



INDIAN INSTITUTE OF
INFORMATION
TECHNOLOGY

DevOps & its Applications (CS457)

Assignment-1

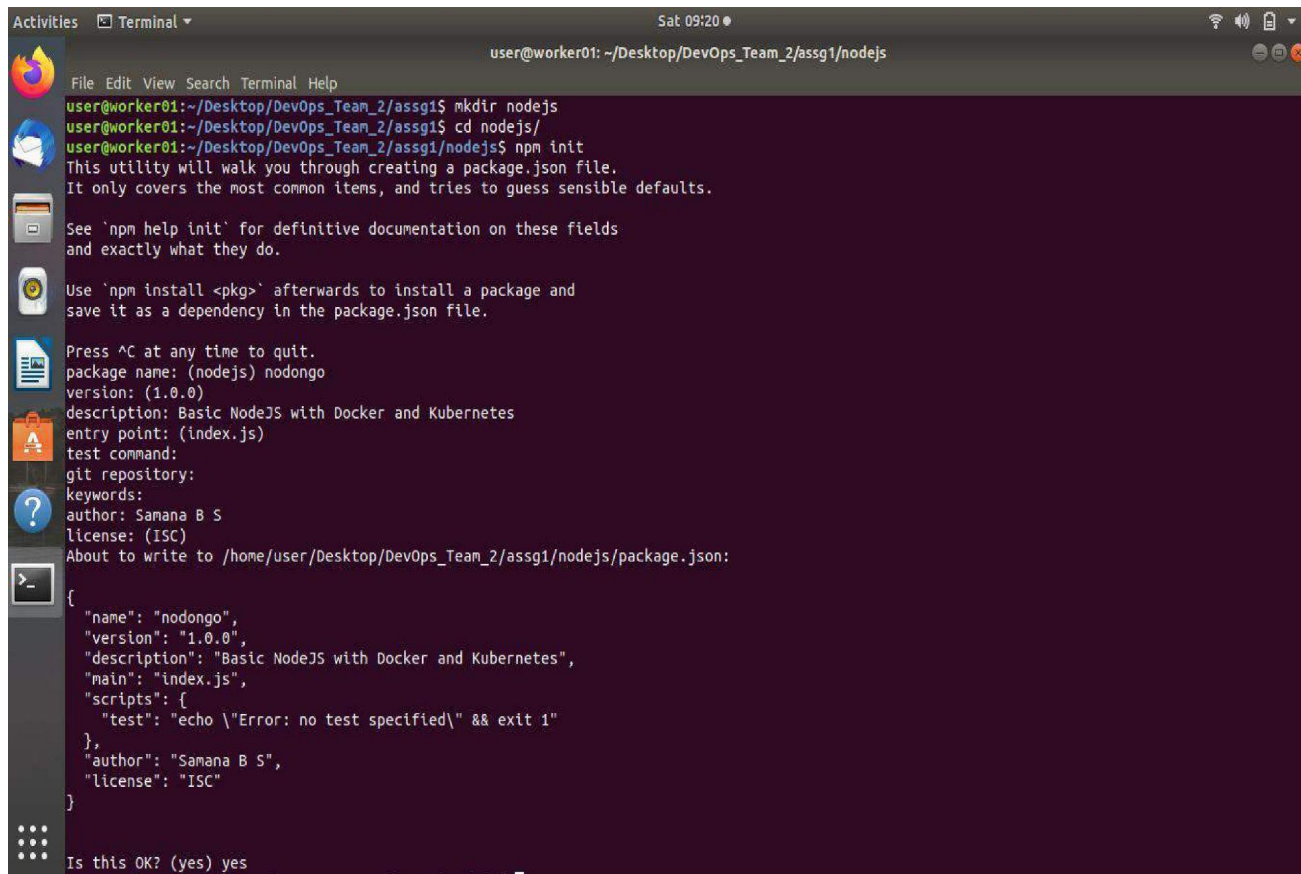
Under the guidance - Dr. Uma S

Submitted by:

Advay Aggarwal(18BCS002)
Meghana Hadimani(18BCS002)
Perumulla Tushar(18BCS065)
Rahul Priyadarshi(18BCS074)
Samana B S(18BCS088)

Developing and deploying a Node.js app from Docker to Kubernetes.

Step1: Making a separate directory and initializing The Node Application.

A terminal window titled 'Terminal' with a dark background. The user is at a prompt 'user@worker01: ~/Desktop/DevOps_Team_2/assg1/nodejs'. The terminal shows the execution of 'mkdir nodejs', 'cd nodejs', and 'npm init'. The 'npm init' process prompts for package name, version, description, entry point, test command, git repository, keywords, author, and license. The user provides values for most of these, and the terminal displays the resulting 'package.json' file content. The prompt 'Is this OK? (yes) yes' is visible at the bottom.

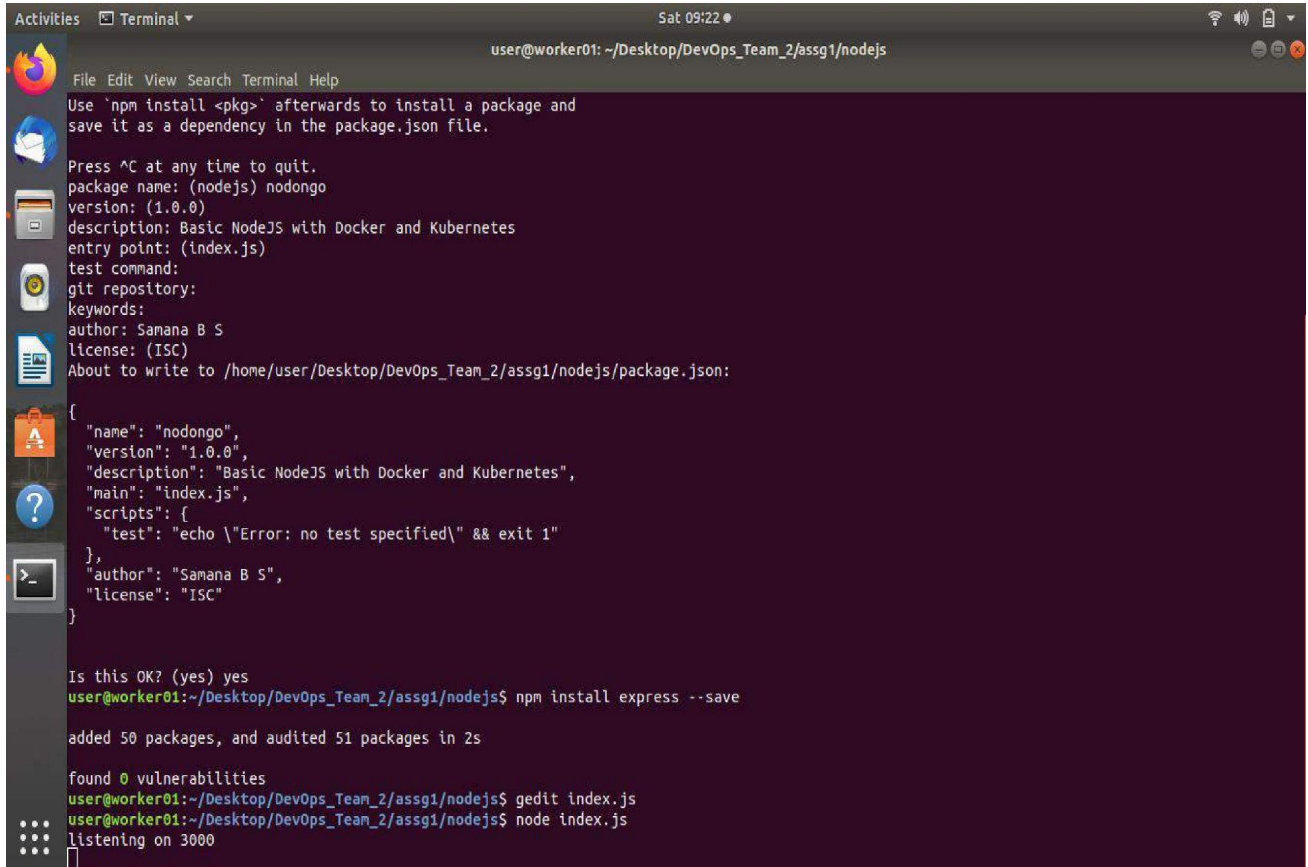
```
user@worker01: ~/Desktop/DevOps_Team_2/assg1/nodejs
user@worker01:~/Desktop/DevOps_Team_2/assg1$ mkdir nodejs
user@worker01:~/Desktop/DevOps_Team_2/assg1$ cd nodejs/
user@worker01:~/Desktop/DevOps_Team_2/assg1/nodejs$ npm init
This utility will walk you through creating a package.json file.
It only covers the most common items, and tries to guess sensible defaults.

See 'npm help init' for definitive documentation on these fields
and exactly what they do.

Use 'npm install <pkg>' afterwards to install a package and
save it as a dependency in the package.json file.

Press ^C at any time to quit.
package name: (nodejs) nodongo
version: (1.0.0)
description: Basic NodeJS with Docker and Kubernetes
entry point: (index.js)
test command:
git repository:
keywords:
author: Samana B S
license: (ISC)
About to write to /home/user/Desktop/DevOps_Team_2/assg1/nodejs/package.json:
{
  "name": "nodongo",
  "version": "1.0.0",
  "description": "Basic NodeJS with Docker and Kubernetes",
  "main": "index.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "author": "Samana B S",
  "license": "ISC"
}
Is this OK? (yes) yes
```

Step2: Installing Express



A terminal window titled "Terminal" with a dark background and light text. The window shows the process of installing Express.js and running a Node.js application. The user is in a directory named ~/Desktop/DevOps_Team_2/assg1/nodejs. The terminal displays the output of the 'npm install' command, which includes the package name, version, description, entry point, test command, git repository, keywords, author, and license. The user then runs 'npm install express --save', which adds 50 packages and audits 51 packages in 2 seconds. Finally, the user runs 'node index.js', which starts a server listening on port 3000.

```
user@worker01: ~/Desktop/DevOps_Team_2/assg1/nodejs

File Edit View Search Terminal Help
Use 'npm install <pkg>' afterwards to install a package and
save it as a dependency in the package.json file.

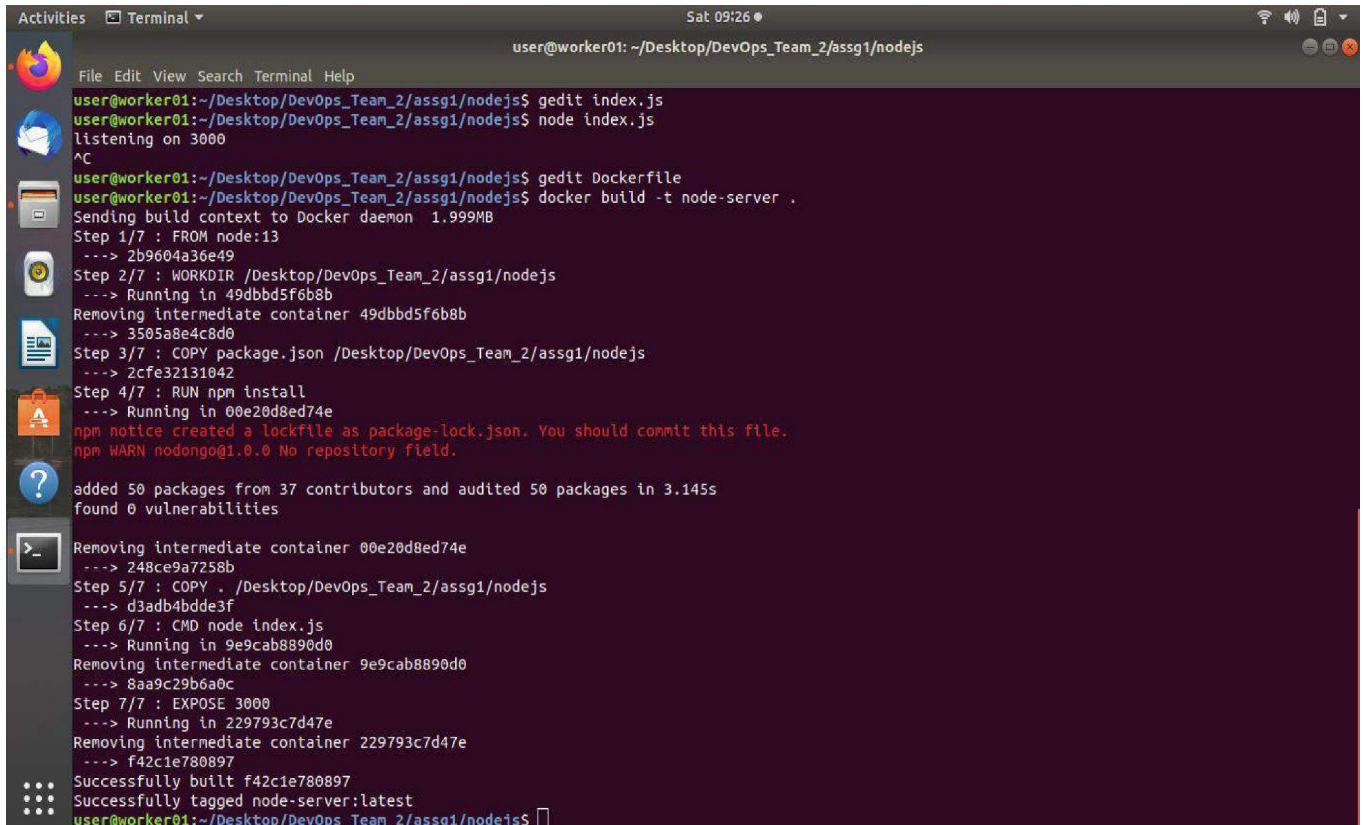
Press ^C at any time to quit.
package name: (nodejs) nodongo
version: (1.0.0)
description: Basic NodeJS with Docker and Kubernetes
entry point: (index.js)
test command:
git repository:
keywords:
author: Samana B S
license: (ISC)
About to write to /home/user/Desktop/DevOps_Team_2/assg1/nodejs/package.json:

{
  "name": "nodongo",
  "version": "1.0.0",
  "description": "Basic NodeJS with Docker and Kubernetes",
  "main": "index.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "author": "Samana B S",
  "license": "ISC"
}

Is this OK? (yes) yes
user@worker01:~/Desktop/DevOps_Team_2/assg1/nodejs$ npm install express --save
added 50 packages, and audited 51 packages in 2s

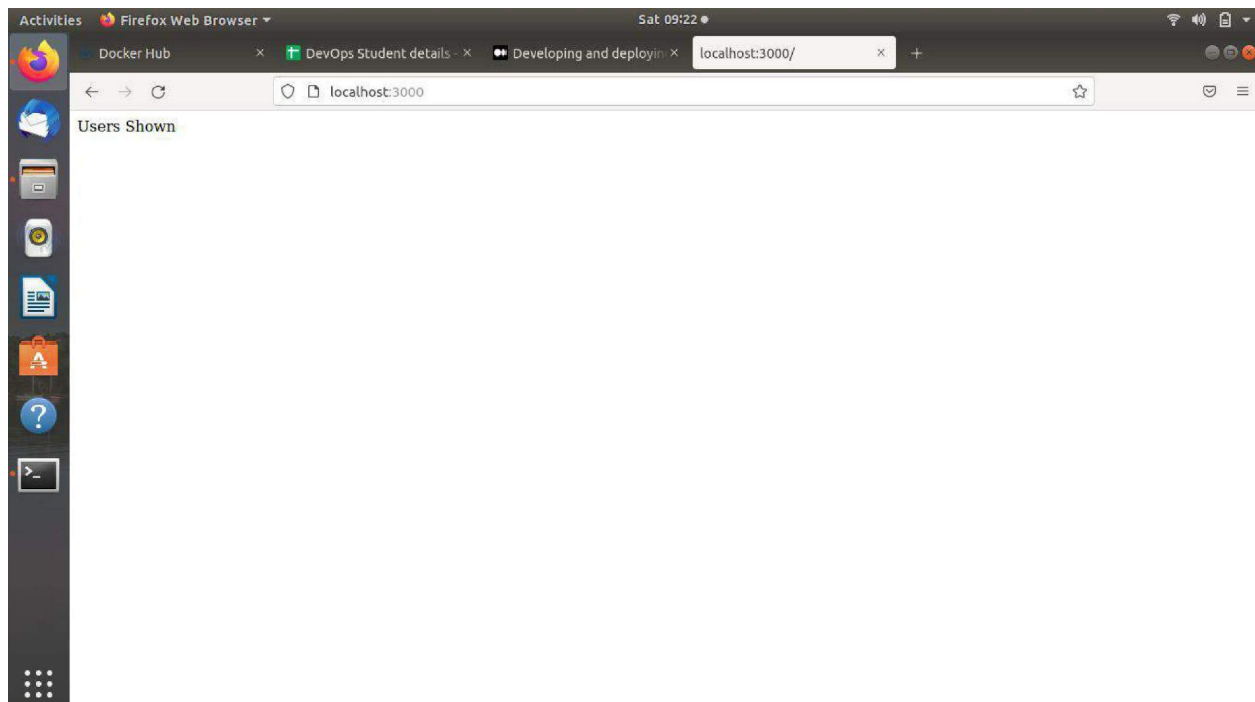
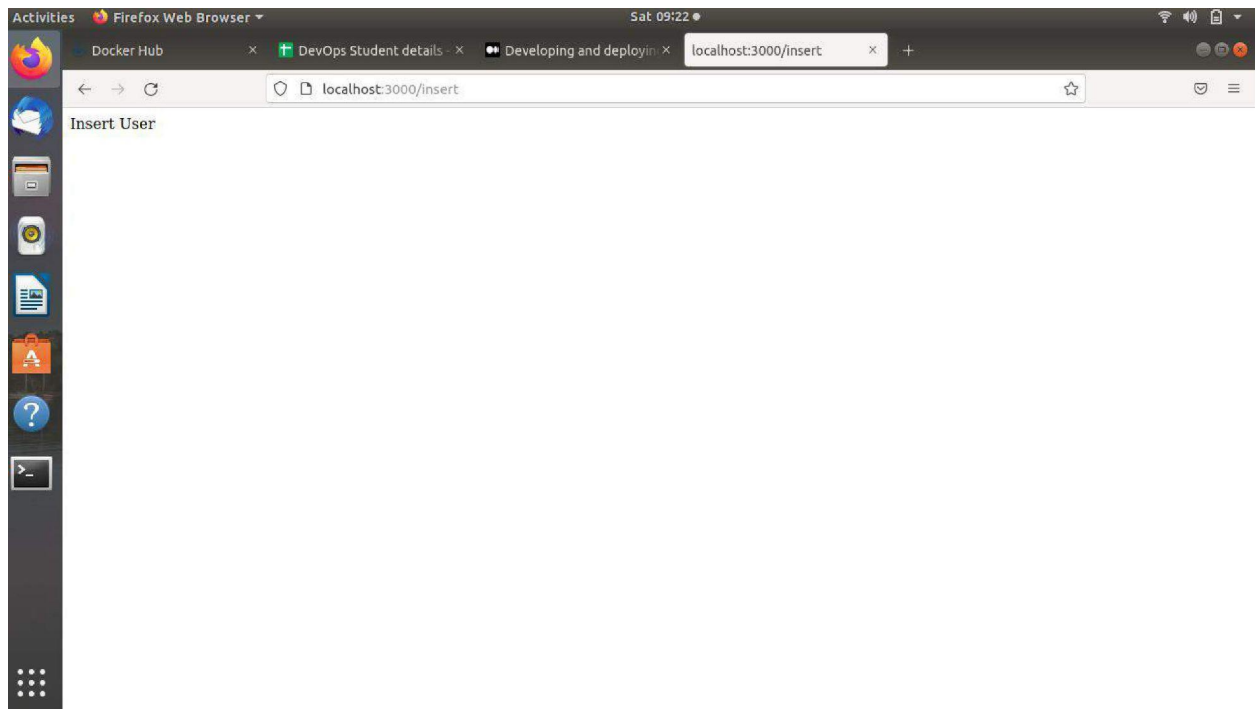
found 0 vulnerabilities
user@worker01:~/Desktop/DevOps_Team_2/assg1/nodejs$ gedit index.js
user@worker01:~/Desktop/DevOps_Team_2/assg1/nodejs$ node index.js
listening on 3000
```

Step3 and Step4: Making Index.js file and writing some code in it to test the application on the kubernetes cluster, then dockerizing the node server for which we create a Dockerfile to build the image as the server and the code is ready to deploy.



```
user@worker01: ~/Desktop/DevOps_Team_2/assg1/nodejs
user@worker01:~/Desktop/DevOps_Team_2/assg1/nodejs$ gedit index.js
user@worker01:~/Desktop/DevOps_Team_2/assg1/nodejs$ node index.js
listening on 3000
^C
user@worker01:~/Desktop/DevOps_Team_2/assg1/nodejs$ gedit Dockerfile
user@worker01:~/Desktop/DevOps_Team_2/assg1/nodejs$ docker build -t node-server .
Sending build context to Docker daemon 1.999MB
Step 1/7 : FROM node:13
--> 2b9604a36e49
Step 2/7 : WORKDIR /Desktop/DevOps_Team_2/assg1/nodejs
--> Running in 49dbbd5f6b8b
Removing intermediate container 49dbbd5f6b8b
--> 3505a8e4c8d0
Step 3/7 : COPY package.json /Desktop/DevOps_Team_2/assg1/nodejs
--> 2cfe32131042
Step 4/7 : RUN npm install
--> Running in 00e20d8ed74e
npm notice created a lockfile as package-lock.json. You should commit this file.
npm WARN nodongo@1.0.0 No repository field.
added 50 packages from 37 contributors and audited 50 packages in 3.145s
found 0 vulnerabilities
Removing intermediate container 00e20d8ed74e
--> 248ce9a7258b
Step 5/7 : COPY . /Desktop/DevOps_Team_2/assg1/nodejs
--> d3adb4bdde3f
Step 6/7 : CMD node index.js
--> Running in 9e9cab8890d0
Removing intermediate container 9e9cab8890d0
--> 8aa9c29b6a0c
Step 7/7 : EXPOSE 3000
--> Running in 229793c7d47e
Removing intermediate container 229793c7d47e
--> f42c1e780897
Successfully built f42c1e780897
Successfully tagged node-server:latest
user@worker01:~/Desktop/DevOps_Team_2/assg1/nodejs$
```

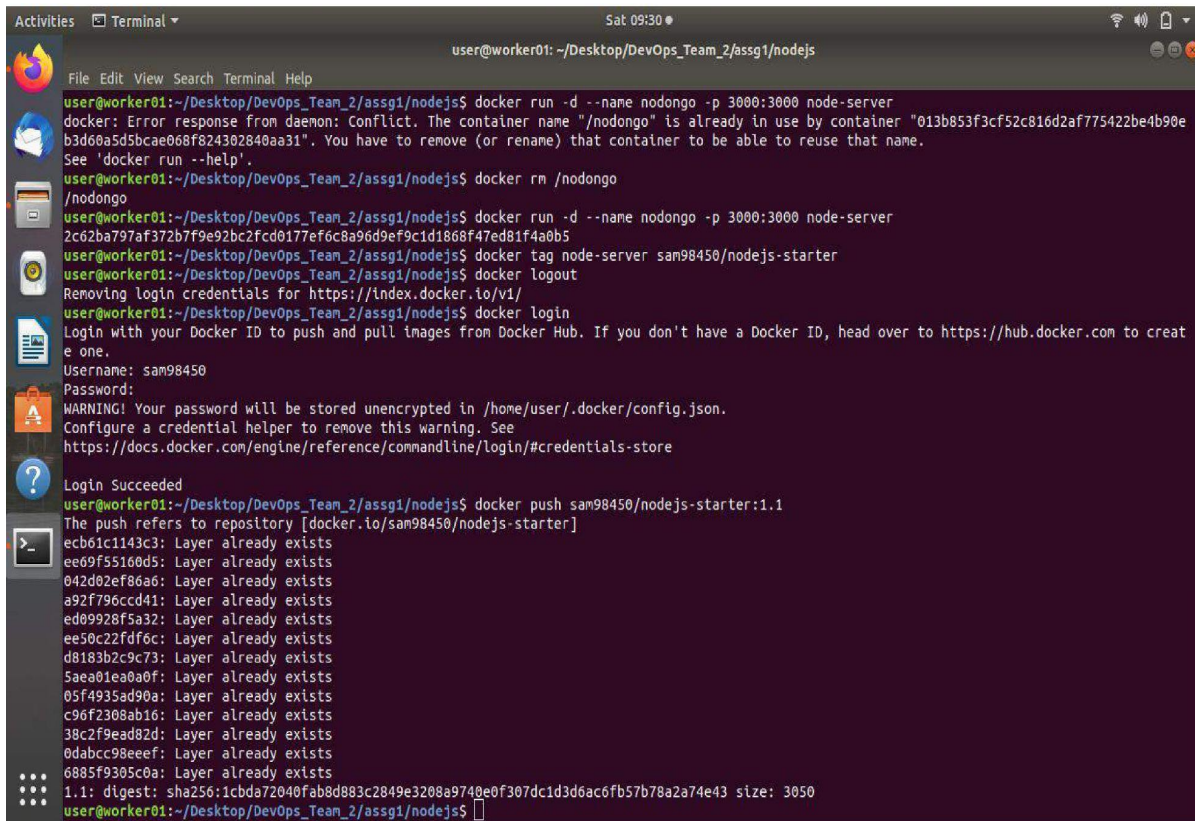
browsing localhost:3000/



Step5 and step6: Now we create and run the container to ensure it works as intended and then we Upload the image to Docker registry Docker Hub

Creating a repo: we have named the repository as Nodejs-starter

We've tagged our existing docker image node-server to samana/nodejs-starter. After that we've pushed our docker image to the registry by using a docker push and tagged it with the 1.1 version.



```
user@worker01: ~/Desktop/DevOps_Team_2/assg1/nodejs$ docker run -d --name nodongo -p 3000:3000 node-server
docker: Error response from daemon: Conflict. The container name "/nodongo" is already in use by container "013b853f3cf52c816d2af775422be4b90e
b3d60a5d5bcae068f824302840aa31". You have to remove (or rename) that container to be able to reuse that name.
See 'docker run --help'.
user@worker01:~/Desktop/DevOps_Team_2/assg1/nodejs$ docker rm /nodongo
/nodongo
user@worker01:~/Desktop/DevOps_Team_2/assg1/nodejs$ docker run -d --name nodongo -p 3000:3000 node-server
2c62ba797af372b7f9e92bc2fcd0177ef6c8a96d9ef9c1d1868f47ed81f4a0b5
user@worker01:~/Desktop/DevOps_Team_2/assg1/nodejs$ docker tag node-server sam98450/nodejs-starter
user@worker01:~/Desktop/DevOps_Team_2/assg1/nodejs$ docker logout
Removing login credentials for https://index.docker.io/v1/
user@worker01:~/Desktop/DevOps_Team_2/assg1/nodejs$ docker login
Login with your Docker ID to push and pull images from Docker Hub. If you don't have a Docker ID, head over to https://hub.docker.com to creat
e one.
Username: sam98450
Password:
WARNING! Your password will be stored unencrypted in /home/user/.docker/config.json.
Configure a credential helper to remove this warning. See
https://docs.docker.com/engine/reference/commandline/login/#credentials-store
Login Succeeded
user@worker01:~/Desktop/DevOps_Team_2/assg1/nodejs$ docker push sam98450/nodejs-starter:1.1
The push refers to repository [docker.io/sam98450/nodejs-starter]
ecb61c1143c3: Layer already exists
ee69f55160d5: Layer already exists
042d02ef86a6: Layer already exists
a92f796ccd41: Layer already exists
ed09928f5a32: Layer already exists
ee50c222df6c: Layer already exists
d8183b2c9c73: Layer already exists
5aea01ea0a0f: Layer already exists
05f4935ad90a: Layer already exists
c96f2308ab16: Layer already exists
38c2f9ead82d: Layer already exists
0dabcc98eeef: Layer already exists
6885f9305c0a: Layer already exists
1.1: digest: sha256:1cbda72040fab8d883c2849e3208a9740e0f307dc1d3d6ac6fb57b78a2a74e43 size: 3050
user@worker01:~/Desktop/DevOps_Team_2/assg1/nodejs$
```

Activities

Firefox Web Browser

Sat 15:56

Docker Hub

DevOps Studen

Developing and

Error from servi

Compress PDF

Screenshot from 20

Inbox (1,005)

← → ↻


https://hub.docker.com/repository/docker/sam98450/nodejs-starter/general


☆


🔒

⌵

☰

 **sam98450 / nodejs-starter**


This repository does not have a description 

 Last pushed: 2 days ago



Docker commands [Public View](#)

To push a new tag to this repository,

`docker push sam98450/nodejs-starter:tagname`

Tags and Scans  **VULNERABILITY SCANNING - DISABLED** [Enable](#)

This repository contains 1 tag(s).

TAG	OS	PULLED	PUSHED
 1.1		2 days ago	2 days ago



[See all](#)

Automated Builds

Manually pushing Images to Hub? Connect your account to GitHub or Bitbucket to automatically build and tag new Images whenever your code is updated, so you can focus your time on creating.

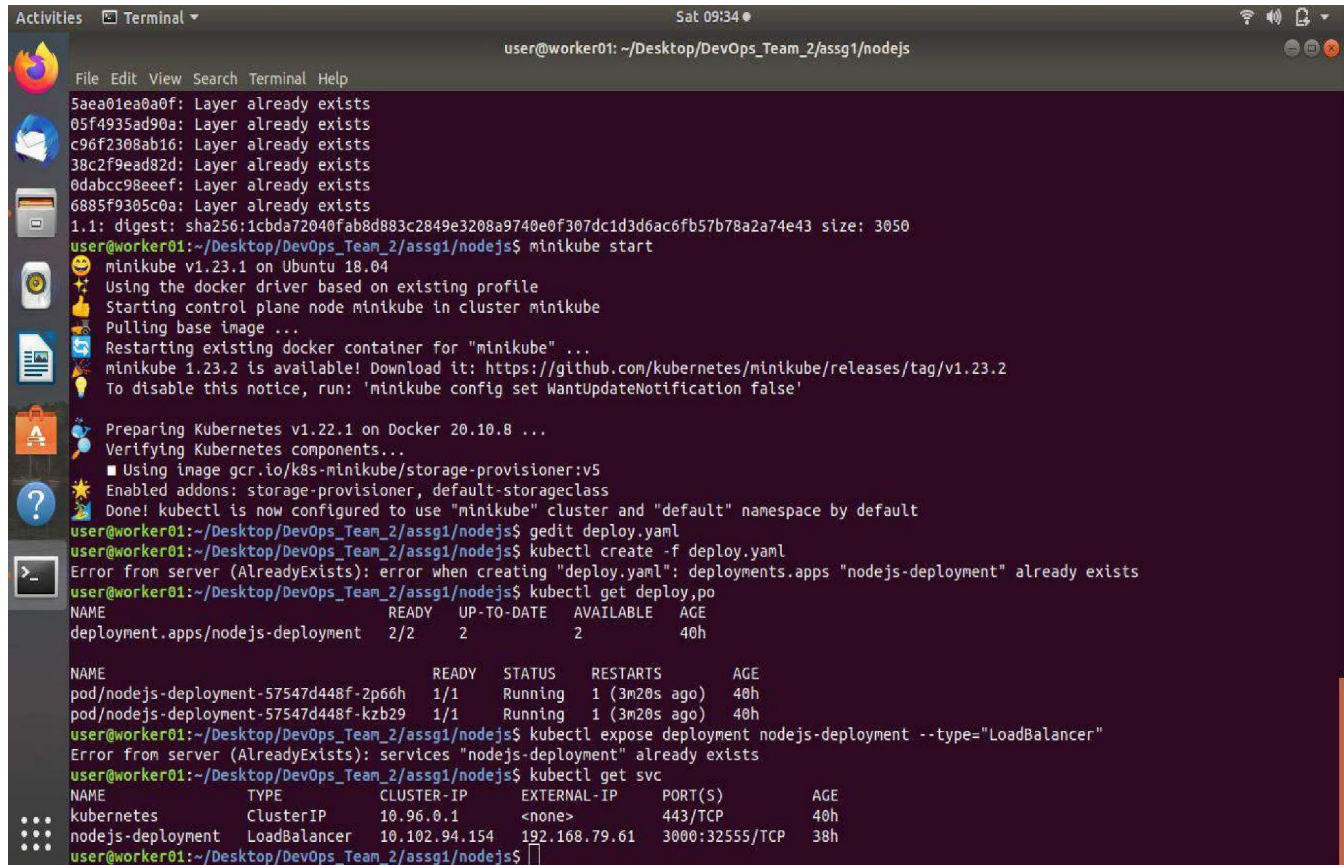
Available on Pro and Team plans.

[Upgrade to Pro](#) [Learn more](#)

Readme  

Repository description is empty. Click [here](#) to edit.

Step7, Step8 and Step9: Start the kubernetes cluster, Defining a YAML file to create a Deployment in kubernetes cluster and As we've created the YAML file, we can go ahead and create a deployment from this YAML file.

A terminal window titled 'Terminal' showing the process of starting a minikube cluster and creating a deployment. The user is at the prompt 'user@worker01: ~/Desktop/DevOps_Team_2/assg1/nodejs'. The terminal output shows minikube starting, pulling the base image, and configuring the cluster. It then shows the user creating a deployment named 'nodejs-deployment' using 'kubectl create -f deploy.yaml'. The deployment is created successfully and is in a 'Running' state. The user then runs 'kubectl get deploy,po' which shows the deployment and its pods. Finally, the user runs 'kubectl expose deployment nodejs-deployment --type="LoadBalancer"' to expose the deployment as a service. The terminal output is as follows:

```
5aea01ea0a0f: Layer already exists
05f4935ad90a: Layer already exists
c96f2308ab16: Layer already exists
38c2f9ead82d: Layer already exists
0dabcc98eeef: Layer already exists
6885f9305c0a: Layer already exists
1.1: digest: sha256:1cbda72040fab8d883c2849e3208a9740e0f307dc1d3d6ac6fb57b78a2a74e43 size: 3050
user@worker01:~/Desktop/DevOps_Team_2/assg1/nodejs$ minikube start
minikube v1.23.1 on Ubuntu 18.04
Using the docker driver based on existing profile
Starting control plane node minikube in cluster minikube
Pulling base image ...
Restarting existing docker container for "minikube" ...
minikube 1.23.2 is available! Download it: https://github.com/kubernetes/minikube/releases/tag/v1.23.2
To disable this notice, run: 'minikube config set WantUpdateNotification false'

Preparing Kubernetes v1.22.1 on Docker 20.10.8 ...
Verifying Kubernetes components...
  ■ Using image gcr.io/k8s-minikube/storage-provisioner:v5
  ■ Enabled addons: storage-provisioner, default-storageclass
  ■ Done! kubectl is now configured to use "minikube" cluster and "default" namespace by default
user@worker01:~/Desktop/DevOps_Team_2/assg1/nodejs$ gedit deploy.yaml
user@worker01:~/Desktop/DevOps_Team_2/assg1/nodejs$ kubectl create -f deploy.yaml
Error from server (AlreadyExists): error when creating "deploy.yaml": deployments.apps "nodejs-deployment" already exists
user@worker01:~/Desktop/DevOps_Team_2/assg1/nodejs$ kubectl get deploy,po
NAME                                READY    UP-TO-DATE    AVAILABLE    AGE
deployment.apps/nodejs-deployment    2/2      2              2            40h

NAME                                READY    STATUS      RESTARTS      AGE
pod/nodejs-deployment-57547d448f-2p66h 1/1      Running     1 (3m20s ago) 40h
pod/nodejs-deployment-57547d448f-kzb29 1/1      Running     1 (3m20s ago) 40h
user@worker01:~/Desktop/DevOps_Team_2/assg1/nodejs$ kubectl expose deployment nodejs-deployment --type="LoadBalancer"
Error from server (AlreadyExists): services "nodejs-deployment" already exists
user@worker01:~/Desktop/DevOps_Team_2/assg1/nodejs$ kubectl get svc
NAME            TYPE        CLUSTER-IP    EXTERNAL-IP    PORT(S)          AGE
kubernetes      ClusterIP   10.96.0.1     <none>         443/TCP          40h
nodejs-deployment LoadBalancer 10.102.94.154 192.168.79.61 3000:32555/TCP   38h
user@worker01:~/Desktop/DevOps_Team_2/assg1/nodejs$
```


Step10: Now the next step is to expose the Deployment to the internet. **Kubectl expose** is used to expose Deployment named nodejs-deployment of the type Load Balancer.

```
Activities Terminal Sat 09:38 user@worker01: ~/Desktop/DevOps_Team_2/assg1/nodejs
File Edit View Search Terminal Help
nodejs-deployment LoadBalancer 10.102.94.154 192.168.79.61 3000:32555/TCP 38h
user@worker01:~/Desktop/DevOps_Team_2/assg1/nodejs$ kubectl apply -f https://raw.githubusercontent.com/google/metallb/v0.9.3/manifests/namespace.yaml
namespace/metallb-system unchanged
user@worker01:~/Desktop/DevOps_Team_2/assg1/nodejs$ kubectl create secret generic -n metallb-system memberlist --from-literal=secretkey="$(openssl rand -base64 128)"
error: failed to create secret secrets "memberlist" already exists
user@worker01:~/Desktop/DevOps_Team_2/assg1/nodejs$ kubectl apply -f https://raw.githubusercontent.com/google/metallb/v0.9.3/manifests/metallb.yaml # On the first install only
Warning: policy/v1beta1 PodSecurityPolicy is deprecated in v1.21+, unavailable in v1.25+
podsecuritypolicy.policy/controller configured
podsecuritypolicy.policy/speaker configured
serviceaccount/controller unchanged
serviceaccount/speaker unchanged
clusterrole.rbac.authorization.k8s.io/metallb-system:controller unchanged
clusterrole.rbac.authorization.k8s.io/metallb-system:speaker unchanged
role.rbac.authorization.k8s.io/config-watcher unchanged
role.rbac.authorization.k8s.io/pod-lister unchanged
clusterrolebinding.rbac.authorization.k8s.io/metallb-system:controller unchanged
clusterrolebinding.rbac.authorization.k8s.io/metallb-system:speaker unchanged
rolebinding.rbac.authorization.k8s.io/config-watcher unchanged
rolebinding.rbac.authorization.k8s.io/pod-lister unchanged
daemonset.apps/speaker unchanged
deployment.apps/controller unchanged
user@worker01:~/Desktop/DevOps_Team_2/assg1/nodejs$ minikube ip
192.168.49.2
user@worker01:~/Desktop/DevOps_Team_2/assg1/nodejs$ gedit configmap.yaml
user@worker01:~/Desktop/DevOps_Team_2/assg1/nodejs$ kubectl create -f configmap.yaml
Error from server (AlreadyExists): error when creating "configmap.yaml": configmaps "config" already exists
user@worker01:~/Desktop/DevOps_Team_2/assg1/nodejs$ kubectl delete svc nodejs-deployment
service "nodejs-deployment" deleted
user@worker01:~/Desktop/DevOps_Team_2/assg1/nodejs$ kubectl expose deployment nodejs-deployment --type="LoadBalancer"
service/nodejs-deployment exposed
user@worker01:~/Desktop/DevOps_Team_2/assg1/nodejs$ kubectl get svc
NAME                TYPE                CLUSTER-IP      EXTERNAL-IP      PORT(S)          AGE
kubernetes           ClusterIP           10.96.0.1        <none>            443/TCP           40h
nodejs-deployment    LoadBalancer       10.100.132.93    192.168.79.61    3000:30216/TCP    16s
user@worker01:~/Desktop/DevOps_Team_2/assg1/nodejs$
```

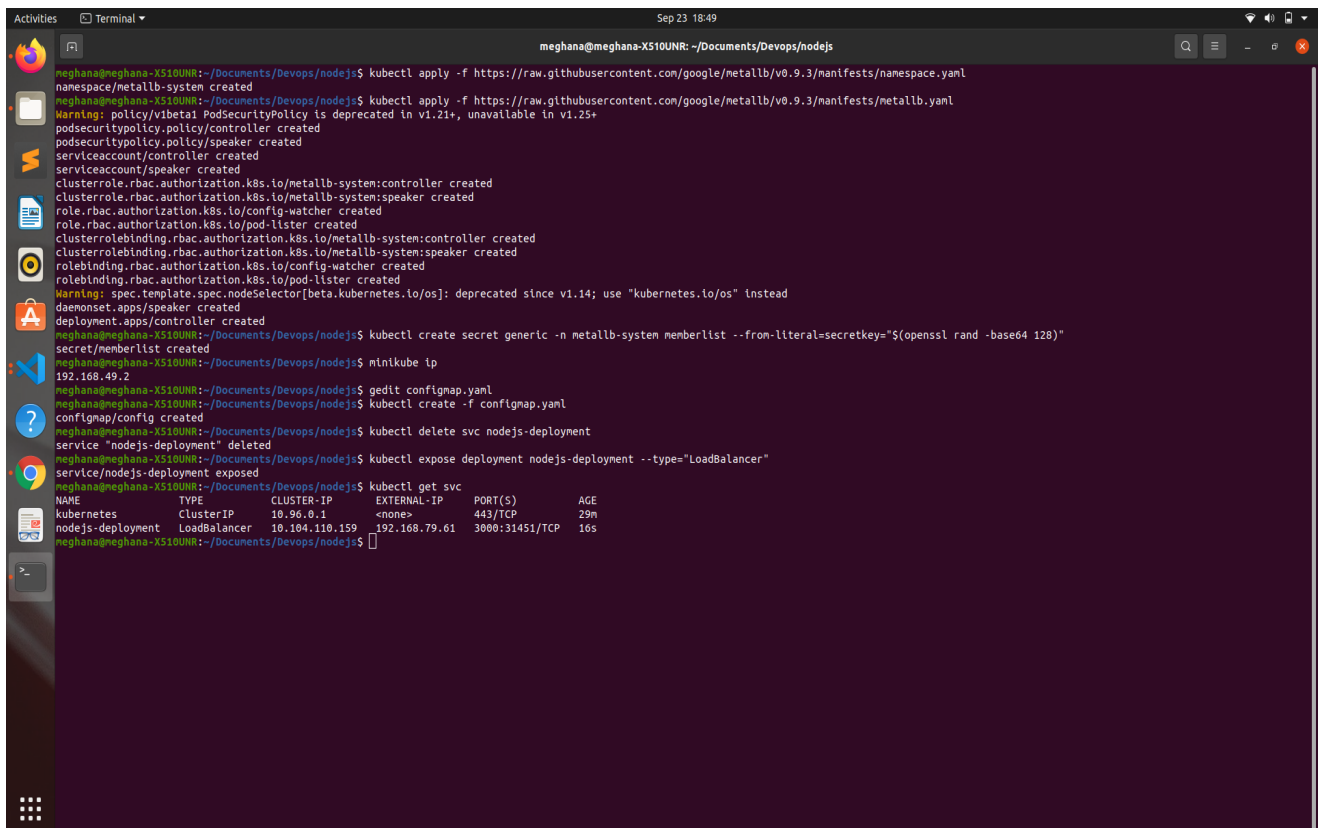
Here when we had done it for the first time we weren't getting the External-IP

```
meghana@meghana-X510UNR: ~/Documents/Devops/nodejs$ docker push meghanahadinani/nodejs-starter:latest
The push refers to repository [docker.io/meghanahadinani/nodejs-starter]
a2250152ede3: Pushed
6d1c9e4599e0: Pushed
c4ff91ddb061: Pushed
18e7e935e99b: Pushed
3ab01e8988bf: Mounted from library/node
c98dc9a94132: Mounted from library/node
3ffdb7e28503: Mounted from library/node
33d693485756: Mounted from library/node
607d71c12b77: Mounted from library/node
052174538f53: Mounted from library/node
8abfe7e7c816: Mounted from library/node
c8b886062a47: Mounted from library/node
16fc2e3ca032: Mounted from library/node
latest: digest: sha256:f915ed1bacda19113703ac8a8dc8c39b8b884d4c543ef41b024bcff0324c12d4 size: 3050
meghana@meghana-X510UNR: ~/Documents/Devops/nodejs$ docker push hello-world:latest
The push refers to repository [docker.io/library/hello-world]
f22b99068db9: Layer already exists
errors:
denied: requested access to the resource is denied
unauthorized: authentication required

meghana@meghana-X510UNR: ~/Documents/Devops/nodejs$ minikube start
minikube v1.23.2 on Ubuntu 20.04
Using the docker driver based on existing profile
Starting control plane node minikube in cluster minikube
Pulling base image ...
Creating docker container (CPUs=2, Memory=2200MB) ...
Preparing Kubernetes v1.22.2 on Docker 20.10.8 ...
  Generating certificates and keys ...
  Booting up control plane ...
  Configuring RBAC rules ...
Verifying Kubernetes components...
Executing "docker container inspect minikube --format={{.State.Status}}" took an unusually long time: 4.705679571s
Restarting the docker service may improve performance.
  Using image gcr.io/k8s-minikube/storage-provisioner:v5
Enabled addons: default-storageclass, storage-provisioner
Done! kubectll is now configured to use "minikube" cluster and "default" namespace by default
meghana@meghana-X510UNR: ~/Documents/Devops/nodejs$ gedit deploy.yaml
meghana@meghana-X510UNR: ~/Documents/Devops/nodejs$ kubectl create -f deploy.yaml
deployment.apps/nodejs-deployment created
meghana@meghana-X510UNR: ~/Documents/Devops/nodejs$ kubectl get deploy,po
NAME                                READY   UP-TO-DATE   AVAILABLE   AGE
deployment.apps/nodejs-deployment   0/2     2             0           41s

NAME                                READY   STATUS              RESTARTS   AGE
pod/nodejs-deployment-7649596568-24lvs  0/1     ContainerCreating    0           41s
pod/nodejs-deployment-7649596568-mvldw  0/1     ContainerCreating    0           41s
meghana@meghana-X510UNR: ~/Documents/Devops/nodejs$ kubectl expose deployment nodejs-deployment --type="LoadBalancer"
service/nodejs-deployment exposed
meghana@meghana-X510UNR: ~/Documents/Devops/nodejs$ kubectl get svc
NAME      TYPE        CLUSTER-IP      EXTERNAL-IP      PORT(S)          AGE
kubernetes  ClusterIP   10.96.0.1        <none>           443/TCP          13m
nodejs-deployment  LoadBalancer  10.109.63.230    <pending>        3000:32169/TCP   36s
```

Step11: As we are using minikube we'll notice that we won't get an external IP because the load balancer will not work on minikube. So we run the minikube ip and after we get it we'll create a config map for the address pool. After that we'll create a config map in the metallb-system namespace. Next, we have to delete the svc and create the service again. Now that's done, we'll be getting External IP.



```
meghana@meghana-XS10UNR: ~/Documents/Devops/nodejs
meghana@meghana-XS10UNR:~/Documents/Devops/nodejs$ kubectl apply -f https://raw.githubusercontent.com/google/metallb/v0.9.3/manifests/namespace.yaml
namespace/metallb-system created
meghana@meghana-XS10UNR:~/Documents/Devops/nodejs$ kubectl apply -f https://raw.githubusercontent.com/google/metallb/v0.9.3/manifests/metallb.yaml
Warning: policy/v1beta1 PodSecurityPolicy is deprecated in v1.21+, unavailable in v1.25+
podsecuritypolicy.policy/controller created
podsecuritypolicy.policy/speaker created
serviceaccount/controller created
serviceaccount/speaker created
clusterrole.rbac.authorization.k8s.io/metallb-system:controller created
clusterrole.rbac.authorization.k8s.io/metallb-system:speaker created
role.rbac.authorization.k8s.io/config-watcher created
role.rbac.authorization.k8s.io/pod-lister created
clusterrolebinding.rbac.authorization.k8s.io/metallb-system:controller created
clusterrolebinding.rbac.authorization.k8s.io/metallb-system:speaker created
rolebinding.rbac.authorization.k8s.io/config-watcher created
rolebinding.rbac.authorization.k8s.io/pod-lister created
Warning: spec.template.spec.nodeSelector[beta.kubernetes.io/os]: deprecated since v1.14; use "kubernetes.io/os" instead
daemonset.apps/speaker created
deployment.apps/controller created
meghana@meghana-XS10UNR:~/Documents/Devops/nodejs$ kubectl create secret generic -n metallb-system memberlist --from-literal=secretkey="$(openssl rand -base64 128)"
secret/memberlist created
meghana@meghana-XS10UNR:~/Documents/Devops/nodejs$ minikube ip
192.168.49.2
meghana@meghana-XS10UNR:~/Documents/Devops/nodejs$ gedit configmap.yaml
meghana@meghana-XS10UNR:~/Documents/Devops/nodejs$ kubectl create -f configmap.yaml
configmap/config created
meghana@meghana-XS10UNR:~/Documents/Devops/nodejs$ kubectl delete svc nodejs-deployment
service "nodejs-deployment" deleted
meghana@meghana-XS10UNR:~/Documents/Devops/nodejs$ kubectl expose deployment nodejs-deployment --type=LoadBalancer
service/nodejs-deployment exposed
meghana@meghana-XS10UNR:~/Documents/Devops/nodejs$ kubectl get svc
NAME                TYPE                CLUSTER-IP      EXTERNAL-IP      PORT(S)          AGE
kubernetes           ClusterIP           10.96.0.1        <none>            443/TCP           29m
nodejs-deployment    LoadBalancer       10.104.110.159   192.168.79.61    3000:31451/TCP    16s
```

```
Activities Terminal Sat 09:47
user@worker01: ~/Desktop/DevOps_Team_2/assg1/nodejs

File Edit View Search Terminal Help
nodejs-deployment LoadBalancer 10.102.94.154 192.168.79.61 3000:32555/TCP 38h
user@worker01:~/Desktop/DevOps_Team_2/assg1/nodejs$ kubectl apply -f https://raw.githubusercontent.com/google/metallb/v0.9.3/manifests/namespace.yaml
namespace/metallb-system unchanged
user@worker01:~/Desktop/DevOps_Team_2/assg1/nodejs$ kubectl create secret generic -n metallb-system memberlist --from-literal=secretkey="$(openssl rand -base64 128)"
error: failed to create secret secrets "memberlist" already exists
user@worker01:~/Desktop/DevOps_Team_2/assg1/nodejs$ kubectl apply -f https://raw.githubusercontent.com/google/metallb/v0.9.3/manifests/metallb.yaml # On the first install only
Warning: policy/v1beta1 PodSecurityPolicy is deprecated in v1.21+, unavailable in v1.25+
podsecuritypolicy.policy/controller configured
podsecuritypolicy.policy/speaker configured
serviceaccount/controller unchanged
serviceaccount/speaker unchanged
clusterrole.rbac.authorization.k8s.io/metallb-system:controller unchanged
clusterrole.rbac.authorization.k8s.io/metallb-system:speaker unchanged
role.rbac.authorization.k8s.io/config-watcher unchanged
role.rbac.authorization.k8s.io/pod-lister unchanged
clusterrolebinding.rbac.authorization.k8s.io/metallb-system:controller unchanged
clusterrolebinding.rbac.authorization.k8s.io/metallb-system:speaker unchanged
rolebinding.rbac.authorization.k8s.io/config-watcher unchanged
rolebinding.rbac.authorization.k8s.io/pod-lister unchanged
daemonset.apps/speaker unchanged
deployment.apps/controller unchanged
user@worker01:~/Desktop/DevOps_Team_2/assg1/nodejs$ minikube ip
192.168.49.2
user@worker01:~/Desktop/DevOps_Team_2/assg1/nodejs$ gedit configmap.yaml
user@worker01:~/Desktop/DevOps_Team_2/assg1/nodejs$ kubectl create -f configmap.yaml
Error from server (AlreadyExists): error when creating "configmap.yaml": configmaps "config" already exists
user@worker01:~/Desktop/DevOps_Team_2/assg1/nodejs$ kubectl delete svc nodejs-deployment
service "nodejs-deployment" deleted
user@worker01:~/Desktop/DevOps_Team_2/assg1/nodejs$ kubectl expose deployment nodejs-deployment --type="LoadBalancer"
service/nodejs-deployment exposed
user@worker01:~/Desktop/DevOps_Team_2/assg1/nodejs$ kubectl get svc
NAME TYPE CLUSTER-IP EXTERNAL-IP PORT(S) AGE
kubernetes ClusterIP 10.96.0.1 <none> 443/TCP 40h
nodejs-deployment LoadBalancer 10.100.132.93 192.168.79.61 3000:30216/TCP 16s
user@worker01:~/Desktop/DevOps_Team_2/assg1/nodejs$
```