BOT DISCORD

INTRODUCCIÓN

A continuación, se muestra un manual de usuario para poder instalar y configurar un Bot de Discord que avise cuándo se enciende o se apague un sistema mediante un mensaje enviado a un canal de texto en un servidor de Discord.

En resumidas cuentas, la configuración consta de la creación de un script **Unit** personalizado (Systemd), que se ejecutará automáticamente al iniciar el sistema, y de otros scripts que facilitarán las herramientas necesarias para llevar a cabo el propósito de la aplicación.

También cabe destacar que se utilizará la API de Discord para realizar el envío de mensajes por parte del Bot utilizando el protocolo **HTTP**.

REQUISITOS PREVIOS

- Sistema Linux instalado en un pc o máquina virtual. Distribución Debian o Ubuntu.
- Navegador web. Mozilla Firefox,Google Chrome, Opera, Microsoft Edge, etc…
- Cuenta de usuario registrado en Discord.
- Tener instalado Systemd, python3 en el sistema linux operativo.

PRIMER PASO

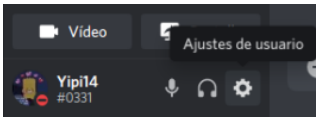
Consiste en la creación del Servidor de Discord donde se aloja el Bot.

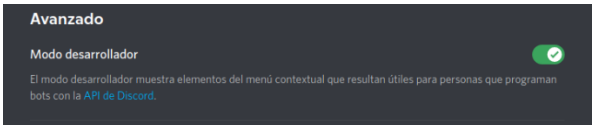
Al iniciar la aplicación de Discord en la esquina inferior izquierda se encuentra un botón con el icono '+' el cual permite añadir un servidor. A continuación, muestra una interfaz muy intuitiva para personalizar el servidor a gusto propio de cada uno.

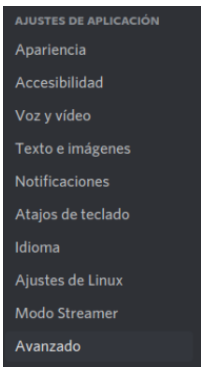
CREACIÓN DEL BOT

A continuación se debe seguir unos pasos previos a la creación del bot desde la aplicación de Discord:

Ajustes de Usuario - AJUSTES DE APLICACIÓN (AVANZADO) - Modo desarrollador (Activado)



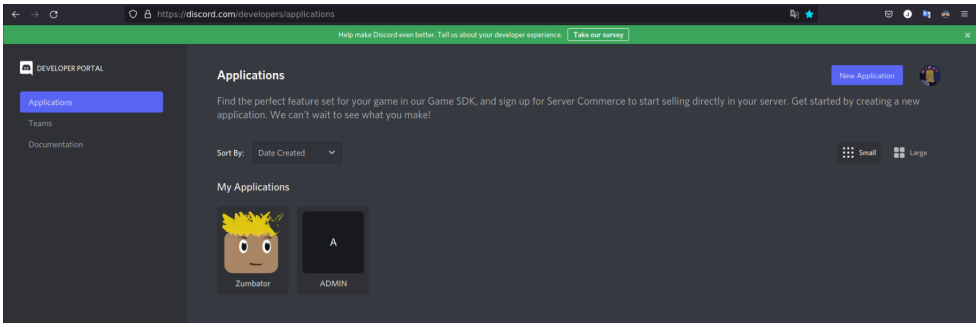


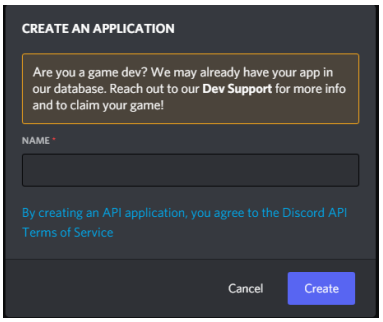


Ahora se muestra la creación del bot:

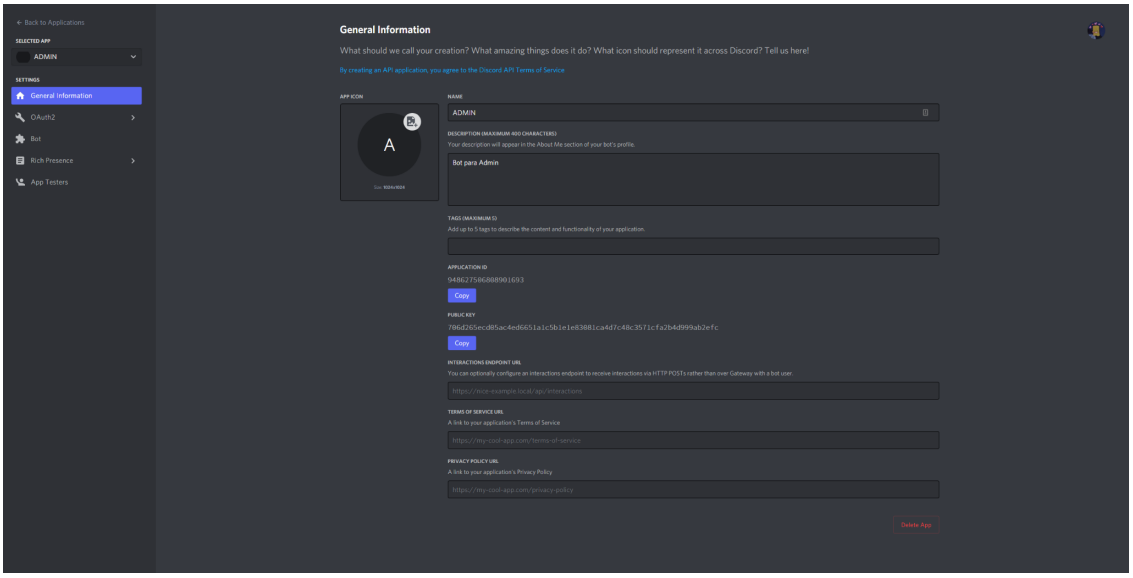
En la descripción del Modo desarrollador se observa en azul resaltado API de Discord. Seleccionar **Applications**. Si no se encuentra desde el link referido anteriormente, introducir la url https://discord.com/developers/applications.

Se mostrará la siguiente interfaz:

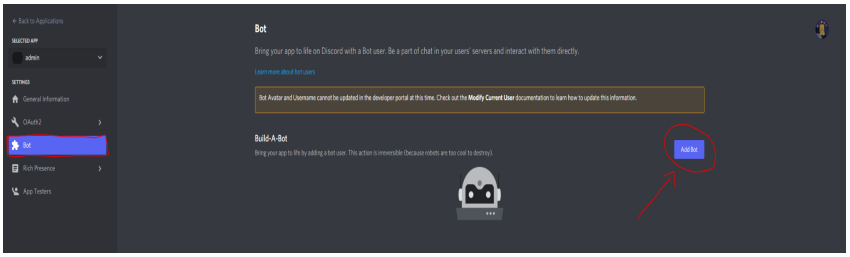


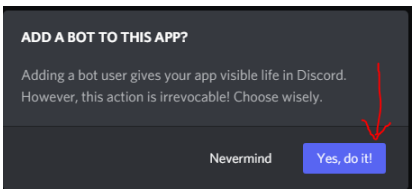


Seleccionar **New Application** que se sitúa en la parte superior derecha.
Muestra una interfaz que solicita un nombre de la aplicación.
El siguiente paso es trabajar sobre la interfaz de la aplicación creada, en la que se observa lo siguiente:

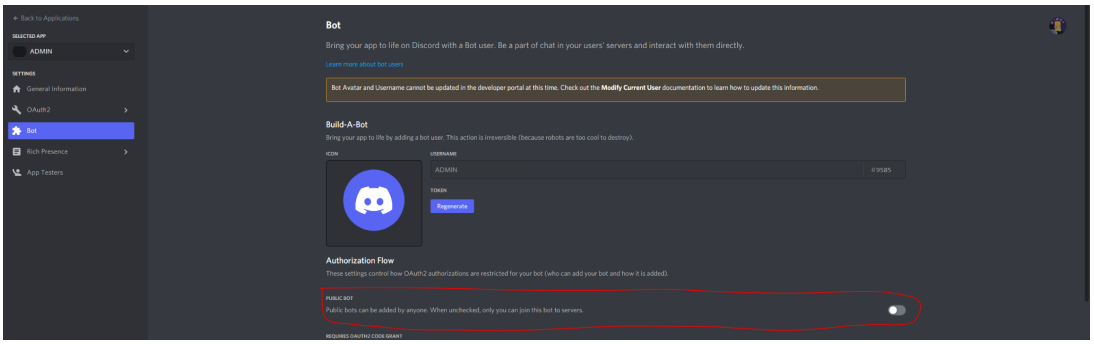


Permite modificar el nombre, crear una descripción, visualizar el ID de la aplicación y más opciones que no se utilizarán para el desarrollo de la aplicación.
Seleccionar el apartado **Bot**. Y seleccionar **Add the Bot**.

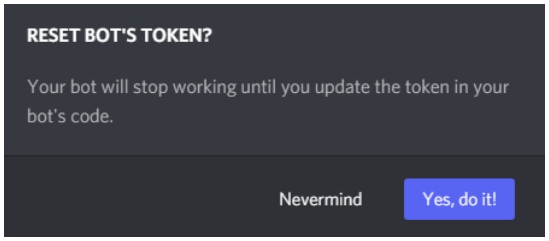




Se muestra la ventana emergente siguiente en la que debemos seleccionar: **Yes, do it!**
Al seleccionar lo anterior se muestra la siguiente interfaz, en la que se debe **deseleccionar** public bot. De esta manera el bot será de uso privado y solo podrá estar en servidores propios del creador. De la otra forma será público y cualquiera podrá utilizarlo.

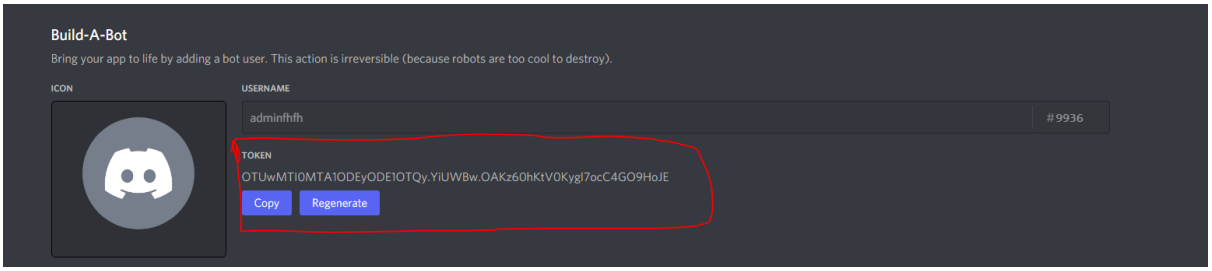


La característica más importante que se encuentra es el **TOKEN,** se sitúa debajo del nombre, el cual se debe regenerar seleccionando **regenerate**.



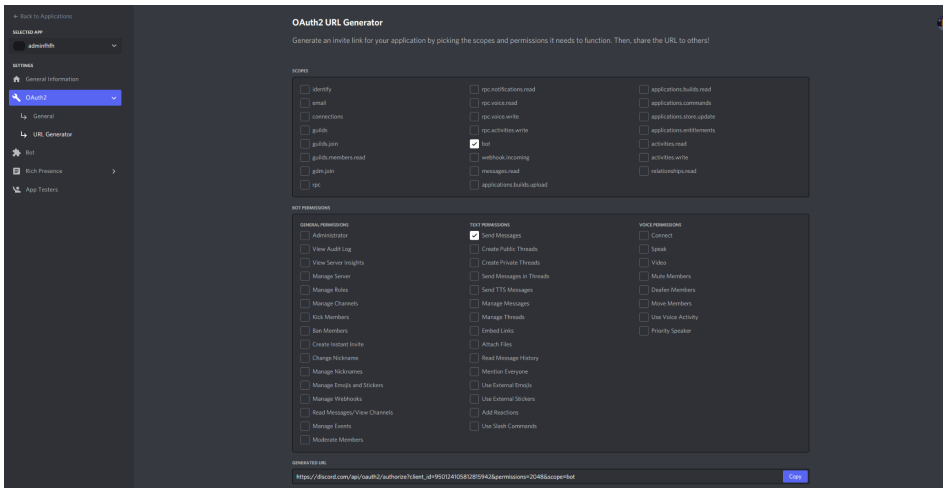
Al darle, se muestra una ventana emergente que nos advierte que el bot dejará de funcionar si no actualizamos en nuestro código el token. Si tenemos la 2FA-Authentication activada nos solicitará el código.

A continuación se muestra el token, se debe guardar en un sitio seguro este conjunto de caracteres alfanuméricos y nunca compartirlo.

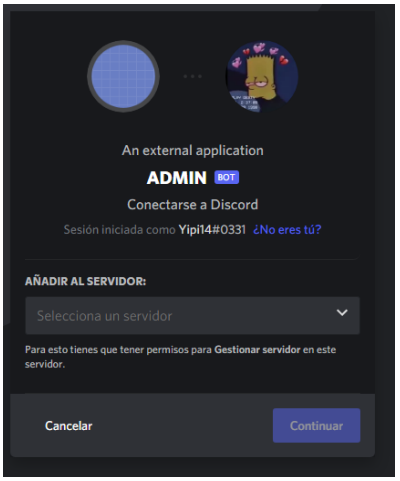


INTRODUCIR EL BOT EN EL SERVIDOR

Ir a la sección OAuth2 – URL Generator – Scopes: Bot – Bot Permissions: Send Messages



Después se copia la URL de abajo generada y se introduce en un navegador web. Se da a la opción de añadir al servidor, seleccionando el servidor al que se desea que trabaje.



CREACIÓN SCRIPTS LINUX

Se debe tener instalado Systemd en la máquina linux utilizada ya que utiliza este sistema de control de los servicios.
También se debe tener instalado python 3, normalmente viene en el sistema por defecto.

Lo primero de todo es crear un script en bash.

Se llamará **BotDiscord.sh** en el se crea dos variables: **BOT_TOKEN** y **CHAT_ID**. El primero se rellena con cualquier string y el segundo con números cualquiera ya que estas dos variables se autorellenan mediante otro script que se creará para llevar de manera automática la instalación de todo.
Posteriormente se hace un case que es referido al primer argumento que acompaña al ejecutarse el script BotDiscord.sh [start/stop]. Cuando sea **start** en la variable **TEXT** guardaremos Sistema-Iniciado:(Fecha del sistema). Lo mismo para stop pero en vez de iniciado,apagado.
Por último se ejecuta un script en python pasando tres argumentos: **TEXT**,**BOT_TOKEN** y **CHAT_ID**.
Será el encargado de enviar los mensajes por el protocolo **HTTP**

```bash
#!/bin/bash

BOT_TOKEN=a87543rada78raw8a4ra1adw
CHAT_ID=12312321321231231233123

case "$1" in
    start)
        TEXT="Sistema-Iniciado:`date +%d/%m/%Y/%R`"
    ;;
    stop)
        TEXT="Sistema-Apagado:`date +%d/%m/%Y/%R`"
    ;;
esac
python3 /usr/bin/post-HTTP.py $TEXT $BOT_TOKEN $CHAT_ID
```

post-HTTP.py importamos la librería sys y requests ya que las vamos a necesitar para el funcionamiento de la aplicación. Definimos una función que se encarga del envío del mensaje guardando la url que conecta con la API de Discord, donde están los corchetes es donde se guarda el ID del canal a enviar el mensaje. Data (Key = “content”) donde se guarda el mensaje a enviar y el header (Key = “authorization”) que lleva como contenido la palabra Bot y su token correspondiente.
Se usa la librería requests y se envía un POST con toda la información anterior.
Finalmente se tiene el main, que es donde se cogen todos los argumentos pasados como vimos en el script anterior que serán el mensaje, el token del Bot y el ID del canal correspondiente donde queremos enviar el mensaje. Y el llamamiento a la función anterior.

```python
#!/usr/bin/python
import sys
import requests

def sendMessage(token,chat_ID,message):
    url = 'https://discord.com/api/v9/channels/{}/messages'.format(chat_ID)
    data = {"content": message}
    header = {"authorization": 'Bot ' + token}
    r = requests.post(url,data=data,headers=header)
    print(r.status_code)

def main():
    info = sys.argv
    message = info[1]
    token = info[2]
    chat_ID = info[3]
    sendMessage(token,chat_ID,message)

if __name__ == "__main__":
    main()
```

Ahora toca crear **BotDiscord.service** que es el archivo **Unit** que ejecutará el daemon del proceso.
Lo primero es entre corchetes [Unit] donde se añade la descripción de la aplicación e indicamos con el **Before** que tiene que ejecutarse antes que esos servicios, se indican con el .target.
Después [Service] en este apartado incluimos **ExecStart** que se refiere a la ejecución nada más comenzar, se le indica el primer script hecho en bash con el argumento **start**

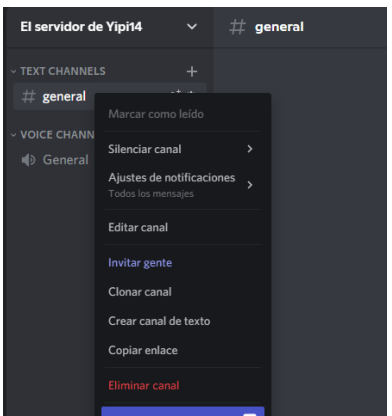
ExecStop es al finalizar la ejecución lo mismo que ExecStart pero pasando el argumento **stop**. De ahí que el case $1 del script en Bash en la variable **TEXT** guarde un texto u otro.
RemainAfterExit=yes nos indica que el servicio se considera activo incluso cuando todos sus procesos han salido.
WantedBy indica en que runlevel se inicia. Multi-user.target, es el que se usa por defecto.

```
[Unit]
Description=Envia un mensaje a un canal de un servidor de Discord mediante un bot cuando el sistema se apaga/reinicia
Before=shutdown.target halt.target poweroff.target reboot.target

[Service]
ExecStart=/usr/bin/BotDiscord.sh start
ExecStop=/usr/bin/BotDiscord.sh stop
RemainAfterExit=yes

[Install]
WantedBy=multi-user.target
```

Ahora se crea el instalador en bash. Se llamará **installer.sh.** Creamos dos ifs para determinar que tienen que pasarse dos argumentos, el **BOT_TOKEN** copiado en un punto anterior y el **CHAT_ID** del chat al que se va a enviar el mensaje, posteriormente se enseña como obtener **CHAT_ID** en **Discord.**
También se copia mediante **cp** el BotDiscord.service anteriormente creado en la carpeta system donde se alojan los servicios. Mediante **chmod 777** le damos los permisos que necesita para ello.
Lo siguiente es reescribir en el script ejecutable por el servicio para que guarde en las variables el token del bot y el id del chat.
Mediante el comando sed optimizamos el espacio del script. Se hace un **chmod 777** al archivo script en bash. Se copia el script en python a usr/bin que es donde se ejecutan los scripts. Se le da permisos de **chmod** 777 para que pueda ser ejecutado.
Finalmente se realizan los comandos de **systemctl** para activar el servicio con **enable** y **start** para activarlo. Por último se refrescan los daemons mediante el último comando.

```
#!/bin/bash

if [[ "$#" == "1" && "$1" == "-h" ]]; then
    echo "Usa: ./installer.sh BOT_TOKEN CHAT_ID"
    exit 1
fi

if [[ "$#" != "2" ]]; then
    echo "Numero de argumentos invalido"
    echo "Usa: ./installer.sh BOT_TOKEN CHAT_ID"
    exit 1
fi


cp BotDiscord.service /etc/systemd/system
chmod 777 /etc/systemd/system/BotDiscord.service

echo -e "#!/bin/bash\n\nBOT_TOKEN=$1\nCHAT_ID=$2\n" > /usr/bin/BotDiscord.sh
sed -n '6,14p' BotDiscord.sh >> /usr/bin/BotDiscord.sh
chmod 777 /usr/bin/BotDiscord.sh
cp post-HTTP.py /usr/bin
chmod 777 /usr/bin/post-HTTP.py

systemctl enable BotDiscord.service
systemctl start BotDiscord.service
systemctl daemon-reload
```

Se guarda el script y se le da permisos de ejecución mediante **chmod 777** y lo ejecutamos siendo usuario root con el siguiente comando que automatiza todo el proceso de instalación: **sudo ./installer BOT_TOKEN CHAT_ID .** Siendo BOT_TOKEN la configuración alfanumérica de caracteres ya mencionada en puntos anteriores y el CHAT_ID se consigue de la siguiente forma:
Hay que dirigirse a la aplicación de Discord, luego al servidor, en text channels con click derecho seleccione el canal al que desea enviar los mensajes y darle a copiar ID. Pegarla en la ejecución del instalador junto con el BOT_TOKEN y listo.



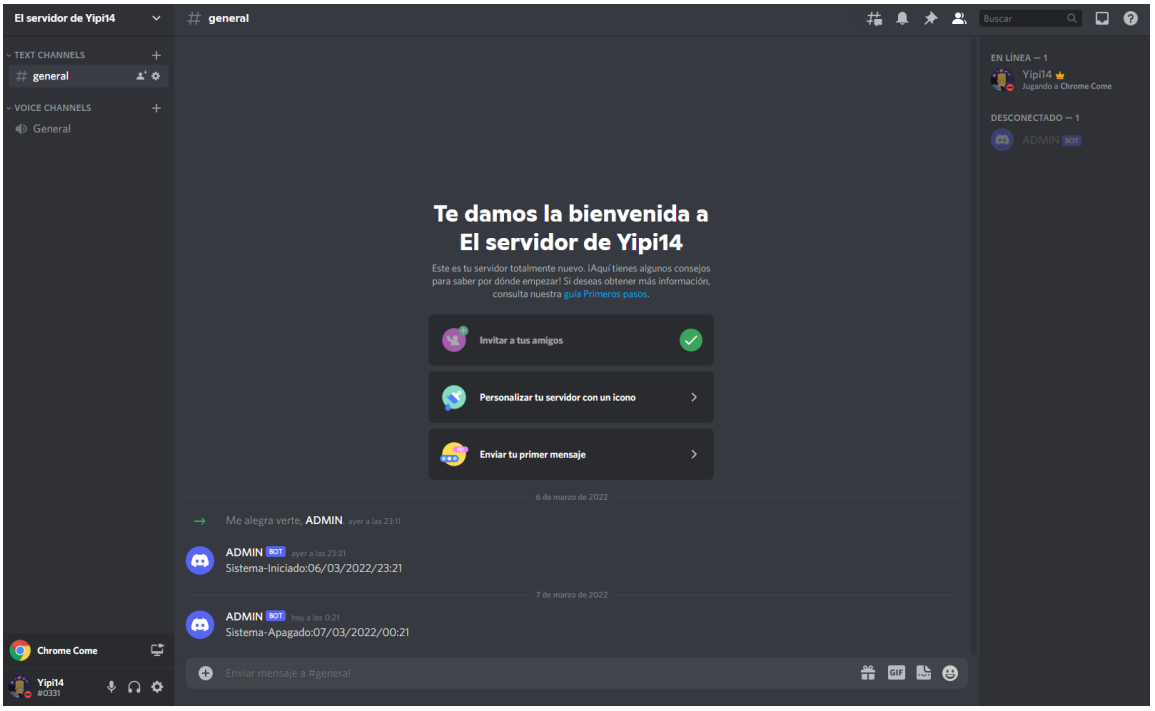
Es importante que el Bot se encuentre en dicho canal y tenga derecho a escribir mensajes.
Pantallazo de la terminal que se observa cómo se crea el symlink al ejecutar correctamente el instalador.

```
root@debian-yipi:/home/yipi/Documentos/BOT/StartStopDiscordMessage# ./installer.sh OTQ4NjI3NTA2ODA4OTAxNjkz.Yh-kNg.51y3L8643TNcV9vaF0B9i66bfcI 950153627262124147
Created symlink /etc/systemd/system/multi-user.target.wants/BotDiscord.service → /etc/systemd/system/BotDiscord.service.
```

Mediante **systemctl status BotDiscord.service** se observa como el servicio está activado.

```
root@debian-yipi:/home/yipi/Documentos/BOT/StartStopDiscordMessage# systemctl status BotDiscord.service
● BotDiscord.service - Envia un mensaje a un canal de un servidor de Discord mediante un bot cuando el sistema se apaga/reinicia
     Loaded: loaded (/etc/systemd/system/BotDiscord.service; enabled; vendor preset: enabled)
     Active: active (exited) since Sun 2022-03-06 23:21:06 CET; 47min ago
    Process: 4351 ExecStart=/usr/bin/BotDiscord.sh start (code=exited, status=0/SUCCESS)
   Main PID: 4351 (code=exited, status=0/SUCCESS)
        CPU: 98ms

mar 06 23:21:06 debian-yipi systemd[1]: Started Envia un mensaje a un canal de un servidor de Discord mediante un bot cuando el sistema se apaga/reinicia.
mar 06 23:21:07 debian-yipi BotDiscord.sh[4354]: 200
root@debian-yipi:/home/yipi/Documentos/BOT/StartStopDiscordMessage#
```

Cabe destacar que además nos proporciona el código de status del script de python. 200 ⟶ Correcto



Se adjunta unas capturas de pantalla del servidor de cómo quedarían los mensajes tras haber iniciado el sistema y tras apagarlo.

CONCLUSIONES

Hasta aquí el desarrollo de la aplicación. Tengo el conocimiento de que se puede realizar mediante node.js que provoca que aparezca conectado el Bot,mientras que en mi caso funciona desconectado. Pero me pareció interesante la idea de aplicar la API de Discord para llevar a cabo el proyecto.

Javier García Pechero