**TRƯỜNG ĐẠI HỌC TRÀ VINH**

**KHOA KỸ THUẬT VÀ CÔNG NGHỆ**

****

**ISO 9001:2015**

**LÂM NGỌC TÀI**

**XÂY DỰNG HỆ THỐNG TRỰC QUAN HÓA KẾT QUẢ HỌC TẬP SINH VIÊN CNTT**

**ĐỒ ÁN TỐT NGHIỆP**

**NGÀNH CÔNG NGHỆ THÔNG TIN**

**TRÀ VINH, NĂM 2024**

**TRƯỜNG ĐẠI HỌC TRÀ VINH**

**KHOA KỸ THUẬT VÀ CÔNG NGHỆ**

**XÂY DỰNG HỆ THỐNG TRỰC QUAN HÓA KẾT QUẢ HỌC TẬP SINH VIÊN CNTT**

**ĐỒ ÁN TỐT NGHIỆP**

**NGÀNH CÔNG NGHỆ THÔNG TIN**

Sinh viên: **Lâm Ngọc Tài**

Lớp: **DA20TTB**

MSSV: **110120152**

GVHD: **TS. Nguyễn Bảo Ân**

**TRÀ VINH, NĂM 2024**

**LỜI MỞ ĐẦU**

Trong quá trình lựa chọn đề tài, tôi quyết định chọn xây dựng hệ thống trực quan hóa kết quả học tập sinh viên công nghệ thông tin. Lý do tôi chọn đề tài này là vì hiện nay, việc đánh giá và theo dõi kết quả học tập của sinh viên thường gặp nhiều khó khăn, đặc biệt trong việc trình bày dữ liệu một cách trực quan và dễ hiểu. Hệ thống trực quan hóa kết quả học tập sẽ giúp cho giảng viên và sinh viên dễ dàng nhận biết được những điểm mạnh và yếu trong quá trình học tập, từ đó có những biện pháp cải thiện kịp thời.

Trong đề tài này, tôi sẽ vận dụng những kiến thức đã học về lập trình, cơ sở dữ liệu, và trực quan hóa dữ liệu để xây dựng một hệ thống hoàn chỉnh, đáp ứng được nhu cầu thực tế của nhà trường và sinh viên.

Đề tài này gồm 5 chương, với nội dung cụ thể như sau:

**Chương 1: Đặt vấn đề**  
Trình bày lý do chọn đề tài, mục tiêu nghiên cứu, nội dung nghiêm cứu, đối tượng, phạm vị và phương pháp tiếp cận.

**Chương 2: Cơ sở lý thuyết**

Trình bày các kiến thức cơ bản và các công nghệ sẽ sử dụng trong đề tài, bao gồm lập trình, cơ sở dữ liệu, và trực quan hóa dữ liệu.

**Chương 3: Thực hiện hóa nghiêm cứu**  
Mô tả quá trình triển khai hệ thống, các công cụ và ngôn ngữ lập trình sử dụng, cùng với các bước kiểm thử hệ thống.

**Chương 4: Kết quả nghiêm cứu**  
Chương này mô tả quá trình triển khai hệ thống, bao gồm việc phát triển, kiểm thử và cài đặt hệ thống trực quan hóa kết quả học tập.

**Chương 5: Kết luận và hướng phát triển**  
Đánh giá hiệu quả của hệ thống đã xây dựng, nêu ra những ưu điểm và nhược điểm, cùng với những hướng phát triển trong tương lai.

**LỜI CẢM ƠN**

Trước hết, em xin gửi lời cảm ơn đến toàn thể quý thầy cô, giảng viên Trường Đại học Trà Vinh, đặc biệt là các thầy cô ở Khoa Kỹ thuật & Công nghệ, bộ môn Công nghệ thông tin, đã tạo điều kiện tốt nhất để em hoàn thành trọn vẹn bài đồ án cơ sở ngành này.

Tiếp theo, em xin tỏ lòng biết ơn sâu sắc đến thầy Phạm Minh Đương – Giảng viên Khoa Kỹ thuật & Công nghệ, Trường Đại học Trà Vinh, trong quá trình hướng dẫn đã vô cùng tâm huyết trong việc truyền đạt kiến thức đến em.

Trong bài báo cáo, do lượng kiến thức và kinh ngiệm còn khiêm tốn, thời gian nghiên cứu ngắn nên vẫn còn một số sai sót nhỏ không đáng kể. Do đó, em kính mong quý thầy cô thông cảm, góp ý để em có thể tiếp thu và cải thiện cho những nghiên cứu trong tương lai.

Sau tất cả, kính chúc các thầy cô luôn dồi dào sức khoẻ.

Em xin chân thành cảm ơn!

*Trà Vinh, ngày …… tháng 6 năm 2024*

Sinh viên thực hiện

**Lâm Ngọc Tài**

**NHẬN XÉT**

………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………

**Giảng viên hướng dẫn**

(ký và ghi rõ họ tên)

UBND TỈNH TRÀ VINH **CỘNG HOÀ XÃ HỘI CHỦ NGHĨA VIỆT NAM**

**TRƯỜNG ĐẠI HỌC TRÀ VINH Độc lập – Tự do – Hạnh Phúc**

**BẢN NHẬN XÉT ĐỒ ÁN, KHÓA LUẬN TỐT NGHIỆP**

Họ và tên sinh viên: MSSV:

Ngành: Khóa:

Tên đề tài:

Họ và tên Giáo viên hướng dẫn:

Chức danh: Học vị:

**NHẬN XÉT**

1. Nội dung đề tài:

1. Ưu điểm:

1. Khuyết điểm:

1. Điểm mới đề tài:

1. Giá trị thực trên đề tài:

7. Đề nghị sửa chữa bổ sung:

8. Đánh giá:

Trà Vinh*, ngày …… tháng …… năm 2024*

Giảng viên hướng dẫn

*(Ký & ghi rõ họ tên)*

**NHẬN XÉT**

………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………

…………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………

**Giảng viên chấm**

(ký và ghi rõ họ tên)

UBND TỈNH TRÀ VINH **CỘNG HOÀ XÃ HỘI CHỦ NGHĨA VIỆT NAM**

**TRƯỜNG ĐẠI HỌC TRÀ VINH Độc lập - Tự do - Hạnh phúc**

**BẢN NHẬN XÉT ĐỒ ÁN, KHÓA LUẬN TỐT NGHIỆP**

Họ và tên người nhận xét:

Chức danh: Học vị:

Chuyên ngành:

Cơ quan công tác:

Tên sinh viên:

Tên đề tài đồ án, khóa luận tốt nghiệp:

**I. Ý KIẾN NHẬN XÉT**

1. Nội dung:

2. Điểm mới các kết quả của đồ án, khóa luận:

3. Ứng dụng thực tế:

**II. CÁC VẤN ĐỀ CẦN LÀM RÕ**

**III. KẾT LUẬN**

*……………, ngày …… tháng …… năm 2024*

Người nhận xét

*(Ký & ghi rõ họ tên)*

**MỤC LỤC**

[CHƯƠNG 1. ĐẶT VẤN ĐỀ 1](#_Toc164766699)

[1.1. Lý do chọn đề tài 1](#_Toc164766700)

[1.2. Mục tiêu 1](#_Toc164766701)

[1.3. Nội dung 1](#_Toc164766702)

[1.4. Đối tượng và phạm vi nghiên cứu 1](#_Toc164766703)

[1.5. Phương pháp nghiên cứu 1](#_Toc164766704)

[CHƯƠNG 2. CƠ SỞ LÝ THUYẾT 2](#_Toc164766705)

[2.1. ...... 2](#_Toc164766706)

[*2.1.1. .......* 2](#_Toc164766707)

[*2.1.2. .......* 2](#_Toc164766708)

[2.2. ........ 2](#_Toc164766709)

[*2.2.1. ....* 2](#_Toc164766710)

[*2.2.2. .....* 2](#_Toc164766711)

[2.3. ....... 2](#_Toc164766712)

[CHƯƠNG 3. HIỆN THỰC HÓA NGHIÊN CỨU 3](#_Toc164766713)

[3.1. ...... 3](#_Toc164766714)

[*3.1.1. .......* 3](#_Toc164766715)

[*3.1.2. .......* 3](#_Toc164766716)

[3.2. .... 3](#_Toc164766717)

[*3.2.1. ....* 3](#_Toc164766718)

[CHƯƠNG 4. KẾT QUẢ NGHIÊN CỨU 4](#_Toc164766719)

[4.1. ...... 4](#_Toc164766720)

[*4.1.1. ......* 4](#_Toc164766721)

[*4.1.2. ....* 4](#_Toc164766722)

[4.2. ...... 4](#_Toc164766723)

[*4.2.1. .......* 4](#_Toc164766724)

[*4.2.2. .......* 4](#_Toc164766725)

[CHƯƠNG 5. KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN 5](#_Toc164766726)

[5.1. Kết luận 5](#_Toc164766727)

[5.2. Hướng phát triển 5](#_Toc164766728)

[DANH MỤC TÀI LIỆU THAM KHẢO 6](#_Toc164766729)

**MỤC LỤC ẢNH**

[Hình 3.3.1. Sơ đồ use case 16](#_Toc170545161)

**Mục Lục Bảng**

[Bảng 3.1. Mô tả Actor 16](#_Toc170579896)

[Bảng 3.2 Mô tả use case 17](#_Toc170579897)

[Bảng 3.3. Mô tả bảng “Student” 19](#_Toc170579898)

[Bảng 3.4. Mô tả bảng “Teacher” 19](#_Toc170579899)

[Bảng 3.5. Môn tả bảng “Subjects” 20](#_Toc170579900)

[Bảng 3.6. Mô tả bảng “Courses” 21](#_Toc170579901)

[Bảng 3.7. Mô tả bảng “academic\_year” 21](#_Toc170579902)

[Bảng 3.8. Mô tả bảng “Semesters” 22](#_Toc170579903)

[Bảng 3.9. Mô tả bảng “semester\_academic\_years” 22](#_Toc170579904)

[Bảng 3.10. Mô tả bảng “Accessments” 22](#_Toc170579905)

[Bảng 3.11. Mô tả bảng “assessmentItems” 23](#_Toc170579906)

[Bảng 3.12. Mô tả bảng “Chapters” 24](#_Toc170579907)

[Bảng 3.13. Mô tả bảng “Classes” 24](#_Toc170579908)

[Bảng 3.14. Mô tả bảng “Clos” 25](#_Toc170579909)

[Bảng 3.15. Mô tả bảng “Plos” 25](#_Toc170579910)

[Bảng 3.16. Mô tả bảng “Pos” 25](#_Toc170579911)

[Bảng 3.17. Mô tả bảng “course\_enrollment” 26](#_Toc170579912)

[Bảng 3.18. Mô tả bảng “Programs” 26](#_Toc170579913)

[Bảng 3.19. Mô tả bảng “rubricItems” 27](#_Toc170579914)

[Bảng 3.20. Mô tả bảng “map\_ clo\_chapters” 27](#_Toc170579915)

[Bảng 3.21. Mô tả bảng “map\_plo\_clo” 28](#_Toc170579916)

[Bảng 3.22. Mô tả bảng “map\_po\_plo” 28](#_Toc170579917)

[Bảng 3.23. Mô tả bảng “refresh\_tokens” 28](#_Toc170579918)

**DANH MỤC TỪ VIẾT TẮT**

(Sắp xếp danh mục từ viết tắt theo thứ tự alphabet của từ viết tắt)

|  |  |
| --- | --- |
| **Từ viết tắt** | **Ý nghĩa** |
|  |  |
|  |  |
|  |  |

thời gian trung bình

# ĐẶT VẤN ĐỀ

## Lý do chọn đề tài

Trong bối cảnh giáo dục hiện đại, việc theo dõi và đánh giá kết quả học tập của sinh viên là một yếu tố quan trọng để nâng cao chất lượng giảng dạy và học tập. Tuy nhiên, phương pháp truyền thống thường gặp nhiều hạn chế về mặt thời gian và công sức. Hệ thống trực quan hóa kết quả học tập giúp cung cấp thông tin nhanh chóng, chính xác và dễ hiểu, từ đó hỗ trợ quá trình ra quyết định của giảng viên và sinh viên.

## Mục tiêu

Phát triển một hệ thống trực quan hóa kết quả học tập của sinh viên công nghệ thông tin.

Cung cấp giao diện thân thiện và dễ sử dụng cho sinh viên và giảng viên.

Hỗ trợ giảng viên trong việc đánh giá và quản lý kết quả học tập của sinh viên.

Nâng cao khả năng đánh giá và theo dõi tiến độ học tập của sinh viên.

## Nội dung

Đề tài này bao gồm các nội dung chính sau:

* Tổng quan về đề tài và tầm quan trọng của việc trực quan hóa kết quả học tập.
* Cơ sở lý thuyết và các công nghệ liên quan đến việc xây dựng hệ thống trực quan hóa dữ liệu.
* Quá trình phân tích yêu cầu và thiết kế hệ thống.
* Quá trình triển khai, kiểm thử và cài đặt hệ thống trực quan hóa kết quả học tập.
* Đánh giá hiệu quả của hệ thống và đề xuất các hướng phát triển tiếp theo.

## Đối tượng và phạm vi nghiên cứu

Đối tượng nghiên cứu của đề tài này là các sinh viên và giảng viên thuộc ngành công nghệ thông tin. Phạm vi nghiên cứu bao gồm:

* Các phương pháp và công nghệ sử dụng trong việc trực quan hóa dữ liệu.
* Thiết kế và triển khai hệ thống trực quan hóa kết quả học tập.
* Đánh giá hiệu quả của hệ thống trong môi trường giáo dục thực tế.

## Phương pháp nghiên cứu

Nghiên cứu tài liệu: Tìm hiểu và tổng hợp các tài liệu liên quan đến trực quan hóa dữ liệu và các hệ thống quản lý kết quả học tập.

Phân tích yêu cầu: Khảo sát và thu thập ý kiến từ sinh viên và giảng viên để xác định các yêu cầu của hệ thống.

Thiết kế hệ thống: Thiết kế kiến trúc tổng thể và các thành phần chi tiết của hệ thống.

Triển khai và kiểm thử: Phát triển, kiểm thử và cài đặt hệ thống trong môi trường thực tế.

Đánh giá và cải tiến: Đánh giá hiệu quả của hệ thống và đưa ra các đề xuất cải tiến dựa trên phản hồi từ người dùng.

# CƠ SỞ LÝ THUYẾT

## Tìm hiểu về NodeJS

Node.js là một môi trường chạy mã JavaScript phía máy chủ, được xây dựng trên nền tảng của công cụ JavaScript V8 của Google Chrome. Node.js cho phép các nhà phát triển xây dựng các ứng dụng mạng hiệu suất cao và mở rộng dễ dàng.

### *Khái niệm và lịch sử phát triển*

**Khái niệm:**

Node.js là một nền tảng mã nguồn mở, đa nền tảng, cho phép chạy mã JavaScript bên ngoài trình duyệt. Được sử dụng chủ yếu để phát triển các ứng dụng mạng thời gian thực với kiến trúc hướng sự kiện (event-driven) và không đồng bộ (asynchronous).

**Lịch sử phát triển:**

Ryan Dahl là người sáng lập ra Node.js vào năm 2009.

Node.js được xây dựng trên công cụ JavaScript V8 của Google Chrome, cho phép thực thi mã JavaScript với hiệu suất cao.

Ban đầu, Node.js được tạo ra nhằm giải quyết vấn đề hiệu suất và khả năng mở rộng của các máy chủ web truyền thống.

### *Kiến trúc và hoạt động của NodeJS*

**Kiến trúc:**

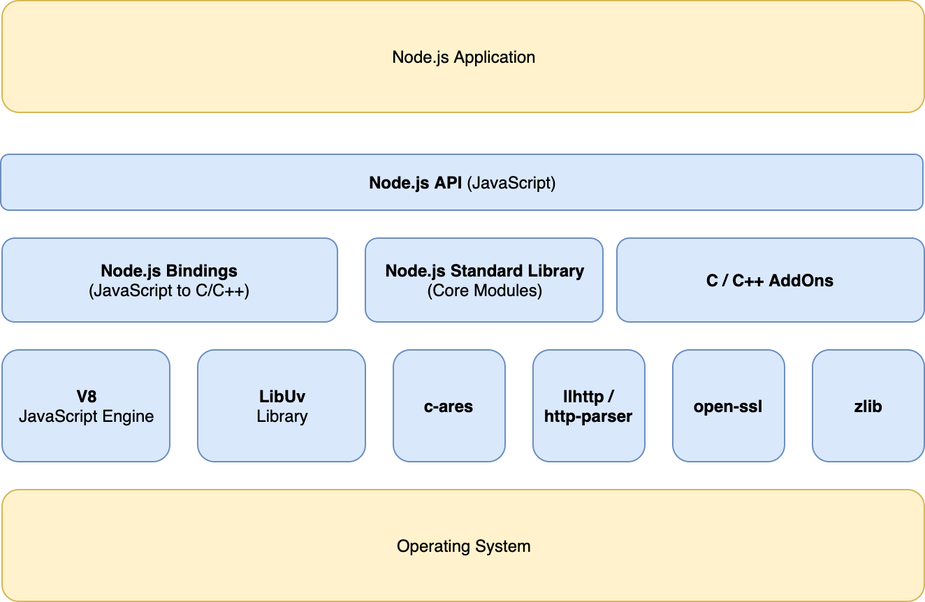
Event-Driven: Node.js sử dụng mô hình lập trình hướng sự kiện, nơi các sự kiện được xử lý bởi một vòng lặp sự kiện (event loop). Điều này giúp tối ưu hóa việc xử lý các tác vụ I/O, làm giảm độ trễ và tăng hiệu suất.

Non-blocking I/O: Các tác vụ I/O (như đọc/ghi file, yêu cầu mạng) được thực hiện không đồng bộ, cho phép các tác vụ khác tiếp tục chạy mà không bị chặn.

**Hoạt động:**

Single Threaded: Mặc dù Node.js chạy trên một luồng đơn, nhưng nhờ vào vòng lặp sự kiện và mô hình non-blocking I/O, nó có thể xử lý nhiều kết nối đồng thời mà không gặp phải vấn đề về hiệu suất.

Call-back Functions: Các hàm gọi lại (callback functions) được sử dụng để xử lý kết quả của các tác vụ không đồng bộ, giúp giảm thiểu thời gian chờ đợi và tối ưu hóa việc sử dụng tài nguyên.



Hình 2.1 Tên hình

### *Các thành phần chính của NodeJS*

V8 Engine: Được phát triển bởi Google, V8 là động cơ JavaScript mạnh mẽ giúp chuyển đổi mã JavaScript thành mã máy (machine code) một cách nhanh chóng.

Libuv: Thư viện đa nền tảng cung cấp các tính năng xử lý I/O không đồng bộ cho Node.js.

Core Modules: Node.js đi kèm với một loạt các module cốt lõi (core modules) như http, fs, path, giúp việc phát triển ứng dụng trở nên dễ dàng hơn.

### *Ưu điểm và nhược điểm của NodeJS*

**Ưu điểm:**

Hiệu suất cao: Nhờ vào V8 Engine và kiến trúc non-blocking I/O.

Mở rộng dễ dàng: Mô hình single-threaded giúp dễ dàng mở rộng các ứng dụng.

Cộng đồng lớn: Có một cộng đồng phát triển mạnh mẽ và nhiều thư viện hỗ trợ qua npm (Node Package Manager).

**Nhược điểm:**

Single-threaded Limitations: Các tác vụ nặng về CPU có thể làm chậm hoặc làm hỏng vòng lặp sự kiện.

Callback Hell: Việc sử dụng quá nhiều hàm gọi lại có thể dẫn đến mã nguồn khó đọc và bảo trì (callback hell).

## Tìm hiểu về ExpressJS

ExpressJS là một framework web nhanh và tối giản cho Node.js, cung cấp một tập hợp các tính năng mạnh mẽ để xây dựng các ứng dụng web và di động. Nó được thiết kế để tạo ra các ứng dụng một cách dễ dàng và hiệu quả, cho phép các nhà phát triển tập trung vào việc viết mã mà không phải lo lắng về các chi tiết phức tạp của HTTP.

### *Lịch sử phát triển*

ExpressJS được tạo ra bởi TJ Holowaychuk vào năm 2010. Nó được xây dựng dựa trên các tiêu chuẩn của Node.js và đã trở thành một trong những framework web phổ biến nhất trong cộng đồng JavaScript. Với một cộng đồng mạnh mẽ và một hệ sinh thái phong phú của các middleware và plugin, ExpressJS đã giúp đơn giản hóa việc phát triển các ứng dụng web phức tạp.

### *Kiến trúc và thành phần*

**Middleware:**

Middleware là các hàm được thực thi trong quá trình xử lý các yêu cầu HTTP. ExpressJS sử dụng middleware để xử lý các tác vụ như xác thực, xử lý dữ liệu đầu vào, và quản lý phiên làm việc (session management).

**Routing:**

Express cung cấp một hệ thống định tuyến mạnh mẽ, cho phép định nghĩa các tuyến đường cho ứng dụng một cách dễ dàng và linh hoạt. Người phát triển có thể tạo ra các tuyến đường cho các phương thức HTTP khác nhau (GET, POST, PUT, DELETE) và gắn kết chúng với các hàm xử lý tương ứng.

**Template Engines:**

Express hỗ trợ nhiều công cụ template khác nhau như Pug, EJS, và Handlebars, giúp việc tạo ra các trang HTML động trở nên dễ dàng hơn.

**Error Handling:**

Express cung cấp cơ chế xử lý lỗi đơn giản nhưng hiệu quả, giúp quản lý và xử lý các lỗi xảy ra trong quá trình xử lý yêu cầu một cách có hệ thống.

### Các tính năng nổi bật

ExpressJS cung cấp nhiều tính năng mạnh mẽ và linh hoạt, bao gồm:

* Định tuyến linh hoạt: ExpressJS cho phép định nghĩa các tuyến đường một cách linh hoạt và dễ dàng, hỗ trợ cả RESTful API và các ứng dụng truyền thống.
* Hệ thống middleware mạnh mẽ: Middleware trong ExpressJS giúp xử lý các yêu cầu và phản hồi một cách hiệu quả, từ việc kiểm tra xác thực, xử lý lỗi, đến việc quản lý cookie và phiên.
* Hỗ trợ view engine: ExpressJS tích hợp tốt với nhiều view engine, giúp tạo ra các giao diện người dùng động một cách dễ dàng.
* Khả năng mở rộng: ExpressJS cho phép mở rộng ứng dụng thông qua các module và plugin, giúp tăng cường chức năng của ứng dụng mà không làm phức tạp mã nguồn.

### Ứng dụng của ExpressJS

ExpressJS được sử dụng rộng rãi trong việc xây dựng các ứng dụng web và API, từ các ứng dụng nhỏ đến các hệ thống lớn. Một số ứng dụng phổ biến của ExpressJS bao gồm:

**Ứng dụng web**: ExpressJS được sử dụng để xây dựng các ứng dụng web với giao diện người dùng phong phú và tương tác.

**API**: ExpressJS là lựa chọn phổ biến để xây dựng các API RESTful, cho phép giao tiếp giữa các dịch vụ và ứng dụng khác nhau.

**Ứng dụng di động**: ExpressJS cũng được sử dụng để xây dựng backend cho các ứng dụng di động, cung cấp dữ liệu và dịch vụ cho các ứng dụng frontend.

## Tìm hiều về ReactJS

ReactJS là một thư viện JavaScript mã nguồn mở được phát triển bởi Facebook, được sử dụng để xây dựng các giao diện người dùng (UI) tương tác và hiệu quả. ReactJS cho phép các nhà phát triển tạo ra các thành phần UI có thể tái sử dụng và quản lý trạng thái của ứng dụng một cách dễ dàng.

### Lịch sử phát triển

ReactJS được phát triển bởi Jordan Walke, một kỹ sư phần mềm tại Facebook, và được phát hành lần đầu vào năm 2013. Ban đầu, ReactJS được sử dụng nội bộ tại Facebook và Instagram trước khi được phát hành ra cộng đồng mã nguồn mở. Từ đó, ReactJS đã trở nên rất phổ biến và được sử dụng rộng rãi trong nhiều dự án web lớn nhỏ.

### Kiến trúc và thành phần

ReactJS dựa trên kiến trúc component-based, nơi các thành phần UI được xây dựng thành các khối độc lập có thể tái sử dụng. Một số thành phần và khái niệm quan trọng trong ReactJS bao gồm:

**Components (Thành phần)**: ReactJS cho phép xây dựng các thành phần UI độc lập, có thể tái sử dụng. Mỗi thành phần có thể chứa logic riêng và giao diện người dùng.

**JSX**: JSX là một cú pháp mở rộng của JavaScript, cho phép viết mã HTML trong JavaScript. JSX giúp dễ dàng mô tả cấu trúc UI của thành phần.

**State và Props**: State là dữ liệu nội bộ của một thành phần, trong khi props là dữ liệu được truyền từ thành phần cha sang thành phần con. Quản lý state và props giúp kiểm soát và cập nhật giao diện người dùng một cách hiệu quả.

**Virtual DOM**: ReactJS sử dụng Virtual DOM để tối ưu hóa việc cập nhật giao diện người dùng. Khi trạng thái của thành phần thay đổi, ReactJS tạo ra một Virtual DOM mới và so sánh với Virtual DOM hiện tại, chỉ cập nhật các phần khác biệt lên DOM thật.

### Các tính năng nổi bậc

ReactJS cung cấp nhiều tính năng mạnh mẽ và linh hoạt, bao gồm:

* Component-based Architecture: Kiến trúc dựa trên thành phần giúp tái sử dụng mã nguồn và dễ dàng quản lý ứng dụng.
* Virtual DOM: Sử dụng Virtual DOM giúp tối ưu hóa hiệu suất và giảm thiểu số lần cập nhật DOM thật.
* Declarative UI: ReactJS cho phép xây dựng UI theo cách khai báo, giúp dễ dàng hiểu và bảo trì mã nguồn.
* One-way Data Flow: Dữ liệu trong ReactJS di chuyển theo một hướng từ cha đến con, giúp dễ dàng theo dõi và kiểm soát luồng dữ liệu.

### Các ứng dụng của ReactJS

ReactJS được sử dụng rộng rãi trong việc phát triển các ứng dụng web và di động. Một số ứng dụng phổ biến của ReactJS bao gồm:

**Single Page Applications (SPA)**: ReactJS là lựa chọn lý tưởng để xây dựng các ứng dụng trang đơn với trải nghiệm người dùng mượt mà và hiệu suất cao.

**Mobile Apps**: Sử dụng React Native, một framework dựa trên ReactJS, để xây dựng các ứng dụng di động cho cả iOS và Android.

**Dynamic Web Applications**: Các ứng dụng web có giao diện người dùng phức tạp và tương tác cao, như các trang thương mại điện tử và mạng xã hội.

## Tìm hiểu chung về MySQL

MySQL là một hệ quản trị cơ sở dữ liệu quan hệ (RDBMS) mã nguồn mở, phổ biến nhất hiện nay. Được phát triển bởi công ty MySQL AB, sau đó được Sun Microsystems mua lại và hiện nay thuộc sở hữu của Oracle Corporation. MySQL được sử dụng rộng rãi trong các ứng dụng web, đặc biệt là các ứng dụng dựa trên nền tảng LAMP (Linux, Apache, MySQL, PHP/Python/Perl).

### Lịch sử phát triển

MySQL được phát triển lần đầu tiên vào năm 1995 bởi Michael Widenius (Monty), David Axmark và Allan Larsson. Ban đầu, nó được thiết kế như một công cụ quản lý cơ sở dữ liệu nhẹ, dễ sử dụng và có hiệu suất cao. Qua nhiều năm, MySQL đã phát triển mạnh mẽ và trở thành một trong những hệ quản trị cơ sở dữ liệu phổ biến nhất trên thế giới.

### Kiến trúc và thành phần

MySQL có kiến trúc client-server, trong đó MySQL server là thành phần chính, chịu trách nhiệm quản lý các cơ sở dữ liệu, xử lý các truy vấn SQL và tương tác với các ứng dụng client.

* MySQL Server: Đây là thành phần chính của MySQL, chịu trách nhiệm quản lý dữ liệu, xử lý các lệnh SQL, và duy trì tính toàn vẹn của dữ liệu.
* MySQL Client: Đây là các công cụ hoặc ứng dụng kết nối với MySQL server để thực hiện các thao tác trên cơ sở dữ liệu.
* Storage Engines: MySQL hỗ trợ nhiều loại storage engines khác nhau như InnoDB, MyISAM, MEMORY, và nhiều loại khác. Mỗi loại storage engine có những đặc điểm và ứng dụng riêng.

### Các tính năng nổi bật

MySQL cung cấp nhiều tính năng nổi bật như:

* Hiệu suất cao: MySQL được tối ưu hóa cho các ứng dụng web và các hệ thống cần xử lý nhiều truy vấn cùng lúc.
* Bảo mật: MySQL cung cấp nhiều cơ chế bảo mật như quản lý người dùng, phân quyền truy cập, và mã hóa dữ liệu.
* Độ tin cậy: MySQL hỗ trợ tính năng backup và recovery, giúp đảm bảo tính toàn vẹn và an toàn của dữ liệu.
* Mã nguồn mở: MySQL là phần mềm mã nguồn mở, cho phép người dùng tự do tải về, sử dụng, và chỉnh sửa mã nguồn.

### Ứng dụng của MySQL

MySQL được sử dụng rộng rãi trong nhiều ứng dụng khác nhau, từ các website nhỏ đến các hệ thống lớn của các công ty công nghệ hàng đầu như Facebook, Twitter, và YouTube. Nó cũng là thành phần chính trong nhiều hệ thống quản lý nội dung (CMS) như WordPress, Joomla, và Drupal.

## Tìm hiểu về VPS

VPS, viết tắt của Virtual Private Server, là một dịch vụ lưu trữ web sử dụng công nghệ ảo hóa để cung cấp các tài nguyên máy chủ riêng biệt trên một máy chủ vật lý. VPS là một lựa chọn phổ biến cho các trang web có lượng truy cập trung bình đến cao, cần nhiều tài nguyên hơn so với dịch vụ lưu trữ chia sẻ nhưng không yêu cầu một máy chủ vật lý riêng.

### Lịch sử phát triển

Dịch vụ VPS xuất hiện như một giải pháp trung gian giữa lưu trữ chia sẻ (shared hosting) và máy chủ riêng (dedicated server). Với sự phát triển của công nghệ ảo hóa, VPS trở nên phổ biến vào những năm 2000, mang lại sự linh hoạt và hiệu suất cao cho người dùng mà không phải chịu chi phí lớn như máy chủ riêng.

### Cách hoạt động của VPS

VPS hoạt động dựa trên công nghệ ảo hóa, chia một máy chủ vật lý thành nhiều máy chủ ảo độc lập. Mỗi VPS có hệ điều hành riêng và có quyền truy cập vào một phần tài nguyên của máy chủ vật lý, bao gồm CPU, RAM, và dung lượng đĩa.

Máy chủ vật lý (Host Machine): Đây là máy chủ thực tế, chứa các tài nguyên vật lý như CPU, RAM, và ổ cứng.

Công nghệ ảo hóa: Phần mềm ảo hóa (như KVM, VMware, hoặc Hyper-V) tạo ra các máy chủ ảo trên máy chủ vật lý.

Máy chủ ảo (VPS): Mỗi VPS hoạt động như một máy chủ riêng biệt, với hệ điều hành và tài nguyên riêng, không bị ảnh hưởng bởi các VPS khác trên cùng máy chủ vật lý.

### Các loại VPS

Có nhiều loại VPS khác nhau, tùy thuộc vào công nghệ ảo hóa và cách thức quản lý:

* Managed VPS: Nhà cung cấp dịch vụ quản lý hoàn toàn VPS, bao gồm bảo mật, cập nhật phần mềm, và hỗ trợ kỹ thuật.
* Unmanaged VPS: Người dùng tự quản lý VPS, từ cài đặt phần mềm đến bảo mật và bảo trì hệ thống.
* Cloud VPS: VPS được triển khai trên một nền tảng đám mây, cho phép mở rộng tài nguyên linh hoạt và tính sẵn sàng cao.

### Lợi ích của việc sử dụng VPS

Sử dụng VPS mang lại nhiều lợi ích so với lưu trữ chia sẻ và máy chủ riêng:

* Hiệu suất cao: VPS cung cấp tài nguyên riêng biệt, giúp đảm bảo hiệu suất ổn định cho trang web hoặc ứng dụng.
* Tự do tùy chỉnh: Người dùng có toàn quyền truy cập root và có thể cài đặt bất kỳ phần mềm hoặc hệ điều hành nào họ muốn.
* Bảo mật tốt hơn: VPS cung cấp môi trường riêng biệt, giảm nguy cơ bị ảnh hưởng bởi các trang web hoặc ứng dụng khác trên cùng máy chủ.
* Khả năng mở rộng: Tài nguyên của VPS có thể được mở rộng dễ dàng mà không cần di chuyển dữ liệu hoặc cấu hình lại hệ thống.

### Các ứng dụng của VPS

VPS được sử dụng rộng rãi trong nhiều trường hợp khác nhau, bao gồm:

* Lưu trữ website: VPS phù hợp cho các trang web có lượng truy cập lớn hoặc yêu cầu tài nguyên cao.
* Lưu trữ ứng dụng: VPS cung cấp môi trường lý tưởng để chạy các ứng dụng web hoặc phần mềm doanh nghiệp.
* Phát triển và thử nghiệm: VPS cung cấp môi trường linh hoạt cho việc phát triển và thử nghiệm các ứng dụng phần mềm.
* Máy chủ game: VPS có thể được sử dụng để chạy các máy chủ game, cung cấp hiệu suất và độ ổn định cao cho người chơi.

## Tìm hiểu về Swagger

Swagger là một bộ công cụ mã nguồn mở giúp thiết kế, xây dựng, và tài liệu hóa các API RESTful. Nó cung cấp các công cụ để tự động tạo ra tài liệu API, giúp cho việc tương tác với API trở nên dễ dàng và nhất quán. Swagger hiện là một phần của OpenAPI Specification (OAS), tiêu chuẩn được chấp nhận rộng rãi trong việc mô tả các API RESTful.

### Lịch sử phát triển

Swagger được tạo ra vào năm 2011 bởi Tony Tam khi làm việc tại Wordnik. Mục đích ban đầu của Swagger là để giúp tự động hóa việc tạo tài liệu cho các API. Với sự phát triển của các API và nhu cầu về việc tài liệu hóa rõ ràng và nhất quán, Swagger nhanh chóng trở nên phổ biến và được tích hợp vào nhiều dự án lớn. Năm 2015, Swagger trở thành một phần của OpenAPI Initiative, một tổ chức dưới sự quản lý của Linux Foundation.

### Cách hoạt động của Swagger

Swagger sử dụng các định nghĩa dựa trên JSON hoặc YAML để mô tả cấu trúc và hành vi của các API RESTful. Các định nghĩa này sau đó có thể được sử dụng để tạo ra tài liệu API động, tạo mã nguồn máy khách (client) và máy chủ (server), và cung cấp giao diện người dùng cho việc thử nghiệm API.

Định nghĩa API: Các định nghĩa API được viết dưới dạng file JSON hoặc YAML, mô tả các endpoint, phương thức HTTP, kiểu dữ liệu, và các thông tin liên quan khác.

Công cụ Swagger: Sử dụng các công cụ như Swagger Editor, Swagger UI, và Swagger Codegen để tạo tài liệu, giao diện thử nghiệm, và mã nguồn từ các định nghĩa API.

Giao diện người dùng: Swagger UI cung cấp một giao diện người dùng trực quan để tương tác và thử nghiệm các API trực tiếp từ trình duyệt.

### Các tính năng nổi bậc

Swagger cung cấp nhiều tính năng mạnh mẽ và hữu ích, bao gồm:

Tài liệu API động: Tự động tạo ra tài liệu API từ các định nghĩa, giúp dễ dàng chia sẻ và duy trì.

Giao diện thử nghiệm: Swagger UI cung cấp một giao diện thử nghiệm tương tác, cho phép người dùng thử nghiệm các endpoint API trực tiếp từ trình duyệt.

Tạo mã nguồn: Swagger Codegen tạo mã nguồn cho các ứng dụng máy khách và máy chủ từ các định nghĩa API, giúp giảm thời gian phát triển.

Tích hợp dễ dàng: Swagger có thể được tích hợp với nhiều ngôn ngữ lập trình và framework khác nhau, bao gồm Node.js, Java, Python, và nhiều ngôn ngữ khác.

### Ứng dụng của Swagger

Swagger được sử dụng rộng rãi trong việc phát triển và tài liệu hóa các API RESTful. Một số ứng dụng phổ biến của Swagger bao gồm:

Phát triển API: Tạo và duy trì tài liệu API nhất quán, giúp giảm thiểu lỗi và tăng cường sự hiểu biết giữa các thành viên trong nhóm phát triển.

Thử nghiệm API: Sử dụng Swagger UI để thử nghiệm các endpoint API một cách dễ dàng và hiệu quả.

Tạo mã nguồn tự động: Sử dụng Swagger Codegen để tự động tạo mã nguồn cho các ứng dụng máy khách và máy chủ, giúp tiết kiệm thời gian và công sức trong quá trình phát triển.

## Tìm hiểu về RESTful API

RESTful API là một kiến trúc phần mềm dành cho các hệ thống mạng. Được xây dựng trên giao thức HTTP, RESTful API cho phép các ứng dụng tương tác với nhau qua Internet thông qua các phương thức chuẩn như GET, POST, PUT, DELETE. Các API này tuân theo các nguyên tắc REST, mang lại sự đơn giản và khả năng mở rộng cao.

### Các nguyên tắc của RESTful API

Giao diện thống nhất (Uniform Interface): Giao diện đồng nhất là yếu tố cốt lõi của REST. Điều này đảm bảo rằng các tương tác giữa client và server diễn ra thông qua một giao diện duy nhất.

Stateless: Mỗi yêu cầu từ client đến server phải chứa tất cả thông tin cần thiết để server hiểu và xử lý yêu cầu. Server không lưu trữ trạng thái client giữa các yêu cầu.

Khả năng lưu trữ (Cacheable): Các phản hồi từ server có thể được đánh dấu là có thể lưu trữ hoặc không. Nếu một phản hồi có thể lưu trữ, client có thể lưu trữ nó để sử dụng lại trong tương lai.

Hệ thống phân tầng (Layered System): Kiến trúc REST có thể được tổ chức thành các lớp, mỗi lớp có trách nhiệm riêng, và client không biết rằng nó đang giao tiếp với server trực tiếp hay thông qua một lớp trung gian.

### Các thành phần chính của RESTful API

Resources: Mọi thứ trong REST đều là resources. Một resource có thể là một đối tượng dữ liệu, một tập hợp các đối tượng, hoặc bất kỳ thứ gì có thể được đại diện bằng một URL.

URL: Mỗi resource trong REST được định danh bằng một URL duy nhất. URL đại diện cho đường dẫn tới resource trên server.

HTTP Methods: RESTful API sử dụng các phương thức HTTP để thực hiện các thao tác khác nhau trên resource. Các phương thức chính bao gồm:

GET: Lấy dữ liệu từ server.

POST: Gửi dữ liệu mới tới server để tạo resource mới.

PUT: Cập nhật dữ liệu của một resource hiện có.

DELETE: Xóa một resource.

### Các bước xây dựng RESTful API

Xác định resources: Bắt đầu bằng việc xác định các resources mà API sẽ quản lý. Mỗi resource nên đại diện cho một thực thể cụ thể trong hệ thống.

Định nghĩa URL cho mỗi resource: Tạo URL duy nhất cho mỗi resource. URL nên có cấu trúc rõ ràng và dễ hiểu.

Xác định HTTP methods cho từng thao tác: Gán các phương thức HTTP phù hợp cho từng thao tác trên resources. Ví dụ, sử dụng GET để lấy thông tin resource, POST để tạo resource mới, v.v.

Thiết kế phản hồi của server: Xác định định dạng phản hồi của server (ví dụ: JSON, XML) và đảm bảo rằng phản hồi chứa tất cả thông tin cần thiết.

Bảo mật API: Áp dụng các biện pháp bảo mật như xác thực, phân quyền, và mã hóa để bảo vệ API khỏi các mối đe dọa tiềm ẩn.

# HIỆN THỰC HÓA NGHIÊN CỨU

## Mô tả bài toán

Xây dựng hệ thống trực quan hóa kết quả học tập sinh viên CNTT bao gồm nhiều bước quan trọng để thu thập, xử lý và hiển thị dữ liệu học tập. Quá trình này bắt đầu bằng việc nhập thông tin kết quả học tập của sinh viên, bao gồm các chỉ số như điểm số, bài tập, dự án và các kỳ thi. Thông tin này cần được cập nhật chi tiết và đầy đủ, bao gồm: mã sinh viên, tên sinh viên, môn học, học kỳ, điểm số, và mô tả chi tiết về kết quả học tập.

Hệ thống cũng bao gồm việc thu thập và xử lý dữ liệu để tạo ra các biểu đồ và đồ thị trực quan, giúp dễ dàng theo dõi tiến độ học tập và hiệu suất của sinh viên theo thời gian. Các biểu đồ này có thể là biểu đồ đường, biểu đồ cột, hoặc biểu đồ tròn, tùy thuộc vào loại dữ liệu và mục đích hiển thị. Ngoài ra, hệ thống còn cung cấp các báo cáo chi tiết, cho phép giảng viên có cái nhìn tổng quan về kết quả học tập và phát hiện sớm các vấn đề để có biện pháp khắc phục kịp thời.

Quá trình nhập liệu và xử lý dữ liệu này là rất quan trọng để đảm bảo rằng hệ thống trực quan hóa hoạt động chính xác và hiệu quả, mang lại những thông tin hữu ích cho người dùng.

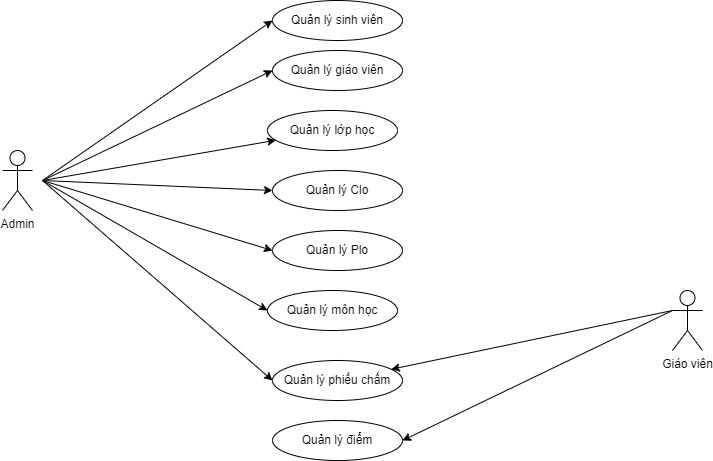


## Yêu cầu chức năng

Để xây dựng hệ thống trực quan hóa kết quả học tập sinh viên CNTT, cần một số yêu cầu chức năng cụ thể để đáp ứng nhu cầu người dùng:

* Hiển thị kết quả học tập: Hiển thị toàn bộ kết quả học tập của sinh viên theo từng môn học, học kỳ.
* Lọc kết quả học tập theo tiêu chí: Cho phép người dùng lọc kết quả học tập theo môn học, học kỳ, hoặc điểm số.
* Tìm kiếm sinh viên theo tên: Người dùng có thể tìm kiếm thông tin kết quả học tập của sinh viên dựa trên tên hoặc mã sinh viên.
* Tìm kiếm kết quả theo môn học: Người dùng có thể tìm kiếm kết quả học tập theo tên môn học hoặc mã môn học.
* Xem thông tin chi tiết: Cung cấp thông tin chi tiết về kết quả học tập của từng sinh viên, bao gồm: mã sinh viên, tên sinh viên, môn học, học kỳ, điểm số, và mô tả chi tiết về kết quả.
* Biểu đồ và đồ thị trực quan: Hiển thị biểu đồ và đồ thị trực quan về tiến độ và hiệu suất học tập của sinh viên, bao gồm các loại biểu đồ như biểu đồ đường, biểu đồ cột, biểu đồ tròn, và biểu đồ radar.
* Báo cáo tổng hợp: Cung cấp các báo cáo tổng hợp về kết quả học tập của sinh viên theo từng lớp học, học kỳ, năm học.
* Chức năng xuất dữ liệu: Cho phép xuất dữ liệu kết quả học tập dưới dạng các file như Excel, PDF để tiện lưu trữ và chia sẻ.

## Đặt tả hệ thống



Hình 3.1 Sơ đồ use case

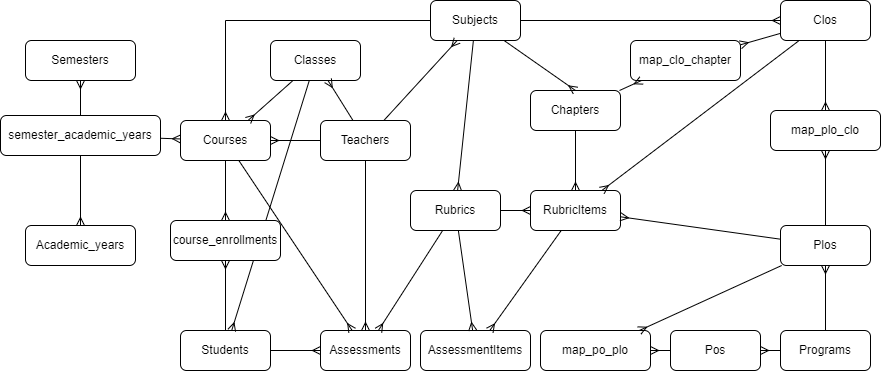
Bảng 3.1. Mô tả Actor

|  |  |  |
| --- | --- | --- |
| **STT** | **Tên Actor** | **Ý nghĩa** |
| 1 | Giáo viên | Người dùng hệ thống |
| 2 | Admin | Người quản trị hệ thống |

Bảng 3.2 Mô tả use case

|  |  |  |
| --- | --- | --- |
| **STT** | **Tên Use-case** | **Ý Nghĩa** |
| 1 | Quản lý sinh viên | Quản trị viên có thể quản lý thông tin của các sinh viên thêm, sửa, xoá nếu cần thiết. |
| 2 | Quản lý giáo viên | Quản trị viên có thể quản lý thông tin của giáo trên trang web và thêm, sửa, xoá nếu cần thiết. |
| 3 | Quản lý lớp học | Quản trị viên có thể quản lý thông tin của lớp web và thêm, sửa, xoá nếu cần thiết. |
| 4 | Quản lý Clo | Quản trị viên có thể quản lý chuẩn đầu ra học phần và thêm, sửa, xoá nếu cần thiết. |
| 5 | Quản lý Plo | Quản trị viên có thể quản lý thông tin chuẩn đầu ra đào tạo và thêm, sửa, xoá nếu cần thiết. |
| 6 | Quản lý môn học | Quản trị viên có thể quản lý thông tin của môn học trên trang web và thêm, sửa, xoá nếu cần thiết. |
| 7 | Quản lý phiếu chấm | Quản trị viên hoặc giáo viên cáo thể tạo phiếu chấm. |
| 8 | Quản lý điểm | Giáo viên có thể tìm kiếm, lọc các tiêu chí trên hệ thống để xem kết quả học tập của sinh viên |

## Lược đồ dữ liệu



Hình 3.2. Lược đồ cơ sở dữ liệu

**Mô tả các bảng:**

Bảng 3.3. Mô tả bảng “Student”

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| STT | Tên thuộc tính | Loại | Ràng buộc | Ý nghĩa/ghi chú |
| 1 | student\_id | int | khóa chính | mã khóa sinh viên |
| 2 | teacher\_id | int | khóa ngoại | mã khóa giáo viên |
| 3 | studentCode | varchar(9) | duy nhất | mã sinh viên |
| 4 | email | varchar(255) |  | email của sinh viên |
| 5 | name | varchar(255) |  | tên sinh viên |
| 6 | date\_of\_birth | date |  | ngày sinh |
| 7 | isDelete | tinyint |  | ẩn sinh viên |
| 8 | createdAt | timestamp |  | ngày tạo |
| 9 | updatedAt | timestamp |  | ngày cập nhật |

Bảng 3.4. Mô tả bảng “Teacher”

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| STT | Tên thuộc tính | Loại | Ràng buộc | Ý nghĩa/ghi chú |
| 1 | teacher\_id | int | khóa ngoại | mã khóa giáo viên |
| 2 | name | varchar(255) |  | tên giáo viên |
| 3 | teacherCode | varchar(20) | duy nhất | mã giáo viên |
| 4 | email | varchar(255) | duy nhất | email của giáo viên |
| 5 | password | varchar(255) |  | mật khẩu |
| 6 | permission | tinyint |  | phân quyền |
| 7 | imgURL | text |  | link ảnh hình đại diện |
| 8 | isBlock | tityint |  | chặn giáo viên |
| 9 | typeTeacher | enum('GVCV', 'GVGD') |  | loại giáo viên |
| 10 | isDelete | tinyint |  | ẩn giáo viên |
| 11 | createdAt | timestamp |  | ngày tạo |
| 12 | updatedAt | timestamp |  | ngày cập nhật |

Bảng 3.5. Môn tả bảng “Subjects”

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| STT | Tên thuộc tính | Loại | Ràng buộc | Ý nghĩa/ghi chú |
| 1 | subject\_id | int | khóa chính | mã khóa môn học |
| 2 | teacher\_id | int | khóa ngoại | mã khóa giáo viên |
| 3 | subjectName | text |  | tên môn học |
| 4 | subjectCode | varchar(6) | duy nhất | mã môn học |
| 5 | description | text |  | mô tả khóa học |
| 6 | munberCredits | int |  | số tính chỉ |
| 7 | numberCreditsTheory | int |  | số tính chỉ lý thuyết |
| 8 | numberCreditsPractice | int |  | số tính chỉ thực hành |
| 9 | typesubject | enum('Đại cương','Cơ sở ngành','Chuyên ngành','Thực tập và Đồ án') |  | loại môn học |
| 10 | isDelete | tinyint |  | ẩn môn học |
| 11 | createdAt | timestamp |  | ngày tạo |
| 12 | updatedAt | timestamp |  | ngày cập nhật |

Bảng 3.6. Mô tả bảng “Courses”

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| STT | Tên thuộc tính | Loại | Ràng buộc | Ý nghĩa/ghi chú |
| 1 | course\_id | Int | khóa chính | mã khóa khóa học |
| 2 | teacher\_id | int | khóa ngoại | mã khóa giáo viên |
| 3 | class\_id | int | khóa ngoại | mã khóa lớp |
| 4 | subject\_id | int | khóa ngoại | mã khóa môn học |
| 5 | id\_semester\_academic\_year | int | khóa ngoại | mã khóa học kì năm học |
| 6 | courseName | text |  | tên khóa học |
| 7 | courseCode | varchar(20) | duy nhất | mã khoán học |
| 8 | isDelete | tinyint |  | ẩn khóa học |
| 9 | createdAt | timestamp |  | ngày tạo |
| 10 | updatedAt | timestamp |  | ngày cập nhật |

Bảng 3.7. Mô tả bảng “academic\_year”

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| STT | Tên thuộc tính | Loại | Ràng buộc | Ý nghĩa/ghi chú |
| 1 | course\_id | int | khóa chính | mã khóa khóa học |
| 2 | startDate | timestamp |  | năm bắt đầu |
| 3 | endDate | timestamp |  | năm kết thúc |
| 4 | description | text |  | mô tả năm học |
| 5 | isDelete | tinyint |  | ẩn năm học |
| 6 | createdAt | timestamp |  | ngày tạo |
| 7 | updatedAt | timestamp |  | ngày cập nhật |

Bảng 3.8. Mô tả bảng “Semesters”

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| STT | Tên thuộc tính | Loại | Ràng buộc | Ý nghĩa/ghi chú |
| 1 | semester\_id | int | khóa chính | mã khóa học kì |
| 2 | descriptionShort | text |  | mô tả ngắn |
| 3 | descriptionLong | text |  | mô tả dài |
| 4 | codeSemester | varchar(20) |  | mã năm học |
| 5 | isDelete | tinyint |  | ẩn năm học |
| 6 | createdAt | timestamp |  | ngày tạo |
| 7 | updatedAt | timestamp |  | ngày cập nhật |

Bảng 3.9. Mô tả bảng “Semester\_academic\_years”

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| STT | Tên thuộc tính | Loại | Ràng buộc | Ý nghĩa/ghi chú |
| 1 | id\_semester\_academic\_year | int | khóa chính | mã khóa năm học học kì |
| 2 | academic\_year\_id | int |  | mã khóa năm học |
| 3 | semester\_id | int |  | mã khóa học kỳ |
| 4 | isDelete | tinyint |  | ẩn năm học học kỳ |
| 5 | createdAt | timestamp |  | ngày tạo |
| 6 | updatedAt | timestamp |  | ngày cập nhật |

Bảng 3.10. Mô tả bảng “Accessments”

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| STT | Tên thuộc tính | Loại | Ràng buộc | Ý nghĩa/ghi chú |
| 1 | accessment\_id | int | khóa chính | mã khóa lần đánh giá |
| 2 | teacher\_id | int | khóa ngoại | mã khóa giáo viên |
| 3 | student\_id | int | khóa ngoại | mã khóa lớp |
| 4 | rubric\_id | int | khóa ngoại | mã khóa phiếu chấm |
| 4 | course\_id | int | khóa ngoại | mã khóa khóa học |
| 5 | description | text |  | mô tả |
| 6 | totalScore | double(8,2) |  | điểm của lần chấm |
| 7 | date | date |  | ngày chấm |
| 8 | place | text |  | địa chỉ |
| 9 | isDelete | tinyint |  | ẩn khóa học |
| 10 | createdAt | timestamp |  | ngày tạo |
| 11 | updatedAt | timestamp |  | ngày cập nhật |

Bảng 3.11. Mô tả bảng “assessmentItems”

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| STT | Tên thuộc tính | Loại | Ràng buộc | Ý nghĩa/ghi chú |
| 1 | assessmentItem\_id | int | khóa chính | mã khóa mục đánh giá |
| 2 | assessment\_id | int | khóa ngoại | mã khóa lần đánh giá |
| 3 | rubricsItem\_id | int | khóa ngoại | mã khóa mục phiếu chấm |
| 4 | assessmentScore | double(8,2) |  | điểm mục đánh giá |
| 5 | isDelete | tinyint |  | ẩn mục đánh giá |
| 6 | createdAt | timestamp |  | ngày tạo |
| 7 | updatedAt | timestamp |  | ngày cập nhật |

Bảng 3.12. Mô tả bảng “Chapters”

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| STT | Tên thuộc tính | Loại | Ràng buộc | Ý nghĩa/ghi chú |
| 1 | chapter\_id | int | khóa chính | mã khóa chương |
| 2 | subject\_id | int | khóa ngoại | mã khóa môn học |
| 3 | chapterName | varchar(100) |  | tên chương học |
| 4 | description | text |  | Mô tả chương |
| 5 | isDelete | tinyint |  | ẩn chương |
| 6 | createdAt | timestamp |  | ngày tạo |
| 7 | updatedAt | timestamp |  | ngày cập nhật |

Bảng 3.13. Mô tả bảng “Classes”

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| STT | Tên thuộc tính | Loại | Ràng buộc | Ý nghĩa/ghi chú |
| 1 | class\_id | int | khóa chính | mã khóa lớp |
| 2 | teacher\_id | int | khóa ngoại | mã khóa giáo viên |
| 3 | className | varchar(255) |  | tên lớp |
| 4 | classNameShort | varchar(20) | duy nhất | tên lớp ngắn |
| 5 | classCode | varchar(10) | duy nhất | mã lớp |
| 6 | isDelete | tinyint |  | ẩn lớp |
| 7 | createdAt | timestamp |  | ngày tạo |
| 8 | updatedAt | timestamp |  | ngày cập nhật |

Bảng 3.14. Mô tả bảng “Clos”

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| STT | Tên thuộc tính | Loại | Ràng buộc | Ý nghĩa/ghi chú |
| 1 | clo\_id | int | khóa chính | mã khóa CĐR môn học |
| 2 | subject\_id | int | khóa ngoại | mã khóa môn học |
| 3 | cloName | varchar(20) |  | tên CĐR môn học |
| 4 | description | text |  | Mô tả CĐR môn học |
| 6 | isDelete | tinyint |  | ẩn CĐR môn học |
| 7 | createdAt | timestamp |  | ngày tạo |
| 8 | updatedAt | timestamp |  | ngày cập nhật |

Bảng 3.15. Mô tả bảng “Plos”

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| STT | Tên thuộc tính | Loại | Ràng buộc | Ý nghĩa/ghi chú |
| 1 | plo\_id | int | khóa chính | mã khóa CĐR chương trình |
| 2 | program\_id | int | khóa ngoại | mã khóa chương trình |
| 3 | ploName | varchar(20) |  | tên CĐR chương trình |
| 4 | description | text |  | Mô tả CĐR chương trình |
| 6 | isDelete | tinyint |  | ẩn CĐR chương trình |
| 7 | createdAt | timestamp |  | ngày tạo |
| 8 | updatedAt | timestamp |  | ngày cập nhật |

Bảng 3.16. Mô tả bảng “Pos”

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| STT | Tên thuộc tính | Loại | Ràng buộc | Ý nghĩa/ghi chú |
| 1 | po\_id | int | khóa chính | mã khóa mục tiêu trương trình đào tạo |
| 2 | program\_id | int | khóa ngoại | mã khóa chương trình |
| 3 | poName | varchar(255) |  | tên mục tiêu chương trình đào tạo |
| 4 | description | text |  | Mô tả |
| 6 | isDelete | tinyint |  | ẩn mục tiêu chương trình đào tạo |
| 7 | createdAt | timestamp |  | ngày tạo |
| 8 | updatedAt | timestamp |  | ngày cập nhật |

Bảng 3.17. Mô tả bảng “course\_enrollment”

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| STT | Tên thuộc tính | Loại | Ràng buộc | Ý nghĩa/ghi chú |
| 1 | id\_detail\_courses | int | khóa chính | mã khóa course\_enrollment |
| 2 | student\_id | int | khóa ngoại | mã khóa sinh viên |
| 3 | course\_id | int | khóa ngoại | mã khóa khóa học |
| 6 | isDelete | tinyint |  | ẩn course\_enrollment |
| 7 | createdAt | timestamp |  | ngày tạo |
| 8 | updatedAt | timestamp |  | ngày cập nhật |

Bảng 3.18. Mô tả bảng “Programs”

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| STT | Tên thuộc tính | Loại | Ràng buộc | Ý nghĩa/ghi chú |
| 1 | program\_id | int | khóa chính | mã khóa chương trình |
| 2 | programName | text |  | tên chương trình |
| 3 | description | text |  | mô tả chương trình |
| 4 | isDelete | tinyint |  | ẩn chương trình |
| 5 | createdAt | timestamp |  | ngày tạo |
| 6 | updatedAt | timestamp |  | ngày cập nhật |

Bảng 3.19. Mô tả bảng “Rubrics”

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| STT | Tên thuộc tính | Loại | Ràng buộc | Ý nghĩa/ghi chú |
| 1 | rubric\_id | int | khóa chính | mã khóa phiếu chấm |
| 2 | subject\_id | int | khóa ngoại | mã khóa môn học |
| 3 | teacher\_id | int | khóa ngoại | mã khóa giáo viên |
| 4 | rubricName | varchar(255) |  | tên phiếu chấm |
| 5 | comment | text |  | ghi chú |
| 6 | isDelete | tinyint |  | ẩn phiếu chấm |
| 7 | createdAt | timestamp |  | ngày tạo |
| 8 | updatedAt | timestamp |  | ngày cập nhật |

Bảng 3.20. Mô tả bảng “RubricItems”

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| STT | Tên thuộc tính | Loại | Ràng buộc | Ý nghĩa/ghi chú |
| 1 | rubricsItem\_id | int | khóa chính | mã khóa mục phiếu chấm |
| 2 | chapter\_id | int | khóa ngoại | mã khóa sinh viên |
| 3 | clo\_id | int | khóa ngoại | mã khóa khóa học |
| 4 | rubric\_id | int | khóa ngoại | mã khóa phiếu chấm |
| 5 | plo\_id | int | khóa ngoại | mã khóa CĐR chương trình |
| 6 | description | text |  | mô tả |
| 7 | maxScore | double(8,2) |  | điểm mục phiếu chấm |
| 8 | isDelete | tinyint |  | ẩn mục phiếu chấm |
| 9 | createdAt | timestamp |  | ngày tạo |
| 10 | updatedAt | timestamp |  | ngày cập nhật |

Bảng 3.21. Mô tả bảng “map\_ clo\_chapters”

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| STT | Tên thuộc tính | Loại | Ràng buộc | Ý nghĩa/ghi chú |
| 1 | id\_clo\_chapter | int | khóa chính | mã khóa map\_clo\_chapter |
| 2 | clo\_id | int | khóa ngoại | mã khóa CĐR môn học |
| 3 | chapter\_id | int | khóa ngoại | mã khóa chương |
| 6 | isDelete | tinyint |  | ẩn |
| 7 | createdAt | timestamp |  | ngày tạo |
| 8 | updatedAt | timestamp |  | ngày cập nhật |

Bảng 3.22. Mô tả bảng “map\_plo\_clo”

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| STT | Tên thuộc tính | Loại | Ràng buộc | Ý nghĩa/ghi chú |
| 1 | id\_plo\_clo | int | khóa chính | mã khóa map\_plo\_clo |
| 2 | clo\_id | int | khóa ngoại | mã khóa CĐR môn học |
| 3 | plo\_id | int | khóa ngoại | mã khóa CĐR chương trình |
| 6 | isDelete | tinyint |  | ẩn |
| 7 | createdAt | timestamp |  | ngày tạo |
| 8 | updatedAt | timestamp |  | ngày cập nhật |

Bảng 3.23. Mô tả bảng “map\_po\_plo”

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| STT | Tên thuộc tính | Loại | Ràng buộc | Ý nghĩa/ghi chú |
| 1 | id\_po\_plo | int | khóa chính | mã khóa map\_po\_plo |
| 2 | clo\_id | int | khóa ngoại | mã khóa CĐR môn học |
| 3 | plo\_id | int | khóa ngoại | mã khóa CĐR chương trình |
| 6 | isDelete | tinyint |  | ẩn |
| 7 | createdAt | timestamp |  | ngày tạo |
| 8 | updatedAt | timestamp |  | ngày cập nhật |

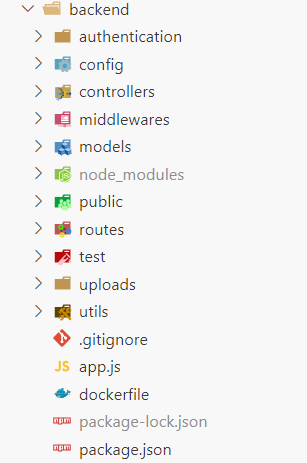
Bảng 3.24. Mô tả bảng “refresh\_tokens”

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| STT | Tên thuộc tính | Loại | Ràng buộc | Ý nghĩa/ghi chú |
| 1 | id | int | khóa chính | mã khóa |
| 2 | teacher\_id | int | khóa ngoại | mã khóa giáo viên |
| 2 | token | varchar(255) |  | refresh token |
| 3 | expired | tinyint |  | có hết hạn |
| 6 | revoked | tinyint |  | có thu hồi |
| 7 | createdAt | timestamp |  | ngày tạo |
| 8 | updatedAt | timestamp |  | ngày cập nhật |

## Kiến trúc hệ thống

### Kiến trúc hệ thống backend

Kiến trúc hệ thống backend là một phần quan trọng trong việc phát triển các ứng dụng web và dịch vụ trực tuyến. Đối với hệ thống trực quan hóa kết quả học tập sinh viên CNTT sử dụng Express.js, kiến trúc backend được thiết kế để đảm bảo tính mở rộng, bảo mật, hiệu suất và dễ bảo trì.



Hình 3.3. Kiến trúc hệ thống backend

Dưới đây là một phân tích chi tiết về các thành phần chính của hệ thống backend:

**Layered architecture**

Hệ thống backend được thiết kế theo kiến trúc lớp (layered architecture), với mỗi lớp có một vai trò cụ thể và rõ ràng:

* Presentation Layer: Xử lý các yêu cầu HTTP từ client, định tuyến các yêu cầu đến các controller tương ứng và trả về phản hồi. Lớp này bao gồm các routes và middleware.
* Business Logic Layer: Chứa các logic nghiệp vụ của ứng dụng. Lớp này được hiện thực bởi các controller.
* Data Access Layer: Tương tác với cơ sở dữ liệu để truy xuất và lưu trữ dữ liệu. Lớp này được hiện thực bởi các models và các thao tác cơ sở dữ liệu.
* Utility Layer: Chứa các hàm và thư viện tiện ích dùng chung trong toàn bộ ứng dụng.

**Thư mục chính và vai trò của chúng**

Dựa trên cấu trúc thư mục đã trình bày, dưới đây là mô tả chi tiết về từng thư mục và vai trò của chúng:

* authentication: Xử lý xác thực và phân quyền người dùng. Bao gồm các middleware như authMiddleware.js để bảo vệ các route.
* config: Chứa các file cấu hình hệ thống như kết nối cơ sở dữ liệu, cấu hình môi trường (environment variables).
* controllers: Chứa các controller để xử lý logic nghiệp vụ cho các yêu cầu HTTP.
* middlewares: Chứa các middleware để xử lý các yêu cầu HTTP trước khi chúng đến controller.
* models: Chứa các mô hình dữ liệu (models) sử dụng ORM (Object-Relational Mapping) để tương tác với cơ sở dữ liệu.
* routes: Định nghĩa các route cho ứng dụng, mỗi route có thể liên kết với một controller cụ thể.
* utils: Chứa các hàm tiện ích dùng chung.
* app.js: File chính của ứng dụng, nơi cấu hình Express và kết nối các thành phần khác. Chứa các middleware chung, kết nối cơ sở dữ liệu, và khởi động server.

**Quy trình xử lý yêu cầu HTTP**

Dưới đây là mô tả quy trình xử lý một yêu cầu HTTP từ khi nhận được đến khi phản hồi:

* Nhận yêu cầu: Khi một yêu cầu HTTP đến từ client, nó sẽ được tiếp nhận bởi app.js.
* Xử lý middleware: Yêu cầu sẽ đi qua các middleware toàn cục (như bodyParser, cors, authMiddleware) để xử lý các chức năng như phân tích body, xử lý CORS, và xác thực người dùng.
* Định tuyến: Sau khi qua các middleware, yêu cầu sẽ được định tuyến đến route tương ứng dựa trên URL và phương thức HTTP. Ví dụ: /api/users/:id sẽ được xử lý bởi userRoutes.js.
* Xử lý trong controller: Route sẽ gọi hàm tương ứng trong controller, ví dụ getUser trong userController.js, để thực hiện logic nghiệp vụ.
* Truy xuất dữ liệu: Controller có thể tương tác với model để truy xuất hoặc cập nhật dữ liệu trong cơ sở dữ liệu.
* Trả về phản hồi: Sau khi xử lý xong, controller sẽ gửi phản hồi về cho client qua HTTP response.

**Bảo mật và Quản lý lỗi**

* Bảo mật: Hệ thống sử dụng các middleware để xác thực và phân quyền người dùng, bảo vệ các tài nguyên nhạy cảm và đảm bảo rằng chỉ những người dùng có quyền mới có thể truy cập.
* Quản lý lỗi: Sử dụng middleware để quản lý lỗi toàn cục, ghi log lỗi và trả về phản hồi lỗi có ý nghĩa cho client.

### Kiến trúc hệ thống Frontend

Cấu trúc thư mục frontend của một ứng dụng React

A screenshot of a computer

Description automatically generated

Hình 3.4. Kiến trúc hệ thống Frontend

**Thư mục chính và vai trò của chúng**

node\_modules: Chứa các gói npm được cài đặt. Thư mục này được tạo tự động khi bạn chạy lệnh npm install hoặc yarn install.

public: Chứa các file tĩnh như hình ảnh, biểu tượng, và file index.html là điểm vào của ứng dụng.

src: Chứa mã nguồn chính của ứng dụng React.

**Chi tiết các thư mục con trong src**

components: Chứa các thành phần (components) của ứng dụng. Mỗi component thường đại diện cho một phần tử giao diện (UI) độc lập và có thể tái sử dụng.

Helper: Chứa các hàm tiện ích (utility functions) và các module hỗ trợ khác mà các component có thể sử dụng.

layouts: Chứa các layout components, đại diện cho bố cục tổng thể của trang hoặc các phần lớn của trang.

service: Chứa các service để làm việc với API hoặc các dịch vụ bên ngoài khác.

**Các file quan trọng trong src**

App.css: Chứa các style toàn cục cho ứng dụng.

App.js: Thành phần chính của ứng dụng, là nơi cấu hình các routes và điều hướng.

App.test.js: Chứa các test case cho thành phần App.js.

index.css: Chứa các style toàn cục khác có thể áp dụng cho ứng dụng.

index.js: Điểm vào của ứng dụng React. File này render component App vào DOM.

logo.svg: Logo của ứng dụng.

reportWebVitals.js: Đo lường hiệu suất của ứng dụng.

setupTests.js: Cấu hình cho các test case.

**Các file cấu hình và môi trường**

.env: Chứa các biến môi trường.

.gitignore: Liệt kê các file và thư mục sẽ bị Git bỏ qua.

Dockerfile: File cấu hình Docker để tạo container cho ứng dụng.

package-lock.json và package.json: Chứa các thông tin về các gói npm và các script của ứng dụng.

README.md: File hướng dẫn và tài liệu của dự án.

## Xây dựng hệ thống Backend

### Kết nối cơ sở dữ liệu

Quá trình kết nối dữ liệu trong Node.js với Sequelize và dotenv bao gồm các bước sau đây:

**Import các thư viện cần thiết**

A close-up of a computer code

Description automatically generated

Hình 3.5. Import các thư viện cần thiết kết nối cơ sở dữ liệu

Sequelize: Được import từ gói sequelize để sử dụng các tính năng ORM.

dotenv: Được import để quản lý các biến môi trường, giúp bảo mật thông tin nhạy cảm như thông tin đăng nhập cơ sở dữ liệu.

**Tạo một instance của Sequelize với các biến môi trường**

A computer screen shot of a computer code

Description automatically generated

Hình 3.6. Tạo instance của Sequelize

Sequelize: Khởi tạo một instance của Sequelize với các tham số như tên cơ sở dữ liệu, tên người dùng, mật khẩu và các thông tin khác được lấy từ các biến môi trường.

process.env.DB\_NAME: Tên cơ sở dữ liệu.

process.env.DB\_USER: Tên người dùng cơ sở dữ liệu.

process.env.DB\_PASSWORD: Mật khẩu cơ sở dữ liệu.

process.env.DB\_HOST: Địa chỉ host của cơ sở dữ liệu.

process.env.DB\_DIALECT: Loại cơ sở dữ liệu (MySQL, PostgreSQL, SQLite, v.v.).

**Kiểm tra kết nối tới cơ sở dữ liệu**

**A computer code with green and blue text

Description automatically generated**

Hình 3.7. Kiểm tra kết nối với dữ liệu

testConnection: Hàm bất đồng bộ để kiểm tra kết nối tới cơ sở dữ liệu.

sequelize.authenticate(): Phương thức này kiểm tra xem kết nối tới cơ sở dữ liệu có thành công hay không.

console.log: In ra thông báo thành công nếu kết nối được thiết lập.

console.error: In ra lỗi nếu không thể kết nối tới cơ sở dữ liệu.

### Định Nghĩa Mô Hình với Sequelize

Đoạn mã dưới đây là mô tả cách định nghĩa một mô hình sinh viên (StudentModel) sử dụng Sequelize ORM. Mô hình này bao gồm các thông tin chi tiết về sinh viên và mối quan hệ với một mô hình lớp học (ClassModel). Dưới đây là phân tích chi tiết từng phần của đoạn mã:

**Import các thư viện cần thiết**

****

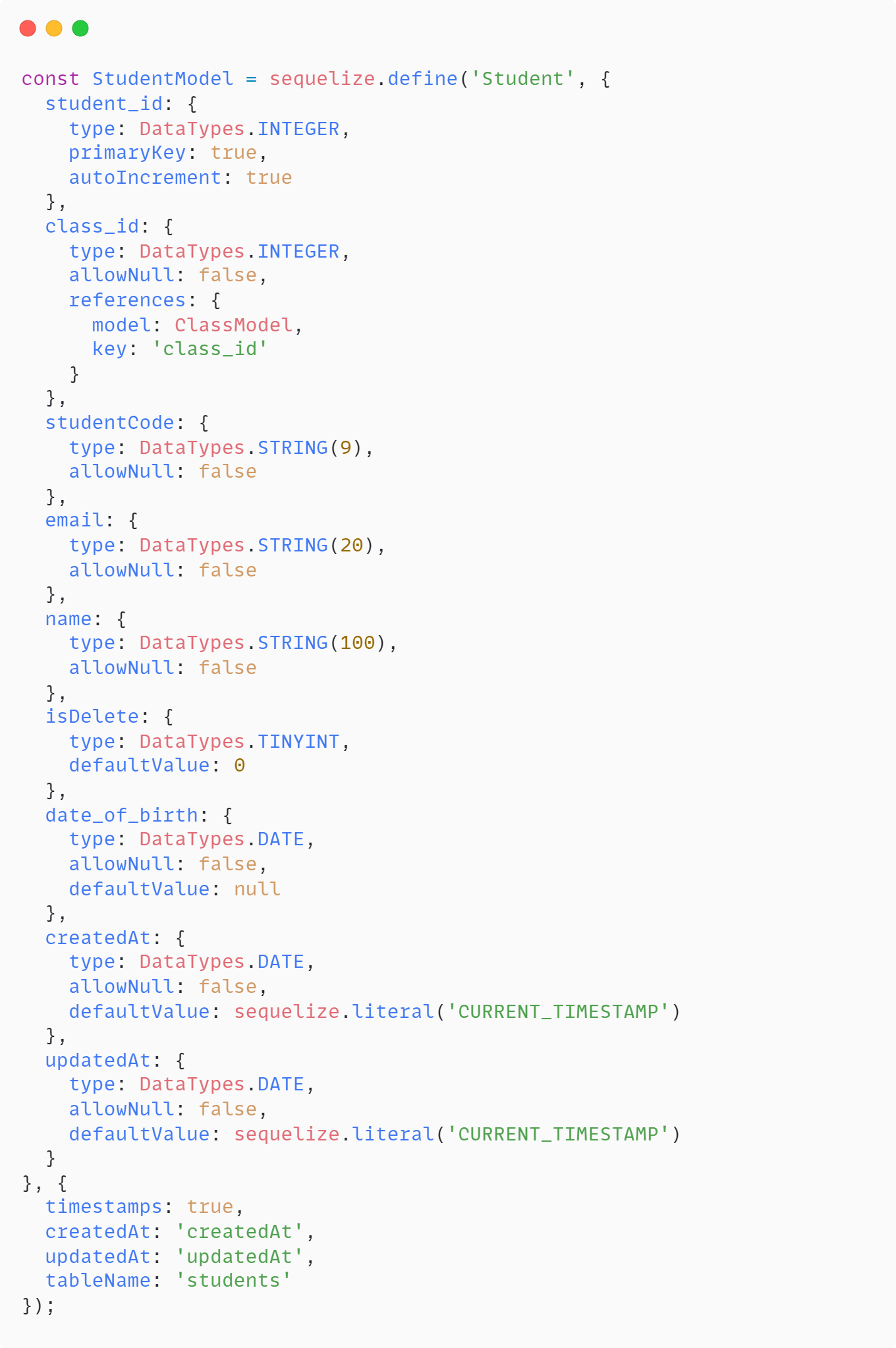
Hình 3.8. Import các thư viên tạo modal

**DataTypes**: Được import từ Sequelize để xác định kiểu dữ liệu của các thuộc tính trong mô hình.

**sequelize**: Instance của Sequelize, được cấu hình để kết nối tới cơ sở dữ liệu, được import từ file cấu hình (../config/database).

**ClassModel**: Mô hình lớp học, được import để thiết lập mối quan hệ với mô hình sinh viên.

**Định nghĩa mô hình**

****

Hình 3.9. Định nghĩa modal

sequelize.define('Student', {...}): Hàm định nghĩa mô hình với tên là Student.

student\_id: Khóa chính, tự động tăng.

class\_id: Khóa ngoại, tham chiếu tới class\_id trong ClassModel.

studentCode: Mã sinh viên, không được phép để trống.

email: Email sinh viên, không được phép để trống.

name: Tên sinh viên, không được phép để trống.

isDelete: Cờ đánh dấu xóa, mặc định là 0 (chưa bị xóa).

date\_of\_birth: Ngày sinh, không được phép để trống.

createdAt: Ngày tạo bản ghi, mặc định là thời gian hiện tại.

updatedAt: Ngày cập nhật bản ghi, mặc định là thời gian hiện tại.

timestamps: true: Tự động thêm các trường createdAt và updatedAt.

tableName: 'students': Tên bảng trong cơ sở dữ liệu là students.

**Thiết lập quan hệ giữa mô hình**

A close-up of a computer code

Description automatically generated

Hình 3.10. Thiết lập quan hệ mô hình

**belongsTo**: Thiết lập quan hệ "belongs to" giữa StudentModel và ClassModel qua khóa ngoại class\_id.

**Xuất mô hình**

A white background with black and white clouds

Description automatically generated with medium confidence

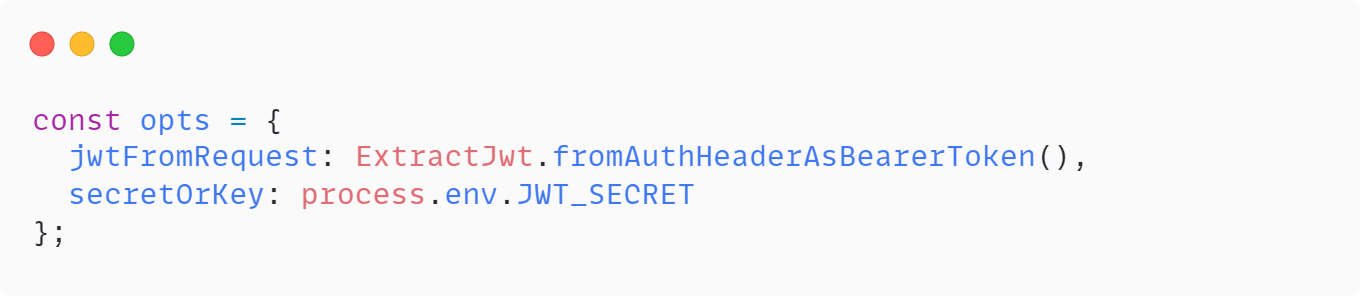
Hình 3.11. Xuất mô hình

**module.exports**: Xuất khẩu mô hình StudentModel để có thể sử dụng trong các module khác của ứng dụng.

### Xác thực người dùng

**Cấu hình bảo mật:**

Đoạn mã trên cấu hình passport.js sử dụng JWT (JSON Web Token) để xác thực người dùng thông qua passport-jwt strategy.



Hình 3.12. Cấu hình JWT Strategy

opts: Một đối tượng chứa các tùy chọn cho JwtStrategy.

jwtFromRequest: Xác định cách JWT được lấy từ request. Ở đây sử dụng fromAuthHeaderAsBearerToken() để lấy JWT từ tiêu đề Authorization dưới dạng Bearer token.

secretOrKey: Khóa bí mật để xác thực JWT, được lấy từ biến môi trường process.env.JWT\_SECRET.

A computer screen shot of a code

Description automatically generated

Hình 3.13. Sử dụng JWT Strategy với Passport

passport.use: Hàm này đăng ký JwtStrategy với Passport.

new JwtStrategy(opts, async (jwt\_payload, done) => {...}): Khởi tạo một strategy mới với các tùy chọn đã cấu hình.

jwt\_payload: Payload của JWT sau khi được giải mã.

done: Hàm callback được gọi sau khi hoàn thành quá trình xác thực.

TeacherModel.findByPk(jwt\_payload.id): Tìm người dùng trong cơ sở dữ liệu bằng ID từ payload của JWT.

Nếu tìm thấy người dùng, hàm done được gọi với người dùng đó.

Nếu không tìm thấy, hàm done được gọi với false.

Nếu có lỗi xảy ra trong quá trình tìm kiếm, hàm done được gọi với lỗi đó.

**Xây dựng middleware xác thực**

A close-up of a computer screen

Description automatically generated

Hình 3.14. Kiểm tra Access Token

req.cookies.accessToken: Lấy Access Token từ cookie của yêu cầu.

Nếu không có Access Token, trả về mã trạng thái 401 cùng với thông báo lỗi.

A computer code with text

Description automatically generated with medium confidence

Hình 3.15. Xác thực Access Token

jwt.verify(token, process.env.JWT\_SECRET): Xác thực Access Token bằng khóa bí mật (JWT\_SECRET).

TeacherModel.findByPk(decoded.id): Tìm người dùng bằng ID được giải mã từ Access Token.

Nếu không tìm thấy người dùng, trả về mã trạng thái 401 cùng với thông báo lỗi.

A computer code with text

Description automatically generated

Hình 3.16. Kiểm tra Refresh Token

req.cookies.refreshToken: Lấy Refresh Token từ cookie của yêu cầu.

RefreshTokenModel.findOne: Tìm Refresh Token trong cơ sở dữ liệu với token và teacher\_id.

Kiểm tra xem Refresh Token có tồn tại, không bị thu hồi hoặc hết hạn. Nếu không hợp lệ, trả về mã trạng thái 401 cùng với thông báo lỗi.

### Tạo và Quản Lý API Routes với Express.js

Router trong Express.js giúp quản lý các tuyến đường một cách có tổ chức và dễ bảo trì. Nó cho phép bạn nhóm các tuyến đường liên quan lại với nhau và áp dụng middleware cho nhóm đó.

A screen shot of a computer code

Description automatically generated

Hình 3.17. Định nghĩa các tuyến đường cho môn học với middleware xác thực

router.get('/subjects', ensureAuthenticated, SubjectController.index):

Phương thức: GET

Đường dẫn: /subjects

Middleware: ensureAuthenticated để xác thực người dùng trước khi cho phép truy cập.

Hành động: Gọi phương thức index của SubjectController để lấy danh sách tất cả các môn học.

router.post('/subject', ensureAuthenticated, SubjectController.create):

Phương thức: POST

Đường dẫn: /subject

Middleware: ensureAuthenticated để xác thực người dùng trước khi cho phép truy cập.

Hành động: Gọi phương thức create của SubjectController để tạo một môn học mới.

router.get('/subject/', ensureAuthenticated, SubjectController.getByID):

Phương thức: GET

Đường dẫn: /subject/:subject\_id

Middleware: ensureAuthenticated để xác thực người dùng trước khi cho phép truy cập.

Hành động: Gọi phương thức getByID của SubjectController để lấy thông tin chi tiết của một môn học cụ thể dựa trên subject\_id.

## Xây dựng hệ thống Frontend

### Cấu hình kết nối với thư viện, framework

**Tailwild**

**A computer screen shot of a code

Description automatically generated**

Hình 3.18. Cấu hình tệp tailwind.config.js

Tệp cấu hình này giúp tích hợp Tailwind CSS với NextUI trong dự án React. Nó xác định các tệp cần quét để tìm các lớp Tailwind, mở rộng chủ đề nếu cần, kích hoạt chế độ tối thông qua lớp class, và tích hợp các thành phần giao diện từ NextUI. Điều này giúp bạn dễ dàng sử dụng các thành phần giao diện hiện đại và tùy chỉnh chúng theo ý muốn trong dự án React của mình.

**FireBase**

**A screen shot of a computer code

Description automatically generated**

Hình 3.19. Cấu hình kết nối fire base

### Tạo Giao Diện Người Dùng

Mục tiêu: Phát triển giao diện người dùng (UI) thân thiện và dễ sử dụng.

Công cụ: Sử dụng React và các thư viện UI như NextUI, Ant Design, hoặc Tailwind CSS.

A computer screen shot of a computer screen

Description automatically generated

Hình 3.20. Khởi tạo root và render ứng dụng

ReactDOM.createRoot: Tạo một gốc (root) mới để render ứng dụng React.

React.StrictMode: Chế độ kiểm tra nghiêm ngặt để phát hiện các vấn đề tiềm ẩn trong ứng dụng.

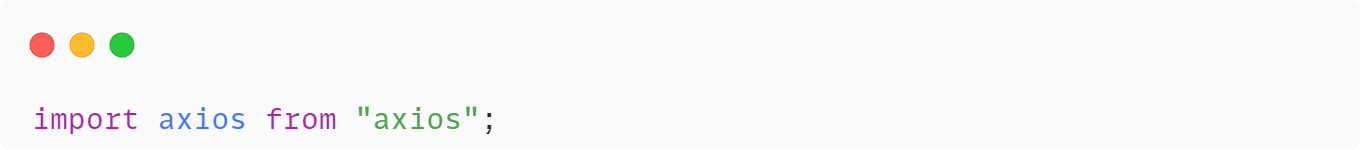
BrowserRouter: Cung cấp các tính năng điều hướng cho ứng dụng.

NextUIProvider: Cung cấp các thành phần UI từ NextUI.

App: Thành phần gốc được render bên trong các provider.

### Cấu Hình Axios

**Import thư viện Axios**

****

Hình 3.21. Import thư viện Axios

**Axios là t**hư viện giúp thực hiện các yêu cầu HTTP dễ dàng.

**Tạo Instance Axios với Cấu Hình Mặc Định**

A computer screen shot of a computer

Description automatically generated

Hình 3.22. Instance axios với cấu hình mặc định

axios.create: Tạo một instance Axios mới với các cấu hình mặc định.

withCredentials: Cho phép gửi cookie trong các yêu cầu cross-origin.

baseURL: URL cơ bản cho tất cả các yêu cầu được cấu hình từ biến môi trường.

**Thiết Lập Interceptors cho Axios**

**A screenshot of a computer code

Description automatically generated**

Hình 3.23. Thiết lập interceptors cho Axios

**interceptors.response.use**: Thiết lập interceptor cho phản hồi từ Axios.

response => response: Trả về phản hồi nếu không có lỗi.

async error => {...}: Hàm bất đồng bộ để xử lý lỗi.

originalRequest: Lưu trữ yêu cầu ban đầu để thử lại sau khi lấy token mới.

error.response.status === 401: Kiểm tra lỗi 401 (Unauthorized).

!originalRequest.\_retry: Đảm bảo yêu cầu không được thử lại nhiều lần.

originalRequest.\_retry = true: Đánh dấu yêu cầu là đã thử lại.

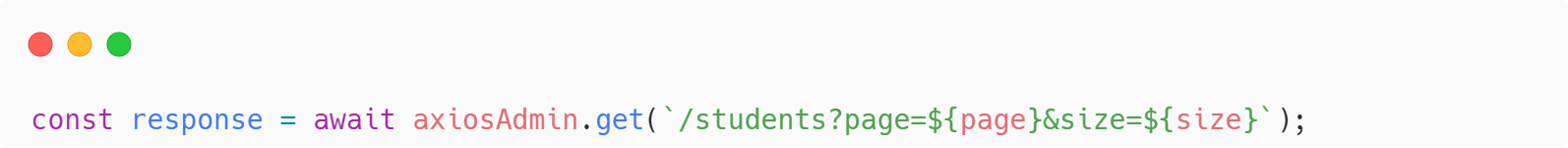
axios.post(${process.env.REACT\_APP\_API\_DOMAIN\_CLIENT}/refresh-token, {}, { withCredentials: true }): Gửi yêu cầu đến endpoint /refresh-token để lấy token mới.

response.data.accessToken: Lấy token mới từ phản hồi.

originalRequest.headers['Authorization'] = Bearer ${newAccessToken}: Cập nhật header Authorization với token mới.

axiosAdmin(originalRequest): Thử lại yêu cầu ban đầu với token mới.

**Sử dụng**

****

Hình 3.24. Ví dụ về sử dụng axiosAdmin

axiosAdmin.get: Sử dụng phương thức GET của instance axiosAdmin để gửi yêu cầu.

/students?page=${page}&size=${size}: Endpoint API.

# KẾT QUẢ NGHIÊN CỨU

## Kết quả nghiên cứu Backend

### Khởi chạy

Khởi chạy bằng cách chạy câu lệnh npm start

A screenshot of a computer code

Description automatically generated

Hình 4.1. Khởi chạy backend

Khi bạn chạy lệnh npm start, nodemon sẽ giám sát tất cả các tệp trong dự án và tự động khởi động lại server nếu có bất kỳ thay đổi nào. Ta sẽ thấy thông báo trong console cho biết server đang chạy và kết nối cơ sở dữ liệu thành công.

### Xác thực người dùng

Đăng nhập người dùng bằng teacherCode và password



Hình 4.2. Api xử lý đăng nhập

Dữ liệu đầu vào teacherCode, password từ body của yêu cầu HTTP POST do người dùng gửi đến.

A computer screen shot of text

Description automatically generated

Hình 4.3. Tìm người dùng trong cơ sở dữ liệu

Sử dụng hàm findOne của mô hình TeacherModel để tìm kiếm người dùng trong cơ sở dữ liệu dựa trên teacherCode.

Nếu không tìm thấy người dùng nào khớp với teacherCode, trả về mã trạng thái 400 và thông báo lỗi.

A computer screen shot of a computer screen

Description automatically generated

Hình 4.4. Kiểm tra mật khẩu

Dòng 1: Sử dụng bcrypt.compare để so sánh mật khẩu do người dùng nhập với mật khẩu đã mã hóa được lưu trữ trong cơ sở dữ liệu.

Dòng 2-4: Nếu mật khẩu không khớp, trả về mã trạng thái 400 và thông báo lỗi.

A computer screen shot of a program

Description automatically generated

Hình 4.5. khởi tạo các biến cần thiết.

Dòng 1: Tạo đối tượng payload chứa thông tin người dùng cần thiết cho mã thông báo.

Dòng 2: Sử dụng jwt.sign để tạo mã thông báo truy cập (access token) với thời gian hết hạn là 30 phút.

Dòng 3: Tạo mã thông báo làm mới (refresh token) với thời gian hết hạn là 7 ngày.

A computer code with text

Description automatically generated with medium confidence

Hình 4.6. Thu hồi và hết hạn các biến cần thiết.

Sử dụng hàm update của mô hình RefreshTokenModel để cập nhật trạng thái của các mã thông báo làm mới cũ, đánh dấu chúng là đã thu hồi và hết hạn, dựa trên teacher\_id của người dùng.

A close-up of a sign

Description automatically generated

Hình 4.7. Lưu token vào cơ sở dữ liệu

Sử dụng hàm create của mô hình RefreshTokenModel để lưu mã thông báo làm mới mới vào cơ sở dữ liệu, gắn với teacher\_id của người dùng.

A computer code with numbers and symbols

Description automatically generated

Hình 4.8. Đặt mã thông báo trong cookies.

Đặt mã thông báo truy cập vào cookie với các tùy chọn bảo mật như httpOnly, secure, sameSite và thời gian hết hạn là 30 phút đối với accessToken, 7 ngày đối với refreshToken.

### Các Api

Khi chưa đăng nhập

A screenshot of a computer

Description automatically generated

Khi chưa đăng nhập và thực hiện yêu cầu GET đến đường dẫn http://localhost:1509/api/user, hệ thống sẽ trả về một thông báo lỗi với mã trạng thái 401 Unauthorized. Thông báo lỗi chi tiết sẽ là một đối tượng JSON với nội dung như sau:

A green text on a white background

Description automatically generated

Hình 4.9. Phải hồi khi chưa đăng nhập khi gọi end point

Điều này cho biết rằng yêu cầu đã thiếu mã thông báo truy cập (access token) cần thiết để xác thực và ủy quyền người dùng. Để khắc phục lỗi này, bạn cần đảm bảo rằng mã thông báo truy cập hợp lệ được gửi kèm theo yêu cầu.

Khi đã đăng nhập

A screenshot of a computer

Description automatically generated

Khi có token khi đăng nhập thì sẽ lấy accessToken trên cookie

### Swagger

Các Api cũng được triển khai trên Swagger cho người dùng thao tác dễ dàng

A screenshot of a computer

Description automatically generated

Hình 4.10. Giao diện swagger

Đây là nơi hiển thị cũng như các API và cũng là nơi tương tác.

## Kết quả nghiêm cứu Frontend

### Giao diện đăng nhập

A screenshot of a computer

Description automatically generated

Hình 4.11. Giao diện đăng nhập.

Người dùng đăng nhập bằng tài khoảng được cấp bao gồm mã giáo viên và mật khẩu.

### Giao diện trang chủ

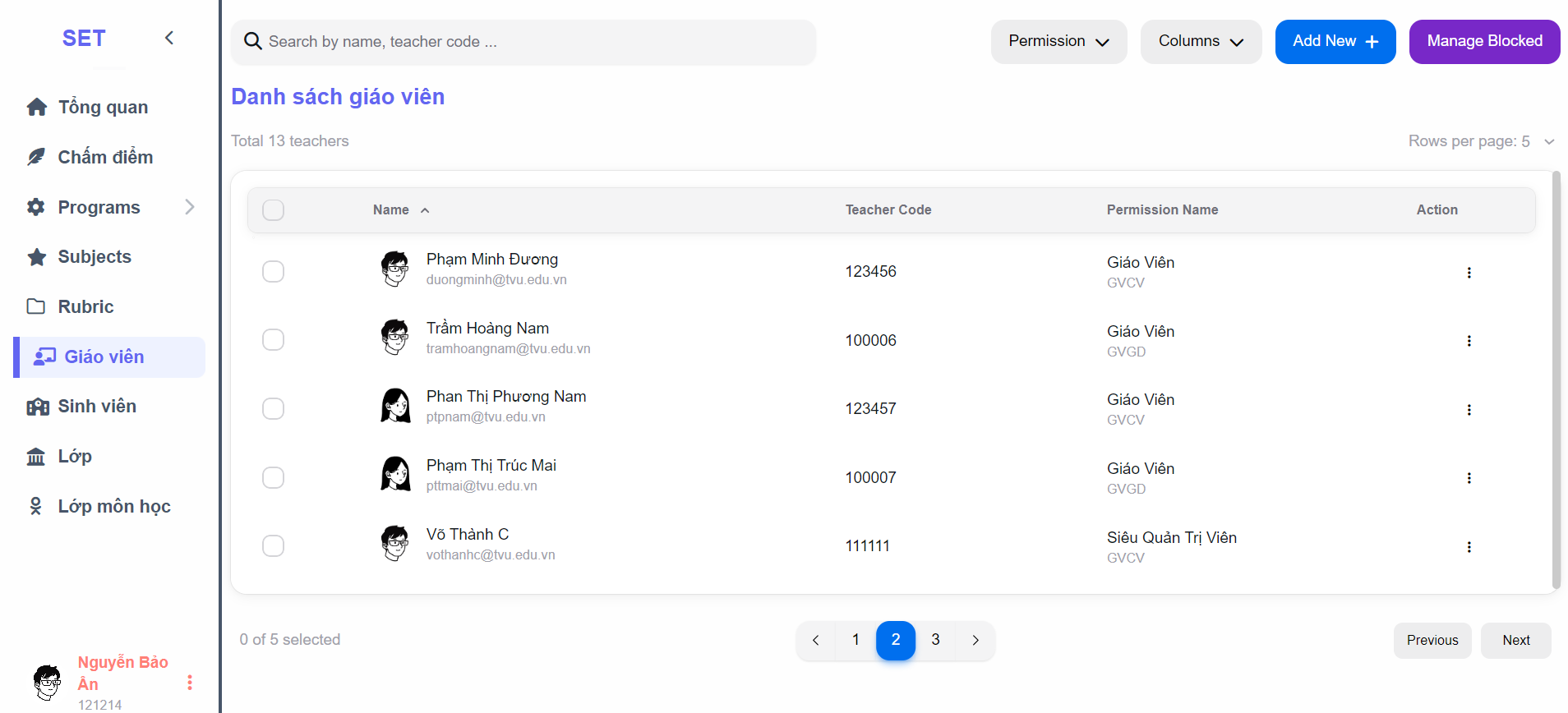
A screenshot of a computer

Description automatically generated

Hình 4.12. Giao diện trang chủ

Trang chủ là nơi có biểu đồ hiển thị trực quan kết quả học tập của các sinh viên. Tại đây, người dùng có thể dễ dàng theo dõi và so sánh thành tích học tập của các sinh viên thông qua các biểu đồ tương tác. Các biểu đồ này cung cấp cái nhìn tổng quan về điểm số, tiến bộ học tập theo thời gian, và sự phân bố thành tích giữa các sinh viên. Giao diện trực quan và sinh động giúp người dùng nhanh chóng nắm bắt thông tin và đưa ra các quyết định phù hợp dựa trên dữ liệu trực quan và phân tích chi tiết.

### Giao diện quản lý giáo viên



Hình 4.13. Danh sách giáo viên.

Trang quản lý giáo viên là một phần của hệ thống quản lý học tập, nơi cung cấp giao diện trực quan để quản lý thông tin và quyền truy cập của giáo viên. Trang này bao gồm các thành phần chính như sau:

Thanh tìm kiếm: Nằm ở phía trên cùng của trang, thanh tìm kiếm cho phép người dùng tìm kiếm giáo viên theo tên hoặc mã số giáo viên. Người dùng chỉ cần nhập tên hoặc mã số vào ô tìm kiếm và hệ thống sẽ hiển thị danh sách các giáo viên tương ứng.

Danh sách giáo viên: Phần trung tâm của trang hiển thị danh sách giáo viên với các thông tin cơ bản bao gồm tên, mã số giáo viên, quyền truy cập và các hành động có thể thực hiện. Mỗi giáo viên được hiển thị dưới dạng một hàng trong bảng, với các cột:

Name: Tên của giáo viên.

Teacher Code: Mã số giáo viên.

Permission Name: Quyền truy cập của giáo viên (ví dụ: Giáo viên, Quản trị viên, Siêu quản trị viên).

Action: Các hành động có thể thực hiện, chẳng hạn như xem chi tiết, chỉnh sửa hoặc xóa giáo viên.

Bộ lọc: Người dùng có thể sử dụng các bộ lọc để lọc danh sách giáo viên theo quyền truy cập hoặc các tiêu chí khác. Các bộ lọc này giúp người dùng dễ dàng tìm kiếm và quản lý giáo viên theo các nhóm cụ thể.

Nút Columns có thể lựa chọn các cột nào được hiển thị trong bảng

Nút thêm mới và quản lý khóa: Nút "Add New" cho phép người dùng thêm giáo viên mới vào hệ thống. Nút "Manage Blocked" cho phép quản lý các giáo viên bị khóa, chẳng hạn như mở khóa hoặc xem chi tiết lý do khóa.

Điều hướng phân trang: Ở phía dưới của danh sách giáo viên, hệ thống cung cấp các nút điều hướng phân trang để người dùng có thể di chuyển qua các trang khác nhau của danh sách giáo viên, giúp dễ dàng quản lý khi số lượng giáo viên lớn.

Menu điều hướng bên trái: Bao gồm các mục chính của hệ thống quản lý học tập như Tổng quan, Chấm điểm, Programs, Subjects, Rubric, Giáo viên, Sinh viên, Lớp, và Lớp môn học. Người dùng có thể dễ dàng chuyển đổi giữa các chức năng khác nhau của hệ thống thông qua menu này.

**Khi chọn một vài giáo viên**

**A screenshot of a computer

Description automatically generated**

Hình 4.14. Giao diên khi tương tác với bảng giáo viên.

Khi nhấn vào giáo viên bất kì sẽ hiển thị thanh chứ năng ở trên hiển thị số lượng giáo viên được chọn. Khi ta nhấn vào Download sẽ tả file excel chứa thông tin của giáo viên ta đã chọn. Hoặc nhấn Block sẽ chặn các giáo viên đã chọn.

**Khôi phục giáo viên đã chặn**

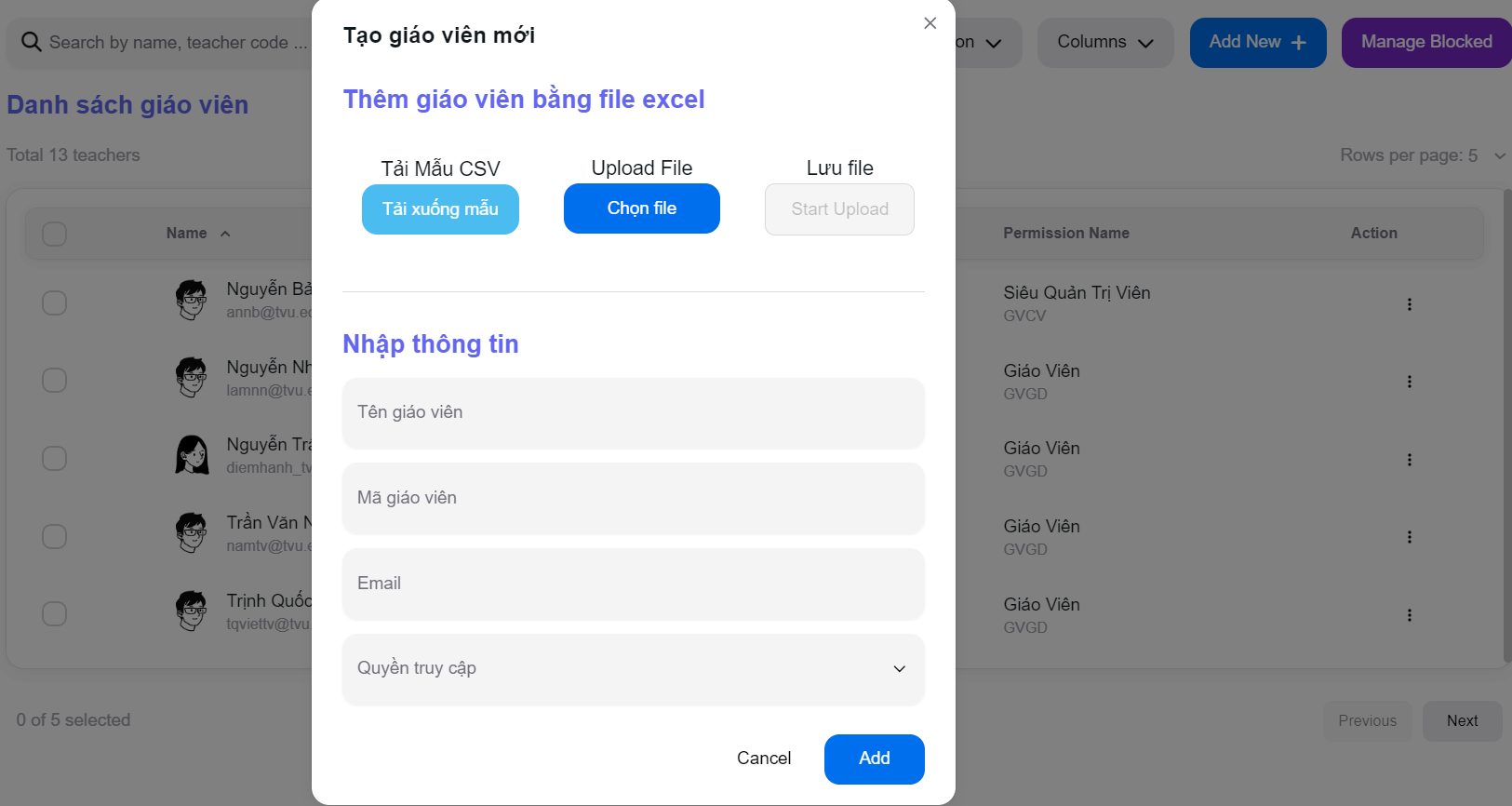
**A screenshot of a chat

Description automatically generated**

Hình 4.15. Giao diện danh sách giáo viên đã chặn

Khi nhấn vào “Manage Blocked” sẽ mở danh sách chặn. Nhấn những giáo viên ta muốn bỏ chặn nhấn “Unlock”.

**Tạo giáo viên mới**



Hình 4.16. Giao diện tạo giáo viên mới

Khi nhấn vào Add new thì giao diện tạo giáo viên mới sẽ xuất hiện. Có hai lựa chọn để tạo giáo viên mới đó là nhập giáo viên mới bằng excel, hoặc là nhập giáo viên thông qua form bên với.

### Giao diện quản lý sinh viên

A screenshot of a computer

Description automatically generated

Hình 4.17. Giao diện quản lý sinh viên.

# KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

## Kết luận

(Trình bày những kết quả đạt được, những đóng góp mới và những đề xuất mới. Phần kết luận cần ngắn gọn, không có lời bàn và bình luận thêm)

## Hướng phát triển

(Kiến nghị về những hướng nghiên cứu tiếp theo)

**DANH MỤC TÀI LIỆU THAM KHẢO**

(Trình bày tài liệu tham khảo theo chuẩn IEEE, chỉ liệt kê các tài liệu có sử dụng để tham khảo và phải có trích dẫn trong bài. Khuyến khích sử dụng phần mềm quản lý tài liệu tham khảo, ví dụ: Zotero, Mendeley, Endnote,…)