# **NOTLAR**

* Mesela bir id ile update işlemi gerçekleştirmek istiyoruz diyelim. Bu id database’ de kayıtlı değilse o id hatasının bana dönülmesini istiyorum. Bu gibi durumları kontrol edebilmemiz için WorkService’ dan direkt dto’ ları dönmek yerine bir obje dönmemiz lazım. Buna da Result Pattern deniyor.
* IdentityDbContext özünde DbContext’ ten kalıtsal yollarla geçmiştir.
* User Manager, kullanıcı oluşturabilir.
* Role Manager, rol oluşturabilir.
* Sign In Manager, ilgili kullanıcının giriş çıkış işlemlerini gerçekleştirebilir.
* Authentication: Giriş yetkisi. Girebilir mi, giremez mi?

Örnek;

Bir öğrenci, bir öğretmen, bir personelin okula girmeye yetkisi var ama herhangi birinin yok.

* Authorization: İlgili kişinin nerelere girme hakkı var?

Örnek;

Bir öğrencinin sınıfa girmeye hakkı var ama öğretmenler odasına girmeye hakkı yok.

* Claims: Kişiye özgü bilgileri tutmak için kullanılır.

Örnek;

En sevdiğiniz renk? gibi…

* Authentication ile authorization’ ın farkı mesela öğretmenin hem öğretmenler odasına hem sınıfa girme yetkisi var ama öğrencinin sadece sınıfa girme yetkisi var. Gerekli hakları authorization’ da veririz. Authentication’ da sadece okula girip giremeyeceğine bakılır.
* Microsoft Asp Net Core Identity kütüphanesinde authentication ve authorization kullanabildiğimiz gibi aynı zamanda bu kütüphaneyi dilediğimiz gibi manipüle edebiliriz. Çünkü bizim Context de dahil birçok nesnemiz Generic.