🔧Strimzi Kafka Operator — Local Build & Minikube Deployment Blueprint

This blueprint captures **everything that worked** to successfully build, configure, and run a custom version of the Strimzi Kafka Operator with support for Kafka `3.9.1` using **Podman + Minikube + GitHub Container Registry**.

📁Repo & Directory Structure

- **Repo cloned**: Custom fork or official

  ```
  git clone https://github.com/strimzi/strimzi-kafka-operator.git
  ```

- Your local directory: `~/github/strimzi-kafka-operator`

🧱Java + Maven Setup

- Java version needed: **Java 21 JDK**, but Maven is configured to build with Java 17 source compatibility.

```
sudo apt install openjdk-21-jdk
javac -version    # should show javac 17.* or 21.*
mvn -version
```

🔧Maven Build

Skip broken `config-model-generator` and just build the operator:

```
mvn clean install -DskipTests -pl cluster-operator -am -Dcheckstyle.skip
```

📦JAR Preparation for Docker

```
cp cluster-operator/target/cluster-operator-*.jar docker-images/operator/tmp/
cluster-operator.jar
```

📠Podman Build & Push

Dockerfile (simplified)

Make sure `docker-images/operator/Dockerfile` exists and looks like:

```
FROM strimzi/base:latest

USER root
RUN mkdir -p /opt/strimzi && useradd -r -u 1001 -g 0 strimzi

COPY tmp/cluster-operator.jar /opt/strimzi/cluster-operator.jar
USER 1001
WORKDIR /opt/strimzi
ENTRYPOINT ["java", "-jar", "/opt/strimzi/cluster-operator.jar"]
```

Build & Push

```
cd docker-images/operator
podman build -t ghcr.io/arbaz6400/strimzi-kafka-operator:latest .
podman login ghcr.io
podman push ghcr.io/arbaz6400/strimzi-kafka-operator:latest
```

📌Minikube Setup

```
minikube start --driver=podman
kubectl create namespace strimzi
```

📥Install CRDs & Operator

```
kubectl apply -f install/cluster-operator -n strimzi
```

🛠️Patch the Deployment to Use Your Image & Fix Versions

```
kubectl -n strimzi set image deployment/strimzi-cluster-operator \
  strimzi-cluster-operator=ghcr.io/arbaz6400/strimzi-kafka-operator:latest

kubectl -n strimzi set env deployment/strimzi-cluster-operator \
  STRIMZI_KAFKA_IMAGES="3.9.0=quay.io/strimzi/kafka:0.46.0-
kafka-3.9.0,3.9.1=quay.io/strimzi/kafka:0.46.0-kafka-3.9.1,4.0.0=quay.io/
strimzi/kafka:0.46.0-kafka-4.0.0" \
  STRIMZI_KAFKA_CONNECT_IMAGES="3.9.0=quay.io/strimzi/kafka:0.46.0-
kafka-3.9.0,3.9.1=quay.io/strimzi/kafka:0.46.0-kafka-3.9.1,4.0.0=quay.io/
strimzi/kafka:0.46.0-kafka-4.0.0" \
  STRIMZI_KAFKA_MIRROR_MAKER_2_IMAGES="3.9.0=quay.io/strimzi/kafka:0.46.0-
kafka-3.9.0,3.9.1=quay.io/strimzi/kafka:0.46.0-kafka-3.9.1,4.0.0=quay.io/
strimzi/kafka:0.46.0-kafka-4.0.0"

kubectl delete pod -n strimzi -l name=strimzi-cluster-operator
```

🔍Verify Operator is Running

```
kubectl get pods -n strimzi
kubectl logs -n strimzi -l name=strimzi-cluster-operator --tail=100 -f
```

You should **not** see `InvalidConfigurationException`.

🗜️Deploy Kafka Cluster (3.9.1)

Save as `kafka-cluster.yaml`:

```
apiVersion: kafka.strimzi.io/v1beta2
kind: Kafka
metadata:
  name: my-cluster
  namespace: strimzi
spec:
  kafka:
    version: 3.9.1
    replicas: 1
    listeners:
      - name: plain
        port: 9092
```

```
          type: internal
          tls: false
      storage:
        type: ephemeral
    zookeeper:
      replicas: 1
      storage:
        type: ephemeral
    entityOperator:
      topicOperator: {}
      userOperator: {}
```

Apply:

```
kubectl apply -f kafka-cluster.yaml -n strimzi
kubectl get pods -n strimzi -w
```

🔗Success Criteria

- Operator pod: `1/1 Running`
- Kafka, Zookeeper, EntityOperator: All pods are `Running`
- Logs show:
  `Cluster Operator is now ready (Kafka versions: [3.9.0, 3.9.1, 4.0.0])`

You're done. 🎉