**TRƯỜNG ĐẠI HỌC GIAO THÔNG VẬN TẢI**

**KHOA CÔNG NGHỆ THÔNG TIN**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**



**ĐỒ ÁN TỐT NGHIỆP**

Đề tài

**Xây dựng trang quản lý sinh viên**

Giảng viên hướng dẫn: ThS. Phạm Xuân Tích

Sinh viên thực hiện: Trần Minh Hiệp

Mã sinh viên: 160704493

Lớp: CNTT1

Khóa: 57

**Hà Nội - 01/2021**

# LỜI CẢM ƠN

Đầu tiên, em xin gửi lời cảm ơn chân thành và sự tri ân sâu sắc đối với các thầy cô của trường Đại Học Giao Thông Vận Tải, đặc biệt là các quý Thầy, Cô khoa Công Nghệ Thông Tin của trường, những người đã trực tiếp giảng dạy và truyền đạt những kiến thức bổ ích, kinh nghiệm quý báu cho tôi trong suốt thời gian qua, đó chính là nền tảng cơ bản, là những hành trang vô cùng quý giá cho em bước tới sự nghiệp trong tương lai. Đặc biệt em xin bày tỏ lời cảm ơn sâu sắc đến Thầy. Phạm Xuân Tích, thầy đã tận tình quan tâm giúp đỡ, hướng dẫn em trong suốt quá trình làm đồ án tốt nghiệp. Trong thời gian làm việc với thầy, em đã tiếp thu thêm được nhiều kiến thức bổ ích và học tập được tinh thần làm việc, hiệu quả. Thầy đã giúp em rất nhiều trong việc tìm hiểu công nghệ mà em áp dụng trong đồ án của mình. Nhờ đó mà em đã có thể hoàn thiện đồ án này. Sau cùng, em xin gửi lời cảm ơn chân thành tới gia đình, bạn bè những đã người luôn động viên, đóng góp ý kiến và giúp đỡ tôi vượt qua những khó khăn trong quá trình học tập, nghiên cứu và hoàn thành đồ án tốt nghiệp này cũng như trong cuộc sống.

Sinh viên

Trần Minh Hiệp

# MỤC LỤC

[LỜI CẢM ƠN - 1 -](#_Toc61241878)

[MỤC LỤC - 2 -](#_Toc61241879)

[DANH MỤC BẢNG BIỂU - 4 -](#_Toc61241880)

[DANH MỤC HÌNH ẢNH - 5 -](#_Toc61241881)

[MỞ ĐẦU - 6 -](#_Toc61241882)

[CHƯƠNG 1: TỔNG QUAN - 7 -](#_Toc61241883)

[1.1. Yêu cầu chức năng: - 7 -](#_Toc61241884)

[1.2. Công nghệ sử dụng: - 7 -](#_Toc61241885)

[1.2.1. Ngôn ngữ lập trình Java: - 7 -](#_Toc61241886)

[1.2.2. Spring framework: - 8 -](#_Toc61241887)

[1.2.3. Hibernate Framework - 10 -](#_Toc61241888)

[1.2.4. Hệ quản trị cơ sở dữ liệu MySql: - 12 -](#_Toc61241889)

[1.2.5. Angular: - 12 -](#_Toc61241890)

[CHƯƠNG 2: THIẾT KẾ ỨNG DỤNG - 15 -](#_Toc61241891)

[2.1. Biểu đồ phân rã chức năng: - 15 -](#_Toc61241892)

[2.2. Thiết kế cơ sở dữ liệu: - 16 -](#_Toc61241893)

[2.2.1. Mô hình ERD: - 16 -](#_Toc61241894)

[2.2.2. Chi tiết các bảng: - 17 -](#_Toc61241895)

[2.2.3. Mô tả mỗi quan hệ sinh viên: - 24 -](#_Toc61241896)

[CHƯƠNG 3: XÂY DỰNG VÀ PHÁT TRIỂN - 25 -](#_Toc61241897)

[3.1. Cấu hình Security: - 25 -](#_Toc61241898)

[3.1.1. Server: - 25 -](#_Toc61241899)

[3.1.2. Client: - 27 -](#_Toc61241900)

[3.2. Cấu hình kết nối cơ sở dữ liệu: - 27 -](#_Toc61241901)

[3.3. Entitiy: - 29 -](#_Toc61241902)

[3.4. Data Transfer Object (DTO): - 31 -](#_Toc61241903)

[3.5. Data Access Object (DAO): - 33 -](#_Toc61241904)

[3.6. Service: - 35 -](#_Toc61241905)

[3.7. Rest Controller: - 35 -](#_Toc61241906)

[3.8. Components: - 37 -](#_Toc61241907)

[3.9. Routing: - 38 -](#_Toc61241908)

[CHƯƠNG 4: XÂY DỰNG GIAO DIỆN - 39 -](#_Toc61241909)

[4.1. Giao diện đăng nhập: - 39 -](#_Toc61241910)

[4.2. Giao diện thanh công cụ và menu: - 41 -](#_Toc61241911)

[4.3. Giao diện dánh sách sinh viên: - 41 -](#_Toc61241912)

[4.4. Giao diện điểm học: - 42 -](#_Toc61241913)

[4.5. Giao diện form sinh viên: - 43 -](#_Toc61241914)

[4.6. Giao diện quản lý tài khoản: - 45 -](#_Toc61241915)

[4.7. Giao diện form tài khoản: - 46 -](#_Toc61241916)

[4.8. Giao diện quản lý lớp học phần: - 47 -](#_Toc61241917)

[4.9. Giao diện form lớp học phần: - 47 -](#_Toc61241918)

[CHƯƠNG 5: TỔNG KẾT VÀ ĐÁNH GIÁ - 49 -](#_Toc61241919)

[5.1. Những điểm chương trình thực hiện được: - 49 -](#_Toc61241920)

[5.2. Những điểm chương trình chưa thực hiện được: - 49 -](#_Toc61241921)

[5.3. Hướng phát triển: - 49 -](#_Toc61241922)

# DANH MỤC BẢNG BIỂU

Bảng 2.1. Mô tả chi tiết các trường bảng user ...................................................................17

Bảng 2.2. Mô tả chi tiết các trường bảng user\_infor .........................................................17

Bảng 2.3. Mô tả chi tiết các trường bảng students ............................................................18

Bảng 2.4. Mô tả chi tiết các trường bảng study\_score ......................................................19

Bảng 2.5. Mô tả chi tiết các trường bảng category ............................................................20

Bảng 2.6. Mô tả chi tiết các trường bảng role ...................................................................20

Bảng 2.7. Mô tả chi tiết các trường bảng permission ........................................................20

Bảng 2.8. Mô tả chi tiết các trường bảng class ..................................................................21

Bảng 2.9. Mô tả chi tiết các trường bảng class\_detail .......................................................21

Bảng 2.10. Mô tả chi tiết các trường bảng subjects ...........................................................22

Bảng 2.11. Mô tả chi tiết các trường bảng unit .................................................................22

Bảng 2.12. Mô tả chi tiết các trường bảng department .....................................................23

Bảng 2.13. Mô tả chi tiết các trường bảng course .............................................................23

# DANH MỤC HÌNH ẢNH

Hình 1.1. Mô hình kiến trúc Spring Framework .................................................................9

Hình 1.2. Mô hình kiến trúc Hibernate Framework ..........................................................11

Hình 2.1. Biểu đồ phân rã chức năng ................................................................................15

Hình 2.2. Mô hình ERD .....................................................................................................16

Hình 2.3. Mô hình ERD quan hệ sinh viên .......................................................................24

Hình 3.1. Các thông tin cấu hình database và Hibernate trong file application.yml .........28

Hình 3.2. Package chứa các Entity ....................................................................................29

Hình 3.3. Package chứa các đối tượng DTO .....................................................................31

Hình 3.4. Package chứa các lớp DAO ...............................................................................34

Hình 3.5. Package chứa các lớp Service ............................................................................35

Hình 3.6. Package chứa các Restcontroller .......................................................................36

Hình 3.7. Cấu trúc ứng dụng Angular - component ..........................................................37

Hình 4.1. Trang đăng nhập ................................................................................................39

Hình 4.2. Trang đăng nhập – báo lỗi khi thiếu thông tin ...................................................40

Hình 4.3. Trang đăng nhập – báo lỗi khi sai thông tin ......................................................40

Hình 4.4. Thanh công cụ ...................................................................................................41

Hình 4.5. Trang quản lý sinh viên .....................................................................................42

Hình 4.6. Trang điểm học ..................................................................................................43

Hình 4.7. Trang thông tin sinh viên – phần thông tin bố mẹ .............................................43

Hình 4.8. Trang thông tin sinh viên – phần thông tin cá nhân ..........................................44

Hình 4.9. Menu user đăng nhập .........................................................................................45

Hình 4.10. Trang quản lý tài khoản ...................................................................................46

Hình 4.11. Form tạo tài khoản ...........................................................................................46

Hình 4.12. Trang quản lý lớp học phần .............................................................................47

Hình 4.13. Trang tạo/thông tin lớp học phần .....................................................................48

# MỞ ĐẦU

Ngày nay công nghệ thông tin đã có những bước tiến phát triển mạnh mẽ về cả chiều sâu và chiều rộng. Máy tính điện tử giờ đây không còn là thứ quý hiếm mà đang ngày càng trở thành một công cụ làm việc và giải trí của con người. Những năm gần đây, việc cập nhật các thông tin cũng như thực hiện việc quản lý thông tin qua các hệ thống trực tuyến ngày càng trở nên phổ biến. Nó giúp cho công việc quản lý thông tin trở nên dễ dàng hơn, giảm thiểu thời gian tra cứu, chi phí vầ lưu trữ thông tin được lâu dài hơn.

Nhận thấy sự thiết thực ấy, em đã lựa chọn thực hiện xây dựng “Website quản lý sinh viên sử dụng ngôn ngữ Java kết hợp với framework Spring boot và Angular 8” để báo cáo trong kì đồ án tốt nghiệp nhằm đáp ứng một cách đầy đủ nhất cho những điều ở trên.

# CHƯƠNG 1: TỔNG QUAN

## Yêu cầu chức năng:

Hệ thống quản lý sinh viên cần có những chức năng về quản lý thông tin, điểm học, lớp học sinh viên tham gia. Ngoài ra còn có những chức năng quản lý những mục liên quan như môn học, v.v.... Một số chức năng dự kiến:

* Có thể xem danh sách sinh viên.
* Thực hiện tìm kiếm sinh viên.
* Có thể thêm hoặc sửa thông tin sinh viên.
* Xóa sinh viên ra khỏi hệ thống.
* Phân quyền các tài khoản truy cập các mục.
* Có thể xem danh sách các lớp học phần.
* Tìm kiếm lớp học phần.
* Thực hiện thêm hoặc sửa thông tin lớp học.
* Hệ thống có đa ngôn ngữ.
* Xóa lớp học ra khỏi hệ thống.
* Chuyển lớp hoặc xóa sinh viên ra khỏi một lớp học.
* Tạo tài khoản đăng nhập hệ thống.

## 1.2. Công nghệ sử dụng:

### 1.2.1. Ngôn ngữ lập trình Java:

**Java** là một nền tảng phát triển các ứng dụng [phần mềm](https://vi.wikipedia.org/wiki/Ph%E1%BA%A7n_m%E1%BB%81m) có vị trí rất lớn trong những năm cuối [thế kỉ 20](https://vi.wikipedia.org/wiki/Th%E1%BA%BF_k%E1%BB%B7_20), đầu [thế kỉ 21](https://vi.wikipedia.org/wiki/Th%E1%BA%BF_k%E1%BB%B7_21). Đánh dấu sự trưởng thành của mô hình lập trình hướng đối tượng, nó được coi là một nền tảng mang tính cách mạng trong ngành phần mềm. Mô hình máy ảo [Virtual Machine](https://vi.wikipedia.org/w/index.php?title=Virtual_Machine&action=edit&redlink=1) đã cho phép các ứng dụng viết bằng Java có thể chạy trên nhiều hệ điều hành khác nhau.

Ban đầu, Java chủ yếu dùng để phát triển các [applet](https://vi.wikipedia.org/w/index.php?title=Applet&action=edit&redlink=1), các ứng dụng nhúng vào trình duyệt, góp phần làm sinh động các [trang web](https://vi.wikipedia.org/wiki/Website) tĩnh vốn hết sức tẻ nhạt. Tuy nhiên, cùng với sự phát triển của công nghệ thông tin và nhu cầu của xã hội, Java applet đã dần mất đi vị trí của nó và thay vào đó, các công ty, cộng đồng ủng hộ Java đã phát triển nó theo một hướng khác. Hiện nay, công nghệ Java được chia làm ba nhánh:

* [**J2SE**](https://vi.wikipedia.org/wiki/J2SE)**:** Gồm các đặc tả, công cụ, [API](https://vi.wikipedia.org/wiki/Giao_di%E1%BB%87n_l%E1%BA%ADp_tr%C3%ACnh_%E1%BB%A9ng_d%E1%BB%A5ng) của nhân Java giúp phát triển các ứng dụng trên desktop và định nghĩa các phần thuộc nhân của Java.
* [**J2EE**](https://vi.wikipedia.org/wiki/J2EE)**:** Gồm các đặc tả, công cụ, API mở rộng J2SE để phát triển các ứng dụng quy mô xí nghiệp, chủ yếu để chạy trên máy chủ (server). Bộ phận hay được nhắc đến nhất của công nghệ này là công nghệ Servlet/[JSP](https://vi.wikipedia.org/wiki/JSP): sử dụng Java để làm các ứng dụng web.
* [**J2ME**](https://vi.wikipedia.org/wiki/J2ME)**:** Gồm các đặc tả, công cụ, API mở rộng để phát triển các ứng dụng Java chạy trên điện thoại di động, thẻ thông minh, thiết bị điện tử cầm tay, robo và những ứng dụng điện tử khác

Java bao gồm các bộ phận:

* Máy ảo Java: JVM
* Bộ công cụ phát triển: J2SDK
* Các đặc tả chi tiết kĩ thuật (specifications)
* Ngôn ngữ lập trình (programming language)
* Có 6 mục tiêu chính trong việc xây dựng ngôn ngữ Java:
* Đơn giản, hướng đối tượng và quen thuộc.
* Mạnh mẽ và an toàn.
* Kiến trúc trung lập và di động.
* Thực thi với hiệu suất cao.
* Dịch ra bytecode, phân luồng và năng động.
* Dễ sử dụng cho người dùng Java.

### 1.2.2. Spring framework:

Spring là một Framework phát triển các ứng dụng Java được sử dụng bởi hàng triệu lập trình viên. Nó giúp tạo các ứng dụng có hiệu năng cao, dễ kiểm thử, sử dụng lại code…

Spring nhẹ và trong suốt (nhẹ: kích thước nhỏ, version cơ bản chỉ khoảng 2MB; trong suốt: hoạt động một cách trong suốt với lập trình viên)

Spring là một mã nguồn mở, được phát triển, chia sẻ và có cộng đồng người dùng rất lơn.

Spring Framework được xây dựng dựa trên 2 nguyên tắc design chính là: [Dependency Injection](https://stackjava.com/design-pattern/dependency-injection-di-la-gi.html) và Aspect Oriented Programming.

Những tính năng core (cốt lõi) của Spring có thể được sử dụng để phát triển Java Desktop, ứng dụng mobile, Java Web. Mục tiêu chính của Spring là giúp phát triển các ứng dụng J2EE một cách dễ dàng hơn dựa trên mô hình sử dụng POJO (Plain Old Java Object)

Spring được chia làm nhiều module khác nhau, tùy theo mục đích phát triển ứng dụng mà ta dùng 1 trong các module đó.

Kiến trúc tổng thể của Spring Framework:



Hình 1.1. Mô hình kiến trúc Spring Framework

* **Test:** Tầng này cung cấp khả năng hỗ trợ kiểm thử với JUnit và TestNG.
* **Spring Core Container:** Bao gồm các module spring core, beans, context và expression languate (EL)
* Spring core, bean cung cấp tính năng [IOC và Dependency Injection](https://stackjava.com/design-pattern/dependency-injection-di-la-gi.html).
* Spring Context hỗ trợ đa ngôn ngữ (internationalization), các tính năng Java EE như EJB, JMX.
* Expression Language được mở rộng từ Expresion Language trong JSP. Nó cung cấp hỗ trợ việc setting/getting giá trị, các method cải tiến cho phép truy cập collections, index, các toán tử logic…
* **Web:** gồm Web, Web-Servlet… hỗ trợ việc tạo ứng dụng web, gọi là Spring MVC.
* **Data Access/Integration:** gồm JDBC, ORM, OXM, JMS và module Transaction, những module này cung cấp khả năng giao tiếp với database.
* **AOP, Aspects, Instrumentation:** Những module này hỗ trợ cài đặt lập trình hướng khía cạnh (Aspect Oriented Programming), hỗ trợ tích hợp với AspectJ.

Các lợi ích của Spring:

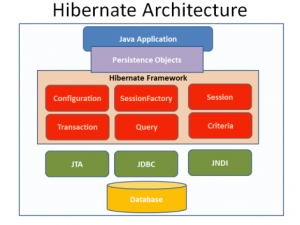
* Spring cho phép lập trình viên sử dụng POJOs. Việc sử dụng POJOs giúp không phải làm việc với EJB, ứng dụng, các luồng chạy, cấu hình… đơn giản hơn rất nhiều.
* Spring được tổ chức theo kiểu mô đun. Số lượng các gói và các lớp khá nhiều, nhưng chỉ cần quan tâm đến những gì cần và không cần quan tâm đến phần còn lại.
* Spring hỗ trợ sử dụng khá nhiều công nghệ như [ORM Framework](https://stackjava.com/uncategorized/orm-la-gi-tong-quan-ve-orm-framework.html), các logging framework, JEE, các thư viện tạo lịch trình (Quartz và JDK timer)…
* Module Web của Spring được thiết kế theo mô hình MVC nên nó cung cấp đầy đủ các tính năng giúp thay thế các web framework khác như Struts.

### 1.2.3. Hibernate Framework

Hibernate là 1 [ORM Framework](https://stackjava.com/uncategorized/orm-la-gi-tong-quan-ve-orm-framework.html): thực hiện mapping cơ sở dữ liệu quan hệ sang các object trong ngôn ngữ hướng đối tượng. Hibernate là 1 Framework cho persistence layer: Thực hiện giao tiếp giữa tầng ứng dụng với tầng dữ liệu (kết nối, truy xuất, lưu trữ…)

Kiến trúc hibernate framework:

* Mỗi table trong database là một object trong Hibernate. Do đó, bạn cần có một java bean cho mỗi table trong database. Các java bean này sẽ có các getters/setters và một số ngoại lệ theo quy ước của Hibernate.
* Mỗi file mapping (ánh xạ) có dạng \*\*\*.hbm.xml có nhiệm vụ đặc tả mỗi quan hệ giữa các thuộc tính của đối tượng và các trường trong bảng CSDL (cột nào ứng với thuộc tính nào của class, kiểu giá trị là gì, quan hệ giữa các bảng chuyển thành quan hệ giữa các lớp ra sao…).



Hình 1.2. Mô hình kiến trúc Hibernate Framework

* File Hibernate.cfg.xml: Đây là tập tin được load đầu tiên khi khởi chạy ứng dụng Hibernate. Nó chứa các thông tin về kết nối (database kết nối là gì (MySQL, Oracle, SQL server…), username và password kết nối với database là gì, tên database được kết nối), driver kết nối.

Ưu điểm của Hibernate Framework:

* Dễ sử dụng: dễ dàng quản lí các kết nối database và dễ fix bug, cung cấp sẵn nhiều API truy vấn.
* Tính độc lập: không cần quan tâm tới cơ sở dữ liệu sử dụng khi viết câu lệnh SQL.
* Tính hướng đối tượng: tập trung xử lý theo hướng đối tượng, phù hợp sử dụng trong các case CRUD (Create, Read, Update, Delete).
* Tính tin cậy: Hibernate đã được kiểm thử và khá an toàn trong các truy vấn.

Nhược điểm của Hibernate Framework:

* Không hỗ trợ các câu truy vấn phức tạp.
* Một số trường hợp vẫn phải dùng native SQL do Hibernate không thể cover hết tất cả các cú pháp của các hệ quản trị cơ sử dữ liệu.
* Bị hạn chế sự can thiệp vào câu lệnh SQL do nó được tự động sinh ra.

### 1.2.4. Hệ quản trị cơ sở dữ liệu MySql:

**MySQL** là [hệ quản trị cơ sở dữ liệu](https://vi.wikipedia.org/wiki/H%E1%BB%87_qu%E1%BA%A3n_tr%E1%BB%8B_c%C6%A1_s%E1%BB%9F_d%E1%BB%AF_li%E1%BB%87u) [mã nguồn mở](https://vi.wikipedia.org/wiki/Ph%E1%BA%A7n_m%E1%BB%81m_ngu%E1%BB%93n_m%E1%BB%9F) phổ biến nhất thế giới và được các nhà phát triển rất ưa chuộng trong quá trình phát triển ứng dụng. Vì MySQL là hệ quản trị cơ sở dữ liệu tốc độ cao, ổn định và dễ sử dụng, có tính khả chuyển, hoạt động trên nhiều hệ điều hành cung cấp một hệ thống lớn các hàm tiện ích rất mạnh. Với tốc độ và tính bảo mật cao, MySQL rất thích hợp cho các ứng dụng có truy cập CSDL trên internet.

Máy chủ cơ sở dữ liệu MySQL đáp ứng nhiều tính năng linh hoạt, nó có sức chứa để xử lý các ứng dụng được nhúng sâu với 1MB dung lượng để chạy kho dữ liệu khổng lồ lên đến hàng terabytes thông tin. Đặc tính này của MySQL là sự linh hoạt về flatform với tất cả các phiên bản của Windows, Unix và Linux đang được hỗ trợ. Và đương nhiên, tính chất mã nguồn mở của MySQL cho phép tùy biến theo ý muốn để thêm các yêu cầu phù hợp cho database server. Với kiến trúc storage-engine cho phép các chuyên gia cơ sở dữ liệu cấu hình máy chủ cơ sở dữ liệu MySQL đặc trưng cho các ứng dụng chuyên biệt. Dù ứng dụng là website dung lượng lớn phục vụ hàng triệu người/ngày hay hệ thống xử lý giao dịch tốc độ cao thì MySQL đều đáp ứng được khả năng xử lý khắt khe của mọi hệ thống. MySQL đảm bảo sự tin cậy và có thể sử dụng ngay. MySQL đưa ra nhiều tùy chọn có thể dùng ngay từ cấu hình tái tạo server/client tốc độ cao, để các nhà phân phối thứ 3 có thể đưa ra những điều hướng có thể dùng ngay duy nhất cho server cơ sở dữ liệu MySQL. MySQL là nơi cho các website trao đổi thường xuyên bởi nó có engine xử lý tốc độ cao, khả năng chèn dữ liệu cực nhanh và hỗ trợ mạnh các chức năng chuyên dụng của web. Các tính năng này cũng được sử dụng cho môi trường lưu trữ dữ liệu mà MySQL tăng cường đến hàng terabyte cho các server đơn. Những tính năng khác như chỉ số băm, bảng nhớ chính, bảng lưu trữ và cây nhị phân đã được gói lại để giảm các yêu cầu lưu trữ. MySQL tích hợp các tính năng bảo mật an toàn tuyệt đối. Với việc xác nhận truy cập cơ sở dữ liệu, MySQL trang bị các kỹ thuật mạnh, chỉ có người sử dụng đã được xác nhận mới truy cập được vào cơ sở dữ liệu. SSH và SSL cũng được hỗ trợ để đảm bảo kết nối an toàn và bảo mật. Tiện ích backup và recovery cung cấp bởi MySQL và các hãng phần mềm thứ 3 cho phép backup logic và vật lý cũng như recovery toàn bộ hoặc tại một thời điểm.

### 1.2.5. Angular:

Angular là một Javascript Framework được phát triển bởi google để xây dựng các Single Page Application (SPA) bằng JavaScript, HTML và TypeScript. Angular cung cấp các tính năng tích hợp cho animation, http service, materials và có các tính năng như auto-complete, navigation, toolbar, menus, … Code được viết bằng TypeScript, biên dịch thành JavaScript và hiển thị giống nhau trong trình duyệt.

Phiên bản đầu tiền của angular là AngularJS được bắt đầu từ năm 2009 và đc ra mắt vào 20/10/2010 bởi lập trình viên Misko Hevery tại Google. Lúc đó angular js được viết theo mô hình MVC. Tháng 3 năm 2015 phiên bản bản angular 2 ra đời nhằm thay thế Angular Js với các khái niệm mới nhằm đơn giản hóa và tối ưu cho quá trình phát triển sử dụng framework này. Angular 2 thay đổi hoàn toàn so với angular js bằng việc thay Controllers và $scope (Angular js) bằng components và directives. Components = directives + template, tạo nên view của ứng dụng và xử lí các logic trên view. Angular 2 hoàn toàn được viết bằng Typescript. Angular 2 nhanh hơn angular js, hỗ trợ đa nền tảng đa trình duyệt, cấu trúc cdoe được tổ chức đơn giản và dễ sử dụng hơn. Angular 4 ra mắt vào tháng 3/2017 đây là một phiên bản nâng cấp từ Angular 2 nên kiến trúc không thay đổi nhiều ngoài việc giảm thiểu code được tạo ra từ đó giảm kích thước tệp được đóng gói, đẩy nhanh quá trình phát triển ứng dụng. Angular 5 được phát hành vào 1/11/2017 với mục tiêu thay đổi về tốc độ và kích thước nên nó nhanh hơn và nhỏ hơn angular 4. Các tính năng mới so với angular 4:

* **Sử dụng HTTPClient**thay vì sử dụng HTTP: bởi vì nó nhanh, an toàn và hiệu quả hơn.
* Với phiên bản Angular 5 mặc định sử dụng RxJs 5.5
* **Multiple export aliases**: Một component có thể được xuất bằng nhiều bí danh (aliases) để giảm bớt quá trình di chuyển.
* **Internationalized Pipes for Number, Date, and Currency:** Các pipe mới được giới thiệu để tiêu chuẩn hóa tốt hơn.
* Tối ưu hóa build production bằng việc sử dụng công cụ build optimizer được tích hợp sẵn vào trong CLI. Công cụ này tối ưu tree shark và loại bỏ code dư thừa.
* Cải thiện tốc độ biên dịch bằng việc dùng TypeScript transforms, giờ đây khi build sẽ sử dụng lệnh “ng serve –aot”. AOT sẽ cải thiện performace khi load page và nó được dùng để deploy app lên production.

Angular 6 được phát hành vào 4/5/2018 với những thay đổi so với Angular 5:

* **Cập nhật CLI, command line interface**: thêm 1 số lệnh mới như ng-update để chuyển từ version trước sang version hiện tại; ng-add để thêm các tính năng của ứng dụng để trở thành một ứng dụng web tiến bộ.
* **Angular Element:** Cho phép các component của **Angular** được triển khai dưới dạng component web, sau đó có thể được sử dụng trong bất kỳ trang HTML nào một cách dễ dàng.
* **Multiple Validators:** cho phép nhiều Validators được áp dụng trên form builder.
* **Tree-shakeable providers:** giúp loại bỏ mã code chết.
* Sử dụng RxJS 6 với syntax thay đổi.

Angular 7 được phát hành vào 18/10/2018 với những thay đổi như:

* Hỗ trợ TypeScript 3.1
* ScrollingModule: Để scroll load dữ liệu.
* Drag and Drop: Chúng ta có thể dễ dàng thêm tính năng kéo và thả vào một mục.
* Angular 7.0 đã cập nhật RxJS 6.3.

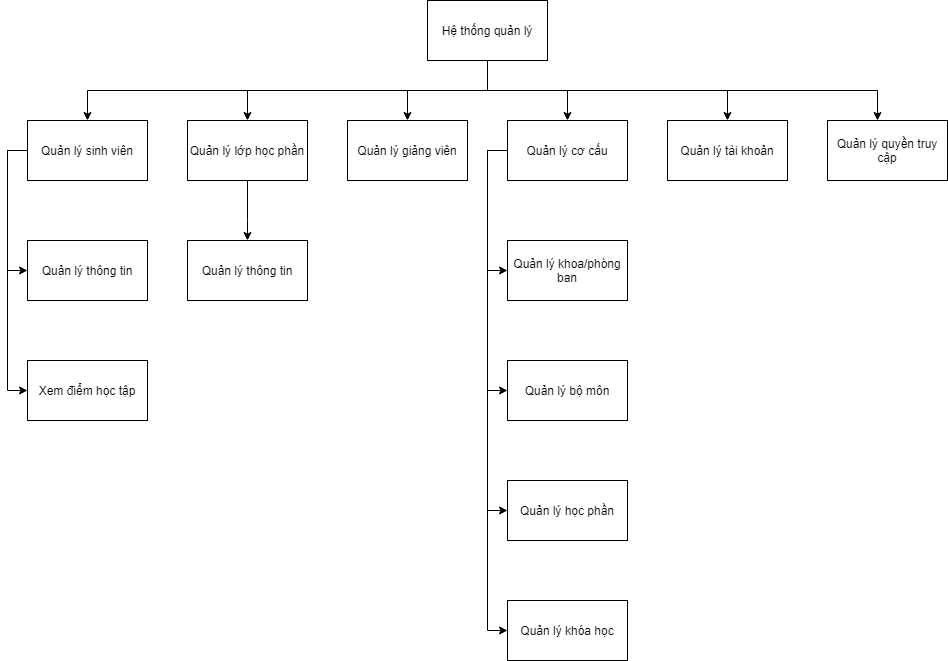
Angular 8 được phát hành vào 3/4/2019, được nâng cấp hỗ trợ TypeScript 3.4.

Angular được load với tính năng Power-packaged. Một số tính năng như:

* Cơ chế Two-Way Data Binding: Đây là tính năng cool nhất của Angular. Data binding tự động và rất nhanh tức là bất cứ thay đổi nào trên view đều được tự động cập nhật vào component class và ngược lại.
* Hỗ trợ cơ chế Routing mạnh mẽ: Angular có cơ chế routing tải trang một cách bất đồng bộ trên cùng một trang cho phép chúng ta tạo SPA.
* Mở rộng HTML: Angular cho phép chúng ta sử dụng cấu trúc lập trình giống như điều kiện if, vòng lặp for...để render các control.
* Thiết kế module hoá: Angular tiếp cận theo hướng thiết kế module hoá. Bạn phải tạo các Angular Module để tổ chức tốt hơn và quản lý source code.
* Hỗ trợ làm việc với hệ thống Backend: Angular được xây dựng hỗ trợ làm việc với backend server và thực thi bất cứ logic nào và nhận dữ liệu về.
* Cộng đồng tốt: Angular được hỗ trợ bởi Google và cộng đồng.

# CHƯƠNG 2: THIẾT KẾ ỨNG DỤNG

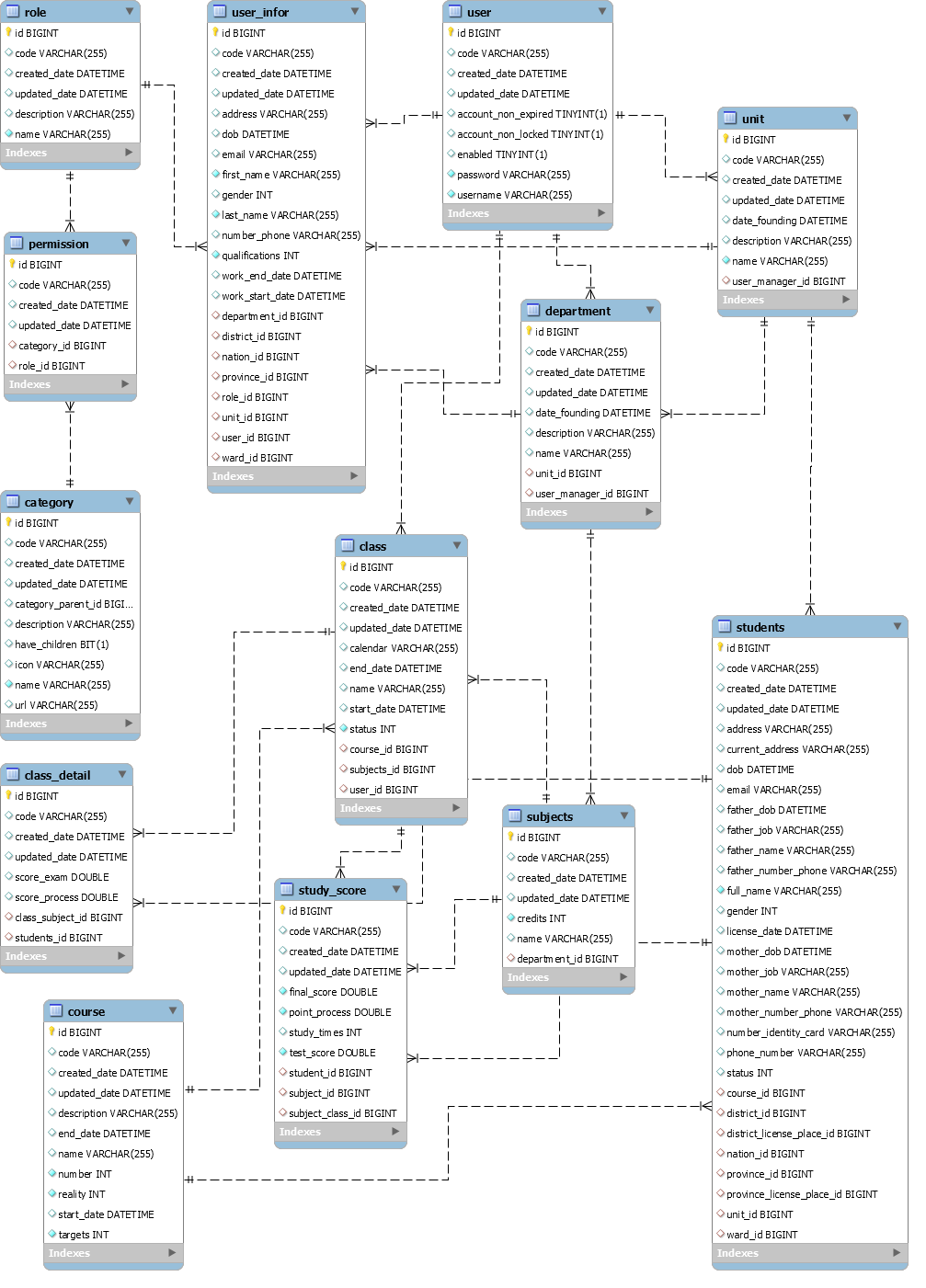
## 2.1. Biểu đồ phân rã chức năng:



Hình 2.1. Biểu đồ phân rã chức năng

## 2.2. Thiết kế cơ sở dữ liệu:

### 2.2.1. Mô hình ERD:



Hình 2.2. Mô hình ERD

### 2.2.2. Chi tiết các bảng:

#### 2.2.2.1. Bảng user

|  |  |  |  |
| --- | --- | --- | --- |
| **Tên cột** | **Kiểu dữ liệu** | **Giá trị mặc định** | **Mô tả** |
| id | Bigint | Tự tăng | Khóa chính của bảng |
| code | Varchar(50) | null | Mã tài khoản người dùng |
| created\_date | Datetime | Null | Ngày tạo bản ghi |
| updated\_date | Datetime | Null | Ngày cập nhật bản ghi |
| account\_non\_locked | Tinyint | 1 | Tài khoản không bị khóa |
| account\_non\_expired | Tinyint | 1 | Tài khoản không hết hạn |
| enabled | Tinyint | 1 | Tài khoản đang được dùng |
| password | Varchar(50) |  | Mật khẩu |
| username | Varchar(50) |  | Tên đăng nhập |

Bảng 2.1. Mô tả chi tiết các trường bảng user

#### 2.2.2.2. Bảng user\_infor (thông tin tài khoản người dùng)

|  |  |  |  |
| --- | --- | --- | --- |
| **Tên cột** | **Kiểu dữ liệu** | **Giá trị mặc định** | **Mô tả** |
| id | Bigint | Tự tăng | Khóa chính của bảng |
| code | Varchar(50) | null | Mã tài khoản người dùng |
| created\_date | Datetime | Null | Ngày tạo bản ghi |
| updated\_date | Datetime | Null | Ngày cập nhật bản ghi |
| adress | Varchar(255) | Null | Địa chỉ |
| dob | Datetime | Null | Ngày sinh |
| email | Varchar(50) | Null | Địa chỉ email |
| gender | Int | 2 | Giới tính: 0 – nam, 1 – nữ, 2 - khác |
| number\_phone | Varchar(50) | Null | Số điện thoại |
| role\_id | Bigint |  | Id của chức danh |
| first\_name | Varchar(50) |  | Họ |
| last\_name | Varchar(50) |  | Tên |
| unit\_id | Bigint |  | Id khoa / phòng ban |
| user\_id | Bigint |  | Id tài khoản đăng nhập |
| qualifications | Int | Null | Trình độ chuyên môn |
| work\_start\_date | Datetime | Null | Ngày bắt đầu làm việc |
| work\_end\_date | Datetime | Null | Ngày kết thúc làm việc |
| nation\_id | Bigint | Null | Quốc tịch |
| district\_id | Bigint | Null | Quận/huyện/thành phố |
| proince\_id | Bigint | Null | Tỉnh/thành phố |
| ward\_id | Bigint | Null | Phường/xã |
| department\_id | Bigint | Null | Id bộ môn |

Bảng 2.2. Mô tả chi tiết các trường bảng user\_infor

#### 2.2.2.3. Bảng students

|  |  |  |  |
| --- | --- | --- | --- |
| **Tên cột** | **Kiểu dữ liệu** | **Giá trị mặc định** | **Mô tả** |
| id | Bigint | Tự tăng | Khóa chính của bảng |
| code | Varchar(50) | null | Mã tài khoản người dùng |
| created\_date | Datetime | Null | Ngày tạo bản ghi |
| updated\_date | Datetime | Null | Ngày cập nhật bản ghi |
| address | Varchar(255) | Null | Địa chỉ |
| dob | Datetime | Null | Ngày sinh |
| email | Varchar(50) | Null | Địa chỉ email |
| phone\_number | Varchar(50) | Null | Số điện thoại |
| nation\_id | Bigint | Null | Quốc tịch |
| father\_dob | Datetime | Null | Ngày sinh của bố |
| father\_name | Varchar(50) | Null | Tên cuả bố |
| father\_job | Varchar(50) | Null | Nghề nghiệp của bố |
| father\_number\_phone | Varchar(50) | Null | Số điện thoại của bố |
| mother\_dob | Datetime | Null | Ngày sinh của mẹ |
| mother\_name | Varchar(50) | Null | Tên cuả mẹ |
| mother\_job | Varchar(50) | Null | Nghề nghiệp của mẹ |
| mother\_number\_phone | Varchar(50) | Null | Số điện thoại của mẹ |
| license\_date | Datetime | Null | Ngày cấp CMND/CCCD/hộ chiếu |
| number\_identity\_card | Varchar(50) | Null | Số CMND/CCCD/hộ chiếu |
| district\_id | Bigint | Null | Quận/huyện/thành phố |
| district\_license\_place\_id | Bigint | Null | Quận/huyện/thành phố - nơi cấp CMND/CCCD |
| province\_id | Bigint | Null | Tỉnh/thành phố |
| province\_license\_place\_id | Bigint | Null | Tỉnh/thành phố - nơi cấp CMND/CCCD |
| ward\_id | Bigint | Null | Phường xã |
| status | Int | 1 | Trạng thái học: 0 – nghỉ học, 1 – đang học, 2 – hoàn thành |
| gender | Int | 2 | Giới tính: 0 – nam, 1 – nữ, 2 - khác |
| current\_address | Varchar(255) | Null | Địa chỉ hiện tại |
| course\_id | Bigint |  | Khóa học |
| unit\_id | Bigint |  | Khoa |
| full\_name | Varchar(50) |  | Tên đầy đủ |

Bảng 2.3. Mô tả chi tiết các trường bảng students

#### 2.2.2.4. Bảng study\_score (điểm sinh viên)

|  |  |  |  |
| --- | --- | --- | --- |
| **Tên cột** | **Kiểu dữ liệu** | **Giá trị mặc định** | **Mô tả** |
| id | Bigint | Tự tăng | Khóa chính của bảng |
| code | Varchar(50) | null | Mã tài khoản người dùng |
| created\_date | Datetime | Null | Ngày tạo bản ghi |
| updated\_date | Datetime | Null | Ngày cập nhật bản ghi |
| final\_score | Double | Null | Điểm tổng kết học phần |
| point\_process | Double |  | Điểm quá trình |
| study\_times | Int |  | Lần học |
| test\_score | Double |  | Điểm thi |
| student\_id | Bigint |  | Id của sinh viên |
| subject\_id | Bigint |  | Id của học phần |
| subject\_class\_id | Bigint |  | Id cảu lớp học phần |

Bảng 2.4. Mô tả chi tiết các trường bảng study\_score

#### 2.2.2.5. Bảng category (danh mục)

|  |  |  |  |
| --- | --- | --- | --- |
| **Tên cột** | **Kiểu dữ liệu** | **Giá trị mặc định** | **Mô tả** |
| id | Bigint | Tự tăng | Khóa chính của bảng |
| code | Varchar(50) | null | Mã tài khoản người dùng |
| created\_date | Datetime | Null | Ngày tạo bản ghi |
| updated\_date | Datetime | Null | Ngày cập nhật bản ghi |
| Description | Varchar(255) | Null | Mô tả danh mục |
| Name | Varchar(50) |  | Tên danh mục |
| url | Varchar(50) |  | Đường dẫn truy cập |
| Category\_parent\_id | Bigint | Null | Id danh mục cha |
| Have\_children | Tinyint | Null | Có danh mục con |
| icon | Varchar(50) | Null | Biểu tượng |

Bảng 2.5. Mô tả chi tiết các trường bảng category

#### 2.2.2.6. Bảng role (chức danh)

|  |  |  |  |
| --- | --- | --- | --- |
| **Tên cột** | **Kiểu dữ liệu** | **Giá trị mặc định** | **Mô tả** |
| id | Bigint | Tự tăng | Khóa chính của bảng |
| code | Varchar(50) |  | Mã tài khoản người dùng |
| created\_date | Datetime | Null | Ngày tạo bản ghi |
| updated\_date | Datetime | Null | Ngày cập nhật bản ghi |
| name | Varchar(50) |  | Tên chức danh |
| description | Varchar(255) |  | Mô tả chức danh |

Bảng 2.6. Mô tả chi tiết các trường bảng brand

#### 2.2.2.7. Bảng permission (quyền truy cập danh mục ứng với chức danh)

|  |  |  |  |
| --- | --- | --- | --- |
| **Tên cột** | **Kiểu dữ liệu** | **Giá trị mặc định** | **Mô tả** |
| id | Bigint | Tự tăng | Khóa chính của bảng |
| code | Varchar(50) | null | Mã tài khoản người dùng |
| created\_date | Datetime | Null | Ngày tạo bản ghi |
| updated\_date | Datetime | Null | Ngày cập nhật bản ghi |
| role\_id | Bigint |  | Id chức danh |
| category\_id | Bigint |  | Id danh mục |

Bảng 2.7. Mô tả chi tiết các trường bảng permission

#### 2.2.2.8. Bảng class (lớp học phần)

|  |  |  |  |
| --- | --- | --- | --- |
| **Tên cột** | **Kiểu dữ liệu** | **Giá trị mặc định** | **Mô tả** |
| id | Bigint | Tự tăng | Khóa chính của bảng |
| code | Varchar(50) | null | Mã tài khoản người dùng |
| created\_date | Datetime | Null | Ngày tạo bản ghi |
| updated\_date | Datetime | Null | Ngày cập nhật bản ghi |
| calendar | Varchar(255) | Null | Lịch học |
| end\_date | Datetime | Null | Ngày kết thúc |
| start\_date | Datetime | Null | Ngày bắt đầu |
| name | Varchar(50) |  | Tên lớp học phần |
| status | Int | 0 | Trạng thái: 0 – chuẩn bị, 1 – đang học, 2 – kết thúc |
| course\_id | Bigint |  | Khóa học |
| subjects\_id | Bigint |  | Học phần |
| user\_id | Bigint | Null | Giảng viên phụ trách |

Bảng 2.8. Mô tả chi tiết các trường bảng class

#### 2.2.2.9. Bảng class\_detail

|  |  |  |  |
| --- | --- | --- | --- |
| **Tên cột** | **Kiểu dữ liệu** | **Giá trị mặc định** | **Mô tả** |
| id | Bigint | Tự tăng | Khóa chính của bảng |
| code | Varchar(50) | null | Mã tài khoản người dùng |
| created\_date | Datetime | Null | Ngày tạo bản ghi |
| updated\_date | Datetime | Null | Ngày cập nhật bản ghi |
| Score\_exam | Double |  | Điểm thi |
| Score\_process | Double |  | Điểm quá trình |
| Class\_subject\_id | Bigint |  | Lớp học phần |
| Student\_id | Bigint |  | Id sinh viên |

Bảng 2.9. Mô tả chi tiết các trường bảng class\_detail

#### 2.2.2.10. Bảng subjects (học phần)

|  |  |  |  |
| --- | --- | --- | --- |
| **Tên cột** | **Kiểu dữ liệu** | **Giá trị mặc định** | **Mô tả** |
| id | Bigint | Tự tăng | Khóa chính của bảng |
| code | Varchar(50) |  | Mã tài khoản người dùng |
| created\_date | Datetime | Null | Ngày tạo bản ghi |
| updated\_date | Datetime | Null | Ngày cập nhật bản ghi |
| credits | Int |  | Số tín chỉ |
| name | Varchar(50) |  | Tên học phần |
| department\_id | bigint |  | Id bộ môn |

Bảng 2.10. Mô tả chi tiết các trường bảng subjects

#### 2.2.2.11. Bảng unit (khoa / phòng ban)

|  |  |  |  |
| --- | --- | --- | --- |
| **Tên cột** | **Kiểu dữ liệu** | **Giá trị mặc định** | **Mô tả** |
| id | Bigint | Tự tăng | Khóa chính của bảng |
| code | Varchar(50) | null | Mã tài khoản người dùng |
| created\_date | Datetime | Null | Ngày tạo bản ghi |
| updated\_date | Datetime | Null | Ngày cập nhật bản ghi |
| date\_founding | Datetime | Null | Ngày thành lập |
| name | Varchar(50) |  | Tên khoa/phòng ban |
| description | Varchar(255) | Null | Mô tả |
| user\_manager\_id | Bigint | Null | Id người quản lý |

Bảng 2.11. Mô tả chi tiết các trường bảng unit

#### 2.2.2.12. Bảng department (bộ môn)

|  |  |  |  |
| --- | --- | --- | --- |
| **Tên cột** | **Kiểu dữ liệu** | **Giá trị mặc định** | **Mô tả** |
| id | Bigint | Tự tăng | Khóa chính của bảng |
| code | Varchar(50) | null | Mã tài khoản người dùng |
| created\_date | Datetime | Null | Ngày tạo bản ghi |
| updated\_date | Datetime | Null | Ngày cập nhật bản ghi |
| date\_founding | Datetime | Null | Ngày thành lập |
| name | Varchar(50) |  | Tên bộ môn |
| description | Varchar(255) | Null | Mô tả |
| user\_manager\_id | Bigint | Null | Id người quản lý |
| unit\_id | Bigint |  | Id khoa |

Bảng 2.12. Mô tả chi tiết các trường bảng department

#### 2.2.2.13. Bảng course (khóa học)

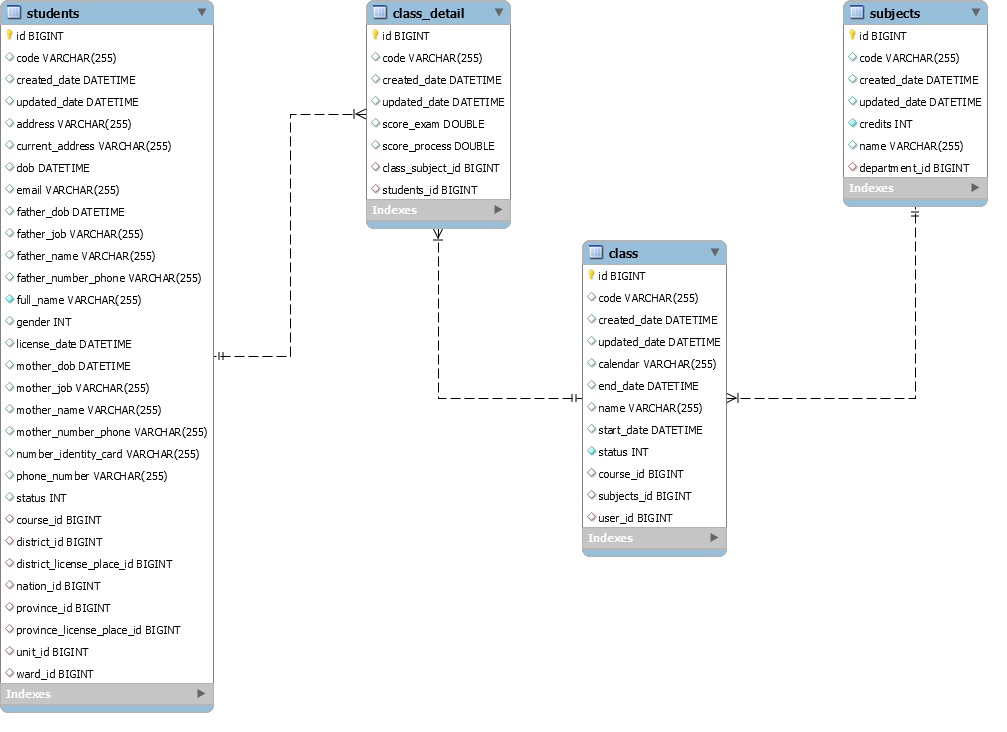
|  |  |  |  |
| --- | --- | --- | --- |
| **Tên cột** | **Kiểu dữ liệu** | **Giá trị mặc định** | **Mô tả** |
| id | Bigint | Tự tăng | Khóa chính của bảng |
| code | Varchar(50) | null | Mã tài khoản người dùng |
| created\_date | Datetime | Null | Ngày tạo bản ghi |
| updated\_date | Datetime | Null | Ngày cập nhật bản ghi |
| description | Varchar(255) | Null | Mô tả |
| name | Varchar(50) |  | Tên khóa học |
| end\_date | Datetime | Null | Ngày kết thúc |
| start\_date | Datetime | Null | Ngày bắt đầu |
| targets | Bigint | Null | Chỉ tiêu tuyển sinh |
| reality | Bigint | Null | Thực tế tuyển sinh |
| number | Int | Null | Số khoa |

Bảng 2.13. Mô tả chi tiết các trường bảng course

### 2.2.3. Mô tả mỗi quan hệ sinh viên:

Với mỗi thực thể class (lớp học phần) tương ứng với một thực thể subjects (học phần). Mỗi thực thể class có nhiều class\_detail (chi tiết lớp học phần), các thực thể class\_detail tương ứng với một thực thể students (sinh viên) cùng với các thông tin về điểm học cảu sinh viên thuộc lớp học phần.

Sơ đồ quan hệ được biểu diễn như hình bên dưới:



Hình 2.3. Mô hình ERD quan hệ sinh viên

# CHƯƠNG 3: XÂY DỰNG VÀ PHÁT TRIỂN

## 3.1. Cấu hình Security:

### 3.1.1. Server:

Trong file “*pom.xml*” khai báo các thư viện cho security như sau:

<**dependency**>  
 <**groupId**>org.springframework.boot</**groupId**>  
 <**artifactId**>spring-boot-starter-security</**artifactId**>  
</**dependency**>

<**dependency**>  
 <**groupId**>org.springframework.cloud</**groupId**>  
 <**artifactId**>spring-cloud-starter-security</**artifactId**>  
</**dependency**>

<**dependency**>  
 <**groupId**>org.springframework.security</**groupId**>  
 <**artifactId**>spring-security-test</**artifactId**>  
 <**scope**>test</**scope**>  
</**dependency**>  
<**dependency**>  
 <**groupId**>io.jsonwebtoken</**groupId**>  
 <**artifactId**>jjwt</**artifactId**>  
 <**version**>0.9.0</**version**>  
</**dependency**>

Trong package “*com.datn.app.config”* tạo các class *SecurityConfig, Oauth2AuthorizationSererConfig, Oauth2ResourceServerConfig* để xác thực quyền truy cậpcác đường link API lấy dữ liệu trong ứng dụng.

*Oauth2AuthorizationSererConfig* là đối tượng quyết định việc cấp quyền truy cập vào dữ liệu cho client và chứa các thông tin chung để tạo mã Jwt (Json Web Token) như thời gian hết hạn, khóa bí mật, v.v…. Các giá trị của thông tin được lưu trong file “*application.properties*” và được tham chiếu các trường trong class thông qua annotation @Value của Spring .

* Khai báo thông tin cấu hình trong “*application.properties”*:

oauth.clientId=Hiep-GTVT  
oauth.clientSecret=abc@123456oauth.accessTokenValidity=36000  
oauth.refreshTokenValidity=240000  
oauth.signingKey=doantn2020  
oauth.resourceId=app-api

* Khai báo thông tin trong class *Oauth2AuthorizationSererConfig*:

@Value("${oauth.signingKey}")  
private String signingKey;  
@Value("${oauth.accessTokenValidity}")  
private int accessTokenValidity;  
@Value("${oauth.refreshTokenValidity}")  
private int refreshTokenValidity;

* Phương thức tạo mã Jwt: Jwt được tạo bởi các phương thức và theo chuẩn của thư viện Oauth2 Security của spring boot, phương thức tokenServices() được đánh dấu là @Bean để thực hiện cài đặt một số thuộc tính như thời gian hết hạn cảu Jwt, v.v... theo như chúng ta muốn.

@Bean  
public DefaultTokenServices tokenServices() {  
 final DefaultTokenServices defaultTokenServices = new DefaultTokenServices();  
 defaultTokenServices.setTokenStore(tokenStore());  
 defaultTokenServices.setTokenEnhancer(accessTokenConverter());  
 defaultTokenServices.setAccessTokenValiditySeconds(accessTokenValidity);  
 defaultTokenServices.setRefreshTokenValiditySeconds(refreshTokenValidity);  
 defaultTokenServices.setSupportRefreshToken(true);  
 return defaultTokenServices;  
}

* Phương thức kiểm tra mã Jwt: Jwt sẽ được tự động kiểm tra bởi thư viện security của spring boot khi có request gửi về.

Lớp *Oauth2ResourceServerConfig* là server nơi chứa thông tin dữ liệu cần chia sẻ.

Cơ chế hoạt động của Oauth2 gồm 3 bước chính:

* Client request quyền truy cập và resource owner chấp thuận cấp quyền.
* Client gửi authorization grant do resource owner cung cấp lên cho authorization server, sau khi xác nhận, server sẽ gửi về access token cho client.
* Client sử dụng access token này trong quá trình liên lạc với resource server. Resource server sẽ kiểm tra tính hợp lệ của token, nếu đúng trả protected resource về cho client.

### 3.1.2. Client:

Trong project angular thêm thư viện @angular-jwt thông qua lệnh terminal “*npm install @auth0/angular-jwt*”. Thư viện này cung cấp một *HttpInterceptor* tự động cài mã Jwt vào header mỗi request của HttpClient. Lớp *AuthInterceptor* thực thi interface *HttpInterceptor* với phương thức intercept gán mã Jwt và header mỗi request gửi về server đồng thời khai báo một providers HTTP\_INTERCEPTOR trong “*app.module.ts*” để ứng dụng thực thi phương thức intercept.

* Phương thức intercept():

intercept(request: HttpRequest<*unknown*>, next: HttpHandler): Observable<*HttpEvent*<*unknown*>> {  
 *const* accessToken = *this*.storageService.getAccessToken();  
 *if* (accessToken == *null* || *this*.jwtHelperService.isTokenExpired(accessToken)){  
 request = request.clone({  
 headers: request.headers  
 .set(*environment*.config.auth.header, *environment*.config.auth.auth\_basic + ' ' + *btoa*(`${*environment*.config.auth.client\_id}:${*environment*.config.auth.client\_secret}`))  
 .set('Content-type', 'application/x-www-form-urlencoded; charset=utf-8')  
 });  
 }*else* {  
 request = request.clone({  
 headers: request.headers  
 .set(*environment*.config.auth.header, *environment*.config.auth.auth\_bearer + ' ' + accessToken)  
 });  
}

* Khai báo providers:

{  
 provide: ***HTTP\_INTERCEPTORS***,  
 useClass: AuthInterceptor,  
 multi: true  
}

Trong thư viên angular-jwt còn cung cấp một lớp *JwtHelperService* với các phương thức sẵn có phục vụ cho việc giải mã, kiểm tra các thông tin như thời gian tạo, thời gian hết hạn, v.v… của mã Jwt.

## 3.2. Cấu hình kết nối cơ sở dữ liệu:

Thông tin cấu hình kết nối tới cơ sở dữ liệu được viết trong file “*application.yml*”, lưu trữ ở vị trí “*src/main/resources*” ở phần mã java. Các thông tin kết nối cơ sở dữ liệu bao gồm:

* Url: đường link localhost chỉ đến database sẽ kết nối thuộc hệ quản trị cở sở dữ liệu MySql.
* Username: tài khoản đăng nhập vào hệ quản trị cơ sở dữ liệu MySql với các quyền cần thiết như tạo/sửa bảng, thêm/sửa/xóa bản ghi, v.v….
* Password: mật khẩu tương ứng với username.
* Driver: thư viện của Java để kết nối chương trình viết bằng mã Java với hệ quản trị cơ sở dữ liệu MySql.

Các thông tin được khai báo trong file “*application.yml”* như sau:

* Spring.datasource.url=jdbc:mysql://localhost:3306/do\_an\_db?serverTimeZone=UTC&userLegacyDateTimeCode=false&useUnicode=yes&characterEncoding=UTF-8&reWriteBatchedInserts=true
* Spring.datasource.username=root
* Spring.datasource.password=15011996
* Spring.datasource.driver-class-name=com.mysql.cj.jdbc.Driver

Ngoài các thông tin kết nối tới cơ sở dữ liệu còn có các thông tin cấu hình cho Hibernate và JPA với mục đích như cho ứng dụng biết loại cơ sở dữ liệu đang sử dụng để hibernate chuyển đổi câu truy vấn SQL, thu thập số liệu thống kê để hibernate điều chỉnh hiệu suất, hiển thị và định dạng câu truy vấn SQL ra console hoặc file log, v.v…. Thông tin các cấu hình được mô tả chi tiết như hình dưới đây (Hình 3.1).

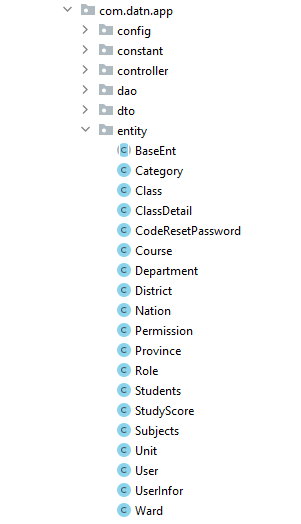


Hình 3.1. Các thông tin cấu hình database và Hibernate

trong file application.yml

## 3.3. Entitiy:

Entity là các đối tượng trong ứng dụng, mỗi đối tượng tương ứng với một bảng trong cơ sở dữ liệu. Các đối tượng đều được đánh dấu bằng hai annotation @Entity và @Table thuộc “*javax.persistence*” nhằm thông báo cho hibernate biết rằng đây là một thực thể và là một bảng trong cơ sở dữ liệu. Các trường của đối tượng mặc định tương ứng với trường dữ liệu trong cơ sở dữ liệu, nếu trường thông tin có các cài đặt khác như giá trị default, not null, v.v… sẽ khai báo thông qua anotation @Column. Các trường của đối tượng không phải là một trường thông tin trong cơ sở dữ liệu thì khai báo kèm với annotation @Transient. Khóa chính của thực thể trong cơ sở dữ liệu được đánh với bởi annotation @Id trong trường thông tin của đối tượng. Các đối tượng Entity được khai báo trong package “*com.datn.app,entity*”.



Hình 3.2. Package chứa các Entity

Ví dụ đối tượng *User* được khai báo như sau:

@Entity  
@Table  
public class User extends BaseEnt implements UserDetails {  
 @Column(unique = true, nullable = false)  
 private String username;  
 @Column(nullable = false)  
 private String password;  
 @Column(columnDefinition = "boolean default true")  
 private boolean accountNonExpired = true;  
 @Column(columnDefinition = "boolean default true")  
 private boolean accountNonLocked = true;  
 @Column(columnDefinition = "boolean default true")  
 private boolean enabled = true;  
  
 public User() { }  
  
 @Override  
 public UserDto convertToDto() {  
 return AppUtil.*mapToDtoAndEnt*(this, UserDto.class);  
 }

}

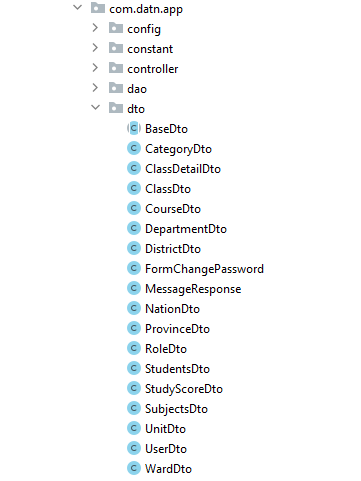
Các đối tượng đều kế thừa một lớp trừu tượng *BaseEnt* chứa các thông tin chung và các phương thức chung của tất cả đối tượng. Lớp *BaseEnt* được mô tả như sau:

@MappedSuperclass  
public abstract class BaseEnt {  
 @Id  
 @GeneratedValue(strategy = GenerationType.*IDENTITY*)  
 private Long id;  
 private String code;  
 private Date createdDate;  
 private Date updatedDate;  
  
 public BaseEnt() { }  
  
 public BaseEnt(Long id, String code, Date createdDate, Date updatedDate) {  
 this.id = id;  
 this.code = code;  
 this.createdDate = createdDate;  
 this.updatedDate = updatedDate;  
 }  
  
 public abstract BaseDto convertToDto();  
  
}

Anotation @MappedSuperClass dùng để chỉ định đây không phải là một thực thể trong cơ sở dữ liệu. Phương thức asDto() chuyển đổi dữ liệu sang Data Transfer Object để trả kết quả về phía client và các phương thức getter , setter.

## 3.4. Data Transfer Object (DTO):

Data Transfer Object (DTO) – đối tượng truyền dữ liệu là một đối tượng mang nhiều dữ liệu trong các quy trình. DTO được sử dụng khi chúng ta muốn truyền dữ liệu với nhiều thông tin trong một lần gửi yêu cầu từ client đến server. Các đối tượng này là các lớp đơn giản chỉ có các phương thức getter, setter, contructor và không có bất kỳ phương thức logic nghiệp vụ nào. Các lớp nghiệp vụ của server lấy dữ liệu từ cơ sở dữ liệu, xử lý dữ liệu và chuyển đổi thành đối tượng DTO và gửi nó đên client. Đối với client các đối tượng DTO chỉ để đọc. Các đối tương DTO được khai báo trong package “*com.datn.app.dto*”.



Hình 3.3. Package chứa các đối tượng DTO

Ví dụ khai báo đối tượng *UserDto*:

@JsonInclude(JsonInclude.Include.*NON\_NULL*)  
@JsonIgnoreProperties(ignoreUnknown = true)  
public class UserDto extends BaseDto{  
 private String username;  
 private String password;  
 private Long userInforId;  
 private String firstName;  
 private String lastName;  
 private int gender; *//0: nam, 1: nữ, 2: khác* private String genderString;  
 private Date dob;  
 private String address;  
 private String email;  
 private String numberPhone;  
 private Long roleId;  
 private String roleName;  
 private Long unitId;  
 private String unitName;  
 private String currentPosition;  
 private Date workStartDate;  
 private Date workEndDate;  
 private String qualifications;  
 private Long nationId;  
 private String nationName;  
 private Long provinceId;  
 private String provinceName;  
 private Long districtId;  
 private String districtName;  
 private Long wardId;  
 private String wardName;  
 private String roleCode;  
 private Long departmentId;  
 private String departmentName;  
  
 private boolean accountNonLocked;  
  
 public UserDto() { }

}

Giống như các entity, các đối tượng DTO đều kế thừa lớp trừu tượng *BaseDto* có thông tin và phương thức chung của các đối tượng. Phương thức convertToEnt () để chuyển đổi một đối tượng DTO về đối tượng Entity. Các đối tượng DTO không nhất thiết phải tương ứng với đối tượng Entity vì DTO chỉ chứa các thông tin mà phía client cần đọc, một ứng dụng có rất nhiều đối tượng DTO. Mô tả lớp *BaseDto* như sau:

public abstract class BaseDto {  
 private Long id;  
 private String code;  
  
 public BaseDto() { }  
  
 public abstract BaseEnt convertToEnt();

}

## 3.5. Data Access Object (DAO):

Data Access Object (DAO) – đối tượng truy cập dữ liệu là một trong những Pattern thuộc nhóm cấu trúc. DAO được sử dụng để phân tách logic lưu trữ dữ liệu trong một lớp riêng biệt. Theo cách này, các service được che dấu về cách các hoạt động cấp thấp để truy cập cơ sở dữ liệu được thực hiện thay vì có logic trực tiếp với cơ sở dữ liệu hay bất kỳ cơ chế lưu trữ nào ứng dụng cần sử dụng má sẽ để các logic này giao tiếp với lớp trung gian DAO. Lớp DAO sẽ giao tiếp với cơ sở dữ liệu như thực hiện các công việc liên quan đến lưu trữ và truy vấn dữ liệu. DAO dựa trên nguyên tắc thiết kế abstraction và encapsulation, nó sẽ bảo vệ phần còn lại của ứng dụng khỏi mọi thay đổi trong lớp lưu trữ như thay đổi công nghệ lưu trữ, thay đổi hệ quản trị cơ sở dữ liệu, v.v….

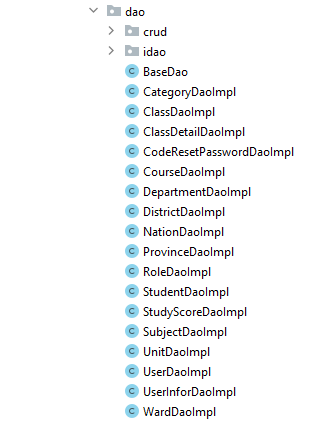
Các thành phần của DAO bao gồm:

* **Data Access Object (DAO)**: là các interface định nghĩa các phương thức trừu tượng việc triển khai truy cập dữ liệu.
* **Data Access Object Concrete**: là các lớp thực thi các interface DAO, lấy dữ liệu từ nguồn dữ liệu.
* **Data Transfer Object (DTO)**: là các lớp đơn giản để lưu trữ dữ liệu được truy xuất bằng các lớp DAO.

Các interface DAO được khai báo trong package “*com.datn.app.dao.idao*” và “*com.datn.app.dao.crud*”. Các interface trong *idao* đều kế thừa interface *JpaRepository* khai báo các phương thức trừu tượng không cần triển khai theo nguyên tắc khai báo của JPA hoặc thông qua annotation @Query nhằm giảm thiểu lượng mã phải viết. Ví dụ UserDao*:*

public interface UserDao extends JpaRepository<User, Long>, CrudUserDao {  
 User findByUsername(String username);  
}

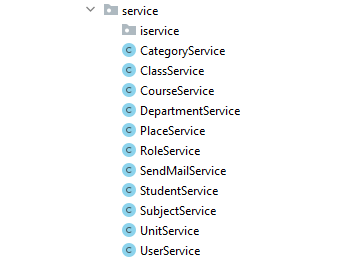
Các interface trong *crud* khai báo các phương thức trừu tượng thực thi việc truy cập dữ liệu với các truy vấn phức tạp và không theo nguyên tắc khai báo của JPA, nó sẽ được triển khai trong các lớp thực thi các interface này trong package “*com.datn.app.dao*”.



Hình 3.4. Package chứa các lớp DAO

## 3.6. Service:

Service là các interface và các lớp thực thi các interface này gồm các phương thức logic nghiệp vụ. Các phương thức xử lý dữ liệu được truy vấn từ cơ sở dữ liệu, chuyển đổi dạng dữ liệu trả về cho client những dữ liệu được yêu cầu hoặc xử lý các dữ liệu nhận từ client trước khi lưu trữ vào cơ sở dữ liệu.

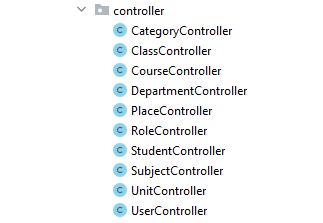


Hình 3.5. Package chứa các lớp Service

Các service được khai báo trong package “*com.datn.app.service*”. Các Service được đánh dấu bởi annotation @Service để thông báo cho Spring đây là một Service Component, annotation @Transactional để chỉ định các phương thức của service sẽ hoạt động trong một giao dịch đảm bảo tính toàn vẹn dữ liệu nếu có lỗi xảy ra trong trình giao tiếp với cơ sở dữ liệu.

## 3.7. Rest Controller:

Rest Controller bao gồm các lớp tiếp nhận những yêu cầu xử lý được gửi từ phía client. Nó gọi tới các service xử lý logic nghiệp vụ để lấy thông tin dữ liệu cần thiết và trả về cho client dữ liệu dưới dạng Json.



Hình 3.6. Package chứa các Restcontroller

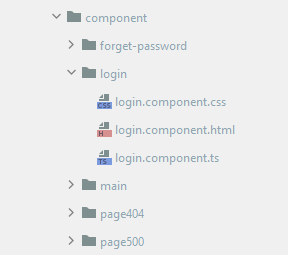
\*Mô tả một restcontroller (UserController):

@RestController  
@RequestMapping("/user")  
public class UserController{  
 @Autowired  
 private UserService userService;  
  
 @PostMapping("")  
 public ResponseEntity<UserDto> saveOrUpdate(@RequestBody UserDto dto, HttpServletRequest request){  
 return new ResponseEntity<>(userService.saveOrUpdate(dto), HttpStatus.*OK*);  
 }  
  
 @DeleteMapping("/{id}")  
 public ResponseEntity<MessageResponse> deleteById(@PathVariable Long id, HttpServletRequest request){  
 return new ResponseEntity(userService.deleteById(id), HttpStatus.*OK*);  
 }  
  
 @GetMapping("/search")  
 public ResponseEntity<Page<UserDto>> search(@RequestParam(required = false) String page,  
 @RequestParam(required = false) String limit,  
 @RequestParam(required = false) String code,  
 @RequestParam(required = false) String name,  
 @RequestParam(required = false) String username,  
 @RequestParam(value = "unit", required = false) String unitId){  
 Pageable pageable = AppUtil.*getPageable*(page, limit);  
 return new ResponseEntity(userService.search(pageable, code, name, username, unitId), HttpStatus.*OK*);  
 }  
  
 @GetMapping("/search-teacher")  
 public ResponseEntity<Page<UserDto>> searchTeacher(@RequestParam(required = false) String page,  
 @RequestParam(required = false) String limit,  
 @RequestParam(required = false) String code,  
 @RequestParam(required = false) String name,  
 @RequestParam(value = "department", required = false) String departmentId,  
 @RequestParam(value = "unit", required = false) String unitId){  
 Pageable pageable = AppUtil.*getPageable*(page, limit);  
 return new ResponseEntity(userService.searchTeacher(pageable, code, name, departmentId, unitId), HttpStatus.*OK*);  
 }

}

## 3.8. Components:

Component là các thành phần tạo nên một ứng dụng hoàn chỉnh. Một component có ba thành phần chính là code TypeScript, template HTML và CSS. Một ứng dụng angular có rất nhiều component, mỗi component được đặt trong một thư mục như hình (Hình 3.7):



Hình 3.7. Cấu trúc ứng dụng Angular - component

* **[component name].component.css:**được sử dụng để chứa mã style cho giao diện web.
* **[component name].component.html:**chứa các mã giao diện html.
* **[component name].component.ts:**chứa các mã xử lý logic.

Các component được tạo ra đều được khai báo ở mục “*declarations*” trong file “*app.module.ts*” để xác định các component sẽ được sử dụng. Mục “*imports*” khai báo các module ứng dụng sử dụng trong đó có một số module quan trọng như *BrowserModule* – module cần thiết để mọi ứng dụng angular chạy trên trình duyệt, *HttpCientModule* – module cung cấp các phương thức để giao tiếp với server, v.v….

## 3.9. Routing:

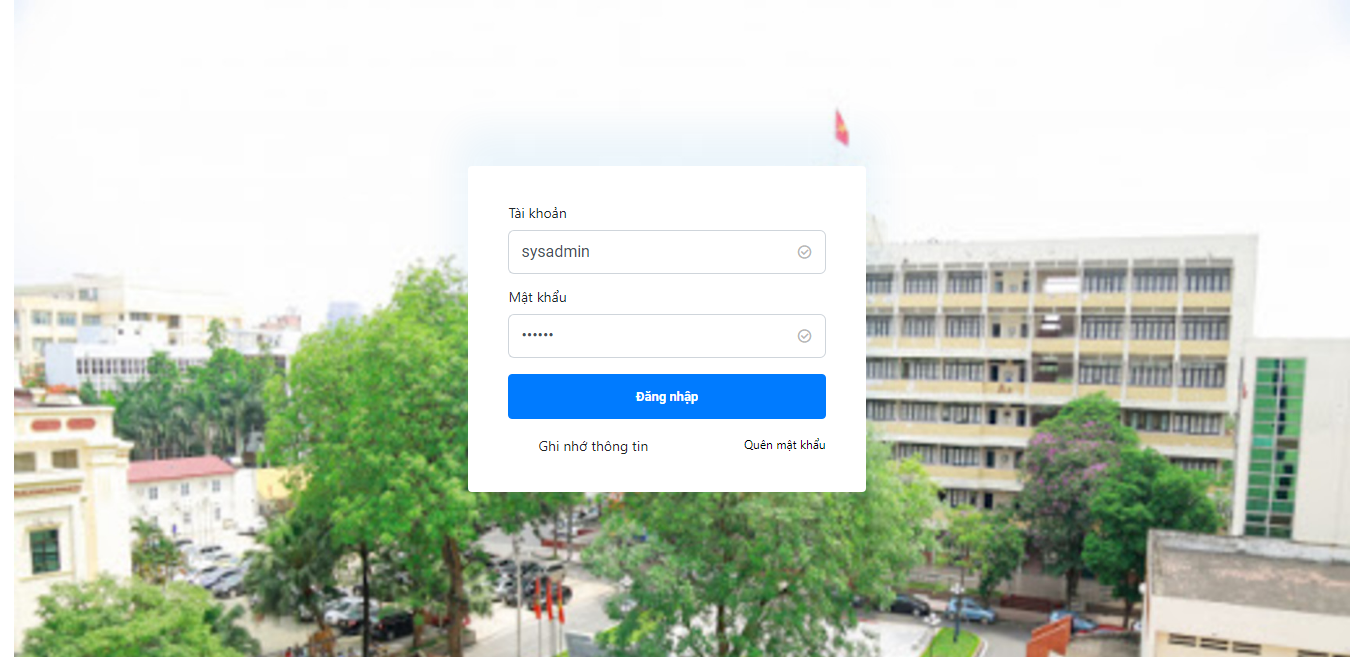
Routing – bộ định tuyến dùng để xử lý điều hướng từ chế độ xem này sang chế độ xem khác khi người dùng thực hiện các tác vụ. Các route được khai báo trong file “*app-routing.module.ts*” và được import vào “app.module.ts”. Mỗi route có hai thành phần chính là path – url chỉ dẫn đến view cần xem, component – thành phần view tương ứng với url được khai báo ở path.

*const* routes: *Routes* = [  
 {path: '', redirectTo: 'login', pathMatch: 'full'},  
 {path: 'login', component: LoginComponent},  
 {path: 'forget-password', component: ForgetPasswordComponent},  
 {path: 'home', component: MainComponent, loadChildren: () => *import*('./component/main/main.module').then(m => m.MainModule)},  
 {path: 'error', children: [  
 {path: '404', component: Page404Component},  
 {path: '500', component: Page500Component}  
 ]}  
];

Ngoài các routing và component, ứng dụng Angular cũng có các service chứa các phương thức dùng để gửi các yêu cầu lên server. Dữ liệu nhận về ở dạng Json sẽ được tùy biến thành các đối tượng model để hiển thị lên cho người dùng. Các service phải được đánh dấu với annotation @Injectable({providedIn: ‘root’}).

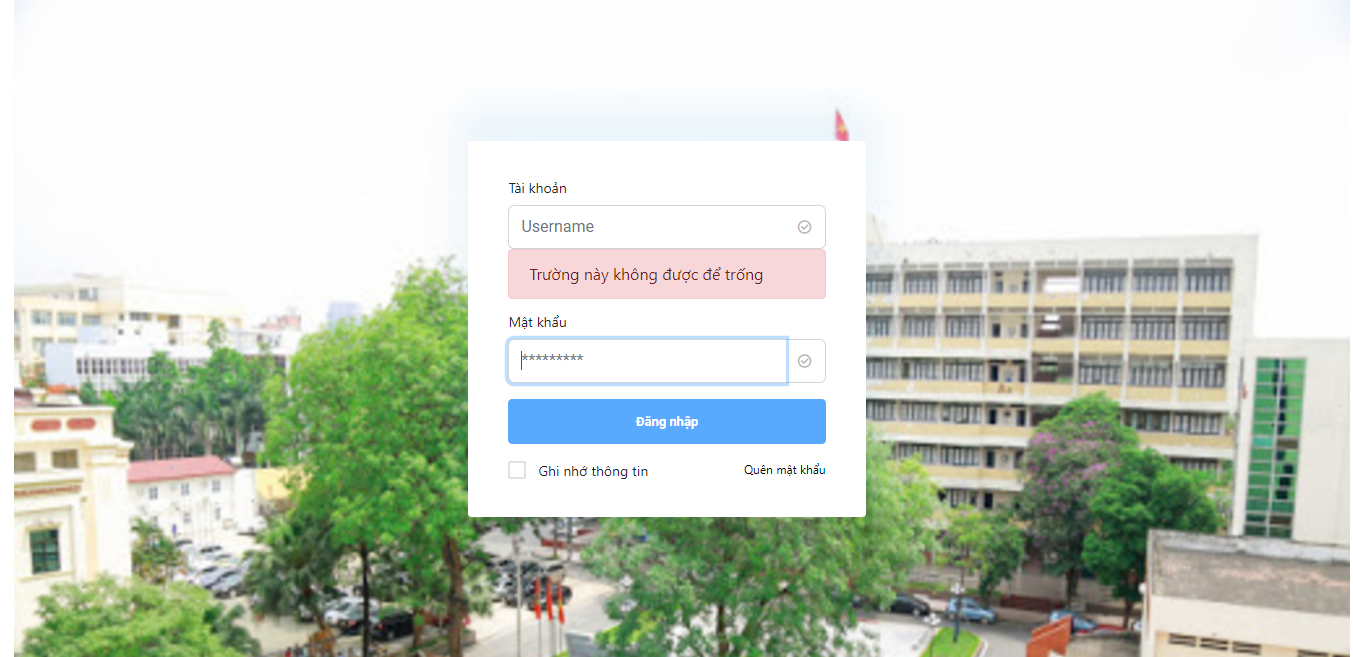
# CHƯƠNG 4: XÂY DỰNG GIAO DIỆN

## 4.1. Giao diện đăng nhập:



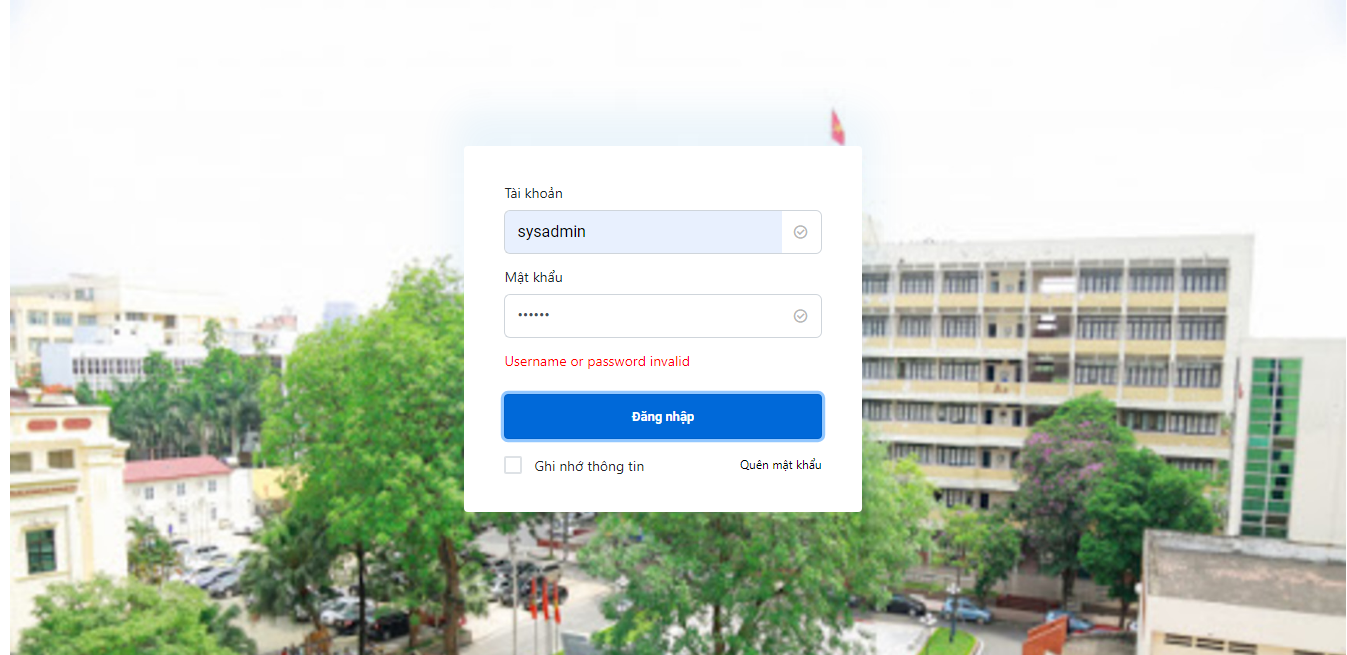
Hình 4.1. Trang đăng nhập

Khi người dùng truy cập hệ thống mặc định được điều hướng đến trang đăng nhập. Người dùng nhập tài khoản và mật khẩu để tiến hành truy cập hệ thống. Nút đăng nhập sẽ bị disable nếu người dùng không nhập đủ thông tin và có thống báo ở dưới trường thông tin bị thiếu như hình dưới đây.



Hình 4.2. Trang đăng nhập – báo lỗi khi thiếu thông tin

Khi thông tin đăng nhập của tài khoản sai hoặc tài khoản bị khóa sẽ có thông báo như sau:



Hình 4.3. Trang đăng nhập – báo lỗi khi sai thông tin

Khi đăng nhập thành công sẽ có một chuồi token (Json web token - JWT) được trả về và được lưu trữ trong localstorage của trình duyệt. Chuỗi token này được sử dụng xác thực tính hợp lệ của các request gửi lên service để thực hiện lấy hoặc lưu dữ liệu theo yêu cầu của người dùng. Token này được đính kèm ở mục ‘Authorization’ trong header cảu các request.

## 4.2. Giao diện thanh công cụ và menu:

Thanh menu bên trái là các danh mục tương ứng với chức danh của tài khoản đăng nhập. Thanh công cụ bên trên trang web bao gồm:

* Logo Trường Giao thông vận tải
* Nút thanh đổi ngôn ngữ: tiếng việt và tiếng anh
* Ô tìm kiếm chung
* Biểu tượng người dùng



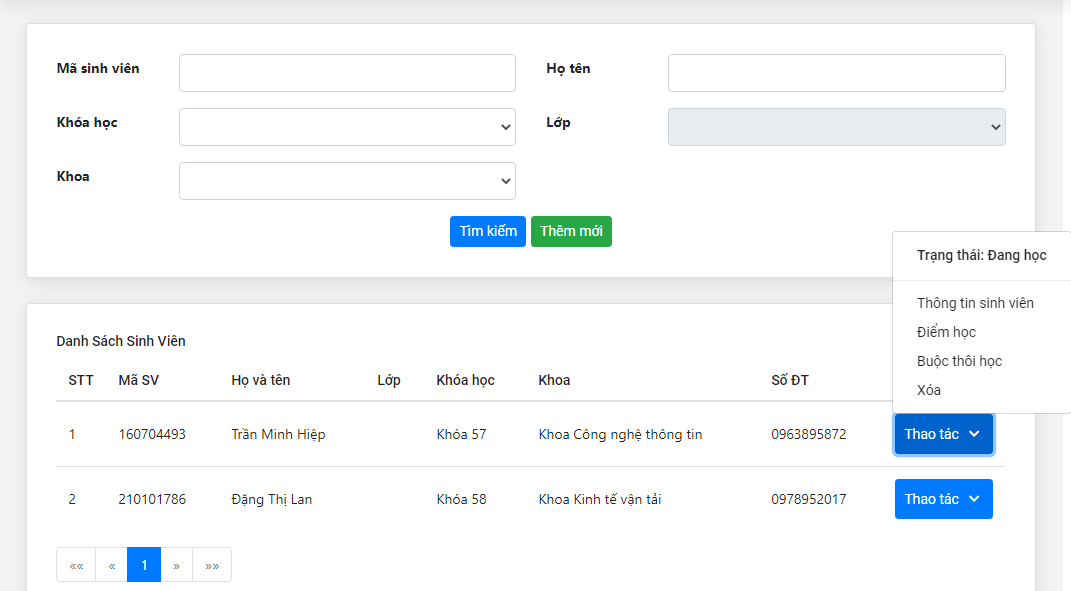
Hình 4.4. Thanh công cụ

Khi người dùng ấn vào logo hệ thống điều hướng về trang chủ hệ thống. Khi người dùng thực hiện thao tác thay đổi ngôn ngữ, hệ thống ngôn ngữ sẽ thay đổi sang ngôn ngữ được chọn, ngôn ngữ mặc định khi truy cập hệ thống là tiếng việt. Khi người dùng ấn vào biểu tượng người dùng có một menu hiện ra. Menu gồm tên và email tài khoản đăng nhập và một số thao tác khác.

## 4.3. Giao diện dánh sách sinh viên:

Khi người dùng chọn danh mục ‘danh sách sinh viên’, hệ thống sẽ điều hướng đến trang danh sách sinh viên.

Trang web gồm hai phần: form tìm kiếm và bảng danh sách sinh viên. Ở form tìm kiếm người dùng nhập các thông tin và ấn nút ‘tìm kiếm’ để có thể tìm kiếm sinh viên một cách nhanh nhất. Trong phần danh sách sinh viên hiển thị những thông tin cơ bản cảu sinh viên như mã sinh viên, tên, lớp, khoa, v.v.... Khi ấn nút thao tác sẽ hiện ra một menu hiển thị trạng thái cảu sinh viên và các thao tác có thể thực hiện.

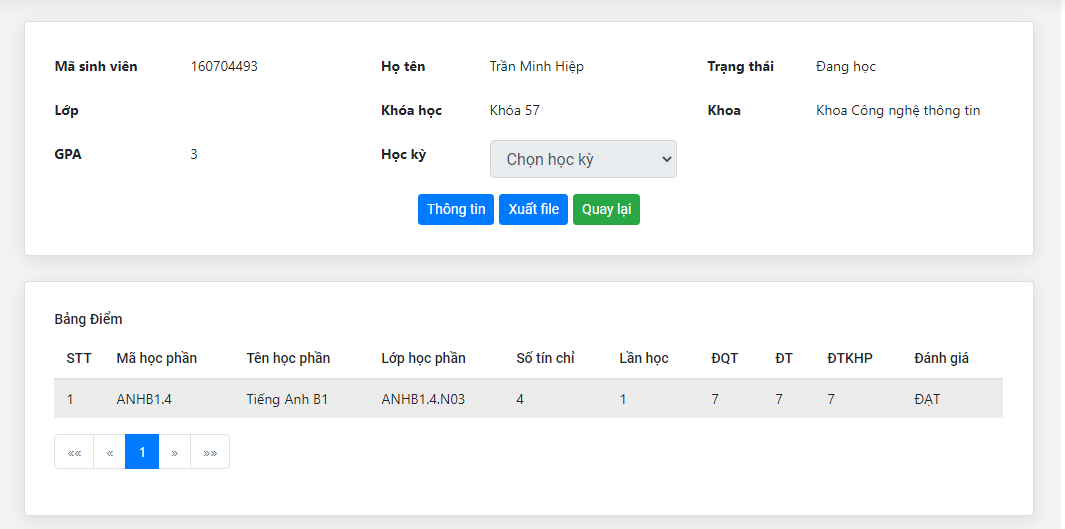


Hình 4.5. Trang quản lý sinh viên

Khi ấn nút’ xóa’ sẽ có một popup thông báo, người dùng ấn nút ‘OK’ trên popup hệ thống sẽ gửi một request yêu cầu xóa lến service để xóa sinh viên được chon. Khi người dùng ấn nút ‘buộc thôi học’ sẽ có một popup thông báo, người dùng ấn nút ‘OK’ trên popup hệ thống sẽ gửi request về service thực hiện yêu cầu thay đổi trạng thái của sinh viên được chọn sang ‘nghỉ học’. Khi ấn nút ‘điểm học’ hệ thống điều hướng đến trang điểm học cảu sinh viên. Khi ân nút ‘thông tin sinh viên’ hoặc nút ‘thêm mới’ trên form tìm kiếm hệ thống điều hướng đến trang thông tin sinh viên để thực hiện thêm mới hoặc sửa thông tin của sinh viên.

## 4.4. Giao diện điểm học:

Khi người dùng thực hiện thao tác xem điểm học sinh viên ở trang danh sách sinh viên. Trang này gồm hai phần: form thông tin và bảng điểm. Form thông tin gồm một số thông tin cơ bản và điểm trung bình tích lũy (GPA) cảu sinh viên. Phần bảng điểm gồm danh sách các học phần đã học của sinh viên cùng với thông tin về điểm và đánh giá đối với học phần đó.



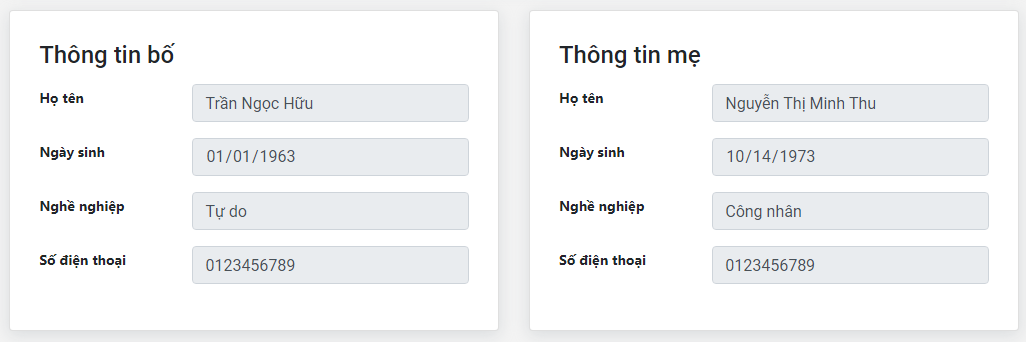
Hình 4.6. Trang điểm học

Nút ‘thông tin’ để điều hướng đến trang thông tin. Nút ‘quay lại’ dùng để quay lại trang danh sách sinh viên. Nút ‘xuất file’ dùng để xuất bảng điểm ra một file excel.

## 4.5. Giao diện form sinh viên:

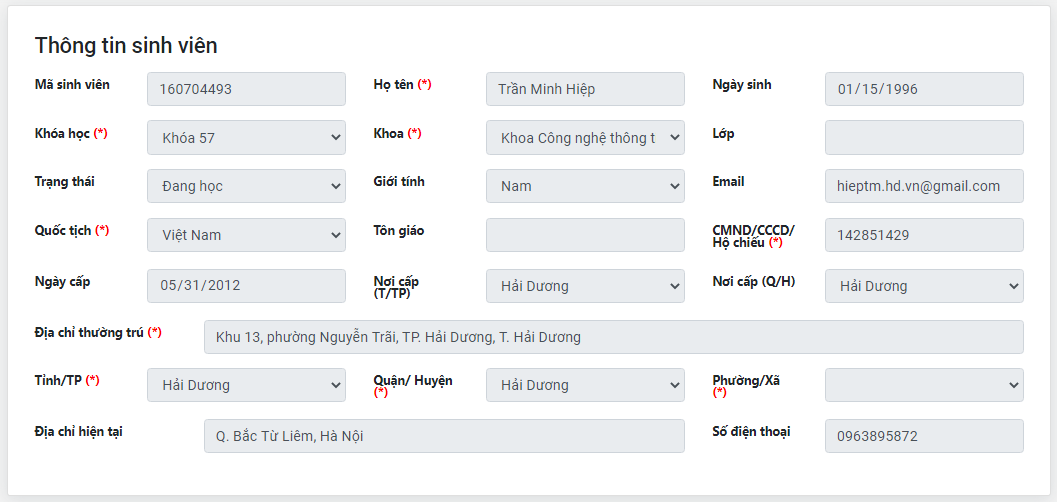
Trang form sinh viên được sử đụng đẻ thêm mới hoặc xem và sửa thông tin sinh viên. Trang gồm ba phần: thông tin cá nhân, thông tin bố, thông tin mẹ.

Phần thông tin bố/mẹ bao gồm các thông tin: tên, ngày sinh, nghề nghiệp, số điện thoại. Phần thông tin bố/mẹ được hiển thị như hình:



Hình 4.7. Trang thông tin sinh viên – phần thông tin bố mẹ

Phần thông tin cá nhân sinh viên bao gồm các thông tin như mã sinh viên, tên, ngyaf sinh, quốc tịch, số CMND/CCCD/hộ chiếu, quê quán, địa chỉ, v.v.... Thông tin cá nhân sinh viên được hiển thị như hình dưới đây:



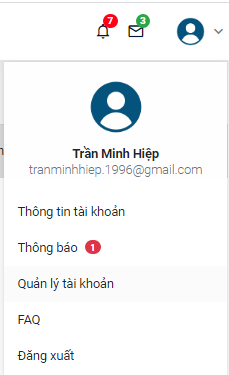
Hình 4.8. Trang thông tin sinh viên – phần thông tin cá nhân

Phần thao tác của trang được để ở cuối. Các nút thao thao tác khi thực hiện thêm mới và cập nhật hiển thị khác nhau. Khi thêm mới có các thao tác: lưu, hủy và quay lại. Khi cập nhập có các thao tác: cập nhật, điểm học, quay lại , lưu, hủy. Ấn nút điểm học hệ thống sẽ điều hướng đến trang điểm học. Khi ấn nút cập nhật, nó sẽ ẩn đi và thay thế bằng hai nút thao tác lưu và hủy. Thao tác lưu để thực hiện lưu thông tin đã nhập hoặc thay đổi. Thao tác hủy thực hiện tải lại trang về thông tin ban đầu. Thao tác quy lại điều hướng về trang danh sách sinh viên.

Khi thêm mới hay cập nhật các trường có dấu sao đỏ là những trường thông tin bắt buộc phải có dữ liệu, nếu không có sẽ có thông báo lỗi hiện ra.

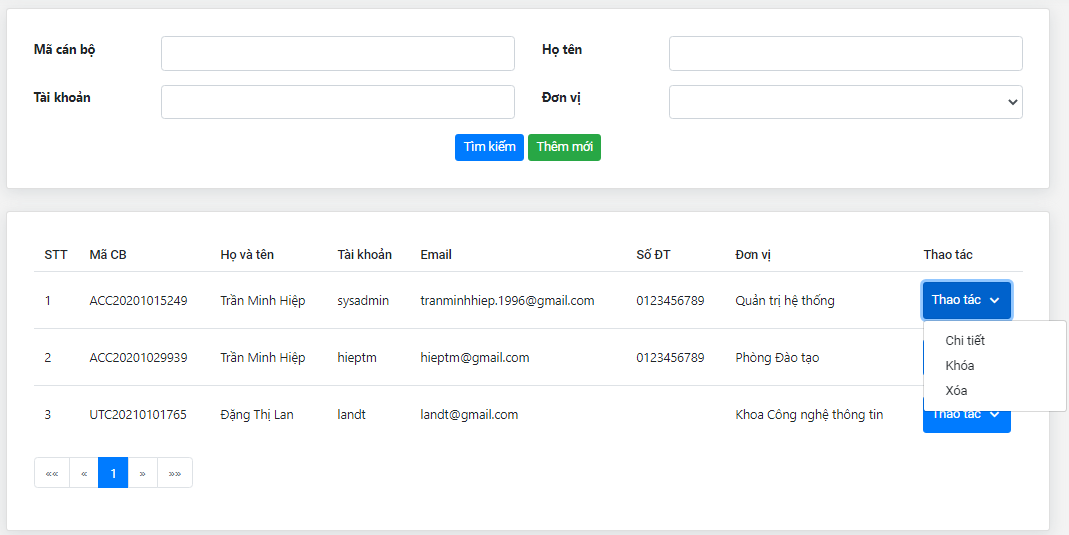
Các thao tác được thực hiện thành công hay có lỗi đều sẽ có thông báo hiện ra.

## 4.6. Giao diện quản lý tài khoản:

Giao diện quản lý tài khoản là trang quản lý các tài khoản truy cập hệ thống. Mỗi tài khoản là một cán bộ hay giảng viên của trường. Người dùng ấn vào biểu tượng người dùng trên thanh công cụ rồi chọn quản lý tài khoản. Để truy cập được trang này tài khoản đăng nhập phải có chức danh quản trị hệ thống, nếu không sẽ có thông báo lỗi quyền truy cập hiển thị ra.

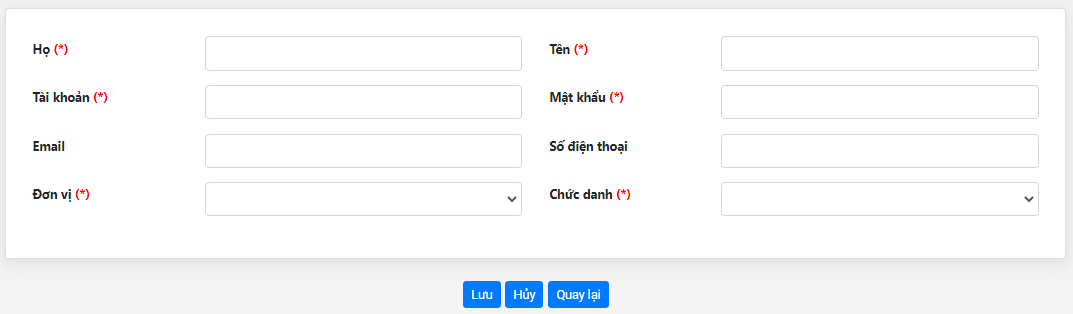
Hình 4.9. Menu user đăng nhập

Trang quản lý tài khoản gồm 2 phần: form tìm kiếm và phần danh sách tài khoản. Form tìm kiếm cho phép người dùng tìm kiếm tài khoản theo các trường thông tin trên form. Phần danh sách hiển thị danh sách các tài khoản và các thao tác đi kèm. Thao tác chi tiết sử dụng để xem chi tiết tài khoản. Thao tác xóa thực hiện xóa tài khoản ra khỏi hệ thống. Thao tác khóa thực hiện việc khóa tài khoản, khi tài khoản bị khóa sẽ không thể đăng nhập vào hệ thống. Thao tác thêm mới trên form tìm kiếm để điều hướng đến trang thêm mới thực hiện thêm mới một tài khoản.



Hình 4.10. Trang quản lý tài khoản

## 4.7. Giao diện form tài khoản:



Hình 4.11. Form tạo tài khoản

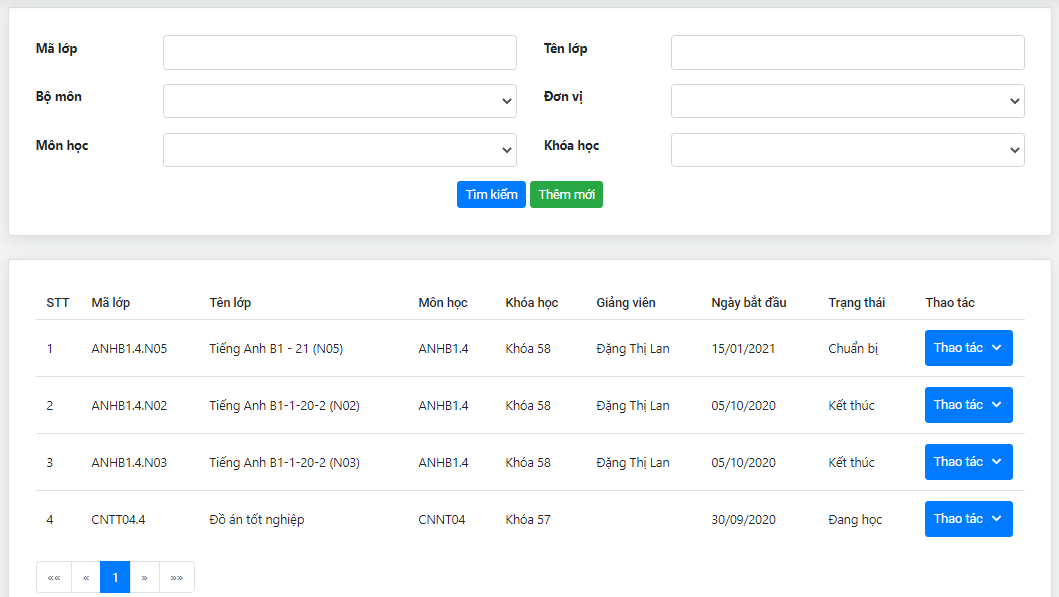
Trang form tài khoản gồm các trường thông tin cần để tạo ra một tài khoản truy cập hệ thống. Các trường thông tin có dấu sao đỏ là các trường bắt buộc. Nếu không có sẽ có thông báo lỗi hiện ra.

Các nút thao thao tác khi thực hiện thêm mới và cập nhật hiển thị khác nhau. Khi thêm mới có các thao tác: lưu, hủy và quay lại. Khi cập nhập có các thao tác: cập nhật, quay lại , lưu, hủy. Khi ấn nút cập nhật, nó sẽ ẩn đi và thay thế bằng hai nút thao tác lưu và hủy. Thao tác lưu để thực hiện lưu thông tin đã nhập hoặc thay đổi. Thao tác hủy thực hiện tải lại trang về thông tin ban đầu. Thao tác quy lại điều hướng về trang danh sách tài khoản.

Các thao tác được thực hiện thành công hay có lỗi đều sẽ có thông báo hiện ra.

## 4.8. Giao diện quản lý lớp học phần:

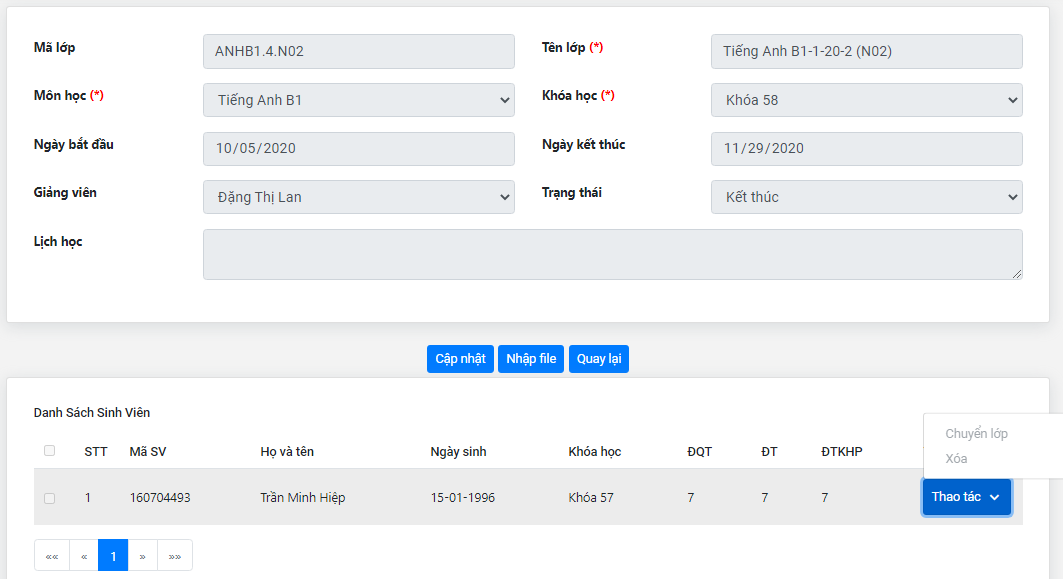
Khi người dùng chọn danh mục ‘quản lý lớp học phần’, hệ thống sẽ điều hướng đến trang quản lý lớp học phần.



Hình 4.12. Trang quản lý lớp học phần

Trang gồm hai phần: form tìm kiếm và phần danh sách lớp học. Form tìm kiếm cho phép người dùng tra cứu lớp học phần theo các thông tin trên form. Phần danh sách hiển thị các lớp học phần và các thao tác tướng ứng. Các lớp có trạng thái khác nhau có thao tác khác nhau. Thao tác chung là xem chi tiết. Lớp học phần ở trạng thái chuẩn bị có thêm các thao tác: xóa, nhập file. Lớp học phần ở trạng thái đang học có thêm thao tác nhập file. Thao tác xóa sử dụng để xóa lớp học được chọn ra khỏi hệ thống. Thao tác nhập file để thực hiện thêm danh sách sinh viên tham gia lớp học vào thông tin lớp học. Nút thêm mới sử dụng mở form thêm mới một lớp học.

## 4.9. Giao diện form lớp học phần:



Hình 4.13. Trang tạo/thông tin lớp học phần

Trang form lớp học phần gồm hai phần: phần thông tin lớp và phần danh sách sinh viên tham gia lớp học. Khi thêm mới phần danh sách sinh viên sẽ bị ẩn đi. Các trường thông tin có đánh dấu sao đỏ là các trường thông tin bắt buộc.

Các nút thao thao tác khi thực hiện thêm mới và cập nhật hiển thị khác nhau. Khi thêm mới có các thao tác: lưu, hủy và quay lại. Khi cập nhập có các thao tác: cập nhật, nhập file, quay lại , lưu, hủy. Khi ấn nút cập nhật, nó sẽ ẩn đi và thay thế bằng hai nút thao tác lưu và hủy. Thao tác lưu để thực hiện lưu thông tin đã nhập hoặc thay đổi. Thao tác hủy thực hiện tải lại trang về thông tin ban đầu. Thao tác quy lại điều hướng về trang danh sách lớp học. Thao tác nhập file thực hiện thêm một danh sách sinh viên tham gia lớp học vào thông tin lớp học.

Phần danh sách sinh viên có các thao tác chuyển lớp và xóa. Các checkbox được sử dụng thực hiện thao tác chuyển lớp và xóa nhiều sinh viên cùng một lần. Các thao tác này không thể thực hiện khi lớp học ở trạng thái kết thúc.

Các thao tác được thực hiện thành công hay có lỗi đều sẽ có thông báo hiện ra.

# CHƯƠNG 5: TỔNG KẾT VÀ ĐÁNH GIÁ

## 5.1. Những điểm chương trình thực hiện được:

Chương trình hoạt động tương đối ổn định. Phần lớn chức năng dự kiến đã hoàn thành. Các chức năng đã hoàn chỉnh:

* Hoàn thành báo cáo và sản phẩm đúng thời hạn.
* Nắm bắt được cơ sở các công nghệ sử dụng.
* Xem danh sách, tìm kiếm tài khoản người dùng.
* Xem chi tiết tài khoản người dùng.
* Thực hiện xóa/khóa tài khoản.
* Xem danh sách, tìm kiếm sinh viên.
* Xem chi tiết thông tin, điểm học cảu sinh viên.
* Thực hiện xóa, thêm mới, cập nhật thông tin sinh viên.
* Xem danh sách, tìm kiếm lớp học phần.
* Thêm mới, cập nhật, xóa lớp học phần.
* Thực hiện thao tác chuyển lớp, xóa sinh viên.
* Thực hiện được việc chuyển đổi đa ngôn ngữ.

## 5.2. Những điểm chương trình chưa thực hiện được:

Ngoài những điều đã đạt được, hệ thống vẫn còn nhiều thiếu sót và chưa hoàn thành được như:

* Thiết kế trang chủ, thống kê thông tin.
* Thiết kế trang thông tin tài khoản đăng nhập.
* Chưa thực hiện được việc tìm kiếm chung bằng ô tìm kiếm trên thanh công cụ.
* Một số trang khác chỉ thực hiện đổ danh sách dữ liệu.

## 5.3. Hướng phát triển:

Để tiếp tục phát triển hệ thống trong tương lai cần nâng cao, cải thiện những điều đã làm đươc, giải quyết và phát triển những điểm chưa thực hiện được.

* Cải thiện giao diện.
* Thêm các chức năng chưa có.
* Tích hợp chức năng chat với tài khoản khác.
* Tích hợp chức năng thông báo khi có yêu cầu thực hiện chức năng nào đó từ tìa khoản có quyền hạn cao hơn.
* Tích hợp với hệ thống đăng ký học.
* Phát triển hệ thống trên nền tảng di động.