BBM488 Web Servisleri Laboratuvarı – Ödev 4

AD= Umut Öztürk

NUMARA= 21328394

GRUP NUMARASI= 5

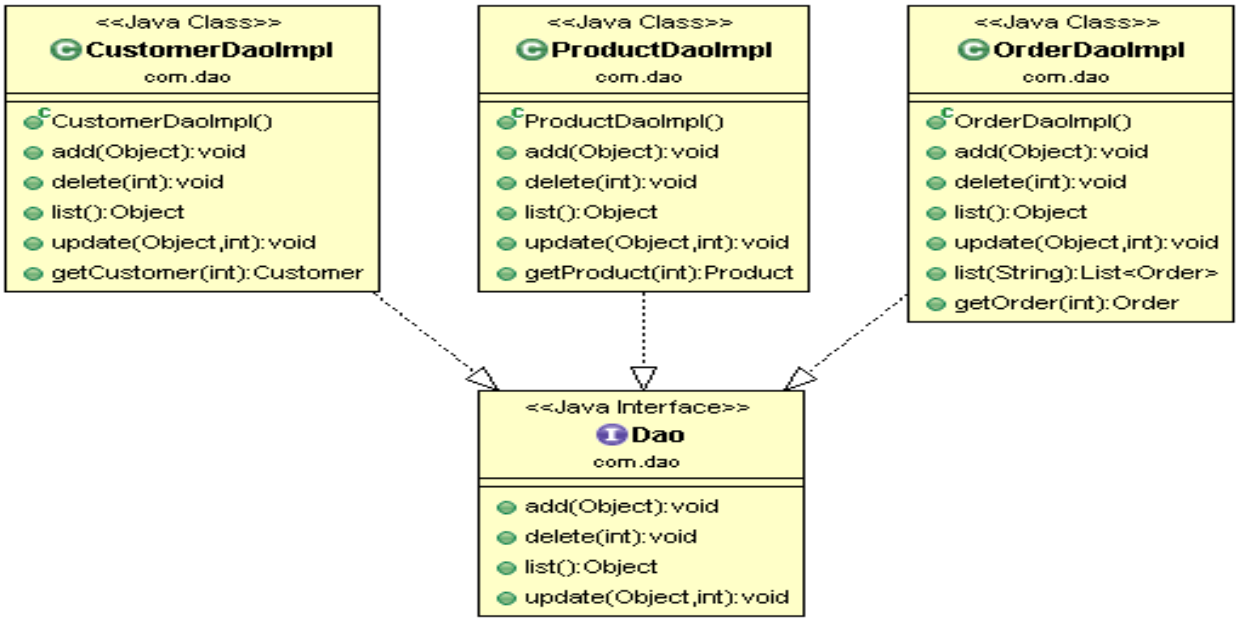
KONU= Spring MVC , Hibernate ve Angular JS

DERS SORUMLULARI : Dr. Gönenç Ercan, Dr. Ali Seydi Keçeli, Dr. Aydın Kaya

TESLİM TARİHİ= 10.05.2017

1. Sınıf çizeneği (class diagram)

Dao sınıfları için class diagramları:



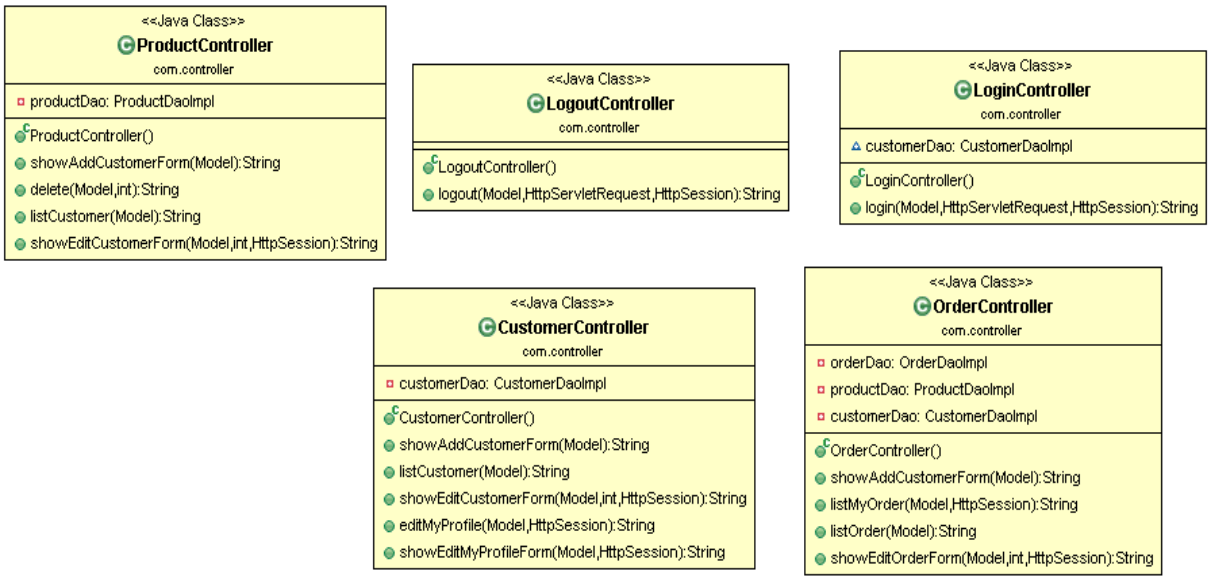
Sistemde kayıtlı olan nesneler (order,product,customer) için veritabanına ekleme ,çıkarma ve güncelleme yapan sınıflar yaratıldı.Bu sınıflar adeta veritabanı işlemlerini yapan sınıflara benzetebiliriz. Zaten bu sınıflar Hibernate ile ilişkisi bulunmaktadır.Bu ilişkiyi kullanarak nesneler üzerinde ekleme,silme,güncelleme ve listeleme gibi işlemleri gerçekleştirebilirler.

Model için class diagramları:

Bu model sınıflar projede rol oynayan objeleri temsil eder.Bu sınıflar sadece değişkenler ve o değişkenlere ait getter ve setter methodlara sahiptirler.Onun dışında hiçbir işlevleri yoktur.Dikkat edilmesi gereken diğer nokta ise veritabanı tasarımı sebebiyle aralarında oluşan ilişkiler gösterilmiştir.Bir siparişin sadece 1 adet müşterisi ve ürünü olabiliyor.Fakat bir müşterinin ise birden fazla siparişi olabiliyor.Diğer önemli bir nokta ise bazı özel notasyonlar sayesinde bu model sınıfları veritabanında örnek şema/şablon olarak haritalandırabiliyor(persist).Bu sayede SQL kodu yazılmadan o notasyonlar sayesinde veritabanı tablolarınızı oluşturabiliyorsunuz. Aşağıdaki resimden bu ilişkileri detaylıca görebilirsiniz:

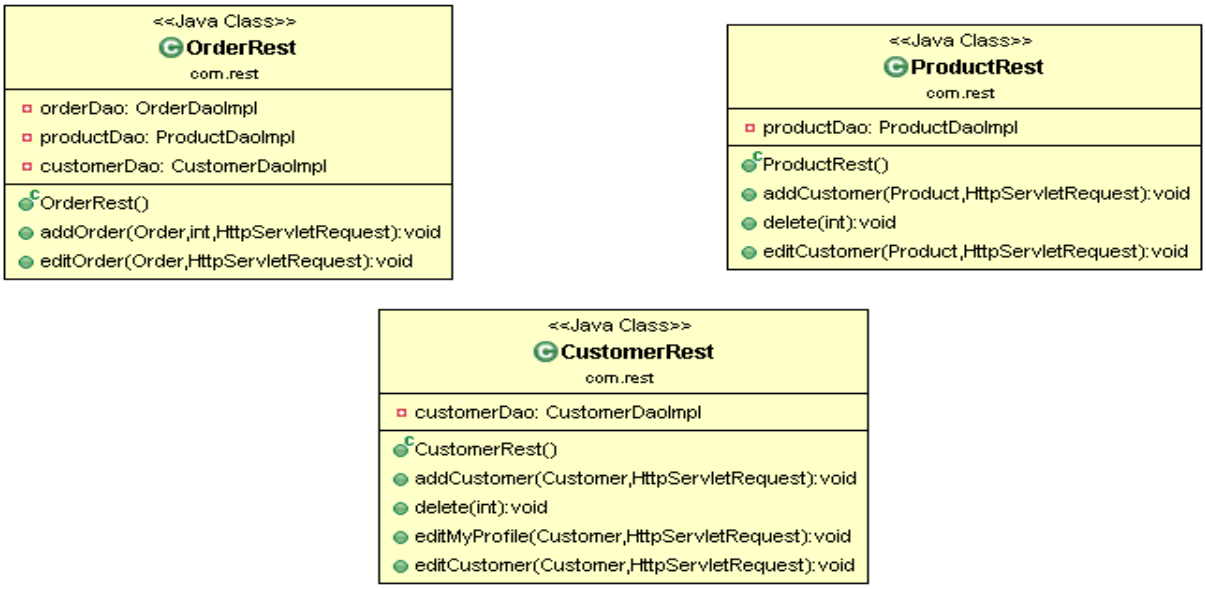

<<Java Class>>
Order
com.model
order_id: int
orderState: String
numberOfProduct: String
Order()
getOrder_id():int
setOrder_id(int):void
getCustomer():Customer
setCustomer(Customer):void
getProduct():Product
setProduct(Product):void
getOrderState():String
setOrderState(String):void
getNumberOfProduct():String
setNumberOfProduct(String):void
-orders
0..*
-customer
0..1
-orders
0..*
-product
0..1
<<Java Class>>
Customer
com.model
customer_id: int
username: String
password: String
name: String
surname: String
floor: String
building: String
roomNumber: String
Customer()
Customer(String,String,String,String,String,String,String)
getCustomer_id():int
setCustomer_id(int):void
getUsername():String
setUsername(String):void
getPassword():String
setPassword(String):void
getName():String
setName(String):void
getSurname():String
setSurname(String):void
getFloor():String
setFloor(String):void
getBuilding():String
setBuilding(String):void
getRoomNumber():String
setRoomNumber(String):void
getOrders():Set<Order>
setOrders(Set<Order>):void
<<Java Class>>
Product
com.model
product_id: int
productName: String
price: String
Product()
getProduct_id():int
setProduct_id(int):void
Product(String,String)
getProductName():String
setProductName(String):void
getPrice():String
setPrice(String):void
getOrders():Set<Order>
setOrders(Set<Order>):void

Controller için class diagramları:



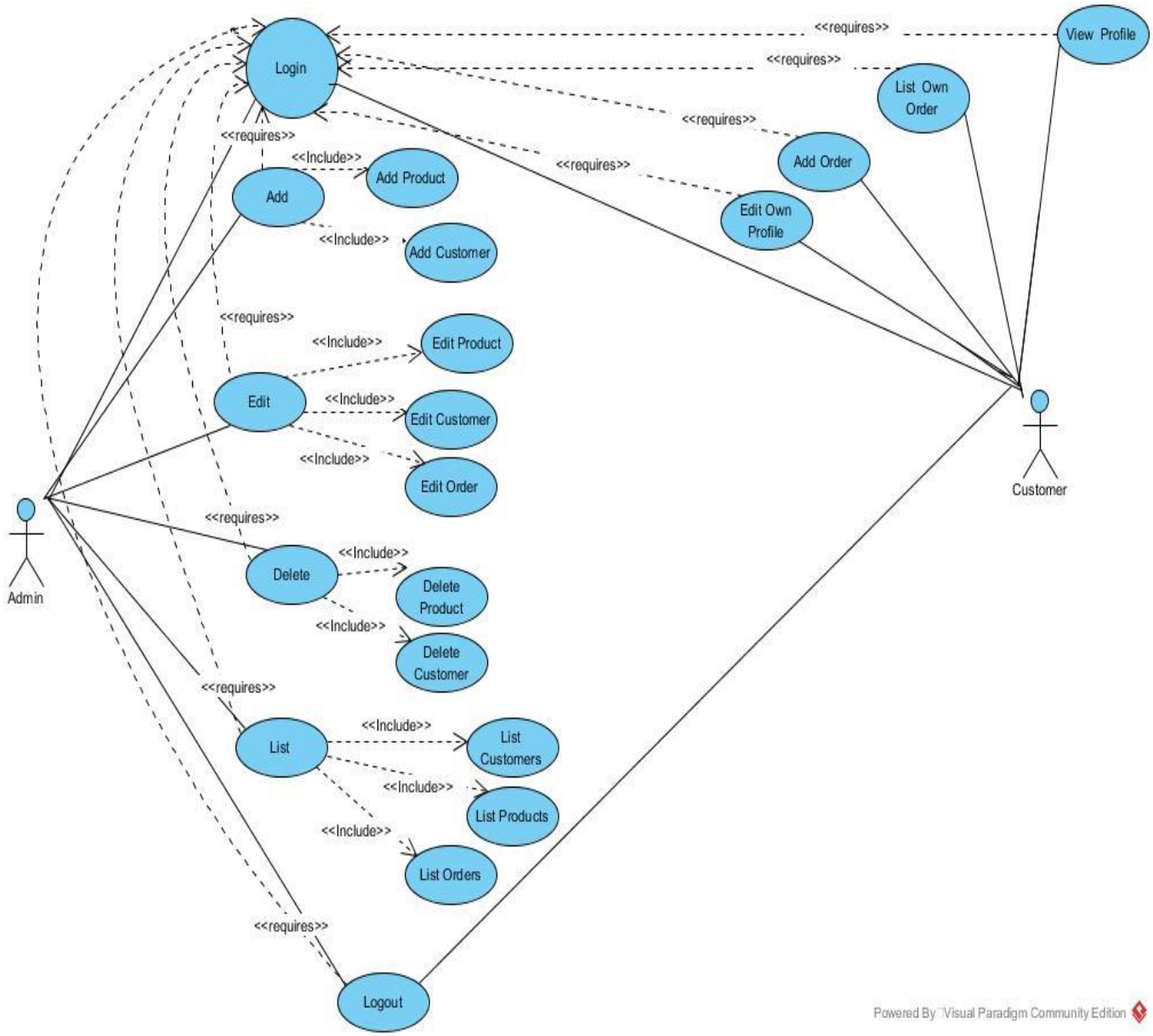
Bu controller sınıfları proje açısından oldukça önemlidir.Çünkü jsp sayfalarından gelen requestleri/istekleri bu sınıfların methodları tarafından yakalanır.Daha sonrasında o requesti yakalayan method ilgili jsp sayfasına response olarak yönlendirecektir. Kısacası bu sınıflar jsp sayfalarıyla iletişim kurarlar.Bu projede order,product ,customer,login ve logout nesneleri için ayrı ayrı controller sınıfları tanımlanmıştır.Bu sayede her bir nesne ile ilgili olan jsp sayfalarıyla ilişki kurulacaktır. Bu sınıflar daha çok sayfa yönlendirilmesi işlemlerinde kullanılmıştır.Özellikle de listelemenin olduğu sayfalar için kullanılmıştır.

Restful Web Servisi için class diagramı:



Bu sınıflar restful servis mantığında çalışır. **Spring MVC** içerisinde hazır gelen **@RestController** notasyonu sayesinde **Angular JS** kodlarıyla bağlantı kurulmuştur.Bu sınıfların en önemli özeliği; **Angular JS** ile **Json/Xml** gibi özel formatta verilerin gönderilmesini sağlamaktır.Bu sınıflar sayfa yönderilmesinde etkili değildir.Sayfa yönlendirilmeleri daha çok controller sınıfları içinde yapılır.Bu sınıflar sadece **@Controller** notasyonu içerir.Bu restful tabanlı sınıflar ekleme,silme ve güncelleme işlemlerinin yapıldığı sayfalarla iletişim halindedir.

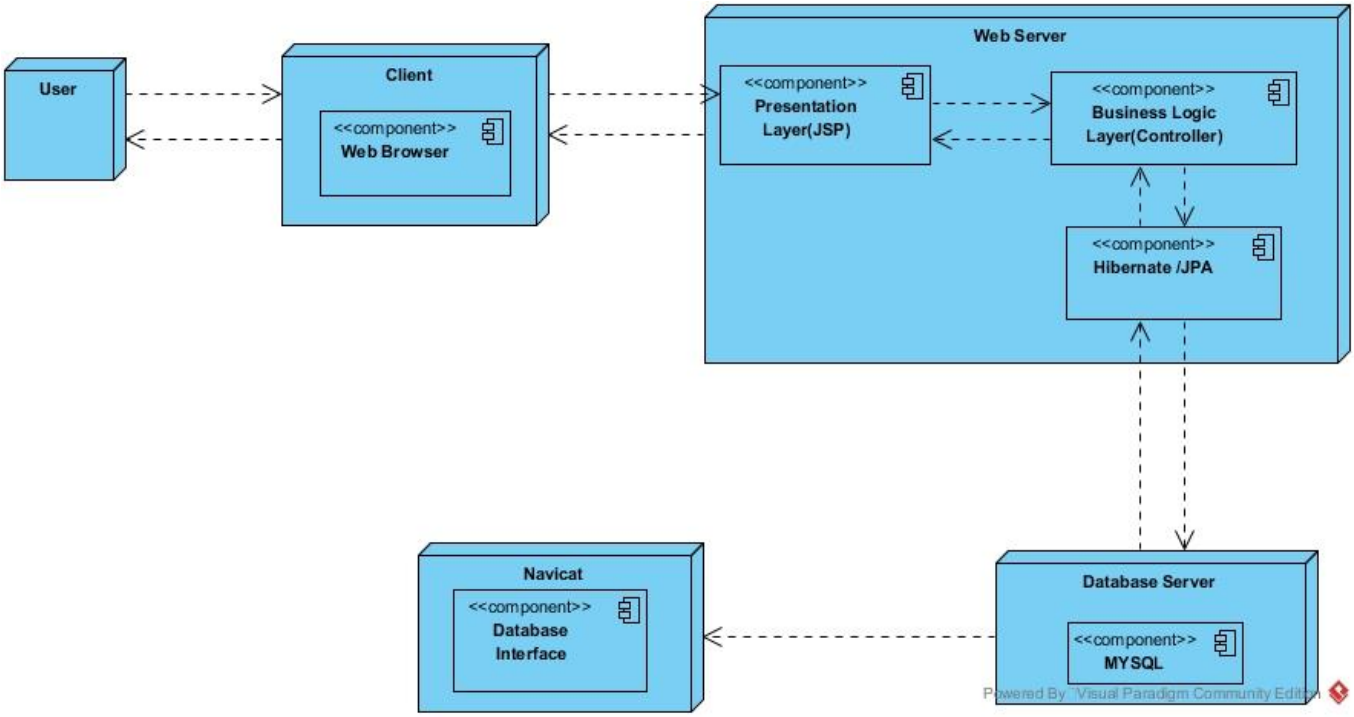
2. Sistemin her bir kullanıcı işlevini içeren kullanım senaryo çizeneği (use case diagram)



Sistemde iki adet aktor bulunmaktadır; admin ve customer.Admin ürün,müşteri ekleyebiliyor ve bunları da güncelleyebiliyor.Ayrıca siparişlerin durmunlarında da güncelleme yapabiliyor.Listeleme işlemi yapabiliyor; tüm müşterileri,tüm ürünleri ve verilen siparişleri listelebiliyor.Fakat bu tüm işleri

gerçekleştirebilmesi için login/giriş işlemi yapması gerekiyor.Diğer aktor ise müşteridir.Müşterinin yapacağı işlemler kısıtlıdır.Kendi bilgilerini güncelleyebiliyor,sipariş verebiliyor ve verdiği siparişleri listeleyebiliyor.Bu işlemleri yapabilmesi için öncelikle login/giriş işlemi yapması gerekiyor.

3. Yayılma çizeneği (deployment diagram)



Kullanıcı sisteme giriş yapmak için bilgisayarında kurulu olan web tarayıcısını açması ve login işlemini yapması gerekmektedir. Bu işlem sebebiyle web sunucusuna bir istek gönderilir.Web sunucusu kendi sistemimde **Apache Tomcat**dir. Bu yazılım içerisinde farklı kaplar(container) bulunur.Bu kapların amacı ilgili yapıları okumak ve işlemektir. Sonra da kullanıcıya gönderilmek üzere bir cevap (response) oluşturmaktır.Bu iletişim genellikle HTTP protokolü üzerinden yürür.Ayrıca sistemde sadece JSP sayfaları çalışmıyor bu sayfaların kontrollerini sağlayan arka tarafta çalışan controller sınıfları bulunur.Bu sınıflar request'i yakalayıp yapılmak istenen işi gerçekleştirir.Ayrıca contoller sınıfları veritabanı bağlantısı sağlayan ve verileri veritabanına ekleyen Dao sınıflarıyla bağlantısı vardır.Bu sınıflar ise Hibernate ile bağlantıları vardır.Bu bağlantılar sayesinde veritabanına bağlanıp veriler üzerinde işlem yapılır.Ayrıca kullanıcı veritabanı üzerinde yaptığı işlemleri daha rahat görebilsin diye Navicat kullanılmıştır.Bu sayede oluşan tablolar ve içerikleri rahat bir şekilde görülebilecektir.

4.Kaynakça

http://www.java2blog.com/2016/08/spring-mvc-angularjs-example.html

http://www.devglan.com/spring-mvc/spring-mvc-angularjs-integration-example

https://www.javainterviewpoint.com/angularjs-spring-mvc-integration/