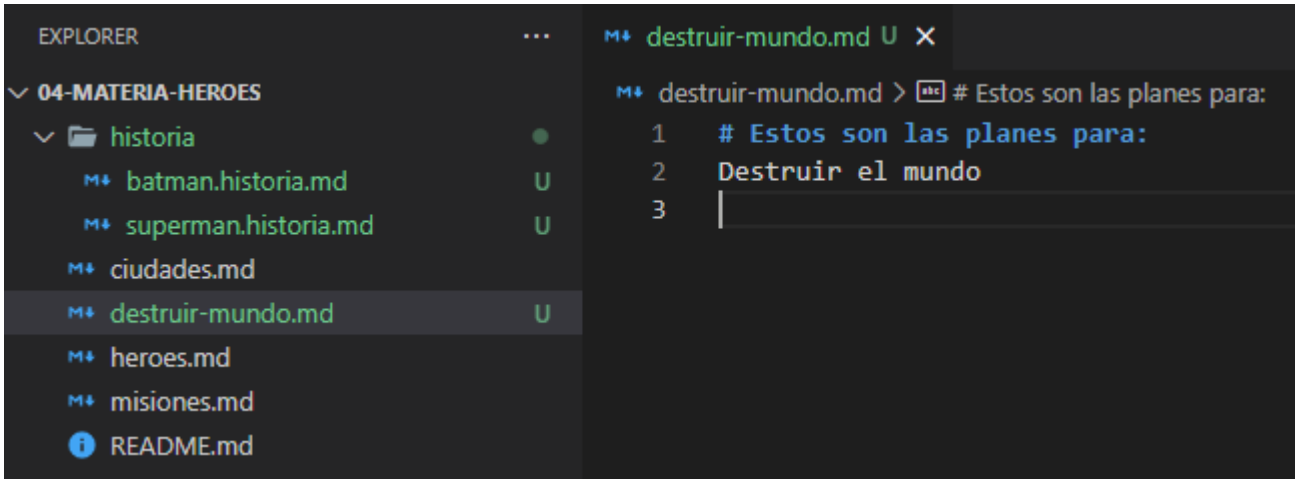
Práctica 05

Contenido

1 Cambiar el nombre y eliminar archivos con git

Creamos mediante visual studio un nuevo archivo llamado “destruir-mundo.md”:



Guardamos los cambios del archivo.

```
PS C:\NIEVES\DAW\ENDES\Práctica GIT\Materia-Heroes> git add .
PS C:\NIEVES\DAW\ENDES\Práctica GIT\Materia-Heroes> git commit -m "Des
truir-mundo añadido"
[main 6b616f0] Destruir-mundo añadido
 1 file changed, 3 insertions(+)
 create mode 100644 destruir-mundo.md
PS C:\NIEVES\DAW\ENDES\Práctica GIT\Materia-Heroes>
```

Lo subimos al escenario:

```
git add .
```

Hacemos commit:

```
git commit -m "Destruir-mundo añadido"
```

Vamos a cambiar el nombre a este archivo:

```
git mv destruir-mundo.md salvar-mundo.md
```

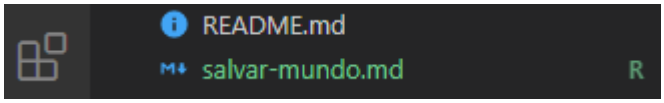
Aunque el comando git mv es para mover un archivo al moverlos a la misma ruta con diferente nombre lo que hacemos es renombrarlo.

Al renombrarlo vemos que si usamos el comando `git s` se muestra el archivo con una R que indica que fue renombrado:

```
PS C:\NIEVES\DAW\ENDES\Práctica GIT\Materia-Heroes> git mv destruir-mu
ndo.md salvar-mundo.md
PS C:\NIEVES\DAW\ENDES\Práctica GIT\Materia-Heroes> git commit -m "Des
truir mundo renombrado"
[main daf280e] Destruir mundo renombrado
 1 file changed, 0 insertions(+), 0 deletions(-)
 rename destruir-mundo.md => salvar-mundo.md (100%)
```

Visu

al studio también muestra una R de renombrado:



El archivo renombrado está además ya subido al stage listo para hacer commit:

```
git commit -m "Destruir mundo renombrado"
```

Podemos también borrar un archivo con un comando de git:

```
git rm salvar-mundo.md
```

Después de ejecutar este comando el archivo queda marcado con la letra D de borrado:

```
PS C:\NIEVES\DAW\ENDES\Práctica GIT\Materia-Heroes> git commit -m "sal
var mundo borrado"
[main b4403ad] salvar mundo borrado
 1 file changed, 3 deletions(-)
 delete mode 100644 salvar-mundo.md
PS C:\NIEVES\DAW\ENDES\Práctica GIT\Materia-Heroes> git status
On branch main
nothing to commit, working tree clean
```

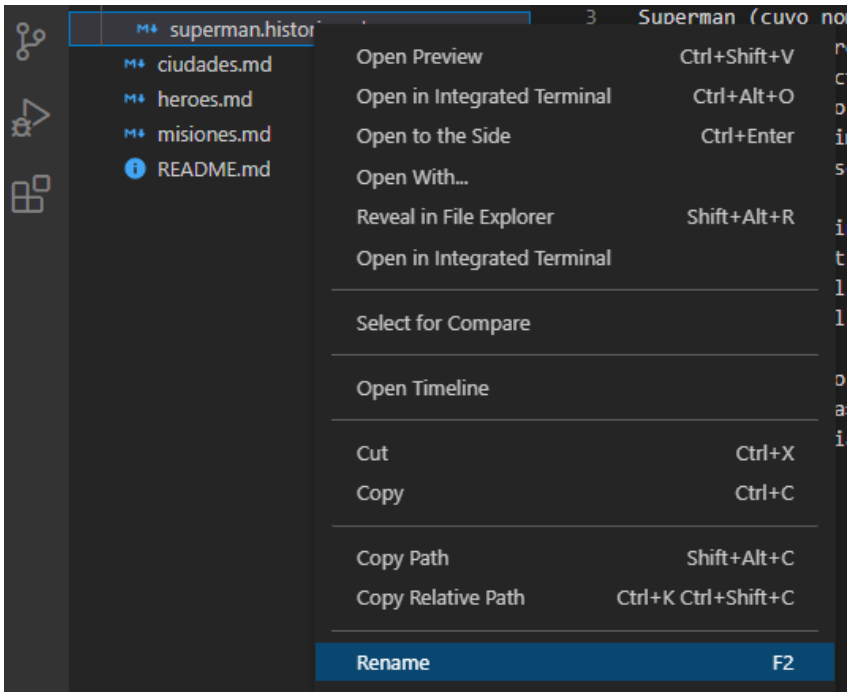
Para confirmar el borrado hacemos commit:

```
git commit -m "salvar mundo borrado"
```

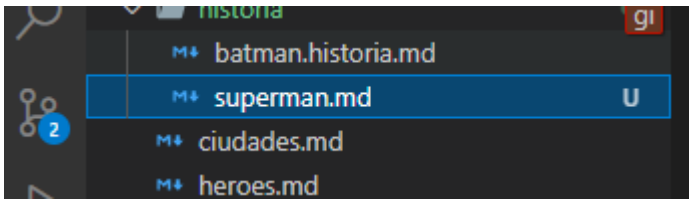
2 Cambiar el nombre y eliminar archivos fuera de git

2.1 Renombrar

Vamos a renombrar el archivo “superman.historia.md” desde la interfaz de visual studio. Pulsamos botón derecho sobre el archivo y elegimos la opción Rename:



Lo renombramos como “superman.md”



Vemos que el archivo renombrado queda marcado con la U de untracked (sin seguimiento) en vez de con la R de rename como sucedía en el apartado anterior al renombrar un archivo con el comando git.

Si hacemos un `git status` o `git s` vemos lo siguiente:

```
PS C:\NIEVES\DAW\ENDES\Práctica GIT\Materia-Heroes> git s
 D historia/superman.historia.md
?? historia/superman.md
```

Para git es como si hubiéramos borrado el archivo original que como vemos queda marcado con una D, y el archivo renombrado lo marca con dos interrogaciones considerando que es un archivo nuevo.

Si ahora subimos los cambios al stage:

```
git add .
```

Si ahora hacemos `git s`

```
PS C:\NIEVES\DAW\ENDES\Práctica GIT\Materia-Heroes> git add .
PS C:\NIEVES\DAW\ENDES\Práctica GIT\Materia-Heroes> git s
R  historia/superman.historia.md -> historia/superman.md
```

Vemos que ahora el sólo aparece el archivo renombrado marcado con la R de renamend (renombrado). Al hacer `git add .` git ha analizado los archivos y se ha dado cuenta de que son el mismo y se ha producido un renombrado.

Hacemos commit de los cambios:

```
git commit -m "Historia superman renombrada"
```

Hacemos un `git lg` para ver nuestro commits:

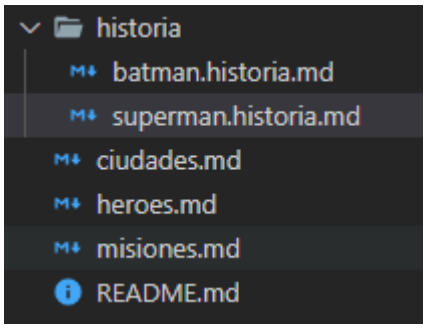
```
PS C:\NIEVES\DAW\ENDES\Práctica GIT\Materia-Heroes> git commit -m "His
toria superman renombrada"
[main cc61ab5] Historia superman renombrada
 1 file changed, 0 insertions(+), 0 deletions(-)
 rename historia/{superman.historia.md => superman.md} (100%)
```

Vamos a movernos al commit “Salvar-mundo borrado”:

```
git reset --hard b4403ad
```

```
PS C:\NIEVES\DAW\ENDES\Práctica GIT\Materia-Heroes> git lg
* cc61ab5 - (34 seconds ago) Historia superman renombrada - Nieves (HE
AD -> main)
* b4403ad - (5 minutes ago) salvar mundo borrado - Nieves
* daf280e - (7 minutes ago) Destruir mundo renombrado - Nieves
* 6b616f0 - (11 minutes ago) Destruir-mundo añadido - Nieves
* 7170f67 - (2 weeks ago) agregamos los heroes Linterna Verde y Robin
- Nieves
* 27b9dc3 - (2 weeks ago) agregada la historia de batman y superman -
Nieves
* 07470c5 - (2 weeks ago) heroes.agregado - Nieves
* 05914eb - (2 weeks ago) misiones.md agregado - Nieves
* 2010295 - (2 weeks ago) readme.md agregado - Nieves
PS C:\NIEVES\DAW\ENDES\Práctica GIT\Materia-Heroes> git reset --hard b
4403ad
HEAD is now at b4403ad salvar mundo borrado
```

Observamos que el archivo de historia de superman vuelve a tener su nombre original:



Si queremos deshacer el reset y volver al punto anterior hacemos:

```
git reset --hard cc61ab5
```

```
PS C:\NIEVES\DAW\ENDES\Práctica GIT\Materia-Heroes> git reflog
b4403ad (HEAD -> main) HEAD@{0}: reset: moving to b4403ad
cc61ab5 HEAD@{1}: commit: Historia superman renombrada
b4403ad (HEAD -> main) HEAD@{2}: commit: salvar mundo borrado
daf280e HEAD@{3}: commit: Destruir mundo renombrado
6b616f0 HEAD@{4}: commit: Destruir-mundo añadido
7170f67 HEAD@{5}: reset: moving to 7170f67
7170f67 HEAD@{6}: reset: moving to 7170f67
05914eb HEAD@{7}: reset: moving to 05914eb
07470c5 HEAD@{8}: reset: moving to 07470c5
1a38d87 HEAD@{9}: commit: historia agregada
07470c5 HEAD@{10}: reset: moving to HEAD
07470c5 HEAD@{11}: reset: moving to 07470c5
7170f67 HEAD@{12}: commit: agregamos los heroes Linterna Verde y Robin
27b9dc3 HEAD@{13}: reset: moving to 27b9dc3
9e0baef HEAD@{14}: commit: agregamos el heroe Linterna Verde
27b9dc3 HEAD@{15}: commit (amend): agregada la historia de batman y su
perman
5188561 HEAD@{16}: commit: historia agregado
07470c5 HEAD@{17}: commit: heroes.agregado
05914eb HEAD@{18}: commit: misiones.md agregado
2010295 HEAD@{19}: commit (initial): readme.md agregado
PS C:\NIEVES\DAW\ENDES\Práctica GIT\Materia-Heroes> git reset --hard c
c61ab5
HEAD is now at cc61ab5 Historia superman renombrada
```

El comando reflog nos permite ver el commit “Historia de superman renombrada” que no veríamos con git log:

```
PS D:\Git\04-Materia-Heroes> git reset --hard 3411dd0
HEAD is now at 3411dd0 Historia de superman renombrada
```

```
PS C:\NIEVES\DAW\ENDES\Práctica GIT\Materia-Heroes> git s
PS C:\NIEVES\DAW\ENDES\Práctica GIT\Materia-Heroes> git status
On branch main
nothing to commit, working tree clean
```

2.2 Borrar archivos

Borramos el archivo batman.hitoria.md desde visual studio pulsando botón derecho sobre él y eligiendo la opción Delete.

Si ejecutamos `git s` tras borrarlo:

```
PS C:\NIEVES\DAW\ENDES\Práctica GIT\Materia-Heroes> git s
 D historia/batman.historia.md
```

Vemos que el archivo aparece marcado con la D de deleted (borrado) pero no está subido al escenario como sucedía cuando borrábamos con un comando git.

Lo subimos al escenario:

```
git add .
```

Hacemos commit:

```
git commit -m "Historia de Batman borrado"
```

```
PS C:\NIEVES\DAW\ENDES\Práctica GIT\Materia-Heroes> git add .
PS C:\NIEVES\DAW\ENDES\Práctica GIT\Materia-Heroes> git commit -m "His
toria de Batman borrado"
[main 7b23275] Historia de Batman borrado
 1 file changed, 5 deletions(-)
 delete mode 100644 historia/batman.historia.md
PS C:\NIEVES\DAW\ENDES\Práctica GIT\Materia-Heroes> git s
PS C:\NIEVES\DAW\ENDES\Práctica GIT\Materia-Heroes>
```

3 Ignorar archivos que no deseamos

Vamos a crear algunos carpetas y archivos de prueba para luego configurar que git no les de seguimiento.

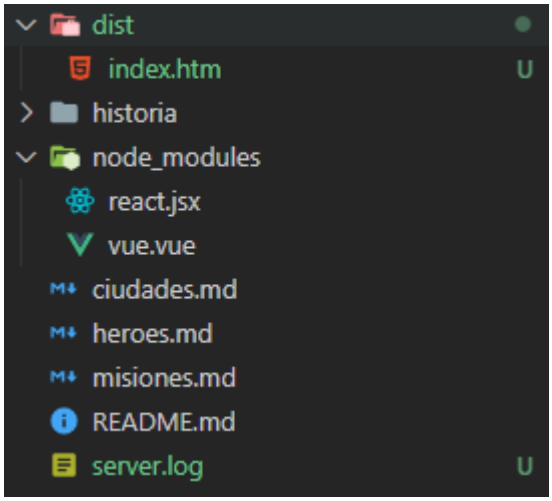
Creamos las carpetas:

- dist
- node_module

Y los siguientes archivos:

- server.log
- dist/index.html
- node_modules/react.jsx
- node_modules/vue.vue

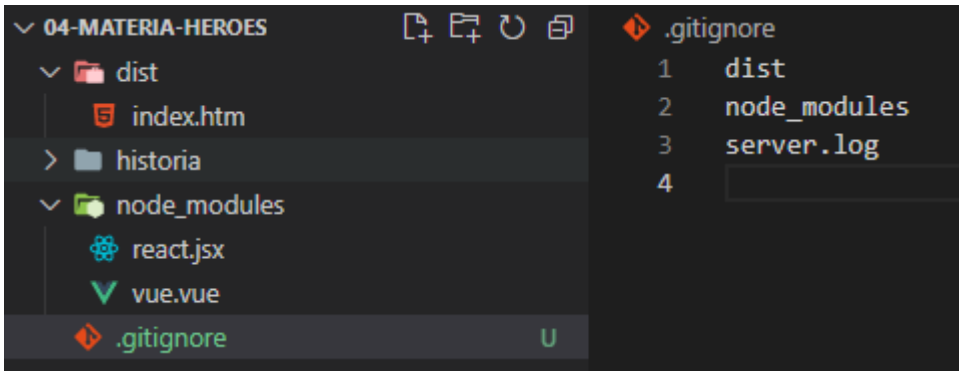
En los archivos podemos poner el texto que queramos o dejarlos vacíos.



Si hacemos `git s` vemos que git nos indica que los nuevos archivos no tienen seguimiento:

```
PS C:\NIEVES\DAW\ENDES\Práctica GIT\Materia-Heroes> git s
?? dist/
?? node_module/
?? server.log
```

Ahora lo que queremos es configurar que git ignore estos archivos porque no queremos darles seguimiento. Para ello creamos en la carpeta raíz del proyecto un archivo llamado **.gitignore** y dentro de este archivo escribiremos los nombres de archivos y directorios que queremos que git ignore. Podemos usar comodines para indicar los archivos:



Si después de escribir este archivo y guardar los cambios hacemos nuevamente `git s`:

```
PS C:\NIEVES\DAW\ENDES\Práctica GIT\Materia-Heroes> git s
?? .gitignore
```

Vemos que efectivamente git está ignorando todos los archivos de las dos carpetas nuevas y el archivo server.log , sólo aparece como archivo sin seguimiento el archivo que hemos creado **.gitignore**

A este archivo si le vamos a dar seguimiento ya que queremos que se guarden las configuraciones que hemos hecho:

```
git commit -m "Archivo .gitignore creado"
```

```
PS C:\NIEVES\DAW\ENDES\Práctica GIT\Materia-Heroes> git s
?? .gitignore
PS C:\NIEVES\DAW\ENDES\Práctica GIT\Materia-Heroes> git add .
PS C:\NIEVES\DAW\ENDES\Práctica GIT\Materia-Heroes> git commit -m "Arc
hivo .gitignore creado"
[main 8b22076] Archivo .gitignore creado
 1 file changed, 6 insertions(+)
 create mode 100644 .gitignore
PS C:\NIEVES\DAW\ENDES\Práctica GIT\Materia-Heroes> git s
PS C:\NIEVES\DAW\ENDES\Práctica GIT\Materia-Heroes>
```