**ĐẠI HỌC QUỐC GIA TP. HỒ CHÍ MINH  
TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN**



**BÁO CÁO ĐỒ ÁN**

**MÔN HỌC**

**CÔNG NGHỆ .NET**

**Đề tài: ASP.NET Core**

**❖ Giảng viên hướng dẫn ❖**

ThS. Huỳnh Hồ Thị Mộng Trinh

**❖ Lớp ❖**

SE310.K21.PMCL

**❖ Sinh viên thực hiện ❖**

Nguyễn Việt Tiến – 16521233

**Tp. Hồ Chí Minh, tháng 03 năm 2020**

# **NHẬN XÉT CỦA GIẢNG VIÊN HƯỚNG DẪN**

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

*......................., ngày ............. tháng ………. năm 2020*

**Người nhận xét**

**(Ký tên và ghi rõ họ tên)**

# **LỜI CẢM ƠN**

Trong quá trình làm bài báo cáo, em đã nhận được sự giúp đỡ, đóng góp ý kiến và chỉ bảo nhiệt tình của thầy cô, gia đình và bạn bè.

Em xin gửi lời cảm ơn chân thành đến ThS. Huỳnh Hồ Thị Mộng Trinh - giảng viên khoa Công nghệ phần mềm - Trường ĐH CNTT- ĐHQG Tp HCM người đã tận tình hướng dẫn, góp ý và chỉ bảo em trong suốt quá trình làm báo cáo.

Ngoài ra, em cũng xin gửi lời cảm ơn đến những thầy cô giáo trong trường ĐH CNTT- ĐHQG Tp HCM nói chung, các thầy cô trong Khoa Công nghệ phần mềm nói riêng đã dạy dỗ cho em kiến thức về lập trình, qua đó em có được cơ sở lý thuyết vững vàng và từ đó hoàn thành tốt đồ án.

Cuối cùng, em xin chân thành cảm ơn gia đình và bạn bè, đã luôn tạo điều kiện, quan tâm, giúp đỡ, động viên em trong suốt quá trình hoàn thành đồ án.

*Thành phố Hồ Chí Minh, ngày 16 tháng 03 năm 2020*

# **MỤC LỤC**

[**NHẬN XÉT CỦA GIẢNG VIÊN HƯỚNG DẪN** 2](#_Toc35173282)

[**LỜI CẢM ƠN** 3](#_Toc35173283)

[**MỤC LỤC** 4](#_Toc35173284)

[**I: Giới thiệu về ASP.NET Core** 5](#_Toc35173285)

[1.1 ASP.NET Core là gì? 5](#_Toc35173286)

[1.2 Sự hình thành và phát triển của ASP.NET Core 6](#_Toc35173287)

[1.3 ASP.NET Core và ASP.NET 8](#_Toc35173288)

[**II: Cài đặt và cấu hình môi trường phát triển ASP.NET Core** 14](#_Toc35173289)

[**III: Xây dựng RESTful APIs với ASP.NET Core** 15](#_Toc35173290)

[RESTful API là gì? 17](#_Toc35173291)

[Tạo mới Web API project: 18](#_Toc35173292)

[Thêm một Model 22](#_Toc35173293)

[Thêm Controller và Scaffolding 24](#_Toc35173294)

[Tạo Database với Migrations 26](#_Toc35173295)

[CRUD Method 28](#_Toc35173296)

[Test API sử dụng Postman 28](#_Toc35173297)

# **I:** **Giới thiệu về ASP.NET Core**

## 1.1 ASP.NET Core là gì?

[ASP.NET Core](https://docs.microsoft.com/en-us/aspnet/core/?view=aspnetcore-3.1) là một open-source mới và framework đa nền tảng (cross-platform) cho việc xây dựng những ứng dụng hiện tại dựa trên kết nối đám mây, giống như web apps, IoT và backend cho mobile. Với ASP.NET Core, bạn có thể:

* Xây dựng những ứng dụng web và servies, ứng dụng IoT và backends cho mobiles.
* Sử dụng các công cụ phát triển yêu thích của bạn trên Windows, MacOS và Linux.
* Deploy hệ thống vào đám mây hoặc tại chỗ.
* Chạy trên .[NET Core hoặc .NET Framework](https://docs.microsoft.com/en-us/dotnet/standard/choosing-core-framework-server#when-to-choose-net-framework).

[ASP.NET Core](https://docs.microsoft.com/en-us/aspnet/core/?view=aspnetcore-3.1) có một số thay đổi kiến trúc lớn, đó là kết quả của việc học hỏi rất nhiều từ các framework module hóa khác. ASP.NET Core không còn dựa trên System.Web.dll nữa. Nó được dựa trên một tập hợp các gói, các module hay cũng được gọi là các Nuget packages. Điều này cho phép bạn tối ưu ứng dụng của bạn để chỉ bao gồm những packages nào cần thiết. Lợi ích của nó là giúp cho ứng dụng nhỏ hơn, bảo mật chặt chẽ hơn, giảm sự phức tạp, tối ưu hiệu suất hoạt động và giảm chi phí, thời gian cho việc phát triển.

**Vì sao chúng ta nên sử dụng ASP.NET Core ?**

ASP.NET Core đạt được các lợi ích như sau:

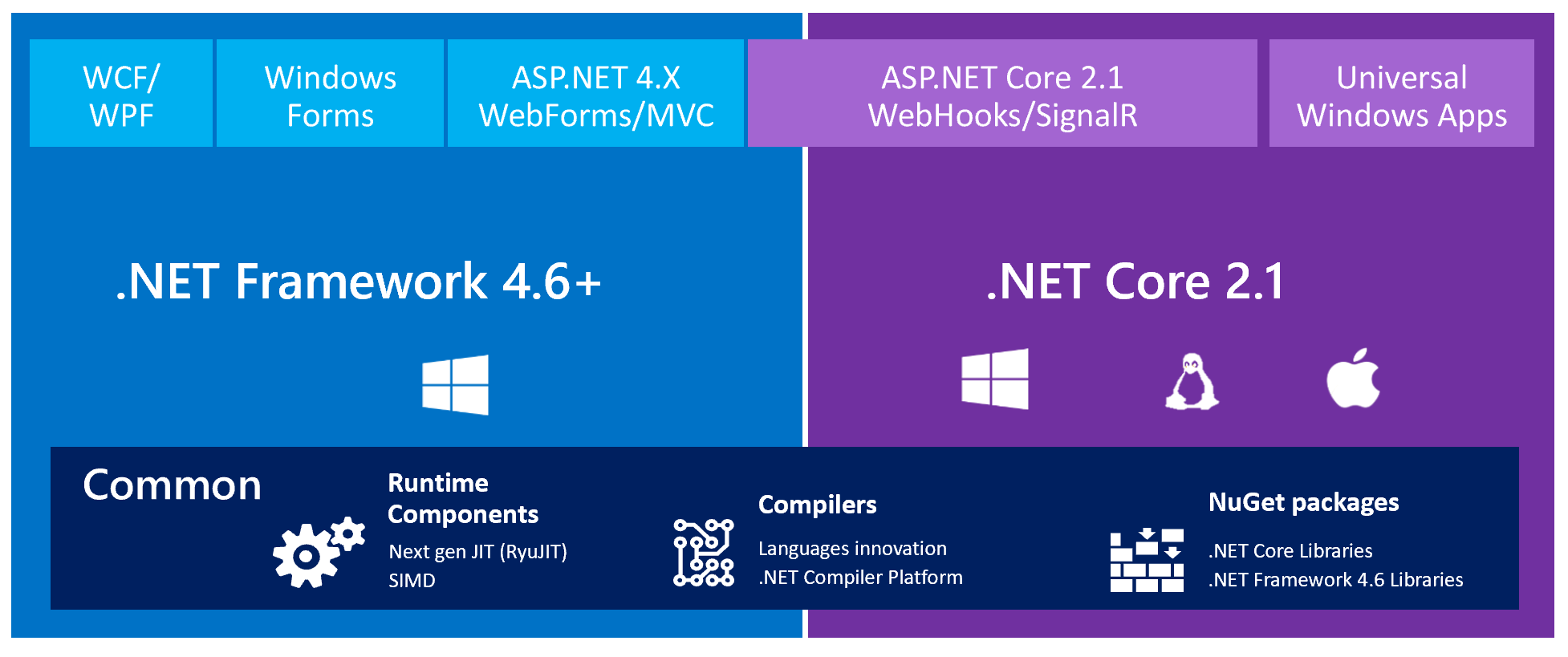
* Hợp nhất việc xây dựng web UI và web APIs.
* Tích hợp những client-side frameworks hiện đại và những luồng phát triển.
* Khả năng phát triển và chạy trên Windows, macOS và Linux.
* Hệ thống được cấu hình dựa trên môi trường đám mây.
* Dependency injection được xây dựng sẵn.
* HTTP request được tối ưu nhẹ hơn.
* Có thể host trên IIS hoặc self-host trong process riêng của bạn.
* Được xây dựng trên .NET Core, hỗ trợ app versioning.
* Những công cụ mới để đơn giản hóa quá trình phát triển ứng dụng web hiện đại.
* Mã nguồn mở và tập trung vào cộng đồng.

## 1.2 Sự hình thành và phát triển của ASP.NET Core

Vào cuối năm 1996, Microsoft đã phát hành một framework với tên gọi là ASP ( Active Server Pages) hoặc ASP Classic. ASP.Net là kỹ thuật lập trình và phát triển ứng dụng web ở phía Server (Server-side) dựa trên nền tảng của Microsoft để tạo ra web pages 1 cách hoàn toàn chủ động.

Thay đổi lớn nhất đến từ việc Microsoft đã phát hành và giới thiệu .NET Framework vào năm 2002. Việc xây dựng một trang web giờ đây tương tự như việc xây dựng Windowsform. Với Web Forms, Microsoft đã cố gắng mang lại những mô hình Visual Basic vào Web. Tuy nhiên, Web Form không có kiến trúc Project cố định được xác định trước để tạo ra các ứng dụng Web cho khách hàng. Lập trình viên cần phải vận dụng một cách linh hoạt, để tạo nên các ứng dụng, điều này đòi hỏi một khả năng sáng tạo cao hơn bình thường. Bên cạnh đó, Web Forms không thể tự động thực hiện kiểm tra Unit Tetsing, vì khi kết thúc, code behind Web Forms xử lí rất nhiều sự kiện. Ngoài ra Web Forms ít có khả năng hỗ trợ cho sự phát triển song song giữa ASPX và các tập tin code behind.

Vào năm cuối năm 2009, Microsoft đã phát hành phiên bản ASP.NET MVC đầu tiên. Có thể nói sự ra đời của mô hình MVC đã làm thay đổi toàn bộ về lập trình, người ta bắt đầu cảm thấy những ưu điểm vượt trội hơn mà mô MVC mang lại so với Web Form vốn đã quá cũ và không đáp ứng được nhu cầu ngày càng cao của lập trình. MVC là mô hình code bao gồm 3 phần Model, View và Controlerl hoạt động với nhau, tạo thành một mô hình khép kín giúp website xử lý thông tin một cách nhanh chóng và mượt mà, tự nhiên hơn hẳn so với Web Form sử dụng ViewState.



Vào tháng 6, năm 2016, Microsoft giới thiệu ASP.NET Core phiên bản đầu tiên. ASP.NET Core không phải là phiên bản tiếp theo của ASP.NET 4.6. Nó thực ra được viết lại trên framework ASP.NET 4.6 hiện tại những kích thước nhỏ hơn và nhiều modular hơn, đến năm 2017 phiên bản thứ 2 được phát hành và ở thời điểm hiện tại (2020) Microsoft đã phát hành ASP.NET Core 3.1.

A picture containing indoor

Description automatically generated

## 1.3 ASP.NET Core và ASP.NET

.NET Framework được Microsoft đưa ra chính thức từ năm 2002. .NET Framework chỉ hoạt động trên Windows. Những nền tảng ứng dụng như WPF, Winforms, ASP.NET(1-4) hoạt động dựa trên .NET Framework.

  Cho đến năm 2013, Microsoft định hướng đi đa nền tảng và phát triển .NET Core. .NET core hiện được sử dụng trong các ứng dụng Universal Windows platform và ASP.NET Core.

A screenshot of a cell phone

Description automatically generated

ASP.NET Core không phải là phiên bản tiếp theo của ASP.NET. Nó là một cái tên mới được xây dựng từ đầu. Nó có một sự thay đổi lớn về kiến trúc và kết quả là nó gọn hơn, phân chia module tốt hơn. ASP.NET Core có thể chạy trên cả .NET Core hoặc full .NET Framework.

A screenshot of a social media post

Description automatically generated

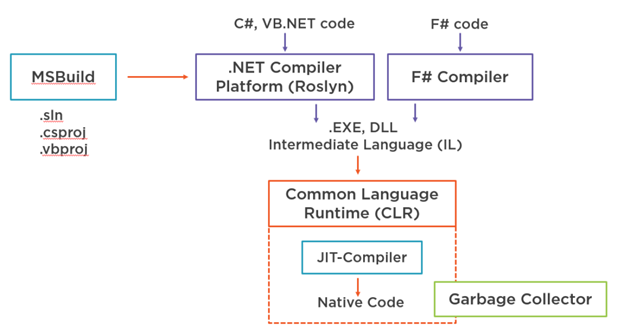
.NET Framework là một nền tảng (Platform) + các framework + hệ thống thư viện (Library) do Microsoft phát triển từ 2001. Đừng để từ “Framework” trong tên gọi gây nhầm lẫn. .NET Framework thực sự bao trùm nhiều định nghĩa.

Thứ nhất, .NET Framework là một platform để thực thi ứng dụng.

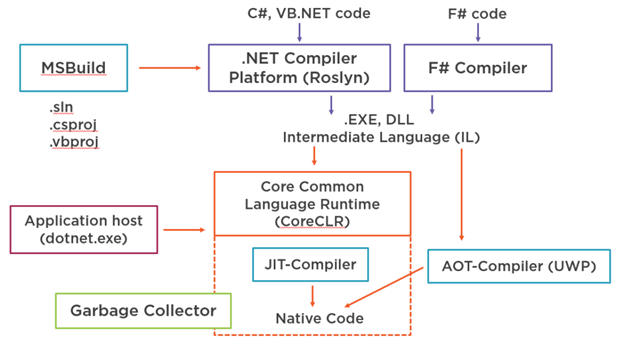
Chương trình bạn viết bằng bất kỳ ngôn ngữ lập trình .NET nào sẽ dịch sang mã trung gian (**Intermediate Language**) dù chương trình ở dạng thư viện **dll** hoặc tự thực thi **exe**.

Khi chạy, mã IL sẽ được một trình biên dịch khác có tên gọi là **Just-in-Time** (JIT) compiler chuyển tiếp thành một dạng mã máy (**machine instruction**) và thực thi trong một môi trường khép kín do .NET quản lý. Mỗi trường thực thi này có tên gọi là Common Language Runtime. Cả quy trình code – dịch sang mã IL – dịch sang mã máy – thực thi như vậy được gọi là **toolchain** của .NET Framework.

Thứ hai, .NET Framework tạo ra nhiều loại framework khác nhau giúp bạn nhanh chóng bổ sung những tính năng riêng để tạo thành chương trình hoàn chỉnh. .NET Framework cung cấp hệ thống thư viện class khổng lồ chung mà tất cả các framework trên nó đều có thể sử dụng. Ngoài ra, với từng framework lại có hệ thống thư viện class hỗ trợ riêng. Đây chính là vai trò “thư viện” của .NET Framework. Trên thực tệ, hệ thống thư viện của .NET Framework còn ảnh hưởng đến tất cả các ngôn ngữ hỗ trợ. Ví dụ: ngôn ngữ lập trình C# chỉ có hệ thống cú pháp chứu không có hệ thống kiểu dữ liệu và thư viện riêng của mình. Tất cả những thứ này trong C# đều đến từ .NET Framework. Dưới đây là hình ảnh minh họa toolchain của .NET FrameWork:



Khi bạn đã hiểu .NET Framework theo khía cạnh như trên, bạn sẽ cần hình dung tương tự về .NET Core..NET Core cũng là một hệ thống trọn vẹn bao gồm nền tảng dịch mã trung gian và thực thi ứng dụng, các framework để phát triển các loại ứng dụng, và là hệ thống thư viện hỗ trợ. Hệ thống của .NET Core được xây dựng mới hoàn toàn và độc lập với .NET Framework. Dưới đây là minh họa toolchaine của .NET Core.



Về mặt ý tưởng nó có những điểm tương tự như trong .NET Framework. Tuy nhiên nó không liên quan đến .NET Framework. Thứ nhất, môi trường thực thi của .NET Core được gọi là **CoreCLR**. CoreCLR có thể chạy trên nhiều hệ điều hành. Hiện nay CoreCLR có thể hoạt động trên Windows, Linux và macOS.

CoreCLR sử dụng một trình biên dịch trung gian tương tự JIT compiler của .NET Framework. Mặc dù tên gọi giống nhau nhưng JIT compilter của .NET Core không phải là JIT của .NET Framework. JIT của .NET Core có thể dịch mã IL sang mã máy của 3 nền tảng nó hỗ trợ. Một sự khác biệt nữa về runtime nằm ở chỗ, CoreCLR và mã máy được tải và kích hoạt bởi một tiến trình khác, dotnet.exe, trong khi CLR được kiểm soát bởi hệ điều hành Windows.

Với vai trò framework, .NET Core cung cấp khung sườn cho phát triển ứng dụng web (ASP.NET Core), phát triển ứng dựng desktop trên windows (Windows Forms và WPF, từ .NET Core 3.1). Về tính năng này, .NET Core và .NET Framework gần như tương đương nhau. Với vai trò thư viện, .NET Core cũng cung cấp hệ thống class cho các ngôn ngữ nó hỗ trợ (hiện nay có C#, VB.NET và F#).

Tuy nhiên ở đây cần nhấn mạnh rằng: Thứ nhất, hệ thống thư viện của .NET Core và .NET Framework là hoàn toàn độc lập nhau. Tuy nhiên chúng đều là các file chứa mã trung gian IL cho nên về lý thuyết chúng có thể sử dụng thư viện của nhau. Trên thực tế, bắt đầu từ .NET 2.0 bạn có thể tham chiếu tới các thư viện viết trên .NET Framework. Điều này giúp việc chuyển đổi sang .NET Core dễ dàng hơn. Tuy nhiên, việc tham chiếu này cũng có giới hạn. Thư viện xây dựng trên .NET Framework sẽ không chạy được trên .NET Core nếu nó phụ thuộc vào những API không được .NET Core hỗ trợ.

Thứ hai, đội ngũ phát triển .NET Core sử dụng lại nguyên vẹn của các thư viện cơ bản của .NET Framework. Điều này giúp lập trình viên dễ dàng chuyển đổi từ .NET Framework sang .NET Core. Lấy ví dụ, cả trong .NET Framework và .NET Core đều có class Console (System.Console), trong đó đều có các phương thức như Write, WriteLine, Read, ReadLine. Do vậy, nếu bạn đã thành thạo C# (trên .NET Framework), bạn tiếp tục sử dụng nó trên .NET Core mà không cần học thêm gì về ngôn ngữ lập trình nữa.

**Nên sử dụng .NET Framework hay .NET Core ?**

Điều đó tùy thuộc vào ứng dụng mà bạn có ý định phát triển. Đối với các ứng dụng Windows desktop, .NET Framework sẽ là sự lựa chọn của bạn. .NET Core chạy được đa nền tảng và có hiệu năng cao hơn. Nhược điểm duy nhất của nó là số lượng thư viện hỗ trợ vẫn còn hạn chế. .NET Framework có hệ sinh thái lớn hơn với nhiều các thư viện hỗ trợ hơn.

Bảng dưới đây sẽ mô tả chi tiết hơn về sự khác nhau giữa ASP.NET Core và ASP.NET 4x.

|  |  |
| --- | --- |
| **ASP.NET Core** | **ASP.NET 4x** |
| Cross-platform (Windows, MacOS, Linix) | Hỗ trợ Windows |
| Chạy được trên [.NET Core](https://docs.microsoft.com/en-us/dotnet/core/about) và [.NET Framework](https://dotnet.microsoft.com/learn/dotnet/what-is-dotnet-framework) | Chạy trên .NET Framework |
| Một máy có thể có nhiều phiên bản | Mỗi máy chỉ có một phiên bản |
| Các công cụ phát triển bao gồm: Visual Studio, Visual Studio for Mac, Visual Studio Code với ngôn ngữ lập trình là C# hoặc F# | Công cụ phát triển là Visual Studio với ngôn ngữ lập trình là C#, VB hoặc F# |
| Nhỏ, nhẹ được moudle hóa | System.web.dll cồng kềnh |
| Không hỗ trợ WebForms | WebForms được hỗ trợ |
| Dự án mã nguồn mở ([GitHub](https://github.com/dotnet/aspnetcore)) | Bản quyền thuộc về Microsoft |

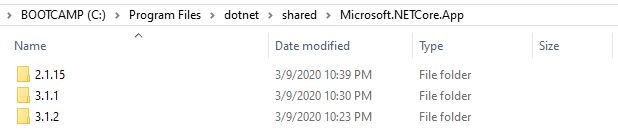
# **II: Cài đặt và cấu hình môi trường phát triển ASP.NET Core**

Tiếp theo chúng ta sẽ xem qua việc cài đặt và cấu hình môi trường để phát triển ứng dụng với ASP.NET Core

* IDE: Bạn có thể lựa chọn 1 trong 3 IDE của Microsoft bao gồm: Visual Studio, Visual Studio Code và Visual Studio for Mac. [Visual Studio 16.4](https://visualstudio.microsoft.com/downloads/) sẽ là phiên bản mà mình sẽ sử dụng trong bài viết này, cùng với đó là **ASP.NET and web development** workload.
* [.NET Core SDK 3.1](https://dotnet.microsoft.com/download/dotnet-core/3.1)

Lưu ý: Bạn cũng có thể kiểm tra xem phiên bản .NET Core đã được cài đặt trên máy của mình thông qua câu lệnh command line: dotnet --version

Hoặc để có thể thấy tất cả các phiên bản .NET Core đã được cài đặt trên PC, các bạn có thể kiểm tra thư mục của .NET SDK với đường dẫn như sau: [C:\Program Files\dotnet\shared\Microsoft.NETCore.App](file:///C:\Program%20Files\dotnet\shared\Microsoft.NETCore.App)



# **III: Xây dựng RESTful APIs với ASP.NET Core**

Các mục của phần demo sẽ bao gồm :

* REST API là gì ?
* Cách tạo một Web API Project
* Thêm 1 Model
* Thêm Controllder và Scaffolding
* Tạo Database với Migrations
* Tesing CRUD Method

Trong phần này, mình sẽ hướng dẫn các bạn cách xây dựng RESTful Web API bằng ASP.NET Core và Entity Framework. API sẽ quản lý những giá trị của các bộ phim được lưu trữ trong cơ sở dữ liệu quan hệ được mô tả trong bảng dưới đây:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Method** | **URL** | **Operation** | **Mô tả** | **Request body** | **Respond body** |
| **GET** | /api/movies | Read | Lấy giá trị của tất cả movies | None | Giá trị của tất cả các movies |
| **GET** | /api/movies{id} | Read | Lấy giá trị của movie theo ID | None | Giá trị của movie |
| **POST** | /api/movies | Create | Tạo mới một movie | Giá trị của movie | Giá trị của movie |
| **PUT** | /api/movies{id} | Update | Cập nhật giá trị của movie theo ID | Giá trị của movie | None |
| **DELETE** | /api/movies{id} | Delete | Xóa giá trị của movie theo ID | None | Giá trị của movie |

Cấu trúc của ASP.NET Core Web API

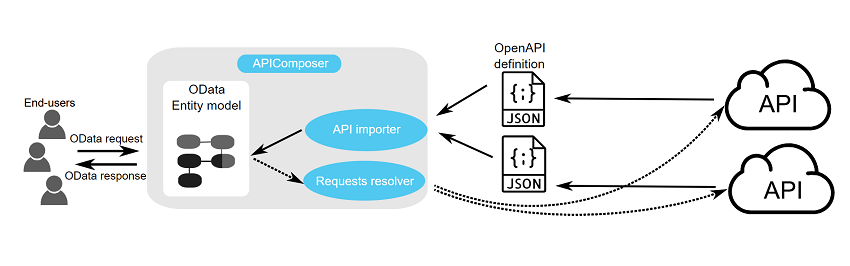


Và tất nhiên, để bắt đầu các bạn cần phải chuẩn bị những công cụ để phát triển và kiểm thử:

* [Visual Studio 2019](https://visualstudio.microsoft.com/downloads/)
* [.NET Core SDK 3.1](https://dotnet.microsoft.com/download/dotnet-corec)
* [Microsoft SQL Server Express](https://www.microsoft.com/en-us/sql-server/sql-server-2019)
* [Postman](https://www.postman.com/downloads/)

Nếu bạn đã sẵn sàng, chúng ta bắt đầu thôi !

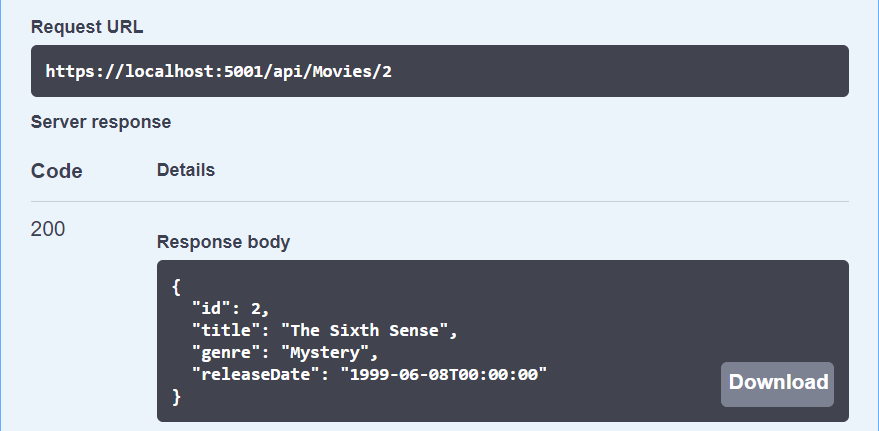
## RESTful API là gì?

Để hiểu rõ hơn về **RESTful API** ta sẽ đi lần lượt giải thích các khái niệm nhở **API**, **REST** hay **RESTful**.

**REST** (**RE**presentational **S**tate **T**ransfer) là một dạng chuyển đổi cấu trúc dữ liệu, một kiểu kiến trúc để viết API. Nó sử dụng phương thức HTTP đơn giản để tạo cho giao tiếp giữa các máy. Vì vậy, thay vì sử dụng một URL cho việc xử lý một số thông tin người dùng, REST gửi một yêu cầu HTTP như GET, POST, DELETE, vv đến một URL để xử lý dữ liệu.

**API** (**A**pplication **P**rogramming **I**nterface) là một tập các quy tắc và cơ chế mà theo đó, một ứng dụng hay một thành phần sẽ tương tác với một ứng dụng hay thành phần khác. API có thể trả về dữ liệu mà bạn cần cho ứng dụng của mình ở những kiểu dữ liệu phổ biến như JSON hay XML.

**RESTful API là một tiêu chuẩn dùng trong việc thiết kế các API cho các ứng dụng web để quản lý các resource**. RESTful là một trong những kiểu thiết kế API được sử dụng phổ biến ngày nay để cho các ứng dụng (web, mobile…) khác nhau giao tiếp với nhau. **RESTful API** không sử dụng session và cookie, nó sử dụng một access\_token với mỗi request. Dữ liệu trả về thường có cấu trúc như sau:



## Tạo mới Web API project:

Ở màn hình khởi động của Visual Studio 2019, chọn ***Create a new project***

A screenshot of a social media post

Description automatically generated

Tiếp đến chọn ***ASP.NET Core Web Application***, sau đó đặt tên cho project và solution và click OK

A screenshot of a social media post

Description automatically generated

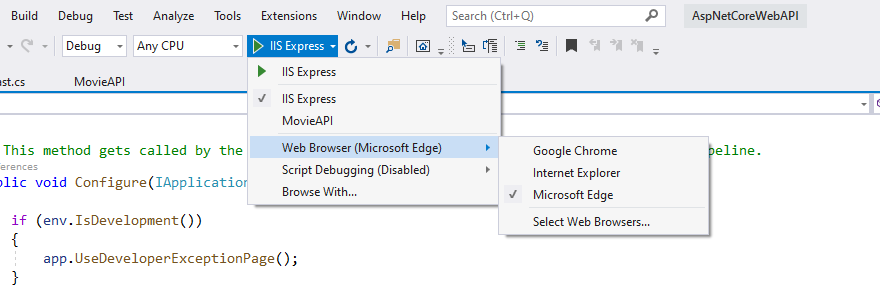
Trong mục tiếp theo, chọn ***.NET Core*** và ***ASP.NET Core 3.1*** như trong hình và chọn API, sau đó chọn **Create**

A screenshot of a cell phone

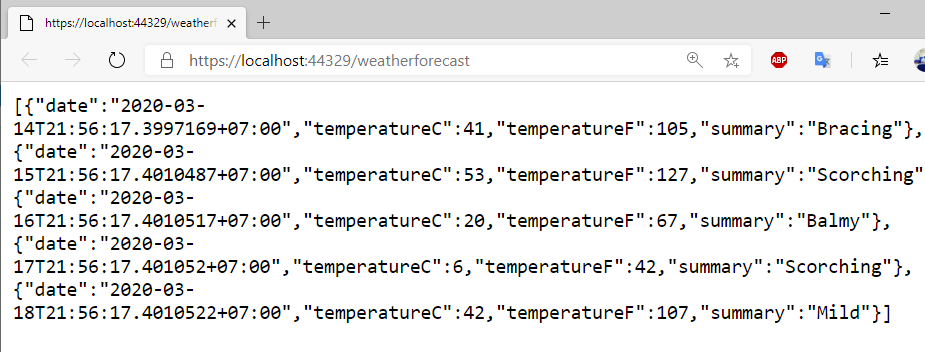
Description automatically generated

**Routing và URL**

Sau khi tạo xong project, nhấn **Ctrl+F5** để chạy ứng dụng và quan sát kết quả mặc định. Khi chạy trên **IIS Express**, mình chọn web browser mặc định là **Microsoft Edge**



Sau khi chương trình được thực thi, ta có kết quả như sau:



Đường dẫn URL mặc định: [https://localhost:{port}/weatherforecast](https://localhost:%7bport%7d/weatherforecast) chúng ta có thể tìm thấy và cài đặt trong mục ***Properties\launchSettings.json***

A screenshot of a social media post

Description automatically generated

Và các giá trị hiển thị trên khi chạy mặc định là kết quả trả về của phương thức Get của Controller **WheatherController.cs**

A screenshot of a social media post

Description automatically generated

[HttpGet] biểu diễn 1 phương thức để phản hồi lại HTTP GET request. Đường dẫn URL cho mỗi phương thức được cấu trúc như sau:

Thay thế chuỗi [controller] trong (Route("api/[controller]")). bằng tên của controller. Trong ví dụ , tên của class controller là WeatherForecastController, thì tên của controller là WeatherForecast

ASP.NET Core routing không phân biệt chữ hoa và chữ thường.

## Thêm một Model

Trong ***Solution Explorer***, click phải chuột vào project. Sau đó chọn ***Add > New Folder*** và đặt tên thư mục này là ***Models***

Click phải vào thư mục **Models**, và chọn **Add -> Class**, đặt tên class là **Movie.cs**

Sau đó thêm những thuộc tính như sau vào class:

**A screenshot of a cell phone

Description automatically generated**

Movie.cs

**Data Anotation**

Như các bạn đã thấy, trong file Movie.cs, ta đã có sử dụng data anotation

Data Annotations dùng để thêm ý nghĩa mở rộng vào dữ liệu thông qua các thẻ thuộc tính. Thuận lợi của việc sử dụng tính năng của Data Annotation là chúng ta có thể quản lý dữ liệu được định nghĩa trong một vị trí hoặc không cần phải viết lại nhiều điều luật trên nhiều vị trí khác nhau.

Ta sẽ thấy được hiệu quả của những data annotations trong quá trình thêm các giá trị vào database thông qua API client. Để thực hiện được điều này, chương trình của chúng ta cần được thực thi.

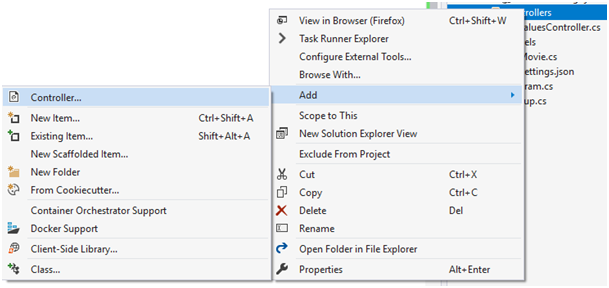
## Thêm Controller và Scaffolding

Trong mục này, chúng ta sẽ tạo controller và scaffold cho Movie model. [Entity Framework Core (EF Core)](https://docs.microsoft.com/en-us/ef/core/) sẽ được sử dụng để làm việc với database.

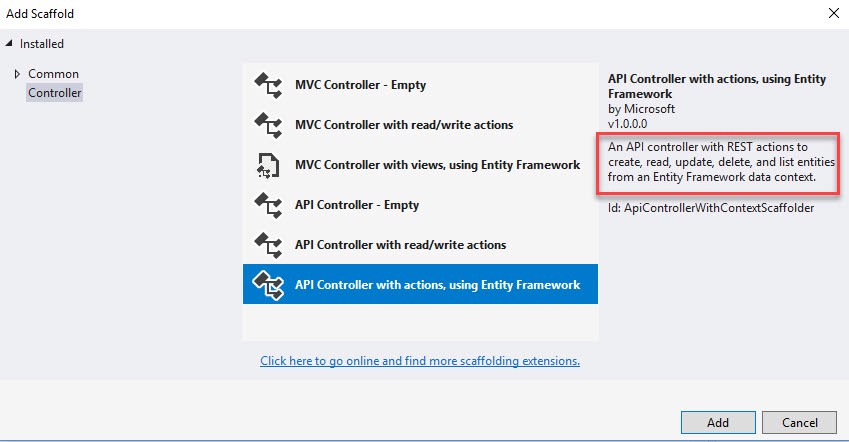
EF core là một object-relational mapping (ORM) framework , dùng để ánh xạ các đơn vị dữ liệu được mô tả bằng class vào cơ sở dữ liệu quan hệ. Nó cho phép các nhà phát triển làm việc với dữ liệu bằng cách sử dụng các đối tượng của các lớp cụ thể của miền mà không cần tập trung vào các bảng và cột cơ sở dữ liệu cơ bản nơi dữ liệu này được lưu trữ .

Chúng ta sẽ viết các class model trước, sau đó tạo database với EF Core. Đây là cách thức phát triển đang được sử dụng rất phổ biến hiện nay với tên gọi là **“Code-first”**.

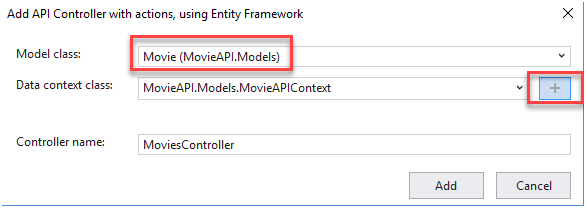
Click phải chuột vào thư mục **Controllers** và chọn **Add ->Controller**



Trong hộp thoại Add Scaffold, chọn **API Controller with actions, using Entity Framework** sau đó nhấn Add.



Trong hộp thoại tiếp theo, chọn ***Movie*** trong mục ***Model class***, sau đó chọn dấu + để thêm vào ***Data context class***.



Đến đây thì scaffolding tool bắt đầu hoạt động.

Scaffolding được hiểu là việc tự động khởi tạo của database context và và CRUD (Create, Read, Update, Delete) method.

## Tạo Database với Migrations

Migration là kỹ thuật trong việc tương tác với cơ sở dữ liệu, theo đó việc thay đổi về cấu trúc CSDL ở code sẽ được cập nhật lên CSDL đảm bảo dữ liệu đang tồn tại không bị mất, lịch sử (phiên bản) cập nhật được lưu lại sau mỗi lần cập nhật.

Ta sẽ thực hiện việc thêm 1 Migration bằng các bước như sau:

Vào ***Tools -> NuGet Package Manager > Package Manager Console(PMC)*** và chạy câu lệnh sau: Add-Migration Inittial

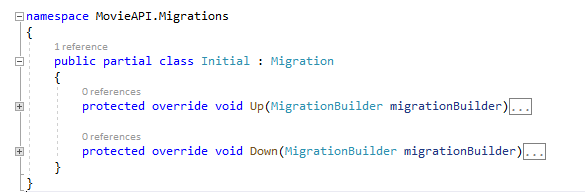
Sau lệnh này, nó sẽ tạo ra 2 file trong thư mục ***Migrations***:

****

Số 20200312164055 sinh ra theo thời điểm chạy lệnh. 2 file này chứa thông tin để có thể cập nhật (hoặc tạo) database đúng cấu trúc Model ở thời điểm tạo Migration.

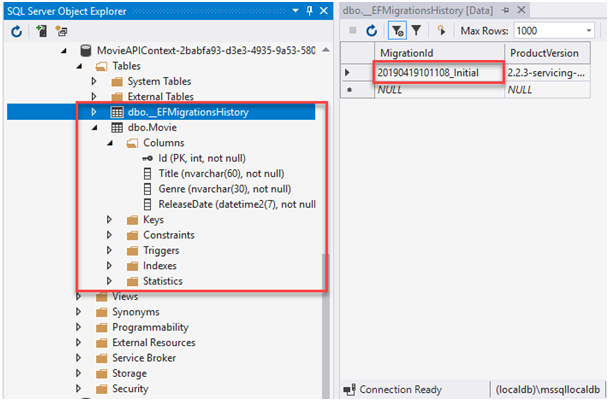
***MovieAPIContextModelSnapshot.cs*** là snapshot (ảnh chụp) để tạo được cấu trúc database theo các Model hiện tại.

Mỗi một Migration có một lớp kế thừa từ lớp Migration được tạo ra, trong nó có hai phương thức là **Up** và **Down** - để thực hiện chuyển từ phiên bản thấp đến phiên bản này (Up) hoặc đang từ phiên bản này lùi về phiên bản trước (Down).

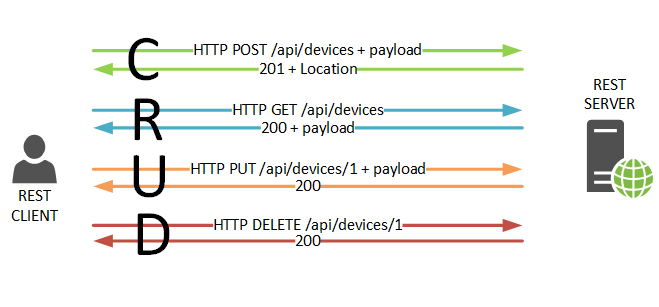
****

Bước tiếp theo, chạy câu lệnh Update-Database trong PMC.

Sau đó, ta kiểm tra xem database đã được tạo hay chưa. Chọn ***View -> SQL Server Object Explorer****.* Chúng ta sẽ thấy database đã được khởi tạo như hình sau



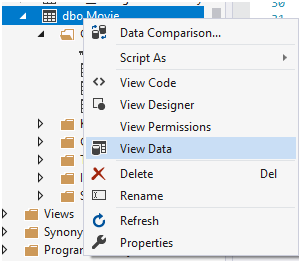
## CRUD Method



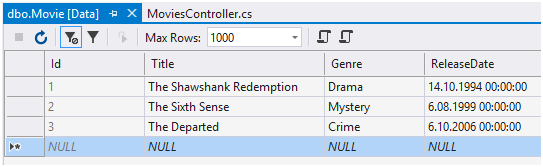
Các phương thức CRUD trong MovieController Các phương thức đều trả về các HTTP ResponseMessage. Nó giúp người thực hiện test API dễ dàng hơn trong việc kiểm tra.

## Test API sử dụng Postman

Trước hết, hãy thêm một vài dữ liệu mẫu vào bảng movie trong database nhé. Vào mục **SQL Server Object Explorer**, click phải vào bảng **Movie** -> **View Data**

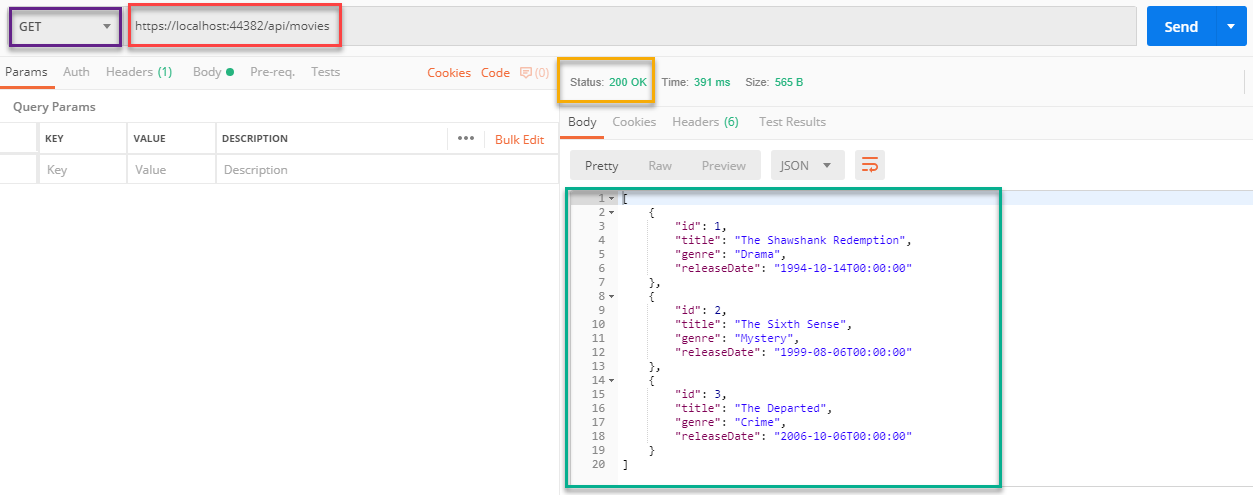


Nhập thủ công một vài thông tin vào database.



Trường ID chúng ta không cần phải nhập vì SQL Server đã tự động làm việc này cho chúng ta.

Khởi động Postman, chạy chương trình, copy URL [http://localhost:xxxxx](about:blank) và dán vào thanh Url trong Postman.



Kết quả sẽ được như hình trên. Chúng ta có chính xác một mảng lưu giá trị của các phim mà lúc nãy chúng ta đã thêm vào database.

Như vậy thông qua bài viết này, mình đã giúp các bạn biết được cách tạo một Web API với ASP.NET Core, cùng với đó là giải thích các thành phần có liên quan, cũng như việc kiểm thử API. Hi vọng bài viết này sẽ có ích và phục vụ tốt cho các bạn đang muốn bắt đầu với .NET Core. Xin cảm ơn và hẹn gặp lại ^\_^

**TÀI LIỆU THAM KHẢO**

1. Microsoft document - Introduction to ASP.NET Core

<https://docs.microsoft.com/en-us/aspnet/core/web-api/?view=aspnetcore-3.1>

1. Microsoft document - Create web APIs with ASP.NET Core

<https://docs.microsoft.com/en-us/aspnet/core/tutorials/first-web-api?view=aspnetcore-3.1&tabs=visual-studio>

1. Medium - How To Build a RESTful API with ASP.NET Core

<https://medium.com/net-core/how-to-build-a-restful-api-with-asp-net-core-fb7dd8d3e5e3>

1. Learn ASP.NET Core 3.1 - Full Course for Beginners [Tutorial]

<https://youtu.be/C5cnZ-gZy2I>