T.C. Mudanya Üniversitesi
Mikroservis Tasarımı ve Geliştirme Dersi
Final Araştırma Ödevi

Okul No:	234001077
Ad:	Oktay
Soyad:	AYDOĞAN

1. Amaç

Bu belge, BMB 529 – Mikroservis Tasarımı ve Geliştirme dersi, final sınavı dahilinde olan, sınavın geliştirilmesinde yardımcı olan ek teknoloji hakkında bilgileri sunmaktır. Araştırmanın detayları, final sınavı proje klasöründe bulunan HELP.md dosyasında bulunmaktadır.

2. Genel Bakış

Bu belgede, Spring Boot uygulaması ve Docker container entegrasyonunun genel bir bakışı sunulacaktır. Docker Compose ve "**spring-boot-docker-compose**" dependency'sinin rolü, uygulamanın başlatılmasında ve yönetiminde nasıl kullanıldığı açıklanacaktır. Spring Boot ve Docker'ın birlikte nasıl çalıştığını ve bu entegrasyonun geliştirme ve test süreçlerinde nasıl kolaylık sağladığını ele alacağız.

3. Spring Boot Uygulaması ve Docker Container Entegrasyonu

3.1. Dependency Tanıtımı

Spring Boot uygulamasında kullanılan **"spring-boot-docker-compose"** dependency'si, Docker container'larının Spring Boot uygulamaları ile kolayca yönetilmesini sağlar. Bu dependency, uygulama başlatıldığında Docker container'larının da otomatik olarak başlatılmasını ve gerektiğinde durdurulmasını destekler.

```
dependencies { Edit Starters...
    developmentOnly 'org.springframework.boot:spring-boot-docker-compose'
}
```

Bu dependency, geliştirme sürecinde kullanılır ve Docker Compose entegrasyonunu sağlar. Bu sayede, uygulama başlatıldığında gerekli tüm container'lar da otomatik olarak başlatılır.

3.2. Docker Compose Nedir?

Docker Compose, birden fazla Docker container'ını tanımlamak ve çalıştırmak için kullanılan bir araçtır. Docker Compose kullanarak, uygulamanızın tüm servislerini tek bir YAML dosyasında tanımlayabilir ve bu servisleri tek bir komutla başlatabilirsiniz. Bu, özellikle mikroservis mimarilerinde ve geliştirme/test ortamlarında büyük kolaylık sağlar.

3.3. Compose.yaml Dosyası

compose.yaml dosyası, Docker Compose tarafından kullanılacak container'ların yapılandırmasını içerir.

Örneğin:

```yaml
services:
  postgres:
    image: 'postgres:latest'
    environment:
      - 'POSTGRES_DB=users'
      - 'POSTGRES_PASSWORD=postgres'
      - 'POSTGRES_USER=postgres'
    ports:
      - "5430:5432"
```

Bu dosyada:

- **version**: Docker Compose dosya formatının versiyonunu belirtir.
- **services**: Başlatılacak container'ların tanımlandığı bölümdür.
    - **postgres**: PostgreSQL container'ı için yapılandırma. Bu container, uygulamanın ihtiyaç duyduğu veritabanı servislerini sağlar.

environment anahtarı, PostgreSQL container'ının başlatılmasında kullanılacak çevresel değişkenleri tanımlar:

- **POSTGRES_DB**: Oluşturulacak veritabanının adı.
- **POSTGRES_PASSWORD**: Veritabanı kullanıcısının şifresi.
- **POSTGRES_USER**: Veritabanı kullanıcısının adı.

ports anahtarı, PostgreSQL container'ının dış dünyaya hangi port üzerinden erişileceğini belirtir. Bu örnekte, container'ın içindeki 5432 portu, dış dünyada 5430 portuna yönlendirilmiştir.

3.4. Spring Boot Uygulamasının Başlatılması

Spring Boot uygulaması başlatıldığında, org.springframework.boot:spring-boot-docker-compose dependency'si, compose.yaml dosyasını kullanarak tanımlı Docker container'larını başlatır. Bu süreç şu adımlarla gerçekleşir:

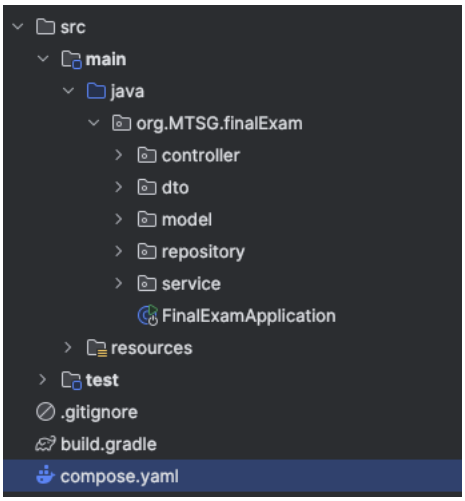
1. **Uygulama Başlatma**: Spring Boot uygulaması, main metodu çalıştırıldığında başlatılır.
2. **Docker Compose Entegrasyonu**: spring-boot-docker-compose dependency'si, uygulama başlatma sürecinde compose.yaml dosyasını okur.
3. **Container'ların Başlatılması**: Docker Compose, compose.yaml dosyasındaki tanımlamalara göre gerekli container'ları başlatır.
4. **Servislerin Bağlanması**: Uygulama, başlatılan container'larla iletişim kurarak çalışmaya başlar.

3.5. Uygulamanın Çalışma Süreci

- **Geliştirme Ortamı**: Geliştirme sırasında, Docker Compose tüm bağımlı servisleri (örneğin, veritabanı, cache, mesaj kuyrukları) başlatarak geliştirme ortamının hazır olmasını sağlar.
- **Test Ortamı**: Otomatik testler sırasında, Docker Compose gerekli servisleri başlatarak testlerin doğru bir şekilde çalışmasını sağlar.
- **Bağımlılık Yönetimi**: Docker Compose, servislerin başlatılma sırasını yönetir ve bağımlılıklar doğru şekilde sağlanır.

3.6. Örnek Proje Yapısı

Bir Spring Boot projesinde Docker Compose entegrasyonu için proje yapısı şu şekilde olabilir:



4. Kaynaklar

- **Spring Boot Documentation** - *https://spring.io/projects/spring-boot*
- **Docker Documentation** - *https://docs.docker.com/compose/*
- **Spring Boot Docker Compose Support** - *https://docs.spring.io/spring-boot/reference/features/dev-services.html#features.dev-services.docker-compose*