**Hello Fresh**

**Devops-Test Documentation**

**Step 1: Installation and setup for development**

1. **Install minikube and hyper-V on local windows 11 machine.**
2. **Install docker for windows**
3. **Install Nginx controller on minikube.**
4. **Install python 3.10.2 on local machine.**

**Step 2: Build rest API using python.**

* **Used python Kubernetes client to read kubernetes configuration**
* **Used Flask for REST API development**

Screenshot 1: app.py

from flask import Flask,jsonify,request,render\_template,Response

from kubernetes import client, config

from kubernetes.client import ApiClient

from kubernetes.client.rest import ApiException

**Step 3: Create Docker File to create docker image for the application**

Screenshot 2: Dockerfile

FROM python:3.10.2-alpine

## Multi commands in same line to Minimize the Number of Layers

RUN pip install flask && pip install kubernetes

## for better caching

# Recommended to add the lines which are used for installing dependencies & packages earlier inside the Dockerfile

RUN mkdir /app

ADD . /app

WORKDIR /app

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

**Step 4: Write k8s yaml file to create k8s objects**

Screenshot 3: k8s/devops-test.yaml

---

# create namespace

apiVersion: v1

kind: Namespace

metadata:

  name: devops-test

---

# config map for the container app to read port number

apiVersion: v1

kind: ConfigMap

metadata:

  name: port-config

  namespace: devops-test

data:

  SERVE\_PORT: '5001'

---

# create service account

apiVersion: v1

kind: ServiceAccount

metadata:

  name: mypython-app-sa

  namespace: devops-test

---

## Create clusterRole and ClusterRoleBinding with required permission.

kind: ClusterRole

apiVersion: rbac.authorization.k8s.io/v1

metadata:

  name: k8s-cluster-role

  namespace: devops-test

rules:

    - apiGroups:

        - ""

        - "apps"

        - "batch"

        - "extensions"

      resources:

        - "configmaps"

        - "cronjobs"

        - "deployments"

        - "events"

        - "ingresses"

        - "jobs"

        - "pods"

        - "pods/attach"

        - "pods/exec"

        - "pods/log"

        - "pods/portforward"

        - "secrets"

        - "services"

      verbs:

        - "create"

        - "delete"

        - "describe"

        - "get"

        - "list"

        - "patch"

        - "update"

---

kind: ClusterRoleBinding

apiVersion: rbac.authorization.k8s.io/v1

metadata:

  name: k8s-cluster-role

  namespace: devops-test

subjects:

- kind: ServiceAccount

  name: mypython-app-sa

  namespace: devops-test

roleRef:

  kind: ClusterRole

  name: k8s-cluster-role

  apiGroup: rbac.authorization.k8s.io

---

# Create deployment with 2 replicas

apiVersion: apps/v1

kind: Deployment

metadata:

  name: mypython-app-rest-deployment

  namespace: devops-test

spec:

  replicas: 2

  selector:

    matchLabels:

      component: mypython-app-rest

  template:

    metadata:

      labels:

        component: mypython-app-rest

    spec:

      serviceAccountName: mypython-app-sa

      containers:

        - name: mypython-app-rest

          image: vinaykumar6/vinay-devops-test

          ports:

            - containerPort: 5001

          env:

            - name: SERVE\_PORT

              valueFrom:

                configMapKeyRef:

                  name: port-config

                  key: SERVE\_PORT

          livenessProbe:

            httpGet:

              path: /healthz

              port: 5001

            initialDelaySeconds: 50

            periodSeconds: 30

---

# Cluster IP service to direct traffic to pod using labels and selectors

apiVersion: v1

kind: Service

metadata:

  name: mypython-app-rest-cluster-ip-service

  namespace: devops-test

spec:

  type: ClusterIP

  selector:

    component: mypython-app-rest

  ports:

    - port: 5001

      targetPort: 5001

---

# Ingress route to allow traffic inside the cluster

apiVersion: networking.k8s.io/v1

kind: Ingress

metadata:

  name: ingress-service

  namespace: devops-test

  annotations:

    kubernetes.io/ingress.class: 'nginx'

    nginx.ingress.kubernetes.io/use-regex: 'true'

    nginx.ingress.kubernetes.io/service-upstream: 'true'

spec:

  rules:

    - http:

       paths:

        - path: /configs/\*

          pathType: Prefix

          backend:

            service:

              name: mypython-app-rest-cluster-ip-service

              port:

                number: 5001

        - path: /config-service/\*

          pathType: Prefix

          backend:

            service:

              name: mypython-app-rest-cluster-ip-service

              port:

                number: 5001

**Step 5: Write a bash script to create docker image and deploy to minikube cluster**

Screenshot 4: deploy.sh

#! /bin/bash

## build docker image and tag it

echo " ==> Building docker image"

docker build -t vinaykumar6/vinay-devops-test .

## login to docker hub

## Username and password not mentioned, fetched from Windows credential manager

echo " ==> Before docker login"

docker login

# push image to docker hub repository

echo " ==> push image to docker hub"

docker push vinaykumar6/vinay-devops-test

## k8s will pull image form docker hub and deploy application

echo " ==> Apply k8s yaml file to the cluster"

if [[ $(kubectl apply -f k8s | grep -c 'deployment unchanged') == 1 ]]; then

    echo " ==> Patching deployment to pull latest image from docker hub"

    kubectl rollout restart deployment mypython-app-rest-deployment -n devops-test

    echo " ==> success"

else

    echo " ==> k8s objects created. deployment patch not required"

fi

**Step 6: verify k8s objects**

Screenshot 5: kubectl command from PowerShell

**Graphical user interface

Description automatically generated**

Screenshot 6: minikube command from PowerShell

**Graphical user interface, text

Description automatically generated with medium confidence**

**Step 7: Unit Testing using Soap UI(Test cases – 10)**

|  |  |  |  |
| --- | --- | --- | --- |
| **TEST CASE** | **OPERATION** | **REQUEST** | **RESPONSE** |
| TEST\_01  Create a config map in minikube cluster in default namespace, expected response is 200 OK. | POST  http://172.21.43.230:80/configs | **{**  **"name" : "my-first-config-map",**  **"namespace" : "default",**  **"metadata": {**  **"key" : "test-key-1-patched",**  **"value" : "value-1-patched"**  **}**  **}** | **HTTP/1.1 200 OK**  **Date: Sat, 05 Mar 2022 10:00:41 GMT**  **Content-Type: application/json**  **Content-Length: 38**  **Connection: keep-alive**  **{**  **"message": "config map created"**  **}** |

TEST\_01 SoapUI:

Graphical user interface, text, application

Description automatically generated

Verify in minikube cluster

Graphical user interface, text

Description automatically generated

|  |  |  |  |
| --- | --- | --- | --- |
| **TEST CASE** | **OPERATION** | **REQUEST** | **RESPONSE** |
| TEST\_02  Try creating an existing configmap. It should return an error with response code 409 CONFLICT | POST  http://172.21.43.230:80/configs | **{**  **"name" : "my-first-config-map",**  **"namespace" : "default",**  **"metadata": {**  **"key" : "test-key-1-patched",**  **"value" : "value-1-patched"**  **}**  **}** | **HTTP/1.1 409 CONFLICT**  **Date: Sat, 05 Mar 2022 10:03:01 GMT**  **Content-Type: application/json**  **Content-Length: 43**  **Connection: keep-alive**  **{**  **"error": "config map already exists"**  **}** |

TEST\_02 SoapUI:

Graphical user interface, text, application

Description automatically generated

|  |  |  |  |
| --- | --- | --- | --- |
| **TEST CASE** | **OPERATION** | **REQUEST** | **RESPONSE** |
| TEST\_03  Update an existing configmap. It should return success with http status code 200 OK | PATCH  http://172.21.43.230:80/configs/{cmap\_name} | **{**  **"metadata": {**  **"key" : "test-key-1-patched",**  **"value" : "value-1-patched"**  **}**  **}** | **HTTP/1.1 200 OK**  **Date: Sat, 05 Mar 2022 10:06:58 GMT**  **Content-Type: application/json**  **Content-Length: 52**  **Connection: keep-alive**  **{**  **"message": "config map updated succcessfully"**  **}** |

TEST\_03 SoapUI:

**Graphical user interface, text, application

Description automatically generated**

Verify in minikube cluster

**Graphical user interface, text, application

Description automatically generated**

|  |  |  |  |
| --- | --- | --- | --- |
| **TEST CASE** | **OPERATION** | **REQUEST** | **RESPONSE** |
| TEST\_04  Update configmap that does not exist. It should return error with http status code 404 NOT FOUND | PATCH  [http://172.21.43.230:80/configs/{cmap\_name}](http://172.21.43.230:80/configs/%7bcmap_name%7d)  cmap\_name=test-123 | **{**  **"metadata": {**  **"key" : "test-key-1-patched",**  **"value" : "value-1-patched"**  **}**  **}** | **HTTP/1.1 404 NOT FOUND**  **Date: Sat, 05 Mar 2022 14:49:05 GMT**  **Content-Type: application/json**  **Content-Length: 38**  **Connection: keep-alive**  **{**  **"error": "config map not found"**  **}** |

TEST\_04 SoapUI:

**Graphical user interface, text, application

Description automatically generated**

|  |  |  |  |
| --- | --- | --- | --- |
| **TEST CASE** | **OPERATION** | **REQUEST** | **RESPONSE** |
| TEST\_05  Get details of configmap by passing configmap name in path parameter | GET  [http://172.21.43.230:80/configs/{cmap\_name}](http://172.21.43.230:80/configs/%7bcmap_name%7d)  cmap\_name=my-first-config-map | cmap\_name=my-first-config-map | **HTTP/1.1 200 OK**  **Date: Sat, 05 Mar 2022 15:06:08 GMT**  **Content-Type: application/json**  **Content-Length: 583**  **Connection: keep-alive**  **{"apiVersion": "v1", "data": {"key": "test-key-2", "value": "value-2"}, "kind": "ConfigMap", "metadata": {"creationTimestamp": "2022-03-05T08:51:54+00:00", "managedFields": [{"apiVersion": "v1", "fieldsType": "FieldsV1", "fieldsV1": {"f:data": {".": {}, "f:key": {}, "f:value": {}}}, "manager": "OpenAPI-Generator", "operation": "Update", "time": "2022-03-05T08:51:54+00:00"}], "name": "my-second-config-map", "namespace": "default", "resourceVersion": "72723", "selfLink": "/api/v1/namespaces/default/configmaps/my-second-config-map", "uid": "d136fb03-fdce-45a3-8c59-1a9fe8e3ac96"}}** |

TEST\_05 SoapUI:

**Graphical user interface, text, application

Description automatically generated**

|  |  |  |  |
| --- | --- | --- | --- |
| **TEST CASE** | **OPERATION** | **REQUEST** | **RESPONSE** |
| TEST\_06  Get details of configmap by passing configmap name in path parameter. Pass the configmap name that does not exist. It should return error 404 NOT FOUND | GET  [http://172.21.43.230:80/configs/{cmap\_name}](http://172.21.43.230:80/configs/%7bcmap_name%7d) | cmap\_name=my-first-config-map-na | **HTTP/1.1 404 NOT FOUND**  **Date: Sat, 05 Mar 2022 14:54:13 GMT**  **Content-Type: application/json**  **Content-Length: 38**  **Connection: keep-alive**  **{**  **"error": "config map not found"**  **}** |

TEST\_06 SoapUI:

**Graphical user interface, text, application, Word

Description automatically generated**

|  |  |  |  |
| --- | --- | --- | --- |
| **TEST CASE** | **OPERATION** | **REQUEST** | **RESPONSE** |
| TEST\_07  Get list of all configmap from the cluster in default namespace. It should return success with http response code 200 OK | GET  <http://172.21.43.230:80/configs> | NA | **HTTP/1.1 200 OK**  **Date: Sat, 05 Mar 2022 09:31:15 GMT**  **Content-Type: application/json**  **Content-Length: 1795**  **Connection: keep-alive**  **{"apiVersion": "v1", "items": [{"data": {"key": "test-key-1-patched", "value": "value-1-patched"}, "metadata": {"creationTimestamp": "2022-03-05T08:37:00+00:00", "managedFields": [{"apiVersion": "v1", "fieldsType": "FieldsV1", "fieldsV1": {"f:data": {".": {}, "f:key": {}, "f:value": {}}}, "manager": "OpenAPI-Generator", "operation": "Update", "time": "2022-03-05T09:24:10+00:00"}], "name": "my-first-config-map", "namespace": "default", "resourceVersion": "74571", "selfLink": "/api/v1/namespaces/default/configmaps/my-first-config-map", "uid": "1e411ac1-0801-4c72-b857-602217222f25"}}, {"data": {"key": "test-key-2", "value": "value-2"}, "metadata": {"creationTimestamp": "2022-03-05T08:51:54+00:00", "managedFields": [{"apiVersion": "v1", "fieldsType": "FieldsV1", "fieldsV1": {"f:data": {".": {}, "f:key": {}, "f:value": {}}}, "manager": "OpenAPI-Generator", "operation": "Update", "time": "2022-03-05T08:51:54+00:00"}], "name": "my-second-config-map", "namespace": "default", "resourceVersion": "72723", "selfLink": "/api/v1/namespaces/default/configmaps/my-second-config-map", "uid": "d136fb03-fdce-45a3-8c59-1a9fe8e3ac96"}}, {"data": {"key": "test-key-3", "value": "value-3"}, "metadata": {"creationTimestamp": "2022-03-05T08:52:12+00:00", "managedFields": [{"apiVersion": "v1", "fieldsType": "FieldsV1", "fieldsV1": {"f:data": {".": {}, "f:key": {}, "f:value": {}}}, "manager": "OpenAPI-Generator", "operation": "Update", "time": "2022-03-05T08:52:12+00:00"}], "name": "my-third-config-map", "namespace": "default", "resourceVersion": "72741", "selfLink": "/api/v1/namespaces/default/configmaps/my-third-config-map", "uid": "8108a586-0894-4485-9e81-46c41da7b492"}}], "kind": "ConfigMapList", "metadata": {"resourceVersion": "74978", "selfLink": "/api/v1/namespaces/default/configmaps"}}** |

TEST\_07 SoapUI:

**Graphical user interface, text, application

Description automatically generated**

|  |  |  |  |
| --- | --- | --- | --- |
| **TEST CASE** | **OPERATION** | **REQUEST** | **RESPONSE** |
| TEST\_08  Delete an existing configmap. It should return 204 NO CONTENT | DELETE  [http://172.21.43.230:80/configs/{cmap\_name}](http://172.21.43.230:80/configs/%7bcmap_name%7d)  cmap\_name=my-first-config-map | **{}** | **HTTP/1.1 204 NO CONTENT**  **Date: Sat, 05 Mar 2022 14:59:46 GMT**  **Content-Type: application/json**  **Connection: keep-alive** |

TEST\_08 SoapUI:

**Graphical user interface, text, application

Description automatically generated**

Verify in minikube cluster

**A screenshot of a computer

Description automatically generated with medium confidence**

|  |  |  |  |
| --- | --- | --- | --- |
| **TEST CASE** | **OPERATION** | **REQUEST** | **RESPONSE** |
| TEST\_09  Fetch a deleted configmap. It should return 404 NOT FOUND | DELETE  [http://172.21.43.230:80/configs/{cmap\_name}](http://172.21.43.230:80/configs/%7bcmap_name%7d)  cmap\_name=my-first-config-map | **{}** | **HTTP/1.1 404 NOT FOUND**  **Date: Sat, 05 Mar 2022 15:01:20 GMT**  **Content-Type: application/json**  **Content-Length: 38**  **Connection: keep-alive**  **{**  **"error": "config map not found"**  **}** |

TEST\_09 SoapUI:

**Graphical user interface, application

Description automatically generated**

|  |  |  |  |
| --- | --- | --- | --- |
| **TEST CASE** | **OPERATION** | **REQUEST** | **RESPONSE** |
| TEST\_10  Search a configmap based on key passed as Query parameter. It should return 200 OK | GET  http://172.21.43.230:80 /config-service/search ?metadata=my-second-config-map | **NA** | **HTTP/1.1 200 OK**  **Date: Sat, 05 Mar 2022 15:03:25 GMT**  **Content-Type: application/json**  **Content-Length: 695**  **Connection: keep-alive**  **{"apiVersion": "v1", "items": [{"data": {"key": "test-key-2", "value": "value-2"}, "metadata": {"creationTimestamp": "2022-03-05T08:51:54+00:00", "managedFields": [{"apiVersion": "v1", "fieldsType": "FieldsV1", "fieldsV1": {"f:data": {".": {}, "f:key": {}, "f:value": {}}}, "manager": "OpenAPI-Generator", "operation": "Update", "time": "2022-03-05T08:51:54+00:00"}], "name": "my-second-config-map", "namespace": "default", "resourceVersion": "72723", "selfLink": "/api/v1/namespaces/default/configmaps/my-second-config-map", "uid": "d136fb03-fdce-45a3-8c59-1a9fe8e3ac96"}}], "kind": "ConfigMapList", "metadata": {"resourceVersion": "90477", "selfLink": "/api/v1/namespaces/default/configmaps"}}** |

TEST\_10 SoapUI:

**Graphical user interface, text, application

Description automatically generated**