh Phúc**

**\*\*\*\*\*\*\* \*\*\*\*\*\*\***

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

Họ và tên Sinh viên 1 : NGUYỄN VĂN PHÁP MSSV 1: 18110333

Họ và tên Sinh viên 2 : ĐOÀN LÊ BÌNH AN MSSV 2: 18110244

Ngành: Công nghệ Thông tin

Tên đề tài: TÌM HIỂU SPRING BOOT VÀ XÂY DỰNG WEBSITE MINH HỌA

Họ và tên Giáo viên hướng dẫn: TS.NGUYỄN THÀNH SƠN

**NHẬN** **XÉT**

1. Về nội dung đề tài & khối lượng thực hiện:

............................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................

1. Ưu điểm:

............................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................

1. Khuyết điểm

............................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................

1. Đề nghị cho bảo vệ hay không ?
2. Đánh giá loại :
3. Điểm :

Tp*. Hồ Chí Minh, ngày* *tháng năm 201*

Giáo viên hướng dẫn

*(Ký & ghi rõ họ tên**)*

# LỜI CẢM ƠN

Trước hết chúng em xin tỏ lòng biết ơn sâu sắc đến TS.Nguyễn Thành Sơn, thầy đã tận tình hướng dẫn, giúp đỡ chúng em rất nhiều trong suốt quá trình thực hiện tiểu luận này. Sự hiểu biết sâu sắc về khoa học cũng như kinh nghiệm của thầy chính là tiền đề giúp chúng hoàn thành tiểu luận.

Chúng em trân trọng cảm ơn đến các thầy, cô giáo Khoa công nghệ thông tin - Trường đại học Sư Phạm Kỹ Thuật thành phố Hồ Chí Minh, các thầy cô bộ môn đã giảng dạy, tạo điều kiện cho chúng em hoàn thành bài tiểu luận này.

Đồng thời chúng tôi cũng gửi lời cảm ơn đến anh chị và bạn bè sinh viên cùng chuyên ngành cũng như các anh em trên các blog, diễn đàn đã chia sẻ tài liệu, góp ý kiến giúp chúng tôi trong suốt thời gian qua.

Mặc dù đã cố gắn hết sức, những cuốn tiểu luận này không tránh khỏi những thiếu sót. Rất mong quý thầy cô và bạn bè đóng góp ý kiến để chúng em hoàn thiện bài tiểu luận này tốt hơn.

Xin chân thành cảm ơn.

Tp.Hồ Chí Minh, ngày 20 tháng 09 năm 2021

Nhóm sinh viên thực hiện

Đoàn Lê Bình An & Nguyễn Văn Pháp

# MỤC LỤC

[LỜI CẢM ƠN 2](#_Toc82263641)

[MỤC LỤC 3](#_Toc82263642)

[DANH MỤC HÌNH 6](#_Toc82263643)

[DANH MỤC BẢNG 7](#_Toc82263644)

[DANH SÁCH TỪ VIẾT TẮT 8](#_Toc82263645)

[MỞ ĐẦU 9](#_Toc82263646)

[i. TÍNH CẤP THIẾT CỦA ĐỀ TÀI 9](#_Toc82263647)

[ii. MỤC ĐÍCH CỦA ĐỀ TÀI 10](#_Toc82263648)

[iii. CÁCH TIẾP CẬN VÀ PHẠM VI NGHIÊN CỨU 10](#_Toc82263649)

[iv. Ý NGHĨA KHOA HỌC VÀ THỰC TIỄN 11](#_Toc82263650)

[NỘI DUNG 11](#_Toc82263651)

[Chương 1 CƠ SỞ LÝ THUYẾT 11](#_Toc82263652)

[1.1. SPRING BOOT 11](#_Toc82263653)

[1.1.1. Giới thiệu 11](#_Toc82263654)

[1.1.2. Các tính năng trong spring boot 13](#_Toc82263655)

[1.1.3. Auto-congiguration trong spring boot 15](#_Toc82263656)

[1.1.4. Phát triển ứng dụng web với spring boot 20](#_Toc82263657)

[1.1.4. Phát triển Web Service trong Spring Boot 36](#_Toc82263658)

[1.1.5. Spring Boot JPA 36](#_Toc82263659)

[1.2. SPRING SECURITY 47](#_Toc82263660)

[1.2.1 Giới thiệu 47](#_Toc82263661)

[1.2.2 Các thành phần cốt lõi 47](#_Toc82263662)

[1.2.3. Áp dụng Spring Security trong thực tế 49](#_Toc82263663)

[1.3. AJAX 50](#_Toc82263664)

[1.3.1. Giới thiệu 50](#_Toc82263665)

[1.3.2. Sử dụng Ajax trong xây dựng website 51](#_Toc82263666)

[1.4. MYSQL 64](#_Toc82263667)

[1.4.1. Thuật ngữ thường gặp của MySQL 64](#_Toc82263668)

[1.4.2. MySQL và lịch sử phát triển 65](#_Toc82263669)

[1.4.3. Ưu điểm và nhược điểm của MySQL 66](#_Toc82263670)

[1.4.4. Hoạt động của MySQL 66](#_Toc82263671)

[1.4.5. Lý do Sử Dụng MySQL 67](#_Toc82263672)

[1.5. Ý TƯỞNG VỀ CÁCH LẬP TRÌNH ĐỂ TẠO MỘT WEBSITE HỌC IELTS TRỰC TUYẾN. 68](#_Toc82263673)

[1.5.1. Ý tưởng lập ra website học IELTS trực tuyến 68](#_Toc82263674)

[1.5.2. Cách triển khai cài đặt 70](#_Toc82263675)

[Chương 2. TỔNG QUAN VỀ CÁC TRANG WEBSITE HỌC IELTS TRỰC TUYẾN NỔI TIẾNG 71](#_Toc82263676)

[2.1. IELTS Fighter 71](#_Toc82263677)

[2.1.1. Giới thiệu 71](#_Toc82263678)

[2.1.2. Lịch sử phát triển 71](#_Toc82263679)

[2.1.3. Ưu và nhược điểm 71](#_Toc82263680)

[2.1.4. Các tính năng nổi bật của trang web 71](#_Toc82263681)

[2.2. VOA Learning English 71](#_Toc82263682)

[2.1.1. Giới thiệu 71](#_Toc82263683)

[2.1.2. lịch sử phát triển 72](#_Toc82263684)

[2.1.3. Ưu và nhược điểm 72](#_Toc82263685)

[2.1.4. Các tính năng nổi bật của trang web 72](#_Toc82263686)

[2.3. Elllo.org 72](#_Toc82263687)

[2.3.1. Giới thiệu 73](#_Toc82263688)

[2.3.2. Lịch sử phát triển 73](#_Toc82263689)

[2.3.3. Ưu và nhược điểm 73](#_Toc82263690)

[2.3.4. Các tính năng nổi bật của trang web: 73](#_Toc82263691)

[2.4. Duolingo 74](#_Toc82263692)

[2.4.1. Giới thiệu 74](#_Toc82263693)

[2.4.2. Lịch sử phát triển 74](#_Toc82263694)

[2.4.3. Ưu và nhược điểm 75](#_Toc82263695)

[2.4.4. Các tính năng nổi bật của trang web 75](#_Toc82263696)

[2.5. IELTS-Blog 75](#_Toc82263697)

[2.5.1. Giới thiệu 75](#_Toc82263698)

[2.5.2. Lịch sử phát triển 75](#_Toc82263699)

[2.5.3. Ưu và nhược điểm 75](#_Toc82263700)

[2.5.4. Các tính năng nổi bật của trang web 76](#_Toc82263701)

[Chương 3 PHÂN TÍCH THIẾT KẾ HỆ THỐNG WEBSITE HỌC IELTS TRỰC TUYẾN 76](#_Toc82263702)

[3.1. ĐẶC TẢ ĐỀ TÀI 76](#_Toc82263703)

[3.1.1. Mô tả chung về hệ thống website học ielts trực tuyến 76](#_Toc82263704)

[3.1.2. Xác định giới hạn hệ thống thông tin của NHÓM 78](#_Toc82263705)

[3.2. THIẾT KẾ THÀNH PHẦN DỮ LIỆU 79](#_Toc82263706)

[3.2.1. Mô hình thực thể kết hợp (ERD) 79](#_Toc82263707)

[3.2.2. Danh sách lược đồ quan hệ 80](#_Toc82263708)

[3.2.3. Từ điển dữ liệu 81](#_Toc82263709)

[3.3. MÔ HÌNH USECASE 86](#_Toc82263710)

[3.4. PHÂN TÍCH THIẾT KẾ THÀNH PHẦN CHỨC NĂNG 87](#_Toc82263711)

[3.4.1. BIỂU ĐỒ SEQUENCE DIAGRAM 87](#_Toc82263712)

[3.5. Phân tích thiết kế và đặc tả thành phần giao diện 88](#_Toc82263713)

[3.6. KIẾN TRÚC HỆ THỐNG 89](#_Toc82263714)

[Chương 4 KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN 91](#_Toc82263715)

[4.1 KẾT QUẢ ĐẠT ĐƯỢC 91](#_Toc82263716)

[4.2 HẠN CHẾ 91](#_Toc82263717)

[4.3 HƯỚNG PHÁT TRIỂN 91](#_Toc82263718)

[TÀI LIỆU THAM KHẢO 92](#_Toc82263719)

[PHỤ LỤC 1: HƯỚNG DẪN TRIỂN KHAI ỨNG DỤNG 94](#_Toc82263720)

# DANH MỤC HÌNH

[Hình 1. 1 Các tính năng trong spring boot [4] 13](#_Toc78916041)

[Hình 1. 2 Auto-congiguration trong spring boot [5] 16](#_Toc78916042)

[Hình 1. 3 Annotations trong spring boot[5] 18](#_Toc78916043)

[Hình 1. 4 Sơ đồ luồng đi [6] 22](#_Toc78916044)

[Hình 1. 5 Chi tiết cấu trúc source code 28](#_Toc78916045)

[Hình 1. 6 Tổ chức source code theo mô hình 3 lớp [8] 29](#_Toc78916046)

[Hình 1. 7 Tổ chức source code theo mô hình kết hợp 30](#_Toc78916047)

[Hình 1. 8 Cấu trúc thư mục project spring boot 38](#_Toc78916048)

[Hình 1. 9 Dữ liệu 40](#_Toc78916049)

[Hình 1. 10 Các hàm hỗ trợ trong Spring boot JPA 44](#_Toc78916050)

[Hình 3. 1Mô hình Client Server [12] 55](#_Toc78916051)

[Hình 3. 2 Cách MySQL hoạt động [12] 58](#_Toc78916052)

# DANH MỤC BẢNG

# DANH SÁCH TỪ VIẾT TẮT

|  |  |
| --- | --- |
| AJAX | Asynchronous JavaScript and XML |
| AOP | Aspect – oriented programming |
| API | Application programming interface |
| CSDL | Cở sở dữ liệu |
| DAO | Data Access Object |
| DI | Dependency Injection |
| DOM | Document Object Model |
| EJB | Enterprise Java Bean |
| HTML | HyperText Markup Language |
| IoC | Inversion of Control |
| JSM | Java Message System |
| JVM | Java Virtual Machine |
| MOM | Message - Oriented Middleware |
| OOP | Object – oriented programming |
| ORM | Object Relational Mapping |
| RDBMS | Relational database management system |
| SPA | Single page application |
| URL | Uniform Resource Locator |

# MỞ ĐẦU

# TÍNH CẤP THIẾT CỦA ĐỀ TÀI

Vai trò quan trọng của Tiếng Anh đối với những người làm việc trong lĩnh vực Công nghệ thông tin (CNTT) cũng như các lĩnh vực khác là điều không thể phủ nhận. Tuy nhiên việc học một ngoại ngữ, dù rất phổ thông như tiếng Anh, dường như vẫn luôn là một bức tường cao và dày rất khó vượt qua.

Nhiều trường đại học hiện nay đã lấy điểm thi Ielts là điều kiện đầu ra của sinh viên. Tuy mức điểm của mỗi trường không giống nhau, nhưng Ielts đã thể hiện phần nào tầm quan trọng của mình trong quá trình học tập của sinh viên. Vậy điều kiện tiên quyết của mỗi sinh viên muốn ra trường là học và thi Ielts. Bên cạnh đó nhu cầu học tập và làm việc trên máy vi tính ngày càng tăng. Vì vậy, một phần mềm học Ielts trực tuyến là một nhu cầu thiết yếu.

Mặt khác, ngày nay hầu hết các hệ thống đều được phát triển trên nền web bởi vì các ứng dụng web đem lại khả năng phát triển và triển khai sản phẩm nhanh chóng, giảm thời gian đưa sản phẩm ra thị trường, có khả năng cập nhật tính năng liên tục hàng ngày.

Ứng dụng web có thời gian phát triển nhanh, với chi phí phát triển và triển khai thấp, tin cậy, có thể truy cập từ bất cứ đâu, mọi người có thể sử dụng mà không mất thời gian tìm hiểu nhờ giao diện trực quan.

Chính vì những lý do trên nên chúng em chọn đề tài “Tìm hiểu Framework Spring Boot và xây dựng website minh họa”. Nội dung tiểu luận này sẽ tập trung tìm hiểu và giới thiệu về Spring Boot, một framework với những đặc tính vượt trội và ưu việt trong việc tạo ra các hệ thống đòi hỏi sự module hóa và có khả năng sử dụng lại cao. Đồng thời tìm hiểu thêm một số kỹ thuật khác hiện đang được các công ty phần mềm sử dụng để tích hợp với Spring Boot tạo nên một ứng dụng web như: AJAX, MySQL.

Sau khi tìm hiểu chúng tôi sẽ vận dụng kết quả tìm hiểu được vào việc xây dựng

một ứng dụng nhằm mục đích minh họa cho phần lý thuyết đã trình bày. Chúng em sẽ xây dựng một Website học Ielts trực tuyến .

# MỤC ĐÍCH CỦA ĐỀ TÀI

Nghiên cứu và tìm hiểu về các công nghệ spring boot, ajax và các vấn đề liên quan trong quá trình xây dựng website.

Ứng dụng các công nghệ đã tìm hiểu để xây dựng một Website học Ielts trực tuyến

# CÁCH TIẾP CẬN VÀ PHẠM VI NGHIÊN CỨU

**Đối tượng nghiên cứu**

* Spring boot
* MySQL

**Phạm vi nghiên cứu**

Tiểu luận sẽ tập trung trình bày kết quả nghiên cứu của nhóm về các nội dung: Spring boot, Mysql, Ajax. Mỗi phần nhóm sẽ giới thiệu sơ lược và trình bày những nội dung cơ bản nhất mà nó mang lại cho các nhà phát triển phần mềm.

Sau khi tìm hiểu, nhóm sẽ vận dụng kết quả tìm hiểu được vào việc xây dựng một ứng dụng nhằm mục đích minh họa cho phần lý thuyết đã trình bày.

**Phương pháp nghiên cứu**

Tìm kiếm và nghiên cứu các tài liệu về Spring Boot và các công nghệ có liên quan đến việc phát triển một ứng dụng web enterprise như MySQL, Maven của các tác giả trong và ngoài nước, các bài báo, thông tin trên mạng,… sau đó chọn lọc và sắp xếp lại theo ý tưởng của mình.

Dựa trên kết quả tìm hiểu được để xây dựng một ứng dụng Website học Ielts trực tuyến có áp dụng tất cả những nội dung đã nghiên cứu nhằm mục đích minh họa cho phần cơ sở lý thuyết sẽ trình bày trong nội dung luận văn này.

# Ý NGHĨA KHOA HỌC VÀ THỰC TIỄN

Trong thời đại mà Tiếng Anh là vô cùng quan trọng, việc ứng dụng vào hoạt động “Tìm hiểu Spring Boot Framwork và xây dựng website minh họa” đã mang lại nhiều ý nghĩa như sau

* Đề xuất nghiên cứu tìm hiểu về spring boot framework và xây dựng web site minh họa cung cấp nguồn tài liệu cho độc giả các kiến thức về spring boot trong hoàn cảnh tài liệu nghiên cứu tiếng việt về spring boot còn ít
* Kết quả nghiên cứu giúp hiểu được các tính năng trong spring boot, các phát triển một ứng dụng web với spring boot.
* Kết quả nghiên cứu giúp xây dựng thành công một website Website học Ielts trực tuyến.

# NỘI DUNG

# Chương 1 CƠ SỞ LÝ THUYẾT

## SPRING BOOT

### Giới thiệu

#### Springboot

Spring được xem là một Framework mang đến khá nhiều ưu điểm cho việc phát triển phần mềm. Thế nhưng không thể phủ nhận rằng nó vẫn tồn tại nhiều hạn chế, đặc biệt là quá nhiều cấu hình được sử dụng trong phần mềm này. Chính vì thế, Spring Boot được ra đời đời nhằm khắc phục những điểm yếu và xây dựng một hướng phát triển mới, tốt hơn. [2]

Spring Boot thực chất đây chính là một phần mở rộng quan trọng của Spring framework. Đây chính là ứng dụng giúp làm giảm được sự phức tạp khi lập trình Spring cho các thành viên. Theo đó các thành viên chỉ phải tập trung cho việc nâng cao business cho ứng dụng.[1]

Sử dụng Spring boot được xem là một tiêu chuẩn cho sự cho cấu hình giúp nâng cao năng suất cho developer. Được xem là cấu hình thiết kế phần mềm tối ưu nhất, tạo ra ứng dụng độc lập được ứng dụng trong tương lai.[2]

Đối với Spring Boot thì có thể thực hiện một dự án Spring được một cách nhanh chóng và tiến hành đơn giản nhất. Khi đó sẽ sử dụng cấu hình Sublime Text để phát triển luôn mà không cần phải thực hiện lại bước cài đặt netbean hay eclipse nữa.

Bạn có thể tìm thấy dự án Spring Boot tại http://projects.spring.io/spring-boot/ và tài liệu rất phong phú tại

<http://docs.spring.io/spring-boot/docs/current/reference/htmlsingle/> [2]

#### Ưu điểm.

Như đã giới thiệu ở trên thì bạn có thể viết được nghĩa của Spring Boot là gì ? Sử dụng Spring Boot sẽ mang đến những lợi ích nổi bật như sau:

* Giúp tạo được ứng dụng độc lập dựa trên Spring, có thể tự chạy được  java – jar.[2]
* Có ít cấu hình, có khả năng tự động cấu hình lại Spring khi cần, từ đó giúp các thành viên có thể tiết kiệm thời gian viết code và tăng thêm năng suất.
* Giữ đầy đủ các tính năng của Spring Framework.[2]
* Spring boot không yêu cầu cấu hình XML và không sinh code cấu hình.[2]
* Không cần phải triển khai file WAR mà thực hiện nhúng trực tiếp các ứng dụng server.[2]
* Giúp cung cấp nhiều plugin. [2]

#### Công dụng của Spring boot trong đề tài nghiên cứu.

Trong thực tế Spring boot được sử dụng với 2 mục đích sau :

* spring boot để phát triển web service

Các hệ thống web lớn hiện nay thường có 2 nền tảng là app và web, khi chúng ta dùng web order hàng hóa thì ngay lập tức app sẽ đồng bộ theo dữ liệu mà chúng ta mới thao tác trên web. Vậy để làm được điều đó tiki đã sử dụng cơ chế web service, dùng chung 1 database và các hàm sử dụng cho web và app cũng sẽ chung với nhau. Có nghĩa là chúng ta chỉ cần viết hàm order thì hàm đó sẽ dùng chung luôn cho web và app

Để xây dựng web service trong java, chúng ta cũng có rất nhiều cách tuy nhiên nó khá là khó và cũng hơi phức tạp. Vì vậy nhà cung cấp Spring framework đã phát triển ra cho chúng ta 1 công nghệ cực kì hữu ích đó chính là Spring boot

Khi spring boot ra đời, việc xây dựng API web service không còn là cái gì đó quá to tát và nó cũng giúp cho chúng ta tiết kiệm được thời gian rất nhiều

Trong giới hạn đề tài nhóm xây dựng webservice và sử dụng cho website kết hợp với ajax với mục đích đem lại trải nghiệm tốt hơn cho người dùng

* Spring boot phát triển ứng dụng web

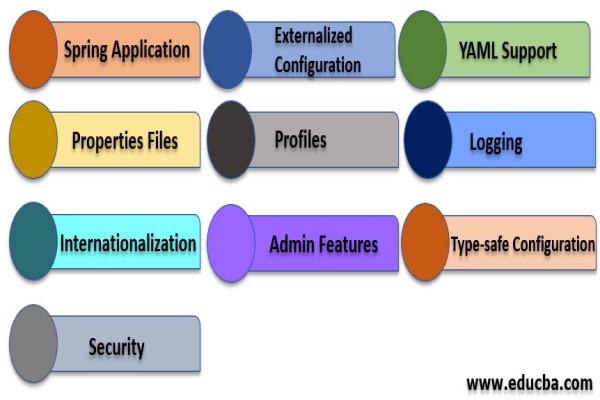
Như chúng ta cũng đã biết Spring framework có cho ra đời 1 nền tảng làm web đó là Spring mvc. Tuy nhiên trong quá trình làm việc với spring mvc, nhà cung cấp họ thấy spring mvc cấu hình quá nhiều và có rất nhiều thư viện dùng thường xuyên tuy nhiên mỗi khi tạo project spring mvc mới chúng ta phải tải lại những thư viện đó vì vậy rất là mất công và mất thời gian. Từ những khó khăn đó, công nghệ spring boot ra đời để giảm thiểu cấu hình và tích hợp sẵn những thư viện hay dùng để mỗi khi tạo project mới không phải cấu hình nhiều và tải lại những thư viện dùng nhiều

Trong giới hạn đề tài nhóm sử dụng spring boot để phát triển website

### Các tính năng trong spring boot

Khi một nhà phát triển sử dụng Spring, người ta có thể dễ dàng loại bỏ các mã soạn sẵn, nhưng trong quá trình hoàn thành chức năng được nhập từ mã khác, mọi thứ trong mã cần phải tương thích, điều này trở thành một nhiệm vụ lớn đối với nhà phát triển. Vì vậy, để giải quyết quy trình khắc phục sự cố tương thích rườm rà này. Spring Boot có các tính năng trong để xử lý nó. [4]

Bây giờ chúng ta hãy xem xét các tính năng khác nhau trong Spring Boot :[4]



**Hình 1. 1 Các tính năng trong spring boot [4]**

* **Spring Application**

Một trong những tính năng của SpringBoot đó là Spring Application. Đây là một class trong Spring Boot cho phép nhà phát triển khởi động ứng dụng Spring một cách thuận tiện bằng cách khởi động nó từ phương thức chính. Nhiều khi sử dụng phương thức .run () sẽ cho phép ứng dụng được chạy. [4]

* **Externalized Configuration**

Sử dụng tính năng này cho phép nhà phát triển làm việc với cùng một ứng dụng trong các môi trường khác nhau. Từ các file properties, các đối số biến môi trường của file YAML được sử dụng trực tiếp cho các ***Bean*** bằng annotation ***@Value*** hoặc sử dụng ***@ConfigurationProperties*** để liên kết với các đối tượng có cấu trúc. [4]

* **Hỗ trợ file JAML**

Như chúng ta đã nói về việc sử dụng các file YAML cho **Externalized Configuration**, Spring Boot cung cấp khả năng hỗ trợ YAML linh hoạt để chỉ định cấu hình phân cấp. [4]

* **Các file Properties**

Việc hỗ trợ các file YAML như chúng ta đã thảo luận ở trên là một giải pháp thay thế thành công cho các file properties. Các file properties không là gì ngoài một văn bản bao gồm một tập hợp các thuộc tính cần thiết để ứng dụng chạy. Ví dụ: sử dụng server-port = 8080 có nghĩa là cổng được sử dụng để chạy máy chủ là 8080. [4]

* **Profiles**

Ngoài ra nói đến những tính năng của SpringBoot còn phải nói đến ***Profile.***Bằng cách sử dụng cấu hình, người ta có thể tách biệt các phần khác nhau của ứng dụng và cung cấp cho các môi trường khác nhau theo yêu cầu. Việc sử dụng spring.profiles.active = {EnvironmentName} chỉ cung cấp cho các cấu hình được đề cập. Các phần cần hạn chế phải được đánh dấu bằng annotation ***@Profile***. [4]

* **Ghi Log**

Trong Spring Boot, “Commons Logging” được sử dụng cho tất cả các bản ghi nội bộ. Nói chung, phần này của các tính năng vẫn không thay đổi trừ khi cần tùy chỉnh một cách rõ ràng. [4]

* **I18N**

Sử dụng tính năng này Spring Boot có thể đáp ứng cho những người dùng có sở thích ngôn ngữ khác nhau. [4]

* **Những tính năng của Admin**

Tính năng này trong Spring Boot cho phép người dùng có các tính năng liên quan đến Admin cụ thể cho ứng dụng. Việc truy cập và quản lý ứng dụng có thể được thực hiện từ xa. Người ta có thể sử dụng thuộc tính spring.application.admin.enabled để kích hoạt các tính năng liên quan đến Admin. [4]

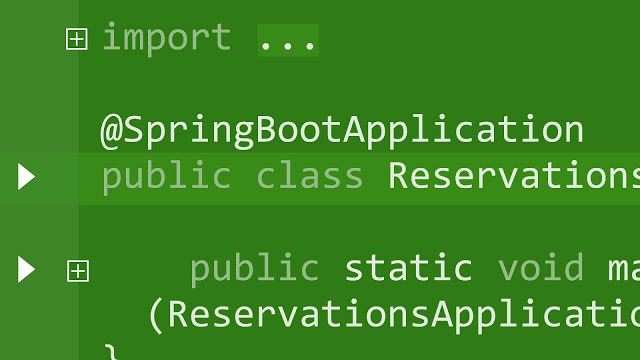
* **Type-safe Configuration**

Sử dụng annotation ***@Value (“$ {property}”)*** có thể trở nên phức tạp khi một người xử lý nhiều thuộc tính hoặc có thể dữ liệu có tính phân cấp. Spring Boot cung cấp type-safe configurations (các cấu hình an toàn)  kiểu để chống lại loại tình huống này.[4]

Và tính năng của SpringBoot cuối cùng tôi đề cập tới là Security (Bảo mật). Khi Spring Security được bật trong classpath, các ứng dụng web được bảo mật theo mặc định. Spring Boot xác định việc sử dụng httpBasic hoặc formLogin theo yêu cầu. [4]

### Auto-congiguration trong spring boot

Configuration (cấu hình) là thành phần trung tâm của Spring Framework. Mặc dù Spring Boot tự động cấu hình mọi thứ cho chúng ta nhưng vẫn phải có thứ gì đó cho Spring biết cách để chạy ứng dụng. Mục này chúng em sẽ nói về cách mà Spring Boot tự động cấu hình như thế nào.

[](https://2.bp.blogspot.com/-gT4Ym2ZocBg/WyJIW19BPXI/AAAAAAAAUpQ/zSBMhE8WS4kjOvU6QRWR0etfDmwEuv7tQCLcBGAs/s1600/550572-636247470560462813-16x9.jpg)

**Hình 1. 2**  **Auto-congiguration trong spring boot** [5]

Khi thêm Spring Boot vào ứng dụng, có một file JAR tên là springboot-autoconfigure chứa các class cấu hình (configuration classes). Mỗi class trong các class này đều có sẵn trong application's classpath và đều có cơ hội đóng góp vào việc cấu hình ứng dụng. [5]

Có một cấu hình cho Thymeleaf, một cấu hình cho Spring Data JPA, một cấu hình cho Spring MVC và một cấu hình cho hàng tá các thứ khác mà có thể muốn hoặc không muốn sử dụng trong ứng dụng Spring. [5]

Từ phiên bản Spring 4 đã hỗ trợ cấu hình có điều kiện. Cấu hình có điều kiện có sẵn trong ứng dụng, nhưng sẽ bị bỏ qua trừ khi đáp ứng được điều kiện được chỉ định. Để thực hiện điều này, ta chỉ cần implement Condition interface và override matches() method. [5]

Ví dụ, có lớp điều kiện đơn giản sau:

|  |
| --- |
| Example.java |
| package readinglist;  import org.springframework.context.annotation.Condition;  import org.springframework.context.annotation.ConditionContext;  import org.springframework.core.type.AnnotatedTypeMetadata;  public class JdbcTemplateCondition implements Condition {  @Override  public boolean matches(ConditionContext context,  AnnotatedTypeMetadata metadata) {  try {  context.getClassLoader().loadClass(  "org.springframework.jdbc.core.JdbcTemplate");  return true;  } catch (Exception e) {  return false;  }  }  } |

Ta có thể sử dụng lớp điều kiện trên khi khai báo các beans trong Java:

|  |
| --- |
| Example.java |
| @Conditional(JdbcTemplateCondition.class)  public MyService myService() {  ...  } |

Trong trường hợp này, MyService sẽ chỉ được tạo nếu và chỉ nếu JdbcTemplateCondition thoả mãn. Có nghĩa là MyService bean sẽ chỉ được tạo nếu JdbcTemplate có sẵn trên classpath. Nếu không, khai báo bean sẽ bị bỏ qua.[5]

Mặc dù điều kiện được hiển thị ở đây khá đơn giản, Spring Boot định nghĩa một số điều kiện thú vị hơn và áp dụng chúng cho các lớp cấu hình tự động trong Spring Boot. Spring Boot áp dụng cấu hình có điều kiện bằng cách định nghĩa một số conditional annotations đặc biệt và sử dụng chúng trong các lớp cấu hình của nó. Bảng sau liệt kê các chú thích có điều kiện mà Spring Boot cung cấp.[5]

[](https://3.bp.blogspot.com/-HQWxr0QYz6Q/WyJDa8hMy7I/AAAAAAAAUpE/MJKlxGaBEYECUdr_E0KNbjaQgZcBx29gQCLcBGAs/s1600/anh-fig.JPG)

**Hình 1. 3 Annotations trong spring boot[5]**

Cơ bản, bạn không bao giờ cần phải xem mã nguồn cho các lớp cấu hình tự động của Spring Boot. Nhưng để minh họa cách sử dụng annotation trong bảng trên, hãy xem đoạn trích này từ DataSourceAutoConfiguration (được cung cấp như một phần của thư viện cấu hình tự động của Spring Boot):

|  |
| --- |
| Example.java |
| @Configuration  @ConditionalOnClass({DataSource.class, EmbeddedDatabaseType.class})  @EnableConfigurationProperties(DataSourceProperties.class)  @Import({Registrar.class, DataSourcePoolMetadataProvidersConfiguration.class  })  public class DataSourceAutoConfiguration {  ...  } |
|  |
|  |
|  |
|  |
|  |
|  |
|  |
|  |

Như ta có thể thấy, DataSourceAutoConfiguration là một lớp được đánh dấu bởi @Configuration annotation (và một số thứ khác), trong đó, nó import một số cấu hình bổ sung từ các lớp cấu hình khác và định nghĩa một vài bean riêng của nó.

Điều quan trọng nhất cần lưu ý ở đây là DataSourceAutoConfiguration được đánh dấu bằng @ConditionalOnClass để yêu cầu cả DataSource và EmbeddedDatabaseType đều có sẵn trên classpath. Nếu chúng không có sẵn, thì điều kiện sai và bất kỳ cấu hình nào được cung cấp bởi DataSourceAutoConfiguration sẽ bị bỏ qua.[5]

Trong DataSourceAutoConfiguration có một lớp JdbcTemplateConfiguration lồng nhau cung cấp cấu hình tự động của một JdbcTemplate bean:

|  |
| --- |
| Example.java |
| @Configuration  @Conditional(DataSourceAutoConfiguration.DataSourceAvailableCondition.class)  protected static class JdbcTemplateConfiguration {  @Autowired(required = false)  private DataSource dataSource;  @Bean  @ConditionalOnMissingBean(JdbcOperations.class)  public JdbcTemplate jdbcTemplate() {  return new JdbcTemplate(this.dataSource);  }  ...  } |

JdbcTemplateConfiguration là một annotation với low-level @Conditional để yêu cầu rằng DataSourceAvailableCondition pass — về cơ bản yêu cầu DataSource bean sẵn có hoặc một nó sẽ được tạo ra bởi việc cấu hình tự động. Giả sử rằng một DataSource bean sẽ có sẵn, phương thức jdbcTemplate() của Bean đã định cấu hình một JdbcTemplate bean. Nhưng jdbcTemplate() được chú thích bằng.

@ConditionalOnMissingBean sao cho bean sẽ được cấu hình chỉ khi chưa có một bean kiểu JdbcOperations (interface mà JdbcTemplate implement). [5]

**Một số ví dụ về điều kiện trong cấu hình tự động:**

■ Vì H2 có trong classpath, một embedded H2 database bean sẽ được tạo. Bean này thuộc loại javax.sql.DataSource, bean này sẽ được JPA implement (cụ thể là Hibernate) sử dụng để truy cập database. [5]

■ Vì Hibernate Enity Manager có trên classpath (thông qua Spring Data JPA), auto-configuration sẽ cấu hình các beans cần cho việc hỗ trợ Hibernate, bao gồm

LocalContainerEntityManagerFactoryBean và JpaVendorAdapter của Spring. [5]

■ Vì Spring Data JPA có trên classpath, Spring Data JPA sẽ được cấu hình cho việc tự động tạo các repository implementations từ các repository interfaces. [5]

■ Vì Thymeleaf có trên classpath, Thymeleaf sẽ được cấu hình như một view option cho Spring MVC, bao gồm một Thymeleaf template resolver, template engine và view resolver. Template resolver được cấu hình để resolve các template từ địa chỉ /templates. [5]

■ Vì Spring MVC có trên classpath, DispatcherServlet của Spring sẽ được cấu hình và Spring MVC sẽ được enabled. [5]

■ Vì đây là Spring MVC web application, một resource handler sẽ được đăng ký để xử lý các static content từ /static. Resource handler cũng sẽ xử lý static content từ /public, /resources và /META-INF/resources. [5]

■ Vì Tomcat có trên classpath, một embedded Tomcat container sẽ được khởi chạy để lắng nghe request từ cổng 8080. [5]

Vấn đề chính ở đây là, Spring Boot auto-configuration chịu trách nhiệm cấu hình Spring để bạn có thể tập trung vào việc viết ứng dụng. [5]

### Phát triển ứng dụng web với spring boot

#### 1.1.4.1. Luồng đi trong Spring Boot

##### Hai mô hình quen thuộc trong spring boot

Cấu trúc source code của Spring Boot được dựa trên hai mô hình là **mô hình MVC** và **mô hình 3 lớp**.

**Mô hình ba lớp (three tier)**

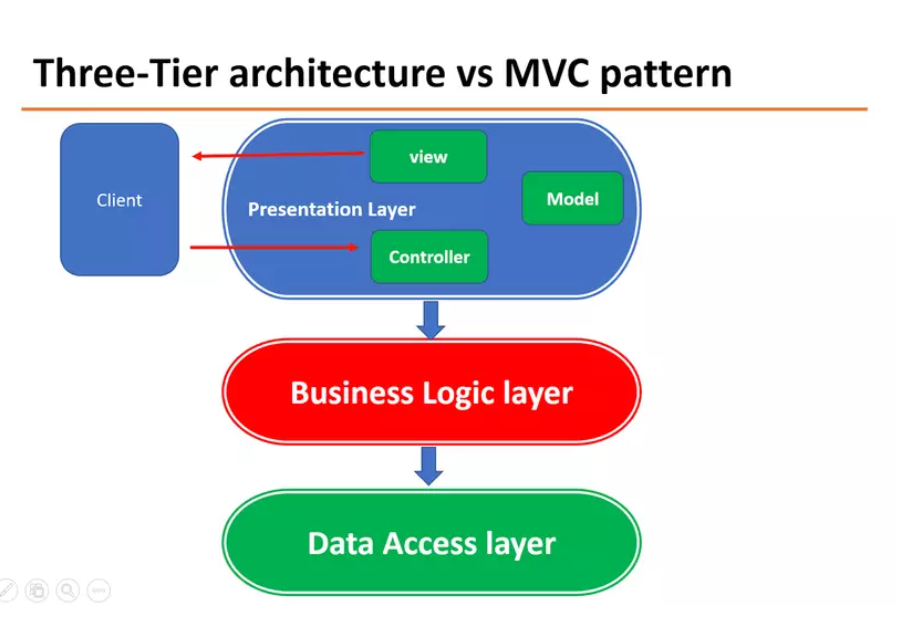
Đây là mô hình tổ chức source code rất phổ biến trong Spring Boot. Cụ thể, ứng dụng được chia làm 3 tầng (tier hoặc layer) như sau:

* **Presentation layer:** tầng này tương tác với người dùng, bằng View, Controller (trong MVC) hoặc API (nếu có).[6]
* **Business logic layer:** Chứa toàn bộ logic của chương trình, các đa số code nằm ở đây[6]
* **Data access layer:** Tương tác với database, trả về kết quả cho tầng business logic[6]

Trong Spring Boot, thì có một số thành phần đại diện cho từng lớp:

* **Service:** Chứa các business logic code[6]
* **Repository:** Đại diện cho tầng data access[6]

Còn presentation layer thì chúng ta sẽ bàn tiếp qua mô hình MVC ngay bên dưới.



**Mô hình MVC**

Do Spring Boot chỉ là wrapper cho Spring, chúng ta vẫn sử dụng ngầm các module Spring khác bên dưới, ví dụ như Spring MVC. Và tất nhiên khi dùng Spring MVC thì sẽ tuân theo mô hình MVC. [6]

Mô hình MVC chia tầng presentation làm 3 phần:

* **Model:** các cấu trúc dữ liệu của toàn chương trình, có thể đại diện cho trạng thái của ứng dụng[6]
* **View**: lớp giao diện, dùng để hiển thị dữ liệu ra cho user xem và tương tác[6]
* **Controller:** kết nối giữa Model và View, điều khiển dòng dữ liệu[6]

Dữ liệu từ Model qua Controller sau đó được gửi cho View hiển thị ra. Và ngược lại, khi có yêu cầu mới từ View, thì sẽ qua Controller thực hiện thay đổi dữ liệu của Model.

Tuy nhiên, MVC chỉ mô tả luồng đi của dữ liệu, nó không nói rõ như code đặt ở đâu (ở Model, View hay Controller), rồi lưu trữ Model vào database như thế nào,... Do đó, đối với ứng dụng hoàn chỉnh như Spring Boot thì cần kết hợp cả mô hình MVC và 3-tier lại với nhau.

##### Áp dụng vào Spring Boot

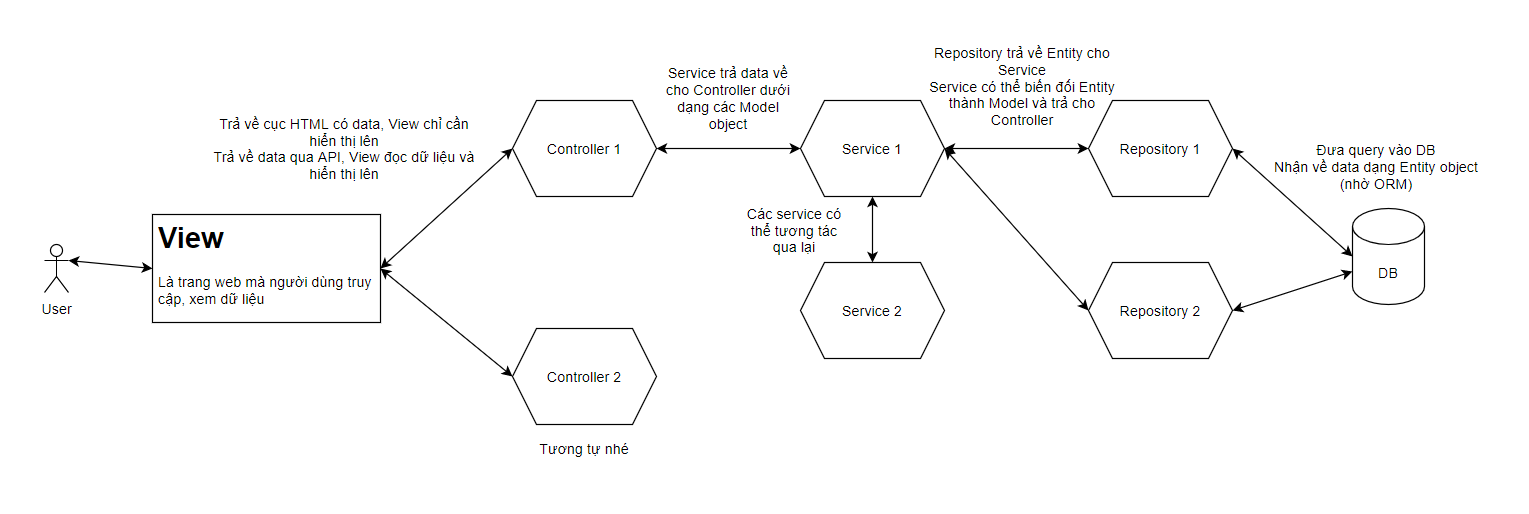
Kết hợp hai mô hình lại, chúng ta có được ứng dụng Spring Boot hoàn chỉnh, gồm các thành phần sau:

* Controller: trả về View (có chứa data sẵn, dạng trang HTML), hoặc Model thể hiện dưới dạng API cho View (View viết riêng bằng React, Vue, hoặc Angular).[6]
* Service: chứa các code tính toán, xử lý. Khi Controller yêu cầu, thì Service tương ứng sẽ tiếp nhận và cho ra dữ liệu trả cho Controller (trả về Model). Controller sẽ gửi về View như trên.[6]
* Repository: Service còn có thể tương tác với service khác, hoặc dùng Repository để gọi DB. Repository là thằng trực tiếp tương tác, đọc ghi dữ liệu trong DB và trả cho service.[6]

**Model và View**

* Model chỉ đơn giản là các đối tượng được Service tính toán xong trả về cho Controller.
* View có 2 loại, một là dạng truyền thống là trả về 1 cục HTML có data. Lúc này Controller sẽ pass dữ liệu vào View và return về (Spring MVC có thể sử dụng JSP hoặc template engine như Thymeleaf làm điều đó).View dạng 2 là dạng View tách riêng (không thuộc về project Spring boot). Thường có trong các hệ thống dùng API. View sẽ được viết riêng bằng React, Angular,... Controller sẽ đưa dữ liệu Model thông qua API cho View, và cũng nhận lại các yêu cầu qua API. [6]

Sơ đồ luồng đi



**Hình 1. 4 Sơ đồ luồng đi [6]**

Để hiểu rõ hơn về các thành phần trong Spring Boot, chúng ta hãy xem qua sơ đồ luồng đi và sự tương tác giữa chúng. Phần này có thể hơi khó hiểu tí, nhưng bạn cần nắm kĩ.

Sơ đồ trên chúng ta sẽ xét theo chiều kim đồng hồ:

* Đầu tiên, user sẽ vào View để xem, tương tác
* Khi user bắt đầu load dữ liệu (ví dụ click nút Reload), thì 1 request từ View gửi cho Controller (kiểu như "ê, cho tao cái danh sách user với")[6]
* Controller nhận được yêu cầu, bắt đầu đi hỏi ông Service (trong code là gọi method của Service) [6]
* Service nhận được yêu cầu từ Controller, đối với các code đơn giản có thể tính toán và trả về luôn. Nhưng các thao tác cần đụng tới database thì Service phải gọi Repository để lấy dữ liệu trong DB[6]
* Repository nhận được yêu cầu từ Service, sẽ thao tác với DB. Data lấy ra trong DB được hệ thống ORM (như JPA hoặc Hibernate) mapping thành các object (trong Java). Các object này gọi là Entity. [6]

Và bây giờ sẽ là đi ngược lại trả về cho user:

* Service nhận các Entity được Repository trả về, biến đổi nó. Biến đổi ở đây là có thể thực hiện tính toán, thêm bớt các field,... và cuối cùng biến Entity thành Model. Model sẽ được trả lại cho Controller. [6]
* Controller nhận được Model, nó sẽ return cho View. Có 2 cách, một là dùng template engine pass dữ liệu Model vào trang HTML, rồi trả về cục HTML (đã có data) cho client. Cách 2 là gửi qua API, View tự parse ra và hiển thị tương ứng (hiển thị thế nào tùy View). [6]
* Khi View hiển thị xong, user sẽ thấy danh sách user hiện lên trang web. [6]

Một số mẹo hay để tổ chức luồng đi cho tốt:

* Giữ cho Controller càng ít code càng tốt. Vì Controller chỉ là trung gian kết nối thông, nên không nên chứa nhiều code, thay vào đó nên bỏ vào Service.
* Nên tách bạch Service rõ ràng. Không nên cho 1 service thực hiện nhiều công việc, nên tách ra nhiều Service.

#### Dependency injection áp dụng vào Spring Boot

Spring là một framework được xây dựng dựa trên nguyên lý Dependency injection. Bản thân Spring có chứa IoC container, có nhiệm vụ tạo và quản lý các module:

* IoC container của Spring gọi là **Application context[7]**
* Các module chứa trong IoC container được Spring gọi là các **Bean [7]**

Spring Boot sử dụng các **annotation** dạng như @Component để đánh dấu lên class, chỉ ra rằng class đó cần tạo một module. Ngoài @Component, còn có các annotation khác như @Repository, @Controlller, @Service,... cũng được đánh dấu là module.

Khi ứng dụng Spring Boot chạy, thì IoC container sẽ thực hiện quá trình như sau:

* Quét tìm (scan) các class được đánh dấu là Bean, và tạo một object singleton, bỏ vào IoC container**[7]**
* Khi có một Bean phụ thuộc vào Bean khác, thì IoC sẽ tìm trong container, nếu chưa có thì tạo, nếu đã có thì lấy ra và inject vào bean cần nó**[7]**

#### Bean và ApplicationContext trong Spring Boot

##### 1.1.4.3.1. Bean và ApplicationContext

**Bean**

Trong documentation của Spring framework, thì **bean** được định nghĩa như sau:

“In Spring, the objects that form the backbone of your application and that are managed by the Spring IoC container are called beans. A bean is an object that is instantiated, assembled, and otherwise managed by a Spring IoC container”. **[7]**

Nói một cách đơn giản, bean là những module chính của chương trình, được tạo ra và quản lý bởi Spring IoC container. **[7]**

Các bean có thể phụ thuộc lẫn nhau, như ví dụ về Car, Engine và ChinaEngine từ đầu series tới giờ. Sự phụ thuộc này được mô tả cho IoC biết nhờ cơ chế Dependency injection. **[7]**

**ApplicationContext**

**ApplicationContext** là khái niệm Spring Boot dùng để chỉ Spring IoC container, tương tự như bean là đại diện cho các dependency. **[7]**

Ngoài ra bạn có thể sẽ nghe nói về BeanFactory. Nó cũng đại loại như ApplicationContext, đại diện cho Spring IoC container nhưng ở mức cơ bản. ApplicationContext thì ở mức cao hơn, cung cấp nhiều tính năng hơn BeanFactory như i18n, resolving messages, publishing events, ... **[7]**

Khi ứng dụng Spring chạy, Spring IoC container sẽ quét toàn bộ packages, tìm ra các bean và đưa vào ApplicationContext. Cơ chế đó là Component scan. **[7]**

**Cách lấy bean ra từ Context**

Tất nhiên trước khi lấy bean ra từ context thì phải có context rồi. Câu hỏi đặt ra là biến context ở đâu?

Đó là ngay dòng bắt đầu chương trình Spring Boot. Câu lệnh sau.

|  |
| --- |
| Example.java |
| @SpringBootApplication  public class Application {  public static void main(String[] args) {  SpringApplication.run(Application.class, args);  }  } |

Dòng method SpringApplication.run() sẽ return về một object ApplicationContext interface, đại diện cho IoC container. **[7]**

|  |
| --- |
| Example.java |
| ApplicationContext context = SpringApplication.run(Application.class, args); |

Chúng ta có thể lấy ra bean từ đây, dùng method getBean().**[7]**

|  |
| --- |
| Example.java |
| ApplicationContext context = SpringApplication.run(Application.class, args); // Lấy ra bean có class cụ thể  Car car = context.getBean(Car.class);  // Lấy ra theo tên và class  // Tuy là Engine.class nhưng Engine lại là interface  Engine engine = context.getBean("ChinaEngine", Engine.class); |

##### 1.1.4.3.2. Kĩ thuật inject bean vào bean khác

Ví dụ chúng ta có hai bean là Car và Engine (như ví dụ từ đầu series tới giờ). Và Car thì phụ thuộc vào Engine, do đó theo Dependency injection thì chúng ta cần inject Engine vào trong Car.

* **Sử dụng @Autowired**

Chúng ta sử dụng annotation @Autowired để báo cho Spring biết tự động tìm và inject bean phù hợp vào vị trí đặt annotation. Ví dụ.

|  |
| --- |
| Example.java |
| // Annotation chỉ đánh dấu lên class  public interface Engine {  void run();  }  @Component  public class ChinaEngine implements Engine {  @Override  public void run() {}  }  @Component  public class Car {  // Báo cho Spring tìm bean nào phù hợp với Engine interface  // Và có một bean phù hợp là ChinaEngine  // Nó tương đương với = new ChinaEngine()  @Autowired  private final Engine engine;  } |

Cách dùng @Autowired trên field là không được khuyến khích, do nó sử dụng Java reflection để inject. Chúng ta nên cân nhắc đổi qua dùng inject theo kiểu constructor hoặc setter.

* **Inject qua constructor hoặc setter**

Code inject theo kiểu constructor-based nên dùng khi các module là bắt buộc. Khi đó Spring Boot khi tạo bean (tạo object, gọi constructor) thì sẽ đưa các phụ thuộc vào constructor khi gọi. **[7]**

Ví dụ class Car đã được sửa lại để inject Engine vào qua constructor.

|  |
| --- |
| Example.java |
| @Component  public class Car {  private final Engine engine;    // Thêm @Required để setter luôn được gọi để inject  @Required  public void setEngine(Engine engine) {  this.engine = engine;  }  } |

Hoặc dùng kiểu setter-based như sau. Spring Boot sau khi tạo xong bean Car sẽ gọi thêm method setEngine() sau đó.

Cách dùng setter để inject thường dùng trong trường hợp phụ thuộc vòng, module A phụ thuộc vào B và ngược lại. Do đó, nếu cả hai đều sử dụng constructor based injection thì Spring Boot sẽ không biết nên tạo bean nào trước. Vì thế, giải pháp là một bean sẽ dùng constructor, một bean dùng setter như trên. **[7]**

#### Cấu trúc dự án Spring Boot

##### 1.1.4.4.1. Cấu trúc ứng dụng

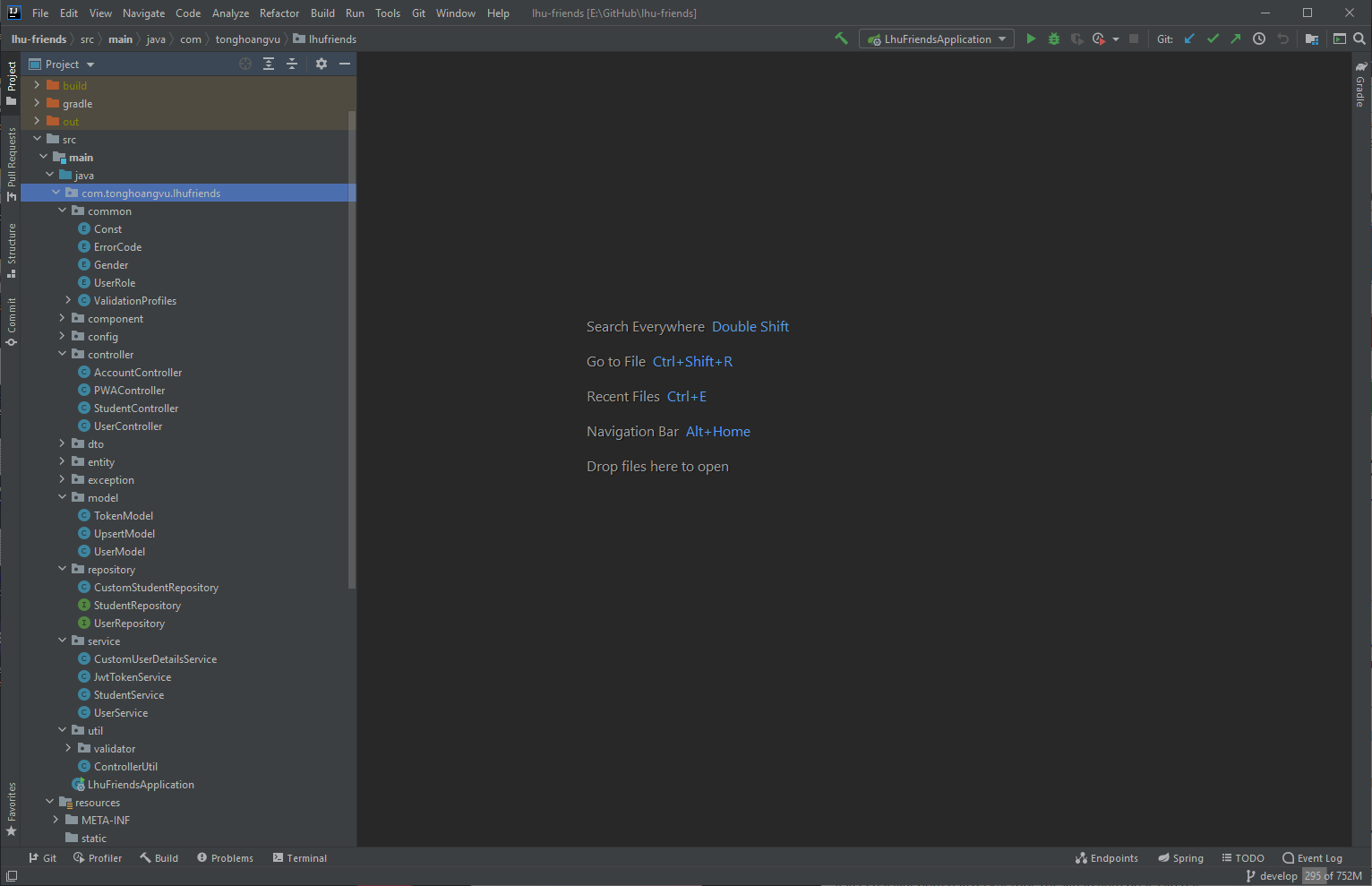
**Cấu trúc chung của ứng dụng**

Dù cho project được tạo với Maven hay Gradle thì cấu trúc chung vẫn tương tự nhau, do tuân theo một template có sẵn (tên là Archetype):[8]

* Thư mục gốc chứa các file như pom.xml (của Maven), build.gradle và các file khác như .gitignore,... dùng để cấu hình dự án. [8]
* Thư mục .mvn hoặc .gradle là thư mục riêng của Maven và Gradle, đừng nên đụng tới hay exclude nó ra khỏi source code. [8]
* Code được chứa trong thư mục src.
* Thư mục build ra chứa các file class, file JAR. Với Maven là target còn Gradle là build. [8]

Như đã nói ở trên thì source code chứa trong thư mục src. Chúng ta sẽ đi sâu hơn trong phần tiếp theo.

**Chi tiết cấu trúc source code**



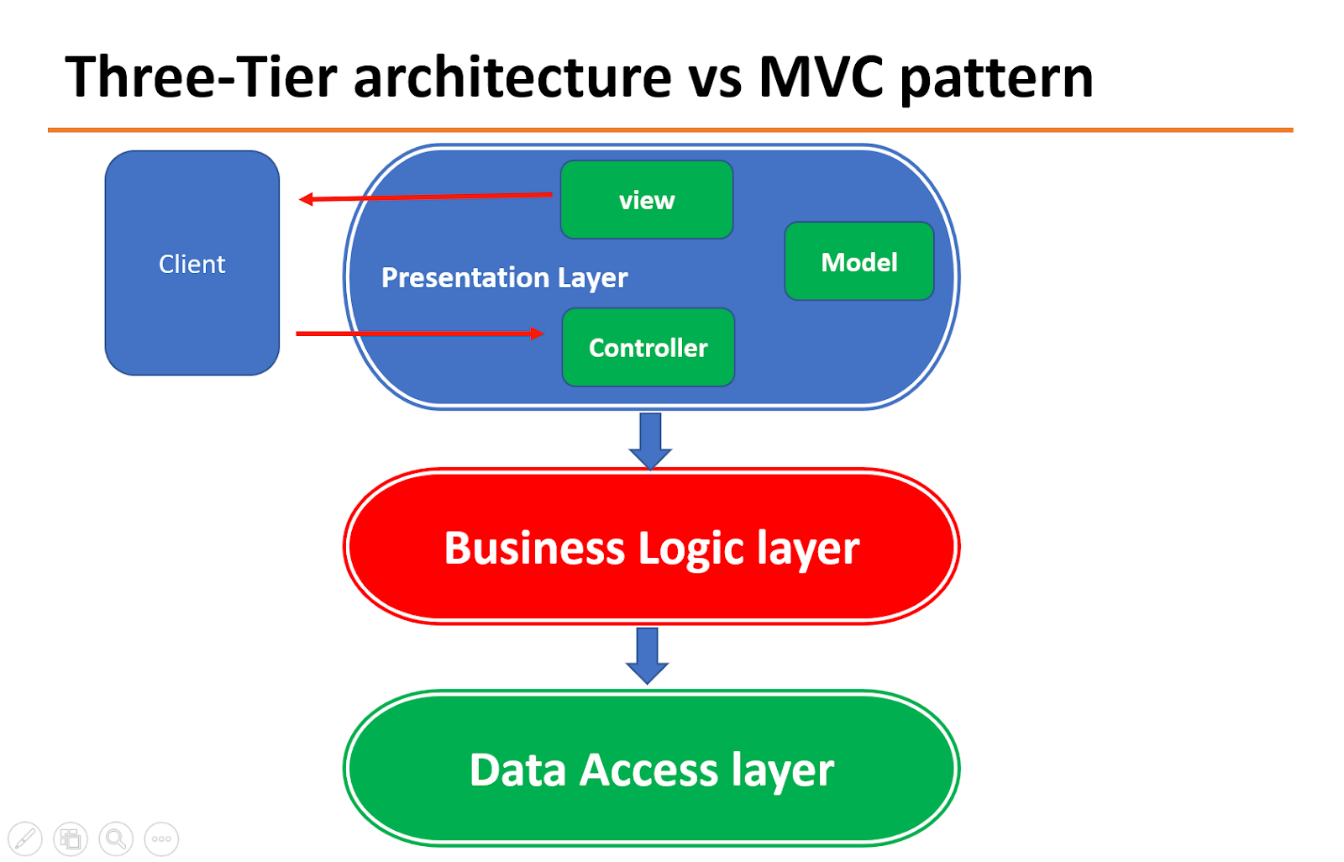
**Hình 1. 5 Chi tiết cấu trúc source code**

Như hình, thư mục chính chúng ta cần quan tâm là src/main/java/<tên package>. Mọi code java đều nằm trong này:

* Tên package chính được đặt dạng ngược với tên miền. Ví dụ như tonghoangvu.com thì đặt thành com.tonghoangvu. Cộng thêm tên project nữa.
* Có các package con, mỗi package đại diện cho các class thuộc layer cụ thể (ví dụ như service, controller,...)
* Thư mục resources chứa các tài nguyên của ứng dụng như hình ảnh, static file, properties file,...

Ngoài ra còn có src/test dùng để chứa các test class, dùng cho unit test.

##### 1.1.4.4.2. Tổ chức source code theo mô hình kết hợp mô hình 3 lớp và mô hình mvc



**Hình 1. 6 Tổ chức source code theo mô hình 3 lớp [8]**

Tương ứng với từng thành phần của mô hình 3 lớp, thì chúng ta sẽ có các thư mục và cách đặt tên tương ứng:

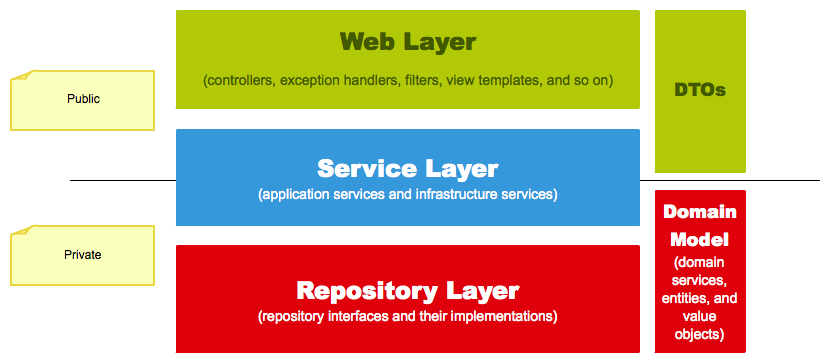
* Controller layer: đặt trong controller, các class là controller sẽ có hậu tố Controller (ví dụ UserController, AuthController,...)
* Service layer: đặt trong service, các class có hậu tố là Service và thường tương ứng với controller (ví dụ UserService,...)
* Data access layer: layer này bao gồm repository (đặt trong repository và hậu tố tương tự), DTO, model, entity...

Ngoài ra, với các loại khác thì:

* util package chứa các lớp util (xử lý các tác vụ khác), ví dụ như convert end date, tính toán đơn giản,...
* common package chứa các class định nghĩa như enum, interface, class dùng chung và đơn giản
* exception package chứa các class có nhiệm vụ xử lý exception trong Spring Boot.
* component chứa các bean được định nghĩa còn lại nhưng không thuộc layer nào.

Xong, hầu hết ứng dụng Spring Boot có tới 80% cấu trúc tương tự. Khác biệt là rất nhỏ, chỉ đơn giản nằm trong cách đặt tên.

Sơ đồ trên dùng để tổ chức source code trong chương trình. Nhờ đó chúng ta chia thành các Controller, Service, Repository tương ứng với các layer. Tuy nhiên, nếu xét về mặt tổ chức data, thì sơ đồ sẽ trở thành như sau.



**Hình 1. 7 Tổ chức source code theo mô hình kết hợp**

Mô hình này cũng gồm có 3 lớp, trong đó tên các layer được đổi thành các thành phần tương ứng trong Spring Boot.

Theo đó, tương ứng với từng layer thì data sẽ có dạng khác nhau. Nói cách khác, mỗi layer chỉ nên xử lý một số loại data nhất định. Mỗi dạng data sẽ có nhiệm vụ, mục đích khác nhau.

##### 1.1.4.4.3. Tổ chức code theo tính năng

Cách này có hơi khác với cách tổ chức theo mô hình 3 lớp ở trên. Cụ thể thay vì chia thành các package dựa theo layer, thì cách này chia theo tính năng. Nghĩa là mỗi tính năng, ví dụ user package thì sẽ chứa nào là UserController, UserService,...[8]

#### 1.1.4.5. Xử lý request trong Controller

##### 1.1.4.5.1. Controller

Controller trong ứng dụng Spring Boot là nơi tiếp nhận request và trả về response cho client. Có thể hiểu controller chính là lớp trung gian giữa server của bạn và bên ngoài.[9]

Về mặt code, Controller chỉ đơn thuần là một bean được đánh dấu với @Controller hoặc @RestController.

Trong Spring Boot, có hai dạng Controller, tương ứng hai annotation trên:

* @Controller có thể trả về View qua một String hoặc JSON data trong response body (nếu được chỉ định). Thích hợp cho các controller có routing, chuyển trang các kiểu. [9]
* @RestController chỉ có thể trả về data trong response body. Thích hợp cho các controller để cung cấp API. [9]

Do đó, ta có thể nói @RestController = @Controller + @ResponseBody.

**Các hoạt động của controller**

Như hình trên, khi client gửi một request tới server Spring Boot, thì nó sẽ đi qua thứ gọi là Front controller trước. Đây là controller có sẵn, nó có tác dụng sau:

* Phân giải request, tìm coi request gọi tới method nào của controller nào để gọi đúng tới đó.[9]
* Các data của request sẽ được parse ra và mapping tương ứng vào các tham số controller method (có @RequestParam, @PathVariable, @Header,... tương ứng). [9]
* Đặc biệt, Spring MVC có thể parse được các data phức tạp như enum, List hay object. Ví dụ enum trong request là dạng string, vẫn sẽ được parse đúng thành enum. [9]
* Nếu data không thể parse được, front controller sẽ trả về bad request (hoặc có cơ chế khác để chúng ta ghi đè lại việc này). [9]

Với chiều ngược lại cũng tương tự như vậy. Dữ liệu trả về từ controller sẽ được build thành response và trả cho client.

##### 1.1.4.5.2. Controller mapping

**Các loại HTTP request**

HTTP request sẽ gồm 2 thông tin quan trọng:

* Request tới URL nào (request tới đâu) [9]
* HTTP method là gì (thể hiện hành động gì đấy với URL) [9]

Trong controller, chỉ cần nắm được hai thông tin trên thì sẽ bắt được mọi request được gửi tới, sau đó mới xử lý tiếp.

Trong Rest API design, thì người ta thường dùng **danh từ** trong URL để chỉ đối tượng được tác động. Còn các HTTP method để đại diện cho **hành động** nào sẽ áp dụng lên đối tượng đó. [9]

Ví dụ như:

1. Request tới GET /users có đối tượng tác động là users (tất cả user), và hành động là GET (lấy thông tin)
2. Request tới PUT /users/123 có đối tượng là users/123 (user có mã là 123) và hành động là PUT (cập nhật thông tin)

Thường thì theo khuyến nghị người ta sử dụng đúng HTTP method với các hành động CRUD tương ứng:

* Create: dùng POST method
* Read: dùng GET method
* Update: dùng PUT method
* Delete: dùng DELETE method

Hầu hết các ứng dụng web đều sử dụng 4 hành động CRUD cơ bản trên tới hơn 2/3 rồi. Ngoài ra có thể có các hành động khác mà không có method tương ứng, như login thì có thể thêm vào endpoint như POST /login (dùng POST sẽ an toàn hơn, đọc thêm về các HTTP method để hiểu rõ hơn ý nghĩa của chúng). [9]

**Bắt các request**

Spring Boot dùng các annotation sau, đánh dấu lên từng **method** của controller, để chỉ định rằng khi HTTP method tương ứng gọi tới thì method sẽ được thực thi. [9]

|  |
| --- |
| Example.java |
| @RestController  public class UserController {  @GetMapping("/users")  public ResponseEntity<?> getAllUsers() {}    @DeleteMapping("/users/{id}")  public void deleteUser(@PathVariable("id") int id) {}  } |

Ví dụ trên có 2 method, bắt tương ứng hai request là GET /users và DELETE /users/{id}. Khi có request tương ứng gửi tới, thì hai method trên sẽ thực thi và trả về kết quả cho client.

Các annotation phổ biến như @GetMapping, @PostMapping, @PutMapping,... có dạng là tên HTTP method cộng với từ "mapping". Ngoài ra còn có thể dùng @RequestMapping và chỉ định thuộc tính method như sau. [9]

|  |
| --- |
| Example.java |
| @RequestMapping(value = "/users", method = RequestMethod.GET) |

Ngoài ra, @RequestMapping còn có thể dùng bên trên class controller, để chỉ định endpoint gốc cho toàn bộ method bên trong nó. [9]

Ví dụ như sau.

|  |
| --- |
| Example.java |
| @RestController  @RequestMapping("/users")  public class UserController {  // Kết hợp với route gốc ở trên, ta có /users/info  @GetMapping("/info")  } |

##### 1.1.4.5.3. Nhận request data trong Controller

Controller nhận dữ liệu từ request, tùy vào dữ liệu nằm ở đâu mà chúng ta có cách lấy ra khác nhau:

* Request param (query string) [9]
* Path variable[9]
* Request body[9]
* Header[9]

**Request param (query string)**

Ví dụ như request sau GET /users?age=18&name=Dũng thì chúng ta có 2 request param là age = 18 và name = Dũng. Khi đó, muốn lấy được hai giá trị trên chúng ta dùng @RequestParam như sau.

|  |
| --- |
| Example.java |
| @RestController  public class UserController {  ...  @GetMapping("/users")  public ResponseEntity<?> getAllUsers(  @RequestParam("age") int age,  @RequestParam("name") String name) {  // Lúc này hai biến age và name đã có dữ liệu tương ứng  }  } |

Trong trường hợp @RequestParam có thêm các tham số khác, thì chúng ta phải viết như sau. Ví dụ cả hai trường age và name trên đều là optional, không bắt buộc, nên chúng ta dùng thuộc tính required = false cho @RequestParam (mặc định là true).

|  |
| --- |
| Example.java |
| @RestController  public class UserController {  ...  @GetMapping("/users")  public ResponseEntity<?> getAllUsers(  @RequestParam(value = "age", required = false) Integer age,  @RequestParam(value = "name", required = false) String name) {  // Lúc này hai biến age và name đã có dữ liệu tương ứng  }  } |

Lúc này, do biến age có thể không có, nên phải cho nó kiểu Integer có giá trị null để biết được age có được gửi lên hay không. Nếu là kiểu primitive thì nó sẽ luôn có giá trị mặc định.

Ngoài ra @RequestParam cũng có thuộc tính defaultValue, nếu request không chỉ định thì giá trị default đó sẽ được sử dụng.

**Path variable**

Path variable là một phần trong đường dẫn URL, ví dụ GET /users/123/info thì 123 là path variable. Sử dụng @PathVariable để làm việc này, tương tự như dùng @RequestParam.

|  |
| --- |
| Example.java |
| @GetMapping("/users/{id}/info")  public ResponseEntity<?> getUserInfo(  @PathVariable(value = "id", defaultValue = "0") int userId) {  // id là tên path variable, tương ứng trên url {id}  } |

@PathVariable cũng có các thuộc tính tương tự @RequestParam.

**Request body**

Request method PUT, POST mới có request body, đây là nơi chứa data chính để gửi lên. Thường thì request body sẽ ở dạng JSON hoặc form-data, khi vào controller sẽ được tự động parse ra thành Object (ví dụ DTO). [9]

|  |
| --- |
| Example.java |
| @PostMapping("/login")  public ResponseEntity<?> login(@RequestBody LoginDto loginDto) {  // Dữ liệu trong request body có thể là JSON, form-data,...  // Tuy nhiên khi vào controller sẽ bị parse thành object hết  } |

Đây là ví dụ về class LoginDto ở trên (có dùng lombok).

**Header**

|  |
| --- |
| Example.java |
| @PostMapping("/login")  public ResponseEntity<?> login(@Header("Authorization") String authHeader) {  // Biến authHeader sẽ có giá trị là giá trị của Authorization header  } |

Ví dụ như trên chúng ta thực hiện xác thực người dùng bằng Basic authentication. Thông tin username, password được encode trong header có tên là Authorization. Muốn lấy được value trong header, thì chúng ta sử dụng @Header như trên.

* + 1. **Phát triển Web Service trong Spring Boot**

### Spring Boot JPA

**Spring Boot JPA** là một phần trong hệ sinh thái Spring Data, nó tạo ra một layer ở giữa tầng service và database, giúp chúng ta thao tác với database một cách dễ dàng hơn, tự động config và giảm thiểu code thừa thãi.[10]

**Spring Boot JPA** đã wrapper Hibernate và tạo ra một interface mạnh mẽ. Nếu như gặp khó khăn khi làm việc với Hibernate, **Spring JPA** sẽ giúp chúng ta làm.

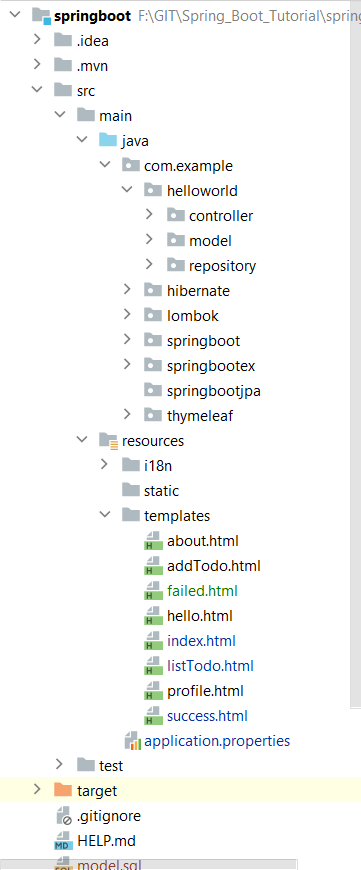
**Cài đặt**

Để thêm Spring JPA vào project, cần thêm dependency spring-boot-starter-data-jpa.

Ngoài ra, để connect tới MySql, chúng ta cần driver tương ứng, vì vậy phải bổ sung thêm cả dependency mysql-connector-java vào pom.xml.

|  |
| --- |
| Porm.xml |
| <?xml version="1.0" encoding="UTF-8"?>  <project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"  xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">  <modelVersion>4.0.0</modelVersion>  <packaging>pom</packaging>  <parent>  <groupId>org.springframework.boot</groupId>  <artifactId>spring-boot-starter-parent</artifactId>  <version>2.0.5.RELEASE</version>  <relativePath /> <!-- lookup parent from repository -->  </parent>  <groupId>me.loda.spring</groupId>  <artifactId>spring-boot-learning</artifactId>  <version>0.0.1-SNAPSHOT</version>  <name>spring-boot-learning</name>  <description>Everything about Spring Boot</description>  <properties>  <java.version>1.8</java.version>  </properties>  <dependencies>  <!--spring mvc, rest-->  <dependency>  <groupId>org.springframework.boot</groupId>  <artifactId>spring-boot-starter-web</artifactId>  </dependency>  <!--spring jpa-->  <dependency>  <groupId>org.springframework.boot</groupId>  <artifactId>spring-boot-starter-data-jpa</artifactId>  </dependency>  <!-- mysql connector -->  <dependency>  <groupId>mysql</groupId>  <artifactId>mysql-connector-java</artifactId>  </dependency>  </dependencies>  <build>  <plugins>  <plugin>  <groupId>org.springframework.boot</groupId>  <artifactId>spring-boot-maven-plugin</artifactId>  </plugin>  </plugins>  </build>  </project> |

Cấu trúc thư mục:



**Hình 1. 8 Cấu trúc thư mục project spring boot**

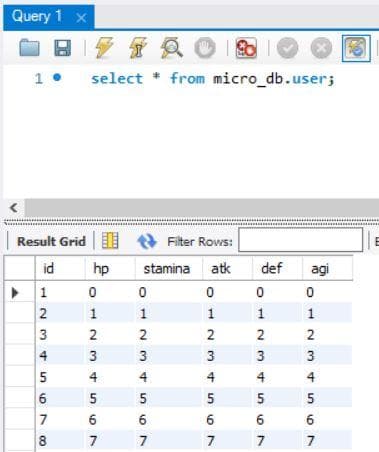
**Tạo Table và dữ liệu**

Trước khi bắt đầu, chúng ta cần tạo ra dữ liệu trong Database. Ở đây chúng em chọn MySQL.

Dưới đây là SQL Script để tạo DATABASE micro\_db. Chứa một TABLE duy nhất là User.Khi chạy script này, nó sẽ tự động insert vào db 100 User.

|  |
| --- |
| Scipt.sql |
| CREATE DATABASE micro\_db;  use micro\_db;  CREATE TABLE `user`  (  `id` bigint(20) NOT NULL AUTO\_INCREMENT,  `hp` int NULL DEFAULT NULL,  `stamina` int DEFAULT NULL,  `atk` int DEFAULT NULL,  `def` int DEFAULT NULL,  `agi` int DEFAULT NULL,  PRIMARY KEY (`id`)  );  DELIMITER $$  CREATE PROCEDURE generate\_data()  BEGIN  DECLARE i INT DEFAULT 0;  WHILE i < 100 DO  INSERT INTO `user` (`hp`,`stamina`,`atk`,`def`,`agi`) VALUES (i,i,i,i,i);  SET i = i + 1;  END WHILE;  END$$  DELIMITER ;  CALL generate\_data(); |

Sau khi chạy xong script trên, chúng ta kiểm tra database đã có dữ liệu chưa.



**Hình 1. 9 Dữ liệu**

**Tạo Model User**

Khi đã có dữ liệu trong Database. Chúng ta sẽ tạo một Class trong Java để mapping thông tin.

Phần này chúng ta cần có một chút kiến thức về Hibernate. Nếu bạn chưa biết những Annotation ở dưới đây để làm gì thì hãy tạm dừng và [tìm hiểu Hibernate tại đây](https://loda.me/hibernate-la-gi-loda1554623701594).

|  |
| --- |
| User.java |
| @Entity  @Table(name = "user")  @Data  public class User implements Serializable {  private static final long serialVersionUID = -297553281792804396L;  @Id  @GeneratedValue(strategy = GenerationType.IDENTITY)  private Long id;  // Mapping thông tin biến với tên cột trong Database  @Column(name = "hp")  private int hp;  @Column(name = "stamina")  private int stamina;  // Nếu không đánh dấu @Column thì sẽ mapping tự động theo tên biến  private int atk;  private int def;  private int agi;  } |

Tới đây là chúng ta làm được nửa đường rồi.

**Vấn đề của Hibernate truyền thống**

Thông thường, khi đã định nghĩa Entity tương ứng với Table trong DB thông qua Hibernate. Thì nhiệm vụ tiếp theo sẽ là tạo ra các class thao tác với DB.

Ví dụ muốn query lấy tất cả User bằng Hibernate truyền thống sẽ như sau:

|  |
| --- |
| User.java |
| // Giả sử đã có đối tượng session rồi  Session session = getSession();  try {  // Tất cả các lệnh hành động với DB thông qua Hibernate  // đều phải nằm trong 1 giao dịch (Transaction)  // Bắt đầu giao dịch  session.getTransaction().begin();  // Tạo một query  String sql = "Select u from " + User.class.getName() + " u ";  // Tạo đối tượng Query.  Query<User> query = session.createQuery(sql);  // Thực hiện truy vấn và lấy ra dữ liệu.  List<User> users = query.getResultList();  // In ra màn hình  for (User user : users) {  System.out.println(user);  }  // Commit dữ liệu và kết thúc session.  session.getTransaction().commit();  } catch (Exception e) {  e.printStackTrace();  // Rollback trong trường hợp có lỗi xẩy ra.  session.getTransaction().rollback();  } |

Mặc dù Hibernate đã làm rất tốt và giảm thiểu code cho việc thao tác với Database xuống rồi, những nó vẫn chưa hẳn là dễ dàng

Mục đích ban đầu của Hibernate là giúp người lập trình dễ sử dụng, tuy nhiên, trên thực tế, nhiều người gặp khó khăn trong việc sử dụng với Hibernate hơn cả jdbc.

Nắm được vấn đề này, **Spring Data** đã wrapper lên Hibernate một lớp nữa gọi là **Spring JPA**, giúp cho mọi thao tác với DB của chúng ta rút ngắn xuống còn một dòng và tất nhiên là làm mờ Hibernate xuống đáng kể để tránh rắc rối cho người lập trình.

**JpaRepository**

Để sử dụng **Spring JPA**, bạn cần sử dụng interface JpaRepository.

Yêu cầu của interface này đó là bạn phải cung cấp 2 thông tin:

1. Entity (Đối tượng tương ứng với Table trong DB)
2. Kiểu dữ liệu của khóa chính (primary key)

Ví dụ: Tôi muốn lấy thông tin của bảng User thì làm như sau:

|  |
| --- |
| test.java |
| import org.springframework.data.jpa.repository.JpaRepository;  import org.springframework.stereotype.Repository;  @Repository  public interface UserRepository extends JpaRepository<User, Long> {  } |

@Repository đánh dấu UserRepository là một Bean và chịu trách nhiệm giao tiếp với DB.

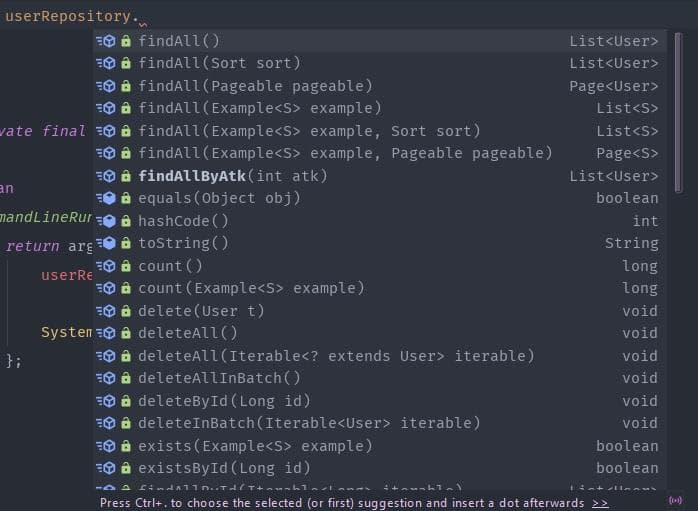
**Spring Boot** sẽ tự tìm thấy và khởi tạo ra đối tượng UserRepository trong Context. Việc tạo ra UserRepository hoàn toàn tự động và tự config, vì chúng ta đã kế thừa JpaRepository.[10]

Bây giờ, việc lấy ra toàn bộ User sẽ như sau:

|  |
| --- |
| test.java |
| @Autowired  UserRepository userRepository;  userRepository.findAll()  .forEach(System.out::println); |

Đơn giản và ngắn gọn hơn rất nhiều.

Nếu bạn tìm kiếm thì sẽ thấy UserRepository có hàng chục method mà chúng ta không cần viết lại nữa. Vì nó kế thừa JpaRepository rồi.



**Hình 1. 10 Các hàm hỗ trợ trong Spring boot JPA**

**Demo**

Bây giờ chúng ta sẽ làm ứng dụng Demo các tính năng cơ bản với JpaRepository

Bước đầu tiên là config thông tin về MySQL trong application.properties

|  |
| --- |
| User.java |
| application.properties  spring.datasource.url=jdbc:mysql://localhost:3306/micro\_db?useSSL=false  spring.datasource.username=root  spring.datasource.password=root  ## Hibernate Properties  # The SQL dialect makes Hibernate generate better SQL for the chosen database  spring.jpa.properties.hibernate.dialect = org.hibernate.dialect.MySQL5InnoDBDialect  # Hibernate ddl auto (create, create-drop, validate, update)  spring.jpa.hibernate.ddl-auto = update  logging.level.org.hibernate = ERROR |

**Spring JPA** sẽ tự kết nối cho chúng ta, mà không cần thêm một đoạn code nào cả.

|  |
| --- |
| User.java |
| User.java  @Entity  @Table(name = "user")  @Data  public class User implements Serializable {  private static final long serialVersionUID = -297553281792804396L;  @Id  @GeneratedValue(strategy = GenerationType.IDENTITY)  private Long id;  // Mapping thông tin biến với tên cột trong Database  @Column(name = "hp")  private int hp;  @Column(name = "stamina")  private int stamina;  // Nếu không đánh dấu @Column thì sẽ mapping tự động theo tên biến  private int atk;  private int def;  private int agi;  } |

|  |
| --- |
| App.java |
| import org.springframework.boot.SpringApplication;  import org.springframework.boot.autoconfigure.SpringBootApplication;  import org.springframework.context.ApplicationContext;  import lombok.RequiredArgsConstructor;  @SpringBootApplication  @RequiredArgsConstructor  public class App {  public static void main(String[] args) {  ApplicationContext context = SpringApplication.run(App.class, args);  UserRepository userRepository = context.getBean(UserRepository.class);  // Lấy ra toàn bộ user trong db  userRepository.findAll()  .forEach(System.out::println);  // Lưu user xuống database  User user = userRepository.save(new User());  // Khi lưu xong, nó trả về User đã lưu kèm theo Id.  System.out.println("User vừa lưu có ID: " + user.getId());  Long userId = user.getId();  // Cập nhật user.  user.setAgi(100);  // Update user  // Lưu ý, lúc này đối tượng user đã có Id.  // Nên nó sẽ update vào đối tượng có Id này  // chứ không insert một bản ghi mới  userRepository.save(user);  // Query lấy ra user vừa xong để kiểm tra xem.  User user2 = userRepository.findById(userId).get();  System.out.println("User: " + user);  System.out.println("User2: " + user2);  // Xóa User khỏi DB  userRepository.delete(user);  // In ra kiểm tra xem userId còn tồn tại trong DB không  User user3 = userRepository.findById(userId).orElse(null);  System.out.println("User3: " + user2);  }  } |

OUTPUT chương trình:

|  |
| --- |
| OutPut |
| User vừa lưu có ID: 104  // sau khi update, cả 2 đối tượng user đều có giá trị agi mới  User: User(id=104, hp=0, stamina=0, atk=0, def=0, agi=100)  User2: User(id=104, hp=0, stamina=0, atk=0, def=0, agi=100)  // Sau khi xóa, user không còn tồn tại  User3: null |

## SPRING SECURITY

### Giới thiệu

#### Spring Security

Spring Security là một module nổi bật trong hệ sinh thái Spring. Spring Security cung cấp các dịch vụ bảo mật toàn diện cho các ứng dụng doanh nghiệp có nền tảng Java EE.

Spring Security cung cấp 2 cơ chế cơ bản:

* **Authentication** (xác thực): là tiến trình thiết lập một principal. Principal có thể hiểu là một người, hoặc một thiết bị, hoặc một hệ thống nào đó có thể thực hiện một hành động trong ứng dụng của bạn.
* **Authorization** (phân quyền) hay **Access-control**: là tiến trình quyết định xem một principal có được phép thực hiện một hành động trong ứng dụng của bạn hay không. Trước khi diễn tiến tới Authorization, principal cần phải được thiết lập bởi Authentication.

Ta có thể thấy đây là 2 cơ chế khá phổ biến trong các dịch vụ bảo mật, không chỉ riêng Spring Security.

#### Ưu điểm

#### Công dụng của Spring Security trong đề tài nghiên cứu

### Các thành phần cốt lõi

**Security, SecurityContext và Authentication**

**SecurityContext** là interface cốt lõi của Spring Security, lưu trữ tất cả các chi tiết liên quan đến bảo mật trong ứng dụng. Khi chúng ta kích hoạt Spring Security trong ứng dụng thì SecurityContext cũng sẽ được kích hoạt theo.

Chúng ta sẽ không truy cập trực tiếp vào SecurityContext, thay vào đó sẽ sử dụng lớp **SecurityContextHolder**. Lớp này lưu trữ security context hiện tại của ứng dụng, bao gồm chi tiết của principal đang tương tác với ứng dụng. Spring Security sẽ dùng một đối tượng **Authentication** để biểu diễn thông tin này. Đoạn code dưới đây sẽ giúp chúng ta lấy được username của principal đã được xác thực (username ở đây ta nên hiểu là username trong cặp username - password mà người dùng nhập vào khi đăng nhập):

Object principal = SecurityContextHolder.getContext().getAuthentication().getPrincipal();

if (principal instanceof UserDetails) {

String username = ((UserDetails) principal).getUsername();

} else {

String username = principal.toString();

}

Đoạn code này có thể đặt ở bất kỳ đâu trong ứng dụng.

**UserDetails và UserDetailsService**

Trong đoạn code trên, chúng ta có được một principal từ đối tượng Authentication. Principal đơn giản chỉ là một đối tượng và sẽ được ép kiểu sang **UserDetails**.

UserDetails là một interface cốt lõi của Spring Security. Nó đại diện cho một principal nhưng theo một cách mở rộng và cụ thể hơn. Vậy UserDetails cung cấp cho ta những thông tin gì? UserDetails bao gồm các method sau:

* getAuthorities(): trả về danh sách các quyền của người dùng
* getPassword(): trả về password đã dùng trong qúa trình xác thực
* getUsername(): trả về username đã dùng trong qúa trình xác thực
* isAccountNonExpired(): trả về true nếu tài khoản của người dùng chưa hết hạn
* isAccountNonLocked(): trả về true nếu người dùng chưa bị khóa
* isCredentialsNonExpired(): trả về true nếu chứng thực (mật khẩu) của người dùng chưa hết hạn
* isEnabled(): trả về true nếu người dùng đã được kích hoạt

Chúng ta có thể thấy UserDetails mới chỉ cung cấp các phương thức để truy cập các thông tin cơ bản của người dùng. Để mở rộng thêm các thông tin, chúng ta sẽ tạo một lớp CustomUserDetails implements org.springframework.security.userdetails.

UserDetails (tên lớp là tùy ý, bạn đặt tên thế nào cũng được).

Câu hỏi tiếp theo đặt ra là ta sẽ tạo implementation của UserDetails ở đâu trong ứng dụng? Câu trả lời là ta sẽ dùng **UserDetailsService**. UserDetailsService là một interface có duy nhất một phương thức:

UserDetails **loadUserByUsername**(String username) throws UsernameNotFoundException;

Tham số truyền vào chỉ gồm có username của người dùng. Ta sẽ tìm kiếm trong CSDL, record thỏa mãn username. Nếu không tìm thấy, ta sẽ ném ra ngoại lệ UsernameNotFoundException.

Phương thức loadUserByUsername() sẽ trả về một implementation của UserDetails. Implementation ở đây có thể là:

* org.springframework.security.core.userdetails.User
* CustomUserDetails implements

org.springframework.security.userdetails.UserDetails mà mình đã nói ở trên

Nhiệm vụ của chúng ta là cần phải tạo một lớp UserDetailsServiceImpl implements UserDetailsService.

**GrantedAuthority**

Ở phần trên, mình đã đề cập đến phương thức getAuthorities(). Phưong thức này sẽ trả về một tập hợp các đối tượng **GrantedAuthority**. Một GrantedAuthority là một quyền được ban cho principal. Các quyền đều có tiền tố là **ROLE\_**, ví dụ như ROLE\_ADMIN, ROLE\_MEMBER ...

### Áp dụng Spring Security trong thực tế

Trong bài viết lần này, mình sẽ hướng dẫn các bạn xây dựng các chức năng sau:

* Đăng nhập
* Đăng xuất
* Phân quyền

Người dùng được chia là 3 nhóm:

* Khách (guest)
* Thành viên (member), có quyền ROLE\_MEMBER
* Quản trị viên (admin), có cả quyền ROLE\_MEMBER và ROLE\_ADMIN

Ứng dụng của chúng ta sẽ gồm có 4 trang:

* Trang đăng nhập /login
* Trang chủ /: chỉ cho phép ROLE\_MEMBER truy cập
* Trang admin /admin: chỉ cho ROLE\_ADMIN truy cập
* Trang 403 /403: nếu ROLE\_MEMBER vào trang admin, sẽ bị redirect về trang này

## AJAX

### Giới thiệu

#### AJAX.

AJAX là viết tắt của cụm từ Asynchronous JavaScript and XML (JavaScript và XML bất đồng bộ) là một kỹ thuật cho phép tải về nội dung của một hoặc nhiều trang mới mà không cần phải tải lại toàn bộ nội dung của trang hiện tại.[11]

#### Ưu điểm.

Hầu hết các website hiện nay đều sử dụng AJAX, nổi tiếng trong số đó có Facebook và Google.

Khi tìm kiếm trên Google và gõ một số chữ cái đầu tiên Google sẽ hiển thị một loạt các từ khoá liên quan để gợi ý. Các từ khoá gợi ý này là kết quả của việc sử dụng công nghệ AJAX để gửi yêu cầu tới máy chủ (với dữ liệu là các chữ cái mà người dùng đã nhập vào) để lấy về kết quả từ khoá gợi ý mà không cần phải tải lại trang.

Khi sử dụng trình duyệt để truy cập trang Facebook và cuộn xuống phía dưới thì nội dung của các bài viết cũ hơn sẽ được hiển thị mà không cần phải tải lại trang.

Các ưu điểm của AJAX:

* Thực hiện các cuộc gọi lại (call-back): Quy chế hoạt động của nó đơn giản là gửi một phần trang web đến server. Ajax giúp cải thiện hiệu suất mạng, tăng tốc website. [16]
* Thực hiện các cuộc gọi không đồng bộ đến một máy chủ. Như vậy, Ajax tiết kiệm được thời gian cho người dùng, đồng thời giảm tải thời gian load website. [16]
* Tạo ra sự đa dạng cho website, giúp website có thể load từng phần động mà không cồng kềnh. Ajax được sử dụng ở bất cứ nơi nào trên một trang web. [16]
* Tăng trải nghiệm cho người dùng khi tiết kiệm thời gian load thông tin, không phải post lại trang lên máy chủ. [16]
* Tiết kiệm băng thông và tăng hiệu suất của website, đồng thời dữ liệu cũng được lưu vào account. [16]

#### Công dụng của Ajax trong đề tài nghiên cứu.

Ajax được viết bằng Javascript chạy trên client, tức là mỗi browser sẽ chạy độc lập hoàn toàn không ảnh hưởng lẫn nhau, giúp chúng ta đem lại cho người dùng trải nghiệm tốt hơn. Khi cần một thay đổi nhỏ thì sẽ không cần load lại cả trang web, làm trang web phải tải lại nhiều thứ không cần thiết.

Từ những ưu điểm của AJAX, Nhóm đã sử dụng kỹ thuật này để làm phương thức trao đổi dữ liệu với máy chủ và cập nhật một hay nhiều phần của trang web, hoàn toàn không reload lại toàn bộ trang nhằm nâng cao trải nghiệm cho người dùng.

### Sử dụng Ajax trong xây dựng website

#### Sự Kiện trong Ajax

Khi gửi yêu cầu tới máy chủ sử dụng AJAX thì thông thường chúng ta sẽ muốn đăng ký một hay nhiều nhiệm vụ cần thực hiện nhận được kết quả gửi về từ máy chủ. Sự kiện máy chủ gửi về kết quả cho trình duyệt sẽ được theo dõi sử dụng thuộc tính onreadystatechange của đối tượng XMLHttpRequest.[11]

Trước tiên chúng ta cần tìm hiểu hai thuộc tính readyState và status của đối tượng XMLHttpRequest. [11]

**Thuộc Tính readyState**

Thuộc tính readyState của đối tượng XMLHttpRequest được sử dụng để đặc trưng cho **trạng thái** của yêu cầu (request) gửi tới máy chủ. Thuộc tính này có các giá trị sau: [11]

* 0: Yêu cầu chưa được gửi đi. [11]
* 1: Đã thiết lập kết nối với máy chủ thành công. [11]
* 2: Máy chủ đã nhận được yêu cầu gửi tới. [11]
* 3: Máy chủ đang xử lý yêu cầu gửi tới. [11]
* 4: Yêu cầu được xử lý thành công và máy chủ hoàn tất việc gửi về kết quả cho trình duyệt. [11]

**Thuộc Tính status**

Thuộc tính này được dùng để xác định trạng thái của kết quả gửi về trình duyệt từ server. Thuộc tính status có những giá trị sau:

* 200: Trang yêu cầu đã được tìm thấy và máy chủ trả về kết quả. [11]
* 404: Trang yêu cầu không tồn tại. [11]

**Thuộc Tính onreadystatechange**

Sử dụng thuộc tính onreadystatechange của đối tượng XMLHttpRequest chúng ta có thể **bắt** được sự kiện diễn ra khi thuộc tính readyState của đối tượng này thay đổi. Thuộc tính này nhận giá trị là một hàm và đoạn mã của hàm sẽ được thực thi khi giá trị của thuộc tính readyState thay đổi: [11]

|  |
| --- |
| Example.java |
| // cấu hình XMLHttpRequest  var xhttp = new XMLHttpRequest();  // đăng ký các nhiệm vụ sẽ được thực hiện sử dụng onreadystatechange  xhttp.onreadystatechange = function() {  // mã lệnh sẽ được thực hiện  } |

Ví dụ để bắt sự kiện máy chủ đã hoàn tất việc xử lý yêu cầu và gửi về kết quả cho trình duyệt chúng ta dùng đoạn mã sau:

|  |
| --- |
| Example.java |
| // cấu hình XMLHttpRequest  xhttp.open("GET", "/get-name.php?id=1", true);  xhttp.onreadystatechange = function() {  // mã lệnh sẽ được thực thi khi máy chủ gửi về kết quả (readyState = 4)  // và trang yêu cầu đã được tìm thấy (status = 200)  if (xhttp.readyState == 4 && xhttp.status == 200) {  alert("OK");  }  };  // gửi request  xhttp.send(); |

#### XMLHttpRequest

Công nghệ AJAX được phát triển dựa trên đối tượng XMLHttpRequest. [11]

**Đối Tượng XMLHttpRequest**

Đối tượng XMLHttpRequest được sử dụng để thực hiện việc toàn bộ quy trình trao đổi thông tin giữa trình duyệt (máy khách) và máy chủ mà không yêu cầu phải tải lại trang. [11]

Cú pháp sử dụng để tạo đối tượng XMLHttpRequest như sau:

**var** myXMLHttp = **new** XMLHttpRequest();

Hầu hết các trình duyệt phổ biến hiện nay (IE7+, Chrome, Firefox, Safari và Opera) đều hỗ trợ đối tượng XMLHttpRequest

Với IE5 và IE6 chúng ta sử dụng đối tượng ActiveX thay vì XMLHttpRequest:

**var** myXMLHttp = **new** ActiveXObject("Microsoft.XMLHTTP");

Cú pháp để tạo đối tượng XMLHttpRequest hỗ trợ IE5 và IE6 như sau:

|  |
| --- |
| Example.java |
| var myXMLHttp;  if (window.XMLHttpRequest) {  myXMLHttp = new XMLHttpRequest();  } else { // IE6, IE5  myXMLHttp = new ActiveXObject("Microsoft.XMLHTTP");  } |

**AJAX Hoạt Động Như Thế Nào**

Quy trình hoạt động của AJAX được thực hiện qua 3 bước:

* Bước 1: Tạo đối tượng XMLHttpRequest để gửi một yêu cầu từ trình duyệt (request) tới máy chủ. [11]
* Bước 2: Sử dụng đối tượng XMLHttpRequest vừa tạo ở trên và đăng ký các nhiệm vụ cần thực hiện khi máy chủ trả về kết quả (response). [11]
* Bước 3: Khi máy chủ trả về kết quả, thực hiện các nhiệm vụ đã đăng ký ở bước 2. [11]

#### Response

Khi máy chủ trả về kết quả về cho trình duyệt, chúng ta có thể sử dụng đối tượng XMLHttpRequest để lấy ra dữ liệu trả về thông qua 1 trong 2 thuộc tính sau:

* responseText: Sử dụng thuộc tính này khi dữ liệu trả về thuộc kiểu chuỗi. [11]
* responseXML: Sử dụng thuộc tính này khi dữ liệu trả về thuộc kiểu XML. [11]

**Thuộc Tính responseText**

Thuộc tính responseText được sử dụng khi kết quả trả về không thuộc kiểu XML. Cách sử dụng thuộc tính này khá đơn giản:

|  |
| --- |
| Example.java |
| // cấu hình XMLHttpRequest  xhttp.open("GET", "/get-name.php?id=1", true);  xhttp.onreadystatechange = function() {  // mã lệnh sẽ được thực thi khi máy chủ gửi về kết quả (readyState = 4)  // và trang yêu cầu đã được tìm thấy (status = 200)  if (xhttp.readyState == 4 && xhttp.status == 200) {  alert(xhttp.responseText);  }  };  // gửi request  xhttp.send(); |

**Thuộc Tính responseXML**

Sử dụng thuộc tính responseXML khi kết quả trả về thuộc kiểu XML. Khác với kiểu dữ liệu chuỗi khi kết quả trả về thuộc kiểu XML, chúng ta sẽ cần phải phân tích các thẻ trong XML.

|  |
| --- |
| Example.java |
| // cấu hình XMLHttpRequest  xhttp.open("GET", "/get-name.php?id=1", true);  xhttp.onreadystatechange = function() {  if (xhttp.readyState == 4 && xhttp.status == 200) {  var result = "";  var xmlDoc = xhttp.responseXML;  var names = xmlDoc.getElementsByTagName("NAME");  for (i = 0; i < names.length; i++) {  result += names[i].childNodes[0].nodeValue + "<br>";  }  document.getElementById("students").innerHTML = result;  }  };  // gửi request  xhttp.send(); |

#### Request

**Gửi Request**

Để gửi một yêu cầu tới máy chủ chúng ta sử dụng hai phương thức open() và

send() của đối tượng XMLHttpRequest. Ví dụ như sau:

|  |
| --- |
| Example.java |
| XMLHttpObj.open(method, url, async)  XMLHttpObj.send(data); |

Ở trên phương thức open() có 3 tham số trong đó:

* **method**: Quy định phương thức HTTP nào sẽ được sử dụng để gửi yêu cầu tới máy chủ. Các phương thức HTTP phổ biến hay sử dụng là GET,

POST, PUT và DELETE. [11]

* **url**: Địa chỉ URL để gửi yêu cầu tới máy chủ. Địa chỉ này có thể là tương đối hoặc tuyệt đối. Ví dụ: mysite.com/get-data.html hoặc /get-data.html. [11]
* **async**: Nhận 1 trong 2 giá trị true hoặc false, dùng để quy định yêu cầu gửi đi là đồng bộ (async = false) hay bất đồng bộ (async = true). [11]

Phương thức send(data) chỉ nhận 1 đối số truyền vào data là dữ liệu truyền lên máy chủ trong trường hợp phương thức HTTP gửi đi là **POST**. Đối với các phương thức HTTP còn lại thì send() sẽ không nhận đối số truyền vào. [11]

Bây giờ chúng ta hãy tham khảo 2 ví dụ sử dụng lần lượt các phương thức là GET và POST.

**Phương Thức GET**

Đoạn mã dưới đây sẽ gửi một yêu cầu bất đồng bộ tới trang /get-name.php?id=1 sử dụng phương thức **GET**.

|  |
| --- |
| Example.java |
| xhttp.open("GET", "/get-name.php?id=1", true);  xhttp.send(); |

**Phương Thức POST**

Đoạn mã dưới đây sẽ gửi một yêu cầu bất đồng bộ tới địa chỉ trang /set-name.php sử dụng phương thức POST.

|  |
| --- |
| Example.java |
| xhttp.open("POST", "/set-name.php", true);  xhttp.send("id=1&name=A"); |

Bạn cũng để ý ở trên chúng ta truyền vào phương thức send() đối số với giá trị *id=1&name=A* để gửi dữ liệu.

#### jQuery Ajax

**Phương thức “load()”**

Dùng phương thức trong các trường hợp cần load dữ liệu đơn giản và không dùng đến dữ liệu trả về.

Đầu tiên chuẩn bị phần body của file HTML

|  |
| --- |
|  |
| <div>          <a href="#" id="load-du-lieu">Load dữ liệu</a>  </div>  <div id="noidung"> |

Tiếp theo chuẩn bị cặp thẻ “<script>”

|  |
| --- |
|  |
| <script type="text/javascript">  $(document).ready(function() {  $('#load-du-lieu').click(function(e) {  e.preventDefault();  $('#noidung').load('vidu.html');  });  });  </script> |

Trong thẻ <script> có sử dụng:

* + $(document).ready(function(){…}): tương tự với sự kiện “onload”, tức là code chỉ được thực thi khi trang được load hoàn tất. [17]
  + Sự kiện “click” đối số mặc định “e” được sử dụng phương thức “preventDefault()” giúp chặn lại việc load lại trang từ trình duyệt. [17]
  + $(‘#id-can-lay’): trỏ tới thẻ HTML mà chúng ta cần lấy có “id” là “#id-can-lay”. Tương tự việc sử dụng cho “class”. [17]
  + load() : có đối số là đường dẫn đến file cần lấy dữ liệu. [17]

Nội dung của file index.html

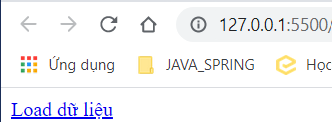
|  |
| --- |
| Index.html |
| <!DOCTYPE html>  <html>  <head>  <meta charset="utf-8">  <meta http-equiv="X-UA-Compatible" content="IE=edge">  <title>Hướng dẫn jQuery Ajax - Thienanblog.com</title>  <script type="text/javascript" src="jquery-1.11.3.min.js"></script>  </head>  <body>  <div>  <a href="#" id="load-du-lieu">Load dữ liệu</a>  </div>  <div id="noidung">    </div>  <script type="text/javascript">  $(document).ready(function() {  $('#load-du-lieu').click(function(e) {  e.preventDefault();  $('#noidung').load('vidu.html');  });  });  </script>  </body>  </html> |

Tiếp theo đến nội dung file vidu.html

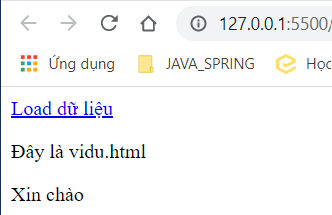
|  |
| --- |
| vidu.html |
| <!DOCTYPE html>  <html lang="en">  <head>  <meta charset="UTF-8">  <meta http-equiv="X-UA-Compatible" content="IE=edge">  <meta name="viewport" content="width=device-width, initial-scale=1.0">  <title>ViDu</title>  </head>  <body>  <p>Đây là vidu1.html</p>  <p id="chuoi-can-lay"> Xin chào </p>  </body>  </html> |

Tiến hành chạy file index.html

* + Load file “index.html”

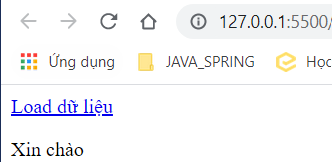


* + Nhấn vào nút “Load dữ liệu”



* + Load dữ liệu trong thẻ <p> có id là “chuoi-can-lay”

|  |
| --- |
| Index.html |
| <script type="text/javascript">  $(document).ready(function() {  $('#load-du-lieu').click(function(e) {  e.preventDefault();  console.log($('#noidung').load('vidu.html #chuoi-can-lay'));  });  });  </script> |



**Phương thức “get()”**

Phương thức “get()” chính là dùng “method” là GET trong <form>. Dùng phương thức trong các trường hợp cần load dữ liệu đơn giản và không dùng đến dữ liệu trả về.

Nội dung trong thẻ <script>

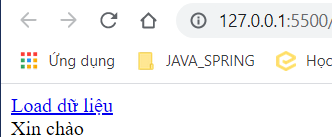
|  |
| --- |
| Index.html |
| <script type="text/javascript">  $(document).ready(function() {  $('#load-du-lieu').click(function(e) {  e.preventDefault();  $.get('vidu.html', function(ketqua) {  $('#noidung').html(ketqua);  $('#noidung').html($('#chuoi-can-lay').html());  });    });  });  </script> |

Đoạn code này cho chúng ta thấy rằng:

* + Để sử dụng được phương thức “get()”, chúng ta cần dùng biến “$” sau đó sử dụng thẳng phương thức “get()”. [17]
  + Đối số truyền vào không thể tách lọc dữ liệu trả về thẳng trong đường dẫn như phương thức “load()”. [17]
  + Kết quả trả về sẽ thông qua đối số thứ 2 của phương thức “get()” là một Anonymous Function [[1]](#footnote-1)có đối số mặc định trả về là “ketqua”. [17]

Để tách lọc dữ liệu từ “ketqua” chúng ta cần

Kết quả khi chạy đoạn code trên:



**Phương thức post()**

Phương thức “post()” chính là dùng “method” là POST trong <form>.

Cách dùng phương thức “post()” giống với phương thức “get()”. Nhưng khác nhau về kỹ thuật truyền dữ liệu. [17]

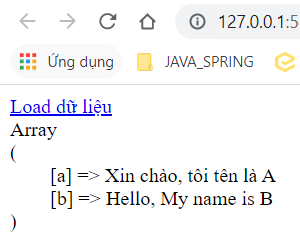
Chuẩn bị thẻ <script>

|  |
| --- |
| Index.html |
| <script type="text/javascript">  $(document).ready(function() {  $('#load-du-lieu').click(function(e) {  e.preventDefault();  $.post('vidu.php', {  a: "Xin chào, tôi tên là A",  b: "Hello, My name is B"  },function(ketqua) {  $('#noidung').html(ketqua);  });    });  });  </script> |

Tiếp theo chuẩn bị file vidu.php

|  |
| --- |
| vidu.php |
| <?php  echo '<pre>' .print\_r($\_POST,true). '</pre>'; |

* + Ở phương thức “post()”, đối số thứ 2 sẽ sử dụng “một đối tượng” (Object) chứa những dữ liệu mà chúng ta cần truyền qua đường dẫn “vidu.php” để nhận và trả về dữ liệu. Kết quả trả về sẽ được truyền vào đối số thú 3. [17]
* Kết quả sao khi chạy đoạn code trên



**Phương thức “ajax()”**

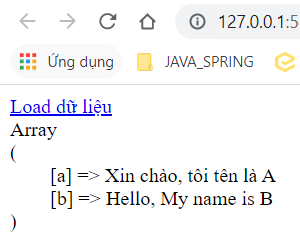
Cách sử dụng phương thức này giống cách gọi của phương thức “get()” và “post()”, nhưng thay vì truyền đường dẫn trực tiếp vào phương thức thì phương thức “ajax()” chấp nhận đối số đầu tiên chính là cấu hình cho kĩ thuật Ajax. [17]

Chuẩn bị nội dung thẻ <script>

|  |
| --- |
| Index.html |
| <script type="text/javascript">  $(document).ready(function() {  $('#load-du-lieu').click(function(e) {  e.preventDefault();  $.ajax({  url: 'vidu.php',  type: 'POST',  dataType: 'html',  data: {  a: "Xin chào, tôi tên là A",  b: "Hello, My name is B"  }  }).done(function(ketqua) {  $('#noidung').html(ketqua);  });    });  });  </script> |

Đối số đầu tiên truyền vào cho phương thức “ajax()” chính là một đối tượng (Object) gồm các thuộc tính cấu hình để kĩ thuật Ajax được thực thi. Trong đó: [17]

* + url: chuỗi chứa đường dẫn tới file cần lấy và trả về dữ liệu. [17]
  + type: phương thức gửi đi tương tự như của “<form>”, mặc định là “GET” nếu như không có dữ liệu truyền vào. [17]
  + dataType: xác định dữ liệu trả về thuộc dạng nào? Nếu không truyền dữ liệu vào thì Jquery tự động nhận biết kiểu dữ liệu (script, html, json, …). [17]
  + data: truyền dữ liệu sang đường dẫn chỉ định để thực hiện xử lý và trả về dữ liệu. [17]
  + “done()”:
* Kết quả sau khi chạy đoạn code trên



## MYSQL

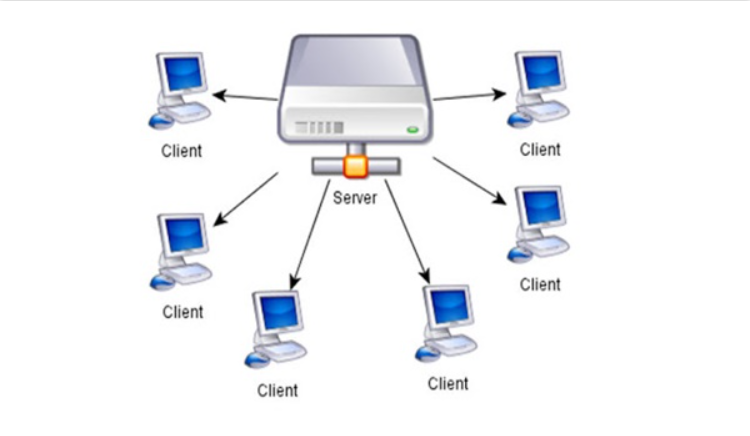
* + 1. **Thuật ngữ thường gặp của MySQL**
       1. *Database*

Databasr là tập hợp dữ liệu theo cùng một cấu trúc được đặt trong một bộ dữ liệu chung. Database được tổ chức sắp xếp giống như một bảng tính có sự liên kết chặt chẽ với nhau. [12]

* + - 1. *Open source*

Open source được hiểu là mã nguồn mở. Open source cho phép mọi người sử dụng, cài đặt và chỉnh sửa theo ý của bản thân khi cài đặt một phần mềm nào đó. Hiện nay có nhiều ngôn ngữ lập trình mã nguồn mở phổ biến như PHP hoặc CMS WordPress.

* + - 1. *Mô hình Client-server*



**Hình 3. 1Mô hình Client Server [12]**

MySQL Client-Server [[2]](#footnote-2)là mô hình chạy phần mềm RDBMS[[3]](#footnote-3). Một máy client sẽ liên lạc với máy server trong một mạng nhất định. Mỗi client có thê gửi một request từ giao diện người dùng trên màn hình và server sẽ trả về như kết quả mong muốn.

* + - 1. *MySQL Server là gì?*

Đây là máy tính hay một hệ các máy tính cài đặt phần mềm MySQL dành cho server để giúp lưu trữ dữ liệu trên đó để máy khách có thể truy cập vào quản lý. [12]

* + - 1. *MySQL Client là gì?*

MySQL Client là tên gọi chung của mọi phần mềm có thể thực hiện truy vấn MySQL server và nhận kết quả trả về. Hay nói cách khác, MySQL Client là đoạn mã PHP script trên máy tính hay trên cùng server dùng để kết nối tới MySQL database. [12]

### MySQL và lịch sử phát triển

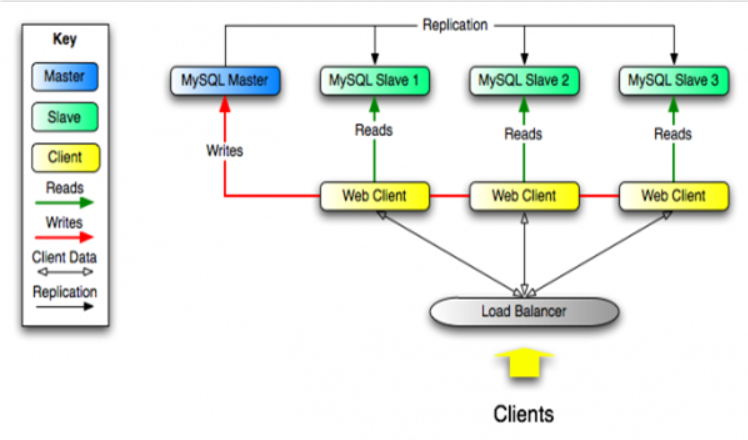
MySQL là một hệ thống quản trị cơ sở dữ liệu mã nguồn mở (gọi tắt là RDBMS) hoạt động theo mô hình client-server. MySQL được tích hợp apache, PHP. MySQL quản lý dữ liệu thông qua các cơ sở dữ liệu. Mỗi cơ sở dữ liệu có thể có nhiều bảng quan hệ chứa dữ liệu. MySQL cũng có cùng một cách truy xuất và mã lệnh tương tự với ngôn ngữ SQL. [12]

MySQL được phát triển, phân phối và hỗ trợ bởi Oracle Corporation. Phần mềm được đặt tên theo tên của con gái người đồng sáng lập Monty Widenius: My. Ban đầu MySQL được phát triển bởi một công ty Thụy Điển MySQL AB vào năm 1994. [13]

MySQL có thể được cài đặt trên nhiều nền tảng hệ điều hành khác nhau như Windows, Linux và Mac OS X. Ngoài MySQL ra, còn một số DBMS khác như: Oracle, PostgreSQL, SQLite, … [13]

**Lịch sử hình thành và phát triển của MySQL**

* Công ty Thụy Điển MySQL AB phát triển MySQL vào năm 1994. [12]
* Phiên bản đầu tiên của MySQL phát hành năm 1995. [12]
* Công ty Sun Microsystems mua lại MySQL AB trong năm 2008. [12]
* Năm 2010 tập đoàn Oracle mua lại Sun Microsystems. Ngay lúc đó, đội ngũ phát triển của MySQL tách ra thành 1 nhánh riêng gọi là MariaDB. Oracle tiếp tục phát triển MySQL lên phiên bản 5.5. [12]
* 2013 MySQL phát hành phiên bản 5.6. [12]
* 2015 MySQL phát hành phiên bản 5.7. [12]
* MySQL đang được phá triển lên phiên bản 8.0. [12]
  + 1. **Ưu điểm và nhược điểm của MySQL**
       1. *Ưu điểm*
* Dễ sử dụng: MySQL đơn giản, dễ sử dụng.
* Bảo mật cao: MySQL sở hữu khá nhiều tính năng bảo mật, bao gồm các loại hình bảo mật cấp cao.
* Đa tính năng: MySQL cung cấp nhiều tính năng mà bất cứ hệ quản trị nào cơ sở dữ liệu quan hệ nào cũng phải mong đợi.
* Vận hành mạnh mẽ và mở rộng dễ dàng: MySQL có khả năng xử lý một lượng lớn dữ liệu. Bên cạnh đó, người dùng có thể mở rộng nó nếu có nhu cầu.
* Nhanh chóng: Tốc độ hoạt động của MySQL nhanh hơn các phần mềm khác nhờ các tiêu chuẩn được tích hợp sẵn.
* Có thể khôi phục dữ liệu: MySQL cho phép người dùng khôi phục dữ liệu, tránh khỏi ảnh hưởng của các sự cố.
  + - 1. *Nhược điểm*
* Bị giới hạn: MySQL bị hạn chế về một vài tính năng mà các ứng dụng có thể sẽ cần đến.
* Độ tin cậy không quá cao: So với các hệ quản trị cơ sở dữ liệu quan hệ khác, độ tin cậy của MySQL không quá cao.
* Bị hạn chế về dung lượng: Số bản ghi trong MySQL càng tăng thì truy xuất dữ liệu càng trở nên khó khăn do hạn chế về dung lượng.
  + 1. **Hoạt động của MySQL**



**Hình 3. 2 Cách MySQL hoạt động [12]**

Trong môi trường MySQL, máy khách (client) và máy chủ (server) tương tác qua lại với nhau theo nguyên lý:

* MySQL tạo ra các bảng nhằm lưu trữ dữ liệu, đồng thời định nghĩa mối quan hệ giữa các bảng đó. [12]
* Client gửi yêu cầu SQL bằng lệnh đặc biệt lên trên MySQL. [12]
* Ứng dụng trên server nhận được và phản hồi thông tin, trả kết quả về máy khách (client). [12]
  + 1. **Lý do Sử Dụng MySQL**
* MySQL là CSDL có tốc độ khá cao, ổn định và khá dễ sử dụng có thể hoạt động được trên khá nhiều hệ điều hành. [12]
* Tính bảo mật mạnh và sử dụng được trên nhiều ứng dụng mà MySQL còn hoàn toàn được sử dụng miễn phí. [12]
* MySQL không chỉ dừng lại ở bổ trợ cho PHP và Perl, mà nó còn bổ trợ cho nhiều ngôn ngữ khác, Nó là nơi để lưu trữ thông tin trên các trang web được viết bằng Perl hoặc PHP. [12]
  1. **Ý TƯỞNG VỀ CÁCH LẬP TRÌNH ĐỂ TẠO MỘT WEBSITE HỌC IELTS TRỰC TUYẾN.**
     1. **Ý tưởng lập ra website học IELTS trực tuyến**

#### Các công cụ hỗ trợ lập trình

* Spring tools tuite 4

Spring tool suite (STS) là một extension mở rộng cho eclipse, chuyên dùng để hỗ trợ phát triển các ứng dụng bằng Spring Framework

* MySQL Workbench

MySQL Workbench chính là một chương trình giúp cho người lập trình có thể giao tiếp với hệ cơ sở dữ liệu MySQL thay vì phải sử dụng các lệnh Command-line phức tạp và mất thời gian. MySQL Workbench được thiết kế đơn giản, dễ sử dụng và có thể thích ứng với nhiều hệ điều hành như là Microsoft Windowns, Max OS, Linux hay Ubuntu.

* Postman

Postman là một công cụ cho phép chúng ta thao tác với API, phổ biến nhất làREST. Postman hiện là một trong những công cụ phổ biến nhất được sử dụng trong thử nghiệm các API. Với Postman, ta có thể gọi Rest API mà không cần viết dòng code nào.

#### Các chức năng cần có của một website học IELTS trực tuyến

* Các chức năng dành cho admin
  + - Đăng ký, đăng nhập, reset mật khẩu,thay đổi thông tin, đăng xuất.
    - Quản lý user
    - Quản lý bài ngữ pháp
    - Quản lý bài từ vựng
    - Quản lý bài đọc Ielts
    - Quản lý bài nghe Ielts
* Các chức năng dành cho client
  + - Đăng ký, đăng nhập, reset mật khẩu, thay đổi thông tin, đăng xuất.
    - Xem các bài giảng về ngữ pháp, từ vựng có trên website
    - Làm bài tập từ vựng
    - Làm bài tập ngữ pháp
    - Thi thử bài đọc Ielts
    - Thi thử bài nghe Ielts
    - Tra cứu kết quả quá trình học tập

#### Ứng dụng các công nghệ trong quá trình xây dựng các chức năng

* Các chức năng của các công nghệ trong đề tài nghiên cứu
* **Spring Boot** : Khởi tạo và chạy dự án và xây dựng web app
* **Spring Security** : Mã hóa mật khẩu và xác thực người dùng
* **Spring Data JPA** : Thao tác với cơ sở dữ liệu
* **Ajax :** Tải về nội dung của một hoặc nhiều phần của trang hiện tại mà không cần phải tải lại toàn bộ nội dung của trang
* Các công nghệ cần sử dụng cụ thể trong từng chức năng
* Các chức năng dành cho admin
  + - Đăng ký, đăng nhập, reset mật khẩu,thay đổi thông tin, đăng xuất : Spring boot , Spring Security
    - Quản lý user : Spring boot , Ajax
    - Quản lý bài ngữ pháp : Spring boot , Ajax
    - Quản lý bài từ vựng : Spring boot , Ajax
    - Quản lý bài đọc Ielts : Spring boot , Ajax
    - Quản lý bài nghe Ielts : Spring boot , Ajax
* Các chức năng dành cho client
  + - Đăng ký, đăng nhập, reset mật khẩu, thay đổi thông tin, đăng xuất: Spring boot , Spring Security
    - Xem các bài giảng về ngữ pháp, từ vựng có trên website : Spring boot , Ajax
    - Làm bài tập từ vựng : Spring boot , Ajax
    - Làm bài tập ngữ pháp : Spring boot , Ajax
    - Thi thử bài đọc Ielts : Spring boot , Ajax
    - Thi thử bài nghe Ielts : Spring boot , Ajax
    - Tra cứu kết quả quá trình học tập: Spring boot , Ajax
    1. **Cách triển khai cài đặt**

Bước 1: Phân tích thiết kế hệ thống website học IELTS trực tuyến

Bước 2: Cài đặt các công cụ hỗ trợ lập trình như ở mục [1.5.1.1](#CCHT)

Bước 3 : Khởi tạo và chạy dự án bằng spring boot , xây dựng cấu trúc project cấu trúc đã nêu ở mục [1.1.4.1](#_Cấu_trúc_dự)

Bước 4 : Xây dựng kết nối cơ sở dữ liệu bằng các mapping các entities đã thiết kế trong quá trình thiết kế hệ thống bằng xuống MySQL WorkBench bằng cách sử dụng Spring boot JPA

Bước 5: Xây dựng chức năng đăng nhập đăng kí , áp dụng Spring security để mã hóa mật khẩu và xác thực người dùng

Bước 6: Xây dựng các chức năng khác theo nghiệp vụ đã thiết kế trong mục phân tích thiết kế hệ thống website học IELTS trực tuyến

**Chương 2. TỔNG QUAN VỀ CÁC TRANG WEBSITE HỌC IELTS TRỰC TUYẾN NỔI TIẾNG**

## IELTS Fighter

### Giới thiệu

Ielts-fighter.com là website luyện thi ielts online miễn phí đầu tiên và chất lượng tại Việt Nam. Các bài học trên website được phân loại theo các mức độ phù hợp với từng trình độ của người học. Bên cạnh đó, Ielts Fighter còn chia sẻ hữu ích về ielts, các kinh nghiệm luyện thi ielts, nguồn tài liệu học ielts phong phú. Website còn cung cấp lộ trình học phù hợp cho từng trình độ.

* + 1. **Lịch sử phát triển**

Thành lập từ tháng 5/2016, sau hơn năm hoạt động, IELTS Fighter chính thức cán mốc 100,000 học viên. Con số này đã cho thấy được chất lượng giảng dạy cũng như uy tín của thương hiệu trong cộng đồng giảng dạy IELTS. [15]

Từ tháng 9/2017, IELTS Fighter trở thành đối tác của IDP Việt Nam và trở thành địa điểm thi IELTS chính thức của IDP tại Hà Nội và Hồ Chí Minh. [15]

* + 1. **Ưu và nhược điểm**
       1. *Ưu điểm*
* Giao diện dễ nhìn, phù hợp với người dùng, chúng ta dễ dàng lựa chọn mục tiêu mà mình muốn học.
* Các chương tình học được liệt kê rõ ràng.
  + - 1. *Nhược điểm*
* File âm thanh của bài nghe hơi rè, khó nghe cho người mới bắt đầu.
  + 1. **Các tính năng nổi bật của trang web**
* Có thể tải bải nghe về máy để nghe lúc không có mạng.
* Ở mỗi bài học, có video bài giảng.
  1. **VOA Learning English**
     1. **Giới thiệu**

VOA Learning English là ứng dụng học tiếng Anh miễn phí được xây dựng từ chương trình đặc biệt của đài tiếng nói Hoa Kỳ. Ứng dụng giúp bạn dễ dàng tự học luyện nghe tiếng Anh mỗi ngày để cải thiện kỹ năng nghe và vốn từ vựng của mình, thông qua việc nghe tin tức trên radio, video đa dạng.

* + 1. **lịch sử phát triển**

VOA Learning English là kênh học tiếng Anh được VOA phát triển vào năm 1959 với tên ban đầu Special English. Nếu bạn hỏi 10 người học tiếng Anh thì chắc chắn có tới 9 người biết đến cái tên này.

Các bản tin ngắn được phát sóng cố định dưới tên Special English trong hơn nửa thập kỷ đã trở thành tài liệu học tiếng Anh uy tín của hàng triệu người.

Năm 2014 sau khi mở rộng thêm nội dung và tài liệu học tập, Special English được đổi tên thành Learning English.

* + 1. **Ưu và nhược điểm** 
       1. *Ưu điểm*
* Giao diện khá đơn giản, cung cấp đầy đủ thông tin cần thiết và hỗ trợ mình tìm kiếm dễ dàng
* Nội dung phong phú, tin tức cập nhật mới nhất
* Hệ thống các bài viết có chủ đề liên quan đến IELTS
* Định dạng video sinh động
  + - 1. *Nhược điểm*
* VOA Learning English chỉ cung cấp các file nghe chứa giọng Anh Mỹ đặc trưng. Đối với bài thi IELTS, ngoài giọng Anh – Mỹ, bạn còn bắt gặp giọng Anh – Anh, Anh – Úc, Anh – Canada hoặc thậm chí Anh – Ấn
  + 1. **Các tính năng nổi bật của trang web**
* Nghe Audios tin tức thế giới mỗi ngày bằng tiếng Anh
* Xem video bằng tiếng Anh với nhiều nội dung thú vị
* Các bài nghe sẽ được tô sáng và tự động cuộn theo quá trình nghe
* Tra từ điển, dịch ngĩa từ vựng trực tiếp khi nghe
* Điều chỉnh tốc độ nghe và tự động lặp lại
* Tải nội dung bài học sử dụng Offline, sử dụng miễn phí
  1. **Elllo.org**
     1. **Giới thiệu**

Elllo.org là trang học tiếng anh trực tuyến chuyên về kỹ năng nghe. Chúng ta có thể nghe các bài nói từ các diễn giả trên khắp thé giới về đa dạng các chủ đề khác nhau như thể thao, du lịch, giải trí, …

Có cả phần phụ đề ở phía dưới, rất thuận tiện cho việc học tiếng anh. Ngoài ra có rất nhiều các bài hát, video clips tiếng anh thú vị. Elllo.org đặc biệt hữu ích đối với những bạn có nhu cầu luyện nghe tiếng anh.

* + 1. **Lịch sử phát triển**

Elllo.ord được tạo ra năm 2003 bởi một giáo viên dạy tiếng Anh ở Nhật Bản, mang đến 2500 bài học tiếng anh miễn phí cho học viên ở hơn 100 quốc gia trên thế giới. [16]

Trải qua hơn 13 năm phát triển, Elllo đã xây dựng hệ thống chủ đề học phong phú, đa dạng, nhờ sự góp sức của rất nhiều giảng viên trên thế giới. [16]

* + 1. **Ưu và nhược điểm**
       1. *Ưu điểm*
* Cung cấp hàng nghìn transcript, slide show, audio notes trong đời thực giúp người đọc cảm thấy hứng thú trong việc nâng cao khả năng nghe nói tiếng Anh.
* Tốc độ thực và nội dung gần gũi với cuộc sống, kết hợp đoạn audio ngắn giải thích các ví dụ về một số từ mới trọng tâm trong bài.
* Hỗ trợ tính năng phụ đề phù hợp cho những bạn mới đầu nghe.
  + - 1. *Nhược điểm*
    1. **Các tính năng nổi bật của trang web:**
* Đối với Audio: mỗi audio bao gồm 5 phần:
  + Audio: đoạn hội thoại dạng âm thanh dành cho luyện nghe.
  + Script: bản text của hội thoại.
  + Vocab: giải thích nghĩa và cách sử dụng từ vựng mới trong bài nghe, ví dụ cụ thể.
  + Quiz: được thiết kế dưới dạng câu hỏi trắc nghiệm, hỏi các vấn đề liên quan đến nội dung của đoạn hội thoại.
  + Vocabulary challenge: củng cố kiến thức sử dụng từ vựng bằng thử thách điền từ còn thiếu vào chỗ trống.
* Đối với Video: mỗi vidoe gồm 3 phần:
  + Video: dạng hình ảnh và âm thanh.
  + Transcript: nội dung bài nói của video.
  + Quiz: trả lời những câu hỏi liên quan đến nội dung được đề cập đến trong video.
  1. **Duolingo**
     1. **Giới thiệu**

Duolingo là một trang web học tiếng anh online, miễn phí tuyệt vời giúp học ngữ pháp và từ vựng hiểu quả. Cách dạy tiếng anh online tạo đây sẽ được xây dựng thông qua trò chơi và những câu đố tương tác giúp chúng ta có thể học trực tuyesn hoặc chọn học trên ứng dụng di động. [19]

Ở Duolingo, các bài học được phân chia theo nhiều chủ đề khác nhau. Các chủ đề vô cùng thân quen với đời sống giúp người học dễ dàng tiếp thu như: Thực phẩm, quần áo, gia đình, ... Để chuyển sang chủ đề khác, người học sẽ phải hoàn thành các bài kiểm tra một cách chính xác nhất thông qua việc học lặp đi lặp lại các từ vựng, cụm từ. [19]

* + 1. **Lịch sử phát triển**

Duolingo khởi động đợt beta kín vào ngày 30 tháng 11 năm 2011 và thu hút hơn 300.000 lượt người đăng ký. [20]

Vào ngày 19 tháng 6 năm 2012, Duolingo chính thức công khai ra mắt. [20]

Vào năm 2013, Apple chọn Duolingo cho danh hiệu Iphone App of the year (ứng dụng iphone của năm), trở thành ứng dụng giáo dục đầu tiên đạt được thành tích này. [20]

Duolingo giành giải Crunchies 2014 ở hạng mục Best Education. [20]

Startup và là ứng dụng giáo dục được tải về nhiều nhất trên Google Play trong năm 2013 và 2014. [20]

Tính đến tháng 1 năm 2014, Duolingo đạt trên 60 triệu người dùng, trong đó có khoảng 20 triệu người tích cực sử dụng. [20]

* + 1. **Ưu và nhược điểm**
       1. *Ưu điểm*
* Giao diện thân thiện, đẹp mắt, hỗ trợ tiếng việt.
* Công cụ học 100% miễn phí.
* Sử dụng được hầu hết trên các thiết bị di động.
* Được thiết kế đầy đủ từ các level cơ bản đến nâng cao.
  + - 1. *Nhược điểm*
* Duolingo thiết kế các level từ điểm xuất phát lên cao dần nên thường chỉ phù hợp vói người chưa biết gì, nên dẫn đến tình trạng gây chán nản đối với người đã có kiến thức từ trước.
  + 1. **Các tính năng nổi bật của trang web**
* Cách dạy tiếng anh được xây dựng thông qua trò chơi và những câu đố tương tác. [19]
* Các bải học được phân chia theo nhiều chủ đề khác nhau. [19]
* Các chủ đề vô cùng thân quen với đời sống. [19]

## IELTS-Blog

### Giới thiệu

Ielts-Blog là website cực kỳ lâu đời và kinh điển hỗ trợ các thí sinh luyện thi IELTS trực tuyến miễn phí trên toàn thế giới. Chúng ta có thể tìm thấy các bài viết mẫu chia theo từng thang điểm mà mình mong muốn, các lời khuyên từ những người đã tham gia thi IELTS đạt điểm cao.

Tuy nhiên, website khá chú trọng tới kỹ năng writing (viết) của IELTS.

### Lịch sử phát triển

Ban đầu được tạo ra bởi Simone Braverman vào năm 2005, trong thời gian đó IELTS-Blog.com đã trở thành một nguồn tài nguyên lớn, chứa đầy thông tin về các kỳ thi IELTS gần đây, các lời khuyên được cung cấp bởi nhiều nhà văn và cộng tác viên từ khắp nơi trên thế giới và nhiều liên kết hữu ích khác tài nguyên IELTS. [19]

Trong năm 2011, trang web này đã được hơn 1.445.000 người truy cập và gần 30.000 người đăng ký nhận các bản cập nhật hàng ngày. [19]

### Ưu và nhược điểm

* + - 1. *Ưu điểm*
* Giao diện thân thiện, đẹp mắt
* Web luyện thi ielts trực tuyến miễn phí trên toàn thế giới.
* Được thiết kế đầy đủ từ các level cơ bản đến nâng cao.
  + - 1. *Nhược điểm*
* Trang web chú trọng về phần viết hơn các phần (nghe, nói, đọc).
* Không có file nghe cho cách phần nghe.

### Các tính năng nổi bật của trang web

* Trang có chia sẻ các lời khuyên luyện thi ielts.
* Chia sẻ các bài thi ielts gần nhất.

# Chương 3 PHÂN TÍCH THIẾT KẾ HỆ THỐNG WEBSITE HỌC IELTS TRỰC TUYẾN

* 1. **ĐẶC TẢ ĐỀ TÀI**
     1. **Mô tả chung về hệ thống website học ielts trực tuyến**

Yêu cầu hệ thống : Hệ thống website học Ielts trực tuyến , là một website chia sẻ các bài học và các nguồn tài liệu về Ielts , luyện tập để thi Ielts . Hệ thống đáp ứng tốt việc quản trị và cập nhật nội dung . Hệ thống bảo mật và hoạt động ổn định

**Các tính năng có trong website**

* Ngôn ngữ lập trình: JAVA , JAVASCRIPT
* Hệ thống cơ sở dữ liệu: MySQL

Hệ thống website học Ielts trực tuyến sẽ có những chức năng chủ yếu sau:

|  |  |
| --- | --- |
| **Chức năng** | **Mô tả** |
| Các chức năng dành cho admin | |
| Đăng ký, đăng nhập, reset mật khẩu, thay đổi thông tin, đăng xuất. | * Người dùng muốn tham gia vào hệ thống, cần phải đăng kí tài khoản người dùng. Người dùng cần cung cấp thông tin để xác nhận tài khoản. * Chức năng đăng nhập: dùng để đăng nhập vào hệ thống. * Chức năng reset mật khẩu: dùng để lấy mật khẩu mặc định khi người dùng quên mật khẩu. * Chức năng thay đổi thông tin: dùng để thay đổi thông tin cá nhân của người dùng. * Chức năng đăng xuất: dùng để thoát khỏi hệ thống. |
| Quản lý user | Người dùng thực hiện các công việc thêm, cập nhật và xóa user. |
| Quản lý bài từ vựng | Người dùng thực hiện các công việc thêm, cập nhật và xóa bài từ vựng. |
| Quản lý bài ngữ pháp | Người dùng thực hiện các công việc thêm, cập nhật và xóa bài ngữ pháp. |
| Quản lý bài đọc Ielts | Người dùng thực hiện các công việc thêm, cập nhật và xóa bài đọc Ielts. |
| Quản lý bài nghe Ielts | Người dùng thực hiện các công việc thêm, cập nhật và xóa nghe. |
| Các chức năng dành cho client | |
| Đăng ký, đăng nhập, reset mật khẩu, thay đổi thông tin, đăng xuất. | * Người dùng muốn tham gia vào hệ thống, cần phải đăng kí tài khoản người dùng. Người dùng cần cung cấp thông tin để xác nhận tài khoản. * Chức năng đăng nhập: dùng để đăng nhập vào hệ thống. * Chức năng reset mật khẩu: dùng để lấy mật khẩu mặc định khi người dùng quên mật khẩu. * Chức năng thay đổi thông tin: dùng để thay đổi thông tin cá nhân của người dùng. * Chức năng đăng xuất: dùng để thoát khỏi hệ thống. |
| Xem các bài giảng về ngữ pháp | Người dùng muốn xem lại các bài giảng về ngữ pháp. |
| Xem các bài giảng về từ vựng | Người dùng muốn xem lại các bài giảng về từ vựng. |
| Xem các bài giảng về bài đọc | Người dùng muốn xem lại các bài giảng về bài đọc. |
| Xem các bài giảng về bài nghe | Người dùng muốn xem lại các bài giảng về bài nghe. |
| Tra cứu kết quả quá trình học tập | Người dùng thực hiện tra cứu kết quả quá trình học tập của bản thân. |

**Giao diện**

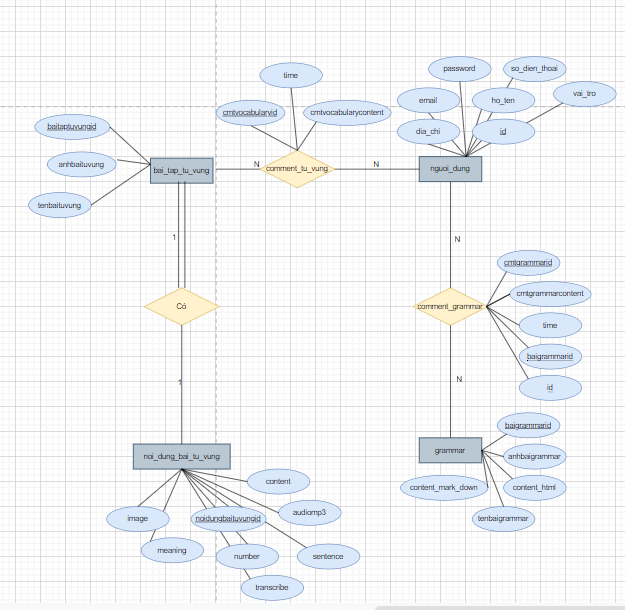
Được thiết kế theo quy chuẩn về CSS và giao diện thiết kế phù hợp với thói quen người dùng

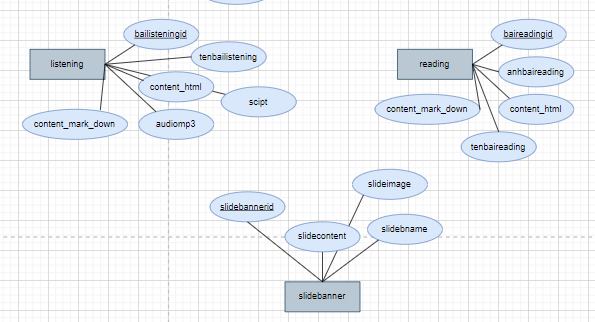
* Trang chủ được thiết kế theo phong cách hiện đại hấp dẫn người sử dụng
* Hiển thị đây đủ các thông tin về danh mục các bài giảng, tài liệu
* Trang quản trị thiết kế để tiện lới trong việc post bài cũng như chỉnh sửa bài viết
  + 1. **Xác định giới hạn hệ thống thông tin của NHÓM**

Admin: chỉ có thể chỉnh sửa thông tin cá nhân, quản lý và phân quyền người dùng, quản lý các bài học (bao gồm: bài học từ vựng, bài học ngữ pháp, bài đọc ielts, bài nghe ielts)

User: chỉnh sửa thông tin cá nhân, chỉ có quyền xem lại các bài giảng (bao gồm bài giảng về từ vựng, bài giảng về ngữ pháp, bài giảng về bài đọc, bài giảng về bài nghe), tra cứu kết quả học tập trong quá trình học tập.

* 1. **THIẾT KẾ THÀNH PHẦN DỮ LIỆU**
     1. **Mô hình thực thể kết hợp (ERD)**





### Danh sách lược đồ quan hệ

**Nguoi\_dung**(id, ho\_ten, email, so\_dien\_thoai, dia\_chi, password, vai\_tro)

**Grammar**(baigrammarid, anhbaigrammar, content\_html, tenbaigrammar, content\_mark\_down)

**Comment\_grammar**(cmtgrammarid, cmtgrammarcontent, time, baigrammar\_id, nguoidung\_id)

**Comment\_tu\_vung**(cmtvocabularyid, time, baitaptuvungid, cmtvocabularycontent,nguoidung\_id )

**Bai\_tap\_tu\_vung**(baitaptuvungid, anhbaituvung, tenbaituvung)

**Noi\_dung\_bai\_tu\_vung**(baitaptuvungid, image, meaning, transcribe, sentence, audiomp3,content, number)

**Slidebanner**(slidebannerid, slidecontent, slidename, slideimage)

**Listening**(bailisteningid, anhbailistening, content\_html, tenbailistening, content\_mark\_down, audio, script)

**Reading**(baireadingid, anhbaireading, content\_html, tenbaireading, content\_mark\_down)

### Từ điển dữ liệu

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 1. **BẢNG nguoi\_dung** | | | | | |
| **TT** | **Thuộc tính** | **Diễn giải** | **Kiểu dữ liệu** | **Ràng buộc** | **Ghi chú** |
| 1 | Id | Id của người dùng | bigint(20) | PRIMARY KEY | AUTO\_INCREMENT |
| 2 | dia\_chi | Địa chỉ của người dùng | varchar(255) | DEFAULT NULL |  |
| 3 | email | Eamil dùng để đăng nhập vào Website | varchar(255) | DEFAULT NULL |  |
| 4 | ho\_ten | Họ tên của người dùng | varchar(255) | DEFAULT NULL |  |
| 5 | passowrd | Mật khẩu dùng để đăng nhập vào website | varchar(255) | DEFAULT NULL |  |
| 6 | so\_dien\_thoai | Số điện thoại người dùng | varchar(255) | DEFAULT NULL |  |
| 7 | vai\_tro | Vai trò người dùng đăng nhập | int(11) | DEFAULT NULL |  |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **BẢNG ket\_qua** | | | | | |
| **TT** | **Thuộc tính** | **Diễn giải** | **Kiểu dữ liệu** | **Ràng buộc** | **Ghi chú** |
| 1 | ketquaid | Id của kết quả bài thi | int(11) | PRIMARY KEY | AUTO\_INCREMENT |
| 2 | correctlisten | Câu nghe đúng | int(11) | DEFAULT NULL |  |
| 3 | correctread | Câu chọn đúng | int(11) | DEFAULT NULL |  |
| 4 | time | Thời gian làm bài | datetime | DEFAULT NULL |  |
| 5 | socaudung | Số câu đúng | int(11) | DEFAULT NULL |  |
| 6 | socausai | Số câu sai | int(11) | DEFAULT NULL |  |
| 7 | baithithuid | Id bài thi thử | int(11) | FOREIGN KEY |  |
| 8 | nguoidungid | Id người dùng | bigint(20) | FOREIGN KEY |  |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **BẢNG comment\_grammar** | | | | | |
| **TT** | **Thuộc tính** | **Diễn giải** | **Kiểu dữ liệu** | **Ràng buộc** | **Ghi chú** |
| 1 | cmtgrammarid | Id của nhận xét ngữ pháp | int(11) | PRIMARY KEY | AUTO\_INCREMENT |
| 2 | cmtgrammarcontent | Nội dung nhận xét ngữ pháp | archar(255) | DEFAULT NULL |  |
| 3 | time | Thời gian nhận xét | datetime | DEFAULT NULL |  |
| 4 | baigrammarid | Id bài ngữ pháp | int(11) |  |  |
| 5 | id | Id người dùng | bigint(20) |  |  |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **BẢNG bai\_grammar** | | | | | |
| **TT** | **Thuộc tính** | **Diễn giải** | **Kiểu dữ liệu** | **Ràng buộc** | **Ghi chú** |
| 1 | baigrammarid | Id bài ngữ pháp | int(11) | PRIMARY KEY | AUTO\_INCREMENT |
| 2 | anhbaigrammar | Ảnh của bài ngữ pháp | varchar(255) | DEFAULT NULL |  |
| 3 | content\_html | Nội dung của bài ngữ pháp dưới dạng html | text |  |  |
| 4 | content\_mark\_down | Nội dung của bài ngữ pháp dưới dạng markdown | text |  |  |
| 5 | tenbaigrammar | Tên bài ngữ pháp | varchar(255) | DEFAULT NULL |  |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **BẢNG comment\_tu\_vung** | | | | | |
| **TT** | **Thuộc tính** | **Diễn giải** | **Kiểu dữ liệu** | **Ràng buộc** | **Ghi chú** |
| 1 | cmtvocabularyid | Id của nhận xét từ vựng | int(11) | PRIMARY KEY | AUTO\_INCREMENT |
| 2 | cmtvocabularycontent | Nội dung nhận xét từ vựng | varchar(255) | DEFAULT NULL |  |
| 3 | time | Thời gian nhận xét | datetime | DEFAULT NULL |  |
| 4 | baitaptuvungid | Id bài tập từ vựng | int(11) | FOREIGN KEY |  |
| 5 | id | Id người dùng | bigint(20) | FOREIGN KEY |  |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **BẢNG bai\_tap\_tu\_vung** | | | | | |
| **TT** | **Thuộc tính** | **Diễn giải** | **Kiểu dữ liệu** | **Ràng buộc** | **Ghi chú** |
| 1 | baitaptuvungid | Id bài tập từ vựng | int(11) | PRIMARY KEY | AUTO\_INCREMENT |
| 2 | anhbaitaptuvung | Ảnh của bài thi thử | varchar(255) | DEFAULT NULL |  |
| 3 | tenbaitaptuvung | Tên của bài thi thử | varchar(255) | DEFAULT NULL |  |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **BẢNG noi-dung\_bai \_tu\_vung** | | | | | |
| **TT** | **Thuộc tính** | **Diễn giải** | **Kiểu dữ liệu** | **Ràng buộc** | **Ghi chú** |
| 1 | noidungbaituvungid | Id của nội dung bài từ vựng | int(11) | PRIMARY KEY | AUTO\_INCREMENT |
| 2 | audiomp3 | Tên hiển thị của người dùng | varchar(255) | DEFAULT NULL |  |
| 3 | content | Eamil dùng để đăng nhập vào Website | varchar(255) | DEFAULT NULL |  |
| 4 | image | Mật khẩu dùng để đăng nhập vào website | varchar(255) | DEFAULT NULL |  |
| 5 | meaning | Tên của người dùng | varchar(255) | DEFAULT NULL |  |
| 6 | number | Địa chỉ | int(11) | DEFAULT NULL |  |
| 7 | sentence | Giới tính | varchar(255) | DEFAULT NULL |  |
| 8 | transcribe | Số điện thoại | varchar(255) | DEFAULT NULL |  |
| 9 | baitaptuvungid | Ngày sinh | int(11) |  |  |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **BẢNG slidebanner** | | | | | |
| **TT** | **Thuộc tính** | **Diễn giải** | **Kiểu dữ liệu** | **Ràng buộc** | **Ghi chú** |
| 1 | slidebannerid |  |  | PRIMARY KEY |  |
| 2 | slidename |  |  |  |  |
| 3 | slidecontent |  |  |  |  |
| 4 | slideimage |  |  |  |  |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **BẢNG bai\_listening** | | | | | |
| **TT** | **Thuộc tính** | **Diễn giải** | **Kiểu dữ liệu** | **Ràng buộc** | **Ghi chú** |
| 1 | bailisteningid | Id bài nghe | int(11) | PRIMARY KEY | AUTO\_INCREMENT |
| 2 | anhbailistening | Ảnh của bài nghe | varchar(255) | DEFAULT NULL |  |
| 3 | content\_html | Nội dung của bài nghe dưới dạng html | text |  |  |
| 4 | content\_mark\_down | Nội dung của bài nghe dưới dạng markdown | text |  |  |
| 5 | tenbainghe | Tên bài nghe | varchar(255) | DEFAULT NULL |  |
| 6 | Audio | File nghe | Varchar(255) |  |  |
| 7 | transcipt | Bài dịch | Text |  |  |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **BẢNG bai\_reading** | | | | | |
| **TT** | **Thuộc tính** | **Diễn giải** | **Kiểu dữ liệu** | **Ràng buộc** | **Ghi chú** |
| 1 | baireadingid | Id bài đọc | int(11) | PRIMARY KEY | AUTO\_INCREMENT |
| 2 | anhbaireading | Ảnh của bài ngữ pháp | varchar(255) | DEFAULT NULL |  |
| 3 | content\_html | Nội dung của bài đọc dưới dạng html | text |  |  |
| 4 | content\_mark\_down | Nội dung của bài đọc dưới dạng markdown | text |  |  |
| 5 | tenbaireading | Tên bài đọc | varchar(255) | DEFAULT NULL |  |

* 1. **MÔ HÌNH USECASE**

#### Xác định các tác nhân và UseCase

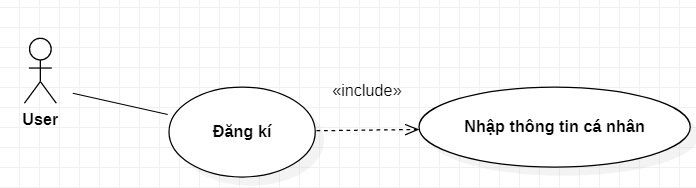
Qua khảo sát hiện trạng thực tế , kết hợp với việc tham khảo tài liệu và phân tích ý kiến của người quản trị , phần mềm học IELTS trực tuyến có các tác nhân sau

|  |  |  |
| --- | --- | --- |
| STT | Tác nhân | UserCase |
| 1 | Admin ,client | UserCase tổng quan |
| 2 | Admin ,client | Đăng kí |
| 3 | Admin ,client | Đăng nhập |
| 4 | Admin | Quản lý người dùng |
| 5 | Admin | Quản lý bài ngữ pháp |
| 6 | Admin | Quản lý bài từ vựng |
| 7 | Admin | Quản lý bài đọc |
| 8 | Admin | Quản lý bài nghe |
| 9 | Client | Xem các bài giảng về ngữ pháp |
| 10 | Client | Xem các bài giảng về từ vựng |
| 11 | Client | Xem các bài giảng về bài đọc |
| 12 | Client | Xem các bài giảng về bài nghe |
| 13 | Client | Tra cứu kết quả quá trình học tập |
| 14 | Client | Làm bài tập bài nghe |
| 15 | Client | Làm bài tập bài đọc |
| 16 | Client | Thi thử |
| 17 |  |  |
| 16 |  |  |

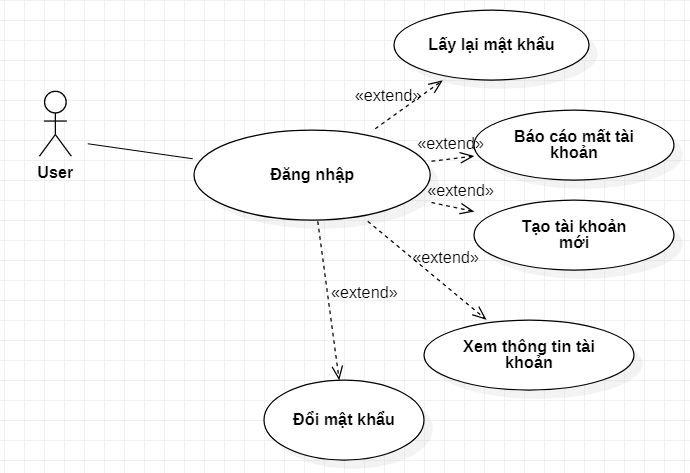
#### Biểu đồ UseCase tổng quan



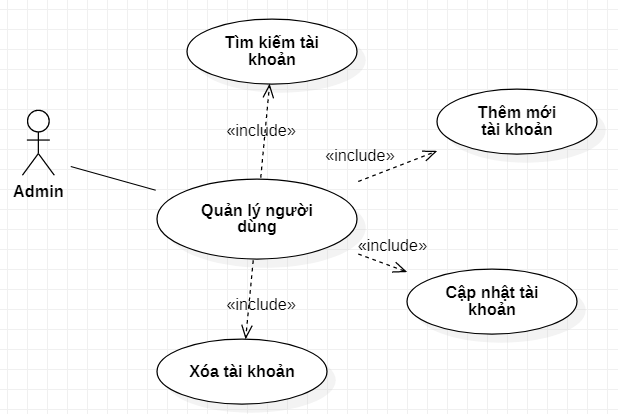
#### UserCase Đăng kí



#### UserCase Đăng nhập



#### UserCase Quản lý người dùng

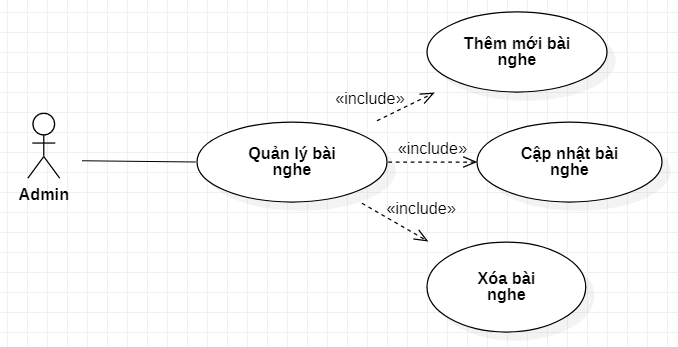


#### UserCase Quản lý bài ngữ pháp

#### UserCase Quản lý bài từ vựng

#### UserCase Quản lý bài đọc

#### UserCase Quản lý bài nghe



#### UserCase Xem các bài giảng từ vựng

#### UserCase Xem các bài giảng ngữ pháp

#### UserCase Xem các bài giảng bài đọc

#### UserCase Xem các bài giảng bài nghe

#### UserCase Làm bài tập từ vựng

#### UserCase Làm bài tập ngữ pháp

#### UserCase Thi thử

#### UserCase Tra cứu kết quả quá trình học tập

* 1. **PHÂN TÍCH THIẾT KẾ THÀNH PHẦN CHỨC NĂNG** 
     1. **BIỂU ĐỒ SEQUENCE DIAGRAM**

#### Admin

Đăng ký, đăng nhập, reset mật khẩu,thay đổi thông tin, đăng xuất.

Quản lý user

Quản lý bài ngữ pháp

Quản lý bài từ vựng

Quản lý bài đọc Ielts

Quản lý bài nghe Ielts

#### Client

Đăng ký, đăng nhập, reset mật khẩu, thay đổi thông tin, đăng xuất.

Xem các bài giảng về ngữ pháp, từ vựng có trên website

Xem các bài giảng bài đọc

Xem các bài giảng bài nghe

Làm bài tập từ vựng

UserCase Làm bài tập ngữ pháp

* 1. **Phân tích thiết kế và đặc tả thành phần giao diện**

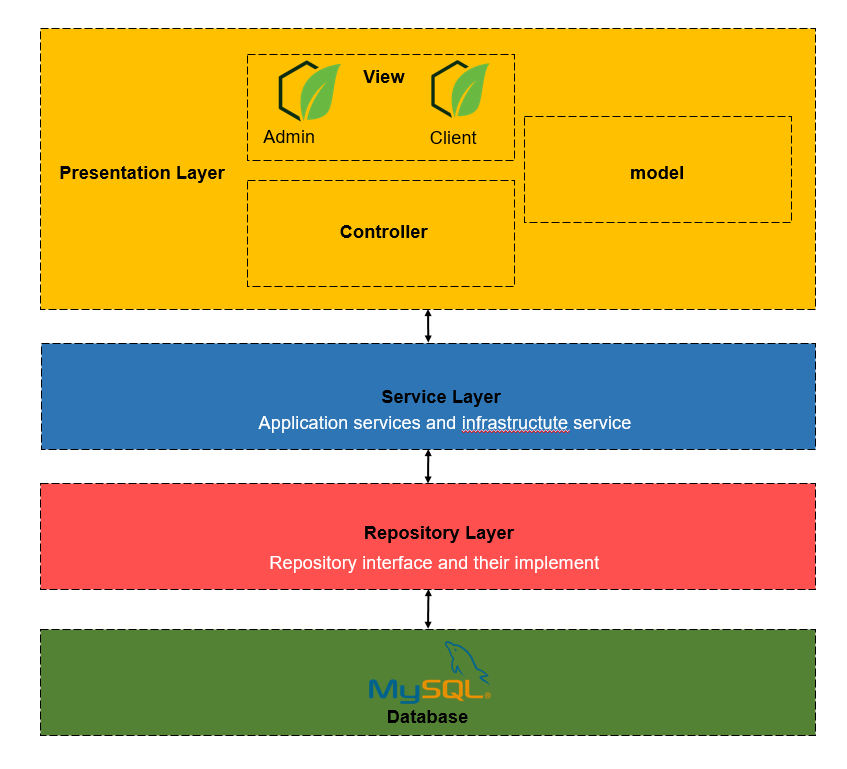
#### Giao diện phía Admin

* Đăng ký, đăng nhập, reset mật khẩu,thay đổi thông tin, đăng xuất.
* Quản lý user
* Quản lý bài ngữ pháp
* Quản lý bài từ vựng
* Quản lý bài đọc Ielts
* Quản lý bài nghe Ielts

#### Giao diện phía Client

* Đăng ký, đăng nhập, reset mật khẩu, thay đổi thông tin, đăng xuất.
* Xem các bài giảng ngữ pháp, từ vựng
* Xem các bài giảng bài đọc
* Xem các bài giảng bài nghe
* Làm bài tập từ vựng
* UserCase Làm bài tập ngữ pháp
  1. **KIẾN TRÚC HỆ THỐNG**

Tổng quan về hoạt động của trang web được mô tả trong hình sau :



Sơ đồ thể hiện các thành phần của trang web:

* Controller: trả về View (có chứa data sẵn, dạng trang HTML), hoặc Model thể hiện dưới dạng API cho View (View viết riêng bằng React, Vue, hoặc Angular).[6]
* Model : các đối tượng được Service tính toán xong trả về cho Controller
* View : View có 2 loại, một là dạng truyền thống là trả về HTML có data. Lúc này Controller sẽ pass dữ liệu vào View và return về (Sử dụng template engine như Thymeleaf làm điều đó).View dạng 2 là dạng View tách riêng (không thuộc về project Spring boot). Thường có trong các hệ thống dùng API. View sẽ được viết riêng bằng Ajax Controller sẽ đưa dữ liệu Model thông qua API cho View, và cũng nhận lại các yêu cầu qua API. [6]
* Service: chứa các code tính toán, xử lý. Khi Controller yêu cầu, thì Service tương ứng sẽ tiếp nhận và cho ra dữ liệu trả cho Controller (trả về Model). Controller sẽ gửi về View như trên.[6]
* Repository: Service còn có thể tương tác với service khác, hoặc dùng Repository để gọi DB. Repository là thằng trực tiếp tương tác, đọc ghi dữ liệu trong DB và trả cho service.[6]
* Database : Sử dụng MySql để lưu trữ dữ liệu

# Chương 4 KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

## KẾT QUẢ ĐẠT ĐƯỢC

## HẠN CHẾ

## HƯỚNG PHÁT TRIỂN

# TÀI LIỆU THAM KHẢO

1. <https://hocspringboot.net/2020/10/03/phan-biet-spring-va-springboot/>
2. <https://hocspringmvc.net/spring-boot-la-gi-nhung-uu-diem-cua-spring-boot/>
3. <https://laptrinhjavaweb.com/springboot>
4. <https://hocspringboot.net/2020/10/17/spring-boot-co-gi-hay-va-nhung-tinh-nang-cua-springboot/>
5. <https://tranngockhoa.blogspot.com/2018/06/automatic-congifuration-trong-spring.html>
6. https://viblo.asia/p/luong-di-trong-spring-boot-ORNZqdELK0n
7. https://viblo.asia/p/dependency-injection-ap-dung-vao-spring-boot-nhu-the-nao-phan-2-3P0lP1wG5ox
8. <https://viblo.asia/p/cau-truc-du-an-spring-boot-the-nao-cho-chuan-ORNZqdwbK0n>
9. https://viblo.asia/p/spring-boot-xu-ly-request-trong-controller-nhu-the-nao-phan-1-gGJ59ANj5X2
10. h[ttps://loda.me/spring-boot-11-huong-dan-spring-boot-jpa-my-sql-loda1558687596060/](https://loda.me/spring-boot-11-huong-dan-spring-boot-jpa-my-sql-loda1558687596060/)
11. <https://www.codehub.com.vn/Hoc-AJAX/Gioi-Thieu>
12. <https://thienanblog.com/javascript/jquery/huong-dan-jquery-ajax-toan-tap/>
13. <https://wiki.matbao.net/mysql-la-gi-huong-dan-toan-tap-ve-mysql/#mysql-la-gi>
14. <https://s.qdht.vn/nap/app/mysql-la-gi-uu-nhuoc-diem-va-tam-can-thiet-cua-mysql.html#qmenu3>
15. <https://ielts-fighter.com/tin-tuc/Gioi-thieu-ve-cong-ty-IMAP-Viet-Nam_mt1537218361.html>
16. <https://chuyendetienganh1.blogspot.com/2020/04/hoc-tieng-anh-hieu-qua-tai-ellloorg.html>
17. <https://bizfly.vn/techblog/top-11-trang-web-hoc-tieng-anh-online-mien-phi-tot-nhat-hien-nay.html>
18. <https://vi.wikipedia.org/wiki/Duolingo>
19. <https://www.ielts-blog.com/about-ielts-blog-and-me/>

# PHỤ LỤC 1: HƯỚNG DẪN TRIỂN KHAI ỨNG DỤNG

1. Anonymous Function hay còn gọi là hàm ẩn danh, là một hàm được sinh ra đúng vào thời điểm chạy của chương trình. Thông thường khi khai báo một hàm thì trình biên dịch sẽ lưu lại trong bộ nhớ nên chúng ta có thể gọi ở trên hay dưới vị trí khai báo đều được, nhưng đối với Anonymous Function thì nó sẽ được sinh ra khi trình biên dịch xử lý tới vị trí của nó/ [↑](#footnote-ref-1)
2. Máy tính cài đặt và chạy phần mềm RDBMS được gọi là client (máy khách). Mỗi khi chúng cần truy cập dữ liệu, chúng kết nối tới máy chủ (server) RDBMS. Cách thức này chính là mô hình “client-server”. [↑](#footnote-ref-2)
3. Với RDBMS là viết tắt của Relational Database Management System. [↑](#footnote-ref-3)