**Reference**:

1. https://medium.com/better-programming/getting-started-with-kubernetes-for-python-254d4c1d2041

**Requirements**:

1. Docker installed on Ubuntu:

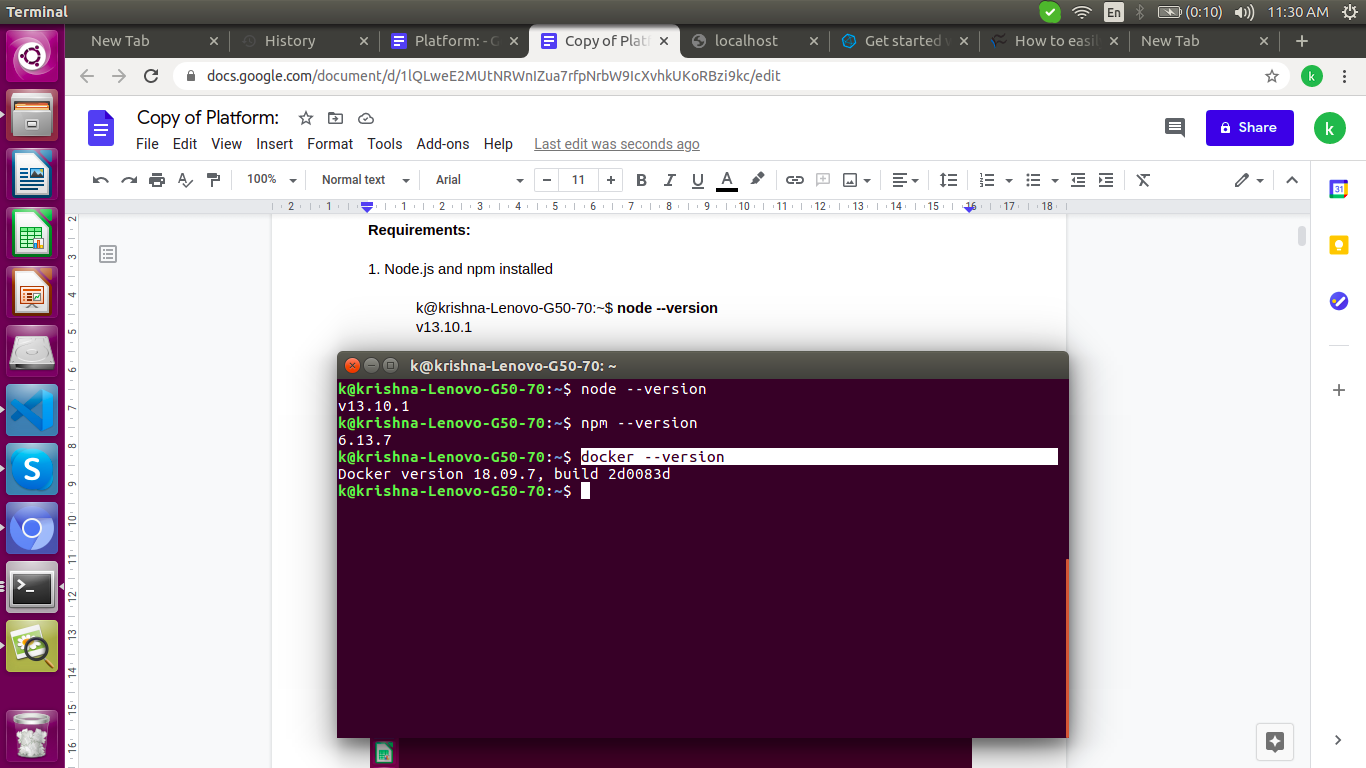
If not installed, can check link - <https://www.digitalocean.com/community/tutorials/how-to-install-and-use-docker-on-ubuntu-16-04>

If installed, verify the Docker version.

k@krishna-Lenovo-G50-70:~$ **docker --version**

Docker version 18.09.7, build 2d0083d

[pTwo]



2. Minikube installed on Ubuntu (Kubernetes):

If not installed, then follow the link - <https://matthewpalmer.net/kubernetes-app-developer/articles/install-kubernetes-ubuntu-tutorial.html>

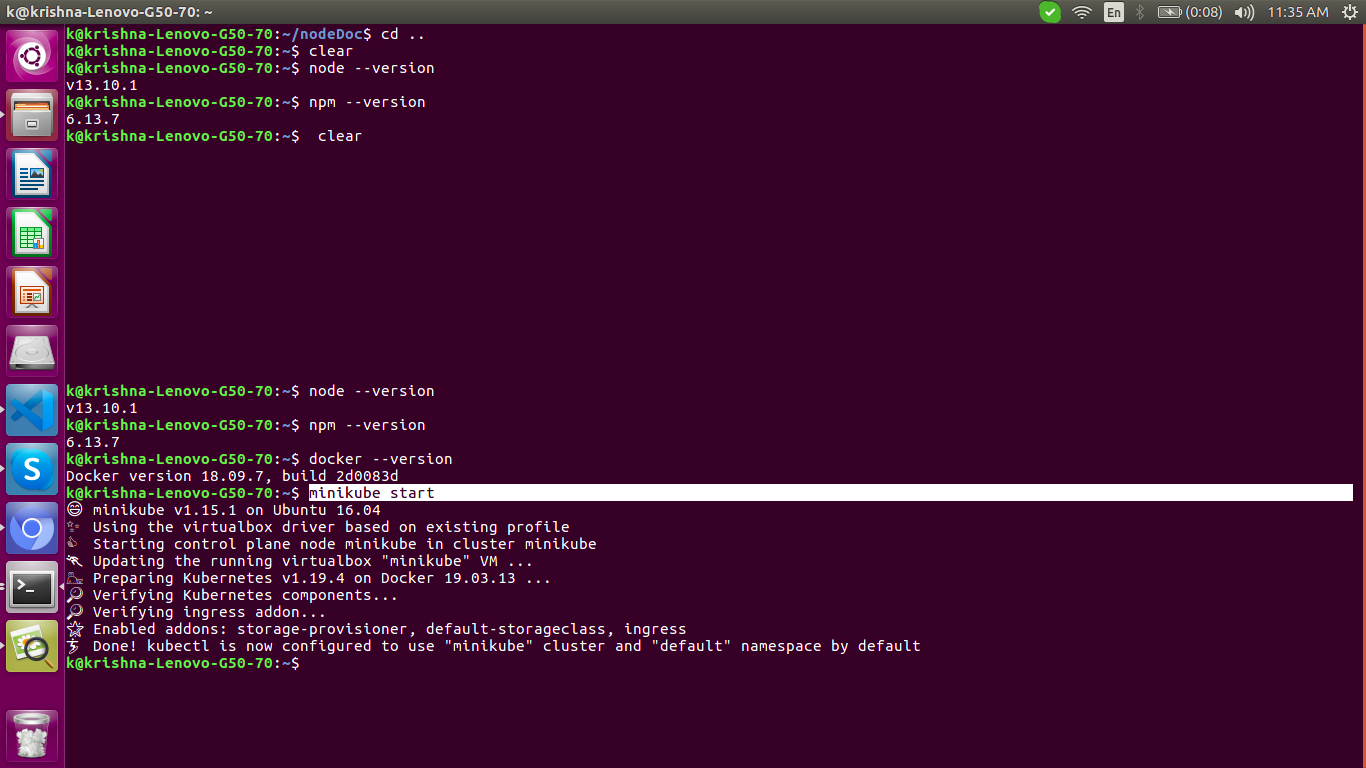
and

<https://gist.github.com/gonzaloplaza/f62fdcfdb6aac3d15a0fe0d750715729>

If installed, verfiy:

k@krishna-Lenovo-G50-70:~$ **minikube start**

[pThree]



k@krishna-Lenovo-G50-70:~$ **kubectl api-versions**

[pFour]

**Steps**:

1. Create a directory and cd into it:

k@krishna-Lenovo-G50-70:~$ **mkdir pyKubeOne**

k@krishna-Lenovo-G50-70:~$ **cd pyKubeOne**

2. Create a python main file app.py as:

k@krishna-Lenovo-G50-70:~/pyKubeOne$ **nano app.py**

The content of app.py is:

from flask import Flask

from flask\_restful import Resource, Api

import os

app = Flask(\_\_name\_\_)

api = Api(app)

class sayHello(Resource):

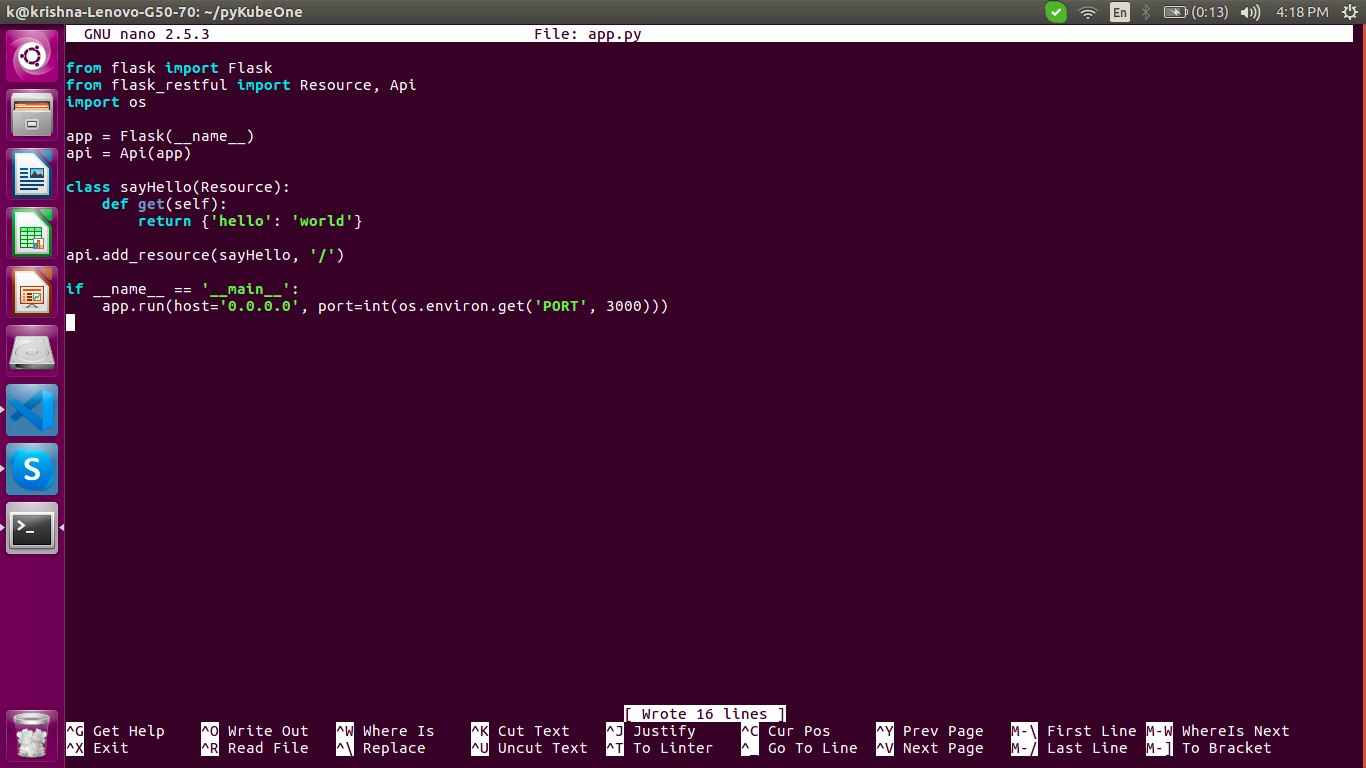
def get(self):

return {'hello': 'world'}

api.add\_resource(sayHello, '/')

if \_\_name\_\_ == '\_\_main\_\_':

app.run(host='0.0.0.0', port=int(os.environ.get('PORT', 3000)))



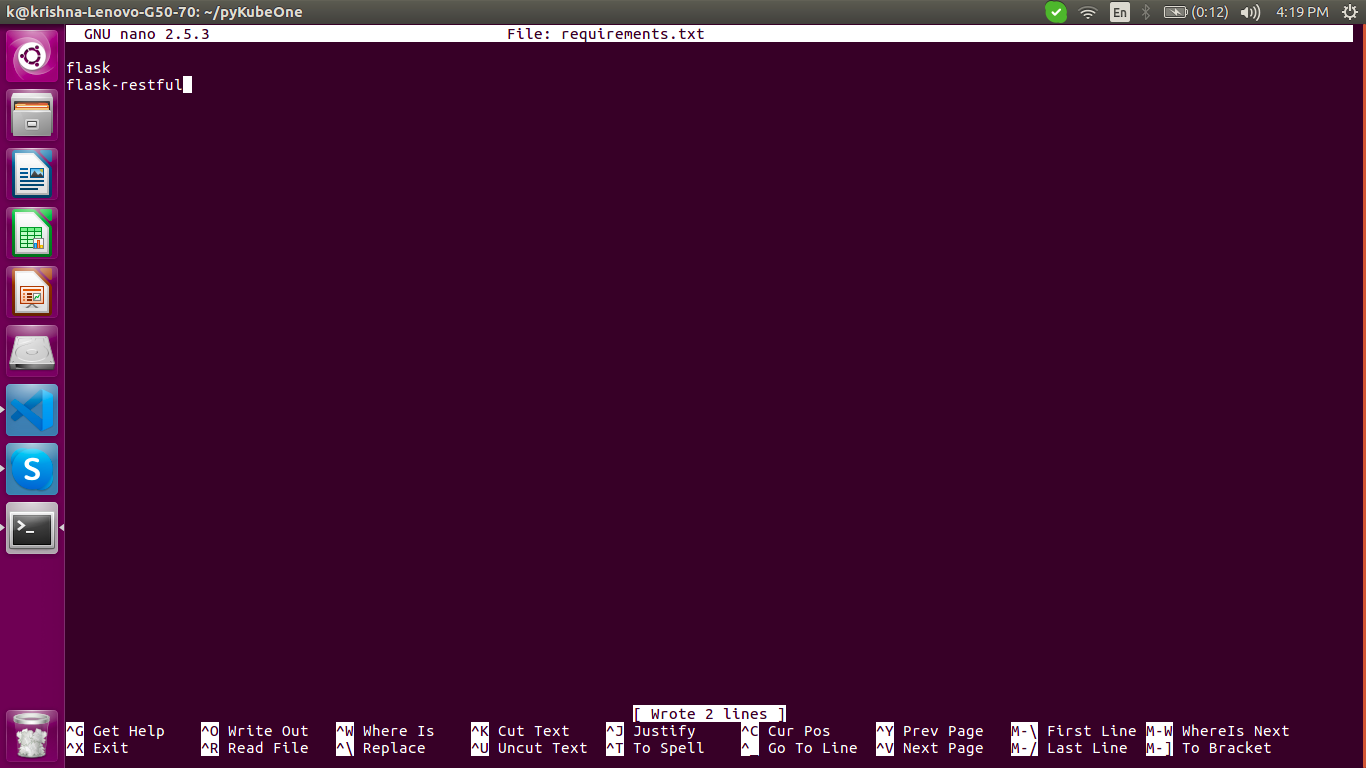
3. Create a requirements.txt file as shown:

k@krishna-Lenovo-G50-70:~/pyKubeOne$ **nano requirements.txt**

The content of requirements.txt is:

flask

flask-restful



4.Now we need to build the docker image.

k@krishna-Lenovo-G50-70:~/pyKubeOne$ **sudo docker build -t py-kube-one-image .**

We will get an error since the Dockerfile is not created for building the image.

So, create a Dockerfile as:

The content of Dockerfile is:

FROM python:3.6-slim-buster

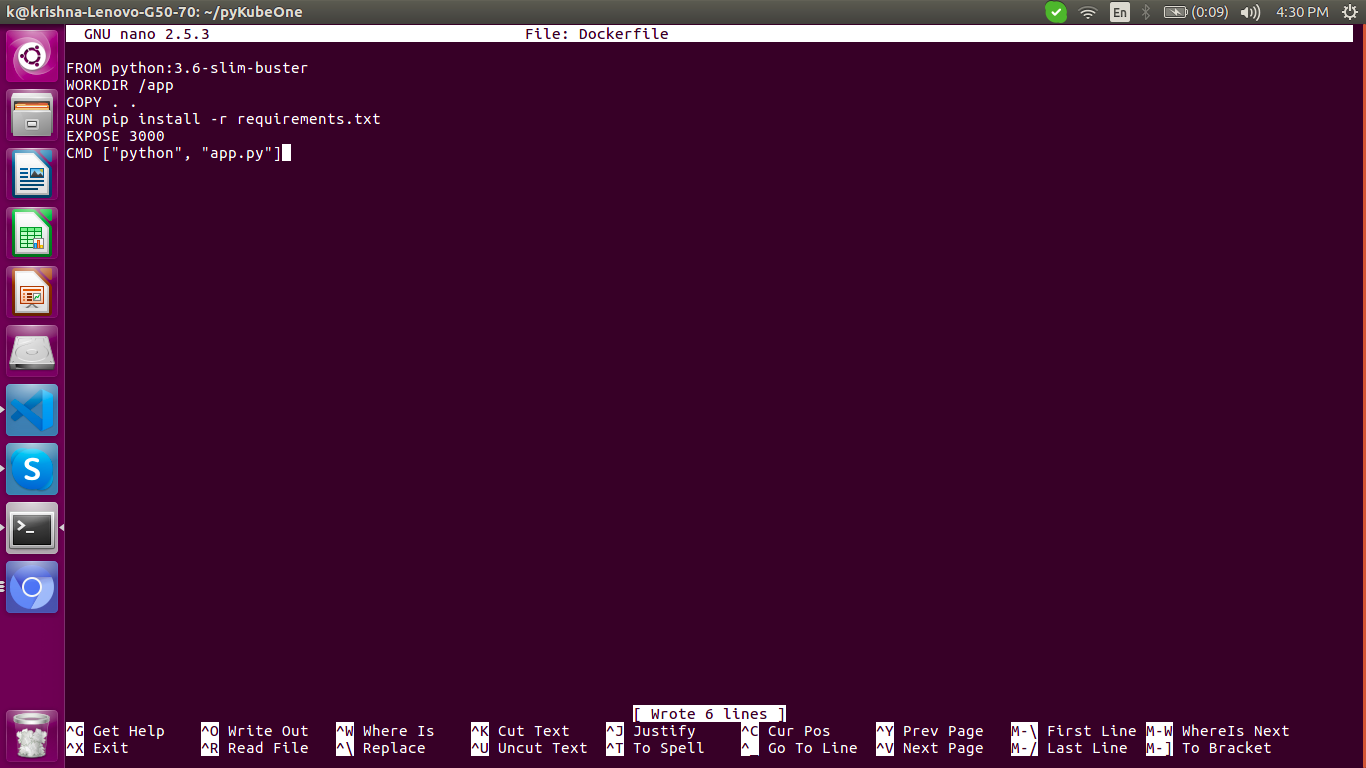
WORKDIR /app

COPY . .

RUN pip install -r requirements.txt

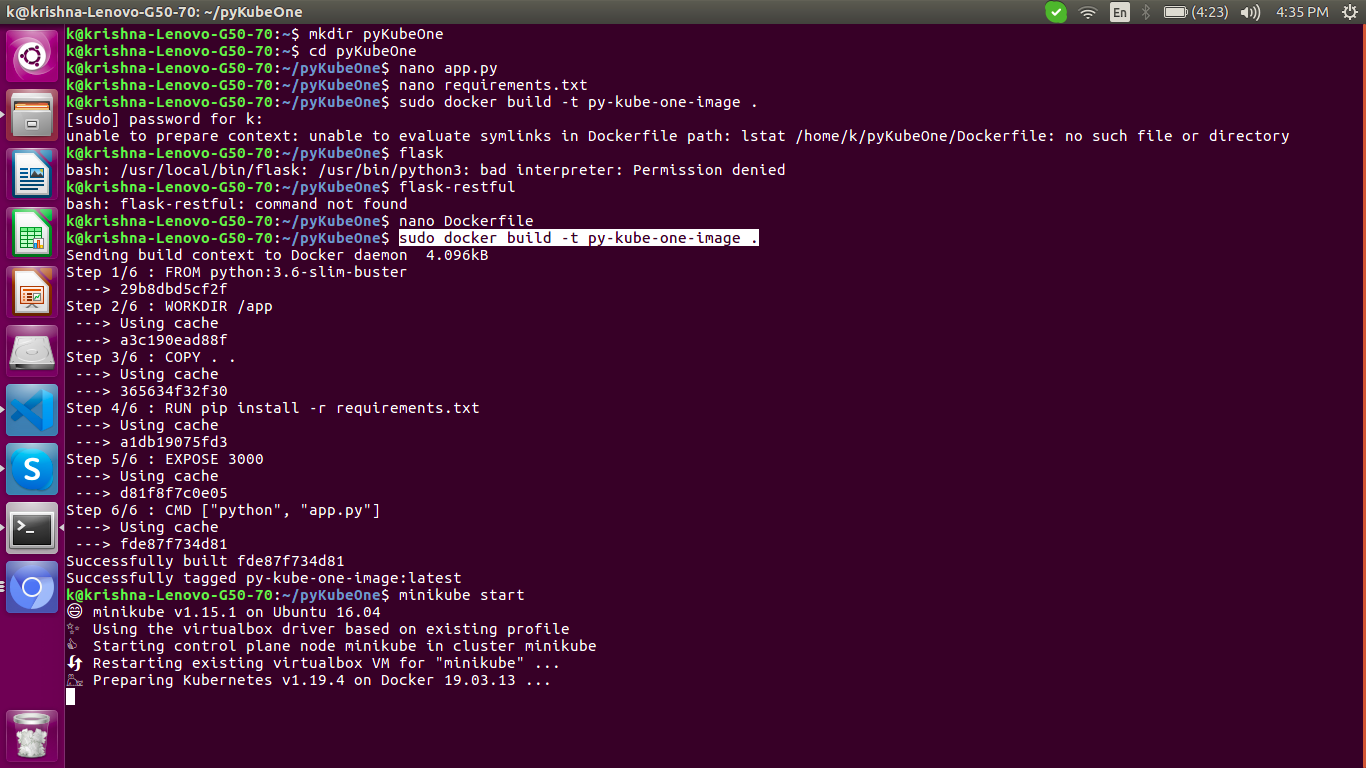
EXPOSE 3000

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



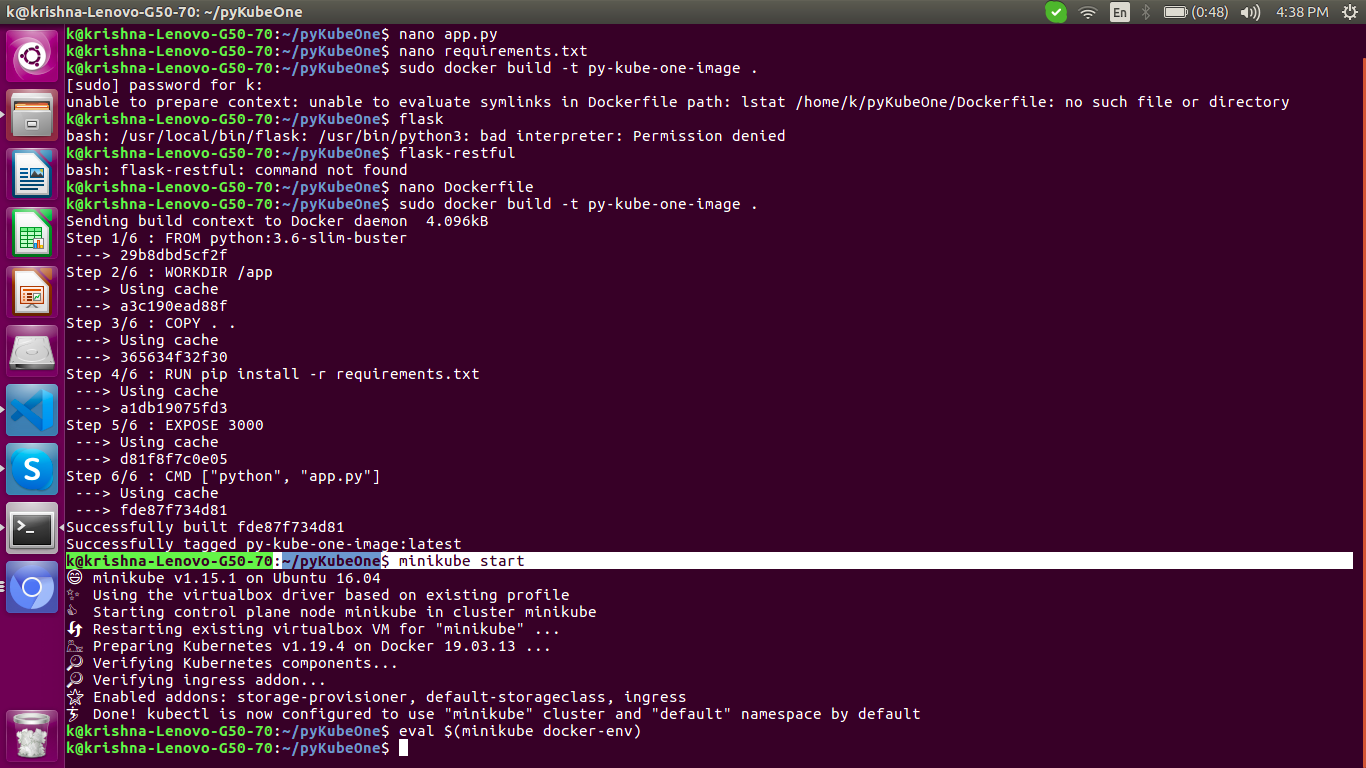
5.We will build the image with the previous command:

k@krishna-Lenovo-G50-70:~/pyKubeOne$ **sudo docker build -t py-kube-one-image .**



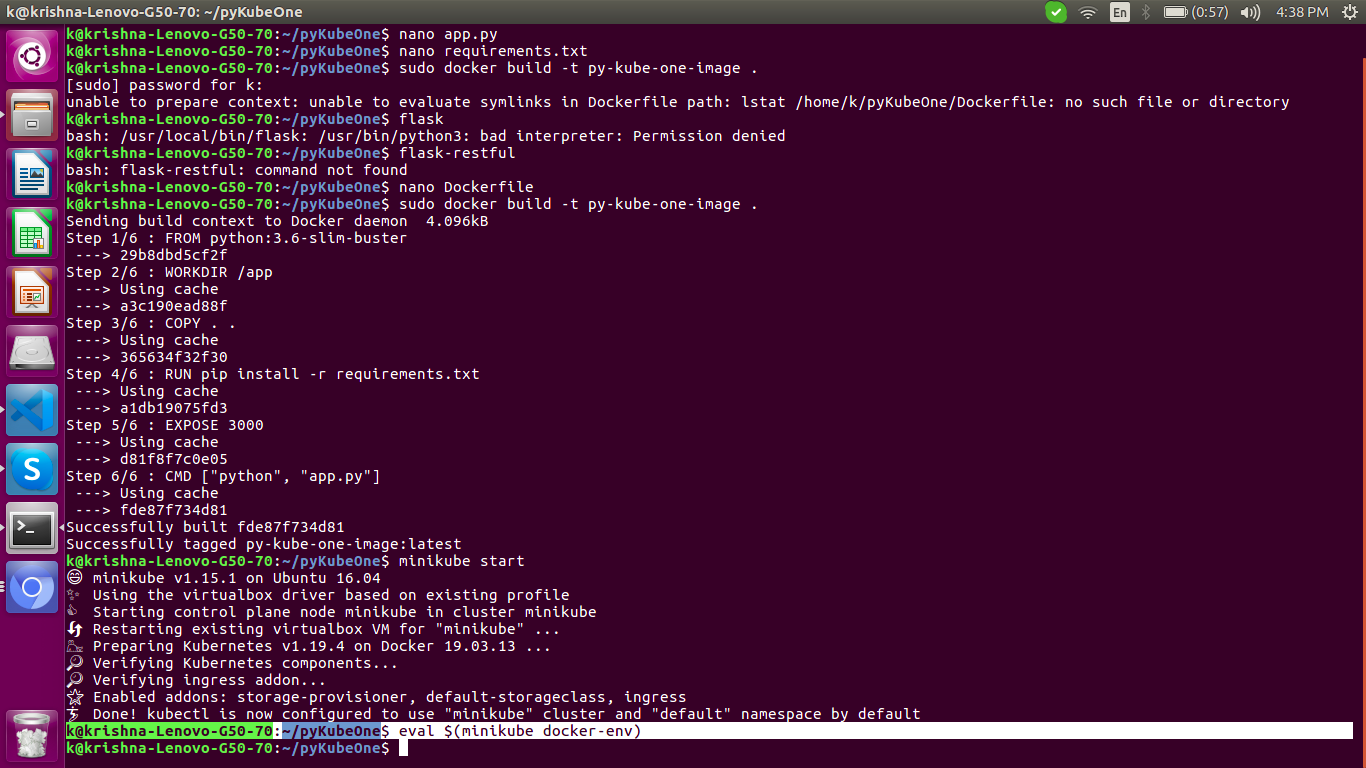
6. Once the image is build, we start kubernetes - minikube on Ubuntu 16.04 as:

k@krishna-Lenovo-G50-70:~/pyKubeOne$ **minikube start**



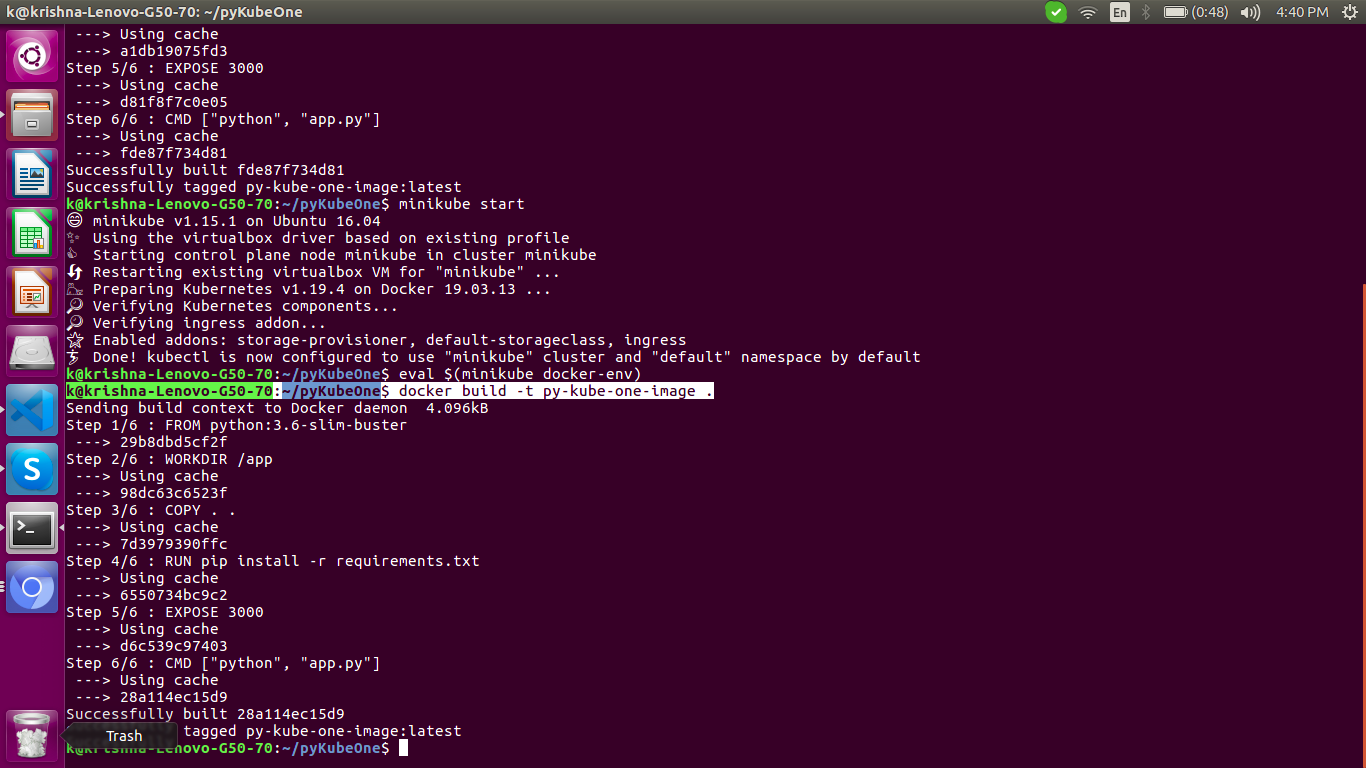
To make the docker image available to kubernetes, we run the command:

k@krishna-Lenovo-G50-70:~/pyKubeOne$ **eval $(minikube docker-env)**



Now, once again run the 'build image' command as:

k@krishna-Lenovo-G50-70:~/pyKubeOne$ **docker build -t py-kube-one-image .**



7. Now, we will create a deployment.yml file:

The content of deployment.yml is:

apiVersion: apps/v1

kind: Deployment

metadata:

creationTimestamp: null

labels:

app: py-kube-one-image

name: py-kube-one-image

spec:

replicas: 3

selector:

matchLabels:

app: py-kube-one-image

strategy: {}

template:

metadata:

creationTimestamp: null

labels:

app: py-kube-one-image

spec:

containers:

- image: py-kube-one-image:latest

name: py-kube-one-image

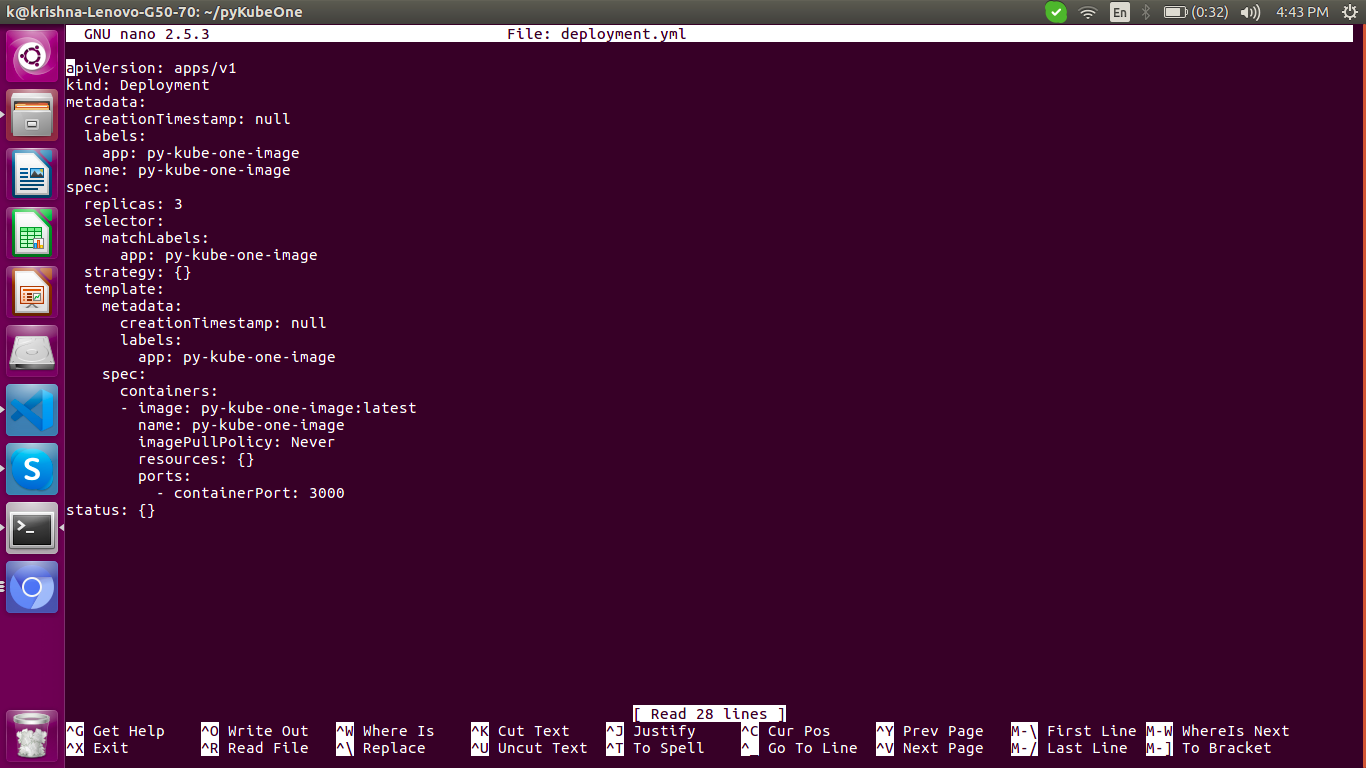
imagePullPolicy: Never

resources: {}

ports:

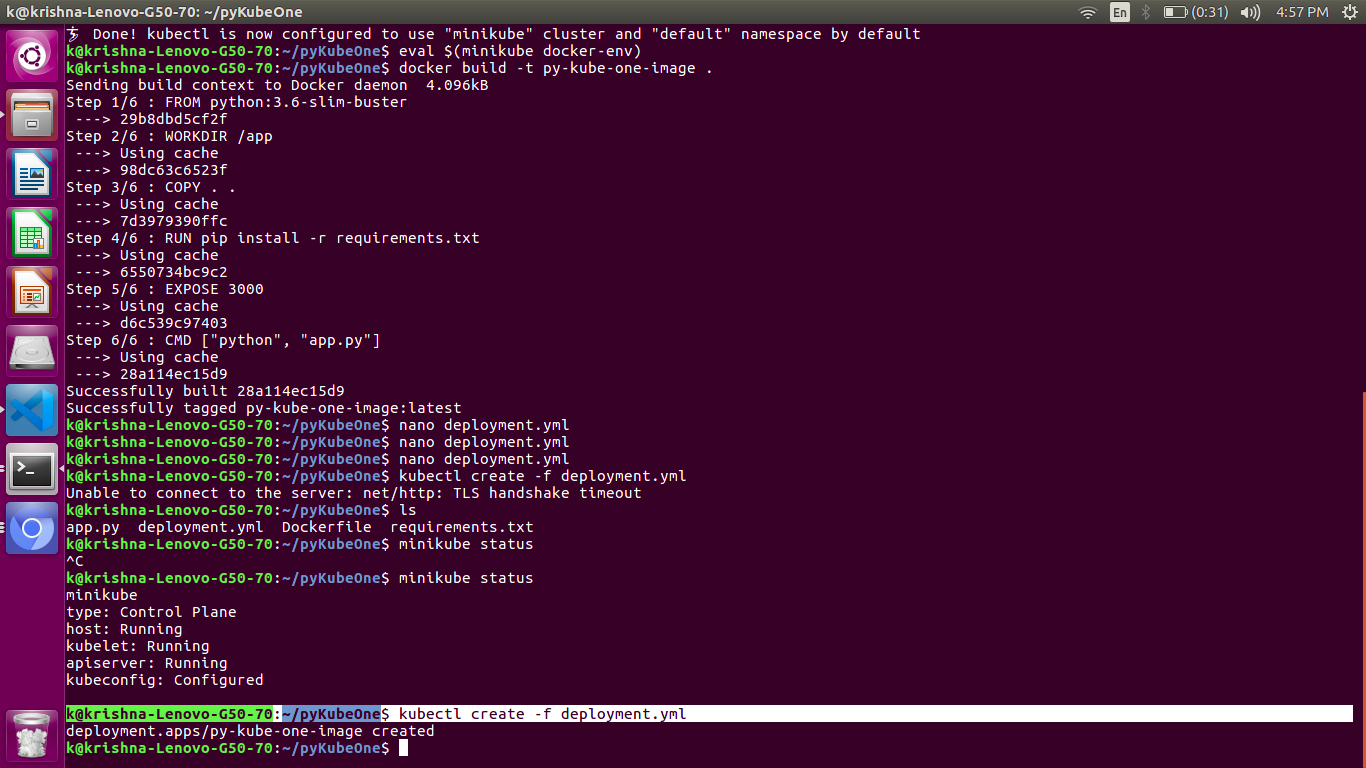
- containerPort: 3000

status: {}



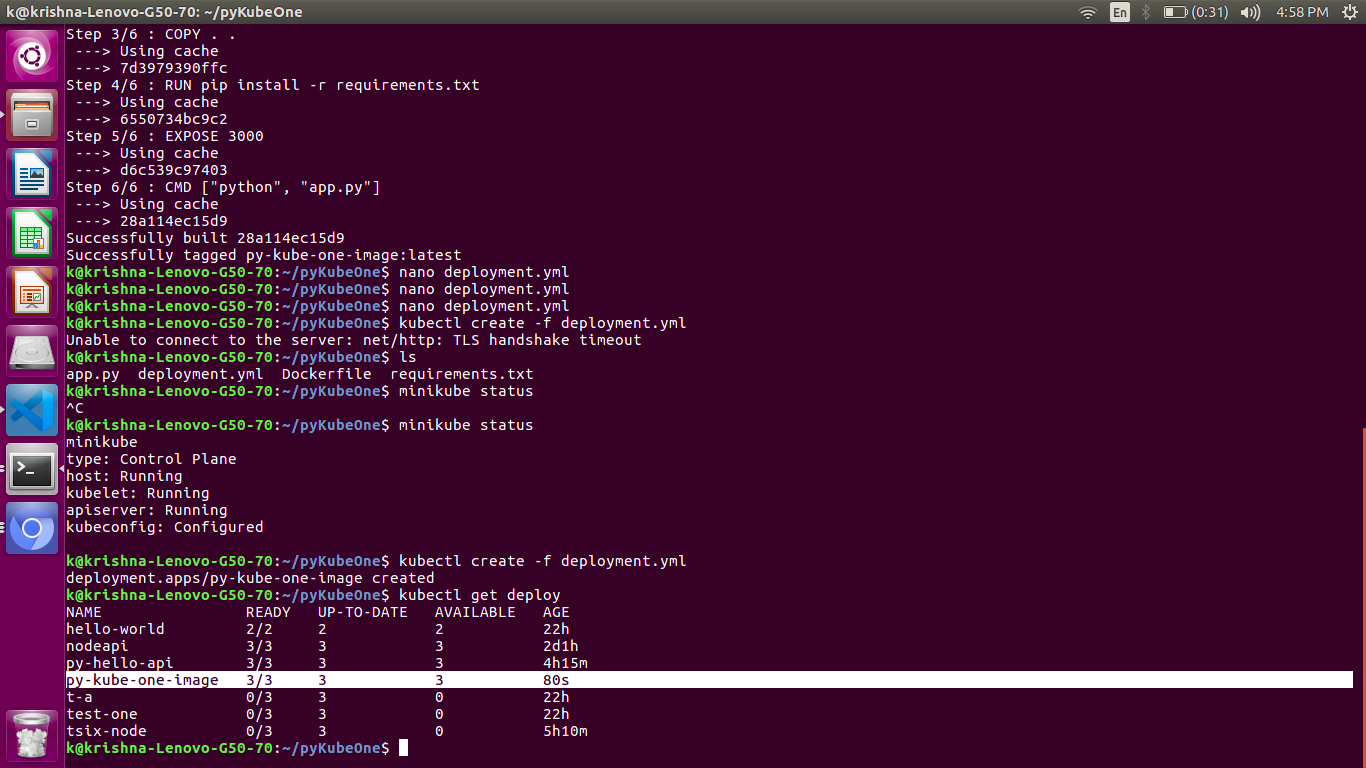
8. Create a deployment with command:

k@krishna-Lenovo-G50-70:~/pyKubeOne$ **kubectl create -f deployment.yml**



9. We can verify the deployment as :

k@krishna-Lenovo-G50-70:~/pyKubeOne$ **kubectl get deploy**



10. Create a service.yml file to expose these 3 instances to outside world.

The content of service.yml is:

apiVersion: v1

kind: Service

metadata:

name: py-kube-one

labels:

service: py-kube-one

spec:

selector:

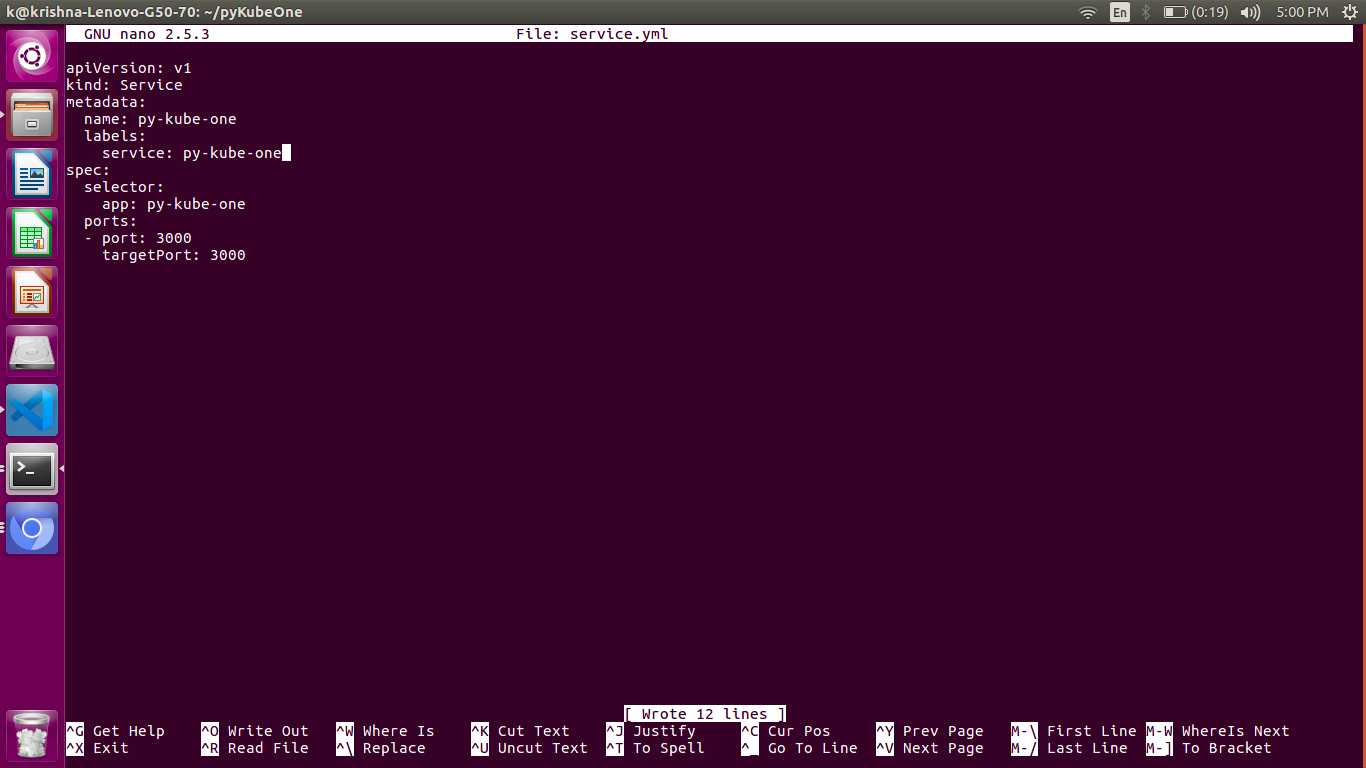
app: py-kube-one

ports:

- port: 3000

targetPort: 3000

k@krishna-Lenovo-G50-70:~/pyKubeOne$ **kubectl create -f service.yml**



11. Now, we will be able to access the pods from the local system with:

k@krishna-Lenovo-G50-70:~/pyKubeOne$ **kubectl get svc**

Create a minikube-ingress.yml as:

k@krishna-Lenovo-G50-70:~/pyKubeOne$ **nano minikube-ingress.yml**

The content of file is:

apiVersion: extensions/v1beta1

kind: Ingress

metadata:

name: minikube-ingress

annotations:

spec:

rules:

- host: hello.world

http:

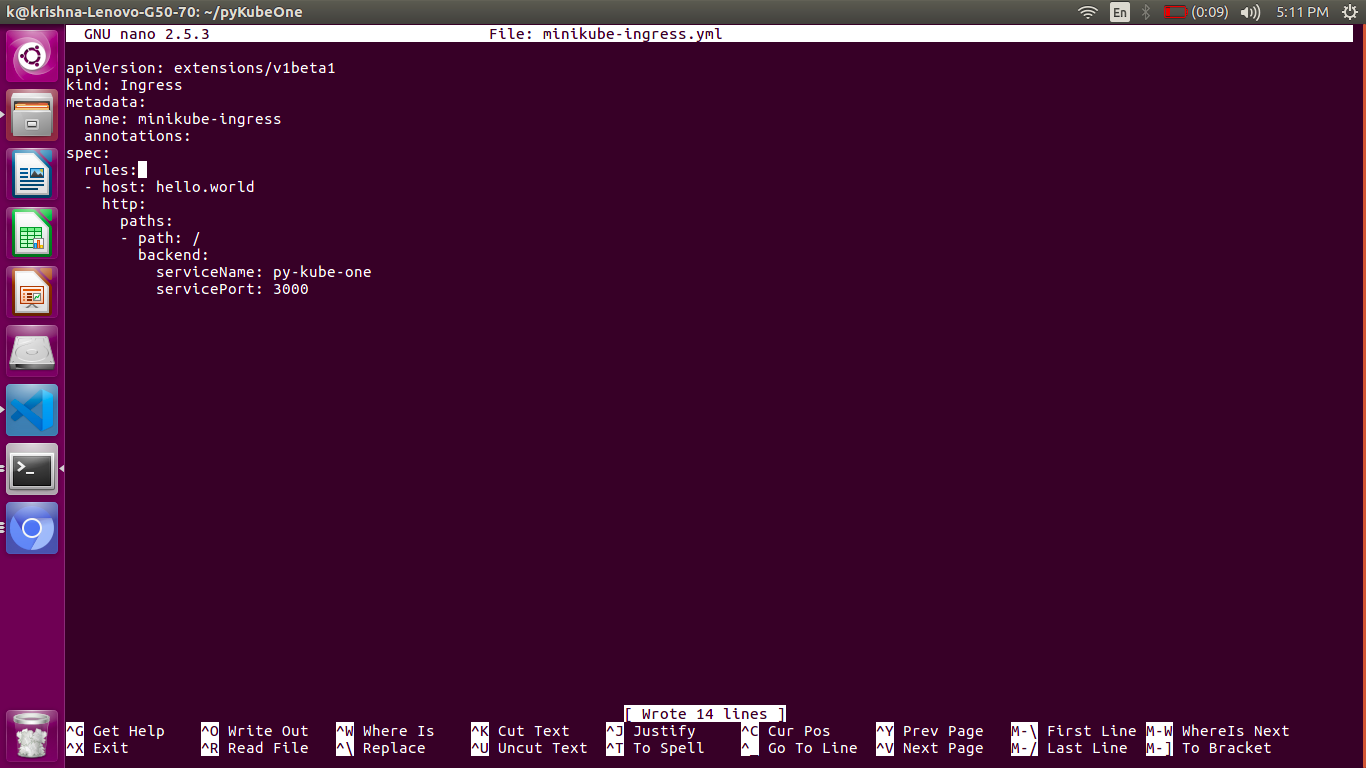
paths:

- path: /

backend:

serviceName: py-kube-one

servicePort: 3000



k@krishna-Lenovo-G50-70:~/pyKubeOne$ **minikube addons enable ingress**

🔎 Verifying ingress addon...

🌟 The 'ingress' addon is enabled

k@krishna-Lenovo-G50-70:~/pyKubeOne$ **kubectl apply -f minikube-ingress.yml**

Warning: extensions/v1beta1 Ingress is deprecated in v1.14+, unavailable in v1.22+; use networking.k8s.io/v1 Ingress

ingress.extensions/minikube-ingress configured

12.

k@krishna-Lenovo-G50-70:~/pyKubeOne$ **echo "$(minikube ip) hello.world" | sudo tee -a /etc/hosts**

[sudo] password for k:

192.168.99.102 hello.world

13.

k@krishna-Lenovo-G50-70:~/pyKubeOne$ **curl http://hello.world**

