Python Microservice within Kubernetes

Created by Craig Brown, last modified on Dec 05, 2022

Installation & Setup

- Install Kubernetes in previous page on Kubernetes
- Python can be installed from download page - https://www.python.org/downloads/. Though easy to install from HomeBrew for Mac. (Can't remember which way I installed)

Local Setup

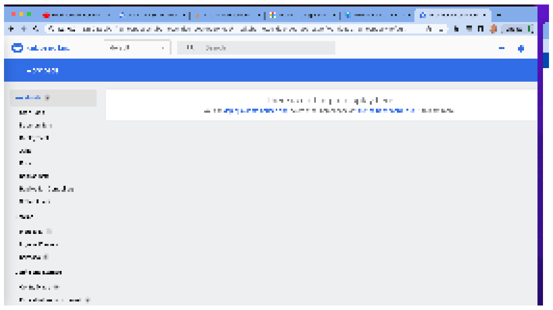
Start your cluster

From a terminal with administrator access (but not logged in as root), run:

```
minikube start
```

For the insight of the minikube run the following cmd and following the instructions

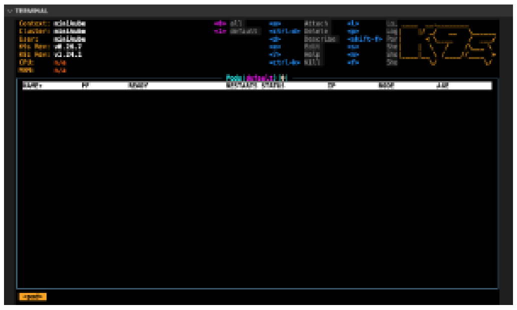
```
minikube dashboard
```



Manage your cluster

Its easier to manage your cluster via k9s. Run

```
k9s
```



Create a new namespace

This will create a new namespace, `test`, where the service can be deployed too. If no namespace is provide then they will be located to the default namespace

```
kubectl create namespace test
```

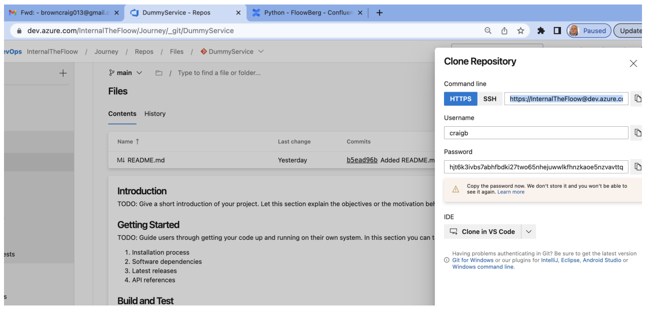
Dummy/MVP Project

Have created a new Azure DevOps Project "Journey" and created the new repository. https://dev.azure.com/InternalTheFloow/Journey/_git/DummyService.

Can clone the repo locally by running the following:

```
git clone https://InternalTheFloow@dev.azure.com/InternalTheFloow/Journey/_git/DummyService
```

May need to generate a password and insert on request.


Fwd: - browncraig013@gmail.c
DummyService - Repos
Python - FloowBerg - Confluen
dev.azure.com/InternalTheFloow/Journey/_git/DummyService
Paused
Update
evOps InternalTheFloow / Journey / Repos / Files / DummyService
main
Type to find a file or folder...
Files
Contents
History
ests
s

Name	Last change	Commits
README.md	Yesterday	b5ead96b Added README.m

Introduction

TODO: Give a short introduction of your project. Let this section explain the objectives or the motivation beh

Getting Started

TODO: Guide users through getting your code up and running on their own system. In this section you can t

1. Installation process
2. Software dependencies
3. Latest releases
4. API references

Build and Test

Clone Repository

Command line: HTTPS SSH https://InternalTheFloow@dev.azure.c

Username: craigb

Password: hjt6k3ivbs7abhfbdki27two65nhejuwwlkfhnzkaoe5nzvavttq

Copy the password now. We don't store it and you won't be able to see it again. Learn more

IDE: Clone in VS Code

Having problems authenticating in Git? Be sure to get the latest version Git for Windows or our plugins for IntelliJ, Eclipse, Android Studio or Windows command line.

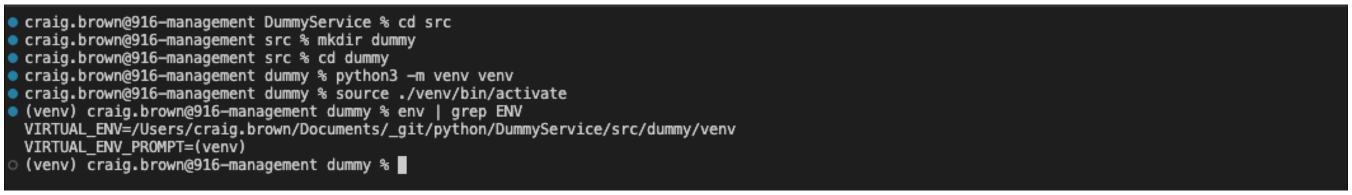
Create a service

Create a directory

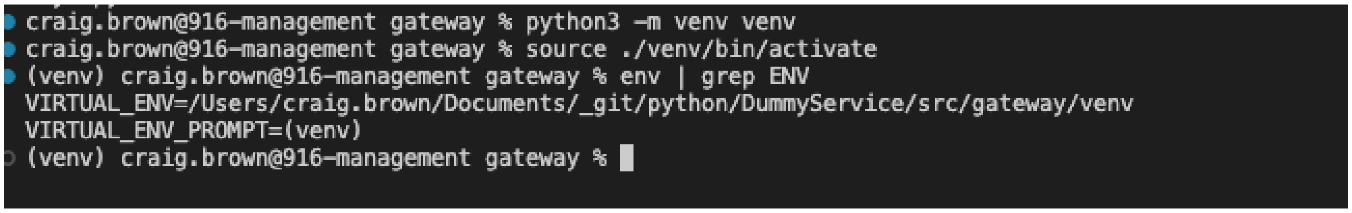
```
mkdir dummy
cd dummy
```

Make a python virtual environment as normal and then run our virtual environment

```
python3 -m venv venv
source ./venv/bin/activate
env | grep ENV
```



```
craig.brown@916-management DummyService % cd src
craig.brown@916-management src % mkdir dummy
craig.brown@916-management src % cd dummy
craig.brown@916-management dummy % python3 -m venv venv
craig.brown@916-management dummy % source ./venv/bin/activate
(venv) craig.brown@916-management dummy % env | grep ENV
VIRTUAL_ENV=/Users/craig.brown/Documents/_git/python/DummyService/src/dummy/venv
VIRTUAL_ENV_PROMPT=(venv)
(venv) craig.brown@916-management dummy %
```



```
craig.brown@916-management gateway % python3 -m venv venv
craig.brown@916-management gateway % source ./venv/bin/activate
(venv) craig.brown@916-management gateway % env | grep ENV
VIRTUAL_ENV=/Users/craig.brown/Documents/_git/python/DummyService/src/gateway/venv
VIRTUAL_ENV_PROMPT=(venv)
(venv) craig.brown@916-management gateway %
```

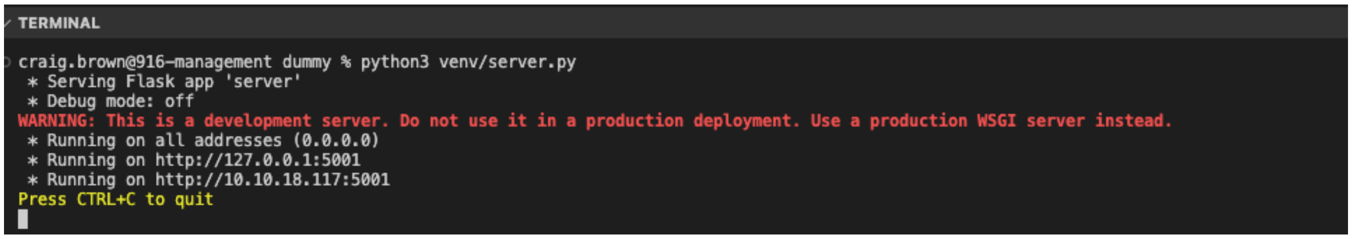
Create a server file for python under the directory called server.py

Install some useful dependancy by running the pip3 install command

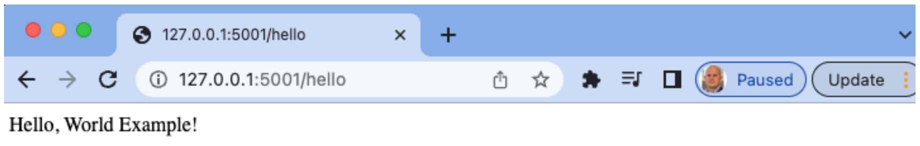
- pip3 install jedi
- pip3 install pylint
- pip3 install flask

Can run the service

```
python3 venv/server.py
```



```
TERMINAL
craig.brown@916-management dummy % python3 venv/server.py
 * Serving Flask app 'server'
 * Debug mode: off
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
 * Running on all addresses (0.0.0.0)
 * Running on http://127.0.0.1:5001
 * Running on http://10.10.18.117:5001
Press CTRL+C to quit
```



Hello, World Example!

Docker Container for a python service

Create a docker file called Dockerfile in the service directory from "python:3.10-slim-bullseye", exposing on 5001

```
FROM python:3.10-slim-bullseye

RUN apt-get update \
  && apt-get install -y --no-install-recommends --no-install-suggests \
  build-essential \
  && pip install --no-cache-dir --upgrade pip

WORKDIR /app
COPY ./requirements.txt /app
RUN pip install --no-cache-dir --requirement /app/requirements.txt
COPY . /app

EXPOSE 5000

CMD ["python3", "server.py"]
```

To freeze the current requirements for the application run the following. This is so the docker build knows what dependency it needs to install for this appliaction

```
pip3 freeze > requirements.txt
```

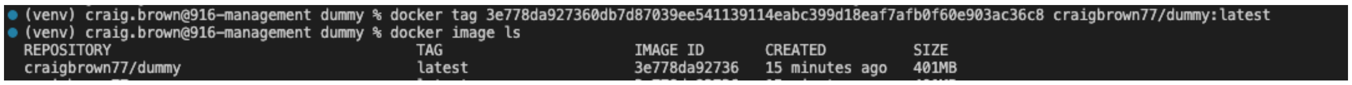
To build the docker file run

```
docker build .
```

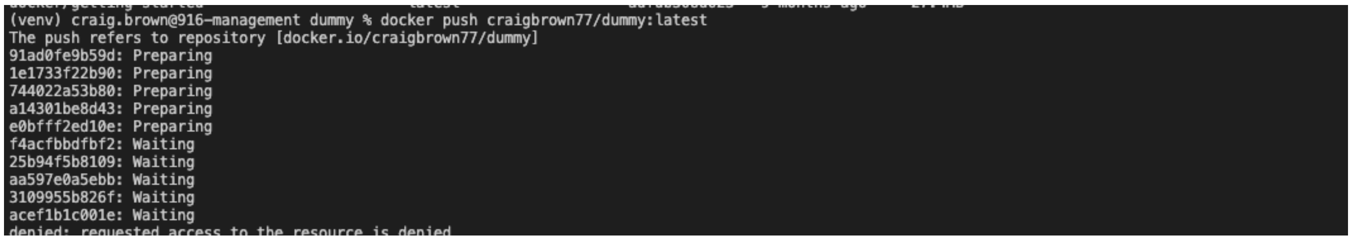
Push Docker Image - Docker Hub

Initially going to push to a Docker Hub that I have setup but will also try ECR that we have already setup on AWS in the past.

```
docker tag 3e778da927360db7d87039e craigbrown77/dummy:latest
docker image ls
docker push craigbrown77/dummy:latest
```



```
(venv) craig.brown@916-management dummy % docker tag 3e778da927360db7d87039ee541139114eabc399d18eaf7afb0f60e903ac36c8 craigbrown77/dummy:latest
(venv) craig.brown@916-management dummy % docker image ls
 REPOSITORY                          TAG                   IMAGE ID       CREATED          SIZE
 craigbrown77/dummy                  latest                3e778da92736   15 minutes ago   401MB
```



```
(venv) craig.brown@916-management dummy % docker push craigbrown77/dummy:latest
The push refers to repository [docker.io/craigbrown77/dummy]
91ad0fe9b59d: Preparing
1e1733f22b90: Preparing
744022a53b80: Preparing
a14301be8d43: Preparing
e0bfff2ed10e: Preparing
f4acfbbdfbf2: Waiting
25b94f5b8109: Waiting
aa597e0a5ebb: Waiting
3109955b826f: Waiting
acef1b1c001e: Waiting
denied: requested access to the resource is denied
```

If you get access denied will need to verify login by running

```
docker login -u "craigbrown77" -p "<password here>" docker.io
```

To pull the docker image we just need to run the following, though its our Kubernetes that will be pulling the docker image.

```
docker pull craigbrown77/dummy:latest
```

Push Docker Image - AWS ECR

https://docs.aws.amazon.com/AmazonECR/latest/userguide/docker-push-ecr-image.html

```
aws ecr get-login-password --region eu-west-1 | docker login --username AWS --password-stdin 127038058659.dkr.
ecr.eu-west-1.amazonaws.com
```

```
docker tag 3e778da927360db7d87039e 127038058659.dkr.ecr.eu-west-1.amazonaws.com/ecr-journey:
dummy___ImageRepository__
docker image ls
docker push 127038058659.dkr.ecr.eu-west-1.amazonaws.com/ecr-journey:dummy___ImageRepository__
```

The reason for __ImageRepository__ and not latest is this will get replaced with build number in the CI.

Kubernetes Config for pulling the Docker Images

Create a directory called manifest under the service (dummy) folder. this is going to contain all our Kubernetes configuration

```
mkdir manifests
cd manifests
```

Create a ymal file called dummy-deploy.yml, which will be a "Kind" Deployment (for more information see Understanding Kubernetes Objects). This will setup our Kubernetes cluster and service

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: dummy
  labels:
    app: dummy
spec:
  replicas: 2
  selector:
    matchLabels:
      app: dummy
  strategy:
    type: RollingUpdate
    rollingUpdate:
      maxSurge: 3
  template:
    metadata:
      labels:
        app: dummy
    spec:
      containers:
        - name: auth
          image: craigbrown77/dummy
          ports:
            - containerPort: 5001
          envFrom:
            - configMapRef:
                name: dummy-configmap
            - secretRef:
                name: dummy-secret
```

The envFrom will use "dummy-configmap" and "dummy-secret". There nothing is this files at the moment but place for future reference. So created configmap.yaml

```yaml
apiVersion: v1
kind: ConfigMap
metadata:
  name: dummy-configmap
data:
    PLACEHOLDER: "nothing"
```

and secret.yaml

```yaml
apiVersion: v1
kind: Secret
metadata:
  name: dummy-secret
stringData:
  PLACEHOLDER: nothing
type: Opaque
```

Will need to create "service.yml"

```yaml
apiVersion: v1
kind: Service
metadata:
  name: dummy
spec:
  selector:
    app: dummy
  type: ClusterIP
  ports:
    - port: 5001
      targetPort: 5001
      protocol: TCP
```

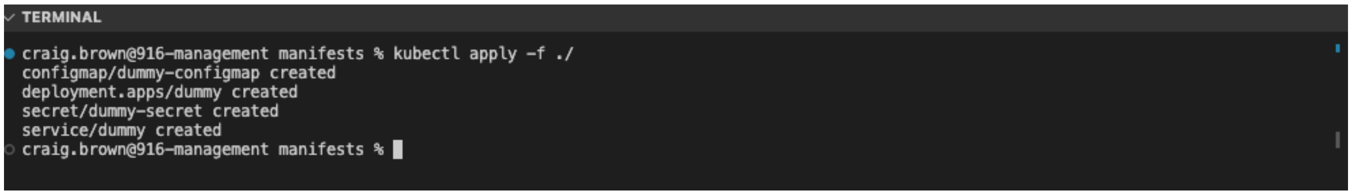
Deploy Kubernetes Local

- Need miniKube running

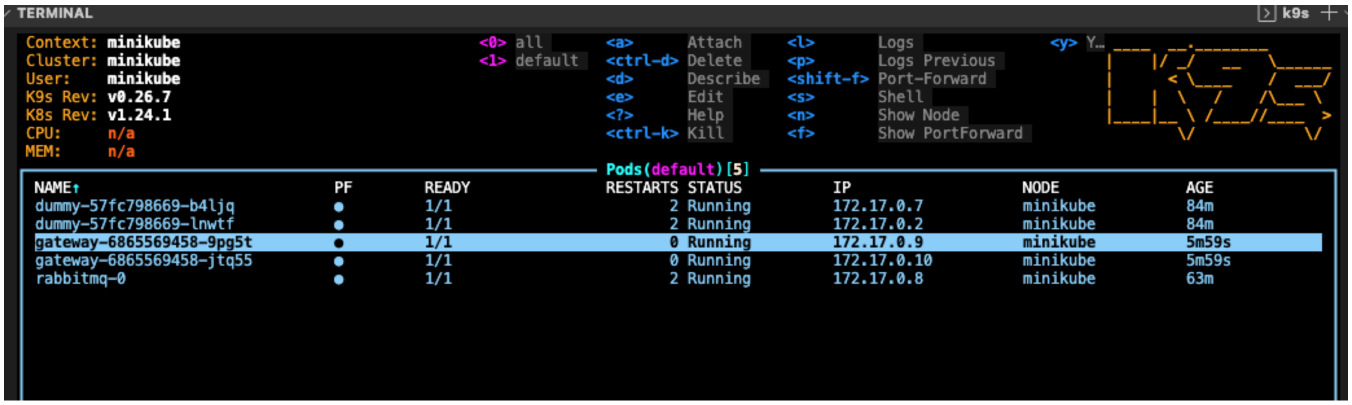
```
kubectl apply -f ./
```

To run apply to a namespace will need to include the namespace in the command or in the yaml file.

```
kubectl apply -f ./ -n test
```



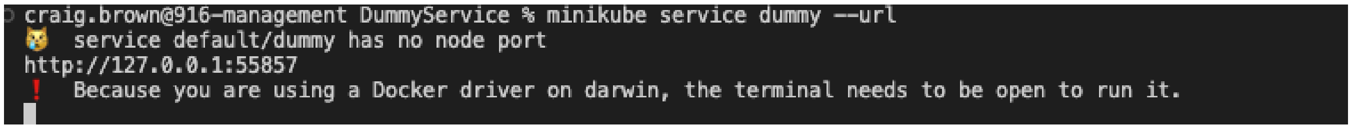
```
TERMINAL
craig.brown@916-management manifests % kubectl apply -f ./
configmap/dummy-configmap created
deployment.apps/dummy created
secret/dummy-secret created
service/dummy created
craig.brown@916-management manifests %
```



```
TERMINAL                                                                                          k9s
Context: minikube          <0> all         <a>      Attach     <l>         Logs            <y> Y…
Cluster: minikube          <1> default     <ctrl-d> Delete     <p>         Logs Previous
User:    minikube                          <d>      Describe   <shift-f>   Port-Forward
K9s Rev: v0.26.7                           <e>      Edit       <s>         Shell
K8s Rev: v1.24.1                           <?>      Help       <n>         Show Node
CPU:     n/a                               <ctrl-k> Kill       <f>         Show PortForward
MEM:     n/a
                                           Pods(default)[5]
NAME↑                          PF    READY    RESTARTS STATUS    IP           NODE        AGE
dummy-57fc798669-b4ljq         ●     1/1             2 Running   172.17.0.7   minikube    84m
dummy-57fc798669-lnwtf         ●     1/1             2 Running   172.17.0.2   minikube    84m
gateway-6865569458-9pg5t       ●     1/1             0 Running   172.17.0.9   minikube    5m59s
gateway-6865569458-jtq55       ●     1/1             0 Running   172.17.0.10  minikube    5m59s
rabbitmq-0                     ●     1/1             2 Running   172.17.0.8   minikube    63m
```

Can quickly test http service via running

```
minikube service dummy --url
```

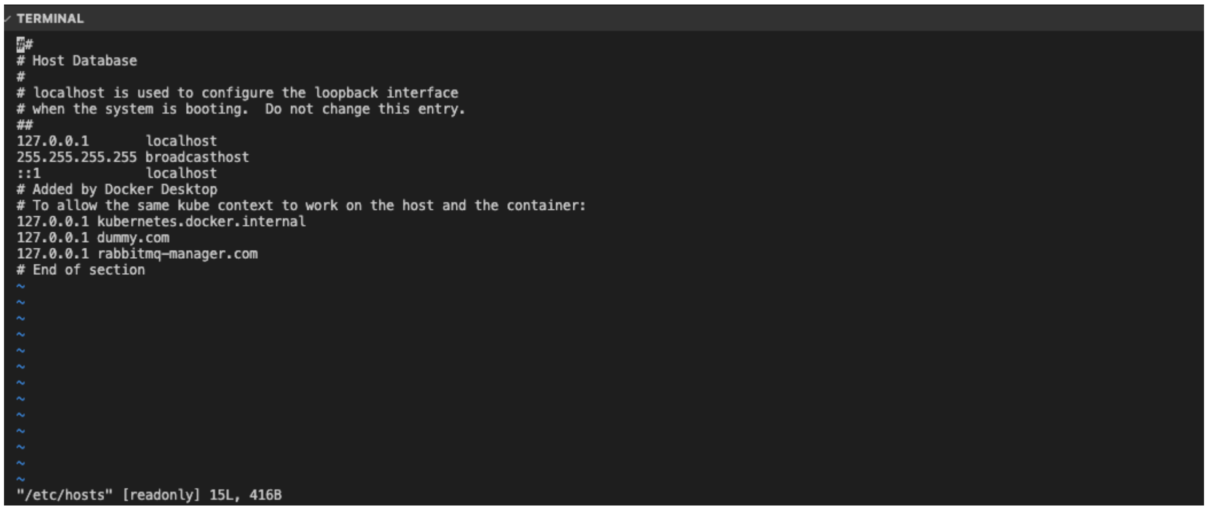


```
craig.brown@916-management DummyService % minikube service dummy --url
😿  service default/dummy has no node port
http://127.0.0.1:55857
❗  Because you are using a Docker driver on darwin, the terminal needs to be open to run it.
```

Or by forwarding the port

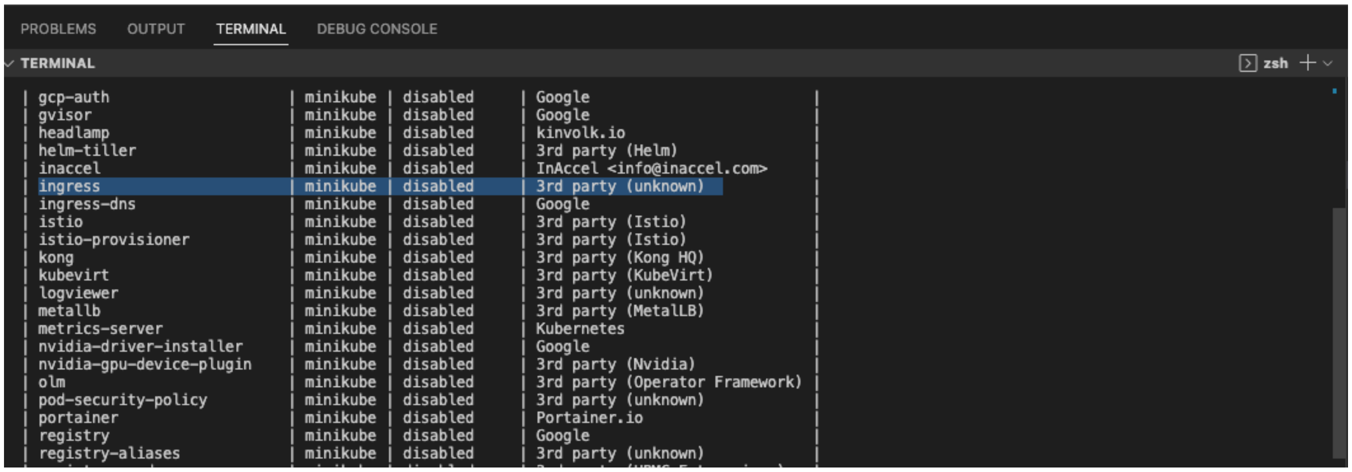
```
kubectl port-forward service/dummy 8081:5001
```

To get localhost to be route to localhost need to update the host file by running "sudo vim /etc/hosts" and adding the names



```
##
# Host Database
#
# localhost is used to configure the loopback interface
# when the system is booting.  Do not change this entry.
##
127.0.0.1       localhost
255.255.255.255 broadcasthost
::1             localhost
# Added by Docker Desktop
# To allow the same kube context to work on the host and the container:
127.0.0.1 kubernetes.docker.internal
127.0.0.1 dummy.com
127.0.0.1 rabbitmq-manager.com
# End of section
~
~
~
~
~
~
~
~
~
~
~
~
~
"/etc/hosts" [readonly] 15L, 416B
```

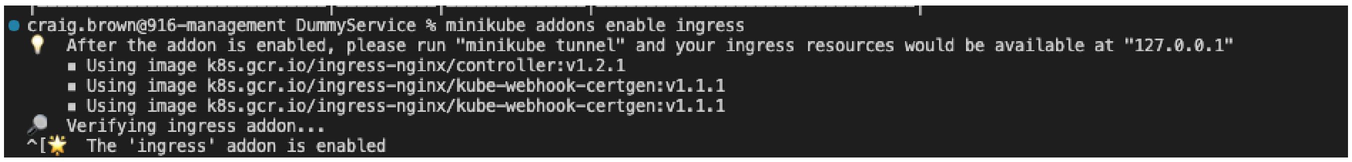
```
minikube addons list
```



```
| gcp-auth                    | minikube | disabled     | Google                          |
| gvisor                      | minikube | disabled     | Google                          |
| headlamp                    | minikube | disabled     | kinvolk.io                      |
| helm-tiller                 | minikube | disabled     | 3rd party (Helm)                |
| inaccel                     | minikube | disabled     | InAccel <info@inaccel.com>      |
| ingress                     | minikube | disabled     | 3rd party (unknown)             |
| ingress-dns                 | minikube | disabled     | Google                          |
| istio                       | minikube | disabled     | 3rd party (Istio)               |
| istio-provisioner           | minikube | disabled     | 3rd party (Istio)               |
| kong                        | minikube | disabled     | 3rd party (Kong HQ)             |
| kubevirt                    | minikube | disabled     | 3rd party (KubeVirt)            |
| logviewer                   | minikube | disabled     | 3rd party (unknown)             |
| metallb                     | minikube | disabled     | 3rd party (MetalLB)             |
| metrics-server              | minikube | disabled     | Kubernetes                      |
| nvidia-driver-installer     | minikube | disabled     | Google                          |
| nvidia-gpu-device-plugin    | minikube | disabled     | 3rd party (Nvidia)              |
| olm                         | minikube | disabled     | 3rd party (Operator Framework)  |
| pod-security-policy         | minikube | disabled     | 3rd party (unknown)             |
| portainer                   | minikube | disabled     | Portainer.io                    |
| registry                    | minikube | disabled     | Google                          |
| registry-aliases            | minikube | disabled     | 3rd party (unknown)             |
```

Will need ingress addon

```
minikube addons enable ingress
```



```
craig.brown@916-management DummyService % minikube addons enable ingress
💡  After the addon is enabled, please run "minikube tunnel" and your ingress resources would be available at "127.0.0.1"
    ▪ Using image k8s.gcr.io/ingress-nginx/controller:v1.2.1
    ▪ Using image k8s.gcr.io/ingress-nginx/kube-webhook-certgen:v1.1.1
    ▪ Using image k8s.gcr.io/ingress-nginx/kube-webhook-certgen:v1.1.1
🔎  Verifying ingress addon...
^[🌟  The 'ingress' addon is enabled
```

So when ever we want to run or test this mircoservice architecture locally, we need to run the tunnelling command. While this is running when ever we send to our loop back address (127.0.0.1), they are going to minikube via the ingress

```
minikube tunnel
```

Another option is to use ingress-dns https://minikube.sigs.k8s.io/docs/handbook/addons/ingress-dns/ but will need again sudo access to initial setup

Issues:

can check pods

```
kubectl describe pods dummy-6688688558-fmcfj -n test
```

If minikube has the issue getting no auth from ECR.

```
kubectl create secret docker-registry ecr \
  --docker-server=127038058659.dkr.ecr.eu-west-1.amazonaws.com \
  --docker-username=AWS \
  --docker-password=$(aws ecr get-login-password) \
  --namespace=kube-system
```

- https://skryvets.com/blog/2021/03/15/kubernetes-pull-image-from-private-ecr-registry/
- https://pet2cattle.com/2022/05/pull-private-image-from-ecr

OR

If connecting to an AWS ECR image, follow the step below to allow minikube to access the AWS image.

https://minikube.sigs.k8s.io/docs/tutorials/configuring_creds_for_aws_ecr/

Resources:

- https://otonomoio-my.sharepoint.com/:p:/g/personal/craigb_otonomo_io/EUBzHxmDqMNLsOGB_Te2rHUBslWrFKB09ckHs2Rl2s05dw?e=KKDaew

Useful Tuition Video