LØAD

DEWINTER HAN & MYNY YANNICK

OVERZICHT

- ✓Inspiratie
- ✓Demonstratie
- ✓Toepassingen
- ✓Doelstellingen vs realiteit
- ✓Extra

INSPIRATIE

DEMONSTRATIE

TOEPASSINGEN

TOEPASSINGEN GEZIEN IN DE LES

- ✓Entity Framework code first

```
public class AppIdentityDbContext : IdentityDbContext<AppIdentityUser, AppIdentityRole, string>
{

    11 references
    public DbSet<Blog> Blogs { get; set; }

    7 references
    public DbSet<JobOpening> JobOpenings { get; set; }

    0 references
    public DbSet<Partner> Partners { get; set; }

    7 references
    public DbSet<HomePageSlider> HomePageSliders { get; set; }

    30 references
    public DbSet<IntakeFormCompleted> IntakeFormsCompleted { get; set; }

    1 reference
    public DbSet<BikeFitIntakeForm> BikeFitIntakeForms { get; set; }
    1 reference
    public DbSet<CoachingIntakeForm> CoachingIntakeForms { get; set; }
    1 reference
    public DbSet<PersonalTrainingIntakeForm> PersonalTrainingIntakeForms { get; set; }
    1 reference
    public DbSet<PhysioTherapyIntakeForm> PhysioTherapyIntakeForms { get; set; }
    1 reference
    public DbSet<ZwiftBarIntakeForm> ZwiftBarIntakeForms { get; set; }


        1 reference
        public AppIdentityDbContext(DbContextOptions<AppIdentityDbContext> options) : base(options)
        {

        }
}
```

TOEPASSINGEN GEZIEN IN DE LES

- ✓ Generic class

```
public class PagingList <T> : List<T>
{
    99+ references
    public int PageIndex { get; private set; }
    58 references
    public int TotalPages { get; set; }

    1 reference
    public PagingList(List<T> items, int count, int pageIndex, int pageSize)
    {
        PageIndex = pageIndex;
        TotalPages = (int)Math.Ceiling(count / (double)pageSize);
        this.AddRange(items);
    }
```

TOEPASSINGEN GEZIEN IN DE LES

- ✓Identity

```
1 reference
public class SecurityController : Controller
{
    private readonly UserManager<AppIdentityUser> userManager;
    //Nodig om applicatierollen toe te voegen of te testen
    private readonly RoleManager<AppIdentityRole> roleManager;
    //Nodig om user te valideren en authenticatiecookie te schrijven en lezen
    private readonly SignInManager<AppIdentityUser> signInManager;
    private readonly IEmailSender _emailSender;


    0 references
    public SecurityController(UserManager<AppIdentityUser> userManager, RoleManager<AppIdentityRole> roleManager, SignInManager<AppIdentityUser> signInManager,
    {

        this.userManager = userManager;
        this.roleManager = roleManager;
        this.signInManager = signInManager;
        _emailSender = emailSender;

    }

    0 references
```

TOEPASSINGEN GEZIEN IN DE LES

- ✓ Linq

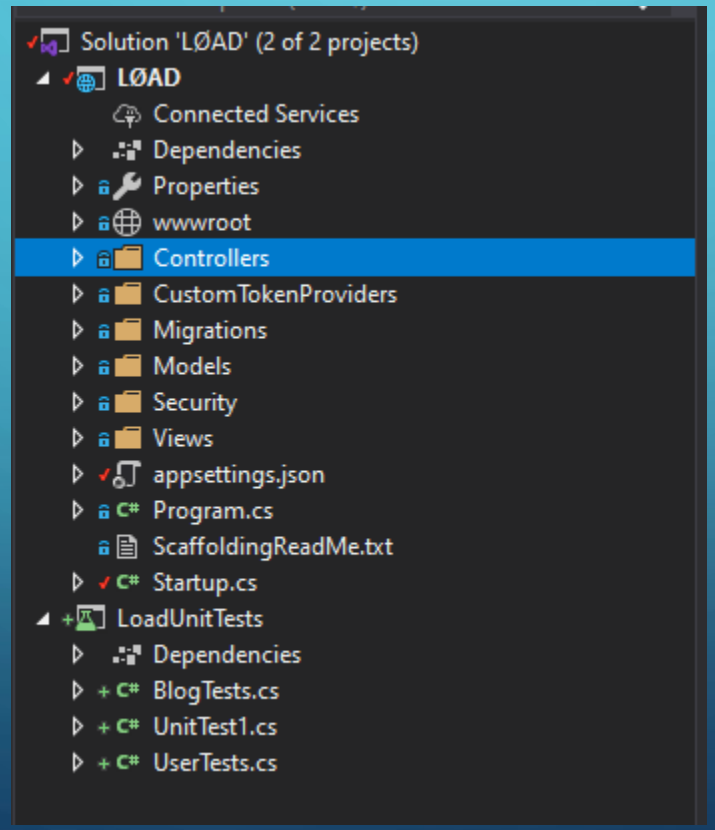
```
var users =  (from user in _context.Users join userRoles in _context.UserRoles
              on user.Id equals userRoles.UserId join roles in _context.Roles
              on userRoles.RoleId equals roles.Id
              where roles.Name == "User" where user.LastName.Contains(lastName) select user)
              .OrderBy(x => x.LastName);
```

```
var register = await _context.Users.Include(m => m.IntakeForm)
    .FirstOrDefaultAsync(m => m.Id == id);
```

```
newBlog.Author = (from e in _context.Users where e.UserName == User.Identity.Name select e).FirstOrDefault();
```

TOEPASSINGEN GEZIEN IN DE LES

- ✓MVC



TOEPASSINGEN GEZIEN IN DE LES

✓Razor View

```
@foreach (var item in Model)
{
    <li id="s1">
        <img src="@item.HomePageSlider.Image" />
    </li>
}
```

TOEPASSINGEN GEZIEN IN DE LES

- ✓ Unit tests

```
0 references
public void BlogViewsShouldNotBeNegative()
{
    Blog blog = new Blog();
    Assert.That(blog.Views, Is.Not.Negative);
}

[Test]
0 references
public void CreateBlogWithCorrectInfoShoudSucceed()
{

    Blog blog = new Blog()
    {
        Title = "TestTitle",
        Content = "TestContent",
        Category = BlogCategory.Lopen,
        PublishDate = DateTime.Now,
        Views = 0
    };

    Assert.AreEqual(blog.Title.GetType(), typeof(string));
    Assert.AreEqual(blog.Content.GetType(), typeof(string));
    Assert.AreEqual(blog.Category.GetType(), typeof(BlogCategory));
    Assert.AreEqual(blog.PublishDate.GetType(), typeof(DateTime));
    Assert.AreEqual(blog.Views.GetType(), typeof(int));
}
```

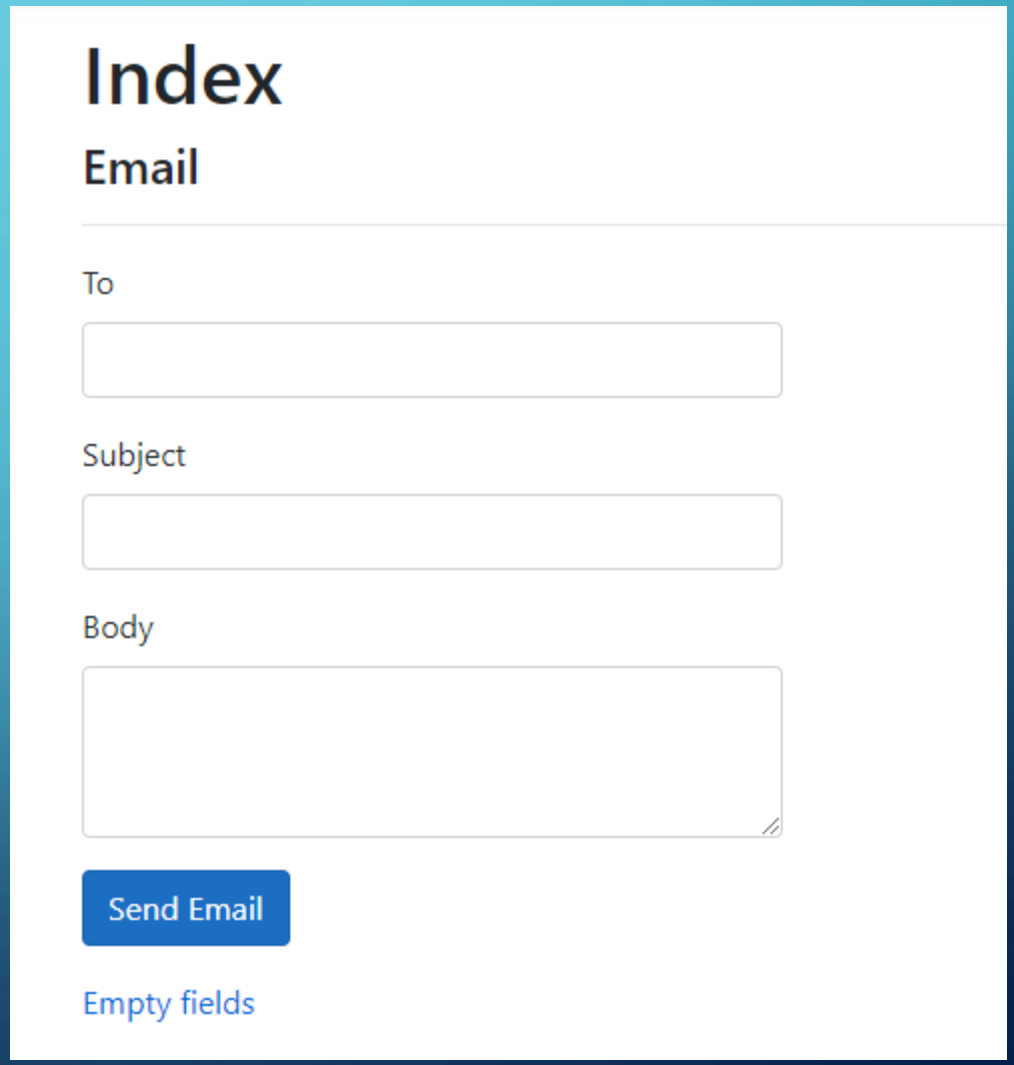
TOEPASSINGEN GEZIEN IN DE LES

- ✓ ViewModels

```
5 references
public class DashboardViewModel
{

    [Display(Name = "Totaal aantal geregistreerde gebruikers")]
    3 references
    public int TotalAmountOfUsers { get; set; }

    [Display(Name = "Totaal aantal views op de blogs")]
    3 references
    public int TotalAmountOfViews { get; set; }

    [Display(Name = "Aantal intakeforms te behandelen")]
    3 references
    public int AmountOfIntakeformsToHandle { get; set; }

    [Display(Name = "Totaal aantal Admins")]
    3 references
    public int AmountOfAdmins { get; set; }
}
```

NIEUWE TOEPASSINGEN

- ✓Emailservice to customers



NIEUWE TOEPASSINGEN

- ✓Two factor authentication

```
IdentityResult result = userManager.CreateAsync(user, obj.Password).Result;

if (result.Succeeded)
{
    var token = await userManager.GenerateEmailConfirmationTokenAsync(user);
    var callback = Url.Action(nameof(ConfirmEmail), "Security", new { token, email = user.Email }, Request.Scheme);
    var message = new Message(user.Email, "Confirm Email", callback);

    _emailSender.SendEmail(message);
    userManager.AddToRoleAsync(user, "User").Wait();
    return RedirectToAction(nameof(ConfirmYourEmail));
}
```

```
2 references
public async Task<IActionResult> ConfirmEmail(string token, string email)
{
    var user = await userManager.FindByEmailAsync(email);
    if (user == null)
        return View("Error");

    var result = await userManager.ConfirmEmailAsync(user, token);
    return View(result.Succeeded ? nameof(ConfirmEmail) : "Error");
}
```

NIEUWE TOEPASSINGEN

✓Pagination

Blogs

Title	Content	Views	Publish Date	Category	Author	
Test5	dsds	0	13/07/2020	Wielrennen		Details
Test4	ddvsdb	0	13/07/2020	Wielrennen		Details

First 1 2 3 Last

DOELSTELLINGEN VS REALITEIT

DOELSTELLINGEN VS REALITEIT: (NIET-) FUNCTIONELE VEREISTEN

(Niet-) Functionele vereisten
De klanten moeten zich kunnen registreren aan de hand van een registratieformulier, en de gegevens hiervan moeten bewaard worden in een database.
De klanten moeten een intakeformulier kunnen invullen en aanpassen naar aanleiding van een afspraak, en de gegevens hiervan moeten bewaard worden in de database.
De klanten moeten zich kunnen aanmelden aan de hand van hun e-mailadres en wachtwoord.
De administrators moeten een nieuwe administrator kunnen aanmaken/aanpassen/verwijderen.
De administrator moet blogberichten kunnen toevoegen, wijzigen en verwijderen.
Via een dashboard in de klantenzone, moet de administrator de klanten kunnen opzoeken aan de hand van hun naam, e-mailadres of geboortedatum.
Het wachtwoord van de klanten moet geëncrypteerd worden.
De administrator moet in staat zijn om vacatures toe te voegen/aan te passen/te verwijderen aan/op/van de website.
Zodra een intaketest werd ingevuld, moet dit worden opgeslagen in een lijst in de AdminZone.
De administrator moet in staat zijn om externe partners toe te voegen/aan te passen/te verwijderen.
De administrator moet in staat zijn nieuwe posts op de homepage te laden verschijnen/aan te passen/te verwijderen

EXTRA

EXTRA

- ✓Two-factor authentication
- ✓E-mailservice
- ✓Homepage sliders
- ✓Pagination