

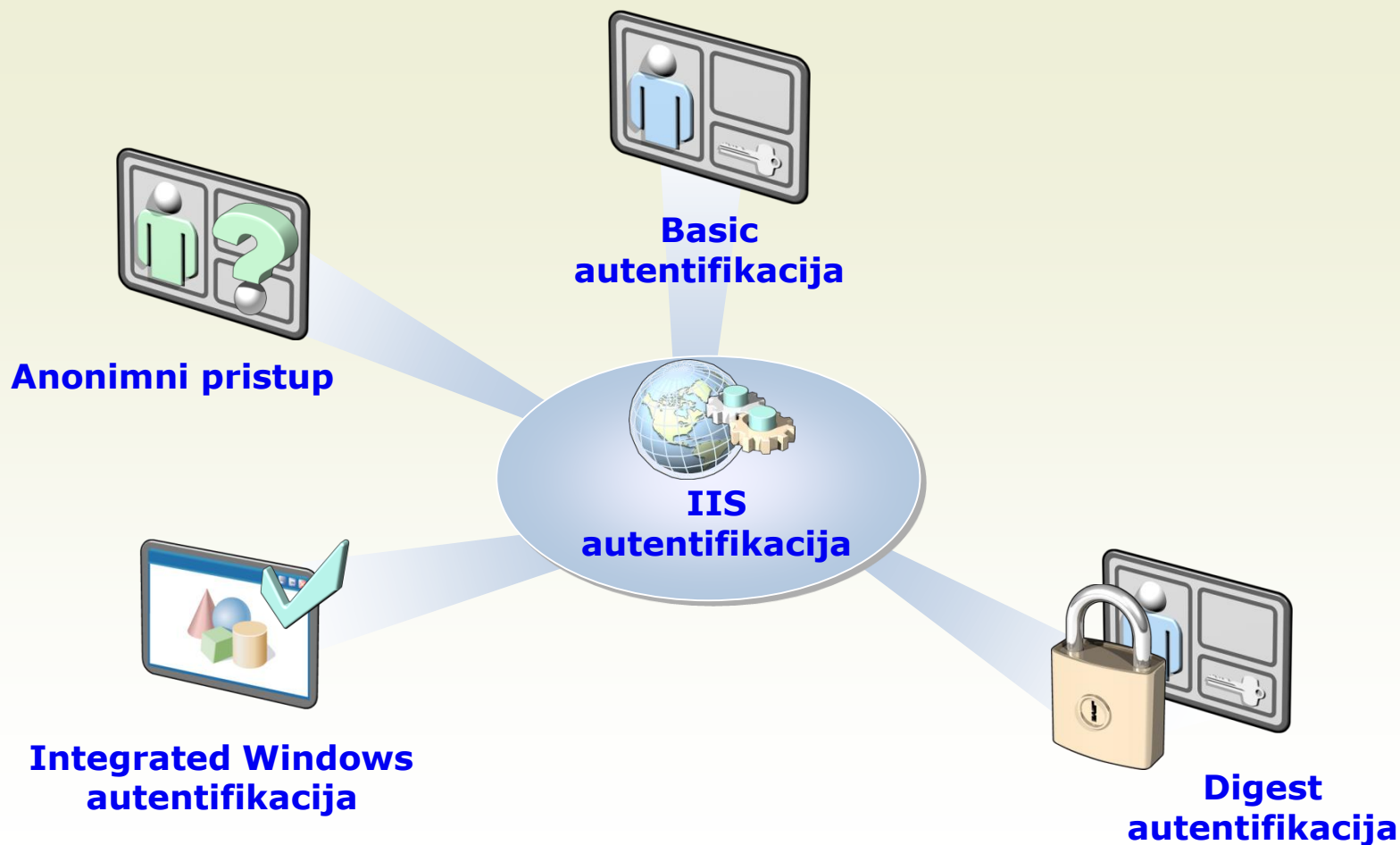
# MVC 5

## (Model-View-Controller)

### Autorizacija i autentifikacija

# IIS autentifikacija

ČILIŠTE



# Tipovi ASP.NET autentifikacije

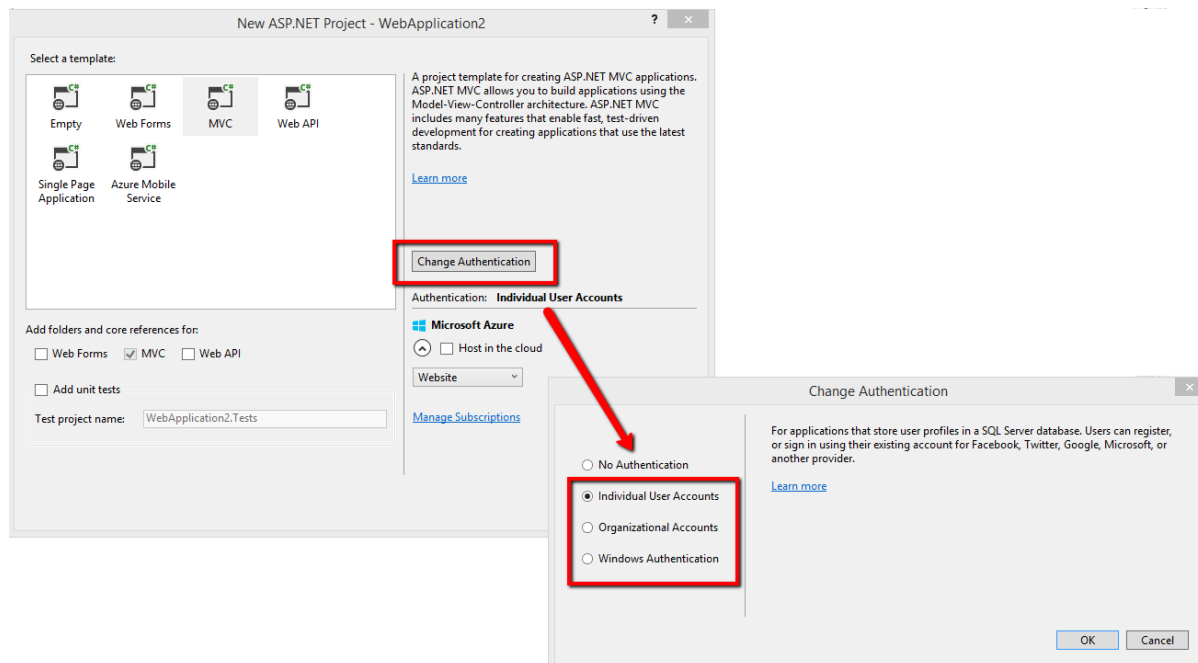
- ASP.NET autentifikacija omogućava korisnicima autentifikaciju (ovjeru pristupnih podataka) na web site-u
- ASP.NET podržava različite tipove autentifikacije:
  - Anonymous
  - ASP.NET Impersonation
  - Forms
  - Windows
- Windows autentifikacija autentificira korisnika na osnovu Windows user account-a
- Forms autentifikacija omogućuje programerima implementiranje prilagođene autentifikacije

# Što je ASP.NET Identity?

- Koristimo ASP.NET Identity za autentifikaciju korisnika prema prilagođenom data source-u
- ASP.NET podržava prijavu preko Facebook, Twitter, Google, LinkedIn i drugih računa poznatih socijalnih mreža
- ASP.NET Identity koristi Entity Framework Code First tehnologiju za implementiranje rada s bazom. Iz toga proizlazi da mijenjajući kod odgovarajućih C# klasa (tj. njezinih svojstava) koje koristi ASP.NET Identity možemo mijenjati strukturu baze u pozadini koja čuva podatke o korisnicima
- Podržane su Role (Admin, Korisnik, ...) uz pomoć kojih možemo upravljati grupama korisnika koji posjeduju istu rolu (ulogu)

# Opcija kod kreiranja projekta ...

- Kod kreiranja novog projekta odaberite jednu od željenih opcija autentifikacije ili ju kasnije konfigurirajte u Web.config datoteci
- Mi ćemo uglavnom se baviti opcijom – Individual User Account



# Prijava prvog korisnika

- Tijekom registracije prvog korisnika, se automatski kreira baza podataka unutar App\_Data direktorija, koji je prije bio prazan

The screenshot shows a web browser window with the address bar containing `http://localhost:27904/Account/Register`. The page title is "Register." and the subtitle is "Create a new account." The registration form includes fields for "Email", "Password", and "Confirm password", along with a "Register" button. The footer text is "© 2015 - My ASP.NET Application".

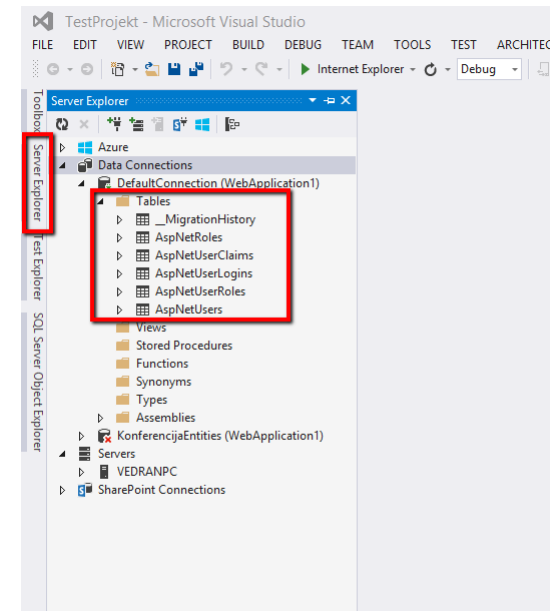
On the right side, a file explorer window displays the project structure. The "App\_Data" directory is expanded, showing two files: `aspnet-WebApplication1-20150915112549.mdf` and `aspnet-WebApplication1-20150915112549_log.ldf`, both of which are highlighted with a red box.

At the bottom, a table lists the files in the App\_Data directory:

Name	Date modified	Type	Size
aspnet-DataAnnotationDemo-20141115113300.mdf	18.09.2015. 15:56	SQL Server Databa...	3.136 KB
aspnet-DataAnnotationDemo-20141115113300_log.ldf	18.09.2015. 15:56	SQL Server Databa...	1.024 KB

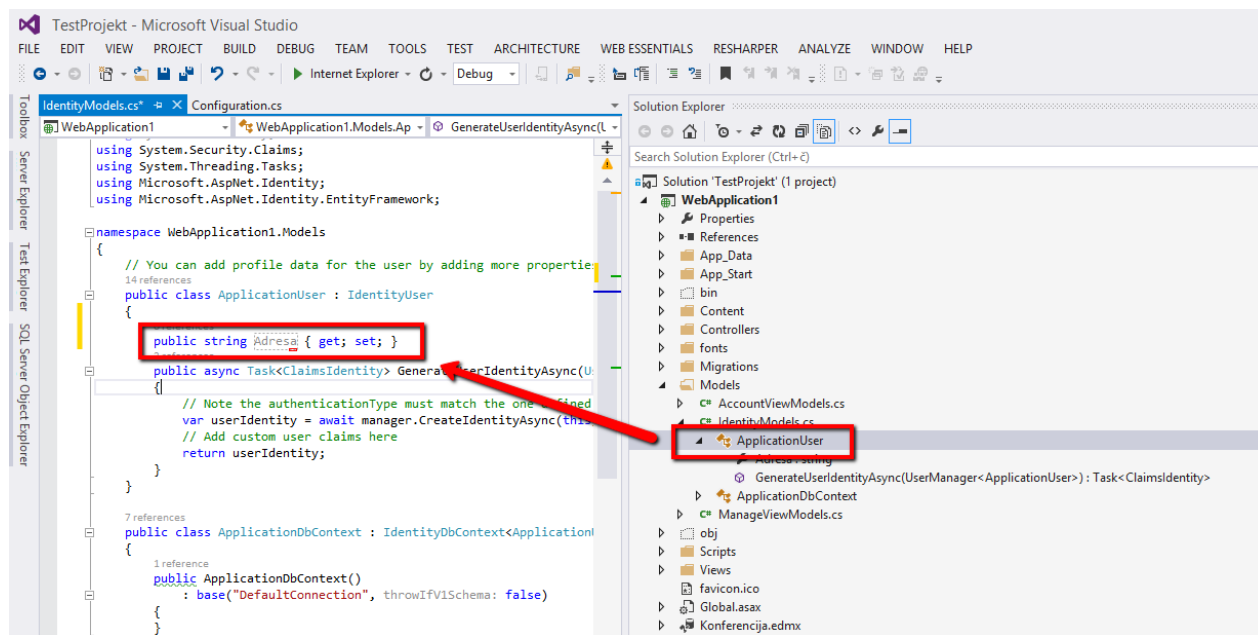
# Kako vidjeti objekte baze?

- Tablice kreirane u bazi možemo vidjeti kroz DefaultConnection podatkovnu konekciju (Data Connection) unutar Server Explorera
- Nemojte mijenjati dizajn baze direktno kroz Server Explorer ili pomoću SQL Server Management Studia, jer ćete takve promjene izgubiti kod slijedećeg generiranja baze – CodeFirst EF tehnologija



# Kako proširiti ASP.NET Identity (I)

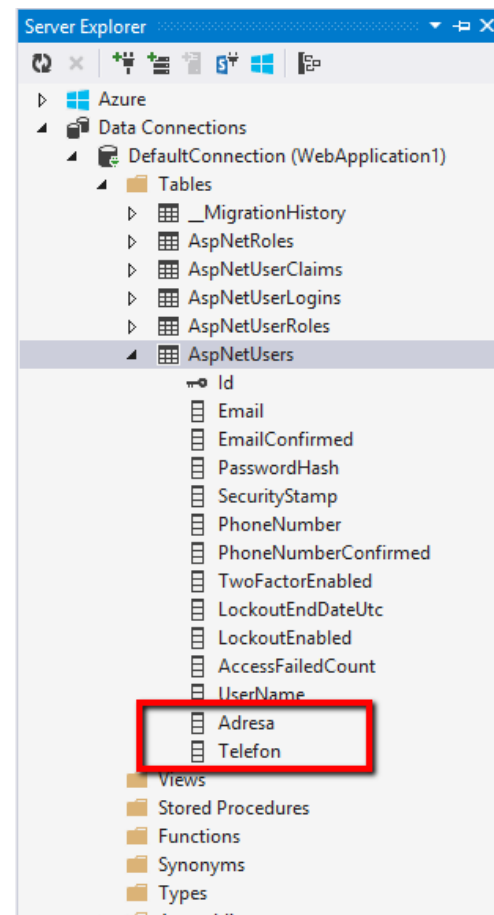
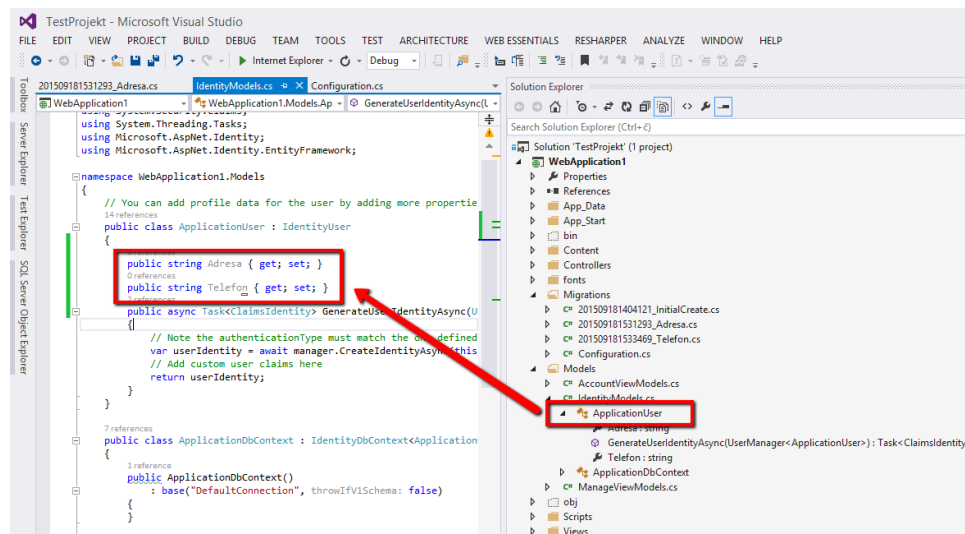
- ASP.NET Identity proširujemo slijedećim koracima:
  - Otvorite NuGet konzolu:  
Tools -> NuGet Package Manager -> Package Manager Console
  - Izvršite naredbu:  
PM> Enable-Migrations –EnableAutomaticMigrations
  - Dodajte svojstvo Adresa klasi ApplicationUser:





# Kako proširiti ASP.NET Identity (II)

- ... nastavak sa prethodne stranice ...
- Dodajte svojstvo Adresa klasi ApplicationUser
- Izvršite naredbe:  
PM> Add-Migration Adresa  
PM> Update-Database
- Ponovite korake na ovoj stranici za svojstvo Telefon
- Provjerite rezultat u Server Exploreru ...



# HttpContext.User objekt

- Na nivou aplikacije možemo bilo gdje u kodu dohvatiti podatke o trenutno logiranom korisniku preko objekta HttpContext.User koji ima neke od slijedećih svojstava i metoda:

`HttpContext.User.IsInRole("Admin")`

`HttpContext.User.Identity.Name`

`HttpContext.User.Identity.IsAuthenticated`

# Uloge (role)

- Na nivou aplikacije možemo definirati proizvoljan broj uloga/rola
- Korisnicima aplikacije možemo pridodijeliti pripadnost proizvoljnom broju rola
- Moguće je kreirati vlastite administracijske stranice za upravljanje korisnicima i rolama
- Mi ćemo promjene raditi direktno na bazi

# Authorize atribut

- Primjenom ovog atributa na cijelu Controller klasu ili na određenu akcijsku metodu definiramo koji korisnik ili koja rola je autorizirana za pristup toj metodi
- Višestruka primjena Authorize atributa će se ponašati kao da je među njima logička AND operacija
- Ako želimo simulirati OR operaciju moramo proširiti/prilagoditi Authorize atribut

# Primjeri Authorize atributa

- Ako želimo da prava na akcijsku metodu imaju samo korisnici koji imaju pridodijeljene role Admin i Editor, onda to definiramo ovako:  
`[Authorize(Roles = "Admin, Editor")]`
- Ako želimo da prava na akcijsku metodu imaju samo određeni korisnici onda to definiramo ovako:  
`[Authorize(Users = "vzdesic@gmail.com,vzdesic@test.com")]`

# Primjena Authorize atributa

**[Authorize(Roles = "Admin")]**

```
public class AdministracijaController : Controller {  
    //Ovo može izvršiti bilo koji user sa Admin rolom  
    public ActionResult Index() {  
        return View();  
    }  
}
```

//Ovo može izvršiti samo navedeni user: vzdesic@gmail.com

**[Authorize(Users = "vzdesic@gmail.com")]**

```
public ActionResult KreirajRolu() {  
    return View();  
}  
}
```

# Proširivanje Authorize atributa

- Ako želimo promijeniti defaultno ponašanje Authorize atributa, to je moguće uvođenjem nove klase, koja nasljeđuje Authorize atribut
- Potrebno je override-ati metodu AuthorizeCore ako želimo promijeniti sam način i pravila autorizacije
- Metoda vraća true ako su prava i autorizacija potvrđena, a inače false
- Primjer na slijedećem slide-u ...

# Proširivanje Authorize atributa (II)

```
public class PrilagodeniAuthorize : AuthorizeAttribute
{
    protected override bool AuthorizeCore(HttpContextBase httpContext) {
        foreach (var item in Roles.Split(',')) {
            if (httpContext.User.IsInRole(item))
                return true;
        }

        if(httpContext.User.Identity.Name != "" &&
            Users.Contains(httpContext.User.Identity.Name))
            return true;

        return false;
    }
}
```



# Active Directory Security

- Korisnik može također koristiti Windows račun za logiranje u aplikaciju
- Ova se opcija najčešće koristi u okviru korporacijskih Intranet aplikacija
- Može se postaviti odabirom "Windows" autentifikacijskog moda tijekom kreiranja projekta ili kasnije u Web.config datoteci