TRƯỜNG ĐẠI HỌC YERSIN ĐÀ LẠT

KHOA CÔNG NGHỆ THÔNG TIN



BÁO CÁO MÔN HỌC

LẬP TRÌNH WEB 2

Hệ Thống Quản Lý Film

GVHD : Nguyễn Đức Tấn

SVTH - Mã số SV : Phạm Thanh Sang – 12101038

Cao Gia Khang - 12101022

Nguyễn Ái Nhi - 12101017

Khóa học : 2023 - 2024

Đà Lạt, tháng 6 - 2024

PHẦN NHẬN XÉT CỦA GIẢNG VIÊN

Điểm:

Ngày 14 tháng 6 năm 2024

*Ký và ghi rõ họ tên*



GIỚI THIỆU

Trong kỷ nguyên số hiện nay, Web API (Application Programming Interface) đã trở thành một thành phần không thể thiếu trong việc phát triển các ứng dụng web và di động. Web API cho phép các ứng dụng giao tiếp với nhau, chia sẻ dữ liệu và chức năng một cách liền mạch, bất kể nền tảng hay công nghệ sử dụng. Từ các dịch vụ mạng xã hội, thương mại điện tử đến các ứng dụng tài chính, Web API đóng vai trò quan trọng trong việc kết nối và mở rộng khả năng của các hệ thống.

Trong bài báo cáo này nhóm em sẽ trình bày một cái nhìn tổng quan về Web API, bao gồm khái niệm, nguyên lý hoạt động, và lợi ích mà nó mang lại. Chúng ta cũng sẽ khám phá các phương pháp tốt nhất khi thiết kế và triển khai Web API, cùng với các công nghệ và công cụ hỗ trợ phổ biến hiện nay. Thông qua các ví dụ và trường hợp thực tế, báo cáo sẽ minh họa cách Web API đã và đang cách mạng hóa cách mà chúng ta xây dựng và tương tác với các ứng dụng web.

Em thay mặt nhóm em xin cảm ơn thầy Nguyễn Đức Tấn đã có những hướng dẫn và sự giúp đỡ của thầy đã góp phần cho bài báo cáo về ‘Hệ thống quản lý hình ảnh’ trở nên hoàn thiện hơn. Em xin chân thành cảm ơn

DANH MỤC TỪ NGỮ VIẾT TẮT

|  |  |  |  |
| --- | --- | --- | --- |
| STT | Tên viết tắt | Tên đầy đủ | Dịch ra tiếng Việt  (Nếu là tiếng nước ngoài) |
| 1 | ORM | (Object-Relational Mapping) | Ánh xạ quan hệ đối tượng |
| 2 | MVC | (Model-View-Controller) | … |
| 3 | CSDL | Cơ Sở Dữ Liệu | ... |
| 4 | DBMS | Database Management  System | Hệ quản trị cơ sở dữ liệu |
| 5 | API | Application Programming Interface | Giao diện lập trình ứng dụng |
| 6 | HTTP | **HyperText Transfer Protocol**. | Giao thức dùng để truyền tải siêu văn bản |
| 7 | HTTPS | HyperText Transfer Protocol Secure | Phiên bản bảo mật của HTTP |
| 8 | REST | Representational State Transfer | ... |
| 9 | RESTful |  | ... |
| 10 | JSON | JavaScript Object Notation | ... |
| 11 | OAuth2 | Open Authorization 2.0 | ... |
| 12 | JWT | JSON Web Token | ... |

DANH MỤC HÌNH ẢNH

|  |  |
| --- | --- |
| STT | Mô tả |
| 1 | Hình 1. 1. Cách thức hoạt động của Web API |
| 2 | Hình 1. 2. User case hệ thống |
| 3 | Hình 1. 3. User case cho Admin |
| 4 | Hình 1. 4. User case cho User |
| 5 | Hình 1. 5. Sơ đồ quan hệ |
| 6 | Hình 1. 6. Giao diện trang chủ |
| 7 | Hình 1. 7. Giao diện đăng ký tài khoản |
| 8 | Hình 1. 8. Giao diện đăng nhập tài khoản |
| 9 | Hình 1. 9. Giao diện quản lý film |
| 10 | Hình 1. 10. Giao diện quản lý hình ảnh |

BÁO CÁO TIẾN ĐỘ

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Nội dung công việc** | **Phân công** | **T37** | **T38** | **T39** | **T40** | **T41** | **T42** | **T43** | **T44** | **T45** | **T46** | **T47** | **T48** | **Hoàn thành** |
| Phân công công việc |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Tìm hiểu lý thuyết và viết nội dung chương 1 | Khang | **x** | **x** | **x** | **x** |  |  |  |  |  |  |  |  | **☑** |
| Tìm hiểu lý thuyết và viết nội dung chương 2 | Khang | **x** | **x** | **x** | **x** |  |  |  |  |  |  |  |  | **☑** |
| Tìm hiểu lý thuyết và viết nội dung chương 3 | Khang | **x** | **x** | **x** | **x** |  |  |  |  |  |  |  |  | **☑** |
| Tìm hiểu lý thuyết và viết nội dung chương 4 | Khang | **x** | **x** | **x** | **x** |  |  |  |  |  |  |  |  | **☑** |
| Chuẩn bị nội dung bài báo cáo | Sang |  |  |  |  | **x** | **x** | **x** | **x** | **x** | **x** | **x** | **x** | **☑** |
| Chuẩn bị nội dung slide thuyết trình | Sang  Khang |  |  |  |  | **x** | **x** | **x** | **x** | **x** | **x** | **x** | **x** | **☑** |
| Thống nhất nội dung thuyết trình | Sang  Khang |  |  |  |  |  |  | **x** | **x** | **x** | **x** | **x** | **x** | **☑** |
| Xây Dựng Slide | Khang |  |  |  |  |  |  |  |  |  | **x** | **x** | **x** |  |
| Thuyết trình | Cả nhóm |  |  |  |  |  |  |  |  |  |  |  | **x** | **☑** |
| Tổng hợp và chỉnh sửa báo cáo | Sang  Khang |  |  |  |  |  |  |  | **x** | **x** | **x** | **x** | **x** | **☑** |
| Thiết kế template | Thuyền |  |  |  |  |  |  |  |  | **x** | **x** | **x** | **x** | **☑** |
| Xây dựng code backend ứng dụng Web | Sang |  |  |  |  |  |  |  |  | **x** | **x** | **x** | **x** | **☑** |
| Thu thập hình ảnh và kết quả ứng dụng Web | Sang |  |  |  |  |  |  |  |  |  |  | **x** | **x** | **☑** |
| Demo lab | Cả nhóm |  |  |  |  |  |  |  |  |  |  |  | **x** | **☑** |

MỤC LỤC

Chương 1. Kiến thức cơ bản (Cơ sở lý thuyết) …………………………....….tr 8 - 11

1.Tìm hiểu về Web API ? …………………………………………………….….........

1.1 Web API là gì ?..…………………………………………..………………….…...

1.2 Ứng dụng của Web API…………………………………………………….…......

1.3 Ưu và nhược điểm của Web API …………………………………….………..….

1.4 Cách thức hoạt động Web API……....………………………………………..…..

1.5 So sánh Web API với Web MVC……………………………..…….………….....

1. Ngôn ngữ lập trình ………. …………………………………………………….....

2.1. C#………………………………………………………………………………...

3.Các công nghệ sử dụng ………………………………………………………….....

3.1 HTTP và HTTPS………………………………………………………….………

3.2 REST và RESTful APIs……………………………..……………………….…...

3.3 JSON………………………………………………………………..…………….

3.4 OAuth2 ……………………………………………………………………………

3.5 JWT ………………………………………………………………………..……..

3.6 Authentication và Authorization…………………………………………...…..…

3.7 Entity Framework……………………………………………………..…….….…

3.8 HttpClient (ASP.NET Core)……………………………………..…..…….……..

Chương 2. Xây dựng ứng dụng Web (Triển khai thực tế)……………….….tr 11 - 16

1.Giới thiệu đề tài…………………………………………………………………......

2.Yêu cầu chức năng……………………………………………………………….....

3.Phân tích và thiết kế hệ thống ………….....……………………..……….…...........

3.1 Phân tích …………………………………………………………………...….....

3.2 Thiết kế hệ thống …………………………………………………………...........

4. API Endpoints ..........................................................................................................

5. Cơ sở dữ liệu ...........................................................................................................

Chương 3. Giao diện Consume API ..................…………………………....tr 17 - 19

Chương 4. Kết quả chương trình………………………………….......................tr 19

1. Kết luận………………………………………………………………..............….
2. Tài liệu tham khảo…………………………………………………………...........

**Chương 1. Kiến thức cơ bản (Cơ sở lý thuyết)**

1.Tìm hiểu về Web API ?

1.1. Web API là gì ?

- API (Application Programming Interfaceđược) được dịch ra là giao diện lập trình ứng dụng , Nó là các phương thức, giao thức có nhiệm vụ kết nối các thư viện cùng với ứng dụng khác nhau.

- Web API được hiểu là một phương thức thường được sử dụng với mục đích cho phép toàn bộ các ứng dụng khác nhau có thể giao tiếp cũng như trao đổi dữ liệu qua lại. Dữ liệu lúc này sẽ được Web API trả lại thường ở dạng XML hoặc dạng JSON thông qua giao thức HTTPS hay giao thức HTTP.

1.2. Ứng dụng của Web API

- Tự động hóa sản phẩm

Với Web API chúng ta có thể tự động hóa quản lý công việc cũng như cập nhật luồng công việc của mình. Từ đó giúp tăng thêm năng suất cũng như tạo hiệu quả cao hơn cho công việc của mình.

- Khả năng tích hợp đa chức năng

API cho phép lấy toàn bộ nội dung từ bất kỳ trang web hoặc ứng dụng nào một cách thuận tiện, nhanh chóng nếu được cho phép. Từ đó để có thể tăng thêm trải nghiệm cho người sử dụng.

- Cập nhật thông tin thời gian thực tế

API còn có chức năng thay đổi cũng như cập nhật thay đổi theo thời gian thực tế. Với công nghệ này, dữ liệu sẽ có khả năng truyền đi tốt hơn, thông tin được chính xác hơn

1.3. Ưu và nhược điểm của Web API

- Ưu Điểm :

+ Web API là ứng dụng được sử dụng phổ biến trên các ứng dụng desktop, ứng dụng trang web hay ứng dụng mobile.

+ Web API đảm bảo linh hoạt với đa dạng các định dạng dữ liệu khi trả về client.

+ Có khả năng nhanh chóng xây dựng được HTTP service.

+ Mã nguồn mở có nhiệm vụ chính trong việc hỗ trợ chức năng RESTful đầy đủ, đảm bảo sử dụng bất kỳ client nào hỗ trợ JSON và XML.

+ Hỗ trợ đầy đủ đa dạng các thành phần MVC điển hình như: Routing, controller....

+ Web API có thể giao tiếp hai chiều đã được xác nhận trong đa dạng các giao dịch, đảm bảo độ tin cậy cao hơn bao giờ hết.

- Nhược Điểm :

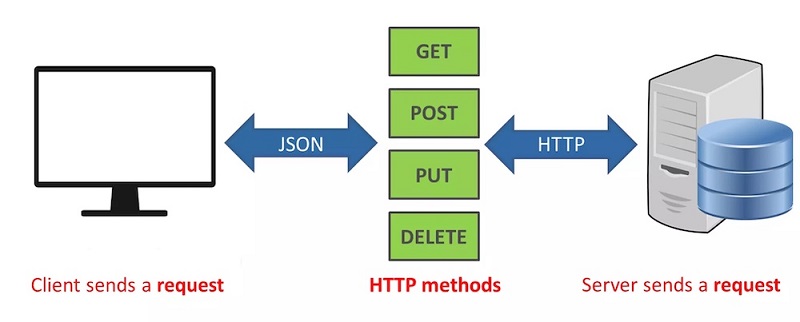
+ Web API vẫn chưa hoàn toàn phải là RESTful service mà mới chỉ được hỗ trợ mặc định GET, POST.

+ Để có thể sử dụng hiệu quả thì lúc này cần có thêm kiến thức chuyên sâu và có kinh nghiệm backend sao cho tốt nhất.

+ Web API tốn nhiều thời gian và chi phí đối với việc phát triển, nâng cấp cũng như vận hành.

+ Web API có thể gặp nhiều vấn đề về bảo mật khi hệ thống lúc này bị tấn công nếu như không giới hạn điều kiện kỹ lưỡng.

1.4. Cách thức hoạt động Web API



Hình 1. 1. Cách thức hoạt động của Web API

- Máy Khách Gửi Yêu Cầu: Máy khách (client) gửi yêu cầu đến địa chỉ của máy chủ (server) qua Internet.

- Máy Chủ Nhận Yêu Cầu: Máy chủ nhận được yêu cầu từ máy khách.

- Máy Chủ Xử Lý Yêu Cầu: Máy chủ đánh giá yêu cầu, kiểm tra xem người gửi có quyền làm những gì không, và sau đó thực hiện yêu cầu đó. Nó có thể lấy thông tin từ cơ sở dữ liệu hoặc thực hiện các thao tác khác.

- Máy Chủ Gửi Phản Hồi: Sau khi hoàn thành xử lý, máy chủ gửi phản hồ trở lại máy khách.

- Máy Khách Nhận Phản Hồi: Máy khách nhận phản hồi máy chủ

1.5. So sánh Web API với Web MVC

1.5.1. Điểm giống nhau:

- Cả hai đều dựa trên nền tảng ASP.NET Core, sử dụng các tính năng cơ bản của framework này như Dependency Injection, Middleware, và Hosting.

- Cả API và MVC đều sử dụng chung một quy trình xử lý HTTP request, bao gồm các middleware và routing.

Cả hai đều sử dụng hệ thống routing của ASP.NET Core để định tuyến các request đến các controller và action tương ứng.

- Cả hai đều sử dụng các controller để xử lý các request. Các controller trong ASP.NET Core đều kế thừa từ lớp ControllerBase.

- Các thuộc tính như [Route], [HttpGet], [HttpPost], v.v. có thể được sử dụng trong cả API và MVC để định nghĩa các endpoint và hành động của chúng.

1.5.2. Điểm khác nhau:

- Mục đích sử dụng :

+ ASP.NET Core API: Chủ yếu được sử dụng để xây dựng các dịch vụ Web API

+ ASP.NET Core MVC: Được sử dụng để xây dựng các ứng dụng web với giao diện người dùng như trang HTML....

- Lớp ControllerBase :

+ API: Các controller thường kế thừa từ ControllerBase và không sử dụng các tính năng liên quan đến giao diện người dùng như View.

+ MVC: Các controller thường kế thừa từ Controller (là lớp con của ControllerBase), hỗ trợ các tính năng render View, ViewBag, và ViewData.

- Kết quả trả về:

+ API: Thường trả về dữ liệu (JSON, XML, v.v.) bằng cách sử dụng các phương thức như Ok(), NotFound(), BadRequest(), v.v.

+ MVC: Thường trả về View (HTML) bằng cách sử dụng phương thức View()

2. Ngôn ngữ lập trình

2.1. C#

- Là một ngôn ngữ lập trình đa mục đích và mạnh mẽ, phát triển bởi Microsoft. Nó được sử dụng rộng rãi trong việc phát triển các ứng dụng desktop, web và di động trên nền tảng Windows. C# hỗ trợ lập trình hướng đối tượng và cung cấp một loạt các tính năng nâng cao như generics, LINQ và async/await.

1. Các công nghệ sử dụng
   1. HTTP và HTTPS

**-** HTTP (Hypertext Transfer Protocol):

+ HTTP là một giao thức truyền tải dữ liệu phổ biến nhất trên Internet.

+ Nó được sử dụng để truyền tải các trang web, hình ảnh, video, âm thanh, và các tài nguyên khác giữa máy chủ và trình duyệt của người dùng.

+ HTTP hoạt động dựa trên mô hình yêu cầu/đáp ứng, trong đó trình duyệt gửi yêu cầu đến máy chủ và máy chủ trả lời bằng dữ liệu mong muốn.

- HTTPS (HTTP Secure):

+ HTTPS là một phiên bản bảo mật của HTTP, sử dụng SSL/TLS (Secure Sockets Layer/Transport Layer Security) để mã hóa dữ liệu trước khi truyền qua mạng.

+ Mục tiêu của HTTPS là đảm bảo tính bí mật và toàn vẹn của dữ liệu trong quá trình truyền tải, ngăn chặn kẻ tấn công từ việc đánh cắp hoặc sửa đổi thông tin.

+ Để sử dụng HTTPS, máy chủ cần có một chứng chỉ SSL/TLS, được cấp phát bởi một tổ chức uy tín, để xác minh danh tính của nó.

* 1. REST và RESTful APIs

- **REST (Representational State Transfer):**

**+** REST là một phong cách kiến trúc sử dụng HTTP để tương tác với các dịch vụ web.

+ Nó dựa trên các nguyên tắc như phân tách giữa máy khách và máy chủ, trạng thái không lưu trữ, và khả năng mở rộng cao.

**- RESTful APIs:**

+ RESTful APIs tuân theo các nguyên tắc của REST, cho phép các ứng dụng giao tiếp với nhau qua HTTP.

+ Các phương thức HTTP như GET, POST, PUT, DELETE được sử dụng để thực hiện các thao tác trên tài nguyên.

* 1. JSON

- JSON (JavaScript Object Notation) là một định dạng dữ liệu phổ biến được sử dụng để truyền dữ liệu giữa máy chủ và trình duyệt web hoặc giữa các ứng dụng. Được xây dựng dựa trên cú pháp của JavaScript, JSON dễ đọc và ghi, và dễ dàng được sử dụng trong một loạt các ngôn ngữ lập trình.

* 1. OAuth2

- OAuth2 là một tiêu chuẩn ủy quyền cho phép các ứng dụng truy cập vào tài nguyên của người dùng trên một dịch vụ mà không cần tiết lộ thông tin đăng nhập của người dùng.

- Nó sử dụng các token truy cập để cấp quyền truy cập tạm thời và giới hạn vào tài nguyên.

* 1. JWT

- JWT là một tiêu chuẩn mở được sử dụng để chia sẻ thông tin an toàn giữa các bên dưới dạng một đối tượng JSON.

- Thông tin trong JWT được ký số để đảm bảo tính toàn vẹn và xác thực của dữ liệu.

* 1. Authentication và Authorization

- Authentication (Xác Thực):

+ Authentication là quá trình xác định xem một người dùng có quyền truy cập vào hệ thống hay không bằng cách xác minh danh tính của họ.

+ Quá trình này thường bao gồm yêu cầu người dùng cung cấp thông tin đăng nhập, như tên người dùng và mật khẩu, sau đó kiểm tra thông tin này với dữ liệu xác thực trong hệ thống.

+ Authentication có thể được thực hiện bằng nhiều cách khác nhau, bao gồm thông qua biến cookie, token, hoặc sử dụng các dịch vụ xác thực bên ngoài như OAuth hoặc OpenID Connect.

- Authorization (Ủy Quyền):

+ Authorization xác định những gì một người dùng được phép làm sau khi họ đã được xác thực thành công trong hệ thống.

+ Quyền truy cập được quản lý thông qua việc xác định các vai trò hoặc quyền hạn cho từng người dùng hoặc nhóm người dùng.

+ Trong hệ thống, quyền truy cập được ánh xạ thành các hành động cụ thể mà người dùng có thể thực hiện, ví dụ như đọc, ghi, cập nhật hoặc xóa dữ liệu.

+ Authorization thường được thực hiện thông qua việc kiểm tra vai trò hoặc quyền hạn của người dùng khi họ cố gắng thực hiện một hành động cụ thể trong hệ thống.

* 1. Entity Framework

- Entity Framework (EF) là một ORM (Object-Relational Mapping) cho .NET, giúp các nhà phát triển làm việc với cơ sở dữ liệu bằng cách sử dụng các đối tượng .NET.

- EF giúp giảm bớt công việc lặp lại và lỗi trong quá trình làm việc với dữ liệu, cung cấp một cách tiếp cận linh hoạt và hiệu quả để tương tác với cơ sở dữ liệu.

* 1. HttpClient (ASP.NET Core)

- HttpClient trong ASP.NET Core là một lớp dùng để gửi yêu cầu HTTP và nhận phản hồi từ các tài nguyên web. Nó dễ dàng tương tác với các API và dịch vụ web.

- Các tính năng chính:

+ Gửi và nhận yêu cầu HTTP (GET, POST, PUT, DELETE,...).

+ Hỗ trợ thao tác không đồng bộ (async/await), giúp ứng dụng không bị chặn.

+ Tái sử dụng kết nối, giúp tối ưu hóa hiệu suất.

**Chương 2. Xây dựng ứng dụng Web (Triển khai thực tế)**

1. Giới thiệu đề tài

Hiện nay, API đã trở thành một phần không thể thiếu trong việc phát triển các ứng dụng và dịch vụ trực tuyến. Với sự phổ biến của Internet và các nền tảng kỹ thuật số, API đóng vai trò quan trọng trong việc kết nối các ứng dụng và dịch vụ với nhau, tạo ra một môi trường hợp nhất và linh hoạt.

Đề tài "Hệ thống Film (Film Management System)" của nhóm em nhằm xây dựng một hệ thống giúp người dùng quản lý nội dung film một cách hiệu quả. Hệ thống sẽ có các chức năng cơ bản như tải ảnh lên hệ thống, tải ảnh về, và phân loại hình ảnh...... Mục tiêu của đề tài là tạo ra một công cụ đơn giản và dễ sử dụng, giúp mọi người tổ chức và quản lý bộ sưu tập hình ảnh của mình một cách thuận tiện và hiệu quả hơn.

1. Yêu cầu chức năng
   1. Chức năng của người quản trị (Admin)

- Quyền của Admin :

+ Tài khoản Admin sẽ có quyền sử dụng nhiều dịch vụ của hệ thống hơn quyền người dùng (User)

+ Có tất cả quyền sử dụng dịch vụ dịch vụ hệ thống mà người dùng có như quản lý tài khoản, quản lý album và ảnh ...

* 1. Chức năng của người dùng (User)

- Quản lý tài khoản :

+ Đăng ký, đăng nhập, và quản lý thông tin cá nhân của mình vào hệ thống .

- Quản lý nội dung film:

+ Xem danh sách các film , actors ,directors .

+ Thêm và xóa film ,actors , directors

+ Phân loại album và ảnh cá nhân.

1. Phân tích và thiết kế hệ thống
   1. Phân tích

- Quản lý tài khoản :

+ Đăng ký: Người dùng cung cấp thông tin cá nhân như tên, email, và mật khẩu để đăng ký tài khoản mới. Hệ thống cần kiểm tra tính hợp lệ của thông tin và xác nhận rằng không có tài khoản nào cùng thông tin đã tồn tại trước đó.

+ Đăng nhập: Người dùng cung cấp thông tin đăng nhập (email và mật khẩu) để truy cập vào hệ thống. Hệ thống cần xác thực thông tin và cấp cho người dùng các quyền làm việc với hệ thống .

+ Phân quyền: Dựa trên quá trình xác thực, hệ thống xác định vai trò của người dùng (Admin, User) và ủy quyền cho họ truy cập vào các tính năng cụ thể .

- Quản lý film:

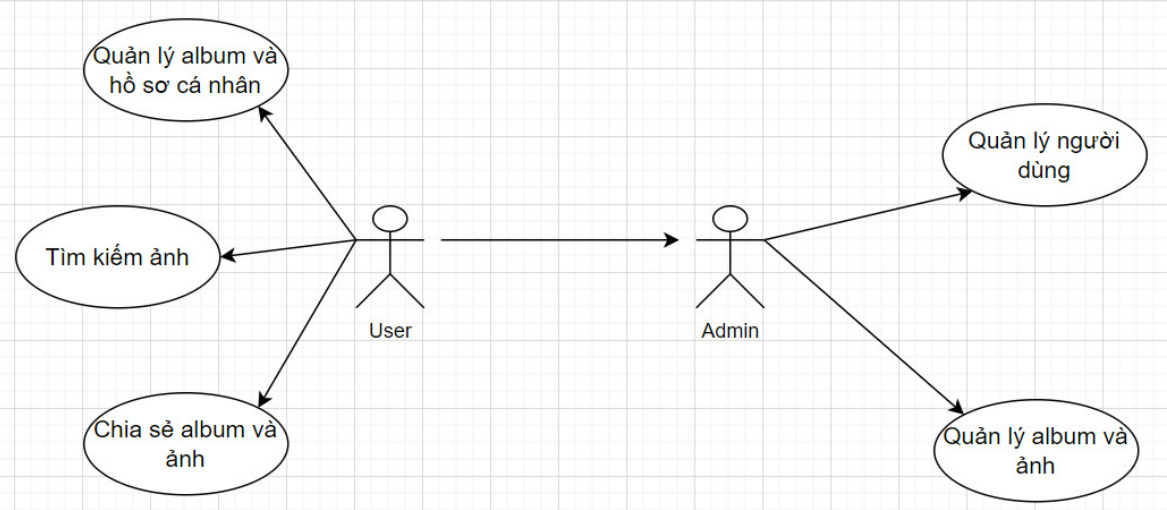
+ Tạo film mới: tạo mới nội dung film gồm tên , ngày ra mắt , actor và directors.

+ Sửa đổi film: Người dùng có thể chỉnh sửa thông tin của film như tên và mô tả sau khi đã tạo.

+ Xóa film: Người dùng có thể xóa film không cần thiết. Hệ thống cần xác nhận trước khi xóa và cần xác định các hành động đi kèm

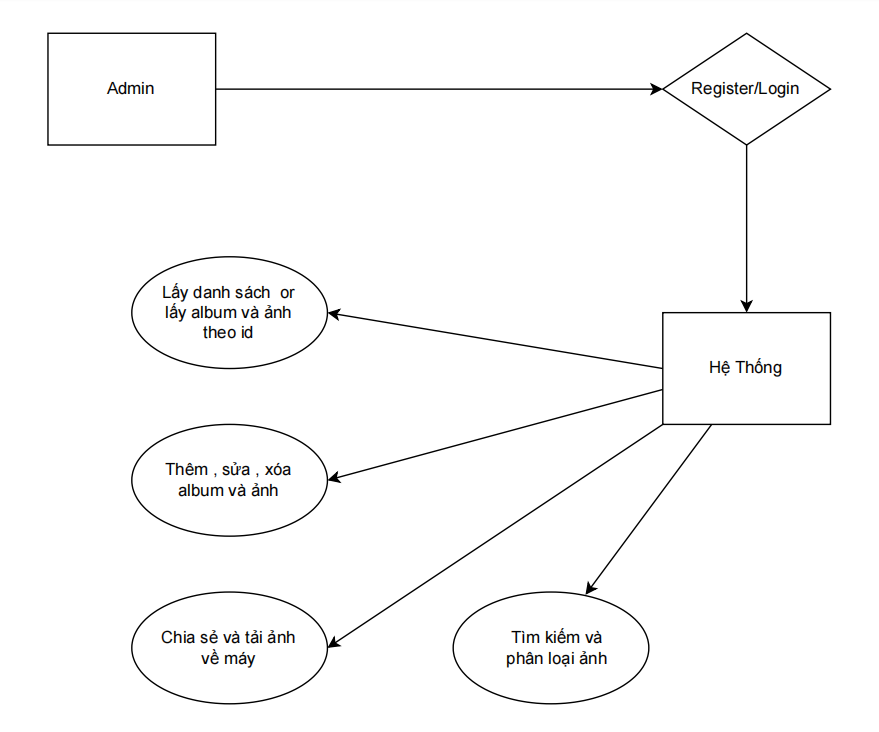
* 1. Thiết kế hệ thống

3.2.1. Biểu đồ user case tổng quát



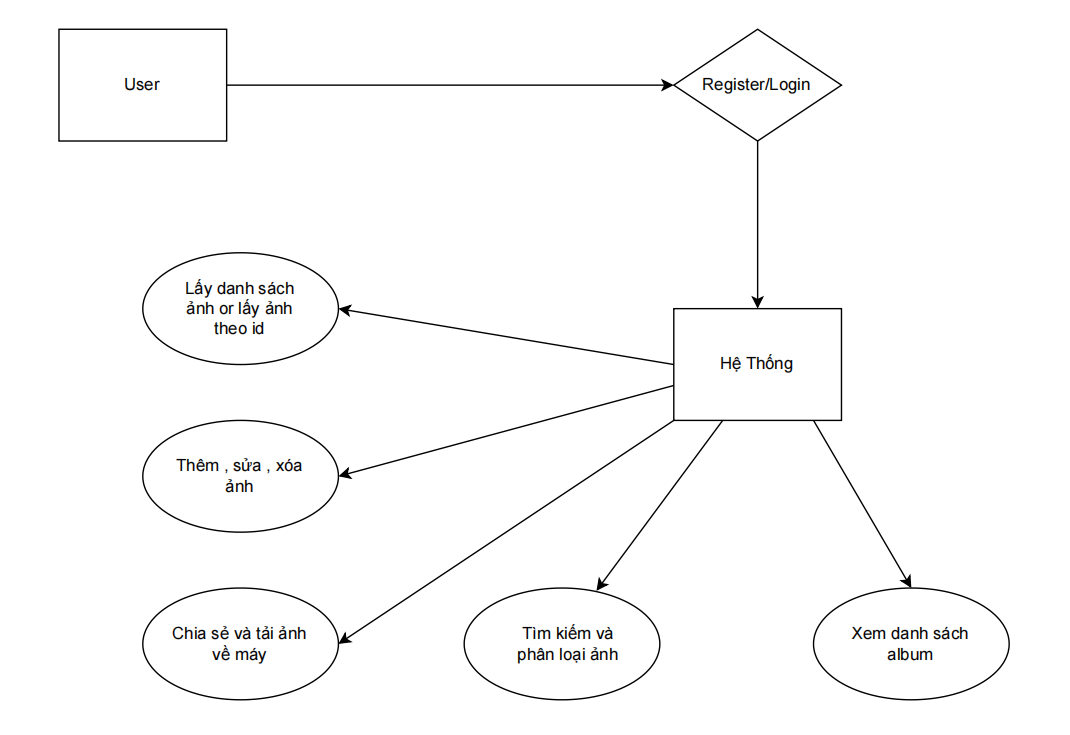
Hình 1. 2. User case hệ thống

3.2.2. Biểu đồ user case cho admin



Hình 1. 3. User case cho Admin

3.2.3. Biểu đồ user case cho user



Hình 1. 4. User case cho User

1. API Endpoints

- Quản lý tài khoản :

+ POST : api/user/register : Đăng ký người dùng

+ POST : api/user/login : Đăng nhập vào hệ thống

- Quản lý film :

+ GET : api/Films/get-all-films: Lấy danh sách tất cả film

+ GET : api/ Films /get-films-by-id: Lấy film theo id

+ POST : api/ Films /addfilm-image : them film mới

+ PUT : api/ Films /update- Films: Sửa nội dung của film

+ DELETE : api/ Films /delete- Films: Xóa films

- Quản lý actor :

- GET : api/ actor /get-all- actor: Lấy danh sách tất cả actor

- GET : api/ actor /get- actor -by-id: Lấy actor theo id

- POST : api/ actor /add- actor: Thêm actor

- PUT : api/ actor /update- actor: Sửa thông tin

- DELETE : api/ actor /delete- actor: Xóa actor

- Quản lý director :

- GET : api/ director /get-all- director : Lấy danh sách tất cả director

- GET : api/ director /get- director -by-id: Lấy director theo id

- POST : api/ director /add- director : Thêm director

- PUT : api/ director /update- director : Sửa thông tin

- DELETE : api/ director /delete- director : Xóa director

1. Cơ sở dữ liệu
   1. Bảng cơ sở dữ liệu

- Album

public class Films

{

[Key]

public int Id { get; set; }

public string? Name { get; set; }

public string? Description { get; set; }

public DateTime Date { get; set; }

public string? Genre { get; set; }

public decimal? Rate { get; set; }

public List<FilmActors>? FilmActors { get; set; }

public List<FilmDirectors>? FilmDirectors { get; set; }

}

public class Actors

{

[Key]

public int Id { get; set; }

public string? FirstName { get; set; }

public string? LastName { get; set; }

public DateTime DateOfBirth { get; set; }

public List<FilmActors>? FilmActors { get; set; }

}

public class Directors

{

[Key]

public int Id { get; set; }

public string? FirstName { get; set; }

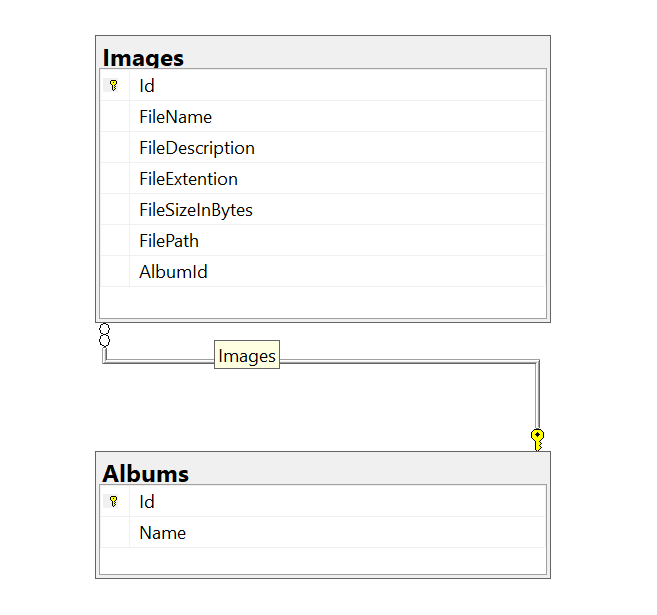
public string? LastName { get; set; }

public DateTime DateOfBirth { get; set; }

public List<FilmDirectors>? FilmDirectors { get; set; }

}

* 1. Mối quan hệ giữa các bảng



Hình 1. 5. Sơ đồ quan hệ

**Chương 3. Giao diện Consume API**

1. Giao diện trang chủ

A white paper with black text

Description automatically generated

Hình 1. 6. Giao diện trang chủ

1. Giao diện quản lý tài khoản
   1. Giao diện đăng nhập

A white paper with black text

Description automatically generated

Hình 1. 8. Giao diện đăng nhập tài khoản

1. Giao diện quản lý film

A screenshot of a computer

Description automatically generated

Hình 1. 9. Giao diện quản lý film

1. Giao diện quản actor

A screenshot of a computer

Description automatically generated

Hình 1. 10. Giao diện quản lý actor

1. Giao diện quản director

A close-up of a white background

Description automatically generated

Hình 1. 10. Giao diện quản lý director

**Chương 4. Kết quả chương trình**

1. Kết luận

Trong quá trình hoàn thiện đề tài, các thành viên trong nhóm đã tìm hiểu về “Hệ thống quản lý hình ảnh” nhưng không thể tránh khỏi những thiếu sót ngoài mong muốn, mong thầy đóng góp thêm ý kiến để nhóm em có thêm kinh nghiệm để hoàn thiện đề tài này hơn. Quá trình phân tích Hệ thống quản lý hình cơ bản đã hoàn thành những công việc sau:- Kiến thức cơ bản- Thiết kế biểu đồ use case tổng quát, biểu đồ cho user và biểu đồ cho admin chi tiết cho từng tác nhân- Thiết kế CSDL- Thiết kế và xây dựng phần mềm- Nắm bắt và thực hiện được quy trình phân tích. Do thời gian hạn chế và kinh nghiệm của từng thành viên chưa nhiều nên việc thực hiện đề tài chưa hoàn thiện như một phần mềm quản lý thực thụ.Hệ thống quản lý hình ảnh này mang tính chất học hỏi kinh nghiệm và dần làm quen với thực tế. Tóm lại, qua báo cáo lần này, các thành viên trong nhóm nói riêng và cả nhóm em nói chung đã học hỏi được rất nhiều kiến thức cũng như kinh nghiệm trong quá trình hoàn thiện đề tài, nắm được vững hơn kiến thức môn học góp phần nâng cao trình độ chuyên môn của mỗi cá nhân.Cuối cùng, nhóm em xin gửi lời cảm ơn đến thầy Nguyễn Đức Tấn đã hỗ trợ nhiệt tình giúp cho bài báo cáo này hoàn thiện hơn!

1. Tài liệu tham khảo

- "Pro ASP.NET MVC Framework" của Steven Sanderson

- RESTful Web APIs" của Leonard Richardson và Mike Amundsen