**Universidad Mayor Real y Pontificia de**

**San Francisco Xavier de Chuquisaca**

**Facultad de Tecnología**

****

**Proyecto Final SIS313 - INFRAESTRUCTURA, PLATAFORMAS TECNOLÓGICAS Y REDES**

**Universitarios:   
Quispe Sucullani José David (Ing. Ciencias de la computación)**

**Fecha de Presentación: 26/06/2025**

Sucre – Bolivia

**INFRAESTRUCTURA DE LOS SERVIDORES**



**Primeros pasos para la configuración de un entorno Node.js para servidores**

# Paso 1: Descargar e instalar NVM (Node Version Manager)

# NVM permite gestionar múltiples versiones de Node.js fácilmente en un mismo sistema.

curl -o- https://raw.githubusercontent.com/nvm-sh/nvm/v0.40.2/install.sh | bash

# Paso 2: Cargar NVM en la sesión actual

# Este comando hace que los comandos de nvm estén disponibles inmediatamente, sin necesidad de reiniciar la terminal.

. "$HOME/.nvm/nvm.sh"

# Paso 3: Instalar la última versión LTS (Long Term Support) de Node.js

# La versión LTS es la recomendada para entornos de producción por su estabilidad y soporte prolongado.

# Puedes verificar las versiones LTS disponibles con `nvm ls-remote --lts` si necesitas una versión específica.

nvm install --lts

# Paso 4: Verificar la instalación de Node.js

# Este comando mostrará la versión de Node.js que ha sido instalada y está activa.

# Por ejemplo: v22.2.0 (dependiendo de la versión LTS actual).

node -v

# Paso 5: Verificar la versión actual seleccionada de Node.js con NVM

# Confirma que la versión que NVM está utilizando coincide con la que acabas de instalar.

nvm current

# Paso 6: Verificar la instalación de npm (el gestor de paquetes de Node.js)

# npm se instala automáticamente con Node.js y es esencial para instalar dependencias de proyectos.

# Debería mostrar la versión de npm, por ejemplo: 10.x.x.

npm -vprimer paso para el balanceo entre servidores

# Paso 1: Descargar e instalar NVM (Node Version Manager)

# Esto permite gestionar múltiples versiones de Node.js fácilmente

curl -o- https://raw.githubusercontent.com/nvm-sh/nvm/v0.40.2/install.sh | bash

# Paso 2: Cargar NVM en la sesión actual (sin reiniciar la terminal)

# Esto hace que los comandos de nvm estén disponibles de inmediato

. "$HOME/.nvm/nvm.sh"

# Paso 3: Instalar la última versión LTS de Node.js

# Puedes verificar la versión exacta con `nvm ls-remote --lts` si deseas una específica

nvm install --lts

# Paso 4: Verificar la instalación de Node.js

# Debería mostrar algo como: v22.2.0 (dependiendo de la versión LTS actual)

node -v

# Paso 5: Verificar la versión actual seleccionada de Node.js

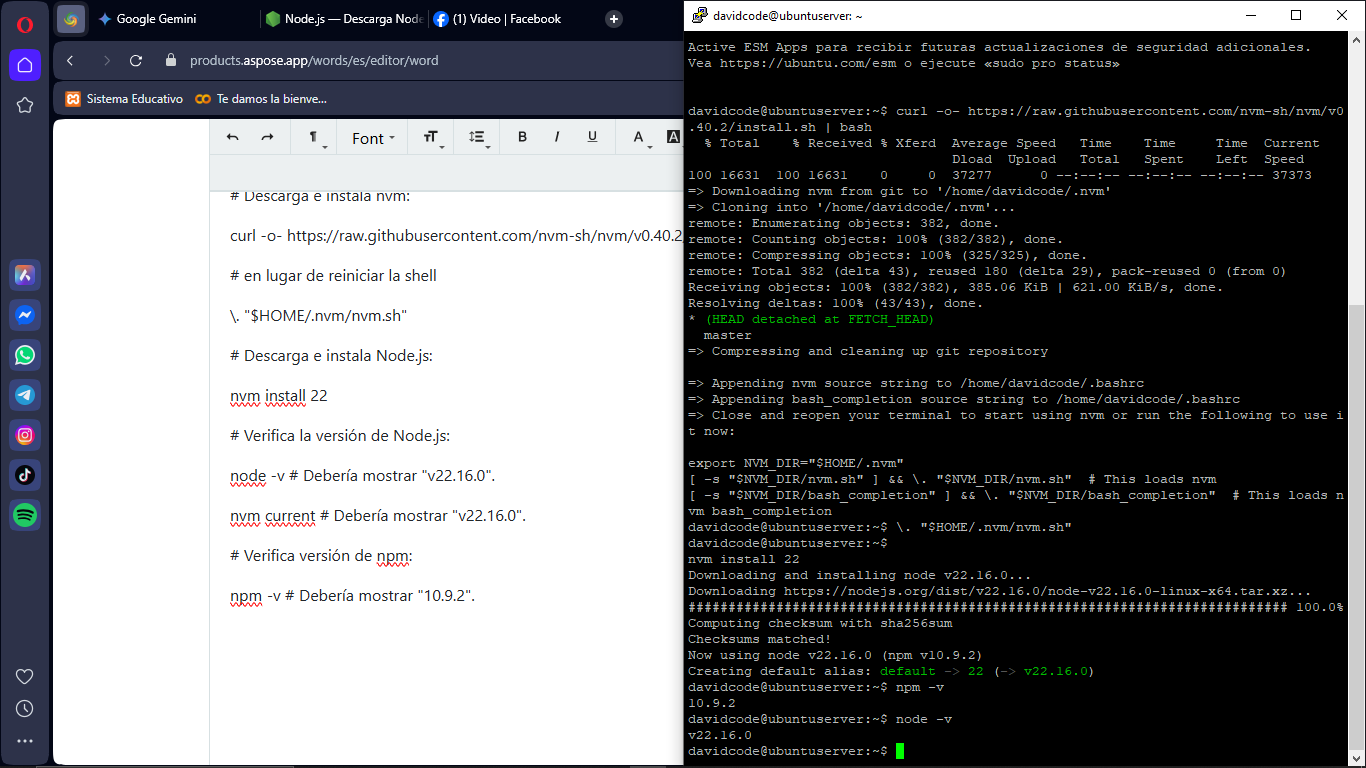
# Debería coincidir con la versión instalada

nvm current

# Paso 6: Verificar la instalación de npm (el gestor de paquetes de Node.js)

# Debería mostrar algo como: 10.x.x

npm -v



# Pasos iniciales para configurar un proyecto Node.js con Express.js

# Paso 1: Crear una nueva carpeta para tu aplicación

# Este comando crea un directorio llamado 'apps' donde residirá tu proyecto.

mkdir apps

# Paso 2: Entrar en la carpeta del proyecto

# Navega al directorio recién creado para trabajar dentro de él.

cd apps

# Paso 3: Inicializar el archivo package.json

# El archivo package.json gestiona las dependencias del proyecto y metadatos.

# Te solicitará información sobre el proyecto (nombre, versión, descripción, etc.).

# Si deseas aceptar los valores por defecto sin preguntas, puedes usar: npm init -y

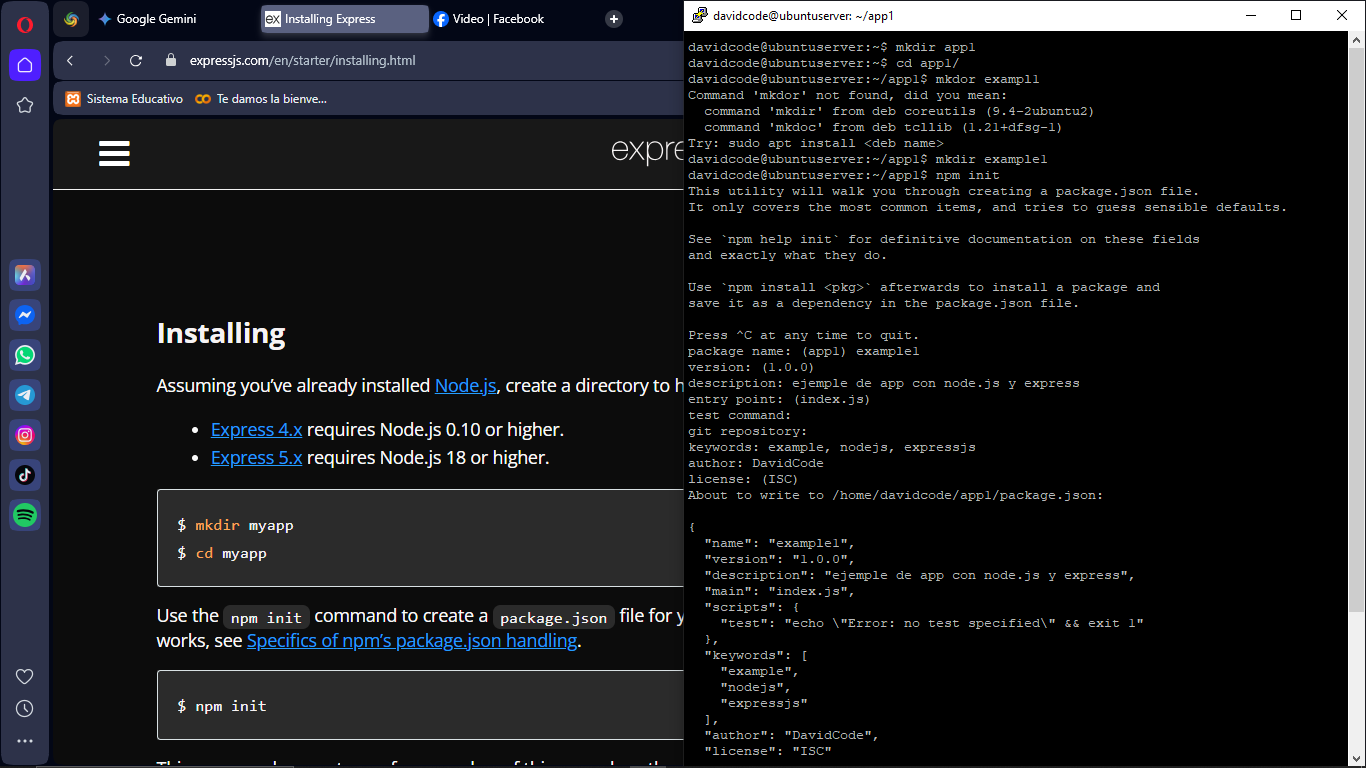
npm init

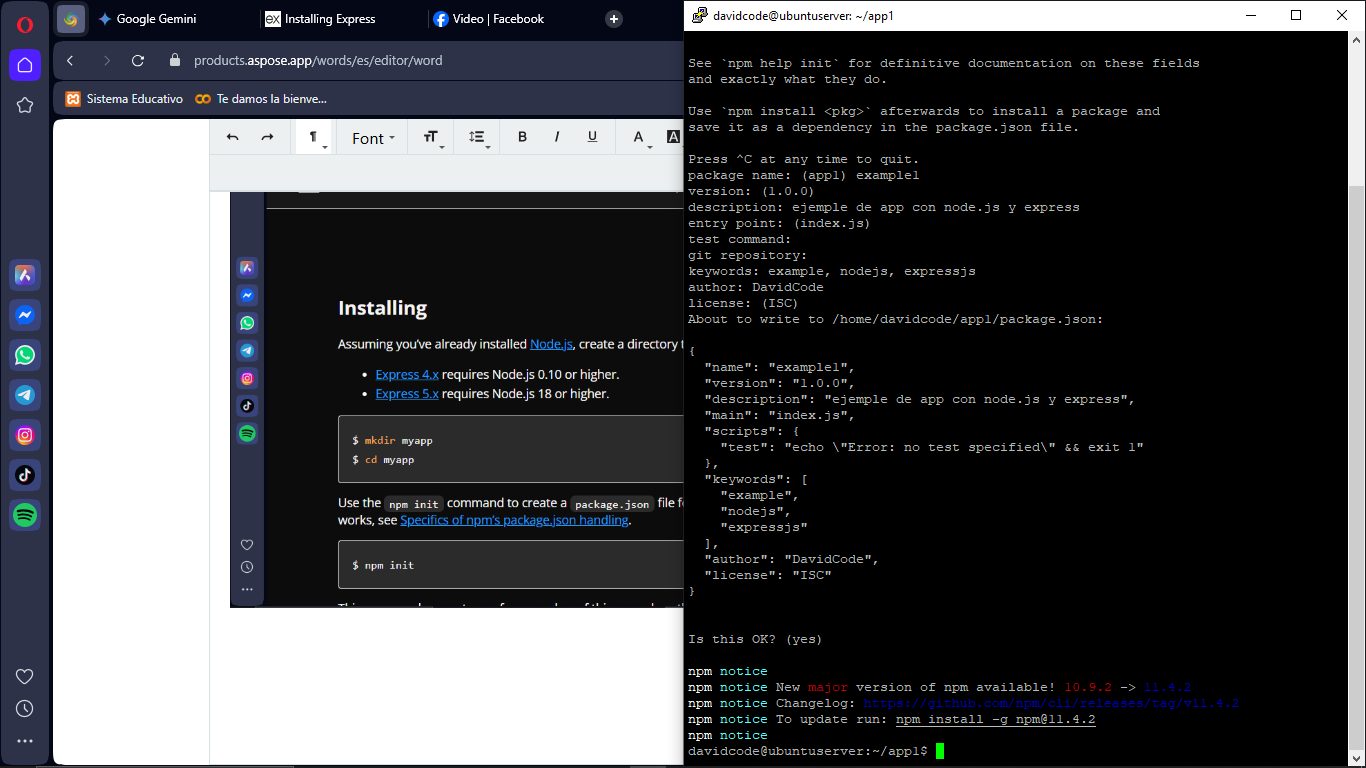
# Paso 4: Instalar Express.js como dependencia del proyecto

# Express.js es un framework de aplicación web minimalista y flexible para Node.js,

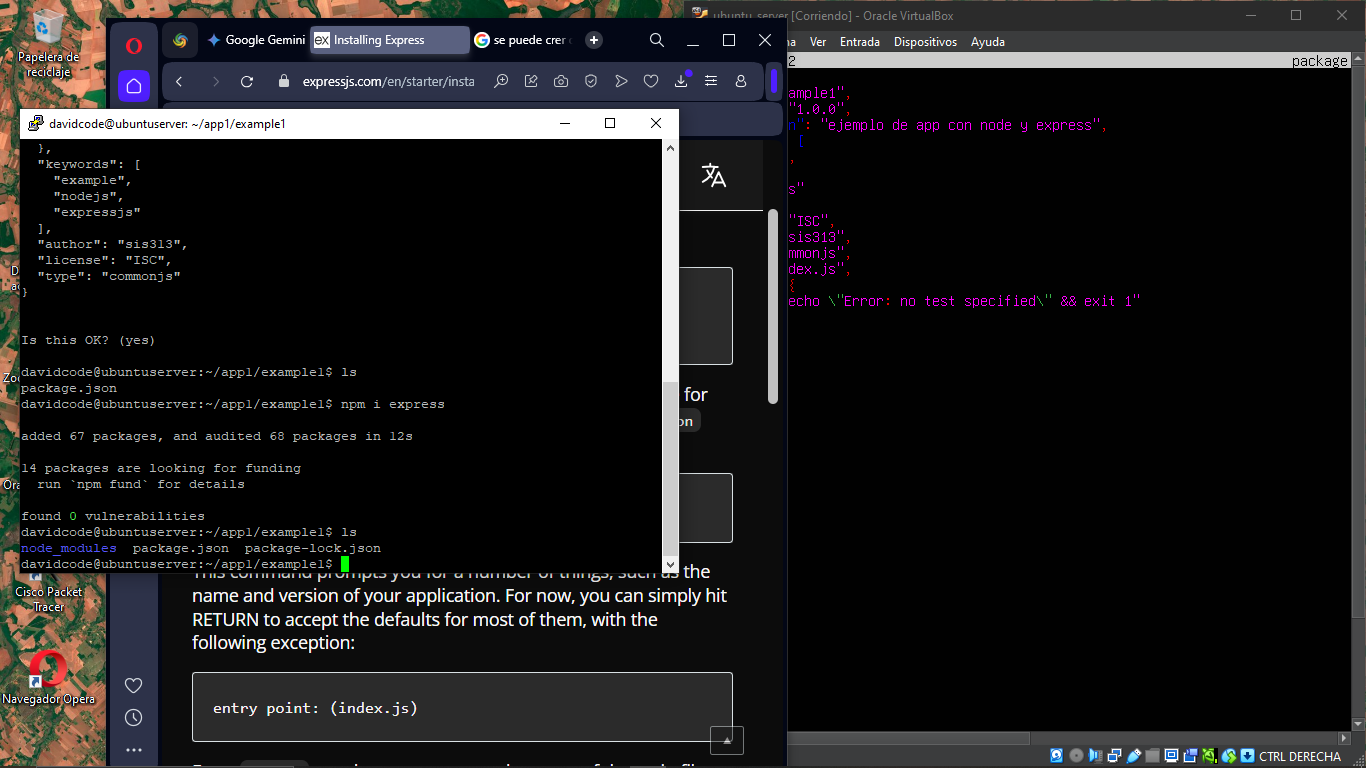
# que facilita la construcción de APIs y aplicaciones web.

npm install express

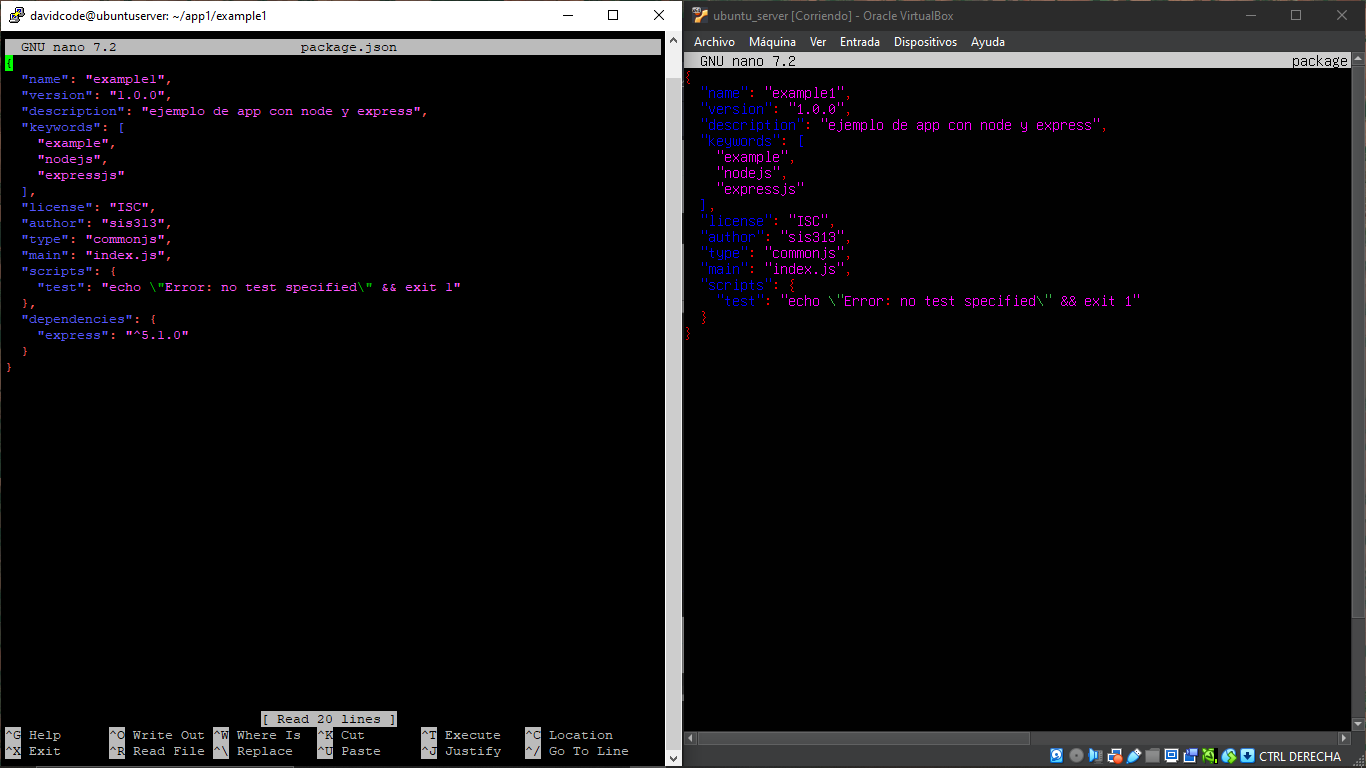




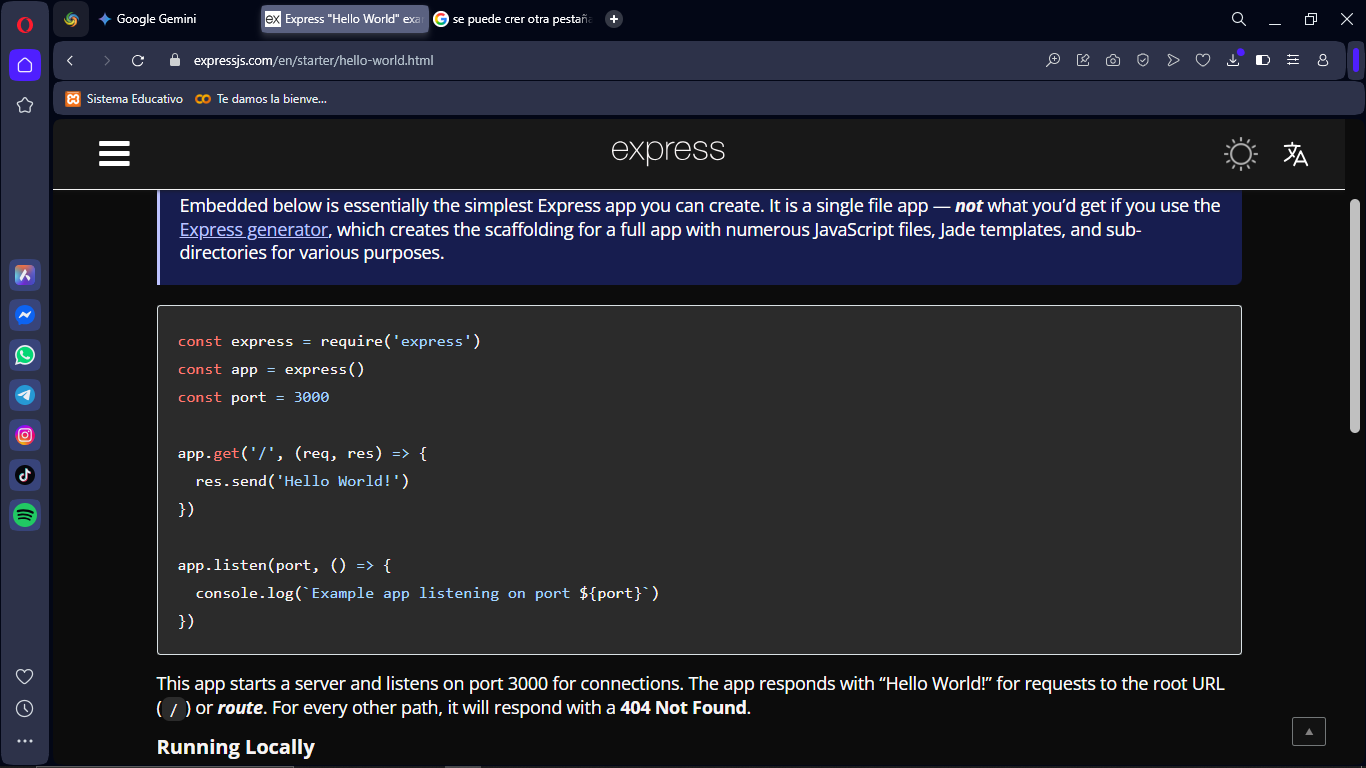
instalacion de express por ahora en un solo terminal o un solo servidor

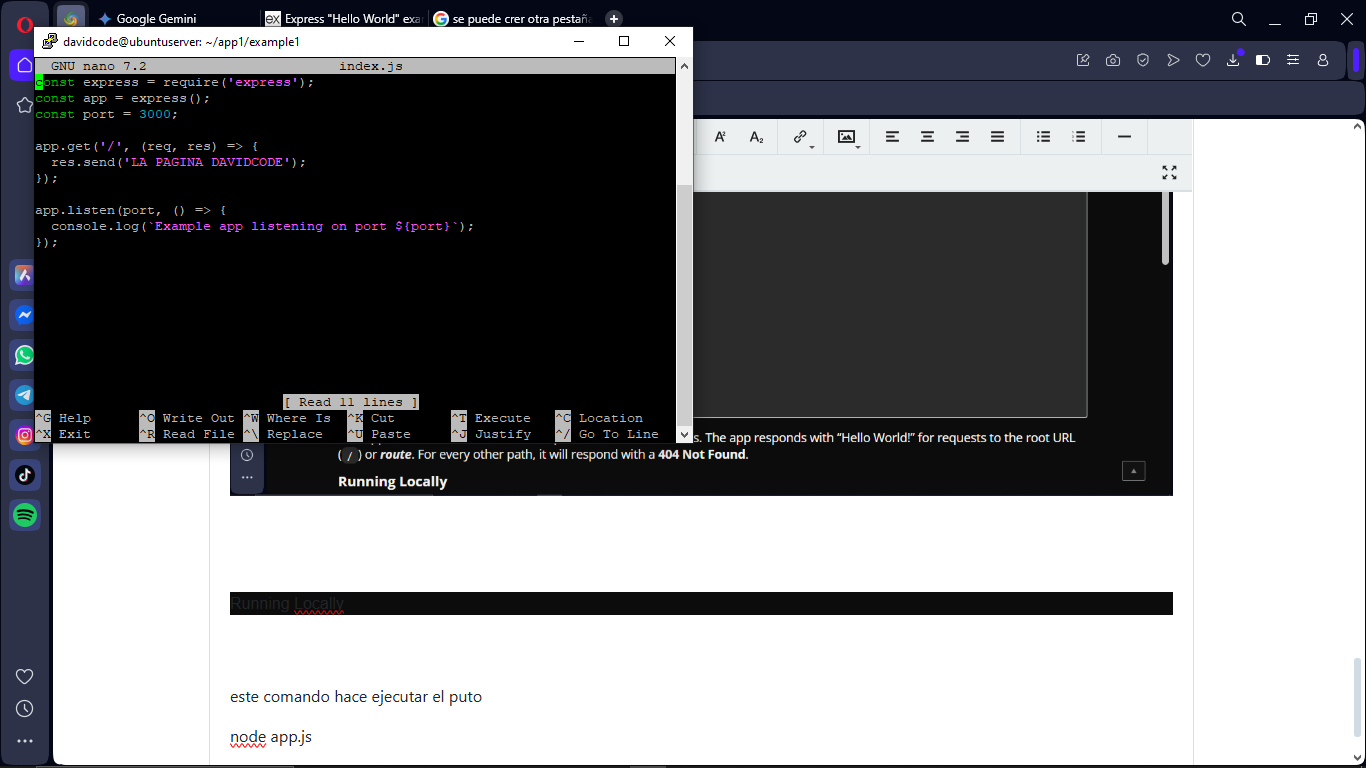


verificacion de expres sin dependencias y con dependencias antes y despues



se copia el codigo para que funcuione la pagina con el puerto 3000  con javascript y diga el mensaje por defecto





Running Locally

# Comandos para ejecutar una aplicación Node.js y gestionar puertos

# Ejecutar la aplicación Node.js

# Este comando inicia tu aplicación Node.js. Asegúrate de reemplazar 'app.js'

# con el nombre real de tu archivo principal de la aplicación.

node app.js

# Configuración del firewall (UFW) para permitir tráfico en un puerto específico

# Esto es crucial para que tu aplicación sea accesible desde fuera del servidor.

# Permitir conexiones TCP en el puerto 3000

# Puedes cambiar '3000' por el número de puerto que tu aplicación esté utilizando.

sudo ufw allow 3000/tcp

# Recargar el firewall para aplicar los cambios

sudo ufw reload

# Copia de carpetas (útil para crear múltiples instancias de tu aplicación)

# Si deseas ejecutar tu aplicación en diferentes puertos o con configuraciones ligeramente distintas,

# puedes duplicar la carpeta de tu proyecto.

# Copiar el contenido de 'example1' a una nueva carpeta 'example2'

# Reemplaza 'example1' con el nombre de tu carpeta de aplicación original

# y 'example2' con el nombre de la nueva carpeta para la instancia duplicada.

cp -r example1 example2

# Pasos adicionales para aplicaciones duplicadas:

# 1. Navega a la nueva carpeta (ej. cd example2).

# 2. Edita el código de tu aplicación (ej. app.js) en la nueva carpeta para:

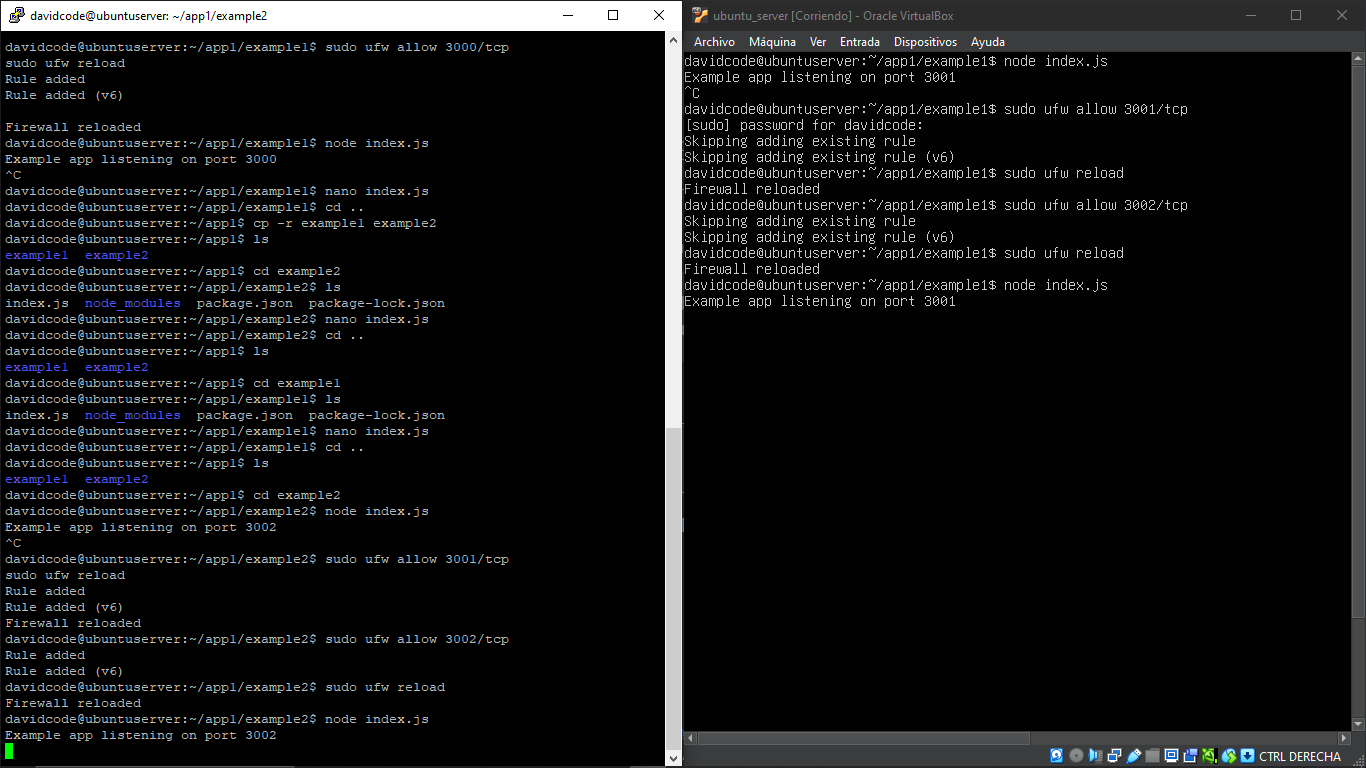
# - Cambiar el puerto en el que escucha la aplicación (ej. de 3000 a 3001).

# - Modificar cualquier mensaje de bienvenida o identificador para que puedas distinguir

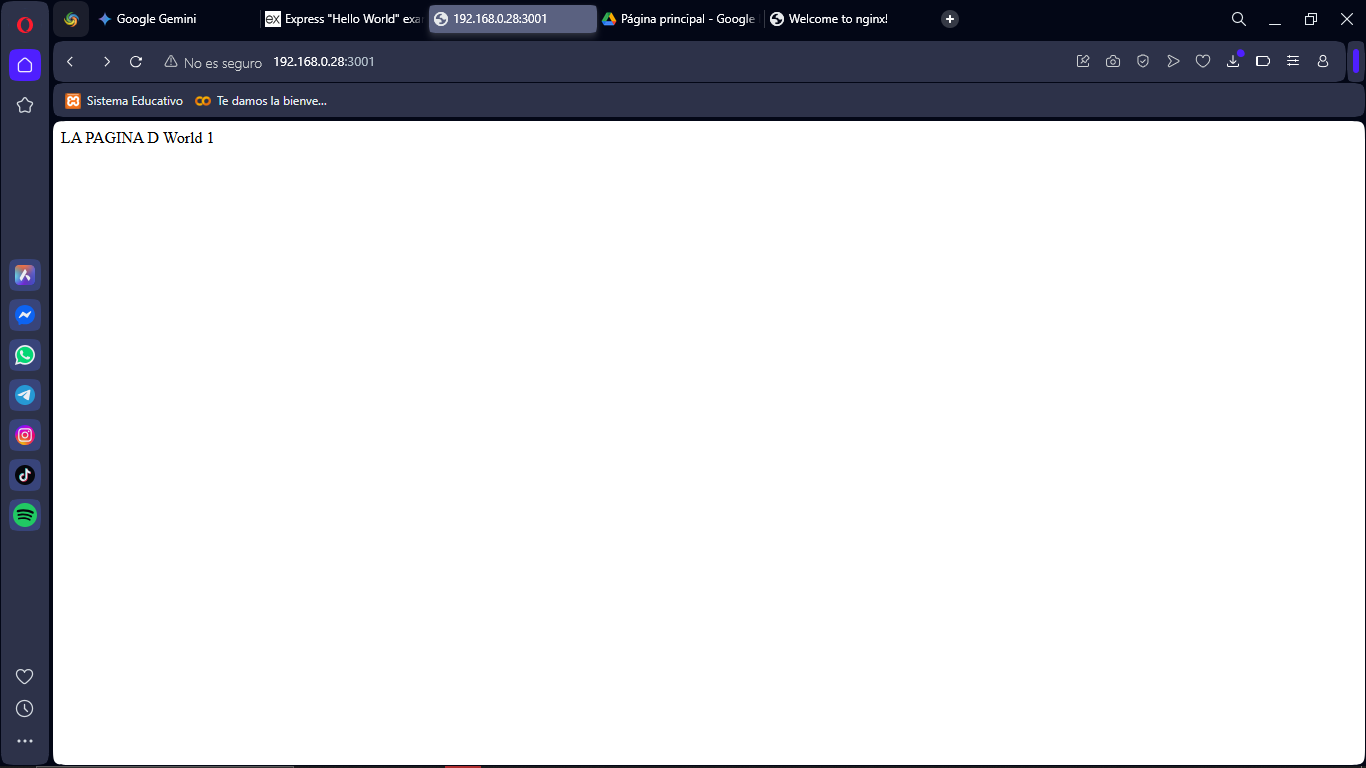
# fácilmente qué instancia de la aplicación está respondiendo.

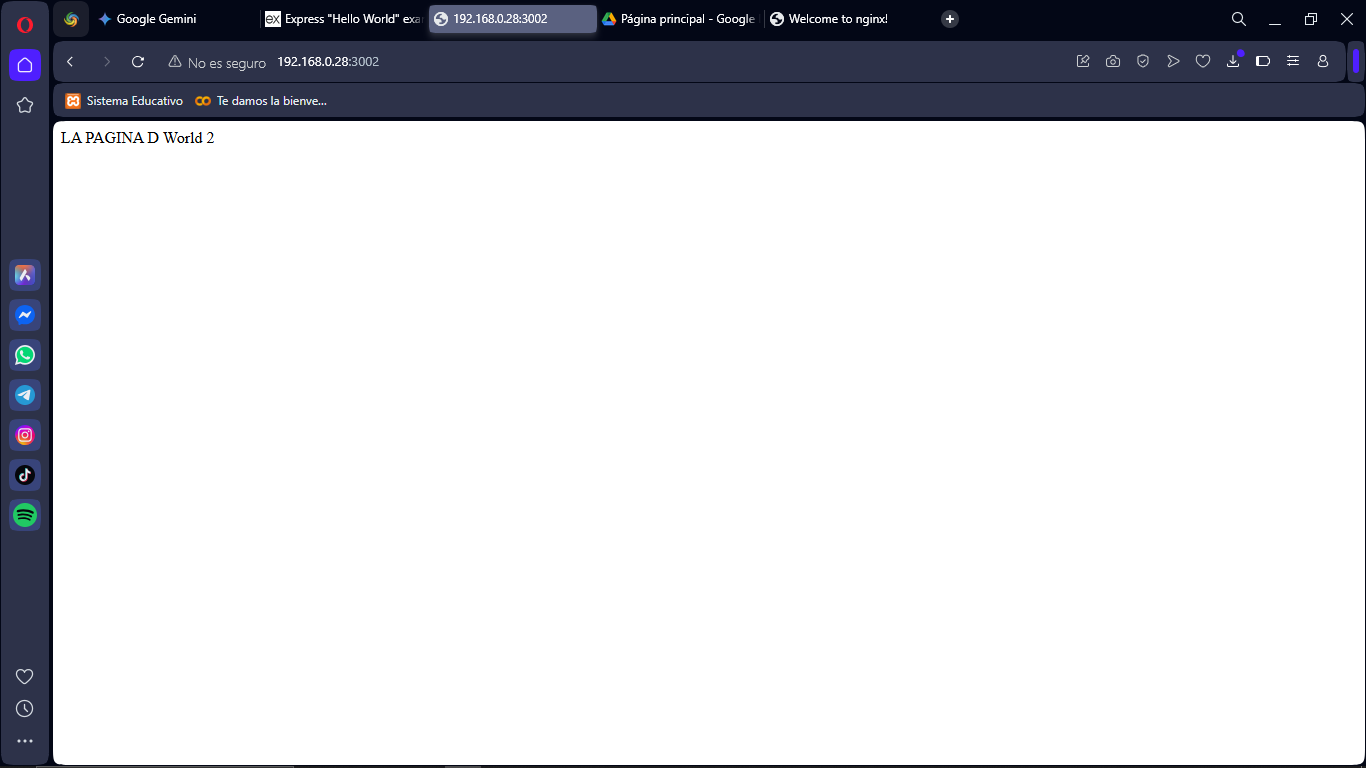
# 3. Abre el nuevo puerto en el firewall (ej. sudo ufw allow 3001/tcp).

# 4. Inicia la nueva instancia de la aplicación (ej. node app.js).



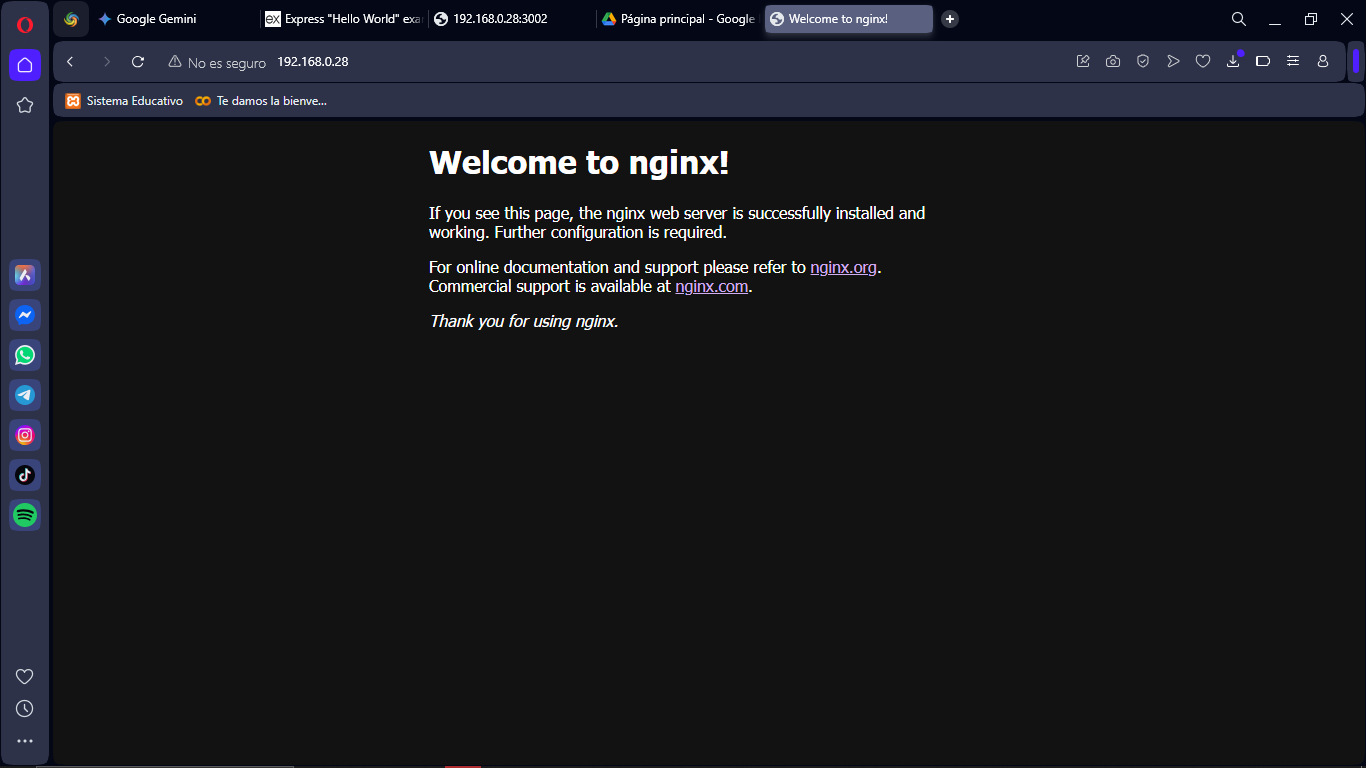
prueba de los creado





/\*INSTALACION DE GNINX PARA EL BALANCEO \*/

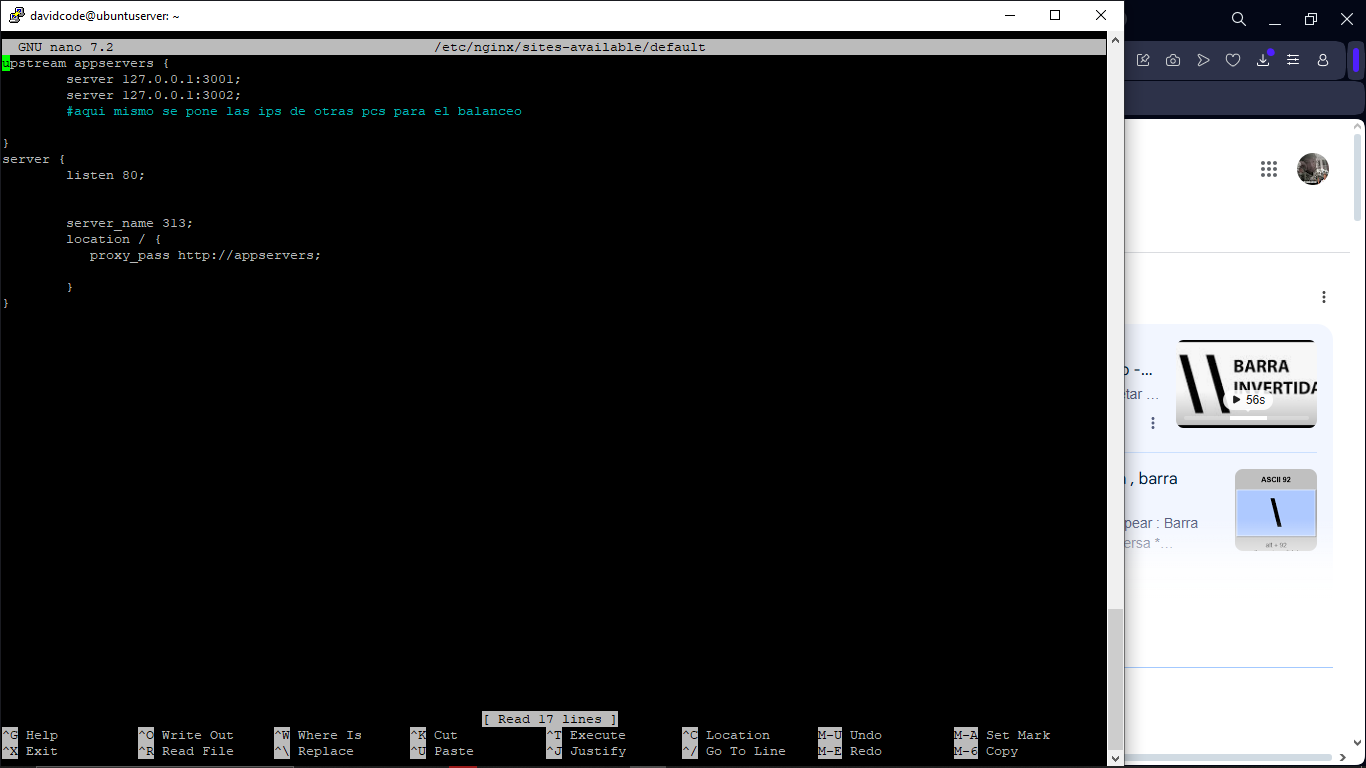
 =>sudo apt install nginx



/\*haces una copia de seguridad de la carpeta por defecto de nginx para editarlo despues \*/

sudo cp /etc/nginx/sites-available/default /etc/nginx/sites-available/313

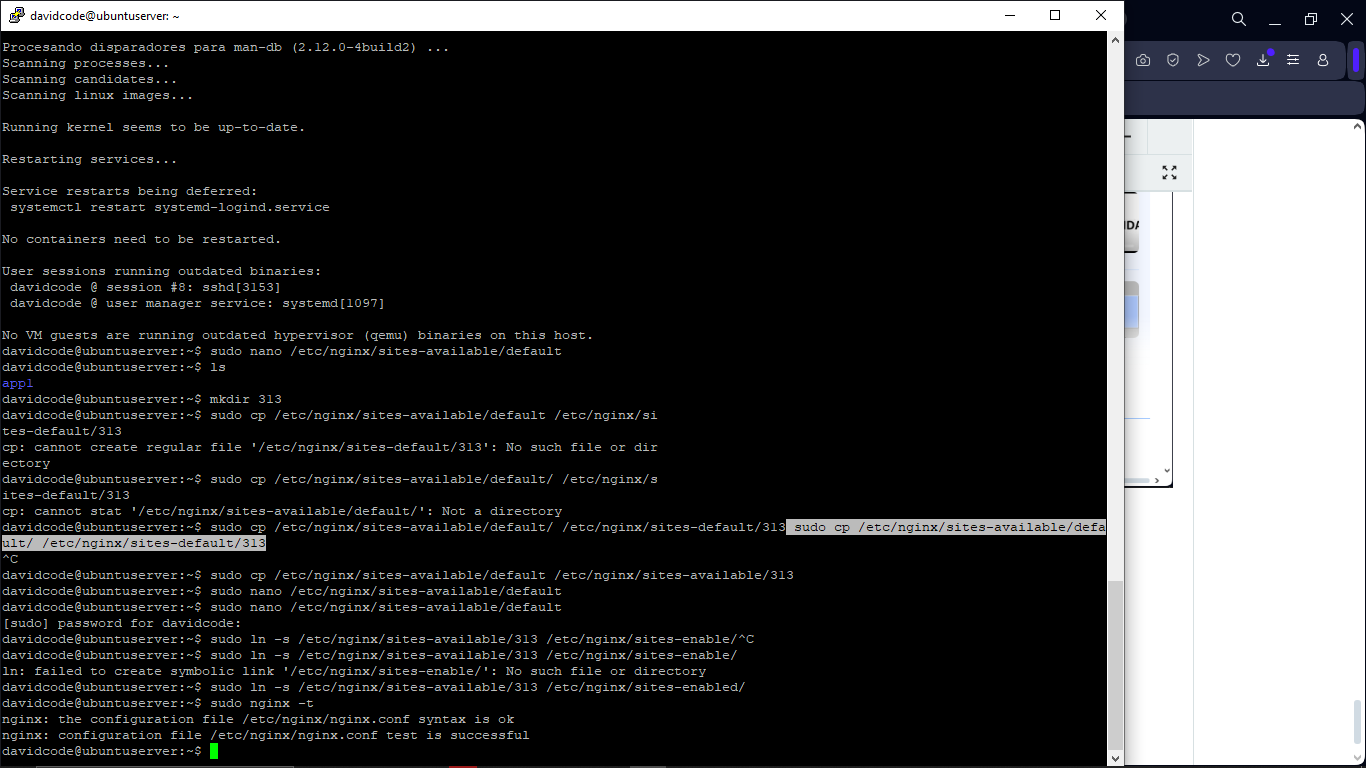
/\*se añade los servers para con las ips para q puedes tener el balanceo entre pcs cambiando las ips \*/



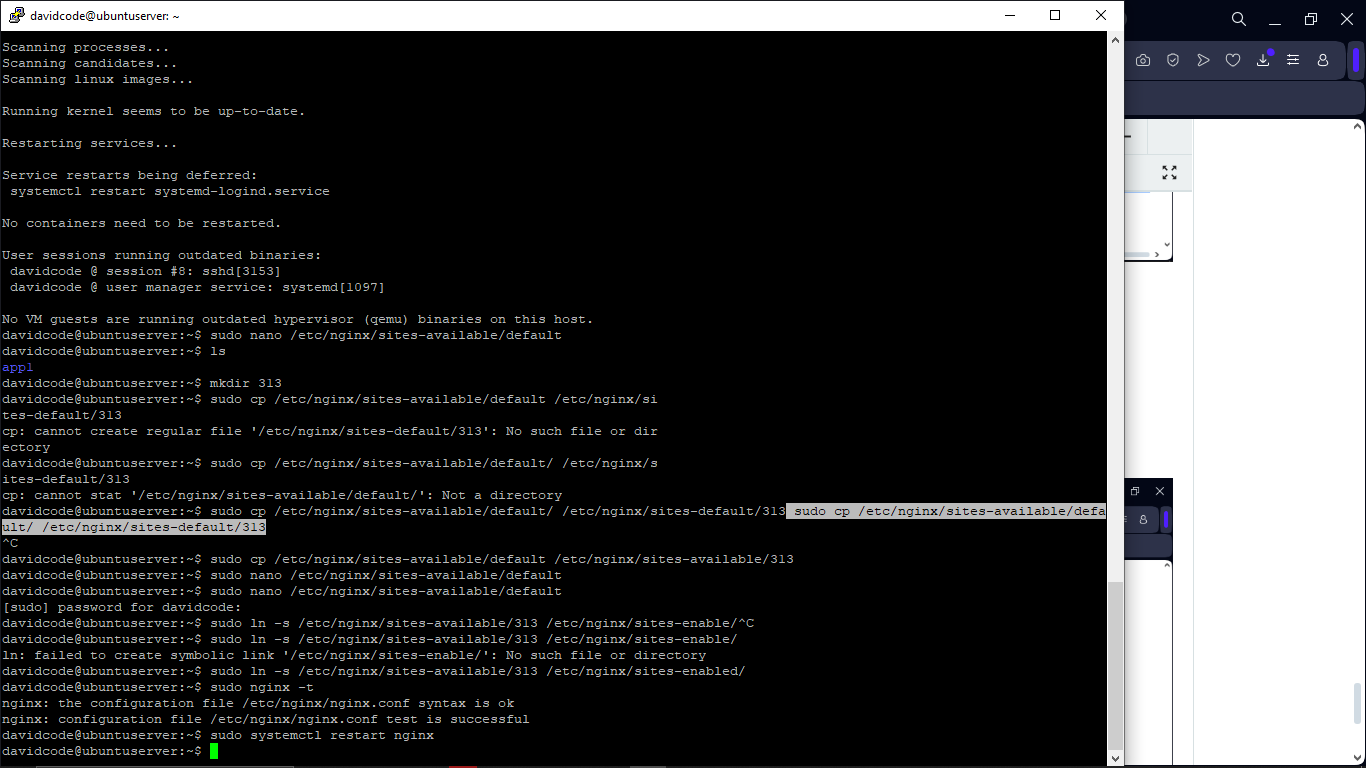
Habilitar y verificar la configuración de NGINX

=> sudo ln -s /etc/nginx/sites-available/313 /etc/nginx/sites-enabled/

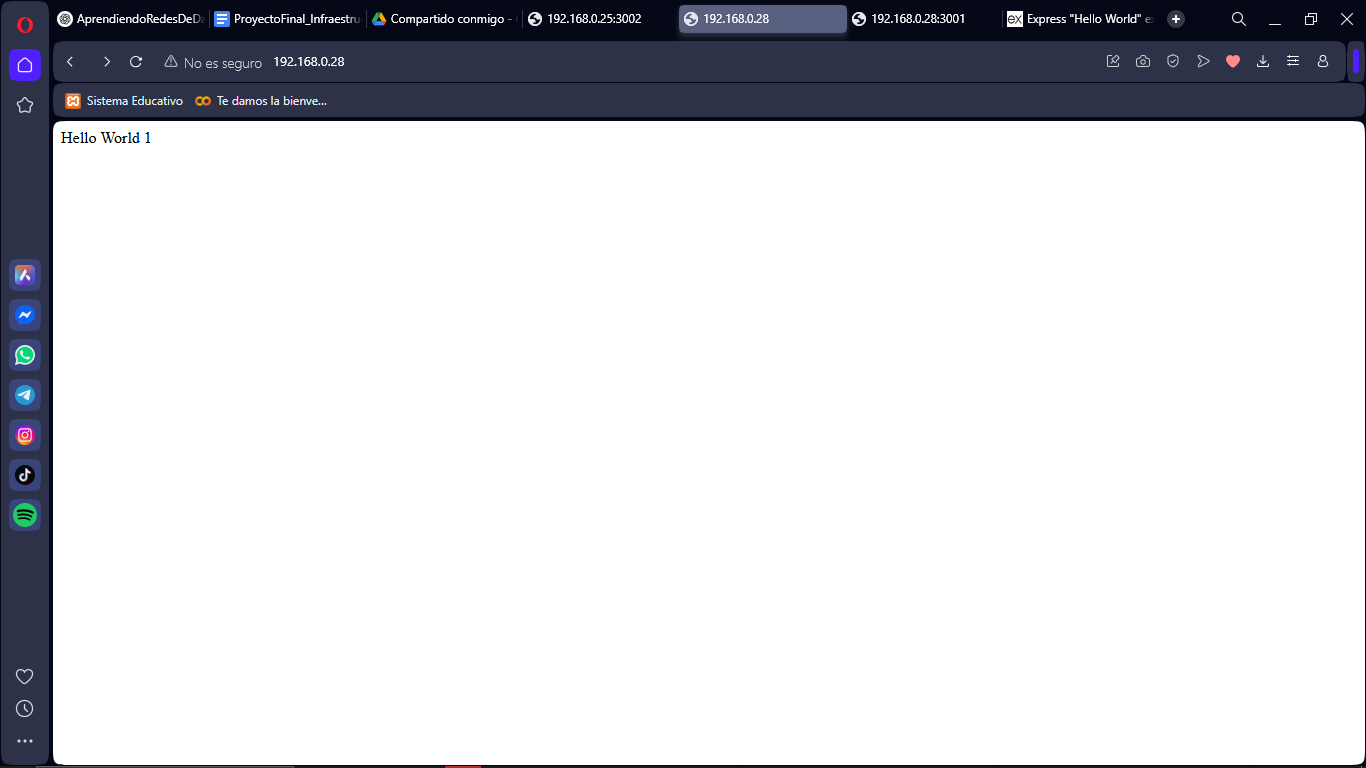
verificar si nginx esta funcionadno y la sintaxis esta correcto



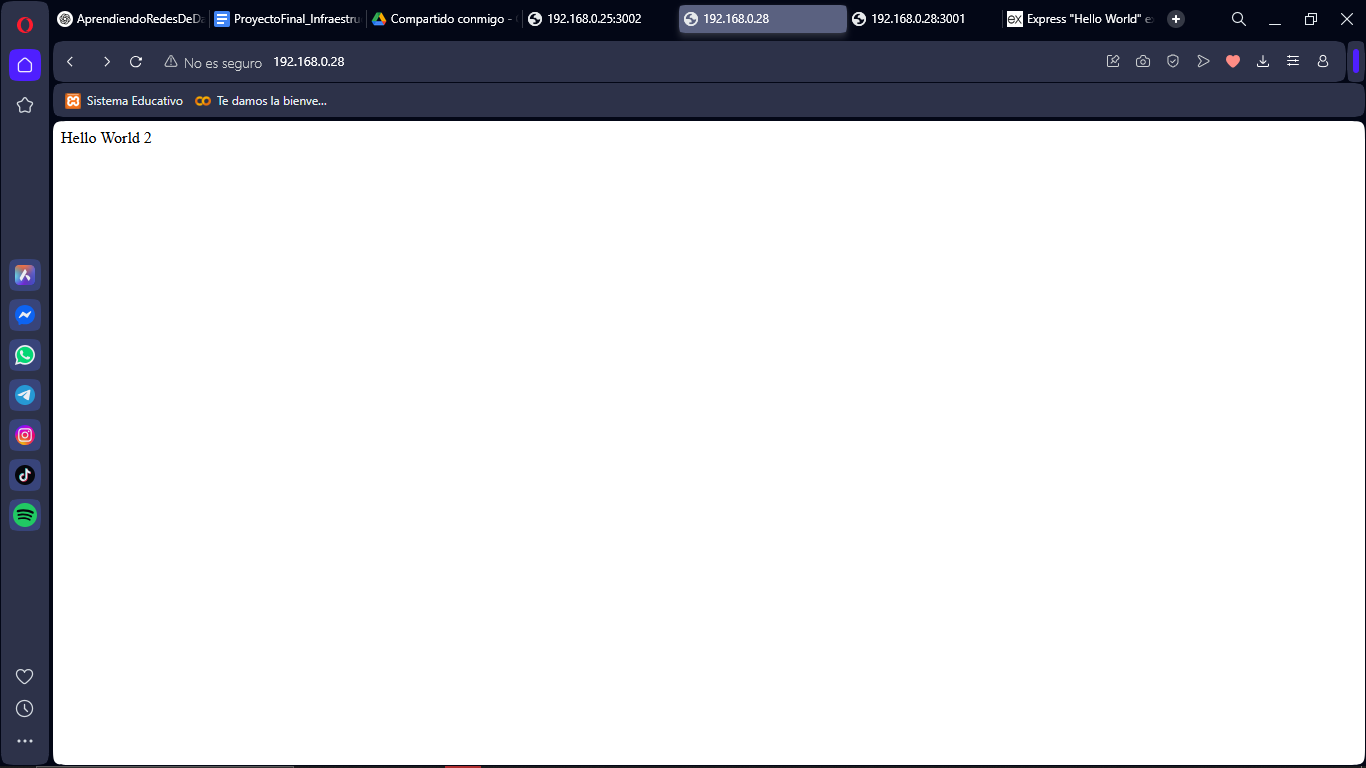
/\*siempre q no haiga errores en la sintaxix toda reiniciar nginx\*/



# Finalización del balanceo de carga entre servidores Se ha configurado y probado el balanceo de carga entre el Servidor 1 (IP: 192.168.0.28) y el Servidor 2 (IP: 192.168.0.25), utilizando el puerto 3001.



Puerto 3002



Servidores de Bases de Datos (2 servidores):

Configurar el Servidor Maestro (MASTER)

PASO 1: Instalar MariaDB en Maestro y Esclavo

En ambos servidores (maestro y esclavo), ejecuta:

=> sudo apt update

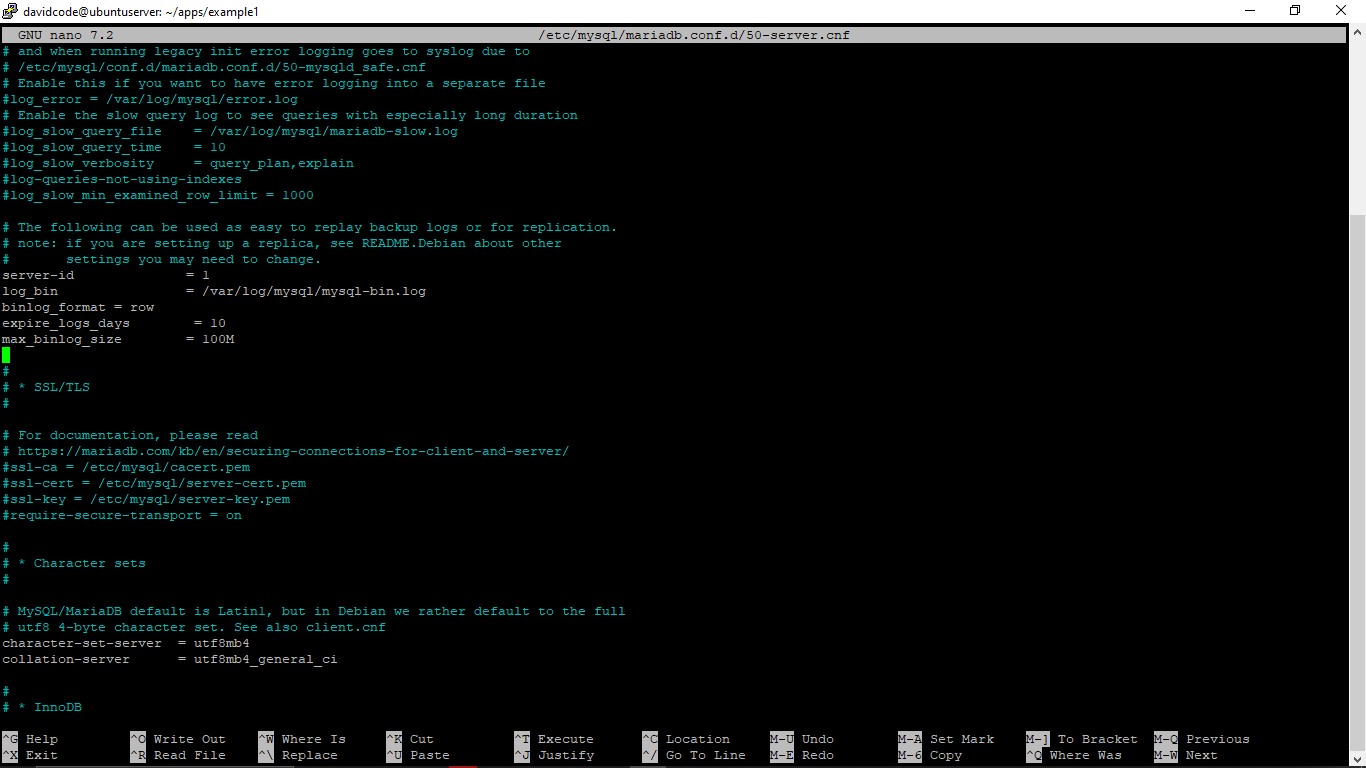
=> sudo apt install mariadb-server -y

Verifica que el servicio esté activo:

=> sudo systemctl status mariadb

PASO 2: Configuración del SERVIDOR MAESTRO

=> sudo nano /etc/mysql/mariadb.conf.d/50-server.cnf



# Configuración de Replicación de MariaDB: Servidor Maestro

# 1. Reiniciar MariaDB

sudo systemctl restart mariadb

# 2. Crear Usuario de Replicación en Maestro

# Acceder a la consola:

sudo mariadb -u root -p

# Dentro de la consola, ejecutar:

CREATE USER 'replicador'@'%' IDENTIFIED BY 'clave\_segura'; # Cambia 'clave\_segura'

GRANT REPLICATION SLAVE ON \*.\* TO 'replicador'@'%';

FLUSH PRIVILEGES;

# Bloquear tablas y obtener estado del log binario:

FLUSH TABLES WITH READ LOCK;

SHOW MASTER STATUS;

# ✅ Anota: File: mysql-bin.XXXXXX y Position: YYY

# ¡NO CERRAR esta sesión de la consola de MariaDB!

**# 3. Crear Dump y Enviarlo al Esclavo**

# Desde OTRA terminal en el Maestro:

sudo mysqldump -u root -p --all-databases > backup.sql



Envíalo al esclavo:

OJO TIENE Q SER EL USUARIO Y LA IP A LA CUAL SE TIENE Q ENVIAR SIN OLVIDAR EL ARCHIVO

=> scp dump.sql usuario@IP\_ESCLAVO:/home/usuario/

Después puedes desbloquear:

=> UNLOCK TABLES;

=> EXIT;

Configurar el SERVIDOR ESCLAVO

=> sudo nano /etc/mysql/mariadb.conf.d/50-server.cnf

Agrega/modifica:

ini

CopiarEditar

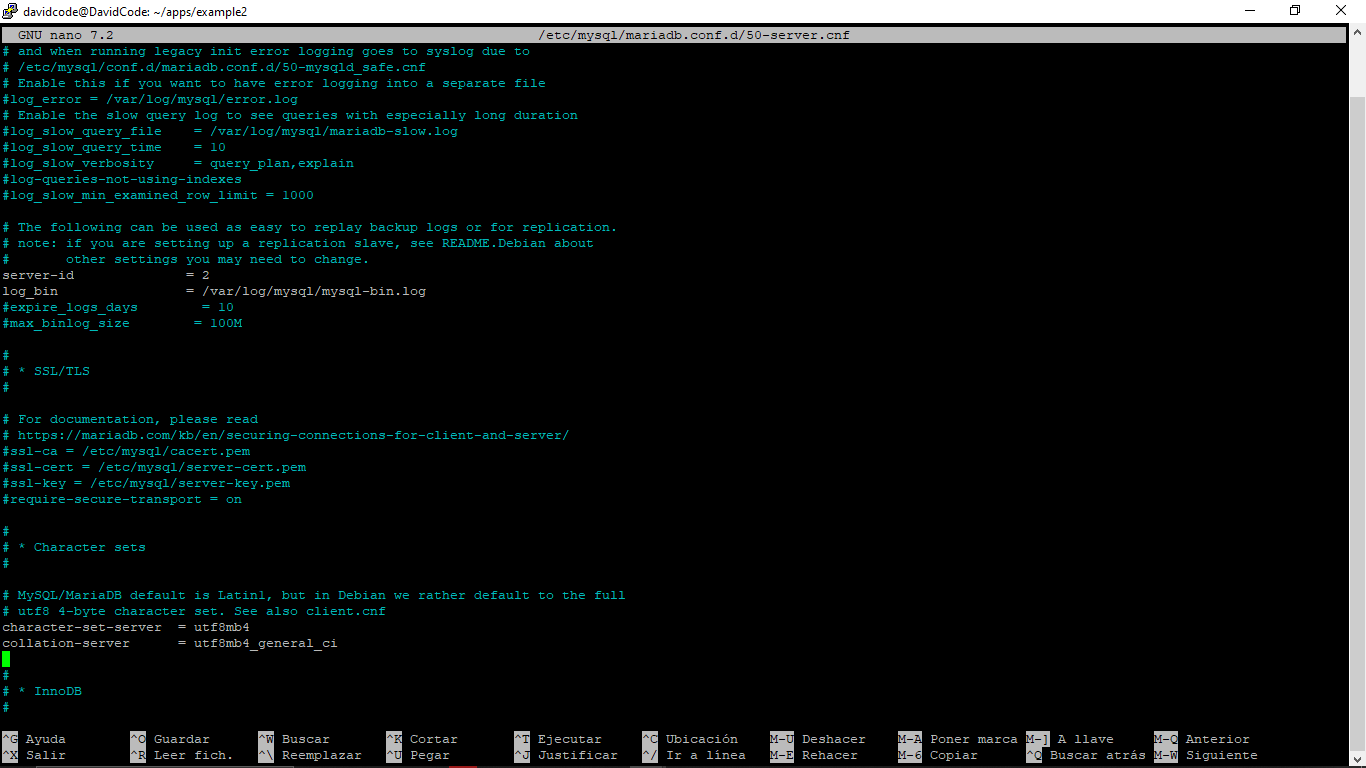
[mysqld]

server-id=2

relay-log=relay-log

log-bin=mysql-bin

read-only=1



Reinicia:

=> sudo systemctl restart mariadb

Conectar esclavo al maestro

Conectar el esclavo con el maestro para con la clave, el mysql-bin que se te dio el maestro y con el master\_log\_pos

=> sudo mariadb -u root -p

CHANGE MASTER TO

MASTER\_HOST='IP\_DEL\_MAESTRO',

MASTER\_USER='replicador',

MASTER\_PASSWORD='clave123',

MASTER\_LOG\_FILE='mysql-bin.000001',

MASTER\_LOG\_POS=XXX;

Iniciar la replicación

Se inicia en el mysql

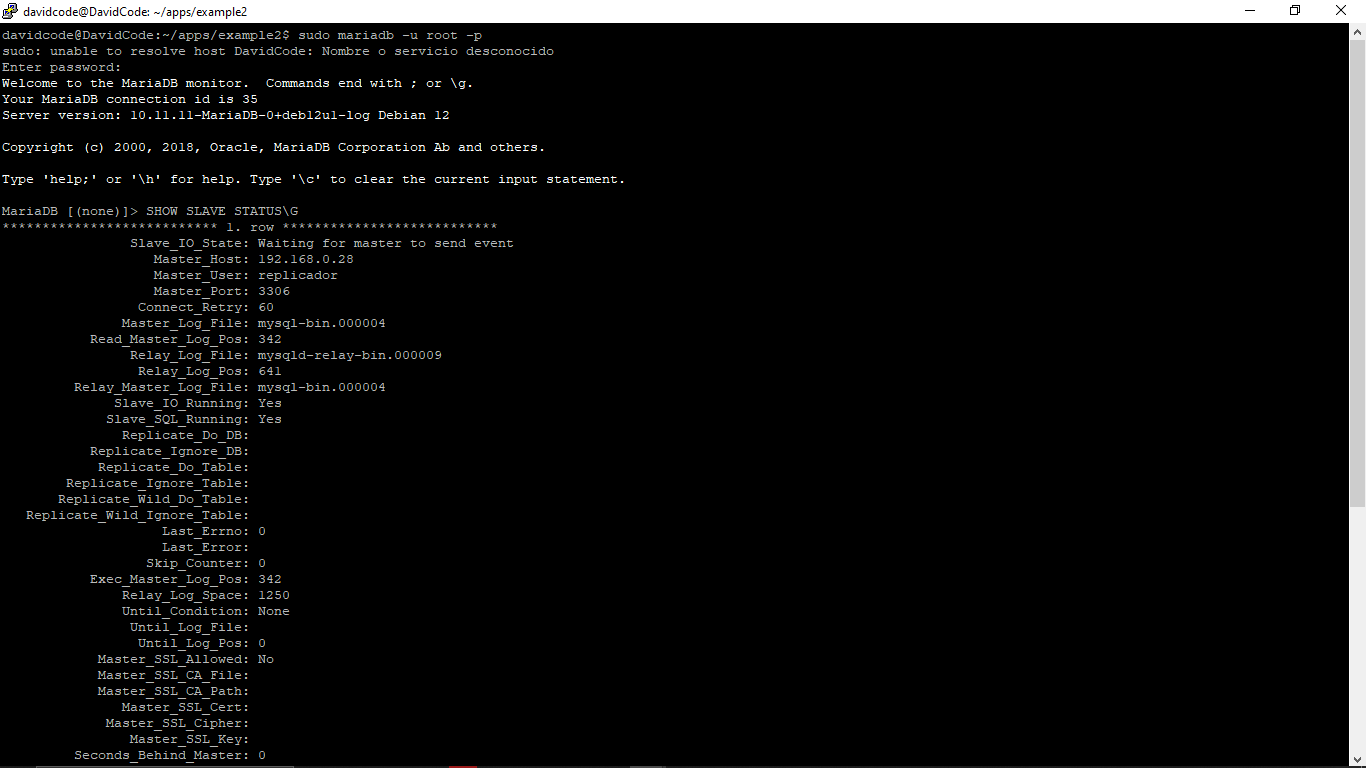
=> START SLAVE;

=> SHOW SLAVE STATUS\G

Verifica que esto diga:

Slave\_IO\_Running: Yes

Slave\_SQL\_Running: Yes



PRUEBA

En el maestro, agrega una tabla o dato nuevo.

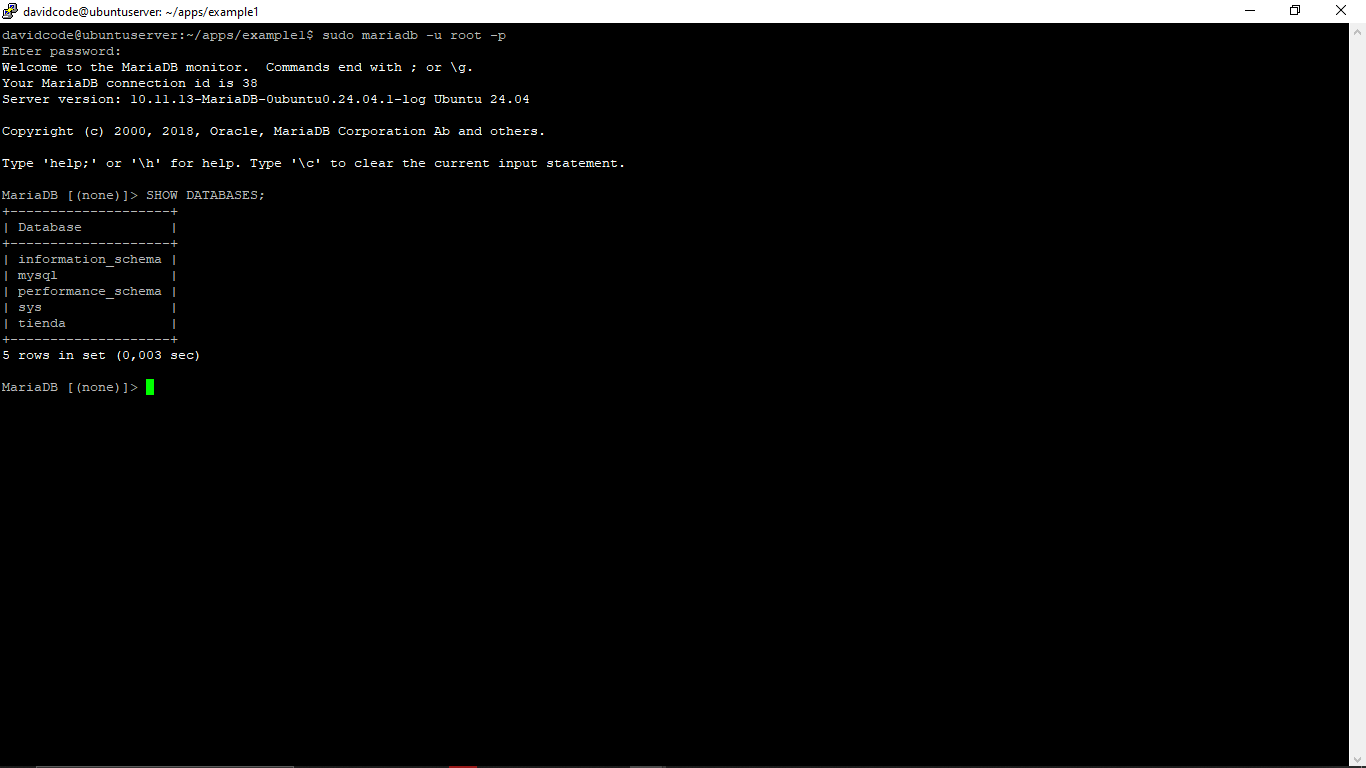
Ojo => te pedira la contraseña con la cual iniciaste la bd

=> Verificar replicación en el Esclavo

sudo mariadb -u root -p

Luego

=> SHOW DATABASES;



En el esclavo, verifica que aparezca automáticamente.

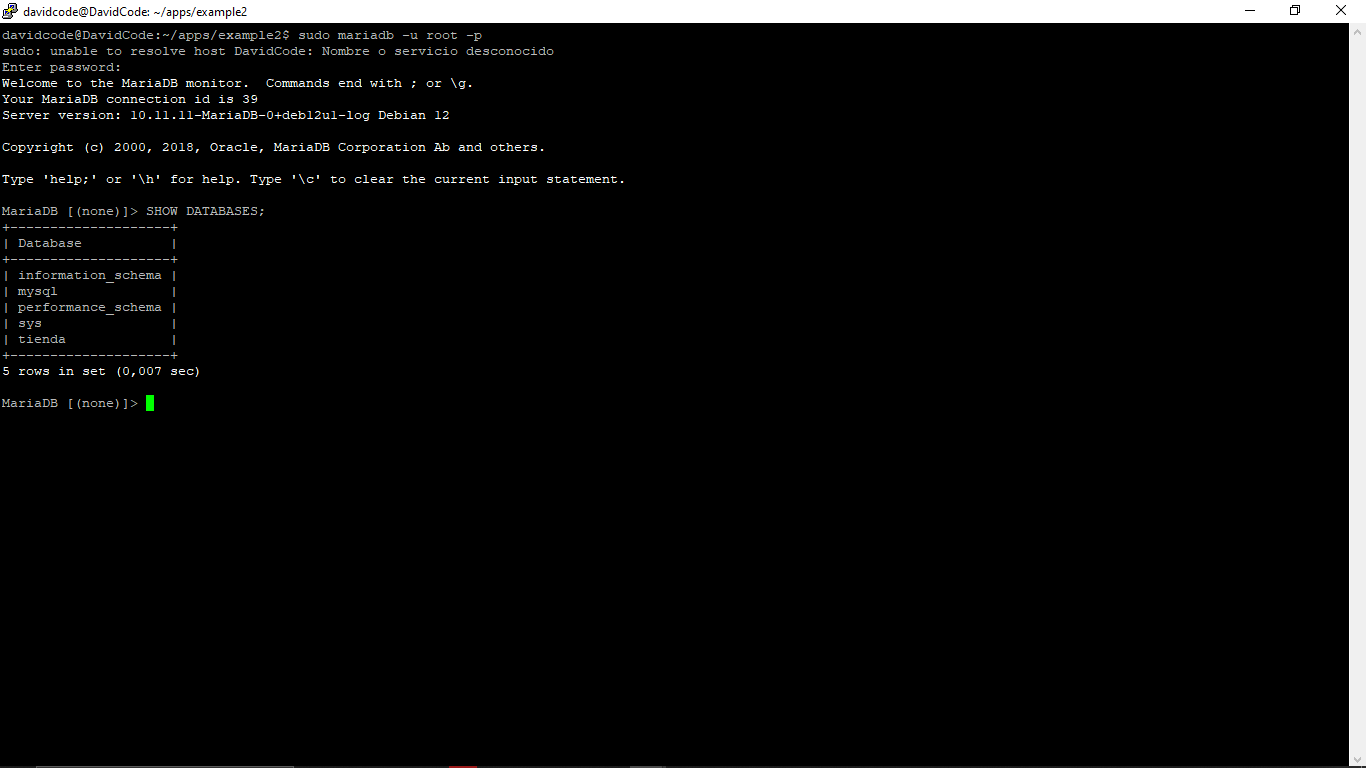
Ojo =>

=> Verificar replicación en el Esclavo

sudo mariadb -u root -p

Luego

=> SHOW DATABASES;

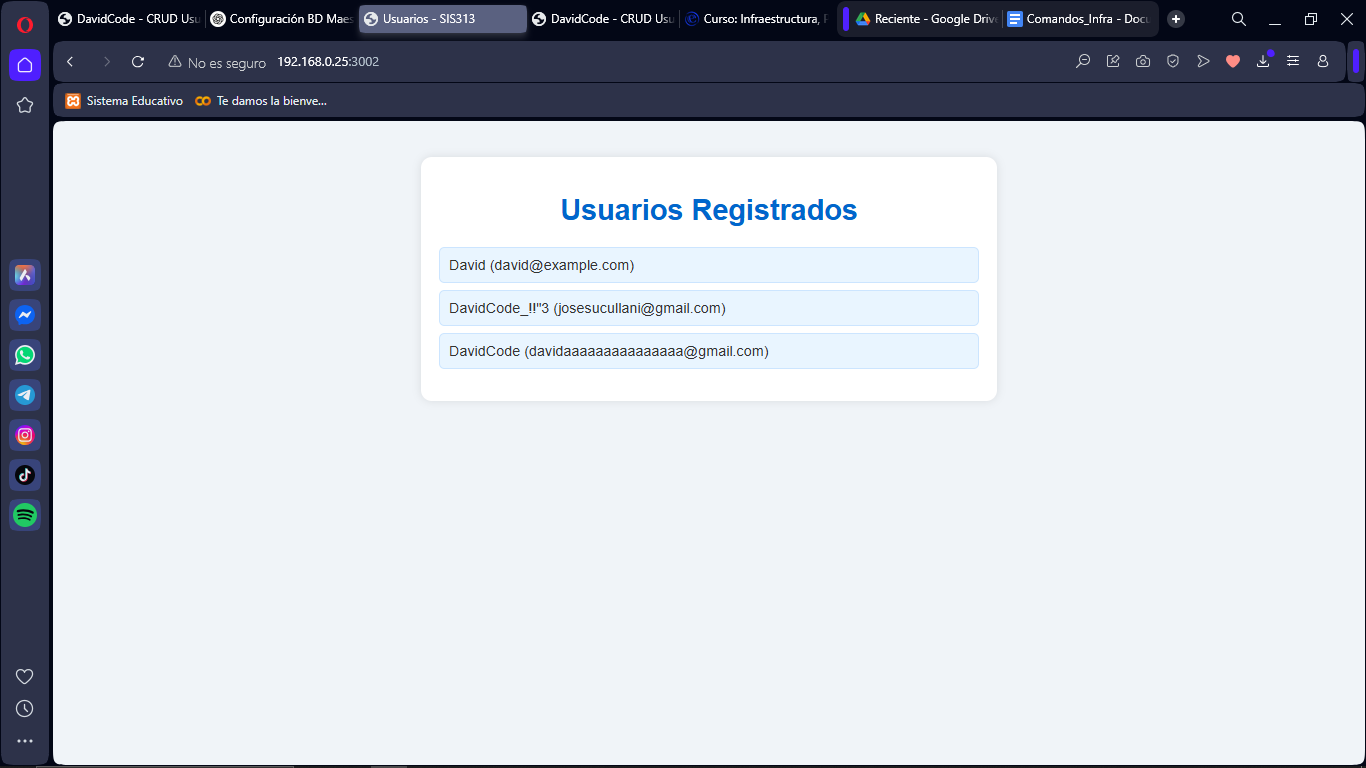


**Funcionamiento de los servidores**

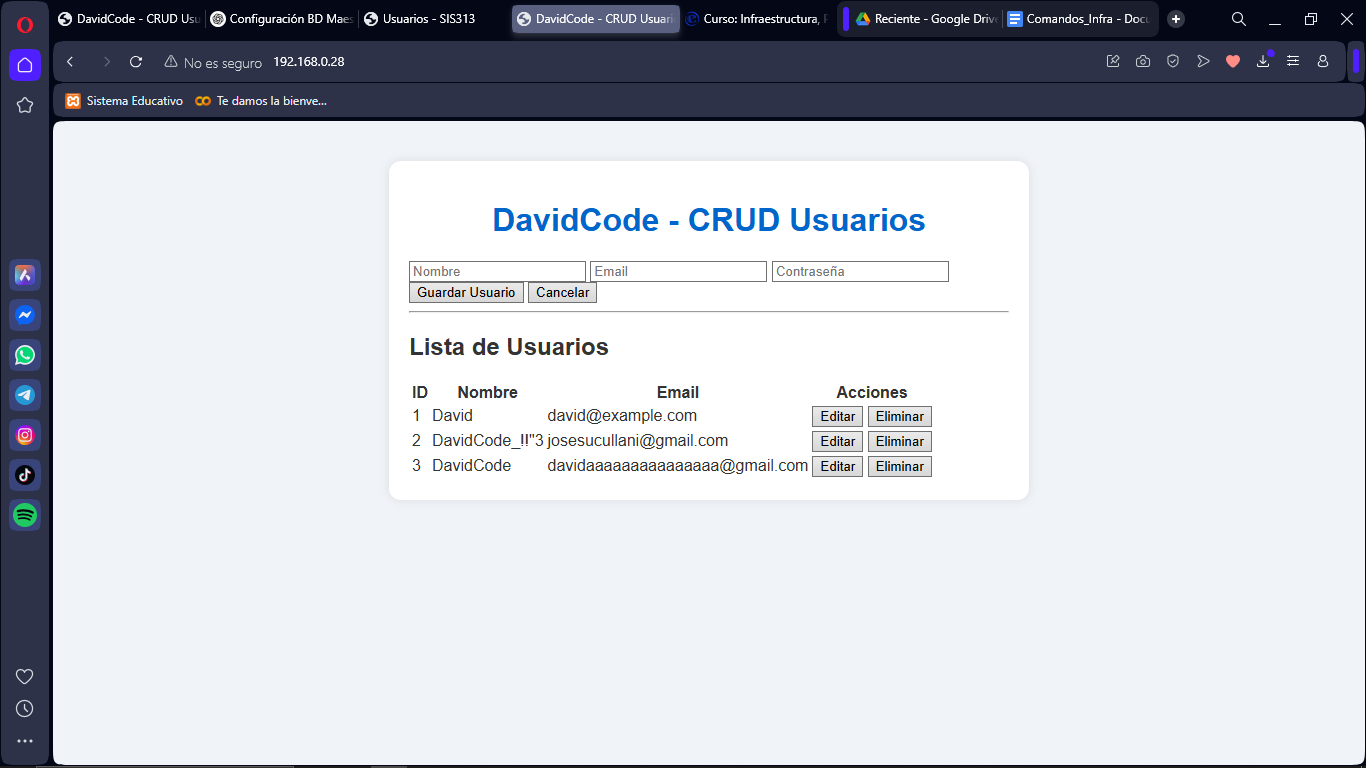
Primero la puerto 3001 con la base de datos maestro

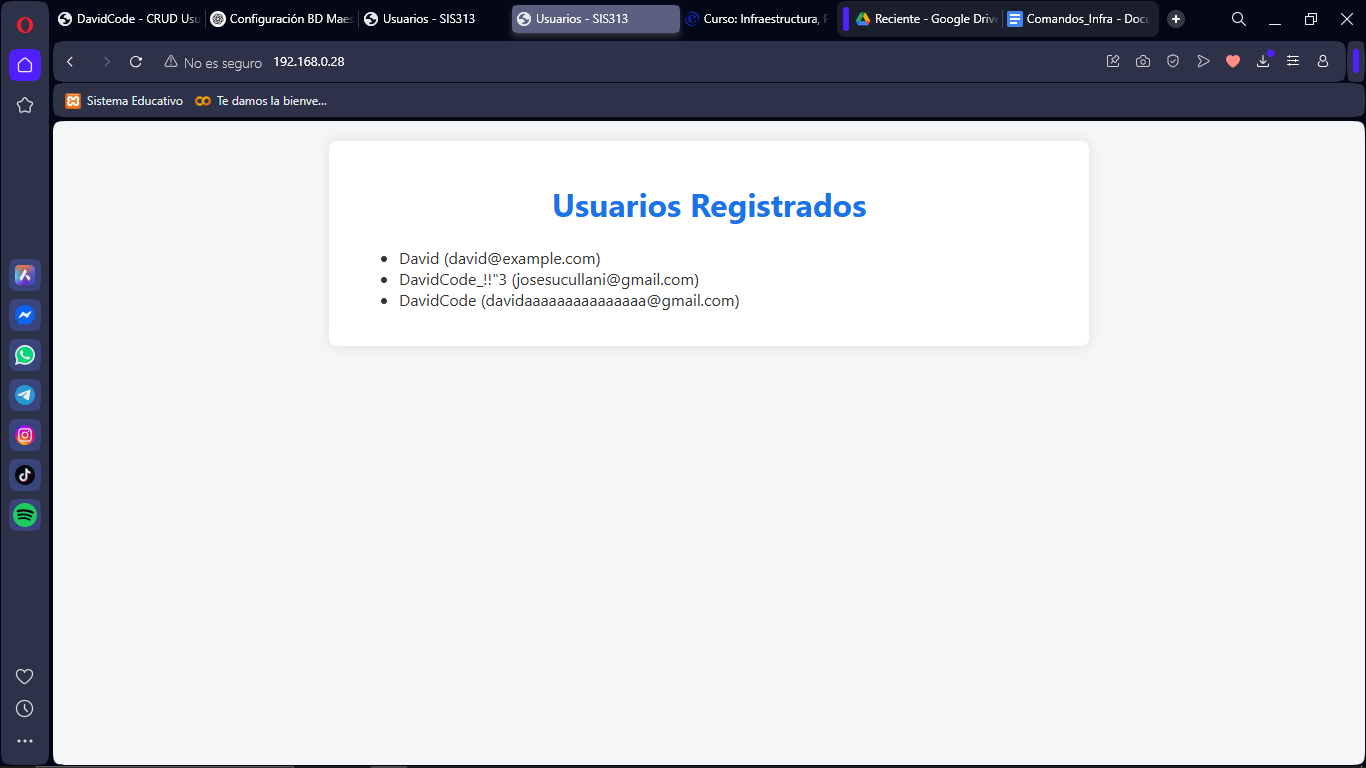


Segundo mostrando la base de datos esclavo con el puerto 3002



El balanceo entre base de datos





Pasos para instalar y simular fallo y recuperación en RAID 1

1. Instalar herramientas necesarias

sudo apt update

sudo apt install mdadm -y

2.- crear discos virtuales (2 archivos de 100 MB para simular discos)

dd if=/dev/zero of=disk1.img bs=1M count=100

dd if=/dev/zero of=disk2.img bs=1M count=100

3. Asignar discos virtuales a dispositivos loop

sudo losetup /dev/loop10 disk1.img

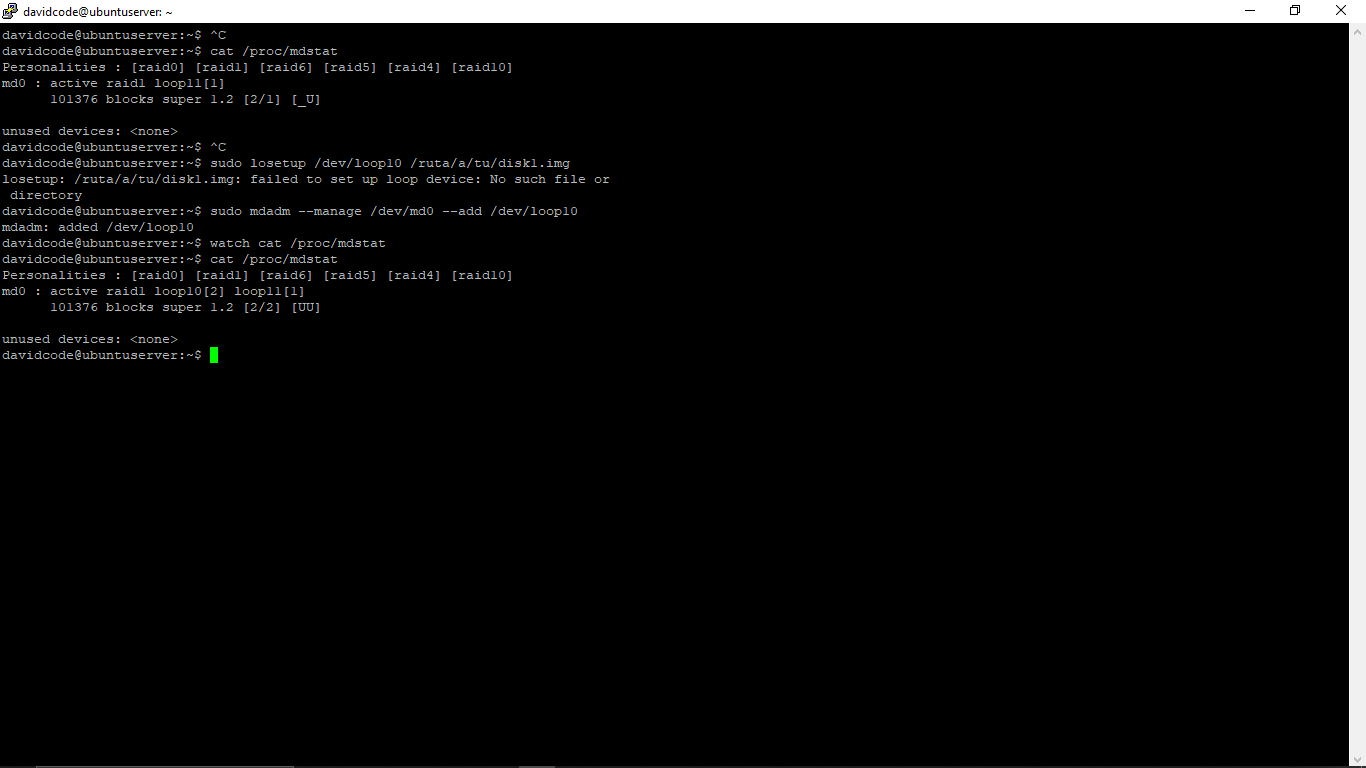
sudo losetup /dev/loop11 disk2.img

4. Crear el arreglo RAID 1 (espejo) con esos discos

sudo mdadm --create --verbose /dev/md0 --level=1 --raid-devices=2 /dev/loop10 /dev/loop11

5.- Verificar estado del RAID

cat /proc/mdstat

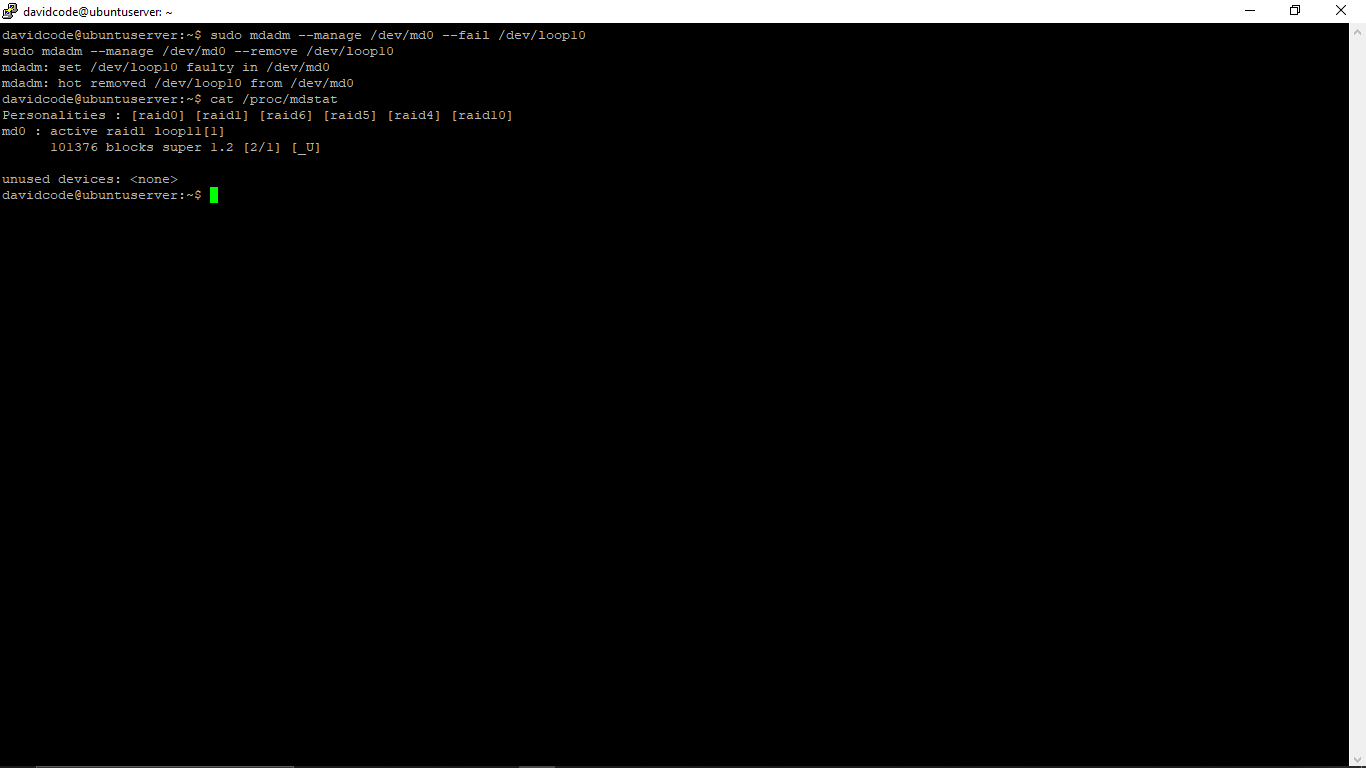


Simular falla de un disco (fallo en /dev/loop10)

sudo mdadm --manage /dev/md0 --fail /dev/loop10

sudo mdadm --manage /dev/md0 --remove /dev/loop10

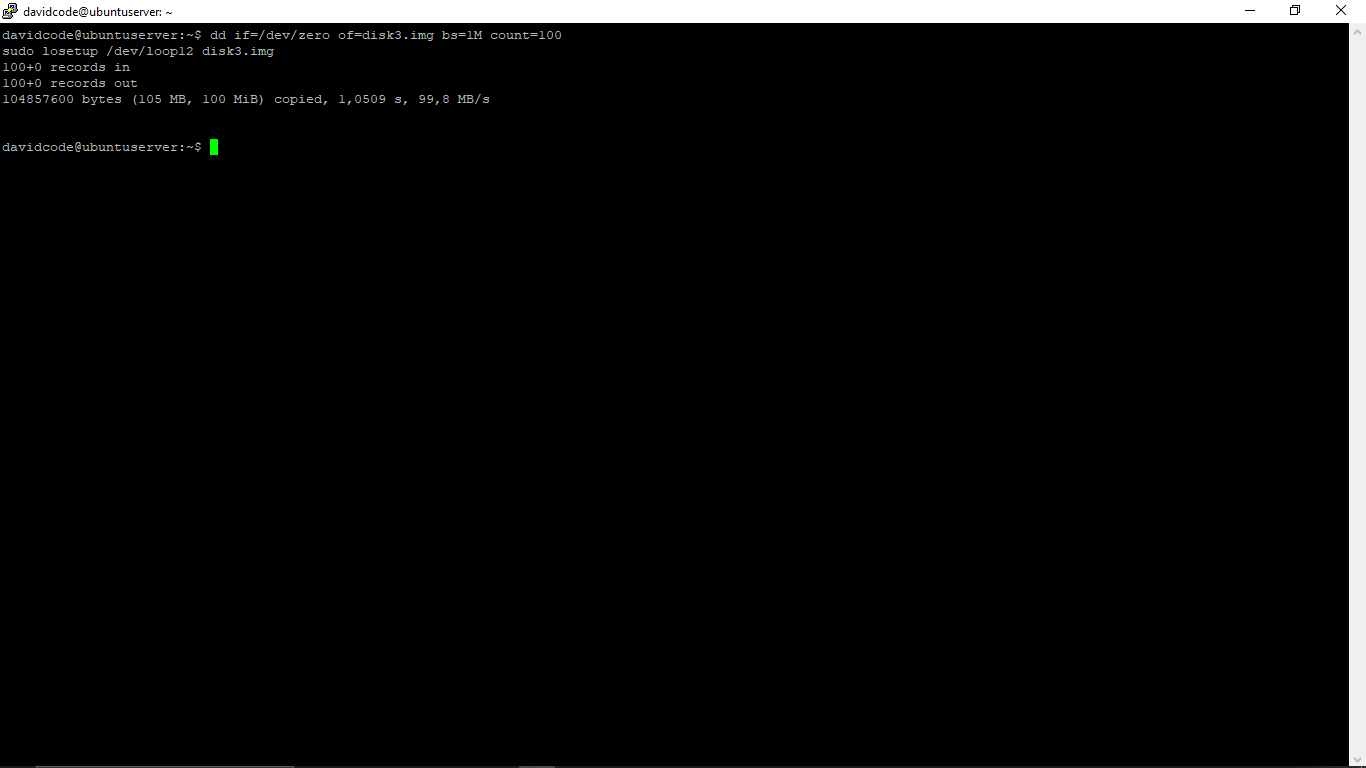
Simula el fallo de disco habiendo U envés de dos UU



10. Crear nuevo disco virtual para reemplazar

dd if=/dev/zero of=disk3.img bs=1M count=100

sudo losetup /dev/loop12 disk3.img



12. Verificar el progreso de reconstrucción

watch cat /proc/mdstat

Verás que el RAID sincroniza el nuevo disco. Cuando termine, tendrás de nuevo:

md0 : active raid1 loop12[2] loop11[1]

104320 blocks [2/2] [UU]

