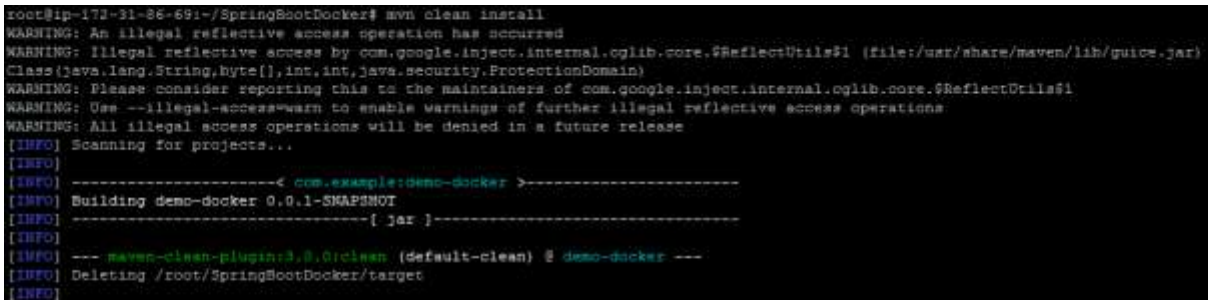
Creating a custom Docker image

- Follow the set of commands shown below to build a custom Docker image:

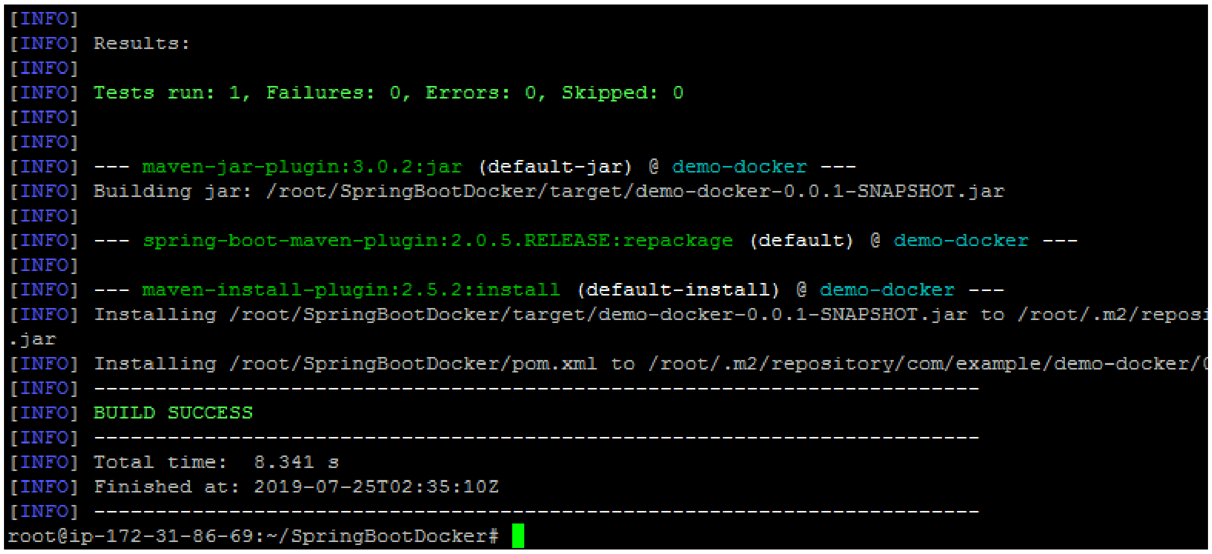
```
git clone https://github.com/ SpringBootDocker.git

ls -lart
```

- Build source code to generate artifacts which can be deployed on Docker host.

  mvn clean install



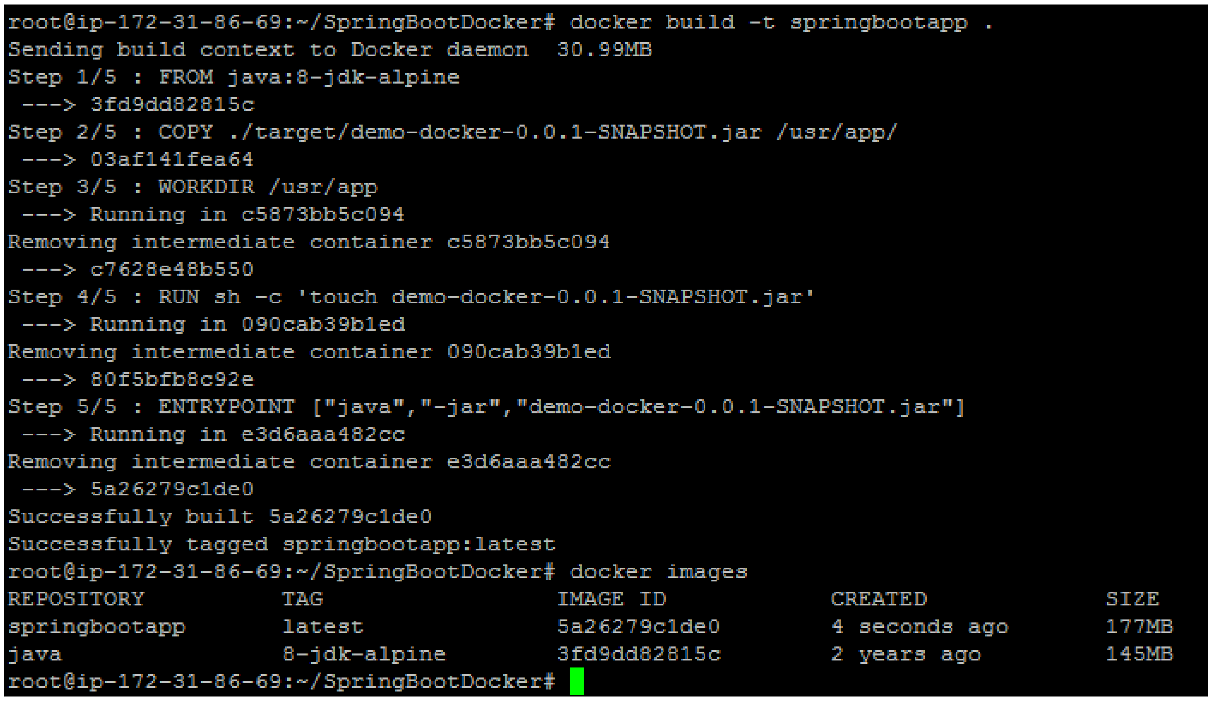
```
root@ip-172-31-86-69:~/SpringBootDocker# mvn clean install
WARNING: An illegal reflective access operation has occurred
WARNING: Illegal reflective access by com.google.inject.internal.cglib.core.$ReflectUtils$1 (file:/usr/share/maven/lib/guice.jar)
Class(java.lang.String,byte[],int,int,java.security.ProtectionDomain)
WARNING: Please consider reporting this to the maintainers of com.google.inject.internal.cglib.core.$ReflectUtils$1
WARNING: Use --illegal-access=warn to enable warnings of further illegal reflective access operations
WARNING: All illegal access operations will be denied in a future release
[INFO] Scanning for projects...
[INFO]
[INFO] ----------------------< com.example:demo-docker >----------------------
[INFO] Building demo-docker 0.0.1-SNAPSHOT
[INFO] --------------------------------[ jar ]---------------------------------
[INFO]
[INFO] --- maven-clean-plugin:3.0.0:clean (default-clean) @ demo-docker ---
[INFO] Deleting /root/SpringBootDocker/target
[INFO]
```



```
[INFO]
[INFO] Results:
[INFO]
[INFO] Tests run: 1, Failures: 0, Errors: 0, Skipped: 0
[INFO]
[INFO]
[INFO] --- maven-jar-plugin:3.0.2:jar (default-jar) @ demo-docker ---
[INFO] Building jar: /root/SpringBootDocker/target/demo-docker-0.0.1-SNAPSHOT.jar
[INFO]
[INFO] --- spring-boot-maven-plugin:2.0.5.RELEASE:repackage (default) @ demo-docker ---
[INFO]
[INFO] --- maven-install-plugin:2.5.2:install (default-install) @ demo-docker ---
[INFO] Installing /root/SpringBootDocker/target/demo-docker-0.0.1-SNAPSHOT.jar to /root/.m2/reposi
.jar
[INFO] Installing /root/SpringBootDocker/pom.xml to /root/.m2/repository/com/example/demo-docker/0
[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESS
[INFO] ------------------------------------------------------------------------
[INFO] Total time:  8.341 s
[INFO] Finished at: 2019-07-25T02:35:10Z
[INFO] ------------------------------------------------------------------------
root@ip-172-31-86-69:~/SpringBootDocker#
```

- Deploy this artifact inside the custom Docker image using **docker build** command line. Follow the steps shown below to create the custom Docker image:

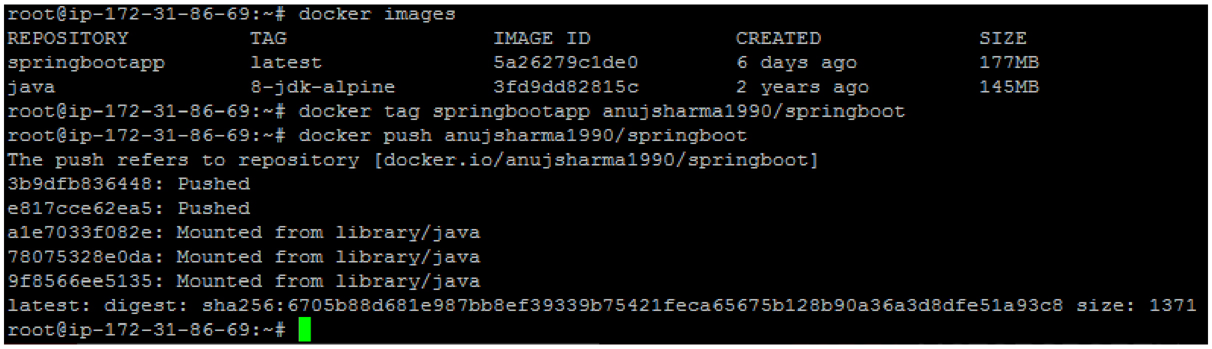
```
docker build -t springbootapp .
```



```
root@ip-172-31-86-69:~/SpringBootDocker# docker build -t springbootapp .
Sending build context to Docker daemon  30.99MB
Step 1/5 : FROM java:8-jdk-alpine
 ---> 3fd9dd82815c
Step 2/5 : COPY ./target/demo-docker-0.0.1-SNAPSHOT.jar /usr/app/
 ---> 03af141fea64
Step 3/5 : WORKDIR /usr/app
 ---> Running in c5873bb5c094
Removing intermediate container c5873bb5c094
 ---> c7628e48b550
Step 4/5 : RUN sh -c 'touch demo-docker-0.0.1-SNAPSHOT.jar'
 ---> Running in 090cab39b1ed
Removing intermediate container 090cab39b1ed
 ---> 80f5bfb8c92e
Step 5/5 : ENTRYPOINT ["java","-jar","demo-docker-0.0.1-SNAPSHOT.jar"]
 ---> Running in e3d6aaa482cc
Removing intermediate container e3d6aaa482cc
 ---> 5a26279c1de0
Successfully built 5a26279c1de0
Successfully tagged springbootapp:latest
root@ip-172-31-86-69:~/SpringBootDocker# docker images
REPOSITORY          TAG                 IMAGE ID            CREATED             SIZE
springbootapp       latest              5a26279c1de0        4 seconds ago       177MB
java                8-jdk-alpine        3fd9dd82815c        2 years ago         145MB
root@ip-172-31-86-69:~/SpringBootDocker#
```

- Push this image to Docker Hub. Follow the command below to do so.

```
docker images

docker tag springbootapp anujsharma1990/springboot

docker push anujsharma1990/springboot
```



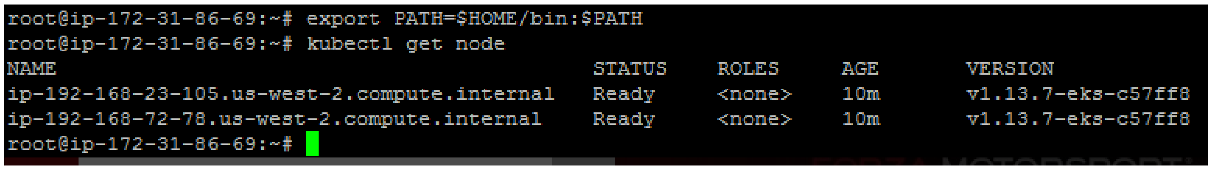
```
root@ip-172-31-86-69:~# docker images
REPOSITORY          TAG                 IMAGE ID            CREATED             SIZE
springbootapp       latest              5a26279c1de0        6 days ago          177MB
java                8-jdk-alpine        3fd9dd82815c        2 years ago         145MB
root@ip-172-31-86-69:~# docker tag springbootapp anujsharma1990/springboot
root@ip-172-31-86-69:~# docker push anujsharma1990/springboot
The push refers to repository [docker.io/anujsharma1990/springboot]
3b9dfb836448: Pushed
e817cce62ea5: Pushed
a1e7033f082e: Mounted from library/java
78075328e0da: Mounted from library/java
9f8566ee5135: Mounted from library/java
latest: digest: sha256:6705b88d681e987bb8ef39339b75421feca65675b128b90a36a3d8dfe51a93c8 size: 1371
root@ip-172-31-86-69:~#
```

Deploying a Spring Boot application to AWS EKS

- Configure **kubectl command line** and deploy containers to AWS EKS.

export PATH=$HOME/bin:$PATH

kubectl get node

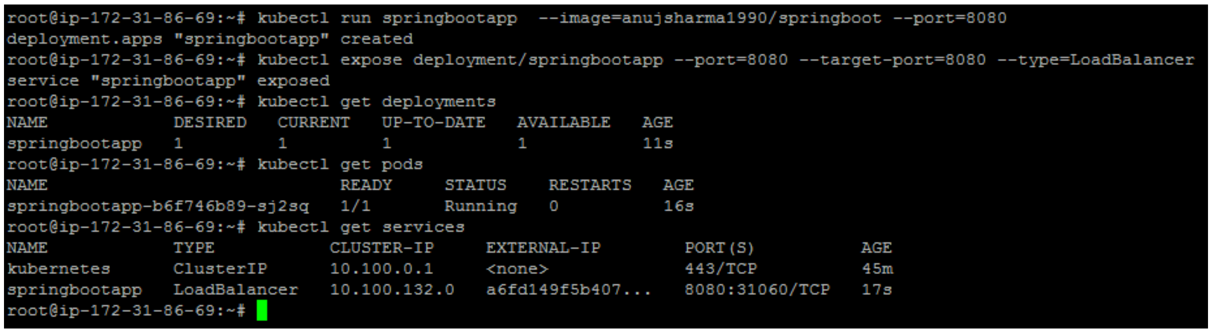


```
root@ip-172-31-86-69:~# export PATH=$HOME/bin:$PATH
root@ip-172-31-86-69:~# kubectl get node
NAME                                          STATUS    ROLES     AGE       VERSION
ip-192-168-23-105.us-west-2.compute.internal  Ready     <none>    10m       v1.13.7-eks-c57ff8
ip-192-168-72-78.us-west-2.compute.internal   Ready     <none>    10m       v1.13.7-eks-c57ff8
root@ip-172-31-86-69:~#
```

- Create Kubernetes deployment and service using the set of commands given below:

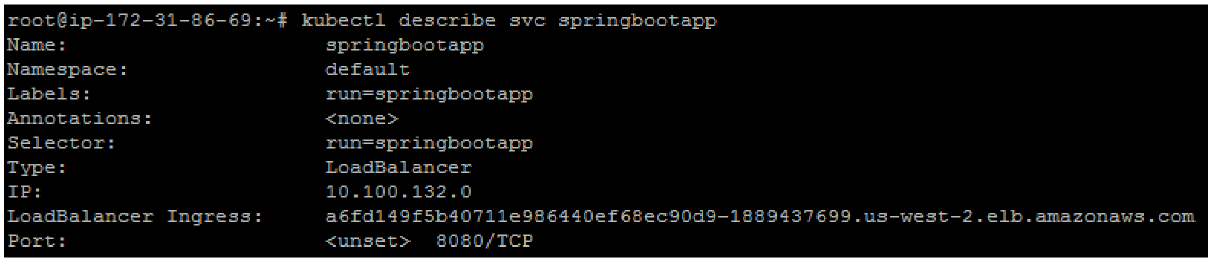
kubectl run springbootapp--image=anujsharma1990/springboot --port=8080

kubectl expose deployment/springbootapp --port=8080 --target-port=8080 --type=LoadBalancer



```
root@ip-172-31-86-69:~# kubectl run springbootapp  --image=anujsharma1990/springboot --port=8080
deployment.apps "springbootapp" created
root@ip-172-31-86-69:~# kubectl expose deployment/springbootapp --port=8080 --target-port=8080 --type=LoadBalancer
service "springbootapp" exposed
root@ip-172-31-86-69:~# kubectl get deployments
NAME            DESIRED   CURRENT   UP-TO-DATE   AVAILABLE   AGE
springbootapp   1         1         1            1           11s
root@ip-172-31-86-69:~# kubectl get pods
NAME                            READY     STATUS    RESTARTS   AGE
springbootapp-b6f746b89-sj2sq   1/1       Running   0          16s
root@ip-172-31-86-69:~# kubectl get services
NAME            TYPE           CLUSTER-IP     EXTERNAL-IP        PORT(S)          AGE
kubernetes      ClusterIP      10.100.0.1     <none>             443/TCP          45m
springbootapp   LoadBalancer   10.100.132.0   a6fd149f5b407...   8080:31060/TCP   17s
root@ip-172-31-86-69:~#
```

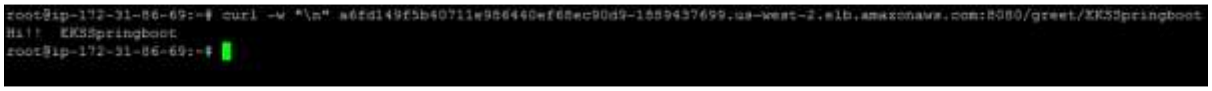
Please Note: Once the pod is deployed, we can get the Load Balancer URL from springbootapp EKS Service. EKS will automatically configure the Load Balancer in AWS.



```
root@ip-172-31-86-69:~# kubectl describe svc springbootapp
Name:                     springbootapp
Namespace:                default
Labels:                   run=springbootapp
Annotations:              <none>
Selector:                 run=springbootapp
Type:                     LoadBalancer
IP:                       10.100.132.0
LoadBalancer Ingress:     a6fd149f5b40711e986440ef68ec90d9-1889437699.us-west-2.elb.amazonaws.com
Port:                     <unset>  8080/TCP
```

- To access the Spring Boot application, use the **Load Balancer URL** as shown below.

curl -w "\n" a6fd149f5b40711e986440ef68ec90d9-1889437699.us-west-2.elb.amazonaws.com:8080/greet/EKSSpringboot



```
root@ip-172-31-86-69:~# curl -w "\n" a6fd149f5b40711e986440ef68ec90d9-1889437699.us-west-2.elb.amazonaws.com:8080/greet/EKSSpringboot
Hi!!  EKSSpringboot
root@ip-172-31-86-69:~# 
```