+ -
Helvetica Ne...
▼
Step 1 of 7

IBM Developer
SKILLS NETWORK

Scaling and Updating Applications

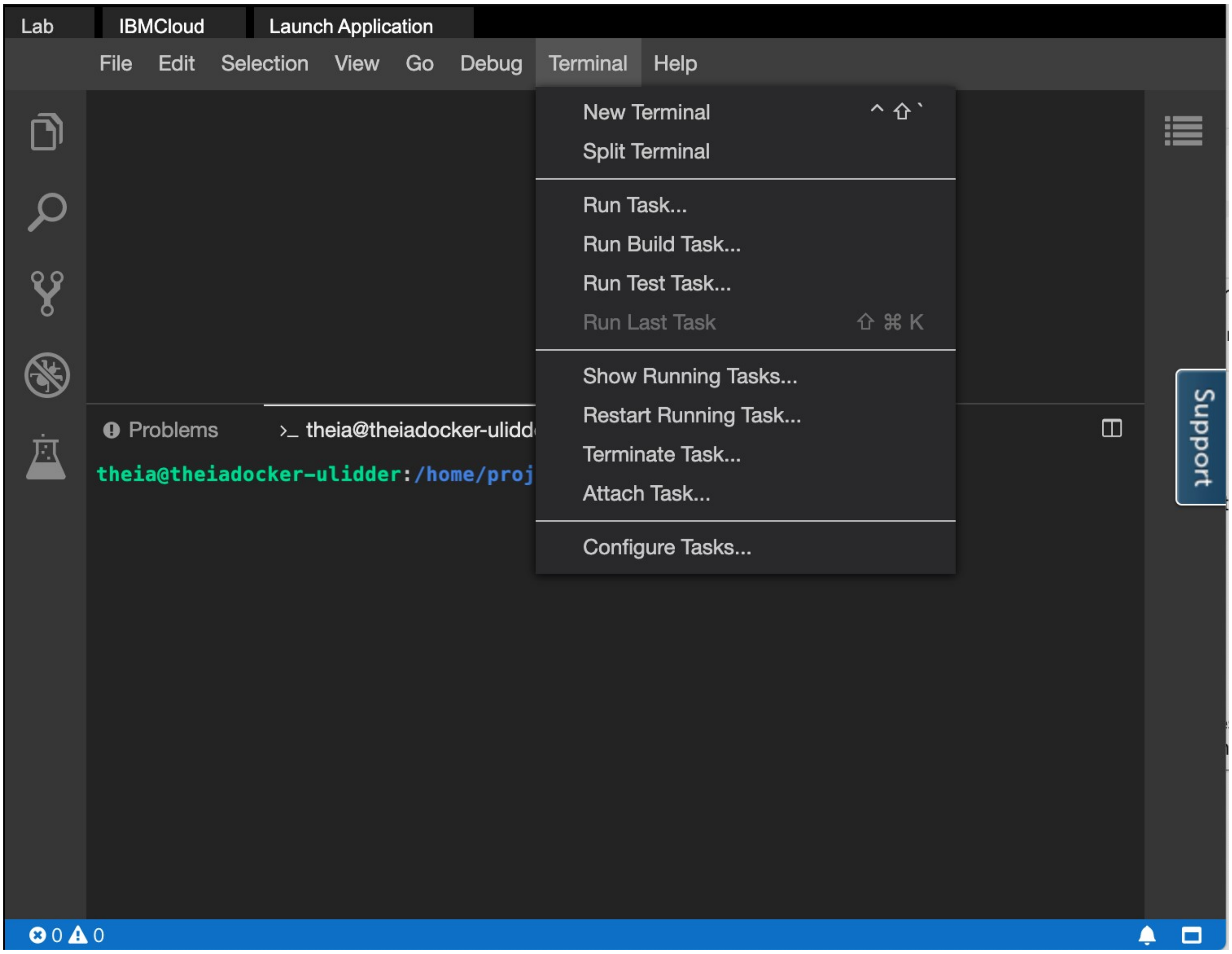
Objectives

In this lab, you will:

- Scale an application with a ReplicaSet
- Apply rolling updates to an application
- Use a ConfigMap to store application configuration

Verify the environment and command line tools

1. If a terminal is not already open, open a terminal window by using the menu in the editor: `Terminal > New Terminal`.



```
theia@theiadocker-ulidder:/home/proj
```

NOTE: It might take sometime for the Termainal Prompt to appear. In case you are unable to see the terminal prompt even after 5 minutes, please close the browser tab and relaunch the lab again.

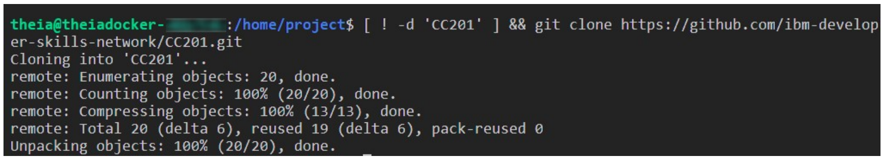
2. Change to your project folder.

NOTE: If you are already in the `/home/project` please skip this step.

```
cd /home/project
```

3. Clone the git repository that contains the artifacts needed for this lab, if it doesn't already exist.

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



```
theia@theiadocker-[redacted]:/home/project$ [ ! -d 'CC201' ] && git clone https://github.com/ibm-developer-skills-network/CC201.git
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.
```

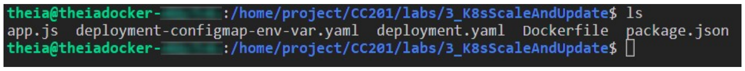
4. Change to the directory for this lab.

```
cd CC201/labs/3_K8sScaleAndUpdate/
```

```
theia@theiadocker-[redacted]:/home/project$ cd CC201/labs/3_K8sScaleAndUpdate/
theia@theiadocker-[redacted]:/home/project/CC201/labs/3_K8sScaleAndUpdate$
```

5. List the contents of this directory to see the artifacts for this lab.

```
ls
```

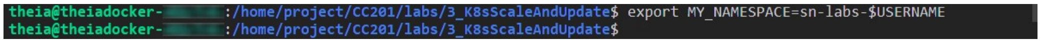


```
theia@theiadocker-[redacted]:/home/project/CC201/labs/3_K8sScaleAndUpdate$ ls
app.js  deployment-configmap-env-var.yaml  deployment.yaml  Dockerfile  package.json
theia@theiadocker-[redacted]:/home/project/CC201/labs/3_K8sScaleAndUpdate$
```

Build and push application image to IBM Cloud Container Registry

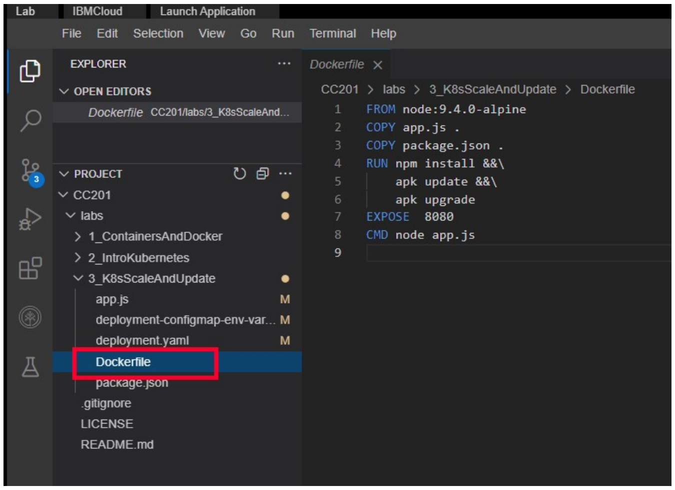
1. Export your namespace as an environment variable so that it can be used in subsequent commands.

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



```
theia@theiadocker-[redacted]:/home/project/CC201/labs/3_K8sScaleAndUpdate$ export MY_NAMESPACE=sn-labs-$USERNAME
theia@theiadocker-[redacted]:/home/project/CC201/labs/3_K8sScaleAndUpdate$
```

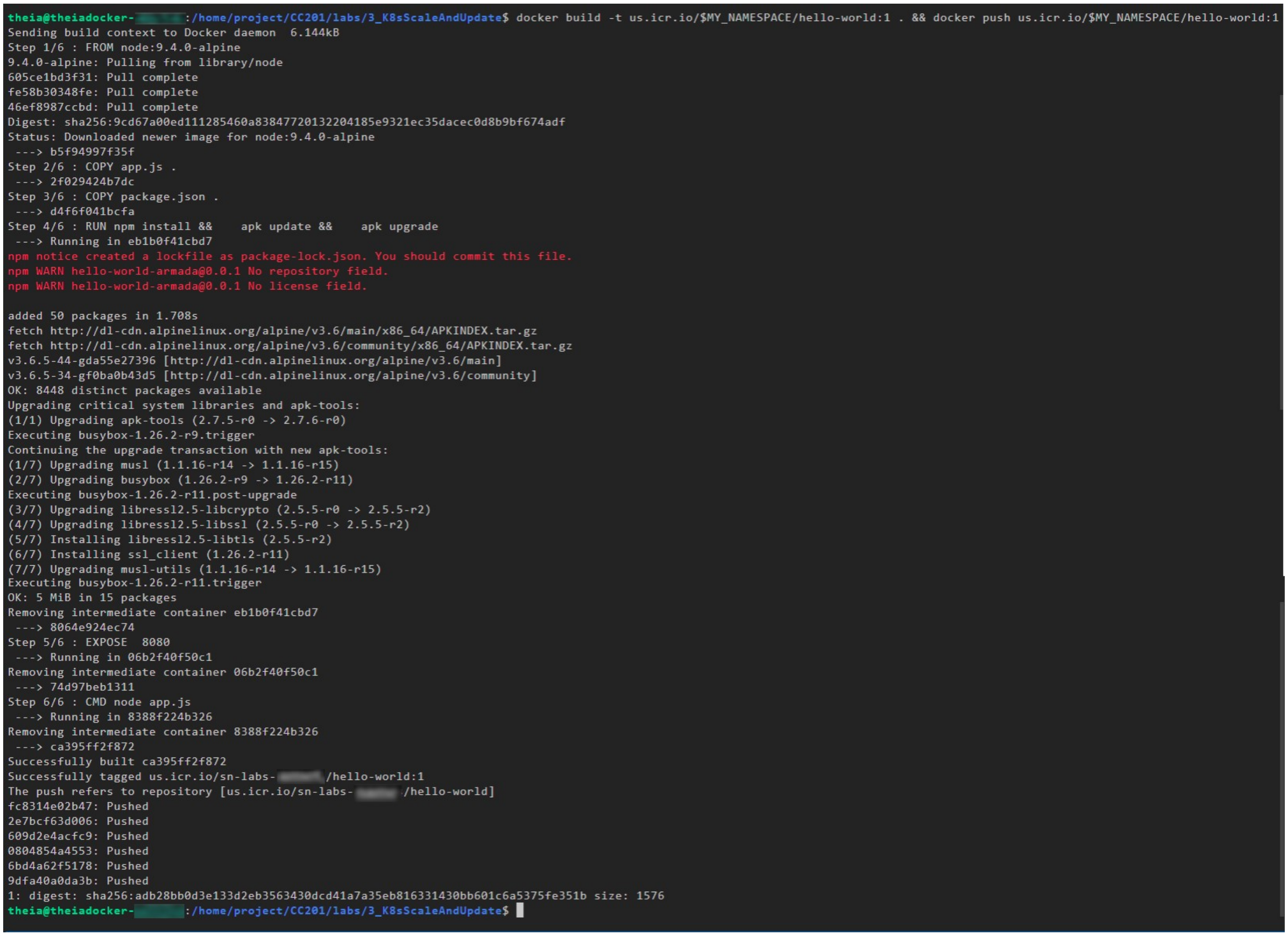
2. Use the Explorer to view the Dockerfile that will be used to build an image.



```
FROM node:9.4.0-alpine
COPY app.js .
COPY package.json .
RUN npm install &&\
    apk update &&\
    apk upgrade
EXPOSE  8080
CMD node app.js
```

3. Build and push the image again, as it may have been deleted automatically since you completed the first lab.

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



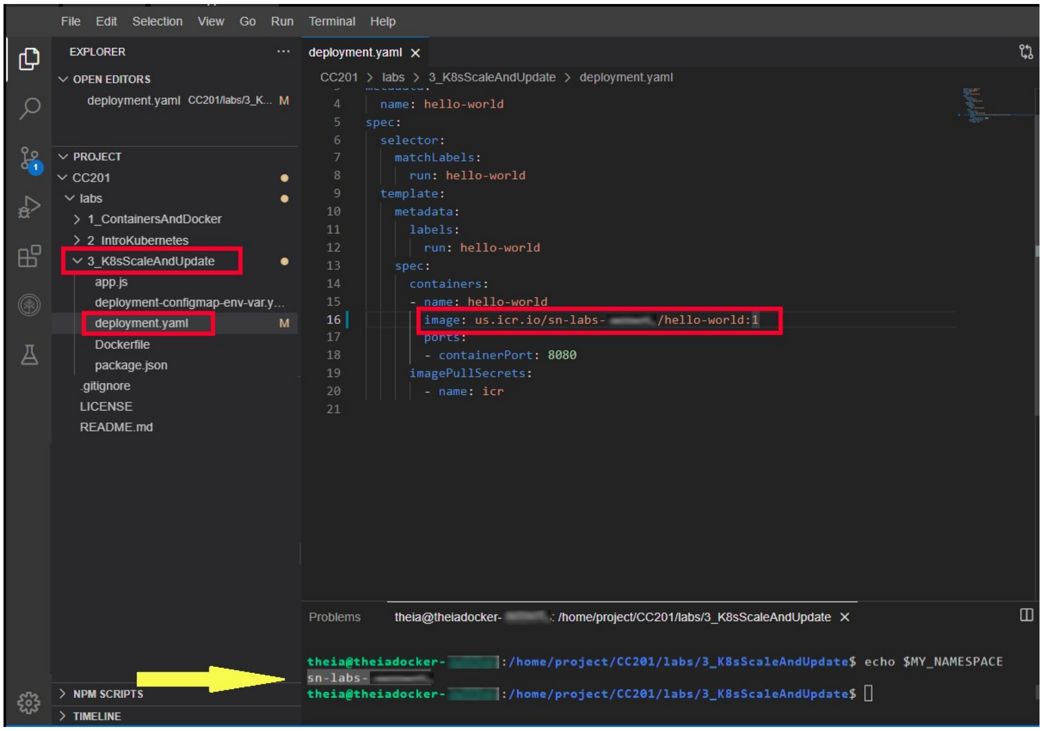
```
theia@theiadocker-[redacted]:/home/project/CC201/labs/3_K8sScaleAndUpdate$ docker build -t us.icr.io/$MY_NAMESPACE/hello-world:1 . && docker push us.icr.io/$MY_NAMESPACE/hello-world:1
Sending build context to Docker daemon  6.144kB
Step 1/6 : FROM node:9.4.0-alpine
9.4.0-alpine: Pulling from library/node
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 .
 ---> 2f029424b7dc
Step 3/6 : COPY package.json .
 ---> d4f6f041bcfa
Step 4/6 : RUN npm install &&    apk update &&    apk upgrade
 ---> Running in eb1b0f41cbd7
npm notice created a lockfile as package-lock.json. You should commit this file.
npm WARN hello-world-armada@0.0.1 No repository field.
npm WARN hello-world-armada@0.0.1 No license field.

added 50 packages in 1.708s
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)
(6/7) Installing ssl_client (1.26.2-r11)
(7/7) Upgrading musl-utils (1.1.16-r14 -> 1.1.16-r15)
Executing busybox-1.26.2-r11.trigger
OK: 5 MiB in 15 packages
Removing intermediate container eb1b0f41cbd7
 ---> 8064e924ec74
Step 5/6 : EXPOSE  8080
 ---> Running in 06b2f40f50c1
Removing intermediate container 06b2f40f50c1
 ---> 74d97beb1311
Step 6/6 : CMD node app.js
 ---> Running in 8388f224b326
Removing intermediate container 8388f224b326
 ---> ca395ff2f872
Successfully built ca395ff2f872
Successfully tagged us.icr.io/sn-labs-[redacted]/hello-world:1
The push refers to repository [us.icr.io/sn-labs-[redacted]/hello-world]
fc8314e02b47: Pushed
2e7bcf63d006: Pushed
609d2e4acfc9: Pushed
0804854a4553: Pushed
6bd4a62f5178: Pushed
9dfa40a0da3b: Pushed
1: digest: sha256:adb28bb0d3e133d2eb3563430dcd41a7a35eb816331430bb601c6a5375fe351b size: 1576
theia@theiadocker-[redacted]:/home/project/CC201/labs/3_K8sScaleAndUpdate$
```

NOTE: If you have tried this lab earlier, there might be a possibility that the previous session is still persistent. In such case, you will see a **'Layer already Exists'** message instead of the **'Pushed'** message in the above output. We would recommend you to continue with the further steps of the lab.

Deploy the application to Kubernetes

1. Use the Explorer to edit `deployment.yaml` in this directory. The path to this file is `CC201/labs/3_K8sScaleAndUpdate/`. You need to insert your namespace where it says `<my_namespace>`. Make sure to save the file when you're done.

   NOTE: To know your namespace, run `echo $MY_NAMESPACE` in the terminal



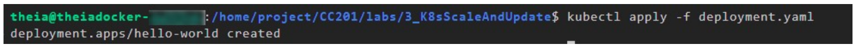
File Edit Selection View Go Run Terminal Help
EXPLORER
OPEN EDITORS
deployment.yaml CC201/labs/3_K... M
PROJECT
CC201
labs
1_ContainersAndDocker
2_IntroKubernetes
3_K8sScaleAndUpdate
app.js
deployment-configmap-env-var.y...
deployment.yaml M
Dockerfile
package.json
.gitignore
LICENSE
README.md
NPM SCRIPTS
TIMELINE
deployment.yaml
CC201 > labs > 3_K8sScaleAndUpdate > deployment.yaml
Problems
theia@theiadocker-[redacted]: /home/project/CC201/labs/3_K8sScaleAndUpdate

```
  name: hello-world
spec:
  selector:
    matchLabels:
      run: hello-world
  template:
    metadata:
      labels:
        run: hello-world
    spec:
      containers:
      - name: hello-world
        image: us.icr.io/sn-labs-[redacted]/hello-world:1
        ports:
        - containerPort: 8080
      imagePullSecrets:
        - name: icr
```

```
theia@theiadocker-[redacted]:/home/project/CC201/labs/3_K8sScaleAndUpdate$ echo $MY_NAMESPACE
sn-labs-[redacted]
theia@theiadocker-[redacted]:/home/project/CC201/labs/3_K8sScaleAndUpdate$
```

2. Run your image as a Deployment.

```
kubectl apply -f deployment.yaml
```

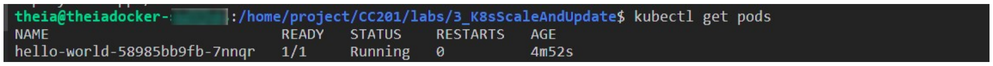


```
theia@theiadocker-[redacted]:/home/project/CC201/labs/3_K8sScaleAndUpdate$ kubectl apply -f deployment.yaml
deployment.apps/hello-world created
```

NOTE: If you have tried this lab earlier, there might be a possibility that the previous session is still persistent. In such a case, you will see an **'Unchanged'** message instead of the **'Created'** message in the above output. We would recommend you to continue with the further steps of the lab.

3. List Pods until the status is "Running".

```
kubectl get pods
```



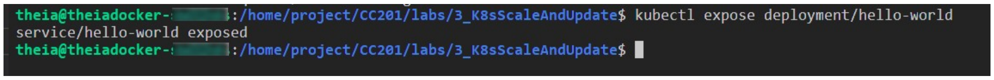
```
theia@theiadocker-[redacted]:/home/project/CC201/labs/3_K8sScaleAndUpdate$ kubectl get pods
NAME                          READY   STATUS    RESTARTS   AGE
hello-world-58985bb9fb-7nnqr  1/1     Running   0          4m52s
```

NOTE: Please move to the next step only after you see the pod status as **'Running'**. In case you see **'Container Creating'** as the output, please re-run the command in a few minutes.

4. In order to access the application, we have to expose it to the internet via a Kubernetes Service.

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

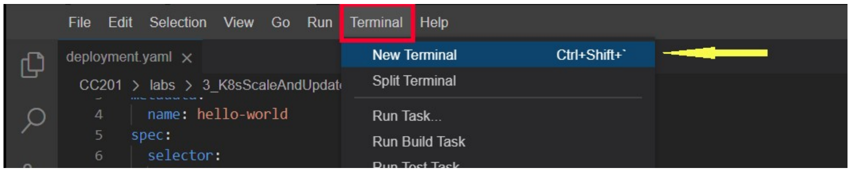
This creates a service of type **ClusterIP**.



```
theia@theiadocker-[redacted]:/home/project/CC201/labs/3_K8sScaleAndUpdate$ kubectl expose deployment/hello-world
service/hello-world exposed
theia@theiadocker-[redacted]:/home/project/CC201/labs/3_K8sScaleAndUpdate$ █
```

5. Open a new terminal window using `Terminal > New Terminal`.

NOTE: Do not close the terminal window you were working on.



```
4    name: hello-world
5  spec:
6    selector:
```

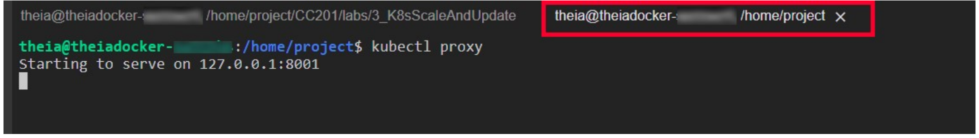
6. Cluster IPs are only accesible within the cluster. To make this externally accessible, we will create a proxy.

Note: This is not how you would make an application externally accessible in a production scenario.

Run this command in the new terminal window since your environment variables need to be accessible in the original window for subsequent commands.

```
kubectl proxy
```

This command will continue running until it exits. Keep it running so that you can continue to access your app.

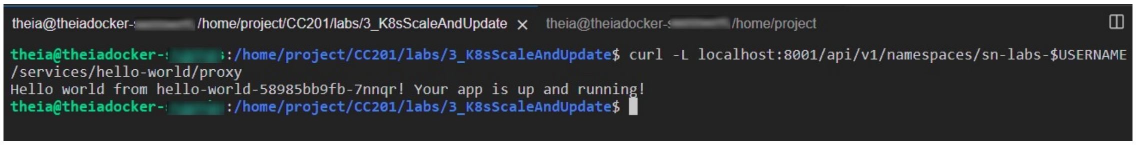

theia@theiadocker-[redacted] /home/project/CC201/labs/3_K8sScaleAndUpdate
theia@theiadocker-[redacted] /home/project ×

```
theia@theiadocker-[redacted]:/home/project$ kubectl proxy
Starting to serve on 127.0.0.1:8001
█
```

7. Go back to your original terminal window, ping the application to get a response.

NOTE: Do not close the terminal window where the `proxy` command is still running.

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


theia@theiadocker-[redacted] /home/project/CC201/labs/3_K8sScaleAndUpdate ×
theia@theiadocker-[redacted] /home/project

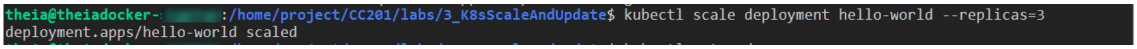
```
theia@theiadocker-[redacted]:/home/project/CC201/labs/3_K8sScaleAndUpdate$ curl -L localhost:8001/api/v1/namespaces/sn-labs-$USERNAME
/services/hello-world/proxy
Hello world from hello-world-58985bb9fb-7nnqr! Your app is up and running!
theia@theiadocker-[redacted]:/home/project/CC201/labs/3_K8sScaleAndUpdate$ █
```

Scaling the application using a ReplicaSet

In real-world situations, load on an application can vary over time. If our application begins experiencing heightened load, we want to scale it up to accommodate that load. There is a simple `kubectl` command for scaling.

1. Use the `scale` command to scale up your Deployment. Make sure to run this in the terminal window that is not running the `proxy` command.

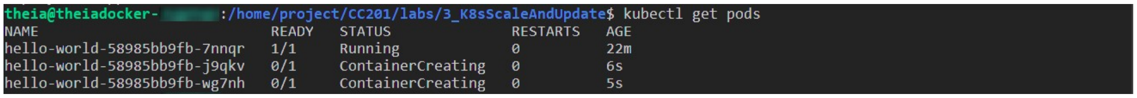
```
kubectl scale deployment hello-world --replicas=3
```



```
theia@theiadocker-[redacted]:/home/project/CC201/labs/3_K8sScaleAndUpdate$ kubectl scale deployment hello-world --replicas=3
deployment.apps/hello-world scaled
```

2. Get Pods to ensure that there are now three Pods instead of just one. In addition, the status should eventually update to "Running" for all three.

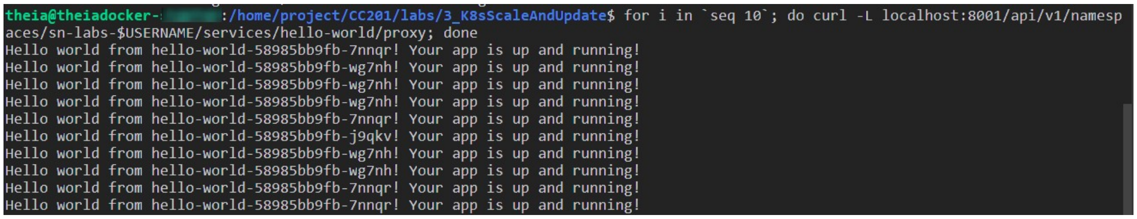
```
kubectl get pods
```



```
theia@theiadocker-[redacted]:/home/project/CC201/labs/3_K8sScaleAndUpdate$ kubectl get pods
NAME                           READY   STATUS              RESTARTS   AGE
hello-world-58985bb9fb-7nnqr   1/1     Running             0          22m
hello-world-58985bb9fb-j9qkv   0/1     ContainerCreating   0          6s
hello-world-58985bb9fb-wg7nh   0/1     ContainerCreating   0          5s
```

3. As you did in the last lab, ping your application multiple times to ensure that Kubernetes is load-balancing across the replicas.

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

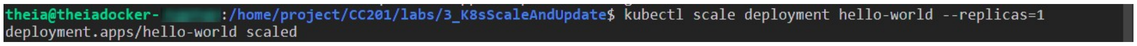


```
theia@theiadocker-[redacted]:/home/project/CC201/labs/3_K8sScaleAndUpdate$ for i in `seq 10`; do curl -L localhost:8001/api/v1/namesp
aces/sn-labs-$USERNAME/services/hello-world/proxy; done
Hello world from hello-world-58985bb9fb-7nnqr! Your app is up and running!
Hello world from hello-world-58985bb9fb-wg7nh! Your app is up and running!
Hello world from hello-world-58985bb9fb-wg7nh! Your app is up and running!
Hello world from hello-world-58985bb9fb-wg7nh! Your app is up and running!
Hello world from hello-world-58985bb9fb-7nnqr! Your app is up and running!
Hello world from hello-world-58985bb9fb-j9qkv! Your app is up and running!
Hello world from hello-world-58985bb9fb-wg7nh! Your app is up and running!
Hello world from hello-world-58985bb9fb-wg7nh! Your app is up and running!
Hello world from hello-world-58985bb9fb-7nnqr! Your app is up and running!
Hello world from hello-world-58985bb9fb-7nnqr! Your app is up and running!
```

You should see that the queries are going to different Pods.

4. Similarly, you can use the `scale` command to scale down your Deployment.

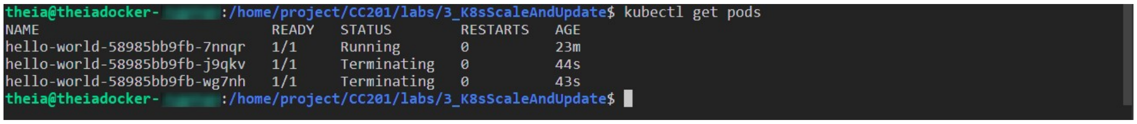
```
kubectl scale deployment hello-world --replicas=1
```



```
theia@theiadocker-[redacted]:/home/project/CC201/labs/3_K8sScaleAndUpdate$ kubectl scale deployment hello-world --replicas=1
deployment.apps/hello-world scaled
```

5. Check the Pods to see that two are deleted or being deleted.

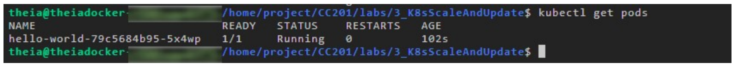
```
kubectl get pods
```



```
theia@theiadocker-[redacted]:/home/project/CC201/labs/3_K8sScaleAndUpdate$ kubectl get pods
NAME                           READY   STATUS        RESTARTS   AGE
hello-world-58985bb9fb-7nnqr   1/1     Running       0          23m
hello-world-58985bb9fb-j9qkv   1/1     Terminating   0          44s
hello-world-58985bb9fb-wg7nh   1/1     Terminating   0          43s
theia@theiadocker-[redacted]:/home/project/CC201/labs/3_K8sScaleAndUpdate$
```

6. Please wait for some time & run the same command again to ensure that only one pod exists.

```
kubectl get pods
```

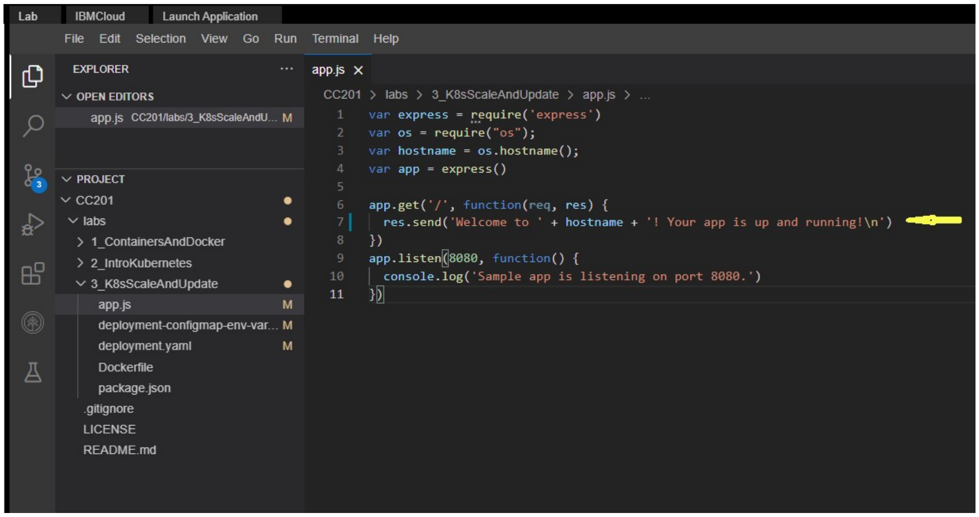


```
theia@theiadocker-[redacted]/home/project/CC201/labs/3_K8sScaleAndUpdate$ kubectl get pods
NAME                           READY   STATUS    RESTARTS   AGE
hello-world-79c5684b95-5x4wp   1/1     Running   0          102s
theia@theiadocker-[redacted]/home/project/CC201/labs/3_K8sScaleAndUpdate$
```

Perform rolling updates

Rolling updates are an easy way to update our application in an automated and controlled fashion. To simulate an update, let's first build a new version of our application and push it to Container Registry.

1. Use the Explorer to edit `app.js`. The path to this file is `CC201/labs/3_K8sScaleAndUpdate/`. Change the welcome message from `'Hello world from ' + hostname + '! Your app is up and running!\n'` to `'Welcome to ' + hostname + '! Your app is up and running!\n'`. Make sure to save the file when you're done.

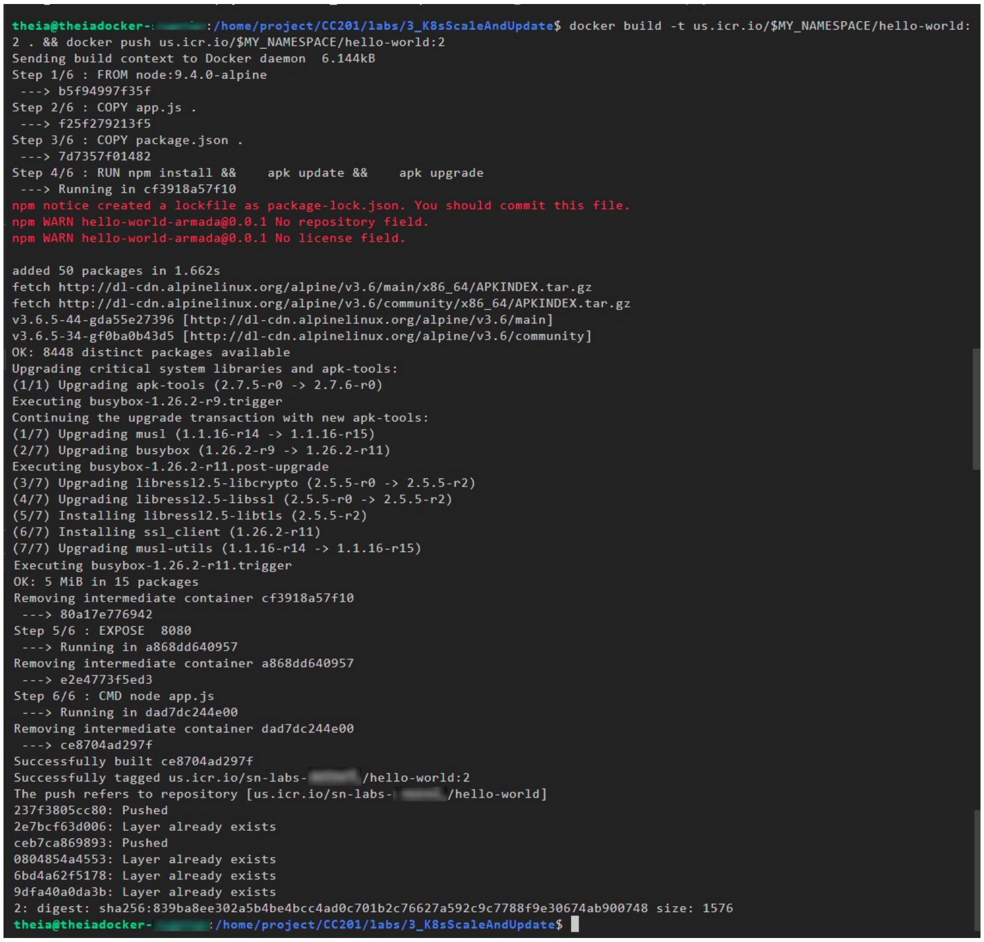


```
var express = require('express')
var os = require("os");
var hostname = os.hostname();
var app = express()

app.get('/', function(req, res) {
  res.send('Welcome to ' + hostname + '! Your app is up and running!\n')
})
app.listen(8080, function() {
  console.log('Sample app is listening on port 8080.')
})
```

2. Build and push this new version to Container Registry. Update the tag to indicate that this is a second version of this application. Make sure to use the terminal window that isn't running the `proxy` command.

> **NOTE:** Do not close the terminal that is running the `proxy` command

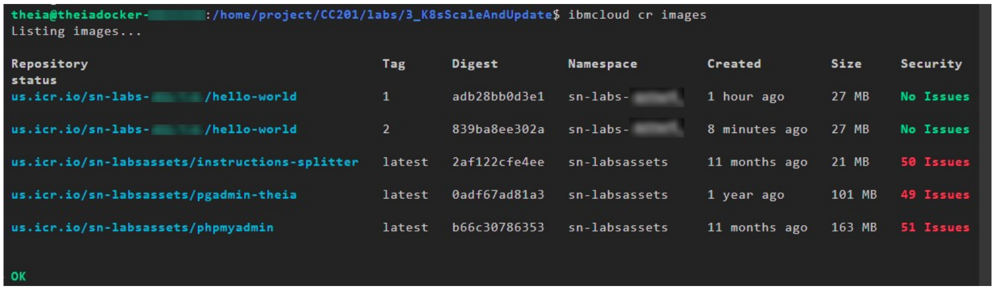
```
docker build -t us.icr.io/$MY_NAMESPACE/hello-world:2 . && docker push us.icr.io/$MY_NAMESPACE/hello-world:2
```



```
theia@theiadocker-[redacted]:/home/project/CC201/labs/3_K8sScaleAndUpdate$ docker build -t us.icr.io/$MY_NAMESPACE/hello-world:
2 . && docker push us.icr.io/$MY_NAMESPACE/hello-world:2
Sending build context to Docker daemon  6.144kB
Step 1/6 : FROM node:9.4.0-alpine
 ---> b5f94997f35f
Step 2/6 : COPY app.js .
 ---> f25f279213f5
Step 3/6 : COPY package.json .
 ---> 7d7357f01482
Step 4/6 : RUN npm install &&    apk update &&    apk upgrade
 ---> Running in cf3918a57f10
npm notice created a lockfile as package-lock.json. You should commit this file.
npm WARN hello-world-armada@0.0.1 No repository field.
npm WARN hello-world-armada@0.0.1 No license field.

added 50 packages in 1.662s
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)
(6/7) Installing ssl_client (1.26.2-r11)
(7/7) Upgrading musl-utils (1.1.16-r14 -> 1.1.16-r15)
Executing busybox-1.26.2-r11.trigger
OK: 5 MiB in 15 packages
Removing intermediate container cf3918a57f10
 ---> 80a17e776942
Step 5/6 : EXPOSE  8080
 ---> Running in a868dd640957
Removing intermediate container a868dd640957
 ---> e2e4773f5ed3
Step 6/6 : CMD node app.js
 ---> Running in dad7dc244e00
Removing intermediate container dad7dc244e00
 ---> ce8704ad297f
Successfully built ce8704ad297f
Successfully tagged us.icr.io/sn-labs-[redacted]/hello-world:2
The push refers to repository [us.icr.io/sn-labs-[redacted]/hello-world]
237f3805cc80: Pushed
2e7bcf63d006: Layer already exists
ceb7ca869893: Pushed
0804854a4553: Layer already exists
6bd4a62f5178: Layer already exists
9dfa40a0da3b: Layer already exists
2: digest: sha256:839ba8ee302a5b4be4bcc4ad0c701b2c76627a592c9c7788f9e30674ab900748 size: 1576
theia@theiadocker-[redacted]:/home/project/CC201/labs/3_K8sScaleAndUpdate$
```

3. List images in Container Registry to see all the different versions of this application that you have pushed so far.

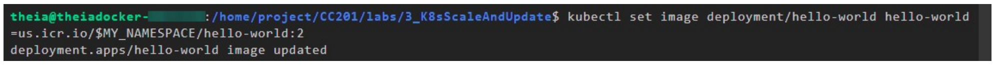
```
ibmcloud cr images
```



```
theia@theiadocker-[redacted]:/home/project/CC201/labs/3_K8sScaleAndUpdate$ ibmcloud cr images
Listing images...

Repository                                  Tag      Digest         Namespace            Created          Size     Security
status
us.icr.io/sn-labs-[redacted]/hello-world                 1        adb28bb0d3e1   sn-labs-[redacted]     1 hour ago       27 MB    No Issues

us.icr.io/sn-labs-[redacted]/hello-world                 2        839ba8ee302a   sn-labs-[redacted]     8 minutes ago    27 MB    No Issues

us.icr.io/sn-labsassets/instructions-splitter   latest   2af122cfe4ee   sn-labsassets   11 months ago    21 MB    50 Issues

us.icr.io/sn-labsassets/pgadmin-theia       latest   0adf67ad81a3   sn-labsassets   1 year ago       101 MB   49 Issues

us.icr.io/sn-labsassets/phpmyadmin          latest   b66c30786353   sn-labsassets   11 months ago    163 MB   51 Issues


OK
```

4. Update the deployment to use this version instead.

```
kubectl set image deployment/hello-world hello-world=us.icr.io/$MY_NAMESPACE/hello-world:2
```

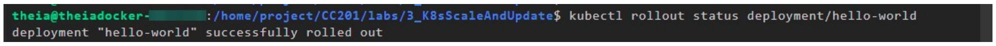


```
theia@theiadocker-[redacted]:/home/project/CC201/labs/3_K8sScaleAndUpdate$ kubectl set image deployment/hello-world hello-world
=us.icr.io/$MY_NAMESPACE/hello-world:2
deployment.apps/hello-world image updated
```

5. Get a status of the rolling update by using the following command:

```
kubectl rollout status deployment/hello-world
```

You should see an output like this, indicating that the rollout succeeded:

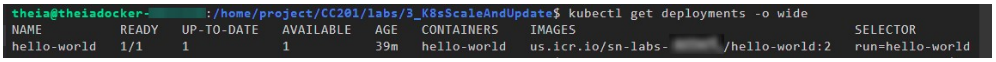


```
theia@theiadocker-[redacted]:/home/project/CC201/labs/3_K8sScaleAndUpdate$ kubectl rollout status deployment/hello-world
deployment "hello-world" successfully rolled out
```

6. You can also get the Deployment with the `wide` option to see that the new tag is used for the image.

```
kubectl get deployments -o wide
```

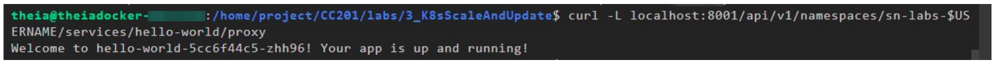
Look for the `IMAGES` column and ensure that the tag is `2`.



```
theia@theiadocker-[redacted]:/home/project/CC201/labs/3_K8sScaleAndUpdate$ kubectl get deployments -o wide
NAME          READY   UP-TO-DATE   AVAILABLE   AGE   CONTAINERS    IMAGES                                  SELECTOR
hello-world   1/1     1            1           39m   hello-world   us.icr.io/sn-labs-[redacted]/hello-world:2   run=hello-world
```

7. Ping your application to ensure that the new welcome message is displayed.

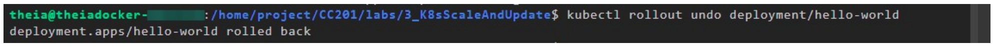
```
curl -L localhost:8001/api/v1/namespaces/sn-labs-$USERNAME/services/hello-world/proxy
```



```
theia@theiadocker-[redacted]:/home/project/CC201/labs/3_K8sScaleAndUpdate$ curl -L localhost:8001/api/v1/namespaces/sn-labs-$US
ERNAME/services/hello-world/proxy
Welcome to hello-world-5cc6f44c5-zhh96! Your app is up and running!
```

8. It's possible that a new version of an application contains a bug. In that case, Kubernetes can roll back the Deployment like this:

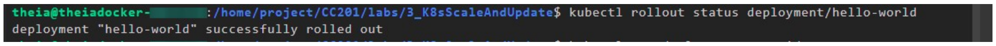
```
kubectl rollout undo deployment/hello-world
```



```
theia@theiadocker-[redacted]:/home/project/CC201/labs/3_K8sScaleAndUpdate$ kubectl rollout undo deployment/hello-world
deployment.apps/hello-world rolled back
```

9. Get a status of the rolling update by using the following command:

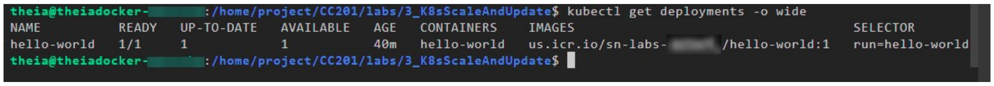
```
kubectl rollout status deployment/hello-world
```



```
theia@theiadocker-[redacted]:/home/project/CC201/labs/3_K8sScaleAndUpdate$ kubectl rollout status deployment/hello-world
deployment "hello-world" successfully rolled out
```

10. Get the Deployment with the `wide` option to see that the old tag is used.

```
kubectl get deployments -o wide
```

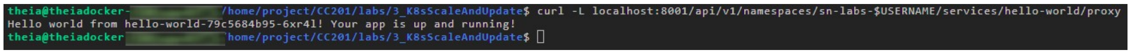


```
theia@theiadocker-[redacted]:/home/project/CC201/labs/3_K8sScaleAndUpdate$ kubectl get deployments -o wide
NAME          READY   UP-TO-DATE   AVAILABLE   AGE   CONTAINERS    IMAGES                                 SELECTOR
hello-world   1/1     1            1           40m   hello-world   us.icr.io/sn-labs-[redacted]/hello-world:1   run=hello-world
theia@theiadocker-[redacted]:/home/project/CC201/labs/3_K8sScaleAndUpdate$
```

Look for the `IMAGES` column and ensure that the tag is `1`.

11. Ping your application to ensure that the earlier '**Hello World..Your app is up & running!**' message is displayed.

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



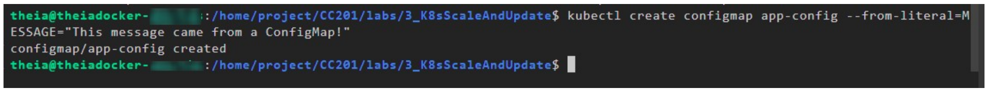
```
theia@theiadocker-[redacted]/home/project/CC201/labs/3_K8sScaleAndUpdate$ curl -L localhost:8001/api/v1/namespaces/sn-labs-$USERNAME/services/hello-world/proxy
Hello world from hello-world-79c5684b95-6xr4l! Your app is up and running!
theia@theiadocker[redacted]home/project/CC201/labs/3_K8sScaleAndUpdate$
```

Using a ConfigMap to store configuration

ConfigMaps and Secrets are used to store configuration information separate from the code so that nothing is hardcoded. It also lets the application pick up configuration changes without needing to be redeployed. To demonstrate this, we'll store the application's message in a ConfigMap so that the message can be updated simply by updating the ConfigMap.

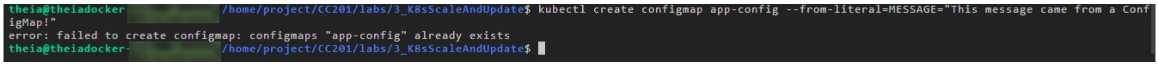
1. Create a ConfigMap that contains a new message.

```
kubectl create configmap app-config --from-literal=MESSAGE="This message came from a ConfigMap!"
```



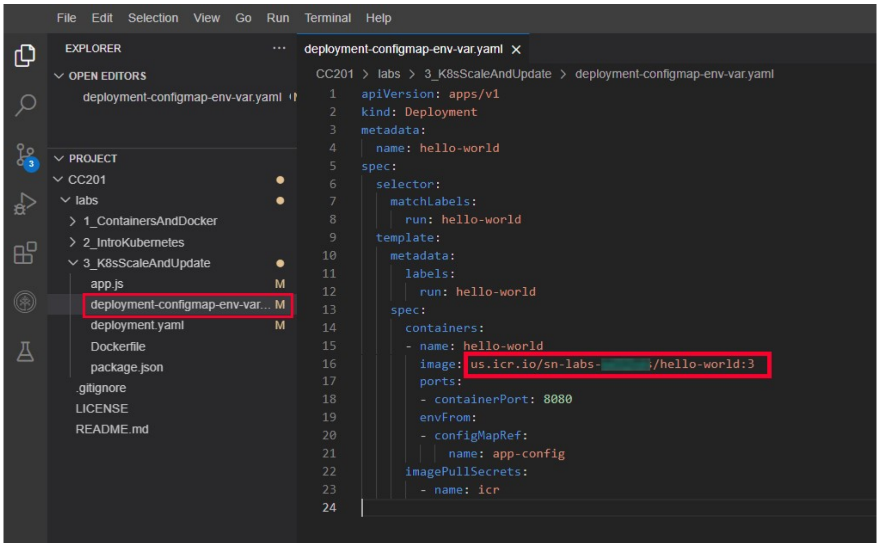
```
theia@theiadocker-[redacted]:/home/project/CC201/labs/3_K8sScaleAndUpdate$ kubectl create configmap app-config --from-literal=M
ESSAGE="This message came from a ConfigMap!"
configmap/app-config created
theia@theiadocker-[redacted]:/home/project/CC201/labs/3_K8sScaleAndUpdate$
```

NOTE: If you have tried this lab earlier, there might be a possibility that the previous session is still persistent. In such a case, you will see an **'error: failed to create configmap: configmaps "app-config" already exists'** message, instead of the **'Created'** message as below. We would recommend you to continue with the further steps of the lab.



```
theia@theiadocker[redacted]/home/project/CC201/labs/3_K8sScaleAndUpdate$ kubectl create configmap app-config --from-literal=MESSAGE="This message came from a Conf
igMap!"
error: failed to create configmap: configmaps "app-config" already exists
theia@theiadocker[redacted]/home/project/CC201/labs/3_K8sScaleAndUpdate$
```

2. Use the Explorer to edit `deployment-configmap-env-var.yaml`. The path to this file is `CC201/labs/3_K8sScaleAndUpdate/`. You need to insert your namespace where it says `<my_namespace>`. Make sure to save the file when you're done.


File Edit Selection View Go Run Terminal Help
EXPLORER
OPEN EDITORS
deployment-configmap-env-var.yaml
PROJECT
CC201
labs
1_ContainersAndDocker
2_IntroKubernetes
3_K8sScaleAndUpdate
app.js M
deployment-configmap-env-var... M
deployment.yaml M
Dockerfile
package.json
.gitignore
LICENSE
README.md
deployment-configmap-env-var.yaml
CC201 > labs > 3_K8sScaleAndUpdate > deployment-configmap-env-var.yaml

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: hello-world
spec:
  selector:
    matchLabels:
      run: hello-world
  template:
    metadata:
      labels:
        run: hello-world
    spec:
      containers:
      - name: hello-world
        image: us.icr.io/sn-labs-[redacted]/hello-world:3
        ports:
        - containerPort: 8080
        envFrom:
        - configMapRef:
            name: app-config
      imagePullSecrets:
        - name: icr

```

3. In the same file, notice the section reproduced below. The bottom portion indicates that environment variables should be defined in the container from the data in a ConfigMap named `app-config`.

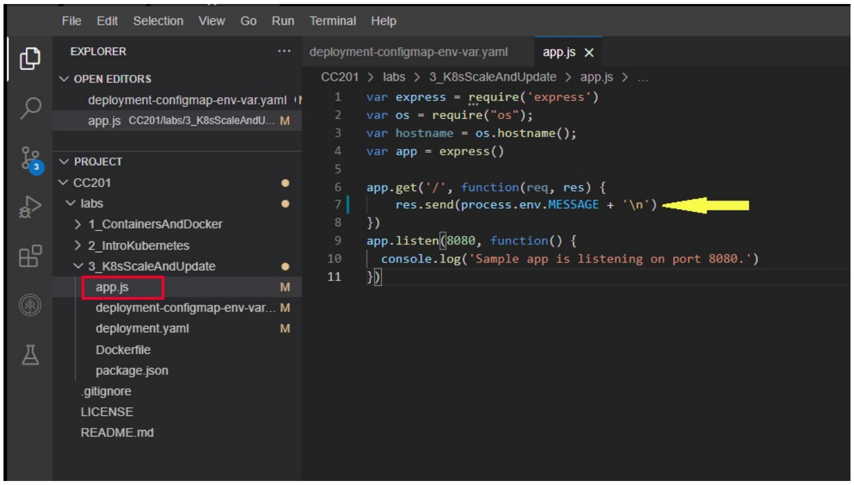
```
containers:
- name: hello-world
  image: us.icr.io/<my_namespace>/hello-world:3
  ports:
  - containerPort: 8080
  envFrom:
  - configMapRef:
    name: app-config
```

4. Use the Explorer to open the `app.js` file. The path to this file is `CC201/labs/3_K8sScaleAndUpdate/`. Find the line that says, `res.send('Welcome to ' + hostname + '! Your app is up and running!\n')`.

Edit this line to look like the following:

```
res.send(process.env.MESSAGE + '\n')
```

Make sure to save the file when you're done. This change indicates that requests to the app will return the environment variable `MESSAGE`.

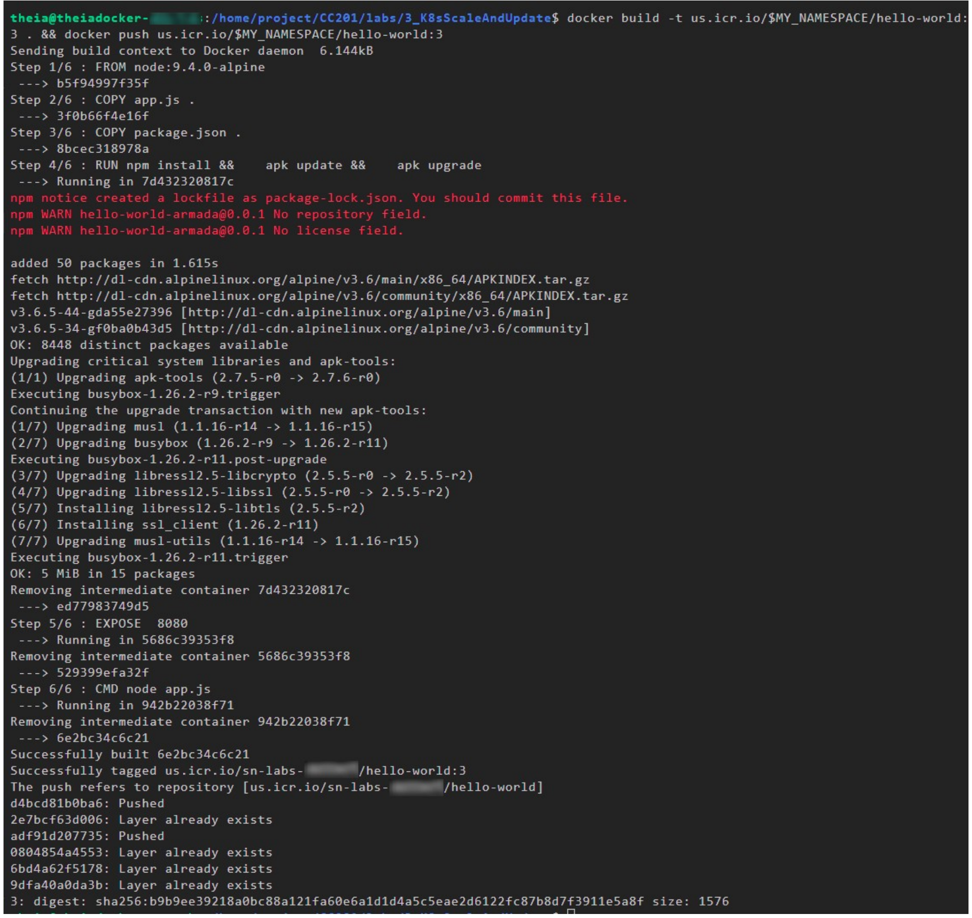

File Edit Selection View Go Run Terminal Help
EXPLORER
OPEN EDITORS
deployment-configmap-env-var.yaml
app.js CC201/labs/3_K8sScaleAndU... M
PROJECT
CC201
labs
1_ContainersAndDocker
2_IntroKubernetes
3_K8sScaleAndUpdate
app.js M
deployment-configmap-env-var... M
deployment.yaml M
Dockerfile
package.json
.gitignore
LICENSE
README.md
deployment-configmap-env-var.yaml
app.js
CC201 > labs > 3_K8sScaleAndUpdate > app.js > ...

```
var express = require('express')
var os = require("os");
var hostname = os.hostname();
var app = express()

app.get('/', function(req, res) {
    res.send(process.env.MESSAGE + '\n')
})
app.listen(8080, function() {
  console.log('Sample app is listening on port 8080.')
})
```

5. Build and push a new image that contains your new application code.

```
docker build -t us.icr.io/$MY_NAMESPACE/hello-world:3 . && docker push us.icr.io/$MY_NAMESPACE/hello-world:3
```

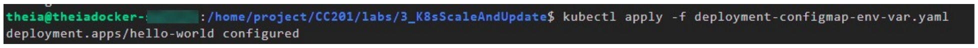
The `deployment-configmap-env-var.yaml` file is already configured to use the tag `3`.



```
theia@theiadocker-[redacted]:/home/project/CC201/labs/3_K8sScaleAndUpdate$ docker build -t us.icr.io/$MY_NAMESPACE/hello-world:
3 . && docker push us.icr.io/$MY_NAMESPACE/hello-world:3
Sending build context to Docker daemon  6.144kB
Step 1/6 : FROM node:9.4.0-alpine
 ---> b5f94997f35f
Step 2/6 : COPY app.js .
 ---> 3f0b66f4e16f
Step 3/6 : COPY package.json .
 ---> 8bcec318978a
Step 4/6 : RUN npm install &&    apk update &&    apk upgrade
 ---> Running in 7d432320817c
npm notice created a lockfile as package-lock.json. You should commit this file.
npm WARN hello-world-armada@0.0.1 No repository field.
npm WARN hello-world-armada@0.0.1 No license field.

added 50 packages in 1.615s
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)
(6/7) Installing ssl_client (1.26.2-r11)
(7/7) Upgrading musl-utils (1.1.16-r14 -> 1.1.16-r15)
Executing busybox-1.26.2-r11.trigger
OK: 5 MiB in 15 packages
Removing intermediate container 7d432320817c
 ---> ed77983749d5
Step 5/6 : EXPOSE  8080
 ---> Running in 5686c39353f8
Removing intermediate container 5686c39353f8
 ---> 529399efa32f
Step 6/6 : CMD node app.js
 ---> Running in 942b22038f71
Removing intermediate container 942b22038f71
 ---> 6e2bc34c6c21
Successfully built 6e2bc34c6c21
Successfully tagged us.icr.io/sn-labs-[redacted]/hello-world:3
The push refers to repository [us.icr.io/sn-labs-[redacted]/hello-world]
d4bcd81b0ba6: Pushed
2e7bcf63d006: Layer already exists
adf91d207735: Pushed
0804854a4553: Layer already exists
6bd4a62f5178: Layer already exists
9dfa40a0da3b: Layer already exists
3: digest: sha256:b9b9ee39218a0bc88a121fa60e6a1d1d4a5c5eae2d6122fc87b8d7f3911e5a8f size: 1576
```

6. Apply the new Deployment configuration.

```
kubectl apply -f deployment-configmap-env-var.yaml
```



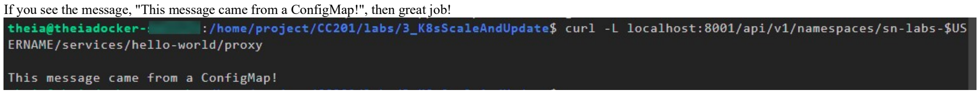
```
theia@theiadocker-[redacted]:/home/project/CC201/labs/3_K8sScaleAndUpdate$ kubectl apply -f deployment-configmap-env-var.yaml
deployment.apps/hello-world configured
```

7. Ping your application again to see if the message from the environment variable is returned.

> **NOTE:** You can run this command again. As it may not show the `"This message came from a ConfigMap!"` message right away.

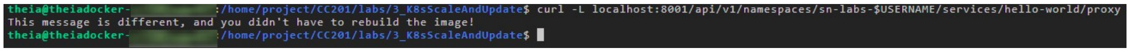
```
curl -L localhost:8001/api/v1/namespaces/sn-labs-$USERNAME/services/hello-world/proxy
```

If you see the message, "This message came from a ConfigMap!", then great job!



```
theia@theiadocker-[redacted]:/home/project/CC201/labs/3_K8sScaleAndUpdate$ curl -L localhost:8001/api/v1/namespaces/sn-labs-$US
ERNAME/services/hello-world/proxy

This message came from a ConfigMap!
```

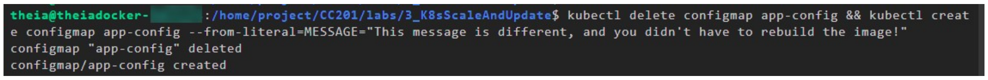
NOTE: If your previous session is still persisting, you might see the below output. If so, we would recommend you to move to the further steps of the lab.



```
theia@theiadocker-[redacted]:/home/project/CC201/labs/3_K8sScaleAndUpdate$ curl -L localhost:8001/api/v1/namespaces/sn-labs-$USERNAME/services/hello-world/proxy
This message is different, and you didn't have to rebuild the image!
theia@theiadocker-[redacted]:/home/project/CC201/labs/3_K8sScaleAndUpdate$
```

8. Because the configuration is separate from the code, the message can be changed without rebuilding the image. Using the following command, delete the old ConfigMap and create a new one with the same name but a different message.

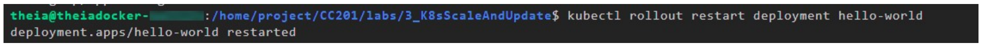
```
kubectl delete configmap app-config && kubectl create configmap app-config --from-literal=MESSAGE="This message is different, and you didn't have to rebuild the image!"
```



```
theia@theiadocker-[redacted]:/home/project/CC201/labs/3_K8sScaleAndUpdate$ kubectl delete configmap app-config && kubectl creat
e configmap app-config --from-literal=MESSAGE="This message is different, and you didn't have to rebuild the image!"
configmap "app-config" deleted
configmap/app-config created
```

9. Restart the Deployment so that the containers restart. This is necessary since the environment variables are set at start time.

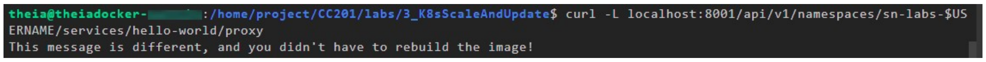
```
kubectl rollout restart deployment hello-world
```



```
theia@theiadocker-[redacted]:/home/project/CC201/labs/3_K8sScaleAndUpdate$ kubectl rollout restart deployment hello-world
deployment.apps/hello-world restarted
```

10. Ping your application again to see if the new message from the environment variable is returned.

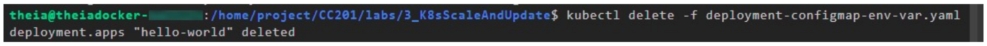
```
curl -L localhost:8001/api/v1/namespaces/sn-labs-$USERNAME/services/hello-world/proxy
```



```
theia@theiadocker-[redacted]:/home/project/CC201/labs/3_K8sScaleAndUpdate$ curl -L localhost:8001/api/v1/namespaces/sn-labs-$US
ERNAME/services/hello-world/proxy
This message is different, and you didn't have to rebuild the image!
```

11. Delete the Deployment.

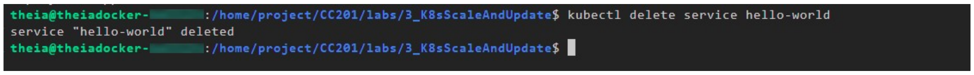
```
kubectl delete -f deployment-configmap-env-var.yaml
```



```
theia@theiadocker-[redacted]:/home/project/CC201/labs/3_K8sScaleAndUpdate$ kubectl delete -f deployment-configmap-env-var.yaml
deployment.apps "hello-world" deleted
```

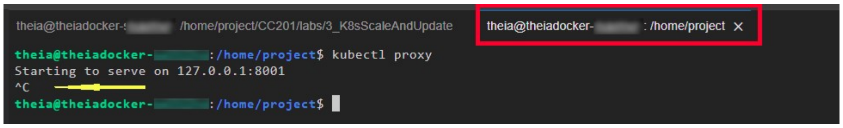
12. Delete the Service.

```
kubectl delete service hello-world
```



```
theia@theiadocker-[redacted]:/home/project/CC201/labs/3_K8sScaleAndUpdate$ kubectl delete service hello-world
service "hello-world" deleted
theia@theiadocker-[redacted]:/home/project/CC201/labs/3_K8sScaleAndUpdate$
```

13. Return to the other terminal window that is running the `proxy` command and kill it using `Ctrl+C`.



theia@theiadocker-[redacted] /home/project/CC201/labs/3_K8sScaleAndUpdate
theia@theiadocker-[redacted] : /home/project ×

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

Congratulations! You have completed the lab for the third module of this course.

> **Note:** Please delete your project from SN labs environment before signing out to ensure that further labs run correctly. To do the same, click on this link

Changelog

Date	Version	Changed by	Change Description
2022-04-07	1.1	Samaah Sarang	Updated Lab instructions & images
2022-04-13	1.2	Samaah Sarang	Updated Lab instructions
2022-04-14	1.3	K Sundararajan	Updated Lab instructions & images
2022-04-18	1.4	K Sundararajan	Updated Lab instructions
2022-04-19	1.5	K Sundararajan	Updated Lab instructions

Â© IBM Corporation 2022. All rights reserved.

Continue