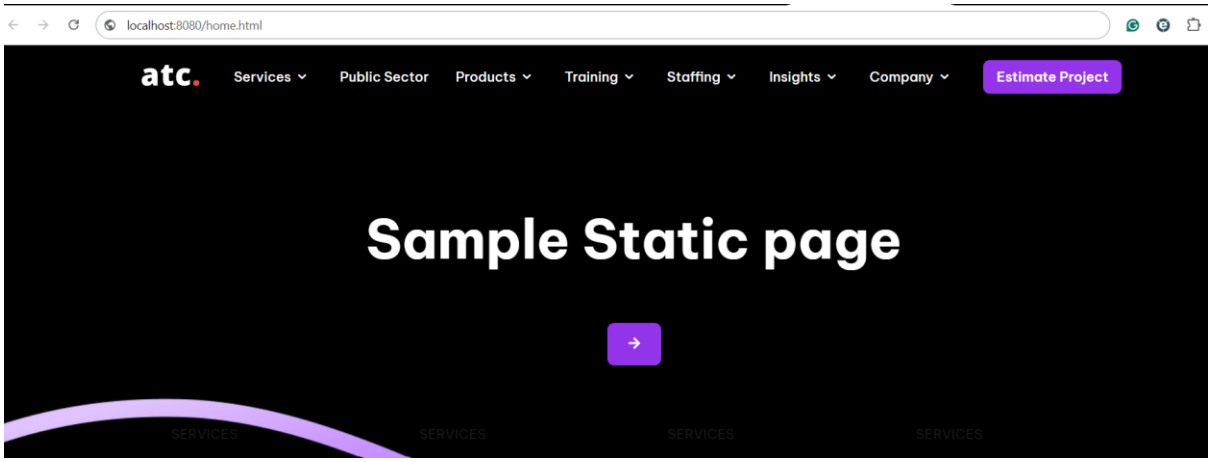
Project: Simple Static Web Application using html and Javascript

The server will start on port 8080. This can access it by navigating to

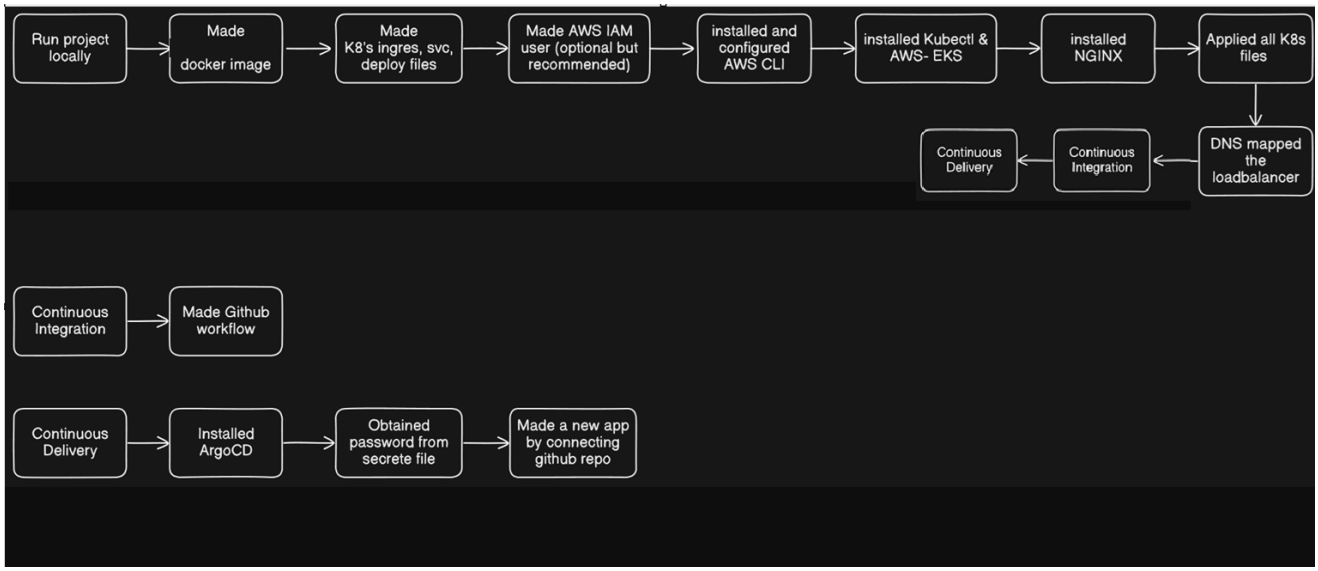
http://localhost:8080/home.html

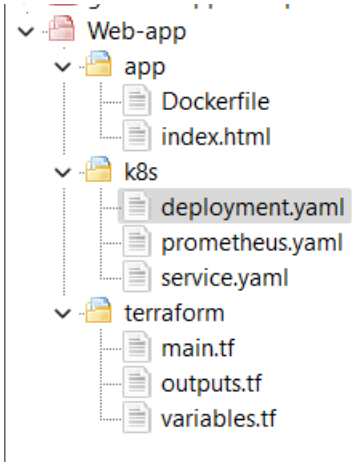


DevOps practices include the following:

- Creating Dockerfile
- Containerization
- Continuous Integration (CI)
- Continuous Deployment (CD)

The Project Flow and folder Structure:





Docker File:

The Dockerfile will be used to containerize a simple static web app. I used nginx for this static HTML page. If it's a EP level Application We can go with *Multi-stage Docker Image (Distroless*)

Deploying the docker Image
Build the Docker image and push it to a container registry (Docker Hub).

 # Use an official NGINX image as the base

FROM nginx:alpine

COPY ./index.html /usr/share/nginx/html/

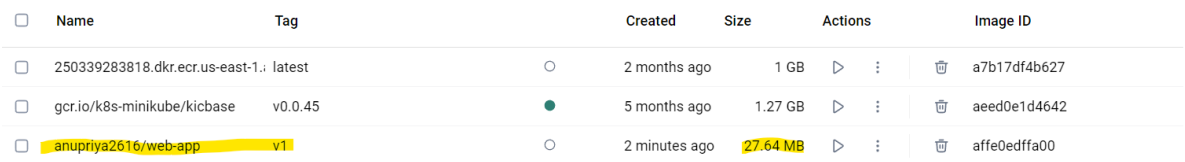
EXPOSE 80

CMD ["nginx", "-g", "daemon off;"]

docker build -t anupriya2616/web-app:v1 .

docker push anupriya2616/web-app:v1

```
PS C:\Users\anupr\go-web-app-devops> docker build -t anupriya2616/web-app:v1
 .
[+] Building 55.5s (7/15)                                  docker:desktop-linux
 => [base 1/6] FROM docker.io/library/golang:1.21@sha256:4746d26432a  50.4s
 => => sha256:532d43a0bc41875119c835fd75616e87d5df2a3714 126B / 126B  19.3s
 => => sha256:4f4fb700ef54461cfa02571ae0db9a0dc1e0cdb55774 32B / 32B  19.8s
 => => extracting sha256:3cbbe86a28c2f6b3c3e0e8c6dcfba369e1ea656cf8da  1.8s
 => => extracting sha256:6ed93aa58a52c9abc1ee472f1ac74b73d3adcccc2c30  7.7s
 => => extracting sha256:1f46bd02dde39f0741d70614fc607bf03c1a0cd60d52  6.8s
 => [stage-1 1/3] FROM gcr.io/distroless/base:latest@sha256:74ddbf52  32.8s
 => => resolve gcr.io/distroless/base:latest@sha256:74ddbf52d93fafbdd  0.1s
 => => sha256:ad04bf079b9ed668d38fe2138cfe57584779598 2.27kB / 2.27kB  0.0s
 => => sha256:fab58a7ef52ea73a8c91e19d80e590a03596ba0 1.71kB / 1.71kB  0.0s
```



Name	Tag		Created	Size	Actions	Image ID
250339283818.dkr.ecr.us-east-1.:	latest	○	2 months ago	1 GB	▷ ⋮	a7b17df4b627
gcr.io/k8s-minikube/kicbase	v0.0.45	●	5 months ago	1.27 GB	▷ ⋮	aeed0e1d4642
anupriya2616/web-app	v1	○	2 minutes ago	27.64 MB	▷ ⋮	affe0edffa00

Kubernetes deployment

Apply the Kubernetes deployment and service:

The Kubernetes deployment and service files will allow you to deploy the web application and expose it.

Deployment.yaml

```yaml
apiVersion: apps/v1 #version 1
kind: Deployment #kind of the file
metadata:
  name: web-app-deployment
spec:
  replicas: 2 #to make sure always the pod will be 2
  selector:
    matchLabels:
      app: web-app
  template:
    metadata:
      labels:
        app: web-app
    spec:
      containers:
        - name: web-app
          image: anupriya2616/web-app #docker image
          ports:
            - containerPort: 80
```

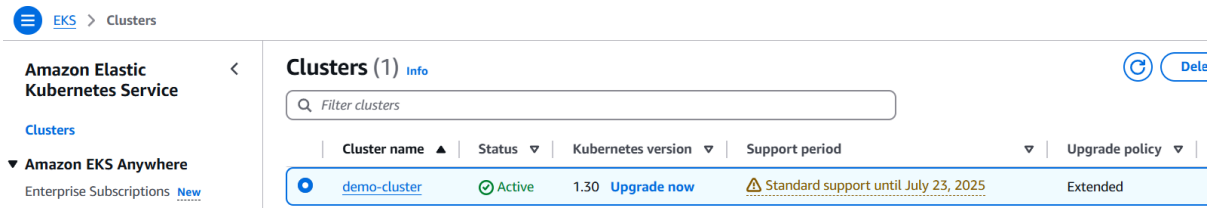
Service.yaml

```yaml
apiVersion: v1
kind: Service
metadata:
  name: web-app-service
spec:
  selector:
    app: web-app
  ports:
    - protocol: TCP  #type of the protocol
      port: 80
      targetPort: 80
  type: LoadBalancer #to expose to the outside IP.
```

kubectl apply -f ./k8s/deployment.yaml

kubectl apply -f ./k8s/service.yaml

```
PS C:\Users\anupr\go-web-app-devops\k8s\manifests> kubectl apply -f deployment.yaml
deployment.apps/go-web-app unchanged
PS C:\Users\anupr\go-web-app-devops\k8s\manifests> kubectl get pods
NAME                                          READY   STATUS    RESTARTS       AGE
go-web-app-5d66858747-94r96                   1/1     Running   0              2m3s
grafana-7f64ccf765-wfz8t                      1/1     Running   4 (4m6s ago)   93d
guestbook-ui-5fbf7fddd6-hkts8                 1/1     Running   2 (4m6s ago)   91d
prometheus-alertmanager-0                     1/1     Running   4 (4m6s ago)   93d
prometheus-kube-state-metrics-75b5bb4bf8-7ddr7   1/1     Running   7 (4m6s ago)   93d
prometheus-prometheus-node-exporter-rhgw7     1/1     Running   4 (4m6s ago)   93d
prometheus-prometheus-pushgateway-84557d6c79-2zrv6   1/1     Running   4 (4m6s ago)   93d
prometheus-server-644d686bc6-hkrll            2/2     Running   8 (4m6s ago)   93d
```



Prometheus monitoring

Add the Prometheus monitoring configurations

You can add a basic Prometheus configuration to monitor the deployment.

Terraform

Run Terraform to create the infrastructure and deploy the application:

Used Terraform to create a Kubernetes cluster (local Minikube), configure Kubernetes, and deploy the resources.

main.tf

- **Purpose**: In this main file I have defined the primary infrastructure resources (EC2 instances,EKS clusters).

variables.tf

- **Purpose**: In this file I defined input variables that can be used in the configuration files. (AWS_region setup)

outputs.tf

- **Purpose**: Used this file to define output values that are displayed after the Terraform apply operation is completed. (web_app_service_url)

terraform init #initializes a working directory containing Terraform configuration files. (aws config)

terraform apply #ensures that the resources specified in the configuration are created/modified.