

Implementación y Configuración del Servidor de Bases de Datos para Codearts Solutions

30/06/2025

Ángel Moreno García

CodeArts Solutions

Madrid

Visión general

Este documento detalla la instalación, configuración, seguridad, conexión y respaldo del servidor de bases de datos implementado para **Codearts Solutions**. Se abordan las fases necesarias para garantizar un sistema eficiente, seguro y escalable que soporte las necesidades de gestión de datos de la empresa.

Introducción

1. Contexto y objetivos

Codearts Solutions requiere la implementación de un servidor de bases de datos robusto y seguro para optimizar la gestión de la información corporativa. Este proyecto busca instalar y configurar un sistema que permita gestionar usuarios con permisos diferenciados, asegurar el acceso y mantener respaldos automáticos, garantizando así la continuidad y seguridad del negocio.

2. Selección del Motor de Base de Datos

Se evaluaron dos motores principales: **MySQL/MariaDB** y PostgreSQL. La elección final fue **MariaDB**, por su compatibilidad, rendimiento y facilidad de configuración en el entorno Linux utilizado.

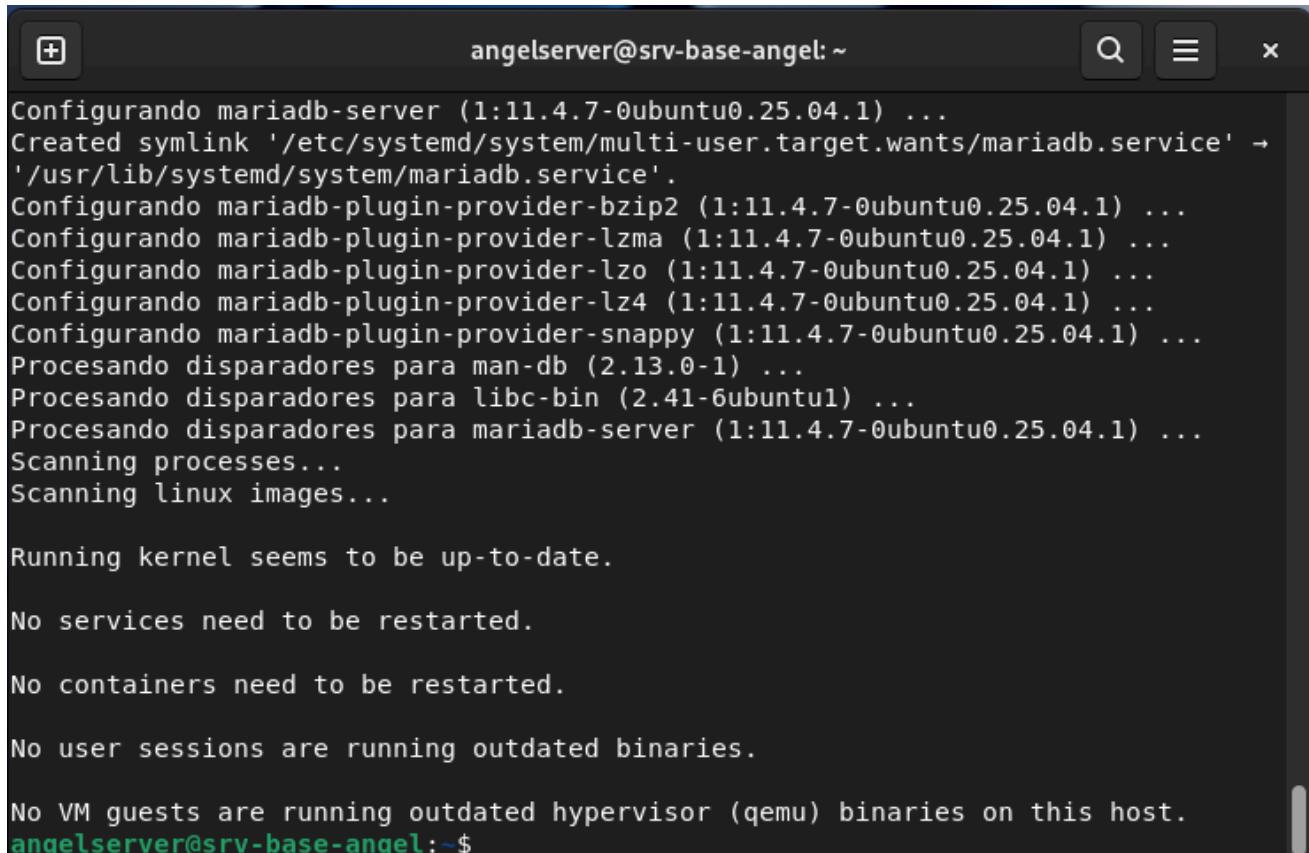
Fase 1: Instalación y Configuración del Servidor

1. Preparación del Entorno

Actualización de repositorios e instalación de MariaDB: Para ello usaremos los comandos en la terminal

```
sudo apt update
```

```
sudo apt install mariadb-server
```



The screenshot shows a terminal window with a dark theme. The title bar reads "angelserver@srv-base-angel: ~". The window contains the following text output:

```
Configurando mariadb-server (1:11.4.7-0ubuntu0.25.04.1) ...
Created symlink '/etc/systemd/system/multi-user.target.wants/mariadb.service' →
'/usr/lib/systemd/system/mariadb.service'.
Configurando mariadb-plugin-provider-bzip2 (1:11.4.7-0ubuntu0.25.04.1) ...
Configurando mariadb-plugin-provider-lzma (1:11.4.7-0ubuntu0.25.04.1) ...
Configurando mariadb-plugin-provider-lzo (1:11.4.7-0ubuntu0.25.04.1) ...
Configurando mariadb-plugin-provider-lz4 (1:11.4.7-0ubuntu0.25.04.1) ...
Configurando mariadb-plugin-provider-snappy (1:11.4.7-0ubuntu0.25.04.1) ...
Procesando disparadores para man-db (2.13.0-1) ...
Procesando disparadores para libc-bin (2.41-6ubuntu1) ...
Procesando disparadores para mariadb-server (1:11.4.7-0ubuntu0.25.04.1) ...
Scanning processes...
Scanning linux images...

Running kernel seems to be up-to-date.

No services need to be restarted.

No containers need to be restarted.

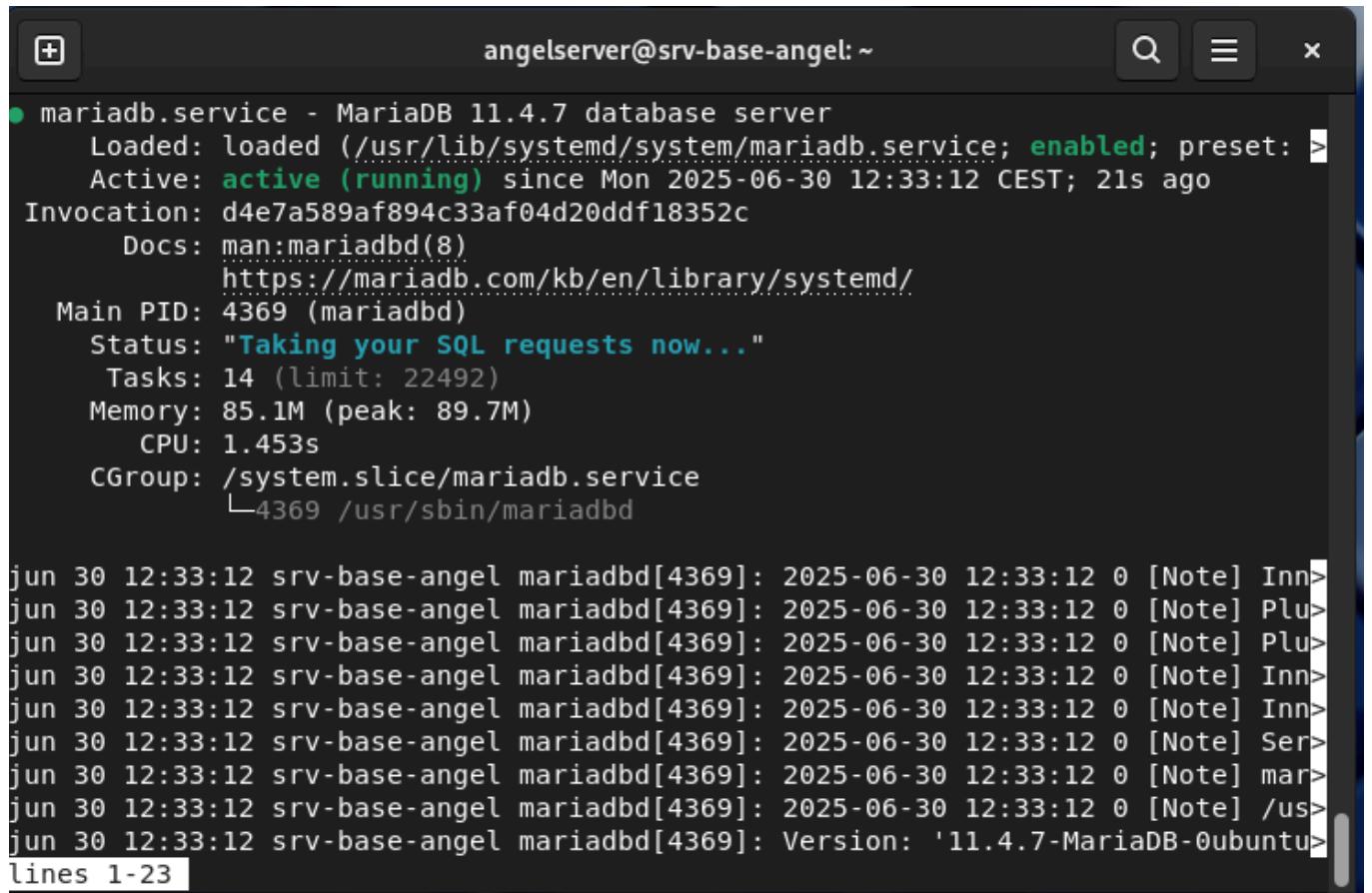
No user sessions are running outdated binaries.

No VM guests are running outdated hypervisor (qemu) binaries on this host.
angelserver@srv-base-angel:~$
```

2. Verificación del Servicio

Para comprobar que el servicio MariaDB está activo y funcionando usaremos:

sudo systemctl status mariadb



```
mariadb.service - MariaDB 11.4.7 database server
   Loaded: loaded (/usr/lib/systemd/system/mariadb.service; enabled; preset: >
   Active: active (running) since Mon 2025-06-30 12:33:12 CEST; 21s ago
     Invocation: d4e7a589af894c33af04d20ddf18352c
       Docs: man:mariadb(8)
              https://mariadb.com/kb/en/library/systemd/
    Main PID: 4369 (mariadb)
      Status: "Taking your SQL requests now..."
        Tasks: 14 (limit: 22492)
      Memory: 85.1M (peak: 89.7M)
        CPU: 1.453s
      CGroup: /system.slice/mariadb.service
              └─4369 /usr/sbin/mariadb

jun 30 12:33:12 srv-base-angel mariadb[4369]: 2025-06-30 12:33:12 0 [Note] Inn>
jun 30 12:33:12 srv-base-angel mariadb[4369]: 2025-06-30 12:33:12 0 [Note] Plu>
jun 30 12:33:12 srv-base-angel mariadb[4369]: 2025-06-30 12:33:12 0 [Note] Plu>
jun 30 12:33:12 srv-base-angel mariadb[4369]: 2025-06-30 12:33:12 0 [Note] Inn>
jun 30 12:33:12 srv-base-angel mariadb[4369]: 2025-06-30 12:33:12 0 [Note] Inn>
jun 30 12:33:12 srv-base-angel mariadb[4369]: 2025-06-30 12:33:12 0 [Note] Ser>
jun 30 12:33:12 srv-base-angel mariadb[4369]: 2025-06-30 12:33:12 0 [Note] mar>
jun 30 12:33:12 srv-base-angel mariadb[4369]: 2025-06-30 12:33:12 0 [Note] /us>
jun 30 12:33:12 srv-base-angel mariadb[4369]: Version: '11.4.7-MariaDB-0ubuntu>
lines 1-23
```

El servicio debe aparecer como active (running). Se incluye captura de pantalla con el resultado.

3. Verificación del Servicio

Para garantizar que *MariaDB* arranque automáticamente al encender el servidor:

sudo systemctl enable mariadb

```
angelserver@srv-base-angel:~$ sudo systemctl enable mariadb
Synchronizing state of mariadb.service with SysV service script with /usr/lib/sys
stemd/systemd-sysv-install.
Executing: /usr/lib/systemd/systemd-sysv-install enable mariadb
angelserver@srv-base-angel:~$ █
```

Fase 2: Creación y Gestión de Bases de Datos y Usuarios

1. Acceso al Shell de MariaDB

Se accede al sistema de gestión de bases de datos con el usuario root de *MariaDB*, usaremos el comando en bash: ***sudo mariadb***

2. Creación de la Base de Datos

Se crea la base de datos para **Codearts Solutions** en SQL:

CREATE DATABASE codearts_db;

Explicación: La base de datos codearts_db contendrá toda la información relacionada con los sistemas internos.

3. Creación de Usuarios y Asignación de Permisos

Se crean tres usuarios con diferentes niveles de acceso para asegurar la segregación de funciones y la seguridad.

- 
- Usuario administrador con permisos totales:

```
CREATE USER 'admin_db'@'localhost' IDENTIFIED BY 'ContraseñaSegura1!';
```

```
GRANT ALL PRIVILEGES ON codearts_db.* TO 'admin_db'@'localhost';
```

- Usuario desarrollador con permisos de lectura y escritura:

```
CREATE USER 'dev_user'@'localhost' IDENTIFIED BY 'ContraseñaSegura2!';
```

```
GRANT SELECT, INSERT, UPDATE ON codearts_db.* TO 'dev_user'@'localhost';
```

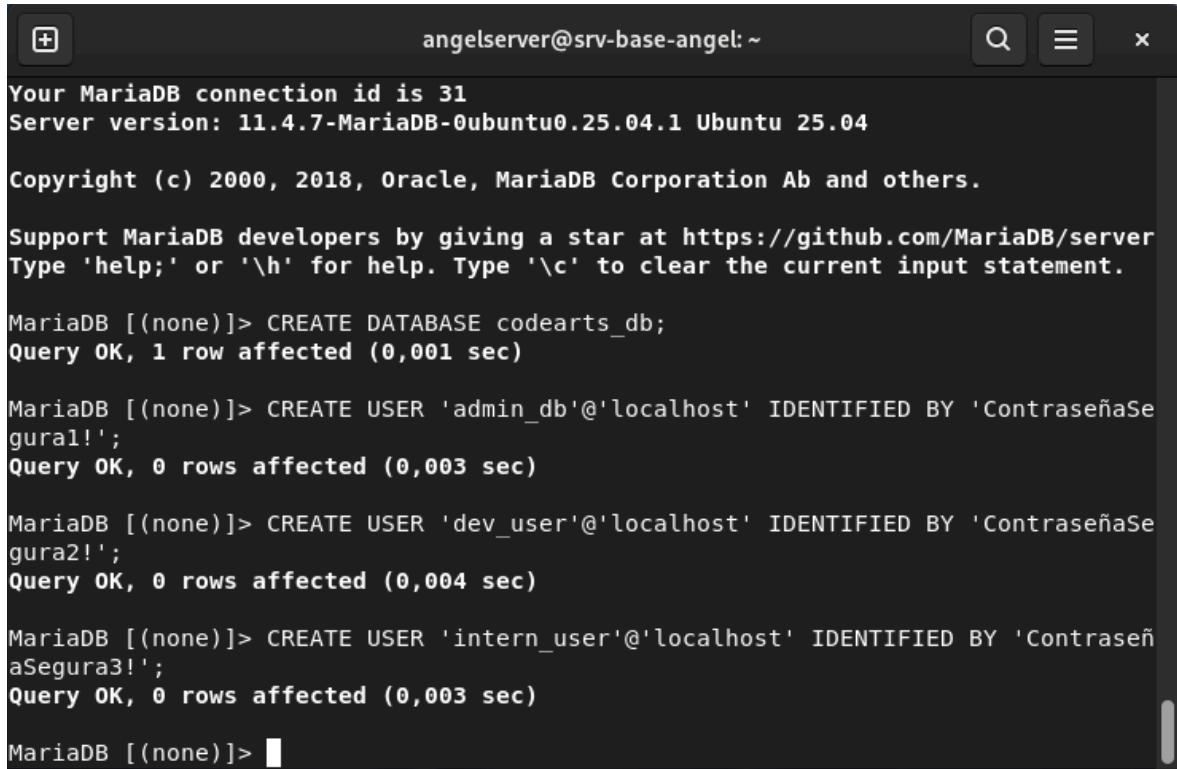
- Usuario interno con permisos solo de lectura:

```
CREATE USER 'intern_user'@'localhost' IDENTIFIED BY 'ContraseñaSegura3!';
```

```
GRANT SELECT ON codearts_db.* TO 'intern_user'@'localhost';
```

- Finalmente, recargar privilegios para aplicar cambios:

```
FLUSH PRIVILEGES;
```



```

+ angelserver@srv-base-angel: ~
Your MariaDB connection id is 31
Server version: 11.4.7-MariaDB-0ubuntu0.25.04.1 Ubuntu 25.04

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Support MariaDB developers by giving a star at https://github.com/MariaDB/server
Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MariaDB [(none)]> CREATE DATABASE codearts_db;
Query OK, 1 row affected (0,001 sec)

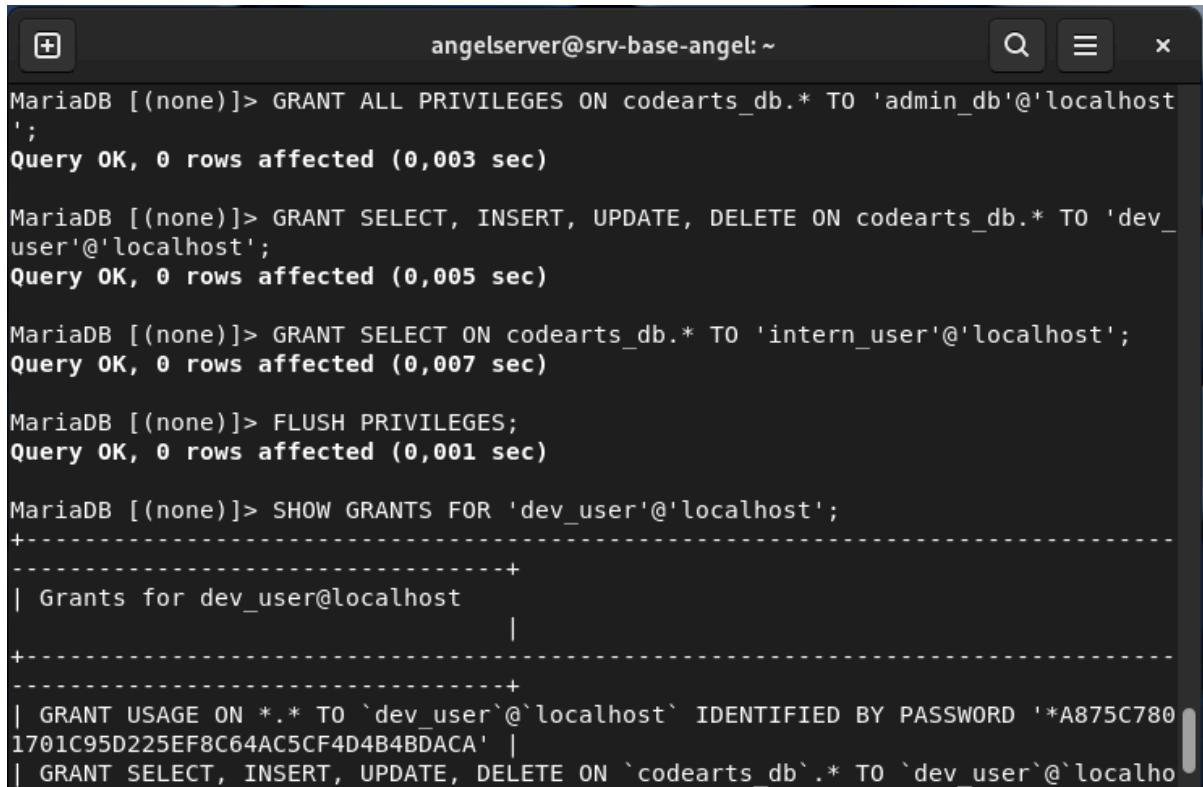
MariaDB [(none)]> CREATE USER 'admin_db'@'localhost' IDENTIFIED BY 'ContraseñaSegura!';
Query OK, 0 rows affected (0,003 sec)

MariaDB [(none)]> CREATE USER 'dev_user'@'localhost' IDENTIFIED BY 'ContraseñaSegura2!';
Query OK, 0 rows affected (0,004 sec)

MariaDB [(none)]> CREATE USER 'intern_user'@'localhost' IDENTIFIED BY 'ContraseñaSegura3!';
Query OK, 0 rows affected (0,003 sec)

MariaDB [(none)]>

```



```

+ angelserver@srv-base-angel: ~
MariaDB [(none)]> GRANT ALL PRIVILEGES ON codearts_db.* TO 'admin_db'@'localhost';
Query OK, 0 rows affected (0,003 sec)

MariaDB [(none)]> GRANT SELECT, INSERT, UPDATE, DELETE ON codearts_db.* TO 'dev_user'@'localhost';
Query OK, 0 rows affected (0,005 sec)

MariaDB [(none)]> GRANT SELECT ON codearts_db.* TO 'intern_user'@'localhost';
Query OK, 0 rows affected (0,007 sec)

MariaDB [(none)]> FLUSH PRIVILEGES;
Query OK, 0 rows affected (0,001 sec)

MariaDB [(none)]> SHOW GRANTS FOR 'dev_user'@'localhost';
+-----+
| Grants for dev_user@localhost
| |
+-----+
| GRANT USAGE ON *.* TO `dev_user`@`localhost` IDENTIFIED BY PASSWORD '*A875C7801701C95D225EF8C64AC5CF4D4B4BDACA'
| |
| GRANT SELECT, INSERT, UPDATE, DELETE ON `codearts_db`.* TO `dev_user`@`localhost` IDENTIFIED BY PASSWORD '*A875C7801701C95D225EF8C64AC5CF4D4B4BDACA'
+-----+

```

Fase 3: Conexión de la Aplicación Web a la Base de Datos

1. Instalación del Servidor Web y Lenguaje

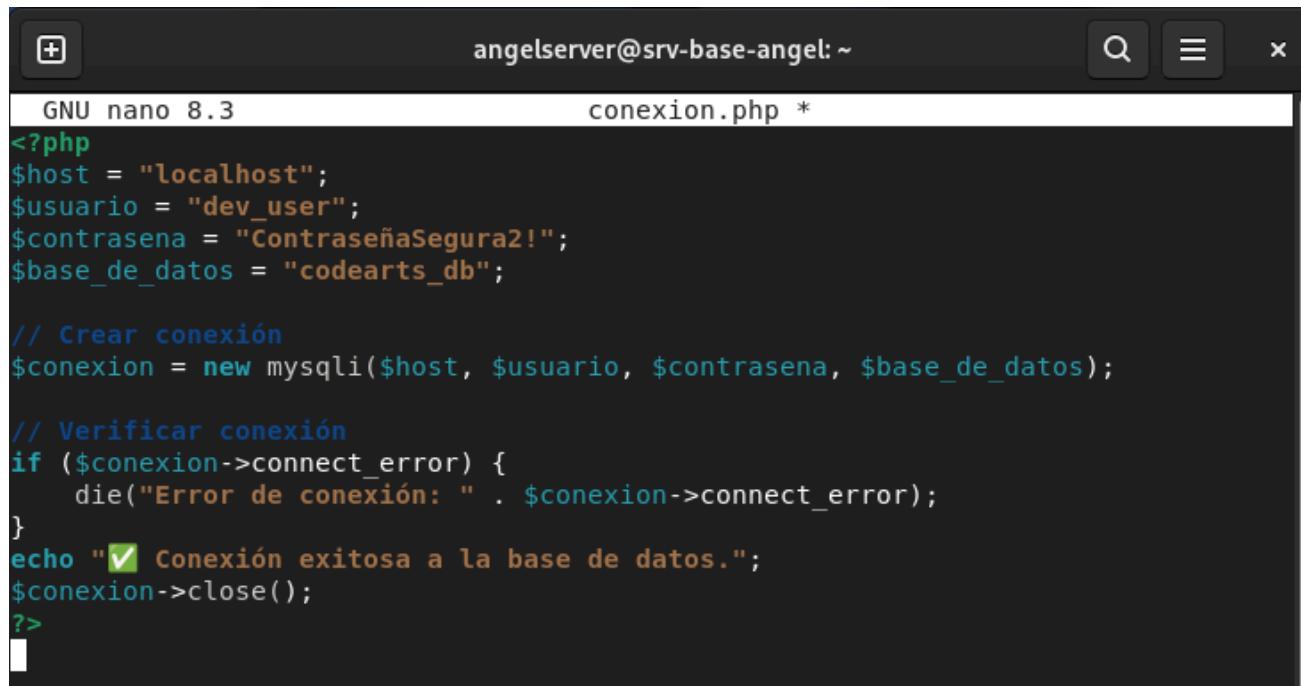
Para permitir la conexión y manipulación de datos desde una aplicación web, se instala **Apache** y **PHP**, para ello usaremos los comandos:

```
sudo apt install apache2 php libapache2-mod-php php-mysql
```

2. Creación del Script de Conexión (conexion.php)

Se crea el archivo `conexion.php` en `/var/www/html` con el siguiente contenido:

```
<?php  
  
$host = "localhost";  
  
$usuario = "dev_user";  
  
$contrasena = "ContraseñaSegura2!";  
  
$base_de_datos = "codearts_db";  
  
  
// Crear conexión  
  
$conexion = new mysqli($host, $usuario, $contrasena, $base_de_datos);  
  
Verificar  
  
// conexión  
  
if ($conexion->connect_error) {  
  
    die("Error de conexión: " . $conexion->connect_error);  
  
}  
  
echo "✓ Conexión exitosa a la base de datos.";  
  
$conexion->close();  
  
?>
```



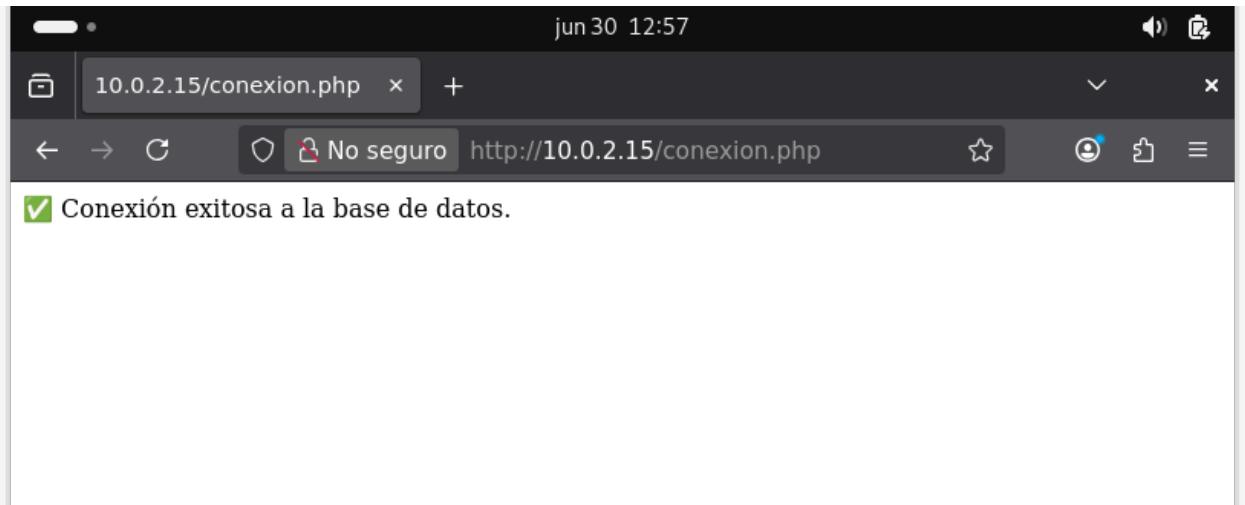
GNU nano 8.3 connexion.php *

```
<?php
$host = "localhost";
$usuario = "dev_user";
$contrasena = "ContraseñaSegura2!";
$base_de_datos = "codearts_db";

// Crear conexión
$conexion = new mysqli($host, $usuario, $contrasena, $base_de_datos);

// Verificar conexión
if ($conexion->connect_error) {
    die("Error de conexión: " . $conexion->connect_error);
}
echo "✓ Conexión exitosa a la base de datos.";
$conexion->close();
?>
```

3. Resultado y comprobación de connexion.php:



4. Pruebas de Consultas SQL desde PHP

Se crea un archivo test.php para realizar operaciones básicas: SELECT, INSERT, UPDATE y DELETE, y verificar que la aplicación interactúa correctamente con la base de datos. Se documentan todos los resultados y posibles errores.

- Abre la terminal y ejecuta: sudo nano /var/www/html/test.php

- En este ejemplo, se introdujo un código a modo de test:

```
<?php  
$host = "localhost";  
$usuario = "dev_user";  
$contrasena = "ContraseñaSegura2!";  
$base_de_datos = "codearts_db";  
  
// Crear conexión  
$conexion = new mysqli($host, $usuario, $contrasena, $base_de_datos);  
  
// Verificar conexión  
if ($conexion->connect_error) {  
    die("Error de conexión: " . $conexion->connect_error);  
}  
echo "<input checked="" type='checkbox'> Conexión exitosa.<br><br>";  
  
// --- INSERT ---  
$sql_insert = "INSERT INTO prueba (nombre, valor) VALUES ('Elemento1', 100);  
if ($conexion->query($sql_insert) === TRUE) {  
    echo "Insert: Registro insertado correctamente.<br>";  
} else {  
    echo "Insert: Error al insertar registro: " . $conexion->error . "<br>";  
}
```

```
// --- SELECT ---
$sql_select = "SELECT * FROM prueba";
$result = $conexion->query($sql_select);

if ($result->num_rows > 0) {
    echo "<br>SELECT: Resultados:<br>";
    while ($row = $result->fetch_assoc()) {
        echo "ID: " . $row["id"] . " - Nombre: " . $row["nombre"] . " - Valor: " . $row["valor"] . "<br>";
    }
} else {
    echo "SELECT: No hay resultados.<br>";
}

// --- UPDATE ---
$sql_update = "UPDATE prueba SET valor = 200 WHERE nombre = 'Elemento1'";
if ($conexion->query($sql_update) === TRUE) {
    echo "<br>Update: Registro actualizado correctamente.<br>";
} else {
    echo "<br>Update: Error al actualizar registro: " . $conexion->error . "<br>";
}

// --- DELETE ---
$sql_delete = "DELETE FROM prueba WHERE nombre = 'Elemento1'";
if ($conexion->query($sql_delete) === TRUE) {
    echo "<br>Delete: Registro eliminado correctamente.<br>";
} else {
    echo "<br>Delete: Error al eliminar registro: " . $conexion->error . "<br>";
}

$conexion->close();
?>
```

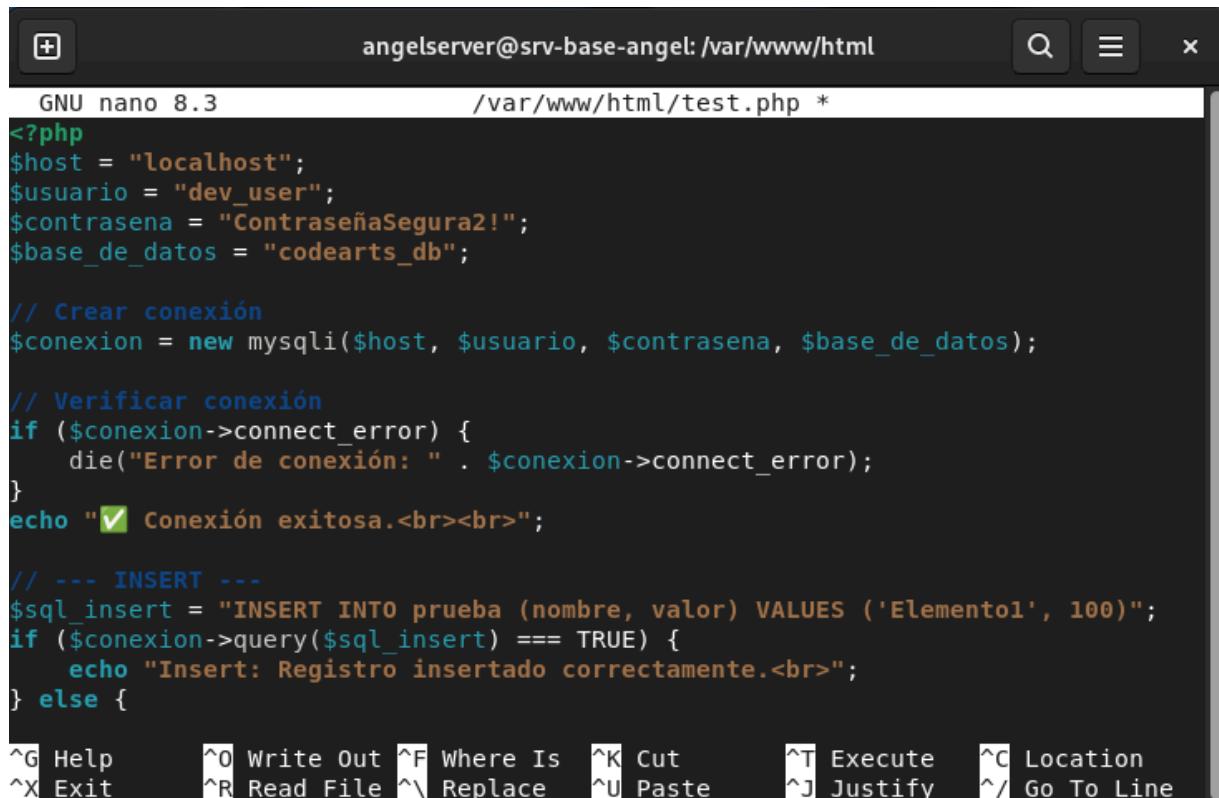
- Después, crearemos la tabla *prueba* en tu base de datos, para ello nos conectamos a MySQL con ***sudo mysql -u root -p***, e introducimos el prompt de MySQL:

```
USE codearts_db;
```

```
CREATE TABLE prueba (
    id INT AUTO_INCREMENT PRIMARY KEY,
    nombre VARCHAR(50),
    valor INT
);
EXIT;
```

- Probamos el script, entrando en el navegador, en nuestro caso con esta dirección:

<http://10.0.2.15/test.php>



```

GNU nano 8.3          /var/www/html/test.php *
<?php
$host = "localhost";
$usuario = "dev_user";
$contrasena = "ContraseñaSegura2!";
$base_de_datos = "codearts_db";

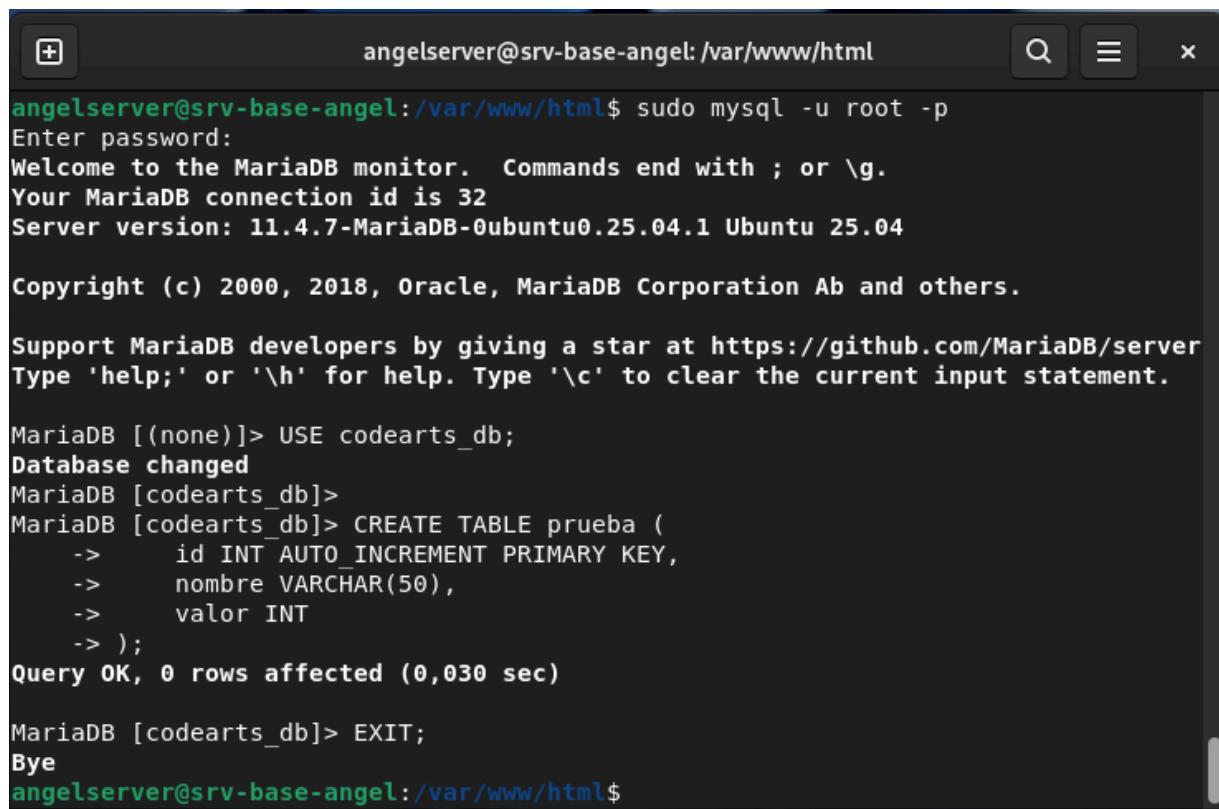
// Crear conexión
$conexion = new mysqli($host, $usuario, $contrasena, $base_de_datos);

// Verificar conexión
if ($conexion->connect_error) {
    die("Error de conexión: " . $conexion->connect_error);
}
echo "✓ Conexión exitosa.<br><br>";

// --- INSERT ---
$sql_insert = "INSERT INTO prueba (nombre, valor) VALUES ('Elemento1', 100)";
if ($conexion->query($sql_insert) === TRUE) {
    echo "Insert: Registro insertado correctamente.<br>";
} else {
    echo "Error: No se pudo insertar el registro." . $conexion->error;
}

^G Help      ^O Write Out ^F Where Is  ^K Cut      ^T Execute   ^C Location
^X Exit     ^R Read File ^\ Replace   ^U Paste    ^J Justify   ^/ Go To Line

```



```

angelserv@srv-base-angel:/var/www/html$ sudo mysql -u root -p
Enter password:
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 32
Server version: 11.4.7-MariaDB-0ubuntu0.25.04.1 Ubuntu 25.04

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Support MariaDB developers by giving a star at https://github.com/MariaDB/server
Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MariaDB [(none)]> USE codearts_db;
Database changed
MariaDB [codearts_db]> CREATE TABLE prueba (
    ->     id INT AUTO_INCREMENT PRIMARY KEY,
    ->     nombre VARCHAR(50),
    ->     valor INT
    -> );
Query OK, 0 rows affected (0,030 sec)

MariaDB [codearts_db]> EXIT;
Bye
angelserv@srv-base-angel:/var/www/html$

```

5. Resultado y comprobación de test.php:



Fase 4: Seguridad y Respaldo de Datos

1. Restricción de Conexiones Remotas

Se edita el archivo de configuración para restringir el acceso al servidor local únicamente:

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

Se verifica que la línea esté activa (sin comentar). Luego se reinicia el servicio:

bind-address = 127.0.0.1

→ → → → → → → → → → →

sudo systemctl restart mariadb

GNU nano 8.3 /etc/mysql/mariadb.conf.d/50-server.cnf

```
# * Basic Settings
#
#user                      = mysql
pid-file                  = /run/mysqld/mysqld.pid
basedir                    = /usr
#datadir                   = /var/lib/mysql
#tmpdir                     = /tmp

# Broken reverse DNS slows down connections considerably and name resolve is
# safe to skip if there are no "host by domain name" access grants
#skip-name-resolve

# Instead of skip-networking the default is now to listen only on
# localhost which is more compatible and is not less secure.
bind-address               = 127.0.0.1

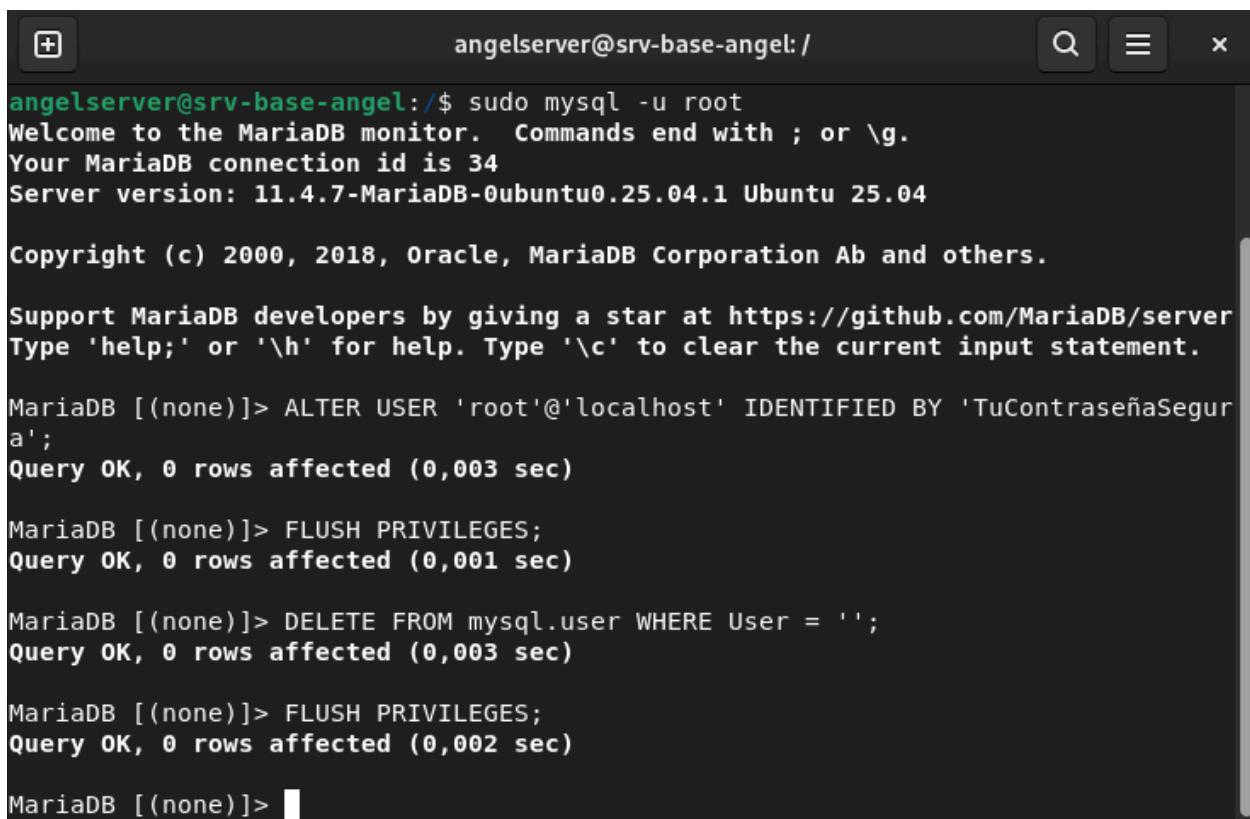
#
# * Fine Tuning
#
^G Help          ^O Write Out  ^F Where Is   ^K Cut        ^T Execute    ^C Location
^X Exit          ^R Read File  ^V Replace     ^U Paste      ^J Justify    ^/ Go To Line
```

2. Ejecución de mysql_secure_installation

Se ejecuta el script para asegurar la instalación:

sudo mysql_secure_installation

- *Se establece la contraseña root.*
- *Se eliminan usuarios anónimos.*
- *Se deshabilita login remoto de root.*
- *Se elimina la base de datos de prueba.*
- *Se recargan los privilegios.*



```
angelsrv@srv-base-angel:/$ sudo mysql -u root
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 34
Server version: 11.4.7-MariaDB-0ubuntu0.25.04.1 Ubuntu 25.04

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Support MariaDB developers by giving a star at https://github.com/MariaDB/server
Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MariaDB [(none)]> ALTER USER 'root'@'localhost' IDENTIFIED BY 'TuContraseñaSegura';
Query OK, 0 rows affected (0,003 sec)

MariaDB [(none)]> FLUSH PRIVILEGES;
Query OK, 0 rows affected (0,001 sec)

MariaDB [(none)]> DELETE FROM mysql.user WHERE User = '';
Query OK, 0 rows affected (0,003 sec)

MariaDB [(none)]> FLUSH PRIVILEGES;
Query OK, 0 rows affected (0,002 sec)

MariaDB [(none)]> █
```

3. Creación de Copias de Seguridad Automáticas

Se crea un script de respaldo `/usr/local/bin/backup_db.sh` con el siguiente contenido:

```
#!/bin/bash  
  
fecha=$(date +%F)  
  
mysqldump -u root -p'MiContraseñaRoot' codearts_db >  
/srv/backups/codearts_db_$fecha.sql
```

Se otorgan permisos de ejecución:

```
sudo chmod +x /usr/local/bin/backup_db.sh
```

Se programa en crontab para ejecución diaria a las 3 AM:

```
sudo crontab -e
```

Se agrega la línea al final del documento:

```
0 3 * * * /usr/local/bin/backup_db.sh
```



The screenshot shows a terminal window titled "angelserver@srv-base-angel: /". The title bar also displays "GNU nano 8.3". The main area of the terminal shows the contents of the file "/usr/local/bin/backup_db.sh". The script is as follows:

```
#!/bin/bash  
fecha=$(date +%F)  
mysqldump -u root -p'TuPasswordRoot' codearts_db > /srv/backups/codearts_db_$fecha.sql
```

```

GNU nano 8.3          /tmp/crontab.6y3X7r/crontab
#
# To define the time you can provide concrete values for
# minute (m), hour (h), day of month (dom), month (mon),
# and day of week (dow) or use '*' in these fields (for 'any').
#
# Notice that tasks will be started based on the cron's system
# daemon's notion of time and timezones.
#
# Output of the crontab jobs (including errors) is sent through
# email to the user the crontab file belongs to (unless redirected).
#
# For example, you can run a backup of all your user accounts
# at 5 a.m every week with:
# 0 5 * * 1 tar -zcf /var/backups/home.tgz /home/
#
# For more information see the manual pages of crontab(5) and cron(8)
#
# m h dom mon dow   command
0 3 * * * /usr/local/bin/backup_db.sh

```

^G Help ^O Write Out ^F Where Is ^K Cut ^T Execute ^C Location
 ^X Exit ^R Read File ^\ Replace ^U Paste ^J Justify ^/ Go To Line

4. Monitoreo y Auditoría mediante Logs

Se verifica la configuración del archivo de logs en:

`/etc/mysql/mariadb.conf.d/50-server.cnf:`

En este caso la opción de log de errores está desactivada, por lo que el servidor no genera logs en archivo. Para activar el log abriremos y editaremos el archivo

/etc/mysql/mariadb.conf.d/50-server.cnf, buscaremos la línea del log_error y la descomentaremos. También se puede añadir si no existe: log_error = /var/log/mysql/error.log

Guarda y cierra.

Luego, crea la carpeta si no existe y asigna permisos, después reinicia:

`sudo mkdir -p /var/log/mysql`

`sudo chown mysql:mysql /var/log/mysql`

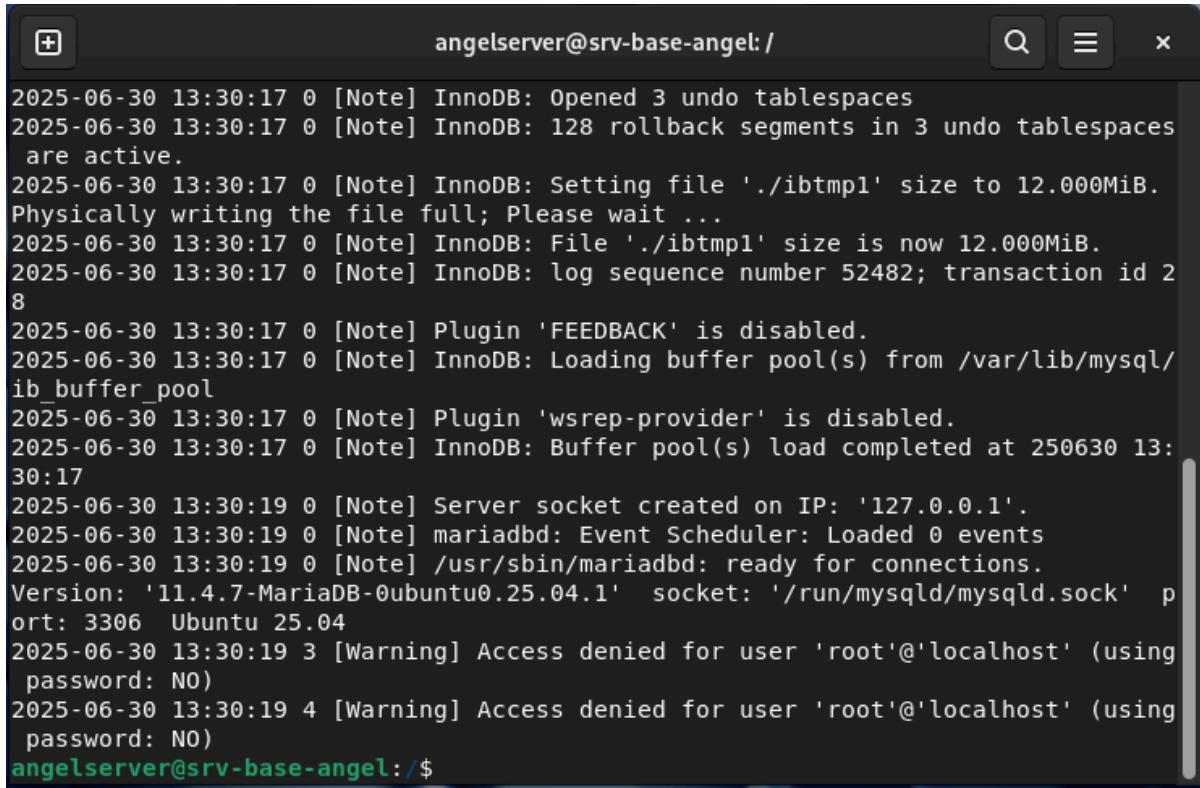
`sudo systemctl restart mariadb`

5. Resultado

Podrás ver y consultar los logs con:

sudo cat /var/log/mysql/error.log

Se explica la importancia de monitorizar intentos fallidos y auditorías para detectar posibles brechas de seguridad.

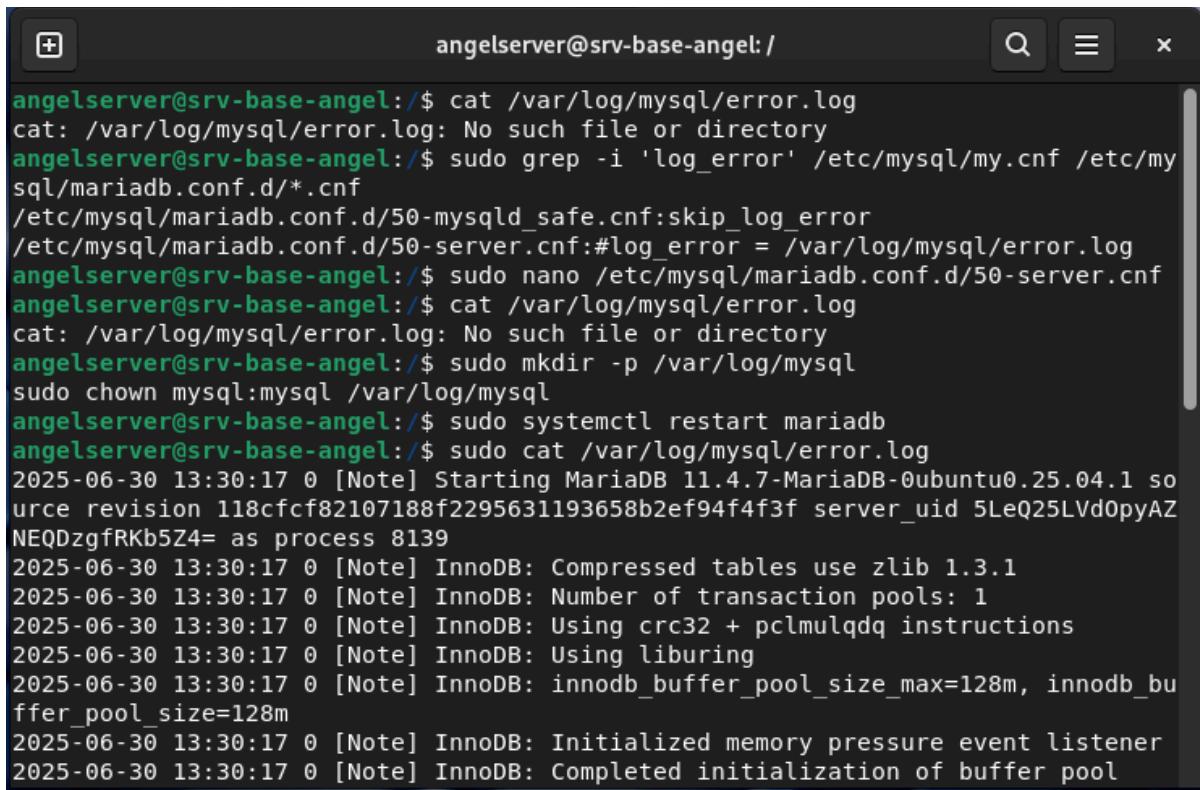


angelserver@srv-base-angel:/

```

2025-06-30 13:30:17 0 [Note] InnoDB: Opened 3 undo tablespaces
2025-06-30 13:30:17 0 [Note] InnoDB: 128 rollback segments in 3 undo tablespaces
are active.
2025-06-30 13:30:17 0 [Note] InnoDB: Setting file './ibtmp1' size to 12.000MiB.
Physically writing the file full; Please wait ...
2025-06-30 13:30:17 0 [Note] InnoDB: File './ibtmp1' size is now 12.000MiB.
2025-06-30 13:30:17 0 [Note] InnoDB: log sequence number 52482; transaction id 2
8
2025-06-30 13:30:17 0 [Note] Plugin 'FEEDBACK' is disabled.
2025-06-30 13:30:17 0 [Note] InnoDB: Loading buffer pool(s) from /var/lib/mysql/
ib_buffer_pool
2025-06-30 13:30:17 0 [Note] Plugin 'wsrep-provider' is disabled.
2025-06-30 13:30:17 0 [Note] InnoDB: Buffer pool(s) load completed at 250630 13:
30:17
2025-06-30 13:30:19 0 [Note] Server socket created on IP: '127.0.0.1'.
2025-06-30 13:30:19 0 [Note] mariadb: Event Scheduler: Loaded 0 events
2025-06-30 13:30:19 0 [Note] /usr/sbin/mariadb: ready for connections.
Version: '11.4.7-MariaDB-0ubuntu0.25.04.1' socket: '/run/mysqld/mysqld.sock' p
ort: 3306 Ubuntu 25.04
2025-06-30 13:30:19 3 [Warning] Access denied for user 'root'@'localhost' (using
password: NO)
2025-06-30 13:30:19 4 [Warning] Access denied for user 'root'@'localhost' (using
password: NO)
angelserver@srv-base-angel:/$

```



angelserver@srv-base-angel:/

```

angelserver@srv-base-angel:/$ cat /var/log/mysql/error.log
cat: /var/log/mysql/error.log: No such file or directory
angelserver@srv-base-angel:/$ sudo grep -i 'log_error' /etc/mysql/my.cnf /etc/my
sql/mariadb.conf.d/*.cnf
/etc/mysql/mariadb.conf.d/50-mysqld_safe.cnf:skip_log_error
/etc/mysql/mariadb.conf.d/50-server.cnf:#log_error = /var/log/mysql/error.log
angelserver@srv-base-angel:/$ sudo nano /etc/mysql/mariadb.conf.d/50-server.cnf
angelserver@srv-base-angel:/$ cat /var/log/mysql/error.log
cat: /var/log/mysql/error.log: No such file or directory
angelserver@srv-base-angel:/$ sudo mkdir -p /var/log/mysql
sudo chown mysql:mysql /var/log/mysql
angelserver@srv-base-angel:/$ sudo systemctl restart mariadb
angelserver@srv-base-angel:/$ sudo cat /var/log/mysql/error.log
2025-06-30 13:30:17 0 [Note] Starting MariaDB 11.4.7-MariaDB-0ubuntu0.25.04.1 so
urce revision 118cfccf82107188f2295631193658b2ef94f4f3f server_uid 5LeQ25LVd0pyAZ
NEQDzgfRkb5Z4= as process 8139
2025-06-30 13:30:17 0 [Note] InnoDB: Compressed tables use zlib 1.3.1
2025-06-30 13:30:17 0 [Note] InnoDB: Number of transaction pools: 1
2025-06-30 13:30:17 0 [Note] InnoDB: Using crc32 + pclmulqdq instructions
2025-06-30 13:30:17 0 [Note] InnoDB: Using liburing
2025-06-30 13:30:17 0 [Note] InnoDB: innodb_buffer_pool_size_max=128m, innodb_bu
ffer_pool_size=128m
2025-06-30 13:30:17 0 [Note] InnoDB: Initialized memory pressure event listener
2025-06-30 13:30:17 0 [Note] InnoDB: Completed initialization of buffer pool

```

Conclusión

La instalación y configuración del servidor de bases de datos ha permitido a **Codearts Solutions** disponer de una infraestructura estable, segura y eficiente para la gestión de su información. Se han definido niveles de acceso, implementado copias de seguridad automáticas y conectado una aplicación web funcional, garantizando así integridad y disponibilidad de los datos.

Este sistema no solo cumple con los requisitos actuales, sino que también está preparado para escalar y adaptarse a futuras necesidades de la empresa.