

İsteğe Bağlı: OpenShift Dahili Kayıt Defterinden Misafir Defteri Uygulamasını Dağıt



Amaçlar

Bu laboratuvarın sonunda:

- Basit bir misafir defteri uygulaması oluşturacak ve dağıtacaksınız.
- Güncellemeleri yaymak için OpenShift görüntü akışlarını kullanacaksınız.
- Misafir defteri uygulamasının çok katmanlı bir versiyonunu dağıtacaksınız.

Ön Koşul

Final Ödevi'nde verilen Docker komutlarını kullanarak Misafir Defteri uygulamasını oluşturmuş ve itmiş olmalısınız.

OpenShift Dahili Kayıt Defterinden Misafir Defteri Uygulamasını Dağıt

Kurs boyunca tartışıldığı gibi, IBM Cloud Container Registry, görüntülerin güvenliğini sağlamak için yaygın güvenlik açıkları ve zafiyetler için tarama yapar. Ancak OpenShift ayrıca dahili bir kayıt defteri de sağlar – görüntü akışları ve görüntü akış etiketleri konusunu hatırlayın. Dahili kayıt defterini kullanmanın da avantajları vardır. Örneğin, dağıtımlar için görüntüleri çekerken daha az gecikme olur. Ya her ikisini de kullanabilseydik - IBM Cloud Container Registry'yi görüntülerimizi taramak için kullanıp, ardından daha düşük gecikme için bu görüntüleri otomatik olarak dahili kayıt defterine ithal etseydik?

Lütfen önceki laboratuvardan kaldığınız adımlardaki aşağıdaki komutlarla devam edin.

1. IBM Cloud Container Registry'deki görüntünüze işaret eden bir görüntü akışı oluşturun.

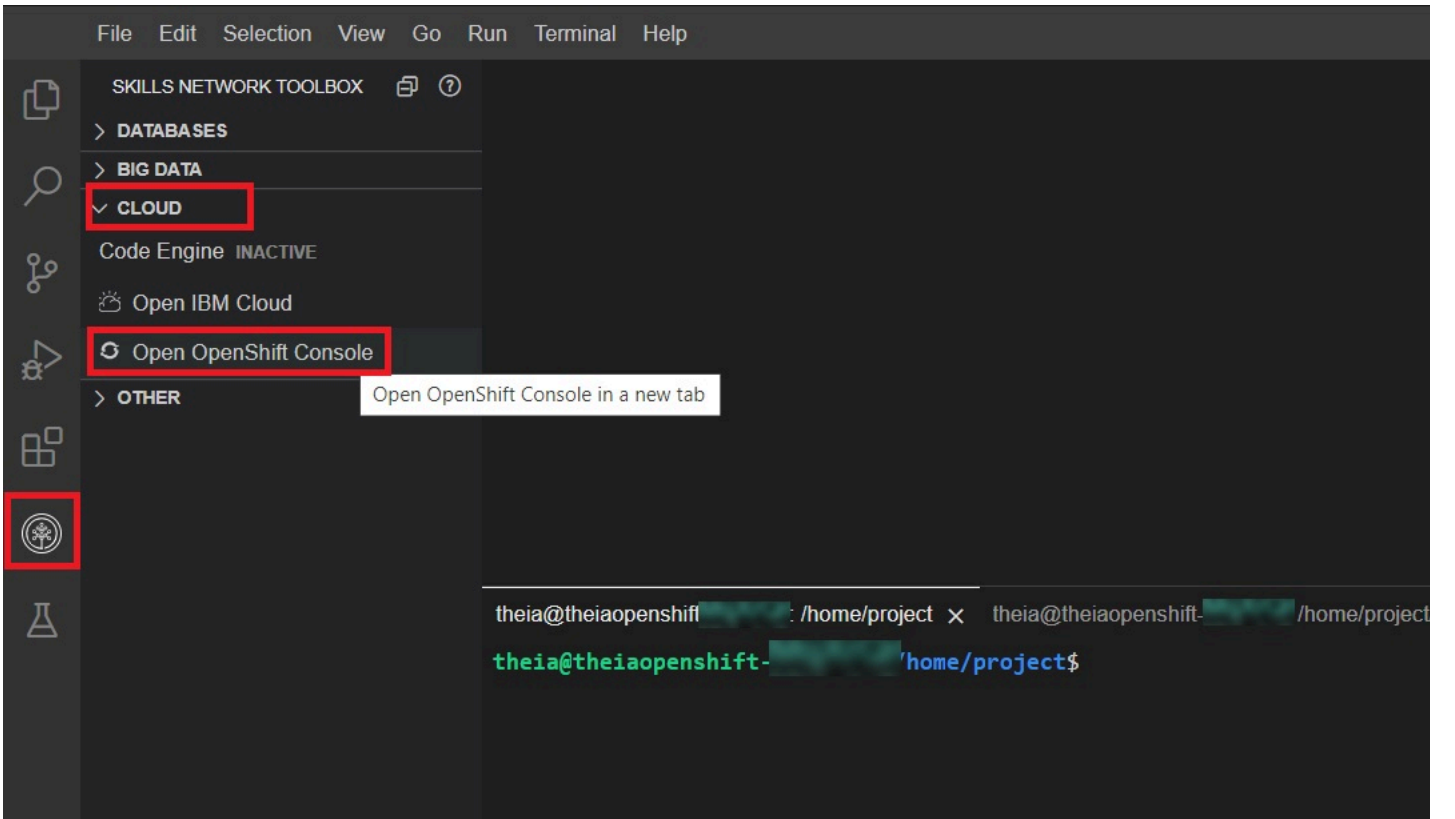
```
oc tag us.icr.io/$MY_NAMESPACE/guestbook:v1 guestbook:v1 --reference-policy=local --scheduled
```

--reference-policy=local seçeneği ile, IBM Cloud Container Registry'den bir görüntü kopyası yerel ön belleğe aktarılır ve bu, kümenin projelerine bir görüntü akışı olarak sunulur. --schedule seçeneği, IBM Cloud Container Registry'den iç kayıt defterine periyodik olarak görüntü aktarımını ayarlar. Varsayılan sıklık 15 dakikadır.

```
theia@theiaopenshift- [redacted] :/home/project/guestbook/v1/guestbook$ oc tag us.icr.io/$MY_NAMESPACE/guestbook:v1 guestbook:v1 --re
ed
Tag guestbook:v1 set to import us.icr.io/sn-labs-[redacted] guestbook:v1 periodically.
theia@theiaopenshift- [redacted] :/home/project/guestbook/v1/guestbook$
```

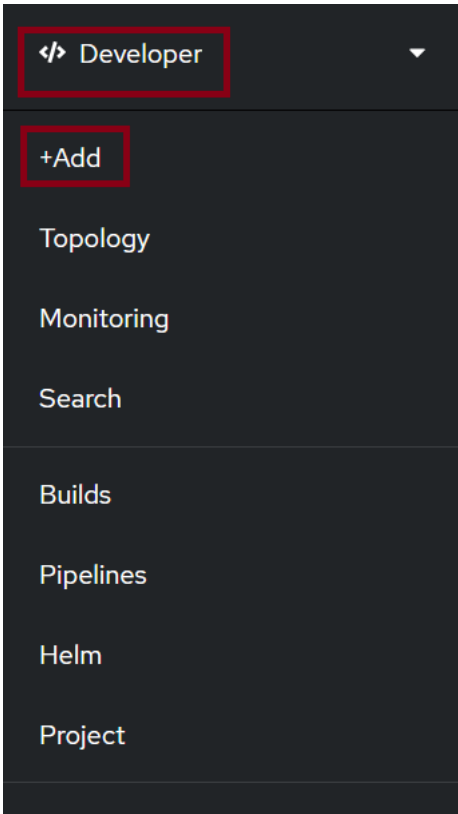
Şimdi bu görüntü akışını kullanarak guestbook uygulamasını dağıtmak için OpenShift web konsoluna gidelim.

2. Skills Network Tool üzerine tıklayın ve açılır menüyü açmak için Cloud seçeneğini seçin. Ardından Open OpenShift console seçeneğine tıklayın, bu Open Shift Web konsolunu yeni bir pencerede açacaktır.



Not: Şu anda OpenShift konsolunda bazı zorluklar yaşıyoruz. Eğer ekranınızın yüklenmesi uzun sürüyorsa, lütfen OpenShift konsolu tarayıcı sekmesini kapatın ve tekrar başlatın. Ekranın yüklenmesi 10 dakikaya kadar sürebilir.

3. Geliştirici görünümünden, bu projeye yeni bir uygulama eklemek için **+Add** butonuna tıklayın.



4. Uygulamayı iç kayıt defterindeki bir görüntü kullanarak dağıtmak için **Container Image** seçeneğine tıklayın.


Add

Select a way to create an application, component or service from one of the options.

Quick Starts


- Deploying an application with a pipeline
- Getting started with a sample
- Adding health checks to your sample application

[See all Quick Starts →](#)




Samples

Create an application from a code sample




From Git

Import code from your Git repository to be built and deployed




Container Image

Deploy an existing image from an image registry or image stream tag




From Catalog

Browse the catalog to discover, deploy and connect to services




Database

Browse the catalog to discover database services to add to your application



Operator Backed

Browse the catalog to discover and deploy operator managed services



Helm Chart

Browse the catalog to discover and install Helm Charts

5. Image altında, “Image stream tag from internal registry” seçeneğine geçin.

Deploy Image

Image

Deploy an existing image from an image stream or image registry.

☒ Image name from external registry

Enter an image name

To deploy an image from a private repository, you must [create an image pull secret](#) with your image registry credentials.

☐ Allow images from insecure registries

☐ Image stream tag from internal registry

6. Projenizi ve yeni oluşturduğunuz görüntü akışını ve etiketini (guestbook ve v1, sırasıyla) seçin. Bu alanlar için yalnızca bir seçeneğiniz olmalı, çünkü yalnızca bir projeye erişiminiz var ve yalnızca bir görüntü akışı ve bir görüntü akışı etiketi oluşturduunuz.

Image

Deploy an existing image from an image stream or image registry.


☐ Image name from external registry

☒ Image stream tag from internal registry

Project * Image Stream * Tag *

sn-labs- / guestbook : v1

Runtime Icon

 openshift

The icon represents your image in Topology view. A label will also be added to the resource defining the icon.

7. Tüm varsayılan değerleri koruyun ve en altta **Create** butonuna tıklayın. Bu, uygulamayı oluşturacak ve sizi Topoloji görünümüne yönlendirecektir.

Image

Deploy an existing image from an image stream or image registry.

- ☐ Image name from external registry
- ☒ Image stream tag from internal registry

Project * Image Stream * Tag *

sn-labs- / guestbook : v1

Runtime Icon

 openshift

The icon represents your image in Topology view. A label will also be added to the resource defining the icon.

General

Application Name

guestbook-app

A unique name given to the application grouping to label your resources.

Name *

guestbook

A unique name given to the component that will be used to name associated resources.

Resources

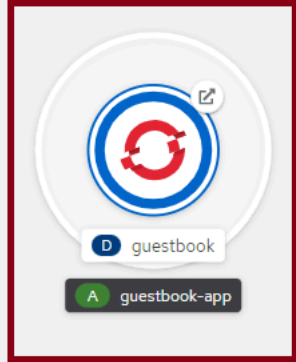
Select the resource type to generate

- ☒ Deployment
- apps/Deployment
- A Deployment enables declarative updates for Pods and ReplicaSets.

Create Cancel

8. Topoloji görünümünden guestbook Dağıtımına tıklayın. Bu, uygulamayı çalıştıran Pod'u ve onu açığa çıkaran Service ve Route'u görebileceğiniz bu Dağıtımın **Resources** sekmesine yönlendirecektir.

Lütfen bekleyin, çünkü Topoloji görünümündeki dağıtımların çalışması zaman alabilir.



guestbook

Actions

Health Checks

Container guestbook does not have health checks to ensure your application is running correctly. [Add Health Checks](#)

Details

Resources

Monitoring

Pods

guestbook-56dbf59579-mz2l5 Running View

Builds

No Build Configs found for this resource.

Services

guestbook
Service port: 3000-tcp → Pod Port: 3000

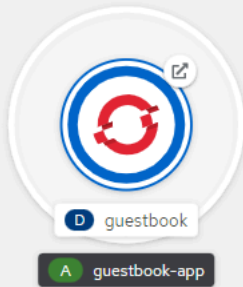
Routes

RT guestbook
Location:
<http://guestbook-sn-labs-...labs-prod-openshift-san-a4563ldc5778dc637lc67d206ba9ae5c-0000.us-east.containers.appdomain.cloud>

Not: Lütfen Topoloji görünümündeki opensearch-console dağıtımını silmeyin, çünkü bu OpenShift konsolunun düzgün çalışması için gereklidir.

9. Guestbook'u görmek için Route konumuna (bağlantı) tıklayın.

Not: Uygulamayı başlatmadan önce pod'un durumunun **'Running'** olarak değişmesini bekleyin.



guestbook

Actions

Health Checks

Container guestbook does not have health checks to ensure your application is running correctly. [Add Health Checks](#)

Details

Resources

Monitoring

Pods

guestbook-56dbf59579-mz2l5 Running View

Builds

No Build Configs found for this resource.

Services

guestbook
Service port: 3000-tcp → Pod Port: 3000

Routes

RT guestbook
Location:
<http://guestbook-sn-labs-...labs-prod-openshift-san-a4563ldc5778dc637lc67d206ba9ae5c-0000.us-east.containers.appdomain.cloud>

10. Birkaç giriş yaparak guestbook'u deneyin. **Submit** butonuna bastıktan sonra, girişlerin giriş kutusunun üstünde görünmesi gerekir.

➤ ↻ ⚠ Not secure | guestbook-sn-labs-...labs-prod-openshift-san-a45631dc5778dc6371c67d206ba9ae5c-0000.us-east

Guestbook - v1

hello
wishes for the day!

http://guestbook-sn-labs-...labs-prod-openshift-san-a45631dc5778dc6371c67d206ba9ae5c-0000.us-east.containers.appdomain.cloud
[/env](#) [/info](#)

Misafir Defterini Güncelle

Misafir defterini güncelleyelim ve OpenShift'in görüntü akışlarının uygulamalarımızı kolayca güncellemeye nasıl yardımcı olabileceğine bakalım.

1. public dizinindeki index.html dosyasını düzenlemek için Explorer'ı kullanın. Bu dosyanın yolu guestbook/v1/guestbook/public/index.html.

2. Başlığı daha spesifik hale getirelim. 12. satırda bulunan `<h1>Guestbook - v1</h1>` ifadesini, adınızı içerecek şekilde değiştirin. Örneğin `<h1>Alex'in Misafir Defteri - v1</h1>` gibi. İşiniz bittiğinde dosyayı kaydetmeyi unutmayın.

```
index.html x
guestbook > v1 > guestbook > public > index.html
1 <!DOCTYPE html>
2 <html lang="en">
3   <head>
4     <meta content="text/html; charset=utf-8" http-equiv="Content-Type">
5     <meta charset="utf-8">
6     <meta content="width=device-width" name="viewport">
7     <link href="style.css" rel="stylesheet">
8     <title>Lavanya's Guestbook - v1</title>
9   </head>
10  <body>
11    <div id="header">
12      <h1>Alex's Guestbook - v1</h1>
13    </div>
14
15    <div id="guestbook-entries">
16      <link href="https://afeld.github.io/emoji-css/emoji.css" rel="stylesheet">
17      <p>Waiting for database connection... <i class='em em-boat'></i></p>
18    </div>
19
20    <div>
21      <form id="guestbook-form">
22        <input autocomplete="off" id="guestbook-entry-content" type="text">
23        <a href="#" id="guestbook-submit">Submit</a>
24      </form>
25    </div>
26
27  </body>
28</html>
```

3. Uygulamayı aynı etiketle tekrar oluşturun ve gönderin. Bu, önceki görüntüyü üzerine yazacaktır.

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

```
theia@theiaopenshift-... /home/project/guestbook/v1/guestbook$ docker build . -t us.icr.io/$MY_NAMESPACE/guestbook:v1 && doc
ook:v1
Sending build context to Docker daemon 98.3kB
Step 1/14 : FROM golang:1.15 as builder
--> 40349a2425ef
Step 2/14 : RUN go get github.com/codegangsta/negroni
--> Using cache
--> a7657fc96c64
Step 3/14 : RUN go get github.com/gorilla/mux github.com/xyproto/simpleredis
--> Using cache
--> 1f28b8fef54e
Step 4/14 : COPY main.go .
--> Using cache
--> 3458048c5c1e
Step 5/14 : RUN go build main.go
--> Using cache
--> 823973fe49e6
Step 6/14 : FROM ubuntu:18.04
--> f5cbcd4244ba
Step 7/14 : COPY --from=builder /go//main /app/guestbook
--> Using cache
--> 0350722f466e
Step 8/14 : ADD public/index.html /app/public/index.html
--> a9644611258d
Step 9/14 : ADD public/script.js /app/public/script.js
--> 89215eb5fecb
Step 10/14 : ADD public/style.css /app/public/style.css
--> 7760ade3c7d3
Step 11/14 : ADD public/jquery.min.js /app/public/jquery.min.js
--> 97abdf76f88d
Step 12/14 : WORKDIR /app
--> Running in 1b78236b819b
Removing intermediate container 1b78236b819b
--> 6f056b0cf5b7
```

4. OpenShift iç kayıt defterine görüntümüzü içe aktarırken belirttiğimiz --schedule seçeneğini hatırlayın. Sonuç olarak, OpenShift belirttiğimiz etikete itilen yeni görüntüleri düzenli olarak içe aktaracaktır. Yeni oluşturduğumuz görüntüyü aynı etikete ittiğimiz için, OpenShift güncellenmiş görüntüyü yaklaşık 15 dakika içinde içe aktaracaktır. OpenShift'in görüntüyü otomatik olarak içe aktarmasını beklemek istemiyorsanız, aşağıdaki komutu çalıştırın.

```
oc import-image guestbook:v1 --from=us.icr.io/$MY_NAMESPACE/guestbook:v1 --confirm
```

```
theia@theiaopenshift: /home/project/guestbook/v1/guestbook$ oc import-image guestbook:v1 --from=us.icr.io/$MY_NAMESPACE/guestbook
imagestream.image.openshift.io/guestbook imported

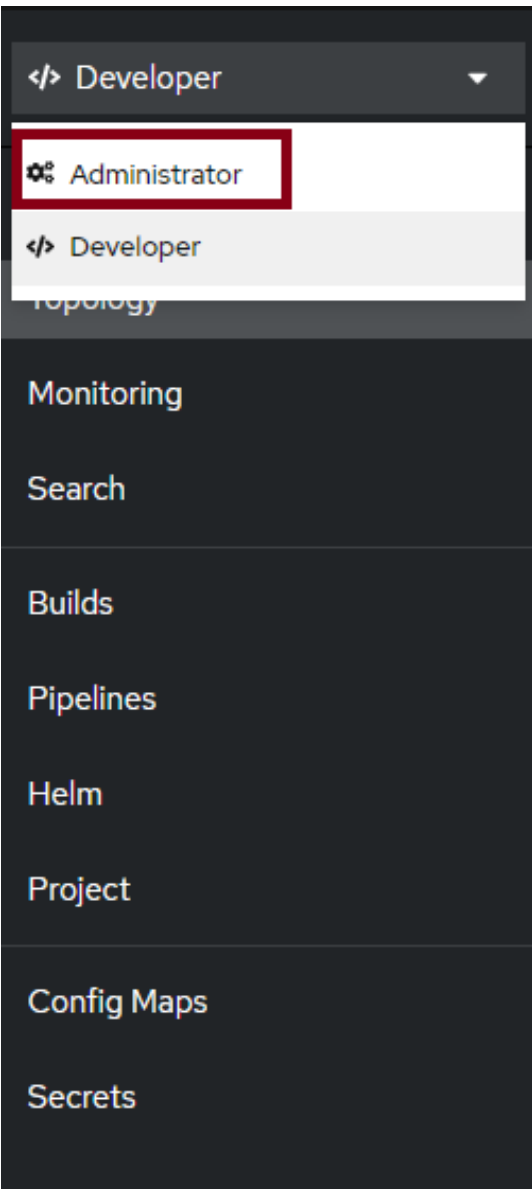
Name:          guestbook
Namespace:     sn-labs-
Created:       2 minutes ago
Labels:        <none>
Annotations:   openshift.io/image.dockerRepositoryCheck=2022-04-11T08:28:50Z
Image Repository: image-registry.openshift-image-registry.svc:5000/sn-labs- guestbook
Image Lookup:   local=false
Unique Images: 2
Tags:          1

v1
updates automatically from registry us.icr.io/sn-labs- /guestbook:v1
prefer registry pullthrough when referencing this tag

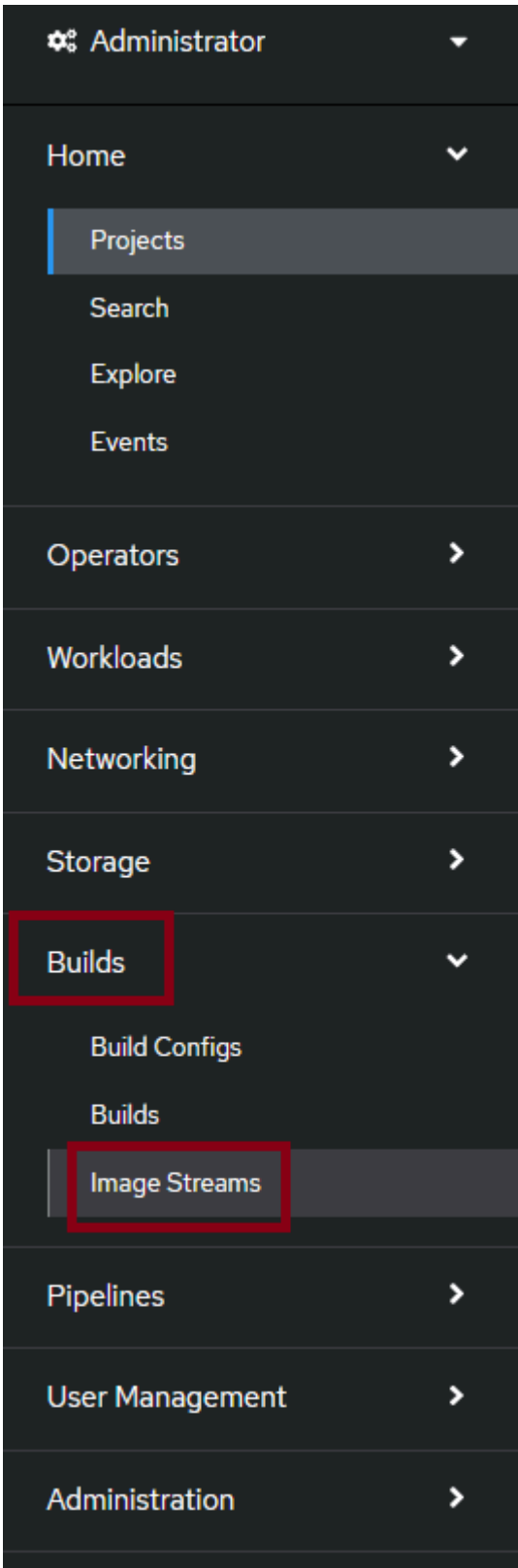
* us.icr.io/sn-labs- /guestbook@sha256:f16401f8452ae414e5feafbb621ef20a0b9a12aafc800679465d809525ee454d
  Less than a second ago
  us.icr.io/sn-labs- /guestbook@sha256:11ee56c6d46d80f0fda0790f50121bf3a4760240d74baef01a5c6bded6e94ef7
    2 minutes ago

Image Name:      guestbook:v1
Docker Image:    us.icr.io/sn-labs- /guestbook@sha256:f16401f8452ae414e5feafbb621ef20a0b9a12aafc800679465d809525ee454d
Name:            sha256:f16401f8452ae414e5feafbb621ef20a0b9a12aafc800679465d809525ee454d
Created:         Less than a second ago
Annotations:     image.openshift.io/dockerLayersOrder=ascending
Image Size:      31.53MB in 6 layers
Layers:          26.71MB sha256:08a6abff89437fab99b52abbefed82ea907f12845c30eeb94f6b93c69be93166
                  4.791MB sha256:1158f7aa125a353046e120906d415a3f0e7b2fb43fa61ffed49645f9eb423948
                  650B   sha256:98e5b3b04689c7d80aebec108d7d88619d06fe3771382c6fa44d7273b74d5faa
                  608B   sha256:e3277739aac4aaac2fbabf0e282914400e62c5e67da5cea7825eb5b0a519aca0
                  545B   sha256:5a480e70e8fe40c001754b4143e4bb13a945b38e5adf7082a83ef70145d45b0d
                  29.88kB sha256:5e702cd921b48b45d5f1ad2fa8d75390a12189f4c249acf995cc5185f3015a83
Image Created:   28 seconds ago
Author:          <none>
Arch:           amd64
```

5. Yönetici perspektifine geçin, böylece görüntü akışlarını görebilirsiniz.





6. Navigasyonda **Builds** > **Image Streams**'e tıklayın.



7. guestbook görüntü akışına tıklayın.


Image Streams

Name ▾ Search by name... 

Name ↑	Labels ↓	Cr
 guestbook	No labels	



8. **History** menüsüne tıklayın. Burada yalnızca bir giriş görüyorsanız, OpenShift'in yeni görüntünüzü henüz içe aktarmadığı anlamına gelir. Birkaç dakika bekleyin ve sayfayı yenileyin. Sonunda, bu görüntü akış etiketi için yeni bir sürümün içe aktarıldığını gösteren ikinci bir girişi görmelisiniz. Bu işlem biraz zaman alabilir çünkü varsayılan içe aktarma sıklığı 15 dakikadır.

[Image Streams](#) > [Image Stream Details](#)



 guestbook

Details [YAML](#) [History](#)

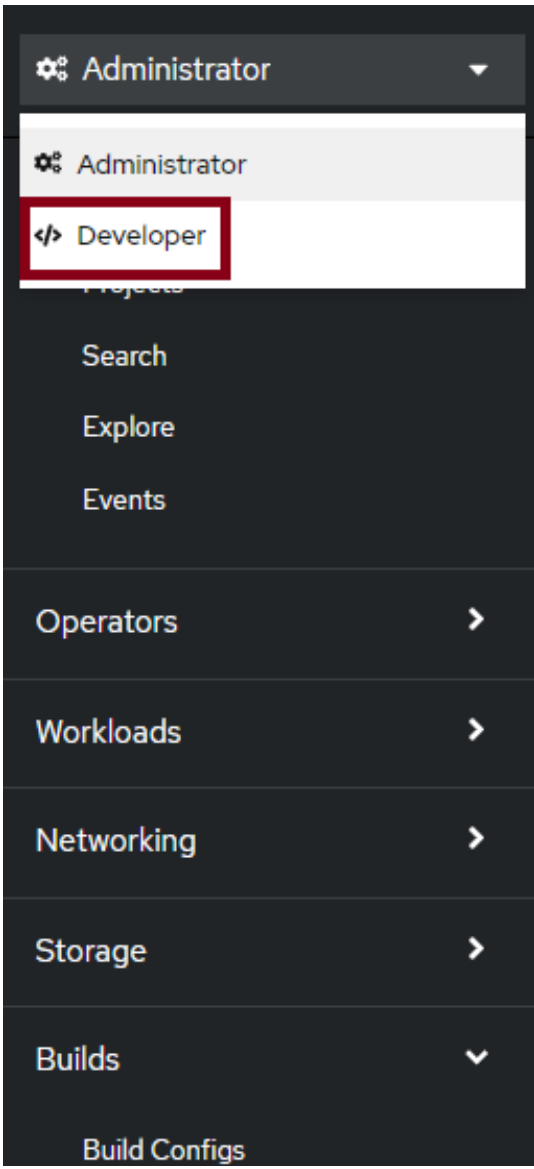
a minute ago

 guestbook:v1
from us.icr.io/sn-labs-/guestbook
sha256:f16401f8452ae414e5feafbb621ef20a0b9a12aafc800679465d809525ee454d

3 minutes ago

 guestbook:v1
from us.icr.io/sn-labs-/guestbook
sha256:11ee56c6d46d80f0fda0790f50121bf3a4760240d74baef01a5c6bded6e94ef7

9. **Geliştirici** perspektifine geri dönün.



Not: OpenShift konsolunun ve Geliştirici perspektifinin yüklenmesi için lütfen biraz bekleyin.

10. Tarayıcıda misafir defterini tekrar görüntüleyin. Eğer sekmeyi hâlâ açık tutuyorsanız, oraya gidin. Değilse, guestbook Dağıtımı'ndan Yolu tekrar tıklayın. Bu sayfada yeni başlığınızı görmelisiniz! OpenShift yeni sürümümüzü içe aktardı ve Dağıtım görüntü akışına işaret ettiğinden, bu yeni sürümü çalıştırmaya başladı.

⚠ Not secure | guestbook-sn-labs-...labs-prod-openshift-san-a45631dc5778dc6371c67d206ba9ae5c-0000.us-east.contai

Alex's Guestbook

<http://guestbook-sn-labs-...labs-prod-openshift-san-a45631dc...east.containers.appdomain.cloud/>

Misafir Defteri Depolama

1. Tarayıcıdaki misafir defterinden, giriş kutusunun altındaki `/info` bağlantısına tıklayın. Bu, misafir defteri için bir bilgi uç noktasıdır.

Alex's Guestbook

http://guestbook-sn-labs- labs-prod-openshift-san-a45631dc
east.containers.appdomain.cloud/
[/env/info](#)

“Bellek içi veri deposu (redis değil)” yazdığımı göreceksiniz. Şu anda yalnızca misafir defteri web ön yüzünü dağıttık, bu nedenle girişleri takip etmek için bellek içi veri deposunu kullanıyor. Ancak bu çok dayanıklı değil, çünkü herhangi bir güncelleme veya hatta Pod’un yeniden başlatılması, girişlerin kaybolmasına neden olacaktır. Ama bunu doğrulayalım.

← → ↻ ⚠ Not secure | guestbook-sn-labs- labs-prod-openshift-san-a45631dc5778dc6371c67d206ba9ae5c-0000

In-memory datastore (not redis)

2. Tarayıcıda misafir defteri uygulamasına geri dönmek için Route konumuna tekrar tıklayın. Önceki girişlerinizin artık görünmediğini görmelisiniz. Bunun nedeni, en son bölümde güncelleniz dağıtıldığında misafir defterinin yeniden başlatılmasıdır. Yeniden başlatmalardan sonra misafir defteri girişlerini kalıcı hale getirmenin bir yoluna ihtiyacımız var.

↻ ⚠ Not secure | guestbook-sn-labs- labs-prod-openshift-san-a45631dc5778dc6371c67d206ba9ae5c-0000.us-east.co

Alex's Guestbook

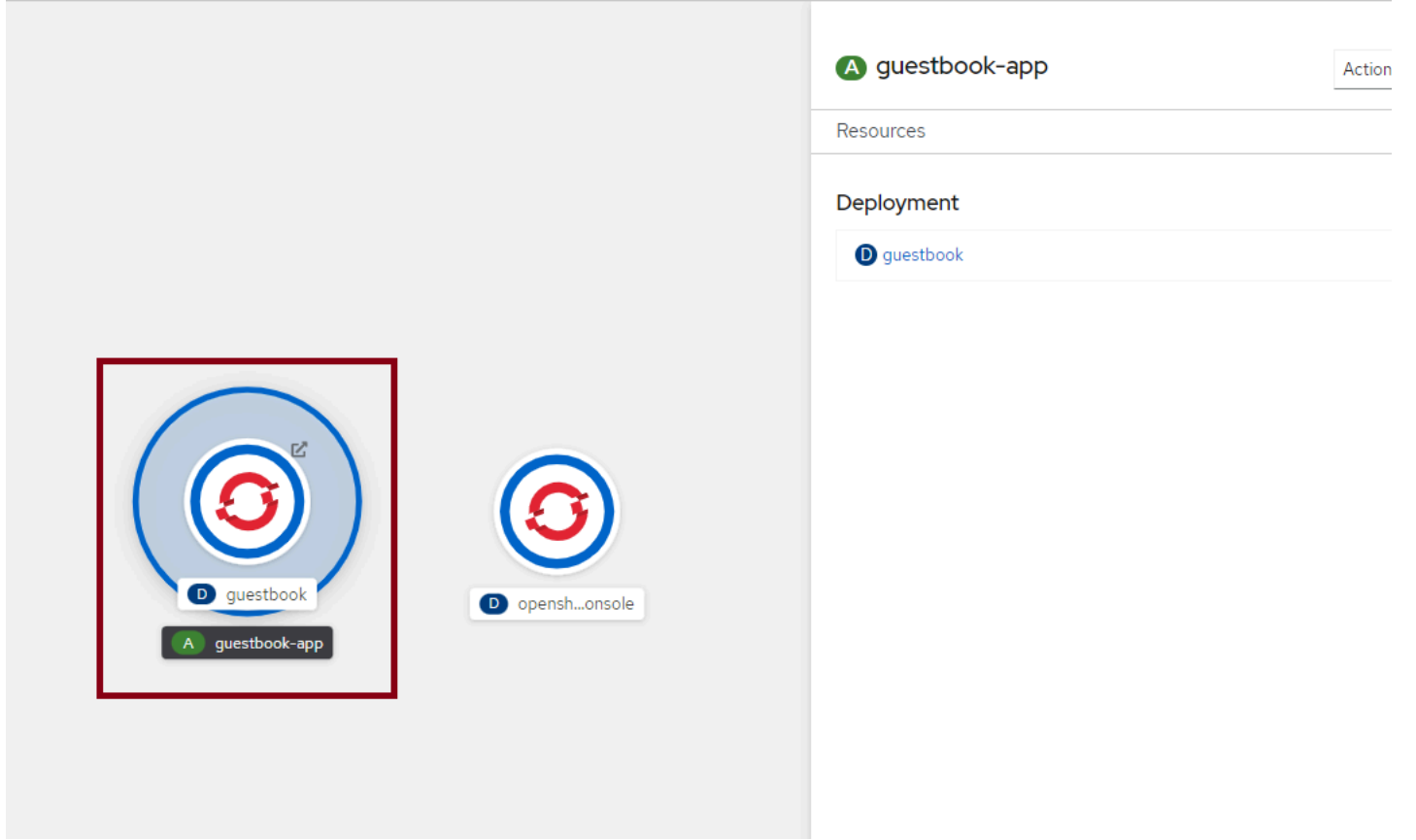
http://guestbook-sn-labs- labs-prod-openshift-san-a45631c
east.containers.appdomain.clou

Not: Şu anda OpenShift konsoluyla belirli zorluklar yaşıyoruz. Görsel akışın güncellenmesi zaman aldığı için eski girişlerinizi görebilirsiniz. Laboratuvarın sonraki adımlarına geçebilirsiniz.

Misafir defterini sil

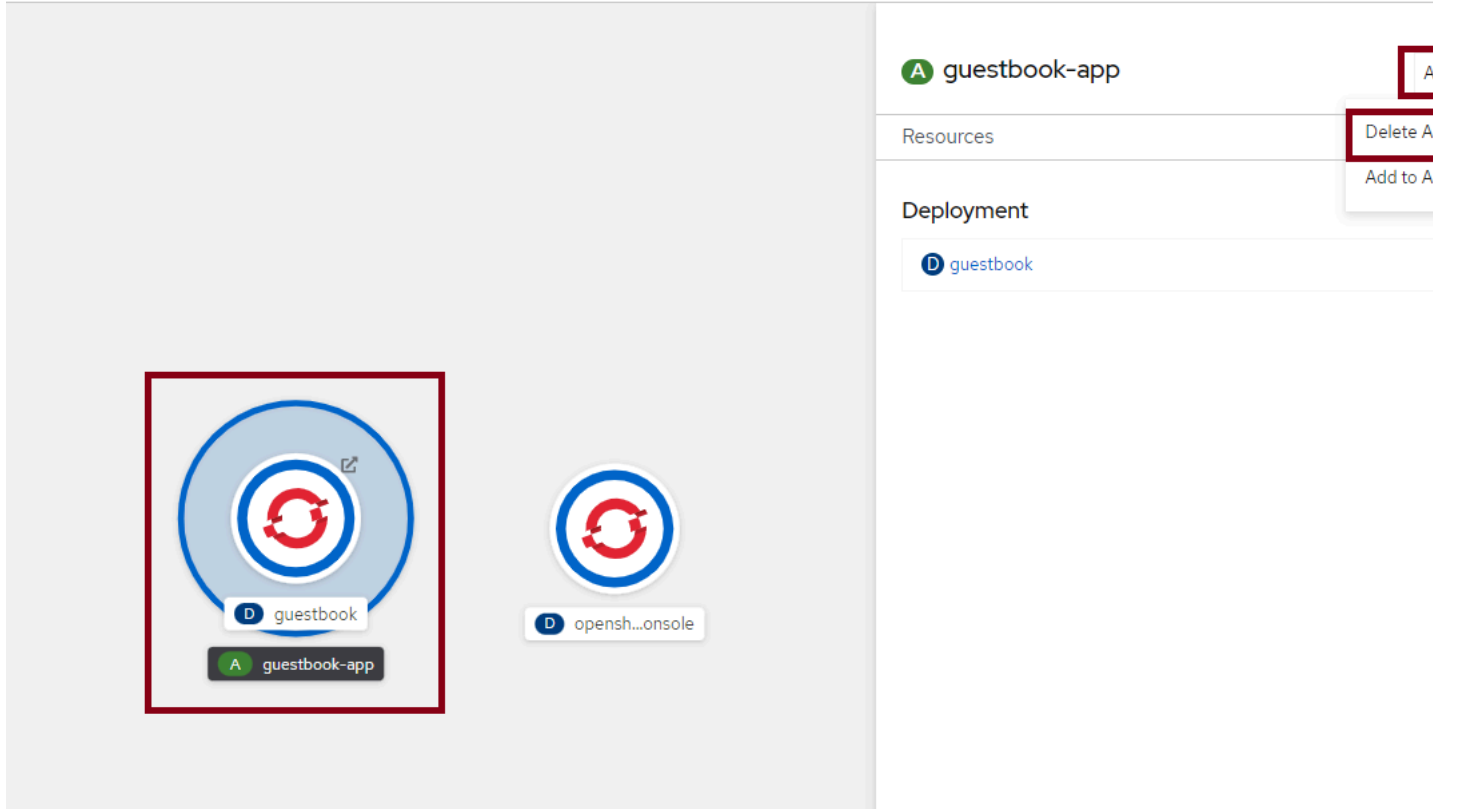
Daha karmaşık bir misafir defteri sürümünü dağıtmak için, bu basit sürümü silin.

1. Topoloji görünümünden, guestbook-app uygulamasına tıklayın. Bu, guestbook Dağıtımını çevreleyen açık gri dairedir.



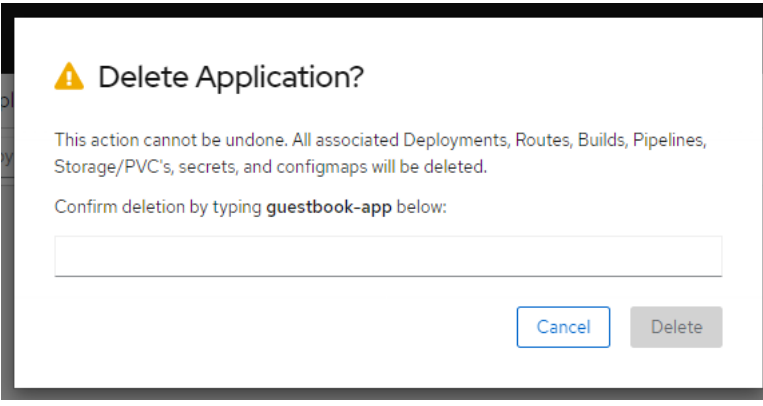
The screenshot shows the K8s dashboard. On the left, the 'Topologies' view displays two applications: 'guestbook' and 'guestbook-app'. The 'guestbook-app' application is highlighted with a red box. On the right, the 'Details' sidebar for 'guestbook-app' is visible. It includes an 'Action' button, a 'Resources' section, and a 'Deployment' section showing 'guestbook'.

2. Eylemler > Uygulamayı Sil seçeneğine tıklayın.



The screenshot shows the K8s dashboard. On the left, the 'Topologies' view displays two applications: 'guestbook' and 'guestbook-app'. The 'guestbook-app' application is highlighted with a red box. On the right, the 'Details' sidebar for 'guestbook-app' is visible. It includes an 'Action' button, a 'Resources' section, and a 'Deployment' section showing 'guestbook'. A red box highlights the 'Delete' button in the 'Action' menu.

3. Uygulama adını yazın ve Sil butonuna tıklayın.



Redis Master ve Slave Dağıtımı

Misafir defterinin etkili olabilmesi için kalıcı depolama gerektiğini gösterdik. Bunu sağlamak için Redis'i dağıtalım. Redis, bir veritabanı, önbellek ve mesaj aracı olarak kullanılan açık kaynaklı, bellek içi veri yapısı deposudur.

Bu uygulama, misafir defteri web ön yüzünün v2 versiyonunu kullanır ve 1) depolama için bir Redis master ve 2) bir kopyalanmış Redis slave seti ekler. Bu bileşenlerin her biri için Kubernetes Dağıtımları, Pod'lar ve Servisler bulunmaktadır. Kubernetes üzerinde çok katmanlı bir uygulama oluştururken en büyük endişelerden biri, bu ayrı ayrı dağıtılan bileşenler arasındaki bağımlılıkları çözmektir.

Çok katmanlı bir uygulamada, hizmet bağımlılıklarının çözülmesi için iki ana yöntem vardır. v2/guestbook/main.go kodu her birinin örneklerini sağlar. Redis için, master uç noktası ortam değişkenleri aracılığıyla keşfedilir. Bu ortam değişkenleri, Redis servisleri başlatıldığında ayarlanır, bu nedenle hizmet kaynakları, misafir defteri Pod'ları başlamadan önce oluşturulmalıdır. Sonuç olarak, uygulama bileşenlerini oluştururken belirli bir sırayı takip edeceğiz. Önce Redis bileşenleri oluşturulacak, ardından misafir defteri uygulaması.

Not: Eğer bu laboratuvarı daha önce denediyseniz, önceki oturumun hala kalıcı olma ihtimali vardır. Böyle bir durumda, dağıtımları oluşturmak için **Apply** komutunu çalıştırdığınızda **'Created'** mesajı yerine **'Unchanged'** mesajını göreceksiniz. Laboratuvarın sonraki adımlarına devam etmenizi öneririz.

1. Laboratuvar ortamındaki terminalden v2 dizinine geçin.

```
cd ../../v2
```

```
theia@theiaopenshift: /home/project/guestbook/v1/guestbook$ cd ../../v2
theia@theiaopenshift: /home/project/guestbook/v2$ █
```

2. Aşağıdaki komutu çalıştırın veya redis-master-deployment.yaml dosyasını Gezginde açarak Redis master için Dağıtım yapılandırmasıyla tanışın.

```
cat redis-master-deployment.yaml
```

```

theia@theiaopenshift- [redacted] /home/project/guestbook/v2$ cat redis-master-deployment.yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: redis-master
  labels:
    app: redis
    role: master
spec:
  replicas: 1
  selector:
    matchLabels:
      app: redis
      role: master
  template:
    metadata:
      labels:
        app: redis
        role: master
    spec:
      containers:
        - name: redis-master
          image: redis:5.0.5
          ports:
            - name: redis-server
              containerPort: 6379
          volumeMounts:
            - name: redis-storage
              mountPath: /data
          volumes:
            - name: redis-storage
              emptyDir: {}
theia@theiaopenshift- [redacted] /home/project/guestbook/v2$

```

3. Redis ana Dağıtımını oluşturun.

```
oc apply -f redis-master-deployment.yaml
```

```

theia@theiaopenshift- [redacted] /home/project/guestbook/v2$ oc apply -f redis-master-deployment.yaml
deployment.apps/redis-master created
theia@theiaopenshift- [redacted] /home/project/guestbook/v2$

```

4. Dağıtımın oluşturulduğunu doğrulayın.

```
oc get deployments
```

```

theia@theiaopenshift- [redacted] /home/project/guestbook/v2$ oc get deployments
NAME                READY   UP-TO-DATE   AVAILABLE   AGE
openshift-web-console 1/1     1             1           14m
redis-master         1/1     1             1           34s
theia@theiaopenshift- [redacted] /home/project/guestbook/v2$

```

5. Dağıtım tarafından oluşturulan Pod' u görmek için Pod' ları listeleyin.

```
oc get pods
```

```

theia@theiaopenshift- [redacted] /home/project/guestbook/v2$ oc get pods
NAME                                READY   STATUS    RESTARTS   AGE
openshift-web-console-77d78f965-6kr12 2/2     Running   0           14m
redis-master-d98597c5b-f2g54          1/1     Running   0           58s
theia@theiaopenshift- [redacted] /home/project/guestbook/v2$

```

Ayrıca OpenShift web konsolunda Topoloji görünümüne geri dönebilir ve Dağıtımın orada görüldüğünü görebilirsiniz.

6. Aşağıdaki komutu çalıştırın veya redis-master-service.yaml dosyasını Gezginde açarak Redis ana hizmetinin yapılandırmasıyla tanışın.

```
cat redis-master-service.yaml
```

```
theia@theiaopenshift-.../home/project/guestbook/v2$ cat redis-master-service.yaml
apiVersion: v1
kind: Service
metadata:
  name: redis-master
  labels:
    app: redis
    role: master
spec:
  ports:
    - port: 6379
      targetPort: redis-server
  selector:
    app: redis
    role: master
theia@theiaopenshift-.../home/project/guestbook/v2$
```

Hizmetler, Pod etiketlerine dayalı olarak yük dengelemesi yapmak için Pod'ları bulur. Önceki adımda oluşturduğunuz Pod'un etiketleri app=redis ve role=master'dır. Hizmetin seçim alanı, Hizmete gönderilen trafiği hangi Pod'ların alacağını belirler.

7. Redis master Hizmetini oluşturun.

```
oc apply -f redis-master-service.yaml
```

```
theia@theiaopenshift-.../home/project/guestbook/v2$ oc apply -f redis-master-service.yaml
service/redis-master created
theia@theiaopenshift-.../home/project/guestbook/v2$
```

Topology görünümünde redis-master Dağıtımına tıkladığınızda, **Kaynaklar** sekmesinde redis-master Servisini görmelisiniz.

The screenshot displays the OpenShift console interface. On the left, the 'Topology' view shows a cluster of pods, with the 'redis-master' pod highlighted by a red rectangular box. On the right, the 'redis-master' service details are shown. The 'Resources' tab is active, displaying the 'Pods' section with one pod, 'redis-master-d98597c5b-f2g54', in a 'Running' state. Below this, the 'Services' section is highlighted with a red rectangular box, showing the 'redis-master' service with the configuration 'Service port: TCP/6379 → Pod Port: redis-server'. A 'Health Checks' warning is visible at the top right, stating that the container does not have health checks.

8. Aşağıdaki komutu çalıştırın veya Redis köle için Dağıtım yapılandırmasını tanımak amacıyla redis-slave-deployment.yaml dosyasını Gezginde açın.

```
cat redis-slave-deployment.yaml
```



```

theia@theiaopenshift: /home/project/guestbook/v2$ cat redis-slave-deployment.yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: redis-slave
  labels:
    app: redis
    role: slave
spec:
  replicas: 2
  selector:
    matchLabels:
      app: redis
      role: slave
  template:
    metadata:
      labels:
        app: redis
        role: slave
    spec:
      containers:
        - name: redis-slave
          image: redis:5.0.5
          command: ["/bin/sh"]
          args: ["-c", "redis-server --slaveof redis-master 6379"]
          ports:
            - name: redis-server
              containerPort: 6379
          volumeMounts:
            - name: redis-storage
              mountPath: /data
          volumes:
            - name: redis-storage
              emptyDir: {}
theia@theiaopenshift: /home/project/guestbook/v2$

```

9. Redis köle Dağıtımını oluşturun.

```
oc apply -f redis-slave-deployment.yaml
```

```

theia@theiaopenshift: /home/project/guestbook/v2$ oc apply -f redis-slave-deployment.yaml
deployment.apps/redis-slave created
theia@theiaopenshift: /home/project/guestbook/v2$

```

10. Dağıtımın oluşturulduğunu doğrulayın.

```
oc get deployments
```

```

theia@theiaopenshift: /home/project/guestbook/v2$ oc get deployments
NAME                READY   UP-TO-DATE   AVAILABLE   AGE
openshift-web-console 1/1     1             1           25m
redis-master         1/1     1             1           12m
redis-slave          2/2     2             2           29s
theia@theiaopenshift: /home/project/guestbook/v2$

```

11. Dağıtım tarafından oluşturulan Pod'u görmek için Pod'ları listeleyin.

```
oc get pods
```

```
theia@theiaopenshift- /home/project/guestbook/v2$ oc get pods
NAME                                READY   STATUS    RESTARTS   AGE
openshift-web-console-77d78f965-6kr12 2/2     Running   0           26m
redis-master-d98597c5b-f2g54          1/1     Running   0           13m
redis-slave-76dfcf6864-7qkmq          1/1     Running   0           66s
redis-slave-76dfcf6864-lqnnq          1/1     Running   0           66s
theia@theiaopenshift- /home/project/guestbook/v2$
```

Ayrıca OpenShift web konsolundaki Topoloji görünümüne geri dönebilir ve Dağıtımın orada görüldüğünü görebilirsiniz.

12. Aşağıdaki komutu çalıştırın veya redis-slave-service.yaml dosyasını Gezginde açarak Redis slave için Hizmet yapılandırmasını tanıyın.

```
cat redis-slave-service.yaml
```

```
theia@theiaopenshift- /home/project/guestbook/v2$ cat redis-slave-service.yaml
apiVersion: v1
kind: Service
metadata:
  name: redis-slave
  labels:
    app: redis
    role: slave
spec:
  ports:
    - port: 6379
      targetPort: redis-server
  selector:
    app: redis
    role: slave
```

13. Redis köle Servisini oluşturun.

```
oc apply -f redis-slave-service.yaml
```

```
theia@theiaopenshift- /home/project/guestbook/v2$ oc apply -f redis-slave-service.yaml
service/redis-slave created
```

Topology görünümünde redis-slave Dağıtımına tıkladığınızda, **Kaynaklar** sekmesinde redis-slave Servisini görmelisiniz.

redis-slave

Health Checks

Container redis-slave does not have health checks to ensure your application is running correctly. Add Health Checks

Details Resources Monitoring

Pods

redis-slave-76dfcf6864-7qkmq	Running	View logs
redis-slave-76dfcf6864-lqnnq	Running	View logs

Builds

No Build Configs found for this resource.

Services

redis-slave	Service port: TCP/6379 → Pod Port: redis-server
-------------	---

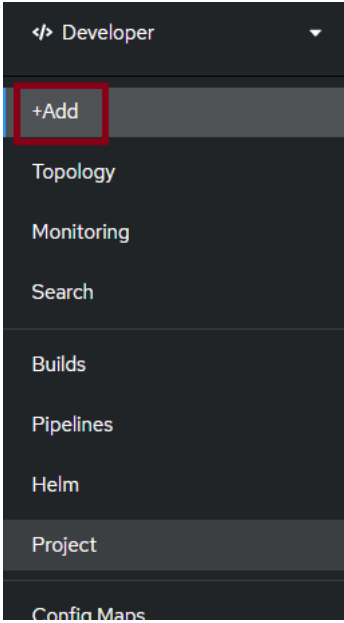
Routes

No Routes found for this resource.

v2 misafir defteri uygulamasını dağıt

Artık kalıcı depolama için Redis'i kullanacak misafir defteri uygulamasının ikinci sürümünü dağıtma zamanı.

1. Bu projeye yeni bir uygulama eklemek için **+Ekle** butonuna tıklayın.



OpenShift'te mevcut çeşitli seçenekleri göstermek için bu misafir defteri uygulamasını bir OpenShift derlemesi ve repodaki Dockerfile ile dağıtacağız.

2. **Git Deposu** (Git'ten İçer Aktar) seçeneğine tıklayın.

lavanyar-console.openshift-sandbox.labs.cognitiveclass.ai/add/ns/sn-labs-lavanyar

Dell Lavs

Skills Network OpenShift Lab

Developer

+Add

Topology

Search

Builds

Pipelines

Helm

Project

ConfigMaps

Secrets

Project: sn-labs-lavanyar

View all samples

View all quick starts

Developer Catalog

All services

Browse the catalog to discover, deploy and connect to services

Database

Browse the catalog to discover database services to add to your application

Operator Backed

Browse the catalog to discover and deploy operator managed services

Helm Chart

Browse the catalog to discover and install Helm Charts

Git Repository

Import from Git

Import code from your Git repository to be built and deployed

Samples

Create an application from a code sample

3. Aşağıdaki URL'yi **Git Repo URL** kutusuna yapıştırın.

<https://github.com/ibm-developer-skills-network/guestbook>

Onaylı bir onay işareti görmelisiniz kutunun dışına tıkladığınızda.

Not: Kopyalanacak Git Repo URL'sinde boşluk olmadığından emin olun.

Developer

+Add

Topology

Search

Project: sn-labs-lavanyar

Application: All applications

Git

Git Repo URL *

<https://github.com/ibm-developer-skills-network/guestbook>

Validated

4. Gelişmiş Git Seçeneklerini Göster butonuna tıklayın.

Developer

+Add

Topology

Search

Builds

Pipelines

Helm

Project: sn-labs-haroonmd

Application: All applications

Import from Git

Git

Git Repo URL *

https://github.com/ibm-developer-skills-network/guestbook

Validated

Show advanced Git options

5. Dockerfile, depolamanın kökünde olmadığı için OpenShift'e nerede olduğunu söylememiz gerekiyor. **Context Dir** kutusuna /v2/guestbook yazın.

Helm

Project

ConfigMaps

Secrets

Hide advanced Git options

Git reference

Optional branch, tag, or commit.

Context dir

/v2/guestbook

Optional subdirectory for the source code, used as a context directory for build.

6. Hedef port olarak 3000 girin ve geri kalan varsayılan seçenekleri olduğu gibi bırakın, ardından **Oluştur** butonuna tıklayın. OpenShift'e bir Dockerfile verdiğimiz için, bir BuildConfig ve Dockerfile'ı kullanarak bir imaj oluşturacak, bunu dahili kayıt defterine itecektir ve bu imajı bir Dağıtım için kullanacaktır.

Source Secret

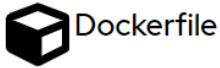
Select Secret name

Secret with credentials for pulling your source code.



Multiple import strategies detected

The Dockerfile at Dockerfile is recommended.



Edit Import Strategy

General

Application

guestbook-app

Select an Application to group this component.

Name *

guestbook

guestbook-app

A unique name given to the application grouping to label your resources.

Name *

guestbook

A unique name given to the component that will be used to name associated resources.

Resource type

Deployment

Resource type to generate. The default can be set in [User Preferences](#).

Pipelines

☐ Add pipeline

Advanced options

Target port

3000

Target port for traffic.

☒ Create a route

Exposes your component at a public URL

[Show advanced Routing options](#)

Click on the names to access advanced options for Health checks, Build configuration, Deployment, Scaling, Resource limits, and Labels.

Create

Cancel

7. Topoloji görünümünden, guestbook Dağıtımına tıklayın.

Kaynaklar sekmesinde, tarayıcıda guestbook'u yüklemek için Route konumuna tıklayın. Başlığın "Guestbook - v2" yerine "Guestbook - v1" dediğini fark edin.

Not: Lütfen yönlendirme bağlantısına tıklamadan önce Yapıların tamamlanmasını bekleyin

The screenshot shows the Kubernetes dashboard interface. On the left, the 'Resources' view displays a cluster of resources: 'redis-master', 'opensh...onsole', 'redis-slave', 'guestbook-git', and 'guestb...it-app'. The 'guestbook-git' resource is highlighted with a red box. On the right, the 'Details' view for 'guestbook-git' is shown. It includes a 'Health Checks' section with a message: 'Container guestbook-git does not have health checks application is running correctly. [Add Health Checks](#)'. Below this, the 'Resources' tab is active, showing a 'Waiting for the build' message: 'Waiting for the first build to run successfully. You may "ImagePullBackOff" and "ErrImagePull" errors while w'. The 'Builds' section shows 'Build #1 is complete (a few seconds ago)'. The 'Services' section shows 'guestbook-git' with 'Service port: 3000-tcp → Pod Port: 3000'. The 'Routes' section shows 'guestbook-git' with a location: 'http://guestbook-git-sn-labs-...labs-prod-...a45631dc5778dc6371c67d206ba9ae5c-0000-us-east.containers.appdomain.cloud'.

8. Tarayıcıdaki guestbook'tan, giriş kutusunun altındaki /info bağlantısına tıklayın.

Guestbook - v2

http://guestbook-git-sn-labs-... labs-prod-openshift-san-a45631dc5778dc6371c67d206ba9ae5c-0000.us-east.containers.appdomain.cloud/
[/env](#) [/info](#)

Artık bellek içi veri deposunu kullanmadığımız için Redis hakkında bilgi verdiğini fark edin.

← → ↻ ⚠ Not secure | guestbook-git-sn-labs-... labs-prod-openshift-san-a45631dc5778dc6371c67d206ba9ae5c-0000.us-east.co

```
# Server
redis_version:5.0.5
redis_git_sha1:00000000
redis_git_dirty:0
redis_build_id:442b43d467cd2b03
redis_mode:standalone
os:Linux 3.10.0-1160.59.1.el7.x86_64 x86_64
arch_bits:64
multiplexing_api:epoll
atomicvar_api:atomic-builtin
gcc_version:8.3.0
process_id:1
run_id:ddfb01eeefa82ace731a16c65dae1ddc6cbe031d3
tcp_port:6379
uptime_in_seconds:1637
uptime_in_days:0
hz:10
configured_hz:10
lru_clock:5500602
executable:/data/redis-server
config_file:
```

```
# Clients
connected_clients:1
client_recent_max_input_buffer:2
client_recent_max_output_buffer:0
blocked_clients:0
```

```
# Memory
used_memory:1944584
used_memory_human:1.85M
used_memory_rss:10461184
used_memory_rss_human:9.98M
used_memory_peak:1963600
used_memory_peak_human:1.87M
used_memory_peak_perc:99.03%
used_memory_overhead:1923354
used_memory_startup:791240
used_memory_dataset:21230
used_memory_dataset_perc:1.84%
allocator_allocated:1919168
allocator_active:2138112
allocator_resident:5660672
```

- Uygulamanızı Openshift'te oturum sürekliliği veya diğer hatalar nedeniyle silmek ve yeniden dağıtmak isterseniz, lütfen adımları [buradan](#) takip edin.

Yukarıdakileri yaptıktan sonra, guestbook uygulamanıza herhangi bir yönlendirme görmüyorsanız, terminalde aşağıdaki komutu çalıştırarak uygulama yönlendirmesini alın:

```
oc status
```

Misafir defteri uygulaması, rota üzerine tıkladıktan sonra bir süre ‘Veritabanı bağlantısı bekleniyor’ durumunu gösterebilir; bu nedenle kutuya eklenen girişler görünmeyecektir. Uygulamanın hazır olması için bir süre beklemeniz ve ardından girişlerinizi eklemeniz gerekebilir, böylece doğru bir şekilde görünürler.

© IBM Corporation. Tüm hakları saklıdır.