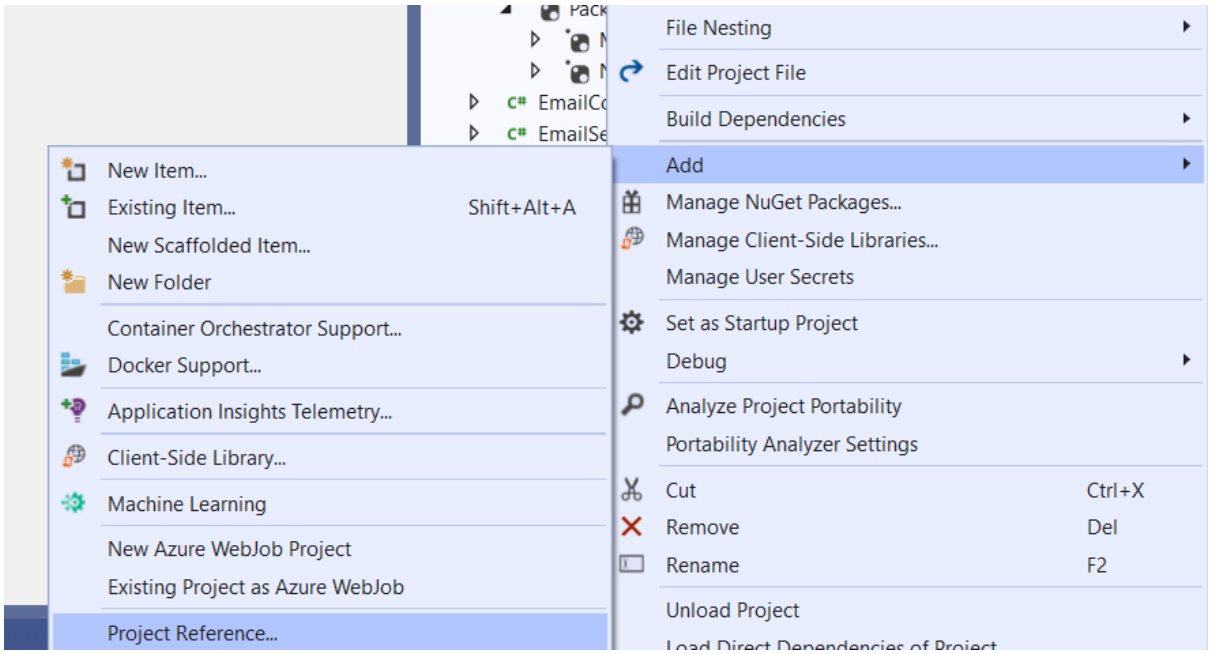
1 Forget pasword in ASP.NET Core MVC web application

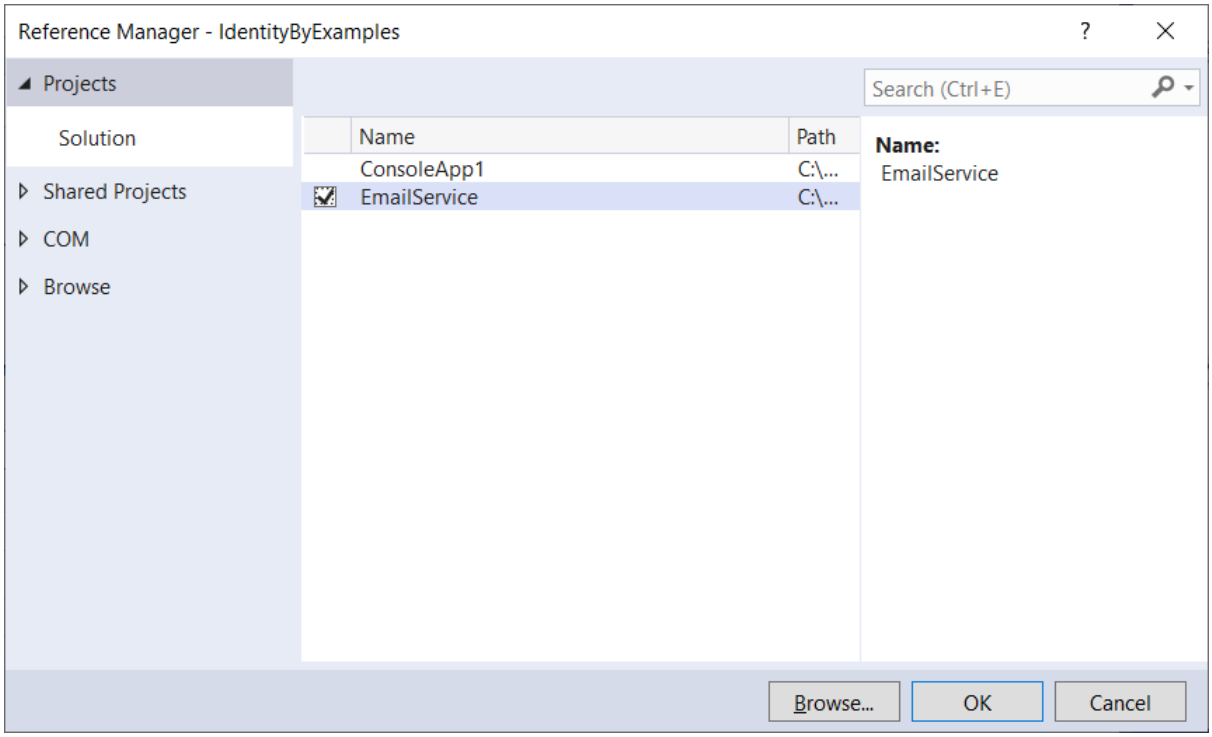
Voor de email-integratie wordt een **EmailService** class library gebruikt van CodeMaze die refereert naar **NETCore.MailKit** NuGet Package waarmee emails kunnen worden gestuurd.

1.1 Leg een project referentie naar het EmailService project

Voeg een project reference toe door de rechtermuisknop te klikken op het project IdentityByExamples en dan Add..Project Reference te selecteren:

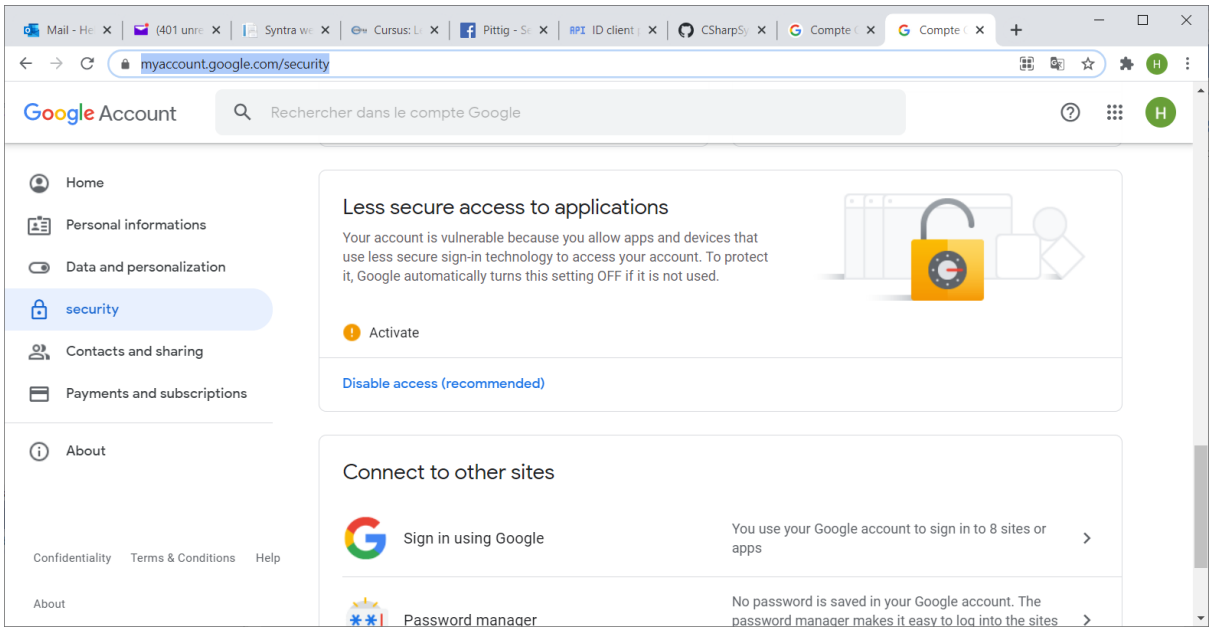


Vink het EmailService project aan en klik op de OK button:



1.2 Google smtp server connectie settings configureren

Voor de email-integratie wordt er gebruik gemaakt van google smpt server. Om deze te kunnen aanspreken heb je een google email account nodig. Om de app te kunnen testen, moet je tijdelijk “Less secure access to applications” aanzetten op je google account:
https://myaccount.google.com/security



Open appsettings.json en voeg de volgende EmailConfigurationsettings toe:

```
{
  "Logging": {
    "LogLevel": {
      "Default": "Information",
      "Microsoft": "Warning",
      "Microsoft.Hosting.Lifetime": "Information"
    }
  },
  "ConnectionStrings": {
    "sqlConnection": "server=.; database=DemoEmail; Integrated Security=true"
  },
  "EmailConfiguration": {
    "From": "hcobpo@gmail.com",
    "SmtpServer": "smtp.gmail.com",
    "Port": 587,
    "Username": "hcobpo@gmail.com",
    "Password": "**********"
  },
  "AllowedHosts": "*"
}
```

1.3 Email service toevoegen aan ConfigureServices

Open Startup.cs en voeg de volgende lijnen toe om de emailservice te registreren :

```
        public void ConfigureServices(IServiceCollection services)
        {
            services.AddDbContext<ApplicationContext>(opts =>

opts.UseSqlServer(Configuration.GetConnectionString("sqlConnection")));
```

```
            services.AddIdentity<IdentityUser, IdentityRole>(opt =>
                {
                    opt.Password.RequiredLength = 4;
                    opt.Password.RequireDigit = false;
                    opt.Password.RequireUppercase = false;
                    opt.Password.RequireNonAlphanumeric = false;
                })
               .AddEntityFrameworkStores<ApplicationContext>()//ADDED 3
                .AddDefaultTokenProviders(); // Enable Token Generation

            services.AddScoped<IUserClaimsPrincipalFactory<IdentityUser>,
CustomClaimsFactory>();
            services.AddAutoMapper(typeof(Startup));

            var emailConfig = Configuration
                .GetSection("EmailConfiguration")
                .Get<EmailConfiguration>();
                services.AddSingleton(emailConfig);
                services.AddScoped<IEmailSender, EmailSender>();

            services.AddControllersWithViews();
        }
```

1.4 Forgot password action methoden toevoegen

Open AcountController.cs en voeg de volgende code toe :

```
namespace IdentityByExamples.Controllers
{
    public class AccountController : Controller
    {

        //ADDED 40: inject this email service in the Account controller:
        private readonly IEmailSender _emailSender;
        //ADDED 34: SignInManager: private field + constructor
        private readonly SignInManager<IdentityUser> _signInManager;
        //ADDED 18: 2 private fields + constructor
        private readonly IMapper _mapper;
        private readonly UserManager<IdentityUser> _userManager;
        public AccountController(IMapper mapper, UserManager<IdentityUser> userManager, SignInManager<IdentityUser>
signInManager, IEmailSender emailSender)
        {
            _mapper = mapper;
            _userManager = userManager;
            _signInManager = signInManager;
            _emailSender = emailSender;
        }


        [HttpGet]
        public IActionResult Register()
        {
            return View();
        }

        [HttpPost]
        [ValidateAntiForgeryToken]
        public async Task<IActionResult> Register(UserRegistrationModel userModel)
        {

            if (!ModelState.IsValid)
            {
                return View(userModel);
            }

            var user = _mapper.Map<IdentityUser>(userModel);

            var result = await _userManager.CreateAsync(user, userModel.Password);
            if (!result.Succeeded)
            {
                foreach (var error in result.Errors)
                {
```

```
                    ModelState.TryAddModelError(error.Code, error.Description);
                }

                return View(userModel);
            }

            await _userManager.AddToRoleAsync(user, "Visitor");

            return RedirectToAction(nameof(HomeController.Index), "Home");
        }

        [HttpGet]
        public IActionResult Login(string returnUrl=null)
        {
            ViewData["ReturnUrl"] = returnUrl;
            return View();
        }

        [HttpPost]
        [ValidateAntiForgeryToken]
        public async Task<IActionResult> Login(UserLoginModel userModel, string returnUrl = null)
        {

            if (!ModelState.IsValid)
            {
                return View(userModel);
            }

            var user = await _userManager.FindByEmailAsync(userModel.Email);
            if (user != null &&
                await _userManager.CheckPasswordAsync(user, userModel.Password))
            {
                var identity = new ClaimsIdentity(IdentityConstants.ApplicationScheme);
                identity.AddClaim(new Claim(ClaimTypes.NameIdentifier, user.Id));
                identity.AddClaim(new Claim(ClaimTypes.Name, user.UserName));

                var result = await _signInManager.PasswordSignInAsync(userModel.Email, userModel.Password,
userModel.RememberMe, false);

                if (result.Succeeded)
                {
                    return RedirectToLocal(returnUrl);
                }
                else
                {
                    ModelState.AddModelError("", "Invalid UserName or Password");
                    return View();
                }
            }
            else
            {
                ModelState.AddModelError("", "Invalid UserName or Password");
                return View();
            }

        }
        private IActionResult RedirectToLocal(string returnUrl)
        {
            if (Url.IsLocalUrl(returnUrl))
                return Redirect(returnUrl);
            else
                return RedirectToAction(nameof(HomeController.Index), "Home");
        }


        [HttpPost]
        [ValidateAntiForgeryToken]
        public async Task<IActionResult> Logout()
        {
            await _signInManager.SignOutAsync();

            return RedirectToAction(nameof(HomeController.Index), "Home");
        }
        // ForgotPassword action method
        [HttpGet]
        public IActionResult ForgotPassword()
        {
            return View();
        }
        //ForgotPassword post action method
        [HttpPost]
        [ValidateAntiForgeryToken]
        public async Task<IActionResult> ForgotPassword(ForgotPasswordModel forgotPasswordModel)
        {
            if (!ModelState.IsValid)
```

```
                return View(forgotPasswordModel);

            var user = await _userManager.FindByEmailAsync(forgotPasswordModel.Email);
            if (user == null)
                return RedirectToAction(nameof(ForgotPasswordConfirmation));

            var token = await _userManager.GeneratePasswordResetTokenAsync(user);
            var callback = Url.Action(nameof(ResetPassword), "Account", new { token, email = user.Email },
Request.Scheme);

            var message = new Message(new string[] { user.Email }, "Reset password token", callback, null);
            await _emailSender.SendEmailAsync(message);

            return RedirectToAction(nameof(ForgotPasswordConfirmation));
        }


        public IActionResult ForgotPasswordConfirmation()
        {
            return View();
        }
        [HttpGet]
        public IActionResult ResetPassword(string token, string email)
        {
            var model = new ResetPasswordModel { Token = token, Email = email };
            return View(model);
        }

        [HttpPost]
        [ValidateAntiForgeryToken]
        public async Task<IActionResult> ResetPassword(ResetPasswordModel resetPasswordModel)
        {
            if (!ModelState.IsValid)
                return View(resetPasswordModel);

            var user = await _userManager.FindByEmailAsync(resetPasswordModel.Email);
            if (user == null)
                RedirectToAction(nameof(ResetPasswordConfirmation));

            var resetPassResult = await _userManager.ResetPasswordAsync(user, resetPasswordModel.Token,
resetPasswordModel.Password);
            if (!resetPassResult.Succeeded)
            {
                foreach (var error in resetPassResult.Errors)
                {
                    ModelState.TryAddModelError(error.Code, error.Description);
                }

                return View();
            }

            return RedirectToAction(nameof(ResetPasswordConfirmation));
        }

        [HttpGet]
        public IActionResult ResetPasswordConfirmation()
        {
            return View();
        }
    }
}
```

1.5 ForgotPassword en ForgotPasswordConfirmation Razor views toevoegen

Voeg onder de folder Views/Account de volgende razor views toe:

ForgotPassword.cshtml

```
@model IdentityByExamples.Models.ForgotPasswordModel

<h1>ForgotPassword</h1>

<div class="row">
    <div class="col-md-4">
        <form asp-action="ForgotPassword">
            <div asp-validation-summary="ModelOnly" class="text-danger"></div>
            <div class="form-group">
```

```
                <label asp-for="Email" class="control-label"></label>
                <input asp-for="Email" class="form-control" />
                <span asp-validation-for="Email" class="text-danger"></span>
            </div>
            <div class="form-group">
                <input type="submit" value="Submit" class="btn btn-primary" />
            </div>
        </form>
    </div>
</div>

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

ForgotPasswordConfirmation.cshtml

```
<h1>ForgotPasswordConfirmation</h1>

<p>
    The link has been sent, please check your email to reset your password.
</p>
```

1.6 Link naar Forgotpassword toevoegen aan Login Razor view

Open de Razor view **Login.cshtml** onder Views/Account en voeg de volgende razor code toe:

```
@model IdentityByExamples.Models.UserLoginModel
<h1>Login</h1>

<div class="row">
    <div class="col-md-4">
        @*<form asp-action="Login"> ADDED 31: ReturnUrl*@XHTML
        <form asp-action="Login" asp-route-returnUrl="@ViewData["ReturnUrl"]">
            <div asp-validation-summary="ModelOnly" class="text-danger"></div>
            <div class="form-group">
                <label asp-for="Email" class="control-label"></label>
                <input asp-for="Email" class="form-control" />
                <span asp-validation-for="Email" class="text-danger"></span>
            </div>
            <div class="form-group">
                <label asp-for="Password" class="control-label"></label>
                <input asp-for="Password" class="form-control" />
                <span asp-validation-for="Password" class="text-danger"></span>
            </div>
            <div class="form-group form-check">
                <label class="form-check-label">
                    <input class="form-check-input" asp-for="RememberMe" /> @Html.DisplayNameFor(model
=> model.RememberMe)
                </label>
            </div>
            <div class="form-group">
                <a asp-action="ForgotPassword">Forgot Password</a>
            </div>
            <div class="form-group">
                <input type="submit" value="Log In" class="btn btn-primary" />
            </div>
        </form>
    </div>
</div>

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

1.7 Run de app en test de forgot pasword link en functionaliteit

2 Referenties

www.codemaze.com

https://dotnetcoretutorials.com/2018/03/18/common-errors-sending-email-mailkit/

https://myaccount.google.com/security