

3. Derste yapılan emlakcepte uygulamasını Singleton ve Factory pattern'larını uygulayın. Oluşturulan objeler bu pattern'ler ile oluşturulmalıdır.(Spring içerisinde BeanFactory gibi davranmasını bekliyoruz) (20 PUAN)

User, Realty, UserDao, RealtyDao, UserService, RealtyService sınıfları oluşturuldu. Sınıflara singleton design pattern uygulandı.

```
1 package service;
2
3 import Abstract.Entity;
4
5 public class UserService implements Entity{
6     private static UserService userServiceInstance;
7
8     private UserService() {}
9
10    public static UserService getInstance() {
11        if(userServiceInstance==null) {
12            userServiceInstance=new UserService();
13        }
14        return userServiceInstance;
15    }
16 }
17
```

Nesne oluşturmada kullanılacak olan nesnesi oluşturulacak bütün sınıf tiplerini barındıran InstanceType isimli bir Enum oluşturuldu.

```
1 package instanceCreation;
2
3 public enum InstanceType {
4     REALTY,USER,REALTYDAO,USERDAO,REALTYSERVICE,USERSERVICE
5 }
6
```

Factory Design ile nesne oluşturacak olan InstanceCreator sınıfı tanımlandı. Bu sınıfa da singleton design pattern uygulanarak tek bir örneği oluşacak şekilde güncellendi.

```
public class InstanceCreator {
    private static InstanceCreator instanceCreatorInstance;

    private InstanceCreator() {}

    public static InstanceCreator getInstance() {
        if(instanceCreatorInstance==null) {
            instanceCreatorInstance=new InstanceCreator();
        }
        return instanceCreatorInstance;
    }
}
```

Nesne oluşturmada referans tutucu olarak kullanılmak üzere bir Entity arayüzü oluşturuldu. Nesnesi oluşturulacak olan bütün sınıflar bu arayüzü implemente ettirildi.

```
1 package Abstract;
2
3 public interface Entity {
4
5 }
6
```

InstanceCreator sınıfı içerisine dönüş tipi Entity (bütün sınıfları dönebilir) olan parametre olarak InstanceType enumu alan factoryMethod isimli bir metot tanımlandı. Metot girilen enum bilgisine göre istenilen sınıfın nesnesini dönmektedir.

```
11 public class InstanceCreator {
12     private static InstanceCreator instanceCreatorInstance;
13
14     private InstanceCreator() {}
15
16     public static InstanceCreator getInstance() {
17         if(instanceCreatorInstance==null) {
18             instanceCreatorInstance=new InstanceCreator();
19         }
20         return instanceCreatorInstance;
21     }
22
23     public Entity factoryMethod(InstanceType instanceType) {
24         Entity entity=null;
25         switch(instanceType) {
26             case REALTY:
27                 entity=Realty.getInstance();
28                 break;
29             case USER:
30                 entity=User.getInstance();
31                 break;
32             case REALTYDAO:
33                 entity=RealtyDao.getInstance();
34                 break;
35             case USERDAO:
36                 entity=UserDao.getInstance();
37                 break;
38             case REALTYSERVICE:
39                 entity=RealtyService.getInstance();
40                 break;
41             case USERSERVICE:
42                 entity=UserService.getInstance();
43                 break;
44         }
45         return entity;
46     }
47 }
48
49
```

Main üzerinde factoryMethod denemeleri yapıldı. Aynı tip için nesne talebi geçildiğinde hep aynı örneği döndü.

```
Main.java X
1 import Abstract.Entity;
2
3
4
5
6 public class Main {
7
8     public static void main(String[] args) {
9
10         InstanceCreator instanceCreator=InstanceCreator.getInstance();
11         System.out.println(instanceCreator.toString());
12
13         InstanceCreator instanceCreator2=InstanceCreator.getInstance();
14         System.out.println(instanceCreator2.toString());
15
16         Entity model=instanceCreator.factoryMethod(InstanceType.REALTY);
17         System.out.println(model.toString());
18
19         Entity model2=instanceCreator.factoryMethod(InstanceType.REALTY);
20         System.out.println(model2.toString());
21
22         Entity realtyDao=instanceCreator.factoryMethod(InstanceType.REALTYDAO);
23         System.out.println(realtyDao.toString());
24
25         Entity userService=instanceCreator.factoryMethod(InstanceType.USERSERVICE);
26         System.out.println(userService.toString());
27     }
28 }
29
30
```

Problems Javadoc Declaration Console X

<terminated> Main [Java Application] /Users/mariami/.p2/pool/plugins/org.eclipse.justj.openjdk.hotspot.jre.full.macosx.x86\_64\_17.0.4.v20221004-1257

instanceCreation.InstanceCreator@3f8f9dd6  
instanceCreation.InstanceCreator@3f8f9dd6  
model.Realty@1edf1c96  
model.Realty@1edf1c96  
dao.RealtyDao@6996db8  
service.UserService@7fbe847c