EJERCICIOS INICIO con GIT

a) Instalación: https://youtu.be/ES2xtLyI-B8

b) Explicación en video : https://youtu.be/imFd4zI7224

1) Crear nueva carpeta "Ejercicios". Crear en ella el documento : Ejercicio1.txt , que incluye la siguiente frase:
El router es la puerta de entrada a internet y encargado de que la comunicación entre todos tus dispositivos sea lo más óptima posible·

```
Adrian@AdrianConejo7 MINGW64 /c/hardware
$ mkdir Ejercicios

Adrian@AdrianConejo7 MINGW64 /c/hardware
$ cd Ejercicios/

Adrian@AdrianConejo7 MINGW64 /c/hardware/Ejercicios
$ touch Ejercicio1.txt

Adrian@AdrianConejo7 MINGW64 /c/hardware/Ejercicios
$ nano Ejercicio1.txt
e
```

2) Crear nuevo documento, Ejercicio2.txt, **en la carpeta "Ejercicios".** Escribir en él la frase:
Posibilidad de trabajar varios usuarios sobre el mismo documento : Este punto es el que destaca Google Drive sobre Dropbox.

```
drian@AdrianConejo7 MINGW64 /c/hardware/Ejercicios
 touch Ejercicio2.txt

drian@AdrianConejo7 MINGW64 /c/hardware/Ejercicios
 nano
```

3) Sobre la carpeta "Ejercicios" .... Boton derecho.... Git Bash Here (https://git-for-windows.github.io/)

Si en Git elegimos: Git Bash Here , ¿qué hace?
"Al abrir una terminal de Git Bash en un directorio específico (dentro de una carpeta), podemos trabajar con Git en ese directorio y realizar diversas operaciones, como confirmar cambios, hacer seguimientos, crear nuevas ramas. Esta se hará utilizando comandos de Git en la línea de comandos."

4) En la ventana de comandos que aparece, escribir el comando: `$ git init`

```
Adrian@AdrianConejo7 MINGW64 /c/hardware/Ejercicios
$ git init
Initialized empty Git repository in C:/hardware/Ejercicios/.git/
```

5) Escribir el sisguiente comando (con tu nombre de usuario)
`$ git config user.name matias`

```
Adrian@AdrianConejo7 MINGW64 /c/hardware/Ejercicios (master)
$ git config user.name adrian
```

¿Puedes explicarme el siguiente comando? git config user.name matias

Al ejecutar este comando, se establece el nombre de usuario "matias" en el repositorio de Git local (la carpeta donde estas trabajando). Si trabajas con otros colaboradores, permite identificar fácilmente quién realizó cada modificación. Es importante tener en cuenta que este comando solo configura el nombre del usuario en el repositorio de Git local (una carpeta). Para configurar el nombre de usuario en todos los repositorios de Git en el equipo, se puede utilizar el modificador **--global** junto con el comando **git config**. Por ejemplo:

```
git config --global user.name matias
```

6) Escribir el sisguiente comando (con tu email):

`$ git config user.email` matias@ibserveis.com

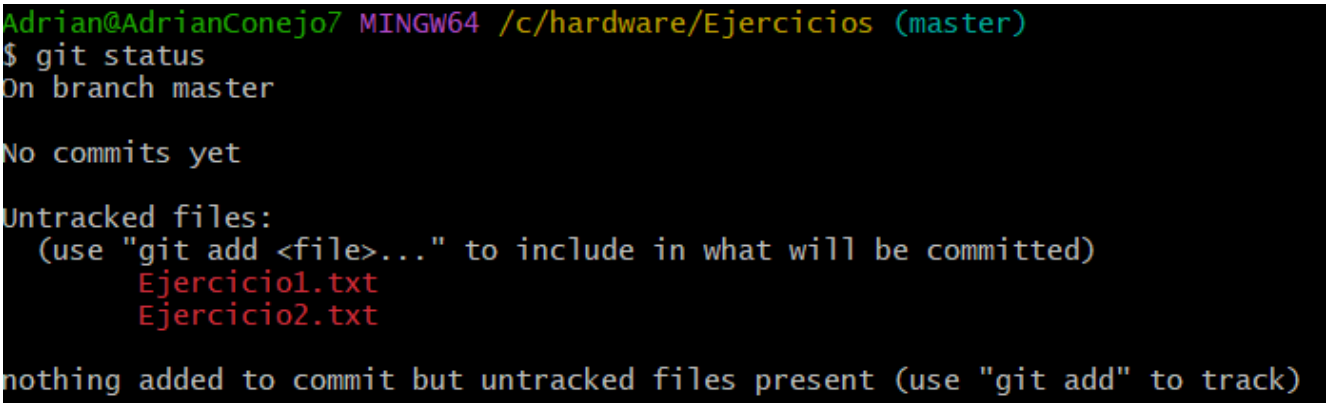
7) Explica con tus palabras que pueden haber hecho el comando anterior,

Este comando: `$ git config user.email` matias@ibserveis.com,
indica a Git la dirección de correo electrónico que se utilizará para identificar al usuario que está haciendo cambios en el repositorio. Esta información es importante porque se utiliza para rastrear la autoría de los cambios realizados en el repositorio.

8) Escribir el comando: `$git status`

¿Cuantos archivos hay "untracked"? ¿Qué significa?

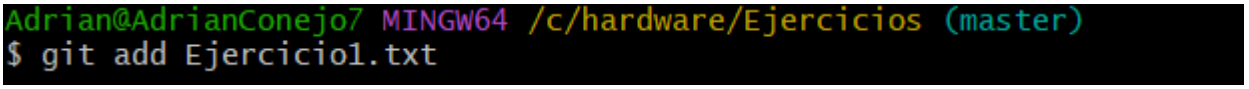
Hay dos archivos en untracked



"Untracked" ¿Qué significa en git?

Un archivo "untracked" es aquel que Git no está supervisando actualmente. Esto puede deberse a que es un archivo nuevo que nunca ha sido añadido al control de versiones, o puede ser un archivo que anteriormente estaba "tracked" pero fue eliminado del seguimiento.

9) Escribir el comando: `$git add Ejercicio1.txt`



10) Escribir el comando: `$git status`

¿Qué archivo no está "controlado"?

El archivo Ejercicio2.txt

¿Qué significa lo siguiente en git?

```
Changes to be committed: (use "git rm --cached <file>..." to unstage)
new file: prueba.docx
```

Este resultado de Git indica que el archivo **prueba.docx** se ha añadido al área de staging y está listo para ser confirmado (cometido) en el repositorio de Git.

La sección "Changes to be committed" muestra los archivos que se han añadido al área de staging y que están listos para ser confirmados. En este caso, hay un archivo nuevo llamado **prueba.docx** que se ha añadido al área de staging, lo que significa que Git está preparado para incluir este archivo en la próxima confirmación. Si se ejecuta el comando **git commit**, el archivo **prueba.docx** se confirmará y se guardará en el repositorio de Git.

11) Escribir el comando: `$git commit -m "Inicio texto, version 1"`

```
Adrian@AdrianConejo7 MINGW64 /c/hardware/Ejercicios (master)
$ git commit -m "inicio texto, version 1"
[master (root-commit) a389eec] inicio texto, version 1
 1 file changed, 1 insertion(+)
 create mode 100644 Ejercicio1.txt
```

12) Escribir el comando: `$git status`

¿Qué significa lo que contesta Git ?

El mensaje indica que el repositorio Git actual se encuentra en la rama "master" y que existen archivos sin seguimiento (untracked files) en el directorio de trabajo.

Explicación en video:

a) Configuraciones iniciales: https://youtu.be/lSpTok3_Jqw

b) Primer repositorio: https://youtu.be/m4wh8GhzcYg

13) Abrir documento Ejercicio1.txt , escribir la siguiente frase;

Si buscas nuevo router puede que el **Asus RT AC88U** te interese.

Guardar.

```
Adrian@AdrianConejo7 MINGW64 /c/hardware/Ejercicios (master)
$ nano Ejercicio1.txt
```

14) Escribir el comando: `$git status`

¿Se ha dado cuenta que archivo se ha modificado?

Si

```
Adrian@AdrianConejo7 MINGW64 /c/hardware/Ejercicios (master)
$ git status
On branch master
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   Ejercicio1.txt

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        Ejercicio2.txt

no changes added to commit (use "git add" and/or "git commit -a")
```

15) Escribir los siguientes comandos al sistema Git: `$git add Ejercicio1.txt`
`$git status`

```
Adrian@AdrianConejo7 MINGW64 /c/hardware/Ejercicios (master)
$ git status
On branch master
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        modified:   Ejercicio1.txt

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        Ejercicio2.txt
```

16) Añadimos "Ejercicio2.txt" al sistema: `$git add Ejercicio2.txt`

```
Adrian@AdrianConejo7 MINGW64 /c/hardware/Ejercicios (master)
$ git add Ejercicio2.txt
```

17) Realizamos "commit" conjunto los archivos :

`$git commit -m "Actualización Ejercicio1.txt y de Ejercicio2.txt"`

```
Adrian@AdrianConejo7 MINGW64 /c/hardware/Ejercicios (master)
$ git commit -m "Actualización Ejercicios1.txt y de Ejercicio2.txt"
[master 1af18a0] Actualización Ejercicios1.txt y de Ejercicio2.txt
 2 files changed, 2 insertions(+), 1 deletion(-)
 create mode 100644 Ejercicio2.txt
```

18) Escribir el comando: `$git status`

```
Adrian@AdrianConejo7 MINGW64 /c/hardware/Ejercicios (master)
$ git status
On branch master
nothing to commit, working tree clean
```

19) Escribir el comando: `$git log`

Ahí esta nuestra "línea del tiempo" donde podemos retroceder a cada versión

```
Adrian@AdrianConejo7 MINGW64 /c/hardware/Ejercicios (master)
$ git log
commit 1af18a0520a407a63d92a5fac94ee851b1f68e2c (HEAD -> master)
Author: adrian <aconejov75@alumnes.santjosepobrer.es>
Date:   Tue Mar 14 20:51:43 2023 +0100

    Actualización Ejercicios1.txt y de Ejercicio2.txt

commit a389eec0606348cb005610116fe4b3585670aacf
Author: adrian <aconejov75@alumnes.santjosepobrer.es>
Date:   Tue Mar 14 20:37:51 2023 +0100

    inicio texto, version 1
```

17) Escribir el siguiente comando,

substituyendo < aquí …. hash> por los 6 primeros caracteres del primer commit 'Inicio texto':

`$git checkout <aquí se escriben los 6 primeros n°del hash>`

```
Adrian@AdrianConejo7 MINGW64 /c/hardware/Ejercicios (master)
$ git checkout a389ee
Note: switching to 'a389ee'.

You are in 'detached HEAD' state. You can look around, make experimental
changes and commit them, and you can discard any commits you make in this
state without impacting any branches by switching back to a branch.

If you want to create a new branch to retain commits you create, you may
do so (now or later) by using -c with the switch command. Example:

  git switch -c <new-branch-name>

Or undo this operation with:

  git switch -

Turn off this advice by setting config variable advice.detachedHead to false

HEAD is now at a389eec inicio texto, version 1
```

18) Ir a la carpeta de "Ejercicios" y comprobar que texto hay en los archivos- ¿Es el texto inicial?

Si

19) Volver a colocar a línea del tiempo al último commit : `$ git checkout master`

En la carpeta Ejercicios., comprobar que texto tienen los archivos, en que estado se encuentran

```
Adrian@AdrianConejo7 MINGW64 /c/hardware/Ejercicios ((a389eec...))
$ git checkout master
Previous HEAD position was a389eec inicio texto, version 1
Switched to branch 'master'
```

El estado en el que se encuentran es el de la última modificación.

20) Utilizar esta otra forma git log: : `$git log --pretty=oneline`

```
Adrian@AdrianConejo7 MINGW64 /c/hardware/Ejercicios (master)
$ git log --pretty=oneline
1af18a0520a407a63d92a5fac94ee851b1f68e2c (HEAD -> master) Actualización Ejercici
os1.txt y de Ejercicio2.txt
a389eec0606348cb005610116fe4b3585670aacf inicio texto, version 1
```