C# Web

PRO/PRW - C# Web 1

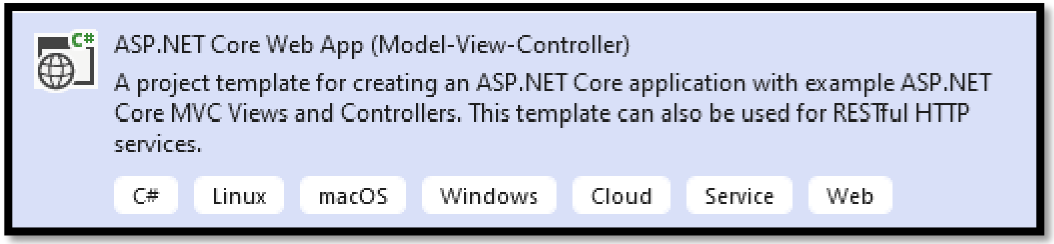
DE HOGESCHOOL
MET HET NETWERK

Hogeschool PXL – Elfde-Liniestraat 24 – B-3500 Hasselt
www.pxl.be - www.pxl.be/facebook

Doel

- MVC
  - – Entity Framework Core – SQL Server
  - – MVCSportStore

MVCSportStore

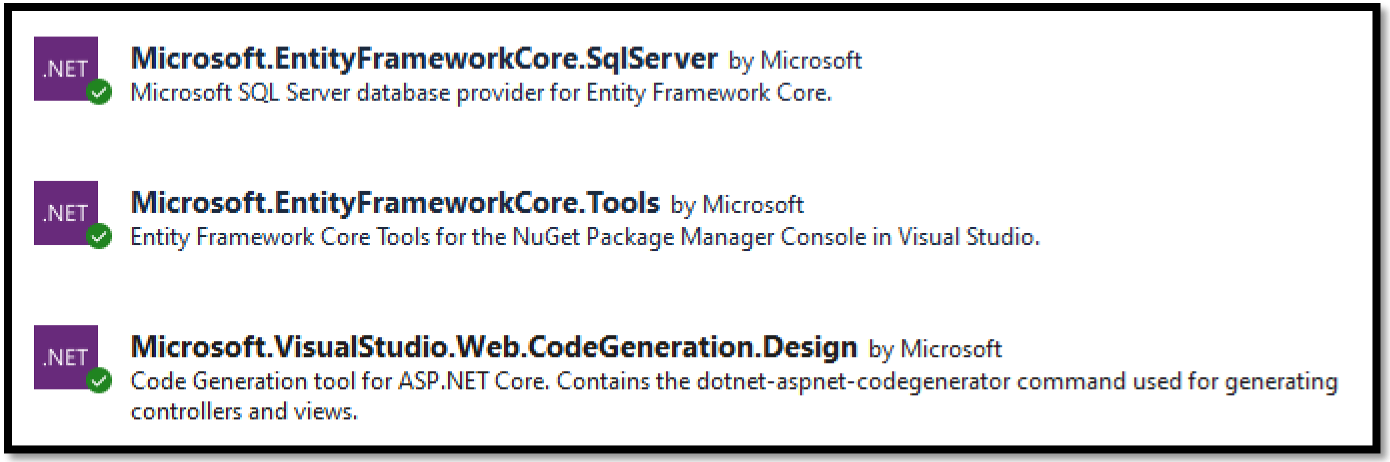


ASP.NET Core Web App (Model-View-Controller)

A project template for creating an ASP.NET Core application with example ASP.NET Core MVC Views and Controllers. This template can also be used for RESTful HTTP services.

- New visual studio project
  - ASP.Net Core MVC
    - MVCSportStore

Nuget Package Manager



Microsoft.EntityFrameworkCore.SqlServer by Microsoft
Microsoft SQL Server database provider for Entity Framework Core.

Microsoft.EntityFrameworkCore.Tools by Microsoft
Entity Framework Core Tools for the NuGet Package Manager Console in Visual Studio.

Microsoft.VisualStudio.Web.CodeGeneration.Design by Microsoft
Code Generation tool for ASP.NET Core. Contains the dotnet-aspnet-codegenerator command used for generating controllers and views.

ASP.Net Core MVCSportStore

DATA

Data - DbContext

- Project - Add folder Data
  - Data - Add class StoreDbContext

```
using Microsoft.EntityFrameworkCore;

namespace MVCSportStore.Data
{
    public class StoreDbContext : DbContext
    {
         public StoreDbContext(DbContextOptions<StoreDbContext> options) : base(options)
         { }
}
```

Data – DbContext – connection string

Connectionstring

- **Appsettings.json**

Adding a Configuration Setting in the appsettings.json File in the project Folder

```
{
		"Logging": {
				"LogLevel": {
				"Default": "Information",
				"Microsoft": "Warning",
				"Microsoft.Hosting.Lifetime": "Information"
				}
		},
		"AllowedHosts": "*",
		"ConnectionStrings":
		   {
		     "SportStoreConnection":
		         "Server=(localdb)\\MSSQLLocalDB;
		         Database=MVCSportStore;
		         MultipleActiveResultSets=true"
		   }
}
```

Data – DbContext – Program.cs

```
var builder = WebApplication.CreateBuilder(args);

var connectionString =
   builder.Configuration.GetConnectionString("SportStoreConnection");

builder.Services.AddDbContext<StoreDbContext>(
   options => options.UseSqlServer(connectionString));
```

SQL Server

Program.cs

```
connectionString =….GetConnectionString("SportStoreConnection");

builder.Services.AddDbContext<StoreDbContext>(
   options => options.UseSqlServer(connectionString));
```

Appsettings.json

```
"ConnectionStrings":{"SportStoreConnection":"Server=(localdb)\\MSSQL …
```

Data/StoreDbContext.cs

```
namespace MVCSportStore.Data
{
    public class StoreDbContext : DbContext
    {
        public StoreDbContext(DbContextOptions<StoreDbContext> options) : 
base(options)
```

ASP.Net Core MVCSportStore

MODELS – MODEL VALIDATION

Models - Data - Product

- Add Folder -> Data -> Models/Data
  - Add Class -> Product.cs

```
using System.ComponentModel.DataAnnotations;
using System.ComponentModel.DataAnnotations.Schema;

namespace MVCSportStore.Models.Data
{
    public class Product
    {
        public long? ProductId { get; set; }
        public string? Name { get; set; }
        public string? Description { get; set; }
        [Column(TypeName = "decimal(8,2)")]
        public decimal? Price { get; set; }
        [Required]
        public string? Category { get; set; }
    }
}
```

Models - Data - Product

```
namespace MVCSportStore.Models.Data
{
    public class Product
    {
        …
        [Required]
        public string? Category { get; set; }
        //Get property
        public decimal ProductPrice =>
                       (Price == null) ? 0 : (decimal)Price;
    }
}
```

Product -constructoren

```
namespace MVCSportStore.Models.Data
{
    public class Product
    {
        public Product()
        {
        }
        public Product(string[] data)
        {
            //name;description;category;price
            Name = data[0];
            Description = data[1];
            Category = data[2];
            Price = decimal.Parse(data[3]);
        }
       …
```

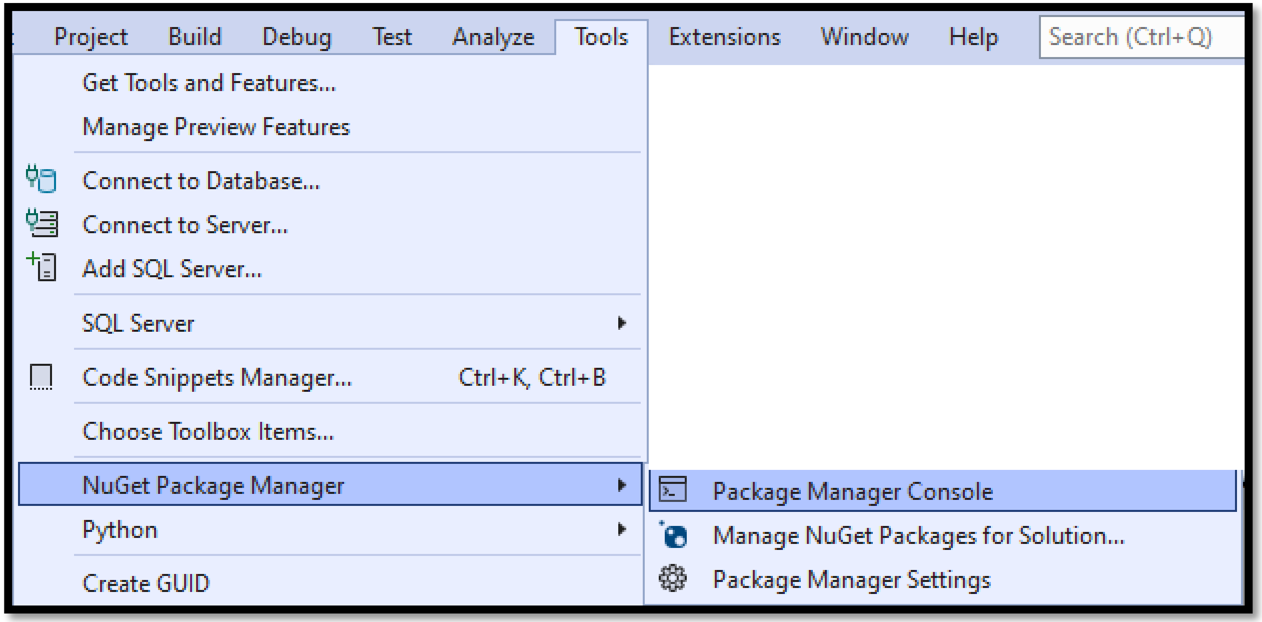
Data – DbContext - Products

```
using Microsoft.EntityFrameworkCore;

namespace MVCSportStore.Data
{
    public class StoreDbContext : DbContext
    {
        public StoreDbContext(DbContextOptions<StoreDbContext> options) : base(options) {}

        public DbSet<Product> Products { get; set; }

}
```

ASP.Net Core MVCSportStore

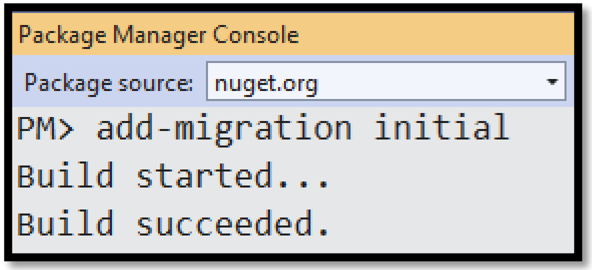
NUGET PACKAGE MANAGER MIGRATION

Tools - Nuget Package Manager



Package Manager Console

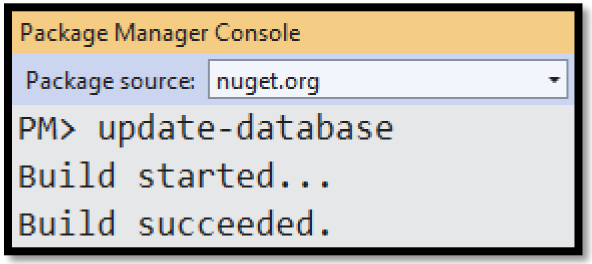
- `Add-migration initial`
- `Update-Database`



Package Manager Console

Package source: nuget.org

```
PM> add-migration initial
Build started...
Build succeeded.
```



Package Manager Console

Package source: nuget.org

```
PM> update-database
Build started...
Build succeeded.
```

ASP.Net Core MVCSportStore

DEFAULT DATA SEEDDATA

Data - DefaultData

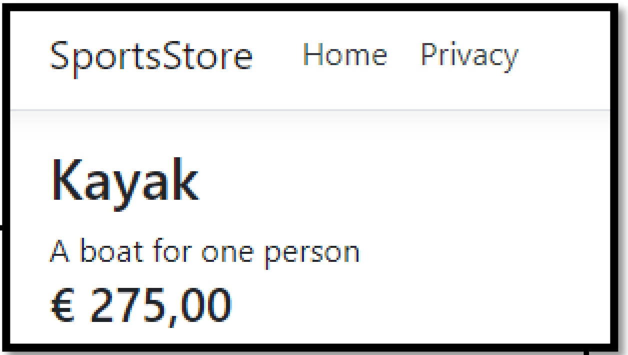
Data folder

- Add class DefaultData.cs

```
namespace MVCSportStore.Data
{
    public static class DefaultData
    {
        public static IEnumerable<string> Products => GetProductList();
        private static IEnumerable<string> GetProductList()
        {
            List<string> products = new List<string>();
                //Teams data
            return products;
        }
    }
}
```

Data - SeedData

```
namespace MVCSportStore.Data
{
    public static class SeedData
    {
        public static void EnsurePopulated(WebApplication app)
        {
            using (var scope = app.Services.CreateScope())
            {
                var context =
scope.ServiceProvider.GetRequiredService<StoreDbContext>();
                if(!context.Products.Any())
                {
                    //foreach product in DefaultData
                }
            }
        }
    }
}
```



Kayak

A boat for one person

€ 275,00

Data - SeedData

```
…
var context = scope.ServiceProvider.GetRequiredService<StoreDbContext>();
if (!context.Products.Any())
{
    foreach (var productString in DefaultData.Products)
    {
        var product = new Product(productString.Split(';'));
        context.Products.Add(product);
    }
    context.SaveChanges();
}
```

Data – SeedData – Program.cs

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

SeedData.EnsurePopulated(app);

app.Run();
```

ASP.Net Core MVCSportStore

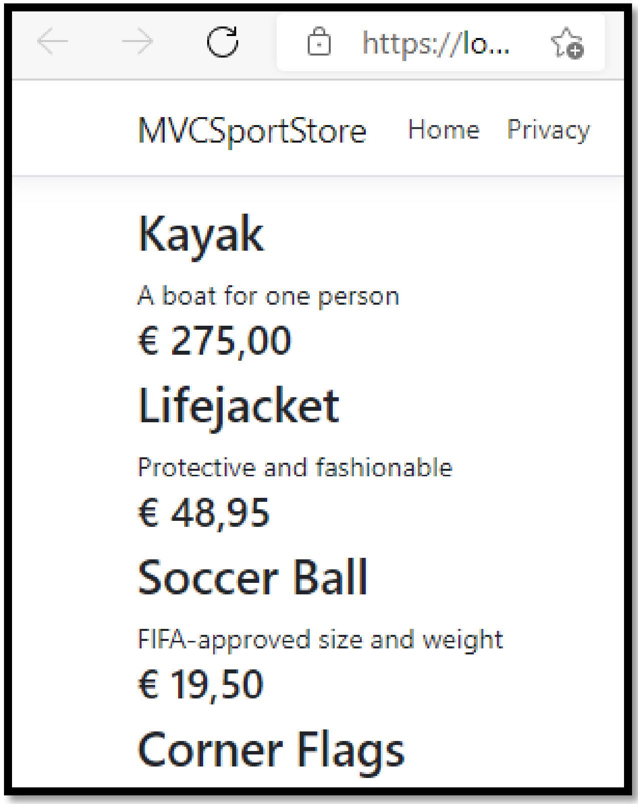
HOMECONTROLLER DEPENDENCY INJECTION

HomeController

```
namespace MVCSportStore.Controllers
{
    public class HomeController : Controller
    {
        private StoreDbContext _context;
        public HomeController(StoreDbContext context)
        {
            _context = context;
        }
        public IActionResult Index()
        {
            return View(_context.Products);
        }
    }
}
```

Views – Home – Index.cshtml

```
@model IEnumerable<Product>

@foreach (var p in Model)
{
    <div>
        <h3>@p.Name</h3>
        @p.Description
        <h4>
            @p.ProductPrice.ToString("c")
        </h4>
    </div>
}
```



Kayak

A boat for one person

€ 275,00

Lifejacket

Protective and fashionable

€ 48,95

Soccer Ball

FIFA-approved size and weight

€ 19,50

Corner Flags

ASP.Net Core MVCSportStore

VIEWMODELS PAGING

Models – PagingInfo

```
namespace MVCSportStore.ViewModels
{
    public class PagingInfo
    {
        public int TotalItems { get; set; }
        public int PageItems { get; set; }
        public int CurrentPage { get; set; }
        public int TotalPages =>
(int)Math.Ceiling((decimal)TotalItems / ItemsPerPage);
    }
}
```

Models – ProductModel

```
namespace MVCSportStore.ViewModels
{
    public class ProductModel
    {
        public IEnumerable<Product> Products { get; set; }
        public PagingInfo PagingInfo { get; set; }
    }
}
```

Models – PagingSettings

```
namespace MVCSportStore.ViewModels
{
    public static class PagingSettings
    {
        public static int ProductPagination = 4;
        public static bool PagingSettingsPage = false;

    }
}
```

!!Extra

Maak een PagingSettingsPage

- **(SettingsController)**
- **HttpGet/Post Form voor de Product Pagination property te wijzigen**

ASP.Net Core MVCSportStore

OEFENING 12 - SPORTSTORE