C# Web

PRO/PRW - C# Web 1

DE HOGESCHOOL MET HET NETWERK
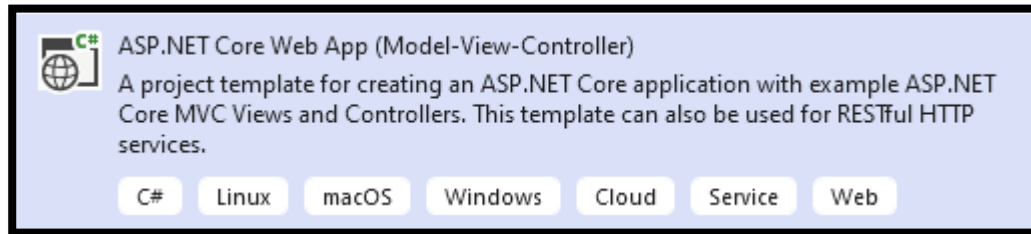
Doel

- 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.

C#   Linux   macOS   Windows   Cloud   Service   Web

- 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**

```csharp
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

```csharp
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

```csharp
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

```csharp
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

```csharp
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; }

}
```
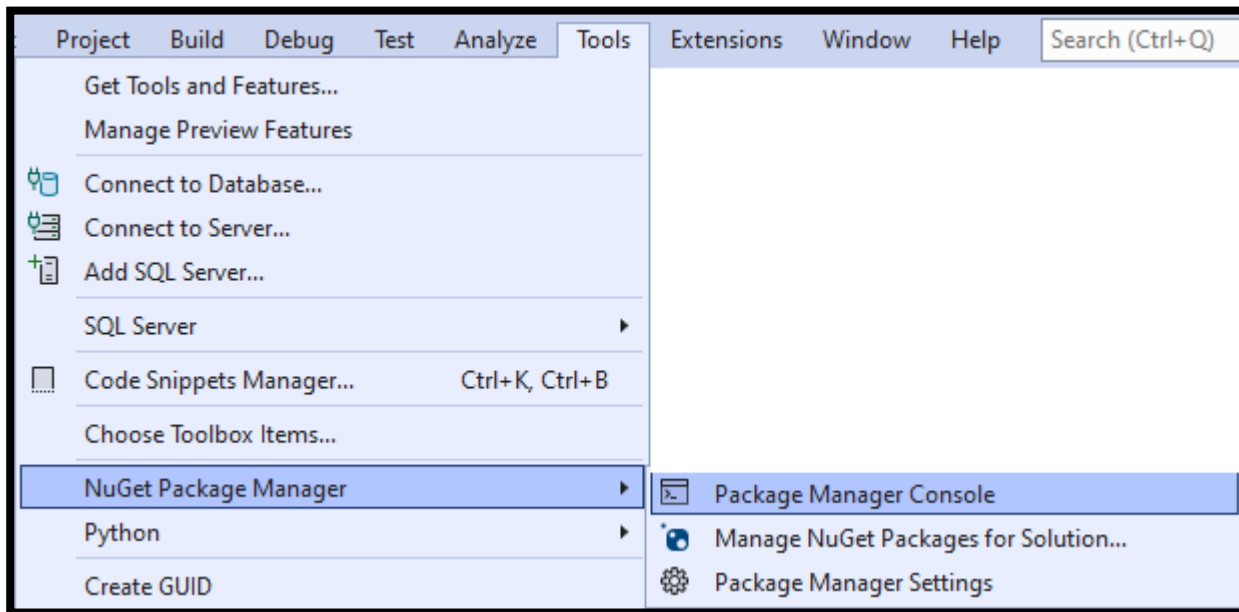
ASP.Net Core MVCSportStore

NUGET PACKAGE MANAGER MIGRATION

Tools - Nuget Package Manager

Package Manager Console

- Add-migration initial
- Update-Database

ASP.Net Core MVCSportStore

DEFAULT DATA SEEDDATA

Data - DefaultData

```
Data folder
Add Folder DefaultData
•  Add class SeedData.cs

namespace MVCSportStore.Data.DefaultData
{
    public static class SeedData
    {
        private 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
                }
            }
        }
    }
}
```

SportsStore   Home   Privacy

Kayak

A boat for one person

€ 275,00

Data - SeedData

```csharp
…
var context = scope.ServiceProvider.GetRequiredService<StoreDbContext>();
if (!context.Products.Any())
{
    foreach (var productString in 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
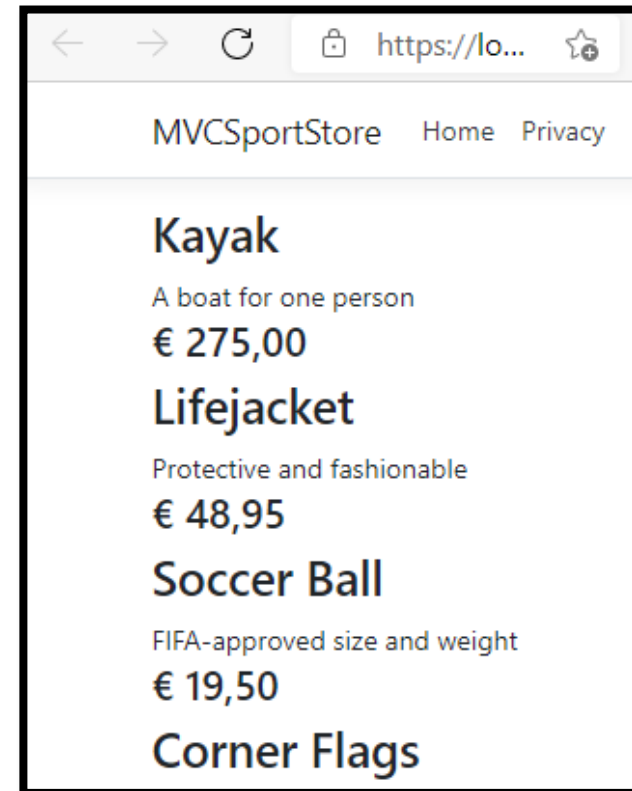
HomeController

```
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>
}
```


MVCSportStore   Home   Privacy

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

STYLING THE CONTENT _LAYOUT.CSHTML

Layout.cshtml

```html
<!DOCTYPE html>
<html>
<head>
    <meta name="viewport" content="width=device-width" />
    <title>Sport Store</title>
    <link rel="stylesheet" href="~/lib/bootstrap/dist/css/bootstrap.min.css" />
</head>
<body>
    <div class="bg-dark text-white p-2">
        <span class="navbar-brand ml-2">SPORT STORE</span>
    </div>
    <div class="row m-1 p-1">
        <div id="categories" class="col-3">
            Navigation menu
        </div>
        <div class="col-9">
            @RenderBody()
        </div>
    </div>
```


SPORT STORE

Navigation menu

Kayak
A boat for one person
€ 275,00

Lifejacket
Protective and fashionable
€ 48,95

Soccer Ball
FIFA-approved size and weight
€ 19,50

ASP.Net Core MVCSportStore

VIEWMODELS
PAGING

ViewModels – PagingInfo

Add Folder
- ViewModels

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

ViewModels – ProductModel

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

ViewModels – PagingSettings

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

    }
}
```

ASP.Net Core MVCSportStore

MODELS - PRODUCTS
PAGING – PRODUCT REPOSITORY

ViewModels – ProductRepository Products – GetProducts()

```
public class ProductRepository
{
    StoreDbContext _context;
    public ProductRepository(StoreDbContext context)
    {
        _context = context;
    }
    private IEnumerable<Product> GetProducts(int productPage)
    {
        return _context.Products
                .OrderBy(p => p.ProductId)
                .Skip((productPage - 1) * PagingSettings.ProductPagination)
                .Take(PagingSettings.ProductPagination);
    }
}

!Opgelet dit is geen Dependency Injection – Geen controller class
```

ViewModels – ProductRepository PagingInfo – GetPagingInfo()

```csharp
public class ProductRepository
{
    StoreDbContext _context;
    public ProductRepository(StoreDbContext context)
    {
        _context = context;
    }
    ...
    private PagingInfo GetPagingInfo(int productPage)
    {
        var pagingInfo = new PagingInfo();
        pagingInfo.CurrentPage = productPage;
        pagingInfo.ItemsPerPage = PagingSettings.ProductPagination;
        pagingInfo.TotalItems = _context.Products.Count();
        return pagingInfo;
    }
```

ViewModels – ProductRepository PagingInfo – GetProductModel()

```csharp
public class ProductRepository
{

    StoreDbContext _context;
    public ProductRepository(StoreDbContext context)
    {
        _context = context;
    }
    ...
    public ProductModel GetProductModel(int productPage)
    {

            var productModel = new ProductModel();
            productModel.Products = GetProducts(productPage);
            productModel.PagingInfo = GetPagingInfo(productPage);
            return productModel;

    }
}
```

ASP.Net Core MVCSportStore

PRODUCTPAGINATION HOMECONTROLLER

HomeController – Index

```
public class HomeController : Controller
{

    private StoreDbContext _context;
    private ProductRepository _repo;
    public HomeController(StoreDbContext context)
    {

        _context = context;
        _repo = new ProductRepository(_context);

    }
    public IActionResult Index(int id = 1)
    {

        var productModel = _repo.GetProductModel(id);
        return View(productModel);
        //return View(_context.Products);

    }


//_context -> Dependency Injection
//_repo -> normal class with _context as constructor parameter
```

Views – Home - Index

```
@model MVCSportStore.ViewModels.ProductModel

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

SPORT STORE

Navigation menu

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
Give your playing field a professional touch
€ 34,95

Views – Home - Index

```
@model ProductModel
@foreach (var p in Model.Products)
{
    …
}
<br />
<div class="row">
    @for(int i=1;i<=Model.PagingInfo.TotalPages;i++)
    {
        <div class="col text-center">
            <a asp-controller="Home" asp-action="Index" asp-route-id="@i">@i</a>
        </div>
    }
</div>
```

SPORT STORE

Navigation
menu

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
Give your playing field a professional touch
€ 34,95

1   2   3

HomeController – Index

```csharp
public class HomeController : Controller
{

    private StoreDbContext _context;
    private ProductRepository _repo;
    public HomeController(StoreDbContext context)
    {
        _context = context;
        _repo = new ProductRepository(_context);
    }
    public IActionResult Index(int id = 1)
    {
        var productsList = _repo.GetProductModel(id);
        return View(productsList);
    }
}
```

ASP.Net Core MVCSportStore

OEFENING 12 - SPORTSTORE

Models – PagingSettings

Oefening 12: MVC – SportStore

   1) Voeg een ProductController toe
   a. Voeg een Create actie toe

   2) Voeg een hyperlink naar de Create actie toe in de Home/Index view
   a. Na het toevoegen in de Product/Create navigeren we terug naar de
Home/Index

   3) Maak een PagingSettingsView pagina (SettingsController)
   a. HttpGet/Post Form voor de ProductPagination property te wijzigen

Oef12 – ProductController Dependency Injection

```csharp
public class ProductController : Controller
{
        StoreDbContext _context;
        public ProductController(StoreDbContext context)
        {
            _context = context;
        }
```

Oef12 – ProductController - Create

```
[HttpGet]
public IActionResult Create()
{

    return View();
}
[HttpPost]
public IActionResult Create(Product product)
{

    if (ModelState.IsValid)
    {
        try
        {
            _context.Products.Add(product);
            _context.SaveChanges();
            return RedirectToAction("Index", "Home");
        }
        catch (Exception ex)
        {
            ModelState.AddModelError("", "Invalid Data! -> " +
ex.Message.Substring(50));
        }
    }
    return View();
}
```

Oef12 – hyperlink
-> Create Action

```
Views/Home/Index.cshtml
…
<div>

        <a class="btn border border-secondary m-1"
                asp-controller="Product" asp-action="Create">
        Create new product
        </a>
</div>
…
```

HomeController – Index
Paging Buttons

```
<div class="row">
    @for (int i = 1; i <= Model.PagingInfo.TotalPages; i++)
    {
        <span style="float:left;width:50px">
            <a class="btn border border-secondary m-1"
asp-controller="Home" asp-action="Index" asp-route-id="@i">@i</a>
        </span>
    }
</div>
```

SPORT STORE

Navigation menu

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
Give your playing field a professional touch
€ 34,95

1 2 3

SettingsController - PagingSettingsView PagingSettings - ProductPagination

```csharp
namespace MVCSportStore.ViewModels
{

    public static class PagingSettings
    {

        public static int ProductPagination = 4;
        public static bool PagingSettingsPage = false;
    }
    public class PagingSettingViewModel
    {

        public int ProductPagination { get; set; }
    }
}
```

SettingsController - PagingSettingsView PagingSettings - ProductPagination

```csharp
public class SettingsController : Controller
{
    [HttpGet]
    public IActionResult PagingSettingsView()
    {
        PagingSettingViewModel paging = new PagingSettingViewModel();
        paging.ProductPagination = PagingSettings.ProductPagination;
        return View(paging);
    }
    [HttpPost]
    public IActionResult PagingSettingsView(PagingSettingViewModel paging)
    {
        PagingSettings.ProductPagination = paging.ProductPagination;
        return RedirectToAction("Index", "Home");
    }
}
```
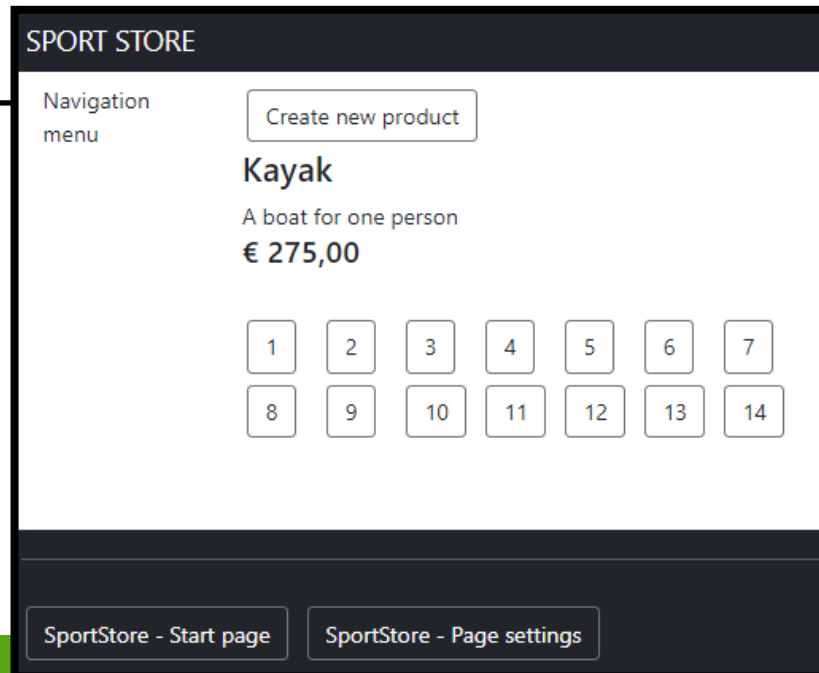
SettingsController - PagingSettingsView
PagingSettings - ProductPagination

```
@model MVCSportStore.ViewModels.PagingSettingViewModel
<h4>Paging settings</h4>
<hr />
<div class="row">
  <div class="col-md-4">
    <form asp-action="PagingSettingsView">
      <div asp-validation-summary="All" class="text-danger"></div>
        <div class="form-group">
          <label class="control-label">Products in one page</label>
          <input asp-for="ProductPagination" class="form-control" />
          <span asp-validation-for="ProductPagination" class="text-danger"></span>
        </div>
        <br />
      <div class="form-group">
        <input type="submit" value="Save" class="btn btn-primary" />
      </div>
    </form>
  </div>
</div>
```

SettingsController - PagingSettingsView _Layout -> Styling the footer

```html
<footer class="bg-dark text-white p-1 footer fixed-bottom">
  <hr>
  <span class="navbar-brand ml-2">
    <a class="btn border border-secondary m-1 text-white"
        asp-controller="Home" asp-action="Index">
        SportStore - Start page</a>
    <a class="btn border border-secondary m-1 text-white"
        asp-controller="Settings" asp-action="PagingSettingsView">
        SportStore - Page settings</a>
  </span>
</footer>
```

Pluralsight

.Net Core 3.1
https://app.pluralsight.com/library/courses/building-aspdotnet-core-mvc-web-applications
.Net 6
https://app.pluralsight.com/library/courses/asp-dot-net-core-6-fundamentals/table-of-contents

https://app.pluralsight.com/library/courses/csharp-collections-beginning