**BỘ CÔNG THƯƠNG**

**TRƯỜNG ĐẠI HỌC CÔNG NGHIỆP TP.HCM**

**KHOA CÔNG NGHỆ THÔNG TIN**

**NGÀNH KỸ THUẬT PHẦN MỀM**

A logo with a red and blue letter and a drop of water

Description automatically generated

**Báo cáo Kiến trúc và thiết kế phần mềm**

**HỆ THỐNG QUẢN LÝ LỚP HỌC TÍN CHỈ**

**Môn học:** Kiến trúc và thiết kế phần mềm

**GVHD:** Võ Văn Hải

Trịnh Anh Chung 19494291

Hoàng Tiến Mạnh Đức 19435491

Phạm Hữu Lộc 19507901

**Năm học: 2023-2024**

# LỜI CẢM ƠN

Nhóm chúng em xin gửi lời cảm ơn sâu sắc tới Thầy Hải về sự hỗ trợ và chỉ dẫn quý báu trong môn Kiến trúc và Thiết kế phần mềm.

Trong suốt thời quan vừa qua đã mang lại cho nhóm em những kiến thức bổ ích, nhờ vào sự dẫn dắt tận tình của Thầy. Nhóm em rất biết ơn vì những kiến thức và kỹ năng mà Thầy đã chia sẻ, dạ nhóm em xin chân thành cảm ơn.

Ngoài ra, em cũng xin gửi lời cảm ơn tới tất cả bạn bè trong lớp KTTKPM đã luôn gắn bó, cùng học tập và giúp đỡ em trong những buổi học và đặc biệt em cảm ơn hai bạn đồng hành đã cùng em cố gắng trong suốt quá trình thực hiện hệ thống này.

Em xin chân thành cảm ơn!

# MỤC LỤC

[LỜI CẢM ƠN 2](#_Toc167065978)

[MỤC LỤC 3](#_Toc167065979)

[DANH MỤC CÁC HÌNH VẼ 4](#_Toc167065980)

[CHƯƠNG 1 : MÔ TẢ YÊU CẦU 5](#_Toc167065981)

[1.1 Yêu cầu ban đầu 5](#_Toc167065982)

[1.2 Giả thuyết mới 7](#_Toc167065983)

[CHƯƠNG 2 : KIẾN TRÚC PHẦN MỀM 8](#_Toc167065984)

[2.1 Kiến trúc phần mềm 8](#_Toc167065985)

[2.2 Các nguyên tắc cơ bản 10](#_Toc167065986)

[CHƯƠNG 3 : CƠ SỞ LÝ THUYẾT 11](#_Toc167065987)

[3.1 Kiến trúc là gì 11](#_Toc167065988)

[3.2 Kiến trúc 12](#_Toc167065989)

[3.3 Điểm mạnh và yếu 13](#_Toc167065990)

[CHƯƠNG 4 : Biện luận chọn kiến trúc 14](#_Toc167065991)

[4.1 Kiến trúc đã chọn 14](#_Toc167065992)

[4.2 Lý do sử dụng kiến trúc 14](#_Toc167065993)

[4.3 So sánh với kiến trúc khác 14](#_Toc167065994)

[CHƯƠNG 5 : DEMO VÀ MÔ PHỎNG 15](#_Toc167065995)

[5.1 Trình bày triển khai dự án 15](#_Toc167065996)

[5.1.1 Springboot 16](#_Toc167065997)

[5.1.2 Database MySql 17](#_Toc167065998)

[5.1.3 ReactJs 18](#_Toc167065999)

[Hiện thực 19](#_Toc167066000)

[5.1.4 Trang chủ Admin 19](#_Toc167066001)

[5.1.5 Trang chủ Sinh viên 20](#_Toc167066002)

[5.1.6 Trang chủ giảng viên 20](#_Toc167066003)

[5.1.7 Đăng lý học phần 21](#_Toc167066004)

[5.1.8 Lịch học sinh viên 21](#_Toc167066005)

[5.1.9 Kết quả học tập 22](#_Toc167066006)

[5.1.10 Tra cứu công nợ 22](#_Toc167066007)

[5.1.11 Nhập điểm quá trình học tập 23](#_Toc167066008)

[5.1.12 Hình ảnh code của các project 23](#_Toc167066009)

[5.1.13 Linh video 25](#_Toc167066010)

[CHƯƠNG 6 : KẾT LUẬN 25](#_Toc167066011)

[6.1 Kết quả nhận thấy 25](#_Toc167066012)

[6.2 So với kiến trúc khác 26](#_Toc167066013)

[6.3 Đúc kết lại 26](#_Toc167066014)

# DANH MỤC CÁC HÌNH VẼ

[***Hình 3‑1 Client – Server*** 12](#_Toc167065957)

[***Hình 3‑2 Kiến trúc 3-tiers*** 13](#_Toc167065958)

[***Hình 4‑1 Mô hình kiến trúc 3-tiers áp dụng vào hệ thống*** 14](#_Toc167065959)

[***Hình 5‑1 Spring Boot Framework*** 16](#_Toc167065960)

[***Hình 5‑2 Databse MySQL*** 17](#_Toc167065961)

[***Hình 5‑3 ReactJS Framework*** 18](#_Toc167065962)

[***Hình 5‑4 Màn hình Trang chủ Admin*** 19](#_Toc167065963)

[***Hình 5‑5 Màn hình Trang chủ Sinh viên*** 20](#_Toc167065964)

[***Hình 5‑6 Màn hình Trang chủ giảng viên*** 20](#_Toc167065965)

[***Hình 5‑7 Màn hình Đăng ký học phần*** 21](#_Toc167065966)

[***Hình 5‑8 Màn hình Lịch học sinh viên*** 21](#_Toc167065967)

[***Hình 5‑9 Màn hình Kết quả học tập*** 22](#_Toc167065968)

[***Hình 5‑10 Màn hình Tra cứu công nợ*** 22](#_Toc167065969)

[***Hình 5‑11 Màn hình Nhập điểm quá trình học tập*** 23](#_Toc167065970)

[***Hình 5‑12 Code project LoadBalancer*** 23](#_Toc167065971)

[***Hình 5‑13 Code project Gateway*** 24](#_Toc167065972)

[***Hình 5‑14 Code project Server*** 24](#_Toc167065973)

[***Hình 5‑15 Code project FontEnd*** 25](#_Toc167065974)

# : MÔ TẢ YÊU CẦU

## Yêu cầu ban đầu

* Quản lý sinh viên:

Thông tin cá nhân:

- Họ tên, ngày sinh, giới tính

- Khoa, ngành học, lớp, mã sinh viên

- Địa chỉ, số điện thoại, email, ảnh thẻ,…

Tình trạng học tập:

- Số tín chỉ đã đạt, số môn đã học

- Điểm môn học, điểm trung bình tích lũy

- Học bổng (nếu có), vi phạm kỷ luật (nếu có)

Trạng thái sinh viên:

- Đang học, bảo lưu, nghỉ học, thôi học

Thông tin cựu sinh viên:

- Năm tốt nghiệp, bằng cấp

- Nơi làm việc (nếu có)

* Quản lý môn học:

- Mã môn học: Mã duy nhất cho mỗi môn học

- Tên môn học: Tên đầy đủ của môn học

- Số tín chỉ: Số tín chỉ của môn học

- Khoa/ngành đào tạo: Khoa/ngành quản lý môn học

- Mô tả môn học: Nội dung tóm tắt về môn học

- Giảng viên: Giảng viên phụ trách giảng dạy môn học

- Sách giáo khoa/Tài liệu tham khảo: Danh sách sách giáo khoa và tài liệu tham khảo cho môn học

- Môn học tiên quyết: Danh sách các môn học tiên quyết cần hoàn thành trước khi đăng ký môn học này

- Loại môn học: Bắt buộc, tự chọn

- Số lượng sinh viên tối đa: Số lượng sinh viên tối đa cho mỗi lớp học phần

* Đăng ký học phần:

- Lịch đăng ký: Thời gian bắt đầu và kết thúc đăng ký học phần

- Danh sách lớp học phần: Danh sách các lớp học phần được mở trong học kỳ, bao gồm thông tin về môn học, giảng viên, thời gian học, địa điểm học, số lượng sinh viên hiện tại,…

- Quá trình đăng ký: Sinh viên đăng nhập, chọn học kì đăng kí, chọn môn học, lớp học phần, xác nhận đăng ký

- Quản lý danh sách dự bị: Danh sách sinh viên dự bị cho các lớp học phần đã đầy

- Xác nhận đăng ký: Sinh viên nhận thông báo xác nhận đăng ký và quyết định đóng học phí

* Cung cấp thông tin

- Thông tin sinh viên: Sinh viên tra cứu thông tin học tập của bản thân

- Thông tin môn học: Tra cứu thông tin môn học, giảng viên, thời gian học, điểm số

- Lịch học: Tra cứu lịch học theo tuần

- Thông tin cựu sinh viên: Tra cứu thông tin cựu sinh viên, bằng cấp

Thông báo: Nhận thông báo chung từ nhà trường

* Quản lý quy trình xét tốt nghiệp

- Điều kiện xét tốt nghiệp: Số tín chỉ, điểm số, các môn học bắt buộc

- Đăng ký xét tốt nghiệp: Sinh viên đăng ký xét tốt nghiệp theo quy định

- Kiểm tra điều kiện: Hệ thống kiểm tra điều kiện xét tốt nghiệp cho sinh viên

- Cấp bằng tốt nghiệp: Sinh viên đạt chuẩn được cấp bằng tốt nghiệp

- Lưu trữ thông tin: Lưu trữ thông tin sinh viên tốt nghiệp

* Hỗ trợ các tiện ích khác

- Thay đổi thông tin cá nhân: Sinh viên thay đổi thông tin cá nhân của bản thân

- Thống kê: Hệ thống thống kê số liệu liên quan đến sinh viên, môn học, đăng ký học phần, điểm

- Bảo mật: Hệ thống đảm bảo an toàn thông tin cho sinh viên

## Giả thuyết mới

Ngoài các yêu cầu ban đầu, hệ thống có thể được mở rộng thêm một số tính năng sau:

- Thanh toán học phí trực tuyến: Sinh viên có thể thanh toán học phí trực tuyến qua hệ thống.

- Quản lý tài nguyên học tập: Hệ thống cung cấp cho sinh viên quyền truy cập vào các tài nguyên học tập trực tuyến, như bài giảng, bài tập, tài liệu tham khảo.

- Kết nối với các hệ thống khác: Hệ thống có thể kết nối với các hệ thống khác của trường đại học, như hệ thống quản lý sinh viên, hệ thống quản lý thư viện, hệ thống lms của trường.

- Phân tích dữ liệu: Hệ thống có thể phân tích

# : KIẾN TRÚC PHẦN MỀM

## Kiến trúc phần mềm

Kiến trúc phần mềm (Software Architecture) là một tập hợp các cấu trúc cần thiết để lập trình lên hệ thống phần mềm, bao gồm các phần tử, mối quan hệ giữa chúng và các thuộc tính của chúng. Đồng thời, kiến trúc phần mềm là việc ứng dụng nền móng để xây dựng dự án phần mềm. Nói cách khác, kiến trúc phần mềm là bản thiết kế của một hệ thống phần mềm, mô tả cấu trúc tổng thể của hệ thống và cách thức các phần mềm con tương tác với nhau.

Kiến trúc phần mềm bao gồm các yếu tố sau:

- Cấu trúc phần mềm: Mô tả cách thức các phần mềm con được tổ chức và sắp xếp trong hệ thống.

- Giao diện phần mềm: Mô tả cách thức các phần mềm con tương tác và trao đổi thông tin với nhau.

- Thuộc tính phần mềm: Mô tả các tính chất quan trọng của hệ thống phần mềm, như hiệu suất, bảo mật, khả năng mở rộng, khả năng bảo trì...

- Quy tắc thiết kế: Quy định các nguyên tắc và hướng dẫn cho việc thiết kế và phát triển các phần mềm con trong hệ thống.

Lợi ích của việc thiết kế kiến trúc phần mềm:

- Tăng tính hiệu quả: Kiến trúc phần mềm giúp tối ưu hóa hiệu suất của hệ thống phần mềm, đảm bảo hệ thống hoạt động nhanh chóng và hiệu quả.

- Tăng tính bảo mật: Kiến trúc phần mềm giúp bảo vệ hệ thống phần mềm khỏi các mối đe dọa an ninh mạng, đảm bảo tính bảo mật cho dữ liệu và thông tin của người dùng.

- Tăng tính khả dụng: Kiến trúc phần mềm giúp đảm bảo hệ thống phần mềm hoạt động ổn định và liên tục, hạn chế tối đa thời gian gián đoạn.

- Tăng tính khả năng mở rộng: Kiến trúc phần mềm giúp hệ thống phần mềm dễ dàng mở rộng để đáp ứng nhu cầu tăng trưởng về dữ liệu và người dùng.

- Tăng tính khả năng bảo trì: Kiến trúc phần mềm giúp hệ thống phần mềm dễ dàng bảo trì và sửa lỗi, tiết kiệm thời gian và chi phí cho việc bảo trì hệ thống.

Quy trình thiết kế kiến trúc phần mềm:

Quy trình thiết kế kiến trúc phần mềm thường bao gồm các bước sau:

- Phân tích yêu cầu: Xác định các yêu cầu chức năng và phi chức năng của hệ thống phần mềm.

- Thiết kế mô hình kiến trúc: Xây dựng mô hình kiến trúc mô tả cấu trúc tổng thể của hệ thống phần mềm.

- Lựa chọn công nghệ: Lựa chọn các công nghệ phù hợp để triển khai hệ thống phần mềm.

- Thiết kế chi tiết: Thiết kế chi tiết các phần mềm con trong hệ thống phần mềm.

- Thực hiện: Viết mã và triển khai hệ thống phần mềm.

- Kiểm thử: Kiểm thử hệ thống phần mềm để đảm bảo đáp ứng các yêu cầu và hoạt động chính xác.

- Bảo trì: Bảo trì hệ thống phần mềm để sửa lỗi và nâng cấp tính năng.

Các mô hình kiến trúc phần mềm phổ biến:

Có nhiều mô hình kiến trúc phần mềm khác nhau, mỗi mô hình phù hợp với các loại hệ thống phần mềm và nhu cầu cụ thể. Một số mô hình kiến trúc phần mềm phổ biến bao gồm:

- Kiến trúc hướng lớp (Layered architecture): Chia hệ thống phần mềm thành các lớp riêng biệt, mỗi lớp có chức năng và trách nhiệm cụ thể.

- Kiến trúc hướng dịch vụ (Service-oriented architecture - SOA): Hệ thống phần mềm được xây dựng như một tập hợp các dịch vụ độc lập, có thể tương tác với nhau qua các giao diện được định nghĩa rõ ràng.

- Kiến trúc hướng sự kiện (Event-driven architecture - EDA): Hệ thống phần mềm sử dụng các sự kiện để kích hoạt và điều phối các hành động.

- Kiến trúc vi mô (Microservices architecture): Hệ thống phần mềm được chia thành các dịch vụ nhỏ, độc lập, có thể triển khai và quản lý riêng biệt.

## Các nguyên tắc cơ bản

- Tính mô đun (Modularity): Hệ thống được chia thành các mô đun riêng biệt, dễ dàng bảo trì và mở rộng. Mỗi mô đun có chức năng và nhiệm vụ cụ thể, được kết nối với nhau thông qua các giao diện và được định nghĩa rõ ràng.

- Tính linh hoạt (Flexibility): Hệ thống nên dễ dàng thay đổi, mở rộng và bảo trì.

- Khả năng bảo mật (Security): Hệ thống cần đảm bảo an toàn thông tin cho dữ liệu của sinh viên và nhà trường.

- Khả năng mở rộng (Scalability): Hệ thống có thể mở rộng để đáp ứng nhu cầu tăng trưởng về dữ liệu và người dùng. Ví dụ, có thể nâng cấp phần cứng, phần mềm hoặc thay đổi cấu trúc cơ sở dữ liệu để đáp ứng nhu cầu sử dụng ngày càng cao.

- Khả năng sử dụng (Usability): Hệ thống cần có giao diện người dùng thân thiện, dễ sử dụng cho cả sinh viên và cán bộ quản lý.

- Tính sẵn sàng (Availability): Hệ thống hoạt động ổn định và liên tục, hạn chế tối đa thời gian gián đoạn. Có thể áp dụng các biện pháp dự phòng như sao lưu dữ liệu, cân bằng tải và phục hồi sau thảm họa để đảm bảo tính sẵn sàng của hệ thống.

# : CƠ SỞ LÝ THUYẾT

## Kiến trúc là gì

Kiến trúc hệ thống đóng vai trò nền tảng cho việc xây dựng và vận hành một hệ thống thành công. Nó là bản mô tả chi tiết về cấu trúc, hành vi và các khía cạnh khác của hệ thống, bao gồm các thành phần, cách thức tương tác và mối quan hệ với môi trường xung quanh. Việc thiết kế kiến trúc hệ thống bài bản đảm bảo hệ thống đáp ứng được các yêu cầu về chức năng, hiệu suất, khả năng mở rộng, bảo mật và khả năng bảo trì.

Kiến trúc hệ thống đóng vai trò quan trọng trong việc phát triển và vận hành các hệ thống phần mềm, mang lại nhiều lợi ích thiết thực sau:

* Xác định giải pháp tối ưu: Giúp xác định cấu trúc tổng thể của hệ thống, bao gồm các thành phần, mối quan hệ và cách thức hoạt động của chúng
* Tối ưu hiệu suất và bảo mật: Cho phép phân bổ tài nguyên hợp lý, đảm bảo hệ thống hoạt động trơn tru, hiệu quả và đáp ứng nhu cầu sử dụng.
* Nâng cao chất lượng phần mềm: Thúc đẩy việc áp dụng các nguyên tắc thiết kế phần mềm tốt nhất, đảm bảo tính mô đun, khả năng tái sử dụng và dễ bảo trì.
* Tăng cường khả năng cộng tác: Cung cấp một bức tranh toàn cảnh về hệ thống, giúp các bên liên quan dễ dàng hiểu và thảo luận về thiết kế.
* Đơn giản hóa việc quản lý hệ thống: Cho phép theo dõi và giám sát hiệu suất hệ thống một cách dễ dàng và hiệu quả

Nhìn chung, kiến trúc hệ thống đóng vai trò nền tảng cho việc phát triển và vận hành các hệ thống phần mềm thành công. Nó giúp đảm bảo hệ thống đáp ứng đầy đủ các yêu cầu, hoạt động hiệu quả, bảo mật và dễ dàng quản lý, từ đó mang lại lợi ích thiết thực cho doanh nghiệp và người dùng.

## Kiến trúc

Kiến trúc Client – Server là hệ thống gồm 2 loại phần tử chức năng: Server cung cấp 1 số dịch vụ, client là phần tử sử dụng dịch vụ bằng cách truy suất đến server tương ứng.

A diagram of a cloud network

Description automatically generated

***Hình 3‑1 Client – Server***

Kiến trúc 3 đối tác (3-tiers Architecture) là sự cả tiến của kiến trúc Client – Server. Kiến trúc 3-tiers chia hệ thống chia thành ba phần riêng biệt:

* Tầng biểu diễn (Presentation Layer): cung cấp giao diện người dùng cho người dùng tương tác với hệ thống, hiển thị thông tin và kết quả xử lý từ tầng logic, thu thập dữ liệu nhập từ người dùng và truyền đến tầng logic. Tầng này gồm các thành phần như: giao diện web, ứng dụng di động, ...
* Tầng logic (Business Logic Layer): thực hiện các nghiệp vụ chính của hệ thống, xử lý các quy tắc business và logic xử lý nghiệp vụ, truy cập và thao tác dữ liệu thông qua tầng truy cập dữ liệu. Tầng này gồm các thành phần như: services, APIs, ...
* Tầng truy cập dữ liệu (Data Access Layer): tương tác với cơ sở dữ liệu để lưu trữ, truy xuất và quản lý dữ liệu, che giấu các chi tiết phức tạp của cơ sở dữ liệu khỏi tầng logic, cung cấp các hàm truy cập dữ liệu cho tầng logic. Nó bao gồm các cơ sở dữ liệu máy chủ như- Oracle, MySQL, DB2, …



***Hình 3‑2 Kiến trúc 3-tiers***

## Điểm mạnh và yếu

**Ưu điểm**:

* Đơn giản: Dễ dàng thiết kế, triển khai và quản lý.
* Có thể mở rộng: Có thể mở rộng bằng cách thêm nhiều server.
* Bảo mật: Dữ liệu được lưu trữ tập trung trên server, giúp tăng cường bảo mật.

**Nhược điểm**:

* Khó khăn trong việc phát triển: Phát triển ứng dụng client-server phức tạp có thể khó khăn hơn do sự phụ thuộc chặt chẽ giữa client và server.
* Khó khăn trong việc triển khai: Việc triển khai ứng dụng client-server có thể phức tạp hơn do cần quản lý cả client và server.
* Ít linh hoạt: Khó khăn trong việc thay đổi hoặc cập nhật ứng dụng mà không ảnh hưởng đến cả client và server.

# : Biện luận chọn kiến trúc

## Kiến trúc đã chọn

A screenshot of a computer

Description automatically generated

***Hình 4‑1 Mô hình kiến trúc 3-tiers áp dụng vào hệ thống***

## Lý do sử dụng kiến trúc

Lý do chọn 3 – tiers bởi vì:

* **Tính mô đun hóa cao**: Chia tách hệ thống thành 3 lớp (giao diện người dùng, logic nghiệp vụ và truy cập dữ liệu) giúp dễ dàng phát triển, bảo trì và mở rộng.
* **Khả năng bảo mật tốt**: Việc tách biệt các lớp giúp hạn chế truy cập trái phép vào dữ liệu và logic nghiệp vụ.
* **Khả năng tái sử dụng cao**: Các thành phần trong mỗi lớp có thể được tái sử dụng cho các dự án khác nhau.
* **Dễ dàng triển khai và vận hành**: Kiến trúc đơn giản, dễ hiểu giúp việc triển khai và vận hành hệ thống trở nên dễ dàng hơn.

## So sánh với kiến trúc khác

Sau khi tìm hiểu, tham khảo các loại kiến trúc, so sánh kiến trúc 3-tier với các kiến trúc khác cụ thể:

|  |  |  |
| --- | --- | --- |
| **Kiến trúc** | **Ưu điểm** | **Nhược điểm** |
| **3-tier** | Mô đun hóa cao, bảo mật tốt, tái sử dụng cao, dễ triển khai | Hiệu suất có thể bị ảnh hưởng do giao tiếp giữa các tầng |
| **Microservice** | Khả năng mở rộng cao, linh hoạt, dễ dàng bảo trì, độ tin cậy cao | Phức tạp hơn, việc giao tiếp khó, dễ bị tấn công |
| **Client-server** | Đơn giản, dễ phát triển, hiệu suất cao | Khó mở rộng, bảo trì phức tạp, bảo mật thấp |
| **Monolithic** | Giao tiếp dễ dàng, đơn giản, dễ dàng triển khai | Khó mở rộng, khó bảo trì |

# : DEMO VÀ MÔ PHỎNG

## Trình bày triển khai dự án

**Một số công nghệ sử dụng hiện thực dự án:**

### Springboot



***Hình 5‑1*** ***Spring Boot Framework***

Lí do chọn Spring boot:

Đơn giản và nhanh chóng triển khai: Spring Boot giúp đơn giản hóa quá trình triển khai ứng dụng bằng cách cung cấp một cấu hình mặc định tự động cho các thành phần quan trọng. Điều này giúp giảm thời gian và công sức trong việc cài đặt và cấu hình.

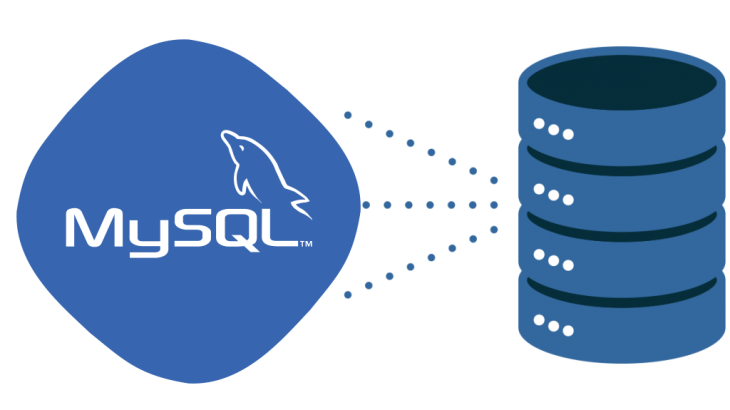
Hỗ trợ mạnh mẽ cho Spring Framework: Spring Boot là một phần mở rộng của Spring Framework, nó tận dụng các tính năng mạnh mẽ của Spring Framework như Dependency Injection, Spring MVC, Spring Data, và Spring Security. Điều này giúp cho việc phát triển ứng dụng dễ dàng hơn và tận dụng được các thành phần có sẵn của Spring.

Tích hợp dễ dàng với các công nghệ khác: Spring Boot cung cấp tích hợp dễ dàng với nhiều công nghệ và framework phổ biến khác như Hibernate, JPA, Thymeleaf, ReactJS, AngularJS, v.v. Điều này giúp tận dụng các công nghệ phù hợp để phát triển ứng dụng quản lý đào tạo sinh viên theo hệ thống tín chỉ trường đại học một cách linh hoạt và tiện lợi.

Hỗ trợ phát triển ứng dụng RESTful API: Spring Boot cung cấp hỗ trợ mạnh mẽ cho việc phát triển các ứng dụng RESTful API thông qua Spring MVC và Spring Data REST. Điều này cho phép dễ dàng tạo ra các endpoint API để quản lý dữ liệu và tương tác với ứng dụng từ các thiết bị và nền tảng khác nhau.

Quản lý tốt các thành phần và phụ thuộc: Spring Boot hỗ trợ quản lý các phụ thuộc và thành phần của ứng dụng một cách tự động thông qua Spring Boot Starter và Spring Boot Actuator. Điều này giúp giảm thiểu sự phức tạp trong việc quản lý các thư viện và phiên bản, đồng thời cung cấp khả năng giám sát và quản lý ứng dụng một cách dễ dàng.

### Database MySql



***Hình 5‑2*** ***Databse MySQL***

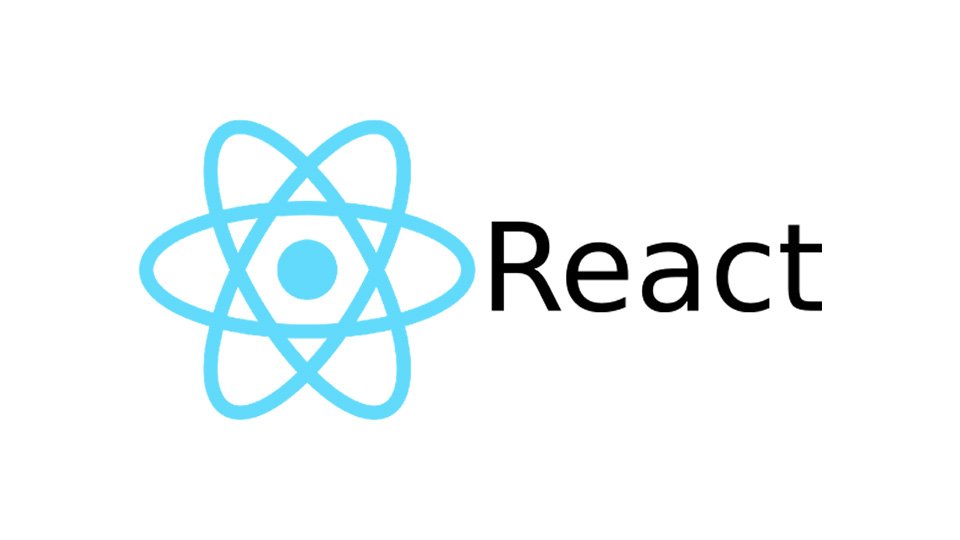
* Lí do chọn Database MySql:

MySQL cung cấp các tính năng mạnh mẽ để quản lý dữ liệu trong môi trường hệ thống trường đại học, bao gồm khả năng lưu trữ và truy xuất dữ liệu hiệu quả, hỗ trợ giao diện người dùng đơn giản và linh hoạt.

Ngoài ra, MySQL cũng có khả năng mở rộng, cho phép mở rộng cấu trúc cơ sở dữ liệu và xử lý các tác vụ phức tạp như quản lý lịch hẹn, báo cáo và đồ thị thống kê.

Bằng cách sử dụng MySQL, chúng ta có thể xây dựng và duy trì một cơ sở dữ liệu đáng tin cậy và hiệu quả, phục vụ tốt cho nhu cầu quản lý thông tin trong quản lý đào tạo sinh viên theo hệ thống tín chỉ trường đại học.

### ReactJs



***Hình 5‑3*** ***ReactJS Framework***

Lí do chọn ReactJS:

Hiệu suất cao: ReactJS sử dụng cơ chế virtual DOM, cho phép cập nhật chỉ các thành phần cần thiết thay vì toàn bộ trang web. Điều này giúp cải thiện hiệu suất và tăng tốc độ tải trang.

Tính linh hoạt và tái sử dụng: ReactJS cho phép chia giao diện thành các thành phần độc lập (component-based), giúp dễ dàng quản lý và tái sử dụng mã nguồn. Điều này rất hữu ích trong việc xây dựng các trang web phức tạp như hệ thống quản lý đào tạo sinh viên theo hệ thống tín chỉ trường đại học.

Cộng đồng hỗ trợ mạnh mẽ: ReactJS có một cộng đồng phát triển đông đảo và nhiều tài liệu hướng dẫn phong phú. Điều này giúp dễ dàng tìm kiếm giải pháp, tham khảo các ví dụ và nhận được hỗ trợ từ cộng đồng khi gặp khó khăn trong quá trình phát triển.

Tương thích với các công nghệ khác: ReactJS có khả năng tương thích tốt với các thư viện và framework khác như Redux, React Router, Material-UI, giúp mở rộng tính năng và thiết kế giao diện linh hoạt hơn.

Trải nghiệm người dùng tốt: Nhờ cơ chế virtual DOM và khả năng tương tác linh hoạt, ReactJS cung cấp trải nghiệm người dùng mượt mà và tương tác đáng tin cậy trên các trình duyệt và thiết bị khác nhau.

## Hiện thực

### Trang chủ Admin

A screenshot of a computer

Description automatically generated

***Hình 5‑4 Màn hình Trang chủ Admin***

### Trang chủ Sinh viên

A screenshot of a computer

Description automatically generated

***Hình 5‑5 Màn hình Trang chủ Sinh viên***

### Trang chủ giảng viên

A screenshot of a computer

Description automatically generated

***Hình 5‑6 Màn hình Trang chủ giảng viên***

### Đăng lý học phần

A screenshot of a computer

Description automatically generated

***Hình 5‑7 Màn hình Đăng ký học phần***

### Lịch học sinh viên

A screenshot of a computer

Description automatically generated

***Hình 5‑8 Màn hình Lịch học sinh viên***

### Kết quả học tập

***A screenshot of a computer

Description automatically generated***

***Hình 5‑9 Màn hình Kết quả học tập***

### Tra cứu công nợ

A screenshot of a computer

Description automatically generated

***Hình 5‑10 Màn hình Tra cứu công nợ***

### Nhập điểm quá trình học tập

A screenshot of a computer

Description automatically generated

***Hình 5‑11 Màn hình Nhập điểm quá trình học tập***

### Hình ảnh code của các project

A screenshot of a computer program

Description automatically generated

***Hình 5‑12 Code project LoadBalancer***

A screenshot of a computer program

Description automatically generated

***Hình 5‑13 Code project Gateway***

A screenshot of a computer program

Description automatically generated

***Hình 5‑14 Code project Server***

A screenshot of a computer program

Description automatically generated

***Hình 5‑15 Code project FontEnd***

### Linh video

https://drive.google.com/drive/folders/1TKhTFkehs23UQV--9XlXmngrs\_oV4TuG?usp=drive\_link

# : KẾT LUẬN

## Kết quả nhận thấy

Sau khi sử dụng kiến trúc 3-tiers vào hệ thống thì em nhận thấy rằng kiến trúc này có rất nhiều đặc tính hay như:

- Tính mô đun hóa cao: Việc chia tách hệ thống thành 3 lớp rõ ràng giúp dễ dàng quản lý và phát triển từng phần riêng biệt, từ giao diện người dùng đến logic nghiệp vụ và truy cập dữ liệu.

- Khả năng bảo mật tốt: Sự phân tách giữa các lớp giúp hạn chế truy cập trái phép vào dữ liệu và logic nghiệp vụ, tăng cường tính bảo mật cho hệ thống.

- Khả năng tái sử dụng cao: Các thành phần trong mỗi lớp có thể được tái sử dụng cho các dự án khác nhau, giúp tiết kiệm thời gian và công sức trong việc phát triển phần mềm.

- Dễ dàng triển khai và vận hành: Kiến trúc đơn giản, rõ ràng, dễ hiểu giúp việc triển khai và vận hành hệ thống trở nên dễ dàng và hiệu quả hơn.

## So với kiến trúc khác

So sánh với các kiến trúc khác như Microservice, Client-server, và Monolithic, kiến trúc 3-tiers có những ưu điểm riêng biệt như sự đơn giản, tính mô đun hóa cao, khả năng bảo mật tốt.

Tuy nhiên, nó vẫn có những mặt hạn chế về hiệu suất có thể bị ảnh hưởng do giao tiếp giữa các tầng, đặc biệt là trong môi trường có lượng dữ liệu lớn và yêu cầu xử lý cao.

## Đúc kết lại

Tóm lại, mỗi kiến trúc đều có những điểm mạnh và điểm yếu riêng, và sự lựa chọn kiến trúc phụ thuộc vào các yêu cầu cụ thể của dự án.

Và ở đây nhóm em chọn kiến trúc 3-tiers để thực hiện dự án vì một lựa chọn phổ biến và hợp lý trong nhiều trường hợp nhờ tính mô đun hóa cao và khả năng bảo mật tốt.