Simple Kubernetes Configuration for a Three-Service Application

Grok

June 1, 2025

1 Introduction

This document provides Kubernetes manifests for deploying a three-service application (React front-end, Node.js/Express backend, MongoDB database) using `kubectl`. The setup is minimal, focusing on basic `Deployment`, `Service`, and `Secret` resources, suitable for development or small-scale deployments. Comments explain adaptability to requirement changes.

2 Prerequisites

- A Kubernetes cluster (e.g., Minikube, GKE) with `kubectl` configured.
- Docker images for front-end and backend pushed to a registry (e.g., Docker Hub).
- MongoDB credentials stored securely (e.g., in a `.env` file).

3 Kubernetes Secret

Defining sensitive data for MongoDB credentials.

Listing 1: secret.yaml

```
apiVersion: v1
kind: Secret
metadata:
  name: mongodb-credentials # Change if naming convention changes
type: Opaque
data:
  mongo-user: bW9uZ29fdXNlcg== # Base64-encoded 'mongo_user'; update if
      credentials change
  mongo-password: bW9uZ29fcGFzc3dvcmQ= # Base64-encoded 'mongo_password';
      update if credentials change
```

4 Frontend Deployment and Service

Configuring the React front-end.

Listing 2: frontend-deployment.yaml

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: frontend # Change if naming convention changes
spec:
  replicas: 2 # Adjust for scaling needs
  selector:
    matchLabels:
      app: frontend
  template:
    metadata:
      labels:
        app: frontend # Change if label changes
    spec:
      containers:
      - name: frontend
        image: your_dockerhub_username/frontend:latest # Update with your
            Docker Hub image
        ports:
        - containerPort: 3000 # Change if frontend framework uses different
            port (e.g., Angular: 4200)
---
apiVersion: v1
kind: Service
metadata:
  name: frontend-service # Change if naming convention changes
spec:
  selector:
    app: frontend
  ports:
  - port: 80
    targetPort: 3000 # Align with containerPort
  type: LoadBalancer # Change to ClusterIP or NodePort for different access
      needs
```

5 Backend Deployment and Service

Configuring the Node.js/Express backend.

Listing 3: backend-deployment.yaml

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: backend # Change if naming convention changes
spec:
```

```
  replicas: 2 # Adjust for scaling needs
  selector:
    matchLabels:
      app: backend
  template:
    metadata:
      labels:
        app: backend # Change if label changes
    spec:
      containers:
      - name: backend
        image: your_dockerhub_username/backend:latest # Update with your Docker Hub image
        ports:
        - containerPort: 5000 # Change if backend framework uses different port (e.g., FastAPI: 8000)
        env:
        - name: MONGO_URI
          valueFrom:
            secretKeyRef:
              name: mongodb-credentials # Align with Secret name
              key: mongo-user
        - name: MONGO_PASSWORD
          valueFrom:
            secretKeyRef:
              name: mongodb-credentials
              key: mongo-password
---
apiVersion: v1
kind: Service
metadata:
  name: backend-service # Change if naming convention changes
spec:
  selector:
    app: backend
  ports:
  - port: 80
    targetPort: 5000 # Align with containerPort
  type: ClusterIP # Change to LoadBalancer or NodePort if external access needed
```

6 Database Deployment and Service

Configuring the MongoDB database.

Listing 4: database-deployment.yaml

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: mongodb # Change if naming convention changes
```

```
spec:
  replicas: 1 # Adjust for scaling (MongoDB typically single replica)
  selector:
    matchLabels:
      app: mongodb
  template:
    metadata:
      labels:
        app: mongodb # Change if label changes
    spec:
      containers:
      - name: mongodb
        image: mongo:7.0 # Change to different DB (e.g., postgres:16) if needed
        ports:
        - containerPort: 27017 # Change if DB uses different port
        env:
        - name: MONGO_INITDB_ROOT_USERNAME
          valueFrom:
            secretKeyRef:
              name: mongodb-credentials # Align with Secret name
              key: mongo-user
        - name: MONGO_INITDB_ROOT_PASSWORD
          valueFrom:
            secretKeyRef:
              name: mongodb-credentials
              key: mongo-password
        volumeMounts:
        - name: mongo-data
          mountPath: /data/db # Change if DB uses different data path
      volumes:
      - name: mongo-data
        persistentVolumeClaim:
          claimName: mongo-pvc # Align with PVC name
---
apiVersion: v1
kind: Service
metadata:
  name: mongodb-service # Change if naming convention changes
spec:
  selector:
    app: mongodb
  ports:
  - port: 27017
    targetPort: 27017 # Align with containerPort
  type: ClusterIP # Change if external access needed
---
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: mongo-pvc # Change if naming convention changes
spec:
```

```
  accessModes:
    - ReadWriteOnce # Adjust based on storage needs
  resources:
    requests:
      storage: 10Gi # Adjust size as needed
```

7 How to Run

1. Ensure Docker images are built and pushed to a registry (e.g., Docker Hub).
2. Save manifests as `secret.yaml`, `frontend-deployment.yaml`, `backend-deployment.yaml`, and `database-deployment.yaml`.
3. Apply manifests: `kubectl apply -f secret.yaml -f frontend-deployment.yaml -f backend-deployment.yaml -f database-deployment.yaml`.
4. Verify pods: `kubectl get pods`.
5. Access the front-end via the `frontend-service` LoadBalancer IP: `kubectl get svc frontend-service`.
6. Delete resources: `kubectl delete -f .`.

8 Adapting to Requirement Changes

- **Change Frontend Framework**: Update `frontend-deployment.yaml` image and port (e.g., Angular uses port 4200).
- **Change Backend Framework**: Update `backend-deployment.yaml` image and port (e.g., FastAPI uses Python image and port 8000).
- **Change Database**: Update `database-deployment.yaml` image and volume paths (e.g., `postgres:16` for PostgreSQL).
- **Scaling**: Adjust `replicas` in `Deployment` specs.
- **External Access**: Change `Service` type to `NodePort` or `LoadBalancer` as needed.