🛠️ Implementation Guide

This guide helps you build, deploy, and benchmark a high-performance data pipeline using Go, Kafka, MongoDB, and Kubernetes. It works on any Linux or WSL2 environment with Docker and Minikube.

☑️ Prerequisites

- Docker and Docker CLI installed
- Kubernetes CLI (`kubectl`) installed
- Minikube installed
- Go 1.22 or later
- Helm (for deploying Kafka/MongoDB charts)

1️⃣ Environment Setup

Install Go

```
wget https://go.dev/dl/go1.22.2.linux-amd64.tar.gz
sudo tar -C /usr/local -xzf go1.22.2.linux-amd64.tar.gz
echo 'export PATH=$PATH:/usr/local/go/bin' >> ~/.bashrc
source ~/.bashrc
go version
```

Install kubectl

```
curl -LO "https://dl.k8s.io/release/$(curl -s
https://dl.k8s.io/release/stable.txt)/bin/linux/amd64/kubectl"
chmod +x kubectl
sudo mv kubectl /usr/local/bin/
kubectl version --client
```

Install Minikube

```
curl -LO https://storage.googleapis.com/minikube/releases/latest/minikube-
linux-amd64
sudo install minikube-linux-amd64 /usr/local/bin/minikube
```

Start Minikube

```
minikube start --driver=docker --cpus=4 --memory=8192
```

---

2 Clone and Build

```
git clone https://github.com/<your-username>/go-k8s-data-pipeline.git
cd go-k8s-data-pipeline

# Build dummy producer
cd dummy-producer
docker build -t dummy-producer:latest .
minikube image load dummy-producer:latest

# Build ETL consumer
cd ../etl-consumer
docker build -t etl-consumer:latest .
minikube image load etl-consumer:latest
```

---

3 Deploy Infrastructure

Deploy Kafka

```
kubectl create namespace kafka
helm repo add bitnami https://charts.bitnami.com/bitnami
helm install kafka bitnami/kafka --namespace kafka
```

Deploy MongoDB

```
kubectl create namespace mongo
helm install mongodb bitnami/mongodb --namespace mongo
```

---

4 Deploy Apps

```
kubectl apply -f k8s/producer-deployment.yaml
kubectl apply -f k8s/consumer-deployment.yaml
```

---

5 Benchmarking the Pipeline

```
chmod +x scripts/benchmark.sh
./scripts/benchmark.sh
```

This script fetches the last 100 lines from the ETL consumer logs and counts the number of messages processed.

---

6 Monitoring and Observability

For advanced performance monitoring:

- Use `pprof` in Go
- Deploy Prometheus and Grafana via Helm
- Add dashboards to monitor throughput and memory usage

---

7 Recording Your Demo

You can use screen recording tools like:

- **OBS Studio** (Open Source)
- **Kazam** or **SimpleScreenRecorder** on Linux

Record:

- Minikube dashboard
- Pod logs and metrics
- Benchmark results

📌 Notes

- This setup uses open-source tools and runs entirely on your local machine.
- Ideal for prototyping, learning, or showcasing ETL pipelines.
- Adapt deployment YAMLs and Dockerfiles for cloud environments (EKS, AKS, GKE).