Java Spring Boot Projemizi H2 Database ile Kullanma

Bu yazı sayesinde hazır olan Java Spring Boot projemizi, H2 Database ile birlikte test edeceğiz. Bunun için öncelikle bilgisayarımıza h2database.com sitesi üzerinden gerekli yüklemeyi yapmamız gerekiyor. Sonrasında Java Spring Boot ile birlikte kullanacağımız için gerekli dependency eklemeleri yapacağız.

Dependency Eklemeleri;

```
<dependency>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-data-jpa</artifactId>
</dependency>
<dependency>
    <groupId>com.h2database</groupId>
    <artifactId>h2</artifactId>
    <scope>runtime</scope>
</dependency>
```

Bu 2 dependency'i projemize ekliyoruz. Sonrasında application.properties dosyasına gerekli girdileri girmemiz gerekiyor. Bunlar;

```
spring.h2.console.enabled=true
spring.datasource.url=jdbc:h2:mem:testdb
spring.datasource.driverClassName = org.h2.Driver
spring.datasource.username=sa
spring.datasource.password=1234
spring.jpa.database-platform=org.hibernate.dialect.H2Dialect
```

Burada username ve password istediğiniz gibi girebilirsiniz. H2 Console kısmında bu önemli olacak. Bilgisayarımıza H2 kurduktan sonra Windows arama çubuğundan veya Spring Boot projemizi çalıştırdıktan sonra **http://localhost:8080/h2-console/** ile ulaşabiliriz.

Bunları yaptıktan sonra bir model (data) class oluştururuz.

```
@Data
@AllArgsConstructor
@NoArgsConstructor
@Entity
public class User {

    @Id
    @GeneratedValue(strategy = GenerationType.AUTO)
    @Column(name = "ID", nullable = false)
    private Integer id;
    @Enumerated(EnumType.STRING)
    private UserType userType; // bireysel & kurumsal & yeniTip
    private String name;
    private String email;
    private String photo;
    private String bio;
```

Burada @Id önemli ve yapılması gereken bir değişken olarak bizi karşılıyor. Sınıf yapımıza göre istenilen anotasyonları uyguluyoruz. Sonrasında database içerisine spesifik olarak istediğimiz değişkenleri seçebiliriz veya tüm değişkenleri kullanarak User sınıfını database içerisine alabiliriz.

Ben bu örnek için;

```
public User(UserType userType, String name, String email) {
    super();
    this.userType = userType;
    this.name = name;
    this.email = email;
}
```

Yukarıda bulunan constructor'ı kullanacağım. Bu constructor içerisinde userType, name ve email değişkenleri bulunuyor. UserType Sınıfı bir enum class olarak tanımlandı.

```
public enum UserType {

    CORPORATE,
    INDIVIDUAL

}
```

Bu işlemleri yaptıktan sonra artık test için bir User oluşturmak ve bunu database içerisinde görmek kaldı.

Bunun için main sınıfımız içerisinde işlemler yapacağız.

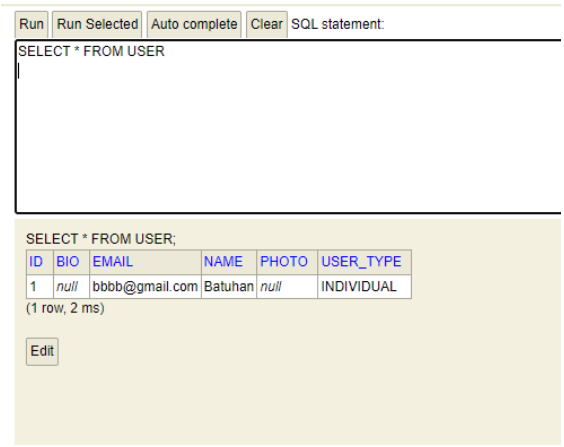
```
public static void main(String[] args) {


    ConfigurableApplicationContext configurableApplicationContext = SpringApplication.run(EmlakBuradaApplication.class, args);

    UserRepository userRepository = configurableApplicationContext.getBean(UserRepository.class);


    UserType testUserType = UserType.INDIVIDUAL;
    User testUser = new User(testUserType,"Batuhan","bbbb@gmail.com");
    userRepository.save(testUser);
}
```

Test için artık bir User oluşturduk. Spring Boot projemizi çalıştırdığımız zaman database içerisine oluşturduğumuz User kişisi oluşmuş olacak. Bu testi controller yazarak Postman ile test edebiliriz.



Görüldüğü üzere H2 console ile birlikte SELECT * FROM USER sorgusu ile oluşturduğumuz User gözüküyor. 😊