

Kubernetes YAML manifests for your **Node.js application deployment** on EKS:

✓ 1. deployment.yaml – Node.js App Deployment

yaml

CopyEdit

```
apiVersion: apps/v1
```

```
kind: Deployment
```

```
metadata:
```

```
  name: nodejs-app
```

```
  labels:
```

```
    app: nodejs-app
```

```
spec:
```

```
  replicas: 2
```

```
  selector:
```

```
    matchLabels:
```

```
      app: nodejs-app
```

```
  template:
```

```
    metadata:
```

```
      labels:
```

```
        app: nodejs-app
```

```
    spec:
```

```
      containers:
```

```
        - name: nodejs-app
```

```
          image: <your-ecr-or-dockerhub-image>
```

```
          ports:
```

```
            - containerPort: 3000
```

```
          envFrom:
```

```
            - configMapRef:
```

```
              name: nodejs-config
```

- secretRef:
 name: nodejs-secrets

✓ 2. service.yaml – LoadBalancer Service

Use LoadBalancer if you want public access via an external IP or domain.

yaml

CopyEdit

apiVersion: v1

kind: Service

metadata:

 name: nodejs-service

spec:

 selector:

 app: nodejs-app

 ports:

 - protocol: TCP

 port: 80

 targetPort: 3000

 type: LoadBalancer

Change type: ClusterIP if you only need internal access (inside cluster).

✓ 3. configmap.yaml – App Configuration (Optional)

yaml

CopyEdit

apiVersion: v1

kind: ConfigMap

metadata:

 name: nodejs-config

data:

NODE_ENV: production

APP_PORT: "3000"

☒ 4. secret.yaml – Sensitive Env Vars (Optional)

yaml

CopyEdit

apiVersion: v1

kind: Secret

metadata:

name: nodejs-secrets

type: Opaque

data:

DB_USER: bXl1c2Vy # base64 encoded

DB_PASS: c2VjcmV0 # base64 encoded

⚠ Encode your secret values using:

bash

CopyEdit

echo -n 'your-value' | base64

☒ Deploy All

bash

CopyEdit

kubectl apply -f configmap.yaml

kubectl apply -f secret.yaml

kubectl apply -f deployment.yaml

kubectl apply -f service.yaml

Would you like to add:

- ☒ HPA (auto-scaling)?

- ☒ Ingress (ALB)?
- ☒ PVC or EFS (storage)?
- ☒ Helm chart structure?

Let me know, I'll provide that next.

You said:

yes

ChatGPT said:

Great! Let's now extend your Kubernetes deployment with additional production-grade components for your Node.js app on EKS:

☒ 5. Horizontal Pod Autoscaler (HPA) – Auto-Scaling

This automatically scales your pods based on CPU usage.

hpa.yaml

yaml

CopyEdit

```
apiVersion: autoscaling/v2
```

```
kind: HorizontalPodAutoscaler
```

```
metadata:
```

```
  name: nodejs-app-hpa
```

```
spec:
```

```
  scaleTargetRef:
```

```
    apiVersion: apps/v1
```

```
    kind: Deployment
```

```
    name: nodejs-app
```

```
  minReplicas: 2
```

```
  maxReplicas: 5
```

```
  metrics:
```

```
    - type: Resource
```

```
      resource:
```

name: cpu

target:

type: Utilization

averageUtilization: 50

✓ Requires metrics-server to be installed in the cluster.

✓ 6. Ingress with ALB Ingress Controller (Optional)

You can route traffic via domain like app.mydomain.com.

ingress.yaml

yaml

CopyEdit

apiVersion: networking.k8s.io/v1

kind: Ingress

metadata:

name: nodejs-app-ingress

annotations:

alb.ingress.kubernetes.io/scheme: internet-facing

alb.ingress.kubernetes.io/target-type: ip

alb.ingress.kubernetes.io/listen-ports: '[{"HTTP": 80}]'

alb.ingress.kubernetes.io/group.name: nodejs-app-group

spec:

ingressClassName: alb

rules:

- host: app.example.com

http:

paths:

- path: /

pathType: Prefix

backend:

```
service:
  name: nodejs-service
  port:
    number: 80
```

Replace app.example.com with your Route53 domain.

ALB Ingress Controller must be installed and configured with proper IAM service account.

✓ 7. Persistent Volume (EBS or EFS) – Optional Storage

If your app needs to store data, use PVC. EFS is ideal for shared storage.

Example EFS-backed pvc.yaml

```
yaml
CopyEdit
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: nodejs-pvc
spec:
  accessModes:
    - ReadWriteMany
  storageClassName: efs-sc
  resources:
    requests:
      storage: 5Gi
```

You'll need EFS CSI driver and StorageClass efs-sc configured.

✓ 8. Logs & Monitoring with CloudWatch (via Fluent Bit)

Install **CloudWatch agent** or **Fluent Bit** DaemonSet to ship logs.

```
bash
```

CopyEdit

```
helm repo add fluent https://fluent.github.io/helm-charts
```

```
helm upgrade --install fluent-bit fluent/fluent-bit \
```

```
--namespace kube-system \
```

```
--set cloudwatch.enabled=true \
```

```
--set cloudwatch.region=us-east-1 \
```

```
--set cloudwatch.logGroupName=/eks/nodejs-app
```

✓ 9. Optional: Helm Chart Structure

Want to package this app as a Helm chart?

bash

CopyEdit

```
helm create nodejs-app
```

Directory structure:

pgsql

CopyEdit

nodejs-app/

├── charts/

├── templates/

| ├── deployment.yaml

| ├── service.yaml

| ├── configmap.yaml

| ├── secret.yaml

| ├── ingress.yaml

| ├── hpa.yaml

└── values.yaml

You can then install:

bash

CopyEdit

```
helm install nodejs-app ./nodejs-app
```

✓ Deployment Order (Standard)

bash

CopyEdit

```
kubectl apply -f configmap.yaml
```

```
kubectl apply -f secret.yaml
```

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

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

```
kubectl apply -f hpa.yaml
```

```
kubectl apply -f ingress.yaml # if using ALB + domain
```

```
kubectl apply -f pvc.yaml # if using EFS/EBS
```