Model ASP.NET MVC Core 3.1 Web App - Deel 2

1 Start project

In deel 1 zijn we gekomen tot deze code:

https://github.com/CSharpSyntraWest/PittigRestoMVC

We starten vanaf deze code

Dit hebben we in Deel 1 reeds gedaan tijdens de virtuele les:

- Aanmaken van ASP.Net Core Web applicatie met MVC architectuur design pattern
- opzetten configuratie Sql Server database , gebruik van ORM Entity FrameWork Core en migraties opzetten
- Opzetten Authenticatie en Authorisatie met ASP.Net Identity en beheren van Individuele User Accounts in SQL Server.
- Gebruik van LINQ to Entities om gegevens op te halen uit de database
- Asynchroon aangroepen van methoden
- Areas gebruiken en de routing configuratie aanpassen
- Aanmaken van model voor Category
- Aanmaken van controllers voor CRUD operaties op Category
- Aanmaken van Views voor Create/Read/Update/Delete (CRUD) van een Category in de database

In dit deel (Deel 2) gaan we het verder werken op het eerste deel

- Enkele Layout verbeteringen aanbrengen
- modellen, controllers en views toevoegen voor Subcategory, MenuItem, Coupon, ApplicationUser

1 Aanpassen layout footer van alle razor views

Zet een donkere achtergrond met lichtgekleurde links in de footer in de Razor views:

1. **Open Views\Shared_Layout.cshtml**
2. **Pas de class style aan in de <footer..> tag:**

```
<footer class="border-top footer text-muted">
    <div class="container">
          &copy; 2020 - PittigRestoMVC - <a asp-area="" asp-controller="Home" asp-
          action="Privacy">Privacy</a>
    </div>
     </footer>
```

Naar

```
<footer class="border-top pl-3 footer text-white-50" style="background-
color:#343a40">
```

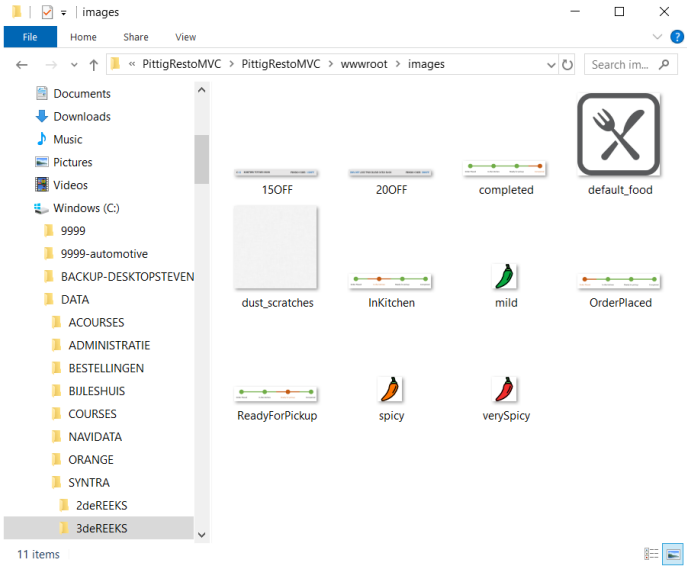
```
    <div class="container">
        &copy; 2020 - PittigRestoMVC - <a asp-area="" asp-controller="Home" asp-
        action="Privacy">Privacy</a>
    </div>
</footer>
```

Op deze website kan je een meer informatie vinden over de bootstrap styling mogelijkheden: https://getbootstrap.com/docs/4.1/components

Een website waar je mooie icons kan vinden: https://fontawesome.com/icons?d=gallery

2 Toevoegen van background image en vaste images voor web app

1. **Maak een nieuwe folder images onder de wwwroot folder van het project PittigRestoMVC**
2. **Download de images.zip van github en extract de images en kopieer deze onder de wwwroot/images**



3. **Voeg een style toe aan wwwroot/styles/site.css om de dust_scratches.png image te gebruiken als background image in al onze razor views:**

```
.search-background {

    background:url('/images/dust_scratches.png');
}
```

4. **Open Views\Shared_Layout.cshtml en gebruik en zet in de body tag class =“search-background”**

```
<body class="search-background">
```

3 Aanmaken models voor SubCategory

1. **Onder de folder models, voeg een nieuwe class toe voor SubCategory.**

```
public class SubCategory
{
    [Key]
    public int Id { get; set; }
```

```
        [Display(Name = "Naam Sub categorie")]
        [Required]
        public string Name { get; set; }

        [Required]
        [Display(Name ="Categorie")]
        public int CategoryId { get; set; }

        [ForeignKey("CategoryId")]
        public virtual Category Category { get; set; }

}
```

2. **Voeg een public Dbset<SubCategory> toe ApplicationDbContext**

```
public DbSet<SubCategory> SubCategory { get; set; }
```

3. **Voeg EF Core EntityFramework migrations toe voor subcategory via de Package Manager Console:**
   `PM> Add-Migration AddSubCategory`
4. **Update de database om de nieuwe tabel SubCategory toe te voegen aan de database (Package Manager console):**
   `PM> update-database`

4 Gemengde informatie in SubCategory razor views tonen via ViewModel class

We gaan een ViewModel aanmaken voor de SubCategory razor views gaan om gemengde informatie te tonen (ook bv de lijst van bestaande categories en subcategories, een statusmessage,…)

1. **Maak een nieuwe folder ViewModels onder de folder Models**
2. **Voeg een nieuwe class "SubCategoryAndCategoryViewModel" toe onder de folder Models/ViewModels**

```
namespace PittigRestoMVC.Models.ViewModels
{
    public class SubCategoryAndCategoryViewModel
    {
        public IEnumerable<Category> CategoryList { get; set; }
        public SubCategory SubCategory { get; set; }
        public List<string> SubCategoryList { get; set; }
        public string StatusMessage { get; set; }
    }
}
```

5 Dropdown list voorzien in SubCategory razor views (maakt gebruik van extension methods)

In de vorige lessen hebben we de dropdown ListItems naar de razor view doorgegeven via de ViewBag.
In dit voorbeeld gaan we dit doen op een andere manier: via de viewmodel voorzien we een List van items via een property. Het viewmodel geven we dan door aan de razor view.
Het omzetten van items van gelijk welk type naar SelectListItem type (voor de razor list items) maken we via een generieke extension method, zodat we dit kunnen hergebruiken voor alle soorten types (category, subcategory,..) in ons project

1. **Maak een nieuwe folder "Extensions" aan in PittigRestoMVC. We definiëren 2 classes** `IEnumerableExtension` **en** `ReflectionExtension`

   Elke class bevat een extension method:

2. **een extension method die een IEnumerable<T> items naar een** `IEnumerable<SelectListItem>` **omvormt:**
   (opm: de property Selected zal op true staan voor geselecteerde item(s) in de lijst op de razor view)

```
namespace PittigRestoMVC.Extensions
{
    public static class IEnumerableExtension
    {
        public static IEnumerable<SelectListItem> ToSelectListItem<T>(this
IEnumerable<T> items, int selectedValue)
        {
            return from item in items
                   select new SelectListItem
                   {
                       Text = item.GetPropertyValue("Name"),
                       Value = item.GetPropertyValue("Id"),
                       Selected =
item.GetPropertyValue("Id").Equals(selectedValue.ToString())
                   };
        }

    }
}
```

3. **De bovenstaande extension method maakt gebruik van de volgende extension method om de waarde (in string) op te vragen via de naam van een property van een item van type T. (Deze maakt gebruik van Reflection:**
   `item.GetType().GetProperty(propertyName).GetValue)`

```
namespace PittigRestoMVC.Extensions
{
    public static class ReflectionExtension
    {
        public static string GetPropertyValue<T>(this T item, string
propertyName)
        {
            return item.GetType().GetProperty(propertyName).GetValue(item,
null).ToString();
        }
    }
}
```

5. **Maak een SubCategoryController aan onder Areas/Admin/Controllers**
   We gaan de `SubCategoryAndCategoryViewModel` **gebruiken om de koppelen aan de razor view:**

```
namespace PittigRestoMVC.Areas.Admin.Controllers
{
    [Area("Admin")]
    public class SubCategoryController : Controller
    {
        private readonly ApplicationDbContext _db;

        public SubCategoryController(ApplicationDbContext context)
        {
            _db= context;
        }
        //GET - CREATE
        public async Task<IActionResult> Create()
        {
            SubCategoryAndCategoryViewModel model = new
SubCategoryAndCategoryViewModel()
            {
                CategoryList = await _db.Category.ToListAsync(),
                SubCategory = new Models.SubCategory(),
                SubCategoryList = await _db.SubCategory.OrderBy(p =>
p.Name).Select(p => p.Name).Distinct().ToListAsync()
            };
            return View(model);
         }

     }

}
```

6. **Maak een Create.cshtml razor view onder de folder Areas/Admin/Views/SubCategory**

6 Een Partial Razor View maken

Een Partial Razor View kan worden hergebruikt in andere views. We gaan een Partial Razor Views aanmaken voor o.a de “Back to list” om op alle create.cshtml razor views te tonen en een Partial razor view om een status message te tonen

1. **Maak een Partial Razor View “_CreateAndBackToListButton.cshtml” aan onder de folder Views/Shared/**

```
<div class="row">
    <div class="col-4">
        <input type="submit" class="btn btn-info form-control" value="Create" />
    </div>
    <div class="col-8">
        <a asp-action="Index" class="btn btn-success form-control">Terug naar
lijst</a>
    </div>
</div>
```

Maak een Partial Razor View “_StatusMessage.cshtml” aan onder de folder Views/Shared/

(opmerking: deze partial view is gekopieerd van

```
@model string

@if (!String.IsNullOrEmpty(Model))
{
    var statusMessageClass = Model.StartsWith("Error") ? "danger" : "success";
    <div class="alert alert-@statusMessageClass alert-dismissible" role="alert">
        <button type="button" class="close" data-dismiss="alert" aria-
label="Close"><span aria-hidden="true">&times;</span></button>
        @Model
    </div>
}
```

7 Verwijzen naar een Partial Razor View in een Razor view

We gaan verwijzen naar de Partial Razor views vanuit create.cshtml van Category en SubCategory

1. **Open de Razor view Areas/Admin/Views/Categories/create.cshtml en pas de razor code aan (deze code zal ook de layout wat aanpassen):**

```
@model Category
@{
    ViewData["Title"] = "Nieuw";
    Layout = "~/Views/Shared/_Layout.cshtml";
}

<br />
<h2 class="text-info">Aanmaken nieuwe categorie</h2>
<br />

<form method="post" asp-action="Create">
    <div class="border backgroundWhite">
        <div asp-validation-summary="ModelOnly" class="text-danger"></div>
        <div class="form-group row">
            <div class="col-2">
                <label asp-for="Name" class="col-form-label"></label>
            </div>
            <div class="col-5">
                <input asp-for="Name" class="form-control" />
            </div>
            <span asp-validation-for="Name" class="text-danger"></span>
        </div>
        <div class="form-group row">
            <div class="col-5 offset-2">
                <partial name="_CreateAndBackToListButton" />
            </div>
        </div>
    </div>
</form>

@section Scripts{
    @{ await Html.RenderPartialAsync("_ValidationScriptsPartial");}
}
```

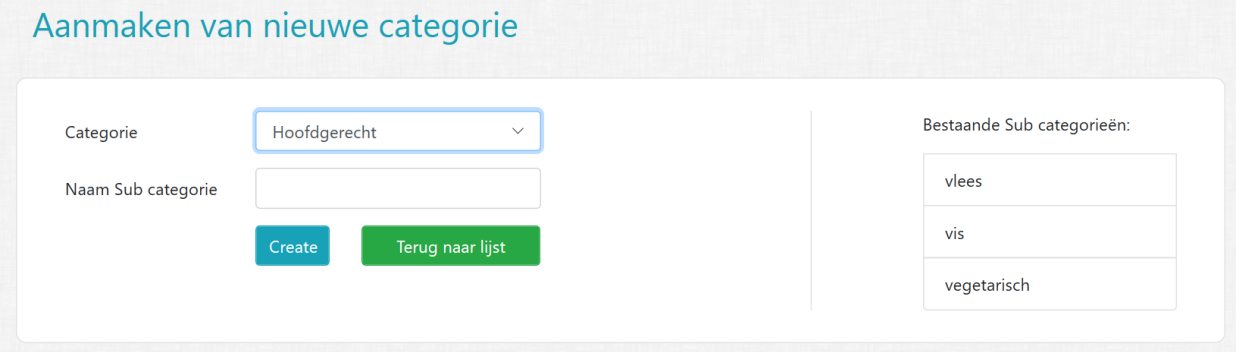
2. **Maak een nieuwe Razor view Areas/Admin/Views/SubCategories/create.cshtml en pas de razor code aan:**

```
@model PittigRestoMVC.Models.ViewModels.SubCategoryAndCategoryViewModel
@using PittigRestoMVC.Extensions

@{
    ViewData["Title"] = "Nieuw";
    Layout = "~/Views/Shared/_Layout.cshtml";
}

<br />
<h2 class="text-info">Aanmaken van nieuwe subcategorie</h2>
<br />

<partial name="_StatusMessage" model="Model.StatusMessage" />

<div class="border backgroundWhite row">
    <div class="col-8 border-right">
        <form asp-action="Create">
            <div asp-validation-summary="ModelOnly" class="text-danger"></div>
            <div class="form-group row">
                <div class="col-3">
                    <label asp-for="SubCategory.CategoryId" class="col-form-
label"></label>
                </div>
                <div class="col-5">
                    <select id="ddlCategorylist" asp-for="SubCategory.CategoryId"
asp-items="Model.CategoryList.ToSelectListItem(Model.SubCategory.CategoryId)"
class="form-control"></select>
                </div>
            </div>
            <div class="form-group row">
                <div class="col-3">
                    <label asp-for="SubCategory.Name" class="col-form-
label"></label>
                </div>
                <div class="col-5">
                    <input asp-for="SubCategory.Name" class="form-control" />
                </div>
                <span asp-validation-for="SubCategory.Name" class="text-
danger"></span>
            </div>
            <div class="form-group row">
                <div class="col-5 offset-3">
                    <partial name="_CreateAndBackToListButton" />
                </div>
            </div>
        </form>
    </div>
    <div class="col-3 offset-1">
        @if (Model.SubCategoryList.Count() > 0)
        {
            <p> Bestaande Sub categorieën: </p>

            <div id="SubCategoryList">
                <ul class="list-group"></ul>
            </div>
        }
    </div>
</div>
```

```
@section Scripts{
    @{ await Html.RenderPartialAsync("_ValidationScriptsPartial");}
```

8 Dynamisch de lijst van subcategorieën ophalen via javascript (Ajax)

In de SubCategory/Create.cshtml razor view willen we bij het selecteren van een Category in de dropdownlist (zonder refreshing van de browser pagina) de lijst van bestaande subcategorieën ophalen en tonen in de view:



Dit kan via javascript (een ajax asynchrone aanroep) vanuit de razor view.

De ajax call doet een http get request naar de web app. De web app voorzien een action method die de gegevens ophaalt uit de database en teruggeeft in json tekst.

1. **Maak de action method aan in de SubCategoriesController:**

```
[ActionName("GetSubCategory")]
public async Task<IActionResult> GetSubCategory(int id)
{
    List<SubCategory> subCategories = new List<SubCategory>();

    subCategories = await (from subCategory in _db.SubCategory
                           where subCategory.CategoryId == id
                           select subCategory).ToListAsync();
    return Json(new SelectList(subCategories, "Id", "Name"));
}
```

Voeg de javascript met de ajax call toe onderaan de razor view "Areas/Admin/Views/SubCategories/create.cshtml":

```
<script>function updateSubCategoryList() {
        var categorySelected = document.getElementById("ddlCategorylist").value;

        $list = $('#SubCategoryList');

        $.ajax({
            url: '/Admin/SubCategories/GetSubCategory/' + categorySelected,
            type: 'GET',
            dataType: 'text',
            success: function (data) {
                results = JSON.parse(data);
                $list.html('');
                $list.append(' <ul class="list-group"> ');
                for (i in results) {
```

```
                    $list.append('<li class="list-group-item">' + results[i].text +
'</li>');
                }
                $list.append('</ul>');
            }

        });

    }

    $(document).ready(function () {
        updateSubCategoryList();
    });

    $("#ddlCategorylist").on("change", function () {
        updateSubCategoryList();
    });</script>

    }
```