



Blazor Server

C# Web 2

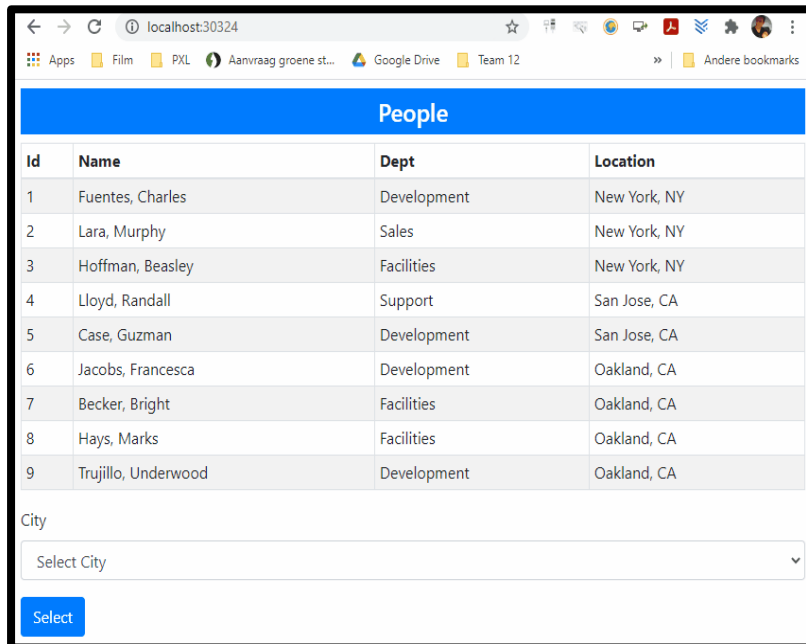
DE HOGESCHOOL MET HET NETWERK

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



Startproject

- Visual Studio – **People-WebService (BlazorStartProject)**
-> **BlazorPeopleList & routing**
 - ASP .Net Core MVC applicatie voor een groot bedrijf met meerdere departementen
 - Toont de mensen in het bedrijf
 - Naam
 - Departement
 - Locatie (stad)
 - Na het selecteren van een stad worden de mensen met die locatie gemarkeerd



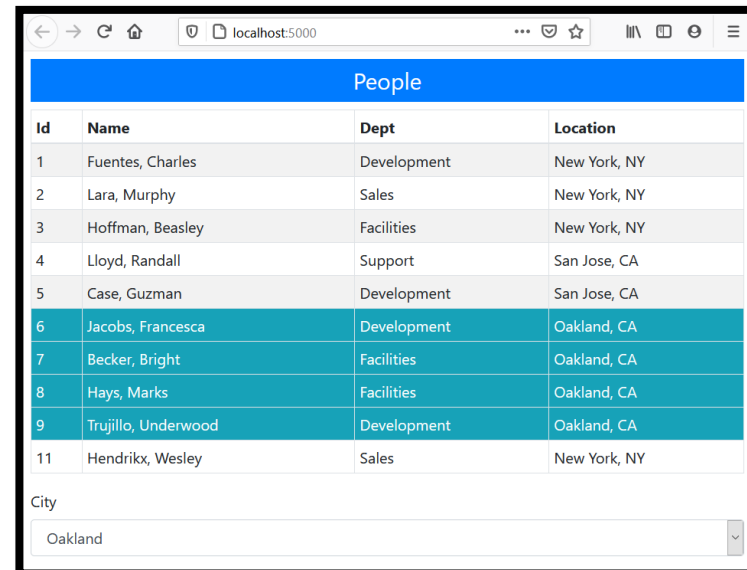
People			
Id	Name	Dept	Location
1	Fuentes, Charles	Development	New York, NY
2	Lara, Murphy	Sales	New York, NY
3	Hoffman, Beasley	Facilities	New York, NY
4	Lloyd, Randall	Support	San Jose, CA
5	Case, Guzman	Development	San Jose, CA
6	Jacobs, Francesca	Development	Oakland, CA
7	Becker, Bright	Facilities	Oakland, CA
8	Hays, Marks	Facilities	Oakland, CA
9	Trujillo, Underwood	Development	Oakland, CA

City

Select City

Select

BlazorPeopleList



People			
Id	Name	Dept	Location
1	Fuentes, Charles	Development	New York, NY
2	Lara, Murphy	Sales	New York, NY
3	Hoffman, Beasley	Facilities	New York, NY
4	Lloyd, Randall	Support	San Jose, CA
5	Case, Guzman	Development	San Jose, CA
6	Jacobs, Francesca	Development	Oakland, CA
7	Becker, Bright	Facilities	Oakland, CA
8	Hays, Marks	Facilities	Oakland, CA
9	Trujillo, Underwood	Development	Oakland, CA
11	Hendrixx, Wesley	Sales	New York, NY

City

Oakland

ASP.Net Core MVC – Blazor Server

ROUTING

Routing

- Doel:
 - 1 html pagina die verschillende Razor componenten kan tonen afhankelijk van veranderingen in de url
 - Wij zullen een “beheer” gedeelte uitwerken waarmee je vanuit een overzicht van personen een persoon kan bewerken. Er zijn dus 2 Razor componenten die we willen tonen:
 - Overzicht van personen
 - Bewerken van een persoon
- Stappen:
 - Html pagina (host) bouwen die de componenten kan tonen
 - Router component toevoegen aan de host pagina
 - De 2 Razor componenten bouwen en associëren met bepaalde url's

Host html pagina

Voeg een **Razor View** “_BlazorServer_Host.cshtml” (≠ Razor Component) toe in de folder *Views/Shared* :

```
@page "/"
@{
    Layout = null;
}
<!DOCTYPE html>
<html>
<head>
    <title>@ViewBag.Title</title>
    <link rel="stylesheet" href="~/lib/bootstrap/dist/css/bootstrap.min.css" />
    <base href="~/\" />
</head>
<body>
    <div class="m-2">
        <component type="typeof(PeopleApp.Blazor.Manage.Routed)" render-mode="Server"/>
    </div>
    <script src="_framework/blazor.server.js"></script>
</body>
</html>
```

- De html bevat een Razor component (die we dadelijk nog gaan bouwen). Deze component zal een router zijn die verschillende Razor componenten kan tonen.
- De script tag voegt de nodige Blazor Server functionaliteit toe om te communiceren vanuit de browser met de applicatie op de server.

Host html pagina

Voeg een *BlazorController* toe in de *Controllers* folder die de *_BlazorServer_Host* view toont:

```
using Microsoft.AspNetCore.Mvc;

namespace PeopleApp.Controllers
{
    public class BlazorController : Controller
    {
        public IActionResult Index()
        {
            return View("_BlazorServer_Host");
        }
    }
}
```

Zorg er in *Startup.cs* voor dat alle requests voor url's die beginnen met *"/manage/"* naar de *Index* actie van de *BlazorController* gestuurd worden.

```
app.MapDefaultControllerRoute();
app.MapBlazorHub();
app.MapFallbackToController("/manage/{*path:nonfile}", "Index", "Blazor");
```

Router component

- Voeg een folder “*Manage*” toe aan de “*Blazor*” folder
- Voeg een Razor Component “*Routed*” toe aan de “*Manage*” folder:

```
<Router AppAssembly="typeof(Program).Assembly">
  <Found>
    <RouteView RouteData="@context"></RouteView>
  </Found>
  <NotFound>
    <h4 class="bg-danger text-white text-center p-2">
      No matching route found
    </h4>
  </NotFound>
</Router>
```

- De *Router* gaat afhankelijk van de url een bepaalde Razor component proberen te tonen op de plaats waar de *RouteView* tag staat.
- Als de *Router* geen geschikte Razor component vindt om te tonen, dan wordt de html in de *NotFound* tag getoond.

PeopleOverview

- Voeg een Razor Component "PeopleOverview" toe aan de "Manage" folder:

```
@page "/"manage/"
@page "/"manage/people"

<h4 class="bg-primary text-white text-center p-2">Manage People</h4>
<table class="table table-sm table-bordered table-striped">
  <thead>
    <tr>
      <th>Id</th><th>Name</th><th>Dept</th><th>Location</th><th></th>
    </tr>
  </thead>
  <tbody>
    @foreach (Person p in People)
    {
      <tr>
        <td>@p.Id</td>
        <td>@p.Surname, @p.Firstname</td>
        <td>@p.Department.Name</td>
        <td>@p.Location.City, @p.Location.State</td>
        <td>
          <NavLink class="btn btn-sm btn-warning" href="@GetEditUrl(p.Id)">Edit</NavLink>
        </td>
      </tr>
    }
  </tbody>
</table>

@code {
  [Inject]
  private IPersonRepository PersonRepository { get; set; }

  public IEnumerable<Person> People => PersonRepository.GetAll();

  public string GetEditUrl(long personId)
  {
    return $"/manage/people/edit/{personId}";
  }
}
```


PeopleOverview

- De *@page* directives definiëren welke routes (url's) overeenkomen met de component (de *Router* component gebruikt deze om te bepalen wanneer deze component getoond wordt)
- De *NavLink* tag genereert een a-tag die er voor zorgt dat een andere Razor component (editeer component) geladen wordt door de Router component. De editeer component gaan we dadelijk maken
- De *GetEditUrl* methode bouwt de url naar de editeer component

PeopleList

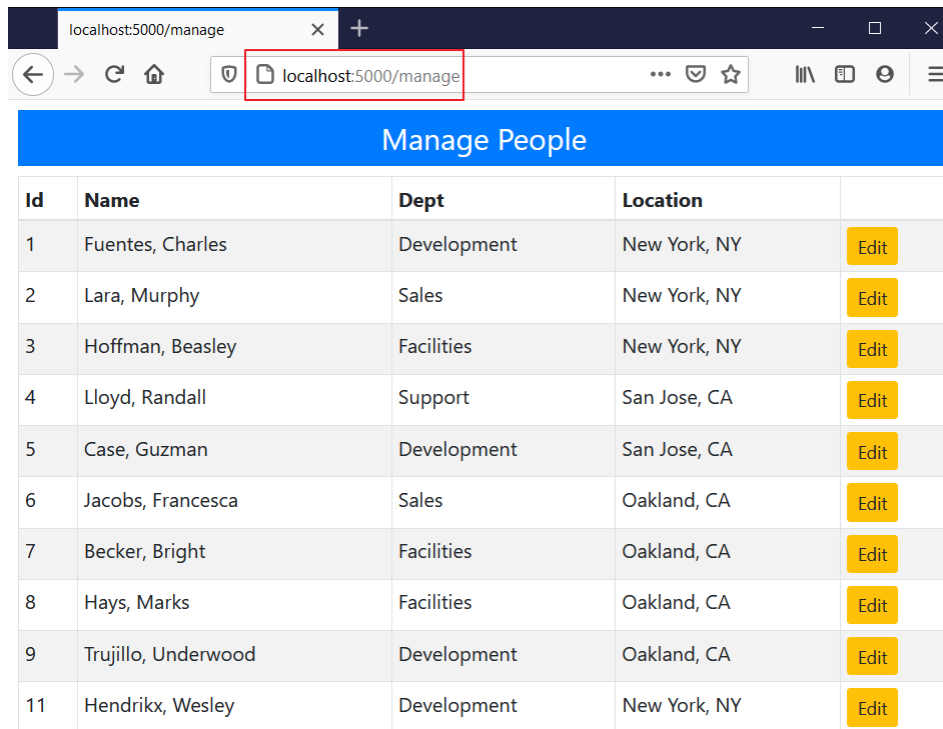
Voeg een link naar “manage” toe aan de PeopleList component om gemakkelijk naar het beheergedeelte te kunnen navigeren:

```
<div class="form-group">
  <label for="city">City</label>
  <select name="city" class="form-control" @bind="SelectedCity">
    <option disabled selected>Select City</option>
    @foreach (Location location in Locations)
    {
      <option value="@location.City" selected="@location.City ==
SelectedCity)">
        @location.City
      </option>
    }
  </select>
</div>

<div class="form-group">
  <a href="manage" class="btn btn-primary">Manage people</a>
</div>
```

PeopleOverview

- Start de applicatie en klik op de “Manage people” knop



Manage People				
Id	Name	Dept	Location	
1	Fuentes, Charles	Development	New York, NY	Edit
2	Lara, Murphy	Sales	New York, NY	Edit
3	Hoffman, Beasley	Facilities	New York, NY	Edit
4	Lloyd, Randall	Support	San Jose, CA	Edit
5	Case, Guzman	Development	San Jose, CA	Edit
6	Jacobs, Francesca	Sales	Oakland, CA	Edit
7	Becker, Bright	Facilities	Oakland, CA	Edit
8	Hays, Marks	Facilities	Oakland, CA	Edit
9	Trujillo, Underwood	Development	Oakland, CA	Edit
11	Hendriks, Wesley	Development	New York, NY	Edit

EditPerson

- Voeg een Razor Component “EditPerson” toe aan de “Manage” folder:

```
@page "/manage/people/edit/{id:long}"
<h4 class="bg-primary text-white text-center p-2">Edit person: @Person.Firstname @Person.Surname</h4>
```

```
@code {
    [Inject]
    public IPersonRepository PersonRepository { get; set; }

    [Parameter]
    public long Id { get; set; }

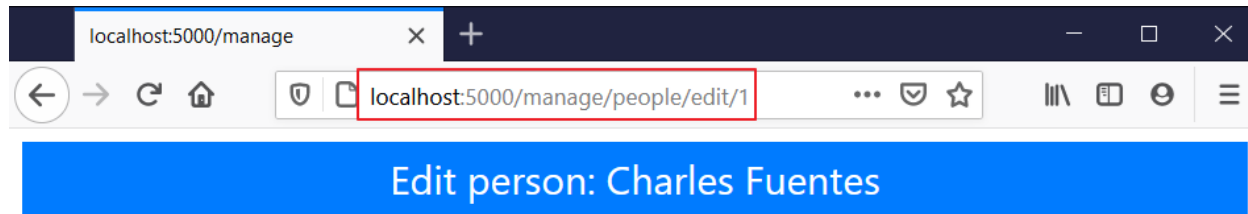
    public Person Person { get; set; }

    protected override void OnParametersSet()
    {
        Person = PersonRepository.GetById(Id);
    }
}
```

- De *@page* directive zorgt er voor dat deze component geladen wordt door de Router als er genavigeerd wordt naar bv. “/manage/people/edit/1” waarbij 1 de *Id* van de persoon is
- Het *[Parameter]* attribuut zal er voor zorgen dat de *Id* property de waarde krijgt van de id in de url
- Door de *OnParametersSet* methode te overriden kunnen we code uitvoeren net na dat alle properties met het *[Parameter]* attribuut een waarde hebben gekregen. Dit is het moment waarop we de persoon met de juiste id kunnen ophalen uit de repository

EditPerson

- Start de applicatie en klik op de “Manage people” knop
- Klik op de *Edit* knop bij een persoon



Entity Framework scope problem

- We hebben in onze componenten telkens 1 of meer repositories geïnjecteerd m.b.v. het [Inject] attribuut. De concrete implementaties van deze repositories maken gebruik van een DbContext class die zorgt voor de connectie met de database
- Met Blazor Server wordt er gebruik gemaakt van een permanente connectie tussen de browser en de applicatie op de server. Dit zorgt ervoor dat er slechts 1 instantie van de DbContext class gebruikt wordt door alle Razor componenten. Dit kan voor allerlei problemen zorgen (corrupte data)
- Oplossing: zorg er voor dat elke Razor component een eigen instantie van de DbContext gebruikt.
- Voeg de volgende code toe bovenaan in elke Razor component:

```
@using Microsoft.Extensions.DependencyInjection
@inherits OwningComponentBase
```

- Vervolgens kan je een repository als volgt gebruiken (doe dit voor alle geïnjecteerde repositories in de 3 Razor componenten):

```
private IPersonRepository PersonRepository;
override protected void OnInitialized()
{
    PersonRepository = ScopedServices.GetRequiredService<IPersonRepository>();
}
```

Bron:

<https://blazor-university.com/dependency-injection/component-scoped-dependencies/owning-multiple-dependencies-the-right-way/>

<https://learn.microsoft.com/en-us/aspnet/core/blazor/fundamentals/dependency-injection?view=aspnetcore-8.0#utility-base-component-classes-to-manage-a-di-scope>

ASP.Net Core MVC – Blazor Server

FORMS

EditPerson form

- Voeg formulier toe aan de Razor Component “EditPerson” :

```
<h4 class="bg-primary text-white text-center p-2">Edit person: @Person.Firstname @Person.Surname</h4>
<EditForm Model="Person">
  <div class="form-group">
    <label>ID</label>
    <InputNumber class="form-control" @bind-Value="Person.Id" disabled />
  </div>
  <div class="form-group">
    <label>Firstname</label>
    <InputText class="form-control" @bind-Value="Person.Firstname" />
  </div>
  <div class="form-group">
    <label>Surname</label>
    <InputText class="form-control" @bind-Value="Person.Surname" />
  </div>
  <div class="form-group">
    <label>Department ID</label>
    <InputNumber class="form-control" @bind-Value="Person.DepartmentId" />
  </div>
  <div class="text-center">
    <button type="submit" class="btn btn-primary">Submit</button>
    <NavLink class="btn btn-secondary" href="/manage/people">Back</NavLink>
  </div>
</EditForm>
```


EditPerson form

- Voor een Blazor formulier gebruik je de *EditForm* tag
 - Het *Model* attribuut wijst naar het object dat de gegevens in het formulier bevat
- De *InputNumber* tag zorgt voor een invulveld van het type *number*
 - Het *@bind-Value* attribuut zorgt ervoor dat wijzigingen in het invulveld doorvloeien naar het achterliggende object (Person) en omgekeerd
- De *InputText* tag zorgt voor een invulveld van het type *text*

EditPerson form - Validation

- Voeg een css stylesheet “*blazorValidation.css*” toe in de *wwwroot* folder:
 - Om validatiefouten mooi weer te kunnen geven in het formulier

```
.validation-errors {  
    background-color: red;  
    color: white;  
    padding: 8px;  
    text-align: center;  
    font-size: 16px;  
    font-weight: 500;  
}  
  
div.validation-message {  
    color: red;  
    font-weight: 500;  
}  
  
.modified.valid {  
    border: solid 3px green;  
}  
  
.modified.invalid {  
    border: solid 3px red;  
}
```

EditPerson form - Validation

- Stop validatie tags in het formulier:

```
<link href="css/blazorValidation.css" rel="stylesheet"/>
<h4 class="bg-primary text-white text-center p-2">...</h4>
<EditForm Model="Person">
  <DataAnnotationsValidator />
  <ValidationSummary />
  <div class="form-group">
    <label>ID</label>
    <InputNumber class="form-control" @bind-Value="Person.Id" disabled />
  </div>
  <div class="form-group">
    <label>Firstname</label>
    <ValidationMessage For="@(() => Person.Firstname)" />
    <InputText class="form-control" @bind-Value="Person.Firstname" />
  </div>
  <div class="form-group">
    <label>Surname</label>
    <ValidationMessage For="@(() => Person.Surname)" />
    <InputText class="form-control" @bind-Value="Person.Surname" />
  </div>
  <div class="form-group">
    <label>Department ID</label>
    <ValidationMessage For="@(() => Person.DepartmentId)" />
    <InputNumber class="form-control" @bind-Value="Person.DepartmentId" />
  </div>
  <div class="text-center">
    <button type="submit" class="btn btn-primary">Submit</button>
    <NavLink class="btn btn-secondary" href="/manage/people">Back</NavLink>
  </div>
</EditForm>
```

- DataAnnotationsValidator tag: activeert validatie in het formulier
- ValidationSummary tag: hier wordt een samenvatting van alle validatiefouten getoond
- ValidationMessage tag: hier wordt de validatiefout voor een bepaalde property van het model (Person) getoond

EditPerson form - Validation

- Voeg data annotaties toe aan de *Person* klasse:

```
using System.ComponentModel.DataAnnotations;

namespace PeopleApp.Data {

    public class Person {

        public long Id { get; set; }

        [Required(ErrorMessage = "A firstname is required")]
        [MinLength(3, ErrorMessage = "Firstnames must be 3 or more characters")]
        public string Firstname { get; set; }

        [Required(ErrorMessage = "A surname is required")]
        [MinLength(3, ErrorMessage = "Surnames must be 3 or more characters")]
        public string Surname { get; set; }

        public Department Department { get; set; }

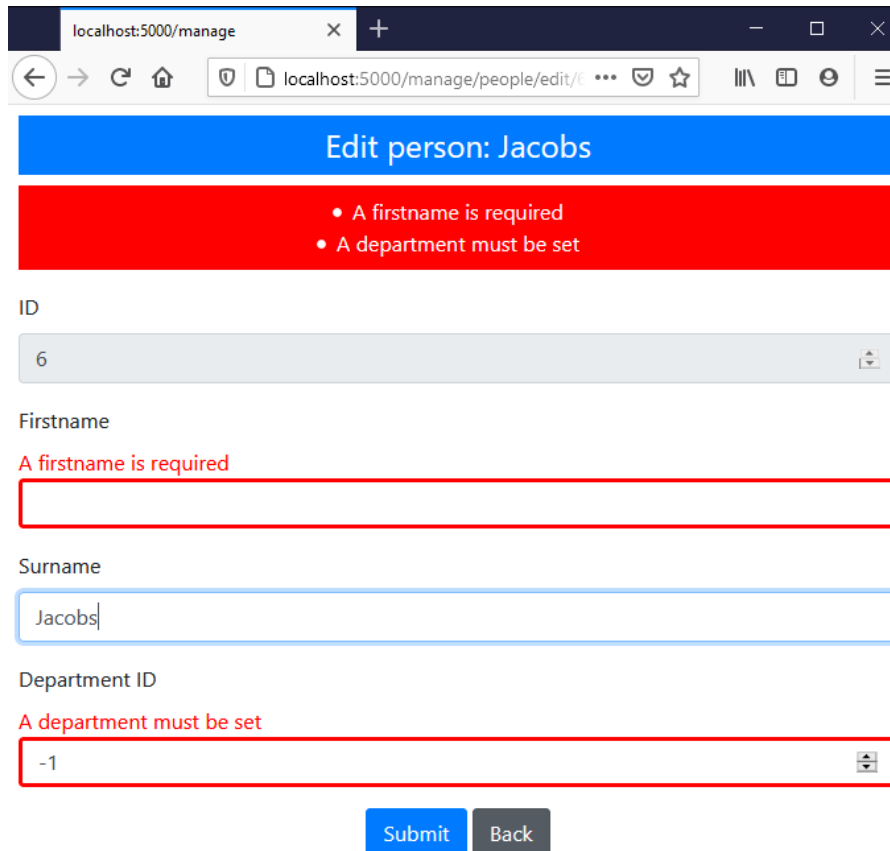
        [Range(1, long.MaxValue, ErrorMessage = "A department must be set")]
        public long DepartmentId { get; set; }

        public Location Location { get; set; }
        public long LocationId { get; set; }
    }
}
```

- Required: verplicht veld
- MinLength: veld moet minimaal uit een bepaald aantal karakters bestaan
- Range: waarde moet tussen een minimum en een maximum liggen
- ...

EditPerson form - Validation

- Start de applicatie en klik op de “Manage people” knop
- Klik op de *Edit* knop bij een person
- Geef enkele ongeldige waarden in



The screenshot shows a web browser window with the address bar displaying 'localhost:5000/manage'. The page title is 'Edit person: Jacobs'. A red error message box at the top of the form contains the following text:

- A firstname is required
- A department must be set

The form fields are as follows:

- ID:** A text input field containing the value '6'.
- Firstname:** A text input field that is empty. A red error message 'A firstname is required' is displayed above the field.
- Surname:** A text input field containing the value 'Jacobs'.
- Department ID:** A text input field containing the value '-1'. A red error message 'A department must be set' is displayed above the field.

At the bottom of the form, there are two buttons: 'Submit' (blue) and 'Back' (grey).

EditPerson form - Submit

- Bij het klikken op de submit knop willen we de person aanpassen in de database en vervolgens naar het overzicht van personen navigeren
 - Injecteer NavigationManager
 - Voorzie een HandleValidSubmit methode

```
@code {  
    public IPersonRepository PersonRepository;  
  
    override protected void OnInitialized()  
    {  
        PersonRepository = ScopedServices.GetRequiredService<IPersonRepository>();  
    }  
  
    [Inject]  
    public NavigationManager NavigationManager { get; set; }  
  
    [Parameter]  
    public long Id { get; set; }  
  
    public Person Person { get; set; }  
  
    protected override void OnParametersSet()  
    {  
        Person = PersonRepository.GetById(Id);  
    }  
  
    public void HandleValidSubmit()  
    {  
        PersonRepository.Update(Person);  
        NavigationManager.NavigateTo("/manage/people");  
    }  
}
```

EditPerson form - Submit

- Koppel de HandleValidSubmit methode aan de OnValidSubmit event van het formulier

```
@page "/manage/people/edit/{id:long}"
@using Microsoft.Extensions.DependencyInjection
@inherits OwningComponentBase
<link href="blazorValidation.css" rel="stylesheet"/>
<h4 class="bg-primary text-white text-center p-2">Edit person:
@Person.Firstname @Person.Surname</h4>
<EditForm Model="Person" OnValidSubmit="HandleValidSubmit">
    <DataAnnotationsValidator />
    <ValidationSummary />
    <div class="form-group">
...

```

EditPerson form - Submit

- Start de applicatie en klik op de “Manage people” knop
- Klik op de *Edit* knop bij een person
- Pas de persoon aan, klik op Submit -> Je wordt geleid naar het overzicht waarin je de aanpassing onmiddellijk kan zien

localhost:5000/manage

localhost:5000/manage/people/edit/1

Edit person: CharlesEDITED FuentesEDITED

ID

1

Firstname

CharlesEDITED

Surname

FuentesEDITED

Department ID

2

Submit Back

localhost:5000/manage

localhost:5000/manage/people

Manage People

Id	Name	Dept	Location	
1	FuentesEDITED, CharlesEDITED	Development	New York, NY	Edit
2	Lara, Murphy	Sales	New York, NY	Edit
3	Hoffman, Beasley	Facilities	New York, NY	Edit
4	Lloyd, Randall	Support	San Jose, CA	Edit
5	Case, Guzman	Development	San Jose, CA	Edit
6	Jacobs, Francesca	Sales	Oakland, CA	Edit
7	Becker, Bright	Facilities	Oakland, CA	Edit
8	Hays, Marks	Facilities	Oakland, CA	Edit
9	Trujillo, Underwood	Development	Oakland, CA	Edit
11	Hendrikx, Wesley	Development	New York, NY	Edit