MANUAL DE DESPLIEGUE

GRUPO NO.12

SISTEMAS OPERATIVOS II

Creación de dockerfile

Para empezar la configuración del despliegue de nuestra aplicación primerose procedió a realizar los dockerfile del backend y del frontend.

Se utiliza la imagen de golang, se crea una carpeta donde se copian todos los archivos necesarios para el servidor, luego de ello se hace una descarga de las librerías.

Se corre el servidor en el puerto 3000

```
# syntax=docker/dockerfile:1
 1
 2
     FROM golang:1.16-alpine
 3
 4
     WORKDIR /app
 5
     COPY go.mod ./
     COPY go.sum ./
 8
     RUN go mod download
10
11
     COPY
12
13
     RUN go build -o /servidor
14
15
     EXPOSE 3000
16
17
     CMD [ "/servidor" ]
18
```

Creación de dockerfile

Se descarga la imagen base de node y luego se crea una carpeta donde se va a ejecutar el frontend.

se copia el package.json y un npm install para crear el node_modules, se copia en la carpeta y se hace un run build.

En la segunda fase se hace la configuración de nginx y se expone en el puerto 3001

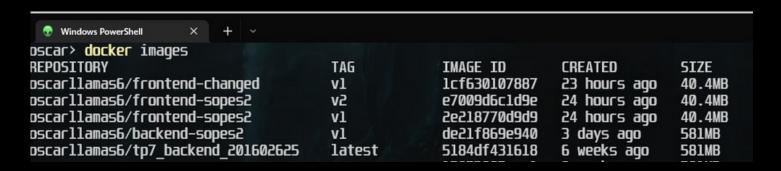
```
# Creando build de la AppWeb
 2
     FROM node:14-alpine as build
     WORKDIR /app
     ENV PATH /app/node_modules/.bin:$PATH
 4
     COPY package.json ./
 5
     RUN npm install
 6
     COPY . ./
 8
     RUN npm run build
 9
     # configurando nginx
10
     FROM nginx:stable-alpine
11
     COPY --from=build /app/dist /usr/share/nginx/html
12
13
     RUN rm /etc/nginx/conf.d/default.conf
14
     # new
15
     COPY nginx/nginx.conf /etc/nginx/conf.d/default.conf
16
17
     EXPOSE 3001
     CMD ["nginx", "-g", "daemon off;"]
18
```

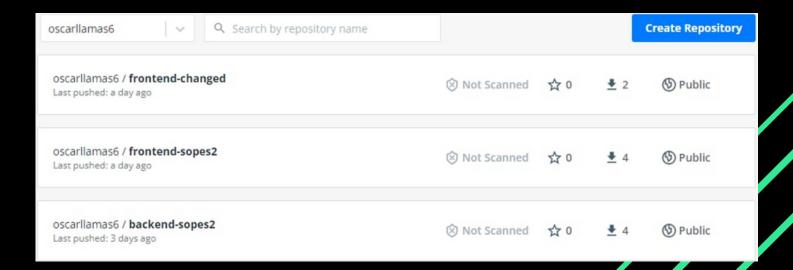
Crear imágenes

Luego de crear los dockerfile se procedió a crear las imágenes, los comandos utilizados fueron los siguientes:

> docker build -t usuario/imagen:latest.

Luego de construir las imágenes se procede a subirlas en dockerhub > docker push usuario/imagen:latest.





Creación de archivos YAML

Creación de los deployments y services

Se crea el un deployment para el backend en el namespace sopes2, de igual forma se crean 3 réplicas y se configura el rollingUpdate y se le asigna la imagen al deployment para crear los contenedores y pods.

```
apiVersion: apps/v1
 2
     kind: Deployment
     metadata:
        name: backend-deployment
 4
       namespace: sopes2
 6
        labels:
          app: backend-deployment
     spec:
 9
        replicas: 3
10
       minReadySeconds: 20
11
        strategy:
12
          type: RollingUpdate
          rollingUpdate:
13
14
            maxSurge: 1
            maxUnavailable: 0
15
        selector:
16
          matchLabels:
17
            app: backend-deployment
18
        template:
19
          metadata:
20
21
            annotations:
              linkerd.io/inject: enabled
22
23
            labels:
              app: backend-deployment
24
25
          spec:
            containers:
26
27

    name: backend-deployment

              image: docker.io/oscarllamas6/backend-sopes2:v1
28
```

Creación de archivos YAML

Se crea el deployment del frontend con la imagen del frontend, de igual forma se crean dos replicas y el rollingUpdate, de igual forma se configuran las variables de entorno.

```
apiVersion: apps/v1
     kind: Deployment
     metadata:
        name: frontend-deployment
       namespace: sopes2
       labels:
         app: frontend-deployment
     spec:
       replicas: 2
       minReadySeconds: 20
11
       strategy:
12
          type: RollingUpdate
13
          rollingUpdate:
14
            maxSurge: 1
            maxUnavailable: 0
15
        selector:
17
         matchLabels:
18
            app: frontend-deployment
        template:
19
20
          metadata:
21
            annotations:
22
              linkerd.io/inject: enabled
            labels:
23
24
              app: frontend-deployment
          spec:
25
26
            containers:
27
            - name: frontend-deployment
              image: docker.io/oscarllamas6/frontend-sopes2:v2
28
              imagePullPolicy: Always
29
30
              ports:
31
              - containerPort: 3001
```

Creación de archivos YAML - services

Se crean los servicios de ClusterIP para el backend y un loadBalancer para el frontend.

```
apiVersion: v1
     kind: Service
     metadata:
       name: backend-service
       namespace: sopes2
       labels:
         app: backend-deployment
     spec:
       type: ClusterIP
       ports:
11
          - port: 3000
12
           targetPort: 3000
           protocol: TCP
13
       selector:
14
         app: backend-deployment
15
```

```
apiVersion: v1
1
     kind: Service
     metadata:
       name: frontend-service
4
5
       namespace: sopes2
     spec:
       selector:
         app: frontend-deployment
       ports:
10
         port: 3001
           targetPort: 3001
11
       type: LoadBalancer
12
```

Configuración de Kubernetes

Setear gcloud y kubectl

Instalar gcloud, correr el siguiente comando en Windows Powerll y ejecutar instalador

```
# Correr comando para iniciar configuración
> gcloud init

#5e mostraraá un mensaje como el siguiente, darle "Y" para aceptar y loggearnos en GCP

Network diagnostic detects and fixes local network connection issues.
Checking network connection...done.
Reachability Check passed.
Network diagnostic passed (1/1 checks passed).

You must log in to continue. Would you like to log in (Y/n)? Y

# Se mostrará una lista de proyectos, escogemos el proyecto o creamos uno nuevo.

You are logged in as: [<your-account-email>].

Pick cloud project to use:
[1] sopes2-proyecto-329600
Please enter numeric choice or text value (must exactly match list item): 2
```

Configuración de Kubernetes

Setear gcloud y kubectl

Instalar gcloud, correr el siguiente comando en Windows Powerll y ejecutar instalador

```
# Correr comando para iniciar configuración
> gcloud init

#Se mostraraá un mensaje como el siguiente, darle "Y" para aceptar y loggearnos en GCP

Network diagnostic detects and fixes local network connection issues.
Checking network connection...done.
Reachability Check passed.
Network diagnostic passed (1/1 checks passed).

You must log in to continue. Would you like to log in (Y/n)? Y

# Se mostrará una lista de proyectos, escogemos el proyecto o creamos uno nuevo.

You are logged in as: [<your-account-email>].

Pick cloud project to use:
[1] sopes2-proyecto-329600
Please enter numeric choice or text value (must exactly match list item): 2
```

Configuración de Kubernetes

```
# Configuramos una región y zona predeterminada, ejemplo us-central1-a

> gcloud config set compute/zone us-central1-a

# Verificamos que haya sido configurada correctamente

> gcloud config list compute/zone

# Instalamos kubectl

> gcloud components install kubectl

# Creamos cluster kubernetes

> gcloud container clusters create squidgames --num-nodes=3 --no-enable-ip-alias

# Recuperando credenciales para Kubectl

> gcloud container clusters get-credentials k8s-demo --zone=us-central1-c

# Creamos reglas de firewall para los puertos

> gcloud compute firewall-rules create fwrule-kubernetes --allow tcp:30000-32767
```

Configuración de Kubernetes

Comandos kubectl

```
# Levantar recursos Frontend
```

- > kubectl apply -f frontend-deployment.yaml -f frontend-service.yaml
- # Borrar recursos Frontend
- > kubectl delete -f frontend-deployment.yaml -f frontend-service.yaml
- # Levantar recursos Backend
- > kubectl apply -f backend-deployment.yaml -f backend-service.yaml
- # Borrar recursos Backend
- > kubectl delete -f backend-deployment.yaml -f backend-service.yaml

Seteando nueva imagen en deployment ejecutandose.

- > kubectl set image deployments/<deployment name> <label name>=<new image registry name:tage>
- # Ejemplo
- > kubectl set image deployments/frontend-deployment frontend-deployment=oscarllamas6/frontend-changed:v1

Verificar estado del rolling update

> kubectl rollout status deployment/frontend-deployment

Retroceder con rollout updates

> kubectl rollout undo deployments/frontend-deployment

Prometheus y Grafana (Linkerd)

Comandos para instalar y setear Linkerd

Configuración de Kubernetes

Comandos kubectl

```
# Levantar recursos Frontend
```

- > kubectl apply -f frontend-deployment.yaml -f frontend-service.yaml
- # Borrar recursos Frontend
- > kubectl delete -f frontend-deployment.yaml -f frontend-service.yaml
- # Levantar recursos Backend
- > kubectl apply -f backend-deployment.yaml -f backend-service.yaml
- # Borrar recursos Backend
- > kubectl delete -f backend-deployment.yaml -f backend-service.yaml

Seteando nueva imagen en deployment ejecutandose.

- > kubectl set image deployments/<deployment name> <label name>=<new image registry name:tage>
- # Eiemplo
- > kubectl set image deployments/frontend-deployment frontend-deployment=oscarllamas6/frontend-changed:v1

Verificar estado del rolling update

> kubectl rollout status deployment/frontend-deployment

Retroceder con rollout updates

> kubectl rollout undo deployments/frontend-deployment