### **1. Cloud Provider: AWS**

We’ll use:

* EKS (Elastic Kubernetes Service) for container orchestration
* ALB for load balancing
* EC2 (or Fargate) for compute
* RDS (or containerized PostgreSQL) for database
* CloudWatch for monitoring
* IAM roles & policies for security

### **2. Infrastructure as Code (IaC) with Terraform**

Create a Terraform setup for:

* VPC, subnets, route tables
* EKS cluster + worker nodes
* Security groups, IAM roles
* RDS/PostgreSQL (optional if not containerized)
* S3 bucket for artifact storage
* ALB

#### **Example Folder Structure:**

terraform/

- main.tf

- variables.tf

- outputs.tf

- eks/

- vpc/

- rds/

### **3. Containerization with Docker**

Create Dockerfiles for each service:

Frontend (Dockerfile):

FROM node:18

WORKDIR /app

COPY . .

RUN npm install && npm run build

CMD ["npm", "start"]

Backend (Dockerfile):

FROM node:18

WORKDIR /app

COPY . .

RUN npm install

CMD ["node", "server.js"]

### **4. Kubernetes Deployment (EKS)**

Write Kubernetes manifest

k8s/

- namespace.yaml

- frontend-deployment.yaml

- frontend-service.yaml

- backend-deployment.yaml

- backend-service.yaml

- postgres-deployment.yaml (if needed)

- ingress.yaml (for ALB)

- hpa.yaml (for auto-scaling)

Use ALB Ingress Controller for load balancing.

### **5. CI/CD Pipeline (GitHub Actions or CodePipeline)**

#### **GitHub Actions Example Workflow:**

.github/workflows/deploy.yml

name: CI/CD Pipeline

on:

push:

branches:

- main

jobs:

build\_and\_push:

runs-on: ubuntu-latest

steps:

- name: Checkout

uses: actions/checkout@v2

- name: Build and Push Docker Images

run: |

docker build -t <frontend-image> ./frontend

docker build -t <backend-image> ./backend

docker push <frontend-image>

docker push <backend-image>

deploy:

needs: build\_and\_push

runs-on: ubuntu-latest

steps:

- name: Configure kubectl

run: |

aws eks update-kubeconfig --region us-east-1 --name my-cluster

- name: Deploy to EKS

run: |

kubectl apply -f k8s/

### **6. IAM Roles and Security**

* Use IAM Roles for Service Accounts (IRSA) in EKS.
* Set up Security Groups for ALB, EC2, RDS.
* Apply least privilege policies in IAM roles.
* Enable RBAC in Kubernetes.

### **7. Auto-Scaling**

* Use Kubernetes HPA (Horizontal Pod Autoscaler).
* Use Cluster Autoscaler on EKS worker nodes.
* Enable EC2 Auto Scaling Groups if not using Fargate.

### **8. Load Balancer**

* Use AWS ALB with ALB Ingress Controller
* Configure health checks, SSL termination (via ACM)

### **9. GitHub Repository Structure**

project-root/

│

├── terraform/

├── k8s/

├── frontend/

│ └── Dockerfile

├── backend/

│ └── Dockerfile

├── .github/

│ └── workflows/

│ └── deploy.yml

└── README.md