

**UNIVERSIDAD DE LAS AMERICAS FACULTAD
DE INGENIERIA Y CIENCIAS APLICADAS
ING. EN CIBERSEGURIDAD**

ITIZ3201 ADMINISTRACIÓN DE BASE DE DATOS

PROYECTO INTEGRADOR

TEMA

**Proyecto Integrador- Solución de base de datos Courier
contenerizada**

Docente:

Patricio Moreno

Autor:

William Morales- Ana Belen Gavilanes- Jonathan Lema

Fecha:

2024

OBJETIVO PROPUESTO DE LA ACTIVIDAD:

El trabajo por realizar tiene como finalidad implementar una solución de base de datos contenerizada que permita cubrir los siguientes puntos:

- Plantear el despliegue de una solución de base de datos contenerizada
- Efectuar el despliegue de una solución de base de datos contenerizada
- Validar el despliegue de una solución de base de datos contenerizada
- Analizar los riesgos de seguridad a nivel privilegios que puede poseer una base de datos contenerizada
- Plantear una alternativa de solución a los riesgos de seguridad a nivel privilegios que puede poseer una base de datos contenerizada, basándose en un enfoque RBA
- Implementar una solución de seguridad basada en RBA para la operación segura de una base de datos contenerizada

INDICACIONES:

En grupos de 2 o 3 estudiantes se deberá construir, como habilitador para poder rendir las evaluaciones del tercer progreso y eventualmente del examen de recuperación, una solución de base de datos contenerizada.

Para empezar, se deberá implementar la mencionada solución de base de datos contenerizada, basándose en las buenas prácticas vistas y aplicadas en clase:

- Archivos de configuración deberán residir en un almacenamiento persistente.
- Archivos de la base de datos deberán residir en un almacenamiento persistente
- No se deberá utilizar el puerto de conexión por default
- Se deberá mapear el puerto de conexión de la instancia de base de datos hacia el host donde se ejecuta el contenedor
- Se deberá utilizar un cliente gráfico SQL para acceder y manejar la instancia de base de datos contenerizada.

Posteriormente, se deberá diseñar e implementar en la base de datos contenerizada, un modelo de datos que permita solventar una necesidad de almacenamiento de datos planteada por el docente.

Una vez implementado el modelo de datos, se deberá analizar los riesgos de seguridad que podría existir en el acceso a dichos datos, con el fin de plantear e implementar una solución basada en un enfoque RBA, la cual permitirá que la base de datos contenerizada opere de manera segura, una vez más, en base a las buenas prácticas vistas y aplicadas en clase:

- Deberá existir un rol de base de datos con privilegios de solo lectura
- Deberá existir al menos un rol de base de datos con privilegios de manipular los datos
- Deberá existir un primer usuario de base de datos a quien se le deberá conceder el primer rol.
- Deberá existir un segundo usuario de base de datos a quien se le deberá conceder el segundo rol.

- Deberán generarse evidencias del correcto funcionamiento de la solución de seguridad basada en RBA.

FORMA DE TRABAJO:

El líder de cada uno de los grupos de trabajo remitirá tan pronto como le sea posible un correo al docente solicitando se indique la necesidad de almacenamiento que se deberá implementar en la base de datos contenerizada.

Cada uno de los grupos procederá a implementar una solución de base de datos contenerizada en base a las indicaciones dadas y considerando además los siguientes puntos:

- La solución de base de datos contenerizada deberá implementarse en una máquina virtual con Linux
- La solución de base de datos contenerizada debe implementarse utilizando la última versión de PostgreSQL.
- El cliente gráfico SQL para acceder a la base de datos contenerizada deberá instalarse en el computador host (este cliente se usará en la defensa del proyecto).
- El diseño de la base de datos implementado deberá estar acompañado de su diagrama de diseño físico de la base de datos, y del diccionario de datos.
- El contenido de la base de datos contenerizada deberá ser generado aleatoriamente utilizando Python y el módulo Faker (<https://faker.readthedocs.io/en/master/>)

ESPECIFICACIONES DE ENTREGA:

Se deberá crear un proyecto de GitHub, conteniendo la estructura de carpetas adecuada para la solución de base de datos contenerizada implementada, incluyendo la documentación de como instalar y ejecutar la solución.

Cada grupo deberá generar un PDF formal con el siguiente contenido:

1. Portada

2. Descripción de la Solución Implementada

La solución se implementó en una máquina virtual con Linux, utilizando la última versión de PostgreSQL. El contenedor de la base de datos se creó utilizando Dockerfile, y se mapeó el puerto 5432 del contenedor al puerto 5432 del host.

Modelo de datos

El modelo de datos implementado se basa en las siguientes tablas:

Tabla	Descripción
Cliente	Almacena información básica sobre los clientes, como su nombre, apellido, dirección y número de teléfono.
Destinatario	Almacena información sobre los destinatarios de las encomiendas, como su nombre, apellido, dirección y número de teléfono.
Encomienda	Almacena información sobre las encomiendas, como su número de seguimiento, fecha de creación, fecha de entrega, etc.
Envío	Almacena información sobre los envíos, como su tipo, peso, dimensiones, etc.
Estado encomienda	Almacena el estado actual de la encomienda.
Repartidor	Almacena información sobre los repartidores, como su nombre, apellido, número de identificación, etc.
Ruta	Almacena información sobre las rutas de reparto, como su nombre, origen, destino, etc.
Ticket	Almacena información sobre los tickets de servicio, como su número, fecha de creación, etc.
Tipo envío	Almacena información sobre los tipos de envío, como su nombre, descripción, etc.

Diagrama de diseño lógico de la base de datos bdcourier con power designer

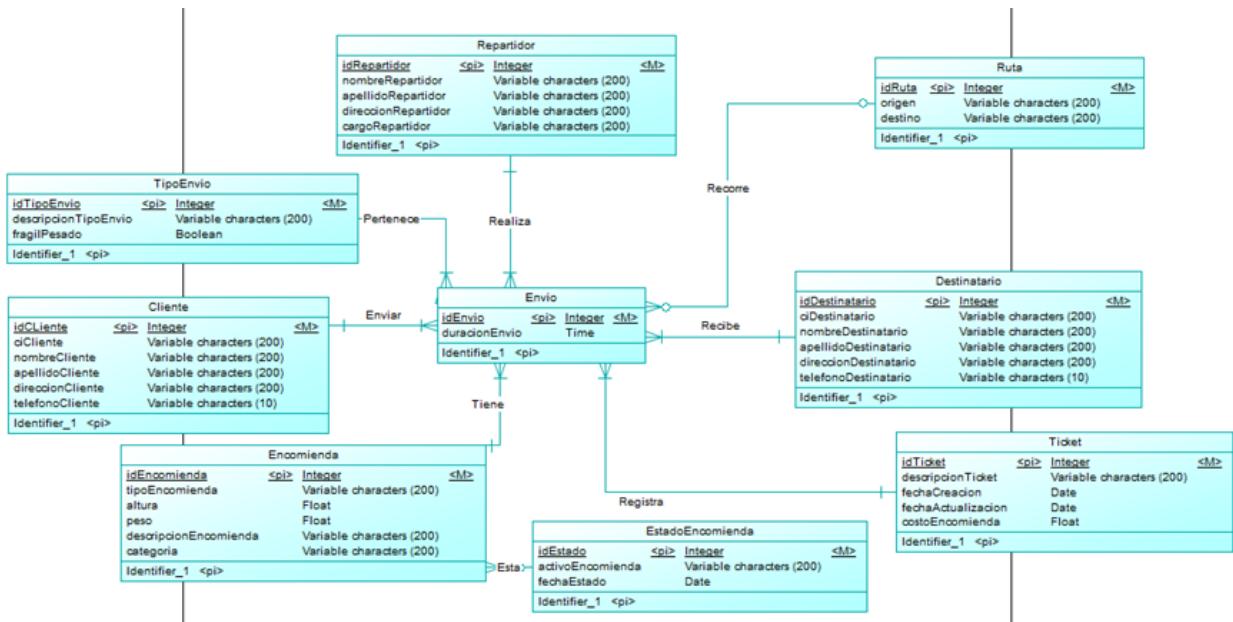
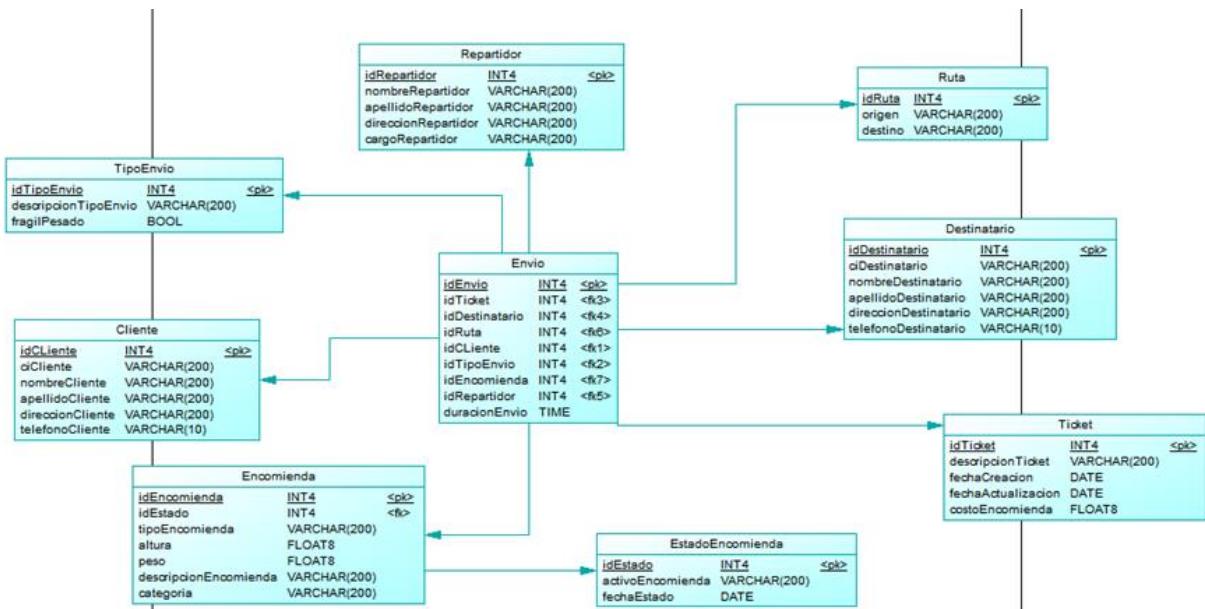


Diagrama de diseño físico de la base de datos bdcourier con power designer



Análisis de riesgos de seguridad

El análisis de riesgos de seguridad identificó los siguientes riesgos potenciales:

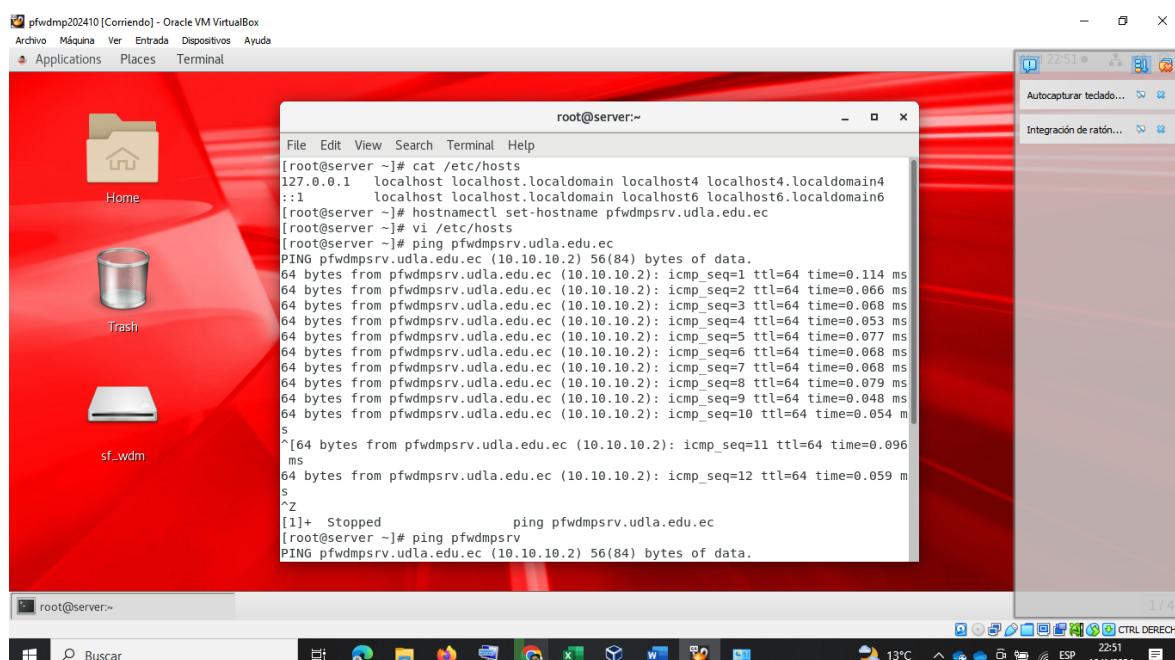
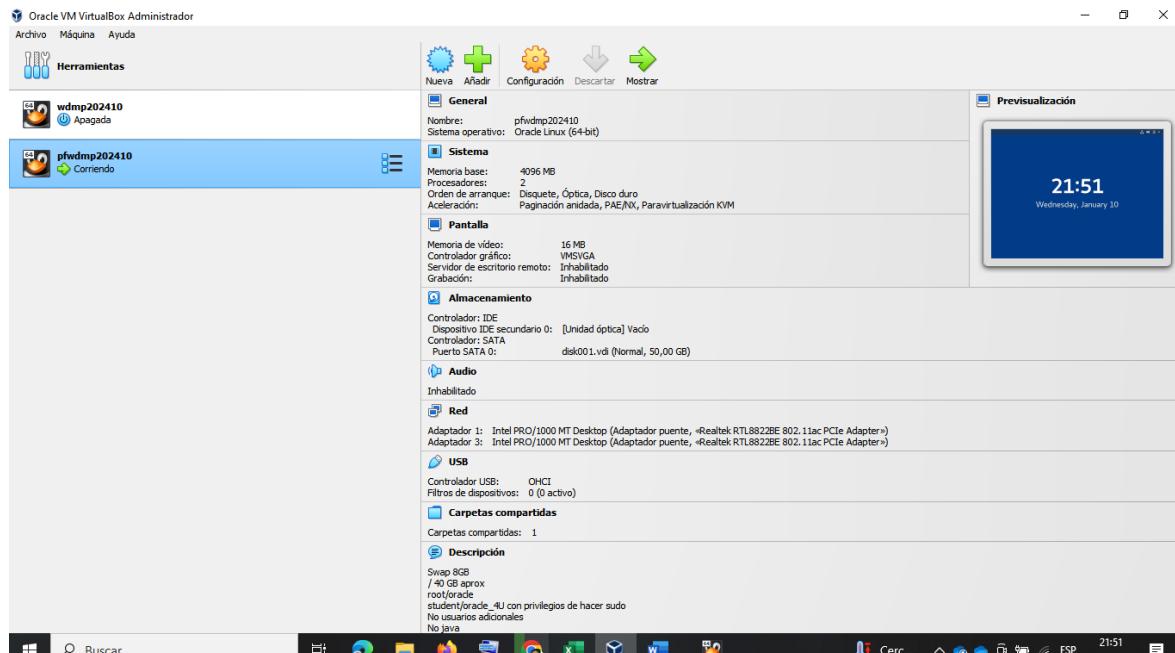
- Un usuario malintencionado podría acceder a los datos de los clientes, destinatarios, encomiendas, datos de clientes, etc.
- Un usuario malintencionado podría modificar o eliminar los datos de los clientes, destinatarios, encomiendas, etc.

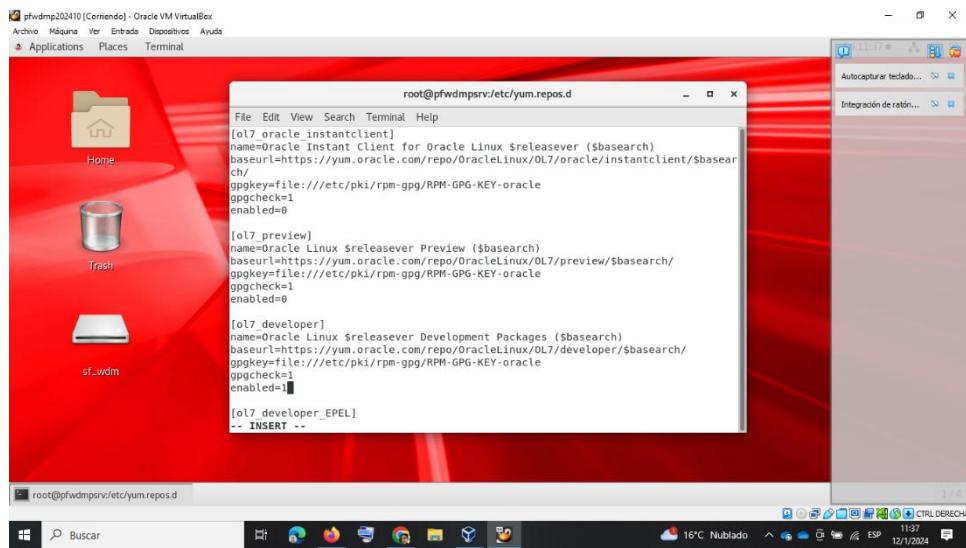
- Un usuario malintencionado podría crear nuevas encomiendas o tickets.

Para mitigar estos riesgos, se implementó una solución de seguridad basada en RBA.

Desarrollo:

1. Creación de la máquina virtual Oracle





2. Instalación de Docker en OL 7.6

- Vamos a ingresar el comando: cd /etc/yum.repos.d/ para ir al repositorio que nos va a servir para instalar Docker
- Ingresamos el comando: **vi public-yum-ol7.repo** para realizar modificaciones para la instalación de **Docker**

```

student@server:/etc/yum.repos.d
File Edit View Search Terminal Help
[student@jelvdockersrv ~]$ sudo su
[sudo] password for student:
[root@jelvdockersrv student]# cd /etc/yum.repos.d/
[root@jelvdockersrv yum.repos.d]# ll
total 20
-rw-r--r--. 1 root root 129 Dec 23 2020 iso.repo
-rw-r--r--. 1 root root 13093 Dec 23 2020 public-yum-ol7.repo
[root@jelvdockersrv yum.repos.d]# vi public-yum-ol7.repo

```

- Vamos a colocar **1** en las líneas de **enabled** que se encuentran con valor de **0**:

```

[ol7 developer]
name=Oracle Linux $releasever Development Packages ($basearch)
baseurl=https://yum.oracle.com/repo/OracleLinux/OL7/developer/$basearch/
gpgkey=file:///etc/pki/rpm-gpg/RPM-GPG-KEY-oracle
gpgcheck=1
enabled=1

[ol7 developer_EPEL]
name=Oracle Linux $releasever Development Packages ($basearch)
baseurl=https://yum.oracle.com/repo/OracleLinux/OL7/developer_EPEL/$basearch/
gpgkey=file:///etc/pki/rpm-gpg/RPM-GPG-KEY-oracle
gpgcheck=1
-- INSERT --

```

- Ingresamos el comando **yum repoinfo** para refreshar el repositorio de paquetes
- Ingresamos el comando: **yum install device-mapper-persistent-data lvm2** para realziar la instalación

```

student@server:/etc/yum.repos.d
File Edit View Search Terminal Help
[root@jelvdockersrv yum.repos.d]# yum install device-mapper-persistent-data lvm2

```

```
student@server:/etc/yum.repos.d
File Edit View Search Terminal Help

=====
Package          Arch    Version      Repository   Size
=====
Updating:
lvm2            x86_64  7:2.02.187-6.0.5.el7_9.5  ol7_latest  1.3 M
yum-utils        noarch   1.1.31-54.0.1.el7_8     ol7_latest  122 k
Updating for dependencies:
device-mapper    x86_64  7:1.02.170-6.0.5.el7_9.5  ol7_latest  297 k
device-mapper-event x86_64  7:1.02.170-6.0.5.el7_9.5  ol7_latest  192 k
device-mapper-event-libs x86_64  7:1.02.170-6.0.5.el7_9.5  ol7_latest  192 k
device-mapper-libs  x86_64  7:1.02.170-6.0.5.el7_9.5  ol7_latest  325 k
lvm2-libs        x86_64  7:2.02.187-6.0.5.el7_9.5  ol7_latest  1.1 M
lvm2-python-libs x86_64  7:2.02.187-6.0.5.el7_9.5  ol7_latest  189 k

Transaction Summary
=====
Upgrade 2 Packages (+6 Dependent packages)

Total download size: 3.7 M
Is this ok [y/d/N]: Exiting on user command
Your transaction was saved, rerun it with:
  yum load-transaction /tmp/yum_save_tx.2023-04-25.11-46.f0cfAp.yumtx
[root@jelvdockersrv yum.repos.d]#
```

```
student@server:/etc/yum.repos.d
File Edit View Search Terminal Help

=====
Verifying : 7:device-mapper-libs-1.02.170-6.0.5.el7_9.5.x86_64      7/16
Verifying : 7:device-mapper-event-libs-1.02.170-6.0.5.el7_9.5.x86_64  8/16
Verifying : 7:lvm2-2.02.180-8.el7.x86_64                           9/16
Verifying : 7:lvm2-libs-2.02.180-8.el7.x86_64                      10/16
Verifying : 7:device-mapper-1.02.149-8.el7.x86_64                  11/16
Verifying : yum-utils-1.1.31-50.0.1.el7.noarch                     12/16
Verifying : 7:lvm2-python-libs-2.02.180-8.el7.x86_64                13/16
Verifying : 7:device-mapper-libs-1.02.149-8.el7.x86_64              14/16
Verifying : 7:device-mapper-event-1.02.149-8.el7.x86_64             15/16
Verifying : 7:device-mapper-event-libs-1.02.149-8.el7.x86_64         16/16

Updated:
lvm2.x86_64 7:2.02.187-6.0.5.el7_9.5  yum-utils.noarch 0:1.1.31-54.0.1.el7_8

Dependency Updated:
device-mapper.x86_64 7:1.02.170-6.0.5.el7_9.5
device-mapper-event.x86_64 7:1.02.170-6.0.5.el7_9.5
device-mapper-event-libs.x86_64 7:1.02.170-6.0.5.el7_9.5
device-mapper-libs.x86_64 7:1.02.170-6.0.5.el7_9.5
lvm2-libs.x86_64 7:2.02.187-6.0.5.el7_9.5
lvm2-python-libs.x86_64 7:2.02.187-6.0.5.el7_9.5

Complete!
[root@jelvdockersrv yum.repos.d]#
```

-Vamos a añadir el repositorio en donde se descarga Docker con el comando: **yum-config-manager –add-repo <https://download.docker.com/linux/centos/docker-ce.repo>**

```
student@server:/etc/yum.repos.d
File Edit View Search Terminal Help
[root@jelvdockersrv yum.repos.d]# yum-config-manager --add-repo https://download.docker.com/linux/centos/docker-ce.repo
```

```
student@server:/etc/yum.repos.d
File Edit View Search Terminal Help
[root@jelvdockersrv yum.repos.d]# yum-config-manager --add-repo https://download.docker.com/linux/centos/docker-ce.repo
Loaded plugins: langpacks
adding repo from: https://download.docker.com/linux/centos/docker-ce.repo
grabbing file https://download.docker.com/linux/centos/docker-ce.repo to /etc/yum.repos.d/docker-ce.repo
repo saved to /etc/yum.repos.d/docker-ce.repo
[root@jelvdockersrv yum.repos.d]#
```

- Vamos a proceder a instalar **Docker** con el comando: **yum install Docker**:

```
student@server:/etc/yum.repos.d
File Edit View Search Terminal Help
[root@jelvdockersrv yum.repos.d]# yum-config-manager --add-repo https://download.docker.com/linux/centos/docker-ce.repo
Loaded plugins: langpacks
adding repo from: https://download.docker.com/linux/centos/docker-ce.repo
grabbing file https://download.docker.com/linux/centos/docker-ce.repo to /etc/yum.repos.d/docker-ce.repo
repo saved to /etc/yum.repos.d/docker-ce.repo
[root@jelvdockersrv yum.repos.d]# ll
total 24
-rw-r--r--. 1 root root 1919 Apr 19 07:29 docker-ce.repo
-rw-r--r--. 1 root root 129 Dec 23 2020 iso.repo
-rw-r--r--. 1 root root 13093 Apr 25 11:01 public-yum-ol7.repo
[root@jelvdockersrv yum.repos.d]# yum install docker
```

- Verificamos que se realizó la instalación:

```
student@server:/etc/yum.repos.d
File Edit View Search Terminal Help
Verifying : docker-buildx-plugin-0.10.4-1.el7.x86_64 4/10
Verifying : 3:docker-ce-23.0.4-1.el7.x86_64 5/10
Verifying : slirp4netns-0.4.3-4.el7_8.x86_64 6/10
Verifying : 2:container-selinux-2.119.2-1.911c772.el7_8.noarch 7/10
Verifying : docker-ce-rootless-extras-23.0.4-1.el7.x86_64 8/10
Verifying : containerd.io-1.6.20-3.1.el7.x86_64 9/10
Verifying : docker-compose-plugin-2.17.2-1.el7.x86_64 10/10

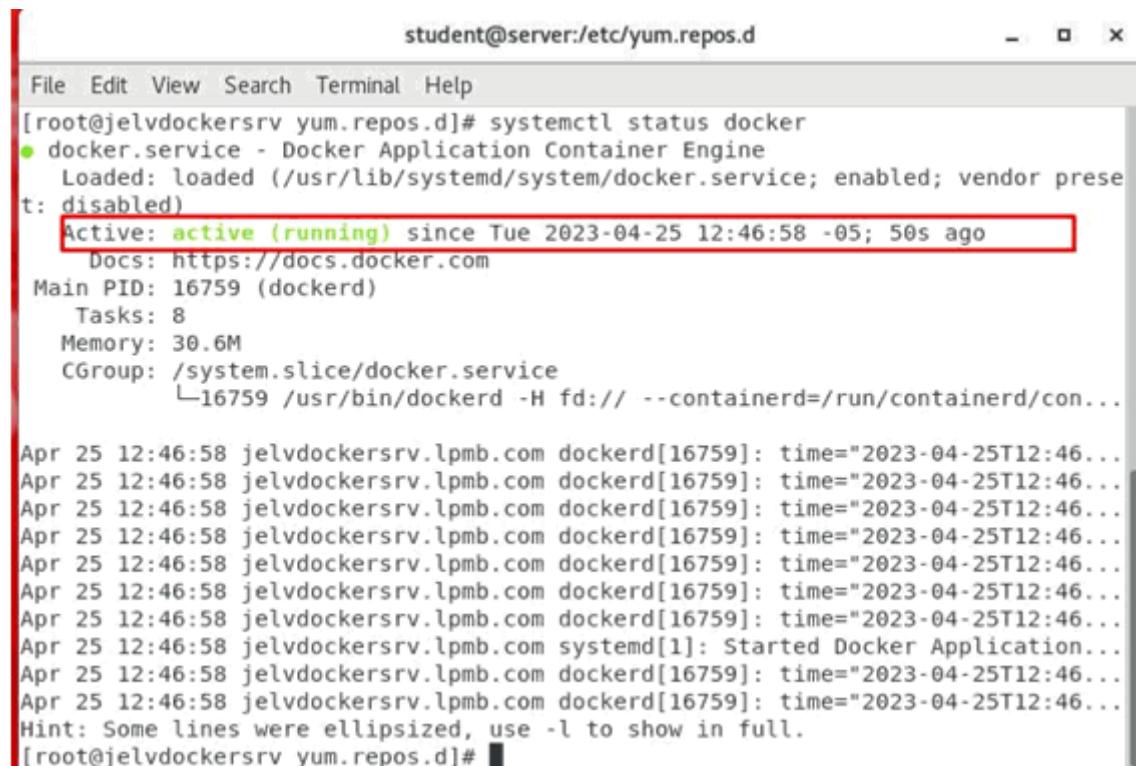
Installed:
 docker-ce.x86_64 3:23.0.4-1.el7

Dependency Installed:
 container-selinux.noarch 2:2.119.2-1.911c772.el7_8
 containerd.io.x86_64 0:1.6.20-3.1.el7
 docker-buildx-plugin.x86_64 0:0.10.4-1.el7
 docker-ce-cli.x86_64 1:23.0.4-1.el7
 docker-ce-rootless-extras.x86_64 0:23.0.4-1.el7
 docker-compose-plugin.x86_64 0:2.17.2-1.el7
 fuse-overlayfs.x86_64 0:0.7.2-6.el7_8
 fuse3-libs.x86_64 0:3.6.1-4.el7
 slirp4netns.x86_64 0:0.4.3-4.el7_8

Complete!
[root@jelvdockersrv yum.repos.d]#
```

- habilita el servicio con el comando: **systemctl enable Docker**

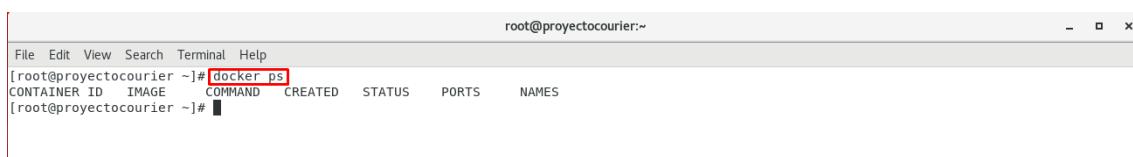
- Iniciamos el servicio de Docker con el comando: **systemctl start docker**
- Comprobamos que el servicio esta levantado con el comando: **systemctl status docker**



```
student@server:/etc/yum.repos.d
File Edit View Search Terminal Help
[root@jelvdockersrv yum.repos.d]# systemctl status docker
● docker.service - Docker Application Container Engine
  Loaded: loaded (/usr/lib/systemd/system/docker.service; enabled; vendor prese
t: disabled)
  Active: active (running) since Tue 2023-04-25 12:46:58 -05; 50s ago
    Docs: https://docs.docker.com
  Main PID: 16759 (dockerd)
    Tasks: 8
   Memory: 30.6M
     CGroup: /system.slice/docker.service
             └─16759 /usr/bin/dockerd -H fd:// --containerd=/run/containerd/con...
Apr 25 12:46:58 jelvdockersrv.lpmb.com dockerd[16759]: time="2023-04-25T12:46...
Hint: Some lines were ellipsized, use -l to show in full.
[root@jelvdockersrv yum.repos.d]#
```

3. Instalar Postgresql en contenedor de Docker

- Verificar que no hay contenedores corriendo en docker con el comando: docker ps



```
root@proyectocourier:~
File Edit View Search Terminal Help
[root@proyectocourier ~]# docker ps
CONTAINER ID  IMAGE      COMMAND   CREATED   STATUS    PORTS      NAMES
[root@proyectocourier ~]#
```

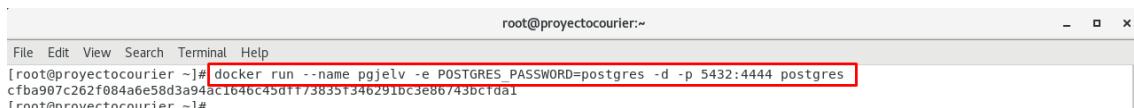
- Realizar la descarga de la imagen de PostgreSQL con el comando: docker pull postgres

```
[root@proyectocourier ~]# docker pull postgres
Using default tag: latest
latest: Pulling from library/postgres
2f44b7a888fa: Pull complete
6d49150dabe2: Pull complete
18d6a86d0fbf: Pull complete
4c9385c30bce: Pull complete
550091272acc: Pull complete
2720859ac49e: Pull complete
b8091cf53545: Pull complete
f3ca5fbdb89cd: Pull complete
22fbbce47a56: Downloading [=====]
b3b5e3b65b95: Download complete
917e5b76e085: Download complete
7f21ce9572c6: Download complete
4ea3941c8572: Download complete
848fee03c034: Download complete
] 50.57MB/108.6MB
```

- Con el comando **docker images** verificamos que ya se encuentra descargada la imagen de **postgres**:

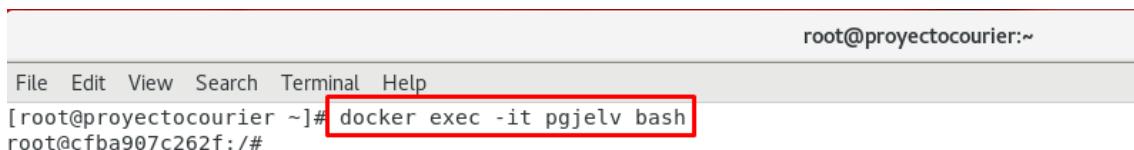
```
[root@proyectocourier ~]# docker images
REPOSITORY      TAG          IMAGE ID      CREATED       SIZE
postgres        <none>      bdc467e80232   7 days ago   425MB
postgres        latest       75b7bff7c3ad   7 days ago   425MB
[root@proyectocourier ~]#
```

- Ejecución de un contenedor de PostgreSQL bajo el nombre pgjelv que permita mapear el puerto por default de PostgreSQL al puerto 4444 del host, que posea el archivo de configuración y el directorio de datos por fuera del contenedor: docker run --name pgjelv -e POSTGRES_PASSWORD=postgres -d -p 5432:4444 postgres



A screenshot of a terminal window titled 'root@proyectocourier:~'. The window shows a command line with the following text:
File Edit View Search Terminal Help
[root@proyectocourier ~]# docker run --name pgjelv -e POSTGRES_PASSWORD=postgres -d -p 5432:4444 postgres
cfba907c262f084a6e58d3a94ac1646c45d1f78351346291bc3e86743bctdal
[root@proyectocourier ~]#

- Vamos a conectarnos dentro del contenedor que se está ejecutando de Postgres con el comando **docker exec -it pgjelv bash**



A screenshot of a terminal window titled 'root@proyectocourier:~'. The window shows a command line with the following text:
File Edit View Search Terminal Help
[root@proyectocourier ~]# docker exec -it pgjelv bash
root@cfba907c262f:/#

- Observamos el **id** con el que estamos conectados:

```
root@cfba907c262f:/# id postgres
uid=999(postgres) gid=999(postgres) groups=999(postgres),101(ssl-cert)
root@cfba907c262f:/#
```

- Nos conectamos con el **id postgres**:

```
root@cfba907c262f:/# su - postgres
postgres@cfba907c262f:~$
```

- Ingresamos el comando **psql** que es el utilitario para ingresar a **postgres**:

```
postgres@cfba907c262f:~$ psql
psql (16.1 (Debian 16.1-1.pgdg120+1))
Type "help" for help.

postgres=#
```

- Con el comando **\l** observamos las bases de datos existentes:

```
root@cfba907c262f:/# su - postgres
postgres@cfba907c262f:~$ psql
psql (16.1 (Debian 16.1-1.pgdg120+1))
Type "help" for help.

postgres=# \l
                                         List of databases
   Name   | Owner    | Encoding | Locale Provider | Collate      | Ctype       | ICU Locale | ICU Rules | Access privileges
   +-----+-----+-----+-----+-----+-----+-----+-----+
postgres | postgres | UTF8   | libc        | en_US.utf8  | en_US.utf8  |             |             | =c/postgres +
template0 | postgres | UTF8   | libc        | en_US.utf8  | en_US.utf8  |             |             | postgres=CTc/postgres +
template1 | postgres | UTF8   | libc        | en_US.utf8  | en_US.utf8  |             |             | =c/postgres +
(3 rows)
postgres=#

```

- Observamos como estamos conectados con el comando **\conninfo**:

```
postgres=# \conninfo
You are connected to database "postgres" as user "postgres" via socket in "/var/run/postgresql" at port "5432".
postgres=#

```

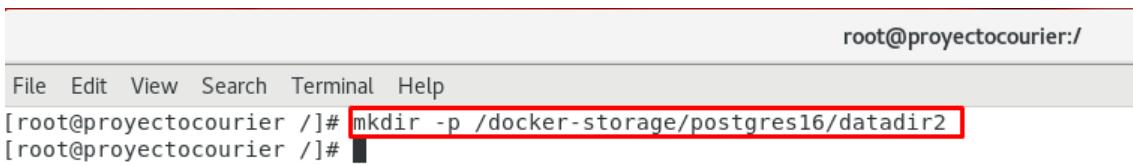
- Salimos del contenedor con el comando **exit**:

```
You are connected to database "postgres" as user "postgres" via socket in "/var/run/postgresql" at port "5432".
postgres=# exit
postgres@cfba907c262f:~$ exit
logout
root@cfba907c262f:~/# exit
exit
[root@proyectocourier ~]#
```

- Detenemos la ejecución del contenedor pgjelv:

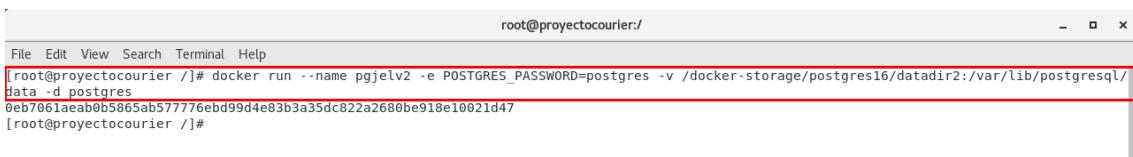
```
[root@proyectocourier ~]#
[root@proyectocourier ~]# docker stop pgjelv
pgjelv
[root@proyectocourier ~]#
```

- Vamos a crear el directorio donde se va a almacenar la base de datos por fuera del contenedor, detenemos la ejecución del contenedor pgjelv.
- Creamos el directorio con el comando **mkdir -p /docker-storage/postgres16/datadir2**



```
root@proyectocourier:/ 
File Edit View Search Terminal Help
[root@proyectocourier /]# mkdir -p /docker-storage/postgres16/datadir2
[root@proyectocourier /]#
```

- Vamos a correr un contenedor de postgres en el directorio creado con el nombre de pgjelv2 con el comando: **docker run --name pgjelv2 -e POSTGRES_PASSWORD=postgres -v /docker-storage/postgres16/datadir2:/var/lib/postgresql/data -d postgres**

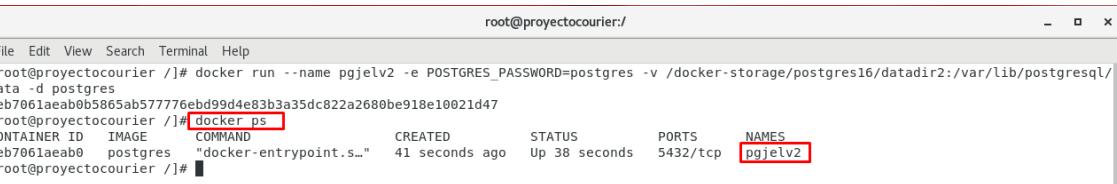


```
root@proyectocourier:/ 
File Edit View Search Terminal Help
[root@proyectocourier /]# docker run --name pgjelv2 -e POSTGRES_PASSWORD=postgres -v /docker-storage/postgres16/datadir2:/var/lib/postgresql/
data -d postgres
0eb7061aeab0b5865ab577776ebd99d4e83b3a35dc822a2680be918e10021d47
[root@proyectocourier /]#
```

- Con el comando docker -ps observamos que se encuentra corriendo el contenedor **pgjelv2**:



```
root@proyectocourier:/ 
File Edit View Search Terminal Help
[root@proyectocourier /]# docker run --name pgjelv2 -e POSTGRES_PASSWORD=postgres -v /docker-storage/postgres16/datadir2:/var/lib/postgresql/
data -d postgres
0eb7061aeab0b5865ab577776ebd99d4e83b3a35dc822a2680be918e10021d47
[root@proyectocourier /]# docker ps
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES
0eb7061aeab0 postgres "docker-entrypoint.s..." 41 seconds ago Up 38 seconds 5432/tcp pgjelv2
[root@proyectocourier /]#
```



```
root@proyectocourier:/ 
File Edit View Search Terminal Help
[root@proyectocourier /]# docker run --name pgjelv2 -e POSTGRES_PASSWORD=postgres -v /docker-storage/postgres16/datadir2:/var/lib/postgresql/
data -d postgres
0eb7061aeab0b5865ab577776ebd99d4e83b3a35dc822a2680be918e10021d47
[root@proyectocourier /]# docker ps
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES
0eb7061aeab0 postgres "docker-entrypoint.s..." 41 seconds ago Up 38 seconds 5432/tcp pgjelv2
[root@proyectocourier /]#
```

- Vamos a ejecutar el comando **docker exec -it pgjelv2 bash** para ingresar al bash del contenedor:

```
root@proyectocourier:/#
File Edit View Search Terminal Help
[root@proyectocourier /]# docker exec -it pgjelv2 bash
root@0eb7061aeab0:/#
```

- Ingresamos el comando: **su – postgres**

```
root@0eb7061aeab0:/# su - postgres
postgres@0eb7061aeab0:~$
```

- Ingresamos con el comando **psql**:

```
postgres@0eb7061aeab0:~$ psql
psql (16.1 (Debian 16.1-1.pgdg120+1))
Type "help" for help.

postgres=#
```

- Nos salimos del contenedor con el comando exit:

```
postgres@0eb7061aeab0:~$ psql
psql (16.1 (Debian 16.1-1.pgdg120+1))
Type "help" for help.

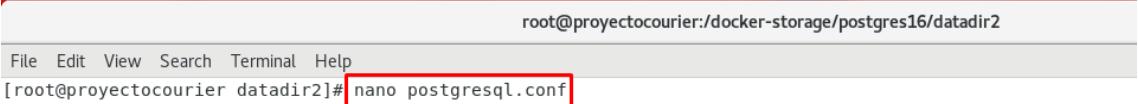
postgres=# exit
postgres@0eb7061aeab0:~$ exit
logout
root@0eb7061aeab0:/# exit
[root@proyectocourier /]#
[root@proyectocourier /]#
[root@proyectocourier /]#
```

- Accedemos al directorio: **cd /docker-storage/postgres16/datadir2**:

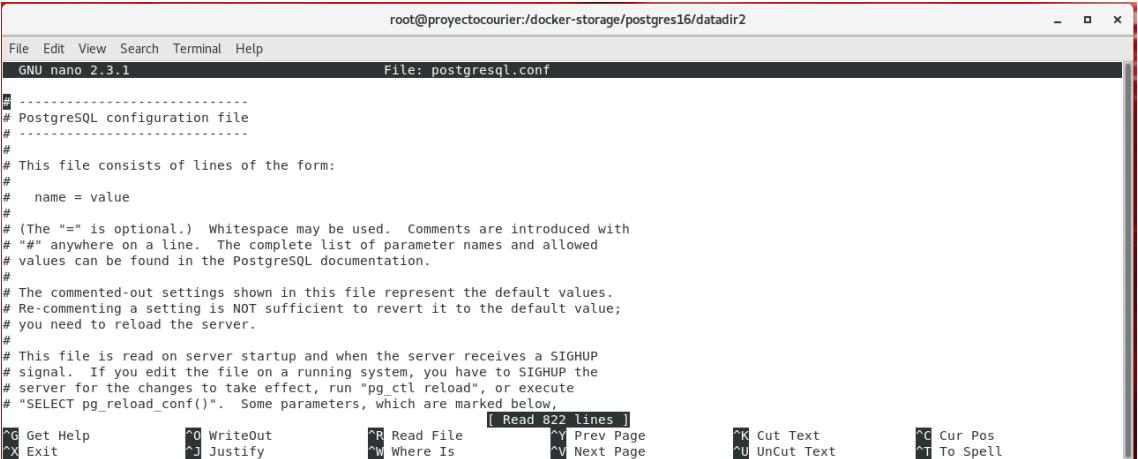
```
root@proyectocourier:/docker-storage/postgres16/datadir2
File Edit View Search Terminal Help
[root@proyectocourier /]# cd /docker-storage/postgres16/datadir2
[root@proyectocourier datadir2]#
```

4. Cambiar de puerto por defecto de Postgresql a puerto 4444

- Vamos a editar el archivo **postgresql.conf** con el comando: **nano postgresql.conf**:

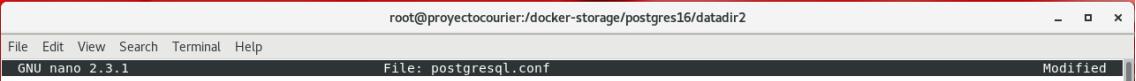


```
root@proyectocourier:/docker-storage/postgres16/datadir2
File Edit View Search Terminal Help
[root@proyectocourier datadir2]# nano postgresql.conf
```

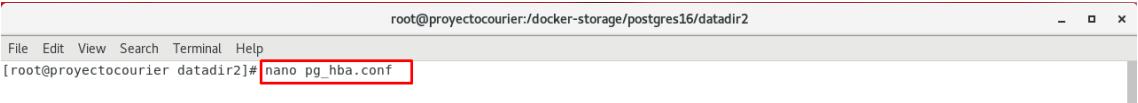
```
root@proyectocourier:/docker-storage/postgres16/datadir2
File Edit View Search Terminal Help
GNU nano 2.3.1          File: postgresql.conf
-----
# PostgreSQL configuration file
-----
#
# This file consists of lines of the form:
#
#     name = value
#
# (The "=" is optional.) Whitespace may be used. Comments are introduced with
# "#" anywhere on a line. The complete list of parameter names and allowed
# values can be found in the PostgreSQL documentation.
#
# The commented-out settings shown in this file represent the default values.
# Re-commenting a setting is NOT sufficient to revert it to the default value;
# you need to reload the server.
#
# This file is read on server startup and when the server receives a SIGHUP
# signal. If you edit the file on a running system, you have to SIGHUP the
# server for the changes to take effect, run "pg_ctl reload", or execute
# "SELECT pg_reload_conf()". Some parameters, which are marked below,
# -----
[ Read 822 lines ]
^G Get Help      ^O WriteOut      ^R Read File      ^Y Prev Page      ^K Cut Text
^X Exit          ^J Justify       ^W Where Is       ^V Next Page      ^U Uncut Text
                                      ^C Cur Pos        ^T To Spell
```

- Cambiamos el puerto al puerto 4444:



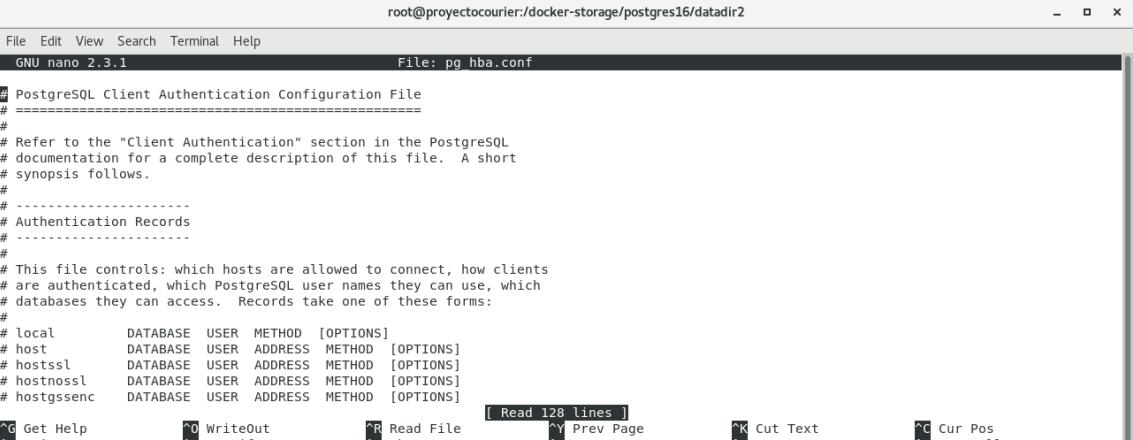
```
root@proyectocourier:/docker-storage/postgres16/datadir2
File Edit View Search Terminal Help
GNU nano 2.3.1          File: postgresql.conf
Modified
-----
# - Connection Settings -
listen_addresses = '*'          # comma-separated list of addresses;
                                 # defaults to 'localhost'; use '*' for all
                                 # (change requires restart)
port = 4444                      # (change requires restart)
max_connections = 100            # (change requires restart)
#reserved_connections = 0        # (change requires restart)
#superuser_reserved_connections = 3 # (change requires restart)
#unix_socket_directories = '/var/run/postgresql' # comma-separated list of directories
#unix_socket_group = ''          # (change requires restart)
#unix_socket_permissions = 0777   # begin with 0 to use octal notation
#bonjour = off                   # advertise server via Bonjour
#bonjour_name = ''               # defaults to the computer name
^G Get Help      ^O WriteOut      ^R Read File      ^Y Prev Page      ^K Cut Text
^X Exit          ^J Justify       ^W Where Is       ^V Next Page      ^U Uncut Text
                                      ^C Cur Pos        ^T To Spell
```

- Vamos a modificar el archivo: **nano pg_hba.conf**:



```
root@proyectocourier:/docker-storage/postgres16/datadir2
File Edit View Search Terminal Help
[root@proyectocourier datadir2]# nano pg_hba.conf
```

FACULTAD DE INGENIERÍA Y CIENCIAS APLICADAS

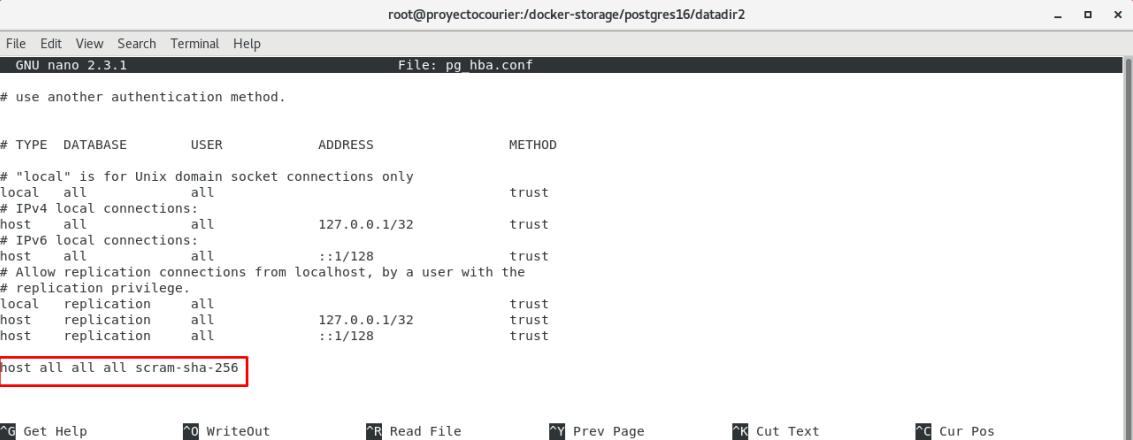


```
root@proyectocourier:/docker-storage/postgres16/datadir2
File Edit View Search Terminal Help
GNU nano 2.3.1          File: pg_hba.conf

# PostgreSQL Client Authentication Configuration File
# =====
# Refer to the "Client Authentication" section in the PostgreSQL
# documentation for a complete description of this file. A short
# synopsis follows.
#
# -----
# Authentication Records
# -----
#
# This file controls: which hosts are allowed to connect, how clients
# are authenticated, which PostgreSQL user names they can use, which
# databases they can access. Records take one of these forms:
#
# local      DATABASE  USER  METHOD [OPTIONS]
# host       DATABASE  USER  ADDRESS METHOD [OPTIONS]
# hostssl    DATABASE  USER  ADDRESS METHOD [OPTIONS]
# hostnossl  DATABASE  USER  ADDRESS METHOD [OPTIONS]
# hostgssenc DATABASE  USER  ADDRESS METHOD [OPTIONS]

[ Read 128 lines ]
^G Get Help      ^O WriteOut      ^R Read File      ^Y Prev Page      ^K Cut Text      ^C Cur Pos
^X Exit          ^J Justify       ^W Where Is       ^V Next Page      ^U Uncut Text     ^T To Spell
```

- Observamos que se encuentre correctamente la última línea y guardamos el archivo:



```
root@proyectocourier:/docker-storage/postgres16/datadir2
File Edit View Search Terminal Help
GNU nano 2.3.1          File: pg_hba.conf

# use another authentication method.

# TYPE   DATABASE      USER      ADDRESS            METHOD
# "local" is for Unix domain socket connections only
local   all            all                  trust
# IPv4 local connections:
host    all            all      127.0.0.1/32      trust
# IPv6 local connections:
host    all            all      ::1/128           trust
# Allow replication connections from localhost, by a user with the
# replication privilege.
local   replication   all                  trust
host   replication   all      127.0.0.1/32      trust
host   replication   all      ::1/128           trust

host all all all scram-sha-256

^G Get Help      ^O WriteOut      ^R Read File      ^Y Prev Page      ^K Cut Text      ^C Cur Pos
^X Exit          ^J Justify       ^W Where Is       ^V Next Page      ^U Uncut Text     ^T To Spell
```

- Detenemos la ejecución del contenedor y volvemos a ejecutarlo:



```
root@proyectocourier:/docker-storage/postgres16/datadir2
File Edit View Search Terminal Help
[root@proyectocourier datadir2]# docker stop pgjelv2
pgjelv2
[root@proyectocourier datadir2]# docker start pgjelv2
pgjelv2
[root@proyectocourier datadir2]# docker ps
CONTAINER ID        IMAGE               COMMAND             CREATED            STATUS              PORTS              NAMES
0eb7061aeab0        postgres            "docker-entrypoint.s..."   14 minutes ago   Up 12 seconds   5432/tcp          pgjelv2
[root@proyectocourier datadir2]#
```

- Ingresamos al **bash** del contenedor y como usuario **postgres** con el comando:
docker exec -it pgjelv2 bash



```
root@proyectocourier:/docker-storage/postgres16/datadir2
File Edit View Search Terminal Help
[root@proyectocourier datadir2]# docker exec -it pgjelv2 bash
root@0eb7061aeab0:#
```

- Nos va a aparecer un error por el puerto

```
root@proyectocourier:docker-storage/postgres16/datadir2
File Edit View Search Terminal Help
[root@proyectocourier datadir2]# docker exec -it pgjelv2 bash
root@0eb7061aeab0:/# su - postgres
postgres@0eb7061aeab0:~$ psql
psql: error: connection to server on socket "/var/run/postgresql/.s.PGSQL.5432" failed: No such file or directory
Is the server running locally and accepting connections on that socket?
postgres@0eb7061aeab0:~$
```

- Vamos a cambiar al nuevo puerto 4444, ya no nos aparecerá el error del puerto:

```
root@proyectocourier:docker-storage/postgres16/datadir2
File Edit View Search Terminal Help
[root@proyectocourier datadir2]# docker exec -it pgjelv2 bash
root@0eb7061aeab0:/# su - postgres
postgres@0eb7061aeab0:~$ psql
psql: error: connection to server on socket "/var/run/postgresql/.s.PGSQL.5432" failed: No such file or directory
Is the server running locally and accepting connections on that socket?
postgres@0eb7061aeab0:~$ psql -p 4444
psql (16.1 (Debian 16.1-1.pgdg120+1))
Type "help" for help.

postgres=#
```

- Ingresamos el comando **show port;** para ver el puerto cambiado:

```
root@proyectocourier:docker-storage/postgres16/datadir2
File Edit View Search Terminal Help
[root@proyectocourier datadir2]# docker exec -it pgjelv2 bash
root@0eb7061aeab0:/# su - postgres
postgres@0eb7061aeab0:~$ psql
psql: error: connection to server on socket "/var/run/postgresql/.s.PGSQL.5432" failed: No such file or directory
Is the server running locally and accepting connections on that socket?
postgres@0eb7061aeab0:~$ psql -p 4444
psql (16.1 (Debian 16.1-1.pgdg120+1))
Type "help" for help.

postgres=# show port;
port
-----
4444
(1 row)

postgres=#

```

- Ingresamos exit:

```
postgres=#
postgres=# exit
postgres@0eb7061aeab0:~$
```

- Nos conectamos como host con el comando **psql -h 0eb7061aeab0 -p 4444** al puerto **4444**: (con el container id 0eb7061aeab0)

```
postgres@0eb7061aeab0:~$ psql -h 0eb7061aeab0 -p 4444
Password for user postgres:
psql (16.1 (Debian 16.1-1.pgdg120+1))
Type "help" for help.

postgres=#

```

- Observamos que nos estamos conectando por TCP/IP y ya no por socket con el comando **\conninfo**:

```
root@proyectocourier:/docker-storage/postgres16/datadir2
File Edit View Search Terminal Help
postgres@0eb7061aeab0:~$ psql -h 0eb7061aeab0 -p 4444
Password for user postgres:
psql: error: connection to server at "0eb7061aeab0" (172.17.0.2), port 4444 failed: FATAL:  password authentication failed for user "postgres"
postgres@0eb7061aeab0:~$ psql -h 0eb7061aeab0 -p 4444
Password for user postgres:
psql (16.1 (Debian 16.1-1.pgdg120+1))
Type "help" for help.

postgres=\c
You are connected to database "postgres" as user "postgres" on host "0eb7061aeab0" (address "172.17.0.2") at port "4444".
postgres=#

```

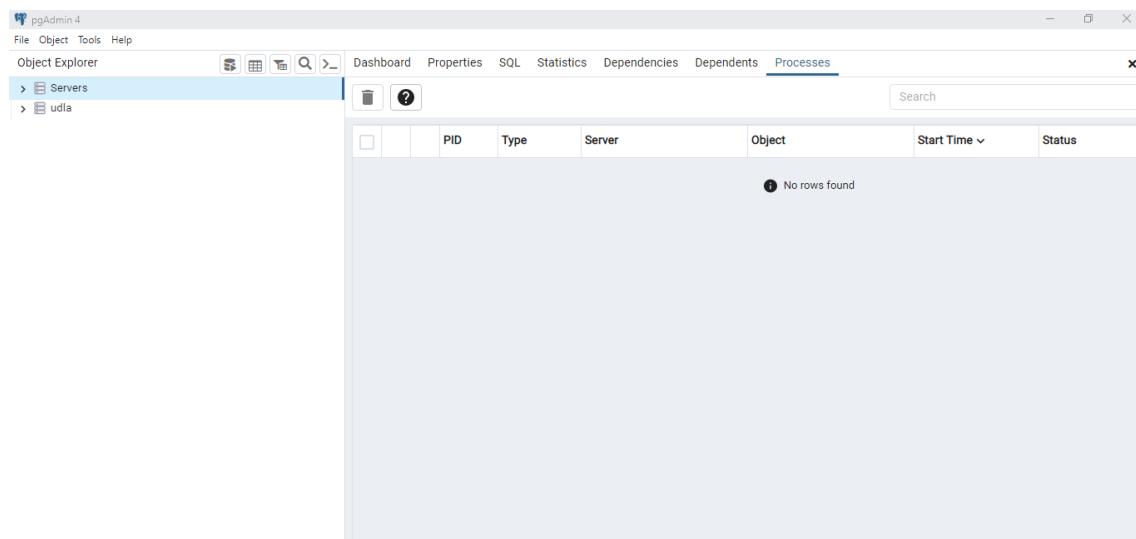
- Vamos a detener la ejecución del contenedor para correr con el puerto 15532:4444:
- **docker run --name pgjelv2 -p 15532:4444 -e POSTGRES_PASSWORD=postgres -v /docker-storage/postgres16/datadir2:/var/lib/postgresql/data -d postgres**

```
[root@proyectocourier datadir2]# docker stop pgjelv2
pgjelv2
[root@proyectocourier datadir2]# docker rm pgjelv2
pgjelv2
[root@proyectocourier datadir2]# docker run --name pgjelv -p 15532:4444 -e POSTGRES_PASSWORD=postgres -v /docker-storage/postgres15/datadir:/var/lib/postgresql/data -d postgres
docker: Error response from daemon: Conflict. The container name "/pgjelv" is already in use by container "cfba907c262f084a6e58d3a94ac1646c45dff73835f346291bc3e86743bcfdal". You have to remove (or rename) that container to be able to reuse that name.
See 'docker run --help'.
[root@proyectocourier datadir2]# docker run --name pgjelv2 -p 15532:4444 -e POSTGRES_PASSWORD=postgres -v /docker-storage/postgres16/datadir2:/var/lib/postgresql/data -d postgres
93f649b360985a0f6e86e01d89bf3a383c041ef06b44ec9eb1fa52399770ff9
[root@proyectocourier datadir2]#

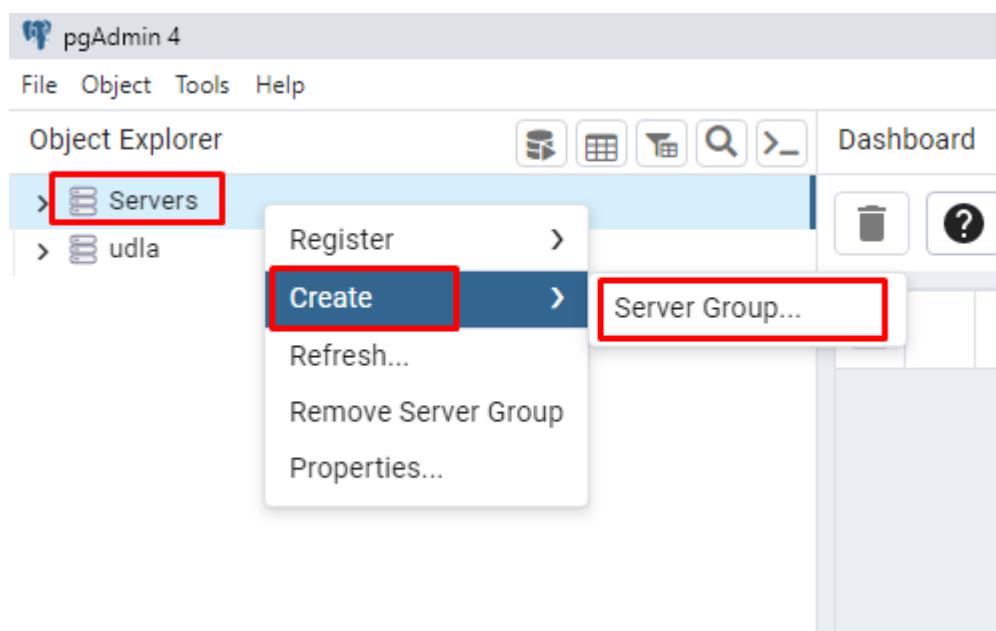
```

5. Conexión de contenedor de Postgresql con PGAdmin 4 en Windows:

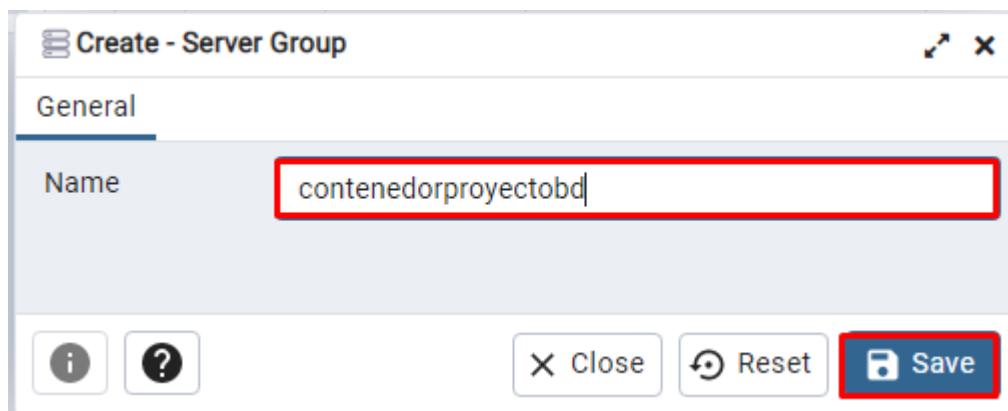
- Abrir PGAdmin 4:



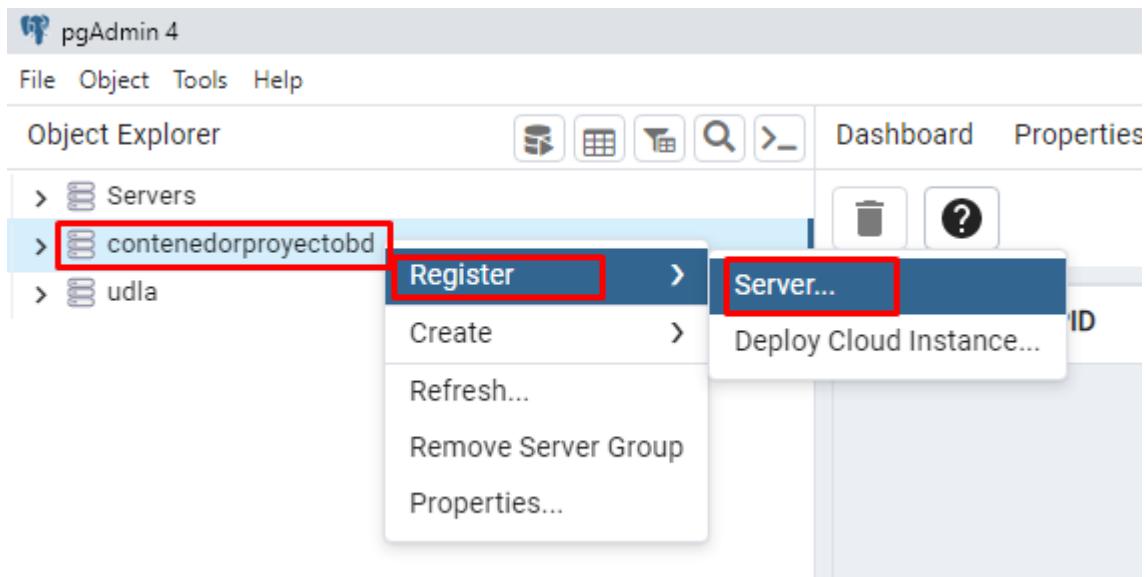
- Crear un nuevo "Server Group", damos clic en la opción **Servers -> Create -> Server Group**



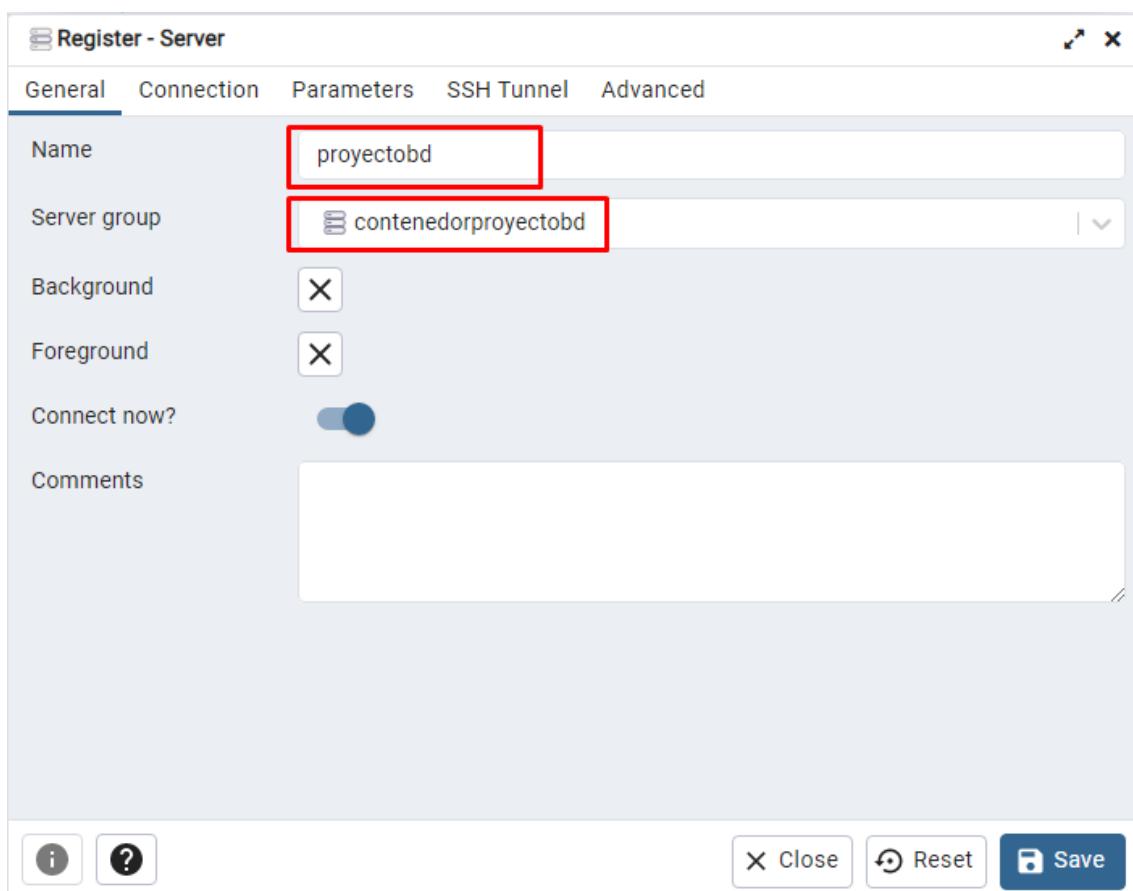
- Ingresamos el nombre del Server Group: "**contenedor proyecto bd**" y clic en el botón "Save"



- Vamos a registrar un nuevo servidor en el Server Group "contenedor proyecto bd", damos clic derecho en "contenedor proyecto bd" -> Register -> Server



- Ingresamos el nombre del servidor: **proyectobd**



- Damos clic en la pestaña "Connections" e ingresamos:

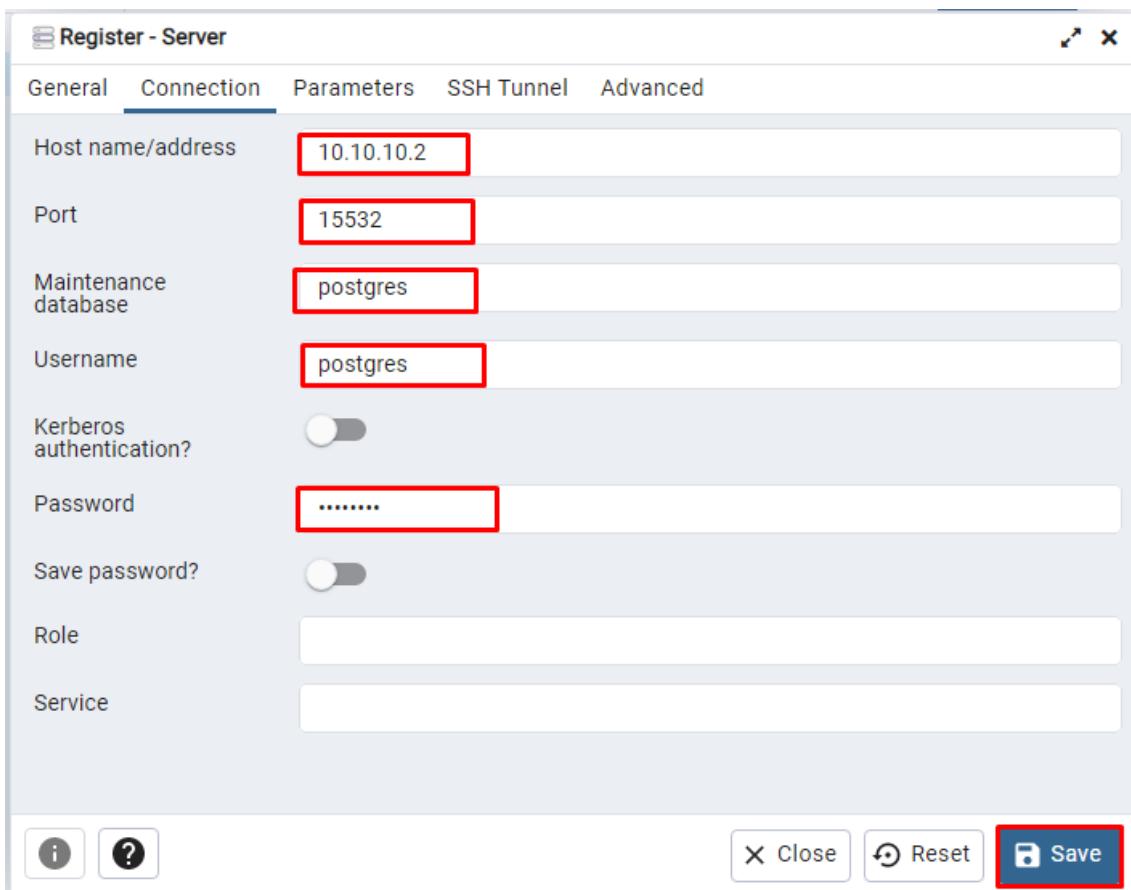
Host name/address: 10.10.10.2

Port: 15532

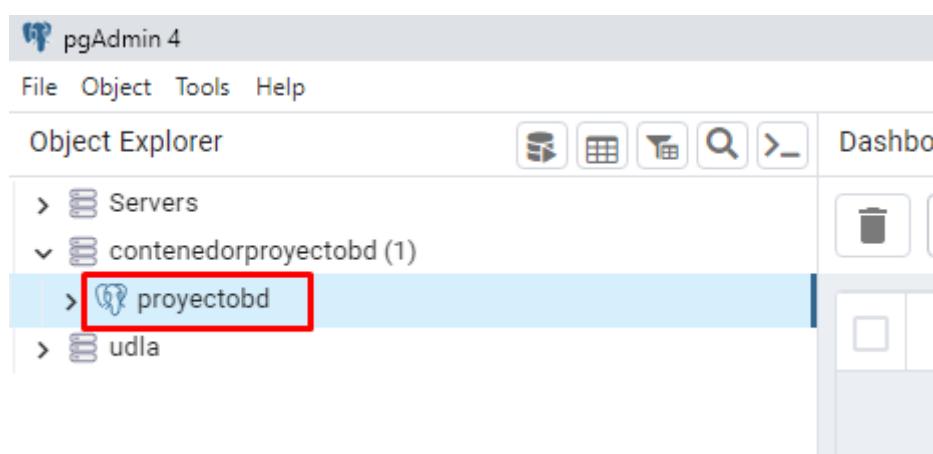
Maintance database: postgres

Username: postgres

Pssword: postgres



- Observamos que nos parece el servidor ya registrado y establecido conexión con el contenedor de Postgresql:



- Observamos le estatus del firewall de la maquina virtual: **systemctl status firewalld**

```
root@proyectocourier:docker-storage/postgres16/datadir2
File Edit View Search Terminal Help
[root@proyectocourier datadir2]# systemctl status firewalld
● firewalld.service - firewalld - dynamic firewall daemon
   Loaded: loaded (/usr/lib/systemd/system/firewalld.service; disabled; vendor preset: enabled)
     Active: active (running) since Fri 2024-01-12 13:14:46 -05; 4s ago
       Docs: man:firewalld(1)
   Main PID: 28225 (firewalld)
      Tasks: 2
     Memory: 25.9M
        CGroup: /system.slice/firewalld.service
           └─28225 /usr/bin/python -Es /usr/sbin/firewalld --nofork --nopid

Jan 12 13:14:45 proyectocourier.lpmb.com systemd[1]: Starting firewalld - dynamic firewall daemon...
Jan 12 13:14:46 proyectocourier.lpmb.com systemd[1]: Started firewalld - dynamic firewall daemon.
Jan 12 13:14:48 proyectocourier.lpmb.com firewalld[28225]: WARNING: ip6tables not usable, disabling IPv6 firewall.
[root@proyectocourier datadir2]#
```

- Vamos a bajar el firewall de la máquina virtual:

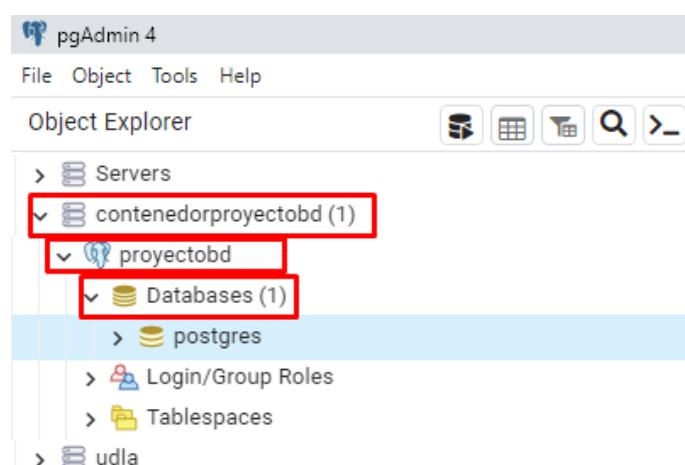
```
[root@proyectocourier datadir2]# systemctl stop firewalld
[root@proyectocourier datadir2]# systemctl disable firewalld
[root@proyectocourier datadir2]# systemctl status firewalld
● firewalld.service - firewalld - dynamic firewall daemon
   Loaded: loaded (/usr/lib/systemd/system/firewalld.service; disabled; vendor preset: enabled)
     Active: inactive (dead) since Fri 2024-01-12 13:33:16 -05; 9s ago
       Docs: man:firewalld(1)
   Main PID: 28225 (code=exited, status=0/SUCCESS)

Jan 12 13:14:45 proyectocourier.lpmb.com systemd[1]: Starting firewalld - dynamic firewall daemon...
Jan 12 13:14:46 proyectocourier.lpmb.com systemd[1]: Started firewalld - dynamic firewall daemon.
Jan 12 13:14:48 proyectocourier.lpmb.com firewalld[28225]: WARNING: ip6tables not usable, disabling IPv6 firewall.
Jan 12 13:33:14 proyectocourier.lpmb.com systemd[1]: Stopping firewalld - dynamic firewall daemon...
Jan 12 13:33:16 proyectocourier.lpmb.com systemd[1]: Stopped firewalld - dynamic firewall daemon.
[root@proyectocourier datadir2]#
```

- Observamos la conexiones activas: **netstat -ant**

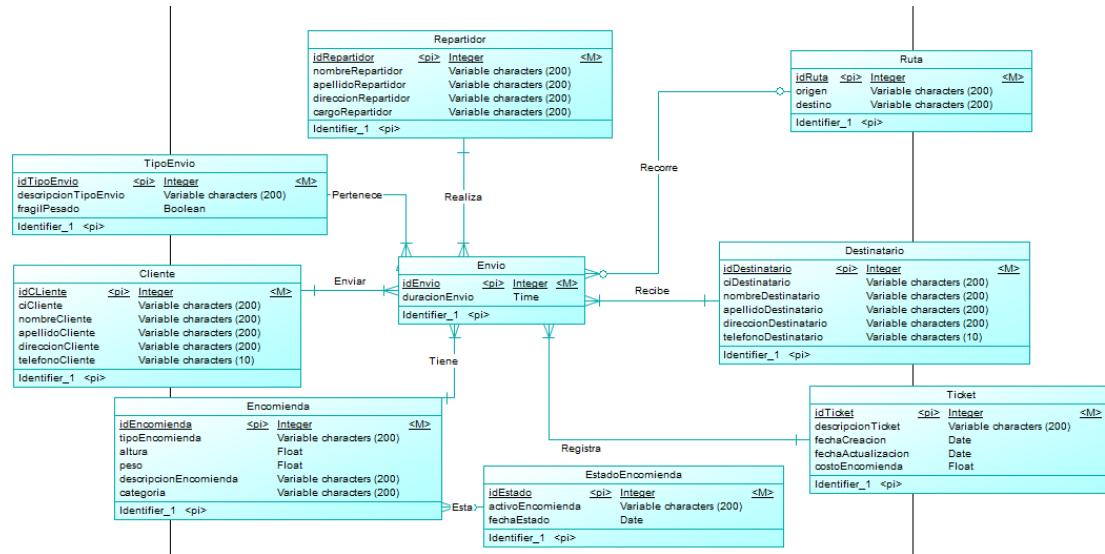
```
[root@proyectocourier datadir2]#
[root@proyectocourier datadir2]# netstat -ant
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address          Foreign Address        State
tcp        0      0 0.0.0.0:15532          0.0.0.0:*
tcp        0      0 0.0.0.0:111           0.0.0.0:*
tcp        0      0 0.0.0.0:6000          0.0.0.0:*
tcp        0      0 192.168.122.1:53        0.0.0.0:*
tcp        0      0 0.0.0.0:22           0.0.0.0:*
tcp        0      0 127.0.0.1:631          0.0.0.0:*
tcp        0      0 127.0.0.1:25           0.0.0.0:*
```

- Validamos que se realizó la conexión correcta con PGADMIN:

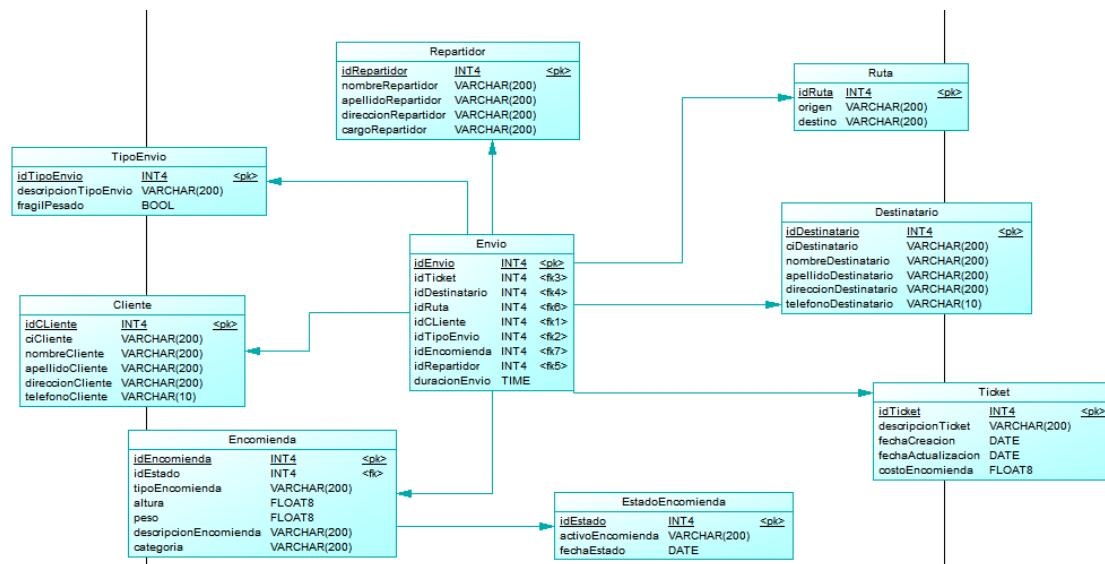


6. Modelado de la Base de Datos bdcourier

Modelo lógico:

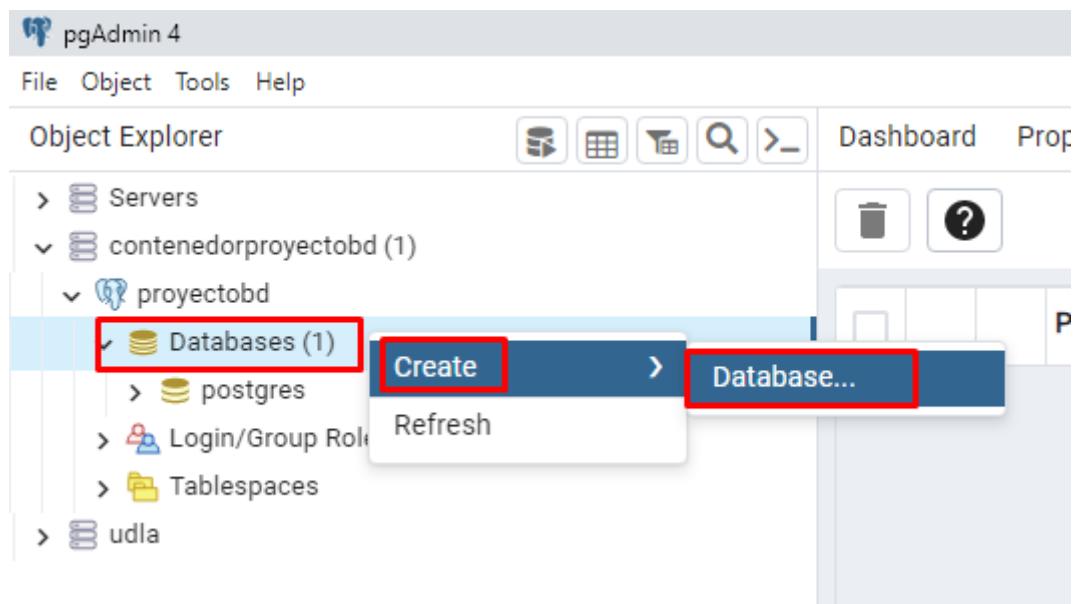


Modelo físico:

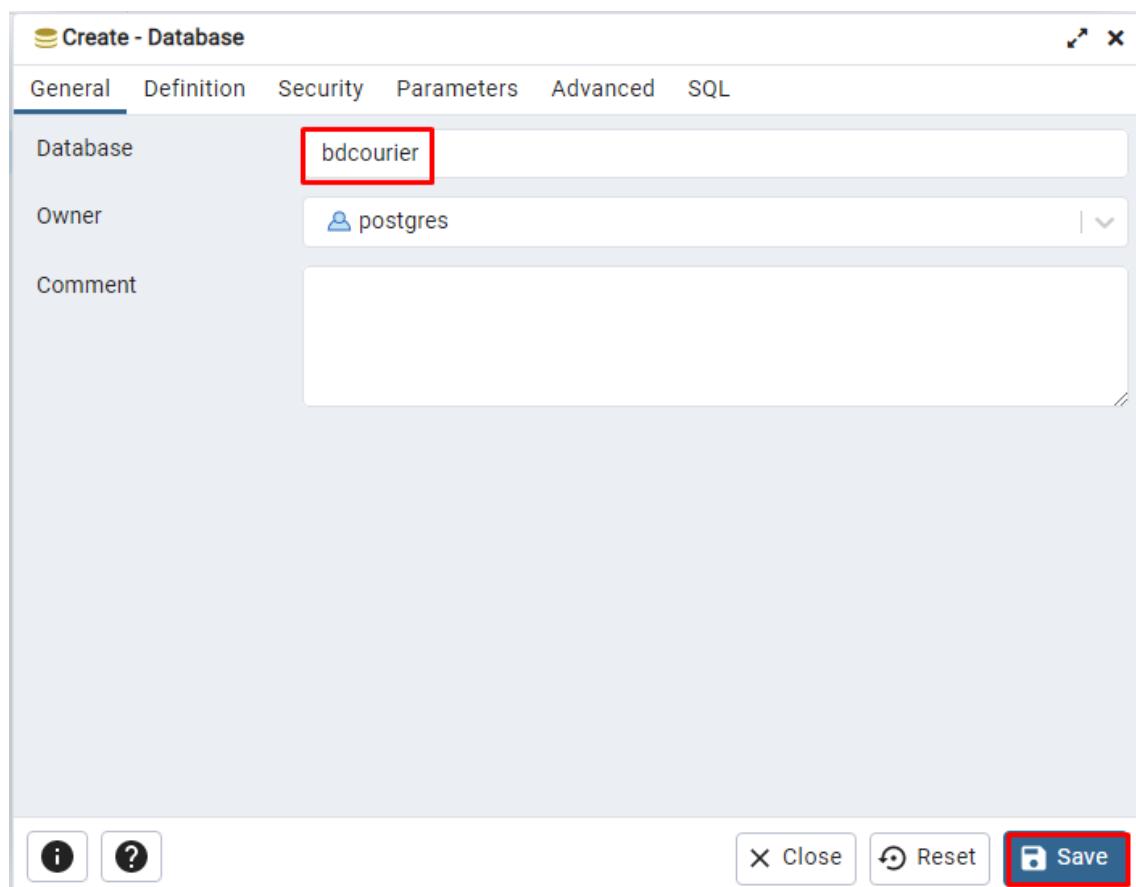


7. Creación de Base de Datos bdcourier en PGAdmin 4

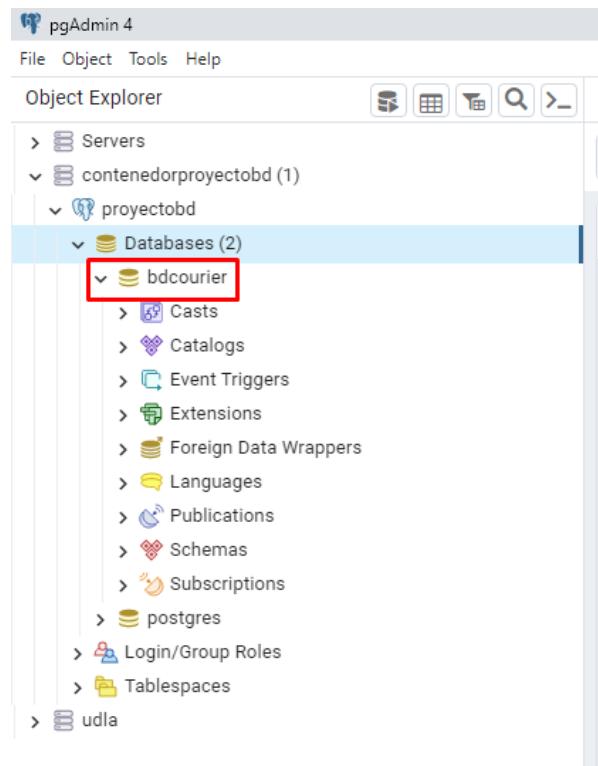
- Desde PGAdmin 4 ir a Databases -> Create -> Database



- Ingresar el nombre de la base de datos "**bdcourier**" y dar clic en "**Save**":



- Observamos que ya se encuentra creada la base da datos "bdcourier" en PGAdmin 4:



- Desde el contenedor de postgresql también observamos que se encuentra creada la base de datos.

Nos conectamos al contenedor: (Password: postgres)

```
root@proyectocourier:/docker-storage/postgres16/datadir2
File Edit View Search Terminal Help
[root@proyectocourier datadir2]# docker ps
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES
93f649b36098 postgres "docker-entrypoint.s..." 45 minutes ago Up 45 minutes 5432/tcp, 0.0.0.0:15532->4444/tcp pgjelv2
[root@proyectocourier datadir2]# docker exec -it pgjelv2 bash
root@93f649b36098:/# su - postgres
postgres@93f649b36098:~$ psql -h 93f649b36098 -p 4444
Password for user postgres:
psql (16.1 (Debian 16.1-1.pgdg120+1))
Type "help" for help.

postgres=#

```

- Observamos las bases de datos creadas en el contenedor de Postgresql con el comando: \l
- Y observamos la base de datos "bdcourier":

```
[root@proyectocourier datadir2]# docker ps
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES
93f649b36098 postgres "docker-entrypoint.s..." 45 minutes ago Up 45 minutes 5432/tcp, 0.0.0.0:15532->4444/tcp pgjelv2
[root@proyectocourier datadir2]# docker exec -it pgjelv2 bash
root@93f649b36098:/# su - postgres
postgres@93f649b36098:~$ psql -h 93f649b36098 -p 4444
Password for user postgres:
psql (16.1 (Debian 16.1-1.pgdg120+1))
Type "help" for help.

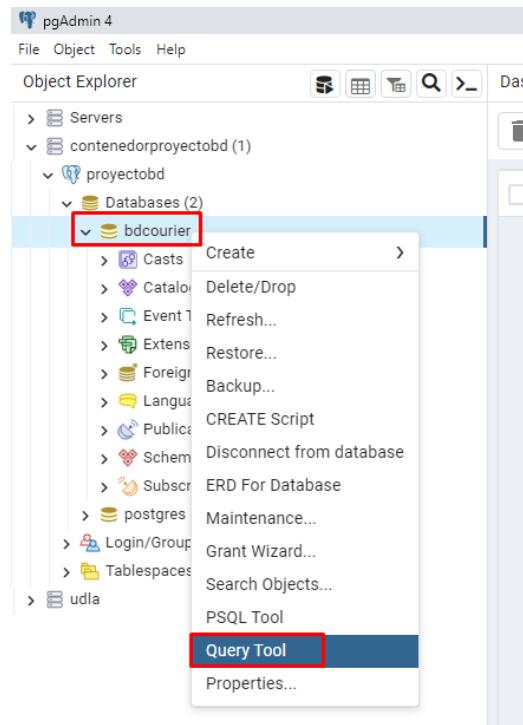
postgres=# \l
                                         List of databases
   Name    | Owner | Encoding | Locale Provider | Collate | Ctype | ICU Locale | ICU Rules | Access privileges
   bdcourier | postgres | UTF8 | libc | en_US.utf8 | en_US.utf8 |
   postgres | postgres | UTF8 | libc | en_US.utf8 | en_US.utf8 |
 template0 | postgres | UTF8 | libc | en_US.utf8 | en_US.utf8 |
 template1 | postgres | UTF8 | libc | en_US.utf8 | en_US.utf8 |
(4 rows)

postgres=#

```

8. Crear tablas en base de datos “bdcourier”

- Desde PGAdmin 4 vamos a ingresar en “bdcourier” -> “Query Tool”



- Ingresamos el script de la base de datos modelada:

FACULTAD DE INGENIERÍA Y CIENCIAS APLICADAS

```

58
59     /*=====
60     * Table: CLIENTE
61     *=====
62     create table CLIENTE (
63         IDCLIENTE      INT4          not null,
64         CILIENTE       VARCHAR(200)   null,
65         NOMBRECLIENTE  VARCHAR(200)   null,
66         APELLIDOCLIENTE VARCHAR(200)   null,
67         DIRECCIONCLIENTE VARCHAR(200)   null,
68         TELEFONOCLIENTE VARCHAR(10)    null,
69         constraint PK_CLIENTE primary key (IDCLIENTE)
70     );
71
72     /*=====
73     * Index: CLIENTE_PK
74     *=====
75     create unique index CLIENTE_PK on CLIENTE (
76         IDCLIENTE
77     );
78
79     /*=====
80     * Table: DESTINATARIO
81     *=====
82     create table DESTINATARIO (
83         IDESTINATARIO   INT4          not null,
84         CIDESTINATARIO  VARCHAR(200)   null,
85         NOMBREDESTINATARIO VARCHAR(200)   null,
86         APELLIDODESTINATARIO VARCHAR(200)   null,
87         DIRECCIONDESTINATARIO VARCHAR(200)   null,
88         TELEFONODESTINATARIO VARCHAR(10)    null,
89         constraint PK_DESTINATARIO primary key (IDDESTINATARIO)
90     );
91
92     /*=====
93     * Index: DESTINATARIO_PK
94     *=====
95     create unique index DESTINATARIO_PK on DESTINATARIO (

```

- Damos clic en el botón “Play” para que se ejecuten las sentencias SQL de creación e las tablas del modelado dentro de la base de datos “bdcourier”:

The screenshot shows the pgAdmin 4 interface with the following details:

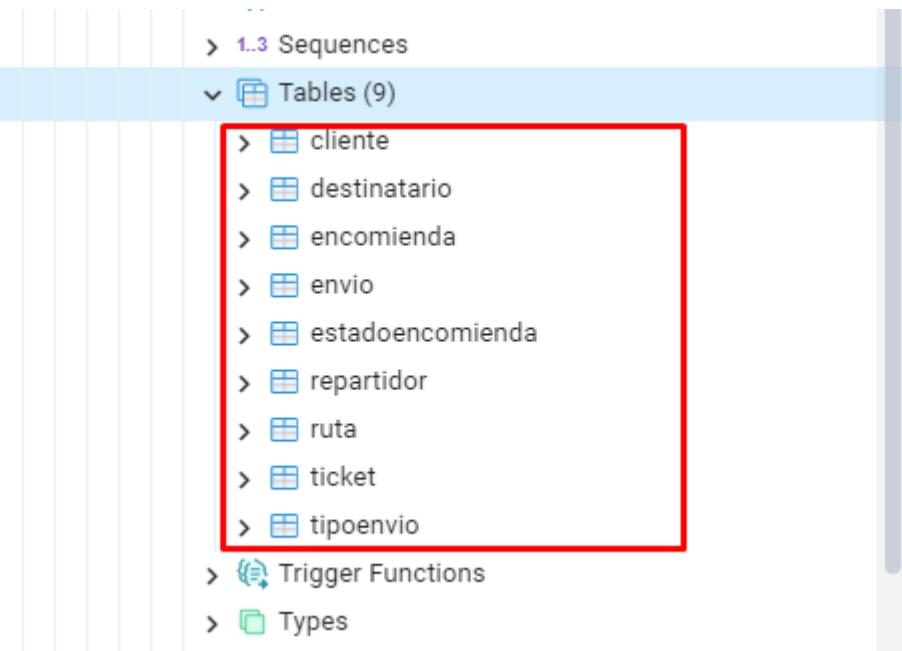
- File Object Tools Help**: Standard application menu.
- Object Explorer**: Shows the database structure. Under the **contenedorproyectobd** schema, there is a **projectobd** database which contains the **bdcourier** schema. The **bdcourier** schema is selected.
- Dashboard Properties SQL Statistics Dependencies Dependents Processes**: Top navigation bar for the current connection.
- bdcourier/postgres@proyectobd**: Connection information.
- Query History**: A list of previous queries.
- Query**: The main query editor window containing the SQL code from the previous screenshot. The entire code block is highlighted with a red rectangle.
- Buttons**: A toolbar with various icons, including a play button which is highlighted with a red box.
- Data Output Messages Notifications**: Bottom navigation tabs for the query results.

- Observamos que se ha ejecutado correctamente el query:

The screenshot shows the pgAdmin interface with the 'Messages' tab selected. The message content is:

```
ALTER TABLE
Query returned successfully in 590 msec.
```

- Visualizamos que se han creado correctamente las tablas en la base de datos "bcdcourier":



- Conexión desde PgAdmin instalado en Windows y desde el cual se pueda consultar la tabla cliente:
- Salimos de la conexión actual con exit:

```
postgres=# exit
postgres@93f649b36098:~$ exit
logout
root@93f649b36098:/# exit
exit
[root@proyectocourier datadir2]#
[root@proyectocourier datadir2]#
```

- Ingresamos desde la máquina virtual como usuario postgres por el puerto 4444 y a la base de datos bdcourier:

```
docker exec -it pgjelv2 bash  
su - postgres  
psql -p 4444 -d bdcourier
```

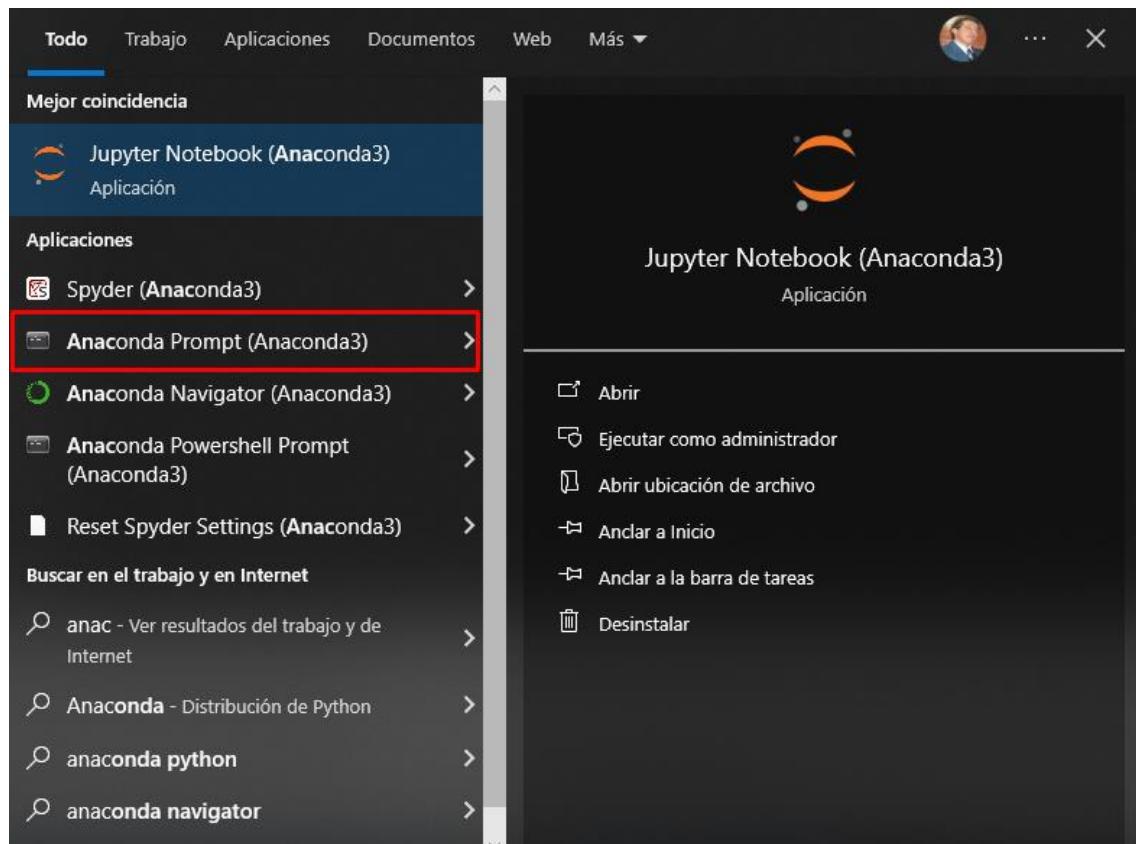
```
root@proyectocourier:/docke.../postgres16/datadir2  
File Edit View Search Terminal Help  
[root@proyectocourier datadir2]# docker exec -it pgjelv2 bash  
root@93f649b36098:/# su - postgres  
postgres@93f649b36098:~$ psql -p 4444 -d bdcourier  
psql (16.1 (Debian 16.1-1.pgdg120+1))  
Type "help" for help.  
  
bdcourier=#
```

- Realizamos un select para consultar los datos de la base de datos bdcourier y validar la conexión:

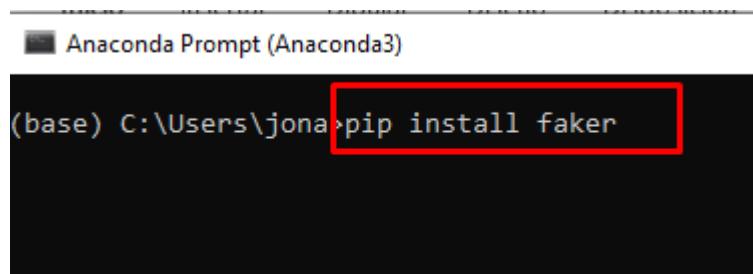
```
root@proyectocourier:/docke.../postgres16/datadir2  
File Edit View Search Terminal Help  
[root@proyectocourier datadir2]# docker exec -it pgjelv2 bash  
root@93f649b36098:/# su - postgres  
postgres@93f649b36098:~$ psql -p 4444 -d bdcourier  
psql (16.1 (Debian 16.1-1.pgdg120+1))  
Type "help" for help.  
  
bdcourier=# select * from cliente;  
 idcliente | cicliente | nombreciente | apellidocliente | direccioncliente | telefonocliente  
-----+-----+-----+-----+-----+-----  
(0 rows)  
bdcourier=#[
```

9. Poblar la Base de datos “bdcourier” con Faker

- Abrimos la consola de Anaconda:



- En la consola de Anaconda ingresamos el comando para instalar faker: **pip install faker**

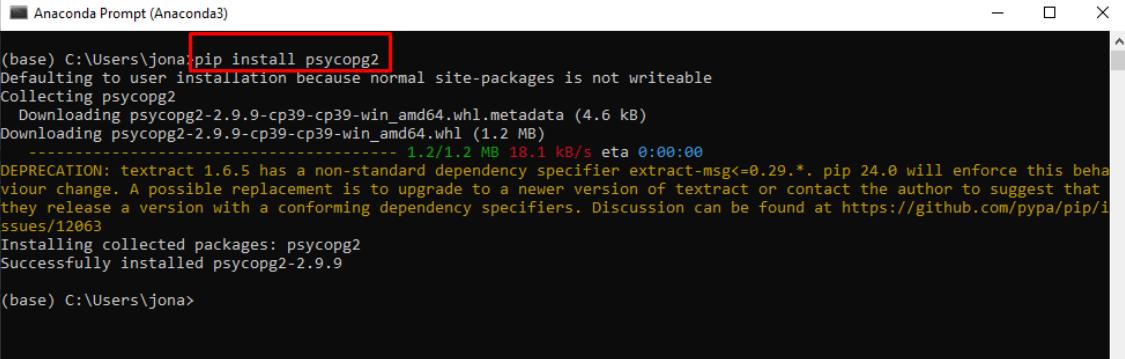


- Esperamos a que se realice la descarga e instalación del módulo "Faker" en Python

```
(base) C:\Users\jona>pip install faker
Defaulting to user installation because normal site-packages is not writeable
Collecting faker
  Downloading Faker-22.2.0-py3-none-any.whl.metadata (15 kB)
Requirement already satisfied: python-dateutil>=2.4 in c:\programdata\anaconda3\lib\site-packages (from faker) (2.8.2)
Requirement already satisfied: six>=1.5 in c:\programdata\anaconda3\lib\site-packages (from python-dateutil>=2.4->faker) (1.12.0)
  Downloading Faker-22.2.0-py3-none-any.whl (1.7 MB)
----- 1.7/1.7 MB 21.0 kB/s eta 0:00:00
DEPRECATION: textextract 1.6.5 has a non-standard dependency specifier extract-msg<=0.29.*. pip 24.0 will enforce this behaviour change. A possible replacement is to upgrade to a newer version of textextract or contact the author to suggest that they release a version with a conforming dependency specifiers. Discussion can be found at https://github.com/pypa/pip/issues/12063
Installing collected packages: faker
  WARNING: The script faker.exe is installed in 'C:\Users\jona\AppData\Roaming\Python\Python39\Scripts' which is not on PATH.
    Consider adding this directory to PATH or, if you prefer to suppress this warning, use --no-warn-script-location.
Successfully installed faker-22.2.0
(base) C:\Users\jona>
```

- En la consola de Anaconda ingresamos el comando para instalar faker: **pip install**

faker psycopg2



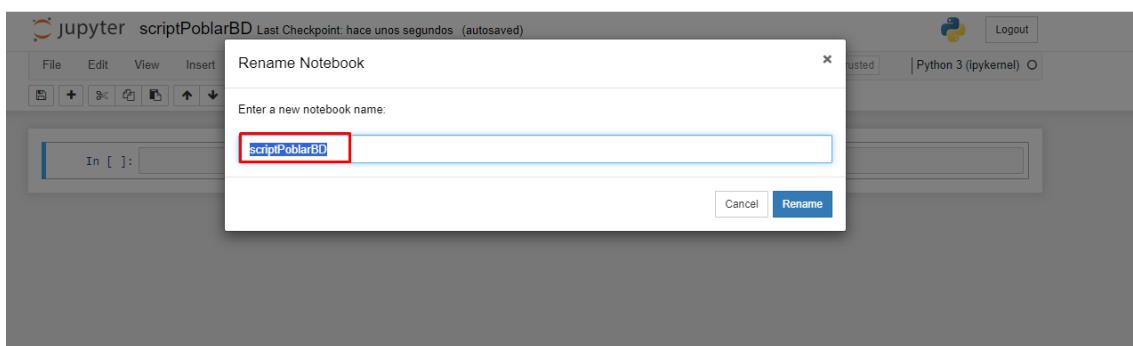
```
(base) C:\Users\jona>pip install psycopg2
Defaulting to user installation because normal site-packages is not writeable
Collecting psycopg2
  Downloading psycopg2-2.9.9-cp39-cp39-win_amd64.whl.metadata (4.6 kB)
  Downloading psycopg2-2.9.9-cp39-cp39-win_amd64.whl (1.2 MB)
    1.2/1.2 MB 18.1 kB/s eta 0:00:00
DEPRECATION: textract 1.6.5 has a non-standard dependency specifier extract-msg<=0.29.*. pip 24.0 will enforce this behaviour change. A possible replacement is to upgrade to a newer version of textract or contact the author to suggest that they release a version with a conforming dependency specifiers. Discussion can be found at https://github.com/pypa/pip/issues/12063
Installing collected packages: psycopg2
Successfully installed psycopg2-2.9.9

(base) C:\Users\jona>
```

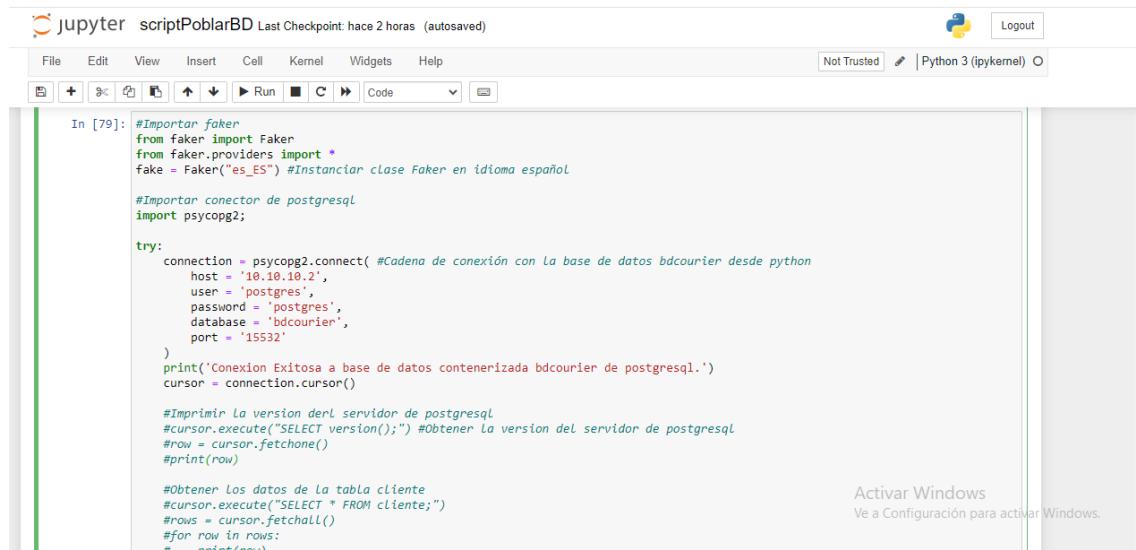
- Abrimos Jupyter Notebook



- Crear un script llamado "scriptPoblarBD.py"



- Creamos el script para la inserción de datos en las tablas utilizando la librería de Python Faker:



```

jupyter scriptPoblarBD Last Checkpoint: hace 2 horas (autosaved)

File Edit View Insert Cell Kernel Widgets Help
Not Trusted | Python 3 (ipykernel) O

In [79]: #Importar faker
from faker import Faker
from faker.providers import *
fake = Faker("es_ES") #Instanciar clase Faker en idioma español

#Importar conector de postgresql
import psycopg2;

try:
    connection = psycopg2.connect( #Cadena de conexión con la base de datos bdcourier desde python
        host = '10.10.10.2',
        user = 'postgres',
        password = 'postgres',
        database = 'bdcourier',
        port = '15532'
    )
    print('Conexion Exitosa a la base de datos contenerizada bdcourier de postgresql.')
    cursor = connection.cursor()

    #Imprimir La version del servidor de postgresql
    #cursor.execute("SELECT version();") #Obtener la version del servidor de postgresql
    #row = cursor.fetchone()
    #print(row)

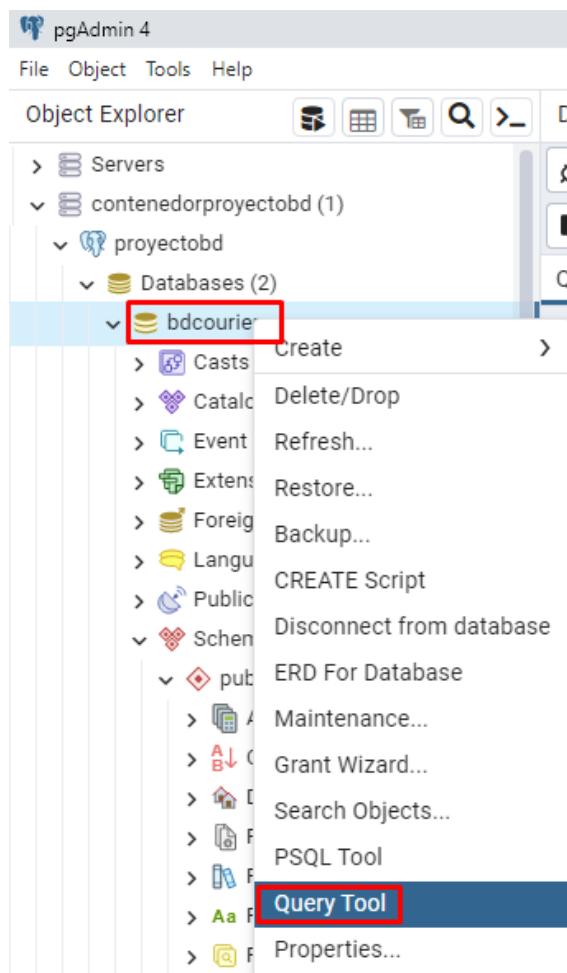
    #Obtener los datos de la tabla cliente
    #cursor.execute("SELECT * FROM cliente;")
    #rows = cursor.fetchall()
    #for row in rows:
    #    print(row)

```

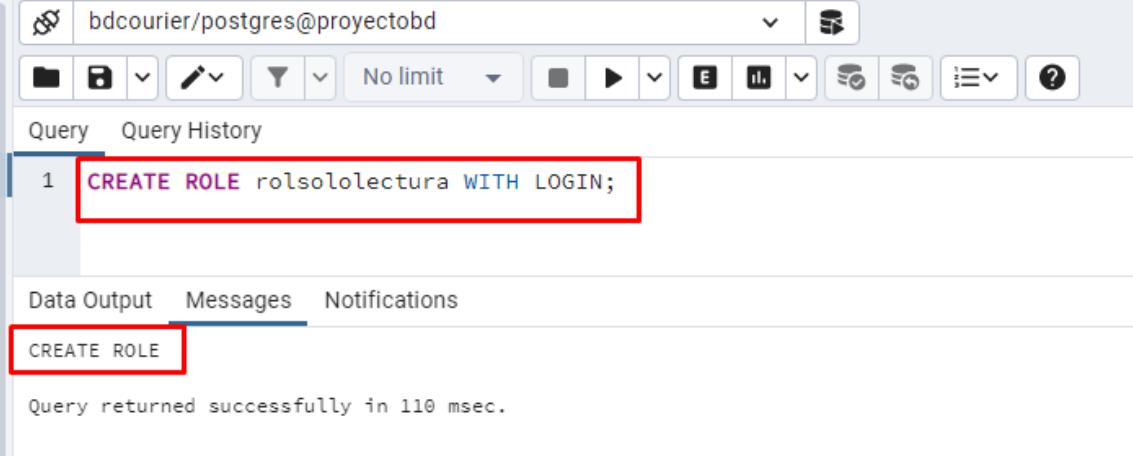
Activar Windows
Ve a Configuración para activar Windows.

10. Creación de usuarios roles

- Desde Pgadmin ingresamos a “bdcourier” -> “Query Tool” para ingresar las sentencias sql para crear los roles y usuarios

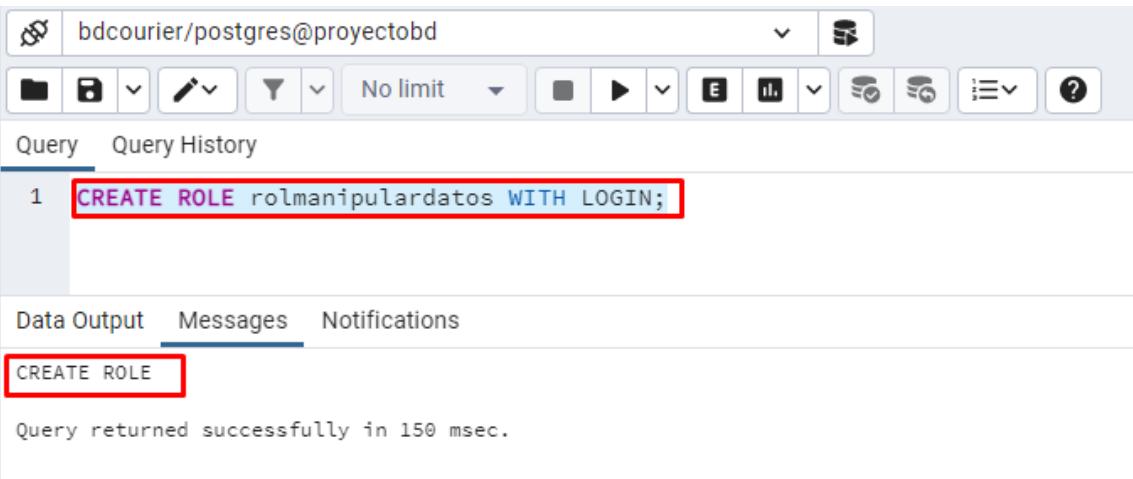


- Creamos el primer role llamado "rolsololectura":
CREATE ROLE rolsolectura WITH LOGIN;



The screenshot shows the pgAdmin 4 interface with the connection set to 'bdcourier/postgres@proyectobd'. In the 'Query' tab, the command **CREATE ROLE rolsolectura WITH LOGIN;** is entered and highlighted with a red box. Below the query, the status bar shows 'CREATE ROLE' and the message 'Query returned successfully in 110 msec.'

- Creamos el segundo rol llamado "rolmanipulardatos":
CREATE ROLE rolmanipulardatos WITH LOGIN;



The screenshot shows the pgAdmin 4 interface with the connection set to 'bdcourier/postgres@proyectobd'. In the 'Query' tab, the command **CREATE ROLE rolmanipulardatos WITH LOGIN;** is entered and highlighted with a red box. Below the query, the status bar shows 'CREATE ROLE' and the message 'Query returned successfully in 150 msec.'

- Creamos el primer usuario llamado "usuariolectura":
CREATE USER usuariolectura WITH PASSWORD 'userlectura1';

The screenshot shows the pgAdmin 4 interface. The connection is set to 'bdcourier/postgres@proyectobd'. The query editor has a single line of code: 'CREATE USER usuariolectura WITH PASSWORD 'userlectural';'. This line is highlighted with a red rectangle. Below the editor, the status bar shows 'Data Output' is selected, followed by 'Messages' and 'Notifications'. The messages pane displays the command 'CREATE ROLE' and the message 'Query returned successfully in 74 msec.'

- Creamos el segundo usuario llamado "**usuariomanipulardatos**":
CREATE USER usuariomanipulardatos WITH PASSWORD 'usermanipulardatos2';

The screenshot shows the pgAdmin 4 interface. The connection is set to 'bdcourier/postgres@proyectobd'. The query editor has a single line of code: 'CREATE USER usuariomanipulardatos WITH PASSWORD 'usermanipulardatos2';'. This line is highlighted with a red rectangle. Below the editor, the status bar shows 'Data Output' is selected, followed by 'Messages' and 'Notifications'. The messages pane displays the command 'CREATE ROLE' and the message 'Query returned successfully in 157 msec.'

- Concedemos el primer rol "**rolsololectura**" al primer usuario "**usuariolectura**":
GRANT rolsololectura TO usuariolectura;

The screenshot shows the pgAdmin 4 interface. The connection is set to 'bdcourier/postgres@proyectobd'. The query editor has a single line of code: 'GRANT rolsololectura TO usuariolectura;'. This line is highlighted with a red rectangle. Below the editor, the status bar shows 'Data Output' is selected, followed by 'Messages' and 'Notifications'. The messages pane displays the command 'GRANT ROLE' and the message 'Query returned successfully in 140 msec.'

- Concedemos el segundo rol "**rolmanipulardatos**" al segundo usuario "**usuariomanipulardatos**":
GRANT rolmanipulardatos TO usuariomanipulardatos;

The screenshot shows a pgAdmin 4 interface. The connection is set to 'bdcourier/postgres@proyectobd'. The query tab contains the command: 'GRANT rolmanipulardatos TO usuariomanipulardatos;'. The message tab shows the result: 'GRANT ROLE' followed by 'Query returned successfully in 155 msec.'.

```
1 GRANT rolmanipulardatos TO usuariomanipulardatos;
```

GRANT ROLE

Query returned successfully in 155 msec.

- Asignamos los privilegios de solo lectura al rol "**rolsololectura**":

```
DO $$  
DECLARE  
    r RECORD;  
BEGIN  
    FOR r IN SELECT tablename FROM pg_tables WHERE schemaname = 'public'  
    LOOP  
        EXECUTE 'GRANT SELECT ON ' || quote_ident(r.tablename) || ' TO rolsololectura';  
    END LOOP;  
END  
$$;
```

The screenshot shows a pgAdmin 4 interface. The connection is set to 'bdcourier/postgres@proyectobd'. The query tab contains a DO block with code to grant select privileges. The message tab shows the result: 'DO' followed by 'Query returned successfully in 334 msec.'

```
1 DO $$  
2 DECLARE  
3     r RECORD;  
4 BEGIN  
5     FOR r IN SELECT tablename FROM pg_tables WHERE schemaname = 'public'  
6     LOOP  
7         EXECUTE 'GRANT SELECT ON ' || quote_ident(r.tablename) || ' TO rolsololectura';  
8     END LOOP;  
9 END  
10 $$;  
11
```

DO

Query returned successfully in 334 msec.

- Asignamos los privilegios de solo manipulación de datos al rol "**rolmanipulardatos**":

```
DO $$  
DECLARE  
    r RECORD;  
BEGIN  
    FOR r IN SELECT tablename FROM pg_tables WHERE schemaname = 'public'
```

```


LOOP
  EXECUTE 'GRANT SELECT, INSERT, UPDATE, DELETE ON ' || quote_ident(r.tablename) || '
TO rolmanipulardatos;
END LOOP;
END
$$;

DO $$
DECLARE
  r RECORD;
BEGIN
  FOR r IN SELECT sequence_name FROM information_schema.sequences WHERE
sequence_schema = 'public'
  LOOP
    EXECUTE 'GRANT USAGE, SELECT ON SEQUENCE public.' || 
quote_ident(r.sequence_name) || ' TO rolmanipulardatos;
  END LOOP;
END
$$;


```

The screenshot shows the pgAdmin III interface with a query editor window. The query is a PL/pgSQL script that grants permissions on tables and sequences to the 'rolmanipulardatos' role. The code is highlighted with a red rectangle. The status bar at the bottom right shows a green message: 'Query returned successfully in 199 msec.'

```


1 DO $$ 
2 DECLARE
3   r RECORD;
4 BEGIN
5   FOR r IN SELECT tablename FROM pg_tables WHERE schemaname = 'public'
6   LOOP
7     EXECUTE 'GRANT SELECT, INSERT, UPDATE, DELETE ON ' || quote_ident(r.tablename) || ' TO rolmanipulardatos';
8   END LOOP;
9 END
10 $$;
11 
12 DO $$ 
13 DECLARE
14   r RECORD;
15 BEGIN
16   FOR r IN SELECT sequence_name FROM information_schema.sequences WHERE sequence_schema = 'public'
17   LOOP
18     EXECUTE 'GRANT USAGE, SELECT ON SEQUENCE public.' || quote_ident(r.sequence_name) || ' TO rolmanipulardatos;
19   END LOOP;
20 END
21 $$;
~~


```

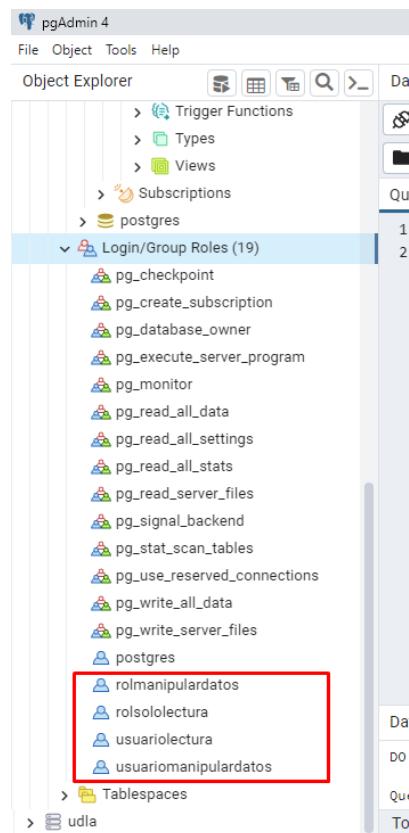
- Observamos los usuarios y roles existentes en la base de datos:

\du+

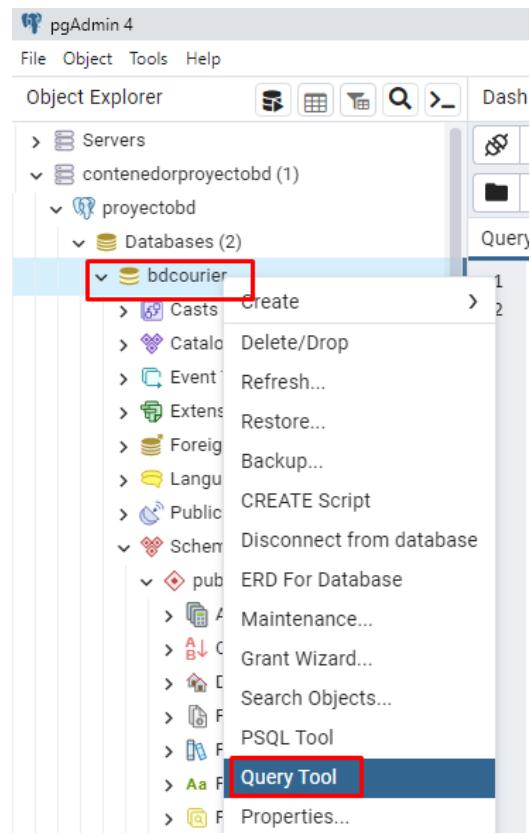
```
root@proyectocourier:/docker-storage/postgres16/datadir2
File Edit View Search Terminal Help
bdcourier=# \du+
                                         List of roles
      Role name           Attributes            | Description
-----+-----+-----+
postgres          | Superuser, Create role, Create DB, Replication, Bypass RLS |
rolmanipulardatos | |
rolsololectura    | |
usuariolectura    | |
usuariolectura    | |
usuariomanipulardatos| |
bdcourier=#

```

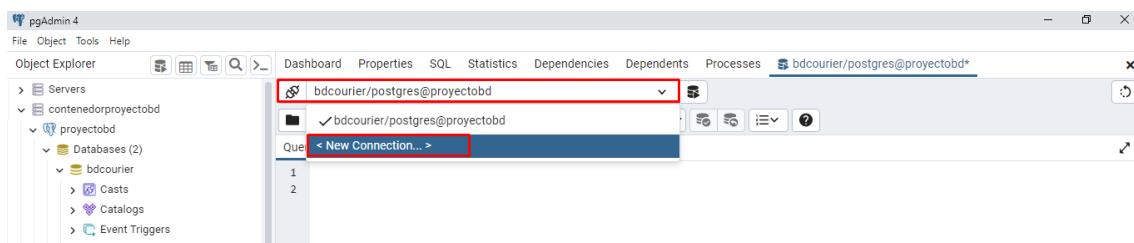
- Desde PGAdmin 4 también podemos observar los usuarios y roles creados ingresando a “Login/Group Roles”



- Nos vamos a conectar como usuario “usuariolectura”:



- Damos clic en la opción de la conexión actual y seleccionamos " New connection >"



- Ingresamos los parámetros para la nueva conexión:

Add New Connection

Server	proyectobd
Database	bdcourier
User	usuariolectura
Role	rolsololectura

Ingresamos el password para la conexión: "**userlectura1**"

Connect to server

Please enter the password for the user 'usuariolectura' to connect the server - "proyectobd"

Save Password

! connection failed: FATAL: password authentication failed for user "usuariolectura"

Observamos que se realizó la conexión correctamente:

The screenshot shows a PostgreSQL client interface. At the top, the connection parameters are displayed: "bdcourier/rolsololectura@proyectobd". Below this is a toolbar with various icons for database management. The main area is divided into two tabs: "Query" (selected) and "Query History". In the "Query" tab, there are two numbered lines of text input: "1" and "2".

- Realizamos una consulta a la base de datos "bdcourier" y se observa que si es posible:

The screenshot shows the MySQL Workbench interface with a query editor containing the following SQL code:

```
1 select * from cliente;
2
```

The results pane displays a table with 10 rows of data, each representing a client. The columns are labeled: idcliente [PK] integer, cicliente character varying, nombreciente character varying, apellidocliente character varying, direccioncliente character varying, and telefonocliente character varying. The data is as follows:

	idcliente [PK] integer	cicliente character varying	nombreciente character varying	apellidocliente character varying	direccioncliente character varying	telefonocliente character varying
1	643	8314519454	Modesto	Bilbao	Acceso de Heracio Cepeda 78 Piso 5	4845275854
2	644	3324432821	Encarnacion	Viñas	Ronda Ruben Blazquez 16	4807099525
3	645	1910477913	Florinda	Laguna	Pasadizo Pánfilo Carbajo 42 Apt. 60	4902153387
4	646	2181276270	Ariel	Arnaiz	Callejón Melania Gras 38	4721035037
5	647	4960045774	Pio	Mendizábal	Plaza de Glauco Gárate 715 Puerta 1	4887222399
6	648	3108956064	Macarena	Leon	Rambia Yago Lumbreras 66	4820230232
7	649	9453394204	Diana	Palomar	Avenida Clemente Agustí 91 Puerta 1	4836738812
8	650	7985145517	Moisés	González	Calle de Ainara Gálvez 8	4884845652
9	651	1803961941	Máxima	Baeza	Acceso de Jimena Sevillano 10	4881216601
10	652	123783296	Cayetana	Casas	Pasadizo de Leonardo Mena 94	4921525515

- Intentamos realizar un INSERT INTO en la tabla “cliente” y no se nos permite ya que el rol “rolsololectura” asignado al usuario “usuariolectura” solo permite la lectura de los datos de la base de datos:

insert into cliente
(cicliente,nombreciente,apellidocliente,direccioncliente,telefonocliente) values
('1723593596','Jonathan','Lema','La Mena','0996523254');

The screenshot shows the MySQL Workbench interface with a query editor containing the following SQL code:

```
1 insert into cliente (cicliente,nombreciente,apellidocliente,direccioncliente,telefonocliente) values ('1723593596',
```

The results pane displays an error message: "ERROR: permission denied for table cliente".

- Intentamos realizar un UPDATE en la tabla “cliente” y no se nos permite:

UPDATE cliente
SET nombreciente = 'Pedro'
WHERE cicliente = '8314519454';

The screenshot shows the MySQL Workbench interface. The connection is set to 'bdcourier/rolsololectura@proyectobd'. The toolbar includes standard icons for file operations, search, and execution. The 'No limit' dropdown is set to 'No limit'. Below the toolbar, there are tabs for 'Query' (selected) and 'Query History'. The main query editor contains the following code:

```
1 UPDATE cliente
2 SET nombrecliente = 'Pedro'
3 WHERE cicliente = '8314519454';
4
5
```

The first three lines of the query are highlighted with a red box. The 'Messages' tab is selected at the bottom, showing the error:

ERROR: permission denied for table cliente
SQL state: 42501

- Intentamos eliminar un registro en la tabla "cliente" y no se nos permite:
DELETE FROM cliente WHERE cicliente = '8314519454';

The screenshot shows the MySQL Workbench interface. The connection is set to 'bdcourier/rolsololectura@proyectobd'. The toolbar and 'No limit' dropdown are identical to the previous screenshot. The 'Query' tab is selected. The query editor contains the following code:

```
1 DELETE FROM cliente WHERE cicliente = '8314519454';
2
3
```

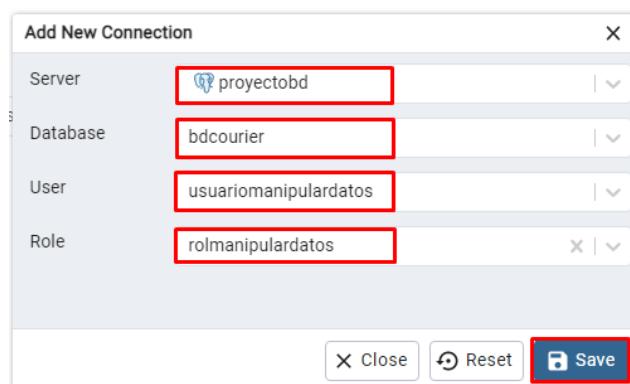
The first line of the query is highlighted with a red box. The 'Messages' tab is selected at the bottom, showing the error:

ERROR: permission denied for table cliente
SQL state: 42501

- Vamos a conectarnos a la base de datos "bdcourier" con el usuario "usuariomanipulardatos" dando clic en la opción de la conexión actual y seleccionamos "New connection >":

The screenshot shows the pgAdmin interface with the 'Connections' tab selected. At the bottom of the connection list, there is a blue button labeled '< New Connection... >'. This button is highlighted with a red box.

-Ingresamos los parámetros para la nueva conexión:



- Ingresamos el password para la conexión: "**usermanipulardatos2**"
- Observamos que se realizó la conexión correctamente:

The screenshot shows the pgAdmin interface with the connection dropdown set to 'bdcourier/rolmanipulardatos@proyectobd'. Below the dropdown, the 'Query' tab is selected. The connection name is also highlighted with a red box.

- Realizamos una consulta a la base de datos "bdcourier" y se observa que si es posible:

The screenshot shows a MySQL Workbench interface. The connection is set to 'bdcourier/rolmanipulardatos@proyectobd'. The query window contains the following code:

```

1 select * from repartidor;
2
3

```

The results window displays the data from the 'repartidor' table:

	idrepartidor [PK] integer	nombrerepartidor character varying	apellidorepartidor character varying	direccionrepartidor character varying	cargorepartidor character varying
1	11	Cayetana	Caro	Alameda Guadalupe Ponce 925 Piso 0	Repartidor 27y
2	12	Victoriano	Ferrández	Avenida Amor Montenegro 525	Repartidor 28C
3	13	Perlita	Guzmán	Cuesta de Valentina Simó 80	Repartidor 360
4	14	Fito	Bermúdez	Ronda de Maxi Villegas 8	Repartidor 51I
5	15	Bernardino	Escobar	Via Jesusa Taboada 18 Piso 1	Repartidor 73j
6	16	Ana Sofía	Morales	Pasaje de Marisela Valcárcel 41	Repartidor 99W
7	17	Nacho	Patiño	Camino de Rosa Mateos 39	Repartidor 97c
8	18	Ale	Palomares	Callejón de Ruth Pereira 70	Repartidor 64U
9	19	Luciana	Batlle	Plaza Magdalena Vall 3	Repartidor 62d
10	20	Xiomara	Tello	Urbanización de Natalio Pujol 175 Apt. 06	Repartidor 53Q

- Intentamos realizar un INSERT INTO en la tabla repartidor y si se nos permite:

INSERT INTO

```

repartidor(nombrerepartidor,apellidorepartidor,direccionrepartidor,cargorepartidor)
VALUES ('Jorge','Mena','La tola','Repartido 10a');

```

The screenshot shows a MySQL Workbench interface. The connection is set to 'bdcourier/rolmanipulardatos@proyectobd'. The query window contains the following code:

```

1 INSERT INTO repartidor(nombrerepartidor,apellidorepartidor,direccionrepartidor,cargorepartidor)
2 VALUES ('Jorge','Mena','La tola','Repartido 10a');
3
4

```

The results window shows the message:

```

INSERT 0 1

```

Below the results, it says: "Query returned successfully in 251 msec."

- Observamos que si se ingresó el registro correctamente en la tabla repartidor:
SELECT * FROM repartidor;

The screenshot shows the pgAdmin III interface with a connection to 'bdcourier/rolmanipulardatos@proyectobd'. In the 'Query' tab, the following SQL code is entered:

```

1 SELECT * FROM repartidor;
2
3

```

The results are displayed in a table titled 'Data Output' with the following data:

	idrepartidor [PK] integer	nombrerepartidor character varying	apellidorepartidor character varying	direccionrepartidor character varying	cargorepartidor character varying
1	11	Cayetana	Caro	Alameda Guadalupe Ponce 925 Piso 0	Repartidor 27y
2	12	Victoriano	Ferrández	Avenida Amor Montenegro 525	Repartidor 28C
3	13	Perlita	Guzmán	Cuesta de Valentina Simó 80	Repartidor 36o
4	14	Fito	Bermúdez	Ronda de Maxi Villegas 8	Repartidor 51I
5	15	Bernardino	Escobar	Via Jesusa Taboada 18 Piso 1	Repartidor 73j
6	16	Ana Sofía	Morales	Pasaje de Marisela Valcárcel 41	Repartidor 99W
7	17	Nacho	Patiño	Camino de Rosa Mateos 39	Repartidor 97c
8	18	Ale	Palomares	Callejón de Ruth Pereira 70	Repartidor 64U
9	19	Luciana	Batlle	Plaza Magdalena Vall 3	Repartidor 62d
10	20	Xiomara	Tello	Urbanización de Natalio Pujol 175 Apt. 06	Repartidor 53Q
11	21	Jorge	Mena	La tola	Repartido 10a

A message bar at the bottom right says: 'Successfully run. Total query runtime: 309 msec. 11 rows affected.'

- Intentamos realizar un UPDATE en la tabla "repartidor" y si se nos permite:

UPDATE repartidor

SET nombrerepartidor = 'Susana'

WHERE apellidorepartidor = Mena;

The screenshot shows the pgAdmin III interface with a connection to 'bdcourier/rolmanipulardatos@proyectobd'. In the 'Query' tab, the following SQL code is entered:

```

1 UPDATE repartidor
2 SET nombrerepartidor = 'Susana'
3 WHERE apellidorepartidor = 'Mena';
4
5

```

The results are displayed in the 'Messages' tab with the following output:

```

UPDATE 1
Query returned successfully in 333 msec.

```

- Observamos que se actualizo el registro correctamente:

The screenshot shows the MySQL Workbench interface. The query editor contains the following SQL code:

```

1 select * from repartidor;
2
3
4

```

The results pane displays a table with 11 rows of data. The last row, which corresponds to the record with idrepartidor 21 and nombrerepartidor 'Susana', is highlighted with a red box.

	idrepartidor [PK] integer	nombrerepartidor character varying	apellidorepartidor character varying	direccionrepartidor character varying	cargorepartidor character varying
1	11	Cayetana	Caro	Alameda Guadalupe Ponce 925 Piso 0	Repartidor 27y
2	12	Victoriano	Ferrández	Avenida Amor Montenegro 525	Repartidor 28C
3	13	Perlita	Guzmán	Cuesta de Valentina Simó 80	Repartidor 360
4	14	Fito	Bermúdez	Ronda de Maxi Villegas 8	Repartidor 511
5	15	Bernardino	Escobar	Via Jesusa Taboada 18 Piso 1	Repartidor 73j
6	16	Ana Sofía	Morales	Pasaje de Marisela Valcárcel 41	Repartidor 99W
7	17	Nacho	Patiño	Camino de Rosa Mateos 39	Repartidor 97c
8	18	Ale	Palomares	Callejón de Ruth Pereira 70	Repartidor 64U
9	19	Luciana	Batlle	Plaza Magdalena Vall 3	Repartidor 62d
10	20	Xiomara	Tello	Urbanización de Natalio Pujol 175 Apt. 06	Repartidor 53Q
11	21	Susana	Mena	La tola	Repartido 10a

Below the table, a message box indicates: "Successfully run. Total query runtime: 155 msec. 11 rows affected." with a green checkmark icon.

- Intentamos eliminar el registro actualizado recientemente:

DELETE FROM repartidor WHERE nombrerepartidor = 'Susana' ;

The screenshot shows the MySQL Workbench interface. The query editor contains the following SQL code:

```

1 DELETE FROM repartidor WHERE nombrerepartidor = 'Susana';
2
3
4

```

The results pane displays the output of the DELETE query:

```

Data Output Messages Notifications
DELETE 1
Query returned successfully in 237 msec.

```

The message "Query returned successfully in 237 msec." is highlighted with a red box.

- Observamos que se eliminó correctamente el registro:

select * from repartidor;

The screenshot shows a PostgreSQL query interface. The query entered is:

```
1 select * from repartidor;
```

The results table has the following columns:

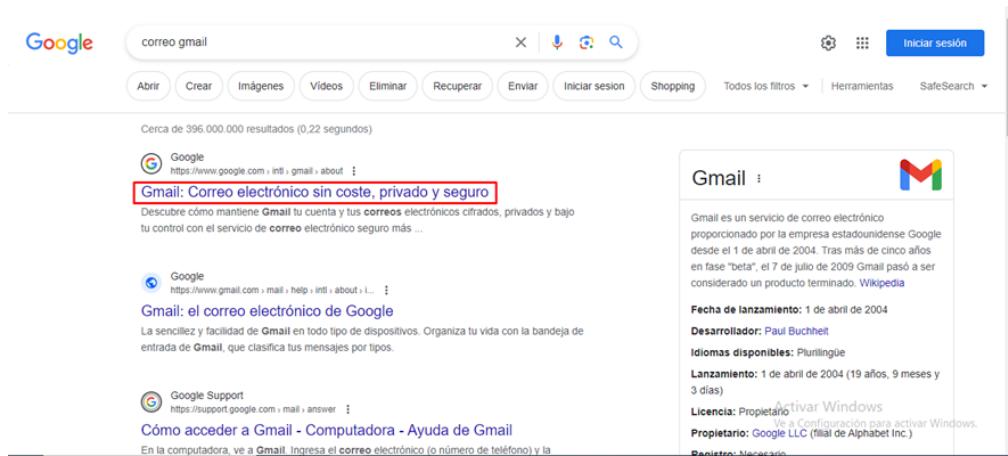
	idrepartidor [PK] integer	nombrerepartidor character varying	apellidorepartidor character varying	direccionrepartidor character varying	cargorepartidor character varying
1	11	Cayetana	Caro	Alameda Guadalupe Ponce 925 Piso 0	Repartidor 27y
2	12	Victoriano	Ferrández	Avenida Amor Montenegro 525	Repartidor 28C
3	13	Perilita	Guzmán	Cuesta de Valentina Simó 80	Repartidor 360
4	14	Fito	Bermúdez	Ronda de Maxi Villegas 8	Repartidor 511
5	15	Bernardino	Escobar	Via Jesusa Taboada 18 Piso 1	Repartidor 73j
6	16	Ana Sofía	Morales	Pasaje de Marisela Valcárcel 41	Repartidor 99W
7	17	Nacho	Patíño	Camino de Rosa Mateos 39	Repartidor 97c
8	18	Ale	Palomares	Callejón de Ruth Pereira 70	Repartidor 64U
9	19	Luciana	Batlle	Plaza Magdalena Vall 3	Repartidor 62d
10	20	Xiomara	Tello	Urbanización de Natalio Pujol 175 Apt. 06	Repartidor 53Q

A green success message at the bottom right says: "Successfully run. Total query runtime: 182 msec. 10 rows affected."

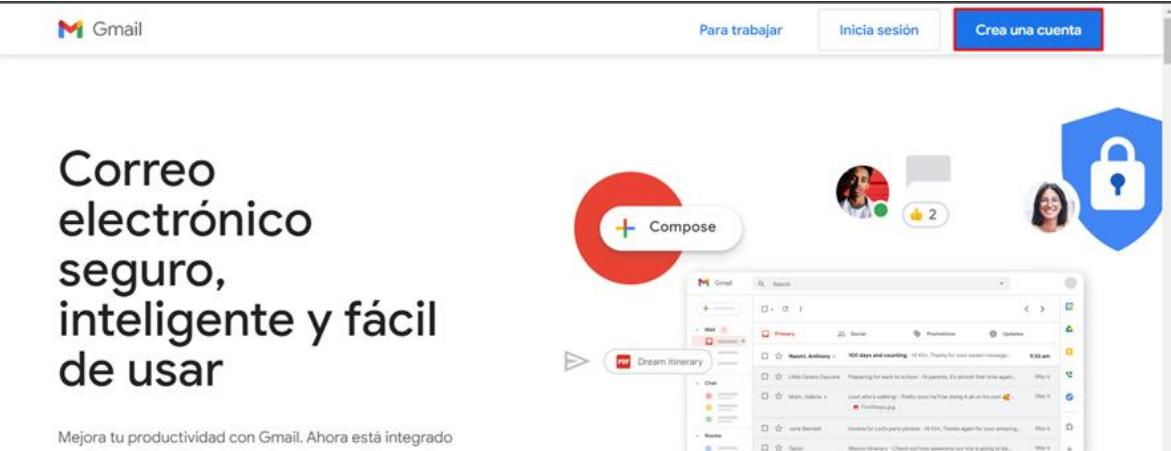
11. Creación de Github

- Crear correo de Gmail para crear cuenta en Github

- Accedemos a Gmail



- Dar clic en el botón “Crear una cuenta”



Gmail

Para trabajar

Inicia sesión

Crea una cuenta

Compose

Dream itinerary

Anthony : 100 days and counting

Uma Goshwara : Preparing for books to arrive! 10 pages. It's almost that time again...

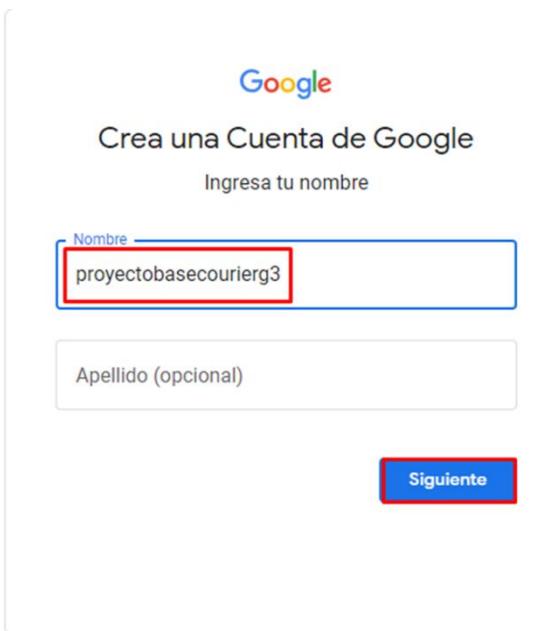
Alex_Alexia : Love what's cooking? Pretty soon he'll be doing it all on his own!

Jane Beaman : Invites for Lucy's party phew... 10 items. Thanks again for your amazing...

Taylor : Mexico itinerary - Check out these awesome new trips I'm going to take...

Mejora tu productividad con Gmail. Ahora está integrado

- Ingresar nombre del proyecto: “**proyectobasecourierg3**”



Google

Crea una Cuenta de Google

Ingresa tu nombre

Nombre

Apellido (opcional)

Siguiente

- Ingresar información para la cuenta de Gmail y dar clic en “Siguiente”:

FACULTAD DE INGENIERÍA Y CIENCIAS APLICADAS

Google

Información básica

Completa tu fecha de nacimiento y género.

Día 01 Mes Enero Año 2000

Género Prefiero no decirlo

Por qué solicitamos la fecha de nacimiento y el género

Siguiente

- Ingresamos el nombre del correo de Gmail para el proyecto “**proyectobasecourierg3**” y dar clic en el botón “**Siguiente**”:

Google

Elige tu dirección de Gmail

Elige una dirección de Google o crea una nueva.

proyectobasecourierg51@gmail.com

proyectobasecourierg@gmail.com

Crear tu propia dirección de Gmail

Crea una dirección de Gmail
proyectobasecourierg3 @gmail.com

Puedes usar letras, números y signos de puntuación

Siguiente

- Ingresar el password para el correo del proyecto: Password: **bdcourier**

The screenshot shows a 'Google' logo at the top. Below it, the heading 'Crea una contraseña segura' is displayed. A sub-instruction reads: 'Crea una contraseña segura con letras, números y símbolos combinados.' Two input fields are present: 'Contraseña' containing 'bdcourier' and 'Confirmación' also containing 'bdcourier'. A checked checkbox labeled 'Mostrar contraseña' (Show password) is visible. At the bottom right is a blue 'Siguiente' (Next) button.

- Ingresar un número telefónico para validar la cuenta de correo de Gmail creada para el proyecto:

The screenshot shows a 'Google' logo at the top. Below it, the heading 'Demuestra que no eres un robot' is displayed. A sub-instruction reads: 'Recibir un código de verificación en tu teléfono'. An input field for 'Número de teléfono' shows '0996863282', preceded by a dropdown menu set to the flag of Venezuela. At the bottom right is a blue 'Siguiente' (Next) button.

- Se aceptan los términos y condiciones de Google para la creación del correo electrónico en gmail:

Google
Privacidad y Condiciones

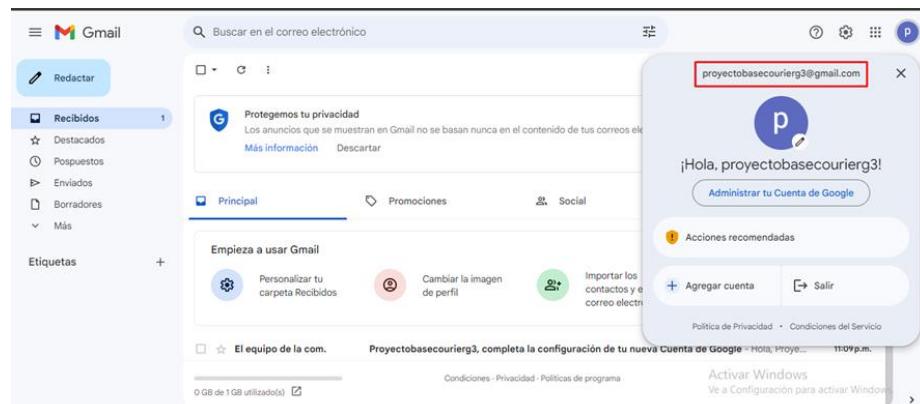
Para crear una cuenta de Google, deberás aceptar las [Condiciones del Servicio](#) que se encuentran a continuación.

Además, cuando creas una cuenta, procesamos tu información como se describe en nuestra [Política de Privacidad](#), incluidos estos puntos clave:

Datos que procesamos cuando usa Google

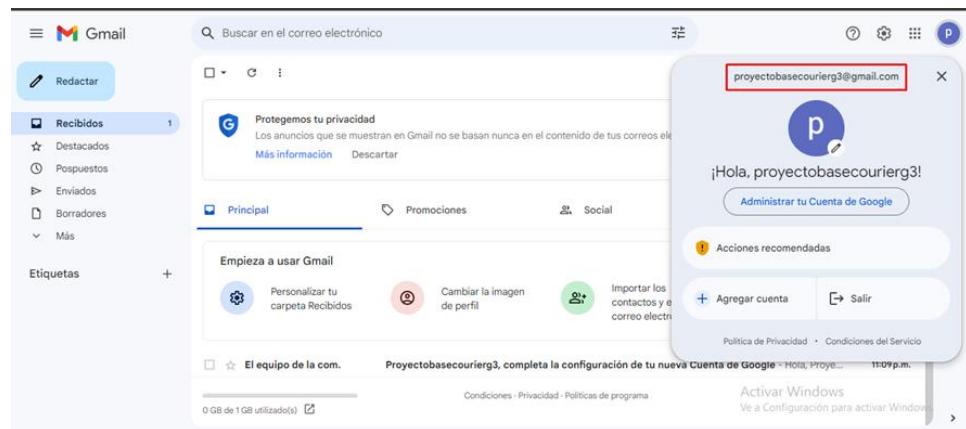
- Cuando configura una cuenta de Google, almacenamos la información que proporciona, como su nombre, dirección de correo electrónico y número de teléfono.
- Por ejemplo, cuando usa los servicios de Google para escribir un mensaje en Gmail o un comentario en un video de YouTube, almacenamos la información que genera.
- Por ejemplo, cuando busca un restaurante en Google Maps o mira un video en YouTube, procesamos información sobre esa actividad, lo que incluye los siguientes datos: el video que miró, los ID del

- Observamos que se ha creado correctamente el correo electrónico para el proyecto:

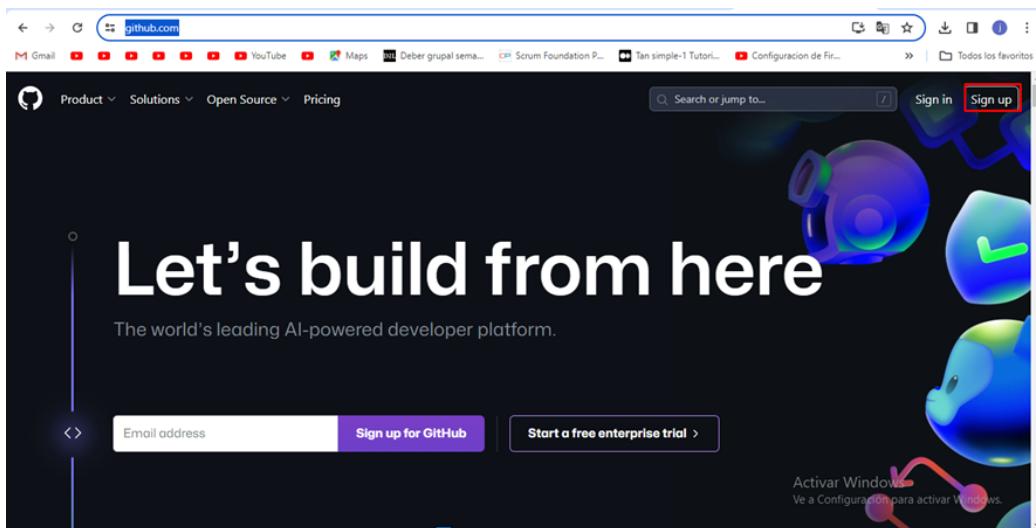


Vamos a crear la cuenta de Github para el proyecto.

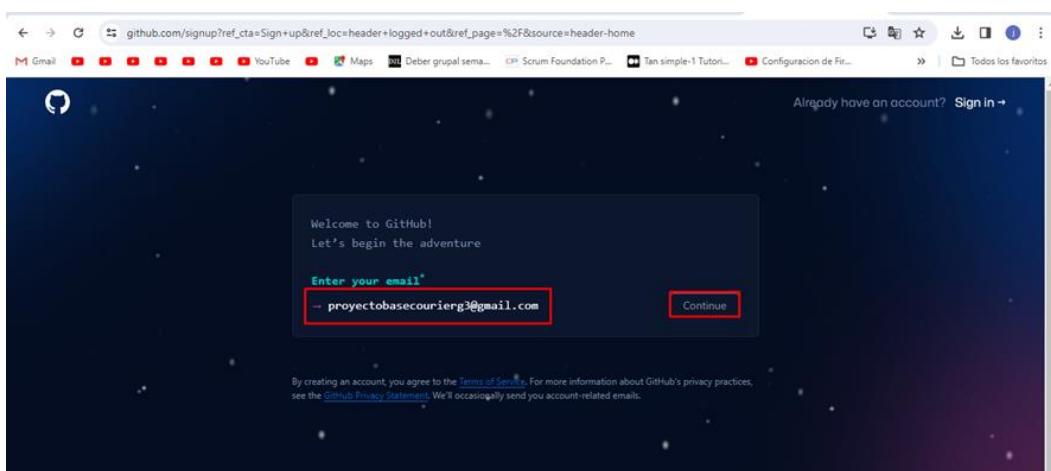
- Ingresamos a la página oficial de Github con el link: <https://github.com/>



- En la página web oficial de Github damos clic en el botón “Sign up” para crear la cuenta de Github para el proyecto:

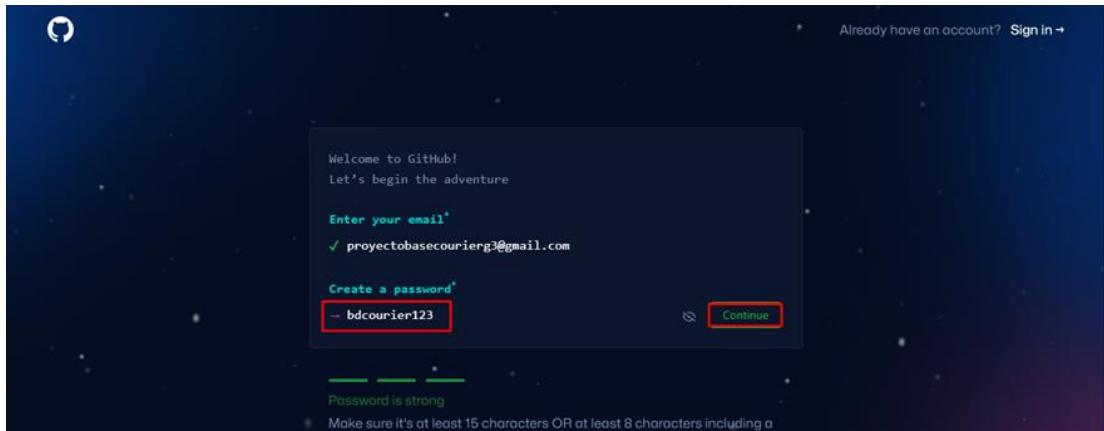


- Ingresar el correo de Gmail creado para el proyecto en el input de “Enter your *email” en la página web de Github y dar clic en el botón “Continue”:



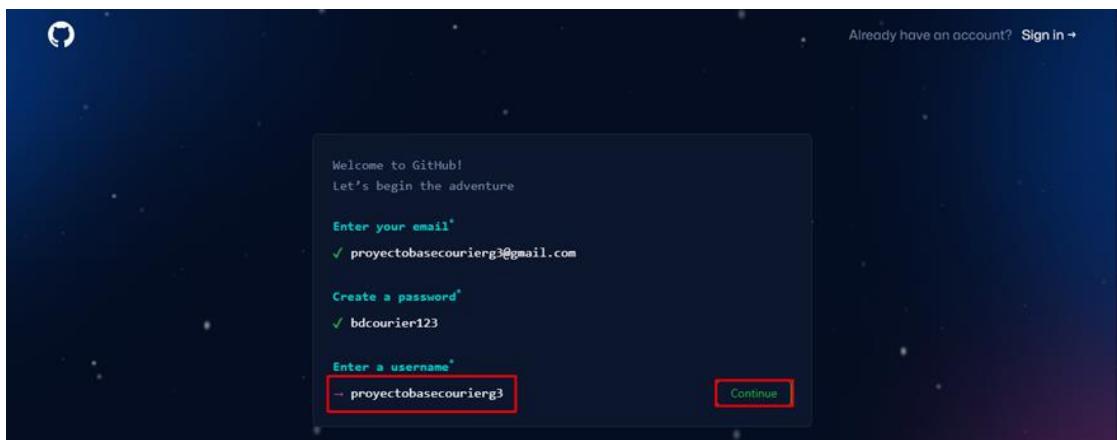
- Crear el password para la cuenta de Github del proyecto:

Password: bdcourier123

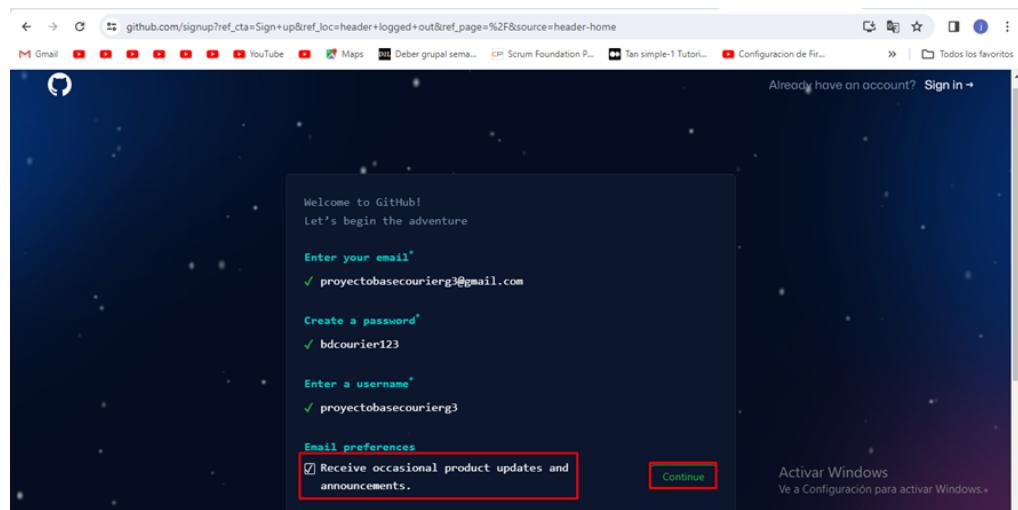


Ingresamos un username:

Username: proyectobasecourierg3

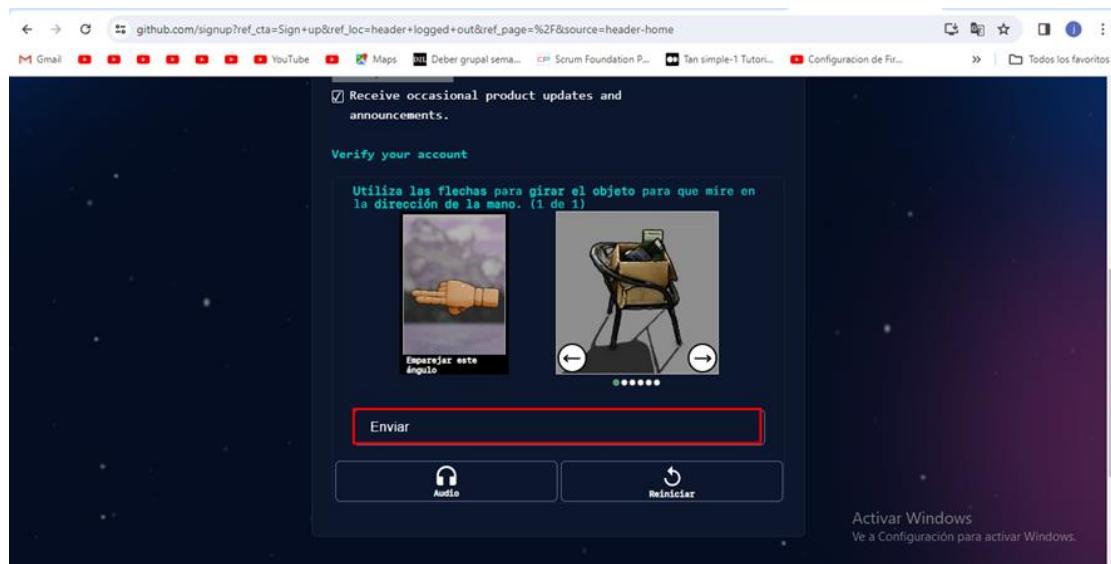


- Dar clic en check “Email preferences” y dar clic en botón “Continue”:

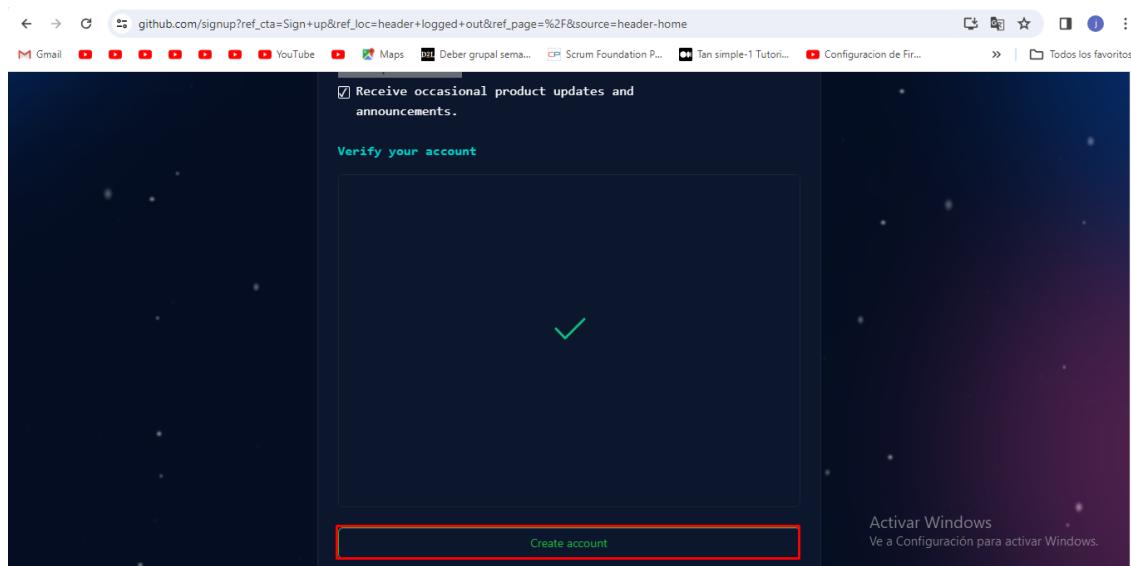


- Verificar la cuenta de Github:

FACULTAD DE INGENIERÍA Y CIENCIAS APLICADAS

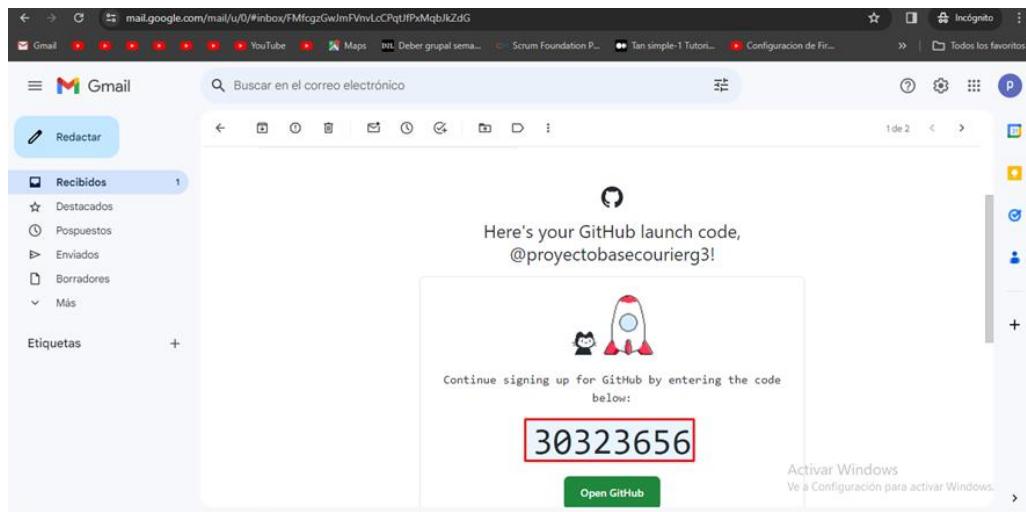


- Dar clic en el botón “**Create account**”:

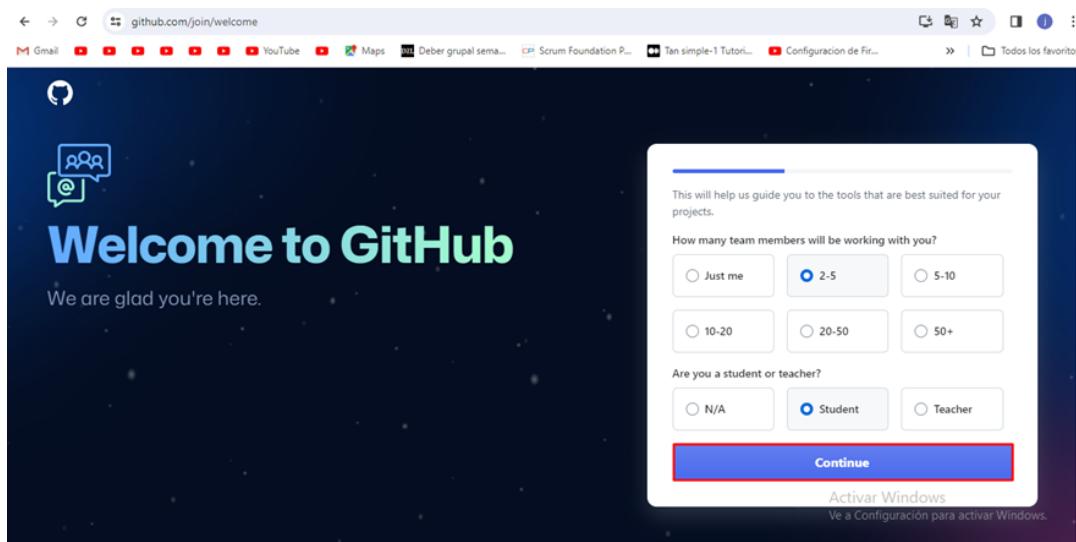


- Ingresar el código recibido en el correo: “proyectobasecourierg3@gmail.com”

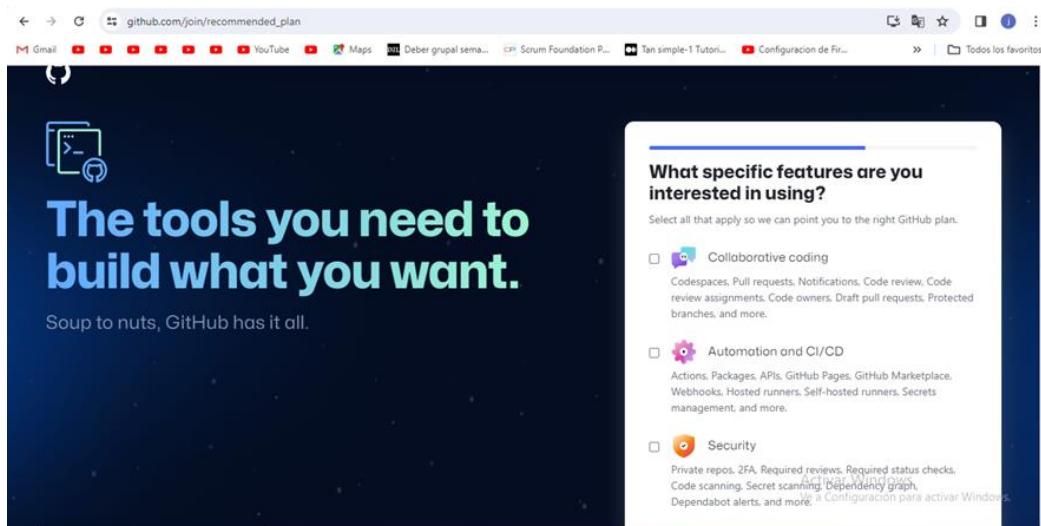
FACULTAD DE INGENIERÍA Y CIENCIAS APLICADAS



- Damos clic en el botón “Continue”:

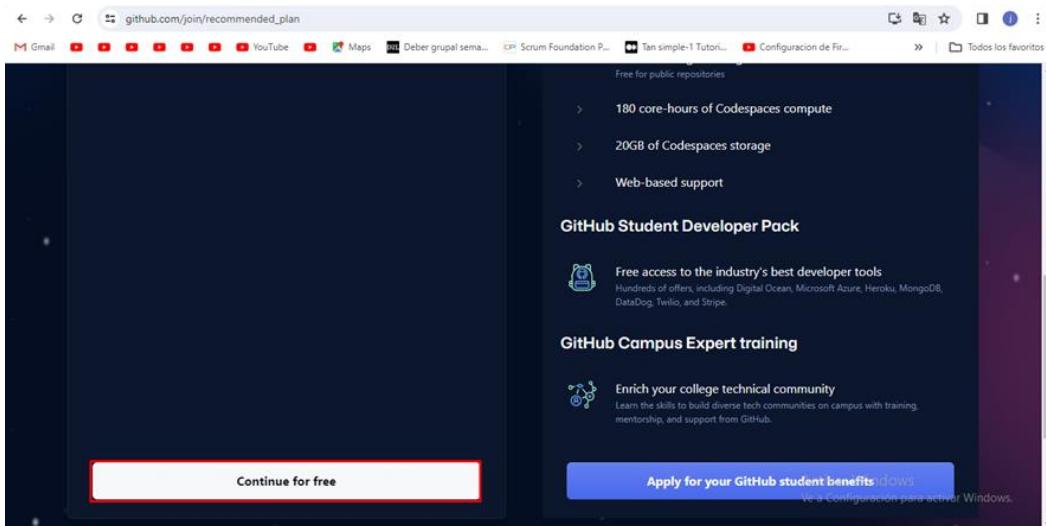


- Ingresamos la configuración básica de Github:

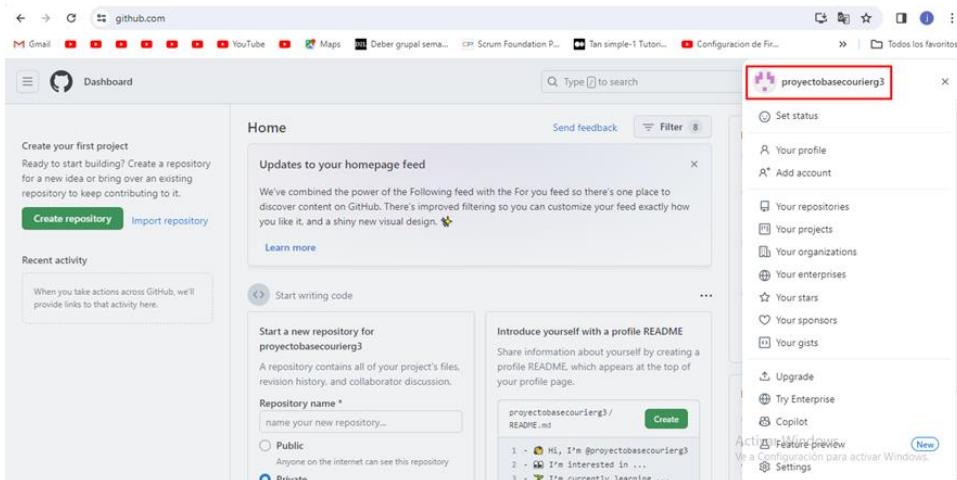


- Seleccionamos una cuenta free de Github para el proyecto:

FACULTAD DE INGENIERÍA Y CIENCIAS APLICADAS



- Observamos que se creó la cuenta de Github para el proyecto exitosamente:



Crear un proyecto nuevo en Github

- Damos clic en el botón “Create repository” de la página web de github del proyecto

FACULTAD DE INGENIERÍA Y CIENCIAS APLICADAS

The screenshot shows the GitHub homepage. On the left, there's a sidebar with a 'Create your first project' section containing a 'Create repository' button (which is highlighted with a red box) and an 'Import repository' link. Below this is a 'Recent activity' section with a note about providing links to actions taken across GitHub. On the right, there's a 'Home' section with a 'Updates to your homepage feed' summary and a 'Start writing code' button.

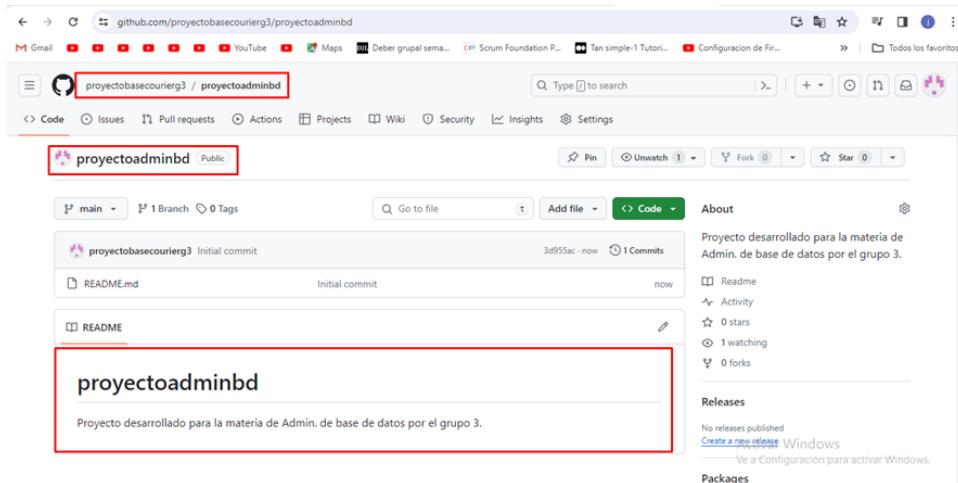
- Colocamos el nombre del repositorio: “**proyectoadminbd**”

The screenshot shows the 'Create a new repository' form on GitHub. The 'Repository name' field contains 'proyectoadminbd'. Other fields include 'Owner' set to 'proyectobasecourierg3', 'Description (optional)' with the text 'Proyecto desarrollado para la materia de Admin. de base de datos por el grupo 3.', and 'Visibility' set to 'Public'. The 'Create repository' button is at the bottom.

- Ingresamos la siguiente configuración para la creación del repositorio nuevo:

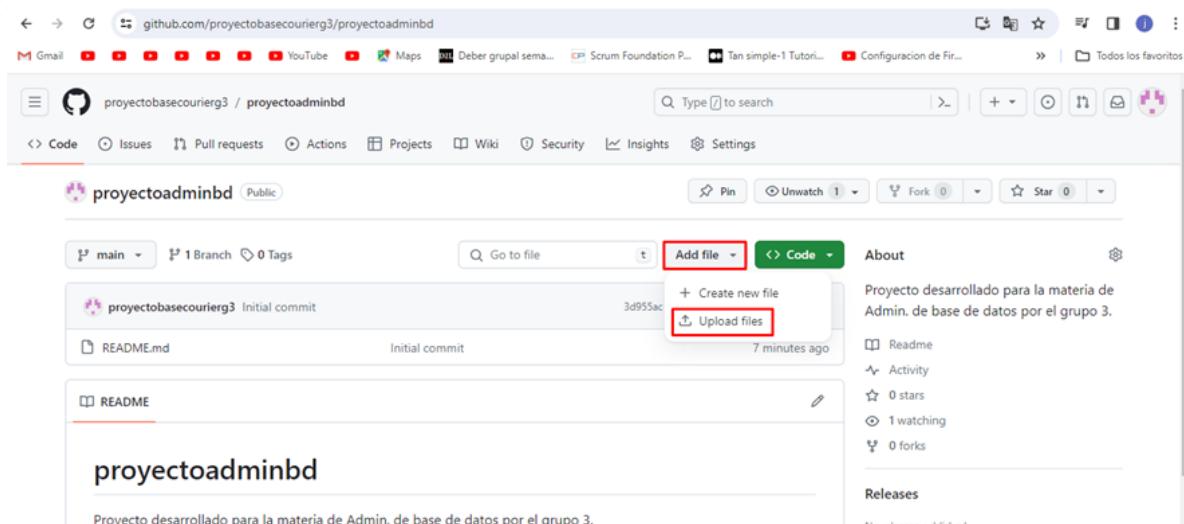
The screenshot shows the 'Create a new repository' form with more detailed configuration. Under 'Initialize this repository with:', 'Add a README file' is checked. Under 'Choose a license', 'License: None' is selected. The 'Create repository' button is at the bottom.

- Observamos que se a creado el repositorio “**proyectoadminbd**” correctamente:



Subir los archivos del proyecto a repositorio de Github

Damos clic en el botón “Add files” -> “Upload files”



Subimos los archivo del proyecto al repositorio:

Modelo lógico de base de datos:

FACULTAD DE INGENIERÍA Y CIENCIAS APLICADAS

The screenshot shows a GitHub repository named 'proyectoadminbd'. In the file list, there is a single file named 'ModeloCourierDB.cdm' which is highlighted with a red box. Below the file list, there is a 'Commit changes' button.

Modelo físico de base de datos:

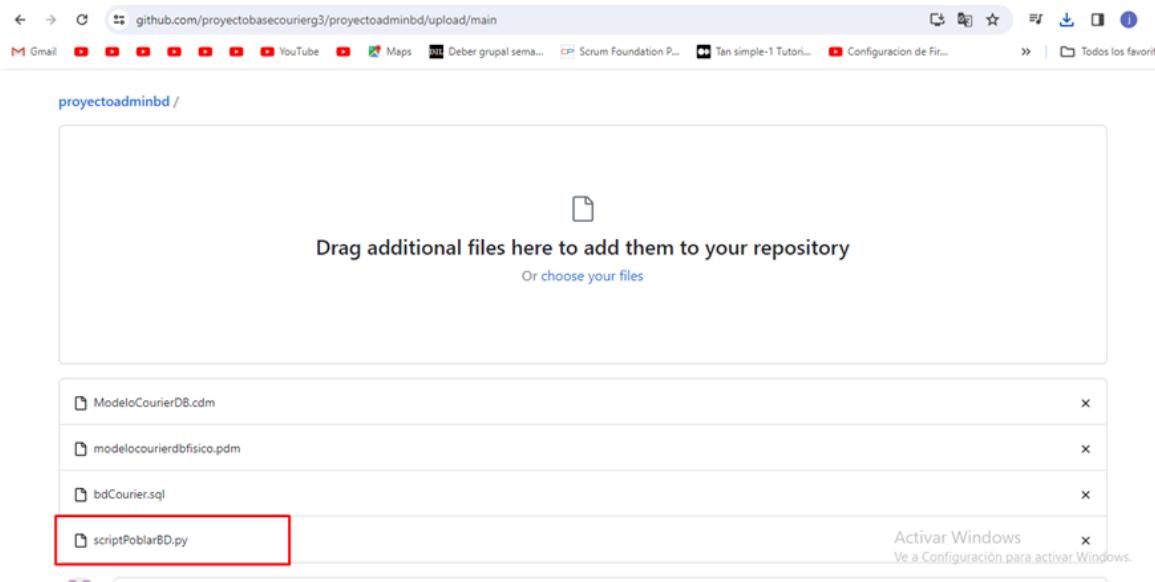
The screenshot shows a GitHub repository named 'proyectoadminbd'. In the file list, there is a single file named 'modelocourierdbfisico.pdm' which is highlighted with a red box. Below the file list, there is a 'Commit changes' button.

Script de base de datos:

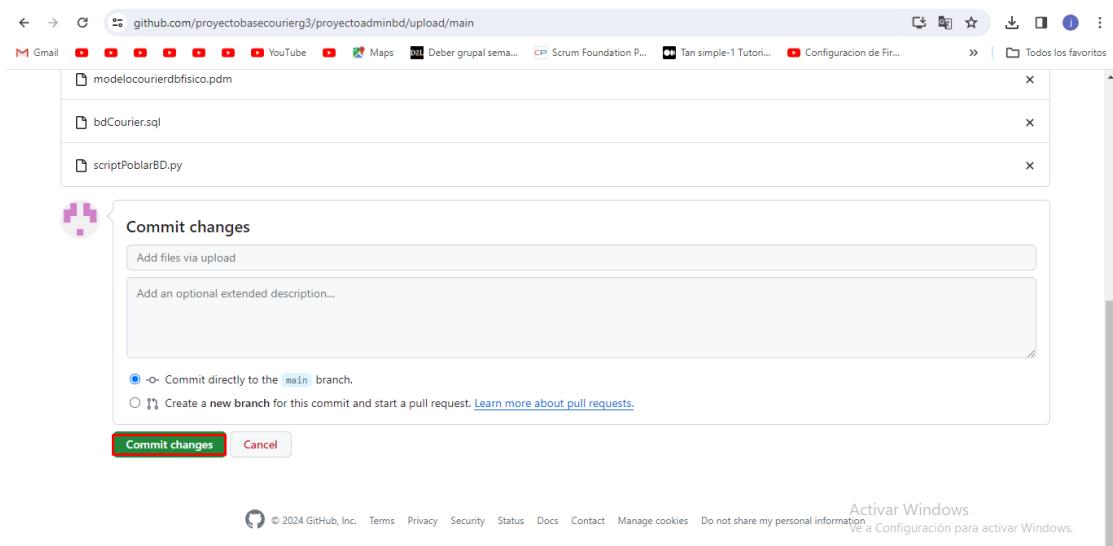
The screenshot shows a GitHub repository named 'proyectoadminbd'. In the file list, there is a single file named 'bdCourier.sql' which is highlighted with a red box. Below the file list, there is a 'Commit changes' button.

Documento .pdf de proyecto:

Script para poblar base de datos con Faker:



Dar clic en el botón "Commit changes" para guardar los cambios



Se observan los archivos del proyecto subidos al repositorio en Github

3. Diccionario de Datos

El diccionario de datos que has creado describe las tablas y campos de una base de datos para un servicio de courier. La base de datos consta de las siguientes tablas:

Tabla: CLIENTE

Esta tabla almacena la información de los clientes de la empresa de Courier.

Campo	Descripción	Tipo de dato	Explicación	Nulo
IDCLIENTE	Identificador único de cada cliente	SERIAL	Autonumerado	No
CICLIENTE	Cédula de identidad del cliente	VARCHAR(200)	Opcional	Sí
NOMBRECLIENTE	Nombre del cliente	VARCHAR(200)	Obligatorio	No
APELLIDOCLIENTE	Apellido del cliente	VARCHAR(200)	Obligatorio	No
DIRECCIONCLIENTE	Dirección del cliente	VARCHAR(200)	Obligatorio	No
TELEFONOCLIENTE	Número de teléfono del cliente	VARCHAR(10)	Opcional	Sí

Tabla: DESTINATARIO

Esta tabla almacena la información de los destinatarios de las encomiendas.

Campo	Descripción	Tipo de dato	Explicación	Nulo
IDDESTINATARIO	Identificador único de cada destinatario	SERIAL	Autonumerado	No
CIDESTINATARIO	Cédula de identidad del destinatario	VARCHAR(200)	Opcional	Sí
NOMBREDESTINATARIO	Nombre del destinatario	VARCHAR(200)	Obligatorio	No
APELLIDODESTINATARIO	Apellido del destinatario	VARCHAR(200)	Obligatorio	No

FACULTAD DE INGENIERÍA Y CIENCIAS APLICADAS

DIRECCIONDESTINATARIO	Dirección del destinatario	VARCHAR(200)	Obligatorio	No
TELEFONODESTINATARIO	Número de teléfono del destinatario	VARCHAR(10)	Opcional	Sí

Tabla: ENCOMIENDA

Esta tabla almacena la información de las encomiendas que maneja la empresa.

Campo	Descripción	Tipo de dato	Explicación	Nulo
IDENCOMIENDA	Identificador único de cada encomienda	SERIAL	Autonumerado	No
IDESTADO	Identificador del estado actual de la encomienda	INT4	Relacionado con la tabla ESTADOENCOMIENDA	No
TIPOENCOMIENDA	Tipo de encomienda	VARCHAR(200)	Obligatorio	No
ALTURA	Altura de la encomienda en centímetros	FLOAT8	Obligatorio	No
PESO	Peso de la encomienda en kilogramos	FLOAT8	Obligatorio	No
DESCRIPCIONENCOMIENDA	Descripción adicional de la encomienda	VARCHAR(200)	Opcional	Sí
CATEGORIA	Categoría de la encomienda	VARCHAR(200)	Obligatorio	No

Tabla: ENVIO

Esta tabla almacena la información del envío de las encomiendas.

Campo	Descripción	Tipo de dato	Explicación	Nulo
IDENVIO	Identificador único de cada envío	SERIAL	Autonumerado	No
IDTICKET	Identificador del ticket asociado al envío	INT4	Relacionado con la tabla TICKET	No
IDDESTINATARIO	Identificador del destinatario	INT4	Relacionado con la tabla DESTINATARIO	No
IDRUTA	Identificador de la ruta asignada al envío	INT4	Relacionado con la tabla RUTA	Sí
IDCLIENTE	Identificador del cliente que solicita el envío	INT4	Relacionado con la tabla CLIENTE	No

FACULTAD DE INGENIERÍA Y CIENCIAS APLICADAS

IDTIPOENVIO	Identificador del tipo de envío	INT4	Relacionado con la tabla TIPOENVIO	No
IDENCOMIENDA	Identificador de la encomienda enviada	INT4	Relacionado con la tabla ENCOMIENDA	No
IDREPARTIDOR	Identificador del repartidor asignado al envío	INT4	Relacionado con la tabla REPARTIDOR	No
DURACIONENVIO	Duración estimada del envío	TIME	Opcional	Sí

Tabla: ESTADO ENCOMIENDA

Esta tabla almacena en estado en el que se encuentra el despacho de la encomienda.

Campo	Descripción	Tipo de dato	Explicación	Nulo
IDESTADO	Identificador único de cada estado	SERIAL	Autonumerado	No
ACTIVOENCOMIENDA	Descripción del estado actual de la encomienda	VARCHAR(200)	Obligatorio	No
FECHAESTADO	Fecha y hora en que se registró el estado actual	DATE	Obligatorio	No

Tabla: REPARTIDOR

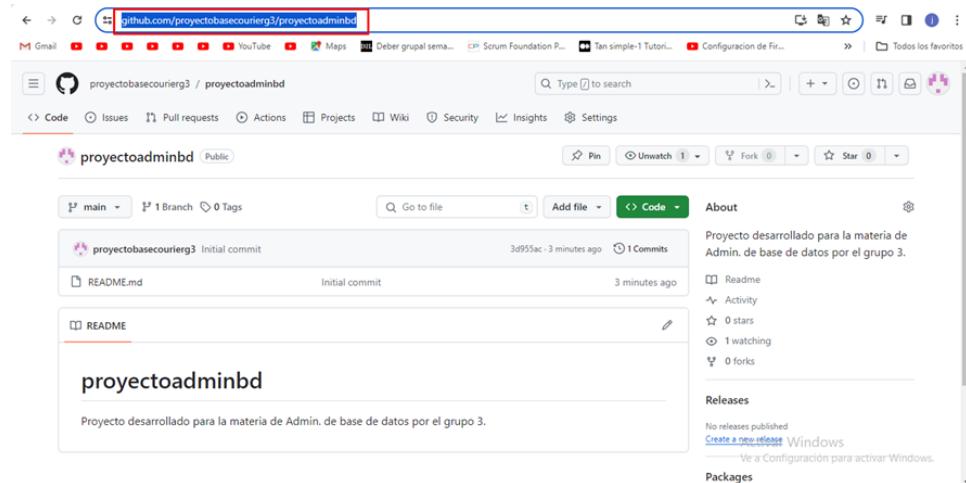
Esta tabla almacena la información de los repartidores de la empresa.

Campo	Descripción	Tipo de dato	Explicación	Nulo
IDREPARTIDOR	Identificador único de cada repartidor	SERIAL	Autonumerado	No
NOMBREREPARTIDOR	Nombre del repartido			

4. Instrucciones para acceder al repositorio en GitHub

Para acceder al repositorio de Github ingresamos la url del proyecto en el navegador:

Repositorio: <https://github.com/proyectobasecourierg3/proyectoadminbd>



5. Informe de cumplimiento de los siguientes criterios de desempeño:

1.- Criterio

Definición de los Objetivos de Trabajo

Porcentaje de Cumplimiento

100%

Evidencia(s)

- Realizar reuniones para abordar los temas de implementación la ejecución del proyecto
- Cumplir con los horarios establecidos previamente para realizar el proyecto
- Implementar una solución con una base de datos contenerizada

2.- Criterio

Definición de Cronograma

Porcentaje de Cumplimiento

100%

Evidencia(s)

21/12/2023	Modelado de la base de datos
12/1/2024	Implementación de la base de datos contenerizada
12/1/2024	Cambio de puerto
12/1/2024	Conexión entre pgAdmin con la base contenerizada de Postgres
13/1/2024	Creación de la base desde pg admin con el script que ya anteriormente se modelo
13/1/2024	Desarrollo de script para poblar la base de datos utilizando la librería Faker
13/1/2024	Creación de roles y usuarios
15/1/2024	Desarrollo del documento formal
15/1/2024	Crear cuenta de gib hub
15/1/2024	Subir de archivos al repositorio

3.- Criterio

Definición de Roles

Porcentaje de Cumplimiento

100%

Evidencia(s)

Modelador de Base de Datos:

- Es el encargado de diseñar y crear modelos de base de datos que representen la estructura y relación de los datos en un sistema.

Administrador de Base de Datos:

- Es el encargado de la gestión y administración de bases de datos, asegurando su rendimiento, seguridad y disponibilidad.

Desarrollador:

- Es el encargado de escribir el código para construir aplicaciones o sistemas según los requisitos del proyecto.

4.- Criterio

Asignación de roles

Porcentaje de Cumplimiento

100%

Evidencia(s)

William Morales - Modelador de Base de Datos

Belén Gavilanes – Administrador de Base de datos

Jonathan Lema - Desarrollador

5.- Criterio

Asignación de Responsabilidades

Porcentaje de Cumplimiento

100%

Evidencia(s)

William Morales - Modelador de Base de Datos:

- Diseñar y crear el modelo de base de datos que reflejen los requisitos del sistema.

Belén Gavilanes – Administrador de Base de Datos:

- Gestionar y administrar del sistema de datos.

Jonathan Lema - Desarrollador:

- Escribir código de alta calidad que cumpla con los requisitos del proyecto.

6.- Criterio

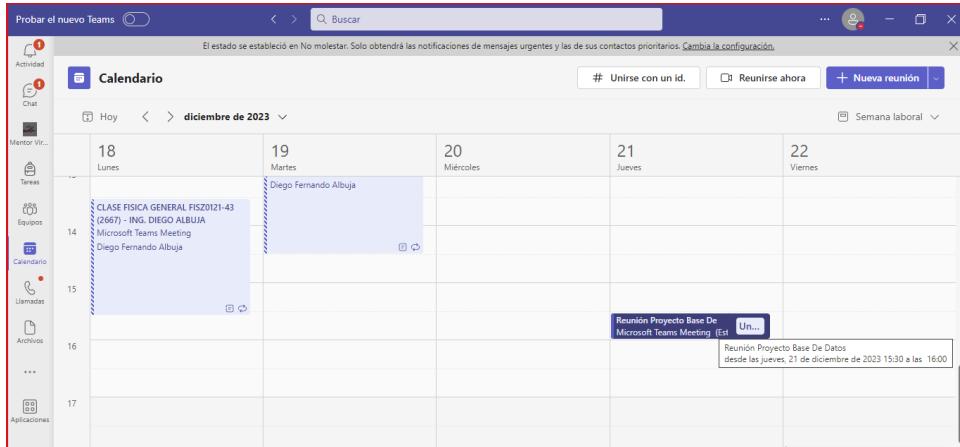
Cronograma de reuniones de trabajo

Porcentaje de Cumplimiento

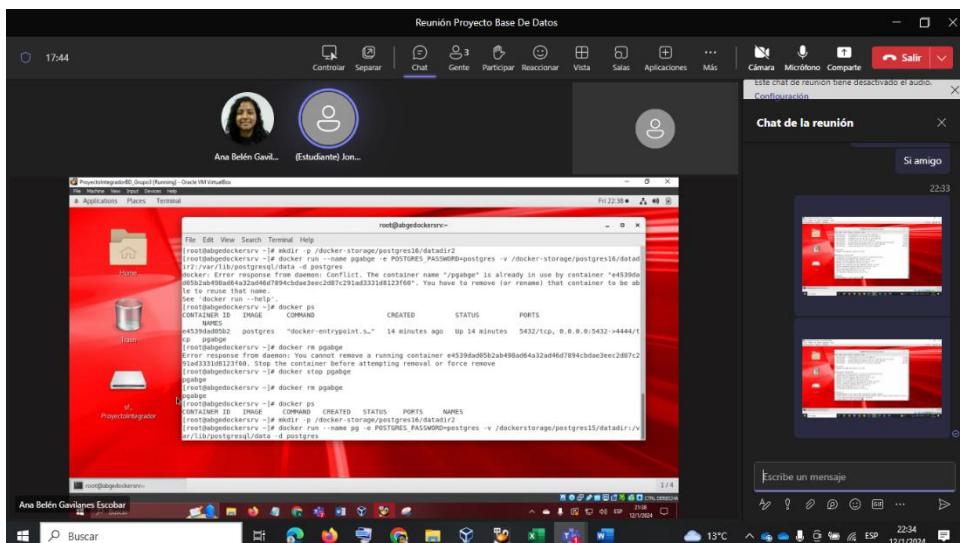
100%

Evidencia(s)

Primera reunión el 21 de diciembre

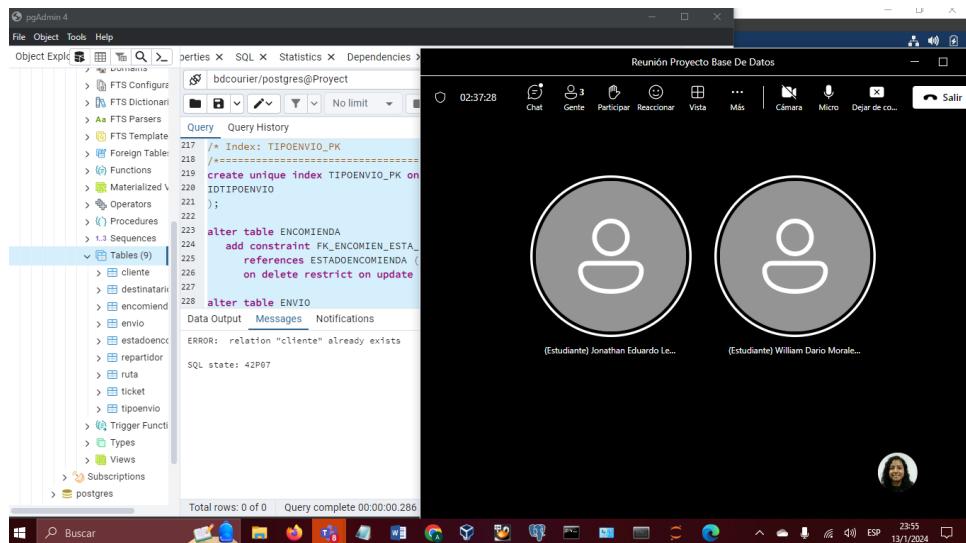


Segunda reunión el 12 de enero

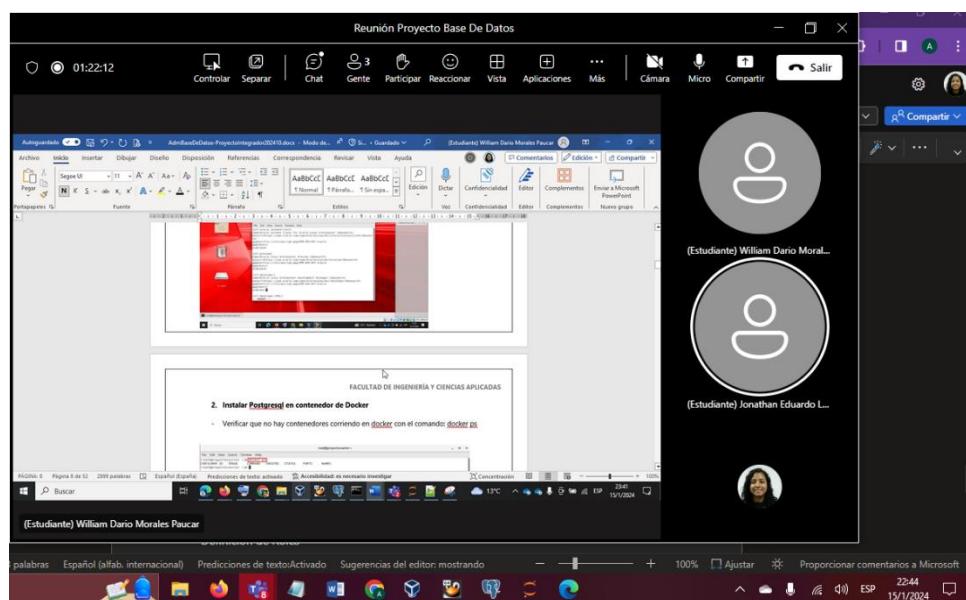


Tercera reunión el 13 de enero

FACULTAD DE INGENIERÍA Y CIENCIAS APLICADAS



Cuarta reunión el 15 de enero



7.- Criterio

Ideas aportadas para la implementación de la solución por cada participante

Porcentaje de Cumplimiento

100%

Evidencia(s)

8.- Criterio

Aporte individual para la consecución de los Objetivos de Trabajo

Porcentaje de Cumplimiento

Evidencia(s)

Nos aseguramos de comprender claramente los objetivos del proyecto. Si hay alguna

FACULTAD DE INGENIERÍA Y CIENCIAS APLICADAS

ambigüedad, buscamos aclaraciones para garantizar un entendimiento completo mediante búsquedas en internet o comunicándonos con el docente.

Cada miembro tuvo un compromiso personal con los objetivos del equipo. Demuéstranos interés y entusiasmo por lograr el éxito de la correcta implementación y ejecución del proyecto.

9.- Criterio

Aporte individual para la resolución de posibles conflictos

Porcentaje de Cumplimiento

Evidencia(s)

Para la realización de este proyecto tuvimos una comunicación abierta y honesta, establecimos un canal donde todos los miembros del equipo nos sentimos cómodos expresando nuestras opiniones y preocupaciones sin temor a represalias, estando conscientes de que en algún tema uno tiene más conocimiento que otro, pero cada miembro fue paciente y ante cualquier problema nos reunímos y buscábamos la solución.

Cada miembro del grupo aporto una idea para poder dar solución al presente proyecto, como, por ejemplo: Cuando tuvimos una dificultad que a uno de nuestros compañeros no se podía conectar la base de datos contenerizada al pg admin debido a un error de conexión en virtual box, ante este problema nos reunimos y le dimos solución en conjunto.

FACULTAD DE INGENIERÍA Y CIENCIAS APLICADAS