**Práctica - Uso de Git y GitHub**

**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.

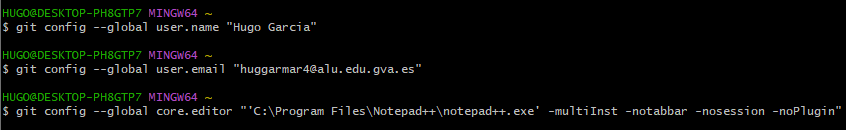
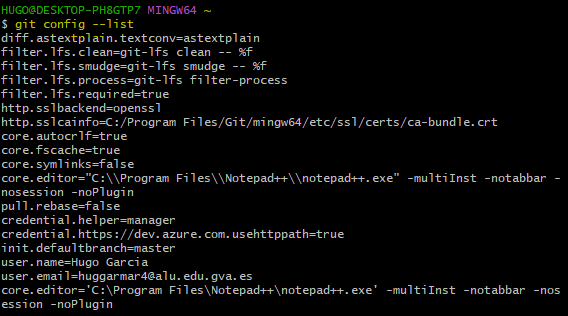
**git --version**

1. 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.

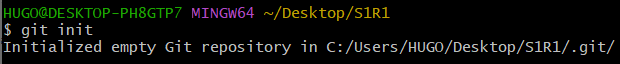
**git config --global user.name "Hugo Garcia"**

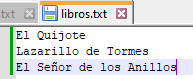
**git config --global user.email "huggarmar4@alu.edu.gva.es"**

**git config --global core.editor "'C:\Program Files\Notepad++\notepad++.exe' -multiInst -notabbar -nosession -noPlugin"**

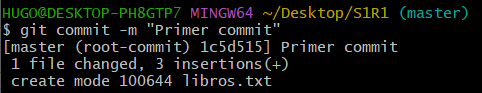
**git config --list**

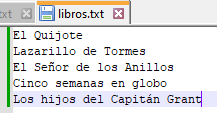
1. Crea una carpeta denominada S1R1. Realiza las siguientes acciones en ella:
2. Crea un repositorio Git.

**git init**

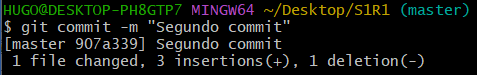
1. Crea un fichero denominado libros.txt. Añade tres títulos de libros cada uno en una línea distinta.
2. Haz un primer commit.

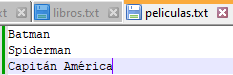
**git add libros.txt**

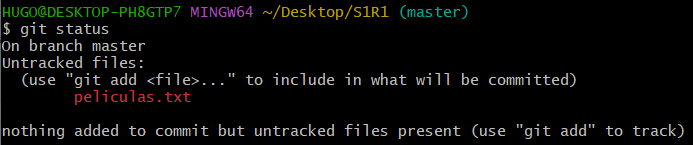
**git commit -m “Primer commit”**

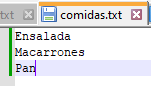
1. d) Añade dos libros al archivo libros.txt.
2. Haz un segundo commit.

**git add libros.txt**

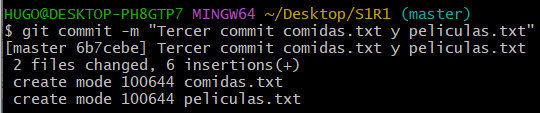
**git commit -m “Segundo commit”**

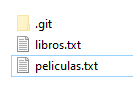
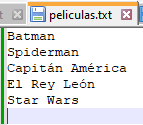
1. Crea un fichero denominado peliculas.txt. Añade tres títulos de películas a dicho archivo.
2. Haz una captura de pantalla del comando git status.

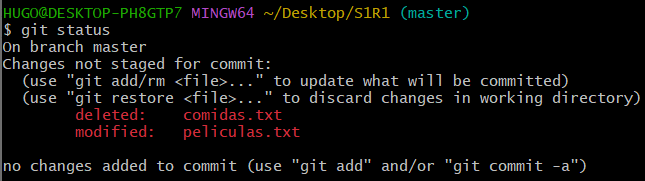
**git status**

1. Crea un fichero denominado comidas.txt. Añade tres nombres de comidas a dicho archivo.
2. Haz un tercer commit que incluya los archivos peliculas.txt y comidas.txt.

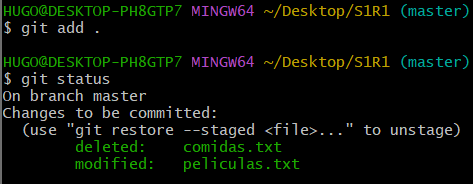
**git add .**

**git commit -m “Tercer commit comidas.txt y peliculas.txt”**

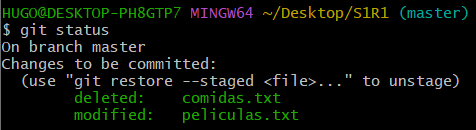
1. Elimina el archivo comidas.txt desde el navegador de archivos.
2. Añade dos películas más al archivo peliculas.txt.
3. Haz una captura de pantalla que muestre los cambios en el directorio de trabajo.

**git status**

1. Añade los cambios al área de preparación.

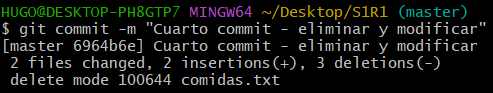
**git add.**

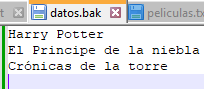
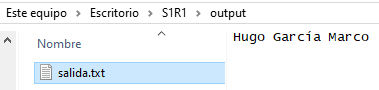
1. Haz una captura de pantalla del comando git status. Debe indicar que se ha borrado el archivo comidas.txt y que se ha modificado el archivo peliculas.txt.

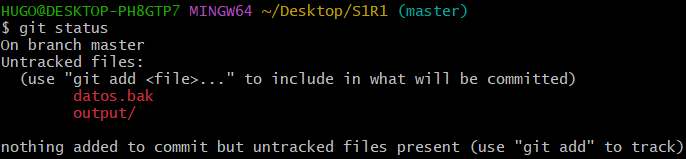
**git status**

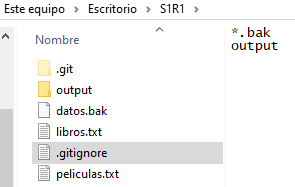
1. Haz un cuarto commit.

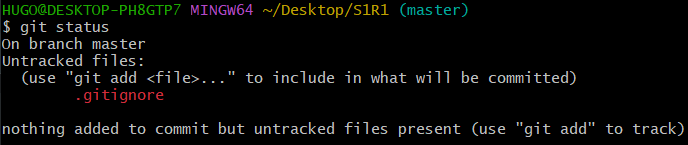
**git commit -m “Cuarto commit, eliminar y modificar”**



1. 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.
2. 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.
3. Haz una captura de pantalla del comando git status. Deben aparecer los archivos datos.bak y output/salida.txt como archivos nuevos (color rojo).

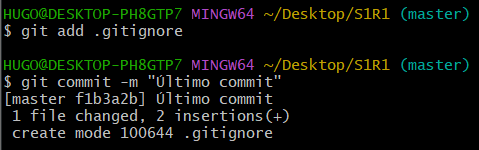
**git status**

1. Crea un archivo .gitignore para que los ficheros con extensión .bak y el contenido de la carpeta output/ no se incluyan en el repositorio.
2. 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.

**git status**

1. Haz un último commit para incluir el archivo .gitignore en el repositorio.

**git add .gitignore**

**git commit -m “Último commit”**

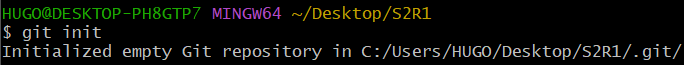
1. Haz una captura de pantalla que muestre el histórico de cambios del repositorio.

**git log**

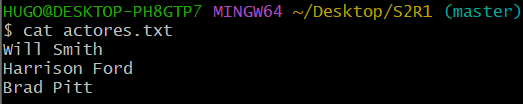
**Fundamentos de Git II**

1. Crea una carpeta denominada S2R1. Realiza las siguientes acciones en ella:
2. Crea un repositorio Git.

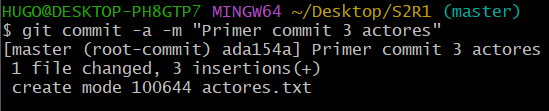
**git init**



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

**cat actores** 

1. Haz un primer commit.

**git commit -a -m “Primer commit 3 actores”**

1. Crea una rama denominada test.

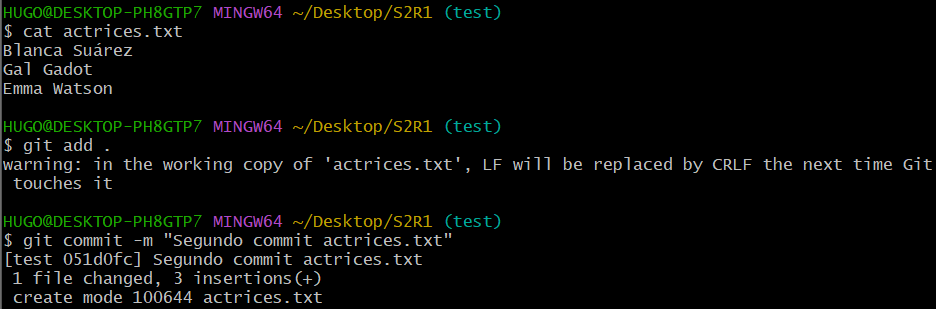
**git branch test**

1. Cambia a la rama test

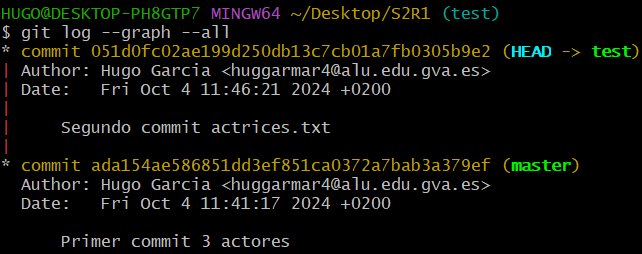
**git checkout test**

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

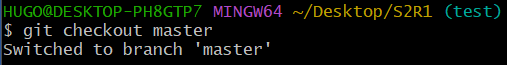
**git add .**

**git commit -m “Segundo commit actrices.txt”**

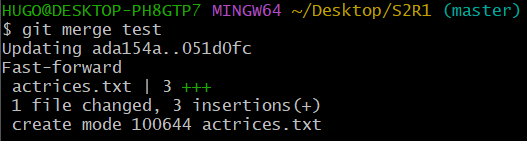
1. Haz una captura de pantalla del resultado del comando git log --graph --all.

**git log --graph --all**

1. Cambia a la rama master.

**git checkout master**

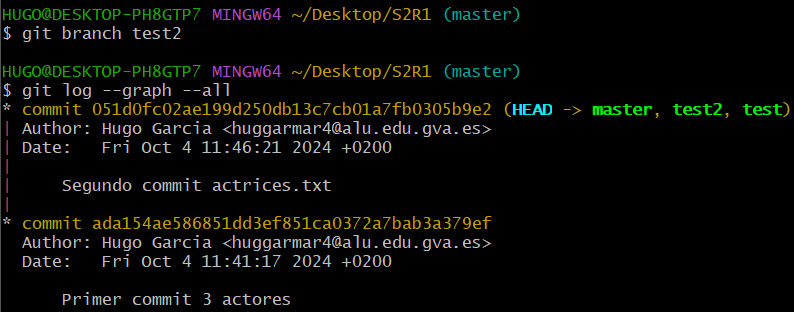
1. Incorpora los cambios de la rama test a la rama master. Haz una captura de pantalla de los comandos que has utilizado y de su resultado.

**git merge test**

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

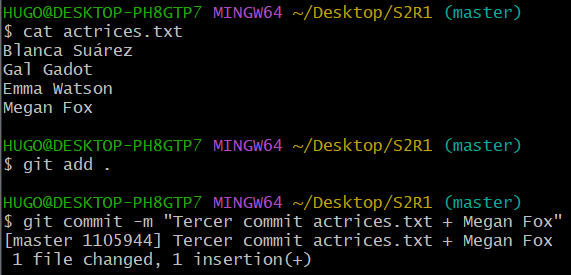
**git branch test 2**

**git log –graph --all**

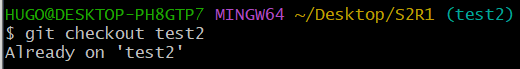


1. En la rama master, añade una actriz al fichero actrices.txt y haz un commit.

**git add .**

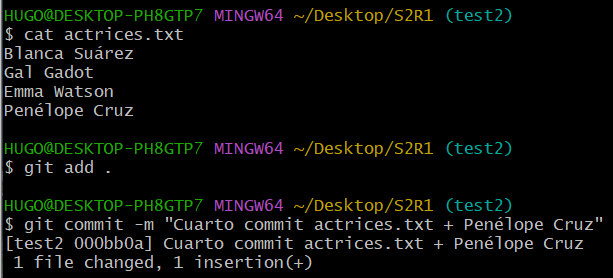
**git commit -m “Tercer commit”**

1. Cambia a la rama test2

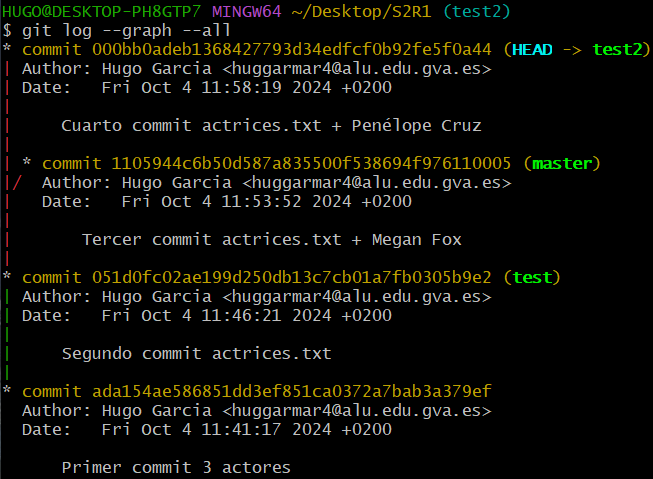
**git checkout test2**

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

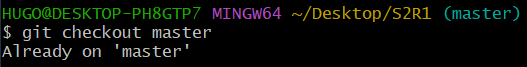
**git add .**

**git commit -m “Cuarto commit”**

1. 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.

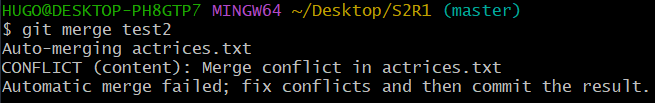
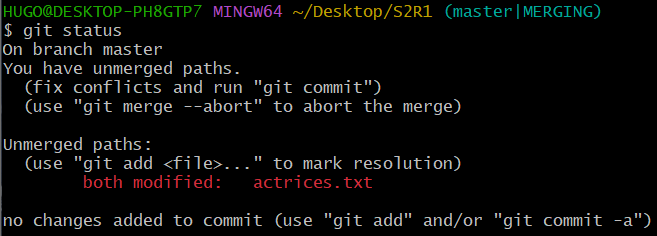
**git log --graph --all**

1. Cambia a la rama master

**git checkout master**

1. 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.

**git merge test2**

**git status**

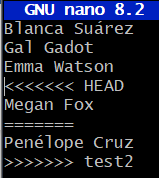
1. Resuelve el conflicto incorporando los dos nombres de actrices.

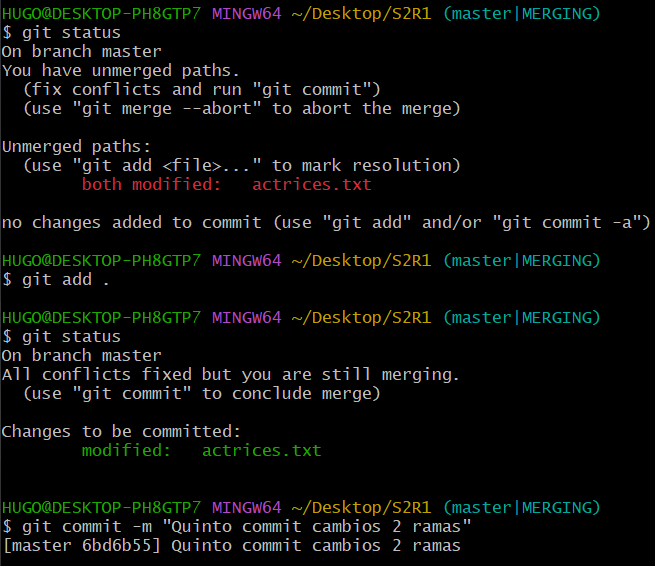
**Git status**

**git add .**

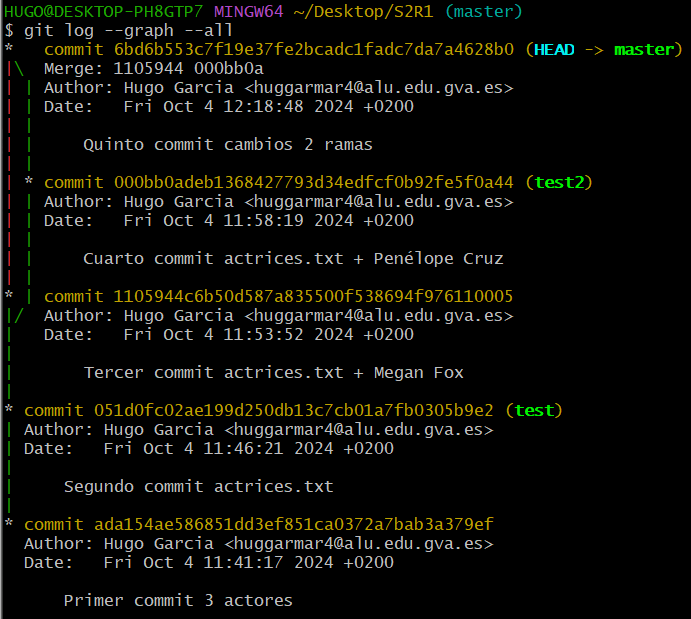
**git status**

**git commit -m “Quinto commmit”**

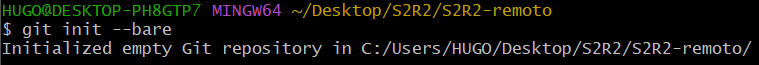




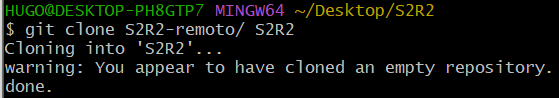
1. Haz una captura de pantalla del resultado del comando git log --graph --all. Observa que se ha creado un nuevo commit que integra los dos caminos anteriores.

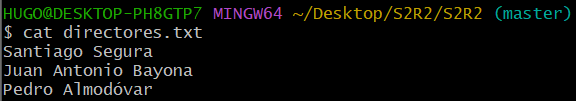
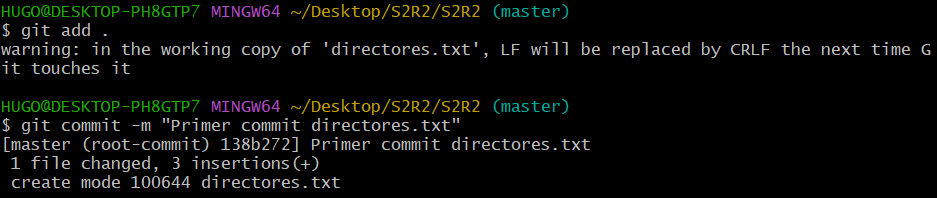
**git log --graph --all**

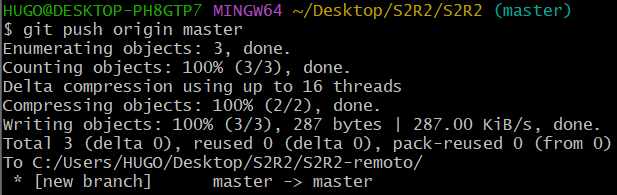
1. Crea una carpeta denominada S2R2-remoto. Inicializa un repositorio Git en su interior mediante el comando git init --bare. Esta carpeta se utilizará como repositorio remoto.

**git init --bare**

1. 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:

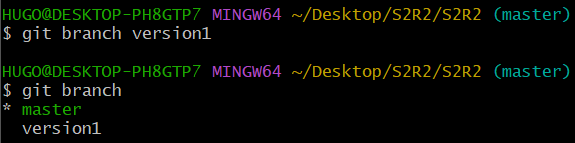
**git** **clone S2R2-remoto/ S2R2**

1. Crea un archivo denominado directores.txt. Añade el nombre de tres directores de cine.
2. Haz un commit.
3. Realiza un push al repositorio remoto. Adjunta captura de pantalla del resultado

**git push origin master**

1. Crea una rama denominada version1.

**git branch version1**

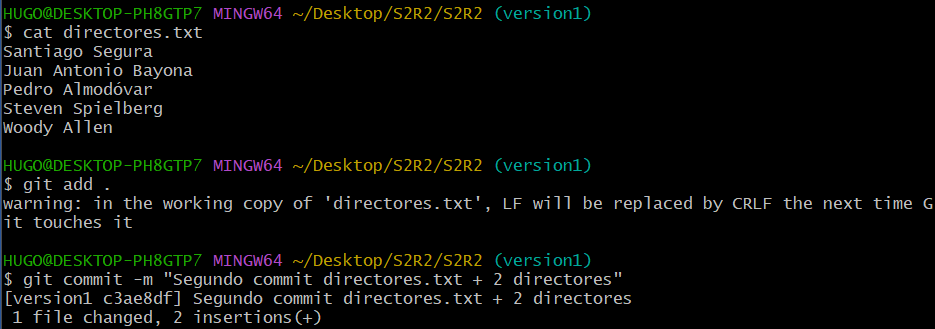
**git branch**

1. Cambia a la rama version1.

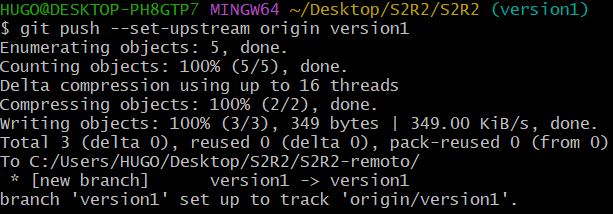
**git checkout version1**

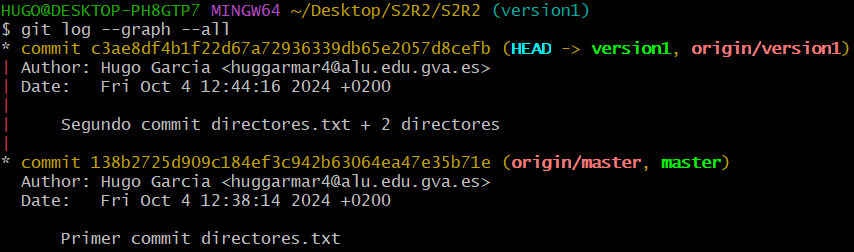
1. En la rama version1 añade el nombre de dos directores de cine más al archivo directores.txt y haz un commit de los cambios.

**git add .**

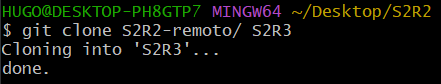
**git commit -m “Segundo commit”**

1. 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.

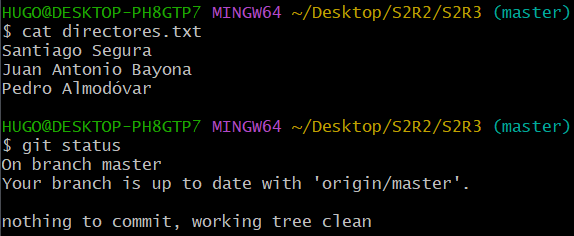
**git push --set-upstream origin version1**

**git log --graph --all**

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

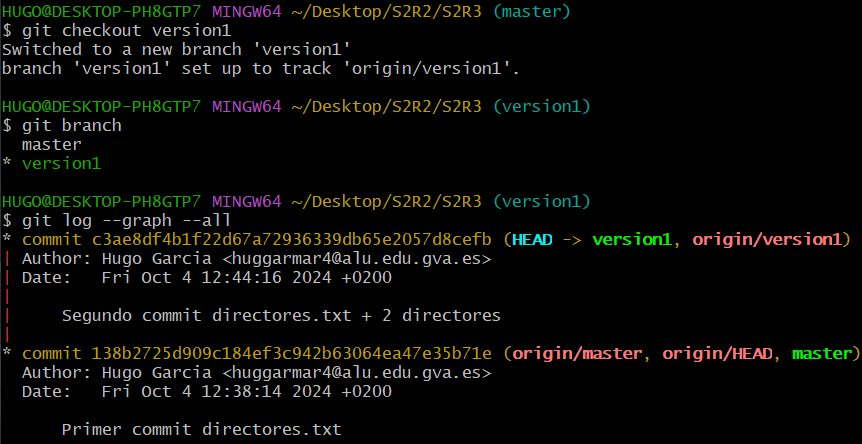
**git clone S2R2-remoto/ S2R3**

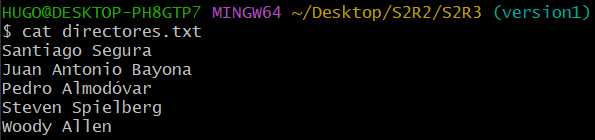
1. Muestra en la consola el contenido del fichero directores.txt y el resultado del comando git status. Debe mostrar tres directores.

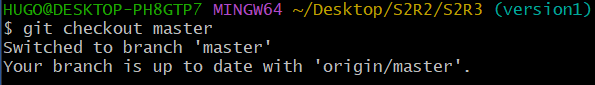
**git status** 

1. 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 creacion local y enlazado con su correspondiente remota.

**git checkout version1**

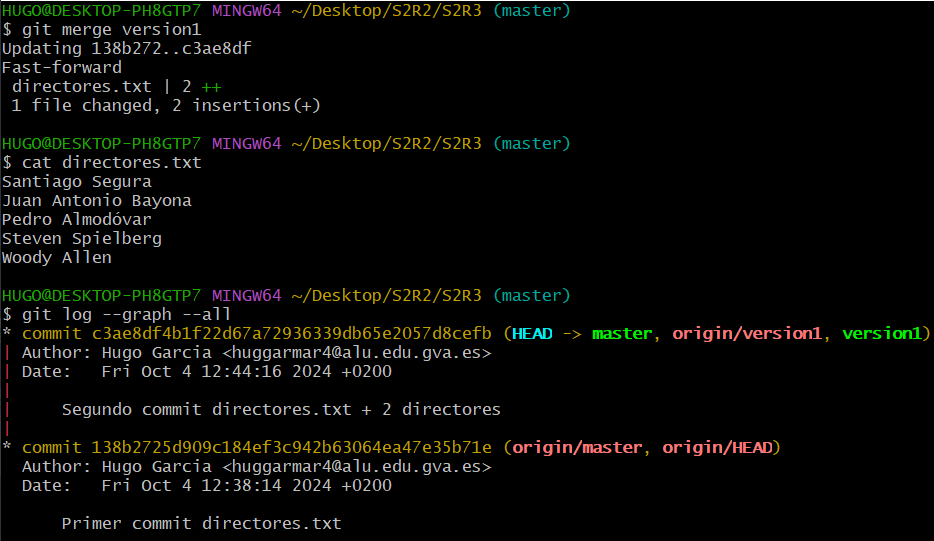
**git branch git log --graph --all**

1. Muestra el contenido del fichero directores.txt por la pantalla. Comprueba que se muestran los 5 nombres de directores esperados. Adjunta captura de pantalla.
2. Cambia a la rama master.

**git checkout master**

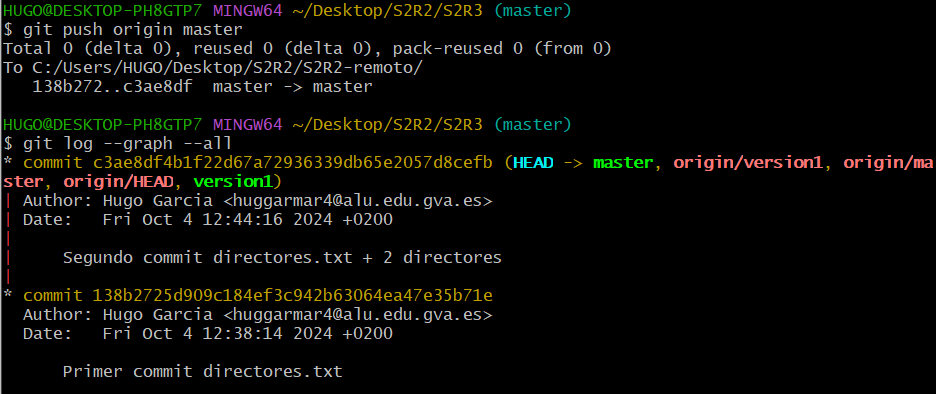
1. Incorpora los cambios de la rama version1 a la rama master.

**git merge version1**

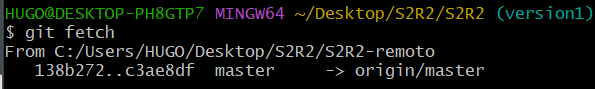
**git log --graph --all**

1. Sube la rama master actualizada al servidor. Adjunta captura de pantalla del resultado del comando.

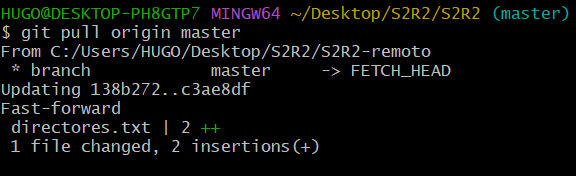
**git push origin master**

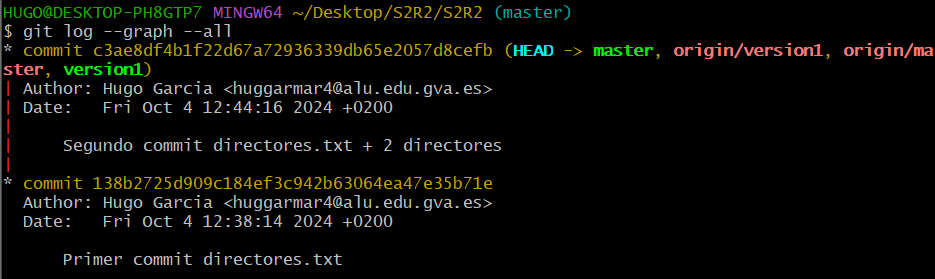
**git log --graph --all**

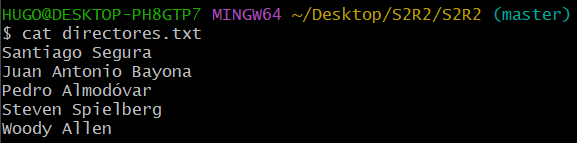
1. Vuelve de nuevo a la carpeta S2R2 y realiza las siguientes acciones:
2. Obtén los cambios que hay en el repositorio remoto sin fusionarlos en la rama local. Adjunta captura de pantalla del resultado del comando utilizado.

**git** **fetch**

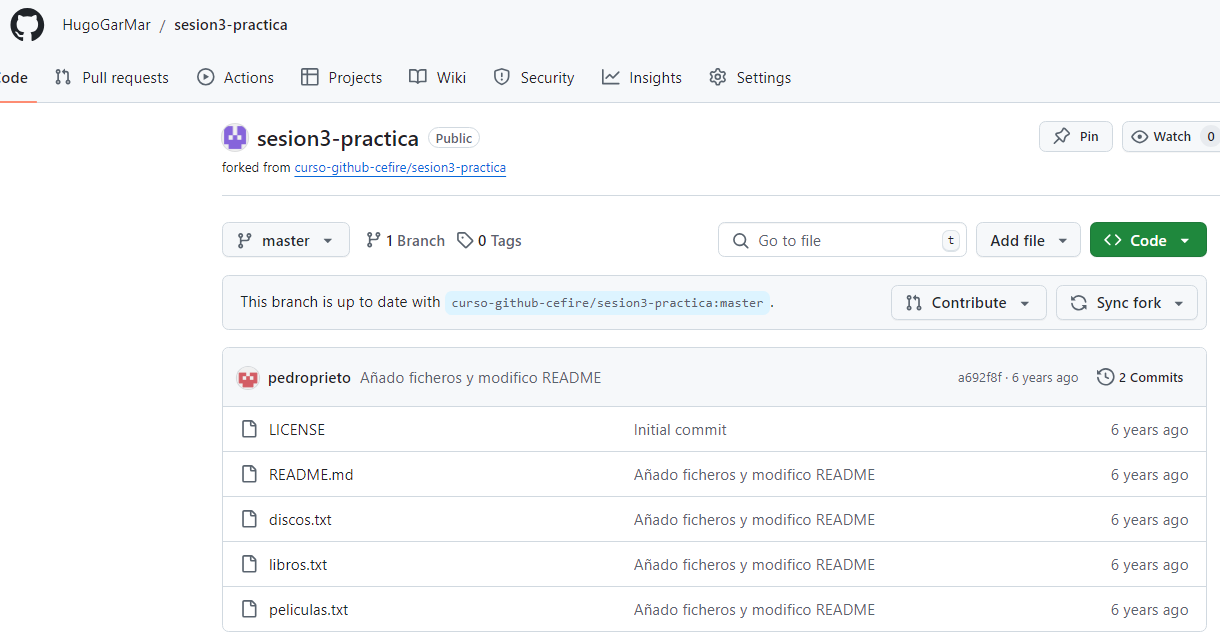
1. Actualiza la rama master local con el contenido de la rama master del repositorio remoto. Adjunta captura de pantalla del resultado del comando utilizado.

**git pull origin master**

**git log --graph --all**

1. Comprueba que aparecen los 5 nombres de directores esperados.

**4. GitHub**

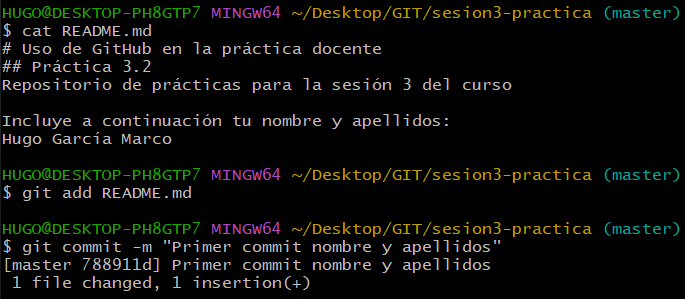
* 1. Crea una cuenta en GitHub
  2. Añade tu dirección de correo de educación
  3. Solicita un descuento para uso educativo: <https://education.github>. com/pack
  4. Haz un fork del repositorio localizado en la siguiente url: <https://github>. com/curso-github-cefire/sesion3-practica. A partir de este momento todas las tareas que se indican se deben realizar en tu repositorio (el que has clonado mediante el fork).

**git clone URL**

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

**git add README.md**

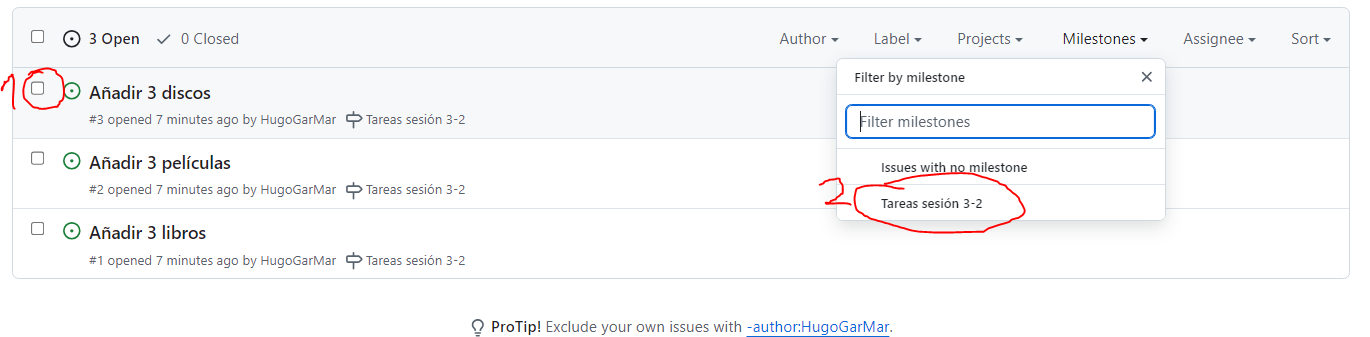
**git commit -m “Primer commmit”**

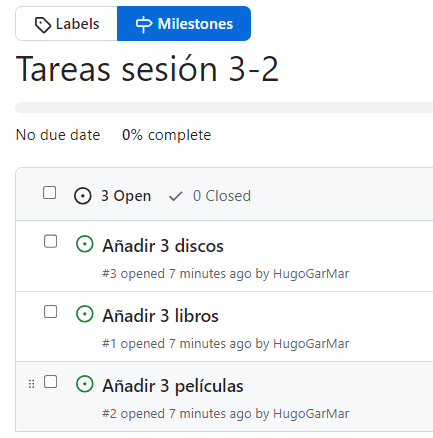


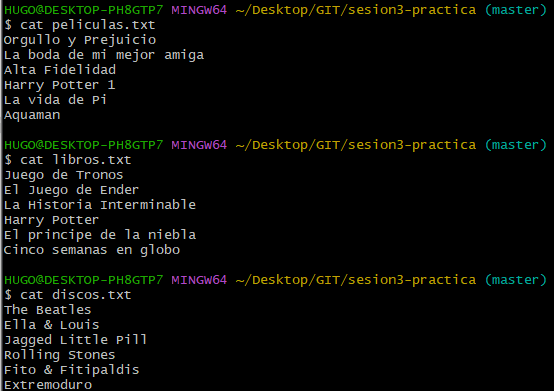
1. Crea 3 issues con los siguientes títulos. Si no ves la pestaña de issues, actívala desde los ajustes (settings) del repositorio.

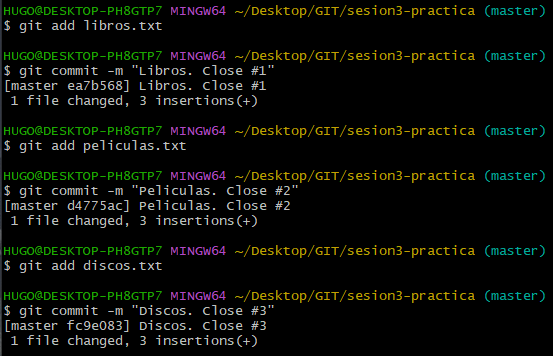
**Settings > General > Features > issues**

* Anadir 3 libros
* Anadir 3 películas
* Anadir 3 discos

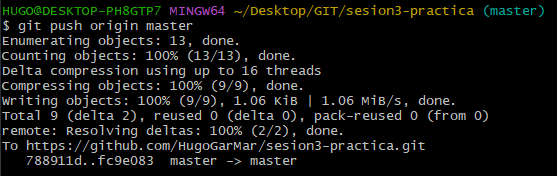
1. Crea una milestone denominada Tareas sesion 3-2 que contenga los 3 issues creados.



1. 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.

**git commit -m “Libros Close #1”**

1. Haz una captura de pantalla de los comandos que has utilizado para subir los cambios a GitHub.

**git** **push origin master**

1. 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.
2. Realiza una pull request indicando en el mensaje que has completado la tarea.

**5. Entrega de la tarea**

Crea un documento con las capturas pedidas (Fundamentos de Git I, Fundamentos de Git II y GitHub) y guárdalo en formato PDF.

Guarda dicho documento en la carpeta capturas del repositorio y sube los cambios a GitHub. La entrega de la tarea se realiza a través de la pull request en GitHub.