

# Desarrollo WEB en entorno cliente

20/09/2022

Daniela Amoasii Marin

## PRACTICA 1

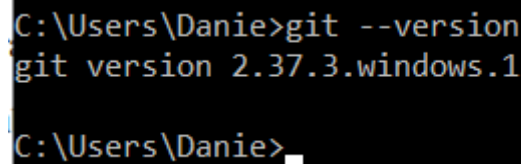
### Uso de Git y GitHub

#### Tabla de contenido

1. Fundamentos de Git I .....	1
2. Fundamentos de Git II .....	11
3. GitHub .....	28
4. Resumen de Comandos Aprendidos .....	32

## 1. Fundamentos de Git I

- 1) Instala Git en tu sistema operativo. Adjunta una captura de pantalla en la que aparezca el resultado de la ejecución del comando `git --version`.

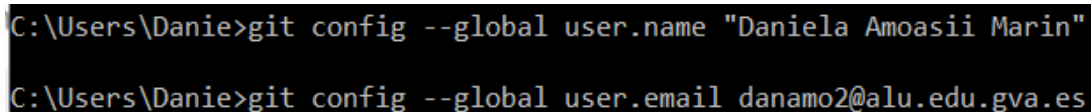


```
C:\Users\Danie>git --version
git version 2.37.3.windows.1
C:\Users\Danie>
```

Fig. 1. `git --version` se emplea para verificar si ya está instalado git y si es así, cual es la versión.

- 2) Realiza la configuración de Git según lo indicado en el tema (nombre, correo electrónico y editor de preferencia). Adjunta una captura de pantalla con el resultado de la ejecución de los comandos de configuración.

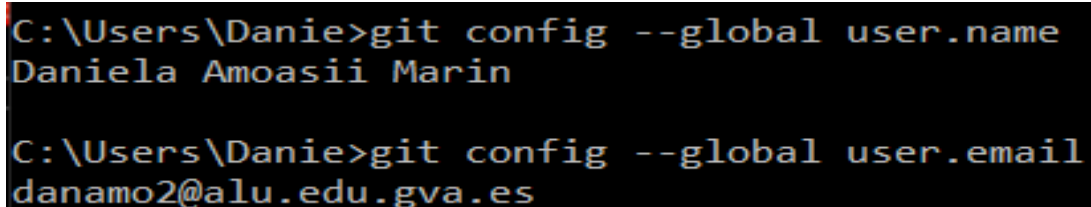
\*Realizar la configuración es imprescindible para poder realizar posteriormente los *comits*.



```
C:\Users\Danie>git config --global user.name "Daniela Amoasii Marin"
C:\Users\Danie>git config --global user.email danamo2@alu.edu.gva.es
```

Fig. 2. `--global` indica que la configuración se realiza a nivel de usuario. Esto hará que en los repositorios aparezcamos como el autor.

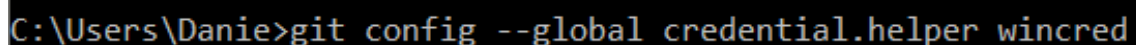
Comprobamos que la configuración se ha realizado correctamente.



```
C:\Users\Danie>git config --global user.name
Daniela Amoasii Marin
C:\Users\Danie>git config --global user.email
danamo2@alu.edu.gva.es
```

Fig. 3. Se solicita el nombre y correo del usuario, si la configuración se realizó correctamente, se devolverá el nombre del usuario y su correo electrónico.

No se solicita, pero realizare la configuración de almacenamiento de credenciales, esto se realiza para que no se solicite el usuario y contraseña cada vez que se trabaje con repositorios remotos en GitHub. Como estoy trabajando en Windows se ejecutará el siguiente comando:



```
C:\Users\Danie>git config --global credential.helper wincred
```

Fig. 4. Comando para el almacenamiento de credenciales.

Veamos la lista de las configuraciones que hemos realizado con el comando `git config --list`.

```

C:\Users\Danie>git config --global credential.helper wincred

C:\Users\Danie>git config --list
diff.astextplain.textconv=astextplain
filter.lfs.clean=git-lfs clean -- %f
filter.lfs.smudge=git-lfs smudge -- %f
filter.lfs.process=git-lfs filter-process
filter.lfs.required=true
http.sslbackend=openssl
http.sslcainfo=C:/Program Files/Git/mingw64/ssl/certs/ca-bundle.crt
core.autocrlf=true
core.fscache=true
core.symlinks=true
pull.rebase=false
credential.helper=manager-core
credential.https://dev.azure.com.usehttppath=true
init.defaultbranch=master
user.name=Daniela Amoasii Marin
user.email=danamo2@alu.edu.gva.es
pull.rebase=true
filter.lfs.clean=git-lfs clean -- %f
filter.lfs.smudge=git-lfs smudge -- %f
filter.lfs.process=git-lfs filter-process
filter.lfs.required=true
credential.helper=wincred

```

Fig. 5. Lista de las configuraciones realizadas.

3) Crea una carpeta denominada S1R1. Realiza las siguientes acciones en ella:



Fig. 6. Carpeta creada en el escritorio.

a) Crea un repositorio Git.

Dentro de la carpeta S1R1 crearé otra carpeta denominada “repo1”, una vez creada la carpeta nos posicionamos en dicha carpeta y posteriormente creamos el repositorio.

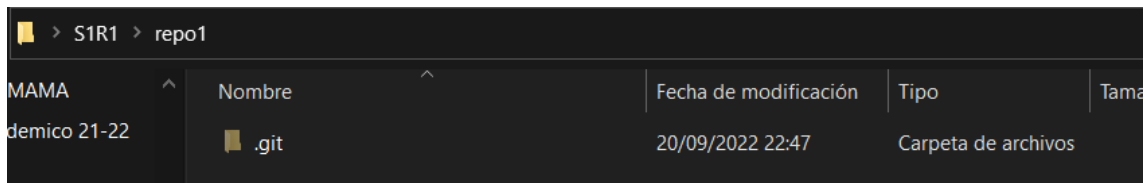
```

C:\Users\Danie> rem Nos desplazamos a la carpeta S1R1/repo1
C:\Users\Danie> cd C:\Users\Danie\Desktop\S1R1\repo1
C:\Users\Danie\Desktop\S1R1\repo1> rem Ejecutamos comando de creacion de repositorio
C:\Users\Danie\Desktop\S1R1\repo1> git init
Initialized empty Git repository in C:/Users/Danie/Desktop/S1R1/repo1/.git/

```

Fig. 7. *git init* se emplea para crear un repositorio git.

Comprobamos que se ha creado el repositorio, para ellos vamos a la carpeta repo1 de S1R1 y mostramos las carpetas ocultas.



Nombre	Fecha de modificación	Tipo	Tamaño
.git	20/09/2022 22:47	Carpeta de archivos	

Fig. 8. Se observa que se ha creado el repositorio ya que se muestra la carpeta *.git*.

Comprobamos el estatus del repositorio.

```

C:\Users\Danie\Desktop\S1R1\repo1> git status
On branch master

No commits yet

nothing to commit (create/copy files and use "git add" to track)

```

Fig. 9. El comando *git status*, indica que nos encontramos en la rama master y que todavía no se han realizado acciones.

- b) Crea un fichero denominado libros.txt. Añade tres títulos de libros cada uno en una línea distinta.

Se crea un Word con los tres libros y se guarda como libros.txt

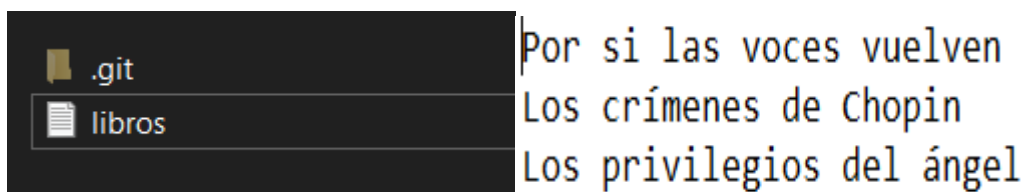


Fig. 10. Izquierda: documento denominado libros. Derecha: contenido del documento libros.

- c) Haz un primer *commit*.

Primero hare un git status para ver que está ocurriendo en el repositorio:

```

C:\Users\Danie\Desktop\S1R1\repo1>git status
On branch master

No commits yet

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

nothing added to commit but untracked files present (use "git add" to track)

```

Fig. 11. Se muestra un archivo sin seguimiento en rojo. Además, nos recomienda añadir el archivo mediante *git add*.

Seguimos las recomendaciones de git y añadimos el archivo, de tal modo el archivo pasará del estado sin seguimiento a un archivo que ya presenta seguimiento y por ende pasará al área de *staged* (preparado).

```

C:\Users\Danie\Desktop\S1R1\repo1>git add libros.txt
C:\Users\Danie\Desktop\S1R1\repo1>git status
On branch master

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
        new file:   libros.txt

```

Fig. 12. Tras haber añadido el archivo al repositorio, comprobaremos su estado. Al aparecer el archivo añadido en verde indica que ya está preparado.

A continuación, procede confirmar los cambios para pasarlos a la base de datos como la primera versión, esto se realizará con el comando *commit*.

```

C:\Users\Danie\Desktop\S1R1\repo1>git commit -m "Primera versión"
[master (root-commit) 4430752] Primera versión
1 file changed, 3 insertions(+)
create mode 100644 libros.txt

```

Fig. 13. Confirmamos los cambios de libros.txt, además con *-m* se puede añadir un comentario.

d) Añade dos libros al archivo libros.txt.

```

Por si las voces vuelven
Los crímenes de Chopin
Los privilegios del ángel
Harry Potter
El día que se perdió la cordura

```

Fig. 14. Nuevo contenido del archivo libros.txt.

Tras haber guardado los cambios en el archivo de libros, comprobamos que ocurre en el repositorio con el comando `status`.

```
C:\Users\Danie\Desktop\S1R1\repo1>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:   libros.txt

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

Fig. 15. Se indica que hay cambios en el archivo `libros.txt` que aún no se han guardado.

e) Haz un segundo *commit*.

Primero realizamos el comando `git add` para pasar el archivo al estado preparado y posteriormente se empleará `commit`.

```
C:\Users\Danie\Desktop\S1R1\repo1>git add libros.txt

C:\Users\Danie\Desktop\S1R1\repo1>git status
On branch master
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        modified:   libros.txt

C:\Users\Danie\Desktop\S1R1\repo1>git commit -m "Segunda Versión"
[master 7dcd8ce] Segunda Versión
1 file changed, 2 insertions(+)

C:\Users\Danie\Desktop\S1R1\repo1>git status
On branch master
nothing to commit, working tree clean
```

Fig. 16. Se guardan y confirman los cambios realizados en el archivo de los libros.

f) Crea un fichero denominado `películas.txt`. Añade tres títulos de películas a dicho archivo.

```
In time
Jumanji
Sing
```

Fig. 17. Contenido del archivo `películas.txt`

g) Haz una captura de pantalla del comando `git status`.

```

C:\Users\Danie\Desktop\S1R1\repo1>git status
On branch master
Untracked files:
  (use "git add <file>..." to include in what will be committed)
    peliculas.txt

nothing added to commit but untracked files present (use "git add" to track)

```

Fig. 18. Indica que el archivo películas no está en seguimiento y se recomienda realizar un *git add* para pasarlo al estado "preparado".

- h) Crea un fichero denominado comidas.txt. Añade tres nombres de comidas a dicho archivo.

```

Tacos
Pizza
Tortilla de patatas CON CEBOLLA

```

Fig. 19. Contenido del archivo comidas.txt.

- i) Haz un tercer commit que incluya los archivos peliculas.txt y comidas.txt.

Primero comprobamos el estado del repositorio:

```

C:\Users\Danie\Desktop\S1R1\repo1>git status
On branch master
Untracked files:
  (use "git add <file>..." to include in what will be committed)
    comidas
    peliculas.txt

nothing added to commit but untracked files present (use "git add" to track)

```

Fig. 20. Se indica que hay dos archivos que no se encuentran en seguimiento.

Se añaden los dos archivos nuevos al área de preparación y se comprueba el estado:

```

C:\Users\Danie\Desktop\S1R1\repo1>git add .
C:\Users\Danie\Desktop\S1R1\repo1>git status
On branch master
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
    new file:   comidas
    new file:   peliculas.txt

```

A continuación, ya se puede realizar el *commit* para confirmar los documentos.

```
C:\Users\Danie\Desktop\S1R1\repo1>git commit -m "Tercera versión"
[master 456f6bb] Tercera versión
2 files changed, 6 insertions(+)
create mode 100644 comidas
create mode 100644 peliculas.txt
```

Fig. 21. Se realiza el commit exitosamente.

- j) Elimina el archivo comidas.txt desde el navegador de archivos.

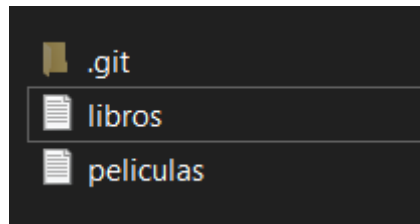


Fig. 22. Se ha eliminado el archivo de comidas del directorio de trabajo.

- k) Añade dos películas más al archivo peliculas.txt.

```
In time
Jumanji
Sing
Los increíbles
Espejo, espejo
```

Fig. 23. Nuevo contenido del archivo peliculas.txt.

- l) Haz una captura de pantalla que muestre los cambios en el directorio de trabajo.

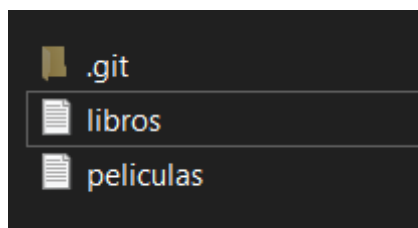


Fig. 24. Contenido del directorio de trabajo.

```
C:\Users\Danie\Desktop\S1R1\repo1>git status
On branch master
Changes not staged for commit:
  (use "git add/rm <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        deleted:    comidas
        modified:   peliculas.txt

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

Fig. 25. Situación del directorio. Muestra que se ha borrado el archivo de comidas y que se ha modificado el archivo de películas.

- m) Añade los cambios al área de preparación.



```
C:\Users\Danie\Desktop\S1R1\repo1>git add .
```

Fig. 26. Se guardan los cambios realizados en todos los archivos.

- n) Haz una captura de pantalla del comando `git status`. Debe indicar que se ha borrado el archivo `comidas.txt` y que se ha modificado el archivo `peliculas.txt`.

```
C:\Users\Danie\Desktop\S1R1\repo1>git status
On branch master
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        deleted:      comidas
        modified:     peliculas.txt
```

Fig. 27. Se muestra que se ha eliminado el archivo de comidas y que se ha modificado el archivo de películas.

- o) Haz un cuarto *commit*.

```
C:\Users\Danie\Desktop\S1R1\repo1>git commit -m "Cuarta Versión"
[master e574824] Cuarta Versión
 2 files changed, 3 insertions(+), 4 deletions(-)
 delete mode 100644 comidas
```

Fig. 28. Se confirman los cambios guardados.

- p) Crea un archivo denominado `datos.bak`. Añade tres títulos de libros a dicho archivo. ¡IMPORTANTE! No añadas el archivo al área de preparación ni hagas ningún *commit*.

```
El perfume
El principito
A sangre fria
```

Fig. 29. Contenido del archivo `datos.bak`.

- q) Crea una subcarpeta denominada `output`. Crea un archivo denominado `salida.txt` en su interior. Escribe tu nombre y apellidos en dicho archivo. ¡IMPORTANTE! No añadas los archivos al área de preparación ni hagas ningún *commit*.



Fig. 30. Contenido del directorio de trabajo. Contenido de la carpeta `output`. Contenido de `salida.txt`.

- r) Haz una captura de pantalla del comando `git status`. Deben aparecer los archivos `datos.bak` y `output/salida.txt` como archivos nuevos (color rojo).

```
C:\Users\Danie\Desktop\S1R1\repo1>git status
On branch master
Untracked files:
  (use "git add <file>..." to include in what will be committed)
    datos.bak
    output/

nothing added to commit but untracked files present (use "git add" to track)
```

Fig. 31. Muestra dos archivos que no se encuentran en seguimiento.

- s) Crea un archivo `.gitignore` para que los ficheros con extensión `.bak` y el contenido de la carpeta `output/` no se incluyan en el repositorio.

```
*.bak
*output/
```

Fig. 32. Contenido del archivo `.gitignore`.

- t) Haz una nueva captura de pantalla del comando `git status`. Ahora no deben aparecer los archivos `datos.bak` y `output/salida.txt` como archivos nuevos, sino que en su lugar debe aparecer únicamente el archivo `.gitignore`.

```
C:\Users\Danie\Desktop\S1R1\repo1>git status
On branch master
Untracked files:
  (use "git add <file>..." to include in what will be committed)
    .gitignore

nothing added to commit but untracked files present (use "git add" to track)
```

Fig. 33. Vemos que ya no se detectan los archivos `datos.bak` y `output/`.

- u) Haz un último commit para incluir el archivo `.gitignore` en el repositorio.

Primero hay que guardar los cambios con `git add`.

```
C:\Users\Danie\Desktop\S1R1\repo1>git add .
C:\Users\Danie\Desktop\S1R1\repo1>git status
On branch master
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
    new file:   .gitignore
```

Fig. 34. Se guardan los cambios y se comprueba que se han guardado correctamente.

Se confirman los cambios realizados con el commit.

```
C:\Users\Danie\Desktop\S1R1\repo1>git commit -m "Quinta Versión"
[master 47d5a93] Quinta Versión
1 file changed, 2 insertions(+)
create mode 100644 .gitignore
```

Fig. 35. Se realiza el ultimo commit.

v) Haz una captura de pantalla que muestre el histórico de cambios del repositorio.

```
C:\Users\Danie\Desktop\S1R1\repo1>git log --graph
* commit 47d5a9328d045a506a9043d6c8a1a98a82468c65 (HEAD -> master)
  Author: Daniela Amoasii Marin <danamo2@alu.edu.gva.es>
  Date:   Wed Sep 21 00:52:14 2022 +0200

    Quinta Versión

* commit e574824e4b90dd9ee1d6306f177c6a9b2d363a8c
  Author: Daniela Amoasii Marin <danamo2@alu.edu.gva.es>
  Date:   Wed Sep 21 00:18:16 2022 +0200

    Cuarta Versión

* commit 456f6bb2a21373e7314799f4b37c61937375f6c2
  Author: Daniela Amoasii Marin <danamo2@alu.edu.gva.es>
  Date:   Tue Sep 20 23:59:19 2022 +0200

    Tercera versión

* commit 7dcd8cec5206082bc64f12f1588c2002b1b34e4f
  Author: Daniela Amoasii Marin <danamo2@alu.edu.gva.es>
  Date:   Tue Sep 20 23:37:56 2022 +0200

    Segunda Versión

* commit 443075214fe7d5d602f36044c52f8dd9729a8b35
  Author: Daniela Amoasii Marin <danamo2@alu.edu.gva.es>
  Date:   Tue Sep 20 23:18:49 2022 +0200

    Primera versión
```

## 2. Fundamentos de Git II

- 1) Crea una carpeta denominada S2R1. Realiza las siguientes acciones en ella:

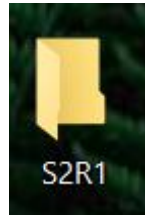


Fig. 36. Carpeta creada en el escritorio.

- a) Crea un repositorio Git.

Primero hay que moverse a la carpeta que deseamos (`cd ruta`) convertir en repositorio, posteriormente se crea el repositorio con el comando `git init`.

```
C:\Users\Danie\Desktop>cd C:\Users\Danie\Desktop\S2R1
C:\Users\Danie\Desktop\S2R1>git init
Initialized empty Git repository in C:/Users/Danie/Desktop/S2R1/.git/
C:\Users\Danie\Desktop\S2R1>git status
On branch master

No commits yet

nothing to commit (create/copy files and use "git add" to track)
```

Fig. 37. Se crea el repositorio git S2R1 y se comprueba su estado. Se encuentra en la rama master y el repositorio está vacío.

- b) Crea un fichero denominado actores.txt. Añade tres nombres de actores cada uno en una línea distinta.

Abrimos Brackets y añadimos los tres actores, a continuación, le damos a guardar como y denominamos el fichero como "actores.txt".

```
Drew Barrymore
Will Smith
Milla Jovovich
```

Fig. 38. Contenido del fichero "actores.txt"

- c) Haz un primer commit.

Primero se tiene que hacer un `git add` para guardar los cambios y posteriormente se realiza la confirmación de los cambios (`commit`).

Realizaré un `git add` . ya que es más rápido y además solo tengo un archivo creado, por lo cual no supone un problema ya que no hay nada mas para guardar.

```

C:\Users\Danie\Desktop\S2R1>git add .

C:\Users\Danie\Desktop\S2R1>git status
On branch master

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
        new file:   actores.txt

C:\Users\Danie\Desktop\S2R1>git commit -m "Primera Versión"
[master (root-commit) 49e0d48] Primera Versión
 1 file changed, 3 insertions(+)
 create mode 100644 actores.txt

```

Fig. 39. Se guardan los cambios, se comprueba que los cambios se han guardado correctamente (fichero en verde) y por último se confirman los cambios.

d) Crea una rama denominada test.

```

C:\Users\Danie\Desktop\S2R1>git branch test

C:\Users\Danie\Desktop\S2R1>git branch
* master
test ←

```

Fig. 40. Se crea la rama test y se confirma que se ha creado correctamente.

e) Cambia a la rama test.

```

C:\Users\Danie\Desktop\S2R1>git checkout test
Switched to branch 'test'

C:\Users\Danie\Desktop\S2R1>git branch
  master
* test

```

Fig. 41. Nos cambiamos a la rama "test" y comprobamos que realmente estamos en dicha rama. El asterisco y el nombre en verde indica que nos encontramos en dicha rama.

f) En la rama test crea un fichero denominado actrices.txt. Añade tres nombres de actrices y realiza un commit en dicha rama.

```

Kerry Washington
Margor Robbie
Anne Hathaway

```

Fig. 42. Contenido del fichero "actrices.txt".

```

C:\Users\Danie\Desktop\S2R1>git add actresses.txt

C:\Users\Danie\Desktop\S2R1>git status
On branch test
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
    new file:   actresses.txt

C:\Users\Danie\Desktop\S2R1>git commit -m "Segunda Versión"
[test 50f6f92] Segunda Versión
1 file changed, 3 insertions(+)
create mode 100644 actresses.txt

```

Fig. 43. Primero se guardan los cambios, se comprueba que los cambios se han guardado correctamente y en que rama, finalmente se confirman los cambios guardados.

g) Haz una captura de pantalla del resultado del comando `git log --graph --all`.

```

C:\Users\Danie\Desktop\S2R1>git log --graph --all
* commit 50f6f92c298fb06cb06eec747691329d223208c3 (HEAD -> test)
  Author: Daniela Amoasii Marin <danamo2@alu.edu.gva.es>
  Date:   Wed Sep 21 13:04:26 2022 +0200

  Segunda Versión

* commit 49e0d485a9845942be4a9cc26215f168d756fb91 (master)
  Author: Daniela Amoasii Marin <danamo2@alu.edu.gva.es>
  Date:   Wed Sep 21 12:42:02 2022 +0200

  Primera Versión

```

Fig. 44. Vemos que la primera versión se encuentra guardada en la rama master y la segunda versión está guardada en la rama test. Además, nos indica que ahora nos encontramos en la rama test.

h) Cambia a la rama master.

```

C:\Users\Danie\Desktop\S2R1>git checkout master
Already on 'master'

C:\Users\Danie\Desktop\S2R1>git branch
* master
test

```

Fig. 45. Cambiamos a la rama master y confirmamos que nos encontramos en dicha rama.

i) Incorpora los cambios de la rama test a la rama master. Haz una captura de pantalla de los comandos que has utilizado y de su resultado.

```
C:\Users\Danie\Desktop\S2R1>git merge test
Updating 49e0d48..50f6f92
Fast-forward
 actrices.txt | 3 +++
 1 file changed, 3 insertions(+)
 create mode 100644 actrices.txt
```

Fig. 46. Master ha incorporado la rama test.

- j) Crea una segunda rama denominada test2. La rama test2 apunta al mismo commit que la rama master en este momento.

```
C:\Users\Danie\Desktop\S2R1>git branch test2

C:\Users\Danie\Desktop\S2R1>git branch
* master
  test
  test2
```

Fig. 47. Se crea la nueva rama y se comprueba que se ha creado correctamente.

- k) En la rama master, añade una actriz al fichero actrices.txt y haz un commit.

Kerry Washington  
 Margor Robbie  
 Anne Hathaway  
 Dakota Johnson

Fig. 48. Contenido del fichero actrices.txt tras haber añadido una actriz más.

```
C:\Users\Danie\Desktop\S2R1>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:   actrices.txt

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

C:\Users\Danie\Desktop\S2R1>git add .

C:\Users\Danie\Desktop\S2R1>git status
On branch master
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        modified:   actrices.txt

C:\Users\Danie\Desktop\S2R1>git commit -m "Tercera Versión"
[master 94c37dc] Tercera Versión
 1 file changed, 2 insertions(+), 1 deletion(-)
```

Fig. 49. Se comprueba el estado del repositorio. Se guardan los cambios realizados, se comprueba que se han guardado correctamente. Se confirma los cambios guardados.

l) Cambia a la rama test2

```
C:\Users\Danie\Desktop\S2R1>git checkout test2
Switched to branch 'test2'

C:\Users\Danie\Desktop\S2R1>git branch
  master
  test
* test2
```

Fig. 50. Cambiamos a la rama test2 y confirmamos que realmente estamos en dicha rama.

m) En la rama test2, añade una actriz al fichero actrices.txt y haz otro commit.

```
Kerry Washington
Margor Robbie
Anne Hathaway
Karen Gillan
```

Fig. 51. Contenido del fichero actrices.txt en la rama test2.

```
C:\Users\Danie\Desktop\S2R1>git status
On branch test2
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:   actrices.txt

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

C:\Users\Danie\Desktop\S2R1>git add .

C:\Users\Danie\Desktop\S2R1>git status
On branch test2
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
    modified:   actrices.txt

C:\Users\Danie\Desktop\S2R1>git commit -m "Cuarta Versión"
[test2 ac30d99] Cuarta Versión
1 file changed, 2 insertions(+), 1 deletion(-)
```

Fig. 52. Primero compruebo que ocurre en el repositorio. Guardamos los cambios realizados y se comprueba que se han guardado correctamente. Por último, se confirma los cambios. También se comprueba que en todos los pasos nos encontramos en la rama test2.

n) Haz una captura de pantalla del resultado del comando git log --graph --all. Debe haber dos caminos distintos: uno para la rama master y otro para la rama test2.



```

C:\Users\Danie\Desktop\S2R1>git log --graph --all
* commit ac30d99a79084c29c471727b16b323c152e5b229 (HEAD -> test2)
  Author: Daniela Amoasii Marin <danamo2@alu.edu.gva.es>
  Date:   Wed Sep 21 14:25:00 2022 +0200

      Cuarta Versión

* commit 94c37dc11a5bed1d9a49540a5a4fc317d4c49ffd (master)
  Author: Daniela Amoasii Marin <danamo2@alu.edu.gva.es>
  Date:   Wed Sep 21 14:14:28 2022 +0200

      Tercera Versión

* commit 50f6f92c298fb06cb06eec747691329d223208c3 (test)
  Author: Daniela Amoasii Marin <danamo2@alu.edu.gva.es>
  Date:   Wed Sep 21 13:04:26 2022 +0200

      Segunda Versión

* commit 49e0d485a9845942be4a9cc26215f168d756fb91
  Author: Daniela Amoasii Marin <danamo2@alu.edu.gva.es>
  Date:   Wed Sep 21 12:42:02 2022 +0200

      Primera Versión

```

Fig. 53. En mi caso tengo una rama más ya que no he llegado a borrar la rama test. También se muestra que la rama test2 deriva de la rama master (líneas rojas).

o) Cambia a la rama master.

```

C:\Users\Danie\Desktop\S2R1>git checkout master
Switched to branch 'master'

C:\Users\Danie\Desktop\S2R1>git branch
* master
  test
  test2

```

Fig. 54. Cambio a la rama master y compruebo que realmente estoy en dicha rama.

p) Incorpora los cambios de la rama test2 a la rama master. ¿Se produce un conflicto? De ser así realiza una captura del comando git status.

```

C:\Users\Danie\Desktop\S2R1>git merge test2
Auto-merging actrices.txt
CONFLICT (content): Merge conflict in actrices.txt
Automatic merge failed; fix conflicts and then commit the result.

C:\Users\Danie\Desktop\S2R1>git status
On branch master
You have unmerged paths.
  (fix conflicts and run "git commit")
  (use "git merge --abort" to abort the merge)

Unmerged paths:
  (use "git add <file>..." to mark resolution)
    both modified:   actrices.txt

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

```

Fig. 55. Se produce conflicto a la hora de incorporar la rama test2 a la master. Esto es debido a que se han producido en ambas ramas cambios en el mismo archivo.

q) Resuelve el conflicto incorporando los dos nombres de actrices.

Estando en la rama master, vamos al directorio de trabajo y abrimos el archivo con el conflicto, en este caso el archivo “actrices.txt”. Eliminamos los caracteres de más dejando solo las dos actrices que queremos que se guarden. Guardamos y cerramos el archivo.

```

Kerry Washington
Margor Robbie
Anne Hathaway
<<<<<< HEAD
Dakota Johnson
=====
Karen Gillan
>>>>>> test2

```

Fig. 56. Archivo con conflicto. Los caracteres especiales muestran donde se encuentra el conflicto

```

Kerry Washington
Margor Robbie
Anne Hathaway
Dakota Johnson
Karen Gillan

```

Fig. 57. Una vez eliminados los caracteres especiales obtenemos el siguiente resultado.

```

C:\Users\Danie\Desktop\S2R1>git add .

C:\Users\Danie\Desktop\S2R1>git status
On branch master
All conflicts fixed but you are still merging.
  (use "git commit" to conclude merge)

Changes to be committed:
  modified:   actrices.txt

C:\Users\Danie\Desktop\S2R1>git commit -m "Quinta Versión"
[master 4517094] Quinta Versión

C:\Users\Danie\Desktop\S2R1>

```

Fig. 58. Se guarda el cambio realizado en el archivo, comprobamos el estado del repositorio y como todo está en verde procedemos a confirmar los cambios.

- r) Haz una captura de pantalla del resultado del comando `git log --graph --all`. Observa que se ha creado un nuevo commit que integra los dos caminos anteriores.

```

C:\Users\Danie\Desktop\S2R1>git log --graph --all
*   commit 451709489f92f7150f65293bfb6b99b0e7bc8fb9 (HEAD -> master)
|  \
|   Merge: 94c37dc ac30d99
|   Author: Daniela Amoasii Marin <danamo2@alu.edu.gva.es>
|   Date:   Wed Sep 21 14:45:11 2022 +0200
|
|       Quinta Versión
|
| *   commit ac30d99a79084c29c471727b16b323c152e5b229 (test2)
|   \
|    Author: Daniela Amoasii Marin <danamo2@alu.edu.gva.es>
|    Date:   Wed Sep 21 14:25:00 2022 +0200
|
|       Cuarta Versión
|
| *   commit 94c37dc11a5bed1d9a49540a5a4fc317d4c49ffd
|   /
|    Author: Daniela Amoasii Marin <danamo2@alu.edu.gva.es>
|    Date:   Wed Sep 21 14:14:28 2022 +0200
|
|       Tercera Versión
|
| *   commit 50f6f92c298fb06cb06eec747691329d223208c3 (test)
|   \
|    Author: Daniela Amoasii Marin <danamo2@alu.edu.gva.es>
|    Date:   Wed Sep 21 13:04:26 2022 +0200
|
|       Segunda Versión
|
| *   commit 49e0d485a9845942be4a9cc26215f168d756fb91
|   \
|    Author: Daniela Amoasii Marin <danamo2@alu.edu.gva.es>
|    Date:   Wed Sep 21 12:42:02 2022 +0200
|
|       Primera Versión

```

Fig. 59. Vemos que el nuevo commit integra el camino master anterior y el test2.

- 2) Crea una carpeta denominada S2R2-remoto. Inicializa un repositorio Git en su interior mediante el comando `git init --bare`. Esta carpeta se utilizará como repositorio remoto.

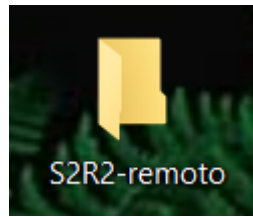


Fig. 60. Carpeta creada en el escritorio.

Primero nos desplazamos al interior de la carpeta y posteriormente se crea el repositorio.

```
C:\Users\Danie\Desktop\S2R1>cd ..  
C:\Users\Danie\Desktop>cd S2R2-remoto  
C:\Users\Danie\Desktop\S2R2-remoto>git init --bare  
Initialized empty Git repository in C:/Users/Danie/Desktop/S2R2-remoto/
```

Fig. 61. Se crea un repositorio, sin embargo, este no está activo para poder trabajar en él directamente. Esto se especifica así ya que, si el repositorio estuviera activo, no se podrían realizar acciones push sobre él.

- 3) Clona el repositorio S2R2-remoto en una carpeta denominada S2R2. Adjunta captura de pantalla del resultado del comando de clonado. A continuación, realiza las siguientes acciones en el repositorio S2R2:

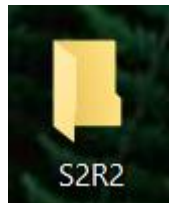


Fig. 62. Carpeta creada en el escritorio.

```
C:\Users\Danie\Desktop\S2R2-remoto>cd C:\Users\Danie\Desktop\S2R2  
C:\Users\Danie\Desktop\S2R2>git clone C:\Users\Danie\Desktop\S2R2-remoto repo1  
Cloning into 'repo1'...  
warning: You appear to have cloned an empty repository.  
done.
```

Fig. 63. Se clona el repositorio, proporcionando la ruta del repositorio remoto, a continuación, se le pide que la clonación se guarde en una carpeta llamada repo1.

- a) Crea un archivo denominado `directores.txt`. Añade el nombre de tres directores de cine.

```
Martin Scorsese  
Kathryn Bigelow  
Wes Anderson
```

Fig. 64. Contenido del archivo `directores.txt`.

- b) Haz un commit.

```

C:\Users\Danie\Desktop\S2R2>cd repo1
C:\Users\Danie\Desktop\S2R2\repo1>git status
On branch master

No commits yet

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

nothing added to commit but untracked files present (use "git add" to track)
C:\Users\Danie\Desktop\S2R2\repo1>git add .
C:\Users\Danie\Desktop\S2R2\repo1>git status
On branch master

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
    new file:   directores.txt

C:\Users\Danie\Desktop\S2R2\repo1>git commit -m "Primera Versión"
[master (root-commit) c983747] Primera Versión
 1 file changed, 6 insertions(+)
 create mode 100644 directores.txt

```

Fig. 65. Primero nos movemos al repositorio, después se comprueba que ocurre dentro, se nos indica que hay documentos no atendidos, por lo cual se guarda los cambios del fichero con add, se comprueba que se han guardado correctamente, como el archivo está en verde se procede a confirmar los cambios.

c) Realiza un push al repositorio remoto. Adjunta captura de pantalla del resultado.

push: subir

origin: subir el repositorio local al repositorio remoto de donde lo hemos descargado.

master: subir los cambios a la rama master del original.

```

C:\Users\Danie\Desktop\S2R2\repo1>git push origin master
Enumerating objects: 3, done.
Counting objects: 100% (3/3), done.
Delta compression using up to 12 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 307 bytes | 307.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
To C:\Users\Danie\Desktop\S2R2-remoto
 * [new branch]      master -> master

```

Fig. 66. Se envía el repositorio actual al repositorio remoto y a la rama master.

```
C:\Users\Danie\Desktop\S2R2\repo1>git log --all
commit c983747ee782519eb7f20e673a83c27446c27a96 (HEAD -> master, origin/master)
Author: Daniela Amoasii Marin <danamo2@alu.edu.gva.es>
Date:   Wed Sep 21 16:49:52 2022 +0200
```

Fig. 67. Comprobamos que el push ha funcionado, ya que en esta imagen se me indica que la rama master del directorio actual está conectada con el repositorio remoto.

d) Crea una rama denominada version1.

```
C:\Users\Danie\Desktop\S2R2\repo1>git branch version1

C:\Users\Danie\Desktop\S2R2\repo1>git branch
* master
version1 ←
```

Fig. 68. Se crea la nueva rama y se comprueba si se ha creado correctamente.

e) Cambia a la rama version1.

```
C:\Users\Danie\Desktop\S2R2\repo1>git checkout version1
Switched to branch 'version1'

C:\Users\Danie\Desktop\S2R2\repo1>git branch
  master
* version1
```

Fig. 69. Cambiamos a la rama version1 y comprobamos que estamos en ella (aparece en verde).

f) En la rama version1 añade el nombre de dos directores de cine más al archivo directores.txt y haz un commit de los cambios.

```
Martin Scorsese
Kathryn Bigelow
Wes Anderson
Pedro Almodovar
Darren Aronofsky
```

Fig. 70. Contenido actual del archivo directores.txt

```

C:\Users\Danie\Desktop\S2R2\repo1>git status
On branch version1
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:   directores.txt

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

C:\Users\Danie\Desktop\S2R2\repo1>git add .

C:\Users\Danie\Desktop\S2R2\repo1>git status
On branch version1
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        modified:   directores.txt

C:\Users\Danie\Desktop\S2R2\repo1>git commit -m "Version 2"
[version1 2a8d45b] Version 2
1 file changed, 5 insertions(+), 6 deletions(-)

```

Fig. 71. Como siempre se mira que ocurre en el repositorio, se guardan los cambios, y si están bien guardados se confirma los cambios.

- g) Realiza un push de la rama al repositorio remoto de manera que quede asociada a la rama remota del mismo nombre. Adjunta captura de pantalla del resultado.

```

C:\Users\Danie\Desktop\S2R2\repo1>git push origin version1
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 12 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 330 bytes | 55.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
To C:\Users\Danie\Desktop\S2R2-remoto
 * [new branch]      version1 -> version1

C:\Users\Danie\Desktop\S2R2\repo1>git log --all
commit 2a8d45bf0536d78a9fc688ef4b1075a773d437c5 (HEAD -> version1, origin/version1)
Author: Daniela Amoasii Marin <danamo2@alu.edu.gva.es>
Date:   Wed Sep 21 17:22:02 2022 +0200

    Version 2

commit c983747ee782519eb7f20e673a83c27446c27a96 (origin/master, master)
Author: Daniela Amoasii Marin <danamo2@alu.edu.gva.es>
Date:   Wed Sep 21 16:49:52 2022 +0200

    Primera Versión

```

Fig. 72. se hace el push a una rama que llamaremos version1, a continuación, comprobamos que la relación se ha realizado correctamente.

- 4) Clona el repositorio S2R2-remoto en una segunda carpeta denominada S2R3. Realiza las siguientes acciones sobre ella:



Fig. 73. Carpeta creada en el escritorio.

```
C:\Users\Danie\Desktop\S2R2\repo1>cd C:\Users\Danie\Desktop\S2R3
C:\Users\Danie\Desktop\S2R3>git clone C:\Users\Danie\Desktop\S2R2-remoto repo1
Cloning into 'repo1'...
done.
```

Fig. 74. Se clona el repositorio, proporcionando la ruta del repositorio remoto, a continuación, se le pide que la clonación se guarde en una carpeta llamada repo1.

- a) Muestra en la consola el contenido del fichero directores.txt y el resultado del comando git status. Debe mostrar tres directores.

```
C:\Users\Danie\Desktop\S2R3\repo1>type directores.txt
Martin Scorsese
Kathryn Bigelow
Wes Anderson
```

Fig. 75. Contenido del archivo directores.txt.

```
C:\Users\Danie\Desktop\S2R3\repo1>git status
On branch master
Your branch is up to date with 'origin/master'.

nothing to commit, working tree clean
```

Fig. 76. No hay nada para confirmar, ya que no se han modificado los archivos.

- b) Cambia a la rama version1. Muestra el resultado del comando. Comprueba que se crea una rama local version1 con el contenido de la rama remota origin/version1 y enlazada con ella. Al clonar el repositorio la rama no existía (solo se clona la rama principal, master), pero al cambiar a una rama que existe en el remoto se produce su creación local y enlazado con su correspondiente remota.



```
C:\Users\Danie\Desktop\S2R3\repo1>git checkout version1
Switched to a new branch 'version1'
branch 'version1' set up to track 'origin/version1'.

C:\Users\Danie\Desktop\S2R3\repo1>git branch
  master
* version1
```

Fig. 77. Se ha cambiado a la rama version1 la cual apunta a origin/version1. Posteriormente comprobamos que realmente estemos en dicha rama.

- c) Muestra el contenido del fichero directores.txt por la pantalla. Comprueba que se muestran los 5 nombres de directores esperados. Adjunta captura de pantalla.

```
C:\Users\Danie\Desktop\S2R3\repo1>type directores.txt
Martin Scorsese
Kathryn Bigelow
Wes Anderson
Pedro Almodovar
Darren Aronofsky
```

Fig. 78. Contenido del archivo directores.txt.

- d) Cambia a la rama master.

```
C:\Users\Danie\Desktop\S2R3\repo1>git checkout master
Switched to branch 'master'
Your branch is ahead of 'origin/master' by 1 commit.
(use "git push" to publish your local commits)

C:\Users\Danie\Desktop\S2R3\repo1>git branch
* master
  version1
```

Fig. 79. Cambiamos a la rama master y comprobamos que efectivamente estamos dentro.

- e) Incorpora los cambios de la rama version1 a la rama master.

```
C:\Users\Danie\Desktop\S2R3\repo1>git merge version1
Auto-merging directores.txt
CONFLICT (content): Merge conflict in directores.txt
Automatic merge failed; fix conflicts and then commit the result.
```

Fig. 80. Al intentar incorporar los cambios de la version1 nos surgen conflictos.

En este paso no debería surgir conflictos, ya que se supone que en el archivo directores de la rama master solo hay 3 directores y nada más, y en la rama version1 se han añadido dos nombres más, con lo cual debería incorporar los cambios sin ningún problema, sin embargo, en el archivo que se encuentra en la rama master he dejado dos saltos de línea, con lo cual git interpreta que hay información en esos espacios. Para resolver el conflicto me dirijo al directorio

de trabajo, abro el archivo de directores.txt y elimino los espacios y los caracteres especiales, combinando de tal modo ambos archivos tal como se observará en las siguientes figuras.

```
Martin Scorsese
Kathryn Bigelow
Wes Anderson
<<<<<< HEAD
=====
Pedro Almodovar
Darren Aronofsky
>>>>>> version1
```

} Dos saltos de línea

Fig. 81. Contenido del archivo que produce conflicto debido a los saltos de línea dejados.

```
Martin Scorsese
Kathryn Bigelow
Wes Anderson
Pedro Almodovar
Darren Aronofsky
```

Fig. 82. Resultado tras haber eliminado los saltos de línea y los caracteres especiales.

Ahora para fusionar ambas ramas se procede a realizar un add para guardar los cambios y un commit para confirmar los cambios guardados.

```
C:\Users\Danie\Desktop\S2R3\repo1>git add .
C:\Users\Danie\Desktop\S2R3\repo1>git commit -m "Fusion de la rama amster y la version1"
[master c9226c1] Fusion de la rama amster y la version1
C:\Users\Danie\Desktop\S2R3\repo1>git status
On branch master
Your branch is ahead of 'origin/master' by 3 commits.
(use "git push" to publish your local commits)

nothing to commit, working tree clean
```

Fig. 83. Finalmente guardo los cambios, confirmo dichos cambios y compruebo que se haya realizado correctamente.

- f) Sube la rama master actualizada al servidor. Adjunta captura de pantalla del resultado del comando.

```
C:\Users\Danie\Desktop\S2R3\repo1>git push origin master
Enumerating objects: 10, done.
Counting objects: 100% (10/10), done.
Delta compression using up to 12 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (6/6), 592 bytes | 592.00 KiB/s, done.
Total 6 (delta 1), reused 0 (delta 0), pack-reused 0
To C:\Users\Danie\Desktop\S2R2-remoto
c983747..c9226c1 master -> master
```

Fig. 84. Se sube el repositorio actual al remoto.

5) Vuelve de nuevo a la carpeta S2R2 y realiza las siguientes acciones:

```
C:\Users\Danie\Desktop\S2R3\repo1>cd C:\Users\Danie\Desktop\S2R2\repo1
```

Fig. 85. Volvemos al repositorio de la carpeta S2R2.

a) Obtén los cambios que hay en el repositorio remoto sin fusionarlos en la rama local. Adjunta captura de pantalla del resultado del comando utilizado.

```
C:\Users\Danie\Desktop\S2R2\repo1>git fetch origin
remote: Enumerating objects: 10, done.
remote: Counting objects: 100% (10/10), done.
remote: Compressing objects: 100% (3/3), done.
remote: Total 6 (delta 1), reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (6/6), 572 bytes | 44.00 KiB/s, done.
From C:\Users\Danie\Desktop\S2R2-remoto
c983747..c9226c1 master -> origin/master
```

Fig. 86. Se indica que ha conectado con el repositorio remoto y que ha detectado que existe un cambio.

b) Actualiza la rama master local con el contenido de la rama master del repositorio remoto. Adjunta captura de pantalla del resultado del comando utilizado.

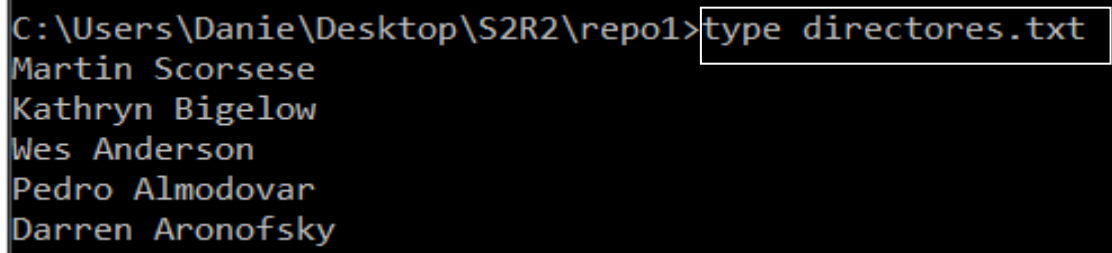
Hay dos opciones de realizar esta acción, ambas igual de buenas. Por un lado, se puede usar `git pull` o se puede hacer de forma manual con `git merge origin/master`. Para que el segundo modo funcione adecuadamente tiene que posicionarte en la rama que quieres actualizar los cambios, luego le indicas que desde el repositorio origin actualiza la rama master.

```
C:\Users\Danie\Desktop\S2R2\repo1>git checkout master
Switched to branch 'master'
Your branch is behind 'origin/master' by 3 commits, and can be fast-forwarded.
(use "git pull" to update your local branch)

C:\Users\Danie\Desktop\S2R2\repo1>git merge origin/master
Updating c983747..c9226c1
Fast-forward
 directorios.txt | 11 +++++-----
 1 file changed, 5 insertions(+), 6 deletions(-)
```

Fig. 87. Al cambiarme a la rama master salta un aviso indicando que estoy tres pasos atrás. Se le indica que actualice la rama master y ahora ya están la información actualizada con la del repositorio remoto.

c) Comprueba que aparecen los 5 nombres de directores esperados.



```
C:\Users\Danie\Desktop\S2R2\repo1>type directores.txt
Martin Scorsese
Kathryn Bigelow
Wes Anderson
Pedro Almodovar
Darren Aronofsky
```

*Fig. 88. Comprobamos el contenido del archivo directores.txt, comprobando así que la información se ha actualizado correctamente.*

### 3. GitHub

- 1) Crea una cuenta en GitHub.

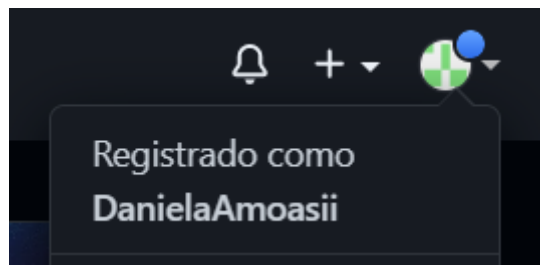


Fig. 89. Nombre de usuario de GitHub.

- 2) Añade tu dirección de correo de educación.

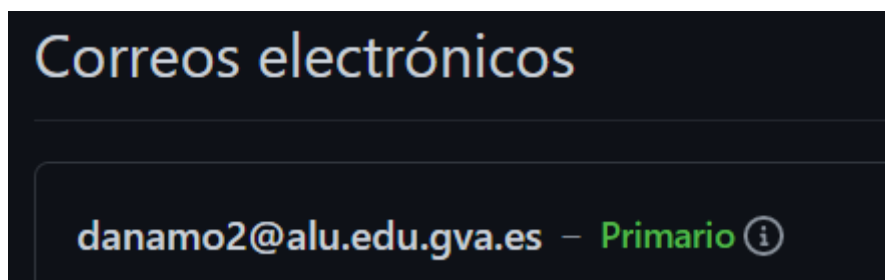


Fig. 90. Uso del email educativo.

- 3) Solicita un descuento para uso educativo: <https://education.github.com/pack>.

Hecho.

- 4) Haz un fork del repositorio localizado en la siguiente url: <https://github.com/curso-github-cefire/sesion3-practica>. A partir de este momento todas las tareas que se indican se deben realizar en tu repositorio (el que has clonado mediante el fork).

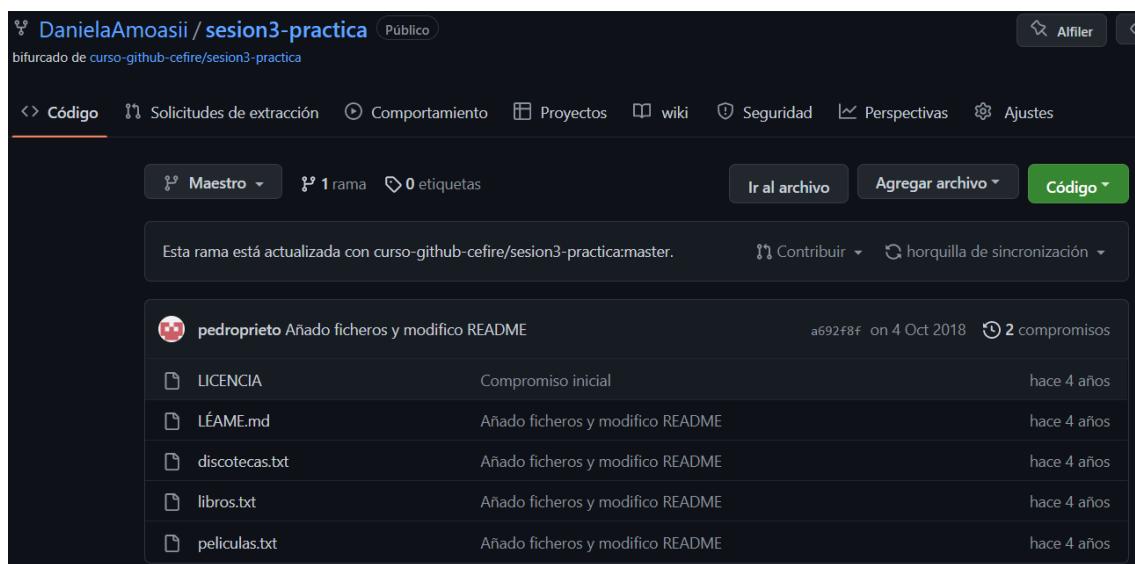


Fig. 91. Copia realizada mediante un fork a mi cuenta.

- a) Realiza un primer commit para poner tu nombre y apellidos en el fichero README.md

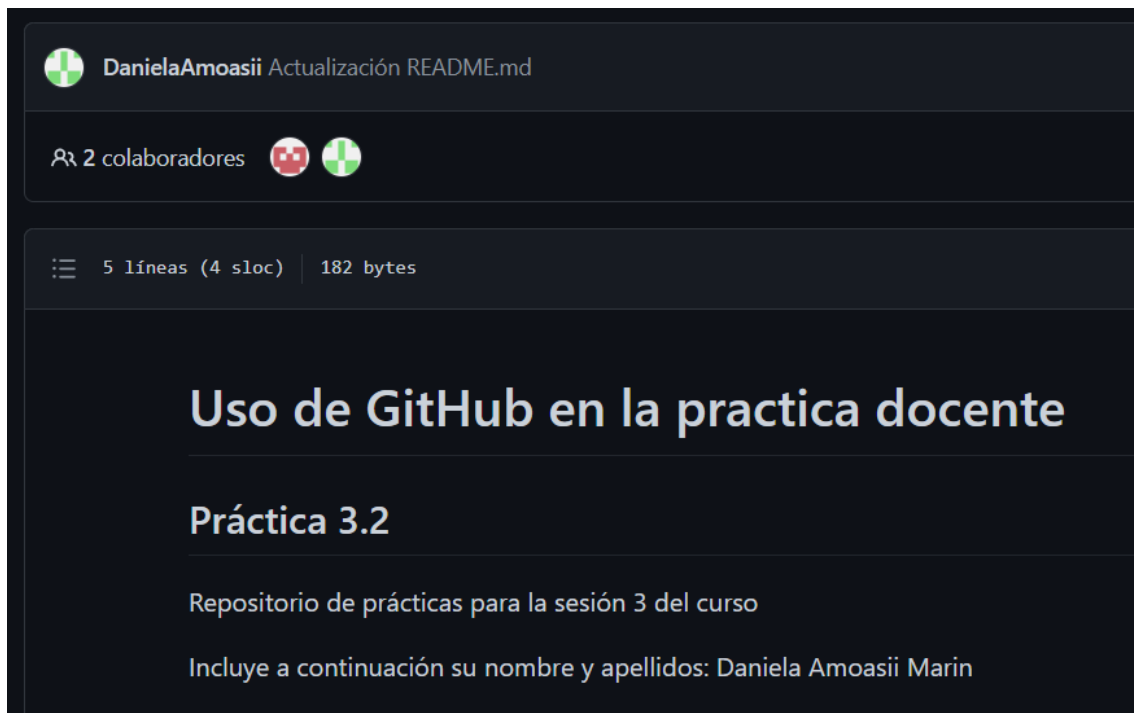


Fig. 92. Se ha añadido mi nombre al archivo.

- b) Crea 3 issues con los siguientes títulos. Si no ves la pestaña de issues, actívala desde los ajustes (settings) del repositorio.



Fig. 93. Se han creado 3 issues.

- c) Crea una milestone denominada Tareas sesión 3-2 que contenga los 3 issues creados.

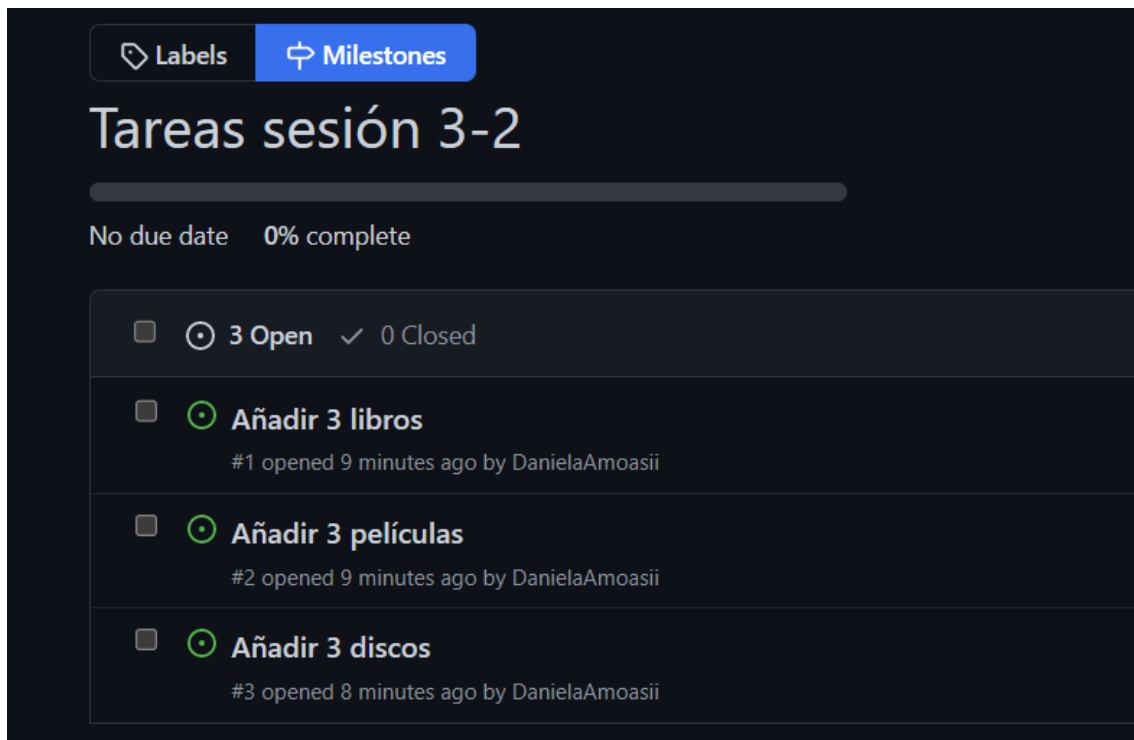


Fig. 94. Se ha creado un milestone con los tres issues.

- d) Modifica los ficheros correspondientes y realiza 3 commits para realizar cada una de las tareas que se indican en los issues. El mensaje del commit debe hacer que se cierren los issues correspondientes de manera automática.

Primero me bajare el repositorio a mi ordenador para poder trabajar en el y subir los cambios al repositorio.

```
C:\Users\Danie\Desktop\GIT_3_3>git clone https://github.com/DanielaAmoasii/sesion3-practica.git
Cloning into 'sesion3-practica'...
remote: Enumerating objects: 13, done.
remote: Counting objects: 100% (3/3), done.
remote: Compressing objects: 100% (3/3), done.
remote: Total 13 (delta 0), reused 0 (delta 0), pack-reused 10
Receiving objects: 100% (13/13), done.
```

Fig. 95. Se clona el repositorio remoto.

Una vez que se ha descargado el repositorio, modifíco los archivos añadiendo los elementos requeridos por el ejercicio, a continuación, compruebo el estado del repositorio y se indica que hay 3 ficheros modificados. Guardo los cambios y se hace un commit indicando que se cierren los 3 issues creados anteriormente. Por último, se subirá el repositorio al remoto y comprobaremos si los issues se han cerrado adecuadamente.

```

C:\Users\Danie\Desktop\GIT_3_3\sesion3-practica>git status
On branch master
Your branch is up to date with 'origin/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:   discos.txt
        modified:   libros.txt
        modified:   peliculas.txt

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

C:\Users\Danie\Desktop\GIT_3_3\sesion3-practica>git add .

C:\Users\Danie\Desktop\GIT_3_3\sesion3-practica>git commit -m "Issues Realizados Close #1 #2 #3"
[master 3fa46ac] Issues Realizados Close #1 #2 #3
 3 files changed, 9 insertions(+)

C:\Users\Danie\Desktop\GIT_3_3\sesion3-practica>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

```

Fig. 96. Se guardan los cambios, y también se confirma los cambios realizados.

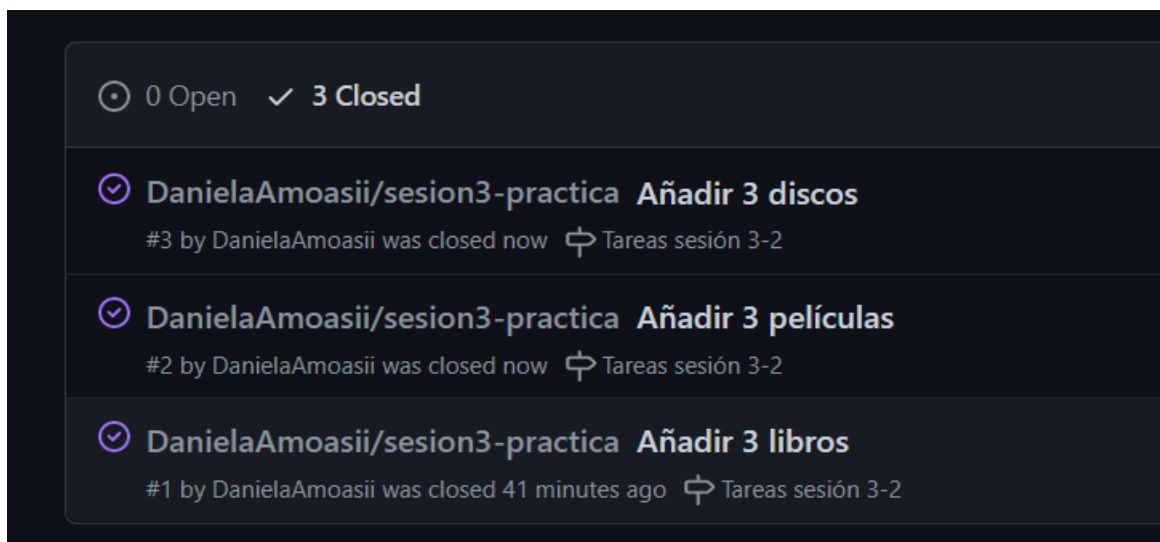


Fig. 97. Issues cerradas.

- e) Haz una captura de pantalla de los comandos que has utilizado para hacer los commits y subir los cambios a GitHub.

```

C:\Users\Danie\Desktop\GIT_3_3\sesion3-practica>git commit -m "Issus realizados Close #1 Close #2 Close #3"

```

Fig. 98. Código empleado para cerrar las issues.

- f) Incluye las capturas de pantalla en el repositorio dentro de la carpeta capturas. Añádelas también al repositorio de manera que queden guardadas en tu repositorio en GitHub.

Simplemente creo una carpeta llamada capturas dentro del repositorio y guardo el Word con las capturas en formato de PDF dentro de dicha carpeta.

- g) Realiza una pull request indicando en el mensaje que has completado la tarea.



## 4. Resumen de Comandos aprendidos

### Comandos clave

git --version.....	Comprobar si está instalado y cuál es la versión.
git config --global user.name "Nombre y apellido" .....	Configurar el nombre del usuario.
git config --global user.email correo@loquesea.com.....	Configurar el correo del usuario.
git config --global user.name.....	Mostrar cual es el nombre de usuario.
git config --global user.email.....	Mostrar cual es el correo del usuario.
git config --global credential.helper wincred.....	No pedir la contraseña al trabajar en remoto.
git init.....	Crear un repositorio git.
git status.....	Mostrar que ocurre en el repositorio.
git add archivo.extension.....	Guardar cambios realizados en un archivo del rep.
git add . .....	Guardar cambios de todos los archivos del rep.
git commit -m "Mensaje" .....	Confirmar los cambios guardados.
git log .....	Mostrar registro de confirmaciones (commit).
git log --graph .....	Mostrar registro de confirmación con una grafica.
git branch nombreRama.....	Crear una nueva rama.
git branch .....	Mostrar las ramas disponibles y en cual estas.
git checkout nombreRama .....	Cambiar a otra rama.
git merge nombreRama .....	Fusionar la rama en la cual estas con la que indicas.
git init --bare .....	Iniciar un repositorio remoto (No en uso).
git clone ruta nombreNuevoRepositorio .....	Clonar un repositorio.
git push nombreRepositorioRemoto nombreRama...	Enviar cambios al repositorio remoto.
git pull .....	Traer y fusionar cambios del remoto
git merge nomreRepositorioRemoto/nombreRama..	Traer y fusionar cambios del remto y de la rama.