Kullanıcı Yönetimi (ASP.NET Core Identity Frameworkü Üzerinde Kimlik Doğrulama)

22 Nisan 2020 Çarşamba 17:22

- Kullanıcı Yönetimine Giriş
- Kimlik Doğrulama (Authentication) ve Yetkilendirme (Authorization) Arasındaki Fark
- Asp.Net Core Identity (Kimlik) İle Kullanıcı Yönetimi
- Yeni Projelerin Identity İle Başlatılması
- Mevcut Projelere Identity'nin Eklenmesi
- Temel Sınıflara Kalıtım Yoluyla Property Ekleme
- Temel Sınıfların Varsayılan Primary Key'lerini Değiştirme
- Identity ile Kimlik Doğrulama İşlemleri (Register, Login, Logout)
  - Alan (Area) Yapısı
  - Razor Pages
  - Partial View
- Kimlik Doğrulama İşlemlerinde Yeniden Doğrulama (Eposta Onayı - Mail Confirm)
  - Hosting Üzerinde E-Posta Yapılandırması
  - Servis İşlemleri
    - Servis Alma
    - Servislerin Yaşam Ömürleri
  - E-Posta Gönderme
- Kimlik Doğrulama İşlemlerinde Şifre Ayarları

- 10 - ETicaretSQ...

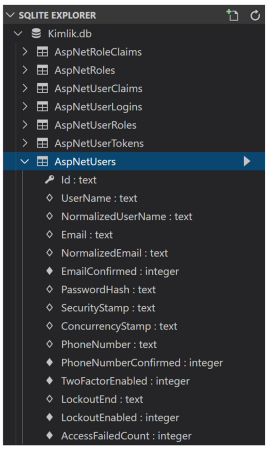
ASP.NET Core'da Identity'e Giriş

- https://docs.microsoft.com/tr-tr/aspnet/core/security/authentication/identity?view=aspnetcore-3.1&tabs=netcore-cli
- https://www.gencayyildiz.com/blog/asp-net-core-identity-yazi-dizisi/ (Çok Detaylı, mutlaka bakın)

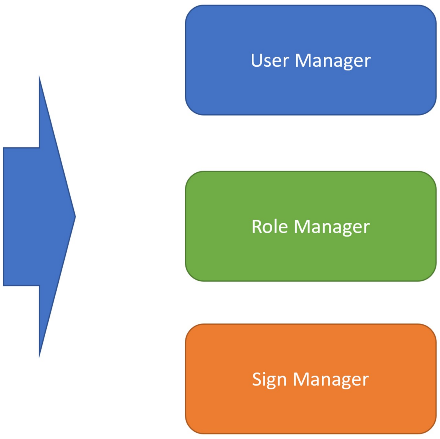
ASP.NET Identity:

- Kullanıcı arabirimi (UI) oturum açma işlevini destekleyen bir API'dir.
- Kullanıcıları, şifreleri, profil verilerini, rolleri, hak taleplerini, jetonları, e-posta onayını ve daha fazlasını yönetir.
- Kullanıcılar, Identity'de depolanan giriş bilgileriyle bir hesap oluşturabilir veya harici bir giriş sağlayıcı kullanabilirler.
- Desteklenen harici giriş sağlayıcıları arasında Facebook, Google, Microsoft Hesabı ve Twitter bulunur .
- Identity kaynak kodu GitHub'da mevcuttur.
- Identity genellikle kullanıcı adlarını, parolaları ve profil verilerini depolamak için bir SQL Server veritabanı kullanılarak yapılandırılır. Alternatif olarak, SQLite, MySQL gibi veritabanı yönetim sistemleri yanında Azure Tablo Deposu da kullanılabilir.

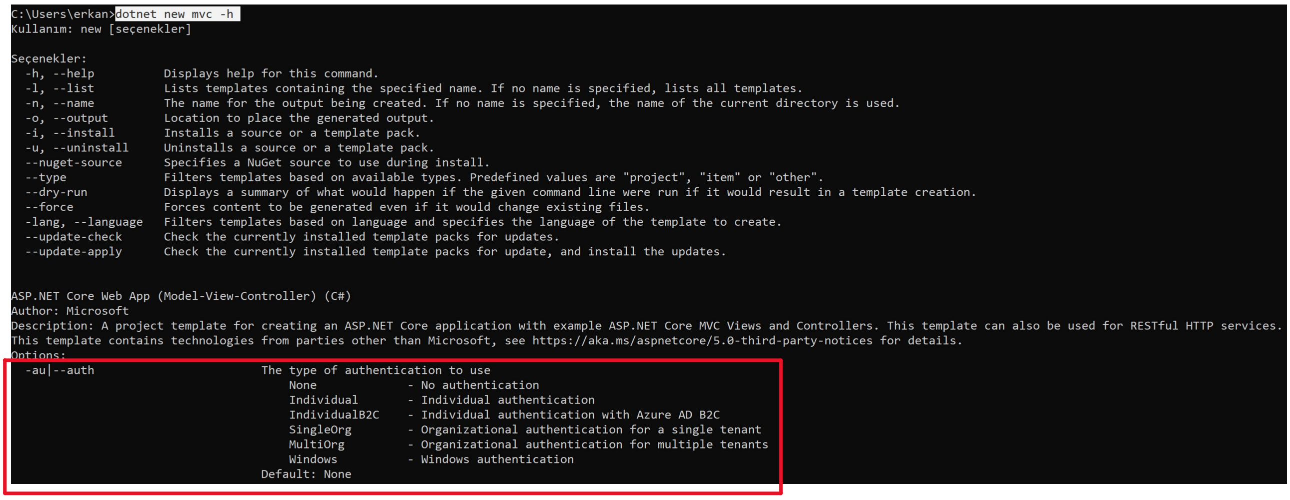
Identity



```
SQLITE EXPLORER
  Kimlik.db
    AspNetRoleClaims
    AspNetRoles
    AspNetUserClaims
    AspNetUserLogins
    AspNetUserRoles
    AspNetUserTokens
    AspNetUsers
      Id : text
      UserName : text
      NormalizedUserName : text
      Email : text
      NormalizedEmail : text
      EmailConfirmed : integer
      PasswordHash : text
      SecurityStamp : text
      ConcurrencyStamp : text
      PhoneNumber : text
      PhoneNumberConfirmed : integer
      TwoFactorEnabled : integer
      LockoutEnd : text
      LockoutEnabled : integer
      AccessFailedCount : integer
```

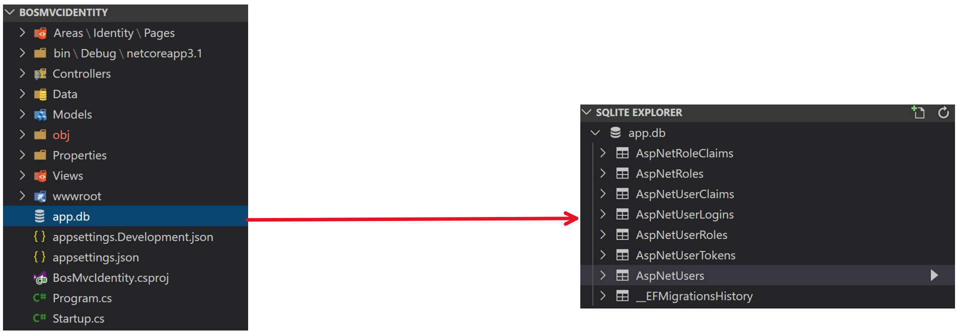


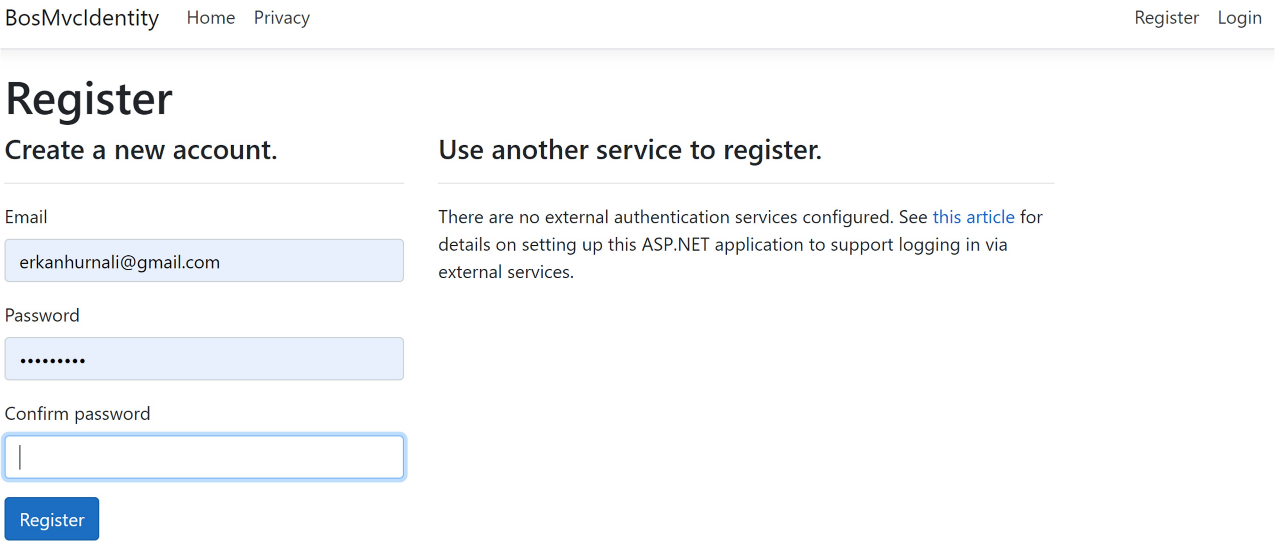
Identity'li boş bir mvc web uygulaması oluşturma



```
C:\Users\erkan>dotnet new mvc -h
Kullanım: new [seçenekler]

Seçenekler:
  -h, --help          Displays help for this command.
  -l, --list          Lists templates containing the specified name. If no name is specified, lists all templates.
  -n, --name          The name for the output being created. If no name is specified, the name of the current directory is used.
  -o, --output        Location to place the generated output.
  -i, --install       Installs a source or a template pack.
  -u, --uninstall     Uninstalls a source or a template pack.
  --nuget-source      Specifies a NuGet source to use during install.
  --type              Filters templates based on available types. Predefined values are "project", "item" or "other".
  --dry-run           Displays a summary of what would happen if the given command line were run if it would result in a template creation.
  --force             Forces content to be generated even if it would change existing files.
  -lang, --language   Filters templates based on language and specifies the language of the template to create.
  --update-check      Check the currently installed template packs for updates.
  --update-apply      Check the currently installed template packs for update, and install the updates.


ASP.NET Core Web App (Model-View-Controller) (C#)
Author: Microsoft
Description: A project template for creating an ASP.NET Core application with example ASP.NET Core MVC Views and Controllers. This template can also be used for RESTful HTTP services.
This template contains technologies from parties other than Microsoft, see https://aka.ms/aspnetcore/5.0-third-party-notices for details.
Options:
  -au|--auth                    The type of authentication to use
                                    None              - No authentication
                                    Individual        - Individual authentication
                                    IndividualB2C     - Individual authentication with Azure AD B2C
                                    SingleOrg         - Organizational authentication for a single tenant
                                    MultiOrg          - Organizational authentication for multiple tenants
                                    Windows           - Windows authentication
                                Default: None
```

- dotnet new mvc --auth Individual
- dotnet ef database update (Gerekirse migration'ları uygulamak için)





Register

Create a new account.

Email

erkanhurnali@gmail.com

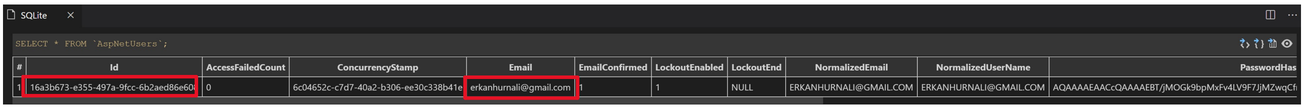
Password

•••••••••

Confirm password

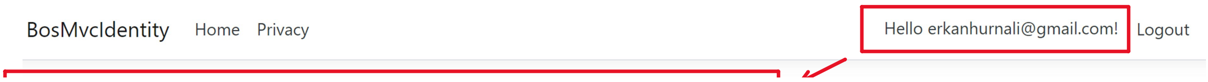
Use another service to register.

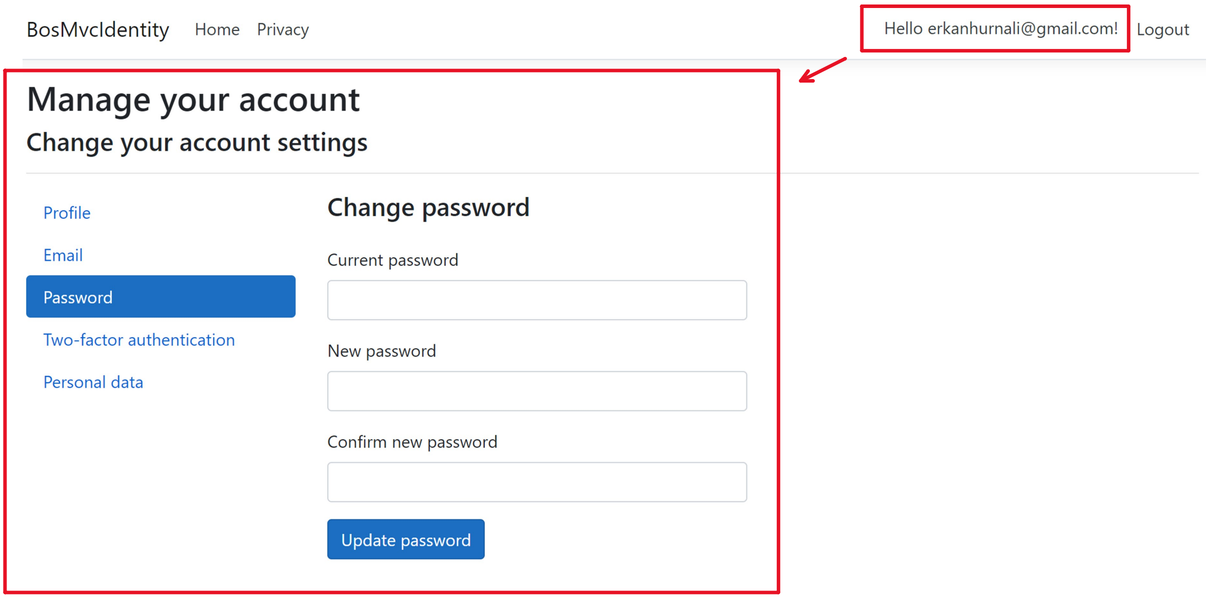
There are no external authentication services configured. See this article for details on setting up this ASP.NET application to support logging in via external services.



```
SELECT * FROM `AspNetUsers`;
```

#	Id	AccessFailedCount	ConcurrencyStamp	Email	EmailConfirmed	LockoutEnabled	LockoutEnd	NormalizedEmail	NormalizedUserName	PasswordHas
1	16a3b673-e355-497a-9fcc-6b2aed86e60	0	6c04652c-c7d7-40a2-b306-ee30c338b41e	erkanhurnali@gmail.com	1	1	NULL	ERKANHURNALI@GMAIL.COM	ERKANHURNALI@GMAIL.COM	AQAAAAEAACcQAAAAEBT/jMOGk9bpMxFv4LV9F7JjMZwqCf





Manage your account

Change your account settings

Change password

Current password

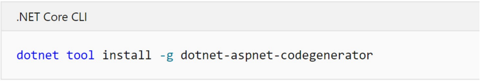
New password

Confirm new password

İskele Yapısı İle Mevcut Bir Projeye Identity Ekleme

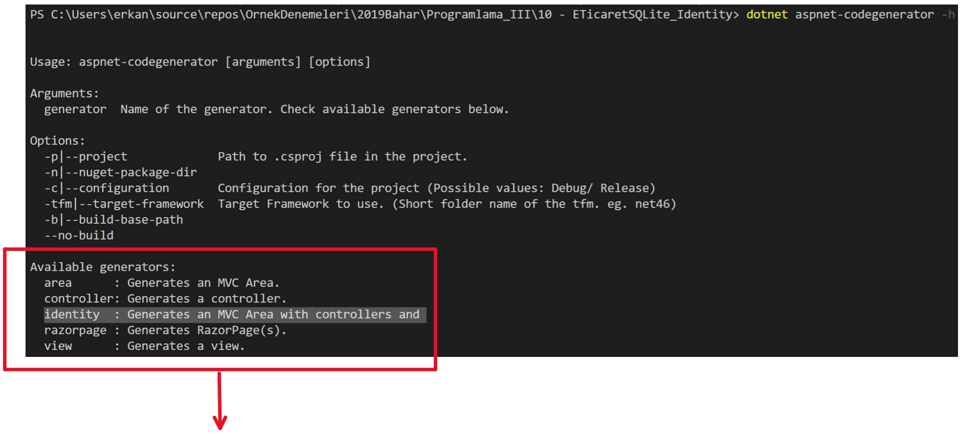
Hatırlatma

ASP.NET Core iskele kurucu daha önce yüklemediyseniz şimdi yükleyin:



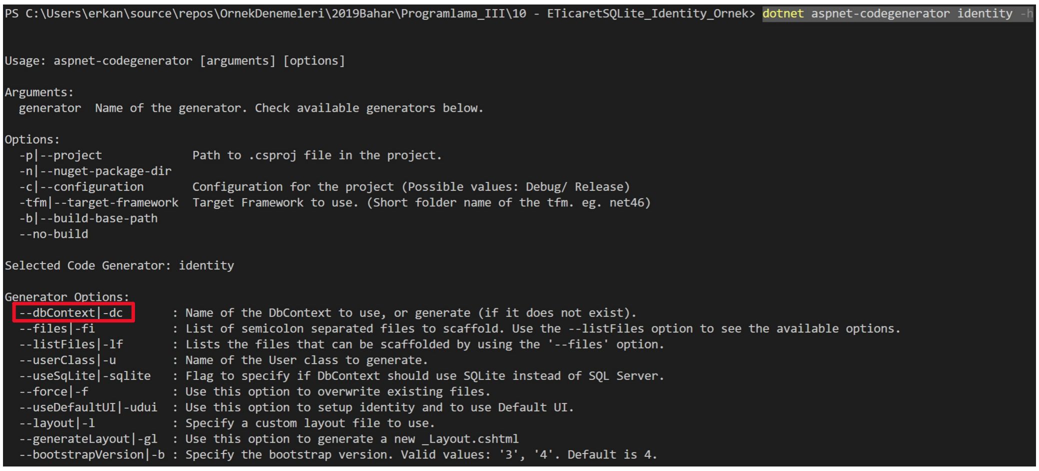
.NET Core CLI

```
dotnet tool install -g dotnet-aspnet-codegenerator
```



```
PS C:\Users\erkan\source\repos\OrnekDenemeleri\2019Bahar\Programlama_III\10 - ETicaretSQLite_Identity> dotnet aspnet-codegenerator -h


Usage: aspnet-codegenerator [arguments] [options]

Arguments:
  generator  Name of the generator. Check available generators below.

Options:
  -p|--project             Path to .csproj file in the project.
  -n|--nuget-package-dir
  -c|--configuration       Configuration for the project (Possible values: Debug/ Release)
  -tfm|--target-framework  Target Framework to use. (Short folder name of the tfm. eg. net46)
  -b|--build-base-path
  --no-build

Available generators:
  area      : Generates an MVC Area.
  controller: Generates a controller.
  identity  : Generates an MVC Area with controllers and
  razorpage : Generates RazorPage(s).
  view      : Generates a view.
```

dotnet add package Microsoft.VisualStudio.Web.CodeGeneration.Design   (CodeGenerator İçin Gerekli)
dotnet add package Microsoft.EntityFrameworkCore.Design
dotnet add package Microsoft.EntityFrameworkCore.SqlServer



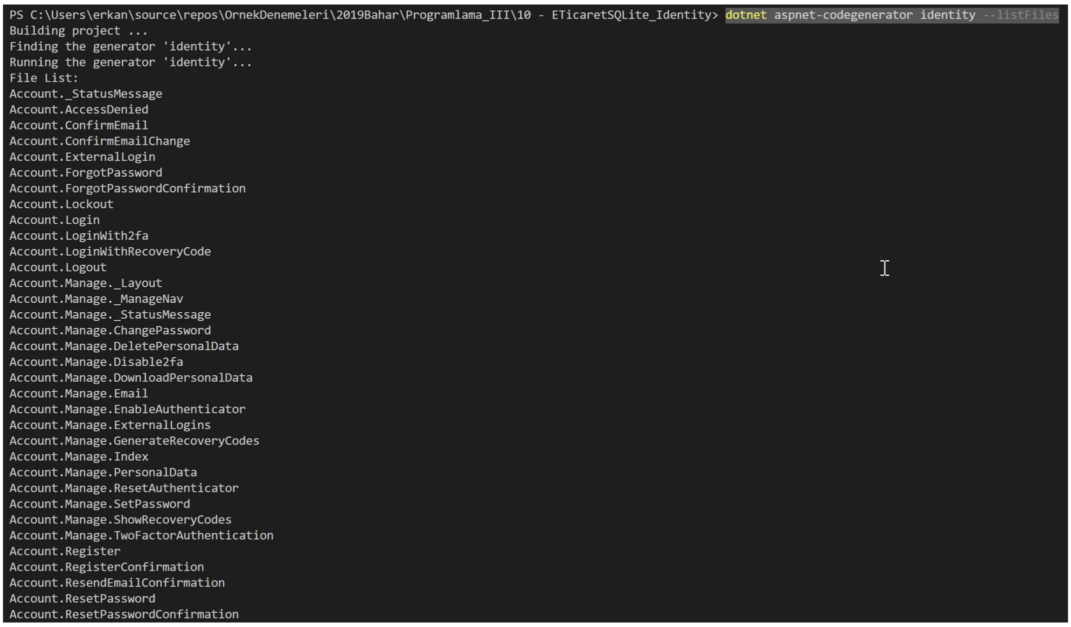
```
PS C:\Users\erkan\source\repos\OrnekDenemeleri\2019Bahar\Programlama_III\10 - ETicaretSQLite_Identity_Ornek> dotnet aspnet-codegenerator identity -h


Usage: aspnet-codegenerator [arguments] [options]

Arguments:
  generator  Name of the generator. Check available generators below.

Options:
  -p|--project             Path to .csproj file in the project.
  -n|--nuget-package-dir
  -c|--configuration       Configuration for the project (Possible values: Debug/ Release)
  -tfm|--target-framework  Target Framework to use. (Short folder name of the tfm. eg. net46)
  -b|--build-base-path
  --no-build

Selected Code Generator: identity

Generator Options:
  --dbContext|-dc      : Name of the DbContext to use, or generate (if it does not exist).
  --files|-fi          : List of semicolon separated files to scaffold. Use the --listFiles option to see the available options.
  --listFiles|-lf      : Lists the files that can be scaffolded by using the '--files' option.
  --userClass|-u       : Name of the User class to generate.
  --useSqLite|-sqlite  : Flag to specify if DbContext should use SQLite instead of SQL Server.
  --force|-f           : Use this option to overwrite existing files.
  --useDefaultUI|-udui  : Use this option to setup identity and to use Default UI.
  --layout|-l          : Specify a custom layout file to use.
  --generateLayout|-gl  : Use this option to generate a new _Layout.cshtml
  --bootstrapVersion|-b : Specify the bootstrap version. Valid values: '3', '4'. Default is 4.
```

Kurulması Gereken Diğer Paketler

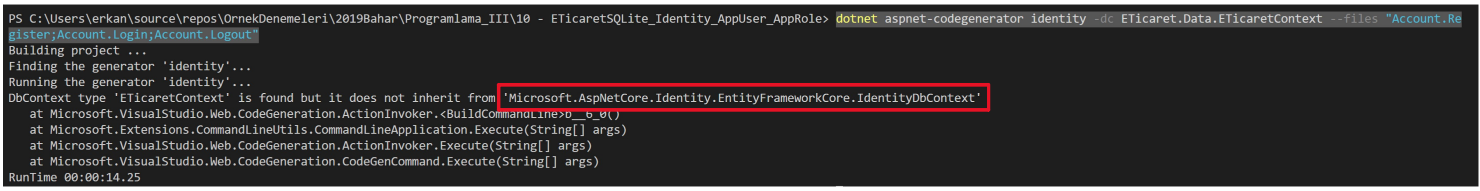
- dotnet add package Microsoft.AspNetCore.Identity.EntityFrameworkCore
- dotnet add package Microsoft.AspNetCore.Identity.UI

Temel Sınıflar İle Identity Yapılandırması

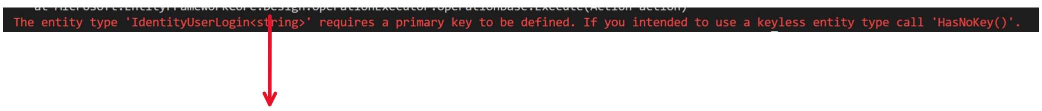
- dotnet aspnet-codegenerator identity -dc ETicaret.Data.ETicaretContext --files "Account.Register;Account.Login;Account.Logout"
- dotnet aspnet-codegenerator identity -dc ETicaret.Data.ETicaretContext (tüm dosyaları ekler)



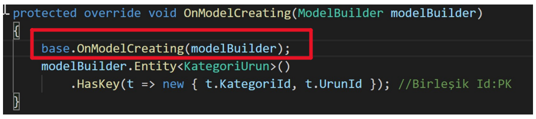
```
PS C:\Users\erkan\source\repos\OrnekDenemeleri\2019Bahar\Programlama_III\10 - ETicaretSQLite_Identity> dotnet aspnet-codegenerator identity --listFiles
Building project ...
Finding the generator 'identity'...
Running the generator 'identity'...
File List:
Account._StatusMessage
Account.AccessDenied
Account.ConfirmEmail
Account.ConfirmEmailChange
Account.ExternalLogin
Account.ForgotPassword
Account.ForgotPasswordConfirmation
Account.Lockout
Account.Login
Account.LoginWith2fa
Account.LoginWithRecoveryCode
Account.Logout
Account.Manage._Layout
Account.Manage._ManageNav
Account.Manage._StatusMessage
Account.Manage.ChangePassword
Account.Manage.DeletePersonalData
Account.Manage.Disable2fa
Account.Manage.DownloadPersonalData
Account.Manage.Email
Account.Manage.EnableAuthenticator
Account.Manage.ExternalLogins
Account.Manage.GenerateRecoveryCodes
Account.Manage.Index
Account.Manage.PersonalData
Account.Manage.ResetAuthenticator
Account.Manage.SetPassword
Account.Manage.ShowRecoveryCodes
Account.Manage.TwoFactorAuthentication
Account.Register
Account.RegisterConfirmation
Account.ResendEmailConfirmation
Account.ResetPassword
Account.ResetPasswordConfirmation
```



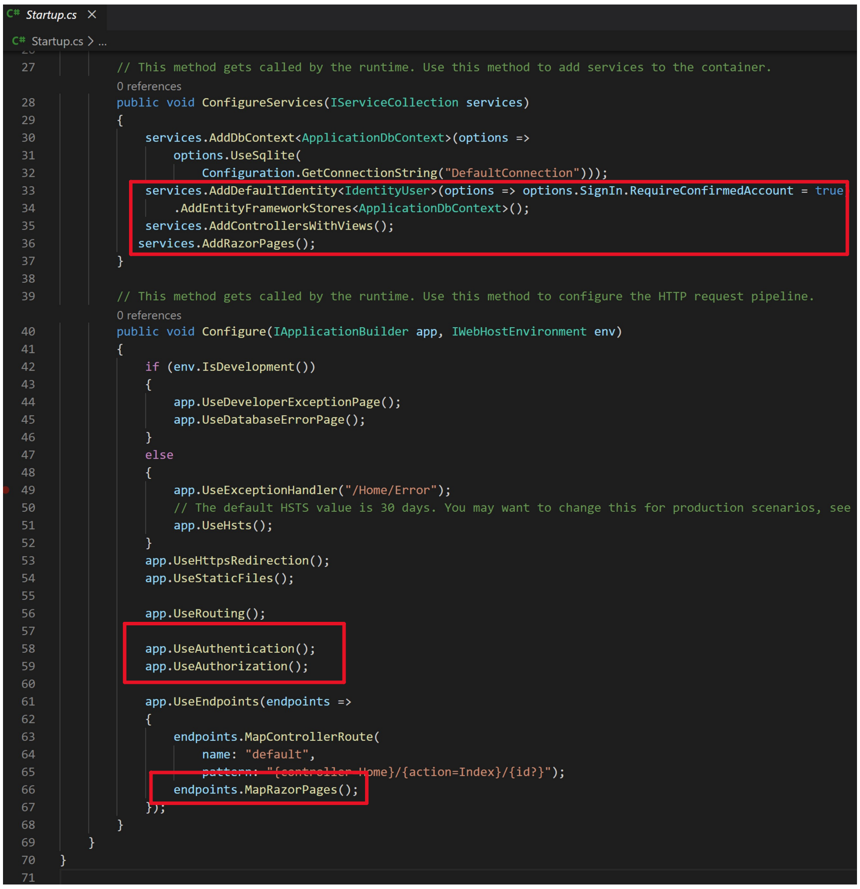
```
PS C:\Users\erkan\source\repos\OrnekDenemeleri\2019Bahar\Programlama_III\10 - ETicaretSQLite_Identity_AppUser_AppRole> dotnet aspnet-codegenerator identity -dc ETicaret.Data.ETicaretContext --files "Account.Re
gister;Account.Login;Account.Logout"
Building project ...
Finding the generator 'identity'...
Running the generator 'identity'...
DbContext type 'ETicaretContext' is found but it does not inherit from 'Microsoft.AspNetCore.Identity.EntityFrameworkCore.IdentityDbContext'
   at Microsoft.VisualStudio.Web.CodeGeneration.ActionInvoker.<BuildCommandLine>b__6_0()
   at Microsoft.Extensions.CommandLineUtils.CommandLineApplication.Execute(String[] args)
   at Microsoft.VisualStudio.Web.CodeGeneration.ActionInvoker.Execute(String[] args)
   at Microsoft.VisualStudio.Web.CodeGeneration.CodeGenCommand.Execute(String[] args)
RunTime 00:00:14.25
```



```
   at Microsoft.EntityFrameworkCore.Design.OperationExecutor.OperationBase.Execute(Action action)
The entity type 'IdentityUserLogin<string>' requires a primary key to be defined. If you intended to use a keyless entity type call 'HasNoKey()'.
```



```
protected override void OnModelCreating(ModelBuilder modelBuilder)
{
    base.OnModelCreating(modelBuilder);
    modelBuilder.Entity<KategoriUrun>()
        .HasKey(t => new { t.KategoriId, t.UrunId }); //Birleşik Id:PK
}
```

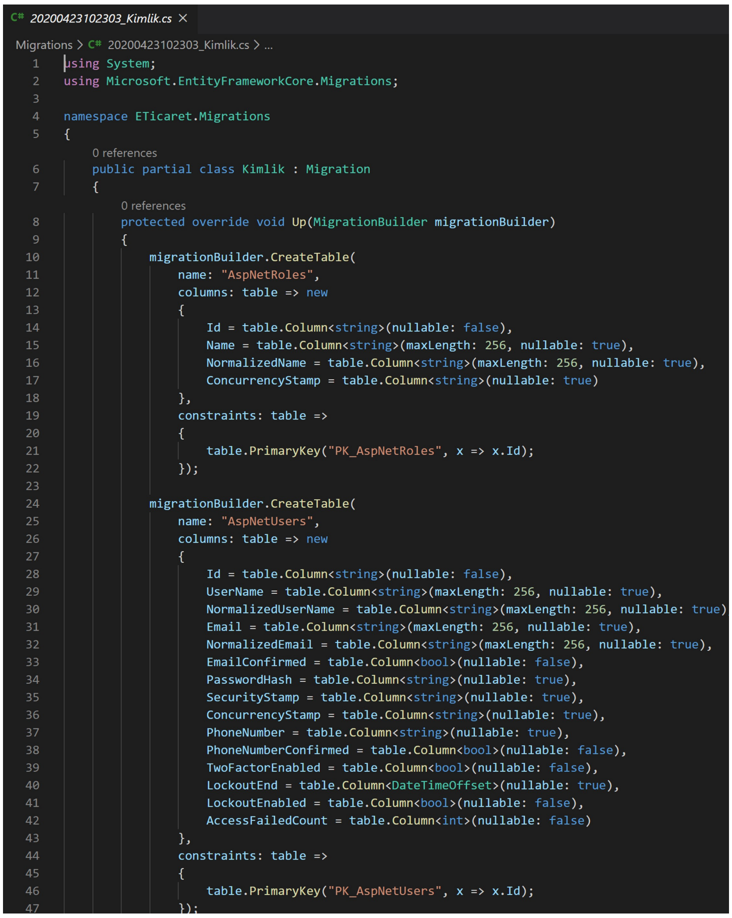


Startup.cs

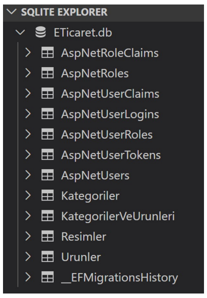
Startup.cs > ...

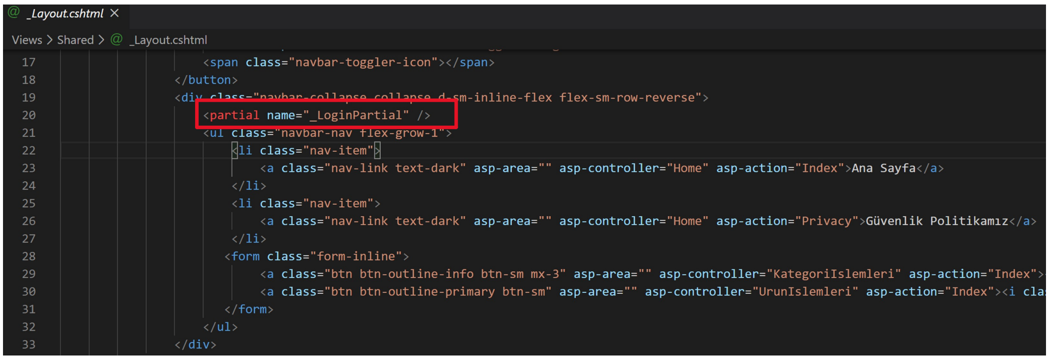
```
        // This method gets called by the runtime. Use this method to add services to the container.
        0 references
        public void ConfigureServices(IServiceCollection services)
        {
            services.AddDbContext<ApplicationDbContext>(options =>
                options.UseSqlite(
                    Configuration.GetConnectionString("DefaultConnection")));
            services.AddDefaultIdentity<IdentityUser>(options => options.SignIn.RequireConfirmedAccount = true
                .AddEntityFrameworkStores<ApplicationDbContext>();
            services.AddControllersWithViews();
           services.AddRazorPages();
        }

        // This method gets called by the runtime. Use this method to configure the HTTP request pipeline.
        0 references
        public void Configure(IApplicationBuilder app, IWebHostEnvironment env)
        {
            if (env.IsDevelopment())
            {
                app.UseDeveloperExceptionPage();
                app.UseDatabaseErrorPage();
            }
            else
            {
                app.UseExceptionHandler("/Home/Error");
                // The default HSTS value is 30 days. You may want to change this for production scenarios, see
                app.UseHsts();
            }
            app.UseHttpsRedirection();
            app.UseStaticFiles();

            app.UseRouting();

            app.UseAuthentication();
            app.UseAuthorization();

            app.UseEndpoints(endpoints =>
            {
                endpoints.MapControllerRoute(
                    name: "default",
                    pattern: "{controller=Home}/{action=Index}/{id?}");
                endpoints.MapRazorPages();
            });
        }
    }
}
```

- dotnet ef migrations add Kimlik
- dotnet ef database update

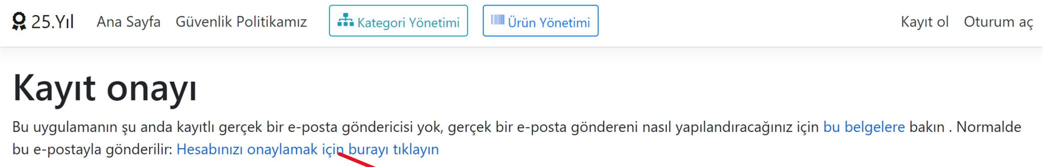


```csharp
using System;
using Microsoft.EntityFrameworkCore.Migrations;

namespace ETicaret.Migrations
{
    0 references
    public partial class Kimlik : Migration
    {
        0 references
        protected override void Up(MigrationBuilder migrationBuilder)
        {
            migrationBuilder.CreateTable(
                name: "AspNetRoles",
                columns: table => new
                {
                    Id = table.Column<string>(nullable: false),
                    Name = table.Column<string>(maxLength: 256, nullable: true),
                    NormalizedName = table.Column<string>(maxLength: 256, nullable: true),
                    ConcurrencyStamp = table.Column<string>(nullable: true)
                },
                constraints: table =>
                {
                    table.PrimaryKey("PK_AspNetRoles", x => x.Id);
                });

            migrationBuilder.CreateTable(
                name: "AspNetUsers",
                columns: table => new
                {
                    Id = table.Column<string>(nullable: false),
                    UserName = table.Column<string>(maxLength: 256, nullable: true),
                    NormalizedUserName = table.Column<string>(maxLength: 256, nullable: true)
                    Email = table.Column<string>(maxLength: 256, nullable: true),
                    NormalizedEmail = table.Column<string>(maxLength: 256, nullable: true),
                    EmailConfirmed = table.Column<bool>(nullable: false),
                    PasswordHash = table.Column<string>(nullable: true),
                    SecurityStamp = table.Column<string>(nullable: true),
                    ConcurrencyStamp = table.Column<string>(nullable: true),
                    PhoneNumber = table.Column<string>(nullable: true),
                    PhoneNumberConfirmed = table.Column<bool>(nullable: false),
                    TwoFactorEnabled = table.Column<bool>(nullable: false),
                    LockoutEnd = table.Column<DateTimeOffset>(nullable: true),
                    LockoutEnabled = table.Column<bool>(nullable: false),
                    AccessFailedCount = table.Column<int>(nullable: false)
                },
                constraints: table =>
                {
                    table.PrimaryKey("PK_AspNetUsers", x => x.Id);
                });
```



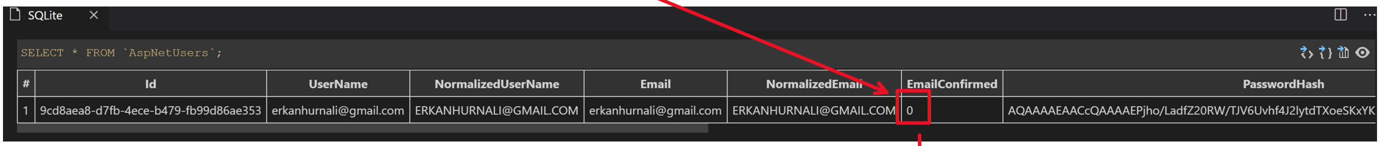


```html
                    <span class="navbar-toggler-icon"></span>
                </button>
                <div class="navbar-collapse collapse d-sm-inline-flex flex-sm-row-reverse">
                    <partial name="_LoginPartial" />
                    <ul class="navbar-nav flex-grow-1">
                        <li class="nav-item">
                            <a class="nav-link text-dark" asp-area="" asp-controller="Home" asp-action="Index">Ana Sayfa</a>
                        </li>
                        <li class="nav-item">
                            <a class="nav-link text-dark" asp-area="" asp-controller="Home" asp-action="Privacy">Güvenlik Politikamız</a>
                        </li>
                      <form class="form-inline">
                            <a class="btn btn-outline-info btn-sm mx-3" asp-area="" asp-controller="KategoriIslemleri" asp-action="Index">
                            <a class="btn btn-outline-primary btn-sm" asp-area="" asp-controller="UrunIslemleri" asp-action="Index"><i cla
                      </form>
                    </ul>
                </div>
```



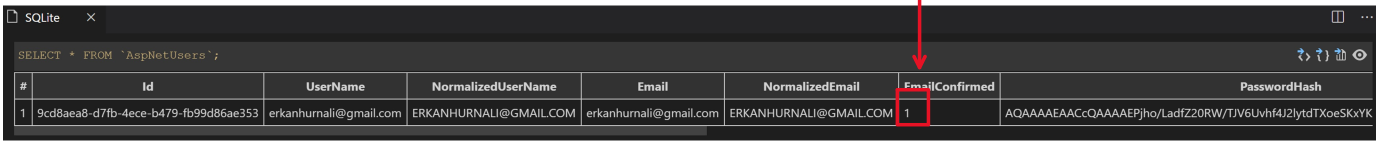
Kayıt onayı

Bu uygulamanın şu anda kayıtlı gerçek bir e-posta göndericisi yok, gerçek bir e-posta göndereni nasıl yapılandıracağınız için bu belgelere bakın . Normalde bu e-postayla gönderilir: Hesabınızı onaylamak için burayı tıklayın



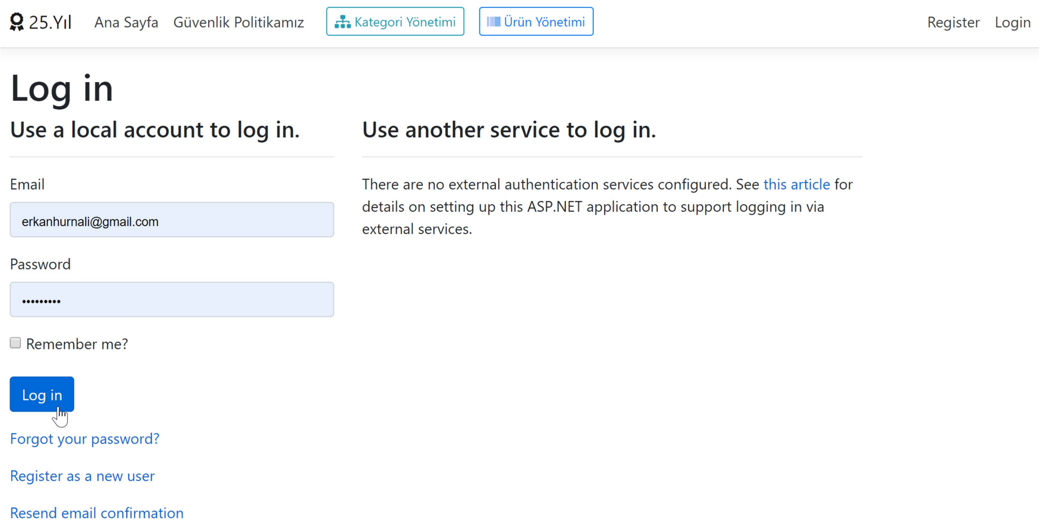
```sql
SELECT * FROM `AspNetUsers`;
```

#	Id	UserName	NormalizedUserName	Email	NormalizedEmail	EmailConfirmed	PasswordHash
1	9cd8aea8-d7fb-4ece-b479-fb99d86ae353	erkanhurnali@gmail.com	ERKANHURNALI@GMAIL.COM	erkanhurnali@gmail.com	ERKANHURNALI@GMAIL.COM	0	AQAAAAEAACcQAAAAEPjho/LadfZ20RW/TJV6Uvhf4J2lytdTXoeSKxYK



```sql
SELECT * FROM `AspNetUsers`;
```

#	Id	UserName	NormalizedUserName	Email	NormalizedEmail	EmailConfirmed	PasswordHash
1	9cd8aea8-d7fb-4ece-b479-fb99d86ae353	erkanhurnali@gmail.com	ERKANHURNALI@GMAIL.COM	erkanhurnali@gmail.com	ERKANHURNALI@GMAIL.COM	1	AQAAAAEAACcQAAAAEPjho/LadfZ20RW/TJV6Uvhf4J2lytdTXoeSKxYK



25.Yıl Ana Sayfa Güvenlik Politikamız Kategori Yönetimi Ürün Yönetimi Register Login

Log in

Use a local account to log in.

Email

erkanhurnali@gmail.com

Password

•••••••••

□ Remember me?

Log in

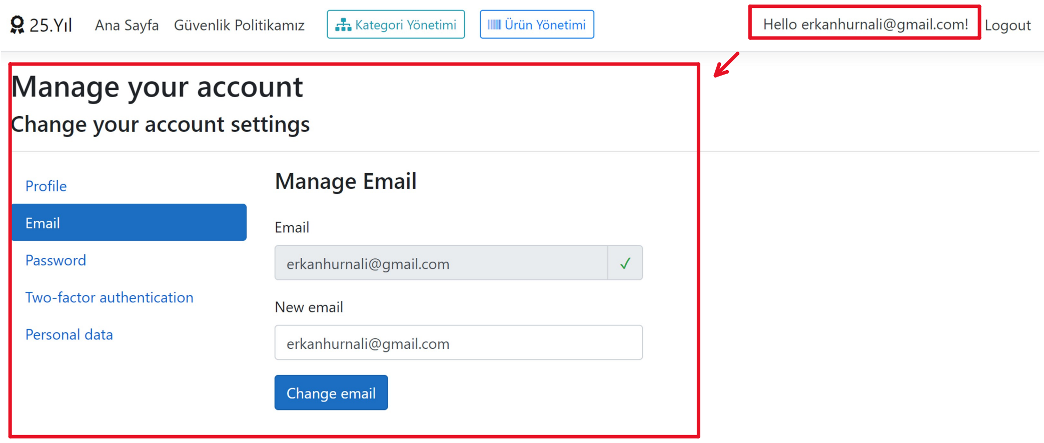
Forgot your password?

Register as a new user

Resend email confirmation

Use another service to log in.

There are no external authentication services configured. See this article for details on setting up this ASP.NET application to support logging in via external services.



25.Yıl Ana Sayfa Güvenlik Politikamız Kategori Yönetimi Ürün Yönetimi Hello erkanhurnali@gmail.com! Logout

Manage your account

Change your account settings

- Profile
- Email
- Password
- Two-factor authentication
- Personal data

Manage Email

Email

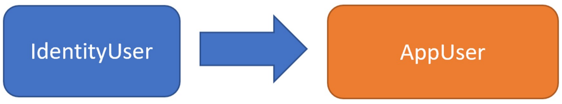
erkanhurnali@gmail.com ✓

New email

erkanhurnali@gmail.com

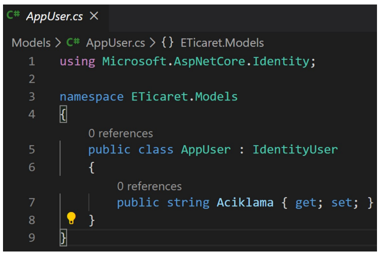
Change email

Temel Sınıflara Kalıtım Yoluyla Property Ekleme

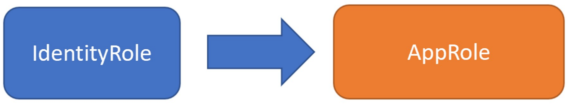

IdentityUser
AppUser

AppUser.cs

Models > AppUser.cs > {} ETicaret.Models

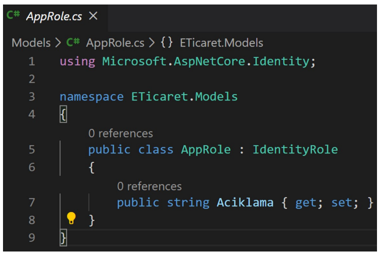


```
using Microsoft.AspNetCore.Identity;

namespace ETicaret.Models
{
    0 references
    public class AppUser : IdentityUser
    {
        0 references
        public string Aciklama { get; set; }
    }
}
```


IdentityRole
AppRole

AppRole.cs

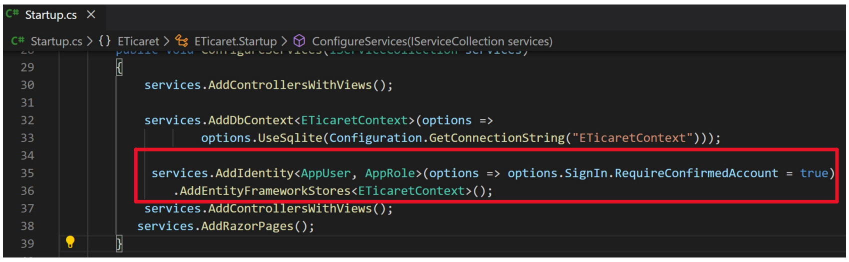
Models > AppRole.cs > {} ETicaret.Models



```
using Microsoft.AspNetCore.Identity;

namespace ETicaret.Models
{
    0 references
    public class AppRole : IdentityRole
    {
        0 references
        public string Aciklama { get; set; }
    }
}
```

Startup.cs

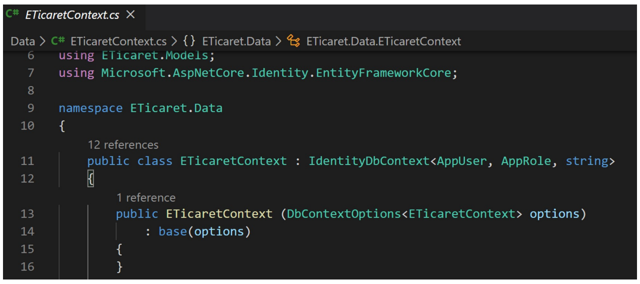
Startup.cs > {} ETicaret > ETicaret.Startup > ConfigureServices(IServiceCollection services)



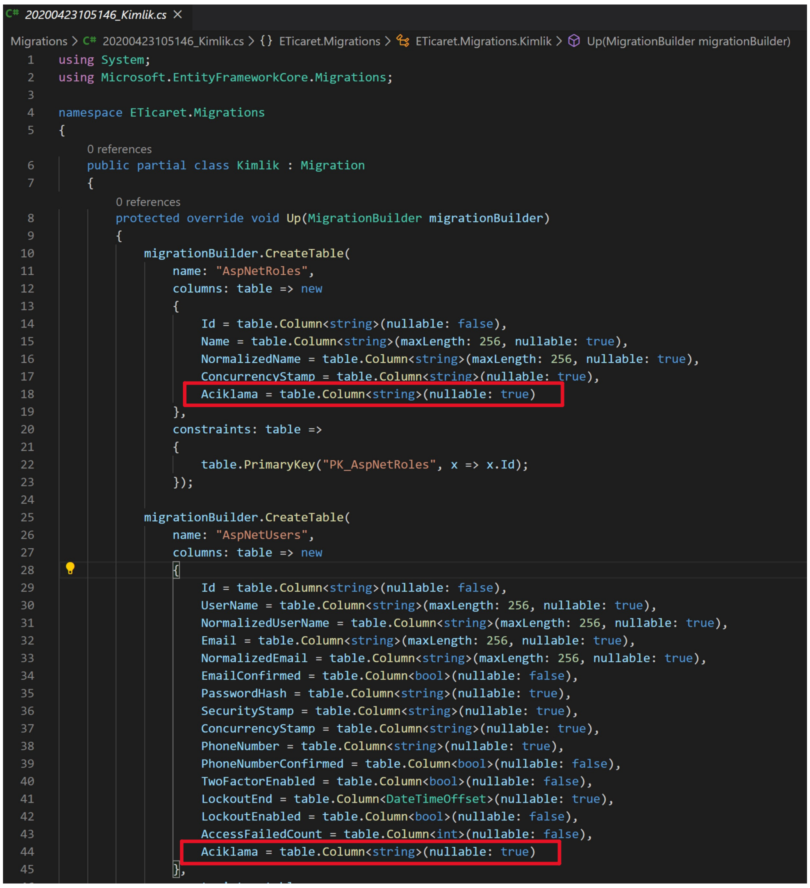
```
        public void ConfigureServices(IServiceCollection services)
        {
            services.AddControllersWithViews();

            services.AddDbContext<ETicaretContext>(options =>
                    options.UseSqlite(Configuration.GetConnectionString("ETicaretContext")));

             services.AddIdentity<AppUser, AppRole>(options => options.SignIn.RequireConfirmedAccount = true)
                 .AddEntityFrameworkStores<ETicaretContext>();
            services.AddControllersWithViews();
           services.AddRazorPages();
        }
```

ETicaretContext.cs

Data > ETicaretContext.cs > {} ETicaret.Data > ETicaret.Data.ETicaretContext

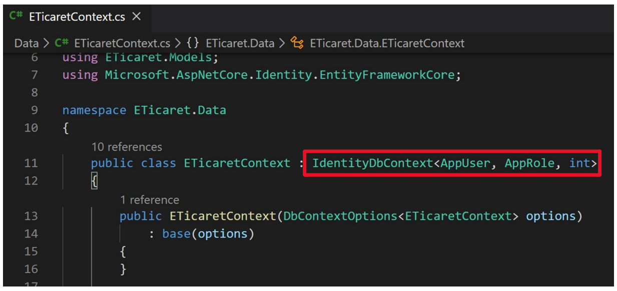


```
using ETicaret.Models;
using Microsoft.AspNetCore.Identity.EntityFrameworkCore;

namespace ETicaret.Data
{
    12 references
    public class ETicaretContext : IdentityDbContext<AppUser, AppRole, string>
    {
        1 reference
        public ETicaretContext (DbContextOptions<ETicaretContext> options)
            : base(options)
        {
        }
```

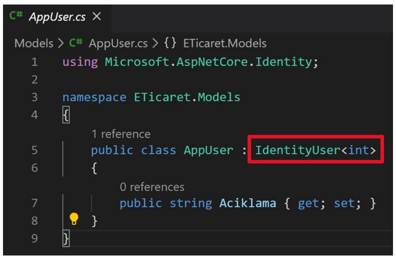

20200423105146_Kimlik.cs
Migrations > 20200423105146_Kimlik.cs > {} ETicaret.Migrations > ETicaret.Migrations.Kimlik > Up(MigrationBuilder migrationBuilder)
0 references
0 references

```
using System;
using Microsoft.EntityFrameworkCore.Migrations;

namespace ETicaret.Migrations
{
    public partial class Kimlik : Migration
    {
        protected override void Up(MigrationBuilder migrationBuilder)
        {
            migrationBuilder.CreateTable(
                name: "AspNetRoles",
                columns: table => new
                {
                    Id = table.Column<string>(nullable: false),
                    Name = table.Column<string>(maxLength: 256, nullable: true),
                    NormalizedName = table.Column<string>(maxLength: 256, nullable: true),
                    ConcurrencyStamp = table.Column<string>(nullable: true),
                    Aciklama = table.Column<string>(nullable: true)
                },
                constraints: table =>
                {
                    table.PrimaryKey("PK_AspNetRoles", x => x.Id);
                });

            migrationBuilder.CreateTable(
                name: "AspNetUsers",
                columns: table => new
                {
                    Id = table.Column<string>(nullable: false),
                    UserName = table.Column<string>(maxLength: 256, nullable: true),
                    NormalizedUserName = table.Column<string>(maxLength: 256, nullable: true),
                    Email = table.Column<string>(maxLength: 256, nullable: true),
                    NormalizedEmail = table.Column<string>(maxLength: 256, nullable: true),
                    EmailConfirmed = table.Column<bool>(nullable: false),
                    PasswordHash = table.Column<string>(nullable: true),
                    SecurityStamp = table.Column<string>(nullable: true),
                    ConcurrencyStamp = table.Column<string>(nullable: true),
                    PhoneNumber = table.Column<string>(nullable: true),
                    PhoneNumberConfirmed = table.Column<bool>(nullable: false),
                    TwoFactorEnabled = table.Column<bool>(nullable: false),
                    LockoutEnd = table.Column<DateTimeOffset>(nullable: true),
                    LockoutEnabled = table.Column<bool>(nullable: false),
                    AccessFailedCount = table.Column<int>(nullable: false),
                    Aciklama = table.Column<string>(nullable: true)
                },
```

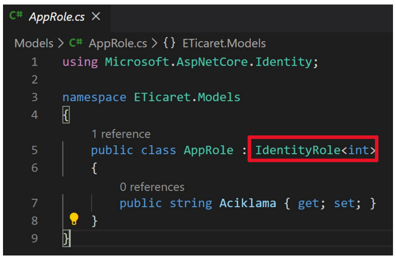
Varsayılan Primary Key Türünün Değiştirilmesi


ETicaretContext.cs
Data > ETicaretContext.cs > {} ETicaret.Data > ETicaret.Data.ETicaretContext
10 references
1 reference

```
using ETicaret.Models;
using Microsoft.AspNetCore.Identity.EntityFrameworkCore;

namespace ETicaret.Data
{
    public class ETicaretContext : IdentityDbContext<AppUser, AppRole, int>
    {
        public ETicaretContext(DbContextOptions<ETicaretContext> options)
            : base(options)
        {
        }
```



```
using Microsoft.AspNetCore.Identity;

namespace ETicaret.Models
{
    public class AppUser : IdentityUser<int>
    {
        public string Aciklama { get; set; }
    }
}
```



```
using Microsoft.AspNetCore.Identity;

namespace ETicaret.Models
{
    public class AppRole : IdentityRole<int>
    {
        public string Aciklama { get; set; }
    }
}
```

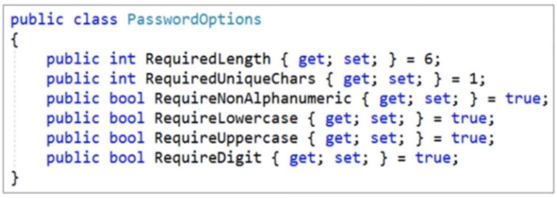
E-Posta Doğrulaması

```
using System.Net;
using System.Net.Mail;
using System.Threading.Tasks;
using Microsoft.AspNetCore.Identity.UI.Services;
public class EmailSender : IEmailSender
{
    public Task SendEmailAsync(string email, string subject, string htmlMessage)
    {
        MailMessage mail = new MailMessage();
        mail.IsBodyHtml = true;
        mail.To.Add(email);
        mail.From = new MailAddress("admin@optikanaliz.email", "Yönetim", System.Text.Encoding.UTF8);
        mail.Subject = subject;
        mail.Body = htmlMessage;
        mail.IsBodyHtml = true;
        SmtpClient smp = new SmtpClient();
        smp.Credentials = new NetworkCredential("admin@optikanaliz.email", "***");
        smp.Port = 587;
        smp.Host = "mail.optikanaliz.email";
        smp.EnableSsl = false;
        smp.Send(mail);
        return Task.CompletedTask;
    }
}
```

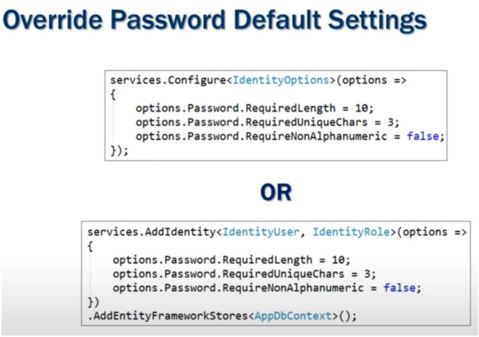
Şifre Ayarları

```
services.Configure<IdentityOptions>(options => {
        // Default Password settings.
         options.Password.RequireDigit = true;
        options.Password.RequireLowercase = true;
         options.Password.RequireNonAlphanumeric = true;
        options.Password.RequireUppercase = true;
        options.Password.RequiredLength = 6;
        options.Password.RequiredUniqueChars = 1;
});
```

Kaynak <https://docs.microsoft.com/tr-tr/aspnet/core/security/authentication/identity-configuration?view=aspnetcore-3.1>



```
public class PasswordOptions
{
    public int RequiredLength { get; set; } = 6;
    public int RequiredUniqueChars { get; set; } = 1;
    public bool RequireNonAlphanumeric { get; set; } = true;
    public bool RequireLowercase { get; set; } = true;
    public bool RequireUppercase { get; set; } = true;
    public bool RequireDigit { get; set; } = true;
}
```



Override Password Default Settings

```
services.Configure<IdentityOptions>(options =>
{
    options.Password.RequiredLength = 10;
    options.Password.RequiredUniqueChars = 3;
    options.Password.RequireNonAlphanumeric = false;
});
```

OR

```
services.AddIdentity<IdentityUser, IdentityRole>(options =>
{
    options.Password.RequiredLength = 10;
    options.Password.RequiredUniqueChars = 3;
    options.Password.RequireNonAlphanumeric = false;
})
.AddEntityFrameworkStores<AppDbContext>();
```