Lesson 05 Demo 05

Deploying the Flask Application with Redis

Objective: To deploy and verify a Flask application integrated with Redis in a Kubernetes environment, demonstrating end-to-end containerized application setup and management

Tools required: kubeadm, kubectl, kubelet, and containerd

Prerequisites: A Kubernetes cluster should already be set up (refer to the steps provided in Lesson 02, Demo 01 for guidance). Ensure you have a Docker account or create one at **https://www.docker.com/**.

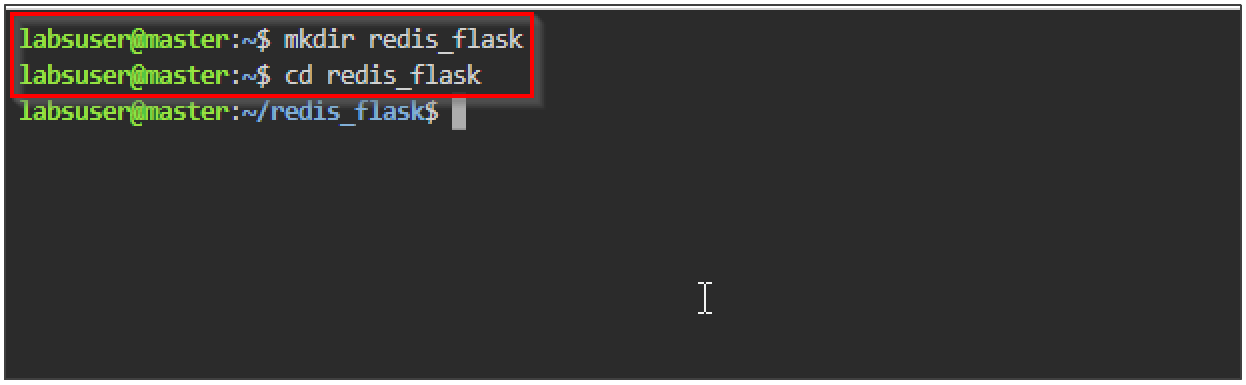
Steps to be followed:

1. Create a directory and add the necessary files
2. Create and tag the Flask image
3. Log into Docker and push the Flask image
4. Create the Redis and Flask deployments
5. Create the Redis and Flask services
6. Verify the Flask application

Step 1: Create a directory and add the necessary files

1.1 Create and navigate to the **redis_flask** directory by using the following commands:

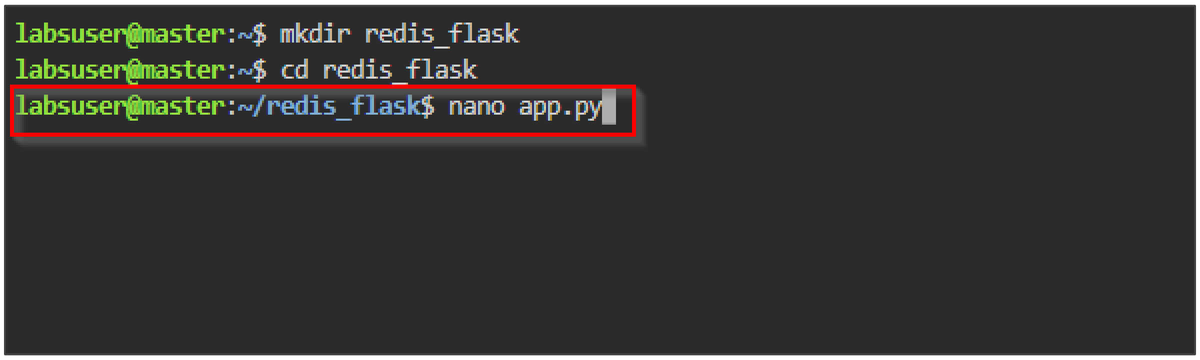
mkdir redis_flask
cd redis_flask



```
labsuser@master:~$ mkdir redis_flask
labsuser@master:~$ cd redis_flask
labsuser@master:~/redis_flask$
```

1.2 Create an **app.py** file by using the following command:

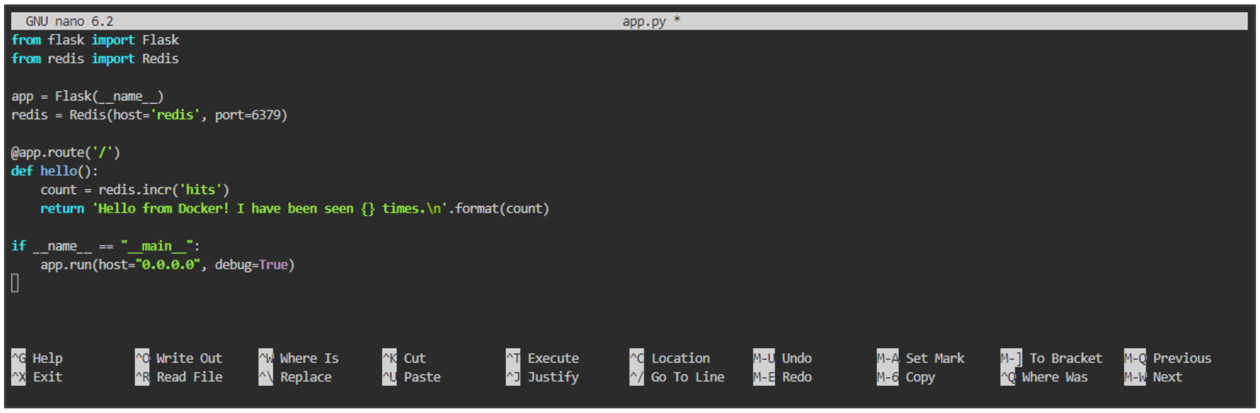
nano app.py



```
labsuser@master:~$ mkdir redis_flask
labsuser@master:~$ cd redis_flask
labsuser@master:~/redis_flask$ nano app.py
```

1.3 Add the following code to the **app.py** file:

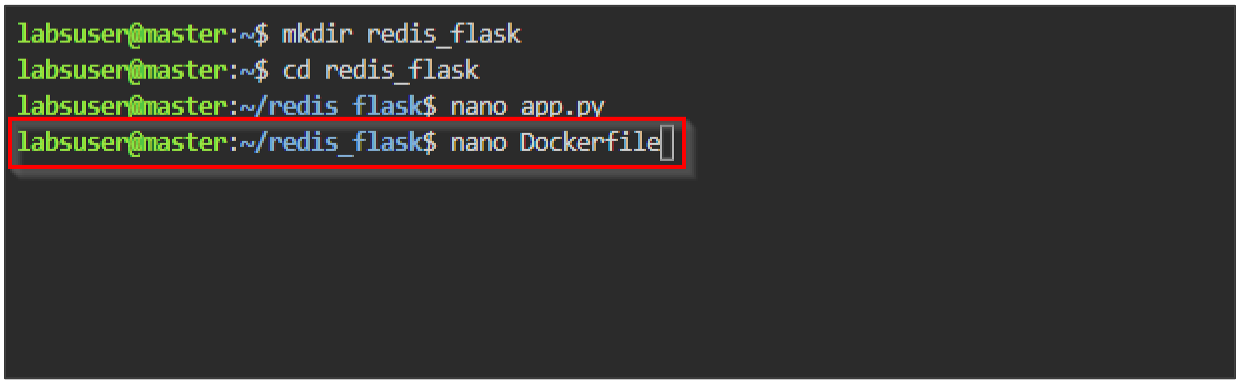
```
from flask import Flask
from redis import Redis

app = Flask(__name__)
redis = Redis(host='redis', port=6379)

@app.route('/')
def hello():
    count = redis.incr('hits')
    return 'Hello from Docker! I have been seen {} times.\n'.format(count)

if __name__ == "__main__":
    app.run(host="0.0.0.0", debug=True)
```



```
  GNU nano 6.2                                     app.py *
from flask import Flask
from redis import Redis

app = Flask(__name__)
redis = Redis(host='redis', port=6379)

@app.route('/')
def hello():
    count = redis.incr('hits')
    return 'Hello from Docker! I have been seen {} times.\n'.format(count)

if __name__ == "__main__":
    app.run(host="0.0.0.0", debug=True)

^G Help      ^O Write Out  ^W Where Is   ^K Cut       ^T Execute   ^C Location   M-U Undo   M-A Set Mark   M-] To Bracket  M-Q Previous
^X Exit      ^R Read File  ^\ Replace    ^U Paste     ^J Justify   ^/ Go To Line M-E Redo   M-6 Copy       ^Q Where Was    M-W Next
```

1.4 Create a file named **Dockerfile** by using the command:

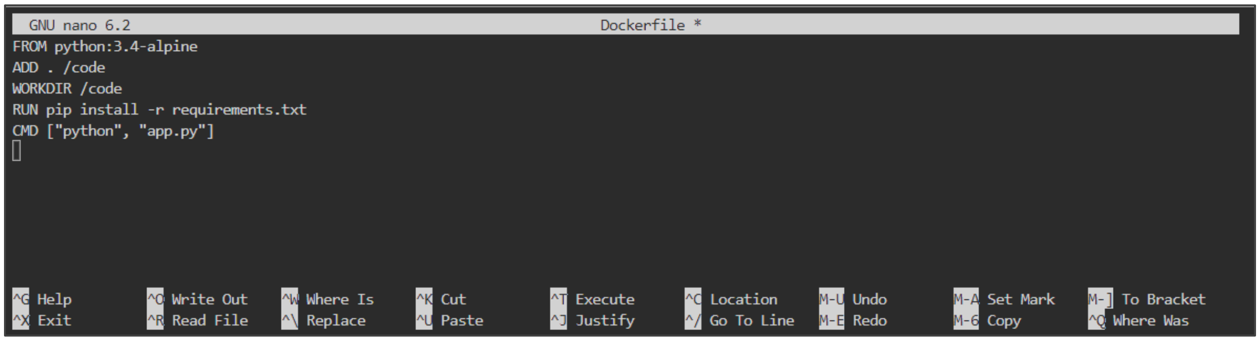
nano Dockerfile



```
labsuser@master:~$ mkdir redis_flask
labsuser@master:~$ cd redis_flask
labsuser@master:~/redis_flask$ nano app.py
labsuser@master:~/redis_flask$ nano Dockerfile
```

1.5 Add the following code to the **Dockerfile**:

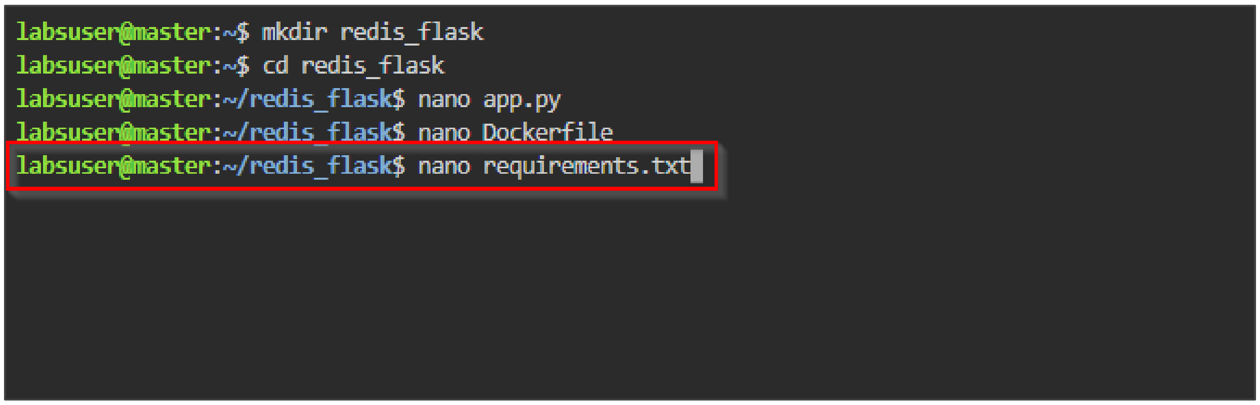
FROM python:3.4-alpine
ADD . /code
WORKDIR /code
RUN pip install -r requirements.txt
CMD ["python", "app.py"]



```
  GNU nano 6.2                                   Dockerfile *
FROM python:3.4-alpine
ADD . /code
WORKDIR /code
RUN pip install -r requirements.txt
CMD ["python", "app.py"]


^G Help      ^O Write Out  ^W Where Is  ^K Cut      ^T Execute  ^C Location    M-U Undo   M-A Set Mark  M-] To Bracket
^X Exit      ^R Read File  ^\ Replace   ^U Paste    ^J Justify  ^/ Go To Line  M-E Redo   M-6 Copy      ^Q Where Was
```

1.6 Create a **requirements.txt** file by using the command:

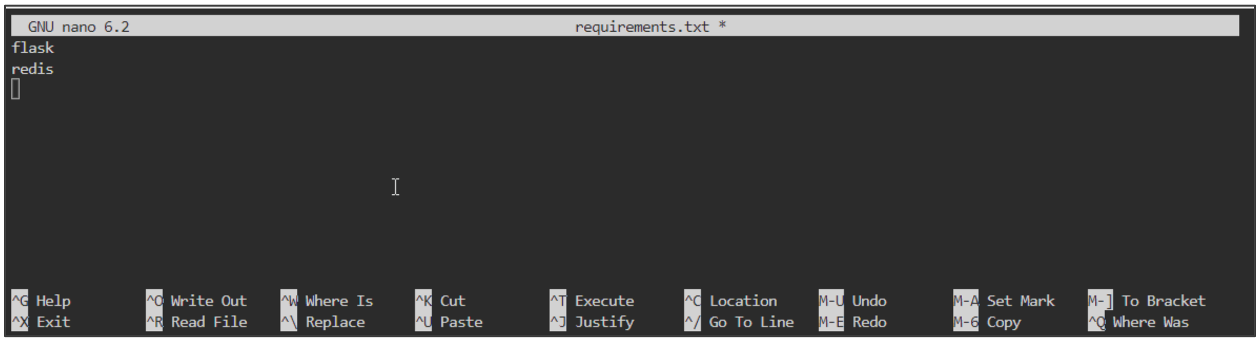
nano requirements.txt



```
labsuser@master:~$ mkdir redis_flask
labsuser@master:~$ cd redis_flask
labsuser@master:~/redis_flask$ nano app.py
labsuser@master:~/redis_flask$ nano Dockerfile
labsuser@master:~/redis_flask$ nano requirements.txt
```

1.7 Add the following code to the **requirements.txt** file:

flask
redis

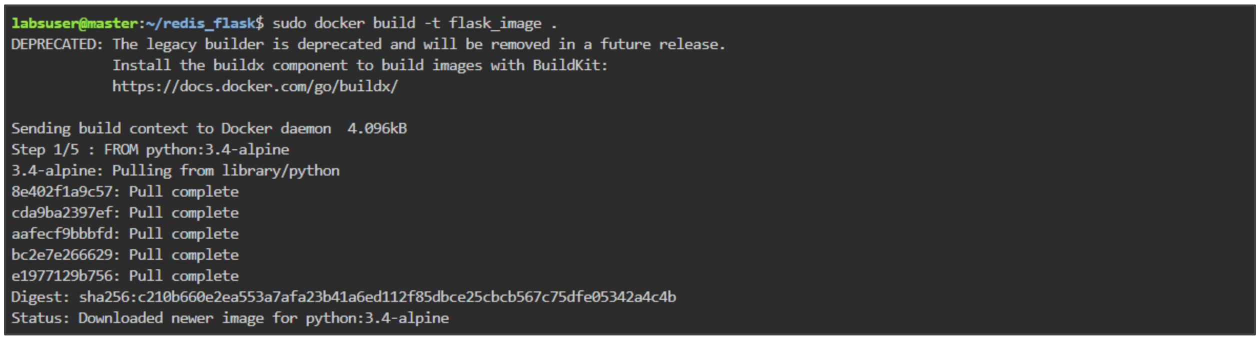


```
  GNU nano 6.2                                    requirements.txt *
flask
redis

^G Help       ^O Write Out  ^W Where Is   ^K Cut        ^T Execute    ^C Location   M-U Undo      M-A Set Mark  M-] To Bracket
^X Exit       ^R Read File  ^\ Replace    ^U Paste      ^J Justify    ^/ Go To Line M-E Redo      M-6 Copy      ^Q Where Was
```

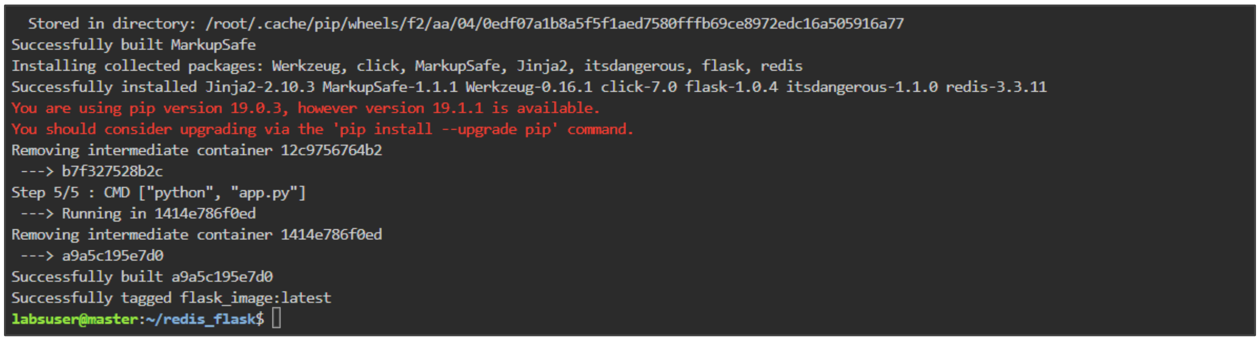
Step 2: Create and tag the Flask image

2.1 Create a Flask app image by using the following command:

sudo docker build -t flask_image .



```
labsuser@master:~/redis_flask$ sudo docker build -t flask_image .
DEPRECATED: The legacy builder is deprecated and will be removed in a future release.
            Install the buildx component to build images with BuildKit:
            https://docs.docker.com/go/buildx/

Sending build context to Docker daemon  4.096kB
Step 1/5 : FROM python:3.4-alpine
3.4-alpine: Pulling from library/python
8e402f1a9c57: Pull complete
cda9ba2397ef: Pull complete
aafecf9bbbfd: Pull complete
bc2e7e266629: Pull complete
e1977129b756: Pull complete
Digest: sha256:c210b660e2ea553a7afa23b41a6ed112f85dbce25cbcb567c75dfe05342a4c4b
Status: Downloaded newer image for python:3.4-alpine
```

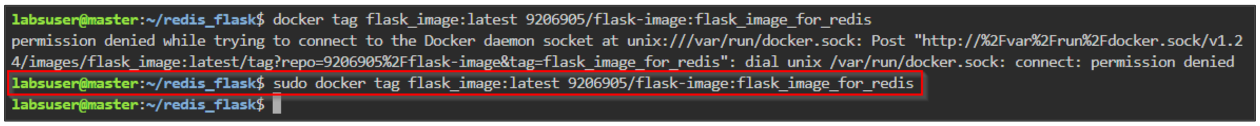


```
  Stored in directory: /root/.cache/pip/wheels/f2/aa/04/0edf07a1b8a5f5f1aed7580fffb69ce8972edc16a505916a77
Successfully built MarkupSafe
Installing collected packages: Werkzeug, click, MarkupSafe, Jinja2, itsdangerous, flask, redis
Successfully installed Jinja2-2.10.3 MarkupSafe-1.1.1 Werkzeug-0.16.1 click-7.0 flask-1.0.4 itsdangerous-1.1.0 redis-3.3.11
You are using pip version 19.0.3, however version 19.1.1 is available.
You should consider upgrading via the 'pip install --upgrade pip' command.
Removing intermediate container 12c9756764b2
 ---> b7f327528b2c
Step 5/5 : CMD ["python", "app.py"]
 ---> Running in 1414e786f0ed
Removing intermediate container 1414e786f0ed
 ---> a9a5c195e7d0
Successfully built a9a5c195e7d0
Successfully tagged flask_image:latest
labsuser@master:~/redis_flask$
```

2.2 Replace the **<docker-id>** with your docker username and tag the image by using the following commands, as shown in the screenshot below:

sudo docker tag flask_image:latest <docker-id>/flask-image:flask_image_for_redis

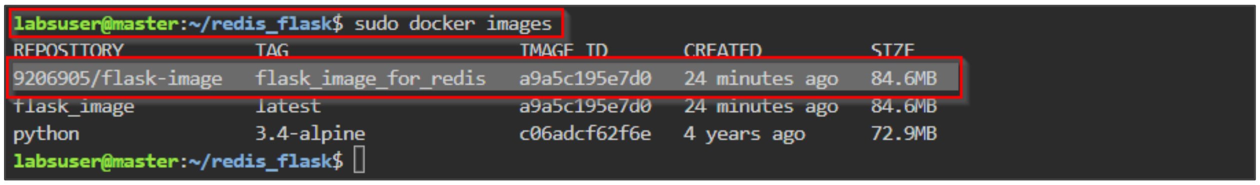
Note: If your Docker username is Alex, the above command can be written as follows:
sudo docker tag flask_image:latest Alex/flask-image:flask_image_for_redis



```
labsuser@master:~/redis_flask$ docker tag flask_image:latest 9206905/flask-image:flask_image_for_redis
permission denied while trying to connect to the Docker daemon socket at unix:///var/run/docker.sock: Post "http://%2Fvar%2Frun%2Fdocker.sock/v1.2
4/images/flask_image:latest/tag?repo=9206905%2Fflask-image&tag=flask_image_for_redis": dial unix /var/run/docker.sock: connect: permission denied
labsuser@master:~/redis_flask$ sudo docker tag flask_image:latest 9206905/flask-image:flask_image_for_redis
labsuser@master:~/redis_flask$
```

2.3 Verify the tagged image by using the following command:

sudo docker images

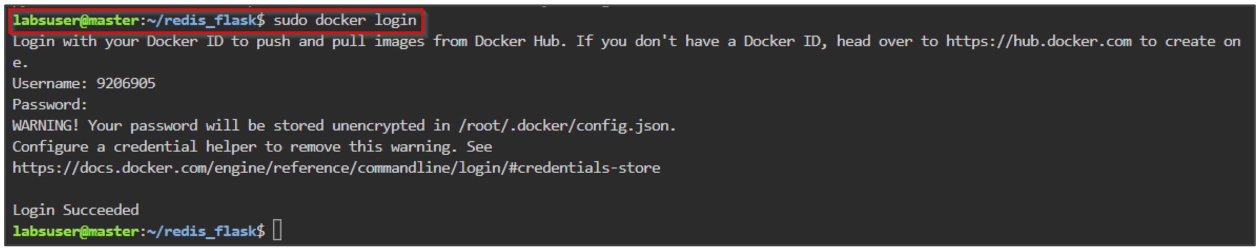


```
labsuser@master:~/redis_flask$ sudo docker images
REPOSITORY            TAG                      IMAGE ID       CREATED          SIZE
9206905/flask-image   flask_image_for_redis    a9a5c195e7d0   24 minutes ago   84.6MB
flask_image           latest                   a9a5c195e7d0   24 minutes ago   84.6MB
python                3.4-alpine               c06adcf62f6e   4 years ago      72.9MB
labsuser@master:~/redis_flask$ 
```

Step 3: Log into Docker and push the Flask image

3.1 Log into Docker using the following command:

sudo docker login

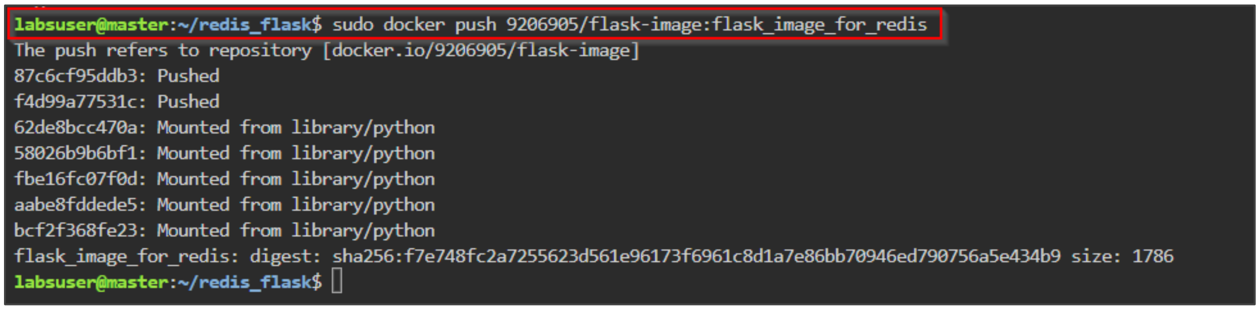


```
labsuser@master:~/redis_flask$ sudo docker login
Login with your Docker ID to push and pull images from Docker Hub. If you don't have a Docker ID, head over to https://hub.docker.com to create on
e.
Username: 9206905
Password:
WARNING! Your password will be stored unencrypted in /root/.docker/config.json.
Configure a credential helper to remove this warning. See
https://docs.docker.com/engine/reference/commandline/login/#credentials-store

Login Succeeded
labsuser@master:~/redis_flask$ 
```

3.2 Replace the **<docker-id>** with your docker username and push the Flask image to the Docker repository by using the following command as shown in the screenshot below:

sudo docker push <docker-id>/flask-image:flask_image_for_redis

Note: If your docker username is alex, the above command can be written as follows:
sudo docker push alex/flask-image:flask_image_for_redis

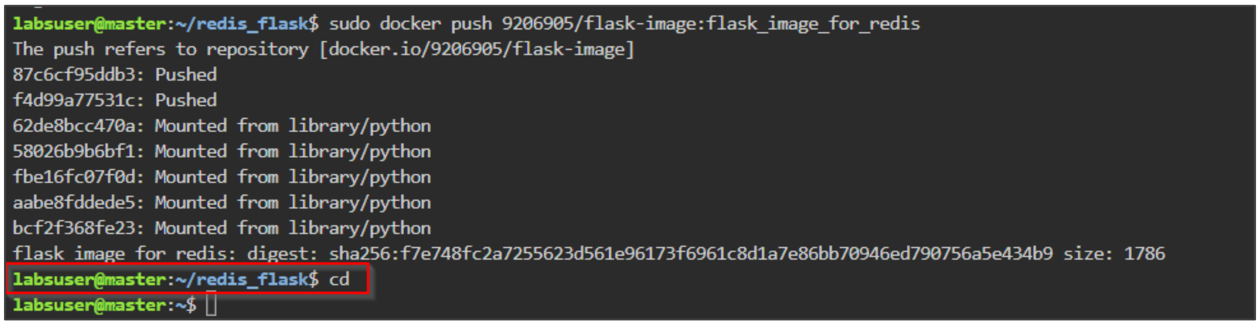


```
labsuser@master:~/redis_flask$ sudo docker push 9206905/flask-image:flask_image_for_redis
The push refers to repository [docker.io/9206905/flask-image]
87c6cf95ddb3: Pushed
f4d99a77531c: Pushed
62de8bcc470a: Mounted from library/python
58026b9b6bf1: Mounted from library/python
fbe16fc07f0d: Mounted from library/python
aabe8fddede5: Mounted from library/python
bcf2f368fe23: Mounted from library/python
flask_image_for_redis: digest: sha256:f7e748fc2a7255623d561e96173f6961c8d1a7e86bb70946ed790756a5e434b9 size: 1786
labsuser@master:~/redis_flask$ 
```

Step 4: Create the Redis and Flask deployments

4.1 Navigate to the home directory using the following command:

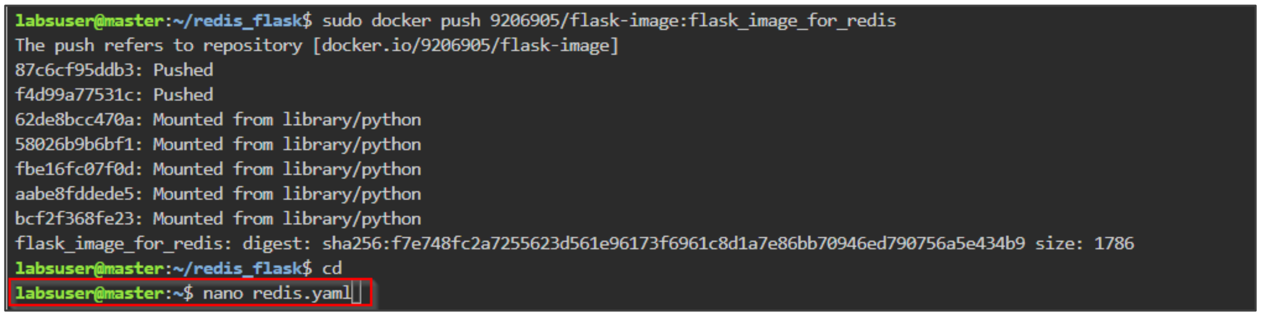
cd



```
labsuser@master:~/redis_flask$ sudo docker push 9206905/flask-image:flask_image_for_redis
The push refers to repository [docker.io/9206905/flask-image]
87c6cf95ddb3: Pushed
f4d99a77531c: Pushed
62de8bcc470a: Mounted from library/python
58026b9b6bf1: Mounted from library/python
fbe16fc07f0d: Mounted from library/python
aabe8fddede5: Mounted from library/python
bcf2f368fe23: Mounted from library/python
flask_image_for_redis: digest: sha256:f7e748fc2a7255623d561e96173f6961c8d1a7e86bb70946ed790756a5e434b9 size: 1786
labsuser@master:~/redis_flask$ cd
labsuser@master:~$
```

4.2 Create the **redis.yaml** file by using the following command:

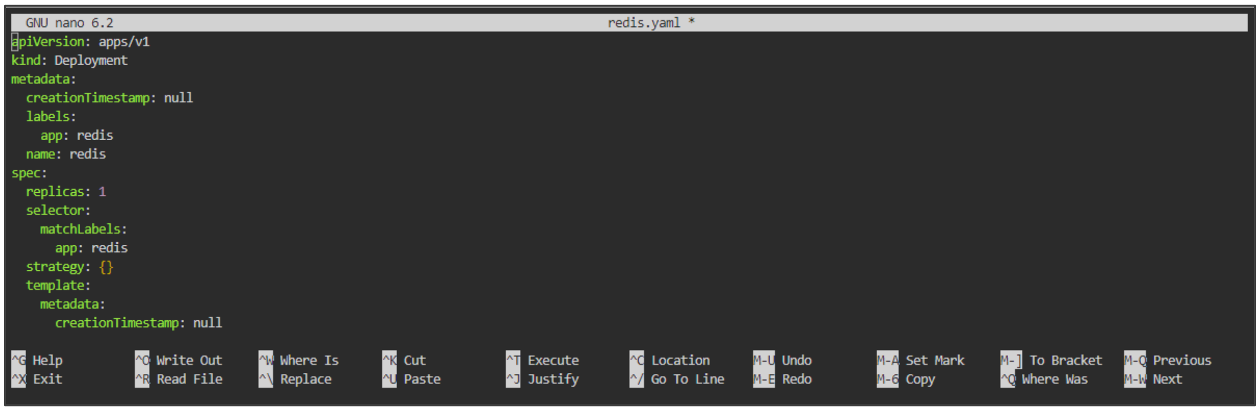
nano redis.yaml



```
labsuser@master:~/redis_flask$ sudo docker push 9206905/flask-image:flask_image_for_redis
The push refers to repository [docker.io/9206905/flask-image]
87c6cf95ddb3: Pushed
f4d99a77531c: Pushed
62de8bcc470a: Mounted from library/python
58026b9b6bf1: Mounted from library/python
fbe16fc07f0d: Mounted from library/python
aabe8fddede5: Mounted from library/python
bcf2f368fe23: Mounted from library/python
flask_image_for_redis: digest: sha256:f7e748fc2a7255623d561e96173f6961c8d1a7e86bb70946ed790756a5e434b9 size: 1786
labsuser@master:~/redis_flask$ cd
labsuser@master:~$ nano redis.yaml
```

4.3 Add the following code to the **redis.yaml** file:

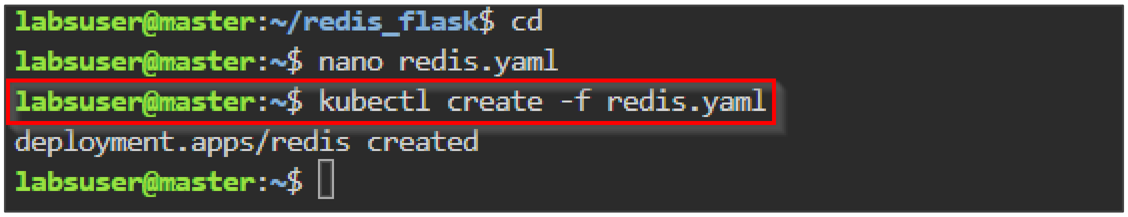
```
apiVersion: apps/v1
kind: Deployment
metadata:
  creationTimestamp: null
  labels:
    app: redis
  name: redis
spec:
  replicas: 1
  selector:
    matchLabels:
      app: redis
  strategy: {}
  template:
    metadata:
      creationTimestamp: null
      labels:
        app: redis
    spec:
      containers:
      - image: redis
        name: redis
        resources: {}
status: {}
```



```
  GNU nano 6.2                                  redis.yaml *
apiVersion: apps/v1
kind: Deployment
metadata:
  creationTimestamp: null
  labels:
    app: redis
  name: redis
spec:
  replicas: 1
  selector:
    matchLabels:
      app: redis
  strategy: {}
  template:
    metadata:
      creationTimestamp: null

^G Help       ^O Write Out  ^W Where Is   ^K Cut        ^T Execute    ^C Location   M-U Undo      M-A Set Mark  M-] To Bracket  M-Q Previous
^X Exit       ^R Read File  ^\ Replace    ^U Paste      ^J Justify    ^/ Go To Line M-E Redo      M-6 Copy      ^Q Where Was    M-W Next
```

4.4 Create the Redis deployment resource by using the following command:

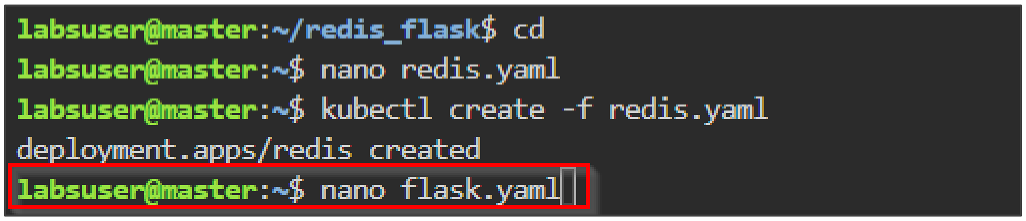
kubectl create -f redis.yaml



```
labsuser@master:~/redis_flask$ cd
labsuser@master:~$ nano redis.yaml
labsuser@master:~$ kubectl create -f redis.yaml
deployment.apps/redis created
labsuser@master:~$ 
```

4.5 Create the **flask.yaml** file by using the following command:

nano flask.yaml



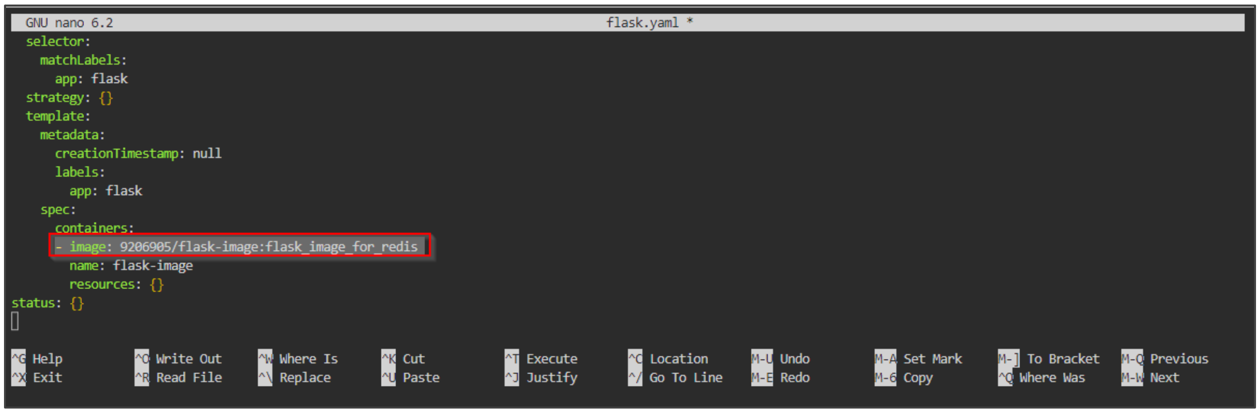
```
labsuser@master:~/redis_flask$ cd
labsuser@master:~$ nano redis.yaml
labsuser@master:~$ kubectl create -f redis.yaml
deployment.apps/redis created
labsuser@master:~$ nano flask.yaml
```

4.6 Add the following code to the **flask.yaml** file:

```
apiVersion: apps/v1
kind: Deployment
metadata:
  creationTimestamp: null
  labels:
    app: flask
  name: flask
spec:
  replicas: 1
  selector:
    matchLabels:
      app: flask
  strategy: {}
  template:
    metadata:
```

```
    creationTimestamp: null
    labels:
      app: flask
  spec:
    containers:
    - image: 9206905/flask-image:flask_image_for_redis
      name: flask-image
      resources: {}
status: {}
```

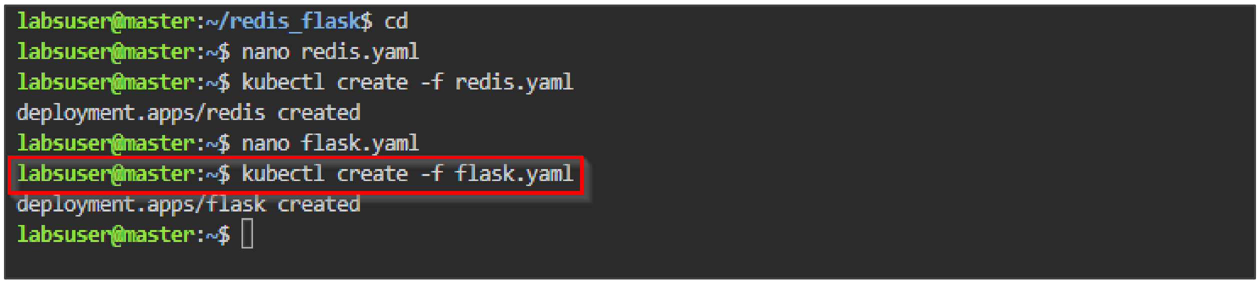
Note: Replace the image repository in the YAML file with yours accordingly, as shown in the highlighted line in the screenshot below:



```
  GNU nano 6.2                                   flask.yaml *
  selector:
    matchLabels:
      app: flask
  strategy: {}
  template:
    metadata:
      creationTimestamp: null
      labels:
        app: flask
    spec:
      containers:
      - image: 9206905/flask-image:flask_image_for_redis
        name: flask-image
        resources: {}
status: {}
```

4.7 Create the Flask deployment resource by using the following command:

kubectl create -f flask.yaml

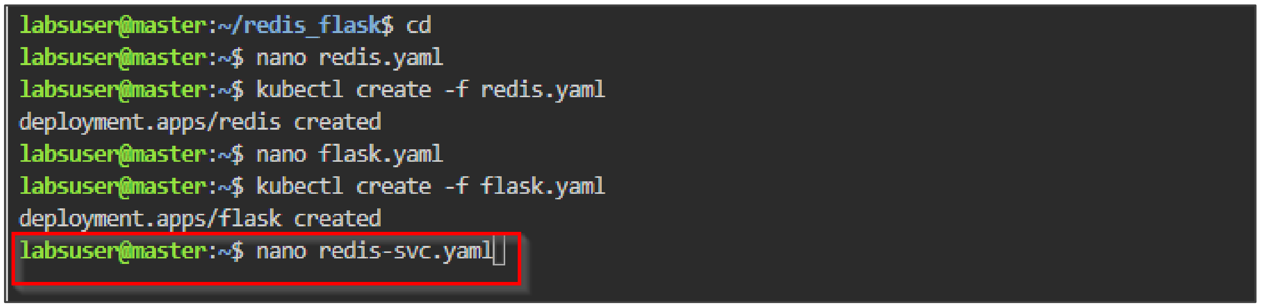


```
labsuser@master:~/redis_flask$ cd
labsuser@master:~$ nano redis.yaml
labsuser@master:~$ kubectl create -f redis.yaml
deployment.apps/redis created
labsuser@master:~$ nano flask.yaml
labsuser@master:~$ kubectl create -f flask.yaml
deployment.apps/flask created
labsuser@master:~$
```

Step 5: Create the Redis and Flask services

5.1 Create the **redis-svc.yaml** file by using the following command:

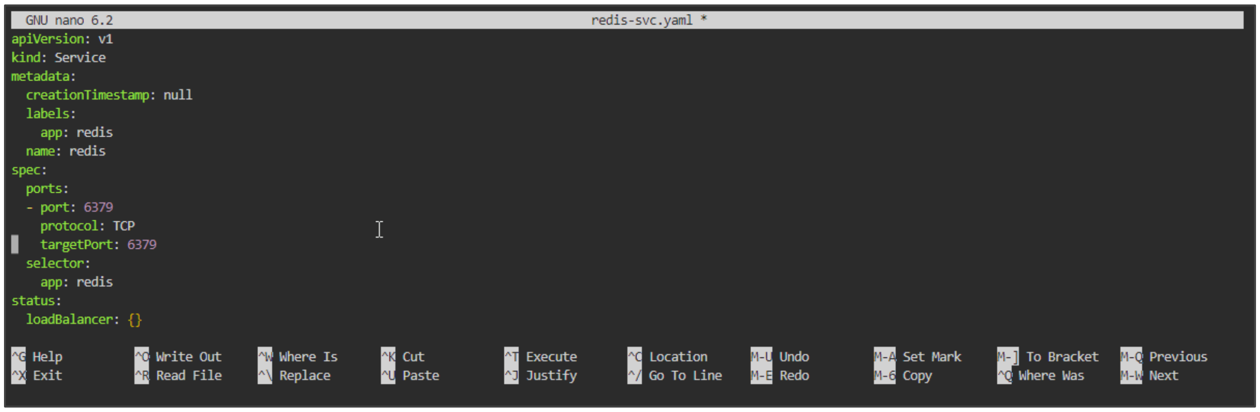
nano redis-svc.yaml



```
labsuser@master:~/redis_flask$ cd
labsuser@master:~$ nano redis.yaml
labsuser@master:~$ kubectl create -f redis.yaml
deployment.apps/redis created
labsuser@master:~$ nano flask.yaml
labsuser@master:~$ kubectl create -f flask.yaml
deployment.apps/flask created
labsuser@master:~$ nano redis-svc.yaml
```

5.2 Add the following code to the **redis-svc.yaml** file:

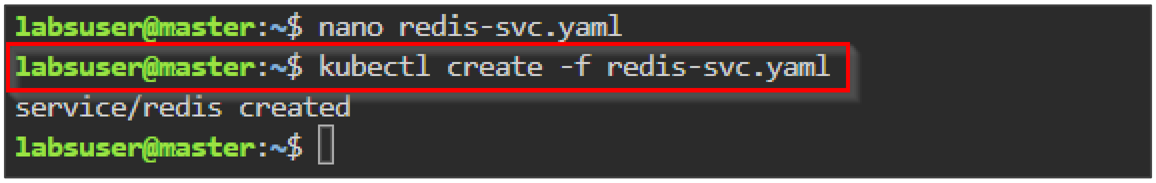
```
apiVersion: v1
kind: Service
metadata:
  creationTimestamp: null
  labels:
    app: redis
  name: redis
spec:
  ports:
  - port: 6379
    protocol: TCP
    targetPort: 6379
  selector:
    app: redis
status:
  loadBalancer: {}
```



```
  GNU nano 6.2                                     redis-svc.yaml *
apiVersion: v1
kind: Service
metadata:
  creationTimestamp: null
  labels:
    app: redis
  name: redis
spec:
  ports:
  - port: 6379
    protocol: TCP
    targetPort: 6379
  selector:
    app: redis
status:
  loadBalancer: {}

^G Help      ^O Write Out  ^W Where Is  ^K Cut    ^T Execute  ^C Location   M-U Undo  M-A Set Mark  M-] To Bracket  M-Q Previous
^X Exit      ^R Read File  ^\ Replace   ^U Paste  ^J Justify  ^/ Go To Line M-E Redo  M-6 Copy      ^Q Where Was    M-W Next
```

5.3 Create the Redis service resource using the following command:

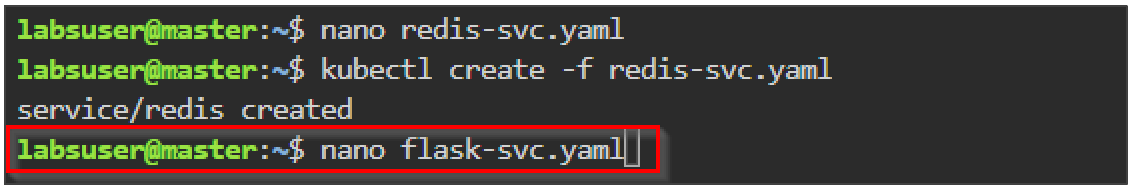
kubectl create -f redis-svc.yaml



```
labsuser@master:~$ nano redis-svc.yaml
labsuser@master:~$ kubectl create -f redis-svc.yaml
service/redis created
labsuser@master:~$ 
```

5.4 Create the **flask-svc.yaml** file by using the following command:

nano flask-svc.yaml

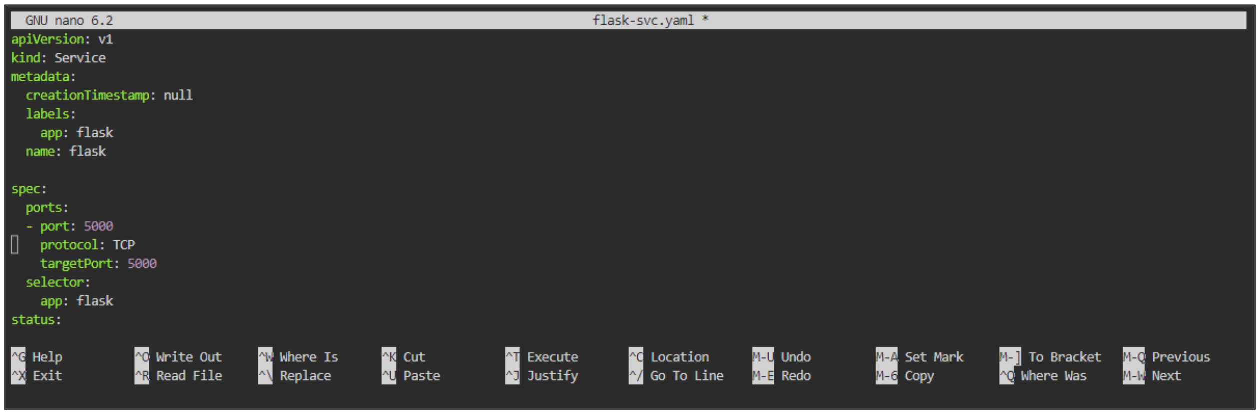


```
labsuser@master:~$ nano redis-svc.yaml
labsuser@master:~$ kubectl create -f redis-svc.yaml
service/redis created
labsuser@master:~$ nano flask-svc.yaml
```

5.5 Add the following code to the **flask-svc.yaml** file:

```
apiVersion: v1
kind: Service
metadata:
  creationTimestamp: null
  labels:
    app: flask
  name: flask
```

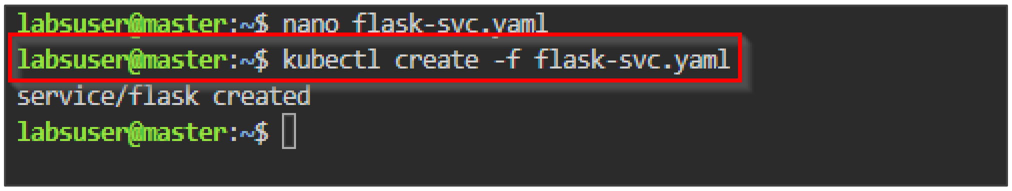
```
spec:
  ports:
  - port: 5000
    protocol: TCP
    targetPort: 5000
  selector:
    app: flask
status:
  loadBalancer: {}
```



```
  GNU nano 6.2                                   flask-svc.yaml *
apiVersion: v1
kind: Service
metadata:
  creationTimestamp: null
  labels:
    app: flask
  name: flask

spec:
  ports:
  - port: 5000
    protocol: TCP
    targetPort: 5000
  selector:
    app: flask
status:

^G Help       ^O Write Out  ^W Where Is   ^K Cut        ^T Execute    ^C Location   M-U Undo      M-A Set Mark  M-] To Bracket  M-Q Previous
^X Exit       ^R Read File  ^\ Replace    ^U Paste      ^J Justify    ^/ Go To Line M-E Redo      M-6 Copy      ^Q Where Was    M-W Next
```

5.6 Create the Flask service resource by using the following command:

kubectl create -f flask-svc.yaml

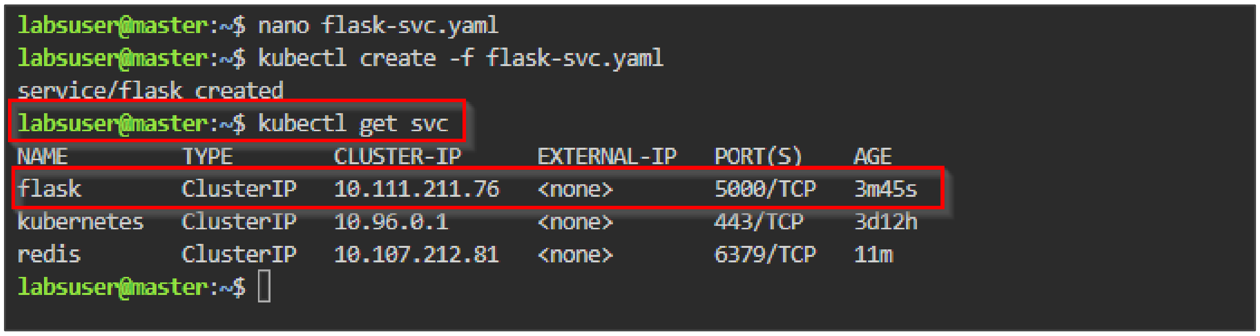


```
labsuser@master:~$ nano flask-svc.yaml
labsuser@master:~$ kubectl create -f flask-svc.yaml
service/flask created
labsuser@master:~$
```

Step 6: Verify the Flask application

6.1 Verify the Flask service by using the following command:

kubectl get svc

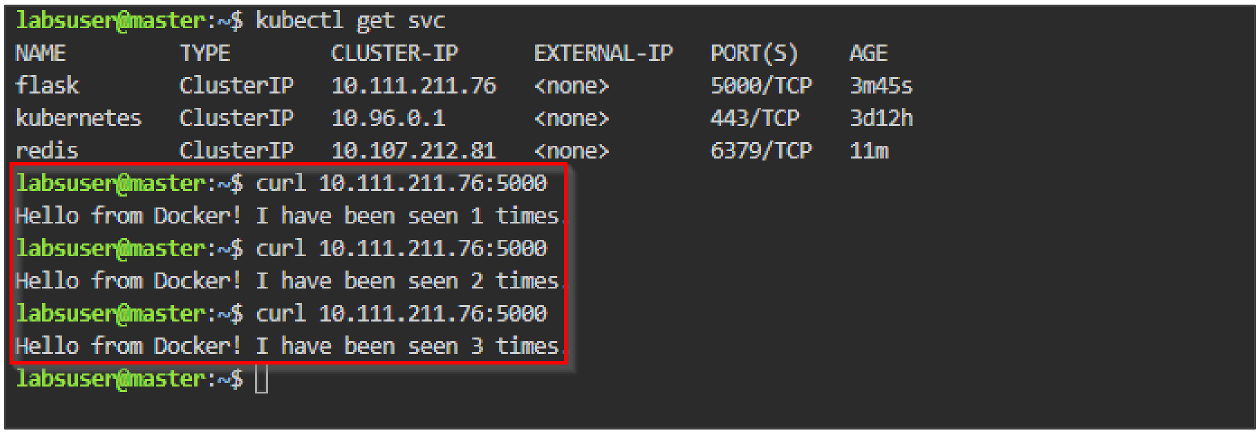


```
labsuser@master:~$ nano flask-svc.yaml
labsuser@master:~$ kubectl create -f flask-svc.yaml
service/flask created
labsuser@master:~$ kubectl get svc
NAME         TYPE        CLUSTER-IP       EXTERNAL-IP   PORT(S)    AGE
flask        ClusterIP   10.111.211.76    <none>        5000/TCP   3m45s
kubernetes   ClusterIP   10.96.0.1        <none>        443/TCP    3d12h
redis        ClusterIP   10.107.212.81    <none>        6379/TCP   11m
labsuser@master:~$
```

Note: **Copy the IP and port number and write them in the following format:**
curl <ClusterIP:PortNumber>

6.2 Verify if the Flask app is working by using the following command, as shown in the screenshot below:

curl 10.111.211.76:5000



```
labsuser@master:~$ kubectl get svc
NAME         TYPE        CLUSTER-IP       EXTERNAL-IP   PORT(S)    AGE
flask        ClusterIP   10.111.211.76    <none>        5000/TCP   3m45s
kubernetes   ClusterIP   10.96.0.1        <none>        443/TCP    3d12h
redis        ClusterIP   10.107.212.81    <none>        6379/TCP   11m
labsuser@master:~$ curl 10.111.211.76:5000
Hello from Docker! I have been seen 1 times.
labsuser@master:~$ curl 10.111.211.76:5000
Hello from Docker! I have been seen 2 times.
labsuser@master:~$ curl 10.111.211.76:5000
Hello from Docker! I have been seen 3 times.
labsuser@master:~$
```

By following these steps, you have successfully set up and deployed a containerized Flask application with Redis integration on Kubernetes.