Abraxas DevOps Exercise
General process

Victor Hernández

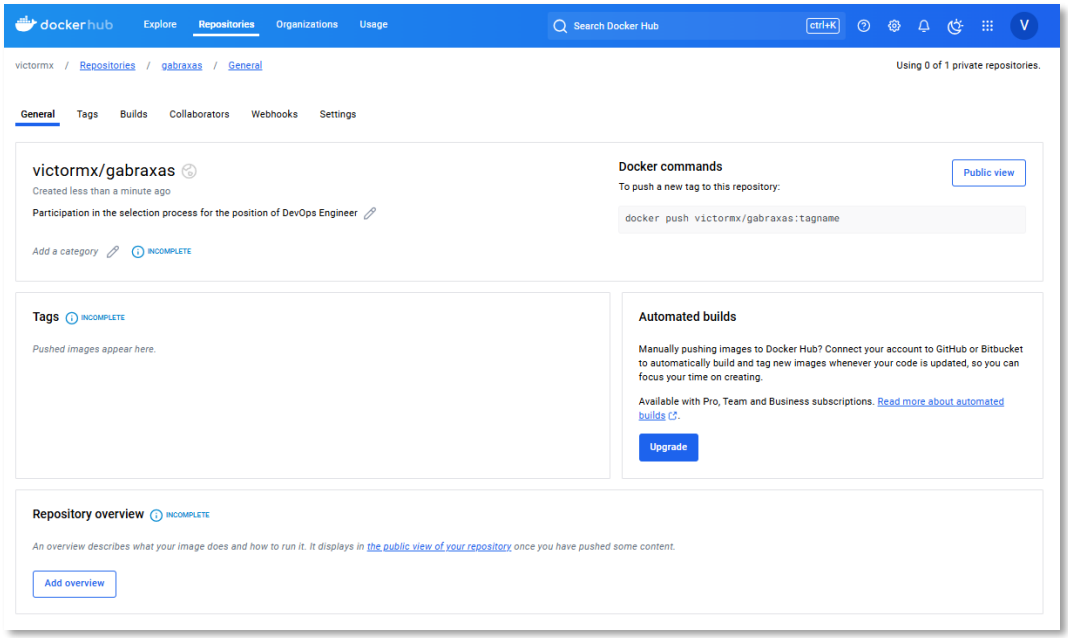
November, 2024

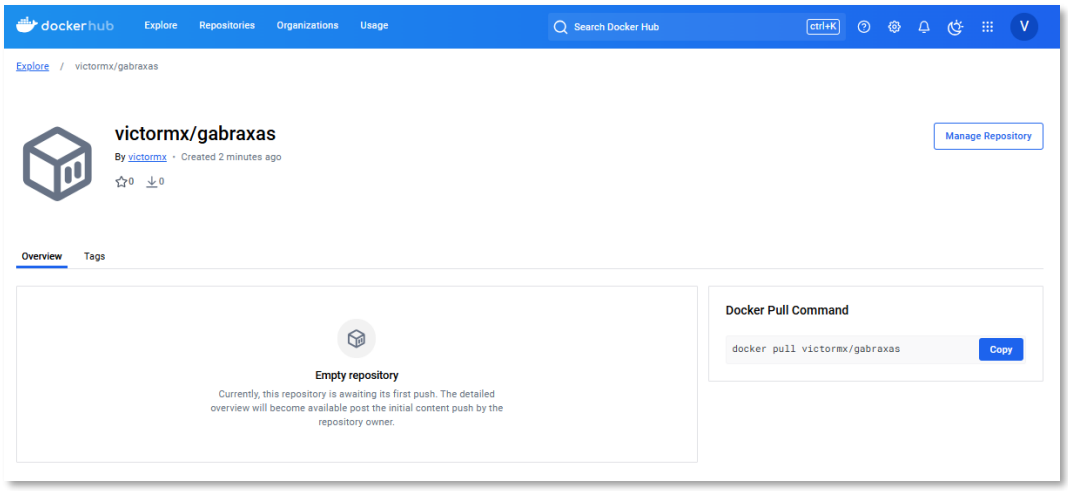
Table of Contents

Prepare image repository

1. Login to DockerHub.
   https://hub.docker.com/

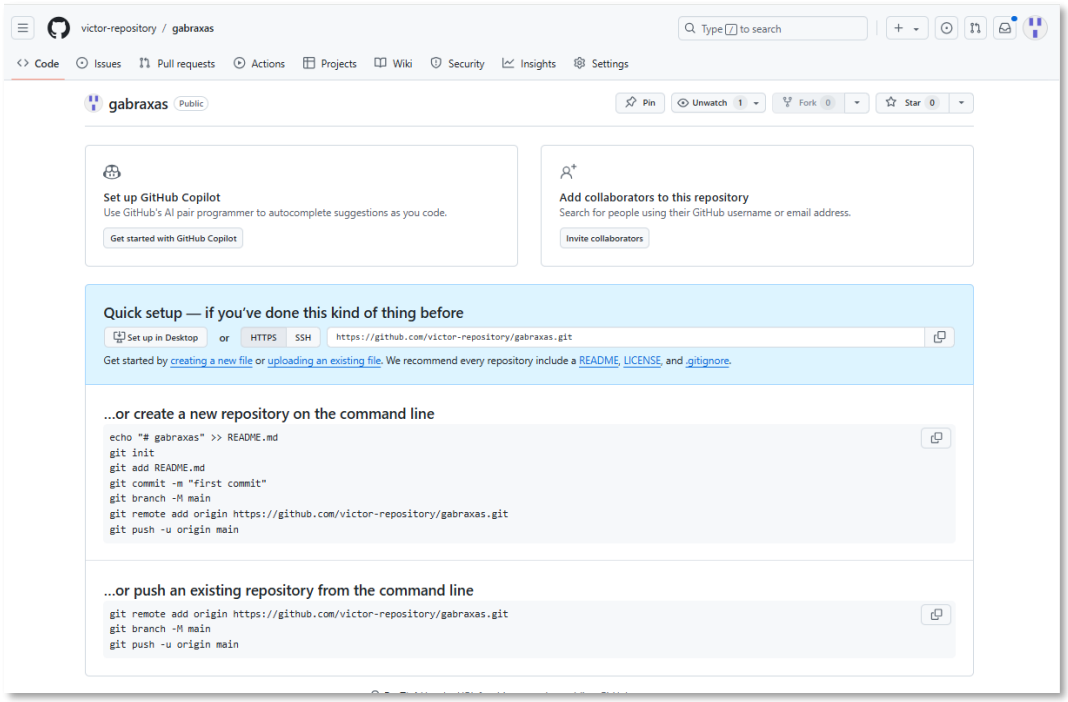
2. Create an image repository.
   https://hub.docker.com/r/victormx/gabraxas



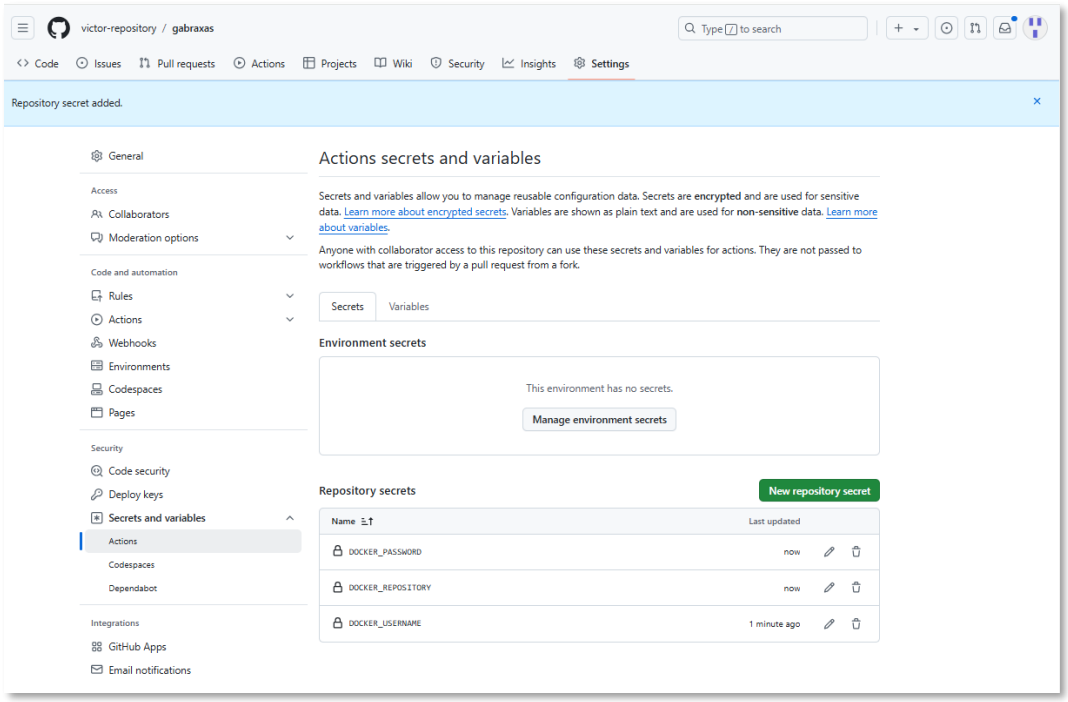


Git code repository

1. Create an empty public repository.
   https://github.com/victor-repository/gabraxas



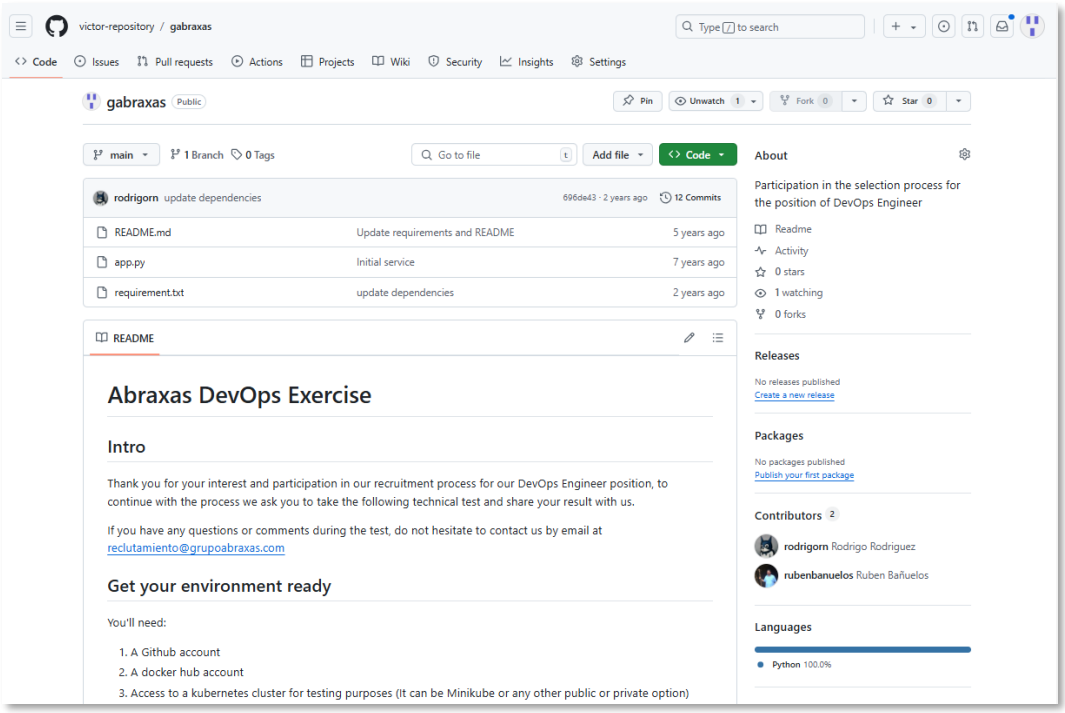
2. Create the following Secrets-Actions with the values from the image repository to establish connection with DockerHub.
   DOCKER_USERNAME
   DOCKER_PASSWORD
   DOCKER_REPOSITORY



Preparations on local computer equipment

1. Create a local folder called **gabraxas** and enter the folder.
2. Clone base repository from https://github.com/Grupo-Abraxas/devops-exercise
3. Enter the folder called **devops-exercise**.
4. Rename Master branch to Main.
5. Set up the new remote repository with following commands:

```
git remote set-url origin https://github.com/victor-repository/gabraxas.git
git branch -M main
git push -u origin main
git remote -v
```



Coding

Create the following folder structure:

1. A folder named app to relocate the **app.py** and **requirement.txt** files containing the application code, associating the code repository on GitHub.
   https://github.com/victor-repository/gabraxas

2. A folder called **.github/workflows** to place the **build.yaml** file with the configuration for continuous integration and delivery to the DockerHub image repository.
   https://hub.docker.com/r/victormx/gabraxas

3. A folder called **deployment** to place the **load-balancer.yaml** file with the deployment configuration for kubernetes, in this case it was validated through minikube.

4. A folder called docs to locate information related to the exercise.

Execute the build

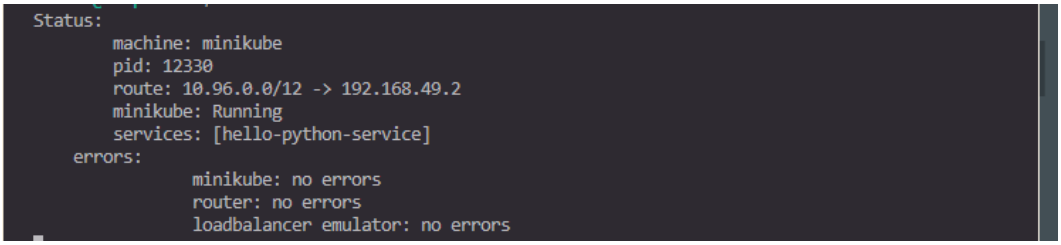
1. Run the following commands in a terminal:

```
git status
git add .
git commit -m "Comentario sobre el cambio"
git push origin main
```

Execute the deployment

1. The first time, a tunnel is established in minikube, by running the following command, through a separate terminal:

```
minikube tunnel  -c
```



2. Deploy the configuration to Kubernetes (minikube) with the following command:

```
kubectl apply -f ./deployment/load-balancer.yaml
```

Check service

1. Service validation can be performed by previously obtaining the external IP of the deployed service with the name hello-python-service and with the help of the curl command.

```
kubectl get service hello-python-service
curl <<External IP>>:80
```