



Desafío No. 8

Bootcamp DevOps 63703

Presentado por:
Marco Vanegas
2023

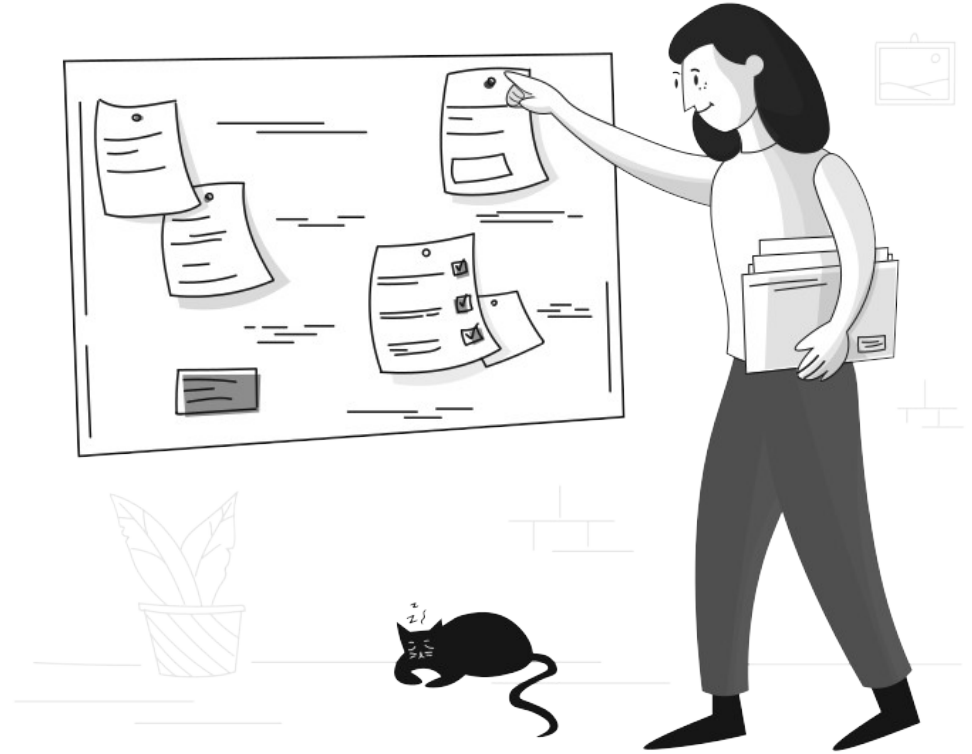
Parte I

Entregables

1. El entregable de esta práctica será un repo con el código del Dockerfile y el link a la imagen de DockerHub. Así mismo, cualquier otro tipo de archivo secundario para la correcta construcción de la imagen será necesario que lo suban.
2. Desarrollar un pipeline de CI/CD en GitHubAction que realice el build de la imagen y lo publique a DockerHub.

Requisitos mínimos

- Algún archivo que sea agregado de forma externa con la opción de utilizar un volumen para almacenarlo.
- Algún servicio que se pueda acceder de forma remota (como puede ser una base dedatos, un servicio web, etc).
- Se tendrá que poder acceder desde la máquina host donde se ejecute ese contenedor.



Illustrations by [Pixeltrue](#) on [icons8](#)



1. Se ejecuta el comando “docker compose up”.

```
dev@dev-ThinkPad-T530: ~/Desktop/Bootcamp DevOps/Desafios/Solucion_Desafios/Solucion_Desafio_8/docker_image$ docker compose up
[+] Running 10/10
✓ my_mongo 9 layers [██████████] 0B/0B Pulled 254.1s
✓ 707e32e9fc56 Pull complete 57.7s
✓ c7ac84d07e95 Pull complete 3.4s
✓ ce678af55db4 Pull complete 16.8s
✓ e6212b74a0e2 Pull complete 12.2s
✓ 08077ff6df71 Pull complete 13.4s
✓ 5c1db0580f35 Pull complete 13.9s
✓ 9d294053e6f8 Pull complete 14.9s
✓ c2aad3066658 Pull complete 239.9s
✓ e596cadf5785 Pull complete 19.9s
[+] Building 58.9s (11/11) FINISHED docker:default
=> [my_app internal] load .dockerignore 0.9s
=> => transferring context: 2B 0.0s
=> [my_app internal] load build definition from Dockerfile 0.6s
=> => transferring dockerfile: 192B 0.1s
=> [my_app internal] load metadata for docker.io/library/node:18 4.1s
=> [my_app auth] library/node:pull token for registry-1.docker.io 0.0s
=> [my_app 1/5] FROM docker.io/library/node:18@sha256:ee0a21d64211d92d4340b225c556e9ef1a8bce1d5b03b49f5f07bf1dbbaa5626 0.0s
=> [my_app internal] load build context 3.1s
=> => transferring context: 101.78kB 2.5s
=> CACHED [my_app 2/5] RUN mkdir -p /home/app/volume 0.0s
=> CACHED [my_app 3/5] WORKDIR /home/app 0.0s
=> [my_app 4/5] COPY . . 18.3s
=> [my_app 5/5] RUN npm install 26.9s
=> [my_app] exporting to image 1.7s
=> => exporting layers 1.5s
=> => writing image sha256:795aea9d2ef6d4babec85d9c2048a348adfb655635ad9a45618be7ef4aa137e3 0.1s
=> => naming to docker.io/library/docker_image-my_app 0.1s
[+] Running 3/3
✓ Network docker_image_default Created 2.3s
✓ Container docker_image-my_mongo-1 Created 3.4s
✓ Container docker_image-my_app-1 Created 0.7s
Attaching to docker_image-my_app-1, docker_image-my_mongo-1
docker_image-my_app-1 | Escuchando...
docker_image-my_mongo-1 | about to fork child process, waiting until server is ready for connections.
```

2. Se validan las imágenes creadas.

```
dev@dev-ThinkPad-T530:~/Desktop/Bootcamp DevOps/Desafios/Solucion_Desafios/Solucion_Desafio_8/docker_image$ docker images
```

REPOSITORY	TAG	IMAGE ID	CREATED	SIZE
docker_image-my_app	latest	795aea9d2ef6	6 minutes ago	1.11GB
mongo	latest	3be86e9501b0	43 hours ago	748MB

```
dev@dev-ThinkPad-T530:~/Desktop/Bootcamp DevOps/Desafios/Solucion_Desafios/Solucion_Desafio_8/docker_image$
```

Se validan los contenedores creados.

```
dev@dev-ThinkPad-T530:~/Desktop/Bootcamp DevOps/Desafios/Solucion_Desafios/Solucion_Desafio_8/docker_image$ docker ps
```

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS	NAMES
814bd232e0cf	docker_image-my_app	"docker-entrypoint.s..."	7 minutes ago	Up 7 minutes	0.0.0.0:3000->3000/tcp, :::3000->3000/tcp	docker_image-my_app-1
d757fcd1059b	mongo	"docker-entrypoint.s..."	7 minutes ago	Up 7 minutes	0.0.0.0:27017->27017/tcp, :::27017->27017/tcp	docker_image-my_mongo-1

```
dev@dev-ThinkPad-T530:~/Desktop/Bootcamp DevOps/Desafios/Solucion_Desafios/Solucion_Desafio_8/docker_image$
```

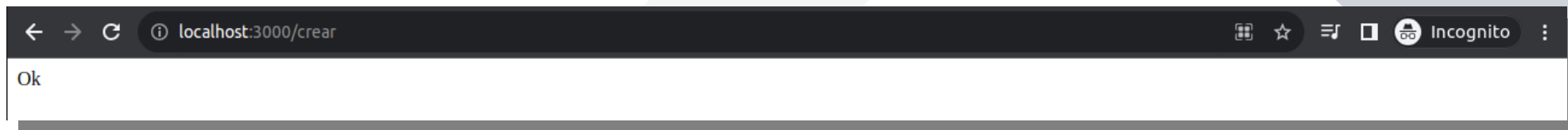
Se valida la red creada.

```
dev@dev-ThinkPad-T530:~/Desktop/Bootcamp DevOps/Desafios/Solucion_Desafios/Solucion_Desafio_8/docker_image$ docker network ls
```

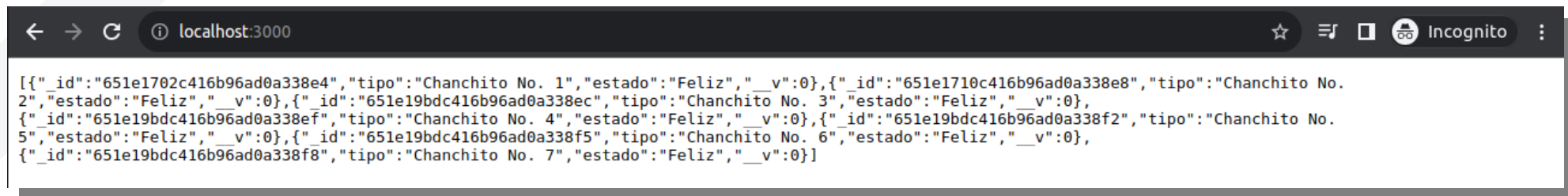
NETWORK ID	NAME	DRIVER	SCOPE
6dfdcb7d6d81	bridge	bridge	local
7bee808628b7	docker_image_default	bridge	local
6e7c75fc8b1e	host	host	local
accde97a86ec	my_network	bridge	local
93027ec969bc	none	null	local

```
dev@dev-ThinkPad-T530:~/Desktop/Bootcamp DevOps/Desafios/Solucion_Desafios/Solucion_Desafio_8/docker_image$
```

3. Ahora se ingresa a la URL <http://localhost:3000/crear> para crear varios registros de forma automática. Para ello, solo es necesario presionar la tecla F5 las veces deseadas.



Luego se valida la creación del registro ingresando a <http://localhost:3000>



4. Ahora se valida el funcionamiento del volumen creado para la app.

Se crea un archivo de prueba en el S.O. o host anfitrión en la ruta \$HOME/Desktop/host_volume.

```
dev@dev-ThinkPad-T530:~/Desktop/host_volume$ touch Prueba.txt
dev@dev-ThinkPad-T530:~/Desktop/host_volume$ ls -l
total 0
-rw-rw-r-- 1 dev dev 0 oct  4 21:18 Prueba.txt
dev@dev-ThinkPad-T530:~/Desktop/host_volume$ echo Hola Mundo > Prueba.txt
dev@dev-ThinkPad-T530:~/Desktop/host_volume$ cat Prueba.txt
Hola Mundo
dev@dev-ThinkPad-T530:~/Desktop/host_volume$
```

Posteriormente se ingresa al contenedor mediante el comando “docker exec -it <nombre_contenedor> bash”, y se verifica la existencia y contenido del archivo que debe hallarse en la ruta \$HOME/Desktop/host_volume.

```
dev@dev-ThinkPad-T530:~/Desktop/Bootcamp DevOps/Desafios/Solucion_Desafios/Solucion_Desafio_8/docker_image$ docker ps
CONTAINER ID   IMAGE          COMMAND                  CREATED        STATUS        PORTS                               NAMES
6b5285d28ccb   docker_image-my_app  "docker-entrypoint.s..." 13 minutes ago Up 13 minutes 0.0.0.0:3000->3000/tcp, :::3000->3000/tcp  docker_image-my_app-1
acba8d1dead5   mongo          "docker-entrypoint.s..." 13 minutes ago Up 13 minutes 0.0.0.0:27017->27017/tcp, :::27017->27017/tcp  docker_image-my_mongo-1
dev@dev-ThinkPad-T530:~/Desktop/Bootcamp DevOps/Desafios/Solucion_Desafios/Solucion_Desafio_8/docker_image$ docker exec -it docker_image-my_app-1 bash
root@6b5285d28ccb:/home/app# cd volume
root@6b5285d28ccb:/home/app/volume# ls -l
total 4
-rw-rw-r-- 1 node node 11 Oct  5 02:18 Prueba.txt
root@6b5285d28ccb:/home/app/volume# cat Prueba.txt
Hola Mundo
root@6b5285d28ccb:/home/app/volume#
```

5. En esta parte, se utilizará **GitHub Actions** para construir y subir la imagen que contiene la app al registro de DockerHub, para lo cual es necesario crear un repositorio.

hub.docker.com/repository/create?namespace=orpimel

Join the DockerCon online event Oct 4-5th, live from Los Angeles. [Watch now.](#)

docker hub Search Docker Hub Explore Repositories Organizations Help Upgrade orpimel

Repositories Create Using 0 of 1 private repositories. [Get more](#)

Create repository

Namespace


Repository Name*


Short description

A short description to identify your repository. If the repository is public, this description is used to index your content on Docker Hub and in search engines, and is visible to users in search results.

Visibility

Using 0 of 1 private repositories. [Get more](#)

☒ Public  Appears in Docker Hub search results

☐ Private  Only visible to you

[Cancel](#) [Create](#)

Pushing images

You can push a new image to this repository using the CLI:

```
docker tag local-image:tagname new-repo:tagname
docker push new-repo:tagname
```

Make sure to replace `tagname` with your desired image repository tag.

6. Se configuran las credenciales de DockerHub en Git.

The screenshot shows the GitHub interface for managing repository secrets and variables. On the left is a sidebar with navigation links: General, Access (Collaborators, Moderation options), Code and automation (Branches, Tags, Rules, Actions, Webhooks, Environments, Codespaces, Pages), Security (Code security and analysis, Deploy keys, Secrets and variables), and Codespaces. The 'Secrets and variables' section is expanded, and 'Actions' is selected. The main area is titled 'Actions secrets and variables' and contains explanatory text about secrets and variables, a 'New repository secret' button, and two panels: 'Environment secrets' (empty) and 'Repository secrets' (containing DOCKER_PASSWORD and DOCKER_USERNAME).

General

Access

- Collaborators
- Moderation options

Code and automation

- Branches
- Tags
- Rules
- Actions
- Webhooks
- Environments
- Codespaces
- Pages

Security

- Code security and analysis
- Deploy keys
- Secrets and variables**
- Actions
- Codespaces

Actions secrets and variables

Secrets and variables allow you to manage reusable configuration data. Secrets are **encrypted** and are used for sensitive data. [Learn more about encrypted secrets](#). Variables are shown as plain text and are used for **non-sensitive** data. [Learn more about variables](#).

Anyone with collaborator access to this repository can use these secrets and variables for actions. They are not passed to workflows that are triggered by a pull request from a fork.

Secrets **Variables** [New repository secret](#)

Environment secrets

[Manage environments](#)

There are no secrets for this repository's environments.

Repository secrets

DOCKER_PASSWORD	Updated now	
DOCKER_USERNAME	Updated 1 minute ago	

7. Se crea el workflow.

BootcampDevOps2023 / .github / workflows / docker-image-desafio-8.yml in main

Cancel changes

Commit changes...

Edit

Preview

Spaces 2 No wrap

```
10  build:
11    runs-on: ubuntu-latest
12
13    steps:
14      - name: Verificar código
15        uses: actions/checkout@v2
16
17      - name: Configurar Docker Buildx
18        uses: docker/setup-buildx-action@v1
19
20      - name: Login to DockerHub
21        uses: docker/login-action@v1
22        with:
23          username: ${ secrets.DOCKER_USERNAME }
24          password: ${ secrets.DOCKER_PASSWORD }
25
26      - name: Build and push
27        uses: docker/build-push-action@v2
28        with:
29          context: ${ github.workspace }/Solucion_Desafio_8/docker_image
30          file: ${ github.workspace }/Solucion_Desafio_8/docker_image/Dockerfile
31          push: true
32          tags: orpimel/desafio_8:v1.0.0
33
```

Use **Control + Shift + m** to toggle the **tab** key moving focus. Alternatively, use **esc** then **tab** to move to the next interactive element on the page.

Use **Control + Space** to trigger autocomplete in most situations.

Marketplace Documentation

Search Marketplace for Actions

Featured Actions

Setup Node.js environment

3.2k

By actions

Setup a Node.js environment by adding problem matchers and optionally downloading and adding it to the PATH

Upload a Build Artifact

2.5k

By actions

Upload a build artifact that can be used by subsequent workflow steps

Setup Go environment

1.2k

By actions

Setup a Go environment and add it to the PATH

Setup .NET Core SDK

819

By actions

8. Se ejecuta el workflow .

The screenshot shows the GitHub Actions interface for the repository 'BambooThink / BootcampDevOps2023'. The 'Actions' tab is selected, displaying a workflow named 'Construir y subir la imagen de la Solución del Desafío 8' (docker-image-desafio-8.yml). A search bar for workflow runs is present. Below the workflow title, there is a 'Help us improve GitHub Actions' section and a '1 workflow run' section. The workflow run is titled 'Create docker-image-desafio-8.yml' and shows a successful status (green checkmark). The commit message is 'Construir y subir la imagen de la Solución del Desafío 8 #1: Commit a334920 pushed by BambooThink' on the 'main' branch. The run completed 1 minute ago and took 58 seconds.

Actions New workflow

All workflows

Construir y subir imagen de Docker

Construir y subir la imagen de la Solu...

Management

Caches

Runners Beta

Construir y subir la imagen de la Solución del Desafío 8 docker-image-desafio-8.yml Filter workflow runs ...

Help us improve GitHub Actions Give feedback ×

Tell us how to make GitHub Actions work better for you with three quick questions.

1 workflow run Event ▾ Status ▾ Branch ▾ Actor ▾

✓ Create docker-image-desafio-8.yml main 1 minute ago ...

Construir y subir la imagen de la Solución del Desafío 8 #1: Commit a334920 pushed by BambooThink 58s

9. Se valida la existencia de la imagen en DockerHub.

orpimel

Repositories

desafio_8

General

Using 0 of 1 private repositories. [Get more](#)

General


Tags


Builds


Collaborators

Webhooks

Settings

 **orpimel / desafio_8**

Description
Imagen correspondiente a la solución del desafío No. 8 del Bootcamp en DevOps. 

 Last pushed: 2 minutes ago



Docker commands

[Public View](#)

```
docker push orpimel/desafio_8:tagname
```

Tags


This repository contains 1 tag(s).

Tag	OS	Type	Pulled	Pushed
 v1.0.0		Image	---	2 minutes ago


[See all](#) [Go to Advanced Image Management](#)

Automated Builds

Manually pushing images to Hub? Connect your account to GitHub or Bitbucket to automatically build and tag new images whenever your code is updated, so you can focus your time on creating.

Available with Pro, Team and Business subscriptions. [Read more about automated builds](#) .

[Upgrade](#)

Repository overview 

Parte II

Entregables

1. Instructivo con los pasos seguidos para la creación del cluster de Kubernetes.
2. Archivos .yaml utilizados para levantar correctamente la aplicación.

Requisitos mínimos

- * Tendrá que ser un deployment si o si (no pod, no replica set).
- * Tendrá que tener algún tipo de volumen o secreto configurado.
- * Tendrá que ser expuesto a fuera del cluster (ClusterIP).
- * Tendrá que tener entre 3 y 5 réplicas idealmente.
- * Tendrá que tener un método de rollback configurado distinto al default.



Illustrations by [Pixeltrue](#) on [icons8](#)



1. Una vez instalado Minikube, se procede con su ejecución. Es importante destacar, que previamente se inicializó Docker. De esta forma, se crearía un clúster por defecto.

```
dev@dev-ThinkPad-T530:~/Desktop/Bootcamp DevOps/Desafios/Solucion_Desafios$ minikube start
🐹 minikube v1.31.2 on Ubuntu 22.04
🌟 Using the qemu2 driver based on existing profile
👉 Starting control plane node minikube in cluster minikube
🔄 Updating the running qemu2 "minikube" VM ...
🔧 Preparing Kubernetes v1.27.4 on Docker 24.0.4 ...
🔗 Configuring bridge CNI (Container Networking Interface) ...
  ▪ Using image gcr.io/k8s-minikube/storage-provisioner:v5
  ▪ Using image docker.io/kubernetesui/dashboard:v2.7.0
  ▪ Using image docker.io/kubernetesui/metrics-scraper:v1.0.8
💡 Verifying Kubernetes components...
💡 Some dashboard features require the metrics-server addon. To enable all features please run:

    minikube addons enable metrics-server

🌟 Enabled addons: default-storageclass, dashboard
🔧 Done! kubectl is now configured to use "minikube" cluster and "default" namespace by default
dev@dev-ThinkPad-T530:~/Desktop/Bootcamp DevOps/Desafios/Solucion_Desafios$ service docker status
● docker.service - Docker Application Container Engine
   Loaded: loaded (/lib/systemd/system/docker.service; enabled; vendor preset: enabled)
   Active: active (running) since Thu 2023-10-05 15:14:29 -05; 1h 20min ago
     TriggeredBy: ● docker.socket
   Docs: https://docs.docker.com
  Main PID: 1359 (dockerd)
    Tasks: 10
   Memory: 64.8M
      CPU: 2.523s
   CGroup: /system.slice/docker.service
           └─1359 /usr/bin/dockerd -H fd:// --containerd=/run/containerd/containerd.sock

oct 05 15:13:50 dev-ThinkPad-T530 dockerd[1359]: time="2023-10-05T15:13:50.129199227-05:00" level=info msg="Starting up"
oct 05 15:13:50 dev-ThinkPad-T530 dockerd[1359]: time="2023-10-05T15:13:50.231028698-05:00" level=info msg="detected 127.0.0.53 nameserver, a>
oct 05 15:13:58 dev-ThinkPad-T530 dockerd[1359]: time="2023-10-05T15:13:58.308367028-05:00" level=info msg="[graphdriver] using prior storage>
oct 05 15:14:08 dev-ThinkPad-T530 dockerd[1359]: time="2023-10-05T15:14:08.140882818-05:00" level=info msg="Loading containers: start."
oct 05 15:14:11 dev-ThinkPad-T530 dockerd[1359]: time="2023-10-05T15:14:11.232398198-05:00" level=info msg="Default bridge (docker0) is assign>
oct 05 15:14:11 dev-ThinkPad-T530 dockerd[1359]: time="2023-10-05T15:14:11.764072495-05:00" level=info msg="Loading containers: done."
oct 05 15:14:15 dev-ThinkPad-T530 dockerd[1359]: time="2023-10-05T15:14:15.372438251-05:00" level=info msg="Docker daemon" commit=1a79695 gra>
```

2. Se aplican los manifiestos y se valida la creación de los recursos.

```
dev@dev-ThinkPad-T530:~/Desktop/Bootcamp DevOps/Desafios/Solucion_Desafios/Solucion_Desafio_8/minikube$ kubectl apply -f .
persistentvolumeclaim/mongo-data created
deployment.apps/myapp created
service/myapp-service created
statefulset.apps/mongo created
service/mongo-service created
dev@dev-ThinkPad-T530:~/Desktop/Bootcamp DevOps/Desafios/Solucion_Desafios/Solucion_Desafio_8/minikube$ kubectl get all
```

NAME	READY	STATUS	RESTARTS	AGE
pod/mongo-0	0/1	ContainerCreating	0	5s
pod/myapp-65c5dc6987-4htnv	0/1	ContainerCreating	0	6s
pod/myapp-65c5dc6987-4rh5w	0/1	ContainerCreating	0	6s
pod/myapp-65c5dc6987-7r7vx	0/1	ContainerCreating	0	6s
pod/myapp-65c5dc6987-gf8pg	0/1	ContainerCreating	0	7s
pod/myapp-65c5dc6987-p8sf6	0/1	ContainerCreating	0	6s

NAME	TYPE	CLUSTER-IP	EXTERNAL-IP	PORT(S)	AGE
service/kubernetes	ClusterIP	10.96.0.1	<none>	443/TCP	64m
service/mongo-service	ClusterIP	10.97.204.236	<none>	27017/TCP	6s
service/myapp-service	LoadBalancer	10.99.208.251	<pending>	3000:30739/TCP	8s

NAME	READY	UP-TO-DATE	AVAILABLE	AGE
deployment.apps/myapp	0/5	5	0	8s

NAME	DESIRED	CURRENT	READY	AGE
replicaset.apps/myapp-65c5dc6987	5	5	0	8s

NAME	READY	AGE
statefulset.apps/mongo	0/1	7s

```
dev@dev-ThinkPad-T530:~/Desktop/Bootcamp DevOps/Desafios/Solucion_Desafios/Solucion_Desafio_8/minikube$ kubectl get pvc
```

NAME	STATUS	VOLUME	CAPACITY	ACCESS MODES	STORAGECLASS	AGE
mongo-data	Bound	pvc-20927e07-46e0-487d-9d30-9917142e9533	100Mi	RWO	standard	15s

```
dev@dev-ThinkPad-T530:~/Desktop/Bootcamp DevOps/Desafios/Solucion_Desafios/Solucion_Desafio_8/minikube$
```

3. Una vez se creen los recursos y se estabilicen, se ejecuta `minikube tunnel` para poder acceder a la aplicación desde el navegador.

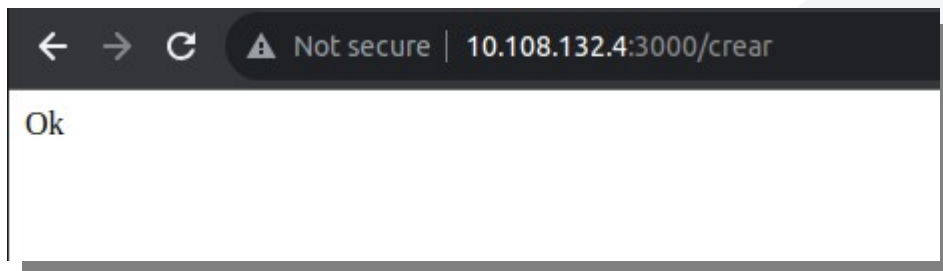
```
minikube tunnel dev@dev-ThinkPad-T530:~/Desktop/Bootcamp DevOps/Desafios/Solucion_Desafios/Solucion_Desafio_8/minikube$ minikube tunnel
! Executing "docker container inspect minikube --format={{.State.Status}}" took an unusually long time: 2.232673282s
💡 Restarting the docker service may improve performance.
[sudo] password for dev:
Status:
  machine: minikube
  pid: 114817
  route: 10.96.0.0/12 -> 192.168.49.2
  minikube: Running
  services: [myapp-service]
errors:
  minikube: no errors
  router: no errors
  loadbalancer emulator: no errors
```

```
dev@dev-ThinkPad-T530:~/Desktop/Bootcamp DevOps/Desafios/Solucion_Desafios/Solucion_Desafio_8/minikube$ kubectl get svc
NAME            TYPE          CLUSTER-IP      EXTERNAL-IP      PORT(S)          AGE
kubernetes      ClusterIP     10.96.0.1       <none>           443/TCP          36m
mongo-service    ClusterIP     10.101.49.219   <none>           27017/TCP        34m
myapp-service    LoadBalancer 10.108.132.4    10.108.132.4    3000:30480/TCP   34m
```

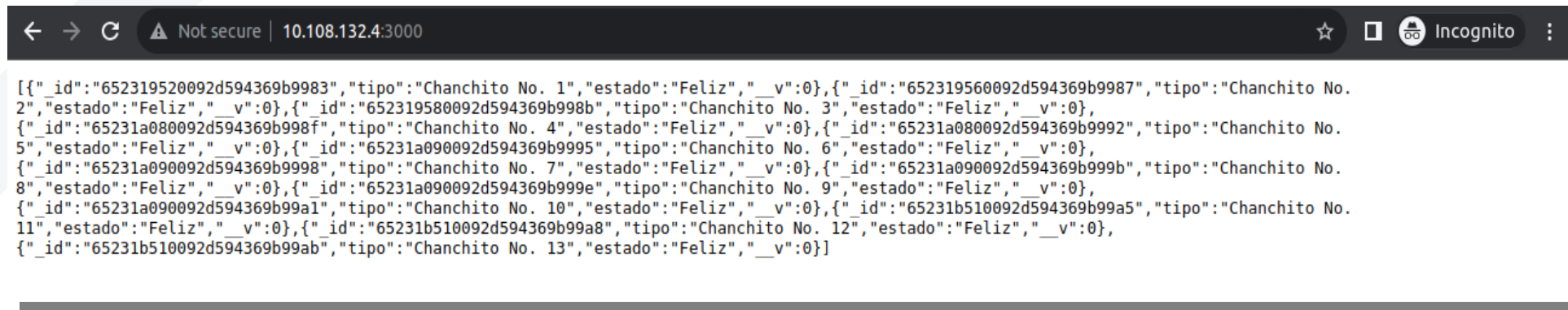
NAME	TYPE	CLUSTER-IP	EXTERNAL-IP	PORT(S)	AGE
kubernetes	ClusterIP	10.96.0.1	<none>	443/TCP	36m
mongo-service	ClusterIP	10.101.49.219	<none>	27017/TCP	34m
myapp-service	LoadBalancer	10.108.132.4	10.108.132.4	3000:30480/TCP	34m

```
dev@dev-ThinkPad-T530:~/Desktop/Bootcamp DevOps/Desafios/Solucion_Desafios/Solucion_Desafio_8/minikube$
```


4. Se valida el funcionamiento de los pods. Para ello, mediante un navegador del host anfitrión se ingresa a las rutas <http://localhost:30000> para listar los registros y <http://localhost:30000/crear> para generar nuevos registros en la BD.



Se presiona la tecla **F5** varias veces para crear múltiples registros.



Listado de los registros creados en la BD.

Repositorio de GitHub donde se encuentra el código, junto con los pasos en detalle:

https://github.com/BambooThink/BootcampDevOps2023/tree/main/Solucion_Desafio_8