📦 3-Tier Kubernetes Application Deployment

This is a **3-tier web application** composed of:

- **Frontend**: ReactJS
- **Backend**: NodeJS (Express)
- **Database**: MongoDB

🐳 Docker Build & Push

Frontend
```bash
docker login
cd ./frontend
docker build -t <username>/frontend .
docker push <username>/frontend
```

Backend
```bash
cd ./backend
docker build -t <username>/backend .
docker push <username>/backend
```

☸️ Kubernetes Deployment

🔧 Pre-setup

1. **Install NGINX Ingress Controller**
```bash
helm repo add ingress-nginx https://kubernetes.github.io/ingress-nginx
helm repo update
```

```
# Deploy as a Deployment
helm install ingress-nginx ingress-nginx/ingress-nginx \
  --namespace ingress-nginx --create-namespace \
  --set controller.kind=Deployment \
  --set controller.service.type=NodePort \
  --set controller.hostPort.enabled=true

# OR deploy as a DaemonSet
helm install ingress-nginx ingress-nginx/ingress-nginx \
  --namespace ingress-nginx --create-namespace \
  --set controller.kind=DaemonSet \
  --set controller.service.type=NodePort \
  --set controller.hostPort.enabled=true
```

2. **Check Ingress Resources**
```bash
kubectl get pods -n ingress-nginx
kubectl get svc -n ingress-nginx
kubectl get ingressclass
```

3. **Install AWS CLI**
```bash
aws configure
# Ensure proper access policy to use EBS for dynamic PV
```

4. **Build & Deploy Cluster Health Check App**
```bash
cd ./k8s/cluster_healtz/
# Follow instructions in README.md
```

🚀 K8S Deployment Order

1. Database
```bash
cd ./k8s/database
kubectl apply -f mongo-secret.yaml
kubectl apply -f mongo-storageclass.yaml
kubectl apply -f mongo-statefulset.yaml
```

2. Backend
```bash
cd ./k8s/backend
kubectl apply -f app-secrets.yaml
kubectl apply -f backend-deployment.yaml
kubectl apply -f backend-service.yaml
kubectl apply -f backend-ingress.yaml
```

3. Frontend
```bash
cd ./k8s/frontend
kubectl apply -f frontend-deployment.yaml
kubectl apply -f frontend-service.yaml
kubectl apply -f frontend-ingress.yaml
```

☁️ AWS ALB Configuration

- Create an **ALB** with a target group pointing to your K8s **EC2 node instances**
- Use **HTTPS** (with ACM SSL Certificate) on port that maps to **Ingress NodePort**
- Add **Route53 A-record (alias)** pointing to the ALB

Ingress Cloud load balancers are costly as most of the time billing will be per request so to avoid this the Kubernetes solution is Ingress, we can run an external software-based load balancer in our cluster.

1. Ingress controller

- This controller is used to execute the ingress resources which contain routing rules and bring the external traffic based on routing rules to the internal service.
- This controller will automatically monitor the existing resources and also identify new ingress resources.
- This will be a third-party controller (tools) which we need to install one time in our cluster as a controller.
- We are using nginx as an ingress controller.

2. Ingress resources

- In k8s Ingress resource is type of object which is used to define routing rules based on path of incoming traffic to internal cluster service.
- API for ingress resource networking.k8s.io/v1
- Ingress can be used for reverse proxy means to expose multiple services under the same IP.
- Ingress can be used to delete SSL/TLS certificates.

1. install helm (one-time setup)
https://helm.sh/docs/intro/install/

2. Install the nginx ingress controller with helm (one-time setup)
```
helm install my-release oci://ghcr.io/nginxinc/charts/nginx-ingress --version 1.0.0 --set controller.kind=daemonset --set rbac.create=true
```

3. Copy ingress LoadBalancer service IP address
```
kubectl get svc
```

4. Create a mapping of some hostname to ingress controller LB - IP
```
sudo vi /etc/hosts
add <lb-ip> app.example.com
```

5. Create pods and related services using ingress/deployment.yml
kubectl apply -f deployment.yml

6. create routing rule by applying ingress resource also known as Ingress use ingress.yml
kubectl apply -f ingress.yml

7. curl app.example.com - To get the IP address of the pod
curl app.example.com/name - To get the name defined in the app