Model ASP.NET MVC Core 3.1 Web App - Deel 7

1 Constanten en utility methoden toevoegen

Voeg aan de static SD class in de folder Utilities de volgende constanten en methoden toe

```
using PittigRestoMVC.Models;
using System;


namespace PittigRestoMVC.Utility
{
    public static class SD
    {
        public const string DefaultFoodImage = "default_food.png";


        public const string ManagerUser = "Manager";
        public const string KitchenUser = "Kitchen";
        public const string FrontDeskUser = "FrontDesk";
        public const string CustomerEndUser = "Customer";

        public const string ssShoppingCartCount = "ssCartCount";
        public const string ssCouponCode = "ssCouponCode";

        public const string StatusSubmitted = "Verstuurd";
        public const string StatusInProcess = "In bereiding";
        public const string StatusReady = "Klaar voor ophaling";
        public const string StatusCompleted = "Afgehandeld";
        public const string StatusCancelled = "Geannuleerd";

        public const string PaymentStatusPending = "In Behandeling";
        public const string PaymentStatusApproved = "Goedgekeurd";
        public const string PaymentStatusRejected = "Afgewezen";

        public static string ConvertToRawHtml(string source)
        {
            char[] array = new char[source.Length];
            int arrayIndex = 0;
            bool inside = false;

            for (int i = 0; i < source.Length; i++)
            {
                char let = source[i];
                if (let == '<')
                {
                    inside = true;
                    continue;
                }
                if (let == '>')
                {
                    inside = false;
                    continue;
                }
                if (!inside)
                {
                    array[arrayIndex] = let;
                    arrayIndex++;
                }
            }
            return new string(array, 0, arrayIndex);
        }

        public static double DiscountedPrice(Coupon couponFromDb, double
OriginalOrderTotal)
        {
            if (couponFromDb == null)
            {
                return OriginalOrderTotal;
            }
            else
            {
                if (couponFromDb.MinimumAmount > OriginalOrderTotal)
                {
                    return OriginalOrderTotal;
                }
                else
                {
                    //everything is valid
                    if (Convert.ToInt32(couponFromDb.CouponType) ==
(int)Coupon.ECouponType.Euro)
                    {
                        //10 van 100
                        return Math.Round(OriginalOrderTotal -
couponFromDb.Discount, 2);
                    }
                    if (Convert.ToInt32(couponFromDb.CouponType) ==
(int)Coupon.ECouponType.Percent)
                    {
                        //10% van 100
                        return Math.Round(OriginalOrderTotal - (OriginalOrderTotal
* couponFromDb.Discount / 100), 2);
                    }
                }
            }
            return OriginalOrderTotal;
        }

    }
}
```

2 Emails Sturen

2.1 Email Service toevoegen en registreren als Singleton

1. **Maak een nieuwe folder ‘Services’ aan en maak hierin een class EmailOptions**

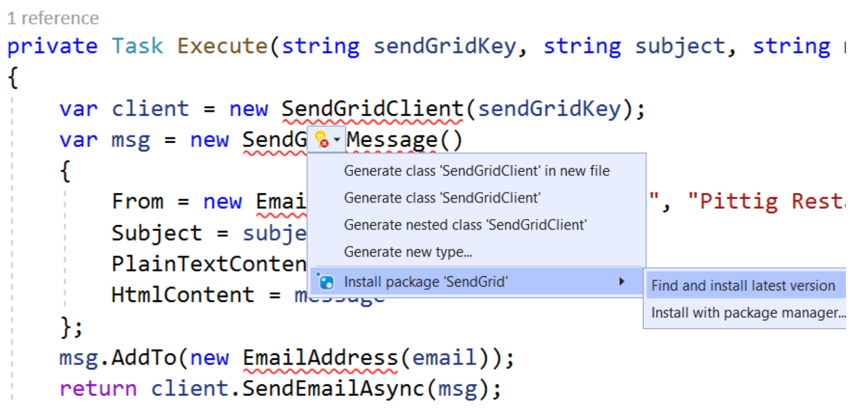
```
namespace PittigRestoMVC.Service
{
    public class EmailOptions
    {
        public string SendGridKey { get; set; }
    }
}
```

2. Maak onder de folder 'Services' een class EmailSender (afgeleide class van IEmailSender)

(Installeer de NuGet Package SendGrid)

```
using Microsoft.AspNetCore.Identity.UI.Services;
using Microsoft.Extensions.Options;
using SendGrid;
using SendGrid.Helpers.Mail;
using System.Threading.Tasks;

namespace PittigRestoMVC.Service
{
    public class EmailSender : IEmailSender
    {
        public EmailOptions Options { get; set; }

        public EmailSender(IOptions<EmailOptions> emailOptions)
        {
            Options = emailOptions.Value;
        }

        public Task SendEmailAsync(string email, string subject, string message)
        {
            return Execute(Options.SendGridKey, subject, message, email);
        }

        private Task Execute(string sendGridKey, string subject, string message,
string email)
        {
            var client = new SendGridClient(sendGridKey);
            var msg = new SendGridMessage()
            {
                From = new EmailAddress("admin@Pittig.com", "Pittig Restaurant"),
                Subject = subject,
                PlainTextContent = message,
                HtmlContent = message
            };
            msg.AddTo(new EmailAddress(email));
            return client.SendEmailAsync(msg);

        }
    }
}
```

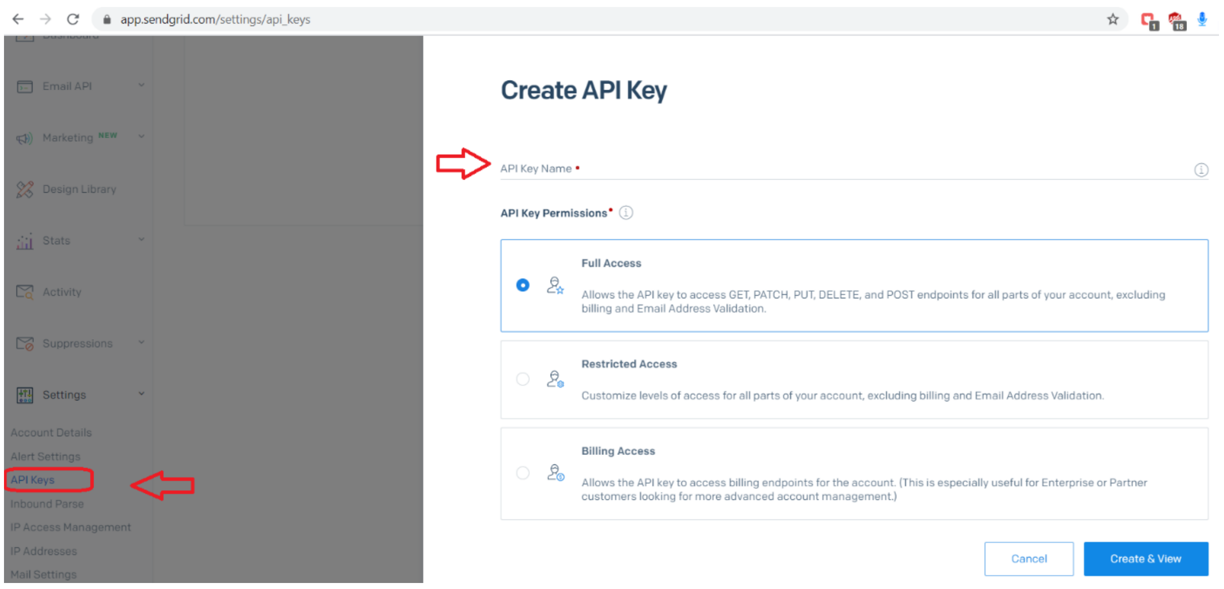
3. Installeer de laatste versie van NuGet Package SendGrid


```
1 reference
private Task Execute(string sendGridKey, string subject, string
{
    var client = new SendGridClient(sendGridKey);
    var msg = new SendG   Message()
    {
        From = new Emai                                     ", "Pittig Rest
        Subject = subje
        PlainTextConten
        HtmlContent = me
    };
    msg.AddTo(new EmailAddress(email));
    return client.SendEmailAsync(msg);
```
Generate class 'SendGridClient' in new file
Generate class 'SendGridClient'
Generate nested class 'SendGridClient'
Generate new type...
Install package 'SendGrid'
Find and install latest version
Install with package manager...

4. **Open Startup.cs en registreer in de methode ConfigureServices de EmailSender als Singleton**

```
public void ConfigureServices(IServiceCollection services)
{
    services.AddDbContext<ApplicationDbContext>(options =>
        options.UseSqlServer(
            Configuration.GetConnectionString("DefaultConnection")));

    services.AddIdentity<IdentityUser, IdentityRole>()
        .AddDefaultTokenProviders()
        .AddEntityFrameworkStores<ApplicationDbContext>();

    services.AddScoped<IDbInitializer, DbInitializer>();
    services.AddSingleton<IEmailSender, EmailSender>();
    services.Configure<EmailOptions>(Configuration);
    services.AddControllersWithViews();
    services.AddRazorPages().AddRazorRuntimeCompilation();
}
```

5. **Creatie van API Key voor SendGrid**
   Maak een Account aan op https://sendgrid.com/
   Klik onder Settings op API Keys
   Vul een Naam in (bv Syntra-West), selecteer de nodige permissies en Creëer de API Waarde

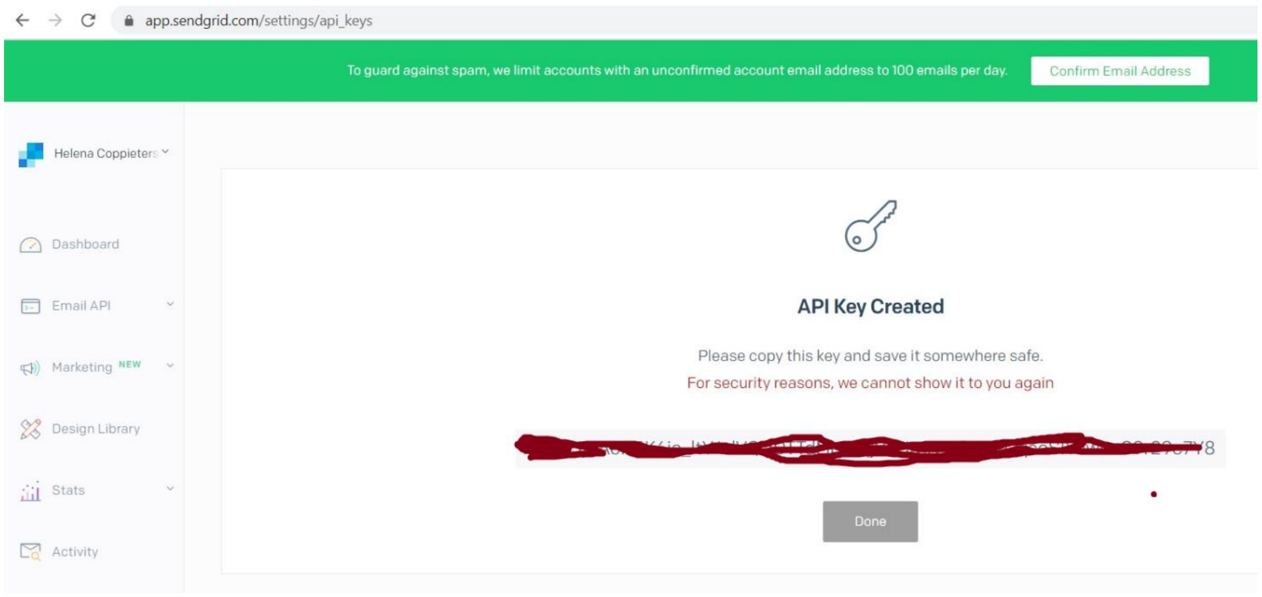


Create API Key

API Key Name *

API Key Permissions *

- Full Access — Allows the API key to access GET, PATCH, PUT, DELETE, and POST endpoints for all parts of your account, excluding billing and Email Address Validation.
- Restricted Access — Customize levels of access for all parts of your account, excluding billing and Email Address Validation.
- Billing Access — Allows the API key to access billing endpoints for the account. (This is especially useful for Enterprise or Partner customers looking for more advanced account management.)



To guard against spam, we limit accounts with an unconfirmed account email address to 100 emails per day.

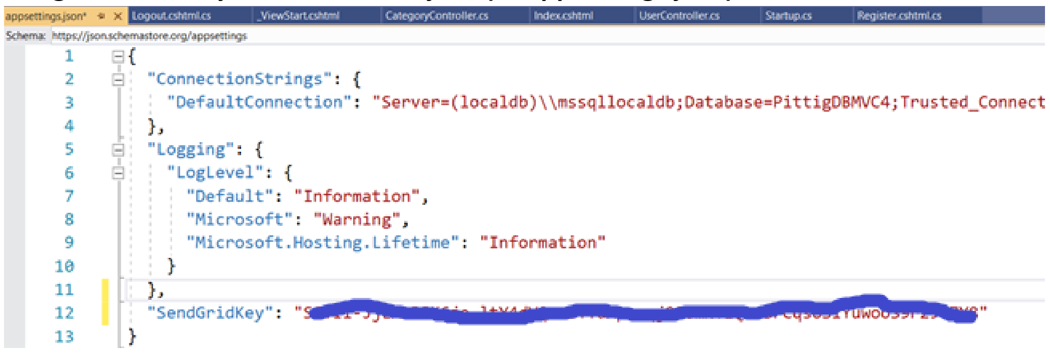
API Key Created

Please copy this key and save it somewhere safe.

For security reasons, we cannot show it to you again

[redacted]

6. Voeg deze API Key toe aan secrets.json (of appsettings.json)



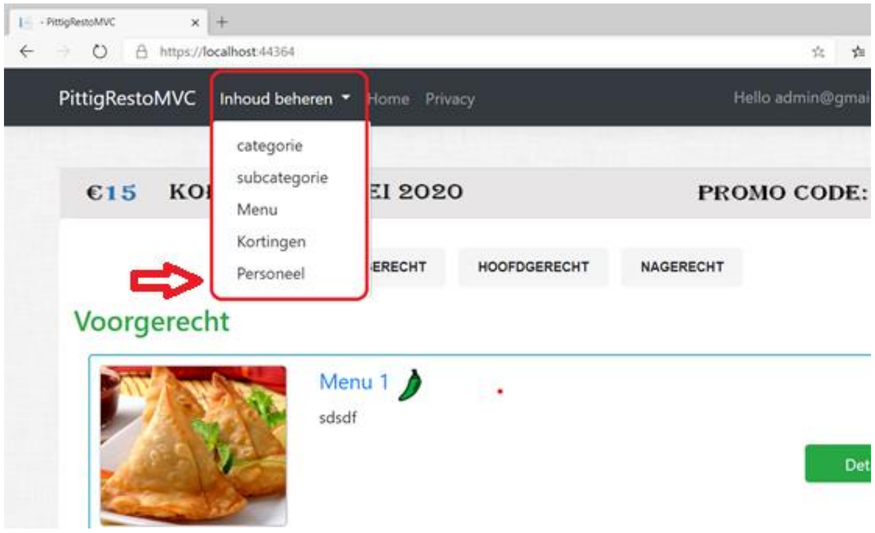
```
{
  "ConnectionStrings": {
    "DefaultConnection": "Server=(localdb)\\mssqllocaldb;Database=PittigDBMVC4;Trusted_Connect
  },
  "Logging": {
    "LogLevel": {
      "Default": "Information",
      "Microsoft": "Warning",
      "Microsoft.Hosting.Lifetime": "Information"
    }
  },
  "SendGridKey": "[redacted]"
}
```

appsettings.json

7. **Open Areas/Identity/Pages/Register.cshtml.cs (code behind) van Razor page Register**
   Pas de methode OnPostAsync aan :

```
        public async Task<IActionResult> OnPostAsync(string returnUrl = null)
        {
            string role = Request.Form["rdUserRole"].ToString();
            role = String.IsNullOrEmpty(role) ? SD.CustomerEndUser : role;
            returnUrl = returnUrl ?? Url.Content("~/");
            ExternalLogins = (await
_signInManager.GetExternalAuthenticationSchemesAsync()).ToList();
            if (ModelState.IsValid)
            {
                //var user = new IdentityUser { UserName = Input.Email, Email =
Input.Email };
                var user = new ApplicationUser
                {
                    UserName = Input.Email,
                    Email = Input.Email,
                    Name = Input.Name,
                    City = Input.City,
                    StreetAddress = Input.StreetAddress,
                    State = Input.State,
                    PostalCode = Input.PostalCode,
                    PhoneNumber = Input.PhoneNumber
                };
                var result = await _userManager.CreateAsync(user, Input.Password);
                if (result.Succeeded)
                {
                    _logger.LogInformation("User created a new account with
password.");
                    await _userManager.AddToRoleAsync(user, role);
                    if (role == SD.CustomerEndUser)
                    {
                        await _signInManager.SignInAsync(user, isPersistent: false);
                        return RedirectToPage("RegisterConfirmation", new { email =
Input.Email, returnUrl = returnUrl });
                        //return RedirectToAction("Index", "Home", new { area =
"Customer" });
                    }

                    //return RedirectToAction("Index", "User", new { area = "Admin"
});

                    var code = await
_userManager.GenerateEmailConfirmationTokenAsync(user);
                    var callbackUrl = Url.Page(
                        "/Account/ConfirmEmail",
                        pageHandler: null,
                        values: new { userId = user.Id, code = code },
                        protocol: Request.Scheme);
                    await _emailSender.SendEmailAsync(Input.Email, "Confirm your 
email",
                        $"Please confirm your account by <a 
href='{HtmlEncoder.Default.Encode(callbackUrl)}'>clicking here</a>.");

                        //if (_userManager.Options.SignIn.RequireConfirmedAccount)
                        //{
                        //    return RedirectToPage("RegisterConfirmation", new { 
email = Input.Email, returnUrl = returnUrl });
                        //}
                        //else
                        //{
                            await _signInManager.SignInAsync(user, isPersistent: 
false);
                            return LocalRedirect(returnUrl);
                        //}
                    }

                    foreach (var error in result.Errors)
                    {
                        ModelState.AddModelError(string.Empty, error.Description);
                    }
                }

                return Page();
            }
```

3 Personeel beheren



3.1 Aanmaken UserController

1. **Maak onder de folder Areas/Admin/Controllers een nieuwe lege MVC Controller aan met naam ‘UserController’:**

```
using System;
using System.Linq;
using System.Security.Claims;
using System.Threading.Tasks;
using Microsoft.AspNetCore.Authorization;
using Microsoft.AspNetCore.Mvc;
using Microsoft.EntityFrameworkCore;
using PittigRestoMVC.Data;
using PittigRestoMVC.Utility;

namespace PittigRestoMVC.Areas.Admin.Controllers
{
    [Area("Admin")]
    [Authorize(Roles = SD.ManagerUser)]
    public class UserController : Controller
    {
        private readonly ApplicationDbContext _db;

        public UserController(ApplicationDbContext db)
        {
            _db = db;
        }

        public async Task<IActionResult> Index()
        {
            var claimsIdentity = (ClaimsIdentity)this.User.Identity;
            var claim = claimsIdentity.FindFirst(ClaimTypes.NameIdentifier);

            return View(await _db.ApplicationUser.Where(u => u.Id != claim.Value).ToListAsync());
        }


        public async Task<IActionResult> Lock(string id)
        {
            if (id == null)
            {
                return NotFound();
            }

            var applicationUser = await _db.ApplicationUser.FirstOrDefaultAsync(m => m.Id == id);

            if (applicationUser == null)
            {
                return NotFound();
            }

            applicationUser.LockoutEnd = DateTime.Now.AddYears(100);

            await _db.SaveChangesAsync();

            return RedirectToAction(nameof(Index));
        }

        public async Task<IActionResult> UnLock(string id)
        {
            if (id == null)
            {
                return NotFound();
            }

            var applicationUser = await _db.ApplicationUser.FirstOrDefaultAsync(m => m.Id == id);

            if (applicationUser == null)
            {
                return NotFound();
            }

            applicationUser.LockoutEnd = DateTime.Now;

            await _db.SaveChangesAsync();

            return RedirectToAction(nameof(Index));
        }
    }
}
```

2. Maak onder de folder Areas/Admin/Views een nieuwe folder ‘User’ aan en daarin een Razor View ‘Index.cshtml’:

```
@model IEnumerable<ApplicationUser>
```

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

<br /><br />
<div class="border backgroundWhite">
    <div class="row">
        <div class="col-6">
            <h2 class="text-info"> Gebruikerslijst</h2>
        </div>
        <div class="col-6 text-right">
            <a asp-area="Identity" asp-page="/Account/Register" class="btn btn-info"><i 
class="fas fa-plus"></i> &nbsp; Nieuw Personeelslid</a>
        </div>
    </div>
    <br />
    <div>
        @if (Model.Count() > 0)
        {
            <table class="table table-striped border">
                <tr class="table-secondary">
                    <th>
                        @Html.DisplayNameFor(m => m.Name)
                    </th>
                    <th>
                        @Html.DisplayNameFor(m => m.Email)
                    </th>
                    <th>
                        @Html.DisplayNameFor(m => m.PhoneNumber)
                    </th>
                    <th></th>
                    <th></th>
                </tr>
                @foreach (var item in Model)
                {
                    <tr>
                        <td>
                            @Html.DisplayFor(m => item.Name)
                        </td>
                        <td>
                            @Html.DisplayFor(m => item.Email)
                        </td>
                        <td>
                            @Html.DisplayFor(m => item.PhoneNumber)
                        </td>
                        <td>

                            @if (item.LockoutEnd == null || item.LockoutEnd < 
DateTime.Now)
                            {
                            <a class="btn btn-success text-white" asp-action="Lock" 
asp-route-id="@item.Id">
                                <i class="fas fa-lock-open"></i>
                            </a>
                            }
                            else
                            {
                            <a class="btn btn-danger text-white" asp-action="UnLock" 
asp-route-id="@item.Id">
```

```
                                    <i class="fas fa-lock"></i>
                                </a>
                                }
                            </td>
                        </tr>
                    }
                </table>
            }
            else
            {
                <p>Er bestaan nog geen gebruikers...</p>
            }
        </div>
    </div>
```

4 Instellen van User Rollen toevoegen aan Register Razor pagina

1. **De Manager rol mag Users Registreren en een rol toekennen. Hiervoor wordt een radiobutton-list toegevoegd aan de Register.cshtml Razor page:**

Voeg ook een using directive (bovenaan) toe naar PittigRestoMVC.Utility namespace(voor het gebruiken van de class SD):

```
@page
@using PittigRestoMVC.Utility
@model RegisterModel


@{
    ViewData["Title"] = "Registreer";
}

<br />
<h2 class="text-info">Een nieuwe account aanmaken</h2>
<br />

<form method="post" asp-route-returnUrl="@Model.ReturnUrl">
    <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="Input.Name" class="col-form-label"></label>
            </div>
            <div class="col-5">
                <input asp-for="Input.Name" class="form-control" />
            </div>
            <span asp-validation-for="Input.Name" class="text-danger"></span>
        </div>
        <div class="form-group row">
            <div class="col-2">
                <label asp-for="Input.Email" class="col-form-label"></label>
            </div>
            <div class="col-5">
                <input asp-for="Input.Email" class="form-control" />
            </div>
            <span asp-validation-for="Input.Email" class="text-danger"></span>
        </div>
        <div class="form-group row">
            <div class="col-2">
                <label asp-for="Input.PhoneNumber" class="col-form-label"></label>
            </div>
            <div class="col-5">
                <input asp-for="Input.PhoneNumber" class="form-control" />
            </div>
            <span asp-validation-for="Input.PhoneNumber" class="text-danger"></span>
        </div>
        <div class="form-group row">
            <div class="col-2">
                <label asp-for="Input.StreetAddress" class="col-form-label"></label>
            </div>
            <div class="col-5">
                <input asp-for="Input.StreetAddress" class="form-control" />
            </div>
            <span asp-validation-for="Input.StreetAddress" class="text-danger"></span>
        </div>
        <div class="form-group row">
            <div class="col-2">
                <label asp-for="Input.City" class="col-form-label"></label>
            </div>
            <div class="col-5">
                <input asp-for="Input.City" class="form-control" />
            </div>
            <span asp-validation-for="Input.City" class="text-danger"></span>
        </div>
        <div class="form-group row">
            <div class="col-2">
                <label asp-for="Input.State" class="col-form-label"></label>
            </div>
            <div class="col-5">
                <input asp-for="Input.State" class="form-control" />
            </div>
            <span asp-validation-for="Input.State" class="text-danger"></span>
        </div>
        <div class="form-group row">
            <div class="col-2">
                <label asp-for="Input.PostalCode" class="col-form-label"></label>
            </div>
            <div class="col-5">
                <input asp-for="Input.PostalCode" class="form-control" />
            </div>
            <span asp-validation-for="Input.PostalCode" class="text-danger"></span>
        </div>

        <div class="form-group row">
            <div class="col-2">
                <label asp-for="Input.Password" class="col-form-label"></label>
            </div>
            <div class="col-5">
                <input asp-for="Input.Password" class="form-control" />
            </div>
            <span asp-validation-for="Input.Password" class="text-danger"></span>
        </div>

        <div class="form-group row">
            <div class="col-2">
                <label asp-for="Input.ConfirmPassword" class="col-form-label"></label>
            </div>
            <div class="col-5">
                <input asp-for="Input.ConfirmPassword" class="form-control" />
            </div>
            <span asp-validation-for="Input.ConfirmPassword" class="text-danger"></span>
        </div>
        @if (User.IsInRole(SD.ManagerUser))
        {
            <div class="form-group row">
                <div class="col-2">
                </div>
                <div class="col-5">
                    <input type="radio" name="rdUserRole" value="@SD.KitchenUser" checked /> Keuken
                    <input type="radio" name="rdUserRole" value="@SD.FrontDeskUser" checked /> Balie
                    <input type="radio" name="rdUserRole" value="@SD.ManagerUser" checked /> Manager
                </div>
            </div>
        }
        <div class="form-group row">
            <div class="col-5 offset-2">
                <button type="submit" class="btn btn-primary form-control">Registreer</button>
            </div>
        </div>
    </div>
</form>

@section Scripts {
    <partial name="_ValidationScriptsPartial" />
}
```

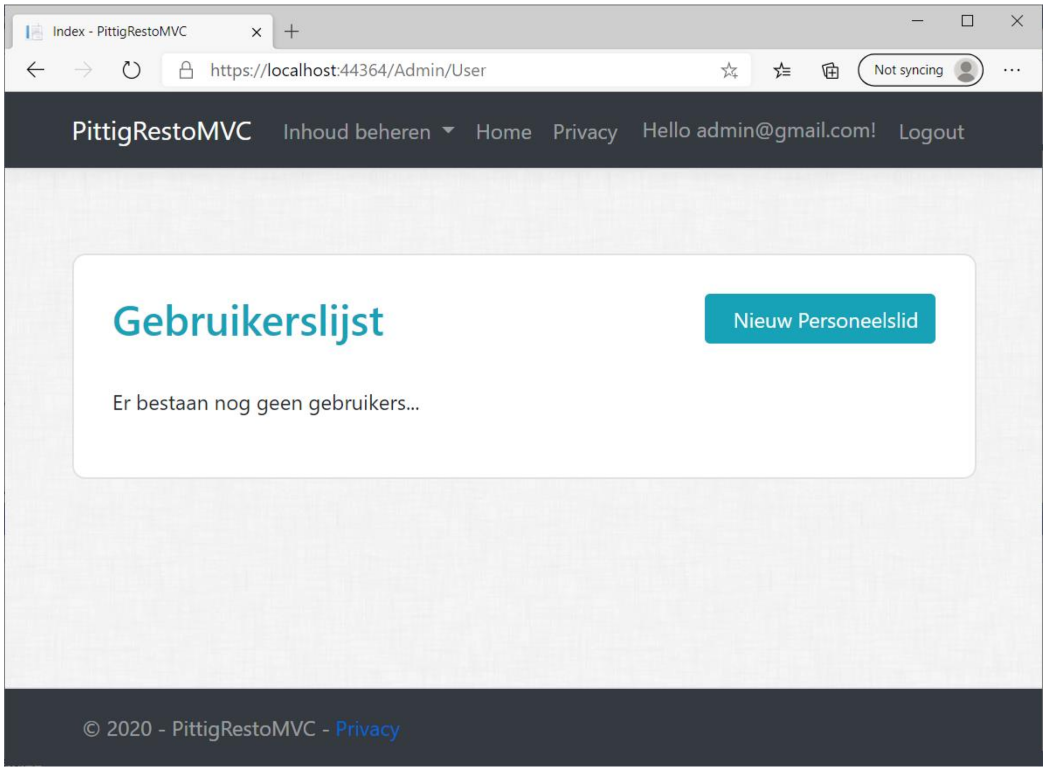
4.1 Toevoegen van Rollen aan Razor code behind:

1. **Open Register.cshtml.cs, de code behind van de Register Razor page en pas de methode** `OnPostAsync` **aan:**

```
using System;
using System.Collections.Generic;
using System.ComponentModel.DataAnnotations;
using System.Text.Encodings.Web;
using System.Threading.Tasks;
using Microsoft.AspNetCore.Authorization;
using Microsoft.AspNetCore.Identity;
using Microsoft.AspNetCore.Identity.UI.Services;
using Microsoft.AspNetCore.Mvc;
using Microsoft.AspNetCore.Mvc.RazorPages;
using Microsoft.Extensions.Logging;
using PittigRestoMVC.Models;
using PittigRestoMVC.Utility;

namespace PittigRestoMVC.Areas.Identity.Pages.Account
{
    [AllowAnonymous]
    public class RegisterModel : PageModel
    {
        private readonly SignInManager<IdentityUser> _signInManager;
        private readonly UserManager<IdentityUser> _userManager;
        private readonly ILogger<RegisterModel> _logger;
        private readonly IEmailSender _emailSender;
        private readonly RoleManager<IdentityRole> _roleManager;

        public RegisterModel(
            UserManager<IdentityUser> userManager,
            SignInManager<IdentityUser> signInManager,
            ILogger<RegisterModel> logger,
            IEmailSender emailSender,
            RoleManager<IdentityRole> roleManager)
        {
            _userManager = userManager;
            _signInManager = signInManager;
            _logger = logger;
            _emailSender = emailSender;
            _roleManager = roleManager;
        }

        [BindProperty]
        public InputModel Input { get; set; }
        public string ReturnUrl { get; set; }

        public class InputModel
        {
            [Required]
            [EmailAddress]
            [Display(Name = "Email")]
            public string Email { get; set; }

            [Required]
            [StringLength(100, ErrorMessage = "Het {0} moet minstens {2} en
maximimaal {1} tekens bevatten.", MinimumLength = 6)]
            [DataType(DataType.Password)]
            [Display(Name = "Wachtwoord")]
            public string Password { get; set; }

            [DataType(DataType.Password)]
            [Display(Name = "Bevestig Wachtwoord")]
            [Compare("Password", ErrorMessage = "Het wachtwoord en bevestiging van
wachtwoord komen niet overeen")]
            public string ConfirmPassword { get; set; }

            [Required]
            [Display(Name = "Naam")]
            public string Name { get; set; }
            [Display(Name = "Straat")]
            public string StreetAddress { get; set; }
            [Display(Name = "TelNr")]
            public string PhoneNumber { get; set; }
            [Display(Name = "Gemeente")]
            public string City { get; set; }
            [Display(Name = "Provincie")]
            public string State { get; set; }
            [Display(Name = "Postcode")]
            public string PostalCode { get; set; }

        }

        public void OnGet(string returnUrl = null)
        {
            ReturnUrl = returnUrl;
        }

        public async Task<IActionResult> OnPostAsync(string returnUrl = null)
        {
            string role = Request.Form["rdUserRole"].ToString();
            role = String.IsNullOrEmpty(role) ? SD.CustomerEndUser : role;
            if (ModelState.IsValid)
            {
                var user = new ApplicationUser
                {
                    UserName = Input.Email,
                    Email = Input.Email,
                    Name = Input.Name,
                    City = Input.City,
                    StreetAddress = Input.StreetAddress,
                    State = Input.State,
                        PostalCode = Input.PostalCode,
                        PhoneNumber = Input.PhoneNumber
                    };
                    var result = await _userManager.CreateAsync(user, Input.Password);

                    if (result.Succeeded)
                    {
                        _logger.LogInformation("User created a new account with 
password.");

                        await _userManager.AddToRoleAsync(user, role);
                        if (role == SD.CustomerEndUser)
                        {
                            await _signInManager.SignInAsync(user, isPersistent: 
false);
                            return RedirectToAction("Index", "Home", new { area = 
"Customer" });
                        }

                        return RedirectToAction("Index", "User", new { area = "Admin" 
});

                        //var code = await 
_userManager.GenerateEmailConfirmationTokenAsync(user);
                        //var callbackUrl = Url.Page(
                        //    "/Account/ConfirmEmail",
                        //    pageHandler: null,
                        //    values: new { userId = user.Id, code = code },
                        //    protocol: Request.Scheme);

                        //await _emailSender.SendEmailAsync(Input.Email, "Confirm your 
email",
                        //    $"Please confirm your account by <a 
href='{HtmlEncoder.Default.Encode(callbackUrl)}'>clicking here</a>.");

                    }
                    foreach (var error in result.Errors)
                    {
                        ModelState.AddModelError(string.Empty, error.Description);
                    }
                }

                return Page();
            }
        }
}
```

4.2 Test de app uit:

1.log in als admin@gmail.com en open de Personeels-pagina:


Index - PittigRestoMVC

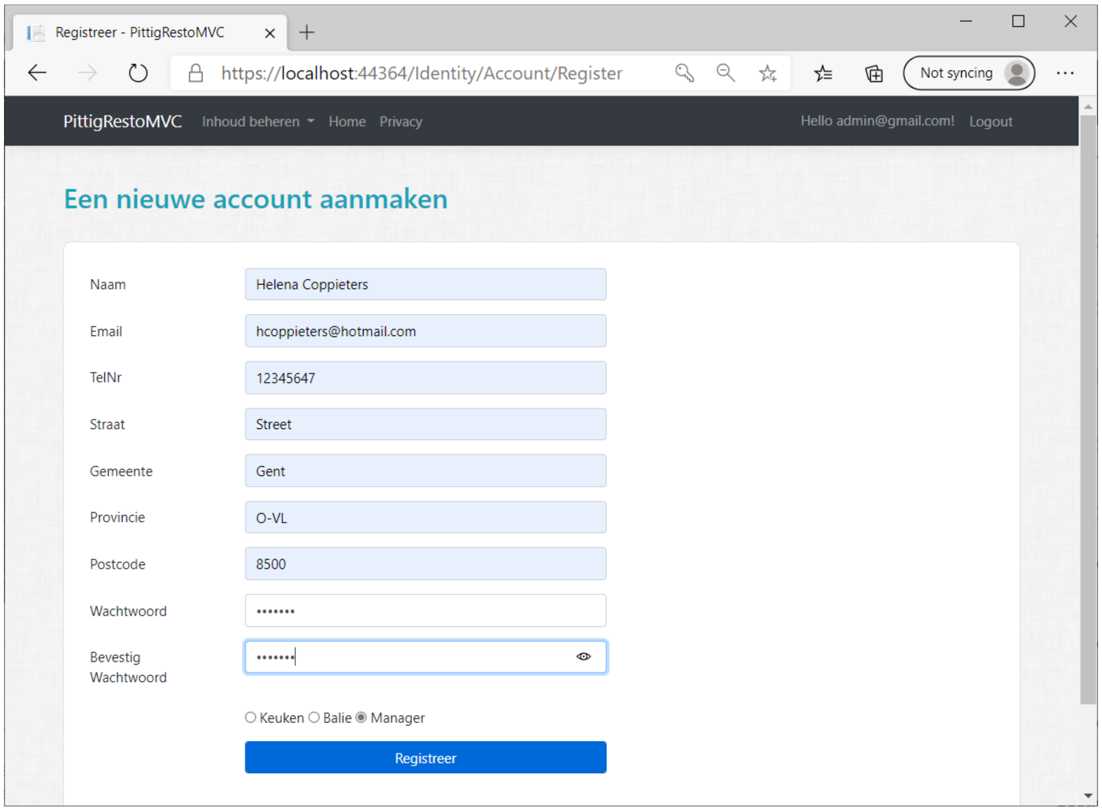
https://localhost:44364/Admin/User

Gebruikerslijst

Er bestaan nog geen gebruikers...

© 2020 - PittigRestoMVC - Privacy

2. **Klik op de button "Nieuw personeelslid" en voeg een paar gebruikers toe met verschillende rol bv een user met Manager-rol en user met Balie-rol:**

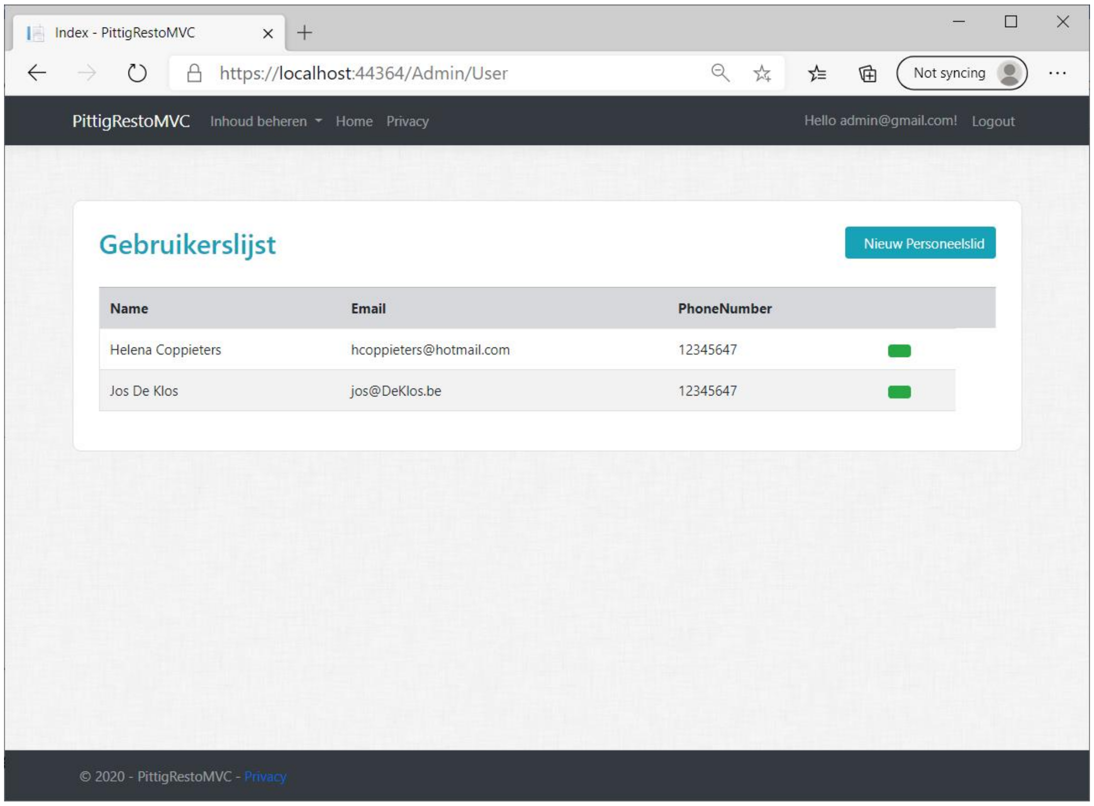

Registreer - PittigRestoMVC

https://localhost:44364/Identity/Account/Register

Een nieuwe account aanmaken

Naam	Helena Coppieters
Email	hcoppieters@hotmail.com
TelNr	12345647
Straat	Street
Gemeente	Gent
Provincie	O-VL
Postcode	8500
Wachtwoord	•••••••
Bevestig Wachtwoord	•••••••

Keuken Balie [✓ Manager]

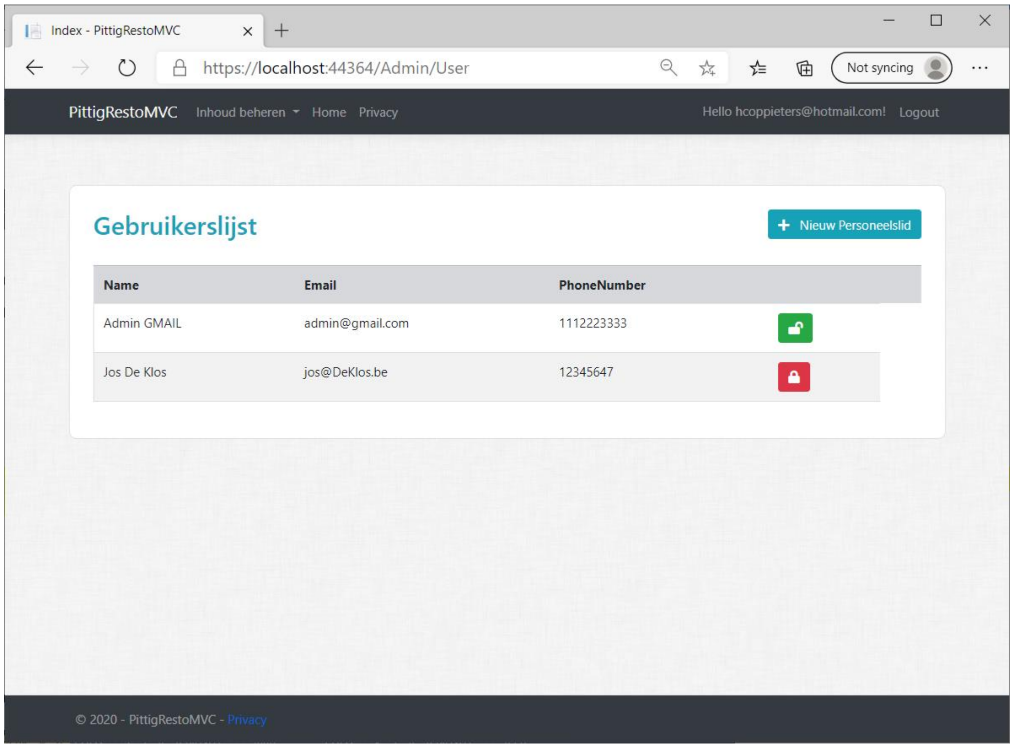


Gebruikerslijst

Name	Email	PhoneNumber	
Helena Coppieters	hcoppieters@hotmail.com	12345647	🟩
Jos De Klos	jos@DeKlos.be	12345647	🟩

3. De lock-icoontjes, kunnen bv van de website font-awesome worden gehaald: Open Views/Shared/_Layout.cshtml en voeg hiervoor de volgende lijn toe aan de head:

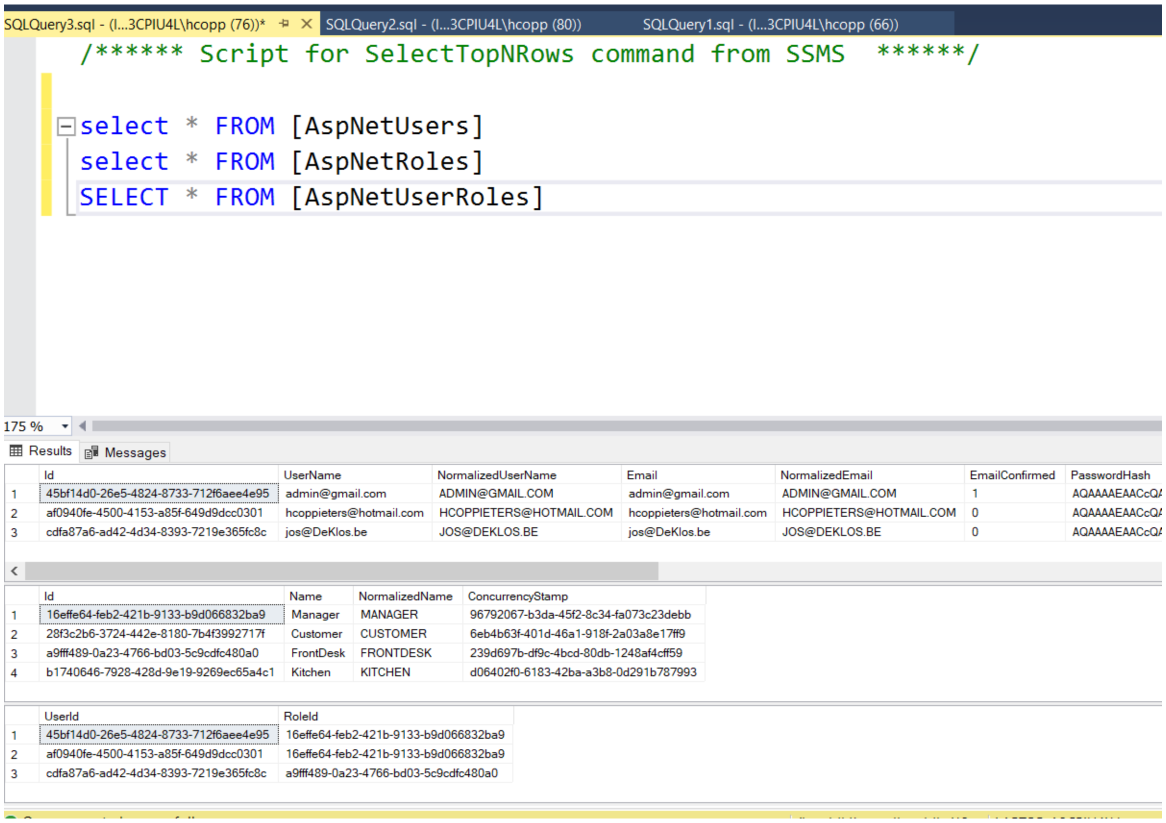
```
<head>
    <meta charset="utf-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <title>@ViewData["Title"] - PittigRestoMVC</title>
    <link rel="stylesheet" href="~/lib/bootstrap/dist/css/bootstrap.min.css" />
    <link rel="stylesheet"
href="https://use.fontawesome.com/releases/v5.5.0/css/all.css" integrity="sha384-
B4dIYHKNBt8Bc12p+WXckhzcICo0wtJAoU8YZTY5qE0Id1GSseTk6S+L3BlXeVIU"
crossorigin="anonymous">
    <link rel="stylesheet" href="~/css/site.css" />
</head>
```



Gebruikerslijst

Name	Email	PhoneNumber	
Admin GMAIL	admin@gmail.com	1112223333	🔓
Jos De Klos	jos@DeKlos.be	12345647	🔒

4. Controleer in de database of de users en hun rollen correct zijn aangemaakt:


SQLQuery3.sql - (l...3CPIU4L\hcopp (76))*
SQLQuery2.sql - (l...3CPIU4L\hcopp (80))
SQLQuery1.sql - (l...3CPIU4L\hcopp (66))
175 %
Results
Messages

```
/****** Script for SelectTopNRows command from SSMS  ******/

select * FROM [AspNetUsers]
select * FROM [AspNetRoles]
SELECT * FROM [AspNetUserRoles]
```

	Id	UserName	NormalizedUserName	Email	NormalizedEmail	EmailConfirmed	PasswordHash
1	45bf14d0-26e5-4824-8733-712f6aee4e95	admin@gmail.com	ADMIN@GMAIL.COM	admin@gmail.com	ADMIN@GMAIL.COM	1	AQAAAAEAACcQA
2	af0940fe-4500-4153-a85f-649d9dcc0301	hcoppieters@hotmail.com	HCOPPIETERS@HOTMAIL.COM	hcoppieters@hotmail.com	HCOPPIETERS@HOTMAIL.COM	0	AQAAAAEAACcQA
3	cdfa87a6-ad42-4d34-8393-7219e365fc8c	jos@DeKlos.be	JOS@DEKLOS.BE	jos@DeKlos.be	JOS@DEKLOS.BE	0	AQAAAAEAACcQA

	Id	Name	NormalizedName	ConcurrencyStamp
1	16effe64-feb2-421b-9133-b9d066832ba9	Manager	MANAGER	96792067-b3da-45f2-8c34-fa073c23debb
2	28f3c2b6-3724-442e-8180-7b4f3992717f	Customer	CUSTOMER	6eb4b63f-401d-46a1-918f-2a03a8e17ff9
3	a9fff489-0a23-4766-bd03-5c9cdfc480a0	FrontDesk	FRONTDESK	239d697b-df9c-4bcd-80db-1248af4cff59
4	b1740646-7928-428d-9e19-9269ec65a4c1	Kitchen	KITCHEN	d06402f0-6183-42ba-a3b8-0d291b787993

	UserId	RoleId
1	45bf14d0-26e5-4824-8733-712f6aee4e95	16effe64-feb2-421b-9133-b9d066832ba9
2	af0940fe-4500-4153-a85f-649d9dcc0301	16effe64-feb2-421b-9133-b9d066832ba9
3	cdfa87a6-ad42-4d34-8393-7219e365fc8c	a9fff489-0a23-4766-bd03-5c9cdfc480a0

5 Models toevoegen voor Bestellingen

1. Voeg onder de folder models een class voor OrderHeader toe

```
using PittigRestoMVC.Models;
using System;
using System.ComponentModel.DataAnnotations;
using System.ComponentModel.DataAnnotations.Schema;
namespace PittigRestoMVC.Models
{
    public class OrderHeader
    {
        [Key]
        public int Id { get; set; }

        [Required]
        public string UserId { get; set; }

        [ForeignKey("UserId")]
        public virtual ApplicationUser ApplicationUser { get; set; }

        [Required]
        [Display(Name = "Datum bestelling")]
        public DateTime OrderDate { get; set; }

        [Required]
        public double OrderTotalOriginal { get; set; }

        [Required]
        [DisplayFormat(DataFormatString = "{0:00}")]
        [Display(Name = "Totaal bestelling")]
        public double OrderTotal { get; set; }

        [Required]
        [Display(Name = "Tijdstip ophaling")]
        public DateTime PickUpTime { get; set; }

        [Required]
        [NotMapped]
        [Display(Name = "Datum ophaling")]
        public DateTime PickUpDate { get; set; }

        [Display(Name = "Kortingscode")]
        public string CouponCode { get; set; }
        [Display(Name = "Korting")]
        public double CouponCodeDiscount { get; set; }
        public string Status { get; set; }
        [Display(Name = "Status betaling")]
        public string PaymentStatus { get; set; }
        [Display(Name = "Commentaar")]
        public string Comments { get; set; }


        [Display(Name = "Naam ophaler")]
        public string PickupName { get; set; }
```

```
            [Display(Name = "TelNr")]
            public string PhoneNumber { get; set; }

            public string TransactionId { get; set; }
        }
    }
```

2. **Voeg onder de folder models eveneens een class voor OrderDetails toe**

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

    [Required]
    public int OrderId { get; set; }

    [ForeignKey("OrderId")]
    public virtual OrderHeader OrderHeader { get; set; }

    [Required]
    public int MenuItemId { get; set; }

    [ForeignKey("MenuItemId")]
    public virtual MenuItem MenuItem { get; set; }

    public int Count { get; set; }

    public string Name { get; set; }
    public string Description { get; set; }

    [Required]
    public double Price { get; set; }

}
```

3. **Voeg aan ApplicationDbContext.cs een Properties toe voor de DbSet van OrderHeader en OrderDetails items**

```
        public DbSet<OrderHeader> OrderHeader { get; set; }
        public DbSet<OrderDetails> OrderDetails { get; set; }
```

4. **Voeg in Package manager console een Migration toe voor het aanmaken van de OrderHeader en OrderDetails tabellen in de database**

```
PM> Add-Migration AddOrderTablesToDb
PM> update-database
```

6 Winkelwagen toevoegen

6.1 Aanmaken model voor Winkelwagen

5. **Voeg onder de folder models een nieuwe class toe**

```
public class ShoppingCart
 {
     public ShoppingCart()
```

```
        {
            Count = 1;
        }


        public int Id { get; set; }

        public string ApplicationUserId { get; set; }

        [NotMapped]
        [ForeignKey("ApplicationUserId")]
        public virtual ApplicationUser ApplicationUser { get; set; }

        public int MenuItemId { get; set; }

        [NotMapped]
        [ForeignKey("MenuItemId")]
        public virtual MenuItem MenuItem { get; set; }



        [Range(1,int.MaxValue, ErrorMessage ="Geef een waarde van minstens {1} aub")]
        public int Count { get; set; }
    }
```

6. **Voeg aan ApplicationDbContext.cs een Property toe voor de DbSet van ShoppingCart items**

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

7. **Voeg in Package manager console een Migration toe voor het aanmaken van de ShoppingCart tabel in de database**

```
PM> Add-Migration AddShoppingCartToDb
PM> update-database
```

6.2 ViewModel toevoegen voor Winkelwagen razor page

Maak onder de folder Models/ViewModels een class OrderDetailsCart aan:

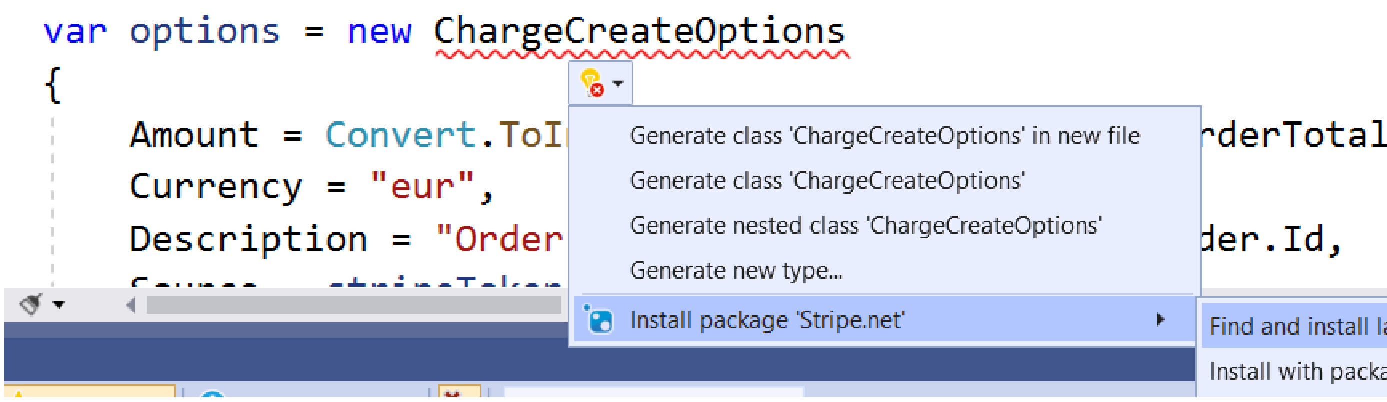
```
public class OrderDetailsCart
{
    public List<ShoppingCart> listCart { get; set; }
    public OrderHeader OrderHeader { get; set; }
}
```

6.3 Cart Controller toevoegen

Maak onder de folder Areas/Customer/Controllers een Lege MVC Controller toe met naam ‘CartController’

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Security.Claims;
using System.Threading.Tasks;
using Microsoft.AspNetCore.Http;
using Microsoft.AspNetCore.Identity.UI.Services;
using Microsoft.AspNetCore.Mvc;
using Microsoft.EntityFrameworkCore;
using PittigRestoMVC.Data;
using PittigRestoMVC.Models;
using PittigRestoMVC.Models.ViewModels;
using PittigRestoMVC.Utility;
using Stripe;

namespace PittigRestoMVC.Areas.Customer.Controllers
{
    [Area("Customer")]
    public class CartController : Controller
    {
        private readonly ApplicationDbContext _db;
        private readonly IEmailSender _emailSender;

        [BindProperty]
        public OrderDetailsCart detailCart { get; set; }

        public CartController(ApplicationDbContext db, IEmailSender emailSender)
        {
            _db = db;
            _emailSender = emailSender;
        }

        public async Task<IActionResult> Index()
        {

            detailCart = new OrderDetailsCart()
            {
                OrderHeader = new Models.OrderHeader()
            };

            detailCart.OrderHeader.OrderTotal = 0;

            var claimsIdentity = (ClaimsIdentity)User.Identity;
            var claim = claimsIdentity.FindFirst(ClaimTypes.NameIdentifier);

            var cart = _db.ShoppingCart.Where(c => c.ApplicationUserId == claim.Value);
            if (cart != null)
            {
                detailCart.listCart = cart.ToList();
            }

            foreach (var list in detailCart.listCart)
            {
                list.MenuItem = await _db.MenuItem.FirstOrDefaultAsync(m => m.Id == list.MenuItemId);
                detailCart.OrderHeader.OrderTotal = detailCart.OrderHeader.OrderTotal + 
(list.MenuItem.Price * list.Count);
                list.MenuItem.Description = SD.ConvertToRawHtml(list.MenuItem.Description);
                if (list.MenuItem.Description.Length > 100)
                {
                    list.MenuItem.Description = list.MenuItem.Description.Substring(0, 99) + "...";
                }
            }
            detailCart.OrderHeader.OrderTotalOriginal = detailCart.OrderHeader.OrderTotal;

            if (HttpContext.Session.GetString(SD.ssCouponCode) != null)
            {
                detailCart.OrderHeader.CouponCode = HttpContext.Session.GetString(SD.ssCouponCode);
                var couponFromDb = await _db.Coupon.Where(c => c.Name.ToLower() == 
detailCart.OrderHeader.CouponCode.ToLower()).FirstOrDefaultAsync();
                detailCart.OrderHeader.OrderTotal = SD.DiscountedPrice(couponFromDb, 
detailCart.OrderHeader.OrderTotalOriginal);
            }


            return View(detailCart);

        }


        public async Task<IActionResult> Summary()
        {

            detailCart = new OrderDetailsCart()
            {
                OrderHeader = new Models.OrderHeader()
            };

            detailCart.OrderHeader.OrderTotal = 0;

            var claimsIdentity = (ClaimsIdentity)User.Identity;
            var claim = claimsIdentity.FindFirst(ClaimTypes.NameIdentifier);
            ApplicationUser applicationUser = await _db.ApplicationUser.Where(c => c.Id ==
claim.Value).FirstOrDefaultAsync();
            var cart = _db.ShoppingCart.Where(c => c.ApplicationUserId == claim.Value);
            if (cart != null)
            {
                detailCart.listCart = cart.ToList();
            }

            foreach (var list in detailCart.listCart)
            {
                list.MenuItem = await _db.MenuItem.FirstOrDefaultAsync(m => m.Id == list.MenuItemId);
                detailCart.OrderHeader.OrderTotal = detailCart.OrderHeader.OrderTotal +
(list.MenuItem.Price * list.Count);

            }
            detailCart.OrderHeader.OrderTotalOriginal = detailCart.OrderHeader.OrderTotal;
            detailCart.OrderHeader.PickupName = applicationUser.Name;
            detailCart.OrderHeader.PhoneNumber = applicationUser.PhoneNumber;
            detailCart.OrderHeader.PickUpTime = DateTime.Now;


            if (HttpContext.Session.GetString(SD.ssCouponCode) != null)
            {
                detailCart.OrderHeader.CouponCode = HttpContext.Session.GetString(SD.ssCouponCode);
                var couponFromDb = await _db.Coupon.Where(c => c.Name.ToLower() ==
detailCart.OrderHeader.CouponCode.ToLower()).FirstOrDefaultAsync();
                detailCart.OrderHeader.OrderTotal = SD.DiscountedPrice(couponFromDb,
detailCart.OrderHeader.OrderTotalOriginal);
            }

            return View(detailCart);
        }


        [HttpPost]
        [ValidateAntiForgeryToken]
        [ActionName("Summary")]
        public async Task<IActionResult> SummaryPost(string stripeToken)
        {
            var claimsIdentity = (ClaimsIdentity)User.Identity;
            var claim = claimsIdentity.FindFirst(ClaimTypes.NameIdentifier);


            detailCart.listCart = await _db.ShoppingCart.Where(c => c.ApplicationUserId ==
claim.Value).ToListAsync();

            detailCart.OrderHeader.PaymentStatus = SD.PaymentStatusPending;
            detailCart.OrderHeader.OrderDate = DateTime.Now;
            detailCart.OrderHeader.UserId = claim.Value;
            detailCart.OrderHeader.Status = SD.PaymentStatusPending;
            detailCart.OrderHeader.PickUpTime =
Convert.ToDateTime(detailCart.OrderHeader.PickUpDate.ToShortDateString() + " " +
detailCart.OrderHeader.PickUpTime.ToShortTimeString());

            List<OrderDetails> orderDetailsList = new List<OrderDetails>();
            _db.OrderHeader.Add(detailCart.OrderHeader);
            await _db.SaveChangesAsync();
            detailCart.OrderHeader.OrderTotalOriginal = 0;


            foreach (var item in detailCart.listCart)
            {
                item.MenuItem = await _db.MenuItem.FirstOrDefaultAsync(m => m.Id == item.MenuItemId);
                OrderDetails orderDetails = new OrderDetails
                {
                    MenuItemId = item.MenuItemId,
                    OrderId = detailCart.OrderHeader.Id,
                    Description = item.MenuItem.Description,
                    Name = item.MenuItem.Name,
                    Price = item.MenuItem.Price,
                    Count = item.Count
                };
                detailCart.OrderHeader.OrderTotalOriginal += orderDetails.Count * orderDetails.Price;
                _db.OrderDetails.Add(orderDetails);

            }

            if (HttpContext.Session.GetString(SD.ssCouponCode) != null)
            {
                detailCart.OrderHeader.CouponCode = HttpContext.Session.GetString(SD.ssCouponCode);
                var couponFromDb = await _db.Coupon.Where(c => c.Name.ToLower() ==
detailCart.OrderHeader.CouponCode.ToLower()).FirstOrDefaultAsync();
                detailCart.OrderHeader.OrderTotal = SD.DiscountedPrice(couponFromDb,
detailCart.OrderHeader.OrderTotalOriginal);
            }
            else
            {
                detailCart.OrderHeader.OrderTotal = detailCart.OrderHeader.OrderTotalOriginal;
            }
            detailCart.OrderHeader.CouponCodeDiscount = detailCart.OrderHeader.OrderTotalOriginal -
detailCart.OrderHeader.OrderTotal;

            _db.ShoppingCart.RemoveRange(detailCart.listCart);
            HttpContext.Session.SetInt32(SD.ssShoppingCartCount, 0);
            await _db.SaveChangesAsync();

            var options = new ChargeCreateOptions
            {
                Amount = Convert.ToInt32(detailCart.OrderHeader.OrderTotal * 100),
                Currency = "eur",
                Description = "Order ID : " + detailCart.OrderHeader.Id,
                Source = stripeToken

            };
            var service = new ChargeService();
            Charge charge = service.Create(options);

            if (charge.BalanceTransactionId == null)
            {
                detailCart.OrderHeader.PaymentStatus = SD.PaymentStatusRejected;
            }
            else
            {
                detailCart.OrderHeader.TransactionId = charge.BalanceTransactionId;
            }

            if (charge.Status.ToLower() == "succeeded")
            {
                detailCart.OrderHeader.PaymentStatus = SD.PaymentStatusApproved;
                detailCart.OrderHeader.Status = SD.StatusSubmitted;
            }
            else
            {
                detailCart.OrderHeader.PaymentStatus = SD.PaymentStatusRejected;
            }
            await _db.SaveChangesAsync();
            return RedirectToAction("Index", "Home");

        }


        public IActionResult AddCoupon()
        {
            if (detailCart.OrderHeader.CouponCode == null)
            {
                detailCart.OrderHeader.CouponCode = "";
            }
            HttpContext.Session.SetString(SD.ssCouponCode, detailCart.OrderHeader.CouponCode);

            return RedirectToAction(nameof(Index));
        }

        public IActionResult RemoveCoupon()
        {

            HttpContext.Session.SetString(SD.ssCouponCode, string.Empty);

            return RedirectToAction(nameof(Index));
        }


        public async Task<IActionResult> Plus(int cartId)
        {
            var cart = await _db.ShoppingCart.FirstOrDefaultAsync(c => c.Id == cartId);
            cart.Count += 1;
            await _db.SaveChangesAsync();
            return RedirectToAction(nameof(Index));
        }

        public async Task<IActionResult> Minus(int cartId)
        {
            var cart = await _db.ShoppingCart.FirstOrDefaultAsync(c => c.Id == cartId);
            if (cart.Count == 1)
            {
                _db.ShoppingCart.Remove(cart);
                await _db.SaveChangesAsync();

                var cnt = _db.ShoppingCart.Where(u => u.ApplicationUserId ==
cart.ApplicationUserId).ToList().Count;
                HttpContext.Session.SetInt32(SD.ssShoppingCartCount, cnt);
            }
            else
            {
                cart.Count -= 1;
                await _db.SaveChangesAsync();
            }

            return RedirectToAction(nameof(Index));
        }

        public async Task<IActionResult> Remove(int cartId)
        {
            var cart = await _db.ShoppingCart.FirstOrDefaultAsync(c => c.Id == cartId);

            _db.ShoppingCart.Remove(cart);
            await _db.SaveChangesAsync();

            var cnt = _db.ShoppingCart.Where(u => u.ApplicationUserId ==
cart.ApplicationUserId).ToList().Count;
            HttpContext.Session.SetInt32(SD.ssShoppingCartCount, cnt);


            return RedirectToAction(nameof(Index));
        }
        }
    }
```

6.4 NuGet Package voor Stripe Installeren



```
var options = new ChargeCreateOptions
{
    Amount = Convert.ToI
    Currency = "eur",
    Description = "Order
```

Generate class 'ChargeCreateOptions' in new file
Generate class 'ChargeCreateOptions'
Generate nested class 'ChargeCreateOptions'
Generate new type...
Install package 'Stripe.net'
Find and install la
Install with packa
rderTotal
der.Id,

6.5 Toevoegen van stripe settings class in folder Utility

```
public class StripeSettings
{
    public string SecretKey { get; set; }
    public string PublishableKey { get; set; }
}
```

6.6 Toevoegen Stripe API Key aan appsettings.json

```
{
  "ConnectionStrings": {
    "DefaultConnection":
"Server=(localdb)\\mssqllocaldb;Database=PittigDBMVC;Trusted_Connection=True;MultipleActi
veResultSets=true"
  },
  "Logging": {
    "LogLevel": {
      "Default": "Information",
      "Microsoft": "Warning",
      "Microsoft.Hosting.Lifetime": "Information"
    },
    "Stripe": {
      "SecretKey": "sk_test_kGzUGDqcDrmVHhktqZZUFyI7",
      "PublishableKey": "pk_test_DS6jsf1BPuhpojL8FqWXbBpT"
    },
    "SendGridKey": "SG.rX1GLOk-SceT42UNLMt2Iw.2ipjqz34QBAc05-2_pvV2xDtRuFf-
yLTtCbc2cw28K8"
  }
      }
```

6.7 In Startup.cs: Aanpassen session settings en configuratie voor Stripe

```
public void ConfigureServices(IServiceCollection services)
{
    services.AddDbContext<ApplicationDbContext>(options =>
        options.UseSqlServer(
            Configuration.GetConnectionString("DefaultConnection")));
    services.AddIdentity<IdentityUser,IdentityRole>()
```

```
                .AddDefaultTokenProviders()
                .AddEntityFrameworkStores<ApplicationDbContext>();// options =>
options.SignIn.RequireConfirmedAccount = true

            services.AddSingleton<IEmailSender, EmailSender>();
            services.Configure<EmailOptions>(Configuration);
            services.Configure<StripeSettings>(Configuration.GetSection("Stripe"));
            services.AddScoped<IDbInitializer, DbInitializer>();
            services.AddControllersWithViews();
            services.AddRazorPages().AddRazorRuntimeCompilation();

            services.AddSession(options =>
            {
                options.Cookie.IsEssential = true;
                options.IdleTimeout = TimeSpan.FromMinutes(30);
                options.Cookie.HttpOnly = true;
            });
        }

        public void Configure(IApplicationBuilder app, IWebHostEnvironment env,
IDbInitializer dbInitializer)
        {
            if (env.IsDevelopment())
            {
                app.UseDeveloperExceptionPage();
                app.UseDatabaseErrorPage();
            }
            else
            {
                app.UseExceptionHandler("/Home/Error");

                app.UseHsts();
            }
            app.UseRouting();
            StripeConfiguration.ApiKey = Configuration.GetSection("Stripe")["SecretKey"];
            dbInitializer.Initialize();
            app.UseHttpsRedirection();
            app.UseStaticFiles();
            app.UseCookiePolicy();
            app.UseSession();
            app.UseAuthentication();
            app.UseAuthorization();
            app.UseEndpoints(endpoints =>
            {
                endpoints.MapControllerRoute(
                    name: "areas",
                    pattern: "{area=Customer}/{controller=Home}/{action=Index}/{id?}");
                endpoints.MapRazorPages();
            });
        }
```

6.8 Index Razor View toevoegen voor winkelwagen

Maak een nieuwe folder ‘Cart’ aan onder de folder Areas/Customer/Views en

Maak hierin een razor view Index.cshtml

```
@model PittigRestoMVC.Models.ViewModels.OrderDetailsCart

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

<form method="post">
    @if (Model.listCart.Count > 0)
    {
        <br />
        <input id="userId" asp-for="@Model.listCart[0].ApplicationUserId" hidden />
        <div class="backgroundWhiteBorder">
            <div class="container">
                <div class="card">
                    <div class="card-header bg-dark text-light ml-0 row container">
                        <div class="col-6">
                            <i class="fa fa-shopping-cart"></i> &nbsp;
                            Winkelwagen
                        </div>
                        <div class="col-6 text-right">
                            <a asp-area="Customer" asp-controller="Home" asp-
action="Index" class="btn btn-outline-info btn-sm">Verder winkelen</a>
                        </div>
                    </div>
                    <div class="card-body">
                        @foreach (var item in Model.listCart)
                        {
                            <div class="row">
                                <div class="d-none d-lg-block col-lg-2 text-center py-2">
                                    <img src="@item.MenuItem.Image" class="rounded"
width="120" height="80" />
                                </div>
                                <div class="col-12 text-sm-center col-lg-5 text-lg-left">
                                    <h4><strong>@item.MenuItem.Name</strong></h4>
                                    <h4><small>@item.MenuItem.Description</small></h4>
                                </div>
                                <div class="col-12 text-sm-center col-lg-5 text-lg-right
row">

                                    <div class="col-4 text-md-right" style="padding-
top:5px;">
                                        <h6><strong>€@item.MenuItem.Price <span
class="text-muted">x</span> @item.Count </strong></h6>
                                    </div>
                                    <div class="col-6 col-sm-4 col-lg-6">
                                        <div class="float-right mx-1">
                                            <button type="submit" class="btn btn-primary"
asp-action="plus" asp-route-cartId="@item.Id">
                                                <i class="fas fa-plus"></i>
                                            </button>
                                        </div>
                                        <div class="float-right mx-1">
                                            <button type="submit" class="btn btn-danger"
asp-action="minus" asp-route-cartId="@item.Id">
                                                <i class="fas fa-minus"></i>
                                            </button>
                                        </div>
                                    </div>
```

```
                  <div class="col-2 col-sm-4 col-lg-2 text-right">
                    <button type="submit" class="btn btn-outline-
danger" asp-action="remove" asp-route-cartId="@item.Id">
                      <i class="fas fa-trash"></i>
                    </button>
                  </div>
                </div>
              </div>
              <hr />
            }

            <div class="row">
              <div class="col-12 col-md-5">
                <div class="row">
                  <div class="col-7">
                    <input asp-for="@Model.OrderHeader.CouponCode"
id="txtCouponCode" class="form-control" placeholder="kortingscode..." />
                  </div>
                  <div class="col-5" style="margin-top:2px;">
                    <button type="submit" class="btn btn-sm form-
control btn-outline-success" id="btnCoupon" asp-action="AddCoupon">
                      Toevoegen
                    </button>
                    <button type="submit" class="btn btn-sm form-
control btn-outline-danger" style="display:none" id="btnRemoveCoupon" asp-
action="RemoveCoupon">
                      Verwijderen
                    </button>
                  </div>
                </div>
              </div>
              <div class="col-12 col-md-6 offset-md-1 col-lg-4 offset-lg-3
pr-4">
                <ul class="list-group">
                  <li class="list-group-item d-flex justify-content-
between bg-light">
                    <span class="text-info"> Totaal (EUR)</span>
                    <strong class="text-info">€ <span
id="txtOrderTotal">@Model.OrderHeader.OrderTotal</span></strong>
                  </li>
                </ul>
              </div>
            </div>


          </div>
          <div class="card-footer">
            <div class="col-12 col-lg-4 offset-lg-8 col-md-6 offset-md-6">
              <a asp-action="Summary" asp-area="Customer" asp-
controller="Cart" class="btn btn-success form-control">Overzicht</a>
            </div>
          </div>
        </div>
      </div>
    </div>
  }
  else
  {
        <div class="backgroundWhiteBorder"> Winkelwagen is leeg...</div>
    }
</form>

@section Scripts{
    <script>
        $(function () {
            var couponCode = document.getElementById("txtCouponCode").value;

            if (couponCode.length > 0) {
                document.getElementById('btnCoupon').style.display = 'none';
                document.getElementById('btnRemoveCoupon').style.display = '';
            }
            else {
                document.getElementById('btnCoupon').style.display = '';
                document.getElementById('btnRemoveCoupon').style.display = 'none';
            }
        });
    </script>
}
```

7 Views/Shared/_Layout.cshtml aanpassen:

7.1 tonen/verbergen van delen op basis van rollen en winkelwagen tonen

```
@using PittigRestoMVC.Utility
@using Microsoft.AspNetCore.Http
@inject IHttpContextAccessor HttpContextAccessor

<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="utf-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <title>@ViewData["Title"] - PittigRestoMVC</title>
    <link rel="stylesheet" href="~/lib/bootstrap/dist/css/bootstrap.min.css" />
    <link rel="stylesheet" href="https://use.fontawesome.com/releases/v5.5.0/css/all.css"
integrity="sha384-B4dIYHKNBt8Bc12p+WXckhzcICo0wtJAoU8YZTY5qE0Id1GSseTk6S+L3BlXeVIU"
crossorigin="anonymous">
    <link rel="stylesheet" href="~/css/site.css" />
</head>
<body class="search-background">
    <header>
        <nav class="navbar navbar-expand-sm navbar-toggleable-sm navbar-dark bg-dark border-bottom
box-shadow mb-3">
            <div class="container">
                <a class="navbar-brand" asp-area="" asp-controller="Home" asp-
action="Index">PittigRestoMVC</a>
                <button class="navbar-toggler" type="button" data-toggle="collapse" data-
target=".navbar-collapse" aria-controls="navbarSupportedContent"
                        aria-expanded="false" aria-label="Toggle navigation">
                    <span class="navbar-toggler-icon"></span>
                </button>
                <div class="navbar-collapse collapse d-sm-inline-flex flex-sm-row-reverse">
                    <partial name="_LoginPartial" />
                    <ul class="navbar-nav flex-grow-1">
                        @if (User.IsInRole(SD.ManagerUser))
                        {
                            <li class="nav-item dropdown text-white-50">
                                <a class="nav-link dropdown-toggle" href="#"
id="navbarDropDownMenuLink" role="button"
                                   data-toggle="dropdown" aria-haspopup="true" aria-
expanded="false">
                                    Inhoud beheren
                                </a>
                                <div class="dropdown-menu" aria-
labelledby="navbarDropDownMenuLink">
                                    <a class="dropdown-item" asp-action="Index" asp-
controller="Category" asp-area="Admin">categorie</a>
                                    <a class="dropdown-item" asp-action="Index" asp-
controller="SubCategory" asp-area="Admin">subcategorie</a>
                                    <a class="dropdown-item" asp-action="Index" asp-
controller="MenuItem" asp-area="Admin">Menu</a>
                                    <a class="dropdown-item" asp-action="Index" asp-
controller="Coupon" asp-area="Admin">Kortingen</a>
                                    <a class="dropdown-item" asp-action="Index" asp-
controller="User" asp-area="Admin">Personeel</a>

                                </div>
                            </li>
                        }
                        @if
(HttpContextAccessor.HttpContext.Session.GetInt32(@SD.ssShoppingCartCount) != null)
                        {
                            <li style="color:white">
                                <a asp-area="Customer" asp-controller="Cart" asp-action="Index"
class="nav-link">
                                    @{
                                        var count =
HttpContextAccessor.HttpContext.Session.GetInt32(@SD.ssShoppingCartCount);
                                    }
                                    <i class="fas fa-shopping-cart"></i> &nbsp; (@count)
                                </a>
                            </li>
                        }
                        else
                        {
                            <li style="color:white">
                                <a href="#" class="nav-link">
                                    <i class="fas fa-shopping-cart"></i> &nbsp; (0)
                                </a>
                            </li>
                        }
                        <li class="nav-item">
                            <a class="nav-link" asp-area="" asp-controller="Home" asp-
action="Index">Home</a>
                        </li>
                        <li class="nav-item">
                            <a class="nav-link" asp-area="" asp-controller="Home" asp-
action="Privacy">Privacy</a>
                        </li>


                    </ul>
                </div>
            </div>
        </nav>
    </header>
    <div class="container">
        <main role="main" class="pb-3">
            @RenderBody()
        </main>
    </div>

    <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>
    <script src="~/lib/jquery/dist/jquery.min.js"></script>
    <script src="~/lib/bootstrap/dist/js/bootstrap.bundle.min.js"></script>
    <script src="~/js/site.js" asp-append-version="true"></script>
    @RenderSection("Scripts", required: false)
</body>
</html>
```

8 Bestellingen toevoegen

8.1 Viewmodel class aanmaken voor OrderDetails

Maak in een folder Models/Viewmodels de class OrderDetailsViewModel aan:

```
namespace PittigRestoMVC.Models.ViewModels
{
    public class OrderDetailsViewModel
    {
        public OrderHeader OrderHeader { get; set; }
        public List<OrderDetails> OrderDetails { get; set; }
    }
}
```

8.2 Viewmodel class aanmaken voor OrderDetails

Maak in een folder Models/Viewmodels de class OrderDetailsViewModel aan:

```
namespace PittigRestoMVC.Models.ViewModels
{
    public class OrderDetailsViewModel
    {
        public OrderHeader OrderHeader { get; set; }
        public List<OrderDetails> OrderDetails { get; set; }
    }
}
```

8.3 Paging Info model aanmaken onder de folder Models (om te gebruiken in orderdetails)

```
public class PagingInfo
{
    public int TotalItem { get; set; }

    public int ItemsPerPage { get; set; }

    public int CurrentPage { get; set; }

    public int totalPage => (int)Math.Ceiling((decimal)TotalItem / ItemsPerPage);

    public string urlParam { get; set; }
  }
```

8.4 Viewmodel class aanmaken voor `OrderListViewModel`

Maak in een folder Models/Viewmodels de class `OrderListViewModel` aan:

```
namespace PittigRestoMVC.Models.ViewModels
{
    public class OrderListViewModel
    {
        public IList<OrderDetailsViewModel> Orders { get; set; }
```

```
            public PagingInfo PagingInfo { get; set; }
        }
    }
```

8.5 Order Controller toevoegen

Maak onder de folder Areas/Customer/Controllers een Lege MVC Controller toe met naam 'OrderController'

OrderController.cs

9 Referenties

https://docs.microsoft.com/en-us/aspnet/core/security/authentication/accconfirm?view=aspnetcore-3.1&tabs=visual-studio

https://fontawesome.com/6?next=%2Fv5.5.0%2Ficons%2Flock-open

http://bekenty.com/send-e-mail-in-asp-net-core-with-sendgrid/

https://hostlaunch.io/docs/how-to-get-a-sendgrid-api-key/

https://www.youtube.com/watch?v=ddSymc0hE0Ahttps://app.sendgrid.com/guide/integrate/langs/csharp

https://www.codeproject.com/Articles/1276970/NET-Core-Razor-Page-Email-Form-using-SendGrid-and

https://sendgrid.com/docs/API_Reference/index.html