

Final Proje

Amaçlar

Bu laboratuvar çalışmasında:

- Basit bir Misafir Defteri uygulaması oluşturacak ve dağıtacaksınız
- Misafir Defteri uygulamasını Yatay Pod Ölçekleyici kullanarak otomatik ölçeklendireceksiniz
- Sürekli Güncellemeler ve Geri Alma işlemleri gerçekleştireceksiniz

Proje Genel Bakışı

Misafir Defteri Uygulaması

Misafir defteri, Docker ve Kubernetes ile inşa edip dağıtacağımız basit bir web uygulamasıdır. Uygulama, herhangi bir metin girebileceğiniz ve gönderebileceğiniz bir metin girişi olan bir web ön yüzünden oluşmaktadır. Tüm bunlar için Kubernetes Dağıtımları ve Pod'lar oluşturacağız. Ardından Misafir Defteri uygulamasına Yatay Pod Ölçeklendirmesi uygulayacağız ve son olarak Rolling Updates ve Rollbacks üzerinde çalışacağız.

Ortamı ve komut satırı araçlarını doğrulayın

1. Eğer bir terminal açık değilse, editördeki menüyü kullanarak bir terminal penceresi açın: Terminal > New Terminal.

Not: Terminal istemcisinin görünmesi için bir süre bekleyin.

2. Proje klasörünüze geçin.

Not: Eğer zaten /home/project klasöründeyseniz, bu adımı atlayabilirsiniz.

```
cd /home/project
```

3. Bu laboratuvar için gerekli olan artefaktları içeren git deposunu klonlayın.

```
[ ! -d 'guestbook' ] && git clone https://github.com/ibm-developer-skills-network/guestbook
```

4. Bu laboratuvar için dizine geçin.

```
cd guestbook
```

5. Bu dizinin içeriğini listeleyin ve bu laboratuvar için olan eserleri görün.

```
ls
```

Ziyaretçi Defteri Uygulamasını Oluştur

Başlamak için, ziyaretçi defteri uygulaması için web ön yüzünü oluşturacak ve dağıtacağız.

1. v1/guestbook dizinine geçin.

```
cd v1/guestbook
```

2. Dockerfile, çok aşamalı derlemeler olarak adlandırılan daha gelişmiş bir stratejiyi içerir, bu konuda daha fazla bilgi edinmek için [buraya](#) göz atabilirsiniz.

Görüntünüzü oluşturmak ve itmek için gerekli Docker komutlarıyla Dockerfile'ı tamamlayın. Bu dosyanın yolu guestbook/v1/guestbook/Dockerfile.

▼ İpucu!

FROM talimatı yeni bir derleme aşamasını başlatır ve sonraki talimatların üzerine inşa edileceği temel görüntüyü belirtir.

COPY komutu, dosyaları görüntümüze kopyalamamıza olanak tanır.

ADD komutu, dosyaları/dizinleri bir Docker görüntüsüne kopyalamak için kullanılır.

RUN talimatı komutları çalıştırır.

EXPOSE talimatı, belirli bir portu belirtilen bir protokolle bir Docker Konteyneri içinde açar.

CMD talimatı, bir konteyneri çalıştırmak için bir varsayılan sağlar, yani konteynerinizde çalışması gereken bir yürütülebilir dosyadır.

Tamamlanan dockerfile'ın ekran görüntüsünü alın ve **Dockerfile.png** olarak kaydedin.

3. Ad alanınızı bir ortam değişkeni olarak dışa aktarın, böylece sonraki komutlarda kullanılabilir hale gelsin.

```
export MY_NAMESPACE=sn-labs-$USERNAME
```

4. Docker Build komutunu kullanarak misafir defteri uygulamasını oluşturun.

▼ İpucu!

```
docker build . -t us.icr.io/$MY_NAMESPACE/guestbook:v1
```

5. Görüntüyü IBM Cloud Container Registry'ye gönderin.

▼ İpucu!

```
docker push us.icr.io/$MY_NAMESPACE/guestbook:v1
```

Not: Eğer bu laboratuvarı daha önce denediyseniz, önceki oturumun hala devam ediyor olma ihtimali vardır. Bu durumda, yukarıdaki çıktıda **'Pushed'** mesajı yerine **'Layer already Exists'** mesajını göreceksiniz. Laboratuvarın sonraki adımlarına geçmenizi öneririz.

6. Resmin başarıyla yüklendiğini doğrulayın.

```
ibmcloud cr images
```

Adım 6'nın çıktısının ekran görüntüsünü alın ve `crimages.png` dosya adıyla `.jpg` veya `.png` olarak kaydedin. Ekran görüntüsünü Peer Assignment'da yüklemeniz istenecek.

7. `v1/guestbook` dizinindeki `deployment.yml` dosyasını açın ve uygulamanın dağıtımı için kodu görüntüleyin:

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: guestbook
  labels:
    app: guestbook
spec:
  replicas: 1
  selector:
    matchLabels:
      app: guestbook
  strategy:
    rollingUpdate:
      maxSurge: 25%
      maxUnavailable: 25%
    type: RollingUpdate
  template:
    metadata:
      labels:
        app: guestbook
    spec:
      containers:
        - image: us.icr.io/<your sn labs namespace>/guestbook:v1
          imagePullPolicy: Always
          name: guestbook
          ports:
            - containerPort: 3000
              name: http
          resources:
            limits:
              cpu: 50m
            requests:
              cpu: 20m
```

Not: `<your sn labs namespace>` ifadesini SN lablar ad alanınızla değiştirin. SN lablar ad alanınızı kontrol etmek için lütfen `ibmcloud cr namespaces` komutunu çalıştırın.

- Aşağıdaki gibi görünmelidir:

8. Dağıtımı uygulamak için:

```
kubectl apply -f deployment.yml
```

9. Yeni bir Terminal açın ve uygulamanızı görüntülemek için aşağıdaki komutu girin:

```
kubectl port-forward deployment.apps/guestbook 3000:3000
```

10. Uygulamanızı 3000 numaralı portta başlatın. Sağdaki Skills Network butonuna tıklayın, bu **“Skills Network Toolbox”**’u açacaktır. Ardından **Diğer** seçeneğine, sonra **Uygulama Başlat** seçeneğine tıklayın. Buradan portu girip başlatmalısınız.

11. Artık çalışan uygulamanızı görebilmelisiniz. Aşağıda verilecek olan uygulama URL’sini kopyalayın:

Dağıtılan uygulamanızın bir ekran görüntüsünü alın ve `app.png` dosya adıyla `.jpg` veya `.png` olarak kaydedin. Ekran görüntüsünü Peer Assignment'da yüklemeniz istenecektir.

12. Birkaç giriş yaparak misafir defterini deneyin. **Gönder** butonuna bastıktan sonra giriş kutusunun üstünde görünmeleri gerekir.

Misafir Defteri uygulamasını Yatay Pod Autoscaler kullanarak otomatik ölçeklendirin

1. kubectl autoscale deployment kullanarak Misafir Defteri dağıtımını otomatik ölçeklendirin

▼ İpucu!

```
kubectl autoscale deployment guestbook --cpu-percent=5 --min=1 --max=10
```

2. Yeni oluşturulan HorizontalPodAutoscaler'ın mevcut durumunu kontrol etmek için şunu çalıştırabilirsiniz:

```
kubectl get hpa guestbook
```

Mevcut kopyalar 0'dır çünkü sunucuda yük yok.

Yatay Pod Otomatik Ölçekleyicinizin bir ekran görüntüsünü alın ve hpa.png dosya adıyla .jpg veya .png olarak kaydedin. Ekran görüntüsünü Peer Assignment'da yüklemeniz istenecektir.

3. Yeni bir terminal açın ve otomatik ölçeklemeyi gözlemlemek için uygulamaya yük oluşturmak üzere aşağıdaki komutu girin (Lütfen port yönlendirme komutunuzun çalıştığından emin olun. Uygulamanızı durdurduysanız, lütfen uygulamayı 3000 numaralı portta yeniden çalıştırmak için port yönlendirme komutunu çalıştırın.)

```
kubectl run -i --tty load-generator --rm --image=busybox:1.36.0 --restart=Never -- /bin/sh -c "while sleep 0.01; do wget -q -O- <your app URL>;
```

- Lütfen yukarıdaki komuttaki <your app URL> kısmını uygulamanızın URL'si ile değiştirin.

Not: Önceki görevin 11. adımında aldığınız kopyalanmış URL'yi kullanın.

Komut aşağıdaki gibi olacaktır:

Not: Eğer load generator already exists hatası alırsanız, load-generator ifadesinin sonuna bir sayı ekleyin, örneğin load-generator-1, load-generator-2.

- Uygulama üzerindeki artan yükü gösterecek benzer bir çıktı almaya devam edeceksiniz:

Not: 1. terminalde daha fazla komut çalıştırmaya devam edin.

4. Aşağıdaki komutu çalıştırarak otomatik ölçeklendirmeye bağlı olarak replikaların arttığını gözlemleyin:

```
kubectl get hpa guestbook --watch
```

5. Yukarıdaki komutu 5-10 dakika sonra tekrar çalıştırın ve uygulamanızın otomatik ölçeklendiğini gösteren replikaların sayısındaki artışı göreceksiniz.

Autoscaler detaylarınızın ekran görüntüsünü alın ve hpa2.png dosya adıyla .jpg veya .png olarak kaydedin. Ekran görüntüsünü Peer Assignment’da yüklemeniz istenecektir.

6. Aşağıdaki komutu çalıştırarak yatay pod otomatik ölçekleyicisinin detaylarını gözlemleyin:

```
kubectl get hpa guestbook
```

- Lütfen yük oluşturucu ve port yönlendirme komutlarının çalıştığı diğer terminalleri kapatın.

Misafir Defteri Uygulamasında Rolling Güncellemeleri ve Geri Alma İşlemlerini Gerçekleştirin

Not: Lütfen tüm komutları 1. terminalde çalıştırın, yeni bir terminal kullanılması gerektiği belirtilmedikçe.

1. Lütfen index.html dosyasındaki başlığı ve başlık kısmını <Adınız> **Misafir Defteri - v2** ve **Misafir Defteri - v2** gibi uygun başka bir başlık ve başlık ile güncelleyin.

▼ İpucu!

2. Güncellenmiş uygulama görüntünüzü oluşturmak ve itmek için aşağıdaki komutu çalıştırın:

▼ İpucu!

```
docker build . -t us.icr.io/$MY_NAMESPACE/guestbook:v1 && docker push us.icr.io/$MY_NAMESPACE/guestbook:v1
```

Güncellenmiş görüntünüzün bir ekran görüntüsünü alın ve upguestbook.png dosya adıyla .jpg veya .png olarak kaydedin. Ekran görüntüsünü Peer Assignment’da yüklemeniz istenecek.


3. deployment.yml dosyasındaki CPU değerlerini aşağıdaki gibi **cpu: 5m** ve **cpu: 2m** olarak güncelleyin:

▼ İpucu!

4. deployment.yml dosyasındaki değişiklikleri uygulayın.

▼ İpucu!

```
kubectl apply -f deployment.yml
```

 Adım 4’ün çıktısının detaylarının ekran görüntüsünü alın ve deployment.png dosya adıyla .jpg veya .png olarak kaydedin. Ekran görüntüsünü Peer Assignment’da yüklemeniz istenecektir.

5. Uygulamayı başlatmak için port-forward komutunu tekrar çalıştırın:

```
kubectl port-forward deployment.apps/guestbook 3000:3000
```

6. Uygulamanızı 3000 numaralı portta başlatın. Sağdaki “**Skills Network Toolbox**” butonuna tıklayın. Ardından **Diğer** ve sonra **Uygulamayı Başlat** seçeneğine tıklayın. Buradan portu girip başlatmalısınız.

7. Güncellenmiş uygulama içeriğini aşağıda göreceksiniz:

Güncellenmiş uygulamanızın ekran görüntüsünü alın ve up-app.png dosya adıyla .jpg veya .png olarak kaydedin. Ekran görüntüsünü Peer Assignment’da yüklemeniz istenecektir.

Not: Lütfen sonraki adımları çalıştırmadan önce uygulamayı durdurun.

8. Dağıtım sürümlerinin geçmişini görmek için aşağıdaki komutu çalıştırın:

```
kubect1 rollout history deployment/guestbook
```

9. Dağıtım yayılmasının Revizyon detaylarını görmek için aşağıdaki komutu çalıştırın:

```
kubect1 rollout history deployments guestbook --revision=2
```

Doğru Revizyonun detaylarının ekran görüntüsünü alın ve rev.png dosya adıyla .jpg veya .png olarak kaydedin. Ekran görüntüsünü Peer Assignment’da yüklemeniz istenecek.

10. Aşağıdaki komutu çalıştırarak replikasetleri alın ve şu anda kullanılan dağıtımı gözlemleyin:

```
kubect1 get rs
```

11. Aşağıdaki komutu çalıştırarak dağıtımı geri alın ve Sürüm 1 olarak ayarlayın:

```
kubect1 rollout undo deployment/guestbook --to-revision=1
```

12. Aşağıdaki komutu çalıştırarak Rollout geri alındıktan sonra replikasyon setlerini alın. Kullanılan dağıtım aşağıdaki gibi değişmiş olacaktır:

```
kubect1 get rs
```

Adım 9’un çıktısının ekran görüntüsünü alın ve rs.png dosya adıyla .jpg veya .png olarak kaydedin. Ekran görüntüsünü Peer Assignment’da yüklemeniz istenecektir.

Tebrikler! Bu kursun final projesini tamamladınız. Laboratuvar ortamından çıkış yapmayın (tarayıcıyı kapatabilirsiniz) veya laboratuvar sırasında oluşturulan herhangi bir nesneyi silmeyin, çünkü bunlar bir sonraki laboratuvar için gerekecektir, Opsiyonel: OpenShift İç Registry’den Guestbook Uygulamasını Dağıtın.

