

TRƯỜNG ĐẠI HỌC TRÀ VINH  
KHOA KỸ THUẬT VÀ CÔNG NGHỆ  
BỘ MÔN CÔNG NGHỆ THÔNG TIN



## BÁO CÁO

# THỰC TẬP TỐT NGHIỆP

Cán bộ hướng dẫn tại nơi thực tập:

**Họ và tên: Nhan Minh Phúc**

Giáo viên hướng dẫn:

**Họ và tên: Nguyễn Hoàng Duy Thiện**

Sinh viên báo cáo:

**Họ và tên: Nguyễn Đức Mạnh**

Mã số SV: 110121202

Lớp: DA21TTB

Trà Vinh, tháng 4 năm 2025

## LỜI CẢM ƠN

Tôi xin gửi lời tri ân sâu sắc đến Ban Giám đốc “Trung tâm Ngoại ngữ - Tin học Victory”, đặc biệt là thầy Minh Phúc, người đã trao cho tôi một cơ hội thực tập vô cùng quý giá trong suốt 6 tuần vừa qua. Với một sinh viên chuyên ngành công nghệ phần mềm như tôi, việc được thầy tin tưởng giao cho nhiệm vụ ứng dụng trí tuệ nhân tạo vào hệ thống quả thực là một thử thách không nhỏ, thậm chí có chút “bối rối” ban đầu. Áp lực từ những điều mới mẻ xung quanh đôi lúc khiến tôi cảm thấy chùn bước trong những tuần đầu tiên. Tôi đã từng nghĩ đến việc bỏ cuộc, nhưng chính sự tin tưởng, bổ xung kiến thức mới hằng tuần và việc thầy hoàn toàn không gây áp lực đã giúp tôi vượt qua giai đoạn khó khăn đó. Tôi thực sự trân trọng điều này, vì nó cho tôi thấy thầy đặt niềm tin vào khả năng của tôi, và đó là động lực lớn nhất để tôi cố gắng hoàn thành nhiệm vụ được giao.

Tuy nhiên, sau khi hoàn thành sản phẩm, tôi nhận thấy mình vẫn còn nhiều thiếu sót. Tôi xin gửi lời xin lỗi chân thành đến thầy vì có lẽ sản phẩm cuối cùng chưa thực sự ấn tượng và đột phá hơn kỳ vọng của thầy. Trong quá trình thực tập, tôi đã dành khá nhiều thời gian cho việc nghiên cứu lý thuyết và đôi khi có những bước đi chưa đúng hướng, dẫn đến việc lãng phí thời gian. Đây là một bài học kinh nghiệm quý báu mà tôi sẽ ghi nhớ. Một lần nữa, tôi xin chân thành cảm ơn thầy và xin lỗi vì những thiếu sót của mình.

Bên cạnh đó, tôi vô cùng biết ơn sự tận tình và chu đáo của thầy Duy Thiện. Thầy đã luôn cập nhật những thông tin mới nhất, kịp thời nhất và chu đáo chuẩn bị mọi giấy tờ, thủ tục cần thiết để tôi có thể thực tập tại Victory một cách thuận lợi. Tôi xin cảm ơn thầy vì đã luôn theo sát và phản hồi các báo cáo tuần của tôi một cách nhanh chóng trong suốt 6 tuần qua. Sự quan tâm và hỗ trợ kịp thời của thầy đã giúp tôi cảm thấy an tâm và có động lực để tiếp tục công việc tại trung tâm.

Cuối cùng, tôi xin chân thành cảm ơn thầy Minh Phúc và thầy Duy Thiện đã tạo điều kiện và hỗ trợ tôi hoàn thành bài báo cáo thực tập này. Những kinh nghiệm và kiến thức tôi thu nhận được trong suốt quá trình thực tập sẽ là nền tảng vững chắc cho sự phát triển của tôi trong tương lai.

**Nguyễn Đức Mạnh**

## PHIẾU ĐÁNH GIÁ BÁO CÁO KẾT QUẢ THỰC TẬP

(Dùng cho giáo viên chấm báo cáo thực tập)

Họ tên sinh viên thực tập: Nguyễn Đức Mạnh

Mã số SV: 110121202

**A. Điểm của cán bộ hướng dẫn thực tập tại cơ quan (CBHDTT):.....**

Điểm chấm báo cáo thực tập thang điểm tối đa là 10 dựa vào phiếu chấm ở **mục B**

**B. Phiếu chấm điểm báo cáo thực tập tốt nghiệp:**

Nội dung đánh giá	Điểm tối đa	Điểm thực
<b>1. Hình thức trình bày</b>	<b>1.5</b>	
1.1 Đúng định dạng của Khoa (trang bìa, trang lời cảm ơn, trang đánh giá thực tập của Khoa, trang mục lục và các nội dung báo cáo).	0.5	
1.2 Sử dụng đúng phông chữ theo quy định (Times New Roman, size 13).	0.5	
1.3 Trình bày mạch lạc, súc tích, không có lỗi chính tả.	0.5	
<b>2. Lịch làm việc</b>	<b>1.0</b>	
2.1 Có lịch làm việc đầy đủ cho các tuần.	0.5	
2.2 Hoàn thành tốt kế hoạch công tác ghi trong lịch làm việc (thông qua nhận xét của cán bộ hướng dẫn).	0.5	
<b>3. Nội dung thực tập</b>	<b>7.5</b>	
3.1 Có được sự hiểu biết tốt về cơ quan nơi thực tập	1.0	
3.2 Thể hiện tốt các nhiệm vụ, công việc được giao thông qua báo cáo	1.0	
3.3 Chất lượng bài báo cáo (số liệu, hình vẽ, hình chụp, biểu đồ, ...)	2.5	
3.4 Thể hiện được kiến thức và kỹ năng học được sau quá trình thực tập	2.0	
3.5 Những đóng góp cho cơ quan nơi thực tập	1.0	
<b>TỔNG CỘNG</b>	<b>10.0</b>	

**C. Điểm tổng hợp = (GVCB+CBHD)/2: .....**

Trà Vinh, ngày.....tháng.....năm 2024

**GIÁO VIÊN CHẤM BÁO CÁO**

(ký, ghi rõ họ tên)

## MỤC LỤC

NHẬT KÝ THỰC TẬP TỐT NGHIỆP .....	1
Phần 1 GIỚI THIỆU VỀ CƠ QUAN NƠI THỰC TẬP TỐT NGHIỆP .....	2
1.1. Thông tin chung.....	2
1.2. Cơ cấu tổ chức .....	3
Phần 2 NỘI DUNG THỰC TẬP .....	4
2.1. Yêu cầu công việc.....	4
2.2. Nghiên cứu.....	4
2.2.1 Bài báo “Test-Driven Development and LLM-based Code Generation” .....	4
2.2.2 LLMs model .....	5
2.2.3 Phần mềm LM Studio hỗ trợ chạy Mô hình LLM trên máy tính .....	6
2.2.4 Python với việc biên dịch và thực thi mã nguồn C++ .....	7
2.4. Thực nghiệm.....	8
2.4.1 Chạy Mô hình LLM cục bộ trên phần mềm LM Studio.....	8
2.4.2 Kiểm tra tương thích của “Khung đánh giá TGen” với mô hình LLM .....	9
2.4.3 Kiểm tra hiệu quả việc biên dịch – thực thi – testcase của mã nguồn C++ bằng ngôn ngữ Python .....	12
2.5. Thiết kế hệ thống .....	14
2.5.1 Lựa chọn công nghệ.....	14
2.5.2 Quy trình xử lý chung của hệ thống .....	15
2.5.3 Giao diện người dùng .....	18
2.5.4 Chuẩn hóa cấu trúc tập tin thông tin bài toán .....	18
2.6. Sản phẩm đạt được.....	20
2.6.1 Hoạt động tổng quát của hệ thống .....	20
2.6.2 Giao diện người dùng .....	21
2.7. Những điểm cần cải tiến .....	25
2.7.1 Mô hình LLM với phần mềm LM Studio .....	25
2.7.2 Server Flask python .....	26
Phần 3 KẾT LUẬN .....	28

3.1. Tự nhận xét, đánh giá .....	28
3.2. Kết luận, kiến nghị.....	29

## **DANH MỤC HÌNH ẢNH – BẢNG BIỂU**

Hình 1.1 Cơ cấu tổ chức của Trung tâm Ngoại ngữ - Tin học Victory .....	3
Hình 2.1 Giao diện thiết lập máy chủ API của phần mềm LM Studio .....	8
Hình 2.2 Khung đánh giá TGen trong bài báo khoa học “Test-Driven Development and LLM-based Code Generation” .....	15
Hình 2.3 Quy trình xử lý chung của hệ thống .....	17
Hình 2.4 Cấu trúc trình bày bài toán trong tập tin “.json” .....	19
Hình 2.5 Cấu trúc trình bày bài toán trong tập tin “.docx” .....	20
Hình 2.6 Cấu trúc trình bày bài toán trong tập tin “.txt” .....	20
Hình 2.7 Giao diện tab mô tả bài toán .....	21
Hình 2.8 Giao diện tab kết quả tạo mã nguồn .....	23
Hình 2.9 Giao diện tab thống kê .....	24

## NHẬT KÝ THỰC TẬP TỐT NGHIỆP

TT	NGÀY THÁNG	NỘI DUNG, CÔNG VIỆC THỰC HIỆN	GHI CHÚ
Tuần 1	Từ 17/02/2025 Đến 23/02/2025	<ul style="list-style-type: none"> <li>- Tiếp nhận yêu cầu công việc xây dựng <b>hệ thống tạo mã nguồn tự động với các mô hình ngôn ngữ lớn dựa trên bài báo “Test-Driven Development and LLM-based Code Generation”</b> [1] tại đơn vị thực tập.</li> <li>- Tìm hiểu tổng quang về yêu cầu và tìm kiếm các tài liệu liên quan.</li> <li>- Xác định các công nghệ cần có để thực hiện công việc và tìm hiểu các công nghệ đã xác định.</li> <li>- Thiết kế kiến trúc hệ thống.</li> </ul>	
Tuần 2	Từ 24/02/2025 Đến 02/3/2025	<ul style="list-style-type: none"> <li>- Nghiên cứu lựa chọn các Mô hình LLM phù hợp cho từng phần trong hệ thống đã thiết kế.</li> <li>- Thử nghiệm chạy local đối với các Mô hình LLM đã chọn.</li> <li>- Nghiên cứu tạo – cấu hình – sử dụng Mô hình LLM ở dạng máy chủ ảo (cung cấp API).</li> <li>- Nghiên cứu mã nguồn Python tương tác với Mô hình LLM API.</li> <li>- Tìm hiểu và phân tích cách mà Mô hình LLM phản hồi với từng yêu cầu để xử lý các ngoại lệ.</li> </ul>	
Tuần 3	Từ 03/3/2025 Đến 09/3/2025	<ul style="list-style-type: none"> <li>- Cấu hình Mô hình LLM để ràng buộc cách hoạt động và trả lời để tối ưu nhất có thể.</li> <li>- Hoàn thiện hệ thống API theo kiến trúc đã thiết kế.</li> <li>- Tiếp nhận yêu cầu nghiên cứu thêm về Mô hình LLM để cải thiện tốc độ và tính cá nhân hoá hơn.</li> <li>- Tìm hiểu sâu hơn về việc can thiệp vào cách hoạt động của các Mô hình LLM.</li> </ul>	
Tuần 4	Từ 10/3/2025 Đến 16/3/2025	<ul style="list-style-type: none"> <li>- Nghiên cứu chạy Mô hình LLM trực tiếp trên local mà không thông qua phần mềm.</li> <li>- Thử lại với nhiều Mô hình LLM khác nhau.</li> <li>- Thực hiện can thiệp để đọc các thông số tiền xử lý – hậu xử lý dữ liệu của Mô hình LLM.</li> <li>- Đánh giá tính ổn định và tính khả thi của việc pretraining, RAG, fineturning cho Mô hình LLM local khi việc cho yêu cầu tạo mã nguồn tự động.</li> </ul>	
Tuần 5	Từ 17/3/2025 Đến 23/3/2025	<ul style="list-style-type: none"> <li>- Viết giao diện người dùng.</li> <li>- Viết lại API theo Flask framework.</li> <li>- Test thử hệ thống với nhiều bài toán C++.</li> </ul>	
Tuần 6	Từ 24/3/2025 Đến 30/3/2025	<ul style="list-style-type: none"> <li>- Viết tài liệu cho hệ thống và đẩy lên Github.</li> <li>- Viết báo cáo thực tập.</li> </ul>	

## Phần 1

### GIỚI THIỆU VỀ CƠ QUAN NƠI THỰC TẬP TỐT NGHIỆP

#### 1.1. Thông tin chung

Trung tâm Ngoại ngữ - Tin học Victory, trực thuộc Trường Đại học Trà Vinh, là địa chỉ đào tạo uy tín tại tỉnh Trà Vinh, chuyên cung cấp các khóa học ngoại ngữ và tin học chất lượng cao. Trung tâm được ủy quyền bởi Hội đồng Khảo thí tiếng Anh Cambridge với mã số quốc tế VN370, là đơn vị khảo thí tiếng Anh Cambridge duy nhất tại Trà Vinh.

##### Địa chỉ liên hệ:

- **Địa chỉ:** Số 105 Kiên Thị Nhẫn, Phường 7, Thành phố Trà Vinh, tỉnh Trà Vinh.
- **Số điện thoại:** (02943) 862214 hoặc (02946) 274244.
- **Email:** [vifl@tvu.edu.vn](mailto:vifl@tvu.edu.vn).
- **Website:** [victory.tvu.edu.vn](http://victory.tvu.edu.vn).
- **Facebook:** [Ngoại ngữ – Tin học Victory Trà Vinh](#).

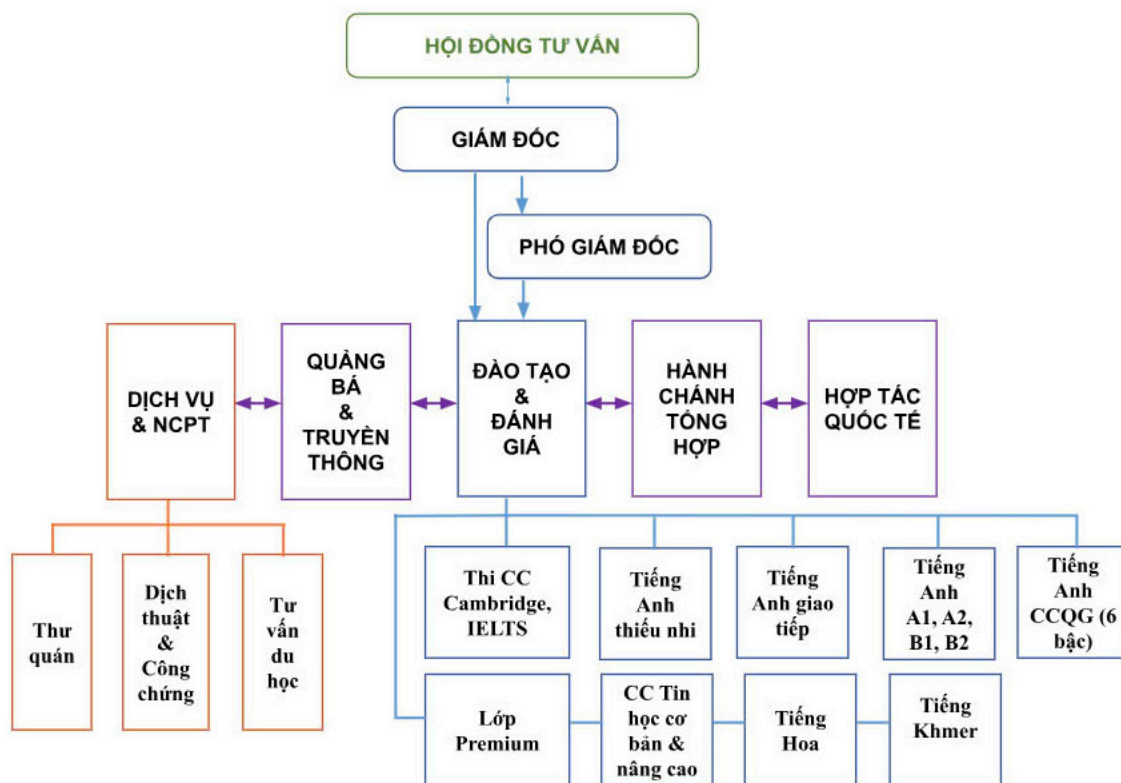
##### Các chương trình đào tạo tại trung tâm bao gồm:

- **Tiếng Anh thiếu nhi và thiếu niên:** Dành cho các em từ 6 đến 15 tuổi, giúp phát triển toàn diện kỹ năng nghe, nói, đọc, viết.
- **Tiếng Anh giao tiếp cho người lớn:** Tập trung nâng cao khả năng giao tiếp thực tế trong công việc và cuộc sống hàng ngày.
- **Luyện thi chứng chỉ quốc tế:** Chuẩn bị cho các kỳ thi như IELTS, TOEIC với đội ngũ giảng viên giàu kinh nghiệm.
- **Đào tạo tin học:** Từ cơ bản đến nâng cao, bao gồm các khóa học về ứng dụng công nghệ thông tin và phần mềm văn phòng.

Trung tâm cũng thường xuyên khai giảng các lớp Tiếng Hoa, Tiếng Nhật, Tiếng Khmer và các lớp Tin học, đáp ứng nhu cầu học tập đa dạng của học viên. Với đội ngũ giảng viên có trình độ cao và phương pháp giảng dạy tiên tiến, Trung tâm Ngoại ngữ - Tin học Victory cam kết mang đến môi trường học tập hiệu quả và chất lượng cho học viên tại Trà Vinh.



## 1.2. Cơ cấu tổ chức



Hình 1.1 Cơ cấu tổ chức của Trung tâm Ngoại ngữ - Tin học Victory

Bộ phận dịch vụ & Nghiên cứu phát triển (NCPT): Bao gồm các mảng Thư quán, Dịch thuật & Công chứng, và Tư vấn du học.

Bộ phận quảng bá & Truyền thông: Phụ trách quảng bá hình ảnh và truyền thông của trung tâm.

Bộ phận đào tạo & Đánh giá: Quản lý các chương trình đào tạo và thi chứng chỉ như Cambridge, IELTS, lớp Premium, tiếng Anh thiếu nhi, tiếng Anh giao tiếp, tiếng Hoa, tiếng Khmer, tin học cơ bản & nâng cao, và các chứng chỉ tiếng Anh (A1, A2, B1, B2, CCQG 6 bậc).

Bộ phận hành chính tổng hợp: Hỗ trợ công tác hành chính, nhân sự và tài chính của trung tâm.

Bộ phận hợp tác quốc tế: Phát triển quan hệ hợp tác với các tổ chức và trung tâm đào tạo quốc tế.

## Phần 2

### NỘI DUNG THỰC TẬP

#### 2.1. Yêu cầu công việc

Nghiên cứu và phát triển tại đơn vị thực tập, tôi được giao nhiệm vụ xây dựng một hệ thống tự động tạo mã nguồn từ mô tả bài toán bằng ngôn ngữ tự nhiên. Hệ thống này tận dụng sức mạnh của các mô hình ngôn ngữ lớn (LLMs) để chuyển đổi những yêu cầu từ con người thành mã nguồn hoàn chỉnh, giúp lập trình viên rút ngắn thời gian phát triển phần mềm và nâng cao hiệu suất làm việc.

Dự án được triển khai dựa trên các nguyên tắc và phương pháp tiên tiến được trình bày trong bài báo khoa học “Test-Driven Development and LLM-based Code Generation”. Điểm cốt lõi của hệ thống là áp dụng triết lý phát triển theo hướng kiểm thử (Test-Driven Development - TDD), đảm bảo rằng mã nguồn được tạo ra không chỉ chính xác mà còn có độ tin cậy cao. Mỗi đoạn mã được sinh ra sẽ được kiểm chứng thông qua các bài kiểm thử tự động, giúp phát hiện và loại bỏ lỗi ngay từ những giai đoạn đầu của quá trình phát triển.

Hệ thống này hứa hẹn sẽ mang lại nhiều lợi ích quan trọng trong lĩnh vực phát triển phần mềm, từ việc giảm thiểu sai sót trong lập trình đến việc tăng cường khả năng tự động hóa trong quy trình sản xuất phần mềm.

#### 2.2. Nghiên cứu

##### 2.2.1 Bài báo “Test-Driven Development and LLM-based Code Generation”

Bài báo khoa học “Test-Driven Development and LLM-based Code Generation” khám phá một hướng đi đầy hứa hẹn: tích hợp phương pháp Phát triển Hướng theo Kiểm thử (TDD) vào quy trình tạo mã nguồn bằng Mô hình Ngôn ngữ Lớn (LLM). Nghiên cứu trên tập trung vào việc chứng minh rằng: khi cung cấp các bộ kiểm thử chi tiết cùng với yêu cầu bài toán thì hiệu suất tạo mã nguồn của các LLM tiên tiến như GPT-4 và Llama 3 có thể được cải thiện đáng kể.

##### Điểm nổi bật của nghiên cứu:

- **Thách thức hiện tại:** Mặc dù LLM đã chứng tỏ khả năng tạo mã nguồn ấn tượng, nhưng kết quả thường tiềm ẩn các lỗi logic hoặc chưa hoàn toàn

đáp ứng các tiêu chí cụ thể, từ đó đặt ra nhu cầu cấp thiết về các cơ chế kiểm chứng hiệu quả.

- **Giải pháp TDD (Test Driven Development):** Nghiên cứu này đề xuất áp dụng TDD, một quy trình phát triển phần mềm mang tính lặp lại, trong đó việc viết các bài kiểm tra được ưu tiên thực hiện trước khi xây dựng logic chức năng. Cách tiếp cận này đảm bảo mã nguồn được phát triển bám sát các yêu cầu đã được xác định.
- **Khung đánh giá TGen:** Các tác giả đã giới thiệu TGen, một khung đánh giá được xây dựng dựa trên các nguyên tắc cốt lõi của TDD, nhằm mục đích lượng hóa và nâng cao hiệu quả của việc kết hợp TDD với khả năng tạo mã của LLM. TGen vận hành theo cơ chế tự cải tiến liên tục: các yêu cầu và chức năng mong muốn được cụ thể hóa thông qua các bài kiểm tra, sau đó LLM sẽ tạo mã. Một hệ thống phản hồi thông minh dựa trên LLM sẽ tiếp tục điều chỉnh và hoàn thiện mã cho đến khi nó vượt qua tất cả các bài kiểm tra đã được đặt ra.

### **Kết luận:**

Nghiên trên đã khẳng định việc tích hợp phương pháp Phát triển Hướng theo Kiểm thử (TDD) là một hướng đi đầy triển vọng để khai thác tối đa tiềm năng tạo mã của các Mô hình Ngôn ngữ Lớn (LLM). Bằng chứng từ nghiên cứu trên cho thấy việc cung cấp các bộ kiểm thử trong suốt quá trình tạo mã, kết hợp với cơ chế tự sửa lỗi lặp đi lặp lại, mang lại sự cải thiện đáng kể về độ chính xác và độ tin cậy của mã nguồn được tạo ra. Các tác giả mạnh mẽ khuyến nghị việc áp dụng rộng rãi các phương pháp tiếp cận dựa trên kiểm thử như một giải pháp then chốt để tối ưu hóa lợi ích của LLM trong bối cảnh phát triển phần mềm hiện đại.

### **2.2.2 LLMs model**

Trong quá trình ứng dụng trí tuệ nhân tạo (AI) vào công việc, việc điều chỉnh mô hình là một bước quan trọng để đảm bảo sự phù hợp với yêu cầu nghiệp vụ, bối cảnh tổ chức và các yếu tố bảo mật. Các phương pháp điều chỉnh mô hình phổ biến bao gồm Fine-tuning, Retrieval-Augmented Generation (RAG), Parameter-Efficient Fine-Tuning (PEFT) và Knowledge Distillation. Tuy nhiên, việc tạo mã nguồn (mã nguồn) thường là một ngoại lệ, nơi mà việc tinh chỉnh mô hình có thể không mang lại hiệu quả tối ưu.

Điều này xuất phát từ đặc thù của công việc tạo mã nguồn, vốn tập trung vào việc giải quyết một bài toán cụ thể được mô tả bằng ngôn ngữ tự nhiên, thay vì tuân theo các ràng buộc hoặc phong cách đặc thù như trong các tác vụ khác. Do đó, việc tận dụng các mô hình ngôn ngữ lớn (LLM) đã được huấn luyện sẵn thường là lựa chọn hiệu quả nhất để chuyển đổi mô tả bằng ngôn ngữ tự nhiên thành mã nguồn.

Sự đa dạng của các mô hình ngôn ngữ lớn (LLM) hiện nay có thể tạo ra một ấn tượng ban đầu về sự dễ dàng trong việc lựa chọn. Tuy nhiên, khi phải đối diện với vô số tùy chọn, việc xác định được mô hình phù hợp nhất với nhu cầu cụ thể lại trở thành một bài toán không đơn giản.

Với yêu cầu về một LLM có khả năng hiểu sâu sắc ngôn ngữ tự nhiên và tạo mã nguồn chất lượng cao, Qwen (Alibaba Cloud) nổi lên như một ứng cử viên đáng chú ý. Mô hình này không chỉ thể hiện khả năng xử lý ngôn ngữ một cách thông minh mà còn chứng minh hiệu quả vượt trội trong việc tạo ra mã nguồn. Được tối ưu hóa cho nhiều tình huống thực tế, Qwen có tiềm năng ứng dụng rộng rãi, từ việc trở thành một trợ lý AI thông minh đến việc hỗ trợ đắc lực cho các nhà phát triển phần mềm.

### **2.2.3 Phần mềm LM Studio hỗ trợ chạy Mô hình LLM trên máy tính**

LM Studio là một ứng dụng máy tính mạnh mẽ được thiết kế để chạy các Mô hình Ngôn ngữ Lớn (LLM) trực tiếp trên máy tính cục bộ. Về cốt lõi, LM Studio đóng vai trò là một nền tảng toàn diện để tải xuống, chạy và thử nghiệm nhiều loại LLM mã nguồn mở, chẳng hạn như Llama 3.1, Phi-3 và Gemma 2. Công cụ thân thiện với người dùng khi cho phép họ khai thác sức mạnh của các mô hình ngôn ngữ nâng cao mà không cần dựa vào các API bên ngoài hoặc kết nối internet.

LM Studio có 3 chế độ hoạt động “User” chỉ hiển thị giao diện trò chuyện như trên các chat AI phổ biến trên internet, “Power User”: cho phép truy cập vào các thông số tải và suy luận, định cấu hình cũng như các tính năng trò chuyện nâng cao như chèn, chỉnh sửa, “Developer”: Chế độ nâng cao cho phép truy cập toàn diện vào LM Studio.

Tại chế độ “Power User”, LM Studio cho phép chạy một hoặc nhiều Mô hình LLM cùng lúc ở chế độ máy API khi cho phép thiết lập CORS, thiết lập port. Và một số API tương tác kèm theo là cấu trúc body để dễ dàng tích hợp với các dịch vụ khác.

#### 2.2.4 Python với việc biên dịch và thực thi mã nguồn C++

Nhờ vào sự hỗ trợ của các thư viện linh hoạt và mạnh mẽ như “Subprocess”, “Pytest” và “Unittest”, cùng với chính sức mạnh và tính linh hoạt vốn có của ngôn ngữ Python. Python đã trở thành một lựa chọn phổ biến và mạnh mẽ trong lĩnh vực kiểm thử tự động, đặc biệt là khi nói đến việc đảm bảo chất lượng cho các dự án phát triển bằng C++.

Thư viện “Subprocess” đóng vai trò then chốt trong việc tương tác với hệ điều hành, cho phép Python thực hiện các lệnh bên ngoài như trình biên dịch C++ (g++, clang++), trình liên kết và thậm chí cả việc chạy trực tiếp các chương trình C++ đã được biên dịch. Điều này tạo ra một cầu nối liền mạch giữa môi trường kiểm thử Python và quá trình xây dựng và thực thi ứng dụng C++.

## 2.4. Thực nghiệm

### 2.4.1 Chạy Mô hình LLM cục bộ trên phần mềm LM Studio

Cài đặt LM Studio theo hướng dẫn từ trang chủ <https://lmstudio.ai>.

Thiết lập LM Studio trở thành máy chủ API cục bộ và chạy Mô hình LLM.

B1: Chọn chế độ: Ở góc dưới bên trái của phần mềm, ta cần chọn vào chế độ “Power User” hoặc “Developer”.

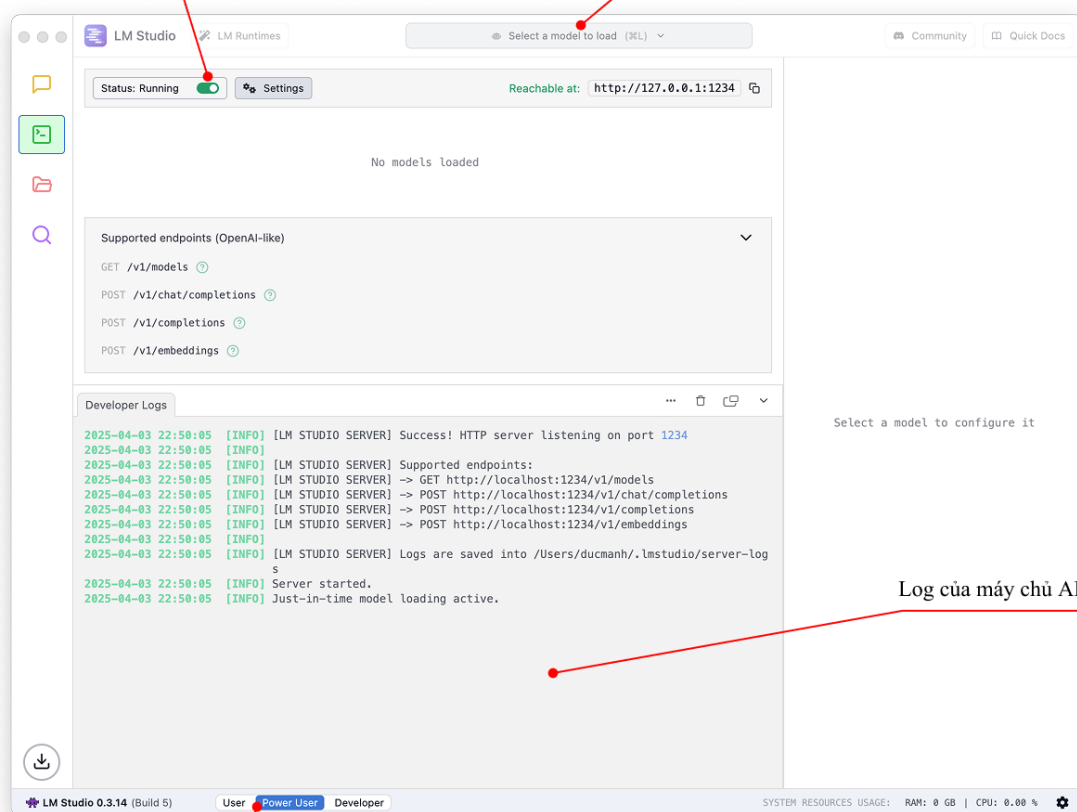
B2: Tìm Mô hình LLM: Tại thanh bên trái, tìm biểu tượng kính lúp. Một modal hiện lên và cho phép tìm kiếm, tìm model với từ khóa “Qwen2.5-Mã nguồn-7b-Instruct-GGUF”.

B3: Tải model: Trong các model đang khớp hãy tìm model thuộc “lmstudio-community” và tải về (Có thể chọn phiên bản phù hợp, nhưng mặc định LM Studio sẽ tải phiên bản phù hợp với cấu hình của máy).

B4: Thiết lập máy chủ API: Tại thanh bên trái, tìm biểu tượng terminal. Lần lượt thực hiện bật máy chủ API cục bộ, chọn Mô hình LLM vừa tải ở bước B3.

Tắt/bật máy chủ API cục bộ

Lựa chọn LLM model cần sử dụng



Log của máy chủ API cục bộ

Chọn chế độ hoạt động

Hình 2.1 Giao diện thiết lập máy chủ API của phần mềm LM Studio

## 2.4.2 Kiểm tra tương thích của “Khung đánh giá TGen” với mô hình LLM

Dựa trên “Khung đánh giá TGen” được trình bày trong bài báo “Test-Driven Development and LLM-based Code Generation”, một trong những yêu cầu tiên quyết cốt lõi đối với các mô hình ngôn ngữ lớn (LLM) có thể được tóm gọn như sau: Trong quá trình xử lý các yêu cầu hiện tại, mô hình LLM cần duy trì và tận dụng thông tin từ các tương tác trước đó để nâng cao chất lượng và độ chính xác của kết quả ở yêu cầu đang được xử lý.

Tuy nhiên, theo cơ chế hoạt động phổ biến của hầu hết các mô hình LLM hiện nay – vốn hoạt động độc lập giữa các phiên – khả năng ghi nhớ thông tin và ngữ cảnh thường bị giới hạn trong phạm vi một phiên tương tác duy nhất. Để kiểm chứng rõ ràng hơn khả năng này và tìm ra giải pháp cho việc ứng dụng khung đánh giá TGen cho mô hình LLM thì hai thử nghiệm đơn giản được thiết kế để kiểm tra lần lượt như sau:

- Thử nghiệm 1: Kiểm tra xem khả năng lưu giữ thông tin xuyên phiên của mô hình LLM. Gửi hai yêu cầu với hai phiên riêng biệt lần lượt, điều kiện tiên quyết là phiên sau nội dung yêu cầu phải có tính kế thừa thông tin ở phiên trước đó nhưng không được chứa dữ liệu phản hồi của yêu cầu trước đó.
- Thử nghiệm 2: Kiểm tra xem mô hình LLM có được thiết kế để hiểu được dữ liệu của các phiên trước đó khi cung cấp dữ liệu đó ở phiên hiện tại hay không. Thử nghiệm gồm ba yêu cầu với ba phiên riêng biệt lần lượt, nội dung yêu cầu ở phiên sau phải có tính kế thừa thông tin ở phiên trước đó và chứa dữ liệu phản hồi của các phiên trước đó.

### 2.4.2.1 Thử nghiệm 1: Kiểm tra khả năng lưu giữ thông tin và ngữ cảnh của mô hình

*LLM ở các phiên riêng biệt khi không cung cấp thông tin của các phiên trước.*

Kết quả chứng minh được mô hình LLM không có khả năng lưu giữ thông tin và ngữ cảnh của mô hình trên nhiều phiên tương tác khác nhau. Cụ thể, ở yêu cầu đầu tiên, mô hình đã tạo mã C++ theo đúng yêu cầu. Tuy nhiên, khi ở yêu cầu thứ hai, yêu cầu mô hình nâng cấp đoạn mã C++ đã tạo trước đó, mô hình đã không thể truy xuất và nhận diện thông tin từ phiên trước do đó mô hình đã tạo ra một đoạn mã hoàn toàn mới bằng ngôn ngữ Python, cho thấy rõ ràng sự thiếu hụt khả năng ghi nhớ và liên tục ngữ cảnh giữa các yêu cầu khác nhau.

Bảng 2.1 Kết quả khi kiểm tra khả năng lưu giữ thông tin và ngữ cảnh của mô hình LLM ở các phiên riêng biệt khi không cung cấp thông tin của các phiên trước đó

Lần	Người dùng	Mô hình LLM
1	<pre>[   {     "role": "assistant",     "content": "Simple answer, no description, no comments. For mã nguồn, do not add features beyond requirements"   },   {     "role": "user",     "content": "Write me a C++ program to calculate the sum from 1 to n, where n is entered from the keyboard."   } ]</pre>	<pre>{   "role": "assistant",   "content":   """cpp\n#include&lt;iostream&gt;\nusing namespace std;\n\nint main() {\n int n, sum = 0;\n    cout &lt;&lt; \"Enter a number: \";\n    cin &gt;&gt; n;\n for(int i = 1; i &lt;= n; ++i) {\n sum += i;\n    }\n    cout &lt;&lt; \"Sum from 1 to \" &lt;&lt; n &lt;&lt; \" is: \" &lt;&lt; sum &lt;&lt; endl;\n    return 0;\n}\n\"""" }</pre>
2	<pre>[   {     "role": "assistant",     "content": "Simple answer, no description, no comments. For mã nguồn, do not add features beyond requirements"   },   {     "role": "user",     "content": "Upgrade the program to catch an exception when the user enters n as a character. If the user enters a character, print 0 and exit."   } ]</pre>	<pre>{   "role": "assistant",   "content": "Certainly! Below is a simple Python program that handles exceptions if the user inputs a non- integer value (e.g., a character):\n\n```python\ndef get_integer_input():\n    try:\n # Attempt to get an integer input from the user\n        n = int(input("Please enter an integer: \n"))\n        return n\n    except ValueError:\n        # If a ValueError occurs, print 0 and exit the program\n print(0)\n        exit()\n\n# Call the function to get the user's input\nuser_input = get_integer_input()\nprint(f"You entered: {user_input}")\n\n```\n\nThis mã nguồn will catch any `ValueError` that occurs if the user enters a non- integer value, print 0, and then exit the program. If the user enters an integer, it will continue to execute and print the entered value." }</pre>

#### 2.4.2.2 Thử nghiệm 2: Kiểm tra khả năng lưu giữ thông tin và ngữ cảnh của mô hình LLM ở các phiên riêng biệt khi cung cấp thông tin của các phiên trước.

Kết quả cho thấy mô hình LLM có khả năng ứng dụng được “Khung đánh giá TGen” khi kết quả phản hồi ở các yêu cầu sau không chỉ có sự liên kết chặt chẽ với các



yêu cầu trước đó mà còn cho thấy sự cải thiện đáng kể về chất lượng và độ chính xác của kết quả theo từng bước tương tác. Để cho ngắn gọn và dễ quan sát thì về mặt trình bày sẽ khác nhiều so với kết quả ở thử nghiệm 1 do ở thử nghiệm này chỉ lấy giá trị ở thuộc tính “content”.

Bảng 2.2 Kết quả kiểm tra khả năng lưu giữ thông tin và ngữ cảnh của mô hình LLM ở các phiên riêng biệt khi cung cấp thông tin của các phiên trước đó

Lần	Người dùng	Mô hình LLM
1	"Write me a C++ program to calculate the sum from 1 to n, where n is entered from the keyboard."	<pre>#include &lt;iostream&gt; using namespace std; int main() {     int n, sum = 0;     cout &lt;&lt; "Enter a positive integer: ";     cin &gt;&gt; n;     for (int i = 1; i &lt;= n; ++i) {         sum += i;     }     cout &lt;&lt; "Sum from 1 to " &lt;&lt; n &lt;&lt; " is: " &lt;&lt; sum &lt;&lt; endl;     return 0; }</pre>
2	Modify the C++ program so that when asked to enter the number n, it does not print any other information. Only print the result, no other information!	<pre>#include &lt;iostream&gt; using namespace std; int main() {     int n, sum = 0;     cin &gt;&gt; n;     for (int i = 1; i &lt;= n; ++i) {         sum += i;     }     cout &lt;&lt; sum &lt;&lt; endl;     return 0; }</pre>
3	Upgrade the program to catch an error when the user enters n which is not a number but a character. If the user enters a character, print 0 and exit.	<pre>#include &lt;iostream&gt; using namespace std; int main() {     int n, sum = 0;     if (!(cin &gt;&gt; n)) {         cout &lt;&lt; 0 &lt;&lt; endl;         return 1;     }     for (int i = 1; i &lt;= n; ++i) {         sum += i;     }     cout &lt;&lt; sum &lt;&lt; endl;     return 0; }</pre>

### 2.4.3 Kiểm tra hiệu quả việc biên dịch – thực thi – testcase của mã nguồn C++ bằng ngôn ngữ Python

Việc này kiểm tra tính hiệu được độ hiệu quả khi dùng Python để biên dịch – thực thi – chạy testcase với mã nguồn C++ có sẵn và bộ kiểm thử có sẵn. Về dữ liệu thì sẽ sử dụng lại mã nguồn C++ được tạo ở “Thử nghiệm 2” và một bộ kiểm thử để làm dữ liệu cho kiểm tra này. Về cấu trúc thư mục, các tệp tin được đặt cùng cấp với nhau, gồm các tệp tin:

- “test-engine.py”: Mã nguồn Python để biên dịch – thực thi – chạy testcase đối với mã nguồn C++.

```
import json

import subprocess

cpp_tệp_tin = "program-cpp.cpp"
exe_tệp_tin = "./program-cpp"

# Biên dịch chương trình C++

compile_process = subprocess.run(["g++", cpp_tệp_tin, "-o", exe_tệp_tin], capture_output=True,
text=True)

if compile_process.returncode != 0:

    print(compile_process.stderr)

    exit(1)

# Nạp testcase

with open('testcase.json', 'r') as f:

    test_cases = json.loads(f.read())

# Chạy mã nguồn C++ với từng testcase và so sánh kết quả

for i, test in enumerate(test_cases, 1):

    process = subprocess.run(

        [exe_tệp_tin, input=test["input"], text=True, capture_output=True

    )

    result = process.stdout.strip()

    expected = test["output"]

    print(f"\nTest {i:<5} Input: {test['input']:<10} Output: {result:<10} Expected:

{expected:<10} Status: {'PASSED' if result == expected else f'FAILED'}")
```

- “propram-cpp.cpp”: Chứa mã nguồn C++ cần biên dịch – thực thi – chạy với testcase.

```
#include <iostream>

using namespace std;

int main() {
    int n, sum = 0;
    if (!(cin >> n)) {
        cout << 0 << endl;
        return 1;
    }
    for (int i = 1; i <= n; ++i) {
        sum += i;
    }
    cout << sum << endl;
    return 0;
}
```

- “testcase.json”: Chứa bộ kiểm thử với nhiều test khác nhau.

```
[
  { "input": "5", "output": "15" },
  { "input": "3", "output": "6" },
  { "input": "ch", "output": "0" },
  { "input": "8", "output": "36" }
]
```

Kết quả sau khi thực hiện chạy tệp tin “test-engine.py” trên terminal:

Test 1	Input: 5	Output: 15	Expected: 15	Status: PASSED
Test 2	Input: 3	Output: 6	Expected: 6	Status: PASSED
Test 3	Input: ch	Output: 0	Expected: 0	Status: PASSED
Test 4	Input: 8	Output: 36	Expected: 36	Status: PASSED

Kết quả thử nghiệm đã chứng minh rằng việc tận dụng ngôn ngữ lập trình Python và hệ sinh thái phong phú của nó, đặc biệt là thư viện subprocess, mang lại hiệu quả cao trong việc tự động hóa quy trình biên dịch, thực thi và kiểm thử mã nguồn C++. Quá trình này được thực hiện với tốc độ rất nhanh. Để toàn bộ quá trình biên dịch, thực thi và kiểm tra với bộ kiểm thử đã được hoàn thành chỉ trong khoảng 0.44 giây

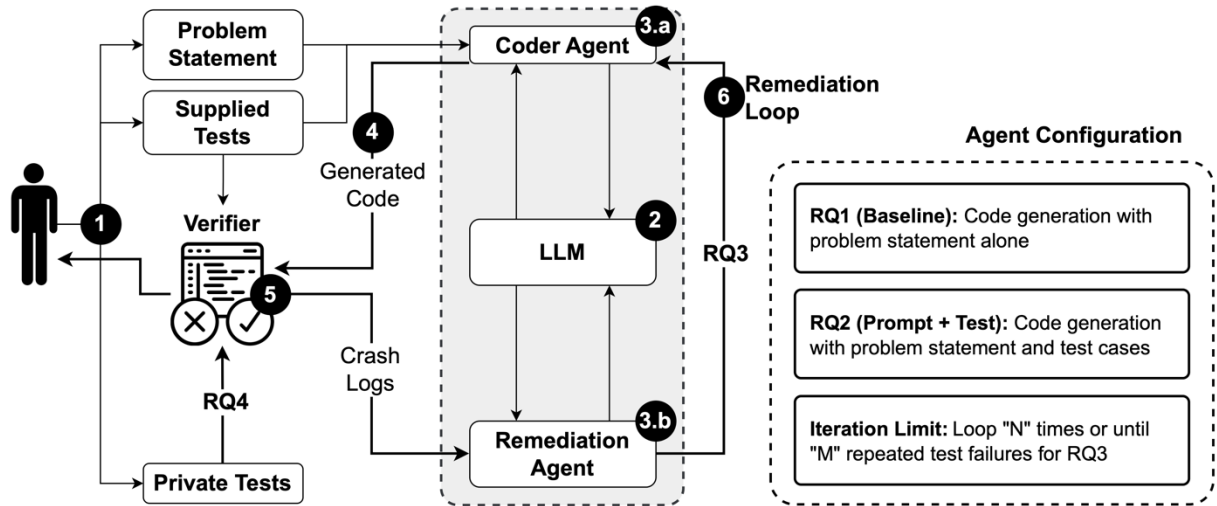
## 2.5. Thiết kế hệ thống

### 2.5.1 Lựa chọn công nghệ

Việc lựa chọn ngôn ngữ lập trình Python cho việc xây dựng back-end của hệ thống là một quyết định có tính phát triển cao, với nhu cầu tương tác sâu rộng và linh hoạt với mô hình ngôn ngữ lớn (LLM). Python cung cấp bộ công cụ và thư viện đầy đủ cho việc xử lý đa dạng các loại dữ liệu và đa dạng cấu trúc dữ liệu giúp cho nó có khả năng thích ứng cao khi có sự thay đổi trong cách triển khai mô hình LLM. Ví dụ, việc chuyển đổi từ việc chạy LLM cục bộ thông qua LM Studio sang việc kết nối trực tiếp với một dịch vụ LLM trên đám mây hoặc LLM trực tiếp trên máy cục bộ sẽ trở nên thuận tiện và ít phức tạp hơn nhờ vào tính linh hoạt của Python. Trong hệ sinh thái Python, Flask là một framework micro web framework được ưa chuộng và chứng minh được sự hiệu quả trong việc xây dựng các API back-end đơn giản nhưng mạnh mẽ. Sự nhẹ nhàng và dễ sử dụng của Flask cho phép đội ngũ phát triển nhanh chóng tạo ra các API cần thiết để giao tiếp với mô hình LLM và phục vụ dữ liệu cho giao diện người dùng.

Đối với việc xây dựng giao diện người dùng (front-end), ReactJS là một thư viện rất được ưa chuộng hiện nay. ReactJS mang lại nhiều lợi ích quan trọng. Kiến trúc dựa trên component của ReactJS giúp phân tách ứng dụng thành các module độc lập, từ đó tăng cường khả năng bảo trì, tái sử dụng mã và dễ dàng nâng cấp các thành phần riêng lẻ mà không ảnh hưởng đến toàn bộ ứng dụng. Việc sử dụng ReactJS là một lựa chọn hợp lý bởi khả năng tương tác hiệu quả với back-end thông qua các API và khả năng xử lý dữ liệu một cách linh hoạt nhờ vào sức mạnh của JavaScript. Với ReactJS, việc xây dựng một giao diện người dùng trực quan, phản ứng nhanh nhạy và cung cấp trải nghiệm tốt cho người dùng.

### 2.5.2 Quy trình xử lý chung của hệ thống



Hình 2.2 Khung đánh giá TGen trong bài báo khoa học “Test-Driven Development and LLM-based Code Generation”

Trong bài báo khoa học “Test-Driven Development and LLM-based Code Generation”, “Khung đánh giá TGen” được giới thiệu là một quy trình tích hợp phương pháp Phát triển Hướng theo Kiểm thử (TDD) nhằm mục đích chính là đánh giá độ chính xác của mã nguồn do các mô hình ngôn LLM tạo ra. Khung đánh giá TGen sử dụng một cách tiếp cận độc đáo với hai bộ kiểm thử riêng biệt: “Supplied Tests” được cung cấp cho LLM cùng với mô tả chi tiết về bài toán cần giải quyết, và “Private Tests” được giữ kín, chỉ sử dụng để kiểm tra mã nguồn đã được tạo.

Tuy nhiên, để ứng dụng hiệu quả “Khung đánh giá TGen” với mục tiêu tối ưu hóa quá trình tạo mã nguồn, điều này là khác biệt so với mục đích ban đầu của khung đánh giá trong bài báo khoa học, vốn tập trung vào việc kiểm tra và đánh giá độ chính xác của mã đã được tạo ra bởi mô hình LLM. Chính sự khác biệt căn bản này trong mục tiêu sử dụng đã đặt ra nhu cầu cần thiết phải điều chỉnh một số khía cạnh của khung đánh giá TGen để nó phù hợp hơn với việc hướng dẫn và hỗ trợ mô hình LLM trong quá trình tạo mã hiệu quả. Với mục tiêu tận dụng tối đa những ưu điểm của “Khung đánh giá TGen” trong việc tạo mã nguồn bằng mô hình LLM, đặc biệt là đối với ngôn ngữ lập trình C++.

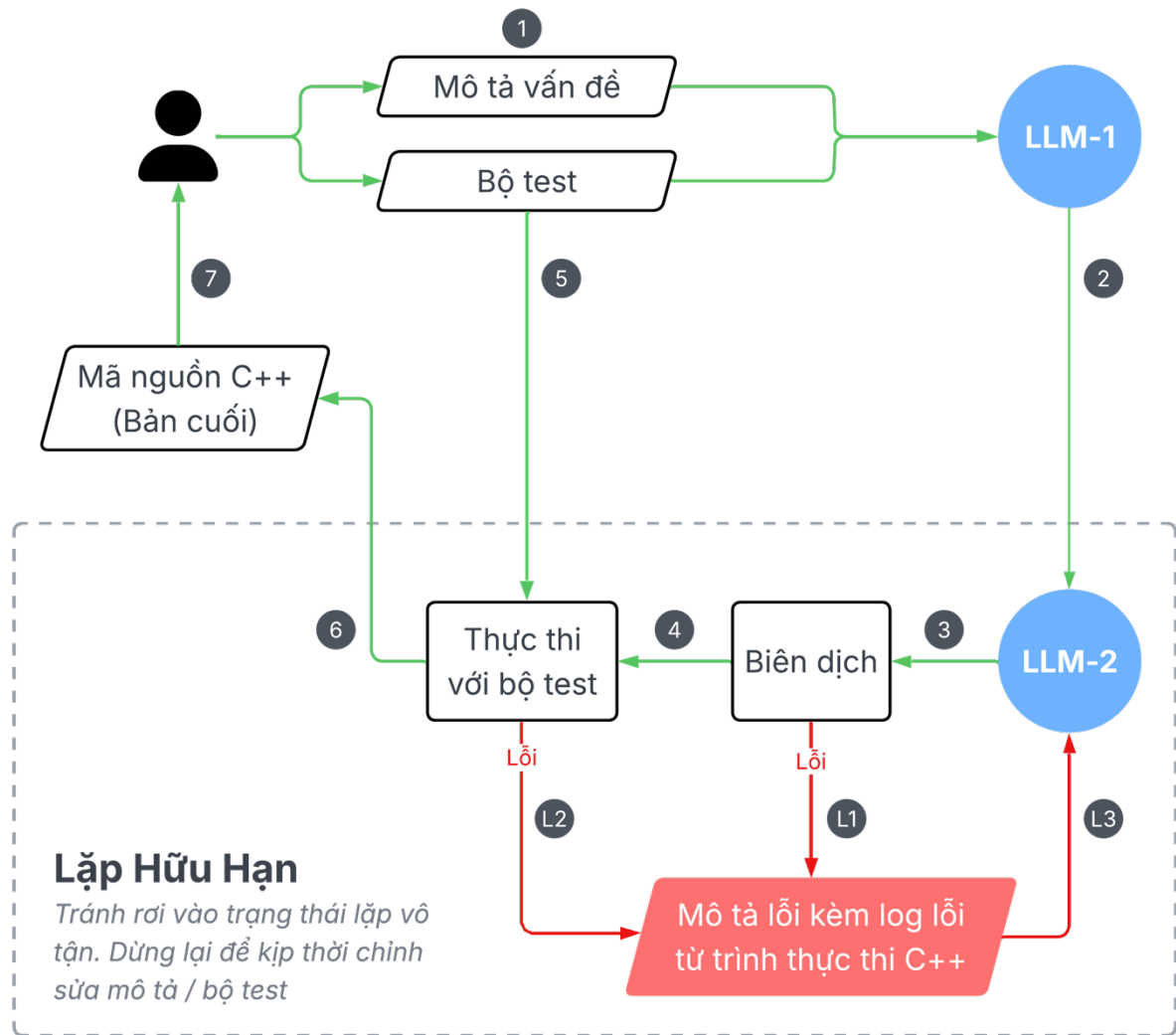
Tôi đã thiết kế lại khung đánh giá này ở một số điểm quan trọng như sau:

- Hợp nhất bộ kiểm thử: Thay vì duy trì hai bộ kiểm thử riêng biệt, thì cần hợp nhất “Supplied Tests” và “Private Tests” thành một bộ duy nhất. Bộ kiểm thử hợp nhất này sẽ được cung cấp đầy đủ cho mô hình LLM ngay

từ đầu. Mục đích của việc này là cung cấp cho LLM một bức tranh toàn diện và rõ ràng hơn về các yêu cầu và tiêu chí đánh giá, từ đó kỳ vọng sẽ dẫn đến kết quả tạo mã chính xác và đáp ứng tốt nhất các yêu cầu đặt ra.

- Xử lý thông tin yêu cầu: Để tăng cường hiệu quả tương tác với mô hình LLM, thì cần thiết việc thực hiện các bước xử lý thông tin mà người dùng cung cấp trước khi gửi đến mô hình. Cụ thể:
  - Chuyển đổi sang tiếng Anh: Do phần lớn các mô hình LLM hiện nay được huấn luyện và hoạt động hiệu quả nhất với ngôn ngữ tiếng Anh, việc chuyển đổi yêu cầu của người dùng sang tiếng Anh có thể giúp cải thiện đáng kể chất lượng phản hồi của mô hình.
  - Chuẩn hóa ngữ nghĩa: Để đảm bảo tính nhất quán và rõ ràng trong yêu cầu thì cần chuẩn hóa ngữ nghĩa của thông tin người dùng cung cấp. Điều này bao gồm việc làm rõ các thuật ngữ, cấu trúc câu và đảm bảo rằng yêu cầu được diễn đạt một cách mạch lạc và không gây hiểu lầm cho mô hình LLM.

### Quy trình xử lý dựa trên khung đánh giá TGen hoàn chỉnh:



Hình 2.3 Quy trình xử lý chung của hệ thống

Quy trình thực hiện được thiết kế có bảy giai đoạn chính, đánh số từ 1 đến 7, thể hiện lộ trình tối ưu khi hệ thống đạt được mục tiêu tạo mã nguồn thành công ngay trong lần đầu tiên. Trong trường hợp mã nguồn được tạo ra gặp lỗi trong quá trình biên dịch, quy trình sẽ chuyển sang bước tiếp theo theo nhánh được ký hiệu “L1”, sau đó tiếp tục theo nhánh “L3”. Tương tự, nếu mã nguồn C++ không vượt qua các bộ kiểm thử do người dùng cung cấp, quy trình sẽ rẽ nhánh theo hướng “L2”, rồi đến nhánh “L3”.

Kiến trúc hệ thống bao gồm mô hình LLM-1, chịu trách nhiệm xử lý và chuẩn hóa dữ liệu đầu vào do người dùng cung cấp. Mô hình LLM-2 đóng vai trò trung tâm, đảm nhiệm việc tạo mã nguồn C++ dựa trên mô tả đã được chuẩn hóa bởi LLM-1. Đặc biệt, thiết kế quy trình vẫn đảm bảo tính khả thi trong trường hợp các hệ thống có giới hạn về tài nguyên, cho phép triển khai chỉ với một mô hình LLM duy nhất để thực hiện cả hai vai trò xử lý dữ liệu đầu vào và tạo mã nguồn.

### 2.5.3 Giao diện người dùng

Giao diện (front-end) được thiết kế gồm hai trang chính, mang lại trải nghiệm người dùng liền mạch và dễ dàng điều hướng.

Trang đầu tiên là trang giới thiệu, đóng vai trò là điểm khởi đầu, cung cấp cái nhìn tổng quan về ứng dụng, mục đích sử dụng và có thể bao gồm các hướng dẫn cơ bản hoặc thông tin chào mừng người dùng.

Trang hoạt động chính, trung tâm của ứng dụng, được cấu trúc theo dạng tab để tối ưu hóa việc hiển thị và quản lý các chức năng khác nhau. Trang này bao gồm ba tab chính:

- Tab mô tả bài toán: Tại đây, người dùng sẽ được cung cấp đầy đủ thông tin chi tiết về bài toán hoặc yêu cầu cần xử lý. Tab này cho phép nhập thông tin bằng cách gõ phím hoặc upload tệp tin theo cấu trúc đã được quy định và cho phép tải về tệp tin mẫu ở ngay tại tab này.
- Tab kết quả tạo mã nguồn: Sau khi hệ thống xử lý thông tin đầu vào, tab này sẽ hiển thị kết quả là đoạn mã được tạo ra. Người dùng có thể xôi hoặc sao chép mã.
- Tab thống kê: Tab này cung cấp các số liệu thống kê quan trọng liên quan đến quá trình xử lý và kết quả tạo mã nguồn. Các thông tin này gồm thời gian xử lý, số lượng mã được tạo, tỉ lệ cải thiện mã nguồn, chất lượng mã nguồn,....

### 2.5.4 Chuẩn hóa cấu trúc tệp tin thông tin bài toán

Nhằm hỗ trợ việc tương tác tốt hơn nhập thông tin trực tiếp trên giao diện, upload tệp tin là một tính năng quan trọng và đang trở thành xu hướng chung trong việc truyền tải thông tin trên các hệ thống web.

Bên cạnh đó việc sử dụng thông tin của bài toán ở dạng tệp tin cho các mục đích khác khi người dùng cần trao đổi cho các bên khác như giữa giảng viên với sinh viên, giáo viên với học sinh, trưởng nhóm với các thành viên nhóm,...

Vì thế cần chuẩn hóa cấu trúc trình bày dữ liệu ở các dạng tệp tin phổ biến nhằm giúp hệ thống xử lý chính xác dữ liệu mà còn dễ dàng cho người dùng đọc được.



Để nâng cao trải nghiệm người dùng và tạo điều kiện thuận lợi cho việc nhập liệu thông tin một cách trực quan, tính năng tải lên tệp tin (upload tệp tin) đóng vai trò vô cùng quan trọng và đang trở thành một xu hướng thiết yếu trong các hệ thống web hiện đại. Thay vì chỉ giới hạn ở việc nhập liệu trực tiếp qua giao diện, khả năng tải lên tệp tin cho phép người dùng truyền tải thông tin một cách nhanh chóng và hiệu quả hơn, đặc biệt khi dữ liệu cần cung cấp có cấu trúc phức tạp hoặc dung lượng lớn.

Bên cạnh đó, việc lưu trữ thông tin về bài toán cần giải quyết dưới dạng tệp tin mang lại nhiều lợi ích thiết thực khác. Người dùng có thể dễ dàng sử dụng các tệp tin này cho nhiều mục đích khác nhau, chẳng hạn như chia sẻ thông tin với đồng nghiệp, cộng tác viên, hoặc trong môi trường giáo dục, giữa giảng viên và sinh viên, giáo viên và học sinh. Việc này tạo ra sự linh hoạt trong việc trao đổi và tham khảo thông tin, góp phần nâng cao hiệu quả làm việc và học tập.

Do đó, việc chuẩn hóa cấu trúc trình bày dữ liệu trong các định dạng tệp tin phổ biến là một yêu cầu then chốt. Điều này không chỉ đảm bảo rằng hệ thống có thể xử lý dữ liệu một cách chính xác và hiệu quả mà còn giúp người dùng dễ dàng đọc hiểu và quản lý thông tin. Việc áp dụng các tiêu chuẩn chung cho cấu trúc dữ liệu trong các định dạng như .json, .txt, hay .docx sẽ tạo ra sự nhất quán, giảm thiểu sai sót trong quá trình xử lý và nâng cao tính tiện dụng của hệ thống.

- Cấu trúc trình bày dữ liệu tệp tin “.json”:

```
{
  "problem": "Viết một chương trình...",
  "input_desc": "Một số nguyên duy...",
  "output_desc": "Gồm n số nguyên cách nhau ...",
  "test_case": [
    {"input": "<Dữ liệu đầu vào>", "output": "<Dữ liệu đầu ra>"},
    {"input": "<Dữ liệu đầu vào>", "output": "<Dữ liệu đầu ra>"},
    {"input": "<Dữ liệu đầu vào>", "output": "<Dữ liệu đầu ra>"}
  ]
}
```

Hình 2.4 Cấu trúc trình bày bài toán trong tệp tin “.json”

- Cấu trúc trình bày dữ liệu tệp tin “.docx”:

TÊN VẤN ĐỀ	
<b>Problem:</b> <Viết một chương trình ...>	
<b>Input Desc:</b> <Gồm một dãy số, cách nhau ...>	
<b>Output Desc:</b> <Kết quả số nguyên, trên một dòng duy ...>	
<b>Test Cases:</b>	
Input	Output
< Dữ liệu đầu vào>	< dữ liệu đầu ra >
< Dữ liệu đầu vào >	< dữ liệu đầu ra >
<...>	<...>

Hình 2.5 Cấu trúc trình bày bài toán trong tệp tin “.docx”

- Cấu trúc trình bày dữ liệu tệp tin “.txt”:

```

__Problem: Viết một chương trình ....
__Input_Desc: Gồm 2 số nguyên a và ....
__Output_Desc: In kết quả ra màn ....
__Test_Case:
<Dữ liệu đầu vào> -> <Dữ liệu đầu ra>
<Dữ liệu đầu vào> -> <Dữ liệu đầu ra>
<...> -> <...>

```

Hình 2.6 Cấu trúc trình bày bài toán trong tệp tin “.txt”

## 2.6. Sản phẩm đạt được

### 2.6.1 Hoạt động tổng quát của hệ thống

Hệ thống đã được chứng minh là vận hành ổn định và hiệu quả theo đúng “Quy trình xử lý chung của hệ thống” đã được thiết kế ở trên. Để đảm bảo tính toàn diện của quá trình kiểm nghiệm, hệ thống đã trải qua các thử nghiệm với nhiều mức độ phức tạp khác nhau của bài toán, được phân loại thành ba cấp độ: dễ, trung bình và khó. Việc kiểm thử trên dải độ khó đa dạng này cho phép đánh giá một cách toàn diện khả năng của hệ thống trong việc xử lý các yêu cầu khác nhau, từ những tác vụ cơ bản đến những thách thức phức tạp hơn. Các bài toán ở mức độ dễ thường tập trung vào các cấu trúc lập trình cơ bản và các yêu cầu đơn giản. Các bài toán mức độ trung bình đòi hỏi khả năng tư duy logic sâu hơn và có thể liên quan đến việc sử dụng các thuật toán hoặc cấu trúc dữ liệu phổ biến. Cuối cùng, các bài toán ở mức độ khó thường bao gồm các yêu cầu phức tạp, đòi hỏi sự kết hợp của nhiều kỹ năng lập trình và kiến thức chuyên sâu.

Kết quả từ các thử nghiệm này cho thấy hệ thống có khả năng xử lý trọn vẹn các tác vụ trong phạm vi đã được thiết kế.

## 2.6.2 Giao diện người dùng

Xây dựng giao diện người dùng đã thành công, giao diện chú trọng đến khía cạnh thẩm mỹ (UI) và kết hợp với các nguyên tắc trải nghiệm người dùng (UX). Thiết kế được triển khai hướng đến việc ngăn chặn các thao tác không hợp lệ, đồng thời tổ chức và trình bày dữ liệu một cách phong phú nhưng vẫn đảm bảo người dùng có thể dễ dàng nắm bắt được những thông tin cốt lõi và quan trọng nhất.

### 2.6.2.1 Tab mô tả bài toán

localhost:3000/generate

### AI Generate CPP Code

Tạo code mới Kết quả Thống kê

Tải lên file (.json, .txt, .docx) File mẫu cấu trúc đề bài

**Đề bài**

Write a program to calculate the sum of even numbers in a list of integers.

**Mô tả dữ liệu đầu vào**

The input data is a sequence of numbers, the first number is the number of elements in the list, the remaining numbers are the numbers in the list.

**Mô tả dữ liệu đầu ra**

The output is a single integer representing the sum of all even numbers in the list.

**Các bộ dữ liệu mẫu**

4 7 4 6 2	12	-
3 3 2 1	2	-

+ More test

Tạo code

Hình 2.7 Giao diện tab mô tả bài toán

Các ô nhập thông tin được tách biệt: Phân chia các trường nhập liệu thông tin giúp người dùng cung cấp mô tả bài toán một cách chính xác và có cấu trúc hơn. Cách

bố trí này hướng dẫn người dùng điền đầy đủ các chi tiết cần thiết, nhằm tăng cường sự rõ ràng và giảm thiểu sự mơ hồ trong yêu cầu.

Khu vực nhập bộ kiểm thử được thiết kế rõ ràng: Để tạo điều kiện thuận lợi cho việc cung cấp các bộ kiểm thử theo quy trình (TDD), khu vực nhập liệu được thiết kế để phân biệt một cách trực quan giữa dữ liệu đầu vào và dữ liệu đầu ra mong đợi cho mỗi trường hợp kiểm thử. Bên cạnh đó, mỗi test case được trang bị một nút “Hủy” cho phép người dùng dễ dàng loại bỏ một trường hợp kiểm thử cụ thể khỏi bộ kiểm thử khi cần thiết, mang lại sự linh hoạt và khả năng kiểm soát cao trong quá trình xây dựng bộ kiểm thử.

Ràng buộc về bộ kiểm thử: Hệ thống được thiết kế để yêu cầu người dùng cung cấp ít nhất một bộ kiểm thử trước khi cho phép thực hiện yêu cầu tạo mã nguồn. Quy tắc này được áp dụng nhằm đảm bảo rằng quy trình tạo mã tuân thủ theo nguyên tắc của TDD, nơi các kiểm thử đóng vai trò là tiêu chí đánh giá và hướng dẫn quá trình phát triển. Việc này giúp tăng khả năng tạo ra mã nguồn chính xác và đáp ứng đúng yêu cầu của người dùng ngay từ đầu.

Nút tải lên tệp tin bài toán: Như đã đề cập ngoài các ô nhập liệu trực tiếp, thì còn có một nút chức năng cho phép người dùng tải lên tệp tin chứa mô tả bài toán. Nút này, cho phép người dùng dễ dàng chọn và tải các tệp tin đã được chuẩn bị sẵn từ thiết bị của họ. Tính năng này đặc biệt hữu ích khi người dùng đã có sẵn mô tả bài toán ở các định dạng tài liệu điện tử.

Nút tải cấu trúc mẫu tệp tin bài toán: Để hỗ trợ người dùng trong việc chuẩn bị các tệp tin theo đúng định dạng mà hệ thống có thể xử lý, thì giao diện cũng cung cấp ba nút chức năng với biểu tượng của từng loại tệp tin mà hệ thống cho phép. Khi người dùng nhấp vào một trong các nút này, hệ thống sẽ cung cấp các tệp tin mẫu tương ứng với từng loại định dạng tệp tin được hỗ trợ (ví dụ: JSON, DOCX, TXT). Các tệp tin mẫu này sẽ trình bày rõ ràng cấu trúc dữ liệu cần thiết, bao gồm các trường thông tin bắt buộc và định dạng dữ liệu mong đợi cho cả mô tả bài toán và bộ kiểm thử.

### 2.6.2.2 Tab kết quả mã nguồn đã tạo

The screenshot shows a web browser at `localhost:3000/generate` with the title "AI Generate CPP Code". It has three buttons: "Tạo code mới", "Kết quả" (highlighted), and "Thống kê". Below the buttons, the "Code C++ đã tạo" section displays the following C++ code:

```
#include <iostream>
using namespace std;

int main() {
    int n;
    cin >> n;
    while (n-- > 0) {
        int num, first_digit, last_digit;
        cin >> num;
        last_digit = num % 10;
        if (num >= 10) {
            while (num >= 10) {
                num /= 10;
            }
            first_digit = num;
        } else {
            first_digit = num;
        }
        cout << (first_digit == last_digit ? "YES" : "NO") << " ";
    }
    return 0;
}
```

Below the code, the "Kết quả kiểm thử" section shows a table of test results:

STT	Đầu vào	KQ mong muốn	KQ thực tế	Trạng thái	Thời gian
1	4 58 626 131 891	NO YES YES NO	NO YES YES NO	Pass	0.8608s
2	3 697 67 67	NO NO NO	NO NO NO	Pass	0.0064s
3	8 0 87 333 98736 606 4444 3333 87928	NO NO YES NO YES YES YES YES	YES NO YES NO YES YES YES YES	Failed (Output mismatch)	0.0049s

Hình 2.8 Giao diện tab kết quả tạo mã nguồn

Sau khi quy trình tạo mã nguồn hoàn tất, tab “Kết quả” sẽ được kích hoạt, cho phép người dùng truy cập và xem xét các sản phẩm được tạo ra. Nội dung chính của tab này là mã nguồn C++ được tạo mới nhất (Mặc định mới nhất là mã nguồn ổn định và chính xác hơn), cùng với toàn bộ lịch sử các phiên bản mã nguồn C++ đã được tạo ra trong suốt quá trình xử lý. Việc cung cấp cả phiên bản tối ưu và các phiên bản trung gian tạo điều kiện cho người dùng theo dõi quá trình phát triển của mã và hiểu rõ hơn về cách mô hình LLM đã tiếp cận bài toán.

Để tối ưu hóa trải nghiệm người dùng, giao diện được trang bị chức năng sao chép nhanh mã nguồn. Thông qua một nút bấm trực quan, người dùng có thể dễ dàng

sao chép đoạn mã C++ mong muốn vào bộ nhớ tạm, phục vụ cho các mục đích sử dụng tiếp theo như tích hợp vào dự án hiện có hoặc phân tích sâu hơn trong môi trường phát triển tích hợp (IDE).

Phía dưới, một bảng kết quả chi tiết được hiển thị, cung cấp thông tin về hiệu suất của mã nguồn đã tạo khi chạy thử nghiệm với bộ kiểm thử do chính người dùng cung cấp. Bảng này có hai cột chính cần quan tâm là kết quả đầu ra mong đợi, kết quả đầu ra thực tế do mã nguồn tạo ra, và trạng thái của từng test case (ví dụ: Đạt, Không đạt).

### 2.6.2.3 Tab thống kê



Hình 2.9 Giao diện tab thống kê

Tab “Thông kê” cung cấp một cái nhìn toàn diện về các dữ liệu được thu thập và tổng hợp trong suốt quá trình hoạt động của hệ thống khi được yêu cầu tạo mã nguồn đến khi kết thúc. Mục đích của tab này là cung cấp cho người dùng những thông tin chi tiết và trực quan về hiệu suất và tiến trình của quá trình tạo mã.

Trong số các biểu đồ được hiển thị, có hai biểu đồ nổi bật sau:

- Biểu đồ “Tỉ lệ vượt qua test theo thời gian”: Biểu đồ này trực quan hóa sự thay đổi của tỉ lệ mã nguồn C++ vượt qua bộ kiểm thử theo tiến trình thời gian. “Thời gian” được định nghĩa là các lần tạo mã nguồn C++ thành công mới. Theo dõi sự biến động của tỉ lệ này qua các lần tạo kế tiếp cho phép người dùng đánh giá được hiệu quả của quá trình tự cải thiện và tinh chỉnh mã nguồn. Một xu hướng tăng lên trong biểu đồ này sẽ phản ánh sự cải thiện về chất lượng và độ chính xác của mã nguồn được tạo ra qua các lần thử nghiệm.
- Biểu đồ “Thời gian tạo code”: Biểu đồ này cung cấp thông tin chi tiết về thời gian cần thiết để tạo ra mã nguồn trong mỗi lần thử nghiệm. Dữ liệu này được ghi lại cho từng lần tạo mã, cho phép người dùng theo dõi sự khác biệt về thời gian phản hồi của mô hình LLM trong các lần tạo khác nhau. Thông tin này có thể hữu ích trong việc phân tích hiệu suất của mô hình và xác định các yếu tố có thể ảnh hưởng đến thời gian tạo mã.

## 2.7. Những điểm cần cải tiến

### 2.7.1 Mô hình LLM với phần mềm LM Studio

Cần cải thiện hiệu suất tối ưu trong việc xử lý của Mô hình LLM, cần tiếp tục thử nghiệm với nhiều loại model khác, không chỉ nhằm mục đích xác định mô hình đơn lẻ có hiệu suất vượt trội nhất, mà còn để đánh giá tiềm năng của việc kết hợp và vận hành song song nhiều mô hình chuyên biệt cho từng nhiệm vụ. Nếu theo hướng tiếp cận đa mô hình này, mỗi mô hình sẽ được thiết kế hoặc lựa chọn để đảm nhận một loại tác vụ cụ thể, tận dụng tối đa điểm mạnh riêng của từng kiến trúc. Ví dụ, một mô hình chuyên biệt cho việc tạo mã nguồn, có khả năng sinh ra các đoạn mã nguồn chất lượng cao, hiệu quả và tuân thủ các tiêu chuẩn lập trình tốt nhất. Đồng thời, một mô hình khác sẽ được tối ưu hóa cho việc hiểu ngôn ngữ tự nhiên, đảm bảo khả năng diễn giải chính xác và sâu sắc các yêu cầu và ngữ cảnh khác nhau. Điểm đặc biệt trong chiến lược này là việc

nghiên cứu và triển khai một mô hình thứ ba, có vai trò tạo ra các prompt (lời nhắc) tối ưu để gửi đến mô hình tạo mã nguồn. Mô hình này sẽ chịu trách nhiệm phân tích yêu cầu bằng ngôn ngữ tự nhiên, chuyển đổi chúng thành các prompt có cấu trúc và thông tin đầy đủ, giúp mô hình tạo mã nguồn hoạt động hiệu quả nhất. Sự phối hợp nhịp nhàng giữa các mô hình chuyên biệt này sẽ mang lại kết quả xử lý cao hơn và khả năng xử lý các tác vụ phức tạp tốt hơn.

Song song với việc khám phá các kiến trúc mô hình khác nhau, cần nghiên cứu phương án triển khai trực tiếp các mô hình trên máy tính, thay vì phải thông qua các nền tảng trung gian như LM Studio. Mục tiêu là tận dụng tối đa sức mạnh tính toán và các tài nguyên phần cứng sẵn có của máy tính. Việc triển khai trực tiếp giảm thiểu độ trễ và loại bỏ các rào cản về truyền dữ liệu, cho phép mô hình tiếp cận toàn bộ tiềm năng của bộ vi xử lý (CPU), bộ xử lý đồ họa (GPU) và bộ nhớ (RAM) của máy tính, từ đó cho phép sử dụng các model có tham số cao hơn đồng nghĩa với việc độ chính xác sẽ cao hơn. Hướng đi này cũng mang lại lợi ích về mặt bảo mật và quyền riêng tư, khi dữ liệu và quá trình xử lý được giữ hoàn toàn trên thiết bị.

### **2.7.2 Server Flask python**

Tái cấu trúc toàn diện kiến trúc mã nguồn. Thay vì duy trì một khối mã nguồn đơn lẻ, chúng tôi đã tiến hành phân tách các module chức năng thành nhiều tệp tin riêng biệt. Giúp làm rõ ràng hơn đáng kể luồng xử lý của ứng dụng, giúp dễ dàng phát triển dễ dàng theo dõi và hiểu được cách các thành phần tương tác với nhau. Việc mô-đun hóa mã nguồn tạo điều kiện thuận lợi cho việc bảo trì và nâng cấp hệ thống trong tương lai. Khi cần thay đổi hoặc thêm mới một chức năng, có thể tập trung vào module cụ thể mà không ảnh hưởng đến các phần khác của ứng dụng. Cuối cùng, việc phân tách rõ ràng cũng giúp xử lý lỗi hiệu quả hơn khi các tình huống đặc biệt xảy ra.

Để nâng cao hiệu quả khai thác tài nguyên và mang lại trải nghiệm người dùng tốt hơn, cần tích hợp hệ thống cơ sở dữ liệu vào server. Điều này cho phép lưu trữ một cách có hệ thống các dữ liệu quan trọng, bao gồm thông tin người dùng, phiên làm việc và đặc biệt là dữ liệu do người dùng tạo ra. Việc lưu trữ dữ liệu này mang lại nhiều lợi ích. Nó cho phép sử dụng lại các tài nguyên đã được tạo trước đó, thay vì phải khởi tạo lại dữ liệu mỗi khi cần, giúp tiết kiệm thời gian và tài nguyên hệ thống. Hơn nữa, việc lưu trữ phiên làm việc giúp duy trì trạng thái của người dùng giữa các phiên tương tác, mang



lại trải nghiệm liền mạch và nhất quán. Dữ liệu người dùng được lưu trữ một cách an toàn và có thể được khai thác để cung cấp các tính năng cá nhân hóa và phân tích sâu hơn trong tương lai.

Cải thiện trải nghiệm người dùng là điều đặc biệt quan trọng, đó là việc nâng cấp giao thức truyền thông giữa server và client. Cụ thể, là việc sử dụng giao thức WebSocket. Thay vì sử dụng các phương thức truyền thống dựa trên yêu cầu-phản hồi (request-response), WebSocket cho phép thiết lập một kết nối hai chiều liên tục giữa server và client. Điều này tạo điều kiện cho việc truyền dữ liệu theo dạng streaming một cách liên tục trong suốt quá trình tạo mã nguồn. Nhờ đó, người dùng có thể dễ dàng nắm bắt tiến độ xử lý theo thời gian thực, quan sát từng bước tạo mã và nhận được phản hồi ngay lập tức. Việc nâng cấp này mang lại một trải nghiệm tương tác mượt mà và trực quan hơn, giúp người dùng cảm thấy họ có thể kiểm soát và hiểu rõ hơn về quá trình hoạt động của hệ thống.

## Phần 3

### KẾT LUẬN

#### 3.1. Tự nhận xét, đánh giá

Qua 06 tuần thực tập tốt nghiệp tại Trung tâm Ngoại ngữ - Tin học Victory, bản thân tôi đã có cơ hội được thử sức mình với lĩnh vực trí tuệ nhân tạo cụ thể là sử dụng Mô hình LLM để phát triển một hệ thống tự động tạo mã nguồn từ mô tả bài toán bằng ngôn ngữ tự nhiên. Đây là một trải nghiệm vô cùng quý báu, giúp tôi vận dụng những kiến thức phần mềm đã học vào thực tế, đồng thời học hỏi thêm nhiều điều mới mẻ về lĩnh vực trí tuệ nhân tạo, đặc biệt là ứng dụng của LLM trong việc phát triển phần mềm.

Ưu điểm của hệ thống: Thực hiện việc tạo mã nguồn từ mô tả bằng ngôn ngữ tự nhiên. Test mã nguồn được tạo bởi AI theo mô hình TDD thành công. Giao diện người dùng dễ dàng tiếp cận và sử dụng. Tốc độ xử lý trong trường hợp xấu nhất (5 lần tạo lại mã nguồn) trong khoảng thời gian khá ngắn và chấp nhận được (Khoảng từ 1.5 phút đến 3.5 phút),...

Nhược điểm của hệ thống: Chưa có hệ thống xác thực – phân quyền. Chưa có hệ thống lưu trữ dữ liệu của người dùng. Hệ thống có thể bị crash trong trường hợp Mô hình LLM gặp sự cố khi phản hồi quá lâu hoặc cấu trúc data phản hồi chưa được nhận biết và xử lý. Giao thức truyền thông còn hạn chế khi chỉ sử dụng giao thức HTTP.

Những hạn chế trên của hệ thống là những điểm cần cải thiện về mặt kỹ thuật và phản ánh một phần quá trình học tập và trưởng thành của bản thân tôi trong suốt thời gian thực tập. Việc hệ thống còn thiếu sót ở một số khía cạnh trên được bản thân tôi nhìn nhận như những bài học kinh nghiệm sâu sắc về tầm quan trọng của việc phân bổ thời gian và quản lý dự án hiệu quả.

Cụ thể, việc tôi ưu tiên nghiên cứu lý thuyết nền tảng và thử nghiệm với các bài toán nhỏ ở giai đoạn đầu, dù cần thiết cho việc nắm vững kiến thức, đã phần nào làm giảm thời gian dành cho việc triển khai và hoàn thiện các tính năng nâng cao của hệ thống trong giai đoạn sau. Điều này dẫn đến việc một số mục tiêu, như tích hợp hệ thống xác thực và lưu trữ dữ liệu, chưa thực hiện được.

Tuy nhiên, tôi nhận thức rõ rằng việc nhận diện và phân tích những hạn chế này là một phần quan trọng của quá trình học tập và rèn luyện. Thông qua quá trình đó, tôi

đã có cơ hội nhìn nhận lại phương pháp làm việc của mình, hiểu rõ hơn về sự cân bằng giữa nghiên cứu lý thuyết và ứng dụng thực tế, cũng như tầm quan trọng của việc lập kế hoạch và quản lý thời gian một cách khoa học trong các dự án phát triển phần mềm. Mặc dù còn những điểm chưa hoàn thiện, tôi tin rằng những nỗ lực đã bỏ ra đã giúp tôi hoàn thành được các yêu cầu cơ bản của đợt thực tập và thu được những kinh nghiệm quý báu, làm nền tảng cho sự phát triển chuyên môn trong tương lai.

### **3.2. Kết luận, kiến nghị**

Qua quá trình thực tập, tôi đã tích lũy được những kinh nghiệm thực tiễn và kiến thức chuyên môn vô cùng giá trị. Đây không chỉ là hành trang quan trọng, củng cố sự tự tin khi bước vào môi trường làm việc chuyên nghiệp, mà còn là nguồn động lực to lớn thúc đẩy tôi không ngừng hoàn thiện và phát triển bản thân trong lĩnh vực Công nghệ Thông tin.

Với mong muốn đóng góp vào việc nâng cao chất lượng chương trình thực tập cho các sinh viên khóa sau, dựa trên những trải nghiệm cá nhân, tôi xin mạnh dạn đề xuất một số kiến nghị tới Bộ môn Công nghệ Thông tin như sau:

- Xây dựng quy trình thực tập đồng bộ và minh bạch: Thiết lập một quy trình rõ ràng, chi tiết từ giai đoạn đăng ký đến khi kết thúc thực tập, đồng thời đảm bảo thông tin được truyền đạt một cách nhất quán đến toàn bộ sinh viên. Điều này giúp sinh viên có sự chuẩn bị tốt hơn và chủ động hơn trong suốt quá trình thực tập.
- Tăng cường hoạt động kết nối sinh viên trước thực tập: Tổ chức các buổi gặp mặt, trao đổi thông tin chung trước khi sinh viên bắt đầu thực tập tại các đơn vị khác nhau. Hoạt động này tạo cơ hội để sinh viên nắm bắt thông tin một cách thống nhất, đồng thời xây dựng mạng lưới liên lạc, hỗ trợ lẫn nhau trong quá trình thực tập.
- Phát triển hệ thống tài liệu hướng dẫn thực tập toàn diện: Xây dựng và cung cấp hệ thống tài liệu chi tiết, dễ dàng tra cứu, bao gồm các hướng dẫn cụ thể về quy trình, yêu cầu, và các thông tin liên quan đến kỳ thực tập. Bên cạnh đó, nên bổ sung các tài liệu tham khảo về các dự án thực tế, các yêu cầu công việc thường gặp trong ngành, giúp sinh viên có cái nhìn thực tế hơn về môi trường làm việc chuyên nghiệp và các thách thức trong lĩnh vực Công nghệ Thông tin.

## **TÀI LIỆU THAM KHẢO**

- [1] M. N. Noble Saji Mathews, "Cornell University," 11 6 2024. [Online]. Available: <https://arxiv.org/abs/2402.13521>.