Instalación paquetes generales

Este comando actualiza la lista de paquetes disponibles en el sistema. Básicamente, se asegura de que el sistema conozca las versiones más recientes de los paquetes disponibles en los repositorios configurados. Es un paso preliminar necesario antes de instalar o actualizar paquetes.

```
# Actualizar el sistema
# Instalación de paquetes necesarios

apt-get update

apt-get install redis git curl wget unzip nginx mysql-server supervisor -y
```

Instalación PHP

Este bloque agrega un repositorio de PHP actualizado, luego instala PHP 8.2 junto con una serie de extensiones esenciales como soporte para bases de datos (MySQL, PostgreSQL, SQLite), manejo de archivos ZIP, gráficos, redes, y otros módulos necesarios para el desarrollo y ejecución de aplicaciones PHP modernas.

```
# Instalación de PHP 8.2 y paquetes necesarios que Laravel necesita

add-apt-repository ppa:ondrej/php
apt-get update
apt-get -y install php8.2-{cli,common,fpm,mysql,mbstring,zip,gd,curl,dev,pgsql,
sqlite3,imap,bcmath,soap,intl,readline,gmp,redis,memcached,memcache}
apt install php8.2-xml
```

Instalación composer

Otro paquete indispensable es composer, este comando descarga el instalador de **Composer**, un administrador de dependencias para PHP, desde su sitio oficial y lo guarda temporalmente en el directorio `/tmp`. Composer es esencial para gestionar las bibliotecas y paquetes que se utilizan en proyectos PHP.

```
# Instalación de Composer

cd ~

curl -sS https://getcomposer.org/installer -o /tmp/composer-setup.php

HASH=`curl -sS https://composer.github.io/installer.sig`

php -r "if (hash_file('SHA384', '/tmp/composer-setup.php') === '$HASH') { echo
'Installer verified'; } else { echo 'Installer corrupt'; unlink('composer-setup.
php'); } echo PHP_EOL;"

sudo php /tmp/composer-setup.php --install-dir=/usr/local/bin --filename=composer
```

Configuración MySql

En este caso la configuración básica de MySql es agregarle una contraseña

```
# Configuración de MySQL

ALTER USER 'root'@'localhost' IDENTIFIED WITH caching_sha2_password BY 'testroot';
```

Configuración Nginx

Esta configuración de **Nginx** define cómo el servidor web debe manejar las solicitudes para el dominio `diploy.sh`. A continuación, se detallan los aspectos principales:

1. **server_name**: Define el nombre del servidor o dominio, en este caso `diploy.sh`.
2. **root**: Establece el directorio raíz para el proyecto Laravel, apuntando a `/var/www/cf-laravel-integrador/public`, que es el directorio público de Laravel.
3. **index**: Establece `index.php` como el archivo de inicio para las solicitudes.
4. **location /**: Intenta servir las solicitudes de archivos estáticos, y si no los encuentra, las redirige a `index.php`.
5. **location ~ .php$**: Configura el manejo de archivos PHP usando PHP-FPM a través del socket `php8.2-fpm.sock`, permitiendo la ejecución de scripts PHP.

Finalmente, luego de hacer estos cambios en `/etc/nginx/sites-available/default`, es necesario reiniciar el servicio de Nginx con el comando `service nginx restart` para aplicar la configuración.

```
server {
    server_name diploy.sh;
    root /var/www/cf-laravel-integrador/public;

    index index.php;

    add_header X-Frame-Options "SAMEORIGIN";
    add_header X-Content-Type-Options "nosniff";

    location / {
        try_files $uri $uri/ /index.php?$query_string;
    }

    location ~ \.php$ {
        fastcgi_pass unix:/var/run/php/php8.2-fpm.sock;
        fastcgi_param SCRIPT_FILENAME $realpath_root$fastcgi_script_name;
        include fastcgi_params;
    }

    location ~ /\.(?!well-known).* {
        deny all;
    }
}
```

Supervisord

Esta configuración de Supervisor se encarga de gestionar y monitorear los trabajadores de colas de Laravel, asegurando que siempre estén activos y funcionando correctamente. A continuación se describen los parámetros más relevantes:

process_name: Define el nombre del proceso y agrega un sufijo numérico para distinguir múltiples instancias.

command: Especifica el comando a ejecutar, en este caso, el comando de Laravel para procesar las colas (php artisan queue:work).

autostart y autorestart: Indican que el proceso debe iniciarse automáticamente al arrancar el servidor y reiniciarse si se detiene inesperadamente.

stopasgroup y killasgroup: Garantizan que tanto el proceso principal como los procesos secundarios se detengan o finalicen juntos.

user: Define al usuario (root) bajo el cual se ejecuta el proceso.

numprocs: Establece el número de procesos simultáneos, en este caso, 1.

redirect_stderr: Redirige los errores estándar a los archivos de log.

stdout_logfile: Especifica el archivo de log donde se registrará la salida estándar del proceso (/root/worker.log).

stopwaitsecs: Define el tiempo de espera antes de que el proceso se fuerce a detener, aquí se da un largo período de 3600 segundos (1 hora).

Esta configuración asegura que los trabajos en cola de Laravel se procesan continuamente en segundo plano.

```
# Configuración de supervisor

[program:laravel-worker]
process_name=%(program_name)s_%(process_num)02d
command=php /var/www/cf-laravel-integrador/artisan queue:work
autostart=true
autorestart=true
stopasgroup=true
killasgroup=true
user=root
numprocs=1
redirect_stderr=true
stdout_logfile=/root/worker.log
stopwaitsecs=3600
```