Práctica 05

Contenido

1 Cambiar el nombre y eliminar archivos con git 2

2 Cambiar el nombre y eliminar archivos fuera de git 3

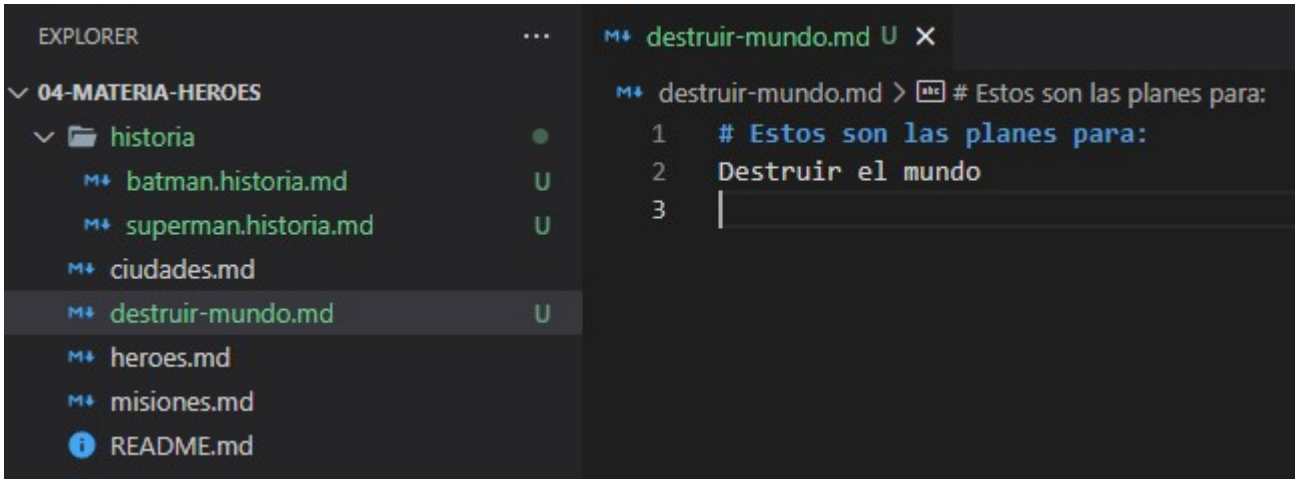
2.1 Renombrar 3

2.2 Borrar archivos 5

3 Ignorar archivos que no deseamos 6

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.

Lo subimos al escenario:

Hacemos commit:

Vamos a cambiar el nombre a este archivo:

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:

[Añade aquí una captura con la ejecución del comando git s]

Visual studio también muestra una R de renombrado:

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

Podemos también borrar un archivo con un comando de git:

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

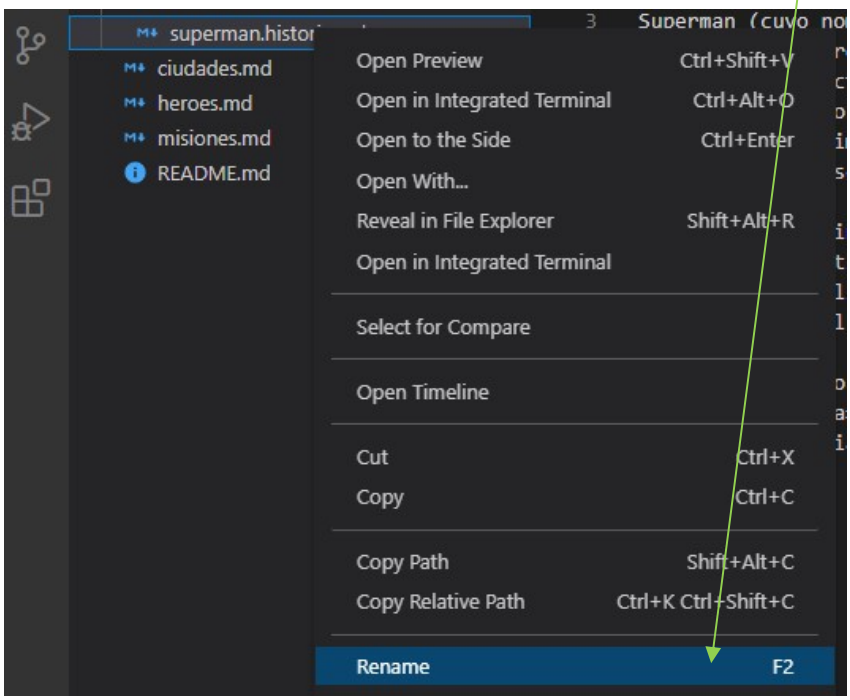
[Añade aquí una captura con la ejecución del comando anterior]

Para confirmar el borrado hacemos commit:

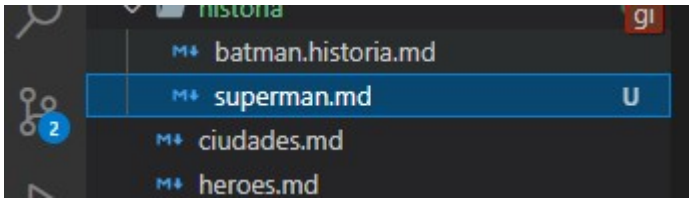
2 Cambiar el nombre y eliminar archivos fuera de git

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:

[Añade aquí una captura con la ejecución del comando anterior]

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`

[Añade aquí una captura con la ejecución del comando git s]

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:

```

```

Hacemos un `git lg` para ver nuestro commits:

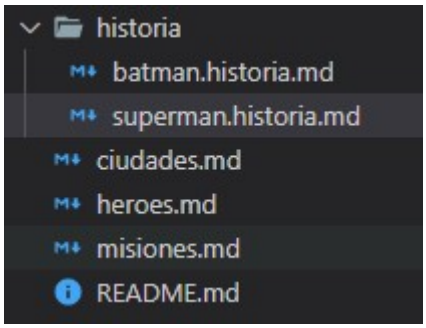
[Añade aquí una captura con la ejecución del comando git s]

Vamos a movernos al commit “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:

```
```

[Añade aquí una captura con la ejecución del comando git s]

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
```

[Añade aquí una captura con la ejecución del comando git s]

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:

[Añade aquí una captura con la ejecución del comando git s]

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:

```
```

[Añade aquí una captura con la ejecución del comando git s]

3 Ignorar archivos que no deseamos

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

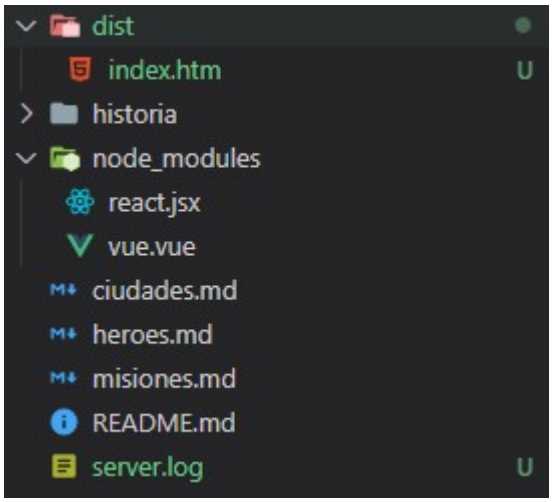
Creamos las carpetas:

- dist
- node_module

Y los siguientes archivos:

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

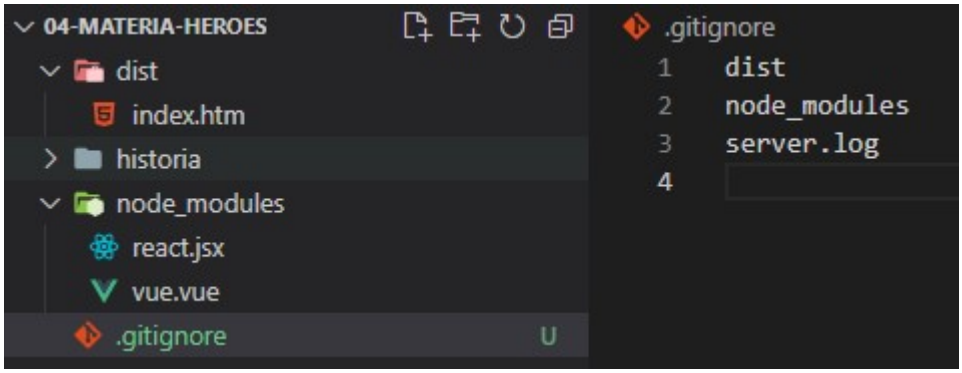
En los archivos podemos poner el texto que queramos o dejarlos vacíos.



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

[Añade aquí una captura con la ejecución del comando git s]

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`:

[Añade aquí una captura con la ejecución del comando git s]

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:

[Añade aquí una captura con la ejecución del comando git s]