NHẬN XÉT CỦA GIẢNG VIÊN HƯỚNG DẪN

....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....

Trà Vinh, ngày ..... tháng …… năm ……

Giáo viên hướng dẫn

(Ký tên và ghi rõ họ tên)

NHẬN XÉT CỦA THÀNH VIÊN HỘI ĐỒNG

....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....
....

Trà Vinh, ngày ..... tháng …… năm ……

Thành viên hội đồng

(Ký tên và ghi rõ họ tên)

LỜI CẢM ƠN

Em xin gửi lời cảm ơn chân thành và sâu sắc nhất tới Cô Phan Thị Phương Nam vì những hướng dẫn tận tình và sự hỗ trợ quý báu mà cô đã dành cho em trong suốt thời gian học tập và nghiên cứu. Cô không chỉ truyền đạt cho em những kiến thức chuyên môn quý giá, mà còn dạy em cách tư duy khoa học và làm việc một cách nghiêm túc, cẩn thận. Những lời khuyên, góp ý và sự động viên của cô đã giúp em vượt qua nhiều khó khăn, hoàn thiện bản thân và đạt được những kết quả tốt hơn trong học tập. Sự tận tâm và nhiệt huyết của cô là nguồn cảm hứng lớn để em không ngừng nỗ lực, học hỏi và phấn đấu. Em cảm thấy vô cùng may mắn khi có cơ hội được học tập dưới sự hướng dẫn của cô. Một lần nữa, em xin gửi lời cảm ơn chân thành đến cô và chúc cô luôn mạnh khỏe, hạnh phúc và thành công trong sự nghiệp giảng dạy và nghiên cứu của mình.

MỤC LỤC

DANH MỤC HÌNH ẢNH

TÓM TẮT ĐỒ ÁN CHUYÊN NGÀNH

Tóm tắt vấn đề nghiên cứu

Trong bối cảnh hội nhập quốc tế, việc nâng cao khả năng sử dụng tiếng Anh, đặc biệt là kỹ năng Listening, ngày càng trở nên quan trọng. Đối với kỳ thi đánh giá năng lực tiếng Anh theo chuẩn VSTEP, kỹ năng Listening đóng vai trò thiết yếu, đòi hỏi thí sinh phải có khả năng phản xạ ngôn ngữ tốt trong nhiều tình huống giao tiếp. Tuy nhiên, các công cụ thi hiện nay còn hạn chế về tính tương tác và độ chuẩn xác, dẫn đến việc luyện tập không hiệu quả.

Hướng tiếp cận

Để xây dựng một website thi tiếng Anh kỹ năng Listening chuẩn VSTEP, cần tiếp cận bài toán một cách hệ thống và khoa học. Trước tiên, cần phân tích và xác định các chức năng chính: phát audio, hiển thị câu hỏi, nhận câu trả lời từ người dùng, chấm điểm tự động, và lưu trữ kết quả. Đồng thời, việc hiểu rõ cấu trúc bài thi Listening chuẩn VSTEP cùng các tiêu chí chấm điểm là rất quan trọng để đảm bảo tính chính xác và chuẩn mực. Về mặt công nghệ, cần lựa chọn các công cụ phù hợp. Phần frontend có thể sử dụng HTML, CSS hoặc React.js để xây dựng giao diện thân thiện và tương tác tốt với người dùng. Phần backend nên sử dụng Node.js (với Express.js) để xử lý logic, tạo API và đảm bảo hiệu suất hoạt động. Đối với cơ sở dữ liệu, lựa chọn NoSQL như MongoDB sẽ mang lại sự linh hoạt trong việc lưu trữ thông tin người dùng, bài thi, và kết quả. Quy trình phát triển cần được chia nhỏ thành các module chính để quản lý hiệu quả hơn, bao gồm: phát triển giao diện người dùng, xây dựng API xử lý dữ liệu, và thiết lập kết nối với cơ sở dữ liệu. Bằng cách tiếp cận có tổ chức và sử dụng các công nghệ phù hợp, website sẽ đáp ứng tốt yêu cầu bài thi Listening chuẩn VSTEP và mang lại trải nghiệm người dùng tối ưu.

Cách giải quyết vấn đề

Để xây dựng một website thi Listening chuẩn VSTEP, cần phân tích yêu cầu và xác định các chức năng chính như phát audio, hiển thị câu hỏi, nhận câu trả lời từ người dùng, tự động chấm điểm và lưu trữ kết quả thi. Về công nghệ, frontend có thể sử dụng HTML, CSS hoặc React.js để thiết kế giao diện người

dùng dễ sử dụng và mang lại trải nghiệm tương tác mượt mà. Backend sẽ sử dụng Node.js kết hợp với Express.js để xử lý các yêu cầu và logic nghiệp vụ, đồng thời tạo API phục vụ các chức năng của website. MongoDB là lựa chọn lý tưởng để lưu trữ thông tin người dùng, bài thi và kết quả, giúp quản lý dữ liệu linh hoạt và dễ mở rộng. Các chức năng chính sẽ bao gồm phát audio bằng HTML5 Audio, hiển thị câu hỏi sau khi audio kết thúc và nhận câu trả lời từ người dùng, so sánh câu trả lời với đáp án đúng và tự động tính điểm, và cuối cùng lưu trữ kết quả thi vào cơ sở dữ liệu MongoDB.

Kết quả đạt được

Kết quả đạt được sẽ là một website thi Listening chuẩn VSTEP hoàn chỉnh, với các chức năng chính hoạt động hiệu quả. Cụ thể, người dùng có thể nghe audio, trả lời câu hỏi sau khi audio kết thúc, và hệ thống sẽ tự động chấm điểm dựa trên câu trả lời của họ. Mọi thông tin người dùng, bài thi và kết quả sẽ được lưu trữ trong cơ sở dữ liệu MongoDB, giúp quản lý và theo dõi kết quả thi một cách dễ dàng. Giao diện người dùng được thiết kế mượt mà và dễ sử dụng, giúp người học có trải nghiệm tốt nhất khi làm bài thi. Backend sử dụng Node.js và Express.js sẽ xử lý logic và API một cách linh hoạt và hiệu quả, đảm bảo hệ thống hoạt động ổn định.

MỞ ĐẦU

Lý do chọn đề tài

Cùng với sự bùng nổ của công nghệ thông tin, việc phát triển các ứng dụng web hiện đại đóng vai trò ngày càng quan trọng trong cuộc sống và công việc hàng ngày. Cơ sở dữ liệu NoSQL đang trở thành những công cụ hàng đầu trong việc xây dựng ứng dụng web với hiệu năng cao, khả năng mở rộng tốt và trải nghiệm người dùng mượt mà. Hệ quản trị cơ sở dữ liệu NoSQL cho phép lưu trữ và truy vấn dữ liệu linh hoạt, phù hợp với nhu cầu phát triển ứng dụng hiện đại. Việc lựa chọn đề tài "Xây dựng hệ thống thi trắc nghiệm trực tuyến kỹ năng listenning theo chuẩn vstep với cơ sở dữ liệu NoSQL" không chỉ nhằm đáp ứng xu hướng phát triển công nghệ mà còn giúp tôi trang bị kiến thức chuyên sâu về những công cụ phổ biến nhất hiện nay, từ đó nâng cao năng lực lập trình và sẵn sàng cho thị trường lao động.

Mục đích

Mục tiêu chính của đề tài là xây dựng một hệ thống thi trắc nghiệm trực tuyến cho kỹ năng Listening theo chuẩn VSTEP, sử dụng cơ sở dữ liệu NoSQL để đảm bảo tính linh hoạt và hiệu quả trong quá trình quản lý dữ liệu. Đồng thời, nghiên cứu các phương pháp tối ưu hóa quy trình phát triển phần mềm. Các mục tiêu cụ thể bao gồm: Tạo ra một sản phẩm thực tiễn, đáp ứng đầy đủ yêu cầu của người dùng, đảm bảo tính khả dụng và hiệu quả. Nghiên cứu chuyên sâu về việc sử dụng React để xây dựng giao diện người dùng linh hoạt, Node.js để xử lý logic phía server và MongoDB để quản lý dữ liệu hiệu quả. Phát triển kỹ năng thực hành cho sinh viên, giúp họ áp dụng kiến thức lý thuyết vào thực tế, đồng thời rèn luyện khả năng lập trình và giải quyết vấn đề trong môi trường phát triển phần mềm.

Đối tượng và phạm vi nghiên cứu

Đối tượng nghiên cứu của đề tài là hệ thống thi trắc nghiệm trực tuyến, đặc biệt là phần thi kỹ năng Listening theo chuẩn VSTEP. Đề tài tập trung vào việc xây dựng và tối ưu hóa hệ thống này, áp dụng các công nghệ hiện đại như React, Node.js và MongoDB trong quá trình phát triển.

Phạm vi nghiên cứu bao gồm các yếu tố sau: công nghệ sử dụng, nghiên cứu và áp dụng các công nghệ React cho việc xây dựng giao diện người dùng, Node.js cho xử lý logic phía server, và MongoDB cho việc quản lý cơ sở dữ liệu NoSQL. Tính năng của hệ thống, phát triển các tính năng chính của hệ thống thi trực tuyến, bao gồm việc tạo đề thi, chấm điểm tự động, và quản lý dữ liệu bài thi của người dùng. Quy trình phát triển phần mềm, nghiên cứu và áp dụng các phương pháp tối ưu hóa trong quá trình phát triển phần mềm, từ việc lên kế hoạch, thiết kế, lập trình đến triển khai và bảo trì hệ thống. Đánh giá hiệu quả, đánh giá tính khả dụng và hiệu quả của hệ thống qua các yếu tố như tốc độ, độ chính xác trong việc chấm điểm và khả năng xử lý đồng thời của hệ thống. Phạm vi nghiên cứu không bao gồm các yếu tố liên quan đến các hình thức thi trực tuyến khác ngoài kỹ năng Listening theo chuẩn VSTEP, cũng như không nghiên cứu sâu về các công nghệ hoặc nền tảng khác ngoài các công nghệ đã chọn.

CHƯƠNG 1: TỔNG QUAN

Hệ thống thi trắc nghiệm trực tuyến đang ngày càng trở nên phổ biến trong việc đánh giá kỹ năng của người học, đặc biệt là trong các kỳ thi chuẩn hóa như VSTEP. Trong bối cảnh nhu cầu luyện thi và đánh giá trình độ tiếng Anh gia tăng, những hệ thống này đã chứng minh được tính hiệu quả cao trong việc giúm người học nâng cao kỹ năng ngôn ngữ và giám sát kết quả một cách khách quan. Tuy nhiên, việc phát triển một hệ thống thi trực tuyến đầy đủ tính năng, đáp ứng được yêu cầu về tính chính xác, bảo mật và khả năng mở rộng vẫn còn gây nhiều thách thức.

Một trong những vấn đề quan trọng nhất trong việc xây dựng hệ thống thi trực tuyến là đạt được hiệu quả cao trong việc quản lý dữ liệu. Điều này đòi hỏi hệ thống có khả năng xử lý và tích hợp dữ liệu một cách hiệu quả, nhất là trong các kỳ thi với số lượng lớn người dùng và dữ liệu phức tạp. Ngoài ra, để hệ thống có thể đáp ứng nhanh chóng với sự thay đổi nhu cầu của người dùng, việc tối ưu hóa quy trình phát triển phần mềm là cách tiếp cận hữu hiệu. Quy trình này cần bảo đảm rằng hệ thống dễ dàng bảo trì, nâng cấp và phát triển trong tương lai.

Trước những yêu cầu và thách thức nêu trên, đề tài này đề xuất nghiên cứu và phát triển hệ thống thi trắc nghiệm trực tuyến cho kỹ năng Listening theo chuẩn VSTEP. Việc áp dụng các công nghệ hiện đại như React, Node.js và MongoDB không chỉ tăng hiệu quả xử lý mà còn cung cấp tính linh hoạt trong việc quản lý dữ liệu và tính năng mở rộng. Hệ thống sẽ tập trung vào việc cung cấp trải nghiệm người dùng mượt mà, đảm bảo giao diện thân thiện, tính năng xử lý logic chính xác và khả năng đáp ứng cao.

Nghiên cứu này cũng sẽ đề cao việc tối ưu hóa quy trình phát triển phần mềm nhằm giúp sinh viên, lập trình viên có thể áp dụng kiến thức lý thuyết vào thực tế. Thông qua quy trình nghiên cứu hệ thống, sinh viên có thể nâng cao kỹ năng tư duy logic, quản lý dự án và giải quyết các vấn đề phức tạp trong lĩnh vực phần mềm. Nghiên cứu cũng đề xuất các giải pháp để cải thiện khả năng bảo mật, tối ưu hiệu suất và giảm thiểu nguy cơ trong quá trình triển khai hệ thống. Tất cả

những yếu tố này đã được xem xét và áp dụng trong phát triển hệ thống, nhằm tạo nên một sản phẩm đầy đủ tính năng và mang lại giá trị thực tiễn cao.

CHƯƠNG 2: NGHIÊN CỨU LÝ THUYẾT

2.1 Cơ sở lý thuyết

Cơ sở lý thuyết của đồ án này được xây dựng trên nền tảng các lý thuyết và phương pháp trong các lĩnh vực phát triển phần mềm, hệ thống thi trắc nghiệm trực tuyến, và các công nghệ web hiện đại.

2.1.1 Phát triển phần mềm và các mô hình phát triển

- Phát triển phần mềm là quá trình xây dựng và duy trì các ứng dụng phần mềm, nhằm đáp ứng các yêu cầu của người dùng và hoạt động của tổ chức. Các mô hình phát triển phần mềm phổ biến bao gồm mô hình Waterfall (thác nước) và mô hình Agile (linh hoạt). Trong đồ án này, phương pháp Agile được lựa chọn vì tính linh hoạt, cho phép phản hồi nhanh chóng với các thay đổi trong yêu cầu và dễ dàng điều chỉnh trong suốt quá trình phát triển.

- Phát triển phần mềm là một quy trình phức tạp, bao gồm nhiều giai đoạn như thu thập yêu cầu, thiết kế, lập trình, kiểm thử và bảo trì. Hiện nay, các mô hình phát triển phần mềm phổ biến bao gồm:

Mô hình Waterfall (thác nước): Là mô hình phát triển tuyến tính, mỗi giai đoạn được thực hiện tuần tự và không thể quay lại. Ưu điểm: Dễ hiểu, phù hợp với dự án nhỏ có yêu cầu rõ ràng. Nhược điểm: Thiếu linh hoạt khi có thay đổi, không phù hợp với các dự án lớn hoặc phức tạp.

Mô hình Agile: Là mô hình linh hoạt, phát triển theo từng "sprint" (chu kỳ ngắn). Ưu điểm: Phù hợp với các dự án yêu cầu thay đổi liên tục, phản hồi nhanh chóng. Nhược điểm: Đòi hỏi sự phối hợp chặt chẽ giữa các bên liên quan.

Trong dự án này, phương pháp Agile được lựa chọn vì tính linh hoạt cao, cho phép dễ dàng điều chỉnh khi có thay đổi yêu cầu [1].

2.1.2 Hệ thống thi trắc nghiệm trực tuyến

- Hệ thống thi trắc nghiệm trực tuyến là ứng dụng phần mềm cho phép tổ chức các kỳ thi, đánh giá và chấm điểm tự động. Các hệ thống này thường yêu cầu tính chính xác, bảo mật và khả năng mở rộng cao để đáp ứng với lượng lớn

người dùng. Các yếu tố chính của hệ thống thi trực tuyến bao gồm giao diện người dùng, quản lý đề thi, tính năng chấm điểm tự động và bảo mật thông tin [2].

- React là một thư viện JavaScript mã nguồn mở được sử dụng để xây dựng giao diện người dùng (UI) trong các ứng dụng web. React nổi bật với khả năng xây dựng các ứng dụng đơn trang (SPA), giúp tăng hiệu suất và cải thiện trải nghiệm người dùng. Nó sử dụng khái niệm "component"(thành phần), cho phép chia nhỏ giao diện thành các phần tái sử dụng được, giúp quản lý mã nguồn hiệu quả và dễ bảo trì [3].

2.1.3 Node.js và JavaScript

Node.js là một nền tảng mã nguồn mở dựa trên công cụ V8 JavaScript của Google, được thiết kế để chạy JavaScript phía server. Với Node.js, các nhà phát triển có thể xây dựng các ứng dụng mạng mạnh mẽ, hiệu quả và có khả năng mở rộng cao. Một trong những đặc điểm nổi bật của Node.js là khả năng xử lý bất đồng bộ (asynchronous) và hướng sự kiện (event-driven), cho phép nó quản lý nhiều kết nối đồng thời mà không làm giảm hiệu suất. Điều này đặc biệt phù hợp trong các môi trường có lượng truy cập lớn hoặc cần xử lý nhiều yêu cầu trong thời gian thực, chẳng hạn như các ứng dụng trò chuyện (chat applications), game trực tuyến, hoặc hệ thống quản lý dữ liệu thời gian thực.

Khác với mô hình luồng (thread-based) truyền thống, Node.js sử dụng một vòng lặp sự kiện đơn luồng (single-threaded event loop), giúp giảm đáng kể chi phí bộ nhớ và tránh tình trạng tắc nghẽn (blocking). Điều này không chỉ tăng hiệu suất tổng thể mà còn giúp ứng dụng hoạt động ổn định hơn khi đối mặt với lượng yêu cầu lớn.

Ngoài ra, Node.js đi kèm với một hệ sinh thái phong phú thông qua npm (Node Package Manager), kho lưu trữ thư viện JavaScript lớn nhất thế giới. Với hàng triệu gói thư viện được cung cấp sẵn, các nhà phát triển có thể dễ dàng tích hợp các tính năng phức tạp vào ứng dụng mà không cần phải viết lại từ đầu. Điều này giúp tiết kiệm thời gian phát triển và tạo điều kiện thuận lợi cho việc xây dựng các giải pháp sáng tạo.

Node.js cũng hỗ trợ phát triển các ứng dụng đa dạng, từ các API RESTful, ứng dụng web toàn diện (full-stack) đến các dịch vụ microservices. Nhờ sự phổ biến của

JavaScript và cộng đồng nhà phát triển rộng lớn, Node.js đã trở thành một công cụ không thể thiếu trong lĩnh vực phát triển web hiện đại.

Tóm lại, với khả năng xử lý đồng thời mạnh mẽ, tính năng mở rộng dễ dàng, và hệ sinh thái thư viện phong phú, Node.js là một nền tảng lý tưởng để xây dựng các hệ thống yêu cầu hiệu suất cao, khả năng mở rộng và đáp ứng nhanh chóng với nhu cầu của người dùng [4].

2.1.4 MongoDB và cơ sở dữ liệu NoSQL

- **MongoDB** là một cơ sở dữ liệu NoSQL (Not Only SQL) mã nguồn mở, được thiết kế để lưu trữ và quản lý dữ liệu không có cấu trúc, đặc biệt là dữ liệu dạng tài liệu (document). MongoDB sử dụng mô hình dữ liệu kiểu "Document-Oriented", nơi dữ liệu được lưu trữ dưới dạng các tài liệu BSON (Binary JSON) thay vì các bảng và hàng như trong các hệ quản trị cơ sở dữ liệu quan hệ (RDBMS).

Dưới đây là một số khái niệm cơ bản trong MongoDB:

1. Database (Cơ sở dữ liệu): MongoDB cho phép tạo nhiều cơ sở dữ liệu trên một máy chủ. Mỗi cơ sở dữ liệu chứa các bộ sưu tập (collection) riêng biệt. Cơ sở dữ liệu trong MongoDB không cần phải được khai báo trước khi sử dụng, chúng sẽ được tạo tự động khi bạn insert dữ liệu vào.

2. Collection (Bộ sưu tập): Collection trong MongoDB tương đương với một bảng trong các cơ sở dữ liệu quan hệ. Các tài liệu trong một collection có thể có cấu trúc khác nhau, nghĩa là bạn không cần phải tuân thủ một schema cụ thể. Collection thường được sử dụng để nhóm các tài liệu có mối liên hệ với nhau.

3. Document (Tài liệu): Document là đơn vị dữ liệu cơ bản trong MongoDB, được lưu trữ dưới dạng BSON (Binary JSON). Một document tương tự như một bản ghi (row) trong các cơ sở dữ liệu quan hệ. Tài liệu có thể chứa các trường (field) với các kiểu dữ liệu khác nhau (số, chuỗi, mảng, v.v.), và có thể chứa các tài liệu con.

4. BSON (Binary JSON): BSON là định dạng dữ liệu nhị phân mà MongoDB sử dụng để lưu trữ tài liệu. Nó tương tự như JSON nhưng có thể lưu trữ nhiều kiểu dữ liệu hơn và tối ưu hơn về mặt hiệu suất.

5. Index (Chỉ mục): MongoDB cho phép tạo chỉ mục để tăng tốc độ truy vấn. Các chỉ mục giúp tìm kiếm nhanh chóng các tài liệu trong collection. Một số chỉ mục cơ bản bao gồm chỉ mục đơn (single-field index), chỉ mục đa trường (compound index), và chỉ mục đầy đủ (text index).

6. Replica Set (Bộ tái tạo): Replica Set là một nhóm các server MongoDB chứa bản sao của cùng một dữ liệu. Điều này đảm bảo tính sẵn sàng cao và phục hồi sau sự cố, vì nếu một node (máy chủ) bị lỗi, các node khác vẫn có thể tiếp tục phục vụ yêu cầu.

7. Sharding (Phân mảnh): Sharding là phương pháp phân phối dữ liệu lớn trên nhiều máy chủ để tối ưu hiệu suất và khả năng mở rộng. Dữ liệu trong MongoDB có thể được chia thành các phân mảnh (shard) và lưu trữ trên các máy chủ khác nhau, giúp xử lý lượng lớn dữ liệu.

8. Aggregation (Tổng hợp): Aggregation trong MongoDB là quá trình xử lý dữ liệu để thu thập và tính toán thông tin từ các tài liệu trong collection. MongoDB cung cấp các biểu thức aggregation mạnh mẽ, chẳng hạn như $match, $group, $sort, $project, và nhiều toán tử khác.

9. Primary Key và _id: Mỗi tài liệu trong MongoDB có một trường _id duy nhất để xác định tài liệu đó. Trường này là khóa chính (primary key) mặc định. Nếu không chỉ định trường _id khi tạo tài liệu, MongoDB sẽ tự động tạo giá trị duy nhất cho nó.

10. Change Streams: Change Streams là tính năng trong MongoDB giúp theo dõi và nhận thông báo khi có sự thay đổi trong cơ sở dữ liệu. Điều này rất hữu ích trong các ứng dụng cần phản ứng với sự thay đổi dữ liệu thời gian thực.

MongoDB cung cấp sự linh hoạt cao trong việc xử lý dữ liệu phi cấu trúc và có thể mở rộng dễ dàng, là một lựa chọn phổ biến trong các ứng dụng web, phân tích dữ liệu, và hệ thống xử lý dữ liệu lớn.

- Cơ sở dữ liệu NoSQL (Not Only SQL) là một loại cơ sở dữ liệu phi quan hệ, không sử dụng mô hình bảng quan hệ như các cơ sở dữ liệu SQL truyền thống. Thay vì sử dụng các bảng với các hàng và cột, NoSQL có thể lưu trữ dữ liệu dưới nhiều hình thức khác nhau như tài liệu (document), cặp khóa-giá trị

(key-value), đồ thị (graph), hoặc cột (column-family). Các cơ sở dữ liệu NoSQL được thiết kế để có thể mở rộng và đáp ứng các yêu cầu của các ứng dụng lớn, phân tán, với các loại dữ liệu không cấu trúc.

1. Các loại cơ sở dữ liệu NoSQL:

Cơ sở dữ liệu Key-Value (Cặp khóa-giá trị)**:** Dữ liệu được lưu trữ dưới dạng cặp khóa và giá trị. Đây là kiểu cơ sở dữ liệu đơn giản nhất trong NoSQL. Ví dụ: **Redis, Riak**.

Cơ sở dữ liệu Document (Tài liệu)**:** Dữ liệu được lưu trữ dưới dạng tài liệu (thường là JSON, BSON, hoặc XML). Mỗi tài liệu có thể có cấu trúc khác nhau. Ví dụ: **MongoDB, CouchDB.**

Cơ sở dữ liệu Column Family (Cột)**:** Dữ liệu được tổ chức thành các cột thay vì hàng. Thường được sử dụng trong các ứng dụng cần khả năng mở rộng cao và truy vấn nhanh. Ví dụ: **Apache Cassandra, HBase**.

Cơ sở dữ liệu Graph (Đồ thị)**:**Dữ liệu được tổ chức dưới dạng đồ thị với các đỉnh và cạnh. Rất hữu ích trong các ứng dụng như mạng xã hội, nơi các mối quan hệ giữa các đối tượng cần được mô hình hóa. Ví dụ: **Neo4j, ArangoDB**.

2. Các đặc điểm của NoSQL:

Khả năng mở rộng (Scalability): NoSQL hỗ trợ mở rộng ngang (horizontal scaling), tức là có thể dễ dàng thêm nhiều máy chủ để mở rộng hệ thống, điều này giúp hệ thống có thể xử lý khối lượng dữ liệu lớn và tăng trưởng mạnh mẽ.

Không có cấu trúc cố định: Các cơ sở dữ liệu NoSQL không yêu cầu dữ liệu phải tuân thủ một cấu trúc cố định như các bảng trong cơ sở dữ liệu SQL. Điều này giúp NoSQL linh hoạt hơn trong việc lưu trữ dữ liệu không cấu trúc hoặc bán cấu trúc.

Tối ưu cho các ứng dụng phân tán: NoSQL thường được sử dụng trong các ứng dụng cần phân tán dữ liệu và xử lý trong môi trường phân tán (distributed systems). Điều này giúp duy trì tính nhất quán và hiệu suất cao ngay cả khi hệ thống có sự thay đổi về số lượng máy chủ hoặc yêu cầu tải cao.

Hiệu suất cao: Do việc thiết kế của NoSQL tập trung vào hiệu suất cho các truy vấn đọc/ghi lớn, nó thường có hiệu suất cao hơn so với cơ sở dữ liệu quan hệ trong một số tình huống.

Khả năng không hỗ trợ ACID: Nhiều cơ sở dữ liệu NoSQL không hoàn toàn tuân thủ các nguyên lý ACID (Atomicity, Consistency, Isolation, Durability). Thay vào đó, nhiều cơ sở dữ liệu NoSQL sử dụng mô hình BASE (Basically Available, Soft state, Eventually consistent) để tối ưu hóa khả năng mở rộng và hiệu suất.

3. Lợi ích của NoSQL:

Mở rộng linh hoạt: NoSQL giúp các ứng dụng dễ dàng mở rộng và xử lý một lượng lớn dữ liệu không cấu trúc.

Hiệu suất cao: Các cơ sở dữ liệu NoSQL được tối ưu hóa cho các tác vụ đọc/ghi nhanh và có thể xử lý khối lượng dữ liệu khổng lồ.

Độ trễ thấp: Các cơ sở dữ liệu này thường có độ trễ thấp trong các truy vấn, giúp nâng cao hiệu quả của các ứng dụng thời gian thực.

Các ứng dụng của NoSQL:

Mạng xã hội: Cơ sở dữ liệu đồ thị như Neo4j được sử dụng để lưu trữ các mối quan hệ giữa người dùng trong mạng xã hội.

Ứng dụng web và di động: Các ứng dụng cần khả năng mở rộng lớn và phản hồi nhanh thường sử dụng NoSQL.

Phân tích dữ liệu lớn: NoSQL hỗ trợ các ứng dụng yêu cầu phân tích dữ liệu lớn và phức tạp, chẳng hạn như Hadoop với HBase.

Internet of Things (IoT)**:** Các hệ thống IoT cần quản lý và phân tích lượng dữ liệu lớn và không cấu trúc.

4. Các ví dụ về cơ sở dữ liệu NoSQL phổ biến:

MongoDB: Là một cơ sở dữ liệu tài liệu (document-oriented database) phổ biến, MongoDB lưu trữ dữ liệu dưới dạng JSON (hoặc BSON).Dễ dàng mở rộng và hỗ trợ các truy vấn phức tạp.

Cassandra: Một cơ sở dữ liệu cột (column-family database) mạnh mẽ, rất phù hợp cho các ứng dụng phân tán với dữ liệu lớn.

Cassandra được tối ưu hóa cho khả năng mở rộng và độ sẵn sàng cao.

Redis: Là một cơ sở dữ liệu key-value rất nhanh, thường được sử dụng trong các ứng dụng yêu cầu bộ nhớ đệm (cache) hoặc hàng đợi tin nhắn.

CouchDB: Cơ sở dữ liệu tài liệu với khả năng đồng bộ hóa giữa các máy chủ và hỗ trợ lưu trữ dữ liệu dưới dạng JSON.

Neo4j: Một cơ sở dữ liệu đồ thị, phù hợp cho các ứng dụng mà mối quan hệ giữa các đối tượng là quan trọng, như mạng xã hội [5].

- Các nguyên lý thiết kế hệ thống bao gồm tính phân tách trách nhiệm, khả năng mở rộng và bảo mật. Phân tách trách nhiệm giúp đảm bảo các thành phần của hệ thống có thể hoạt động độc lập, dễ dàng thay thế và bảo trì.

- Khả năng mở rộng là yêu cầu quan trọng cho các hệ thống trực tuyến, đặc biệt khi số lượng người dùng gia tăng.

- Bảo mật hệ thống đảm bảo dữ liệu của người dùng, kết quả thi và thông tin liên quan được bảo vệ an toàn khỏi các nguy cơ tấn công mạng [6] .

2.2 Cơ sở lý luận

Cơ sở lý luận của đồ án này được xây dựng dựa trên các lý thuyết và nguyên lý trong lĩnh vực phát triển phần mềm, thiết kế hệ thống, và kiểm tra, đánh giá trong giáo dục. Các lý thuyết này giúp định hướng và giải thích các quyết định kỹ thuật cũng như các phương pháp được áp dụng trong đồ án.

2.2.1 Lý thuyết về phát triển phần mềm

Phát triển phần mềm là một quá trình phức tạp bao gồm các giai đoạn chính như thiết kế, phát triển, kiểm tra và bảo trì các ứng dụng phần mềm. Đây không chỉ là việc viết mã mà còn đòi hỏi sự phối hợp giữa nhiều nhóm và vai trò khác nhau, từ quản lý dự án, phân tích nghiệp vụ, thiết kế giao diện người dùng (UI/UX), đến kiểm thử phần mềm và triển khai sản phẩm. Mục tiêu cuối cùng của quá trình này là tạo ra các sản phẩm phần mềm đáp ứng tốt nhu cầu của người dùng và đảm bảo tính ổn định, hiệu quả trong suốt vòng đời sản phẩm.

Một trong những lý thuyết quan trọng và phổ biến nhất trong phát triển phần mềm hiện đại là phương pháp luận Agile. Agile không chỉ là một khung lý thuyết mà còn là một triết lý làm việc, tập trung vào tính linh hoạt, khả năng thích ứng, và sự hợp tác liên tục giữa các thành viên trong nhóm. Agile được xây dựng dựa trên 12 nguyên tắc cốt lõi, trong đó nhấn mạnh việc cung cấp phần mềm hoạt động được theo từng chu kỳ ngắn, phản hồi nhanh chóng với các thay đổi, và ưu tiên sự tương tác con người hơn là quy trình cứng nhắc.

- **Phương pháp luận Agile: Scrum và Kanban**

Hai khung làm việc nổi bật nhất trong Agile là **Scrum** và **Kanban**, mỗi khung có cách tiếp cận riêng để tối ưu hóa quy trình phát triển phần mềm:

1. Scrum

Scrum là một phương pháp được tổ chức theo các chu kỳ làm việc ngắn, gọi là **Sprint**, thường kéo dài từ 1 đến 4 tuần. Các Sprint này đại diện cho một vòng lặp phát triển, trong đó nhóm tập trung hoàn thành một tập hợp các công việc cụ thể đã được lên kế hoạch từ trước. Mỗi Sprint kết thúc với một sản phẩm "có thể phát hành" (potentially shippable product), cho phép nhóm nhận phản hồi từ khách hàng hoặc người dùng cuối trước khi tiếp tục các giai đoạn tiếp theo.

Scrum dựa trên ba trụ cột chính:

Minh bạch (Transparency): Tất cả các thành viên trong nhóm phải hiểu rõ tiến độ và mục tiêu của dự án.

Thanh tra (Inspection): Nhóm thường xuyên kiểm tra tiến độ và chất lượng công việc trong các cuộc họp ngắn, như Daily Standup, Sprint Review và Sprint Retrospective.

Thích nghi (Adaptation): Dựa trên phản hồi, nhóm sẽ điều chỉnh cách làm việc hoặc ưu tiên trong Sprint tiếp theo.

Scrum còn quy định các vai trò cụ thể như:

Scrum Master: Người hỗ trợ nhóm vận hành theo đúng phương pháp Scrum, loại bỏ các trở ngại.

Product Owner: Người chịu trách nhiệm quản lý danh sách công việc (Product Backlog) và đảm bảo sản phẩm đáp ứng nhu cầu của khách hàng.

Nhóm phát triển: Các thành viên chịu trách nhiệm thực hiện công việc trong Sprint.

2. Kanban

Kanban, khác với Scrum, tập trung vào việc tối ưu hóa quy trình làm việc thông qua việc trực quan hóa các công việc trên bảng Kanban (Kanban Board). Bảng này chia công việc thành các cột như "Việc cần làm" (To Do), "Đang thực hiện" (In Progress), và "Hoàn thành" (Done). Các nhiệm vụ được đại diện bằng thẻ (card), và nhóm sẽ di chuyển thẻ qua các cột tương ứng khi hoàn thành từng giai đoạn công việc.

Kanban không yêu cầu các chu kỳ làm việc cố định như Sprint, mà thay vào đó nhấn mạnh việc duy trì luồng công việc liên tục và tránh tình trạng quá tải cho nhóm. Một trong những nguyên tắc cốt lõi của Kanban là giới hạn số lượng công việc đang thực hiện (WIP – Work in Progress), nhằm đảm bảo nhóm không bị phân tâm hoặc giảm hiệu suất.

Ưu điểm của Agile, Scrum và Kanban trong phát triển phần mềm

Tính linh hoạt: Agile cho phép các nhóm thích nghi nhanh chóng với các yêu cầu thay đổi từ phía khách hàng hoặc thị trường.

Cải thiện sự cộng tác: Thông qua các cuộc họp ngắn hàng ngày và phản hồi thường xuyên, các thành viên trong nhóm có thể giao tiếp tốt hơn và hiểu rõ vai trò của mình.

Phản hồi liên tục: Việc phát hành phần mềm theo từng giai đoạn nhỏ giúp nhóm nhận phản hồi sớm, giảm thiểu rủi ro phát triển sai hướng.

Tối ưu hóa hiệu suất: Cả Scrum và Kanban đều cung cấp các công cụ và nguyên tắc giúp nhóm làm việc hiệu quả hơn, từ việc quản lý ưu tiên đến tối ưu hóa quy trình.

Kết luận: Phương pháp luận Agile, đặc biệt là Scrum và Kanban, đã trở thành tiêu chuẩn vàng trong phát triển phần mềm hiện đại. Với cách tiếp cận linh

hoạt, tập trung vào phản hồi liên tục và cải tiến quy trình, Agile giúp các nhóm phát triển không chỉ đạt được chất lượng sản phẩm cao mà còn đáp ứng nhanh chóng với nhu cầu thay đổi của khách hàng, đồng thời cải thiện hiệu suất làm việc và sự hài lòng của các thành viên trong nhóm [7] .

2.2.2 Lý thuyết về hệ thống thi trắc nghiệm trực tuyến

Hệ thống thi trắc nghiệm trực tuyến là một công cụ hiện đại trong lĩnh vực giáo dục, được thiết kế nhằm tạo ra một môi trường học tập và kiểm tra vừa hiệu quả, vừa công bằng, đảm bảo tính chính xác và khả năng tự động hóa cao. Sự phát triển của các nền tảng này không chỉ đáp ứng nhu cầu kiểm tra, đánh giá trong bối cảnh học tập trực tuyến (e-learning), mà còn hỗ trợ nâng cao chất lượng giáo dục thông qua việc áp dụng các công nghệ tiên tiến và các lý thuyết kiểm tra hiện đại.

Các yếu tố cốt lõi của hệ thống thi trắc nghiệm trực tuyến

Độ tin cậy (Reliability): Một hệ thống thi trắc nghiệm hiệu quả phải đảm bảo độ tin cậy cao, tức là các bài kiểm tra cần cung cấp kết quả nhất quán và ổn định qua nhiều lần thực hiện. Để đạt được điều này, hệ thống cần áp dụng các thuật toán tạo đề thi ngẫu nhiên từ ngân hàng câu hỏi lớn, tránh lặp lại câu hỏi giữa các bài kiểm tra khác nhau. Đồng thời, hệ thống cũng phải bảo mật tốt để ngăn chặn các hành vi gian lận, đảm bảo rằng kết quả phản ánh đúng năng lực thực sự của người học.

Độ hợp lý (Validity): Độ hợp lý đề cập đến khả năng của bài kiểm tra trong việc đo lường chính xác các mục tiêu học tập đã được đặt ra. Điều này đòi hỏi các câu hỏi trong hệ thống phải được thiết kế một cách cẩn thận, phù hợp với nội dung và mục tiêu giảng dạy. Ngoài ra, hệ thống cần cung cấp các chức năng giúp giảng viên phân loại câu hỏi theo các cấp độ tư duy (như ghi nhớ, hiểu, vận dụng, và phân tích) dựa trên thang đo Bloom, từ đó xây dựng được các bài thi phù hợp với nhiều đối tượng học viên.

Tính công bằng: Công bằng là một yếu tố thiết yếu trong bất kỳ hệ thống thi cử nào. Hệ thống thi trắc nghiệm trực tuyến cần đảm bảo mọi thí sinh đều được kiểm tra trong điều kiện tương đương, không bị ảnh hưởng bởi yếu tố ngoài lề như thời gian truy cập, lỗi kỹ thuật, hoặc khả năng tiếp cận thiết bị. Để đạt được điều

này, hệ thống cần hỗ trợ các tính năng như kiểm tra kết nối trước khi thi, giám sát từ xa thông qua webcam hoặc AI để phát hiện hành vi bất thường, và cung cấp các phương án hỗ trợ cho người dùng gặp vấn đề kỹ thuật trong quá trình làm bài.

Phản hồi nhanh chóng và chính xác: Một trong những ưu điểm nổi bật của hệ thống thi trắc nghiệm trực tuyến là khả năng cung cấp phản hồi ngay lập tức. Sau khi hoàn thành bài thi, thí sinh có thể nhận được kết quả ngay lập tức, cùng với đánh giá chi tiết về từng câu hỏi, điểm mạnh và điểm yếu của mình. Điều này không chỉ giúp người học cải thiện hiệu suất mà còn tiết kiệm thời gian cho giảng viên trong việc chấm điểm và phân tích kết quả.

Hệ thống thi trắc nghiệm trực tuyến là giải pháp tiên tiến, mang lại nhiều lợi ích vượt trội so với phương pháp kiểm tra truyền thống. Với các yếu tố như độ tin cậy, độ hợp lý, tính công bằng, và khả năng phản hồi nhanh chóng, hệ thống không chỉ nâng cao hiệu quả dạy và học mà còn góp phần xây dựng một môi trường giáo dục minh bạch, công bằng và hiện đại hơn. Để đạt được điều này, việc thiết kế và vận hành hệ thống cần được thực hiện một cách cẩn thận, dựa trên các nguyên tắc và lý thuyết kiểm tra giáo dục, tận dụng tối đa sức mạnh của công nghệ [8] .

2.2.3 Lý thuyết về thiết kế giao diện người dùng (UI)

Giao diện người dùng là yếu tố quan trọng trong việc xây dựng các ứng dụng trực tuyến, đặc biệt là đối với các hệ thống thi trắc nghiệm, nơi người dùng cần dễ dàng tương tác với hệ thống. Lý thuyết về nguyên lý thiết kế giao diện người dùng bao gồm việc tạo ra những giao diện dễ sử dụng, trực quan, và có thể dễ dàng điều hướng. Các nguyên lý thiết kế như tính nhất quán (Consistency), đơn giản hóa (Simplicity), và phản hồi người dùng (User Feedback) sẽ được áp dụng trong việc xây dựng giao diện cho hệ thống thi trắc nghiệm trực tuyến [9] .

2.2.4 Lý thuyết về cơ sở dữ liệu NoSQL

Trong việc thiết kế hệ thống thi trực tuyến, cơ sở dữ liệu là yếu tố quan trọng để lưu trữ và quản lý dữ liệu người dùng, đề thi, kết quả thi và các thông tin liên quan. Cơ sở dữ liệu NoSQL (như MongoDB) giúp giải quyết vấn đề mở rộng và linh hoạt trong việc lưu trữ dữ liệu không có cấu trúc cố định. Các lý thuyết về NoSQL chỉ ra rằng việc sử dụng MongoDB giúp hệ thống dễ dàng

quản lý dữ liệu phi cấu trúc và có thể mở rộng một cách hiệu quả khi lượng dữ liệu và số lượng người dùng gia tăng [10] .

2.2.5 Lý thuyết về tối ưu hóa hiệu suất hệ thống

Hệ thống thi trực tuyến phải đảm bảo khả năng xử lý hiệu quả khi có lượng lớn người dùng truy cập đồng thời, đặc biệt trong các kỳ thi có quy mô lớn với hàng nghìn hoặc thậm chí hàng chục nghìn thí sinh tham gia cùng một thời điểm. Việc đảm bảo hiệu suất cao không chỉ giúp hệ thống hoạt động ổn định mà còn tăng trải nghiệm người dùng và giảm thiểu rủi ro gián đoạn trong quá trình thi.

Các thách thức về hiệu suất trong hệ thống thi trực tuyến

Khả năng xử lý đồng thời (Concurrency): Trong các kỳ thi trực tuyến, có thể xảy ra tình trạng nhiều người dùng gửi yêu cầu đồng thời, chẳng hạn như tải câu hỏi, gửi câu trả lời, hoặc lưu tạm thời bài làm. Điều này đặt ra thách thức lớn cho hệ thống phải xử lý mọi yêu cầu nhanh chóng và không để xảy ra tình trạng nghẽn cổ chai (bottleneck).

Giảm thiểu độ trễ (Latency): Độ trễ thấp là yếu tố quan trọng trong một kỳ thi trực tuyến. Thời gian phản hồi chậm, như việc tải câu hỏi hoặc nộp bài thi, có thể gây ảnh hưởng tiêu cực đến trải nghiệm của thí sinh và làm mất sự tin cậy đối với hệ thống.

Khả năng mở rộng (Scalability): Hệ thống phải có khả năng mở rộng linh hoạt để đáp ứng số lượng người dùng tăng đột biến. Điều này đặc biệt quan trọng trong các kỳ thi quy mô lớn hoặc trong trường hợp nhiều trường học hoặc tổ chức cùng sử dụng hệ thống tại một thời điểm.

Độ tin cậy (Reliability): Hệ thống cần đảm bảo hoạt động liên tục, không bị gián đoạn ngay cả trong những tình huống tải nặng hoặc gặp sự cố bất ngờ. Điều này đòi hỏi phải có các biện pháp sao lưu và phục hồi dữ liệu nhanh chóng.

Ứng dụng lý thuyết tối ưu hóa hiệu suất hệ thống

Lý thuyết tối ưu hóa hiệu suất hệ thống chỉ ra rằng việc sử dụng các công nghệ phù hợp là chìa khóa để đạt được mục tiêu xử lý hiệu quả và đảm bảo khả năng mở rộng. Trong bối cảnh này, **Node.js** nổi bật như một lựa chọn lý tưởng cho

các hệ thống thi trực tuyến nhờ khả năng xử lý bất đồng bộ mạnh mẽ và khả năng hoạt động tốt trong môi trường đa người dùng.

1. Node.js và mô hình xử lý bất đồng bộ Node.js hoạt động dựa trên một vòng lặp sự kiện đơn luồng (single-threaded event loop) và cơ chế xử lý bất đồng bộ (asynchronous I/O). Điều này có nghĩa là: Khi một yêu cầu được gửi đến hệ thống, Node.js không cần phải chờ hoàn thành yêu cầu đó trước khi tiếp tục xử lý các yêu cầu khác. Thay vào đó, nó sử dụng các callback hoặc promise để thực hiện các tác vụ khi dữ liệu sẵn sàng. Mô hình này giúp giảm thiểu tối đa tình trạng tắc nghẽn (blocking) và cho phép xử lý hàng nghìn kết nối đồng thời mà không cần tăng đáng kể tài nguyên phần cứng. Ví dụ, khi một thí sinh gửi câu trả lời, Node.js có thể lưu trữ câu trả lời đó vào cơ sở dữ liệu mà không làm gián đoạn việc phục vụ các thí sinh khác đang tải câu hỏi.

2. Khả năng mở rộng với các kiến trúc microservices Node.js hỗ trợ tốt trong việc xây dựng các hệ thống theo kiến trúc microservices, nơi các thành phần của hệ thống (như xử lý câu hỏi, lưu trữ bài làm, quản lý người dùng) được tách rời và có thể triển khai độc lập. Kiến trúc này có nhiều lợi ích: **Mở rộng linh hoạt:** Các thành phần riêng lẻ có thể được mở rộng theo nhu cầu, ví dụ, tăng số lượng máy chủ xử lý câu hỏi trong giờ cao điểm mà không ảnh hưởng đến các thành phần khác. **Giảm tải:** Tách biệt các dịch vụ giúp giảm tải áp lực lên từng phần của hệ thống, cải thiện hiệu suất tổng thể.

3. Lưu trữ tạm thời và bộ nhớ đệm (Caching) Node.js kết hợp với các công nghệ bộ nhớ đệm như Redis hoặc Memcached có thể giảm thiểu thời gian phản hồi trong các tình huống tải cao. Bộ nhớ đệm có thể được sử dụng để lưu trữ tạm thời: Câu hỏi thi đã được tạo sẵn. Thông tin cấu hình bài thi. Trạng thái hiện tại của bài làm. Việc sử dụng bộ nhớ đệm giúp giảm số lượng truy vấn trực tiếp đến cơ sở dữ liệu, từ đó giảm độ trễ và tăng tốc độ xử lý.

4. Cân bằng tải (Load Balancing) Hệ thống thi trực tuyến cần triển khai các phương pháp cân bằng tải để phân phối yêu cầu đều giữa các máy chủ. Node.js tích hợp tốt với các công cụ như Nginx hoặc HAProxy để thực hiện điều này, giúp giảm thiểu nguy cơ một máy chủ bị quá tải và gây ra tình trạng gián đoạn.

5. Quản lý cơ sở dữ liệu hiệu quả Node.js có thể làm việc hiệu quả với các hệ quản trị cơ sở dữ liệu như MongoDB, PostgreSQL, hoặc MySQL. Đối với hệ thống thi trực tuyến:

Cơ sở dữ liệu NoSQL (như MongoDB): Phù hợp để lưu trữ dữ liệu phi cấu trúc, như các bài thi hoặc câu hỏi có định dạng đa dạng.

Cơ sở dữ liệu quan hệ (như PostgreSQL): Phù hợp để lưu trữ dữ liệu liên quan đến người dùng, điểm số, hoặc lịch sử bài thi.

Việc tối ưu hóa các truy vấn cơ sở dữ liệu và sử dụng các cơ chế chỉ mục (indexing) giúp đảm bảo rằng dữ liệu được truy xuất nhanh chóng ngay cả khi số lượng thí sinh và bài thi tăng cao.

Hệ thống thi trực tuyến phải đảm bảo hiệu suất tốt khi xử lý nhiều người dùng cùng lúc. Lý thuyết về tối ưu hóa hiệu suất hệ thống chỉ ra rằng việc sử dụng các công nghệ như Node.js giúp tối ưu hóa việc xử lý đồng thời, giảm thiểu độ trễ và đảm bảo hệ thống có thể mở rộng mà không làm giảm hiệu suất [11] .

2.3 Phương pháp nghiên cứu

2.3.1 Nghiên cứu lý thuyết

Nghiên cứu các tài liệu lý thuyết về các công nghệ phần mềm, cơ sở dữ liệu NoSQL, thiết kế hệ thống thi trắc nghiệm trực tuyến và các phương pháp phát triển phần mềm giúp tạo nền tảng kiến thức vững chắc để xây dựng hệ thống.

2.3.2 Phân tích yêu cầu và thiết kế hệ thống

Phương pháp này bao gồm việc thu thập yêu cầu từ người dùng và các bên liên quan để xác định tính năng và chức năng cần có của hệ thống. Sau đó, các yêu cầu này sẽ được chuyển thành mô hình thiết kế hệ thống chi tiết, bao gồm thiết kế giao diện người dùng (UI) và cấu trúc cơ sở dữ liệu.

2.3.3 Phát triển phần mềm

Sử dụng các công nghệ hiện đại như React cho giao diện người dùng, Node.js cho xử lý phía server và MongoDB cho lưu trữ dữ liệu. Phương pháp phát triển phần mềm này tập trung vào việc xây dựng hệ thống qua từng bước nhỏ và dễ dàng kiểm tra, bảo trì.

2.3.4 Mô phỏng và thử nghiệm

Sau khi phát triển xong các tính năng chính, hệ thống sẽ được thử nghiệm với các kịch bản thực tế để đảm bảo tính chính xác và hiệu quả. Các bài kiểm tra này sẽ giúp phát hiện các lỗi, xác minh độ tin cậy của hệ thống và đánh giá hiệu suất khi có nhiều người dùng truy cập đồng thời.

2.3.5 Phân tích dữ liệu và đánh giá hiệu suất

Sau khi triển khai, hệ thống sẽ được đánh giá qua các chỉ số như tốc độ phản hồi, khả năng xử lý đồng thời, độ chính xác trong việc chấm điểm và bảo mật thông tin.

2.3.6 Kiểm thử phần mềm

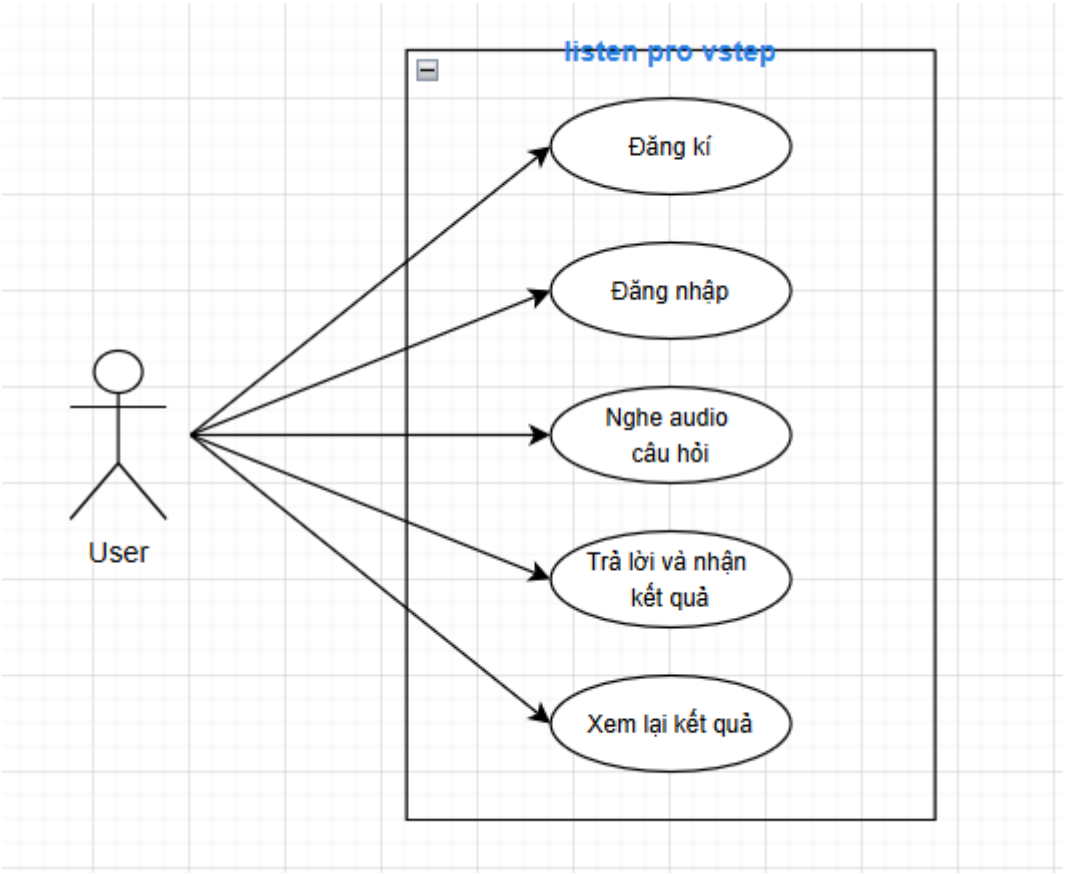
Trong quá trình phát triển và triển khai, hệ thống sẽ được kiểm thử qua các phương pháp như kiểm thử đơn vị (unit testing), kiểm thử tích hợp (integration testing), và kiểm thử hệ thống (system testing). Điều này đảm bảo rằng hệ thống hoạt động chính xác và không có lỗi.

CHƯƠNG 3: HIỆN THỰC HÓA NGHIÊN CỨU

3.1 Phân tích nhu cầu

3.1.1 Yêu cầu chức năng

Người dùng: Đăng ký, đăng nhập. Nghe audio câu hỏi, trả lời và nhận kết quả chấm thi tự động. Xem lại kết quả thi.



Hình 1: Biểu đồ use case

3.1.2 Yêu cầu phi chức năng

Hiệu năng: Hệ thống có khả năng xử lý mượt mà ngay cả khi nhiều người dùng truy cập cùng lúc.

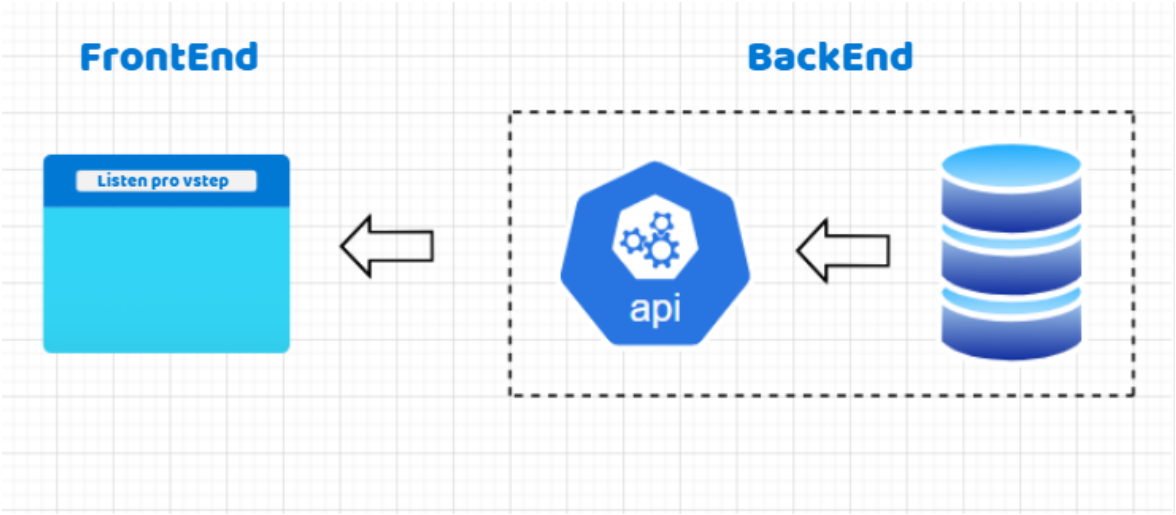
Bảo mật: Dữ liệu người dùng, đề thi, và kết quả thi phải được bảo vệ an toàn.

Khả năng mở rộng: Hệ thống có thể bổ sung tính năng mới dễ dàng khi nhu cầu thay đổi.

3.2 Thiết kế hệ thống

3.2.1 Kiến trúc hệ thống

Hệ thống được xây dựng dựa trên kiến trúc ba tầng truyền thống nhằm đảm bảo tính linh hoạt, dễ bảo trì và mở rộng:



Hình 2. Kiến trúc hệ thống

1. Tầng giao diện (Frontend):

Công nghệ sử dụng: Sử dụng React.js để phát triển giao diện người dùng. Đây là một thư viện JavaScript mạnh mẽ, cung cấp khả năng xây dựng giao diện động và hiệu quả.

Cấu trúc giao diện: Tích hợp React Router để điều hướng linh hoạt giữa các trang, tạo trải nghiệm mượt mà cho người dùng mà không cần tải lại toàn bộ trang. Thiết kế giao diện responsive, đảm bảo hoạt động tốt trên cả thiết bị di động, máy tính bảng và máy tính để bàn.

Các trang giao diện chính:

Trang đăng nhập: Cho phép người dùng nhập tài khoản, mật khẩu để truy cập hệ thống.

Trang làm bài thi: Cung cấp giao diện trực quan để người dùng làm bài, bao gồm khu vực phát Audio, lựa chọn câu trả lời, và nút nộp bài.

Trang kết quả: Hiển thị kết quả bài thi kèm theo các đánh giá và lời nhận xét.

2. Tầng xử lý (Backend):

Công nghệ sử dụng: Sử dụng Node.js với Express.js để xây dựng các API RESTful, đảm bảo khả năng xử lý nhanh và mở rộng dễ dàng.

Các chức năng chính: Xử lý các yêu cầu từ phía frontend, bao gồm việc truy xuất dữ liệu, xác thực người dùng và trả về kết quả bài thi. Cung cấp tính năng chấm điểm tự động dựa trên đáp án đã lưu. Hỗ trợ phân quyền, đảm bảo chỉ người dùng hợp lệ mới có thể truy cập các tài nguyên.

3. Tầng dữ liệu (Database):

Công nghệ sử dụng: MongoDB được lựa chọn làm cơ sở dữ liệu chính, nhờ vào tính năng linh hoạt và khả năng lưu trữ dữ liệu không cấu trúc.

Cấu trúc dữ liệu:Cơ sở dữ liệu được chia thành các collections chính:

Users: Lưu trữ thông tin tài khoản người dùng, bao gồm vai trò (người thi hoặc quản trị viên), thông tin cá nhân và lịch sử bài thi.

Questions: Lưu trữ câu hỏi, tệp Audio liên quan, và các thông tin cấu hình cho bài thi.

Answers: Lưu trữ đáp án đúng để phục vụ cho chức năng chấm điểm tự động.

Exams: Thông tin về các bài thi, bao gồm danh sách câu hỏi, thời gian thi và trạng thái.

Results: Lưu trữ kết quả bài thi, bao gồm điểm số, đánh giá và thời gian làm bài của từng người dùng.

3.2.2 Chức năng hệ thống

Hệ thống được thiết kế để đảm bảo các chức năng cốt lõi hoạt động hiệu quả, tạo trải nghiệm người dùng tốt nhất. Các chức năng chính bao gồm:

1. Quản lý người dùng:

- Hỗ trợ đăng ký và đăng nhập, với tính năng xác thực tài khoản để bảo mật thông tin người dùng.

- Hỗ trợ phân quyền giữa người thi và quản trị viên.

2. Làm bài thi:

- Tích hợp trình phát Audio cho từng câu hỏi, đảm bảo chất lượng âm thanh rõ ràng và mượt mà.

- Cung cấp giao diện thân thiện để chọn đáp án cho từng câu hỏi.

- Hỗ trợ tính năng lưu tiến độ làm bài, cho phép tiếp tục làm bài nếu người dùng bị gián đoạn.

3. Tự động chấm điểm:

- Sau khi nộp bài thi, hệ thống tự động tính điểm dựa trên đáp án đã lưu trữ.

- Tích hợp thuật toán chấm điểm với khả năng đánh giá cả câu trả lời đúng một phần.

4. Hiển thị kết quả:

- Hiển thị điểm số trực quan dưới dạng biểu đồ hoặc bảng thống kê.

- Cung cấp nhận xét tổng quan về kết quả bài thi, cùng với gợi ý cải thiện.

- Lưu trữ kết quả để người dùng có thể xem lại bất kỳ lúc nào.

5. Hỗ trợ hướng dẫn:

- Trang hướng dẫn được thiết kế chi tiết, giải thích các bước làm bài và quy định thi.

- Cung cấp các mẹo giúp người dùng tối ưu hóa thời gian làm bài.

3.3 Phát triển hệ thống

3.2.1 Công nghệ sử dụng

Frontend: React.js cung cấp khả năng xây dựng các thành phần giao diện tái sử dụng, hỗ trợ quản lý trạng thái ứng dụng mượt mà và cải thiện hiệu suất tổng thể.

Backend: Node.js với framework Express.js. Node.js hỗ trợ xử lý đồng thời nhiều yêu cầu nhờ vào kiến trúc event-driven, trong khi Express.js giúp đơn giản hóa việc tạo API RESTful, giúp việc giao tiếp giữa frontend và backend trở nên dễ dàng hơn.

Database: NoSQL (MongoDB Atlas) để quản lý dữ liệu trên nền tảng cloud. MongoDB cho phép lưu trữ dữ liệu dưới dạng tài liệu JSON-like, phù hợp với các ứng dụng yêu cầu linh hoạt và dễ mở rộng.

3.3.2 Quy trình phát triển

Bước 1: Phân tích yêu cầu, xây dựng tài liệu thiết kế. Trong bước này, các yêu cầu của người dùng được thu thập và phân tích kỹ lưỡng để xác định các chức năng cần có của hệ thống. Từ đó, tài liệu thiết kế chi tiết sẽ được xây dựng để định hướng cho các giai đoạn tiếp theo.

Bước 2: Phát triển giao diện người dùng. Sử dụng React.js và các thư viện hỗ trợ để tạo ra giao diện trực quan, dễ sử dụng và tương tác tốt trên nhiều thiết bị.

Bước 3: Xây dựng API xử lý logic nghiệp vụ. Các API RESTful sẽ được phát triển trên nền tảng Node.js và Express.js để phục vụ các yêu cầu từ frontend, như xử lý đăng nhập, chấm điểm và lưu trữ dữ liệu.

Bước 4: Tích hợp frontend với backend qua API. Kết nối hai thành phần này thông qua các endpoint được định nghĩa rõ ràng, đảm bảo dữ liệu luân chuyển mượt mà giữa giao diện và máy chủ.

Bước 5: Thử nghiệm và kiểm tra hệ thống. Hệ thống được kiểm tra toàn diện để đảm bảo tính ổn định và đáp ứng đúng yêu cầu.

3.4 Kết quả thực hiện

Hệ thống đã được triển khai và thử nghiệm thành công với các tính năng cơ bản:

- Đăng nhập, đăng ký hoạt động ổn định với giao diện trực quan, dễ dàng sử dụng cho cả người dùng mới và quen thuộc. Quá trình này được tích hợp với cơ chế bảo mật mạnh mẽ, bao gồm mã hóa mật khẩu và xác thực thông qua token.

- Làm bài thi với audio được phát tự động, câu hỏi hiển thị đúng theo yêu cầu từng bước và thời gian làm bài được giám sát chặt chẽ. Người dùng có thể điều chỉnh âm lượng hoặc tua lại audio trong khoảng thời gian cho phép để tăng hiệu quả làm bài.

- Hệ thống tự động chấm điểm dựa trên việc so sánh câu trả lời của người dùng với đáp án đã được lưu trữ, sau đó tính toán và trả về điểm số một cách nhanh chóng. Quá trình này không chỉ đảm bảo tính chính xác mà còn giúp tiết kiệm thời gian chấm thi, đặc biệt trong các kỳ thi với số lượng lớn thí sinh. Kết quả hiển thị chi tiết, bao gồm cả điểm tổng và các phần điểm từng câu hỏi, giúp người dùng dễ dàng đánh giá hiệu suất làm bài thi của mình.

- Giao diện người dùng được thiết kế thân thiện và dễ sử dụng, phù hợp với nhiều đối tượng người dùng khác nhau, từ học sinh đến giáo viên. Hệ thống hỗ trợ tốt trên nhiều thiết bị, bao gồm máy tính, máy tính bảng, và điện thoại thông minh, đảm bảo trải nghiệm nhất quán và tiện lợi cho người dùng ở mọi nơi, mọi lúc.

CHƯƠNG 4: KẾT QUẢ NGHIÊN CỨU

4.1 Thiết kế giao diện

Giao diện chính bao gồm:

1. **Trang đăng nhập:**

- Giao diện đăng nhập được thiết kế đơn giản nhưng chuyên nghiệp, đảm bảo người dùng dễ dàng truy cập vào hệ thống.

- Bao gồm form nhập thông tin tài khoản với hai trường chính: tên đăng nhập (hoặc email) và mật khẩu.

- Hỗ trợ thông báo lỗi nếu tài khoản hoặc mật khẩu không hợp lệ, đi kèm gợi ý để người dùng kiểm tra lại thông tin.

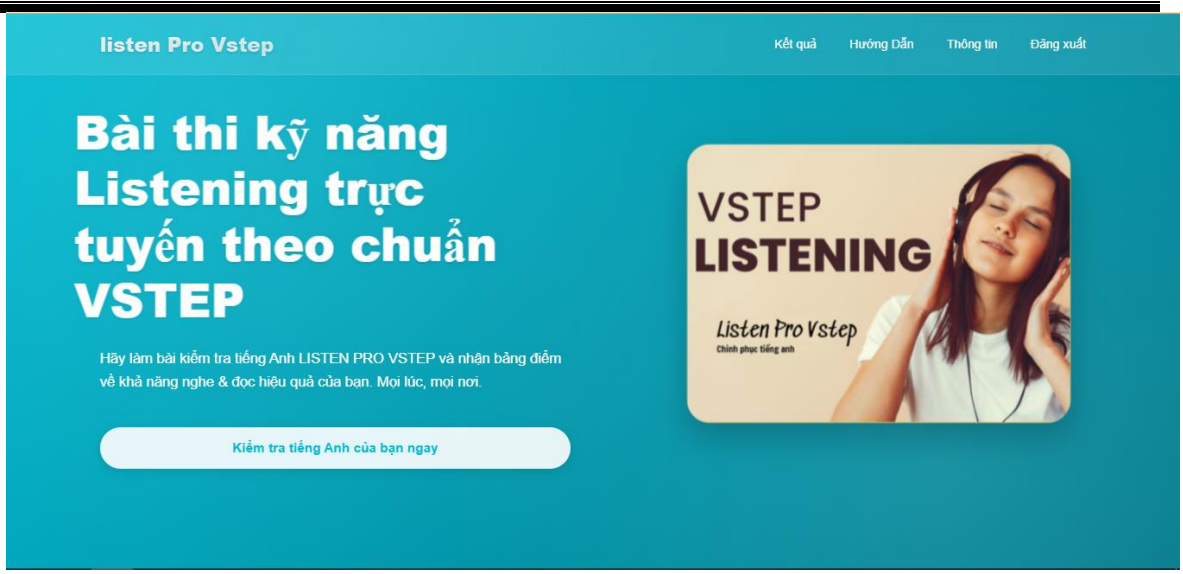
Hình 3: Trang đăng nhập và đăng ký

2. **Trang chính:**

- Đây là giao diện đầu tiên mà người dùng nhìn thấy sau khi đăng nhập thành công.

- Hiển thị danh sách các bài thi mà người dùng có thể tham gia, được sắp xếp theo danh mục hoặc thứ tự ưu tiên.

- Bao gồm thanh điều hướng chính, giúp người dùng dễ dàng truy cập các mục như: Làm bài thi, Xem kết quả, hoặc Hướng dẫn.



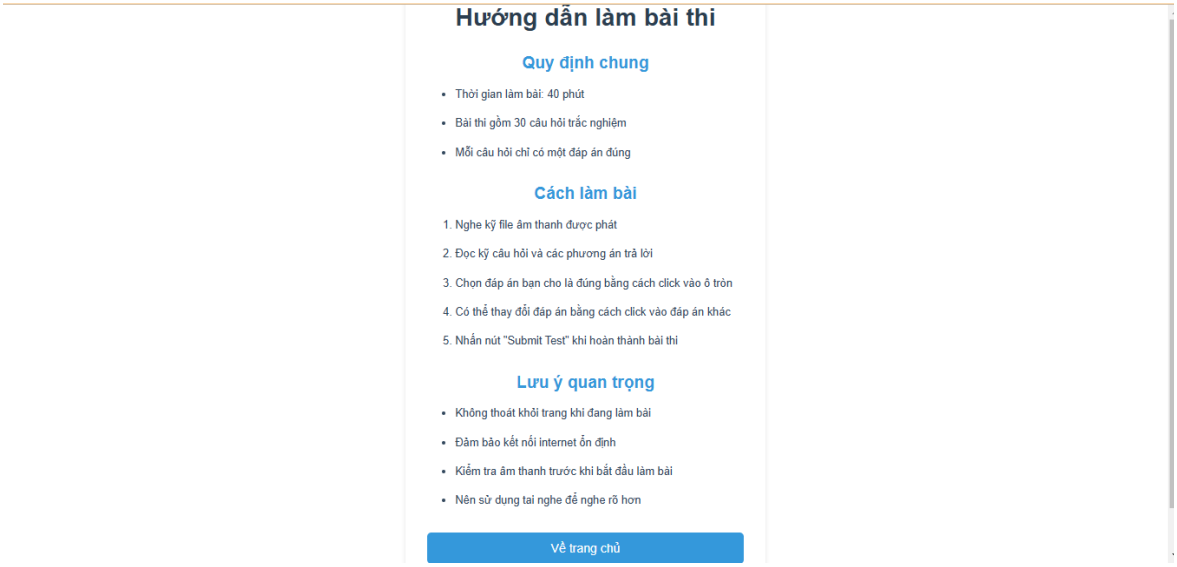
Hình 4: Trang giao diện chính

3. Trang hướng dẫn:

- Trang này cung cấp thông tin chi tiết về quy trình thi và các quy định cần tuân thủ.

- Hỗ trợ phần câu hỏi thường gặp (FAQ) để giải đáp các thắc mắc phổ biến.

- Thiết kế giao diện trực quan, dễ theo dõi, với các biểu tượng minh họa sinh động.



Hình 5: Trang hướng dẫn làm bài thi

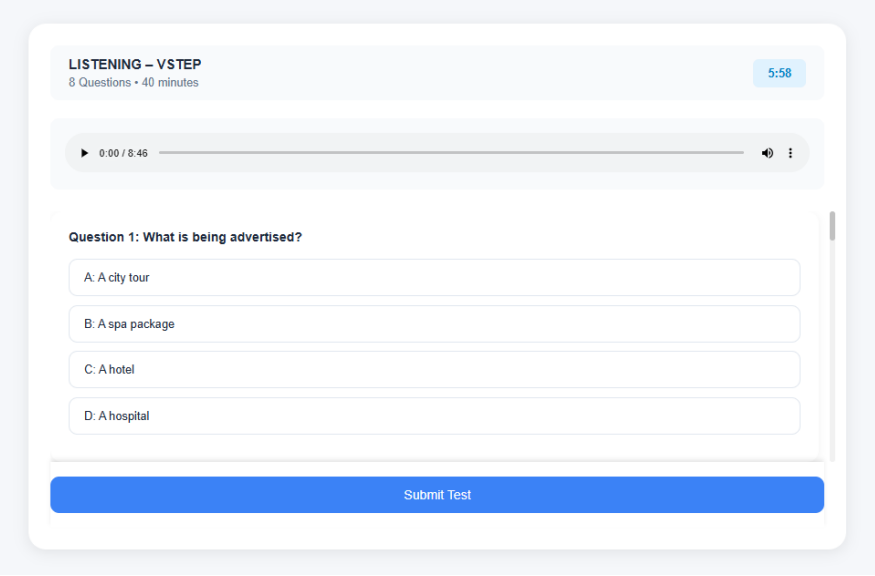
4. Trang làm bài thi:

- Giao diện làm bài thi được thiết kế để tối ưu hóa trải nghiệm của người dùng, bao gồm:

Khu vực phát Audio: Một trình phát âm thanh được tích hợp, cho phép người dùng phát, tạm dừng hoặc tua lại trong một giới hạn cho phép. Hiển thị tiến trình nghe (thanh tiến trình) để người dùng biết được thời lượng còn lại của file Audio.

Khu vực chọn câu trả lời: Hiển thị danh sách các câu hỏi cùng các tùy chọn trả lời dưới dạng radio buttons hoặc checkbox (tùy thuộc vào loại câu hỏi). Thiết kế thân thiện với người dùng, đảm bảo kích thước nút và khoảng cách đủ lớn để dễ dàng thao tác trên thiết bị di động.

Khu vực submit: Cung cấp nút "Nộp bài" rõ ràng, yêu cầu xác nhận trước khi hoàn tất. Hiển thị thời gian làm bài còn lại dưới dạng đồng hồ đếm ngược, giúp người dùng quản lý thời gian hiệu quả. Ngoài ra, hệ thống sẽ tự động lưu bài làm trong trường hợp bị mất kết nối hoặc người dùng rời khỏi trang.



Hình 6: Trang làm bài thi

5. Trang kết quả:

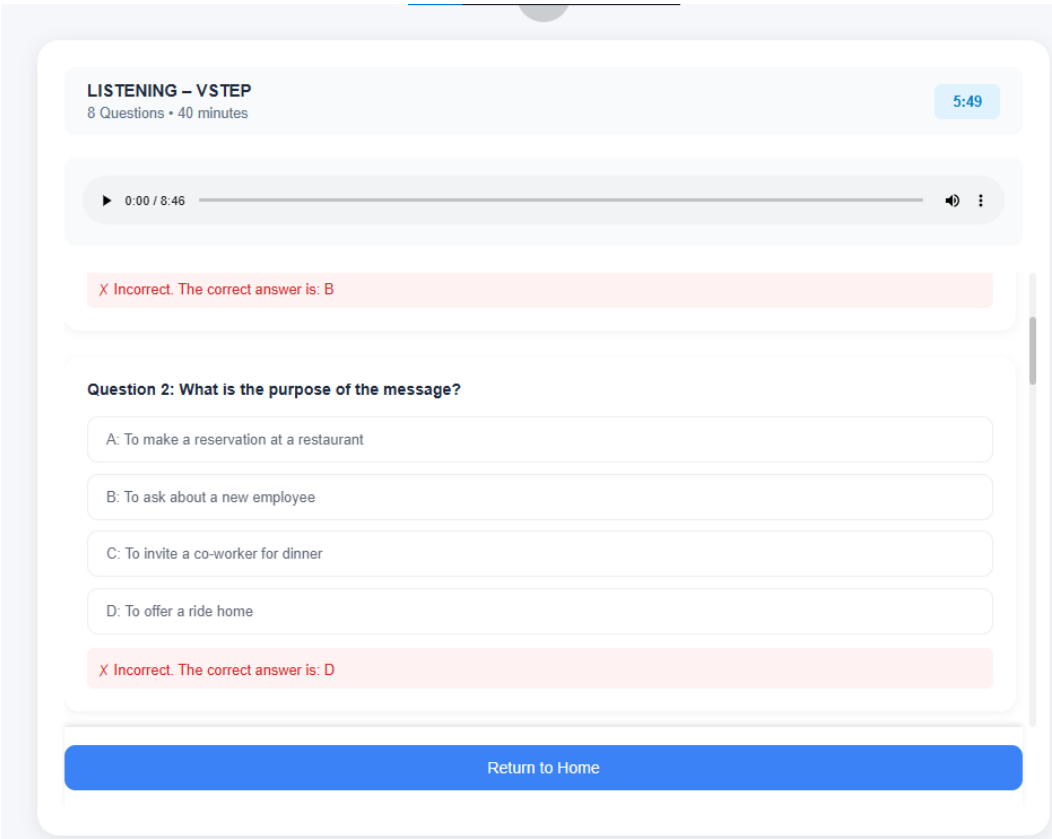
- Giao diện hiển thị kết quả được thiết kế bắt mắt và chi tiết, bao gồm:

Điểm số: Hiển thị tổng điểm dưới dạng số và biểu đồ trực quan (ví dụ: biểu đồ tròn hoặc thanh).

Đánh giá: Cung cấp nhận xét tổng quan về kết quả bài thi, ví dụ: "Tốt," "Khá," hoặc "Cần cải thiện."

Chi tiết câu trả lời: Hiển thị danh sách câu hỏi, câu trả lời của người dùng và đáp án đúng để người dùng tự đánh giá.

So sánh hiệu suất: Nếu người dùng đã thi nhiều lần, kết quả các lần trước cũng được hiển thị để so sánh tiến độ.



Hình 7. Kết quả thi

4.2 Kế quả đạt được

Sau quá trình thực hiện đồ án, Xây dựng hệ thống thi trắc nghiệm trực tuyến kỹ năng Listening theo chuẩn Vstep với cơ sở dữ liệu NoSQL đã được hoàn thiện với đầy đủ các chức năng, bao gồm đăng ký, đăng nhập, làm bài thi trắc nghiệm, và xem kết quả. Hệ thống có khả năng xử lý nhiều người dùng đồng thời với hiệu suất ổn định, tốc độ phản hồi nhanh và khả năng mở rộng cao, nhờ vào việc sử dụng công nghệ Node.js và MongoDB. Giao diện người dùng được thiết kế đơn giản và thân thiện, dễ sử dụng, hỗ trợ tốt trên nhiều thiết bị và giúp người dùng dễ dàng thao tác trong suốt quá trình làm bài thi. Hệ thống cũng cung

cấp các tính năng quản lý bài thi và người dùng dành cho giảng viên, giúp giảng viên tạo và quản lý đề thi, cũng như chấm điểm tự động. Các giao diện chức năng như trang đăng nhập, làm bài thi, và quản lý bài thi đều được thiết kế rõ ràng và dễ sử dụng. Hệ thống có thể mở rộng thêm tính năng trong tương lai như hỗ trợ các loại bài thi khác nhau hoặc tích hợp công cụ phân tích kết quả học tập.

CHƯƠNG 5: KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

Kết luận: Sau khi hoàn thành đồ án, hệ thống thi trắc nghiệm trực tuyến đã được xây dựng hoàn chỉnh với các chức năng chính như đăng nhập, làm bài thi, chấm điểm tự động và quản lý kết quả. Hệ thống đã đạt được hiệu suất ổn định, khả năng mở rộng cao và bảo mật tốt. Giao diện người dùng thân thiện, dễ sử dụng và hỗ trợ trên nhiều thiết bị. Các kết quả đạt được đáp ứng được yêu cầu về tính năng và hiệu suất, đồng thời mang lại trải nghiệm người dùng tốt. Hệ thống có thể phát triển thêm các tính năng mới như hỗ trợ nhiều loại bài thi khác nhau và tích hợp công cụ phân tích kết quả học tập.

Hướng phát triển: Các nghiên cứu tiếp theo có thể tập trung vào việc cải thiện khả năng chấm điểm tự động cho các câu hỏi tự luận bằng cách sử dụng công nghệ học máy. Ngoài ra, có thể phát triển thêm các công cụ phân tích kết quả học tập, tạo ra hệ thống đánh giá tiến trình học của người dùng và hỗ trợ các hình thức thi phức tạp hơn.

DANH MỤC TÀI LIỆU THAM KHẢO

[1]. Agile Manifesto**:** https://agilemanifesto.org/.

[2]. Best Practices for Online Exams***:*** *https://www.classmarker.com/*

[3]. React Documentation*: https://reactjs.org/*

[4]. Website chính thức: *https://www.w3schools.com/*

[5]. Website chính thức: https://www.mongodb.com

[6]. Website chính thức: *https://martinfowler.com/architecture/.*

[7]. Scrum Guide*: **https://scrumguides.org/***

[8]. Educational Assessment Resources***:*** *https://www.ets.org/*

[9]. UX/UI Best Practices***:*** *https://uxdesign.cc/*

[10]. NoSQL Tutorial***:*** *https://www.tutorialspoint.com/nosql/index.htm*

[11]. System Performance Analysis***:*** *https://highscalability.com/*

[12]. Cao Lê Viết Tiến. (n.d.). NoSQL là gì? Cơ sở dữ liệu NoSQL là gì?. Vietnix. Truy cập từ https://vietnix.vn/nosql-la-gi/#co-so-du-lieu-nosql-la-gi

[13]. Meier, A., & Kaufmann, M. (2019). *SQL & NoSQL Databases*. Springer Fachmedien Wiesbaden: Springer Vieweg.

[14]. Celko, J. (2014). *Joe Celko's Complete Guide to NoSQL*. Amsterdam: Elsevier/Morgan Kaufmann.

[15]. Sarrion, E. (2022). *Web Development with JavaScript: Building Web Applications*. Birmingham, UK: Packt Publishing Ltd.

[16]. Bradshaw, S. (2020). *Database Management with Open Source Software: Object-Oriented Databases*. Beijing: O'Reilly Media, Inc.

PHỤ LỤC