Desafío 11

Este desafío busca poner en práctica como desplegar una aplicación y los servicios necesarios en **Kubernetes**, para eso necesitamos nuestro docker compose del **Desafío 10**, el desafío consiste en realizar realizar un Deployment de la aplicación y que esta se conecte con su base de datos.

Como se mencionó anteriormente se utilizara el mismo **Dockerfile** del **Desafío 10**

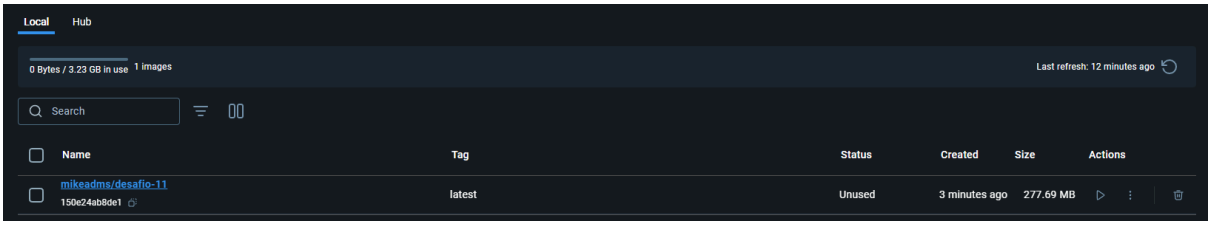
Dockerfile

App > Dockerfile

```
# Etapa 1: Construcción
FROM node:18-alpine AS builder

# Establecer directorio de trabajo
WORKDIR /app

# Copiar los archivos necesarios para instalar dependencias
COPY package*.json ./

# Instalar dependencias
RUN npm install

# Copiar el resto del código fuente
COPY . .

# Compilar TypeScript a JavaScript
RUN npm run build

# Etapa 2: Imagen final
FROM node:18-alpine

# Establecer directorio de trabajo
WORKDIR /app

# Copiar solo los archivos necesarios desde la etapa de construcción
COPY --from=builder /app/dist ./dist
COPY --from=builder /app/node_modules ./node_modules

# Exponer el puerto de la aplicación
EXPOSE 3000

# Comando por defecto para ejecutar la aplicación
CMD ["node", "dist/main.js"]
```

Seguidamente en nuestra consola de Powershell usaremos el comando **docker build --no-cache -t mikeadms/desafio-11:latest .** para subir nuestra imagen.

```
PS C:\Users\Michael\Desktop\Desafio-11\App> docker build --no-cache -t mikeadms/desafio-11:latest .
[+] Building 50.4s (14/14) FINISHED                                                                  docker:desktop-linux
 => [internal] load build definition from Dockerfile                                                                 0.0s
 => => transferring dockerfile: 789B                                                                                 0.0s
 => [internal] load metadata for docker.io/library/node:18-alpine                                                    1.2s
 => [auth] library/node:pull token for registry-1.docker.io                                                          0.0s
 => [internal] load .dockerignore                                                                                    0.0s
 => => transferring context: 2B                                                                                      0.0s
 => [builder 1/6] FROM docker.io/library/node:18-alpine@sha256:02376a266c84acbf45bd19440e08e48b1c8b980374173340046   0.0s
 => [internal] load build context                                                                                    0.0s
 => => transferring context: 12.39kB                                                                                 0.0s
 => CACHED [builder 2/6] WORKDIR /app                                                                                0.0s
 => [builder 3/6] COPY package*.json ./                                                                              0.0s
 => [builder 4/6] RUN npm install                                                                                   40.6s
 => [builder 5/6] COPY . .                                                                                           0.0s
 => [builder 6/6] RUN npm run build                                                                                  3.2s
 => [stage-1 3/4] COPY --from=builder /app/dist ./dist                                                               0.0s
 => [stage-1 4/4] COPY --from=builder /app/node_modules ./node_modules                                               2.8s
 => exporting to image                                                                                               1.4s
 => => exporting layers                                                                                              1.4s
 => => writing image sha256:150e24ab8de1bd0f26fa6c3933f58061d98fcd075c11ba2e899dc13f4a0a7b17                         0.0s
 => => naming to docker.io/mikeadms/desafio-11:latest                                                                0.0s

View build details: docker-desktop://dashboard/build/desktop-linux/desktop-linux/wouojtjhxp6yjl1w4do7u6pbe

What's next:
    View a summary of image vulnerabilities and recommendations → docker scout quickview
```



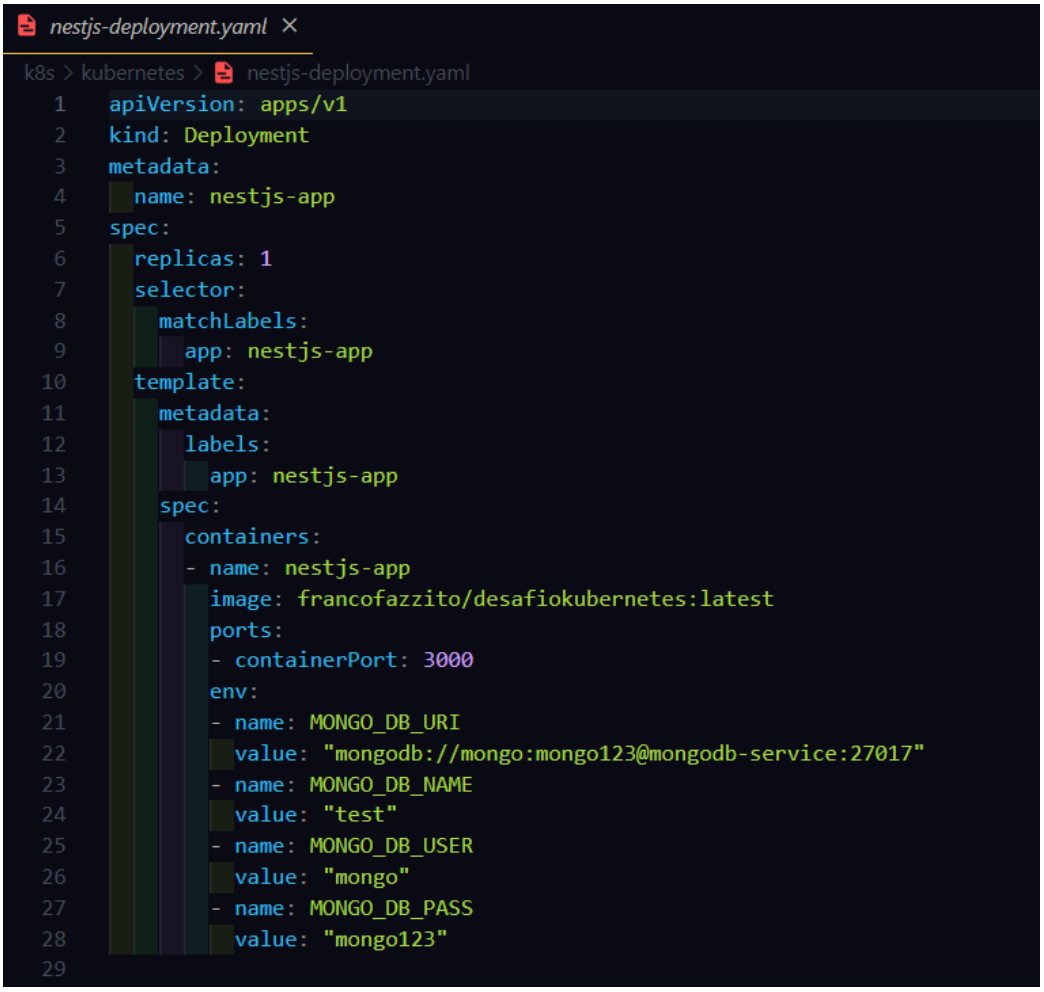
Luego creamos el archivo **nest.js-service** con los manifiestos de **kubernetes** para el deployment de nuestra base de datos y de nuestra app

nestjs-service.yaml

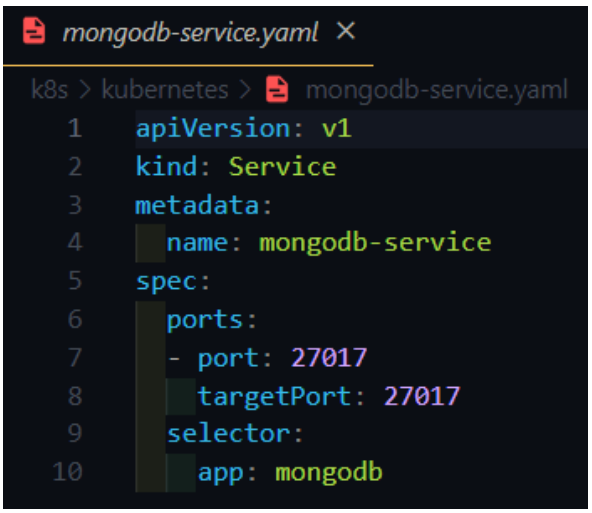
k8s > kubernetes > nestjs-service.yaml

```yaml
apiVersion: v1
kind: Service
metadata:
  name: nestjs-service
spec:
  ports:
  - port: 3000
    targetPort: 3000
  selector:
    app: nestjs-app
  type: LoadBalancer
```

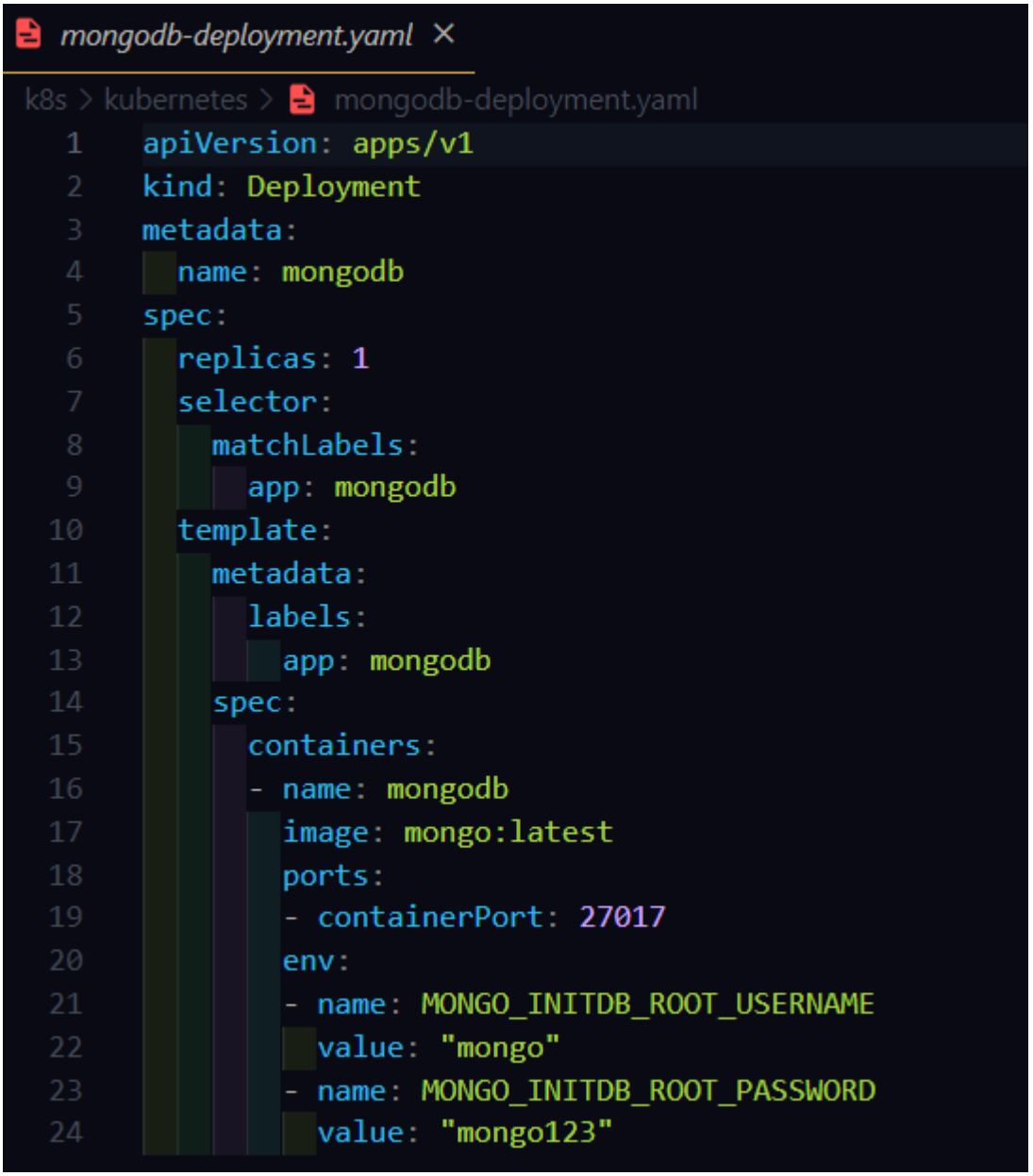
Seguidamente del documento **nest.js-deployment** ya que despliega una aplicación **NestJS** en **Kubernetes**, definiendo los contenedores, réplicas y configuraciones de red necesarias para su ejecución y disponibilidad en el clúster.



El siguiente paso sería nuestro **Mongodb-service** que define un servicio en **Kubernetes** para exponer y acceder a una base de datos **MongoDB**, asegurando la comunicación entre la aplicación y la base de datos dentro del clúster.



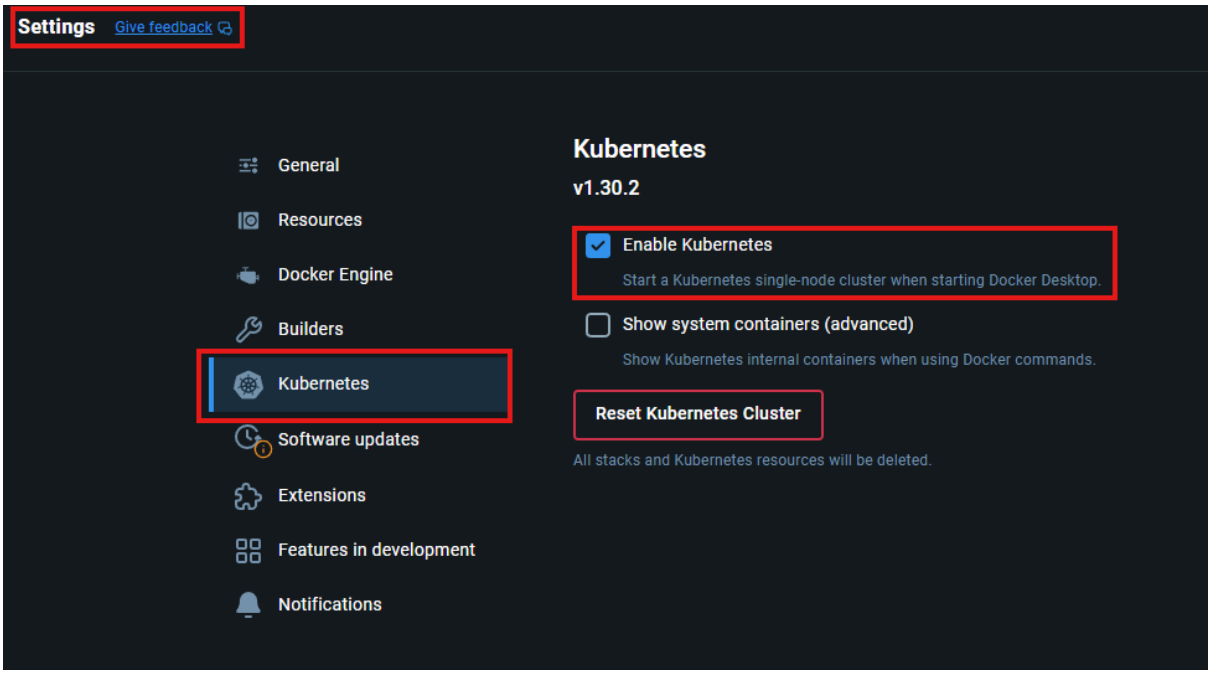
Y por último **Mongodb-deployment** que es el archivo que define la configuración para desplegar instancias (pods) de **MongoDB** en **Kubernetes**, gestionando la replicación y asegurando la alta disponibilidad de la base de datos.



Ya creado todos los manifiestos aplicamos sobre el cluster los siguientes comandos.

```
kubectl apply -f mongodb-deployment.yaml
kubectl apply -f mongodb-service.yaml
kubectl apply -f nestjs-deployment.yaml
kubectl apply -f nestjs-service.yaml
```

Si al ejecutar unos de estos comandos da error verificar en Docker Desktop que esté habilitado en las configuraciones, lo hacemos en el engranaje de configuraciones



Para visualizar nuestra App corriendo colocamos el comando **kubectl get all**

```
PS C:\Users\Michael\Desktop\Desafio-11\k8s\kubernetes> kubectl config current-context
error: current-context is not set
PS C:\Users\Michael\Desktop\Desafio-11\k8s\kubernetes> kubectl apply -f mongodb-deployment.yaml
deployment.apps/mongodb created
PS C:\Users\Michael\Desktop\Desafio-11\k8s\kubernetes> kubectl apply -f mongodb-service.yaml
service/mongodb-service created
PS C:\Users\Michael\Desktop\Desafio-11\k8s\kubernetes> kubectl apply -f nestjs-deployment.yaml
deployment.apps/nestjs-app created
PS C:\Users\Michael\Desktop\Desafio-11\k8s\kubernetes> kubectl apply -f nestjs-service.yaml
service/nestjs-service created
PS C:\Users\Michael\Desktop\Desafio-11\k8s\kubernetes> kubectl get all
NAME                             READY   STATUS              RESTARTS   AGE
pod/mongodb-7978649c45-cwvnl     0/1     ContainerCreating   0          74s
pod/nestjs-app-788f98989d-6w89s  0/1     ContainerCreating   0          14s

NAME                     TYPE           CLUSTER-IP      EXTERNAL-IP   PORT(S)          AGE
service/kubernetes       ClusterIP      10.96.0.1       <none>        443/TCP          29m
service/mongodb-service  ClusterIP      10.103.83.251   <none>        27017/TCP        20s
service/nestjs-service   LoadBalancer   10.103.83.127   localhost     3000:31003/TCP   8s

NAME                         READY   UP-TO-DATE   AVAILABLE   AGE
deployment.apps/mongodb      0/1     1            0           74s
deployment.apps/nestjs-app   0/1     1            0           14s

NAME                                   DESIRED   CURRENT   READY   AGE
replicaset.apps/mongodb-7978649c45     1         1         0       74s
replicaset.apps/nestjs-app-788f98989d  1         1         0       14s
```

Con el comando **kubectl logs -f <ID POD>** podemos ver los registros de nuestro cluster.

```
PS C:\Users\Michael\Desktop\Desafio-11\k8s\kubernetes> kubectl logs -f pod/nestjs-app-788f98989d-6w89s
[Nest] 1  - 09/23/2024, 11:14:25 PM     LOG [NestFactory] Starting Nest application...
[Nest] 1  - 09/23/2024, 11:14:25 PM     LOG [InstanceLoader] MongooseModule dependencies initialized +33ms
[Nest] 1  - 09/23/2024, 11:14:25 PM     LOG [InstanceLoader] ConfigHostModule dependencies initialized +1ms
[Nest] 1  - 09/23/2024, 11:14:25 PM     LOG [InstanceLoader] AppModule dependencies initialized +0ms
[Nest] 1  - 09/23/2024, 11:14:25 PM     LOG [InstanceLoader] ConfigModule dependencies initialized +0ms
[Nest] 1  - 09/23/2024, 11:14:25 PM     LOG [InstanceLoader] MongooseCoreModule dependencies initialized +25ms
[Nest] 1  - 09/23/2024, 11:14:25 PM     LOG [RoutesResolver] AppController {/}: +5ms
[Nest] 1  - 09/23/2024, 11:14:25 PM     LOG [RouterExplorer] Mapped {/, GET} route +3ms
[Nest] 1  - 09/23/2024, 11:14:25 PM     LOG [NestApplication] Nest application successfully started +2ms
```