

Đại học quốc gia thành phố Hồ Chí Minh
Trường đại học Bách khoa
Khoa Khoa học và kỹ thuật máy tính



Đồ án chuyên ngành
BÁO CÁO

**XÂY DỰNG HỆ THỐNG GIÁO DỤC
THÔNG MINH DỰA TRÊN MÔ HÌNH
NGÔN NGỮ LỚN**

Ngành: Khoa học máy tính

THESIS COMMITTEE:

- GVHD 1:** Assoc. Prof. Võ Thị Ngọc Châu, PhD
GVHD 2: Assoc. Prof. Nguyễn Hứa Phùng, PhD

—oo—

Sinh viên 1: Đỗ Phương Nam (2114111)

Sinh viên 2: Phan Phạm Thi (2114857)

Sinh viên 3: Nguyễn Trường Tuấn Anh (2112796)

Ho Chi Minh City, June 2024



PROTESTATION

Authors

Đỗ Phương Nam

Phan Phạm Thị

Nguyễn Trường Tuấn Anh



ACKNOWLEDGEMENTS

Authors

Đỗ Phương Nam

Phan Phạm Thị

Nguyễn Trường Tuấn Anh



Abstract

Contents

1	Giới thiệu về đề tài	12
1.1	Động lực	12
1.2	Mục tiêu	13
1.3	Phạm vi	13
1.4	Ý nghĩa của đề tài	14
1.4.1	Ý nghĩa của dự án dưới góc nhìn thực tiễn	14
1.4.2	Ý nghĩa của dự án theo góc nhìn khoa học	14
1.5	Cấu trúc báo cáo	14
2	Kiến thức nền tảng	16
2.1	Kiến thức nền tảng về giáo dục và giáo dục thông minh	16
2.1.1	Khái quát về giáo dục	16
2.1.2	Khái quát về giáo dục thông minh	17
2.2	Mô hình ngôn ngữ	17
2.3	Các thuật ngữ kỹ thuật liên quan đến giáo dục thông minh	18
2.3.1	Đồ thị tri thức (Knowledge graph)	18
2.3.2	Knowledge tracing	19
2.4	Kiến trúc công nghệ	20
2.4.1	Công cụ xử lý ngôn ngữ tự nhiên - Framework tích hợp LLMs	20
2.4.1.1	LangChain	20
2.4.1.2	LlamaIndex	21
2.4.1.3	Tổng quan	22
2.4.2	Hệ quản trị cơ sở dữ liệu đồ thị	23
2.4.2.1	Các công nghệ khác	23



2.4.2.2	So sánh giữa các công nghệ	23
2.4.2.3	Lựa chọn	23
2.4.3	Framework xây dựng giao diện người dùng	24
2.4.3.1	Các công nghệ khác	24
2.4.3.2	So sánh giữa các công nghệ	24
2.4.3.3	Lựa chọn	24
2.4.4	Framework phát triển API	25
2.4.4.1	Các công nghệ khác	25
2.4.4.2	So sánh giữa các công nghệ	25
2.4.4.3	Lựa chọn	25
2.5	Tổng kết	25
3	Công trình liên quan	27
3.1	Prompt engineering	27
3.2	Retrieval-Augmentation Generation	27
3.3	Knowledge tracing/Student Profiling	28
3.4	Hệ thống đề xuất lộ trình học	29
3.5	Kết luận	29
4	Hệ thống đề xuất	31
4.1	Mô tả hệ thống	31
4.2	Khảo sát yêu cầu	32
4.2.1	Người dùng	32
4.2.2	Yêu cầu chức năng	32
4.2.2.1	User Management	32
4.2.2.2	Learning Resource Recommendation System	33
4.2.2.3	Exercise Generator	33
4.2.2.4	AI Tutor	33
4.2.2.5	Progress Tracking	34
4.2.2.6	Course Management	34
4.2.2.7	Admin Management	35
4.2.3	Yêu cầu phi chức năng	35



4.2.4	Yêu cầu dữ liệu	37
4.2.4.1	Các thực thể và thuộc tính dữ liệu	37
4.2.4.2	Mối quan hệ dữ liệu	38
4.3	Kết luận	38
5	Phân tích và thiết kế	39
5.1	Phân tích	39
5.1.1	Business Process Modeling	39
5.1.2	Use case diagrams	40
5.1.2.1	Course Management	41
5.1.2.2	User Management	42
5.1.2.3	Learning Resource Recommendation System	43
5.1.2.4	Progress Tracking	44
5.1.2.5	Exercise Generator	46
5.1.2.6	AI Tutor	47
5.2	Thiết kế	49
5.2.1	Kiến trúc hệ thống	49
5.2.1.1	Phân tích kiến trúc hệ thống phổ biến hiện nay	49
5.2.1.2	Lựa chọn kiến trúc hệ thống	50
5.3	Database	54
5.3.1	EER diagram	54
5.3.2	Lược đồ CSDL quan hệ ánh xạ	56
5.3.3	Class diagram	57
5.3.4	Sitemap	58
5.3.4.1	Sinh viên	58
5.3.4.2	Giảng viên	59
5.3.4.3	Manager	60
5.4	Giải thuật	60
5.4.1	Giải thuật đề xuất bài học	60
5.4.2	Workflow của LLM và Function Calling	61
5.4.2.1	Phân tích Mục tiêu và Xác định Khái niệm	61



5.4.2.2	Function Calling để Trích xuất Dữ liệu	61
5.4.2.3	Trích xuất và Đè xuất Tài nguyên Học tập	62
5.4.2.4	Giải thích kết quả	62
5.5	Thiết kế giao diện	64
5.5.1	Landing Page	64
5.5.2	Authentication	65
5.5.2.1	Log In Screen	65
5.5.2.2	Forgot Password Screen	66
5.5.2.3	Reset Password Screen	66
5.5.3	Course Management (Student)	67
5.5.3.1	Courses List Screen	67
5.5.3.2	Detailed Course Screen	68
5.5.3.3	Define Goal Modal	70
5.5.3.4	Modules	70
5.5.3.5	Choices of Learning Styles	71
5.5.4	User Profile	74
5.5.4.1	Profile Information	74
5.5.4.2	Change Password, Avatar	74
5.5.5	Dashboard Screen	75
5.5.6	Exercise Practice	76
5.5.6.1	Code Exercise	76
5.5.7	AI Tutor	76
5.5.8	Progress Tracking	79
6	Hiện thực và kiểm thử	81
6.1	Hiện thực	81
6.1.1	Kiến trúc hệ thống	81
6.1.1.1	Frontend (VueJS với Vuetify và Tailwind)	81
6.1.1.2	Backend (FastAPI với DDD architecture)	81
6.1.2	Quá trình triển khai	82
6.1.2.1	Tạo cấu trúc cơ sở dữ liệu	82



6.1.2.2	Xây dựng các API	82
6.1.2.3	Quản lý bài học và đề xuất học tập	83
6.1.2.4	Giao diện người dùng	83
6.2	Kiểm thử	83
6.2.1	Tổng quan về kiểm thử	83
6.2.1.1	Loại kiểm thử thực hiện:	83
6.2.2	Kiểm thử API	84
6.2.2.1	Kịch bản Kiểm thử	84
6.2.2.1.a	1. Kiểm thử API Chào mừng (Welcome API)	84
6.2.2.1.b	2. Kiểm thử API Lấy Hoạt động (Get Activities API)	84
6.2.2.1.c	3. Kiểm thử API Lấy Danh sách Khóa học (Courses List API)	85
6.2.2.1.d	4. Kiểm thử API Lấy Chi tiết Khóa học (Get Course for Student API)	87
6.2.2.1.e	5. Kiểm thử API Lấy Danh sách Khóa học Đề xuất (Recommended Courses List API)	87
6.2.2.1.f	6. Kiểm thử API Thêm Hoạt động (Add Activity API)	88
6.2.2.2	Kết quả kiểm thử API	88
7	Tổng kết	89
7.1	Kết quả đạt được	89
7.2	Hướng phát triển	89

List of Figures

5.1	Activity diagram	39
5.2	Use case diagram	40
5.3	Detailed use case with id 1: Course Management	41
5.4	Detailed use case with id 2: User Management	42
5.5	Detailed use case with id 3: Adaptive Learning Path Management of Learner Actor	43
5.6	Detailed use case with id 3: Learning Resource Recommendation System of System Actor	43
5.7	Detailed use case with id 4: Progress Tracking of Learner Actor	44
5.8	Detailed use case with id 4: Progress Tracking of System Actor	44
5.9	Detailed use case with id 5: Exercise Generator	46
5.10	Detailed use case with id 6: AI Tutor of Learner Actor	47
5.11	Detailed use case with id 6: AI Tutor of System Actor	47
5.12	Component-based Diagram cho toàn bộ hệ thống	53
5.13	Component-based Diagram cho những công nghệ được chọn	53
5.14	Sơ đồ EER của hệ thống	54
5.15	Lược đồ CSDL quan hệ ánh xạ của hệ thống	56
5.16	Class diagram hiện thực của hệ thống	57
5.17	Sitemap cho đối tượng Sinh viên	58
5.18	Sitemap cho đối tượng Giảng viên	59
5.19	Sitemap cho đối tượng Quản trị viên	60
5.20	Landing Page	64
5.21	Log In Screen	65
5.22	Forgot Password Screen	66



5.23 Reset Password Screen	66
5.24 Courses List Screen	67
5.25 Modal Learning Outcomes	67
5.26 Detailed Course Screen - Description	68
5.27 Detailed Course Page - Lessons	68
5.28 Detailed Course Page - Exercises	69
5.29 Detailed Course Page - Recommend Lessons	69
5.30 Modal để lấy mục tiêu của sinh viên	70
5.31 Chi tiết Lesson	71
5.32 Modal cho phép sinh viên lựa chọn kiểu học phù hợp	71
5.33 Giao diện khi sinh viên chọn học theo cách học làm Quiz với tiến độ tự do	72
5.34 Giao diện khi sinh viên chọn học theo cách học đọc tài liệu với tiến độ tự do	73
5.35 Giao diện khi sinh viên chọn học theo cách học làm bài tập Coding với tiến độ tự do	73
5.38 Dashboard - Student	75
5.39 Giao diện thực hành code	76
5.40 Lịch sử submission	76
5.41 Các tính năng của AI Tutor	77
5.42 Giải thích từng dòng code	77
5.43 Hỗ trợ sửa lỗi	78
5.44 Cung cấp gợi ý cho bài tập	78
5.45 Progress Tracking Page	79
6.1 API endpoints	82

List of Tables

2.1	So sánh giữa LangChain và LlamaIndex	22
7.1	Kế hoạch 15 Tuần	90

Chapter 1

Giới thiệu về đề tài

1.1 Động lực

Trong bối cảnh toàn cầu hóa và sự phát triển không ngừng của công nghệ, việc nâng cao chất lượng giáo dục, đặc biệt trong lĩnh vực lập trình, ngày càng trở nên quan trọng. Một trong những thách thức lớn hiện nay là làm thế nào để cá nhân hóa quá trình học lập trình, đáp ứng sự đa dạng trong nhu cầu và khả năng của từng học viên. Các hệ thống giáo dục truyền thống thường gặp khó khăn trong việc tạo ra lộ trình học phù hợp với từng học viên, đặc biệt là trong lập trình, nơi mà khả năng tư duy và phong cách học của mỗi học viên đều khác nhau.

Sự phát triển của công nghệ trí tuệ nhân tạo (AI) và mô hình ngôn ngữ lớn (LLM) đang mở ra cơ hội lớn để xây dựng các hệ thống dạy học lập trình hiệu quả hơn. LLM, với khả năng hiểu và sinh ngôn ngữ tự nhiên, không chỉ giúp học viên giải quyết các vấn đề lập trình nhanh chóng mà còn hỗ trợ cá nhân hóa học tập. LLM có thể cung cấp tài liệu học tập và bài giảng được tùy chỉnh, cũng như hỗ trợ giải quyết bài tập lập trình, từ đó cải thiện hiệu quả học tập.

Tuy nhiên, sự phát triển mạnh mẽ của AI cũng gây ra mối lo ngại về việc sinh viên sử dụng AI để làm bài tập mà không thực sự học hỏi. Điều này đặt ra câu hỏi liệu AI có thể giúp sinh viên tự tìm ra lời giải cho các vấn đề thay vì đưa ra câu trả lời trực tiếp, giúp họ phát triển kỹ năng tư duy giải quyết vấn đề một cách hiệu quả.

Việc chọn giáo dục lập trình làm lĩnh vực nghiên cứu và ứng dụng hệ thống có hai lý do chính:

- Thứ nhất, dạy học lập trình ngày càng trở nên phổ biến trong thời đại internet, giúp học viên phát triển tư duy logic và khả năng giải quyết vấn đề, vì nó yêu cầu phân tích và tư duy hệ thống.
- Thứ hai, lĩnh vực giáo dục lập trình rất gần gũi với ngành khoa học và kỹ thuật máy tính, nơi mà nhóm nghiên cứu đang theo học, tạo ra sự liên kết thực tiễn giữa nghiên cứu và



ứng dụng trong thực tế.

1.2 Mục tiêu

Từ động lực đã nêu, mục tiêu của đề tài này gồm các điểm chính như sau:

- **Xây dựng hệ thống học tập cá nhân hóa:** Hệ thống sẽ tập trung vào việc cá nhân hóa trải nghiệm học cho từng học viên, phù hợp với khả năng, trình độ, và nhu cầu học tập riêng biệt. Mỗi học viên có một lộ trình học và cách giải thích riêng, giúp họ tiến bộ một cách hiệu quả nhất. Với khả năng phân tích và hiểu ngữ cảnh của LLM, hệ thống có thể đưa ra các phản hồi phù hợp với nhu cầu học tập của từng học viên.
- **Xác định tính khả thi của việc tích hợp LLM trong giáo dục:** Đánh giá khả năng tích hợp mô hình ngôn ngữ lớn vào hệ thống giáo dục, đặc biệt là trong bối cảnh giáo dục lập trình. Chúng tôi sẽ nghiên cứu tính hiệu quả của LLM trong việc cá nhân hóa và nâng cao khả năng tiếp thu kiến thức của học viên.
- **Giải quyết vấn đề lạm dụng LLM trong giải bài tập:** Hiện nay, sinh viên có thể lợi dụng LLM để giải bài tập lập trình mà không thực sự học. Mục tiêu của đề tài là nghiên cứu xem liệu có thể khiến LLM không trực tiếp đưa ra lời giải cho học sinh mà thay vào đó là cung cấp các câu hỏi gợi mở hoặc hướng dẫn giúp học viên tự giải quyết vấn đề. Điều này nhằm phát triển tư duy giải quyết vấn đề của học viên, thay vì chỉ đưa ra câu trả lời.

1.3 Phạm vi

Đề tài này sẽ được thực hiện trong khoảng thời gian 7-8 tháng, bao gồm hai học kỳ tại trường đại học. Phạm vi nghiên cứu chủ yếu tập trung vào lĩnh vực giáo dục công nghệ thông tin, đặc biệt là trong việc giảng dạy lập trình. Cụ thể, hệ thống sẽ được xây dựng để hỗ trợ việc học lập trình cho sinh viên, thông qua việc cá nhân hóa các bài giảng, hướng dẫn và hỗ trợ học tập dựa trên các mô hình ngôn ngữ lớn.

Về mặt công nghệ, nhóm thực hiện đề tài sẽ không xây dựng lại các mô hình ngôn ngữ lớn từ đầu mà sẽ tận dụng các mô hình hiện có, chẳng hạn như GPT, Gemini, hoặc các mô hình tương tự, để khai thác và tùy chỉnh chúng nhằm đáp ứng các yêu cầu cụ thể của hệ thống. Mô hình ngôn ngữ này sẽ được sử dụng để cung cấp các phản hồi tự động, đề xuất bài học, tạo các bài kiểm tra và giúp cá nhân hóa trải nghiệm học lập trình của sinh viên.

Mặc dù hệ thống có thể áp dụng cho các lĩnh vực giáo dục khác, trong phạm vi của đề tài này, mục tiêu chính là tập trung vào việc ứng dụng trong lĩnh vực lập trình và công nghệ thông tin, nhằm hỗ trợ việc giảng dạy và học tập môn học này.



1.4 Ý nghĩa của đề tài

1.4.1 Ý nghĩa của dự án dưới góc nhìn thực tiễn

- Dự án này mang lại ý nghĩa thực tiễn quan trọng trong việc áp dụng công nghệ AI vào giáo dục, cụ thể là trong việc cá nhân hóa quá trình học tập của học viên. Nhờ vào AI và LLM, hệ thống học tập có thể tạo ra một trải nghiệm học tập hấp dẫn và hiệu quả hơn cho học viên, từ đó giúp tăng động lực học tập.
- AI có thể đóng vai trò như một trợ giảng 1-1, giúp giảm tải áp lực cho giáo viên. AI sẽ hỗ trợ trả lời các câu hỏi cơ bản, đánh giá và chấm điểm tự động cho các bài tập, giúp giáo viên có thể tập trung vào việc hỗ trợ học viên ở mức độ sâu hơn. Điều này không chỉ cải thiện hiệu quả giảng dạy mà còn tạo ra một môi trường học tập linh hoạt và phù hợp với từng cá nhân học viên.

1.4.2 Ý nghĩa của dự án theo góc nhìn khoa học

- Dự án này có ý nghĩa khoa học đáng kể trong việc ứng dụng AI và LLM vào giáo dục, đặc biệt là trong giáo dục thông minh. Việc tích hợp LLM sẽ không chỉ mở rộng khả năng của các hệ thống dạy học mà còn đánh giá được hiệu quả của chúng trong việc nâng cao khả năng giải quyết vấn đề và học lập trình.
- Dự án sẽ chứng minh tính ứng dụng của LLM trong việc thực hiện các tác vụ lập trình và giải quyết vấn đề, đồng thời khai thác các lý thuyết giáo dục trong khoa học nhận thức, như lý thuyết tải nhận thức (cognitive load theory) và học thuyết kiến tạo (constructivism). Việc áp dụng LLM vào giáo dục có thể giúp học viên không chỉ tiếp thu kiến thức mà còn phát triển khả năng tự học và tư duy phản biện.

1.5 Cấu trúc báo cáo

Báo cáo này được tổ chức thành 8 chương chính, mỗi chương sẽ trình bày một phần quan trọng trong quá trình thực hiện và phát triển hệ thống. Cấu trúc báo cáo như sau:

- **Chương 1 – Giới thiệu:** Cung cấp cái nhìn tổng quan về động cơ, mục tiêu và phạm vi của đề tài, cũng như ý nghĩa của dự án cả trong thực tiễn và trong khoa học. Giới thiệu về cấu trúc báo cáo để hiểu rõ cách thức bố trí và tiếp cận các phần nội dung.
- **Chương 2 – Kiến thức và công nghệ nền tảng:** Trình bày các kiến thức cơ bản và công nghệ nền tảng cần thiết để phát triển hệ thống, bao gồm các khái niệm, lý thuyết và công nghệ được sử dụng trong hệ thống đề xuất.



- **Chương 3 – Công trình liên quan:** Phân tích các nghiên cứu, công trình và hệ thống liên quan đã có trước đây, chỉ ra sự tương đồng và khác biệt với công trình của đề tài này, từ đó xác định các khoảng trống và cơ hội nghiên cứu thêm.
- **Chương 4 – Hệ thống đề xuất:** Mô tả chi tiết hệ thống được đề xuất, bao gồm các mục tiêu, chức năng của hệ thống, người dùng mục tiêu và các yêu cầu kỹ thuật, dữ liệu liên quan.
- **Chương 5 – Phân tích và Thiết kế:** Trình bày quá trình phân tích hệ thống và thiết kế các thành phần của nó, bao gồm mô hình hóa quy trình nghiệp vụ, các giải pháp công nghệ, thiết kế hệ thống, cơ sở dữ liệu và giao diện người dùng.
- **Chương 6 – Hiện thực và kiểm thử:** Mô tả quá trình triển khai thực tế hệ thống và các kiểm thử được thực hiện để đảm bảo hệ thống hoạt động chính xác và hiệu quả.
- **Chương 7 – Đánh giá hệ thống:** Đánh giá hiệu quả và hiệu suất của hệ thống, chỉ ra các ưu điểm, nhược điểm và khả năng cải tiến.
- **Chương 8 – Tổng kết:** Tóm tắt kết quả đạt được, các hạn chế của hệ thống hiện tại và hướng phát triển trong tương lai.

Chapter 2

Kiến thức nền tảng

2.1 Kiến thức nền tảng về giáo dục và giáo dục thông minh

2.1.1 Khái quát về giáo dục

Trong phần này, ta sẽ giới thiệu khái niệm cơ bản về giáo dục. Giáo dục là quá trình truyền đạt và tiếp thu kiến thức, kỹ năng và giá trị, giúp con người phát triển toàn diện về trí tuệ, thể chất và đạo đức. Mục tiêu chính của giáo dục là trang bị cho con người những kiến thức cơ bản cần thiết để hòa nhập vào xã hội, đồng thời phát triển các kỹ năng tư duy, giải quyết vấn đề và sáng tạo.

Bên cạnh đó, giáo dục còn hướng đến việc bồi dưỡng phẩm chất đạo đức, khuyến khích mỗi cá nhân phát triển tiềm năng riêng biệt của mình. Đây chính là nền tảng để mỗi người có thể xây dựng sự nghiệp và cuộc sống thành công.

Trong bối cảnh thế giới không ngừng thay đổi, giáo dục ngày nay không chỉ cung cấp kiến thức mà còn giúp con người thích nghi với những thách thức mới, phát triển khả năng học tập suốt đời và duy trì sự phát triển liên tục qua mọi giai đoạn cuộc sống.

Về lý thuyết giáo dục, có nhiều học thuyết liên quan đến lĩnh vực này, kết hợp từ nhiều lĩnh vực khác nhau như tâm lý học và thần kinh học:

- *Học tập kiến tạo (Constructivism)*: Phương pháp học tập này cho rằng con người tự xây dựng sự hiểu biết và kiến thức của mình thông qua trải nghiệm và việc phản ánh lại các trải nghiệm đó. Thay vì tiếp thu thông tin một cách thụ động, người học chủ động tạo ra ý nghĩa thông qua việc liên kết thông tin mới với kiến thức đã có trước đó.
- *Thuyết tải nhận thức (Cognitive load theory)*: Được phát triển bởi John Sweller vào những năm 1980, lý thuyết này tập trung vào cách bộ não xử lý và lưu trữ thông tin trong quá



trình học tập, đặc biệt là những yếu tố ảnh hưởng đến hiệu quả học tập. Thuyết này nhấn mạnh rằng bộ não con người có giới hạn về lượng thông tin có thể xử lý cùng lúc trong trí nhớ ngắn hạn.

2.1.2 Khái quát về giáo dục thông minh

Tuy không có một định nghĩa chính thức và rõ ràng cho thuật ngữ “**giáo dục thông minh**”, trong báo cáo này, thuật ngữ “**thông minh**” được hiểu là cách thức thực hiện mọi hoạt động một cách hiệu quả và hữu ích. Mục tiêu của giáo dục thông minh là bồi dưỡng con người trên nhiều mặt, từ kiến thức đến đạo đức, với phương thức giúp học sinh tiếp thu kiến thức một cách hiệu quả nhất có thể.

Giáo dục thông minh không chỉ liên quan đến việc dạy và học mà còn phải thích ứng với nhu cầu, khả năng, sở thích và điểm mạnh, yếu của từng học sinh. Việc cá nhân hóa quá trình học tập đóng vai trò quan trọng, vì mỗi học sinh có cách tiếp thu khác nhau. Công nghệ trí tuệ nhân tạo (AI) hiện nay đã tạo ra nhiều cơ hội để phát triển giáo dục thông minh, đặc biệt là qua phương pháp học tập cá nhân hóa (personalized learning), nơi mà mỗi học sinh có thể học theo một tốc độ, phong cách và nội dung phù hợp với bản thân.

Học tập cá nhân hóa nhằm tối đa hóa hiệu quả học tập bằng cách điều chỉnh chương trình học và phương pháp giảng dạy để phù hợp với nhu cầu và mục tiêu riêng biệt của mỗi học sinh, thay vì áp dụng một chương trình chung cho tất cả.

2.2 Mô hình ngôn ngữ

Mô hình ngôn ngữ (*language model*) có thể được hiểu ở nhiều góc độ khác nhau, nhưng nhìn chung, một mô hình ngôn ngữ là một hệ thống được thiết kế để hiểu và tạo ra ngôn ngữ tự nhiên của con người. Trong lĩnh vực machine learning, mô hình ngôn ngữ được định nghĩa rõ ràng hơn là một mô hình xác suất của một ngôn ngữ tự nhiên, có khả năng dự đoán một từ hoặc một chuỗi từ dựa trên những từ đã xuất hiện trước đó.

Trong lịch sử phát triển của lĩnh vực xử lý ngôn ngữ tự nhiên (NLP), đã có nhiều loại mô hình ngôn ngữ khác nhau. Dưới đây là các loại mô hình ngôn ngữ chính:

- **Mô hình ngôn ngữ xác suất:** Là loại mô hình ngôn ngữ là trong đó, mỗi một chuỗi từ ngữ nhất định sẽ được gán một xác suất cụ thể. Bài toán chính mà dạng mô hình này giải quyết đó là tính *khả năng xảy ra* (*likelihood*) của một chuỗi các từ vựng bất kì, câu hỏi đặt ra thường là "Liệu một chuỗi các từ w_1, w_2, \dots, w_n có khả năng xuất hiện là bao nhiêu?". Một ví dụ điển hình cho dạng mô hình ngôn ngữ xác suất đó là mô hình ngôn ngữ n-gram. Theo mô hình ngôn ngữ này, thì xác suất của một từ xuất hiện phụ thuộc vào các từ trước



đó:

$$P(w_1, w_2, \dots, w_n) = \prod_{i=1}^n P(w_i | w_{i-(n-1)}, \dots, w_{i-1})$$

- **Mô hình ngôn ngữ dựa trên Neural network:** là một loại mô hình ngôn ngữ sử dụng neural network để hiểu và sinh ra ngôn ngữ tự nhiên của con người. Sức mạnh của neural network nằm ở chỗ là nó có thể nhận diện được các pattern phức tạp, do đó, mô hình dạng này giúp thực hiện nhiều nhiệm vụ xử lý ngôn ngữ tự nhiên (NLP) một cách hiệu quả hơn so với các mô hình thống kê truyền thống đã nói ở trên.
- **Mô hình ngôn ngữ lớn:** Dựa trên khái niệm về mô hình ngôn ngữ, ta có khái niệm cho *Mô hình ngôn ngữ lớn (Large language model)*. Mô hình ngôn ngữ lớn là gì? Nó khác gì so với một mô hình ngôn ngữ? Làm sao để biết được một mô hình ngôn ngữ là 'lớn'? Một mô hình ngôn ngữ lớn là một mô hình ngôn ngữ dựa trên *Deep neural network* và được huấn luyện bằng một tập dữ liệu ngôn ngữ khổng lồ. Việc đánh giá một mô hình là to hay nhỏ có nhiều khía cạnh, có thể là quy mô của mô hình hoặc là khả năng. Về quy mô, mô hình ngôn ngữ lớn, đúng như tên gọi, có kích thước rất lớn, với số lượng tham số lên đến con số hàng tỉ, hoặc hàng trăm tỉ. Một ví dụ tiêu biểu có thể kể đến như mô hình GPT-3 của OpenAI có khoảng 175 tỉ tham số [1]. Một mô hình ngôn ngữ lớn có thể thực hiện được nhiều tác vụ hơn, liên quan đến suy luận, giải quyết vấn đề, những thứ mà các mô hình ngôn ngữ truyền thống gần như không thể thực hiện được.

2.3 Các thuật ngữ kỹ thuật liên quan đến giáo dục thông minh

2.3.1 Đồ thị tri thức (Knowledge graph)

Trước khi nói về thuật ngữ *Đồ thị tri thức*, ta hãy nhìn sơ qua về thuật ngữ *Đồ thị*. Ở đây, nghĩa của từ đồ thị được hiểu theo định nghĩa của nó ở trong *Lý thuyết đồ thị (graph theory)* là một nhánh nghiên cứu của *Toán rời rạc (Discrete mathematics)*: Một đồ thị (graph) là một cấu trúc toán học được sử dụng để mô tả mối quan hệ giữa các đối tượng (được gọi là đỉnh hoặc nút). Các đối tượng này được kết nối với nhau thông qua các cạnh (hay còn gọi là cung) biểu diễn mối quan hệ giữa chúng. Từ định nghĩa của đồ thị, Trong bài báo cáo này, nhóm xin đưa ra định nghĩa của *Đồ thị tri thức (knowledge graph)*: là một dạng thể hiện có cấu trúc, cụ thể hơn là thể hiện bằng đồ thị, trong đó các đỉnh là các thực thể trong thế giới thực như: một đối tượng, một khái niệm, một sự kiện, và các cạnh là mối quan hệ giữa chúng[2].



2.3.2 Knowledge tracing

Knowledge tracing là bài toán mô hình hóa kiến thức của học sinh theo thời gian để ta có thể dự đoán chính xác cách học sinh sẽ thực hiện trong các tương tác sau này [3]. Nhờ có *Knowledge tracing*, giảng viên/người giảng dạy/hệ thống có thể kiểm tra được tiến độ học tập cũng như năng lực của mỗi học sinh *một cách hiệu quả hơn* bằng cách theo dõi tương tác của học viên với các học liệu trực tuyến (tài liệu, các câu hỏi, bài kiểm tra,...)[4].



2.4 Kiến trúc công nghệ

2.4.1 Công cụ xử lý ngôn ngữ tự nhiên - Framework tích hợp LLMs

Trong quá trình phát triển một hệ thống học tập trực tuyến thông minh dựa trên mô hình ngôn ngữ lớn (LLMs), việc lựa chọn công cụ xử lý ngôn ngữ tự nhiên phù hợp là một yếu tố quyết định đến hiệu quả và tính linh hoạt của hệ thống. Với mục tiêu tối ưu hóa trải nghiệm học tập cá nhân hóa, hệ thống cần tích hợp một framework mạnh mẽ để khai thác và xử lý dữ liệu từ nhiều nguồn. Hai công cụ nổi bật trong lĩnh vực này là LangChain và LlamaIndex, đều được thiết kế để hỗ trợ xây dựng các ứng dụng sử dụng LLMs. Báo cáo này sẽ phân tích và so sánh hai framework trên, nhằm xác định công cụ phù hợp nhất cho hệ thống học tập thông minh, dựa trên tính năng, khả năng tích hợp, và yếu tố chi phí.

2.4.1.1 LangChain

Langchain là một thư viện Python được thiết kế để hỗ trợ phát triển các ứng dụng dựa trên mô hình ngôn ngữ lớn (LLMs).

Các tính năng chính:

- Tích hợp dễ dàng với nhiều LLMs
 - Dễ dàng tích hợp và chuyển đổi giữa các mô hình ngôn ngữ lớn khác nhau như **GPT-3**, **GPT-4**, **BERT**, hoặc các mô hình tùy chỉnh.
 - * VD: Sử dụng **GPT-4** cho việc tạo nội dung học tập phức tạp, trong khi sử dụng một mô hình nhẹ hơn như **BERT** cho việc phân tích nhanh câu trả lời của học viên
 - Chuỗi xử lý linh hoạt cho các tác vụ phức tạp
 - * Cho phép xây dựng các chuỗi xử lý (chains) phức tạp, kết hợp nhiều bước xử lý khác nhau.
 - * VD: Trong hệ thống gia sư AI, tạo một chuỗi xử lý để phân tích code của học viên, đánh giá, và đưa ra gợi ý cải thiện.
 - Hỗ trợ truy xuất thông tin và tìm kiếm ngữ nghĩa
 - Cung cấp các công cụ để truy xuất thông tin dựa trên ngữ nghĩa, giúp tìm kiếm thông tin liên quan một cách hiệu quả
 - * VD: Khi học viên đặt câu hỏi, hệ thống có thể tìm kiếm thông tin liên quan từ cơ sở dữ liệu bài giảng bằng cách dùng **FAISS (Facebook AI Similarity Search)** - Một thư viện hỗ trợ sử dụng vector search để tìm kiếm thông tin trong **Knowledge Base**.



- Khả năng xử lý đa dạng các loại dữ liệu
 - Có thể làm việc với nhiều loại dữ liệu khác nhau như văn bản, PDF, hình ảnh (qua tích hợp với các mô hình xử lý hình ảnh).
 - * VD: Xử lý các tài liệu PDF và chuyển đổi chúng thành nội dung học tập.

2.4.1.2 LlamaIndex

LlamaIndex là một framework chuyên biệt cho việc xây dựng các ứng dụng AI với khả năng truy xuất dữ liệu mạnh mẽ.

Các tính năng chính:

- Indexing và truy xuất dữ liệu hiệu quả
 - Chuyên về việc tạo index cho dữ liệu lớn và truy xuất thông tin một cách nhanh chóng và hiệu quả.
 - * VD: Indexing toàn bộ nội dung khóa học và truy xuất thông tin liên quan.
 - Quá trình Indexing trong LlamaIndex như sau:
 - * Data Connectors: Đây là các công cụ nhập dữ liệu từ nhiều nguồn khác nhau như API, PDF, cơ sở dữ liệu, hay các ứng dụng bên ngoài (Gmail, Notion, Airtable). Chúng chịu trách nhiệm thu thập dữ liệu và đưa vào hệ thống dưới dạng tài liệu thống nhất.
 - * Documents/Nodes: Documents là các container chứa dữ liệu từ các nguồn khác nhau, chẳng hạn như từ PDF, API hoặc cơ sở dữ liệu. Nodes là những phần nhỏ của tài liệu, được bổ sung metadata và các mối quan hệ, giúp tăng độ chính xác trong quá trình truy xuất dữ liệu.
 - * Data Indexes: Sau khi dữ liệu được nhập vào, LlamaIndex sẽ giúp tổ chức dữ liệu thành một định dạng có thể truy xuất. Quá trình này bao gồm việc phân tích, tạo các biểu diễn embedding, và suy luận metadata, tạo thành kho kiến thức để sử dụng sau này.
- Tích hợp với nhiều nguồn dữ liệu
 - LlamaIndex có thể làm việc với nhiều nguồn dữ liệu khác nhau như file local, API, cơ sở dữ liệu.
 - * VD: Tích hợp dữ liệu từ nhiều nguồn để tạo nội dung khóa học.
- Hỗ trợ cho các truy vấn phức tạp



- LlamaIndex cho phép thực hiện các truy vấn phức tạp, kết hợp nhiều điều kiện và lọc kết quả.
 - * VD: Tìm kiếm bài học phù hợp dựa trên nhiều tiêu chí.

2.4.1.3 Tổng quan

Tiêu chí	LangChain	LlamaIndex
Overall	Phát triển ứng dụng phức tạp linh hoạt, tích hợp LLM	Tập trung vào tìm kiếm và truy xuất thông tin từ tập dữ liệu lớn một cách nhanh chóng và chính xác
Prompts	Hỗ trợ giao diện tiêu chuẩn cho việc tạo và quản lý prompts, giúp tùy chỉnh và sử dụng lại dễ dàng hơn trên các mô hình khác	Không hỗ trợ chi tiết
Chains	Cung cấp giao diện mạnh mẽ để xây dựng và quản lý chuỗi, cùng với nhiều thành phần có thể tái sử dụng	Không hỗ trợ chuỗi xử lý phức tạp
Agents	Sử dụng LLM để xác định và thực hiện các hành động dựa trên input	Không có cơ chế agent
Data Indexing	Phương pháp indexing qua các chuỗi phức tạp	Lập chỉ mục nhanh chóng các dữ liệu không cấu trúc
Customization	Tùy chỉnh cao cho các workflow và ứng dụng phức tạp	Tùy chỉnh hạn chế, tập trung vào lập chỉ mục và tìm kiếm
Context Retention	Lưu giữ thông tin từ các tương tác trước đó để cho phép các cuộc hội thoại có nhận thức về ngữ cảnh và mạch lạc	Lưu giữ ngữ cảnh cơ bản, phù hợp với nhiệm vụ tìm kiếm
Use Cases	Phù hợp cho các ứng dụng như chatbot, hỗ trợ khách hàng, tạo nội dung phức tạp	Phù hợp cho hệ thống tìm kiếm nội bộ, quản lý kiến thức
Performance	Xử lý tốt các cấu trúc dữ liệu phức tạp	Tối ưu cho tốc độ và độ chính xác trong truy xuất thông tin
Lifecycle Management	Cung cấp bộ đánh giá LangSmith để kiểm tra và gỡ lỗi ứng dụng LLM	Tích hợp công cụ gỡ lỗi và giám sát cho hiệu suất và độ tin cậy

Table 2.1: So sánh giữa LangChain và LlamaIndex

Với những ưu điểm trên, **LangChain** là một lựa chọn tối ưu cho hệ thống vì nó hỗ trợ mạnh mẽ về việc quản lý prompts – một yếu tố quan trọng để tạo ra các hướng dẫn cho các mô hình ngôn ngữ lớn (LLM), cho phép tùy chỉnh và tối ưu chức năng Adaptive Learning Path Management. Ngoài ra, tính năng memory của LangChain giúp lưu giữ và quản lý ngữ cảnh của các cuộc hội thoại trước đó, giúp các tương tác trở nên liền mạch và cá nhân hóa hơn.



2.4.2 Hệ quản trị cơ sở dữ liệu đồ thị

Hệ quản trị cơ sở dữ liệu đồ thị (Graph Database Management Systems - GDBMS) là một loại cơ sở dữ liệu được thiết kế để lưu trữ và truy vấn dữ liệu theo dạng đồ thị, với các nút (nodes), cạnh (edges), và thuộc tính (properties) mô phỏng mối quan hệ giữa các thực thể. Hệ thống này đặc biệt hữu ích trong các trường hợp yêu cầu xử lý các mối quan hệ phức tạp và tìm kiếm các kết nối giữa dữ liệu. Một trong những công nghệ nổi bật trong lĩnh vực này là Neo4j, là một hệ quản trị cơ sở dữ liệu đồ thị mạnh mẽ và phổ biến nhất hiện nay. Báo cáo này sẽ phân tích và so sánh Neo4j với một số lựa chọn thay thế khác để xác định công cụ phù hợp nhất cho hệ thống học tập thông minh.

2.4.2.1 Các công nghệ khác

- **ArangoDB:** ArangoDB là một cơ sở dữ liệu đa mô hình hỗ trợ đồ thị, tài liệu và key-value. Công cụ này mang lại sự linh hoạt khi kết hợp các mô hình dữ liệu khác nhau trong một cơ sở dữ liệu duy nhất.
- **OrientDB:** OrientDB là một cơ sở dữ liệu đồ thị đa mô hình, hỗ trợ đồ thị, tài liệu, đối tượng và mô hình key-value. Nó cung cấp các tính năng mạnh mẽ cho các hệ thống có yêu cầu phức tạp về quan hệ dữ liệu.

2.4.2.2 So sánh giữa các công nghệ

- **Neo4j:** Là công nghệ chuyên dụng cho cơ sở dữ liệu đồ thị, Neo4j sử dụng ngôn ngữ truy vấn Cypher, tối ưu cho việc xử lý và truy vấn các mối quan hệ phức tạp trong dữ liệu. Đây là công cụ lý tưởng cho các ứng dụng cần truy vấn đồ thị phức tạp và thực hiện các phân tích mối quan hệ giữa các thực thể.
- **ArangoDB:** ArangoDB cung cấp sự linh hoạt khi hỗ trợ các mô hình dữ liệu đa dạng. Tuy nhiên, việc kết hợp các mô hình dữ liệu có thể khiến việc làm việc với đồ thị trở nên phức tạp hơn so với Neo4j.
- **OrientDB:** Cũng hỗ trợ đa mô hình giống như ArangoDB, nhưng OrientDB không tối ưu bằng Neo4j trong việc truy vấn các mối quan hệ đồ thị phức tạp. Tuy nhiên, OrientDB có thể phù hợp với các yêu cầu phức tạp khác ngoài đồ thị.

2.4.2.3 Lựa chọn

Với các yếu tố trên, nhóm quyết định lựa chọn Neo4j là công nghệ tối ưu cho hệ thống học tập thông minh nhờ vào khả năng tối ưu hóa dữ liệu đồ thị và hiệu suất cao trong việc truy vấn các mối quan hệ phức tạp.



2.4.3 Framework xây dựng giao diện người dùng

Framework xây dựng giao diện người dùng (UI Framework) giúp phát triển các ứng dụng web với giao diện người dùng linh hoạt và dễ dàng tương tác. Trong số các framework phổ biến hiện nay, Vue.js là một lựa chọn nổi bật nhờ vào sự đơn giản, khả năng mở rộng tốt và dễ dàng tích hợp với các dự án hiện có. Mục tiêu của hệ thống học tập thông minh yêu cầu một framework có thể xây dựng giao diện người dùng mượt mà, dễ dàng mở rộng và duy trì. Báo cáo này sẽ phân tích Vue.js và so sánh với hai lựa chọn thay thế phổ biến: React và Angular, để xác định công cụ phù hợp nhất.

2.4.3.1 Các công nghệ khác

- **React:** React là một thư viện JavaScript mạnh mẽ, phát triển bởi Facebook, chuyên xây dựng giao diện người dùng qua các thành phần (components). React phù hợp với các ứng dụng có tính tương tác cao và khả năng tái sử dụng các thành phần.
- **Angular:** Angular là một framework JavaScript toàn diện, phát triển bởi Google, cung cấp một bộ công cụ đầy đủ để xây dựng các ứng dụng web phức tạp, bao gồm routing, state management và nhiều tính năng hỗ trợ khác.

2.4.3.2 So sánh giữa các công nghệ

- **Vue.js:** Vue.js có cú pháp dễ sử dụng, dễ tiếp cận và khả năng mở rộng tốt. Nó đặc biệt phù hợp với các ứng dụng vừa và nhỏ, dễ dàng tích hợp vào các dự án hiện có mà không làm tăng độ phức tạp.
- **React:** React là thư viện phổ biến, mạnh mẽ, đặc biệt cho các ứng dụng có tính tương tác cao. Tuy nhiên, React yêu cầu sử dụng thêm các thư viện như Redux cho việc quản lý state, điều này làm tăng độ phức tạp của dự án.
- **Angular:** Angular cung cấp nhiều công cụ mạnh mẽ nhưng có cú pháp khá phức tạp và thường yêu cầu người phát triển phải học hỏi nhiều hơn trước khi có thể tận dụng hết các tính năng. Nó phù hợp hơn cho các ứng dụng lớn, nhưng có thể gây khó khăn cho người mới bắt đầu.

2.4.3.3 Lựa chọn

Với cú pháp dễ sử dụng, khả năng mở rộng tốt và sự dễ dàng trong việc tích hợp vào dự án hiện tại, nhóm quyết định chọn Vue.js làm framework xây dựng giao diện người dùng cho hệ thống học tập thông minh.



2.4.4 Framework phát triển API

Framework phát triển API là công cụ quan trọng trong việc xây dựng các hệ thống web hiện đại. FastAPI là một framework Python nhanh chóng, hiệu quả và dễ sử dụng, đặc biệt phù hợp với việc xây dựng các API với hiệu suất cao. Với tính năng tự động tạo tài liệu API và kiểm tra kiểu dữ liệu tự động nhờ vào Pydantic, FastAPI giúp tối ưu hóa quá trình phát triển ứng dụng. Báo cáo này sẽ phân tích FastAPI và so sánh với hai lựa chọn thay thế khác: Flask và Django, nhằm xác định công cụ phù hợp nhất cho hệ thống học tập thông minh.

2.4.4.1 Các công nghệ khác

- **Flask:** Flask là một micro-framework Python đơn giản và dễ sử dụng, phù hợp cho các ứng dụng web nhỏ đến trung bình. Flask linh hoạt nhưng không cung cấp các tính năng tích hợp sẵn như FastAPI, yêu cầu người phát triển phải cấu hình thêm.
- **Django:** Django là một framework Python đầy đủ tính năng, hỗ trợ phát triển các ứng dụng web lớn với các công cụ tích hợp như ORM, hệ thống bảo mật và nhiều tính năng khác.

2.4.4.2 So sánh giữa các công nghệ

- **FastAPI:** FastAPI vượt trội về hiệu suất, hỗ trợ tự động tạo tài liệu API và kiểm tra kiểu dữ liệu nhờ vào Pydantic. Nó là sự lựa chọn lý tưởng cho các ứng dụng yêu cầu tốc độ cao và khả năng phát triển API dễ dàng.
- **Flask:** Flask linh hoạt và dễ sử dụng nhưng thiếu các tính năng tích hợp sẵn như FastAPI. Việc cấu hình và phát triển ứng dụng có thể cần thêm nhiều bước so với FastAPI.
- **Django:** Django cung cấp rất nhiều tính năng mạnh mẽ cho ứng dụng web lớn, nhưng đối với các API nhỏ hoặc cần sự kiểm soát chi tiết, Django có thể trở nên phức tạp và không cần thiết.

2.4.4.3 Lựa chọn

Với hiệu suất cao, tính năng tự động tạo tài liệu API và khả năng dễ dàng tích hợp, nhóm quyết định chọn FastAPI làm framework phát triển API cho hệ thống học tập thông minh.

2.5 Tổng kết

Chương 2 đã cung cấp một cái nhìn tổng quan về các khái niệm và lý thuyết nền tảng trong lĩnh vực giáo dục và giáo dục thông minh. Chúng ta đã tìm hiểu về khái niệm giáo dục, vai trò quan trọng của nó trong việc phát triển toàn diện con người và sự thích nghi với môi trường xã hội.



Đồng thời, khái niệm giáo dục thông minh đã được làm rõ, nhấn mạnh vào sự kết hợp giữa công nghệ và các phương pháp học tập cá nhân hóa, nhằm nâng cao hiệu quả giảng dạy và học tập.

Các khái niệm về mô hình ngôn ngữ, từ mô hình ngôn ngữ xác suất đến các mô hình ngôn ngữ lớn (LLMs), đã được trình bày chi tiết, làm nền tảng để hiểu rõ hơn về công nghệ ứng dụng trong giáo dục thông minh. Đặc biệt, những thuật ngữ quan trọng như đồ thị tri thức (Knowledge Graph), Knowledge Tracing, và các công nghệ hỗ trợ trong giáo dục như LangChain, LlamaIndex, Neo4J, và VueJS đã được giới thiệu, nhằm làm rõ các công cụ và nền tảng công nghệ cần thiết để phát triển hệ thống học tập thông minh hiện đại.

Kết thúc chương này, ta có thể thấy rằng việc áp dụng công nghệ vào giáo dục không chỉ là một xu hướng mà còn là một yêu cầu thiết yếu để nâng cao chất lượng và hiệu quả học tập. Trong chương tiếp theo, chúng ta sẽ đi sâu vào việc thiết kế và triển khai hệ thống giáo dục thông minh, đồng thời đánh giá các yếu tố ảnh hưởng đến sự thành công của hệ thống này.

Chapter 3

Công trình liên quan

3.1 Prompt engineering

Prompt engineering là một hướng đi khá mới do những bước phát triển lớn gần đây của mô hình ngôn ngữ nói chung và mô hình ngôn ngữ lớn nói riêng. Về khái niệm, *Prompt engineering* bao gồm việc thiết kế một cách có chiến lược các hướng dẫn cụ thể cho nhiệm vụ, được gọi là *prompts*, để hướng dẫn đầu ra của mô hình mà không thay đổi các tham số[5]. Một số kĩ thuật tiêu biểu trong prompt engineering có thể kể đến như sau[5]:

- Zero-shot prompting: Mô hình được nhận mô tả của công việc mà người dùng muốn nó thực hiện trong prompt nhưng không có dữ liệu được gán nhãn (labeled) để huấn luyện. Sau đó, mô hình tận dụng kiến thức sẵn có của nó để tạo ra các dự đoán dựa trên prompt đã cung cấp để thực hiện tác vụ.
- Few-shot prompting: Khác với Zero-shot prompting, để thực hiện tác vụ, ngoài đặc tả về công việc sẽ được thực hiện, mô hình còn nhận được một số ví dụ cụ thể về công việc mà người dùng muốn mô hình sẽ làm.
- Chain-of-thought prompting[6]: Kĩ thuật này được sử dụng nhằm cải thiện khả năng suy luận của mô hình. Thay vì chỉ đưa ra một câu trả lời trực tiếp, người dùng đưa cho mô hình cung cấp một chuỗi các bước suy nghĩ logic và có hệ thống trước khi đưa ra đáp án cuối cùng.

3.2 Retrieval-Augmentation Generation

Một trong những vấn đề của LLM đó là chúng có khả năng trả về những thông tin sai lệch, và Retrieval-Augmentation Generation (hay RAG) là kĩ thuật để hạn chế vấn đề đó, khi LLM sẽ



dựa trên một nguồn dữ liệu gọi là *knowledge base* để đưa ra câu trả lời. Có ba giai đoạn chính trong một kĩ thuật RAG[7] thông thường:

- *Indexing*: Raw data từ các nguồn khác nhau (PDF, HTML, Word, Markdown) được làm sạch và trích xuất, sau đó sẽ được chuyển đổi thành định dạng văn bản thuần nhất. Văn bản sẽ được chia ra thành các *chunks* và các chunks này sẽ được biến đổi sang dạng vector và được lưu trong một vector database.
- *Retrieval*: Khi nhận được một câu truy vấn, hệ thống sẽ biến đổi truy vấn thành vector và so sánh độ tương đồng giữa vector biểu diễn cho câu truy vấn và các đoạn văn bản đã được lập chỉ mục. Những đoạn văn bản có độ tương đồng cao nhất sẽ được chọn làm ngữ cảnh mở rộng cho mô hình.
- *Generation*: Truy vấn và các đoạn văn bản được tổng hợp thành một prompt, sau đó mô hình ngôn ngữ lớn sẽ phản hồi cho câu prompt đó. Mô hình có thể dựa trên kiến thức sẵn có hoặc chỉ sử dụng thông tin từ các tài liệu được cung cấp.

Với sự phát triển mạnh mẽ của LLMs trong những năm gần đây, các công trình liên quan đến RAG xuất hiện nhiều. Đầu tiên, đánh chỉ mục (indexing). PDF là một trong những định dạng tài liệu phổ biến nhất trên Internet, và cũng là một trong những nguồn dữ liệu chủ yếu được sử dụng cho việc đánh chỉ mục. Một nghiên cứu được thực hiện ở bài [8] cho thấy, công đoạn trích xuất nội dung từ một tập tin PDF (PDF parsing) có ảnh hưởng rất đáng kể đến chất lượng của việc truy vấn trong RAG. Về nguồn dữ liệu, thay vì chỉ tập trung vào văn bản là chủ yếu, có một số nghiên cứu, kĩ thuật liên quan được phát triển cho các nguồn dữ liệu khác như: Đồ thị tri thức là một dạng dữ liệu có cấu trúc, các dạng dữ liệu giả cấu trúc như PDF[7].

3.3 Knowledge tracing/Student Profiling

Với sự phát triển của máy tính và trí tuệ nhân tạo, nhiều công trình liên quan đến knowledge tracing/student profiling cũng đã được đề xuất và đưa ra, với mục tiêu cuối cùng là mô hình được năng lực của người học, từ đó phục vụ cho các bài toán khác trong giáo dục thông minh, đặc biệt là giáo dục thông minh. Một trong những mô hình tiêu biểu và cổ điển nhất trong lĩnh vực này đó là *Bayesian knowledge tracing*, với ý tưởng đó là mô hình hóa kiến thức của học sinh bằng cách sử dụng các biến ngẫu nhiên I_j , trong đó:

$$I_j = \begin{cases} 1, & \text{Nếu học sinh thành thục kĩ năng thứ } j \\ 0, & \text{ngược lại} \end{cases} \quad (3.1)$$

Ngoài ra, ở trong [9], nhóm tác giả đã xây dựng một mô hình knowledge tracing dựa trên một giả định đó là các khái niệm, mảng kiến thức có liên quan đến nhau về mặt ngữ nghĩa, và nếu



như mức độ thành thạo của người học ở một mảng kiến thức nhất định có thể đo được, và người học có một bước tiến ở một kĩ năng, hay một khái niệm nào đó, thì ở các mảng kiến thức liên quan, người học cũng có một bước tiến nhất định tương đối.

Với sự phát triển của Học sâu trong những năm trở lại đây, một nghiên cứu sử dụng một mô hình với tên gọi là *Deep knowledge tracing* [3] để giải quyết bài toán đã nêu, trong đó, ý tưởng của giải pháp đó là sử dụng mô hình *Recurrent Neural Network (RNN)*. Việc sử dụng RNN cho phép ta có thể nhận diện ra được những hình mẫu phức tạp hơn trong việc mô hình hóa kiến thức của học sinh.

Trong những năm trở lại đây, với sự xuất hiện của Mô hình ngôn ngữ lớn, có nhiều nghiên cứu đánh giá và tìm cách ứng dụng Mô hình ngôn ngữ lớn vào trong bài toán Knowledge tracing. Trong [10], (nhóm) tác giả đã đặt ra vấn đề rằng, do đặc tính *black-box* của các mô hình học sâu, nên kết quả của chúng *không thể giải thích được*, và (nhóm) tác giả đã tận dụng hai đặc tính *suy luận* và *tạo sinh* của mô hình ngôn ngữ lớn để đưa ra một lời giải thích cho hoạt động đánh giá năng lực của học viên của mô hình.

3.4 Hệ thống đề xuất lộ trình học

Rất nhiều nghiên cứu và công trình liên quan đến hệ thống đã được đề xuất và hiện thực. Hệ thống ở bài [11] sử dụng *Đồ thị tri thức* để biểu diễn mối liên hệ giữa các khái niệm (concepts) và các *kết quả học tập* (*learning outcome*). Dựa trên đồ thị tri thức, hệ thống sẽ dùng một giải thuật sinh đường (path generation algorithm) để tạo sinh một lộ trình học phù hợp dựa trên đầu vào là *chủ đề bắt đầu* và *chủ đề mục tiêu* của học sinh. Lộ trình học cũng có thể được thay đổi dựa trên năng lực, điểm số,... (Đây là các tham số động - dynamic parameters) của học sinh. Hệ thống *FOKE*[12] hiện thực một tính năng đề xuất cho người dùng lộ trình học phù hợp dựa trên các yếu tố như năng lực, sở thích, định hướng nghề nghiệp của người học. Trong hệ thống này, một mô hình dữ liệu được cải tiến từ *Đồ thị tri thức* (*knowledge graph*) với tên gọi là *Rừng tri thức* (*knowledge forest*), được sử dụng.

3.5 Kết luận

Chương 3 đã đi qua một số công trình nghiên cứu quan trọng liên quan đến các công nghệ và kỹ thuật ứng dụng trong giáo dục thông minh, đặc biệt là các phương pháp sử dụng mô hình ngôn ngữ lớn (LLMs), kiến thức truy hồi (Retrieval-Augmentation Generation), theo dõi kiến thức (Knowledge Tracing), và các hệ thống đề xuất lộ trình học tập. Các phương pháp này đã đóng góp đáng kể trong việc phát triển các giải pháp giáo dục thông minh, nâng cao khả năng cá nhân hóa trong học tập và hỗ trợ giáo viên trong việc theo dõi tiến trình học tập của học sinh.

Tuy nhiên, mặc dù các công trình này đã đạt được những kết quả đáng kể, vẫn còn một số khoảng trống có thể khai thác trong nghiên cứu và ứng dụng thực tế:



- **Tích hợp LLM với hệ thống giáo dục thông minh:** Mặc dù có sự phát triển mạnh mẽ trong việc sử dụng LLM cho các tác vụ cụ thể như sinh câu trả lời hay đánh giá năng lực, nhưng việc tích hợp chúng vào các hệ thống giáo dục toàn diện và cá nhân hóa chưa được nghiên cứu sâu. Việc kết hợp kiến thức từ LLM với các dữ liệu cụ thể của học sinh, chẳng hạn như quá trình học tập hoặc đặc điểm cá nhân, để tạo ra các lộ trình học tập tối ưu vẫn còn là một thách thức lớn.
- **Đánh giá và cải tiến khả năng suy luận của mô hình ngôn ngữ:** Các kỹ thuật như Chain-of-thought prompting đã được áp dụng để cải thiện khả năng suy luận của mô hình ngôn ngữ, nhưng chưa có nhiều nghiên cứu tập trung vào việc đánh giá và cải tiến sự chính xác trong quá trình suy luận khi áp dụng vào các bài toán giáo dục thực tiễn.
- **Cá nhân hóa dựa trên dữ liệu học tập thực tiễn:** Các hệ thống đề xuất lộ trình học hiện tại phần lớn dựa vào các mô hình tĩnh, như đồ thị tri thức hay các thuật toán sinh đường. Tuy nhiên, chưa có nhiều nghiên cứu tích hợp các yếu tố động như sự thay đổi trong năng lực học sinh theo thời gian, sở thích và các yếu tố ngữ cảnh khác, để xây dựng một hệ thống lộ trình học tập thực sự linh hoạt và cá nhân hóa.

Với những khoảng trống trên, công trình nghiên cứu của chúng tôi sẽ hướng đến việc kết hợp các phương pháp trên, đồng thời cải tiến tính linh hoạt và cá nhân hóa trong giáo dục thông minh. Các nghiên cứu tiếp theo sẽ tập trung vào việc phát triển các mô hình tổng hợp, kết hợp trí tuệ nhân tạo với dữ liệu học tập thực tế, để nâng cao hiệu quả học tập và khả năng dự báo tiềm bô của học sinh.

Chapter 4

Hệ thống đề xuất

4.1 Mô tả hệ thống

Hệ thống học tập trực tuyến thông minh được thiết kế để phục vụ cho sinh viên ngành công nghệ thông tin và những người có nhu cầu học lập trình. Mục tiêu chính của hệ thống là cung cấp trải nghiệm học tập cá nhân hóa thông qua việc ứng dụng các mô hình ngôn ngữ lớn (LLM – Large Language Model), nhằm tối ưu hóa quá trình học tập một cách linh hoạt, hiệu quả và phù hợp với nhu cầu riêng biệt của từng người học.

Hệ thống tập trung vào sự tương tác giữa người học và trí tuệ nhân tạo (AI), tạo ra một môi trường học tập thông minh. Cụ thể, hệ thống hỗ trợ giảng viên dễ dàng tạo, chỉnh sửa và quản lý nội dung học tập. Các chức năng chính của hệ thống bao gồm:

- **Quản lý khóa học:** Giảng viên có thể cung cấp thông tin chi tiết về khóa học, bao gồm tên khóa học, mô tả, mục tiêu học tập, và đối tượng học viên. Các tài liệu học tập có thể được tải lên hoặc liên kết đến các tài nguyên bổ sung.
- **Quản lý nội dung học:** Khóa học được chia thành các module hoặc bài học nhỏ với mục tiêu học tập cụ thể và thời gian hoàn thành dự kiến. Mỗi module hoặc bài học sẽ có mục tiêu học tập rõ ràng để sinh viên biết họ cần đạt được những kỹ năng hoặc kiến thức nào sau khi hoàn thành.
- **Lộ trình học tập cá nhân hóa:** Lộ trình học tập của mỗi sinh viên sẽ được cá nhân hóa thông qua các đề xuất learning item (bao gồm khóa học, module, bài học) dựa trên mục tiêu học tập, nền tảng kiến thức, và hành vi học tập của người dùng (như điểm số, thời gian hoàn thành bài học). Hệ thống sẽ linh hoạt đề xuất các item phù hợp với trình độ hiện tại của người học. Các bài học và module sẽ được điều chỉnh dần theo độ khó tăng lên, giúp người học tiếp cận kiến thức mới một cách hợp lý và tiến bộ liên tục.



- **Gia sư AI hỗ trợ lập trình:** Một trong những tính năng nổi bật của hệ thống là gia sư AI, đóng vai trò như một trợ lý thông minh trong việc giám sát quá trình học của người học. Gia sư AI có khả năng cung cấp các giải thích chi tiết về mã nguồn, gợi ý cách viết mã tối ưu, hỗ trợ sửa lỗi (debug), và đưa ra phản hồi về chất lượng mã lệnh. Tính năng này không chỉ giúp người học nâng cao kỹ năng lập trình mà còn giúp cải thiện hiệu quả học tập thông qua việc tiếp thu kiến thức một cách nhanh chóng và dễ hiểu.
- **Hệ thống tạo bài tập tự động:** Hệ thống có khả năng tự động tạo ra các bài tập lập trình theo chủ đề mà người học mong muốn. Bộ test case cho bài tập sẽ được sinh ra tự động tùy thuộc vào mức độ khó và yêu cầu của bài tập. Người học có thể yêu cầu hệ thống tạo bài tập ở các cấp độ khác nhau (từ cơ bản đến nâng cao) hoặc nhập vào một câu hỏi lập trình cụ thể để hệ thống tạo bộ test case phù hợp.
- **Bài tập sẽ được điều chỉnh linh hoạt:** Bài tập sẽ được điều chỉnh linh hoạt để phù hợp với năng lực và tiến độ học tập của người học. Hệ thống sẽ theo dõi và đánh giá kết quả bài tập dựa trên thời gian giải quyết, độ chính xác, và điểm số, từ đó điều chỉnh lộ trình học tập của người học.
- **Theo dõi tiến độ và báo cáo học tập:** Hệ thống cung cấp công cụ theo dõi tiến độ học tập chi tiết. Sinh viên có thể xem lại thời gian học, số lượng bài tập đã hoàn thành và kết quả của các bài kiểm tra. Hệ thống sẽ cung cấp báo cáo chi tiết về quá trình học tập của sinh viên, bao gồm các bài học đã hoàn thành, kết quả đạt được và các gợi ý từ hệ thống cho bước học tiếp theo. Dựa trên các chỉ số này, hệ thống sẽ đánh giá mức độ tiến bộ của người học và điều chỉnh lộ trình học tập một cách linh hoạt.

4.2 Khảo sát yêu cầu

4.2.1 Người dùng

Các người dùng liên quan trong hệ thống:

- Học viên.
- Quản trị viên.
- Giảng viên

4.2.2 Yêu cầu chức năng

4.2.2.1 User Management

1. **Đăng ký và đăng nhập:** Người dùng có thể tạo tài khoản và đăng nhập bằng thông tin cá nhân hoặc tài khoản mạng xã hội.



2. **Hồ sơ học tập:** Mỗi người dùng có một hồ sơ học tập, lưu trữ thông tin cá nhân, tiến độ học tập, và kết quả các bài kiểm tra/bài tập.
3. **Quản lý lịch sử học tập:** Hệ thống lưu trữ và cho phép người dùng xem lại các bài học, bài tập đã hoàn thành và kết quả đạt được.
4. **Cấu hình cá nhân:** Người dùng có thể cập nhật thông tin cá nhân, độ tuổi.

4.2.2.2 Learning Resource Recommendation System

1. **Cá nhân hóa đề xuất học tập:** Hệ thống tự động đề xuất các tài nguyên học tập (khóa học, module, bài học, bài tập, project) cho mỗi người dùng dựa trên mục tiêu học tập, độ tuổi, trình độ học vấn, phong cách học tập, và hành vi học tập, lịch sử học tập, mức độ thông thạo của người học trên một khái niệm hoặc một đầu ra học tập cụ thể.
2. **Lựa chọn tài nguyên học tập:** Hệ thống cho phép người dùng tự chọn các learning item dựa trên sở thích hoặc mục tiêu cá nhân, cung cấp sự linh hoạt trong việc tiếp cận nội dung học tập.
3. **Theo dõi phản hồi người dùng:** Hệ thống ghi nhận phản hồi của người dùng về các tài nguyên học tập để cải thiện đề xuất và tối ưu hóa trải nghiệm người học.

4.2.2.3 Exercise Generator

1. **Tạo bài tập theo mức độ:** Hệ thống tự động sinh ra các bài tập từ cơ bản đến nâng cao phù hợp với khả năng của người dùng.
2. **Tạo bài tập theo yêu cầu:** Người dùng có thể yêu cầu hệ thống tạo bài tập ở các cấp độ khác nhau hoặc nhập vào một câu hỏi lập trình cụ thể.
3. **Tự động điều chỉnh bài tập:** Dựa trên tiến độ học tập, các bài tập sẽ được nâng cấp để phù hợp với mức độ phát triển của người dùng.
4. **Phân tích kết quả bài tập:** Hệ thống đánh giá bài tập dựa trên thời gian giải, độ chính xác, và kết quả điểm số để điều chỉnh lộ trình học.
5. **Tạo bộ test case tự động:** Hệ thống tự động tạo bộ test case phù hợp với yêu cầu của bài tập mà người dùng upload.

4.2.2.4 AI Tutor

1. Hệ thống có tích hợp một tính năng là Smart Assistant. Khi người dùng tương tác với tài nguyên học tập, sẽ có một hệ thống chatbot hỗ trợ trả lời nội dung liên quan đến tài nguyên học tập mà người dùng đang xem.



2. Ứng với các tài nguyên học tập đòi hỏi người dùng phải thực hành như là các bài tập lập trình, thì hệ thống cũng có tích hợp một số tính năng đi kèm như sau:

- **Hỗ trợ giải thích code:** Người dùng có thể yêu cầu gia sư AI giải thích code hoặc hướng dẫn cách giải quyết các bài tập lập trình.
- **Gợi ý hướng giải:** Khi gặp khó khăn, hệ thống sẽ đưa ra các gợi ý từ gia sư AI nhằm hỗ trợ người dùng hoàn thành bài tập.
- **Đưa ra gợi ý:** Hệ thống có thể hỏi người dùng liệu họ có muốn thay đổi bài tập sang một bài dễ hơn với nội dung tương tự khi gặp vấn đề với bài hiện tại.
- **Tối ưu hóa code:** Gia sư AI cung cấp gợi ý cách viết mã tối ưu hơn.
- **Hỗ trợ sửa lỗi:** Gia sư AI cung cấp hỗ trợ debug khi người dùng gặp lỗi trong code.
- **Đánh giá chất lượng code:** Gia sư AI cung cấp phản hồi kịp thời về chất lượng mã lệnh của người dùng.

4.2.2.5 Progress Tracking

1. **Theo dõi tiến độ học tập:** Hệ thống ghi nhận thời gian học, số lượng bài tập hoàn thành và kết quả các bài kiểm tra.
2. **Đánh giá tiến độ:** Dựa trên các chỉ số như tốc độ giải quyết bài tập và điểm số, hệ thống đánh giá sự tiến bộ của người dùng và điều chỉnh lộ trình học tập.
3. **Báo cáo tiến độ:** Người dùng có thể xem báo cáo chi tiết về quá trình học tập, bao gồm các bài học đã hoàn thành, kết quả đạt được, và gợi ý từ hệ thống cho bước tiếp theo.
4. **Phân tích dữ liệu học tập:** Hệ thống phân tích hành vi học tập của người dùng, bao gồm điểm số, thời gian hoàn thành bài tập, và tần suất yêu cầu hỗ trợ để cải thiện trải nghiệm học tập.

4.2.2.6 Course Management

1. **Tạo khóa học/module:** Giảng viên có thể tạo và thiết kế các khóa học hoặc module học tập mới. Ứng với mỗi một khóa học, giảng viên sẽ phải cung cấp các thông tin về đầu ra học tập (learning outcome), về mô tả của khóa học. Ứng với mỗi khóa học sẽ bao gồm các modules/bài học/projects tương ứng.
2. **Quản lý nội dung:** Giảng viên có thể cập nhật, chỉnh sửa nội dung của khóa học cũng như tài nguyên học tập. Hệ thống cho phép tải lên các tài liệu học tập hoặc liên kết đến các tài nguyên bổ sung.



4.2.2.7 Admin Management

1. Quản lý người dùng:

- Quản trị viên có thể xem danh sách người dùng, tìm kiếm, lọc, và sắp xếp theo các tiêu chí như tên, email, trạng thái tài khoản, và tiến độ học tập.
- Quản trị viên có thể khóa/mở khóa tài khoản người dùng hoặc thay đổi quyền truy cập của người dùng (như chuyển từ người dùng bình thường sang giảng viên).

2. Quản lý phản hồi:

- Xem và xử lý các phản hồi, đánh giá của người dùng về khóa học, giảng viên, và hệ thống.
- Gửi phản hồi cho người dùng về các vấn đề đã được xử lý.

4.2.3 Yêu cầu phi chức năng

• Hiệu năng

- Hệ thống phải xử lý yêu cầu truy cập và tương tác của người dùng một cách nhanh chóng và hiệu quả, với thời gian phản hồi không vượt quá 2 giây cho các thao tác thông thường như đăng nhập, xem khóa học, và tải trang.
- Hệ thống phải có khả năng hỗ trợ ít nhất 1000 người dùng đồng thời mà không làm giảm hiệu năng.
- Thời gian phản hồi của các công cụ AI không được quá 7-8s.

• Khả năng mở rộng

- Hệ thống có khả năng mở rộng khi số lượng người dùng, khóa học, và bài tập tăng lên mà không ảnh hưởng đến chất lượng dịch vụ và thay đổi hệ thống quá nhiều.
- Có thể thêm các chức năng phụ kết hợp với chức năng sẵn có nếu có yêu cầu mở rộng.

• Bảo mật

- Dữ liệu cá nhân của người dùng và kết quả học tập phải được mã hóa trong quá trình truyền tải và lưu trữ.

• Tính khả dụng

- Người dùng có thể dễ dàng điều hướng giữa các khóa học, theo dõi tiến trình học tập và thực hiện các bài kiểm tra mà không gặp khó khăn.



- Giao diện người dùng cần trực quan, dễ sử dụng, màu sắc cơ bản.
- Tính tương thích
 - Hệ thống phải tương thích với nhiều trình duyệt phổ biến (Chrome, Firefox, Safari, Edge) và các hệ điều hành khác nhau (Windows, macOS, Linux).



4.2.4 Yêu cầu dữ liệu

4.2.4.1 Các thực thể và thuộc tính dữ liệu

1. User

- Thực thể: Người dùng
- Thuộc tính: ID, username, password, role (student, admin, teacher)
 - Student: Goal, Birthyear, Background.
 - Teacher
 - Admin

2. Course

- Thực thể: Khóa học
- Thuộc tính: ID, Name, Description, Type, Content.

3. LearningPath

- Thực thể: Lộ trình học tập
- Thuộc tính: ID, ExerciseCount, Level, Time.

4. Exercise

- Thực thể: bài tập
- Thuộc tính: ID bài tập, Document, Difficulty, Type (Quiz, Code).
 - Quiz: ID, Content, Questions, Answer.
 - Code: ID, Content, Language, Testcases.

5. Knowledge

- Thực thể: Dữ liệu kiến thức
- Thuộc tính: Questions and Answers, Document, ExerciseData.



4.2.4.2 Mối quan hệ dữ liệu

- Admin có thể Manage (quản lý) Course, Knowledge, và LearningPath.
- Teacher có thể Upload (tải lên) Course.
- Student Have (có) một hoặc nhiều LearningPath.
- LearningPath bao gồm nhiều Exercise.
- AI Use (sử dụng) dữ liệu từ Knowledge và Course để Generate (sinh ra) các LearningPath, Exercise.
- Exercise bao gồm 2 loại bài tập là Quiz và Code.

4.3 Kết luận

Chương 4 đã trình bày chi tiết về hệ thống đề xuất trong môi trường học tập trực tuyến thông minh. Hệ thống cung cấp các công cụ mạnh mẽ giúp người học có trải nghiệm học tập hiệu quả và tối ưu. Các tính năng như tạo bài tập tự động, theo dõi tiến độ học tập, và hỗ trợ từ gia sư AI giúp người học nâng cao khả năng lập trình và tiếp cận kiến thức một cách linh hoạt, hiệu quả.

Chapter 5

Phân tích và thiết kế

5.1 Phân tích

5.1.1 Business Process Modeling

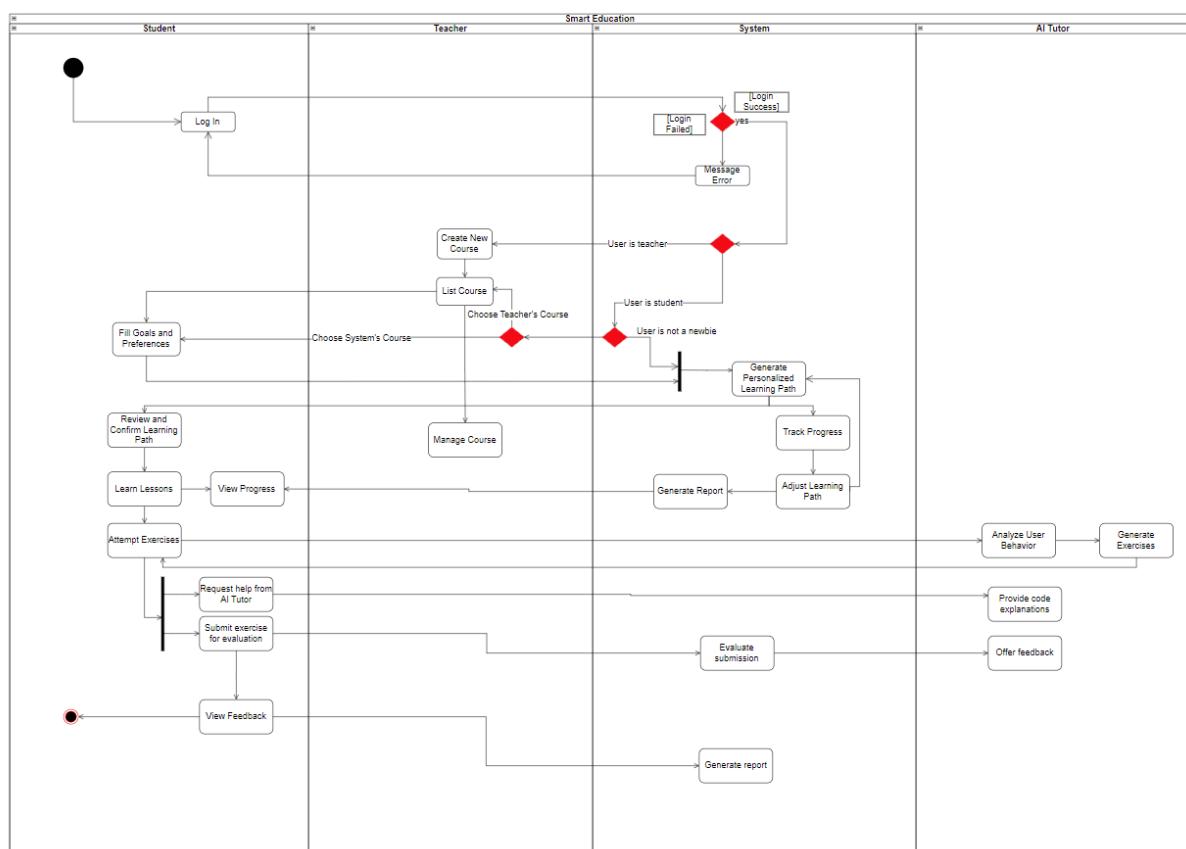


Figure 5.1: Activity diagram

Activity diagram mô tả quy trình tương tác giữa bốn Actors: **Student (Học sinh)**, **Teacher (Giáo viên)**, **System (Hệ thống)**, và **AI Tutor (Gia sư AI)** trong hệ thống giáo dục thông minh.

Học sinh bắt đầu bằng việc đăng nhập và tùy thuộc vào vai trò của họ (giáo viên hoặc học sinh), các hành động khác nhau sẽ diễn ra. Giáo viên có thể tạo khóa học mới hoặc chọn khóa học hiện có để quản lý, trong khi học sinh sẽ xác định mục tiêu học tập và xác nhận lộ trình học do hệ thống đề xuất. Sau đó, học sinh có thể học bài, thực hiện bài tập, và nếu cần có thể yêu cầu trợ giúp từ AI Tutor. Hệ thống cũng tự động theo dõi tiến độ học tập, điều chỉnh lộ trình và đánh giá bài nộp của học sinh. AI Tutor hỗ trợ bằng cách phân tích hành vi, giải thích mã code, và tạo bài tập phù hợp.

5.1.2 Use case diagrams

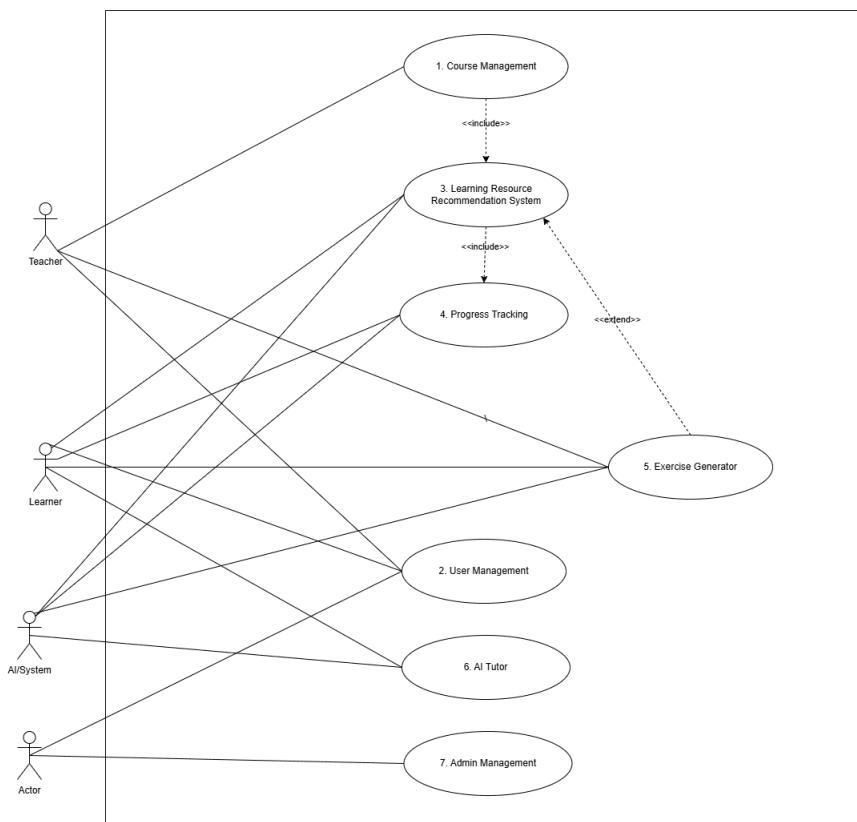


Figure 5.2: Use case diagram

Mô hình use case tổng thể cho hệ thống học tập thông minh bao gồm ba tác nhân chính: Giáo viên, Người học, Quản trị viên và Hệ thống/AI. Các use case bao gồm Quản lý khóa học, Quản lý người dùng, Quản lý lộ trình học tập thích ứng, Theo dõi tiến độ, Tạo bài tập, Quản lý hệ thống cho Quản trị viên và Gia sư AI.

5.1.2.1 Course Management

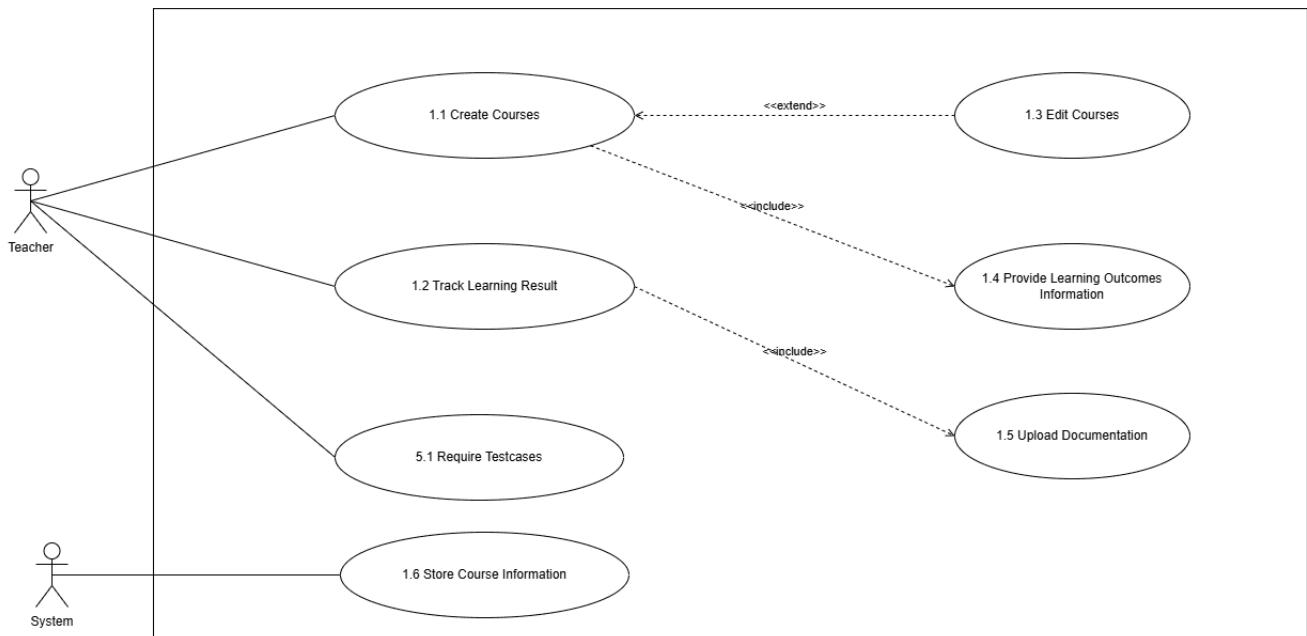


Figure 5.3: Detailed use case with id 1: Course Management

Use case **Quản lý khóa học** bao gồm ba chức năng chính: Tạo khóa học, Theo dõi kết quả học tập và Yêu cầu bài kiểm tra. Use case "**Chỉnh sửa khóa học**" được mở rộng từ "**Tạo khóa học**", cho thấy việc chỉnh sửa là một phần mở rộng tùy chọn của quá trình tạo khóa học. Use case diagram thể hiện các trách nhiệm chính của giáo viên trong việc quản lý khóa học trong hệ thống.

Ngoài ra, ở module này, Hệ thống sẽ có nhiệm vụ lưu trữ thông tin về khóa học mà giáo viên đã tạo ra.

5.1.2.2 User Management

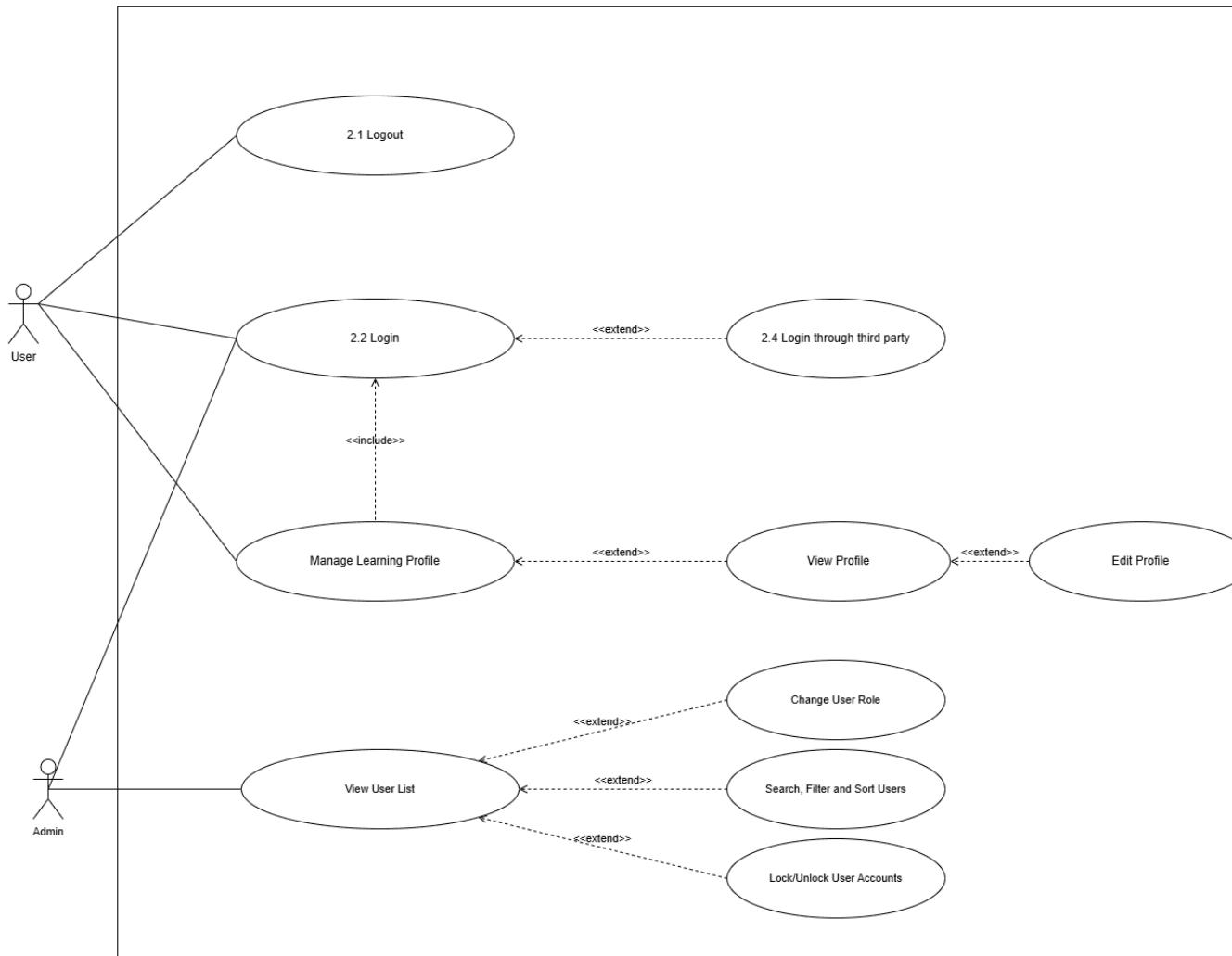


Figure 5.4: Detailed use case with id 2: User Management

Use case **Quản lý người dùng**, tập trung vào tác nhân Người dùng bao gồm các chức năng như Đăng nhập, Đăng xuất, Quản lý hồ sơ học tập và Xem hồ sơ. Use case diagram cho thấy "**Đăng nhập qua third-party**" mở rộng từ use case "**Đăng nhập**" cơ bản, cung cấp một phương thức đăng nhập thay thế. Ngoài ra, "**Xem hồ sơ**" được thể hiện như một phần mở rộng của "**Quản lý hồ sơ học tập**", cho thấy đây là một hành động liên quan nhưng riêng biệt.

Quản trị viên có quyền đăng nhập bằng tài khoản của mình, từ đó "**Xem danh sách người dùng**" sử dụng hệ thống. Ngoài ra, quản trị viên có thể "**Tìm kiếm, Lọc và Sắp xếp theo thông tin người dùng**", "**Vô hiệu hóa tài khoản/Kích hoạt tài khoản**" và "**Thay đổi quyền truy cập của người dùng**". Ví dụ: Thay đổi từ người học sang giảng viên.

5.1.2.3 Learning Resource Recommendation System

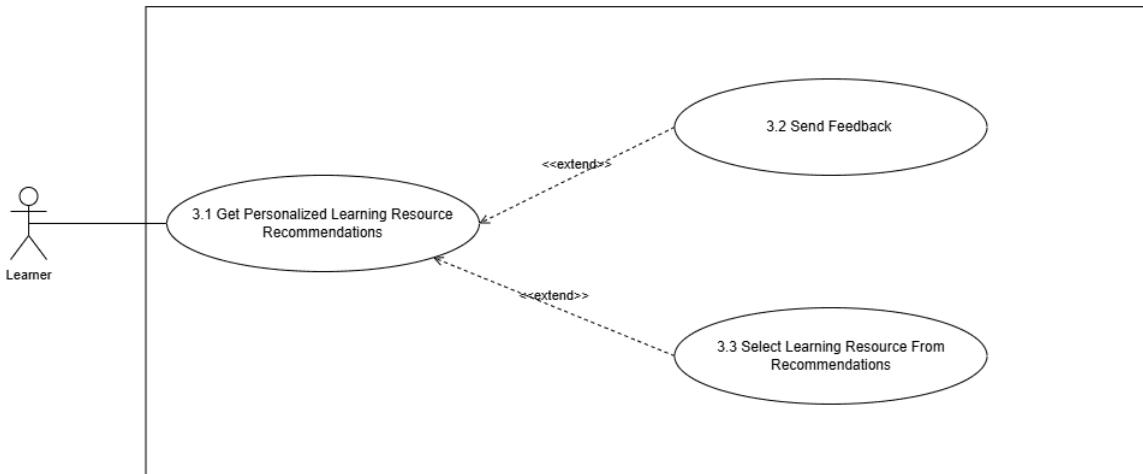


Figure 5.5: Detailed use case with id 3: Adaptive Learning Path Management of Learner Actor

Tác nhân chính trong biểu đồ trên là người học (Learner), có thể tương tác với hệ thống để nhận các đề xuất tài nguyên học tập. Use case "**Get Personalized Learning Resource Recommendations**" cung cấp cho người học các tài nguyên học tập phù hợp với nhu cầu và lịch sử học tập của họ. Use case "**Send Feedback**" có mối quan hệ "<extend>" với "**Get Personalized Learning Resource Recommendations**", vì người học có thể gửi phản hồi về các tài nguyên đã được đề xuất để cải thiện hệ thống. Use case "**Select Learning Resource From Recommendations**" cũng mở rộng từ "**Get Personalized Learning Resource Recommendations**", cho phép người học tự chọn các tài nguyên mà họ thấy phù hợp với mục tiêu cá nhân.

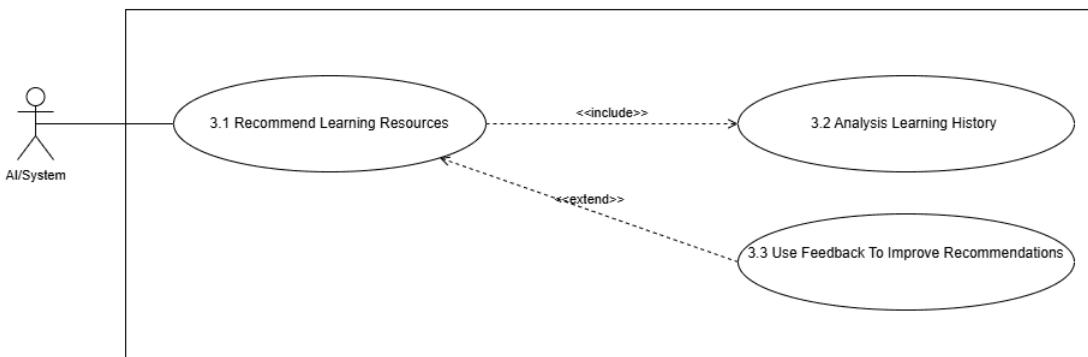


Figure 5.6: Detailed use case with id 3: Learning Resource Recommendation System of System Actor

Tác nhân chính là hệ thống AI, có nhiệm vụ chính là đề xuất tài nguyên học tập cá nhân hóa cho người dùng. Use case "**Recommend Learning Resources**" bao gồm hoạt động "**Analysis Learning History**" (phân tích lịch sử học tập), do quá trình phân tích lịch sử học tập là một phần không thể thiếu khi hệ thống tạo ra các đề xuất. Hệ thống có thể mở rộng thêm use case "**Use**

Feedback To Improve Recommendations" (sử dụng phản hồi để cải thiện đề xuất) dựa trên các phản hồi người dùng, giúp tinh chỉnh và nâng cao chất lượng đề xuất. Mỗi quan hệ "**<include>**" được sử dụng giữa "**Recommend Learning Resources**" và "**Analysis Learning History**" vì quá trình phân tích là cần thiết. Mỗi quan hệ "**<extend>**" giữa "**Recommend Learning Resources**" và "**Use Feedback To Improve Recommendations**" vì cải thiện đề xuất dựa trên phản hồi là một phần mở rộng có thể xảy ra tùy theo ngữ cảnh.

5.1.2.4 Progress Tracking

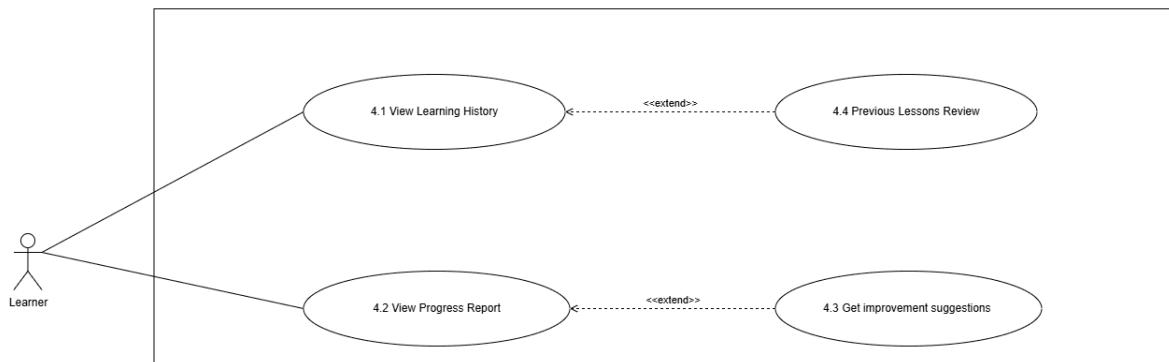


Figure 5.7: Detailed use case with id 4: Progress Tracking of Learner Actor

Use case diagram trên thể hiện các chức năng **Theo dõi tiến độ dành cho Người học**. Có hai use case chính là "**Xem lịch sử học tập**" và "**Xem báo cáo tiến độ**". "**Xem lịch sử học tập**" mở rộng đến "**Xem lại các bài học trước đó**", cho phép người học ôn tập. "**Xem báo cáo tiến độ**" mở rộng đến "**Nhận đề xuất cải thiện**", cung cấp hướng dẫn cá nhân hóa, nhấn mạnh việc theo dõi và phản hồi liên tục để hỗ trợ quá trình học tập của người dùng.

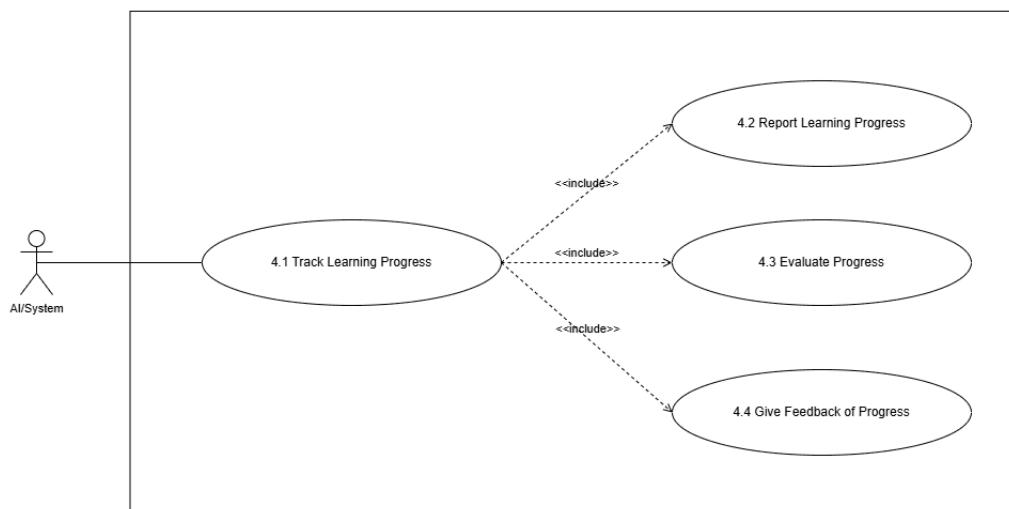


Figure 5.8: Detailed use case with id 4: Progress Tracking of System Actor

Sơ đồ use case của hệ thống **Exercise Generator** mô tả các tương tác giữa người dùng và



hệ thống. Người học có thể yêu cầu hệ thống tạo tự động các bài tập thông qua chức năng "**Request Auto Exercise Generation**", và sau đó thực hiện các bài tập đó để cải thiện kỹ năng của mình thông qua "**Do Exercise**". Bên cạnh đó, giảng viên có thể sử dụng chức năng "**Upload Exercise**" để tải lên các đề bài hoặc bài tập, từ đó hệ thống sẽ tự động xử lý và tạo ra các test case nhằm kiểm tra độ chính xác của bài giải thông qua "**Generate Test Cases**". Sau khi các test case được tạo, giảng viên có thể sử dụng chức năng "**Review Test Cases**" để xem xét và đánh giá các test case nhằm đảm bảo chúng phù hợp với nội dung bài tập và đáp ứng được yêu cầu giảng dạy. Các chức năng này giúp hệ thống tạo ra môi trường học tập linh hoạt, hỗ trợ hiệu quả cho người học và giảng viên.

5.1.2.5 Exercise Generator

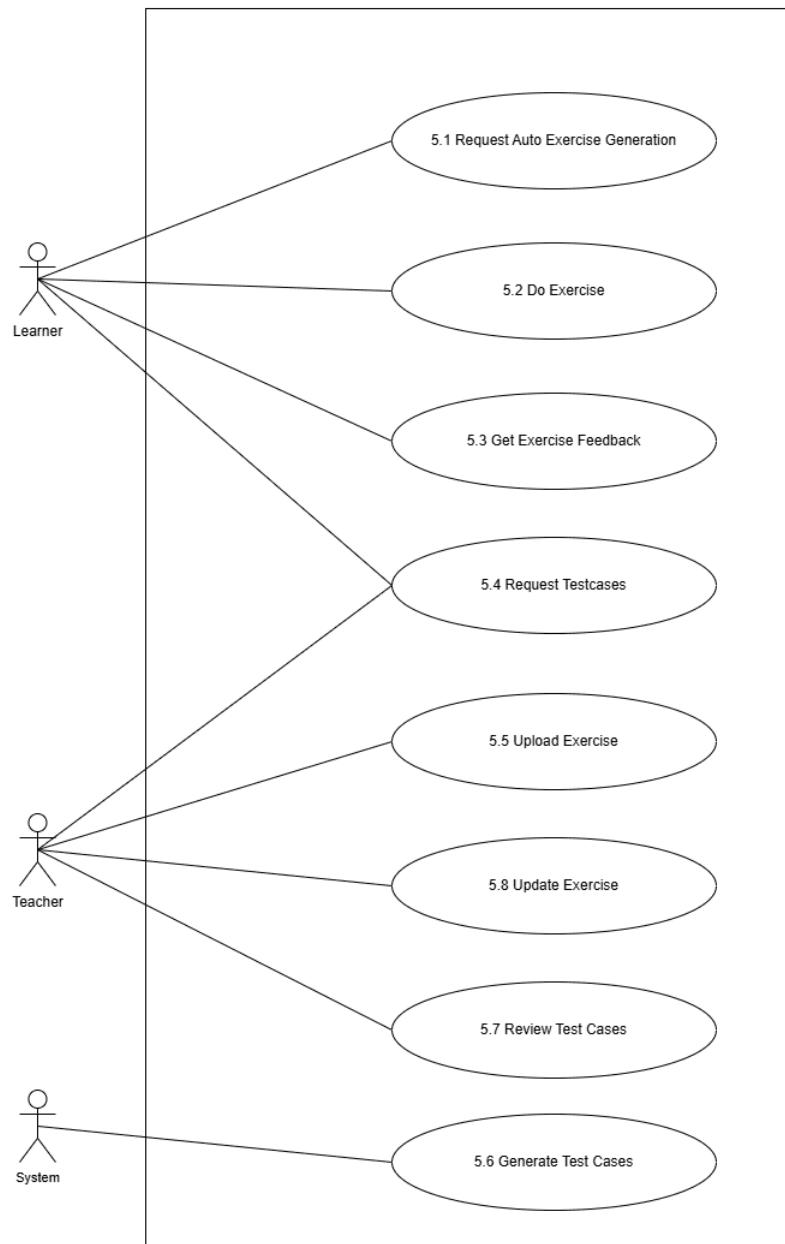


Figure 5.9: Detailed use case with id 5: Exercise Generator

Use case diagram thể hiện các chức năng liên quan đến bài tập dành cho Người học. Có bốn use case chính: "*Yêu cầu tạo bài tập tự động*", "*Làm bài tập*", "*Nhận phản hồi về bài tập*", và "*Yêu cầu bài kiểm tra*". Người học có thể yêu cầu hệ thống tạo bài tập tự động phù hợp với trình độ của họ. Sau khi làm bài tập, người học có thể nhận phản hồi để hiểu rõ hơn về kết quả của mình. Khả năng yêu cầu bài kiểm tra cho phép người học tự đánh giá kiến thức của mình khi cần.

5.1.2.6 AI Tutor

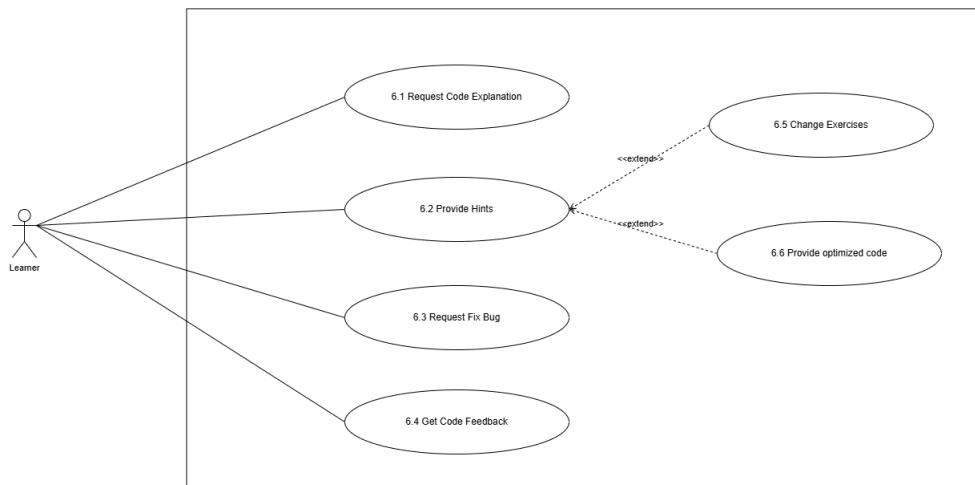


Figure 5.10: Detailed use case with id 6: AI Tutor of Learner Actor

Use case diagram thể hiện các chức năng của Gia sư AI dành cho Người học. Có bốn use case chính: "*Yêu cầu giải thích mã*", "*Yêu cầu gợi ý*", "*Yêu cầu sửa lỗi*", và "*Nhận phản hồi về mã*". Người học có thể yêu cầu giải thích về mã nguồn để hiểu rõ hơn về cấu trúc và logic của code. Chức năng yêu cầu gợi ý có hai chức năng quan hệ "*extend*" là "*Hỗ trợ chuyển đổi bài tập*" dựa trên nhu cầu, học lực của người dùng và "*Cung cấp mã tối ưu hóa*" giúp người học vượt qua các khó khăn trong quá trình học lập trình cũng như cải thiện tối đa kỹ thuật code của người dùng. Việc nhận phản hồi về mã giúp người học cải thiện kỹ năng lập trình của mình một cách liên tục.

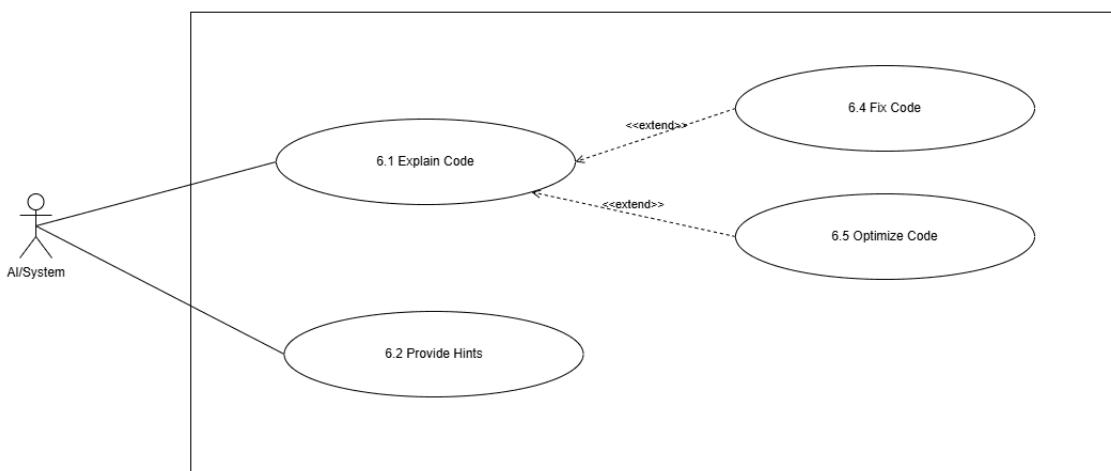


Figure 5.11: Detailed use case with id 6: AI Tutor of System Actor

Use case diagram trên mô tả **hệ thống AI Tutor** với vai trò hỗ trợ người dùng trong việc xử lý và hiểu mã lập trình. Hệ thống có khả năng *giải thích mã lệnh* (*Use case 6.1*) và *cung cấp gợi ý khi người dùng gặp khó khăn* (*Use case 6.2*). Từ chức năng giải thích mã, hệ thống có



thể mở rộng thêm các chức năng khác như *sửa lỗi* (*Use case 6.4*) và *tối ưu mã lệnh* (*Use case 6.5*). Hệ thống nhằm mục tiêu hỗ trợ người dùng học lập trình hiệu quả hơn thông qua các công cụ hỗ trợ trực tiếp từ AI.



5.2 Thiết kế

5.2.1 Kiến trúc hệ thống

5.2.1.1 Phân tích kiến trúc hệ thống phổ biến hiện nay

Hiện nay, có nhiều kiến trúc phần mềm khác nhau, mỗi loại có ưu và nhược điểm riêng, phù hợp với các loại dự án và yêu cầu cụ thể. Dưới đây là một số kiến trúc phần mềm phổ biến và các điểm chính của chúng:

1. Kiến trúc Monolithic

- **Định nghĩa:** Kiến trúc monolithic là một mô hình truyền thống trong đó toàn bộ ứng dụng được xây dựng như một đơn vị duy nhất, tích hợp chặt chẽ.
- **Điểm chính:**
 - Đơn giản để phát triển, triển khai và kiểm thử
 - Hiệu suất tốt do không có overhead của network calls
 - Khó mở rộng và bảo trì khi ứng dụng phức tạp
 - Khó áp dụng công nghệ mới cho từng phần của ứng dụng
- **Ví dụ:** Một ứng dụng web PHP truyền thống, nơi tất cả chức năng (UI, business logic, data access) đều nằm trong cùng một codebase.

2. Kiến trúc Microservices

- **Định nghĩa:** Kiến trúc microservices chia nhỏ ứng dụng thành các dịch vụ độc lập, mỗi dịch vụ chịu trách nhiệm cho một chức năng cụ thể và có thể được phát triển, triển khai độc lập.
- **Điểm chính:**
 - Dễ dàng mở rộng và bảo trì từng service riêng biệt
 - Cho phép sử dụng công nghệ phù hợp nhất cho mỗi service
 - Tăng cường khả năng chịu lỗi của hệ thống
 - Phức tạp hơn trong việc quản lý và điều phối các service
- **Ví dụ:** Netflix sử dụng kiến trúc microservices, với các service riêng biệt cho việc xử lý video, quản lý người dùng, đề xuất nội dung, v.v.

3. Kiến trúc Client-Server (Multitier)

- **Định nghĩa:** Kiến trúc client-server chia ứng dụng thành các tầng (tiers) riêng biệt, thường bao gồm tầng trình bày (presentation), tầng logic nghiệp vụ (business logic), và tầng dữ liệu (data).



• **Điểm chính:**

- Phân tách rõ ràng giữa các tầng chức năng
- Dễ dàng mở rộng từng tầng độc lập
- Cải thiện bảo mật bằng cách kiểm soát quyền truy cập giữa các tầng
- Có thể trở nên phức tạp khi số lượng tầng tăng lên
- **Ví dụ:** Một ứng dụng web ba tầng điển hình bao gồm frontend (React), backend API (Node.js), và cơ sở dữ liệu (PostgreSQL).

4. Kiến trúc Event-Driven

- **Định nghĩa:** Kiến trúc event-driven là một mô hình trong đó việc tạo ra, phát hiện, tiêu thụ và phản ứng với các sự kiện là cốt lõi của hệ thống.
- **Điểm chính:**
 - Tách rời các thành phần hệ thống, giảm sự phụ thuộc trực tiếp
 - Khả năng mở rộng và linh hoạt cao
 - Dễ dàng thêm chức năng mới mà không ảnh hưởng đến các thành phần hiện có
 - Có thể phức tạp trong việc theo dõi luồng xử lý và debug
- **Ví dụ:** Hệ thống thương mại điện tử, khi một đơn hàng được đặt, một sự kiện "OrderPlaced" được phát ra, kích hoạt các quy trình xử lý khác như cập nhật kho, gửi email xác nhận, v.v.

5. Kiến trúc Serverless

- **Định nghĩa:** Kiến trúc serverless cho phép phát triển và chạy ứng dụng mà không cần quản lý trực tiếp máy chủ, tập trung vào việc viết code cho các chức năng (functions) cụ thể.
- **Điểm chính:**
 - Giảm chi phí vận hành và bảo trì hạ tầng
 - Tự động mở rộng theo nhu cầu sử dụng
 - Chỉ trả tiền cho tài nguyên thực sự sử dụng
 - Có thể gặp vấn đề về cold start và vendor lock-in
- **Ví dụ:** Một ứng dụng xử lý ảnh sử dụng AWS Lambda để thực hiện các tác vụ như resize, filter, và lưu trữ ảnh khi người dùng tải lên.

5.2.1.2 Lựa chọn kiến trúc hệ thống

Dựa trên yêu cầu của hệ thống, nhóm sử dụng kiến trúc kết hợp giữa Microservices và Multitier Client-Server. Cụ thể như sau:



1. Presentation Tier (Client Tier):

- Frontend: Vue.js application
- Giao diện người dùng cho sinh viên và giảng viên
- Tương tác với backend thông qua RESTful API

2. Application Tier:

(a) API Gateway:

- Nginx hoặc API Gateway mã nguồn mở
- Xử lý authentication, authorization và định tuyến requests đến các microservices

(b) Microservices (tất cả được xây dựng bằng Fast API):

- User Service:
 - Quản lý thông tin người dùng
 - Xác thực và phân quyền
- Course Management Service:
 - CRUD operations cho khóa học, module, bài học
 - Quản lý nội dung học tập
- Recommendation Learning Service:
 - Đề xuất learning items dựa trên dữ liệu người dùng
- AI Tutor Service:
 - Tích hợp Langchain để xây dựng gia sư AI
 - Xử lý các yêu cầu liên quan đến hỗ trợ lập trình và giải thích mã nguồn
- Exercise Generation Service:
 - Sử dụng Langchain để tạo bài tập và test cases tự động
 - Đánh giá bài làm của người học
- Progress Tracking Service:
 - Theo dõi và ghi nhận tiến độ học tập
 - Tạo báo cáo học tập

3. Data Tier:

(a) Persistence Layer:

- PostgreSQL: Lưu trữ dữ liệu có cấu trúc (thông tin người dùng, khóa học, bài học)



- Neo4j: Lưu trữ và quản lý mối quan hệ giữa các learning items, hỗ trợ tạo lộ trình học tập
- AWS S3: Lưu trữ tài liệu học tập, video bài giảng

(b) Caching Layer:

- Redis: Cải thiện hiệu suất và giảm tải cho database

Phân tích nguyên nhân:

- Phân chia trách nhiệm rõ ràng: Mỗi dịch vụ có một nhiệm vụ cụ thể, dễ bảo trì và phát triển. Ví dụ, FastAPI quản lý API, Langchain xử lý AI, PostgreSQL và Neo4j quản lý dữ liệu.
- Dễ mở rộng: Mỗi dịch vụ có thể mở rộng riêng lẻ khi tải tăng lên, như tăng số lượng instances của FastAPI khi số người dùng tăng.
- Triển khai linh hoạt: Thay đổi hoặc thêm công nghệ mới dễ dàng mà không ảnh hưởng đến toàn hệ thống.

Không dùng:

- Monolithic vì khó mở rộng, bảo trì, và tích hợp công nghệ mới.
- Event-driven vì phức tạp khi quản lý trạng thái trực tiếp và đồng bộ.
- Serverless do phụ thuộc nhà cung cấp, chi phí cao khi tải tăng, và hạn chế tùy chỉnh hiệu suất.

Như vậy, ta có luồng hoạt động của hệ thống như sau:

- Người dùng tương tác với frontend Vue.js (Presentation Tier).
- Requests được gửi đến API Gateway trong Application Tier.
- API Gateway xác thực và định tuyến requests đến microservice phù hợp.
- Microservices trong Application Tier xử lý business logic:
 - Gọi đến Persistence Layer trong Data Tier để lưu trữ hoặc truy xuất dữ liệu.
 - Sử dụng Caching Layer để tối ưu hiệu suất truy vấn.
- Kết quả được trả về cho frontend thông qua API Gateway.

Tóm lại, kiến trúc kết hợp ưu điểm của Microservices và Multitier tận dụng tối đa các công nghệ đã được chọn để xây dựng một hệ thống học tập trực tuyến thông minh, có khả năng mở rộng cao và linh hoạt. Việc phân chia rõ ràng giữa các tiers và microservices giúp dễ dàng phát triển, bảo trì và mở rộng hệ thống trong tương lai.

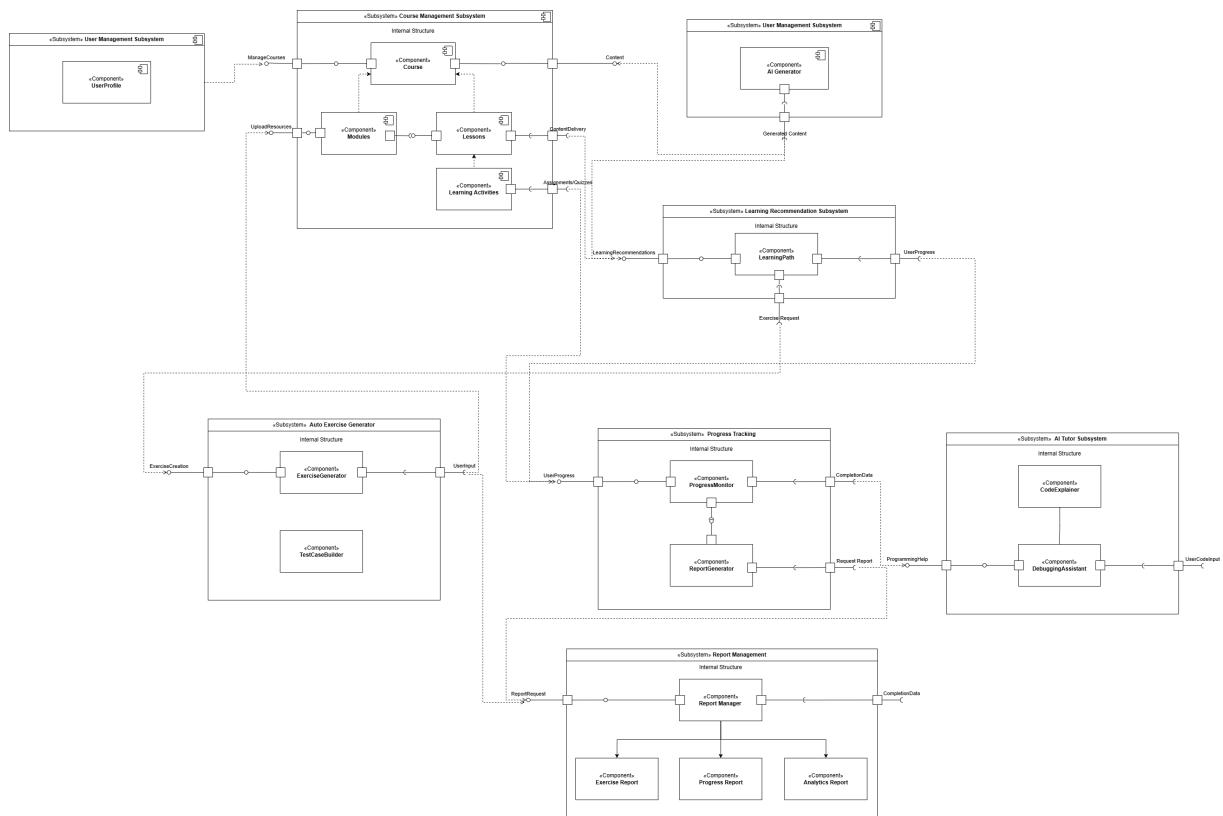


Figure 5.12: Component-based Diagram cho toàn bộ hệ thống

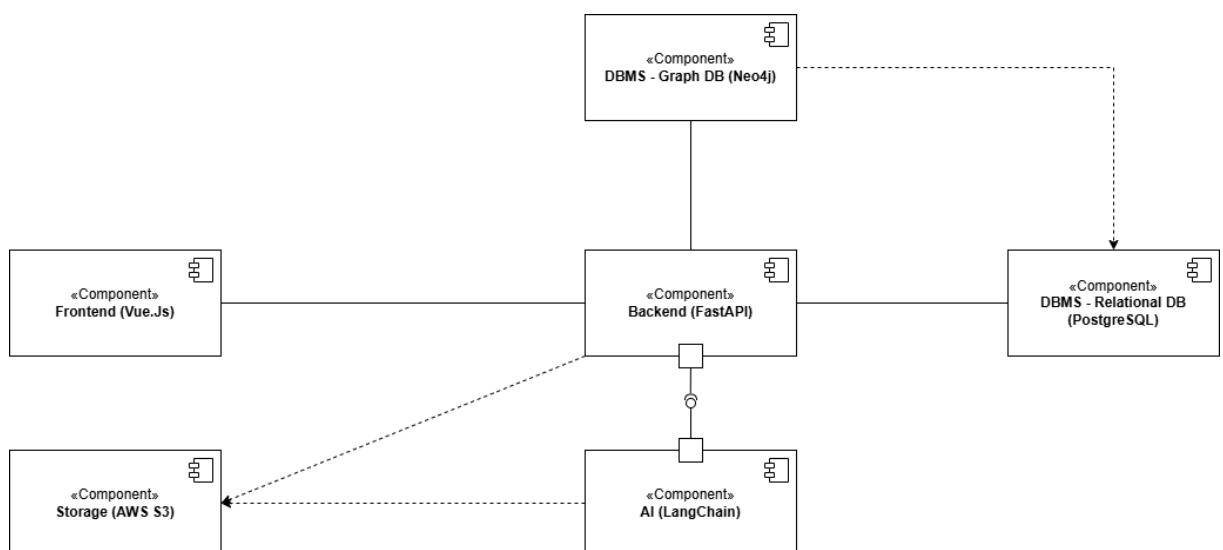


Figure 5.13: Component-based Diagram cho những công nghệ được chọn

5.3 Database

5.3.1 EER diagram

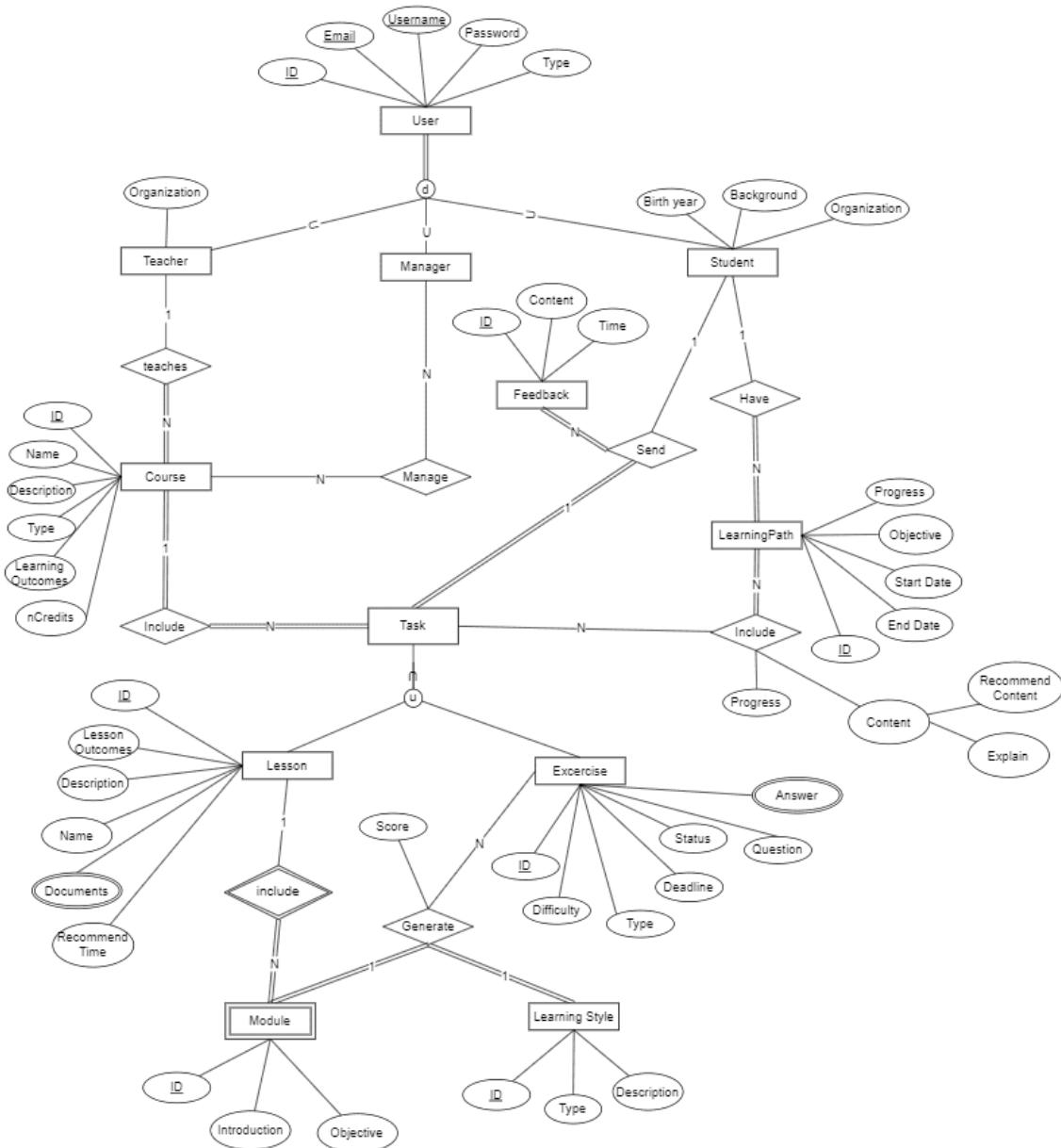


Figure 5.14: Sơ đồ EER của hệ thống

1. User:

- User được chia thành 3 loại là: student, teacher, admin. Mỗi người dùng có các thuộc tính cơ bản như: ID, tên đăng nhập, email, mật khẩu và loại người dùng.
- Mỗi user sẽ có một vai trò cụ thể trong hệ thống.

2. Student:



- Mỗi sinh viên có các thuộc tính riêng như mục tiêu học tập, năm sinh, nền tảng học vấn và báo cáo về kết quả học tập.
 - Mỗi sinh viên có thể có một hoặc nhiều lộ trình học tập. Mỗi lộ trình học tập có thuộc tính như các bài học, cấp độ, thời gian, mục tiêu khóa học và ID. Lộ trình này bao gồm các lesson và bài kiểm tra.
3. Teacher: Giáo viên lưu thuộc tính về nơi làm việc và có thể tải lên các khóa học.
 4. Manager: Quản trị viên có thuộc tính về danh sách feedback, knowledgegraph để có khả năng quản lý các kiến thức, xử lý feedback và quản lý người dùng.
 5. Course: Mỗi khóa học bao gồm các thuộc tính như: ID, tên, mô tả, loại khóa học, mục tiêu đầu ra và danh sách các bài học.
 6. Exercise: Bài tập có các thuộc tính như câu hỏi, đáp án, tài liệu liên quan, độ khó và loại bài tập.
 7. AI: có các thuộc tính về: rule(kiểm tra bài tập đúng chủ đề, cấp độ), câu prompt, sử dụng dữ liệu từ KnowledgeGraph để tạo ra các lộ trình học tập và bài tập (Exercise) cho sinh viên

5.3.2 Lược đồ CSDL quan hệ ánh xạ

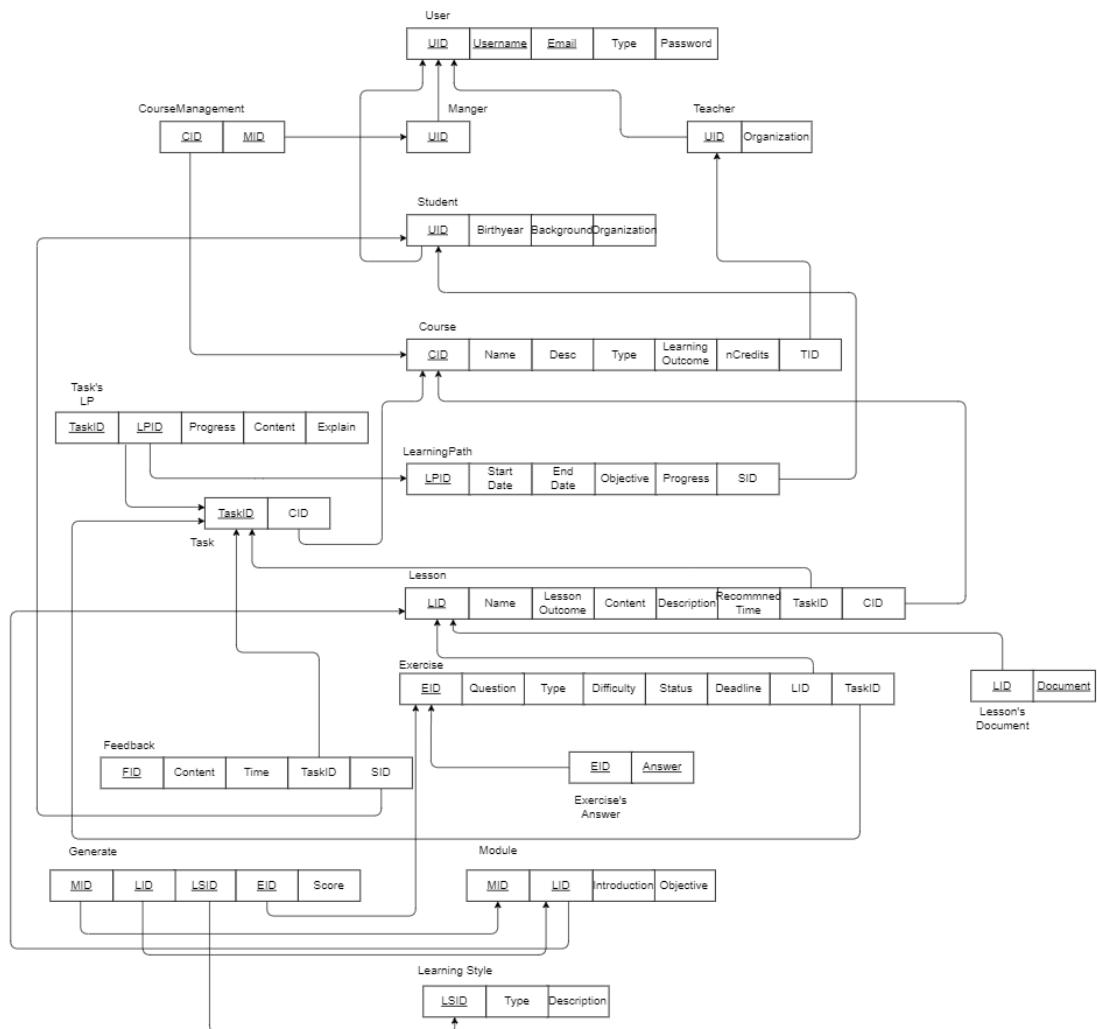


Figure 5.15: Lược đồ CSDL quan hệ ánh xạ của hệ thống

5.3.3 Class diagram

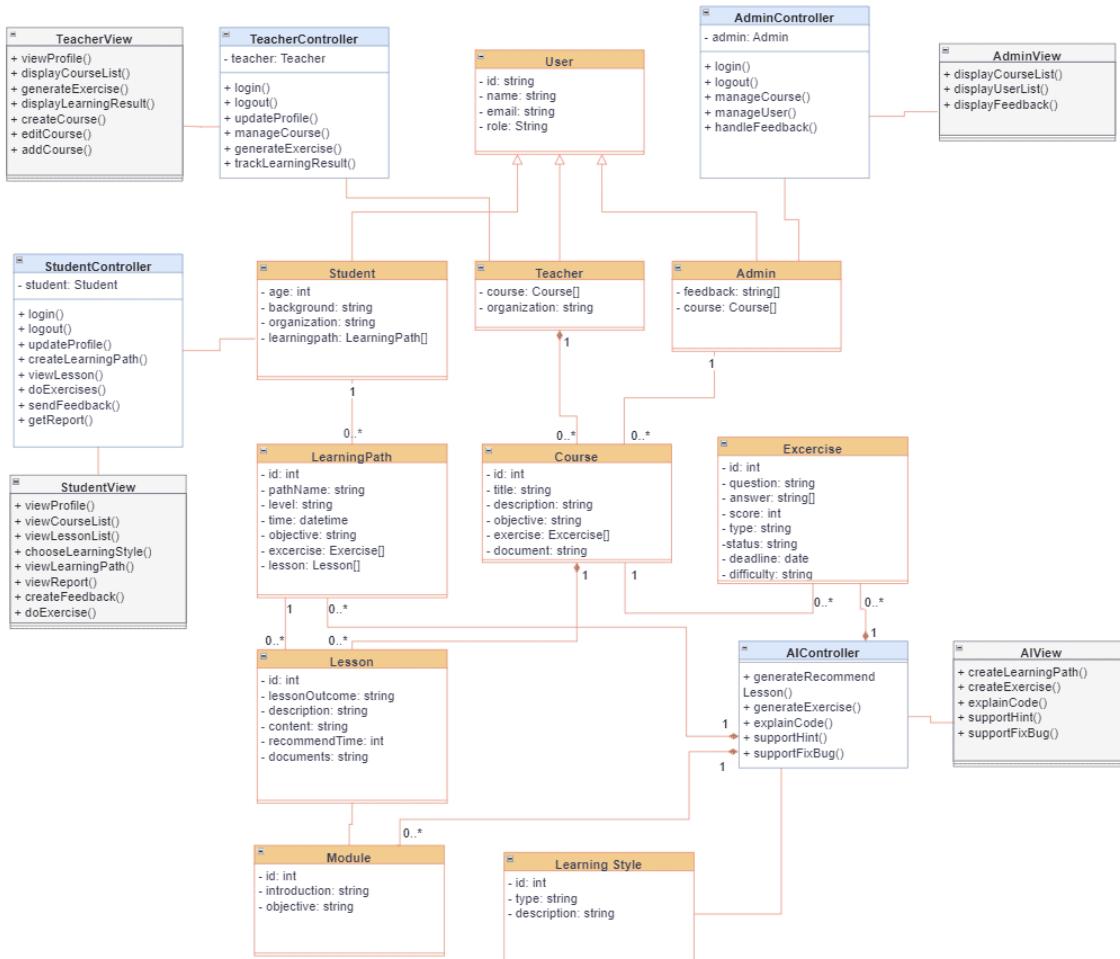


Figure 5.16: Class diagram hiện thực của hệ thống

Class diagram được thiết kế theo mô hình MVC mô tả một hệ thống học tập thông minh với ba vai trò chính là sinh viên, giáo viên và quản trị viên, tất cả đều kế thừa từ lớp người dùng (User). Mỗi người dùng có các chức năng cơ bản như đăng nhập, đăng xuất và cập nhật hồ sơ. Sinh viên chọn theo học khóa học của giảng viên, thực hiện bài tập, và theo dõi tiến độ học tập của mình. Lộ trình học tập bao gồm bài tập được cá nhân hóa cho từng sinh viên. Giáo viên chịu trách nhiệm tạo và quản lý các khóa học, bao gồm việc thêm bài tập, theo dõi kết quả học tập của sinh viên, và tạo ra các bộ câu hỏi kiểm tra. Quản trị viên có quyền quản lý toàn bộ hệ thống, bao gồm khóa học và người dùng, cũng như giải quyết các thắc mắc từ người dùng. AI được sử dụng để hỗ trợ quá trình học tập thông qua việc tự động tạo lộ trình học tập và bài tập. AI dựa vào dữ liệu từ các tài liệu học tập, câu hỏi và đáp án có sẵn, cũng như các bài tập lập trình để cá nhân hóa trải nghiệm học cho người dùng.

5.3.4 Sitemap

5.3.4.1 Sinh viên

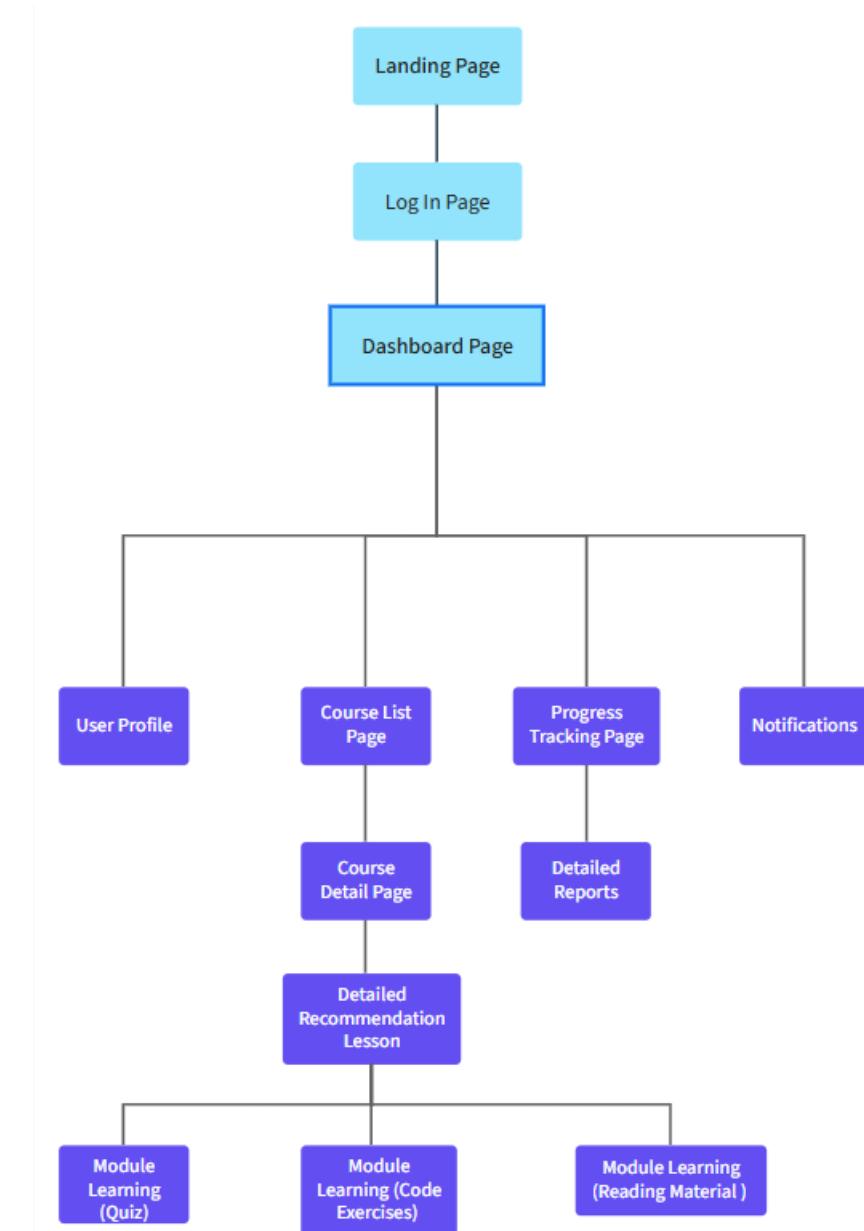


Figure 5.17: Sitemap cho đối tượng Sinh viên

5.3.4.2 Giảng viên

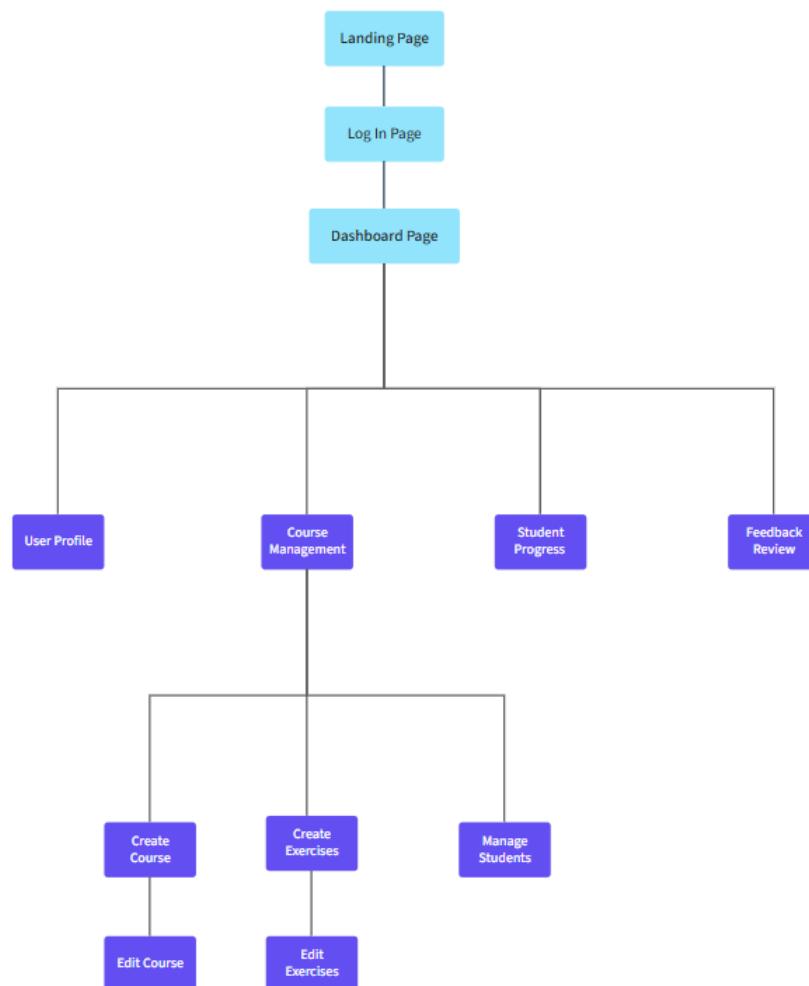


Figure 5.18: Sitemap cho đối tượng Giảng viên

5.3.4.3 Manager

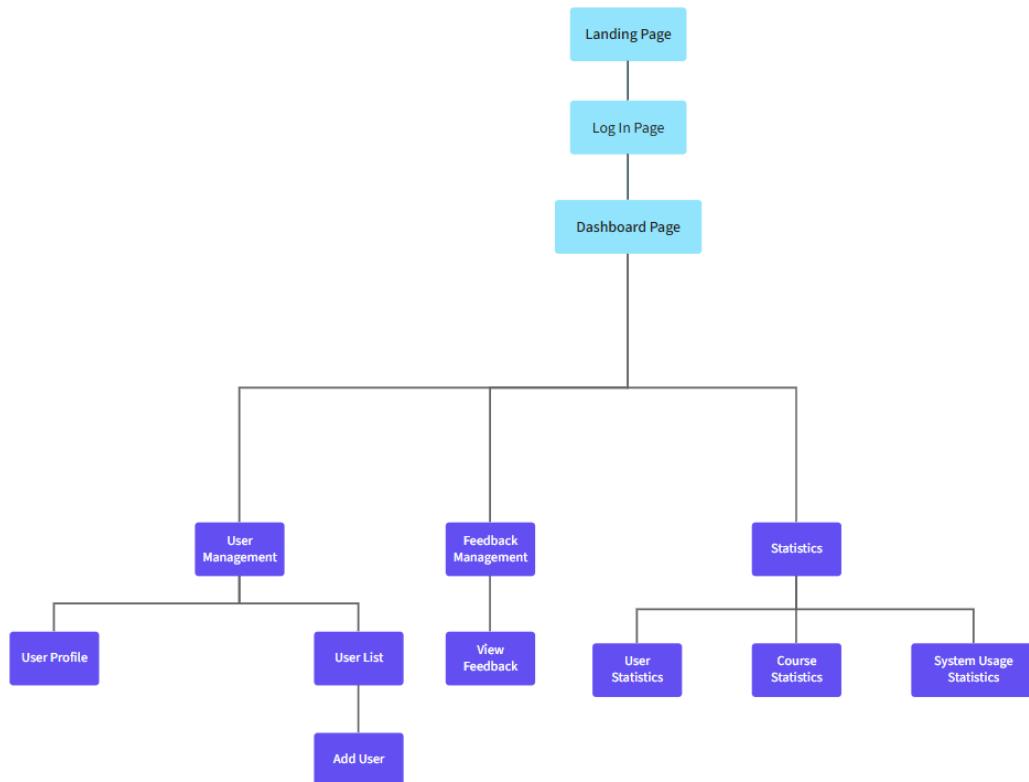


Figure 5.19: Sitemap cho đối tượng Quản trị viên

5.4 Giải thuật

5.4.1 Giải thuật đề xuất bài học

Khi người dùng cung cấp mục tiêu học tập, hệ thống sử dụng LLM để phân tích và xác định các khái niệm quan trọng. Các khái niệm này được trích xuất từ mục tiêu học tập thông qua LLM, và được lưu trữ trong cơ sở dữ liệu đồ thị (graph database) Neo4j. Neo4j sẽ là cơ sở dữ liệu đồ thị để lưu trữ và quản lý các khái niệm học tập, tài nguyên học tập, và mối quan hệ giữa chúng. Đồ thị này bao gồm hai loại thực thể chính: **Node** (điểm nút) và **Relationship** (mối quan hệ). Dưới đây là chi tiết về cấu trúc đồ thị:

Các loại node trong hệ thống bao gồm:

- **Concept:** Mỗi node sẽ đại diện cho một khái niệm học tập.
- **Learning Resource:** Các node tài nguyên học tập sẽ đại diện cho các bài giảng, sách, video, tài liệu học tập, ví dụ: "Video bài giảng về phương trình bậc hai", "Bài viết về đạo hàm".



- **Learner:** Node người học sẽ chứa thông tin về người học, bao gồm các khái niệm mà họ đang học và mức độ thành thạo của họ đối với từng khái niệm.

Các mối quan hệ giữa các node sẽ bao gồm:

- **COVERS:** Quan hệ giữa tài nguyên học tập và khái niệm mà tài nguyên đó đề cập đến. Ví dụ: Một video có thể "COVERS" khái niệm "Phương trình bậc hai". Mỗi quan hệ này sẽ có một thuộc tính difficulty (độ khó).
- **LEARNS:** Quan hệ giữa người học và khái niệm mà họ đang học. Ví dụ: Người học "LEARNS" khái niệm "Lập trình Python". Mỗi quan hệ này sẽ có một thuộc tính proficiency (độ thành thạo) để xác định người học đã nắm vững khái niệm đó đến mức nào.

Để trích xuất các khái niệm liên quan, hệ thống sử dụng kỹ thuật vector embedding để chuyển đổi khái niệm và tài nguyên học tập thành các vector đại diện. Mỗi khái niệm và tài nguyên học tập sẽ có một vector tương ứng trong không gian nhiều chiều.

- **Vector Embedding:** Mỗi khái niệm và tài nguyên học tập được chuyển thành một vector nhờ các phương pháp như Word2Vec, BERT hoặc các mô hình ngôn ngữ khác. Các vector này giúp biểu diễn sự tương đồng ngữ nghĩa giữa các khái niệm và tài nguyên.
- **Cosine Similarity:** Sau khi có vector cho mỗi khái niệm và tài nguyên học tập, hệ thống sử dụng cosine similarity để tính toán sự tương đồng giữa các khái niệm mà người học cần thành thạo và các tài nguyên học tập có sẵn. Cosine similarity được tính bằng công thức:

$$\text{Cosine Similarity} = \frac{\mathbf{A} \cdot \mathbf{B}}{\|\mathbf{A}\| \cdot \|\mathbf{B}\|}$$

5.4.2 Workflow của LLM và Function Calling

LLM đóng vai trò quan trọng trong việc trích xuất các khái niệm và giải thích tài nguyên học tập. LLM sẽ thực hiện các tác vụ sau:

5.4.2.1 Phân tích Mục tiêu và Xác định Khái niệm

Khi nhận được mục tiêu học tập từ người dùng, LLM sẽ phân tích mục tiêu này và xác định các khái niệm cần thiết. Ví dụ, nếu mục tiêu học tập là "Làm chủ lập trình Python", LLM có thể trích xuất các khái niệm như "Biến", "Câu lệnh điều kiện", "Vòng lặp", ...

5.4.2.2 Function Calling để Trích xuất Dữ liệu

Sau khi LLM xác định các khái niệm, LLM sẽ sử dụng function calling để gọi các hàm trong hệ thống nhằm trích xuất các thông tin sau:



- Thông tin người học: LLM gọi hàm để lấy các khái niệm mà người học đã học, cùng với mức độ thành thạo đối với các khái niệm này.
- Thông tin tài nguyên học tập: LLM gọi hàm để lấy danh sách các tài nguyên học tập có sẵn, bao gồm các tài nguyên đã được gán các khái niệm và độ khó của chúng.

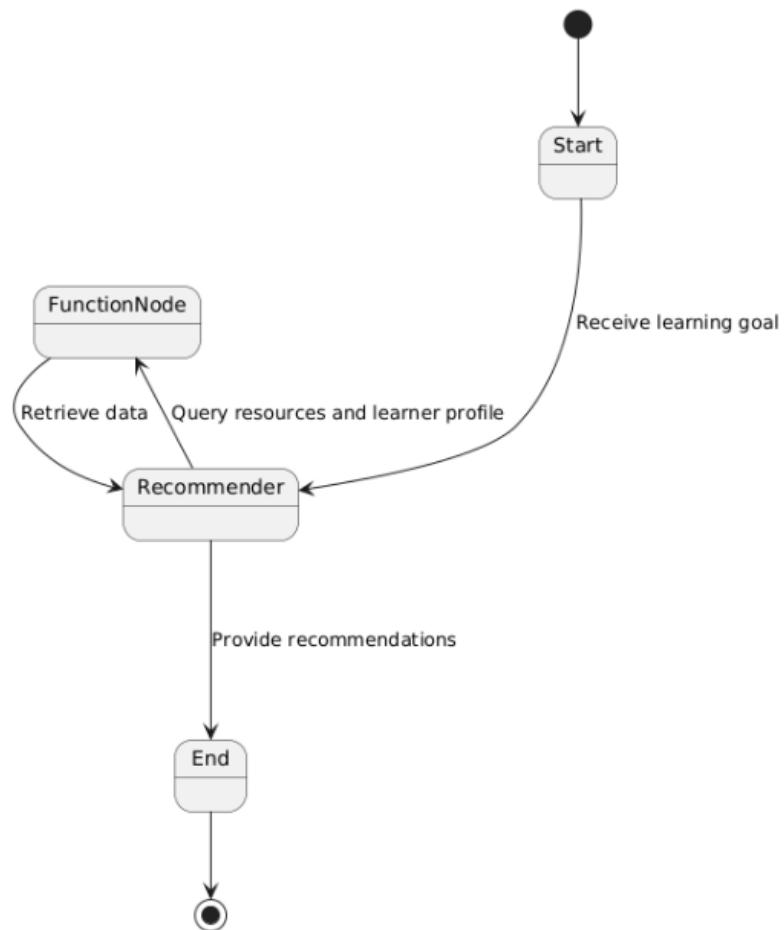
5.4.2.3 Trích xuất và Đề xuất Tài nguyên Học tập

LLM sẽ sử dụng thông tin về người học và tài nguyên học tập để tính toán độ phù hợp giữa người học và tài nguyên học tập. Cụ thể:

- Dựa vào cosine similarity giữa các vector của khái niệm và tài nguyên học tập.
- Xác định những tài nguyên học tập phù hợp nhất cho người học dựa trên mức độ thành thạo của họ và độ khó của tài nguyên học tập.

5.4.2.4 Giải thích kết quả

Sau khi đã chọn ra các tài nguyên học tập, LLM sẽ tiếp tục sử dụng function calling để tạo ra giải thích chi tiết về cách mỗi tài nguyên giúp người học cải thiện khả năng thành thạo các khái niệm cần thiết. LLM sẽ trả về các giải thích dạng tự nhiên cho người học.



5.5 Thiết kế giao diện

5.5.1 Landing Page

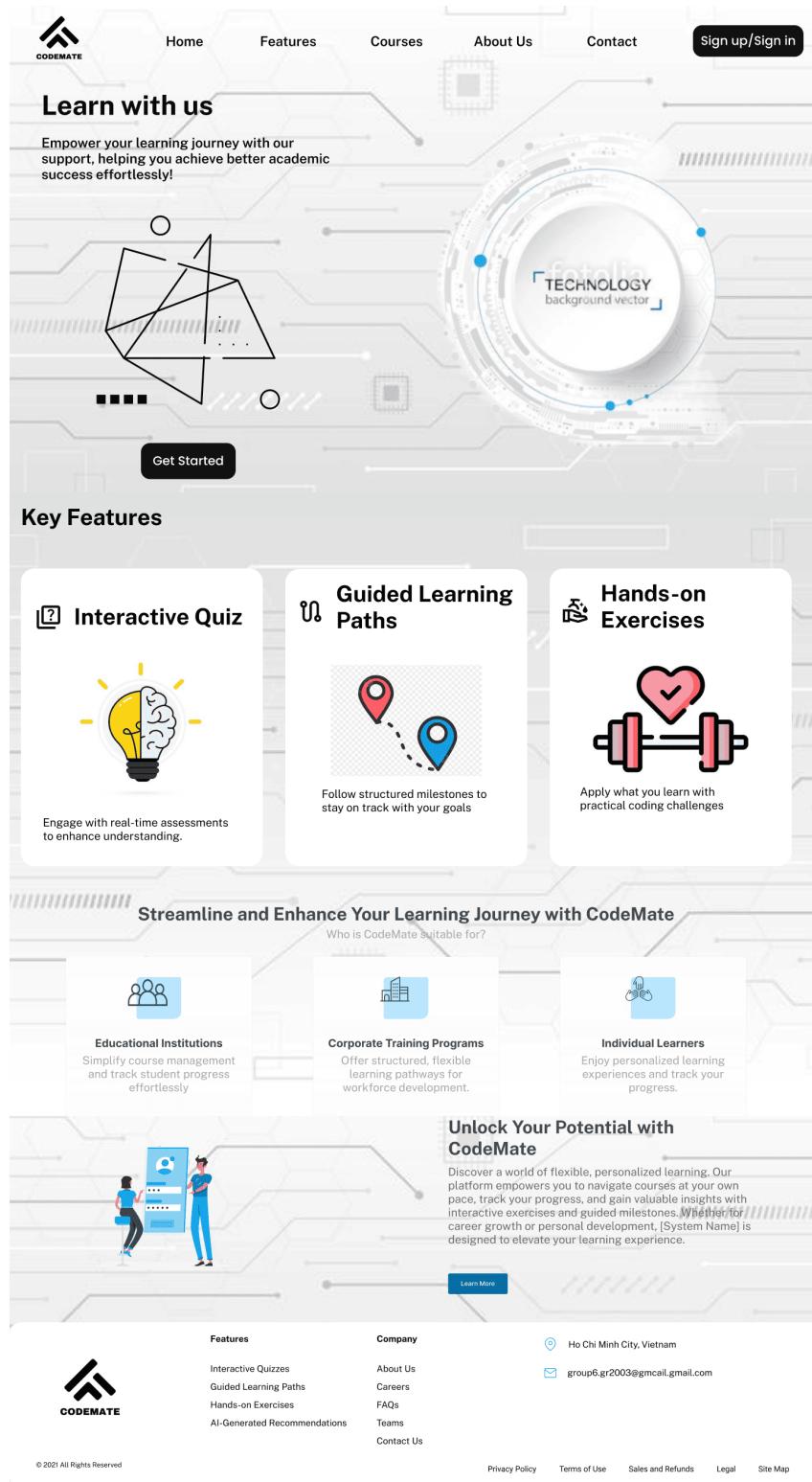


Figure 5.20: Landing Page

Landing Page mô tả tổng quát về website hỗ trợ học tập **CODEMATE**.

5.5.2 Authentication

5.5.2.1 Log In Screen

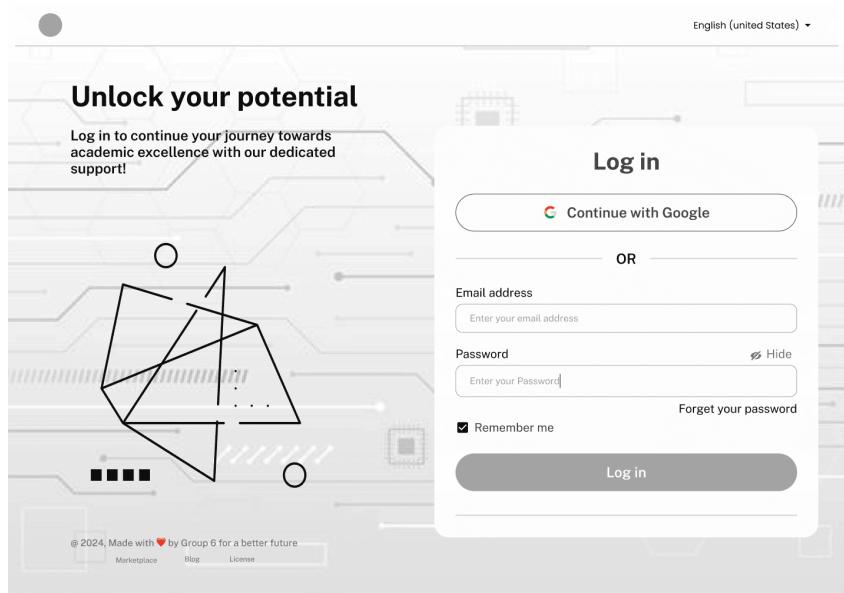


Figure 5.21: Log In Screen

Trang đăng nhập hệ thống, cho phép người dùng đăng nhập vào hệ thống bằng Email của mình hoặc qua phương thức đăng nhập bằng bên thứ ba (**Google**). Hệ thống sẽ nhận diện tên miền riêng của Email để xác định tính xác thực của Email trường Đại học do người dùng cung cấp. Nếu người dùng không dùng Email do trường Đại học cung cấp, sẽ không thể truy cập vào hệ thống. Ngoài ra, role của người dùng cũng sẽ được hệ thống phân loại dựa vào Email Address của người dùng và cơ sở dữ liệu của trường Đại học cung cấp cho hệ thống.

5.5.2.2 Forgot Password Screen

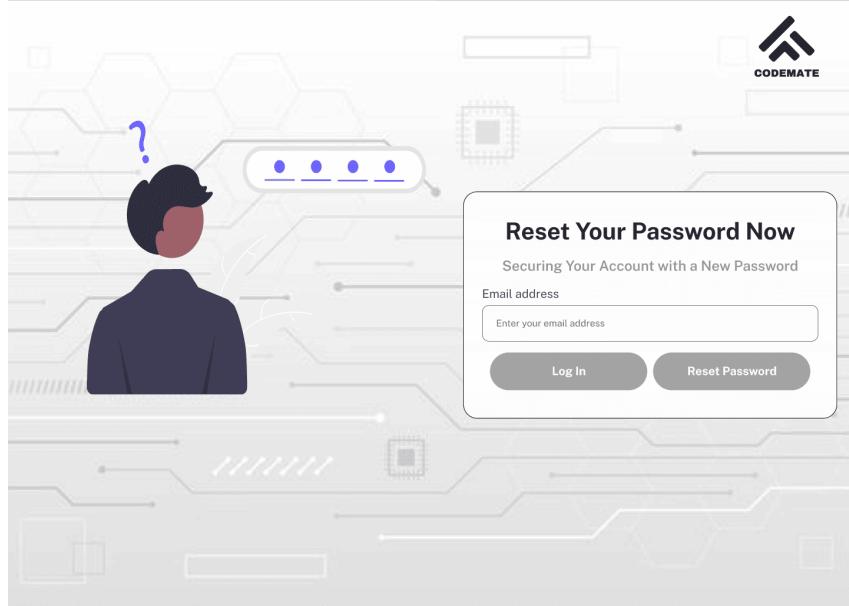


Figure 5.22: Forgot Password Screen

5.5.2.3 Reset Password Screen

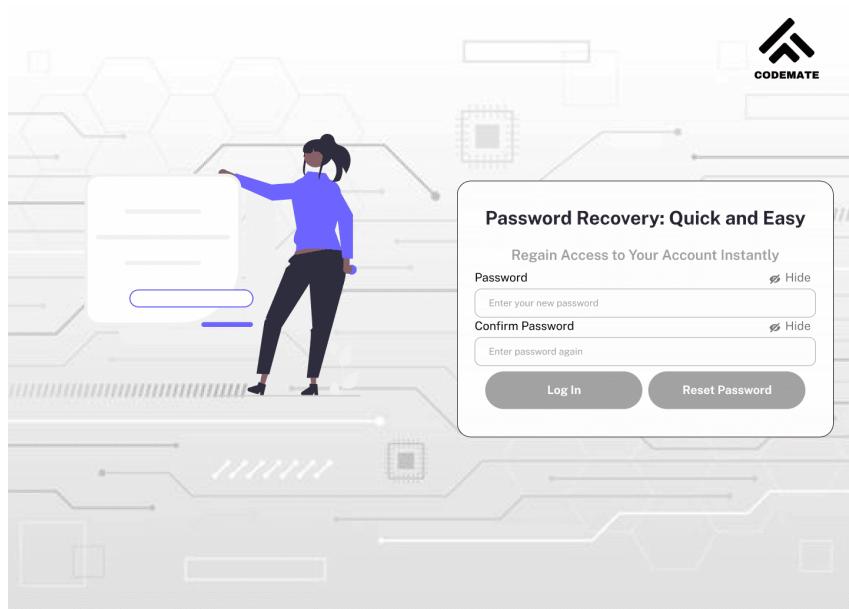


Figure 5.23: Reset Password Screen

Các trang "**Forgot Password**" và "**Reset Password**" hỗ trợ người dùng trong trường hợp quên mật khẩu truy cập vào hệ thống và muốn lấy lại mật khẩu của mình thông qua địa chỉ Email.



5.5.3 Course Management (Student)

5.5.3.1 Courses List Screen

The screenshot shows the 'CODEMATE' platform interface. On the left, there's a sidebar with a search bar, dashboard, courses (selected), notifications, progress tracking, exercises, help, and settings. The main area displays 'Recent Courses' with three cards: 'The Science of Well-Being', 'Python for Everybody Specialization', and 'The Science of Data'. Below this is a section for 'My Courses' with four course cards: 'Python for Everybody Specialization', 'Introduction to Data Science', 'Product Management Basic - Course', and 'BM Data Science Professional Certificate'. Each course card includes a thumbnail, title, duration, student count, learning outcomes, professor information, and 'View Course' and 'View more' buttons.

Figure 5.24: Courses List Screen

Trang danh sách khóa học, hiển thị danh sách các khóa học mà sinh viên đã đăng ký tại học kỳ đó. Đặc biệt là các khóa học liên quan đến lĩnh vực "**Khoa học máy tính**". Sinh viên có thể xem các khóa học mình đã và đang học hoặc các khóa học truy cập gần đây. Mỗi khóa học hiển thị các thông tin như: **Tên khóa học**, **Tên giảng viên**, **Thời gian bắt đầu**, **Thời gian kết thúc**, **Đầu ra môn học** và tác vụ "**Xem chi tiết khóa học**".

A modal window titled 'Learning Outcomes' is shown. It features a flag icon and a close button. The content lists five learning outcomes with icons: a red circle for 'Develop data-driven decision-making skills.', another red circle for 'Understand the basics of data science and its applications.', a green checkmark for 'Analyze and visualize data using Python and relevant libraries.', and a green checkmark for 'Apply statistical methods and machine learning algorithms to solve real-world problems.'

Figure 5.25: Modal Learning Outcomes

Sinh viên có thể xem chi tiết "**Đầu ra môn học**" bằng nút "**View more**"



5.5.3.2 Detailed Course Screen

Trang thông tin chi tiết khóa học, field Description, cho phép sinh viên xem mô tả "**Đầu ra môn học**" do giảng viên cung cấp. Ngoài ra, các thông tin của khóa học như: **Tên môn học**, **Phản trǎm hoàn thành hiện tại khóa học của sinh viên**, **Label trạng thái môn học**, **Số unit của khóa học**, **Tên giảng viên** được hiển thị chung ở một component. Mỗi khóa học sẽ có 3 fields hỗ trợ cho các thông tin: **Description**, **Lessons**, **Exercises**.

The screenshot shows the 'Course Details' page for a 'Data Science' course. On the left is a sidebar with navigation links: Search, Dashboard, Courses (highlighted), Notifications, Progress Tracking, Exercises, Help, and Setting. At the top right are icons for notifications, messages, and user profile (Swetha shankares). The main content area has a large blue header image with a person icon. Below it, the course title 'Data Science' is displayed with a progress bar showing '65%' completed. To the right of the progress bar are completion date '22/8/2024-25/5/2024', chapters '10 Chapters', and author 'Adam Levi'. A 5-star rating is shown. A 'Recommend Lessons' section features a call-to-action button 'Join our Recommend Learning – tailored lessons to reach your goals, one step at a time.' Below the main title, there's a 'Learning Outcomes' section with a bulleted list:

- Understand the basics of data science and its applications.
- Analyze and visualize data using Python and relevant libraries.
- Apply statistical methods and machine learning algorithms to solve real-world problems.
- Develop data-driven decision-making skills.

Figure 5.26: Detailed Course Screen - Description

Field Description chứa thông tin liên quan đến mô tả khóa học và đầu ra môn học do giảng viên cung cấp.

This screenshot shows the 'Lessons' section of the 'Data Science' course page. It includes the same sidebar and header as Figure 5.26. The main content area displays three chapters under the 'Lessons' tab:

Chapter 1: Introduction to Data Science	5 documents	Download	Print	Share
Covers techniques for handling missing data, outliers, and transforming data for analysis.	5 documents	Download	Print	Share
Chapter 1: Introduction to Data Science	5 documents	Download	Print	Share
Covers techniques for handling missing data, outliers, and transforming data for analysis.	5 documents	Download	Print	Share
Chapter 1: Introduction to Data Science	5 documents	Download	Print	Share
Covers techniques for handling missing data, outliers, and transforming data for analysis.	5 documents	Download	Print	Share

Figure 5.27: Detailed Course Page - Lessons



Field Lessons chứa các lessons được chia bởi giảng viên. Mỗi lesson được hiển thị với thông tin: **Tên lesson và Số tài liệu tham khảo được giảng viên đăng tải**

The screenshot shows the 'Data Science' course details page. On the left sidebar, there are links for Dashboard, Courses (selected), Notifications, Progress Tracking, Exercises (selected), Help, and Setting. The main content area displays the course title 'Data Science', completion percentage '65%', and a progress bar indicating 'Completed'. It also shows the date range '22/8/2024 - 25/5/2024', the number of chapters '10 Chapters', and the instructor 'Adam Levi'. Below this, there are five exercises listed:

Description	Due Date	Points	Action
Assignment 1 Data Science	30/10/2024 23:59:59	100 points	Start
Quiz 1: Create Plan Data Science	N/A	100 points	Start
Quiz 2 Data Science	30/10/2024 23:59:59	100 points	Start
Quiz 3 Data Science	N/A	100 points	Start
Quiz 4 Data Science	30/10/2024 23:59:59	100 points	Start

Figure 5.28: Detailed Course Page - Exercises

Field Exercises chứa các bài Quiz/Assignment do giảng viên hoặc hệ thống cung cấp hỗ trợ cho lesson/khoa học đó.

The screenshot shows the 'Data Science' course details page. The sidebar and course summary are identical to Figure 5.28. The main content area displays four recommended lessons:

Lesson Description	Actions
Chapter 3: Introduction to Data Science Covers techniques for handling missing data, outliers, and transforming data for analysis.	View, Edit, Delete, Preview
Chapter 5: Introduction to Data Science Covers techniques for handling missing data, outliers, and transforming data for analysis.	View, Edit, Delete, Preview
Chapter 7: Introduction to Data Science Covers techniques for handling missing data, outliers, and transforming data for analysis.	View, Edit, Delete, Preview
Chapter 8: Introduction to Data Science Covers techniques for handling missing data, outliers, and transforming data for analysis.	View, Edit, Delete, Preview

Figure 5.29: Detailed Course Page - Recommend Lessons

Field Recommend Lessons Chứa các lessons mà hệ thống đã đề xuất cho sinh viên.



5.5.3.3 Define Goal Modal

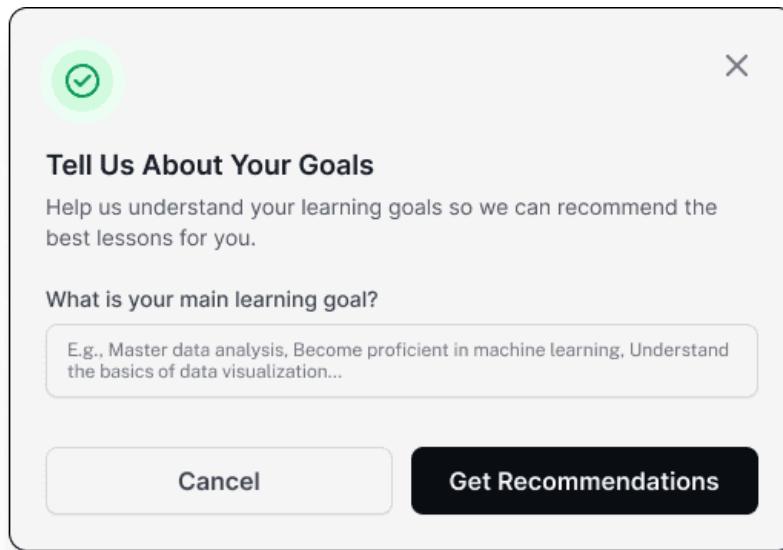


Figure 5.30: Modal để lấy mục tiêu của sinh viên

Sinh viên có thể cung cấp mục tiêu học tập của mình thông qua modal ở trang chi tiết khóa học

5.5.3.4 Modules

Sau khi chọn mục tiêu học tập và được hệ thống đề xuất lesson, sinh viên sẽ được truy cập vào lesson được đề xuất với giao diện chi tiết lesson như sau:



The screenshot shows a user profile for 'Swetha shankaresh' (Student) on the left. The main content area displays a lesson titled 'Chapter 1: Introduction to Data Science'. Key details include a progress bar at 65%, a recommended time of 1.5 hours, and 4 modules completed. A 'Completed' badge is present. To the right, 'Learning Outcomes' are listed:

- After completing this lesson, you'll be able to:
- Explain the process of semantic analysis.
- Perform type checking and type inference.
- Implement symbol tables and manage variable scopes.
- Handle errors that arise during semantic analysis.

Below this, 'Recommend Content:' is described as focusing on semantic analysis, lexical and syntax analysis, and symbol tables. 'Explain:' highlights the strong foundation in lexical and syntax analysis and how it adds meaning to the parsed program. 'Modules:' are listed as 'Check for type compatibility', 'Verify variable declarations', and 'Ensure followed scope rules'.

Figure 5.31: Chi tiết Lesson

Lesson đề xuất sẽ cung cấp đầy đủ thông tin về **Learning Objectives**, **Recommend Content**, **Lời giải thích cho sự đề xuất và các modules cần học**

5.5.3.5 Choices of Learning Styles

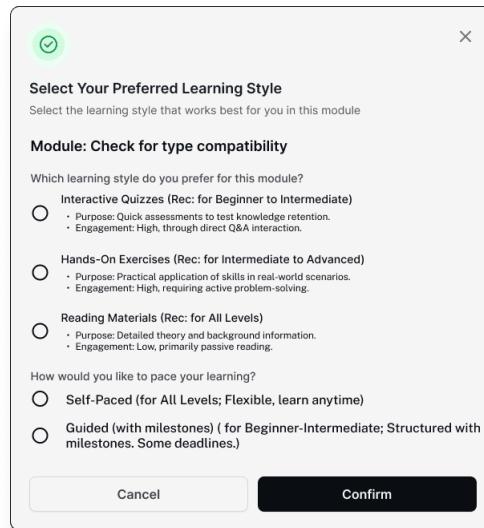


Figure 5.32: Modal cho phép sinh viên lựa chọn kiểu học phù hợp



Sinh viên cung cấp thông tin về cách học mong muốn và tốc độ phù hợp với bản thân ở Modal này để quyết định cách học cho từng module

The screenshot shows the CODEMATE learning platform interface. On the left, there's a sidebar with navigation links: Dashboard, Courses (which is selected and highlighted in grey), Notifications, Progress Tracking, Exercises, Help, and Setting. At the bottom of the sidebar is a user profile for 'Swetha shankaresh' with the status 'Student'. The main content area has a search bar at the top right. Below it, a 'Lesson Details' section shows the title 'Self-Paced with Interactive Quiz Learning Style: "Check for Type Compatibility"'. Underneath the title, there's a welcome message: "'Welcome to the 'Check for Type Compatibility' module! This module uses interactive quizzes to help you test your knowledge. Work through each quiz at your own speed, and check your answers along the way. You can retake any quiz for better understanding.'". To the right of this message is a 'Learning Objectives' section with three bullet points: 'Recognize type compatibility rules.', 'Apply type conversion methods.', and 'Solve common type compatibility problems.'. A large 'Do Quiz' button is located below the text. At the very bottom of the main content area is another user profile for 'Swetha shankaresh'.

Figure 5.33: Giao diện khi sinh viên chọn học theo cách học làm Quiz với tiến độ tự do



The screenshot shows the CodeMate platform interface. On the left is a sidebar with icons for Dashboard, Courses (which is selected), Notifications, Progress Tracking, Exercises, Help, and Setting. At the bottom of the sidebar is a profile picture of a student named Swetha shankaresh. The main content area has a search bar at the top. Below it, a section titled "Lesson Details" contains the title "Self-Paced with Reading Materials: 'Check for Type Compatibility'". A welcome message says: "Welcome to the 'Check for Type Compatibility' module! Here you'll find detailed reading materials. Go through each section at your own pace and absorb the concepts." To the right, under "Learning Objectives", there are two bullet points: "Read and understand the principles of type compatibility." and "Explore examples and scenarios of type mismatches." Below this is a "Reflection Questions" button. The central part of the screen is a rounded rectangle containing the first section of the lesson, titled "Section 1: What is Type Compatibility?". It includes an "Introduction" about type compatibility in TypeScript, "Core Concepts" (Structural Typing), and an example code snippet:

```
interface Person {  
    name: string;  
}  
  
let person: Person = { name: "Alice" };  
let anotherPerson = { name: "Bob", age: 30 }; // Additional properties are allowed  
  
person = anotherPerson; // Compatible because 'anotherPerson' has at least 'name'.
```

Below the code, a note states: "Here, anotherPerson is compatible with Person even though it has an extra property (age), because it matches the required structure of Person."

Figure 5.34: Giao diện khi sinh viên chọn học theo cách học đọc tài liệu với tiến độ tự do

The screenshot shows the CodeMate platform interface, similar to Figure 5.34 but with a different module. The sidebar and user profile are identical. The main content area has a search bar at the top. Below it, a section titled "Lesson Details" contains the title "Self-Paced with Hands-On Exercises: 'Check for Type Compatibility'". A welcome message says: "Welcome to the 'Check for Type Compatibility' module! This module includes hands-on coding exercises. You'll have the freedom to practice type compatibility scenarios and test your code as you go." To the right, under "Learning Objectives", there are three bullet points: "Implement type assertions and conversions.", "Resolve type compatibility issues through code.", and "Understand type safety through practical coding." The central part of the screen is a rounded rectangle containing the first section of the lesson, titled "Exercises List". It lists two exercises:

1. Write a function that takes an any type parameter and returns it as a number. Use type assertion.
2. The following code has a type compatibility error. Identify and fix it.

At the bottom of the exercise list is a "Do all exercises" button.

Figure 5.35: Giao diện khi sinh viên chọn học theo cách học làm bài tập Coding với tiến độ tự do

5.5.4 User Profile

5.5.4.1 Profile Information

The screenshot shows the 'My profile' page with a sidebar containing links for Dashboard, Courses, Notifications, Progress Tracking, Help, and Setting. The main area is titled 'USER INFORMATION' with fields for Username, Email address, First name, Last name, Birth Year, Organization, and Phone Number. It includes a 'Change Image' button next to a placeholder profile picture and a 'Change Password' button.

(a) Trang thông tin tài khoản

The screenshot shows the same 'My profile' page as above, but the 'Edit' button at the top right is highlighted in blue, indicating the user is in edit mode to update their information.

(b) Chính sửa thông tin tài khoản

Trang "Thông tin tài khoản" cung cấp các trường thông tin cá nhân của người dùng bao gồm: Tên người dùng (Username), Địa chỉ email (Email address), Tên (First name), Họ (Last name), Năm sinh (Birth Year), Trường (Organization), và Số điện thoại (Phone Number). Người dùng có thể thay đổi hình ảnh đại diện bằng cách nhấp vào nút Change Image và cập nhật mật khẩu thông qua nút Change Password. Tất cả thông tin có thể được chỉnh sửa bằng cách nhấn vào nút Edit.

5.5.4.2 Change Password, Avatar

The screenshot shows the 'My profile' page with a sidebar containing links for Dashboard, Courses, Notifications, Progress Tracking, Help, and Setting. The main area is titled 'USER INFORMATION' with fields for Old Password, New Password, and Confirm Password. It includes a 'Change Image' button next to a placeholder profile picture and a 'Change Password' button.

(a) Thay đổi mật khẩu

The screenshot shows the 'My profile' page with a sidebar containing links for Dashboard, Courses, Notifications, Progress Tracking, Help, and Setting. The main area is titled 'USER INFORMATION' with a placeholder for a profile picture and a 'Choose File' button. A message says 'Please upload square image size less than 30MB'. It includes a 'Save' button and a 'Change Image' button.

(b) Thay đổi avatar

Trang "Đổi mật khẩu" cho phép người dùng thay đổi mật khẩu bằng cách nhập mật khẩu hiện tại, mật khẩu mới, và xác nhận lại mật khẩu mới. Sau khi điền đầy đủ thông tin, người dùng nhấn nút "Đổi mật khẩu" để hoàn tất. Nếu có lỗi hoặc thành công, hệ thống sẽ hiển thị thông báo tương ứng. Trang "Đổi ảnh đại diện" cho phép người dùng tải lên ảnh mới từ thiết bị của mình và nhấn "Lưu thay đổi" để cập nhật ảnh đại diện.



5.5.5 Dashboard Screen

The dashboard is divided into several sections:

- Left Sidebar:** Labeled "CODEMATE". Includes a search bar ("Search...") and links to "Dashboard", "Courses", "Notifications", "Progress Tracking", "Exercises", "Help", and "Setting".
- Welcome Area:** Displays a welcome message for "Swetha shankaresh", stating "Glad to see you again! Keep up the momentum in your Data Science Basics course." It also includes a "Need help?" button and a "Send Feedback" button.
- Weekly Progress Summary:** Shows a circular progress bar at 65% completion for the "Data Science" category, with a note that 2 lessons are completed.
- Courses Section:** A table titled "Courses" showing three entries:

Name	Members	COMPLETION
Chakra Soft UI Version	3 members	60%
Add Progress Track	2 members	10%
Fix Platform Errors	2 members	100%

Figure 5.38: Dashboard - Student

Trang Dashboard của sinh viên, mô tả tóm tắt về các khóa học mà sinh viên đã và đang tham gia và một số thông tin về khóa học vừa học ở lần truy cập gần nhất.

5.5.6 Exercise Practice

5.5.6.1 Code Exercise

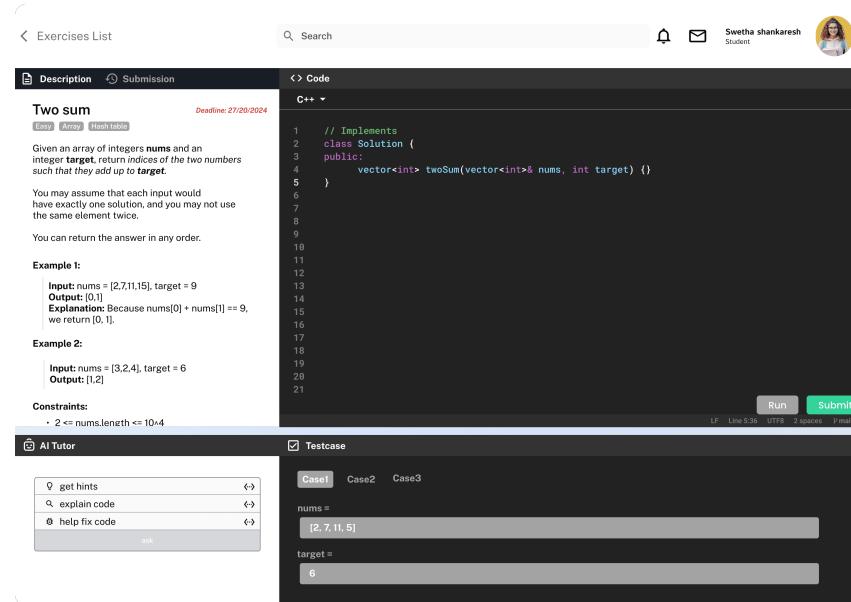


Figure 5.39: Giao diện thực hành code

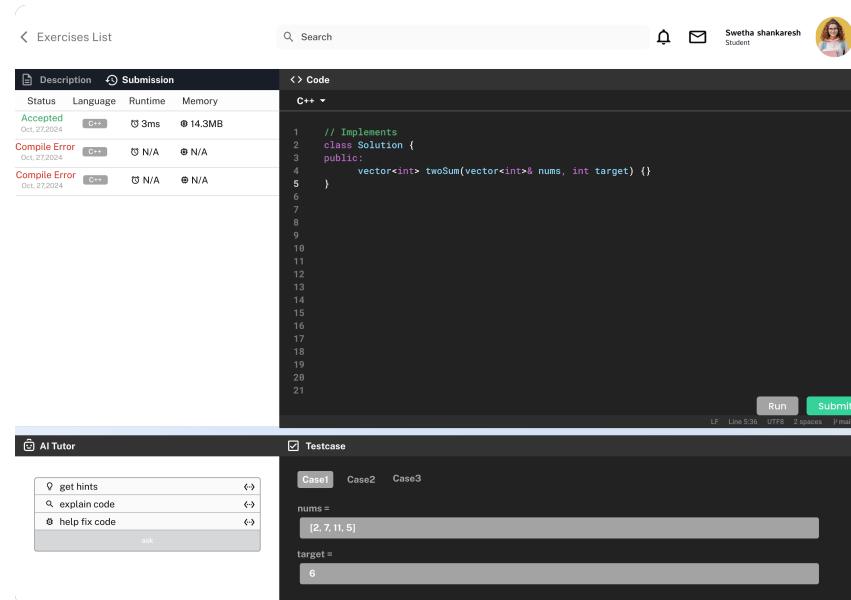


Figure 5.40: Lịch sử submission

5.5.7 AI Tutor

Giao diện AI Tutor được thiết kế gồm ba chức năng chính như sau: cung cấp gợi ý (Get Hints), sửa lỗi (Fix Bug), và giải thích code (Explain Code).

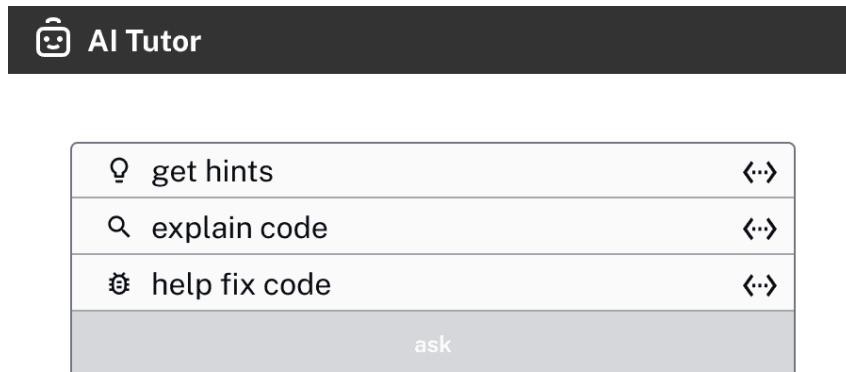


Figure 5.41: Các tính năng của AI Tutor

Trong phần giải thích code, người dùng có thể chọn đoạn mã muốn hiểu rõ hơn; AI Tutor sẽ cung cấp giải thích tổng quát cho đoạn mã đó, và khi người dùng di chuột lên từng dòng, một popup sẽ hiển thị giải thích chi tiết cho từng dòng.

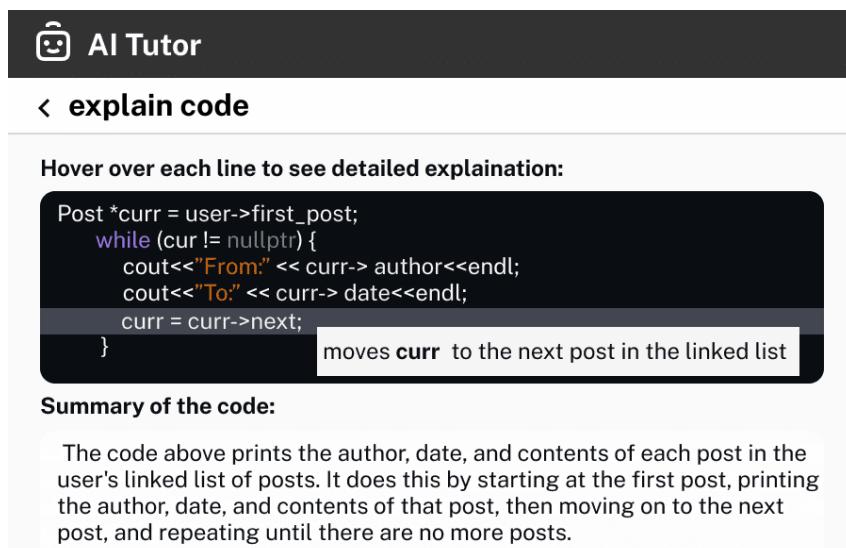


Figure 5.42: Giải thích từng dòng code

Đối với chức năng sửa lỗi, khi người dùng gặp lỗi trong quá trình compile code, AI Tutor sẽ phân tích và giải thích chi tiết nguyên nhân lỗi, đồng thời đề xuất cách khắc phục, giúp người dùng học hỏi từ lỗi sai của mình.



AI Tutor

< help fix code

```
vector<int> numbers = {1, 2, 3, 4, 5};  
for (int i = 0; i <= numbers.size(); i++) {  
    cout << numbers[i] << " ";  
}
```

Fixes: The error occurs on the line `for (int i = 0; i <= numbers.size(); i++)`. When `i` equals `numbers.size()`, the program will try to access an element outside the bounds of the vector `numbers`, causing a runtime error.

Suggested Fixes: Change the loop condition: Modify the condition `<=` to `<` to ensure the loop only accesses elements within the valid range of `numbers`.

Figure 5.43: Hỗ trợ sửa lỗi

Khi người dùng cần gợi ý cho bài tập, họ có thể nhấn nút "Get Hints" để nhận những hướng dẫn hữu ích; nếu chưa rõ, AI Tutor còn có thể đưa ra ví dụ minh họa giúp họ hiểu rõ hơn.

AI Tutor

< get hints

1. use a loop to iterate over the string
2. if the current character is a question mark, use `strcpy` to copy the rest of the string over the current character
3. use `strlen` to get the length of the string
4. use `realloc` to resize the string

Figure 5.44: Cung cấp gợi ý cho bài tập

5.5.8 Progress Tracking

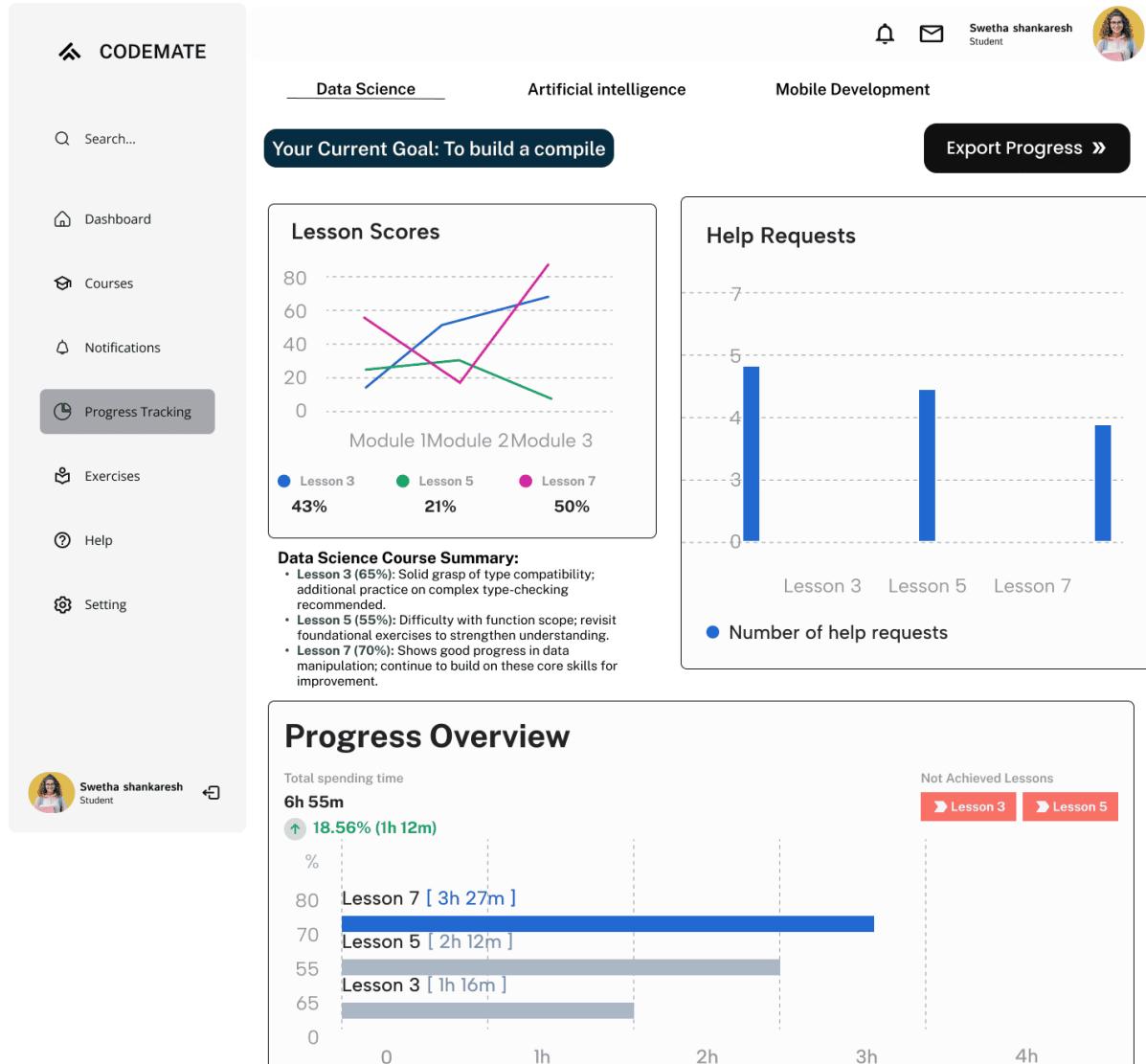


Figure 5.45: Progress Tracking Page

Trang Progress Tracking ghi lại tiến độ học tập của sinh viên theo từng môn học, bao gồm 3 biểu đồ chính:

1. Biểu Đồ Đánh Giá Điểm Số Theo Lesson (Biểu đồ đường)

- **X-Axis:** Tên module (1, 2, 3).
- **Y-Axis:** Điểm số (từ 0 đến 100%).
- **Màu của đường biểu đồ:** Lesson(3,5,7)
- **Đơn vị:** Phân trăm (%).
- **Ý Nghĩa:** Biểu đồ này cho phép sinh viên thấy tỷ lệ hoàn thành mục tiêu của từng



module theo lesson và dễ dàng nhận ra lesson nào đạt hoặc chưa đạt yêu cầu, bị khó khăn ở module nào.

2. Biểu Đồ Theo Dõi Số Lượng Vấn Đề hoặc Yêu Cầu Trợ Giúp (Biểu đồ cột)

- **X-Axis:** Tên lesson (1, 2, 3).
- **Y-Axis:** Số lần sinh viên yêu cầu trợ giúp (Đếm số lần yêu cầu trợ giúp tới AI).
- **Đơn vị:** Lần (Số lần hỏi).
- **Ý Nghĩa:** Biểu đồ này ghi nhận độ khó của từng bài tập, giúp hệ thống nhận biết các phần cần được cải thiện hoặc các bài tập khó để review lại cho giáo viên hoặc sinh viên cần chú trọng học lại.

3. Biểu Đồ Tổng Quan Tiến Độ Học (Progress Overview) (Biểu đồ cột)

- **X-Axis:** Thời gian (Số giờ học).
- **Y-Axis:** Tỷ lệ hoàn thành (%) của lesson.
- **Đơn vị:** Phần trăm (%) hoàn thành lesson.
- **Ý Nghĩa:** Cho thấy tiến độ tổng thể của sinh viên, giúp họ nhận ra tốc độ học của mình và đối chiếu với lộ trình học ban đầu.

Chapter 6

Hiện thực và kiểm thử

6.1 Hiện thực

Mô tả quá trình triển khai hệ thống học tập sử dụng LLMs (Large Language Models) để sinh ra các bài học đề xuất cho sinh viên dựa trên các bài học có sẵn từ giảng viên. Hệ thống được xây dựng trên nền tảng frontend sử dụng VueJS với Vuetify và Tailwind để phát triển giao diện người dùng, và backend được phát triển bằng FastAPI với kiến trúc DDD (Domain-Driven Design) và sử dụng PostgreSQL làm cơ sở dữ liệu.

6.1.1 Kiến trúc hệ thống

Hệ thống được chia thành hai phần chính: frontend và backend.

6.1.1.1 Frontend (VueJS với Vuetify và Tailwind)

Giao diện người dùng được xây dựng với VueJS để tạo ra các trang web động, hỗ trợ các tính năng như: dashboard, course list, course detail, feedback, và giao diện nhập mục tiêu học (goal input). Vuetify được sử dụng để phát triển các component UI, đảm bảo tính thẩm mỹ và khả năng sử dụng cao. Tailwind CSS được sử dụng để tùy chỉnh giao diện và tạo kiểu cho các thành phần UI một cách linh hoạt và dễ dàng. Mỗi trang và chức năng được thiết kế đơn giản và dễ sử dụng, bao gồm các mục như: lời chào, hoạt động gần đây, phản hồi và danh sách các khóa học mà sinh viên đang tham gia.

6.1.1.2 Backend (FastAPI với DDD architecture)

FastAPI được chọn làm framework backend nhờ vào khả năng xử lý nhanh và hỗ trợ RESTful API. Kiến trúc DDD (Domain-Driven Design) được áp dụng để chia hệ thống thành các domain riêng biệt như: Users, Courses, Lessons, Feedback, và Learning Outcomes. Điều này giúp mã nguồn dễ duy trì, mở rộng và dễ hiểu. PostgreSQL được sử dụng để lưu trữ dữ liệu về các khóa học, lessons, learning outcomes, feedback từ sinh viên, và các mục tiêu học của sinh viên. Mỗi

học phần (lesson) của khóa học được lưu trong cơ sở dữ liệu, cùng với các thông tin về mục tiêu học và phản hồi từ sinh viên.

6.1.2 Quá trình triển khai

6.1.2.1 Tạo cấu trúc cơ sở dữ liệu

Các bảng cơ sở dữ liệu được thiết kế để lưu trữ các thông tin về khóa học, lessons, các bài học (modules), feedback của sinh viên, và các mục tiêu học. Mỗi bảng liên quan đến các thực thể trong hệ thống được cấu hình phù hợp với yêu cầu của DDD. Các bảng liên quan đến sinh viên và giảng viên cũng được liên kết để theo dõi các hoạt động học tập của sinh viên.

6.1.2.2 Xây dựng các API

Các API cho phép sinh viên và giảng viên tương tác với hệ thống, bao gồm việc đăng nhập, xem danh sách khóa học, xem chi tiết khóa học, gửi feedback, và nhập mục tiêu học. Backend của hệ thống sử dụng FastAPI để xử lý các yêu cầu HTTP nhanh chóng, đảm bảo hiệu suất và khả năng mở rộng.

users	
POST	/v1/users/ Create
GET	/v1/users/ List
POST	/v1/users/upsert Upsert
DELETE	/v1/users/{id} Delete
GET	/v2/users/ List
POST	/v2/users/ Create
DELETE	/v2/users/{id} Delete
dashboard	
GET	/v1/dashboard/welcome/{studentId} Get Welcome Message
POST	/v1/dashboard/activities Get Activities
POST	/v1/dashboard/activities/ Add Activity
courses	
POST	/v1/courses/ Get Courses
GET	/v1/courses/{courseId}/students/{studentId}/ Get Course For Student
PUT	/v1/courses/{courseId}/students/{studentId}/lessons/{lessonId}/bookmark Bookmark Lesson
GET	/v1/courses/{courseId}/students/{studentId}/lessons_recommendation/ Get Lessons Recommendation
recommendation	
GET	/v1/recommend_lessons/{recommendLessonId} Recommend Lesson
GET	/v1/modules/{moduleId}/quizzes Get Module By Quiz
GET	/v1/modules/{moduleId}/quizzes/{quizId} Get Quiz Exercise
PUT	/v1/modules/{moduleId}/quizzes/{quizId}/submit Submit Quiz Answers
PUT	/v1/modules/{moduleId}/quizzes/{quizId}/clear Clear Quiz Answers
GET	/v1/modules/{moduleId}/documents Get Document

Figure 6.1: API endpoints



6.1.2.3 Quản lý bài học và đề xuất học tập

Giảng viên có thể đăng các khóa học cùng với learning outcomes và lessons. Hệ thống sẽ sử dụng LLMs để phân tích mục tiêu học của sinh viên và tự động đề xuất các lessons phù hợp, dựa trên nội dung và mục tiêu học của giảng viên đã đăng. Mỗi lesson sẽ bao gồm các module (quiz, code exercises, reading materials). LLMs sẽ được tích hợp vào backend để tự động tạo nội dung học cho sinh viên. Khi sinh viên nhập mục tiêu học, hệ thống sẽ phân tích mục tiêu và so sánh với các learning outcomes đã có của giảng viên để đề xuất bài học phù hợp.

6.1.2.4 Giao diện người dùng

VueJS và Vuetify tạo ra một giao diện người dùng dễ sử dụng và thân thiện. Trang dashboard của sinh viên hiển thị thông tin về các khóa học đang học, thông báo về các hoạt động gần đây, và phần gửi feedback cho hệ thống. Các sinh viên có thể nhập mục tiêu học vào một modal trong trang Course Detail để hệ thống đề xuất lessons và modules phù hợp với mục tiêu của họ. Hệ thống cũng ghi lại thời gian học của sinh viên để theo dõi tiến độ học tập và tạo ra các báo cáo cho lộ trình học tập tiếp theo.

6.2 Kiểm thử

6.2.1 Tổng quan về kiểm thử

Mục đích của kiểm thử trong hệ thống: Đảm bảo rằng các tính năng hoạt động đúng, hệ thống không có lỗi và có thể chịu được tải trong quá trình sử dụng thực tế.

6.2.1.1 Loại kiểm thử thực hiện:

- Kiểm thử chức năng:** Đảm bảo các chức năng của hệ thống (đăng nhập, nhập mục tiêu học, gửi feedback, xem course details, v.v.) hoạt động đúng.
- Kiểm thử hiệu suất:** Đảm bảo hệ thống có thể xử lý nhiều yêu cầu đồng thời và không gặp phải vấn đề về hiệu suất.
- Kiểm thử bảo mật:** Đảm bảo hệ thống bảo vệ tốt các dữ liệu người dùng và tránh các tấn công bảo mật (ví dụ: SQL Injection, Cross-site Scripting).
- Kiểm thử tích hợp:** Kiểm tra sự hoạt động đồng bộ của các module frontend và backend.
- Kiểm thử hồi quy:** Đảm bảo những thay đổi mới không làm hỏng các tính năng cũ của hệ thống.



6.2.2 Kiểm thử API

6.2.2.1 Kịch bản Kiểm thử

6.2.2.1.a 1. Kiểm thử API Chào mừng (Welcome API)

Mục đích: Kiểm thử khả năng lấy thông báo chào mừng cho sinh viên qua API GET /v1/dashboard/welcome/studentId.

Test Case 1: Lấy thông báo chào mừng với Student ID hợp lệ

- **Mô tả:** Gửi yêu cầu GET đến API /v1/dashboard/welcome/studentId với Student ID hợp lệ.
 - **Kết quả mong đợi:**
 - Mã phản hồi là 200.
 - Trường isSuccess có giá trị true.
 - Trường message là chuỗi ký tự.
 - Trường data chứa các thông tin về khóa học, course_id, và last_accessed.

Test Case 2: Lấy thông báo chào mừng với Student ID không hợp lệ

- **Mô tả:** Gửi yêu cầu GET đến API /v1/dashboard/welcome/studentId với Student ID không hợp lệ.
 - **Kết quả mong đợi:**
 - Mã phản hồi là 500.
 - Thông báo lỗi chỉ ra việc kiểm tra không thành công.

6.2.2.1.b 2. Kiểm thử API Lấy Hoạt động (Get Activities API)

Mục đích: Kiểm thử khả năng lấy danh sách hoạt động của sinh viên qua API POST /v1/dashboard/activities.

Test Case 1: Lấy danh sách hoạt động với phân trang mặc định

- **Mô tả:** Gửi yêu cầu POST đến API /v1/dashboard/activities với Student ID hợp lệ và tham số phân trang mặc định (limit: 0, offset: 0).
 - **Kết quả mong đợi:**



- Mã phản hồi là 200.
- Trường isSuccess có giá trị true.
- Trả về danh sách hoạt động với các trường như activity_id, activity_description, activity_type, activity_date.

Test Case 2: Lấy danh sách hoạt động với giới hạn tùy chỉnh

- **Mô tả:** Gửi yêu cầu POST với Student ID hợp lệ và tham số phân trang (limit: 5, offset: 0).
- **Kết quả mong đợi:**
 - Trả về tối đa 5 hoạt động.
 - Phân trang đúng.

Test Case 3: Lấy danh sách hoạt động với offset

- **Mô tả:** Gửi yêu cầu POST với Student ID hợp lệ và tham số phân trang (limit: 10, offset: 10).
- **Kết quả mong đợi:**
 - Trả về hoạt động từ bản ghi thứ 11.
 - Phân trang đúng.

Test Case 4: Lấy danh sách hoạt động với Student ID không hợp lệ

- **Mô tả:** Gửi yêu cầu POST với Student ID không hợp lệ.
- **Kết quả mong đợi:**
 - Mã phản hồi là 500.
 - Lỗi xác thực chi tiết.

6.2.2.1.c 3. Kiểm thử API Lấy Danh sách Khóa học (Courses List API)

Mục đích: Kiểm thử khả năng lấy danh sách khóa học cho sinh viên qua API POST /v1/courses/.

Test Case 1: Lấy danh sách khóa học với phân trang mặc định



- **Mô tả:** Gửi yêu cầu POST với Student ID hợp lệ và tham số phân trang mặc định (offset: 0, page_size: 10).

- **Kết quả mong đợi:**

- Mã phản hồi là 200.
- Trả về danh sách các khóa học với các thông tin chi tiết như id, name, start_date, end_date, student_list, learning_outcomes.

Test Case 2: Lấy danh sách khóa học với kích thước trang tùy chỉnh

- **Mô tả:** Gửi yêu cầu POST với Student ID hợp lệ và tham số phân trang (offset: 0, page_size: 20).

- **Kết quả mong đợi:**

- Trả về tối đa 20 khóa học.
- Phân trang đúng.

Test Case 3: Lấy danh sách khóa học với offset lớn

- **Mô tả:** Gửi yêu cầu POST với Student ID hợp lệ và tham số phân trang (offset: 100, page_size: 10).

- **Kết quả mong đợi:**

- Trả về phản hồi hợp lý.
- Hệ thống xử lý tốt với các offset lớn.

Test Case 4: Lấy danh sách khóa học với Student ID không hợp lệ

- **Mô tả:** Gửi yêu cầu POST với Student ID không hợp lệ.

- **Kết quả mong đợi:**

- Mã phản hồi là 500.
- Lỗi xác thực chi tiết.



6.2.2.1.d 4. Kiểm thử API Lấy Chi tiết Khóa học (Get Course for Student API)

Mục đích: Kiểm thử khả năng lấy chi tiết khóa học cho sinh viên qua API GET /v1/courses/courseId/students/studentId/.

Test Case 1: Lấy chi tiết khóa học với ID hợp lệ

- **Mô tả:** Gửi yêu cầu GET với courseId và studentId hợp lệ.
- **Kết quả mong đợi:**
 - Mã phản hồi là 200.
 - Trường isSuccess có giá trị true.
 - Trả về các thông tin chi tiết khóa học như course_id, student_id, completed_lessons, time_spent, assignments_done, và các bài học liên quan.

Test Case 2: Lấy chi tiết khóa học với Student ID không khớp với Course ID

- **Mô tả:** Gửi yêu cầu GET với courseId hợp lệ và studentId không khớp.
- **Kết quả mong đợi:** Hệ thống trả về lỗi thích hợp với thông báo lỗi nhất quán.

Test Case 3: Lấy chi tiết khóa học với ID không hợp lệ

- **Mô tả:** Gửi yêu cầu GET với courseId và studentId không hợp lệ.
- **Kết quả mong đợi:**
 - Mã phản hồi là 422.
 - Lỗi xác thực chi tiết.

6.2.2.1.e 5. Kiểm thử API Lấy Danh sách Khóa học Đề xuất (Recommended Courses List API)

Mục đích: Kiểm thử khả năng lấy danh sách khóa học đề xuất cho sinh viên qua API GET /courseId/students/studentId/lessons_recommendation/.

Test Case 1: Lấy danh sách khóa học với phân trang mặc định

- **Mô tả:** Gửi yêu cầu GET với student_id và course_id hợp lệ.
- **Kết quả mong đợi:**



- Mã phản hồi là 200.
- Trả về danh sách khóa học với các chi tiết như course_id, course_name, lesson_id, bookmark, status, title, description.

Test Case 2: Lấy danh sách khóa học với Course ID không hợp lệ

- **Mô tả:** Gửi yêu cầu GET với student_id hợp lệ và course_id không hợp lệ.
- **Kết quả mong đợi:**
 - Mã phản hồi là 500.
 - Lỗi xác thực chi tiết.

Test Case 3: Lấy danh sách khóa học với Student ID không hợp lệ

- **Mô tả:** Gửi yêu cầu GET với student_id không hợp lệ.
- **Kết quả mong đợi:**
 - Mã phản hồi là 500.
 - Lỗi xác thực chi tiết.

6.2.2.1.f 6. Kiểm thử API Thêm Hoạt động (Add Activity API)

Mục đích: Kiểm thử khả năng thêm hoạt động cho sinh viên qua API POST /activities/.

Test Case 1: Thêm hoạt động hợp lệ

- **Mô tả:** Gửi yêu cầu POST với dữ liệu hợp lệ (bao gồm student_id, type, description).
- **Kết quả mong đợi:**
 - Trạng thái trả về là 200 với thông điệp "Successfully added the activity."

Test Case 2: Thêm hoạt động thiếu student_id

- **Mô tả:** Gửi yêu cầu POST mà không có trường student_id.
- **Kết quả mong đợi:**
 - Mã phản hồi là 400.
 - Lỗi xác thực chi tiết.

6.2.2.2 Kết quả kiểm thử API

Chapter 7

Tổng kết

7.1 Kết quả đạt được

7.2 Hướng phát triển



Kế hoạch thực hiện Đồ án Tốt nghiệp

Tuần	Nhiệm Vụ	Kết Quả Mong Đợi	Phụ Trách
1	Authentication: Backend APIs và UI	Đăng nhập/đăng ký hoạt động	Phạm Thị
	Landing Page: Thiết kế UI cơ bản	Trang giới thiệu hệ thống	Tuấn Anh
	Quản lý khóa học (Giảng viên): API CRUD	API để thêm/sửa/xóa khóa học, bài học	Phương Nam
2-3	Quản lý học sinh (Giảng viên): API + UI	Giảng viên xem tiến độ và phản hồi học sinh	Phương Nam
	Progress Tracking: Backend APIs	Xử lý dữ liệu học tập sinh viên	Phạm Thị
	Quản lý người dùng (Admin): API CRUD	API quản lý tài khoản (giảng viên, sinh viên)	Tuấn Anh
4-5-6	Tích hợp LLM: Quiz Generation, Document Generation (Backend APIs)	API nhận input từ sinh viên, giảng viên và tạo câu hỏi trắc nghiệm tự động	Phạm Thị
	Tích hợp LLM: Code Hint Module, Debugging Assistant, Explain Code Module	API cung cấp gợi ý sửa lỗi và giải thích code khi sinh viên yêu cầu	Tuấn Anh
	Xử lý câu trả lời: LLM Evaluation Module	API chấm điểm câu trả lời tự động và đưa ra nhận xét	Phương Nam
7-8	Progress Tracking: UI + Biểu đồ	Giao diện hiển thị tiến độ học tập của sinh viên	Phạm Thị
	Gửi Feedback (Sinh viên): Backend APIs	Gửi phản hồi về bài học/hệ thống	Phương Nam
	Hiển thị Recent Activities: Backend APIs	Xử lý và lưu trữ dữ liệu hoạt động gần đây của sinh viên	Tuấn Anh
9-10	Gửi Feedback: UI tích hợp	Giao diện cho sinh viên gửi feedback	Phương Nam
	Hiển thị Recent Activities: UI tích hợp	Giao diện hiển thị các hoạt động gần đây của sinh viên	Tuấn Anh
	Quản lý hệ thống (Admin): Logs & Lỗi	API và UI để admin theo dõi logs và lỗi	Tuấn Anh
11-12	Xử lý Feedback (Admin): Backend APIs	API xử lý phản hồi từ sinh viên	Phương Nam
	Quản lý bài tập (Giảng viên): API CRUD	API quản lý bài tập trong khóa học	Phạm Thị
	Testing Feedback Module	Kiểm tra và debug toàn bộ module Feedback	Phương Nam
13	Testing & Debugging: Toàn bộ hệ thống	Kiểm tra, fix lỗi phát sinh trên hệ thống	Cả nhóm
14	Triển khai Backend: Deploy APIs	Backend chạy trên server test	Phạm Thị
	Triển khai Frontend: Deploy giao diện	Frontend hoạt động ổn định	Tuấn Anh
15	Báo cáo & Demo: Chuẩn bị thuyết trình	Báo cáo đầy đủ, demo hoàn thiện	Cả nhóm

Table 7.1: Kế hoạch 15 Tuần

Bibliography

- [1] Brown, T. B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., Henighan, T., Child, R., Ramesh, A., Ziegler, D. M., Wu, J., Winter, C., ... Amodei, D. (2020). Language models are few-shot learners. <https://arxiv.org/abs/2005.14165>
- [2] IBM. (n.d.). What is a knowledge graph? [Accessed: 2024-09-22]. <https://www.ibm.com/topics/knowledge-graph>
- [3] Piech, C., Bassen, J., Huang, J., Ganguli, S., Sahami, M., Guibas, L., & Sohl-Dickstein, J. (2015). Deep knowledge tracing. *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 1*, 505–513.
- [4] Abdelrahman, G., Wang, Q., & Nunes, B. (2023). Knowledge tracing: A survey. *ACM Comput. Surv.*, 55(11). <https://doi.org/10.1145/3569576>
- [5] Sahoo, P., Singh, A. K., Saha, S., Jain, V., Mondal, S., & Chadha, A. (2024). A systematic survey of prompt engineering in large language models: Techniques and applications.
- [6] Wei, J., Wang, X., Schuurmans, D., Bosma, M., Chi, E. H., Le, Q., & Zhou, D. (2022). Chain of thought prompting elicits reasoning in large language models. *CoRR*, *abs/2201.11903*. <https://arxiv.org/abs/2201.11903>
- [7] Gao, Y., Xiong, Y., Gao, X., Jia, K., Pan, J., Bi, Y., Dai, Y., Sun, J., Wang, M., & Wang, H. (2024). Retrieval-augmented generation for large language models: A survey. <https://arxiv.org/abs/2312.10997>
- [8] Lin, D. (2024). Revolutionizing retrieval-augmented generation with enhanced pdf structure recognition. <https://arxiv.org/abs/2401.12599>
- [9] Bulathwela, S., Perez-Ortiz, M., Ranawaka, E. S. V., Siriwardana, R. I. P. B., Ganepola, G. A. K. Y., Ravikkumar, T., Pussegoda, S., Novak, E., Yilmaz, E., & Shawe-Taylor, J. (2021). Cross modal, cross cultural, cross lingual, cross domain, and cross site global oer network.
- [10] Li, H., Yu, J., Ouyang, Y., Liu, Z., Rong, W., Li, J., & Xiong, Z. (2024). Explainable few-shot knowledge tracing. *arXiv preprint arXiv:2405.14391*.



- [11] Raj, N. S., & Renumol, V. G. (2022). An improved adaptive learning path recommendation model driven by real-time learning analytics. *Journal of Computers in Education*, 11, 121–148. <https://doi.org/10.1007/s40692-022-00250-y>
- [12] Hu, S., & Wang, X. (2024). Foke: A personalized and explainable education framework integrating foundation models, knowledge graphs, and prompt engineering. <https://arxiv.org/abs/2405.03734>