

T.P. 1 - Creación repositorio en Git y comandos basicos


- Se debe de crear un repositorio **personal** que se llame **TP_AySO**

Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere? [Import a repository.](#)

Required fields are marked with an asterisk (*).



Owner * Repository name *

 upszot / TP_AySO

TP_AySO is available.

Great repository names are short and memorable. Need inspiration? How about [turbo-engine](#) ?

Description (optional)

- ☒  **Public**
Anyone on the internet can see this repository. You choose who can commit.
- ☐  **Private**
You choose who can see and commit to this repository.

Initialize this repository with:

- ☒ **Add a README file**
This is where you can write a long description for your project. [Learn more about READMEs.](#)

Add .gitignore

.gitignore template: **None**

Choose which files not to track from a list of templates. [Learn more about ignoring files.](#)

Choose a license

License: **None**

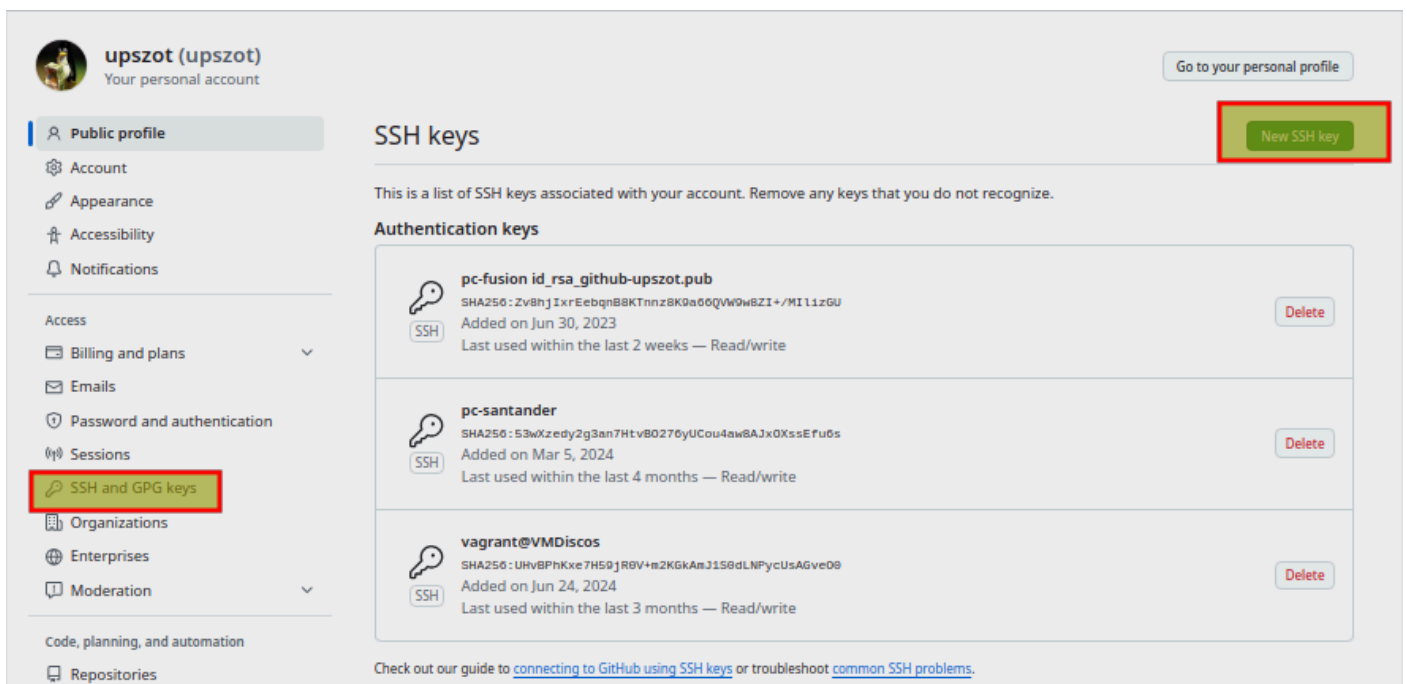
A license tells others what they can and can't do with your code. [Learn more about licenses.](#)

This will set  **master** as the default branch. Change the default name in your [settings](#).

 You are creating a public repository in your personal account.

Create repository

- Desde la VM ejecutar el comando “**ssh-keygen**” y ejecutar “[**Enter**]” en todos los parámetros que pida.
De esta forma se generará un juego de claves de ssh públicas y privadas en el directorio home del usuario “\$HOME” en la carpeta oculta .ssh
- Ver el contenido del archivo (dentro del home del usuario, en la carpeta oculta “.ssh” el archivo con la extensión .pub) donde se encuentra la clave pública.
cat \$HOME/.ssh/*.pub
- copiar la misma en “github.com” dentro del perfil del usuario..



- En la VM: crear una carpeta “repogit” dentro del home del usuario, para ello ejecutar los siguientes comandos:

Comando	Descripción
cd	Con el comando cd (change dir) sin pasarle parámetros, nos aseguramos de cambiarnos al directorio home del usuario con el que estamos logeados.
mkdir repogit	creamos el directorio “repogit”

- Clonamos el repositorio dentro de la carpeta “\$HOME/repogit/” y nos posicionamos dentro del repositorio.

cd repogit	nos movemos dentro del directorio repogit
git clone <URL>	<p>clonamos el repositorio.. Para lo cual hay que reemplazar <URL> por la de sus repositorios. Para eso vamos al boton Verde, tildamos “SSH” y copiamos la dirección.</p> 
cd T[Tab]	<p>Nos cambiamos al directorio del repositorio.. Para lo cual.. podemos ingresar la 1er letra del nombre de la carpeta.. y luego presionar la tecla</p> 

- Una vez dentro de la carpeta del repositorio.. podemos validar el Path donde estamos parados, y listar el contenido de la carpeta, ejecutando los siguientes comandos

pwd	pwd muestra el path donde estamos parados
ls -l	<p>Muestra los archivos / directorios en forma detallada</p> <pre> vagrant@VMPruebas:~/repogit\$ cd TP_AySO/ vagrant@VMPruebas:~/repogit/TP_AySO\$ pwd /home/vagrant/repogit/TP_AySO vagrant@VMPruebas:~/repogit/TP_AySO\$ ls -l total 4 -rw-rw-r-- 1 vagrant vagrant 9 Sep 6 20:26 README.md vagrant@VMPruebas:~/repogit/TP_AySO\$ </pre>

- Ver el contenido del archivo de configuración “os-release“

cat /etc/os-release

```

vagrant@VMPruebas:~/repogit/TP_AySO$ cat /etc/os-release
PRETTY_NAME="Ubuntu 22.04.3 LTS"
NAME="Ubuntu"
VERSION_ID="22.04"
VERSION="22.04.3 LTS (Jammy Jellyfish)"
VERSION_CODENAME=jammy
ID=ubuntu
ID_LIKE=debian
HOME_URL="https://www.ubuntu.com/"
SUPPORT_URL="https://help.ubuntu.com/"
BUG_REPORT_URL="https://bugs.launchpad.net/ubuntu/"
PRIVACY_POLICY_URL="https://www.ubuntu.com/legal/terms-and-policies/privacy-policy"
UBUNTU_CODENAME=jammy
vagrant@VMPruebas:~/repogit/TP_AySO$

```

```

upszot@halley:/tmp/borraime/UTN-FRA_SO_Vagrant/VagrantFiles/1_equipo_Multiples_GUI$ cat /etc/os-release
NAME="Fedora Linux"
VERSION="40 (MATE-Compiz)"
ID=fedora
VERSION_ID=40
VERSION_CODENAME=""
PLATFORM_ID="platform:f40"
PRETTY_NAME="Fedora Linux 40 (MATE-Compiz)"
ANSI_COLOR="0;38;2;60;110;180"
LOGO=fedora-logo-icon
CPE_NAME="cpe:/o:fedoraproject:fedora:40"
DEFAULT_HOSTNAME="fedora"
HOME_URL="https://fedoraproject.org/"
DOCUMENTATION_URL="https://docs.fedoraproject.org/en-US/fedora/f40/system-administrators-guide/"
SUPPORT_URL="https://ask.fedoraproject.org/"
BUG_REPORT_URL="https://bugzilla.redhat.com/"
REDHAT_BUGZILLA_PRODUCT="Fedora"
REDHAT_BUGZILLA_PRODUCT_VERSION=40
REDHAT_SUPPORT_PRODUCT="Fedora"
REDHAT_SUPPORT_PRODUCT_VERSION=40
SUPPORT_END=2025-05-13
VARIANT="MATE-Compiz"
VARIANT_ID=matecompiz
upszot@halley:/tmp/borraime/UTN-FRA_SO_Vagrant/VagrantFiles/1_equipo_Multiples_GUI$

```

- Verificar en la ayuda del comando grep como ignorar las mayusculas y minusculas.
- Posteriormente realizar una búsqueda de **HOME_URL** en el archivo anteriormente citado.
/etc/os-release

man grep	Ver las páginas del manual del comando grep, para buscar cómo realizar una búsqueda sin "case sensitive"
grep HOME_URL /etc/os-release	Realizar la búsqueda normalmente poniendo el string a buscar todo en mayúscula
grep home_url /etc/os-release	Realizar la búsqueda normalmente poniendo el string a buscar todo en minúscula
grep <parámetro> Home_url /etc/os-release	Por último agregar al comando grep el parámetro para poder realizar una búsqueda ignorando las mayúsculas y minúsculas Reemplazar <Parámetro> por el parámetro según el man.

- Mostramos por pantalla el nombre de usuario con el que estamos logeados

whoami	Muestra el nombre de usuario asociado al identificador de usuario efectivo actual. Equivale a id -un
--------	---------------------------------------------------------------------------------------------------------

- Creamos un archivo datos_usuario.txt con el siguiente formato "usuario:<nombre-del-usuario>

echo "Usuario=\$(whoami)" > datos_usuario.txt	<p>El Comando echo devolverá pro standard output...</p> <p>El texto encerrado entre comillas dobles, que se compone por:</p> <ul style="list-style-type: none"> - El literal "Usuario=" - El resultado de la expansión de comandos \$() que dentro de la misma se ejecuta el comando "whoami" <p>Por último se redirecciona con el operador > la salida del echo , hacia el archivo datos_usuario.txt</p>
-----------------------------------------------	---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

- Agregar todos los archivos generados en el Staging Area del repositorio local de git.

git add .	agregamos todos los archivos al staging
git config --global user.email "TU-email" git config --global user.name "Tu-Usuario"	configuramos el usuario de git
git commit -m "ADD: agregado 1er ejercicio sobre datos_usuarios.txt"	realizamos el commit Según las buenas practicas https://www.conventionalcommits.org/es/v1.0.0-beta.3/
<pre>vagrant@VMPruebas:~/repogit/TP_AyS0\$ echo "Usuario=\$(whoami)" > datos_usuario.txt vagrant@VMPruebas:~/repogit/TP_AyS0\$ git status On branch master Your branch is up to date with 'origin/master'. Untracked files: (use "git add <file>..." to include in what will be committed) datos_usuario.txt nothing added to commit but untracked files present (use "git add" to track) vagrant@VMPruebas:~/repogit/TP_AyS0\$ git add . vagrant@VMPruebas:~/repogit/TP_AyS0\$ git commit -m "ADD: agregado 1er ejercicio sobre datos_usuarios.txt" Author identity unknown *** Please tell me who you are. Run git config --global user.email "you@example.com" git config --global user.name "Your Name" to set your account's default identity. Omit --global to set the identity only in this repository. fatal: empty ident name (for <vagrant@VMPruebas>) not allowed vagrant@VMPruebas:~/repogit/TP_AyS0\$ git config --global user.email "[redacted]"; git config --global user.name "[redacted]" vagrant@VMPruebas:~/repogit/TP_AyS0\$ git commit -m "ADD: agregado 1er ejercicio sobre datos_usuarios.txt" [master 41133ad] ADD: agregado 1er ejercicio sobre datos_usuarios.txt 1 file changed, 1 insertion(+) create mode 100644 datos_usuario.txt vagrant@VMPruebas:~/repogit/TP_AyS0\$ git status On branch master Your branch is ahead of 'origin/master' by 1 commit. (use "git push" to publish your local commits) nothing to commit, working tree clean vagrant@VMPruebas:~/repogit/TP_AyS0\$</pre>	
git status	verificamos el estado



- **Añadir** al archivo anterior la información de modelo de cpu, obtenida del archivo cpuinfo, ubicado en el sistema de archivos virtual, /proc filtrando para que solamente agregue 1 entrada, independientemente de la cantidad de cores que tenga su vm.

Lista de comandos / tuberías / operadores que puede utilizar para lograr el objetivo:	
grep	para filtrar un string
tail -n1	para mostrar la -nX Últimas cantidades de líneas (siendo X=1.. mostrará la última línea)
head -n1	idem.. pero mostraría la 1er linea
"model name"	NO es un comando... Sería el string a buscar dentro del archivo... Como dicho string tiene espacios en blanco en el medio se debe encerrar entre comillas (dobles o simples)
	tubería: -> para utilizar la salida de un comando como entrada para el próximo comando.
>>	Operador: Doble redireccionamiento, para añadir en modo append.

- Volvemos a agregar al stage área del repositorio local el archivo modificado y volvemos a comitear. Usar como comentario: **"feat: Añadiendo información de CPU"**
- crear un archivo "lista_ordenada", obteniendo los 10 primeros usuarios (/etc/passwd) que el intérprete de comandos NO sea "nologin", y ordenar dicha lista por el intérprete de comandos (Columna 7)

Para resolver dicha consigna deberá utilizar los siguientes comandos	
grep	Verificar en man, como mostrar todo lo que NO coincida con el string
head	filtrar las 10 primeras líneas
sort	Verificar en el man, como ordenar por la columna número 7 AYUDA: puede buscar en el man el siguiente texto: "ordena según una determinada clave" o en inglés:

- Agregar al stage área del repositorio local el nuevo archivo "lista_ordenada" y volvemos a comitear.
Usar como comentario: **"ADD: Listado de usuarios ordenada"**
- Buscar con qué comando puede realizar un **"dump traffic"** de la red.... dejar el nombre del comando en el archivo **"comando_dump_net"** y agregar también que comando ejecutar para realizar dicha búsqueda..
- Agregar al stage área del repositorio local el nuevo archivo **"comando_dump_net"** y volvemos a comitear.
Usar como comentario: **"ADD: Comando para Capturar y analizar el tráfico de red "**
- Usando técnicas de HereDoc y redireccionamiento, Agregar al **README.md** la siguiente información.

Alumno: <Tu-Nombre>
División: <Tu-División>
Turno: <Turno>

- Agregar al stage área del repositorio local el nuevo archivo y volvemos a comitear.
Siguiendo las buenas practicas de: <https://www.conventionalcommits.org/es/v1.0.0-beta.3/>
Realizar el commit con la convención definida para el Tipo documentación., con el texto que crea conveniente.
- SIN MOVERSE DEL DIRECTORIO, utilizando PATHS Relativo...
- Forzar la Escritura del archivo donde se encuentra el historial de comandos

history -a

agrega (append) los comandos nuevos al archivo
\$HOME/.bash_history

- Copiar dentro del repositorio el archivo **.bash_history** que se encuentra en el home del usuario dejándolo con el nombre **"Historial_comandos_<Tu-Nombre>.txt"**
- Finalmente deberá realizar el último commit con su respectivo comentario y pushear los cambios contra el repositorio remoto de github
- **Todas las acciones(clone, add, commit, push) se deben realizar desde la terminal.**

Para entregar el TP, Por favor complete el siguiente [Formulario](#)

En caso de que lo funcione el Hipervínculo, aca abajo esta la url del formulario:

<https://docs.google.com/forms/d/e/1FAIpQLSdHivbiHgllEbFQGpPG-5-vxRjZWPSnioYZpBHgFjTXleR-GQ/viewform>