T.C.

BİLECİK ŞEYH EDEBALİ ÜNİVERSİTESİ

MÜHENDİSLİK FAKÜLTESİ

BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ

Docker ile çoklu konteynerlar kullanarak paralel iş yapan sistem tasarımı

Aleyna ÇELİK

IBRAHIM KHALIL ATTEIB YACOUB

BİTİRME ÇALIŞMASI

DANIŞMANI : Prof. Dr. Cihan KARAKUZU

Dr. Öğr. Üyesi Burakhan ÇUBUKÇU

BİLECİK

13 Mayıs 2023

T.C.

BİLECİK ŞEYH EDEBALİ ÜNİVERSİTESİ

MÜHENDİSLİK FAKÜLTESİ

BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ

Docker ile çoklu konteynerlar kullanarak paralel iş yapan sistem tasarımı

Aleyna ÇELİK

Ibrahim Khalil Atteib YACOUB

BİTİRME ÇALIŞMASI

DANIŞMANI : Prof. Dr. Cihan KARAKUZU

Dr. Öğr. Üyesi Burakhan ÇUBUKÇU

BİLECİK

13 Mayıs 2023

BİLDİRİM

Bu çalışmada bütün bilgilerin etik davranış ve akademik kurallar çerçevesinde elde edildiğini ve yazım kurallarına uygun olarak hazırlanan bu çalışmada bana ait olmayan her türlü ifade ve bilginin kaynağına eksiksiz atıf yapıldığını bildiririm.

DECLARATION

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all materials and results that are not original to this work.

İmza

Aleyna ÇELİK

Ibrahim Khalil Atteib YACOUB

Tarih: 13 Mayıs 2023

ÖZET

BİTİRME ÇALIŞMASI

Docker ile çoklu konteynerlar kullanarak paralel iş yapan sistem tasarımı

Aleyna ÇELİK

Ibrahim Khalil Atteib YACOUB

Bilecik Şeyh Edebali Üniversitesi
Mühendislik Fakültesi
Bilgisayar Mühendisliği Bölümü

Danışman: Prof. Dr. Cihan KARAKUZU

Dr. Öğr. Üyesi Burakhan ÇUBUKÇU

2023, 25 Sayfa

Jüri Üyeleri	**İmza**
. . . . .	. . . . .
. . . . .	. . . . .
. . . . .	. . . . .

Docker ile çoklu konteynerlar kullanarak paralel işlem yapan sistemler, uygulama ve servislerin farklı konteynerlar içinde izole edilerek çalıştırılmasını sağlayan bir sistem tasarımıdır. Bu sayede her bir konteyner, farklı özellikler ve işlevler için optimize edilebilir ve uygulama ölçeklenebilir hale gelir. Ayrıca, verilen işi belirtilen sayıda konteyner ile bölerek daha esnek hale getirmek amaçlanmıştır. Arayüz, canlı olarak ekranda ilerleme bilgileri göstererek, kullanıcılara toplam iş miktarı ile tamamlanma yüzdesi ve hangi konteynerların en hızlı olduğu gibi önemli bilgileri verir. Bu arayüz, güvenlik testleri veya izinsiz giriş girişimi gibi meşru olmayan amaçlarla kullanılmamalıdır.

ABSTRACT

THESIS

System design using Docker with multiple containers for parallel processing

Aleyna ÇELİK

Ibrahim Khalil Atteib YACOUB

Bilecik Şeyh Edebali University
Engineering Faculty
Department of Computer Engineering

Advisor : Prof. Dr. Cihan KARAKUZU

Assoc. Prof. Dr. Burakhan ÇUBUKÇU

2023, 25 Pages

Jury	Sign
. . . . .	. . . . .
. . . . .	. . . . .
. . . . .	. . . . .

Using multiple containers with Docker to perform parallel processing is a system design that allows applications and services to be run in different containers, isolated from each other. This allows each container to be optimized for different features and functions, making the application scalable. Additionally, the system is designed to make the given task more flexible by dividing it into a specified number of containers. The interface displays progress information live on the screen, providing users with important information such as the total amount of work completed, the completion percentage, and which containers are the fastest. However, this interface should not be used for illegitimate purposes such as security testing or unauthorized access attempts.

ÖNSÖZ

Bitirme çalışmasında başından sonuna kadar emeği geçen ve bizi bu konuya yönlendiren saygı değer hocalarımız ve danışmanlarımız Sayın Prof. Dr. Cihan KARAKUZU ve Dr. Öğr. Üyesi Burakhan ÇUBUKÇU'a tüm katkılarından ve hiç eksiltmediği desteklerinden dolayı teşekkür ederiz.

Aleyna ÇELİK

Ibrahim Khalil Atteib YACOUB

13 Mayıs 2023

İÇİNDEKİLER

ÖNSÖZ **v**

ŞEKİLLER TABLOSU **vii**

1 GİRİŞ **1**

2 PROJE TANITIMI **4**

3 KULLANILAN YAZILIMLAR VE YÖNTEMLER **6**

3.1 Web Uygulama Geliştirmek İçin 6

3.1.1 Laravel 6

3.2 Konteynerize Etmek İçin 7

3.2.1 Docker 7

3.3 Veri Tabanı İçin 9

3.3.1 MYSQL 9

4 PROJE'NİN İŞLEYİŞİ **10**

4.1 Arayüz Kısmı 10

4.2 Arka paln işleyişi 10

4.2.1 Docker Engine API 11

4.2.2 Laravel Özellikleri 11

4.3 Veritabanı kısmı İçin 12

5 SONUÇLAR VE ÖNERİLER **14**

6 EKLER **15**

KAYNAKLAR **16**

ÖZGEÇMİŞ **17**

ŞEKİLLER TABLOSU

Şekil 4.1 Sql ping tabloları 13

1 GİRİŞ

Paralel işlem, modern bilgisayar sistemlerinde performansı artırmak için yaygın olarak kullanılan bir tekniktir. Ancak, bu teknik, karmaşık paralel işlem yapısı tasarımı ve uygulanması zor ve zaman alıcı olabilir. Bu sorunları çözmek için Docker, çoklu konteynerlar kullanarak paralel işlem yapmak için ideal bir çözüm sunar. Docker, 2013 yılında dotCloud adlı bir Platform as a Service (PaaS) sağlayıcısı tarafından geliştirilmiştir ve kurucusu ve CTO'su Solomon Hykes olarak bilinmektedir.

Docker, uygulamaların ve hizmetlerin paketlenmesi, dağıtımı ve çalıştırılmasını kolaylaştıran bir yazılım platformudur. Bu teknoloji, farklı konteynerlar arasında görevleri bölüştürerek, daha hızlı ve verimli bir şekilde çalışan bir sistem oluşturmak için kullanılabilir. Çoklu konteynerlar, birbirleriyle iletişim halinde olan ve her biri farklı bir görevi yerine getiren bir dizi konteynerden oluşan bir yapıdır. Bu yapı, görevler arasında paralelleştirme yapılarak daha hızlı ve verimli bir sistem oluşturulabilir. Konteynerler arasındaki iletişim, Docker Swarm veya Kubernetes gibi konteyner orkestrasyon araçları kullanılarak kolayca yönetilebilir. Bu sayede, tasarım ölçeklenebilir hale getirilebilir.

Bu çalışmada, Laravel PHP kullanılarak bir örnek üzerinde Docker ile çoklu konteynerlar kullanarak paralel işlem yapmanın nasıl oluşturulacağına ve yönetileceğine dair detaylı bilgiler verilecektir. Örneğin veritabanı olarak MySQL kullanılacak. Docker sayesinde, örneğin birden fazla sunucuda birden fazla örneği çalıştırmak mümkün olacak. Böylece, örneğin daha yüksek trafik hacimlerine veya daha karmaşık iş yüklerine ihtiyaç duyulduğunda sistem ölçeklendirilebilir hale getirilebilir.

Docker ayrıca, geliştirme sürecini de hızlandırabilir. Örneğin, farklı uygulama ve hizmetleri ayrı ayrı test etmek yerine, Docker sayesinde bunları farklı konteynerlar içinde birleştirmek ve birlikte test etmek mümkün olacak. Bu sayede, uygulama ve hizmetlerin bütünlüğü ve performansı daha iyi bir şekilde test edilebilir. Ayrıca, Docker sayesinde, uygulama ve hizmetlerin herhangi bir sistemde kolayca çalıştırılması ve dağıtılması da mümkün hale gelir.

Sonuç olarak, Docker çoklu konteynerlar kullanarak paralel işlem yapmak için ideal

bir züm sunan bir teknolojidir. Bu sayede, karmaşık paralel işlem yapısı tasarımı ve uygulanması daha kolay ve zaman açısından daha verimli hale gelir.

Docker'ın çoklu konteynerlar kullanarak paralel işlem yapmaya sağladığı avantajlardan biri, iş yükünü parçalara bölerek her bir parçayı ayrı bir konteyner içinde çalıştırabilme esnekliğidir. Bu, işlemleri paralel olarak gerçekleştirerek performansı artırır. Örneğin, bir web uygulaması üzerinde çalışırken, kullanıcı taleplerini işleyen bir konteyner, veritabanı işlemlerini yapan başka bir konteyner ve resim işleme gibi farklı bir görevi üstlenen bir başka konteyner olabilir. Bu konteynerler, bağımsız olarak çalışabilir ve birbirleriyle iletişim kurabilirler. Bu şekilde, her bir konteyner kendi kaynaklarını etkin bir şekilde kullanarak iş yükünü hızlı ve verimli bir şekilde işleyebilir.

Docker ayrıca, ölçeklendirme kolaylığı sağlar. Yüksek talep veya büyüyen iş yükleri durumunda, Docker Swarm veya Kubernetes gibi konteyner orkestrasyon araçları kullanarak yeni konteyner örnekleri oluşturabilir ve bu konteynerleri mevcut konteyner gruplarına ekleyebilirsiniz. Bu, daha fazla kaynak sağlayarak sistemi ölçeklendirmenize ve iş yükünü dengelemenize yardımcı olur. Aynı şekilde, talep düştüğünde veya iş yükü azaldığında, gereksiz konteyner örneklerini kolayca kaldırabilirsiniz. Bu dinamik ölçeklendirme yeteneği, kaynak kullanımını optimize ederek performansı artırır ve maliyetleri düşürür.

Docker ayrıca, geliştirme sürecini kolaylaştırır ve hızlandırır. Konteyner tabanlı bir geliştirme ortamında, her bir bileşen veya hizmeti ayrı bir konteynerde çalıştırarak, geliştiricilerin uygulama ve hizmetlerin birlikte çalışmasını ve entegrasyonunu daha iyi test etmelerini sağlar. Ayrıca, her bir konteyneri bağımsız olarak güncelleyebilir veya değiştirebilirsiniz, bu da hızlı ve sorunsuz bir şekilde yeni özellikler eklemenize veya hataları düzeltmenize olanak tanır.

Sonuç olarak, Docker'ın çoklu konteynerlar kullanarak paralel işlem yapmaya yönelik sunmuş olduğu çözüm, performansı artırır, ölçeklenebilirlik sağlar, geliştirme sürecini hızlandırır ve sistem yönetimini kolaylaştırır. Docker, paralel işlem ihtiyaç iyaç duyan projeler için ideal bir çözümdür. Özellikle büyük ölçekli uygulamalar, mikro hizmet mimarileri veya veri yoğun iş yükleriyle çalışan sistemler Docker'ın sağladığı çoklu kontey-

nerlar ile paralel işlem yapma yeteneklerinden büyük ölçüde faydalanabilir.

Docker'ın çoklu konteynerlar kullanarak paralel işlem yapmanın yanı sıra birçok avantajı vardır. Bunlardan biri, uygulama ve hizmetlerin bağımsız ve izole bir şekilde çalıştırılabilmesidir. Her bir konteyner, kendi ortamını ve bağımsız olarak çalışan bileşenlerini içerir. Bu, uygulama ve hizmetlerin birbirinden etkilenmeden güvenli bir şekilde çalışmasını sağlar.

Ayrıca, Docker'ın konteyner tabanlı mimarisi, uygulama ve hizmetlerin farklı ortamlarda sorunsuz bir şekilde çalışmasını sağlar. Bir konteyneri bir sistemden diğerine taşımak veya farklı bir ortamda çalıştırmak oldukça kolaydır. Bu da geliştirme sürecini hızlandırır ve dağıtım süreçlerini kolaylaştırır.

Docker ayrıca, kaynakların etkin bir şekilde kullanılmasını sağlar. Her bir konteyner, sadece ihtiyaç duyduğu kaynakları kullanır ve izole edilmiş olduğu için diğer konteynerlere etki etmez. Bu sayede, sistemdeki kaynakların verimli bir şekilde dağıtılması ve kullanılması sağlanır.

Son olarak, Docker ekosistemi geniş bir topluluk tarafından desteklenmektedir. Docker Hub gibi bir merkezi depo, hazır konteyner görüntülerini paylaşmanıza ve kullanmanıza olanak tanır. Ayrıca, Docker, zengin bir API ve komut satırı arayüzü ile birlikte gelir, bu da otomasyon ve yönetim süreçlerini kolaylaştırır.

Docker'ın çoklu konteynerlar kullanarak paralel işlem yapma yetenekleri, modern bilgisayar sistemlerinde performansı artırmak ve verimliliği sağlamak için güçlü bir araçtır. Karmaşık iş yüklerini parçalara ayırarak, her bir parçayı ayrı bir konteynerde çalıştırarak ve konteynerler arasında iletişimi kolayca yöneterek, daha hızlı, ölçeklenebilir ve etkili bir sistem oluşturmak mümkündür. Bu nedenle, Docker, paralel işlem yapmak isteyen geliştiriciler ve sistem yöneticileri için önemli bir teknoloji ve çözüm sunmaktadır.

2 PROJE TANITIMI

Projemizin amacı, Docker kullanarak paralel işlem yapabilen sistemler, uygulamalar ve servisler için çoklu konteynerlar kullanarak bir sistem tasarlamaktır. Docker, izolasyon sağlayarak her bir konteyneri farklı özellikler ve işlevler için optimize etme imkanı sunar ve bu da uygulamanın ölçeklenebilir olmasını sağlar.

Projede, Laravel yapısı kullanılıdI. Bu sayede, arayüz ve backend arasında esneklik sağlayarak birlikte çalışma imkanı elde edildi. Arayüz için başlangıçta bir hazır template kullanıldı, ancak bu templateleri ihtiyaçlarımıza göre özelleştirildi ve değişiklikler yapıldı. Daha sonra backend ile entegrasyonu gerçekleştirerek ayarlamaları tamamlandı.

Projenin güvenliği için, login sayfasını tasarlandı ve erişim kısıtlamaları eklendi. Bu sayede, giriş yapmayan kullanıcıların diğer sayfalara erişmesi engellendi. Kullanıcıların güvenliğini ve verilerin gizliliğini korumak önemli bir hedefimiz oldu.
Son aşamada, Docker işlemlerini gerçekleştirerek konteynerleme işlemleri tamamlandı. Docker'ın avantajlarından yararlanarak, uygulamayı farklı konteynerlar içinde çalıştırılabildi. Böylelikle, her bir konteynerin optimize edilmiş bir şekilde çalışmasını sağlayarak işleri belirtilen sayıda konteynera bölerek daha esnek bir yapı oluşturuldu. Bu da performansı artırırken sistem üzerindeki yükü dengelememize yardımcı oldu.
Projenin arayüzünde, canlı olarak ilerleme bilgilerini gösteren bir özellik bulunmaktadır. Kullanıcılara, toplam iş miktarı, tamamlanma yüzdesi ve en hızlı çalışan konteynerlar gibi önemli bilgiler sunarak işlerin durumunu takip etmeleri sağlandı. Bu sayede kullanıcılar, işlerin ne kadar ilerlediğini ve hangi konteynerların en etkili olduğunu görebilmektedir.
Özetlemek gerekirse, projemizde Docker kullanarak paralel işlem yapabilen bir sistem tasarlandı. Laravel yapısını kullanarak esnek bir yapı oluşturuldu. Güvenlik önlemlerini alarak kullanıcıların ve verilerin güvenliğini sağlandı. Docker ile konteynerleme işlemlerini gerçekleştirerek uygulamayı optimize ettik ve performansı artırıldı. Arayüzde canlı ilerleme bilgilerini gösteren bir özellik ekleyerek kullanıcıların işlerin durumunu takip etmelerini kolaylaştırıldı.

Projemizin başarıyla tamamlanması için bir dizi adım takip edildi. İlk olarak, gereksinimlerimizi belirlendi ve Docker'ı seçtik çünkü konteynerleme teknolojisi olarak bize izolasyon ve esneklik sağlayan bir çözüm sunuyordu. Ardından, Laravel yapısını kullanarak arayüz ve backend arasında etkili bir iletişim ve veri akışı sağlandı.

Arayüz tarafında, hazır bir template kullanarak başlandı, ancak bunu kendi tasarım ve işlevsel gereksinimlerimize uygun şekilde özelleştirildi. Kullanıcılarımızın deneyimini en üst düzeye çıkarmak için kullanıcı dostu ve sezgisel bir arayüz tasarlandı.

Backend tarafında, güvenlik önlemleri almak için login sayfasını tasarlandı ve kullanıcıların kimlik doğrulama sürecini sağlamlaştırıldı. Bu sayede, yalnızca yetkilendirilmiş kullanıcıların sistemimizi kullanabilmesi sağlandı ve veri güvenliğini sağlandı.

Docker kullanarak konteynerleme işlemleri gerçekleştirildi. Her bir konteyner, farklı özelliklere ve işlevlere sahip olduğu için optimize edilebilir hale geldi. İşleri parçalara bölerek belirli sayıda konteynera dağıttık, bu da sistemimizin esnekliğini artırdı ve işleri daha hızlı ve etkin bir şekilde gerçekleştirebildi.

Projenin devamında, kullanıcıların işleri daha etkili bir şekilde yönetmelerini sağlamak için bazı ek özellikler eklendi. Örneğin, kullanıcılar işleri önceliklendirebilir ve farklı konteynerlarda çalışacak işlere öncelik atayabildi. Bu, daha hızlı tamamlanması gereken işlerin öncelikli olarak çalıştırılmasını sağladı ve genel performansı artırdı.

Ayrıca, proje takımının iletişimini kolaylaştırmak için bir bildirim sistemi entegre edildi. Kullanıcılar, tamamlanan işler hakkında anlık bildirimler aldı ve gerektiğinde müdahale edebildi. Bu, ekip üyelerinin işlerin durumunu takip etmesini ve projenin ilerlemesini koordine etmesini kolaylaştırdı.

Projenin güvenliği için, izinsiz erişim girişimlerini engellemek için oturum açma sayfasına ek güvenlik önlemleri eklendi. Kullanıcıların güvenliği ve verilerin korunması en üst düzeyde önem taşırken, sistemimize giriş yapmak isteyen herhangi bir yetkisiz kişinin engellenmesi sağlandı.

Performansı ve ölçeklenebilirliği daha da artırmak için, proje üzerinde sürekli iyileştirmeler ve optimizasyonlar gerçekleştirildi. Örneğin, konteynerler arasında yük dengelemesi yaparak işlerin daha dengeli bir şekilde dağıtılmasını sağlandı ve böylece sistem üzerindeki aşırı yüklenmeyi önlenildi.

Arayüzde canlı ilerleme bilgilerini gösteren bir özellik eklendi. Kullanıcılara toplam iş

miktarını, tamamlanma yüzdesini ve en hızlı çalışan konteynerları göstererek işlerin durumunu takip etmelerini sağlandı. Bu, kullanıcıların proje ilerlemesini daha iyi yönetmelerini ve gerekirse müdahalede bulunmalarını sağlayan değerli bir geribildirim mekanizması sağlandı.
Son olarak, proje yönetimi için bir kontrol paneli eklendi. Bu panel, proje yöneticilerinin işlerin ilerlemesini takip etmelerini, konteyner performansını analiz etmelerini ve gerektiğinde müdahale etmelerini sağlandı. Kontrol paneli, genel bir bakış açısı sunarak proje yönetiminin daha etkin bir şekilde yapılmasına yardımcı oldu.
Tüm bu adımları takip ederek, Docker kullanarak paralel işlem yapabilen bir sistem tasarlandı. Projemiz, Laravel yapısıyla esneklik sağladı, güvenliği ön planda tuttu, konteynerleme ile performansı artırdı ve kullanıcılarımıza canlı ilerleme bilgileri sunarak daha iyi bir deneyim sağladı.

3 KULLANILAN YAZILIMLAR VE YÖNTEMLER

3.1 Web Uygulama Geliştirmek İçin

3.1.1 Laravel

Laravel, popüler bir PHP framework'üdür ve MVC (Model-View-Controller) mimarisini kullanır. Bu mimari, uygulamanın farklı katmanlarını ayırarak kodun düzenli, okunabilir ve bakımı kolay hale gelmesini sağlar. Laravel, model, view, controller ve route gibi temel bileşenlerden oluşur. Model, veri katmanını temsil ederken, view kullanıcı arayüzünü ve controller iş mantığını yönetir. Route ise istekleri doğru controller'a yönlendirir.

Laravel'in MVC yapısı, kodun düzenli tutulmasını, bileşenlerin bağımsız olmasını ve geliştirme sürecinin daha etkili olmasını sağlar. Bu sayede uygulama daha yeniden kullanılabilir, test edilebilir ve bakımı kolay hale gelir.

3.2 Konteynerize Etmek İçin

3.2.1 Docker

Docker, uygulamaların ve hizmetlerin paketlenmesi, dağıtımı ve çalıştırılmasını kolaylaştıran bir yazılım platformudur. Docker, konteyner teknolojisi kullanarak uygulamaların izole edilmiş ve taşınabilir bir şekilde çalışmasını sağlar.

Docker'ın çalışma yapısı şu şekildedir:

- Konteyner: Docker'ın temel yapı birimidir. Konteyner, uygulamanın tüm bağımlılıklarını (kod, çalışma zamanı, kütüphaneler, ortam değişkenleri vb.) bir araya getirir ve izole bir ortamda çalışmasını sağlar. Konteynerlar, bir uygulamayı çalıştırmak için gerekli olan tüm bileşenleri içerir ve bu sayede uygulamaları farklı ortamlarda tutarlı bir şekilde çalıştırabilirsiniz.
- Docker Image (Docker İmajı): Bir Docker konteynerini oluşturmak için kullanılan bir şablondur. Bir Docker imajı, bir veya daha fazla katman (layer) olarak adlandırılan yapı taşlarından oluşur. Her katman, imajın farklı bir bileşenini veya yapılandırmasını temsil eder. Bir Docker imajı, bir veya birden fazla konteyneri oluşturmak için kullanılabilir.
- Dockerfile (Docker Dosyası): Docker imajlarının nasıl oluşturulacağını tanımlayan bir metin dosyasıdır. Dockerfile, bir uygulamanın çalıştırılması için gerekli olan adımları ve komutları belirtir. Dockerfile, uygulama kodunu, çalışma zamanını, bağımlılıkları ve diğer yapılandırmaları imaja dahil etmek için kullanılır.
- Docker Registry: Docker imajlarının depolandığı ve paylaşıldığı bir merkezi kaynak. Docker Hub, en popüler ve yaygın kullanılan Docker Registry'dir. Docker Hub'da, birçok hazır Docker imajı bulabilir ve kendi imajlarınızı da paylaşabilirsiniz. Ayrıca, Docker Registry'lerini yerel olarak da kurabilir ve kullanabilirsiniz.

Docker'ın avantajları şunlardır:

- Taşınabilirlik: Docker, uygulamaların farklı platformlarda ve ortamlarda sorunsuz bir şekilde çalışmasını sağlar. Bir kez oluşturulan Docker imajı, farklı sistemlerde

kolayca dağıtılabilir ve çalıştırılabilir.

- İzolasyon: Docker konteynerleri, uygulamaları ve bağımlılıklarını izole ederek, bir konteynerin diğerlerine etkileşimde bulunmasını engeller. Bu, güvenlik ve istikrar açısından önemli bir avantaj sağlar.
- Hızlı dağıtım ve ölçeklendirme: Docker, uygulamaların hızlı bir şekilde dağıtılmasını ve ölçeklendirilmesini sağlar. İmajların hızlı bir şekilde oluşturulması ve konteynerlerin hızlı bir şekilde başlatılması sayesinde uygulamaların hızlıca yayınlanması mümkün olur.
- Verimlilik: Docker, kaynakların etkin bir şekilde kullanılmasını sağlar. Konteynerler, daha az bellek ve işlemci gücü tüketerek daha fazla uygulamanın çalışmasını sağlar.
- Veri bütünlüğü: Docker, verilerin konteynerlerle birlikte taşınmasını ve yönetilmesini kolaylaştırır. Konteynerler, verileri izole bir şekilde tutar ve bu da veri bütünlüğünü sağlar.
- Hızlı geri alma ve güncelleme: Docker, imaj ve konteyner tabanlı bir yaklaşım kullanarak hızlı geri alma ve güncelleme süreçleri sağlar. Eski bir imaja geri dönerek veya güncellenmiş bir imajla yeni bir konteyner oluşturarak sistemde yapılan değişiklikleri geri almak veya güncellemek kolaydır.
- Kaynak verimliliği: Docker, paylaşılan çekirdek özelliklerini kullanarak kaynakların daha verimli bir şekilde kullanılmasını sağlar. Birçok konteyner, tek bir işletim sistemi çekirdeği üzerinde çalışır ve bu da sistem kaynaklarının daha etkin bir şekilde kullanılmasını sağlar.
- Topluluk desteği: Docker, geniş bir kullanıcı ve geliştirici topluluğuna sahiptir. Bu topluluk, Docker hakkında bilgi paylaşımı, sorun giderme ve yenilikçi çözümler sunma konusunda yardımcı olur. Docker Hub gibi kaynaklar, hazır kullanıma uygun imajlar ve araçlar sağlar.

Docker'ın bu avantajları, uygulama geliştirme ve dağıtım süreçlerinde hızlılık, esneklik, güvenlik ve ölçeklenebilirlik sağlar. Ayrıca, Docker'ın genişletilebilirlik ve entegrasyon yetenekleri, farklı platformlar ve araçlarla uyumlu çalışmayı kolaylaştırır.

Neden Docker ?

3.3 Veri Tabanı İçin

3.3.1 MYSQL

MySQL, açık kaynaklı bir ilişkisel veritabanı yönetim sistemidir. İlişkisel veritabanı yönetim sistemleri, verileri tablolarda saklayarak veriler arasındaki ilişkileri yönetmeyi sağlar. MySQL, istemci-sunucu modeline dayalı olarak çalışır ve bir sunucu üzerinde çalışan çok kullanıcılı bir veritabanı sistemidir.

MySQL, hızlı, güvenilir ve geniş bir kullanıcı tabanına sahip olmasıyla bilinir. Veritabanı yönetimi, veri manipülasyonu, sorgulama, güvenlik, yedekleme ve geri yükleme gibi birçok temel veritabanı işlemini destekler. Ayrıca, MySQL'in geniş bir uyumluluk listesi vardır ve farklı programlama dilleri ve platformlarla kolayca entegre olabilir.

MySQL, birçok web uygulamasında ve büyük ölçekli sistemlerde kullanılan yaygın bir veritabanı çözümüdür. Veri bütünlüğünü koruma, veritabanı güvenliği, yüksek performans, ölçeklenebilirlik ve kullanıcı dostu bir arayüz gibi önemli özelliklere sahiptir.

MySQL, veri tabanı yönetimi için güçlü ve esnek bir seçenek olup, çeşitli projelerde verilerin etkin bir şekilde saklanmasını ve yönetilmesini sağlar.

4 PROJE'NİN İŞLEYİŞİ

4.1 Arayüz Kısmı

Bu projede arayüz kısmı için belirtilen işleyiş aşağıda açıklanmaktadır. Uygun bir template bulunmuş ve projeye uygun hale getirilmiştir. Gereksiz kısımlar temizlenerek sağ menü bölümünde Dashboard, görevler (Ping, Theharvester), sistem logları ve Profil bölümleri oluşturulmuştur.

Projeyi ilk açtığınızda, kullanıcı login sayfasıyla karşılaşacaksınız. Kimlik bilgilerinizi girdikten sonra doğrulama yapıp diğer sayfalara erişebileceksiniz. Giriş yaptıktan sonra karşılaşacağınız ilk sayfa Kontrol Paneli olacaktır. Kontrol panelinde gerçekleştirilebilecek işlemler açıklanacaktır.

Sağ kısımda menü bulunmaktadır. Buradan istediğiniz zaman görevler bölümünden görev oluşturabilir ve önceden oluşturulan görevle ilgili işlemleri görebilirsiniz. Bir Ping görevi oluşturulduğunda, bu konteynırların çıktıları, kayıp yüzdesi, pingin maksimum geri dönüş süresi, minimum geri dönüş süresi ve ortalama milisaniye geri dönüş süresi gibi bilgilere erişebilirsiniz. Ayrıca sağ menüde sistem log kayıtlarını görüntüleyebilirsiniz. Log kayıtlarının altında kullanıcının hangi görevi ne zaman çalıştırdığı, görevin tamamlanma süresi, başarılı veya başarısız olma durumu ile oturum açma ve kapatma bilgileri kolaylıkla görüntülenebilir.

Hesabım bölümü, profili ile ilgili düzenlemeler yapmayı sağlar. Bu bölümde profil bilgilerini güncelleyebilir, şifreniyi değiştirebilir, iletişim tercihlerini yönetebilir ve gizlilik ayarlarını düzenlenebilir. Bu şekilde, profili yönetebilir ve hesap istediğimiz şekilde kişiselleştirebilir hale getirildi.

4.2 Arka paln işleyişi

Projenin alt yapısı, PHP framework'ü olan Laravel kullanılarak geliştirilmiştir. Projenin çalışması için bir docker-compose.yml dosyası kullanılarak Docker'da bir servis oluşturulmuştur. Bu servis, iki konteyner oluşturarak projenin çalışmasını sağlamaktadır. Docker yönetimi için Docker Engine API kullanılmıştır.

4.2.1 Docker Engine API

Docker, Docker Engine API olarak adlandırılan arka plan programı sayesinde etkileşim için bir API sağlamaktadır. Docker Engine API, birçok modern programlama dilinde yer alan HTTP kütüphanesi tarafından erişilen bir RESTful API'dir. Projedeki PHP SDK güncel olmadığı için, Docker API'yi kullanarak Laravel uygulaması ile haberleşme sağlayan DockerService adında bir servis yazılmıştır. Bu sayede konteyner oluşturma, çalıştırma, detaylarını görüntüleme gibi işlemler yapılabilmektedir.

4.2.2 Laravel Özellikleri

Bu projede, Laravel'in sağladığı özellikler kullanılarak geliştirme yapılmıştır. Bir IP'ye ping atma görevi oluşturma süreci şu şekildedir:

Kullanıcı, ping oluşturma görevi formunu doldurmalıdır. Formda, görev başlığı, görev açıklaması, atılacak IP adresi, oluşturulacak konteyner sayısı ve her bir konteynerda kaç adet ping atılacağı gibi fieldler bulunmaktadır.

Form doldurulduktan sonra, PingRequest sınıfı devreye girecektir. Bu, Laravel'in sağladığı bir özellik olan Request'tir ve formdan gelen bilgilerin doğruluğunu sağlamaktadır. Eğer bilgiler doğruysa, PingController'a geçilir. Eğer bilgiler doğrulanamazsa, form sayfasına geri yönlendirilir ve eksik olan fieldlerin altında bir hata mesajı yazılır.

PingController, yeni bir görev oluşturarak kullanıcıyı görev listesi sayfasına yönlendirir ve "oluşturuldu" mesajını yazdırır.

Yeni bir görev oluşturulduğunda, Laravel'in bir diğer özelliği olan "Event (Tetikleyici) & Listener (Dinleyici)" devreye girer.

Event & Listener: Bu proje, yeni bir görev oluşturulduğunda event tetiklenir ve Listener çalışmaya başlar. Listener aracılığıyla DockerService çalıştırılır. Kullanıcıyı Form sayfasında bekletmemek için Listener kuyruğa alınır ve DockerService işlemi tamamlandıktan sonra görev durumu güncellenir. Başarılıysa 1, başarısızsa 2 olarak bildirilir

4.3 Veritabanı kısmı İçin

MySQL veritabanı kullanılarak bir proje için optimize edilmiş bir veritabanı şeması oluşturuldu. Bu şema, "users" (kullanıcılar), "ping" (ping kayıtları) ve "pingcontainers" (ping konteynerleri) adında üç tabloyu içermektedir. Bu düzenleme, veritabanının performansını artırmak ve veri bütünlüğünü sağlamak amacıyla yapılmıştır.

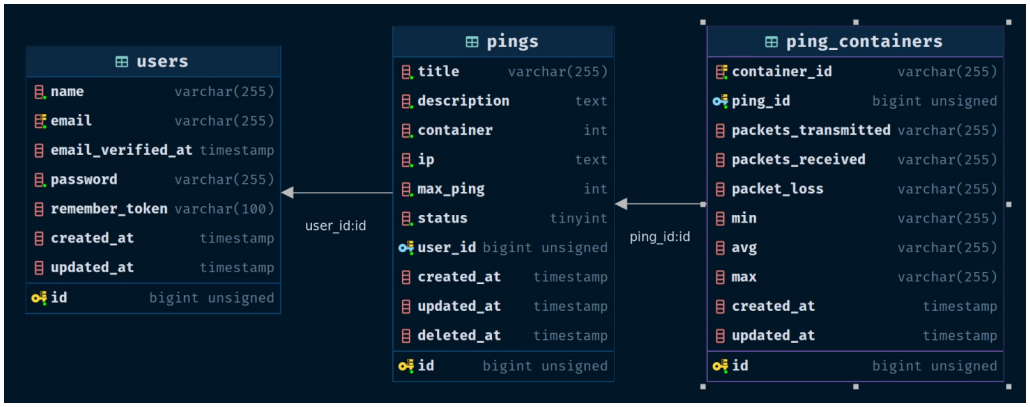
"users" tablosu, kullanıcıların temel bilgilerini içerir ve her kullanıcıya birincil anahtar (ID) ile benzersiz bir kimlik atar. Ayrıca kullanıcı adı, e-posta adresi ve diğer ilgili bilgiler gibi sütunlar da içerir.

"pingcontainers" tablosu, kullanıcıların oluşturabileceği ping konteynerlerini temsil eder. Her konteyner birincil anahtar ile tanımlanır ve kullanıcıya ait olacak şekilde kullanıcı kimliği (user ID) ile ilişkilendirilir. Bu tablo, kullanıcının birden fazla konteyner oluşturabilmesine olanak sağlar.

"ping" tablosu, gerçekleşen ping olaylarını kaydetmek için kullanılır. Her ping kaydı, birincil anahtar ile tanımlanır ve bir konteynere (container ID) ve kullanıcıya (user ID) bağlanır. Bu ilişki, her ping kaydının hangi kullanıcı ve konteyner tarafından oluşturulduğunu gösterir.

Bu veritabanı şeması, kullanıcıların birden fazla konteyner ve görev oluşturabilmesini sağlar. Ayrıca, her görevin bir kullanıcı tarafından oluşturulduğu ilişkisi sayesinde veri bütünlüğünü korur.

Bu düzenleme, veritabanının daha etkili bir şekilde veri saklamasını ve verilere hızlı erişim sağlamasını amaçlamaktadır. Ayrıca, veri bütünlüğünü korumak için gereken ilişkileri de sağlamaktadır.



Şekil 4.1: Sql ping tabloları

5 SONUÇLAR VE ÖNERİLER

6 EKLER

KAYNAKLAR

[1] Brogi,A.,Pahl,C.,andSoldani,J.2020.Onenhancingtheorchestrationofmulti-containerdockerapplications.InAdvancesinService-OrientedandCloudComputing:WorkshopsofESOCC2018,Como,Italy,September12âĂŞ14,2018,RevisedSelectedPapers,volume7,pages21âĂŞ33.SpringerInternationalPublishing. [Ziyaret Tarihi: 30 Mayıs 2022]

[2] Docker2023.Dockeroverview. [Ziyaret Tarihi: 12 Mart 2023]

[3] Ibrahim,M.H.,Sayagh,M.,andHassan,A.E.2021.Astudyofhowdockercomposeisusedtocomposemulti-componentsystems.EmpiricalSoftwareEngineering,26:1-27 [Ziyaret Tarihi: 6 Nisan 2022]

[4] Sharma,V.,Saxena,H.K.,andSingh,A.K.(2020).Docker3formulti-containerswebapplication.In20202ndInternationalConferenceonInnovativeMechanismsforIndustryApplications(ICIMIA),pages589âĂŞ592.IEEE. [Ziyaret Tarihi: 6 Nisan 2022]

[5] McKendrick,R.(2020).MasteringDocker:EnhanceyourcontainerizationandDevOpsskillstodeliverproduction-readyapplications.PacktPublishingLtd. [Ziyaret Tarihi: 6 Nisan 2022]

[6] Warrier,A.(2020).Containersvsvirtualmachines(vms). [Ziyaret Tarihi: 16 Mart 2023.]

ÖZGEÇMİŞ

KİŞİSEL BELGELER

Adı Soyadı :

Uyruğu : T.C.

Doğum Yeri ve Tarihi:

Adres :

Telefon :

E-mail :

EĞİTİM DURUMU

Lisans Öğrenimi : BŞEÜ Bilgisayar Mühendisliği Bölümü

Bitirme Yılı :

Lise :

İŞ DENEYİMLERİ

Yıl :

Kurum :

Stajlar :

İLGİ ALANLARI:

YABANCI DİLLER:

BELİRTMEK İSTEDİĐİNİZ DİĞER ÖZELLİKLER: