ĐẠI HỌC HUẾ
TRƯỜNG ĐẠI HỌC KHOA HỌC
KHOA CÔNG NGHỆ THÔNG TIN

KHOÁ LUẬN
TỐT NGHIỆP ĐẠI HỌC

Đề tài:

WEBSITE HỖ TRỢ QUẢN LÝ KỲ THI

Sinh viên thực hiện: **CHÂU ANH KIỆT**
Khóa: **K44 - HỆ CHÍNH QUY**

Huế, tháng 5 - năm 2024

ĐẠI HỌC HUẾ
TRƯỜNG ĐẠI HỌC KHOA HỌC
KHOA CÔNG NGHỆ THÔNG TIN

KHOÁ LUẬN
TỐT NGHIỆP ĐẠI HỌC
NGÀNH CÔNG NGHỆ THÔNG TIN

Đề tài:

WEBSITE HỖ TRỢ QUẢN LÝ KỲ THI

Sinh viên thực hiện: **CHÂU ANH KIỆT**
Khóa: **K44 - HỆ CHÍNH QUY**
Giáo viên hướng dẫn: **THẠC SĨ TRẦN NGUYÊN PHONG**

Huế, tháng 5 - năm 2024

LỜI CẢM ƠN

Trong thời gian hoàn thành khóa luận này, tôi muốn bày tỏ lòng biết ơn sâu sắc đến tất cả những người đã đóng góp và hỗ trợ cho dự án của tôi. Đặc biệt, tôi muốn gửi lời cảm ơn chân thành đến Th.S Trần Nguyên Phong, giảng viên hướng dẫn của em trong đồ án tốt nghiệp lần này. Thầy đã dành thời gian và năng lượng để hỗ trợ tôi trong quá trình nghiên cứu và làm khóa luận này. Sự chỉ dẫn và góp ý của thầy không chỉ giúp tôi hoàn thiện dự án một cách chuyên nghiệp, mà còn giúp tôi phát triển và học hỏi nhiều hơn trong quá trình này. Sau đó, tôi muốn gửi lời cảm ơn tới các anh trong Công ty 3S: anh Nguyễn Đức Nhật Tùng, anh Phan Văn Hoài Nhân. Các anh đã chỉ bảo em nhiều kiến thức liên quan tới việc xây dựng và triển khai các website, hệ thống quản lý . Em cũng xin được cảm ơn các thầy, cô giảng viên đã dạy em nhiều kiến thức trong những ngày ngồi trên ghế đại học. Em xin chân thành cảm ơn.

BẢNG CHỮ CÁI VIẾT TẮT

ABBREVIATIONS	MEANING
API	Application Programming Interface
MVC	Model-View-Controller
SQL	Structured Query Language
ORM	Object-Relational Mapping
UC	Use Case
HTML	HyperText Markup Language
DI	Dependency Injection
DOM	Document Object Model
AJAX	Asynchronous JavaScript and XML
ERD	Entity Relationship Diagram

DANH MỤC CÁC BẢNG BIỂU

DANH MỤC CÁC HÌNH ẢNH

MỤC LỤC

LỜI MỞ ĐẦU

1. Mô tả vấn đề

Hiện tại việc tạo và nộp bài thi hay bài kiểm tra của trường Đại học Khoa học đều được thực hiện thông qua Google Classroom nhưng vẫn còn những bất cập như khó kiểm soát việc gian lận của sinh viên, khó trong việc phát hiện sinh viên gian lận. Việc tạo kỳ thi, bài kiểm tra vẫn còn rườm rà, chưa thực sự phù hợp với yêu cầu của nhà trường.

2. Mục tiêu

Dự án này nhằm mục đích thiết kế và phát triển một trang web về quản lý kỳ thi và nộp bài thi bằng cách sử dụng ASP.NET Core MVC. Các trọng tâm chính là:

- Tạo ra một hệ thống quản lý kỳ thi hiệu quả và linh hoạt, giúp giảng viên dễ dàng quản lý các kỳ thi và bài kiểm tra và sinh viên dễ dàng tham gia nộp bài thi.

- Tối ưu hóa quy trình tạo, chỉnh sửa và quản lý các kỳ thi và bài kiểm tra, giảm thiểu thủ tục phức tạp và tiết kiệm thời gian.

- Nâng cao khả năng kiểm soát gian lận trong quá trình làm bài thi và bài kiểm tra của sinh viên.

- Quản lý quyền truy cập các tệp.

- Có các chức năng giúp tải tệp một cách dễ dàng.

- Tích hợp với hệ thống đào tạo của nhà trường để đồng bộ hóa thông tin và tạo sự liền mạch trong quy trình học tập và giảng dạy

3. Bố cục của khóa luận

Sau phần Lời mở đầu, luận văn được cấu trúc trong ba chương:

Chương 1: Cơ sở lý thuyết - Trong chương này, luận văn trình bày cơ sở lý thuyết.

Chương 2: Phân tích và Thiết kế Hệ thống - Chương này bao gồm mô tả yêu cầu, các biểu đồ như biểu đồ use case, biểu đồ hoạt động và thiết kế cơ sở

dữ liệu.

Chương 3: Kết quả - Chương này trình bày các giao diện của trang web.

Chương 1. CƠ SỞ LÝ THUYẾT

1.1. ASP.NET Core MVC

1.1.1. Lịch sử và sự phát triển

- ASP.NET MVC được ra mắt vào năm 2009 như một phần mở rộng của ASP.NET, cung cấp một mô hình phát triển web dựa trên mẫu Model-View-Controller (MVC). Nó được thiết kế để cung cấp sự tách biệt rõ ràng giữa các phần khác nhau của ứng dụng (business logic, UI logic và input logic).

- ASP.NET Core: Ra mắt vào năm 2016 như một sự cải tiến lớn từ ASP.NET. ASP.NET kết hợp ASP.NET MVC và Web API thành một framework thống nhất. Đây là một nền tảng mã nguồn mở, đa nền tảng, chạy trên cả Windows, macOS và Linux.

1.1.2. Các tính năng nổi bật

- Đa nền tảng: ASP.NET Core MVC hỗ trợ chạy trên nhiều hệ điều hành như Windows, macOS và Linux, giúp phát triển ứng dụng linh hoạt trên mọi nền tảng.

- Routing: Cung cấp hệ thống routing mạnh mẽ, cho phép ánh xạ các yêu cầu HTTP tới các action cụ thể trong controller dựa trên URL và các mẫu route. Routing cho phép bạn cấu hình các mẫu URL. Có 2 phương pháp routing:

  - Conventional Routing: Định tuyến theo các mẫu URL truyền thống.
  - Attribute Routing: Sử dụng các thuộc tính (attributes) trực tiếp trên các controller và action để xác định routing.

- Model Binding: Tự động ánh xạ dữ liệu từ các yêu cầu HTTP vào các tham số của action trong controller, giúp xử lý dữ liệu dễ dàng và hiệu quả.

- Model Validation: Tích hợp các công cụ validation cho các mô hình, giúp kiểm tra và xác nhận dữ liệu một cách dễ dàng và đáng tin cậy.

Dependency Injection (DI): ASP.NET Core MVC tích hợp sẵn Dependency Injection, cho phép quản lý và tiêm các phụ thuộc một cách dễ dàng, tăng tính linh hoạt và khả năng mở rộng của ứng dụng. Các dịch vụ có thể được cấu hình với các vòng đời khác nhau như transient (tạo mới mỗi khi nó được yêu cầu), scoped (tạo ra một lần cho mỗi phạm vi (scope) của yêu cầu), và singleton (tạo ra chỉ một lần và duy trì trong suốt vòng đời của ứng dụng).

- Filters: Filters là các thành phần cho phép thực hiện các logic trước hoặc sau khi các action được gọi. Các loại filters bao gồm:

- Authorization Filters: Kiểm tra quyền truy cập của người dùng.
- Resource Filters: Thực thi mã trước và sau khi routing diễn ra.
- Action Filters: Thực thi mã trước và sau khi action method thực thi.
- Exception Filters: Xử lý các ngoại lệ xảy ra trong quá trình thực thi action.
- Result Filters: Thực thi mã trước và sau khi kết quả action được xử lý và trả về client.

- Areas: Areas là một cách để tổ chức và phân chia ứng dụng thành các phần chức năng riêng biệt, giúp quản lý mã nguồn một cách dễ dàng và hiệu quả, đặc biệt là cho các ứng dụng lớn và phức tạp.

- Razor view engine: Các view trong ASP.NET Core MVC sử dụng Razor view engine để render các view. Razor là một ngôn ngữ đánh dấu mẫu (template markup language) gọn nhẹ và linh hoạt để định nghĩa các view bằng cách nhúng mã C#. Razor được sử dụng để tạo ra nội dung web động trên máy chủ. Bạn có thể trộn lẫn mã phía server và mã phía client một cách rõ ràng.

- Security: Cung cấp các cơ chế bảo mật mạnh mẽ để bảo vệ ứng dụng khỏi các lỗ hổng bảo mật.

- Authentication: Xác thực người dùng thông qua các phương thức như cookie, bearer tokens, OAuth, OpenID Connect, v.v.
- Authorization: Phân quyền truy cập dựa trên vai trò và chính sách.

- Data Protection: Bảo vệ dữ liệu nhạy cảm thông qua mã hóa và giải mã.
- Antiforgery: Bảo vệ ứng dụng khỏi các cuộc tấn công Cross-Site Request Forgery (CSRF).
- CORS: Cross-Origin Resource Sharing, cho phép hoặc từ chối các yêu cầu từ các domain khác.

1.1.3. Ưu điểm của ASP.NET Core MVC

- Mã nguồn mở: Được hỗ trợ và phát triển bởi cộng đồng cùng với Microsoft.

- Hiệu suất cao: Hiệu năng vượt trội so với các framework khác.

- Bảo mật tốt: Hỗ trợ nhiều tính năng bảo mật mạnh mẽ.

- Tích hợp tốt với các công nghệ hiện đại: Chẳng hạn như Docker, Kubernetes, và các dịch vụ đám mây như Azure.

- Cộng đồng hỗ trợ rộng rãi: Tài liệu phong phú và cộng đồng phát triển rộng lớn.

1.1.4. Kiến trúc của ASP.NET Core MVC

1.1.4.1. Mô hình MVC

Mô hình kiến trúc MVC chia một ứng dụng thành ba nhóm thành phần chính: Model, View và Controller. Mô hình này giúp đạt được sự tách biệt giữa các mối quan tâm khác nhau trong ứng dụng.

Cách hoạt động của mô hình MVC:

1. User Request (Yêu cầu của người dùng): Khi người dùng gửi một yêu cầu đến ứng dụng, yêu cầu đó sẽ được chuyển đến Controller.

2. Controller: Controller chịu trách nhiệm xử lý yêu cầu này, tương tác với Model để thực hiện các hành động của người dùng hoặc truy xuất kết quả của các truy vấn.

3. Model: Model đại diện cho trạng thái của ứng dụng và chứa logic nghiệp vụ hoặc các thao tác cần thực hiện. Controller sẽ sử dụng Model để lấy dữ liệu cần thiết.

4. View: Sau khi có dữ liệu từ Model, Controller sẽ chọn View để hiển thị dữ liệu đó cho người dùng. View sẽ nhận dữ liệu từ Controller và render nội dung giao diện người dùng.

Delineation of Responsibilities (Phân định trách nhiệm):

Việc phân định rõ ràng các trách nhiệm này giúp dễ dàng mở rộng ứng dụng phức tạp vì dễ dàng hơn trong việc lập trình, gỡ lỗi và kiểm thử những thứ có một công việc duy nhất (Model, View, hoặc Controller). Ngược lại, nếu mã triển khai và logic nghiệp vụ được kết hợp trong một đối tượng duy nhất, thì mỗi khi giao diện người dùng thay đổi, đối tượng chứa logic nghiệp vụ cũng phải sửa đổi. Điều này thường dẫn đến lỗi và yêu cầu kiểm thử lại logic nghiệp vụ sau mỗi thay đổi nhỏ của giao diện người dùng.

Trách nhiệm của từng thành phần:

1. Model Responsibilities:

- Model đại diện cho trạng thái của ứng dụng và chứa logic nghiệp vụ hoặc các thao tác cần thực hiện
- Logic nghiệp vụ nên được đóng gói trong Model, cùng với bất kỳ logic triển khai nào cho việc duy trì trạng thái của ứng dụng.
- Các View thường sử dụng các loại ViewModel được thiết kế để chứa dữ liệu cần hiển thị trên View. Controller tạo và điền các đối tượng ViewModel từ Model.

2. View Responsibilities:

- Views chịu trách nhiệm trình bày nội dung thông qua giao diện người dùng.
- Razor view engine: Views sử dụng Razor view engine để nhúng mã .NET vào mã HTML.
- Logic tối thiểu: View chỉ nên chứa logic tối thiểu liên quan đến việc trình bày nội dung. Nếu cần thực hiện nhiều logic trong các tệp View để hiển thị dữ liệu từ một Model phức tạp, hãy xem xét sử dụng View Component, ViewModel để đơn giản hóa View.

3. Controller Responsibilities:

- Controllers là các thành phần xử lý tương tác của người dùng, làm việc với Model và cuối cùng chọn View để render.
- Trong ứng dụng MVC, View chỉ hiển thị thông tin; Controller xử lý và phản hồi các đầu vào và tương tác của người dùng.
- Controller là điểm nhập ban đầu và chịu trách nhiệm chọn loại Model nào để làm việc và View nào để render.

Mô hình MVC giúp tách biệt rõ ràng các phần khác nhau của ứng dụng, làm cho việc phát triển, bảo trì và mở rộng ứng dụng trở nên dễ dàng và hiệu quả hơn.

1.1.4.2. Middleware

Middleware là các thành phần trung gian được cấu hình trong request pipeline để xử lý và đáp ứng các HTTP Request. Mỗi HTTP Request sẽ đi qua các middleware trong pipeline trước khi đến đích, và mỗi middleware có thể thực hiện các chức năng nhất định trên yêu cầu đó trước khi chuyển tiếp nó cho middleware tiếp theo hoặc trả về phản hồi cho client.

Các middleware phổ biến trong ASP.NET Core bao gồm:

- Authentication Middleware: Xác thực người dùng và xác định xem ai đang gửi yêu cầu. Điều này thường bao gồm việc kiểm tra thông tin đăng nhập hoặc mã thông báo xác thực.

- Logging Middleware: Ghi lại các sự kiện quan trọng trong quá trình xử lý yêu cầu, như thông tin về yêu cầu đến và phản hồi đi, các lỗi xảy ra, hoặc thông tin gỡ lỗi.

- Error Handling Middleware: Xử lý các ngoại lệ và lỗi trong quá trình xử lý yêu cầu, đảm bảo rằng phản hồi được trả về cho client là thích hợp và có ý nghĩa.

1.2. SQL Server

SQL Server là một công cụ quan trọng khác mà bạn có thể sử dụng để lưu trữ và quản lý dữ liệu trong các dự án C# và ASP.NET Core MVC của mình. SQL Server là một hệ quản trị cơ sở dữ liệu quan hệ (RDBMS) được phát triển bởi Microsoft, cung cấp một giải pháp mạnh mẽ và có khả năng mở rộng cho việc lưu trữ và truy xuất dữ liệu.

Hình 1.1 - SQL Server

SQL Server cung cấp nhiều lợi ích cho việc lưu trữ dữ liệu trong các ứng dụng ASP.NET Core MVC của bạn:

- Tổ chức dữ liệu: SQL Server tổ chức dữ liệu thành các bảng với các mối quan hệ được xác định, cho phép lưu trữ và truy xuất dữ liệu hiệu quả. Nó hỗ trợ SQL (Structured Query Language) để truy vấn và thao tác dữ liệu.

- Tính toàn vẹn và nhất quán của dữ liệu: SQL Server đảm bảo tính toàn vẹn của dữ liệu bằng cách thực thi các ràng buộc, chẳng hạn như khóa chính, khóa duy nhất và khóa ngoại. Nó cũng hỗ trợ các Transaction, cho phép bạn duy trì tính nhất quán và tính nguyên tử của dữ liệu trong quá trình thực hiện nhiều thao tác cơ sở dữ liệu.

- Khả năng mở rộng và hiệu suất: SQL Server được thiết kế để xử lý các tập dữ liệu lớn và phức tạp. Nó cung cấp các tính năng như lập chỉ mục (indexing), tối ưu hóa truy vấn và cơ chế bộ nhớ đệm để cải thiện hiệu suất. Ngoài ra, SQL Server hỗ trợ các tùy chọn khả dụng cao và mở rộng, chẳng hạn như cụm (clustering) và sao chép (replication).

- Bảo mật và kiểm soát truy cập: SQL Server cung cấp các tính năng bảo mật mạnh mẽ để bảo vệ dữ liệu của bạn. Nó hỗ trợ các cơ chế xác thực và ủy quyền, cũng như mã hóa dữ liệu nhạy cảm. Bạn có thể định nghĩa kiểm soát truy cập chi tiết để hạn chế quyền của người dùng và đảm bảo tính bảo mật của dữ liệu.

- Tích hợp với .NET Framework: SQL Server tích hợp liền mạch với .NET framework, cho phép bạn tận dụng thư viện ADO.NET để truy cập dữ liệu. ADO.NET cung cấp một tập hợp các lớp và API để kết nối với SQL Server, thực thi các truy vấn SQL và truy xuất dữ liệu vào mã C# của bạn.

- Quản lý và quản trị: SQL Server đi kèm với các công cụ quản lý như SQL Server Management Studio (SSMS) và Azure Data Studio, cung cấp giao diện đồ họa để quản lý cơ sở dữ liệu, tạo bảng, viết truy vấn và giám sát hiệu suất. Bằng cách tích hợp SQL Server vào các dự án ASP.NET Core MVC của bạn, bạn có thể lưu trữ, truy xuất và quản lý dữ liệu một cách hiệu quả. Nó cung cấp một giải pháp đáng tin cậy và có khả năng mở rộng để xử lý các yêu cầu dữ liệu của các ứng dụng của bạn.

1.3. Một số thư viện hỗ trợ

1.3.1. jQuery:

jQuery là một thư viện JavaScript phổ biến được sử dụng để tạo ra các hiệu ứng động, thao tác DOM dễ dàng, và tương tác AJAX trong các ứng dụng web.

1.3.2. Bootstrap:

Bootstrap là một framework front-end phổ biến, cung cấp các thành phần giao diện người dùng sẵn có, responsive design, và các utilities CSS và JavaScript hữu ích để tạo ra giao diện web đẹp mắt và dễ sử dụng.

1.3.3. MediatR:

MediatR là một thư viện hỗ trợ cho mô hình truyền tải thông điệp (Mediator pattern) trong các ứng dụng .NET. Nó giúp giảm sự phụ thuộc giữa các thành phần trong ứng dụng và tạo ra các yêu cầu và phản hồi một cách tự nhiên và linh hoạt. Nó giúp triển khai mô hình CQRS bằng cách đóng gói các logic xử lý thành các đối tượng Command/Query riêng biệt. Nó cung cấp hai interface chính: IRequest và INotification, được sử dụng để định nghĩa các Command/Query và Events tương ứng.

1.3.4. Dapper:

Dapper là một micro ORM (Object-Relational Mapping) cho .NET, giúp thao tác với cơ sở dữ liệu một cách đơn giản và hiệu quả hơn bằng cách cung cấp một cách để ánh xạ dữ liệu giữa đối tượng và cơ sở dữ liệu một cách tối ưu.

1.3.5. AutoMapper:

AutoMapper là một thư viện giúp thực hiện ánh xạ dữ liệu tự động giữa các đối tượng của các lớp khác nhau trong ứng dụng. Nó giúp giảm thiểu việc viết mã lặp lại và giúp tạo ra mã nguồn dễ đọc và dễ bảo trì hơn.

Chương 2. PHÂN TÍCH VÀ THIẾT KẾ HỆ THỐNG

2.1. Mô tả yêu cầu

2.1.1. Mô tả tổng quan

Hệ thống website quản lý kỳ thi và nộp bài thi được xây dựng nhằm đáp ứng nhu cầu chống gian lận, kiểm soát gian lận trong việc nộp bài thi của sinh viên, đồng thời quy trình tạo kỳ thi hay tạo bài kiểm tra phù hợp với yêu cầu của giảng viên.

Các mục tiêu chính của dự án bao gồm:

- Quản lý kỳ thi hiệu quả: Trang web sẽ cung cấp các chức năng hỗ trợ việc quản lý kỳ thi đúng yêu cầu của người dùng như tạo, chỉnh sửa, xóa kỳ thi. Ngoài ra, các chức năng như tìm kiếm, xem thông tin kỳ thi cũng được thiết kế sao cho thân thiện với người dùng. Chức năng tạo hay chỉnh sửa kỳ thi phải có bước chọn sinh viên tham gia được liên kết với hệ thống đào tạo giúp thuận tiện trong việc chọn sinh viên tham gia.

- Quản lý bài nộp của sinh viên: Trang web sẽ có các chức năng giúp giảng viên quản lý bài nộp của sinh viên một cách dễ dàng và nhanh chóng như tìm kiếm, xem thông tin bài nộp, xem và tải xuống các tệp đính kèm.

- Nộp bài thi hoặc hủy nộp bài: Trang web sẽ cung cấp chức năng cho phép sinh viên nộp bài thi hay hủy nộp bài nếu cần để thay đổi tệp đính kèm.

- Quyền truy cập các tệp: Hệ thống sẽ có cơ chế kiểm soát quyền truy cập các tệp đề thi hay tệp bài nộp. Giảng viên không thể xem hay tải tệp đề thi hoặc tệp bài nộp của kỳ thi không phải mình tổ chức. Sinh viên không thể xem hay tải tệp đề thi mà sinh viên không được tham gia. Hoặc xem hay tải tệp bài nộp của sinh viên khác.

- Kiểm soát gian lận: Hệ thống sẽ có chức năng ngăn việc nộp bài thi từ các IP không thuộc địa chỉ IP của trường. Phát hiện những trường hợp hai sinh viên trở lên nộp bài thi cùng một địa chỉ IP.

2.1.2. Yêu cầu chức năng

2.1.2.1. Đăng nhập:

Use case này dùng để đăng nhập vào hệ thống, là tiền đề để sử dụng các chức năng trong hệ thống. Đăng nhập thông qua hệ thống đào tạo. Vì vậy người dùng phải sử dụng tài khoản và mật khẩu mà hệ thống đào tạo cung cấp để truy cập vào hệ thống quản lý kỳ thi.

2.1.2.2. Các chức năng của giảng viên:

a. Tạo kỳ thi:

Giảng viên nhập các thông tin của kỳ thi bao gồm: tiêu đề (bắt buộc), hướng dẫn, thời gian bắt đầu, thời gian kết thúc. Ngoài ra, kỳ thi cần có những thuộc tính bổ sung để giảng viên lựa chọn mà thuộc tính đó mô tả yêu cầu nghiệp vụ như:

- Chỉ cho phép những máy thuộc IP của trường mới được phép nộp bài.

- Lưu địa chỉ IP nếu có 2 bài trở lên trùng IP nộp bài thì sẽ đưa vào trạng thái đang xử lý.

- Ngăn nộp bài nếu quá thời gian làm bài.

Giảng viên có thể tải lên hoặc xóa các tệp đề thi nếu cần. Sau bước nhập thông tin là bước chọn sinh viên tham gia. Giảng viên chỉ có thể chọn sinh viên tham gia thông qua danh sách sinh viên được dự thi từ hệ thống đào tạo.

b. Tạo bài kiểm tra:

Giảng viên nhập các thông tin của bài kiểm tra bao gồm: tiêu đề, hướng dẫn (nếu có), thời gian bắt đầu, thời gian kết thúc. Ngoài ra, bài kiểm tra cần có những thuộc tính bổ sung để giảng viên lựa chọn mà thuộc tính đó mô tả yêu cầu nghiệp vụ như:

- Chỉ cho phép những máy thuộc IP của trường mới được phép nộp bài.

- Lưu địa chỉ IP nếu có 2 bài trở lên trùng IP nộp bài thì sẽ đưa vào trạng thái đang xử lý.

- Ngăn nộp bài nếu quá thời gian làm bài.

Giảng viên có thể tải lên hoặc xóa các tệp đề thi nếu cần. Sau bước nhập thông tin là bước chọn sinh viên tham gia. Giảng viên chọn sinh viên từ các lớp học phần mà giảng viên đang giảng dạy. Ngoài ra, giảng viên còn có thể tìm kiếm sinh viên nằm ngoài các lớp học phần mà giảng viên đang giảng dạy để bổ sung vào danh sách sinh viên tham gia.

c. Tìm kiếm kỳ thi/ bài kiểm tra đã tạo:

Giảng viên có thể xem và tìm kiếm các kỳ thi/ bài kiểm tra đã tạo bằng tiêu đề, trạng thái hoặc khoảng thời gian.

d. Chỉnh sửa thông tin kỳ thi/ bài kiểm tra đã tạo

Giảng viên có thể chỉnh sửa thông tin các kỳ thi/ bài kiểm tra đã tạo đồng thời thêm hoặc xóa sinh viên tham gia.

e. Xóa kỳ thi/ bài kiểm tra đã tạo

Giảng viên có thể xóa kỳ thi/ bài kiểm tra

f. Xem bài nộp của các sinh viên tham gia thi

Giảng viên có thể xem và tải xuống các tệp bài nộp của sinh viên. Giảng viên cũng có thể biết được sinh viên có gian lận hay nộp muộn không. Giảng viên cũng có thể thêm nhận xét.

Thông tin bài nộp gồm: Địa chỉ IP, thời gian nộp, trạng thái, danh sách tệp bài nộp.

g. Thêm nhận xét cho bài nộp

Giảng viên có thể thêm nhận xét cho bài nộp của sinh viên.

Thông tin nhận xét gồm: Nội dung, thời gian nhận xét, tên giảng viên

2.1.2.3. Các chức năng của sinh viên:

a. Tìm kiếm kỳ thi/bài kiểm tra mà sinh viên có tham gia:

Sinh viên có thể tìm kiếm kỳ thi/bài kiểm tra thông qua tiêu đề, trạng thái hoặc khoảng thời gian.

b. Xem thông tin kỳ thi/ bài kiểm tra:

Sinh viên có thể xem các thông tin của kỳ thi. Đề thi chỉ hiển thị cho sinh viên khi đến thời gian bắt đầu. Sinh viên có thể tải tệp đề thi. Sinh viên có thể xem nhận xét của giảng viên ở đây.

c. Nộp bài thi:

Sinh viên có thể tải lên hoặc xóa tệp bài nộp cho mỗi kỳ thi/ bài kiểm tra. Trước khi đến giờ làm bài hoặc sau khi nộp bài thi, sinh viên không thể tải lên hay xóa tệp bài nộp.

d. Hủy nộp bài:

Sinh viên có thể hủy nộp bài để tải lên hoặc xóa tệp bài nộp nếu cần. Chỉ được phép hủy nếu đang trong thời gian thi.

2.1.3. Yêu cầu phi chức năng

- Bảo mật và An toàn:

- Hệ thống phải có cơ chế xác thực mạnh mẽ để đảm bảo chỉ người dùng được phép mới có thể truy cập và sử dụng các chức năng quản lý kỳ thi và nộp bài thi.
- Địa chỉ IP của máy tính được lưu trữ một cách an toàn và không được tiết lộ cho bất kỳ ai ngoài người quản trị hệ thống.

- Hiệu suất và Tương tác:

- Giao diện người dùng phải được thiết kế một cách trực quan và dễ sử dụng để tối ưu trải nghiệm người dùng.
- Hệ thống phải xử lý các yêu cầu từ người dùng một cách nhanh chóng và hiệu quả, đảm bảo rằng thời gian đáp ứng là tối thiểu.
- Hệ thống phải có khả năng xử lý tối thiểu 1000 người dùng đồng thời mà không làm giảm hiệu suất.

- Tính Mở rộng và Linh hoạt:

- Hệ thống phải có khả năng mở rộng để có thể chứa đựng một lượng lớn các kỳ thi và bài kiểm tra cũng như số lượng người dùng đồng thời.

- Phải có khả năng mở rộng chức năng và tính năng của hệ thống để có thể đáp ứng được các yêu cầu và nhu cầu mới của người dùng trong tương lai.

- **Tính Di động**:

- Giao diện người dùng phải được tối ưu hóa để có thể sử dụng trên nhiều thiết bị khác nhau, bao gồm cả điện thoại di động và máy tính bảng.

- **Khả năng duy trì:**

- Hệ thống phải được thiết kế dễ dàng bảo trì, bao gồm khả năng cập nhật và nâng cấp mà không làm gián đoạn dịch vụ.
- Mã nguồn và tài liệu liên quan phải được tổ chức rõ ràng và dễ hiểu để hỗ trợ quá trình bảo trì và phát triển trong tương lai.

- **Hỗ trợ và Bảo trì**:

- Hệ thống phải được hỗ trợ và bảo trì thường xuyên để đảm bảo tính ổn định và an toàn của nó.

- Hệ thống phải được thiết kế dễ dàng bảo trì, bao gồm khả năng cập nhật và nâng cấp mà không làm gián đoạn dịch vụ.

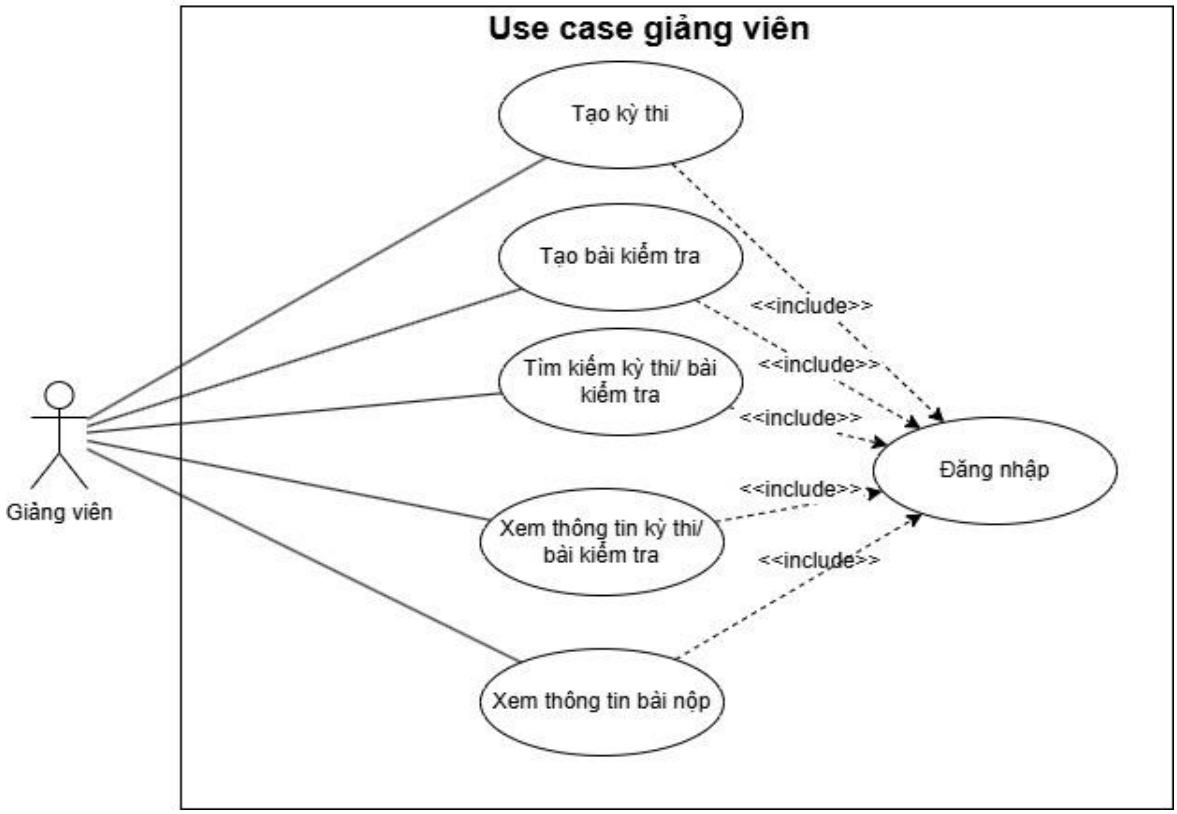
- Mã nguồn và tài liệu liên quan phải được tổ chức rõ ràng và dễ hiểu để hỗ trợ quá trình bảo trì và phát triển trong tương lai.

- Phải có các biện pháp sao lưu dữ liệu thường xuyên và quy trình phục hồi dữ liệu để đảm bảo rằng dữ liệu không bị mất mát trong trường hợp xảy ra sự cố.

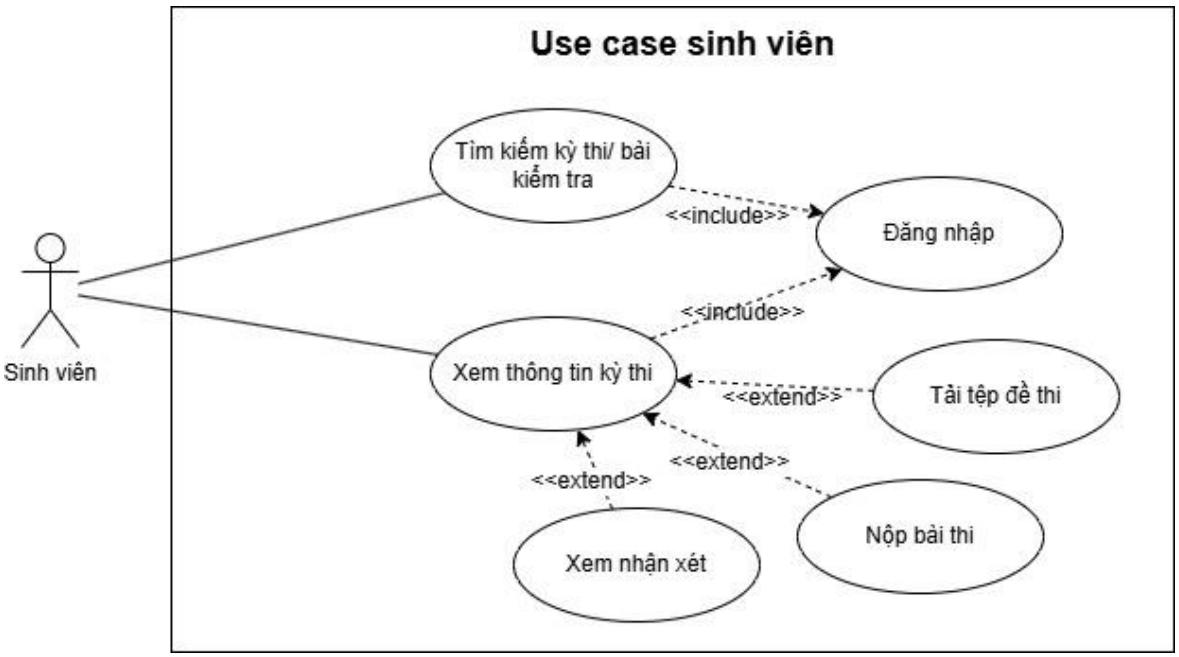
2.2. Phân tích thiết kế hệ thống:

2.2.1. Biểu đồ Use Case:

2.2.1.1. Use case tổng quát:

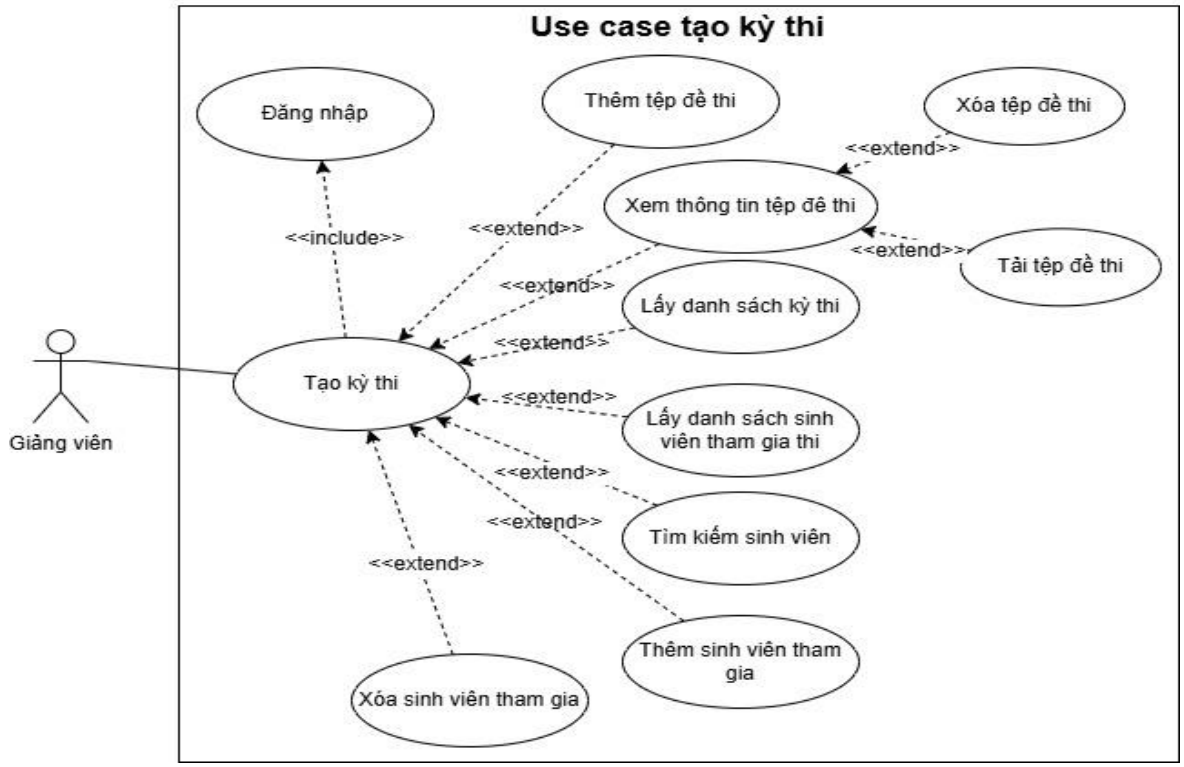


Hình 2.1 - Use case tổng quát (giảng viên)



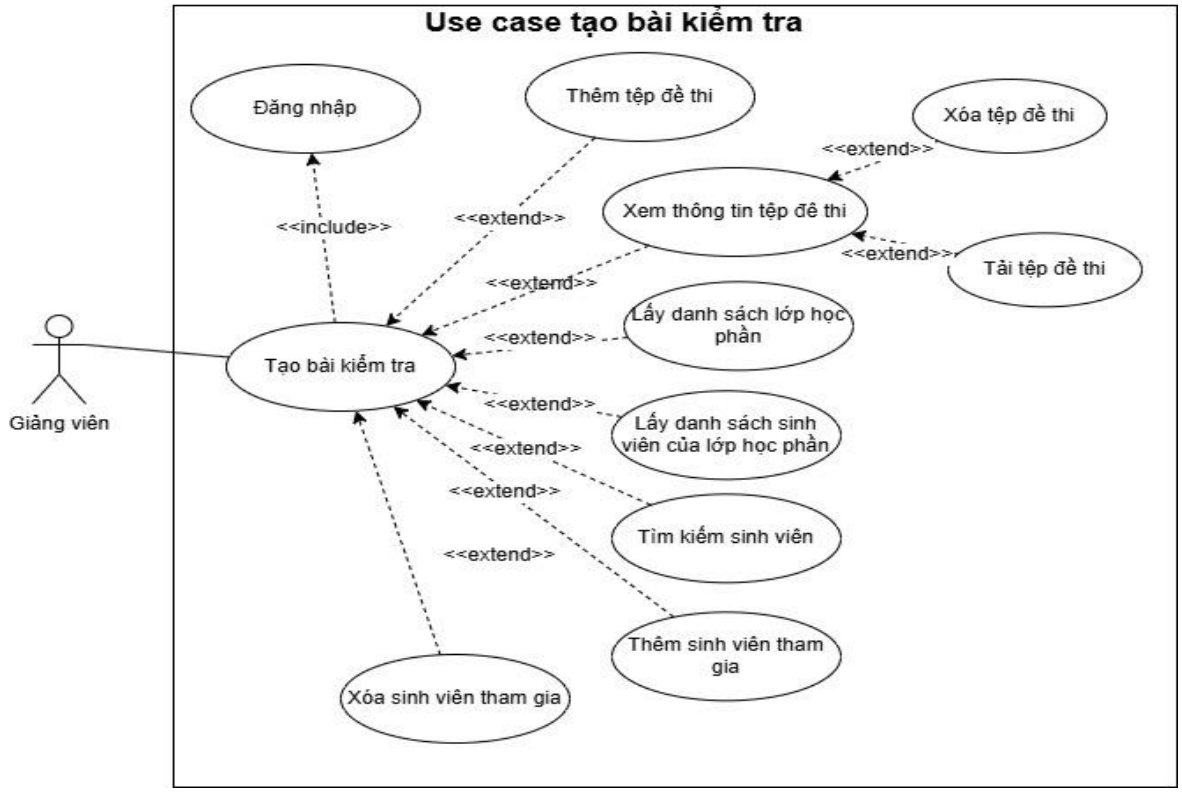
Hình 2.2 - Use case tổng quát (sinh viên)

2.2.1.2. Tạo kỳ thi



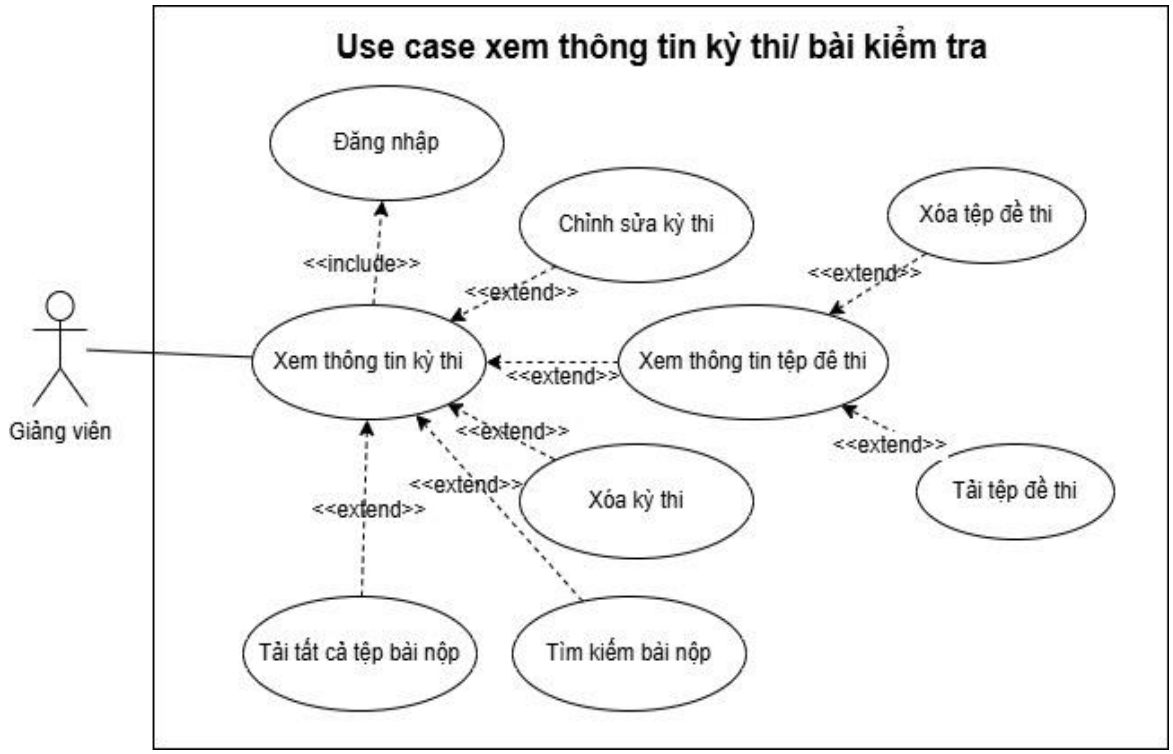
Hình 2.3 - Use case tạo kỳ thi

2.2.1.3. Tạo bài kiểm tra



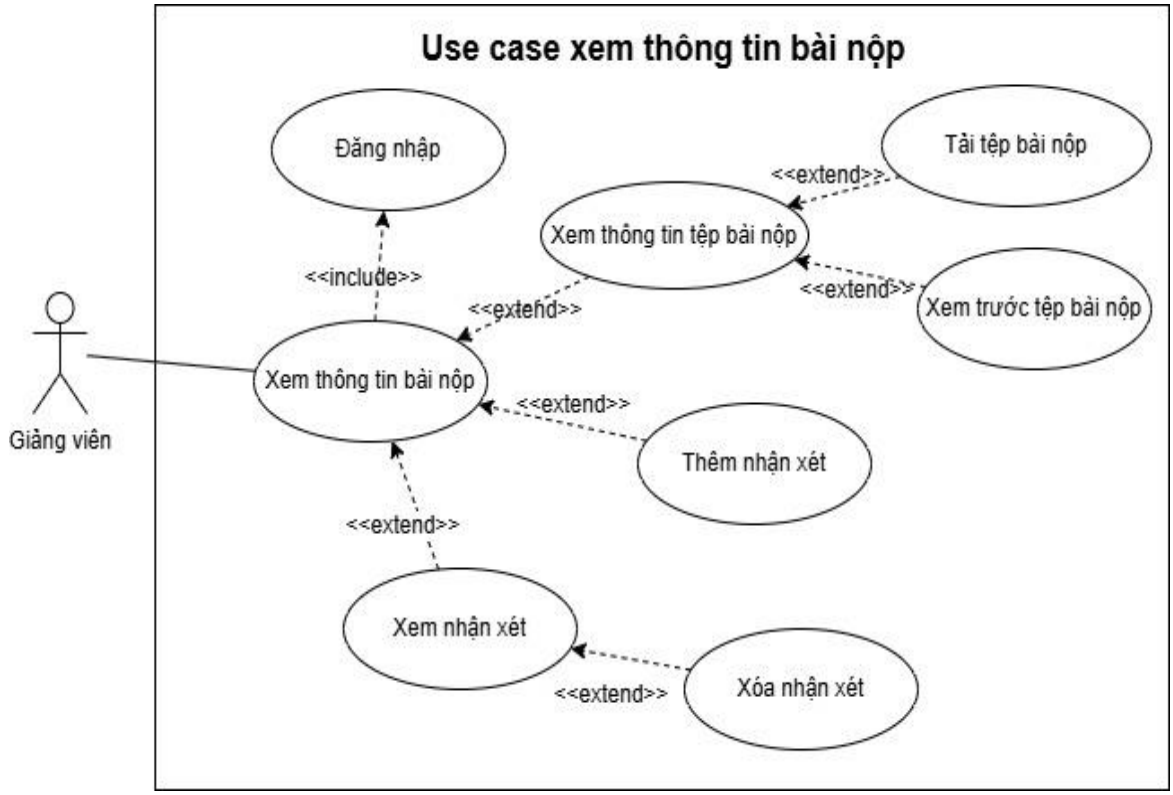
Hình 2.4 - Use case tạo bài kiểm tra

2.2.1.4. Xem thông tin kỳ thi/ bài kiểm tra



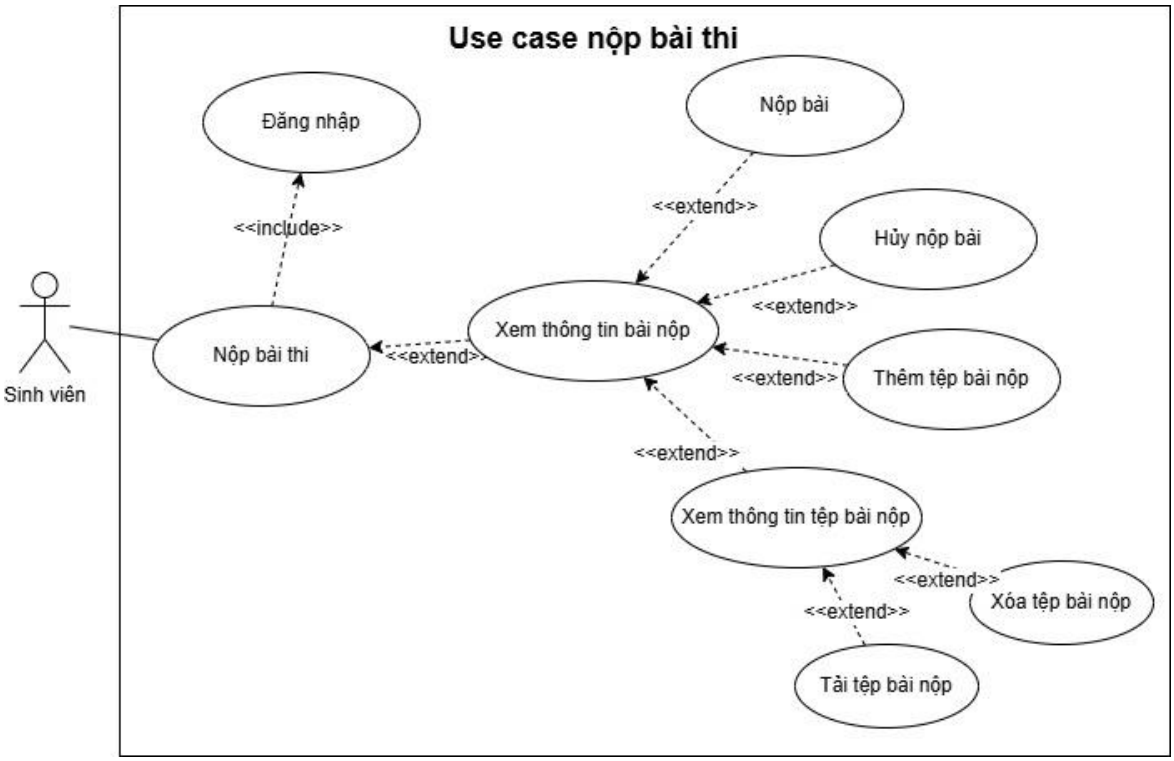
Hình 2.5 - Use case xem thông tin kỳ thi/ bài kiểm tra

2.2.1.5. Xem thông tin bài nộp



Hình 2.6 - Use case xem thông tin bài nộp

2.2.1.6. Nộp bài thi



Hình 2.7 - Use case nộp bài thi

2.2.2. Đặc tả use case:

2.2.2.1. Đăng nhập

- Actors: Giảng viên/ Sinh viên

- Describe: Tác nhân sử dụng tài khoản của hệ thống đào tạo tương ứng của họ để đăng nhập vào hệ thống.

- Precondition: Tác nhân đã có tài khoản trong hệ thống.

- The main event flow:

1. Tác nhân đến trang của hệ thông.
2. Hệ thống chuyển hướng đến trang đăng nhập của hệ thống đào tạo.
3. Tác nhân nhập thông tin đăng nhập của họ vào các trường tương ứng:
   - Tài khoản
   - Mật khẩu
4. Tác nhân xem lại thông tin đăng nhập.

5. Tác nhân nhấn vào nút “Đăng nhập”.

6. Hệ thống đào tạo xác thực thông tin đăng nhập:

7. Nếu có bất kỳ lỗi xác thực nào xảy ra hoặc thông tin đăng nhập đã nhập không chính xác, hệ thống sẽ hiển thị các thông báo lỗi thích hợp.

8. Nếu thông tin đăng nhập chính xác thì chuyển hướng đến trang sinh viên hoặc giảng viên của hệ thống quản lý kỳ thi tùy thuộc vào vai trò của họ lựa chọn.

- Main post-event: Tác nhân đăng nhập thành công vào hệ thống bằng thông tin đăng nhập tài khoản tương ứng của họ và có quyền truy cập vào các tính năng và chức năng của hệ thống dựa trên vai trò của họ với tư cách là Giảng viên hoặc Sinh viên.

2.2.2.2. Tạo kỳ thi

- Actors: Giảng viên

- Describe: Giảng viên muốn tạo kỳ thi.

- Precondition: Giảng viên đã đăng nhập thành công vào tài khoản của họ.

- The main event flow:

1. Giảng viên đến trang tạo kỳ thi.

2. Hệ thống hiển thị giao diện biểu mẫu gồm các trường như: tiêu đề, hướng dẫn, thời gian bắt đầu, thời gian kết thúc, các checkbox như: kiểm tra IP, thi trên máy trường, dừng nộp bài khi quá hạn, nút để tải lên tệp đề thi.

3. Giảng viên nhập thông tin kỳ thi.

4. Giảng viên tải lên tệp đề thi.

5. Nếu dung lượng tệp quá lớn, hệ thống thông báo tệp không hợp lệ.

6. Nếu tệp hợp lệ thì lưu tệp và lưu thông tin tệp vào cơ sở dữ liệu.

7. Giảng viên nhấn nút “Chọn sinh viên”.

8. Nếu tiêu đề chưa được nhập hoặc dữ liệu nhập vào không hợp lệ, hệ thống hiển thị thông báo “Dữ liệu không hợp lệ”

9. Nếu dữ liệu nhập vào hợp lệ, hệ thống lưu thông tin kỳ thi vào cơ sở dữ liệu và chuyển đến giao diện chọn sinh viên tham gia.

10. Giảng viên chọn kỳ thi.

11. Hệ thống lấy danh sách sinh viên được phép dự thi từ hệ thống đào tạo.

12. Giảng viên chọn sinh viên tham gia từ danh sách trên.

13. Giảng viên nhấn nút "Lưu" .

14. Nếu danh sách sinh viên tham gia rỗng thì thông báo chưa chọn sinh viên tham gia.

15. Nếu danh sách sinh viên tham gia khác rỗng thì thông báo tạo thành công và lưu danh sách sinh viên tham gia vào cơ sở dữ liệu.

- Main post-event: Từ danh sách sinh viên tham gia, hệ thống tạo các bài nộp với trạng thái là chưa nộp bài.

2.2.2.3. Tạo bài kiểm tra

- Actors: Giảng viên

- Describe: Giảng viên muốn tạo bài kiểm tra.

- Precondition: Giảng viên đã đăng nhập thành công vào tài khoản của họ.

- The main event flow:

1. Giảng viên đến trang tạo bài kiểm tra.

2. Hệ thống hiển thị giao diện biểu mẫu gồm các trường như: tiêu đề, hướng dẫn, thời gian bắt đầu, thời gian kết thúc, các checkbox như: kiểm tra IP, thi trên máy trường, dừng nộp bài khi quá hạn, nút để tải lên tệp đề thi.

3. Giảng viên nhập thông tin bài kiểm tra.

4. Giảng viên tải lên tệp đề thi.

5. Nếu dung lượng tệp quá lớn, hệ thống thông báo tệp không hợp lệ.

6. Nếu tệp hợp lệ thì lưu tệp và lưu thông tin tệp vào cơ sở dữ liệu.

7. Giảng viên nhấn nút “Chọn sinh viên”.

8. Nếu tiêu đề chưa được nhập hoặc dữ liệu nhập vào không hợp lệ, hệ thống hiển thị thông báo “Dữ liệu không hợp lệ”

9. Nếu dữ liệu nhập vào hợp lệ, hệ thống lưu thông tin kỳ thi vào cơ sở dữ liệu và chuyển đến giao diện chọn sinh viên tham gia.

10. Hệ thống lấy danh sách lớp học phần của giảng viên trong học kỳ và năm học đã chọn.

11. Giảng viên chọn lớp học phần.

12. Hệ thống lấy danh sách sinh viên của lớp học phần.

13. Giảng viên chọn sinh viên tham gia từ danh sách trên.

14. Giảng viên có thể chọn sinh viên khác nằm ngoài danh sách của các lớp học phần đang giảng dạy.

15. Giảng viên nhấn nút “Lưu” .

16. Nếu danh sách sinh viên tham gia rỗng thì thông báo chưa chọn sinh viên tham gia.

17. Nếu danh sách sinh viên tham gia khác rỗng thì thông báo tạo thành công và lưu danh sách sinh viên tham gia vào cơ sở dữ liệu.

- Main post-event: Từ danh sách sinh viên tham gia, hệ thống tạo các bài nộp với trạng thái là chưa nộp bài.

2.2.2.4. Xem thông tin bài nộp

- Actors: Giảng viên

- Describe: Giảng viên xem chi tiết thông tin bài nộp, xem và tải các tệp bài nộp hoặc nhận xét bài nộp.

- Precondition: Giảng viên đã đăng nhập thành công vào tài khoản của họ và là người tổ chức kỳ thi.

- The main event flow:

1. Giảng viên chọn bài nộp muốn xem từ danh sách bài nộp của kỳ thi.

2. Nếu bài nộp có trạng thái là chưa nộp bài hoặc thiếu thì không thể xem được bài nộp

3. Nếu bài nộp có trạng thái khác chưa nộp bài và thiếu thì hiển thị thông tin bài nộp và các tệp bài nộp

4. Giảng viên chọn tệp bài nộp muốn xem hoặc tải.

5. Hệ thống hiển thị bản xem trước của tệp.

6. Giảng viên tải đề thi.

7. Giảng viên đăng nhận xét.

- Main post-event:.

2.2.2.5. Giảng viên xem thông tin kỳ thi

- Actors: Giảng viên

- Describe: Giảng viên xem thông tin kỳ thi, xem và tải các tệp đề thi, xem và tìm kiếm bài nộp.

- Precondition: Giảng viên đã đăng nhập thành công vào tài khoản của họ và là người tổ chức kỳ thi.

- The main event flow:

1. Giảng viên chọn kỳ thi muốn xem từ danh sách kỳ thi.

2. Hệ thống hiển thị thông tin kỳ thi, các tệp đề thi, danh sách bài nộp của kỳ thi.

3. Giảng viên tải tệp đề thi

4. Giảng viên tìm kiếm bài nộp.

5. Giảng viên tải tất cả các tệp bài nộp của kỳ thi.

- Main post-event:

2.2.2.6. Xóa kỳ thi

- Actors: Giảng viên

- Describe: Giảng viên xóa kỳ thi.

- Precondition: Giảng viên đã đăng nhập thành công vào tài khoản của họ và là người tổ chức kỳ thi.

- The main event flow:

1. Giảng viên nhấn nút "xóa" ở kỳ thi muốn xóa

2. Nếu kỳ thi có trạng thái là đã kết thúc và chưa có sinh viên nộp bài thì thông báo không thể xóa kỳ thi.

3. Ngược lại, hiển thị thông báo xác nhận xóa.

4. Nếu giảng viên nhấn nút xác nhận, hệ thống thông báo xóa thành công.

5. Ngược lại trở về giao diện ban đầu.

- Main post-event: Hệ thống xóa kỳ thi, tệp đề thi, bài nộp, tệp bài nộp của kỳ thi.

2.2.2.7. Sinh viên xem thông tin kỳ thi

- Actors: Sinh viên

- Describe: Sinh viên xem thông tin kỳ thi, xem và tải các tệp đề thi, xem thông tin bài nộp của họ.

- Precondition: Sinh viên đã đăng nhập thành công vào tải khoản của họ và sinh viên nằm trong danh sách tham gia kỳ thi.

- The main event flow:

1. Sinh viên chọn kỳ thi muốn xem từ danh sách kỳ thi.

2. Nếu thời gian bắt đầu của kỳ thi lớn hơn thời gian hiện tại, hệ thống hiển thị thông báo chưa đến giờ làm bài, ẩn hướng dẫn và đề thi. Sinh viên không được tải lên tệp bài nộp hay nộp bài.

3. Nếu thời gian bắt đầu của kỳ thi bé hơn hoặc bằng thời gian hiện tại, hệ thống các thông tin của kỳ thi.

- Main post-event:.

2.2.2.8. Nộp bài thi

- Actors: Sinh viên

- Describe: Sinh viên tải lên hoặc xóa các tệp bài nộp và nộp bài.

- Precondition: Sinh viên đã đăng nhập thành công vào tải khoản của họ và sinh viên nằm trong danh sách tham gia kỳ thi.

- The main event flow:

1. Sinh viên chọn kỳ thi muốn nộp bài.

2. Hệ thống hiển thị thông tin kỳ thi.

3. Nếu thời gian bắt đầu lớn hơn thời gian hiện tại và địa chỉ IP không hợp lệ, sinh viên không thể tải lên tệp bài nộp và nộp bài.

4. Ngược lại, sinh viên có thể tải lên hoặc xóa các tệp bài nộp.

5. Nếu tệp bài nộp có dung lượng quá lớn hoặc không hợp lệ thì thông báo tệp không hợp lệ.

6. Ngược lại, lưu tệp vào bộ nhớ của hệ thống và lưu thông tin tệp vào cơ sở dữ liệu.

7. Sinh viên nhấn nút "Nộp bài"

8. Nếu danh sách tệp bài nộp rỗng, thông báo chưa tải lên tệp nào.

9. Ngược lại, cập nhật thông tin của bài nộp.

10. Hệ thống cập nhật lại giao diện.

- Main post-event: Hệ thống cập nhật trạng thái, địa chỉ IP, thời gian nộp của bài nộp.

2.2.2.9. Tìm kiếm kỳ thi

- Actors: Giảng viên/ Sinh viên

- Describe: Hệ thống hiển thị danh sách kỳ thi tìm được khi người dùng tìm kiếm kỳ thi theo tiêu đề, trạng thái, khoảng thời gian.

- Precondition: Người dùng đã đăng nhập thành công vào tài khoản của họ.

- The main event flow:

1. Giảng viên đến trang danh sách kỳ thi.

2. Hệ thống hiển thị tất cả kỳ thi mà sinh viên tham gia hoặc kỳ thì của giảng viên tạo tùy theo loại tài khoản.

3. Giảng viên tìm kiếm theo tiêu đề, trạng thái, khoảng thời gian

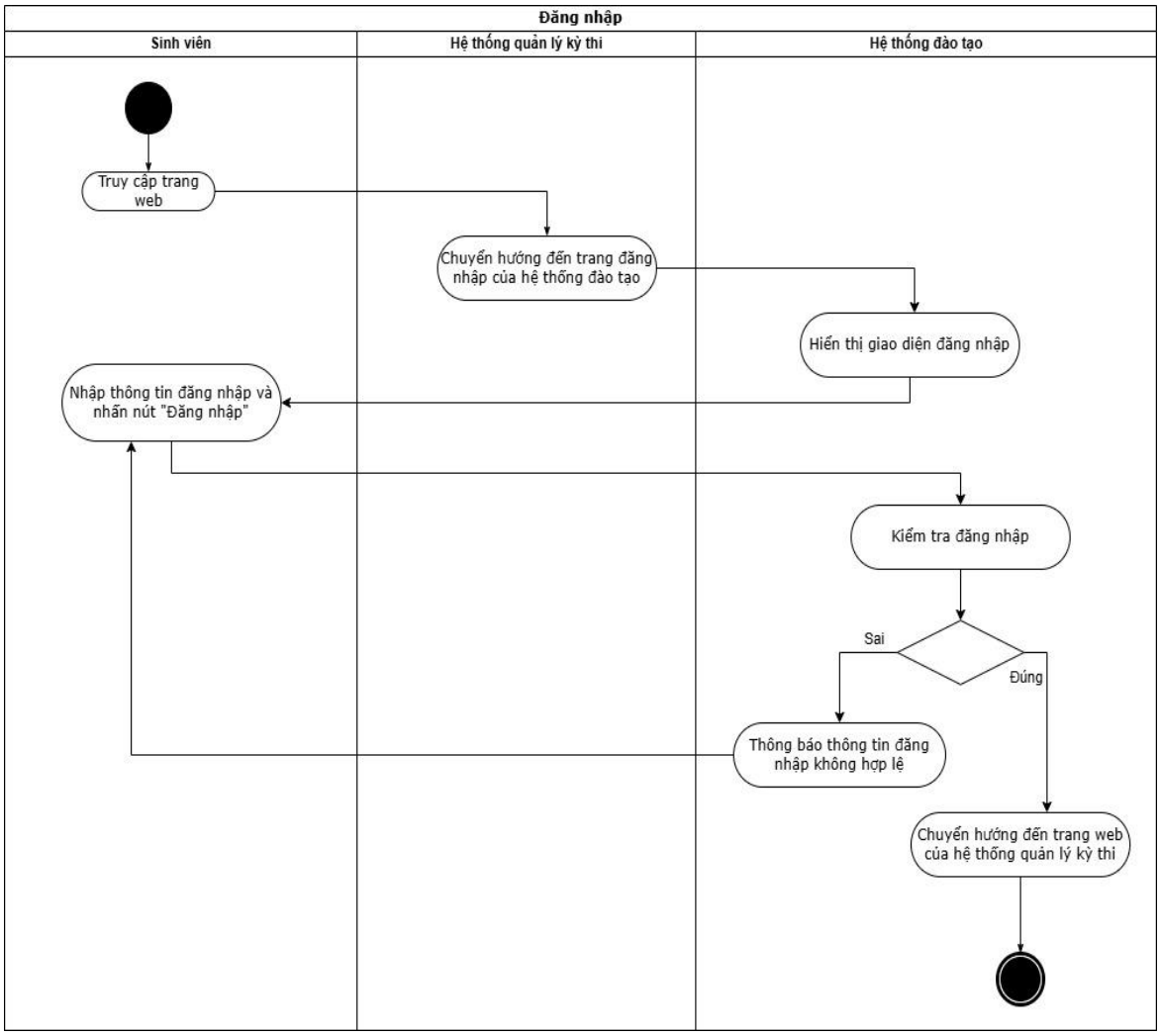
4. Nếu không tìm thấy dữ liệu, hệ thống thông báo không tìm thấy dữ liệu

5. Ngược lại, hệ thống tìm kiếm và phân trang.

- Main post-event:

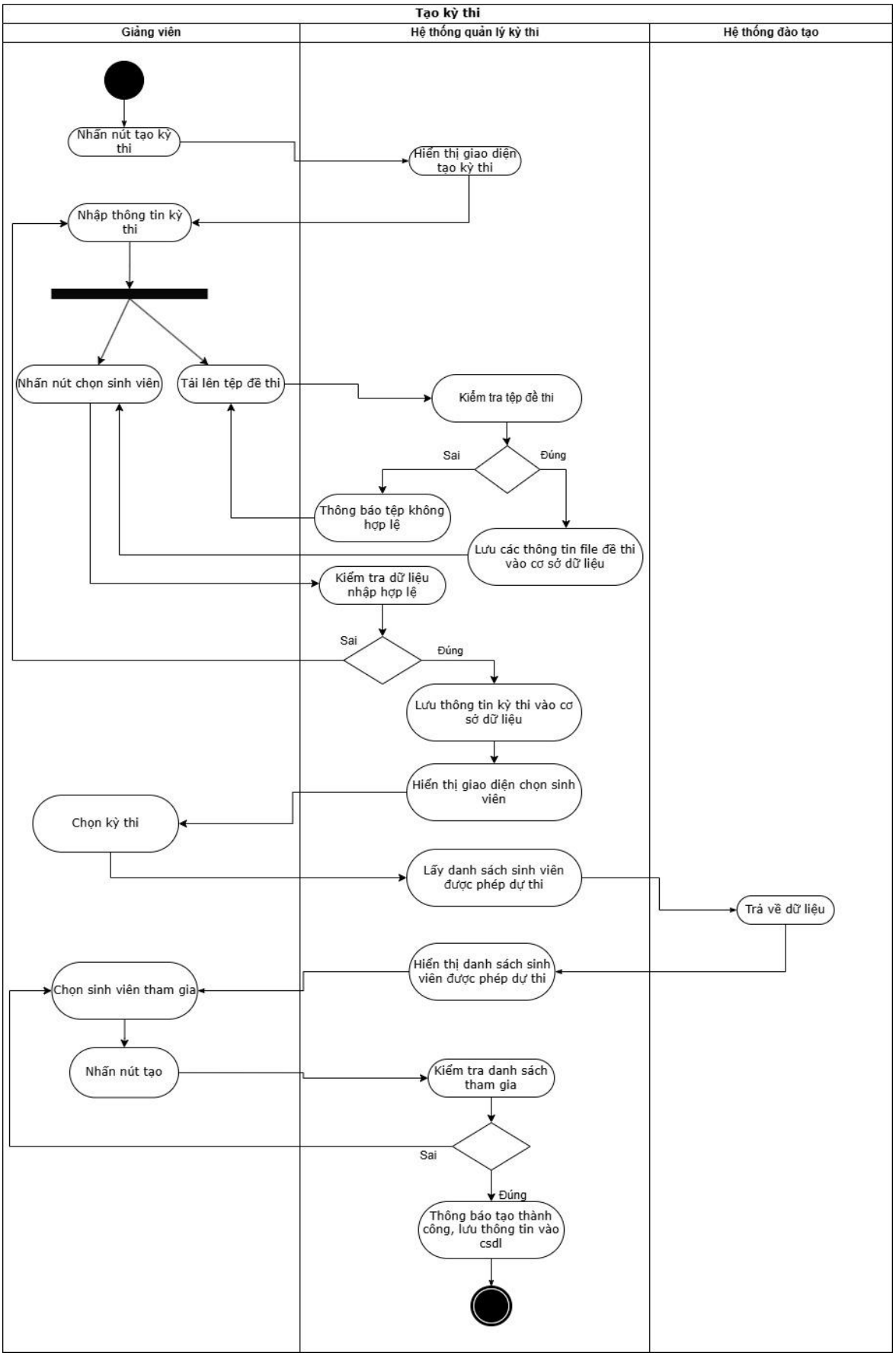
2.2.3. Biểu đồ hoạt động:

2.2.3.1. Đăng nhập:



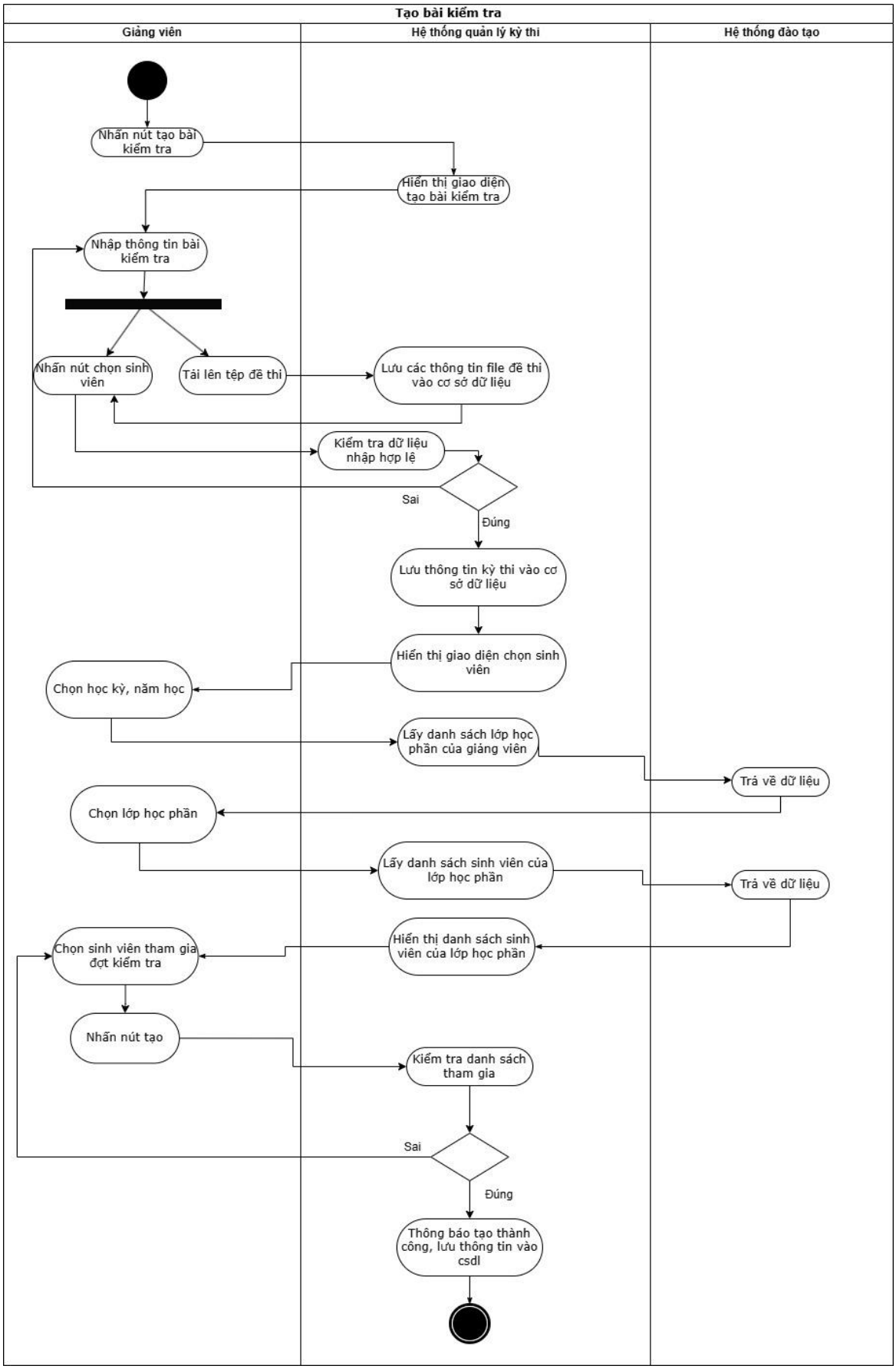
Hình 2.8 - Biểu đồ hoạt động đăng nhập

2.2.3.2. Tạo kỳ thi



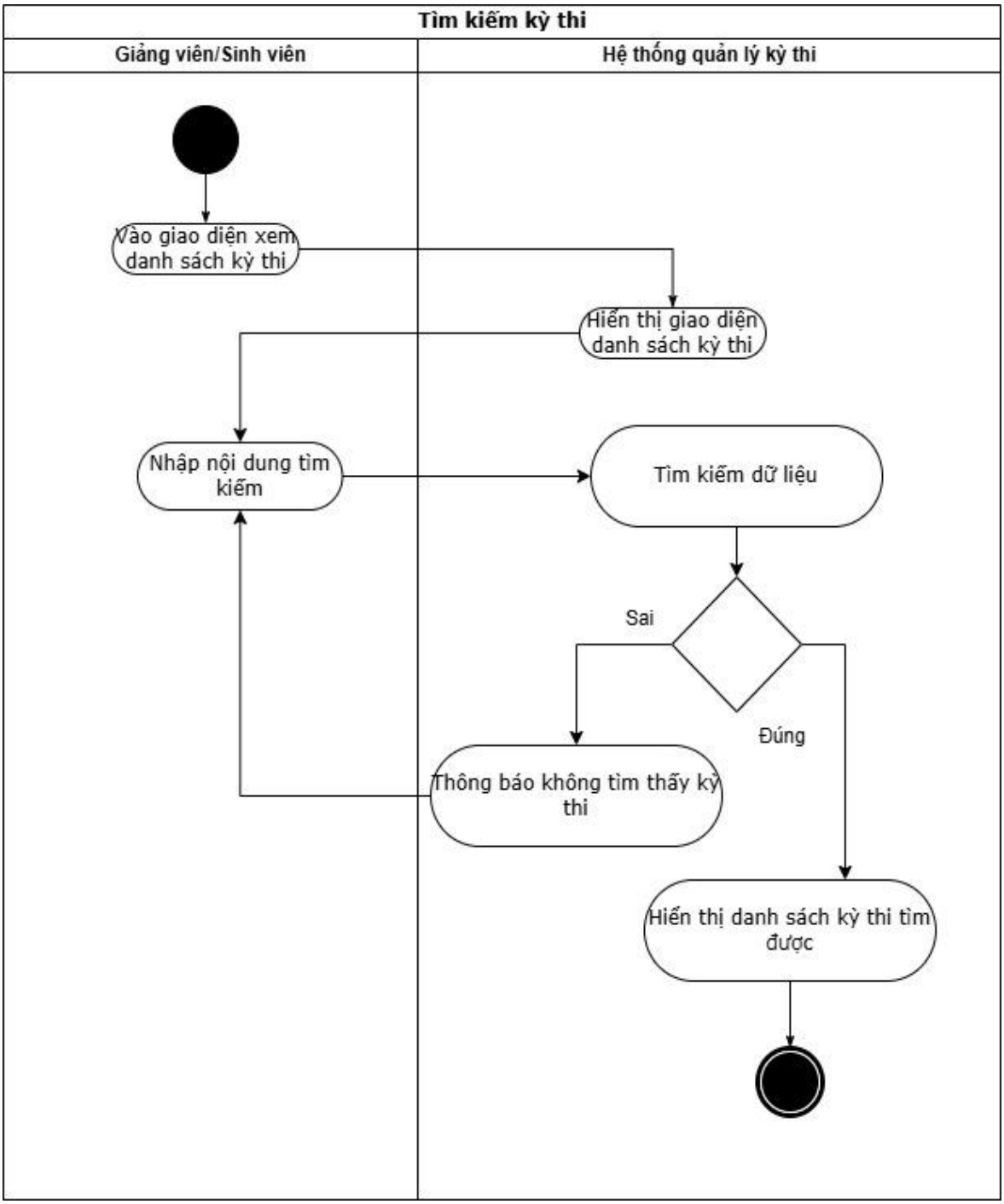
Hình 2.9 - Biểu đồ hoạt động tạo kỳ thi

2.2.3.3. Tạo bài kiểm tra:



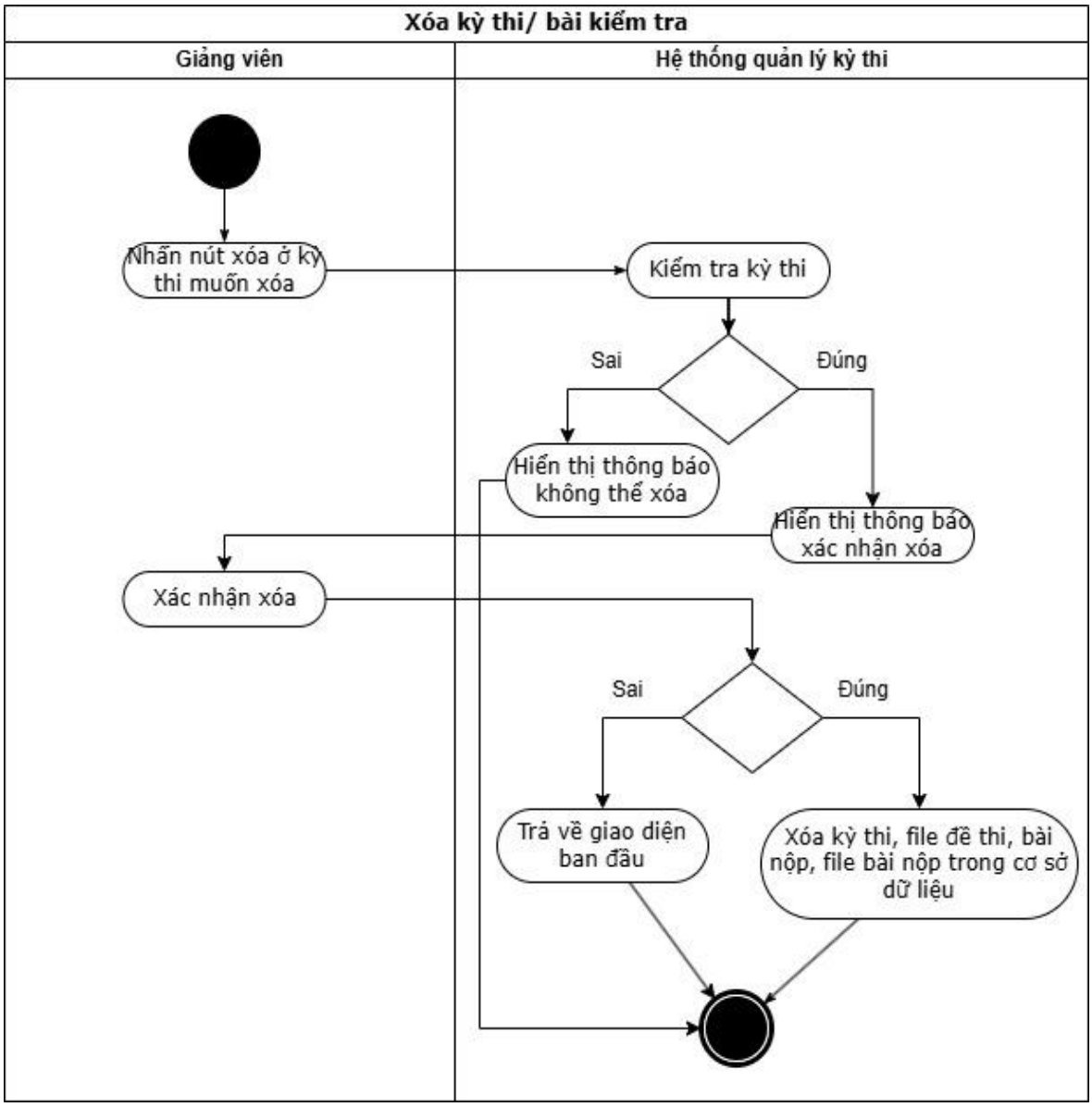
Hình 2.10 - Biểu đồ hoạt động tạo bài kiểm tra

2.2.3.4. Tìm kiếm kỳ thi/ bài kiểm tra:



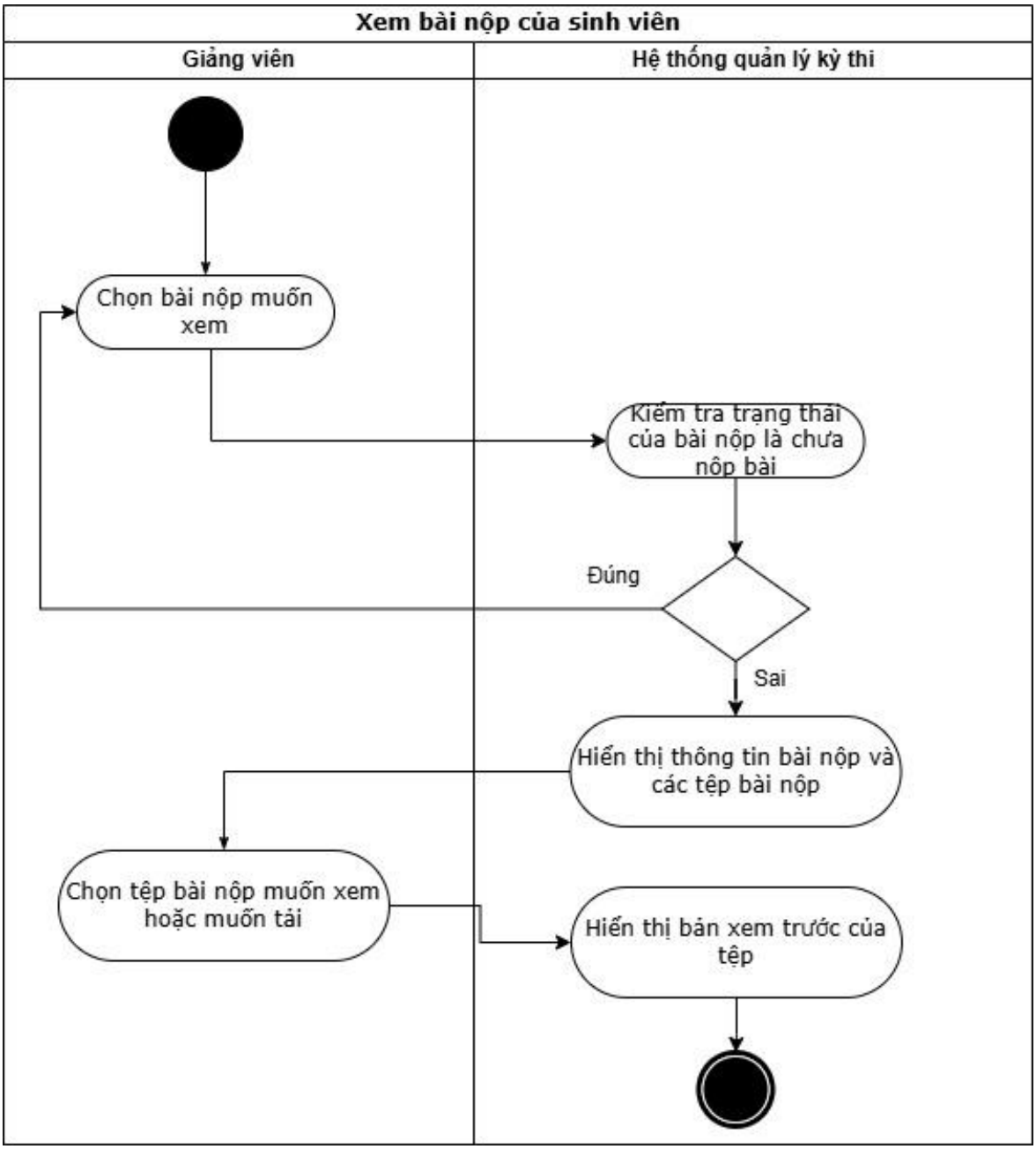
Hình 2.11 - Biểu đồ hoạt động tìm kiếm kỳ thi/ bài kiểm tra

2.2.3.5. Xóa kỳ thi/ bài kiểm tra đã tạo:



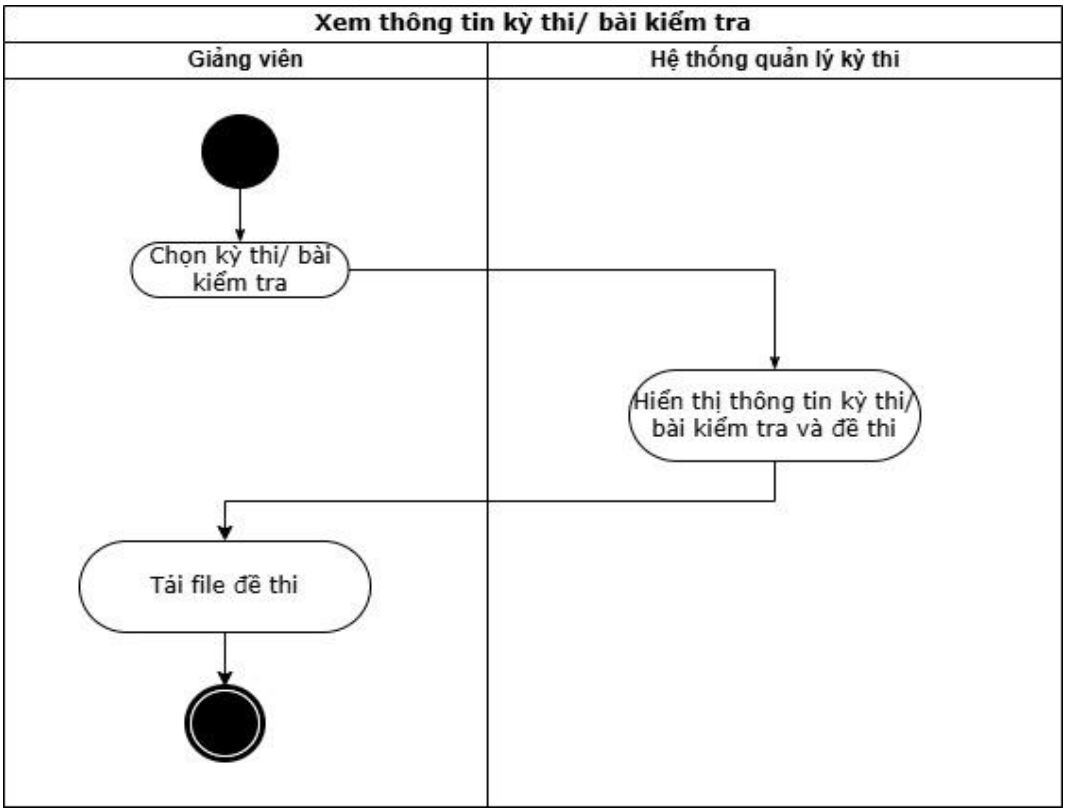
Hình 2.12 - Biểu đồ hoạt động xóa kỳ thi/ bài kiểm tra

2.2.3.6. Xem bài nộp của sinh viên:

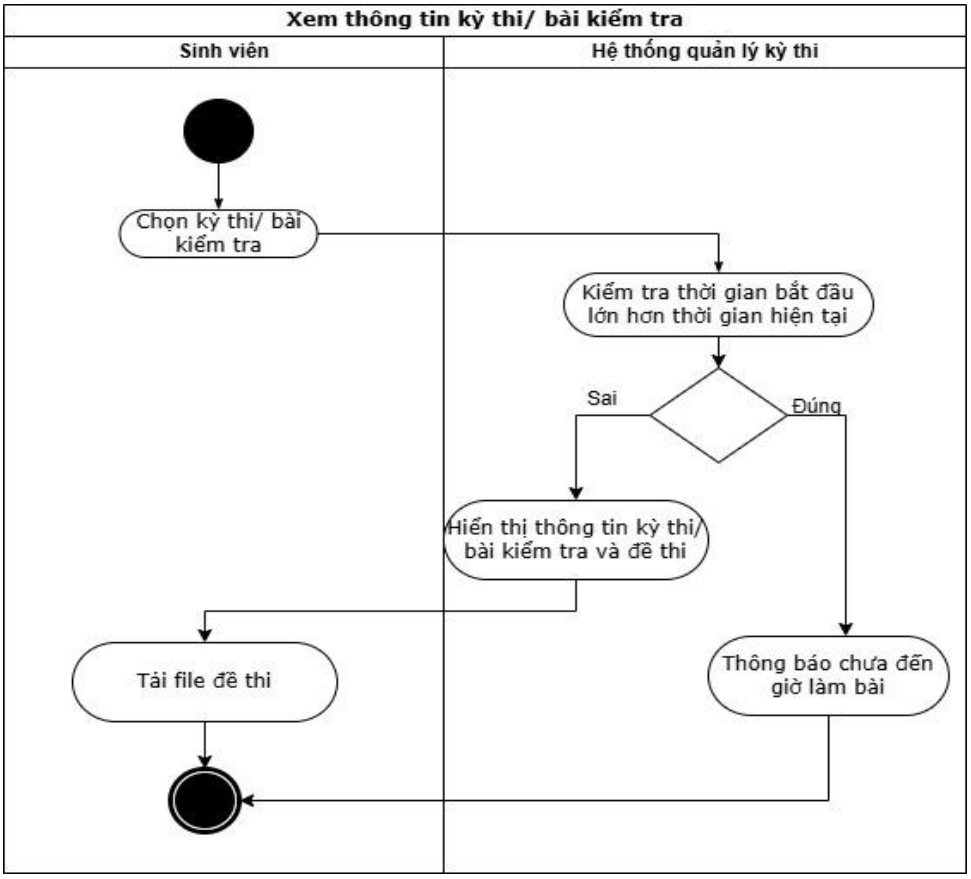


Hình 2.13 - Biểu đồ hoạt động xem bài nộp của sinh viên

2.2.3.7. Xem thông tin kỳ thi/ bài kiểm tra:

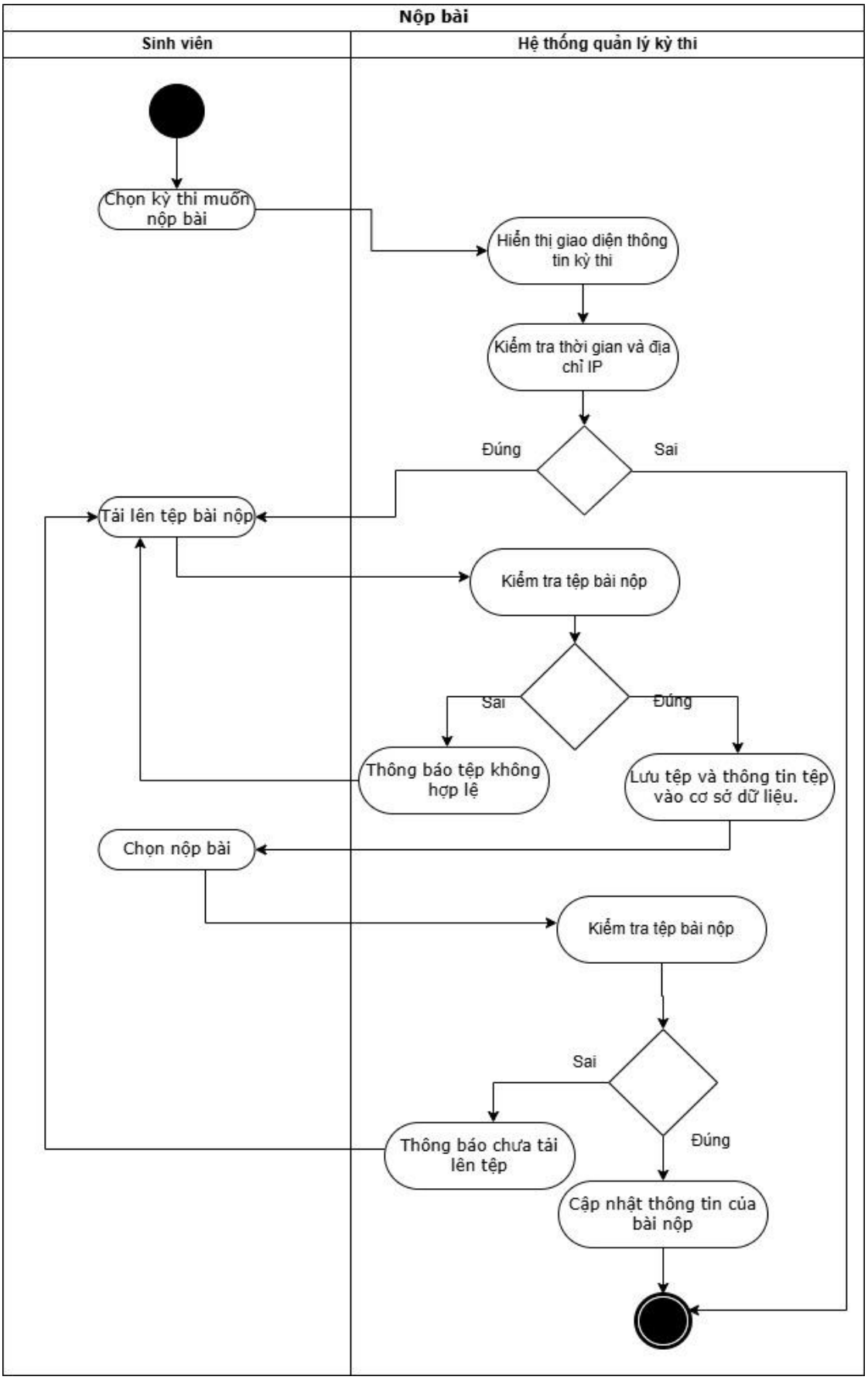


Hình 2.14 - Biểu đồ hoạt động xem thông tin kỳ thi (giảng viên)



Hình 2.15 - Biểu đồ hoạt động xem thông tin kỳ thi (sinh viên)

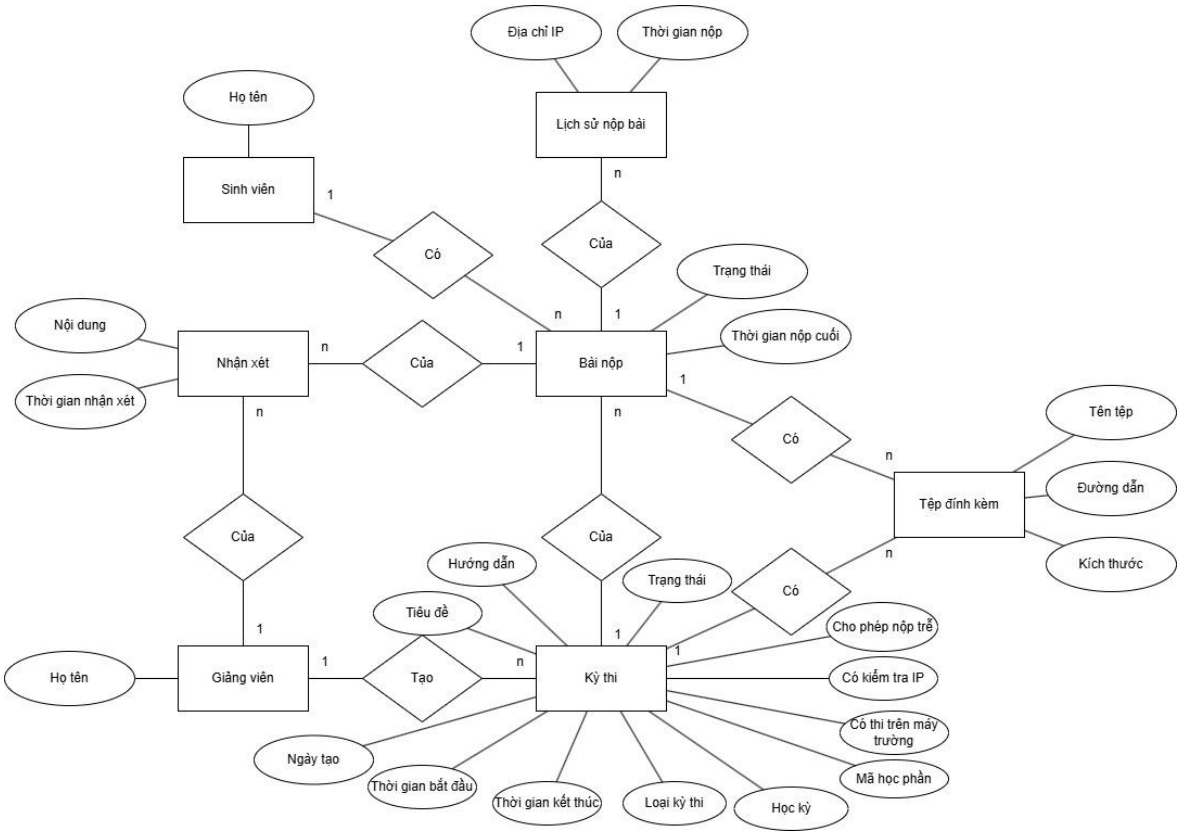
2.2.3.8. Nộp bài thi:



Hình 2.16 - Biểu đồ hoạt động nộp bài thi

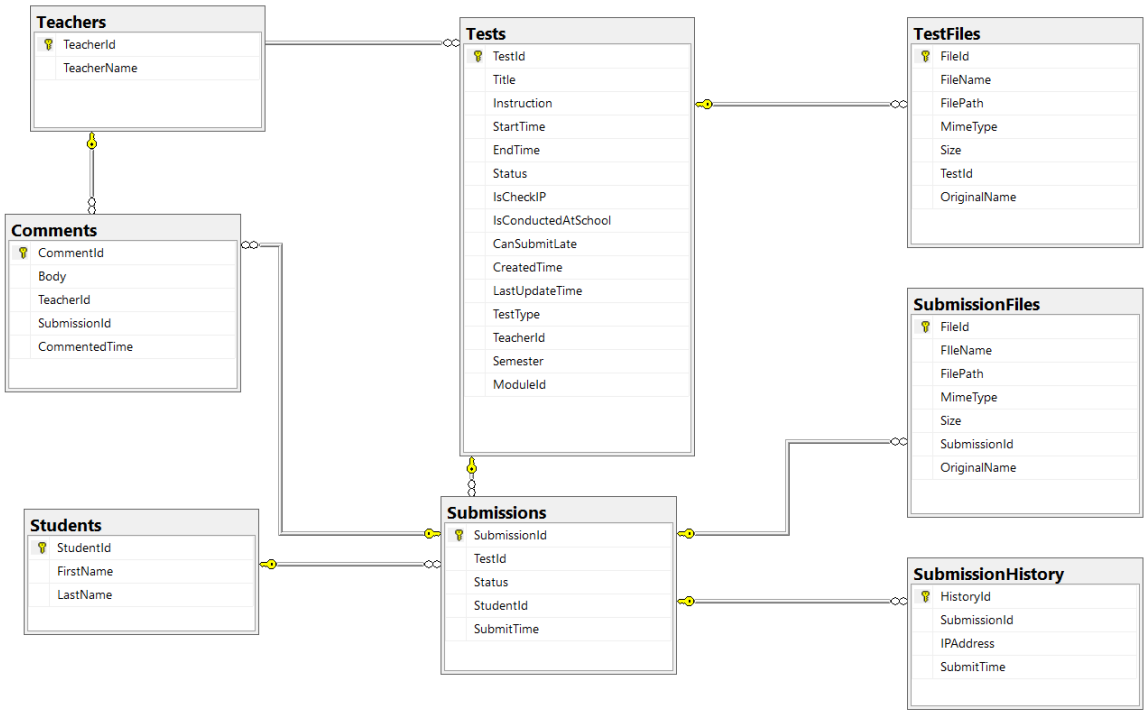
2.2.4. Thiết kế cơ sở dữ liệu:

2.2.4.1. ERD



Hình 2.17 - ERD

2.2.4.2. Cơ sở dữ liệu



Hình 2.17 – Lược đồ Cơ sở Dữ liệu Quan hệ

2.2.4.3. Chi tiết các bảng dữ liệu

Bảng 2.1 - Bảng giảng viên

Column Name	Data Type	Constraint	Description
TeacherId	varchar(50)	Primary key	Mã giảng viên
TeacherName	nvarchar(60)	Not null	Họ và tên của giảng viên

Bảng 2.2 - Bảng sinh viên

Column Name	Data Type	Constraint	Description
StudentId	varchar(50)	Primary key	Mã sinh viên
FirstName	nvarchar(20)	Not null	Tên của sinh viên
LastName	nvarchar(40)	Not null	Họ đệm của sinh viên

Bảng 2.3 - Bảng kỳ thi

Column Name	Data Type	Constraint	Description
TestId	int	Primary key	Mã kỳ thi
Title	nvarchar(255)	Not null	Tiêu đề
Instruction	nvarchar(MAX)		Hướng dẫn
StartTime	datetime		Thời gian bắt đầu
EndTime	datetime		Thời gian kết thúc
Status	nvarchar(20)	Not null	Trạng thái
IsCheckIP	bit	Not null	Kiểm tra địa chỉ IP
IsConductedAtSchool	bit	Not null	Thi trên máy trường

CanSubmitLate	bit	Not null	Có thể nộp muộn
CreatedTime	datetime	Not null	Ngày tạo
LastUpdateTime	datetime		Lần cuối cập nhật
TestType	nvarchar(20)	Not null	Loại kỳ thi
TeacherId	varchar(50)	Not null	Mã giảng viên tạo kỳ thi

Bảng 2.4 - Bảng tệp đề thi

Column Name	Data Type	Constraint	Description
FileId	uniqueidentifier	Primary key	Mã tệp
FileName	nvarchar(MAX)	Not null	Tên tệp lưu ở server
FilePath	nvarchar(MAX)	Not null	Đường dẫn đến tệp
MimeType	nvarchar(255)	Not null	Loại mime
Size	decimal(18, 0)	Not null	Kích cỡ
TestId	int	Not null	Mã kỳ thi
OriginalName	nvarchar(MAX)	Not null	Tên gốc của tệp

Bảng 2.5 - Bảng bài nộp

Column Name	Data Type	Constraint	Description
SubmissionId	int	Primary key	Mã bài nộp
TestId	int	Not null	Mã kỳ thi

SubmitTime	datetime		Thời gian nộp
Status	nvarchar(20)	Not null	Trạng thái
StudentId	varchar(50)	Not null	Mã sinh viên

Bảng 2.6 – Bảng lịch sử nộp bài

Column Name	Data Type	Constraint	Description
HistoryId	int	Primary key	Mã lịch sử nộp bài
SubmissionId	int	Not null	Mã bài nộp
IPAddress	varchar(50)	Not null	Địa chỉ IP nộp bài
SubmitTime	datetime	Not null	Thời gian nộp bài

Bảng 2.7 - Bảng tệp bài nộp

Column Name	Data Type	Constraint	Description
FileId	uniqueidentifier	Primary key	Mã tệp
FileName	nvarchar(MAX)	Not null	Tên tệp lưu ở server
FilePath	nvarchar(MAX)	Not null	Đường dẫn đến tệp
MimeType	nvarchar(255)	Not null	Loại mime
Size	decimal(18, 0)	Not null	Kích cỡ

SubmissiontId	int	Not null	Mã bài nộp
OriginalName	nvarchar(MAX)	Not null	Tên gốc của tệp

Bảng 2.8 - Bảng nhận xét

Column Name	Data Type	Constraint	Description
CommentId	int	Primary key	Mã nhận xét
Body	nvarchar(255)	Not null	Nội dung
TeacherId	varchar(50)	Not null	Mã giảng viên nhận xét
SubmissionId	int	Not null	Mã bài nộp
CommentedTime	datetime	Not null	Thời gian nhận xét

Chương 3. KẾT QUẢ

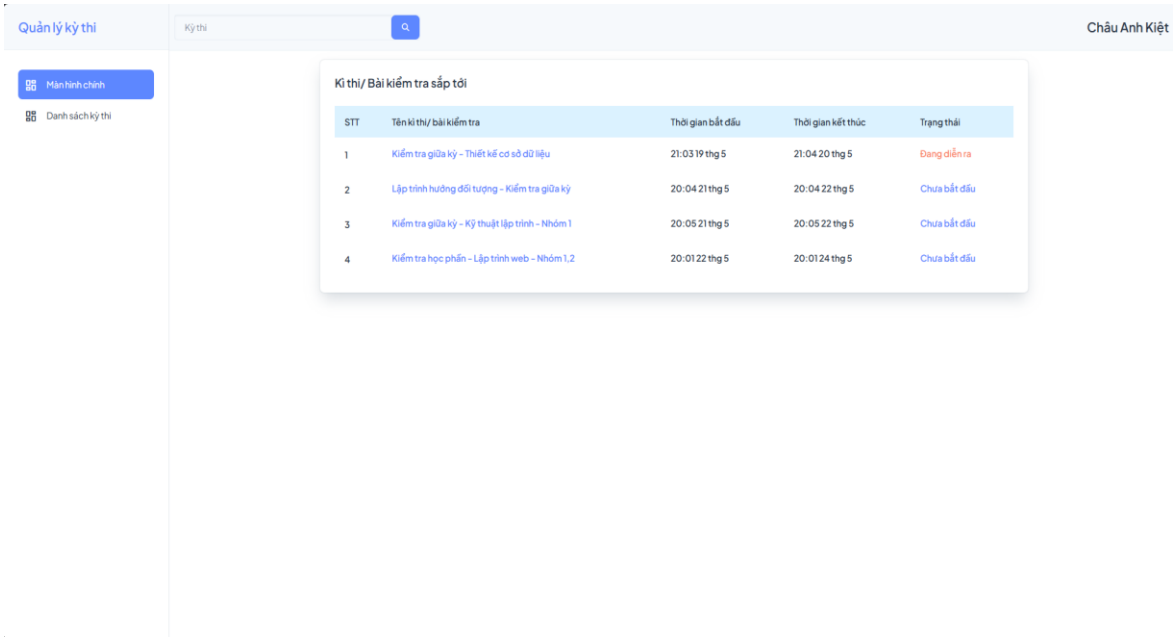
3.1. Giao Diện:

3.1.1. Đăng nhập:

Hình 3.1- Đăng nhập

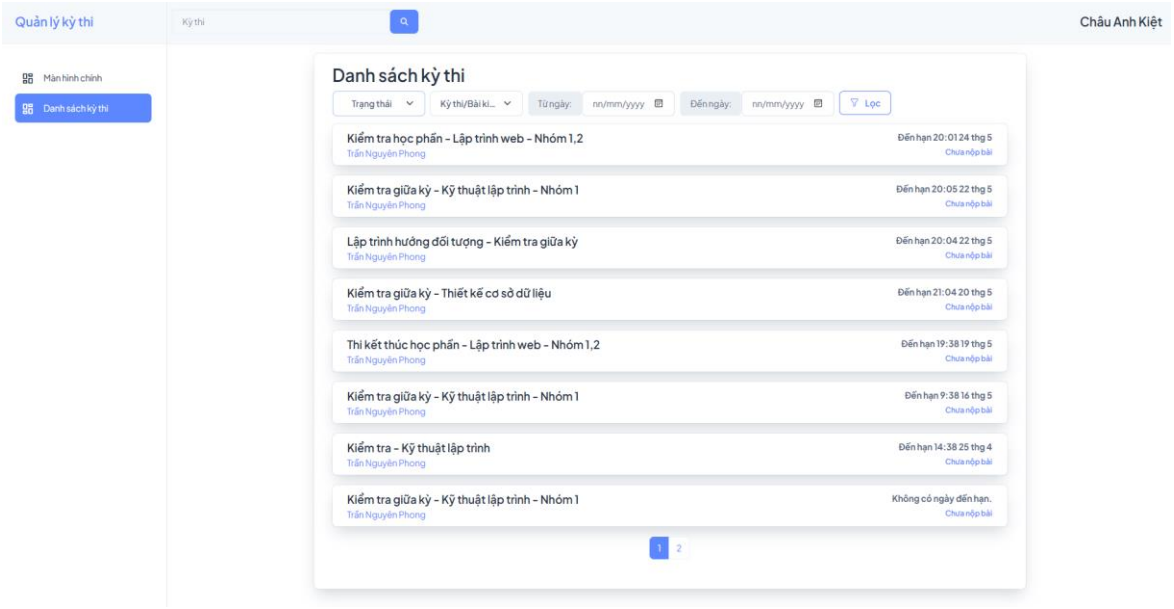
3.1.2. Trang sinh viên

3.1.2.1. Trang chủ



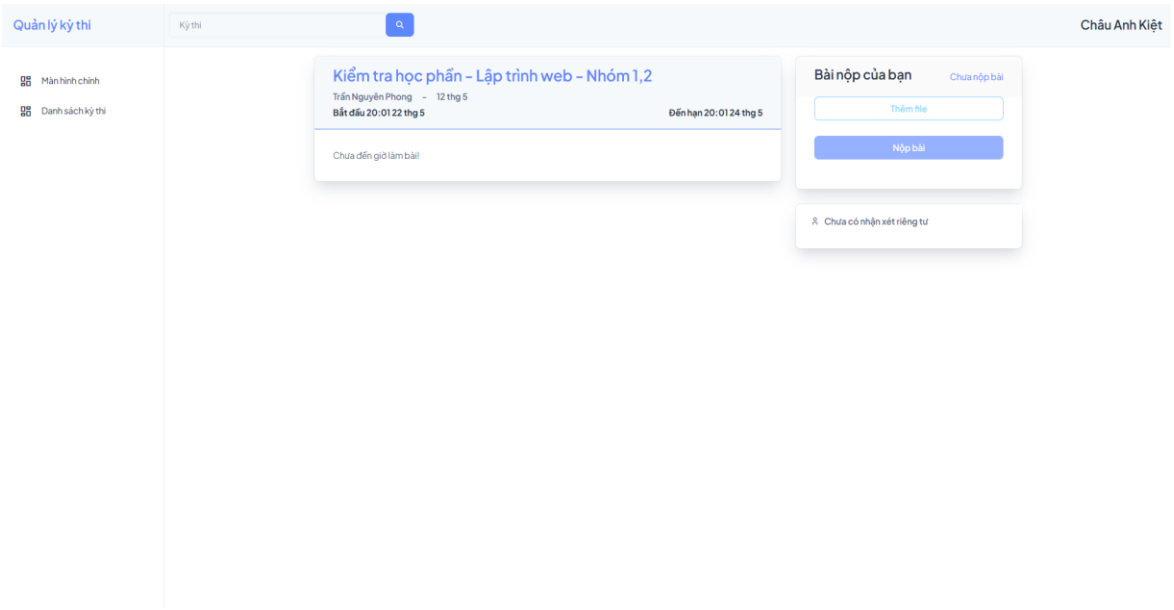
Hình 3.2 - Trang chủ (sinh viên)

3.1.2.2. Danh sách kỳ thi

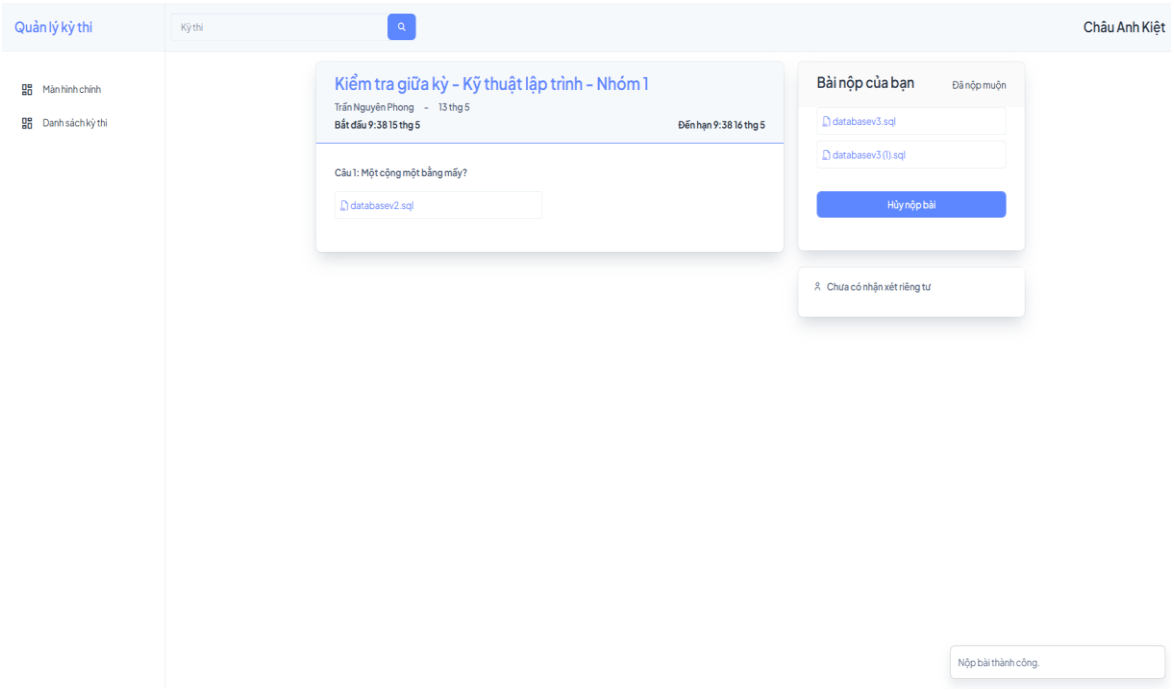


Hình 3.3 - Danh sách kỳ thi (sinh viên)

3.1.2.3. Thông tin kỳ thi



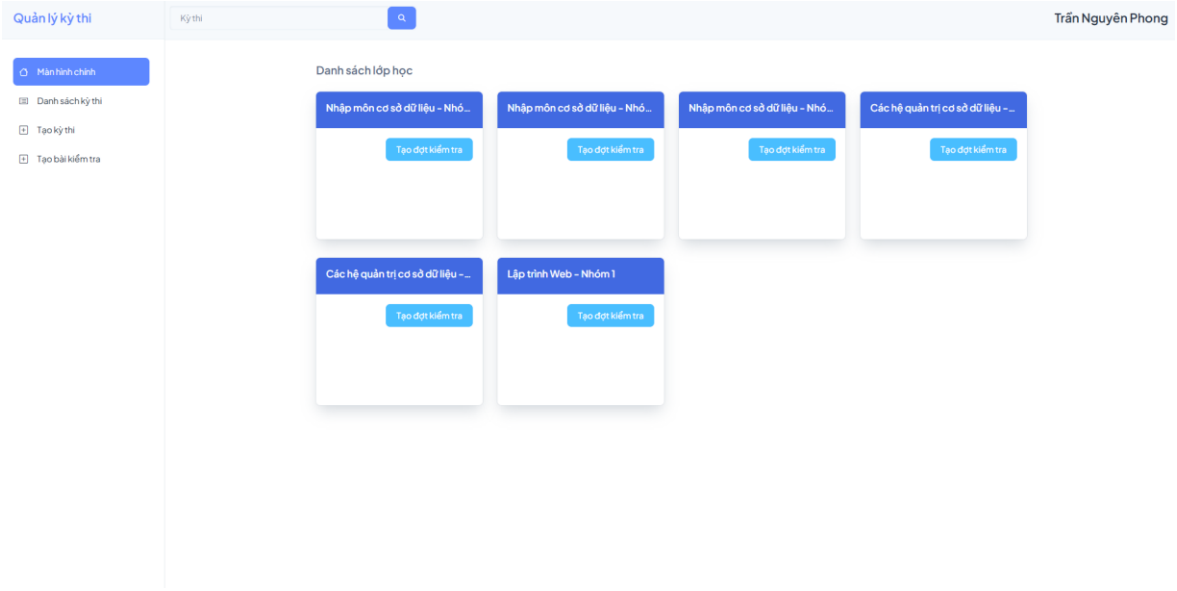
Hình 3.4 - Thông tin kỳ thi chưa bắt đầu (sinh viên)



Hình 3.5 - Thông tin kỳ thi đã bắt đầu (sinh viên)

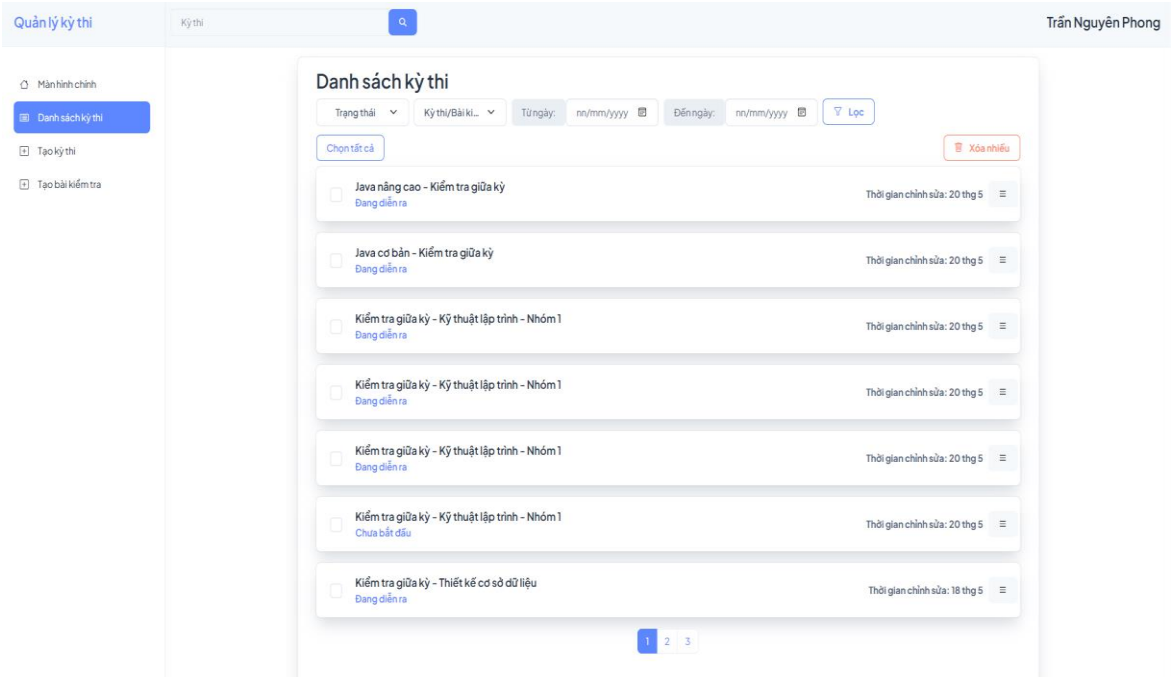
3.1.3. Trang giảng viên

3.1.3.1. Trang chủ



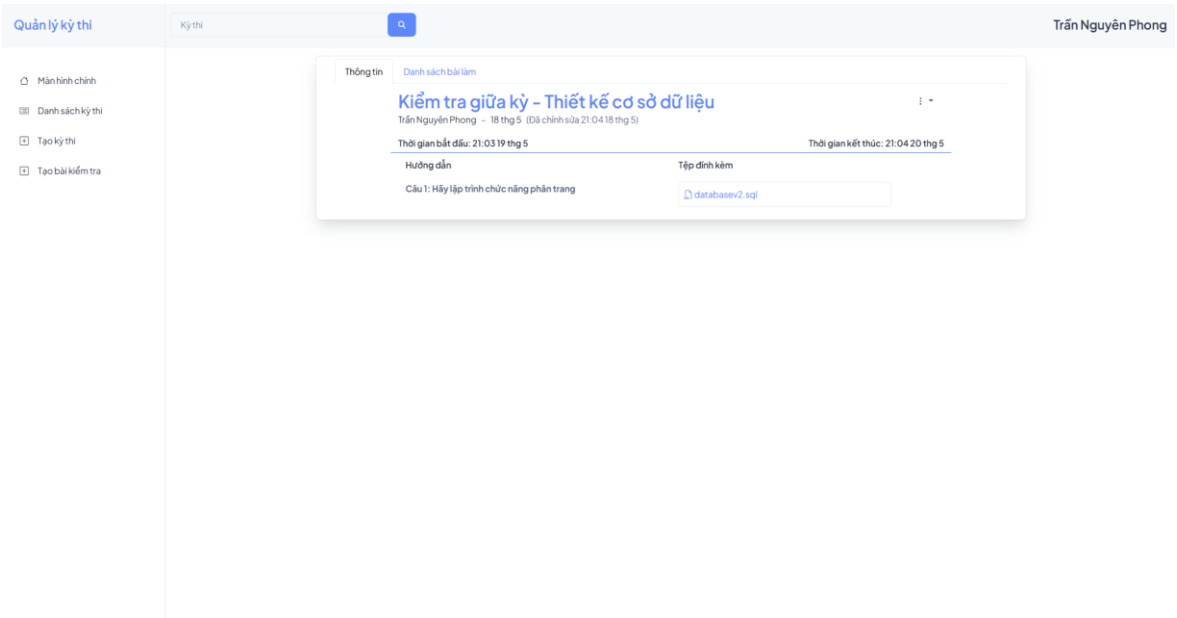
Hình 3.6 – Trang chủ (giảng viên)

3.1.3.2. *Danh sách kỳ thi*

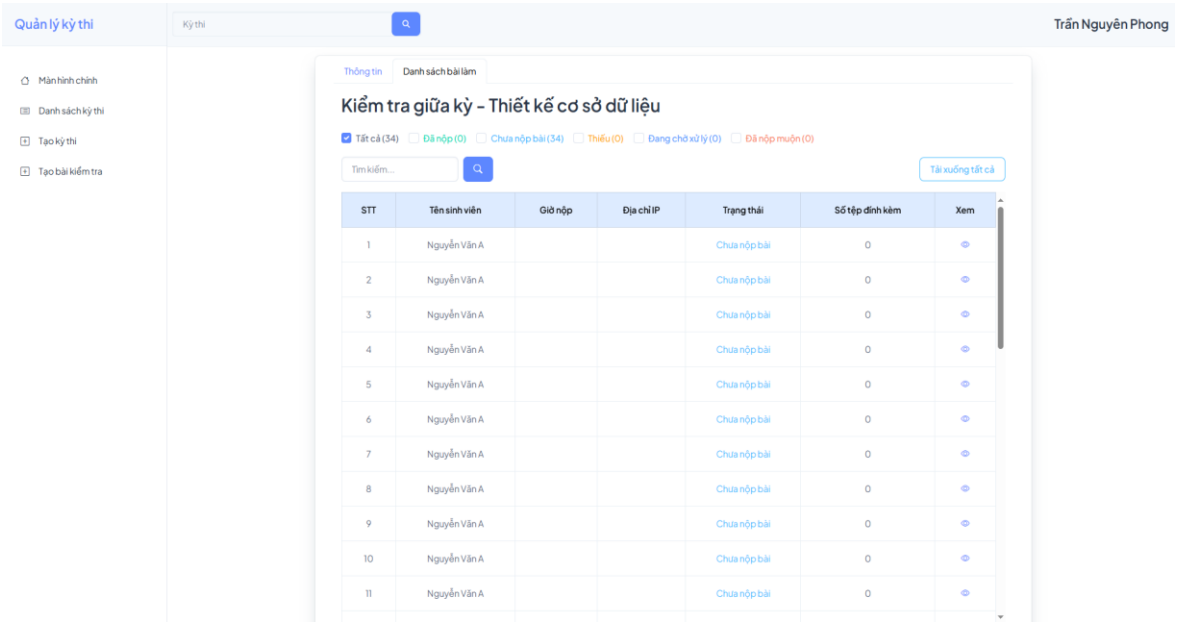


Hình 3.6 - Danh sách kỳ thi (giảng viên)

3.1.3.3. *Xem thông tin kỳ thi*

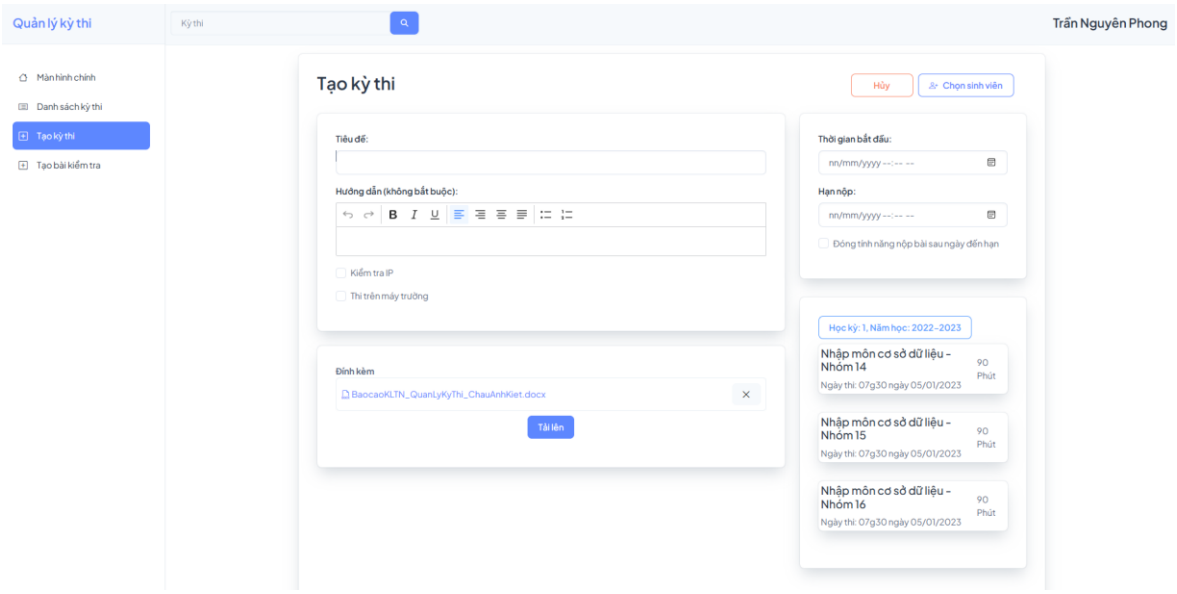


Hình 3.7 - Thông tin kỳ thi (giảng viên)



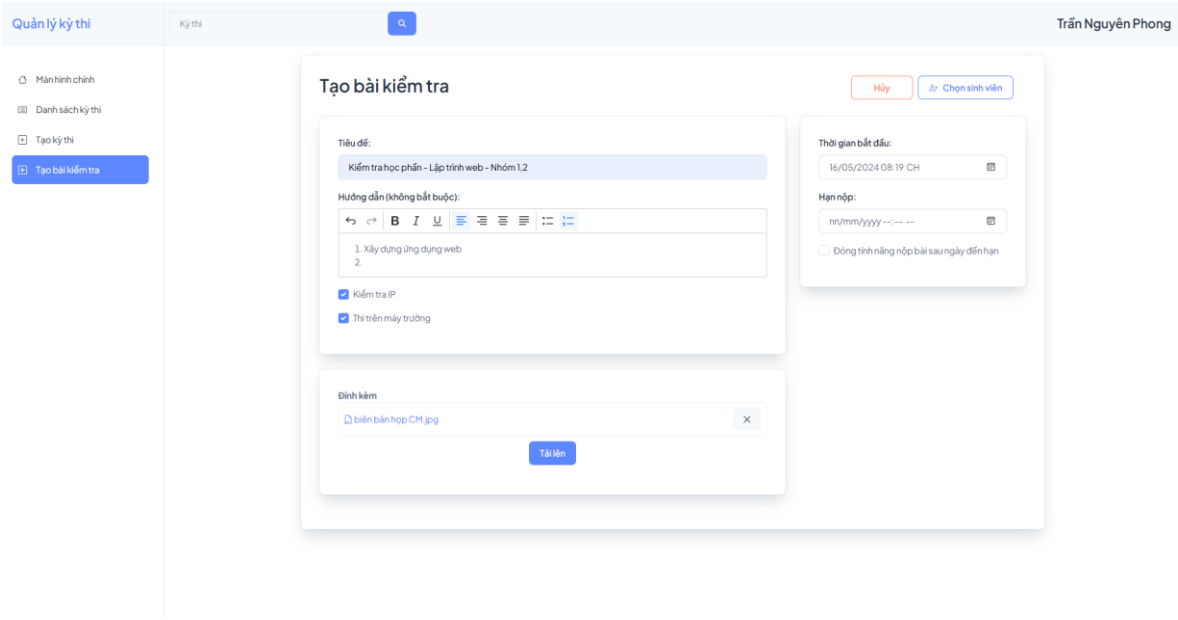
Hình 3.8 - Danh sách bài nộp

3.1.3.4. Tạo kỳ thi

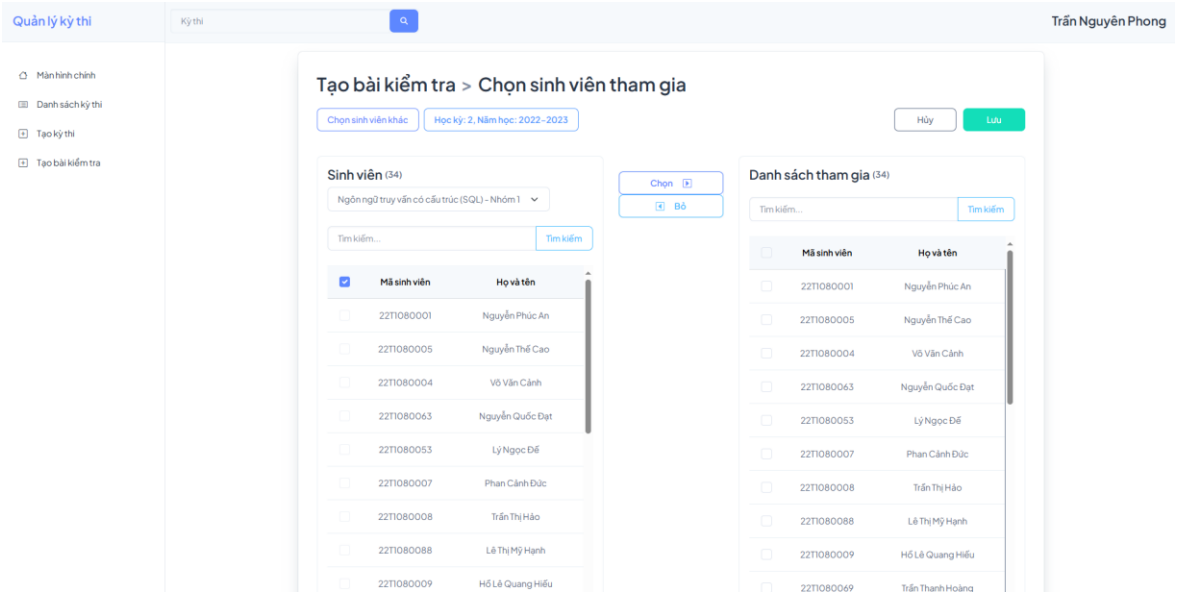


Hình 3.9 - Tạo kỳ thi

3.1.3.5. Tạo bài kiểm tra

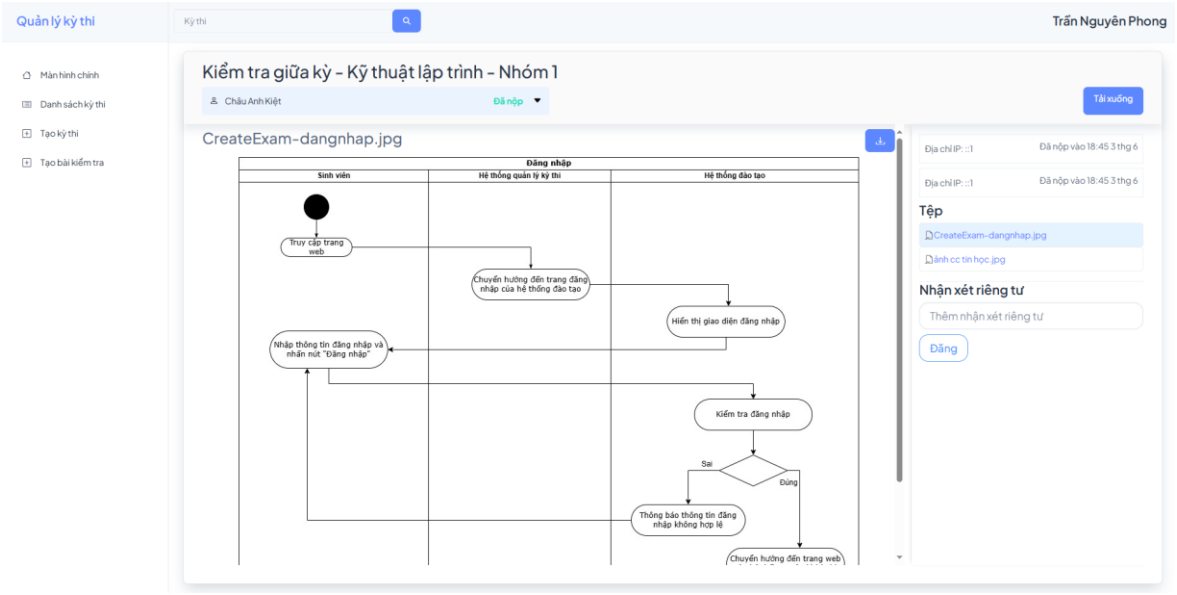


Hình 3.10 - Tạo bài kiểm tra



Hình 3.11 - Chọn sinh viên tham gia

3.1.3.6. Xem thông tin bài nộp



Hình 3.12 - Xem thông tin bài nộp

KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

1. Kết luận

Trong bối cảnh hiện nay, việc quản lý kỳ thi và nộp bài thi qua Google Classroom đã bộc lộ nhiều bất cập, đặc biệt là khó khăn trong việc kiểm soát gian lận và sự không phù hợp với yêu cầu cụ thể của trường Đại học Khoa học Huế. Việc phát triển một hệ thống quản lý kỳ thi và nộp bài thi mới là cần thiết để đáp ứng nhu cầu thực tế của giảng viên và sinh viên.

Hệ thống mới này không chỉ giúp tối ưu hóa quá trình tạo và quản lý kỳ thi, mà còn cung cấp các công cụ kiểm tra gian lận hiệu quả, đảm bảo tính công bằng và minh bạch trong quá trình thi cử. Hơn nữa, với giao diện thân thiện, quy trình rõ ràng, dễ sử dụng hệ thống sẽ giúp giảng viên tiết kiệm thời gian và nâng cao hiệu quả làm việc, đồng thời giúp sinh viên dễ dàng tiếp cận và sử dụng.

2. Hướng phát triển

- Cải tiến và mở rộng tính năng: Liên tục cải tiến và bổ sung các tính năng mới dựa trên phản hồi của giảng viên và sinh viên, như tích hợp công cụ AI để tự động phát hiện gian lận hoặc cung cấp các báo cáo chi tiết về kết quả thi.

- Hỗ trợ đa nền tảng: Đảm bảo hệ thống hoạt động tốt trên nhiều nền tảng và thiết bị khác nhau, bao gồm cả máy tính, máy tính bảng và điện thoại di động, để tăng cường sự tiện lợi và linh hoạt cho người dùng.

- Bảo mật và quyền riêng tư: Tiếp tục nâng cao các biện pháp bảo mật để bảo vệ dữ liệu cá nhân của sinh viên và giảng viên, đảm bảo rằng hệ thống tuân thủ các quy định về bảo vệ dữ liệu.

- Đào tạo và hỗ trợ người dùng: Cung cấp tài liệu hướng dẫn chi tiết để giảng viên và sinh viên có thể sử dụng hệ thống một cách hiệu quả. Cung cấp kênh hỗ trợ kỹ thuật để giải quyết các vấn đề phát sinh kịp thời.

- Phân tích và báo cáo: Phát triển các công cụ phân tích và báo cáo để giảng viên có thể theo dõi tiến độ và hiệu suất của sinh viên, từ đó đưa ra các điều chỉnh cần thiết trong quá trình giảng dạy.

Việc triển khai và phát triển hệ thống quản lý kỳ thi và nộp bài thi mới này sẽ góp phần nâng cao chất lượng giáo dục tại trường Đại học Khoa học Huế, đồng thời tạo ra một môi trường học tập và thi cử công bằng, hiệu quả và thuận tiện cho tất cả các bên liên quan.

TÀI LIỆU THAM KHẢO

[1]. Wikipedia. ASP.NET Core, https://en.wikipedia.org/wiki/ASP.NET_Core (22/04/2024)

[2]. Microsoft (2023). *Overview of ASP.NET Core MVC*, https://learn.microsoft.com/en-us/aspnet/core/mvc/overview?view=aspnetcore-8.0 (22/04/2024).

[3]. Microsoft (2024). *ASP.NET Core Middleware*, https://learn.microsoft.com/en-us/aspnet/core/fundamentals/middleware/?view=aspnetcore-8.0 (22/04/2024).

[4]. Dusan Petkovic (2019). Microsoft SQL Server 2019: A Beginner's Guide.

[5]. Lotusacademy (2023). Microsoft Sql Server là gì? Phần mềm Sql Server là gì? Tổng quan về SQL Server. https://lotusacademy.edu.vn/blog/microsoft-sql-server-la-gi-phan-mem-sql-server-la-gi-tong-quan-ve-sql-server-284 (22/04/2024)

[5]. W3School. jQuery Tutorial. https://www.w3schools.com/jquery/jquery_intro.asp (22/04/2024)

[6]. W3School. Bootstrap 5 Tutorial.

https://www.w3schools.com/bootstrap5/index.php (22/04/2024)

[7] Michelle Nguyen (2023). Kỹ thuật thiết kế kiến trúc phần mềm trong .NET: Phân tích sâu về khuôn mẫu thiết kế kiến trúc phổ biến. https://viblo.asia/p/ky-thuat-thiet-ke-kien-truc-phan-mem-trong-net-phan-tich-sau-ve-khuon-mau-thiet-ke-kien-truc-pho-bien-GAWVpOY3L05 (22/04/2024)

[8] Charles Chung (2023). Tìm hiểu Dapper Micro ORM và sử dụng trong ASP.NET Core Web API. https://hanam88.com/kho-tai-lieu/63/109/tim-hieu-dapper-micro-orm-va-su-dung-trong-asp-net-core-web-api.html (22/04/2024)

[9] Thảo Meo (2022). [C#] Giới thiệu và sử dụng thư viện AutoMapper. https://laptrinhvb.net/bai-viet/chuyen-de-csharp/---Csharp----Gioi-thieu-va-su-dung-thu-vien-AutoMapper-/5cadd38af86180dd.html (22/04/2024)