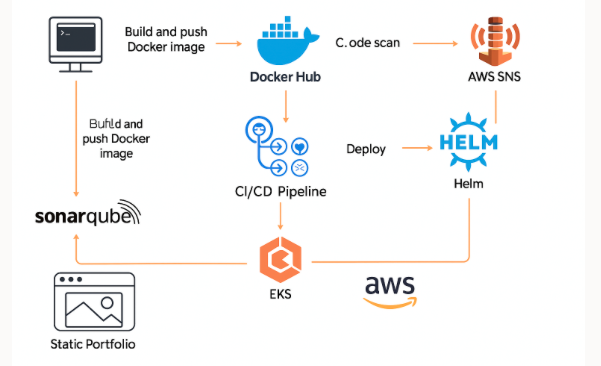
**Deploying a static portfolio to Amazon EKS using Docker, Helm, CloudFormation, GitHub Actions, SonarQube, SNS, and CloudWatch.**

Here's a fully deployable, production-grade DevOps project that integrates:

* ✅ EKS with Helm for container orchestration
* 🐳 Docker for containerization
* 📦 CloudFormation for AWS infrastructure provisioning
* 🔍 SonarQube for code quality scanning
* 📣 AWS SNS for CI/CD notifications
* 📊 AWS CloudWatch for container and cluster observability
* 🚀 GitHub Actions for CI/CD



**📁 GitHub Repo Structure**

static-portfolio/

├── Dockerfile

├── index.html

├── helm-chart/

│ ├── Chart.yaml

│ ├── values.yaml

│ └── templates/

│ ├── deployment.yaml

│ └── service.yaml

├── cloudformation/

│ ├── vpc.yaml

│ ├── iam-roles.yaml

│ ├── eks-cluster.yaml

│ └── eks-nodegroup.yaml

├── .github/

│ └── workflows/

│ └── ci-cd.yaml

└── deploy.sh

**Prerequisite:**

**🧰 Local Setup**

**✅ 1. Development Tools**

* **Git**: for version control
* **Docker**: to build and test your container locally
* **Helm**: to package and deploy your app to Kubernetes
* **kubectl**: to interact with your EKS cluster
* **AWS CLI**: to deploy CloudFormation templates and configure EKS
* **SonarScanner CLI**: for local SonarQube testing (optional)

**📣 SNS Setup**

**✅ 8. SNS Topic**

* Create a topic (e.g., **approval-pipeline-2025**)
* Copy the **Topic ARN** for GitHub Secrets

**🔐 GitHub Secrets**

Add the following secrets to your GitHub repo:

| **Secret Name** | **Description** |
| --- | --- |
| DOCKERHUB\_USERNAME | DockerHub username |
| DOCKERHUB\_PASSWORD | DockerHub password or token |
| SONAR\_TOKEN | SonarQube token |
| AWS\_ACCESS\_KEY\_ID | IAM access key ID |
| AWS\_SECRET\_ACCESS\_KEY | IAM secret access key |
| SNS\_TOPIC\_ARN | ARN of AWS SNS topic |

**✅ Optional: CloudWatch Container Insights**

* Enable via AWS Console or CLI:

aws eks update-cluster-config \

--name StaticPortfolioCluster \

--logging '{"clusterLogging":[{"types":["api","audit","authenticator"],"enabled":true}]}'

**📦 Helm Chart**

**helm-chart/Chart.yaml**

apiVersion: v2

name: static-portfolio

version: 0.1.0

**helm-chart/values.yaml**

image:

repository: <DOCKERHUB\_USERNAME>/static-portfolio

tag: latest

service:

type: LoadBalancer

port: 80

**helm-chart/templates/deployment.yaml**

apiVersion: apps/v1

kind: Deployment

metadata:

name: static-portfolio

spec:

replicas: 1

selector:

matchLabels:

app: static-portfolio

template:

metadata:

labels:

app: static-portfolio

spec:

containers:

- name: static-portfolio

image: "{{ .Values.image.repository }}:{{ .Values.image.tag }}"

ports:

- containerPort: 80

resources:

requests:

cpu: "100m"

memory: "128Mi"

limits:

cpu: "250m"

memory: "256Mi"

**helm-chart/templates/service.yaml**

apiVersion: v1

kind: Service

metadata:

name: static-portfolio

spec:

type: {{ .Values.service.type }}

selector:

app: static-portfolio

ports:

- protocol: TCP

port: {{ .Values.service.port }}

targetPort: 80

**🚀 GitHub Actions CI/CD**

**.github/workflows/ci-cd.yaml**

name: CI/CD to EKS via Helm

on:

push:

branches: [main]

jobs:

build-scan-push-deploy:

runs-on: ubuntu-latest

steps:

- name: Checkout code

uses: actions/checkout@v3

- name: Build Docker image

run: docker build -t ${{ secrets.DOCKERHUB\_USERNAME }}/static-portfolio:latest .

- name: Login to DockerHub

run: echo "${{ secrets.DOCKERHUB\_PASSWORD }}" | docker login -u "${{ secrets.DOCKERHUB\_USERNAME }}" --password-stdin

- name: Push Docker image

run: docker push ${{ secrets.DOCKERHUB\_USERNAME }}/static-portfolio:latest

- name: Run SonarQube scan

env:

SONAR\_TOKEN: ${{ secrets.SONAR\_TOKEN }}

run: |

curl -sSLo sonar-scanner.zip https://binaries.sonarsource.com/Distribution/sonar-scanner-cli/sonar-scanner-cli-5.0.1.3006-linux.zip

unzip sonar-scanner.zip

./sonar-scanner-\*/bin/sonar-scanner

- name: Configure AWS credentials

uses: aws-actions/configure-aws-credentials@v2

with:

aws-access-key-id: ${{ secrets.AWS\_ACCESS\_KEY\_ID }}

aws-secret-access-key: ${{ secrets.AWS\_SECRET\_ACCESS\_KEY }}

aws-region: us-east-1

- name: Setup kubectl

run: aws eks update-kubeconfig --name StaticPortfolioCluster --region us-east-1

- name: Deploy via Helm

run: helm upgrade --install static-portfolio ./helm-chart --namespace default

- name: Notify success via SNS

run: |

aws sns publish \

--topic-arn ${{ secrets.SNS\_TOPIC\_ARN }} \

--message "✅ CI/CD succeeded for branch ${{ github.ref\_name }}"

notify-failure:

if: failure()

runs-on: ubuntu-latest

steps:

- name: Configure AWS credentials

uses: aws-actions/configure-aws-credentials@v2

with:

aws-access-key-id: ${{ secrets.AWS\_ACCESS\_KEY\_ID }}

aws-secret-access-key: ${{ secrets.AWS\_SECRET\_ACCESS\_KEY }}

aws-region: us-east-1

- name: Notify failure via SNS

run: |

aws sns publish \

--topic-arn ${{ secrets.SNS\_TOPIC\_ARN }} \

--message "❌ CI/CD failed for branch ${{ github.ref\_name }}"

**🔐 Required GitHub Secrets**

| **Secret Name** | **Description** |
| --- | --- |
| DOCKERHUB\_USERNAME | DockerHub username |
| DOCKERHUB\_PASSWORD | DockerHub password or token |
| SONAR\_TOKEN | SonarQube token |
| AWS\_ACCESS\_KEY\_ID | IAM access key ID |
| AWS\_SECRET\_ACCESS\_KEY | IAM secret access key |
| SNS\_TOPIC\_ARN | ARN of AWS SNS topic |

**📊 CloudWatch Integration**

* Enabled via eks-cluster.yaml logging block
* You can add container insights via AWS Console or CLI:

aws eks update-cluster-config \

--name StaticPortfolioCluster \

--logging '{"clusterLogging":[{"types":["api","audit","authenticator"],"enabled":true}]}'

Would you like me to generate a matching README.md and starter commit message for this repo? I can also help you write a STAR-format portfolio story based on this project.

**🧭 Phase 1: Prepare Your Environment**

**✅ Step 1: Create Your Static Site**

* Build a simple index.html page with your portfolio content.
* Add a Dockerfile to containerize it:

**✅ Step 2: Create a GitHub Repository**

* Name it something like static-portfolio-with eks
* Push your code: index.html, Dockerfile, and folder structure.

**🏗️ Phase 2: Provision AWS Infrastructure with CloudFormation**

**☁️ AWS Setup**

**✅ 1. AWS Account**

* Must have access to:
  + **CloudFormation**
  + **EKS**
  + **IAM**
  + **SNS**
  + **CloudWatch**

**✅ 2. IAM Roles**

* **EKSClusterRole**: for EKS control plane
* **EKSNodeRole**: for worker nodes
* **GitHub Actions IAM user**: with programmatic access and permissions for:
  + eks:UpdateKubeconfig
  + cloudformation:\*
  + sns:Publish
  + eks:Describe\*
  + logs:\*

**✅ 3. VPC & Networking**

* VPC with:
  + At least 2 public subnets
  + Security group for EKS cluster and nodes
* You can create this manually or via CloudFormation

**CloudFormation templates** to provision the AWS infrastructure for your static portfolio deployment on **Amazon EKS**, including:

* ✅ VPC with public/private subnets
* ✅ EKS Cluster with CloudWatch logging
* ✅ EKS Node Group
* ✅ IAM roles for cluster and nodes

**Step: Deploy the Templates via script**

**Create** a bash script named **deploy.sh** file and run the script in root of the repo in visual studio terminal:

| chmod +x deploy.sh

| ./deploy.sh

**🐳 Phase 3: Containerize and Push to DockerHub**

* Create a repository (e.g., static-portfolio-witheks)
* Generate a **personal access token** for GitHub Actions

**✅ Step 1: Build and Push Docker Image**

docker build -t your-dockerhub-username/static-portfolio:latest .

docker push your-dockerhub-username/static-portfolio:latest

**📦 Phase 4: Create Helm Chart**

**✅ Step 1: Create Helm Chart Structure**

helm-chart/

├── Chart.yaml

├── values.yaml

└── templates/

├── deployment.yaml

└── service.yaml

**✅ Step 2: Define Kubernetes Resources**

* deployment.yaml: defines the pod and container
* service.yaml: exposes the app via LoadBalancer

**🚀 Phase 5: Set Up CI/CD with GitHub Actions**

**✅ Step 1: Create .github/workflows/ci-cd.yaml**

Include steps to:

* Build and push Docker image
* Run SonarQube scan
* Configure AWS credentials
* Update kubectl config
* Deploy via Helm
* Send SNS notifications

**🔍 Phase 6: Integrate SonarQube**

**✅ Set Up SonarQube Server**

* Use Docker or a cloud-hosted version
* Generate a SONAR\_TOKEN
* Add it to GitHub Secrets

1. Launch EC2 (t2.medium or higher)

aws ec2 run-instances \

--image-id ami-0c02fb55956c7d316 \

--instance-type t2.medium \

--key-name awesome-key-east1-2025 \

--security-group-ids sg-08fab7eae506c8657 \

--tag-specifications 'ResourceType=instance,Tags=[{Key=Name,Value=SonarQube-Host}]' \

--query 'Instances[0].InstanceId' \

--output text

1. Log into the ec2 instance via SSH and Install Docker

| sudo yum update -y

| sudo amazon-linux-extras install docker -y

| sudo service docker start

| sudo systemctl enable docker

| sudo usermod -aG docker ec2-user

| docker info

| sudo systemctl status docker

1. Run SonarQube container (same command as above)

docker run -d --name sonarqube \

-p 9000:9000 \

sonarqube:latest

1. Open port 9000 in the EC2 security group
2. Access via http://<EC2\_PUBLIC\_IP>:9000

**🔐 Create a Project in SonarQube**

1. Log into SonarQube

http://<EC2\_PUBLIC\_IP>:9000

username: admin

password: Onggona1317!

1. Click **Create Project**

Example:

Project Key: StaticPortfolio

Project Name: StaticHTMLPortfolio

Branch: main

1. Generate a **project token**

sqp\_4ecf7c1f7c1614ffe4b191a83fb563ed2dbe4ddc

1. Save this token as SONAR\_TOKEN in GitHub Secrets

**📁 Add SonarQube Config to Your Repo**

Create a file named sonar-project.properties:

properties

sonar.projectKey=StaticPortfolio

sonar.projectName=StaticHTMLPortfolio

sonar.sources=.

sonar.language=html

sonar.host.url=http://18.215.168.156:9000

sonar.login=${{ secrets.SONAR\_TOKEN }}

Replace <SONARQUBE\_HOST> with your server IP or DNS.

**📣 Phase 7: Integrate AWS SNS**

**✅ Step 1: Create SNS Topic**

* In AWS Console → SNS → Create topic
* Add SNS\_TOPIC\_ARN to GitHub Secrets

**📊 Phase 8: Enable CloudWatch Monitoring**

**✅ Step 1: Enable Cluster Logging**

aws eks update-cluster-config \

--name StaticPortfolioCluster \

--logging '{"clusterLogging":[{"types":["api","audit","authenticator"],"enabled":true}]}'

**🔐 Phase 9: Add GitHub Secrets**

| **Secret Name** | **Description** |
| --- | --- |
| DOCKERHUB\_USERNAME | DockerHub username |
| DOCKERHUB\_PASSWORD | DockerHub password or token |
| SONAR\_TOKEN | SonarQube token |
| AWS\_ACCESS\_KEY\_ID | IAM access key ID |
| AWS\_SECRET\_ACCESS\_KEY | IAM secret access key |
| SNS\_TOPIC\_ARN | ARN of AWS SNS topic |

**✅ Final Step: Push to GitHub and Watch It Deploy**

* Commit and push your code to main
* GitHub Actions will:
  + Build and scan your code
  + Push the image to DockerHub
  + Deploy to EKS via Helm
  + Notify via SNS
  + Log activity in CloudWatch

Would you like me to generate a matching README.md, starter commit message, or a STAR-format portfolio story for this project? I can also help you test each phase locally before going live.