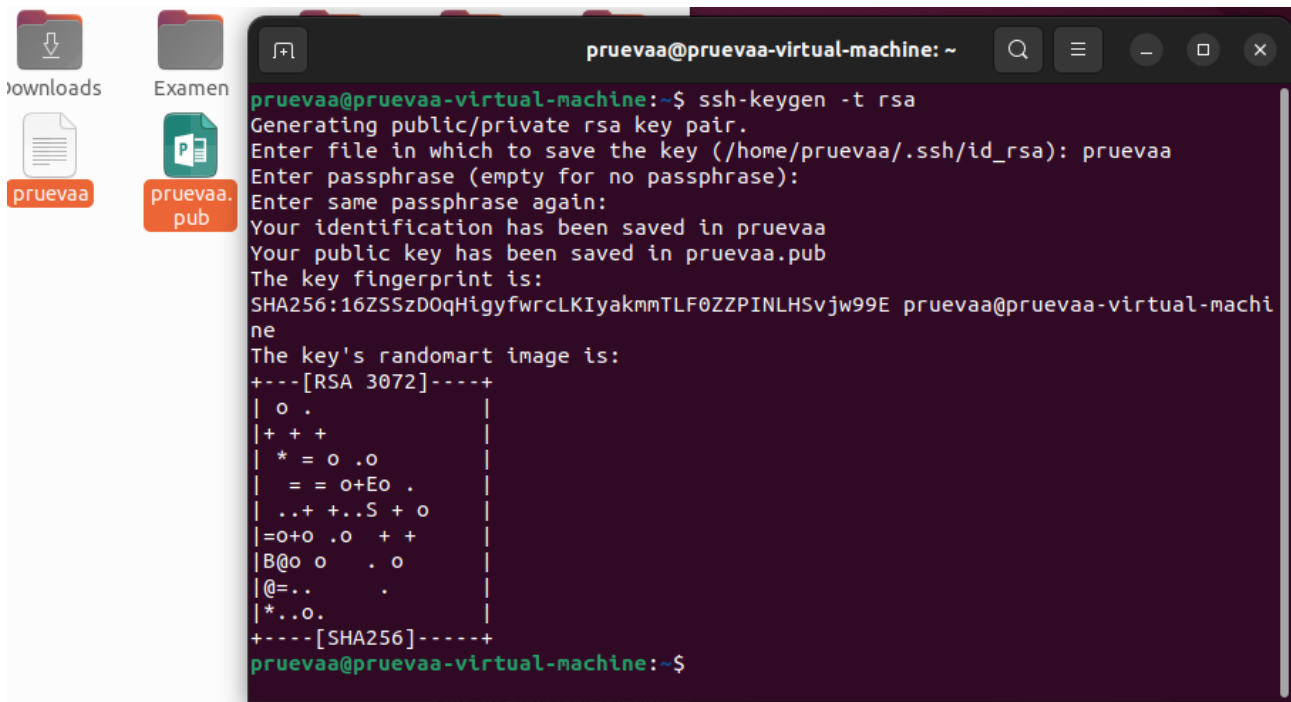


EXAMEN ASO

Tarea 1: Crea una clave SSH en tu sistema operativo y aporta tu clave publica, es decir, .pub

`ssh-keygen -t rsa`



Lo que hemos hecho es crear una clave ssh.

Los pasos :

Poner `ssh-keygen -t rsa`

poner una contraseña

crear 2 archivos uno contraseña publica y otra privada(en la imagen se ven los archivos)

Y listo

Tarea 2: Crea un programa con el nombre “tarea2.py” en Python que muestre por pantalla el porcentaje de espacio ocupado en cada una de las particiones de tu sistema, de forma que se muestre tal que así: /dev/sda1 78,9% /dev/sdb1 18,5%

```
import psutil
```

```
# Obtener todas las particiones del sistema
```

```
particiones = psutil.disk_partitions()
```

```
# Iterar sobre cada partición
```

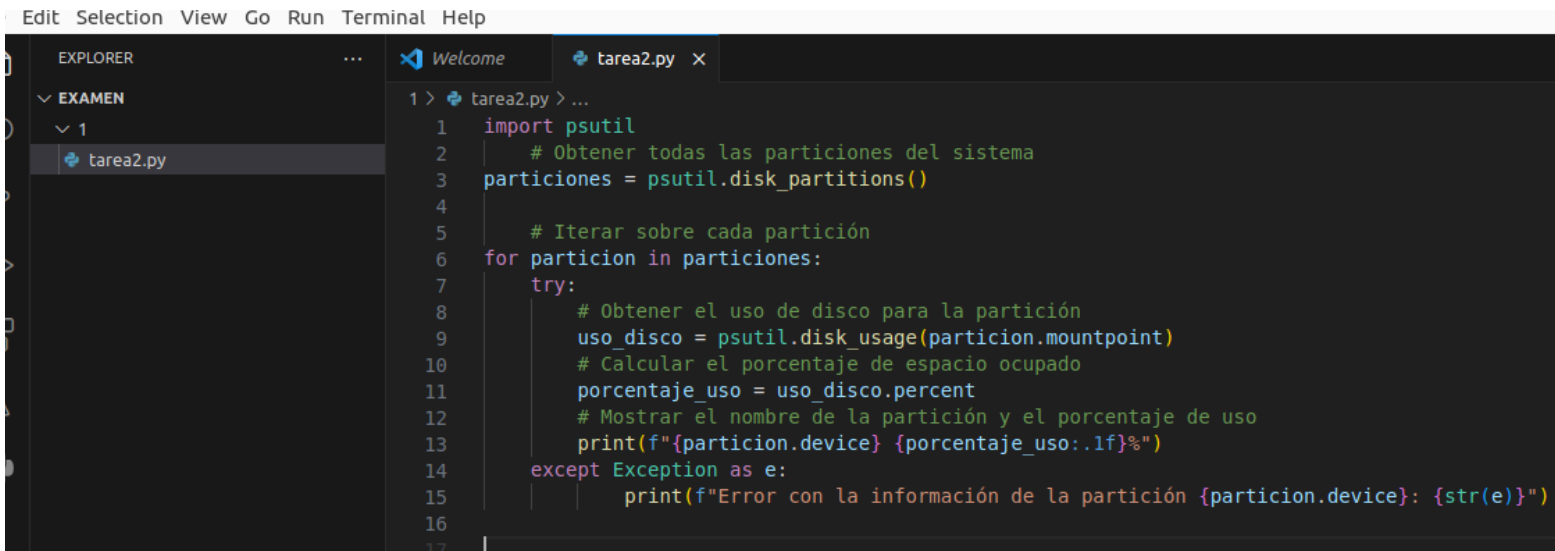
```
for particion in particiones:
```

```
try:
```

```
# Obtener el uso de disco para la partición
```

```
uso_disco = psutil.disk_usage(particion.mountpoint)
```

```
# Calcular el porcentaje de espacio ocupado
porcentaje_uso = uso_disco.percent
# Mostrar el nombre de la partición y el porcentaje de uso
print(f"{particion.device} {porcentaje_uso:.1f}%")
except Exception as e:
    print(f"Error con la información de la partición {particion.device}: {str(e)}")
```

A screenshot of a code editor interface. The top bar shows 'Edit Selection View Go Run Terminal Help'. The left sidebar has an 'EXPLORER' panel with a tree view showing 'EXAMEN' and '1' under it, with 'tarea2.py' selected. The main editor area shows the code for 'tarea2.py' with line numbers 1 through 17. The code imports 'psutil', gets disk partitions, iterates over them, gets disk usage for each, calculates the percentage, and prints it. It also includes an exception handler to print an error message if something goes wrong.

```
1 > tarea2.py > ...
1 import psutil
2     # Obtener todas las particiones del sistema
3 particiones = psutil.disk_partitions()
4
5     # Iterar sobre cada partición
6 for particion in particiones:
7     try:
8         # Obtener el uso de disco para la partición
9         uso_disco = psutil.disk_usage(particion.mountpoint)
10        # Calcular el porcentaje de espacio ocupado
11        porcentaje_uso = uso_disco.percent
12        # Mostrar el nombre de la partición y el porcentaje de uso
13        print(f"{particion.device} {porcentaje_uso:.1f}%")
14    except Exception as e:
15        print(f"Error con la información de la partición {particion.device}: {str(e)}")
16
17
```

Lo que hace es :

importamos la librería
creamos una variable que muestra las particiones de tus sistema
creamos un bucle con una sentencia por cada partición.
En el sacamos todo el uso del disco
lo anterior lo pasa porcentaje
imprime lo las variables

y por si da algún erro mostramos un mensaje.

Tarea 3: Implementa un programa en el fichero llamado “tarea3.py” que ejecute un bucle 5 veces donde creará una carpeta con el nombre folder1, folder2 ...folder5, reando dentro de ellos 10 ficheros con el siguiente nombre y siguiente contenido: nombre fichero: fichero1.txt contenido: Este es el contenido del fichero 1

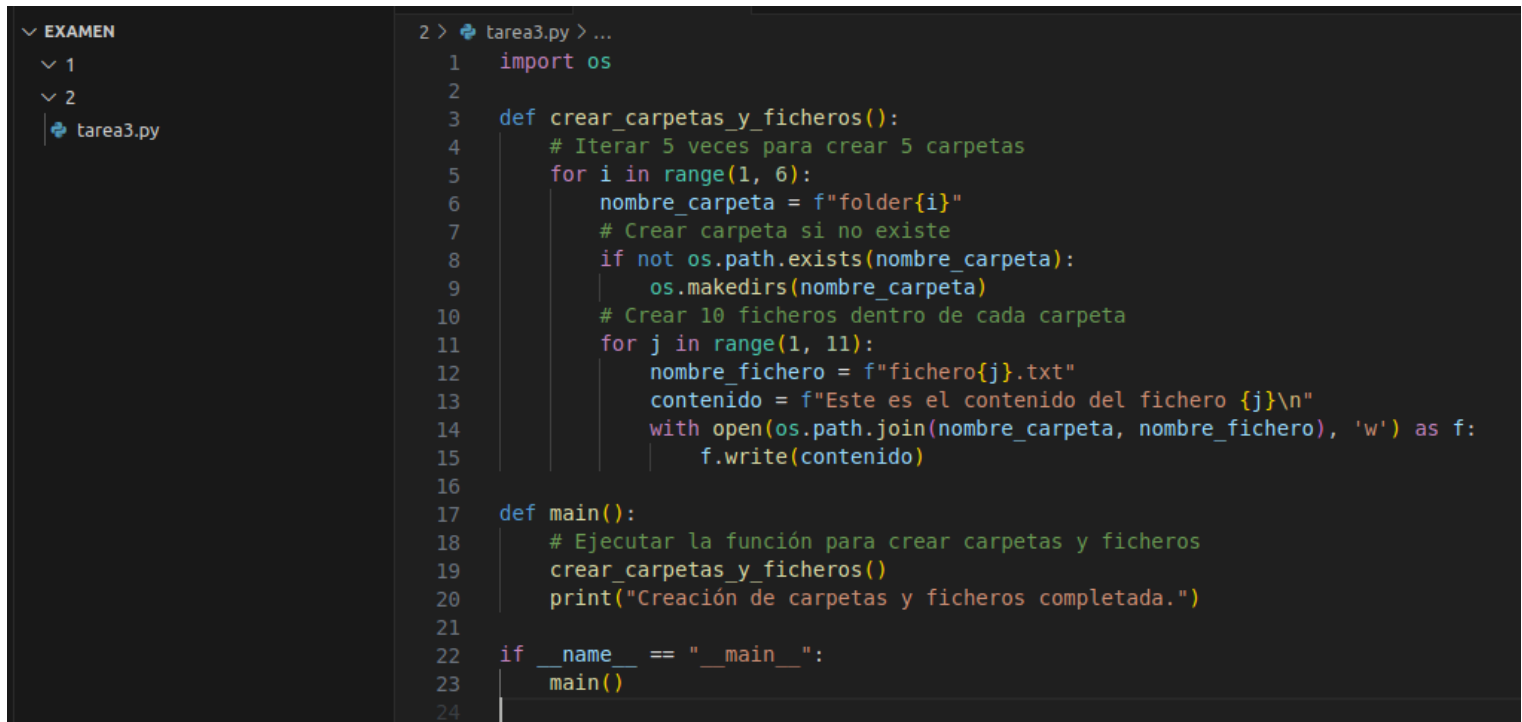
```
import os
```

```
def crear_carpetas_y_ficheros():
    # Iterar 5 veces para crear 5 carpetas
    for i in range(1, 6):
        nombre_carpeta = f"folder{i}"
        # Crear carpeta si no existe
```

```
if not os.path.exists(nombre_carpeta):
    os.makedirs(nombre_carpeta)
# Crear 10 ficheros dentro de cada carpeta
for j in range(1, 11):
    nombre_fichero = f"fichero{j}.txt"
    contenido = f"Este es el contenido del fichero {j}\n"
    with open(os.path.join(nombre_carpeta, nombre_fichero), 'w') as f:
        f.write(contenido)

def main():
    # Ejecutar la función para crear carpetas y ficheros
    crear_carpetas_y_ficheros()
    print("Creación de carpetas y ficheros completada.")

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



```
2 > tarea3.py > ...
1 import os
2
3 def crear_carpetas_y_ficheros():
4     # Iterar 5 veces para crear 5 carpetas
5     for i in range(1, 6):
6         nombre_carpeta = f"folder{i}"
7         # Crear carpeta si no existe
8         if not os.path.exists(nombre_carpeta):
9             os.makedirs(nombre_carpeta)
10        # Crear 10 ficheros dentro de cada carpeta
11        for j in range(1, 11):
12            nombre_fichero = f"fichero{j}.txt"
13            contenido = f"Este es el contenido del fichero {j}\n"
14            with open(os.path.join(nombre_carpeta, nombre_fichero), 'w') as f:
15                f.write(contenido)
16
17 def main():
18     # Ejecutar la función para crear carpetas y ficheros
19     crear_carpetas_y_ficheros()
20     print("Creación de carpetas y ficheros completada.")
21
22 if __name__ == "__main__":
23     main()
24
```

importamos la librería os
creamos una función
con un bucle de 5 veces
creamos una variable para crear carpetas 1,2,3,...
comprobamos si existe las carpetas
y las creamos
creamos otro bucle que se repite 10 veces
y repetimos lo mismo que las carpetas pero con archivos

Tarea 4: Desarrolla el programa con el nombre “tarea4.py” donde se analice el espacio disponible en la partición correspondiente a la raíz(“/”),sacando un mensaje de logging mediante la librería logging en el fichero /home//logs/espacio.log Si el espacio ocupado es mayor que 80% se usara un mensaje de error. Si el espacio ocupado es mayor que 60% y menor que 80% se usará un mensaje de warning Si el espacio ocupado es mayor que 0% y menor que 60% se usará un mensaje de info

```
import os
import logging
import psutil

# Directorio de logs
directorio_logs = '/home/pruevaa/logs/'

# Verificar si el directorio de logs existe, si no existe, crearlo
if not os.path.exists(directorio_logs):
    os.makedirs(directorio_logs)

def configurar_logger():
    # Crear el logger
    logger = logging.getLogger("espacio_logger")
    logger.setLevel(logging.DEBUG)

    # Crear el formateador
    formatter = logging.Formatter('%(asctime)s - %(levelname)s - %(message)s')

    # Crear el manejador para escribir en el archivo
    archivo_log = '/home/pruevaa/logs/espacio.log'
    file_handler = logging.FileHandler(archivo_log)
    file_handler.setLevel(logging.DEBUG)
    file_handler.setFormatter(formatter)

    # Agregar el manejador al logger
    logger.addHandler(file_handler)

    return logger

def analizar_espacio():
    # Obtener el espacio en disco para la raíz "/"
    espacio_raiz = psutil.disk_usage('/')
    espacio_ocupado_porcentaje = espacio_raiz.percent

    # Configurar el logger
    logger = configurar_logger()

    # Determinar el nivel de mensaje según el espacio ocupado
    if espacio_ocupado_porcentaje > 80:
```

```
        logger.error(f"Espacio ocupado en la raíz (/) es mayor que 80%:  
{espacio_ocupado_porcentaje}%")  
    elif espacio_ocupado_porcentaje > 60:  
        logger.warning(f"Espacio ocupado en la raíz (/) es mayor que 60% y menor que 80%:  
{espacio_ocupado_porcentaje}%")  
    elif espacio_ocupado_porcentaje > 0:  
        logger.info(f"Espacio ocupado en la raíz (/) es mayor que 0% y menor que 60%:  
{espacio_ocupado_porcentaje}%")  
  
def main():  
    analizar_espacio()  
  
if __name__ == "__main__":  
    main()
```

```

3 > tarea4.py > ...
1 import os
2 import logging
3 import psutil
4
5 # Directorio de logs
6 directorio_logs = '/home/pruevaa/logs/'
7
8 # Verificar si el directorio de logs existe, si no existe, crearlo
9 if not os.path.exists(directorio_logs):
10     os.makedirs(directorio_logs)
11
12
13 def configurar_logger():
14     # Crear el logger
15     logger = logging.getLogger("espacio_logger")
16     logger.setLevel(logging.DEBUG)
17
18     # Crear el formateador
19     formatter = logging.Formatter('%(asctime)s - %(levelname)s - %(message)s')
20
21     # Crear el manejador para escribir en el archivo
22     archivo_log = '/home/pruevaa/logs/espacio.log'
23     file_handler = logging.FileHandler(archivo_log)
24     file_handler.setLevel(logging.DEBUG)
25     file_handler.setFormatter(formatter)
26
27     # Agregar el manejador al logger
28     logger.addHandler(file_handler)
29
30     return logger
31
32 def analizar_espacio():
33     # Obtener el espacio en disco para la raíz "/"
34     espacio_raiz = psutil.disk_usage('/')
35     espacio_ocupado_porcentaje = espacio_raiz.percent
36
37     # Configurar el logger
38     logger = configurar_logger()
39
40     # Determinar el nivel de mensaje según el espacio ocupado
41     if espacio_ocupado_porcentaje > 80:
42         logger.error(f"Espacio ocupado en la raíz (/) es mayor que 80%: {espacio_ocupado_porcentaje}%")
43     elif espacio_ocupado_porcentaje > 60:
44         logger.warning(f"Espacio ocupado en la raíz (/) es mayor que 60% y menor que 80%: {espacio_ocupado_porcentaje}%")
45     elif espacio_ocupado_porcentaje > 0:
46         logger.info(f"Espacio ocupado en la raíz (/) es mayor que 0% y menor que 60%: {espacio_ocupado_porcentaje}%")
47
48 def main():
49     analizar_espacio()
50
51 if __name__ == "__main__":
52     main()
53

```

importamos las librerías os, logging, psutil.

Creamos una variable donde guardarlos la direccional de logs.

Comprobamos que exista el directorio donde guardar los logs sino se crea.

Hacemos una función que creamos los logs y permitimos que nivel de logs salen.

Creamos el formato como va a salir logs.

Creamos una variable donde indicamos la ruta al archivo logs.

Indicamos parámetros.

Creamos una función donde sacamos parámetros de disco

sacamos el espacio en porcentaje de disco como en el archivo anterior.

Hacemos condicionales con el porcentaje que tiene en disco y dependiendo lo que tiene saldara un “error”

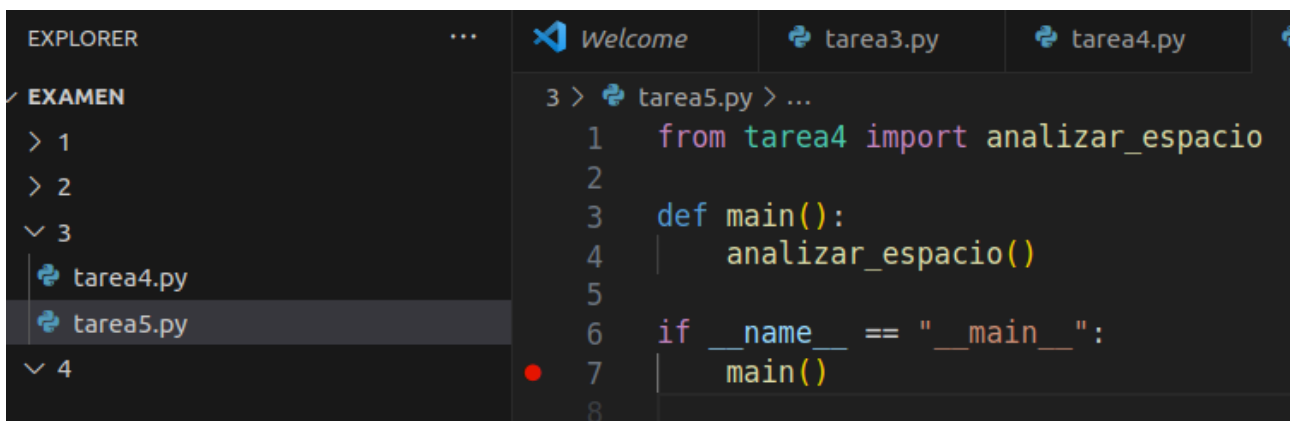
llamamos a las funciones para que funcione.

Tarea 5: define una función dentro del fichero “tarea4.py” y copia el código que creaste en la tarea 4. A continuación, crea el fichero “tarea5.py” e importa el fichero “tarea4.py” y llama a la función definida en él.

```
from tarea4 import analizar_espacio
```

```
def main():  
    analizar_espacio()
```

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

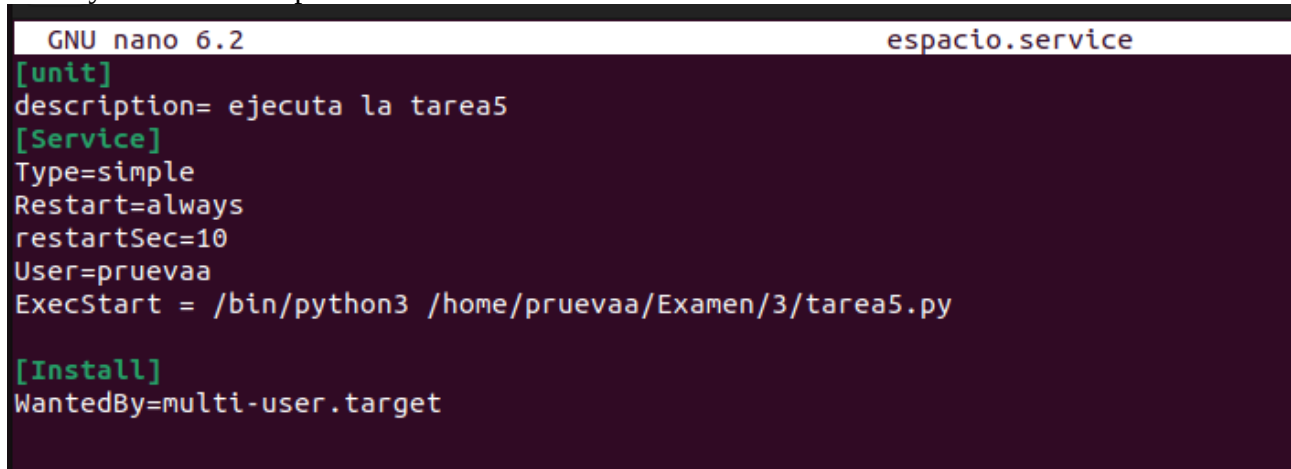


importamos el ejercicio anterior y creamos una función que llame a las funciones del archivo anterior

Tarea 6: Crea un servicio llamado “espacio.service” que llame al fichero creado en la “tarea5.py” cada 10 segundos.

```
cd /etc/systemd/system/  
sudo nano espacio.service  
-----  
[unit]  
description= ejecuta la tarea5  
[Service]  
Type=simple  
Restart=always  
restartSec=10  
User=pruevaa  
ExecStart = /bin/python3 /home/pruevaa/Examen/3/tarea5.py  
  
[Install]  
WantedBy=multi-user.target  
-----  
sudo systemctl daemon-reload
```

```
sudo systemctl restart espacio.service
sudo systemctl status espacio.service
sudo systemctl start espacio.service
```



```
GNU nano 6.2 espacio.service
[Unit]
description= ejecuta la tarea5
[Service]
Type=simple
Restart=always
restartSec=10
User=pruevaa
ExecStart = /bin/python3 /home/pruevaa/Examen/3/tarea5.py

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

Vamos a la carpeta `cd /etc/systemd/system/`
creamos un archivo llamado `espacio.service`

escribimos una descripción
escribimos que el servicio sera simple
se secreteara siempre
se reiniciara cada 10 segundos
indicamos con usuario aremos dicho proceso
indicamos con que se ejecuta y que archivo
Instalación de la unidad y en qué punto del sistema debe ser activada

avisamos que los servicios han sido modificados
reiniciamos nuestro servicio (archivo)
mostramos que funcionan nuestro servicio (teóricamente)
Iniciamos el servicio.

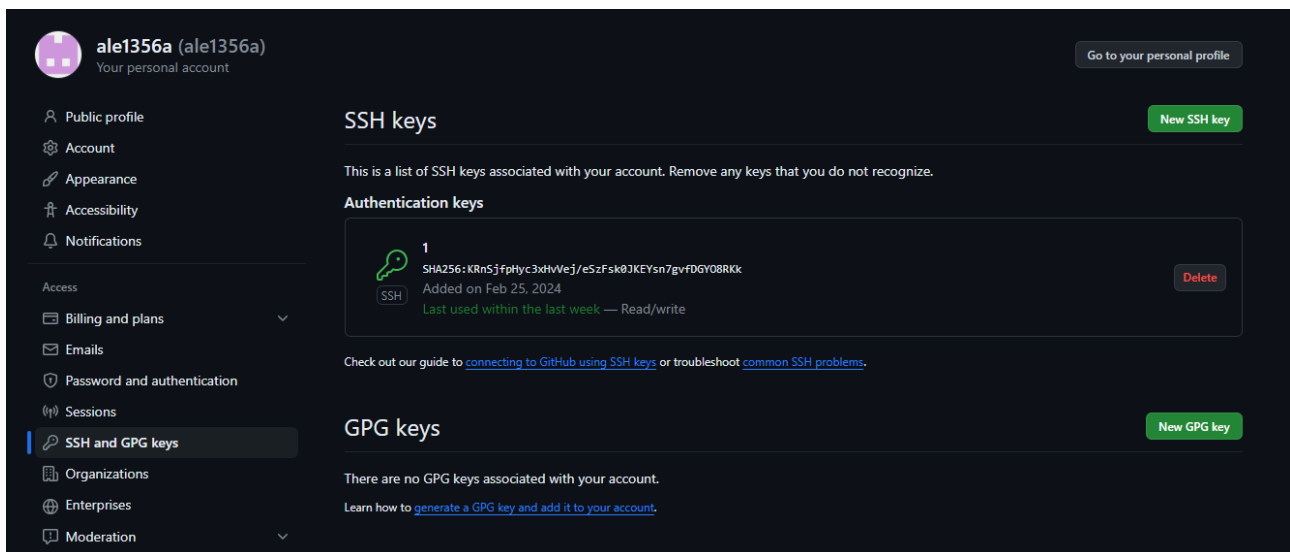
Tarea 7: Crea un repositorio en github y otro en bitbucket y añade allí tu clave publica SSH, así como la mia: `ssh-ed25519 AAAAC3NzaC1lZDI1NTE5AAAAIOP+jPVj13h6gmYJbflcZlIpD7L3hrHD+Aeq75+DVYx 5 ies.fernandosanchez@gmail.com` Sube en estos dos repositorios todas las tareas y documenta la creación e incorporación de las claves publicas en un documento PDF, así como los comandos ejecutados para las subidas de los ficheros a cada uno de los documentos. Sube el documento PDF a continuación al AulaVirtual de la asignatura. Añade tus dos repositorios a la propia entrega del AulaVirtual.


```
pruevaa@pruevaa-virtual-machine:~$ cd /home/pruevaa/.ssh/
```

```
ssh-keygen -t ed25519 -C "alejandromsagra@gmail.com"
```

```
pruevaa@pruevaa-virtual-machine:~/.ssh$ ls  
id_ed25519 id_ed25519.pub
```

agregas una nueva (.pub)



```
mkdir /home/prueva/EXAMEN_ASO  
cd /home/prueva/EXAMEN_ASO
```

```
echo "# fvhjm" >> README.md  
git init  
git add .  
git commit -m "first commit"  
git branch -M main  
git remote add origin git@github.com:ale1356a/EXAMEN_ASO.git  
git push origin main
```

listo