Kafka

To install Kafk we need a docker-compose file (compose file created by chat gpt).
In compose file installed zookeeper, kafka, kafka-ui

Follow this steps to install configure and test airflow

1. Create a folder called “kafka”
2. Past docker compose file script
3. Docker compose up -d
4. Kafka ui should run on port 8082
5. Check all the volume and port details on the docker compose file

Testing kafka

1. Open airflow ui using port 8082
2. Try to communicate send and receive to verify
3. Check in kafka-ui the topics and messages are there

✅ Installation Steps

1. **Create project folder & file**

Shell

```
mkdir kafka-cluster
cd kafka-cluster
nano docker-compose.yml
```

2. **Paste the above YAML** and save.

3. **Start the cluster**

Shell

```
docker compose up -d
```

4. **Verify services**

Shell

```
docker ps
```

You should see `zookeeper` and `kafka` running.

5. **Check logs**

Shell

```
docker logs kafka | grep -i started
```

Look for `Kafka Server started`.

✅ Test Kafka Setup

Create a topic:

Shell

```
docker exec -it kafka bash
kafka-topics --create --topic test-topic --bootstrap-server localhost:9092 --partitions 1 --replication-factor 1
```

List topics:

Shell

```
kafka-topics --list --bootstrap-server localhost:9092
```

Produce messages:

Shell

```
kafka-console-producer --topic test-topic
--bootstrap-server localhost:9092
```

(Type a few lines, press **Enter** after each)

Consume messages:

Shell

```
kafka-console-consumer --topic test-topic
--bootstrap-server localhost:9092 --from-beginning
```

✅ Check Data Persistence

- Stop the cluster:

Shell

```
docker compose down
```

- Start again:

Shell

```
docker compose up -d
```

Your topics and data remain because of mounted volumes.

---

docker-compose.yml

```
services:
  zookeeper:
    image: confluentinc/cp-zookeeper:7.6.0
    container_name: zookeeper
    restart: unless-stopped
    ports:
      - "2181:2181"
    environment:
      ZOOKEEPER_CLIENT_PORT: 2181
      ZOOKEEPER_TICK_TIME: 2000
    volumes:
      - zookeeper-data:/var/lib/zookeeper/data
      - zookeeper-log:/var/lib/zookeeper/log
    networks:
      - kafka-net
    healthcheck:
      test: ["CMD-SHELL", "bash -c '</dev/tcp/localhost/2181'"]
      interval: 10s
      timeout: 5s
      retries: 5

  kafka:
    image: confluentinc/cp-kafka:7.6.0
    container_name: kafka
    restart: unless-stopped
    depends_on:
      zookeeper:
        condition: service_healthy
    ports:
      - "9092:9092"       # Internal listener
      - "29092:29092"     # External listener
    environment:
      KAFKA_BROKER_ID: 1
      KAFKA_ZOOKEEPER_CONNECT: zookeeper:2181
      KAFKA_LISTENERS:
PLAINTEXT://0.0.0.0:9092,PLAINTEXT_HOST://0.0.0.0:29092
      KAFKA_ADVERTISED_LISTENERS:
PLAINTEXT://kafka:9092,PLAINTEXT_HOST://localhost:29092
      KAFKA_LISTENER_SECURITY_PROTOCOL_MAP:
PLAINTEXT:PLAINTEXT,PLAINTEXT_HOST:PLAINTEXT
      KAFKA_INTER_BROKER_LISTENER_NAME: PLAINTEXT
      KAFKA_OFFSETS_TOPIC_REPLICATION_FACTOR: 1
      KAFKA_GROUP_INITIAL_REBALANCE_DELAY_MS: 0
    volumes:
      - kafka-data:/var/lib/kafka/data
    networks:
      - kafka-net
    healthcheck:
      test: ["CMD", "kafka-broker-api-versions", "--bootstrap-server=localhost:9092"]
      interval: 15s
      timeout: 10s
      retries: 5

  kafka-ui:
    image: provectuslabs/kafka-ui:latest
    container_name: kafka-ui
    restart: unless-stopped
    depends_on:
      kafka:
        condition: service_healthy
    ports:
      - "8082:8080" # Access UI on http://localhost:8082
    environment:
      KAFKA_CLUSTERS_0_NAME: local
      KAFKA_CLUSTERS_0_BOOTSTRAPSERVERS: kafka:9092
      KAFKA_CLUSTERS_0_ZOOKEEPER: zookeeper:2181
    networks:
      - kafka-net
    healthcheck:
      test: ["CMD", "curl", "-f", "http://localhost:8082"]
      interval: 10s
      timeout: 5s
      retries: 5

volumes:
  zookeeper-data:
  zookeeper-log:
  kafka-data:

networks:
  kafka-net:
    driver: bridge
```