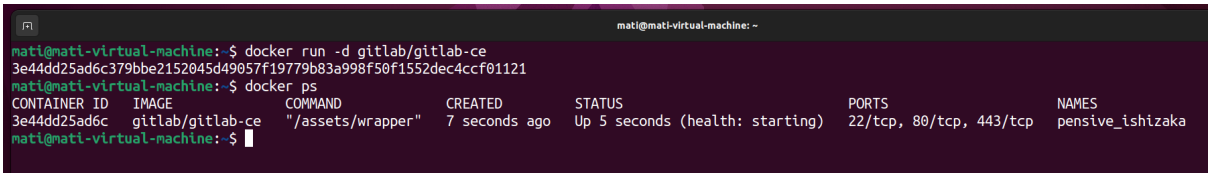
Server Apache & GitLab - CI/CD

1 - Ejecutamos el siguiente comando para correr nuestro contenedor GitLab

docker run -d gitlab/gitlab-ce

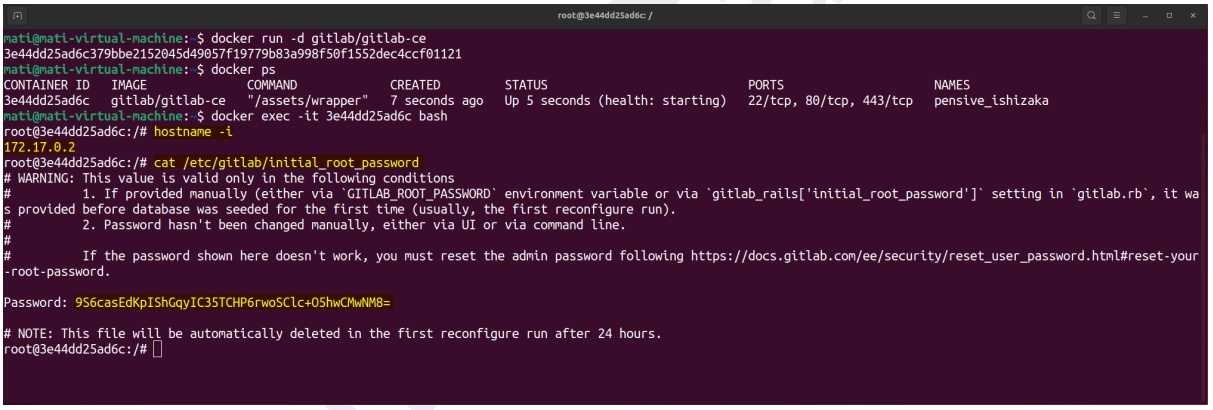


Verificamos que se está ejecutando con el comando:
docker ps

2 - Una vez en ejecución utilizamos el siguiente comando para poder ingresar al contenedor y realizar algunas verificaciones:

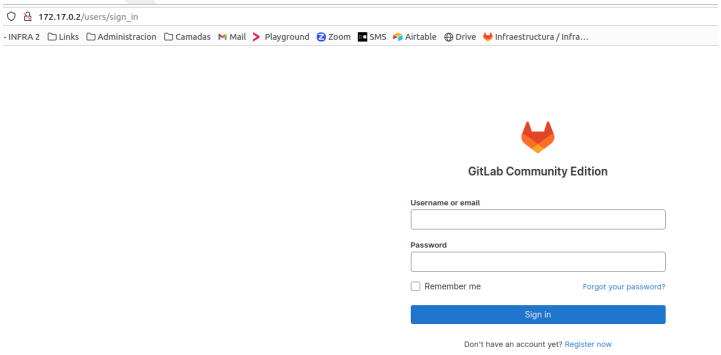
docker exec -it <contenedor ID> bash

Para obtener la IP de Gitlab *: hostname -i*
Para obtener la clave root de Gitlab: *cat /etc/gitlab/initial_root_password*



Podemos verificar que ingresando a la ip que nos arrojó el comando anterior, podemos ingresar a nuestro servidor. Rellenamos los campos:

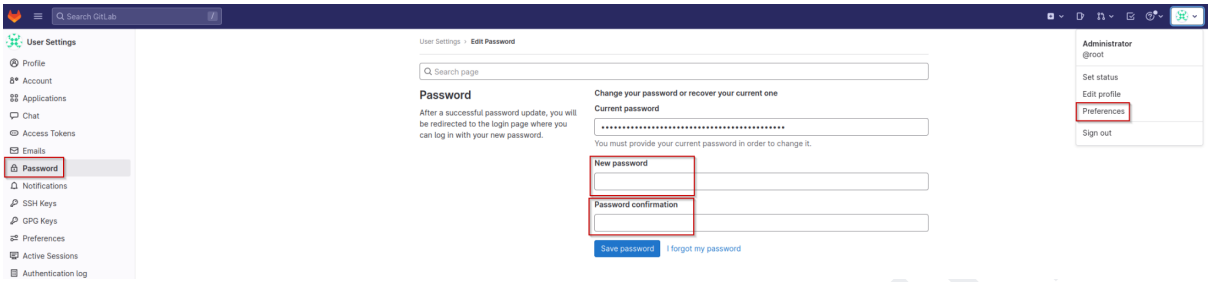
Username: *root*
Password: *el que obtuvimos anteriormente con el comando*



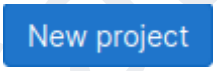
Recomiendo que cambien dicho password por uno más simple en caso de que lo vayamos usar a este ejercicio de prueba (para una mayor comodidad) desde la siguiente ruta:

Perfil>Preferencias>Password

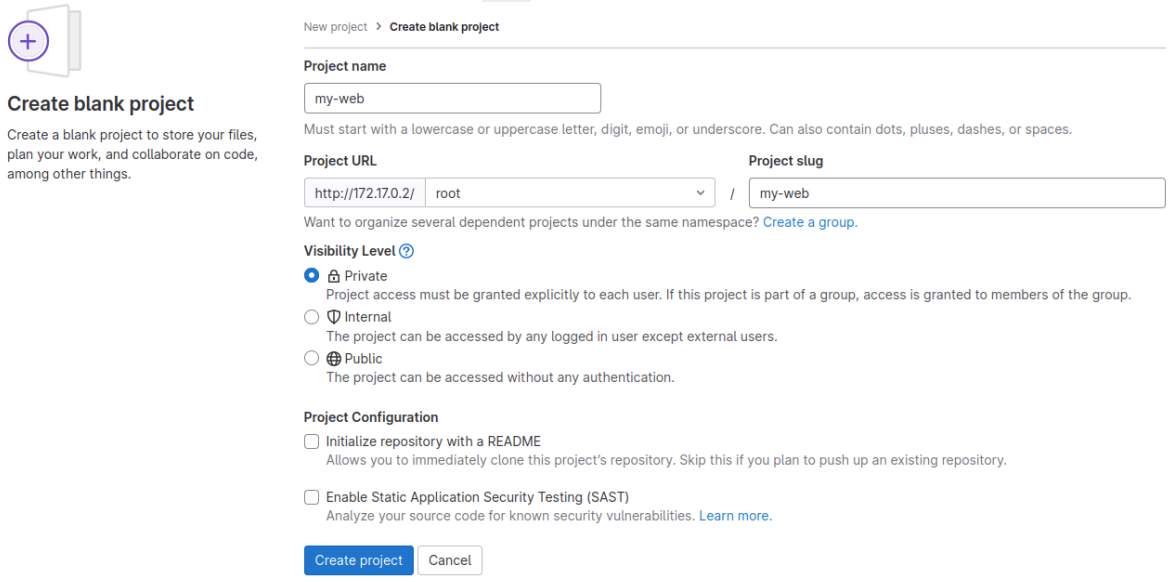
Adjunto captura del lugar exacto



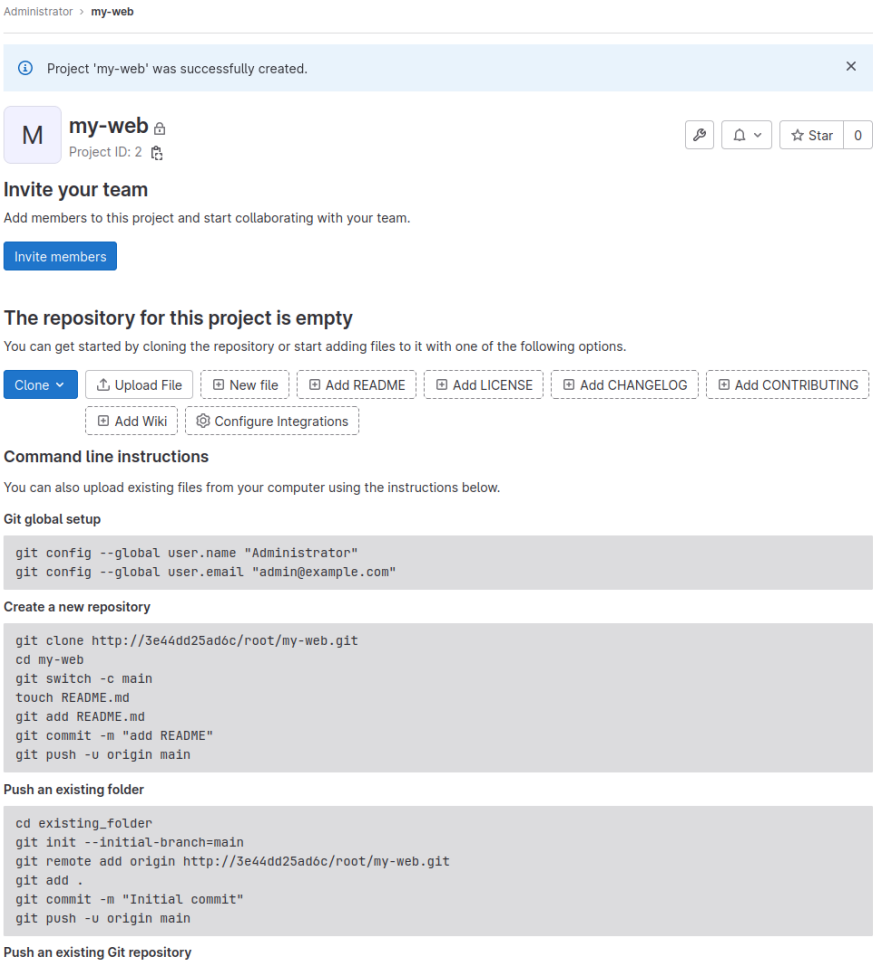
Una vez modificada la clave, nos va a desloguear y volvemos a colocar las credenciales nuevas.

3 - Creamos un nuevo proyecto desde el botón 

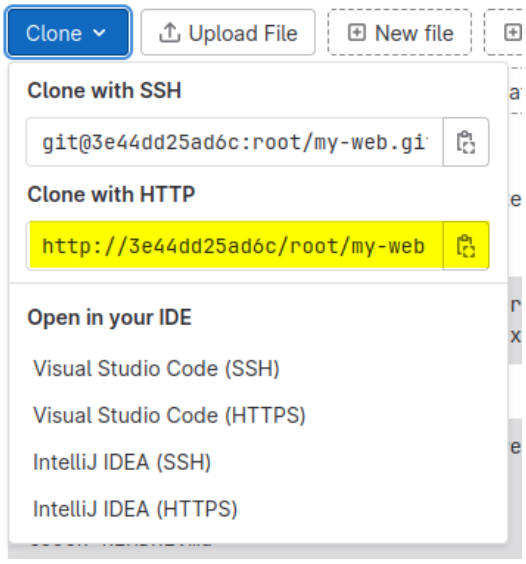
y seteamos los parámetros según nuestros requisitos. Pongo ejemplo los que utilicé yo:



Nos va a llevar a una página muy similar a la siguiente, en donde vamos a tener nuestro proyecto (vacío de momento).



Guardamos el link para poder clonar nuestro repositorio más adelante.



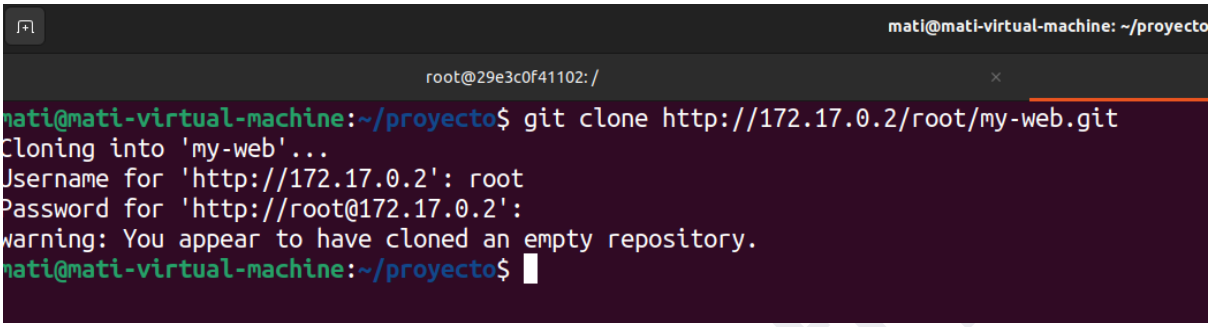
En mi caso es : *http://3e44dd25ad6c/root/my-web.git*

Clonamos nuestro repo localmente y hacemos un push a Gitlab

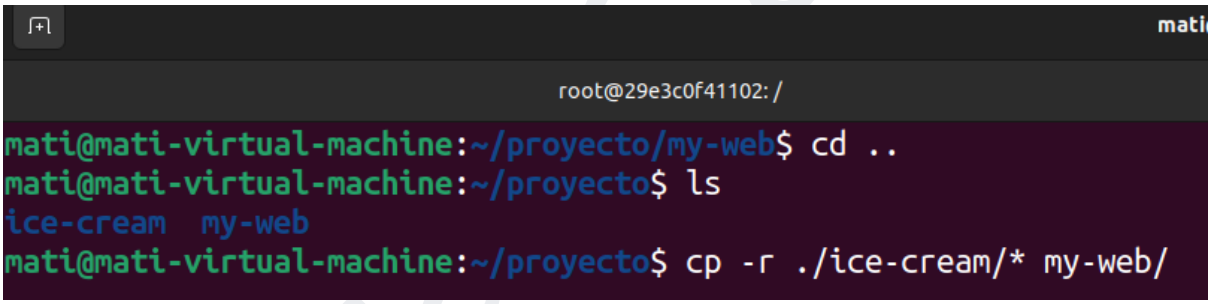
Usamos el siguiente comando para clonar:

git clone http://172.17.0.2/root/my-web.git (revisar que modificamos el clone y le colocamos la ip de nuestro servidor de Gitlab)

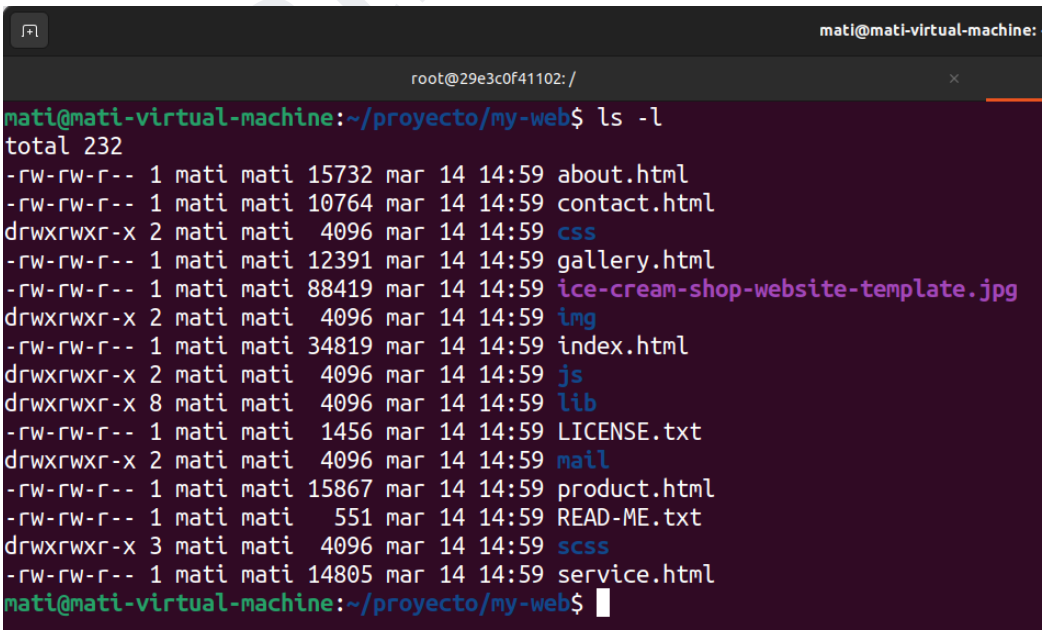
Colocamos nuestras credenciales



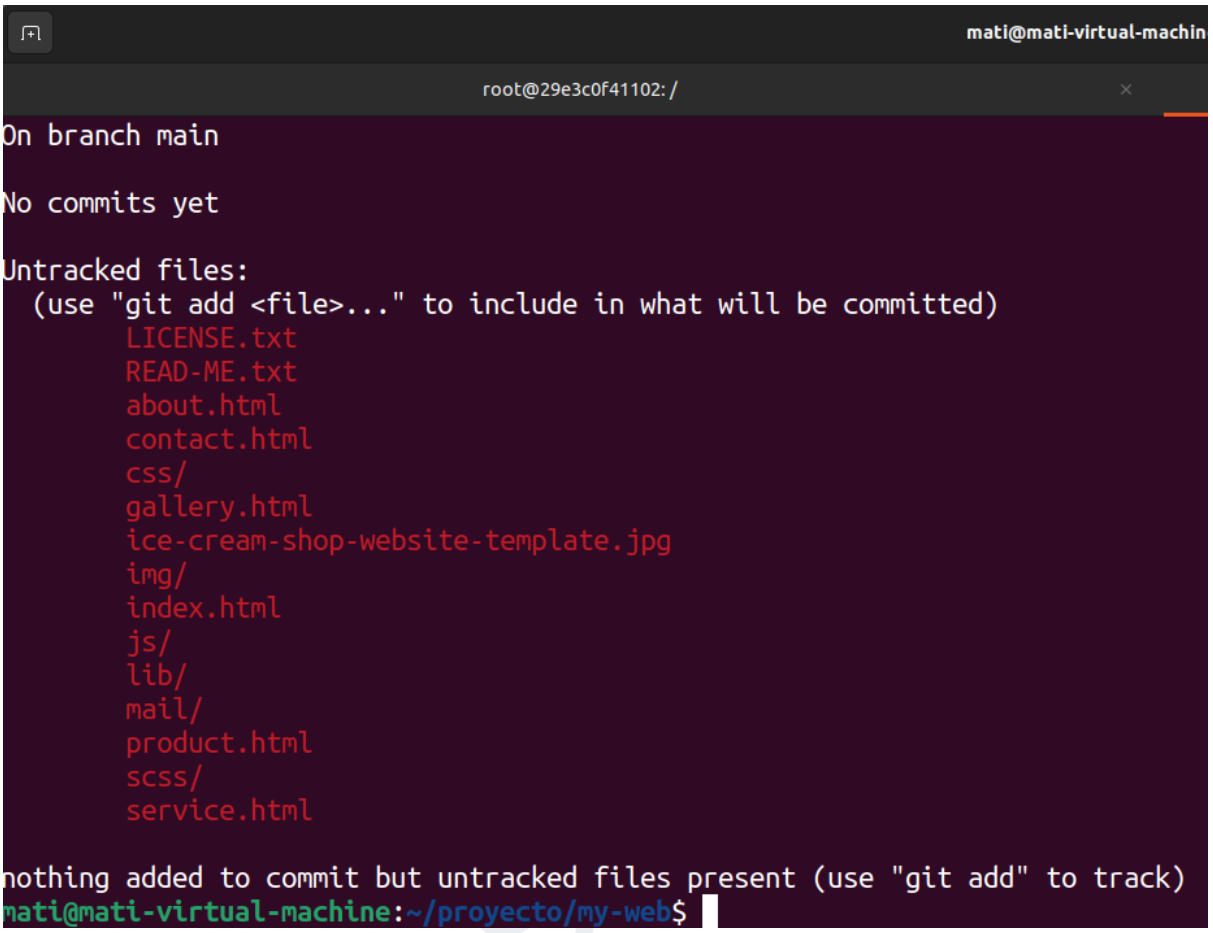
En mi caso voy a subir a dicho directorio de donde clonamos nuestro repositorio un template HTML5



Se ve de la siguiente manera:



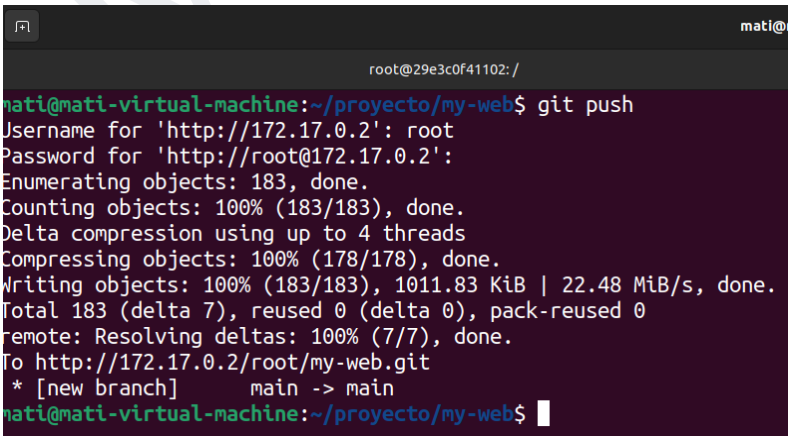
Ahora si realizo un git status me figuran todos los archivos que he copiado dentro de la carpeta.



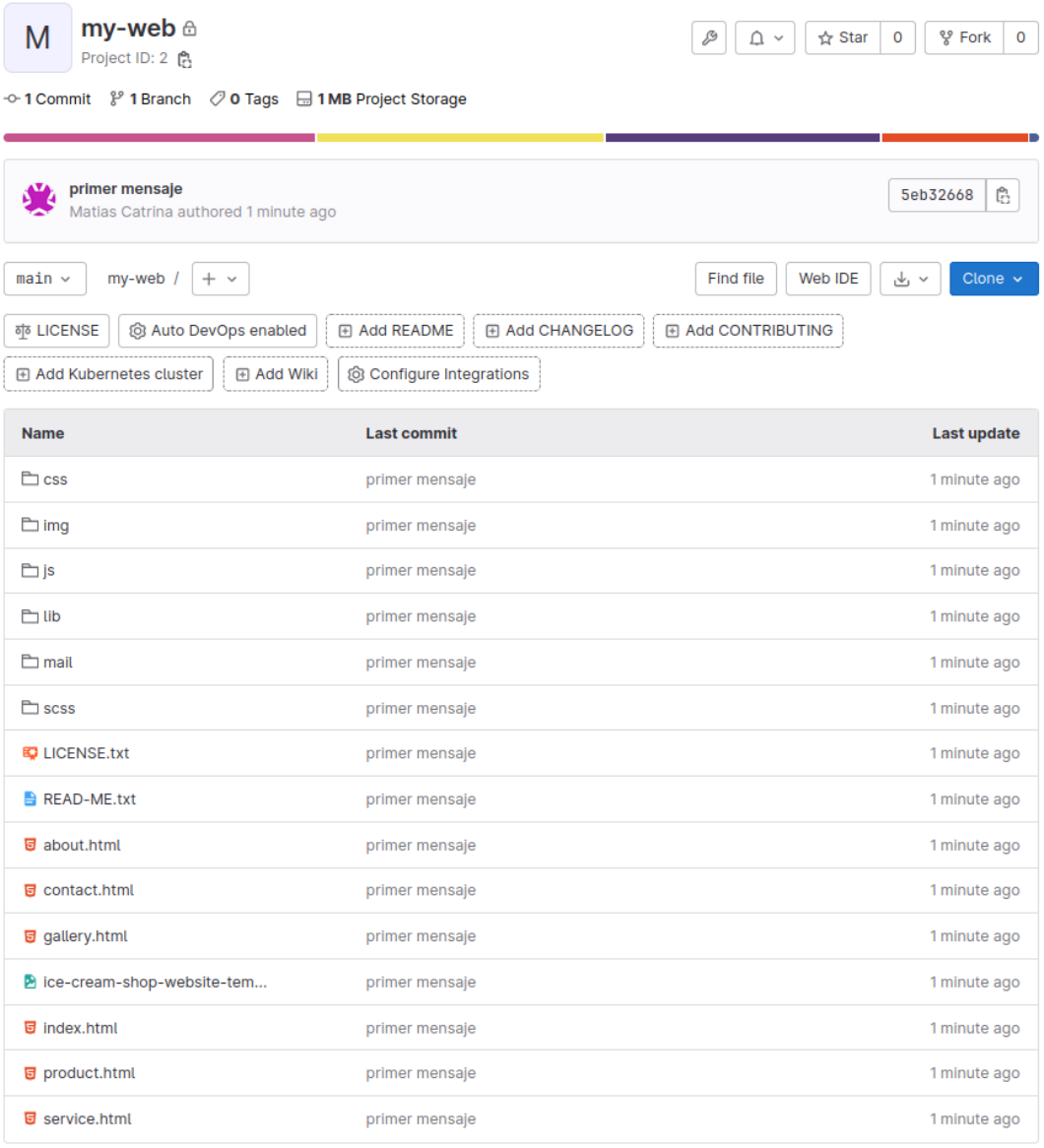
Procedemos a subirlos a nuestro gitlab con los siguientes comandos:

- **git add .**
- **git commit -m “Mensaje commit”**
- **git push**

Nuestro push quedaria algo asi:

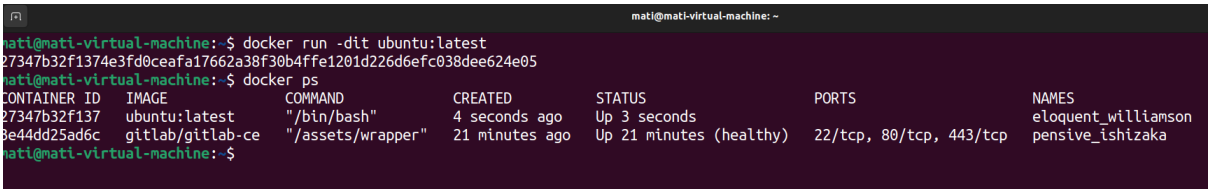


Si corroboramos en nuestro GitLab debería figurar lo siguiente:



4 - Levantamos nuestro segundo contenedor que en este caso va a ser un Apache de la siguiente manera:

docker run -dit ubuntu:latest



5 - Creamos una nueva red en Docker con el comando

docker network create <nombre de nuestra red>

```
mati@mati-virtual-machine:~$ docker network create red1
b9fc308f1a7ba071a95dc9c6153f2bd5a4b500504f243dbb73ed8114d88ec56a
mati@mati-virtual-machine:~$ docker network ls
NETWORK ID     NAME      DRIVER    SCOPE
1943ae68128b   bridge    bridge    local
ffe443daabcd   host      host      local
d3e4ed64eb73   none      null      local
b9fc308f1a7b   red1      bridge    local
mati@mati-virtual-machine:~$
```

Repasemos un poco nuestra infraestructura actual

- **1 Contenedor ejecutando GitLab.**
- **1 Contenedor ejecutando Apache.**
- **1 Red en Docker.**

Al tener 2 contenedores y entre ellos se van a comunicar, vamos a realizar la conexión a una misma red a través del siguiente comando:

docker network connect <nombre de red que creamos> <container id>

```
mati@mati-virtual-machine: ~
mati@mati-virtual-machine:~$ docker ps
CONTAINER ID   IMAGE             COMMAND             CREATED          STATUS                    PORTS                     NAMES
29e3c0f41102   ubuntu:latest     "/bin/bash"         6 minutes ago    Up 6 minutes                                        sleepy_galileo
3e44dd25ad6c   gitlab/gitlab-ce  "/assets/wrapper"   31 minutes ago   Up 31 minutes (healthy)   22/tcp, 80/tcp, 443/tcp   pensive_ishizaka
mati@mati-virtual-machine:~$ docker network ls
NETWORK ID     NAME      DRIVER    SCOPE
1943ae68128b   bridge    bridge    local
ffe443daabcd   host      host      local
d3e4ed64eb73   none      null      local
b9fc308f1a7b   red1      bridge    local
mati@mati-virtual-machine:~$ docker network connect red1 29e3c0f41102
mati@mati-virtual-machine:~$ docker network connect red1 3e44dd25ad6c
```

Podemos verificar que nuestro contenedor ya tiene seteado la nueva red usando:

docker inspect <container id>

```
            "red1": {
                "IPAMConfig": {},
                "Links": null,
                "Aliases": [
                    "29e3c0f41102"
                ],
                "NetworkID": "b9fc308f1a7ba071a95dc9c6153f2bd5a4b500504f243dbb73ed8114d88ec56a",
                "EndpointID": "d98acc1b0ba7cc970442348beff5d44f21d2097f2c9cd234cde9a03559370907",
                "Gateway": "172.19.0.1",
                "IPAddress": "172.19.0.2",
                "IPPrefixLen": 16,
                "IPv6Gateway": "",
                "GlobalIPv6Address": "",
                "GlobalIPv6PrefixLen": 0,
                "MacAddress": "02:42:ac:13:00:02",
                "DriverOpts": {}
            }
        }
    }
]
```


6 - Vamos a ingresar a nuestro contenedor de Apache para poder realizar algunas modificaciones de la siguiente manera:

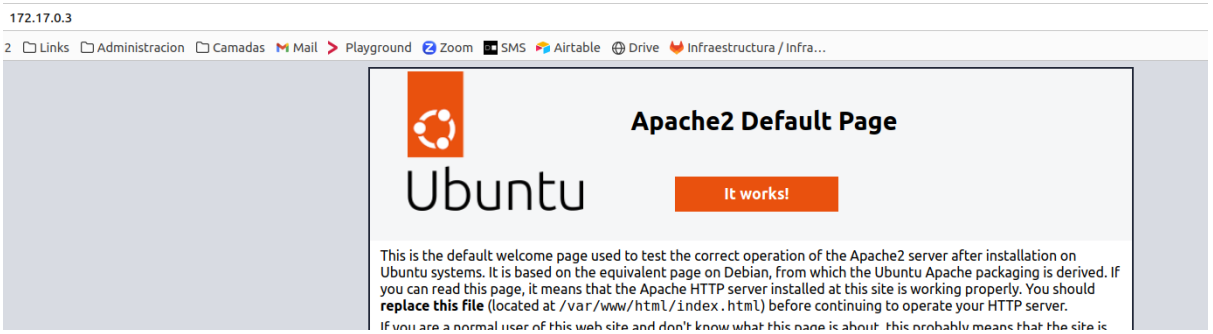
docker exec -it 29e3c0f41102 bash

Luego instalamos y actualizamos utilizando los siguientes comandos:

- ☐ **Actualizamos librerías:** *apt update*
- ☐ **Instalamos wget para después instalar nuestro runner:** *apt install wget*
- ☐ **Instalamos nuestro servidor web Apache:** *apt install apache2*
- ☐ **Instalamos git para utilizarlo más adelante:** *apt install git*
- ☐ **Inicializamos nuestro servidor:** *service apache2 start*

Finalmente deberíamos llegar a la siguiente imagen, en donde nos va a devolver la ip de nuestro servidor Apache.

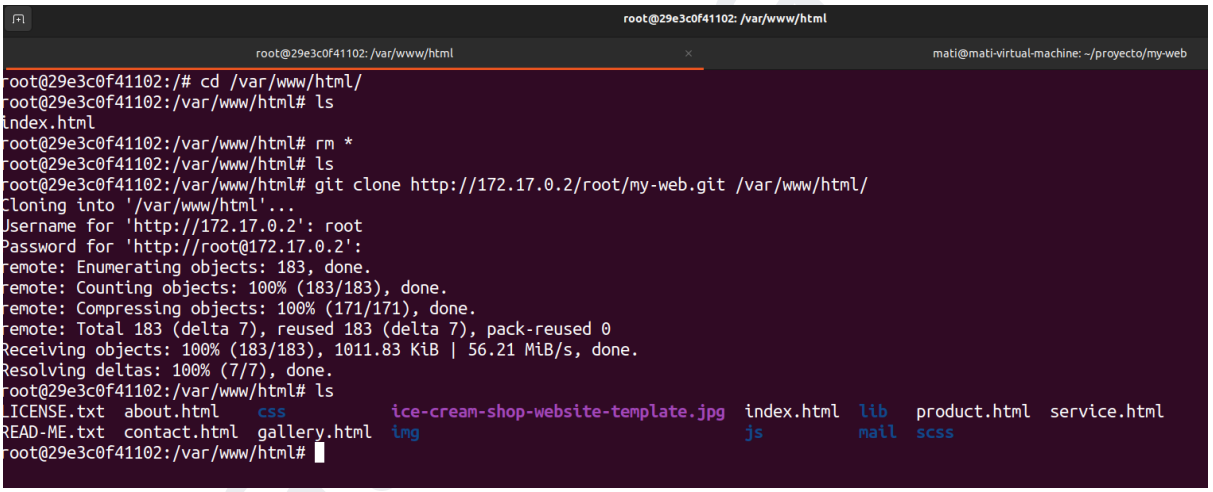
Corroboramos que efectivamente nuestro servidor esta ON ingresando al navegador la ip anterior:



7 - Nos tocaría ahora reemplazar dicha web, por nuestro proyecto que tenemos en GitLab. Ejecutamos lo siguiente:

- *cd /var/www/html* **(nos posicionamos dentro del directorio del servidor)**
- *ls* **(verificamos nuestro que está nuestro archivo index.html)**
- *rm ** **: (borramos todo lo que exista dentro del directorio actual)**
- *git clone <el que utilizamos en el paso 3> /var/www/html/* **(clonamos nuestro repositorio)**

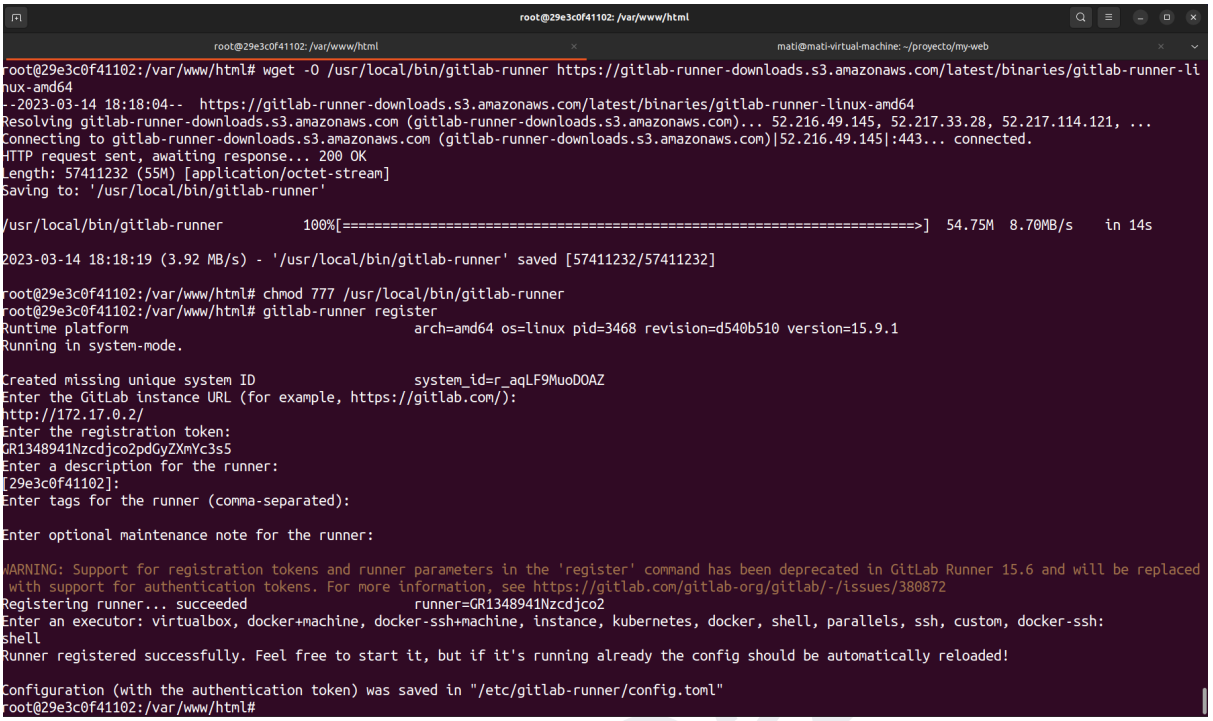
Nos debería quedar algo similar a esto:



8 - Instalamos nuestro runner en el servidor para que se ejecuten las tareas de nuestro pipeline automáticamente. Ejecutamos los siguientes comandos:

- *wget -O /usr/local/bin/gitlab-runner https://gitlab-runner-downloads.s3.amazonaws.com/latest/binaries/gitlab-runner-linux-amd64*
- *chmod 777 /usr/local/bin/gitlab-runner*
- *gitlab-runner register*

Nos quedaría de la siguiente manera:



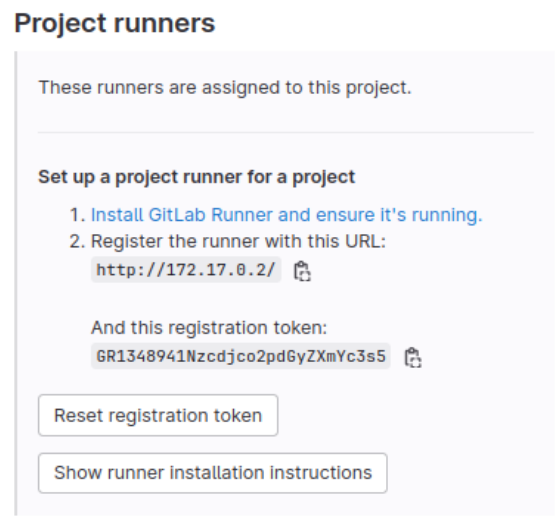
Cuando registramos el runner lo configuramos de la siguiente manera:

```
Enter the GitLab instance URL (for example, https://gitlab.com/):
http://172.17.0.2/
```

```
Enter the registration token:
GR1348941Nzcdjco2pdGyZXmYc3s5
```

Ambos datos los obtenemos desde:

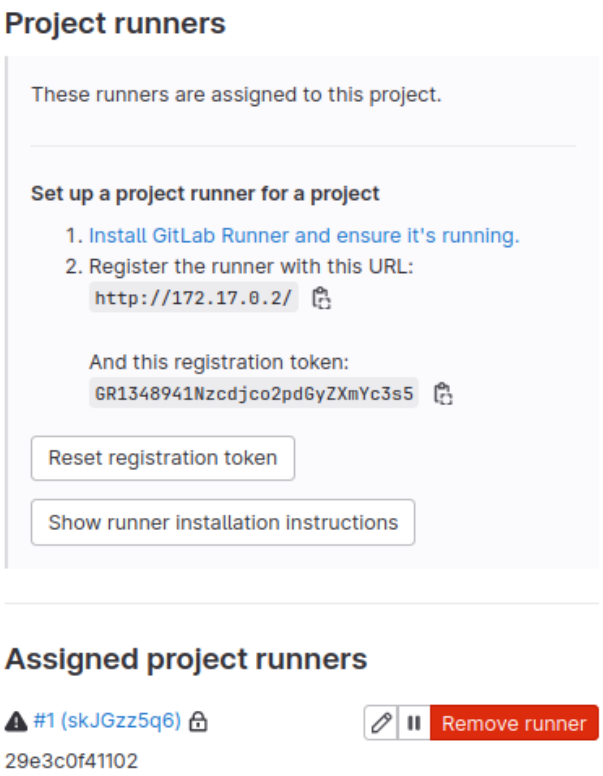
proyecto > settings > CI/CD > Runners



En las opciones de descripción, tags omitimos en caso de que quieran configurar esas opciones lo pueden hacer. Y en la ultima configuración escribimos *shell.* **Porque en este caso vamos a querer ejecutar comandos shell en este runner.**

```
Registering runner... succeeded                     runner=GR1348941Nzcdjco2
Enter an executor: virtualbox, docker+machine, docker-ssh+machine, instance, kubernetes, docker, shell, parallels, ssh, custom, docker-ssh:
shell
```

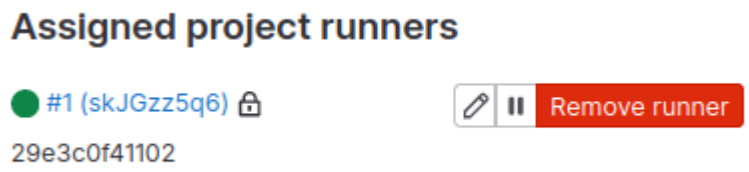
En GitLab nos encontramos con lo siguiente:



Para ejecutar activar nuestro runner, ejecutamos el comando:

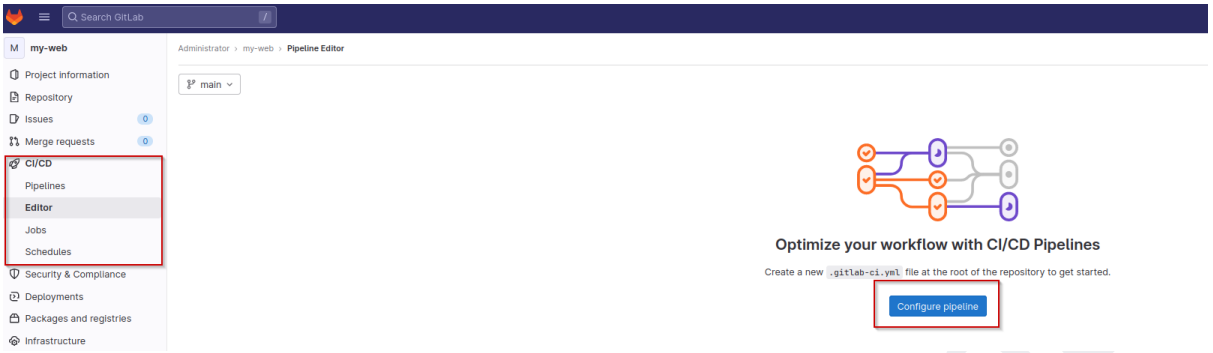
gitlab-runner run

Nos quedaría de esta manera:



9 - Creamos nuestro primer pipeline:

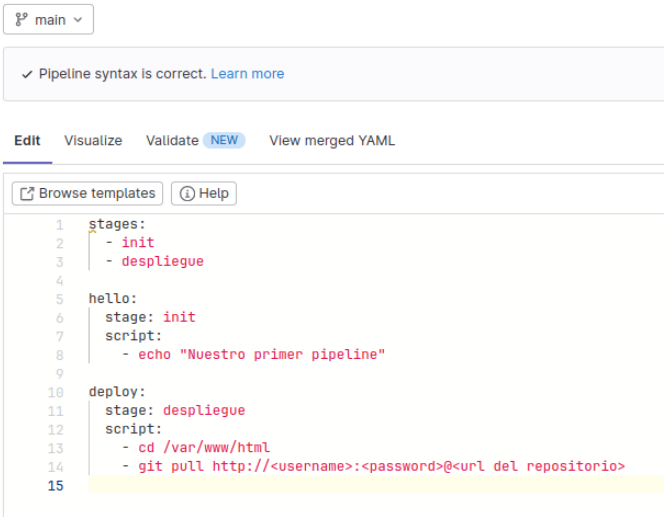
proyecto > CI/CD > Editor **y le damos click al botón** Configure pipeline



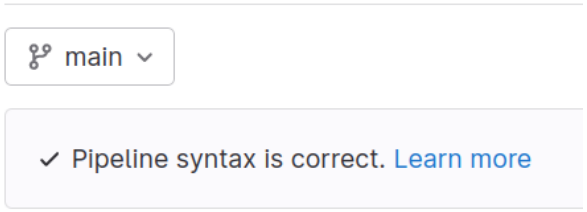
Borramos lo que esté dentro del archivo y pegamos lo siguiente (previamente modificando los parámetros username,password,url del repositorio):

```
stages:
  - init
  - despliegue

hello:
  stage: init
  script:
    - echo "Nuestro primer pipeline"

deploy:
  stage: despliegue
  script:
    - cd /var/www/html
    - git pull http://<username>:<password>@<url del repositorio>
```

Nos quedaria algo asi:

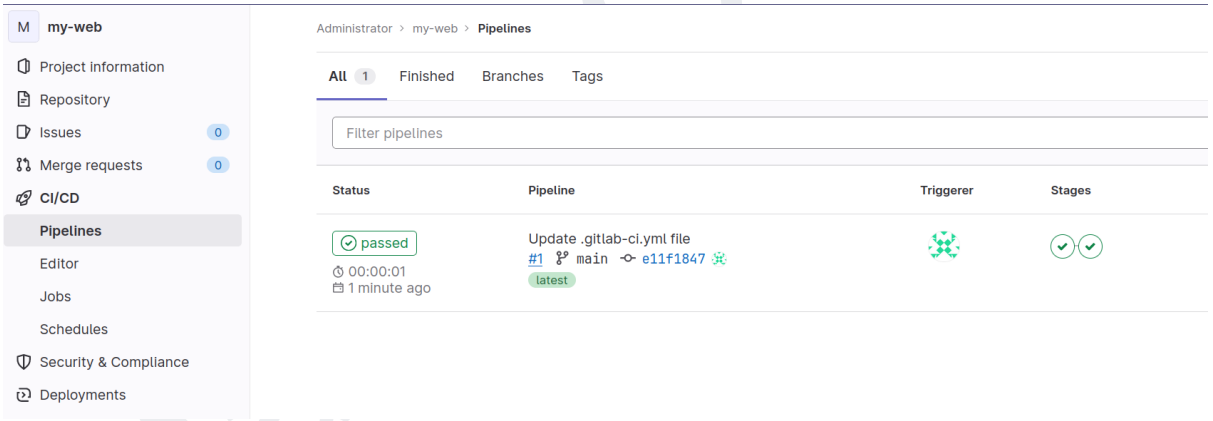


Verificamos que nos dé el check de que la sintaxis de nuestro yaml está bien escrita



Procedemos a commitear los cambios

Y podemos verificar que en pipelines ya figura en correcto funcionamiento nuestro commit:



Finalmente ya podemos realizar las modificaciones, que cuando hagamos el push se realizará de manera automática.