Deploying Llama 3.2 1B with Ollama on Kubernetes

This guide covers deploying the Llama 3.2 1B model using Ollama on Kubernetes via a custom Helm chart and ArgoCD.

Files Created

```
k8s-watchdog/
├── helm/
│   └── ollama/
│       ├── Chart.yaml
│       ├── values.yaml
│       └── templates/
│           ├── _helpers.tpl
│           ├── namespace.yaml
│           ├── deployment.yaml
│           ├── service.yaml
│           └── pvc.yaml
└── argocd/
    └── ollama-app.yaml
```

Helm Chart Overview

Chart.yaml

- **Name**: ollama
- **Version**: 0.1.0
- **AppVersion**: latest

Default Values (values.yaml)

Parameter	Default	Description
`model.name`	`llama3.2:1b`	Model to pull on startup
`model.pullOnStartup`	`true`	Auto-pull model via init container
`service.type`	`ClusterIP`	Service type
`service.port`	`11434`	Ollama API port
`resources.requests.memory`	`2Gi`	Memory request
`resources.limits.memory`	`4Gi`	Memory limit

`resources.requests.cpu`	`1`	CPU request
`resources.limits.cpu`	`2`	CPU limit
`persistence.enabled`	`false`	Use PVC for model storage
`namespace`	`monitoring`	Target namespace

Deployment Methods

Option 1: Direct Helm Install (Testing)

```
# Install the chart
helm install ollama helm/ollama --namespace monitoring --create-namespace

# Check status
kubectl get pods -n monitoring -l app.kubernetes.io/name=ollama

# Watch logs (init container pulls the model)
kubectl logs -n monitoring -l app.kubernetes.io/name=ollama -c pull-model -f

# Check main container
kubectl logs -n monitoring -l app.kubernetes.io/name=ollama -c ollama -f
```

Option 2: ArgoCD (Production)

1. **Update the ArgoCD Application manifest** (`argocd/ollama-app.yaml`):

```yaml

spec:

source:

repoURL: https://github.com/YOUR_USERNAME/k8s-watchdog.git # <-- Update this

```

2. **Push to git**:

```bash

git add .

git commit -m "Add Ollama Helm chart and ArgoCD application"

git push

```

3. **Apply the ArgoCD Application**:

```bash

kubectl apply -f argocd/ollama-app.yaml

```

4. **Check sync status**:

```bash
argocd app get ollama
# or via kubectl
kubectl get application ollama -n argocd
```

Testing the LLM Endpoint

Port Forward to Local Machine

```
kubectl port-forward -n monitoring svc/ollama 11434:11434
```

Test API Endpoints

Check if Ollama is running:

```
curl http://localhost:11434/
# Expected: "Ollama is running"
```

List available models:

```
curl http://localhost:11434/api/tags
```

Test text generation:

```
curl http://localhost:11434/api/generate -d '{
  "model": "llama3.2:1b",
  "prompt": "Explain this Kubernetes error in one sentence: CrashLoopBackOff",
  "stream": false
}'
```

Test with K8s error analysis prompt:

```
curl http://localhost:11434/api/generate -d '{
  "model": "llama3.2:1b",
  "prompt": "You are a Kubernetes expert. Analyze this error and provide a 1-sentence expl
anation:\n\nError: Back-off restarting failed container\nReason: CrashLoopBackOff\nMessage
: back-off 5m0s restarting failed container=nginx pod=nginx-7bf8c77b5b-x2j4k_default\n\nEx
planation:",
  "stream": false
}'
```

Chat completion format:

```
curl http://localhost:11434/api/chat -d '{
  "model": "llama3.2:1b",
  "messages": [
    {
```

```
      "role": "system",
      "content": "You are a Kubernetes expert. Provide brief, actionable explanations."
    },
    {
      "role": "user",
      "content": "What does OOMKilled mean?"
    }
  ],
  "stream": false
}'
```

Customizing the Deployment

Enable Persistent Storage

Create a custom values file (`values-production.yaml`):

```
persistence:
  enabled: true
  size: 20Gi
  storageClass: "standard"  # or your storage class

resources:
  requests:
    memory: "4Gi"
    cpu: "2"
  limits:
    memory: "8Gi"
    cpu: "4"
```

Install with custom values:

```
helm install ollama helm/ollama -f values-production.yaml
```

Enable GPU Support

```
gpu:
  enabled: true

# Add to nodeSelector if needed
nodeSelector:
  nvidia.com/gpu: "true"

# Add tolerations for GPU nodes
tolerations:
  - key: "nvidia.com/gpu"
    operator: "Exists"
    effect: "NoSchedule"
```

Use a Different Model

```
model:
  name: "llama3.2:3b"  # Larger model
  pullOnStartup: true
```

Troubleshooting

Check Pod Status

```
kubectl get pods -n monitoring -l app.kubernetes.io/name=ollama
kubectl describe pod -n monitoring -l app.kubernetes.io/name=ollama
```

Check Init Container (Model Pull)

```
# Logs from model pull
kubectl logs -n monitoring -l app.kubernetes.io/name=ollama -c pull-model

# If init container is stuck
kubectl describe pod -n monitoring -l app.kubernetes.io/name=ollama | grep -A 20 "Init Con
tainers"
```

Check Main Container

```
kubectl logs -n monitoring -l app.kubernetes.io/name=ollama -c ollama
```

Common Issues

Issue	Cause	Solution
Init container timeout	Slow network/large model	Increase timeout or use smaller model
OOMKilled	Insufficient memory	Increase memory limits
Pod pending	No resources	Check node capacity, adjust requests
Model not found	Pull failed	Check init container logs

Manually Pull Model (if init fails)

```
# Exec into the pod
kubectl exec -it -n monitoring deployment/ollama -- /bin/sh

# Pull model manually
ollama pull llama3.2:1b

# Verify
ollama list
```

Service Endpoints

Once deployed, Ollama is accessible within the cluster at:

```
http://ollama.monitoring.svc:11434
```

API Endpoints:

- `GET /` - Health check
- `GET /api/tags` - List models
- `POST /api/generate` - Generate text
- `POST /api/chat` - Chat completion
- `POST /api/embeddings` - Generate embeddings

Integration with K8s Watchdog

The watchdog service will connect to Ollama using:

```
LLM_ENDPOINT = "http://ollama.monitoring.svc:11434"
LLM_MODEL = "llama3.2:1b"
```

Example analysis request from watchdog:

```
import requests

response = requests.post(
    f"{LLM_ENDPOINT}/api/generate",
    json={
        "model": LLM_MODEL,
        "prompt": f"Analyze this K8s error: {error_message}",
        "stream": False
    },
    timeout=10
)
summary = response.json()["response"]
```

Uninstall

Helm

```
helm uninstall ollama -n monitoring
```

ArgoCD

```
kubectl delete application ollama -n argocd
# or
argocd app delete ollama
```