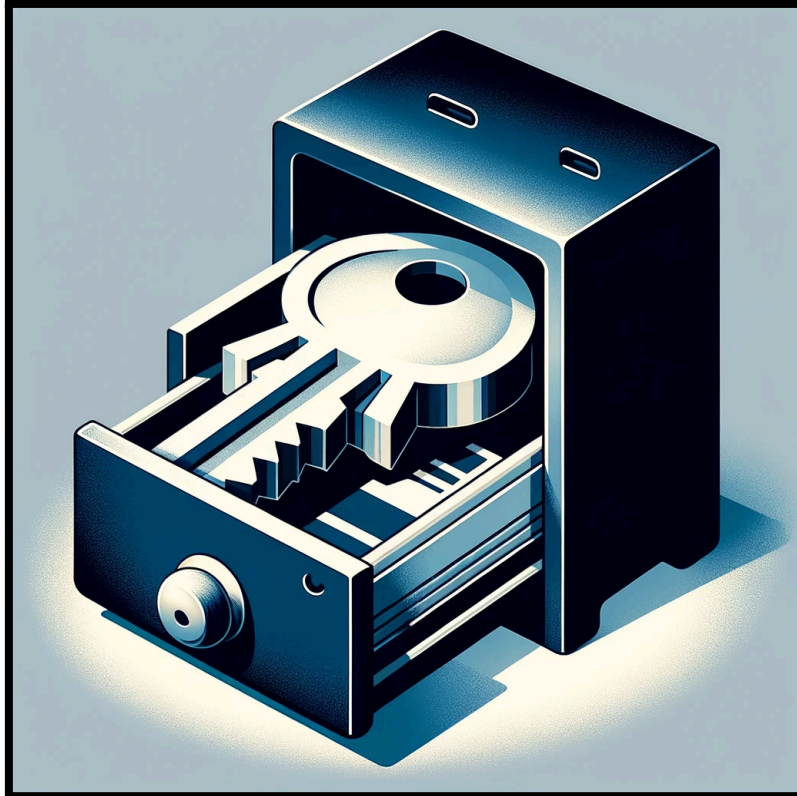


# *Generación de llaves SSH en GitHub*



Autor: Andreeeiii08

Plataforma: GitHub

Data: 10.05.2024

# Índice

<b>Índice</b>	<b>2</b>
<b>Generación de llaves SSH en GitHub</b>	<b>3</b>
1. Introducción	3
2. Proceso explicado	4
2.1. Generación de llave en local	4
2.1.1. Comando a meter en una terminal	4
2.1.2. Nombre del archivo	4
2.1.3. Contraseña	4
2.1.4. Resultado	5
2.2. Copiarla en el GitHub	5
2.2.1. Sacar la llave	5
2.2.2. Donde entrar	5
2.2.3. Añadir la llave	7
2.2.4. Copiar repositorio	8

# Generación de llaves SSH en GitHub

## 1. Introducción

Con ocasión del trabajo final del Módulo 2: Gestión de Bases de Datos unidad formativa 2, se ha creado un [repositorio](#) para llevar el proceso y poder compartir archivos con el equipo.

Ahora bien, nosotros en clase usamos un servidor MySQL dentro de una máquina virtual, más específicamente, un Debian 12. Para sincronizar los archivos del [trabajo](#), la solución más fácil es sincronizar GitHub con dicha máquina.

Una de las formas para hacer lo mencionado anteriormente, es con la generación de llaves SSH. Realmente es un proceso realmente sencillo y rápido, pero personalmente lo he hecho varias veces y nunca me acuerdo de memoria de los comandos necesarios.

Como solución a eso, he creado este documento explicando el rápido proceso de la generación de llaves SSH. Por razones de seguridad, los datos más sensibles serán censurados.

## 2. Proceso explicado

### 2.1. Generación de llave en local

#### *2.1.1. Comando a meter en una terminal*

Nos vamos a cualquier terminal del sistema operativo. Hay que poner `ssh-keygen -t rsa -b 4096`, donde el argumento `-t` especifica el tipo de llave que se generará. En este caso, estamos generando el tipo RSA, que es uno de los algoritmos de cifrado asimétricos más comunes. Por último, `-b` indica la longitud de bits en la llave a generar.



```
usuari@debian12: ~/GitHub
usuari@debian12:~/GitHub$ ssh-keygen -t rsa -b 4096
```

#### *2.1.2. Nombre del archivo*

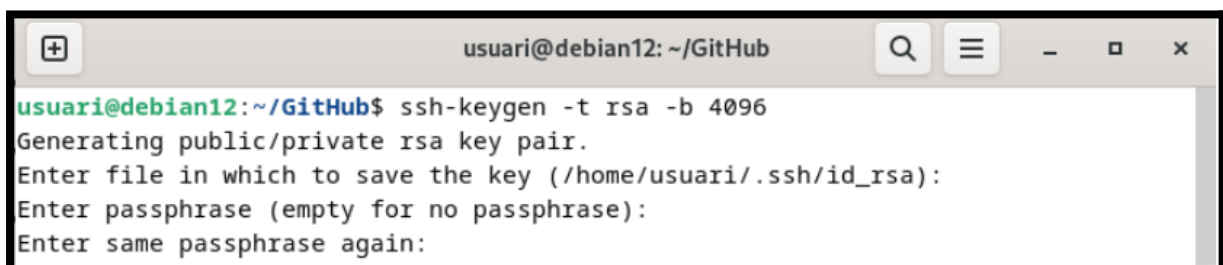
Por defecto, la llave SSH se queda guardada en una carpeta oculta en la ruta: `/.ssh/id_rsa`, donde `id_rsa` es el archivo con los datos de nuestra llave.



```
usuari@debian12: ~/GitHub
usuari@debian12:~/GitHub$ ssh-keygen -t rsa -b 4096
Generating public/private rsa key pair.
Enter file in which to save the key (/home/usuari/.ssh/id_rsa):
```

#### *2.1.3. Contraseña*

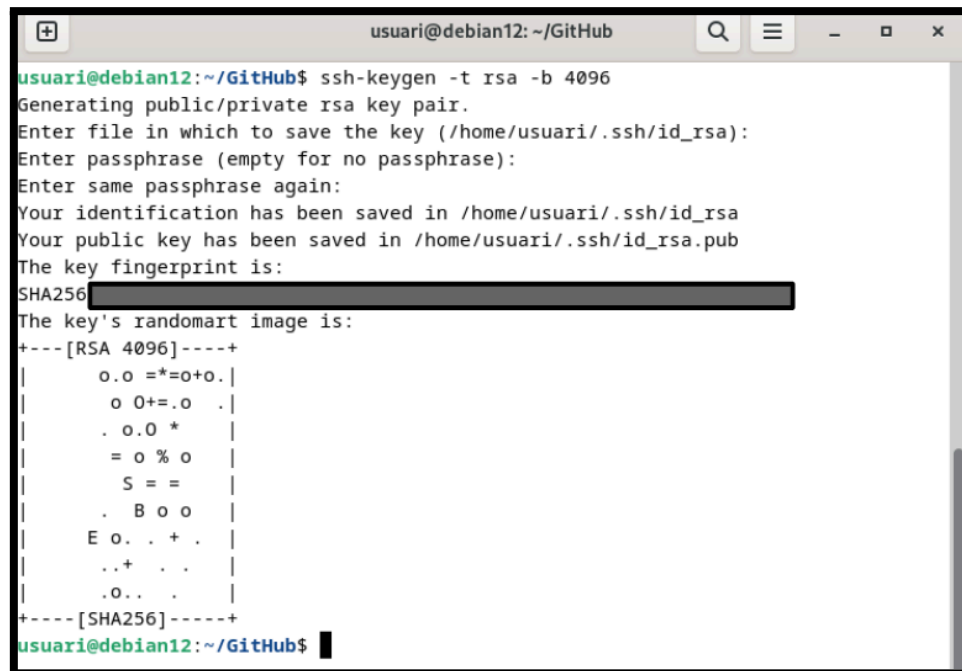
Por razones de seguridad, podemos meter una contraseña adicional que nos pedirán cada vez que usemos esta llave. En caso de no querer ninguna, simplemente le podemos dar al `enter`.



```
usuari@debian12: ~/GitHub
usuari@debian12:~/GitHub$ ssh-keygen -t rsa -b 4096
Generating public/private rsa key pair.
Enter file in which to save the key (/home/usuari/.ssh/id_rsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
```

### 2.1.4. Resultado

Una vez acabado este proceso, se nos confirmará que se ha creado la llave. En caso necesario, también nos indica dónde se ha guardado.

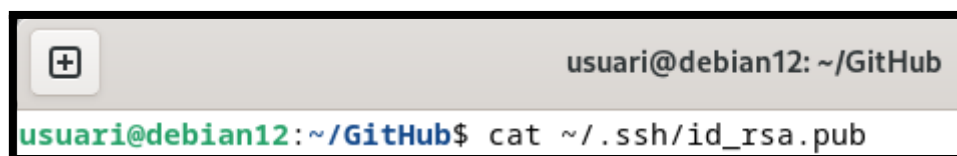


```
usuari@debian12: ~/GitHub
usuari@debian12:~/GitHub$ ssh-keygen -t rsa -b 4096
Generating public/private rsa key pair.
Enter file in which to save the key (/home/usuari/.ssh/id_rsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/usuari/.ssh/id_rsa
Your public key has been saved in /home/usuari/.ssh/id_rsa.pub
The key fingerprint is:
SHA256: [REDACTED]
The key's randomart image is:
+---[RSA 4096]---+
|  o.o  =*=o+o. |
|  o 0+=.o  . |
|  . o.O * |
|  = o % o |
|  S = = |
|  . B o o |
|  E o. . + . |
|  ..+ . . |
|  .O.. . |
+-----[SHA256]-----+
usuari@debian12:~/GitHub$
```

## 2.2. Copiarla en el GitHub

### 2.2.1. Sacar la llave

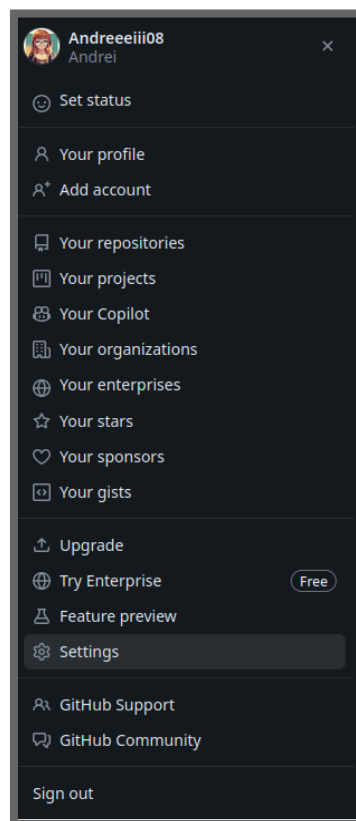
A partir de ahora, podemos comprobar con un “cat” (en caso de sistemas Linux) sobre el archivo creado. Hay que mencionar que nos interesa meter la llave pública dentro del GitHub, así que copiaremos todo el contenido del archivo con el formato “.pub”.



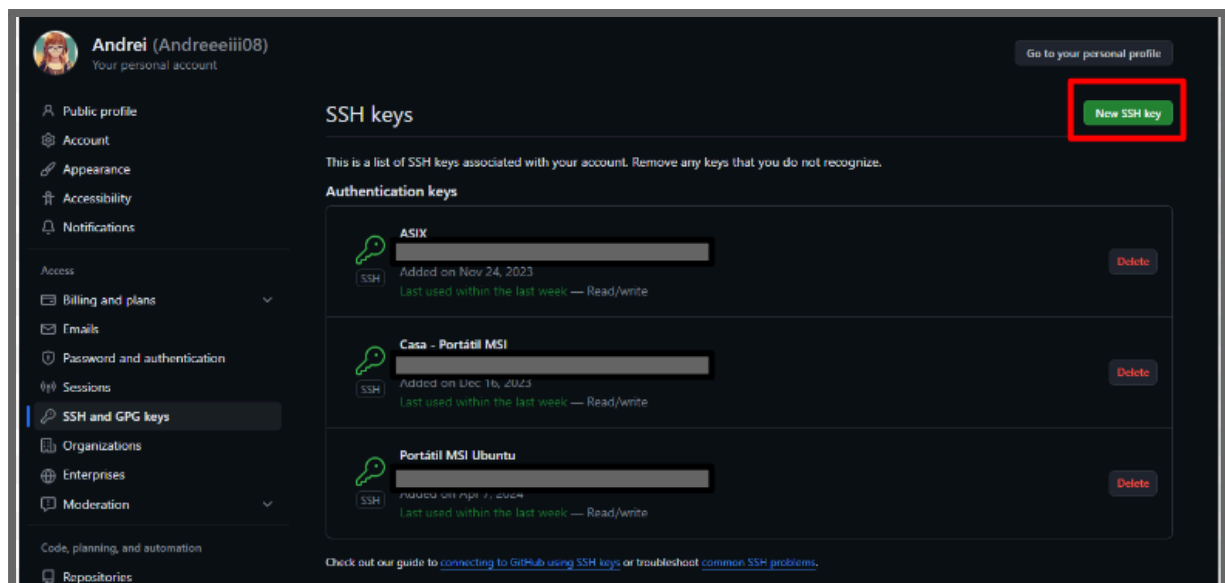
```
usuari@debian12: ~/GitHub
usuari@debian12:~/GitHub$ cat ~/.ssh/id_rsa.pub
```

### 2.2.2. Donde entrar

Desde el menú lateral, nos vamos a ajustes.

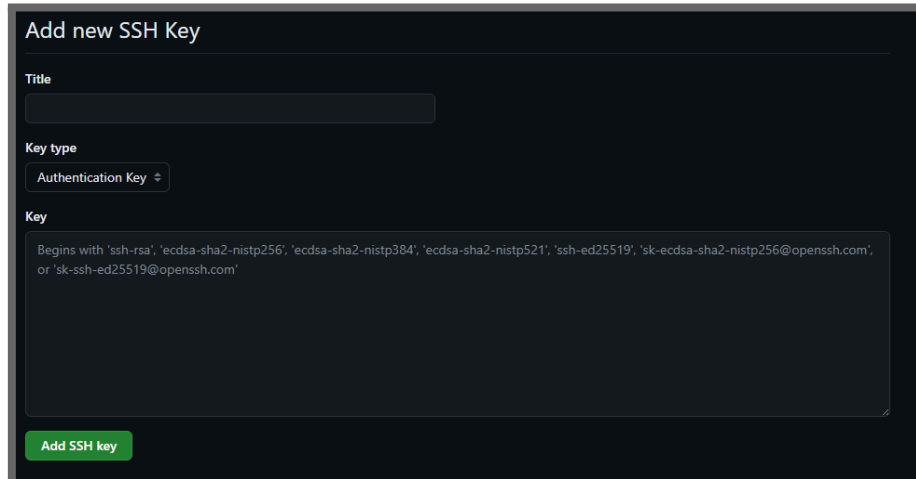


Nos vamos al apartado “SSH and GPG keys” y nos salen todas las llaves SSH que tenemos ya registradas, en caso de tener alguna. Para crear una, nos vamos a “New SSH key”.



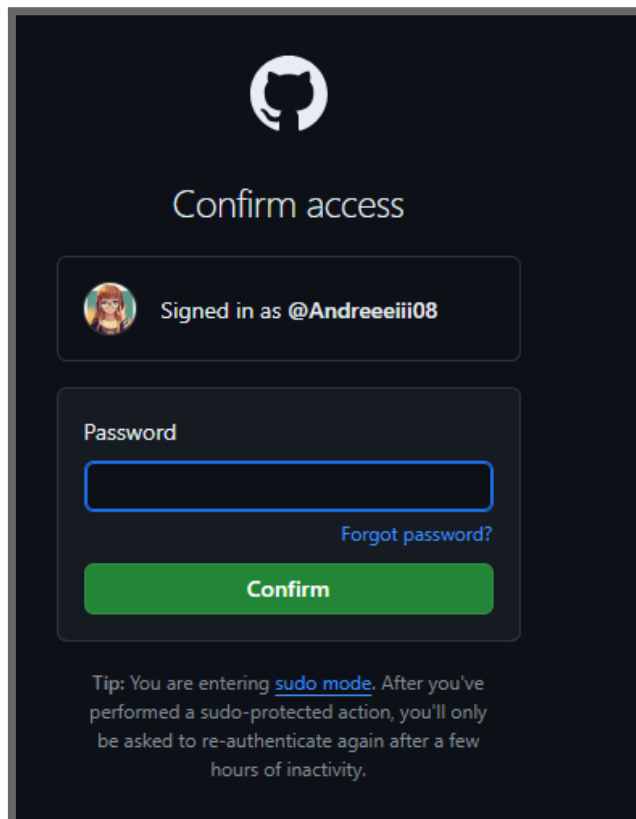
### 2.2.3. Añadir la llave

En esta pantalla le ponemos el nombre que queramos a la llave SSH, mi consejo es que sea un nombre distintivo y fácil de identificar. Donde “key” pegamos lo que ya hemos copiado anteriormente en el apartado “2.1.3.”.



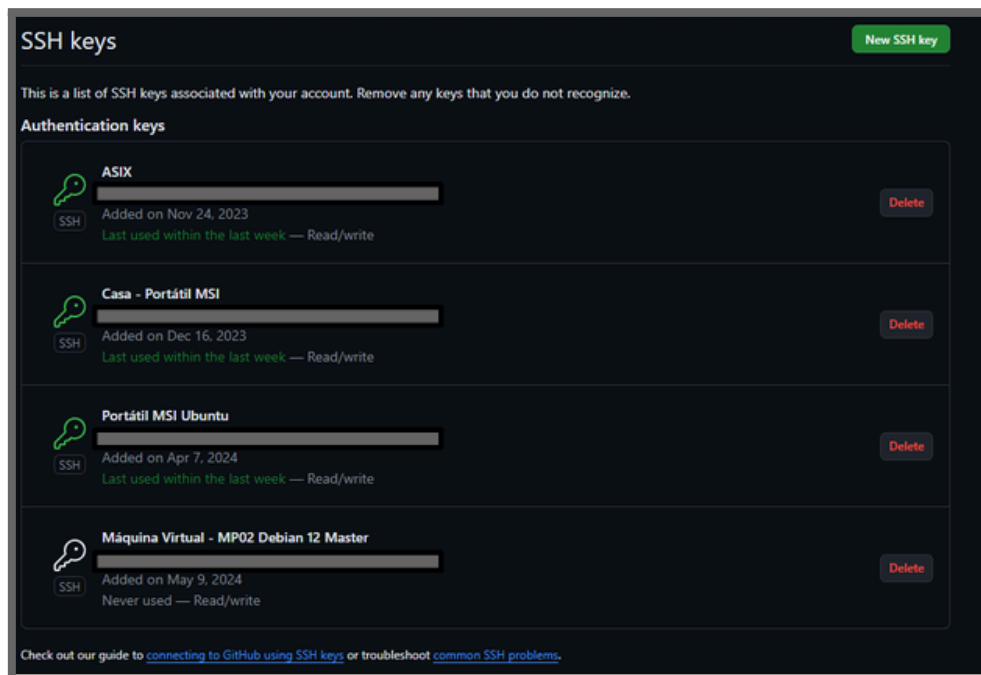
The screenshot shows the 'Add new SSH Key' interface. It has a title input field, a 'Key type' dropdown menu set to 'Authentication Key', and a 'Key' text area. Below the text area, there is a green 'Add SSH key' button. A help text box indicates that the key should begin with 'ssh-rsa', 'ecdsa-sha2-nistp256', 'ecdsa-sha2-nistp384', 'ecdsa-sha2-nistp521', 'ssh-ed25519', 'sk-ecdsa-sha2-nistp256@openssh.com', or 'sk-ssh-ed25519@openssh.com'.

Una vez hecho esto, nos pedirán la contraseña de la cuenta por motivos de seguridad.



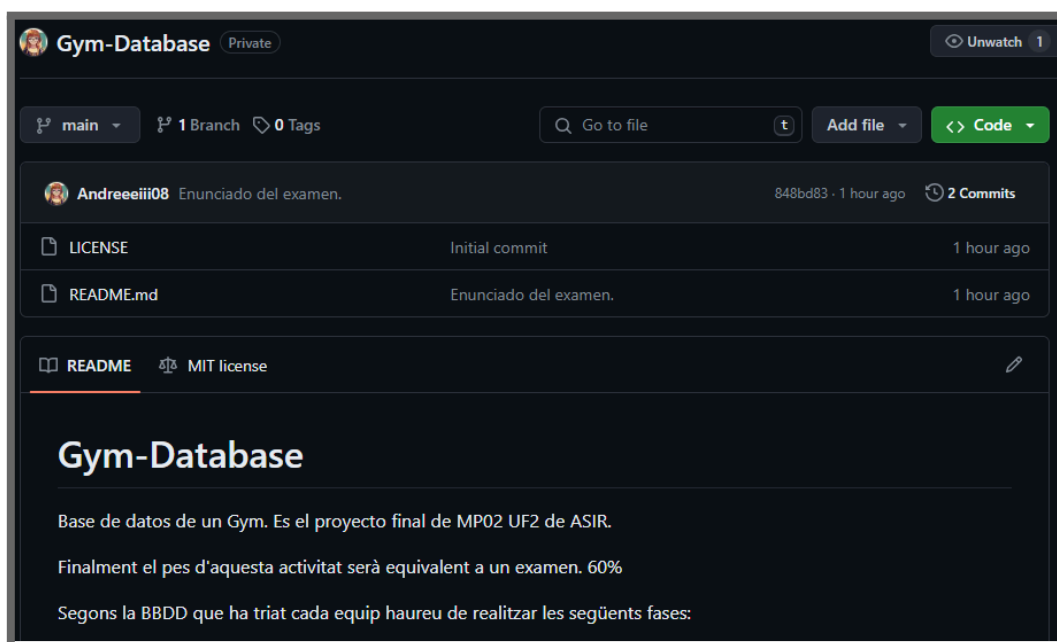
The screenshot shows the 'Confirm access' screen. At the top is the GitHub logo. Below it, the text 'Confirm access' is displayed. A box shows the user is signed in as '@Andreeeiii08' with a profile picture. Below this is a 'Password' input field with a 'Forgot password?' link. A green 'Confirm' button is at the bottom. A tip at the bottom states: 'Tip: You are entering sudo mode. After you've performed a sudo-protected action, you'll only be asked to re-authenticate again after a few hours of inactivity.'

Ahora en el menú anterior, nos saldrá la nueva llave SSH.



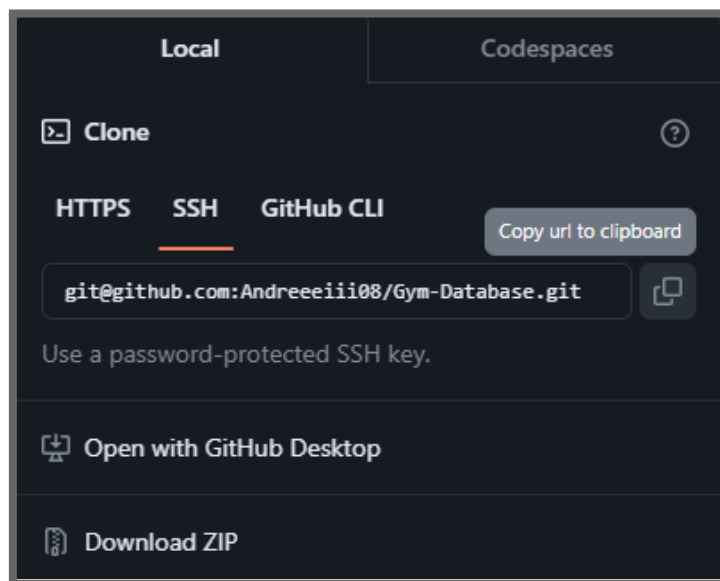
#### 2.2.4. Copiar repositorio

Ya podemos copiar cualquier repositorio. Lo podemos hacer desde el botón verde “code”.



Aquí seleccionamos la opción de SSH y copiamos el enlace.

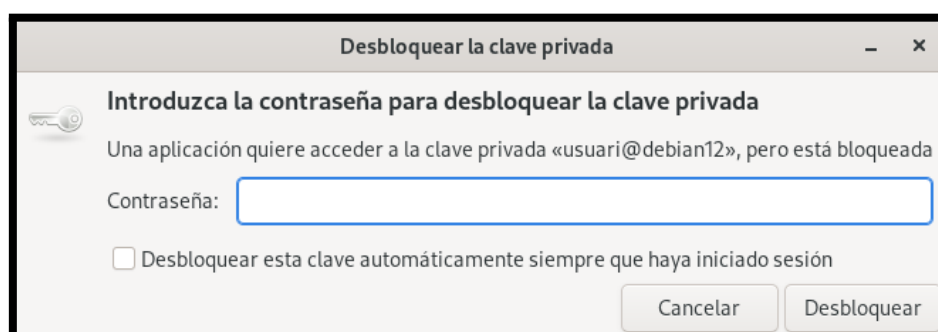




En la carpeta que queremos clonar el repositorio ponemos “git clone” y el enlace. Si es la primera vez que lo hacemos, tendremos que poner “yes” para establecer la conexión.



Por seguridad, en caso de haber establecido antes una contraseña, ahora la tenemos que poner. Podemos marcar el cuadrado de abajo para solo tener que poner la contraseña una vez por sesión.



Ya podemos comprobar que nuestro repositorio está descargado en nuestro equipo.

```
usuari@debian12: ~/GitHub
git clone git@github.com:Andreeeiii08/Gym-Database.git
Clonando en 'Gym-Database'...
The authenticity of host 'github.com (140.82.121.4)' can't be established.
ED25519 key fingerprint is SHA256: [REDACTED].
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'github.com' (ED25519) to the list of known hosts.
remote: Enumerating objects: 7, done.
remote: Counting objects: 100% (7/7), done.
remote: Compressing objects: 100% (7/7), done.
remote: Total 7 (delta 0), reused 0 (delta 0), pack-reused 0
Recibiendo objetos: 100% (7/7), listo.
usuari@debian12: ~/GitHub$ ls
Gym-Database
```

Si miramos las llaves SSH, podemos ver que ya se ha usado nuestra llave que acabamos de crear, como un indicador más que funciona correctamente.

