OtelBurdaApp

Elif Özkan  
 *211307057*  
Kocaeli Üniversitesi Bilişim Sistemleri Mühendisliği

*Özet*—OtelBurdaApp listelenen otel listelerine göre yorum getiren sanal bir bulut ortamında docker aracılığı ile Konteyner mantığı ile ayrılmış bir uygulamadır. Bu uygulama hem kullanıcı dostu olmasıyla beraber sanallaştırma teknolojileri kullanılarak hazırlanmıştır. Dockerfile kullanılarak konteynerlara ayrılıp Mysql yoluyla veriler tutulmuştur. Geliştirilmesi devam edilmesi durumunda yayını da yapılabilir.

*Abstract*—OtelBurdaApp is an application separated by Container logic via Docker in a virtual cloud environment that brings comments according to the listed hotel list. The application is both user-friendly and has been prepared using virtualization technologies. Data was separated into containers using Dockerfile and stored via Mysql. If its development continues, it may also be published.

Keywords— nodejs, bulut bilişim, docker, sanallaştırma, Mysql, Docker

# Docker

Docker aslında bir uygulama sanallaştırma ve dağıtım platformudur. Uygulamanın izole bir ortamda ve verimli kaynak kullanımı ile en optimize şekilde çalışmasını sağlar. Bu uygun çalışma ortamını konteyner adlı bir yapı ile sağlar. Docker konteynerleri, hızlı bir şekilde başlatılabilir ve durdurulabilir. Bu özellik, geliştiricilere uygulamalarını hızlı bir şekilde test etme ve dağıtma imkanı tanır. Ayrıca, daha hızlı iterasyon süreçleri ve sürekli entegrasyon/düzenli dağıtım (CI/CD) uygulamaları için idealdir. Konteyner teknolojisi sayesinde uygulama tüm bağımlılıkları ile birlikte paket halinde bulunur. Farklı ortamlarda da tutarlı bir şekilde çalışmasını sağlar. Aynı zamanda Docker Hub sayesinde geliştiriciler farklı kullanıcıların kullandığı imagelara ulaşabilir ve kendi oluşturduklarını paylaşabilir.

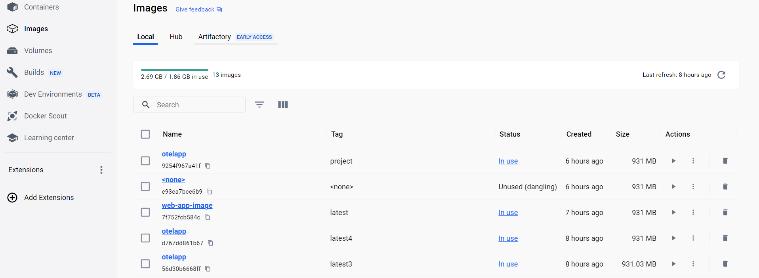
# Docker’da Imagelar

## Image Nedir? Neden Önemlidir?

Docker Image bir uygulamanın ve onun bağımlılıklarının önceden tanımlanmış ve bir paket halinde saklanmış halidir. Docker konteynerları bu image yapıları sayesinde oluşur. Aynı zamanda image izolasyon ve güvenlik ortamını yaratır. Image yapısı sadece bir projeye özgü değildir. Ve Farklı yapılar ve projeler ile kullanılabilir. Docker image'lar genellikle belirli bir işlevi yerine getirmek üzere yapılandırılmıştır. Örneğin, bir web sunucusu, veritabanı sunucusu veya belirli bir programlama dilini içeren image'lar olabilir. Bu, geliştiricilerin ve sistem yöneticilerinin belirli görevleri hızlı bir şekilde yerine getirmelerini sağlar.

## Uygulama için Oluşturduklarım

Ben de sanallaştırma uygulamamı yaparken uygulama için farklı image yapıları oluşturdum. Bu yapıları uygulama içinde Dockerfile dosyası oluşturup build ederek oluşturdum.



**Şekil-1**

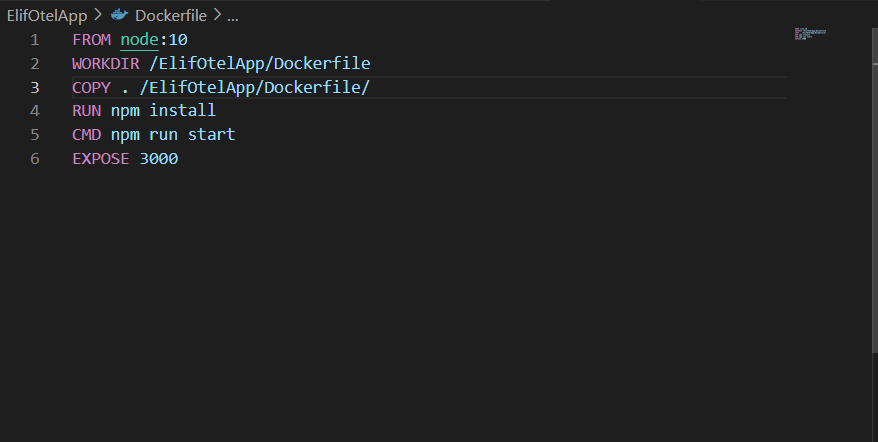
# Neden Uygulama Sanallaştırma

Sanallaştırma pek çok farklı versiyonu olan bir işlemdir. Bunlara değinmek gerekirse uygulama sanallaştırma, ağ sanallaştırma, uygulama sanallaştırma , sunucu sanallaştırma örnek verilebilir. Hepsi de bilgisayar kaynaklarını verimli bir şekilde kullanmayı amaçlar. Mantıksal olarak bölümlemeyi sağlar. Benim uygulama sanallaştırmayı seçme sebebim yaptığım web uygulamasını en izole , en güvenli ve optimize şekilde bu şekilde çalıştırabileceğimi düşünmemdi. Uygulama sanallaştırma, uygulamanın farklı ortamlarda (geliştirme, test, üretim) tutarlı bir şekilde çalışmasını sağlar. Bu, taşınabilir bir geliştirme ve dağıtım süreci sağlar. Bununla birlikte Uygulama sanallaştırma, uygulamanın çalışma ortamını izole ederek, bir uygulamanın diğer uygulamalardan ve sistemden bağımsız olarak çalışmasını sağlar. Uygulama sanallaştırma, konteynerlerin hızlı bir şekilde başlatılabilmesini ve durdurulabilmesini sağlar. Bu, hızlı iterasyon süreçleri ve sürekli dağıtım için önemlidir. Uygulama sanallaştırma, modern uygulama geliştirme süreçlerinde ve mikro hizmet mimarilerinde yaygın olarak kullanılan bir yaklaşımdır. Bu sayede geliştiriciler, uygulama bağımlılıkları ve yapılandırmaları ile birlikte hızlı ve güvenli bir şekilde uygulama geliştirebilir ve dağıtabilirler.

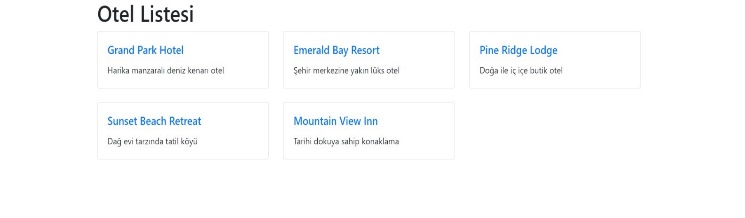
## Uygulama Sanallaştırma Aşamaları

Uygulama sanallaştırma uygulamanın bir sanal makine veya konteyner içinde bölümlemeler halinde çalışır. Uygulamanızın çalışabilmesi için gerekli olan tüm bağımlılıkları ve yapılandırmaları belirleyin. Bu, uygulamanızın doğru bir şekilde sanallaştırılabilmesi için önemlidir. Docker konteynerleri genellikle hafif ve hızlı olmaları nedeniyle popüler bir seçenektir. Eğer Docker veya benzeri bir konteyner teknolojisi kullanıyorsanız, Dockerfile veya benzeri bir konfigürasyon dosyası oluşturun. Bu dosyada, uygulamanızın bağımlılıkları, çalışma zamanı ve konfigürasyonu tanımlanır. Sanal makine veya konteyner içinde, uygulamanızın ve bağımlılıklarının kurulumunu gerçekleştirin. Bu adım, uygulamanızın sanal ortam içinde çalışabilmesi için gereklidir.

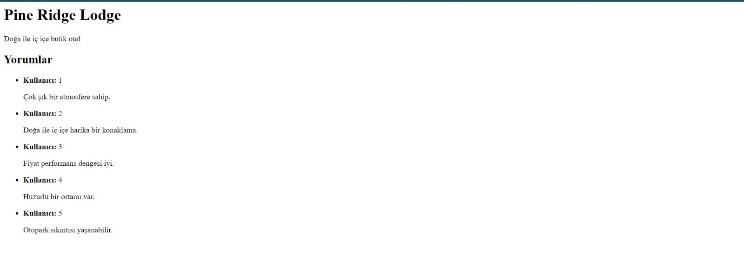
Uygulama sanallaştırma, izolasyon ve güvenlik sağlamak adına önemlidir. Konteynerlerde güvenlik seviyelerini belirleyin, gerekirse kullanıcı izinleri düzenleyin ve gerektiğinde izolasyon önlemleri alın. Bu uygulamanın hem izolasyonunu hem de gerekli güvenliğini sağlar. Performans iyileştirmeleri, kaynak kullanımı optimizasyonları ve diğer iyileştirmeleri uygulayarak uygulamanın daha etkili ve verimli bir şekilde çalışmasını sağlamak için gerektiğinde düzenlemeler yapın. Bu aşamaları sırayla gerçekleştirdiğinizde uygulamanız güvenle sanallaşır. Uygulama sanallaş kısmı bittikten düzenli olarak bakımının yapılması ve izlenmesi gerekli güncellemelerin yapılması da büyük önem taşır. Ben de uygulamamı sanallaştırırken backend ve front end kısımlarının ayarlamasını yaptıktan sonra dockerfile dosyası oluşturdum.



**Şekil-2**

****

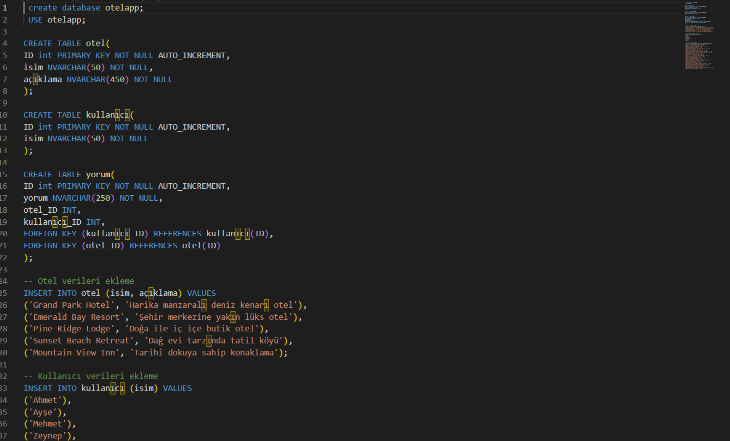
**Şekil-3**

****

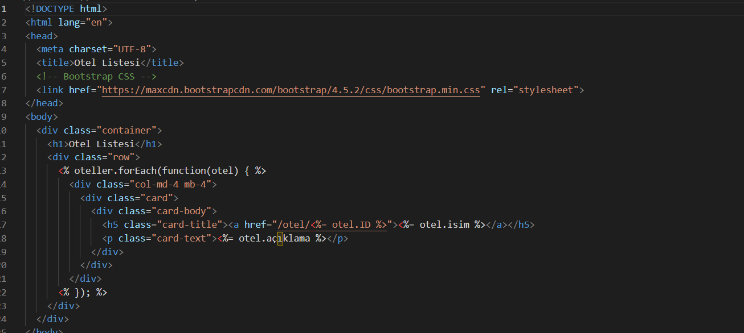
**Şekil-4**

## Uygulama Detayları

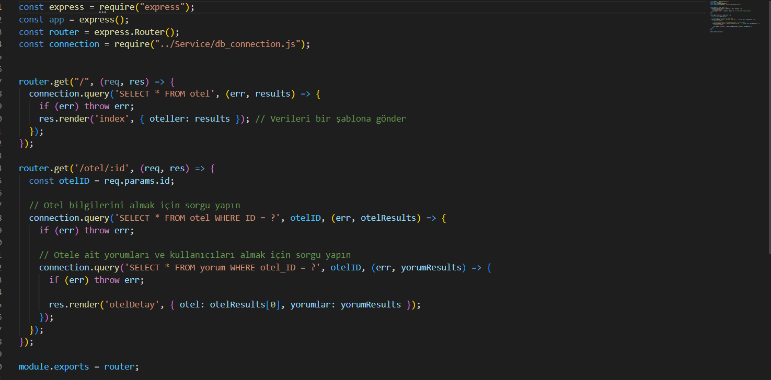
Uygulamamdan bahsetmem gerekirse front end kısmı basit HTML ,CSS yapısından ve ejs dosya formatından arka yüzü için ise javascript, veri tabanı için ise Mysql yapısı kullandım. Kullanıcı yorumları veri tabanında tutulmaktadır. Uygulamanın arka yüzü apiler aracılığı ile veri tabanı ile iletişim kurmaktadır. Veri tabanı yapısını ve uygulama kısmını sanallaştırma ayrı bir ımage ve konteyner yapısına böldüm. Uygulamanın temel amacı ise bilgisi girilen otellerin misafir yorumlarını getirip kullanıcıya oteller hakkında bilgi vermek. Uygulamanın artı ve eksilerini değerlendirecek olursam şu an uygulama oteller hakkında bilgi almak açısından işlevsel olmasının yanı sıra gerekli düzenlemeler ile sanallaştırılması ve yayını tamamlandığında kaynak kullanımı açısından elverişli bir uygulama şu an için ise daha fazla otel bilgisi ilave edilebilir ve detaylandırma yapılarak geliştirilebilir. Piyasadaki benzer uygulamalar bulunmakla beraber bu uygulamanın en büyük artısı sanallaşması ve geliştirilmeye açık olmasıdır.



**Şekil-5.** Veri tabanı kodları

****

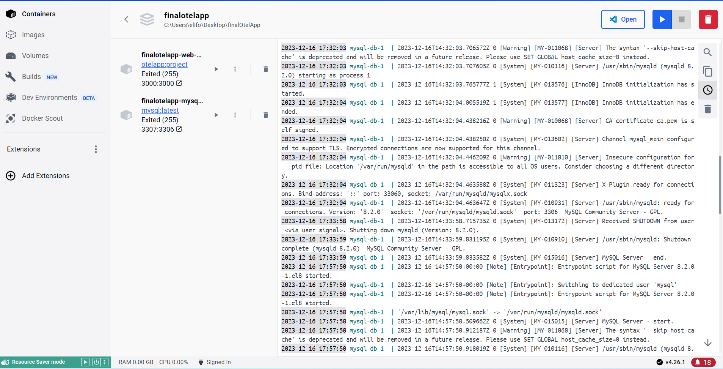
**Şekil-6.** Front end ejs yapısı

****

**Şekil-7.** Back-End api kodlarından bir kısım

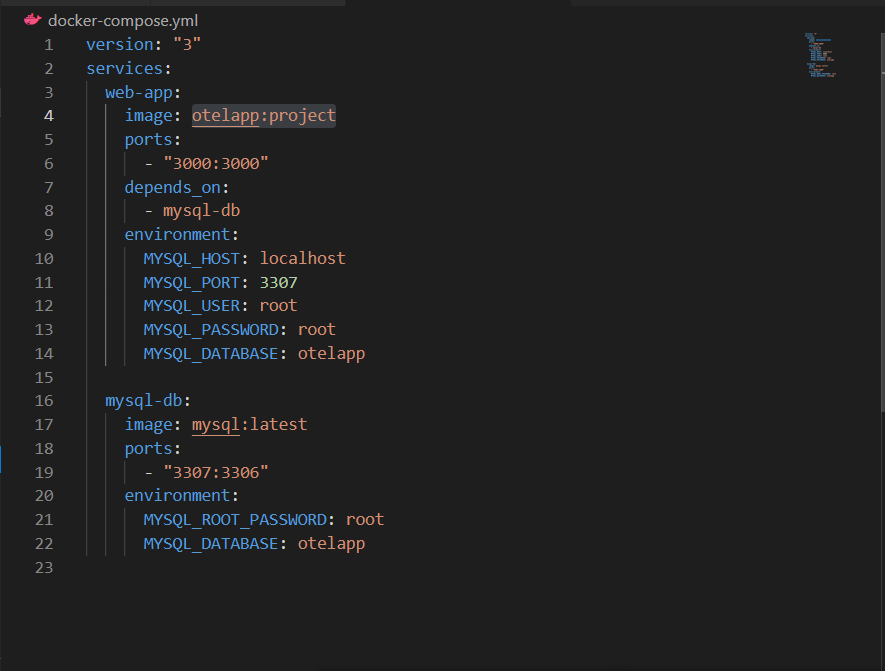
## Uygulamadaki Konteynerlar

Uygulamamı bu şekilde hayata geçirdikten sonra sanallaştırma kısmına başladım. Bu kısımda image yapısını oluşturduktan sonra sonraki aşamalara geçtim ve uygulamamı konteyner yapısı oluşturmaya başladım. Konteyner yapısı hem projem için hem de normal sanallaştırma uygulamaları için önemli bir yeri vardır. Dockerfile dosyası oluşturduktan sonra “docker run” komutunu kullanarak port tanımlamaları ve konteynırın ismi gibi genel ayarları yaptıktan sonra komutu çalıştırıp oluşturmuş oluyoruz.

** Şekil-8**

##### Compose Dosyası

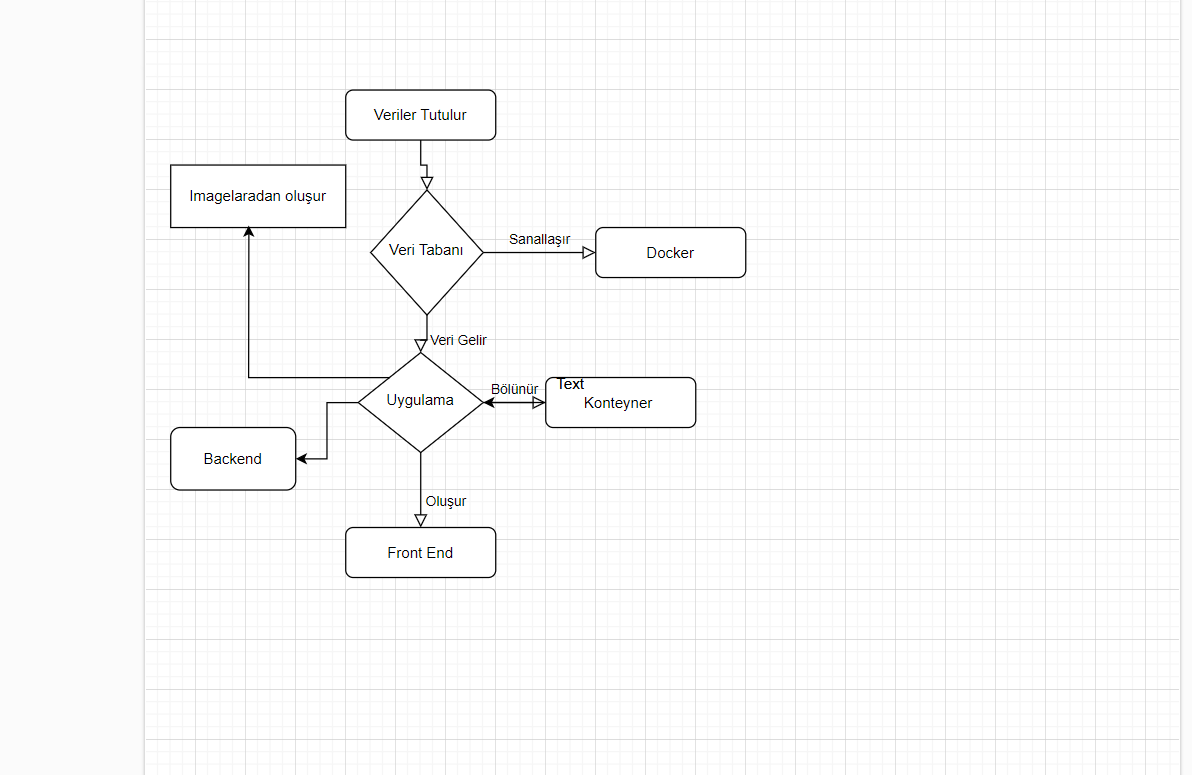
Bu aşamadan sonra .yml uzantılı compose dosyasını oluşturdum. Bu dosyanın asıl amacı projemdeki tanımladığım konteyner yapılar arasında iletişim sağlamak ve gerekli tüm özellikleri sağlayarak çalışmasını sağlamaktır. Fakat ben bu kısımda veri tabanı sürümüm ile alakalı bir sorun yaşadığım için uygulamamı dockerize etme işlemimi tam olarak tamamlayamadım.



**Şekil-11**

# UML Diagramı

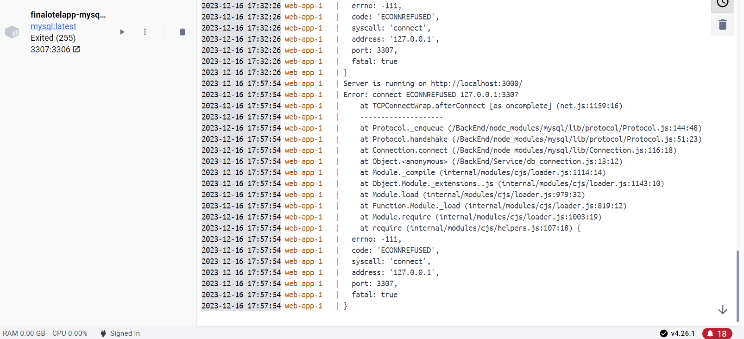
Uygulamanın UML diagramı aşağıda gösterdiğim şekildedir.



**Şekil-9**

## Uygulama Esnasında Karşılaştığım Zorluklar

Bu uygulama sanallaştırma projesi sırasında özellikle sanallaştırmanın ne olduğunu tam olarak bilmediğim için etraflı bir araştırma yapmam gerekti. Daha sonrasında bir uygulamaya back end ve api yazma konusunda bir tecrübem olmadığı için bu kısımla ilgili araştırma yapmam gerekti. Docker’ ı hayatımda ilk kez duyuyordum ve ne işe yaradığını hangi amaçla kullanıldığını ımage almayı bu proje ile öğrendim. AWS üzerinden projemin yayını da yapacaktım ancak gerekli ayarlamaları yapamadığım için bu kısmı gerçekleştiremedim. Bununla birlikte Docker üzerinde kurulum yaparken karşılaştığım en önemli sorun ise konteyner yapılarını birbiri ile haberleştirememek ve veri alış verişi konusunda yaşadığım sıkıntı oldu.



**Şekil-10**

##### Kaynakça

<https://www.youtube.com/watch?v=zOsO996Esck>

<https://bhavan.dev/blog/how-to-deploy-node-js-application-to-aws-ec2-a-step-by-step-guide>

<https://dogukan.dev/nodejs-express-restful-api-gelistirmek>

<https://www.youtube.com/watch?v=4XVfmGE1F_w>

<https://www.youtube.com/watch?v=4G7zjbAKUb4>

<https://www.youtube.com/watch?v=WX0d3tfL_HY&t=223s>

<https://www.youtube.com/watch?v=OPoaiSIOOSQ>

<https://www.youtube.com/watch?v=HFEOJ7wgd0M>

<https://www.youtube.com/watch?v=Vlw8Fo7f_6k>

##### Proje Linkleri

Projemin gerekli linkleri aşağıdadır:

Drive Linki: <https://drive.google.com/drive/folders/1aKpfChPAW-yl532o-apXPEKkQfxwjVya?usp=sharing>

Github Linki: <https://github.com/elif-ozkan/OtelBurdaApp>