PART 1: MinIO Webhook API (FastAPI)

1️⃣ What MinIO Sends (important context)

MinIO sends **S3-compatible event JSON**, like:

```
{
  "Records": [
    {
      "eventName": "s3:ObjectCreated:Put",
      "s3": {
        "bucket": {
          "name": "landing-bucket"
        },
        "object": {
          "key": "raw/data/file1.csv",
          "size": 1024
        }
      }
    }
  ]
}
```

We'll design the API **specifically** to:

- accept this payload
- validate it
- extract bucket + object
- trigger downstream logic (REST call, Spark, Airflow, etc.)

2️⃣ FastAPI Webhook API (Production-ready skeleton)

app/main.py

```
from fastapi import FastAPI, Request, HTTPException, Header
import logging
import requests

app = FastAPI(title="MinIO Webhook Service")

logging.basicConfig(level=logging.INFO)

@app.get("/health")
def health():
    return {"status": "ok"}

@app.post("/webhook/minio")
async def minio_webhook(
    request: Request,
    x_minio_token: str = Header(None)
):
    # Optional auth
    if x_minio_token != "minio-secret-token":
        raise HTTPException(status_code=401, detail="Unauthorized")

    payload = await request.json()
    records = payload.get("Records", [])

    for record in records:
        event = record.get("eventName")
        bucket = record["s3"]["bucket"]["name"]
        obj = record["s3"]["object"]["key"]
        size = record["s3"]["object"].get("size", 0)

        logging.info(
            f"Event={event}, Bucket={bucket}, Object={obj},
Size={size}"
        )

        # Example: call downstream REST API
        requests.post(
```

```
            "http://downstream-service/process",
            json={
                "bucket": bucket,
                "object": obj,
                "size": size
            },
            timeout=5
        )

    return {"status": "processed"}
```

---

requirements.txt

```
fastapi
uvicorn[standard]
requests
```

PART 2: FastAPI Setup on On-Prem Kubernetes

3️⃣ Dockerize the FastAPI App

Dockerfile

```
FROM python:3.11-slim

WORKDIR /app

COPY requirements.txt .
RUN pip install --no-cache-dir -r requirements.txt

COPY app/ app/
```

```
EXPOSE 8000

CMD ["uvicorn", "app.main:app", "--host", "0.0.0.0", "--port", "8000"]
```

Build & push to your **on-prem registry**:

```
docker build -t registry.local/fastapi-minio-webhook:1.0
docker push registry.local/fastapi-minio-webhook:1.0
```

4️⃣ Kubernetes Deployment

`deployment.yaml`

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: fastapi-minio-webhook
spec:
  replicas: 2
  selector:
    matchLabels:
      app: fastapi-minio-webhook
  template:
    metadata:
      labels:
        app: fastapi-minio-webhook
    spec:
      containers:
      - name: fastapi
        image: registry.local/fastapi-minio-webhook:1.0
        ports:
        - containerPort: 8000
        env:
        - name: MINIO_TOKEN
          value: "minio-secret-token"
        resources:
          requests:
```

```
            cpu: "100m"
            memory: "128Mi"
          limits:
            cpu: "500m"
            memory: "512Mi"
        readinessProbe:
          httpGet:
            path: /health
            port: 8000
          initialDelaySeconds: 5
        livenessProbe:
          httpGet:
            path: /health
            port: 8000
          initialDelaySeconds: 15
```

Apply:

```
kubectl apply -f deployment.yaml
```

5️⃣ Kubernetes Service (Internal Only)

MinIO will call this via **cluster DNS**.

`service.yaml`

```
apiVersion: v1
kind: Service
metadata:
  name: fastapi-minio-webhook
spec:
  type: ClusterIP
  selector:
    app: fastapi-minio-webhook
  ports:
  - port: 8000
    targetPort: 8000
```

Apply:

```
kubectl apply -f service.yaml
```

Service DNS:

```
http://fastapi-minio-webhook.default.svc.cluster.local:8000
```

6️⃣ Verify from Inside Cluster

```
kubectl run curltest --rm -it \
  --image=curlimages/curl -- sh

curl http://fastapi-minio-webhook:8000/health
```

Expected:

```
{"status":"ok"}
```

7️⃣ Configure MinIO Webhook

Add webhook target

```
mc admin config set myminio notify_webhook:1 \
  endpoint="http://fastapi-minio-webhook:8000/webhook/minio" \
  auth_token="minio-secret-token"

mc admin service restart myminio
```

Attach event to bucket

```
mc event add myminio/landing-bucket \
  arn:minio:sqs::webhook \
  --event put
```

8️⃣ Test End-to-End

Upload a file:

```
mc cp test.csv myminio/landing-bucket/raw/test.csv
```

Check logs:

```
kubectl logs deploy/fastapi-minio-webhook
```

You should see:

```
Event=s3:ObjectCreated:Put, Bucket=landing-bucket, Object=raw/test.csv
```

Production Hardening (Strongly Recommended)

✅ Idempotency

Use `(bucket, object, etag)` stored in Postgres/Redis to avoid duplicates.

✅ Retry / DLQ

If REST call fails:

- retry with backoff
- push event to Kafka / Redis / file queue

✅ Security

- Namespace isolation

- NetworkPolicy (MinIO → FastAPI only)
- Secrets via Vault / K8s Secrets