

Instalar GLPI en Ubuntu

Primero actualizar el sistema con:

```
sudo apt update  
sudo apt upgrade
```

Instalar Apache, PHP y otros paquetes necesarios:

```
sudo apt install apache2 php mysql-server php-mysql php-curl php-gd  
php-ldap php-xml php-mbstring php-cli php-imap php-xmlrpc php-imagick  
php-json php-bcmath php-ctype php-fileinfo php-zip
```

Descargar y descomprimir la última versión de GLPI:

```
wget https://github.com/glpi-project/glpi/releases/download/9.6.3/glpi-9.6.3.tgz
```

```
tar -xvzf glpi-9.6.3.tgz
```

```
sudo mv glpi /var/www/html/
```

Configurar los permisos y ajustar la configuración de Apache:

```
sudo chown -R www-data:www-data /var/www/html/glpi  
sudo chmod -R 755 /var/www/html/glpi  
sudo a2enmod rewrite  
sudo systemctl restart apache2
```

Instalar MySQL para GLPI

Instalar MySQL Server:

```
sudo apt install mysql-server
```

Crear una base de datos y un usuario para GLPI:


```
mysql -u root -p
```

```
CREATE DATABASE glpidb CHARACTER SET utf8mb4 COLLATE  
utf8mb4_unicode_ci;  
CREATE USER 'glpiuser'@'localhost' IDENTIFIED BY 'password';  
GRANT ALL PRIVILEGES ON glpidb.* TO 'glpiuser'@'localhost';
```

```
FLUSH PRIVILEGES;
```

```
EXIT;
```

Una vez instalado todo ya se puede acceder a GLPI en la web, para ello entrar en **localhost** desde cualquier navegador:

 `localhost/front/central.php`

Aquí instalar GLPI accediendo a localhost en cualquier navegador y usar base de datos creada.

Configurar Docker Compose para Hasura y PostgreSQL

Crear un archivo `docker-compose.yml` con el siguiente contenido:

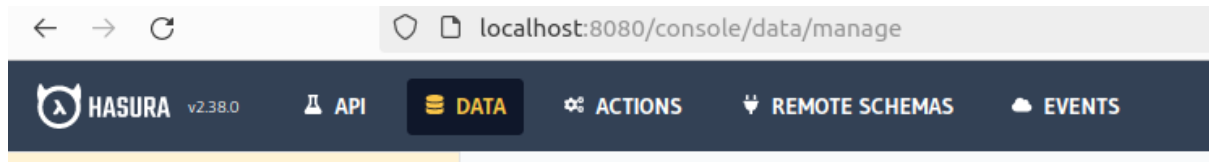
```
version: "3.6"
services:
  postgres:
    image: postgres:15
    restart: always
    volumes:
      - db_data:/var/lib/postgresql/data
    environment:
      POSTGRES_PASSWORD: postgrespassword
  graphql-engine:
    image: hasura/graphql-engine:v2.38.0
    ports:
      - "8080:8080"
    restart: always
    environment:
      ## postgres database to store Hasura metadata
      HASURA_GRAPHQL_METADATA_DATABASE_URL: postgres://postgres:postgrespassword>
      ## this env var can be used to add the above postgres database to Hasura >
      PG_DATABASE_URL: postgres://postgres:postgrespassword@postgres:5432/postg>
      ## enable the console served by server
      HASURA_GRAPHQL_ENABLE_CONSOLE: "true" # set to "false" to disable console
      ## enable debugging mode. It is recommended to disable this in production
      HASURA_GRAPHQL_DEV_MODE: "true"
      HASURA_GRAPHQL_ENABLED_LOG_TYPES: startup, http-log, webhook-log, websocket>
      ## uncomment next line to run console offline (i.e load console assets fr>
      # HASURA_GRAPHQL_CONSOLE_ASSETS_DIR: /srv/console-assets
      ## uncomment next line to set an admin secret
      # HASURA_GRAPHQL_ADMIN_SECRET: myadminsecretkey
      HASURA_GRAPHQL_METADATA_DEFAULTS: '{"backend_configs":{"dataconnector":{">
    depends_on:
      data-connector-agent:
        condition: service_healthy
  data-connector-agent:
    image: hasura/graphql-data-connector:v2.38.0
    restart: always
    ports:
      - 8081:8081
    environment:
      QUARKUS_LOG_LEVEL: ERROR # FATAL, ERROR, WARN, INFO, DEBUG, TRACE
      ## https://quarkus.io/guides/opentelemetry#configuration-reference
      QUARKUS_OPENTELEMETRY_ENABLED: "false"
      ## QUARKUS_OPENTELEMETRY_TRACER_EXPORTER_OTLP_ENDPOINT: http://jaeger:4317
    healthcheck:
      test: ["CMD", "curl", "-f", "http://localhost:8081/api/v1/athena/health"]
      interval: 5s
      timeout: 10s
      retries: 5
      start_period: 5s
volumes:
  db_data:
```

Levantar docker:

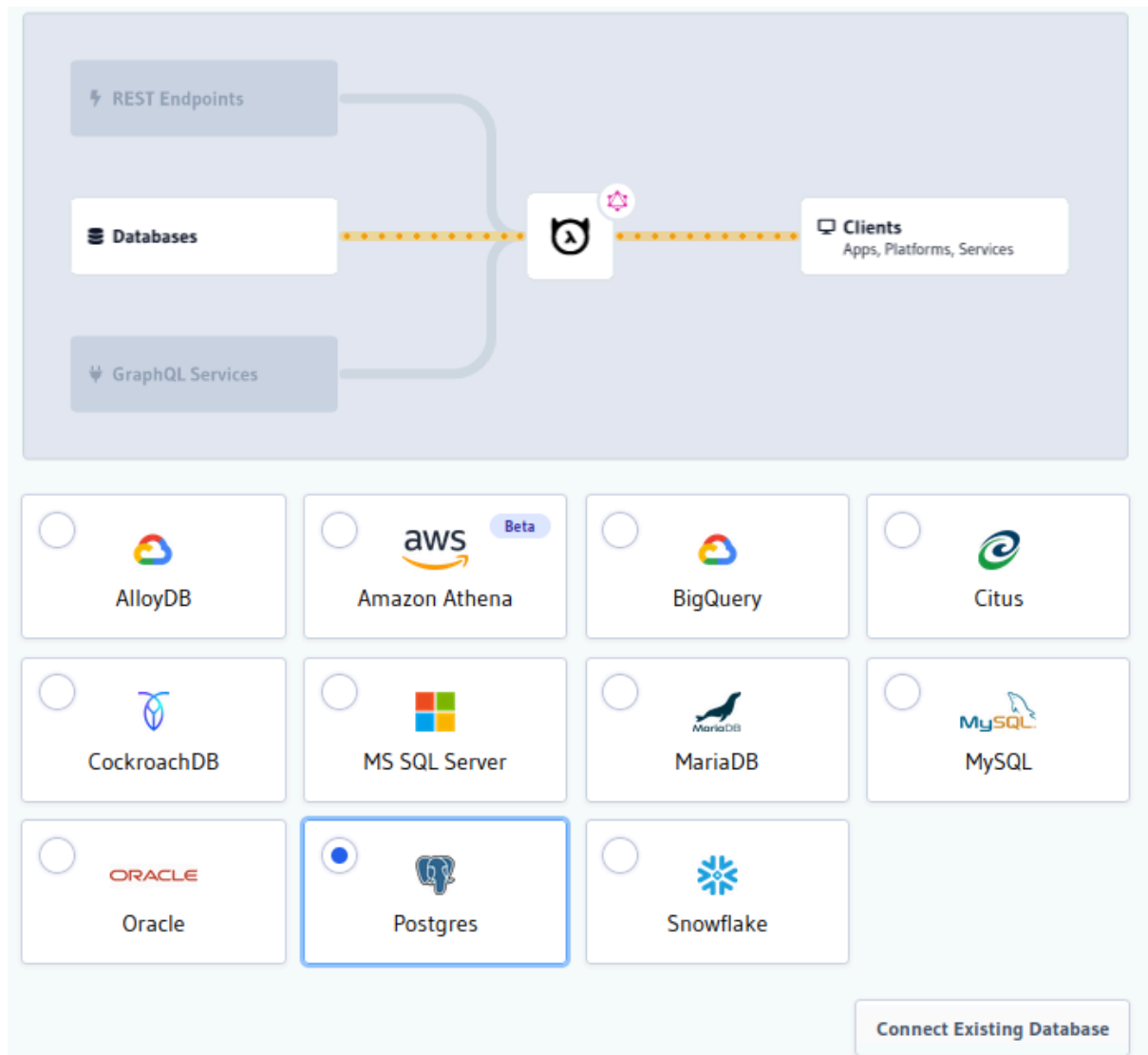
docker-compose up -d

Acceder en localhost:8080

Dirigirse a “DATA”



Escoger postgres




Conectar via URL, **postgres://postgres:postgrespassword@postgres:5432/postgres**

Database name

postgres

Connect Database via ?

☒ Database URL ☐ Environment variable ☐ Connection Parameters ☐ Dynamic URL

**Warning**
This option **exposes sensitive information** such as password and hostname **in your metadata**.
The **recommended way** of adding connections is using an **Environment variable**. [\(Learn More\)](#)

Database URL

postgres://postgres:postgrespassword@postgres:5432/postgres

Servicios activos:

Para comprobar si MySQL está activo:

sudo systemctl status mysql

Para verificar si apache está activo:

sudo systemctl status apache2

Para verificar GLPI:

Abrir **localhost** en cualquier navegador

Para comprobar hasura y postgre:

sudo docker ps

```
CONTAINER ID   IMAGE
347f2b6dc95b   hasura/graphql-engine:v2.38.0
aa614d432aea   postgres:15
576e26681fbc   hasura/graphql-data-connector:v2.38.0
```

Seguridad del sistema y de los datos

Crear una carpeta para los backups, en este caso /home/alex/backupsdb

Tener en cuenta que lo optimo seria hacer los backups en un medio de memoria externo y mantener 2 copias de backups, una semana se hacen en el medio1 y en la semana 2 el medio1 se cambia por el medio2 así sucesivamente.

```
sudo mkdir /home/alex/buckupsdb
```

Crear un script python

```
sudo apt.get install python3
```

```
sudo nano backups.py
```

Pegar esto

```
import subprocess  
from datetime import datetime  
  
DIRECTORIO_RESPALDO = "/home/alex/backupsdb/"  
  
fecha_actual = datetime.now()  
nombre_dia_semana = fecha_actual.strftime("%A").lower()  
  
nombre_contenedor = "aa614d432aea"  
  
with open(f'{DIRECTORIO_RESPALDO}/glpi-{nombre_dia_semana}.sql', 'w') as f:  
    subprocess.Popen(['/usr/bin/mysqldump', '-u', 'alex', '-pPassword1234', 'glpidb'],  
        stdout=f)  
with open(f'{DIRECTORIO_RESPALDO}/postgres-{nombre_dia_semana}.sql', 'w') as f:  
    subprocess.run(['sudo', 'docker', 'exec', nombre_contenedor, 'pg_dump', '-U', 'postgres',  
        '-h', 'localhost', '-W', 'postgres'], stdout=f)  
subprocess.run(['rsync', '-avz', DIRECTORIO_RESPALDO])
```

Para automatizar la ejecución comando

```
crontab -e
```

```
1
```

Escribir lo siguiente

```
0 2 * * * /usr/bin/python3 /home/alex/backups.py
```

Syntaxis, [minutos] [horas] [día_del_mes] [mes] [día_de_la_semana]

Minutos: 0-59.

Horas: 0-23.

Día del mes: 1-31.

Mes: 1-12

Día de la semana: 0-7

En este caso se ejecuta una copia en el minuto 0 de la hora 2, de todos los días del mes, todos los meses, todos los días de la semana.

Instalar Zabbix

```
sudo zypper install php-fpm  
a2enmod proxy_fcgi setenvif  
a2enconf php8.1-fpm
```

```
sudo systemctl restart apache2
```

```
sudo apt install zabbix-server-mysql zabbix-frontend-php zabbix-agent
```