C# Web – MVC

PRO - C# Web 1

DE HOGESCHOOL
MET HET NETWERK

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

Doel

- ASP.Net Core toepassing
  - MVC Web Application
  - Identity Framework

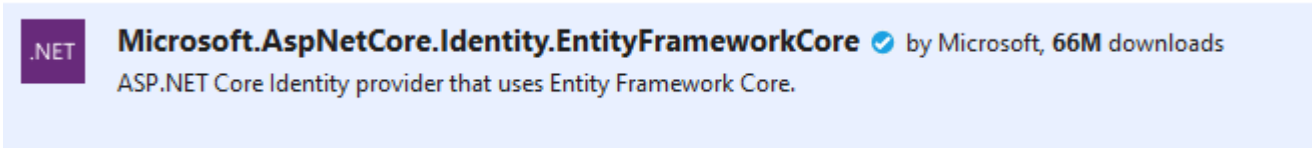
ASP.NET CORE

MVC Web Application
MVCVoertuig

MVCVoertuig - NPM

- Nuget Package Manager
  - EntityFrameworkCore.SqlServer
  - EntityFrameworkCore.Tools

- Nuget Package Manager
  - Identity.EntityFrameworkCore



MVCVoertuig – Data service

Startup.cs

```
public void ConfigureServices(IServiceCollection services)
{
    services.AddControllersWithViews();
    services.AddDbContext<VoertuigDbContext>(opts => {
        opts.UseSqlServer(
        Configuration["ConnectionStrings:VoertuigConnection"]);
    });
}
```

MVCVoertuig – Data service Identity framework

Startup.cs

```
public void ConfigureServices(IServiceCollection services)
{
    services.AddDbContext<VoertuigDbContext>(opts => {
        opts.UseSqlServer(
        Configuration["ConnectionStrings:VoertuigConnection"]);
    });
    services.AddIdentity<IdentityUser, IdentityRole>()
        .AddEntityFrameworkStores<VoertuigDbContext>();
    services.AddControllersWithViews();
}
```

Identity Framework - IdentityUser

- IdentityUser

- IdentityDbContext (Identity.EntityFramework)
  - SQL Server
    - AspNetUsers
      - Primary table to store user information
      - Automatisch aangemaakt bij een migratie

```
namespace MVCVoertuig.Data
{
    public class VoertuigDbContext : IdentityDbContext
    {
        public VoertuigDbContext(DbContextOptions<VoertuigDbContext>
options) : base(options) { }
        public DbSet<Voertuig> Voertuigen { get; set; }
    }
}
```

Identity Framework - Authentication

- Startup.cs

```
public void Configure(IApplicationBuilder app, IWebHostEnvironment
env)
{
    ...
    app.UseAuthentication();
    app.UseAuthorization();
    app.UseEndpoints(endpoints =>
    {
        endpoints.MapControllerRoute(
            name: "default",
            pattern: "{controller=Home}/{action=Index}/{id?}");
     });
    SeedData.EnsurePopulated(app);
}
```

Identity Framework - Authentication

- Controllers - VoertuigenController

```
namespace MVCVoertuig.Controllers
{
    [Authorize]
    public class VoertuigenController : Controller
```

```
NPM
Add-migration identity
Update-database

Test VoertuigenController
=> ERROR
```

Identity Framework – View Model

- ViewModels - RegisterViewModel

```
namespace MVCVoertuig.ViewModels
{
    public class RegisterViewModel
    {
        [Required]
        public string Email { get; set; }
        [Required]
        public string Password { get; set; }
        [Required]
        public string ConfirmPassword { get; set; }
    }
}
```

Identity Framework – View Model

- ViewModels - LoginViewModel

```
namespace MVCVoertuig.ViewModels
{
    public class LoginViewModel
    {
        [Required]
        public string Email { get; set; }
        [Required]
        public string Password { get; set; }
    }
}
```

Identity Framework – AccountController

- Controllers – Add empty controller
  - AccountController

Identity Framework – AccountController

- AccountController - Register

```
namespace MVCVoertuig.Controllers
{
    public class AccountController : Controller
    {
        #region register
        [HttpGet]
        public IActionResult Register()
        {
            return View();
        }
        [HttpPost]
        public IActionResult Register(RegisterViewModel user)
        {
            return View();
        }
        #endregion
    }
}
```

Identity Framework – AccountController

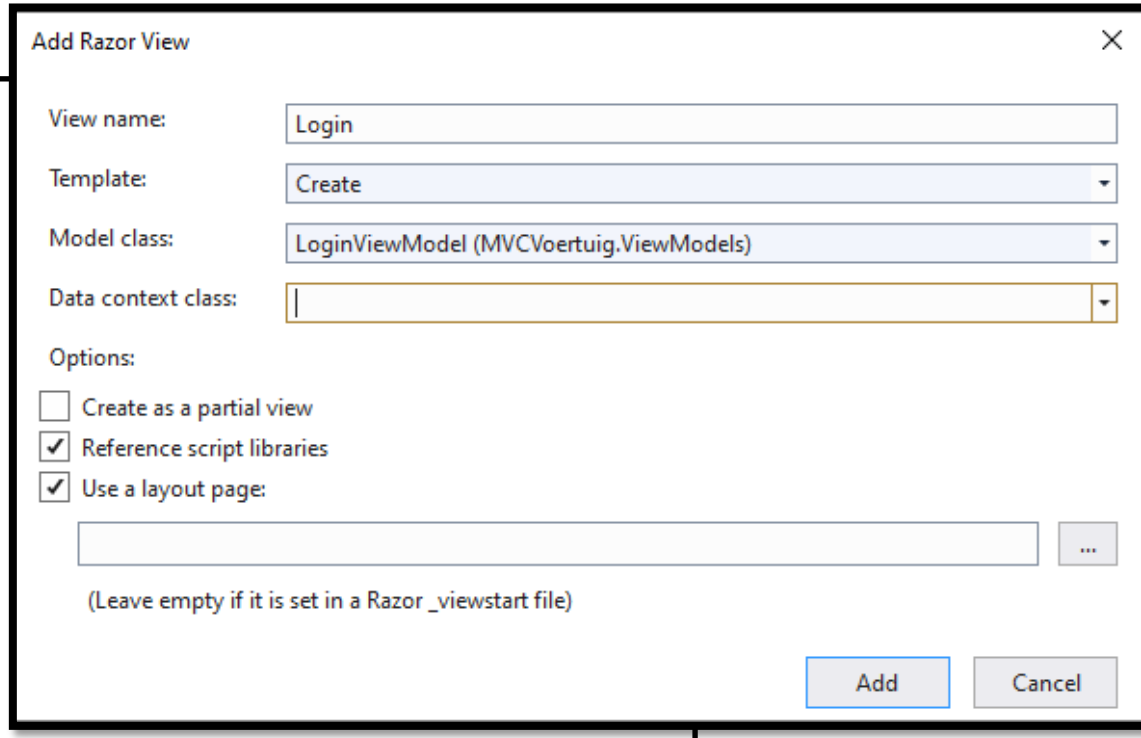
- AccountController - Login

```
namespace MVCVoertuig.Controllers
{
    public class AccountController : Controller
    {
        #region login
        [HttpGet]
        public IActionResult Login()
        {
            return View();
        }
        [HttpPost]
        public IActionResult Login(LoginViewModel user)
        {
            return View();
        }
        #endregion
    }
}
```

Identity Framework – Views

Create Folder Account in Views folder

- Views/Account
  - Scaffold View
    - Name: Login
    - Model: LoginViewModel
    - Template: Create
  - Scaffold View
    - Name: Register
    - Model: RegisterViewModel
    - Template: Create



Identity Framework – Views/Account/Login

```
<div class="row">
    <div class="col-md-4">
        <form asp-action="Login">
            <div asp-validation-summary="ModelOnly" class="text-danger"></div>
            <div class="form-group">
                <label asp-for="Email" class="control-label"></label>
                <input asp-for="Email" class="form-control" />
                <span asp-validation-for="Email" class="text-danger"></span>
            </div>
            <div class="form-group">
                <label asp-for="Password" class="control-label"></label>
                <input asp-for="Password" class="form-control" />
                <span asp-validation-for="Password" class="text-danger"></span>
            </div>
            <div class="form-group">
                <input type="submit" value="Login" class="btn btn-primary" />
            </div>
        </form>
    </div>
</div>
<div>
    <a asp-action="Register">Registreer een nieuwe gebruiker</a>
</div>
```

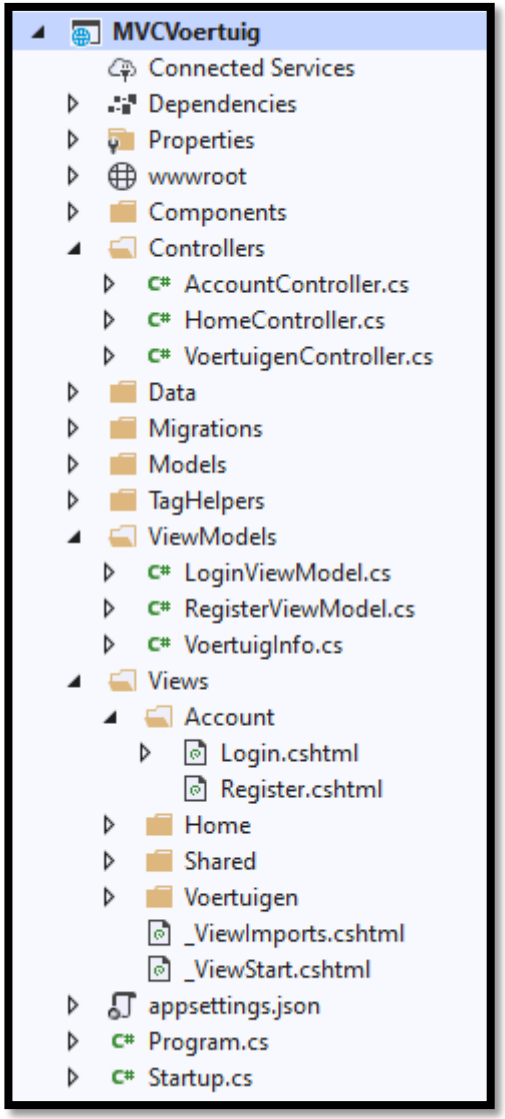
Identity Framework – Views/Account/Register

```
<div class="row">
    <div class="col-md-4">
        <form asp-action="Register">
            <div asp-validation-summary="ModelOnly" class="text-danger"></div>
            <div class="form-group">
                <label asp-for="Email" class="control-label"></label>
                <input asp-for="Email" class="form-control" />
                <span asp-validation-for="Email" class="text-danger"></span>
            </div>
            <div class="form-group">
                <label asp-for="Password" class="control-label"></label>
                <input asp-for="Password" class="form-control" />
                <span asp-validation-for="Password" class="text-danger"></span>
            </div>
            <div class="form-group">
                <label asp-for="ConfirmPassword" class="control-label"></label>
                <input asp-for="ConfirmPassword" class="form-control" />
                <span asp-validation-for="ConfirmPassword" class="text-danger"></span>
            </div>
            <div class="form-group">
                <input type="submit" value="Register" class="btn btn-primary" />
            </div>
        </form>
    </div>
</div>
```

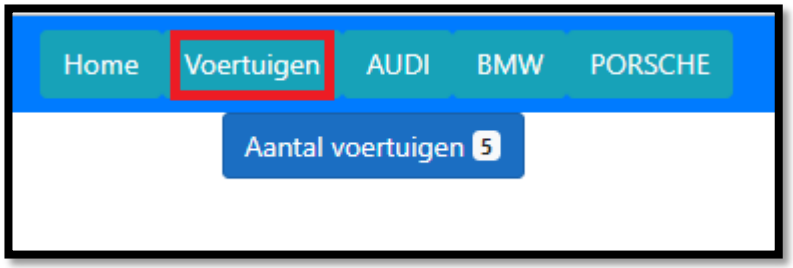
Identity Framework - Overview

Solution explorer

- Controllers
  - AccountController
- ViewModels
  - LoginViewModel
  - RegisterViewModel
- Views
  - Views/Account
    - Login.cshtml
    - Register.cshtml



Identity Framework - Test



[Authorize]

- VoertuigenController
  -> AccountController/Login

Identity Framework - Register

Dependency Injection

```
UserManager<IdentityUser> _userManager;
public AccountController(UserManager<IdentityUser> userManager)
{
    _userManager = userManager;
}
```

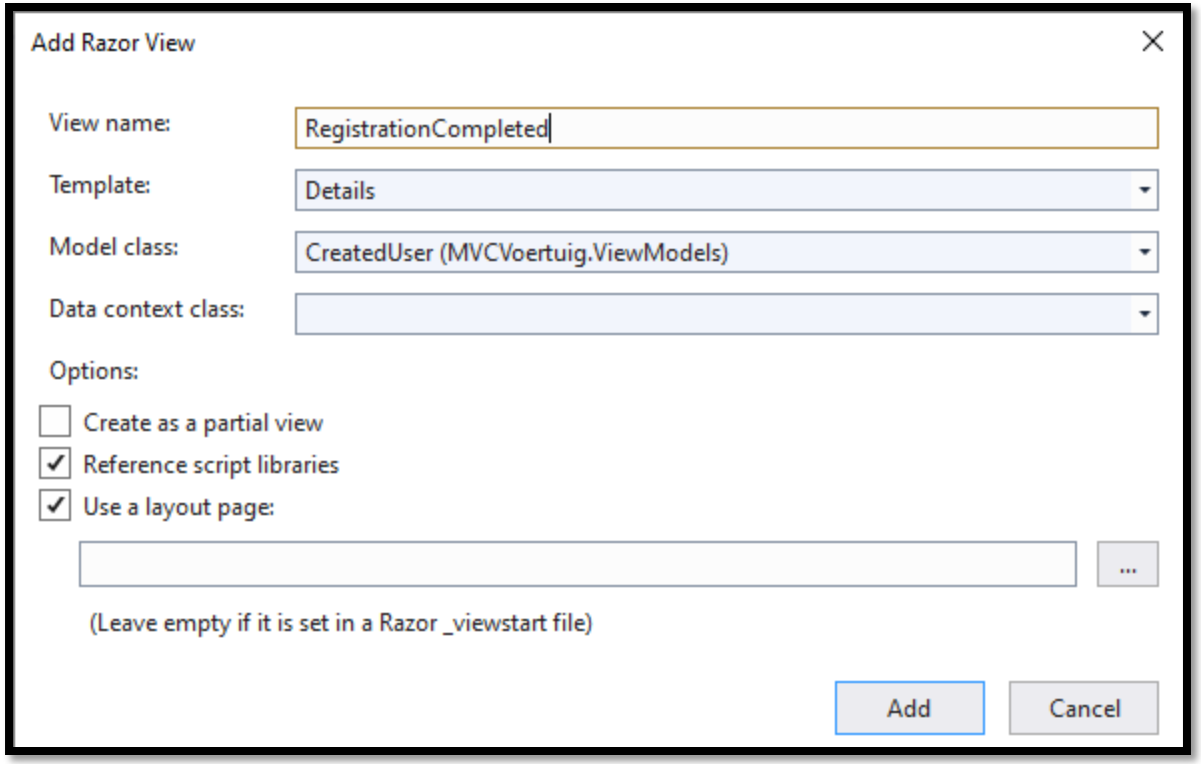
```
[HttpPost]
public async Task<IActionResult> RegisterAsync(RegisterViewModel user)
{
    if (ModelState.IsValid)
    {
        var identityUser = new IdentityUser { UserName = user.Email, Email = user.Email };
        await _userManager.CreateAsync(identityUser, user.Password);
        return View("Login");
    }
    return View();
}
```

Identity Framework - Register

- ViewModels – Add class
  - CreatedUser

```
namespace MVCVoertuig.ViewModels
{
    public class CreatedUser
    {
        public DateTime CreationDate { get; set; }
        public IdentityUser IdentityUser { get; set; }
    }
}
```

Identity Framework - Register



Identity Framework - RegistrationCompleted

```
@model MVCVoertuig.ViewModels.CreatedUser
@{
    ViewData["Title"] = "RegistrationCompleted";
}
<h1>Registration completed</h1>
<div>
    <hr />
    <h4>Welkom @Model.IdentityUser.Email</h4>
    <hr />
    <dl class="row">
        <dt class = "col-sm-2">
            @Html.DisplayNameFor(model => model.CreationDate)
        </dt>
        <dd class = "col-sm-10">
            @Html.DisplayFor(model => model.CreationDate)
        </dd>
    </dl>
</div>
```

Identity Framework - Register

```
[HttpPost]
public async Task<IActionResult> RegisterAsync(RegisterViewModel user)
{
    if (ModelState.IsValid)
    {
        if (!user.Password.Equals(user.ConfirmPassword))
        {
            ModelState.AddModelError("", "Wachtwoord moet identiek zijn!");
            return View(user);
        }
        else
        {
            //registration process
            var createdUser=new CreatedUser();
             // fill properties
            return View("RegistrationCompleted", createdUser);
        }
    }
    return View(user);
}
```

Identity Framework – Register RegisterViewModel

```
namespace MVCVoertuig.ViewModels
{
    public class RegisterViewModel
    {
        [Required]
        public string Email { get; set; }
        [Required]
        public string Password { get; set; }
        [Required]
        [Compare("Password", ErrorMessage="paswoord moet overeenkomen!")]
        public string ConfirmPassword { get; set; }
    }
}
```

Identity Framework – Register ViewModels - RegisterResult

```
namespace MVCVoertuig.ViewModels
{
    public class RegisterResult
    {
        public bool Succeeded { get; set; }
        public CreatedUser CreatedUser { get; set; }
        public List<string> Errors = new List<string>();
    }
}
```

Identity Framework - RegisterUser

```
private async Task<RegisterResult> RegisterUserAsync(RegisterViewModel user)
{
    var registerResult = new RegisterResult();
    var identityUser = new IdentityUser { UserName = user.Email, Email = user.Email };
    var result = await _userManager.CreateAsync(identityUser, user.Password);
    if (result.Succeeded)
    {
        var createdUser = new CreatedUser { CreationDate = DateTime.Now, IdentityUser = identityUser };
        registerResult.Succeeded = true;
        registerResult.CreatedUser = createdUser;
    }
    else
    {
        if (result.Errors.Count() > 0)
        {
            foreach (var error in result.Errors)
            {
                registerResult.Errors.Add(error.Description);
            }
        }
        else
        {
            registerResult.Errors.Add("Er is een probleem om de gebruiker te registreren!");
        }
    }
    return registerResult;
}
```

Identity Framework - Register

```
[HttpPost]
public async Task<IActionResult> RegisterAsync(RegisterViewModel user)
{
    if (ModelState.IsValid)
    {
        if (!user.Password.Equals(user.ConfirmPassword))
        {…}
        else
        {
            var result = await RegisterUserAsync(user);
            if (result.Succeeded)
            {
                return View("RegistrationCompleted", result.CreatedUser);
            }
            else
            {
                foreach (string error in result.Errors)
                    ModelState.AddModelError("", error);
                return View(user);
            }
        }
    }
    return View(user);
}
```

Identity Framework – Register UserExist - FindByEmail

```
private async Task<bool> UserExistAsync(RegisterViewModel user)
{
    bool userExist = false;
    var result = await _userManager.FindByEmailAsync(user.Email);
    if (result != null)
        userExist = true;
    return userExist;
}
```

Identity Framework - Register

```
[HttpPost]
public async Task<IActionResult> RegisterAsync(RegisterViewModel user)
{
    if (ModelState.IsValid)
    {
        if (await UserExistAsync(user))
        {
            ModelState.AddModelError("", "Email adres is al geregistreerd!");
            return View(user);
        }
…
    }
…
}
```

Identity Framework - UserManager

- _userManager.FindByEmailAsync(user.Email)
- _userManager.CreateAsync

```
var identityUser =
    new IdentityUser { UserName = user.Email, Email = user.Email };
var result = await _userManager.CreateAsync(identityUser, user.Password);
```

ASP.NET CORE

MVC Web Application
Identity Framework
Oefening13

- **Register**