****

Infraestructura II

# **Ejercitación GitLab + Docker Hub Parte 1**

**OBJETIVO**

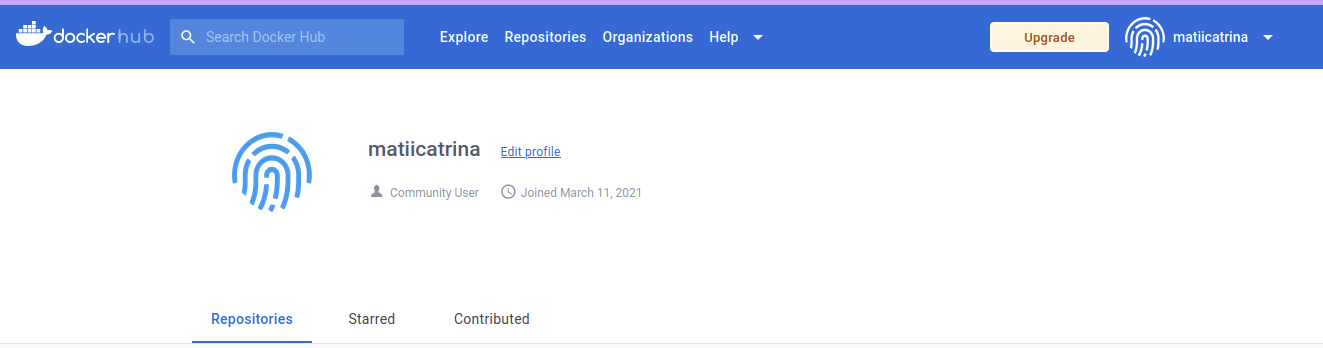
Generar un pipeline en GitLab que me permita pushear el repositorio y generar una imagen de docker hacia el repositorio de Docker Hub.

**RESOLUCIÓN**

La tarea principal es crear una cuenta en Docker Hub sino la tenemos.

Link : <https://hub.docker.com/>

Deberíamos poder llegar a nuestro perfil:



—---------------------------------------------------------------------------

Vamos a lanzar nuestra infraestructura (desde 0 - sin utilizar lo que ya habíamos hecho -) a través de un Docker Compose.

\*\*Necesitamos instalar docker-compose sino lo tenemos con el comando:

sudo apt update

***sudo apt install docker-compose***

**Levantamos GitLab**

Creamos un directorio (en la raíz de nuestro usuario) en donde vamos a persistir todos los datos de nuestro servidor

***mkdir gitlab && cd gitlab***

Ahora creamos un archivo (recomiendo que lo hagan en visual studio code) llamado docker-compose.yml y dentro de él colocamos lo siguiente:

**version: '3.7'**

**services:**

**gitlab:**

**image: 'gitlab/gitlab-ce:latest'**

**restart: always**

**hostname: 'localhost'**

**container\_name: gitlab**

**environment:**

**GITLAB\_OMNIBUS\_CONFIG: |**

**external\_url 'http://localhost'**

**ports:**

**- '8080:80'**

**- '8443:443'**

**volumes:**

**- '$GITLAB\_HOME/config:/etc/gitlab'**

**- '$GITLAB\_HOME/logs:/var/log/gitlab'**

**- '$GITLAB\_HOME/data:/var/opt/gitlab'**

**networks:**

**- gitlab**

**gitlab-runner:**

**image: gitlab/gitlab-runner:alpine**

**container\_name: gitlab-runner**

**restart: always**

**depends\_on:**

**- gitlab**

**volumes:**

**- /var/run/docker.sock:/var/run/docker.sock**

**- '$GITLAB\_HOME/gitlab-runner:/etc/gitlab-runner'**

**networks:**

**- gitlab**

**networks:**

**gitlab:**

**name: gitlab-network**

Nos tiene que quedar similar a la siguiente imagen:



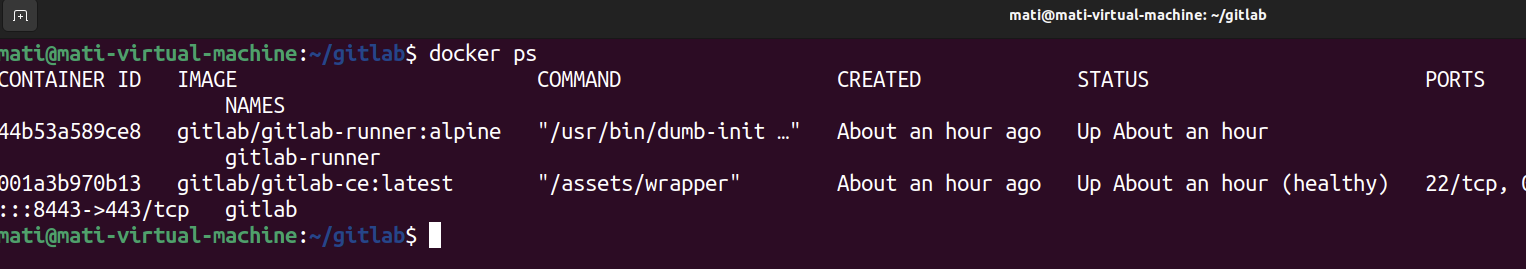
Luego exportamos la variable que va a utilizar el docker-compose

***export GITLAB\_HOME=$HOME/gitlab***

Teniendo ya nuestro docker-compose.yml configurado lo vamos a ejecutar para poder desplegar la configuración GITLAB + RUNNER (2 contenedores). Ejecutamos el siguiente comando:

***sudo -E docker-compose up -d***

Buscamos una soda y esperamos 10 minutitos que todo se ejecute correctamente, deberíamos llegar a la siguiente situación:

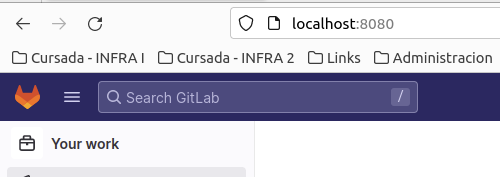


Luego vamos a poder ejecutar el siguiente comando para obtener nuestra clave de root de GitLab:

docker exec -it gitlab grep 'Password:' /etc/gitlab/initial\_root\_password



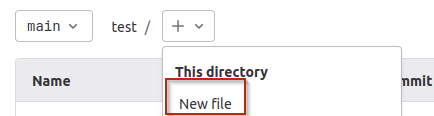
Ya con estas credenciales podemos acceder a nuestro servidor Gitlab a través del localhost:8080



Procedemos a crear un nuevo proyecto y dentro de él vamos a subir una plantilla web, puede ser la misma que estuvieron usando anteriormente o pueden buscar alguna diferente.

En mi caso subí una que encontré en la siguiente web: <https://plantillashtmlgratis.com/en/home/>

Luego procedemos a crear un Dockerfile que va a ser quien construirá nuestra imagen con el proyecto web ya integrado de la siguiente manera:

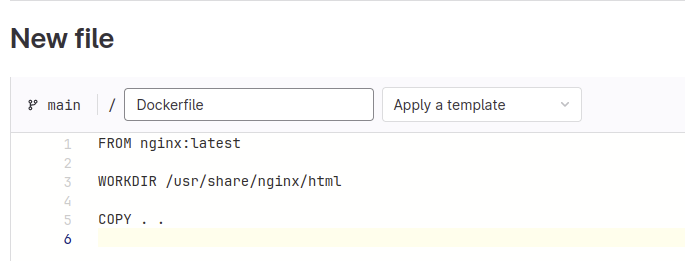


Creamos un archivo Llamado Dockerfile y lo completamos con el siguiente código;

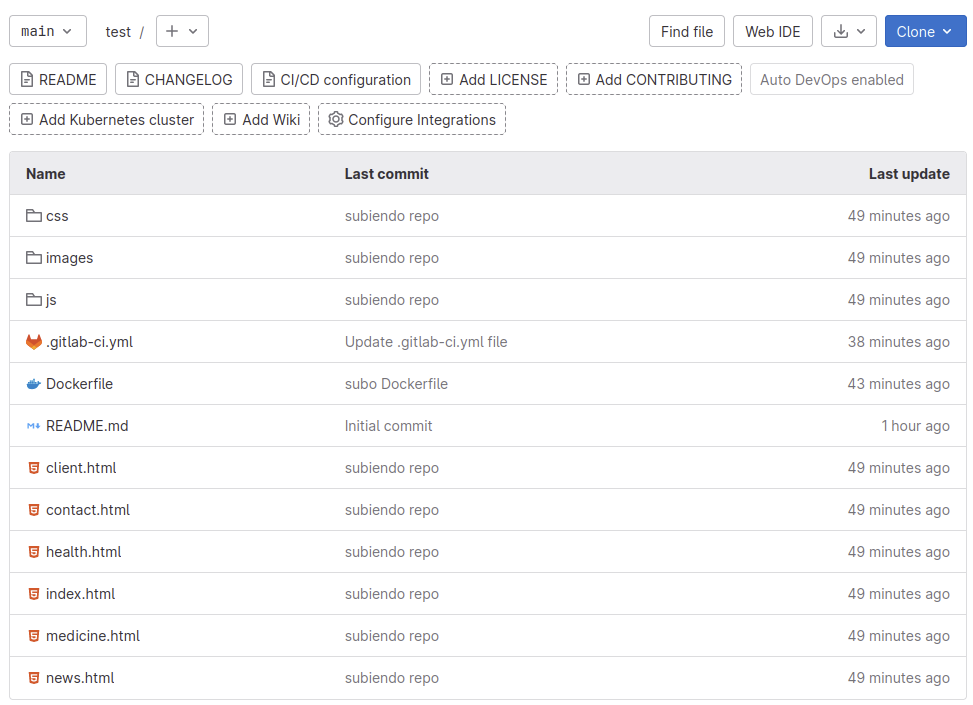
**FROM nginx:latest**

**WORKDIR /usr/share/nginx/html**

**COPY . .**



Guardamos nuestro archivo y nos quedaría de la siguiente manera nuestro Proyecto + el Dockerfile:

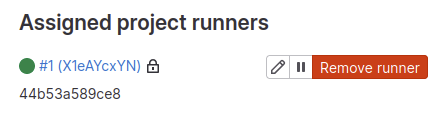


Ya estamos prácticamente listos para que el pipeline corra automáticamente, para ello vamos a tener que configurar nuestro Runner con el token correspondiente al proyecto. Ejecutemos el siguiente comando (modificar solo lo que está en rojo):

***sudo docker exec -it gitlab-runner gitlab-runner register --url "http://gitlab" --clone-url "http://gitlab" --registration-token <TOKEN de nuestro Runner>***

Pulsamos Enter en todas las opciones, menos cuando nos pida **Enter an executor**, allí tipeamos **docker** y pulsamos Enter nuevamente, la siguiente opción que nos preguntará será **Enter the default Docker image**, alli ingresamos **ruby:2.7** y pulsamos Enter.

Ahora si volvemos a ingresar nuevamente a **Settings → CI/CD → Runner**, nos debiera aparecer el runner activado, como se muestra en la siguiente imagen:



Una vez nuestro runner en ejecución falta configurar algunos parámetros dentro del mismo. Para ello vamos a ingresar al contenedor a través del siguiente comando:

***sudo docker exec -it gitlab-runner bash***

Después modificamos el archivo **config.toml** desde la siguiente ruta y con el comando:

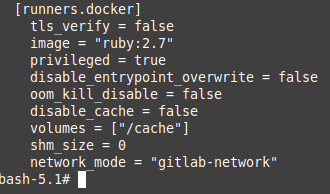
***vi /etc/gitlab-runner/config.toml***

Los siguientes parámetros:

privileged = **true**

network\_mode = **"gitlab-network"**

Deberíamos tener una configuración similar a la siguiente:



**¡¡¡YA CASI ESTAMOS!!!**

Luego dentro de gitlab en la solapa de BUILD vamos a cargar nuestro pipeline:

**stages:**

**- compose**

**stage\_compose:**

**stage: compose**

**image: docker:stable**

**when: manual**

**services:**

**- docker:dind**

**- name: docker:dind**

**alias: thedockerhost**

**variables:**

**DOCKER\_HOST: tcp://thedockerhost:2375/**

**DOCKER\_DRIVER: overlay2**

**DOCKER\_TLS\_CERTDIR: ""**

**script:**

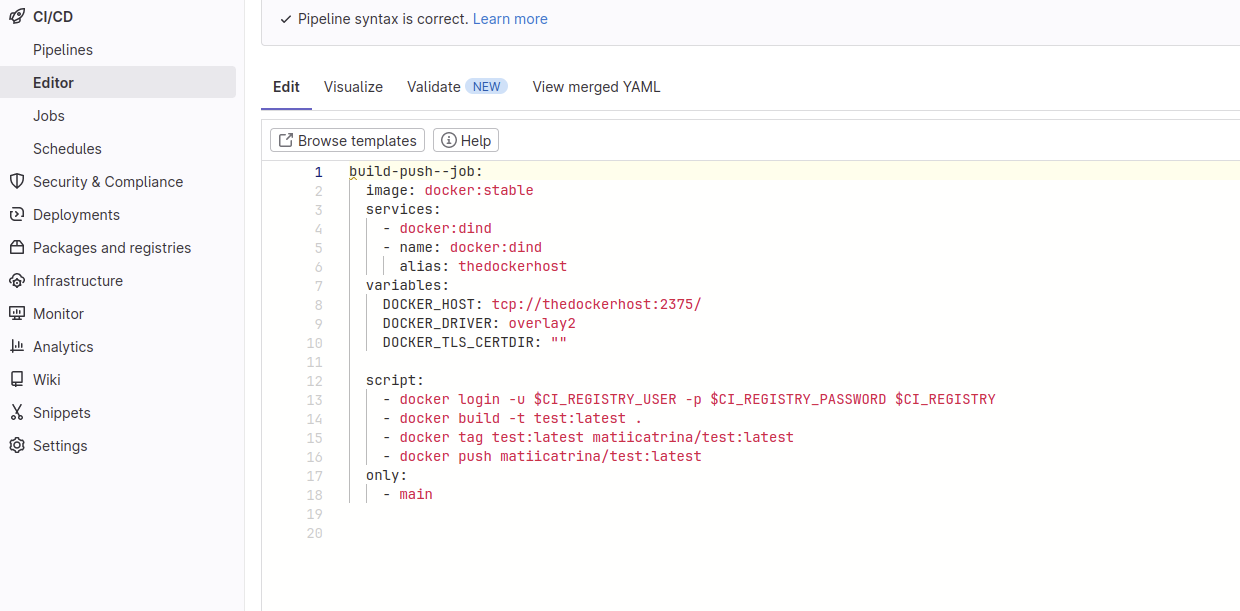
**- docker login -u $CI\_REGISTRY\_USER -p $CI\_REGISTRY\_PASSWORD $CI\_REGISTRY**

**- docker build -t test:latest .**

**- docker tag test:latest $CI\_REGISTRY\_USER/test:latest**

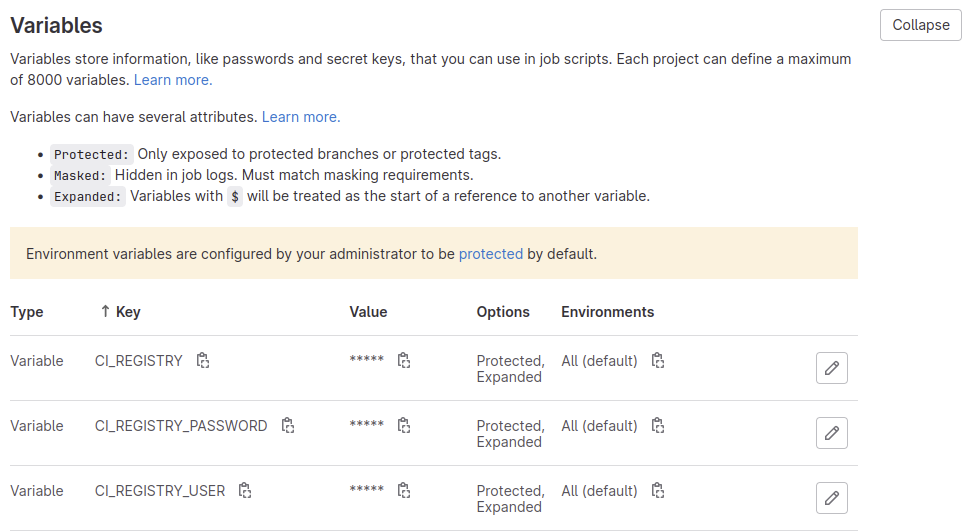
**- docker push $CI\_REGISTRY\_USER/test:latest**

Nos quedaría de la siguiente manera:

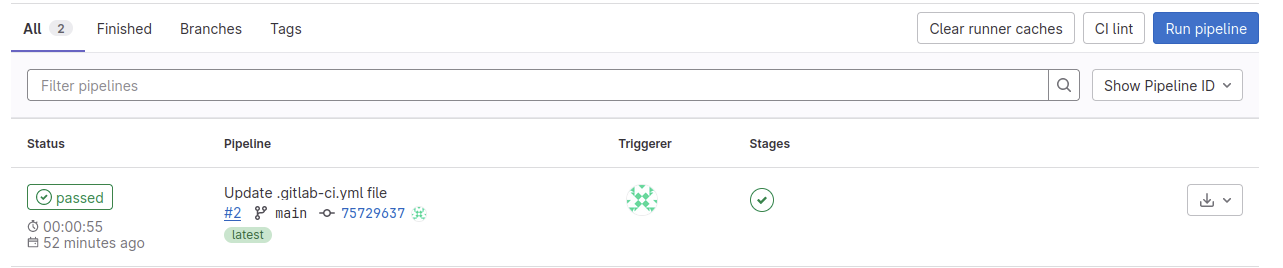


Si prestamos atención, hay 3 variables que vamos a tener que cargar en nuestra sección de variables para que GitLab las utilice. Para ello dentro de nuestro proyecto vamos a Settings > CI/CD > Variables , Una vez ahí cargamos nuestras credenciales de la siguiente manera:

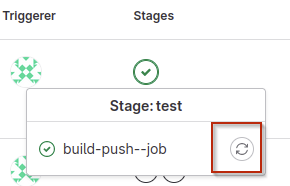
* CI\_REGISTRY = docker.io
* CI\_REGISTRY\_PASSWORD = tupassoword (de dockerhub)
* CI\_REGISTRY\_USER = tuusuario (de dockerhub)



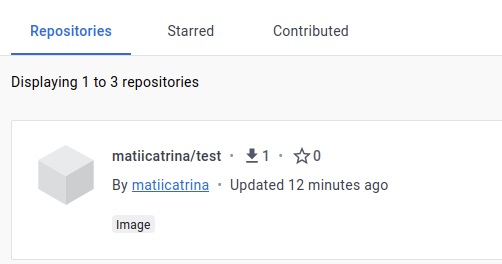
Nuestro pipeline debería pasar a **passed:**

****

Sino lo volvemos a lanzar desde el siguiente botón;



Ahora deberíamos tener nuestra imagen en Docker Hub



**FELICITACIONES**