CAPSTONE PROJECT REPORT

TITLE:

Microservices Orchestration and Management with Kubernetes

https://github.com/mano257200/capstone-final.git

BY:

Emmanuel Louis (DEVOPS)

Mahendravarman R (CyberSecurity)

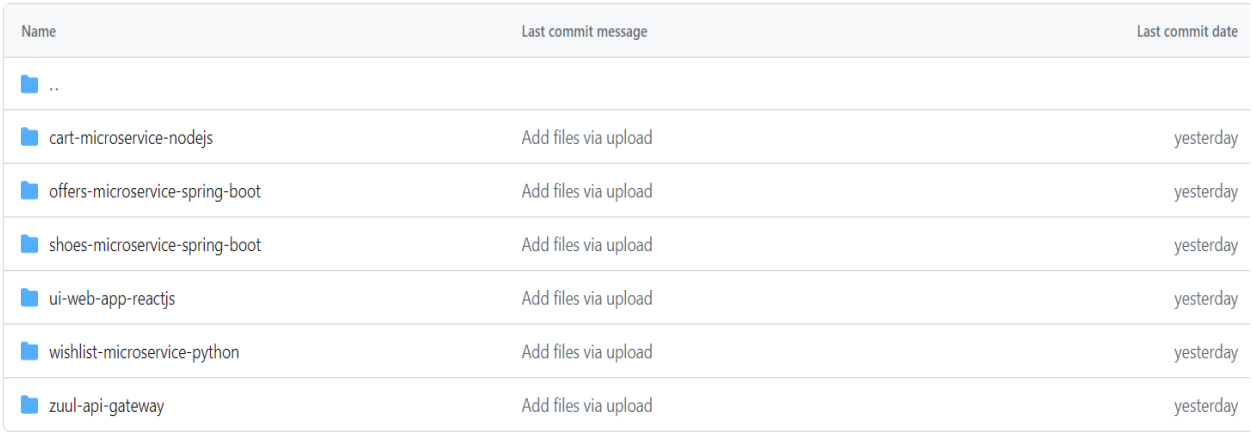
Microservices Orchestration and Management with Kubernetes

Problem Statement: "TechSolutions" is modernizing its application infrastructure by moving to a microservices architecture. Your challenge is to deploy and manage their set of microservices using Kubernetes, ensuring resilience, scalability, and efficient inter-service communication.

Dataset: A set of interconnected microservices applications available on GitHub, simulating an e-commerce platform with services like User Management, Product Catalog, Cart, and Payment.

Step 1: Clone The Application From The Github Repo

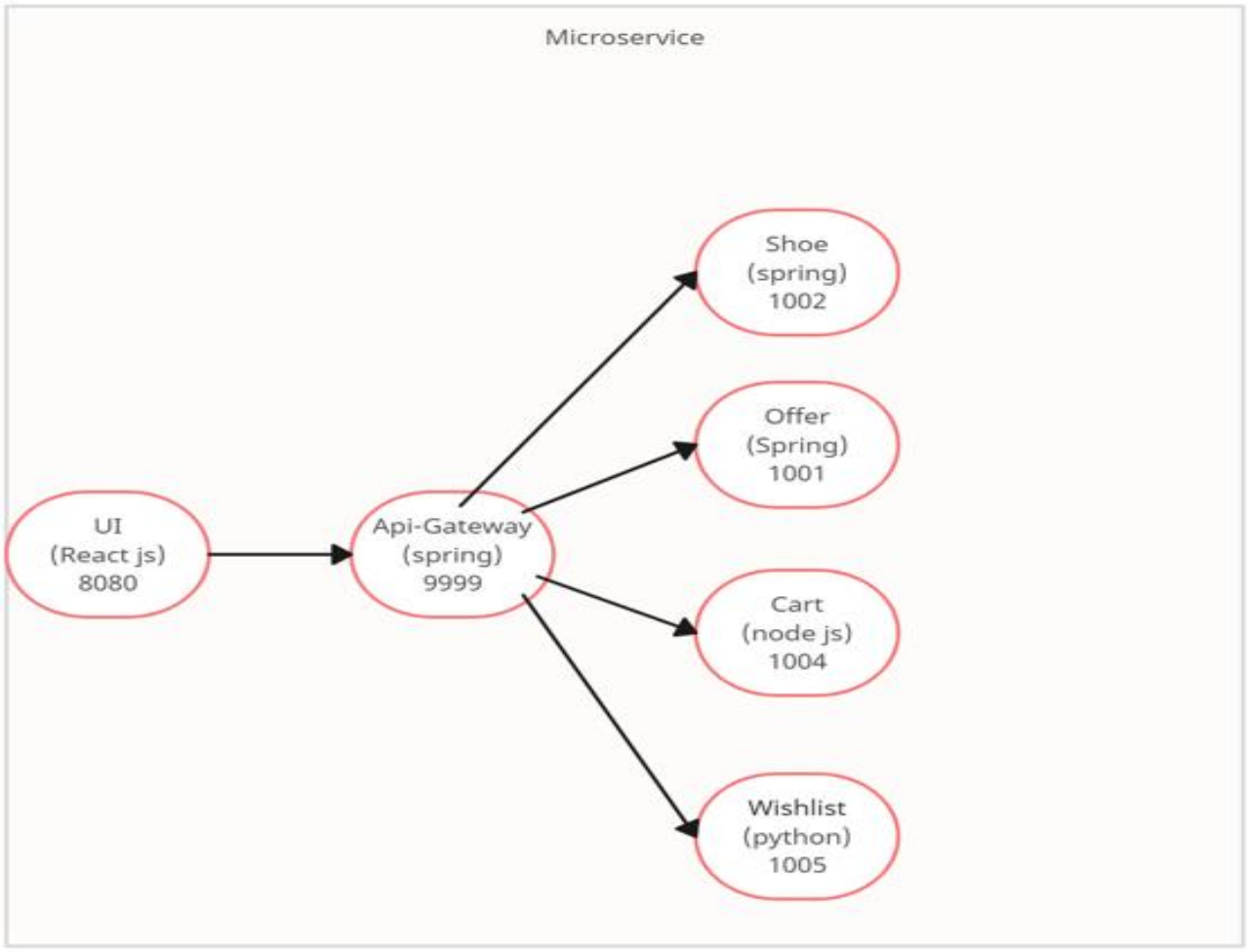
- The Github repository contains sic separate files as each files are six different microservices required for an e-commerce application.
- The microservices such as for e-commerce application user interface , api , offers , wishlist , cart , shoes .



Name	Last commit message	Last commit date
..		
cart-microservice-nodejs	Add files via upload	yesterday
offers-microservice-spring-boot	Add files via upload	yesterday
shoes-microservice-spring-boot	Add files via upload	yesterday
ui-web-app-reactjs	Add files via upload	yesterday
wishlist-microservice-python	Add files via upload	yesterday
zuul-api-gateway	Add files via upload	yesterday

Architecture of the microservice Application:

Architecture of sample multiple microservices developed in different technologies - Spring Boot, Node.js, Python, React.js in a project. Microservices connected by an API Gateway



Step 2: Containerizing The Microservices.

Containerization is a crucial step in deploying microservices using Kubernetes. It involves packaging each microservice into a container image.

Steps to containerize microservices :

- Create a Dockerfile for each microservice.
- In the Dockerfile, specify a base image that aligns with the microservice's requirements.
- Add instructions to copy the microservice's source code and any necessary dependencies into the container image.
- Define the command that should be executed when the container starts.

Containerizing e-commerce app user-interface

Dockerfile For Each Microservices:

UI (React js):

```
FROM node:8
WORKDIR /app
COPY . .
RUN npm install
RUN npm run build
EXPOSE 8080
CMD ["node", "server.js"]
```

Api (Spring):

```
FROM maven as build
WORKDIR /app
COPY . .
RUN mvn install

FROM openjdk:11.0.10-jre
WORKDIR /app
COPY --from=build /app/target/zuul-0.0.1-SNAPSHOT.jar /app
EXPOSE 8080
CMD ["java", "-jar", "zuul-0.0.1-SNAPSHOT.jar"]
```

Shoe (Spring):

```
FROM maven as build
WORKDIR /app
COPY . .
RUN mvn install

#jdk
FROM openjdk:11.0.10-jre
WORKDIR /app
COPY --from=build /app/target/shoes-0.0.1-SNAPSHOT.jar /app

EXPOSE 1002

CMD ["java", "-jar", "shoes-0.0.1-SNAPSHOT.jar"]
```

Offer (Spring):

```
FROM maven as build
WORKDIR /app
COPY . .
RUN mvn install

#jdk
FROM openjdk:11.0.10-jre
WORKDIR /app
COPY --from=build /app/target/offers-0.0.1-SNAPSHOT.jar /app

CMD ["java", "-jar", "offers-0.0.1-SNAPSHOT.jar"]
```

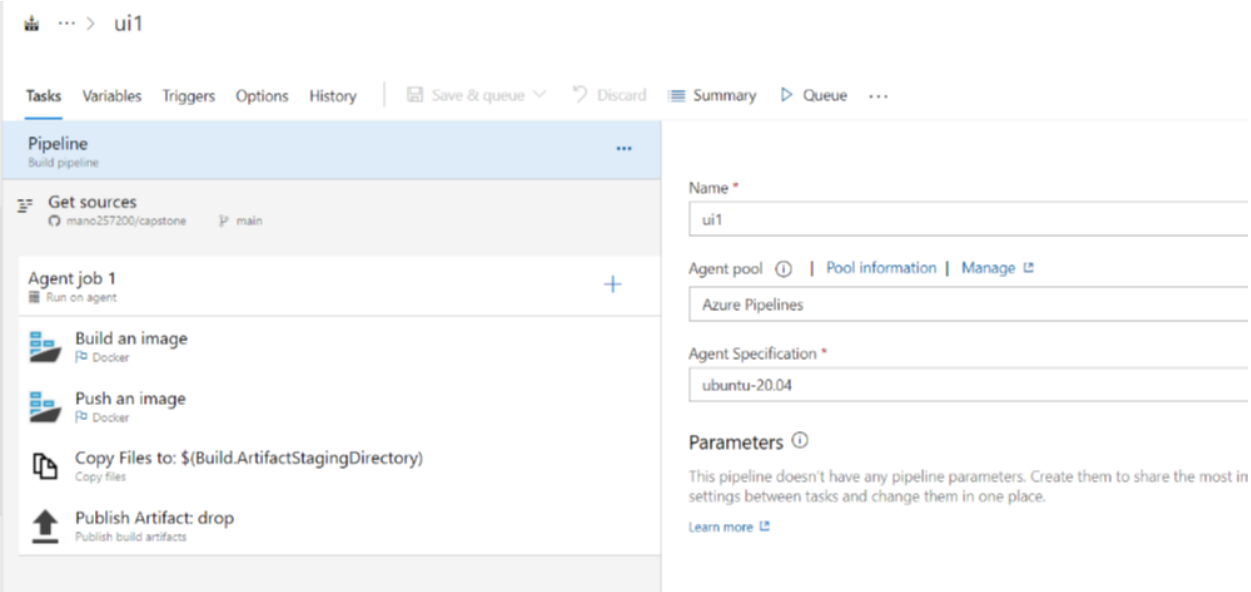
Cart (Node JS):

```
FROM node:14
WORKDIR /app
COPY . .
RUN npm install
CMD [ "node", "index.js" ]
```

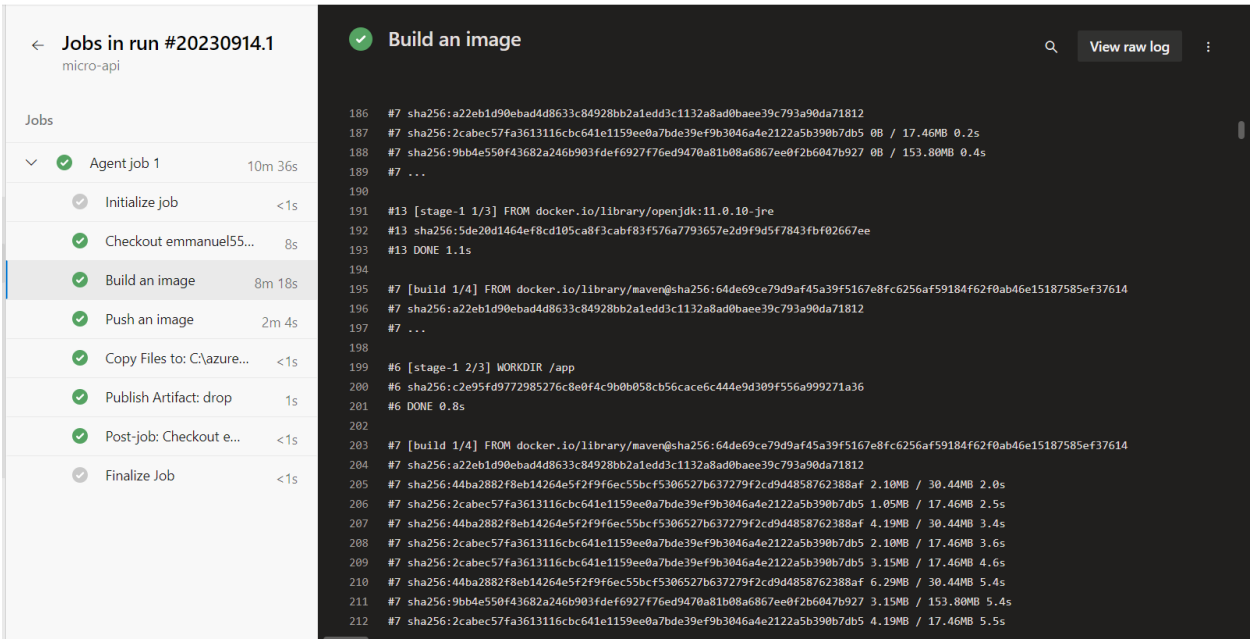
Wishlist (Python):

```
FROM python:3
COPY . .
RUN pip install flask flask_cors
CMD ["python","index.py"]
```

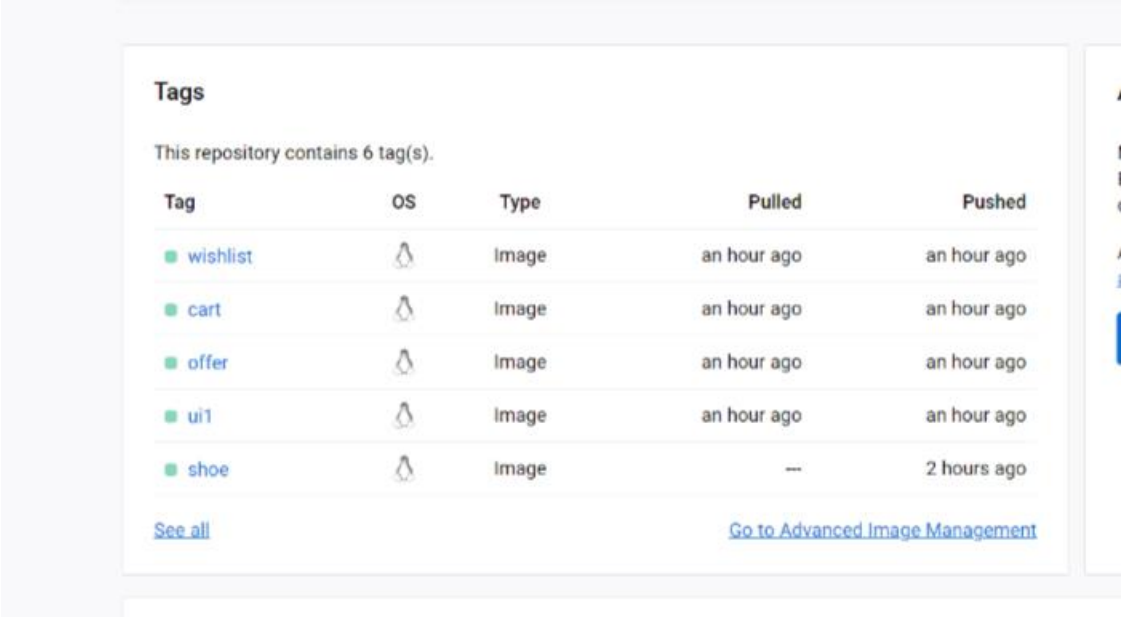
Build and push image using Azure devops pipeline and push the image to the DockerHub and copy to Artifact



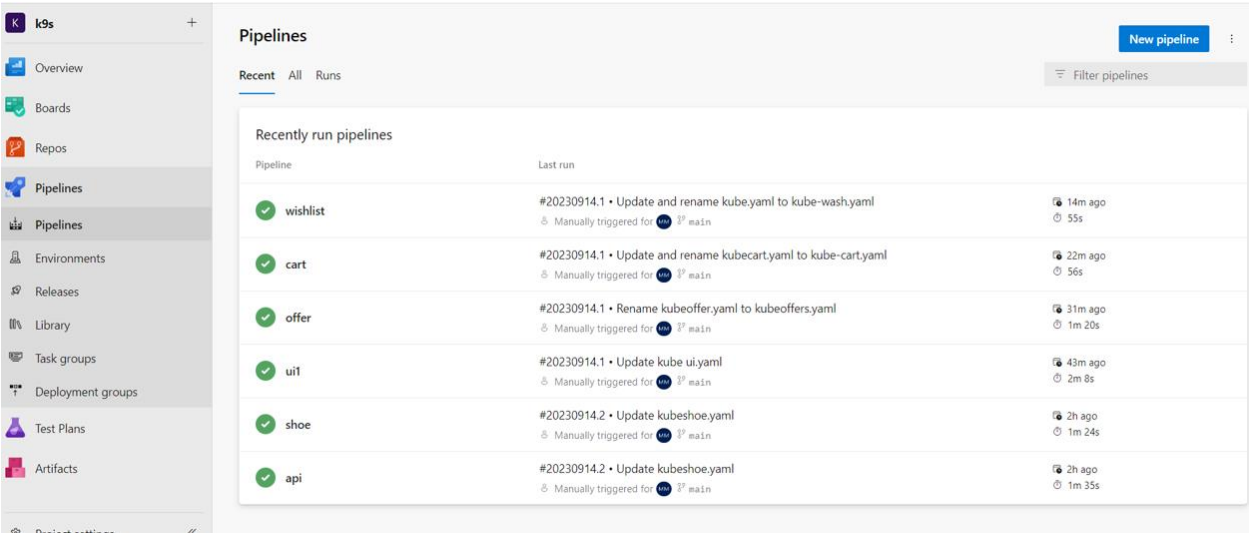
- Create a pipeline in Azure devops
- Add Agent Job to the pipeline like build an image , push an image , Copy files , Publish Artifact for continuous Integration



Pushed images into DockerHub:



CI Pipelines:



Step 3: Deploy Microservices Application in Kubernetes Using Release pipeline in Azure Devops

Create Kubernetes cluster in Azure:

- Create Azure kubernetes service's and Enable public Ip in the nodes

```
Requesting a Cloud Shell.Succeeded.
Connecting terminal...

Welcome to Azure Cloud Shell

Type "az" to use Azure CLI
Type "help" to learn about Cloud Shell

az account set --subscription fc115d67-76b0-4888-8c8c-016128f5251c
mahendravarman [ ~ ]$ az aks get-credentials --resource-group E_rg --name project
Storage fileshare subscription fc115d67-76b0-4888-8c8c-016128f5251c is not registered to Microsoft.CloudShell Namespace. Please follow these instructions "https://aka.ms
gisterCloudShell" to register. In future, unregistered subscriptions will have restricted access to CloudShell service.

mahendravarman [ ~ ]$ az account set --subscription fc115d67-76b0-4888-8c8c-016128f5251c
mahendravarman [ ~ ]$ az aks get-credentials --resource-group E_rg --name project
Merged "project" as current context in /home/mahendravarman/.kube/config
mahendravarman [ ~ ]$ kubectl get pods
No resources found in default namespace.
mahendravarman [ ~ ]$ kubectl get nodes -o wide
NAME                                STATUS   ROLES   AGE    VERSION   INTERNAL-IP   EXTERNAL-IP      OS-IMAGE             KERNEL-VERSION      CONTAINER-RUNTIME
aks-agentpool-39719263-vmss000000   Ready    agent   5m5s   v1.26.6   10.224.0.4    20.121.116.150   Ubuntu 22.04.3 LTS   5.15.0-1041-azure   containerd://1.7.1+azure-1
mahendravarman [ ~ ]$
```

Kubernetes manifest files for each microservices:

UI:

```yaml
---
apiVersion: apps/v1
kind: Deployment
metadata:
  name: "react-ui"
  labels:
    app: react-ui
spec:
  replicas: 1
  selector:
      matchLabels:
        app: "react-ui"
  template:
    metadata:
      labels:
        app: react-ui
    spec:
      containers:
        - name: "react-ui"
          image: "emman777/e-commerce:ui"
          imagePullPolicy: "IfNotPresent"
          ports:
            - containerPort: 8080
```

```yaml
apiVersion: "v1"
kind: "Service"
metadata:
  labels:
    app: "react-ui"
  name: "ui"
spec:
  ports:
  - name: "http"
    port: 8080
    targetPort: 8080
  selector:
    app: "react-ui"
  type: "NodePort"
```

```yaml
---
apiVersion: autoscaling/v2
kind: HorizontalPodAutoscaler
metadata:
  name: react-ui-autoscaler
spec:
  scaleTargetRef:
    apiVersion: apps/v1
    kind: Deployment
    name: react-ui
  minReplicas: 1
  maxReplicas: 5
  metrics:
  - type: Resource
    resource:
      name: cpu
      target:
        type: Utilization
        averageUtilization: 50
  - type: Resource
    resource:
      name: memory
      target:
        type: AverageValue
        averageValue: 100Mi
```

API:

```yaml
apiVersion: "v1"
kind: "Service"
metadata:
  labels:
    app: "zuul-api-gateway"
  name: "zuul-api-gateway"
spec:
  ports:
  - name: "http"
    port: 9999
    targetPort: 9999
    nodePort: 32470
  type: "NodePort"
  selector:
    app: "zuul-api-gateway"
```

```yaml
---
apiVersion: apps/v1
kind: Deployment
metadata:
  name: "zuul-api-gateway"
  labels:
    app: zulu
spec:
  replicas: 1
  selector:
      matchLabels:
        app: "zuul-api-gateway"
  template:
    metadata:
      labels:
        app: "zuul-api-gateway"
    spec:
      containers:
        - name: "zuul-api-gateway"
          image: "emman777/e-comerce:api"
          imagePullPolicy: "IfNotPresent"
          ports:
            - containerPort: 9999
```

Cart:

```yaml
---
apiVersion: v1
kind: Service
metadata:
  labels:
    app: cart-ui
  name: cart
spec:
  ports:
  - name: http
    port: 1004
    targetPort: 1004
  selector:
    app: cart-ui
```

```yaml
---
apiVersion: apps/v1
kind: Deployment
metadata:
  name: cart-ui
  labels:
    app: cart-ui
spec:
  replicas: 1
  selector:
    matchLabels:
      app: cart-ui
  template:
    metadata:
      labels:
        app: cart-ui
    spec:
      containers:
        - name: cart-ui
          image: mahendravarman12/e-commerce:cart
          imagePullPolicy: IfNotPresent
          ports:
            - containerPort: 1004
          readinessProbe:
            httpGet:
              path: /
              port: 1004
```

Offer:

```yaml
---
apiVersion: v1
kind: Service
metadata:
  labels:
    app: offer-ui
  name: offer
spec:
  ports:
  - name: http
    port: 1001
    targetPort: 1001
  selector:
    app: offer-ui
```

```yaml
---
apiVersion: apps/v1
kind: Deployment
metadata:
  name: offer-ui
  labels:
    app: offer-ui
spec:
  replicas: 1
  selector:
    matchLabels:
      app: offer-ui
  template:
    metadata:
      labels:
        app: offer-ui
    spec:
      containers:
        - name: offer-ui
          image: mahendravarman12/e-commerce:offer
          imagePullPolicy: IfNotPresent
          ports:
            - containerPort: 1001
```

Shoe:

```yaml
apiVersion: "v1"
kind: "Service"
metadata:
  labels:
    app: "shoe-ui"
  name: "shoe"
spec:
  ports:
  - name: "http"
    port: 1002
    targetPort: 1002
  selector:
    app: "shoe-ui"
```

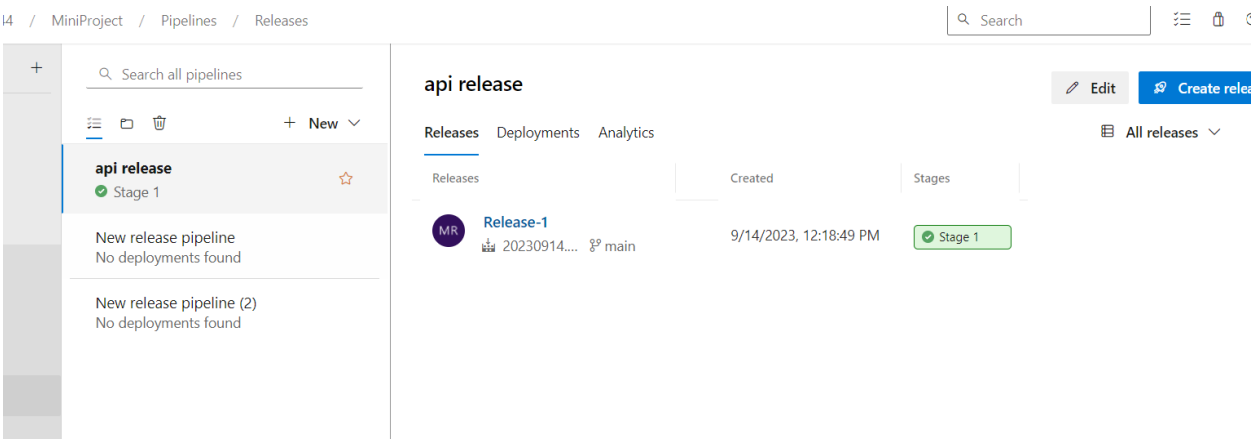
```yaml
---
apiVersion: apps/v1
kind: Deployment
metadata:
  name: "shoe-ui"
  labels:
    app: shoe-ui
spec:
  replicas: 1
  selector:
      matchLabels:
        app: "shoe-ui"
  template:
    metadata:
      labels:
        app: shoe-ui
    spec:
      containers:
        - name: "shoe-ui"
          image: "mahendravarman12/e-commerce:shoe"
          imagePullPolicy: "IfNotPresent"
          ports:
            - containerPort: 1002
```

Wish-list:

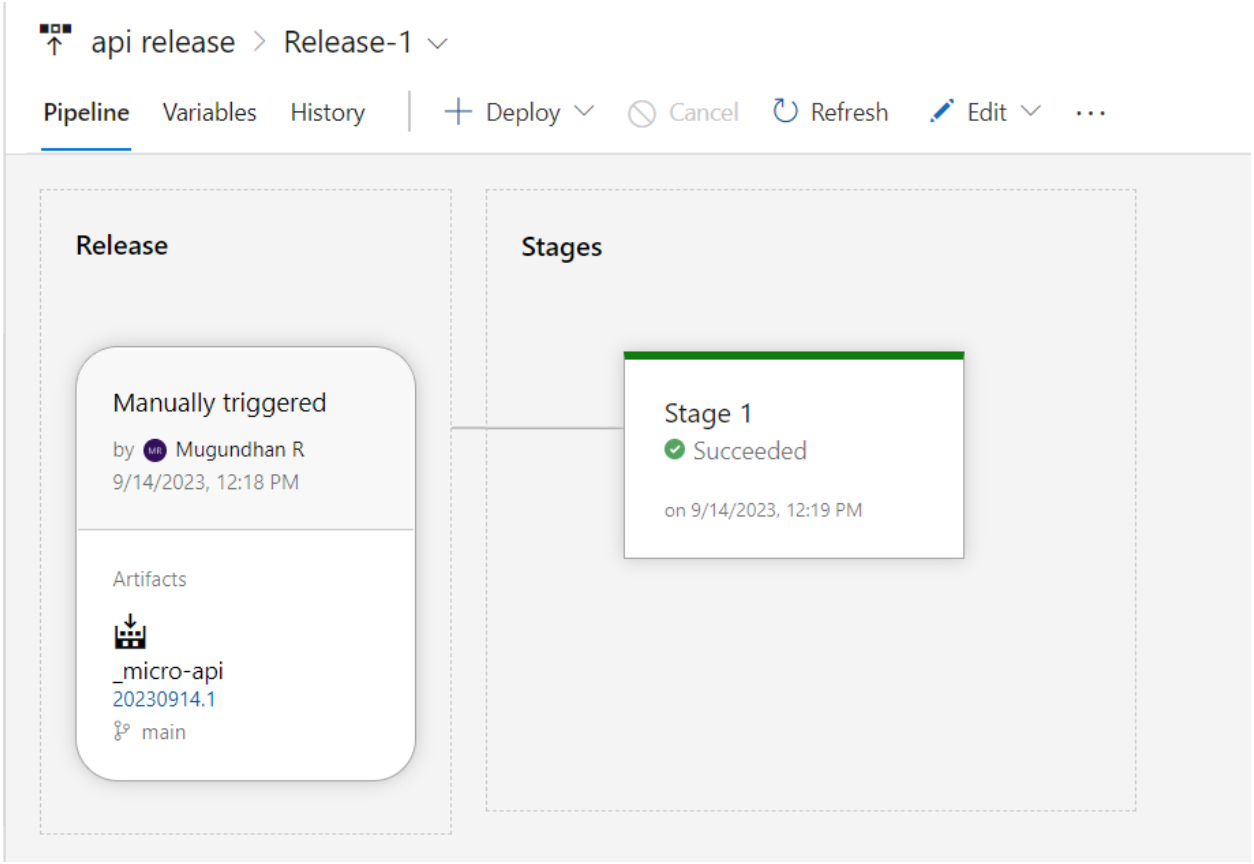
```yaml
---
apiVersion: v1
kind: Service
metadata:
  labels:
    app: wishlist-ui
  name: wishlist
spec:
  ports:
  - name: http
    port: 1003
    targetPort: 5000
  selector:
    app: wishlist-ui
```

```yaml
---
apiVersion: apps/v1
kind: Deployment
metadata:
  name: wishlist-ui
  labels:
    app: wishlist-ui
spec:
  replicas: 1
  selector:
    matchLabels:
      app: wishlist-ui
  template:
    metadata:
      labels:
        app: wishlist-ui
    spec:
      containers:
        - name: shoe-ui
          image: mahendravarman12/e-commerce:wish
          imagePullPolicy: IfNotPresent
          ports:
            - containerPort: 5000
```

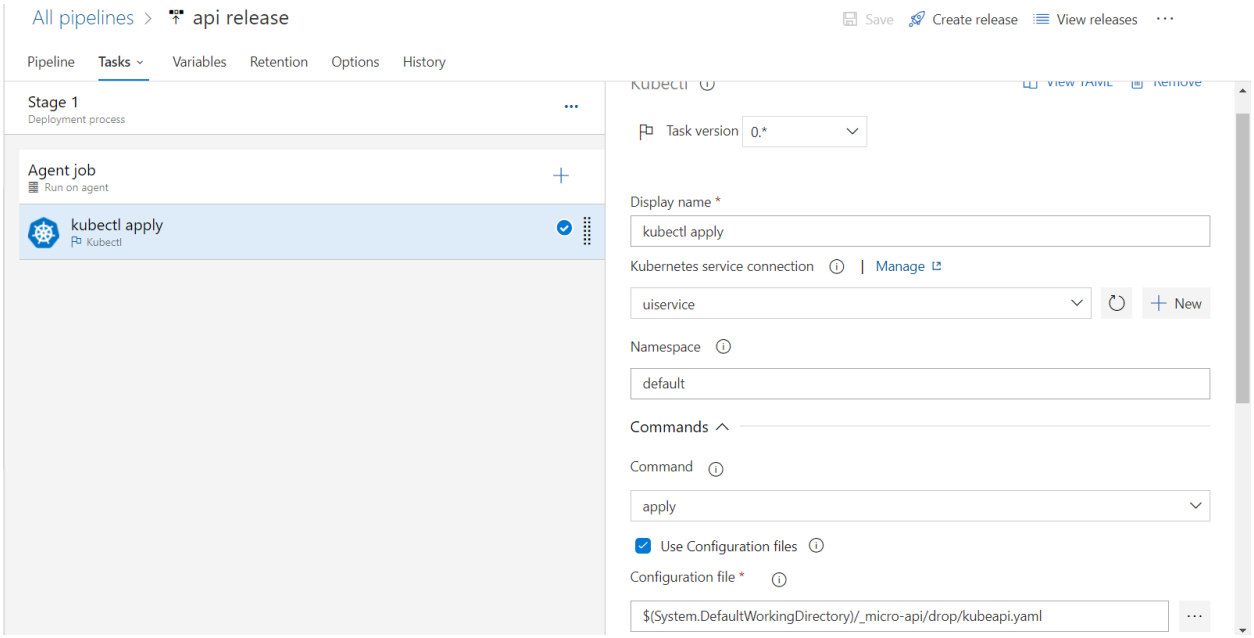
Continuous Deployment Using Release Pipeline:



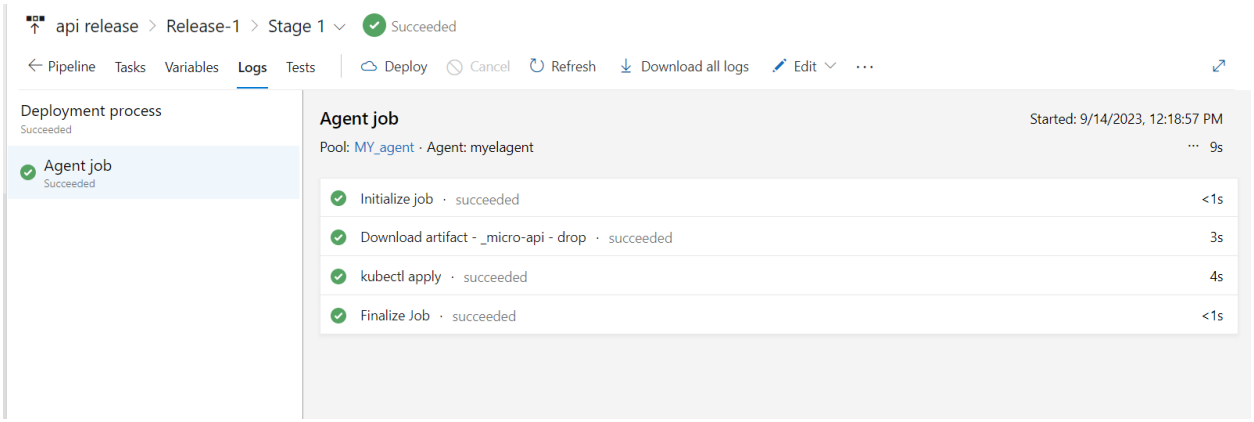
Setup the artifact and connect to Azure kubernetes



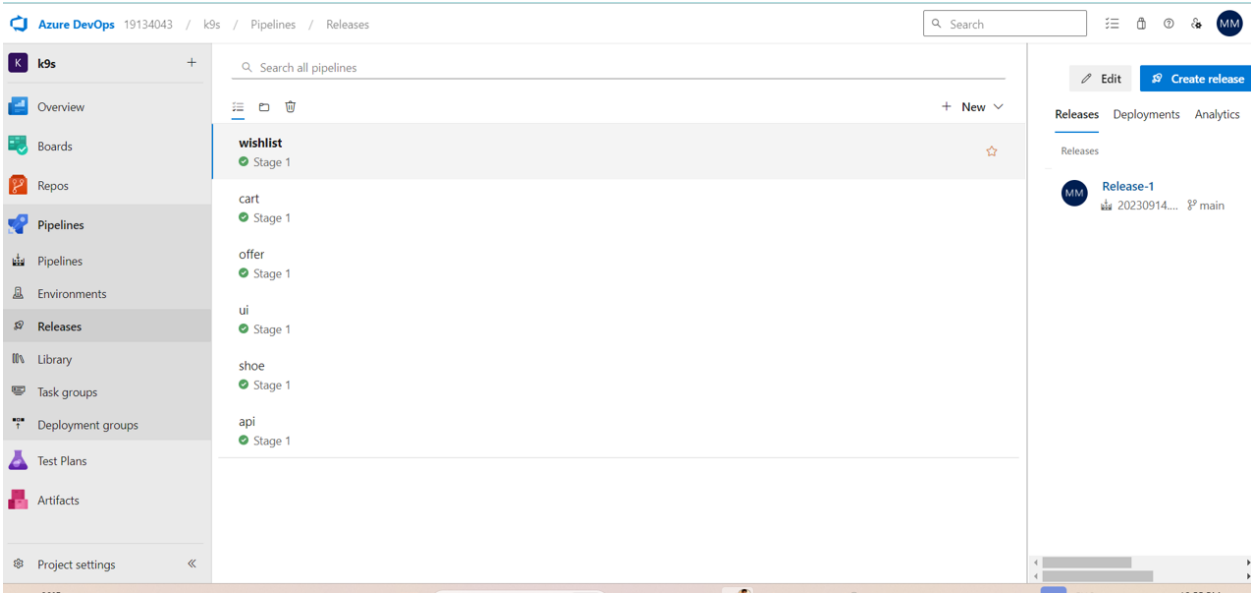
Set up kubernetes credentials and command



Create the Release



All Realease Pipeline for each microservices

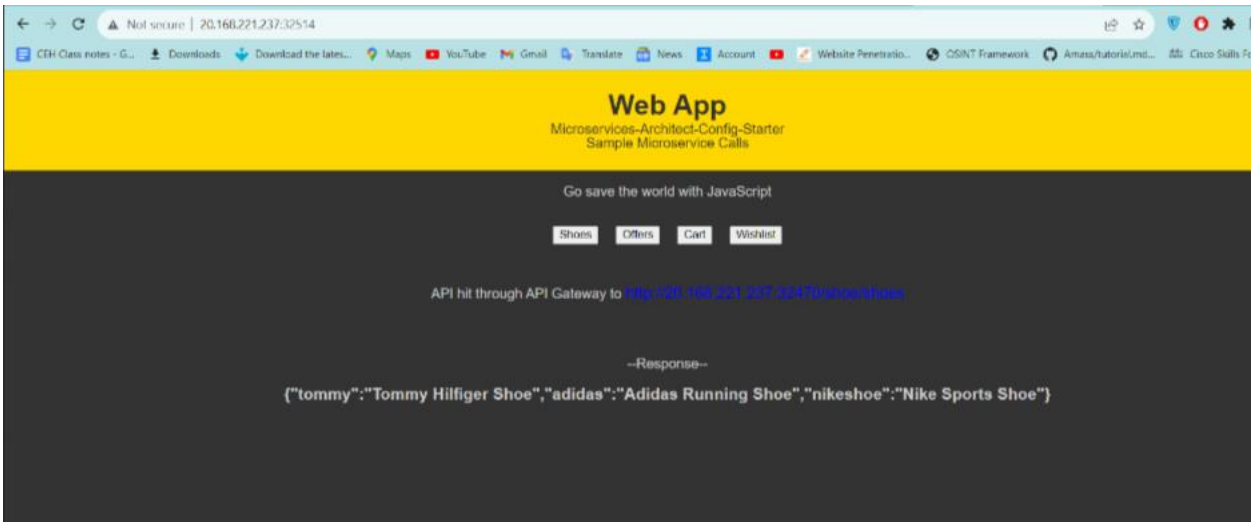


Deployed pods in AKS

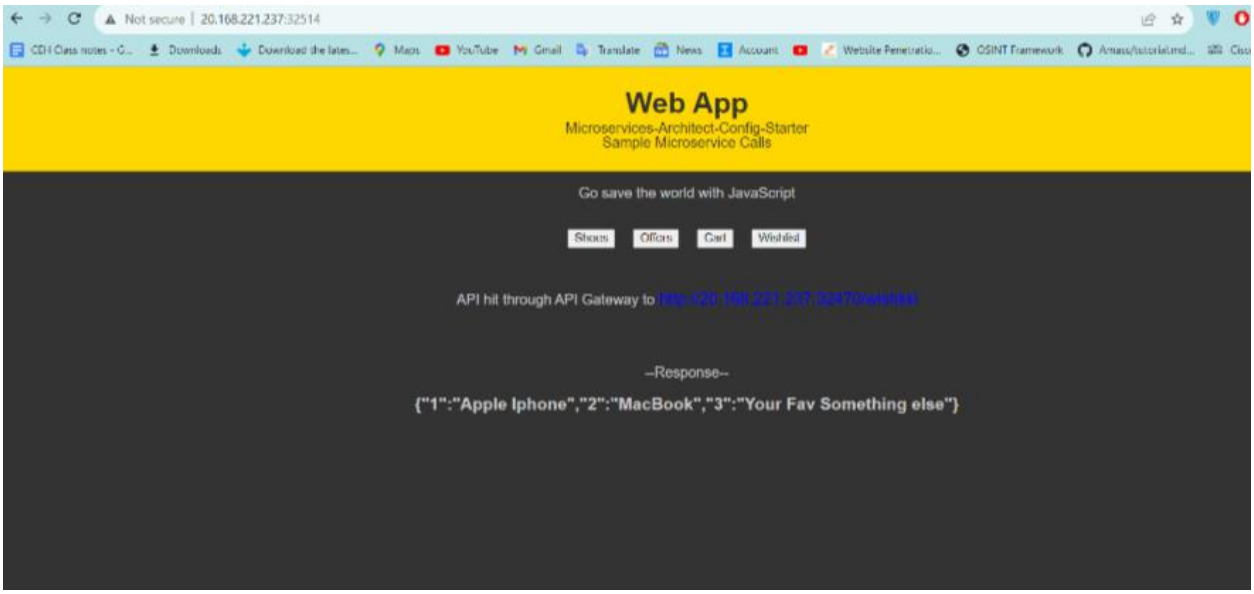
```
replicaset.apps/zuul-api-gateway-76c67ffccd   1         1         1       93m
mahendravarman [ ~ ]$ kubectl get all
NAME                                      READY   STATUS    RESTARTS   AGE
pod/cart-ui-75b6856b4f-xpp6s              1/1     Running   0          22m
pod/offer-ui-9776875fb-k5pw4              1/1     Running   0          28m
pod/shoe-ui-7659798b86-67z9k              1/1     Running   0          91m
pod/wishlist-ui-595f58c788-kplbc          1/1     Running   0          19m
pod/zuul-api-gateway-76c67ffccd-rts54     1/1     Running   0          94m

NAME                          TYPE        CLUSTER-IP     EXTERNAL-IP   PORT(S)          AGE
service/cart                  ClusterIP   10.0.250.87    <none>        1004/TCP         22m
service/kubernetes            ClusterIP   10.0.0.1       <none>        443/TCP          3h15m
service/offer                 ClusterIP   10.0.55.192    <none>        1001/TCP         28m
service/shoe                  ClusterIP   10.0.19.18     <none>        1002/TCP         91m
service/ui                    NodePort    10.0.114.88    <none>        8080:32514/TCP   56m
service/wishlist              ClusterIP   10.0.67.106    <none>        1003/TCP         19m
service/zuul-api-gateway      NodePort    10.0.225.35    <none>        9999:32470/TCP   94m

NAME                                  READY   UP-TO-DATE   AVAILABLE   AGE
deployment.apps/cart-ui               1/1     1            1           22m
deployment.apps/offer-ui              1/1     1            1           28m
deployment.apps/shoe-ui               1/1     1            1           91m
deployment.apps/wishlist-ui           1/1     1            1           19m
deployment.apps/zuul-api-gateway      1/1     1            1           94m

NAME                                            DESIRED   CURRENT   READY   AGE
replicaset.apps/cart-ui-75b6856b4f              1         1         1       22m
replicaset.apps/offer-ui-9776875fb              1         1         1       28m
replicaset.apps/shoe-ui-7659798b86              1         1         1       91m
replicaset.apps/wishlist-ui-595f58c788          1         1         1       19m
replicaset.apps/zuul-api-gateway-76c67ffccd     1         1         1       94m
```

Final Outputs:

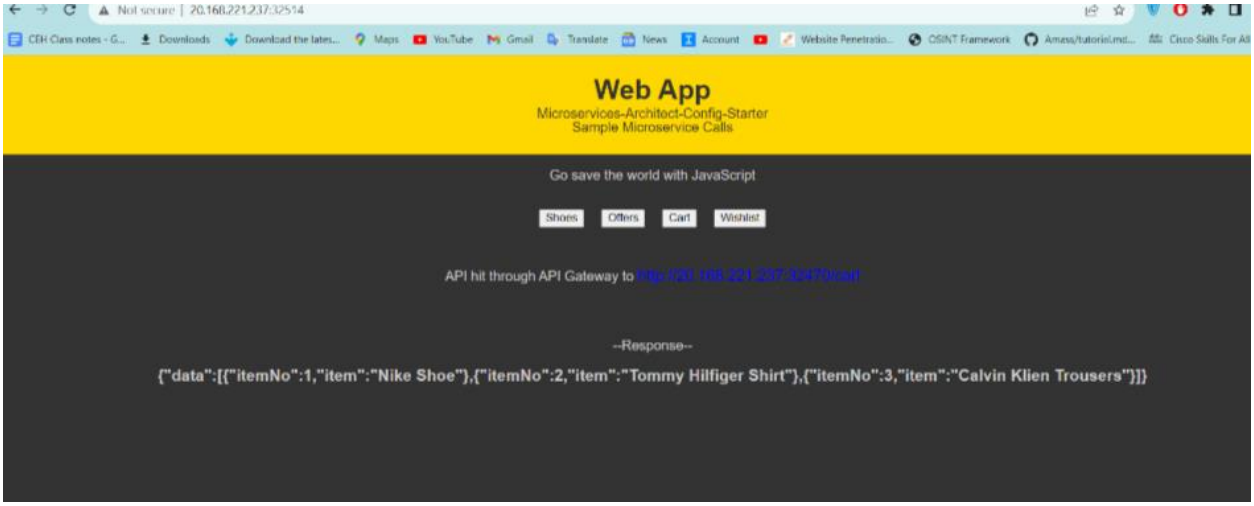
Shoe:



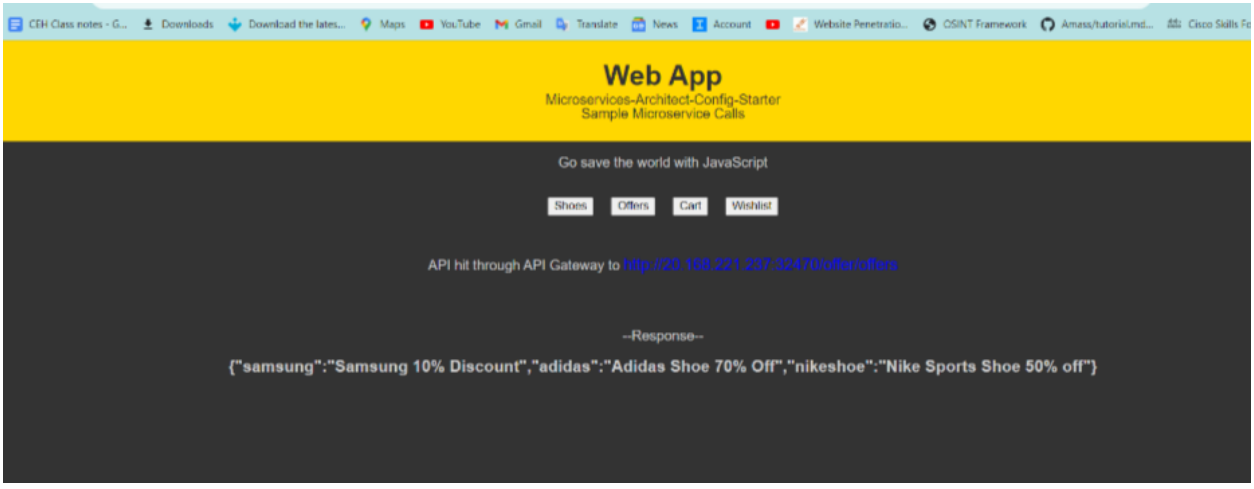
Wish-List:



Cart:


Not secure | 20.168.221.237:32514
Web App
Microservices-Architect-Config-Starter
Sample Microservice Calls
Go save the world with JavaScript
Shoes
Offers
Cart
Wishlist
API hit through API Gateway to
--Response--
{"data":[{"itemNo":1,"item":"Nike Shoe"},{"itemNo":2,"item":"Tommy Hilfiger Shirt"},{"itemNo":3,"item":"Calvin Klien Trousers"}]}

Offer:


Web App
Microservices-Architect-Config-Starter
Sample Microservice Calls
Go save the world with JavaScript
Shoes
Offers
Cart
Wishlist
API hit through API Gateway to
--Response--
{"samsung":"Samsung 10% Discount","adidas":"Adidas Shoe 70% Off","nikeshoe":"Nike Sports Shoe 50% off"}

Step 4: Monitor Microservices using EFK

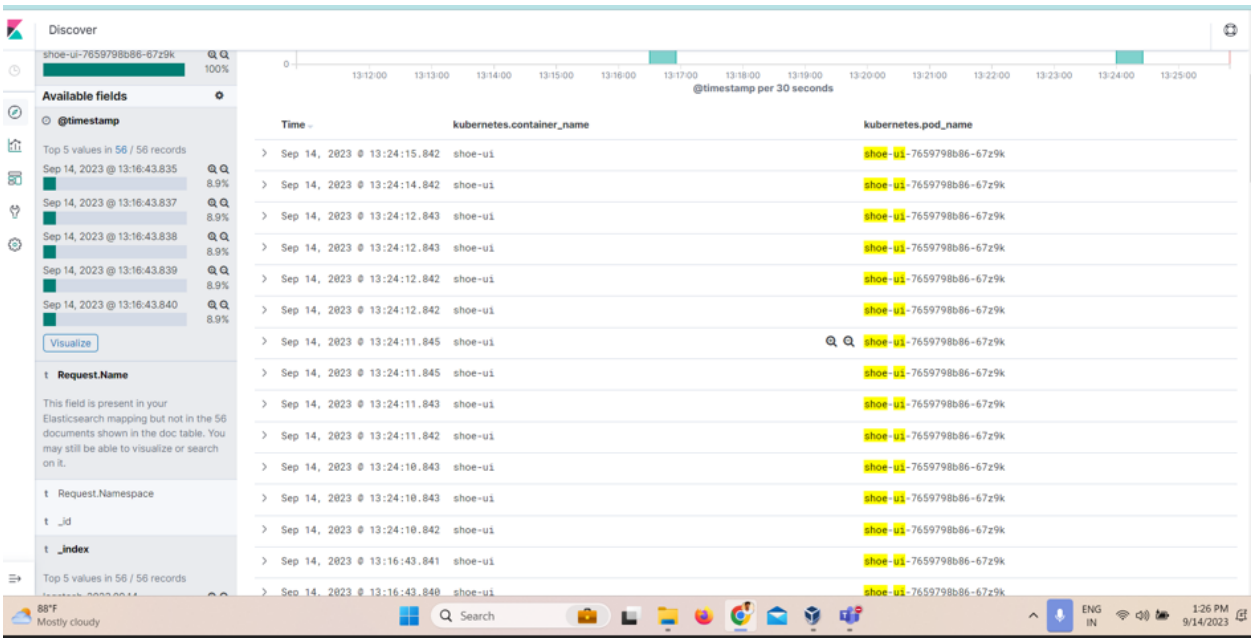
- EFK is an acronym that stands for Elasticsearch, Fluentd, and Kibana. It is a popular open-source stack used for log and event data analysis and visualization.
- Monitor each pods logs in efk

capstone-final / EFK-Log monitoring / Add file

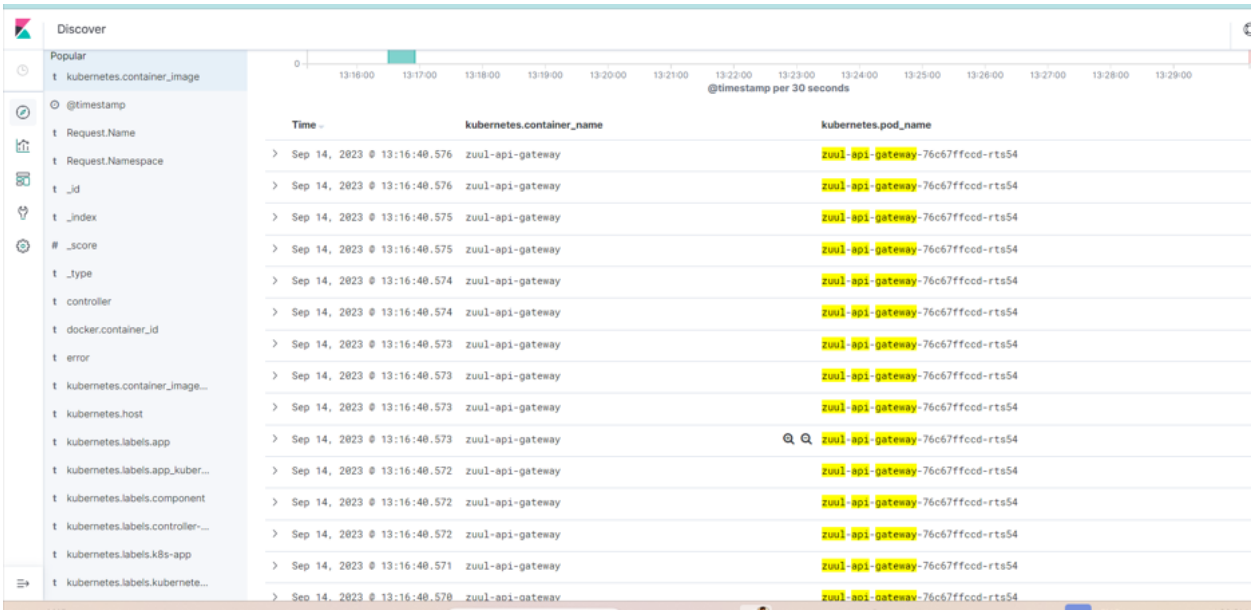
mano257200 Add files via upload 5fc3712 · 6 hours ago History

Name	Last commit message	Last commit date
..		
01_Namespace.yml	Add files via upload	6 hours ago
02_ElasticSearch_Service.yaml	Add files via upload	6 hours ago
03_ElasticSearch_StatefulSet.yaml	Add files via upload	6 hours ago
04_Fluentd_ConfigMap.yaml	Add files via upload	6 hours ago
05_Fluend_DaemonSet.yaml	Add files via upload	6 hours ago
06_Kibana_Deployment.yaml	Add files via upload	6 hours ago
07_Kibana_Service.yaml	Add files via upload	6 hours ago

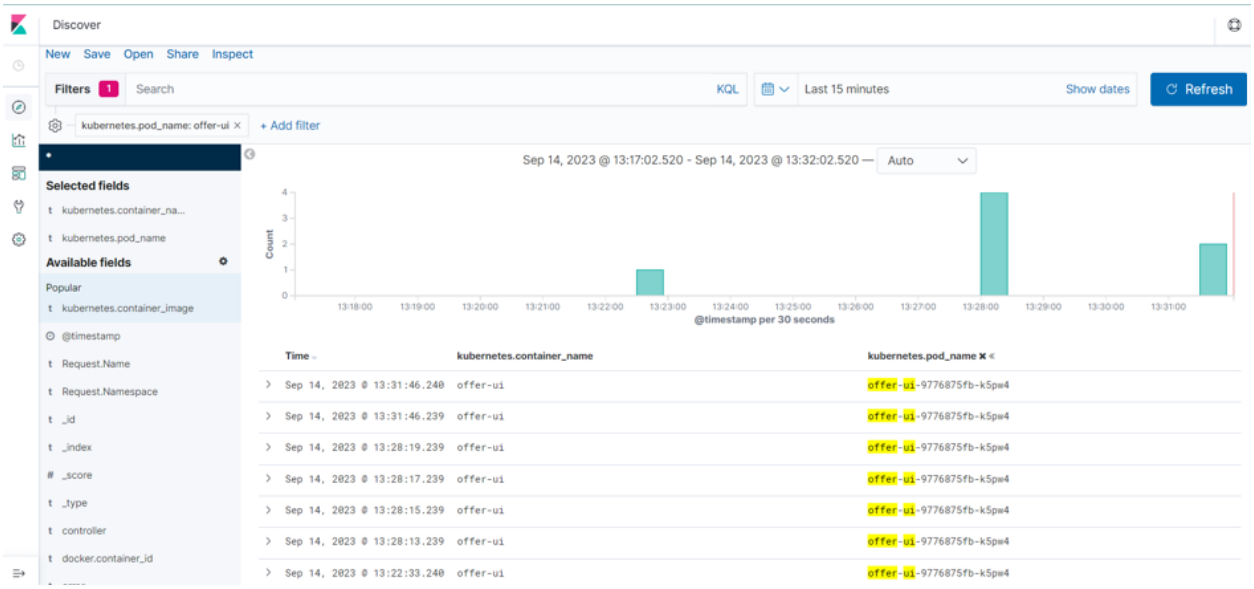
Pod shoe-ui:



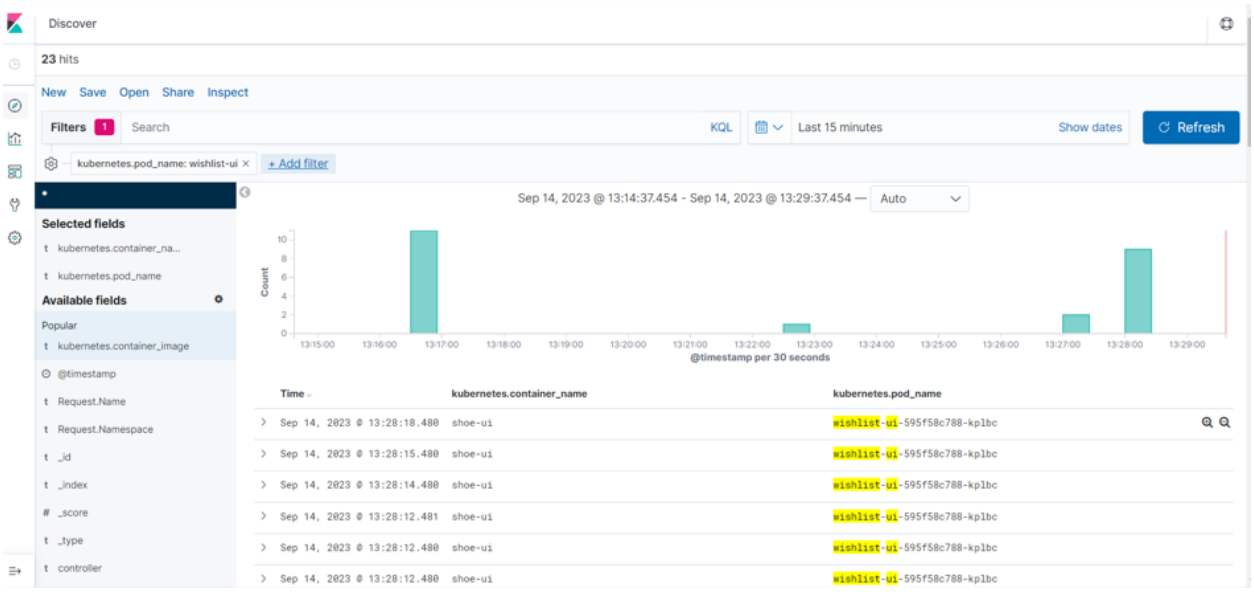
Pod zuul-api:



Pod Offer-ui:



Pod Wishlist-UI



Challenges Faced:

- Difficult to find the correct project git repo in github
- We get our code in github and In UI part the we Face code level problems like ip won't fetch automatically so we give statically ip using variables And In the UI part the code is written by very old version node 8 but latest version is node 18.14
- Setting up a reliable CI/CD pipeline for Kubernetes microservice
- Setting up and managing monitoring and logging solutions in Kubernetes