**04.04.2023**

İlgili günde Bulut Depolama projesine kullanıcı yapısı kazandırılmıştır. Geliştirilen bu kullanıcı yapısı sayesinde kullanıcılar sistemde oturum açarak giriş yapabilecek ve ilerleyen günlerde yapılacak olan geliştirmelerle beraber sistemde kendine ait dosya tanımı olması sağlanabilecektir.

Kullanıcıların Oturumlandırma (Authentication) işlemi bir önceki gün Spring Boot projesine entegre edilen Spring Security paketi aracılığıyla yapılmıştır. Bu sayede gerekli şifreleme algoritmaları ve session-cookie yönetimi modüler bir şekilde projeye dahil edilebilmiştir.

Kullanıcı yapısının oluşturulabilmesi için öncelikle Entity katmanında her biri kullanıcıyı nesne olarak niteleyecek User isminde bir sınıf oluşturulmuştur. Bu sınıfa verilen @Entity ve @Table(“users”) anotasyonları sayesinde kullanıcı kaydetme ve kullanıcının sisteme kayıtlı olma durumunun kontrolü işlemleri Hibernate tarafında veritabanında “users” tablosunda gerçekleşecektir. Veritabanında bir kullanıcıya ait olarak ID,username,password bilgileri saklanacaktır. Bu doğrultuda User sınıfında aşağıdaki parametrelerle ilgili sütunlarla alakalı field’lar oluşturulmuştur. Bunun yanında bu fieldların diğer sınıflar tarafından kullanılabilmesi için getter-setter metotlar da tanımlanmıştır.

@Id

@GeneratedValue(strategy = GenerationType.IDENTITY)

private Long id;

@Column(nullable = false, unique = true, length = 45)

private String username;

@Column(nullable = false, length = 64)

private String password;

Veritabanı üzerinde sorgular çalıştırabilmek için Repository katmanı oluşturulup bu katman içerisinde UserRepository isminde JpaRepository sınıfını kalıtım alan bir interface (arayüz) oluşturulmuştur. Kullanıcıların veritabanında kayıtlı olup olmadığını username bilgisi ile sorgulamak için aşağıdaki tanımlama interface içerisinde yapılmıştır.

@Query("SELECT u FROM User u WHERE u.username = ?1")

public User findByUsername(String email);

Spring Security, bir kullanıcıya ait hesabın özelliklerini tutabilmek için UserDetails sınıfını implemente eden bir sınıf beklemektedir. Bu sınıf içerisinde hesabın aktif olma, kitli olma gibi kriterleri tutulmaktadır. Bunun için CustomUserDetails isminde bir sınıf oluşturulup içerisinde interface’in zorunluğu tuttuğu fieldlar ve fonksiyonlar tanımlanmıştır.

Bir kullanıcıya ait hesap bilgilerinin Spring Security tarafından nesne olarak okunabilmesi için bir servis tanımlanmıştır. Bu servis UserDetailsService arayüzünü implemente edecek şekilde CustomUserDetailsService isminde sınıf olarak oluşturulmuştur. Bu sınıf içerisinde username bilgisi ile hesap bilgilerinin alınabilmesi için aşağıdaki fonksiyon tanımı yapılmıştır.

@Override

public UserDetails loadUserByUsername(String username) throws UsernameNotFoundException {

User user = userRepo.findByUsername(username);

if (user == null) {

throw new UsernameNotFoundException("User not found");

}

return new CustomUserDetails(user);

}