

**İSKENDERUN TEKNİK ÜNİVERSİTESİ**

**İŞLETMEDE MESLEKİ EĞİTİM (İME)**

**GENEL KOORDİNATÖRLÜĞÜ**

**İŞLETMEDE MESLEKİ EĞİTİM (İME)**

**SONUÇ RAPORU**

**[ELİF GENCO]**

**[BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ]**

**10/05/2024**

**İÇİNDEKİLER**

**1.Firma Hakkında Bilgiler 1**

**2.Giriş 3**

**3.Ana Bölümler 4**

**3.1. Firmanın Faaliyet Alanları 4**

**3.2. Firmanın Entegratörünün Alanıyla ilgili Uygulamaları 6**

**3.2.1 Çalışma Ortamı 6**

**3.2.2 Proje Yönetimi 8**

**3.2.3 Yazılım Geliştirme 8**

**3.3. Firmanın Entegratörün Yetilerine/MesIeki Becerilerine Olan Katkıları 9**

**3.4. Entegratörün, Firmanın Uygulamalarına Olan Katkıları9**

**3.4.1 SOLID Prensipleri 9**

**3.4.2 Spring Boot Projesi 16**

**3.4.3 Postman Kullanımı 22**

**3.4.4 Junit Test 24**

**3.4.5 Thymeleaf Kullanımı 26**

**3.4.6 Linux Komutları 29**

**3.4.7 Linux Task 33**

**3.4.8 Docker 38**

**3.4.9 Dockerize İşlemi 47**

**3.4.10 Database Design 49**

**4.Tartışma ve Sonuç 57**

1. **Firma Hakkında Bilgiler**

**Firmanın Adı :** Huawei Telekomünikasyon Dış Ticaret Ltd. Şti**.**

**Firmanın Tarihçesi :**

Huawei Technologies Co. Ltd. veya kısa adıyla Huawei, merkezi [Çin](https://tr.wikipedia.org/wiki/%C3%87in)'in [Shenzen](https://tr.wikipedia.org/wiki/Shenzhen) şehrinde bulunan, [Çin](https://tr.wikipedia.org/wiki/%C3%87in) menşeli ve [çok uluslu bir şirkettir.](https://tr.wikipedia.org/wiki/%C3%87ok_uluslu_%C5%9Firket)

Huawei, 2012 yılı itibarıyla, [Ericsson](https://tr.wikipedia.org/wiki/Ericsson)'u geçerek dünyanın en büyük ağ ve [telekomünikasyon ekipmanı](https://tr.wikipedia.org/w/index.php?title=Telekom%C3%BCnikasyon_ekipman%C4%B1&action=edit&redlink=1) üreticisi konumuna gelmiştir.

Huawei, 1987 yılında, eski bir subay olan [Ren Zhengfei](https://tr.wikipedia.org/wiki/Ren_Zhengfei) tarafından kendisine ait özel bir şirket olarak kurulmuştur.

Huawei'nin temel amacı kurulduğu günden beri [geniş bant ağı](https://tr.wikipedia.org/w/index.php?title=Geni%C5%9F_bant_a%C4%9F%C4%B1&action=edit&redlink=1), [hücresel ağ](https://tr.wikipedia.org/wiki/H%C3%BCcresel_a%C4%9F), [ağ donanımı](https://tr.wikipedia.org/wiki/A%C4%9F_donan%C4%B1m%C4%B1), [yarı iletken](https://tr.wikipedia.org/wiki/Yar%C4%B1_iletken), ağ ve [telekomünikasyon](https://tr.wikipedia.org/wiki/Telekom%C3%BCnikasyon) cihazları üretmek ve bu cihazları [Çin](https://tr.wikipedia.org/wiki/%C3%87in) ve diğer yabancı ülkelere pazarlayıp teknik destek hizmeti sağlamak olmuştur. Şirketin 2015 yılı verilerine göre dünya üzerinde 170.000'den fazla çalışanı vardır ve bunların 76.000'i [AR-GE](https://tr.wikipedia.org/wiki/Ara%C5%9Ft%C4%B1rma_ve_geli%C5%9Ftirme) (Araştırma & Geliştirme) alanında çalışmaktadır.

Huawei, [Çin](https://tr.wikipedia.org/wiki/%C3%87in), [ABD](https://tr.wikipedia.org/wiki/Amerika_Birle%C5%9Fik_Devletleri), [Kanada](https://tr.wikipedia.org/wiki/Kanada), [Birleşik Krallık](https://tr.wikipedia.org/wiki/Birle%C5%9Fik_Krall%C4%B1k), [Pakistan](https://tr.wikipedia.org/wiki/Pakistan), [Fransa](https://tr.wikipedia.org/wiki/Fransa), [Belçika](https://tr.wikipedia.org/wiki/Bel%C3%A7ika), [Almanya](https://tr.wikipedia.org/wiki/Almanya), [Kolombiya](https://tr.wikipedia.org/wiki/Kolombiya), [İsveç](https://tr.wikipedia.org/wiki/%C4%B0sve%C3%A7), [İrlanda](https://tr.wikipedia.org/wiki/%C4%B0rlanda), [Hindistan](https://tr.wikipedia.org/wiki/Hindistan), [Rusya](https://tr.wikipedia.org/wiki/Rusya) ve [Türkiye](https://tr.wikipedia.org/wiki/T%C3%BCrkiye)'de olmak üzere toplam 21 tane AR-GE enstitüsüne sahiptir.

Şirketin ikinci büyük AR-GE merkezi [Türkiye](https://tr.wikipedia.org/wiki/T%C3%BCrkiye)'de bulunmaktadır ve [ABD](https://tr.wikipedia.org/wiki/Amerika_Birle%C5%9Fik_Devletleri) ile yaşanan gerilimin ardından şirket, [Türkiye](https://tr.wikipedia.org/wiki/T%C3%BCrkiye)'yi merkez üs olarak konumlandırmıştır. Şirket, 2013 yılında AR-GE alanındaki 5 milyar $ kazancını 2014'te 6.4 milyar $'a yükseltmiştir. Firmanın 2014'teki karı 34.2 milyar [CNY](https://tr.wikipedia.org/wiki/Renminbi_(Yuan)" \o "Renminbi (Yuan))'dir (15.4 milyar ₺).

Halen 140'tan fazla ülkeye hizmet verip dünyanın en büyük 50 telekomünikasyon operatörünün 45'inin sahibidir. [Google](https://tr.wikipedia.org/wiki/Google), 20 Mayıs'ta [Android](https://tr.wikipedia.org/wiki/Android_(i%C5%9Fletim_sistemi)) işletim sistemi güncellemelerini Huawei telefonlar için sınırladığını duyurdu.

[ABD](https://tr.wikipedia.org/wiki/Amerika_Birle%C5%9Fik_Devletleri) ile [Çin](https://tr.wikipedia.org/wiki/%C3%87in) arasındaki ticaret savaşının bir sonucu olarak görülen bu gelişmeden sonra Huawei, yeni işletim sistemi için [HarmonyOS](https://tr.wikipedia.org/wiki/HarmonyOS) ve [Huawei EMUI](https://tr.wikipedia.org/wiki/Huawei_EMUI) sistemini, [Huawei AppGallery](https://tr.wikipedia.org/wiki/Huawei_AppGallery), [Petal Search](https://tr.wikipedia.org/wiki/Petal_Search), [Petal Haritalar](https://tr.wikipedia.org/wiki/Petal_Haritalar) ve [Huawei Mobil Servisleri](https://tr.wikipedia.org/w/index.php?title=Huawei_Mobil_Servisleri&action=edit&redlink=1)'ni geliştirmeye başladı. Şirket, [Google](https://tr.wikipedia.org/wiki/Google) ile arasında yaşanan anlaşmazlığı son kullanıcıya anlatmak için [huaweimatters.com](https://huaweimatters.com/) sitesini kurdu. Şirket, 2019 yılında duyurduğu[HarmonyOS](https://tr.wikipedia.org/wiki/HarmonyOS) işletim sistemini 2 Haziran 2021 tarihinde tüm dünyaya tanıtımını yaptı.

**Firmanın Adresi :** Saray mahallesi Ahmet Tevfik İleri caddesi Onur Ofis Park İş Merkezi Sit. A1 Blok No:10 B/1 Ümraniye/İstanbul

**Firma Faaliyet Alanı :** Telekomünikasyon

**Mühendis ve Teknik Personel Sayısı :**

+900 Mühendis

+1200 Toplam Çalışan

**Organizasyon Şeması :**



**2. Giriş**

Bu rapor, Bilgi ve iletişim teknolojileri, akıllı cihazlar, bulut tabanlı çözümler ve BT alanında faaliyet gösteren " Huawei Telekomünikasyon Dış Ticaret Ltd. Şti." isimli şirkette "İşletmede Mesleki Eğitim (İME)” programı çerçevesinde gerçekleştirdiğim çalışmaların bir değerlendirmesini sunmaktadır. Çalışmaların konusu, amacı ve rapor içeriği aşağıda özetlenmiştir.

Eğitimin konusu, Huawei Telekomünikasyon Dış Ticaret Ltd. Şti. Şirketindeki telekomünikasyon projelerinde bilgisayar mühendisliği alanında görev almak ve şirketin teknik süreçlerine katkı sağlamaktı. Bu süre zarfında, şirketin teknik altyapısını inceledim, mevcut yazılım sistemlerini değerlendirdim ve ilgili ekiplerle birlikte projelerin yönetimi ve izlenmesi için geliştirme önerileri sundum.

Eğitimin amacı, teorik bilgilerimi pratik uygulamalarla birleştirmek, telekomünikasyon sektöründeki teknik becerilerimi geliştirmek ve iş hayatında edinilen deneyimi yaşamak oldu. Bu süreçte, gerçek projelerde aktif olarak yer aldım, takım çalışması ve iletişim becerilerimi güçlendirdim ve ofis bir ortamda çalışmanın dinamiklerini deneyimledim.

Bu rapor, İME sürecinde yaptığım gözlemler, katkılarım ve deneyimlerimle ilgili ayrıntılı bir değerlendirme sunacaktır. İncelemelerim ve analizlerim doğrultusunda, şirketin mevcut süreçlerinde iyileştirme önerileri sunacak, teknik becerilerim ve Bilgisayar Mühendisliği alanındaki deneyimlerim hakkında genel bir değerIendirme yapacağım.

Bu eğitim, Bilgisayar Mühendisliği öğrencisi olarak mezuniyet öncesi önemli bir deneyim sağlamıştır. Huawei Telekomünikasyon Dış Ticaret Ltd. Şti. şirketindeki çalışmalarım, iş hayatına geçiş aşamasında edinilen becerilerin pratiğe dökülmesi açısından büyük bir önem taşımaktadır. Bu rapor, İME sürecinde elde ettiğim değerli deneyimleri ve kazanımları detaylı bir şekilde sunmayı amaçlamaktadır.

**3. Ana Bölümler**

**3.1. Firmanın Faaliyet Alanları**

Huawei Telekomünikasyon Dış Ticaret Ltd. Şti. Bilgi ve iletişim teknolojileri, akıllı cihazlar, bulut tabanlı çözümler ve BT  alanında faaliyet gösteren bir şirkettir. Şirket, birçok önemli projeye imza atmış olup yurt içinde ve yurt dışında faaliyet göstermektedir. Huawei Telekomünikasyon Dış Ticaret Ltd. Şti.'nun faaliyet alanları şunlardır :

Telefon:Şirket, akıllı telefon üretimi ve gelişimi için başarıyla projeler gerçekleştirmektedir**.**



*Görsel 1*

Tablet:Şirket, tablet üretimi ve gelişimi için başarıyla projeler gerçekleştirmektedir.



*Görsel 2*

Akıllı Saat:Şirket, dijital saat üretimi ve gelişimi için başarıyla projeler gerçekleştirmektedir.



*Görsel 3*

Bilgisayar:Şirket, bigisayar üretimi ve gelişimi için başarıyla projeler gerçekleştirmektedir.



*Görsel 4*

**3.2. Firmanın Entegratörün Alanıyla İlgili Uygulamaları**

***3.2.1. Çalışma Ortamı***



*Görsel 5*



*Görsel 6*



*Görsel 7*

Huawei şirketinde çalışan entegratör, Bilgisayar Mühendisliği alanında profesyonel bir çalışma ortamında yer almaktadır. Firma, entegratörün verimliliğini ve başarısını desteklemek için modem ve teknolojik olarak donatılmış bir ofis sağlamaktadır.

Ofis, Bilgisayar Mühendisliği çalışmaları için gerekli olan donanım ve yazılımlarla donatılmıştır. Entegratör, yüksek performanslı bilgisayarlar, yazılım geliştirme araçları, veri analizi gibi en son teknolojiye sahip ekipmanlardan yararlanmaktadır. Bu, entegratörün projelerde etkili bir şekilde çalışmasını sağlamakta ve Bilgisayar Mühendisliği becerilerini geliştirmesine yardımcı olmaktadır.

Huawei şirketi, entegratörün çalışma ortamında işbirliğini teşvik etmektedir. Entegratör, ekip çalışmasının önemini kavramak ve proje süreçlerinde diğer ekip üyeleriyle etkili iletişim kurmak için uygun bir ortamda bulunmaktadır. Ortak çalışma alanları, toplantı odaları ve iletişim araçları gibi kaynaklar, entegratörün fikir alışverişinde bulunması, problem çözme becerilerini geliştirmesi ve projelerde ekip çalışmasını yönetmesi için elverişlidir.

.

***3.2.2. Proje Yönetimi***

Huawei şirketi, gerçekleştirilmesi gereken bir proje olduğunda projeyi gerçekleştirmek için çalışanlar arasında bu proje için en uygun olan çalışanlar ile takım kurulur, gerekli olan araçlar sağlanır. İhtiyaç duyulan kaynaklar iletim sağlanarak vurgulanır. Bu şekilde çalışanların daha güçlü iletişim kurması sağlanır ve çalışanların projeye olan bağlılığı arttırılmış olur. Proje başarılı bir şekilde sonlandırılır. Bu süreçte yer alan entegratör proje yönetimine hakimiyet sağlar ve deneyim kazanmış olur.

***3.2.3. Yazılım Geliştirme***

Huawei şirketi, birçok alanda yazılım geliştirme faaliyetinde bulunur. Yeni nesil üretime sahip olan “telefon,tablet,akıllı saat,kulaklık,bilgisayar” cihazlarının üretiminde ve gelişiminde yer alır. Ve kullanıcıları için her zaman fayda sağlayıp,işini kolaylaştıracak ürün geliştirir.

Bu ürünlerin gelişimi için gerçeleştirilen projelerde entegratörün yer alması, gerçek ve kapsamlı bir yazılım geliştirme projesinde yer aldığı için yazılım geliştirme faaliyetinin nasıl ilerlediğini, nasıl kapsamlı bir proje oluşturulduğunu,hangi yazılım alanlarının kullanıldığını, hangi teknolojilerin kullanıldığını ve en uygun olan ortamın nasıl oluşturulduğu hakkında bilgi sahibi olur. Bu edinilen bilgiler sayesinde entegratör iş hayatına adım atmış olur ve gelecek iş hayatına fayda sağayacak tecrübeler edinir.

**3.3. Firmanın Entegratörün Yetilerine/MesIeki Becerilerine Olan Katkıları**

Huawei şirketi, telekomünikasyon alanında hizmet sunmaktadır. Bilgisayar Mühendisliği öğrencisi olan entegratöre iş hayatı, proje sürecinin işleyişi, çalışma ortamı, yönetici – çalışan ilişkisi, ekip çalışması, iş yerindeki gizlilik koşulları hakkında önemli bilgiler sunmaktadır.

Entegratörün takım arkadaşları ile birikte bir proje de yer alması, ekip çalışmasına uyum sağlamasını ve takımca çalışmayı tecrübe etmesini sağlar.

Huawei şirketi yapmış olduğu katkılarla entegratörün Bilgisayar Mühendisliği alanındaki yeteneklerini geliştirmesini sağlar.

Entegratörün, İME eğitimi sonunda başlayacak olan kariyer hayatına sağlam adımlar atarak ilerlemesini hedefler. Böylece geleceğe başarılı ve donanımlı mühendisler yetiştirmiş olur.

**3.4. Entegratörün Firmanın Uygulamalarına Olan Katkıları**

Huawei şirketi, Bilgisayar Mühendisliği öğrencisi olan entegratörden önemli katkılar elde etmiştir. Entegratör, SOLİD Prensiplerini dikkate alarak, Spring Boot çalışmalarını yaparak, Postman kullanarak, H2 database kullanarak, Entity kullanımı sağlayarak geliştirmiş olduğu Spring Boot projesi ile gelişimekte olan projeye katkı sağlamıştır.

***3.4.1. SOLİD Prensipleri***

SOLİD Prensipleri Çalışmaları şu şekildedir:

1. Single Responsibility Principle

2. Open / Closed Principle

3. Liskov's Substitution Principle

4. İnterface Segregation Principle

5. Dependency Inversion / Injection Principle

Bu çalışmalar ve araştırmalar yapıldıktan sonra genel bir bilgi sahibi olundu. Sonrasında daha ayrıntılı bir çalışma yapmak için hazırlık yapıldı .

İlk olarak Single Responsibility Principle' i araştırıldı ve örnek çalışma yapıldı.

**Single Responsibility Principle :**

- "Tek Sorumluluk İlkesi" anlamına gelir ve SOLID prensiplerinden biridir.

-Tek Sorumluluk İlkesi, bir sınıfın veya modülün sadece bir tek sorumluluğu

olması gerektiğini savunan bir ilkedir.

- Bir component sadece bir amaçla değiştirilmelidir.

- Eğer birden fazla amaca hizmet ediyorsa bu prensibi ihlal ediyor demektir.

- Kodun küçük ve bir amaca hizmet eden componentler şeklinde tasarlanması

yeniden kullanabilirliği arttırır.

- Componentlerin küçük olması kodların daha rahat anlaşılmasını ve yönetilmesini

sağlar.

- SRP'nin amacı, sınıfları ve modülleri tek bir amaç veya sorumluluk etrafında

gruplamak suretiyle kodun daha sade ve esnek hale gelmesini sağlamaktır.

Bir sınıfın çok sayıda sorumluluğu olduğunda, sınıfın değiştirilmesi veya bakımı

karmaşıklaşabilir ve bir sorumlulukta yapılan değişiklikler, diğer sorumlulukları

etkileyebilir.

Bu şekilde araştırma yapılarak ana mantık öğrenildi. SOLID Prensiplerindeki mantık

kod düzenliliğini,temizliğini,anlaşılabilirliğini sağlamaktır. Herhangi bir yapılacak düzenlemeyi kolaylaştırır.

Yapılan kod çalışması :

public class User {  
  
 private Long id;  
 private Address name;  
   
 //Getter,setter

public class Address {

private String street;

private String city;

private String country;

//Getter,setter

}

// Sorumlulukları her bir sınıfa etkili bir şekilde ayırdık ve böylece bir yerdeki kod değiştirme şansını ve koddaki herhangi bir yerin diğerini bozma şansını azalttık. Güncelleştirmeler gerektiğinde kolayca değişiklik yapabileceğimiz bir yapı kurmuş olduk.

public class AddressService{

public void changeAddress(Address address) {

// Sadece addressle ilgileniyorum ve ondan sorumluyum userdaki değişiklikler beni etkilemez.

}

}

Daha sonra SOLID Principle olan Open/Closed Principle konusuna bakıldı ve gerekli çalışmalar örnekler ile yapıldı.

**Open/Closed Principle :**

* SOLID Prensiplerinden biridir.
* Bir component değişime kapalı ve gelişime açık olmalıdır.
* Genelde backward compatibility sorunlarını çözmek için yararlı olur.
* Örneğin ; hesap sınıfınızı sonradan değiştirmek yerine ondan kalıtım alarak yeni hesap sınıfı oluşturmak ve içindeki kodu override etmek .
* *Yazılım varlıklarının genişletmeye açık, ancak değişiklik için kapalı olması gerektiğini*  belirtir; yani kaynak kodunu  değiştirmeden davranışının genişletilmesine izin verebilir .
* 1990'larda, açık-kapalı prensibi , uygulamaların değiştirilebildiği ve birden fazla uygulamanın oluşturulabildiği ve polimorfik olarak birbirinin yerine geçebildiği soyutlanmış arayüzlerin kullanımına atıfta bulunmak üzere popüler bir şekilde yeniden tanımlandı .
* Bu tanım soyut temel sınıflarından kalıtımı savunur. Arayüz spesifikasyonları miras yoluyla yeniden kullanılabilir ancak uygulamanın böyle olması gerekmez. Mevcut arayüz değişikliklere kapalıdır ve yeni uygulamaların en azından bu arayüzü uygulaması gerekir.
* Bu prensip; sürdürülebilir ve tekrar kullanılabilir yapıda kod yazmanın temelini oluşturur.

Yapılan kod çalışması:

public interface Shape {   
double getArea(); }

Şekillerden biri olan Dikdörtgen alan hesabını getArea metodumla öğrenebilirim.

public class Rectangle implements Shape {  
private double length;  
private double height;  
  
// getters/setters …   
@Override  
public double getArea() {  
return (length \* height);  
} }

public class Circle implements Shape { // circle içinde aynı işlemi gerçekleştiriyorum.  
private double radius;  
// getters/setters …   
@Override  
public double getArea() {  
return (radius \* radius \* Math.*PI*);  
} }

public class AreaManager {  
public double calculateArea(List<Shape> shapes) {  
double area = 0;  
for (Shape shape : shapes) {  
area += shape.getArea();  
}  
return area; }}

Artık programımız Open/Closed prensibine uygun hale gelmiştir. Herhangi bir yeni şekil alanı hesaplamamız gerektiğinde Shape nesnemizden yeni şekli türetmemiz ve alan hesabını kendi içinde yapmamızdır.

Bu çalışma da Liskov’s Substitution Principle çalışıldı.

**Liskov’s Substitution Principle:**

* SOLID Prensiplerinden biridir.
* Kodlarımızda herhangi bir değişiklik yapmaya gerek duymadan alt sınıfları, türedikleri(üst) sınıfların yerine kullanabilmeliyiz.
* Türeyen sınıf yani alt sınıflar ana(üst) sınıfın tüm özelliklerini ve metotlarını aynı işlevi gösterecek şekilde kullanabilme ve kendine ait yeni özellikler barındırabilmelidir.
* Alt seviye sınıflardan oluşan nesnelerin/sınıfların, ana(üst) sınıfın nesneleri ile yer değiştirdikleri zaman, aynı davranışı sergilemesi gerekmektedir. Türetilen sınıflar, türeyen sınıfların tüm özelliklerini kullanabilmelidir.
* Bu prensip, Open/Closed prensibine benzer ve open/closed prensibine uymak liskov prensibinin uygulanmasını kolaylaştırmaktadır.

Yapılan kod çalışması:

public interface Shape {

long area();

}

public class Square implements Shape {

private int size;

public Square(int size) {

this.size = size;

}

@Override

public long area() {

return size \* size;

}

public void setSize(int size) {

this.size = size;

}}

public class Rectangle implements Shape {

private int width;

private int height;

public Rectangle(int width, int height) {

this.width = width;

this.height = height;

}

@Override

public long area() {

return width \* height;

}

public void setWidth(int width) {

this.width = width;

}

public void setHeight(int height) {

this.height = height;

}

}

@Test

public void testRectangleArea() throws Exception {

Shape rectangle = new Rectangle(10, 5);

assertEquals(50, rectangle.area());

}

@Test

public void testSquareArea() throws Exception {

Shape square = new Square(5);

assertEquals(25, square.area());

}

Artık Kare ve Dikdörtgen kendi davranışlarına sahip oldu. Ve her biri ayrı şekil olarak kabul ediliyor. Böylece alan hesaplama her bir şekile özgü matematiksel bir işlem içerebiliyor.

LSP’ye uygunluk tam olarak sınıflardan beklenen davranışları karşılayabilecek bir hiyerarşi düzeni oluşturarak sınıf yapılarımızı geliştirmektir.

Daha sonra bir diğer prensip olan Interface Segregation Principle -Dependency İnversion/ İnjection Principle çalışıldı.

**Interface Segregation Principle -Dependency İnversion/ İnjection Principle :**

* Sorumlulukların hepsini tek bir arayüze toplamak yerine daha özelleştirilmiş birden fazla arayüz oluşturmayı tercih etmemizi söyleyen prensiptir.
* Başka sınıfları kullanan kodların ihtiyaçlarından fazlasına ulaşmasını engellemek ve bu sayede okunabilirliği ve bakım kolaylığını arttırmak için sunulan bir prensiptir.
* Tek bir interface yerine kullanımlarına göre parçalanmış birden fazla interface ile işlemleri yürütmeliyiz. Yani her farklı sorumluluğun kendine özgü bir arayüzü olması gerekmektedir. Böylece interface’i kullanan kişide sadece ihtiyacı olanlarla ilgilenmiş olur.
* Birden fazla amaç için yalnızca bir arayüzümüz var ise buna gerektiğinden fazla method ya da özellik ekliyoruz demektir, bu da IS prensibine aykırı davrandığınız anlamına gelir.
* Nesneler asla ihtiyacı olmayan property/metot vs içeren interfaceleri implement etmeye zorlanmamalıdır.
* Dependency Inversion/ İnjection Principle: Bağımlı olunan componentlerin oluşturulması değilde istenilmesi şeklinde çalışan bir prensiptir.

Yapılan kod çalışması:

public interface Flyable {

void fly();

}

public interface Runnable {

void run();

}

public interface Barkable {

void bark();

}

public class Bird implements Flyable, Runnable {

public void run() {

System.out.println("Kuş,Koşuyorum");

//logic

}

public void fly() {

System.out.println("Kuş, Uçuyorum.");

//logic

}

}

public class Cat implements Runnable {

public void run() {

System.out.println("Kedi,Koşuyorum");

//logi

}

}

public class Dog implements Runnable, Barkable {

public void bark() {

System.out.println("Köpek,Havlıyorum.");

//logic

}

public void run() {

System.out.println("Köpek,Koşuyorum.");

//logic

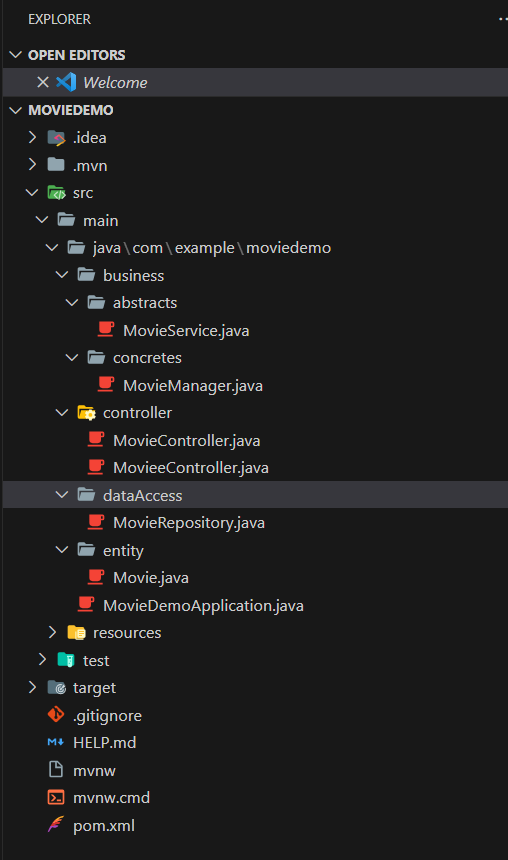
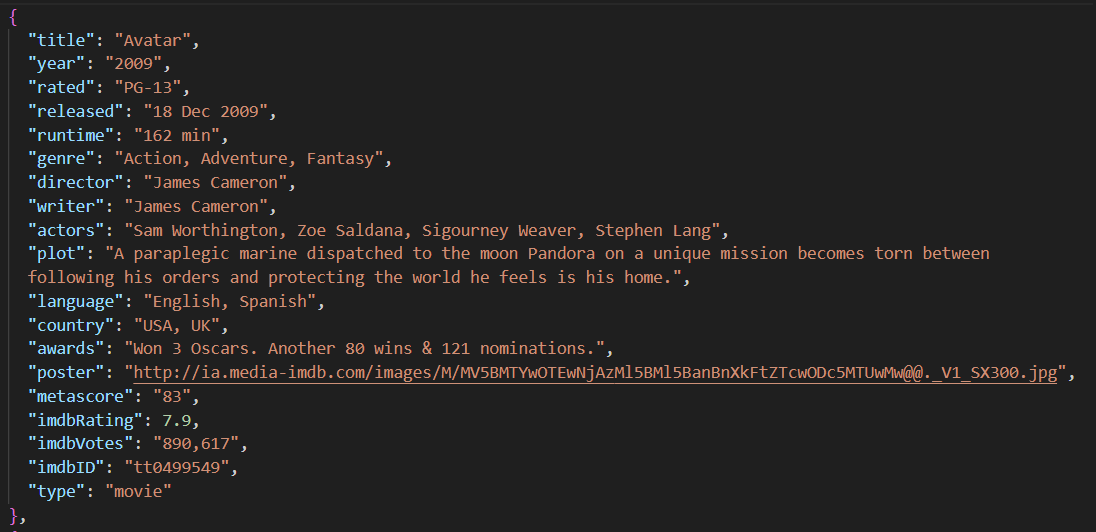
}

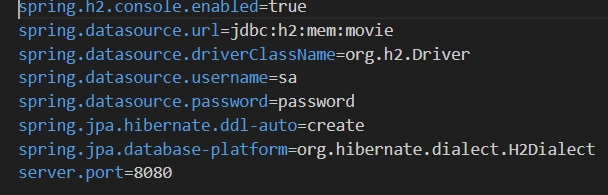
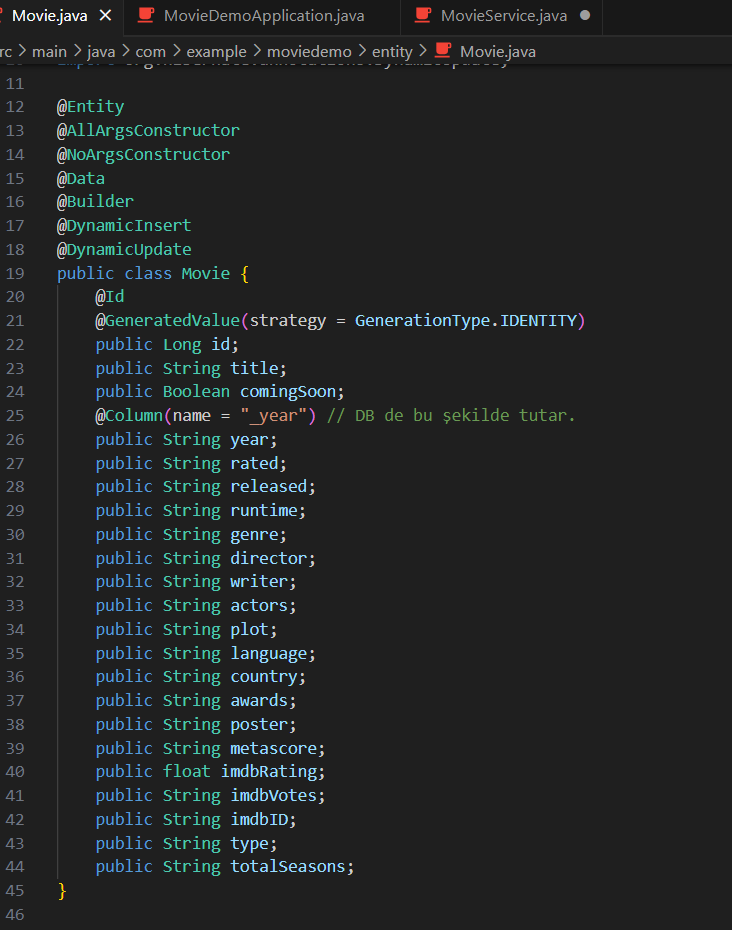
}

Böylece her yeni özellik için bir interface daha yazarak, özelliği barındıran sınıflara implement edebiliriz. Her sınıf, her hayvan kendi yapabildiği özelliği implement ederek, ihtiyaç duymadığı özelliklerden arındırılmış oldu. Kod okunurluğu artarken, esneklikte kazandırmış olduk. Yeni eklenecek olan herhangi bir class için, sadece ihtiyaç duydugu interfaceleri kullanabilmesini sağlamış olduk. SOLİD Prensiplerine dair olan çalışmalar sunuldu.

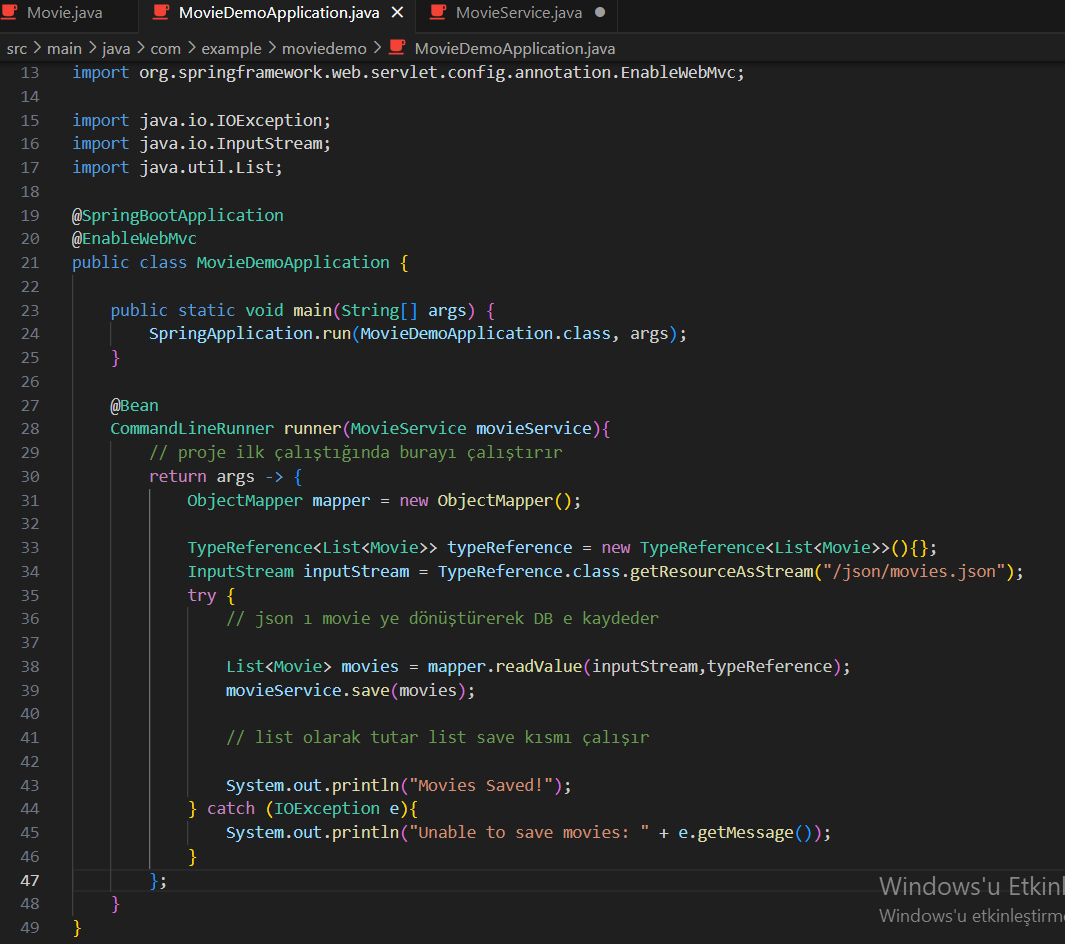
***3.4.2. Spring Boot Projesi***

Çalışmalar sonrasında projeye başlanıldı. Geliştirilen proje şu şekildedir:

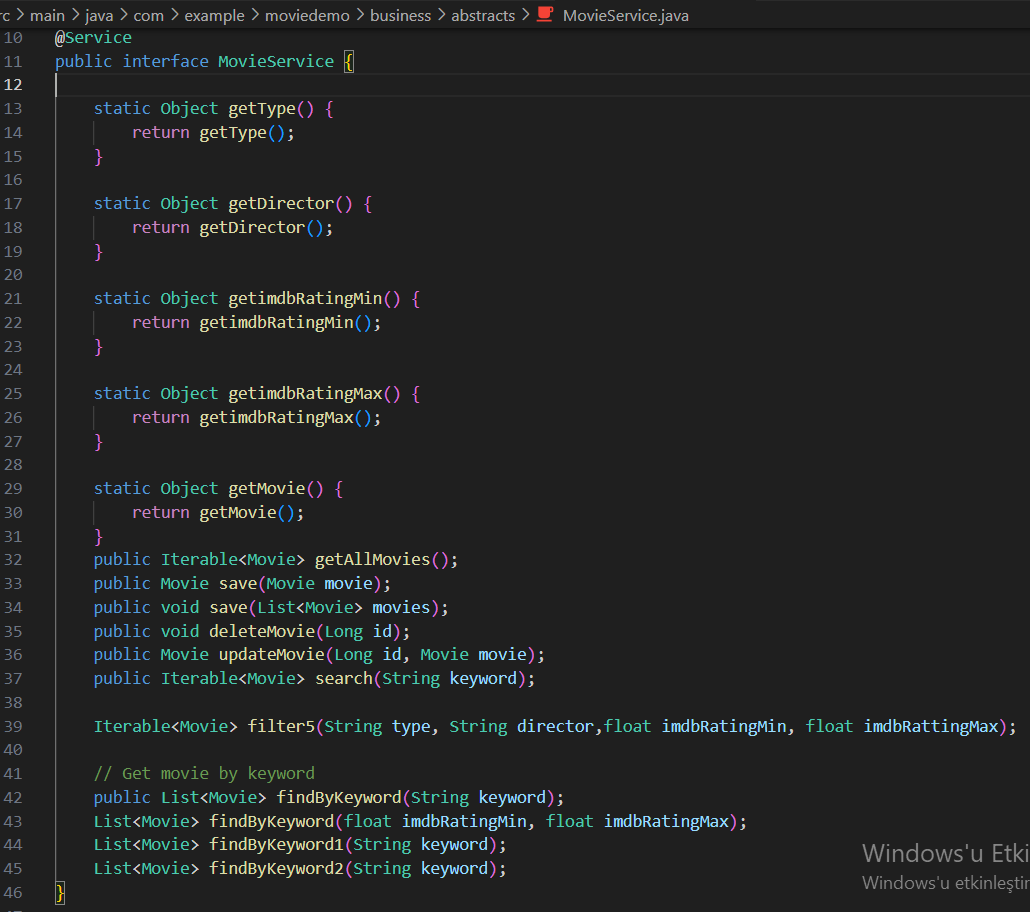
** ** *Görsel 8 Görsel 9*

****

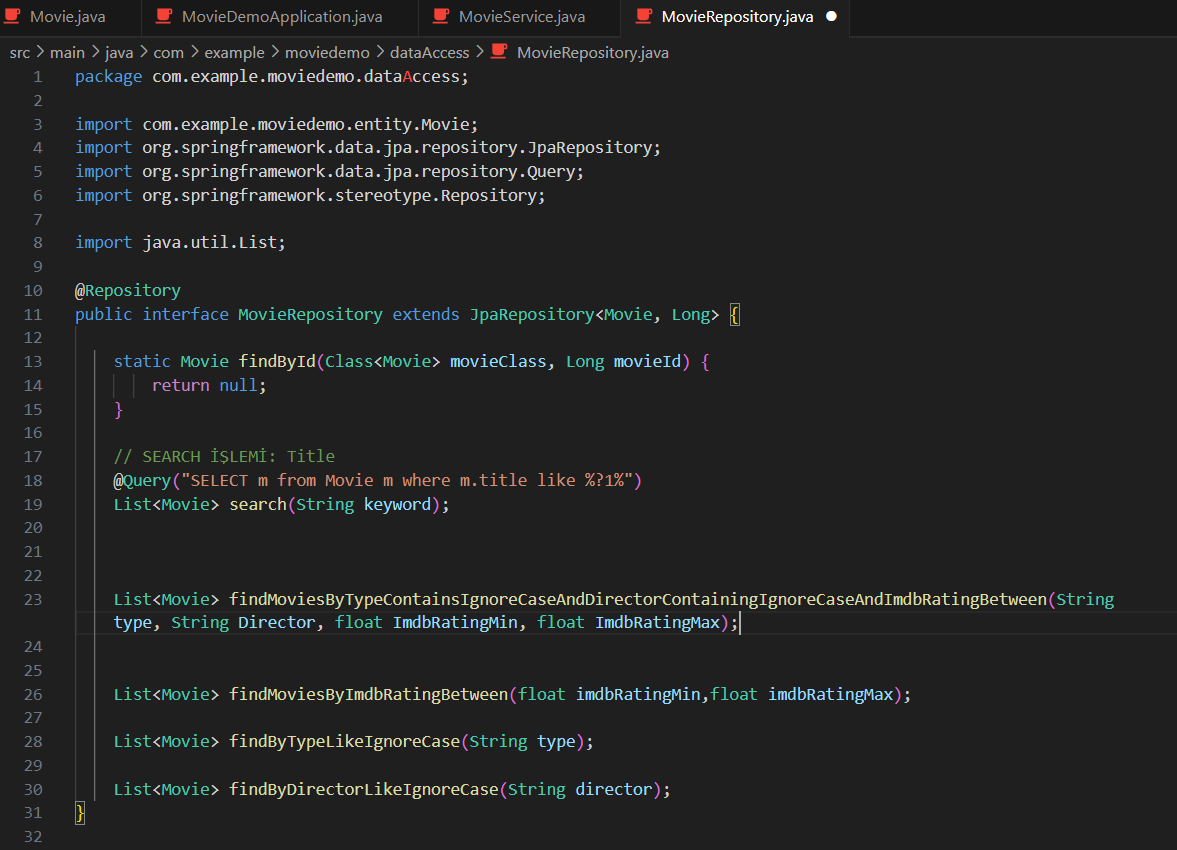
*Görsel 10 Görsel 11*

****

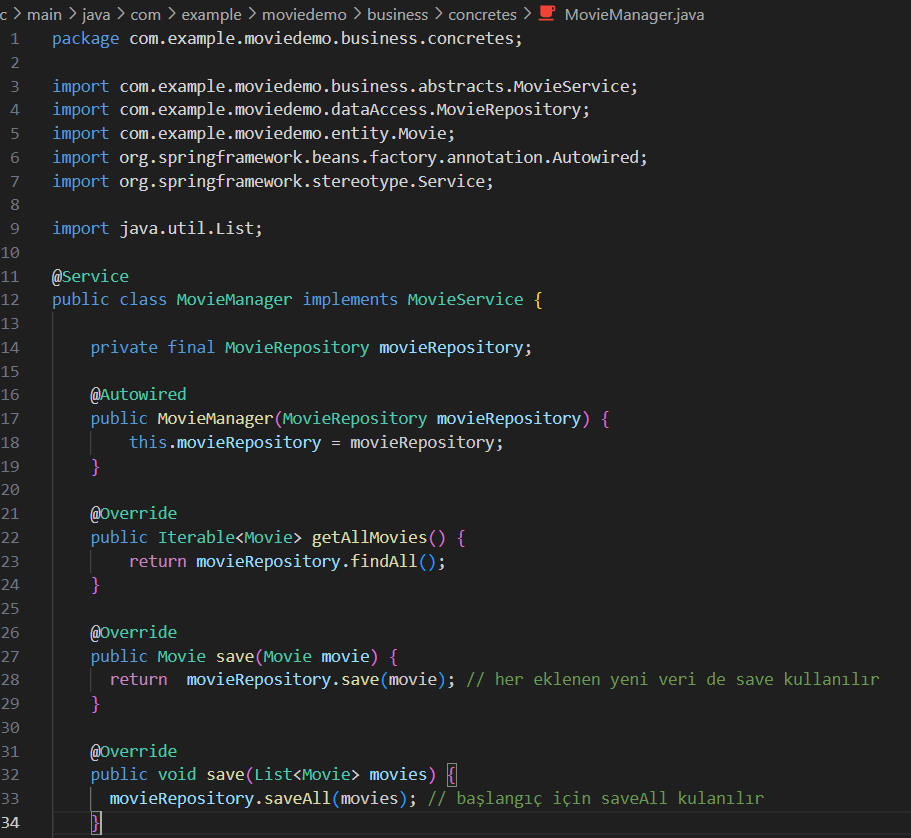
*Görsel 12*

****

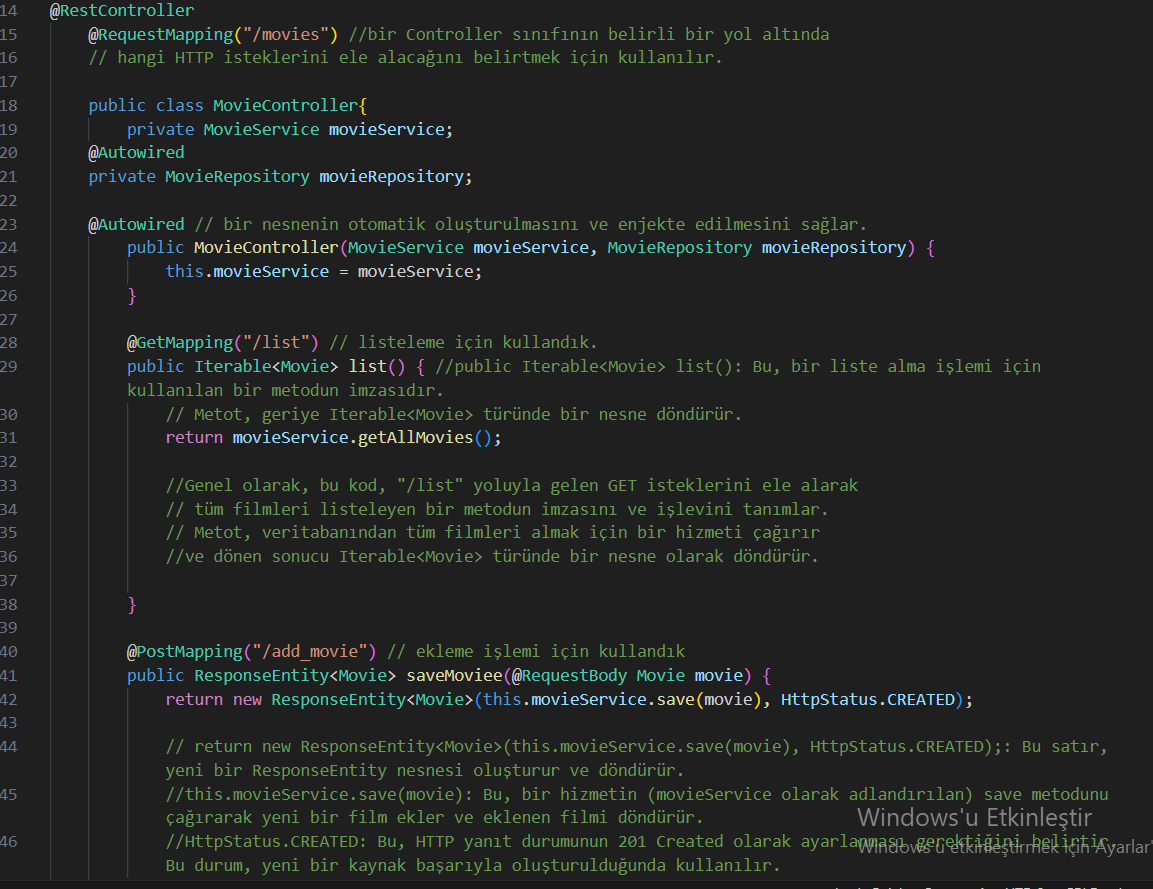
*Görsel 13*

****

*Görsel 14*

****

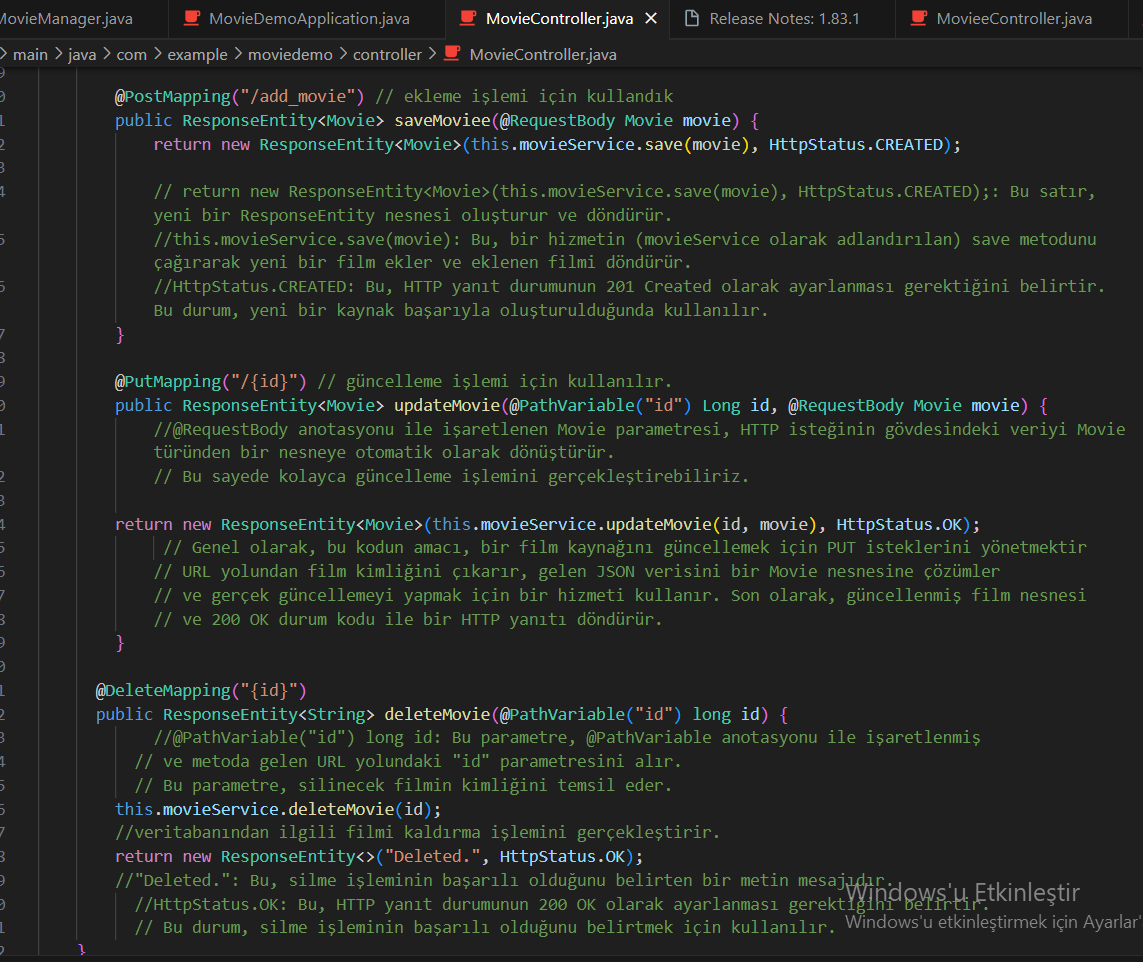
*Görsel 15*

****

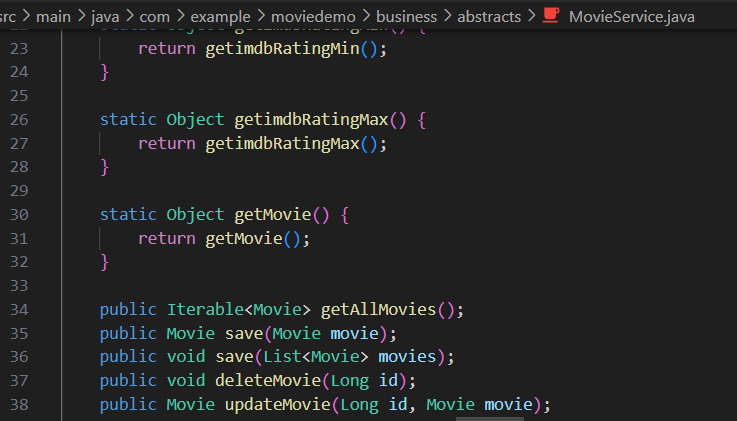
*Görsel 16*

****

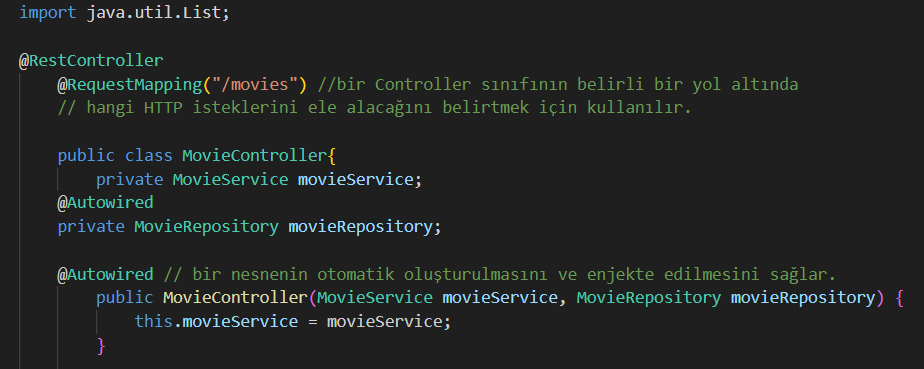
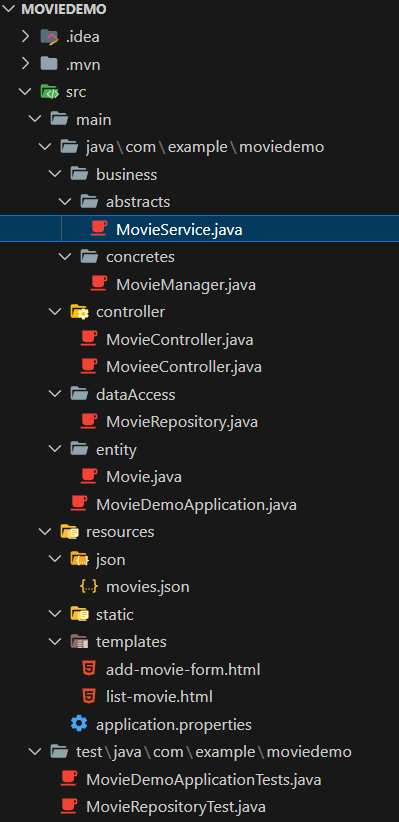
*Görsel 17*

****

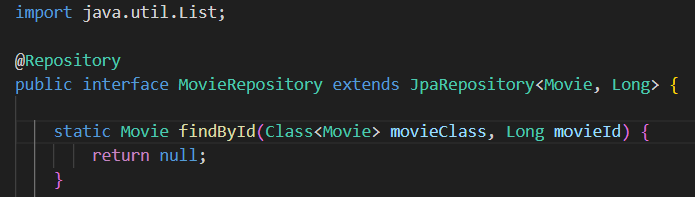
Görsel *18*

****

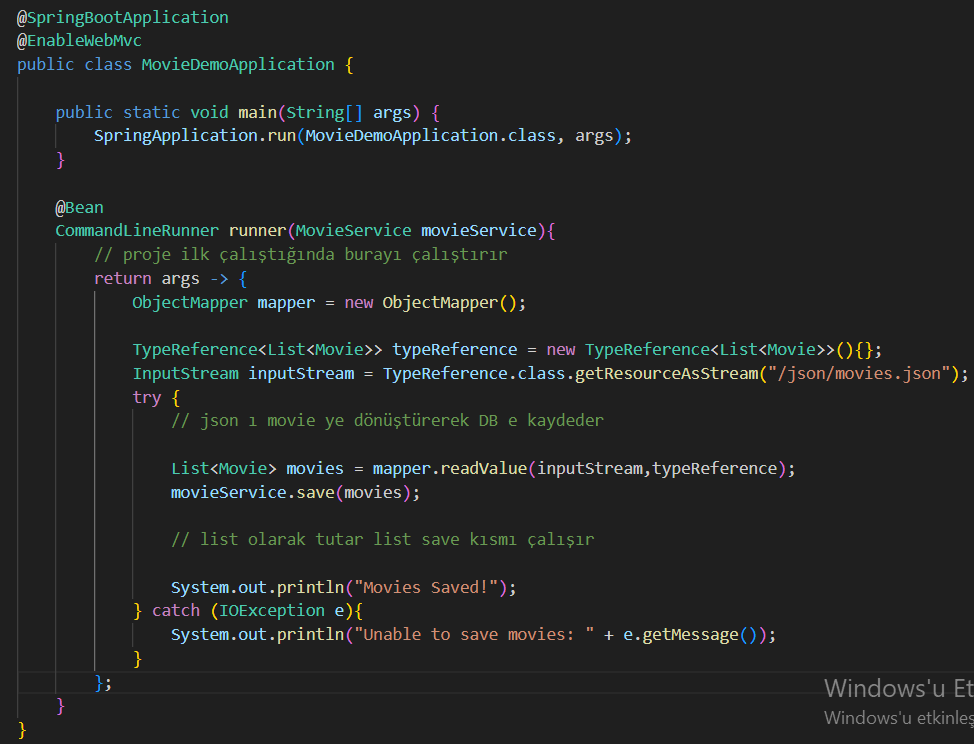
*Görsel 19*

** **

*Görsel 20 Görsel 21*

****

*Görsel 22*

****

*Görsel 23*

***3.4.3. Postman Kullanımı***

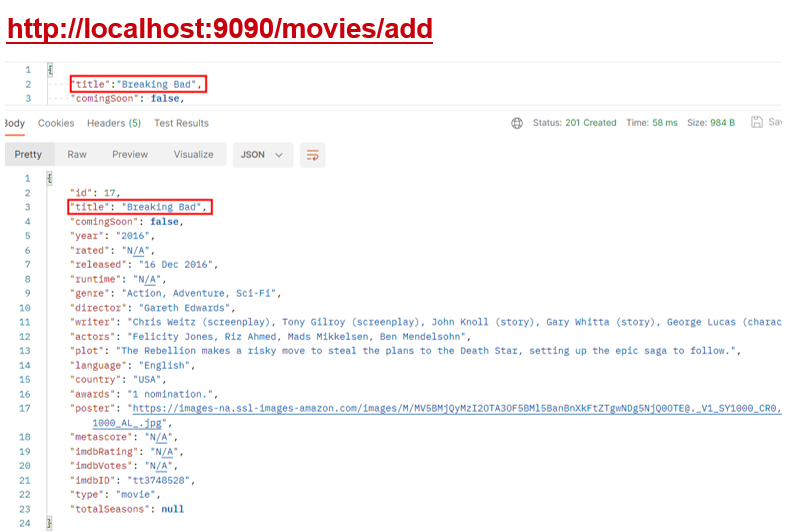
Proje Postman ile istek atılarak proje kontrol edildi. İşlemler şu şekildedir:

****

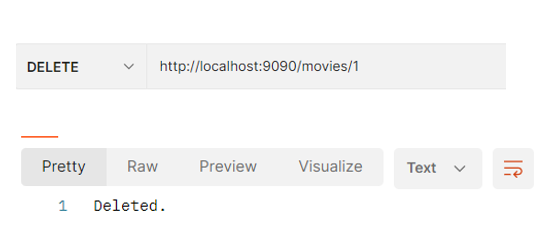
*Görsel 24 Görsel 25*

****

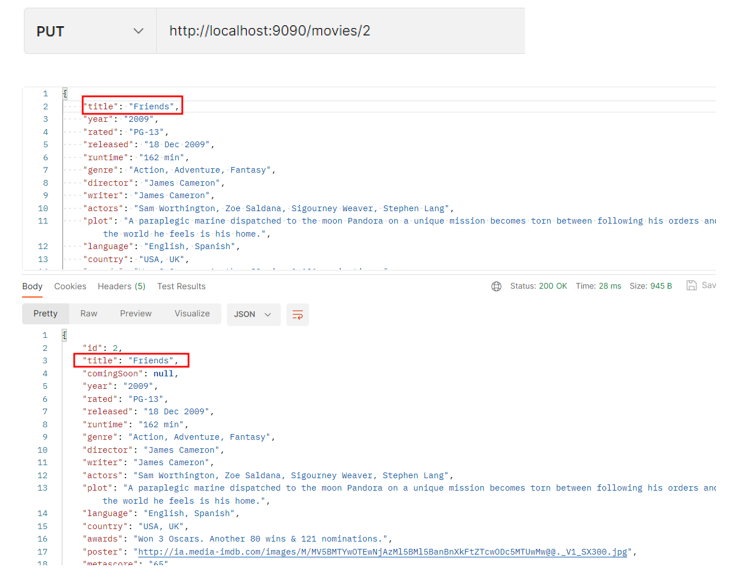
*Görsel 26*

****

*Görsel 27*

****

*Görsel 28*

****

*Görsel 29*

GET: Veri alınır.

POST: Veri gönderir.

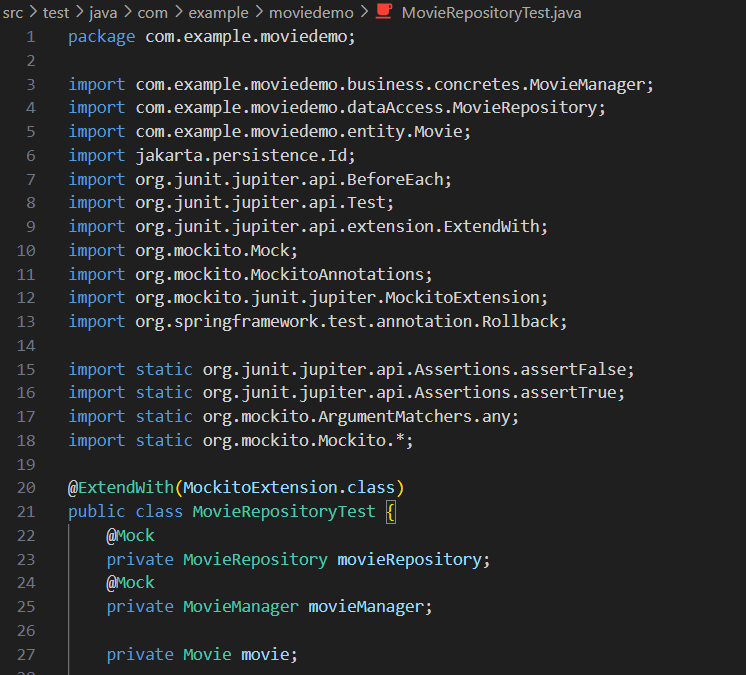
PUT: Güncelleme işlemi yapar.

PATCH: Güncelleme işlemi yapar.

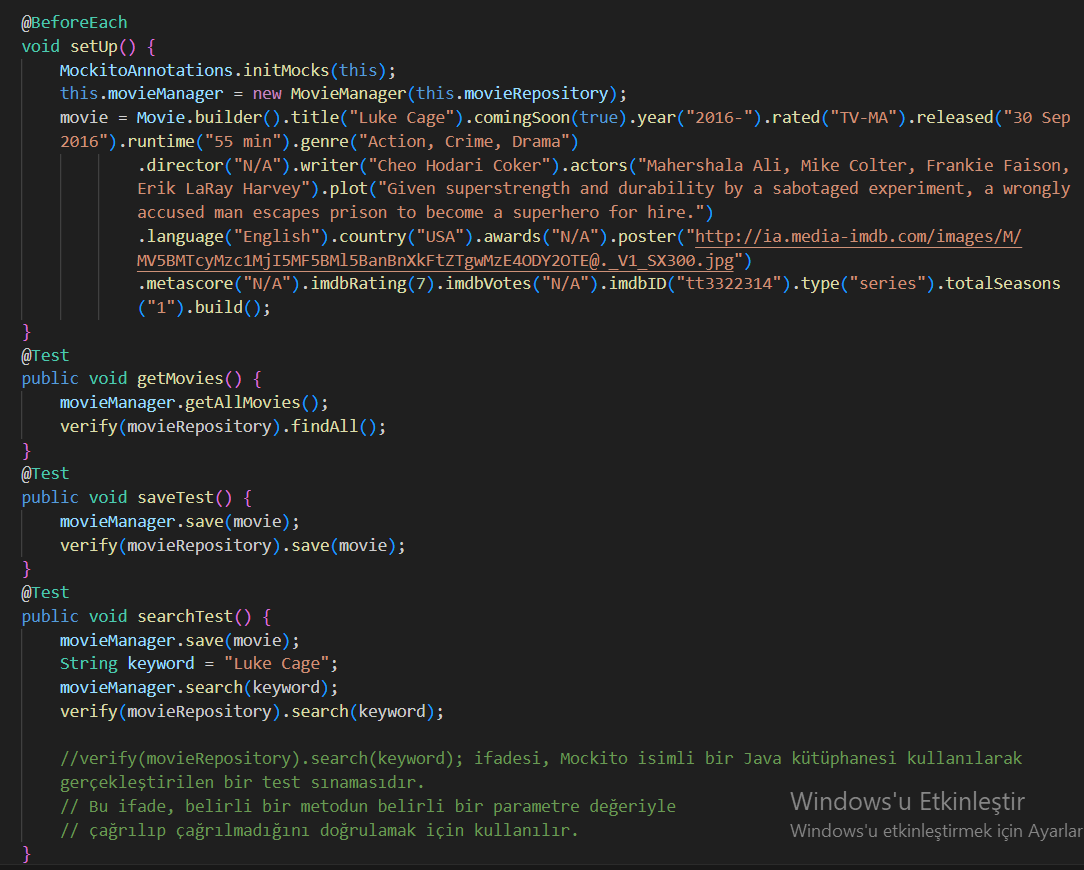
DELETE: Silme işlemi yapar.

***3.4.4. Junit Test***

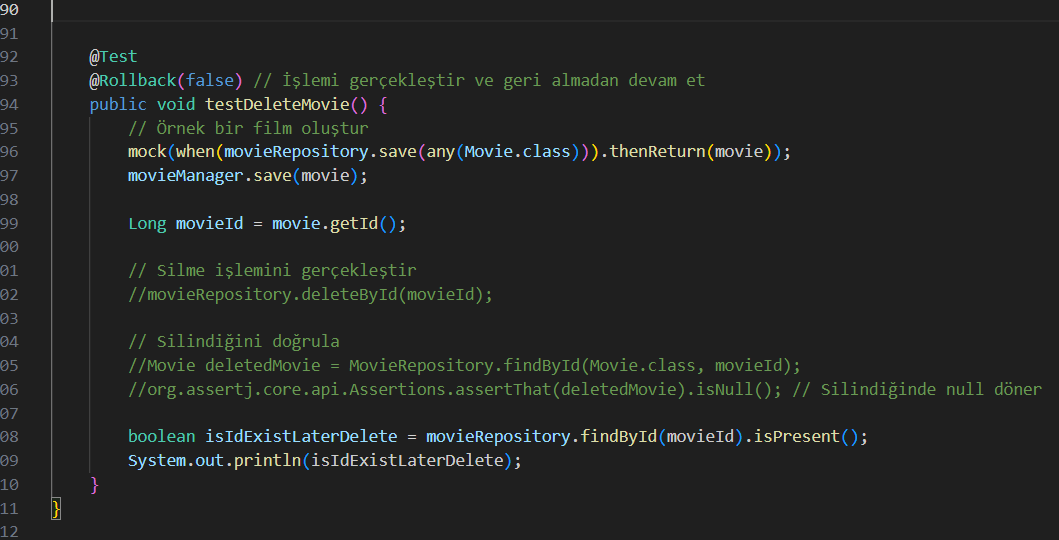
Projenin Junit Testi yazıldı. İşlemler şu şekildedir:

****

*Görsel 30*

********

*Görsel 31 Görsel 32*

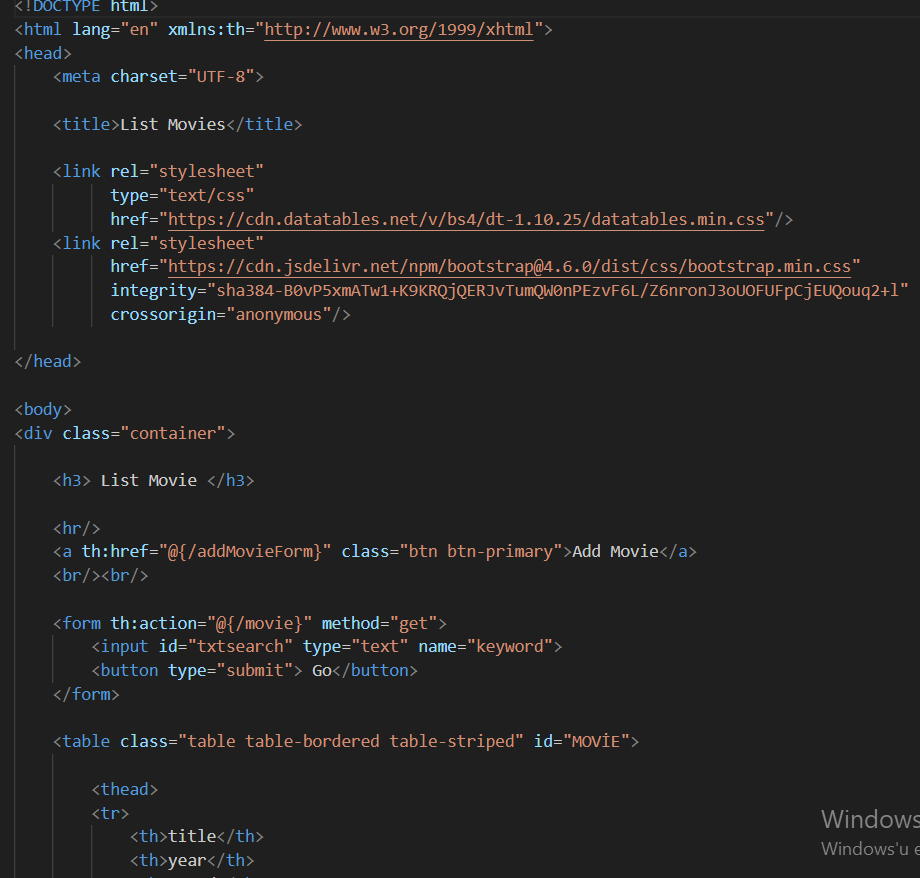
****

*Görsel 33*

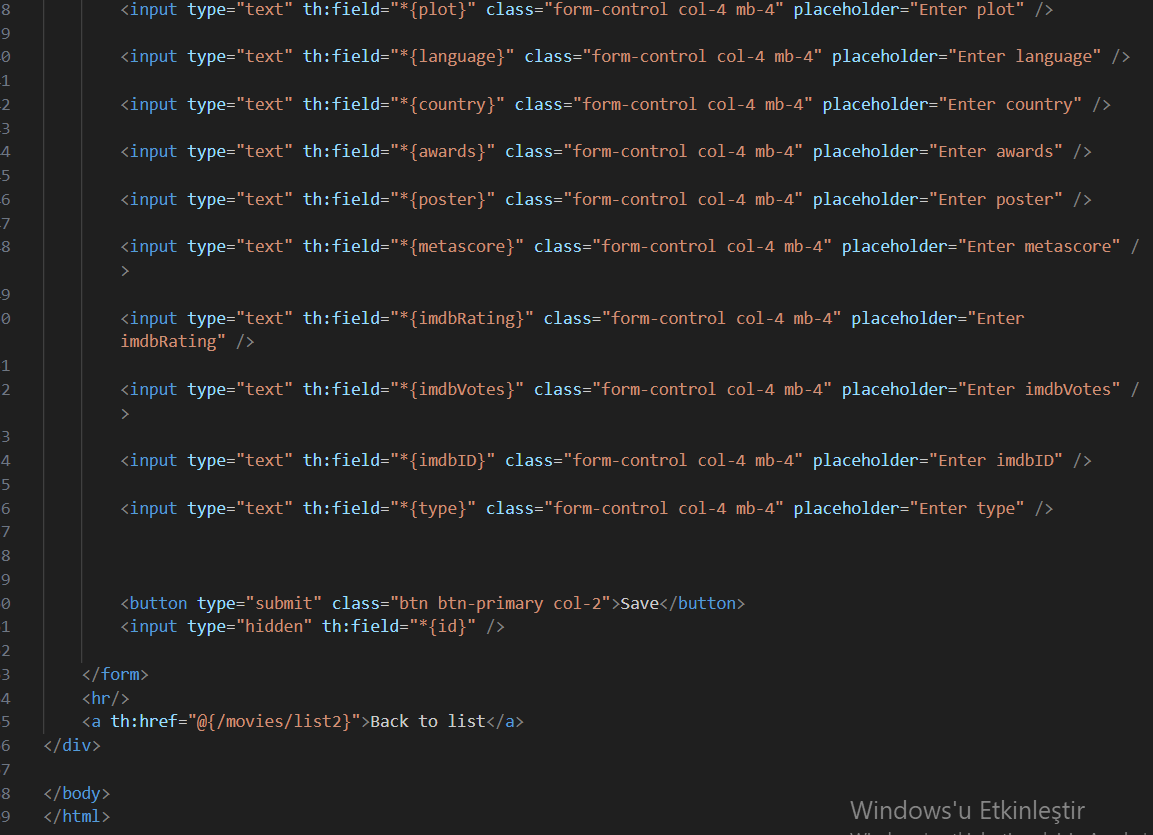
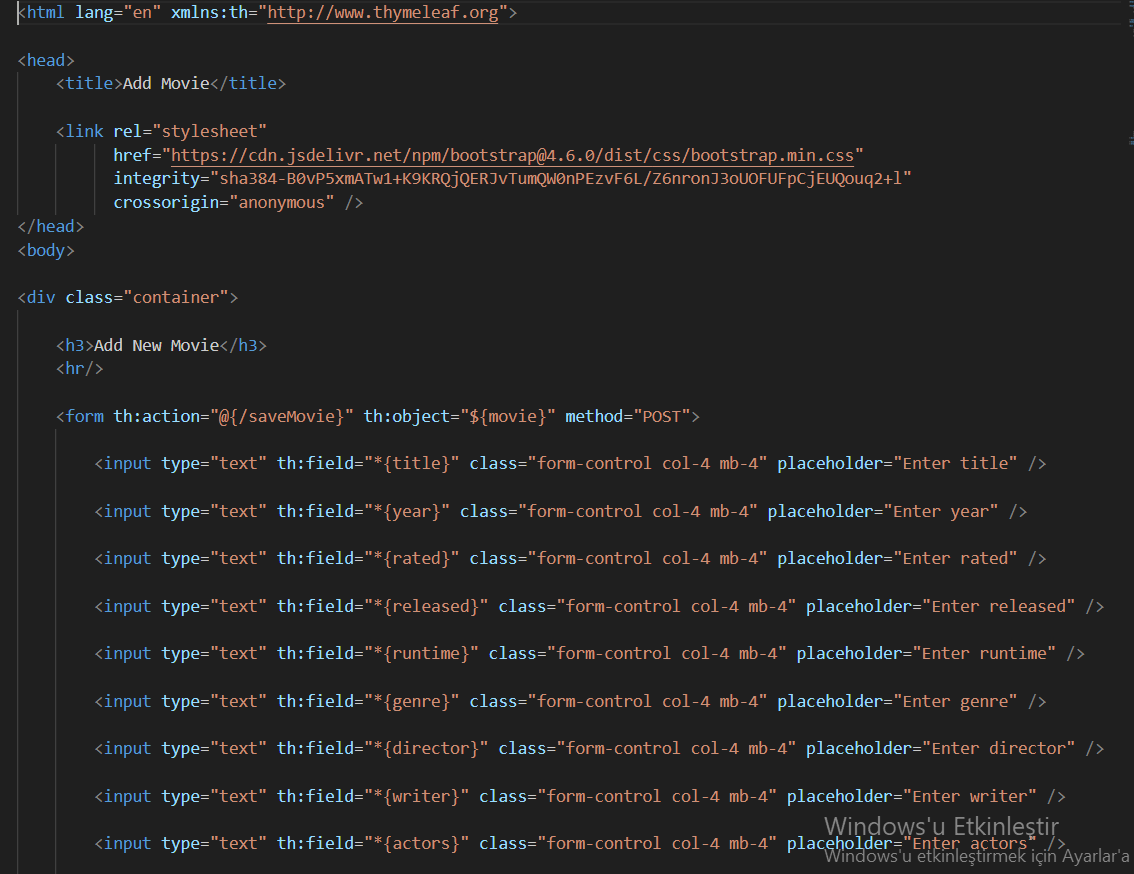
***3.4.5. Thymeleaf Kullanımı***

Thymeleaf araştırılarak Thymeleaf ile projenin frontendi oluşturuldu.

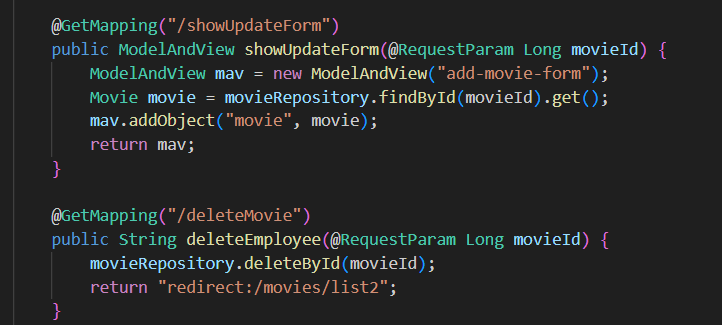
Frontend aşamaları şu şekildedir:

********

*Görsel 34 Görsel 35*

****

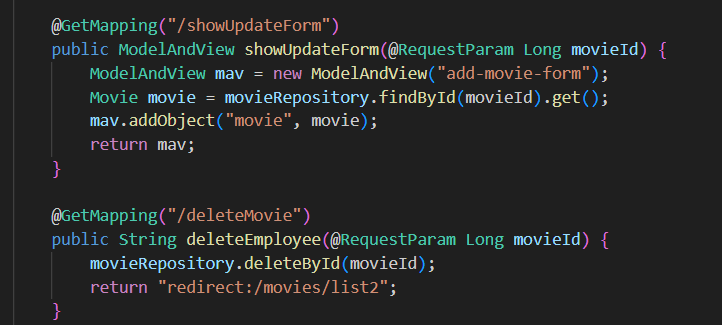
*Görsel 36**Görsel 37*

** **

*Görsel 38 Görsel 39*

****

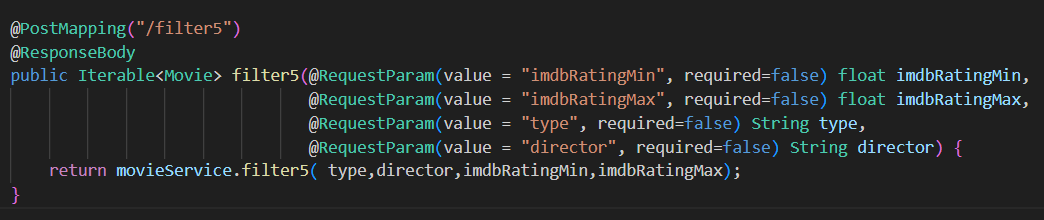
*Görsel 40*

********

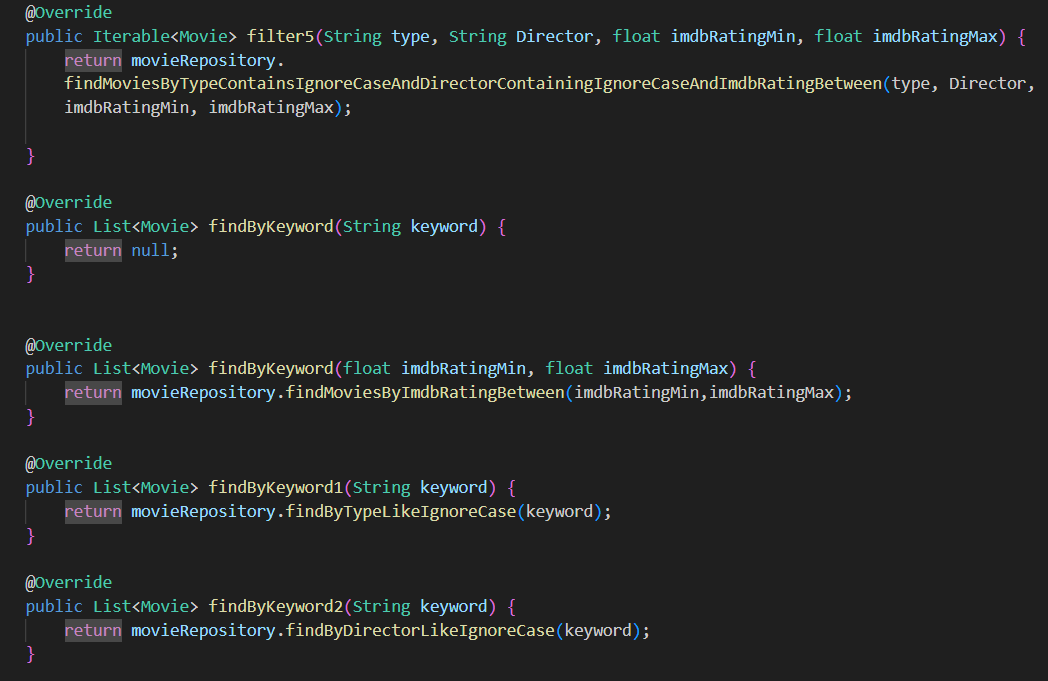
*Görsel 41 Görsel 42*

****

*Görsel 43*

****

*Görsel 44*

****

*Görsel 45*

Bu verilen görsellerle projenin detayları, aşamaları verilmiştir. Kapsamlı bir proje gerçekleştirilmiştir.

Entegratörün gerçekleştirdiği proje entegratör için fayda sağlamıştır ve kapsamlı,gerçek bir projede yer almak gelişimine katkı sağlamıştır. Aynı zamanda bu proje firmaya da gerekli katkıları sağlamıştır.

***3.4.6. Linux Komutları***

Projenin Dockerize işlemi için öncelikle Entegratöre Linux eğitimleri verildi bu eğitimler sayesinde komutlar öğrenildi ve entegratör gerekli işlemleri yapmaya hazır hale geldi.

**Eğitim Kaynağı :** <https://www.youtube.com/playlist?list=PL_c9BZzLwBRKNJSxrcYQld-3qLQfVMvvP>

Çalışılan Linux Komutları:

* Cd -> Dizin değişikliği yapar.
* Ls -> Verileri listeleme işleminde kullanılır.
* Echo -> Stringi yazdırma işleminde kullanılır.
* ( > ) Tekli yönlendirme işlemi yapar. Yönlendirdiğimiz dosya yoksa Shell bizim için oluşturur.Varsa içeriği tamamen silerek yeni yazdırdığımız içeriği yazdırır.

ÖRNEK : Echo $icerik > dosya\_adi

* (>>) Çift yönlendirme işlemi yapar. Bu işlem ile içerik silinmez içeriğe ek yapılır.

ÖRNEK : Echo icerik > > dosya\_adi

* Cat komutu -> Çıktıları terminalde bastırmak için kullanılır.
* Man komut\_adi -> Verilen komut ile kullanabileceğimiz şekilleri verir. Help sayfasına aktarır.
* Cat -n dosya\_adi -> Dosya içerisindeki verileri sayılandırır.
* Grep komutu -> Bir veri içerisinde istenilen kısmı vurgulayarak getirir.

ÖRNEK: cat dosya1 | grep Güvenlik

( dosya1 içerisinde Güvenlik kelimesinin geçtiği satırı Güvenlik kelimesini vurgulayarak getirir.)

* Linux komutlarında büyük/küçük harf duyarlılığı vardır.
* Cron -> Dosya içerisinde zamanlı görevler oluşturabiliriz.
* Gdm3 -> Arayüz ortamını denetler.
* Linux’ta her şey bir dosyadır.
* Touch -> Boş bir dosya oluşturur. (touch dosya\_adi)
* Rm -> Dosya siler. (rm dosya\_adi)
* Mkdir -> Klasör oluşturur. (mkdir klasör\_adi)
* Rmdir -> Klasörü siler. (rmdir klasör\_adi)
* Cd komutunu path belirtmeden kullanırsak ana dizine dönmüş olur.
* Mv -> Dosya taşıma işlemi yapar. (mv dosya\_adi dizin\_adi /)
* Mv -> ile dosyanın, dizinin adını da değiştirebiliriz. (mv eski\_ad yeni\_ad)
* Cp -> kopyalama komutudur. (cp birinci ikinci) (birincini içeriğini ikincinin içeriğine kopyalar.)
* Cp -r deneme / doc (denemenin içeriğini komple doc’un içeriğine atar yani dizini kopyalar.)
* Locate -> Veritabanında istenilen veriyi arar.
* Find -> “locate” veritabanından arama yaptığı için silinmiş dosyayı var olarak bulur. Ama “find” sadece var olan dosyayı bulur.

ÖRNEK: find/ -name içerik

* Tar -cvf -> Arşivleme işlemini yapar.

ÖRNEK : tar -cvf arşiv\_adi arşivlenenin\_adi

* Tar -xvf -> Arşivden çıkarma işlemi yapar.

ÖRNEK : tar -xvf çıkarılacak\_arsivin\_adi

* More -> Dosyanın içeriğini inceleyebiliriz, çıktı verir.
* Head dosya\_adi -> İlk 8 satırı verir.
* Tall dosya\_adi -> Son 8 satırı verir.
* Tac dosya\_adi -> İçeriği tersten yazdırır.

Mv videomeet / video -> dosyanın adını değiştirmiş olur.

* Linux’ta birden fazla kullanıcı hesap açabiliyor.
* Her kullanıcını İD’si vardır.
* Sıfırıncı İD Root kullanıcısını temsil eder.
* Root kullanıcısı sistemdeki en yetkili kullanıcıdır.
* Cat /etc/passwd -> Komutu sistemdeki kullanıcıları verir.
* Cat /etc/passwd | grep Elif -> Sistemdeki Elif adlı kullanıcıları getirir.
* Shell üzerinde komut çalıştıramayan kullanıcının login olmasının anlamı yoktur.
* 0 – 100 arası İD’leri genellikle sistem kullanıcıları oluşturur, onlar kullanır.
* Shadow -> Parola özetlerinin tutulduğu dosyadır.
* Cat /etc/group -> Var olan grupları verir.
* Gruplarda belirli İD’lere bölünür.
* Bir dosyanın sahibi ve grup sahibi olur. Grup sahibi dosya üzerinde hak iddia edebilecek diğer kişilerdir.
* Sahiplik yoksa root kullanıcısı olarak login olmamız gerekiyor yoksa dosya oluşturamayız. Ya da bunun için login olmadan “Sudo” kullanmamız gerekiyor.
* “Sudo” Root yetkisi ile arka planda işlem yapar.
* Sudo touch /etc/dosya1 -> etc içerisine dosya1 dosyasını oluşturur.
* Sudo rm /etc/dosya1 -> sudo ile dosya1’i siler.

“Sudo” komutunu çalıştırabilmek için ;

1. Sudo grubunda var olmak gerekiyor.
2. Sudoers dosyası içerisinde sudo çalıştırma yetkisinin var olması gerekir.

* Passwd kullanici\_adi -> Şifre değiştirme işlemi yapmış oluyoruz.
* Sudo su -> Root olarak terminal içerisine login ediyor.
* Su kullanici\_adi -> Geçeceğimiz hesabın kullanıcı adını yazıyoruz ve o hesaba geçiş yapıyoruz.
* Gedit -> Komutu terminalden gediti başlatarak bir metin dosyasını düzenlemek için kullanılır.
* Ls -> Listeleme işlemi yapar.
* Ls -la -> Daha ayrıntılı listeleme işlemi yapar.
* Ls -lna -> Group İD ve User İD olarak listeleme işlemi yapar.
* Root’un yetkisi kaldırılamaz.
* ./dosya\_adi -> Dosya çalıştırılır.
* Ls -l dosya\_adi | awk -> Bu komut sadece dosya izinlerini görüntüler.
* Reboot ve logout ile tekrar sistemi başlatabiliriz.

Sudo usermod -a -G yetkililer Elif -> Elif adlı kullanıcıyı yetkililer grubuna ekler.

Eğitimler sonrasında Linux Taskları iletildi.

Dosyalamaişlemleri araştırılıp öğrenildi.

* Dosya izinlerini değiştirmek için “chmod” komutu kullanılır.
* Chmod komutunda harfli ve sayısal kullanım vardır.
* Harfli kullanım ;

Chmod u = rwx, g = rw, o = x dosya\_adi (u = user, g = group, o = others)

* Sayısal kullanım ;

8’lik tabana karşılık gelir.

4 -> Okuma

1 -> Çalıştırma

2 -> Yazma

işlemlerine karşılık gelir.

* Chmod 777 dosya\_adi -> Tüm yetkileri verir.
* Chmod -004 dosya\_adi -> Otherdan read yetkisini kaldırdı.
* Chmod -010 dosya\_adi -> Gruptan çalıştırma yetkisini kaldırdı.
* Chmod -200 dosya\_adi -> Userdan yazma yetkisini kaldırdı.
* Chmod +222 dosya\_adi -> Herkese yazma yetkisini verir.
* Sudo adduser yönetici -> yönetici adlı kullanıcı ekliyor.
* Sudo groupadd yetkililer -> yetkililer adlı grup oluşturup, ekliyor.
* Sudo chown yetkilibey : Elif yetki -> yetki dosyasının sahibini Elif yerine yetkilibey yapar.
* Bir kullanıcıyı grup içerisine ya da birden fazla grup içerisine aynı anda ekleyebiliyoruz.
* Usermod komutu ile bu işlemi yapabiliriz.
* Sudo usermod -a -G yetkililer Elif -> Elif kullanıcısını yetkililer grubuna ekler.
* Userdel : Kullanıcıyı tüm sistemden kaldırır.
* Deluser : Sadece o kullanıcıyı o sistemden kaldırır.
* Groupdel : Grubu sistemden kaldırır.
* Ps -> Prosesleri görüntülemeyi sağlayan bir komuttur.
* Ps -a -> Tüm süreçler görüntülenir.
* Kill -> Sonlandırıcıdır.
* Kill – l -> Kill parametrelerini listeleyebiliriz.
* Xkill -> Donan bir programı kapatmak için kullanılır.
* Ps -auroot-> Root kullanıcısının prosesini verir.
* Killall gedit -> Gedit adındaki tüm prosesleri öldürür.
* Gedit : Prosesi yöneten programın adıdır.
* Kill -9 3926 -> Belirtilen prosesin öldürülmesini sağlar.
* Öldürülen proses tekrar kullanıldığında aynı İD adresine sahip olmayabilir. Proses durumuna göre o an ki prosesi alır.
* -9 (SIGKILL), -15(SIGTERM) -> Öldürme işleminde kullanılır.
* Jobs -l -> Arka planda çalışan programları görüntüleyebiliriz.
* Bg -> Arka planda çalışan programı verir.
* Servis kavramı : Arka pland çalışan, çeşitli komutlar bekleyen bir süreç yazılımıdır.
* Service -> komutu ile servisin durumunu görüntüleyebiliriz.
* Service –status -all -> Tüm servisleri verir.
* Bazı servisleri kullancı da yönetebilir, kullanıcının oluşturduğu kendi servisleri olabilir.
* Shell Scripting (Kabuk Programlama) : Shell’de çalıştırılabilecek komutların yazıldığı, derlemeye tabi tutulmadan oluşturulmuş, çalıştırılabilir dosyalardır.

Chmod +x shellscript -> Herkese çalıştırma yetkisi verir.

***3.4.7. Linux Task***

Linux çalışmalarından sonra Linux Task’ları yapılmaya başlandı.

Yapılan Linux Taskları şu şekildedir:

***Task 1* -** Create a file called keys.txt in your home folder and enter keys for future tasks in it. Add x as the first key.

- cd /home

- touch keys.txt

- echo gizli > keys.txt

- cat keys.txt (keys.txt nin içeriğini verir)

key => (gizli)

***Task 2* -** The key is stored in a file called readme in the home directory of the server1 server. Use user user2 and password xxx to log in to the server1 using SSH.

- user1 userına bağlı ol (su user1)

- ssh user2@ server1 (host server1ip değil)

- user2@ server1-VM:/home$ cd

- user2@ server1-VM:~$ ls

data passgen people.csv put\_keys\_here\_to\_ssh\_user3.pub

data.txt passgen.tar put\_keys\_here\_to\_ssh\_user3 readme

- user2@ server1-VM:~$ cat readme

key => (gizli)

***Task 3* -** The key is a filename in the data folder in the home directory with permissions of 644.

- user2@ server1-VM:~$ find ~/data -type f -perm 644 (Bu komut, ~/data dizini altında (~/ ev dizinini temsil eder) bulunan ve dosya tipinde olan (-type f) 644 izinlerine sahip dosyaları bulacaktır.)

/home/user2 /data/? (644 izinli olan dosya adı)

- user2@ server1-VM:~$ ls -l ~/data (Bu komut, ~/data dizinindeki tüm dosyaların listesini ve izinlerini gösterecektir)

key => (gizli)

***Task 4* -** The key is a filename in the data folder in the home directory with the group cronos.

- find ~/data -group cronos

(Bu komut, ~/data dizini altında (~/ ev dizinini temsil eder) bulunan ve cronos grubuna ait olan dosyaları bulacaktır. )

=> ( find ~/data -group cronos

/home/user2/data/?)

- Eğer dosya adını sadece görüntülemek istiyorsanız ve başka bilgilerle birlikte görüntülenmesini istemiyorsanız, ( -printf "%f\n" ) seçeneğini kullanarak sadece dosya adlarını görüntüleyebilirsiniz:

- user2@ server1-VM:~$ find ~/data -group cronos -printf "%f\n"

key => (gizli)

***Task 5* -** The key is stored somewhere on the server and has all of the following properties:

user : x

group : y

- user2@ server1-VM:~$ find / -user x-group y 2>/dev/null -printf "%f\n" ( 2>/dev/null : ile permission denied hatalarını gizleyebiliriz)

hidden\_file

- user2@ server1-VM:~$ cat /var/lib/misc/hidden\_file

key => (gizli)

***Task 6* -** The key is stored in the data.txt file in the home directory of user2 next to the word sordid.

- user2@ server1-VM:/home$ cd /home/user2

User2@ server1-VM:~$ ls

data data.txt passgen.tar people.csv put\_keys\_here\_to\_ssh\_user3 readme

bu şekilde user2 içerisindeki data.txt dosyasının var olduğunu gördük.

- user2@ server1-VM:~$ cat data.txt ile data.txt dosyasının içeriğine eriştik.

- user2@ server1-VM:~$ cat data.txt | grep sordid ile data.txt içerisindeki "sordid" kelimesini bulduk.

- sordid in yanındaki key => (gizli) dir.

***Task 7* -** The key is a string of characters from line 23 to line 30 of the first character of the first name in people.csv file.

- user2@ server1-VM:~$ cd /home/user2

- user2@ server1-VM:~$ ls

data data.txt passgen.tar people.csv put\_keys\_here\_to\_ssh\_user3 readme

- user2@ server1-VM:~$ cat people.csv

User Id,First Name,Last Name,Sex,Email,Phone,Date of birth,Job Title

(gizli )"people.csv" dosyasının içeriğinde bulunan ve ilk sıradaki kişinin adının ilk harfinden başlayarak 23. satırdan 30. satıra kadar olan karakterleri içeren bir anahtarı bulunması isteniyor. burdaki gibi

key => (gizli) olur.

***Task 8* -** The key is stored in /data/goOKzJp939 and can only be read by user user3. Generate an ssh key from your server and add it to put\_keys\_here\_to\_ssh\_user3 in the user2 home directory to ssh user3.

- public keyi user2@ server1'deki put\_keys\_here\_to\_ssh\_user3'a ekledik ki user1den ssh user3@ server1 dediğinde user3 'a geçerken şifre istemeden user3 kullanıcısına bağlanır.

- user3@ server1-VM:/home/user2$ ls

data data.txt passgen.tar people.csv put\_keys\_here\_to\_ssh\_user3 put\_keys\_here\_to\_ssh\_user3.pub readme

- user3@ server1-VM:/home/user2$ cd data

- user3@ server1-VM:/home/user2/data$ ls

(Gizli)

- user3@ server1-VM:/home/user2/data$ cat ?

key => ( gizli)

***Task 9* -** The key is the output of the passgen script contained in passgen.tar on the server1 server. Transfer passgen.tar from a server to your home directory. Untar passgen.tar and make it executable to run.

- user2@server1-VM:~$ scp passgen.tar user1@ip\_adres:/home/user1

( komutu ile user2’deki passgen.tar dosyasını user1 serverine atıyorum artık passgen.tar dosyası user1 serverında yer alır.)

- tar -xvf passgen.tar ile arşivden çıkarıp , chmod +x passgen ile çalıştırmak için kullanıcam ./passgen ile de çalıştırıyorum.

- user1@user1-VM:~$ tar -xvf passgen.tar

passgen

- user1@user1-VM:~$ chmod +x passgen

- user1@user1-VM:~$ ./passgen

key => ( gizli)

***Task 10* -** The key is the output of the running passgen 100 command. Before entering command, create a folder called files and create file1 file2 file3 file4 and enter passgen 100 to generate the key.

home > user > files > file1 file2 file3 file4 (Run passgen100 here)

(Directory must be in files folder [cd files])

- mkdir ile home dizininde files klasörünü oluşturdum.

- files içerisinde touch ile file1 file2 file3 file4 dosyalarını oluşturdum.

- user1@user2-VM:~$ cd /home

user1@user1-VM:/home$ ls

ansible user1 files keys.txt

- user1@user1-VM:/home$ cd files

user1@ user1-VM:/home/files$ ls

file1 file2 file3 file4

- user1@ user1-VM:/home/files$ /home/ user1/passgen 100

key => (gizli)

***Task 11* -** You can continue use keys.txt in your home folder and enter keys for future tasks in it. The key is server2 server IP's known by the server1 server. (See Session 1 for the user and password for the server1 server.)

- user2@server1-VM:~$ nslookup server2(serverın adı biliniyorsa nslookup ile ip adresi bulunur)

Server: gizli

Address: gizli

Name: gizli

Address: gizli

keys.txt ye Address: ip\_adres eklendi ( user1@user1-VM:~$ echo ip\_adres>> keys.txt )

key => (gizli)

***Task 12* -** The key is the other port number in server2 server that is open except for SSH. The server1 server is blocked in server2 server. Find a way to communicate with the server2 server on your server and look at the opened ports.

- netplan yapılandırma dosyası konumu /etc/netplan/ dizini olarak ayarlanmıştır. Diğer olası konumlar /lib/netplan/ ve /run/netplan/ dizinleridir. Ubuntu kurulumunuza bağlı olarak, gerçek Netplan yapılandırma dosyası aşağıdaki üç formdan birini alabilir.

01-netcfg.yaml

01-network-manager-all.yaml

50-cloud-init.yaml

- 50-cloud-init.yaml dosyası içerikteki değişiklikleri kaydetmediği için dosya izni değiştirildi

( sudo chmod 777 50-cloud-init.yaml )

- user1@server-VM:/etc/netplan$ cat 50-cloud-init.yaml

network:

version: - gizli

ethernets:

eth0:

addresses:

- - gizli

match:

macaddress: - gizli

nameservers:

addresses:

- gizli

search:

- gizli

routes:

- to: default

via: - gizli

set-name: eth0

enp6s20:

addresses:

- - gizli eklendi. Böylece Static IP tanımlaması yapılmış oldu .

- "sudo netplan apply" komutu ile değişiklikleri güncelledik

- "ip a" komutu ile istediğimiz ip nin tanımlanıp tanımlanmadığını görürüz.

( user@server-VM:/etc/netplan$ ip a

- nmap -p 1-65535 ip\_adres=> komutu ile ip adresinin 1-65535 arası açık olan port numaralarını verir

key => (x açık olan port numarasıdır.)

***Task 13* -** The key is the user of server2 server. You can get the user by using http get on the opened port of the server2 server. (Expect SSH Port)

- user1@server-VM:~$ curl http://ip:port

( Bu komut, server2 sunucusundaki belirtilen HTTP portuna (örneğin, 80 veya 8080) bir GET isteği gönderir ve sunucudan yanıt alır. )

key => (gizli)

***Task 14* -** The key is the password of server2 server. You can get the password by using http post on the opened port of the server2 server (Expect SSH Port). The password is in Base64 format, decode it by using base64 command.

- user1@server-VM:~$ curl -X POST -d parametre1=password http://ip:port

( Bu komut, curl aracını kullanarak HTTP Post isteği gönderir. -X POST seçeneği ile POST metodu belirtilir. -d seçeneği ile gönderilecek parametreler ve değerleri belirtilir. http://<server2\_ip>:<port> adresine POST isteği gönderilir. )

key => ( gizli )

-user1@server-VM:~$ echo gizli | base64 -d ( Bu komut, Base64 formatındaki şifreyi decode eder ve gerçek şifreyi ekrana yazar. )

key => ( gizli )

user4@server2 -VM:~$ ( user : user4 password: gizli - ssh user4@ip\_adres )

***Task 15*** - The key is the output of listening to port 8080 as UDP protocol on your server. Before listening add your ip to ips.txt in server2 server.

- user4@server2-VM:~$ ls

ips.txt

- user4@server2VM:~$ cat ips.txt

İp\_adres

İp\_adres

İp\_adres

- user4@server2-VM:~$ vi ips.txt (içeriğe ip yi ekledik)

- user4@server2-VM:~$ cat ips.txt

İp\_adres

İp\_adres

İp\_adres

İp\_adres

- user4@server2-VM:~$ nc -u -l 8080

key => ( gizli)

Linux Taskları net bir şekilde açıklanmıştır.

***3.4.8. Docker***

Böylece Entegratör projeyi Dockerize etmek için Docker’a başlangıç yaptı.

Docker için Mentor ile olan görüşme sonucunda bu süreçte kullanılacak kaynakların önemi vurgulanmıştır.

**Eğitim Kaynağı:**

[Docker Nedir Nasıl Kullanılır? | Part #1 | Image Nedir? Container Nedir? Docker Komutları (youtube.com)](https://www.youtube.com/playlist?list=PL_f2F0Oyaj4_xkCDqnRWp4p5ypjDeC0kO)

[Course: A'dan Z'ye Docker | Udemy Business](https://htrdc.udemy.com/course/adan-zye-docker/learn/lecture/19234312#overview)

İşletim Sistemi: Kullanıcının ana bilgisayarda diğer uygulamaları çalıştırmasına izin veren ana yazılımdır.

* Uygulamaların büyük çoğunluğu ortak kütüphanelerden yararlanmalarını ve belirli donanım ve ayrıntıları hakkında endişe etmemelerini sağlayan bir işletim sitemi için yazılmıştır.
* Kısaca işletim sistemi bilgisayar donanımı ile uygulamalar arasında köprü görevi görür.
* İşletim sistemi çok temelde 3 ayrı bileşenden oluşur:

- Uygulama

- Kullanıcı Arayüzü

- Kernel

Docker : En net tanımlamayla open source bir ‘container’ teknolojisidir. Docker, aynı işletim sistemi üzerinde, yüzlerce hatta binlerce birbirinden izole ve bağımsız containerlar sayesinde sanallaştırma sağlayan bir teknolojidir. Web uygulamalarımızın kolayca kurulumunu, testini, çalışmasını ve deploymentını sağlar. Bunun yanında sunucu maliyetlerini önemli ölçüde azaltır.

Neden Docker’a ihtiyacımız vardır?

* Kütüphane ve Bağımlılıkların birbirini etkilemesi.
* Tüm servisler uyumlu işletim sistemine sahip olmalı.
* Servis güncellendiğinde ona uygun olan işletim sistemi ve bağımlılıkların bulunması.
* Geliştirme ortamı oldukça zor ve zaman adına maliyetli.
* Componentlerin kendine ait kütüphaneleri ve bağımlılıkları olduğu anda artık bir “container” olur.
* Kendine ait izole edilmiş bir ortamda çalışır.
* Bağımlılıklar ile işletim sistemi arasındaki iletişimi “Docker” kurar.
* İşletim sisteminin bağımlılıkları ve kütüphaneleri hariç her componentin kendine ait, birbirini etkilemeyecek şekilde bağımlılıkları ve kütüphaneleri vardır. Buna “Container” denir.
* Container VM’e çok benzer.
* Her VM kendi işletim sistemine sahiptir.
* Containerların çalıştığı yerde sadece bir işletim sistemi vardır, bu işletim sisteminin çekirdeği kullanılır. (OS Kernel)

Container türleri :

* LXC
* LXD
* LXCFS

* Docker LXC container türünü kullanır.
* Bizim anladığımız kısma “High Level” , makinenin anladığı kısıma “Low Level” denir.
* Container yönetimi oldukça zor ve Low Level bir iştir.
* Docker bu işi kolaylaştırmak için High Level birçok araç sunuyor.
* Linux Kernel üzerinde Windows Container mümkün değildir.
* Windows üzerinde windows container da mümkün değildir. Docker container ile iletişim kurmak için araya Linux VM ekler. Yani yine Linux Kernel üzerinden işlem yapılmış olur.
* Windows işletim sistemi üzerinde değil de Windows Server üzerinde Docker kurarsak Windows base bir containerı ayağa kaldırabiliriz.
* HYPERVİSOR : Çoklu işletim sistemlerinin aynı donanım üzerinde çalışmasını sağlayan komut setidir.

VM ile Container arasındaki en büyük farkı açıklamak istiyorum.

* Herbir VM’in kendisine ait işletim sistemi vardır. Containerların kendisine ait işletim sistemi yoktur. Çünkü Containerlar yüklü oldukları işletim sisteminin Kernelini kullanır.
* Container :
* Düşük RAM kullanılır.
* Düşük CPU kullanılır.
* Daha az olana ihtiyaç olur.
* VM :
* Yüksek RAM kullanılır.
* Yüksek CPU kullanılır.
* Daha fazla olana ihtiyaç olur.
* Çünkü her VM’in kendisine ait işletim sistemi vardır.
* VM containera göre daha geç açılır kapanır. Çünkü ayrı CPU vardır.
* Image nedir ?

İçerisinde birçok farklı yapıyı barındıran yapılardır. (OS, Application vb. )

* Docker Hub üzerinde birçok İmage havuzu vardır.
* Container İmage’i çalıştırdığımızda elde edilen proses olarak düşünülebilir.
* Docker run -> Bir işlemi çalıştırmak ve container üretmektir.
* Mesela bir uygulamayı örneğin Mango uygulamasını yüklemeden çalıştırabiliriz.
* Docker run Mango -> diyerek çalıştırırız.
* Mangodan kurtulmak için bu uygulamayı durdurmamız yeterlidir. Uygulamayı Standalone çalıştırmış oluyoruz.
* Imageleri Docker üzerinde kullanırız.
* Docker images -> İndirilmiş olan image listesini verir.
* Her Image’in kendine ait İmage İD’si vardır. Docker tarafından bilinir.
* Docker ps - a / -all -> Hangi containerların çalıştırıldığını verir.
* Her containerın kendine ait “name” i var. Ve her imagein kendine ait “tag” ı var.

Çalışan herbir container farklı bir isme sahip olur. Ta ki farklı bir isimle isimlendirip çalıştırınca değiştirebiliriz.

* Docker start ContainerName -> İstenilen containerı çalıştırabiliriz.
* Docker stop ContainerName -> İstenilen containerı durdurabiliriz.
* ContainerName yerine container İD’nin 3 karakterini de yazabiliriz.
* Docker container rm $ (docker container ls -aq) -> ID’ler ile toplu silme işlemi yapar.
* Docker run redis : 5 -> Redisin 5. Sürümünü indirip, çalıştırır.
* Cat etc/os-release -> os-release’nin içeriğini verir.
* Docker run -d redis -> Redisi arka planda çalıştırır. Bizde o sırada işlem yapabiliriz.
* Docker attach 56CC (Container ID) -> ID’si girilen containerı bloklar, komut girdirmez.
* Docker attach -> Enter tuşuna bastıktan sonraki logları verir.
* Post Mapping :
* Docker run -p DIS\_PORT : İC\_PORT Mongo -> MongoDB çalıştırılırken iç ve dış portu veririz. Bu şekilde bağlantı sağlayarak containerı ayağa kaldırır.
* Docker inspect (Container ID / NAME) -> Container bilgilerini verir. Bu kullanım şekli image içinde kullanılır.
* Docker pull mysql -> Mysql’ i indirir.
* Docker host üzerindeki containerlar Stateless olarak çalışır.
* Yani herhangi bir bilgi içerisinde kayıt edilmez
* Container durdurulduğunda kayıt ettiğimiz bilgiler sonsuza kadar silinir.
* Docker bu durumu ortadan kaldırmak için Volume kullanır.

Bunu yapmak için ;

* Bir containerı ayağa kaldırdığımızda kayıt edilecek klasörün DOCKER HOST üzerindeki adresini belirtiriz.
* Böylece container içerisinde bir veri yazıldığında, Docker Enginee bunu alır,bizim mapping yaptığımız klasörün içerisine aktarır.
* Daha sonra container durdurulduğunda bilgiler Docker Host üzerinde kalmaya devam eder.
* Docker run -v /opt/data:/data/db -> Bu şekilde eklenmiş veriler silinmez, kaydolur.
* Docker interaktif terminal :
* Docker containerına bağlanmamızı ve içindeki işletim sistemi veya uygulamalarla etkileşimde bulunmamızı sağlar.

İçeriği görüntüleme, dosya düzenleme, komut çalıştırma ve hata ayıklamaya olanak tanır.

Attach Modu : Bir docker containerına bağlanmak için “docker attach” komutunu kullanabiliriz.

* Docker attach container\_İD

Detach Modu : Docker containerına bağlıysanız, bağlantıyı sonlandırmak istiyorsak klavyeden Ctrl + P ve Ctrl + Q tuşuna basarak detach moduna geçebiliriz.

Bu konteynerın çalışmaya devam etmesini sağlar ancak terminalden ayrılabiliriz.

Docker container logs : Belirli bir docker containerının loglarını görüntülemek için kullanılır.

* containerının adını / İD ’sini vererek kullanabiliriz.
* Docker containerlarının çalışma durumunu izlemek ve hata ayıklamak için oldukça yararlıdır.
* Docker container logs ab123

Docker run –name mongo-server –net custom-network -d mongo

(mongoyu custom-network üzerinden çalıştırır.)

Apt -get update -> işletim sistemindeki güncellemeleri alırız.

Curl -> curl üzerinden indirme işlemi yapabiliriz.

* Apt -get install curl -y

History -> yapılan geçmiş işlemleri gösterir.

Apt -get install nodejs -y -> nodejs’i kurar.

Docker Environment Değişkenleri araştırılıp çalışıldı.

DockerFile ile ENV komutu : DockerFile kullanarak bir imaj oluştururken ‘ENV’ komutunu kulanarak ortam değişkenlerini tanımlayabiliriz.

* ENV MY\_VARIABLE = my\_value

(Bu komut, MY\_VARIABLE adında bir ortam değişkeni oluşturur ve değeri my\_value olarak ayarlar.)

Docker Compose ile Environment anahtarı : Docker Compose kullanarak birden fazla containerı yönetirken ‘environment’ anahtarını kullanarak ortam değişkenlerini tanımlayabilirsiniz.

* Services :

Myservice :

İmage : myimage

Environment :

MY\_VARIABLE = my\_value

(Myservice adındaki konteyner için, MY\_VARIABLE adında bir ortam değişkeni oluşturur ve değeri my\_value olarak ayarlar.)

Docker run ile -e parametresi : Docker run kullanarak bir container başlatırken ‘-e’ parametresini kullanarak ortam değişkenlerini tanımlayabilirsiniz.

* Docker run -e MY\_VARIABLE = my\_value my\_image

(‘my\_image’ imajından bir konteyner başlatırken MY\_VARIABLE adında bir ortam değişkeni oluşturur ve değeri my\_value olarak ayarlar.)

Docker Network Türleri:

* Bridge : Default Docker ağıdır. Bu ağ Docker Host’u üzerinde çalışan containerlar arasında iletişimi sağlar.

* Her container Bridge ağına bağlanarak birbiriyle iletişim kurabilir.
* Docker run mongo -> containerlar default olarak bu ağa bağlanır ve bir İP adresi alır.
* Host : containerların Docker hostunun ağına doğrudan bağlanmasını sağlar.
* Containerlar, Docker Host’unun ağına bağladıkları için aynı İP adreslerini ve ağ bileşenlerini kullanır.
* Bu nedenle host ağıyla aynı İP adres aralığına sahip olurlar ve hostun ağındaki diğer cihazlarla aynı şekilde iletişim kurabilirler.
* Docker run mongo –network = host
* None : Bu ağ türü konteynerın ağa bağlı olmamasını sağlar.
* Containerlar bu ağ türü ile oluşturulduğunda, hiçbir ağ birimi veya İP adresi atanamaz.
* Bu nedenle dış dünya ile iletişim kuramazlar ve yalnızca Docker Host’u üzerinde çalışabilirler.
* Docker run mongo –network = none
* Kullanıcı Tanımlı : Bizlerin kendimize göre tanımlamış olduğumuz networklerdir.
* Docker network create --driver bridge –subnet 182.18.0.0/16 –gateway 182.18.0.1 custom-network

DockerFile : Kendine özgü kuralları olan bir dille yazılan ve bizlerin Docker İmage oluşturmamızı sağlayan dosyalardır.

* From imaj : tag = Oluşturulacak imajın hangi imajdan oluşturulacağını belirten talimattır.
* RUN Komutu : İmaj oluşturulurken Shell’de bir komut çalıştırmak istersek kullanılır.
* RUN apt-get update
* WORKDIR komut\_lokasyonu : Cd komutuyla istediğimiz klasöre geçmek yerine bu talimatı kullanarak geçeriz.
* WORKDIR /usr/src/app (Belirtilen klasör yoksa otomatik oluşturulur.)
* COPY Kaynak Hedef = İmaj içine dosya veya klasör kopyalamak için kullanırız.
* COPY /source/user/src/app
* EXPOSE port : Bu imajdan oluşturulacak containerların hangi portlar üstünden erişilebileceğini yani hangi portların yayınlanacağını belirtiriz.
* EXPOSE 80/tcp
* CMD Komutu : Bu imajdan container yaratıldığı zaman varsayılan olarak çalıştırılmasını istediğimiz komutu bu talimat ile belirleriz.
* CMD java merhaba

DockerFile Örneği :

FROM ubuntu : 18.04

COPY ./app

RUN make /app

CMD python /app/app.py

* Linux sunucuya Java uygulaması kurma adımları:

1. Fiziksel ya da sanal sunucu kur.
2. İşletim sistemi kur.
3. İşletim sistemini güncelle.
4. JRE kur.
5. Uygulamayı sisteme kopyala.
6. Sistem başlatıldığında uygulamanında uygulamanın başlaması için ayar yap.

Terminalde;

1. İlk iki adım yapıldı.
2. Sudo apt-get update -y
3. Sudo apt-get install default-jre -y
4. Git clone ile github üzerinden çek.

* Cd javamerhaba
* Ls
* Cd myapp
* Java merhaba (uygulama sistem üzerinden çalışır.)
* DockerFile dosyası, uzantısız ve D’si büyük bir şekide adlandırılmalıdır.

Daha sık değişen adımları DockerFile’in altına, daha az değişenleri DockerFile’in üstüne koyarsak, yukarıdan aşağı başlatıldığında herhangi bir değişiklik olmazsa cache’i kullanır. Bu durum image oluşturma süresini kısaltır.

* ADD ve COPY Farkı : ADD, copy ile aynı işi yapar. Fakat ADD bunun yanında dosya kaynağının bir url olmasına da izin verir.
* ADD ile kaynak olarak bir .tar dosyası belirtilirse bu dosya imaja .tar olarak sıkıştırılmış haliyle değil de açılarak kopyalanır.
* Örnek: ADD <https://wordpress.org/latest> .tar.gz/temp
* ADD uzak sunucudan alıp kopyalarken .tar’ı açmıyor.
* Folderdakini kopyalarken açar.
* ADD -> Wordpress
* COPY -> Wordpress.com.tr
* ADD + uzak sunucu -> latest.tar.gz
* ENTRYPOINT ve CMD Farkı : CMD ile yazdığımı container ile değiştirebiliyorum.
* ENTRYPOINT ile yazdığımı Run time da değiştiremez.
* CMD [“java”, “merhaba”]
* ENTRYPOINT [“java”, “merhaba”]
* Dockerile içerisinde hem CMD hem de ENTRYPOINT varsa ; Docker CMD talimatında yazılanları ENTRYPOINT talimatına parametre olarak ekler.
* FROM centos : latest
* ENTRYPOINT[“ping”]
* CMD [“127.0.0.1”] -> ENTRYPOINT çalışır sonra CMD ‘yi çalıştırır.
* CMD ile yazdığımı Run time da değiştiririz ve bu şekilde kolaylık sağlar.
* Her Docker imajında ya CMD ya da ENTRYPOINT olmalıdır.

EXEC ve SHELL Form / docker-compose konusuna çalışıldı.

* Exec Form : CMD [“java”,”uygulama”]
* Shell Form : CMD java uygulama
* Exec formda direkt proses çalıştırabilir.
* Shell Formda ise Shell’i açıp onun içerisinde çalıştırabiliriz.
* ENTRYPOINT ve CMD aynı anda kullanılıyorsa “Shell Form” kullanılmaz.
* Build ARG : DockerFile içerisinde ve imaj yaratılırken değişken kullanmak istersek ARG talimatını kullanırız.
* ARG ile oluşturulan değişken sadece imaj yaratılma aşamasında geçerli olur.
* İmage build komutuna --build -arg opsiyonu kullanılarak atanabilir.
* ARG ile tanımlanan değişkenler bu imagedan yaratılan containerın içerisinden erişilemez, ENV Variable değildir.
* Sistemden erişilen değişkenler tanımlamak istersek sadece ENV talimatını kullanıyoruz.
* ARG’ı sadece image oluşturma işlemlerinde değişken tanımlamak istersek kullanırız.
* Shell çalışmazsa ENV Variable’a erişemez.
* Docker Compose :
* Docker-compose up -> komutu ile istemi ayağa kaldırırız.
* Docker-compose down -> komutu ile çalışan tüm servisleri durdurur ve siler. Volume varsa silinmez.
* Docker-compose config -> .yaml dosyasını gösterir.
* Compose için service, container demektir.

Docker-compose.yaml dosyasında ,

* Version
* Services
* Volumes
* Networks
* Secrets kısımları yer alır. Top-level veri bloglarıdır.

Docker Service Mode :

Replicated : Oluşturmak istediğiniz servisin kaç replica içereceğini belirtirsiniz. Swarm uygun olan nodelar üzerinde o sayıda replica oluşturur.

* Aynı uygulamayı neden farklı yerde copy olarak çalıştırmak isteyelim ?

( - Uygulamanın yüksek erişilebilirliğe sahip olması için olabilir. )

( - Uygulamanın yükünün farklı yerlere dağıtılması için olabilir. )

Global : oluştrmak istediğiniz servisin kaç replica içereceğini belirlemezsiniz. Swarm altındaki her node üzerinde 1 replica oluşturulur.

* Docker Swarm Cluster’da oluşturabileceğimiz en temel obje ‘Servisler’dir.
* Docker swarm init –advertise-addr 192.168.0.22

(Adresi docker swarm manager haline getirir.)

* Docker node ls -> Docker swarm classları gösterir.

Service çalıştırma komutu :

* Docker service create [options] image [command] [arg]
* Docker service create –name test nginx (image)
* İnspect : Bir şeyin detaylarını görmek istediğimizde kullanıyoruz.
* Docker service inspect test
* Docker service create –name = glb --mode = global nginx

( global modda service yarattı. )

* Bir container silinsede hemen bir tane daha yaratılır. Global Service’deki kural her node da bu service’in bir tane replicası mutlaka çalışmalıdır.

Entegratör bu eğitim sonucunda :

* Container
* Container – VM
* Docker Images
* Docker Komutları
* Volume Mapping
* Docker attach / detach mode
* Docker environment değişkenleri
* Docker Network Türleri
* DockerFile
* Linux sunucuda JAVA uygulaması kurma adımları
* ADD ve COPY Farkı / ENTRYPOINT ve CMD farkı
* EXEC ve SHELL Form / docker-compose
* Docker Service Mode

Konuları hakkında bilgi edindi.

***3.4.9. Dockerize İşlemi***

Projenin devamı olan Dockerize işlemi için gerekli notlar tutuldu.

Proje Dockerize edildi. Yapılan İşlemler şu şekildedir:

Maven komutlarını çalıştırabilmek için bilgisayarda bulunan "apache-maven-3.9.6-bin.tar.gz " dosyası server1 serverına atıldı.

- C:\Program Files> scp -P 18024 apache-maven-3.9.6-bin.tar.gz user1@ip\_adres:/home/server1

\* Bilgisayarda bulunan "movie\_demo-main.rar" dosyası server1 serverına atıldı.

- C:\Program Files> scp -P 18024 movie\_demo-main.rar user1@ip\_adres:/home/server1

\* mkdir test => "movie\_demo-main.rar" dosyasını test dosyası üzerinde açıp karışmaması sağlandı.

\* unrar x movie\_demo-main.rar test => movie\_demo-main.rar dosyasını test içerisinde açar.

\* user1@server1-VM:~/test/movie\_demo-main$ cd movieDemo

\* user1@server1-VM:~/test/movie\_demo-main/movieDemo$ ls

-HELP.md mvnw mvnw.cmd pom.xml src target

\* user1@server1-VM:~/test/movie\_demo-main4/movieDemo$ ls

- HELP.md mvnw mvnw.cmd pom.xml src target

\* user1@server1-VM:~/test/movie\_demo-main4/movieDemo$ cd target/

\* dog user1@server1-VM:~/test/movie\_demo-main4/movieDemo/target$ ls

- classes generated-sources maven-archiver maven-status movieDemo-0.0.1-SNAPSHOT.jar movieDemo-0.0.1-SNAPSHOT.jar.original

=> JAR dosyasını aşağıdaki komutla arka planda çalıştırın:

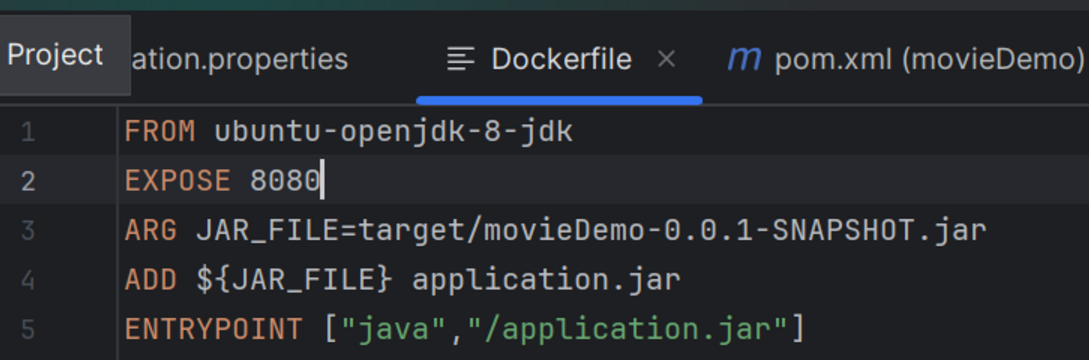
\* user1@server1-VM:~/test/movie\_demo-main4/movieDemo/target$ nohup java -jar movieDemo-0.0.1-SNAPSHOT.jar > /dev/null 2>&1 &

- [1] gizli

\* user1@server1-VM:~/test/movie\_demo-main4/movieDemo/target$ ps aux | grep movieDemo-0.0.1-SNAPSHOT.jar

- user1 gizli java -jar movieDemo-0.0.1-SNAPSHOT.jar

- user1 gizli grep --color=auto movieDemo-0.0.1-SNAPSHOT.jar

****

*Görsel 46*

Bu şekilde projenin Dockerize edilme aşamaları sunuldu. Artık proje tam anlamıyla bitti ve mentore sunularak teslim edildi. Böylece iş yerine katkı sağlandı.

***3.4.10. Database Design***

Proje bittikten sonra mentor yönlendirmesiyle veritabanı çalışmaları yapılarak sosyal medya platformlarının Veritabanı Analizi yapılarak Database Design işlemi yapıldı.

Yapılan veritabanı çalışmaları şu şekildedir:

Veri : İşlenmemiş ham bilgi veridir.

Veritabanı :

* Birbiriyle ilişkisi olan verilerin tutulduğu,
* Kullanım amacına uygun olarak düzenlenmiş, veriler topluluğunun,
* Mantıksal ve fiziksel olarak tanımlarının bulunduğu bilgi depolarıdır.

Veritabanı Yönetim Sistemi :

* Yeni bir veritabanı oluşturmak,
* Veritabanını düzenlemek,
* Kullanmak,
* Geliştirmek,
* Bakımını yapmak için çeşitli karmaşık işlemlerin gerçekleştirildiği bir yazılım sistemidir.

Veritabanı Yönetim Sistemlerinin Sınıflandırılması:

* Veri Modeline Göre;
* Hiyerarşik
* Ağ
* İlişkisel
* Nesneye Yönelik
* Kullanıcı sayısına göre;
* Tek kullanıcılı
* Çok kullanıcılı

“Hiyerarşik veri tabanları” veri tabanları için kullanılan ilk modeldir**.**

*Tablo 1*

****

Günümüzde hemen hemen tüm veritabanları bu yapıdadır.

Neden veritabanı kullanılır ?

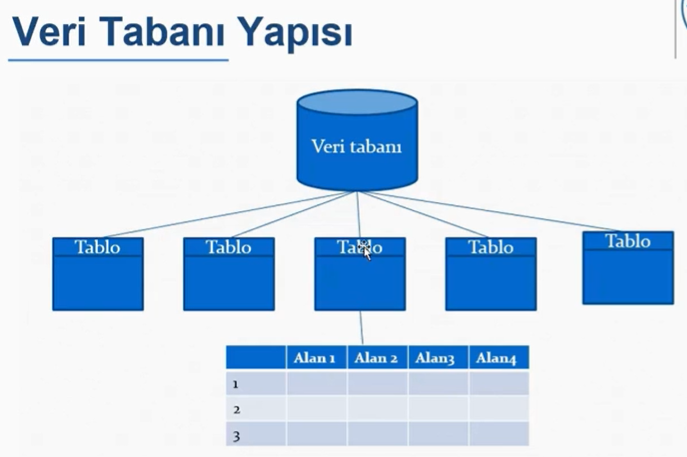
* Verilerin tutulması, saklanması ve erişilmesinde geleneksel yaklaşım verilerin ayrı ayrı dosyalarda gruplanması yaklaşımını kullanmkatadır.
* Verilerin artması, verilere aynı anda erişme ve düzenlenme ihityacı ile geleneksel yaklaşım yetersiz kalmıştır.

Veritabanı Yaklaşımının Avantajları :

* Ortak verilerin tekrarlanmasını önler.
* Verilerin merkezi denetiminin ve tutarlılığının sağlanması.
* Veri paylaşımının sağlanması.
* Fiziksel yapı ve erişim yöntemi karmaşıklıklarının, çok katmanlı mimarilerle kullanıcıdan gizlenmesi.
* Her kullanıcıya yalnız ilgilendiği verilerin, alışık olduğu kolay, anlaşılır yapılarda sunulması.

Veritabanı Yönetim Sistemleri :

* Oracle Database
* IBM DB/2
* Microsoft Access
* Microsoft SQL Server
* MySQL
* SQLite
* CSQL

****

*Görsel 47*

Veri Türleri :

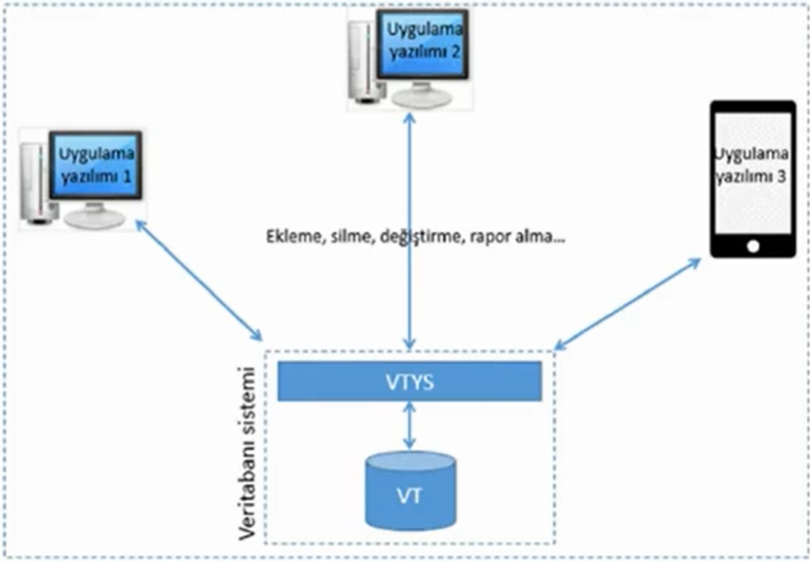
* Veritabanında tutulan kayıtların yapısı hakkında bilgi sahibi olmak için alanların bazı özelliklerinin önceden tanımlanması gerekir.
* Örneğin; Personel sicil numarası mutlaka tam sayı olması gerekirken, ad soyad kısmının harflerden oluşması gibi bu duruma örnek verebiliriz.

SQL Veri Türleri :

* TINYINT / SMALLINT / MEDIUMINT / INT / BIGINT
* FLOAT
* DOUBLE
* DATETIME / DATE
* CHAR / VARCHAR
* TEXT / MEDIUMTEXT
* BOOL

Veritabanı Yönetim Sistemi :

* Veritabanını oluşturmak, tabloları oluşturmak,
* Veritabanından veri okumak ve verileri güncellemek,
* Veri değerlerine ilişkin sınırlamaları gerçekleştirmek,
* Bir kullanıcının işleminin diğer kullanıcıyı engellemesini önlemek,
* Kullanıcıların yetkileri ölçüsünde etkinlikte bulunmalarına izin vermek,
* Veritabanındaki verileri yedekleme.

****

*Görsel 48*

Kaliteli bir veritabanı uygulaması geliştirmenin ön koşulu uygulama amacının yeterli kapsam ve açıklıkta belirlenmesidir.

Primary Key (Birinci Anahtar) :

* Kayıtlarda, iki kaydı birbirinden ayıran bir alanın olması gereklidir. Bu özellik kayıtlara erişimi kolaylaştırmaktadır. Her bir kaydı tek olarak tanımlamak için kullanılan, kayıttaki bir alana ya da alanlara Birincil anahtar denilmektedir.
* Bir kayıta ulaşmayı sağlayacak anahtar veridir.

Foreign Key (Yabancıl Anahtar) :

* Yabancıl anahtar, başka bir tablodaki birincil anahtara karşılık gelen değerleri oluşturan sütun ya da sütun gruplarıdır. Yabancıl Anahtar, tekil olmak zorunda değildir.

Varlıklar Arası İlişkiler :

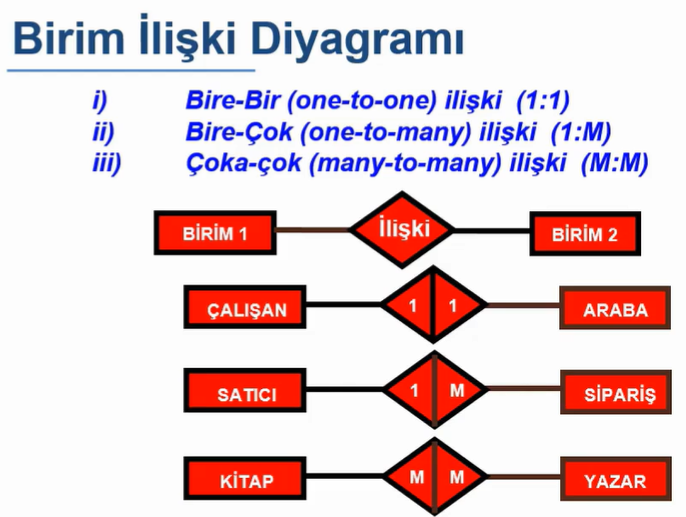
* Bir varlıkla ilişkili olabilecek varlıkların sayısına eşleme sayısı adı verilir.
* Eşleme sayısı n>=2 varlık için söz konusudur ve ikili ilişkilerin ortaya konulması açısından yararlıdır.
* A ve B gibi iki varlık kümesi arasındaki R ilişki kümesi için eşleme durumları şu şekilde ifade ediliyor :

~ Birden-bir (One to One)

~ Birden-çoğa (One to Many)

~ Çoktan-bire (Many to One)

~ Çoktan-çoğa (Many to Many)

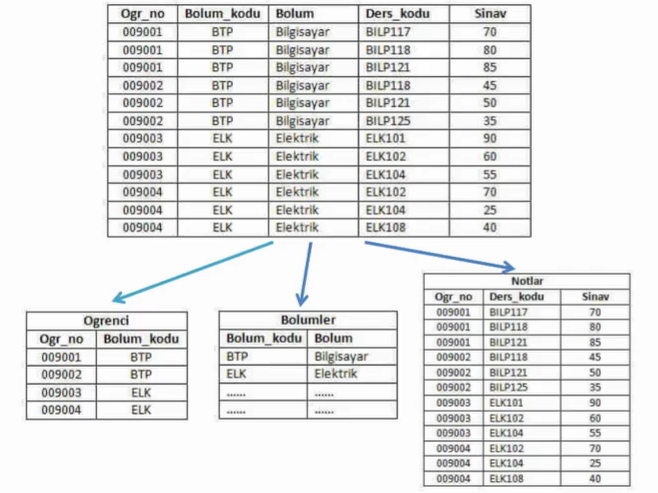
****

*Görsel 49*

Normalizasyon : Veritabanının tasarım aşamasında veri tekrarını, veri kaybını veya veri yetersizliğini önlemek için gerçekleştirilen işlemlere “Normalizasyon” denir.

* Genel olarak normalizasyon için ek tablolar gerekir.

*Tablo 2*

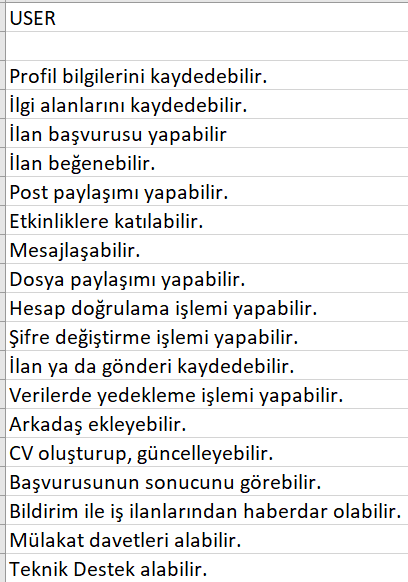
****

* Normalizasyon, verileri belirli bir aralığa veya dağılıma dönüştürerek, veri setlerini daha iyi karşılaştırılabilir hale gelmesini sağlar.
* Normalizasyon işlemi, veri setlerinin daha iyi performans göstermesini sağlar.
* Normalizasyon işlemi, optimizasyon algoritmalarının daha iyi çalışmasını sağlar.
* Normalizasyonun ana amaçlarından biri, veriler arasında büyük ölçek farklılıklarının neden olduğu problemleri ortadan kaldırmaktır.

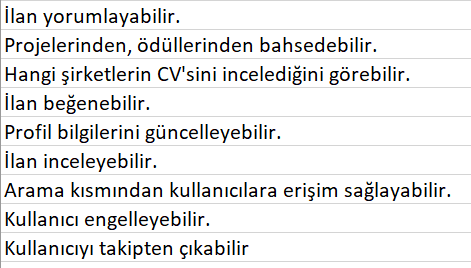
Yapılan Veritabanı çalışmalarından sonra, Sosyal medya platformlarının Veritabanı Analizi yapıldı.

Veritabanı Analizi şu şekildedir:

*Tablo 3 Tablo 4*

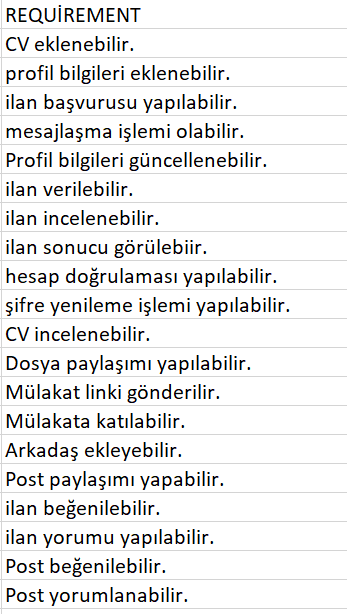
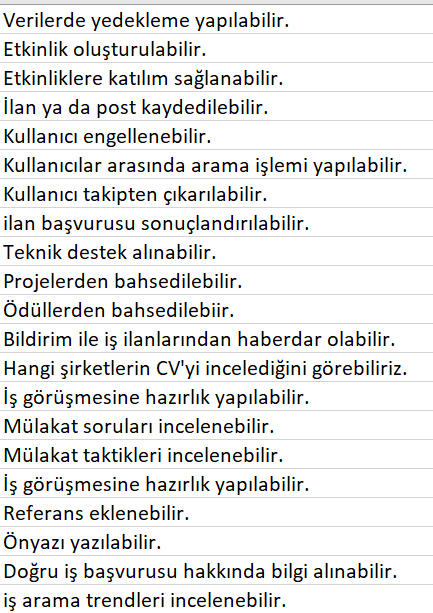
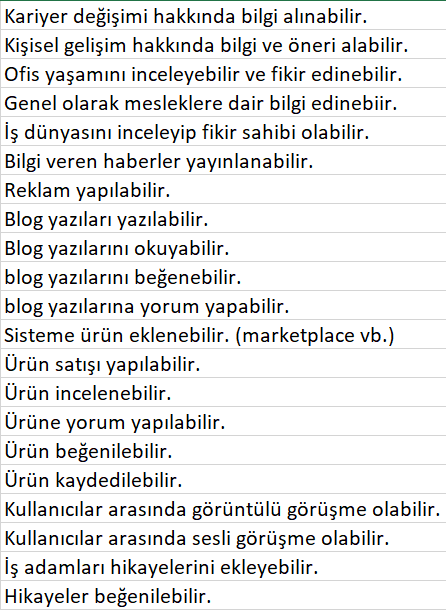
** **

*Tablo 5*

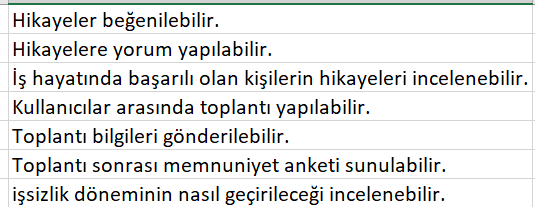
****

*Oluşturulan* Requirement List şu şekildedir:

*Tablo 6 Tablo 7 Tablo 8*

**  **

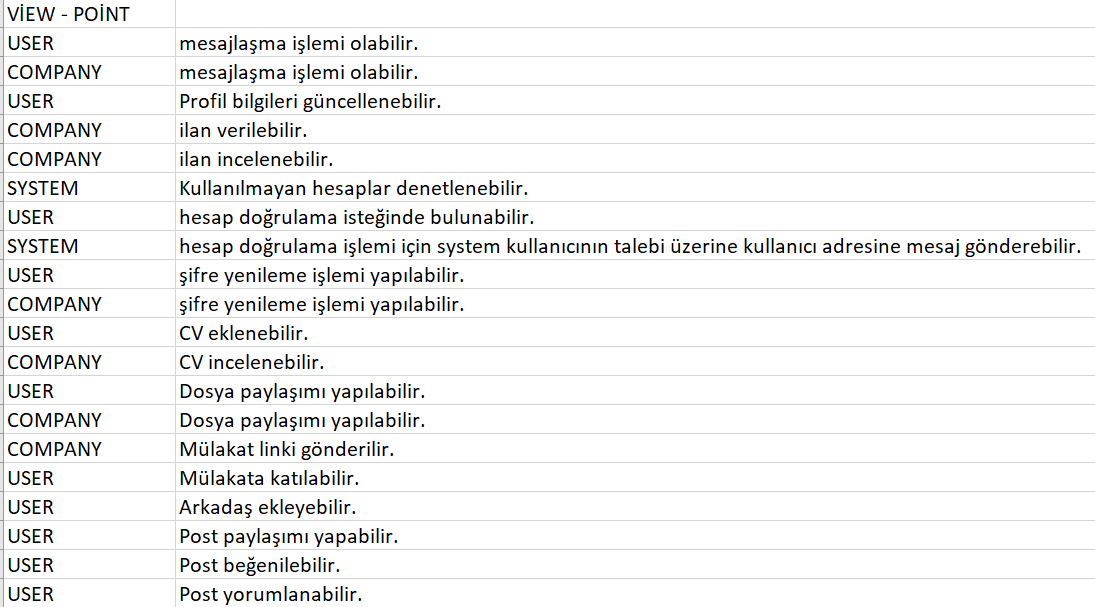
*Tablo 9*

****

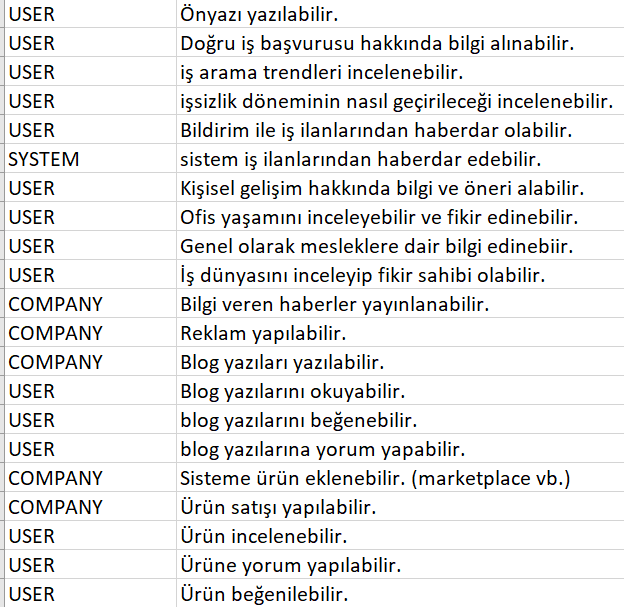
Sonrasında View – Point oluşturuldu.

Oluşturulan View – Point ‘ler şu şekildedir:

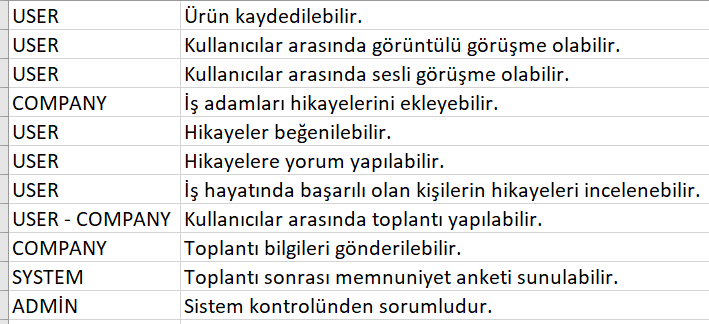
*Tablo 10*

****

*Tablo 11**Tablo 12*

** **

*Tablo 13*

****

Bu şekilde Database Design aşamaları sunuldu.

**4. TARTIŞMA VE SONUÇ**

Yapmış olduğum çalışmalar kapsamında edindiğim beceriler ve tecrübeler;

* Yeni bilgiler,
* Yeni teknolojiler,
* İş yeri kuralları,
* İş yeri ortam koşulları,
* İş yerine adaptasyon,
* İş yeri motivasyonu,
* Çalışma koşulları,
* Gizlilik kuralları,
* Bir işi tamamlama süreci,
* Yönetici – çalışan ilişkisini görmek,
* Ekip arkadaşları ile iletişim kurmak,
* Takım ile çalışmak,
* Takıma uyum sağlamak,
* Sanal Makine kullanımı,
* Linux / Linux komutları öğrenimi,
* SOLİD Prensipleri,
* Spring Boot Framework,
* Spring Boot Projesi gerçekleştirmek,
* Postman Kullanımı,
* Thymeleaf kullanımı,
* H2 database kullanımı,
* Docker / Docker komutları öğrenimi,
* Database,
* Database Design,
* ER Diagram

Yukarıda bir kısmı verilen birçok konu, davranış şekli ve yeni teknolojiler öğrenmiş oldum. Öğrendiğim bilgiler doğrultusunda iş ortamının gerekliliklerini görerek uyum sağlayabildim ve tecrübe edindim.

İş yerinin çalışma koşulları, çalışanlar için en verimli olacak şekilde düzenlenmiştir. Herkesin kendisine özel çalışma ortamının bulunması, ek araçların verilmesi hem çalışanların ergonomik koşullarda çalışmasını sağlamış hem de çalışanlar arasında daha kolay ve sağlıklı bir şekilde iletişim kurulmasını sağlamıştır.

Çalışanların iletişim kurduğu platformun gelişmiş olmasından dolayı bir iş sürecinde çok rahat bir şekilde ilerleme sağlayabiliyorlar. Yöneticiler sayesinde iş süreci sorunsuz bir şekilde hallediliyor ve başarılı bir şekilde sonuç alıyorlar.

Böylece çalışanların rahat ve sorunsuz çalışması iş yerinin gelişimini sağlıyor ve geleceğe yönelik sağlam adımlarla ilerlemesinde katkıda bulunuyor.