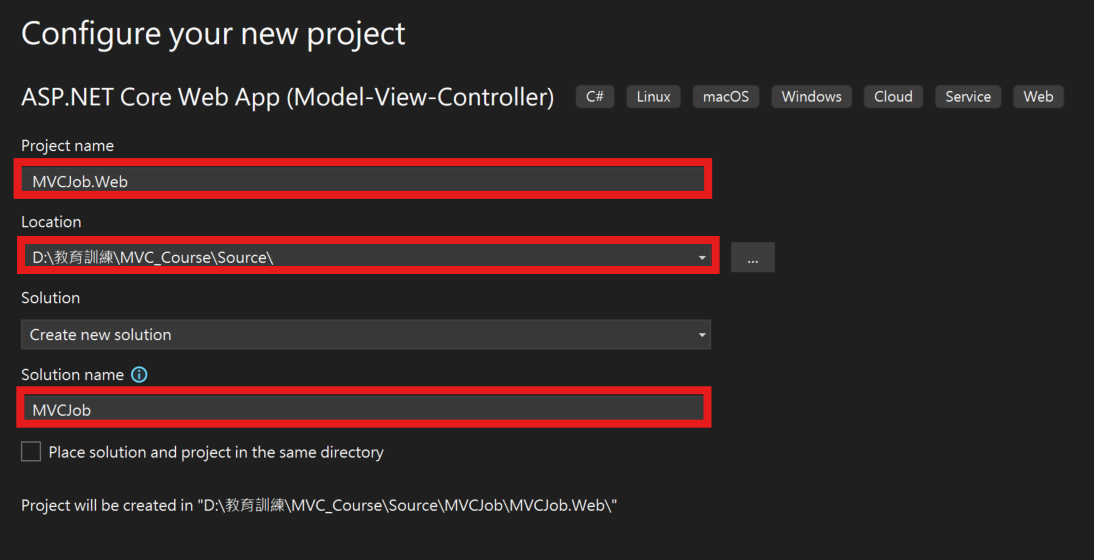
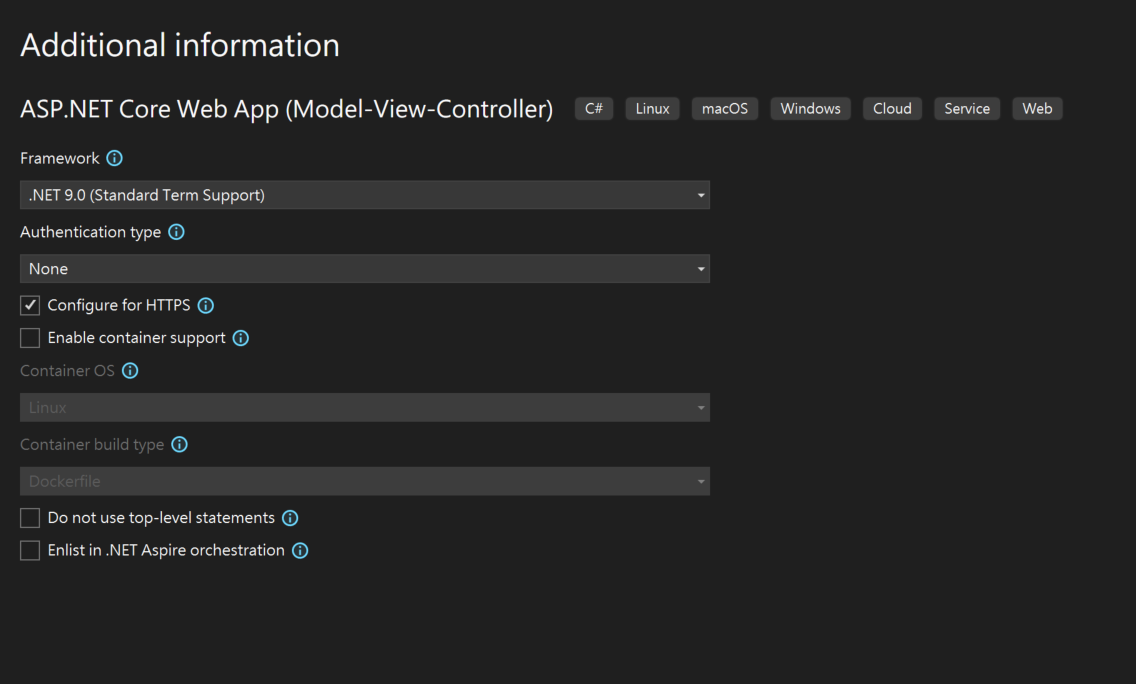
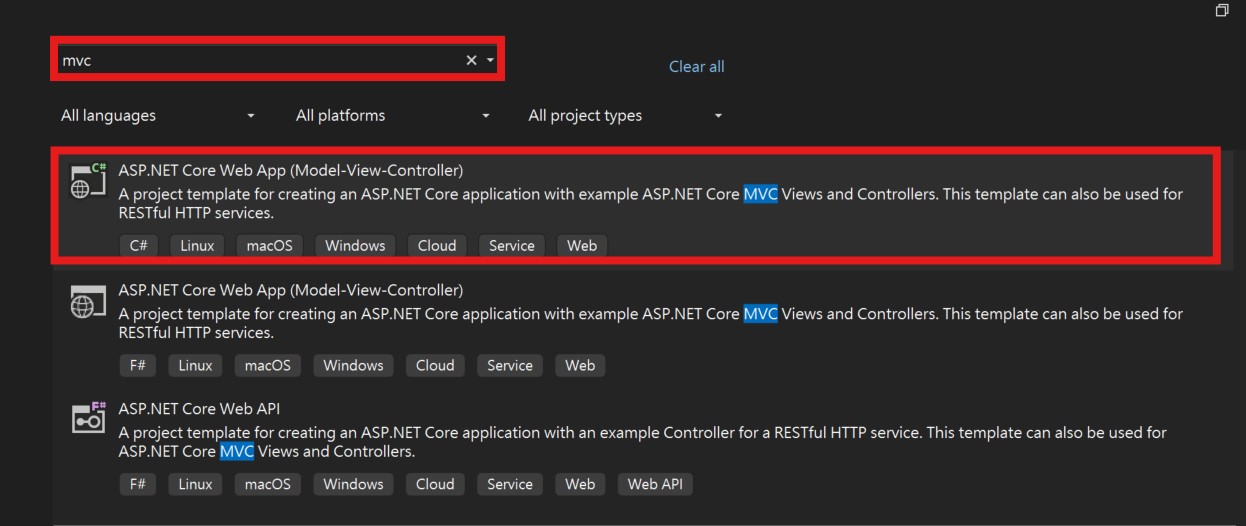
1. 進入Visual Studio 2022=>建立MVC專案: MVCJob.Web





在方案總管=>MVCJob.Web上Double Click

<Project Sdk="Microsoft.NET.Sdk.Web">

<PropertyGroup>

<TargetFramework>net9.0</TargetFramework>

<Nullable>enable</Nullable>

<ImplicitUsings>enable</ImplicitUsings>

</PropertyGroup>

</Project>

可以看到這個專案的屬性, 整個專案的相關屬性在  
Properties=>launchSetting.json

{

"$schema": "https://json.schemastore.org/launchsettings.json",

"profiles": {

"http": {

"commandName": "Project",

"dotnetRunMessages": true,

"launchBrowser": true,

"applicationUrl": "http://localhost:5120",

"environmentVariables": {

"ASPNETCORE\_ENVIRONMENT": "Development"

}

},

"https": {

"commandName": "Project",

"dotnetRunMessages": true,

"launchBrowser": true,

"applicationUrl": "https://localhost:7081;http://localhost:5120",

"environmentVariables": {

"ASPNETCORE\_ENVIRONMENT": "Development"

}

}

}

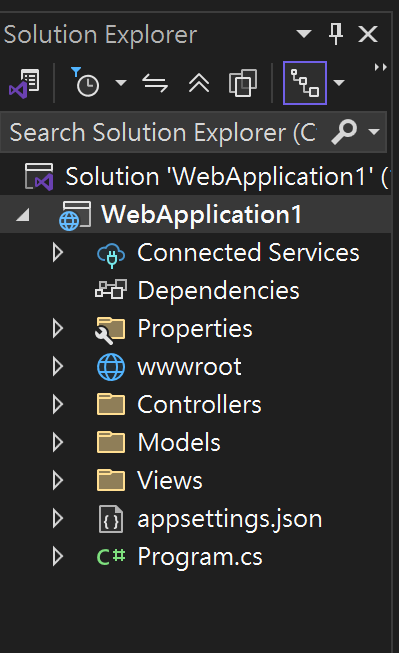
}

其中

"https": {  
 ……

}

這個區塊是在執行https的環境參數



wwwroot: 網頁前端相關檔案(css, javascript, lib)  
Controllers: Controller的類別(HomeController.cs)  
Models: Model的類別(ErrorViewModel.cs)  
View: View的類別(Home, Shared兩個Folder)

在Model上右鍵, Add=>New Item=>Class: Category.cs

namespace MVCJob.Web.Models;

public class Category

{

[Key]

public int Id { get; set; }

[Required]

public string Name { get; set; }

public int DisplayOrder { get; set; }

}

\*注意: 鍵入[Key], 最上方會自動加上

namespace MVCJob.Web.Models;

這一行, 在 appsettings.json加入紅色標示的資料  
{

"Logging": {

"LogLevel": {

"Default": "Information",

"Microsoft.AspNetCore": "Warning"

}

},

"AllowedHosts": "\*",

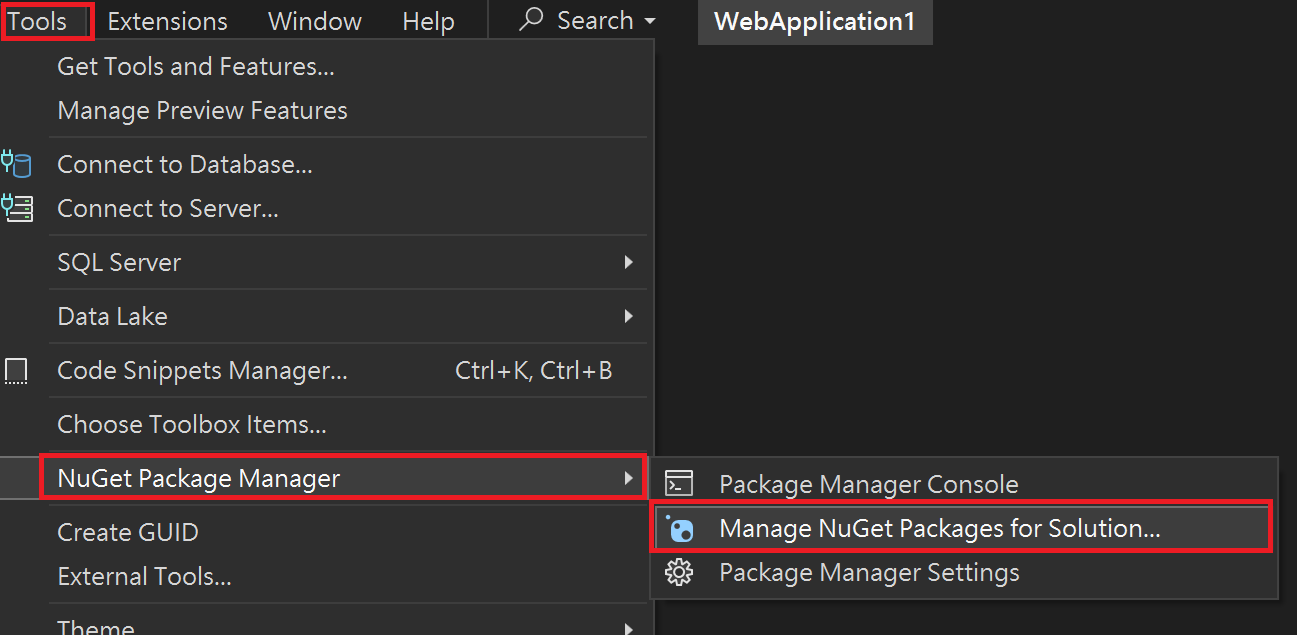
"ConnectionStrings": {

"DefaultConnection": "Server=.;Database=MVCPracticeDB;Trusted\_Connection=True;TrustServerCertificate=True;"

}

}

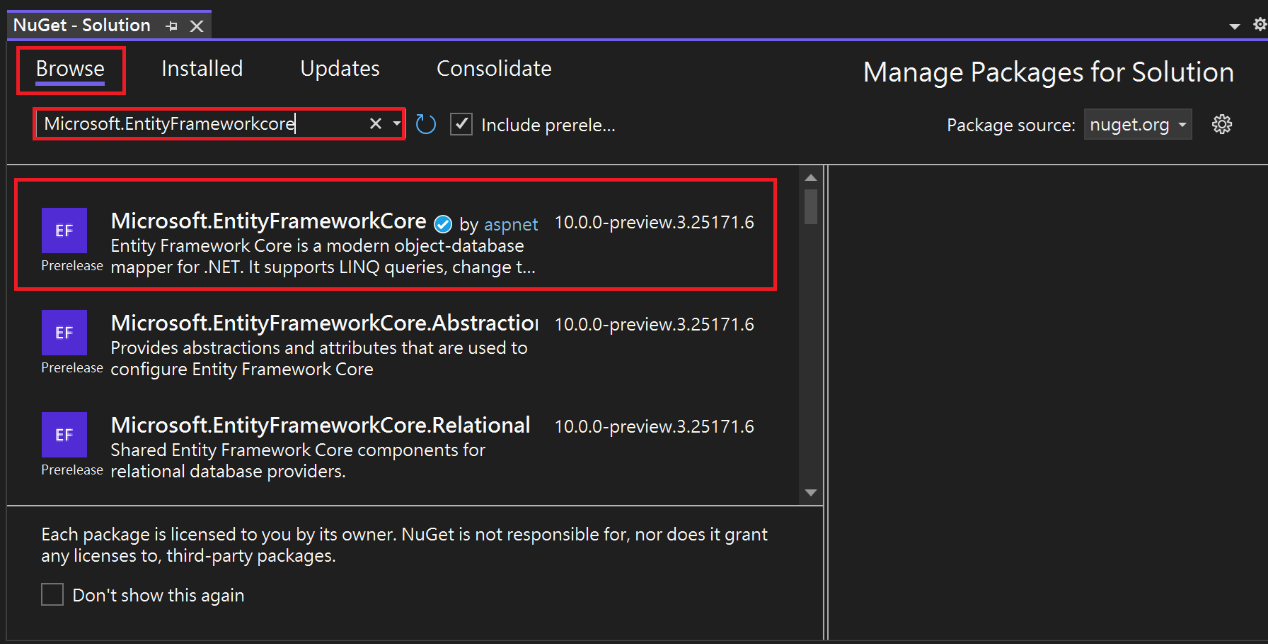
Tools => NuGet Package Manager => Manager NuGet Package for Solution…



依序加入  
  
Microsoft.EntityFrameworkCore

Microsoft.EntityFrameworkCore.SqlServer

Microsoft.EntityFrameworkCore.Tools



展開Solution Explorer, 在MVCJob.Web上右鍵, Add => New Folder: **Data**

在Data資料夾上右鍵=> Add => Class: **ApplicationDbContext  
加入藍色部份的程式碼, 紅色部份程式碼系統會自動生成**

using Microsoft.EntityFrameworkCore;

namespace MVCJob.Web.Data

{

**public class ApplicationDbContext:DbContext**

**{**

**public ApplicationDbContext( DbContextOptions<ApplicationDbContext> options):   
 base(options)**

**{**

**}**

}

}

在program.cs(第8行位置), 加入下面藍色標示

var builder = WebApplication.CreateBuilder(args);

// Add services to the container.

builder.Services.AddControllersWithViews();

**builder.Services.AddDbContext<ApplicationDbContext>(option => option.UseSqlServer(builder.Configuration.GetConnectionString("DefaultConnection")));**

var app = builder.Build();

// Configure the HTTP request pipeline.

if (!app.Environment.IsDevelopment())

{

app.UseExceptionHandler("/Home/Error");

// The default HSTS value is 30 days. You may want to change this for production scenarios, see https://aka.ms/aspnetcore-hsts.

app.UseHsts();

}

app.UseHttpsRedirection();

app.UseRouting();

app.UseAuthorization();

app.MapStaticAssets();

app.MapControllerRoute(

name: "default",

pattern: "{controller=Home}/{action=Index}/{id?}")

.WithStaticAssets();

app.Run();

功能表: Tools => NuGet Package Manager => Package Manager Console

一張含有 文字, 螢幕擷取畫面, 軟體, 多媒體軟體 的圖片

AI 產生的內容可能不正確。

PM> **update-database**

鍵入: update-database

會在SQL Server中產生資料庫

修改ApplicationDbContext.cs

public class ApplicationDbContext:DbContext

{

public ApplicationDbContext(DbContextOptions<ApplicationDbContext> options):base(options)

{

}

**public DbSet<Category> Categories { get; set; }**

}

NuGet鍵入:

PM> **add-migration AddCategoryTableToDB**PM> **update-database**

修改ApplicationDbContext.cs加入下列程式碼

public class ApplicationDbContext:DbContext

{

public ApplicationDbContext(DbContextOptions<ApplicationDbContext> options):base(options)

{

}

public DbSet<Category> categories { get; set; }

**protected override void OnModelCreating(ModelBuilder modelBuilder)**

**{**

**modelBuilder.Entity<Category>().HasData(**

**new Category { Id = 1, Name = "Action", DisplayOrder = 1 },**

**new Category { Id = 2, Name = "SciFi", DisplayOrder = 2 },**

**new Category { Id = 3, Name = "History", DisplayOrder = 3 }**

**);**

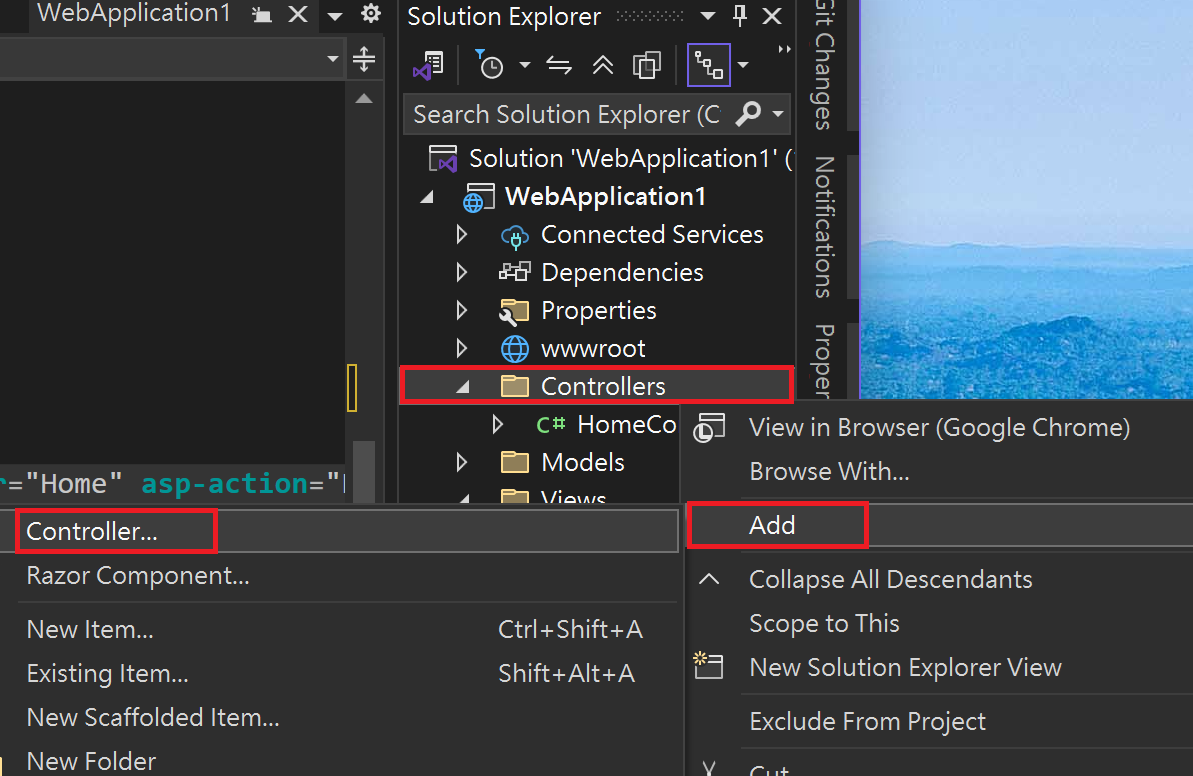
**}**

}

NuGet鍵入:

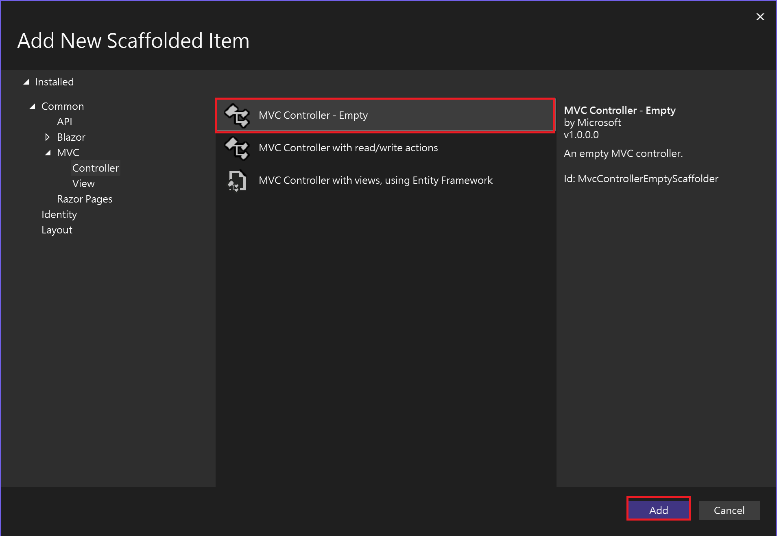
PM> **add-migration SeedCategoryTable**PM> **update-database**

加入Controller: Category.cs

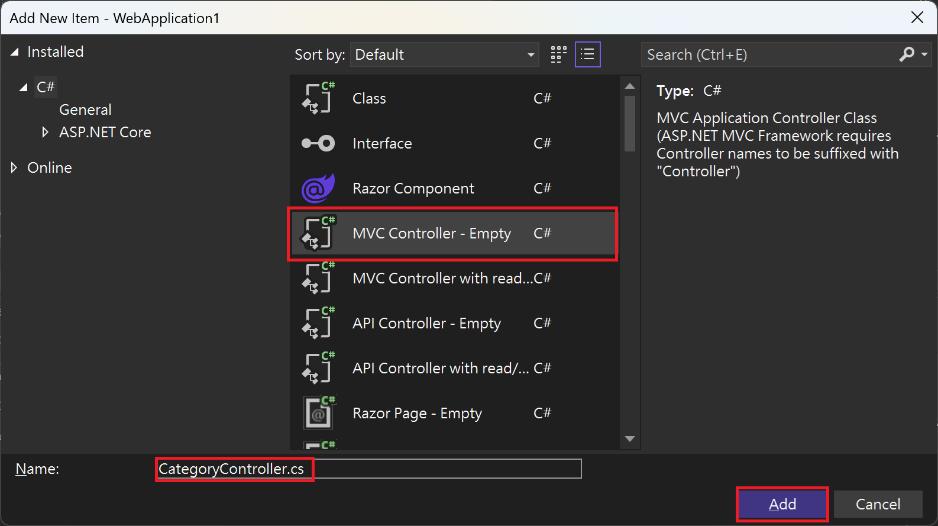


在

Controllers上右鍵 => Add => Controller…



1. 選擇MVC Controller-Empty按Name: CategoryController.cs按Add  
   (Controller必須是xxxxController)

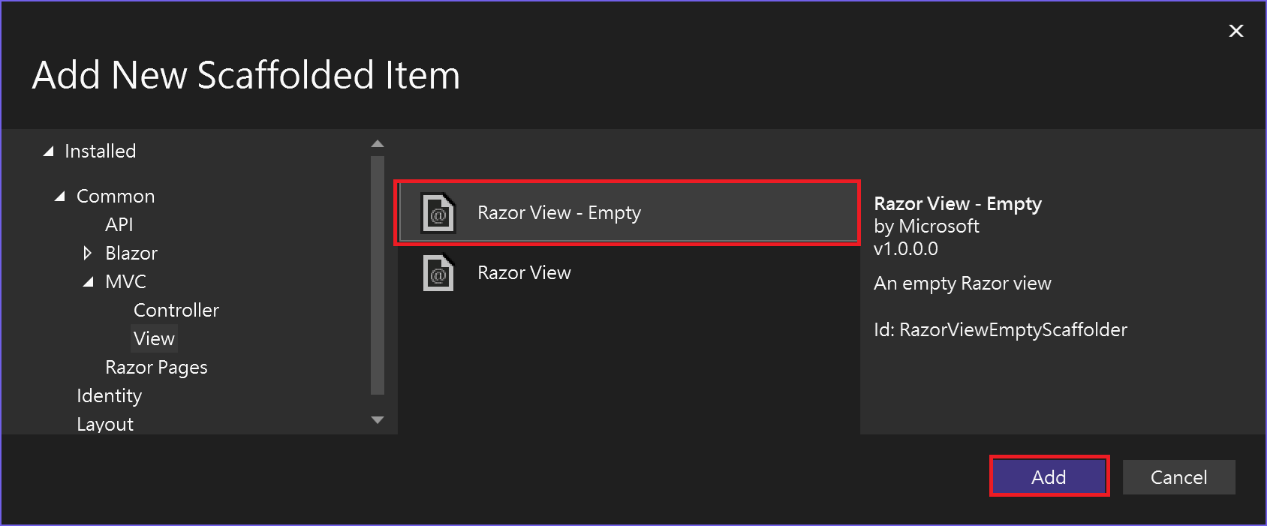


進入CategoryController.cs

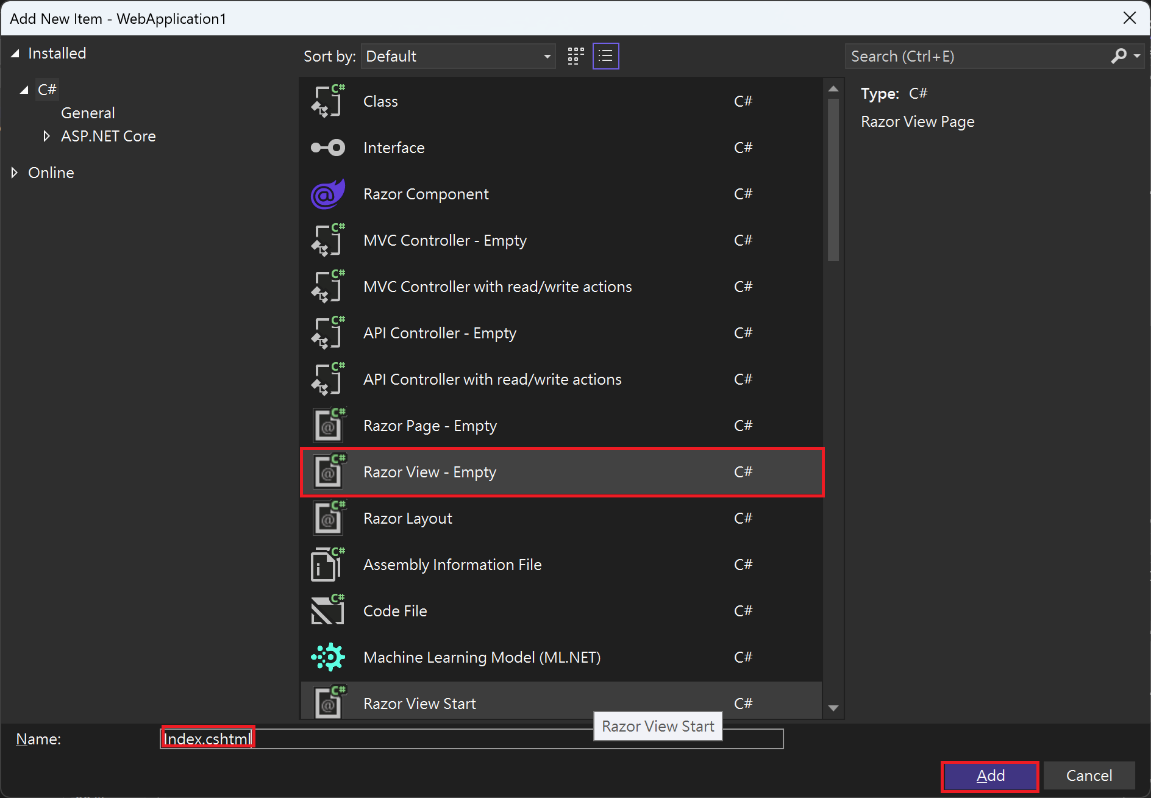
在程式碼: public IActionResult Index() 按右鍵 => Add View…

一張含有 文字, 螢幕擷取畫面, 字型 的圖片

AI 產生的內容可能不正確。

選擇 Razor View-Empty => Add

選擇預設的Razor View -Empty=> Name: Index.cshtml => Add



將Index.cshtml改為

<h1>Category List</h1>

如下圖所示  
一張含有 文字, 螢幕擷取畫面, 軟體, 陳列 的圖片

AI 產生的內容可能不正確。

Shared\\_Layout.cshtml

加入以下的程式碼

<li class="nav-item">

<**a** class="nav-link text-dark" **asp-area**="" **asp-controller**="Home" **asp-action**="Index">Home</**a**>

</li>

<li class="nav-item">

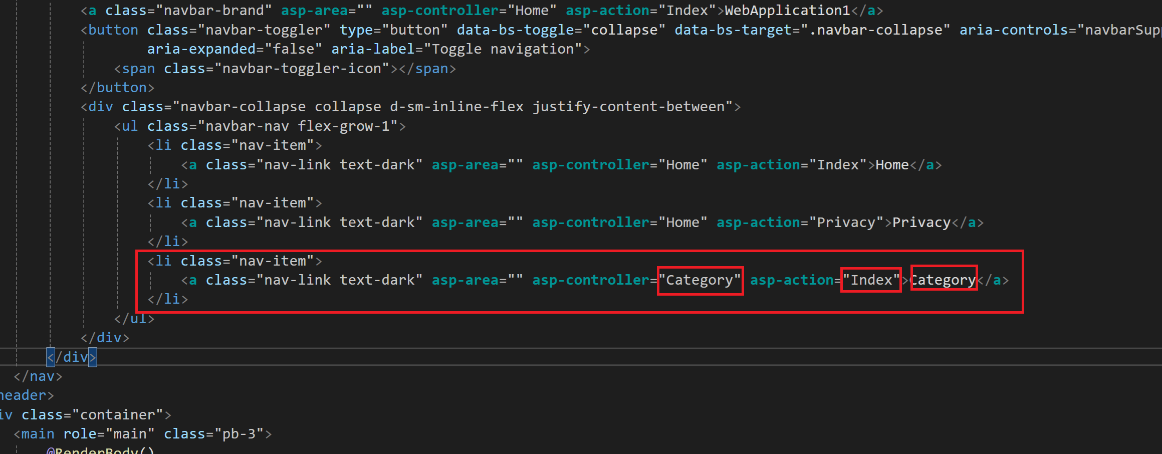
<**a** class="nav-link text-dark" **asp-area**="" **asp-controller**="Home" **asp-action**="Privacy">Privacy</**a**>

</li>

<li class="nav-item">

<**a** class="nav-link text-dark" **asp-area**="" **asp-controller**="Category" **asp-action**="Index">Category</**a**>

</li>



將整塊  
<li class=”nav-item”>  
….  
</li>  
複製貼上, 修改上圖的三個屬性

執行看看

修改CategoryController.cs

using Microsoft.AspNetCore.Mvc;

using MVCJob.Web.Data;

using MVCJob.Web.Models;

namespace MVCJob.Web.Controllers

{

public class CategoryController : Controller

{

**private readonly ApplicationDbContext \_db;**

**public CategoryController(ApplicationDbContext db)**

**{**

**\_db = db;**

**}**

**public IActionResult Index()**

**{**

**List<Category> objCategoryList = \_db.Categories.ToList();**

**return View(objCategoryList);**

**}**

}

}

修改 Category\Index.cshtml

**@model List<Category>**

**<h1>Category List</h1>**

**<table class="table table-border table-striped">**

**<thead>**

**<tr>**

**<th>Category ID</th>**

**<th>Category Name</th>**

**<th>Display Order</th>**

**</tr>**

**</thead>**

**<tbody>**

**@foreach (var obj in Model)**

**{**

**<tr>**

**<td>@obj.Id</td>**

**<td>@obj.Name</td>**

**<td>@obj.DisplayOrder</td>**

**</tr>**

**}**

**</tbody>**

**</table>**