### **Phase 1: Environment and Initial Application Setup**



**# grp2-opentelemetry-demo**

**Prerequisites Setup**

Install AWS CLI ->

brew install awscli

Install Terraform ->

brew install terraform

Verify installations ->

aws --version

terraform --version

**Configure AWS CLI**

aws configure

Enter your AWS credentials ->

AWS Access Key ID:

AWS Secret Access Key:

Default region: \*\*us-east-1\*\*

Default output format: \*\*json\*\*

**Step 1: Create Project Structure and the necessary Terraform Configuration Files**

All the necessary files are Pushed to GitHub

**Step 2: Initialize Terraform Project**

terraform init

**Step 3: Review and Validate the Terraform Plan**

terraform plan

**Step 4: Apply the Terraform Configuration**

terraform apply

**Step 5: Implementing Docker Deployment (Part 1)**

After the infrastructure is deployed:

1. Connect to the Docker EC2 instance

ssh -i keys/opentelemetry-key.pem ec2-user@$(terraform output -raw docker\_instance\_public\_ip)

2. Verify Docker installation

docker --version

docker-compose --version

3. Check if OpenTelemetry demo is running

cd ~/opentelemetry-demo

docker ps

If needed, start the Docker Compose setup

docker-compose up -d

4. Verify all services are running

docker ps --format "table {{.Names}}\t{{.Status}}"

5. Check Docker logs

docker-compose logs

6. Access the application endpoints

# Get public IP

curl http://169.254.169.254/latest/meta-data/public-ipv4

# You can now access the application in your browser at - http://<public-ip>:8080

7. Take screenshots of ->

- EC2 instance details

- Running docker containers

- Docker logs

- Application frontend

**Step 6: Implementing Kubernetes Setup (Part 2)**

1. Connect to the EKS client instance

ssh -i keys/opentelemetry-key.pem ec2-user@$(terraform output -raw eks\_client\_public\_ip)

2. Verify tools installation

kubectl version --client

eksctl version

aws --version

3. Check EKS cluster connection

kubectl get nodes

4. Deploy OpenTelemetry Demo to EKS

# Create namespace

kubectl create namespace opentel-demo

# Get the manifest file

curl -LO https://raw.githubusercontent.com/open-telemetry/opentelemetry-demo/main/kubernetes/opentelemetry-demo.yaml

# Apply the manifest

kubectl apply -f opentelemetry-demo.yaml -n opentel-demo

5. Monitor deployment progress

kubectl get pods -n opentel-demo -w

6. Check all resources

kubectl get all -n opentel-demo

7. Get logs from key pods

# Example: Get logs from the frontend pod

kubectl logs $(kubectl get pods -n opentel-demo -l app=frontend -o name | head -1) -n opentel-demo

8. Access application using port-forwarding

# Forward the frontend service port to your local machine

kubectl port-forward svc/frontend -n opentel-demo 8080:8080

Then access the application at http://localhost:8080 in your browser.

9. Take screenshots of ->

- EKS cluster details

- Pod status

- Service details

- Application frontend accessed through port-forwarding

**Step 7: Clean Up**

# Connect to Docker instance

ssh -i keys/opentelemetry-key.pem ec2-user@$(terraform output -raw docker\_instance\_public\_ip)

# Stop Docker containers

cd ~/opentelemetry-demo

docker-compose down

# Exit the instance

exit

# To destroy all resources when finished with the project

terraform destroy