**kubarnates in microservice**

**food-catering-project/**

**│**

**├── service1/**

**│ ├── Dockerfile**

**│ ├── application.yml**

**│ └── k8s/**

**│ ├── deployment.yaml**

**│ └── service.yaml**

**│**

**├── service2/**

**│ ├── Dockerfile**

**│ ├── application.yml**

**│ └── k8s/**

**│ ├── deployment.yaml**

**│ └── service.yaml**

**│**

**├── service3/**

**│ ├── Dockerfile**

**│ ├── application.yml**

**│ └── k8s/**

**│ ├── deployment.yaml**

**│ └── service.yaml**

**│**

**├── databases/**

**│ ├── mysql/**

**│ │ ├── mysql-deployment.yaml**

**│ │ ├── mysql-service.yaml**

**│ │ └── mysql-pvc.yaml**

**│ │**

**│ └── oracle/**

**│ ├── oracle-deployment.yaml**

**│ ├── oracle-service.yaml**

**│ └── oracle-pvc.yaml**

**│**

**└── k8s-global/**

**└── ingress.yaml**

**🔧 Dockerfile (in each service)**

**dockerfile**

**FROM openjdk:17**

**COPY target/\*.jar app.jar**

**ENTRYPOINT ["java", "-jar", "/app.jar"]**

**⚙️ application.yml (in each service)**

**server:**

**port: 8080 # Change per service**

**spring:**

**datasource:**

**url: jdbc:mysql://mysql-service:3306/your\_db**

**username: root**

**password: password**

**📦 k8s/deployment.yaml (for each service)**

**apiVersion: apps/v1**

**kind: Deployment**

**metadata:**

**name: service1-deployment**

**spec:**

**replicas: 2**

**selector:**

**matchLabels:**

**app: service1**

**template:**

**metadata:**

**labels:**

**app: service1**

**spec:**

**containers:**

**- name: service1**

**image: your-dockerhub/service1:latest**

**ports:**

**- containerPort: 8080**

**🌐 k8s/service.yaml (for each service)**

**apiVersion: v1**

**kind: Service**

**metadata:**

**name: service1**

**spec:**

**selector:**

**app: service1**

**ports:**

**- protocol: TCP**

**port: 80**

**targetPort: 8080**

**type: ClusterIP**

**🐬 databases/mysql/mysql-deployment.yaml**

**apiVersion: apps/v1**

**kind: Deployment**

**metadata:**

**name: mysql**

**spec:**

**selector:**

**matchLabels:**

**app: mysql**

**replicas: 1**

**template:**

**metadata:**

**labels:**

**app: mysql**

**spec:**

**containers:**

**- name: mysql**

**image: mysql:8**

**env:**

**- name: MYSQL\_ROOT\_PASSWORD**

**value: password**

**- name: MYSQL\_DATABASE**

**value: your\_db**

**ports:**

**- containerPort: 3306**

**volumeMounts:**

**- name: mysql-pv**

**mountPath: /var/lib/mysql**

**volumes:**

**- name: mysql-pv**

**persistentVolumeClaim:**

**claimName: mysql-pvc**

**📡 databases/mysql/mysql-service.yaml**

**apiVersion: v1**

**kind: Service**

**metadata:**

**name: mysql-service**

**spec:**

**selector:**

**app: mysql**

**ports:**

**- protocol: TCP**

**port: 3306**

**targetPort: 3306**

**💾 databases/mysql/mysql-pvc.yaml**

**apiVersion: v1**

**kind: PersistentVolumeClaim**

**metadata:**

**name: mysql-pvc**

**spec:**

**accessModes:**

**- ReadWriteOnce**

**resources:**

**requests:**

**storage: 1Gi**

**🛢️ databases/oracle/oracle-deployment.yaml**

**apiVersion: apps/v1**

**kind: Deployment**

**metadata:**

**name: oracle**

**spec:**

**replicas: 1**

**selector:**

**matchLabels:**

**app: oracle**

**template:**

**metadata:**

**labels:**

**app: oracle**

**spec:**

**containers:**

**- name: oracle**

**image: gvenzl/oracle-xe:21-slim**

**env:**

**- name: ORACLE\_PASSWORD**

**value: password**

**ports:**

**- containerPort: 1521**

**volumeMounts:**

**- name: oracle-pv**

**mountPath: /opt/oracle/oradata**

**volumes:**

**- name: oracle-pv**

**persistentVolumeClaim:**

**claimName: oracle-pvc**

**📡 databases/oracle/oracle-service.yaml**

**apiVersion: v1**

**kind: Service**

**metadata:**

**name: oracle-service**

**spec:**

**selector:**

**app: oracle**

**ports:**

**- protocol: TCP**

**port: 1521**

**targetPort: 1521**

**💾 databases/oracle/oracle-pvc.yaml**

**apiVersion: v1**

**kind: PersistentVolumeClaim**

**metadata:**

**name: oracle-pvc**

**spec:**

**accessModes:**

**- ReadWriteOnce**

**resources:**

**requests:**

**storage: 1Gi**

**🌍 k8s-global/ingress.yaml (optional if using Ingress Controller)**

**apiVersion: networking.k8s.io/v1**

**kind: Ingress**

**metadata:**

**name: food-catering-ingress**

**annotations:**

**nginx.ingress.kubernetes.io/rewrite-target: /**

**spec:**

**rules:**

**- host: food.local**

**http:**

**paths:**

**- path: /service1**

**pathType: Prefix**

**backend:**

**service:**

**name: service1**

**port:**

**number: 80**

**- path: /service2**

**pathType: Prefix**

**backend:**

**service:**

**name: service2**

**port:**

**number: 80**

**command step by step:-**

**🚀 Deployment Flow**

**Build Docker Images for each service:**

**docker build -t your-dockerhub/service1:latest ./service1**

**docker build -t your-dockerhub/service2:latest ./service2**

**docker build -t your-dockerhub/service3:latest ./service3**

**Push to Docker Hub:**

**docker push your-dockerhub/service1:latest**

**# Repeat for others**

**Deploy Databases First:**

**kubectl apply -f databases/mysql/**

**kubectl apply -f databases/oracle/**

**Deploy Microservices:**

**kubectl apply -f service1/k8s/**

**kubectl apply -f service2/k8s/**

**kubectl apply -f service3/k8s/**

**(Optional) If using Ingress, apply:**

**kubectl apply -f k8s-global/ingress.yaml**

**NEW UPDATED CODE**

📁 Project Structure

food-catering-project/

├── service1/

│ ├── Dockerfile

│ ├── application.yml

│ └── k8s/

│ ├── deployment.yaml

│ ├── service.yaml

│ ├── configmap.yaml

│ └── secret.yaml

├── service2/

│ ├── Dockerfile

│ ├── application.yml

│ └── k8s/

│ ├── deployment.yaml

│ ├── service.yaml

│ ├── configmap.yaml

│ └── secret.yaml

├── service3/

│ ├── Dockerfile

│ ├── application.yml

│ └── k8s/

│ ├── deployment.yaml

│ ├── service.yaml

│ ├── configmap.yaml

│ └── secret.yaml

├── databases/

│ ├── mysql/

│ │ ├── mysql-deployment.yaml

│ │ ├── mysql-service.yaml

│ │ ├── mysql-pvc.yaml

│ │ └── mysql-secret.yaml

│ └── oracle/

│ ├── oracle-deployment.yaml

│ ├── oracle-service.yaml

│ ├── oracle-pvc.yaml

│ └── oracle-secret.yaml

└── k8s-global/

└── ingress.yaml

🐳 Dockerfile (for each service)

dockerfile

FROM openjdk:17-jdk-slim

COPY target/\*.jar app.jar

ENTRYPOINT ["java", "-jar", "/app.jar"]

⚙️ application.yml (for each service)

server:

port: 8080

spring:

datasource:

url: jdbc:mysql://mysql-service:3306/your\_db

username: ${DB\_USERNAME}

password: ${DB\_PASSWORD}

📦 k8s/deployment.yaml (for each service)

apiVersion: apps/v1

kind: Deployment

metadata:

name: service1-deployment

spec:

replicas: 2

selector:

matchLabels:

app: service1

template:

metadata:

labels:

app: service1

spec:

containers:

- name: service1

image: your-dockerhub/service1:latest

ports:

- containerPort: 8080

env:

- name: DB\_USERNAME

valueFrom:

secretKeyRef:

name: service1-secret

key: db-username

- name: DB\_PASSWORD

valueFrom:

secretKeyRef:

name: service1-secret

key: db-password

volumeMounts:

- name: config-volume

mountPath: /config

readOnly: true

volumes:

- name: config-volume

configMap:

name: service1-config

🌐 k8s/service.yaml (for each service)

apiVersion: v1

kind: Service

metadata:

name: service1

spec:

selector:

app: service1

ports:

- protocol: TCP

port: 80

targetPort: 8080

type: ClusterIP

🗂️ k8s/configmap.yaml (for each service)

apiVersion: v1

kind: ConfigMap

metadata:

name: service1-config

data:

application.yml: |

server:

port: 8080

spring:

datasource:

url: jdbc:mysql://mysql-service:3306/your\_db

🔐 k8s/secret.yaml (for each service)

apiVersion: v1

kind: Secret

metadata:

name: service1-secret

type: Opaque

data:

db-username: cm9vdA== # base64 for 'root'

db-password: cGFzc3dvcmQ= # base64 for 'password'

🐬 databases/mysql/mysql-deployment.yaml

apiVersion: apps/v1

kind: Deployment

metadata:

name: mysql

spec:

selector:

matchLabels:

app: mysql

replicas: 1

template:

metadata:

labels:

app: mysql

spec:

containers:

- name: mysql

image: mysql:8

env:

- name: MYSQL\_ROOT\_PASSWORD

valueFrom:

secretKeyRef:

name: mysql-secret

key: root-password

- name: MYSQL\_DATABASE

value: your\_db

ports:

- containerPort: 3306

volumeMounts:

- name: mysql-pv

mountPath: /var/lib/mysql

volumes:

- name: mysql-pv

persistentVolumeClaim:

claimName: mysql-pvc

📡 databases/mysql/mysql-service.yaml

apiVersion: v1

kind: Service

metadata:

name: mysql-service

spec:

selector:

app: mysql

ports:

- protocol: TCP

port: 3306

targetPort: 3306

💾 databases/mysql/mysql-pvc.yaml

apiVersion: v1

kind: PersistentVolumeClaim

metadata:

name: mysql-pvc

spec:

accessModes:

- ReadWriteOnce

resources:

requests:

storage: 1Gi

🔐 databases/mysql/mysql-secret.yaml

apiVersion: v1

kind: Secret

metadata:

name: mysql-secret

type: Opaque

data:

root-password: cGFzc3dvcmQ= # base64 for 'password'

🛢️ databases/oracle/oracle-deployment.yaml

apiVersion: apps/v1

kind: Deployment

metadata:

name: oracle

spec:

replicas: 1

selector:

matchLabels:

app: oracle

template:

metadata:

labels:

app: oracle

spec:

containers:

- name: oracle

image: gvenzl/oracle-xe:21-slim

env:

- name: ORACLE\_PASSWORD

valueFrom:

secretKeyRef:

name: oracle-secret

key: oracle-password

ports:

- containerPort: 1521

volumeMounts:

- name: oracle-pv

mountPath: /opt/oracle/oradata

volumes:

- name: oracle-pv

persistentVolumeClaim:

claimName: oracle-pvc

📡 databases/oracle/oracle-service.yaml

apiVersion: v1

kind: Service

metadata:

name: oracle-service

spec:

selector:

app: oracle

ports:

- protocol: TCP

port: 1521

targetPort: 1521

💾 databases/oracle/oracle-pvc.yaml

apiVersion: v1

kind: PersistentVolumeClaim

metadata:

name: oracle-pvc

spec:

accessModes:

- ReadWriteOnce

resources:

requests:

storage: 1Gi

🔐 databases/oracle/oracle-secret.yaml

apiVersion: v1

kind: Secret

metadata:

name: oracle-secret

type: Opaque

data:

oracle-password: cGFzc3dvcmQ= # base64 for 'password'

🌍 k8s-global/ingress.yaml (optional if using Ingress Controller)

apiVersion: networking.k8s.io/v1

kind: Ingress

metadata:

name: food-catering-ingress

annotations:

nginx.ingress.kubernetes.io/rewrite-target: /

spec:

rules:

- host: food.local

http:

paths:

- path: /service1

pathType: Prefix

backend:

service:

name: service1

port:

number: 80

- path: /service2

pathType: Prefix

backend:

service:

name: service2

port:

number: 80

- path: /service3

pathType: Prefix

backend:

service:

name: service3

port:

number: 80

🚀 Deployment Flow

Build Docker Images for each service:

docker build -t your-dockerhub/service1:latest ./service1

docker build -t your-dockerhub/service2:latest ./service2

docker build -t your-dockerhub/service3:latest ./service3

Push to Docker Hub:

docker push your-dockerhub/service1:latest

docker push your-dockerhub/service2:latest

docker push your-dockerhub/service3:latest

Deploy Databases First:

kubectl apply -f databases/mysql/

kubectl apply -f databases/oracle/

Deploy Microservices:

kubectl apply -f service1/k8s/

kubectl apply -f service2/k8s/

kubectl apply -f service3/k8s/

(Optional) If using Ingress, apply:

kubectl apply -f k8s-global/ingress.yaml

✅ Conclusion

Your Kubernetes setup for the food-catering-project is now enhanced with best practices:

ConfigMaps are used to manage non-sensitive configuration data, allowing for easy updates without rebuilding images.

Secrets securely store sensitive information like database credentials, ensuring they are not exposed in plain text.

Structured Deployment ensures each microservice is independently deployable and scalable.

Ingress Controller (optional) provides a unified entry point to your services, facilitating better traffic management.

This setup aligns with Kubernetes best practices, promoting scalability, maintainability, and security in your microservices architecture.