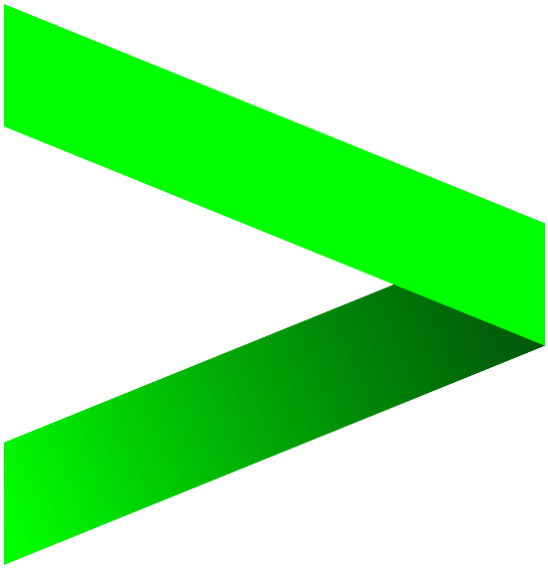
accenturetechnology
Learning and Knowledge Management

Hands-on Workbook

Kubernetes



Version	Revision date	Description	Author	Reviewed by	Approved by
1.0	1-Jun-2018	Initial version	Srinivas.c.mamidi	sreenivasa.cpcl.rao	sreenivasa.cpcl.rao

Table of Contents

Module 1 – Docker setup and basics

Exercise 1.1 Setup Docker development environment on a Linux VM

Scenario

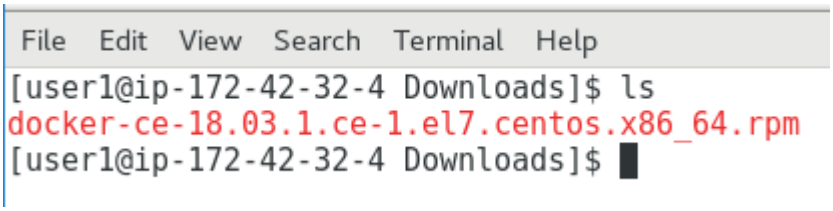
Docker setup on Windows 10 machine

To do tasks:

In this exercise

- Docker setup on Linux
- Testing Docker setup on Linux

Steps

Docker setup on windows	
1	Use below steps to install Docker on Linux (CentOS) environment Download Docker latest CentOS version from “https://download.docker.com/linux/centos/7/x86_64/stable/Packages/” 
2	Go to directory “/home/user1/Downloads” Run command “yum install docker-ce-18.03.1.ce-1.el7.centos.x86_64.rpm” as shown below to install Docker

```
[user1@ip-172-42-0-39 Downloads]$ sudo yum install docker-ce-18.03.1.ce-1.el7.centos.x86_64.rpm
Loaded plugins: fastestmirror, langpacks
Examining docker-ce-18.03.1.ce-1.el7.centos.x86_64.rpm: docker-ce-18.03.1.ce-1.el7.centos.x86_64
Marking docker-ce-18.03.1.ce-1.el7.centos.x86_64.rpm to be installed
Resolving Dependencies
--> Running transaction check
---> Package docker-ce.x86_64 0:18.03.1.ce-1.el7.centos will be installed
--> Processing Dependency: container-selinux >= 2.9 for package: docker-ce-18.03.1.ce-1.el7.centos.x86_64
Loading mirror speeds from cached hostfile
 * base: mirrors.fibergrid.in
 * epel: del-repos.extreme-ix.org
 * extras: mirrors.fibergrid.in
 * nux-dextop: mirror.li.nux.ro
 * updates: mirrors.fibergrid.in
--> Processing Dependency: pigz for package: docker-ce-18.03.1.ce-1.el7.centos.x86_64
--> Running transaction check
---> Package container-selinux.noarch 2:2.55-1.el7 will be installed
--> Processing Dependency: selinux-policy-targeted >= 3.13.1-183 for package: 2:container-selinux-2.55-1.el7.noarch
--> Processing Dependency: selinux-policy-base >= 3.13.1-183 for package: 2:container-selinux-2.55-1.el7.noarch
--> Processing Dependency: selinux-policy >= 3.13.1-183 for package: 2:container-selinux-2.55-1.el7.noarch
---> Package pigz.x86_64 0:2.3.4-1.el7 will be installed
--> Running transaction check
---> Package selinux-policy.noarch 0:3.13.1-166.el7_4.7 will be updated
---> Package selinux-policy.noarch 0:3.13.1-192.el7_5.4 will be an update
--> Processing Dependency: policycoreutils >= 2.5-18 for package: selinux-policy-3.13.1-192.el7_5.4.noarch
---> Package selinux-policy-targeted.noarch 0:3.13.1-166.el7_4.7 will be updated
---> Package selinux-policy-targeted.noarch 0:3.13.1-192.el7_5.4 will be an update
--> Running transaction check
---> Package policycoreutils.x86_64 0:2.5-17.1.el7 will be updated
--> Processing Dependency: policycoreutils = 2.5-17.1.el7 for package: policycoreutils-python-2.5-17.1.el7.x86_64
---> Package policycoreutils.x86_64 0:2.5-22.el7 will be an update
  Verifying  : policycoreutils-python-2.5-22.el7.x86_64                                                  9/25
  Verifying  : 2:container-selinux-2.55-1.el7.noarch                                                    10/25
  Verifying  : libsepol-2.5-8.1.el7.x86_64                                                              11/25
  Verifying  : docker-ce-18.03.1.ce-1.el7.centos.x86_64                                                 12/25
  Verifying  : libselinux-2.5-12.el7.x86_64                                                             13/25
  Verifying  : libselinux-utils-2.5-12.el7.x86_64                                                       14/25
  Verifying  : libselinux-utils-2.5-11.el7.x86_64                                                       15/25
  Verifying  : libselinux-2.5-11.el7.x86_64                                                             16/25
  Verifying  : libsepol-2.5-6.el7.x86_64                                                                17/25
  Verifying  : policycoreutils-python-2.5-17.1.el7.x86_64                                               18/25
  Verifying  : policycoreutils-2.5-17.1.el7.x86_64                                                      19/25
  Verifying  : libsemanage-python-2.5-8.el7.x86_64                                                      20/25
  Verifying  : selinux-policy-3.13.1-166.el7_4.7.noarch                                                 21/25
  Verifying  : selinux-policy-targeted-3.13.1-166.el7_4.7.noarch                                        22/25
  Verifying  : libsemanage-2.5-8.el7.x86_64                                                             23/25
  Verifying  : libselinux-python-2.5-11.el7.x86_64                                                      24/25
  Verifying  : setools-libs-3.3.8-1.1.el7.x86_64                                                        25/25

Installed:
  docker-ce.x86_64 0:18.03.1.ce-1.el7.centos

Dependency Installed:
  container-selinux.noarch 2:2.55-1.el7                         pigz.x86_64 0:2.3.4-1.el7

Dependency Updated:
  libselinux.x86_64 0:2.5-12.el7                libselinux-python.x86_64 0:2.5-12.el7        libselinux-utils.x86_64 0:2.5-12.el7
  libsemanage.x86_64 0:2.5-11.el7               libsemanage-python.x86_64 0:2.5-11.el7       libsepol.x86_64 0:2.5-8.1.el7
  policycoreutils.x86_64 0:2.5-22.el7           policycoreutils-python.x86_64 0:2.5-22.el7   selinux-policy.noarch 0:3.13.1-192.el7_5.4
  selinux-policy-targeted.noarch 0:3.13.1-192.el7_5.4   setools-libs.x86_64 0:3.3.8-2.el7

Complete!
[user1@ip-172-42-0-39 Downloads]$
```

3	Run command “systemctl start docker” to start Docker
4	Run command “docker run hello-world” to see everything working

```
[user1@ip-172-42-0-39 ~]$ sudo docker run hello-world
Unable to find image 'hello-world:latest' locally
latest: Pulling from library/hello-world
9db2ca6ccae0: Pull complete
Digest: sha256:4b8ff392a12ed9ea17784bd3c9a8b1fa3299cac44aca35a85c90c5e3c7afacdc
Status: Downloaded newer image for hello-world:latest

Hello from Docker!
This message shows that your installation appears to be working correctly.

To generate this message, Docker took the following steps:
 1. The Docker client contacted the Docker daemon.
 2. The Docker daemon pulled the "hello-world" image from the Docker Hub.
    (amd64)
 3. The Docker daemon created a new container from that image which runs the
    executable that produces the output you are currently reading.
 4. The Docker daemon streamed that output to the Docker client, which sent it
    to your terminal.

To try something more ambitious, you can run an Ubuntu container with:
 $ docker run -it ubuntu bash

Share images, automate workflows, and more with a free Docker ID:
 https://hub.docker.com/

For more examples and ideas, visit:
 https://docs.docker.com/engine/userguide/

[user1@ip-172-42-0-39 ~]$
```

Run command "docker images" as shown below

```
[root@ip-172-42-32-4 /]# docker images
REPOSITORY          TAG                 IMAGE ID            CREATED             SIZE
hello-world         latest              fce289e99eb9        4 weeks ago         1.84kB
[root@ip-172-42-32-4 /]#
```

Exercise 1.2 Building an app in the Docker way

Scenario

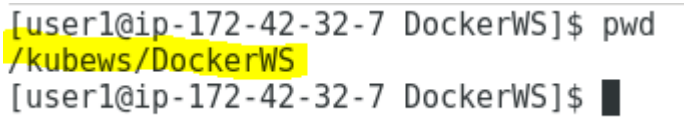
Building a web app in the Docker way

To do tasks:

In this exercise, you will be:

- Writing dockerfile to containerize app, image creation
- Running containers, tagging and pushing to central repository's
- Pulling docker images from central repository

Steps

Containerizing a Python app	
1	Create new workspace as shown below 
2	Understand Docker file as shown below

```
[root@ip-172-42-32-4 DockerWS]# ls
app.py  Dockerfile  requirements.txt
[root@ip-172-42-32-4 DockerWS]# cat Dockerfile
# Use an official Python runtime as a parent image

FROM python:2.7-slim

# Set the working directory to /app

WORKDIR /app

# Copy the current directory contents into the container at /app

ADD . /app

# Install any needed packages specified in requirements.txt

RUN pip install --trusted-host pypi.python.org -r requirements.txt

# Make port 80 available to the world outside this container

EXPOSE 80

# Define environment variable Acc LKM

ENV NAME World

# Run app.py when the container launches

CMD ["python", "app.py"]

# https://docs.docker.com/engine/reference/builder/#dockerignore-file
[root@ip-172-42-32-4 DockerWS]#
```

3	Understand app.py file with below mentioned content

```
[root@ip-172-42-32-4 DockerWS]# cat app.py
from flask import Flask
from redis import Redis, RedisError
import os
import socket

# Connect to Redis
redis = Redis(host="redis", db=0, socket_connect_timeout=2, socket_timeout=2)

app = Flask(__name__)

@app.route("/")
def hello():
    try:
        visits = redis.incr("counter")
    except RedisError:
        visits = "<i>cannot connect to Redis, counter disabled</i>"

    html = "<h3>Hello LKM Accenture {name}!</h3>" \
           "<b>Hostname:</b> {hostname}<br/>" \
           "<b>Visits:</b> {visits}"
    return html.format(name=os.getenv("NAME", "world-Accenture LKM 28th Sep18"), hostname=socket.gethostname(), visits=visits)

if __name__ == "__main__":
    app.run(host='0.0.0.0', port=80)

[root@ip-172-42-32-4 DockerWS]#
```

4

Understand requirements.txt with below mentioned content

```
[root@ip-172-42-32-4 DockerWS]# cat requirements.txt
Flask

Redis
[root@ip-172-42-32-4 DockerWS]#
```

5

Top level of your new directory must contain files as shown below

```
[user1@ip-172-42-32-7 DockerWS]$ ls
app.py  Dockerfile  requirements.txt
[user1@ip-172-42-32-7 DockerWS]$
```

6

Run the build command “docker build -t firsthello .” from “/kubews/DockerWS” to creates a Docker image as shown below

```
Step 1/7 : FROM python:2.7-slim
2.7-slim: Pulling from library/python
5e6ec7f28fb7: Pull complete
76e195ed530a: Pull complete
674dece4375b: Pull complete
c306b9f6d5e3: Pull complete
Digest: sha256:9be0e36b98223ee70e92a610991e3684416a70ae6b5afe22f818b675ee22e014
Status: Downloaded newer image for python:2.7-slim
 ---> 4620f1f365b9
Step 2/7 : WORKDIR /app
Removing intermediate container 0cb18892f88b
 ---> 8061a35caa9e
Step 3/7 : ADD . /app
 ---> e4ac4fbdaff3
Step 4/7 : RUN pip install --trusted-host pypi.python.org -r requirements.txt
 ---> Running in fca4c8374d20
DEPRECATION: Python 2.7 will reach the end of its life on January 1st, 2020. Please upgrade your Python as Python 2.7 won't be maintained after that date. A future version
.
Collecting Flask (from -r requirements.txt (line 1))
  Downloading https://files.pythonhosted.org/packages/7f/e7/08578774ed4536d3242b14dacb4696386634607af824ea997202cd0edb4b/Flask-1.0.2-py2.py3-none-any.whl (91kB)
Collecting Redis (from -r requirements.txt (line 3))
  Downloading https://files.pythonhosted.org/packages/f1/19/a0282b77c23f9f9dbcc6480787a60807c78a45947593a02dbf026636c90d/redis-3.1.0-py2.py3-none-any.whl (63kB)
Collecting itsdangerous>=0.24 (from Flask->-r requirements.txt (line 1))
  Downloading https://files.pythonhosted.org/packages/76/ae/44b03b253d6fade317f32c24d100b3b35c2239807046a4c953c7b89fa49e/itsdangerous-1.1.0-py2.py3-none-any.whl
Collecting Jinja2>=2.10 (from Flask->-r requirements.txt (line 1))
  Downloading https://files.pythonhosted.org/packages/7f/ff/ae64bacdfc95f27a016a7bed8e8686763ba4d277a78ca76f32659220a731/Jinja2-2.10-py2.py3-none-any.whl (126kB)
Collecting Werkzeug>=0.14 (from Flask->-r requirements.txt (line 1))
  Downloading https://files.pythonhosted.org/packages/20/c4/12e3e56473e52375aa29c4764e70d1b8f3efa6682bef8d0aae04fe335243/Werkzeug-0.14.1-py2.py3-none-any.whl (322kB)
Collecting click>=5.1 (from Flask->-r requirements.txt (line 1))
  Downloading https://files.pythonhosted.org/packages/fa/37/45185cb5abbc30d7257104c434fe0b07e5a195a6847506c074527aa599ec/Click-7.0-py2.py3-none-any.whl (81kB)
Collecting MarkupSafe>=0.23 (from Jinja2>=2.10->Flask->-r requirements.txt (line 1))
  Downloading https://files.pythonhosted.org/packages/bc/3a/6bfd7b4b202fa33bddda8e4e3d3acc719f381fd730f9a0e7c5f34e845bd4d/MarkupSafe-1.1.0-cp27-cp27mu-manylinux1_x86_64.whl
Installing collected packages: itsdangerous, MarkupSafe, Jinja2, Werkzeug, click, Flask, Redis
Successfully installed Flask-1.0.2 Jinja2-2.10 MarkupSafe-1.1.0 Redis-3.1.0 Werkzeug-0.14.1 click-7.0 itsdangerous-1.1.0
Removing intermediate container fca4c8374d20
 ---> 32c8b73a989c
Step 5/7 : EXPOSE 80
 ---> Running in cf343cfd6b1a
Removing intermediate container cf343cfd6b1a
 ---> 26567f8db116
Step 6/7 : ENV NAME World
 ---> Running in 029e282004a7
Removing intermediate container 029e282004a7
 ---> 1509300165a9
Step 7/7 : CMD ["python", "app.py"]
 ---> Running in 41528f284346
Removing intermediate container 41528f284346
 ---> 0b25d2c5ab0d
Successfully built 0b25d2c5ab0d
Successfully tagged firsthello:latest
[root@ip-172-42-32-4 DockerWS]#
```

7 Check local Docker image registry using "docker image ls" to see newly created app image.

```
[root@ip-172-42-32-4 DockerWS]# docker image ls
REPOSITORY          TAG                 IMAGE ID            CREATED             SIZE
firsthello          latest              0b25d2c5ab0d        2 minutes ago       131MB
python              2.7-slim            4620f1f365b9        6 days ago          120MB
hello-world         latest              fce289e99eb9        4 weeks ago         1.84kB
[root@ip-172-42-32-4 DockerWS]#
```

8 Run the app, mapping your machine's port 4000 (or any free port) to the container's published port 80 using -p. command is "docker run -p 4000:80 firsthello"

```
[user1@ip-172-42-32-7 DockerWS]$ sudo docker run -p 4000:80 firsthello
 * Serving Flask app "app" (lazy loading)
 * Environment: production
   WARNING: Do not use the development server in a production environment.
   Use a production WSGI server instead.
 * Debug mode: off
 * Running on http://0.0.0.0:80/ (Press CTRL+C to quit)
172.17.0.1 - - [31/Jul/2018 07:02:23] "GET / HTTP/1.1" 200 -
172.17.0.1 - - [31/Jul/2018 07:02:23] "GET /favicon.ico HTTP/1.1" 404 -
172.17.0.1 - - [31/Jul/2018 07:04:08] "GET / HTTP/1.1" 200 -
```

Go to web client, type URL http://localhost:4000 to see app UI

You can run app in the background, in detached mode as shown below.

```
[user1@ip-172-42-32-7 DockerWS]$ sudo docker run -d -p 4000:80 firsthello
c62e5a65f534f299cd7a4bd0fb3e7bc7de1317aaad1be6cef451a68e6fa9264b
[user1@ip-172-42-32-7 DockerWS]$
```

You can run command “docker container stop <container id>” to stop container as shown below

```
[user1@ip-172-42-1-80 ~]$ sudo docker ps
CONTAINER ID        IMAGE               COMMAND             CREATED             STATUS              PORTS                  NAMES
e035b0956e6e        firsthello          "python app.py"     8 minutes ago       Up 8 minutes        0.0.0.0:4000->80/tcp   agitated_kapitsa
[user1@ip-172-42-1-80 ~]$ sudo docker stop e035b0956e6e
e035b0956e6e
[user1@ip-172-42-1-80 ~]$ sudo docker ps
CONTAINER ID        IMAGE               COMMAND             CREATED             STATUS              PORTS               NAMES
[user1@ip-172-42-1-80 ~]$
```

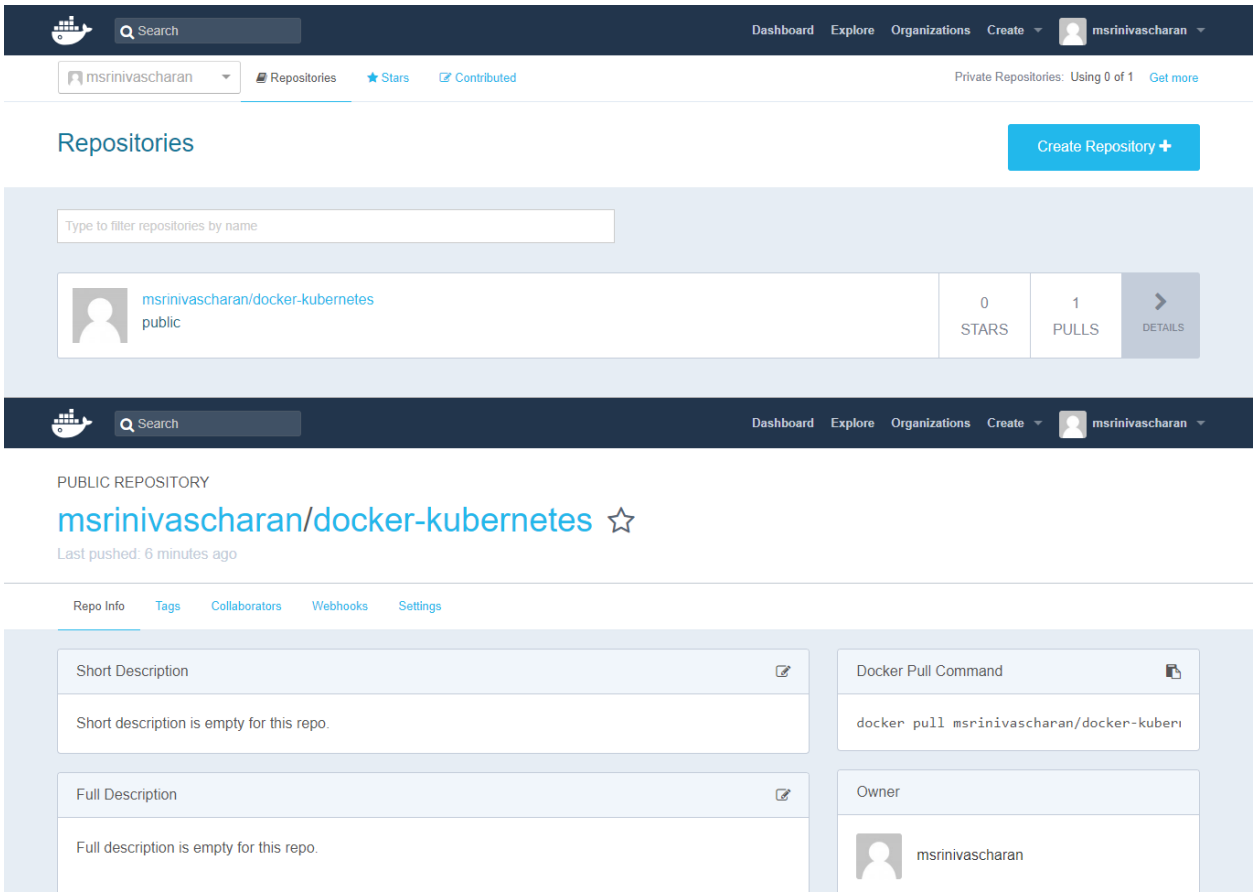
9	Note: Before this step ensure that you have access to https://hub.docker.com/
	
	Log in to the Docker public registry on your local machine as shown below
	
	`[root@ip-172-42-32-4 DockerWS]# docker login`
	`Login with your Docker ID to push and pull images from Docker Hub. If you don't have a Docker ID, head over to https://hub.docker.com to create one.`
	`Username: msrinivascharan`
	`Password:`
	`Login Succeeded`
	`[root@ip-172-42-32-4 DockerWS]#`
	
	Tag the “firsthello” image as shown below, command is “docker tag firsthello msrinivascharan/docker-kubernetes:module1”
	
	`[root@ip-172-42-32-4 DockerWS]# docker tag firsthello msrinivascharan/docker-kubernetes:module1`
	`[root@ip-172-42-32-4 DockerWS]# docker images`
	`REPOSITORY TAG IMAGE ID CREATED SIZE`
	`msrinivascharan/docker-kubernetes module1 0b25d2c5ab0d 13 minutes ago 131MB`
	`firsthello latest 0b25d2c5ab0d 13 minutes ago 131MB`
	`python 2.7-slim 4620f1f365b9 6 days ago 120MB`
	`hello-world latest fce289e99eb9 4 weeks ago 1.84kB`
	`[root@ip-172-42-32-4 DockerWS]#`

10	

You can Upload your tagged image to the repository as shown below, command is "docker push msrinivascharan/docker-kubernetes:module1"

```
[root@ip-172-42-32-4 DockerWS]# docker push msrinivascharan/docker-kubernetes:module1
The push refers to repository [docker.io/msrinivascharan/docker-kubernetes]
36844df495e1: Pushed
4bb4bb2fdea0: Pushed
31b9a82d7f36: Pushed
4717ad73473a: Mounted from library/python
126818daa940: Mounted from library/python
c06903a8f01f: Mounted from library/python
3c816b4ead84: Mounted from library/python
module1: digest: sha256:ba5fe78b8cb806f3138b00d17a21ca00baa0611d429b426911feff2d2b633679 size: 1787
[root@ip-172-42-32-4 DockerWS]#
```

Go to https://hub.docker.com/, you can see recently pushed image as shown below



11

Once image uploaded to docker hub or any repository, you can pull from any ware and run as shown below. Command is "docker run -p 4000:80 msrinivascharan/docker-kubernetes:module1 - on diff machine"

Note: If the image isn't available locally on the machine, Docker pulls it from the registry.

------- End of Exercise -------

Module 2 – Setting up single node cluster with Kubernetes Minikube

Exercise 2.1 Setting up single node Kubernetes cluster using Minikube

Scenario

Kubectl and Minikube installation

To do tasks:

In this exercise, you will be:

- Kubectl installation
- Minikube installation

Steps

Kubectl & Docker, Minikube setup	
1	Go to directory "/etc/yum.repos.d" and create new file "kubernetes.repo"

2	Add below mentioned content to newly created "kubernetes.repo" file, as shown below
	
	`[user1@ip-172-42-0-39 yum.repos.d]$ sudo cat kubernetes.repo`
	`[kubernetes]`
	`name=Kubernetes`
	`baseurl=https://packages.cloud.google.com/yum/repos/kubernetes-el7-x86_64`
	`enabled=1`
	`gpgcheck=1`
	`repo_gpgcheck=1`
	`gpgkey=https://packages.cloud.google.com/yum/doc/yum-key.gpg https://packages.cloud.google.com/yum/doc/rpm-package-key.gpg`
	
	`[user1@ip-172-42-0-39 yum.repos.d]$`
	
	You can copy above mentioned configurations from here or as shown below
	
	`[kubernetes]`
	`name=Kubernetes`
	`baseurl=https://packages.cloud.google.com/yum/repos/kubernetes-el7-x86_64`
	`enabled=1`
	`gpgcheck=1`
	`repo_gpgcheck=1`
	`gpgkey=https://packages.cloud.google.com/yum/doc/yum-key.gpg`
	`https://packages.cloud.google.com/yum/doc/rpm-package-key.gpg`

3	Run command as shown below to install Kubectl command is "yum install -y kubectl"

```
[user1@ip-172-42-0-39 /]$ sudo yum install -y kubectl
Loaded plugins: fastestmirror, langpacks
base                                                                                                                                                  | 3.6 kB  00:00:00
epel/x86_64/metalink                                                                                                                                  | 8.0 kB  00:00:00
epel                                                                                                                                                  | 4.7 kB  00:00:00
http://del-repos.extreme-ix.org/epel/7/x86_64/repodata/repomd.xml: [Errno -1] repomd.xml does not match metalink for epel
Trying other mirror.
epel                                                                                                                                                  | 3.2 kB  00:00:00
extras                                                                                                                                                | 3.4 kB  00:00:00
kubernetes/signature                                                                                                                                  |  454 B  00:00:00
Retrieving key from https://packages.cloud.google.com/yum/doc/yum-key.gpg
Importing GPG key 0xA7317B0F:
 Userid     : "Google Cloud Packages Automatic Signing Key <gc-team@google.com>"
 Fingerprint: d0bc 747f d8ca f711 7500 d6fa 3746 c208 a731 7b0f
 From       : https://packages.cloud.google.com/yum/doc/yum-key.gpg
Retrieving key from https://packages.cloud.google.com/yum/doc/rpm-package-key.gpg
kubernetes/signature                                                                                                                                  | 1.4 kB  00:00:00 !!!
nux-dextop                                                                                                                                            | 2.9 kB  00:00:00
updates                                                                                                                                               | 3.4 kB  00:00:00
xrdp                                                                                                                                                  | 2.9 kB  00:00:00
(1/10): base/7/x86_64/group_gz                                                                                                                        | 166 kB  00:00:00
(2/10): extras/7/x86_64/primary_db                                                                                                                    | 150 kB  00:00:00
(3/10): kubernetes/primary                                                                                                                            |  32 kB  00:00:01
(4/10): epel/x86_64/group_gz                                                                                                                          |  88 kB  00:00:01
(5/10): updates/7/x86_64/primary_db                                                                                                                   | 3.6 MB  00:00:00
(6/10): nux-dextop/x86_64/primary_db                                                                                                                  | 1.8 MB  00:00:01
(7/10): base/7/x86_64/primary_db                                                                                                                      | 5.9 MB  00:00:03
(8/10): epel/x86_64/primary                                                                                                                           | 3.5 MB  00:00:04
(9/10): epel/x86_64/updateinfo                                                                                                                        | 929 kB  00:00:04
(10/10): xrdp/primary_db                                                                                                                              | 1.8 MB  00:00:03
epel                                                                                                                                                             12609/12609
kubernetes                                                                                                                                                         231/231
Resolving Dependencies
--> Running transaction check
---> Package kubectl.x86_64 0:1.11.0-0 will be installed
--> Finished Dependency Resolution

Dependencies Resolved

================================================================================================================================================================
 Package                          Arch                             Version                          Repository                                          Size
================================================================================================================================================================
Installing:
 kubectl                          x86_64                           1.11.0-0                         kubernetes                                         7.5 M

Transaction Summary
================================================================================================================================================================
Install  1 Package

Total download size: 7.5 M
Installed size: 37 M
Downloading packages:
warning: /var/cache/yum/x86_64/7/kubernetes/packages/5736d31b18c9a00419105394d462ecec4847e6a9bbc7b5a4e41790a67e29c817-kubectl-1.11.0-0.x86_64.rpm: He
der V4 RSA/SHA512 Signature, key ID 3e1ba8d5: NOKEY
Public key for 5736d31b18c9a00419105394d462ecec4847e6a9bbc7b5a4e41790a67e29c817-kubectl-1.11.0-0.x86_64.rpm is not installed
5736d31b18c9a00419105394d462ecec4847e6a9bbc7b5a4e41790a67e29c817-kubectl-1.11.0-0.x86_64.rpm                                  | 7.5 MB  00:00:01
Retrieving key from https://packages.cloud.google.com/yum/doc/yum-key.gpg
Importing GPG key 0xA7317B0F:
 Userid     : "Google Cloud Packages Automatic Signing Key <gc-team@google.com>"
 Fingerprint: d0bc 747f d8ca f711 7500 d6fa 3746 c208 a731 7b0f
 From       : https://packages.cloud.google.com/yum/doc/yum-key.gpg
Retrieving key from https://packages.cloud.google.com/yum/doc/rpm-package-key.gpg
Importing GPG key 0x3E1BA8D5:
 Userid     : "Google Cloud Packages RPM Signing Key <gc-team@google.com>"
 Fingerprint: 3749 e1ba 95a8 6ce0 5454 6ed2 f09c 394c 3e1b a8d5
 From       : https://packages.cloud.google.com/yum/doc/rpm-package-key.gpg
Running transaction check
Running transaction test
Transaction test succeeded
Running transaction
  Installing : kubectl-1.11.0-0.x86_64                                                                                                                                1/1
  Verifying  : kubectl-1.11.0-0.x86_64                                                                                                                                1/1

Installed:
  kubectl.x86_64 0:1.11.0-0

Complete!
```

Run command “kubectl” to check the installation

4	Note: Docker installation mandatory before proceeding with Minikube installation Use below steps to install Minikube on Linux (CentOS) environment Step 1: cd to downloads -> curl -Lo minikube https://storage.googleapis.com/minikube/releases/latest/minikube-linux-amd64

```
[root@ip-172-42-32-4 Downloads]# ls
minikube
[root@ip-172-42-32-4 Downloads]#
```

Step 2: sudo mv minikube /usr/bin/

Step 3: sudo chmod +x minikube

```
[root@ip-172-42-32-4 Downloads]#  sudo chmod +x minikube
[root@ip-172-42-32-4 Downloads]# ls
minikube
[root@ip-172-42-32-4 Downloads]# sudo mv minikube /usr/bin/
[root@ip-172-42-32-4 Downloads]# ls
[root@ip-172-42-32-4 Downloads]#
```

5

Use below steps to start Minikube on Linux (CentOS) environment, command is "minikube start --vm-driver=none"

```
[root@ip-172-42-32-4 /]# minikube start --vm-driver=none
Starting local Kubernetes v1.13.2 cluster...
Starting VM...
Getting VM IP address...
Moving files into cluster...
Downloading kubeadm v1.13.2
Downloading kubelet v1.13.2
Finished Downloading kubeadm v1.13.2
Finished Downloading kubelet v1.13.2
Setting up certs...
Connecting to cluster...
Setting up kubeconfig...
Stopping extra container runtimes...
Starting cluster components...
Verifying kubelet health ...
Verifying apiserver health ...
Kubectl is now configured to use the cluster.
==================
WARNING: IT IS RECOMMENDED NOT TO RUN THE NONE DRIVER ON PERSONAL WORKSTATIONS
        The 'none' driver will run an insecure kubernetes apiserver as root that may leave the host vulnerable to CSRF attacks

When using the none driver, the kubectl config and credentials generated will be root owned and will appear in the root home directory.
You will need to move the files to the appropriate location and then set the correct permissions.  An example of this is below:

        sudo mv /root/.kube $HOME/.kube # this will write over any previous configuration
        sudo chown -R $USER $HOME/.kube
        sudo chgrp -R $USER $HOME/.kube

        sudo mv /root/.minikube $HOME/.minikube # this will write over any previous configuration
        sudo chown -R $USER $HOME/.minikube
        sudo chgrp -R $USER $HOME/.minikube

This can also be done automatically by setting the env var CHANGE_MINIKUBE_NONE_USER=true
Loading cached images from config file.


Everything looks great. Please enjoy minikube!
[root@ip-172-42-32-4 /]#
```

6

Run command "" as shown below to Check that kubectl is properly configured by getting the cluster state

```
[user1@ip-172-42-0-39 /]$ kubectl cluster-info
Kubernetes master is running at https://172.42.0.39:8443
KubeDNS is running at https://172.42.0.39:8443/api/v1/namespaces/kube-system/services/kube-dns:dns/proxy

To further debug and diagnose cluster problems, use 'kubectl cluster-info dump'.
[user1@ip-172-42-0-39 /]$
```

7

	Run command "kubectl get nodes" as shown below to see available node details
	
	`[root@ip-172-42-32-4 /]# kubectl get nodes`
	`NAME STATUS ROLES AGE VERSION`
	`minikube Ready master 113m v1.13.2`
	`[root@ip-172-42-32-4 /]#`

------- End of Exercise -------

Module 3 – Kubernetes Cluster basic

Exercise 3.1 Deploying an App to Kubernetes cluster

Scenario

Deploy first app on to Kubernetes cluster

To do tasks:

In this exercise, you will be:

- Deploy first app on Kubernetes using Kubectl
- Basics of Kubectl and its CLI, and App interactions

Steps

Deploying app	

1	Check nodes availability as shown below using command "sudo kubectl get nodes"
	
	`[root@ip-172-42-32-4 /]# kubectl get nodes`
	`NAME STATUS ROLES AGE VERSION`
	`minikube Ready master 113m v1.13.2`
	`[root@ip-172-42-32-4 /]#`

2	Run command "kubectl run kubernetes-bootcamp --image=gcr.io/google-samples/kubernetes-bootcamp:v1 --port=8080" as shown below to deploy the app
	
	`[user1@ip-172-42-32-7 /]$ sudo kubectl run kubernetes-bootcamp --image=gcr.io/google-samples/kubernetes-bootcamp:v1 --port=8080`
	`deployment.apps/kubernetes-bootcamp created`
	`[user1@ip-172-42-32-7 /]$`
	
	The run command creates a new deployment. We need to provide the deployment name and app image location (include the full repository URL for images hosted outside Docker hub). We want to run the app on a specific port, so we add the --port parameter

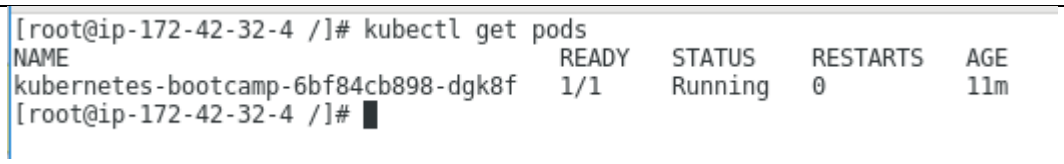
	Above command done 3 things • searched for a suitable node where an instance of the application could be run (we have only 1 available node) • scheduled the application to run on that Node • configured the cluster to reschedule the instance on a new Node when needed

3	Run command "kubectl get deployments" as shown below to get deployed app details
	
	`[root@ip-172-42-32-4 /]# kubectl get deployments`
	`NAME  READY  UP-TO-DATE  AVAILABLE  AGE`
	`kubernetes-bootcamp  1/1  1  1  98s`
	`[root@ip-172-42-32-4 /]#`
	
	We see that there is 1 deployment running a single instance of your app. The instance is running inside a Docker container on your node

4	Run command "kubectl proxy" as shown below in terminal 2 to run proxy
	
	`[user1@ip-172-42-32-7 ~]$ sudo kubectl proxy`
	`[sudo] password for user1:`
	`Starting to serve on 127.0.0.1:8001`
	
	Note: The kubectl command can create a proxy that will forward communications into the cluster-wide, private network. The proxy can be terminated by pressing control-C and won't show any output while its running.

5	Run command "curl http://localhost:8001/version" as shown below in terminal to check proxy
	
	`[root@ip-172-42-32-4 /]# curl http://localhost:8001/version`
	`{`
	`  "major": "1",`
	`  "minor": "13",`
	`  "gitVersion": "v1.13.2",`
	`  "gitCommit": "cff46ab41ff0bb44d8584413b598ad8360ec1def",`
	`  "gitTreeState": "clean",`
	`  "buildDate": "2019-01-10T23:28:14Z",`
	`  "goVersion": "go1.11.4",`
	`  "compiler": "gc",`
	`  "platform": "linux/amd64"`
	`}[root@ip-172-42-32-4 /]#`

6	A Pod represents a set of running containers on your cluster Run command "kubectl get pods"

	
7	Run command "curl http://localhost:8001/api/v1/namespaces/default/pods/< kubernetes-bootcamp-6bf84cb898-dgk8f > /proxy/" to access app UI 
------- End of Exercise -------	

Exercise 3.2 Exploring App Deployed in Kubernetes cluster

Scenario

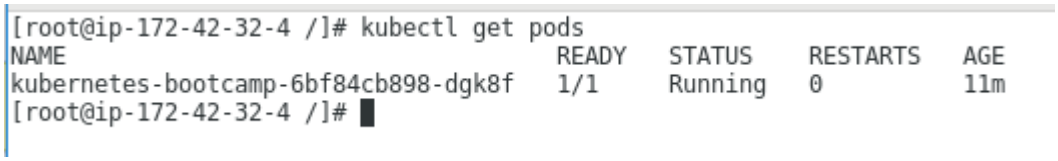
Exploring deployed app using kubctl

To do tasks:

In this exercise, you will be:

- Troubleshoot Kubernetes application using kubectl
- Using kubectl get, describe, log, exec commands

Steps

Exploring App	
1	Run command "sudo kubectl get pod" as shown below to get deployed app details 
2	Run command " kubectl describe pods" as shown below to get more information about pods

```
[root@ip-172-42-32-4 /]# kubectl describe pods
Name:               kubernetes-bootcamp-6bf84cb898-dgk8f
Namespace:          default
Priority:           0
PriorityClassName:  <none>
Node:               minikube/172.42.32.4
Start Time:         Fri, 01 Feb 2019 11:18:05 +0000
Labels:             pod-template-hash=6bf84cb898
                    run=kubernetes-bootcamp
Annotations:        <none>
Status:             Running
IP:                 172.17.0.4
Controlled By:      ReplicaSet/kubernetes-bootcamp-6bf84cb898
Containers:
  kubernetes-bootcamp:
    Container ID:   docker://f04126d00b98164299dcefc0c54a5adc6d025d5f0ffc96dfef3c1c2f9cb51261
    Image:          gcr.io/google-samples/kubernetes-bootcamp:v1
    Image ID:       docker-pullable://gcr.io/google-samples/kubernetes-bootcamp@sha256:0d6b8ee63bb57c5f5b6156f446b3bc3b3c143d233037f3a2f00e279c8fcc64af
    Port:           8080/TCP
    Host Port:      0/TCP
    State:          Running
      Started:      Fri, 01 Feb 2019 11:18:16 +0000
    Ready:          True
    Restart Count:  0
    Environment:    <none>
    Mounts:
      /var/run/secrets/kubernetes.io/serviceaccount from default-token-rmfdk (ro)
Conditions:
  Type              Status
  Initialized       True
  Ready             True
  ContainersReady   True
  PodScheduled      True
Volumes:
  default-token-rmfdk:
    Type:        Secret (a volume populated by a Secret)
    SecretName:  default-token-rmfdk
    Optional:    false
QoS Class:       BestEffort
Node-Selectors:  <none>
Tolerations:     node.kubernetes.io/not-ready:NoExecute for 300s
                 node.kubernetes.io/unreachable:NoExecute for 300s
Events:
  Type    Reason     Age   From               Message
  ----    ------     ----  ----               -------
  Normal  Scheduled  23m   default-scheduler  Successfully assigned default/kubernetes-bootcamp-6bf84cb898-dgk8f to minikube
  Normal  Pulling    23m   kubelet, minikube  pulling image "gcr.io/google-samples/kubernetes-bootcamp:v1"
  Normal  Pulled     22m   kubelet, minikube  Successfully pulled image "gcr.io/google-samples/kubernetes-bootcamp:v1"
  Normal  Created    22m   kubelet, minikube  Created container
  Normal  Started    22m   kubelet, minikube  Started container
[root@ip-172-42-32-4 /]#
```

Note: the describe command can be used to get detailed information about most of the kubernetes primitives: node, pods, deployments. The describe output is designed to be human readable, not to be scripted against.

3

Run command as shown below in terminal 2 to run proxy

```
[user1@ip-172-42-32-7 ~]$ sudo kubectl proxy
[sudo] password for user1:
Starting to serve on 127.0.0.1:8001
```

Note: The kubectl command can create a proxy that will forward communications into the cluster-wide, private network. The proxy can be terminated by pressing control-C and won't show any output while its running.

4

Run command “kubectl logs < kubernetes-bootcamp-6bf84cb898-dgk8f >” as shown below to see logs

```
[root@ip-172-42-32-4 /]# kubectl logs kubernetes-bootcamp-6bf84cb898-dgk8f
Kubernetes Bootcamp App Started At: 2019-02-01T11:18:16.704Z | Running On:  kubernetes-bootcamp-6bf84cb898-dgk8f

Running On: kubernetes-bootcamp-6bf84cb898-dgk8f | Total Requests: 1 | App Uptime: 1016.742 seconds | Log Time: 2019-02-01T11:35:13.446Z
Running On: kubernetes-bootcamp-6bf84cb898-dgk8f | Total Requests: 2 | App Uptime: 1257.826 seconds | Log Time: 2019-02-01T11:39:14.530Z
[root@ip-172-42-32-4 /]#
```

6

	Run command "kubectl exec < kubernetes-bootcamp-6bf84cb898-dgk8f > env" as shown below to execute commands directly on the container to list the environment variables
	
	`[root@ip-172-42-32-4 /]# kubectl exec kubernetes-bootcamp-6bf84cb898-dgk8f env`
	`PATH=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin`
	`HOSTNAME=kubernetes-bootcamp-6bf84cb898-dgk8f`
	`KUBERNETES_PORT_443_TCP=tcp://10.96.0.1:443`
	`KUBERNETES_PORT_443_TCP_PROTO=tcp`
	`KUBERNETES_PORT_443_TCP_PORT=443`
	`KUBERNETES_PORT_443_TCP_ADDR=10.96.0.1`
	`KUBERNETES_SERVICE_HOST=10.96.0.1`
	`KUBERNETES_SERVICE_PORT=443`
	`KUBERNETES_SERVICE_PORT_HTTPS=443`
	`KUBERNETES_PORT=tcp://10.96.0.1:443`
	`NPM_CONFIG_LOGLEVEL=info`
	`NODE_VERSION=6.3.1`
	`HOME=/root`
	`[root@ip-172-42-32-4 /]#`
	
	Note: name of the container omitted in the above command, since we only have a single container in the Pod.

7	Run command "sudo kubectl exec -ti < kubernetes-bootcamp-6bf84cb898-dgk8f > bash" as shown below to open console on the container where we run our application (NodeJS).
	
	`[root@ip-172-42-32-4 /]# sudo kubectl exec -ti kubernetes-bootcamp-6bf84cb898-dgk8f bash`
	`root@kubernetes-bootcamp-6bf84cb898-dgk8f:/#`

8	Run command as shown below to see source code of the containerized app, code is in the server.js file
	
	`[root@ip-172-42-32-4 /]# sudo kubectl exec -ti kubernetes-bootcamp-6bf84cb898-dgk8f bash`
	`root@kubernetes-bootcamp-6bf84cb898-dgk8f:/# cat server.js`
	`var http = require('http');`
	`var requests=0;`
	`var podname= process.env.HOSTNAME;`
	`var startTime;`
	`var host;`
	`var handleRequest = function(request, response) {`
	`  response.setHeader('Content-Type', 'text/plain');`
	`  response.writeHead(200);`
	`  response.write("Hello Kubernetes bootcamp! | Running on: ");`
	`  response.write(host);`
	`  response.end(" | v=1\n");`
	`  console.log("Running On:" ,host, "| Total Requests:", ++requests,"| App Uptime:", (new Date() - startTime)/1000 , "seconds", "| Log Time:",new Date());`
	`}`
	`var www = http.createServer(handleRequest);`
	`www.listen(8080,function () {`
	`    startTime = new Date();;`
	`    host = process.env.HOSTNAME;`
	`    console.log ("Kubernetes Bootcamp App Started At:",startTime, "| Running On: " ,host, "\n" );`
	`});`
	`root@kubernetes-bootcamp-6bf84cb898-dgk8f:/#`

	Run command "curl localhost:8080" as shown below to check that the application is up
	
	`root@kubernetes-bootcamp-6bf84cb898-dgk8f:/# curl localhost:8080`
	`Hello Kubernetes bootcamp! | Running on: kubernetes-bootcamp-6bf84cb898-dgk8f | v=1`
	`root@kubernetes-bootcamp-6bf84cb898-dgk8f:/#`
	
	Note: here we used localhost because we executed the command inside the NodeJS container. In exercise 1.2 step 7, we did by reaching pod through proxy

	You can run “node -v (NodeJS command)” and “node -p '3+2'” commands also

------- End of Exercise -------

Exercise 3.3 Exposing deployed app using services

Scenario

Exposing deployed app using service

To do tasks:

In this exercise, you will be:

- Expose Kubernetes app outside the cluster using kubectl expose command
- View and apply labels to object using kubectl label command

Steps

Exposing App	

1	Run command “kubectl get pods,services” as shown below to get existing pods and services
	
	`[root@ip-172-42-32-4 /]# kubectl get pods,services`
	`NAME READY STATUS RESTARTS AGE`
	`pod/kubernetes-bootcamp-6bf84cb898-dgk8f 1/1 Running 0 62m`
	
	`NAME TYPE CLUSTER-IP EXTERNAL-IP PORT(S) AGE`
	`service/kubernetes ClusterIP 10.96.0.1 <none> 443/TCP 178m`
	`[root@ip-172-42-32-4 /]#`

2	Run command “kubectl expose deployment/kubernetes-bootcamp --type="NodePort" --port 8080” as shown below to create a service
	
	`[root@ip-172-42-32-4 /]# kubectl expose deployment/kubernetes-bootcamp --type="NodePort" --port 8080`
	`service/kubernetes-bootcamp exposed`
	`[root@ip-172-42-32-4 /]#`
	
	You can check newly created sevices using get command “kubectl get services” as shown below
	
	`[root@ip-172-42-32-4 /]# kubectl get services`
	`NAME TYPE CLUSTER-IP EXTERNAL-IP PORT(S) AGE`
	`kubernetes ClusterIP 10.96.0.1 <none> 443/TCP 3h1m`
	`kubernetes-bootcamp NodePort 10.100.178.89 <none> 8080:30119/TCP 67s`
	`[root@ip-172-42-32-4 /]#`
	
	We have now a running Service called kubernetes-bootcamp. Here we see that the Service received a unique cluster-IP, an internal port and an external-IP (the IP of the Node).

3	Run command "kubectl describe service/kubernetes-bootcamp" as shown below to see what port was opened externally
	
	`[root@ip-172-42-32-4 /]# kubectl describe service/kubernetes-bootcamp`
	`Name: kubernetes-bootcamp`
	`Namespace: default`
	`Labels: run=kubernetes-bootcamp`
	`Annotations: <none>`
	`Selector: run=kubernetes-bootcamp`
	`Type: NodePort`
	`IP: 10.100.178.89`
	`Port: <unset> 8080/TCP`
	`TargetPort: 8080/TCP`
	`NodePort: <unset> 30119/TCP`
	`Endpoints: 172.17.0.4:8080`
	`Session Affinity: None`
	`External Traffic Policy: Cluster`
	`Events: <none>`
	`[root@ip-172-42-32-4 /]#`

4	Try below mentioned options to access app UI 1]curl http://<service IP>: target port 2]curl http://<pod IP>:target port 3]curl http://<node IP>:NodePort Note: try all combinations of IP and ports, out side cluster also.

6	Run command "kubectl describe service/kubernetes-bootcamp" as shown below to see label created by default
	
	`[root@ip-172-42-32-4 /]# kubectl describe service/kubernetes-bootcamp`
	`Name: kubernetes-bootcamp`
	`Namespace: default`
	`Labels: run=kubernetes-bootcamp`
	`Annotations: <none>`
	`Selector: run=kubernetes-bootcamp`
	`Type: NodePort`
	`IP: 10.100.178.89`
	`Port: <unset> 8080/TCP`
	`TargetPort: 8080/TCP`
	`NodePort: <unset> 30119/TCP`
	`Endpoints: 172.17.0.4:8080`
	`Session Affinity: None`
	`External Traffic Policy: Cluster`
	`Events: <none>`
	`[root@ip-172-42-32-4 /]#`

7	Run command "kubectl get pods -l run=kubernetes-bootcamp" as shown below to get pods associated with label

```
[root@ip-172-42-32-4 /]# kubectl get pods -l run=kubernetes-bootcamp
NAME                                  READY   STATUS    RESTARTS   AGE
kubernetes-bootcamp-6bf84cb898-dgk8f   1/1     Running   0          80m
[root@ip-172-42-32-4 /]#
```

9 Run command as shown below to add label to pod.

To apply a new label, we use the label command followed by the object type, object name and the new label

"kubectl label pod <kubernetes-bootcamp-6bf84cb898-dgk8f > app=v1"

```
[root@ip-172-42-32-4 /]# kubectl label pod kubernetes-bootcamp-6bf84cb898-dgk8f app=v1
pod/kubernetes-bootcamp-6bf84cb898-dgk8f labeled
[root@ip-172-42-32-4 /]#
```

This will apply a new label app=v1 to our Pod (we pinned the application version to the Pod), and we can check it with the describe pod command.

10 Run command "kubectl get pods -l app=v1" as shown below to see list of pods using the new label

```
[root@ip-172-42-32-4 /]# kubectl get pods -l app=v1
NAME                                  READY   STATUS    RESTARTS   AGE
kubernetes-bootcamp-6bf84cb898-dgk8f   1/1     Running   0          85m
[root@ip-172-42-32-4 /]#
```

11 Run command "kubectl delete service -l run=kubernetes-bootcamp" as shown below to delete service

```
[user1@ip-172-42-32-7 ~]$ sudo kubectl get services
NAME                 TYPE        CLUSTER-IP      EXTERNAL-IP   PORT(S)          AGE
kubernetes           ClusterIP   10.96.0.1       <none>        443/TCP          4h
kubernetes-bootcamp  NodePort    10.101.211.56   <none>        8080:32487/TCP   1h
[user1@ip-172-42-32-7 ~]$ sudo kubectl delete service -l run=kubernetes-bootcamp
service "kubernetes-bootcamp" deleted
[user1@ip-172-42-32-7 ~]$ sudo kubectl get services
NAME         TYPE        CLUSTER-IP   EXTERNAL-IP   PORT(S)   AGE
kubernetes   ClusterIP   10.96.0.1    <none>        443/TCP   4h
[user1@ip-172-42-32-7 ~]$
```

This confirms that our Service was removed. To confirm that route is not exposed anymore you can curl the previously exposed IP and port

Above command proves that the app is not reachable anymore from outside of the cluster

------- End of Exercise -------

Exercise 3.4 Scaling your App

Scenario

Scale a deployment with Kubectl

To do tasks:

In this exercise, you will be:

- Scale deployment with kubectl scale
- Load balancing

Steps

Scaling app	

1	Run command "kubectl get deployments" as shown below to check deployments
	
	`[root@ip-172-42-32-4 /]# kubectl get deployments`
	`NAME  READY  UP-TO-DATE  AVAILABLE  AGE`
	`kubernetes-bootcamp  1/1  1  1  93m`
	`[root@ip-172-42-32-4 /]#`
	
	We should have 1 Pod. If not, run the command again.
	
	The DESIRED state is showing the configured number of replicas
	
	The CURRENT state shows how many replicas are running now
	
	The UP-TO-DATE is the number of replicas that were updated to match the desired (configured) state
	
	The AVAILABLE state shows how many replicas are AVAILABLE to the users

2	Run command as shown below to scale the Deployment to 4 replicas. Use the kubectl scale command, followed by the deployment type, name and desired number of instances
	
	"kubectl scale deployments/kubernetes-bootcamp --replicas=4"
	
	`[root@ip-172-42-32-4 /]# kubectl scale deployments/kubernetes-bootcamp --replicas=4`
	`deployment.extensions/kubernetes-bootcamp scaled`
	`[root@ip-172-42-32-4 /]#`
	
	To list your Deployments once again, run command " kubectl get deployments" as shown below
	
	`[root@ip-172-42-32-4 /]#  kubectl get deployments`
	`NAME  READY  UP-TO-DATE  AVAILABLE  AGE`
	`kubernetes-bootcamp  4/4  4  4  98m`
	`[root@ip-172-42-32-4 /]#`

3	We have 4 instances of the application available. Check if the number of Pods changed using command "kubectl get pods -o wide"
	
	`[root@ip-172-42-32-4 /]# kubectl get pods -o wide`
	`NAME  READY  STATUS  RESTARTS  AGE  IP  NODE  NOMINATED NODE  READINESS GATES`
	`kubernetes-bootcamp-6bf84cb898-524dq  1/1  Running  0  2m26s  172.17.0.5  minikube  <none>  <none>`
	`kubernetes-bootcamp-6bf84cb898-dgk8f  1/1  Running  0  99m  172.17.0.4  minikube  <none>  <none>`
	`kubernetes-bootcamp-6bf84cb898-s4ps4  1/1  Running  0  2m26s  172.17.0.7  minikube  <none>  <none>`
	`kubernetes-bootcamp-6bf84cb898-wd6m7  1/1  Running  0  2m26s  172.17.0.6  minikube  <none>  <none>`
	`[root@ip-172-42-32-4 /]#`

	Note: -o=wide id for Output in the plain-text format with any additional information, and for pods, the node name is included

4	The change was registered in the Deployment events log. To check that, use the describe command "kubectl describe deployments/kubernetes-bootcamp"
	
	`[root@ip-172-42-32-4 /]# kubectl describe deployments/kubernetes-bootcamp`
	`Name: kubernetes-bootcamp`
	`Namespace: default`
	`CreationTimestamp: Fri, 01 Feb 2019 11:18:05 +0000`
	`Labels: run=kubernetes-bootcamp`
	`Annotations: deployment.kubernetes.io/revision: 1`
	`Selector: run=kubernetes-bootcamp`
	`Replicas: 4 desired | 4 updated | 4 total | 4 available | 0 unavailable`
	`StrategyType: RollingUpdate`
	`MinReadySeconds: 0`
	`RollingUpdateStrategy: 25% max unavailable, 25% max surge`
	`Pod Template:`
	`Labels: run=kubernetes-bootcamp`
	`Containers:`
	`kubernetes-bootcamp:`
	`Image: gcr.io/google-samples/kubernetes-bootcamp:v1`
	`Port: 8080/TCP`
	`Host Port: 0/TCP`
	`Environment: <none>`
	`Mounts: <none>`
	`Volumes: <none>`
	`Conditions:`
	`Type Status Reason`
	`---- ------ ------`
	`Progressing True NewReplicaSetAvailable`
	`Available True MinimumReplicasAvailable`
	`OldReplicaSets: <none>`
	`NewReplicaSet: kubernetes-bootcamp-6bf84cb898 (4/4 replicas created)`
	`Events:`
	`Type Reason Age From Message`
	`---- ------ ---- ---- -------`
	`Normal ScalingReplicaSet 3m11s deployment-controller Scaled up replica set kubernetes-bootcamp-6bf84cb898 to 4`
	`[root@ip-172-42-32-4 /]#`
	`NewReplicaSet: kubernetes-bootcamp-5c69669756 (4/4 replicas created)`
	`Events:`
	`Type Reason Age From Message`
	`---- ------ ---- ---- -------`
	`Normal ScalingReplicaSet 17m deployment-controller Scaled up replica set kubernetes-bootcamp-5c69669756 to 4`
	`[user1@ip-172-42-32-7 ~]$`

5	Run command "kubectl expose deployment/kubernetes-bootcamp --type="NodePort" --port 8080" as shown below to create a service
	
	`[user1@ip-172-42-32-7 ~]$ sudo kubectl expose deployment/kubernetes-bootcamp --type="NodePort" --port 8080`
	`service/kubernetes-bootcamp exposed`
	`[user1@ip-172-42-32-7 ~]$`
	
	Note: service creation not required if you didn't disturb (deleted) service created in exercise 3.3 step for the same deployment
	
	You can check newly created sevices using get command " kubectl get services" as shown below

```
[root@ip-172-42-32-4 /]# kubectl get services
NAME                  TYPE        CLUSTER-IP      EXTERNAL-IP   PORT(S)          AGE
kubernetes            ClusterIP   10.96.0.1       <none>        443/TCP          3h39m
kubernetes-bootcamp   NodePort    10.100.178.89   <none>        8080:30119/TCP   39m
[root@ip-172-42-32-4 /]#
```

We have now a running Service called kubernetes-bootcamp. Here we see that the Service received a unique cluster-IP, an internal port and an external-IP (the IP of the Node).

“kubectl describe svc kubernetes-bootcamp”

```
[root@ip-172-42-32-4 /]# kubectl describe svc kubernetes-bootcamp
Name:                     kubernetes-bootcamp
Namespace:                default
Labels:                   run=kubernetes-bootcamp
Annotations:              <none>
Selector:                 run=kubernetes-bootcamp
Type:                     NodePort
IP:                       10.100.178.89
Port:                     <unset>  8080/TCP
TargetPort:               8080/TCP
NodePort:                 <unset>  30119/TCP
Endpoints:                172.17.0.4:8080,172.17.0.5:8080,172.17.0.6:8080 + 1 more...
Session Affinity:         None
External Traffic Policy:  Cluster
Events:                   <none>
[root@ip-172-42-32-4 /]#
```

Try access app UI with service IP and port to see load balancer working, We must send multiple requests to see load balancer working

```
[root@ip-172-42-32-4 /]# curl http://10.100.178.89:8080
Hello Kubernetes bootcamp! | Running on: kubernetes-bootcamp-6bf84cb898-524dq | v=1
[root@ip-172-42-32-4 /]# curl http://10.100.178.89:8080
Hello Kubernetes bootcamp! | Running on: kubernetes-bootcamp-6bf84cb898-wd6m7 | v=1
[root@ip-172-42-32-4 /]# curl http://10.100.178.89:8080
Hello Kubernetes bootcamp! | Running on: kubernetes-bootcamp-6bf84cb898-dgk8f | v=1
[root@ip-172-42-32-4 /]# curl http://10.100.178.89:8080
Hello Kubernetes bootcamp! | Running on: kubernetes-bootcamp-6bf84cb898-524dq | v=1
[root@ip-172-42-32-4 /]# curl http://10.100.178.89:8080
Hello Kubernetes bootcamp! | Running on: kubernetes-bootcamp-6bf84cb898-dgk8f | v=1
[root@ip-172-42-32-4 /]# curl http://10.100.178.89:8080
Hello Kubernetes bootcamp! | Running on: kubernetes-bootcamp-6bf84cb898-dgk8f | v=1
[root@ip-172-42-32-4 /]# curl http://10.100.178.89:8080
Hello Kubernetes bootcamp! | Running on: kubernetes-bootcamp-6bf84cb898-wd6m7 | v=1
[root@ip-172-42-32-4 /]# curl http://10.100.178.89:8080
Hello Kubernetes bootcamp! | Running on: kubernetes-bootcamp-6bf84cb898-dgk8f | v=1
[root@ip-172-42-32-4 /]# curl http://10.100.178.89:8080
Hello Kubernetes bootcamp! | Running on: kubernetes-bootcamp-6bf84cb898-524dq | v=1
[root@ip-172-42-32-4 /]# curl http://10.100.178.89:8080
Hello Kubernetes bootcamp! | Running on: kubernetes-bootcamp-6bf84cb898-dgk8f | v=1
[root@ip-172-42-32-4 /]# curl http://10.100.178.89:8080
Hello Kubernetes bootcamp! | Running on: kubernetes-bootcamp-6bf84cb898-wd6m7 | v=1
[root@ip-172-42-32-4 /]# curl http://10.100.178.89:8080
Hello Kubernetes bootcamp! | Running on: kubernetes-bootcamp-6bf84cb898-wd6m7 | v=1
[root@ip-172-42-32-4 /]# curl http://10.100.178.89:8080
Hello Kubernetes bootcamp! | Running on: kubernetes-bootcamp-6bf84cb898-wd6m7 | v=1
[root@ip-172-42-32-4 /]# curl http://10.100.178.89:8080
Hello Kubernetes bootcamp! | Running on: kubernetes-bootcamp-6bf84cb898-524dq | v=1
[root@ip-172-42-32-4 /]# curl http://10.100.178.89:8080
Hello Kubernetes bootcamp! | Running on: kubernetes-bootcamp-6bf84cb898-dgk8f | v=1
[root@ip-172-42-32-4 /]# curl http://10.100.178.89:8080
Hello Kubernetes bootcamp! | Running on: kubernetes-bootcamp-6bf84cb898-s4ps4 | v=1
[root@ip-172-42-32-4 /]# curl http://10.100.178.89:8080
Hello Kubernetes bootcamp! | Running on: kubernetes-bootcamp-6bf84cb898-s4ps4 | v=1
[root@ip-172-42-32-4 /]# curl http://10.100.178.89:8080
Hello Kubernetes bootcamp! | Running on: kubernetes-bootcamp-6bf84cb898-s4ps4 | v=1
[root@ip-172-42-32-4 /]# curl http://10.100.178.89:8080
Hello Kubernetes bootcamp! | Running on: kubernetes-bootcamp-6bf84cb898-524dq | v=1
[root@ip-172-42-32-4 /]# curl http://10.100.178.89:8080
Hello Kubernetes bootcamp! | Running on: kubernetes-bootcamp-6bf84cb898-s4ps4 | v=1
[root@ip-172-42-32-4 /]# curl http://10.100.178.89:8080
Hello Kubernetes bootcamp! | Running on: kubernetes-bootcamp-6bf84cb898-wd6m7 | v=1
[root@ip-172-42-32-4 /]# curl http://10.100.178.89:8080
Hello Kubernetes bootcamp! | Running on: kubernetes-bootcamp-6bf84cb898-s4ps4 | v=1
[root@ip-172-42-32-4 /]#
```

7

Run command “kubectl scale deployments/kubernetes-bootcamp --replicas=2” as shown below to scale down the deployments

```
[root@ip-172-42-32-4 /]# kubectl scale deployments/kubernetes-bootcamp --replicas=2
deployment.extensions/kubernetes-bootcamp scaled
[root@ip-172-42-32-4 /]#
```

You can check scale in as shown below

```
[root@ip-172-42-32-4 /]# kubectl get pods,deployments
NAME                                        READY   STATUS    RESTARTS   AGE
pod/kubernetes-bootcamp-6bf84cb898-dgk8f    1/1     Running   0          111m
pod/kubernetes-bootcamp-6bf84cb898-wd6m7    1/1     Running   0          14m

NAME                                          READY   UP-TO-DATE   AVAILABLE   AGE
deployment.extensions/kubernetes-bootcamp     2/2     2            2           111m
[root@ip-172-42-32-4 /]#
```

------- End of Exercise -------

Exercise 3.5 Updating App

Scenario

Update a deployed application

To do tasks:

In this exercise, you will be:

- Updating and rollback app using kubectl set and rollout commands

Steps

Updating App	

2	Run commands as shown below to get the deployment and details
	
	`[root@ip-172-42-32-4 /]# kubectl get pods,deployments`
	`NAME READY STATUS RESTARTS AGE`
	`pod/kubernetes-bootcamp-6bf84cb898-dgk8f 1/1 Running 0 111m`
	`pod/kubernetes-bootcamp-6bf84cb898-wd6m7 1/1 Running 0 14m`
	
	`NAME READY UP-TO-DATE AVAILABLE AGE`
	`deployment.extensions/kubernetes-bootcamp 2/2 2 2 111m`
	`[root@ip-172-42-32-4 /]#`

3	Run command as shown below to view the current image version of the app, run a describe command against the Pods

```
[root@ip-172-42-32-4 /]# kubectl describe deployments kubernetes-bootcamp
Name:                   kubernetes-bootcamp
Namespace:              default
CreationTimestamp:      Fri, 01 Feb 2019 11:18:05 +0000
Labels:                 run=kubernetes-bootcamp
Annotations:            deployment.kubernetes.io/revision: 1
Selector:               run=kubernetes-bootcamp
Replicas:               2 desired | 2 updated | 2 total | 2 available | 0 unavailable
StrategyType:           RollingUpdate
MinReadySeconds:        0
RollingUpdateStrategy:  25% max unavailable, 25% max surge
Pod Template:
  Labels:  run=kubernetes-bootcamp
  Containers:
   kubernetes-bootcamp:
    Image:        gcr.io/google-samples/kubernetes-bootcamp:v1
    Port:         8080/TCP
    Host Port:    0/TCP
    Environment:  <none>
    Mounts:       <none>
  Volumes:        <none>
Conditions:
  Type           Status  Reason
  ----           ------  ------
  Progressing    True    NewReplicaSetAvailable
  Available      True    MinimumReplicasAvailable
OldReplicaSets:  <none>
NewReplicaSet:   kubernetes-bootcamp-6bf84cb898 (2/2 replicas created)
Events:
  Type    Reason             Age   From                   Message
  ----    ------             ----  ----                   -------
  Normal  ScalingReplicaSet  31m   deployment-controller  Scaled up replica set kubernetes-bootcamp-6bf84cb898 to 4
  Normal  ScalingReplicaSet  18m   deployment-controller  Scaled down replica set kubernetes-bootcamp-6bf84cb898 to 2
[root@ip-172-42-32-4 /]#
```

4 Run command “kubectl set image deployments/kubernetes-bootcamp kubernetes-bootcamp=jocatalin/kubernetes-bootcamp:v2” as shown below to update the image of the application to version 2

```
[root@ip-172-42-32-4 /]# kubectl set image deployments/kubernetes-bootcamp kubernetes-bootcamp=jocatalin/kubernetes-bootcamp:v2
deployment.extensions/kubernetes-bootcamp image updated
[root@ip-172-42-32-4 /]#
```

The command notified the Deployment to use a different image for the app and initiated a rolling update.

5 Run command as shown below to Check the status of the new Pods, and view the old one terminating

```
[root@ip-172-42-32-4 /]# kubectl get pods
NAME                                  READY   STATUS    RESTARTS   AGE
kubernetes-bootcamp-5bf4d5689b-2s562  1/1     Running   0          2m47s
kubernetes-bootcamp-5bf4d5689b-8g5j8  1/1     Running   0          2m41s
[root@ip-172-42-32-4 /]#
```

<table>
<tr><td></td><td>

```
[root@ip-172-42-32-4 /]# kubectl describe deployments kubernetes-bootcamp
Name:                   kubernetes-bootcamp
Namespace:              default
CreationTimestamp:      Fri, 01 Feb 2019 11:18:05 +0000
Labels:                 run=kubernetes-bootcamp
Annotations:            deployment.kubernetes.io/revision: 2
Selector:               run=kubernetes-bootcamp
Replicas:               2 desired | 2 updated | 2 total | 2 available | 0 unavailable
StrategyType:           RollingUpdate
MinReadySeconds:        0
RollingUpdateStrategy:  25% max unavailable, 25% max surge
Pod Template:
  Labels:  run=kubernetes-bootcamp
  Containers:
   kubernetes-bootcamp:
    Image:        jocatalin/kubernetes-bootcamp:v2
    Port:         8080/TCP
    Host Port:    0/TCP
    Environment:  <none>
    Mounts:       <none>
  Volumes:        <none>
Conditions:
  Type           Status  Reason
  ----           ------  ------
  Available      True    MinimumReplicasAvailable
  Progressing    True    NewReplicaSetAvailable
OldReplicaSets:  <none>
NewReplicaSet:   kubernetes-bootcamp-5bf4d5689b (2/2 replicas created)
Events:
  Type    Reason             Age    From                   Message
  ----    ------             ----   ----                   -------
  Normal  ScalingReplicaSet  39m    deployment-controller  Scaled up replica set kubernetes-bootcamp-6bf84cb898 to 4
  Normal  ScalingReplicaSet  26m    deployment-controller  Scaled down replica set kubernetes-bootcamp-6bf84cb898 to 2
  Normal  ScalingReplicaSet  4m44s  deployment-controller  Scaled up replica set kubernetes-bootcamp-5bf4d5689b to 1
  Normal  ScalingReplicaSet  4m38s  deployment-controller  Scaled down replica set kubernetes-bootcamp-6bf84cb898 to 1
  Normal  ScalingReplicaSet  4m38s  deployment-controller  Scaled up replica set kubernetes-bootcamp-5bf4d5689b to 2
  Normal  ScalingReplicaSet  4m37s  deployment-controller  Scaled down replica set kubernetes-bootcamp-6bf84cb898 to 0
[root@ip-172-42-32-4 /]#
```

</td></tr>
<tr><td>6</td><td>

Access App UI using service

```
[root@ip-172-42-32-4 /]# curl http://10.100.178.89:8080
Hello Kubernetes bootcamp! | Running on: kubernetes-bootcamp-5bf4d5689b-8g5j8 | v=2
[root@ip-172-42-32-4 /]# curl http://10.100.178.89:8080
Hello Kubernetes bootcamp! | Running on: kubernetes-bootcamp-5bf4d5689b-2s562 | v=2
[root@ip-172-42-32-4 /]# curl http://10.100.178.89:8080
Hello Kubernetes bootcamp! | Running on: kubernetes-bootcamp-5bf4d5689b-8g5j8 | v=2
[root@ip-172-42-32-4 /]# curl http://10.100.178.89:8080
Hello Kubernetes bootcamp! | Running on: kubernetes-bootcamp-5bf4d5689b-2s562 | v=2
[root@ip-172-42-32-4 /]# curl http://10.100.178.89:8080
Hello Kubernetes bootcamp! | Running on: kubernetes-bootcamp-5bf4d5689b-8g5j8 | v=2
[root@ip-172-42-32-4 /]# curl http://10.100.178.89:8080
Hello Kubernetes bootcamp! | Running on: kubernetes-bootcamp-5bf4d5689b-2s562 | v=2
[root@ip-172-42-32-4 /]# curl http://10.100.178.89:8080
Hello Kubernetes bootcamp! | Running on: kubernetes-bootcamp-5bf4d5689b-2s562 | v=2
[root@ip-172-42-32-4 /]# curl http://10.100.178.89:8080
Hello Kubernetes bootcamp! | Running on: kubernetes-bootcamp-5bf4d5689b-2s562 | v=2
[root@ip-172-42-32-4 /]# curl http://10.100.178.89:8080
Hello Kubernetes bootcamp! | Running on: kubernetes-bootcamp-5bf4d5689b-8g5j8 | v=2
[root@ip-172-42-32-4 /]#
```

</td></tr>
<tr><td>9</td><td>

Run command as shown below to see the rollout status

</td></tr>
</table>

	[user1@ip-172-42-32-7 ~]$ sudo kubectl rollout status deployments/kubernetes-bootcamp deployment "kubernetes-bootcamp" successfully rolled out [user1@ip-172-42-32-7 ~]$
11	Let's perform another update, and deploy image tagged as v10 "kubectl set image deployments/kubernetes-bootcamp kubernetes-bootcamp=jocatalin/kubernetes-bootcamp:v10" [user1@ip-172-42-32-7 ~]$ sudo kubectl set image deployments/kubernetes-bootcamp kubernetes-bootcamp=jocatalin/kubernetes-bootcamp:v10 deployment.extensions/kubernetes-bootcamp image updated [user1@ip-172-42-32-7 ~]$
12	Use get deployments to see the status of the deployments And something is wrong… We do not have the desired number of Pods available. List the Pods again as shown below [root@ip-172-42-32-4 /]# kubectl get deployments,pods NAME READY UP-TO-DATE AVAILABLE AGE deployment.extensions/kubernetes-bootcamp 2/2 1 2 144m NAME READY STATUS RESTARTS AGE pod/kubernetes-bootcamp-5bf4d5689b-2s562 1/1 Running 0 12m pod/kubernetes-bootcamp-5bf4d5689b-8g5j8 1/1 Running 0 12m pod/kubernetes-bootcamp-c46798fc9-wgqzc 0/1 ImagePullBackOff 0 100s [root@ip-172-42-32-4 /]#
13	Run describe command on the Pods to get more insights

```
      Started:      Fri, 01 Feb 2019 13:30:36 +0000
    Ready:          True
    Restart Count:  0
    Environment:    <none>
    Mounts:
      /var/run/secrets/kubernetes.io/serviceaccount from default-token-rmfdk (ro)
Conditions:
  Type              Status
  Initialized       True
  Ready             True
  ContainersReady   True
  PodScheduled      True
Volumes:
  default-token-rmfdk:
    Type:        Secret (a volume populated by a Secret)
    SecretName:  default-token-rmfdk
    Optional:    false
QoS Class:       BestEffort
Node-Selectors:  <none>
Tolerations:     node.kubernetes.io/not-ready:NoExecute for 300s
                 node.kubernetes.io/unreachable:NoExecute for 300s
Events:
  Type    Reason     Age   From               Message
  ----    ------     ----  ----               -------
  Normal  Scheduled  13m   default-scheduler  Successfully assigned default/kubernetes-bootcamp-5bf4d5689b-8g5j8 to minikube
  Normal  Pulled     13m   kubelet, minikube  Container image "jocatalin/kubernetes-bootcamp:v2" already present on machine
  Normal  Created    13m   kubelet, minikube  Created container
  Normal  Started    13m   kubelet, minikube  Started container


Name:               kubernetes-bootcamp-c46798fc9-wgqzc
Namespace:          default
Priority:           0
PriorityClassName:  <none>
Node:               minikube/172.42.32.4
Start Time:         Fri, 01 Feb 2019 13:40:55 +0000
Labels:             pod-template-hash=c46798fc9
                    run=kubernetes-bootcamp
Annotations:        <none>
Status:             Pending
IP:                 172.17.0.4
Controlled By:      ReplicaSet/kubernetes-bootcamp-c46798fc9
Containers:
  kubernetes-bootcamp:
    Container ID:
    Image:          jocatalin/kubernetes-bootcamp:v10
    Image ID:
    Port:           8080/TCP
    Host Port:      0/TCP
    State:          Waiting
      Reason:       ErrImagePull
    Ready:          False
    Restart Count:  0
    Environment:    <none>
    Mounts:
      /var/run/secrets/kubernetes.io/serviceaccount from default-token-rmfdk (ro)
Conditions:
  Type              Status
  Initialized       True
```

Issue with v10 in the repository. Let's roll back to our previously working version. We'll use the rollout undo command

14

Run command "kubectl rollout undo deployments/kubernetes-bootcamp" as show below to revert the deployment to the previous known state (v2 of the image).

```
[root@ip-172-42-32-4 user1]# kubectl rollout undo deployments/kubernetes-bootcamp
deployment.extensions/kubernetes-bootcamp rolled back
[root@ip-172-42-32-4 user1]#
```

15

To see everything ok after rollback and Pods are running. Check again the image deployed on the them as shown below

```
[root@ip-172-42-32-4 /]# kubectl get pods
NAME                                   READY   STATUS    RESTARTS   AGE
kubernetes-bootcamp-5bf4d5689b-2s562   1/1     Running   1          23m
kubernetes-bootcamp-5bf4d5689b-8g5j8   1/1     Running   1          23m
[root@ip-172-42-32-4 /]#
```

16

You can see deployment is using a stable version of the app (v2). The Rollback was successful.

```
[root@ip-172-42-32-4 /]# kubectl get deployments
NAME                  READY   UP-TO-DATE   AVAILABLE   AGE
kubernetes-bootcamp   2/2     2            2           156m
[root@ip-172-42-32-4 /]# kubectl describe deployments kubernetes-bootcamp
Name:                   kubernetes-bootcamp
Namespace:              default
CreationTimestamp:      Fri, 01 Feb 2019 11:18:05 +0000
Labels:                 run=kubernetes-bootcamp
Annotations:            deployment.kubernetes.io/revision: 4
Selector:               run=kubernetes-bootcamp
Replicas:               2 desired | 2 updated | 2 total | 2 available | 0 unavailable
StrategyType:           RollingUpdate
MinReadySeconds:        0
RollingUpdateStrategy:  25% max unavailable, 25% max surge
Pod Template:
  Labels:  run=kubernetes-bootcamp
  Containers:
   kubernetes-bootcamp:
    Image:        jocatalin/kubernetes-bootcamp:v2
    Port:         8080/TCP
    Host Port:    0/TCP
    Environment:  <none>
    Mounts:       <none>
  Volumes:        <none>
Conditions:
  Type           Status  Reason
  ----           ------  ------
  Available      True    MinimumReplicasAvailable
  Progressing    True    NewReplicaSetAvailable
OldReplicaSets:  <none>
NewReplicaSet:   kubernetes-bootcamp-5bf4d5689b (2/2 replicas created)
Events:
  Type    Reason             Age    From                   Message
  ----    ------             ----   ----                   -------
  Normal  ScalingReplicaSet  60m    deployment-controller  Scaled up replica set kubernetes-bootcamp-6bf84cb898 to 4
  Normal  ScalingReplicaSet  47m    deployment-controller  Scaled down replica set kubernetes-bootcamp-6bf84cb898 to 2
  Normal  ScalingReplicaSet  25m    deployment-controller  Scaled up replica set kubernetes-bootcamp-5bf4d5689b to 1
  Normal  ScalingReplicaSet  25m    deployment-controller  Scaled down replica set kubernetes-bootcamp-6bf84cb898 to 1
  Normal  ScalingReplicaSet  25m    deployment-controller  Scaled up replica set kubernetes-bootcamp-5bf4d5689b to 2
  Normal  ScalingReplicaSet  25m    deployment-controller  Scaled down replica set kubernetes-bootcamp-6bf84cb898 to 0
  Normal  ScalingReplicaSet  14m    deployment-controller  Scaled up replica set kubernetes-bootcamp-c46798fc9 to 1
  Normal  ScalingReplicaSet  2m36s  deployment-controller  Scaled down replica set kubernetes-bootcamp-c46798fc9 to 0
[root@ip-172-42-32-4 /]#
```

------- End of Exercise -------

Module 4 – Kubernetes Cluster Advanced

====

Exercise 4.1 Deploying Firsthello app onto Kubernetes cluster

Scenario

Deploy simple containerized app on to Kubernetes cluster

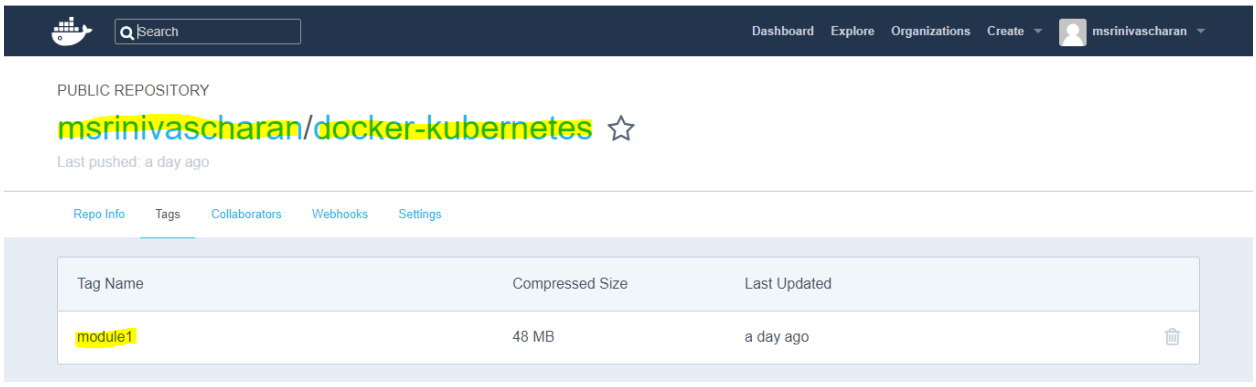
To do tasks:

In this exercise, you will be:

- Deploy Python based containerized app on to Kubernetes cluster
- Using Kubernetes replication controller and services

Note: Exercise 1.1 is prerequisite to this exercise

Steps

Deploying containerized app on to Kubernetes cluster	
1	Ensure Docker daemon is running
2	Stop and delete “firsthello” app containers if any, use below mentioned commands 1. sudo docker container ls -a 2. sudo docker stop <containerid> 3. sudo docker ps -f "status=exited" 4. sudo rm <containerid>
3	Delete “firsthello” images from local repository if any, use below mentioned commands 1. sudo docker images 2. sudo docker rmi <imageid>
4	Copy details of “firsthello” image pushed (part of exercise 1.1) to Dockerhub as shown below  Create workplace and add depoyment and service configurations as shown below

```
[user1@ip-172-42-32-7 FirsthelloWS]$ pwd
/kubews/FirsthelloWS
[user1@ip-172-42-32-7 FirsthelloWS]$ ls
firsthellorc.yaml  firsthellosvc.yaml
[user1@ip-172-42-32-7 FirsthelloWS]$
```

firsthellorc.yaml

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: firsthello-deployment
  labels:
    app: firsthello
spec:
  replicas: 3
  selector:
    matchLabels:
      app: firsthello
  template:
    metadata:
      labels:
        app: firsthello
    spec:
      containers:
      - name: firsthello
        image: msrinivascharan/docker-kubernetes:module1
        ports:
        - containerPort: 80
```

firsthellosvc.yaml

```
kind: Service
apiVersion: v1
metadata:
  name: firsthello-service
spec:
  selector:
    app: firsthello
  ports:
  - protocol: TCP
    port: 8011
    targetPort: 80
    name: fh
  type: NodePort
```

5	Deploy app onto Kubernetes cluster with create command as shown below

```
[user1@ip-172-42-32-7 FirsthelloWS]$ ls
firsthellorc.yaml  firsthellosvc.yaml
[user1@ip-172-42-32-7 FirsthelloWS]$ sudo kubectl create -f firsthellorc.yaml
[sudo] password for user1:
replicationcontroller/firsthello created
[user1@ip-172-42-32-7 FirsthelloWS]$ sudo kubectl get rc
NAME         DESIRED   CURRENT   READY     AGE
firsthello   1         1         1         13s
[user1@ip-172-42-32-7 FirsthelloWS]$ sudo kubectl get pods
NAME                       READY     STATUS    RESTARTS   AGE
firsthello-f6md6           1/1       Running   0          17s
mysql-6bfc69bb74-s6rf9     1/1       Running   0          5h
[user1@ip-172-42-32-7 FirsthelloWS]$
```

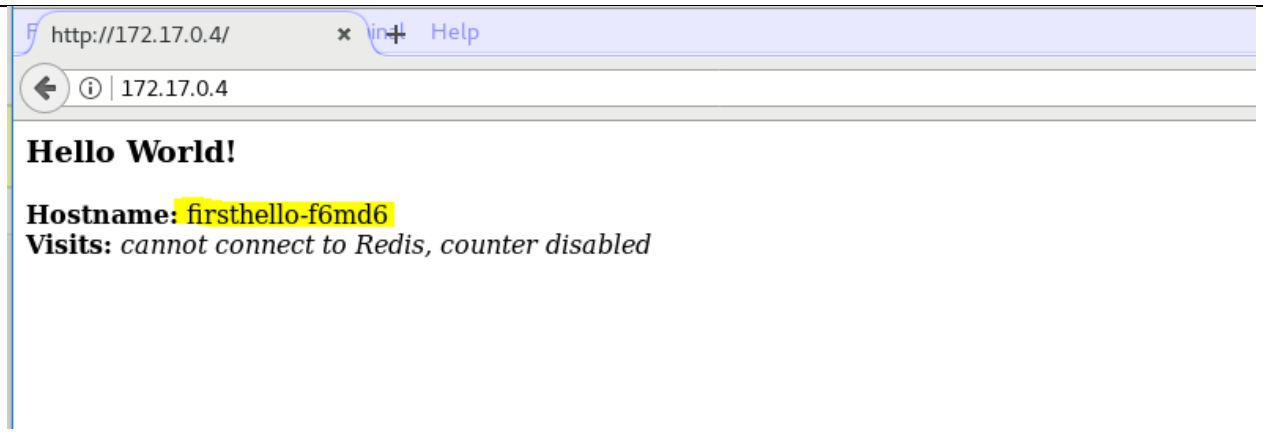
6 Create service to expose deployed app using create command as shown below

```
[user1@ip-172-42-32-7 FirsthelloWS]$ ls
firsthellorc.yaml  firsthellosvc.yaml
[user1@ip-172-42-32-7 FirsthelloWS]$ sudo kubectl create -f firsthellosvc.yaml
service/firsthellosvc created
[user1@ip-172-42-32-7 FirsthelloWS]$ sudo kubectl get services
NAME            TYPE        CLUSTER-IP      EXTERNAL-IP   PORT(S)   AGE
firsthellosvc   ClusterIP   10.99.124.224   <none>        80/TCP    10s
kubernetes      ClusterIP   10.96.0.1       <none>        443/TCP   14d
[user1@ip-172-42-32-7 FirsthelloWS]$ sudo kubectl describe service/firsthellosvc
Name:              firsthellosvc
Namespace:         default
Labels:            app=firsthello
Annotations:       <none>
Selector:          app=firsthello
Type:              ClusterIP
IP:                10.99.124.224
Port:              <unset>  80/TCP
TargetPort:        80/TCP
Endpoints:         172.17.0.4:80
Session Affinity:  None
Events:            <none>
[user1@ip-172-42-32-7 FirsthelloWS]$
```

Run command as shown below to get “firsthellosvc” service URL

```
[user1@ip-172-42-32-7 FirsthelloWS]$ sudo /usr/local/bin/minikube service firsthellosvc --url
http://172.42.32.7:31882
[user1@ip-172-42-32-7 FirsthelloWS]$
```

You can see app UI as shown below



You can check downloaded image and running container as shown below

```
[user1@ip-172-42-32-7 FirsthelloWS]$ sudo docker images
REPOSITORY                                  TAG                 IMAGE ID            CREATED             SIZE
msrinivascharan/docker-kubernetes           module1             b0340533957b        29 hours ago        132MB
nginx                                       latest              8b89e4bb5f15        2 weeks ago         109MB
mysql                                       5.6                 da785d6c6ed8        2 weeks ago         256MB
hello-world                                 latest              2cb0d9787c4d        3 weeks ago         1.85kB
k8s.gcr.io/kube-proxy-amd64                 v1.10.0             bfc21aadc7d3        4 months ago        97MB
k8s.gcr.io/kube-controller-manager-amd64    v1.10.0             ad86dbed1555        4 months ago        148MB
k8s.gcr.io/kube-scheduler-amd64             v1.10.0             704ba848e69a        4 months ago        50.4MB
k8s.gcr.io/kube-apiserver-amd64             v1.10.0             af20925d51a3        4 months ago        225MB
k8s.gcr.io/etcd-amd64                       3.1.12              52920ad46f5b        4 months ago        193MB
k8s.gcr.io/kube-addon-manager               v8.6                9c16409588eb        5 months ago        78.4MB
k8s.gcr.io/k8s-dns-dnsmasq-nanny-amd64      1.14.8              c2ce1ffb51ed        6 months ago        41MB
k8s.gcr.io/k8s-dns-sidecar-amd64            1.14.8              6f7f2dc7fab5        6 months ago        42.2MB
k8s.gcr.io/k8s-dns-kube-dns-amd64           1.14.8              80cc5ea4b547        6 months ago        50.5MB
k8s.gcr.io/pause-amd64                      3.1                 da86e6ba6ca1        7 months ago        742kB
k8s.gcr.io/kubernetes-dashboard-amd64       v1.8.1              e94d2f21bc0c        7 months ago        121MB
gcr.io/k8s-minikube/storage-provisioner     v1.8.1              4689081edb10        8 months ago        80.8MB
wordpress                                   4.8-apache          fcf3e41b8864        8 months ago        408MB
jocatalin/kubernetes-bootcamp               v2                  b6556396ebd4        24 months ago       211MB
gcr.io/google-samples/kubernetes-bootcamp   v1                  8fafd8af70e9        24 months ago       211MB
gcr.io/google-samples/gb-frontend           v4                  e2b3e8542af7        2 years ago         512MB
gcr.io/google_samples/gb-redisslave         v1                  5f026ddffa27        2 years ago         110MB
k8s.gcr.io/redis                            e2e                 e5e67996c442        3 years ago         419MB
[user1@ip-172-42-32-7 FirsthelloWS]$
```

```
[user1@ip-172-42-32-7 FirsthelloWS]$ sudo docker container ls
CONTAINER ID        IMAGE                                 COMMAND                  CREATED             STATUS              PORTS               NAMES
805db3743fa3        c2ce1ffb51ed                          "/dnsmasq-nanny -v=2…"   About a minute ago  Up About a minute                       k8s_dnsmasq_kube-dns-86f4d74b45-v9vmk_kube-system_a1e5cb43-8a66-11e8-9525-0a3f336079ae_902
38ce26f525af        msrinivascharan/docker-kubernetes     "python app.py"          7 minutes ago       Up 7 minutes                            k8s_firsthello_firsthello-f6md6_default_62ed6cca-9583-11e8-a645-0a3f336079ae_0
a0aa8e866172        k8s.gcr.io/pause-amd64:3.1            "/pause"                 7 minutes ago       Up 7 minutes                            k8s_POD_firsthello-f6md6_default_62ed6cca-9583-11e8-a645-0a3f336079ae_0
c19d6efe7999        da785d6c6ed8                          "docker-entrypoint.s…"   5 hours ago         Up 5 hours                              k8s_mysql_mysql-6bfc69bb74-s6rf9_default_ea4df940-9557-11e8-a645-0a3f336079ae_0
a39364dc69ae        k8s.gcr.io/pause-amd64:3.1            "/pause"                 5 hours ago         Up 5 hours                              k8s_POD_mysql-6bfc69bb74-s6rf9_default_ea4df940-9557-11e8-a645-0a3f336079ae_0
ebaf9237917c        4689081edb10                          "/storage-provisioner"   5 hours ago         Up 5 hours                              k8s_storage-provisioner_storage-provisioner_kube-system_b6aff20a-8a73-11e8-9698-0a3f336079ae_25
feae7645f7af        bfc21aadc7d3                          "/usr/local/bin/kube…"   5 hours ago         Up 5 hours                              k8s_kube-proxy_kube-proxy-5n45r_kube-system_8d2fe455-919b-11e8-bd99-0a3f336079ae_3
c85e423a692e        k8s.gcr.io/pause-amd64:3.1            "/pause"                 5 hours ago         Up 5 hours                              k8s_POD_kube-proxy-5n45r_kube-system_8d2fe455-919b-11e8-bd99-0a3f336079ae_3
8d2f9e91663a        k8s.gcr.io/pause-amd64:3.1            "/pause"                 5 hours ago         Up 5 hours                              k8s_POD_storage-provisioner_kube-system_b6aff20a-8a73-11e8-9698-0a3f336079ae_15
653af0e9271d        k8s.gcr.io/pause-amd64:3.1            "/pause"                 5 hours ago         Up 5 hours                              k8s_POD_kube-dns-86f4d74b45-v9vmk_kube-system_a1e5cb43-8a66-11e8-9525-0a3f336079ae_16
df308f8bdb5f        k8s.gcr.io/pause-amd64:3.1            "/pause"                 5 hours ago         Up 5 hours                              k8s_POD_kubernetes-dashboard-5498ccf677-7g556_kube-system_9f72274b7-8a73-11e8-9698-0a3f336079ae_15
52e298376b2d        ad86dbed1555                          "kube-controller-man…"   5 hours ago         Up 5 hours                              k8s_kube-controller-manager_kube-controller-manager-minikube_kube-system_ecce8d95c7a4b43e140388e6dfdddcb6_5
bc6adedab442        af20925d51a3                          "kube-apiserver --ad…"   5 hours ago         Up 5 hours                              k8s_kube-apiserver_kube-apiserver-minikube_kube-system_88d02914db0d5a2c89b870607a602e8c_5
```

Exercise 4.2 Deploying webserver on to Kubernetes cluster

Scenario

Deploy Nginx on Kubernetes cluster

To do tasks:

In this exercise, you will be:

- Deploy Nginx on Kubernetes cluster
- kubectl, clusters, the control plane, namespaces, pods, services, replication controllers, and labels.

Note: Alternatively, you can deploy Tomcat also using specifications available in /kubews/TomcatWS

Steps

Deploying Nginx	

1	Create workspace “kubews” as shown below
	
	`[user1@ip-172-42-32-7 /]$ sudo mkdir kubews`
	`[user1@ip-172-42-32-7 /]$ ls`
	`bin boot data dev etc home kubews lib lib64 media mnt opt proc root run sbin srv sys tmp usr var`
	`[user1@ip-172-42-32-7 /]$`

2	Add configuration file to create new namespace, command is “sudo vi development-ns.yaml”.
	
	`[user1@ip-172-42-32-7 kubews]$ pwd`
	`/kubews`
	`[user1@ip-172-42-32-7 kubews]$ sudo vi development-ns.yaml`
	
	Add configurations as shown below
	
	`[user1@ip-172-42-32-7 kubews]$ sudo cat development-ns.yaml`
	`kind: "Namespace"`
	
	`apiVersion: "v1"`
	
	`metadata:`
	
	`  name: "development"`
	
	`  labels:`
	
	`    name: "development"`
	`[user1@ip-172-42-32-7 kubews]$`
	
	Create namespace “Development using command “sudo kubectl create -f development-ns.yaml” as shown below

```
[user1@ip-172-42-32-7 kubews]$ sudo kubectl create -f development-ns.yaml
namespace/development created
[user1@ip-172-42-32-7 kubews]$
```

Check newly created namespace as shown below

```
[user1@ip-172-42-32-7 kubews]$ sudo kubectl get namespaces
NAME          STATUS    AGE
default       Active    1d
development   Active    2m
kube-public   Active    1d
kube-system   Active    1d
[user1@ip-172-42-32-7 kubews]$
```

3

Add configuration file to create new pod, command is “sudo vi nginx.yaml”.

```
[user1@ip-172-42-32-7 kubews]$ pwd
/kubews
[user1@ip-172-42-32-7 kubews]$ sudo vi nginx.yaml
```

Add configurations as shown below

```
[user1@ip-172-42-32-7 kubews]$ sudo cat nginx.yaml
apiVersion: v1

kind: Pod

metadata:

  name: nginx

  labels:

    name: "development"

spec:

  containers:

  - name: nginx-server

    image: nginx

    ports:

    - containerPort: 80
[user1@ip-172-42-32-7 kubews]$
```

Create pod nginx using command “sudo kubectl create -f nginx.yaml” as shown below

```
[user1@ip-172-42-32-7 kubews]$ sudo kubectl create -f nginx.yaml
pod/nginx created
[user1@ip-172-42-32-7 kubews]$
```

Check newly created pod as shown below

```
[user1@ip-172-42-32-7 kubews]$ sudo kubectl get pods
NAME                                   READY     STATUS    RESTARTS   AGE
kubernetes-bootcamp-7799cbcb86-mjtxx   1/1       Running   0          3h
kubernetes-bootcamp-7799cbcb86-pgnfn   1/1       Running   0          3h
nginx                                  1/1       Running   0          1m
[user1@ip-172-42-32-7 kubews]$
```

```
[user1@ip-172-42-32-7 kubews]$ sudo kubectl describe pod/nginx
Name:         nginx
Namespace:    default
Node:         minikube/172.42.32.7
Start Time:   Thu, 19 Jul 2018 12:49:18 +0000
Labels:       name=development
Annotations:  <none>
Status:       Running
IP:           172.17.0.3
Containers:
  nginx-server:
    Container ID:   docker://3819f183c0e1519b232eaf4e2f7a3b715c32edb607c5adbea26f31431bfa123a
    Image:          nginx
    Image ID:       docker-pullable://nginx@sha256:4a5573037f358b6cdfa2f3e8a9c33a5cf11bcd1675ca72ca76fbe5bd77d0d682
    Port:           80/TCP
    Host Port:      0/TCP
    State:          Running
      Started:      Thu, 19 Jul 2018 12:49:25 +0000
    Ready:          True
    Restart Count:  0
    Environment:    <none>
    Mounts:
      /var/run/secrets/kubernetes.io/serviceaccount from default-token-d4bll (ro)
Conditions:
  Type           Status
  Initialized    True
  Ready          True
  PodScheduled   True
Volumes:
  default-token-d4bll:
    Type:        Secret (a volume populated by a Secret)
    SecretName:  default-token-d4bll
    Optional:    false
QoS Class:       BestEffort
Node-Selectors:  <none>
Tolerations:     node.kubernetes.io/not-ready:NoExecute for 300s
                 node.kubernetes.io/unreachable:NoExecute for 300s
Events:
  Type    Reason                 Age   From               Message
  ----    ------                 ----  ----               -------
  Normal  Scheduled              4m    default-scheduler  Successfully assigned nginx to minikube
  Normal  SuccessfulMountVolume  4m    kubelet, minikube  MountVolume.SetUp succeeded for volume "default-token-d4bll"
  Normal  Pulling                4m    kubelet, minikube  pulling image "nginx"
  Normal  Pulled                 4m    kubelet, minikube  Successfully pulled image "nginx"
  Normal  Created                4m    kubelet, minikube  Created container
  Normal  Started                4m    kubelet, minikube  Started container
[user1@ip-172-42-32-7 kubews]$
```

Note: you can delete this pod using command “sudo kubectl delete pod nginx”. In the next step, we will create pod for Nginx using replication controller

4

Add configuration file to create new replication controller, command is “sudo vi nginxrc.yaml”.

```
[user1@ip-172-42-32-7 kubews]$ pwd
/kubews
[user1@ip-172-42-32-7 kubews]$ sudo vi nginxrc.yaml
[user1@ip-172-42-32-7 kubews]$
```

Add configurations as shown below

```
[user1@ip-172-42-32-7 kubews]$ sudo cat nginxrc.yaml
apiVersion: v1

kind: ReplicationController

metadata:

  name: rcnginx

  labels:

    name: "development"

spec:

  replicas: 1

  template:

    metadata:

      labels:

        app: nginx

    spec:

      containers:

      - name: nginx

        image: nginx

        ports:

        - containerPort: 80
```

Create replication controller nginx using command “sudo kubectl create -f nginx.yaml” as shown below

```
[user1@ip-172-42-32-7 kubews]$ sudo kubectl create -f nginxrc.yaml
replicationcontroller/rcnginx created
[user1@ip-172-42-32-7 kubews]$
```

Check newly created replication control “rcnginx” as shown below

```
[user1@ip-172-42-32-7 kubews]$ sudo kubectl get rc
NAME      DESIRED   CURRENT   READY     AGE
rcnginx   1         1         1         59s
[user1@ip-172-42-32-7 kubews]$
```

```
[user1@ip-172-42-32-7 kubews]$ sudo kubectl get pods
NAME                                   READY     STATUS    RESTARTS   AGE
kubernetes-bootcamp-7799cbcb86-mjtxx   1/1       Running   0          4h
kubernetes-bootcamp-7799cbcb86-pgnfn   1/1       Running   0          4h
rcnginx-nlj7j                          1/1       Running   0          18m
[user1@ip-172-42-32-7 kubews]$
```

```
[user1@ip-172-42-32-7 kubews]$ sudo kubectl describe rc
Name:         rcnginx
Namespace:    default
Selector:     app=nginx
Labels:       name=development
Annotations:  <none>
Replicas:     1 current / 1 desired
Pods Status:  1 Running / 0 Waiting / 0 Succeeded / 0 Failed
Pod Template:
  Labels:  app=nginx
  Containers:
   nginx:
    Image:        nginx
    Port:         80/TCP
    Host Port:    0/TCP
    Environment:  <none>
    Mounts:       <none>
  Volumes:        <none>
Events:
  Type    Reason            Age   From                    Message
  ----    ------            ----  ----                    -------
  Normal  SuccessfulCreate  3m    replication-controller  Created pod: rcnginx-nlj7j
[user1@ip-172-42-32-7 kubews]$
```

5

Scale out pods as shown below using command “sudo kubectl scale --replicas=4 rc rcnginx”

```
[user1@ip-172-42-32-7 kubews]$ sudo kubectl scale --replicas=4 rc rcnginx
replicationcontroller/rcnginx scaled
[user1@ip-172-42-32-7 kubews]$ sudo kubectl get pods
NAME                                   READY     STATUS              RESTARTS   AGE
kubernetes-bootcamp-7799cbcb86-mjtxx   1/1       Running             0          4h
kubernetes-bootcamp-7799cbcb86-pgnfn   1/1       Running             0          4h
rcnginx-gmr4b                          0/1       ContainerCreating   0          3s
rcnginx-nlj7j                          1/1       Running             0          20m
rcnginx-pbvbc                          0/1       ContainerCreating   0          3s
rcnginx-vrl58                          0/1       ContainerCreating   0          3s
[user1@ip-172-42-32-7 kubews]$
```

Little later, run get pods commands to see the updated status of pods

```
[user1@ip-172-42-32-7 kubews]$ sudo kubectl get pods
NAME                                   READY     STATUS    RESTARTS   AGE
kubernetes-bootcamp-7799cbcb86-mjtxx   1/1       Running   0          4h
kubernetes-bootcamp-7799cbcb86-pgnfn   1/1       Running   0          4h
rcnginx-gmr4b                          1/1       Running   0          2m
rcnginx-nlj7j                          1/1       Running   0          22m
rcnginx-pbvbc                          1/1       Running   0          2m
rcnginx-vrl58                          1/1       Running   0          2m
[user1@ip-172-42-32-7 kubews]$
```

```
[user1@ip-172-42-32-7 kubews]$ sudo kubectl get rc
NAME      DESIRED   CURRENT   READY     AGE
rcnginx   4         4         4         25m
[user1@ip-172-42-32-7 kubews]$
```

6

Add configuration file to create new service “nginxsvc” , command is “sudo vi nginxsvc.yaml”.

```
[user1@ip-172-42-32-7 kubews]$ pwd
/kubews
[user1@ip-172-42-32-7 kubews]$ sudo vi nginxsvc.yaml
```

Add configurations as shown below

```
[user1@ip-172-42-32-7 kubews]$ sudo cat nginxsvc.yaml
apiVersion: v1

kind: Service

metadata:

  name: nginxsvc

  labels:

    app: nginx

spec:

  ports:

  - port: 80

    protocol: TCP

  selector:

    app: nginx
[user1@ip-172-42-32-7 kubews]$
```

Create service nginxsvc using command "sudo kubectl create -f nginxsvc.yaml" as shown below

```
[user1@ip-172-42-32-7 kubews]$ sudo kubectl create -f nginxsvc.yaml
service/nginxsvc created
```

Check newly created service as shown below

```
[user1@ip-172-42-32-7 kubews]$ sudo kubectl get service
NAME                TYPE        CLUSTER-IP      EXTERNAL-IP   PORT(S)          AGE
kubernetes          ClusterIP   10.96.0.1       <none>        443/TCP          1d
kubernetes-bootcamp NodePort    10.99.186.215   <none>        8080:31216/TCP   8h
nginxsvc            ClusterIP   10.106.244.82   <none>        80/TCP           12s
[user1@ip-172-42-32-7 kubews]$
```

```
[user1@ip-172-42-32-7 kubews]$ sudo kubectl get service nginxsvc
NAME       TYPE        CLUSTER-IP      EXTERNAL-IP   PORT(S)   AGE
nginxsvc   ClusterIP   10.106.244.82   <none>        80/TCP    3m
[user1@ip-172-42-32-7 kubews]$
```

7	Because every node in a Kubernetes cluster runs a kube-proxy, the kube-proxy watches the Kubernetes API server for the addition and removal of services. For each new service, kube-proxy opens a (randomly chosen) port on the local node. Any connections made to that port are proxied to one of the corresponding backend pods. Go to browser and check Nginx homepage as shown below 
------- End of Exercise -------	

Exercise 4.3 Deploying multi-tier app on to Kubernetes cluster

Scenario

Deploy multi-tier web app on Kubernetes cluster

To do tasks:

In this exercise, you will be:

- Build and deploy a simple, multi-tier web application using Kubernetes and Docker

This example consists of the following components:

1. A single-instance Redis master to store guestbook entries
2. Multiple replicated Redis instances to serve reads
3. Multiple web frontend instances

The guestbook application uses Redis to store its data. It writes its data to a Redis master instance and reads data from multiple Redis slave instances.

The guestbook application has a web frontend serving the HTTP requests written in PHP. It is configured to connect to the redis-master Service for write requests and the redis-slave service for Read requests.

Steps

Deploying multi-tier web app	
1	Note: delete previous exercise related objects from cluster before starting this exercise Create workspace “/kubews/GuestbookAppWS” as shown below

```
[user1@ip-172-42-32-7 GuestbookAppWS]$ pwd
/kubews/GuestbookAppWS
[user1@ip-172-42-32-7 GuestbookAppWS]$
```

2

Create configuration file “redis-master-deployment.yaml” as shown below, this is to specifies a Deployment controller that runs a single replica Redis master Pod

```
[user1@ip-172-42-32-7 GuestbookAppWS]$ sudo cat redis-master-deployment.yaml
apiVersion: apps/v1 # for versions before 1.9.0 use apps/v1beta2

kind: Deployment

metadata:

  name: redis-master

  labels:

    app: redis

spec:

  selector:

    matchLabels:

      app: redis

      role: master

      tier: backend

  replicas: 1

  template:

    metadata:

      labels:

        app: redis

        role: master

        tier: backend

    spec:

      containers:

      - name: master
```

```
        image: k8s.gcr.io/redis:e2e  # or just image: redis
        resources:
          requests:
            cpu: 100m
            memory: 100Mi
        ports:
        - containerPort: 6379
```

3

Apply the Redis Master Deployment from file using command "sudo kubectl create -f redis-master-deployment.yaml"

```
[user1@ip-172-42-32-7 GuestbookAppWS]$ sudo kubectl create -f redis-master-deployment.yaml
deployment.apps/redis-master created
[user1@ip-172-42-32-7 GuestbookAppWS]$
```

You can check newly created pod as shown below.

```
[user1@ip-172-42-32-7 GuestbookAppWS]$ sudo kubectl get pods
NAME                                READY     STATUS    RESTARTS   AGE
kubernetes-bootcamp-7799cbcb86-mjtxx   1/1       Running   1          1d
kubernetes-bootcamp-7799cbcb86-pgnfn   1/1       Running   1          1d
rcnginx-gmr4b                       1/1       Running   1          21h
rcnginx-nlj7j                       1/1       Running   1          21h
rcnginx-pbvbc                       1/1       Running   1          21h
rcnginx-vrl58                       1/1       Running   1          21h
redis-master-55db5f7567-npp5w       1/1       Running   0          5m
[user1@ip-172-42-32-7 GuestbookAppWS]$
```

Run the following command to view the logs from the Redis Master Pod

```
[user1@ip-172-42-32-7 GuestbookAppWS]$ sudo kubectl logs -f redis-master-55db5f7567-npp5w
                _._
           _.-``__ ''-._
      _.-``    `.  `_.  ''-._           Redis 2.8.19 (00000000/0) 64 bit
  .-`` .-```.  ```\/    _.,_ ''-._
 (    '      ,       .-`  | `,    )     Running in stand alone mode
 |`-._`-...-` __...-.``-._|'` _.-'|     Port: 6379
 |    `-._   `._    /     _.-'    |     PID: 1
  `-._    `-._  `-./  _.-'    _.-'
 |`-._`-._    `-.__.-'    _.-'_.-'|
 |    `-._`-._        _.-'_.-'    |           http://redis.io
  `-._    `-._`-.__.-'_.-'    _.-'
 |`-._`-._    `-.__.-'    _.-'_.-'|
 |    `-._`-._        _.-'_.-'    |
  `-._    `-._`-.__.-'_.-'    _.-'
      `-._    `-.__.-'    _.-'
          `-._        _.-'
              `-.__.-'

[1] 20 Jul 10:45:24.924 # Server started, Redis version 2.8.19
[1] 20 Jul 10:45:24.924 # WARNING you have Transparent Huge Pages (THP) support enabled in your kernel. This will create latency and memory usage is
es with Redis. To fix this issue run the command 'echo never > /sys/kernel/mm/transparent_hugepage/enabled' as root, and add it to your /etc/rc.loca
in order to retain the setting after a reboot. Redis must be restarted after THP is disabled.
[1] 20 Jul 10:45:24.924 # WARNING: The TCP backlog setting of 511 cannot be enforced because /proc/sys/net/core/somaxconn is set to the lower value
 128.
[1] 20 Jul 10:45:24.924 * The server is now ready to accept connections on port 6379
```

4

Create configuration file “redis-master-service.yaml” as shown below, this is to apply a Service to proxy the traffic to the Redis master Pod.

```
[user1@ip-172-42-32-7 GuestbookAppWS]$ sudo cat redis-master-service.yaml
apiVersion: v1

kind: Service

metadata:

  name: redis-master

  labels:

    app: redis

    role: master

    tier: backend

spec:

  ports:

  - port: 6379

    targetPort: 6379

  selector:

    app: redis

    role: master

    tier: backend
```

Apply the Redis Master Service from the above redis-master-service.yaml file, command is “sudo kubectl create -f redis-master-service.yaml”

```
[user1@ip-172-42-32-7 GuestbookAppWS]$ sudo kubectl create -f redis-master-service.yaml
service/redis-master created
[user1@ip-172-42-32-7 GuestbookAppWS]$
```

Check newly created services as shown below

```
[user1@ip-172-42-32-7 GuestbookAppWS]$ sudo kubectl get services
NAME                  TYPE        CLUSTER-IP      EXTERNAL-IP   PORT(S)          AGE
kubernetes            ClusterIP   10.96.0.1       <none>        443/TCP          2d
kubernetes-bootcamp   NodePort    10.99.186.215   <none>        8080:31216/TCP   1d
nginxsvc              ClusterIP   10.106.244.82   <none>        80/TCP           22h
redis-master          ClusterIP   10.109.178.54   <none>        6379/TCP         1m
[user1@ip-172-42-32-7 GuestbookAppWS]$
```

You can run describe command on newly created service

Note: This manifest file creates a Service named redis-master with a set of labels that match the labels previously defined, so the Service routes network traffic to the Redis master Pod.

5

Create configuration file “redis-slave-deployment.yaml” as shown below, this is to create Redis slaves

```
[user1@ip-172-42-32-7 GuestbookAppWS]$ sudo cat redis-slave-deployment.yaml
apiVersion: apps/v1 # for versions before 1.9.0 use apps/v1beta2

kind: Deployment

metadata:

  name: redis-slave

  labels:

    app: redis

spec:

  selector:

    matchLabels:

      app: redis

      role: slave

      tier: backend

  replicas: 2

  template:

    metadata:

      labels:
```

```
      labels:
        app: redis
        role: slave
        tier: backend
    spec:
      containers:
      - name: slave
        image: gcr.io/google_samples/gb-redisslave:v1
        resources:
          requests:
            cpu: 100m
            memory: 100Mi
        env:
        - name: GET_HOSTS_FROM
          value: env
        ports:
        - containerPort: 6379
```

Create the Redis Slave Deployment from the file, command is "sudo kubectl create -f redis-slave-deployment.yaml"

```
[user1@ip-172-42-32-7 GuestbookAppWS]$ sudo  kubectl create -f redis-slave-deployment.yaml
deployment.apps/redis-slave created
[user1@ip-172-42-32-7 GuestbookAppWS]$
```

Check newly created Redis slave pods as shown below

```
[user1@ip-172-42-32-7 GuestbookAppWS]$ sudo kubectl get pods
NAME                                READY     STATUS    RESTARTS   AGE
kubernetes-bootcamp-7799cbcb86-mjtxx   1/1       Running   1          1d
kubernetes-bootcamp-7799cbcb86-pgnfn   1/1       Running   1          1d
rcnginx-gmr4b                       1/1       Running   1          22h
rcnginx-nlj7j                       1/1       Running   1          23h
rcnginx-pbvbc                       1/1       Running   1          22h
rcnginx-vrl58                       1/1       Running   1          22h
redis-master-55db5f7567-npp5w       1/1       Running   0          1h
redis-slave-584c66c5b5-9vkmz        1/1       Running   0          2m
redis-slave-584c66c5b5-j69v5        1/1       Running   0          2m
[user1@ip-172-42-32-7 GuestbookAppWS]$
```

6

Create configuration file “redis-slave-service.yaml” as shown below, this is to create Redis slave service.

```
[user1@ip-172-42-32-7 GuestbookAppWS]$ sudo cat redis-slave-service.yaml
apiVersion: v1

kind: Service

metadata:

  name: redis-slave

  labels:

    app: redis

    role: slave

    tier: backend

spec:

  ports:

  - port: 6379

  selector:

    app: redis

    role: slave

    tier: backend
[user1@ip-172-42-32-7 GuestbookAppWS]$
```

Create the Redis Slave Service from the above redis-slave-service.yaml file command is "sudo kubectl create -f redis-slave-service.yaml"

```
[user1@ip-172-42-32-7 GuestbookAppWS]$ sudo kubectl create -f redis-slave-service.yaml
service/redis-slave created
[user1@ip-172-42-32-7 GuestbookAppWS]$
```

Check the newly created service as shown below

```
[user1@ip-172-42-32-7 GuestbookAppWS]$ sudo kubectl get service
NAME                  TYPE        CLUSTER-IP      EXTERNAL-IP   PORT(S)          AGE
kubernetes            ClusterIP   10.96.0.1       <none>        443/TCP          2d
kubernetes-bootcamp   NodePort    10.99.186.215   <none>        8080:31216/TCP   1d
nginxsvc              ClusterIP   10.106.244.82   <none>        80/TCP           22h
redis-master          ClusterIP   10.109.178.54   <none>        6379/TCP         38m
redis-slave           ClusterIP   10.111.101.87   <none>        6379/TCP         2m
[user1@ip-172-42-32-7 GuestbookAppWS]$
```

7	Create configuration file “frontend-deployment.yaml” as shown below, this is to create Guest book app frontend.
	
	[user1@ip-172-42-32-7 GuestbookAppWS]$ sudo cat frontend-deployment.yaml
	apiVersion: apps/v1 # for versions before 1.9.0 use apps/v1beta2
	
	kind: Deployment
	
	metadata:
	
	  name: frontend
	
	  labels:
	
	  app: guestbook
	
	spec:
	
	  selector:
	
	  matchLabels:
	
	  app: guestbook
	
	  tier: frontend
	
	  replicas: 3
	
	  template:
	
	  metadata:
	
	  labels:
	
	  app: guestbook

```
        tier: frontend
    spec:
      containers:
      - name: php-redis
        image: gcr.io/google-samples/gb-frontend:v4
        resources:
          requests:
            cpu: 100m
            memory: 100Mi
        env:
        - name: GET_HOSTS_FROM
          value: env
        ports:
        - containerPort: 80
[user1@ip-172-42-32-7 GuestbookAppWS]$
```

Apply the frontend Deployment from the frontend-deployment.yaml file, command is "sudo kubectl create -f frontend-deployment.yaml"

```
[user1@ip-172-42-32-7 GuestbookAppWS]$ sudo kubectl create -f frontend-deployment.yaml
deployment.apps/frontend created
[user1@ip-172-42-32-7 GuestbookAppWS]$
```

Check newly created pods as shown below

```
[user1@ip-172-42-32-7 GuestbookAppWS]$ sudo kubectl get pods
NAME                                   READY     STATUS    RESTARTS   AGE
frontend-6d797fc7dd-2q27s              1/1       Running   0          2m
frontend-6d797fc7dd-6r2vw              1/1       Running   0          2m
frontend-6d797fc7dd-ph5ln              1/1       Running   0          2m
kubernetes-bootcamp-7799cbcb86-mjtxx   1/1       Running   1          1d
kubernetes-bootcamp-7799cbcb86-pgnfn   1/1       Running   1          1d
rcnginx-gmr4b                          1/1       Running   1          23h
rcnginx-nlj7j                          1/1       Running   1          23h
rcnginx-pbvbc                          1/1       Running   1          23h
rcnginx-vrl58                          1/1       Running   1          23h
redis-master-55db5f7567-npp5w          1/1       Running   0          2h
redis-slave-584c66c5b5-9vkmz           1/1       Running   0          40m
redis-slave-584c66c5b5-j69v5           1/1       Running   0          40m
[user1@ip-172-42-32-7 GuestbookAppWS]$
```

```
[user1@ip-172-42-32-7 GuestbookAppWS]$ sudo kubectl get pods -l app=guestbook -l tier=frontend
NAME                       READY     STATUS    RESTARTS   AGE
frontend-6d797fc7dd-2q27s  1/1       Running   0          4m
frontend-6d797fc7dd-6r2vw  1/1       Running   0          4m
frontend-6d797fc7dd-ph5ln  1/1       Running   0          4m
[user1@ip-172-42-32-7 GuestbookAppWS]$
```

Note: The redis-slave and redis-master Services you applied are only accessible within the container cluster because the default type for a Service is ClusterIP. ClusterIP provides a single IP address for the set of Pods the Service is pointing to. This IP address is accessible only within the cluster.

8

Note: Minikube can only expose Services through NodePort.

Create configuration file “frontend-service.yaml” as shown below, this is to create Guest book app frontend service .

```
[user1@ip-172-42-32-7 GuestbookAppWS]$ sudo cat frontend-service.yaml
apiVersion: v1

kind: Service

metadata:

  name: frontend

  labels:

    app: guestbook

    tier: frontend

spec:

  # comment or delete the following line if you want to use a LoadBalancer

  type: NodePort

  # if your cluster supports it, uncomment the following to automatically create

  # an external load-balanced IP for the frontend service.

  # type: LoadBalancer

  ports:

  - port: 80

  selector:


    app: guestbook

    tier: frontend
```

Apply the frontend Service from the frontend-service.yaml file, command is "sudo kubectl create -f frontend-service.yaml"

```
[user1@ip-172-42-32-7 GuestbookAppWS]$ sudo kubectl create -f frontend-service.yaml
service/frontend created
[user1@ip-172-42-32-7 GuestbookAppWS]$
```

Check newly created front end service as shown below

```
[user1@ip-172-42-32-7 GuestbookAppWS]$ sudo kubectl get service
NAME                  TYPE        CLUSTER-IP      EXTERNAL-IP   PORT(S)          AGE
frontend              NodePort    10.97.192.180   <none>        8018:30840/TCP   2m
kubernetes            ClusterIP   10.96.0.1       <none>        443/TCP          2d
kubernetes-bootcamp   NodePort    10.99.186.215   <none>        8080:31216/TCP   1d
nginxsvc              ClusterIP   10.106.244.82   <none>        80/TCP           23h
redis-master          ClusterIP   10.109.178.54   <none>        6379/TCP         1h
redis-slave           ClusterIP   10.111.101.87   <none>        6379/TCP         41m
[user1@ip-172-42-32-7 GuestbookAppWS]$
```

Suppose your guestbook app has been running for a while, and it gets a sudden burst of publicity. You decide it would be a good idea to add more web servers to your frontend. You can do this easily since your servers are defined as a service that uses a Deployment controller.

9

Check App status in cluster as shown below

```
[user1@ip-172-42-32-7 GuestbookAppWS]$ sudo kubectl get po,svc -o wide
NAME                                   READY     STATUS    RESTARTS   AGE       IP           NODE
pod/frontend-6dd96bb6f7-9stb4          1/1       Running   0          5m        172.17.0.9   minikube
pod/frontend-6dd96bb6f7-ktfxx          1/1       Running   0          5m        172.17.0.7   minikube
pod/frontend-6dd96bb6f7-sm4fv          1/1       Running   0          5m        172.17.0.8   minikube
pod/redis-master-55db5f7567-2j9gp      1/1       Running   1          29m       172.17.0.4   minikube
pod/redis-slave-6894998d77-4knrb       1/1       Running   0          12m       172.17.0.5   minikube
pod/redis-slave-6894998d77-wnpjt       1/1       Running   0          12m       172.17.0.6   minikube

NAME                    TYPE        CLUSTER-IP       EXTERNAL-IP   PORT(S)        AGE       SELECTOR
service/frontend        NodePort    10.108.169.43    <none>        80:32326/TCP   1m        app=guestbook,tier=frontend
service/kubernetes      ClusterIP   10.96.0.1        <none>        443/TCP        5d        <none>
service/redis-master    ClusterIP   10.107.157.196   <none>        6379/TCP       28m       app=redis,role=master,tier=backend
service/redis-slave     ClusterIP   10.108.45.211    <none>        6379/TCP       8m        app=redis,role=slave,tier=backend
[user1@ip-172-42-32-7 GuestbookAppWS]$
```

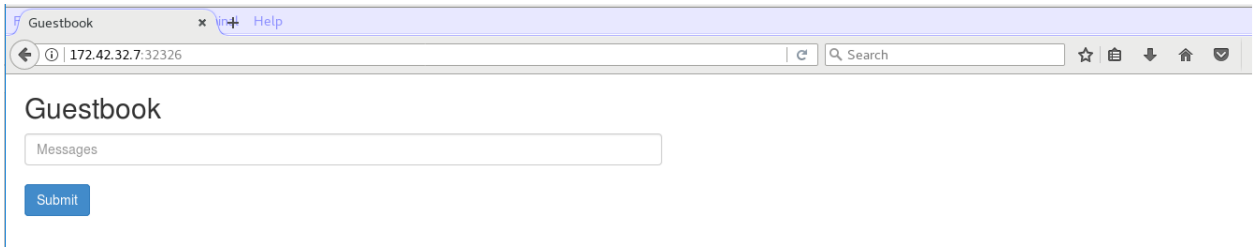
10

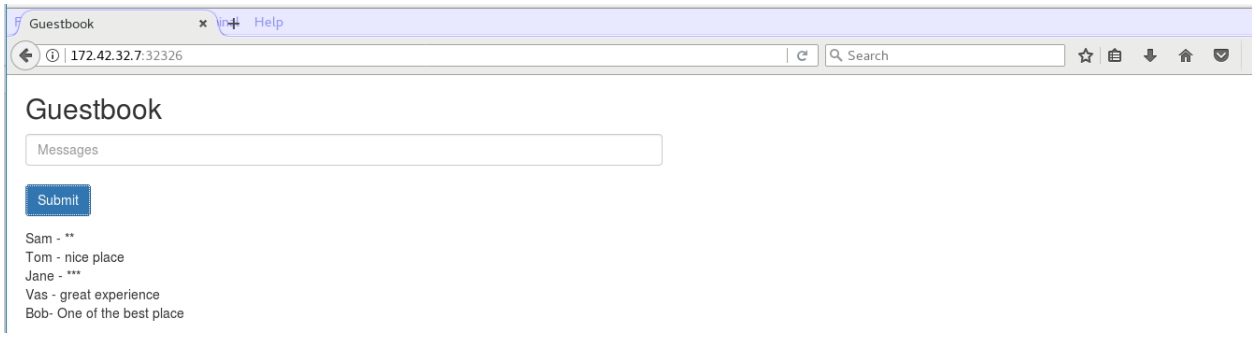
Run command “sudo /usr/local/bin/minikube service frontend --url” to get App frontend service URL

```
[user1@ip-172-42-32-7 GuestbookAppWS]$ sudo /usr/local/bin/minikube service frontend --url
http://172.42.32.7:32326
[user1@ip-172-42-32-7 GuestbookAppWS]$
```

12

Enter URL in the web browser, access Guest Book App hone page



	Enter messages and submit as shown below 

13	We can use kubectl exec to get a shell to a running Container
	
	Get the Redis master pod name, run command "sudo kubectl exec -ti redis-master-55db5f7567-2j9gp bash" as shown below (running) to query Redis guest book database.
	
	`[user1@ip-172-42-32-7 GuestbookAppWS]$ sudo kubectl exec -ti redis-master-55db5f7567-2j9gp bash`
	`[ root@redis-master-55db5f7567-2j9gp:/data ]$ date`
	`Mon Jul 23 13:04:59 UTC 2018`
	`[ root@redis-master-55db5f7567-2j9gp:/data ]$`
	
	Query database as shown below
	
	`[ root@redis-master-55db5f7567-2j9gp:/data ]$ redis-cli KEYS "*"`
	`1) "messages"`
	`[ root@redis-master-55db5f7567-2j9gp:/data ]$ redis-cli KEYS "messages"`
	`1) "messages"`
	`[ root@redis-master-55db5f7567-2j9gp:/data ]$ redis-cli MGET "messages"`
	`1) ",Sam - **,Tom - nice place,Jane - ***,Vas - great experience,Bob- One of the best place"`
	`[ root@redis-master-55db5f7567-2j9gp:/data ]$`

14	Suppose your guestbook app has been running for a while, and it gets a sudden burst of publicity. You decide it would be a good idea to add more web servers to your frontend. You can do this easily since your servers are defined as a service that uses a Deployment controller. Run the command "kubectl scale deployment frontend --replicas=5 " to scale up the number of frontend Pods

```
[user1@ip-172-42-32-7 GuestbookAppWS]$ sudo kubectl get pods
NAME                              READY     STATUS    RESTARTS   AGE
frontend-6dd96bb6f7-9stb4         1/1       Running   0          35m
frontend-6dd96bb6f7-ktfxx         1/1       Running   0          35m
frontend-6dd96bb6f7-sm4fv         1/1       Running   0          35m
redis-master-55db5f7567-2j9gp     1/1       Running   1          59m
redis-slave-6894998d77-4knrb      1/1       Running   0          42m
redis-slave-6894998d77-wnpjt      1/1       Running   0          42m
[user1@ip-172-42-32-7 GuestbookAppWS]$ sudo kubectl scale deployment frontend --replicas=5
deployment.extensions/frontend scaled
[user1@ip-172-42-32-7 GuestbookAppWS]$ sudo kubectl get pods
NAME                              READY     STATUS    RESTARTS   AGE
frontend-6dd96bb6f7-87lwv         1/1       Running   0          3s
frontend-6dd96bb6f7-9stb4         1/1       Running   0          35m
frontend-6dd96bb6f7-ktfxx         1/1       Running   0          35m
frontend-6dd96bb6f7-s8r5b         1/1       Running   0          3s
frontend-6dd96bb6f7-sm4fv         1/1       Running   0          35m
redis-master-55db5f7567-2j9gp     1/1       Running   1          59m
redis-slave-6894998d77-4knrb      1/1       Running   0          42m
redis-slave-6894998d77-wnpjt      1/1       Running   0          42m
[user1@ip-172-42-32-7 GuestbookAppWS]$
```

15

Run the command “kubectl scale deployment frontend --replicas=2” to scale down the number of frontend Pods

```
[user1@ip-172-42-32-7 GuestbookAppWS]$ sudo kubectl get pods
NAME                              READY     STATUS    RESTARTS   AGE
frontend-6dd96bb6f7-87lwv         1/1       Running   0          2m
frontend-6dd96bb6f7-9stb4         1/1       Running   0          38m
frontend-6dd96bb6f7-ktfxx         1/1       Running   0          38m
frontend-6dd96bb6f7-s8r5b         1/1       Running   0          2m
frontend-6dd96bb6f7-sm4fv         1/1       Running   0          38m
redis-master-55db5f7567-2j9gp     1/1       Running   1          1h
redis-slave-6894998d77-4knrb      1/1       Running   0          45m
redis-slave-6894998d77-wnpjt      1/1       Running   0          45m
[user1@ip-172-42-32-7 GuestbookAppWS]$ sudo kubectl scale deployment frontend --replicas=2
deployment.extensions/frontend scaled
[user1@ip-172-42-32-7 GuestbookAppWS]$ sudo kubectl get pods
NAME                              READY     STATUS        RESTARTS   AGE
frontend-6dd96bb6f7-9stb4         1/1       Running       0          38m
frontend-6dd96bb6f7-s8r5b         0/1       Terminating   0          2m
frontend-6dd96bb6f7-sm4fv         1/1       Running       0          38m
redis-master-55db5f7567-2j9gp     1/1       Running       1          1h
redis-slave-6894998d77-4knrb      1/1       Running       0          45m
redis-slave-6894998d77-wnpjt      1/1       Running       0          45m
[user1@ip-172-42-32-7 GuestbookAppWS]$ sudo kubectl get pods
NAME                              READY     STATUS    RESTARTS   AGE
frontend-6dd96bb6f7-9stb4         1/1       Running   0          38m
frontend-6dd96bb6f7-sm4fv         1/1       Running   0          38m
redis-master-55db5f7567-2j9gp     1/1       Running   1          1h
redis-slave-6894998d77-4knrb      1/1       Running   0          45m
redis-slave-6894998d77-wnpjt      1/1       Running   0          45m
[user1@ip-172-42-32-7 GuestbookAppWS]$
```

16

You can use below mentioned commands to remove Guest book app from cluster

	1. kubectl delete deployment -l app=redis 2. kubectl delete service -l app=redis 3. kubectl delete deployment -l app=guestbook 4. kubectl delete service -l app=guestbook
------- End of Exercise -------	

Exercise 4.4 Deploying MySQL onto Kubernetes cluster with persistent volumes (optional)

Scenario

Deploy MySQL database to Kubernetes cluster

To do tasks:

In this exercise, you will be:

- Create Persistent Volume Claims and Persistent Volumes
- Create a Secret
- Deploy MySQL

Steps

Deploy MySQL database Kubernetes single node cluster	
1	Create workspace as shown below [user1@ip-172-42-32-7 MySQLws]$ pwd /kubews/MySQLws [user1@ip-172-42-32-7 MySQLws]$
2	Create and update “mysql-pv.yaml” configuration file as shown below, to create persistent volume and clams

```
kind: PersistentVolume
apiVersion: v1
metadata:
  name: mysql-pv-volume
  labels:
    type: local
spec:
  storageClassName: manual
  capacity:
    storage: 8Gi
  accessModes:
    - ReadWriteOnce
  hostPath:
    path: "/mnt/data"
---
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: mysql-pv-claim
spec:
  storageClassName: manual
  accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: 8Gi
```

3	Create and update “mysql-deployment.yaml” configuration file as shown below, to deploy MySQL

```
apiVersion: v1
kind: Service
metadata:
  name: mysql
spec:
  ports:
  - port: 3306
  selector:
    app: mysql
  clusterIP: None
---
apiVersion: apps/v1 # for versions before 1.9.0 use apps/v1beta2
kind: Deployment
metadata:
  name: mysql
spec:
  selector:
    matchLabels:
      app: mysql
  strategy:
    type: Recreate
  template:
    metadata:
      labels:
        app: mysql
    spec:
      containers:
      - image: mysql:5.6
        name: mysql
        env:
          # Use secret in real usage
        - name: MYSQL_ROOT_PASSWORD
          valueFrom:
            secretKeyRef:
              name: mysql-pass
              key: password
        ports:
        - containerPort: 3306
          name: mysql
        volumeMounts:
        - name: mysql-persistent-storage
          mountPath: /var/lib/mysql
      volumes:
      - name: mysql-persistent-storage
        persistentVolumeClaim:
          claimName: mysql-pv-claim
```

4

Create secret to store MySQL password. Command is “sudo kubectl create secret generic mysql-pass --from-literal=password=password”

```
[user1@ip-172-42-32-7 MySQLws]$ sudo kubectl create secret generic mysql-pass --from-literal=password=password
secret/mysql-pass created
[user1@ip-172-42-32-7 MySQLws]$
```

5

Create PV and PVC using command “sudo kubectl create -f mysql-pv.yaml”

```
[user1@ip-172-42-32-7 MySQLws]$ sudo kubectl create -f mysql-pv.yaml
persistentvolume/mysql-pv-volume created
persistentvolumeclaim/mysql-pv-claim created
[user1@ip-172-42-32-7 MySQLws]$
```

6 Create MySQL pod and service using command "sudo kubectl create -f mysql-deployment.yaml"

```
[user1@ip-172-42-32-7 MySQLws]$ sudo kubectl create -f mysql-deployment.yaml
service/mysql created
deployment.apps/mysql created
[user1@ip-172-42-32-7 MySQLws]$
```

7 Get Pod, service, PV and PVC and secrets details as shown below, command is "sudo kubectl get po,svc,pv,pvc,secrets -o wide"

```
[user1@ip-172-42-32-7 MySQLws]$ sudo kubectl get po,svc,pv,pvc,secrets -o wide
NAME                         READY     STATUS    RESTARTS   AGE       IP           NODE
pod/mysql-6bfc69bb74-qbpq4   1/1       Running   0          2m        172.17.0.4   minikube

NAME                 TYPE        CLUSTER-IP   EXTERNAL-IP   PORT(S)    AGE       SELECTOR
service/kubernetes   ClusterIP   10.96.0.1    <none>        443/TCP    9d        <none>
service/mysql        ClusterIP   None         <none>        3306/TCP   2m        app=mysql

NAME                                CAPACITY   ACCESS MODES   RECLAIM POLICY   STATUS    CLAIM                    STORAGECLASS   REASON    AGE
persistentvolume/mysql-pv-volume    8Gi        RWO            Retain           Bound     default/mysql-pv-claim   manual                   3m

NAME                                     STATUS    VOLUME           CAPACITY   ACCESS MODES   STORAGECLASS   AGE
persistentvolumeclaim/mysql-pv-claim     Bound     mysql-pv-volume   8Gi        RWO            manual         3m

NAME                         TYPE                                  DATA      AGE
secret/default-token-d4bll   kubernetes.io/service-account-token   3         9d
secret/mysql-pass            Opaque                                1         5m
[user1@ip-172-42-32-7 MySQLws]$
```

8 We can use kubectl exec to get a shell to a running Container

Get the MySQL pod name, run command "sudo kubectl exec -ti mysql-6bfc69bb74-qbpq4 bash" as shown below (running) to access MySQL database.

```
[user1@ip-172-42-32-7 MySQLws]$ sudo kubectl get pods
NAME                     READY     STATUS    RESTARTS   AGE
mysql-6bfc69bb74-qbpq4   1/1       Running   0          4m
[user1@ip-172-42-32-7 MySQLws]$ sudo kubectl exec -ti mysql-6bfc69bb74-qbpq4  bash
root@mysql-6bfc69bb74-qbpq4:/#
```

9 After step 8, run below mentioned SQL commands to create database and store some data

1. mysql -u root -p (password is "password")
2. show databases;

	3. CREATE DATABASE pets; 4. USE pets; 5. CREATE TABLE pet (name VARCHAR(20), owner VARCHAR(20), species VARCHAR(20), sex CHAR(1), birth DATE, death DATE); 6. show tables; 7. describe pet;
10	Insert data into newly created pet table using below mentioned SQL commands INSERT INTO pet VALUES ('Puffball','Diane','hamster','f','1999-03-30',NULL); INSERT INTO pet VALUES ('Ulal','Diane','Fox','m','1999-03-30',NULL); INSERT INTO pet VALUES ('Lovely','Diane','Parrot','f','1999-03-30',NULL); INSERT INTO pet VALUES ('Jiss','Diane','Lab','f','1999-03-30',NULL); INSERT INTO pet VALUES ('James','Diane','Beagle','f','1999-03-30',NULL); INSERT INTO pet VALUES ('Snowey','Diane','Husky','f','1999-03-30',NULL); INSERT INTO pet VALUES ('Short','Diane','Spaniel','f','1999-03-30',NULL); INSERT INTO pet VALUES ('Chief','Diane','Sheppard','f','1999-03-30',NULL); Check table data using command “select 8 from pet”
11	Come out of MySQL shell using exit command, delete pods and other related Objects using below mentioned commands. 1. kubectl delete deployment,svc mysql 2. kubectl delete pvc mysql-pv-claim 3. kubectl delete pv mysql-pv-volume
12	Repeat steps from 5 to 8 and check database we created in step 10 and 11 existed or not in newly deployed MySQL instance. Because we used persistent volumes, database exists.
------- End of Exercise -------	

Exercise x.x Setup Kubernetes cluster using Kubeadm

Scenario

Kubernetes cluster setup

To do tasks:

In this exercise

- Kubernetes cluster setup using Kubeadm

Steps

Kubernetes cluster setup	
1	On master node:

	Install Docker using below steps 1] Go to https://download.docker.com/linux/centos/7/x86_64/stable/Packages/, download 2] cd /home/user1/Downloads 3] sudo yum install docker-ce-18.06.1.ce-3.el7.x86_64.rpm 4] cd .. 5] sudo systemctl start docker 6] sudo docker run hello-world 7] sudo docker images
2	On master node: Step 1: cd to directory yum.repos.d using command “cd /etc/yum.repos.d” Step 2: Create file kubernetes.repo using subl and add content to that file as shown below kubernetes.repo [kubernetes] name=Kubernetes baseurl=https://packages.cloud.google.com/yum/repos/kubernetes-el7-x86_64 enabled=1 gpgcheck=1 repo_gpgcheck=1 gpgkey=https://packages.cloud.google.com/yum/doc/yum-key.gpg https://packages.cloud.google.com/yum/doc/rpm-package-key.gpg exclude=kube* Step 3: Install Kubelet, kubectl, Kubeadm with command “yum install -y kubelet kubeadm kubectl --disableexcludes=kubernetes” Step 4: Run command “sudo systemctl enable kubelet” to enable kubelet Step 5: Run command “sudo systemctl start kubelet” to start kubelet
3	On master node: Step 1: Run command “sudo subl /proc/sys/net/bridge/bridge-nf-call-iptables” to override 0 with 1 Step 2: Run command “sudo kubeadm init” to initialize master node Step 3: Copy Kubeadm token to join nodes with master “kubeadm join 172.42.1.55:6443 --token pe8nhr.5yavfa6dufosbrc7 --discovery-token-ca-cert-hash sha256:d1d55b2e83f7cf101de8b82e2258c843cbf0c9a5c8ab7307119835fb9b77ff08” Step 4: Run commands as shown below to use cluster 1] mkdir -p $HOME/.kube 2] sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config 3] sudo chown $(id -u):$(id -g) $HOME/.kube/config
4	On master node:

	Step 1: cd to /home/user1/kubews/ClustersetupWS Step 2: Create new yaml file, command is “sudo subl podntwrkadon.yaml” Step 3: Add configurations to newly created yaml file from https://raw.githubusercontent.com/coreos/flannel/v0.10.0/Documentation/kube-flannel.yml Step 4: