

Administración de Base de Datos

Documentación

Luis Mario Cervantes Suárez
Jessica Peña Montero
Jonathan Peña Pérez

REDES Y SERVICIOS DE COMPUTO – UNIVERSIDAD VERACRUZANA

Contenido

Introducción	3
Tablas de direcciones IP	4
Configuración inicial de servicios	5
Establecer la dirección IP estática a nuestro OMV	5
Acceder a la página de OMV remotamente	7
Conexión por SSH a servidores	7
Instalación de Zero Tier en el servidor NAS	8
Conectarse a el servicio de Zero Tier en Windows	9
Instalación y configuración de Raspberry PI	10
Instalación del servicio de base de datos (Maria DB)	11
Instalación de Zero Tier en el Servidor con el servicio de base de datos	12
Conectando a la base de datos	13
Configuración de OMV	16
Creación de grupos	16
Creación de usuarios	17
Sistema de archivos	18
Creación de carpetas	20
Tipos de permisos	21
Compartir carpetas	24
Gestor de contraseñas	26
Check List	26
Permisos de usuarios	27
Diseño de la base de datos	27
Diagrama Modelo Entidad Relación	27
Diagrama Relacional	29
Normalización	30
Copias de seguridad para la base de datos	35
Montar una carpeta compartida de OMV-NAS	35
Desarrollo de scripts	36
Selección del lenguaje	37
Enviar correos electrónicos desde el CLI del servidor	37
Llamadas al sistema para respaldo	40
Auditoria y Monitoreo	45

Visualizar log de los usuarios el servicio de base de datos	46
Uso de mytop	47
Tipos de respaldo	48
Interfaz gráfica del sistema	49
Selección del lenguaje.....	49
Capturas del trabajo.....	50
Instalación y Configuración de ngrok	51
Conexión del sistema a la base de datos	52
Resultado obtenido.....	53
Conclusión.....	53

Introducción

En la era de la información, el acceso a la tecnología se ha convertido en una necesidad para desarrollar diversas actividades cotidianas y, sin duda, la educación no ha sido ajena a este proceso. La utilización de diferentes herramientas tecnológicas ha permitido que la educación evolucione y se adapte a las nuevas necesidades del mundo moderno.

En este contexto, la Facultad de Estadística e Informática ha planteado un proyecto para crear un servidor que permita a los estudiantes acceder a información relevante sobre sus horarios y la disponibilidad de las aulas en tiempo real, desde cualquier lugar y en cualquier momento.

Para llevar a cabo este proyecto, se ha adquirido el hardware necesario, que incluye un router de la marca tp link, un servidor NAS y una Raspberry Pi 3b+. Además, se cuenta con un proveedor de internet y un modem Huawei, que permitirán la conectividad de los diferentes dispositivos en el sistema.

El objetivo principal de este proyecto es ofrecer a los estudiantes una plataforma en línea donde puedan consultar información relevante para su formación académica, y para lograrlo, se utilizarán diferentes herramientas tecnológicas.

Entre ellas, se empleará el servicio de VPN de ZeroTier para asegurar la privacidad de la información transmitida. Además, se configurará el servidor con una base de datos que almacenará información sobre los horarios y la disponibilidad de las aulas, lo que permitirá la gestión eficiente de la información y evitará conflictos de reserva de espacios.

En conclusión, este proyecto busca integrar tecnología y educación para ofrecer a los estudiantes de la Facultad de Estadística e Informática una herramienta que les permita acceder a información relevante de manera remota, cómoda y segura. La utilización de herramientas como servidores y servicios de red garantizará un acceso eficiente y seguro a la información, lo que sin duda contribuirá al proceso educativo de los estudiantes de esta facultad.

Tablas de direcciones IP

A continuación, se presentarán las direcciones IP de los dispositivos a utilizar en el proyecto, organizadas en una tabla para una mejor visualización. Se indicará la dirección IP del router, del OMV-NAS y del Rserver dentro de la red del router, así como la dirección IP del modem y del router proporcionadas por el ISP.

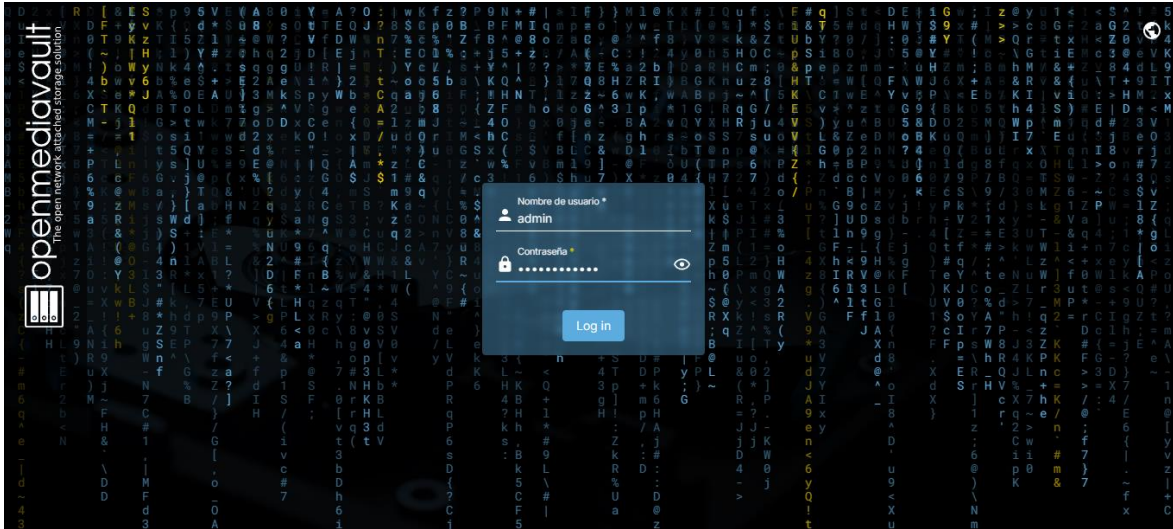
Dispositivo	Dirección IP
Router (Red interna)	192.168.0.1
Omv-nas	192.168.0.2
Rserver	192.168.0.5
Modem (ISP)	192.168.100.1
Router (ISP)	192.168.100.153

Configuración inicial de servicios

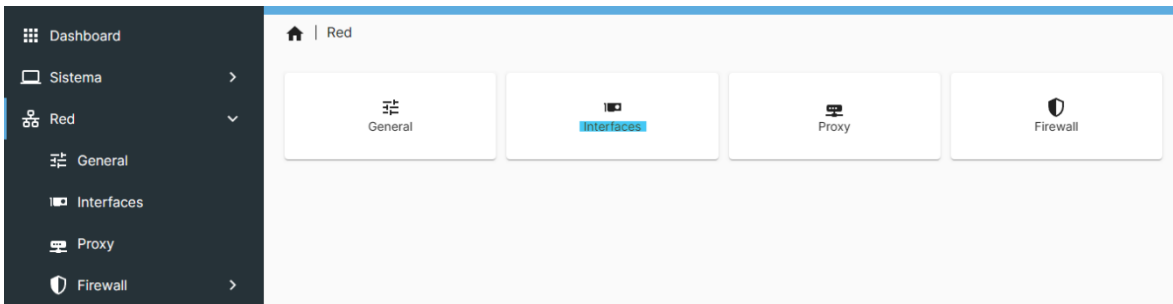
En este documento se presentarán las configuraciones iniciales para los servidores que se utilizarán en un proyecto específico.

Establecer la dirección IP estática a nuestro OMV

1. Entrar a la página de administración de OMV (esto lo hacemos escribiendo la dirección IP que tiene nuestro servidor OMV-nas en la barra de direcciones de nuestro navegador web) y escribiremos nuestras credenciales de acceso.



2. Seguido a eso, no dirigiremos a la sección de red y entraremos a la configuración de interfaces



3. Cuando estemos en la configuración de la interfaz de red, seleccionaremos nuestra interfaz y daremos clic al botón de lápiz para modificar algunos apartados y establecer la dirección ip estática.

Dispositivo ^	Metodo ↕	Dirección ↕	Mascara de Red	Gateway ↕	MTU ↕	WOL ↕	Tags ↕
enp9s0	IPv4: Estatico IPv6: Deshabilitado	IPv4: 192.168.0.2 IPv6: -	IPv4: 255.255.255.0 IPv6: 64	IPv4: 192.168.0.1 IPv6: -	0	✓	
1 Seleccionado / 1 total							

4. Por último, llenamos la información con los parámetros correspondientes a nuestros dispositivos físicos.

IPv4

Metodo *
Estatico

Dirección
192.168.0.2

Mascara de Red
255.255.255.0

Gateway
192.168.0.1

Metric
10

Propiedades avanzadas

Servidores DNS
8.8.8.8

Dirección IP de los servidores DNS que deben resolver los nombres de host.
Buscar dominios

Dominios usados para resolver los nombres de host.
MTU
0

The maximum transmission unit in bytes to set for the device. Set to 0 to use the default value.

☒ WOL

Acceder a la página de OMV remotamente

Ahora que hemos configurado nuestra dirección IP estática en OMV, podemos continuar con la siguiente etapa: realizar pruebas para acceder remotamente al servidor NAS mediante el servicio VPN que ofrece ZeroTier.

Antes de iniciar con las pruebas, debemos instalar el servicio ZeroTier en nuestro servidor NAS, para ello, debemos conectarnos por el protocolo SSH a nuestro servidor OMV-nas.

Conexión por SSH a servidores

Para acceder a la terminal del servidor, utilizaremos el protocolo SSH (Secure Shell), el cual nos permite conectarnos de forma segura y encriptada a través de una red. Por defecto, el puerto que utilizaremos para esta conexión será el 22. Sin embargo, es importante tener en cuenta que en algunos casos puede ser necesario cambiar este puerto por uno diferente para garantizar la seguridad del sistema. Esto es especialmente relevante cuando se accede al servidor de forma remota, desde una ubicación fuera de la red local. En cualquier caso, se tomarán las medidas necesarias para garantizar la integridad y la privacidad de la información transmitida a través del protocolo SSH.

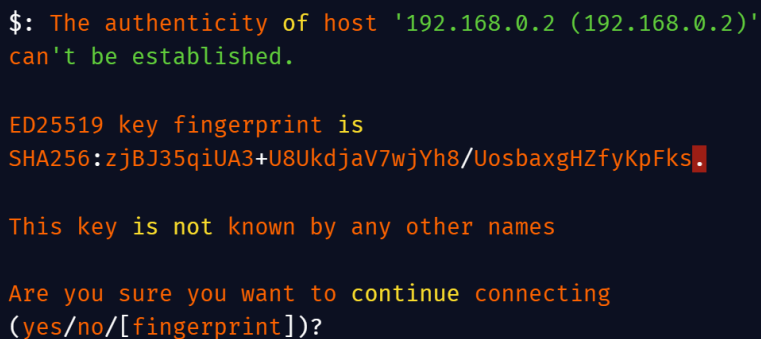
Para ello, debemos escribir en la terminal la siguiente instrucción:



```
$: ssh root@192.168.0.2
```

Donde "root" es el nombre de usuario y "192.168.0.2" es la dirección IP del servidor al cual deseamos acceder. De esta manera, podremos acceder a los recursos y servicios del servidor de manera remota desde una terminal.

Sí nos aparece el siguiente mensaje:



```
$: The authenticity of host '192.168.0.2 (192.168.0.2)'  
can't be established.  
  
ED25519 key fingerprint is  
SHA256:zjBJ35qiUA3+U8UkdjaV7wjYh8/UosbaxgHZfyKpFks.  
  
This key is not known by any other names  
  
Are you sure you want to continue connecting  
(yes/no/[fingerprint])?
```


Debemos escribir "yes" y presionar Enter. Posteriormente, se nos solicitará ingresar la contraseña del usuario que utilizamos para acceder al servidor. Una vez que hayamos ingresado la contraseña correcta, podremos acceder al servidor sin problema alguno mediante SSH.



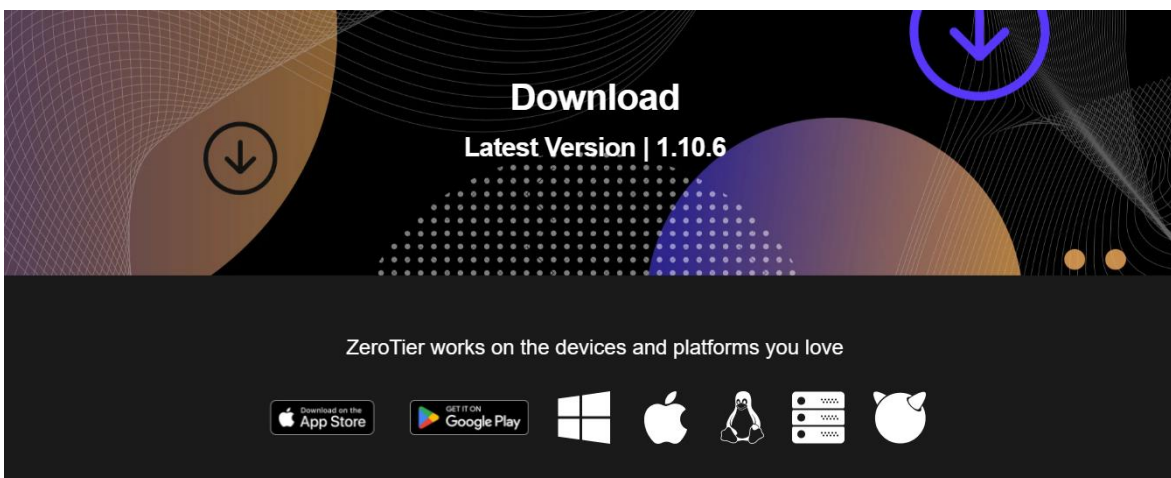
Nota Importante

Es importante destacar que además del protocolo SSH, existe un programa llamado PuTTY que nos permite automatizar el proceso de conexión. Con PuTTY, es posible guardar perfiles de conexión y utilizar una interfaz de usuario gráfica amigable e intuitiva que simplifica la tarea de conexión. Esto hace que el proceso de conexión sea más rápido y sencillo para los usuarios.

Instalación de Zero Tier en el servidor NAS

Zero Tier es un software de red privada virtual que permite crear una red segura y privada entre varios dispositivos, lo que facilita la conexión remota a nuestro servidor NAS.

1. Para hacer la instalación debemos ir a [la página oficial de Zero Tier](#) y buscar el botón de descarga para nuestro sistema operativo.



2. En la sección de descarga tendremos diferentes versiones para S.O, aquí debemos seleccionar el que mejor se adecue a nuestro servidor NAS
3. Si está dispuesto a confiar en SSL para autenticar el sitio, se puede realizar una instalación de una línea con:

```
$: curl -s https://install.zerotier.com | sudo bash
```

4. Nos conectamos por SSH a nuestro servidor omv-nas y escribimos la instrucción

Una vez descargado el servicio, escribiremos el siguiente comando unirse a una red en Zero Tier.

```
$: sudo zerotier-cli join idDeLaRed
```

En la web de administración debemos aceptar la conexión del nuevo dispositivo en la red y finalmente verificamos el estado de conexión en nuestro servidor escribiendo el siguiente comando:

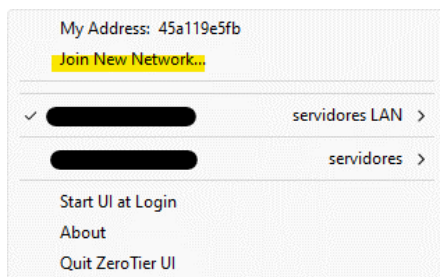
```
$: sudo zerotier-cli listnetwork
```

Y con esta configuración tenemos nuestro servicio de VPN con hamachi ejecutándose correctamente.

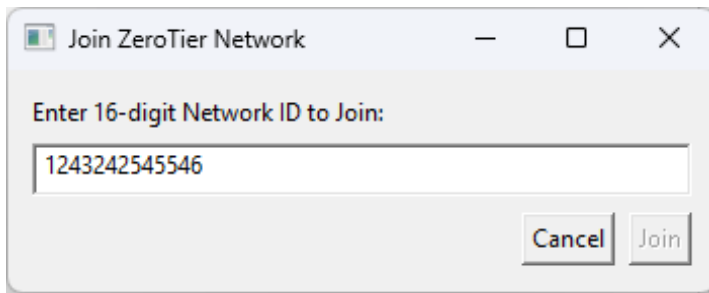
Conectarse a el servicio de Zero Tier en Windows

Para poder conectarnos a Zero Tier desde un equipo con sistema operativo Windows, es necesario descargar el programa desde su página oficial (ver la sección "Instalación de Zero Tier"), instalarlo y registrarnos en el servicio.

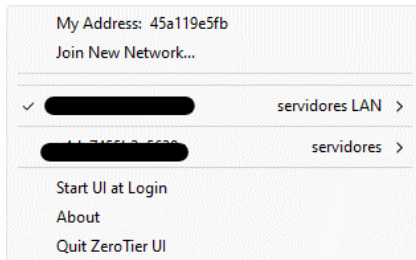
Una vez que hayamos realizado el registro, debemos conectarnos a la red creada en nuestro servidor NAS. Para ello, es necesario hacer clic en "Unirse a una nueva red " y seleccionar la red a la que deseamos conectarnos.



Después, tendremos que ingresar el id de acceso que se asignó previamente en el administrador de Zero Tier

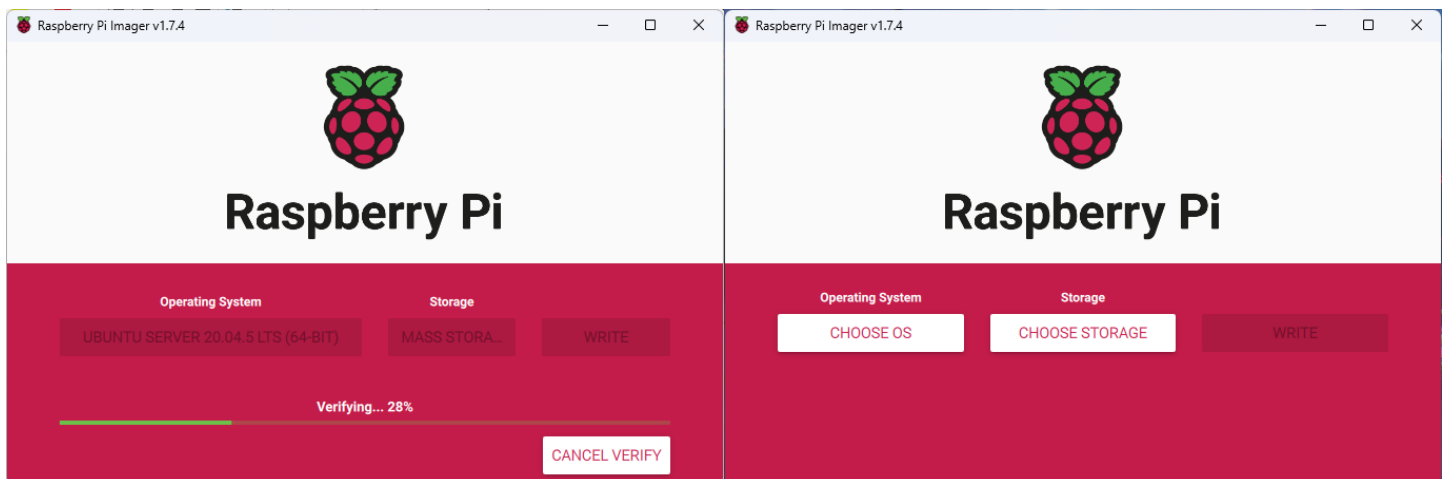


Si todos los pasos se aplicaron correctamente, deberíamos visualizar una pantalla similar a la siguiente imagen:



Instalación y configuración de Raspberry PI

Para el servidor principal, hemos decidido utilizar una Raspberry Pi debido a su bajo costo y gran versatilidad. La Raspberry Pi se encargará de ejecutar los servicios principales del proyecto. Para instalar el sistema operativo en la Raspberry Pi, utilizaremos una tarjeta microSD y el programa Pi Imager, que nos permitirá grabar la imagen del sistema operativo en la tarjeta de manera fácil y rápida. Con este proceso de instalación, lograremos que la Raspberry Pi esté lista para ejecutar los servicios requeridos en el proyecto.



Sitio de descarga: [Raspberry Pi OS – Raspberry Pi](#)

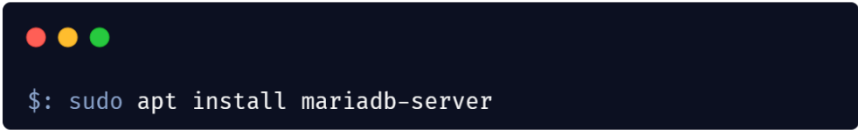
Una vez boteada nuestra sd la ponemos en nuestros raspberry pi y esperamos que termine de instalarse el sistema

Para controlar nuestro rserver podemos hacerlo mediante ssh. (consultar dirección ip pag. #3)

Instalación del servicio de base de datos (Maria DB)

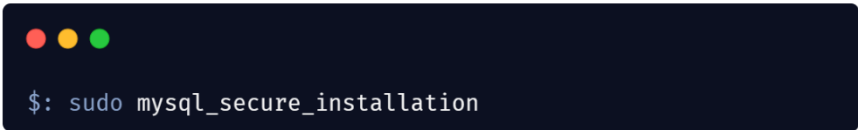
En el servidor Rserver, utilizaremos el servicio de MariaDB para diseñar y administrar nuestra base de datos. Para poder utilizar este servicio, lo primero que haremos será instalarlo en el servidor. De esta manera, podremos comenzar a trabajar en la creación y gestión de la base de datos del proyecto de manera eficiente y segura.

Para instalar el servicio de Maria DB debemos ejecutar el siguiente comando en la terminal:



```
$: sudo apt install mariadb-server
```

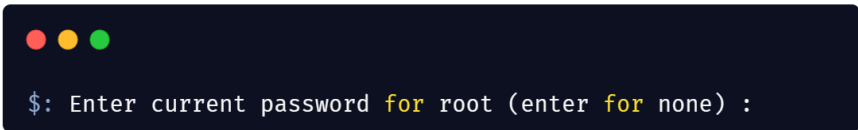
Una vez instalado el servicio de Maria DB server, necesitamos aplicar instalación segura, para ello ejecutamos el siguiente comando:



```
$: sudo mysql_secure_installation
```

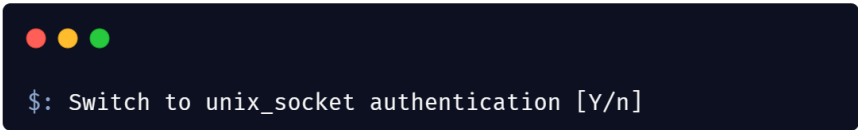
Y seguimos la siguiente configuración **RECOMANDADA**

1. Damos enter para dejar vacia la opcion.



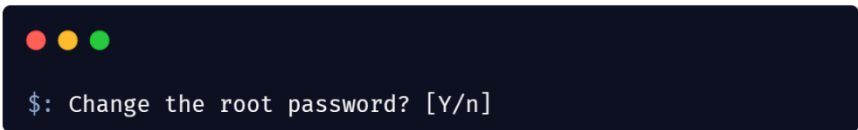
```
$: Enter current password for root (enter for none) :
```

2. Decimos que **no** queremos establecer una contraseña, dado que ya tenemos una para nuestro usuario root



```
$: Switch to unix_socket authentication [Y/n]
```

3. Decimos que **no** queremos cambiar la contraseña



```
$: Change the root password? [Y/n]
```

- Decimos que **si** queremos eliminar usuarios anónimos y que si queremos deshabilitar el acceso remoto

```
$: Remove anonymous users? [Y/n]
```

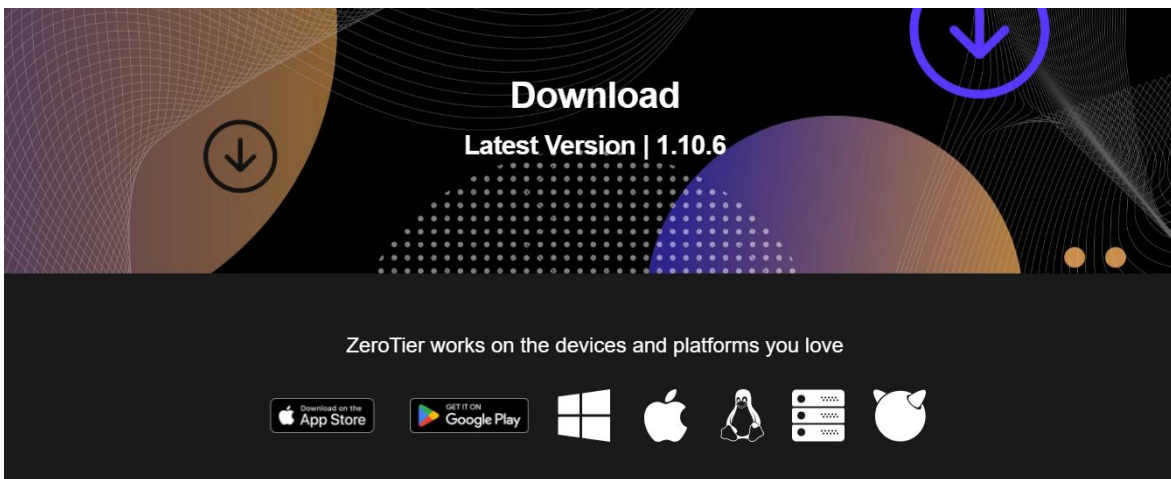
- Y finalmente **aceptamos** que se remuevan la base de datos de prueba y que se restablezcan los permisos sobre las tablas

```
$: Remove test database and access to it? [Y/n]
```

Instalación de Zero Tier en el Servidor con el servicio de base de datos

Zero Tier es un software de red privada virtual que permite crear una red segura y privada entre varios dispositivos, lo que facilita la conexión remota a nuestro rserver.

- Para hacer la instalación debemos ir a [la página oficial de Zero Tier](#) y buscar el botón de descarga para nuestro sistema operativo.



- En la sección de descarga tendremos diferentes versiones para S.O, aquí debemos seleccionar el que mejor se adecue a nuestro rserver
- Si está dispuesto a confiar en SSL para autenticar el sitio, se puede realizar una instalación de una línea con:

```
$: curl -s https://install.zerotier.com | sudo bash
```

4. Nos conectamos por SSH a nuestro servidor omv-nas y escribimos la instrucción

Una vez descargado el servicio, escribiremos el siguiente comando unirse a una red en Zero Tier.

```
$: sudo zerotier-cli join idDeLaRed
```

En la web de administración debemos aceptar la conexión del nuevo dispositivo en la red y finalmente verificamos el estado de conexión en nuestro servidor escribiendo el siguiente comando:

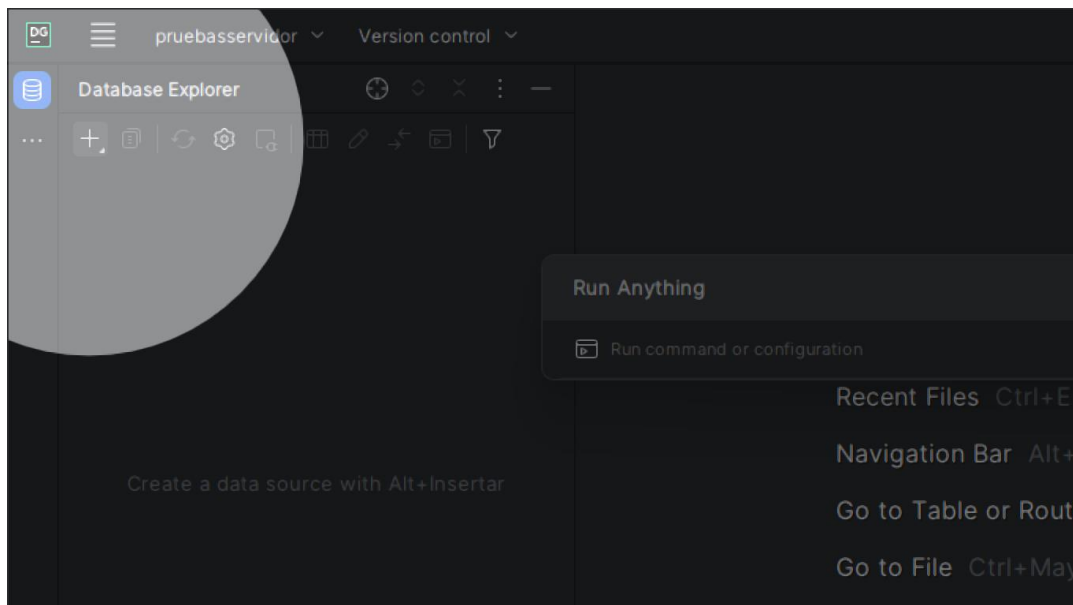
```
$: sudo zerotier-cli listnetwork
```

Y con esta configuración tenemos nuestro servicio de VPN con hamachi ejecutándose correctamente.

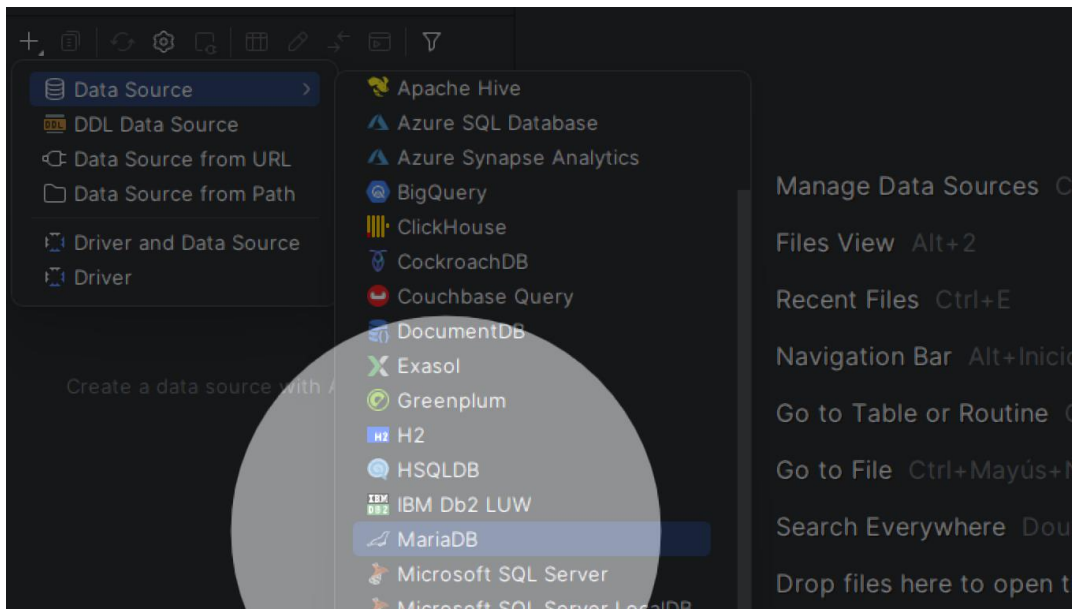
Conectando a la base de datos

Para realizar una conexión a nuestra base de datos, primero debemos descargar el programa DataGrip desde su sitio web oficial: Download DataGrip: [Download DataGrip: Cross-Platform IDE for Databases & SQL \(jetbrains.com\)](https://www.jetbrains.com/datagrip/)

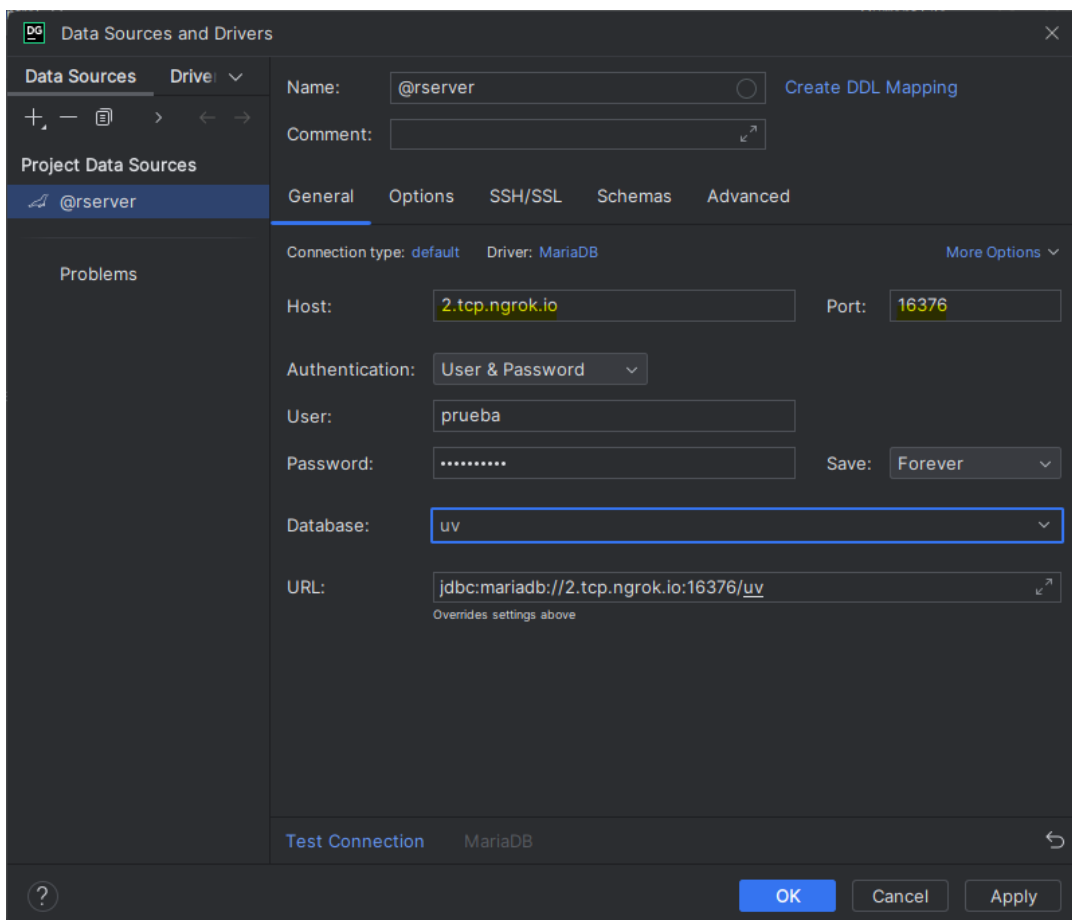
Una vez descargado e instalado el programa, podemos agregar un servicio de base de datos haciendo clic en el botón "+".



Posteriormente, en la sección "data source", seleccionamos MariaDB.

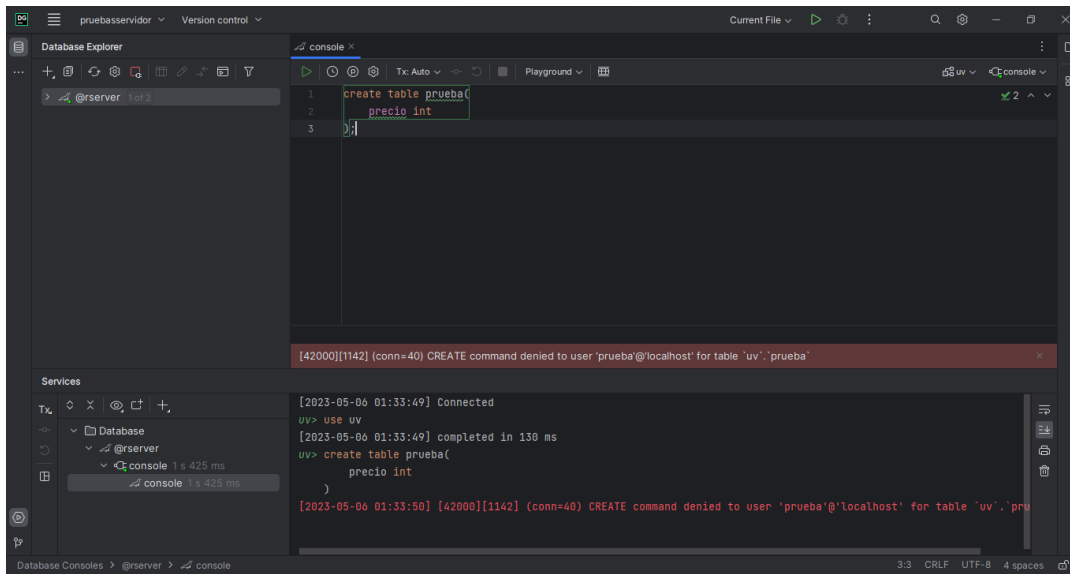


Finalmente, debemos llenar el cuadro con la información de nuestro rserver.



(En host es la dirección de nuestro servidor y en puerto va el numero de puerto del servicio, en muchos casos y por defecto es 3306)

Una vez establecida la conexión, podemos probar realizando alguna consulta en nuestra base.

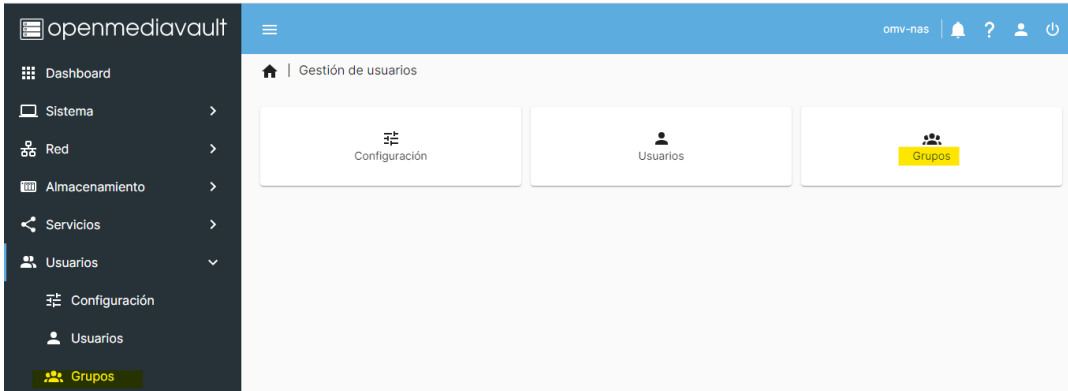


Aunque en la anterior imagen se muestra un error, se debe a los permisos del usuario "prueba", quien solo tiene permisos de SELECT y no de CREATE en la base de datos.

Configuración de OMV

Creación de grupos

Para compartir carpetas en OpenMediaVault, es necesario crear grupos de usuarios en primer lugar. Para hacer esto, se deben seguir los siguientes pasos:



Después de haber ingresado a la sección de "Grupos" en OpenMediaVault, debemos hacer clic en el botón "+" ubicado en la esquina superior derecha para crear un nuevo grupo.

Nombre ^	Miembros ^	Tags ^
TeamAdmonBD	jessica, mariosuarez	equipo de administración de base de datos
Imcservices	mariosuarez	Grupo de luis mario

0 Seleccionado / 2 total

Al crear un nuevo grupo en OpenMediaVault, se deben proporcionar los datos correspondientes según sea necesario, tales como el ID/nombre del grupo y la lista de usuarios que pertenecerán a él(revisar creación de usuarios) aunque no es obligatorio definirlos al momento.

Nombre *
Prueba

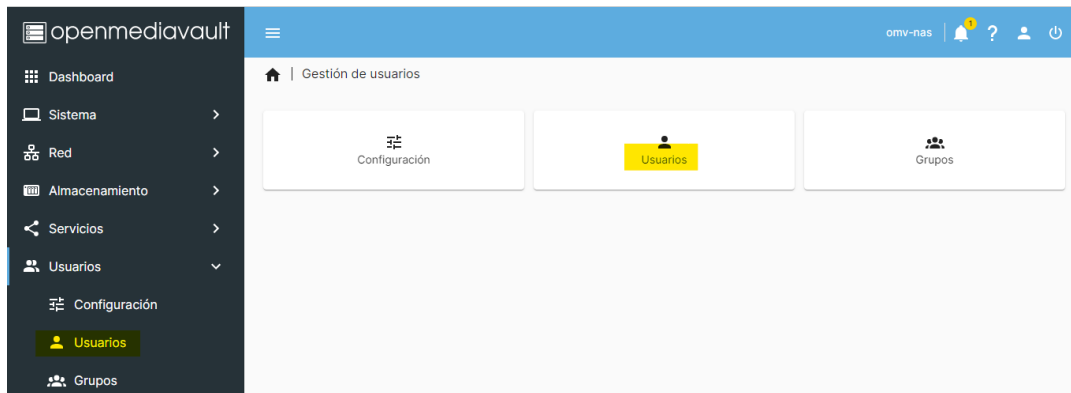
Miembros
Seleccionar miembros ...

Tags

Cancelar Salvar

Creación de usuarios

Luego de haber creado los grupos necesarios, podemos proceder a crear usuarios en OpenMediaVault siguiendo los pasos correspondientes.



Una vez dentro de la sección de "Usuarios" en OpenMediaVault, debemos hacer clic en el botón "+" ubicado en la esquina superior derecha para agregar un nuevo usuario.

Nombre ^	Email ^	Grupos ^	Tags ^
jessica		TeamAdmonBD, users	Adminstradora de servidores
mariosuarez	l.mario.cs31@gmail.com	lmcservices, TeamAdmonBD, users	administrador de servidores
0 Seleccionado / 2 total			

Para crear un nuevo usuario en OpenMediaVault, es necesario ingresar los datos correspondientes según las necesidades, como el nombre de usuario, la contraseña, el grupo al que pertenecerá, entre otros.

Nombre *

JesPrueba

Email

prueba@gmail.com

Contraseña *

.....

👁

Confirmar contraseña

.....

👁

Shell

/bin/sh

▼

Grupos

Prueba

▼

Claves públicas SSH

+

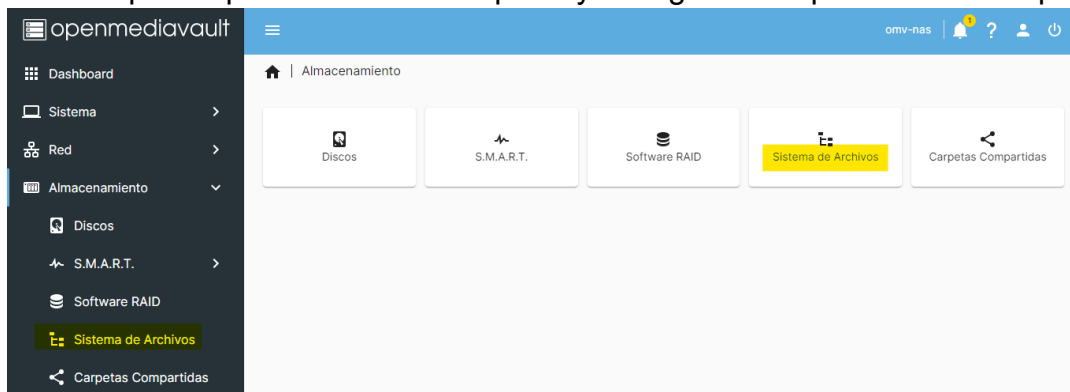
🗑

No hay datos que mostrar.

☐ inhabilita la modificación de la cuenta

Sistema de archivos

Para compartir carpetas en OpenMediaVault, debemos dirigirnos primero a la sección de "Almacenamiento" y posteriormente a la sección de "Sistema de archivos", donde podremos acceder a las carpetas que deseamos compartir y configurar sus permisos correspondientes.



Una vez en la sección de "Sistema de archivos" de OpenMediaVault, debemos hacer clic en el botón "+" ubicado en la esquina superior derecha para agregar una nueva partición. Luego, seleccionamos el tipo de sistema de archivos deseado, en este caso, "ext4".

Dispositivo	Tipo	Disponible	Usado	Montados	Referenciado	Estado
/dev/sdb1	EXT4	457.00 GiB	367.26 MiB	✓	✓	Online
/dev/sdc1	EXT4	7.22 GiB	9.43 MiB	✓		Online

0 Seleccionado / 2 total

Tipo
EXT4

Dispositivo *

Seleccionar un dispositivo ...

Campo requerido.

Cancelar Salvar

BTRFS

EXT3

EXT4

F2FS

JFS

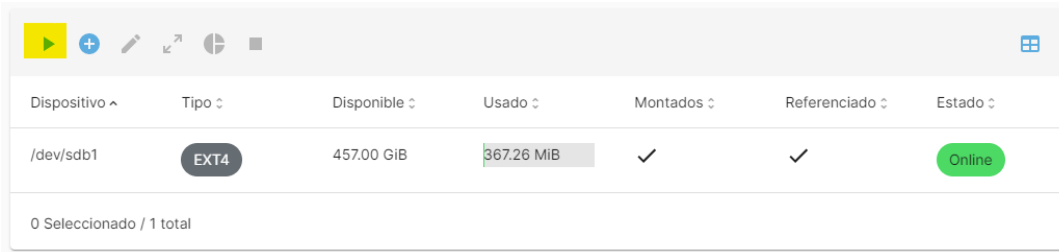
XFS



Nota Importante

"EXT4" es un tipo de sistema de archivos utilizado en sistemas operativos Linux. Es una versión mejorada del sistema de archivos "ext3" y es uno de los sistemas de archivos más comunes en distribuciones de Linux modernas. "ext4" proporciona mejoras en la administración de archivos, mayor capacidad de almacenamiento y soporte para características avanzadas, como el cifrado de archivos y la fragmentación multihilo. Es compatible con una amplia gama de dispositivos de almacenamiento, incluyendo discos duros y unidades de estado sólido.

Después de haber seleccionado el tipo de sistema de archivos "ext4", es necesario montarlo para que esté disponible para su uso. Para hacerlo, podemos seguir los siguientes pasos:



Dispositivo ^	Tipo ▾	Disponible ▾	Usado ▾	Montados ▾	Referenciado ▾	Estado ▾
/dev/sdb1	EXT4	457.00 GiB	367.26 MiB	✓	✓	Online

0 Seleccionado / 1 total

Una vez que hayamos seleccionado el tipo de sistema de archivos y montado la partición correspondiente, debemos llenar los campos necesarios con los datos requeridos para la configuración de la partición, como el nombre de la partición, la ruta de montaje, el tamaño y otros parámetros relevantes.



Sistema de Archivos *

/dev/sdc1 [EXT4, 7.45 GiB]

Sistema de archivos a montar

Advertencia de umbral de uso *

80%

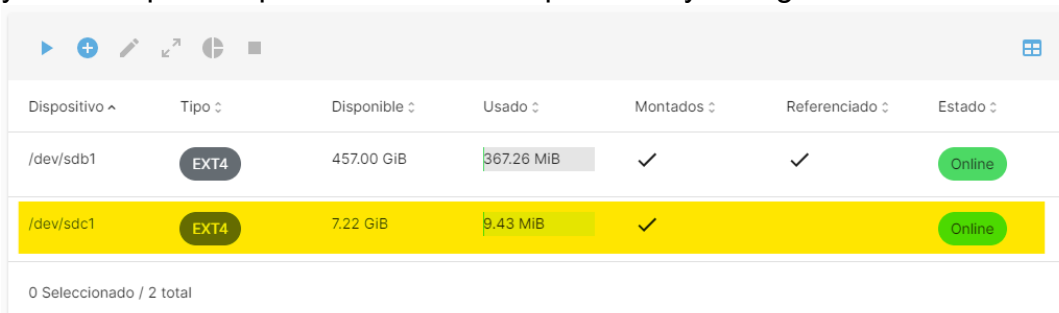
Mandar una notificación cuando el espacio en uso, sobrepase el límite establecido.

Tags

Memoria Azul X

Cancelar Salvar

Finalmente, después de haber realizado todos los pasos necesarios para compartir la carpeta en OpenMediaVault, la partición correspondiente debe aparecer en la sección de "Sistema de archivos" y estar disponible para su uso con los permisos y configuraciones establecidos.

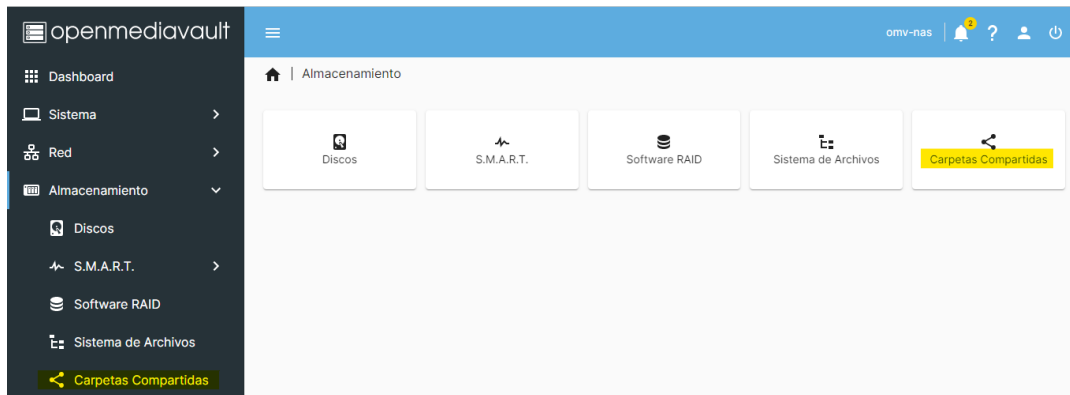


Dispositivo ^	Tipo ▾	Disponible ▾	Usado ▾	Montados ▾	Referenciado ▾	Estado ▾
/dev/sdb1	EXT4	457.00 GiB	367.26 MiB	✓	✓	Online
/dev/sdc1	EXT4	7.22 GiB	9.43 MiB	✓		Online

0 Seleccionado / 2 total

Creación de carpetas

Para crear las carpetas compartidas en OpenMediaVault, debemos dirigirnos a la sección de "Almacenamiento" y posteriormente a la sección de "Carpetas compartidas". Desde allí, podremos crear una nueva carpeta compartida y asignarle los permisos correspondientes posteriormente.



Para crear una nueva carpeta compartida en OpenMediaVault, podemos hacer clic en el botón "+" ubicado en la esquina superior derecha de la sección "Carpetas compartidas", lo que nos permitirá agregar una nueva carpeta y establecer los permisos correspondientes.



Nota Importante

Es importante tener en cuenta que los permisos establecidos al crear una carpeta compartida en OpenMediaVault deben ser coherentes con los permisos asignados posteriormente a los usuarios y grupos que tengan acceso a la carpeta, para garantizar un correcto funcionamiento y seguridad en el acceso a los archivos y carpetas compartidos.

Luego de haber hecho clic en el botón "+" para agregar una nueva carpeta compartida en OpenMediaVault, debemos llenar los campos correspondientes con los datos necesarios para la configuración de la carpeta, como el nombre de la carpeta, la ruta de acceso, los permisos, entre otros.

Nombre *
AdmonBD

Sistema de Archivos *
/dev/sdc1 [Memoria Azul]

El sistema de archivos en el que será creada la carpeta compartida.

Ruta relativa *
AdmonBD_Docs/

La ruta relativa de la carpeta para compartir. La carpeta especificada se creará si no existe.

Permisos *
Administrador: Lectura/Escritura, Usuarios:Lectura/Escritura, Otros: Solo lectura

Modo de archivo de la ruta a las carpetas compartidas.

Tags
Archivos del proyecto X

Cancelar Salvar

Tipos de permisos

Una vez creada la carpeta compartida en OpenMediaVault, podemos seleccionar la carpeta correspondiente y dirigirnos a la sección de "Lista de control de acceso" para configurar los permisos y acceso para los usuarios y grupos específicos que deseamos autorizar.

openmediavault

Dashboard
Sistema
Red
Almacenamiento
Discos
S.M.A.R.T.
Software RAID
Sistema de Archivos
Carpas Compartidas

Almacenamiento | Carpetas Compartidas

Nombre	Dispositivo	Ruta relativa	Ruta absoluta	Referenciado	Tags
AdmonBD	/dev/sdc1	AdmonBD_Docs/	/srv/dev-disk-by-uuid-e1783b34-8d2d-44df-be4f-6886c7dc57ad/AdmonBD_Docs		Archivos del Proy
dir_luismariosuarez	/dev/sdb1	FileMarioSuarez/	/srv/dev-disk-by-uuid-b2a20bcb-350f-470d-980a-12e67669c259/FileMarioSuarez	✓	Archivos de luis r

1 Seleccionado / 2 total

Dentro de la sección de "Lista de control de acceso" en OpenMediaVault, podremos establecer los permisos necesarios para cada usuario o grupo de usuarios que tenga acceso a la carpeta compartida. Para facilitar este proceso, se proporcionará información sobre la función de cada permiso para ayudar a seleccionar los que correspondan con las necesidades y requerimientos específicos de la carpeta compartida.

A continuación se detalla el propósito de cada permiso en la sección de "Lista de control de acceso" en OpenMediaVault:

- "Lectura": Permite al usuario o grupo visualizar el contenido de la carpeta compartida.

- "Escritura": Permite al usuario o grupo crear, modificar o eliminar archivos y carpetas dentro de la carpeta compartida.
- "Ejecución": Permite al usuario o grupo ejecutar archivos dentro de la carpeta compartida.
- "Sin acceso": Este permiso es útil cuando se desea negar el acceso a un usuario o grupo específico a la carpeta compartida. Si se selecciona el permiso de "Sin acceso" para un usuario o grupo, no podrán visualizar ni interactuar con el contenido de la carpeta compartida.
- "Propietario": Permite asignar el propietario de la carpeta compartida.
- "Grupo": Permite asignar el grupo de usuarios que tienen acceso a la carpeta compartida.
- "Otros": Permite definir los permisos para otros usuarios que no pertenecen al grupo asignado.



Nota Importante

La opción de "recursión" (también conocida como "recursividad") en la configuración de la lista de control de acceso en OpenMediaVault, permite aplicar los permisos seleccionados no solo a la carpeta compartida en sí, sino también a todas las subcarpetas y archivos contenidos en ella. Si la opción de recursión está activada, se aplicarán los permisos a todas las carpetas y archivos de la jerarquía de archivos debajo de la carpeta compartida. Esto puede ahorrar tiempo y esfuerzo al aplicar permisos a una gran cantidad de archivos y carpetas en un solo paso.

Nombre: AdmonBD [Archivos del Proyecto]

ruta relativa: /

Permisos de Usuario/Grupo

Nombre ^	Tipo ^	Cuenta del sistema	Permisos ^
1002	User		Read/Write Read-only No access
jessica	User		Read/Write Read-only No access
lmcservices	Group		Read/Write Read-only No access

Prueba
Group
Read/Write
Read-only
No access

107 total

Propietario	Permisos
root	Lectura/Escritura/Ejecución
Permisos del propietario.	
Grupo	Permisos
TeamAdmonBD	Lectura/Escritura/Ejecución
Permisos del grupo.	
Otros	
Solo lectura	
Permisos de otros (Ej: usuarios anónimos FTP)	

☒ Reemplazar
Remplazar todos los permisos.
☒ Recursivo
Aplicar los permisos a los archivos y subdirectorios.

Cancelar
Salvar

Una vez que hemos guardado la configuración de la carpeta compartida, podemos volver a la sección de carpetas compartidas para configurar los permisos. Para ello, seleccionamos la carpeta compartida que acabamos de crear y buscamos la opción de "Permisos" en la barra de herramientas.

En la sección de permisos, podemos asignar permisos específicos a los usuarios y grupos que hemos creado anteriormente. Podemos permitir o negar el acceso a la carpeta compartida, la capacidad de modificar o borrar archivos, o simplemente visualizar el contenido de la carpeta.

Es importante tener en cuenta que los permisos que se asignan aquí deben estar en línea con los permisos que hemos establecido previamente en la lista de control de acceso. Por ejemplo, si hemos permitido el acceso de lectura y escritura a un grupo específico en la lista de control de acceso, debemos asignar al mismo grupo los permisos de lectura y escritura en esta sección de permisos.

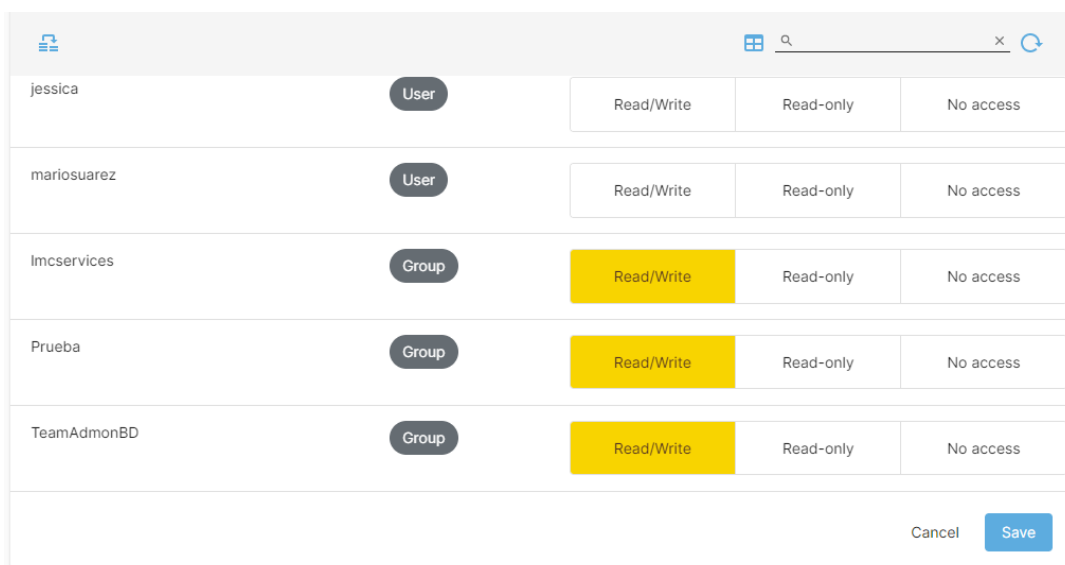
Nombre ^	Dispositivo ^	Ruta relativa ^	Ruta absoluta ^	Referenciado ^	Tags ^
AdmonBD	/dev/sdc1	AdmonBD_Docs/	/srv/dev-disk-by-uuid-e1783b34-8d2d-44df-be4f-6886c7dc57ad/AdmonBD_Docs		Archivos del Proy
dir_luismariosuarez	/dev/sdb1	FileMarioSuarez/	/srv/dev-disk-by-uuid-b2a20bcb-350f-470d-980a-12e67669c259/FileMarioSuarez	✓	Archivos de luis n

1 Seleccionado / 2 total

Entonces, una vez que hemos identificado qué permisos deben asignarse a cada usuario o grupo en particular, podemos proceder a asignarlos en la sección de permisos de la carpeta compartida.

Para asignar permisos, primero debemos seleccionar el usuario o grupo al que deseamos aplicar el permiso y luego seleccionar el tipo de permiso que deseamos asignar (por ejemplo, "Lectura" o "Escritura"). También es posible asignar permisos de forma recursiva a todos los subdirectorios y archivos dentro de la carpeta compartida.

Es importante tener en cuenta que los permisos asignados en esta sección se sumarán a los permisos establecidos en la lista de control de acceso. Por lo tanto, si hemos otorgado permisos de lectura y escritura a un usuario o grupo en la lista de control de acceso, pero solo le otorgamos permisos de lectura en la sección de permisos de la carpeta compartida, el usuario o grupo solo tendrá permisos de lectura para esa carpeta compartida.

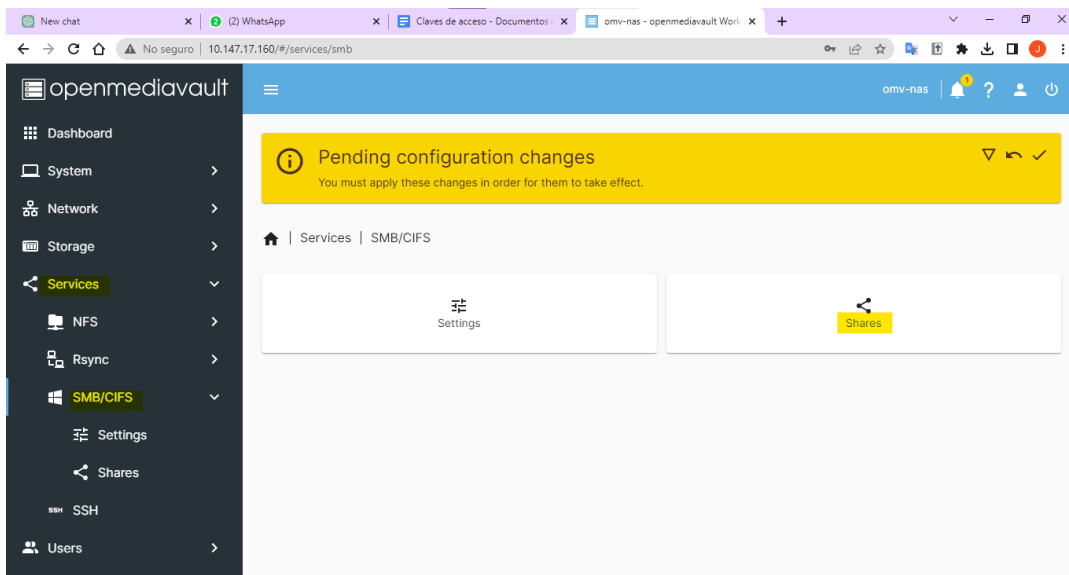


		Read/Write	Read-only	No access
jessica	User			
mariosuarez	User			
Imcservices	Group	Read/Write		
Prueba	Group	Read/Write		
TeamAdmonBD	Group	Read/Write		

Cancel Save

Compartir carpetas

Después, una vez que hemos configurado los permisos de la carpeta compartida, podemos proceder a compartirla a través del protocolo SMB/CIFS para que sea accesible desde otros dispositivos en la red. Para hacerlo, nos dirigimos a la sección de "Servicios" en OpenMediaVault y seleccionamos "SMB/CIFS". Luego, hacemos clic en "Compartir" para crear una nueva compartición.



En la ventana de configuración de la compartición, podemos establecer el nombre de la compartición, seleccionar la carpeta que queremos compartir y asignar los permisos que deseamos otorgar a los usuarios o grupos que accederán a ella.

También es posible establecer configuraciones adicionales, como la habilitación de la compatibilidad con versiones anteriores de SMB o la configuración de opciones avanzadas de seguridad.

Una vez que hayamos configurado todas las opciones necesarias, podemos guardar los cambios y la carpeta compartida estará disponible a través del protocolo SMB/CIFS en la red local.

Enabled ↕	Shared folder ^	Comment ↕	Public ↕	Read-only ↕	Browseable ↕
✓	dir_luismaríasuarez	Archivos de Imcs	No		
0 selected / 1 total					

🏠 | Servicios | SMB/CIFS | Compartidos | Crear

AdmonBD [Archivos del Proyecto]

The location of the files to share.

Comentario

Carpeta que contiene los archivos del proyecto BD

Campo de texto que aparecerá al lado de una compartición cuando los clientes examinen el servidor.

Público

No

If 'Guests allowed' is selected and no login credential is provided, then access as guest. Always access as guest when 'Guests only' is selecting; in this case no password is required to connect to the share. Make sure that the guest user *nobody* can access the files.

☐ Solo lectura
Si se marca esta opción, los usuarios no podrán crear o modificar archivos en el compartido.

☐ Navegable
Controla si la compartición será visible en la lista de comparticiones disponibles en la vista de red de la lista de navegación.

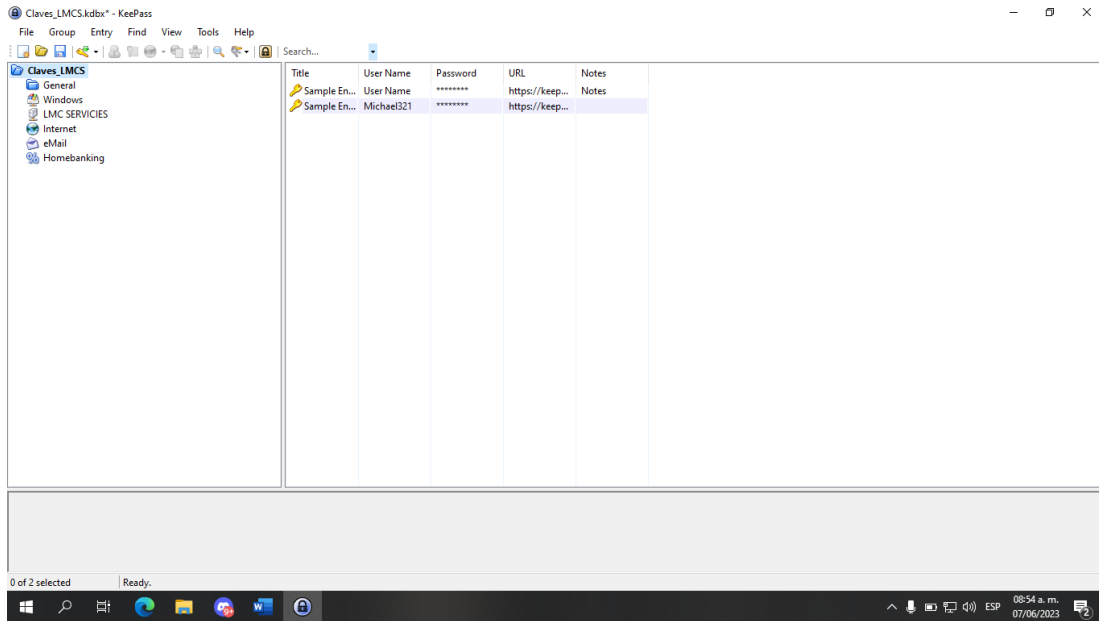
☐ Soporte Time Machine
Habilitar el soporte Time Machine para este compartido.

☐ Heredar ACL
Asegura que, si existen acs por defecto en los directorios padre, serán respetadas cuando se creen nuevos archivos o subdirectorios.

☒ Heredar permisos
Los permisos sobre los nuevos archivos y directorios se establecen por la máscara de creación y la máscara de directorio, pero el parámetro de herencia de permisos prevalece. Puede ser útil especialmente en sistemas con muchos usuarios para permitir una utilización de una misma compartición de una forma flexible para cada usuario.

Gestor de contraseñas

KeePass es un gestor de contraseñas gratuito y de código abierto que te permite almacenar y gestionar tus credenciales de forma segura. Con KeePass, puedes crear contraseñas únicas y complejas para cada sitio web o aplicación que uses, sin tener que recordarlas o escribirlas. KeePass cifra tus contraseñas con un algoritmo seguro y las guarda en una base de datos protegida por una contraseña maestra o una llave de archivo. Así, solo necesitas recordar una contraseña o tener acceso a un archivo para acceder a todas tus contraseñas. KeePass te permite organizar tus contraseñas en grupos, categorías y etiquetas, y también ofrece funciones como el auto-tipeo, el portapapeles seguro, la generación de contraseñas aleatorias y la sincronización con otros dispositivos. KeePass es una herramienta muy útil para mejorar tu seguridad y privacidad en línea.



Check List

Tarea u Objetivo	Realizado
Descargar la versión estable de Open Media Vault	
Descargar la versión estable de raspberry os lite	
Instalar el servicio de openssh-server en ambos servidores	
Instalar el servicio de Zero Tier One en ambos servidores	
Establecer direcciones IP estáticas en los servidores	
Hacer ping a los servidores a las direcciones ip de Zero Tier One	
Comprobar que los servidores tengan conexión a internet	
Instalar el servidor de Maria DB en el servidor de raspberry pi	
Hacer una instalación segura de Maria DB	
Crear usuarios con permisos correspondientes a su rol	
Guardar las claves de acceso en KeePass	
Probar las cuentas de usuario de la base de datos en DataGrip	
Establecer una contraseña segura para la pagina de administración	
Crear grupos en el NAS	

Crear usuarios y asignarlos en grupos en el NAS	
Crear carpetas que se van a compartir	
Establecer los permisos a las carpetas que se van a compartir	
Compartir carpetas con el protocolo CIFS	
Conectar las carpetas compartidas al servidor de raspberry pi	
Comprobar las conexiones de las carpetas compartidas	
Instalar servicio de monitoreo para la base de datos	
Instalar servicio de monitoreo para el servidor	
Realizar pruebas de estrés al servicio de base de datos	

Permisos de usuarios

Los permisos en una base de datos son un aspecto fundamental para garantizar la seguridad y el buen funcionamiento del sistema. Los permisos determinan qué acciones pueden realizar los usuarios o los roles sobre los objetos de la base de datos, como tablas, vistas, procedimientos almacenados, etc. Si los permisos no están bien configurados y asignados, se corre el riesgo de que se produzcan ataques o vulnerabilidades de abuso de privilegios en el servicio y el servidor. Estos ataques pueden comprometer la integridad, la confidencialidad y la disponibilidad de los datos, así como causar daños irreparables al sistema. Por eso, es importante seguir las mejores prácticas para establecer y gestionar los permisos en una base de datos, tales como:

- Seguir el principio de mínimo privilegio, es decir, otorgar solo los permisos necesarios para realizar una tarea específica y no más.
- Crear roles personalizados que agrupen los permisos según las funciones o responsabilidades de los usuarios.
- Asignar los permisos a los roles y no directamente a los usuarios, para facilitar la administración y el control de los accesos.
- Revisar periódicamente los permisos existentes y eliminar o modificar los que ya no sean necesarios o adecuados.
- Auditar y monitorear las actividades de los usuarios y los roles en la base de datos, para detectar y prevenir posibles anomalías o incidentes de seguridad.

Diseño de la base de datos

Diagrama Modelo Entidad Relación

Un modelo entidad-relación (ER) es una técnica utilizada para representar gráficamente los elementos que forman parte de un sistema y las relaciones entre ellos. Es una herramienta

fundamental en el diseño de bases de datos ya que permite visualizar y organizar de manera clara y estructurada la información que se quiere almacenar en la base de datos.

El modelo ER se basa en el concepto de entidad, que puede ser cualquier objeto o cosa en el mundo real que se desea representar en la base de datos, como una persona, un producto o una transacción financiera. Cada entidad se representa mediante un rectángulo en el diagrama ER y se identifica por medio de un atributo o conjunto de atributos únicos, denominado clave.

Las entidades pueden tener relaciones entre sí, que se representan mediante líneas que unen los rectángulos. Estas relaciones pueden ser de diversos tipos, como uno a uno, uno a muchos o muchos a muchos, y se especifican mediante la inclusión de símbolos adicionales en el diagrama ER.

La importancia del modelo ER radica en que permite a los diseñadores de bases de datos comprender con claridad la estructura y las relaciones de los datos que se desean almacenar. Además, al seguir un proceso sistemático para la identificación de entidades, atributos y relaciones, se pueden evitar errores y redundancias en el diseño de la base de datos, lo que a su vez puede mejorar la eficiencia y la calidad de la información almacenada en la base de datos.

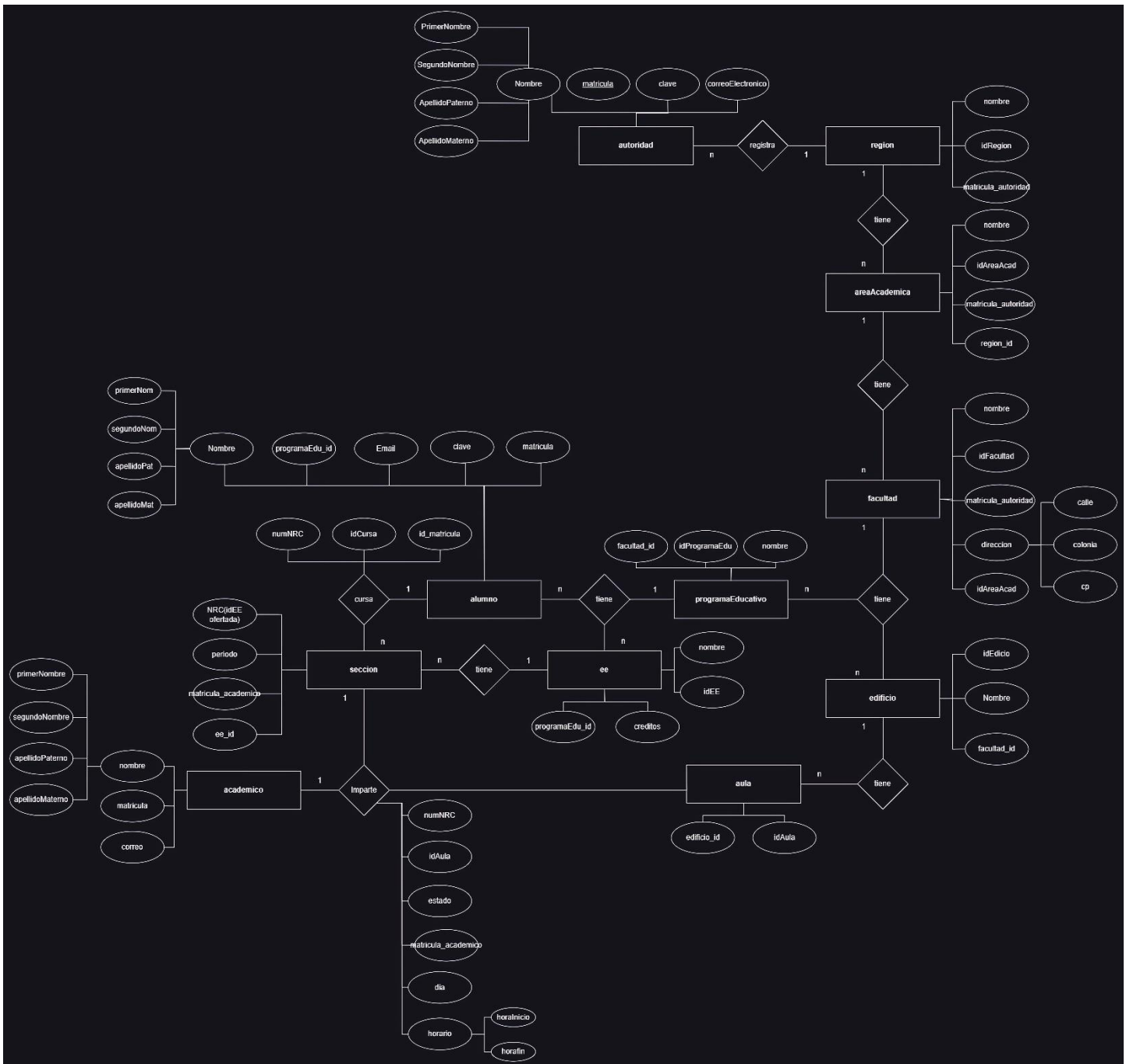


Diagrama Relacional

Un modelo relacional es una estructura de datos que representa la información almacenada en una base de datos en forma de tablas. En un modelo relacional, los datos se organizan en filas y columnas, donde cada fila representa un registro y cada columna representa un atributo o característica de ese registro.

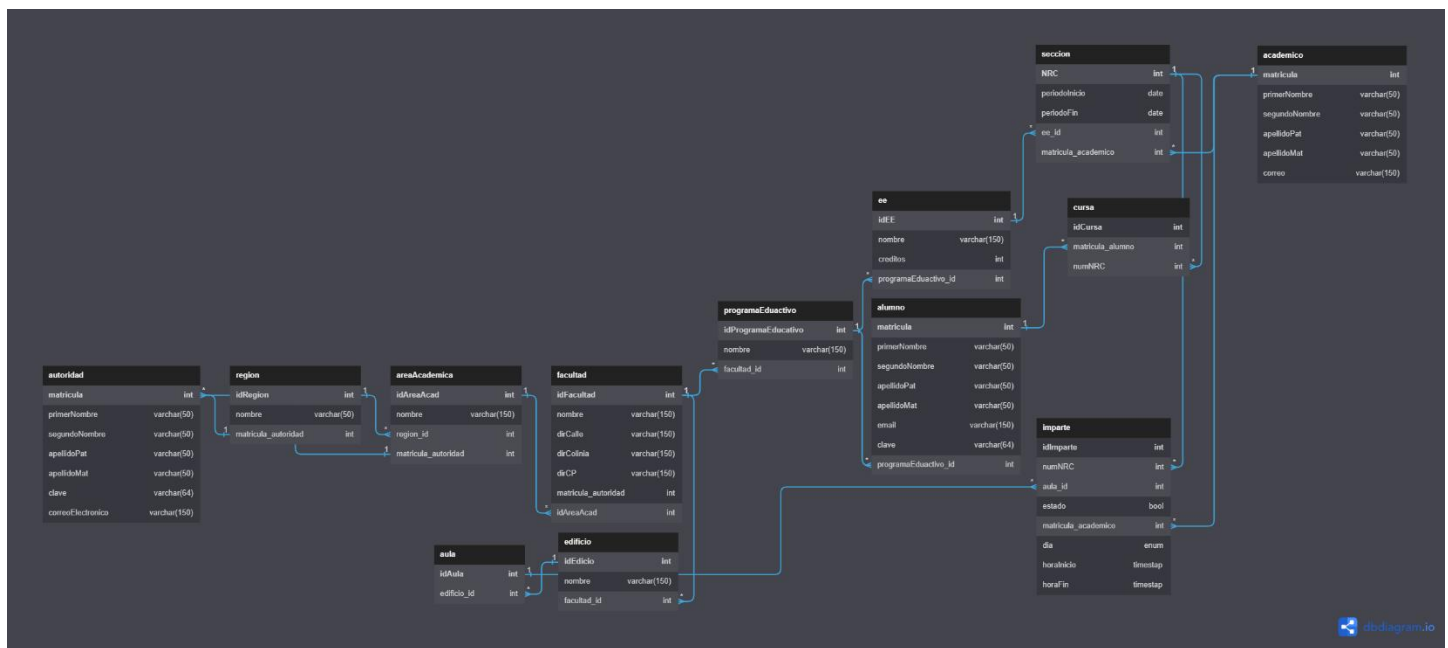
El modelo relacional es importante en el diseño de bases de datos por varias razones. En primer lugar, proporciona una estructura clara y organizada para la información almacenada en la base de

datos. Las tablas facilitan la comprensión de cómo se relacionan los datos entre sí y permiten que los datos se manipulen y se accedan de manera eficiente.

En segundo lugar, el modelo relacional es flexible y escalable. Permite la adición de nuevas tablas y la modificación de las existentes sin afectar al resto de la base de datos. Además, los datos pueden ser indexados para facilitar la búsqueda y el acceso a la información.

En tercer lugar, el modelo relacional permite que los datos sean procesados y analizados con herramientas estándar de consulta, como SQL. Esto facilita la creación de informes y la extracción de información útil de la base de datos.

Por último, el modelo relacional es compatible con múltiples sistemas y aplicaciones. Los datos almacenados en una base de datos relacional pueden ser utilizados por diferentes aplicaciones y sistemas sin necesidad de hacer cambios en la estructura de la base de datos.



Normalización

La normalización en bases de datos es un proceso que se utiliza para diseñar y organizar las tablas de una base de datos relacional de manera eficiente y libre de redundancias. El objetivo principal de la normalización es eliminar la duplicación de datos y asegurar la integridad de la información.

Regla	Descripción
Primera Forma Normal (1FN)	Incluye la eliminación de todos los grupos repetidos.
Segunda Forma Normal (2FN)	Asegura que todas las columnas que no son llave sean completamente dependientes de la llave primaria (PK).
Tercera Forma Normal (3FN)	Elimina cualquier dependencia transitiva. Una dependencia transitiva es aquella en la cual las columnas que no son llave son dependientes de otras columnas que tampoco son llave.

Revisamos el modelo relacional para poder normalizarlo de acuerdo con las reglas de normalización, todo esto antes de implementarlo físicamente:

Tabla autoridad

autoridad	
matricula	int
primerNombre	varchar(50)
segundoNombre	varchar(50)
apellidoPat	varchar(50)
apellidoMat	varchar(50)
clave	varchar(64)
correoElectronico	varchar(150)

La tabla "Autoridad" cumple con la primera forma normal (1FN) debido a que no existen grupos repetidos de datos en sus columnas. En cuanto a la segunda forma normal (2FN), la tabla también cumple, ya que todas las columnas dependen completamente de la clave primaria, que en este caso es la columna "matricula". Por último, en relación a la tercera forma normal (3FN), la tabla sigue cumpliendo, dado que cada columna depende directamente de la clave primaria "matricula" y no presenta dependencias transitivas con otras columnas. En resumen, podemos afirmar que la tabla "Autoridad" está normalizada en cuanto a la 1FN, 2FN y 3FN.

Tabla region

region	
idRegion	int
nombre	varchar(50)
matricula_autoridad	int

La tabla "region" cumple con la primera forma normal (1FN) ya que no existen grupos repetidos de datos en sus columnas. Además, en cuanto a la segunda forma normal (2FN), la tabla también cumple con los requisitos, ya que todas las columnas dependen completamente de la clave primaria "idRegion". Por último, en relación a la tercera forma normal (3FN), la tabla "region" también cumple, dado que cada columna depende directamente de la clave primaria "idRegion" y no presenta dependencias transitivas con otras columnas. En conclusión, podemos afirmar que la tabla "region" está normalizada en cuanto a la 1FN, 2FN y 3FN.

Tabla areaAcademica

areaAcademica	
idAreaAcad	int
nombre	varchar(150)
region_id	int
matricula_autoridad	int

La tabla "areaAcademica" cumple con la primera forma normal (1FN) ya que no existen grupos repetidos de datos en sus columnas. Además, en cuanto a la segunda forma normal (2FN), la tabla también cumple con los requisitos, ya que todas las columnas dependen completamente de la clave primaria "idAreaAcad". Por último, en relación a la tercera forma normal (3FN), la tabla "areaAcademica" también cumple, dado que cada columna depende directamente de la clave primaria "idAreaAcad" y no presenta dependencias transitivas con otras columnas. En conclusión, podemos afirmar que la tabla "areaAcademica" está normalizada en cuanto a la 1FN, 2FN y 3FN.

Tabla facultad

facultad	
idFacultad	int
nombre	varchar(150)
dirCalle	varchar(150)
dirColinia	varchar(150)
dirCP	varchar(150)
matricula_autoridad	int
idAreaAcad	int

La tabla "facultad" cumple con la primera forma normal (1FN) ya que no existen grupos repetidos de datos en sus columnas. Además, en cuanto a la segunda forma normal (2FN), la tabla también cumple con los requisitos, ya que todas las columnas dependen completamente de la clave primaria "idFacultad". Por último, en relación a la tercera forma normal (3FN), la tabla "facultad" también cumple, dado que cada columna depende directamente de la clave primaria "idFacultad" y no presenta dependencias transitivas con otras columnas. En conclusión, podemos afirmar que la tabla "facultad" está normalizada en cuanto a la 1FN, 2FN y 3FN.

Tabla programaEducativo

programaEduactivo	
idProgramaEducativo	int
nombre	varchar(150)
facultad_id	int

La tabla "programaEducativo" cumple con la primera forma normal (1FN) ya que no existen grupos repetidos de datos en sus columnas. Además, en cuanto a la segunda forma normal (2FN), la tabla también cumple con los requisitos, ya que todas las columnas dependen completamente de la clave primaria "idProgramaEducativo". Por último, en relación a la tercera forma normal (3FN), la tabla "programaEducativo" también cumple, dado que cada columna depende directamente de la clave primaria "idProgramaEducativo" y no presenta dependencias transitivas con otras columnas. En conclusión, podemos afirmar que la tabla "programaEducativo" está normalizada en cuanto a la 1FN, 2FN y 3FN.

Tabla ee

ee	
idEE	int
nombre	varchar(150)
creditos	int

La tabla "ee" cumple con la primera forma normal (1FN) ya que no existen grupos repetidos de datos en sus columnas. Además, en cuanto a la segunda forma normal (2FN), la tabla también cumple con los requisitos, ya que todas las columnas dependen completamente de la clave primaria "idEE". Por último, en relación a la tercera forma normal (3FN), la tabla "ee" también cumple, dado que cada columna depende directamente de la clave primaria "idEE" y no presenta dependencias transitivas con otras columnas. En conclusión, podemos afirmar que la tabla "ee" está normalizada en cuanto a la 1FN, 2FN y 3FN.

Tabla aula

aula	
idAula	int
edificio_id	int

La tabla "aula" cumple con la primera forma normal (1FN) ya que no existen grupos repetidos de datos en sus columnas. Además, en cuanto a la segunda forma normal (2FN), la tabla también cumple con los requisitos, ya que todas las columnas dependen completamente de la clave primaria "idAula". Por último, en relación a la tercera forma normal (3FN), la tabla "aula" también cumple, dado que cada columna depende directamente de la clave primaria "idAula" y no presenta dependencias transitivas con otras columnas. En conclusión, podemos afirmar que la tabla "aula" está normalizada en cuanto a la 1FN, 2FN y 3FN.

Tabla edificio

edificio	
idEdificio	int
nombre	varchar(150)
facultad_id	int

La tabla "edificio" cumple con la primera forma normal (1FN) ya que no existen grupos repetidos de datos en sus columnas. Además, en cuanto a la segunda forma normal (2FN), la tabla también cumple con los requisitos, ya que todas las columnas dependen completamente de la clave primaria "idEdificio". Por último, en relación a la tercera forma normal (3FN), la tabla "edificio" también cumple, dado que cada columna depende directamente de la clave primaria "idEdificio" y no presenta dependencias transitivas con otras columnas. En conclusión, podemos afirmar que la tabla "edificio" está normalizada en cuanto a la 1FN, 2FN y 3FN.

Tabla alumno

alumno	
matricula	int
primerNombre	varchar(50)
segundoNombre	varchar(50)
apellidoPat	varchar(50)
apellidoMat	varchar(50)
email	varchar(150)
clave	varchar(64)
programaEduactivo_id	int

La tabla "alumno" cumple con la primera forma normal (1FN) ya que no existen grupos repetidos de datos en sus columnas. Además, en cuanto a la segunda forma normal (2FN), la tabla también cumple con los requisitos, ya que todas las columnas dependen completamente de la clave primaria "matricula". Por último, en relación a la tercera forma normal (3FN), la tabla "alumno" también cumple, dado que cada columna depende directamente de la clave primaria "matricula" y no presenta dependencias transitivas con otras columnas. En conclusión, podemos afirmar que la tabla "alumno" está normalizada en cuanto a la 1FN, 2FN y 3FN.

Tabla imparte

imparte	
idImparte	int
numNRC	int
aula_id	int
estado	bool
matricula_academico	int
dia	enum
horaInicio	timestamp
horaFin	timestamp

La tabla "imparte" cumple con la primera forma normal (1FN) ya que no existen grupos repetidos de datos en sus columnas. Además, en cuanto a la segunda forma normal (2FN), la tabla también cumple con los requisitos, ya que todas las columnas dependen completamente de la clave primaria "idImparte". Por último, en relación a la tercera forma normal (3FN), la tabla "imparte" también cumple, dado que cada columna depende directamente de la clave primaria "idImparte" y no presenta dependencias transitivas con otras columnas. En conclusión, podemos afirmar que la tabla "imparte" está normalizada en cuanto a la 1FN, 2FN y 3FN.

Tabla sección

seccion	
NRC	int
periodoInicio	date
periodoFin	date
ee_id	int
matricula_academico	int

La tabla "seccion" cumple con la primera forma normal (1FN) ya que no existen grupos repetidos de datos en sus columnas. Además, en cuanto a la segunda forma normal (2FN), la tabla también cumple con los requisitos, ya que todas las columnas dependen completamente de la clave primaria "NRC". Por último, en relación a la tercera forma normal (3FN), la tabla "seccion" también cumple, dado que cada columna depende directamente de la clave primaria "NRC" y no presenta dependencias transitivas con otras columnas. En conclusión, podemos afirmar que la tabla "seccion" está normalizada en cuanto a la 1FN, 2FN y 3FN.

Tabla academico

academico	
matricula	int
primerNombre	varchar(50)
segundoNombre	varchar(50)
apellidoPat	varchar(50)
apellidoMat	varchar(50)
correo	varchar(150)

La tabla "academico" cumple con la primera forma normal (1FN) ya que no existen grupos repetidos de datos en sus columnas. Además, en cuanto a la segunda forma normal (2FN), la tabla también cumple con los requisitos, ya que todas las columnas dependen completamente de la clave primaria "matricula". Por último, en relación a la tercera forma normal (3FN), la tabla "academico" también cumple, dado que cada columna depende directamente de la clave primaria "matricula" y no presenta dependencias transitivas con otras columnas. En conclusión, podemos afirmar que la tabla "academico" está normalizada en cuanto a la 1FN, 2FN y 3FN.

Tabla cursa

cursa	
idCursa	int
matricula_alumno	int
numNRC	int

La tabla "cursa" cumple con la primera forma normal (1FN) ya que no existen grupos repetidos de datos en sus columnas. Además, en cuanto a la segunda forma normal (2FN), la tabla también cumple con los requisitos, ya que todas las columnas dependen completamente de la clave primaria "idCursa". Por último, en relación a la tercera forma normal (3FN), la tabla "cursa" también cumple, dado que cada columna depende directamente de la clave primaria "idCursa" y no presenta dependencias transitivas con otras columnas. En conclusión, podemos afirmar que la tabla "cursa" está normalizada en cuanto a la 1FN, 2FN y 3FN.

Copias de seguridad para la base de datos

Crear copias de seguridad de una base de datos es esencial para proteger la información contenida en ella. Las copias de seguridad ayudan a garantizar la integridad y disponibilidad de los datos, proteger contra fallas del sistema, desastres naturales, errores humanos, facilitar la migración y cumplir con las regulaciones y normativas. En resumen, crear copias de seguridad es una práctica fundamental para cualquier empresa o persona que maneje datos importantes.

Montar una carpeta compartida de OMV-NAS

Para almacenar los respaldos de nuestro servicio de base de datos, es necesario crear una carpeta compartida en nuestro servidor de NAS. Para lograr esto, se pueden seguir los siguientes pasos:

1. Crea una carpeta en tu servidor de base de datos

```
$: sudo mkdir folderPrueba
```

2. Montar la carpeta compartida usando la carpeta creada

```
$: sudo mount -t cifs //direccion.servidor/carpetaNAS  
/folderPrueba -o username=usuario,password=contraseña
```

Captura de ejemplo

```
rserver@raspberrypi:~ $ cd /mnt/  
rserver@raspberrypi:/mnt $ sudo mkdir AdmonBD  
rserver@raspberrypi:/mnt $ ls  
AdmonBD  
rserver@raspberrypi:/mnt $ sudo mount -t cifs //192.168.1.100/AdmonBD /mnt/AdmonBD/ -o username=raian,password=F-----  
rserver@raspberrypi:/mnt $ cd AdmonBD/  
rserver@raspberrypi:/mnt/AdmonBD $ ls  
Claves  'Code & More'  Documentos  'Pagina web'  Scripts  SQL  
rserver@raspberrypi:/mnt/AdmonBD $
```

Desarrollo de scripts

Para automatizar tareas las tareas de respaldo y seguridad es necesaria la creación de scripts, estos scripts se encargaran de crear una copia de seguridad con una nomenclatura especifica, comprimir el archivo y enviar un aviso o notificación al DBA.

Para mejorar el siguiente texto y hacerlo más detallado y claro, se pueden añadir más información sobre los scripts que se van a crear para automatizar las tareas de respaldo y seguridad. Una posible forma de mejorar el texto podría ser la siguiente:

- Para automatizar las tareas de respaldo y seguridad de nuestra base de datos, es necesario crear scripts que nos permitan realizar estas tareas de manera programada y sin intervención manual. Estos scripts se encargarán de llevar a cabo las siguientes acciones:
- Crear una copia de seguridad de la base de datos con una nomenclatura específica que nos permita identificarla fácilmente en el futuro. Esta nomenclatura podría incluir la fecha y hora en la que se realizó el respaldo, el nombre de la base de datos, entre otros detalles que sean relevantes para nuestro entorno.
- Comprimir el archivo de la copia de seguridad para reducir su tamaño y facilitar su almacenamiento en nuestro sistema de respaldo.

- Enviar una notificación al administrador de la base de datos (DBA) para informarle sobre el éxito o fracaso de la tarea de respaldo. Esta notificación podría incluir detalles como la hora y fecha en la que se realizó la tarea, la ubicación donde se almacenó el archivo de respaldo, entre otros detalles relevantes.

Selección del lenguaje

Para el desarrollo del script, se ha decidido utilizar el lenguaje de programación Python. Esta elección se debe a varias razones importantes, que incluyen:

1. Python es un lenguaje de programación interpretado, lo que significa que el código fuente se ejecuta línea por línea a medida que se interpreta. Aunque esto puede llevar a una ejecución más lenta en comparación con los lenguajes compilados como C++, Python ofrece una sintaxis simple y legible que facilita el desarrollo rápido de aplicaciones
2. Los desarrolladores tienen experiencia previa en la programación en Python, lo que significa que están familiarizados con la sintaxis, las bibliotecas y los frameworks disponibles en el ecosistema de Python. Esto les permite aprovechar la amplia gama de herramientas y recursos que Python ofrece para el desarrollo de scripts.
3. Python es un lenguaje de programación versátil y de propósito general que se utiliza en una amplia variedad de aplicaciones, desde el desarrollo web y científico hasta la automatización de tareas y el aprendizaje automático. La gran cantidad de bibliotecas y módulos disponibles en Python facilita la implementación de funcionalidades avanzadas en el script.

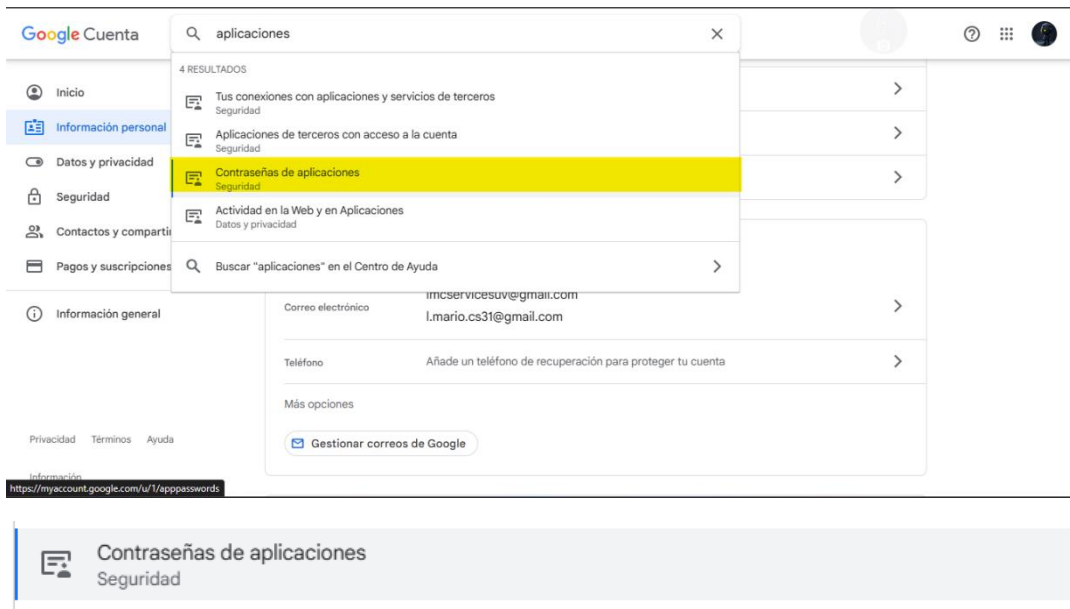
Al utilizar Python para el desarrollo de nuestro script, esperamos aprovechar su simplicidad y legibilidad, lo que puede resultar en un código más fácil de entender y mantener. Además, la amplia comunidad de desarrolladores de Python y la gran cantidad de recursos disponibles en línea facilitan el proceso de desarrollo y resolución de problemas.

Si bien Python puede ser más lento que C++ en términos de ejecución, para muchas aplicaciones, la diferencia de rendimiento es insignificante en comparación con los beneficios de productividad y facilidad de desarrollo que ofrece Python.

Enviar correos electrónicos desde el CLI del servidor

Para enviar correos electrónicos debemos instalar un servicio que nos ayude a cumplir la función, para este proyecto usaremos el servicio de sendmail y mailutils.

1. Primero vamos a crear una cuenta de google y configurarla con la verificación de dos pasos. (2FA)
2. Una vez creada y configurada nuestra cuenta de google, vamos a buscar la opción contraseñas de aplicaciones:



3. Seleccionas las siguientes opciones:

No tienes ninguna contraseña de aplicación.

Selecciona la aplicación y el dispositivo para los que quieres generar la contraseña de aplicación.

Correo

Seleccionar dispositivo

- iPhone
- iPad
- BlackBerry
- Mac
- Windows Phone
- Ordenador con Windows
- Otra (*nombre personalizado*)

GENERAR

4. Ahora vamos a ponerle un nombre a nuestra contraseña.

No tienes ninguna contraseña de aplicación.

Selecciona la aplicación y el dispositivo para los que quieres generar la contraseña de aplicación.

CorreoRserver X

GENERAR

5. Seguido a eso, vamos a generar nuestra clave y nos dara una contraseña y será la que utilizaremos para configurar los servicios de correo electronico

Ahora en nuestro rserver vamos a instalar los servicios.

1. Instalar sendmail mailutils.

```
$: sudo apt install sendmail mailutils
```

2. Ahora vamos a configurar sendmail, para ello ejecutamos el siguiente comando

```
$: sudo nano /etc/mail/sendmail.mc
```

3. Buscamos la linea que contiene “FEATURE(`use_cw_file')dn1” y agregamos las siguientes configuraciones:

```
$: define(`SMART_HOST',`smtp.gmail.com')dn1
define(`RELAY_MAILER_ARGS', `TCP $h 587')dn1
define(`ESMTP_MAILER_ARGS', `TCP $h 587')dn1
define(`confAUTH_OPTIONS', `A p')dn1
TRUST_AUTH_MECH(`EXTERNAL DIGEST-MD5 CRAM-MD5 LOGIN
PLAIN')dn1
define(`confAUTH_MECHANISMS', `EXTERNAL GSSAPI DIGEST
MD5 CRAM-MD5 LOGIN PLAIN')dn1
```

4. Finalmente guardamos el archivo y luego compilamos el archivo con la siguiente linea:

```
$: sudo m4 /etc/mail/sendmail.mc > /etc/mail/sendmail.cf
```

5. Ahora reiniciaremos el servicio para comprobar que no tengamos errores

```
$: sudo systemctl restart sendmail
```

6. Ahora vamos a configurar mailx


```
$: sudo nano /etc/mail.rc
```

7. Y ahora tenemos que agregar la siguiente configuración sustituyendo ciertas cosas con tus datos

```
set smtp-use-starttls
set ssl-verify=ignore
set smtp-auth=login
set smtp=smtps://smtp.gmail.com:587
set from=TuCorreo@gmail.com
set smtp-auth-user=TuCorreo@gmail.com
set smtp-auth-password=TuContraseña
```

8. Finalmente enviamos un correo escribiendo lo siguiente en la terminal

```
$: echo "Este es el cuerpo del correo" | mailx -s "Asunto del correo" destinatario@mail.com
```

Llamadas al sistema para respaldo

Ahora vamos a programar nuestro script en c++ para montar carpetas en nuestro servidor, crear copias de seguridad con Mysql Dump y que nos llegue un correo electrónico para notificarnos sobre la copia de seguridad.

1. getCuentas.py

El script realiza la descryptación de claves almacenadas en un archivo.

La función **desencryptar** toma un texto encriptado y un valor de desplazamiento llamado **shift**. Itera sobre cada carácter del texto y realiza lo siguiente:

- Si el carácter es una letra, determina si es mayúscula o minúscula y establece una base de referencia en el código ASCII correspondiente.

- Calcula el nuevo código ASCII del carácter descriptado aplicando la fórmula del cifrado de César.
- Convierte el código ASCII en el carácter descriptado y lo agrega a la cadena **decryptedText**.
- Si el carácter no es una letra, simplemente lo agrega a la cadena **decryptedText** sin modificarlo.
- Retorna la cadena **decryptedText** que contiene el texto descriptado.

La función **decrypt** realiza lo siguiente:

- Abre el archivo "credenciales.txt" en modo lectura.
- Lee el contenido del archivo y lo almacena en la variable **texto**.
- Divide el texto en líneas y las almacena en la lista **renglones**.
- Verifica si el texto tiene al menos dos caracteres.
- Si el texto tiene al menos dos caracteres, extrae el primer renglón (usuario) y el segundo renglón (clave) de la lista **renglones**. Luego, elimina los espacios en blanco alrededor de ambos renglones y los almacena en las variables **usuario** y **clave**, respectivamente.
- Crea una lista llamada **lista** que contiene el usuario y la clave encriptados.
- Descripta el usuario y la clave llamando a la función **descriptar** con un **shift** de 3 y los agrega a la lista **credenciales**.
- Retorna la lista **credenciales** que contiene el usuario y la clave descriptados.
- Si el texto no tiene al menos dos caracteres, imprime "El archivo no tiene credenciales" y finaliza la ejecución del programa usando **sys.exit()**.

2. hotbk.py

El script realiza el respaldo de una base de datos y archivos, generando una copia de seguridad.

El script realiza las siguientes acciones:

- Importa los módulos necesarios: **os**, **datetime**, **sys**, **getCuentas**, **telegram**, **tar** y **mail**.
- Define una función llamada **respaldo**.
- Establece la ruta de destino para guardar la copia de seguridad en la variable **ruta_guardado**.
- Obtiene la fecha y hora actual utilizando el módulo **datetime**.
- Crea un nombre de archivo para la copia de seguridad utilizando la fecha y hora actual.

- Llama a la función **decrypt** del módulo **getCuentas** para obtener las credenciales de la base de datos.
- Extrae el nombre de usuario y la contraseña de las credenciales obtenidas.
- Ejecuta el comando **mysqldump** para respaldar la base de datos utilizando las credenciales y guarda el resultado en un archivo con extensión **.sql**.
- Comprueba si el respaldo se ejecutó correctamente.
- Si el respaldo fue exitoso, llama a funciones del módulo **telegram** para notificar sobre el respaldo exitoso y realizar las siguientes acciones:
 - Comprime el archivo de respaldo en formato **.tar.gz** utilizando la función **comprimir** del módulo **tar**.
 - Si la compresión fue exitosa, llama a la función **successTar** del módulo **telegram** para notificar el éxito de la compresión.
 - Elimina los archivos de respaldo sin comprimir utilizando el comando **rm**.
 - Llama a la función **correo** del módulo **mail** para enviar la copia de seguridad por correo electrónico.
 - Si el envío del correo fue exitoso, llama a la función **sendmail** del módulo **telegram** para notificar el éxito del envío.
 - En caso contrario, llama a la función **fallomail** del módulo **telegram** para notificar el fallo en el envío del correo.
- Si el respaldo no fue exitoso, llama a la función **failCopia** del módulo **telegram** para notificar el fallo en el respaldo y finaliza la ejecución del programa utilizando **sys.exit()**.
- Finalmente, se llama a la función **respaldo** para ejecutar el respaldo.

3. tar.py

El script se encarga de comprimir un archivo o directorio utilizando el comando tar en Linux.

El script realiza las siguientes acciones:

- Importa el módulo **os**.
- Define una función llamada **comprimir** que toma el nombre del archivo a comprimir como argumento.
- Establece la ruta de destino para guardar el archivo comprimido en la variable **ruta_guardado**.
- Establece la ubicación original del archivo o directorio a comprimir en la variable **ruta_origen**.

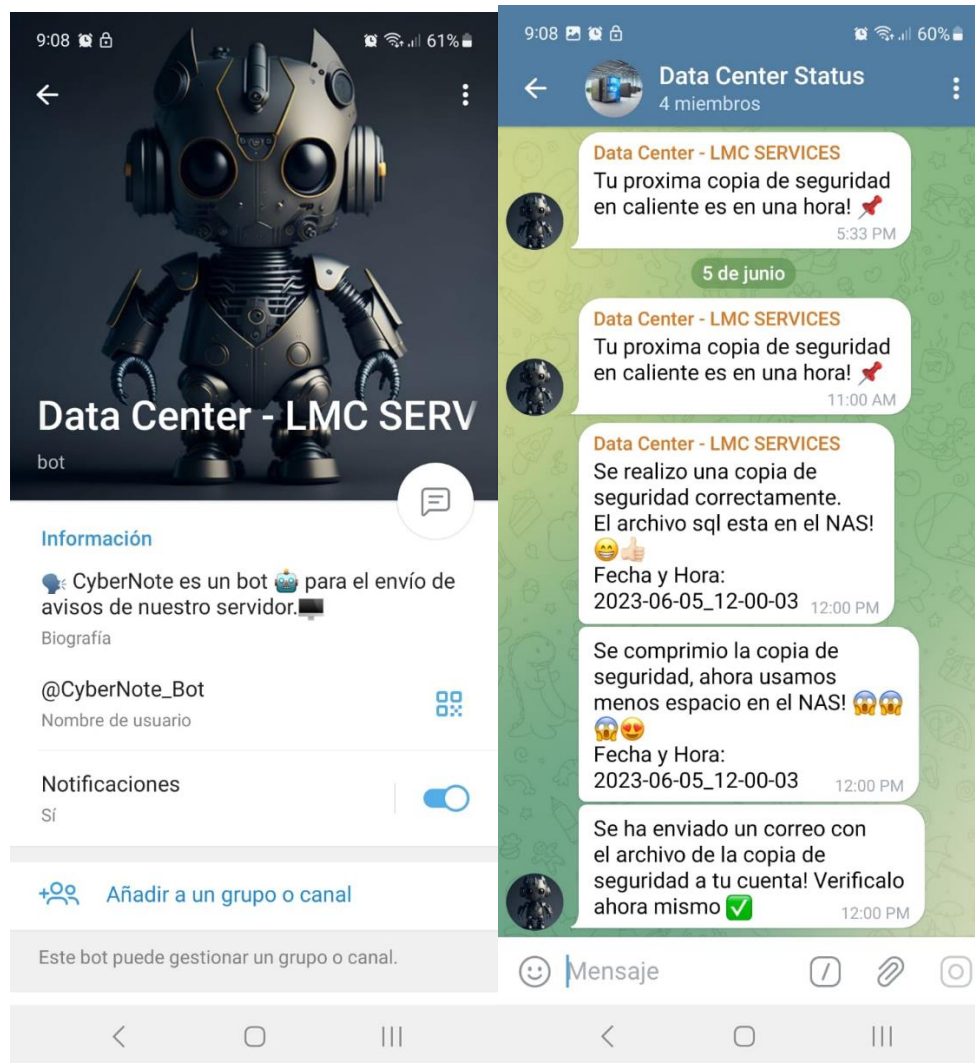
- Crea el nombre del archivo comprimido agregando la extensión **.tar.gz** al nombre del archivo proporcionado.
- Ejecuta el comando **tar** utilizando el comando **os.system()**, con la opción **-czvf** para crear un archivo comprimido en formato **.tar.gz**. El archivo o directorio a comprimir se especifica utilizando la variable **ruta_origen**, y el archivo comprimido resultante se guarda en la ubicación especificada en la variable **ruta_guardado**.
- Comprueba si la ejecución del comando **tar** fue exitosa. Si el comando retorna un código de salida igual a 0, se imprime "Archivo comprimido" y se retorna el valor 0.
- Si el comando **tar** retorna un código de salida distinto de 0, se imprime "Error al comprimir el archivo" y se retorna el valor 1.

4. telegram.py

El script utiliza la librería requests para enviar mensajes a través de la API de Telegram.

El script realiza las siguientes acciones:

- Importa el módulo **requests**.
- Define varias funciones que se utilizan para enviar mensajes a través de Telegram.
- Cada función tiene un propósito específico y recibe la fecha (en formato de cadena) como argumento.
- Dentro de cada función, se establecen las variables **chat_id** y **token** que corresponden al ID del chat y el token del bot de Telegram respectivamente.
- Se define el mensaje que se enviará a través de Telegram utilizando una cadena de formato que incluye la fecha proporcionada.
- Se construye la URL de la API de Telegram utilizando el token y se establece el método "sendmessage".
- Se definen los parámetros para la solicitud POST, que incluyen el chat_id y el texto del mensaje.
- Se realiza la solicitud POST a la URL de la API de Telegram utilizando la función **requests.post()**.
- En algunos casos, se imprime la respuesta JSON retornada por la API de Telegram (comentada en el código).



5. mail.py

El script utiliza el comando mailx en Linux para enviar correos electrónicos.

El script realiza las siguientes acciones:

- Importa el módulo **os**.
- Define una función llamada **correo** que recibe dos argumentos: **fecha** y **copia**.
- Dentro de la función, se definen las variables **asunto**, **cuerpo** y **destinatario** que representan el asunto, el cuerpo y el destinatario del correo electrónico respectivamente. Estos valores se construyen utilizando cadenas de formato que incluyen la fecha proporcionada.
- Se construye la instrucción para enviar el correo electrónico utilizando el comando **echo** y **mailx**. La instrucción incluye el cuerpo del correo electrónico, el asunto, el archivo adjunto y el destinatario.

- Se utiliza la función **os.system()** para ejecutar la instrucción en el sistema operativo.
- Se verifica el valor de retorno de la función **os.system()** para determinar si el comando se ejecutó correctamente o no.
- Si el valor de retorno es 0, se devuelve 0 para indicar que el correo electrónico se envió correctamente.
- Si el valor de retorno no es 0, se devuelve 1 para indicar que ocurrió un error al enviar el correo electrónico.

6. aviso.py

El script utiliza un módulo llamado telegram para realizar una acción específica.

El script realiza las siguientes acciones:

- Importa el módulo **telegram**.
- Llama a la función **wait()** del módulo **telegram**.

Auditoria y Monitoreo

La auditoría y monitoreo del servicio de base de datos son dos elementos fundamentales para garantizar el funcionamiento eficiente y seguro de cualquier sistema de gestión de datos. Estas prácticas son esenciales tanto para las organizaciones que manejan grandes volúmenes de información como para aquellas que operan con bases de datos más pequeñas pero críticas.

La auditoría de la base de datos consiste en examinar y evaluar los registros de actividades y eventos relacionados con el uso y acceso a la base de datos. Su objetivo principal es asegurar que se cumplan los estándares de seguridad, integridad y disponibilidad de los datos, así como garantizar el cumplimiento de las normativas y regulaciones aplicables.

El monitoreo, por su parte, se enfoca en supervisar el rendimiento y el estado de la base de datos en tiempo real. Esto implica analizar diferentes métricas y parámetros clave, como el tiempo de respuesta, el uso de recursos, el crecimiento del tamaño de la base de datos y la detección de posibles problemas o fallas. El monitoreo proactivo permite identificar y solucionar rápidamente los problemas antes de que afecten negativamente a los usuarios y a la funcionalidad del sistema.

La importancia de la auditoría y el monitoreo de la base de datos radica en varios aspectos:

- Seguridad: La base de datos suele contener información confidencial y sensible, como datos personales, financieros o comerciales. La auditoría ayuda a detectar y prevenir accesos no

autorizados, cambios no autorizados en los datos y actividades sospechosas que puedan comprometer la seguridad de la información. El monitoreo continuo permite detectar ataques en tiempo real y tomar medidas correctivas para proteger los datos.

- **Cumplimiento normativo:** Muchas organizaciones deben cumplir con regulaciones específicas que rigen la protección de datos, como la Ley de Protección de Datos Personales. La auditoría de la base de datos ayuda a demostrar que se han implementado las medidas adecuadas para cumplir con estas regulaciones y proporciona registros y trazabilidad en caso de auditorías o investigaciones.
- **Rendimiento óptimo:** El monitoreo constante del rendimiento de la base de datos permite identificar cuellos de botella, problemas de configuración o ineficiencias en las consultas y transacciones. Esto ayuda a optimizar la estructura y el diseño de la base de datos, así como a mejorar la velocidad y la eficiencia en el acceso a los datos.
- **Disponibilidad y continuidad del servicio:** La base de datos es un componente crítico para el funcionamiento de muchas aplicaciones y sistemas empresariales. El monitoreo ayuda a detectar fallos o degradaciones del sistema, permitiendo una respuesta rápida para minimizar el tiempo de inactividad y garantizar la continuidad del servicio.

Visualizar log de los usuarios el servicio de base de datos

```
Mariadb [nombre_bd] > show variables like '%general_log%';
```

y obtendremos una salida como la siguiente:

```
MariaDB [uv]> show variables like '%general_log%';
+-----+-----+
| Variable_name | Value |
+-----+-----+
| general_log   | OFF   |
| general_log_file | raspberrypi.log |
+-----+-----+
2 rows in set (0.006 sec)
```

y ahora activaremos el log con el siguiente comando:

```
MariaDB [uv]> set global general_log=1;
```

y ahora verificamos el estado con la siguiente instrucción:

```
Mariadb [nombre_bd] > show variables like '%general_log%';
```

y veremos que ya esta habilitado el general_log

```
MariaDB [uv]> show variables like '%general_log%';
+-----+-----+
| Variable_name | Value |
+-----+-----+
| general_log   | ON    |
| general_log_file | raspberry.log |
+-----+-----+
2 rows in set (0.006 sec)
```

y ahora podremos ver todos los registros de un usuario desde la ruta :

```
$ sudo cat /var/lib/mysql/nombre_usuario.log
```

```
rserver@raspberrypi:~$ sudo cat /var/lib/mysql/raspberrypi.log
/usr/sbin/mariadb, Version: 10.5.19-MariaDB-0+deb11u2 (Debian 11). started with:
Tcp port: 3306 Unix socket: /run/mysqld/mysqld.sock
Time          Id Command Argument
230521 0:44:34 12 Query show variables like '%general_log%'
230521 0:46:19 12 Quit
```

Uso de mytop

Mytop es una herramienta de monitoreo de bases de datos MySQL que proporciona información en tiempo real sobre el rendimiento y la actividad de una instancia de MySQL. Es una herramienta de línea de comandos que muestra de manera interactiva diversas métricas y estadísticas relevantes del servidor de base de datos.

Mytop recopila datos en tiempo real sobre las consultas ejecutadas, el tiempo de respuesta, el uso de CPU y memoria, así como las conexiones activas. Proporciona una interfaz fácil de usar que muestra esta información de manera tabular y actualizada constantemente.

La función principal de Mytop es permitir a los administradores de bases de datos supervisar y analizar el rendimiento de MySQL en tiempo real. Al observar las consultas y su tiempo de ejecución, los administradores pueden identificar cuellos de botella y optimizar el rendimiento del servidor. Además,

Mytop muestra información sobre las consultas más costosas en términos de tiempo de respuesta, lo que facilita la identificación y solución de problemas de rendimiento.

Además del monitoreo del rendimiento, Mytop también permite verificar el estado de las conexiones activas, el uso de memoria y CPU, así como el tráfico de red. Esto ayuda a identificar cualquier sobrecarga o anomalía en el sistema, lo que es especialmente útil en entornos donde se requiere una alta disponibilidad y rendimiento constante de la base de datos.

Otra característica útil de Mytop es su capacidad para realizar consultas en tiempo real dentro de la interfaz. Esto permite a los administradores ejecutar consultas SQL directamente desde Mytop para obtener información adicional o realizar análisis más detallados sobre los datos almacenados.

Para comenzar a usar mytop, debemos poner la configuración inicial con:

```
$ sudo nano ~/.mytop
```

dentro de este archivo pondremos:

```
user=root
```

```
pass=tuContraseña
```

```
db=uv
```

```
host=127.0.0.1
```

```
delay=4
```

```
port=3306
```

Finalmente, basta con escribir en la terminal :

```
$ mytop --promp
```

Ingresamo nuestra contraseña y obtendremos un monitoreo en tiempo real de que usuarios se han conectado.

```
MariaDB 10.5.19 on 127.0.0.1 load (0.00 0.00 0.00) up 2+08:42:20 [00:59:38]
Queries: 401.0 qps: 0 Slow: 0.0 Se/In/Up/De(%): 27/00/00/00

MyISAM Key Cache Efficiency: 100.0% Bps in/out: 0.1/ 1.8

  Id      User      Host/IP      DB   Time   %   Cmd  State Query
  --      -
  --      -
```

Tipos de respaldo

En el marco de nuestro proyecto de administración en base de datos, se ha decidido utilizar una combinación de tres tipos de respaldos: respaldo completo, respaldo en caliente y respaldo diferencial. Esta elección se basa en las siguientes razones:

1. Respallos completos: Hemos optado por realizar respaldos completos de la base de datos debido a su capacidad de capturar la totalidad de los datos y estructuras en un solo archivo o conjunto de archivos. El respaldo completo garantiza una restauración integral de la base de

datos en caso de pérdida total de datos o corrupción grave. Esta estrategia nos brinda una imagen precisa y completa de la base de datos en un momento específico, asegurando que no se pierda información importante.

2. **Respaldos en caliente:** Hemos seleccionado el respaldo en caliente debido a su capacidad de respaldar la base de datos mientras sigue en funcionamiento. Esta estrategia nos permite capturar los cambios y actualizaciones en tiempo real, minimizando el tiempo de inactividad y evitando interrupciones en la operatividad del sistema. El respaldo en caliente garantiza que nuestros respaldos reflejen la información más reciente, lo que nos brinda mayor protección y disponibilidad de los datos.
3. **Respaldos diferenciales:** Hemos incluido el respaldo diferencial en nuestra estrategia debido a su eficiencia en términos de tiempo y espacio de almacenamiento. Con el respaldo diferencial, se respaldan únicamente los datos que han cambiado desde el último respaldo completo. Esto reduce el tiempo necesario para realizar respaldos y también disminuye el espacio requerido para almacenar los respaldos diferenciales. En caso de necesitar una restauración, solo se requiere el respaldo completo más el último respaldo diferencial, lo que agiliza el proceso de recuperación.

Al utilizar esta combinación de respaldos completo, en caliente y diferencial, buscamos obtener los beneficios individuales de cada uno de ellos y maximizar la seguridad y disponibilidad de nuestros datos:

- El respaldo completo nos asegura la capacidad de restaurar la base de datos en su totalidad en caso de eventos catastróficos.
- El respaldo en caliente nos proporciona respaldos actualizados en tiempo real, minimizando la pérdida de datos y reduciendo el tiempo de inactividad.
- El respaldo diferencial nos ofrece una solución eficiente en términos de espacio y tiempo, al respaldar solo los cambios desde el último respaldo completo.

Interfaz gráfica del sistema

Selección del lenguaje

Nuestro equipo ha decidido utilizar el lenguaje Java en la creación de la interfaz gráfica de nuestro proyecto por las siguientes razones:

- **Portabilidad:** La capacidad de Java para ejecutarse en múltiples plataformas, sin requerir cambios significativos en el código, es una ventaja clave para nosotros. Al utilizar Java, podemos desarrollar una interfaz gráfica que funcione de manera consistente en diferentes sistemas operativos, lo que amplía la audiencia potencial de nuestra aplicación y garantiza una experiencia uniforme para los usuarios, independientemente del sistema en el que estén trabajando.
- **Amplia biblioteca de GUI:** Java ofrece una amplia biblioteca de componentes y herramientas para el desarrollo de interfaces gráficas, como Java Swing y JavaFX. Estas bibliotecas

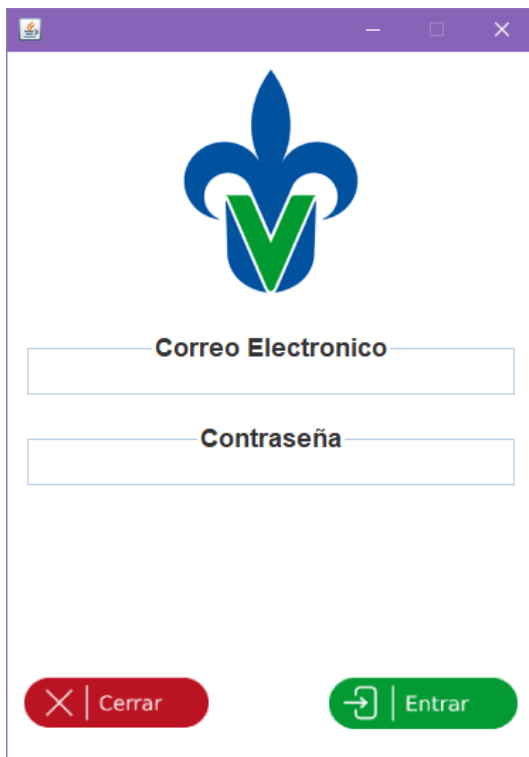
proporcionan una gran variedad de componentes predefinidos y personalizables, lo que nos permite crear una interfaz atractiva y funcional de manera más eficiente. Además, estas bibliotecas cuentan con una gran cantidad de recursos y documentación, lo que facilita el aprendizaje y la implementación de características específicas de la interfaz.

- Rendimiento y estabilidad: Java es conocido por su rendimiento y estabilidad. El lenguaje y su máquina virtual están diseñados para optimizar la ejecución de aplicaciones, lo que se traduce en interfaces gráficas fluidas y responsivas. Además, el manejo automático de la memoria y la gestión de excepciones de Java contribuyen a mejorar la estabilidad de la aplicación en general, reduciendo la posibilidad de bloqueos o errores graves que puedan afectar la experiencia del usuario.

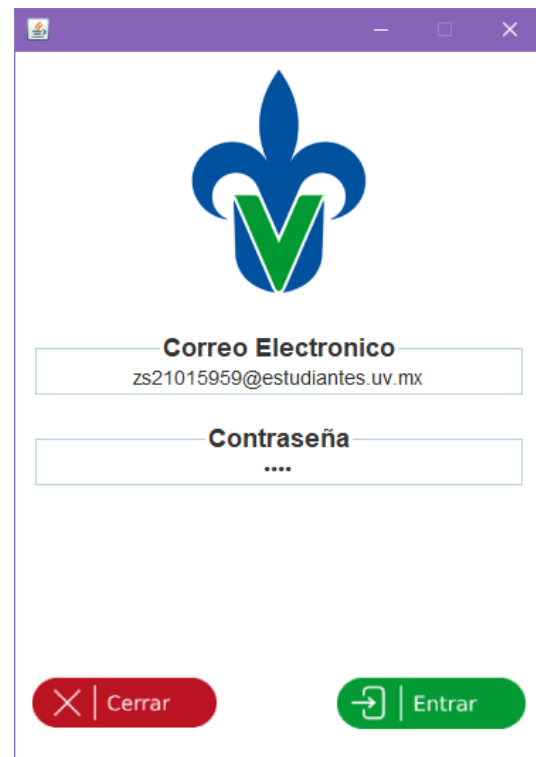
Principio del formulario

Capturas del trabajo

Pantalla de inicio de sesión:




The screenshot shows a web application window with a purple title bar. At the top center is a logo consisting of a blue fleur-de-lis with a green 'V' inside. Below the logo are two input fields. The first is labeled 'Correo Electronico' and is empty. The second is labeled 'Contraseña' and is also empty. At the bottom, there are two buttons: a red one with a white 'X' icon and the text 'Cerrar', and a green one with a white login icon and the text 'Entrar'.




This screenshot is identical to the previous one, but the 'Correo Electronico' field now contains the text 'zs21015959@estudiantes.uv.mx'. The 'Contraseña' field is filled with ten dots '....' to represent masked text. The buttons and logo remain the same.


Pantalla del horario en general:



Jessica
Peña Montero




Lunes



Buscar

MATRICULA: 21015959


Experiencia Educativa	Edificio	Aula	Día	Hora de Inicio	Hora de Fin
Administracion de Base de Datos	FEI	LABRED	Miercoles	11:00:00	14:00:00
Metodologia de la investigacion	FEI	Aula TC	Lunes	11:00:00	13:00:00
Redes	FEI	Aula 404	Lunes	13:00:00	15:00:00
Redes	FEI	Aula 404	Martes	13:00:00	15:00:00
Redes	FEI	LABRED	Miercoles	13:00:00	15:00:00
Estructura de Datos	FEI	F101	Miercoles	11:00:00	13:00:00




Pantalla de horario, pero filtrado por día:



Jessica
Peña Montero




Martes


Buscar

MATRICULA: 21015959

Experiencia Educativa	Edificio	Aula	Día	Hora de Inicio	Hora de Fin
Redes	FEI	Aula 404	Martes	13:00:00	15:00:00



Instalación y Configuración de ngrok

Ngrok es una herramienta que permite crear un túnel seguro entre una máquina local y la internet pública. Permite exponer un servidor local detrás de una red privada o un firewall a través de una dirección URL pública temporal. Con Ngrok, puedes acceder a tu servidor local desde cualquier lugar a través de internet sin necesidad de abrir puertos o configurar reenvíos en tu enrutador.

La forma en que funciona Ngrok es estableciendo una conexión segura entre tu máquina local y los servidores de Ngrok a través de una conexión SSL/TLS. Una vez que la conexión se establece, Ngrok genera una URL pública única que se puede utilizar para acceder a tu servidor local. Esta URL se actualiza cada vez que inicias Ngrok, por lo que cada vez que reinicies la herramienta obtendrás una URL nueva.

En el contexto de la administración de bases de datos, Ngrok puede ser útil en la siguiente situación:

Para facilitar la conexión entre tu programa Java y una base de datos local sin que el cliente necesite utilizar la red VPN utilizada por los administradores, puedes utilizar el servicio de túnel de Ngrok.

El servicio de túnel de Ngrok se configurará en el puerto 3306 del protocolo TCP para conectarlo con nuestro programa Java. Al establecer el túnel de Ngrok en el puerto 3306, podremos redirigir de manera segura las conexiones hacia nuestro programa Java. Esto evitará que el cliente tenga que contar con acceso directo a la red VPN.

Esta configuración resulta beneficiosa al simplificar la configuración y evitar la necesidad de que los clientes tengan acceso a la red VPN. Es especialmente útil en casos en los que los clientes no tienen los permisos necesarios para acceder a la red VPN o cuando se busca evitar complicaciones de configuración de red.

Conexión del sistema a la base de datos

Servicio de ngrok:

```
ngrok (Ctrl+C to quit)
Announcing ngrok-rust: The ngrok agent as a Rust crate: https://ngrok.com/rust

Session Status      online
Account             JessiPeM (Plan: Free)
Update              update available (version 3.3.0, Ctrl-U to update)
Version             3.2.2
Region              United States (us)
Latency             53ms
Web Interface       http://127.0.0.1:4040
Forwarding           tcp://2.tcp.ngrok.io:17056 -> localhost:3306

Connections
  ttl    opn    rt1    rt5    p50    p90
    22     0   0.00   0.01   0.99   22.43
```

Resultado obtenido

Conexión de java al servicio de base de datos con el túnel de ngrok

```
//variables de acceso
String db = "uv";
String url = "jdbc:mysql://2.tcp.ngrok.io:17056/";
String driver = "com.mysql.cj.jdbc.Driver";
Connection cx;

//funcion para conectarse a la base de datos
public Connection conectar(){

    String [] cuenta = decrypt();
    try {
        Class.forName(className: driver);
        cx = DriverManager.getConnection(url+db,cuenta[0],cuenta[1]);
        System.out.println(x: "Se conecto a la db");
    } catch (ClassNotFoundException | SQLException e) {
        System.out.println(x: e);
    }
    return cx;
}
```

```
Archivo creado exitosamente.
05550bc685f04599b689922632e7baf0719d003649362002c83ace6253b03d49
Se conecto a la db
```

Conclusión

En conclusión, el proyecto de administración de bases de datos sobre la Universidad Veracruzana ha sido un esfuerzo significativo para mejorar la gestión y organización de los datos relacionados con esta institución educativa. A lo largo del proyecto, hemos logrado los siguientes resultados destacados:

- Diseño y desarrollo de la estructura de la base de datos: Se ha diseñado una base de datos robusta y escalable que ha permitido almacenar y organizar de manera eficiente la información relevante de la Universidad Veracruzana. Esto ha facilitado la gestión y el acceso a los datos, mejorando la eficiencia y precisión de los procesos administrativos.
- Implementación de funcionalidades clave: Se han implementado funcionalidades esenciales en el sistema de gestión de la base de datos, como la inserción, modificación y eliminación de

registros, así como consultas complejas para extraer información específica. Esto ha permitido a los usuarios obtener datos relevantes de manera rápida y precisa.

- Mejora de la integridad y seguridad de los datos: Se han establecido mecanismos de validación y restricciones en la base de datos para garantizar la integridad de los datos almacenados. Además, se han implementado medidas de seguridad para proteger la confidencialidad de la información sensible.

Durante el desarrollo del proyecto, también nos enfrentamos a desafíos significativos, entre ellos:

- Recopilación de datos: La obtención de datos precisos y actualizados de los diferentes departamentos y unidades de la Universidad Veracruzana ha sido un desafío importante. La coordinación y colaboración con las partes interesadas fue fundamental para asegurar la calidad de los datos utilizados en el proyecto.
- Complejidad de la estructura organizativa: La Universidad Veracruzana es una institución grande y compleja, con múltiples facultades, departamentos y programas académicos. La representación de esta estructura en la base de datos requirió un cuidadoso análisis y diseño para garantizar su adecuada representación y relación.

En general, el proyecto de administración de bases de datos sobre la Universidad Veracruzana ha sido un éxito en términos de mejorar la gestión de los datos y brindar una plataforma sólida para futuras aplicaciones y análisis. Se espera que la implementación de esta base de datos contribuya a la eficiencia y efectividad de los procesos administrativos en la universidad, así como a la toma de decisiones informadas basadas en datos confiables.