

Red Hat OpenShift'e Giriş



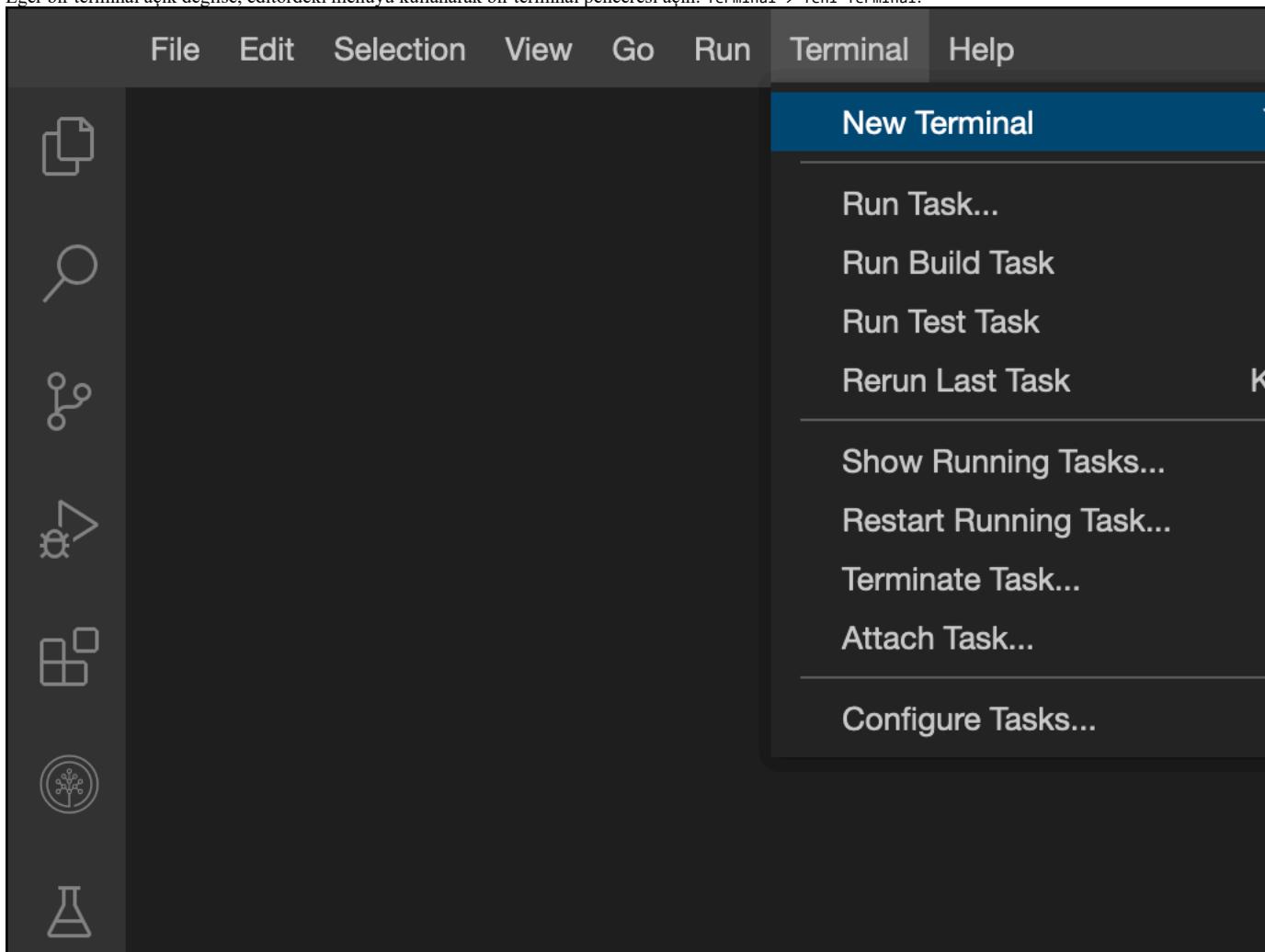
Amaçlar

Bu laboratuvara şunları yapacaksınız:

- oc CLI'yi (OpenShift komut satırı arayüzü) kullanın
- OpenShift web konsolunu kullanın
- s2i ('Source-to-image' yapı stratejisi) kullanarak bir uygulama oluşturun ve dağıtin
- Bir BuildConfig ve bir ImageStream'i inceleyin
- Uygulamayı otomatik ölçeklendirin

Ortamı ve komut satırı araçlarını doğrulayın

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



Not: Terminal istemcisinin görünmesi için biraz bekleyin.

2. oc CLI'nin kurulu olduğunu doğrulayın.

```
oc version
```

```
theia@theiaopenshift- [ ~ ] :/home/project$ oc version
Client Version: 4.13.7
Kustomize Version: v4.5.7
Kubernetes Version: v1.27.16+03a907c
```

Buna benzer bir çıktı görmelisiniz, ancak sürümler farklı olabilir.

3. Proje klasörünüze geçin.

NOT: Eğer zaten home/project klasöründeyseniz bu adımı atlayabilirsiniz.

```
cd /home/project
```

4. Bu laboratuvar için gerekli olan belgeleri içeren git deposunu, eğer zaten mevcut değilse, klonlayın.

```
[ ! -d 'CC201' ] && git clone https://github.com/ibm-developer-skills-network/CC201.git
```

```
theia@theiaopenshift- [ ~ ] :/home/project$ [ ! -d 'CC201' ] && git clone https://github.com/ibm-developer-skills-network/CC201.git
t
Cloning into 'CC201'...
remote: Enumerating objects: 20, done.
remote: Counting objects: 100% (20/20), done.
remote: Compressing objects: 100% (13/13), done.
remote: Total 20 (delta 6), reused 19 (delta 6), pack-reused 0
Unpacking objects: 100% (20/20), done.
```

oc CLI Kullanımı

OpenShift projeleri, ek yönetim işlevleri ile birlikte Kubernetes ad alanlarıdır. Bu nedenle, projeler aynı zamanda bir OpenShift kümesi içinde izolasyon sağlar. Bir OpenShift kümesinde bir projeye zaten erişiminiz var ve oc zaten o küme ve projeyi hedef alacak şekilde ayarlanmıştır.

Bazı temel oc komutlarına bakalım. oc'nun bir kopyası ile birlikte geldiğini unutmayın, bu nedenle tüm kubectl komutları oc ile çalıştırılabilir.

1. Bu ad alanındaki Pod'ları listeleyin.

```
oc get pods
```

```
theia@theiaopenshift- [ ~ ] :/home/project$ oc get pods
NAME                  READY   STATUS    RESTARTS   AGE
openshift-web-console-995896df-vz2tp   2/2     Running   0          4h1m
```

Muhtemelen ortamın bir parçası olan birkaç Pod göreceksiniz. Bunlar hakkında endişelenmenize gerek yok.

2. Kubernetes nesnelerine ek olarak, OpenShift'e özgü nesneleri de alabilirsiniz.

```
oc get buildconfigs
```

```
theia@theiaopenshift- [ ] :/home/project$ oc get buildconfigs
No resources found in sn-labs- [ ] namespace.
```

Henüz bir BuildConfig oluşturmadığınız için bu, herhangi bir kaynak döndürmeyecektir.

3. Şu anda kullanılan OpenShift projesini görüntüleyin.

```
oc project
```

```
theia@theiaopenshift- [ ] :/home/project$ oc project
Using project "sn-labs- [ ] " from context named " [ ] -context" on server "https://c109-e.us-east.containers.cloud.ibm.com:30
807".
theia@theiaopenshift- [ ] :/home/project$ [ ]
```

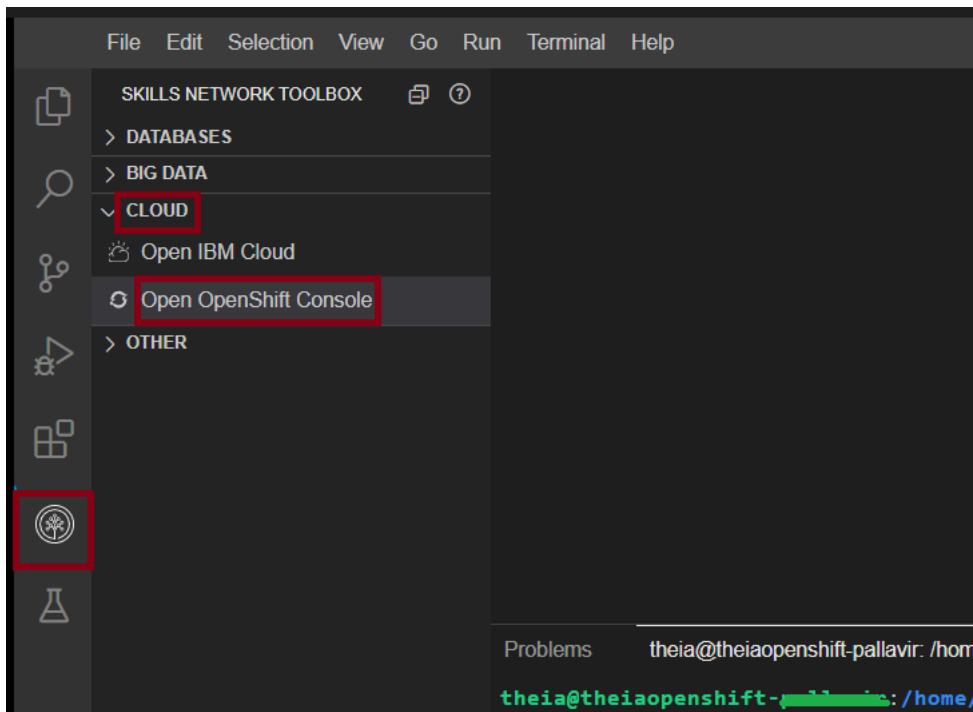
Bu proje size özeldir ve küme içinde izolasyon sağlar, böylece kendi uygulamalarınızı dağıtabilirsiniz.

OpenShift web konsolunu kullanın

CLI'ye ek olarak, OpenShift sevgisel bir web konsolu sağlar. Bu, uygulamaları dağıtmaya, kaynakları görüntülemeye, uygulamaları izlemeye ve günlükleri görmeye olanak tanıdığı için faydalı ve güçlü bir özelliktir ve çok daha fazlasını konsolda yapmanızı sağlar.

Konsolu açalım ve etrafa bakalım.

1. OpenShift web konsolunu açmak için sağdaki Skills Network butonuna tıklayın, bu **Skills Network Toolbox**'u açacaktır. Ardından **Cloud**'a tıklayın ve ardından aşağıdaki resimde gösterildiği gibi **Open OpenShift console**'a tıklayın.

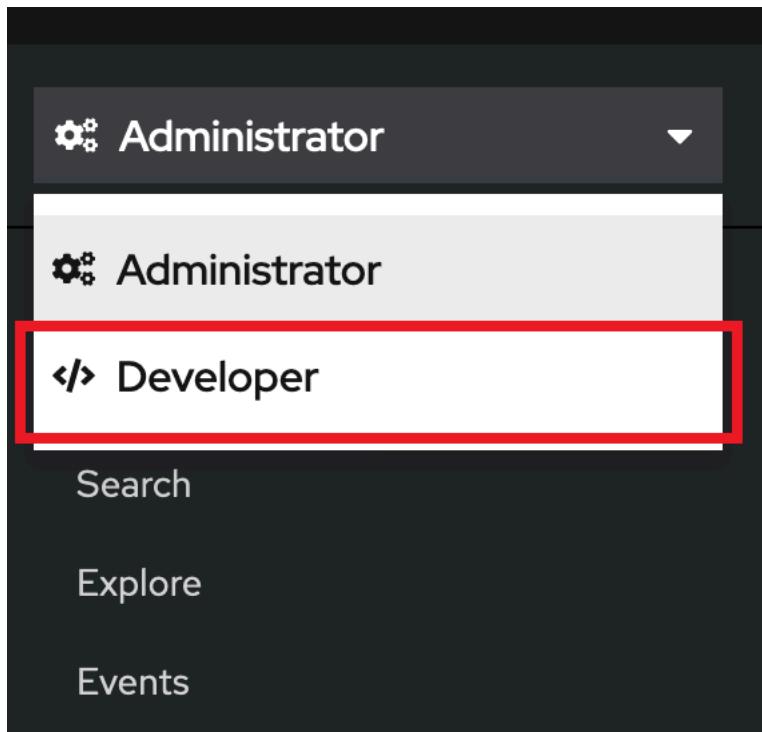


Laboratuvar ortamını açtıktan sonra erişilebilir hale gelmesi birkaç dakika alabilir, bu nedenle bir hata alırsanız bir dakika bekleyip tekrar deneyin.

2. Konsol, size atanmış proje için proje detaylarına açılmalıdır. OpenShift'in size sevgisel ve görsel bir şekilde sunduğu tüm bilgilere göz atın. Bu proje için Dashboard, Overview ve diğer sekmelere tıklayarak ek bilgilere ulaşabilirsiniz. Bu projede mevcut olan kaynakların envanterini, bu projeyi tanımlayan YAML'yi ve çok daha fazlasını görmelisiniz.

3. Sol navigation menüsündeki öğelere aşina olun. Operatörleri, birçok farklı Kubernetes nesnesini ve bu kursta bahsettiğimiz bazı OpenShift'e özgü nesneleri görebilirsiniz. Bu nesnelerin henüz çok fazla örneği olmayacağı, ancak uygulamamızı dağıttığımızda dolacaktır.

4. Sol üstteki **“Administrator”** kelimesine dikkat edin. Bu, Yönetici perspektifinde olduğunuzu gösterir. Ayrıca bir Geliştirici perspektifi de vardır. Her perspektif, o persona için özel iş akışları sunar. Bir uygulama dağıtmaya başlamak için **Geliştirici perspektifine geçin**. (Eğer zaten “Developer” yazıyorsa, değiştirmeyin.)



Bir uygulamayı web konsolunda dağıtmak

Geliştirici perspektifi, uygulama oluşturma ve dağıtma gibi geliştirici kullanım durumlarına özgü iş akışları sunar. Buradan başlayalım! Muhtemelen “Topoloji” görünümündesiniz, bu görünüm uygulamaların görsel temsilini sağlar. Değilse, göz atmak için bu görünümü geçin.

1. Bu projeye yeni bir uygulama ekleyelim. Openshift'te yeni bir uygulama eklemenin birkaç yolu vardır.
2. Yeni bir uygulama eklemek için **+Ekle** butonuna tıklayın.
3. Seçenekler arasında **Git Deposu (Git'ten İçe Aktar)** seçeneğini seçin.

Getting started resources [?](#)

Create applications using samples

Choose a code sample to get started creating an application with.

Basic Quarkus →
Basic Spring Boot →

[View all samples](#)

Build with guided documentation

Follow guided documentation to build applications ai

Get started with Quarkus using s2i →
Get started with Spring →

[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

Container images

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

Samples

Create an application from a code sample

4. **Git'ten İçe Aktar** penceresine yönlendirileceksiniz. OpenShift, sizden yalnızca bir girdi alarak bir uygulama dağıtacak: uygulama kaynağı.
5. **Git Repo URL** kutusuna aşağıda belirtilen örnek URL'yi yapıştırın.

<https://github.com/sclorg/nodejs-ex.git>

The screenshot shows the Skills Network OpenShift Lab interface. On the left, there's a sidebar with various developer tools: Developer (+Add), Topology, Search, Builds, Pipelines, Helm, Project, ConfigMaps, Secrets. The 'Project' section is currently selected. At the top, it says 'Project: sn-labs-manvigli' and 'Application: All applications'. The main area is titled 'Import from Git' under a 'Git' section. A red box highlights the 'Git Repo URL *' input field, which contains 'https://github.com/sclorg/nodejs-ex.git' and has a green checkmark icon. Below it, the word 'Validated' is shown. There's also a link to 'Show advanced Git options'. Another red box highlights a green message box stating 'Builder Image detected.' with a note: 'A Builder Image is recommended.' To the right of this message, there's a 'Node.js 16 (UBI 8)' builder image icon with the text 'BUILDER NODEJS' and a blue 'Edit Import Strategy' button with a pencil icon.

Not: Çeşitli builder görüntülerini görmek için İçe Aktarma Stratejisini Düzenle'ye tıklayın. Uygulamamız için Node.js görüntüsünü kullanacağız. Bu görüntünün seçili olduğundan emin olun.

Project: sn-labs-manv1 Application: All applications

Developer

+Add

Topology

Search

Builds

Pipelines

Helm

Project

ConfigMaps

Secrets

Git

Git Repo URL *

<https://github.com/sclorg/nodejs-ex.git>

Validated

Show advanced Git options

Builder Image detected.

A Builder Image is recommended.

Import Strategy

Revert to recommended

Builder Image

Builder Image

Perl PHP NGINX Httpd .NET Go Ruby Python Java

Jboss Eap Xp3 Openjdk11 OpenShift Jboss Eap Xp4 Openjdk11 OpenShift node Node.js

Builder Image version *

16-ubi8

node Node.js 16 (UBI 8)

BUILDER NODEJS

Build and run Node.js 16 applications on UBI 8. For more information about using this builder image, including OpenShift considerations, see <https://github.com/sclorg/s2i-nodejs-container/blob/master/16/README.md>. Sample repository: <https://github.com/sclorg/nodejs-ex.git>

Run command

start

Optional arguments for npm run.

Create Cancel

6. Diğer varsayılan seçenekleri olduğu gibi bırakın. Ardından aşağı kaydırın ve **Oluştur'a** tıklayın.

Topoloji görünümünde, şimdi yeni oluşturduğunuz uygulamayı görmelisiniz.

NOT: Uygulamanın görünmesi birkaç dakika sürecektir. 3 dakika içinde herhangi bir uygulama görmüyorsanız, tarayıcıyı yenleyin.

Skills Network OpenShift Lab

Project: sn-labs-lavanyar Application: All applications

+Add

Topology

Search

Builds

Pipelines

Helm

Project

ConfigMaps

Secrets

nodejs-ex-git

nodejs-ex-git-app

9+ Anna Salai Construction

Search

Not: İlk derlemenin başarılı bir şekilde çalışmasını bekleyin. Beklerken **ImagePullBackOff** ve **ErrImagePull** hatalarını geçici olarak görebilirsiniz.

Uygulamayı web konsolunda görüntüle

Topoloji görünümü, bir uygulamanın birçok önemli kısmına hızlı bağlantılar sağlar:

- Dış daire, uygulama hakkında bilgi alır.
- Node.js logosu bulunan iç daire, Dağıtım hakkında bilgi verir.
- GitHub simgesi, kod deposuna erişim için kullanılır.
- Onay işaretleri, en son yapıyı gösterir (yapı devam ediyorsa dairesel oklar göreceksiniz).
- Bir kutudan çıkan ok, uygulama harici olarak mevcutsa tarayıcıda uygulamayı görüntülemek için kullanılabilir.

Belirli adımları deneyelim:

1. Node.js logosu bulunan iç daireye tıklayarak Dağıtım hakkında bilgi alabilir ve bu Dağıtım ile ilişkili dört kaynağı gözlemleyebilirsiniz: konteynerleştirilmiş uygulamayı çalıştıran bir Pod; uygulamayı bir konteyner görüntüsüne dönüştürmek için s2i stratejisini kullanan bir Build; uygulamayı bir ağ hizmeti olarak açığa çıkarılan bir Service; ve dışarıdan erişilebilir bir ana bilgisayar adı sağlayan bir Route.

Skill Network OpenShift Lab

Developer

+Add

Topology

Monitoring

Search

Builds

Pipelines

Helm

Project

Config Maps

Secrets

Project: Application: all applications

Display Options Filter by Resource Find by name... / i

A node labeled 'node' with a 'nodejs-ex-git' pod underneath it. Another node labeled 'nodejs...it-app' is shown below it.

Not: Lütfen pod'un durumunun 'Çalışıyor' olarak değiştmesini ve Yapının tamamlanmasını bekleyin.

2. Build #1 yazılı satırda Logları Görüntüle seçeneğine tıklayın.

The screenshot shows the OpenShift web interface for the 'nodejs' application. At the top, there's a navigation bar with 'Actions' on the right. Below it, a 'Health Checks' section indicates that the container does not have health checks to ensure the application is running correctly, with a note to 'Add Health Checks'. The main content area has tabs for 'Details', 'Resources' (which is selected), and 'Monitoring'. The 'Pods' section shows one pod named 'nodejs-776fbdd8bd-wplg5' which is 'Running'. The 'Builds' section shows a build named 'nodejs' that is 'Complete' (indicated by a green checkmark) and was completed 'a few seconds ago'. There's a 'Start Build' button and a 'View logs' button. The 'Services' section shows a service named 'nodejs' with port 8080 mapped to Pod Port 8080. The 'Routes' section shows a route named 'nodejs' with a location pointing to a URL: <http://nodejs-sn-labs-...prod-openshift-sa...a45631dc5778dc6371c67d206ba9ae5c-0000.us-east.containers.appdomain.cloud>.

3. Günlükleri okuyarak birkaç önemli tamamlanmış adımı görün. Depo klonlanır, bir Dockerfile oluşturulur, bir görüntü oluşturulur ve görüntü iç kayıt defterine itilir.

The screenshot shows the Skills Network OpenShift Lab interface. On the left, there's a sidebar with various navigation options: Developer (+Add), Topology, Monitoring, Search, Builds, Pipelines, Helm, Project, Config Maps, and Secrets. The main area is titled 'Project' and shows a 'Builds > Build Details' section for a build named 'nodejs-ex-git-1'. The status is 'Complete'. Below this, there are tabs for 'Details', 'YAML', 'Environment', and 'Logs'. The 'Logs' tab is highlighted with a red box. A message 'Log stream ended.' is displayed above the log output. The log output itself shows 89 lines of command-line output from the build process, including cloning a GitHub repository, copying blobs, generating a Dockerfile, and performing several build steps (STEP 1-5).

```
Cloning "https://github.com/sclorg/nodejs-ex.git" ...
Commit: b27dd9e9ffe53f76a1407d4fe357cf64c8a0ac6f (chore: remove the reference to node 10)
Author: Lucas Holmquist <lholmqui@redhat.com>
Date: Thu Oct 21 13:29:44 2021 -0400
Caching blobs under "/var/cache/blobs".
Getting image source signatures
Copying blob sha256:8a4ce2d3973a8b9ccb73fc982adbffef274e95cb2548098e755b1df847aca0de
Copying blob sha256:71f6d04e5352b855df99a734fa3df8b4ce5c1e73583756a38dae7f0365d48f43
Copying blob sha256:ad62d8acaeb8e10bb459e0fb98054b6cd0769fe4d0485daf504967c8ffcc2df
Copying blob sha256:a7f628200d73a511e8ca006262c54054fd4fb3d4c6260bc0b7a770e402891ef8
Copying config sha256:ac1cc3129e2aab2b495b178f05076d357869f0dab48753cf8593bd24e1640dc8
Writing manifest to image destination
Storing signatures
Generating dockerfile with builder image image-registry.openshift-image-registry.svc:5000/openshift/nodejs-ex:sha256:0442577b059...
Adding transient rw bind mount for /run/secrets/rhsm
Adding transient rw bind mount for /run/secrets/etc-pki-entitlement
Adding transient rw bind mount for /run/secrets/redhat.repo
STEP 1: FROM image-registry.openshift-image-registry.svc:5000/openshift/nodejs@sha256:0442577b059...
STEP 2: LABEL "io.openshift.build.commit.author"="Lucas Holmquist <lholmqui@redhat.com>" ...
STEP 3: ENV OPENSHIFT_BUILD_NAME="nodejs-ex-git-1" OPENSHIFT_BUILD_NAMESPACE="sn-labs-samaahs"
STEP 4: USER root
STEP 5: COPY upload/src /tmp/src
```

4. Bu Yapının **Ayrıntılar** sekmesine tıklayın.
5. Ardından, en alta bulunan **Sahibi** altındaki BC (Build Config) yazılı bağlantıya tıklayın.

The screenshot shows the 'Build Details' page for a build named 'nodejs-ex-git-1'. The 'Details' tab is selected. The 'Annotations' section is highlighted with a red box, showing various Kubernetes labels. The 'Owner' section is also highlighted with a red box, showing 'BC nodejs-ex-git'.

Annotations

3 Annotations [Edit](#)

Triggered By

Image change

Started

Apr 11, 4:17 pm

Created At

Apr 11, 4:17 pm

Owner

BC nodejs-ex-git

6. **Ayrıntılar** ve **YAML** sekmlerine baktığınızda, bu modülde konuştuğumuz birçok kavramı göreceksiniz: tetikleyiciler, yapı stratejisi, web kancaları ve daha fazlası.

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

Environment Logs Events

```
1 kind: Build
2 apiVersion: build.openshift.io/v1
3 metadata:
4   annotations:
5     openshift.io/build-config.name: nodejs
6     openshift.io/build.number: '1'
7     openshift.io/build.pod-name: nodejs-1-build
8   resourceVersion: '334028934'
9   name: nodejs-1
10  uid: efb13c7a-6803-488c-b58a-a1e2cd554401
11  creationTimestamp: '2022-03-31T08:57:35Z'
12  generation: 2
13  namespace: sn-labs
14  ownerReferences:
15    - apiVersion: build.openshift.io/v1
16      kind: BuildConfig
17      name: nodejs
18      uid: 6dc83d68-d0da-46fe-a6c6-331dfa841fc7
19      controller: true
20  labels:
21    app: nodejs
22    app.kubernetes.io/part-of: nodejs-app
23    app.kubernetes.io/instance: nodejs
24    openshift.io/build-config.name: nodejs
25    app.kubernetes.io/component: nodejs
26    openshift.io/build.start-policy: Serial
27    buildconfig: nodejs
28    app.openshift.io/runtime: nodejs
29    app.kubernetes.io/name: nodejs
30    app.openshift.io/runtime-version: 14-ubi7
31  spec:
32    nodeSelector: null
33    output:
34      to:
35        kind: ImageStreamTag
36        name: 'nodejs:latest'
37        pushSecret:
38          name: builder-dockercfg-j9s2b
```

7. Ayrıntılar sekmesinde, Çıktı altındaki IST (ImageStreamTag) yazılı bağlantıya tıklayın.

The screenshot shows the Skills Network OpenShift Lab interface. On the left sidebar, under the 'Developer' section, the 'Builds' option is selected. In the main content area, the 'Build Config Details' page for 'nodejs-ex-git' is displayed. The 'Details' tab is active and highlighted with a red box. Other tabs include 'YAML', 'Builds', 'Environment', and 'Events'. The page shows the following details:

- Name:** nodejs-ex-git
- Namespace:** sn-labs-samaahs
- Labels:** app=nodejs-ex-git, app.kubernetes.io/component=nodejs-ex-git, app.kubernetes.io/instance=nodejs-ex-git, app.kubernetes.io/name=nodejs, app.kubernetes.io/part-of=nodejs-ex-git-app, app.openshift.io/runtime=nodejs, app.openshift.io/runtime-version=14-ubi7
- Annotations:** 3 Annotations
- Created At:** Apr 11, 4:17 pm
- Owner:** No owner

8. Artık yapının çıktısı olarak oluşturulan ImageStreamTag'yi görebilirsiniz. Bu ImageStreamTag'nin işaret ettiği iç kayıt defterindeki görüntütüyü görmek için **Geçmiş** sekmesine tıklayın.

IST nodejs-ex-git:latest

Details YAML History

Apr 11, 4:19 pm

IST nodejs-ex-git:latest
from image-registry.openshift-image-registry.svc:5000/sn-labs-XXXXXXXXXX/nodejs-ex-git
sha256:39bf8ad2306e9a755a75abace734e466c238b6f985f68732e9b859a215f0ac01

9. Topoloji görünümüne geri dönün ve Dağıtım bilgilerinize tıklayın. OpenShift'in sizin için otomatik olarak oluşturduğu Route'a tıklayın. Bu, uygulamayı tarayıcıda açacaktır.

Not: Lütfen bu URL'yi not edin, çünkü bir sonraki bölümde kullanılacaktır.

The screenshot shows the Kubernetes dashboard interface. The top navigation bar includes 'Project' dropdown, 'Application: all applications' dropdown, 'Display Options' dropdown, 'Filter by Resource' dropdown, a search bar 'Find by name...', and a help icon. The left sidebar contains links: '+Add', 'Topology' (which is highlighted with a red box), 'Monitoring', 'Search', 'Builds', 'Pipelines', 'Helm', 'Project', 'Config Maps', and 'Secrets'. The main content area displays a network topology diagram. A central node is labeled 'node' with a blue and white circular icon. Two other nodes are shown: one labeled 'nodejs-ex-git' with a blue circle and a green checkmark, and another labeled 'nodejs...it-app' with a green circle and a white letter 'A'. Below the diagram are several small icons: a magnifying glass, a search icon, a refresh icon, a double arrow icon, and two buttons labeled 'x 1' and 'x 2'.

Otomatik Ölçeklendirme nodejs-ex-git Uygulaması

Artık nodejs-ex-git uygulaması başarıyla çalışmasına göre, gelen yükü karşılayabilmesi için yatay pod otomatik ölçekleyici (HPA) ayarlayalım. nodejs-ex-git uygulamasını bir tarayıcı sekmesinde açık tutmayı unutmayın, böylece talepleri yapmaya ve kaynak tüketmeye devam edebilir ve başarıyla otomatik ölçeklendirilir.

Öncelikle, çalışacak konteynerler için kaynak talepleri ve limitleri ayarlamamız gerekiyor. Bir konteyner CPU veya bellek gibi bir kaynak talep ettiğinde, Kubernetes bunu yalnızca o kaynağı verebilecek bir düğümde planlar. Öte yandan, limitler bir konteynerin belirli bir mikardan fazla kaynak tüketmesini engeller.

Bu durumda, 3 milikoro CPU ve 40 MB RAM talep edeceğiz. Konteynerleri 30 milikoro ve 100 MB ile sınırlayacağız. Bu sayılar, uygulamanın ölçeklenmesini sağlamak için uydurulmuştur.

1. Topoloji görünümünden, nodejs-ex-git Dağıtımına tıklayın. Ardından, İşlemler > Dağıtımları Düzenle'ye tıklayın.

The screenshot shows the Kubernetes UI for the deployment named 'nodejs-ex-git'. The main interface includes tabs for 'Details', 'Resources' (which is selected), and 'Monitoring'. On the left, there's a pod icon for 'nodejs-ex-git' and another for 'opensh...onsole'. The right side shows sections for 'Pods', 'Builds', and 'Services'. A context menu is open over the deployment, with the 'Edit Deployment' option highlighted by a red box.

2. template.spec.containers bölümünde, resources: {} kısmını bulun. Bunu aşağıdaki metinle değiştirin. YAML'nın katı girinti kullandığını unutmayın, bu nedenle boşlukların doğru olduğundan emin olun.

```
resources:
  limits:
    cpu: 30m
    memory: 100Mi
  requests:
    cpu: 3m
    memory: 40Mi
```

D nodejs-ex-git
[Details](#) [YAML](#) [Replica Sets](#) [Pods](#) [Environment](#) [Events](#)

```

140     creationTimestamp: null
141     labels:
142       app: nodejs-ex-git
143       deploymentconfig: nodejs-ex-git
144     spec:
145       containers:
146         - name: nodejs-ex-git
147           image: >-
148             image-registry.openshift-image-registry.svc:5000/sn-labs-nodejs-ex-git@sha256:...
149           ports:
150             - containerPort: 8080
151               protocol: TCP
152             resources:
153               limits:
154                 cpu: 30m
155                 memory: 100Mi
156               requests:
157                 cpu: 3m
158                 memory: 40Mi
159             terminationMessagePath: /dev/termination-log
160             terminationMessagePolicy: File
161             imagePullPolicy: Always
162             restartPolicy: Always
163             terminationGracePeriodSeconds: 30
164             dnsPolicy: ClusterFirst
165             securityContext: {}
166             schedulerName: default-scheduler
167       strategy:
168         type: RollingUpdate
169       ...

```

[Save](#)[Reload](#)[Cancel](#)

3. Kaydet'e tıklayın.

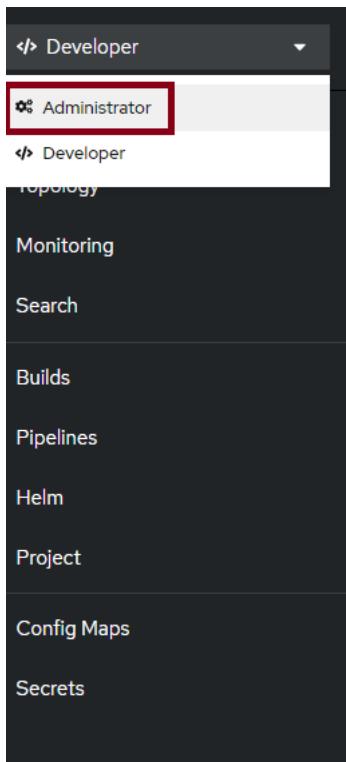
```

140     creationTimestamp: null
141     labels:
142       app: nodejs-ex-git
143       deploymentconfig: nodejs-ex-git
144     spec:
145       containers:
146         - name: nodejs-ex-git
147           image: >-
148             image-registry.openshift-image-registry.svc:5000/sn-labs-nodejs-ex-git@sha256:...
149           ports:
150             - containerPort: 8080
151               protocol: TCP
152             resources:
153               limits:
154                 cpu: 30m
155                 memory: 100Mi
156               requests:
157                 cpu: 3m
158                 memory: 40Mi
159             terminationMessagePath: /dev/termination-log
160             terminationMessagePolicy: File
161             imagePullPolicy: Always
162             restartPolicy: Always
163             terminationGracePeriodSeconds: 30
164             dnsPolicy: ClusterFirst
165             securityContext: {}
166             schedulerName: default-scheduler
167       strategy:
168         type: RollingUpdate
169       ...

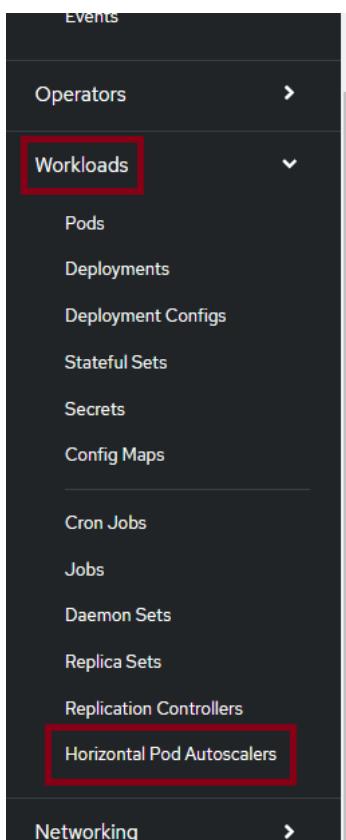
```

[Save](#)[Reload](#)[Cancel](#)

4. Yöneticilik perspektifine geçin.



5. İş Yükleri > Yatay Pod Otomatik Ölçekleyicileri'ni seçin.



6. Yatay Pod Otomatik Ölçekleyici Oluştur'a tıklayın.

Horizontal Pod Autoscalers

No Horizontal Pod Autoscalers Found

7. Aşağıdaki YAML'ı editöre yapıştırın.

```
apiVersion: autoscaling/v2
kind: HorizontalPodAutoscaler
metadata:
  name: nodejs-ex-git-hpa
spec:
  scaleTargetRef:
    apiVersion: apps/v1
    kind: Deployment
    name: nodejs-ex-git
  minReplicas: 1
  maxReplicas: 3
  metrics:
    - type: Resource
      resource:
        name: cpu
        target:
          type: Utilization
          averageUtilization: 10
```

```
1  apiVersion: autoscaling/v2
2  kind: HorizontalPodAutoscaler
3  metadata:
4    name: nodejs-ex-git-hpa
5  spec:
6    scaleTargetRef:
7      apiVersion: apps/v1
8      kind: Deployment
9      name: nodejs-ex-git
10     minReplicas: 1
11     maxReplicas: 3
12     metrics:
13       - type: Resource
14         resource:
15           name: cpu
16           target:
17             type: utilization
18             averageUtilization: 10|
```

Bu HPA, CPU kullanımına dayalı ölçeklendirme yapacağımızı gösteriyor. Genel olarak, CPU kullanımınız %50-90 aralığında olduğunda ölçeklendirme yapmak istersiniz. Bu örnek için, uygulamanın ölçeklendirme gerektirme olasılığını artırmak amacıyla %10 kullanacağız. minReplicas ve maxReplicas alanları, yük durumuna bağlı olarak Dağıtımın her zaman bir ila üç kopya arasında olmasını belirtir.

8. Oluştur'a tıklayın



```

1 apiVersion: autoscaling/v2
2 kind: HorizontalPodAutoscaler
3 metadata:
4   name: nodejs-ex-git-hpa
5 spec:
6   scaleTargetRef:
7     apiVersion: apps/v1
8     kind: Deployment
9     name: nodejs-ex-git
10    minReplicas: 1
11    maxReplicas: 3
12    metrics:
13      - type: Resource
14        resource:
15          name: cpu
16          target:
17            type: Utilization
18              averageUtilization: 10

```

Create**Cancel****Download**

9. nodejs-ex-git üzerindeki yükü artırmak ve Otomatik Ölçeklendirmeyi görüntülemek için Theia'daki terminalde aşağıdaki komutu çalıştırın:

```
for i in $(seq 1000); do curl -s -L <your app URL> & done
```

Not: <your app URL> ile önceki bölümdeki 9. Adımda elde ettiğiniz URL'yi değiştirin.

```
theia@theiaopenshift-nikeshkr:/home/project$ for i in $(seq 1000); do curl labs-nikeshkr.labs-prod-openshift-san-a45631dc5778dc6371c67d206ba9ae5c-0000 loud/ & done
```

Komut, başarılı yük oluşturma işlemini gösteren aşağıdaki çıktıyı vermeye devam edecektir:

```
[> theia@theiaopenshift-nikeshkr: /home/project X
```

```
[965] 2531
[966] 2533
[967] 2536
[968] 2537
[969] 2538
[970] 2539
[971] 2540
[972] 2541
[973] 2543
[974] 2553
[975] 2554
[976] 2556
[977] 2558
[978] 2559
[979] 2560
[980] 2561
[981] 2562
[982] 2563
[983] 2564
[984] 2565
[985] 2572
[986] 2575
[987] 2576
[988] 2577
[989] 2578
[990] 2579
[991] 2580
[992] 2581
[993] 2582
[994] 2583
[995] 2586
[996] 2587
[997] 2588
[998] 2589
[999] 2590
[1000] 2595
```

Not: Autoscaling'i, tarayıcınızda doğrudan uygulama URL'ni çalıştırarak da doğrulayabilirsiniz. Uygulama arayüzüne bazı meyve isimleri ve miktarlarını ekleyin ve 'Kaydet' butonuna tıklayın. Eklenen meyveler kullanıcı arayüzünde görünmese de, bu işlem bir yük değişikliği tetikleyecek ve zamanla pod'ların 3'e autoscale olmasına neden olacaktır.

10. Scale Target altında nodejs-ex-git'e tıklayın.

Horizontal Pod AutoScalers

Name	Labels	Scale Target	Min Pods
HPA nodejs-ex-git-hpa	No labels	nodejs-ex-git	1

11. Beklerseniz, Hem Mevcut Replicalar hem de İstenen Replicalar üçe çıkacaktır. Bu, HPA'nın pod'ların maksimum sayısı olan üçe ölçeklenmesi için yeterli yük tespit etmesindendir. Ayrıca Son Ölçekleme Zamanı ile mevcut ve hedef CPU kullanımını da görebilirsiniz. Hedef, belirlediğimiz gibi açıkça %1'dir. Ölçeklenmenin tetiklenmesinin birkaç dakika sürebileceğini unutmayın.

D nodejs-ex-git

Details YAML ReplicaSets Pods Environment Events

Deployment details



Name

nodejs-ex-git

Update strategy

RollingUpdate

Namespace

NS [REDACTED]

Max unavailable

25% of 3 pods

Labels

Edit

app=nodejs-ex-git app.kubernetes.io/component=nodejs-ex-git app.kubernetes.io/instance=nodejs-ex-git
app.kubernetes.io/name=nodejs-ex-git app.kubernetes.io/part-of=nodejs-ex-git-app app.openshift.io/runtime=nodejs
app.openshift.io/runtime-version=16-ubi8

Max surge

25% greater than 3 pods

Progress deadline seconds

600 seconds

Vay! OpenShift sizin adınıza gerçekten inanılmaz bir iş yaptı. Tek ihtiyacı olan bir kod deposuydu ve bu kodu bir konteyner imajına dönüştürmeyi, o imajı bir kayıt defterine itmeyi, o imajı referans alan bir Deployment oluşturmayı ve uygulamayı bir hostname ile internete açmayı başardı.

Tebrikler! Bu kursun dördüncü modülündeki laboratuvarı tamamladınız.

© IBM Corporation. Tüm hakları saklıdır.