TRƯỜNG ĐẠI HỌC TÀI NGUYÊN VÀ MÔI TRƯỜNG HÀ NỘI

**KHOA CÔNG NGHỆ THÔNG TIN**

****

**PHÁT TRIỂN ỨNG DỤNG THI TRẮC NGHIỆM TRỰC TUYẾN VỚI REACTJS**

**NGUYỄN HUY TÙNG**

**Hà Nội – Năm 2024**

TRƯỜNG ĐẠI HỌC TÀI NGUYÊN VÀ MÔI TRƯỜNG HÀ NỘI

**KHOA CÔNG NGHỆ THÔNG TIN**

****

**PHÁT TRIỂN ỨNG DỤNG THI TRẮC NGHIỆM TRỰC TUYẾN VỚI REACTJS**

|  |  |
| --- | --- |
| **Họ và tên Sinh viên** | **: Nguyễn Huy Tùng** |
| **Ngành đào tạo** | **: Công nghệ thông tin** |

**NGƯỜI HƯỚNG DẪN: ThS.VŨ NGỌC PHAN**

**Hà Nội – Năm 2024**

**CỘNG HÒA XÃ HỘI CHỦ NGHĨA VIỆT NAM**

**Độc lập – Tự do – Hạnh phúc**

# BẢN CAM ĐOAN

Tên tôi là: Nguyễn Huy Tùng

Mã sinh viên: 20111064266 Lớp: ĐH10C6

Ngành: Công nghệ thông tin

Tôi đã thực hiện khóa luận tốt nghiệp với đề tài: Phát triển ứng dụng thi trắc nghiệm trực tuyến với ReactJS.

Tôi xin cam đoan đây là đề tài nghiên cứu của riêng tôi và được sự hướng dẫn của ThS.Vũ Ngọc Phan.

Các nội dung nghiên cứu, kết quả trong đề tài này là trung thực và chưa được công bố dưới bất kỳ hình thức nào. Nếu phát hiện có bất kỳ hình thức gian lận nào tôi xin hoàn toàn chịu trách nhiệm trước pháp luật.

|  |  |  |
| --- | --- | --- |
|  |  | Hà Nội, ngày 20 tháng 03 năm 2024 |
| **Cán bộ hướng dẫn** |  | **Sinh viên** |
|  |  |  |
| ThS.Vũ Ngọc Phan |  | Nguyễn Huy Tùng |

# LỜI CẢM ƠN

Để đạt được kết quả như ngày hôm nay, bên cạnh việc bản thân nỗ lực phấn đấu trau dồi kiến thức thì không thể thiếu được sự quan tâm, chỉ bảo tận tình và những lời động viên chân thành và quý báu từ phía Thầy Cô, gia đình, bạn bè.

Em xin chân thành cảm ơn Thầy Vũ Ngọc Phan, là người đã tận tình hướng dẫn em, giúp em giải quyết các vấn đề, chỉ bảo em trong suốt thời gian thực hiện đề tài.

Em cũng xin gửi lời cảm ơn sâu sắc đến quý Thầy Cô trong Khoa Công nghệ Thông tin Trường Đại học Tài Nguyên và Môi Trường Hà Nội đã tận tình giảng dạy, trang bị cho em những kiến thức quý báu trong những năm học vừa qua.

Vì kiến thức của bản thân còn hạn chế, trong quá trình làm khóa luận, nội dung không tránh khỏi những thiếu sót. Kính mong Thầy Cô, động nghiệp, bạn bè góp ý để khóa luận tốt nghiệp được hoàn thiện tốt đẹp hơn.

|  |  |
| --- | --- |
|  | Hà Nội, ngày 20 tháng 03 năm 2024  Sinh viên thực hiện  Nguyễn Huy Tùng |

# MỤC LỤC

[BẢN CAM ĐOAN i](#_Toc168134665)

[LỜI CẢM ƠN ii](#_Toc168134666)

[MỤC LỤC iii](#_Toc168134667)

[DANH MỤC CÁC CHỮ VIẾT TẮT v](#_Toc168134668)

[DANH MỤC CÁC BẢNG vi](#_Toc168134669)

[DANH MỤC HÌNH ẢNH vii](#_Toc168134670)

[MỞ ĐẦU 1](#_Toc168134671)

[CHƯƠNG 1: TỔNG QUAN CÁC VẤN ĐỀ CẦN NGHIÊN CỨU 2](#_Toc168134672)

[1.1. Lý do chọn đề tài 2](#_Toc168134673)

[1.2. Mục tiêu nghiên cứu 3](#_Toc168134674)

[1.3. Nội dung nghiên cứu 2](#_Toc168134675)

[1.4. Phương pháp nghiên cứu 4](#_Toc168134676)

[1.5. Các công nghệ sử dụng 5](#_Toc168134677)

[1.5.1. Ngôn ngữ lập trình JavaScript 5](#_Toc168134678)

[1.5.2. Ngôn ngữ HTML và CSS 7](#_Toc168134679)

[1.5.3. ReactJS 8](#_Toc168134680)

[1.5.4. NodeJS 10](#_Toc168134681)

[1.5.5. Express 12](#_Toc168134682)

[1.5.6. Mongodb 14](#_Toc168134683)

[1.5.7. Mô hình Web API. 17](#_Toc168134684)

[1.6. Một số công cụ hỗ trợ khác 19](#_Toc168134685)

[1.6.1. Microsoft Visual Studio Code 19](#_Toc168134686)

[1.6.2. Postman 20](#_Toc168134687)

[1.6.3. StarUML 22](#_Toc168134688)

[CHƯƠNG 2: PHÂN TÍCH VÀ THIẾT KẾ HỆ THỐNG 23](#_Toc168134689)

[2.1. Phân tích bài toán 23](#_Toc168134690)

[2.1.1. Yêu cầu chức năng 23](#_Toc168134691)

[2.2.2. Yêu cầu phi chức năng 23](#_Toc168134692)

[2.2. Phân tích yêu cầu của hệ thống 24](#_Toc168134693)

[2.2.1. Tác nhân của hệ thống 24](#_Toc168134694)

[2.2.2. Biểu đồ use case tổng quát của hệ thống 24](#_Toc168134695)

[2.2.3. Biểu đồ phân rã chức năng đăng nhập 25](#_Toc168134696)

[2.2.4. Biểu đồ phân rã chức năng đăng ký 28](#_Toc168134697)

[2.2.5. Biểu đồ phân rã chức năng quản lý tài khoản 31](#_Toc168134698)

[2.2.6. Biểu đồ phân rã chức năng quản lý câu hỏi 34](#_Toc168134699)

[2.2.7. Biểu đồ phân rã chức năng quản lý đáp án 37](#_Toc168134700)

[2.2.8. Biểu đồ phân rã chức năng làm bài 40](#_Toc168134701)

[2.2.9. Biểu đồ phân rã chức năng cập nhật thông tin 43](#_Toc168134702)

[2.3. Thiết kế cơ sở dữ liệu 46](#_Toc168134703)

[2.3.1. Bảng người dùng 46](#_Toc168134704)

[2.3.2. Bảng câu hỏi 47](#_Toc168134705)

[2.3.3. Bảng đáp án 48](#_Toc168134706)

[2.3.4. Sơ đồ mối quan hệ thực thể 49](#_Toc168134707)

[CHƯƠNG 3: TRIỂN KHAI ỨNG DỤNG THI TRẮC NGHIỆM 50](#_Toc168134708)

[3.1. Các chức năng phía người quản trị 50](#_Toc168134709)

[3.1.1. Giao diện trang quản lý người dùng 50](#_Toc168134710)

[3.1.2. Giao diện trang quản lý câu hỏi 51](#_Toc168134711)

[3.1.3. Giao diện trang quản lý đáp án 52](#_Toc168134712)

[3.2. Các chức năng phía người dùng 53](#_Toc168134713)

[3.2.1. Giao diện trang đăng ký 53](#_Toc168134714)

[3.2.2. Giao diện trang đăng nhập 53](#_Toc168134715)

[3.2.3. Giao diện trang làm bài 54](#_Toc168134716)

[3.2.4. Giao diện trang thông tin người dùng 56](#_Toc168134717)

[3.3. Kiểm thử 57](#_Toc168134718)

[3.1.1. Kiểm thử giao diện 57](#_Toc168134719)

[3.1.2. Kiểm thử API 68](#_Toc168134720)

[KẾT LUẬN VÀ KIẾN NGHỊ 73](#_Toc168134721)

[TÀI LIỆU THAM KHẢO 74](#_Toc168134722)

# DANH MỤC CÁC CHỮ VIẾT TẮT

|  |  |  |  |
| --- | --- | --- | --- |
| **STT** | **Chữ viết tắt** | **Tên tiếng anh** | **Nghĩa tiếng việt** |
|  | API | Application Programming Interface | Giao diện lập trình ứng dụng |
|  | CSDL |  | Cơ sở dữ liệu |
|  | DOM | Document Object Model | Mô hình Đối tượng Tài liệu |
|  | HTML | Hypertext Markup Language | Ngôn ngữ Đánh dấu Siêu văn bản |
|  | HTTP | Hypertext Transfer Protocol | Giao thức truyền tải Siêu văn bản |
|  | HTTPS | Hypertext Transfer Protocol Secure | Giao thức truyền tải Siêu văn bản an toàn |
|  | JSON | JavaScript Object Notation | Định dạng Đối tượng JavaScript |
|  | RDBMS | Relational Database Management System | Hệ quản trị cơ sở dữ liệu quan hệ |
|  | SQL | Structured Query Language | Ngôn ngữ truy vấn cấu trúc |
|  | SPAs | Single Page Applications | Đơn trang |
|  | UI | User Interface | Giao diện người dùng |
|  | UML | Unified Modeling Language | Ngôn ngữ mô hình hóa thống nhất |

# DANH MỤC CÁC BẢNG

[Bảng 1. 1 Bảng phân biệt giữa ExpressJS và NodeJS 11](#_Toc166855500)

[Bảng 2. 1 Kịch bản use case chức năng đăng nhập 25](#_Toc166621702)

[Bảng 2. 2 Kịch bản use case chức năng đăng ký 28](#_Toc166621703)

[Bảng 2. 3 Kịch bản use case chức năng quản lý tài khoản 31](#_Toc166621704)

[Bảng 2. 4 Kịch bản use case chức năng quản lý câu hỏi 34](#_Toc166621705)

[Bảng 2. 5 Kịch bản use case chức năng quản lý đáp án 37](#_Toc166621706)

[Bảng 2. 6 Kịch bản use case chức năng làm bài 40](#_Toc166621707)

[Bảng 2. 7 Kịch bản use case chức năng cập nhật thông tin 43](#_Toc166621708)

[Bảng 2. 8 Bảng mô tả người dùng 46](#_Toc166621709)

[Bảng 2. 9 Bảng mô tả câu hỏi 47](#_Toc166621710)

[Bảng 2. 10 Bảng mô tả đáp án 48](#_Toc166621711)

[Bảng 3. 1 Kiểm thử chức năng đăng nhập 57](#_Toc166855473)

[Bảng 3. 2 Kiểm thử chức năng đăng ký 59](#_Toc166855474)

[Bảng 3. 3 Kiểm thử chức năng đổi mật khẩu 62](#_Toc166855475)

[Bảng 3. 4 Kiểm thử chức năng thêm người dùng 64](#_Toc166855476)

[Bảng 3. 5 Kiểm thử chức năng thêm câu hỏi 67](#_Toc166855477)

[Bảng 3. 6 Kiểm thử API đăng nhập 68](#_Toc166855478)

[Bảng 3. 7 Kiểm thử API đăng ký 69](#_Toc166855479)

[Bảng 3. 8 Kiểm thử API thêm người dùng 70](#_Toc166855480)

[Bảng 3. 9 Kiểm thử API cập nhật thông tin người dùng 71](#_Toc166855481)

[Bảng 3. 10 Kiểm thử API xóa người dùng 72](#_Toc166855482)

# DANH MỤC HÌNH ẢNH

[Hình 2. 1 Các tác nhân của hệ thống 24](#_Toc167281521)

[Hình 2. 2 Biểu đồ use case tổng quát của hệ thống 24](#_Toc167281522)

[Hình 2. 3 Biểu đồ phân rã chức năng đăng nhập 25](#_Toc167281523)

[Hình 2. 4 Biểu đồ tuần tự chức năng đăng nhập 26](#_Toc167281524)

[Hình 2. 5 Biểu đồ hoạt động quy trình đăng nhập 27](#_Toc167281525)

[Hình 2. 6 Biểu đồ phân rã chức năng đăng ký 28](#_Toc167281526)

[Hình 2. 7 Biểu đồ tuần tự chức năng đăng ký 29](#_Toc167281527)

[Hình 2. 8 Biểu đồ hoạt động quy trình đăng ký 30](#_Toc167281528)

[Hình 2. 9 Biểu đồ phân rã chức năng quản lý tài khoản 31](#_Toc167281529)

[Hình 2. 10 Biểu đồ tuần tự chức năng quản lý tài khoản 32](#_Toc167281530)

[Hình 2. 11 Biểu đồ hoạt động quy trình quản lý tài khoản 33](#_Toc167281531)

[Hình 2. 12 Biểu đồ phân rã chức năng quản lý câu hỏi 34](#_Toc167281532)

[Hình 2. 13 Biểu đồ tuần tự chức năng quản lý câu hỏi 35](#_Toc167281533)

[Hình 2. 14 Biểu đồ hoạt động quy trình quản lý câu hỏi 36](#_Toc167281534)

[Hình 2. 15 Biểu đồ phân rã chức năng quản lý đáp án 37](#_Toc167281535)

[Hình 2. 16 Biểu đồ tuần tự chức năng quản lý đáp án 38](#_Toc167281536)

[Hình 2. 17 Biểu đồ hoạt động quy trình quản lý đáp án 39](#_Toc167281537)

[Hình 2. 18 Biểu đồ phân rã chức năng làm bài 40](#_Toc167281538)

[Hình 2. 19 Biểu đồ tuần tự chức năng làm bài 41](#_Toc167281539)

[Hình 2. 20 Biểu đồ hoạt động quy trình làm bài 42](#_Toc167281540)

[Hình 2. 21 Biểu đồ phân rã chức năng cập nhật thông tin 43](#_Toc167281541)

[Hình 2. 22 Biểu đồ tuần tự chức năng cập nhật thông tin 44](#_Toc167281542)

[Hình 2. 23 Biểu đồ hoạt động quy trình cập nhật thông tin 45](#_Toc167281543)

[Hình 2. 25 Thiết kế bảng quản lý người dùng trên MongoDB 46](#_Toc167281544)

[Hình 2. 26 Thiết kế bảng quản lý câu hỏi trên MongoDB 47](#_Toc167281545)

[Hình 2. 27 Thiết kế bảng quản lý đáp án trên MongoDB 48](#_Toc167281546)

[Hình 2. 24 Sơ đồ mối quan hệ thực thể 49](#_Toc167281547)

[Hình 3. 1 Giao diện trang quản lý người dùng 50](#_Toc168499230)

[Hình 3. 2 Giao diện trang quản lý câu hỏi 51](#_Toc168499231)

[Hình 3. 3 Giao diện trang quản lý đáp án 52](#_Toc168499232)

[Hình 3. 4 Giao diện trang đăng ký 53](#_Toc168499233)

[Hình 3. 5 Giao diện trang đăng nhập 53](#_Toc168499234)

[Hình 3. 6 Giao diện trang làm bài 54](#_Toc168499235)

[Hình 3. 7 Giao diện trang thông tin bài làm 54](#_Toc168499236)

[Hình 3. 8 Giao diện hiển thị câu hỏi và đáp án 55](#_Toc168499237)

[Hình 3. 9 Giao diện xem kết quả 55](#_Toc168499238)

[Hình 3. 10 Giao diện trang thông tin người dùng 56](#_Toc168499239)

[Hình 3. 11 Kiểm thử API đăng nhập 68](#_Toc168499240)

[Hình 3. 12 Kiểm thử API đăng ký 69](#_Toc168499241)

[Hình 3. 13 Kiểm thử API thêm người dùng 70](#_Toc168499242)

[Hình 3. 14 Kiểm thử API cập nhật thông tin người dùng 71](#_Toc168499243)

[Hình 3. 15 Kiểm thử API xóa người dùng 72](#_Toc168499244)

# MỞ ĐẦU

Trong thời đại số hóa hiện nay, việc áp dụng công nghệ vào giáo dục ngày càng trở nên phổ biến và cần thiết. Đề tài "Phát triển ứng dụng thi trắc nghiệm trực tuyến với ReactJS" được triển khai nhằm tạo ra một công cụ hỗ trợ học tập và kiểm tra kiến thức tiện lợi, dễ sử dụng cho học sinh và giáo viên. Ứng dụng này không chỉ giúp tiết kiệm thời gian và chi phí tổ chức các kỳ thi truyền thống mà còn tạo ra môi trường học tập linh hoạt, giúp học sinh có thể ôn luyện và kiểm tra kiến thức mọi lúc, mọi nơi. Ngoài ra, giáo viên cũng dễ dàng theo dõi và đánh giá kết quả học tập của học sinh một cách nhanh chóng và chính xác.

ReactJS được chọn làm nền tảng phát triển chính cho đề tài này vì những ưu điểm vượt trội của nó trong việc xây dựng các ứng dụng web hiện đại. ReactJS, do Facebook phát triển, là một thư viện JavaScript mạnh mẽ và linh hoạt, cho phép phát triển giao diện người dùng một cách hiệu quả. Sử dụng ReactJS, ứng dụng thi trắc nghiệm trực tuyến có thể tận dụng khả năng tái sử dụng các component, giúp giảm thiểu công sức phát triển và bảo trì. Hơn nữa, với kiến trúc hướng thành phần của ReactJS, ứng dụng có thể dễ dàng mở rộng và tích hợp thêm các tính năng mới trong tương lai. Khả năng quản lý trạng thái linh hoạt và hiệu suất cao của ReactJS cũng đảm bảo rằng người dùng sẽ có trải nghiệm mượt mà và ổn định khi sử dụng ứng dụng.

# CHƯƠNG 1: TỔNG QUAN CÁC VẤN ĐỀ CẦN NGHIÊN CỨU

## 1.1. Lý do chọn đề tài

Phát triển ứng dụng trắc nghiệm trực tuyến là một đề tài có tính ứng dụng cao và mang lại giá trị thực tiễn rõ ràng trong bối cảnh giáo dục hiện đại. Trong thời đại số hóa, nhu cầu học tập và kiểm tra trực tuyến ngày càng tăng, và một ứng dụng trắc nghiệm trực tuyến có thể đáp ứng nhu cầu này một cách hiệu quả, góp phần nâng cao chất lượng giáo dục và cải thiện trải nghiệm học tập của người học.

Trước hết, với sự bùng nổ của công nghệ và internet, việc học tập và kiểm tra trực tuyến đã trở nên phổ biến hơn bao giờ hết. Học sinh và sinh viên ngày nay thường thích ứng dụng các công nghệ mới để tiếp cận kiến thức, tham gia vào quá trình học tập một cách hiệu quả và thú vị. Các ứng dụng trắc nghiệm trực tuyến không chỉ giúp học sinh ôn luyện, làm bài kiểm tra bất kỳ lúc nào mà còn hỗ trợ giáo viên theo dõi, đánh giá kết quả học tập của học sinh một cách chính xác.

Việc cho phép người dùng tự tạo bộ câu hỏi và kiểm tra làm lại một cách linh hoạt giúp tăng cường sự tự chủ và tinh thần sáng tạo trong quá trình học tập. Thầy cô giáo có thể tạo ra các bộ đề phong phú và đa dạng để hỗ trợ việc ôn tập và kiểm tra kiến thức của học sinh, đồng thời đảm bảo tính đa dạng và linh hoạt trong quá trình giảng dạy. Điều này không chỉ giúp học sinh chủ động hơn trong việc học tập mà còn tạo ra nhiều cơ hội để khám phá và nâng cao kiến thức theo cách riêng của mình. Các bộ câu hỏi có thể được tùy chỉnh theo mức độ khó khác nhau, phù hợp với từng đối tượng học sinh, giúp nâng cao hiệu quả học tập và đánh giá chính xác năng lực của từng học sinh.

Ứng dụng trắc nghiệm trực tuyến còn tạo ra một môi trường học tập tích cực, khuyến khích sự cạnh tranh và đam mê trong việc học tập. Việc lưu lại lịch sử trả lời, xếp hạng điểm số cũng giúp tạo ra một cộng đồng học tập đầy động lực và năng động. Học sinh có thể so sánh kết quả của mình với bạn bè và nỗ lực cải thiện điểm số, từ đó thúc đẩy tinh thần học tập và sự tiến bộ không ngừng. Ngoài ra, ứng dụng cũng có thể cung cấp các chức năng phân tích kết quả thi, giúp học sinh nhận ra điểm mạnh và điểm yếu của mình để từ đó có kế hoạch học tập hợp lý.

Tóm lại, lựa chọn đề tài "Phát triển ứng dụng trắc nghiệm trực tuyến với ReactJS" không chỉ đáp ứng nhu cầu thực tiễn trong việc học tập và kiểm tra trực tuyến mà còn thúc đẩy sự phát triển cá nhân và tạo ra một môi trường học tập đa dạng và sôi động. Việc sử dụng công nghệ ReactJS giúp xây dựng một ứng dụng mạnh mẽ, hiệu quả và dễ dàng mở rộng, đáp ứng tốt các yêu cầu của người dùng và mang lại trải nghiệm sử dụng tối ưu. ReactJS là một thư viện JavaScript mạnh mẽ và linh hoạt, giúp tạo ra các giao diện người dùng hiện đại và tương tác cao. Với tính năng cập nhật nhanh chóng và khả năng quản lý trạng thái hiệu quả, ReactJS sẽ giúp ứng dụng trắc nghiệm trực tuyến hoạt động mượt mà, mang lại trải nghiệm tốt nhất cho người dùng.

## 1.2. Mục tiêu nghiên cứu

Mục tiêu chung: Phát triển một ứng dụng trắc nghiệm trực tuyến sử dụng ReactJS nhằm cung cấp một nền tảng học tập và kiểm tra linh hoạt, tiện lợi và hiệu quả cho người dùng.

Mục tiêu cụ thể: Xây dựng giao diện người dùng thân thiện và dễ sử dụng, cho phép người dùng tương tác một cách tự nhiên và trực quan; Tích hợp các tính năng quản lý tài khoản người dùng, bao gồm đăng nhập, đăng ký và quản lý thông tin cá nhân; Phát triển chức năng tạo và quản lý bài kiểm tra, cho phép người dùng tạo ra các bài kiểm tra theo ý muốn và quản lý chúng một cách linh hoạt; Xây dựng chức năng làm bài trắc nghiệm, bao gồm hiển thị câu hỏi, lựa chọn đáp án, và xem kết quả ngay sau khi hoàn thành; Tối ưu hóa ứng dụng để đảm bảo hoạt động mượt mà trên các thiết bị và mạng internet khác nhau; Bảo mật thông tin cá nhân và dữ liệu của người dùng, đảm bảo tính bảo mật cho hệ thống.

## 1.3. Nội dung nghiên cứu

Cơ sở lý thuyết về ngôn ngữ được sử dụng: Tập trung tìm hiểu về các khái niệm cơ bản của ReactJS và cách sử dụng chúng để xây dựng giao diện người dùng linh hoạt. Đồng thời, tập trung vào các công cụ và thư viện phổ biến hỗ trợ việc phát triển ứng dụng ReactJS.

Khảo sát nhu cầu sử dụng: Nghiên cứu về nhu cầu và mong muốn của người dùng trong việc sử dụng ứng dụng trắc nghiệm trực tuyến, từ đó định hình các yếu tố quan trọng cần tính đến trong quá trình phát triển. Phân tích các yêu cầu cụ thể, như tính linh hoạt, dễ sử dụng, và tính năng phong phú.

Xây dựng ứng dụng hoàn chỉnh: Phát triển ứng dụng trắc nghiệm trực tuyến dựa trên cơ sở lý thuyết và thông tin từ khảo sát. Ứng dụng bao gồm các chức năng chính như tạo và quản lý câu hỏi, làm bài trắc nghiệm. Giao diện người dùng được thiết kế đơn giản và dễ sử dụng.

Kiểm thử và đánh giá: Kiểm thử, điều chỉnh và tối ưu hóa ứng dụng trước khi triển khai. Tiến hành các bước kiểm thử chất lượng để đảm bảo tính ổn định và đáng tin cậy của ứng dụng. Thu thập phản hồi từ người sử dụng và thực hiện các điều chỉnh cần thiết để cải thiện trải nghiệm người dùng.

## 1.4. Phương pháp nghiên cứu

- Phương pháp nghiên cứu lý thuyết: thu thập và nghiên cứu các tài liệu, tài nguyên trực tuyến và nghiên cứu liên quan đến ReactJS và phát triển ứng dụng.

- Phương pháp tổng hợp: Sau khi thu thập được các tài liệu và kiến thức liên quan, sẽ tổng hợp để áp dụng vào việc phát triển ứng dụng.

- Phương pháp thực nghiệm: Tiến hành chạy thử ứng dụng đã xây dựng để kiểm tra kết quả đạt được.

- Phương pháp nghiên cứu chuyên gia: Sử dụng chất xám của giáo viên và các đồng nghiệp để xây dựng thành công các chức năng của ứng dụng.

## 1.5. Các công nghệ sử dụng

### 1.5.1. Ngôn ngữ lập trình JavaScript

JavaScript là một ngôn ngữ lập trình đa năng, được sử dụng rộng rãi trong phát triển web để tạo ra các trang web tương tác và động. Được phát triển lần đầu tiên bởi Netscape vào năm 1995, JavaScript đã trở thành một phần không thể thiếu của bộ ba công nghệ web cơ bản cùng với HTML và CSS. JavaScript chạy trên trình duyệt của người dùng, cho phép các nhà phát triển tạo ra các hiệu ứng, xác thực form, và giao diện người dùng phong phú mà không cần tải lại trang. Với sự phát triển của Node.js, JavaScript cũng đã mở rộng phạm vi sử dụng của mình ra phía máy chủ, cho phép xây dựng các ứng dụng mạng nhanh chóng và hiệu quả. Sự phong phú của các thư viện và framework như React, Angular, và Vue.js càng làm tăng thêm sự phổ biến và sức mạnh của JavaScript trong việc xây dựng các ứng dụng web hiện đại.

#### Đặc điểm nổi bật của JavaScript

Độ phổ biến: JavaScript là ngôn ngữ phát triển web phổ biến nhất và được hỗ trợ trên tất cả các trình duyệt hiện đại. Việc sử dụng JavaScript giúp đảm bảo tính tương thích và khả năng tiếp cận rộng rãi cho ứng dụng.

Tính linh hoạt và đa dạng: JavaScript cho phép phát triển một loạt các ứng dụng web, từ các trang web tĩnh đến các ứng dụng động và phức tạp. Khả năng tích hợp JavaScript với HTML và CSS giúp tạo ra giao diện người dùng linh hoạt và thân thiện.

Cộng đồng lớn và hỗ trợ mạnh mẽ: JavaScript có một cộng đồng lớn và tích cực, với hàng ngàn thư viện và framework khác nhau để hỗ trợ phát triển ứng dụng, website. Điều này giúp giải quyết các vấn đề phức tạp một cách nhanh chóng và hiệu quả.

Hiệu suất cao: JavaScript có thể tạo ra các ứng dụng web có hiệu suất cao, đáp ứng nhanh chóng yêu cầu của người dùng mà không cần tải lại trang.

Tính năng cập nhật: JavaScript luôn được cải tiến và cập nhật với những phiên bản mới, giúp ứng dụng luôn cập nhật với công nghệ mới nhất và đáp ứng được các yêu cầu mới từ người dùng.

#### Ưu nhược điểm của JavaScript

JavaScript có nhiều ưu điểm nổi bật, làm cho nó trở thành một trong những ngôn ngữ lập trình phổ biến nhất. Trước hết, JavaScript chạy trên mọi trình duyệt mà không cần bất kỳ phần mềm đặc biệt nào, giúp tạo ra các trang web tương tác và động một cách dễ dàng. Khả năng thao tác DOM cho phép thay đổi nội dung và cấu trúc trang web theo thời gian thực. Thêm vào đó, với sự ra đời của Node.js, JavaScript mở rộng khả năng ra phía máy chủ, giúp xây dựng các ứng dụng mạng hiệu suất cao. Hệ sinh thái phong phú với hàng ngàn thư viện và framework như React, Angular, và Vue.js hỗ trợ phát triển nhanh chóng và quản lý các ứng dụng phức tạp. Sự hỗ trợ mạnh mẽ từ cộng đồng lớn và các tài liệu phong phú cũng là một điểm cộng lớn.

Tuy nhiên, JavaScript cũng có những nhược điểm cần cân nhắc. Là một ngôn ngữ đơn luồng, JavaScript có thể gặp khó khăn trong việc xử lý các tác vụ tính toán nặng, mặc dù điều này có thể được giảm thiểu bằng cách sử dụng Web Workers. Bên cạnh đó, việc quản lý các tác vụ không đồng bộ có thể phức tạp và dễ dẫn đến "callback hell", mặc dù các công cụ như promises và async/await đã cải thiện vấn đề này. JavaScript cũng dễ bị tấn công bảo mật như Cross-Site Scripting (XSS) nếu không được xử lý đúng cách. Hơn nữa, sự khác biệt giữa các trình duyệt có thể gây ra các vấn đề tương thích, yêu cầu các nhà phát triển phải thực hiện nhiều kiểm tra và điều chỉnh mã.

### 1.5.2. Ngôn ngữ HTML và CSS

HTML (viết tắt của Hypertext Markup Language) là ngôn ngữ được sử dụng rộng rãi nhất để viết các trang web. Hypertext là cách mà các trang web (các tài liệu HTML) được kết nối với nhau thông qua các đường link, tạo ra một mạng lưới các trang liên kết. Như tên gọi đã gợi ý, HTML là ngôn ngữ đánh dấu bằng thẻ (Markup Language), có nghĩa là sử dụng các thẻ HTML để đánh dấu một tài liệu văn bản nhằm chỉ cho trình duyệt web cách để cấu trúc và hiển thị nó ra màn hình. Các thẻ này không hiển thị trên trang web cuối cùng mà chỉ định dạng cách nội dung sẽ được trình bày. Ví dụ, các thẻ như <h1>, <p>, và <a> sẽ tạo ra các tiêu đề, đoạn văn và liên kết tương ứng. Vậy nên, HTML đóng vai trò là khung sườn của một trang web, xác định bố cục cơ bản và các thành phần nội dung.

CSS (viết tắt của Cascading Style Sheets) là ngôn ngữ giúp tạo nên phong cách, màu sắc, và hình ảnh cho trang HTML. Nói một cách ngắn gọn, CSS là ngôn ngữ tạo phong cách cho trang web. Trong khi HTML định dạng các phần tử trên website như tạo ra các đoạn văn bản, các tiêu đề, bảng biểu, thì CSS sẽ giúp thêm style vào các phần tử HTML đó. CSS cho phép chúng ta thay đổi bố cục, màu sắc trang, màu chữ, font chữ, và nhiều yếu tố khác để làm cho trang web trở nên hấp dẫn và dễ nhìn hơn. Ví dụ, bạn có thể sử dụng CSS để đặt nền cho trang web, thay đổi màu sắc và kích thước của chữ, thêm khoảng cách giữa các phần tử, và thậm chí tạo ra các hiệu ứng động. CSS còn hỗ trợ tính năng phân tầng (cascading), cho phép các style được áp dụng một cách có thứ tự và linh hoạt, giúp việc quản lý và thay đổi giao diện trang web trở nên dễ dàng hơn.

### 1.5.3. ReactJS

ReactJS là một thư viện JavaScript phổ biến được sử dụng để xây dựng giao diện người dùng (UI) đơn trang (Single Page Applications - SPAs). Được phát triển bởi Facebook, ReactJS tập trung vào việc tạo ra các giao diện người dùng hiệu quả và dễ bảo trì. Dưới đây là một số đặc điểm chính của ReactJS:

*Component-Based Architecture (Kiến trúc dựa trên thành phần):* ReactJS sử dụng mô hình kiến trúc dựa trên thành phần. Mỗi phần của giao diện người dùng được biểu diễn dưới dạng component độc lập, có thể được tái sử dụng và quản lý độc lập.

*Virtual DOM (DOM Ảo):* React sử dụng một biến thể của DOM gọi là Virtual DOM để cải thiện hiệu suất. Thay vì cập nhật trực tiếp DOM mỗi khi có thay đổi, React tạo ra một bản sao của DOM trong bộ nhớ (Virtual DOM), so sánh nó với DOM hiện tại và chỉ cập nhật những phần thay đổi, giúp giảm tải cho trình duyệt.

*JSX (JavaScript XML):* JSX là một phần quan trọng của React, cho phép viết mã JavaScript có cấu trúc giống HTML. JSX giúp tạo ra mã nguồn dễ đọc và hiểu, đồng thời giúp React tạo ra cây phần tử (element tree) một cách dễ dàng.

*Unidirectional Data Flow (Luồng dữ liệu một chiều):* Dữ liệu trong React chuyển động theo một chiều duy nhất từ component cha đến các component con. Điều này giúp dễ dàng theo dõi và duy trì trạng thái ứng dụng.

*React Hooks:* Hooks là một tính năng giới thiệu từ phiên bản React 16.8, giúp bạn sử dụng state và các tính năng React khác trong các hàm component chức năng, thay vì chỉ trong các hàm lifecycle.

*Ecosystem phong phú***:** React có một hệ sinh thái mạnh mẽ với nhiều thư viện hỗ trợ, bao gồm React Router cho điều hướng, Redux cho quản lý trạng thái, và nhiều thư viện khác.

*Community lớn:* Với sự ủng hộ từ Facebook và cộng đồng lập trình rộng lớn, React có một cộng đồng sáng tạo và đa dạng. Điều này đồng nghĩa với việc có nhiều tài nguyên học và hỗ trợ sẵn có.

ReactJS thường được sử dụng trong việc phát triển các ứng dụng web động, các ứng dụng đơn trang (SPA), và có thể được tích hợp với các thư viện và framework khác để tạo ra các ứng dụng web mạnh mẽ và hiệu quả.

* Ưu điểm của ReactJS

- Hiệu suất cao: ReactJS sử dụng Virtual DOM để tối ưu hiệu suất render, giúp giảm thiểu số lần render lại DOM thực sự, tăng hiệu suất của ứng dụng.

- Component-based: ReactJS sử dụng cấu trúc component-based, giúp tái sử dụng code, dễ bảo trì và phát triển ứng dụng một cách linh hoạt.

- Hỗ trợ cộng đồng mạnh mẽ: Cộng đồng ReactJS lớn mạnh, cung cấp nhiều tài liệu, ví dụ và hỗ trợ giúp lập trình viên giải quyết các vấn đề một cách dễ dàng.

- JSX (JavaScript XML) cho phép viết HTML trong JavaScript, làm cho việc phát triển giao diện người dùng trở nên dễ dàng và dễ hiểu hơn.

- ReactJS có thể được kết hợp với các thư viện quản lý trạng thái như Redux để quản lý trạng thái của ứng dụng một cách hiệu quả và dễ dàng.

* Nhược điểm của ReactJS

- Dành cho những người mới bắt đầu, có thể cần thời gian để làm quen với cú pháp và các khái niệm của React.

- Dung lượng tải lớn: ReactJS cần tải thêm một số thư viện và dependencies, làm tăng dung lượng tải của ứng dụng.

- Đồng bộ: ReactJS sử dụng khái niệm không đồng bộ, điều này có thể làm tăng độ phức tạp và khó khăn trong quản lý trạng thái ứng dụng.

- Thư viện: Để làm việc với ReactJS hiệu quả, bạn có thể cần phải học thêm các thư viện và công nghệ bổ sung như Redux, Thunks, Saga, và GraphQL.

### 1.5.4. NodeJS

Node.js là một nền tảng phát triển ứng dụng backend mã nguồn mở, xây dựng trên JavaScript và được thiết kế để thực hiện các ứng dụng mạng có khả năng mở rộng. Node.js cho phép các nhà phát triển sử dụng JavaScript để viết mã máy chủ (server-side code), mà trước đây thường chỉ được sử dụng cho mã nguồn mặt trước (client-side code) trong các ứng dụng web.

*JavaScript Everywhere:* Node.js cho phép sử dụng JavaScript để viết mã cả phía server và phía client, tạo ra một môi trường phát triển đồng nhất và giúp các nhà phát triển dễ dàng chuyển đổi giữa phát triển phía máy chủ và phát triển phía máy khách.

*Single-threaded, Non-blocking I/O:* Node.js hoạt động trên kiến ​​trúc sự kiện (event-driven) và non-blocking I/O model. Điều này có nghĩa là nó sử dụng một luồng duy nhất để xử lý các yêu cầu, nhưng có khả năng xử lý hàng loạt các yêu cầu mà không chờ đợi các hoạt động I/O hoàn tất. Điều này tăng hiệu suất và khả năng mở rộng của ứng dụng.

*Cộng đồng lớn và tích cực:* Node.js có một cộng đồng lớn và tích cực, cung cấp nhiều gói module, thư viện, framework và công cụ phát triển hữu ích.

*Nền tảng mở rộng:* Node.js được thiết kế để xử lý các ứng dụng với khả năng mở rộng cao. Bằng cách sử dụng mô hình non-blocking I/O và sự kiện-driven, Node.js có thể xử lý hàng nghìn kết nối đồng thời mà không cần tạo ra một luồng mới cho mỗi kết nối.

*Phát triển dựa trên package manager npm:* Node.js sử dụng npm (Node Package Manager) làm trình quản lý gói (package manager), cung cấp cho nhà phát triển một cách tiện lợi để quản lý các phụ thuộc và chia sẻ mã nguồn.

*Phù hợp cho các ứng dụng thời gian thực:* Với tính chất sự kiện-driven và non-blocking I/O, Node.js thích hợp cho việc xây dựng các ứng dụng thời gian thực như trò chơi trực tuyến, ứng dụng chat, ứng dụng realtime analytics…

Bảng 1. 1 Bảng phân biệt giữa ExpressJS và NodeJS

|  |  |  |
| --- | --- | --- |
| **Đặc điểm** | **ExpressJS** | **NodeJS** |
| Framework | Là framework của Node.js, sử dụng để xây dựng phần phụ trợ của ứng dụng web. | Không phải là framework |
| Ngôn ngữ lập trình | JavaScript | Nhiều ngôn ngữ lập trình khác nhau như C/C++, TypeScript, CoffeeScript, Ruby |
| Cài đặt | Cần cài đặt ExpressJS cùng Node.js | Chỉ cần cài đặt Node.js |
| Phù hợp | Các dự án có quy mô nhỏ | Các dự án có quy mô lớn |
| Sử dụng | Chỉ được sử dụng ở phía máy chủ | Có thể sử dụng ở cả phía máy chủ và máy khách |
| Tương thích | Tương thích với tất cả các hệ điều hành tương thích với Node.js | Tương thích với tất cả các hệ điều hành chính |
| Cung cấp | Các thành phần định tuyến và middleware để hỗ trợ phát triển ứng dụng web | Nhiều tính năng cho việc xây dựng ứng dụng web |

### 1.5.5. Express

Express.js là một framework web cho Node.js được phát triển để xây dựng các ứng dụng web và API một cách nhanh chóng và dễ dàng. Nó được coi là một trong những framework web phổ biến nhất cho Node.js, được sử dụng rộng rãi trong cộng đồng phát triển web.

*Minh bạch và dễ sử dụng*: Express.js được thiết kế để làm cho việc phát triển ứng dụng web trở nên đơn giản và dễ dàng. Nó cung cấp cú pháp rõ ràng và API trực quan giúp việc xây dựng các ứng dụng trở nên dễ dàng hơn.

*Middleware-based*: Express.js là một framework middleware-based, điều này có nghĩa là bạn có thể sử dụng các middleware để xử lý các yêu cầu và phản hồi HTTP trước khi chúng đến các xử lý cuối cùng. Điều này giúp tăng tính linh hoạt và tái sử dụng code.

*Routing*: Express.js cung cấp một cách tiện lợi để xác định các tuyến đường (routes) cho ứng dụng của bạn. Bạn có thể định nghĩa các tuyến đường cho các phương thức HTTP như GET, POST, PUT và DELETE, và xử lý các yêu cầu được gửi đến các tuyến đường đó.

*Middleware mạnh mẽ*: Express.js đi kèm với nhiều middleware tích hợp sẵn và có thể mở rộng, cho phép bạn thêm các tính năng như xác thực người dùng, ghi nhật ký, nén dữ liệu, và nhiều hơn nữa vào ứng dụng của bạn.

*Hỗ trợ view engine*: Express.js hỗ trợ sử dụng các view engine như Pug, EJS, Handlebars để tạo các giao diện người dùng động dễ dàng hơn.

*RESTful API development*: Express.js là một lựa chọn phổ biến cho việc phát triển RESTful APIs, nhờ vào cú pháp đơn giản và khả năng xử lý các yêu cầu và phản hồi HTTP một cách linh hoạt.

*Môi trường phát triển cộng đồng mạnh mẽ*: Express.js có một cộng đồng phát triển mạnh mẽ và tích cực, với nhiều tài liệu, ví dụ và các module mở rộng hữu ích được cung cấp bởi cộng đồng.

* Ưu điểm của ExpressJS

- Đơn giản và dễ sử dụng: Cú pháp đơn giản giúp nắm bắt và triển khai các tính năng một cách nhanh chóng và hiệu quả.

- Linh hoạt: Không áp đặt cấu trúc cụ thể, cho phép tự do tùy chỉnh và mở rộng ứng dụng theo ý muốn.

- Hỗ trợ middleware: Cung cấp hệ thống middleware mạnh mẽ, linh hoạt trong việc thực hiện các chức năng như xác thực, ghi log, nén dữ liệu và xử lý lỗi.

- Hiệu suất cao: Dựa trên Node.js, nền tảng có hiệu suất cao, cho phép xử lý nhanh chóng các yêu cầu web và có khả năng mở rộng tốt.

* Nhược điểm của ExpressJS

- Thiếu cấu trúc: ExpressJS không áp đặt cấu trúc nghiêm ngặt, dẫn đến việc tổ chức dự án và quản lý mã nguồn khó khăn, đặc biệt với các ứng dụng lớn và phức tạp.

- Khả năng mở rộng: Quản lý mã nguồn và mở rộng có thể trở nên khó khăn với ExpressJS, đặc biệt khi ứng dụng phát triển lớn và phức tạp.

- Cộng đồng hỗ trợ: Mặc dù có một cộng đồng đầy đủ tài liệu, nhưng ExpressJS không có mức độ hỗ trợ như các framework web khác như Angular hoặc React.

### 1.5.6. Mongodb

#### MongoDB

MongoDB được bắt đầu phát triển vào đầu năm 2007 khi công ty 10gen đang phát triển một nền tảng tương tự dịch vụ Azure của Microsoft. Công ty 10gen là một công ty phần mềm có trụ sở tại New York, nay được đổi tên thành MongoDB Inc. Việc phát triển ban đầu tập trung vào xây dựng PaaS (một nền tảng dịch vụ) nhưng sau đó vào năm 2009, MongoDB đã xuất hiện trên thị trường như một dự án mã nguồn mở máy chủ cơ sở dữ liệu và được duy trì bởi chính tổ chức này.

MongoDB là một hệ quản trị cơ sở dữ liệu mã nguồn mở, là CSDL thuộc NoSql và được hàng triệu người sử dụng. MongoDB là một database hướng tài liệu (document), các dữ liệu được lưu trữ trong document kiểu JSON thay vì dạng bảng như CSDL quan hệ nên truy vấn sẽ rất nhanh.

Với CSDL quan hệ chúng ta có khái niệm bảng, các cơ sở dữ liệu quan hệ (như MySQL hay SQL Server...) sử dụng các bảng để lưu dữ liệu thì với MongoDB chúng ta sẽ dùng khái niệm là collection thay vì bảng. So với RDBMS thì trong MongoDB collection ứng với table, còn document sẽ ứng với row, MongoDB sẽ dùng các document thay cho row trong RDBMS. Các collection trong MongoDB được cấu trúc rất linh hoạt, cho phép các dữ liệu lưu trữ không cần tuân theo một cấu trúc nhất định. Thông tin liên quan được lưu trữ cùng nhau để truy cập truy vấn nhanh thông qua ngôn ngữ truy vấn MongoDB

NoSQL là 1 dạng CSDL mã nguồn mở và được viết tắt bởi: None Relational SQL hay có nơi thường gọi là Not-Only SQL. NoSQL được phát triển trên Javascript Framework với kiểu dữ liệu là JSON và dạng dữ liệu theo kiểu key và value. NoSQL ra đời như là 1 mảnh vá cho những khuyết điểm và thiếu xót cũng như hạn chế của mô hình dữ liệu quan hệ RDBMS (Relational Database Management System - Hệ quản trị cơ sở dữ liệu quan hệ) về tốc độ, tính năng, khả năng mở rộng. Với NoSQL bạn có thể mở rộng dữ liệu mà không lo tới những việc như tạo khóa ngoại, khóa chính, kiểm tra ràng buộc. NoSQL bỏ qua tính toàn vẹn của dữ liệu và transaction để đổi lấy hiệu suất nhanh và khả năng mở rộng. NoSQL được sử dụng ở rất nhiều công ty, tập đoàn lớn, ví dụ như FaceBook sử dụng Cassandra do FaceBook phát triển, Google phát triển và sử dụng BigTable.

#### Ưu điểm của mongoDB

Do MongoDB sử dụng lưu trữ dữ liệu dưới dạng Document JSON nên mỗi một collection sẽ có các kích cỡ và các document khác nhau, linh hoạt trong việc lưu trữ dữ liệu, nên bạn muốn gì thì cứ insert vào thoải mái.

Dữ liệu trong MongoDB không có sự ràng buộc lẫn nhau, không có join như trong RDBMS nên khi insert, xóa hay update nó không cần phải mất thời gian kiểm tra xem có thỏa mãn các ràng buộc dữ liệu như trong RDBMS.

MongoDB rất dễ mở rộng (Horizontal Scalability). Trong MongoDB có một khái niệm cluster là cụm các node chứa dữ liệu giao tiếp với nhau, khi muốn mở rộng hệ thống ta chỉ cần thêm một node với vào cluster: Trường dữ liệu “\_id” luôn được tự động đánh index (chỉ mục) để tốc độ truy vấn thông tin đạt hiệu suất cao nhất. Khi có một truy vấn dữ liệu, bản ghi được cached lên bộ nhớ Ram, để phục vụ lượt truy vấn sau diễn ra nhanh hơn mà không cần phải đọc từ ổ cứng.

Hiệu năng cao: Tốc độ truy vấn (find, update, insert, delete) của MongoDB nhanh hơn hẳn so với các hệ quản trị cơ sở dữ liệu quan hệ (RDBMS). Với một lượng dữ liệu đủ lớn thì thử nghiệm cho thấy tốc độ insert của MongoDB có thể nhanh tới gấp 100 lần so với MySQL.

#### Nhược điểm của mongoDB

Một ưu điểm của MongoDB cũng chính là nhược điểm của nó. MongoDB không có các tính chất ràng buộc như trong RDBMS nên khi thao tác với mongoDB thì phải hết sức cẩn thận.

Tốn bộ nhớ do dữ liệu lưu dưới dạng key-value, các collection chỉ khác về value do đó key sẽ bị lặp lại. Không hỗ trợ join nên dễ bị dư thừa dữ liệu.

Khi insert/update/remove bản ghi, MongoDB sẽ chưa cập nhật ngay xuống ổ cứng, mà sau 60 giây MongoDB mới thực hiện ghi toàn bộ dữ liệu thay đổi từ RAM xuống ổ cứng điểu này sẽ là nhược điểm vì sẽ có nguy cơ bị mất dữ liệu khi xảy ra các tình huống như mất điện...

#### Các trường hợp phù hợp với việc sử dụng MongoDB

Quản lý và truyền tải content – Quản lý đa dạng nhiều product của content chỉ trong một kho lưu trữ data cho phép thay đổi và phản hồi nhanh chóng mà không chịu thêm phức tạp thêm từ hệ thống content.

Cấu trúc Mobile và Social – MongoDB cung cấp một platform có sẵn, phản xạ nhanh, và dễ mở rộng cho phép rất nhiều khả năng đột phá, phân tích real-time, và hỗ trợ toàn cầu.

Quản lý data người dùng– Tận dụng khả năng query nhanh chóng cho phân tích real-time trên cơ sở dữ liệu người dùng cực lớn với các mô hình data phức tạp bằng các schema linh hoạt và tự động sharding cho mở rộng chiều ngang.

### 1.5.7. Mô hình Web API.

#### Web API

Web API là một phương thức dùng để cho phép các ứng dụng khác nhau có thể giao tiếp, trao đổi dữ liệu qua lại. Dữ liệu được Web API trả lại thường ở dạng JSON hoặc XML thông qua giao thức HTTP hoặc HTTPS.

#### Những điểm nổi bật của Web API

Web API hỗ trợ restful đầy đủ các phương thức: Get/Post/put/delete dữ liệu. Nó giúp bạn xây dựng các HTTP service một cách rất đơn giản và nhanh chóng. Nó cũng có khả năng hỗ trợ đầy đủ các thành phần HTTP: URI, request/response headers, caching, versioning, content format**.**

- Tự động hóa sản phẩm: Với web API, chúng ta sẽ tự động hóa quản lý công việc, cập nhật luồng công việc, giúp tăng năng suất và tạo hiệu quả công việc cao.

- Khả năng tích hợp linh động: API cho phép lấy nội dung từ bất kỳ website hoặc ứng dụng nào một cách dễ dàng nếu được cho phép, tăng trải nghiệm người dùng. API hoạt động như một chiếc cổng, cho phép các công ty chia sẻ thông tin được chọn nhưng vẫn tránh được những yêu cầu không mong muốn.

- Cập nhật thông tin thời gian thực: API có chức năng thay đổi và cập nhật thay đổi theo thời gian thực. Với công nghệ này, dữ liệu sẽ được truyền đi tốt hơn, thông tin chính xác hơn, dịch vụ cung cấp linh hoạt hơn.

- Có tiêu chuẩn chung dễ sử dụng: Bất kỳ người dùng, công ty nào sử dụng cũng có thể điều chỉnh nội dung, dịch vụ mà họ sử dụng. Hỗ trợ đầy đủ các thành phần MVC như: routing, controller...

#### Cách thực hoạt động của Web API

Đầu tiên là xây dựng URL API để bên thứ ba có thể gửi request dữ liệu đến máy chủ cung cấp nội dung, dịch vụ thông qua giao thức HTTP hoặc HTTPS. Tại web server cung cấp nội dung, các ứng dụng nguồn sẽ thực hiện kiểm tra xác thực nếu có và tìm đến tài nguyên thích hợp để tạo nội dung trả về kết quả.

Server trả về kết quả theo định dạng JSON hoặc XML thông qua giao thức HTTP/HTTPS. Tại nơi yêu cầu ban đầu là ứng dụng web hoặc ứng dụng di động, dữ liệu JSON/XML sẽ được parse để lấy data. Sau khi có được data thì thực hiện tiếp các hoạt động như lưu dữ liệu xuống Cơ sở dữ liệu, hiển thị dữ liệu…

#### Ưu và nhược điểm của Web API

* Ưu điểm

- Web API được sử dụng hầu hết trên các ứng dụng desktop, ứng dụng mobile và ứng dụng website; Linh hoạt với các định dạng dữ liệu khi trả về client: Json, XML hay định dạng khác; Nhanh chóng xây dựng HTTP service: URI, request/response headers; Mã nguồn mở, hỗ trợ chức năng RESTful đầy đủ, sử dụng bởi bất kì client nào hỗ trợ XML, Json; Hỗ trợ đầy đủ các thành phần MVC; Giao tiếp hai chiều được xác nhận trong các giao dịch, đảm bảo độ tin cậy cao.

* Nhược điểm

- Do web API còn khá mới nên chưa thể đánh giá nhiều về nhược điểm của mô hình nay. Tuy nhiên, có hai nhược điểm dễ dàng nhận thấy: Web API chưa hoàn toàn phải là RESTful service, mới chỉ hỗ trợ mặc định GET, POST; Để sử dụng hiệu quả cần có kiến thức chuyên sâu, có kinh nghiệm backend tốt; Tốn thời gian và chi phí cho việc phát triển, nâng cấp và vận hành. Có thể gặp vấn đề về bảo mật khi hệ thống bị tấn công nếu không giới hạn điều kiện kỹ.

## 1.6. Một số công cụ hỗ trợ khác

### 1.6.1. Microsoft Visual Studio Code

Microsoft Visual Studio Code (VSCode) là một trình soạn thảo mã nguồn mở và miễn phí, được phát triển bởi Microsoft. Nó là một ứng dụng chuyên nghiệp dùng để phát triển và debug ứng dụng, hỗ trợ nhiều ngôn ngữ lập trình và framework. Dưới đây là một số điểm đáng chú ý về Visual Studio Code.

Mã nguồn mở và Miễn phí: VSCode là một sản phẩm mã nguồn mở và hoàn toàn miễn phí để sử dụng. Người dùng có quyền truy cập mã nguồn và thậm chí có thể đóng góp vào sự phát triển của nó.

Hỗ trợ nhiều ngôn ngữ: Visual Studio Code hỗ trợ nhiều ngôn ngữ lập trình, bao gồm JavaScript, TypeScript, Python, Java, C++, Go, PHP, và nhiều ngôn ngữ khác nữa. Điều này làm cho nó trở thành một lựa chọn phổ biến cho nhiều nhà phát triển với đa dạng dự án.

Hỗ trợ mở rộng: VSCode có một hệ thống mở rộng mạnh mẽ, cho phép người dùng cài đặt và sử dụng nhiều extension từ cộng đồng để mở rộng chức năng của trình soạn thảo. Có nhiều extension hữu ích như thư viện IntelliSense, Debuggers, Theme, và nhiều thứ khác.

Hỗ trợ Git tích hợp: VSCode có tích hợp sẵn với Git, cung cấp các tính năng quản lý phiên bản, như theo dõi thay đổi, commit, push, và giải quyết xung đột.

Debugging tích hợp: VSCode hỗ trợ môi trường debug tích hợp cho nhiều ngôn ngữ. Người dùng có thể xem biểu đồ stack, quản lý breakpoint, và theo dõi giá trị biến trong quá trình thực thi mã.

Visual Studio Code là một công cụ phổ biến và mạnh mẽ, được ưa chuộng trong cộng đồng lập trình viên và phát triển ứng dụng cho nhiều nền tảng, bao gồm Windows, macOS, và Linux.

### 1.6.2. Postman

Postman là một ứng dụng mã nguồn mở dùng để phát triển và kiểm thử các API (Application Programming Interface). Công nghệ cung cấp môi trường cho các nhà phát triển thực hiện hoạt động tạo, chia sẻ, kiểm thử và quản lý các API của họ.

Nền tảng Postman có các tính năng như tạo yêu cầu HTTP, kiểm thử tự động, quản lý biến môi trường và biến toàn cục. Những tiện ích này sẽ giúp nhà phát triển tiết kiệm thời gian trong việc phát triển và kiểm thử API. Đồng thời, Postman còn có chế độ tương tác thông minh với API, giúp người dùng hiểu rõ hơn về cách API hoạt động và phản hồi của chúng.

* Các chức năng cơ bản của Postman

- Gửi HTTP Request với các phương thức GET, POST, PUT, DELETE.

- Hỗ trợ gửi dữ liệu dưới dạng form (key-value), text, và JSON.

- Hiển thị kết quả trả về dưới nhiều định dạng: text, hình ảnh, XML, JSON.

- Hỗ trợ authorization cho việc xác thực.

- Cho phép tùy chỉnh và thay đổi header của các request để điều chỉnh yêu cầu gửi đi.

* Ưu điểm của Postman

Giao diện thân thiện: Postman cung cấp giao diện người dùng dễ sử dụng và thân thiện, giúp người dùng tạo và quản lý các yêu cầu API một cách hiệu quả.

Tự động hóa kiểm thử: Cung cấp khả năng tạo các bộ kiểm thử tự động để kiểm tra API, giúp giảm thiểu các lỗi và tối ưu hóa hiệu suất của API

Quản lý biến môi trường: Cho phép lập trình viên quản lý và sử dụng các biến môi trường và biến toàn cục, dễ dàng thay đổi các thiết lập môi trường khi cần thiết.

Chia sẻ dễ dàng: Cung cấp khả năng chia sẻ các collection và bộ kiểm thử với đồng nghiệp hoặc cộng đồng, tạo điều kiện cho việc hợp tác và kiểm thử chung.

Tương tác mạnh mẽ với API: Postman cung cấp các tính năng tương tác mạnh mẽ với API, giúp người dùng hiểu rõ cách API hoạt động và phản hồi của chúng.

Quản lý dự án thông minh: Cung cấp các công cụ quản lý tập tin và tài nguyên liên quan đến API, giúp dễ dàng tổ chức và quản lý dự án.

* Nhược điểm của Postman

Giới hạn trong phiên bản miễn phí: Phiên bản miễn phí của Postman có giới hạn về số lượt gửi yêu cầu và các tính năng cao cấp, điều này có thể tạo ra hạn chế đối với các dự án lớn.

Tính tương thích: Mặc dù Postman hỗ trợ nhiều loại yêu cầu API và các kiểu dữ liệu phản hồi nhưng vẫn xuất hiện một số hạn chế khi tương tác với các loại API phức tạp và cần nhiều xử lý.

Quản lý phiên bản và quy trình: Trong môi trường phát triển phần mềm lớn, việc quản lý phiên bản và quy trình thường trở nên phức tạp khi dự án mở rộng và có nhiều người tham gia.

Hiệu suất khi thực hiện kiểm thử lớn: Trong trường hợp thực hiện kiểm thử lớn và phức tạp, việc quản lý và theo dõi các bộ kiểm thử sẽ không hiệu quả.

Yêu cầu kỹ năng: Để sử dụng Postman một cách hiệu quả, người dùng cần phải có kiến thức vững về API và quá trình phát triển phần mềm. Đây sẽ là một thử thách lớn đối với người dùng mới.

### 1.6.3. StarUML

StarUML là một công cụ mô hình hóa nền tảng, được thiết kế để hỗ trợ Unified Modeling Language (UML). Nó hỗ trợ phương pháp tiếp cận Model Driven Architecture (MDA) bằng cách hỗ trợ các khái niệm hồ sơ UML. Cụ thể, StarUML giúp phân tích và thiết kế hệ thống, một phần quan trọng của bất kỳ dự án phần mềm nào.

StarUML giúp tối ưu hóa năng suất và chất lượng của dự án phần mềm bằng cách cho phép mô hình hóa, không phụ thuộc vào người code, ngôn ngữ code, hoặc nền tảng sử dụng. Nhờ vào việc mô hình hóa, các thiết kế trở nên dễ đọc và dễ hiểu hơn.

* StarUML hỗ trợ đầy đủ cho các khái niệm hồ sơ UML, bao gồm:

Lớp (Class): Cho phép bạn mô hình hóa các lớp và các mối quan hệ giữa chúng. Bạn có thể tạo các thuộc tính, phương thức, và các mối quan hệ như kế thừa, kết hợp và phụ thuộc.

Use Case: Hỗ trợ mô tả các tác nhân (actors) và các trường hợp sử dụng (use cases) của hệ thống. Điều này giúp bạn nắm bắt các yêu cầu và chức năng chính của hệ thống từ góc nhìn người dùng.

Sequence: Cho phép bạn mô hình hóa các tương tác giữa các đối tượng theo thứ tự thời gian. Bạn có thể sử dụng sơ đồ này để minh họa cách các đối tượng giao tiếp với nhau qua các thông điệp.

Sơ đồ hoạt động (Activity): Hỗ trợ mô tả các luồng công việc hoặc quy trình. Sơ đồ này giúp bạn biểu diễn các hoạt động, hành động và các quyết định trong một quy trình kinh doanh hoặc hệ thống.

Sơ đồ lớp (Package): Cho phép bạn tổ chức các phần của hệ thống vào các gói. Sơ đồ này giúp bạn quản lý cấu trúc tổng thể của hệ thống và các mối quan hệ giữa các gói.

# CHƯƠNG 2: PHÂN TÍCH VÀ THIẾT KẾ HỆ THỐNG

## 2.1. Phân tích bài toán

### 2.1.1. Yêu cầu chức năng

* Chức năng của người dùng (user)

Đăng nhập và đăng ký: Cho phép người dùng đăng nhập vào hệ thống bằng tài khoản đã đăng ký. Nếu chưa có tài khoản, người dùng có thể đăng ký mới bằng cách cung cấp thông tin cá nhân cần thiết và địa chỉ email.

Quản lý tài khoản: Người dùng có thể cập nhật thông tin cá nhân, đổi mật khẩu tài khoản.

Làm bài thi: Người dùng có thể lựa chọn số câu hỏi muốn làm và bắt đầu làm, làm bài thi theo thời gian. Người dùng sẽ được hiển thị câu hỏi và lựa chọn đáp án tương ứng. Khi hoàn thành, người dùng có thể nộp bài và xem kết quả.

* Chức năng của quản trị viên (admin)

Quản lý người dùng: quản trị viên có thể thêm, sửa, xóa, tạo tài khoản và cung cấp quyền truy cập cho tài khoản.

Quản lý bài thi: Quản trị viên có thể thêm, sửa, xóa, tạo câu hỏi và lựa chọn đáp án cho mỗi câu hỏi.

### 2.2.2. Yêu cầu phi chức năng

- Giao diện thân thiện: Giao diện dễ sử dụng, thân thiện với người dùng.

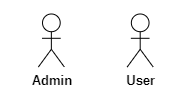
- Bảo mật: Bảo mật thông tin người dùng, đảm bảo chỉ có người dùng đã đăng nhập mới có thể truy cập các chức năng.

- Hiệu suất: Ứng dụng phải có hiệu suất cao, đáp ứng nhanh chóng các yêu cầu của người dùng.

## 2.2. Phân tích yêu cầu của hệ thống

### 2.2.1. Tác nhân của hệ thống

Các tác nhân của hệ thống gồm có:

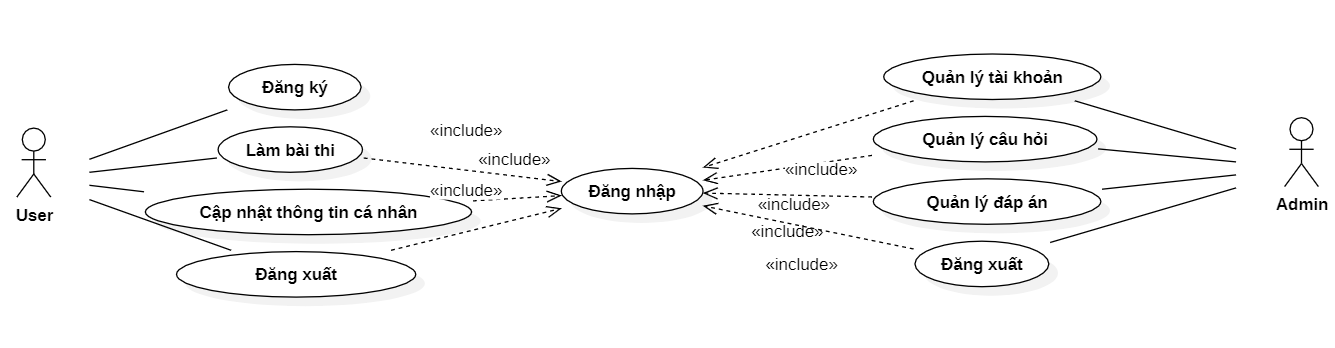


Hình 2. 1 Các tác nhân của hệ thống

Admin: người quản lýlà thành viên quản trị của hệ thống, có các quyền và chức năng như: tạo các tài khoản, quản trị người dùng, tạo câu hỏi và đáp án tương ứng.

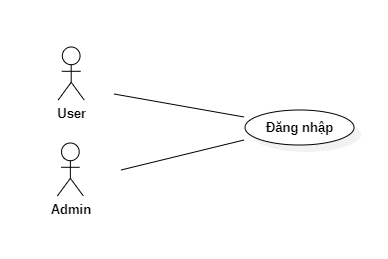
User: là người dùng có chức năng: làm bài thi trắc nghiệm, xem kết quả và đánh giá sau khi làm bài, xem thông tin cá nhân (tên, email,…), thay đổi mật khẩu hoặc thông tin cá nhân.

### 2.2.2. Biểu đồ use case tổng quát của hệ thống



Hình 2. 2 Biểu đồ use case tổng quát của hệ thống

### 2.2.3. Biểu đồ phân rã chức năng đăng nhập



Hình 2. 3 Biểu đồ phân rã chức năng đăng nhập

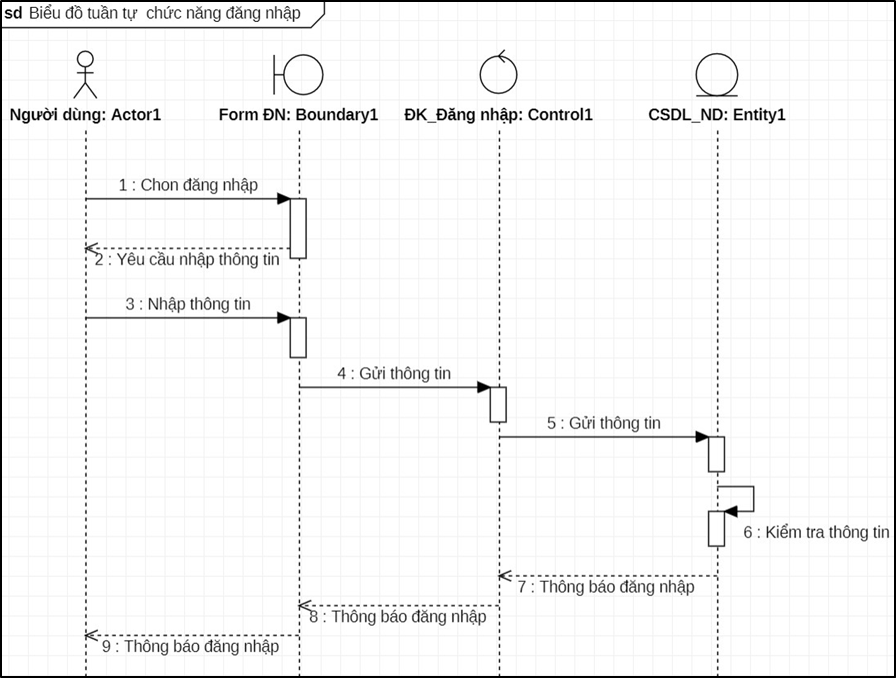
#### Kịch bản use case chức năng đăng nhập

Bảng 2. 1 Kịch bản use case chức năng đăng nhập

|  |  |
| --- | --- |
| Tên use case | Đăng nhập |
| Tác nhân | Người dùng, người quản lý |
| Mô tả | Người dùng và quản lý cần phải đăng nhập để sử dụng chức năng của hệ thống. |
| Kích hoạt | Người dùng và quản lý thực hiện đăng nhập, nhấn nút đăng nhập. |
| Tiền điều kiện | - Tài khoản đã được cấp phép sử dụng.  - Tài khoản đã được xác minh. |
| Hậu điều kiện | - Đăng nhập thất bại sẽ xuất hiện thông báo đăng nhập thất bại và yêu cầu đăng nhập lại.  - Đăng nhập thành công sẽ truy cập được vào hệ thống. |

#### Biểu đồ tuần tự chức năng đăng nhập

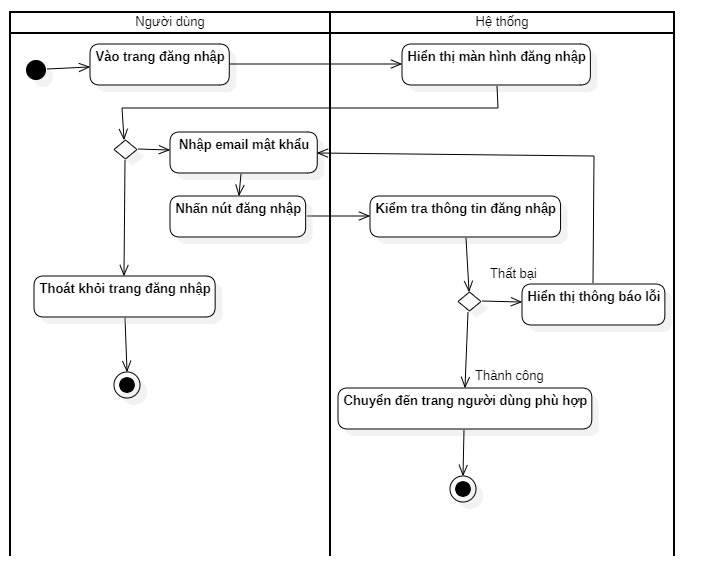
Người dùng chọn đăng nhập. Giao diện đăng nhập yêu cầu nhập thông tin. Người dùng nhập thông tin đăng nhập. Gửi thông tin. Hệ thống kiểm tra thông tin đăng nhập và thông báo kết quả đăng nhập với người dùng.



Hình 2. 4 Biểu đồ tuần tự chức năng đăng nhập

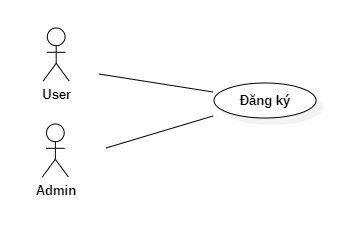
#### Biểu đồ hoạt động quy trình đăng nhập

Người dùng vào trang đăng nhập, hệ thống hiển thị màn hình đăng nhập, người dùng có thể thực hiện nhập email, mật khẩu để đăng nhập hoặc thoát khỏi trang đăng nhập. Khi người dùng nhập đầy đủ thông tin và nhấn nút đăng nhập, hệ thống kiểm tra thông tin này trong cơ sở dữ liệu. Nếu thông tin đúng, hệ thống cho phép đăng nhập và người dùng được điều hướng đến trang phù hợp. Nếu không, hệ thống báo lỗi và người dùng có thể nhập lại hoặc hủy thao tác.



Hình 2. 5 Biểu đồ hoạt động quy trình đăng nhập

### 2.2.4. Biểu đồ phân rã chức năng đăng ký



Hình 2. 6 Biểu đồ phân rã chức năng đăng ký

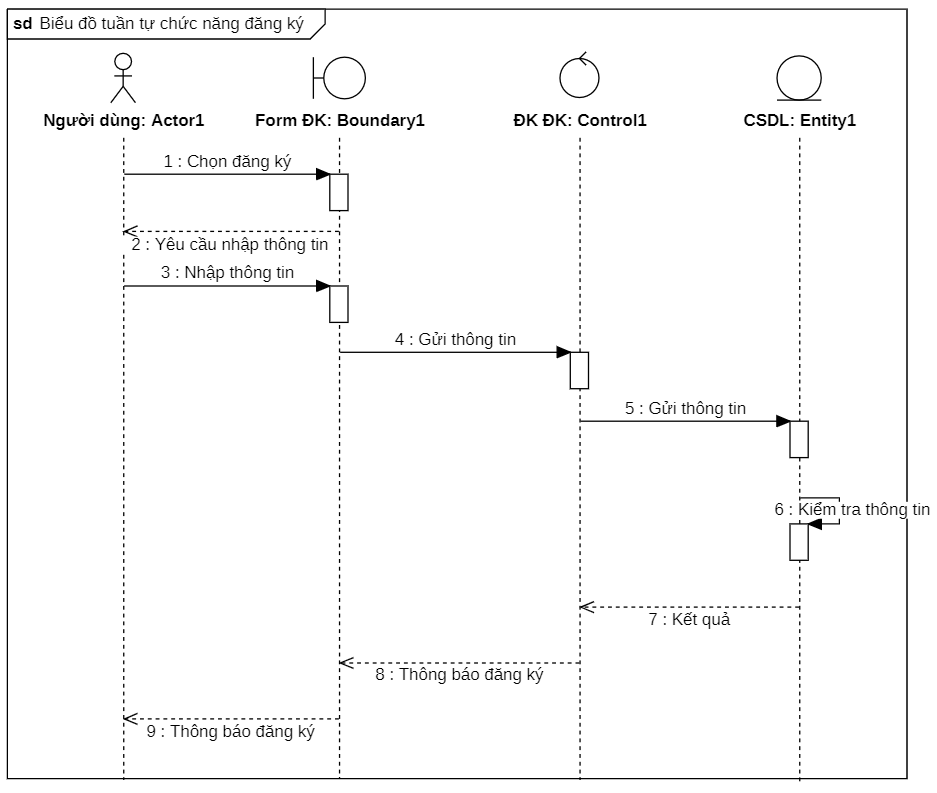
#### Kịch bản use case chức năng đăng ký

Bảng 2. 2 Kịch bản use case chức năng đăng ký

|  |  |
| --- | --- |
| Tên use case | Đăng ký |
| Tác nhân | Người dùng, người quản lý |
| Mô tả | Người dùng và quản lý cần phải đăng ký để đăng nhập và sử dụng chức năng của hệ thống |
| Kích hoạt | Người dùng và quản lý thực hiện đăng ký, nhấn nút đăng ký. |
| Tiền điều kiện | - Tài khoản đã được cấp phép sử dụng.  - Tài khoản đã được xác minh. |
| Hậu điều kiện | - Đăng ký thất bại sẽ xuất hiện thông báo đăng ký thất bại và yêu cầu đăng ký lại.  - Đăng ký thành công sẽ tạo một tài khoản để đăng nhập nhập và hệ thống. |

#### Biểu đồ tuần tự chức năng đăng ký

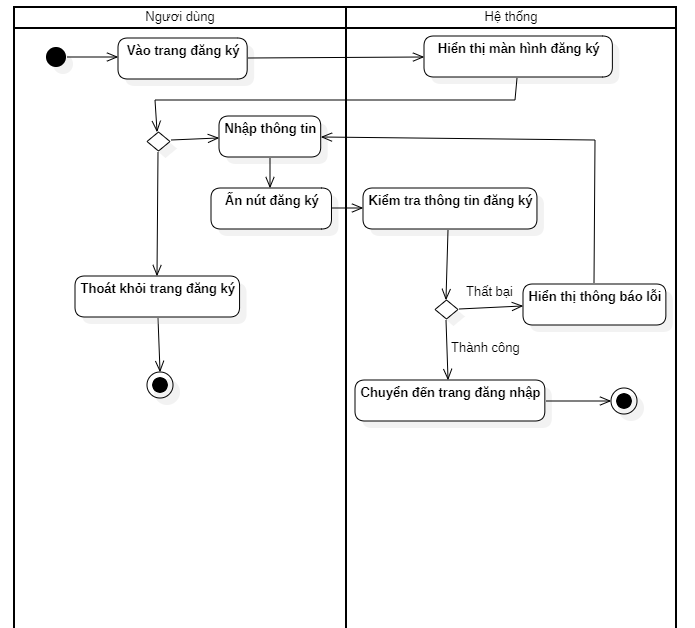
Người dùng chọn đăng ký. Giao diện đăng ký yêu cầu nhập thông tin. Người dùng nhập thông tin đăng ký. Gửi thông tin. Hệ thống kiểm tra thông tin đăng ký và thông báo kết quả đăng nhập với người dùng.



Hình 2. 7 Biểu đồ tuần tự chức năng đăng ký

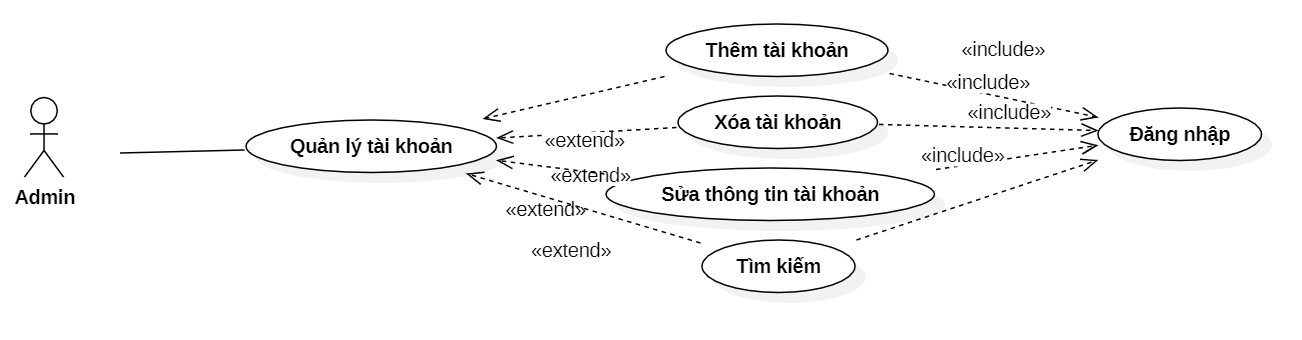
#### Biểu đồ hoạt động quy trình đăng ký

Người dùng vào trang đăng ký, hệ thống hiển thị màn hình đăng ký, người dùng có thể thực hiện nhập email, tên người dùng, mật khẩu và xác nhận mật khẩu để đăng ký tài khoản hoặc thoát khỏi trang đăng ký. Khi người dùng nhập đầy đủ thông tin và nhấn nút đăng ký, hệ thống kiểm tra thông tin này trong cơ sở dữ liệu. Nếu thông tin đúng, hệ thống cho phép đăng ký và người dùng được điều hướng đến trang đăng nhập. Nếu thông tin không hợp lệ, hệ thống yêu cầu người dùng chỉnh sửa.



Hình 2. 8 Biểu đồ hoạt động quy trình đăng ký

### 2.2.5. Biểu đồ phân rã chức năng quản lý tài khoản



Hình 2. 9 Biểu đồ phân rã chức năng quản lý tài khoản

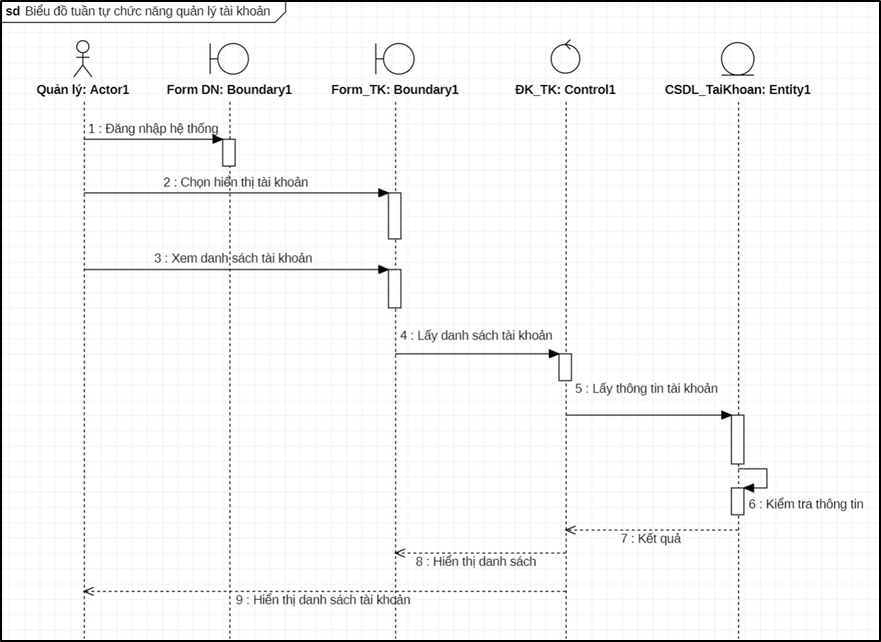
#### Kịch bản use case chức năng quản lý tài khoản

Bảng 2. 3 Kịch bản use case chức năng quản lý tài khoản

|  |  |
| --- | --- |
| Tên use case | Quản lý tài khoản |
| Tác nhân | Người quản lý |
| Mô tả | Quản lý sử dụng chức năng quản lý tài khoản để thực hiện các chức năng (thêm, sửa, xóa) tài khoản. |
| Kích hoạt | Quản lý thực hiện quản lý tài khoản, phải nhấn nút sửa, xóa. |
| Tiền điều kiện | Quản lý điền đầy đủ thông tin chính xác của tài khoản mà mình muốn thêm trước khi nhấn nút sửa. |
| Hậu điều kiện | - Tên đăng nhập không được phép trùng.  - Nếu thất bại, thông báo thất bại và yêu cầu nhập lại thông tin. |

#### Biểu đồ tuần tự chức năng quản lý tài khoản

Người quản lý đã đăng nhập vào hệ thống. Chọn menu chức năng quản lý tài khoản trong trang quản trị và chọn chức năng hiển thị thông tin tài khoản. Giao diện quản lý tài khoản hiển thị. Lấy danh sách thông tin tài khoản. Hệ thống kiểm tra thông tin tài khoản và thông báo kết quả với người quản lý.

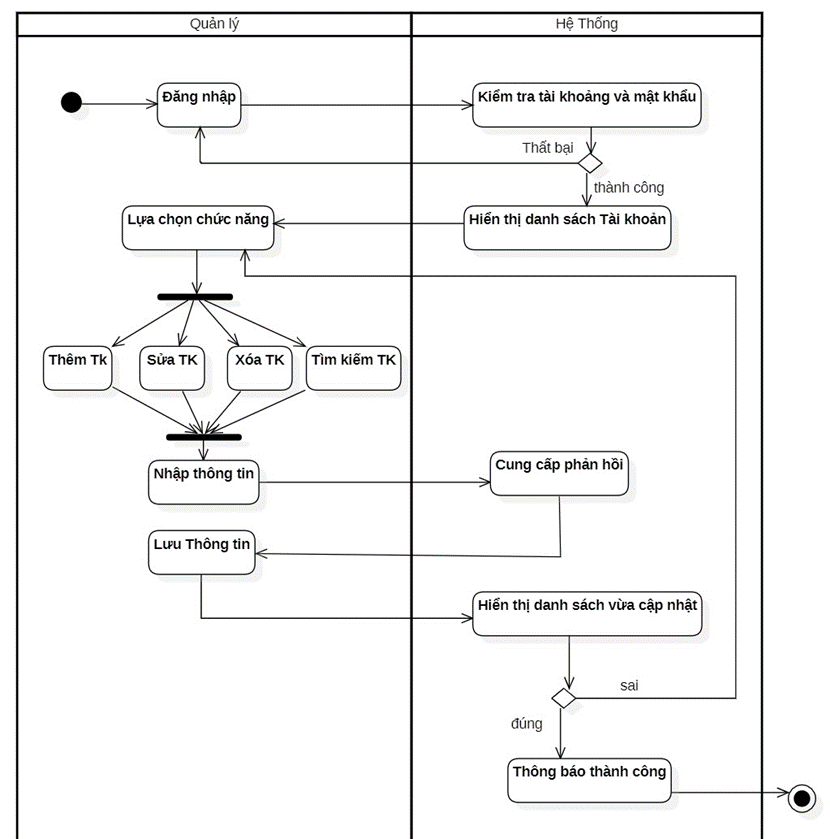


Hình 2. 10 Biểu đồ tuần tự chức năng quản lý tài khoản

#### Biểu đồ hoạt động quy trình quản lý tài khoản

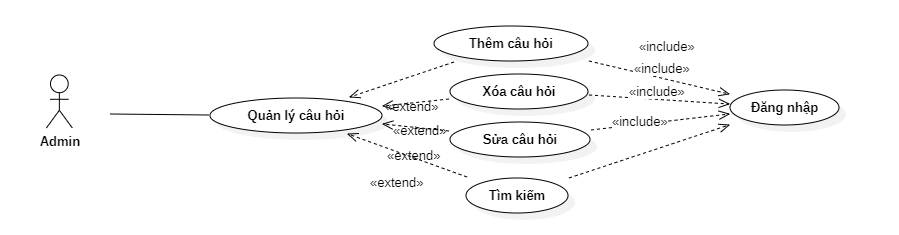
Quản lý đăng nhập vào hệ thống. Hệ thống kiểm tra thông tin tài khoản, nếu thất bại quản lý phải đăng nhập lại và ngược lại thành công hệ thống sẽ hiển thị danh sách tài khoản. Từ đó, quản lý lựa chọn các chức năng: thêm tài khoản, sửa tài khoản, xóa tài khoản, tìm kiếm. Sau khi lựa chọn xong, quản lý nhập thông tin tài khoản. Hệ thống cung cấp thông tin phản hồi. Quản lý lưu thông tin. Hệ thống hiển thị danh sách tài khoản mới cập nhật.

Nếu đúng, hệ thống đưa ra thông báo thành công, sai thì yêu cầu chọn lại chức năng của quản lý tài khoản.



Hình 2. 11 Biểu đồ hoạt động quy trình quản lý tài khoản

### 2.2.6. Biểu đồ phân rã chức năng quản lý câu hỏi



Hình 2. 12 Biểu đồ phân rã chức năng quản lý câu hỏi

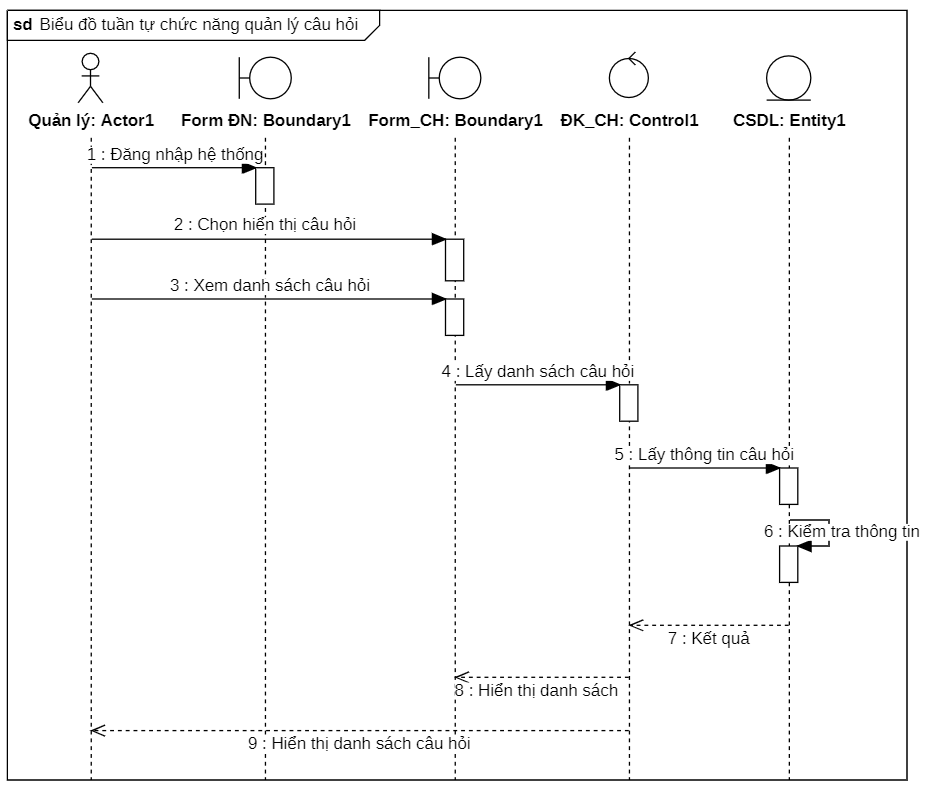
#### Kịch bản use case chức năng quản lý câu hỏi

Bảng 2. 4 Kịch bản use case chức năng quản lý câu hỏi

|  |  |
| --- | --- |
| Tên use case | Quản lý câu hỏi |
| Tác nhân | Người quản lý |
| Mô tả | Quản lý sử dụng chức năng quản lý câu hỏi để thực hiện các chức năng (thêm, sửa, xóa) câu hỏi. |
| Kích hoạt | Quản lý thực hiện quản lý câu hỏi, phải nhấn nút thêm, sửa, xóa. |
| Tiền điều kiện | Quản lý điền đầy đủ thông tin chính xác của câu hỏi mà mình muốn thêm trước khi nhấn nút thêm, sửa và xác nhận lại khi nhấn xóa. |
| Hậu điều kiện | - Tên câu hỏi không được phép trùng.  - Nếu thất bại, thông báo thất bại và yêu cầu nhập lại thông tin. |

#### Biểu đồ tuần tự chức năng quản lý câu hỏi

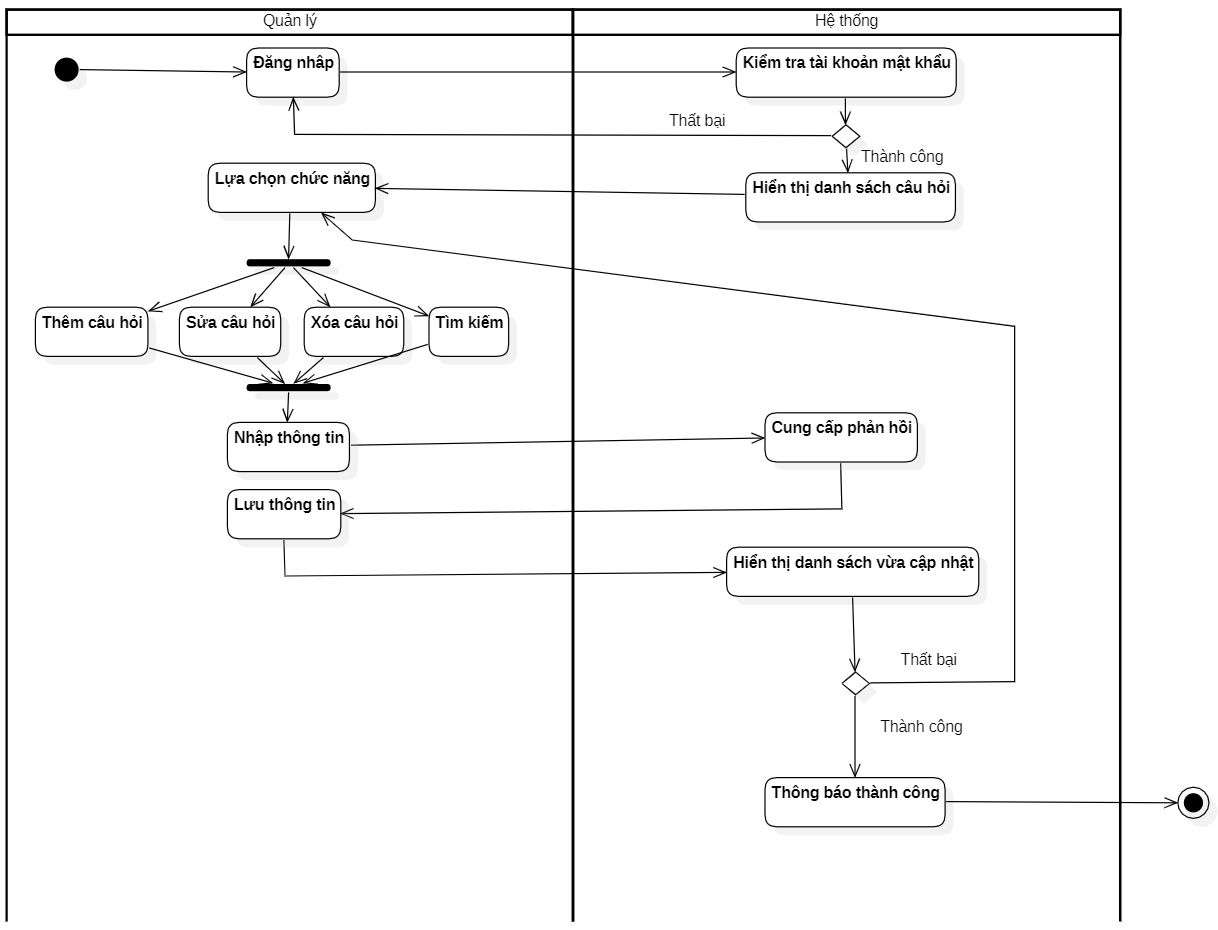
Người quản lý đã đăng nhập vào hệ thống. Sau đó, người quản lý chọn chức năng quản lý câu hỏi trong trang quản lý. Hệ thống nhận yêu cầu và trả về giao diện quản lý câu hỏi. Người quản lý lấy danh sách thông tin câu hỏi từ hệ thống. Hệ thống tiến hành kiểm tra thông tin câu hỏi và sau đó thông báo kết quả cho người quản lý.



Hình 2. 13 Biểu đồ tuần tự chức năng quản lý câu hỏi

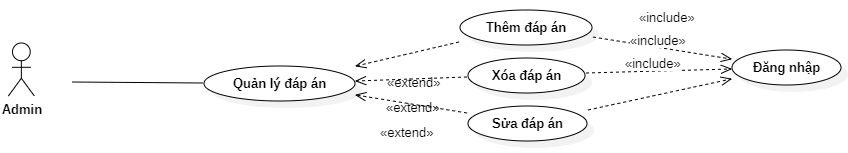
#### Biểu đồ hoạt động quy trình quản lý câu hỏi

Quản lý đăng nhập vào hệ thống. Hệ thống kiểm tra thông tin tài khoản, nếu đúng, hiển thị danh sách câu hỏi. Quản lý chọn chức năng từ danh sách, bao gồm thêm, sửa, xóa câu hỏi. Sau khi chọn, quản lý nhập thông tin tương ứng. Hệ thống xử lý thông tin và cung cấp phản hồi. Nếu thêm, sửa, hoặc xóa thành công, hệ thống thông báo và hiển thị danh sách câu hỏi mới. Nếu không, yêu cầu quản lý chọn lại chức năng.



Hình 2. 14 Biểu đồ hoạt động quy trình quản lý câu hỏi

### 2.2.7. Biểu đồ phân rã chức năng quản lý đáp án



Hình 2. 15 Biểu đồ phân rã chức năng quản lý đáp án

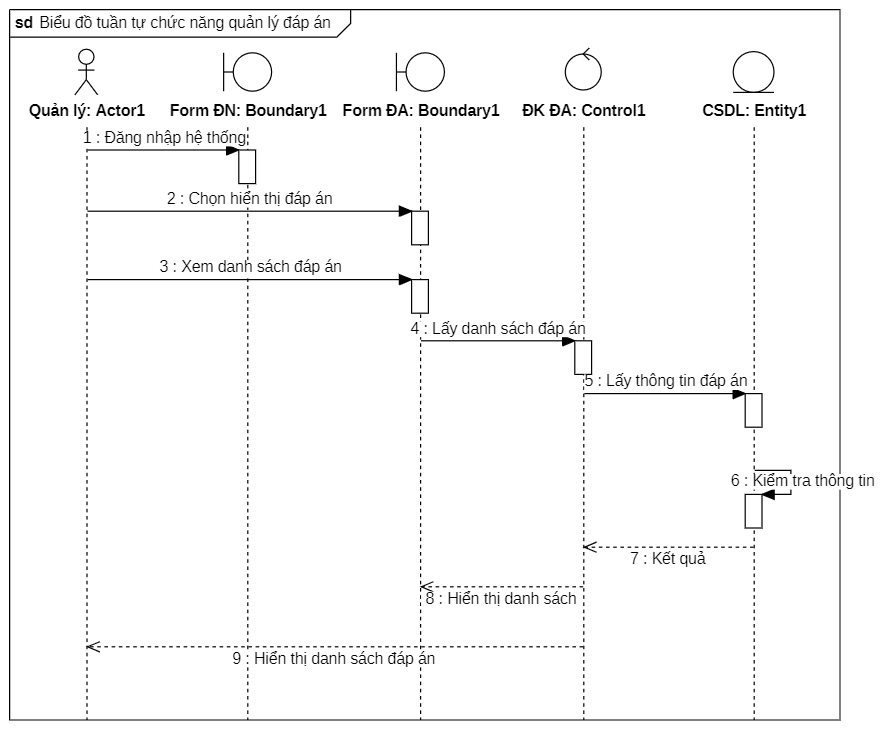
#### Kịch bản use case chức năng quản lý đáp án

Bảng 2. 5 Kịch bản use case chức năng quản lý đáp án

|  |  |
| --- | --- |
| Tên use case | Quản lý đáp án |
| Tác nhân | Người quản lý |
| Mô tả | Quản lý sử dụng chức năng quản lý đáp án để thực hiện các chức năng (thêm, sửa, xóa) đáp án. |
| Kích hoạt | Quản lý thực hiện quản lý đáp án, phải nhấn nút thêm, sửa, xóa. |
| Tiền điều kiện | Quản lý điền đầy đủ thông tin chính xác của đáp án mà mình muốn thêm trước khi nhấn nút thêm, sửa và xác nhận lại khi nhấn xóa. |
| Hậu điều kiện | Nếu thất bại, thông báo thất bại và yêu cầu nhập lại thông tin. |

#### Biểu đồ tuần tự chức năng quản lý đáp án

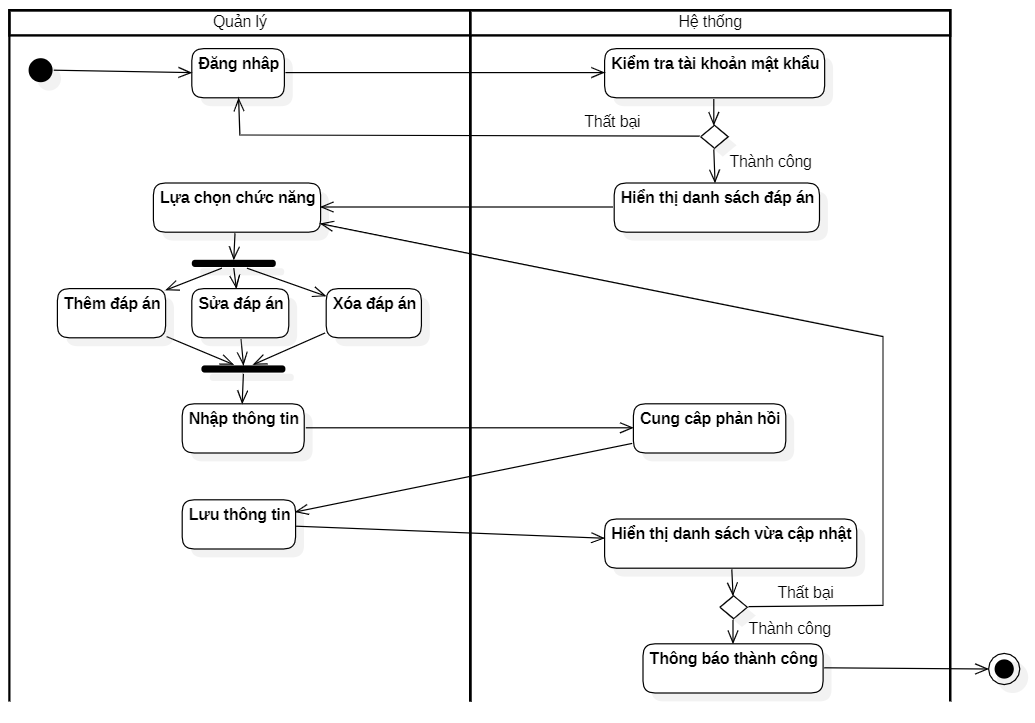
Người quản lý đã đăng nhập vào hệ thống. Sau đó, người quản lý chọn chức năng quản lý đáp án trong trang quản lý. Hệ thống nhận yêu cầu và trả về giao diện quản lý đáp án. Người quản lý lấy danh sách thông tin đáp án từ hệ thống. Hệ thống tiến hành kiểm tra thông tin đáp án và sau đó thông báo kết quả cho người quản lý.



Hình 2. 16 Biểu đồ tuần tự chức năng quản lý đáp án

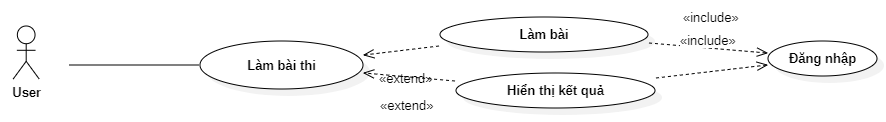
#### Biểu đồ hoạt động quy trình quản lý đáp án

Quản lý đăng nhập vào hệ thống. Hệ thống kiểm tra thông tin tài khoản, nếu đúng, hiển thị danh sách câu hỏi. Quản lý chọn câu hỏi cần quản lý đáp án. Sau đó, quản lý chọn chức năng từ danh sách, bao gồm thêm, sửa, xóa đáp án. Sau khi chọn, quản lý nhập thông tin tương ứng. Hệ thống xử lý thông tin và cung cấp phản hồi. Nếu thêm, sửa, hoặc xóa thành công, hệ thống thông báo và hiển thị danh sách đáp án mới. Nếu không, yêu cầu quản lý chọn lại chức năng.



Hình 2. 17 Biểu đồ hoạt động quy trình quản lý đáp án

### 2.2.8. Biểu đồ phân rã chức năng làm bài



Hình 2. 18 Biểu đồ phân rã chức năng làm bài

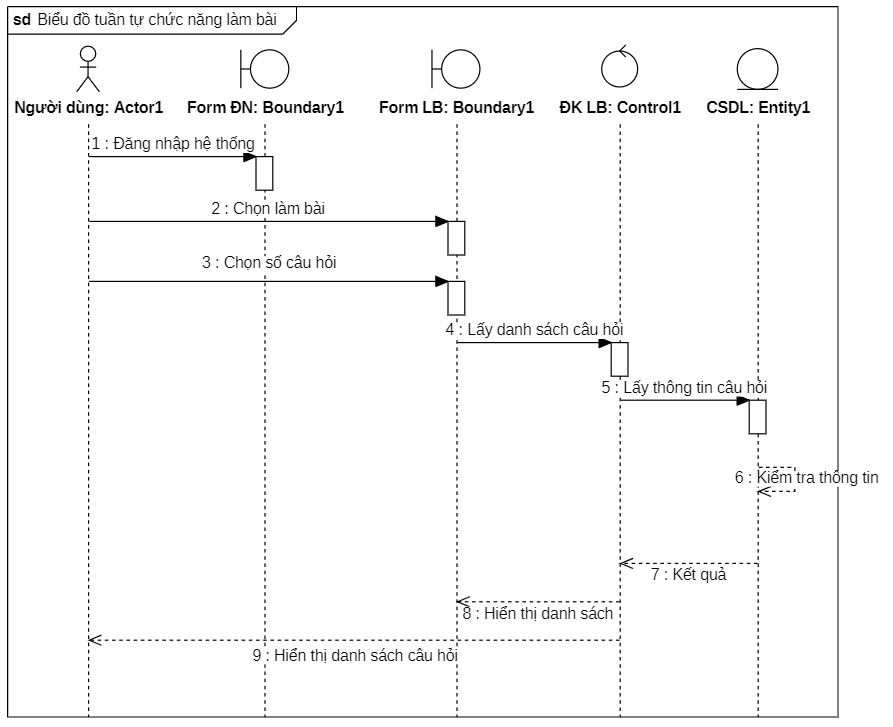
#### Kịch bản use case chức năng làm bài

Bảng 2. 6 Kịch bản use case chức năng làm bài

|  |  |
| --- | --- |
| Tên use case | Làm bài thi |
| Tác nhân | Người dùng |
| Mô tả | Người dùng sử dụng chức làm bài thi để thực hiện trả lời câu hỏi. |
| Kích hoạt | Người dùng thực hiện làm bài. |
| Tiền điều kiện | Người dùng đã đăng nhập vào hệ thống và chọn bài thi. |
| Hậu điều kiện | Kết quả bài thi được lưu lại và hiển thị cho người dùng. |

#### Biểu đồ tuần tự chức năng làm bài

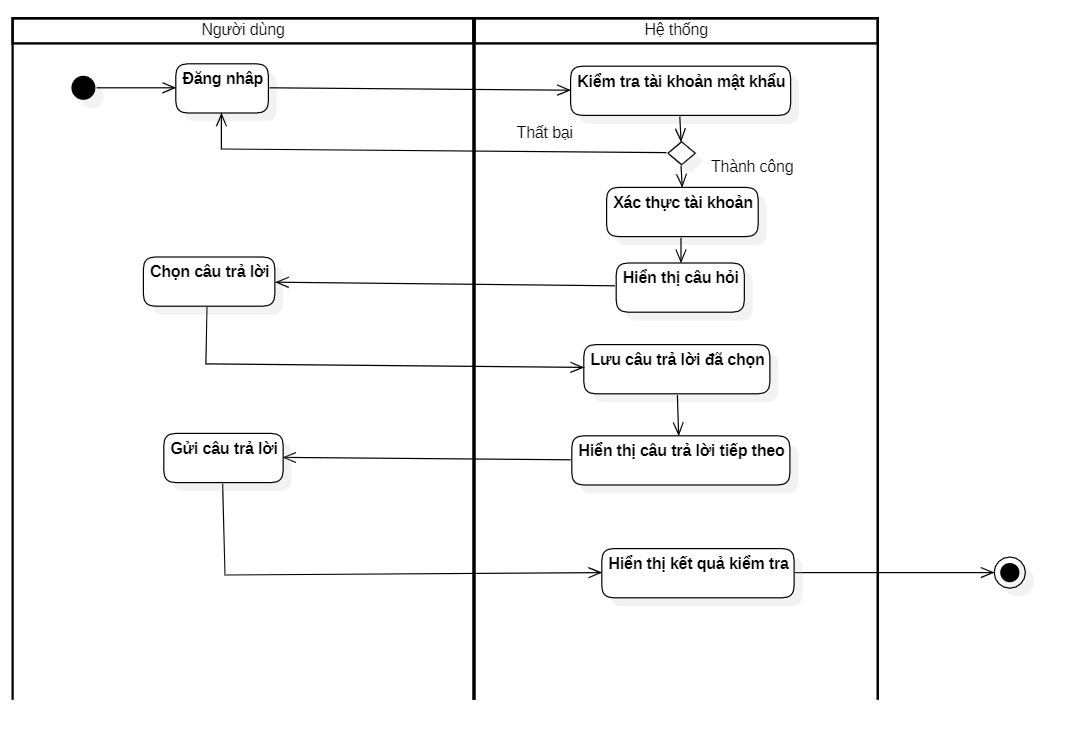
Biểu đồ tuần tự chức năng làm bài bắt đầu khi người dùng chọn bài thi để tham gia. Hệ thống hiển thị câu hỏi đầu tiên cùng các lựa chọn. Người dùng sau đó chọn câu trả lời và tiếp tục chuyển đến câu hỏi tiếp theo. Quá trình này lặp lại cho đến khi người dùng hoàn thành bài thi hoặc hết thời gian. Khi bài thi kết thúc, hệ thống tự động tính điểm và hiển thị kết quả cho người dùng. Kết quả bao gồm số điểm đạt được và chi tiết về các câu trả lời đúng và sai. Người dùng có thể xem lại các câu hỏi và câu trả lời của mình sau khi kết thúc bài thi.



Hình 2. 19 Biểu đồ tuần tự chức năng làm bài

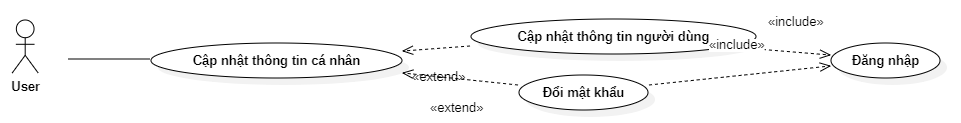
#### Biểu đồ hoạt động quy trình làm bài

Biểu đồ hoạt động quy trình làm bài bắt đầu khi người dùng bắt đầu bài thi. Hệ thống hiển thị câu hỏi đầu tiên và các lựa chọn cho người dùng. Người dùng chọn một trong các lựa chọn và gửi câu trả lời. Hệ thống kiểm tra câu trả lời và lưu lại kết quả. Sau đó, hệ thống di chuyển đến câu hỏi tiếp theo. Quá trình này lặp lại cho đến khi hết thời gian hoặc người dùng hoàn thành tất cả các câu hỏi. Khi bài thi kết thúc, hệ thống hiển thị kết quả và cung cấp các tùy chọn để xem lại câu hỏi và câu trả lời.



Hình 2. 20 Biểu đồ hoạt động quy trình làm bài

### 2.2.9. Biểu đồ phân rã chức năng cập nhật thông tin



Hình 2. 21 Biểu đồ phân rã chức năng cập nhật thông tin

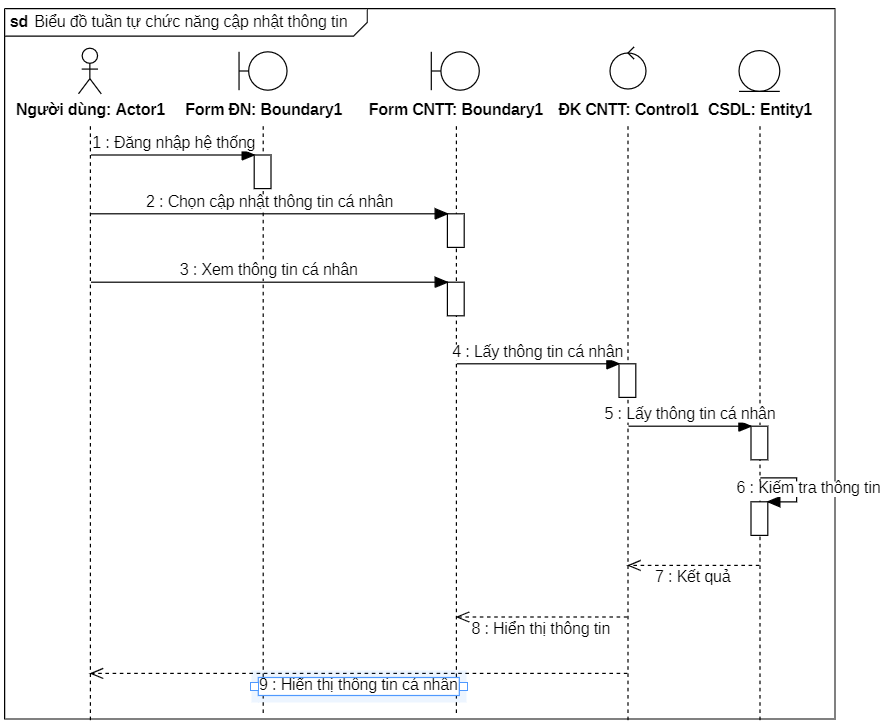
#### Kịch bản use case chức năng cập nhật thông tin

Bảng 2. 7 Kịch bản use case chức năng cập nhật thông tin

|  |  |
| --- | --- |
| Tên use case | Cập nhật thông tin |
| Tác nhân | Người dùng |
| Mô tả | Người dùng sử dụng chức năng cập nhật thông tin để thực hiện các chức năng cập nhật thông tin người dùng và đổi mật khẩu. |
| Kích hoạt | Người dùng thực hiện cập nhật thông tin cá nhân. |
| Tiền điều kiện | Người dùng điền đầy đủ thông tin chính xác của tài khoản mà mình muốn sửa, nhập mật khẩu cũ và mật khẩu mới. |
| Hậu điều kiện | Nếu thất bại, thông báo thất bại và yêu cầu nhập lại thông tin. |

#### Biểu đồ tuần tự chức năng cập nhật thông tin

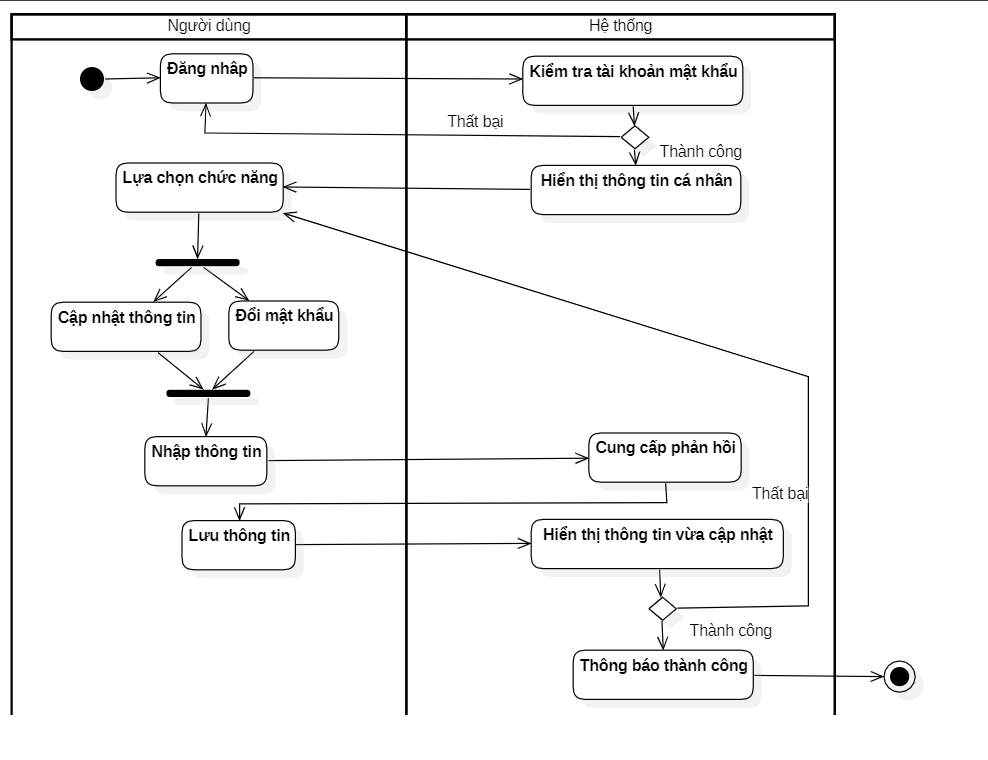
Biểu đồ tuần tự chức năng cập nhật thông tin cá nhân bắt đầu khi người dùng chọn chức năng cập nhật từ giao diện người dùng. Hệ thống sau đó hiển thị một form cho phép người dùng nhập thông tin mới. Người dùng nhập thông tin mới vào các trường tương ứng và sau đó xác nhận việc cập nhật. Hệ thống kiểm tra và xử lý thông tin mới từ người dùng. Nếu quá trình cập nhật thành công, hệ thống hiển thị thông báo xác nhận. Ngược lại, nếu có bất kỳ lỗi nào xảy ra trong quá trình cập nhật, hệ thống sẽ hiển thị một thông báo lỗi và yêu cầu người dùng nhập lại thông tin.



Hình 2. 22 Biểu đồ tuần tự chức năng cập nhật thông tin

#### Biểu đồ hoạt động quy trình cập nhật thông tin

Biểu đồ hoạt động chức năng cập nhật thông tin bắt đầu khi người dùng chọn chức năng cập nhật thông tin từ giao diện người dùng. Hệ thống sau đó hiển thị một form cho phép người dùng nhập thông tin mới. Người dùng nhập thông tin mới vào các trường tương ứng và sau đó gửi yêu cầu cập nhật. Hệ thống kiểm tra tính hợp lệ của thông tin và xác nhận. Nếu thông tin hợp lệ, hệ thống cập nhật dữ liệu và hiển thị thông báo xác nhận cho người dùng. Ngược lại, nếu có lỗi xảy ra, hệ thống hiển thị một thông báo lỗi và yêu cầu người dùng nhập lại thông tin.



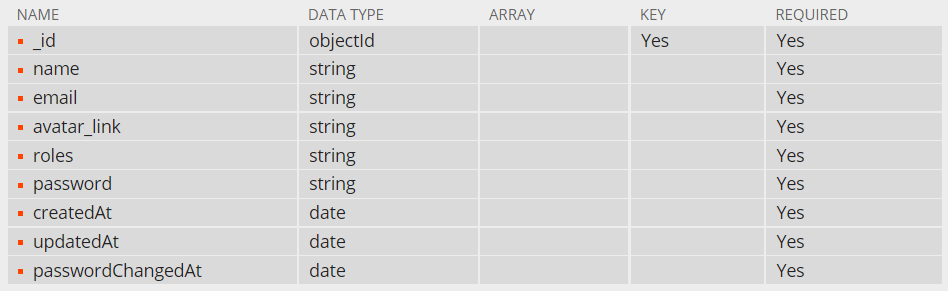
Hình 2. 23 Biểu đồ hoạt động quy trình cập nhật thông tin

## 2.3. Thiết kế cơ sở dữ liệu

### 2.3.1. Bảng người dùng

Bảng 2. 8 Bảng mô tả người dùng

|  |  |  |  |
| --- | --- | --- | --- |
| **Tên** | **Kiểu dữ liệu** | **Mô tả** | **Khóa** |
| \_id | obiectId | Mã người dùng | Khóa chính |
| name | string | Tên người dùng |  |
| email | string | Email người dùng |  |
| avatar\_link | string | Địa chỉ hình ảnh người dùng |  |
| roles | string | Quyền người dùng |  |
| password | string | Mật khẩu người dùng |  |
| createdAt | date | Ngày tạo |  |
| updatedAt | date | Ngày cập nhật |  |
| passwordChangedAt | date | Ngày cập nhật mật khẩu |  |

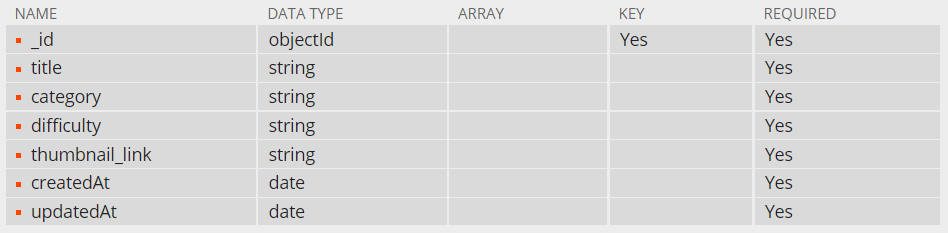


Hình 2. 24 Thiết kế bảng quản lý người dùng trên MongoDB

### 2.3.2. Bảng câu hỏi

Bảng 2. 9 Bảng mô tả câu hỏi

|  |  |  |  |
| --- | --- | --- | --- |
| **Tên** | **Kiểu dữ liệu** | **Mô tả** | **Khóa** |
| \_id | obiectId | Mã câu hỏi | Khóa chính |
| title | string | Nội dung câu hỏi |  |
| category | string | Loại câu hỏi |  |
| difficulty | string | Độ khó câu hỏi |  |
| thumbnail\_link | string | Địa chỉ hình ảnh câu hỏi |  |
| createdAt | date | Ngày tạo |  |
| updatedAt | date | Ngày cập nhật |  |

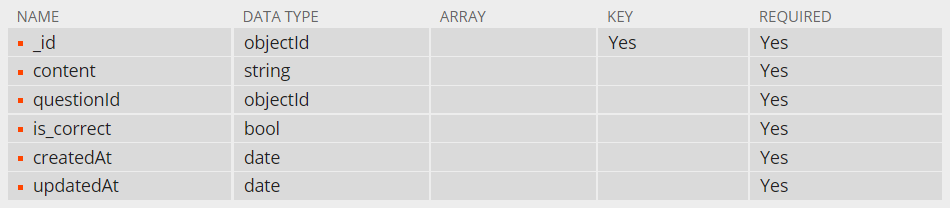


Hình 2. 25 Thiết kế bảng quản lý câu hỏi trên MongoDB

### 2.3.3. Bảng đáp án

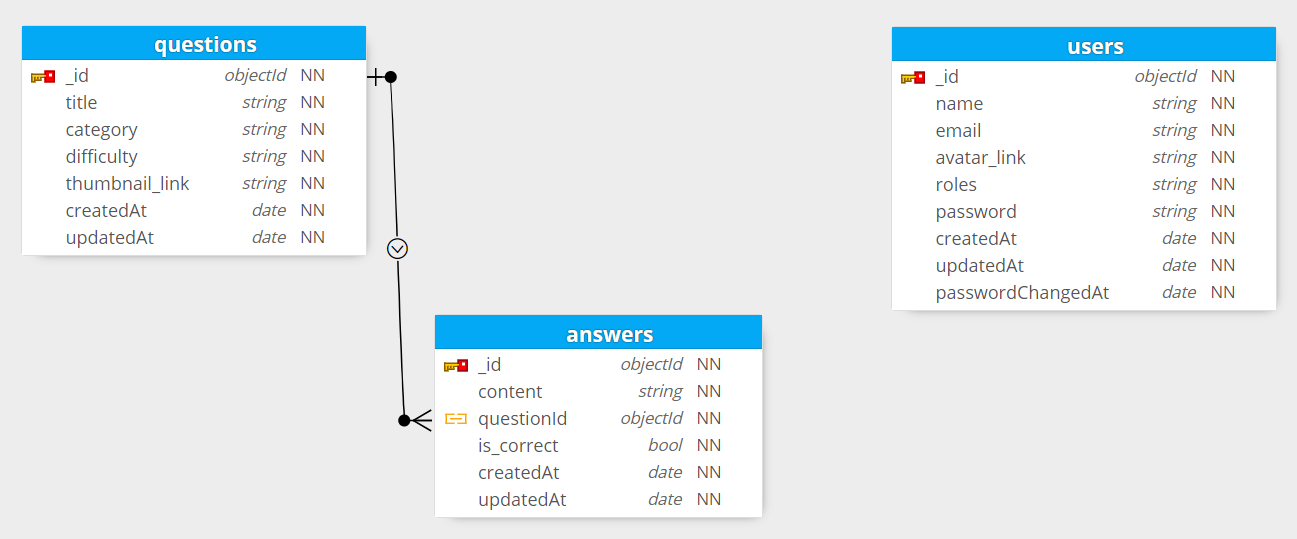
Bảng 2. 10 Bảng mô tả đáp án

|  |  |  |  |
| --- | --- | --- | --- |
| **Tên** | **Kiểu dữ liệu** | **Mô tả** | **Khóa** |
| \_id | obiectId | Mã câu hỏi | Khóa chính |
| content | string | Nội dung đáp án |  |
| questionId | obiectId | Mã câu hỏi | Khóa ngoại |
| is\_correct | boolean | Đáp án câu hỏi |  |
| createdAt | date | Ngày tạo |  |
| updatedAt | date | Ngày cập nhật |  |



Hình 2. 26 Thiết kế bảng quản lý đáp án trên MongoDB

### 2.3.4. Sơ đồ mối quan hệ thực thể



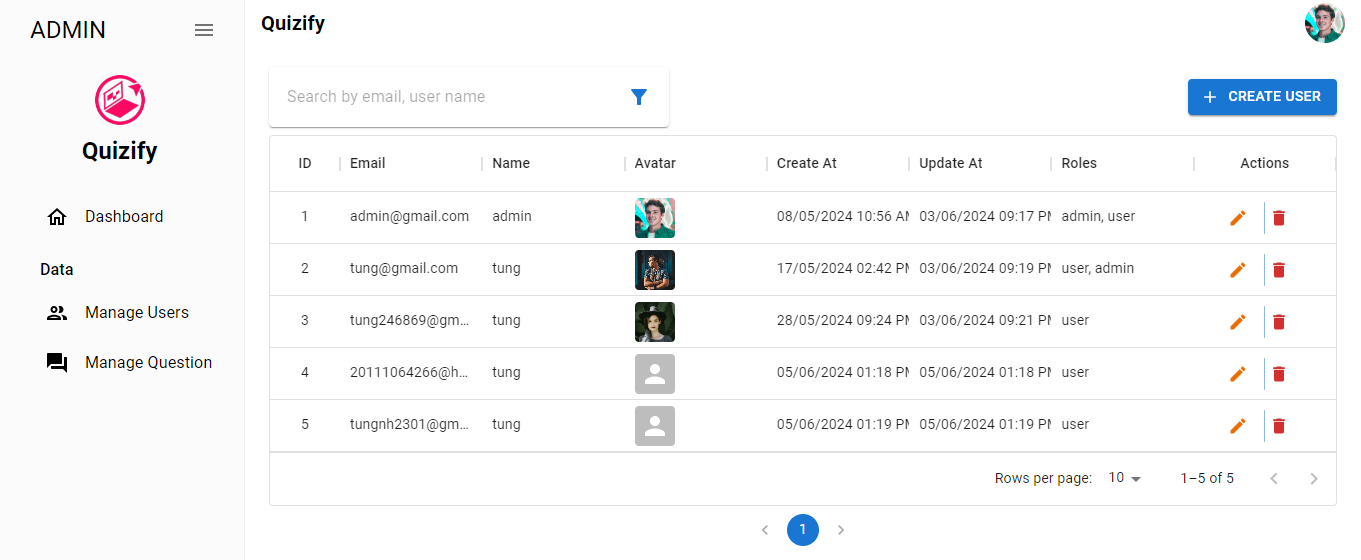
Hình 2. 27 Sơ đồ mối quan hệ thực thể

# CHƯƠNG 3: TRIỂN KHAI ỨNG DỤNG THI TRẮC NGHIỆM

## 3.1. Các chức năng phía người quản trị

### 3.1.1. Giao diện trang quản lý người dùng

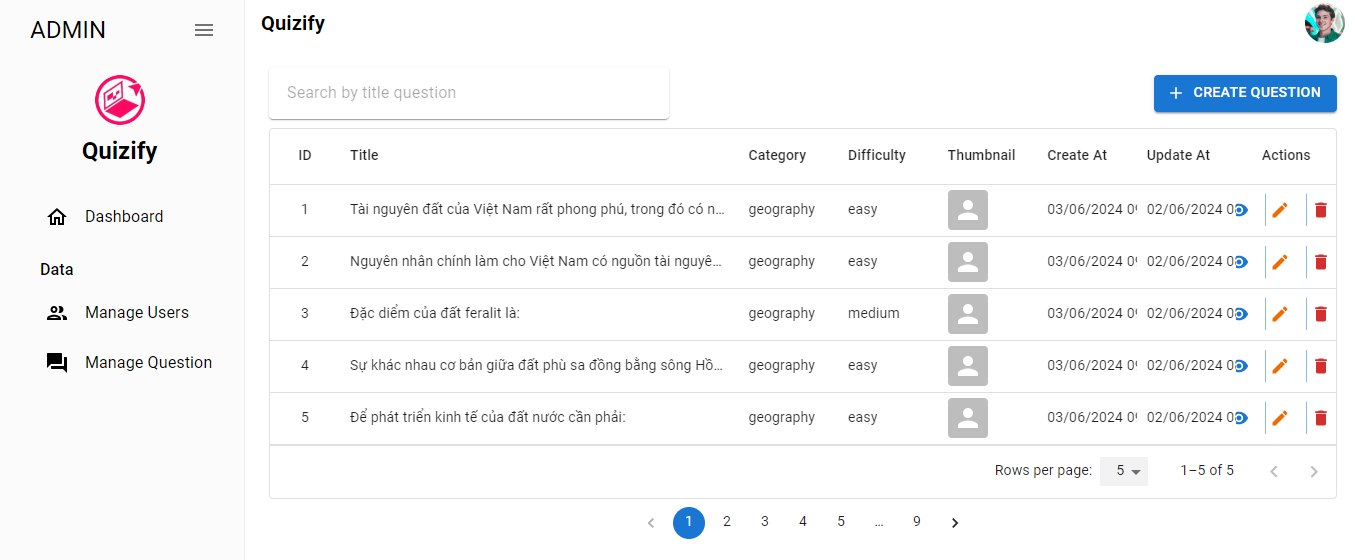
Giao diện trang quản lý người dùng là nơi mà người quản trị có thể thực hiện các hoạt động quản lý liên quan đến người dùng trong hệ thống. Chức năng chính của giao diện này là cung cấp cho người quản trị một công cụ để quản lý thông tin và quyền hạn của người dùng. Tại đây, người quản trị có thể xem danh sách người dùng, thêm người dùng mới, chỉnh sửa thông tin người dùng hiện tại, và xóa người dùng không còn hoạt động. Giao diện cũng cho phép người quản trị thiết lập và điều chỉnh quyền truy cập của từng người dùng, đảm bảo rằng mỗi người dùng chỉ có thể truy cập vào các chức năng và dữ liệu phù hợp với vai trò của họ trong hệ thống. Điều này giúp duy trì an ninh và hiệu quả trong việc quản lý người dùng.



Hình 3. 1 Giao diện trang quản lý người dùng

### 3.1.2. Giao diện trang quản lý câu hỏi

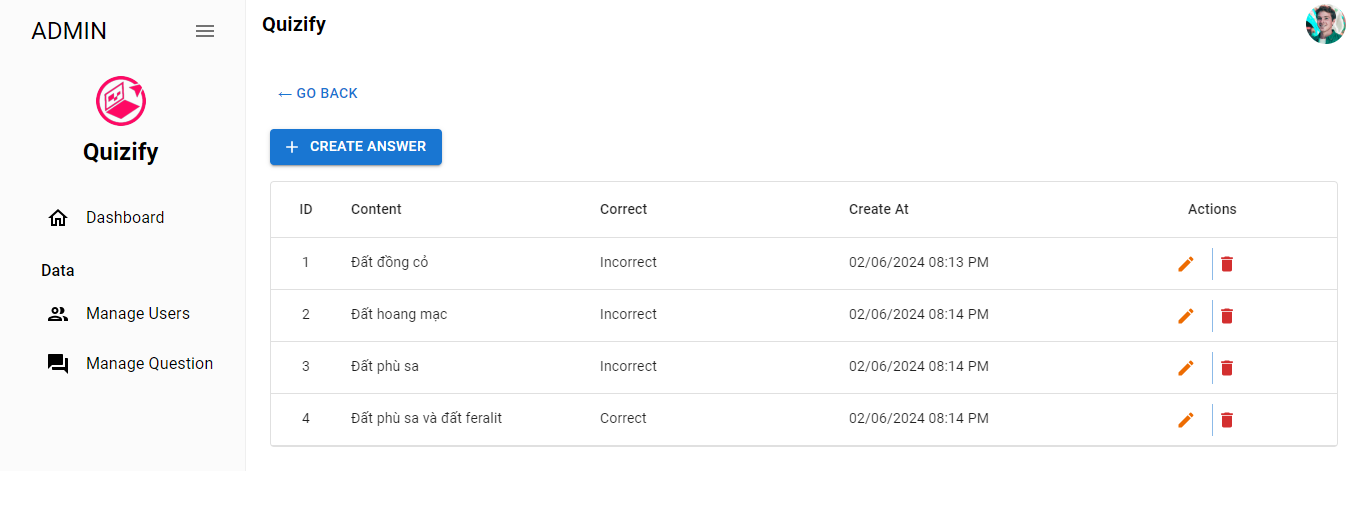
Giao diện trang quản lý câu hỏi là nơi mà người quản trị có thể thực hiện các hoạt động quản lý liên quan đến các câu hỏi trong hệ thống thi trắc nghiệm. Chức năng chính của giao diện này là cung cấp cho người quản trị một công cụ để quản lý, chỉnh sửa và thêm mới các câu hỏi. Tại đây, người quản trị có thể xem danh sách tất cả các câu hỏi, thêm câu hỏi mới, chỉnh sửa nội dung câu hỏi hiện có, và xóa các câu hỏi không còn phù hợp. Việc duy trì và cập nhật kho câu hỏi thường xuyên đảm bảo rằng hệ thống thi trắc nghiệm luôn có các câu hỏi mới mẻ và phù hợp với nhu cầu học tập và kiểm tra của người dùng.



Hình 3. 2 Giao diện trang quản lý câu hỏi

### 3.1.3. Giao diện trang quản lý đáp án

Giao diện trang quản lý đáp án là nơi mà người quản trị có thể thực hiện các hoạt động quản lý liên quan đến các đáp án trong hệ thống thi trắc nghiệm. Chức năng chính của giao diện này là cung cấp cho người quản trị một công cụ để quản lý, chỉnh sửa và thêm mới các đáp án cho từng câu hỏi. Tại đây, người quản trị có thể xem danh sách các đáp án tương ứng với mỗi câu hỏi, thêm đáp án mới, chỉnh sửa nội dung của các đáp án hiện có, và xóa các đáp án không còn phù hợp. Giao diện này cũng cho phép người quản trị thiết lập đáp án đúng cho mỗi câu hỏi, đảm bảo tính chính xác và khách quan trong quá trình chấm điểm.

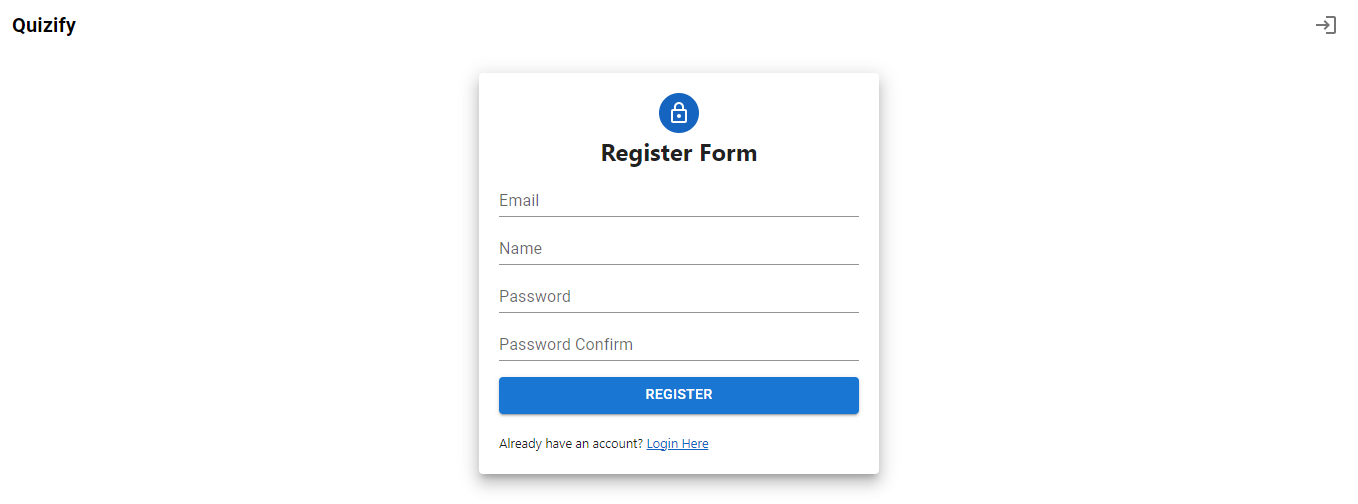


Hình 3. 3 Giao diện trang quản lý đáp án

## 3.2. Các chức năng phía người dùng

### 3.2.1. Giao diện trang đăng ký

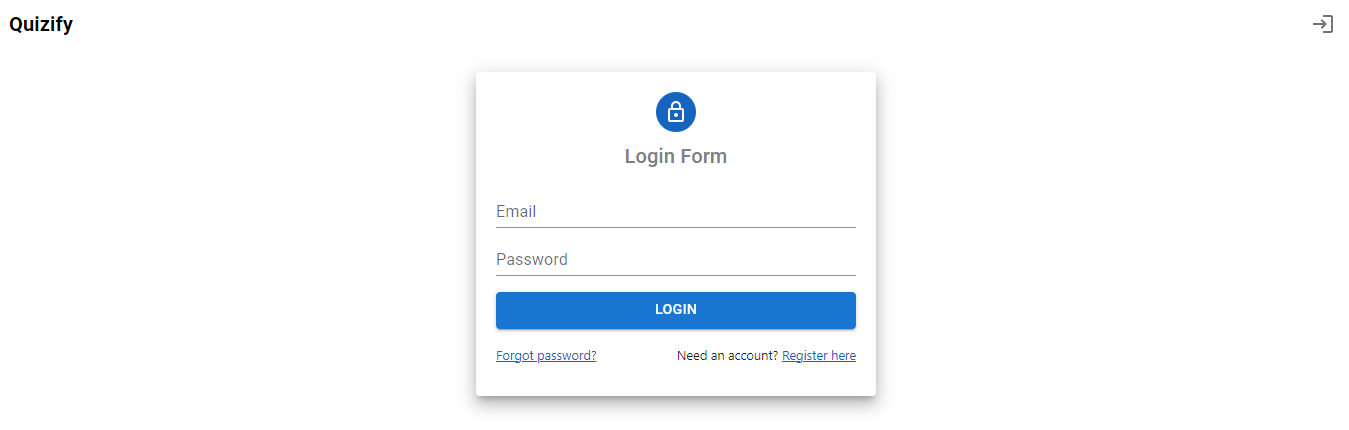
Giao diện trang đăng ký là nơi mà người dùng nhập thông tin đăng ký để tạo mới tài khoản. Tại đây người sử dụng đăng ký để xác nhận và thiết lập tài khoản của họ trong hệ thống.



Hình 3. 4 Giao diện trang đăng ký

### 3.2.2. Giao diện trang đăng nhập

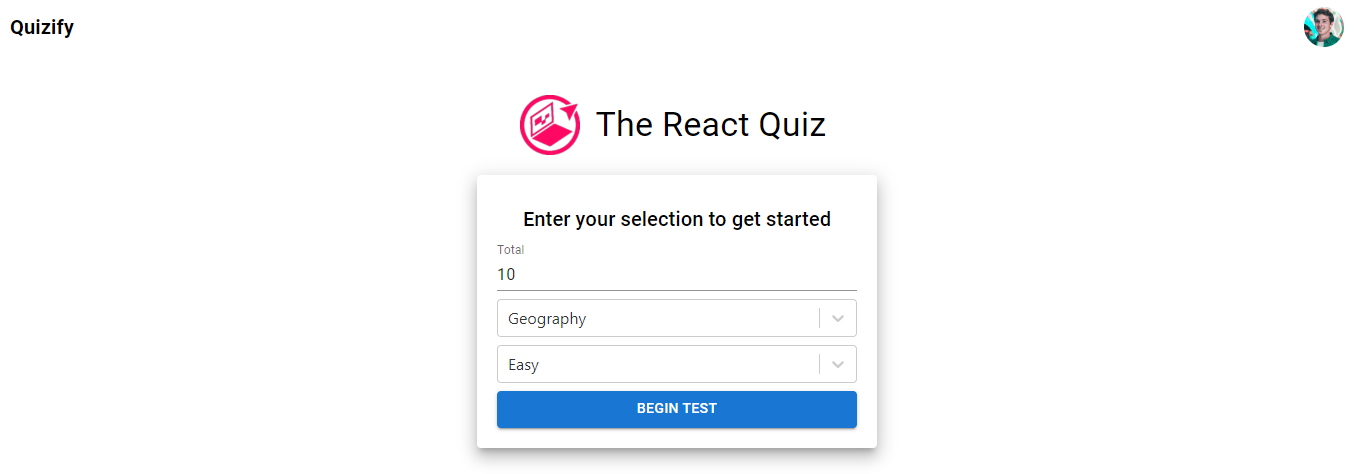
Giao diện trang đăng nhập là nơi mà người dùng nhập thông tin đăng nhập để truy cập vào hệ thống. Tại đây người sử dụng đăng nhập vào hệ thống để truy cập các chức năng của ứng dụng.



Hình 3. 5 Giao diện trang đăng nhập

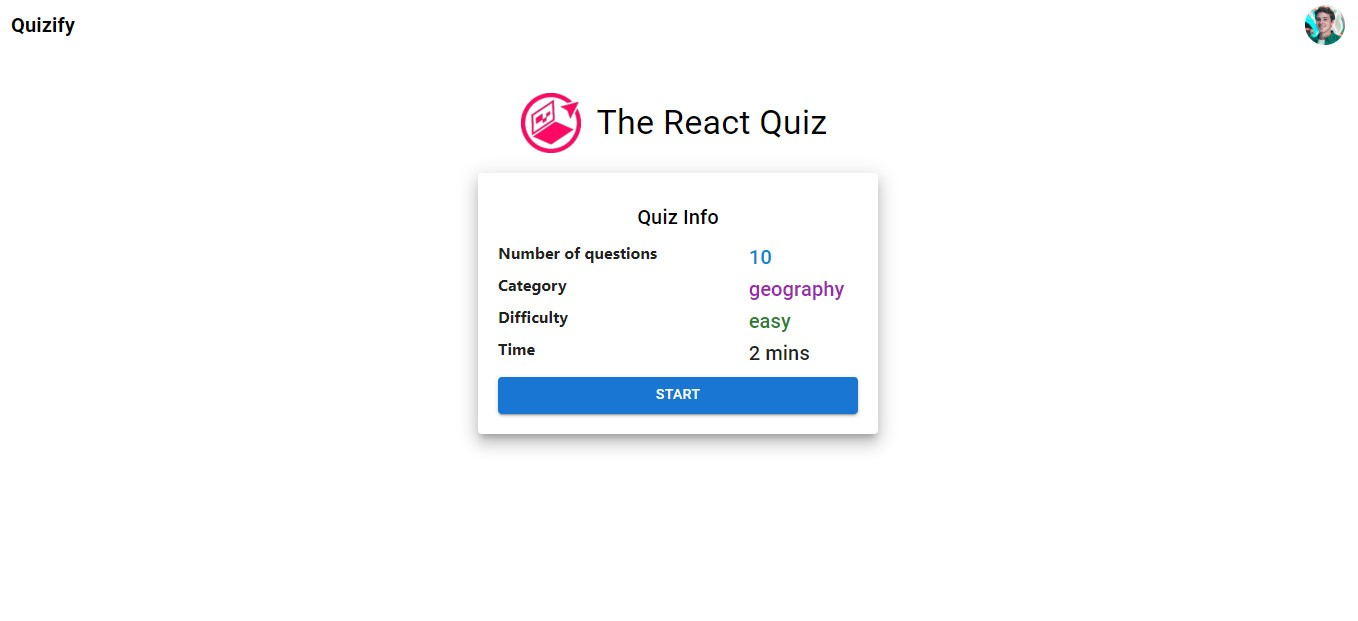
### 3.2.3. Giao diện trang làm bài

Giao diện trang làm bài trong ứng dụng là nơi mà người dùng thực hiện các câu hỏi trong bài thi trắc nghiệm. Chức năng chính của giao diện này là cung cấp một giao diện trực quan và dễ sử dụng để người dùng có thể làm bài thi một cách thuận tiện và hiệu quả. Người dùng chọn số, lĩnh vực và độ khó câu hỏi muốn làm để bắt đầu làm bài, câu hỏi sẽ được lấy "ngẫu nhiên" trong danh sách câu hỏi.



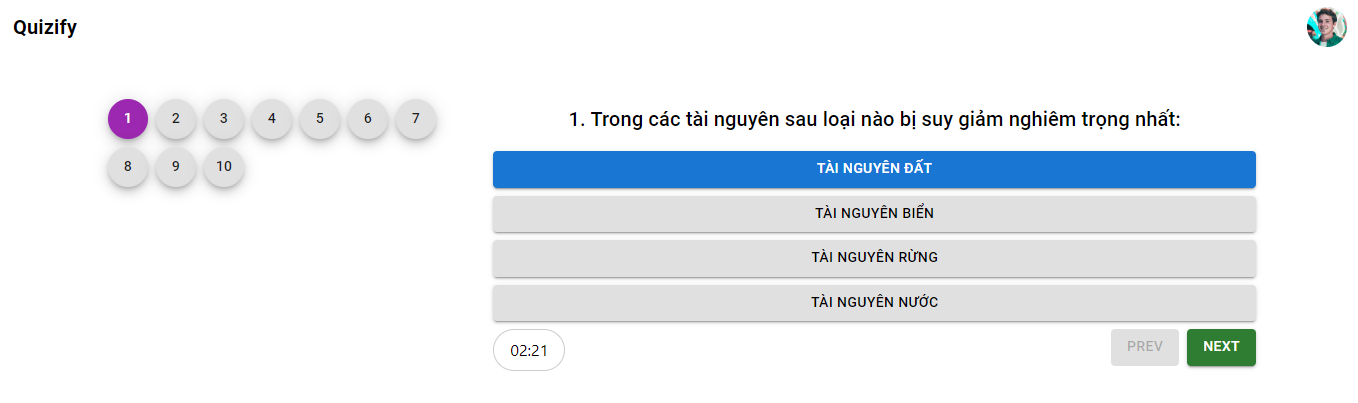
Hình 3. 6 Giao diện trang làm bài

Sau khi bắt đầu làm bài, hệ thống sẽ hiển thị tổng số câu hỏi, lĩnh vực, độ khó và thời gian làm bài kiểm tra.



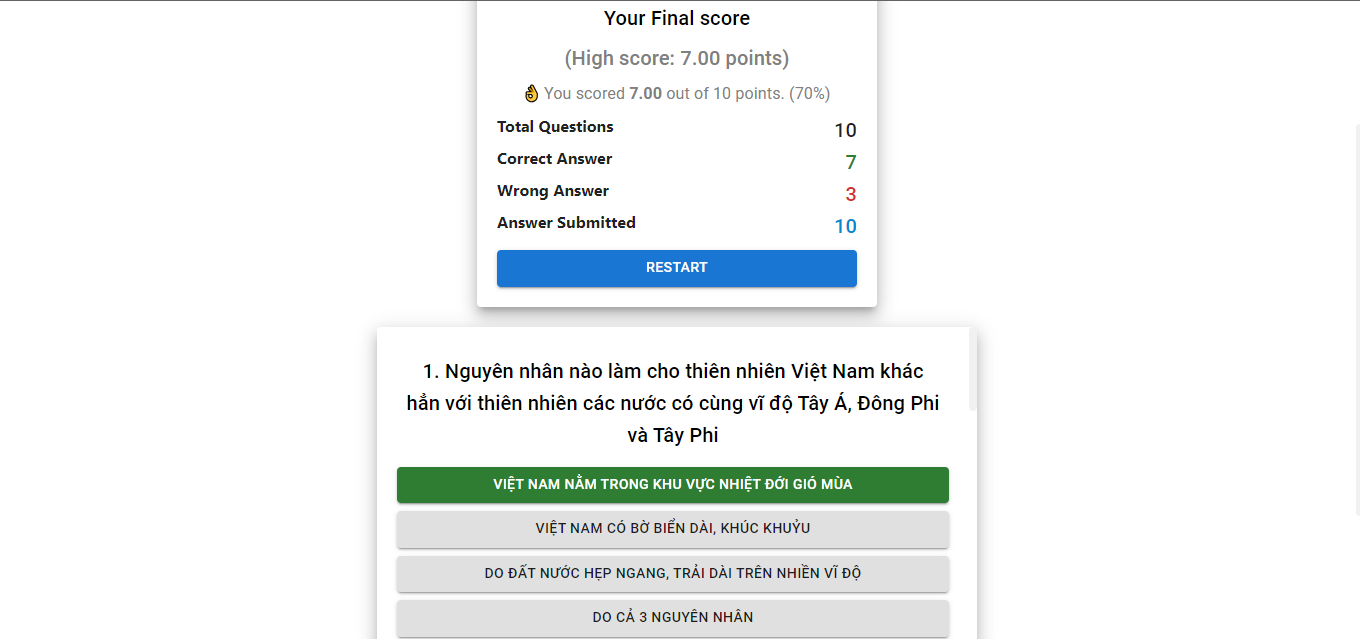
Hình 3. 7 Giao diện trang thông tin bài làm

Mỗi câu hỏi được hiển thị rõ ràng cùng với các lựa chọn đáp án. Người dùng có thể dễ dàng chọn đáp án bằng cách nhấp vào lựa chọn mong muốn. Giao diện bao gồm một bộ đếm thời gian giúp người dùng theo dõi thời gian làm bài còn lại. Điều này giúp người dùng quản lý thời gian hiệu quả hơn, đặc biệt trong các bài thi có giới hạn thời gian. Người dùng có thể dễ dàng điều hướng giữa các câu hỏi thông qua các nút "NEXT" và "PREV". Ngoài ra, giao diện có thể cung cấp một thanh điều hướng hoặc danh sách câu hỏi để người dùng có thể chuyển nhanh đến bất kỳ câu hỏi nào trong bài.



Hình 3. 8 Giao diện hiển thị câu hỏi và đáp án

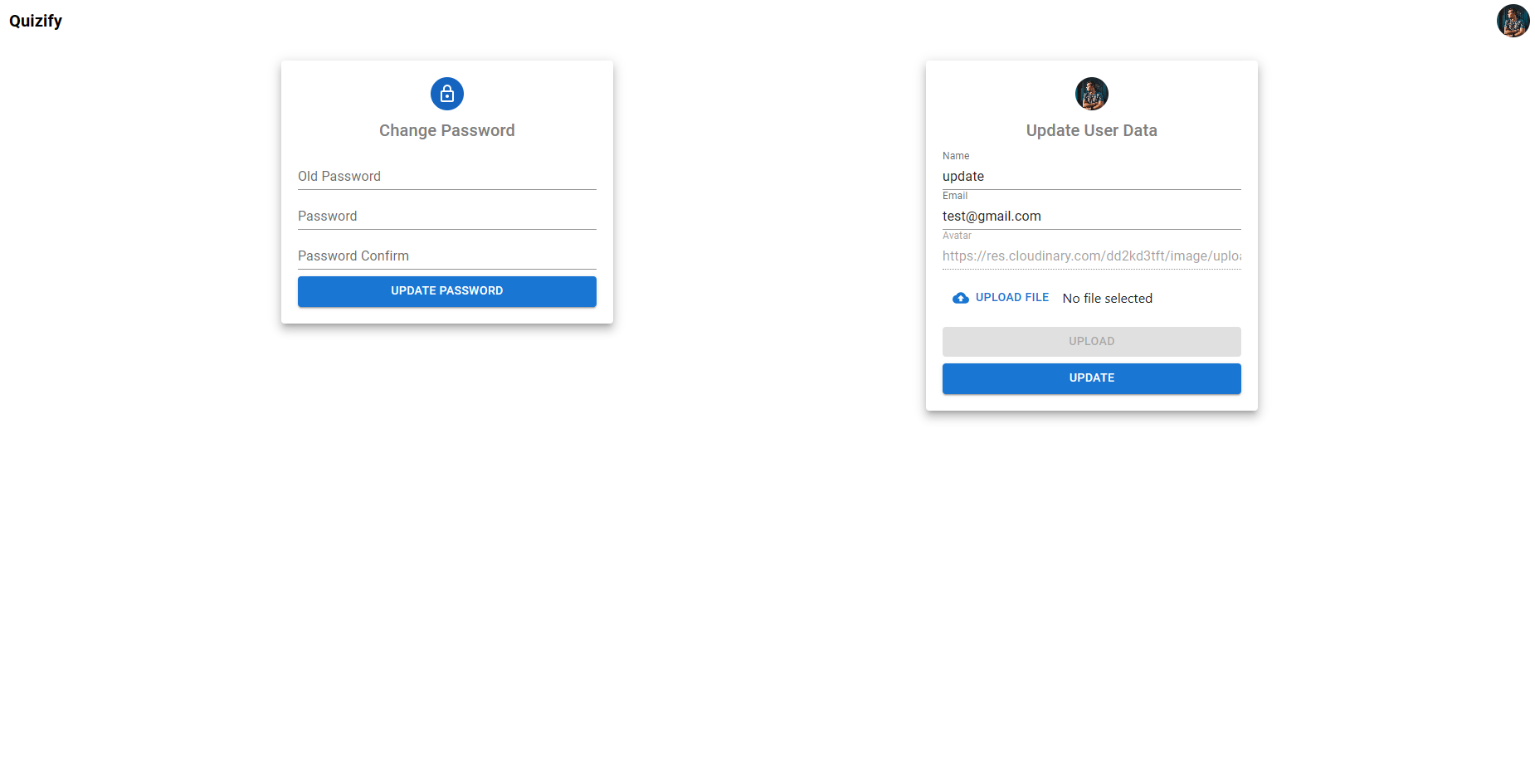
Sau khi hoàn thành bài thi, người dùng có thể nhấn nút " Submit". Hệ thống sẽ xử lý kết quả và hiển thị điểm số cùng với các câu trả lời đúng/sai. Giao diện cung cấp phản hồi chi tiết về kết quả làm bài, giúp người dùng hiểu rõ hơn về những lỗi sai và học hỏi từ đó.



Hình 3. 9 Giao diện xem kết quả

### 3.2.4. Giao diện trang thông tin người dùng

Giao diện trang thông tin người dùng là nơi người dùng có thể xem, chỉnh sửa và quản lý thông tin cá nhân của mình trong hệ thống. Tại đây, người dùng có thể cập nhật thông tin cá nhân như tên, email, đảm bảo rằng dữ liệu của họ luôn được cập nhật và chính xác. Ngoài ra, giao diện cũng cung cấp chức năng đổi mật khẩu, giúp người dùng tăng cường bảo mật cho tài khoản của mình. Người dùng có thể nhập mật khẩu hiện tại và mật khẩu mới, xác nhận mật khẩu để thay đổi, đảm bảo tài khoản luôn được bảo vệ tốt nhất.



Hình 3. 10 Giao diện trang thông tin người dùng

## 3.3. Kiểm thử

### 3.1.1. Kiểm thử giao diện

#### Kiểm thử chức năng đăng nhập

Bảng 3. 1 Kiểm thử chức năng đăng nhập

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **STT** | **Test case** | **Dữ liệu đầu vào** | **Kết quả mong đợi** | **Đầu ra thực tế** | **Kết quả** |
| 1 | Email: Để trống | 1. Dữ liệu đã có trong database  2. Để trống email  3. Click [Đăng nhập] | Hiển thị “This field email is required” | Hiển thị “This field email is required” | Đúng |
| 2 | Email: Chưa đúng định dạng | 1. Dữ liệu đã có trong database  2. Không đúng định dạng email  3. Click [Đăng nhập] | Hiển thị “Please provide a valid email” | Hiển thị “Please provide a valid email” | Đúng |
| 3 | Mật khẩu: Để trống | 1. Dữ liệu đã có trong database  2. Để trống mật khẩu  3. Click [Đăng nhập] | Hiển thị “This field password is required” | Hiển thị “This field password is required” | Đúng |
| 4 | Mật khẩu: Độ dài mật khẩu | 1. Dữ liệu đã có trong database  2. Nhập mật khẩu ngắn  3. Click [Đăng nhập] | Hiển thị “Must be at least 8 characters long” | Hiển thị “Must be at least 8 characters long” | Đúng |
| 5 | Email: Nhập sai email  Mật khẩu: Nhập sai | 1. Dữ liệu đã có trong database  2. Nhập sai email  3. Nhập sai mật khẩu  4. Click [Đăng nhập] | Hiển thị “Incorrect email or password” | Hiển thị “Incorrect email or password” | Đúng |
| 6 | Email: Nhập đúng email  Mật khẩu: Nhập đúng mật khẩu | 1. Dữ liệu đã có trong database  2. Nhập đúng email  3. Nhập đúng mật khẩu  4. Click [Đăng nhập] | Hiển thị “Login success!” | Hiển thị “Login success!” | Đúng |

#### Kiểm thử chức năng đăng ký

Bảng 3. 2 Kiểm thử chức năng đăng ký

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **STT** | **Test case** | **Dữ liệu đầu vào** | **Kết quả mong đợi** | **Đầu ra thực tế** | **Kết quả** |
| 1 | Email: Để trống | 1. Để trống email  2. Click [Đăng ký] | Hiển thị “This field email is required” | Hiển thị “This field email is required” | Đúng |
| 2 | Email: Chưa đúng định dạng | 1. Không đúng định dạng email  2. Click [Đăng ký] | Hiển thị “Please provide a valid email” | Hiển thị “Please provide a valid email” | Đúng |
| 3 | Email: Email đã tồn tại | 1. Dữ liệu đã có trong database  2. Nhập tài khoản email đã tồn tại  3. Click [Đăng ký] | Hiển thị “Email already exists” | Hiển thị “Email already exists” | Đúng |
| 4 | Tên: Để trống | 1. Dữ liệu đã có trong database  2. Để trống tên  3. Click [Đăng ký] | Hiển thị “This field name is required” | Hiển thị “This field name is required” | Đúng |
| 5 | Mật khẩu: Để trống | 1. Dữ liệu đã có trong database  2. Để trống mật khẩu  3. Click [Đăng ký] | Hiển thị “This field password is required” | Hiển thị “This field password is required” | Đúng |
| 6 | Mật khẩu: Độ dài mật khẩu | 1. Dữ liệu đã có trong database  2. Nhập mật khẩu ngắn  3. Click [Đăng ký] | Hiển thị “Must be at least 8 characters long” | Hiển thị “Must be at least 8 characters long” | Đúng |
| 7 | Xác nhận mật khẩu: Để trống | 1. Dữ liệu đã có trong database  2. Để trống mật khẩu  3. Click [Đăng ký] | Hiển thị “This field password is required” | Hiển thị “This field password is required” | Đúng |
| 8 | Xác nhận mật khẩu: Độ dài xác nhận mật khẩu | 1. Dữ liệu đã có trong database  2. Độ dài mật khẩu ngắn  3. Click [Đăng ký] | Hiển thị “Must be at least 8 characters long” | Hiển thị “Must be at least 8 characters long” | Đúng |
| 9 | Xác nhận mật khẩu: Xác nhận mật khẩu không giống | 1. Dữ liệu đã có trong database  2. Xác nhận mật khẩu không giống mật khẩu  3. Click [Đăng ký] | Hiển thị “Password Confirm: Passwords are not the same!" | Hiển thị “Password Confirm: Passwords are not the same!" | Đúng |
| 10 | Email: Nhập đúng email  Tên: Nhập đúng  Mật khẩu: Nhập đúng mật khẩu  Xác nhận mật khẩu nhập đúng | 1. Dữ liệu đã có trong database  2. Nhập đúng email  3. Nhập đúng tên  4. Nhập đúng mật khẩu  5. Nhập đúng xác nhận mật khẩu  6. Click [Đăng ký] | Hiển thị “Register success! | Hiển thị “Register success! | Đúng |

#### Kiểm thử chức năng đổi mật khẩu

Bảng 3. 3 Kiểm thử chức năng đổi mật khẩu

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **STT** | **Test case** | **Dữ liệu đầu vào** | **Kết quả mong đợi** | **Đầu ra thực tế** | **Kết quả** |
| 1 | Mật khẩu cũ: Để trống | 1. Dữ liệu đã có trong database  2. Để trống mật khẩu  3. Click [Đổi mật khẩu] | Hiển thị “This field password Current is required” | Hiển thị “This field password Current is required” | Đúng |
| 2 | Mật khẩu cũ: Độ dài mật khẩu | 1. Dữ liệu đã có trong database  2. Nhập mật khẩu ngắn  3. Click [Đăng ký] | Hiển thị “Must be at least 8 characters long” | Hiển thị “Must be at least 8 characters long” | Đúng |
| 3 | Mật khẩu: Để trống | 1. Dữ liệu đã có trong database  2. Để trống mật khẩu  3. Click [Đổi mật khẩu] | Hiển thị “This field password  Current is required” | Hiển thị “This field password  Current is required” | Đúng |
| 4 | Mật khẩu: Độ dài mật khẩu | 1. Dữ liệu đã có trong database  2. Nhập mật khẩu ngắn  3. Click [Đổi mật khẩu] | Hiển thị “Must be at least 8 characters long” | Hiển thị “Must be at least 8 characters long” | Đúng |
| 5 | Xác nhận mật khẩu: Để trống | 1. Dữ liệu đã có trong database  2. Để trống mật khẩu  3. Click [Đổi mật khẩu] | Hiển thị “This field password is required” | Hiển thị “This field password is required” | Đúng |
| 6 | Xác nhận mật khẩu: Độ dài xác nhận mật khẩu | 1. Dữ liệu đã có trong database  2. Nhập mật khẩu ngắn  3. Click [Đổi mật khẩu] | Hiển thị “Must be at least 8 characters long” | Hiển thị “Must be at least 8 characters long” | Đúng |
| 7 | Xác nhận mật khẩu: Xác nhận mật khẩu không giống | 1. Dữ liệu đã có trong database  2. Xác nhận mật khẩu không giống mật khẩu  3. Click [Đổi mật khẩu] | Hiển thị “Password Confirm: Passwords are not the same!" | Hiển thị “Password Confirm: Passwords are not the same!" | Đúng |
| 8 | Mật khẩu cũ: Nhập đúng  Mật khẩu: Nhập đúng  Xác nhận mật khẩu: Nhập đúng | 1. Dữ liệu đã có trong database  2. Nhập đúng mật khẩu cũ  3. Nhập đúng mật khẩu  3. Nhập đúng xác nhận mật khẩu  4. Click [Đổi mật khẩu] | Hiển thị “Change pasword success!” | Hiển thị “Change pasword success!” | Đúng |

#### Kiểm thử chức năng thêm người dùng

Bảng 3. 4 Kiểm thử chức năng thêm người dùng

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **STT** | **Test case** | **Dữ liệu đầu vào** | **Kết quả mong đợi** | **Đầu ra thực tế** | **Kết quả** |
| 1 | Email: Để trống | 1. Để trống email  2. Click [Đăng ký] | Hiển thị “This field email is required” | Hiển thị “This field email is required” | Đúng |
| 2 | Email: Chưa đúng định dạng | 1. Không đúng định dạng email  2. Click [Đăng ký] | Hiển thị “Please provide a valid email” | Hiển thị “Please provide a valid email” | Đúng |
| 3 | Email: Email đã tồn tại | 1. Dữ liệu đã có trong database  2. Nhập tài khoản email đã tồn tại  3. Click [Đăng ký] | Hiển thị “Email already exists” | Hiển thị “Email already exists” | Đúng |
| 4 | Tên: Để trống | 1. Dữ liệu đã có trong database  2. Để trống tên  3. Click [Đăng ký] | Hiển thị “This field name is required” | Hiển thị “This field name is required” | Đúng |
| 5 | Mật khẩu: Để trống | 1. Dữ liệu đã có trong database  2. Để trống mật khẩu  3. Click [Đăng ký] | Hiển thị “This field password is required” | Hiển thị “This field password is required” | Đúng |
| 6 | Mật khẩu: Độ dài mật khẩu | 1. Dữ liệu đã có trong database  2. Nhập mật khẩu ngắn  3. Click [Đăng ký] | Hiển thị “Must be at least 8 characters long” | Hiển thị “Must be at least 8 characters long” | Đúng |
| 7 | Xác nhận mật khẩu: Để trống | 1. Dữ liệu đã có trong database  2. Để trống mật khẩu  3. Click [Đăng ký] | Hiển thị “This field password is required” | Hiển thị “This field password is required” | Đúng |
| 8 | Xác nhận mật khẩu: Độ dài xác nhận mật khẩu | 1. Dữ liệu đã có trong database  2. Độ dài mật khẩu ngắn  3. Click [Đăng ký] | Hiển thị “Must be at least 8 characters long” | Hiển thị “Must be at least 8 characters long” | Đúng |
| 9 | Xác nhận mật khẩu: Xác nhận mật khẩu không giống | 1. Dữ liệu đã có trong database  2. Xác nhận mật khẩu không giống mật khẩu  3. Click [Đăng ký] | Hiển thị “Password Confirm: Passwords are not the same!" | Hiển thị “Password Confirm: Passwords are not the same!" | Đúng |
| 10 | Quyền: Để trống | 1. Dữ liệu đã có trong database  2. Để trống quyền người dùng  3. Click [Đăng ký] | Hiển thị “Please select at least one option” | Hiển thị “Please select at least one option” | Đúng |
| 11 | Email: Nhập đúng email  Tên: Nhập đúng  Mật khẩu: Nhập đúng mật khẩu  Xác nhận mật khẩu nhập đúng | 1. Dữ liệu đã có trong database  2. Nhập đúng email  3. Nhập đúng tên  4. Nhập đúng mật khẩu  5. Nhập đúng xác nhận mật khẩu  6. Click [Đăng ký] | Hiển thị “Register success! | Hiển thị “Register success! | Đúng |

#### Kiểm thử chức năng thêm câu hỏi

Bảng 3. 5 Kiểm thử chức năng thêm câu hỏi

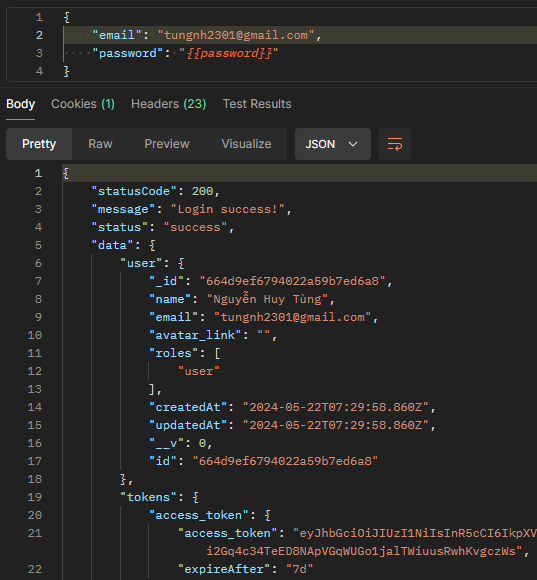
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **STT** | **Test case** | **Dữ liệu đầu vào** | **Kết quả mong đợi** | **Đầu ra thực tế** | **Kết quả** |
| 1 | Câu hỏi: Để trống câu hỏi | 1. Để trống câu hỏi  2. Click [Tạo mới] | Hiển thị “This field title is required” | Hiển thị “This field title is required” | Đúng |
| 2 | Câu hỏi: Độ dài câu hỏi | 1. Nhập câu hỏi ngắn  2. Click [Tạo mới] | Hiển thị “Must be at least 10 characters long” | Hiển thị “Must be at least 10 characters long” | Đúng |
| 3 | Câu hỏi: Câu hỏi đã tồn tại | 1. Dữ liệu đã có trong database  2. Nhập câu hỏi đã tồn tại  3. Click [Tạo mới] | Hiển thị “Question already exists” | Hiển thị “Question already exists” | Đúng |
| 4 | Câu hỏi: Nhập đúng câu hỏi | 1. Nhập câu hỏi đúng  2. Click [Tạo mới] | Hiển thị “Create success!” | Hiển thị “Create success!” | Đúng |

### 3.1.2. Kiểm thử API

#### Kiểm thử API đăng nhập

Bảng 3. 6 Kiểm thử API đăng nhập

|  |  |  |
| --- | --- | --- |
| **Test case** | **Dữ liệu đầu vào** | **Hệ thống trả về** |
| - Email nhập đúng  - Mật khẩu nhập đúng | Nhập đúng email và mật khẩu | - statusCode, message, status  - name, email, avatar, roles, createdAt, updateAt, tokens |

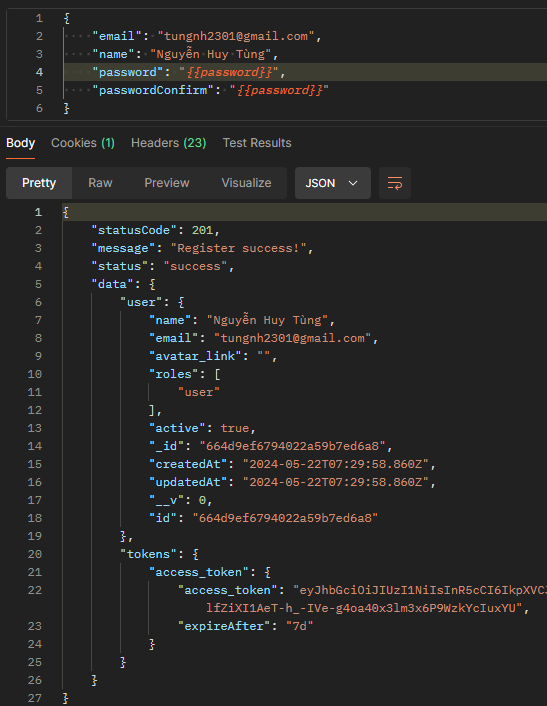


Hình 3. 11 Kiểm thử API đăng nhập

#### Kiểm thử API đăng ký

Bảng 3. 7 Kiểm thử API đăng ký

|  |  |  |
| --- | --- | --- |
| **Test case** | **Dữ liệu đầu vào** | **Hệ thống trả về** |
| - Email nhập đúng  - Mật khẩu nhập đúng  - Xác nhận mật khẩu đúng | Nhập đúng email mật khẩu và xác nhận mật khẩu | - statusCode, message, status  - name, email, avatar, roles, createdAt, updateAt |

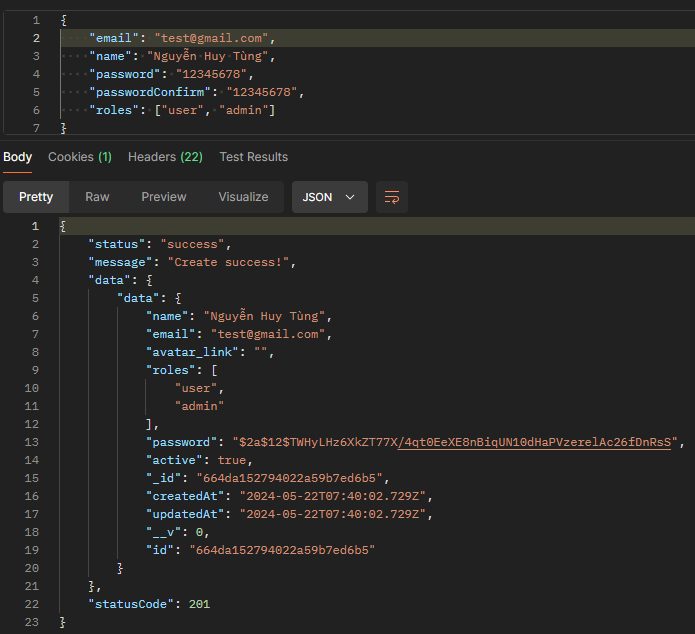


Hình 3. 12 Kiểm thử API đăng ký

#### Kiểm thử API thêm người dùng

Bảng 3. 8 Kiểm thử API thêm người dùng

|  |  |  |
| --- | --- | --- |
| **Test case** | **Dữ liệu đầu vào** | **Hệ thống trả về** |
| - Email nhập đúng  - Mật khẩu nhập đúng  - Xác nhận mật khẩu đúng | Nhập đúng email mật khẩu và xác nhận mật khẩu | - statusCode, message, status,  - name, email, avatar, roles, createdAt, updateAt |

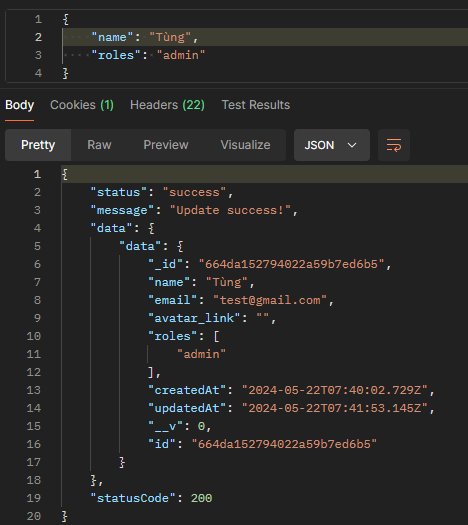


Hình 3. 13 Kiểm thử API thêm người dùng

#### Kiểm thử API cập nhật thông tin người dùng

Bảng 3. 9 Kiểm thử API cập nhật thông tin người dùng

|  |  |  |
| --- | --- | --- |
| **Test case** | **Dữ liệu đầu vào** | **Hệ thống trả về** |
| - Mã người dùng nhập đúng  - Email nhập đúng  - Tên nhập đúng  - Quyền nhập đúng | Nhập đúng mã người dùng Nhập đúng email tên và quyền người dùng | - statusCode, message, status  - name, email, avatar, roles, createdAt, updateAt |

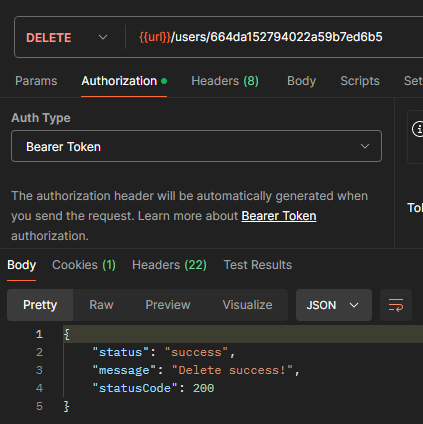


Hình 3. 14 Kiểm thử API cập nhật thông tin người dùng

#### Kiểm thử API xóa người dùng

Bảng 3. 10 Kiểm thử API xóa người dùng

|  |  |  |
| --- | --- | --- |
| **Test case** | **Dữ liệu đầu vào** | **Hệ thống trả về** |
| - Mã người dùng nhập đúng | Nhập đúng mã người dùng | - statusCode  - message  - status |



Hình 3. 15 Kiểm thử API xóa người dùng

# KẾT LUẬN VÀ KIẾN NGHỊ

**Kết quả đạt được**

Sau khi thực hiện khóa luận tốt nghiệp, sinh viên đã đạt được kết quả như sau: Xây dựng thành công một ứng dụng trắc nghiệm trực tuyến linh hoạt và hiệu quả. Đầu tiên, giao diện người dùng thân thiện và dễ sử dụng đã được triển khai, giúp người dùng tương tác một cách tự nhiên và trực quan. Tiếp theo, em đã tích hợp các tính năng quản lý tài khoản và tạo quản lý câu hỏi một cách hoàn chỉnh. Người dùng có thể dễ dàng đăng nhập, đăng ký và quản lý thông tin cá nhân, cũng như tạo và quản lý các câu hỏi tra theo nhu cầu của mình. Ứng dụng cũng đã triển khai thành công chức năng làm bài trắc nghiệm và xem kết quả ngay sau khi hoàn thành. Em đã tối ưu hóa hiệu suất ứng dụng để đảm bảo hoạt động mượt mà trên nhiều thiết bị và mạng internet khác nhau.

Nắm được quy trình cơ bản của hoạt động làm bài thi; Tìm hiểu về xây dựng giao diện trang web bằng ReactJS; Nắm được cách xây dựng RESTful API bằng NodeJS; Hiểu thêm về MongoDB, thực hành trên ứng dụng MongoDB Compass.

**Hạn chế và hướng phát triển**

Do thời gian trong khuôn khổ khóa luận tốt nghiệp có hạn, nên phần phân tích thiết kế còn chưa được đầy đủ, phần giao diện còn đơn giản. Một số chức năng của dự án chưa được như kỳ vọng: chưa có tính năng chuyển đổi ngôn ngữ, chưa có tính năng đổi giao diện sáng tối.

Hướng phát triển tiếp theo là tích hợp tính năng xếp hạng và phản hồi: Thêm tính năng cho phép người dùng xem xếp hạng của họ sau khi hoàn thành bài thi và nhận phản hồi về kết quả của mình; Tối ưu hóa hiệu suất: Tiếp tục tối ưu hóa hiệu suất của ứng dụng bằng cách sử dụng kỹ thuật tải trước dữ liệu và giảm thiểu việc render không cần thiết; Mở rộng tính năng: Bổ sung các tính năng mới như tạo bài thi tùy chỉnh, chia sẻ kết quả với bạn bè, và thống kê kết quả bài thi.

# TÀI LIỆU THAM KHẢO

**A-TÀI LIỆU TIẾNG VIỆT**

[1] Thạc Bình Cường (2002), *Phân tích và thiết kế hệ thống thông tin*, NXB Khoa học và Kỹ thuật.

[2] Trần Văn Định (2014), *Phân tích và thiết kế hệ thống thông tin*, NXB Đại học Quốc gia TP.HCM.

[3] Nguyễn Văn Đức (2004), *Phân tích và thiết kế hệ thống thông tin*, NXB Đại học Quốc gia TP.HCM.

[4] Nguyễn Huy Khánh (2004), *Giáo trình lập trình web*, NXB Đại học Khoa học tự nhiên HCM.

[5] Trịnh Thanh Tùng (2004), *Thiết kế và lập trình web*, NXB Đại học Bách khoa Hà Nội.

[6] Trần Quang Vinh (2007), *UML và các phương pháp phát triển phần mềm*, NXB Thống kê.

**B-TÀI LIỆU TIẾNG ANH**

[7] Alex Sidorrenko (2021), *A Visual Guide to React Rendering*, alexsidorenko

[8] Chris Minnick (2022), Beginning ReactJS Foundations Building User Interfaces with ReactJS: An Approachable Guide, Wiley

[9] Mark Erikson (2020), *A (Mostly) Complete Guide to React Rendering Behavior*, isquaredsoftware.

[10] Robin Wieruch (2018), *The Road to React: Your journey to master plain yet pragmatic React.js*