Kubernetes'e Giriş

Hedefler

Bu laboratuvar sırasında şunları yapacaksınız:

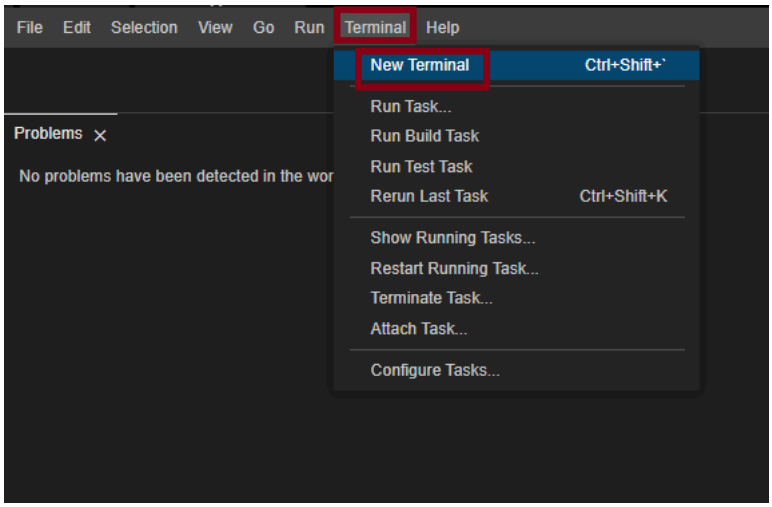
- `kubectl` CLI'yi kullanmak
- Bir Kubernetes Pod'u oluşturmak
- Bir Kubernetes Deployment'ı oluşturmak
- Belirli sayıda replikayı koruyan bir ReplicaSet oluşturmak
- Kubernetes yük dengelemenin nasıl çalıştığını görmek

> **Not: Lütfen laboratuvarı tek bir oturumda, ara vermeden tamamlayın çünkü laboratuvar çevrimdışı moda geçebilir ve hatalara yol açabilir. Laboratuvar sürecinde herhangi bir sorun/hata ile karşılaşırsanız, lütfen laboratuvar ortamından çıkış yapın. Ardından sistem önbelleğinizi ve çerezlerinizi temizleyin ve laboratuvarı tamamlamayı deneyin.**

Ortamı ve komut satırı araçlarını doğrulayın

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

   Not: Eğer terminal zaten görünüyorsa bu adımı atlayın.



2. `kubectl` CLI'nin yüklü olduğunu doğrulayın.

```
kubectl version
```

Aşağıdaki çıktıyı görmelisiniz, ancak sürümler farklı olabilir:

```
theia@theiadocker /home/project$ kubectl version
Client Version: version.Info{Major:"1", Minor:"22", GitVersion:"v1.22.3", GitCommit:"c92036820499fedefec0f847e2054d824aea6cd1", GitTr
"2021-10-27T18:41:28Z", GoVersion:"go1.16.9", Compiler:"gc", Platform:"linux/amd64"}
Server Version: version.Info{Major:"1", Minor:"21", GitVersion:"v1.21.11+IKS", GitCommit:"7d30e1c191e870ff995f9b6ba21452d0325db2ad",
Date:"2022-03-17T16:12:51Z", GoVersion:"go1.16.15", Compiler:"gc", Platform:"linux/amd64"}
theia@theiadocker /home/project$
```

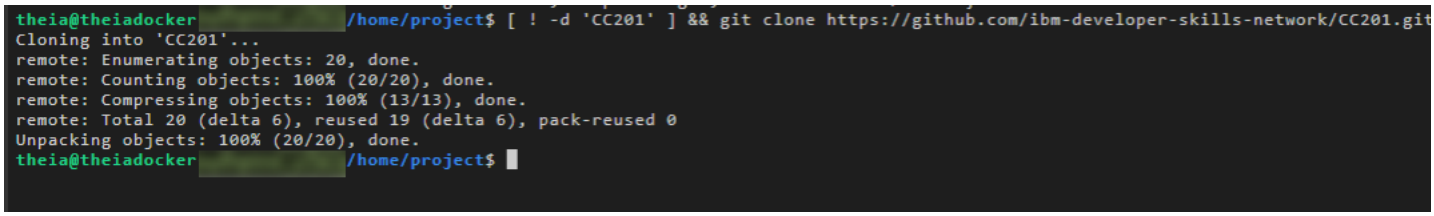
3. Proje klasörünüze geçin.

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

```
cd /home/project
```

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

```
[ ! -d 'CC201' ]  && git clone https://github.com/ibm-developer-skills-network/abahi-CC_201_labs.git CC201
```

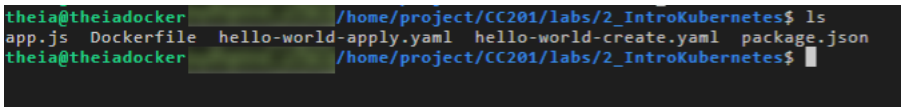


5. Aşağıdaki komutu çalıştırarak bu laboratuvarın dizinine geçin. `cd`, çalışma/mevcut dizini belirtilen isimle olan dizine değiştirecektir, bu durumda **CC201/labs/2_IntroKubernetes**.

```
cd CC201/labs/2_IntroKubernetes/
```

6. Bu dizinin içeriğini listeleyin ve bu laboratuvar için olan artefaktları görün.

```
ls
```



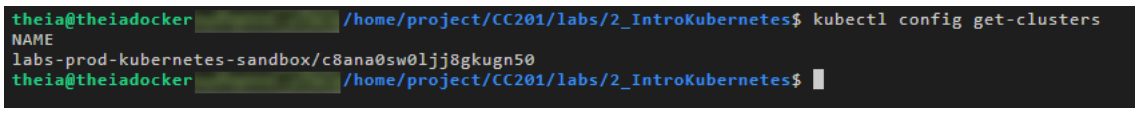
`kubectl` CLI Kullanımı

Kubernetes ad alanlarının bir küme sanallaştırmanıza olanak tanıdığını hatırlayın. Bir Kubernetes kümesinde zaten bir ad alanına erişiminiz var ve `kubectl` bu küme ve ad alanını hedef alacak şekilde ayarlanmıştır.

Şimdi bazı temel `kubectl` komutlarına bakalım.

1. `kubectl`, uygun küme hedefini belirlemek için yapılandırma gerektirir. Aşağıdaki komutla küme bilgilerini alın:

```
kubectl config get-clusters
```



2. Bir `kubectl` bağlamı, bir küme, bir kullanıcı ve bir ad alanı dahil olmak üzere bir dizi erişim parametresidir. Mevcut bağlamınızı aşağıdaki komutla görüntüleyin:

```
kubectl config get-contexts
```

```
theia@theiadocker-[redacted] /home/project/CC201/labs/2_IntroKubernetes$ kubectl config get-contexts
CURRENT   NAME                    CLUSTER                                          AUTHINFO        NAMESPACE
*         [redacted]-context      labs-prod-kubernetes-sandbox/c8ana0sw0ljj8gkugn50 [redacted]      sn-labs-[redacted]
theia@theiadocker-[redacted] /home/project/CC201/labs/2_IntroKubernetes$
```

3. Namespace'inizdeki tüm Pod'ları listeleyin. Eğer bu sizin için yeni bir oturumsa, hiçbir Pod göremeyeceksiniz.

```
kubectl get pods
```

```
theia@theiadocker-[redacted] /home/project/CC201/labs/2_IntroKubernetes$ kubectl get pods
No resources found in sn-labs-[redacted] namespace.
theia@theiadocker-[redacted] /home/project/CC201/labs/2_IntroKubernetes$
```

Bir Pod Oluşturma İmperatif Komut ile

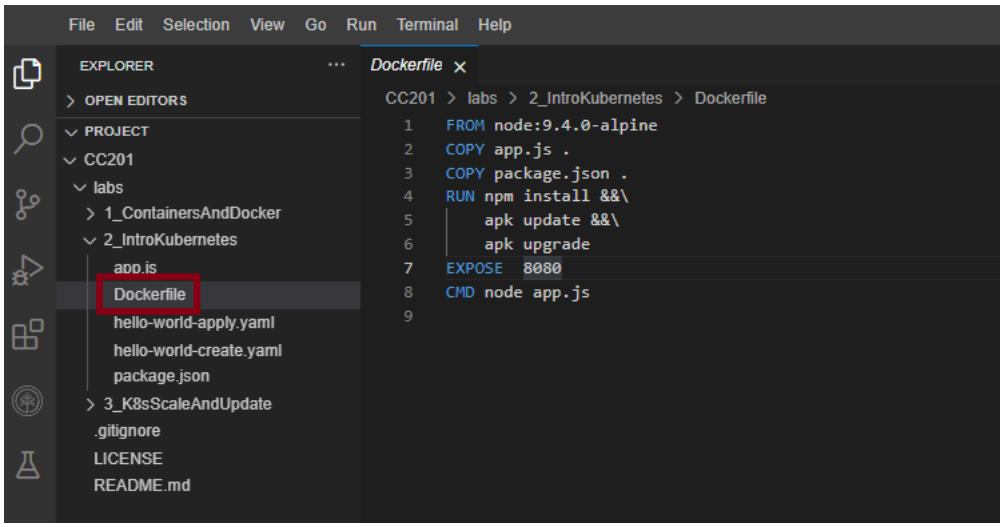
Artık ilk Pod'unuzu oluşturma zamanı. Bu Pod, son laboratuvarınızda IBM Cloud Container Registry'ye yüklediğiniz `hello-world` imajını çalıştıracak. Bu modülün videolarında açıklandığı gibi, bir Pod'u imperatif veya deklaratif olarak oluşturabilirsiniz. Önce imperatif olarak yapalım.

1. İlerleyen komutlarda kullanılabilmesi için ad alanınızı bir ortam değişkeni olarak dışa aktarın.

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

```
theia@theiadocker-[redacted] /home/project/CC201/labs/2_IntroKubernetes$ export MY_NAMESPACE=sn-labs-$USERNAME
theia@theiadocker-[redacted] /home/project/CC201/labs/2_IntroKubernetes$
```

2. Pencerenin sol tarafındaki Keşfet simgesine (bir kağıt parçası gibi görünür) tıklayın ve ardından bu laboratuvar için dizine gidin: `CC201 > labs > 2_IntroKubernetes`. `Dockerfile` dosyasına tıklayın. Bu, imajımızı oluşturmak için kullanılacak dosyadır.



3. İmajı tekrar oluşturun ve ittirin, çünkü ilk laboratuvarı tamamladığınızdan beri otomatik olarak silinmiş olabilir.

```
docker build -t us.icr.io/$MY_NAMESPACE/hello-world:1 . && docker push us.icr.io/$MY_NAMESPACE/hello-world:1
```

```
theia@theiadocker-[redacted]/home/project/CC201/labs/2_IntroKubernetes$ docker build -t us.icr.io/$MY_NAMESPACE/hello-world:1 . &&
ACE/hello-world:1
Sending build context to Docker daemon  6.656kB
Step 1/6 : FROM node:9.4.0-alpine
9.4.0-alpine: Pulling from library/node
605ce1bd3f31: Pull complete
fe58b30348fe: Pull complete
46ef8987ccbd: Pull complete
Digest: sha256:9cd67a00ed111285460a83847720132204185e9321ec35dacec0d8b9bf674adf
Status: Downloaded newer image for node:9.4.0-alpine
 ---> b5f94997f35f
Step 2/6 : COPY app.js .
 ---> 28350e465969
Step 3/6 : COPY package.json .
 ---> 45bf6db4af5f
Step 4/6 : RUN npm install &&    apk update &&    apk upgrade
 ---> Running in a37db9ced1bc
npm notice created a lockfile as package-lock.json. You should commit this file.
npm WARN hello-world-demo@0.0.1 No repository field.
npm WARN hello-world-demo@0.0.1 No license field.

added 50 packages in 2.085s
fetch http://dl-cdn.alpinelinux.org/alpine/v3.6/main/x86_64/APKINDEX.tar.gz
fetch http://dl-cdn.alpinelinux.org/alpine/v3.6/community/x86_64/APKINDEX.tar.gz
v3.6.5-44-gda55e27396 [http://dl-cdn.alpinelinux.org/alpine/v3.6/main]
v3.6.5-34-gf0ba0b43d5 [http://dl-cdn.alpinelinux.org/alpine/v3.6/community]
OK: 8448 distinct packages available
Upgrading critical system libraries and apk-tools:
(1/1) Upgrading apk-tools (2.7.5-r0 -> 2.7.6-r0)
Executing busybox-1.26.2-r9.trigger
Continuing the upgrade transaction with new apk-tools:
(1/7) Upgrading musl (1.1.16-r14 -> 1.1.16-r15)
(2/7) Upgrading busybox (1.26.2-r9 -> 1.26.2-r11)
Executing busybox-1.26.2-r11.post-upgrade
(3/7) Upgrading libressl2.5-libcrypto (2.5.5-r0 -> 2.5.5-r2)
(4/7) Upgrading libressl2.5-libssl (2.5.5-r0 -> 2.5.5-r2)
(5/7) Installing libressl2.5-libtls (2.5.5-r2)
```

4. `hello-world` imajını Kubernetes'te bir konteyner olarak çalıştırın.

```
kubectl run hello-world --image us.icr.io/$MY_NAMESPACE/hello-world:1 --overrides='{"spec":{"imagePullSecrets":[{"name":"icr"}],"containers":[{"
```

`--overrides` seçeneği, bu görüntüyü IBM Cloud Container Registry'den çekmek için gerekli kimlik bilgilerini belirtmemizi sağlar. Bunun, Kubernetes'e açıkça ne yapması gerektiğini söylediğimiz için zorunlu bir komut olduğunu unutmayın: `hello-world` çalıştır.

```
theia@theiadocker-rajashreep:/home/project/CC201/labs/2_IntroKubernetes$ kubectl run hello-world --imag
e us.icr.io/$MY_NAMESPACE/hello-world:1 --overrides='{"spec":{"imagePullSecrets":[{"name":"icr"}],"cont
ainers":[{"name":"hello-world","image":"us.icr.io/'$MY_NAMESPACE'/hello-world:1","securityContext":{"al
lowPrivilegeEscalation":false,"runAsNonRoot":true,"runAsUser":999,"capabilities":{"drop":["ALL"]},"secc
ompProfile":{"type":"RuntimeDefault"}}}]}}'
pod/hello-world created
```

5. Ad alanınızdaki Pod'ları listeleyin.

```
kubectl get pods
```

```
theia@theiadocker-[redacted]/home/project/CC201/labs/2_IntroKubernetes$ kubectl get pods
NAME          READY   STATUS    RESTARTS   AGE
hello-world   1/1     Running   0          34s
theia@theiadocker-[redacted]/home/project/CC201/labs/2_IntroKubernetes$
```

Harika, önceki komut gerçekten bizim için bir Pod oluşturdu. Bu Pod'a otomatik olarak oluşturulmuş bir isim verildiğini görebilirsiniz.

Kaynağın daha fazla detayı için çıktıda geniş seçeneklerini de belirtebilirsiniz.

```
kubectl get pods -o wide
```

```
theia@theiadocker-                /home/project/CC201/labs/2_IntroKubernetes$ kubectl get pods -o wide
NAME          READY   STATUS    RESTARTS   AGE   IP              NODE           NOMINATED NODE   READINESS GATES
hello-world   1/1     Running   0          59s   172.17.183.177   10.241.64.24   <none>           <none>
theia@theiadocker-                /home/project/CC201/labs/2_IntroKubernetes$
```

6. Pod'u daha fazla detay almak için tanımlayın.

```
kubectl describe pod hello-world
```

```
theia@theiadocker-                /home/project/CC201/labs/2_IntroKubernetes$ kubectl describe pod hello-world
Name:                 hello-world
Namespace:            sn-labs-
Priority:             1
Priority Class Name:  normal
Node:                 10.241.64.24/10.241.64.24
Start Time:           Fri, 08 Apr 2022 05:15:40 +0000
Labels:               run=hello-world
Annotations:          cni.projectcalico.org/containerID: c89fd419d56a582514d497f0b01b939cf745343036e9a45f135235e7d5bc528e
                      cni.projectcalico.org/podIP: 172.17.183.177/32
                      cni.projectcalico.org/podIPs: 172.17.183.177/32
                      kubernetes.io/limit-ranger:
                        LimitRanger plugin set: cpu, ephemeral-storage, memory request for container hello-world; cpu, ephemeral-sto
                      kubernetes.io/psp: ibm-privileged-psp
Status:               Running
IP:                   172.17.183.177
IPs:
  IP:  172.17.183.177
Containers:
  hello-world:
    Container ID:   containerd://31c934f489c232a36729b3e3f013a5619f11fc8f95ee8a1007f9f540dc4d420a
    Image:          us.icr.io/sn-labs-                /hello-world:1
    Image ID:       us.icr.io/sn-labs-                /hello-world@sha256:a04a56181ae9136e4b7033d5284ce9d68fe812c21b28592ffb292d8b496b6b
    Port:           <none>
    Host Port:      <none>
    State:          Running
      Started:      Fri, 08 Apr 2022 05:15:46 +0000
    Ready:          True
    Restart Count:  0
    Limits:
      cpu:                500m
      ephemeral-storage:  5Gi
      memory:             512Mi
    Requests:
      cpu:                200m
      ephemeral-storage:  512Mi
      memory:             128Mi
    Environment:          <none>
    Mounts:
      /var/run/secrets/kubernetes.io/serviceaccount from kube-api-access-bjdzp (ro)
Conditions:
  Type              Status
```

Not: Çıktı, **Namespace, Pod Adı, IP adresi, pod'un çalışmaya başladığı zaman** gibi pod parametrelerini ve ayrıca **konteyner ID'si, görüntü adı ve ID'si, çalışma durumu ve bellek/CPU sınırları** gibi konteyner parametrelerini gösterir.

7. Pod'u sil.

```
kubectl delete pod hello-world
```

```
theia@theiadocker-                /home/project/CC201/labs/2_IntroKubernetes$ kubectl delete pod hello-world
pod "hello-world" deleted
theia@theiadocker-                /home/project/CC201/labs/2_IntroKubernetes$
```

Bu komut, pod'un silinmesi için biraz zaman alır. Terminal istemi tekrar görünene kadar lütfen bekleyin.

8. Hiç pod olmadığını doğrulamak için Pod'ları listeleyin.

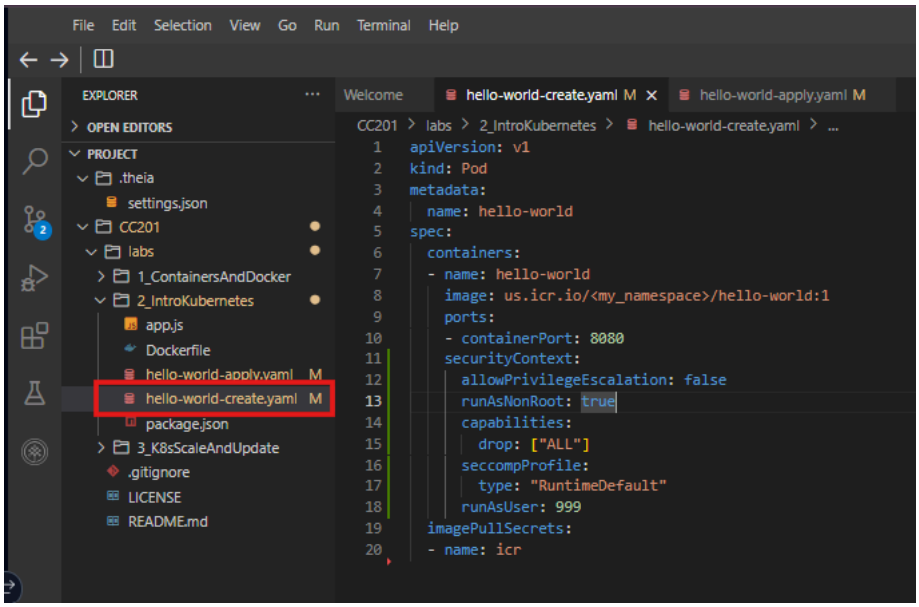
```
kubectl get pods
```

```
theia@theiadocker-[redacted] /home/project/CC201/labs/2_IntroKubernetes$ kubectl get pods
No resources found in sn-labs-[redacted] namespace.
theia@theiadocker-[redacted] /home/project/CC201/labs/2_IntroKubernetes$
```

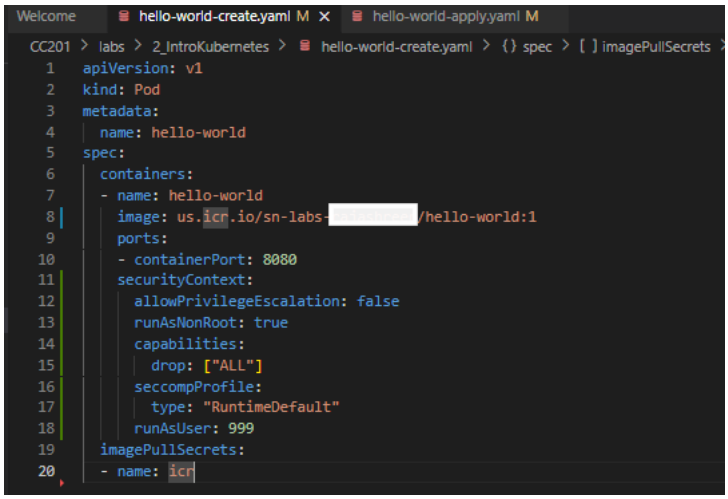
Zorlayıcı nesne yapılandırması ile bir Pod oluştur

Zorlayıcı nesne yapılandırması, bir yapılandırma dosyası kullanarak alınacak eylemi belirterek nesneleri oluşturmanıza olanak tanır (örneğin, oluştur, güncelle, sil). Bu dizinde size sağlanan bir yapılandırma dosyası, `hello-world-create.yaml`'dır.

1. Yapılandırma dosyasını görüntülemek ve düzenlemek için Explorer'ı kullanın. Pencerenin sol tarafındaki Explorer simgesine (bir kağıt parçasına benziyor) tıklayın ve ardından bu laboratuvar için dizine gidin: `CC201 > labs > 2_IntroKubernetes`. Yapılandırma dosyasını görüntülemek için `hello-world-create.yaml`'ya tıklayın.



2. `hello-world-create.yaml`'yı düzenlemek için Explorer'ı kullanın. `<my_namespace>` yazan yere ad alanınızı eklemeniz gerekiyor. İşlemi tamamladığınızda dosyayı kaydetmeyi unutmayın.



3. Sağlanan yapılandırma dosyasını kullanarak zorlayıcı bir şekilde bir Pod oluşturun.

```
kubectl create -f hello-world-create.yaml
```

Not edin ki bu gerçekten zorunludur, çünkü Kubernetes'e dosyada tanımlanan kaynakları *oluşturması* için açıkça söylediniz.

```
theia@theiadocker-[redacted] /home/project/CC201/labs/2_IntroKubernetes$ kubectl create -f hello-world-create.yaml
pod/hello-world created
theia@theiadocker-[redacted] /home/project/CC201/labs/2_IntroKubernetes$
```

4. Ad alanınızdaki Pod'ları listeleyin.

```
kubectl get pods
```

```
theia@theiadocker-[redacted] /home/project/CC201/labs/2_IntroKubernetes$ kubectl get pods
NAME          READY   STATUS    RESTARTS   AGE
hello-world   1/1     Running   0          17s
theia@theiadocker-[redacted] /home/project/CC201/labs/2_IntroKubernetes$
```

5. Pod'u sil.

```
kubectl delete pod hello-world
```

```
theia@theiadocker-[redacted] /home/project/CC201/labs/2_IntroKubernetes$ kubectl delete pod hello-world
pod "hello-world" deleted
theia@theiadocker[redacted] /home/project/CC201/labs/2_IntroKubernetes$
```

Bu komut, pod'un silinmesini gerçekleştirmek için biraz zaman alır. Terminal istemi tekrar görünene kadar lütfen bekleyin.

6. Hiç pod olmadığını doğrulamak için Pod'ları listeleyin.

```
kubectl get pods
```

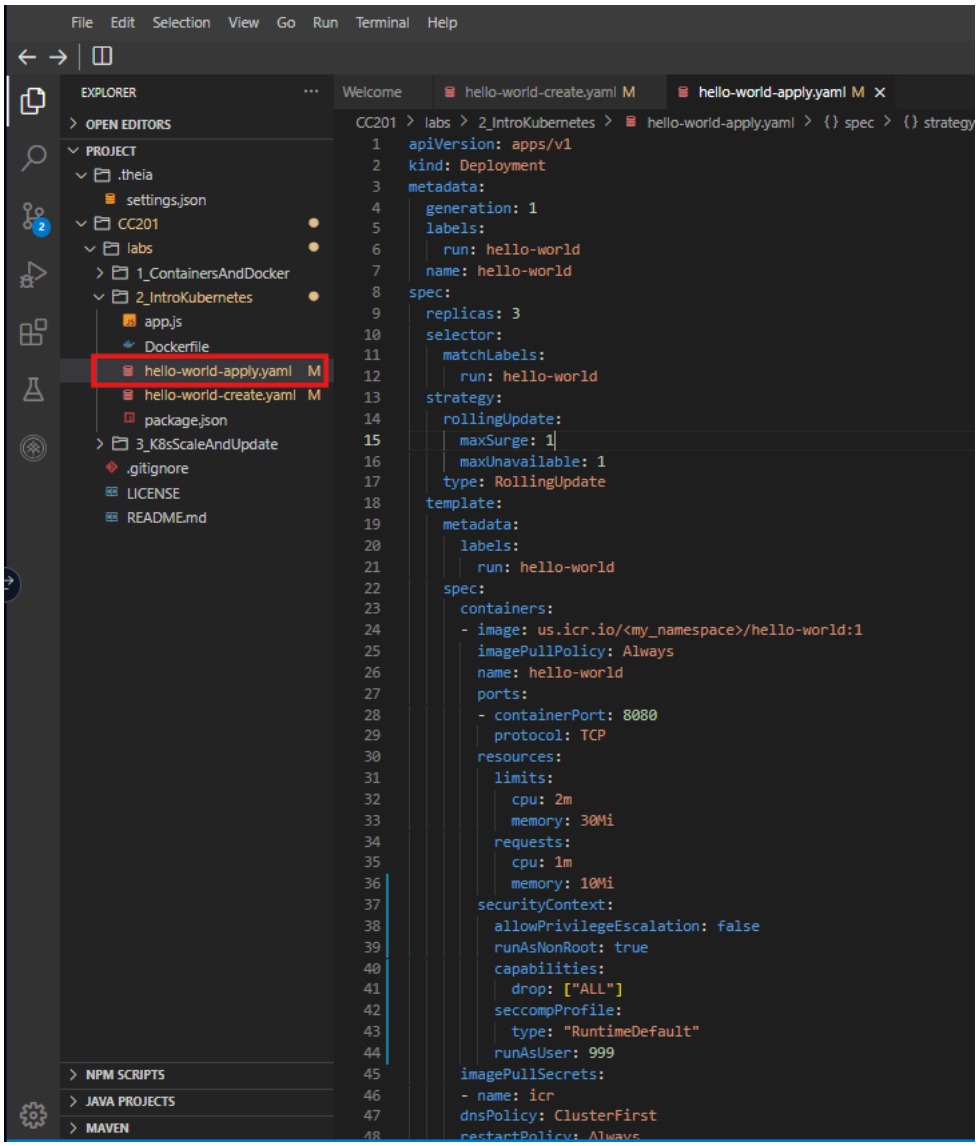
```
theia@theiadocker-[redacted] /home/project/CC201/labs/2_IntroKubernetes$ kubectl get pods
No resources found in sn-labs-[redacted] namespace.
theia@theiadocker-[redacted] /home/project/CC201/labs/2_IntroKubernetes$
```

Deklaratif Bir Komutla Pod Oluşturma

Önceki iki yöntemle Pod oluşturma, emirseldi – `kubectl`'ye ne yapacağını açıkça söyledik. Emirsel komutlar anlaşılması ve çalıştırılması kolay olsa da, üretim ortamı için ideal değildir. Şimdi deklaratif komutlara bakalım.

1. Bu dizinde bir örnek `hello-world-apply.yaml` dosyası bulunmaktadır. Bu dosyayı açmak için Explorer'ı tekrar kullanın. Aşağıdakilere dikkat edin:

- Bir Deployment oluşturuyoruz (`kind: Deployment`).
- Bu Deployment için üç kopya Pod olacak (`replicas: 3`).
- Pod'lar `hello-world` görüntüsünü çalıştırmalıdır (`- image: us.icr.io/<my_namespace>/hello-world:1`).



Şimdilik geri kalanını göz ardı edebilirsiniz. Bu kavramların çoğuna bir sonraki laboratuvarımızda değineceğiz.

2. Explorer'ı kullanarak `hello-world-apply.yaml` dosyasını düzenleyin. `<my_namespace>` ifadesinin olduğu yere kendi ad alanınızı eklemeniz gerekiyor. İşiniz bittiğinde dosyayı kaydetmeyi unutmayın.

Welcome | hello-world-create.yaml M | hello-world-apply.yaml M

CC201 > labs > 2_IntroKubernetes > hello-world-apply.yaml > {} spec > {} template >

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  generation: 1
  labels:
    run: hello-world
  name: hello-world
spec:
  replicas: 3
  selector:
    matchLabels:
      run: hello-world
  strategy:
    rollingUpdate:
      maxSurge: 1
      maxUnavailable: 1
    type: RollingUpdate
  template:
    metadata:
      labels:
        run: hello-world
    spec:
      containers:
      - image: us.icr.io/sn-labs-[redacted]/hello-world:1
        imagePullPolicy: Always
        name: hello-world
        ports:
        - containerPort: 8080
          protocol: TCP
        resources:
          limits:
            cpu: 2m
            memory: 30Mi
          requests:
            cpu: 1m
            memory: 10Mi
        securityContext:
          allowPrivilegeEscalation: false
          runAsNonRoot: true
          capabilities:
            drop: ["ALL"]
          seccompProfile:
            type: "RuntimeDefault"
          runAsUser: 999
      imagePullSecrets:
      - name: icr
      dnsPolicy: ClusterFirst
```

3. Bu yapılandırmayı Kubernetes'te istenen durum olarak ayarlamak için `kubectl apply` komutunu kullanın.

```
kubectl apply -f hello-world-apply.yaml
```

```
theia@theiadocker-[redacted]/home/project/CC201/labs/2_IntroKubernetes$ kubectl apply -f hello-world-apply.yaml
deployment.apps/hello-world created
```

4. Bir Deployment'ın oluşturulduğundan emin olmak için Deployments'ı alın.

```
kubectl get deployments
```

```
theia@theiadocker-[redacted]/home/project/CC201/labs/2_IntroKubernetes$ kubectl get deployments
NAME          READY   UP-TO-DATE   AVAILABLE   AGE
hello-world   3/3     3            3           22s
```

5. Üç kopyanın mevcut olduğunu doğrulamak için Pod'ları listeleyin.

```
kubectl get pods
```

```
theia@theiadocker-                /home/project/CC201/labs/2_IntroKubernetes$ kubectl get pods
NAME                              READY   STATUS    RESTARTS   AGE
hello-world-774ddf45b5-86gn6      1/1     Running   0          42s
hello-world-774ddf45b5-9cbv2      1/1     Running   0          41s
hello-world-774ddf45b5-svpf7      1/1     Running   0          41s
theia@theiadocker-                /home/project/CC201/labs/2_IntroKubernetes$
```

Açıklayıcı yönetim ile Kubernetes'e hangi eylemleri gerçekleştirmesi gerektiğini söylemedik. Bunun yerine, `kubectl` bu Dağıtımın oluşturulması gerektiğini anladı. Şu anda bir Pod'u silerseniz, üç kopyayı korumak için onun yerine yenisi oluşturulacaktır.

6. Önceki adımdan bir Pod adını not edin, aşağıdaki komuttaki `pod_name` kısmını not ettiğiniz pod adıyla değiştirin ve o Pod'u silip pod'ları listeleyin. Bir pod'un sonlandırıldığını görmek için, sadece 2 pod kalması amacıyla **silme** işlemini hemen ardından **al** işlemi ile takip edeceğiz.

   ```
   kubectl delete pod <pod_name> && kubectl get pods
   ```

```
theia@theiadocker-ksundararaja:/home/project/CC201/labs/2_IntroKubernetes$ kubectl delete pod hello-world-5b5467f896-9brft && kubect
pod "hello-world-5b5467f896-9brft" deleted
NAME                           READY   STATUS    RESTARTS   AGE
hello-world-5b5467f896-6jpnd   1/1     Running   0          3m7s
hello-world-5b5467f896-wz45f   1/1     Running   0          3m6s
theia@theiadocker-ksundararaja:/home/project/CC201/labs/2_IntroKubernetes$
```

Bu komut, pod'un silinmesini gerçekleştirmek için biraz zaman alır. Terminal istemi tekrar görününceye kadar bekleyin.

7. Yeni bir pod'un oluşturulduğunu görmek için Pod'ları listeleyin.

   Yeni pod'un oluşturulması biraz zaman alabileceğinden, bu komutu birkaç kez çalıştırmanız gerekebilir.

   ```
   kubectl get pods
   ```

```
NAME                     READY   STATUS    RESTARTS   AGE
hello-world-774ddf45b5-28k7j   1/1     Running   0          36s
hello-world-774ddf45b5-9cbv2   1/1     Running   0          112s
hello-world-774ddf45b5-svpf7   1/1     Running   0          112s
```

Çıktı, üç podun çalıştığını göstermelidir.

Uygulamanın Yük Dengelemesi

Bu uygulamanın kümede üç kopyası bulunduğundan, Kubernetes bu üç örnek arasında istekleri yük dengeleyecektir. Uygulamamızı internete açalım ve Kubernetes'in istekleri nasıl yük dengelediğine bakalım.

1. Uygulamaya erişmek için, onu bir Kubernetes Servisi kullanarak internete açmamız gerekiyor.

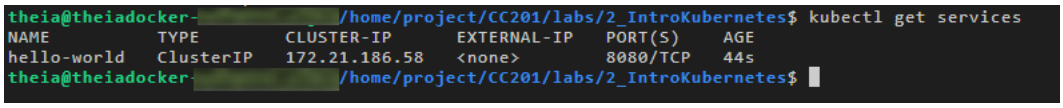
   ```
   kubectl expose deployment/hello-world
   ```

```
theia@theiadocker-                /home/project/CC201/labs/2_IntroKubernetes$ kubectl expose deployment/hello-world
service/hello-world exposed
```

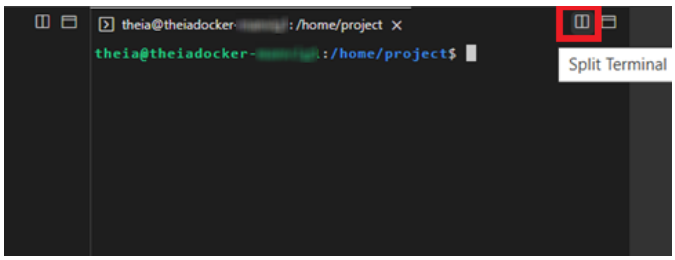
Bu komut, ClusterIP Servisi olarak adlandırılan bir şey oluşturur. Bu, küme içinde erişilebilir bir IP adresi oluşturur.

2. Bu servisin oluşturulduğunu görmek için Servisleri listeleyin.

```
kubectl get services
```

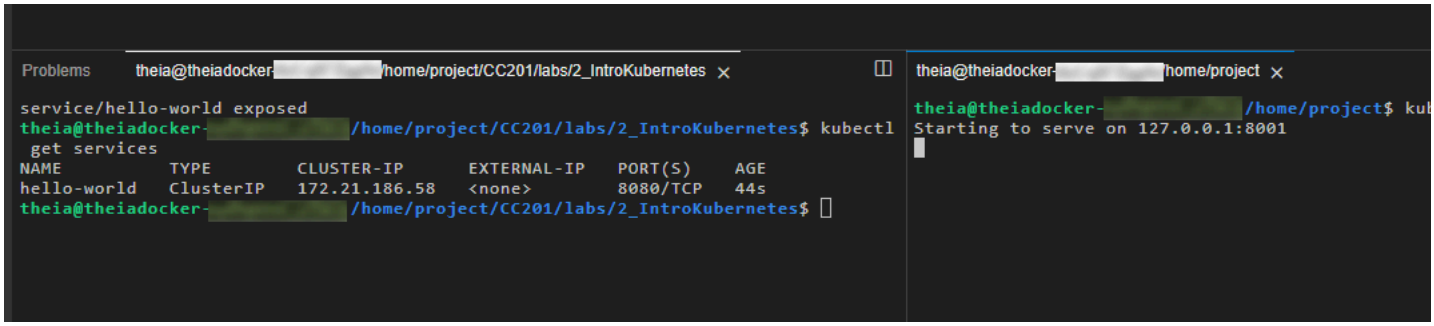


3. Terminal panelinin sağ üst köşesindeki bölme simgesini bularak yeni bir bölünmüş terminal penceresi açın.



4. Küme IP'si küme dışından erişilebilir olmadığından, bir proxy oluşturmalıyız. Bunun, bir uygulamayı üretim senaryosunda dışarıdan erişilebilir hale getirmenin yolu olmadığını unutmayın. Ortam değişkenlerinizin sonraki komutlar için orijinal pencerede erişilebilir olması gerektiğinden, bu komutu yeni terminal penceresinde çalıştırın.

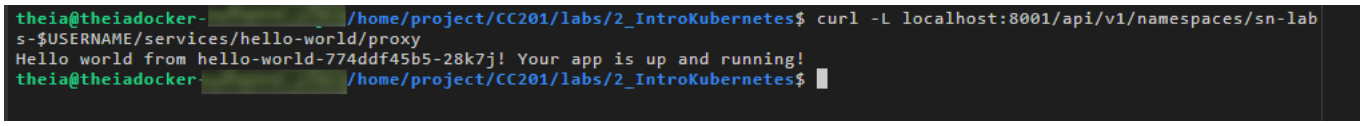
```
kubectl proxy
```



Bu komut, onu sonlandırana kadar sonlanmaz. Uygulamanıza erişmeye devam edebilmek için çalışır durumda tutun.

5. Orijinal terminal penceresinde, bir yanıt almak için uygulamayı pingleyin.

```
curl -L localhost:8001/api/v1/namespaces/sn-labs-$USERNAME/services/hello-world/proxy
```



Bu çıktının Pod adını içerdiğini unutmayın.

6. Kubernetes API proxy'si aracılığıyla hello-world hizmetine art arda on istek göndermek için aşağıdaki komutu çalıştırın. Her istek iletilirken, yanıtın içindeki pod adını not edin (bu, isteği hangi pod'un işlediğini gösterir).

```
for i in `seq 10`; do curl -L localhost:8001/api/v1/namespaces/sn-labs-$USERNAME/services/hello-world/proxy; done
```

```
theia@theiadocker-                /home/project/CC201/labs/2_IntroKubernetes$ for i in `seq 10`; do curl -L localhost:8001/ap
i/v1/namespaces/sn-labs-$USERNAME/services/hello-world/proxy; done
Hello world from hello-world-774ddf45b5-svpf7! Your app is up and running!
Hello world from hello-world-774ddf45b5-9cbv2! Your app is up and running!
Hello world from hello-world-774ddf45b5-28k7j! Your app is up and running!
Hello world from hello-world-774ddf45b5-28k7j! Your app is up and running!
Hello world from hello-world-774ddf45b5-28k7j! Your app is up and running!
Hello world from hello-world-774ddf45b5-28k7j! Your app is up and running!
Hello world from hello-world-774ddf45b5-28k7j! Your app is up and running!
Hello world from hello-world-774ddf45b5-28k7j! Your app is up and running!
Hello world from hello-world-774ddf45b5-svpf7! Your app is up and running!
Hello world from hello-world-774ddf45b5-svpf7! Your app is up and running!
theia@theiadocker-                /home/project/CC201/labs/2_IntroKubernetes$
```

Çıktıda birden fazla Pod adı ve muhtemelen üç Pod adının tamamını görmelisiniz. Bunun nedeni, Kubernetes'in istekleri üç kopya arasında dengeleyerek her isteğin uygulamamızın farklı bir örneğine ulaşabilmesidir.

7. Dağıtımı ve Servisi silin. Bu, eğik çizgiler kullanarak tek bir komutla yapılabilir.

```
kubectl delete deployment/hello-world service/hello-world
```

```
theia@theiadocker                 /home/project/CC201/labs/2_IntroKubernetes$ kubectl delete deployment/hello-world service/h
ello-world
deployment.apps "hello-world" deleted
service "hello-world" deleted
theia@theiadocker-                /home/project/CC201/labs/2_IntroKubernetes$
```

Not: Terminalde daha fazla komut yazarken herhangi bir sorunla karşılaşırsanız, Enter tuşuna basın.

8. `proxy` komutunu çalıştıran terminal penceresine geri dönün ve `Ctrl+C` kullanarak kapatın.

```
theia@theiadocker-                home/project ×

theia@theiadocker-                /home/project$ kubectl proxy
Starting to serve on 127.0.0.1:8001
^C
theia@theiadocker-                /home/project$
```

Tebrikler! Bu kursun ikinci modülü için laboratuvarı tamamladınız.