TRƯỜNG ĐẠI HỌC THÔNG TIN LIÊN LẠC

**KHOA CÔNG NGHỆ THÔNG TIN**



**ĐỀ CƯƠNG**

**THỰC TẬP TỐT NGHIỆP**

**Tên đề tài:**

**XÂY DỰNG WEBSITE MITECHCENTER.VN**

**SỬ DỤNG ASP.NET CORE**

**Lớp: DHCN3A**

**Năm học: 2019 – 2020**

**KHÁNH HOÀ, THÁNG 4/2020**

TRƯỜNG ĐẠI HỌC THÔNG TIN LIÊN LẠC

**KHOA CÔNG NGHỆ THÔNG TIN**



**ĐỀ CƯƠNG**

**THỰC TẬP TỐT NGHIỆP**

**Tên đề tài:**

**XÂY DỰNG WEBSITE MITECHCENTER.VN**

**SỬ DỤNG ASP.NET CORE**

**Lớp: DHCN3A**

**Năm học: 2019 – 2020**

Sinh viên thực hiện: Nguyễn Trọng Nghĩa

**KHÁNH HOÀ, THÁNG 4/2020**

MỞ ĐẦU

**1. Cơ sở khoa học và thực tiễn của đề tài**

Ngày nay, Internet đã trở thành một phần không thể thiếu trong cuộc sống. Với hơn 64 triệu người dùng Internet chiếm tới hơn 66% dân số Việt Nam – *Theo số liệu năm 2019 của vnetwork.vn*, vì vậy nhu cầu của việc tra cứu và tìm kiếm các thông tin cần thiết đến từ phía những người dùng này từ các website là vô cùng lớn. Chính vì vậy, vai trò của website hiện nay là không thể thay thế được. Cũng dễ nhận thấy rằng hầu hết các tác vụ từ đơn giản đến phức tạp đều có thể thực hiện thông qua các website. VD: Nghe nhạc, chỉnh sửa video, chỉnh sửa ảnh, cung cấp thông tin, marketing hay giới thiệu về doanh nghiệp,…

Với sự phát triển mạnh mẽ như vậy của các loại hình website, đã có rất nhiều công nghệ ra đời để góp phần hỗ trợ chúng ta có thể dễ dàng xây dựng nên những website ưng ý như Laravel, Wordpress, Nuxt.js hay ASP.NET Core,…

Tuy nhiên, đối với môi trường phát triển của cơ quan hay doanh nghiệp, đặc biệt là những nơi có vốn đầu tư nhà nước thì việc an toàn bảo mật, tính ổn định và lâu bền luôn được đặt lên hàng đầu, chính vì vậy việc chọn ASP.NET Core là thích hợp nhất.

**2. Lý do chọn đề tài**

Hiện tại website của Trung tâm CNTT & NN – Trường ĐH Thông Tin Liên Lạc hiện đang vận hành bằng Laravel Framework, đây là một Framework tốt và hoàn toàn miễn phí. Tuy nhiên xét về độ bảo mật, tính bền vững thì không thể phủ nhận đi ưu điểm của ASP.NET Core.

Chính vì vậy dưới sự yêu cầu đổi mới nền tảng website để hỗ trợ quảng bá Trung tâm CNTT & NN – Trường ĐH Thông Tin Liên Lạc và nâng cao tính bảo mật và ổn định trong quá trình vận hành nên em đã chọn và thực hiện nhiệm vụ do cơ sở giao phó với đề tài mang tên “**XÂY DỰNG WEBSITE MITECHCENTER.VN SỬ DỤNG ASP.NET CORE**” dưới sự hướng dẫn của ThS. Trần Thị Mỹ Hiền và sự hỗ trợ đến từ đội ngũ các thành viên phát triển ứng dụng của MitechCenter về việc xây dựng, phát triển và hoàn thiện đề tài.

**3. Mục đích của đề tài**

Phân biệt được .NET Core và .NET Framework, nắm vững cấu trúc của ASP.NET Core và mô hình MVC từ đó phát triển ứng dụng và đưa vào sử dụng trong việc hỗ trợ cập nhật website hiện tại của cơ sở nhằm nâng cao tính ổn định bền vững cũng như bảo mật trong quá trình vận hành.

**4. Nhiệm vụ của đề tài**

Hệ thống của website được xây dựng với các yêu cầu phải thực hiện được các chức năng cơ bản đã có trong phiên bản đang vận hành trước đó của website https://mitechcenter.vn nhưng sử dụng công nghệ đổi mới là ASP.NET Core với các yêu cầu như sau:

- Phân biệt được .NET Core và .NET Framework

- Hiểu về mô hình MVC trong ASP.NET Core

- Giới thiệu về ASP.NET Core

- Cách cài đặt môi trường phát triển

- Tiến hành xây dựng website với các chức năng tương tự như trang đang vận hành (Trang chủ, khóa học, liên hệ, bài viết,…)

**5. Phương pháp nghiên cứu**

Nghiên cứu các tài liệu liên quan đến họ .NET, các tài liệu chuẩn do tập đoàn Microsoft cung cấp. Xây dựng được website, sau đó tiến hành cài đặt và chạy thử nghiệm.

Kết hợp với sự hướng dẫn của giáo viên cùng các ý kiến đóng góp, hỗ trợ, chỉ bảo từ đội ngũ phát triển phần mềm và ứng dụng tại Trung tâm CNTT & NN Trường Đại Học Thông Tin Liên Lạc.

**6. Nội dung nghiên cứu**

Chương 1: PHÂN BIỆT .NET CORE VÀ .NET FRAMEWORK

Chương 2: GIỚI THIỆU VỀ ASP.NET CORE

Chương 3: CÀI ĐẶT MÔI TRƯỜNG PHÁT TRIỂN

Chương 4: XÂY DỰNG WEBSITE

Chương 1  
PHÂN BIỆT .NET CORE VÀ .NET FRAMEWORK

1.1. Tổng quan về .NET Core

1.1.1. Khái niệm .NET Core

.NET Core là một mã nguồn mở mới và là một 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 hay back-end cho mobile. Vì là một cross-platform nên .NET Core hỗ trợ hầu hết cho các nền tảng phổ biến hiện nay như Windows, Linux, macOS.

1.1.2. Khi nào sử dụng .NET Core

Theo những định nghĩa được nêu ra như trên thì chúng ta có thể sử dụng .NET Core trong một số trường hợp như sau:

- Cần đa nền tảng (Cross-platform).

- Mục tiêu nhắm đến là kiến trúc Microservices.

- Cần sử dụng đến Docker và Doker containers trong dự án.

- Cần xây dựng ứng dụng Machine Learning với kiến thức sẵn có về C# hoặc F#.

- Cần cho các hệ thống có hiệu xuất cao và có thể mở rộng.

- Các phiên bản cần chạy đa luồng khi xử lý.

- Đòi hỏi hiệu năng cao.

1.2. tổng quan về .NET Framework

1.2.1. Khái niệm .NET Framework

.NET Framework là một nền tảng lập trình và cũng là một nền tảng thực thi ứng dụng chủ yếu trên hệ điều hành Microsoft Windows được phát triển bới Microsoft. Các chương trình được viết trên nền .NET Framework sẽ được triển khai trong môi trường phần mềm (ngược lại với môi trường phần cứng) được biết đến với tên Common Language Runtime (CLR). Môi trường phần mềm này là một máy ảo trong đó cung cấp các dịch vụ như an ninh phần mềm (security), quản lý bộ nhớ (memory management) và các xử lý lỗi ngoại lệ (exception handing).

1.2.2. Khi nào sử dụng .NET Framework

Theo những định nghĩa được nêu ra như trên thì chúng ta có thể sử dụng .NET Framework trong một số trường hợp như sau:

- Phát triển tiếp các ứng dụng cũ đã được sử dụng .NET Framework trước đó.

- Nếu ứng dụng đã sử dụng .NET Framework và Nuget của bên thứ 3 không hỗ trợ cho .NET Core.

- Nếu ứng dụng được thiết kế cho .NET Framework và không hỗ trợ cho .NET Core.

- Nếu ứng dụng sử dụng platfrom không hỗ trợ .NET Core.

Chương 2  
GIỚI THIỆU VỀ ASP.NET CORE

2.1. tổng quan về asp.net core

2.1.1. Khái niệm về ASP.NET Core

ASP.NET Core là một chứng minh cụ thể cho việc sử dụng .NET Core, chính vì vậy ASP.NET Core cũng có những khái niệm kế thừa từ .NET Core như cũng là một mã nguồn mở (open-source) mới và là một 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 back-end mobile, nó là một tập hợp các thư viện chuẩn như một framework để xây dựng ứng dụng web.

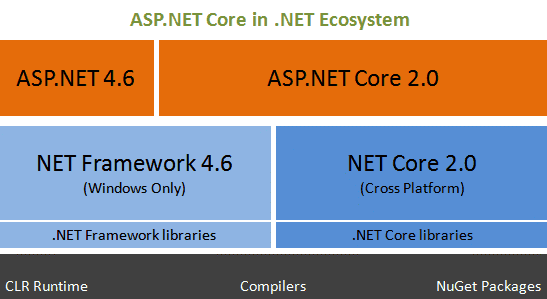
Tuy nhiên ứng dụng ASP.NET Core có thể chạy trên .NET Core hoặc trên phiên bản đầy đủ của .NET Framework. Nhưng không vì thế mà có thể nói ASP.NET Core là một phiên bản tiếp theo của ASP.NET được, nó là một cái tên mới được xây dựng từ đầu với sự thay đổi lớn về kiến trúc trong đó nên kết quả là ASP.NET Core gọn hơn và phân chia module tốt hơn.

ASP.NET Core gồm các thành phần theo hướng module nhằm tối thiểu tài nguyên và chi phí phát triển, điều này giúp giữ lại được sự mềm dẻo trong việc xây dựng giải pháp cho phát triển phần mềm và ứng dụng.

ASP.NET Core là một mã nguồn mở, đây là một thay đổi rất lớn và có tầm quan trọng bậc nhất của ASP.NET Core. Điều mà trước đây khó có một lập trình viên nào có thể nghĩ đến khi đây là một công nghệ do Microsoft phát triển.

2.1.2. Khác biệt quan trọng giữa ASP.NET Core và ASP.NET

Về cơ bản .NET Framework, .NET Core là hai phiên bản .NET khác nhau (nghĩa là mỗi phiên bản có Runtime, Libraries và Tooling riêng). Cụ thể như hình dưới.



Hình 2.1. Mô hình hóa của các nền tảng .NET khác nhau

Vì thế, sự khác nhau quan trọng giữa ASP.NET Core và ASP.NET có thể được nêu ra theo bảng dưới đây:

Bảng 2.1. Sự khác nhau quan trọng giữa ASP.NET Core và ASP.NET

|  |  |  |
| --- | --- | --- |
| Tiêu chí | ASP.NET Core | ASP.NET |
| Lịch sử | Phiên bản đầu tiên được giới thiệu vào ngày 27-06-2016 cùng với bản .Net Core phiển bản đầu tiên | Ra đời vào tháng 01/2002 cùng với sự giới thiệu của .Net Framwork 1.0 |
| Phát triển | Microsoft và cộng đồng | Microsoft |
| Hệ điều hành | Hỗ trợ trên các hệ điều hành Windows, macOS, hoặc Linux | Chỉ hỗ trợ Windows |
| Các công nghệ hỗ trợ | Hỗ trợ các công nghệ MVC, Web API, và SignalR, Entity Framework Core | Hỗ trợ các công nghệ Web Forms, SignalR, MVC, Web API, WebHooks, Web Pages, Entity Framwork |
| Hỗ trợ phiên bản | Nhiều phiên bản trên một máy | Một phiên bản trên một máy |
| Công cụ phát triển | Phát triển ASP.NET Core bằng các công cụ như : Visual Studio, Visual Studio for Mac, và Visual Studio Code hay thậm chí là các Text Editor bất kỳ | Phát triển ASP.NET bằng Visual Studio |
| Ngôn ngữ lập trình | C#, F# | C#, VB, F# |
| Hiệu xuất | Hiệu xuất cao hơn ASP.NET | Tốt |
| Biên dịch | Có thể lựa chọn giữa .NET Framework và .NET Core runtime | Sử dụng .NET Framework runtime |
| Nền tảng | Hoàn toàn được thiết kế mới | Đã có từ lâu |
| Hỗ trợ WebForms | Không | Có |
| System.web.dll | Nhỏ, nhẹ và module hóa | Cồng kềnh |

2.1.2. Các đặc tính quan trọng

Để hiểu rõ về ASP.NET, chúng ta cần nắm các đặc tính quan trọng sau đây:

- Hỗ trợ xây dựng nên những ứng dụng đa nền tảng trên Windows, Mac và Linux.

- ASP.NET Core là một sản phẩm hợp nhất nhiều yếu tố, có thể kể đến là sự hợp nhất ASP.NET MVC và ASP.NET Web API.

- Có thể host trên IIS hoặc tự Host.

- Có sẵn Dependency Injection.

- Dễ dàng tích hợp với các Framework Front-end như Angular, Vue.js,…

- Hỗ trợ cấu hình cho nhiều môi trường.

- Cơ chế HTTP Request pipline mới.

- Hỗ trợ quản lý phiên bản.

- Dùng chung toàn bộ Nuget Package.

2.2. Mô hình MVC trong ASP.NET Core

2.2.1. Khái quát về mô hình MVC

Mô hình MVC là một kiến trúc thiết kế của một phần mềm công nghệ thông tin, trong đó có cả PHP, Node.js,… Khi một phần mềm sử dụng mô hình này thì nó sẽ phải chia ra thành ba thành phần chính **M**odel-**V**iew-**C**ontroller (MVC) với 3 nhiệm vụ khác nhau như sau:

- **Model**: Là thành phần chứa tất cả các phương thức xử lý, truy xuất database, đối tượng mô tả dữ liệu như các Class, hàm xử lý,..

- **View**: Là thành phần đảm nhiệm việc hiển thị thông tin, tương tác với người dùng. Hoặc chúng có thể hiểu nôm na là phần giao diện người dùng.

- **Controller**: Là phần xử lý và điều hướng các hành động của client, từ đó đưa ra các xử lý với database nếu có. Hay nói cách khác thì ***Controller*** *là cầu nối giữa* ***View*** *và* ***Model****.*

2.2.2. MVC trong ASP.NET Core

ASP.NET Core MVC là một Framework mã nguồn mở nhẹ, giúp tối ưu hóa hiệu năng của ứng dụng với ASP.NET Core.

ASP.NET Core MVC cung cấp các tính năng dựa trên mô hình xây dựng website động cho phép phân chia rõ ràng các khối lệnh. Đồng thời cung cấp cho lập trình viên toàn quyền kiểm soát và sử dụng các tiêu chuẩn web mới nhất hiện nay.

Các tính năng cơ bản của ASP.NET Core MVC bao gồm:

- Routing

- Model binding

- Model validation

- Dependency injection

- Filters

- Areas

- Web APIs

- Testability

- Razor view engine

- Strongly typed views

- Tag Helpers

- View Components

a) Routing

Routing sẽ xác định URL và điều khiển thông tin trong ứng dụng với URL mà người dùng web nhập vào. Tất cả các cấu hình Routing của một ứng dụng ASP.NET Core MVC được lưu trữ trong RouteTabe, nó đóng vai trò định tuyến các URL để xác định các lớp xử lý tương ứng khi có request được gửi đến từ người dùng web.

VD:

endpoints.MapControllerRoute(name: "default",pattern: "{controller=Home}/{action=Index}/{id?}");

Hình 2.2. Default Endpoints Route trong project

[Route("[controller]")]

    public class MCMSController : Controller

    {

        public IActionResult Index()

        {

            return View();

        }

    }

Hình 2.3. Route đến trang chủ của CMS

b) Model binding

Trong ASP.NET Core MVC Model binding chuyển đổi dữ liệu yêu cầu từ phía client (from values, route data, query string parameters, HTTP headers) vào bên trong đối tượng để controller có thể xử lý. Kết quả là Controller không phải tìm ra dữ liệu từ đâu đến nên công việc còn lại chỉ đơn giản là kiển tra dữ liệu và tham số từ phương thức action.

public async Task<IActionResult> Login(LoginViewModel model, string returnUrl = null)

{

...

}

Hình 2.4. Ví dụ về model binding

c) Model validation

ASP.NET Core MVC hỗ trợ ràng buộc dữ liệu cho các thuộc tính trong model, các thuộc tính sẽ được kiểm tra ở client xem có hợp lệ hay không trước khi giá trị của thuộc tính đó được gửi về Server. Cũng như trên Server trước khi action của Controller gọi.

using System.ComponentModel.DataAnnotations;

public class LoginViewModel

{

    [Required]

    [EmailAddress]

    public string Email { get; set; }

    [Required]

    [DataType(DataType.Password)]

    public string Password { get; set; }

}

Hình 2.5. Ví dụ về model validation

Sau đó model có thể được gọi trong action của controller như sau:

public async Task<IActionResult> Login(LoginViewModel model, string returnUrl = null)

{

    if (ModelState.IsValid)

    {

      // work with the model

    }

    // At this point, something failed, redisplay form

    return View(model);

}

Hình 2.6. Ví dụ về sử dụng model validation

d) Dependency injection

Trong ASP.NET Core MVC controller có thể gửi yêu cầu cần thiết đến các service thông qua cấu trúc xây dựng của chúng.

@inject SomeService ServiceName

<!DOCTYPE html>

<html lang="en">

<head>

    <title>@ServiceName.GetTitle</title>

</head>

<body>

    <h1>@ServiceName.GetTitle</h1>

</body>

</html>

Hình 2.7. Ví dụ về Dependency injection

e) Filters

Filters giúp các lập trình viên đóng gói “cross-cutting concerrns”. Giống như là xử lý ngoại lệ và phân quyền. Filter được kích hoạt để chạy trước và sau các action của controller. Ví dụ về phần quyền [Authorize] được đặt trước action như sau:

[Authorize]

public class AccountController : Controller

{

    ...

}

Hình 2.8. Ví dụ sử dụng Authorize Filters

f) Areas

Khi một website ASP.NET Core MVC trở nên quá lớn và quá phức tạp, số Controller chắc chắn sẽ tăng lên, với nhiều controller như vậy, chúng ta có thể thuộc về một nhóm như phần Administrator, trang chủ,… Areas cho phép chúng ta chia các Controllers, Modules và các Views tới các vị trí khác nhau trong solution với cùng một thư mục độc lập.

g) Web APIs

Ngoài việc là một nền tảng tuyệt vời để xây dựng các trang web. ASP.NET Core MVC hỗ trợ rất nhiều cho việc xây dựng API Web. Chúng ta có thể xây dựng các dịch vụ tiếp cận nhiều khách hàng bao gồm trình duyệt web và thiết bị di động.

h) Razor view engine

ASP.NET Core MVC views sử dụng Razor view engine để render các code html đến View. Razor được sử dụng để tự động tạo nội dung web trên máy chủ.

<ul>

  @for (int i = 0; i < 5; i++) {

    <li>List item @i</li>

  }

</ul>

Hình 2.9. Ví dụ sử dụng Razor view engine

Chương 3  
CÀI ĐẶT MÔI TRƯỜNG PHÁT TRIỂN

3.1. Tổng quan về các công cụ cho ASP.NET Core

ASP.NET Core có thể được phát triển và hoạt động trên cả Windows, Linux và macOS. Để thực thi ứng dụng, chúng ta cần cài đặt .NET Core runtime. Để phát triển ứng dụng, chúng ta cần cài đặt .NET Core SDK. Ngoài ra chúng ta cũng cần đến công cụ cho việc hỗ trợ viết code, biên dịch và sửa lỗi.

Để phát triển ứng dụng ASP.NET Core chúng ta cần các công cụ sau:

- Bộ ASP.NET Core SDK.

- Một môi trường phát triển ứng dụng tích hợp như Visual Studio, JetBrains Rider hoặc một code editor như Visual Studio Code,…

3.2. Cài đặt ASP.NET Core SDK

3.2.1. Tổng quan về Visual Studio 2019

Visual Studio là IDE riêng của Microsoft ban đầuc có tên là Project Boston và được phát hành vào năm 1997. Sau đó, Microsoft đã kết hợp tất cả các công cụ phát triển của họ và đóng gói thành một sản phẩm duy nhất. Ban đầu phần mềm được chia làm hai phiên bản là **Visual Studio Professional** và **Visual Studio Enterprise**. Phiên bản **Professional** và **Enterprise** cao cấp được đóng gói trong 3 đĩa CD. Qua một quá trình dài phát triển hơn 20 năm, đến hiện nay Microsoft đã cho ra mắt Visual Studio 2019 là phiên bản mới nhất từ nhóm công cụ phát triển tại Microsoft, bao gồm 3 phiên bản chính là:

- Community (Phiên bản miễn phí cho sinh viên và cá nhân)

- Professional (Phiên bản dành cho các team lập trình nhỏ)

- Enterprise (Phiên bản có khả năng mở rộng cao hỗ trợ nhiều giải pháp cho doanh nghiệp)

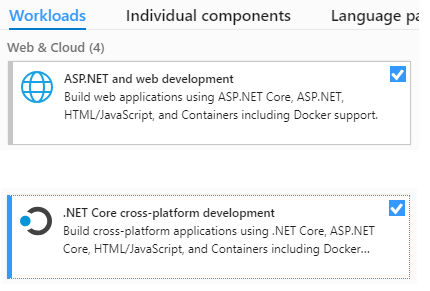
Đối với sinh viên và các đối tượng người dùng là cá nhân, sử dụng phiên bản Community là lụa chọn thích hợp nhất.

3.2.2. Cài đặt ASP.NET Core SDK với Visual studio 2019

Đối với việc sử dụng Visual Studio 2019 trên Windows để cài đặt được .NET Core Runtime và SDK là vô cùng đơn giản.

Từ .NET Core 3.0 trở đi đòi hỏi sử dụng Visual Studio 2019 (v16.3 trở lên), nếu nằm trong các trường hợp sau, chúng ta cần thực hiện để tiến hành nâng cấp và cập nhật để có thể hoàn tất:

- Nếu chua có hoặc đang dùng phiên bản Visual Studio đã cũ, chúng ta cần cài đặt mới Visual Studio 2019 (Sử dụng bản Community miễn phí để phù hợp với sinh viên). Trong quá trình cài đặt, tại **Workloads** chọn **ASP.NET and web development** (dùng để phát triển ứng dụng trên cả ASP.NET và ASP.NET Core) hoặc **.NET Core cross-platform development** (dùng để phát triển ứng dụng trên .NET Core và ASP.NET Core).



Hình 3.1. Chọn Workloads

- Nếu đã cài đặt sẵn Visual Studio 2019, hãy tiến hành update lên build mới nhất. Sau đó tiến hành chạy chương trình **Visual Studio Installer** > chọn **Modify** > chọn mục **Workloads** và thực hiện như trên.

Lựa chọn một trong hai workload trên sẽ giúp chúng ta cài đặt tất cả các thành phần cần thiết cho phát triển và thực thi ứng dụng ASP.NET Core trên Windows. Khi qua trình cài đặt hoàn tất, chúng ta đã sẵng sàng cho việc phát triển và chạy ứng dụng ASP.NET Core.

Chúng ta có thể chạy lệnh **dotnet --version** trên *Command Prompt* hoặc *PowerShell* để kiểm tra kết quả.

3.2.3. Cài đặt ASP.NET Core SDK trực tiếp

Do được thiết kế để phát triển và thực thi đa nền tảng, việc cài đặt ASP.NET Core tương đối dễ dàng trên các hệ điều hành phổ biến hiện nay như Windows, Linux, MacOS. Ở đây chúng ta sẽ tiến hành cài đặt trên Windows.

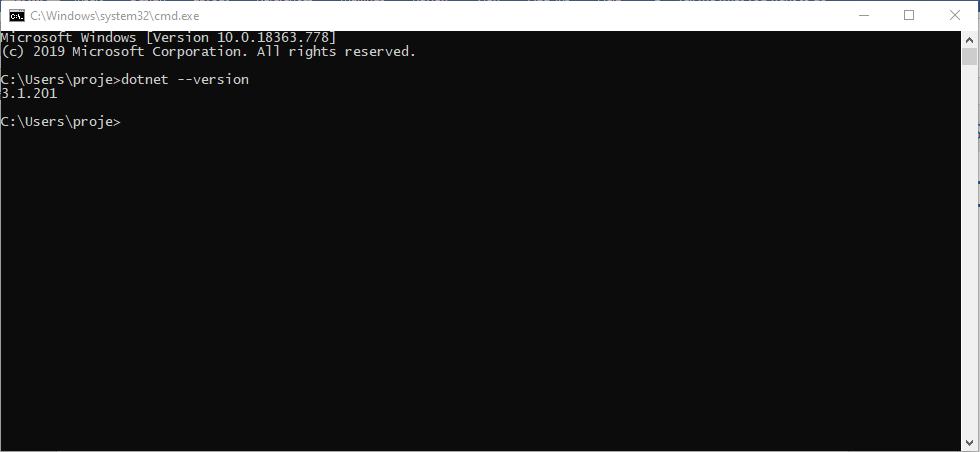
Như đã được trình bày trước đó, để thực thi ứng dụng ASP.NET Core, chúng tacafn cài đặt ASP.NET Core Runtime. Để phát triển ứng dụng, chúng ta cần cài đặt ASP.NET Core SDK. Nhưng thật thuận lợi cho chúng ta, khi cài đặt ASP.NET Core SDK thì đồng thời ASP.NET Core Runtime cũng được cài đặt.

Để tiến hành cài đặt, trước tiên chúng ta cần vào địa chỉ:

<https://dotnet.microsoft.com/download>

Sau đó tiến hành chọn và tải về bộ cài đặt của phiên bản mới nhất rồi tiến hành cài đặt nhu bất kỳ chương trình Windows bình thường nào.

Chúng ta có thể chạy lệnh **dotnet --version** trên *Command Prompt* hoặc *PowerShell* để kiểm tra kết quả.



Hình 3.2. Kiểm tra phiên bản của .NET Core SDK sau khi hoàn tất

3.3. Cơ sở dữ liệu và hệ quản trị cơ sở dũ liệu

3.3.1. Cơ sở dữ liệu SQL Server

a) Khái quát

SQL Server là một hệ quản trị cơ sở dữ liệu quan hệ (Relational Database Management System – RDBMS) sử dụng câu lệnh SQL (Transact-SQL) để trao đổi dữ liệu giữa máy khách và máy cài SQL Server. Một RDBMS bao gồm các database, database engine và các dứng dụng dùng để quản lý dữ liệu và các bộ phận khác nhau trong RDBMS.

SQL Server được tối ưu để chạy trên môi trường cơ sở dữ liệu rất lớn (Very Large Database Enviroment) lên đến TeraBytes và có thể phục vụ cùng lúc cho hàng nghìn người dùng. SQL Server có thể kết hợp với các server khác như Microsoft Internet Information Server (IIS), E-Commerce Server, Proxy Server,..

SQL Server có một vài phiên bản có thể kể đến như sau:

- Enterprise: Chứa tất cả các đặc điểm nổi bật của SQL Server, bao gồm nhân bộ máy cơ sở dữ liệu và các dịch vụ đi kèm cùng với các công cụ cho tạo và quản lý phân cụm SQL Server. Nó có thể quản lý các CSDL lớn tới 524 PetaBytes và đánh địa chỉ 12 TeraBytes bộ nhớ và hỗ trợ tới 640 bộ vi xử lý.

- Standard: Rất thích hợp cho các công ty vừa và nhỏ vì giá thành rẻ hơn nhiều so với bản Enterprise, nhưng lại bị giới hạn một số chức năng cao cấp khác.

- Developer: Có đầy đủ các tính năng của phiên bản Enterprise nhưng giới hạn nhiều điều kiện như số lượng người kết nối vào server cùng một lúc,… đây là phiên bản sử dụng cho phát triển và kiểm tra ứng dụng. Phiên bản này phù hợp cho tất cả các cá nhân, tổ chức xây dựng và kiểm tra ứng dụng.

- Express: SQL Server Express dễ sử dụng và quản trị CSDL đơn giản. Được tích hợp với Microsoft Visual Studio, nên dễ dàng để phát triển các dứng dụng dữ liệu, an toàn trong lưu trữ và nhanh trong triển khai. SQL Server Express là phiên bản **miễn phí**, không giới hạn về số cơ sở dữ liệu hoặc người sử dụng, nhưng nó chỉ dùng cho 1 bộ vi xử lý với 1 GB bộ nhớ và 10 GB file cơ sở dữ liệu. SQL Server Express là lựa chọn tốt cho những người dùng chỉ cần một phiên bản SQL Server 2005 nhỏ gọn, dùng trên máy chủ có cấu hình thấp, những nhà phát triển ứng dụng không chuyên hay những người yêu thích xây dựng các ứng dụng nhỏ.

Đối tượng của SQL Server là các bảng dữ liệu với các cột và các hàng. Cột được gọi là trường dữ liệu và hàng là bản ghi của bảng. Cột dữ liệu và kiểu dữ liệu xác định tạo nên cấu trúc của bảng.

Khi bảng được tôt chức thành một hệ thống cho một mục đích sử dụng cụ thể vào một công việc nào đó thì sẽ trở thành một cơ sở dữ liệu.

b) Cài đặt

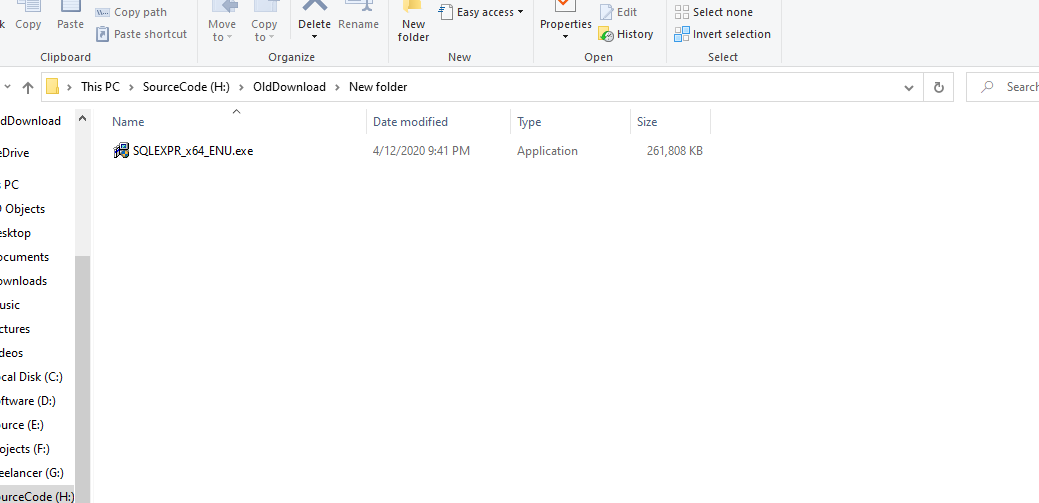
Để có thể sử dụng, và phục vụ cho mục đích phát triển cũng như áp dụng cho mô hình website nhỏ này, nên phiên bản Express của SQL Server là thích hợp nhất vì nó yêu cầu vi xử lý thấp, và cung cấp một phạm vi lưu trữ dữ liệu lên đến 10 GB. Những điều kiện như vậy là hoàn hảo để xây dựng nên một webite với quy mô nhỏ.

- Tải SQL Server

Để cài đặt SQL Server, chúng ta có thể truy cập trực tiếp vào trang chủ của Microsoft để tải bộ công cụ cho máy tính theo địa chỉ:

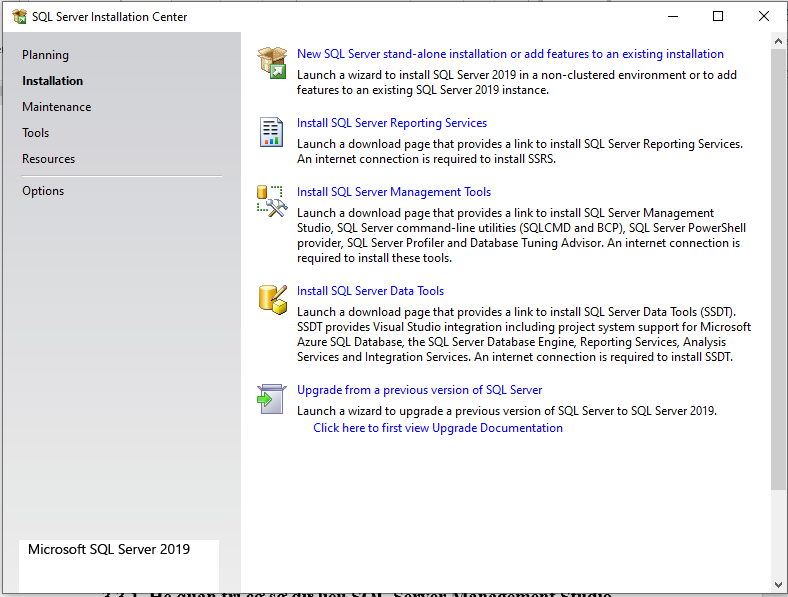
[*https://www.microsoft.com/en-in/sql-server/sql-server-downloads*](https://www.microsoft.com/en-in/sql-server/sql-server-downloads)

Có rất nhiều phiên bản khác nhau của SQL Server như 2005, 2008, 2012, 2014, 2017 và mới nhất là 2019.



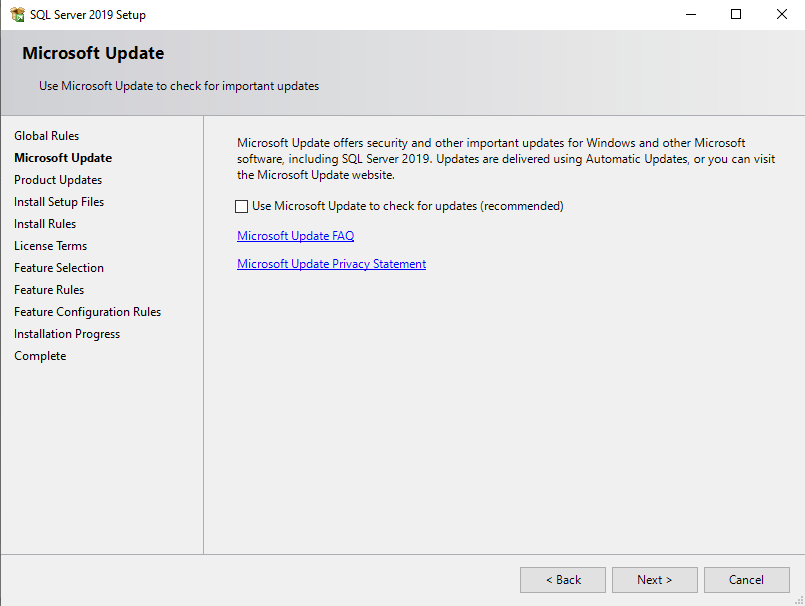
Hình 3.3. Sau khải tải về gói cài đặt của SQL Server 2019

Sau khi tải về, nhấp chuột phải vào và chọn Run as Administrator để cài đặt với quyền quản trị (Admin). Chọn New SQL Server Stand-alone installation or add features to an existing installation.



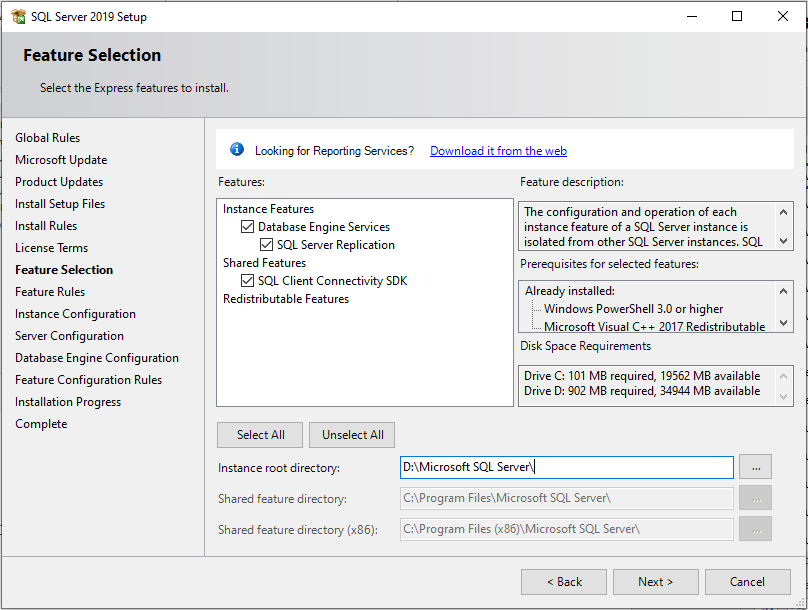
Hình 3.4. Chọn tùy chọn đầu tiên

Sau khi chạy xong, tiến hành nhấp vào **Next**.



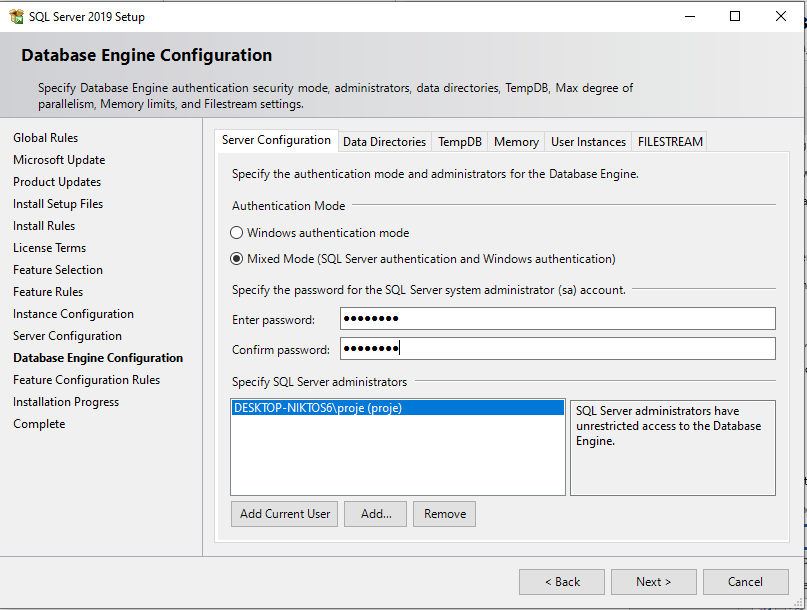
Hình 3.5. Chọn Next

Trong cửa sổ **Feature Selection** chọn tất cả các tính tăng.



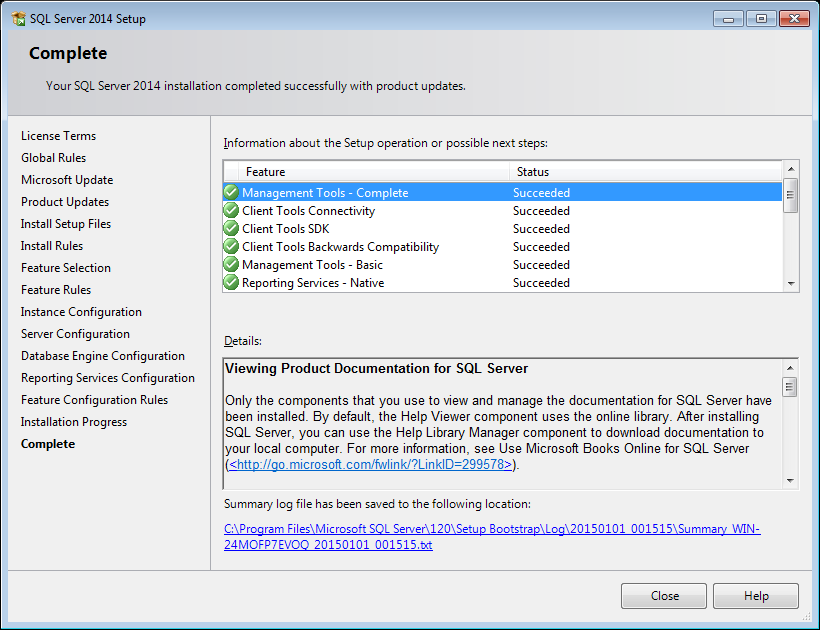
Hình 3.6. Chọn tất cả các tính năng

Sau đó tiếp tục Next ở các màn hình tiếp theo, cho đến cửa sổ **Database Engine Configuration** chọn chế độ **Mix Mode** sẽ cho phép chúng ta đăng nhập vào SQL Server theo 2 cách: Sử dụng xác thực trực tiếp của Windows, sử dụng username/password của SQL Server.



Hình 3.7. Lựa chọn Mix Mode

Sau đó tiếp tục chọn **Next** cho đến khi hoàn thành. Kết quả ta thu được là:



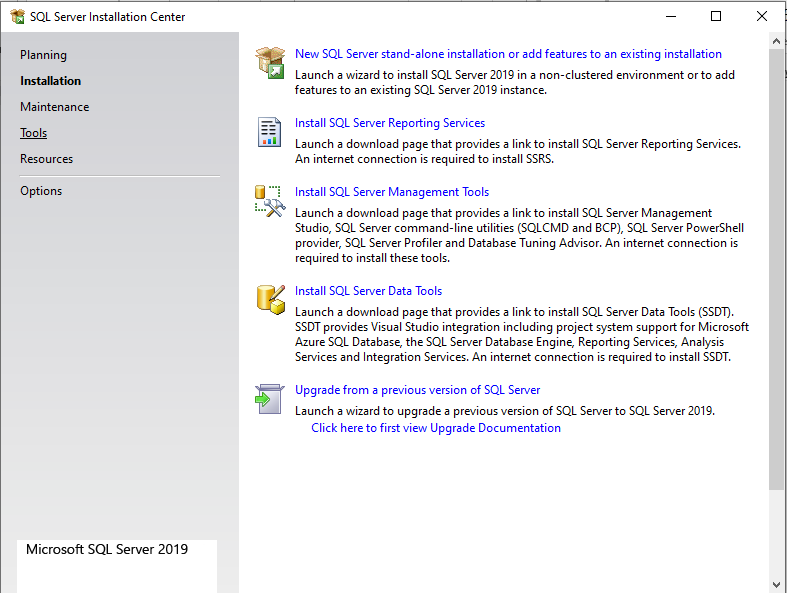
Hình 3.8. Kết quả sau khi cài đặt hoàn tất

3.3.1. Công cụ quản trị cơ sở dữ liệu SQL Server Management Studio

SQL Server Management Studio (SSMS) là một công cụ trong SQL Server, công cụ này giúp kết nối và quản lý SQL Server trên giao diện đồ họa thay vì phải dùng tới dòng lệnh.

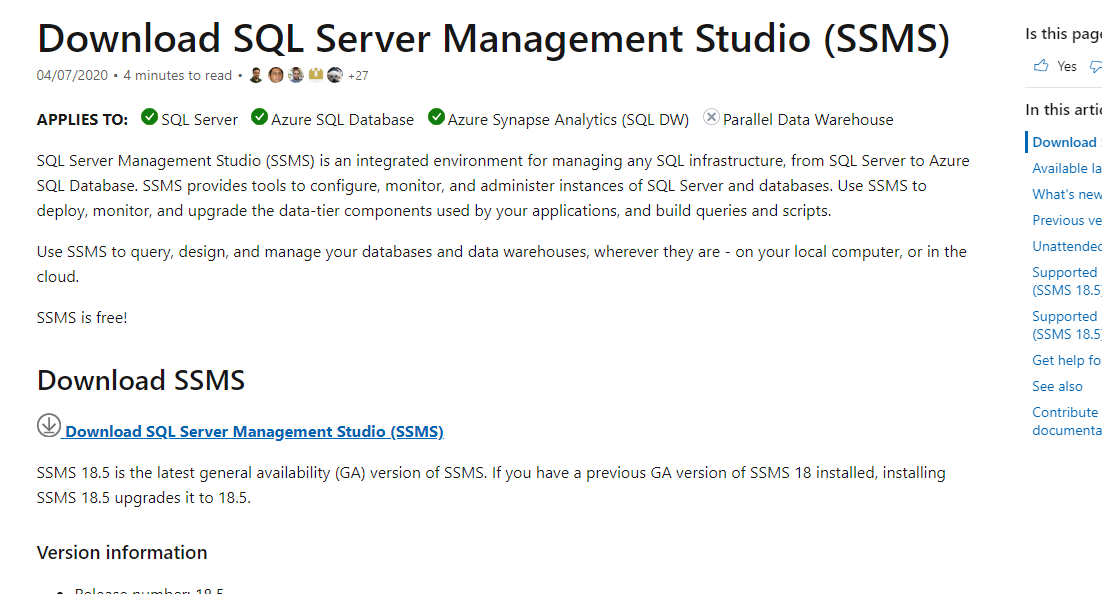
Để kết nối tới bản cài SQL Server từ xa, chúng ta cũng có thể sử dụng phần mềm này hoặc một phần mềm nào đó tương tự. SQL Server Management Studio thường được các nhà quản trị, nhà phát triển hay kiểm thử,… sử dụng.

Để cài đặt SQL Server Management Studio chúng ta cũng thực hiện tương tự như đối với SQL Server. Chúng ta cũng tiến hành mở tệp được tải như trên. Tại hộp thoại của **SQL Server Installation Center** chúng ta tiến hành chọn **Install SQL Server Management Tools**.



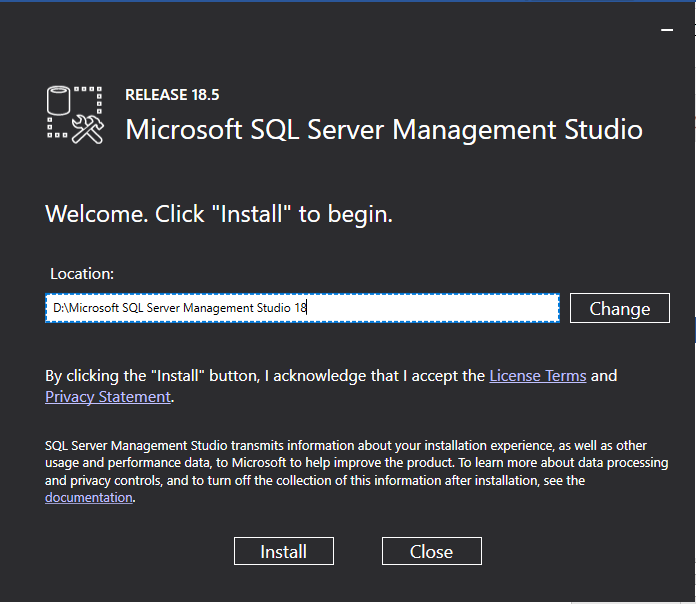
Hình 3.9. Mục chọn để cài đặt SSMS

Tiếp theo, ta sẽ được dẫn đến trang chủ của Microsoft SQL Server Management Studio. Và chọn  **Download SQL Server Management Studio (SSMS)**.



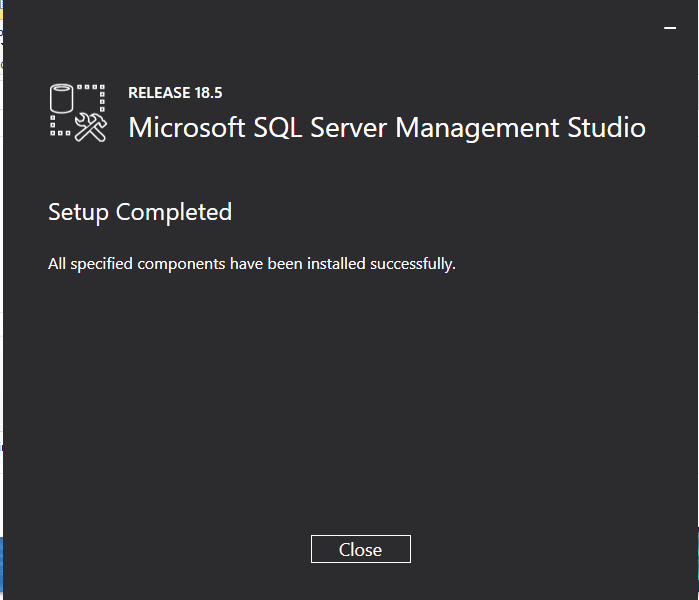
Hình 3.10. Tải tệp cài đặt cho SSMS

Sau khi hoàn tất tải về, khởi động tệp vừa tải và nhấn **Install**.



Hình 3.10. Màn hình cài đặt SSMS

Và đợi cho đến khi hoàn tất việc cài đặt.

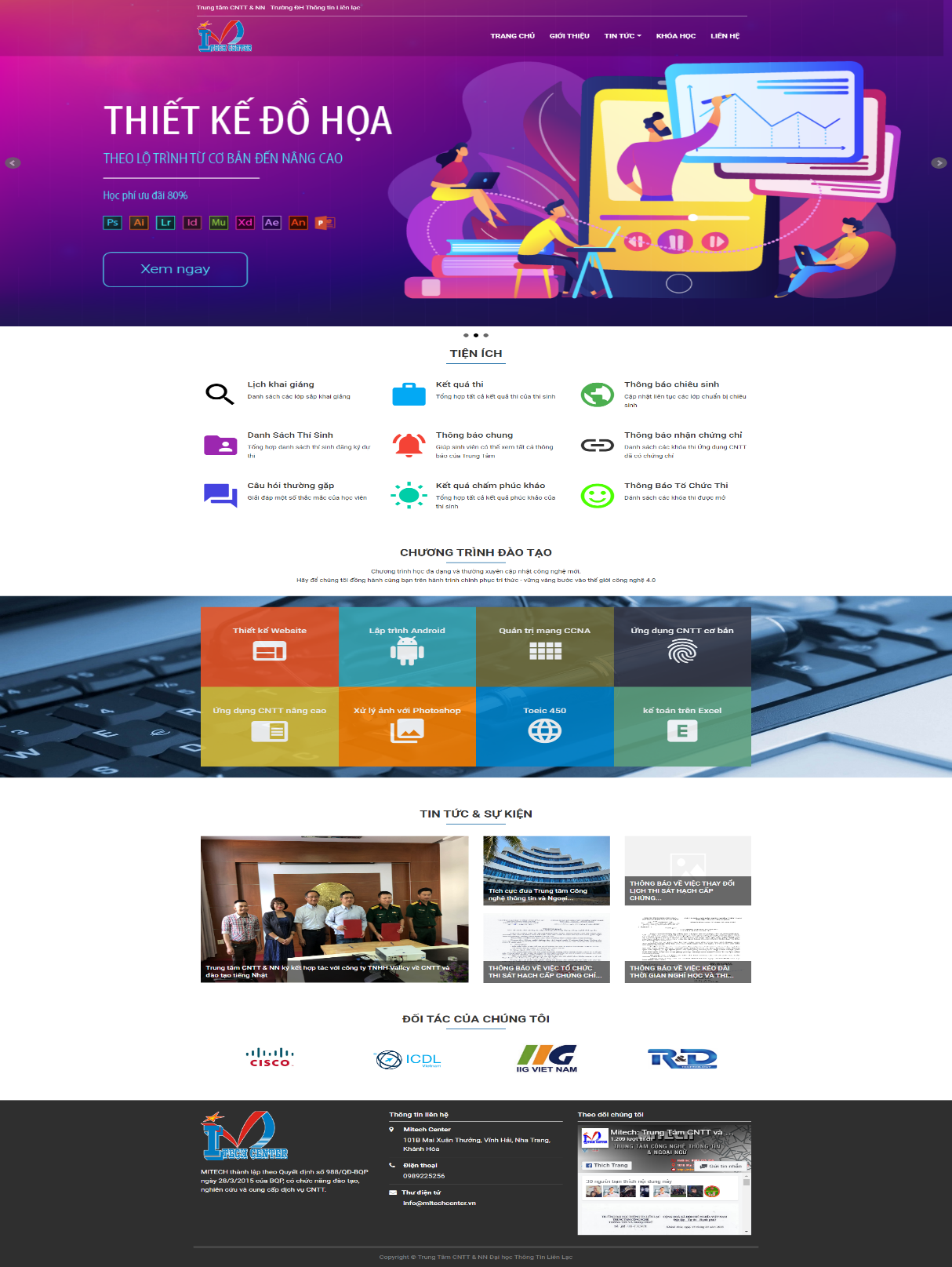


Hình 3.11. Hoàn tất việc cài đặt

Chương 4  
XÂY DỰNG WEBSITE

4.1. Khảo sát hiện trạng

Hiện tại, Trung tâm công nghệ thông tin và ngoại ngữ - Trường Đại Học Thông Tin Liên Lạc đang sử dụng website trên trang MitechCenter.vn để thực hiện việc quảng bá và truyền tải thông tin.

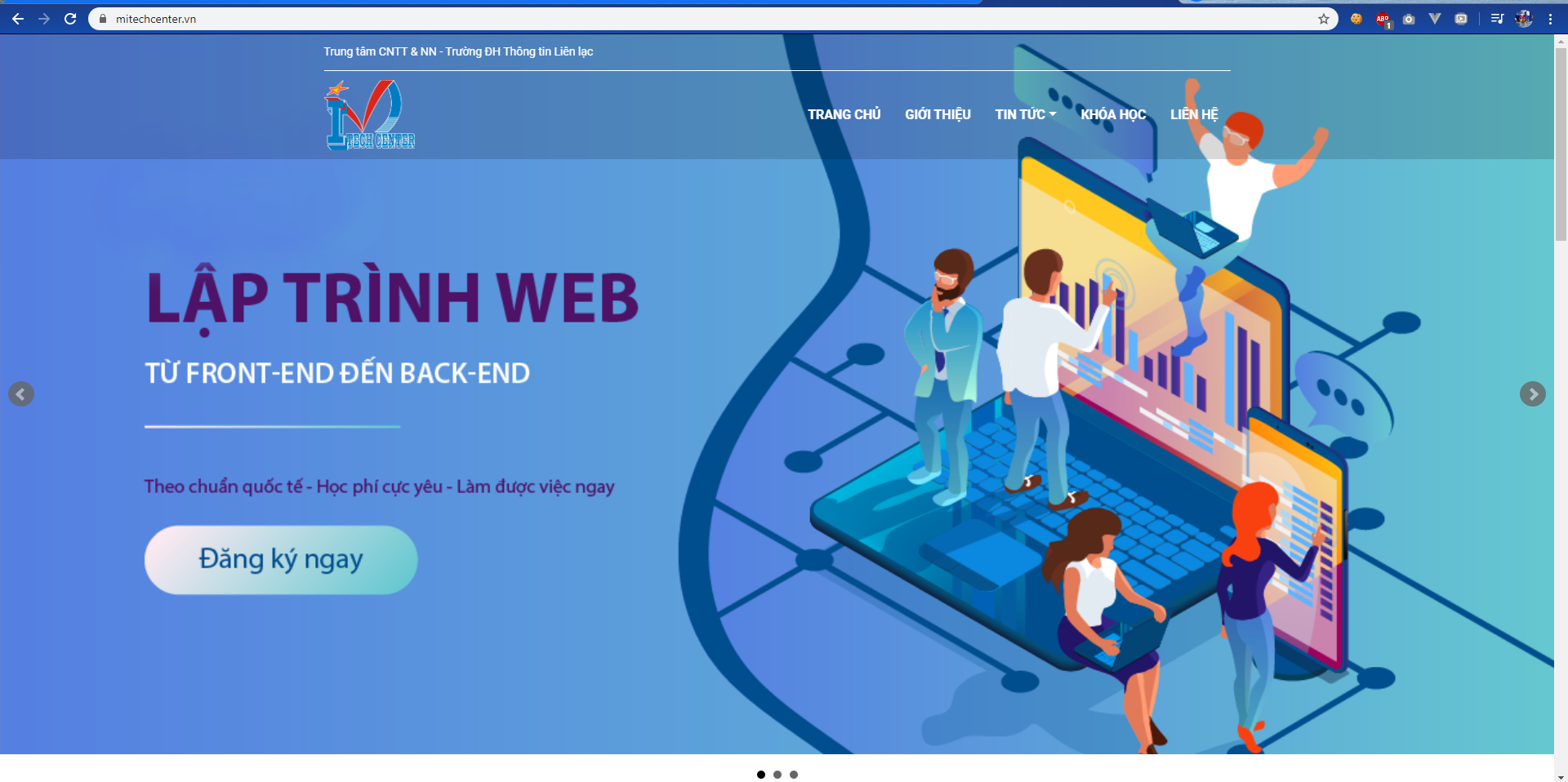


Hình 4.1. Website hiện tại của MitechCenter.vn

Tuy nhiên, hiện tại website này vẫn còn khá sơ sài và còn một số điều bất cập như sau:

- Trang chủ chỉ chủ yếu hiển thị về tin tức và chỉ cung cấp thông tin cho mảng đào tạo mà chưa tập trung hay có phân mục nào giới thiệu về các dịch vụ công nghệ thông tin của trung tâm.

- Chưa hoàn thiện về mặt carousel, khi mà chúng chỉ đơn thuần chỉ là những tấm ảnh tĩnh và không thể nhấp và các nút để truy cập đến thông tin chi tiết. Đây chính là phần dễ thu hút sự chú ý và quan tâm của người dùng nhất với hiệu ứng bắt mắt. Nhưng lại không thể mang đến thông tin chi tiết khi người dùng muốn bấm vào nút trên ảnh.

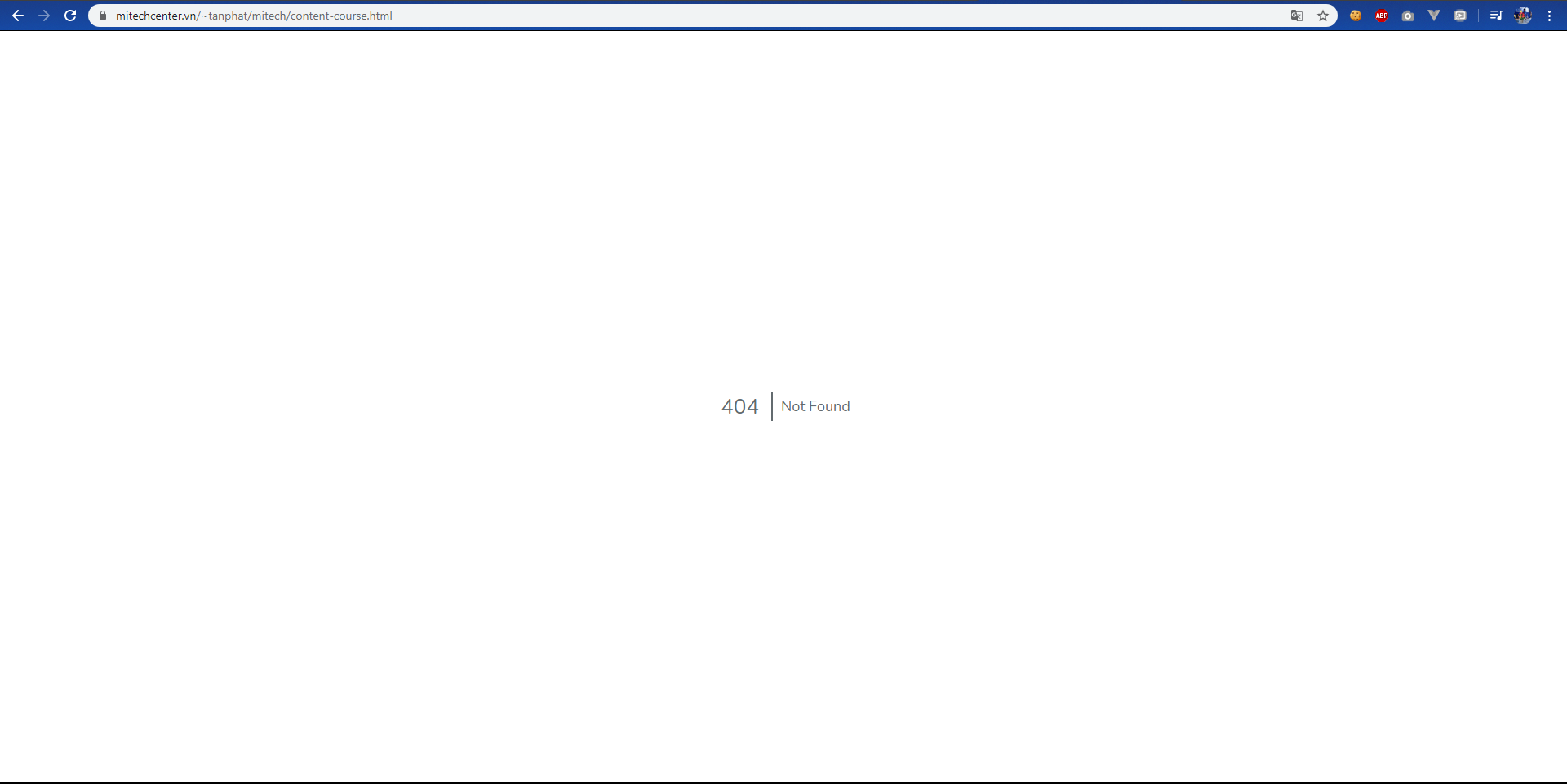


Hình 4.2. Carousel chỉ là các ảnh tỉnh, không nhấn được

- Menu danh mục khóa học khi xem bài viết dẫn đến liên kết không tồn tại.

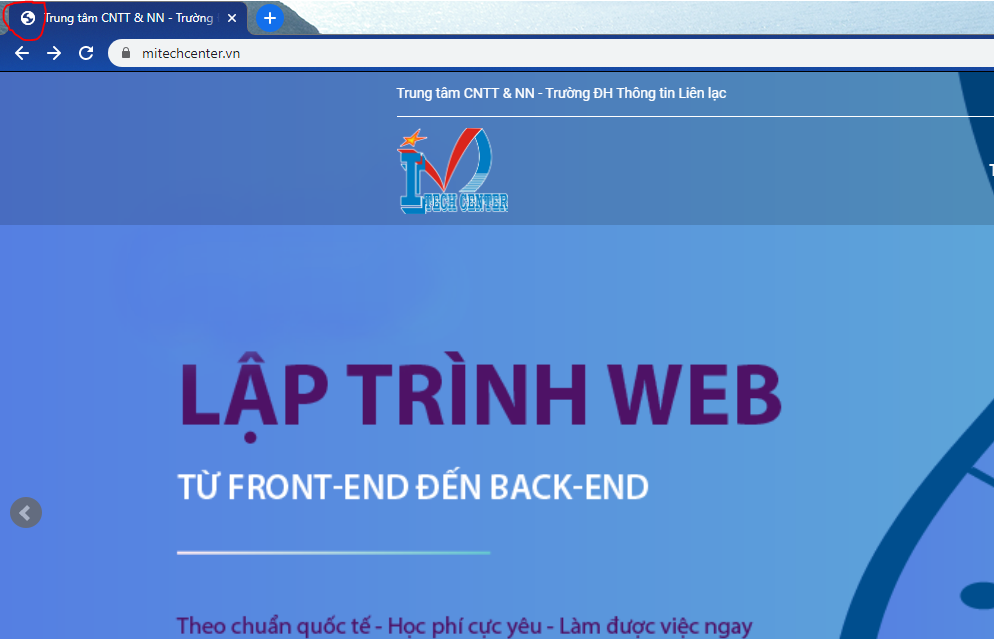


Hình 4.3. Danh mục khóa học khi nhấn vào các bài viết



Hình 4.4. Trang hiển thị lỗi khi nhấn vào đường dẫn bất kỳ tại mục trên

- Website vẫn chưa có favicon

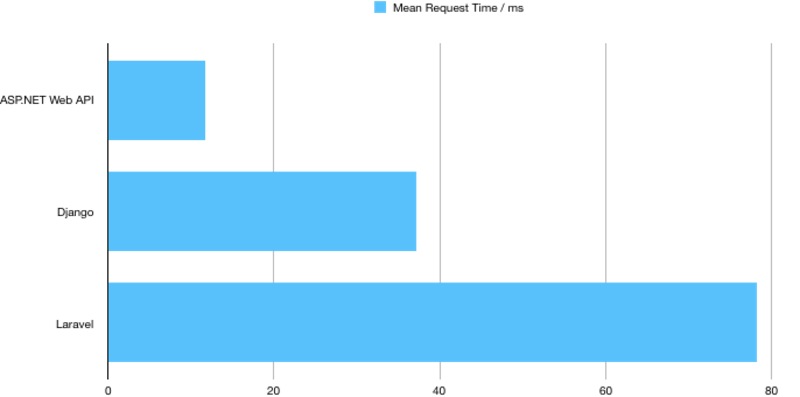


Hình 4.5. Website vẫn không có favicon

Ngoài ra, MitechCenter.vn hiện tại đang được đội ngũ phát triển của Mitech sử dụng Framework là Laravel để thực thi. Đây là một Framework tuyệt vời, hiện Laravel đang là một Framework miễn phí rất được ưa chuộng trên thị trường công nghệ hiện tại. Tuy nhiên, ASP.NET Core cũng không kém cạnh, để so sánh giữa hai công nghệ này, em sẽ đưa ra một số dẫn chứng như sau:

a) Tốc độ xử lý

Theo một bài kiểm tra đánh giá trên trang [ryshin.vn](https://www.ryshin.vn/), họ đã đặt ra một kịch bản sử dụng công cụ **Apache HTTP server benchmarking tool (ab)** để tạo ra tải mô phỏng, đây là mộ công cụ được phát triển bởi Apache Software Foudation, là công cụ tạo ta tải mô phỏng cho Web Server bằng cách gửi các request đồng thời tới máy chủ. Bên cạch đó, họ sử dụng Docker để kiểm nghiệm kết quả độc lập và tránh tình trạng bị ảnh hưởng bởi hệ điều hành. Kết quả thử nghiệm khi họ giả lập 100 người truy cập cùng thời điểm được thể hiện như hình sau:



Hình 4.6. Kết quả về tốc độ phản hồi (Nhỏ hơn là tốt hơn)

Từ đó có thể thấy ASP.NET Core có tốc độ phản hồi nhanh vượt trội, điều này có thể dễ hiểu bởi vì C# là một ngôn ngữ biên dịch còn hai ngôn ngữ còn lại kia là ngôn ngữ thông dịch.

b) Tính bảo mật

ASP.NET Core được xây dựng bởi Microsoft nên vấn đề bảo mật luôn được đặt lên hàng đầu. Các dữ liệu đầu vào luôn được kiểm tra và chuẩn hóa một cách mạnh mẽ nhất nhằm ngăn chặn các cuộc tấn công giả mạo yêu cầu giữa các trang web (CSRF). Còn đối với Laravel Framework, đối tượng nhắm đến chủ yếu là tập trung vào giao diện tương tác với khách hàng, cùng với nhiều lý do khác nên bảo mật của Laravel từ đó cũng kém hơn so với ASP.NET Core.

c) Chi phí

Nếu khi trước sử dụng các công nghệ do Microsoft cung cấp, chúng ta phải trả một khoản phí về bản quyền. Điều này khiến cho các doanh nghiệp e ngại nên dẫn đến sử dụng các Framework như Laravel. Tuy nhiên vào thời điểm này, với sự phát triển của xu hướng OpenSource, .NET Core/ASP.NET Core hiện đang là một mã nguồn mở hoàn toàn miễn phí với khả năng triển khai đa nền tảng, SQL Server có phiên bản Express cũng đang là một lựa chọn miễn phí khác đối với lượng dữ liệu vừa đủ. Chính vì vậy, chúng ta có thể dễ dàng ứng dụng ASP.NET Core để thay thế Laravel mà không cần lo về mặt chi phí khi sử dụng.

Từ những điều trên, cùng những khiếm khuyết chưa hoàn thiện đang tồn tại trong website MiechCenter.vn, việc thực hiện dự án của đề tài **“XÂY DỰNG WEBSITE MITECHCENTER.VN SỬ DỤNG ASP.NET CORE”** là điều vô cùng cần thiết và thực tế trong lúc này.

4.2. Đối tượng sử dụng

Với mục đích sử dụng là cung cấp từ phía quản lý cũng như truyền tải thông tin đến khách hàng hoặc học viên,… hay các đối tượng có nhu cầu tìm hiểu về Trung Tâm CNTT & NN – Trường ĐH Thông Tin Liên Lạc, nên website sẽ chia ra làm hai phần cho hai đối tượng, đó là:

- Quản trị viên: Cho phép quản lý website, đăng và truyền tải thông tin đến người dùng.

- Người dùng: Bao gồm khách hàng hoặc học viên,… hay các đối tượng có nhu cầu tìm hiểu về Trung Tâm CNTT & NN – Trường ĐH Thông Tin Liên Lạc, cho phép họ xem thông tin được truyền tải, thông tin liên hệ hoặc để lại lời nhắn,…

4.3. Hệ thống chức năng

4.3.1. Về phía người dùng

Xây dụng website có khả năng hỗ trợ người dùng ghé thăm với các yêu cầu sau đây:

- Website hỗ trợ responsive trên các thiết bị phổ biến của người dùng.

- Trang chủ có các mục: Danh sách dịch vụ, chương trình đào tạo, tin tức mới, đối tác, slide về các mục giới thiệu.

- Footer cần có: Thông tư về quyết định thành lập trung tâm, địa chỉ liên hệ, hộp toại hiển thị like fanpage.

- Cần có trang giới thiệu về Mitech Center

- Cần có trang liên hệ

- Trang danh sách các khóa học

- Danh mục tin tức

4.3.2. Về phía trang quản lý

Xây dựng được trang quản lý có các chức năng như sau:

- Chỉnh sửa được các thành phần cơ bản tại trang chủ: Carousel, các nội dung text

- Tạo các danh mục của tin tức và khóa học

- Thêm bài viết

- Thêm khóa học

- Thêm chương trình đào tạo

- Kiểm tra tin nhắn liên hệ từ khách hàng thông qua website

4.4. Kiến trúc và thiết kế ứng dụng website

4.4.1. Sơ đồ luồng dữ liệu

4.4.2. Sơ đồ quan hệ thực thể

4.4.3. Cấu trúc dữ liệu

4.5. Tiến hành xây dựng website với ASP.NET Core

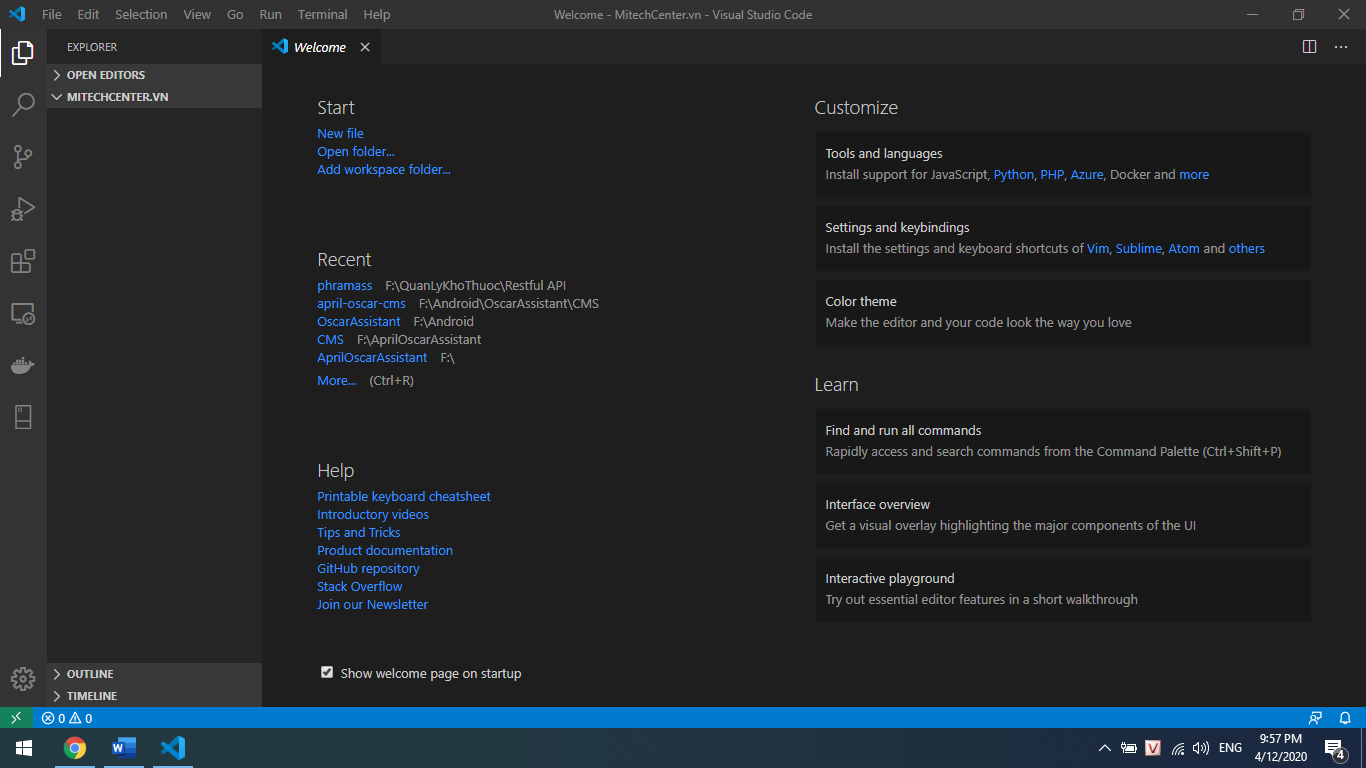
Vì tính gọn nhẹ và linh hoạt của Visual Studio Code (VSCode) rất thuận lợi cho việc phát triển ứng dụng, đặc biệt là trên những máy có cấu hình yếu như laptop của em, nên khi tiến hành xây dựng website này, em xin trình bày theo quy trình được triển khai trên IDE là VSCode.

4.5.1. Khởi tạo dự án website

a) Mở dự án website

- Bước đầu tiên, chúng ta cần mở VSCode lên.

- Tiếp theo, nhấn vào **“File”** > **“Open Folder”** từ thanh công cụ và chọn thư mục sẽ đặt dự án ở đó – cụ thể ở đây là thư mục **“F:\MitechCenter.vn”**, sau đó nhấn **“Select Folder”**.



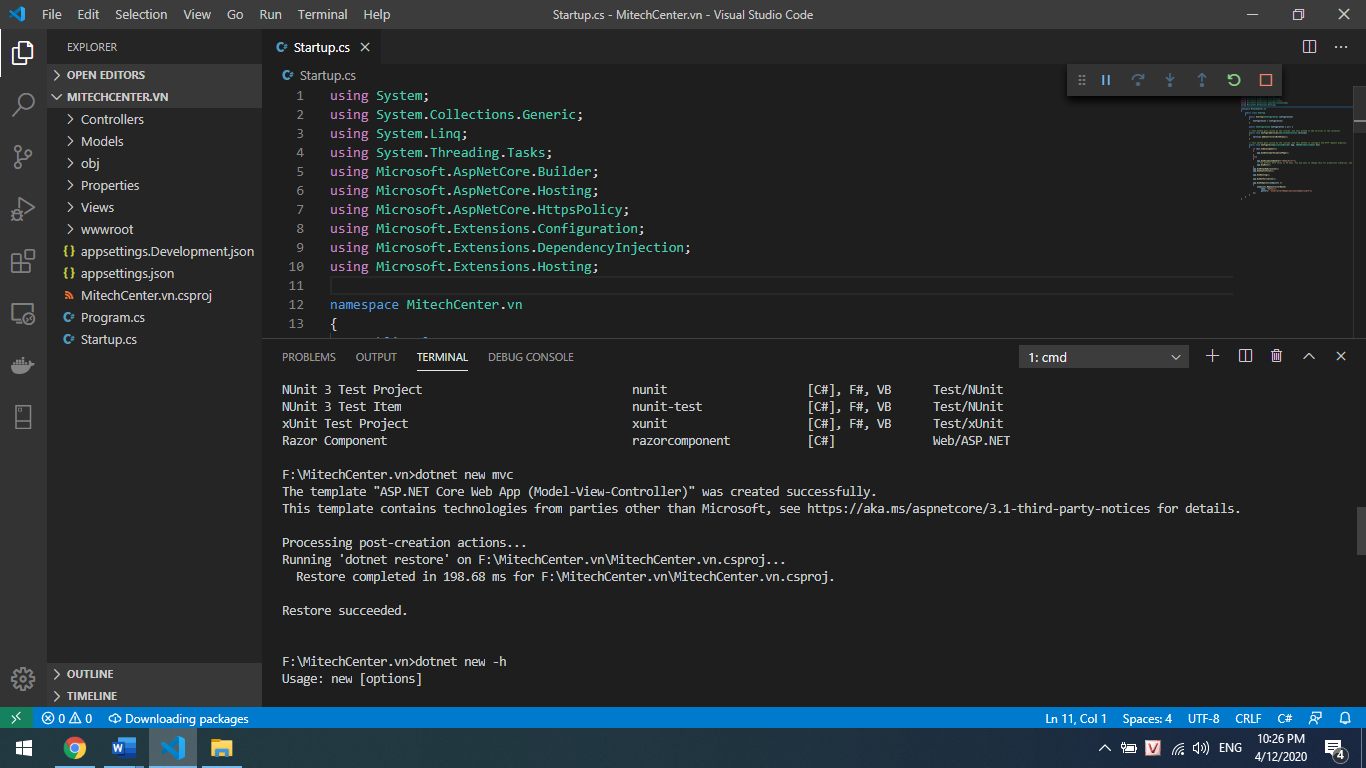
Hình 4.1. Hoàn thành tạo và mở thư mục chứa dự án

b) Khởi tạo dự án .NET Core

- Đầu tiên, cần mở cửa sổ lệnh bằng cách chọn **“Terminal”** > **“New Termial”**.

- Trong cửa sổ lệnh, tiến hành gõ **“dotnet new mvc”**, vì ở đây chúng ta sẽ tiến hành xây dựng website theo mô hình MVC sử dụng ASP.NET Core nên chúng ta sẽ sử dụng câu lệnh nêu trên. Ngoài ra còn có nhiều câu lệnh hữu ích khách để chúng ta có thể tạo dự án với các lựa chọn như ASP.NET Core with Angular, React.js với Redux hoặc không,… chi tiết có thể sử dụng lệnh **“dotnet new -h”**

- Khi câu lệnh hoàn thành, chúng ta sẽ có thư mục dự án bao gồm các tệp tin như hình bên dưới.



Hình 4.2. Sau khi hoàn tất khởi tạo dự án

- Chức năng của các thành phần chính trong dự án trên bao gồm:

+ **Startup.cs**: Startup class - class cấu hình đường ống request, nơi xử lý tất cả requests làm nên ứng dụng.

+ **Program.cs**: Class chứa hàm main khởi đầu của ứng dụng.

+ **MitechCenter.vn.csproj**: Tệp tin dự án định dạng cho những ứng dụng ASP.NET Core. Chứa Project references, Nuget references và các mục liên quan khác.

+ **application.json/appsettings.Development.json**: Chứa những cấu hình, thiết lập cơ bản cho ứng dụng.

+ **Views**: Chứa Razor views. Views là các thành phần để hiển thị giao diện người dùng. Nói chung, UI này hiển thị data model

+ **Controller**: Chứa các MVC controllers, khởi tạo với HomeController.cs. Controllers là các class để xử lý những yêu cầu của trình duyệt.

c) Chạy thử dự án sau khi khởi tạo

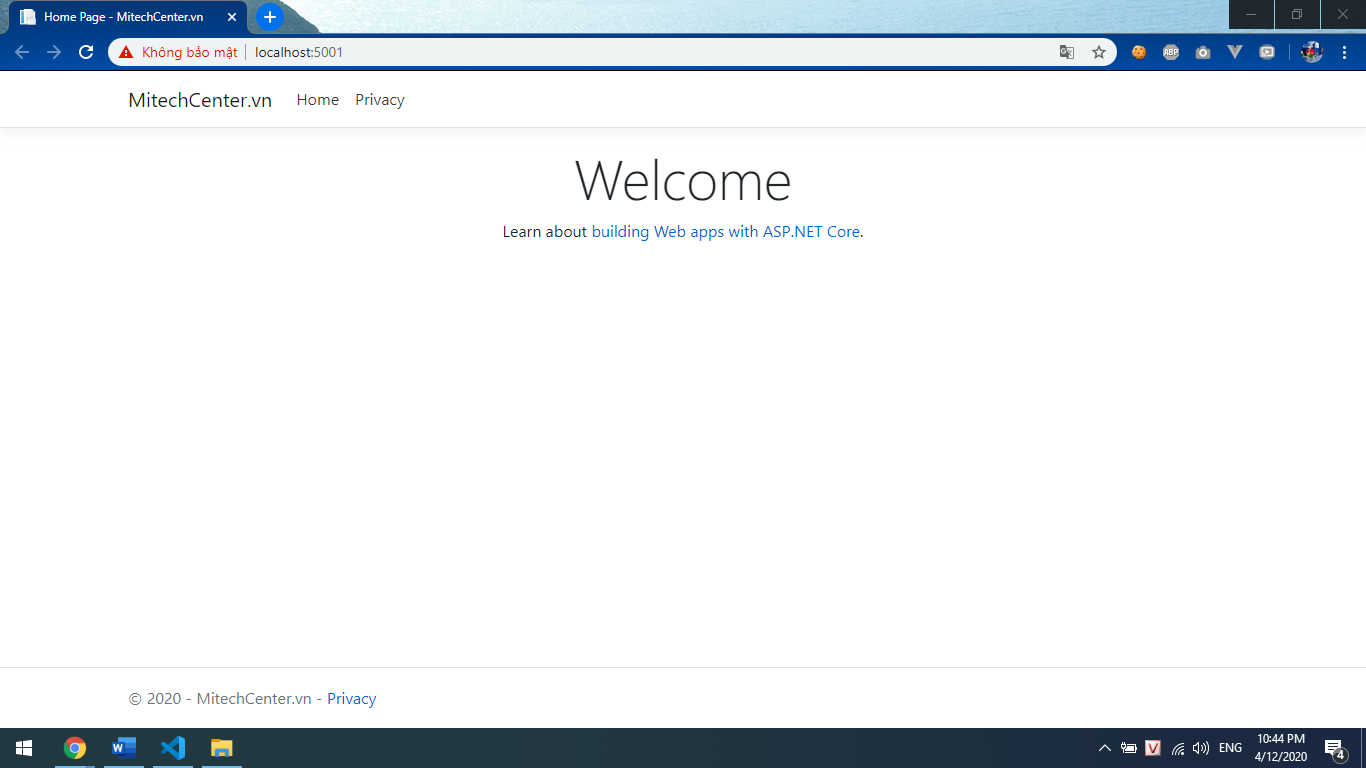
- Gõ lệnh **“dotnet dev-certs https --trust”** để tạo tin tưởng cho phép thực hiện hoàn chỉnh https trên môi trường phát triển. Nhấn chấp nhận **“Yes”** trong hộp thoại như hình dưới để hoàn tất:



Hình 4.3. Cài đặt chứng chỉ https cho môi trường phát triển

- Gõ lệnh **“dotnet run”** hoặc **“dotnet watch run”** và chờ dự án tiến hành build. Nếu sử dụng lệnh **“dotnet watch run”** ta cần thêm thuộc tính **“isBackground: true”** trong tệp **./.vscode/task.json** tại **label** có nội dung **watch**.

- Sau khi build hoàn tất, ta được kết quả như hình dưới.



Hình 4.4. Hoàn tất khởi tạo dự án

4.5.2. Tiến hành tạo trang

Để tạo một trang bất kỳ trong ASP.NET Core MVC, chúng ta lần lượt tiến hành như sau:

a) Tạo Controller

- Đầu tiên, nhấn chuột phải vào thư mục Controllers > New File, sau đó đặt tên controller mới theo quy tắc: ***<Tên controller mong muốn>*Controller.cs**

VD: Ở đây chúng ta cần tạo một trang đăng nhập, nên bước đầu tiên cần tạo controller cho trang đăng nhập này. Vì vậy controller sẽ có tên là:

**“LoginController.cs”**

- Sau đó tiến hành dán đoạn mã sau đây vào để hoàn tất việc tạo controller:

using Microsoft.AspNetCore.Mvc;

namespace SampleMVCApps.Controllers

{

    public class BaseController : Controller

    {

        public IActionResult Index()

        {

            return View();

        }

    }

}

Hình 4.5. Đoạn mã mặc định trong Controller bất kỳ

- Đoạn mã trên cho phép controller tự động trả về View tương ứng khi người dùng vào đường dẫn tham chiếu đến controller này.

b) Tạo View cho trang

- Trước tiên để tạo view cho trang, ta cần tạo thư mục là tên của trang cần tạo trong thư mục **Views**, bằng cách nhấn chuột phải vào thư mục **Views** > **New Folder**. *Lưu ý: Tên thư mục phải trùng với tên của* ***Controller*** *vừa tạo cho trang.*

VD: Ở đây chúng ta sẽ tạo thư mục có tên là **Login**.

- Tiếp theo chúng ta tạo tệp tin **“index.cshtml”** trong thư mục vừa mới tạo. Sau đó tiến hành dán đoạn mã CSHTML sau vào để thực thi thử:

@{

    ViewData["Title"] = "Login Page";

}

<div class="text-center">

    <h1 class="display-4">Xin chào</h1>

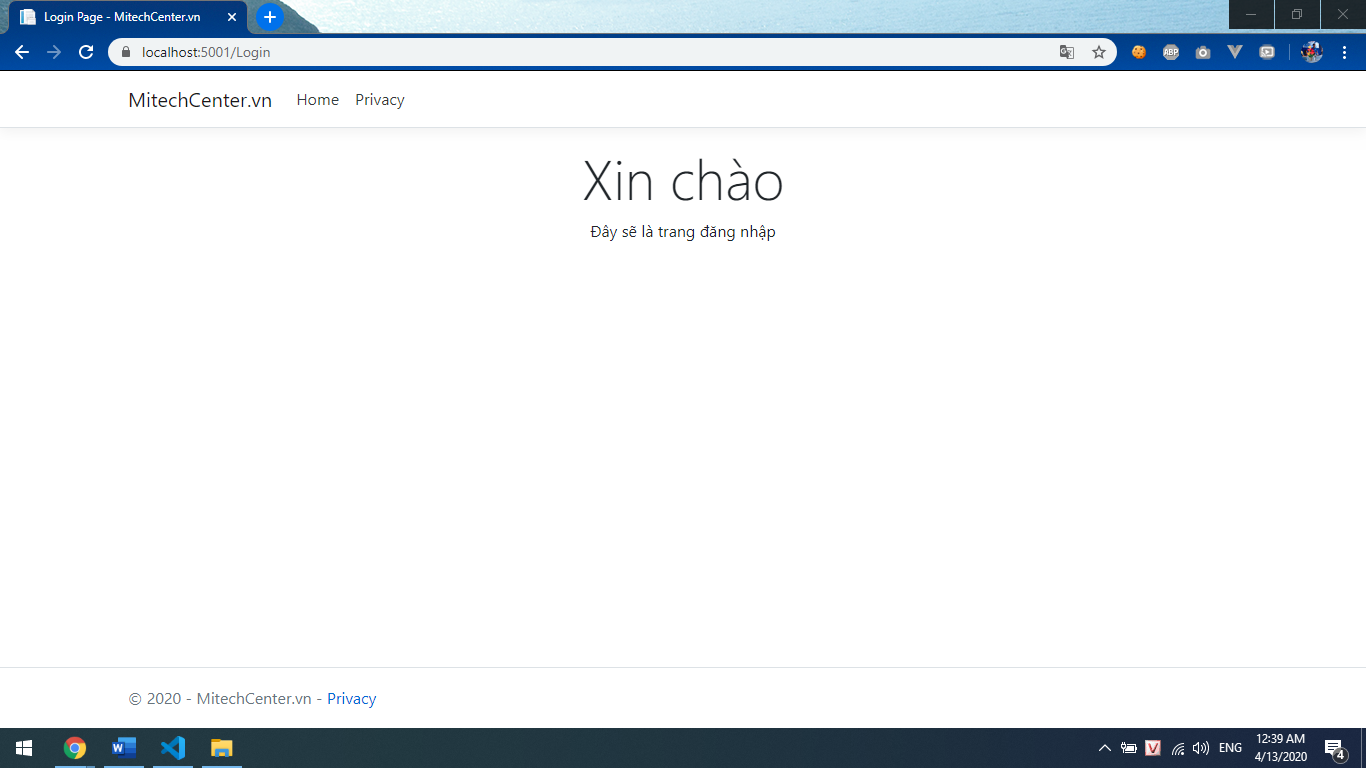
    <p>Đây sẽ là trang đăng nhập</p>

</div>

Hình 4.6. Mã nội dung của trang đăng nhập (Login)

c) Khởi chạy và kiểm tra

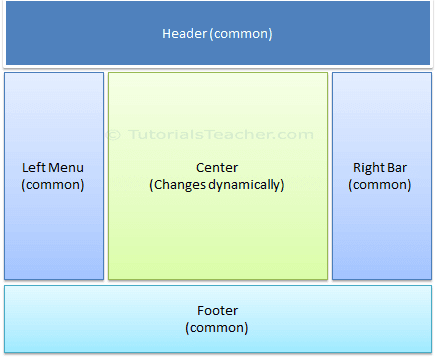
Tiến hành khởi chạy bằng lệnh “dotnet watch run” sau đó vào theo đường dẫn [**https://localhost:50001/Login**](https://localhost:50001/Login) ta thu được kết quả như hình sau:



Hình 4.7. Kết quả thu được sau khi hình thành trang đăng nhập

4.5.3. Tùy chỉnh layout

Trong một website có thể chứa các thành phần chung trong các giao diện người dùng mà vẫn giữ nguyên trong suốt ứng dụng chẳng hạn như logo, tiêu đề, thanh điều hướng bên trái, thanh điều hướng bên phải hoặc phần footer – đó được gọi là Layout của một website. Chúng ta có thể hình dung như sau:

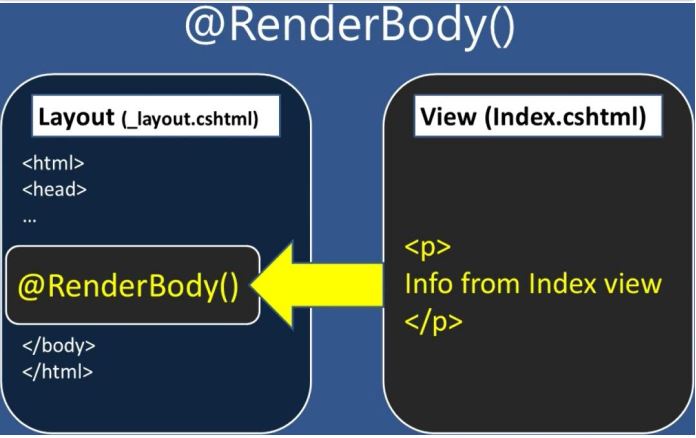


Hình 4.8. Mô hình cho việc hình dung layout trong một website

a) Layout trong ASP.NET Core

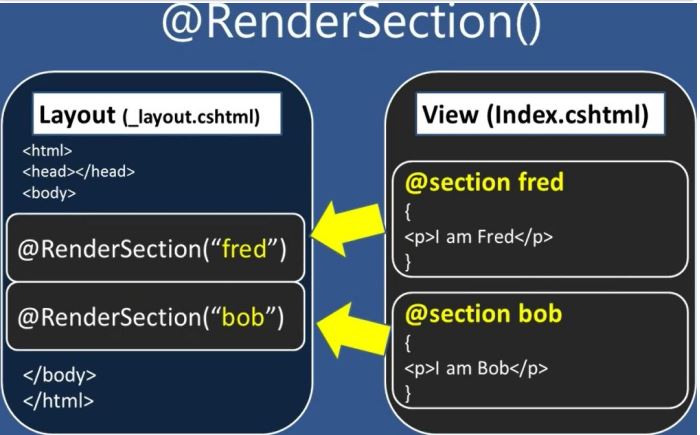
Như đã tìm hiểu ở trên, Views trong ASP.NET Core được tạo ra từ tệp tin có phần mở rộng là **\*.cshtml** được đặt trong thư mục **Views**. Cũng như những website thông thường khác, để giữ cho việc đồng nhất giữa các trang thi tạo website với ASP.NET Core, chúng ta cũng cần thêm header, footer và thành điều hướng,… ở tất cả các view. Tuy nhiên điều này thường gặp nhiều khó khăn và dễ dẫn đến sai sót, đặc biệt là khi dự án của chúng ta có số lượng view quá lớn. Tuy nhiên, trong ASP.NET Core đã định nghĩa sẵn tệp tin **\_Layout.cshtml** trong thư mục **Shared** là con của thư mục **Views** để giúp chúng ta trong công việc thống nhất layout này. Theo mặc định, các view mới được tạo ra trong thư mục Views sẽ kế thừa layout từ tệp tin **\_Layout.cshtml**. Trong tệp tin layout đó, chúng ta cần chúng ta cần chú ý đến các phương thức sau:

- **RenderBody()**: Phương thức chứa tất cả nội dung của những view kế thừa từ \_Layout.cshtml.



Hình 4.9. Cách hoạt động của phương thức RenderBody()

- **RenderSection()**: Layout có thể có hoặc không có phương thức này để giữ chỗ cho những thành phần khác được đánh dấu thông qua khối **“@selection <tên của selection> {}”** trong view kế thừa từ layout đó.



Hình 4.10. Cách hoạt động của phương thức RenderSection()

a) Trang không sử dụng layout mặc định

Với ý định thiết kế trang đăng nhập là một trang tách biệt hoàn toàn so với các layout còn lại của website, nên ở đây chúng ta tiến hành không sử dụng layout mặc định cho trang đăng nhập (Login). Vì vậy, ta tiến hành tạo mới một layout dành riêng cho việc xác thực này. Sau đó thực hiện theo đoạn mã như bên dưới đây ở đầu tệp tin \*.cshtml của trang mong muốn để thay đổi từ layout mặc định sang layout mới.VD:

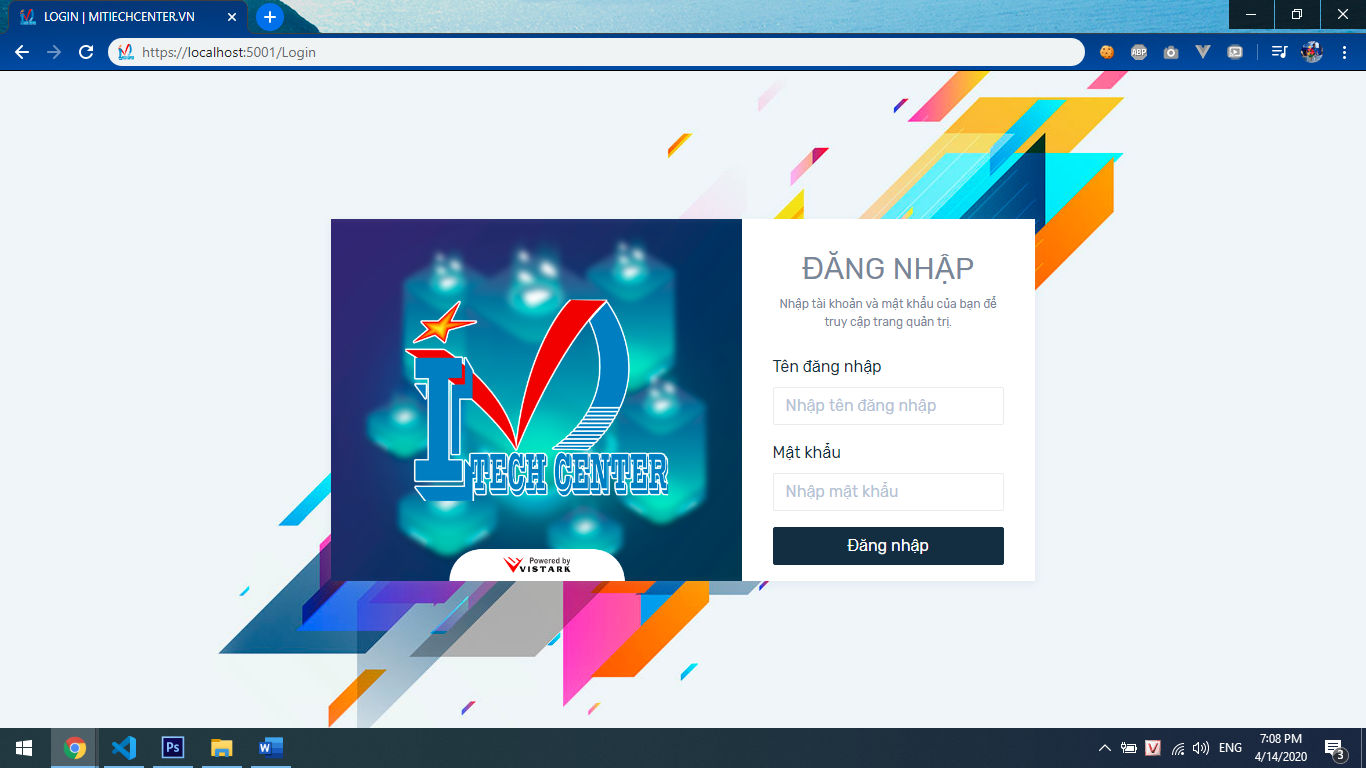
@{

    Layout = "~/Views/Shared/AuthenticationLayout.cshtml";

}

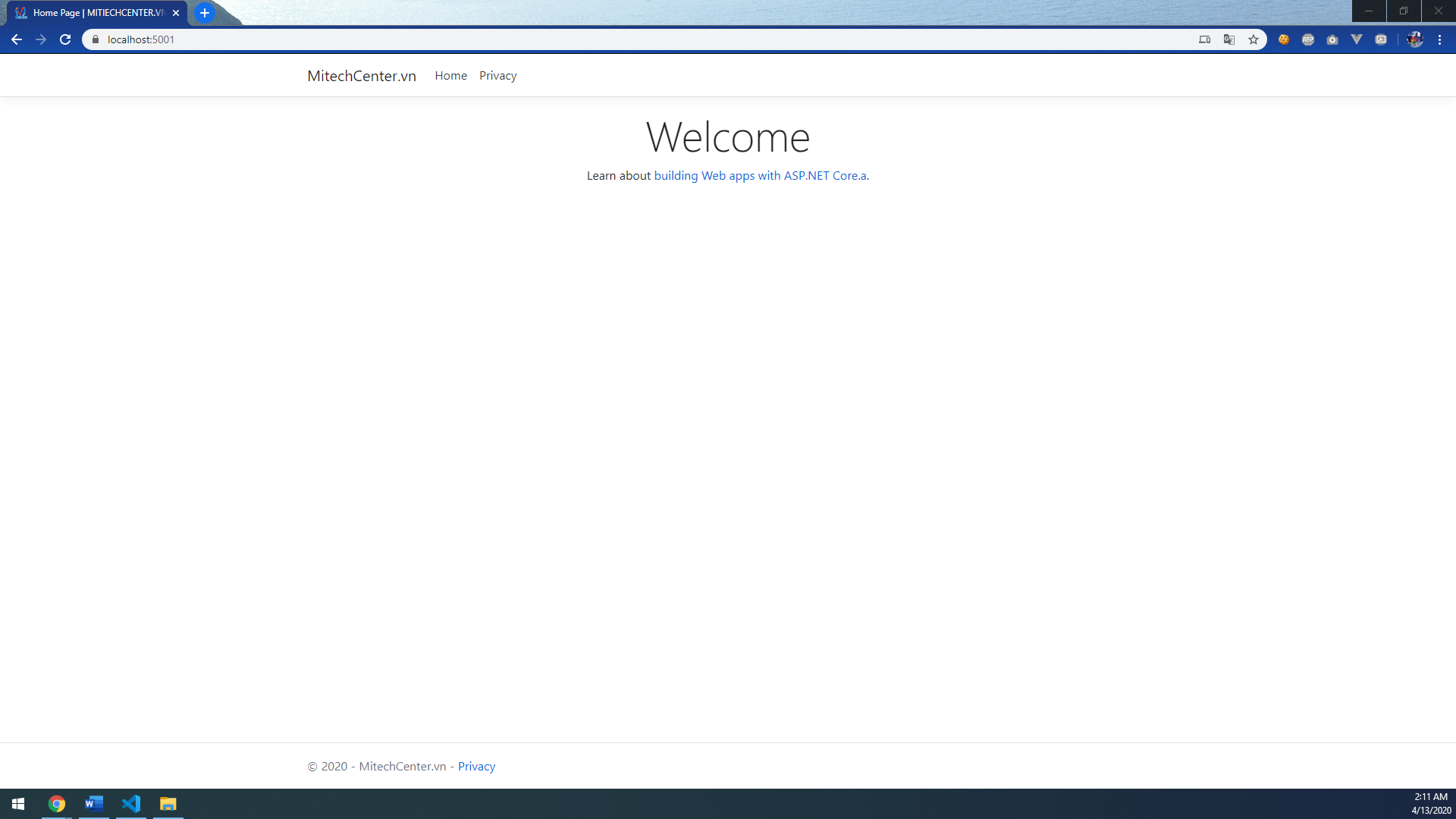
Hình 4.10. Thay đổi layout cho view

Kết quả chúng ta thu được như những hình sau:



Hình 4.11. Trang đăng nhập khi dùng Layout mới

<https://www.adminmart.com/src/html/index.html>



Hình 4.12. Trang chủ khi vẫn dùng Layout mặc định

KẾT LUẬN

ASP.NET Core là mô hình lập trình mới có thể triển khai trên đám mây và có mục đích sử dụng chạy nhiều nền tảng như: MacOS, Linux và Windows. Nó hỗ trợ được ASP.NET MVC, ASP.NET Core Web API,… và con là một mã nguồn mở nhưng vô cùng bảo mật và bền vững với hiệu năng xử lý cao khi được tối ưu tốt mã nguồn, đây đang là một xu hướng công nghệ cho phát triển phần mềm cho tương lai

I. KẾT QUẢ ĐẠT ĐƯỢC

II. KẾT QUẢ CHƯA ĐẠT ĐƯỢC

III. ĐÁNH GIÁ

1. Ưu điểm

2. Nhược điểm

IV. ĐỊNH HƯỚNG PHÁT TRIỂN

V. KIẾN NGHỊ

**TÀI LIỆU THAM KHẢO**

1. <https://thuthuatmarketing.com/digital-marketing/website/tai-sao-doanh-nghiep-can-phai-co-website>, Website

2. [https://sinhvientot.net/asp-net-core-buoc-dot-pha-trong-cong-nghe-net](https://sinhvientot.net/asp-net-core-buoc-dot-pha-trong-cong-nghe-net/), Website

3. <https://docs.microsoft.com/en-us/sql/ssms/sql-server-management-studio-ssms?view=sql-server-ver15>, Website

4. <https://www.adcvietnam.net/tai-sao-nen-chon-thiet-ke-website-bang-ngon-ngu-aspnet-tai-adc>, Website

5. <https://tedu.com.vn/lap-trinh-aspnet-core/xay-dung-ung-dung-aspnet-core-mvc-dau-tien-225.html>, Website

6. <https://topdev.vn/blog/asp-net-core-la-gi/>, Website

7. <https://tedu.com.vn/lap-trinh-aspnet-core/gioi-thieu-ve-aspnet-core-203.html>, Website

8. <https://sinhvientot.net/khi-nao-nen-su-dung-net-core-va-net-framework/>, Website

9. <https://kb.pavietnam.vn/so-sanh-giua-net-framework-va-net-core.html>, Website

10. <https://monamedia.co/mvc-la-gi-ung-dung-cua-mo-hinh-mvc-trong-lap-trinh/>, Website

11. <https://sinhvientot.net/asp-net-core-mvc-tong-quan-ve-asp-net-core-mvc/>, Website

12. <https://tedu.com.vn/lap-trinh-aspnet-core/cai-dat-va-cau-hinh-moi-truong-phat-trien-aspnet-core-204.html>, Website

13. <https://tuhocict.com/cai-dat-moi-truong-va-cong-cu-cho-aspnet-core/>, Website

**KẾ HOẠCH THỰC HIỆN**

|  |  |  |
| --- | --- | --- |
| **STT** | **Nội dung** | **Thời gian thực hiện** |
| 1 | Chương 1: PHÂN BIỆT .NET CORE VÀ .NET FRAMEWORK | 10 – 12/4/2020 |
| 2 | Chương 2: GIỚI THIỆU VỀ ASP.NET CORE | 13 – 14/4/2020 |
| 3 | Chương 3: CÀI ĐẶT MÔI TRƯỜNG PHÁT TRIỂN | 15 – 16/4/2020 |
| 4 | Chương 4: XÂY DỰNG WEBSITE | 17 – 25/4/2020 |
| 5 | Kiểm thử và thử nghiệm | 26 – 30/4/2020 |
| 6 | Hoàn thành đề tài thực tập tại cơ sở | 5/2020 |

**NHẬN XÉT CỦA CÁN BỘ HƯỚNG DẪN**

|  |  |
| --- | --- |
| **Người hướng dẫn khoa học** | *Khánh Hòa, ngày…tháng …năm 2020*  **Người thực hiện** |
| *(Chữ ký, họ và tên)* | *(Chữ ký, họ và tên)* |
| **ThS. Trần Thị Mỹ Hiền** | **Nguyễn Trọng Nghĩa** |