



**UNIVERSIDAD
MODELO**

Ingeniería en Desarrollo de Tecnología y Software

Sistemas Multidimensionales

por Mtro. Alfredo Bolio

Séptimo Semestre

Parcial 2

Reporte Segundo Parcial

Adrián L. Gorocica C.

Tabla de contenido

Objetivo.....	3
Alcance	3
Procesos	4
Repositorios	4
Creación de repositorios.....	4
Clonación de repositorios en local.....	7
Creación de contenedores.....	9
Carga de código fuente inicial	11
Creación de ramas master y develop.....	12
Nube	13
Creación de instancias LightSail	13
Instalación Docker en instancias.....	14
Clonación de repositorio.....	16
Montaje en contenedor.....	17
Configuración de aplicaciones.....	17
Seguridad y redundancia.....	18
Configuración de firewall	18
Creación de instantánea de web.....	19
Creación de segunda instancia de web	20
Creación de subdominio para web	21
Creación de balanceador de carga.....	22
Asignación de instancias a balanceador de carga.....	23
Creación de certificado SSL	24
Resumen.....	27

Objetivo

Tener un registro del flujo del desarrollo de un proyecto web haciendo uso de las tecnologías modernas. El flujo de desarrollo cubrirá las etapas de repositorio, nube y finalmente seguridad y redundancia. Las herramientas por usar serán: Github, para la creación de repositorios; Docker, para el montaje y manejo de instancias, y LightSail, para la gestión y mantenimiento de la web.

Alcance

Los puntos que se seguirán en la etapa de repositorios serán: la creación de un repositorio en Github que servirá para el almacenamiento de todo el proyecto, clonación del repositorio en local, creación de un contenedor, carga inicial de código y finalmente la creación de diferentes ramas para el correcto desarrollo; para la segunda etapa, de montaje en la nube de nuestro proyecto, se utilizará LightSail para crear las instancias, instalar Docker en la instancia creada, clonar el repositorio, montar el contenedor creado en la etapa anterior y configurar las aplicaciones; finalmente, en la última etapa, nos dedicaremos a la seguridad y redundancia al configurar el firewall, crear un subdominio, asignar un balanceador de cargas y terminaremos creando un certificado SSL.

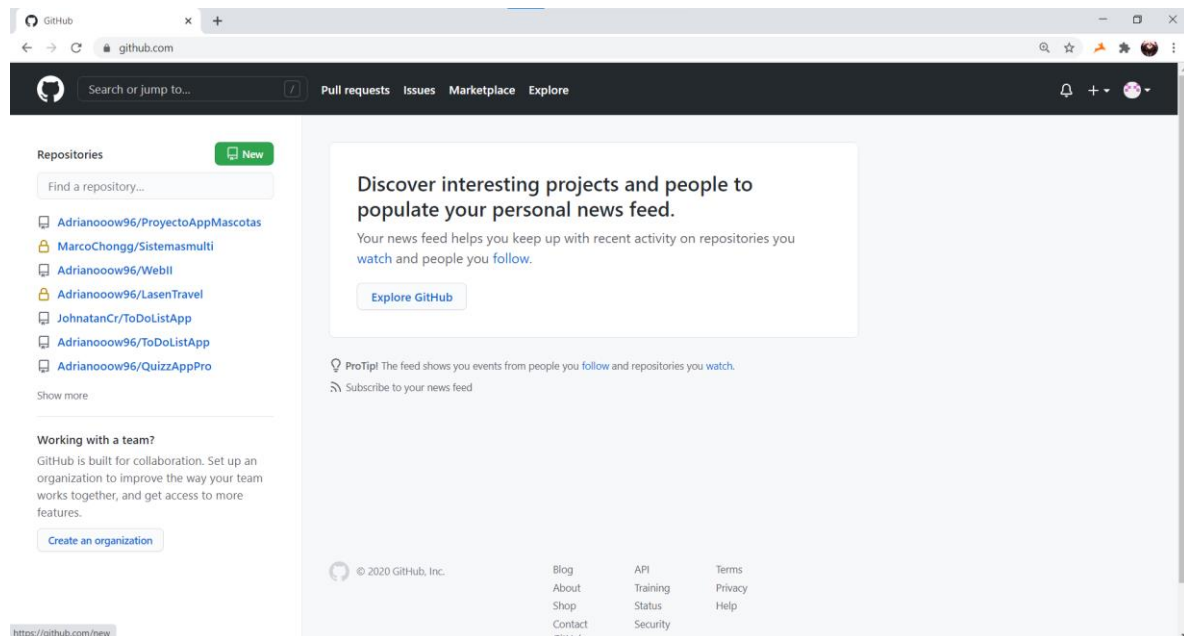
Procesos

Repositorios

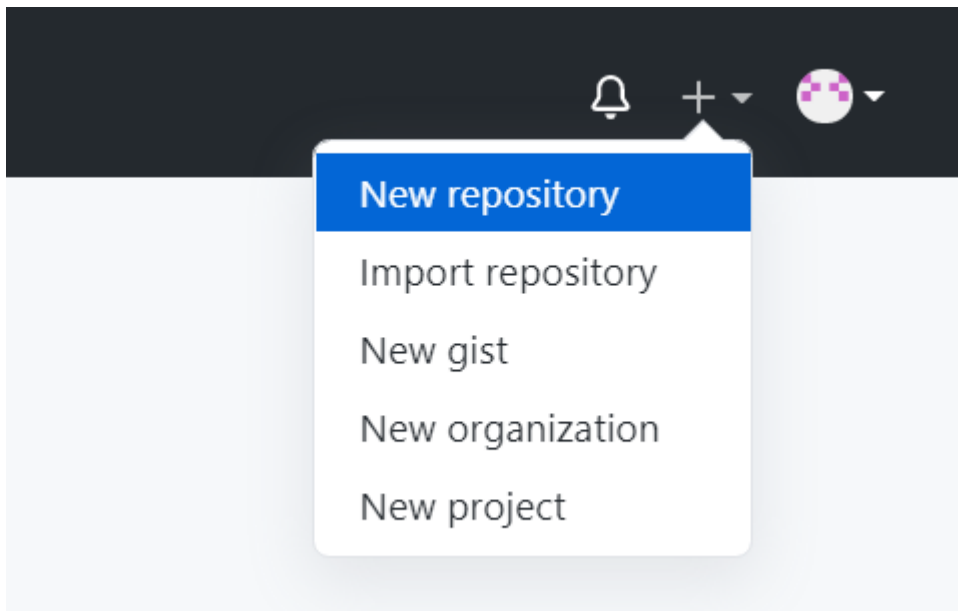
Como primer paso vamos a establecer el entorno de trabajo con un flujo de trabajo con Git, para poder tener un correcto manejo de las versiones de nuestro proyecto y un fácil acceso a nuestros archivos.

Creación de repositorios

Lo primero que deberemos hacer será crear los repositorios para el sitio web, la API y la base de datos. Para ello usaremos la plataforma llamada Github.



Con una cuenta de Github vamos a Nuevo Repositorio y esperamos a que nos cargue la siguiente pantalla.




En la nueva pantalla llenaremos los datos solicitados, brindando un nombre al repositorio, una descripción opcional y otros detalles de preferencias:

Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere?

[Import a repository.](#)

Owner *

 Adrianooow96 ▾



Repository name *

/ apiPrueba ✓

Great repository names are short and memorable. Need inspiration? How about [scaling-giggle?](#)

Description (optional)

Repositorio para la api de los ejercicios del segundo parcial de la materia de Sistemas Multidimensionales

- ☐  **Public**
Anyone on the internet can see this repository. You choose who can commit.
- ☒  **Private**
You choose who can see and commit to this repository.

Initialize this repository with:

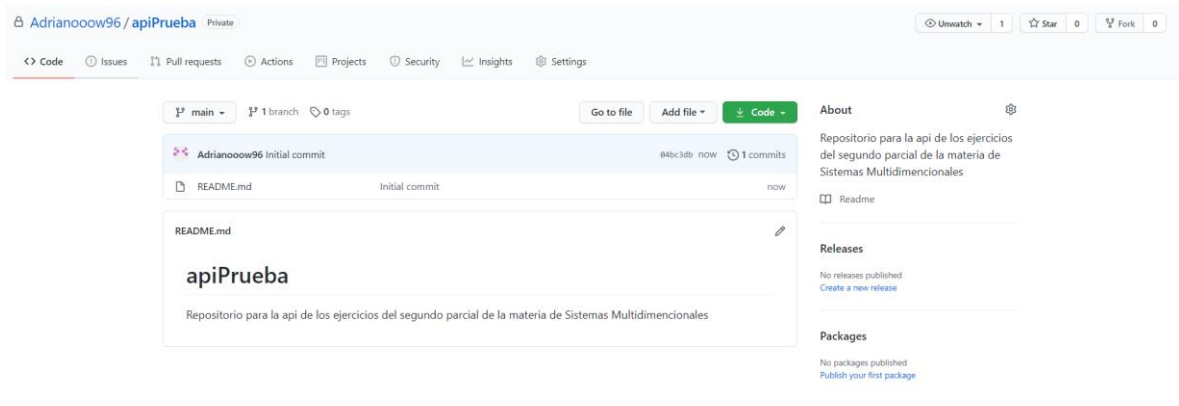
Skip this step if you're importing an existing repository.

- ☒ **Add a README file**
This is where you can write a long description for your project. [Learn more.](#)
- ☐ **Add .gitignore**
Choose which files not to track from a list of templates. [Learn more.](#)
- ☐ **Choose a license**
A license tells others what they can and can't do with your code. [Learn more.](#)

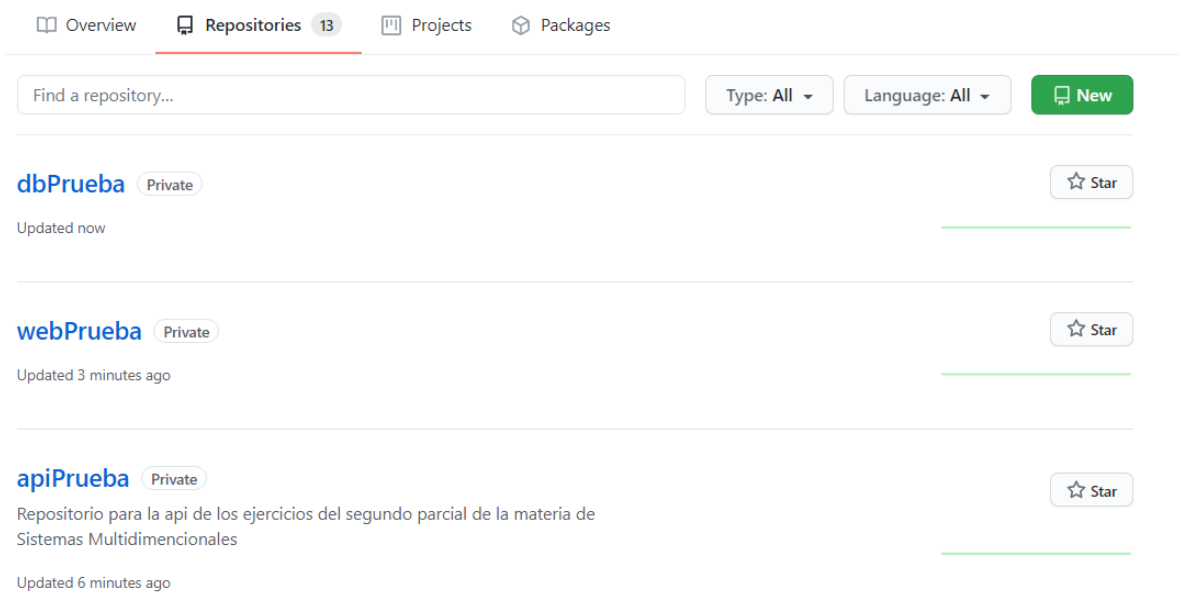
This will set  `main` as the default branch. Change the default name in your [settings](#).

Create repository

Finalmente damos a Create Repository para crear el repositorio con la configuración brindada. Si todo salió correctamente veremos la siguiente pantalla:



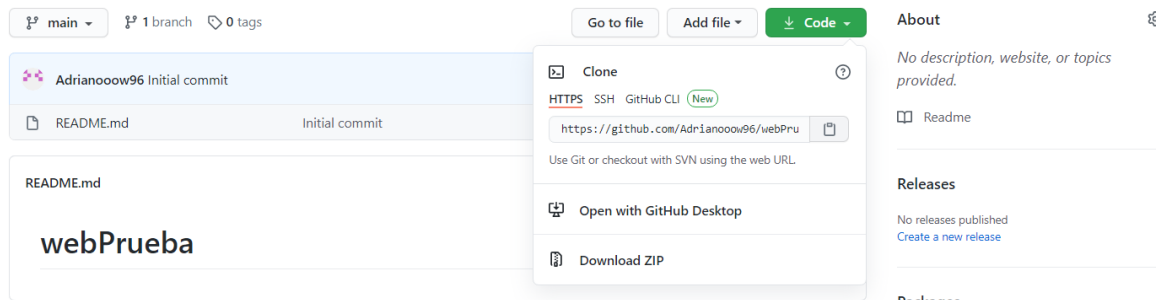
Repetiremos los pasos anteriores para el repositorio de base de datos y el de los archivos de la web.



Clonación de repositorios en local

Una vez con los repositorios creados, procederemos a clonarlos para tener acceso y gestión de ellos desde nuestro dispositivo. Para ello ejecutaremos el comando `git clone` en la consola, ubicados en el directorio que deseemos. El enlace

necesario es otorgado por cada repositorio en la siguiente sección:

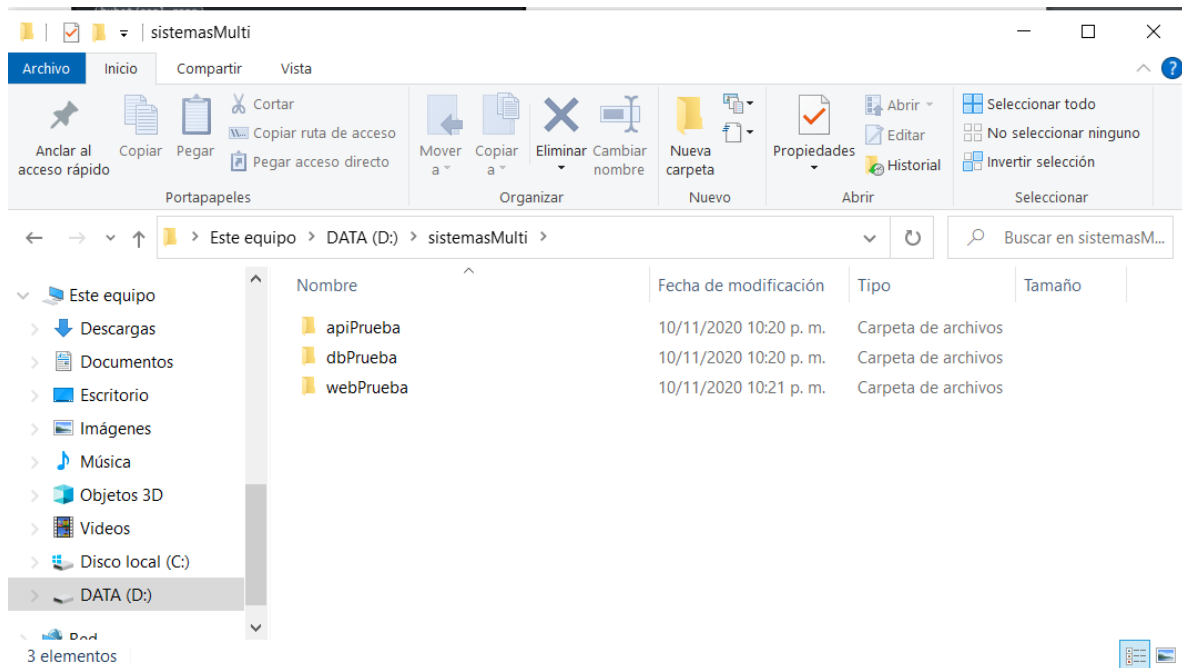


En la siguiente captura mostramos la ejecución exitosa de los comandos necesarios:

```
Windows PowerShell

PS D:\sistemasMulti> git clone https://github.com/Adrianooow96/webPrueba.git
Cloning into 'webPrueba'...
remote: Enumerating objects: 3, done.
remote: Counting objects: 100% (3/3), done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (3/3), 594 bytes | 2.00 KiB/s, done.
PS D:\sistemasMulti> git clone https://github.com/Adrianooow96/dbPrueba
Cloning into 'dbPrueba'...
remote: Enumerating objects: 3, done.
remote: Counting objects: 100% (3/3), done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (3/3), 595 bytes | 3.00 KiB/s, done.
PS D:\sistemasMulti> git clone https://github.com/Adrianooow96/apiPrueba
Cloning into 'apiPrueba'...
remote: Enumerating objects: 3, done.
remote: Counting objects: 100% (3/3), done.
remote: Compressing objects: 100% (2/2), done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (3/3), 673 bytes | 3.00 KiB/s, done.
PS D:\sistemasMulti>
```

Como resultado tendremos los directorios de los repositorios clonados, como podemos ver en la siguiente captura:



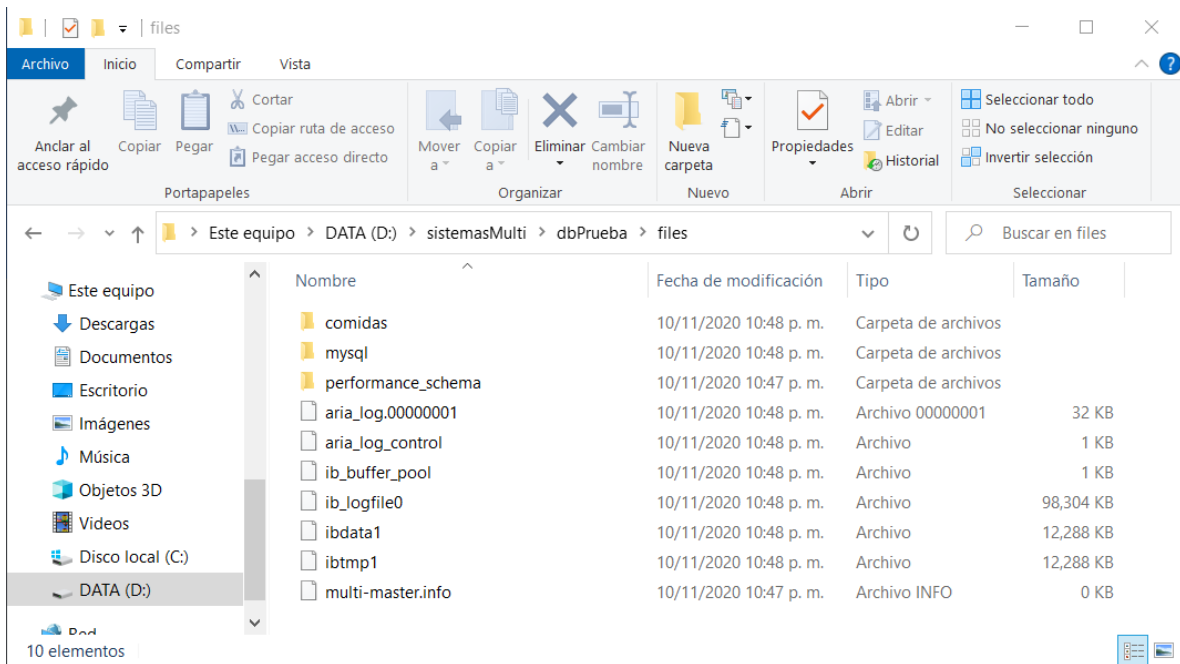
Creación de contenedores

Como siguiente paso, crearemos un archivo docker-compose que nos servirá para ambientar nuestros contenedores. Una vez creados los archivos en cada carpeta, procederemos a correr el siguiente comando en cada una de nuestras carpetas:

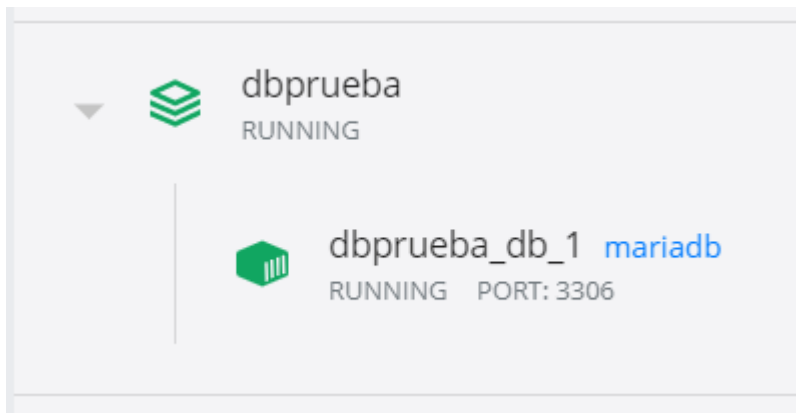
docker-compose up

Y una vez finalizado podremos visualizar los cambios en el directorio.

```
Windows PowerShell
PS D:\sistemasMulti\dbPrueba> docker-compose up -d
Creating network "dbprueba_default" with the default driver
Pulling db (mariadb:...)
latest: Pulling from library/mariadb
Digest: sha256:2960a3d1ddb35dd454066b45005b4f694e18af76648833f1b9d93ab90cee7cf2
Status: Downloaded newer image for mariadb:latest
Creating dbprueba_db_1 ... done
PS D:\sistemasMulti\dbPrueba> cd ..
PS D:\sistemasMulti>
```



Y también podemos visualizar desde Docker que nuestro contenedor está funcionando:



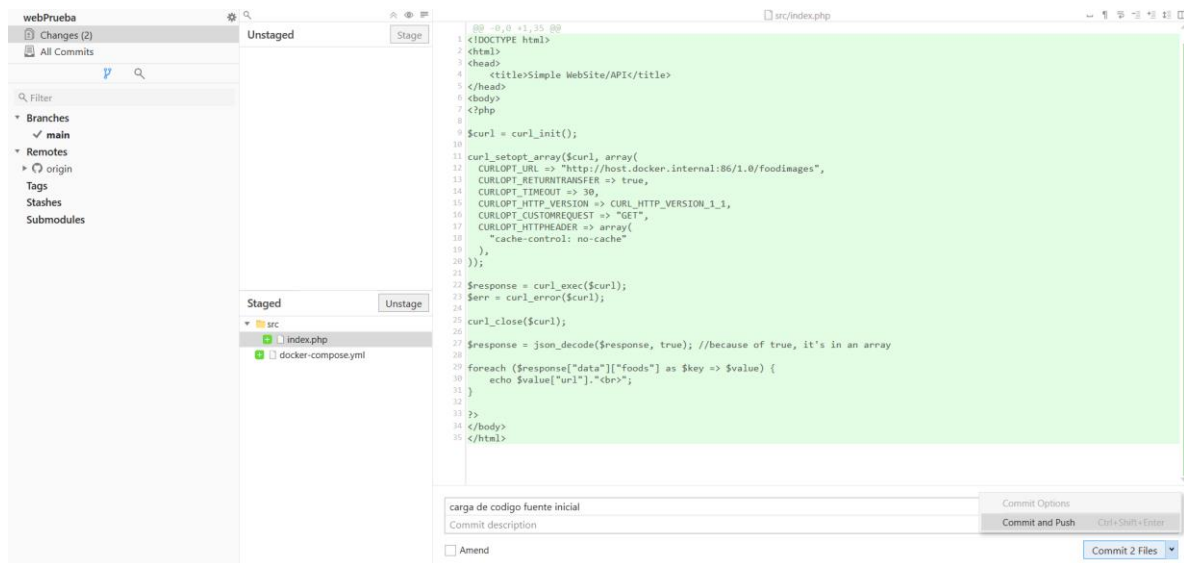
Haremos lo mismo en los otros repositorios para tener todos los contenedores listos.

The screenshot displays the Docker Desktop interface with three services running:

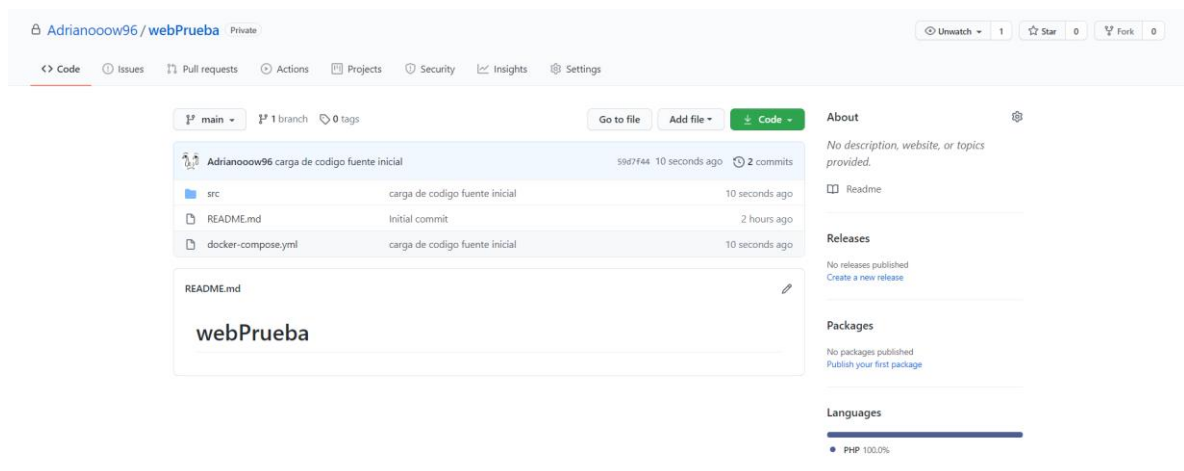
- dbprueba** (RUNNING)
 - dbprueba_db_1** (RUNNING) [mariadb](#) (PORT: 3306)
- webprueba** (RUNNING)
 - websiteclase** (RUNNING) [webdevops/php-apache](#) (PORT: 80)
- apiprueba** (RUNNING)
 - nginxapiclase** (RUNNING) [nginx](#) (PORT: 86)
 - phpapiclase** (RUNNING) [apiprueba_phpapiclase](#) (PORT: 8006)

Carga de código fuente inicial

Seguido de ello vamos a utilizar una herramienta llamada Fork para hacer manejo de nuestros repositorios y cargar todos los archivos que hemos creado.

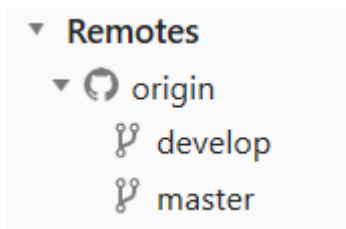


Y podemos ver que nuestros repositorios son actualizados con el código inicial:



Creación de ramas master y develop

Como último paso en esta etapa, vamos a crear una rama develop, que es en la que trabajaremos y una rama master para lanzar nuestro código cuando esté finalizado.



Nube

Una vez que tengamos todo el proyecto terminado, comienza la parte de pasarlo a la nube. Para ello vamos a utilizar LightSail, una plataforma web que nos servirá durante esta etapa y la siguiente.


Creación de instancias LightSail

Como primer paso, vamos a crear nuestras instancias para cada uno de nuestros contenedores. Este proceso ya lo hemos documentado en el archivo Tarea 1 – Parcial 2 – Gorocica Coral Adrian.

Al finalizar el proceso debemos poder observar las instancias:

 Ireland (eu-west-1)

ZONE A


**dbGorocica**
512 MB RAM, 1 vCPU, 20 GB SSD

Running

52.214.157.229
Ireland, Zone A


 Montreal (ca-central-1)

ZONE A

**apiGorocica**
512 MB RAM, 1 vCPU, 20 GB SSD

Running

35.182.38.79
Montreal, Zone A

**webGorocica**
512 MB RAM, 1 vCPU, 20 GB SSD

Running

3.96.188.131
Montreal, Zone A

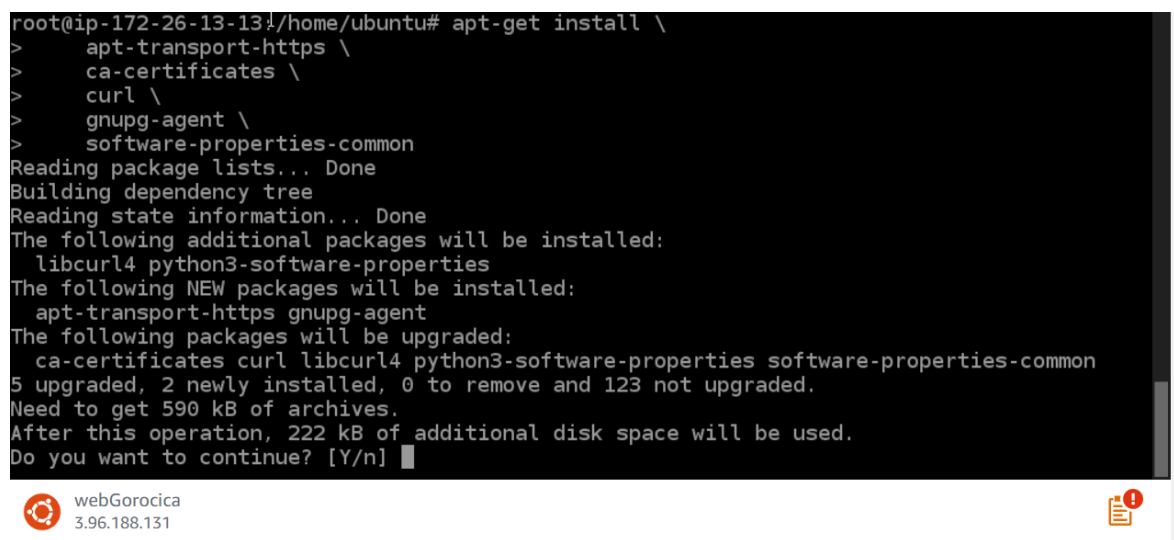
Instalación Docker en instancias

Como siguiente paso vamos a instalar Docker en las instancias creadas para poder tener las instancias ambientadas en la nube.

Para ello accederemos al siguiente enlace

<https://docs.docker.com/engine/install/ubuntu/> y ejecutaremos lo siguiente en la consola de las instancias:

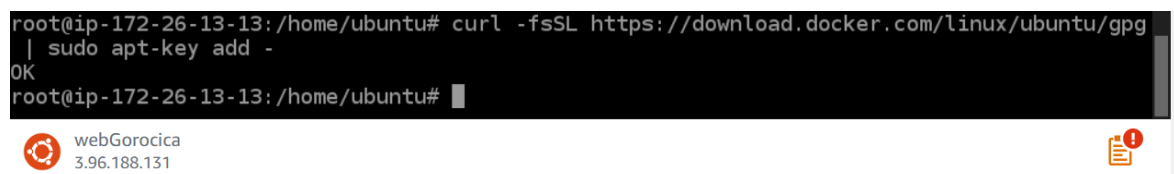
```
root@ip-172-26-13-13:/home/ubuntu# apt-get install \
> apt-transport-https \
> ca-certificates \
> curl \
> gnupg-agent \
> software-properties-common
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  libcurl4 python3-software-properties
The following NEW packages will be installed:
  apt-transport-https gnupg-agent
The following packages will be upgraded:
  ca-certificates curl libcurl4 python3-software-properties software-properties-common
5 upgraded, 2 newly installed, 0 to remove and 123 not upgraded.
Need to get 590 kB of archives.
After this operation, 222 kB of additional disk space will be used.
Do you want to continue? [Y/n]
```



webGorocica
3.96.188.131

Aceptamos y dejamos que la instalación continúe.

```
root@ip-172-26-13-13:/home/ubuntu# curl -fsSL https://download.docker.com/linux/ubuntu/gpg
| sudo apt-key add -
OK
root@ip-172-26-13-13:/home/ubuntu#
```



webGorocica
3.96.188.131

```

root@ip-172-26-13-13:/home/ubuntu# curl -fsSL https://download.docker.com/linux/ubuntu/gpg
| sudo apt-key add -
OK
root@ip-172-26-13-13:/home/ubuntu# apt-key fingerprint 0EBFCD88
pub   rsa4096 2017-02-22 [SCEA]
      9DC8 5822 9FC7 DD38 854A  E2D8 8D81 803C 0EBF CD88
uid           [ unknown] Docker Release (CE deb) <docker@docker.com>
sub   rsa4096 2017-02-22 [S]

root@ip-172-26-13-13:/home/ubuntu#

```



webGorocica
3.96.188.131



```

root@ip-172-26-13-13:/home/ubuntu# add-apt-repository "deb [arch=amd64] https://download.d
ocker.com/linux/ubuntu $(lsb_release -cs) stable"
Hit:1 http://ca-central-1.ec2.archive.ubuntu.com/ubuntu focal InRelease
Get:2 http://ca-central-1.ec2.archive.ubuntu.com/ubuntu focal-updates InRelease [111 kB]
Get:3 http://ca-central-1.ec2.archive.ubuntu.com/ubuntu focal-backports InRelease [98.3 kB
]
Get:4 https://download.docker.com/linux/ubuntu focal InRelease [36.2 kB]
Hit:5 http://security.ubuntu.com/ubuntu focal-security InRelease
Get:6 http://ca-central-1.ec2.archive.ubuntu.com/ubuntu focal-updates/main amd64 Packages
[652 kB]
Get:7 https://download.docker.com/linux/ubuntu focal/stable amd64 Packages [3684 B]
Fetched 902 kB in 1s (1590 kB/s)
Reading package lists... Done
root@ip-172-26-13-13:/home/ubuntu#

```



webGorocica
3.96.188.131



```

root@ip-172-26-13-13:/home/ubuntu# apt-get update
Hit:1 http://ca-central-1.ec2.archive.ubuntu.com/ubuntu focal InRelease
Hit:2 http://ca-central-1.ec2.archive.ubuntu.com/ubuntu focal-updates InRelease
Hit:3 http://ca-central-1.ec2.archive.ubuntu.com/ubuntu focal-backports InRelease
Hit:4 http://security.ubuntu.com/ubuntu focal-security InRelease
Hit:5 https://download.docker.com/linux/ubuntu focal InRelease
Reading package lists... Done
root@ip-172-26-13-13:/home/ubuntu# apt-get install docker-ce docker-ce-cli containerd.io
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  aufs-tools cgroupfs-mount pigz
The following NEW packages will be installed:
  aufs-tools cgroupfs-mount containerd.io docker-ce docker-ce-cli pigz
0 upgraded, 6 newly installed, 0 to remove and 123 not upgraded.
Need to get 91.2 MB of archives.
After this operation, 410 MB of additional disk space will be used.
Do you want to continue? [Y/n]

```



webGorocica
3.96.188.131



Aceptamos una vez más para finalizar.

Seguido de ello realizaremos la instalación de Docker Compose desde el enlace

<https://docs.docker.com/compose/install/> .

```

root@ip-172-26-13-13:/home/ubuntu# curl -L "https://github.com/docker/compose/releases/download/1.27.4/docker-compose-$(uname -s)-$(uname -m)" -o /usr/local/bin/docker-compose
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left   Speed
100 651    100 651    0     0  4458      0  --:--:-- --:--:-- --:--:--  4458
100 11.6M  100 11.6M    0     0 20.7M      0  --:--:-- --:--:-- --:--:-- 20.7M
root@ip-172-26-13-13:/home/ubuntu#

```

webGorocica
3.96.188.131



```

root@ip-172-26-13-13:/home/ubuntu# chmod +x /usr/local/bin/docker-compose
root@ip-172-26-13-13:/home/ubuntu#

```

webGorocica
3.96.188.131



Podemos verificar la instalación escribiendo docker-compose:

```

root@ip-172-26-13-13:/home/ubuntu# docker-compose
Define and run multi-container applications with Docker.

Usage:
  docker-compose [-f <arg>...] [options] [--] [COMMAND] [ARGS...]
  docker-compose -h|--help

Options:
  -f, --file FILE             Specify an alternate compose file
                              (default: docker-compose.yml)
  -p, --project-name NAME     Specify an alternate project name
                              (default: directory name)
  -c, --context NAME          Specify a context name
  --verbose                   Show more output
  --log-level LEVEL           Set log level (DEBUG, INFO, WARNING, ERROR, CRITICAL)
  --no-ansi                   Do not print ANSI control characters
  -v, --version               Print version and exit
  -H, --host HOST             Daemon socket to connect to

  --tls                       Use TLS; implied by --tlsverify
  --tlscacert CA_PATH         Trust certs signed only by this CA
  --tlscert CLIENT_CERT_PATH  Path to TLS certificate file
  --tlskey TLS_KEY_PATH       Path to TLS key file
  --tlsverify                 Use TLS and verify the remote
  --skip-hostname-check       Don't check the daemon's hostname against the
                              name specified in the client certificate
  --project-directory PATH    Specify an alternate working directory
                              (default: the path of the Compose file)
  --compatibility              If set, Compose will attempt to convert keys

```

webGorocica
3.96.188.131

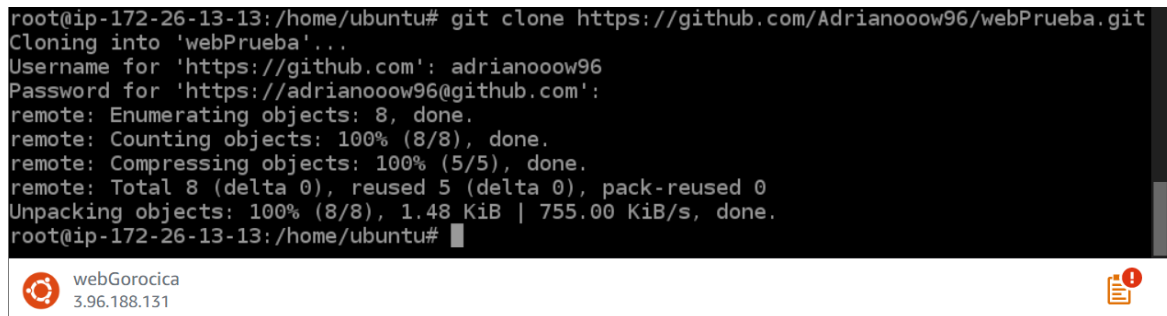


Clonación de repositorio

Una vez que esté instalado docker en nuestras instancias, procederemos a clonar nuestros repositorios de la misma forma que hicimos en el punto 3.1.2 pero ahora

en la consola de nuestras instancias.

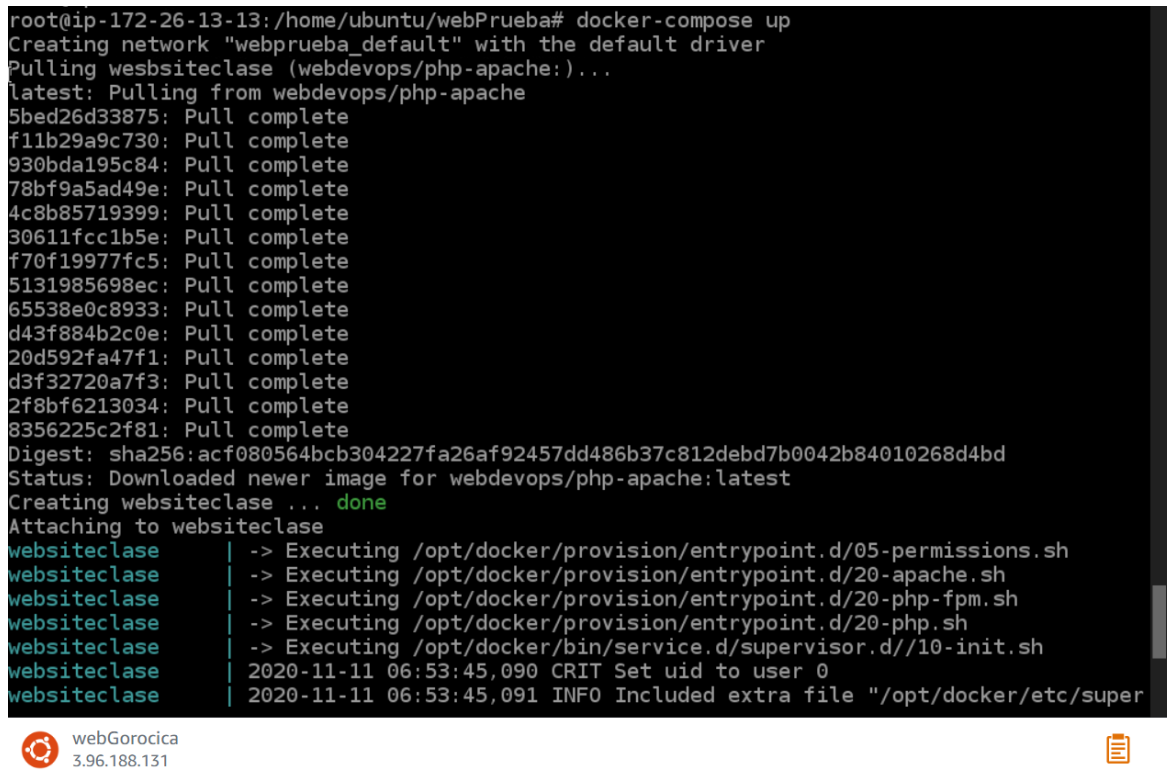
```
root@ip-172-26-13-13:/home/ubuntu# git clone https://github.com/Adrianooow96/webPrueba.git
Cloning into 'webPrueba'...
Username for 'https://github.com': adrianooow96
Password for 'https://adrianooow96@github.com':
remote: Enumerating objects: 8, done.
remote: Counting objects: 100% (8/8), done.
remote: Compressing objects: 100% (5/5), done.
remote: Total 8 (delta 0), reused 5 (delta 0), pack-reused 0
Unpacking objects: 100% (8/8), 1.48 KiB | 755.00 KiB/s, done.
root@ip-172-26-13-13:/home/ubuntu#
```

A terminal window with a black background and white text. It shows the command 'git clone https://github.com/Adrianooow96/webPrueba.git' and its output, including cloning progress and statistics. The prompt is 'root@ip-172-26-13-13:/home/ubuntu#'. Below the terminal, there is a status bar with a red circular icon, the text 'webGorocica 3.96.188.131', and a red icon with an exclamation mark.

Montaje en contenedor

Ya con nuestros repositorios clonados podemos continuar con el montaje de nuestros contenedores al igual que en el punto 3.1.3, pero ahora en la consola de nuestras instancias:

```
root@ip-172-26-13-13:/home/ubuntu/webPrueba# docker-compose up
Creating network "webprueba_default" with the default driver
Pulling wesbsiteclase (webdevops/php-apache:)...
latest: Pulling from webdevops/php-apache
5bed26d33875: Pull complete
f11b29a9c730: Pull complete
930bda195c84: Pull complete
78bf9a5ad49e: Pull complete
4c8b85719399: Pull complete
30611fcc1b5e: Pull complete
f70f19977fc5: Pull complete
5131985698ec: Pull complete
65538e0c8933: Pull complete
d43f884b2c0e: Pull complete
20d592fa47f1: Pull complete
d3f32720a7f3: Pull complete
2f8bf6213034: Pull complete
8356225c2f81: Pull complete
Digest: sha256:acf080564bcb304227fa26af92457dd486b37c812debd7b0042b84010268d4bd
Status: Downloaded newer image for webdevops/php-apache:latest
Creating wesbsiteclase ... done
Attaching to wesbsiteclase
wesbsiteclase | -> Executing /opt/docker/provision/entrypoint.d/05-permissions.sh
wesbsiteclase | -> Executing /opt/docker/provision/entrypoint.d/20-apache.sh
wesbsiteclase | -> Executing /opt/docker/provision/entrypoint.d/20-php-fpm.sh
wesbsiteclase | -> Executing /opt/docker/provision/entrypoint.d/20-php.sh
wesbsiteclase | -> Executing /opt/docker/bin/service.d/supervisor.d//10-init.sh
wesbsiteclase | 2020-11-11 06:53:45,090 CRIT Set uid to user 0
wesbsiteclase | 2020-11-11 06:53:45,091 INFO Included extra file "/opt/docker/etc/super
```

A terminal window with a black background and white text. It shows the command 'docker-compose up' and its output, including network creation, image pulling progress, and container startup logs. The prompt is 'root@ip-172-26-13-13:/home/ubuntu/webPrueba#'. Below the terminal, there is a status bar with a red circular icon, the text 'webGorocica 3.96.188.131', and a red icon with an exclamation mark.

Configuración de aplicaciones

Finalmente terminamos de ambientar nuestros contenedores como lo hicimos de manera local con los archivos que no se cargan al repositorio. Y con eso tendremos todo nuestro proyecto en la nube listo para usar.

Seguridad y redundancia

Después de tener nuestra aplicación montada en la nube, hace falta configurar la seguridad y redundancia.

Configuración de firewall

Primero configuraremos nuestras reglas de firewall para que sólo exista acceso a lo que realmente necesitamos de cada instancia.






Para la instancia de la base de datos quedaría de la siguiente forma:

Firewall

Create rules to open ports to the internet, or to a specific IP address or range.

[Learn more about firewall rules](#) 

 Add rule

Application	Protocol	Port or range / Code	Restricted to		
SSH	TCP	22	Any IP address Lightsail browser SSH/RDP 		
Custom	TCP	3886	Any IP address		









Para la instancia de la API quedaría lo siguiente:

Firewall ?

Create rules to open ports to the internet, or to a specific IP address or range.

[Learn more about firewall rules](#)

[+ Add rule](#)

Application	Protocol	Port or range / Code	Restricted to		
SSH	TCP	22	Any IP address Lightsail browser SSH/RDP ?		
HTTP	TCP	80	Any IP address		
Custom	TCP	86	Any IP address		
HTTPS	TCP	443	Any IP address		









Y para la instancia de la página web

Firewall ?

Create rules to open ports to the internet, or to a specific IP address or range.

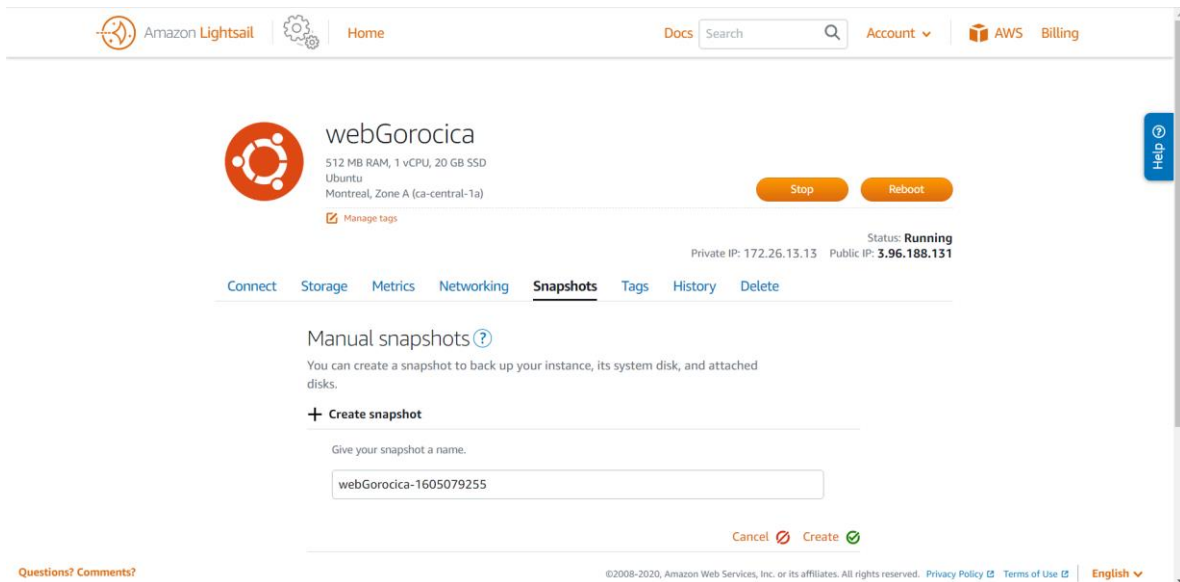
[Learn more about firewall rules](#)

[+ Add rule](#)

Application	Protocol	Port or range / Code	Restricted to		
SSH	TCP	22	Any IP address Lightsail browser SSH/RDP ?		
HTTP	TCP	80	Any IP address		
Custom	TCP	86	Any IP address		
HTTPS	TCP	443	Any IP address		

Creación de instantánea de web

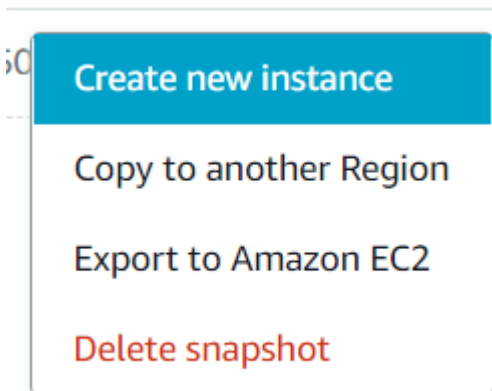
Para poder hacer redundante nuestro sistema, necesitamos copiar la instancia que tenemos de nuestra instancia web. Para ello iremos al apartado Snapshot ubicado dentro de la gestión de nuestra instancia:



Esperamos a que se termine de crear el snapshot para poder avanzar al siguiente punto.


Creación de segunda instancia de web

Una vez lista nuestra instantánea, podemos ir a sus opciones para encontrar la opción Create new instance, que nos servirá para crear una instancia basada en la instantánea, lo que nos es práctico ya que necesitamos una copia de nuestra instancia original.




We will

Al finalizar veremos las dos snapshots coexistiendo como en la siguiente captura:

 Montreal (ca-central-1)

ZONE A




webGorocica

512 MB RAM, 1 vCPU, 20 GB SSD

Running

3.96.188.131
Montreal, Zone A

ZONE B



web2Gorocica

512 MB RAM, 1 vCPU, 20 GB SSD

Running

35.183.185.52
Montreal, Zone B

Creación de subdominio para web

Como siguiente paso vamos a crear un subdominio para acceder a la web que estamos desarrollando.

Para ello accedemos al apartado de Networking dentro de LightSail, accedemos a la configuración de la DNS a la que le vamos a crear un subdominio:

Good morning!

Filter by name, location, tag, or type

[Instances](#) [Databases](#) **[Networking](#)** [Storage](#) [Snapshots](#)

Create static IP

Create DNS zone

Create load balancer

Create distribution

[Learn more about network resources](#)

Sort by **Region** and then by **Type**

 Global

DNS ZONES



idts.com.mx
DNS zone

Manage

Delete

Y creamos un nuevo registro con el subdominio que deseemos:

DNS records

Lightsail currently supports A, CNAME, MX, NS, SRV, and TXT record types.

[Learn about DNS record types](#)

A record



— Associate your domain or a subdomain with an IP address.



Subdomain

gorocica .idts.com.mx

Resolves to

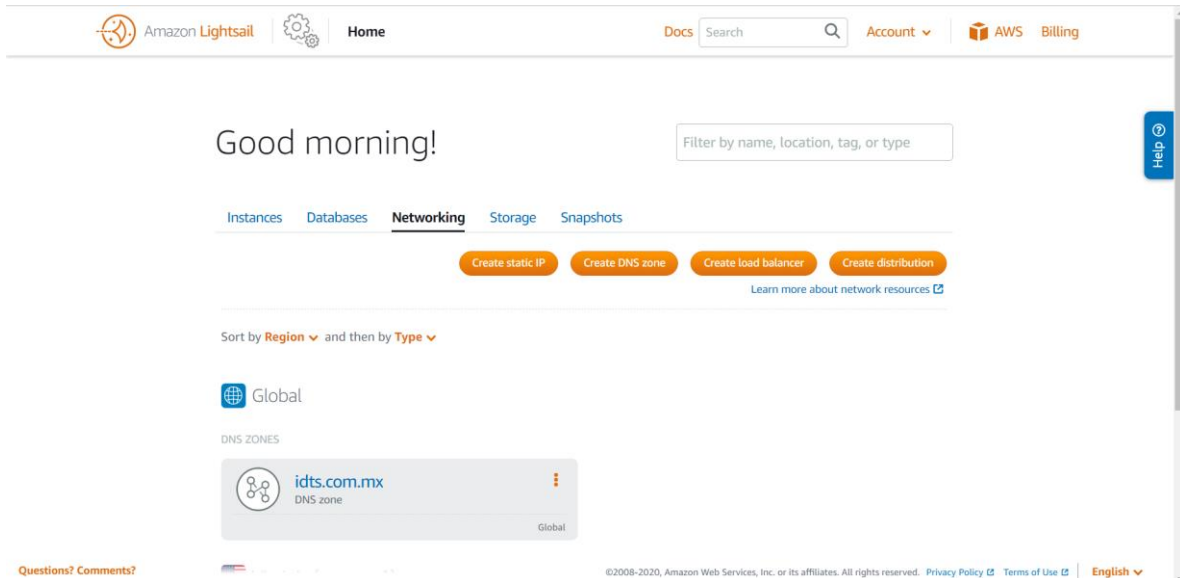
webGorocica



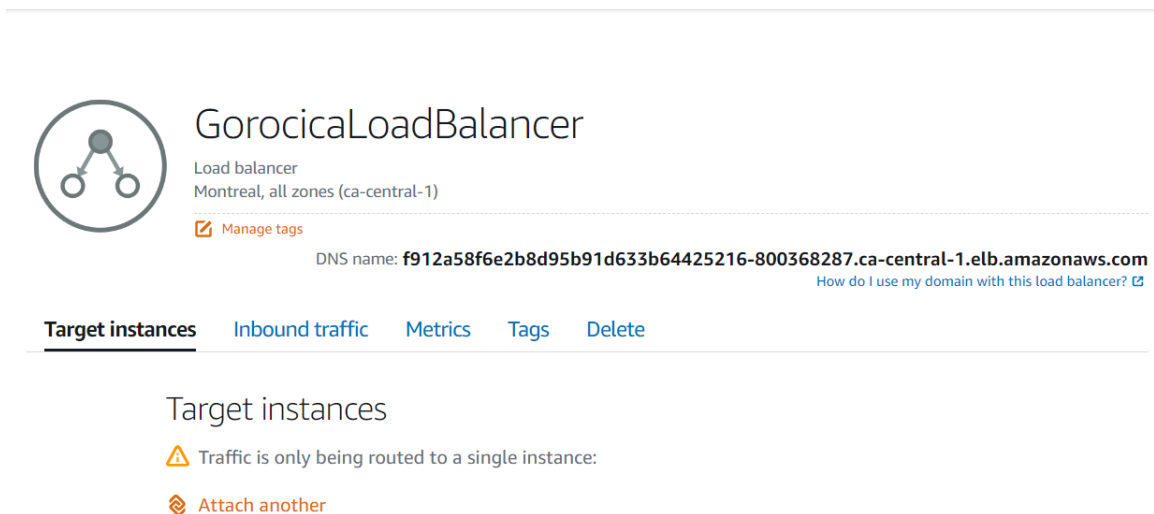
Creación de balanceador de carga

Para uso de la redundancia y el buen flujo de nuestra web, vamos a crear un balanceador de carga que nos permita que nuestra web no se sature. Para ello

nos ubicamos en el mismo apartado de Networking ubicado en nuestra pantalla principal de LightSail:



Y nos vamos a Create load balancer, le asignamos un nombre y la región en la que se encuentre nuestra web, que en este caso es Montreal, y la siguiente página que nos muestre será la configuración del balanceador de carga.






Asignación de instancias a balanceador de carga

En la pantalla de configuración del balanceador de cargas, podremos asignar instancias a las que redirigirá, ahí seleccionamos nuestra instancia de web y la segunda instancia, para que el balanceador se encargue de redirigir a la que considere apropiada sin problemas de saturación a alguna de las dos.

Target instances

Traffic will be evenly distributed to the following instances:



 **Attach another**
All available instances in **Montreal** attached

**webGorocica**Detach 

512 MB RAM, 1 vCPU, 20 GB SSD
Ubuntu

Health Check: **Passed**

Finalmente regresamos a nuestro subdominio para cambiar su dirección de nuestro webGorocica para el balanceador de carga y todo esté correctamente configurado:



A record  

Associate your domain or a subdomain with an IP address.

Subdomain

gorocica .idts.com.mx

Resolves to

 GorocicaLoadBalancer 

Creación de certificado SSL

Como último paso en la seguridad, se debe contar con un certificado SSL en nuestro balanceador de carga. Para ello accedemos una vez más a la

configuración de nuestro balanceador de carga y nos dirigimos a la sección de Inbound Traffic y en el apartado de Certificates accedemos a Create certificate:

Certificates

You may create and store up to two SSL/TLS certificates per load balancer to choose from

Create certificate 

Escribimos el dominio principal para crear el certificado:



Create a certificate

You must associate SSL/TLS certificates with a domain name to encrypt connections.

PRIMARY DOMAIN

Domain names must be a series of labels separated by '.'

CERTIFICATE NAME

Your Lightsail resources must have unique names.

ALTERNATE DOMAINS AND SUBDOMAINS

You can add up to 9 optional domains and subdomains for this certificate.

[Understanding SSL/TLS certificates](#) 

Cancel  Create 

Seguido de ello tendremos que crear registros CNAME para hacer válido el certificado desde la configuración del dominio de la misma forma que creamos un subdominio:

CNAME record



Create a subdomain alias of idts.com.mx and point it to another domain.

Subdomain

Maps to

_74773f71f... .idts.com.mx

_65bc1173acd59b080da91acaa42...

CNAME record



Create a subdomain alias of idts.com.mx and point it to another domain.

Subdomain

Maps to

_8fab575f9... .idts.com.mx

_1c6e29ad2e7e370768ff78f6ece7...

Si todo salió bien, el protocolo HTTPS nos debería dejar seleccionar el certificado creado para que quede en uso:

Protocols

Lightsail load balancers can handle both regular and encrypted web traffic.



HTTP

General web traffic, port 80

Enabled

By default, port 80 always accepts HTTP connections.



HTTPS

Secured web traffic, port 443

Enabled

Select an SSL/TLS certificate to enable HTTPS:

idts-com-mx



Resumen

Con esta práctica pudimos observar lo sencillo que es utilizar las herramientas modernas para el desarrollo seguro de aplicaciones web así como su mantenimiento, basándonos en el diagramado realizado con anterioridad. De no

haber sido por el diagrama o por las herramientas usadas, el proceso hubiera sido más complejo, más tardado y más propenso a fallos.