**1. Create the project.**

**2. Clean the project.**

**3. Setup the password validation settings from IdentityConfig.cs.**

**4. Connection string in Web.config.**

AttachDbFilename=|DataDirectory|\aspnet-CookiesSystem.Web-20151105044018.mdf се изтрива.

пример:

<connectionStrings>

<add name="DefaultConnection" connectionString="Data Source=.;AttachDbFilename=|DataDirectory|\aspnet-CookiesSystem.Web-20151105044018.mdf;Initial Catalog=aspnet-CookiesSystem.Web-20151105044018;Integrated Security=True" providerName="System.Data.SqlClient" />

</connectionStrings>

**5. Register user**

Header Content-Type: application/json

POST

{

"email":"mail@abv.bg",

"password":"123",

"confirmPassword":"123"

}

**6. Login user**

Route: /Token => може да се промени в Startup.Auth.cs. От тук се настройва и AccessTokenExpireTimeSpan.

Данните се подават като query string. В постман може да се ползва x-www-form-urlencoded

username: email

password: \*\*\*\*

grant\_type: "password"

Това връща токена.

**7. Прехвърлям файла ApplicationUser от IdentityModels в проекта с моделите на базата.**

В този клас може да се добавят още пропъртита.

Трябва да се добави инсталирания вече пакет Microsoft.Asp.Identity.Entity.Framework в проекта с моелите, за да се реферира IdentityUser.

**8. Преместваме ApplicationDbContext в Data проета.**

В проекта с Data за DbContext също трябва да имаме реферирано Microsoft.Asp.Identity.Entity.Framework.

**9. Добавяме нов public static клас DatabaseConfig в App\_Start**

пример:

public static class DatabaseConfig

{

public static void Initialize()

{

Database.SetInitializer(new MigrateDatabaseToLatestVersion<CookiesSystemDbContext, Configuration>());

CookiesSystemDbContext.Create().Database.Initialize(true);

}

}

След това извикваме метода в Global.asax

**10. Някои по-важни атрибути:**

[Authorize]

[AllowAnonymous]

[Route("asd")]

[Compare("NameOfOtherPropery", "Some error message")]

**11. При ауторизация в хедъра се добавя:**

Authorization: "Bearer {Token}"

**12. IRepository и EfGenericRepository се добавят в DbName.Data (Проекта в, който е DbContext).**

**13. Прави се отделен проект Name.Services, в който се слагат всички сървиси.**

Най-лесно там да се направи папка Contracts, в която да се сложат интерфейси на всички services.

**14. Може да се създаде отделен проект Name.Common, където да се сложат класове с публични константи.**

**15. Инсталираме ninject.**

\* Пишем в търсачката ninject.web.webapi.webhost и инсталираме това нещо.

\* Инсталиранме всички ъпдейти към ninject.

**16. Попълваме дипендънситата в App\_Start/NinjectWebCommon.cs в метода RegisterServices.**

пример:

private static void RegisterServices(IKernel kernel)

{

kernel.Bind<ICookiesService>().To<CookiesService>();

kernel.Bind(typeof(IRepository<>)).To(typeof(EfGenericRepository<>));

kernel.Bind<CookiesSystemDbContext>().To<CookiesSystemDbContext>();

}