

TRƯỜNG KỸ THUẬT VÀ CÔNG NGHỆ
KHOA CÔNG NGHỆ THÔNG TIN



THỰC TẬP ĐỒ ÁN CƠ SỞ NGÀNH
HỌC KỲ I, NĂM HỌC 2025-2026

**THIẾT KẾ WEBSITE
HỖ TRỢ HỌC TẬP CHO SINH VIÊN
CÔNG NGHỆ THÔNG TIN
TRƯỜNG ĐẠI HỌC TRÀ VINH**

Giảng viên hướng dẫn:
ThS. Nguyễn Hoàng Duy Thiện

Sinh viên thực hiện:
Họ tên: Lê Tuấn Huy
MSSV: 110123105
Lớp: DA23TTC

Vĩnh Long, tháng 01 năm 2026

TRƯỜNG KỸ THUẬT VÀ CÔNG NGHỆ
KHOA CÔNG NGHỆ THÔNG TIN



THỰC TẬP ĐỒ ÁN CƠ SỞ NGÀNH
HỌC KỲ I, NĂM HỌC 2025-2026

**THIẾT KẾ WEBSITE
HỖ TRỢ HỌC TẬP CHO SINH VIÊN
CÔNG NGHỆ THÔNG TIN
TRƯỜNG ĐẠI HỌC TRÀ VINH**

Giảng viên hướng dẫn:
ThS. Nguyễn Hoàng Duy Thiện

Sinh viên thực hiện:
Họ tên: Lê Tuấn Huy
MSSV: 110123105
Lớp: DA23TTC

Vĩnh Long, tháng 01 năm 2026

[illegible]

Giảng viên hướng dẫn
(Ký tên và ghi rõ họ tên)

Lê Tuấn Huy

NHẬN XÉT CỦA THÀNH VIÊN HỘI ĐỒNG

[illegible]

Vĩnh Long, ngày tháng năm 2026

Thành viên hội đồng
(Ký tên và ghi rõ họ tên)

LỜI CẢM ƠN

Trước hết, tôi xin gửi lời cảm ơn chân thành đến Khoa Công nghệ Thông tin, Trường Kỹ thuật và Công nghệ đã tạo điều kiện thuận lợi để tôi được học tập, nghiên cứu và thực hiện đồ án trong môi trường học tập chuyên nghiệp.

Đặc biệt, tôi xin trân trọng gửi lời cảm ơn chân thành và sâu sắc đến thầy Nguyễn Hoàng Duy Thiện, người đã tận tình hướng dẫn, đóng góp nhiều ý kiến quý báu và luôn động viên tôi trong suốt quá trình thực hiện đồ án. Những chỉ dẫn và kinh nghiệm của thầy là nền tảng quan trọng giúp tôi hoàn thành đề tài đúng hướng và đạt được các mục tiêu đề ra.

Trong quá trình thực hiện, tôi đã cố gắng nghiên cứu và vận dụng những kiến thức đã học để hoàn thành đồ án một cách tốt nhất. Kết quả đạt được là quá trình nỗ lực của bản thân tôi. Tuy nhiên, do thời gian và kinh nghiệm còn hạn chế, báo cáo không tránh khỏi những thiếu sót. Tôi rất mong nhận được sự đóng góp ý kiến của quý thầy (cô) để đề tài đồ án cơ sở ngành của tôi được hoàn thiện hơn.

Tôi xin chân thành cảm ơn!

Sinh viên thực hiện

Lê Tuấn Huy

MỤC LỤC

MỞ ĐẦU.....	13
1. Lý do chọn đề tài.....	13
2. Mục tiêu nghiên cứu.....	13
3. Đối tượng nghiên cứu	13
4. Phạm vi nghiên cứu.....	13
5. Phương pháp nghiên cứu.....	14
6. Ý nghĩa đề tài	14
CHƯƠNG 1. TỔNG QUAN.....	15
1.1. Giới thiệu tổng quan đề tài.....	15
1.2. Mục đích xây dựng hệ thống.....	15
1.3. Các chức năng chính của hệ thống.....	15
1.3.1. Chức năng dành cho sinh viên	15
1.3.2. Chức năng dành cho giảng viên	15
1.3.3. Chức năng dành cho quản trị viên.....	16
1.4. Công nghệ sử dụng.....	16
CHƯƠNG 2. NGHIÊN CỨU LÝ THUYẾT	17
2.1. Kiến trúc hệ thống.....	17
2.1.1. Mô hình Client - Server.....	17
2.1.1.1. Mô hình tổng thể.....	17
2.1.1.2. Ưu điểm của mô hình Client – Server	17
2.1.1.3. Nhược điểm của mô hình Client – Server	17
2.1.2. Tìm hiểu về RESTful API.....	18
2.1.2.1. RESTful API là gì?.....	18
2.1.2.2. Cách thức hoạt động của API RESTful	18
2.2. Công nghệ Frontend.....	19
2.2.1. Tìm hiểu về ReactJS.....	19
2.2.1.1. Giới thiệu về ReactJS	19
2.2.1.2. Kiến trúc và nguyên lý hoạt động của ReactJS	19
2.2.1.3. JSX trong ReactJS	20
2.2.1.4. Component trong ReactJS	20
2.2.1.5. State và Hooks trong ReactJS.....	21

2.2.1.6.	Điều hướng trang trong ReactJS	22
2.2.1.7.	Ưu điểm và nhược điểm của ReactJS	22
2.2.2.	Tìm hiểu về Tailwind CSS	23
2.2.2.1.	Giới thiệu về Tailwind CSS	23
2.2.2.2.	Nguyên lý hoạt động của Tailwind CSS	23
2.2.2.3.	Kết hợp Tailwind CSS với ReactJS	24
2.2.2.4.	Ưu điểm và nhược điểm của Tailwind CSS	24
2.2.3.	Tìm hiểu về Vite	24
2.2.3.1.	Giới thiệu về Vite	24
2.2.3.2.	Vai trò của Vite trong dự án ReactJS	25
2.3.	Công nghệ Backend	26
2.3.1.	Tìm hiểu về Node.js	26
2.3.1.1.	Giới thiệu về Node.js	26
2.3.1.2.	Cơ chế hoạt động và kiến trúc độc đáo của Node.js	26
2.3.1.3.	Hệ sinh thái và lợi ích	26
2.3.1.4.	Ứng dụng phổ biến	27
2.3.2.	Tìm hiểu về ExpressJS	27
2.3.2.1.	Giới thiệu về ExpressJS	27
2.3.2.2.	Vai trò và ứng dụng của ExpressJS	27
2.3.2.3.	Các công cụ cốt lõi của ExpressJS	28
2.3.2.4.	RESTful API trong ExpressJS	28
2.3.2.5.	Giao tiếp giữa Frontend và Backend thông qua API RESTful ...	29
2.4.	Công nghệ cơ sở dữ liệu	29
2.4.1.	Tìm hiểu về MySQL	29
2.4.1.1.	Giới thiệu về MySQL	29
2.4.1.2.	Các khái niệm cơ bản trong MySQL	29
2.4.1.3.	Ưu điểm và nhược điểm của MySQL	30
2.4.2.	Tìm hiểu về XAMPP và phpMyAdmin	31
2.4.2.1.	Giới thiệu về XAMPP	31
2.4.2.2.	Ứng dụng của XAMPP	32
2.4.2.3.	Giới thiệu về và phpMyAdmin	32
2.4.2.4.	Các tính năng nổi bật của phpMyAdmin	32

2.5. Xác thực và phân quyền người dùng	33
2.5.1. Xác thực người dùng (Authentication).....	33
2.5.1.1. Mã hóa mật khẩu bằng bcrypt	33
2.5.1.2. Xác thực người dùng bằng JSON Web Token (JWT).....	34
2.5.1.3. Tìm hiểu về Access Token	35
2.5.1.4. Tìm hiểu về Refresh Token	35
2.5.2. Phân quyền người dùng (Authorization).....	36
2.5.2.1. Phân quyền người dùng là gì?	36
2.5.2.2. Phân quyền người dùng theo vai trò (RBAC)	36
CHƯƠNG 3. HIỆN THỰC HÓA NGHIÊN CỨU	37
3.1. Mô tả bài toán.....	37
3.2. Phân tích thiết kế hệ thống.....	37
3.2.1. Đặc tả yêu cầu hệ thống	37
3.2.1.1. Yêu cầu chức năng.....	37
3.2.1.2. Yêu cầu phi chức năng.....	38
3.2.2. Thiết kế chức năng hệ thống	39
3.2.2.1. Sơ đồ Use Case	39
3.2.2.2. Mô tả chi tiết các Use Case.....	39
3.2.3. Thiết kế quy trình	42
3.2.3.1. Mô hình DFD mức 0.....	42
3.2.3.2. Mô hình DFD mức 1.....	43
3.2.4. Thiết kế dữ liệu.....	43
3.2.4.1. Danh sách các thực thể và mối kết hợp	44
3.2.4.2. Chi tiết các thực thể và mối kết hợp	45
3.2.4.3. Mô hình ERD.....	49
3.2.5. Thiết kế giao diện.....	50
3.2.5.1. Giao diện trang đăng ký / đăng nhập.....	50
3.2.5.2. Giao diện trang chủ.....	51
3.2.5.3. Giao diện trang khóa học của tôi	51
3.2.5.4. Giao diện trang chi tiết khóa học.....	52
3.2.5.5. Giao diện trang học tập.....	53
3.2.5.6. Giao diện trang quản lý giảng dạy.....	53

3.2.5.7. Giao diện trang quản trị	54
CHƯƠNG 4. KẾT QUẢ NGHIÊN CỨU	55
4.1. Tổng quan kết quả đạt được	55
4.2. Dữ liệu mẫu của hệ thống	55
4.2.1. Dữ liệu mẫu tài khoản và phân quyền	55
4.2.2. Dữ liệu nghiệp vụ phát sinh từ thao tác trên giao diện	55
4.3. Giao diện và chức năng chính của hệ thống	56
4.3.1. Giao diện cho sinh viên	56
4.3.1.1. Giao diện trang chủ	56
4.3.1.2. Giao diện trang khóa học của tôi	57
4.3.1.3. Giao diện trang chi tiết khóa học	58
4.3.1.4. Giao diện trang học tập	59
4.3.2. Giao diện cho giảng viên	60
4.3.2.1. Giao diện trang quản lý giảng dạy	60
4.3.2.2. Giao diện trang quản lý khóa học	60
4.3.2.3. Giao diện trang tạo khóa học	61
4.3.2.4. Giao diện trang sửa khóa học	61
4.3.2.5. Giao diện quản lý bài học	62
4.3.2.6. Giao diện trang tạo bài học	62
4.3.2.7. Giao diện trang sửa bài học	63
4.3.2.8. Giao diện trang quản lý quiz	63
4.3.2.9. Giao diện trang thêm và sửa quiz	64
4.3.3. Giao diện cho quản trị viên	65
4.3.3.1. Giao diện trang dashboard quản trị	65
4.3.3.2. Giao diện trang quản lý người dùng	65
4.3.3.3. Giao diện trang quản lý khóa học	66
4.3.3.4. Giao diện trang tạo giảng viên	66
4.4. Đánh giá hệ thống	67
4.4.1. Mức độ đáp ứng yêu cầu đề tài	67
4.4.2. Đánh giá về giao diện và trải nghiệm người dùng	67
4.4.3. Đánh giá về hiệu năng và tính ổn định	67
4.4.4. Hạn chế của hệ thống	67

CHƯƠNG 5. KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN.....	68
5.1. Kết luận	68
5.2. Hướng phát triển	68
DANH MỤC TÀI LIỆU THAM KHẢO	69

DANH MỤC HÌNH ẢNH

Hình 2.1. Mô hình Client – Server	17
Hình 3.1. Mô hình Use Case	39
Hình 3.2. Mô hình luồng dữ liệu mức 0	42
Hình 3.3. Mô hình luồng dữ liệu mức 1	43
Hình 3.4. Mô hình ERD.....	49
Hình 3.5. Giao diện trang đăng ký.....	50
Hình 3.6. Giao diện trang đăng nhập	50
Hình 3.7. Giao diện trang chủ	51
Hình 3.8. Giao diện trang khóa học của tôi.....	52
Hình 3.9. Giao diện chi tiết khóa học	52
Hình 3.10. Giao diện trang học tập	53
Hình 3.11. Giao diện trang quản lý giảng dạy	54
Hình 3.12. Giao diện trang quản trị	54
Hình 4.1. Kết quả giao diện trang chủ.....	56
Hình 4.2. Kết quả trang khóa học của tôi.....	57
Hình 4.3. Kết quả giao diện chi tiết khóa học.....	58
Hình 4.4. Kết quả giao diện trang học tập.....	59
Hình 4.5. Kết quả giao diện trang cho giảng viên.....	60
Hình 4.6. Kết quả giao diện trang quản lý khóa học	60
Hình 4.7. Kết quả giao diện trang tạo khóa học.....	61
Hình 4.8. Kết quả giao diện sửa khóa học.....	61
Hình 4.9. Kết quả giao diện trang quản lý bài học.....	62
Hình 4.10. Kết quả giao diện trang tạo bài học	62
Hình 4.11. Kết quả giao diện sửa bài học.....	63
Hình 4.12. Kết quả giao diện trang quản lý quiz.....	63
Hình 4.13. Kết quả giao diện thêm quiz.....	64
Hình 4.14. Kết quả giao diện trang sửa quiz	64
Hình 4.15. Kết quả giao diện trang dashboard quản trị.....	65
Hình 4.16. Kết quả giao diện trang quản lý người dùng	65
Hình 4.17. Kết quả giao diện trang quản lý khóa học	66
Hình 4.18. Kết quả giao diện trang tạo giảng viên.....	66

DANH MỤC BẢNG BIỂU

<i>Bảng 2.1. Ví dụ về Tailwind CSS</i>	23
<i>Bảng 2.2. RESTful API trong ExpressJS</i>	28
<i>Bảng 2.3. Các kiểu dữ liệu phổ biến trong MySQL</i>	30
<i>Bảng 3.1. Chi tiết các Use Case</i>	39
<i>Bảng 3.2. Danh sách các thực thể</i>	44
<i>Bảng 3.3. Danh sách các mối kết hợp</i>	44
<i>Bảng 3.4. Chi tiết thực thể User</i>	45
<i>Bảng 3.5. Chi tiết thực thể Courses</i>	45
<i>Bảng 3.6. Chi tiết thực thể Lessons</i>	46
<i>Bảng 3.7. Chi tiết thực thể Enrollments</i>	46
<i>Bảng 3.8. Chi tiết thực thể User_Progress</i>	47
<i>Bảng 3.9. Chi tiết thực thể Quiz_Questions</i>	47
<i>Bảng 3.10. Chi tiết thực thể Quiz_Results</i>	48
<i>Bảng 3.11. Chi tiết thực thể Notifications</i>	48
<i>Bảng 3.12. Chi tiết thực thể Refresh_tokens</i>	49

DANH MỤC TỪ VIẾT TẮT

Từ viết tắt	Ý nghĩa
API	Application Programming Interface
CRUD	Create – Read – Update – Delete
DB	Database
ERD	Entity Relationship Diagram
DFD	Data Flow Diagram
HTTP	Hypertext Transfer Protocol
JWT	JSON Web Token
RBAC	Role-Based Access Control
REST	Representational State Transfer
SQL	Structured Query Language
UI	User Interface

TÓM TẮT ĐỒ ÁN CƠ SỞ NGÀNH

Trong quá trình học tập, sinh viên ngành Công nghệ Thông tin Trường Đại học Trà Vinh phải tiếp cận khối lượng kiến thức lớn trong khi thời gian học trên lớp có hạn, dẫn đến việc một số nội dung chưa được nắm vững và khó ôn tập lại. Xuất phát từ thực tế đó, đề tài nghiên cứu và xây dựng một website hỗ trợ học tập nhằm giúp sinh viên có thể chủ động xem lại bài học, ôn tập kiến thức và tự đánh giá năng lực thông qua các bài kiểm tra trắc nghiệm. Đề tài được tiếp cận theo hướng xây dựng hệ thống học tập trực tuyến trên nền tảng web, áp dụng mô hình Client – Server với kiến trúc Frontend và Backend tách rời, sử dụng các công nghệ như ReactJS, Node.js, ExpressJS và MySQL. Kết quả đạt được là một hệ thống cơ bản đáp ứng nhu cầu học tập của sinh viên, hỗ trợ giảng viên trong việc quản lý nội dung khóa học và giúp quản trị viên quản lý dữ liệu hệ thống, đồng thời có khả năng mở rộng và phát triển trong tương lai.

MỞ ĐẦU

1. Lý do chọn đề tài

Trong quá trình học tập tại Trường Đại học Trà Vinh, sinh viên phải tiếp cận một khối lượng kiến thức lớn, đặc biệt là các môn học chuyên ngành Công nghệ Thông tin. Tuy nhiên, thời lượng học trên lớp có hạn, trong khi mỗi sinh viên có khả năng tiếp thu và tốc độ học tập khác nhau, dẫn đến việc một số kiến thức chưa được nắm vững ngay trong quá trình học.

Bên cạnh đó, sinh viên thường gặp khó khăn trong việc xem lại bài giảng, hệ thống hoá kiến thức đã học hoặc ôn tập trước các kỳ kiểm tra, thi cử. Việc học tập chỉ dựa vào tài liệu rời rạc hoặc ghi chép cá nhân đôi khi chưa mang lại hiệu quả cao, đặc biệt đối với các môn học có lượng kiến thức lớn và liên tục.

Xuất phát từ thực tế trên, tôi đã nảy sinh ý tưởng xây dựng một website học trực tuyến dành riêng cho sinh viên Trường Đại học Trà Vinh, nhằm hỗ trợ sinh viên trong việc học tập và ôn tập kiến thức ngoài giờ học trên lớp. Thông qua hệ thống này, sinh viên có thể dễ dàng truy cập các bài học theo từng môn, từng năm học, xem lại nội dung bài giảng và củng cố kiến thức đã học.

Từ những lý do trên, tôi quyết định lựa chọn đề tài “Thiết kế Website hỗ trợ học tập cho sinh viên Công nghệ Thông tin Trường Đại học Trà Vinh” làm đề án nghiên cứu và thực hiện.

2. Mục tiêu nghiên cứu

Mục tiêu của đề tài là xây dựng một hệ thống học trực tuyến cơ bản, hoạt động ổn định và phù hợp với nhu cầu học tập của sinh viên Trường Đại học Trà Vinh.

3. Đối tượng nghiên cứu

Đối tượng nghiên cứu của đề tài là hệ thống website học trực tuyến TVULEARN, bao gồm các chức năng phục vụ cho sinh viên, giảng viên và quản trị viên trong môi trường học tập trực tuyến.

4. Phạm vi nghiên cứu

Phạm vi nghiên cứu của đề tài được giới hạn trong việc xây dựng một ứng dụng web học trực tuyến phục vụ nội bộ cho sinh viên Trường Đại học Trà Vinh. Hệ thống tập trung vào các chức năng cơ bản như quản lý người dùng, khóa học, bài học, quiz và theo dõi tiến trình học tập.

Đề tài không hướng đến triển khai thương mại và không mở rộng cho đối tượng ngoài phạm vi sinh viên của Trường Đại học Trà Vinh. Các chức năng nâng cao như ứng dụng di động, livestream bài giảng hoặc tích hợp trí tuệ nhân tạo chưa được xem xét trong phạm vi nghiên cứu của đề tài.

5. Phương pháp nghiên cứu

Phương pháp nghiên cứu tài liệu: tìm hiểu các mô hình hệ thống học trực tuyến và các công nghệ web hiện đại.

Phương pháp nghiên cứu thực nghiệm: Thiết kế giao diện người dùng đảm bảo tính trực quan, dễ sử dụng cho sinh viên, thiết kế cơ sở dữ liệu phù hợp với hệ thống học trực tuyến.

6. Ý nghĩa đề tài

Đề tài mang ý nghĩa thực tiễn trong việc hỗ trợ sinh viên Trường Đại học Trà Vinh có thêm một công cụ học tập và ôn tập hiệu quả bên ngoài giờ học trên lớp. Hệ thống giúp sinh viên chủ động hơn trong việc tiếp cận kiến thức, củng cố nội dung đã học và nâng cao kết quả học tập.

Đồng thời, đề tài cũng giúp tác giả củng cố kiến thức chuyên môn về lập trình web, cơ sở dữ liệu và xây dựng hệ thống phần mềm, tạo nền tảng cho việc học tập và nghiên cứu trong tương lai.

CHƯƠNG 1. TỔNG QUAN

1.1. Giới thiệu tổng quan đề tài

Đề tài “Thiết kế Website hỗ trợ học tập cho sinh viên Công nghệ Thông tin Trường Đại học Trà Vinh” là một hệ thống được xây dựng nhằm hỗ trợ sinh viên Trường Đại học Trà Vinh trong quá trình học tập và ôn luyện kiến thức. Hệ thống cho phép sinh viên truy cập các khóa học, bài học và bài kiểm tra trắc nghiệm mọi lúc, mọi nơi thông qua nền tảng web.

Hệ thống được thiết kế với mục tiêu đơn giản, dễ sử dụng, phù hợp với nhu cầu học tập thực tế của sinh viên. Ngoài việc hỗ trợ sinh viên, TVULEARN còn cung cấp các chức năng cơ bản cho giảng viên trong việc tạo và quản lý nội dung khóa học, cũng như chức năng quản trị hệ thống dành cho quản trị viên.

1.2. Mục đích xây dựng hệ thống

Hỗ trợ sinh viên Trường Đại học Trà Vinh có môi trường học tập trực tuyến để xem lại và ôn tập kiến thức đã học trên lớp.

Giúp sinh viên chủ động hơn trong việc học tập, không bị giới hạn bởi thời gian và không gian.

Tạo điều kiện để giảng viên dễ dàng chia sẻ bài giảng, bài học và bài kiểm tra trắc nghiệm đến sinh viên.

Áp dụng công nghệ thông tin vào hoạt động giảng dạy và học tập, phù hợp với xu hướng chuyển đổi số trong giáo dục hiện nay.

1.3. Các chức năng chính của hệ thống

1.3.1. Chức năng dành cho sinh viên

- Đăng ký tài khoản và đăng nhập hệ thống.
- Xem và tham gia các khóa học.
- Học các bài học theo từng khóa.
- Làm bài kiểm tra trắc nghiệm.
- Theo dõi tiến trình học tập và thống kê kết quả học.

1.3.2. Chức năng dành cho giảng viên

- Tạo khóa học mới.
- Quản lý danh sách khóa học đã tạo.
- Thêm bài học cho từng khóa học
- Tạo, chỉnh sửa và xóa quiz cho từng bài học.

1.3.3. Chức năng dành cho quản trị viên

- Quản lý tài khoản người dùng.
- Quản lý hệ thống khóa học.
- Theo dõi và thống kê dữ liệu tổng quát của hệ thống.

1.4. Công nghệ sử dụng

Các công nghệ đã được sử dụng trong quá trình thực hiện đề tài:

- Frontend: ReactJS kết hợp Tailwind CSS để xây dựng giao diện người dùng.
- Backend: Node.js với ExpressJS để xây dựng các API xử lý nghiệp vụ.
- MySQL, được quản lý thông qua phpMyAdmin, dùng để lưu trữ dữ liệu người dùng, khóa học, bài học và các bài kiểm tra trắc nghiệm.
- Xác thực và phân quyền: JSON Web Token (JWT) để xác thực người dùng và phân quyền theo vai trò.
- Giao tiếp dữ liệu: RESTful API giữa Frontend và Backend.

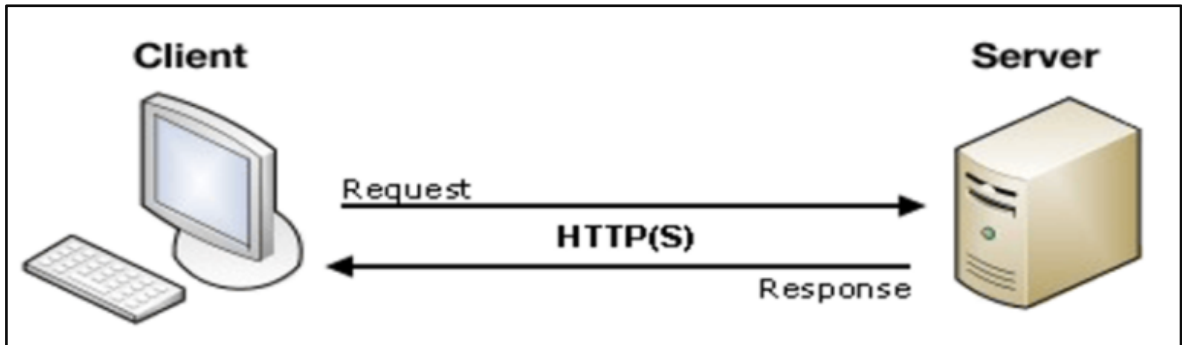
CHƯƠNG 2. NGHIÊN CỨU LÝ THUYẾT

2.1. Kiến trúc hệ thống

2.1.1. Mô hình Client - Server

2.1.1.1. Mô hình tổng thể

Mô hình Client – Server là mô hình phổ biến được sử dụng rộng rãi trong các hệ thống ứng dụng web hiện nay. Theo mô hình này, hệ thống được chia thành hai phần chính là Client và Server.



Hình 2.1. Mô hình Client – Server

Trong mô hình này một máy tính sẽ đóng vai trò là client và máy tính kia đóng vai trò là server. Máy tính client sẽ gửi các yêu cầu (request) đến máy tính server để yêu cầu server thực hiện công việc gì đó. Chẳng hạn khi người dùng duyệt web trên mạng Internet, trình duyệt web sẽ gửi yêu cầu đến web server để web server gửi về trang web tương ứng.

Máy tính server khi nhận được một yêu cầu từ client gửi đến sẽ phân tích yêu cầu để hiểu được client muốn điều gì, để thực hiện đúng yêu cầu của client. Server sẽ gửi kết quả về cho client trong các thông điệp trả lời (reply) [6].

2.1.1.2. Ưu điểm của mô hình Client – Server

- Tập trung hóa: mọi thông tin được quản lý tại một nơi duy nhất, giúp quản trị viên dễ dàng điều hành và xử lý sự cố.
- Bảo mật: dữ liệu được bảo vệ tối đa nhờ kiến trúc tập trung, giúp kiểm soát truy cập và khôi phục file dễ dàng chỉ từ một bản sao lưu duy nhất.
- Khả năng mở rộng: Có thể thêm client hoặc server bất cứ khi nào cần.

2.1.1.3. Nhược điểm của mô hình Client – Server

- Nguy cơ quá tải Server: khi có quá nhiều Client gửi yêu cầu đến cùng một Server quá lớn có thể dẫn đến hiện tượng quá tải (crash) của Server.

- Chi phí cao: thiết lập và bảo trì server tốn kém do yêu cầu tài nguyên lớn.
- Bảo trì liên tục: Server hoạt động không ngừng nghỉ, đòi hỏi phải có nhà quản lý mạng chuyên biệt để tiến hành bảo trì hệ thống đúng mức.
- Sự phụ thuộc vào Server: nếu Server không hoạt động thì các ứng dụng phụ thuộc vào nó cũng không hoạt động được.

2.1.2. Tìm hiểu về RESTful API

2.1.2.1. RESTful API là gì?

RESTful API là những giao diện lập trình ứng dụng tuân thủ nghiêm ngặt các ràng buộc kiến trúc của REST (Representational State Transfer) – một phong cách kiến trúc nhằm thiết lập các tiêu chuẩn chung giữa các hệ thống máy tính trên nền tảng web, giúp chúng giao tiếp với nhau dễ dàng hơn. Mục tiêu cốt lõi của REST là tăng cường hiệu suất, khả năng mở rộng, sự đơn giản và độ tin cậy của hệ thống [7].

2.1.2.2. Cách thức hoạt động của API RESTful

Chức năng cơ bản của API RESTful tương tự như việc duyệt Internet: Client gửi yêu cầu đến Server thông qua API, và Server xử lý rồi trả về phản hồi.

a) Yêu cầu (Request) từ Client

Yêu cầu API RESTful thường được triển khai bằng Giao thức truyền siêu văn bản (HTTP) và bao gồm các thành phần chính sau:

- URL (Endpoint): Xác định tài nguyên cần truy cập.
- Phương thức HTTP: Quy định thao tác với tài nguyên
 - + GET: Lấy dữ liệu
 - + POST: Tạo mới
 - + PUT/PATCH: Cập nhật
 - + DELETE: Xóa
- HTTP Headers: Chứa siêu dữ liệu về định dạng và thông tin trao đổi.
- Data (Body): Dữ liệu gửi kèm trong POST, PUT hoặc PATCH.
- Parameters: Tham số bổ sung như query hoặc cookie.

b) Phản hồi (Response) từ Server

Phản hồi của máy chủ phải chứa các thành phần sau:

- Dòng trạng thái : Chứa mã trạng thái HTTP cho biết kết quả xử lý
 - + 2XX: Thành công (200 OK, 201 Created)
 - + 4XX: Lỗi phía client (400, 404)

+ 5XX: Lỗi phía server (500)

- Message Body: Chứa dữ liệu phản hồi, thường ở dạng JSON hoặc XML.
- Headers: Cung cấp siêu dữ liệu về phản hồi (máy chủ, mã hóa, loại nội dung).

2.2. Công nghệ Frontend

2.2.1. Tìm hiểu về ReactJS

2.2.1.1. Giới thiệu về ReactJS

React được tạo ra lần đầu bởi Jordan Walke, một kỹ sư phần mềm tại Facebook. React được tích hợp vào bảng tin (News Feed) của Facebook vào năm 2011, và sau đó được sử dụng trong Instagram khi nền tảng này được Facebook mua lại vào năm 2012 [1].

React có nhiệm vụ xây dựng giao diện người dùng (UI) bằng cách chia ứng dụng thành các component độc lập, tái sử dụng, giúp code gọn gàng và dễ mở rộng khi ứng dụng phát triển.

2.2.1.2. Kiến trúc và nguyên lý hoạt động của ReactJS

a) Mô hình Component-Based

Mô hình Component-Based là cách thiết kế phần mềm dựa trên các component độc lập, mỗi component đảm nhiệm một chức năng, có thể tái sử dụng và che giấu chi tiết cài đặt. Nhờ giao diện giao tiếp rõ ràng, mô hình này giúp dễ phát triển, bảo trì và mở rộng, và được dùng phổ biến trong các framework hiện đại như ReactJS, VueJS, Angular.

b) Virtual DOM

Virtual DOM là bản sao ảo của DOM thật, được ReactJS sử dụng để quản lý việc cập nhật giao diện. Khi dữ liệu thay đổi, ReactJS tạo Virtual DOM mới, so sánh với bản cũ và chỉ cập nhật những phần thay đổi lên DOM thật. Cách làm này giúp giảm số lần thao tác trực tiếp với DOM, tối ưu hiệu năng và nâng cao trải nghiệm người dùng, đặc biệt trong các ứng dụng lớn.

c) Cơ chế cập nhật giao diện người dùng

ReactJS sử dụng Virtual DOM và thuật toán reconciliation để tối ưu việc cập nhật UI. Khi state/props thay đổi, quá trình cập nhật giao diện diễn ra như sau:

1. Tạo Virtual DOM mới: ReactJS tạo Virtual DOM mới để phản ánh trạng thái UI sau khi state hoặc props thay đổi.

2. So sánh: ReactJS so sánh Virtual DOM mới với Virtual DOM cũ để tìm ra sự khác biệt.

3. Xác định phần tử cần cập nhật: Thuật toán xác định chính xác các phần tử trong DOM thật cần thay đổi.

4. Cập nhật tối ưu: ReactJS gom các thay đổi và chỉ cập nhật những phần tử cần thiết lên DOM thật, giúp cải thiện hiệu suất.

2.2.1.3. JSX trong ReactJS

a) JSX là gì?

Để làm việc hiệu quả với React, cần thêm một yếu tố, đó là JSX. JSX là sự kết hợp giữa JS trong JavaScript và X trong XML. Đây là một phần mở rộng của JavaScript, cho phép chúng ta định nghĩa các React element bằng cú pháp dựa trên thẻ (tag) ngay trong mã JavaScript. Đôi khi JSX bị nhầm lẫn với HTML vì chúng có hình thức tương tự nhau. Tuy nhiên, JSX chỉ đơn giản là một cách khác để tạo ra các React element, giúp lập trình viên không phải viết những lời gọi createElement phức tạp và khó đọc [1].

b) Cách hoạt động của JSX

JSX cho phép nhúng các biểu thức JavaScript thông qua cặp dấu ngoặc nhọn { }. Các biểu thức này có thể bao gồm biến, mảng, đối tượng hoặc hàm, và sẽ được đánh giá tại thời điểm render để tạo ra giao diện động.

Mặc dù có cú pháp giống HTML, JSX không được trình duyệt hiểu trực tiếp. Trước khi được thực thi, mã JSX phải được biên dịch sang JavaScript thuần thông qua công cụ Babel. Quá trình biên dịch này sẽ chuyển các biểu thức JSX thành các lời gọi hàm React.createElement, từ đó React sử dụng để xây dựng và cập nhật cây DOM một cách hiệu quả [1].

2.2.1.4. Component trong ReactJS

a) Khái niệm Component

Component là khối xây dựng của ứng dụng React, thường là một hàm trả về React element. Component giúp chia giao diện thành các phần độc lập, tái sử dụng, nhận dữ liệu qua props và có thể lồng vào nhau để tạo UI phức tạp.

b) Functional Component

Functional Component là component được định nghĩa dưới dạng hàm JavaScript, nhận props và trả về React element. Ban đầu chỉ dùng để hiển thị giao

diện, nhưng từ React 16.8 với Hooks (useState, useEffect), functional component có thể quản lý state và lifecycle, nên được khuyến nghị trong React hiện đại nhờ cú pháp gọn và dễ bảo trì.

c) Props và cơ chế truyền dữ liệu (One-way Data Flow)

Props là các tham số dùng để truyền dữ liệu từ component cha xuống component con, có tính chỉ đọc. ReactJS áp dụng luồng dữ liệu một chiều, trong đó dữ liệu luôn đi từ cha → con thông qua props.

Cơ chế cập nhật:

- Component con không được thay đổi trực tiếp props.
- Khi cần cập nhật dữ liệu, component con gọi callback truyền từ component cha để cha cập nhật state.

Ví dụ:

```
// Component con
function Welcome(props) {
  return (
    <h2>Xin chào, {props.name}</h2>
  );
}
// Component cha
function App() {
  return (
    <div>
      <Welcome name="Nguyễn Văn A" />
      <Welcome name="Trần Thị B" />
    </div>
  );
}
```

Giải thích:

- name là props được truyền từ component cha (App) sang component con.
- Component con không thay đổi được props, chỉ dùng để hiển thị.
- Props giúp component tái sử dụng với dữ liệu khác nhau.

2.2.1.5. State và Hooks trong ReactJS

a) Khái niệm State

State là đối tượng lưu trữ dữ liệu bên trong component, chỉ tồn tại trong phạm vi component đó. Khi state thay đổi, component tự động re-render để cập nhật UI. State mang tính riêng tư và chỉ được thay đổi thông qua cơ chế của React.

Ví dụ, trong nút “Thích”, state có thể dùng để quản lý số lượt thích (likeCount).

b) React Hooks

React Hooks (ra đời từ React 16.8) là một cách tiếp cận mới để quản lý state, side effects (tác dụng phụ) và chia sẻ logic giữa các functional component,.. Trong khi các Class Component truyền thống sử dụng this.state và các lifecycle methods, Functional Component hiện đại sử dụng Hooks để thực hiện các chức năng tương tự.

2.2.1.6. Điều hướng trang trong ReactJS

a) Khái niệm Single Page Application (SPA)

- Khái niệm: Single Page Application (SPA) là ứng dụng web chỉ tải một trang HTML, các thành phần được cập nhật động mà không tải lại toàn bộ trang.

- Mục đích: SPA yêu cầu chuyển trạng thái nhanh và tối ưu hóa trải nghiệm người dùng.

- ReactJS và SPA: ReactJS phù hợp để xây dựng SPA và dùng Virtual DOM để tối ưu cập nhật giao diện, cải thiện hiệu suất.

- Hạn chế mặc định: SPA ReactJS thường dùng Client-side Rendering (CSR), khiến SEO kém do nội dung không sẵn cho công cụ tìm kiếm.

b) Điều hướng trang và React Router

Nhu cầu Routing: ReactJS không cung cấp sẵn giải pháp routing, nên cần dùng thư viện ngoài để quản lý điều hướng và nhiều trang trong ứng dụng.

React Router: là thư viện phổ biến, mạnh mẽ để xử lý điều hướng động (dynamic routing) trong SPA.

Vai trò của Routing Components: giúp điều hướng, quản lý trạng thái và chia ứng dụng thành các module..

2.2.1.7. Ưu điểm và nhược điểm của ReactJS

a) Ưu điểm

Hiệu suất cao: React dùng Virtual DOM để chỉ cập nhật phần thay đổi, giúp tăng tốc ứng dụng.

Kiến trúc Component-Based: chia ứng dụng thành các component độc lập, dễ tái sử dụng, mở rộng và bảo trì.

JSX trực quan: kết hợp HTML và JavaScript, dễ đọc và debug.

Hệ sinh thái lớn: được Meta hỗ trợ, cộng đồng và tài nguyên phong phú.

Đa nền tảng: có thể phát triển ứng dụng di động native với React Native.

b) Nhược điểm

Chỉ là View Layer: React tập trung vào giao diện, cần tích hợp thêm thư viện như React Router, Redux cho routing và quản lý state.

SEO chưa tối ưu: SPA dùng Client-Side Rendering nên cần kết hợp SSR (như Next.js) để cải thiện SEO.

Thay đổi nhanh: Hệ sinh thái React phát triển liên tục, gây khó khăn trong việc theo kịp và bảo trì lâu dài.

2.2.2. Tìm hiểu về Tailwind CSS**2.2.2.1. Giới thiệu về Tailwind CSS**

Tailwind CSS là một framework CSS theo hướng utility-first, được thiết kế để xây dựng giao diện người dùng một cách nhanh chóng và linh hoạt. Framework này cung cấp các lớp tiện ích cấp thấp (low-level utility classes), cho phép lập trình viên tạo ra các thiết kế tùy chỉnh hoàn toàn trực tiếp trong mã HTML, mà không cần viết CSS riêng biệt [2].

2.2.2.2. Nguyên lý hoạt động của Tailwind CSS

Tailwind CSS hoạt động dựa trên nguyên lý utility-first. Thay vì tạo các lớp CSS mang ý nghĩa ngữ nghĩa như `.button`, `.card` hay `.header`, Tailwind cung cấp các lớp tiện ích đơn lẻ, mỗi lớp đảm nhiệm một thuộc tính CSS cụ thể, chẳng hạn như khoảng cách, màu sắc, kích thước chữ hoặc căn chỉnh.

Tailwind CSS sử dụng một tệp cấu hình (`tailwind.config.js`) để tùy chỉnh các giá trị như màu sắc, font chữ, khoảng cách và breakpoint. Trong quá trình build, Tailwind sẽ tạo ra CSS dựa trên các class thực sự được sử dụng trong dự án, giúp giảm kích thước tệp CSS cuối cùng [2].

Ví dụ: Xây dựng một nút (button) màu xanh, chữ trắng, có bo góc

Bảng 2.1. Ví dụ về Tailwind CSS

Thuộc tính	Lớp tiện ích	Ý nghĩa
<code>background-color: #3b82f6;</code>	<code>bg-blue-500</code>	Màu nền xanh dương
<code>color: white;</code>	<code>text-white</code>	Màu chữ trắng
<code>padding: 0.5rem 1rem;</code>	<code>px-4 py-2</code>	Padding ngang và dọc
<code>border-radius: 0.25rem;</code>	<code>rounded</code>	Bo góc

2.2.2.3. Kết hợp Tailwind CSS với ReactJS

Trong ReactJS, Tailwind CSS được sử dụng trực tiếp trong JSX thông qua `className`, giúp mỗi component quản lý cả logic và giao diện. Cách này cho phép định nghĩa UI ngay trong component, giảm phụ thuộc giữa các file và hạn chế xung đột CSS so với cách viết CSS truyền thống.

Ví dụ :

```
function Button() {  
  return (  
    <button className="px-4 py-2 bg-blue-500 text-white  
rounded-lg hover:bg-blue-600">  
      Đăng nhập  
    </button>  
  );  
}
```

2.2.2.4. Ưu điểm và nhược điểm của Tailwind CSS

a) Ưu điểm

Phát triển giao diện nhanh: Các lớp utility cho phép xây dựng giao diện trực tiếp trong HTML mà không cần viết nhiều CSS riêng.

Dễ tùy biến: Thông qua tệp cấu hình, Tailwind cho phép tùy chỉnh màu sắc, font chữ, khoảng cách và breakpoint theo yêu cầu dự án.

Dễ bảo trì: Mỗi class chỉ đảm nhiệm một chức năng cụ thể, giúp dễ đọc và dễ chỉnh sửa giao diện.

Tối ưu hiệu năng: CSS được sinh ra dựa trên các class thực sự được sử dụng, làm giảm kích thước tệp CSS cuối cùng.

b) Nhược điểm

HTML dài và nhiều class: Việc sử dụng nhiều lớp utility có thể làm mã HTML trở nên dài và khó đọc.

Phụ thuộc vào build tool: Tailwind CSS yêu cầu quá trình build để tạo CSS hoàn chỉnh, không thuận tiện cho các dự án rất nhỏ.

2.2.3. Tìm hiểu về Vite

2.2.3.1. Giới thiệu về Vite

Vite là một công cụ build và phát triển ứng dụng web hiện đại, được tạo ra bởi Evan You, người đã phát triển Vue.js. Mục tiêu của Vite là giải quyết các vấn đề về

tốc độ, hiệu suất và trải nghiệm lập trình khi phát triển các ứng dụng web React, Vue và nhiều framework khác.

2.2.3.2. Vai trò của Vite trong dự án ReactJS

a) Khởi tạo nhanh dự án ReactJS

Hỗ trợ đa framework: Vite hỗ trợ sẵn các framework như React, Vue, Svelte, Preact, Lit và SolidJS.

Zero-config: Thiết kế theo triết lý không cần cấu hình phức tạp, giúp khởi tạo dự án nhanh chóng.

Khởi tạo dự án: Tạo dự án React dễ dàng bằng lệnh `npm create vite@latest my-react-app -- --template react`.

b) Cung cấp môi trường phát triển

Vite đóng vai trò là một môi trường phát triển cục bộ (dev server) được tối ưu hóa cao, cung cấp tốc độ phản hồi vượt trội so với các công cụ truyền thống:

- Tốc độ khởi động siêu nhanh: Vite sử dụng Native ES Modules (ESM), chỉ tải module khi cần thay vì đóng gói toàn bộ mã nguồn, giúp khởi động dự án trong vài trăm mili giây.

- Hot Module Replacement (HMR) tức thì: Cho phép cập nhật thay đổi mã nguồn ngay lập tức mà không cần tải lại trang, chỉ reload module bị thay đổi, giúp quá trình phát triển mượt mà hơn.

- Chạy máy chủ phát triển: Máy chủ của Vite được khởi động đơn giản bằng lệnh `npm run dev` hoặc `vite serve`.

c) Build mã nguồn ReactJS thành sản phẩm hoàn chỉnh để triển khai

Khi lập trình viên React cần đưa ứng dụng lên môi trường production, Vite đảm nhiệm vai trò đóng gói và tối ưu hóa mã nguồn:

- Đóng gói production: Vite đóng gói toàn bộ HTML, CSS, JS bằng Rollup.
- Tối ưu hiệu suất: Áp dụng tree-shaking và code-splitting để giảm dung lượng và tăng tốc tải.

- Sử dụng esbuild: Tạo bundle được thu gọn và tối ưu với tốc độ rất nhanh.

- Thực hiện build: Dùng lệnh `npm run build`, các file production được tạo trong thư mục `dist/`.

2.3. Công nghệ Backend

2.3.1. Tìm hiểu về Node.js

2.3.1.1. Giới thiệu về Node.js

Node.js là một môi trường thực thi (runtime environment) mã nguồn mở, đa nền tảng, được sử dụng để phát triển các ứng dụng web phía máy chủ (server-side). Các ứng dụng Node.js được viết bằng ngôn ngữ JavaScript và có thể chạy trên nhiều hệ điều hành khác nhau [5].

2.3.1.2. Cơ chế hoạt động và kiến trúc độc đáo của Node.js

a) Kiến trúc bất đồng bộ và Non-blocking I/O

Tất cả các API của Node.js đều có đặc điểm là không đồng bộ (Asynchronous) và không bị chặn (Non-blocking I/O). Non-blocking I/O nghĩa là Node.js không ngồi chờ các tác vụ tốn thời gian như đọc file, truy vấn cơ sở dữ liệu hay giao tiếp mạng hoàn thành. Thay vào đó, nó giao việc và tiếp tục xử lý các yêu cầu khác.

b) Event Loop (Vòng lặp sự kiện)

Đây là trái tim của Node.js. Event Loop hoạt động như một người quản lý, liên tục kiểm tra xem có yêu cầu mới nào đến hoặc có tác vụ nào vừa hoàn thành không. Khi một thao tác I/O hoàn tất, một sự kiện sẽ được phát đi và Event Loop sẽ kích hoạt hàm gọi lại (callback) tương ứng để xử lý kết quả.

c) Bản chất đơn luồng (Single-threaded)

Node.js hoạt động trên một luồng chính duy nhất cho logic ứng dụng của người dùng. Chính kiến trúc bất đồng bộ đã biến mô hình đơn luồng này thành ưu điểm, giúp Node.js nhẹ, tiết kiệm tài nguyên và có thể xử lý hàng ngàn kết nối đồng thời mà không bị tắc nghẽn.

2.3.1.3. Hệ sinh thái và lợi ích

Sử dụng một ngôn ngữ duy nhất: Node.js cho phép lập trình viên sử dụng JavaScript cho cả phần Frontend và backend. Điều này giúp đồng bộ hóa đội ngũ phát triển và giảm thời gian học công nghệ mới.

Hiệu năng cao: nhờ kiến trúc non-blocking I/O và V8 engine, Node.js xử lý các yêu cầu I/O cực kỳ nhanh, rất phù hợp cho các ứng dụng cần xử lý nhiều kết nối cùng lúc.

Hệ sinh thái NPM khổng lồ: NPM (Node Package Manager) là trình quản lý gói mặc định của Node.js. Đây là một kho lưu trữ trực tuyến chứa hàng triệu gói (thư viện) mã nguồn mở có thể tái sử dụng, giúp tăng tốc độ phát triển dự án.

2.3.1.4. Ứng dụng phổ biến

Xây dựng Backend và REST APIs: đây là ứng dụng phổ biến nhất, tận dụng khả năng xử lý hàng ngàn kết nối đồng thời và tốc độ phản hồi nhanh.

Ứng dụng thời gian thực: Lý tưởng cho các ứng dụng yêu cầu tương tác ngay lập tức như ứng dụng chat, công cụ cộng tác trực tuyến, game online.

Kiến trúc Microservices: Node.js nhẹ và khởi động nhanh, phù hợp để xây dựng các dịch vụ nhỏ, độc lập.

Streaming Data: Hỗ trợ xử lý dữ liệu dạng luồng (ví dụ: video, audio) ngay khi chúng được truyền tải.

2.3.2. Tìm hiểu về ExpressJS

2.3.2.1. Giới thiệu về ExpressJS

ExpressJS là một framework bên thứ ba dành cho Node. Nó là một trong những thư viện npm phổ biến nhất. Nó giúp việc thực hiện các công việc như tạo một máy chủ web hoặc một API HTTP trở nên thực sự dễ dàng. Vì ExpressJS là một framework phát triển ứng dụng web trên Node.js, các lập trình viên có thể sử dụng mã đã có sẵn để xây dựng các ứng dụng web đơn trang (SPA), đa trang hoặc kết hợp cả hai. Ngoài ra, ExpressJS còn cung cấp một kiến trúc MVC (Model-View-Controller) hữu ích để tổ chức các ứng dụng web phía máy chủ [5].

2.3.2.2. Vai trò và ứng dụng của ExpressJS

Mục đích chính của ExpressJS là làm cho việc phát triển các ứng dụng web và API trở nên đơn giản, hiệu quả và dễ bảo trì hơn.

- Xây dựng API RESTful: Hỗ trợ đầy đủ các phương thức HTTP (GET, POST, PUT, DELETE) cho SPA và ứng dụng di động.

- Hỗ trợ ứng dụng Web: Cung cấp API cho Frontend (React, Angular, Vue) hoặc xây dựng web truyền thống với template engine (Pug, EJS).

- Microservices: Kiến trúc nhẹ, tối giản, phù hợp phát triển microservice dễ mở rộng và bảo trì.

2.3.2.3. Các công cụ cốt lõi của ExpressJS

a) Định tuyến (Routing)

Routing (Định tuyến) trong Express.js là cách bạn thiết lập các quy tắc để xử lý các yêu cầu HTTP từ người dùng đến đúng nơi trong ứng dụng của bạn. Mỗi URL sẽ được gán với một route (định tuyến), và mỗi route có thể xử lý các loại yêu cầu khác nhau như GET, POST, PUT, DELETE.

b) Xử lý Request/Response (req và res)

Đối tượng Request (req) : Đại diện cho yêu cầu HTTP đến từ máy khách, chứa thông tin về yêu cầu như các tham số URL, tham số truy vấn, dữ liệu gửi trong yêu cầu POST/PUT, và phương thức HTTP.

Đối tượng Response (res): Đại diện cho phản hồi HTTP gửi lại cho máy khách, chứa các phương thức để gửi phản hồi như res.send(), res.json(), res.status() (thiết lập mã trạng thái HTTP), và res.render() (render view template).

c) Quản lý Middleware

Express middleware là thứ cho phép bổ sung vào các chức năng hiện có của Express. Có thể coi nó như một tiện ích bổ sung của bên thứ ba. Nếu Express không làm được việc đó, có thể tìm một middleware có sẵn hoặc tự viết middleware cho riêng mình.

Để đăng ký một middleware mới cần gọi app.use. Đây là phương thức được dùng để cung cấp middleware. Nó nhận một hàm làm đối số duy nhất, và hàm này được gọi với ba đối số: req, res, và next. Đối số next rất quan trọng. Nó tồn tại để bạn thông báo cho Express khi hàm middleware của bạn hoàn tất [5].

2.3.2.4. RESTful API trong ExpressJS

ExpressJS thường được dùng để xây dựng RESTful API:

Bảng 2.2. RESTful API trong ExpressJS

Phương thức	Chức năng
GET	Lấy dữ liệu
POST	Thêm dữ liệu
PUT	Cập nhật dữ liệu
DELETE	Xóa dữ liệu

Ví dụ:

```
app.get("/api/courses", (req, res) => {  
  res.json({ message: "Danh sách khóa học" });});
```

2.3.2.5. Giao tiếp giữa Frontend và Backend thông qua API RESTful

Đây là mô hình ứng dụng phổ biến và là thế mạnh của ExpressJS, trong đó Backend đóng vai trò cung cấp dữ liệu cho Frontend (SPA như React, Angular, Vue hoặc ứng dụng di động).

Cơ chế giao tiếp:

- Định tuyến (Routing): ExpressJS định nghĩa các endpoint để xử lý yêu cầu từ Frontend.
- Phương thức HTTP: Frontend gửi các yêu cầu như GET, POST, PUT, DELETE để truy xuất và quản lý dữ liệu.
- Xử lý yêu cầu (req): Backend nhận request qua đối tượng req, truy cập dữ liệu từ params, query, body (cần middleware express.json()).
- Gửi phản hồi (res): ExpressJS trả dữ liệu về Frontend bằng JSON qua res.json() và thiết lập mã trạng thái bằng res.status().

2.4. Công nghệ cơ sở dữ liệu

2.4.1. Tìm hiểu về MySQL

2.4.1.1. Giới thiệu về MySQL

MySQL là một hệ quản trị cơ sở dữ liệu (DBMS) mã nguồn mở được phát triển và duy trì bởi Oracle Corporation. Nó được đặt tên theo con gái của một trong những người sáng lập, Michael Widenius. MySQL nổi tiếng với tốc độ, độ tin cậy và tính linh hoạt cao, khiến nó trở thành lựa chọn hàng đầu cho nhiều ứng dụng web và doanh nghiệp lớn.

2.4.1.2. Các khái niệm cơ bản trong MySQL

a) Cơ sở dữ liệu (Database)

Trong MySQL, cơ sở dữ liệu (Database) là tập hợp dữ liệu có tổ chức, dùng để lưu trữ và quản lý thông tin của ứng dụng. Mỗi cơ sở dữ liệu có thể bao gồm nhiều bảng (Table), trong đó bảng là nơi lưu trữ dữ liệu thực tế dưới dạng các hàng (Row) và cột (Column)

b) Kiểu dữ liệu (Data type)

Kiểu dữ liệu xác định loại giá trị mà một cột có thể chứa. Việc định nghĩa đúng kiểu dữ liệu giúp MySQL trích xuất, sửa đổi và cấu trúc dữ liệu một cách chính xác. Các loại kiểu dữ liệu phổ biến bao gồm:

Bảng 2.3. Các kiểu dữ liệu phổ biến trong MySQL

Kiểu dữ liệu	Mô tả
INT	Lưu số nguyên, thường dùng làm khóa chính
BIGINT	Lưu số nguyên lớn
VARCHAR(n)	Lưu chuỗi ký tự có độ dài thay đổi
TEXT	Lưu chuỗi văn bản dài
DATE	Lưu ngày (YYYY-MM-DD)
DATETIME	Lưu ngày và giờ
TIMESTAMP	Lưu thời gian tạo/cập nhật
BOOLEAN	Lưu giá trị đúng/sai
ENUM	Lưu giá trị cố định trong danh sách

c) Truy vấn (Query)

Truy vấn là một câu lệnh SQL (Structured Query Language) được gửi đến máy chủ MySQL để tương tác với dữ liệu. Đây là công cụ chính để người dùng thực hiện các thao tác sau:

- Truy xuất: Sử dụng lệnh SELECT để lấy dữ liệu.
- Thêm: Sử dụng lệnh INSERT INTO để tạo bản ghi mới.
- Sửa đổi: Sử dụng lệnh UPDATE để cập nhật dữ liệu hiện có.
- Xóa: Sử dụng lệnh DELETE để loại bỏ dữ liệu.

2.4.1.3. Ưu điểm và nhược điểm của MySQL**a) Ưu điểm**

Miễn phí và mã nguồn mở: MySQL là phần mềm mã nguồn mở, cho phép bạn sử dụng, sửa đổi và phân phối nó miễn phí.

Tốc độ và hiệu suất cao: MySQL được thiết kế để có tốc độ và hiệu suất cao, đặc biệt là trong các ứng dụng web.

Dễ sử dụng và học: MySQL có cú pháp SQL đơn giản và dễ hiểu.

Khả năng mở rộng: MySQL có thể mở rộng để đáp ứng nhu cầu ngày càng tăng của ứng dụng.

Hỗ trợ nhiều ngôn ngữ lập trình: MySQL hỗ trợ nhiều ngôn ngữ lập trình.

Bảo mật cao: MySQL cung cấp nhiều tính năng bảo mật để bảo vệ dữ liệu của bạn, bao gồm kiểm soát truy cập, mã hóa và kiểm toán.

b) Nhược điểm

Hạn chế về tính năng so với các hệ quản trị CSDL như SQL Server và Oracle.

Khả năng xử lý các truy vấn phức tạp: MySQL có thể gặp khó khăn trong việc xử lý các truy vấn phức tạp hoặc các truy vấn liên quan đến nhiều bảng.

Hạn chế về ACID compliance: ỗ trợ ACID nhưng chưa đáp ứng hoàn toàn các yêu cầu nghiêm ngặt trong một số trường hợp.

Phân mảnh dữ liệu: Theo thời gian, dữ liệu trong MySQL có thể bị phân mảnh, điều này có thể làm giảm hiệu suất.

Quản lý tài nguyên: Trong một số trường hợp, MySQL có thể tiêu thụ nhiều tài nguyên hệ thống.

c) Kết nối MySQL với Node.js (ExpressJS)

Backend sử dụng MySQL thông qua thư viện mysql2 hoặc mysql.

```
const mysql = require("mysql2");
const connection = mysql.createConnection({
  host: "localhost",
  user: "root",
  password: "",
  database: "my_database"});
connection.connect();
```

Backend ExpressJS sẽ gọi database để xử lý dữ liệu và trả kết quả về Frontend.

2.4.2. Tìm hiểu về XAMPP và phpMyAdmin

2.4.2.1. Giới thiệu về XAMPP

XAMPP là phần mềm mã nguồn mở, miễn phí dùng để tạo web server trên máy cá nhân (localhost). Nó cho phép phát triển và chạy website/ứng dụng web PHP–MySQL dễ dàng, ngay cả với người không chuyên về máy chủ.

Các thành phần chính của của XAMPP:

- Apache: là một web server phổ biến, được sử dụng để xử lý các yêu cầu từ người dùng và trả về các trang web.
- PHP: là một ngôn ngữ lập trình kịch bản phía máy chủ, được sử dụng để tạo các trang web động.
- MySQL: là một hệ quản trị cơ sở dữ liệu quan hệ (RDBMS), được sử dụng để lưu trữ dữ liệu của website.
- FTP Server: là một máy chủ FTP, được sử dụng để truyền tải tệp giữa máy tính của bạn và máy tính khác.
- Mail Server: là một máy chủ email, được sử dụng để gửi và nhận email.
- phpMyAdmin: là một công cụ quản lý cơ sở dữ liệu MySQL, được sử dụng để quản lý dữ liệu của website.

2.4.2.2. Ứng dụng của XAMPP

Phát triển ứng dụng web: cung cấp môi trường sẵn sàng để xây dựng, thử nghiệm và triển khai ứng dụng web nhanh chóng.

Học lập trình web: Phù hợp cho người mới bắt đầu thực hành PHP, MySQL mà không cần cấu hình phức tạp.

Thử nghiệm công nghệ mới: dễ dàng test các công nghệ mà không ảnh hưởng đến dự án đang chạy.

Quản lý cơ sở dữ liệu: hỗ trợ quản lý MySQL thông qua phpMyAdmin.

2.4.2.3. Giới thiệu về và phpMyAdmin

phpMyAdmin là công cụ mã nguồn mở viết bằng PHP dùng để quản lý MySQL/MariaDB qua giao diện web. Công cụ này cho phép tạo, chỉnh sửa, xóa CSDL, bảng, dữ liệu, truy vấn SQL và phân quyền người dùng một cách dễ dàng, nhờ giao diện thân thiện và tính năng mạnh mẽ.

2.4.2.4. Các tính năng nổi bật của phpMyAdmin

Quản lý người dùng: thêm, sửa, xóa tài khoản và phân quyền chi tiết.

Quản trị cơ sở dữ liệu: tạo, sửa, xóa cơ sở dữ liệu, bảng và dữ liệu.

Truy vấn MySQL: thực hiện và giám sát câu lệnh SQL trực tiếp, theo dõi quá trình xử lý.

Xuất/nhập dữ liệu: hỗ trợ nhiều định dạng như SQL, CSV, XML.

Sao lưu và khôi phục: backup thủ công và phục hồi dữ liệu dễ dàng.

2.5. Xác thực và phân quyền người dùng

2.5.1. Xác thực người dùng (Authentication)

Xác thực (Authentication) được máy chủ sử dụng khi máy chủ cần biết chính xác ai đang truy cập vào thông tin hoặc trang web của mình. Ngược lại, xác thực được khách hàng (client) sử dụng khi họ cần biết chắc chắn rằng máy chủ đó chính là hệ thống mà nó tự nhận. Trong quá trình xác thực, người dùng hoặc máy tính phải chứng minh danh tính của mình với máy chủ hoặc khách hàng. Xác thực là quá trình xác minh danh tính của người dùng (qua mật khẩu, sinh trắc học) hoặc của máy chủ (qua chứng chỉ số) để đảm bảo các bên đúng là người mà họ tự nhận, nhưng nó không bao gồm việc phân quyền hay giới hạn các hành động mà đối tượng đó được phép thực hiện [4].

2.5.1.1. Mã hóa mật khẩu bằng bcrypt

a) Khái niệm bcrypt

bcrypt là một thuật toán dùng để mã hóa (băm) mật khẩu người dùng. Thuật toán này chuyển mật khẩu ban đầu thành một chuỗi ký tự khác, không thể khôi phục lại mật khẩu gốc. Nhờ đặc điểm này, ngay cả khi dữ liệu bị đánh cắp, kẻ tấn công cũng không thể biết được mật khẩu thật của người dùng.

b) Mục đích sử dụng

Mật khẩu là dữ liệu nhạy cảm cần được bảo vệ; việc lưu trực tiếp trong CSDL có thể gây rò rỉ khi hệ thống bị tấn công. Vì vậy, cần mã hóa mật khẩu trước khi lưu trữ, trong đó bcrypt là giải pháp phổ biến và hiệu quả để đảm bảo an toàn thông tin.

c) Quy trình sử dụng trong hệ thống

bcrypt được sử dụng trong hai giai đoạn chính của hệ thống. Khi người dùng đăng ký tài khoản, mật khẩu được mã hóa bằng bcrypt và chỉ giá trị đã mã hóa được lưu trong cơ sở dữ liệu. Khi người dùng đăng nhập, mật khẩu nhập vào sẽ được so sánh với mật khẩu đã mã hóa để xác thực người dùng.

d) Vai trò trong bảo mật hệ thống

Việc áp dụng bcrypt giúp nâng cao mức độ an toàn cho hệ thống xác thực người dùng, giảm nguy cơ rò rỉ mật khẩu và đảm bảo tính bảo mật cho dữ liệu người dùng trong quá trình vận hành hệ thống.

bcrypt được sử dụng để bảo mật mật khẩu trong quá trình đăng ký và đăng nhập, trong khi JWT được sử dụng để quản lý phiên đăng nhập và phân quyền người dùng sau khi xác thực thành công.

2.5.1.2. Xác thực người dùng bằng JSON Web Token (JWT)

a) Giới thiệu JSON Web Token

JSON Web Token (JWT) là một chuẩn mở (open standard) do IETF ban hành, được định nghĩa trong RFC 7519, dùng để truyền tải thông tin dưới dạng JSON một cách an toàn và nhỏ gọn giữa các thành phần trong hệ thống phần mềm. JWT thường được sử dụng trong các hệ thống xác thực (authentication) và phân quyền (authorization), đặc biệt phổ biến trong RESTful API, microservices và các ứng dụng web hiện đại.

Về bản chất, JWT là một chuỗi ký tự được mã hóa theo chuẩn Base64URL, giúp nó có thể truyền qua HTTP header, cookie hoặc URL mà không gây lỗi định dạng. JWT có khả năng tự xác minh (stateless), nghĩa là máy chủ có thể kiểm tra tính hợp lệ của token mà không cần truy vấn cơ sở dữ liệu hay máy chủ xác thực trung tâm [3].

b) Cấu trúc của JSON Web Token

JWT là một chuỗi được tạo bởi ba thành phần, ngăn cách nhau bởi dấu chấm (.) và được tuần tự hoá (serialize) bằng base64. Nếu xét định dạng tuần tự đơn giản nhất thì JWT sẽ có dạng: xxxxx.yyyyy.zzzzz

Ví dụ:

```
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJleHAiOjEzODY4OTkxMzEsIm1zcyI6ImMzg2ODk4OTUxfQ.uKqU9dTB6gKwG6jQCuXYAiMNdFNrW98H
```

c) Ba thành phần của JSON Web Token

JWT có ba thành phần chính: phần đầu (header), phần dữ liệu (payload) và chữ ký (signature). Khi chúng ta tạo một token, chúng ta truyền phần đầu và phần dữ liệu, sau đó token sẽ tạo ra chữ ký.

Header: chứa thông tin token như thuật toán ký, loại token và tùy chọn khóa.

Payload: chứa dữ liệu như thông tin người dùng (ID, vai trò...).

Signature: được tạo bằng cách ký hoặc mã hóa header và payload với khóa bí mật/ khóa riêng, dùng để xác minh tính toàn vẹn của token.

d) Cách thức hoạt động của JSON Web Token

Khi người dùng đăng nhập thành công, hệ thống sẽ tạo một JSON Web Token (JWT) và gửi về phía client. Token này đại diện cho phiên xác thực và sẽ được client gửi kèm theo mỗi request tiếp theo thông qua header Authorization dưới dạng:

Authorization: Bearer <token>

Mỗi lần nhận request, server sẽ kiểm tra tính hợp lệ của JWT để quyết định có cho phép truy cập vào các tài nguyên được bảo vệ hay không. Do JWT chứa sẵn thông tin người dùng, một số truy vấn cơ sở dữ liệu có thể lược bỏ, giúp tăng hiệu suất.

2.5.1.3. Tìm hiểu về Access Token

a) Khái niệm Access Token

Access Token là một loại JWT được cấp cho người dùng sau khi đăng nhập thành công. Token này có thời gian sống ngắn và được sử dụng để xác thực các yêu cầu gửi từ Frontend đến Backend.

b) Hạn chế của Access Token

Access Token mang lại nhiều lợi ích nhưng thời gian sống quá dài có thể gây rủi ro bảo mật, còn quá ngắn sẽ làm giảm trải nghiệm người dùng. Vì vậy, các hệ thống hiện đại thường kết hợp Access Token với Refresh Token để cân bằng giữa bảo mật và tiện lợi.

2.5.1.4. Tìm hiểu về Refresh Token

a) Khái niệm

Refresh Token là token có thời gian sống dài, dùng để cấp lại Access Token mới khi Access Token hết hạn. Nó được lưu trữ an toàn (ví dụ cookie httpOnly) và không dùng trực tiếp để truy cập tài nguyên, chỉ phục vụ việc làm mới Access Token.

b) Cơ chế hoạt động của Refresh Token

1. Người dùng đăng nhập vào hệ thống bằng tài khoản hợp lệ.
2. Backend xác thực thông tin đăng nhập và cấp một Access Token và một Refresh Token.
3. Frontend sử dụng Access Token để gửi kèm trong các request API.
4. Khi Access Token hết hạn, Frontend gửi Refresh Token lên backend để yêu cầu cấp lại Access Token mới.
5. Backend kiểm tra tính hợp lệ của Refresh Token và cấp Access Token mới cho người dùng mà không cần đăng nhập lại.

2.5.2. Phân quyền người dùng (Authorization)

2.5.2.1. Phân quyền người dùng là gì?

Phân quyền (Authorization) là một quá trình mà qua đó máy chủ xác định xem khách hàng (client) có quyền sử dụng tài nguyên hoặc truy cập vào một tệp tin hay không. Phân quyền thường đi đôi với xác thực để máy chủ có khái niệm (nắm bắt được) danh tính của khách hàng đang gửi yêu cầu truy cập là ai. Loại xác thực cần thiết để thực hiện phân quyền có thể khác nhau; trong một số trường hợp có thể yêu cầu mật khẩu nhưng những trường hợp khác thì không. Trong một số trường hợp, không có sự phân quyền; bất kỳ người dùng nào cũng có thể sử dụng tài nguyên hoặc truy cập tệp tin đơn giản bằng cách yêu cầu nó [4].

2.5.2.2. Phân quyền người dùng theo vai trò (RBAC)

Role-Based Access Control (RBAC), hay kiểm soát truy cập dựa trên vai trò, là một trong những mô hình phân quyền phổ biến và được sử dụng rộng rãi nhất. Trong RBAC, quyền hạn không được gán trực tiếp cho từng người dùng mà được gán cho các vai trò (roles). Sau đó, người dùng sẽ được gán một hoặc nhiều vai trò.

Ví dụ: phân quyền theo vai trò trong hệ thống TVULEARN: Trong hệ thống TVULEARN, thay vì gán từng quyền chi tiết như “xem khóa học”, “tạo bài học”, “tạo quiz”, “quản lý người dùng” cho từng tài khoản riêng lẻ, hệ thống áp dụng mô hình phân quyền theo vai trò (Role-Based Access Control – RBAC).

Cụ thể, hệ thống định nghĩa các vai trò sau:

- Sinh viên: có quyền xem và tham gia các khóa học, học bài học và làm các bài kiểm tra trắc nghiệm.
- Giảng viên: có quyền tạo và quản lý khóa học, thêm bài học cho từng khóa học, cũng như tạo, chỉnh sửa và xóa các câu hỏi trắc nghiệm.
- Quản trị viên: có quyền quản lý tài khoản người dùng, quản lý toàn bộ khóa học và theo dõi thống kê tổng quát của hệ thống.

CHƯƠNG 3. HIỆN THỰC HÓA NGHIÊN CỨU

3.1. Mô tả bài toán

Bài toán đặt ra là xây dựng một hệ thống website hỗ trợ học tập cho sinh viên ngành Công nghệ Thông tin Trường Đại học Trà Vinh, nhằm đáp ứng nhu cầu học tập, ôn tập và đánh giá kiến thức trực tuyến một cách hiệu quả.

Hệ thống cần cung cấp các chức năng chính cho sinh viên như đăng ký và đăng nhập tài khoản, xem và tham gia các khóa học, học bài học theo từng môn, thực hiện các bài kiểm tra trắc nghiệm và theo dõi tiến trình học tập. Thông qua các chức năng này, sinh viên có thể ôn tập kiến thức đã học trên lớp mọi lúc, mọi nơi.

Bên cạnh đó, hệ thống phải hỗ trợ giảng viên trong việc tạo và quản lý khóa học, thêm bài học, xây dựng các câu hỏi trắc nghiệm và quản lý nội dung giảng dạy. Điều này giúp giảng viên dễ dàng tổ chức và cập nhật tài liệu học tập cho sinh viên.

Ngoài ra, hệ thống cần có chức năng dành cho quản trị viên nhằm quản lý tài khoản người dùng, quản lý dữ liệu khóa học và theo dõi các thông tin thống kê tổng quát của hệ thống. Việc phân quyền rõ ràng giữa các vai trò sinh viên, giảng viên và quản trị viên giúp hệ thống hoạt động an toàn, hiệu quả và đúng mục đích sử dụng.

3.2. Phân tích thiết kế hệ thống

3.2.1. Đặc tả yêu cầu hệ thống

3.2.1.1. Yêu cầu chức năng

a) Chức năng dành cho sinh viên

Đăng ký và đăng nhập hệ thống: sinh viên có thể tạo tài khoản và đăng nhập vào hệ thống để sử dụng các chức năng học tập.

Xem và tham gia khóa học: sau khi đăng nhập, sinh viên có thể xem danh sách các khóa học hiện có và đăng ký tham gia những khóa học phù hợp.

Làm bài kiểm tra trắc nghiệm: mỗi bài học có thể đi kèm bài kiểm tra trắc nghiệm giúp sinh viên ôn tập và kiểm tra lại kiến thức đã học.

Xem kết quả và thống kê học tập: sau khi hoàn thành bài kiểm tra, sinh viên có thể xem kết quả đạt được, điểm số và theo dõi tiến trình học tập của bản thân.

b) Chức năng dành cho giảng viên

Quản lý khóa học: giảng viên có thể tạo mới, chỉnh sửa và quản lý các khóa học do mình phụ trách.

Quản lý bài học: giảng viên có thể thêm, chỉnh sửa hoặc xóa bài học trong từng khóa học, bao gồm nội dung bài giảng và tài liệu học tập.

Quản lý bài kiểm tra trắc nghiệm: giảng viên có thể tạo, chỉnh sửa và xóa các câu hỏi trắc nghiệm cho từng bài học nhằm hỗ trợ sinh viên ôn tập kiến thức.

Theo dõi nội dung giảng dạy: giảng viên có thể xem danh sách các bài học và quiz đã tạo để đảm bảo nội dung được xây dựng đầy đủ và hợp lý.

c) Chức năng dành cho quản trị viên

Quản lý tài khoản người dùng: quản trị viên có thể quản lý thông tin tài khoản của sinh viên và giảng viên trong hệ thống.

Quản lý hệ thống khóa học: quản trị viên có quyền quản lý dữ liệu tổng thể của các khóa học nhằm đảm bảo hệ thống hoạt động ổn định.

Thông kê và giám sát hệ thống: quản trị viên có thể theo dõi số lượng người dùng, khóa học và các dữ liệu tổng quan để đánh giá hoạt động của hệ thống.

3.2.1.2. Yêu cầu phi chức năng

Bảo mật: Hệ thống sử dụng cơ chế xác thực và phân quyền bằng JSON Web Token (JWT) nhằm đảm bảo an toàn thông tin người dùng.

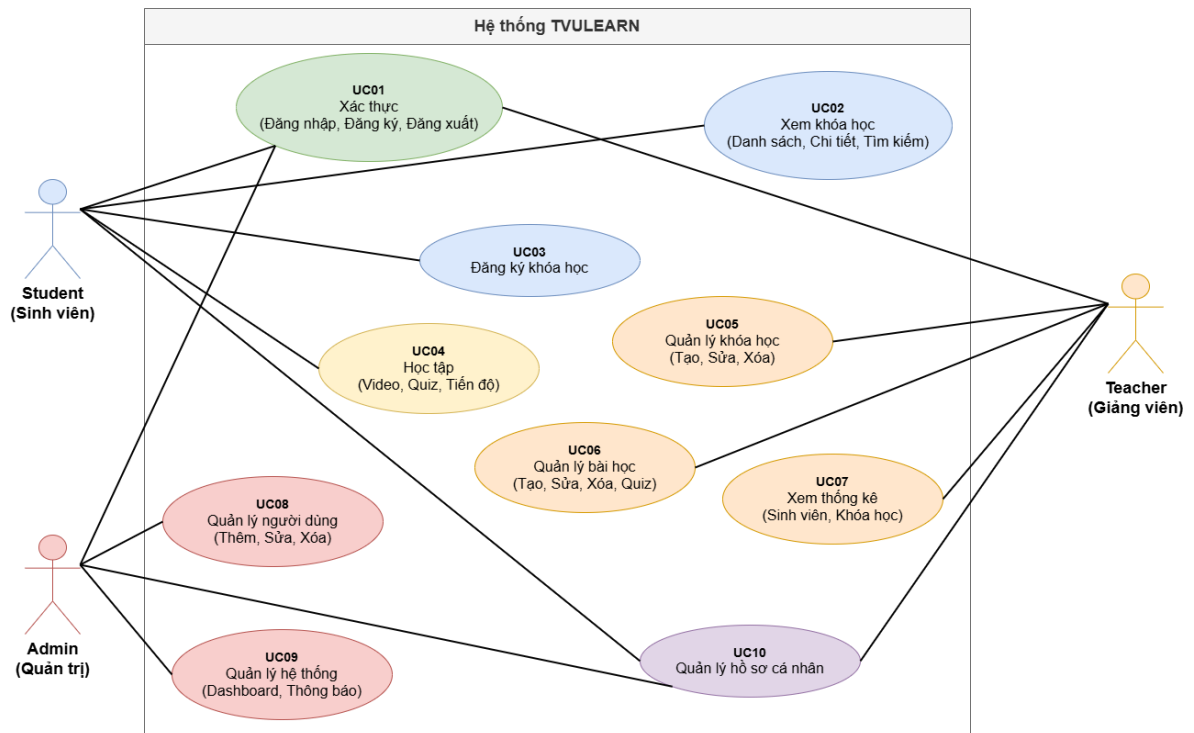
Dễ sử dụng: Giao diện được thiết kế đơn giản, trực quan, giúp người dùng dễ dàng thao tác và sử dụng.

Hiệu năng và ổn định: Hệ thống cần đảm bảo tốc độ phản hồi nhanh và hoạt động ổn định trong quá trình sử dụng.

Khả năng mở rộng: Hệ thống có thể được mở rộng và bổ sung thêm các chức năng mới trong tương lai.

3.2.2. Thiết kế chức năng hệ thống

3.2.2.1. Sơ đồ Use Case



Hình 3.1. Mô hình Use Case

3.2.2.2. Mô tả chi tiết các Use Case

Bảng 3.1. Chi tiết các Use Case

Mã UC	Actors	Mô tả	Điều kiện	Luồng xử lý
UC01 Xác thực	Student, Teacher, Admin	Đăng ký, đăng nhập, đăng xuất hệ thống	Đăng nhập cần có tài khoản	1. Người dùng nhập email, mật khẩu 2. Hệ thống xác thực thông tin 3. Tạo JWT token và phiên đăng nhập 4. Chuyển hướng theo Role

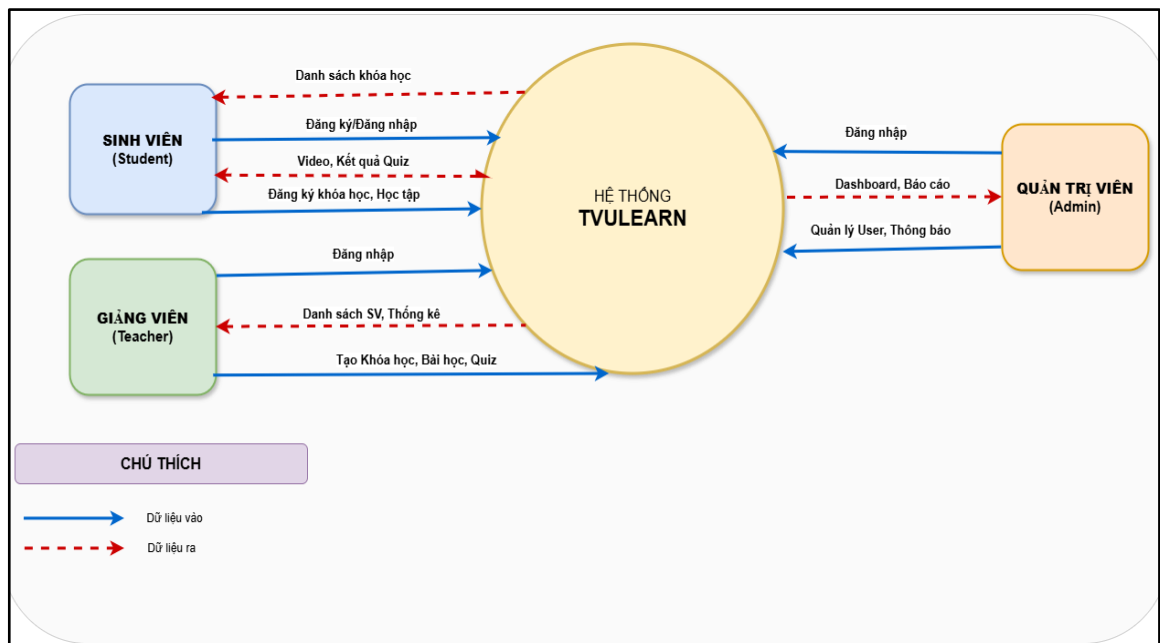
UC02 Xem khóa học	Student	Xem danh sách, tìm kiếm, chi tiết khóa học	Đăng nhập vào hệ thống	<ol style="list-style-type: none"> 1. Người dùng xem danh sách khóa học 2. Hệ thống hiển thị tất cả khóa học 3. Người dùng có thể tìm kiếm, lọc theo năm học 4. Chọn khóa học → Hiển thị chi tiết
UC03 Đăng ký khóa học	Student	Sinh viên đăng ký tham gia khóa học	Sinh viên đã đăng nhập; Chưa đăng ký khóa học	<ol style="list-style-type: none"> 1. Sinh viên xem chi tiết khóa học 2. Nhấn "Đăng ký khóa học" 3. Hệ thống tạo enrollment mới 4. Hiển thị "Đăng ký thành công"
UC04 Học tập	Student	Xem video, làm quiz, theo dõi tiến độ	Sinh viên đã đăng nhập và đăng ký khóa học	<ol style="list-style-type: none"> 1. Sinh viên chọn bài học 2. Hệ thống phát video 3. Hoàn thành video → Cập nhật tiến độ 4. Làm quiz → Chấm điểm và lưu kết quả 5. Xem tiến độ
UC05 Quản lý khóa học	Teacher	Tạo, sửa, xóa khóa học	Giảng viên đã đăng nhập	<ol style="list-style-type: none"> 1. Giảng viên nhấn "Tạo khóa học" 2. Nhập thông tin, upload thumbnail 3. Hệ thống validate và lưu khóa học 4. Hiển thị "Tạo thành công"

UC06 Quản lý bài học	Teacher	Tạo, sửa, xóa bài học và câu hỏi quiz	Giảng viên đã đăng nhập; Khóa học đã tồn tại	<ol style="list-style-type: none"> 1. Giảng viên vào quản lý khóa học 2. Thêm bài học: tiêu đề, video, nội dung 3. Thêm câu hỏi quiz với 4 đáp án 4. Hệ thống lưu bài học và quiz
UC07 Xem thống kê	Teacher	Xem danh sách sinh viên và thống kê khóa học	Giảng viên đã đăng nhập; Có khóa học	<ol style="list-style-type: none"> 1. Giảng viên vào tab "Sinh viên" 2. Hệ thống hiển thị danh sách SV, tiến độ 3. Vào "Thống kê" → Hiển thị biểu đồ, số liệu
UC08 Quản lý người dùng	Admin	Thêm, sửa, xóa người dùng	Admin đã đăng nhập	<ol style="list-style-type: none"> 1. Admin truy cập quản lý người dùng< 2. Hiển thị danh sách tất cả user 3. Thêm/sửa/xóa user 4. Hệ thống thực hiện và thông báo
UC09 Quản lý hệ thống	Admin	Xem dashboard và quản lý thông báo	Admin đã đăng nhập	<ol style="list-style-type: none"> 1. Admin vào Dashboard 2. Hiển thị thống kê tổng quan (user, khóa học) 3. Tạo thông báo hệ thống 4. Gửi cho người dùng được chọn

UC10 Quản lý hồ sơ cá nhân	Student	Xem và cập nhật thông tin cá nhân	Sinh viên đã đăng nhập	<ol style="list-style-type: none"> 1. Người dùng vào "Hồ sơ cá nhân" 2. Hiển thị thông tin hiện tại 3. Sửa thông tin, đổi mật khẩu 4. Hệ thống cập nhật và thông báo
---	---------	-----------------------------------	------------------------	--

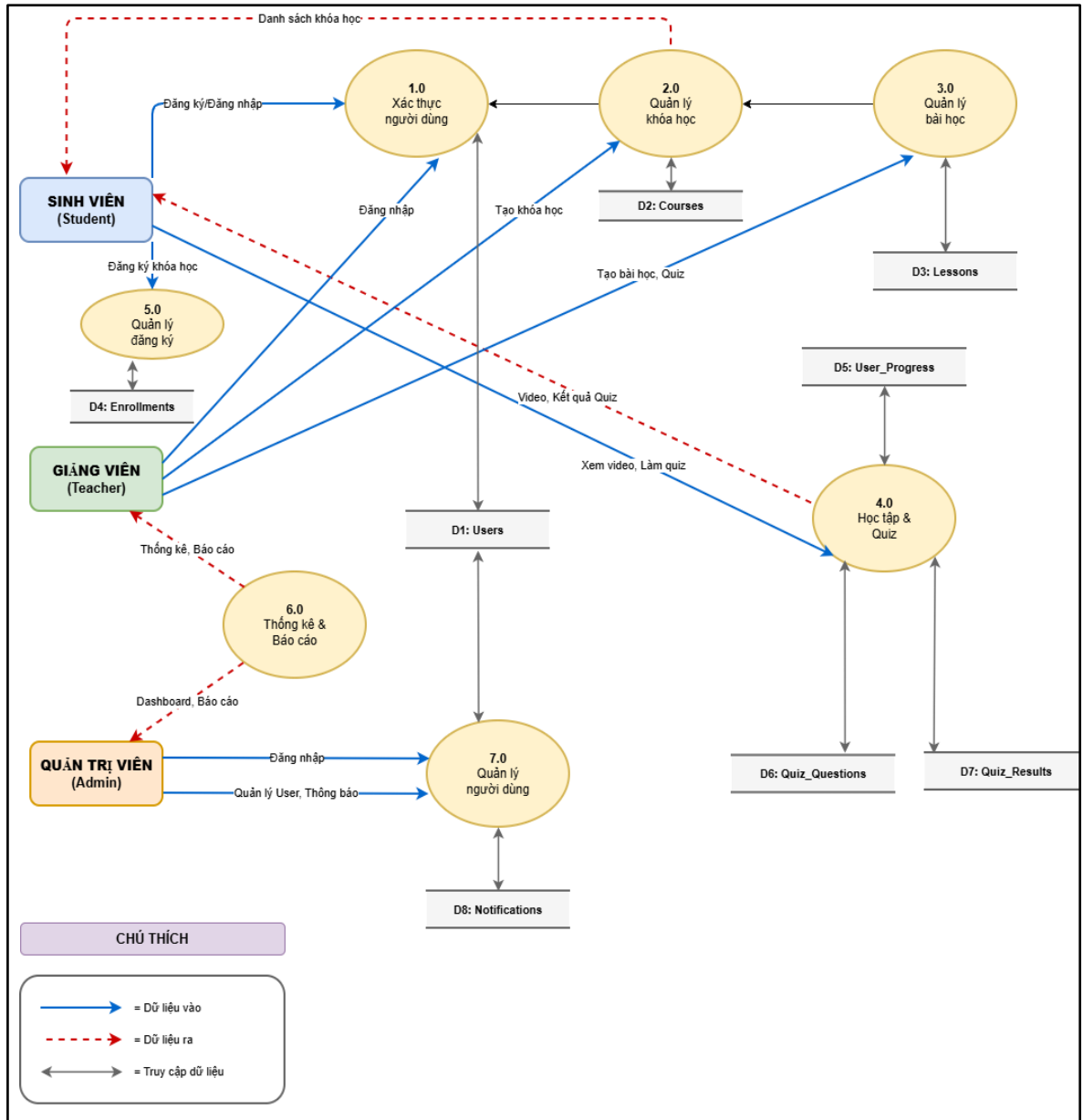
3.2.3. Thiết kế quy trình

3.2.3.1. Mô hình DFD mức 0



Hình 3.2. Mô hình luồng dữ liệu mức 0

3.2.3.2. Mô hình DFD mức 1



Hình 3.3. Mô hình luồng dữ liệu mức 1

3.2.4. Thiết kế dữ liệu

Để đảm bảo việc lưu trữ và quản lý dữ liệu một cách hiệu quả, hệ thống TVULEARN sử dụng hệ quản trị cơ sở dữ liệu MySQL. Cơ sở dữ liệu được thiết kế theo mô hình cơ sở dữ liệu quan hệ, đảm bảo tính nhất quán, toàn vẹn và dễ mở rộng trong tương lai.

Cơ sở dữ liệu phục vụ cho việc quản lý người dùng, khóa học, bài học, quiz, tiến trình học tập và các chức năng liên quan khác của hệ thống.

3.2.4.1. Danh sách các thực thể và mối kết hợp

a) Danh sách các thực thể

Bảng 3.2. Danh sách các thực thể

Thực thể	Mô tả
users	Lưu thông tin người dùng (sinh viên, giảng viên, quản trị viên)
courses	Lưu thông tin các khóa học do giảng viên tạo
lessons	Lưu thông tin các bài học thuộc từng khóa học
enrollments	Thể hiện việc sinh viên đăng ký tham gia một khóa học cụ thể
quiz_questions	Lưu thông tin các câu hỏi trắc nghiệm thuộc bài học
quiz_results	Lưu kết quả làm bài trắc nghiệm của sinh viên
user_progress	Theo dõi tiến trình học tập của sinh viên
notifications	Lưu các thông báo được gửi đến người dùng trong hệ thốn
refresh_tokens	Lưu refresh token phục vụ xác thực và duy trì đăng nhập

b) Danh sách các mối kết hợp

Bảng 3.3. Danh sách các mối kết hợp

Các mối kết hợp	Kiểu quan hệ
User – Course	1 – n
User – Enrollment	1 – n
Course – Enrollment	1 – n
Course – Lesson	1 – n
Lesson – Quiz_question	1 – n
User – Quiz_result	1 – n
Quiz_question – Quiz_result	1 – n
User – User_progress	1 – n
Course – User_progress	1 – n
User – Notification	1 – n
User – Refresh_token	1 – n

3.2.4.2. Chi tiết các thực thể và mối kết hợp

Tên thực thể: Users

Mô tả: lưu trữ thông tin tài khoản của tất cả người dùng trong hệ thống

Chi tiết thực thể:

Bảng 3.4. Chi tiết thực thể User

STT	Thuộc tính	Mô tả	Kiểu dữ liệu	Ràng buộc
1	user_id	Mã người dùng	INT(11)	PK (Khóa chính)
2	Name	tên người dùng	VARCHAR(255)	
3	Email	Địa chỉ email	VARCHAR(255)	
4	password	Mật khẩu	VARCHAR(255)	
5	Role	Vai trò người dùng	ENUM	
6	Note	Ghi chú thêm	TEXT	
7	created_at	Thời gian tạo	DATETIME	

Tên thực thể: Courses

Mô tả: lưu trữ thông tin về các khóa học có trong hệ thống.

Chi tiết thực thể:

Bảng 3.5. Chi tiết thực thể Courses

STT	Thuộc tính	Mô tả	Kiểu dữ liệu	Ràng buộc
1	course_id	Mã khóa học	INT(11)	PK (Khóa chính)
2	title	Tên khóa học	VARCHAR(255)	
3	description	Mô tả chi tiết	TEXT	
4	teacher_id	Mã giảng viên	INT(11)	FK (Tham chiếu users.user_id)
5	thumbnail	Đường dẫn ảnh	VARCHAR(255)	
6	note	Ghi chú	TEXT	
7	created_at	Thời gian tạo	DATETIME	
8	year	Năm học	TINYINT(4)	

Tên thực thể: Lessons

Mô tả: chứa nội dung chi tiết của từng bài học thuộc về một khóa học cụ thể.

Chi tiết thực thể:

Bảng 3.6. Chi tiết thực thể Lessons

STT	Thuộc tính	Mô tả	Kiểu dữ liệu	Ràng buộc
1	lesson_id	Mã định danh	INT(11)	PK (Khóa chính)
2	course_id	Mã khóa học chứa bài này	INT(11)	FK (Tham chiếu courses.course_id)
3	Title	Tên bài học	VARCHAR(255)	
4	video_url	Đường dẫn video	VARCHAR(255)	
5	content	Nội dung	TEXT	
6	order_index	Thứ tự sắp xếp bài	INT(11)	
7	Note	Ghi chú	TEXT	

Tên thực thể: Enrollments

Mô tả: quản lý việc ghi danh của học viên vào các khóa học và trạng thái.

Chi tiết thực thể:

Bảng 3.7. Chi tiết thực thể Enrollments

STT	Thuộc tính	Mô tả	Kiểu dữ liệu	Ràng buộc
1	enrollment_id	Mã đăng ký	INT(11)	PK (khóa chính)
2	user_id	Mã học viên	INT(11)	FK (Tham chiếu users.user_id)
3	course_id	Mã khóa học	INT(11)	FK (Tham chiếu courses.course_id)
4	progress	Tiến độ tổng quan	INT(11)	
5	status	Trạng thái học tập	ENUM('enrolled', 'completed')	DEFAULT 'enrolled'
6	enrolled_at	Thời gian đăng ký	DATETIME	
7	note	Ghi chú	VARCHAR(255)	

Tên thực thể: User_Progress

Mô tả: theo dõi tiến độ của từng bài học cụ thể đối với từng người dùng.

Chi tiết thực thể:

Bảng 3.8. Chi tiết thực thể User_Progress

STT	Thuộc tính	Mô tả	Kiểu dữ liệu	Ràng buộc
1	progress_id	Mã tiến độ chi tiết	INT(11)	PK(khóa chính)
2	user_id	Mã người dùng	INT(11)	FK (Tham chiếu users.user_id)
3	course_id	Mã khóa học	INT(11)	FK (Tham chiếu courses.course_id)
4	lesson_id	Mã bài học	INT(11)	FK (Tham chiếu lessons.lesson_id)
5	completed	Trạng thái hoàn thành	TINYINT(1)	DEFAULT 0
6	completed_at	Thời gian hoàn thành	DATETIME	
*	Ràng buộc kép	Ngăn trùng lặp tiến độ	Unique Key	UNIQUE (user_id, lesson_id)

Tên thực thể: Quiz_Questions

Mô tả: Lưu trữ ngân hàng câu hỏi trắc nghiệm gắn liền với từng bài học.

Chi tiết thực thể:

Bảng 3.9. Chi tiết thực thể Quiz_Questions

STT	Thuộc tính	Mô tả	Kiểu dữ liệu	Ràng buộc
1	question_id	Mã câu hỏi	INT(11)	PK(khóa chính)
2	lesson_id	Mã bài học chứa câu hỏi	INT(11)	FK(Tham chiếu lessons.lesson_id)
3	question	Nội dung câu hỏi	TEXT	
4	options	Các lựa chọn	LONGTEXT (JSON)	CHECK (json_valid(options))
5	correct_answer	Đáp án đúng	VARCHAR(255)	

Tên thực thể: Quiz_Results

Mô tả: Lưu trữ kết quả làm bài kiểm tra trắc nghiệm của người dùng.

Chi tiết thực thể:

Bảng 3.10. Chi tiết thực thể Quiz_Results

STT	Thuộc tính	Mô tả	Kiểu dữ liệu	Ràng buộc
1	result_id	Mã kết quả	INT(11)	PK(khóa chính)
2	user_id	Mã người dùng làm bài	INT(11)	FK (Tham chiếu users.user_id)
3	lesson_id	Mã bài học có bài test	INT(11)	FK (Tham chiếu lessons.lesson_id)
4	course_id	Mã khóa học	INT(11)	FK (Tham chiếu courses.course_id),
5	Score	Điểm số đạt được	INT(11)	
6	total_questions	Tổng số câu hỏi	INT(11)	
7	created_at	Thời gian nộp bài	TIMESTAMP	

Tên thực thể: Notifications

Mô tả: Hệ thống thông báo gửi tới người dùng.

Chi tiết thực thể:

Bảng 3.11. Chi tiết thực thể Notifications

STT	Thuộc tính	Mô tả	Kiểu dữ liệu	Ràng buộc
1	id	Mã thông báo	INT(11)	PK(khóa chính)
2	user_id	Người nhận thông báo	INT(11)	FK (Tham chiếu users.user_id)
3	message	thông báo	VARCHAR(255)	
4	type	Loại thông báo	VARCHAR(50)	DEFAULT 'info'
5	status	Trạng thái đọc	VARCHAR(20)	
6	created_at	Thời gian tạo	TIMESTAMP	

Tên thực thể: Refresh_Tokens

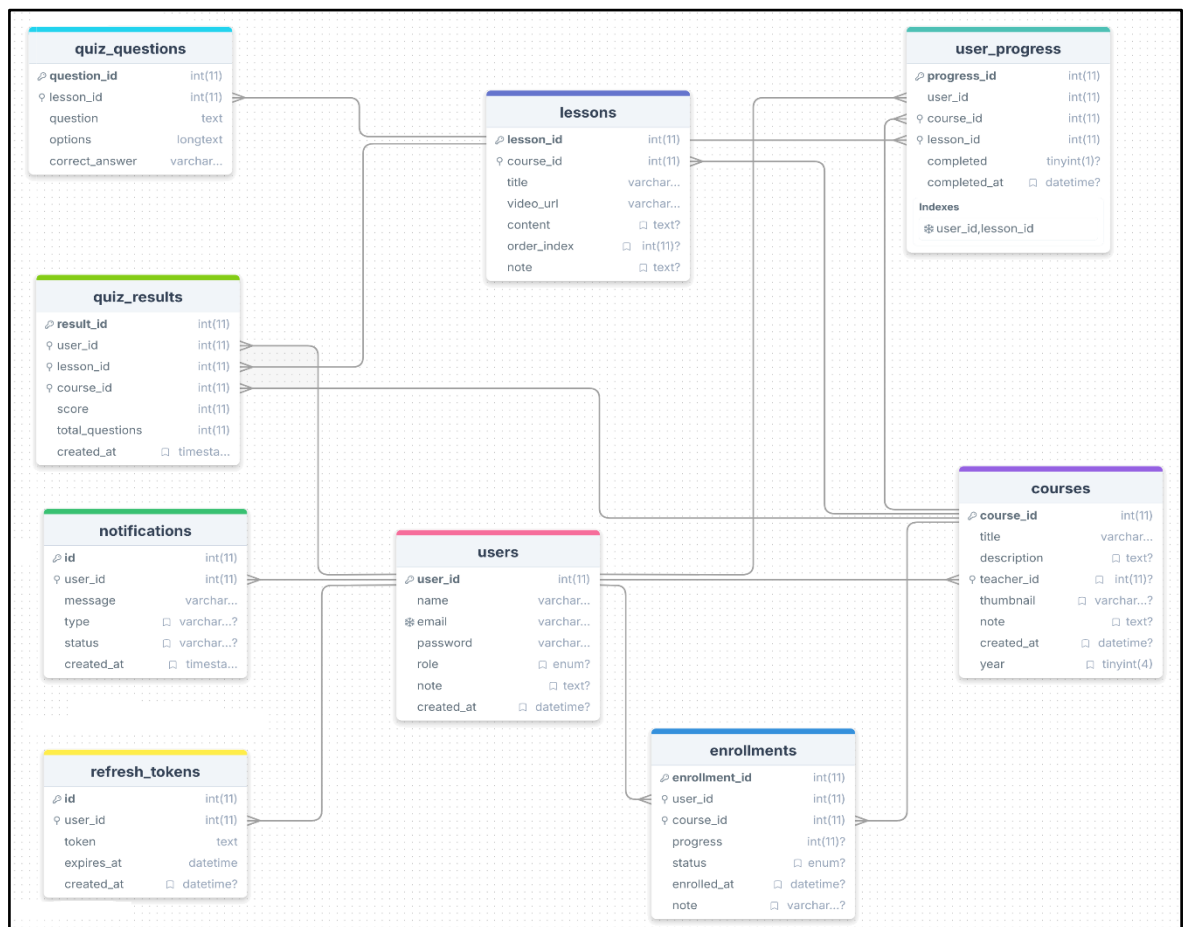
Mô tả: Quản lý phiên đăng nhập và bảo mật thông qua refresh token.

Chi tiết thực thể:

Bảng 3.12. Chi tiết thực thể Refresh_tokens

STT	Thuộc tính	Mô tả	Kiểu dữ liệu	Ràng buộc
1	id	Mã token	INT(11)	PK(khóa chính)
2	user_id	Mã người dùng sở hữu	INT(11)	FK (Tham chiếu users.user_id)
3	token	Chuỗi token	TEXT	
4	expires_at	Thời gian hết hạn	DATETIME	
5	created_at	Thời gian tạo	DATETIME	

3.2.4.3. Mô hình ERD



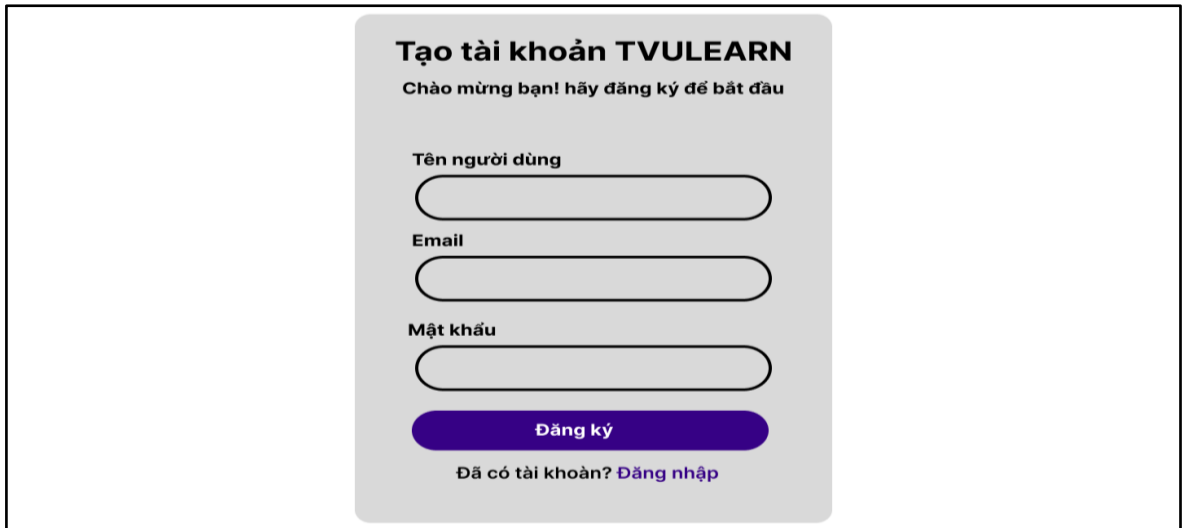
Hình 3.4. Mô hình ERD

3.2.5. Thiết kế giao diện

3.2.5.1. Giao diện trang đăng ký / đăng nhập

a) Đăng ký

Trang đăng ký cho phép người dùng mới tạo tài khoản với form gồm ba trường: Tên người dùng, Email và Mật khẩu. Có nút "Đăng ký" và link "Đăng nhập" ở cuối form để chuyển sang trang đăng nhập cho người đã có tài khoản.

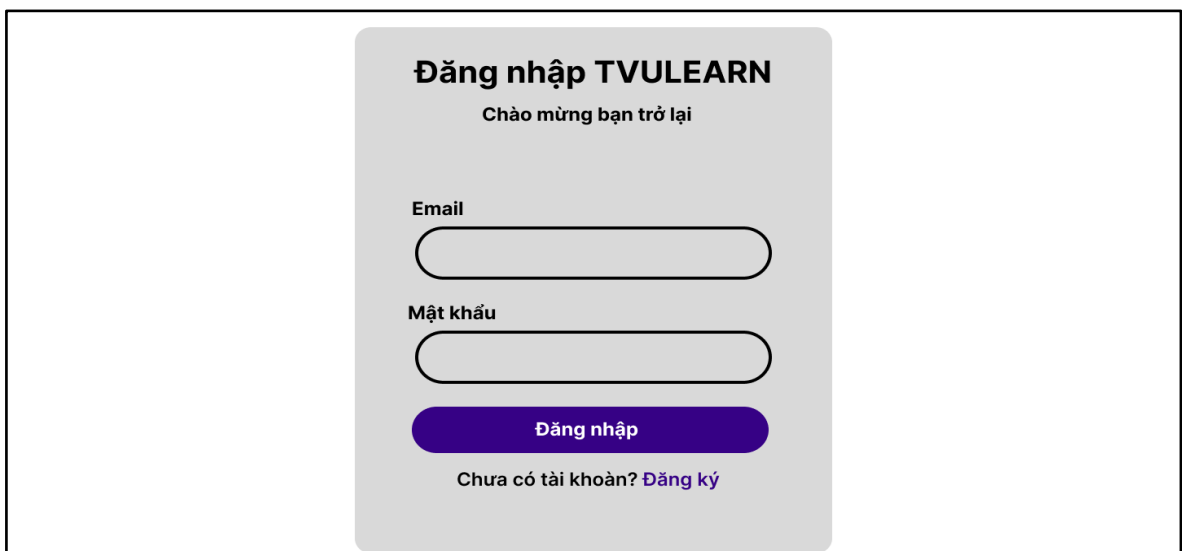


The image shows a registration form titled "Tạo tài khoản TVULEARN" (Create TVULEARN account). Below the title is a welcome message: "Chào mừng bạn! hãy đăng ký để bắt đầu" (Welcome! Register to get started). The form contains three input fields: "Tên người dùng" (Username), "Email", and "Mật khẩu" (Password). Below these fields is a purple button labeled "Đăng ký" (Register). At the bottom, there is a link that says "Đã có tài khoản? Đăng nhập" (Already have an account? Log in).

Hình 3.5. Giao diện trang đăng ký

b) Đăng nhập

Giao diện đăng nhập được thiết kế nhằm cho phép người dùng truy cập vào hệ thống bằng tài khoản đã đăng ký. Người dùng cần nhập email và mật khẩu để xác thực. Giao diện được thiết kế đơn giản, dễ sử dụng, phù hợp với sinh viên



The image shows a login form titled "Đăng nhập TVULEARN" (Log in TVULEARN). Below the title is a welcome message: "Chào mừng bạn trở lại" (Welcome back). The form contains two input fields: "Email" and "Mật khẩu" (Password). Below these fields is a purple button labeled "Đăng nhập" (Log in). At the bottom, there is a link that says "Chưa có tài khoản? Đăng ký" (Don't have an account? Register).

Hình 3.6. Giao diện trang đăng nhập

3.2.5.2. Giao diện trang chủ

Header: thanh ngang toàn chiều rộng, trái có logo, giữa là thanh tìm kiếm, phải có icon thông báo và người dùng.

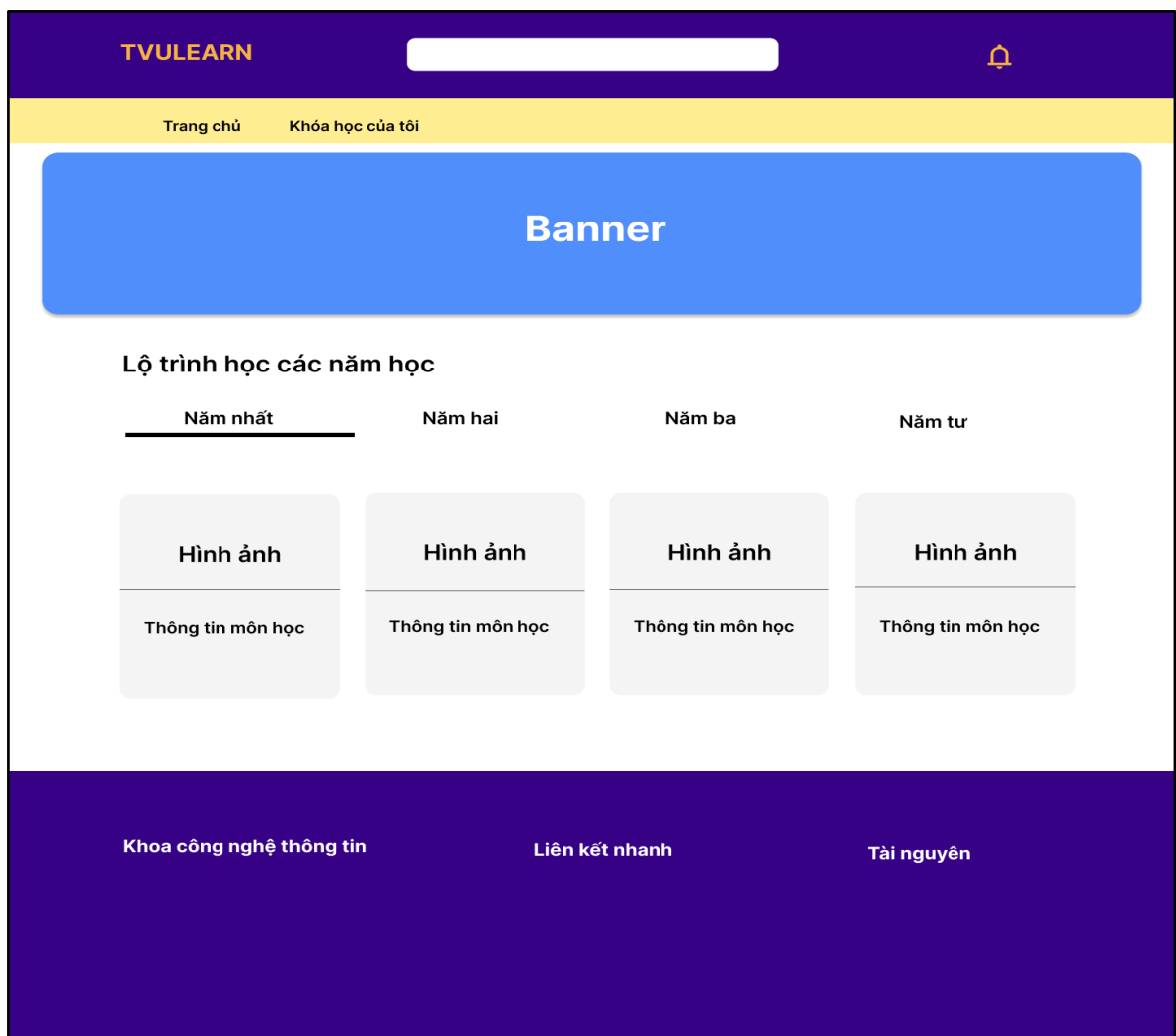
Thanh điều hướng: Chứa liên kết trang chủ và trang khóa học của tôi.

Banner: hiển thị thông tin quan trọng hoặc câu nói động lực, dạng slide.

Phân lọc khóa học: hàng tab gồm bốn nút lọc theo năm học.

Danh sách khóa học: mỗi card gồm hình ảnh khóa học và thông tin môn học.

Footer: Thanh nền tím đậm cuối trang, chia 3 cột đều nhau.



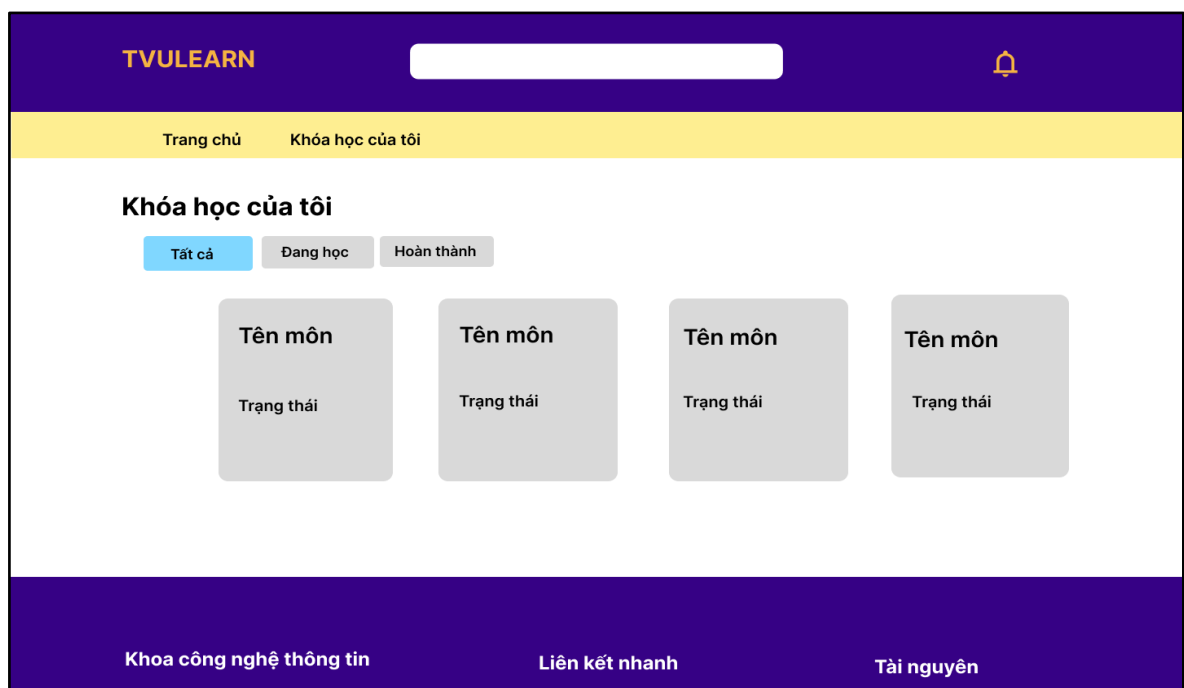
Hình 3.7. Giao diện trang chủ

3.2.5.3. Giao diện trang khóa học của tôi

Trang có phần header và footer tương tự như trang chủ.

Bộ lọc trạng thái: với ba nút gồm "Tất cả", "Đang học", "Hoàn thành".

Danh sách khóa học: hiển thị dạng lưới 4 cột, mỗi card gồm tên khóa học, trạng thái và tiến độ học tập (%).

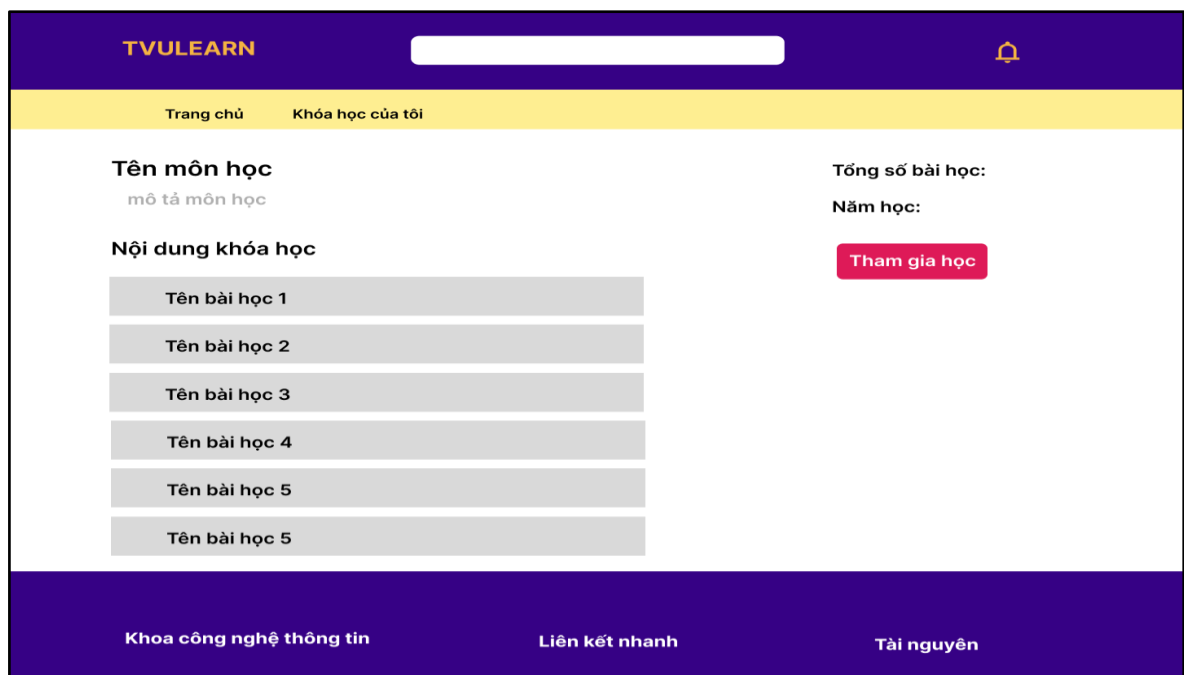


Hình 3.8. Giao diện trang khóa học của tôi

3.2.5.4. Giao diện trang chi tiết khóa học

Trang được chia thành hai phần chính:

- Nội dung khóa học: phần đầu hiển thị tên môn học và mô tả ngắn gọn về môn học. Bên dưới là phần "Nội dung khóa học" liệt kê danh sách các bài học theo thứ tự.
- Thông tin và hành động: hiển thị tổng số bài học và năm học của khóa học. Nút "Tham gia học" được đặt bên dưới để người dùng đăng ký tham gia khóa học.



Hình 3.9. Giao diện chi tiết khóa học

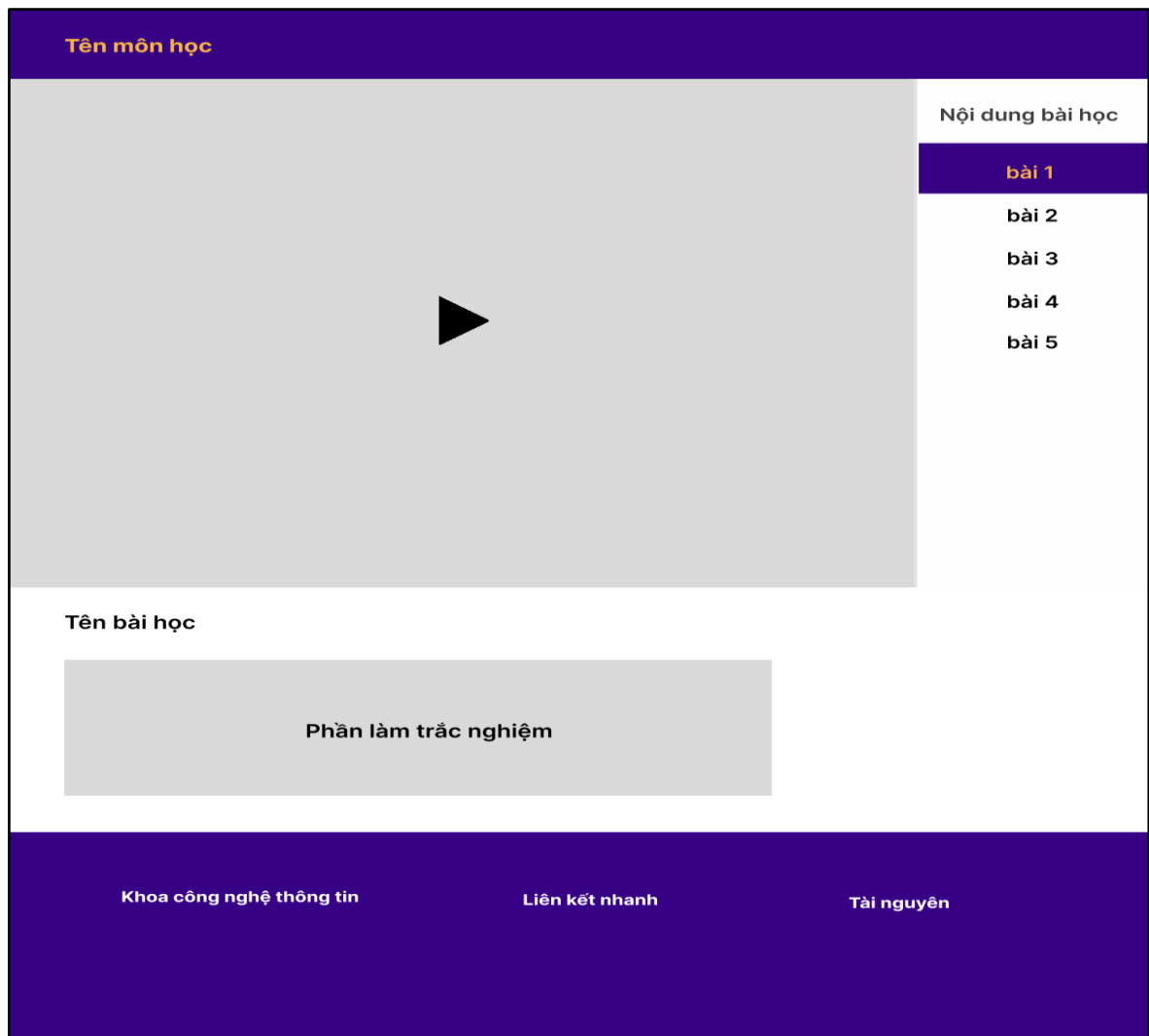
3.2.5.5. Giao diện trang học tập

Header: thanh ngang trên cùng hiển thị tên môn học.

Video player (Cột trái): chiếm phần lớn không gian bên trái, cho phép xem bài giảng với nút play; bên dưới hiển thị tên bài học.

Sidebar nội dung (Cột phải): liệt kê danh sách bài học theo chiều dọc, bài đang học được làm nổi bật, có thể click để chuyển bài.

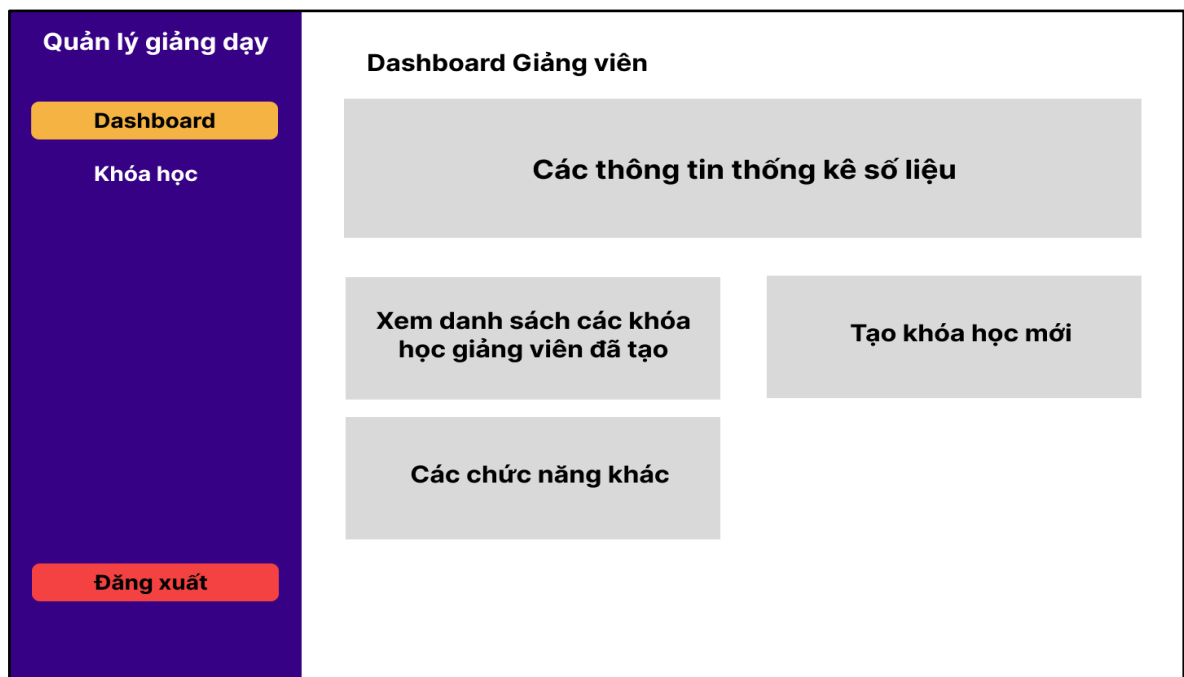
Phần trắc nghiệm: ở dưới video, cho phép thực hiện các bài kiểm tra liên quan.



Hình 3.10. Giao diện trang học tập

3.2.5.6. Giao diện trang quản lý giảng dạy

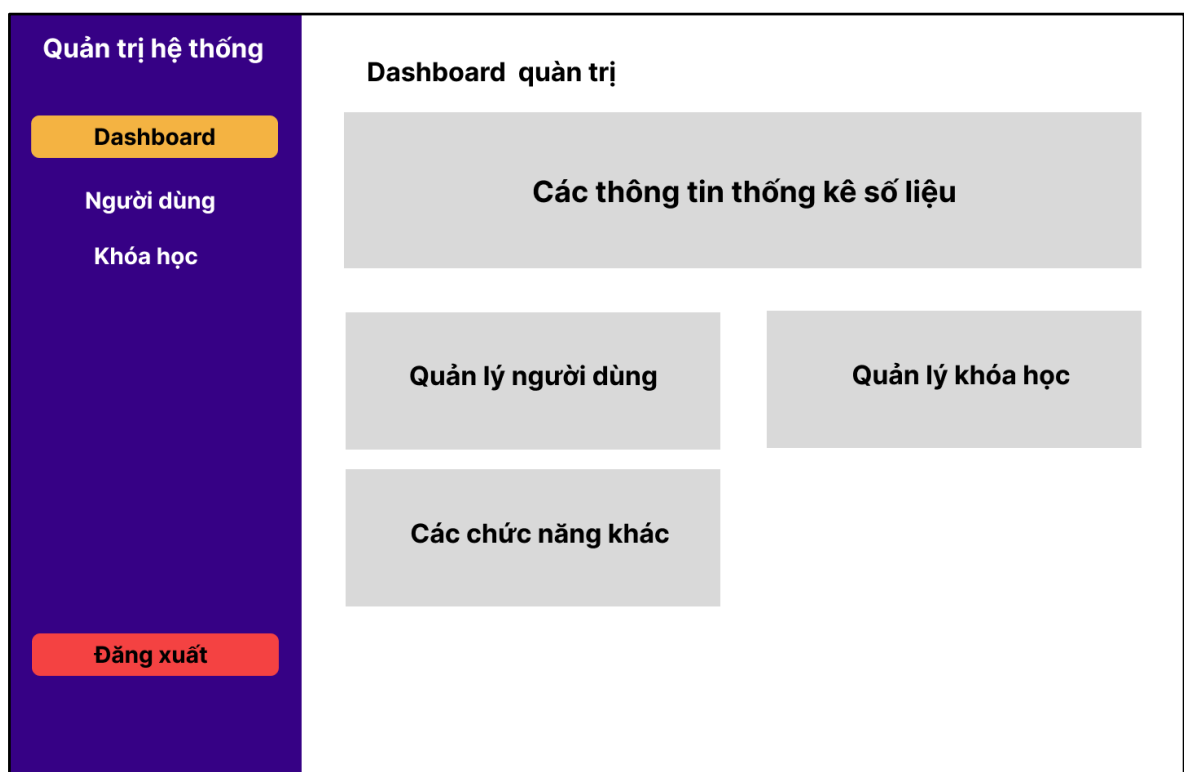
Trang quản lý giảng dạy là màn hình tổng quan, giúp giảng viên truy cập nhanh các chức năng chính của hệ thống quản lý giảng dạy: xem thống kê, quản lý khóa học và tạo khóa học mới.



Hình 3.11. Giao diện trang quản lý giảng dạy

3.2.5.7. Giao diện trang quản trị

Trang Dashboard quản trị là màn hình tổng quan dành cho quản trị viên theo dõi thống kê số liệu toàn hệ thống, truy cập nhanh các phân hệ quản lý người dùng và quản lý khóa học, điều hướng đến các chức năng mở rộng khác.



Hình 3.12. Giao diện trang quản trị

CHƯƠNG 4. KẾT QUẢ NGHIÊN CỨU

4.1. Tổng quan kết quả đạt được

Sau quá trình thực hiện đồ án, hệ thống website hỗ trợ học tập cho sinh viên Công nghệ Thông tin Trường Đại học Trà Vinh đã được xây dựng và hoàn thiện với các chức năng cơ bản theo đúng mục tiêu đề ra. Hệ thống cung cấp nền tảng học tập trực tuyến giúp người dùng truy cập lộ trình học theo năm học, tìm kiếm và xem nội dung học tập, qua đó hỗ trợ quá trình học tập thuận tiện và tập trung hơn so với hình thức quản lý tài liệu rời rạc.

Bên cạnh đó, đồ án đã triển khai đầy đủ các vai trò người dùng gồm sinh viên, giảng viên và quản trị viên với các chức năng tương ứng. Sinh viên có thể đăng ký/đăng nhập và sử dụng các chức năng học tập; giảng viên có thể quản lý nội dung giảng dạy như khóa học/bài học; quản trị viên có thể quản lý người dùng và quản lý hệ thống. Các chức năng chính được tổ chức rõ ràng theo từng nhóm và vận hành ổn định, đáp ứng yêu cầu tổng quan của đề tài và tạo nền tảng để tiếp tục mở rộng các tính năng nâng cao trong giai đoạn sau.

4.2. Dữ liệu mẫu của hệ thống

4.2.1. Dữ liệu mẫu tài khoản và phân quyền

Trong cơ sở dữ liệu, hệ thống chuẩn bị sẵn 03 tài khoản mẫu, tương ứng với 03 vai trò chính, nhằm phục vụ cho việc kiểm thử và minh họa chức năng:

- 01 tài khoản sinh viên (role = student): dùng để kiểm tra các chức năng học tập như tham gia khóa học, học bài, làm bài kiểm tra trắc nghiệm và theo dõi tiến trình học tập.

- 01 tài khoản giảng viên (role = teacher): dùng để kiểm tra các chức năng dành cho giảng viên như tạo khóa học, quản lý bài học, tạo và quản lý quiz trắc nghiệm.

- 01 tài khoản quản trị viên (role = admin): dùng để kiểm tra các chức năng quản trị hệ thống như quản lý người dùng, quản lý khóa học và tạo giảng viên.

4.2.2. Dữ liệu nghiệp vụ phát sinh từ thao tác trên giao diện

Ngoài các tài khoản mẫu được khởi tạo ban đầu, toàn bộ dữ liệu nghiệp vụ còn lại của hệ thống được tạo và thay đổi trực tiếp thông qua giao diện người dùng trong quá trình sử dụng, bao gồm:

Khóa học: được tạo mới, chỉnh sửa hoặc xóa bởi giảng viên hoặc quản trị viên.

Bài học: được thêm mới, chỉnh sửa và gắn với từng khóa học cụ thể.

Quiz trắc nghiệm: bao gồm câu hỏi, các đáp án và đáp án đúng, được tạo và quản lý theo từng bài học.

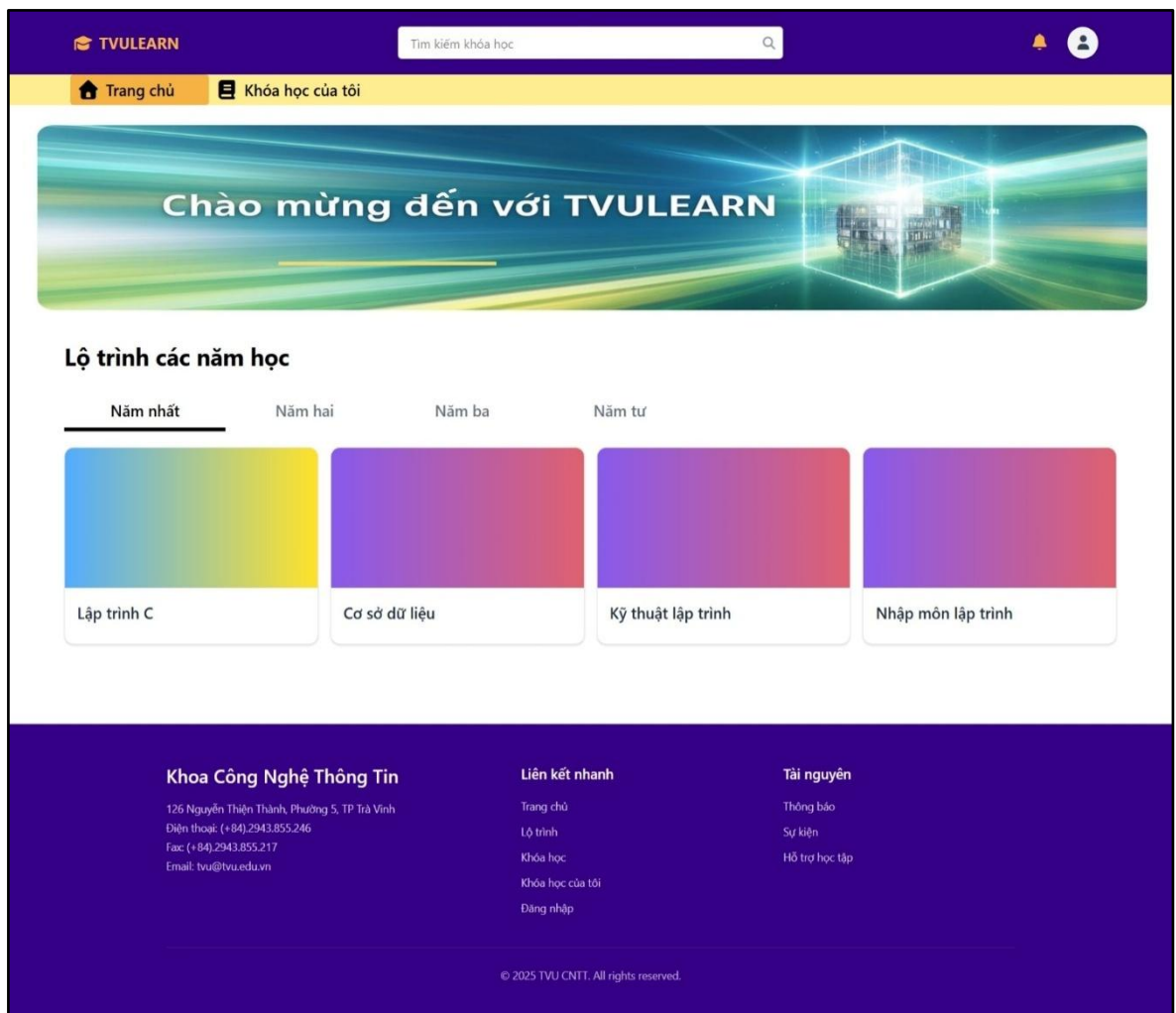
Ghi danh, tiến trình học tập và kết quả làm quiz: phát sinh trong quá trình sinh viên tham gia học tập và làm bài kiểm tra.

Việc cho phép dữ liệu nghiệp vụ được thao tác trực tiếp trên giao diện giúp hệ thống kiểm tra đầy đủ các chức năng thêm – sửa – xóa – hiển thị dữ liệu, đồng thời đảm bảo tính nhất quán giữa cơ sở dữ liệu và cơ chế phân quyền của hệ thống.

4.3. Giao diện và chức năng chính của hệ thống

4.3.1. Giao diện cho sinh viên

4.3.1.1. Giao diện trang chủ



Hình 4.1. Kết quả giao diện trang chủ

Các chức năng của trang chủ:

- Thanh tìm kiếm khóa học: cho phép người dùng nhập tên môn/khóa học.

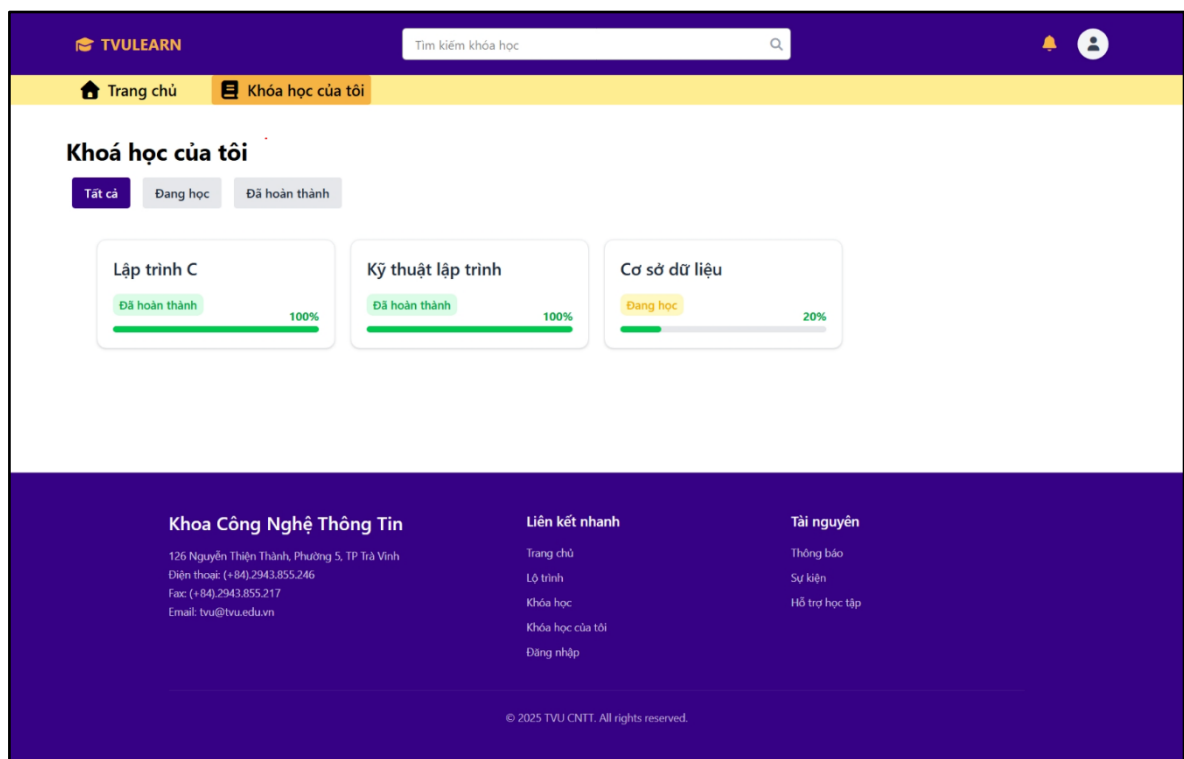
Hệ thống hiển thị gợi ý các khóa học có tên phù hợp với từ khóa đã nhập, giúp tìm nhanh và chính xác.

Chuông thông báo: hiển thị danh sách thông báo gửi từ quản trị viên.

Banner chuyển động: khu vực banner dạng slideshow tạo điểm nhấn cho trang. Tự động chuyển slide sau mỗi 5 giây để hiển thị nhiều nội dung giới thiệu/thông tin.

Danh sách khóa học theo năm học các khóa học được phân theo từng năm (Năm nhất → Năm tư) bằng tab/chuyên mục. Giúp sinh viên dễ dàng chọn đúng môn học phù hợp với lộ trình học của mình.

4.3.1.2. Giao diện trang khóa học của tôi



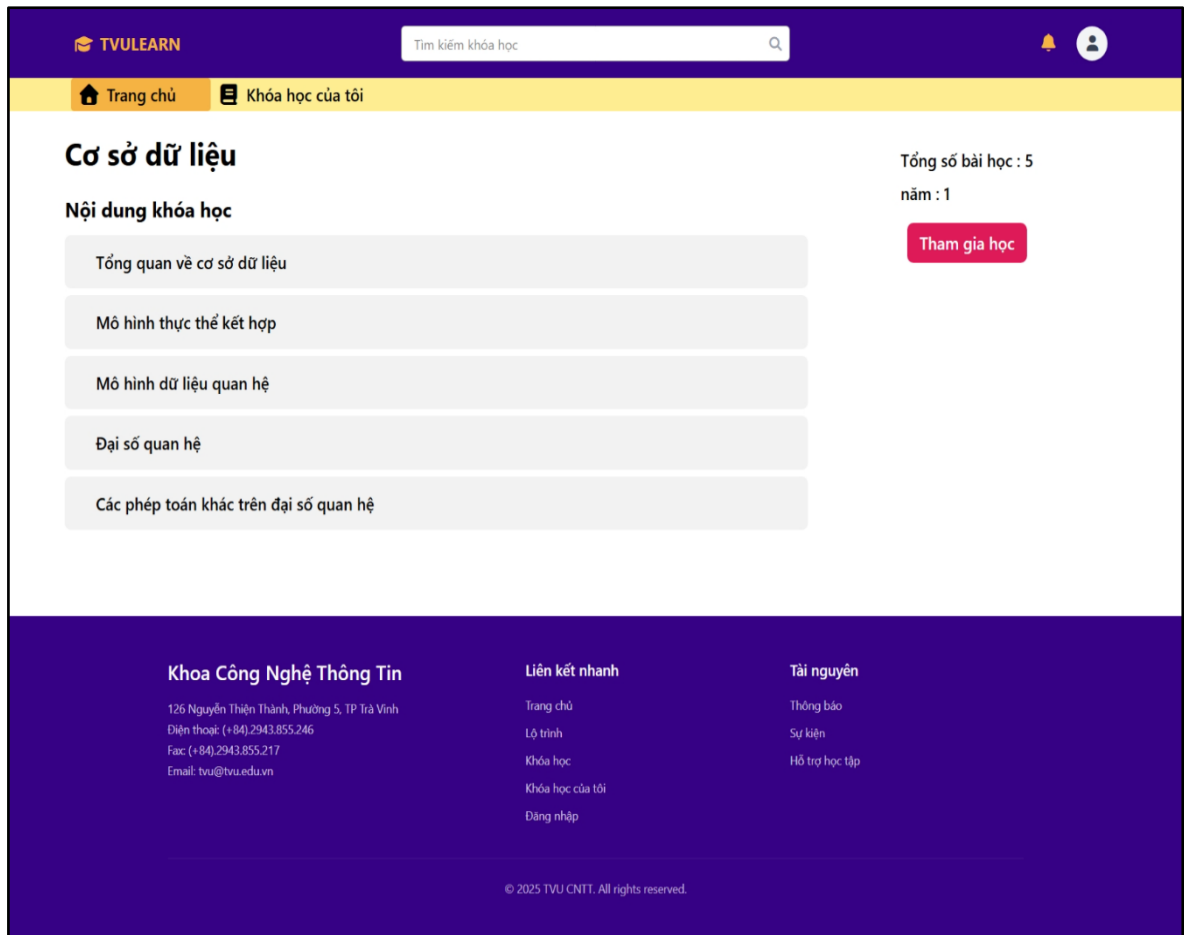
Hình 4.2. Kết quả trang khóa học của tôi

Các chức năng của trang khóa học của tôi:

- Lọc dữ liệu khóa học: cung cấp các nút lọc gồm ba trạng thái (tất cả, đang học, đã hoàn thành).

- Hiển thị tiến độ hoàn thành: mỗi khóa học được hiển thị dưới dạng thẻ (card) kèm thanh tiến độ và phần trăm hoàn thành. Tiến độ được tính dựa trên tỷ lệ số bài học đã học/đã hoàn thành trên tổng số bài học của khóa học. Nhờ đó người dùng dễ dàng biết khóa học đang học tới đâu và phân biệt khóa học đã hoàn thành.

4.3.1.3. Giao diện trang chi tiết khóa học

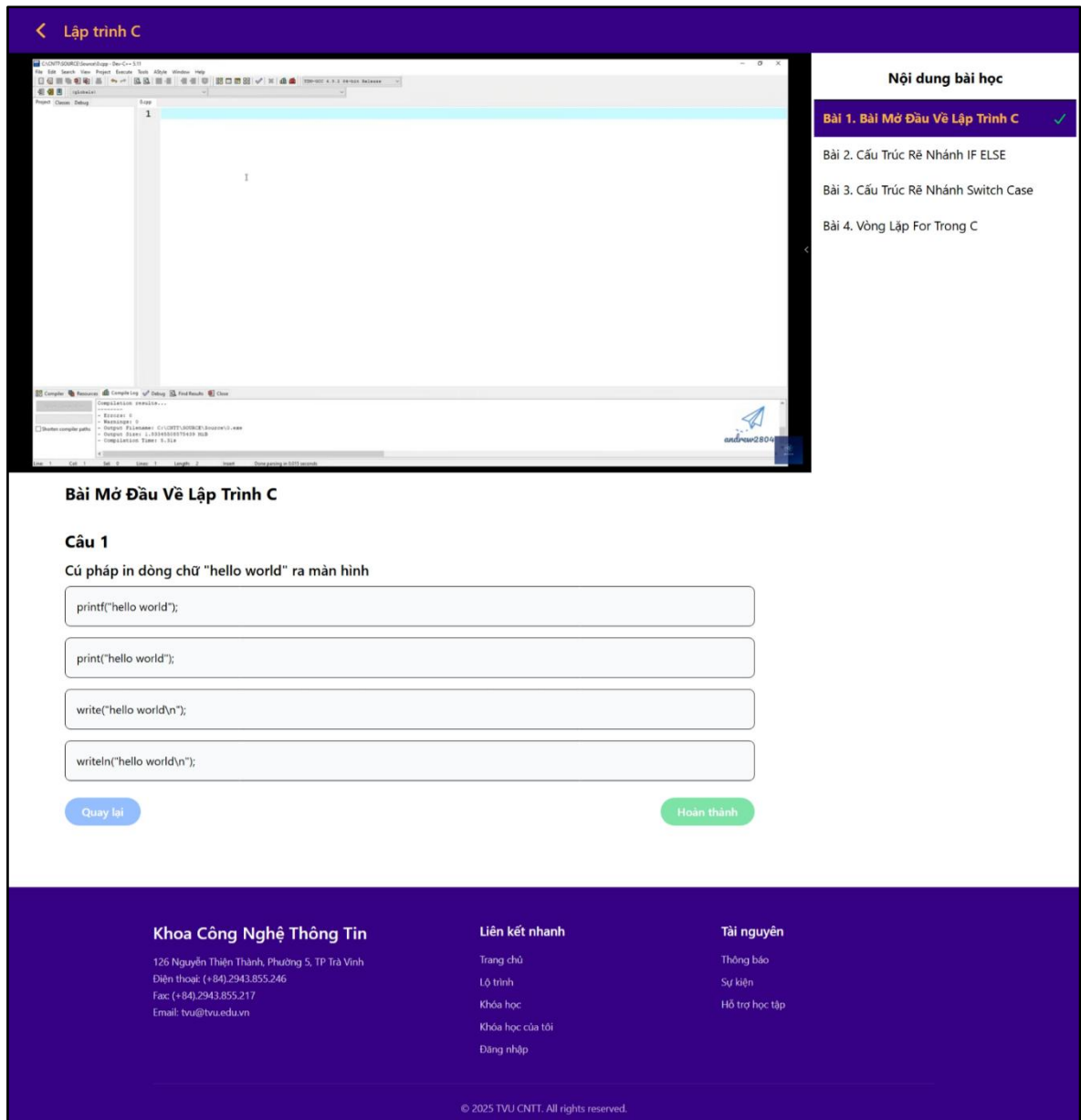


Hình 4.3. Kết quả giao diện chi tiết khóa học

Các chức năng của trang chi tiết khóa học:

- Hiển thị danh sách các bài học (nội dung khóa học) để sinh viên xem trước các chủ đề/bài học sẽ học trong môn học.
- Hiển thị thông tin tổng quan của khóa học gồm: tổng số bài học và năm học của môn học đó.
- Cung cấp nút “Tham gia học” để sinh viên đăng ký/tham gia vào môn học đang chọn và bắt đầu học.

4.3.1.4. Giao diện trang học tập



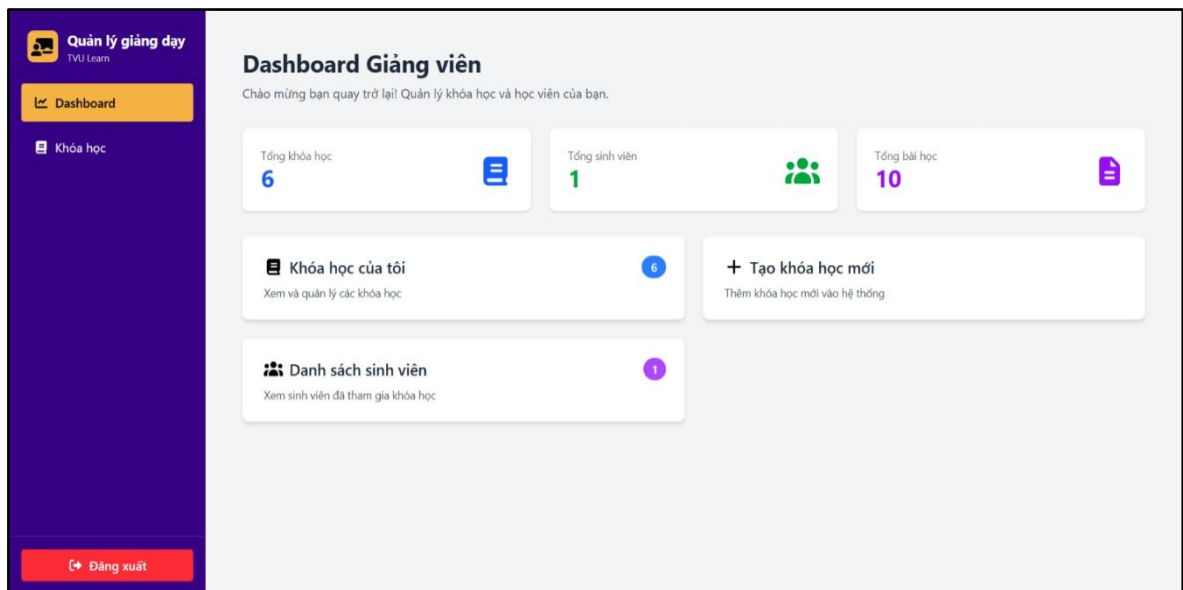
Hình 4.4. Kết quả giao diện trang học tập

Các chức năng của trang học tập:

- Xem video bài học: phát nội dung bài học thông qua video được nhúng từ YouTube (iframe), giúp sinh viên học trực tiếp trên hệ thống.
- Hiển thị danh sách bài học: liệt kê toàn bộ bài trong khóa học ở thanh bên để người học dễ theo dõi tiến trình học tập và chọn nhanh bài cần học.
- Đánh dấu bài học đang chọn/đã hoàn thành: khi bấm vào một bài học, hệ thống tô nổi bật bài đó; đồng thời đánh dấu các bài đã học xong để theo dõi tiến độ.
- Làm bài tập trắc nghiệm: cung cấp phần câu hỏi trắc nghiệm theo từng bài cho phép chọn đáp án, nộp bài và nhận kết quả/điểm số sau khi hoàn thành.

4.3.2. Giao diện cho giảng viên

4.3.2.1. Giao diện trang quản lý giảng dạy

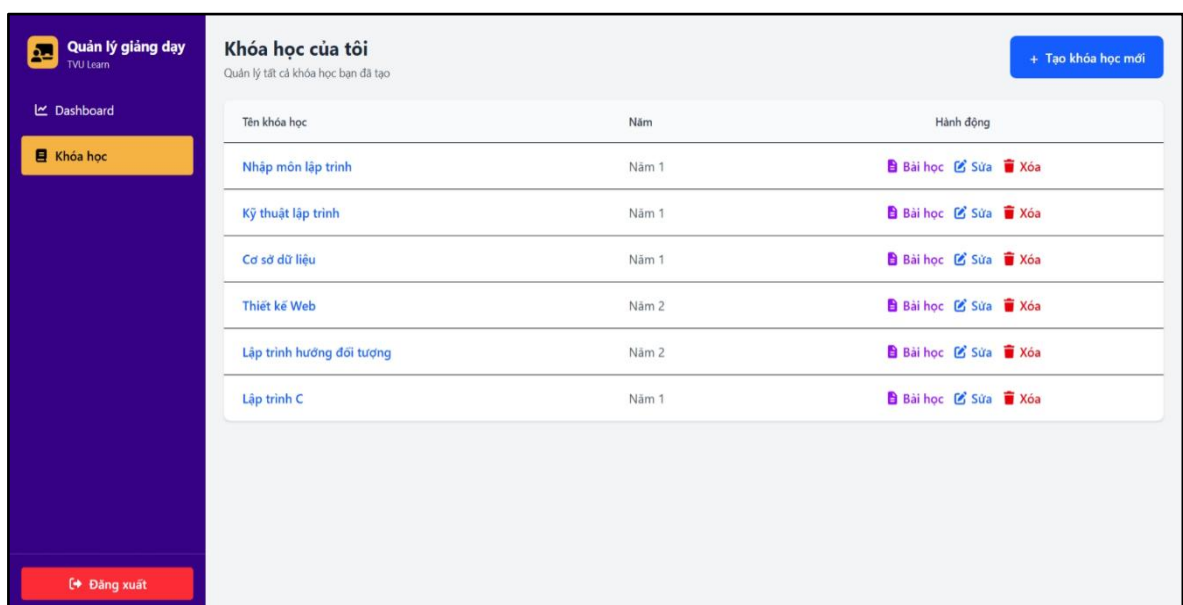


Hình 4.5. Kết quả giao diện trang cho giảng viên

Các chức năng chính trên dashboard:

- Thống kê thông tin: hiển thị tổng quan số liệu của giảng viên.
- Quản lý danh sách khóa học đã tạo.
- Tạo khóa học mới: chuyển đến form tạo khóa học, thêm mới khóa học.
- Danh sách sinh viên: xem danh sách sinh viên đã tham gia các khóa học của giảng viên (phục vụ theo dõi học viên).

4.3.2.2. Giao diện trang quản lý khóa học



Hình 4.6. Kết quả giao diện trang quản lý khóa học

Để vào trang quản lý khóa học giảng viên cần nhấp vào “Khóa học của tôi” hoặc “khóa học” bên menu bên trái. Trang gồm các chức năng:

- Hiện thị danh sách khóa học đã tạo: liệt kê các khóa học của giảng viên theo dạng bảng, gồm tên khóa học và năm để dễ quản lý.
- Tạo khóa học mới: nút “Tạo khóa học mới” để chuyển sang trang tạo khóa học mới
- Quản lý từng khóa học: với các chức năng thêm , sửa, xóa bài học.

4.3.2.3. Giao diện trang tạo khóa học

The screenshot shows the 'Tạo khóa học mới' (Create New Course) form. On the left is a dark purple sidebar with the TVU Learn logo and navigation links: 'Dashboard' and 'Khóa học' (highlighted in orange). At the bottom of the sidebar is a red 'Đăng xuất' (Logout) button. The main content area has a light gray background. At the top, there's a link to 'Quay lại danh sách khóa học' and the title 'Tạo khóa học mới' with the subtitle 'Thiết lập thông tin cơ bản cho khóa học của bạn'. The form itself is a white card with the following fields: 'Tên khóa học' (Course Name) with a placeholder 'Nhập tên khóa học'; 'Mô tả khóa học' (Course Description) with a placeholder 'Nhập mô tả khóa học'; 'Năm học' (Academic Year) with a dropdown menu showing '-- Chọn năm học --'; and 'Ảnh thumbnail khóa học' (Course Thumbnail Image) with a 'Choose File' button and the text 'No file chosen'. At the bottom of the form are two buttons: 'Tạo khóa học' (Create Course) in blue and 'Hủy' (Cancel) in gray.

Hình 4.7. Kết quả giao diện trang tạo khóa học

Để tạo một khóa học mới giảng viên cần nhập các thông tin khóa học gồm: tên khóa học, mô tả, năm học và chọn ảnh đại diện cho khóa học đó.

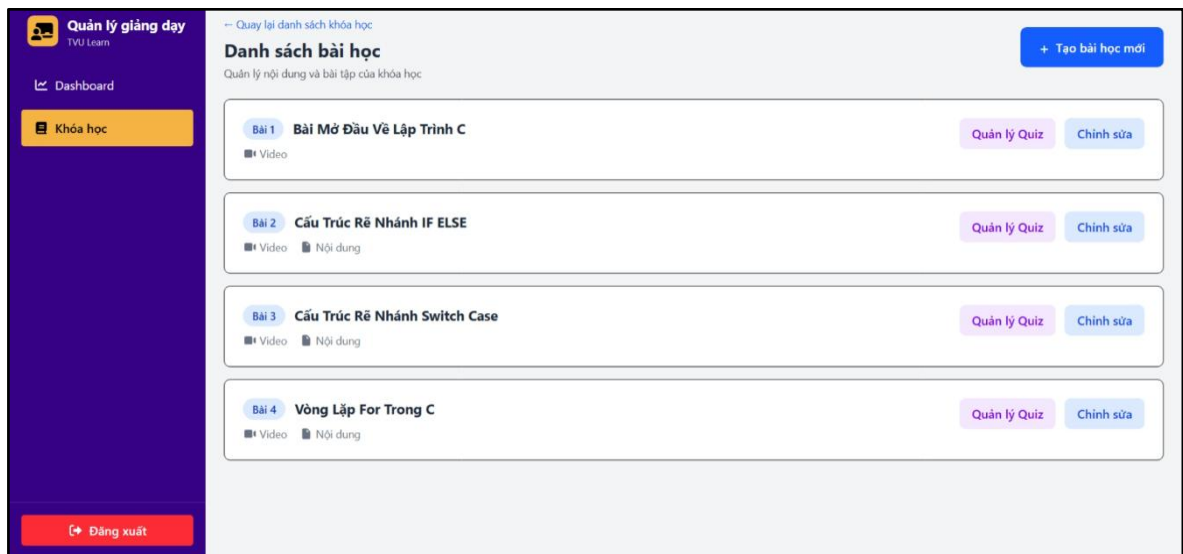
4.3.2.4. Giao diện trang sửa khóa học

The screenshot shows the 'Sửa khóa học' (Edit Course) form. The sidebar is identical to the previous form. The main content area has the title 'Sửa khóa học' and the subtitle 'Cập nhật thông tin khóa học của bạn'. The form is a white card with the following fields: 'Tên khóa học' (Course Name) with the value 'Kỹ thuật lập trình'; 'Mô tả khóa học' (Course Description) with a placeholder 'Nhập mô tả khóa học'; and 'Năm học' (Academic Year) with a dropdown menu showing 'Năm nhất'. At the bottom of the form are two buttons: 'Lưu thay đổi' (Save Changes) in blue and 'Hủy' (Cancel) in gray.

Hình 4.8. Kết quả giao diện sửa khóa học

Trang sửa khóa học cho phép giảng viên chỉnh sửa ba thông tin: tên khóa học, mô tả, năm học.

4.3.2.5. Giao diện quản lý bài học

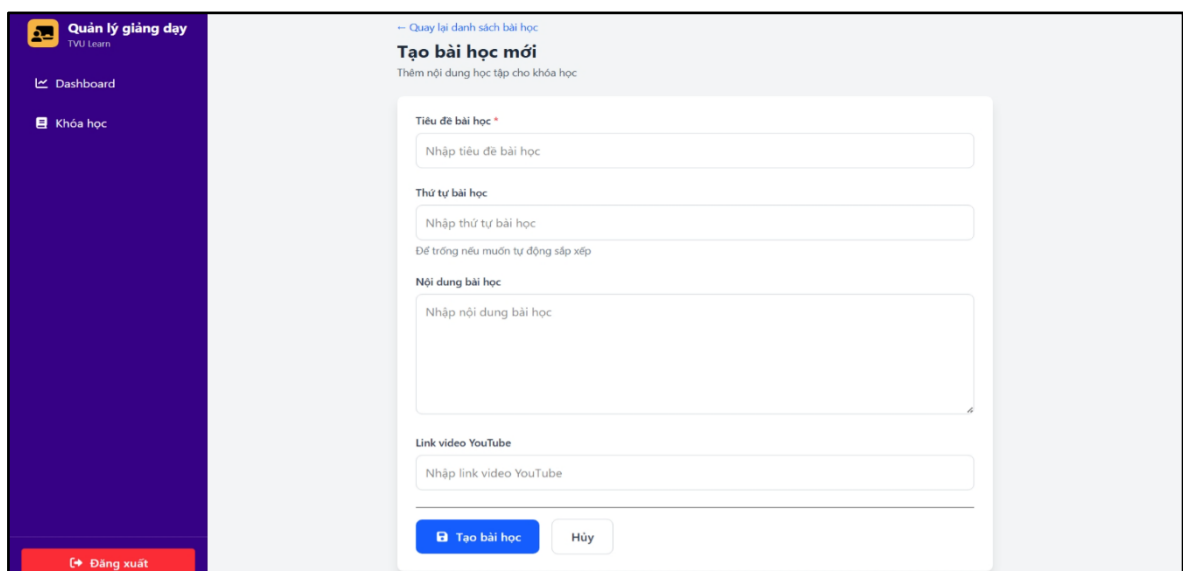


Hình 4.9. Kết quả giao diện trang quản lý bài học

Để vào trang quản lý bài học cần nhấp vào "bài học" trên khóa học muốn quản lý. Trang quản lý bài học gồm các chức năng:

- Hiện thị danh sách bài học: liệt kê các bài học thuộc khóa học.
- Tạo bài học mới: nút “Tạo bài học mới” để thêm bài học mới vào khóa học đang chọn.
- Quản lý từng bài học: quản lý quiz và chỉnh sửa bài học.

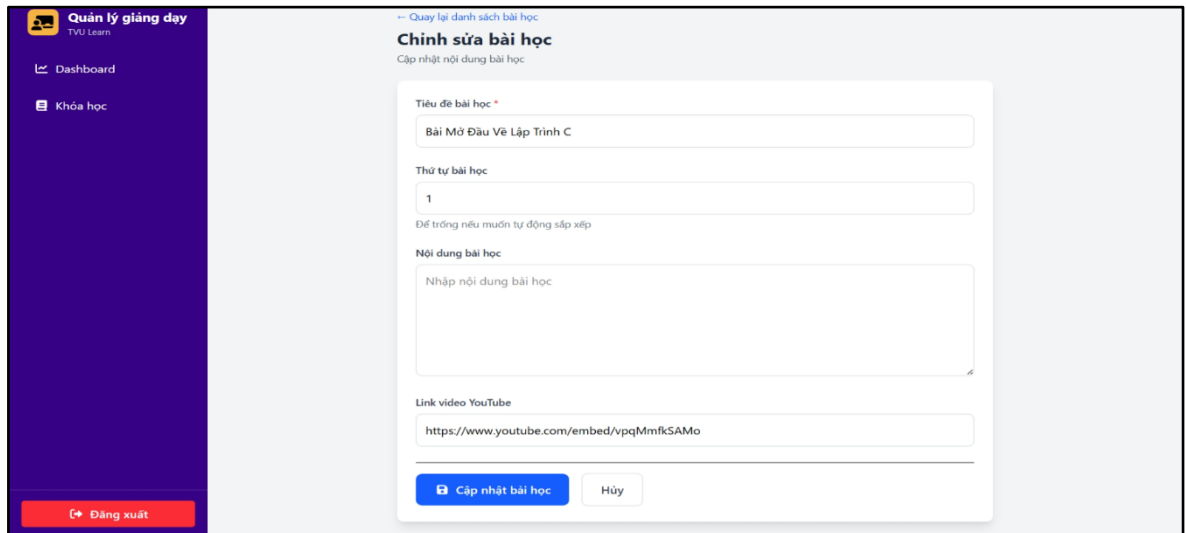
4.3.2.6. Giao diện trang tạo bài học



Hình 4.10. Kết quả giao diện trang tạo bài học

Để tạo bài học mới giảng viên cần: nhập thông tin bài học (tiêu đề bài học, thứ tự, nội dung bài học và nhập link video YouTube để nhúng video vào bài học. Sau đó nhấp nút “Tạo bài học” gửi dữ liệu lên hệ thống để thêm mới bài học vào khóa học.

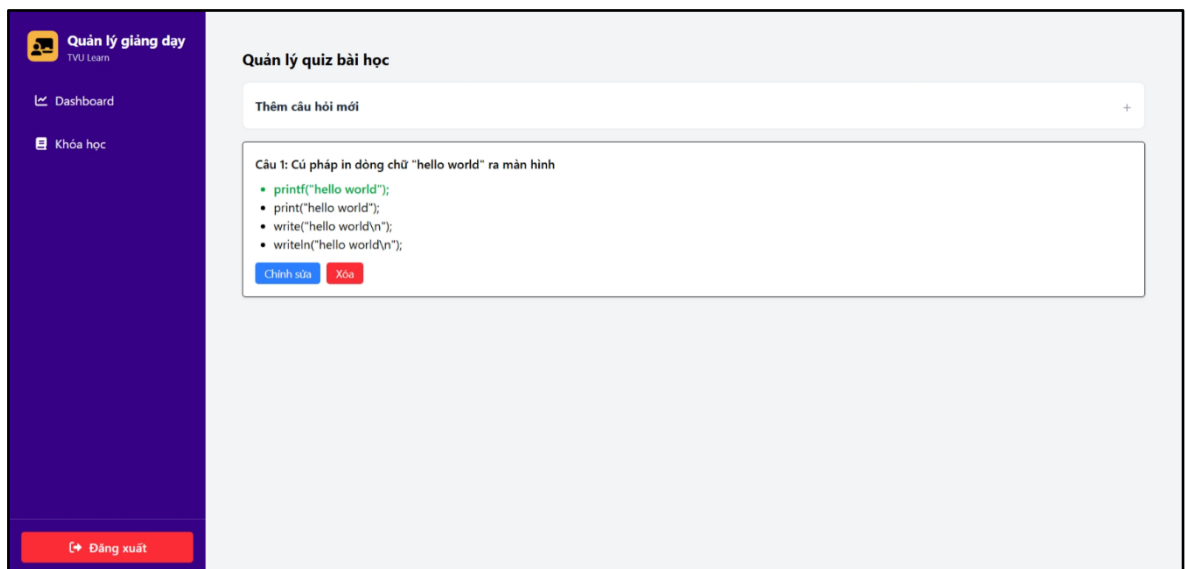
4.3.2.7. Giao diện trang sửa bài học



Hình 4.11. Kết quả giao diện sửa bài học

Trang sửa bài học cho phép giảng viên chỉnh sửa bốn thông tin: tiêu đề bài học, thứ tự bài học, đường dẫn video bài học.

4.3.2.8. Giao diện trang quản lý quiz



Hình 4.12. Kết quả giao diện trang quản lý quiz

Để vào trang quản lý quiz giảng viên cần nhấp vào “Quản lý quiz” tại bài học muốn quản lý. Trang quản lý quiz có các chức năng:

- Hiện thị danh sách các câu hỏi trắc nghiệm thuộc bài học đang chọn.

- Thêm câu hỏi mới: mục “Thêm câu hỏi mới” (nút +) cho phép tạo thêm câu hỏi trắc nghiệm cho bài học.

- Chỉnh sửa: nút “Chỉnh sửa” để cập nhật lại nội dung câu hỏi và các đáp án.

- Xóa câu hỏi: nút “Xóa” để xóa câu hỏi khỏi quiz của bài học.

4.3.2.9. Giao diện trang thêm và sửa quiz

Quản lý giảng dạy
TVU Learn

Dashboard
Khóa học

Đăng xuất

Quản lý quiz bài học

Thêm câu hỏi mới

Câu hỏi *

Nhập câu hỏi của bạn...

Các đáp án *

☒ Đáp án 1 ✓ Đúng

☐ Đáp án 2

☐ Đáp án 3

☐ Đáp án 4

Chọn radio button để đánh dấu đáp án đúng

Lưu câu hỏi Hủy

Câu 1: Cú pháp in dòng chữ "hello world" ra màn hình

- `printf("hello world");`
- `print("hello world");`
- `write("hello world\n");`
- `writeln("hello world\n");`

Chỉnh sửa Xóa

Hình 4.13. Kết quả giao diện thêm quiz

Trang thêm quiz gồm các chức năng:

- Thêm câu hỏi mới: cung cấp form nhập nội dung câu hỏi và 4 đáp án.

- Hiện thị danh sách câu hỏi đã tạo.

- Chỉnh sửa và xóa câu hỏi.

Quản lý giảng dạy
TVU Learn

Dashboard
Khóa học

Đăng xuất

Quản lý quiz bài học

Thêm câu hỏi mới

Câu hỏi

Cú pháp in dòng chữ "hello world" ra màn hình

Các đáp án

☒ `printf("hello world");`

☐ `print("hello world");`

☐ `write("hello world\n");`

☐ `writeln("hello world\n");`

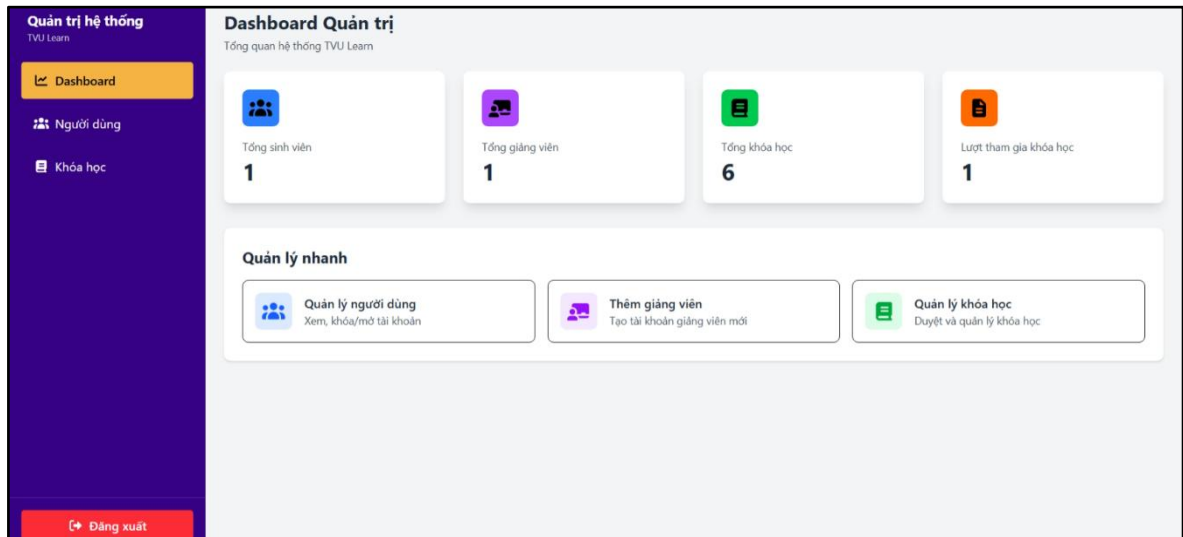
Lưu Hủy

Hình 4.14. Kết quả giao diện trang sửa quiz

Trang sửa quiz cho phép giảng viên sửa lại nội dung của câu hỏi và chỉnh sửa đáp án đúng của câu hỏi đó.

4.3.3. Giao diện cho quản trị viên

4.3.3.1. Giao diện trang dashboard quản trị

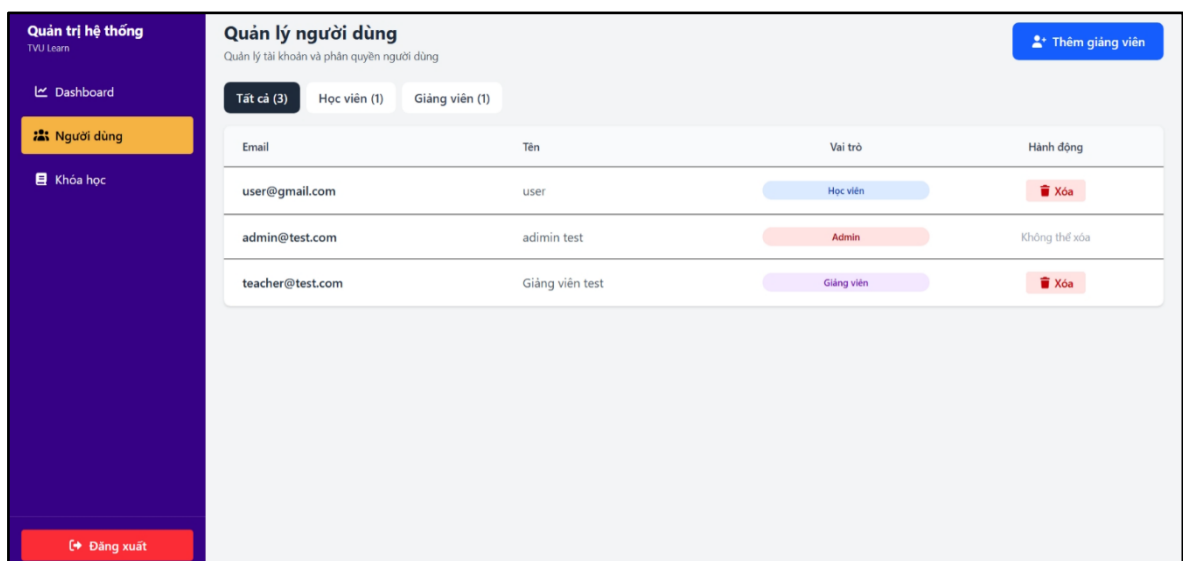


Hình 4.15. Kết quả giao diện trang dashboard quản trị

Các chức năng chính trên dashboard:

- Thống kê thông tin: hiển thị tổng quan số liệu về toàn bộ hệ thống.
- Quản lý người dùng: truy cập trang xem và quản lý tài khoản người dùng.
- Thêm giảng viên: chuyển đến form tạo tài khoản giảng viên mới.
- Quản lý khóa học: quản lý danh sách khóa học.

4.3.3.2. Giao diện trang quản lý người dùng

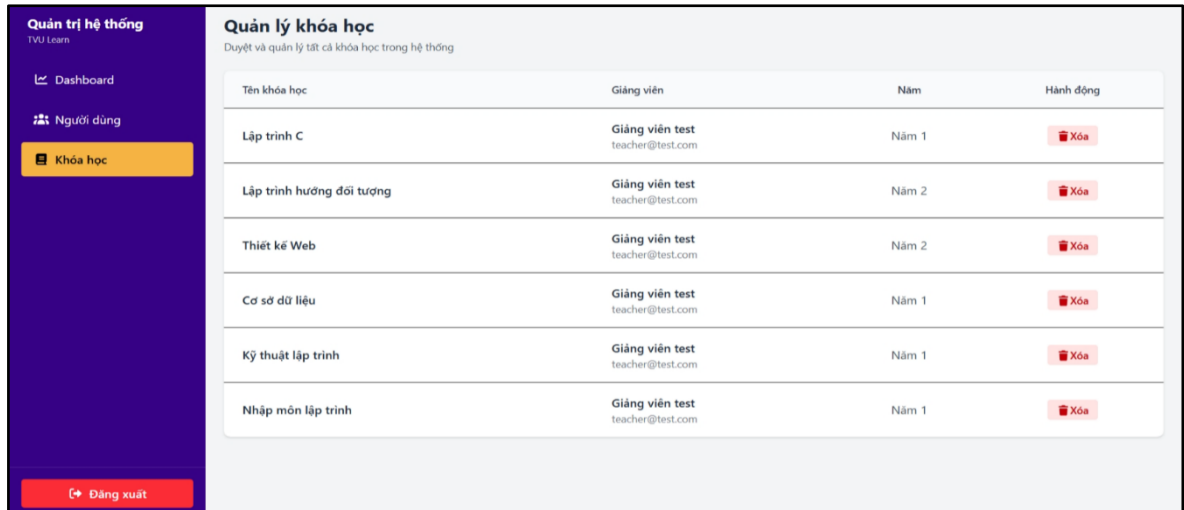


Hình 4.16. Kết quả giao diện trang quản lý người dùng

Trang quản lý người dùng gồm các chức năng:

- Xem danh sách người dùng: hiển thị danh sách tài khoản theo dạng bảng.
- Lọc người dùng theo vai trò: các tab tất cả / học viên / giảng viên.
- Thêm giảng viên: tạo tài khoản giảng viên mới cho hệ thống.
- Xóa tài khoản: nút “Xóa” cho phép admin xóa người dùng.

4.3.3.3. Giao diện trang quản lý khóa học

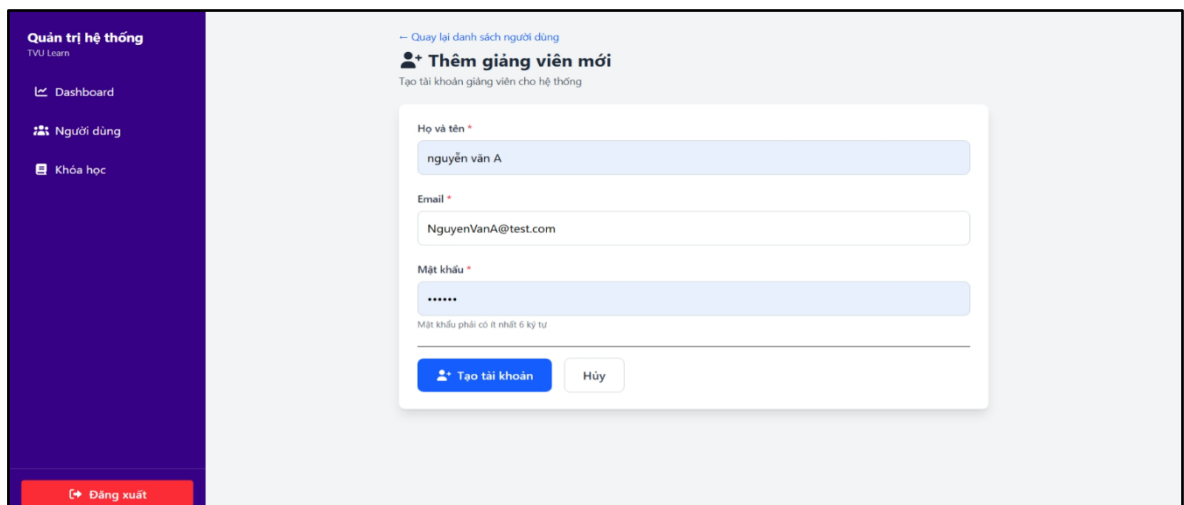


Hình 4.17. Kết quả giao diện trang quản lý khóa học

Trang quản lý khóa học gồm các chức năng:

- Xem danh sách khóa học: hiển thị toàn bộ khóa học trong hệ thống.
- Quản lý khóa học: giúp admin theo dõi khóa học đang tồn tại, thông tin giảng viên tạo/được gán và phân loại theo năm.
- Xóa khóa học: nút “Xóa” ở mỗi dòng cho phép admin xóa khóa học đã chọn.

4.3.3.4. Giao diện trang tạo giảng viên



Hình 4.18. Kết quả giao diện trang tạo giảng viên

Để tạo một giảng viên mới quản trị viên cần:

- Nhập thông tin giảng viên: giảng viên điền tên, email, mật khẩu.
- Tạo khóa học: nút “Tạo tài khoản” gửi dữ liệu lên hệ thống để tạo giảng viên.

4.4. Đánh giá hệ thống

4.4.1. Mức độ đáp ứng yêu cầu đề tài

Hệ thống website hỗ trợ học tập đã đáp ứng các mục tiêu và yêu cầu cơ bản của đề tài. Sinh viên có thể đăng ký, đăng nhập, tham gia khóa học, học theo từng chương và thực hiện các bài kiểm tra trắc nghiệm. Hệ thống cũng hỗ trợ giảng viên quản lý khóa học, bài học và xây dựng bài kiểm tra, đồng thời cho phép quản trị viên quản lý người dùng, khóa học và theo dõi dữ liệu tổng thể của hệ thống.

4.4.2. Đánh giá về giao diện và trải nghiệm người dùng

Giao diện hệ thống được thiết kế đơn giản, trực quan và dễ sử dụng, phù hợp với sinh viên và giảng viên ngành Công nghệ Thông tin. Các chức năng được bố trí hợp lý, giúp người dùng thao tác thuận tiện.

Việc sử dụng ReactJS kết hợp với Tailwind CSS giúp giao diện nhất quán, phản hồi nhanh và hiển thị tốt trên nhiều kích thước màn hình, mang lại trải nghiệm học tập tương đối hiệu quả.

4.4.3. Đánh giá về hiệu năng và tính ổn định

Hệ thống được kiểm thử trong môi trường cục bộ và hoạt động ổn định với thời gian phản hồi chấp nhận được. Tuy nhiên, khả năng chịu tải và tối ưu hiệu năng chưa được đánh giá do chưa triển khai trên môi trường thực tế.

4.4.4. Hạn chế của hệ thống

Bên cạnh những kết quả đạt được, hệ thống vẫn còn tồn tại một số hạn chế. Hiện tại, hệ thống mới chỉ được triển khai thử nghiệm và chưa được đưa lên máy chủ thực tế để phục vụ nhiều người dùng cùng lúc. Một số tính năng nâng cao như ứng dụng di động, thông báo thời gian thực, livestream bài giảng hay tích hợp trí tuệ nhân tạo cũng chưa được triển khai trong phạm vi đề tài. Bên cạnh đó, giao diện và trải nghiệm người dùng vẫn có thể tiếp tục được cải thiện để phù hợp hơn với nhiều đối tượng người dùng khác nhau.

CHƯƠNG 5. KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

5.1. Kết luận

Đề tài “Thiết kế Website hỗ trợ học tập cho sinh viên Công nghệ Thông tin Trường Đại học Trà Vinh” đã đáp ứng được các yêu cầu cơ bản đặt ra ban đầu. Hệ thống được xây dựng nhằm hỗ trợ sinh viên trong việc học tập, ôn luyện kiến thức các môn học chuyên ngành Công nghệ Thông tin thông qua nền tảng trực tuyến.

Website cung cấp đầy đủ các chức năng cần thiết như: đăng ký và đăng nhập tài khoản, tham gia khóa học, học bài học, làm bài kiểm tra trắc nghiệm, theo dõi tiến trình và kết quả học tập. Bên cạnh đó, hệ thống còn hỗ trợ giảng viên trong việc tạo và quản lý khóa học, bài học, quiz; đồng thời cung cấp các chức năng quản lý cơ bản cho quản trị viên.

Về mặt kỹ thuật, hệ thống được xây dựng theo mô hình Client – Server, sử dụng RESTful API, tách biệt Frontend và Backend, giúp hệ thống dễ mở rộng và bảo trì. Việc áp dụng cơ chế xác thực và phân quyền bằng JWT giúp đảm bảo an toàn dữ liệu và kiểm soát truy cập theo vai trò người dùng.

Nhìn chung, hệ thống hoạt động ổn định, đáp ứng tốt mục tiêu hỗ trợ học tập cho sinh viên Công nghệ Thông tin Trường Đại học Trà Vinh và có thể làm nền tảng để tiếp tục phát triển trong tương lai.

5.2. Hướng phát triển

Trong thời gian tới, hệ thống Website hỗ trợ học tập cho sinh viên Công nghệ Thông tin Trường Đại học Trà Vinh có thể tiếp tục được mở rộng và hoàn thiện thêm để nâng cao hiệu quả sử dụng và giá trị ứng dụng thực tế, cụ thể như sau:

- Bổ sung thêm các dạng bài tập và kiểm tra nâng cao như bài tập tự luận, bài tập nộp file, hoặc chấm điểm tự động kết hợp đánh giá của giảng viên.
- Cải thiện giao diện người dùng theo hướng thân thiện hơn trên thiết bị di động, tối ưu trải nghiệm người dùng khi truy cập bằng điện thoại hoặc máy tính bảng.
- Tích hợp thêm các tính năng thông báo thời gian thực (real-time) và nhắc nhở học tập nhằm tăng tính tương tác và hỗ trợ sinh viên học tập hiệu quả hơn.

Với các hướng phát triển trên, hệ thống có tiềm năng trở thành một nền tảng học tập trực tuyến hoàn chỉnh, hỗ trợ tốt cho hoạt động giảng dạy và học tập của sinh viên Trường Đại học Trà Vinh trong tương lai.

DANH MỤC TÀI LIỆU THAM KHẢO

- [1] A. Banks and E. Porcello, *Learning React: Modern Patterns for Developing React Apps*, 2nd ed., O'Reilly Media, Sebastopol, CA, USA, 2020.
- [2] A. De Quattro, *Guide to Tailwind CSS: Practical Guide*, Amazon Digital Services LLC, 2024.
- [3] The FusionAuth Team, “*Breaking down JSON Web Tokens: From pros and cons to building and revoking*,” FusionAuth, 2022. [Online]. Available: <https://fusionauth.io/resources/breaking-down-json-web-tokens.pdf>
- [4] A. Gupta and A. K. Sharma, “*Authentication & Authorization*,” International Journal of Engineering Research & Technology (IJERT), vol. 5, no. 03, Special Issue, ICADEMS Conference, 2017.
- [5] A. Mead, *Learning NodeJS Development*, Packt Publishing Ltd., Birmingham, U.K. & Mumbai, India, 2018.
- [6] Ngô Bá Hùng và Phạm Thế Phi, *Giáo trình Mạng máy tính*, Khoa Công Nghệ Thông Tin, Trường Đại học Cần Thơ, Cần Thơ, Việt Nam, 2005.
- [7] H. Subramanian and P. Raj, *Hands-On RESTful API Design Patterns and Best Practices: Design, Develop, and Deploy Highly Adaptable, Scalable, and Secure RESTful Web APIs*, Packt Publishing Ltd., 2019.

PHỤ LỤC 1. DANH SÁCH CÁC API CHÍNH CỦA HỆ THỐNG

Tên API	Phương thức	URL	Mô tả
Đăng nhập	POST	/api/auth/signin	Xác thực người dùng
Lấy danh sách khóa học	GET	/api/courses	Lấy tất cả khóa học
Chi tiết khóa học	GET	/api/courses/:id	Xem chi tiết khóa học
Ghi danh khóa học	POST	/api/enrollments/enroll	Sinh viên tham gia khóa học
Khóa học của tôi	GET	/api/enrollments/my-courses	Khóa học đã đăng ký
Chi tiết bài học	GET	/api/lessons/:lessonId	Xem nội dung bài học
Lấy quiz theo bài học	GET	/api/quiz/lesson/:lessonId	Lấy câu hỏi quiz
Nộp kết quả quiz	POST	/api/quiz/submit	Lưu kết quả quiz
Đánh dấu hoàn thành	POST	/api/progress/complete	Hoàn thành bài học
Thống kê người dùng	GET	/api/user/stats	Thống kê tiến trình học

PHỤ LỤC 2. TÀI KHOẢN DEMO & PHÂN QUYỀN

Vai trò	Email	Mật khẩu	Mục đích
Student	student@tvu.edu.vn	123456789	Kiểm thử luồng học
Teacher	teacher@tvu.edu.vn	123456789	Quản trị khóa học
Admin	admin@tvu.edu.vn	123456789	Quản trị hệ thống