

IMPORTANTE

Antes de comenzar a usar GIT, necesitamos saber cómo usar la consola

```
howtogeek@ubuntu: ~/Downloads
howtogeek@ubuntu:~/Downloads$ ls
file
howtogeek@ubuntu:~/Downloads$ mv file newfile
howtogeek@ubuntu:~/Downloads$ ls
newfile
howtogeek@ubuntu:~/Downloads$
```

```
Administrator: Command Prompt
Microsoft Windows [Version 6.3.9600]
(c) 2013 Microsoft Corporation. All rights reserved.

C:\Windows\system32>powercfg /energy
Enabling tracing for 60 seconds...
Observing system behavior...
Analyzing trace data...
Analysis complete.

Energy efficiency problems were found.

5 Errors
5 Warnings
25 Informational

See C:\Windows\system32\energy-report.html for details.

C:\Windows\system32>
```

```
Kenny — bash — 78x20
Last login: Mon Apr 13 11:46:14 on ttys000
Kennys-MacBook-Pro:~ Kenny$ mv ~/Documents/Test/TestFile-copy.rtf ~/Documents/
Test2/TestFile-copy.rtf
```

Comandos básicos de Consola

- **ls** (en Mac y Linux muestra los archivos de la carpeta en la que estamos ubicados)
- **dir** (en Windows muestra los archivos de la carpeta en la que estamos ubicados)
- **cd ..** (nos permite retroceder a una carpeta previa)
- **cd *nombre-carpeta*** (nos permite acceder a la carpeta descrita)
- **mkdir *algo*** (crea una carpeta con el nombre "**algo**")
- **touch *archivo.txt*** (crea un archivo de texto "**archivo.txt**")
- **rm *archivo.txt*** (elimina un archivo con el nombre "**archivo.txt**")
- **mv *nombre.txt otro.txt*** (cambia el nombre "**archivo.txt**" a "**otro.txt**")
- **clear** (limpia todo lo que hayamos escrito en la consola / Mac y Linux)
- **cls** (limpia todo lo que hayamos escrito en la consola / Windows)

Práctica básica de consola

1. Con la consola, llegar hasta la carpeta *Escritorio / Desktop* de nuestra máquina.
2. Una vez allí, crear una carpeta llamada "**test-consola**".
3. Ingresar a la carpeta recién creada.
4. Una vez dentro, crear dos archivos, uno llamado "**prueba.html**" y otro llamado "**home.html**".
5. Verificar en consola que hayan sido creados correctamente.
6. ¡Opps! Cometimos un error y creamos un archivo que no era, ahora tenemos que borrar por consola el archivo "**prueba.html**".
7. ¡Opps! Cometimos otro error, y tendremos que cambiar el nombre del archivo "**home.html**" a "**index.html**".
8. Finalmente, probar el siguiente comando "**atom .**" ¿Qué sucedió?, *atentís*, puede no hacer nada en windows.

GIT

■ **¿CÓMO VEMOS Y
COMPARTIMOS
NUESTROS
ARCHIVOS
DESDE
CUALQUIER
LUGAR?**

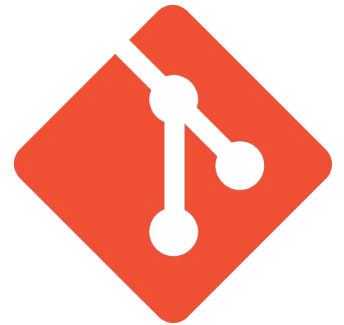


**¿PODEMOS
COMPARTIR
ASÍ NUESTROS
PROYECTOS?**



INTRO

Software para control de versiones multiplataforma! (SCM)



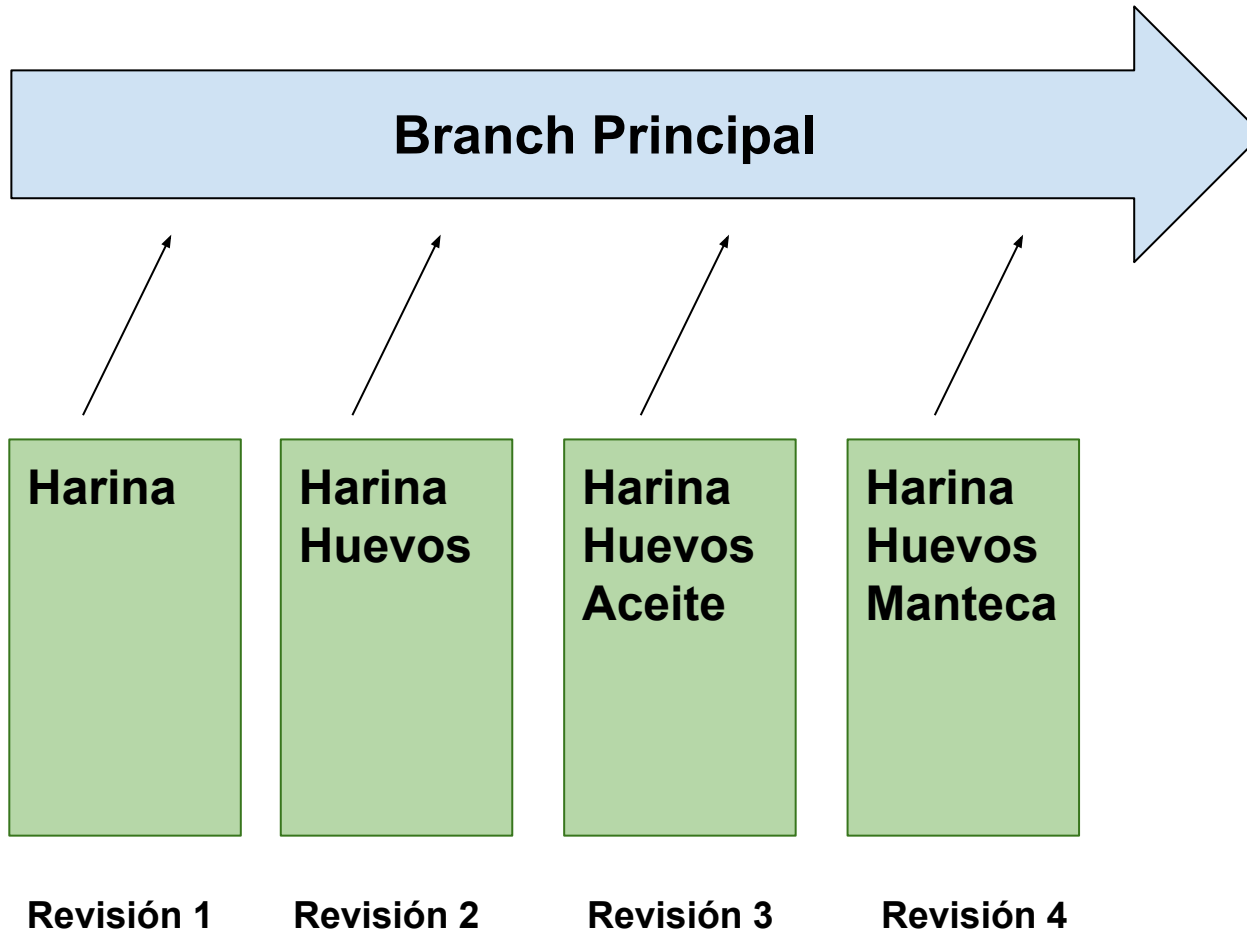


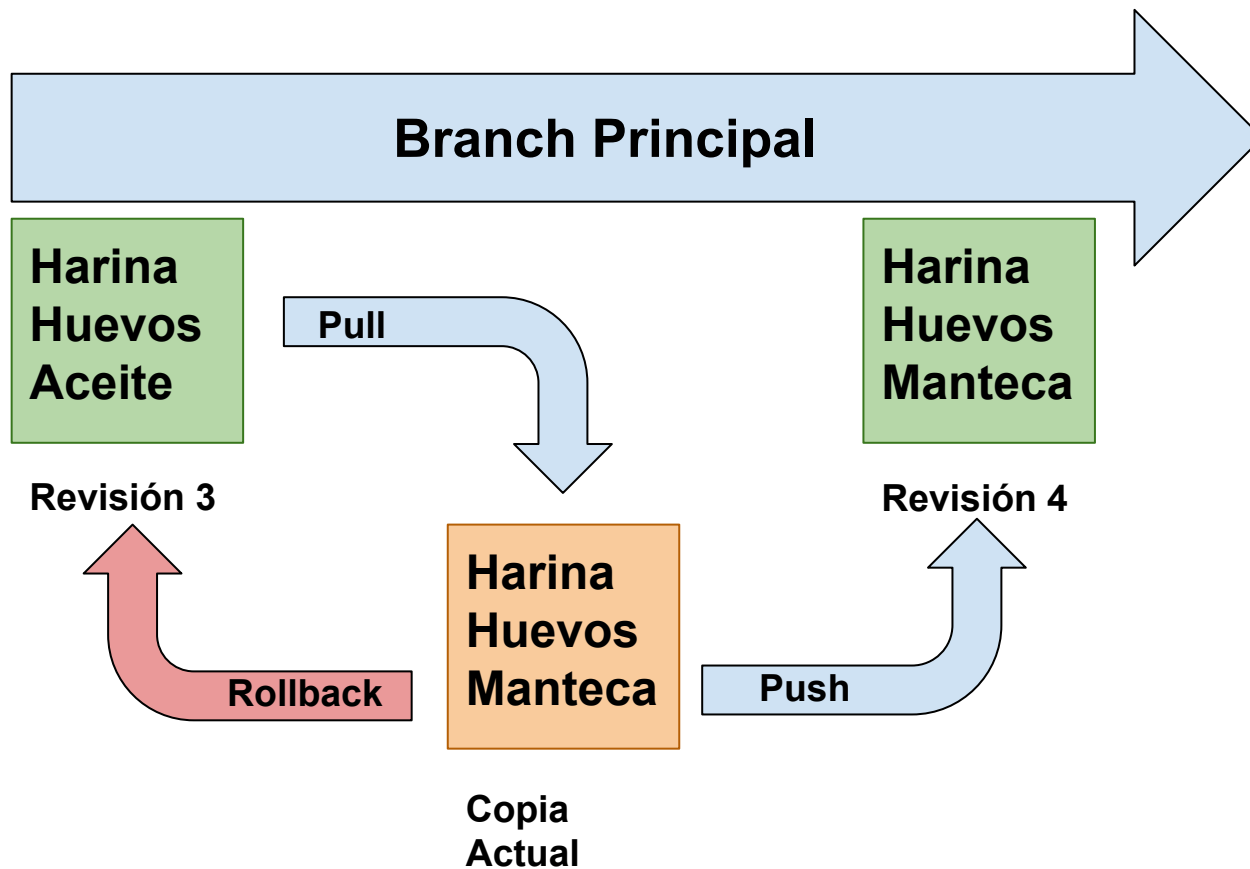
git

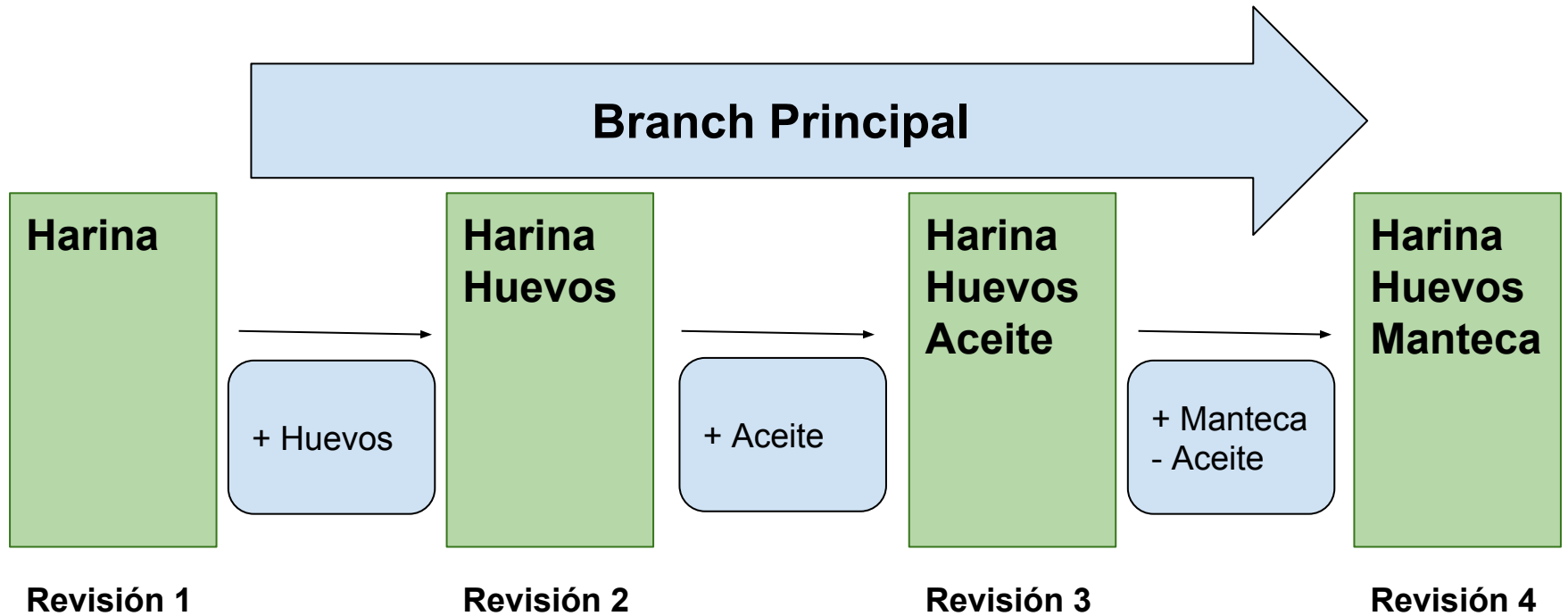


■ GIT | Características

- Backups y recuperación de archivos:
 - Undo a corto plazo.
 - Undo a largo plazo.
- Sincronización:
 - Seguimiento de cambios.
 - Seguimiento de autoría.
 - Resolución de conflictos.
- Branching and merging.
- Forking y Pull Requests.
- Espacio de pruebas.







Revisión 4

**Harina
Huevos
Aceite**

Revisión 6

**Harina
Huevos**

Nuevas Funcionalidades

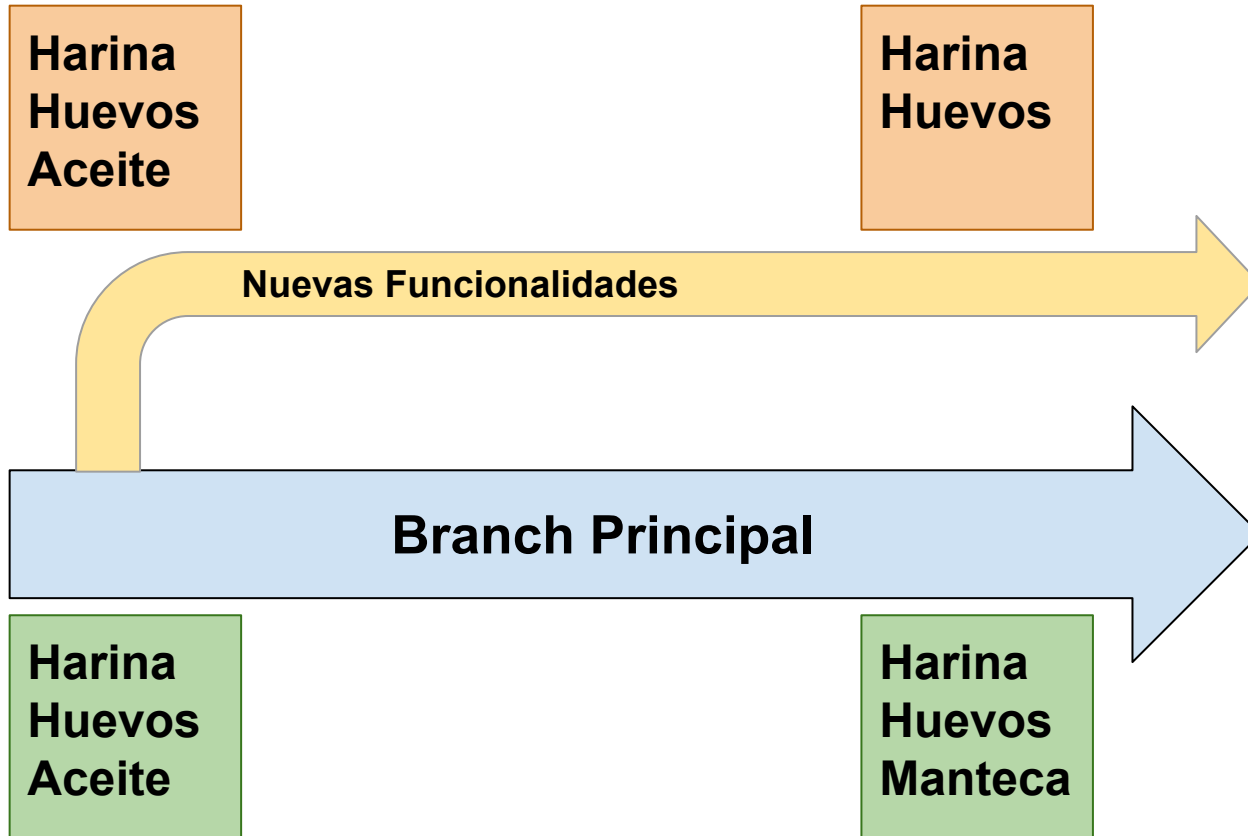
Branch Principal

**Harina
Huevos
Aceite**

Revisión 3

**Harina
Huevos
Manteca**

Revisión 5



Revisión 4

**Harina
Huevos
Aceite**

Revisión 6

**Harina
Huevos
Café**

Nuevas Funcionalidades

Branch Principal

**Harina
Huevos
Aceite**

Revisión 3

**Harina
Huevos
Manteca**

Revisión 5



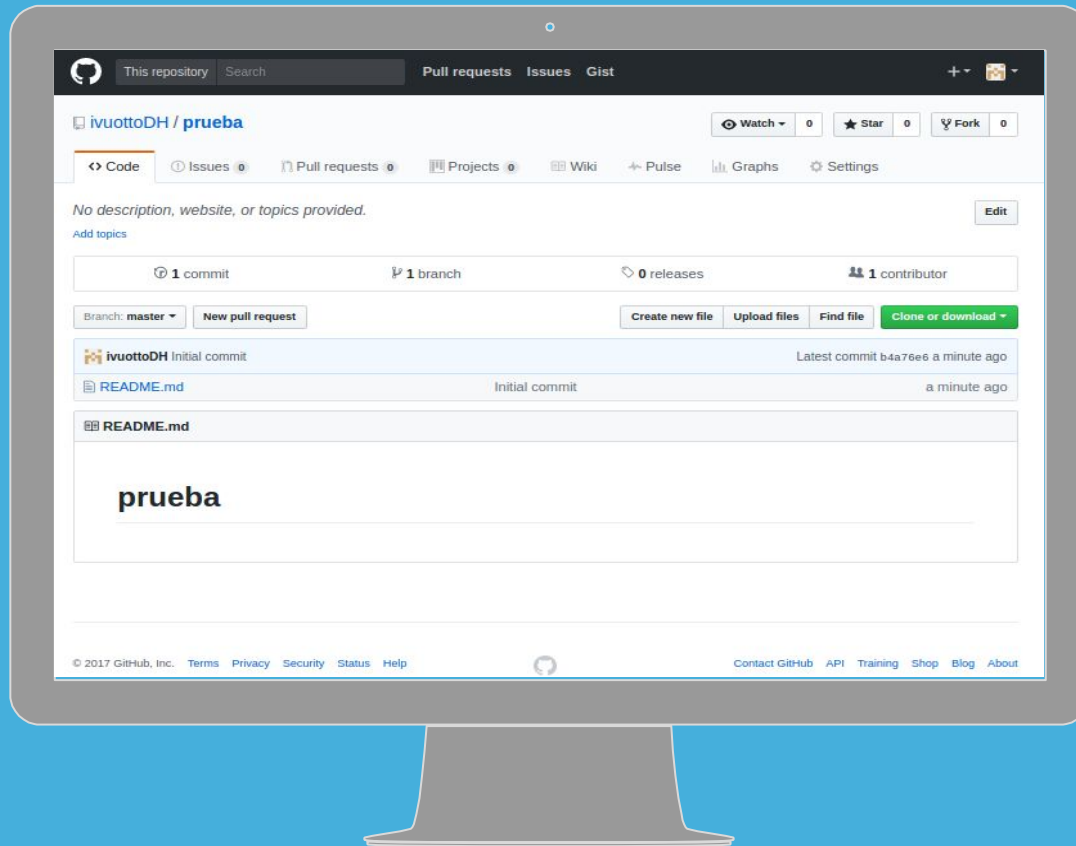
git

En nuestras máquinas.



GitHub

En la nube.

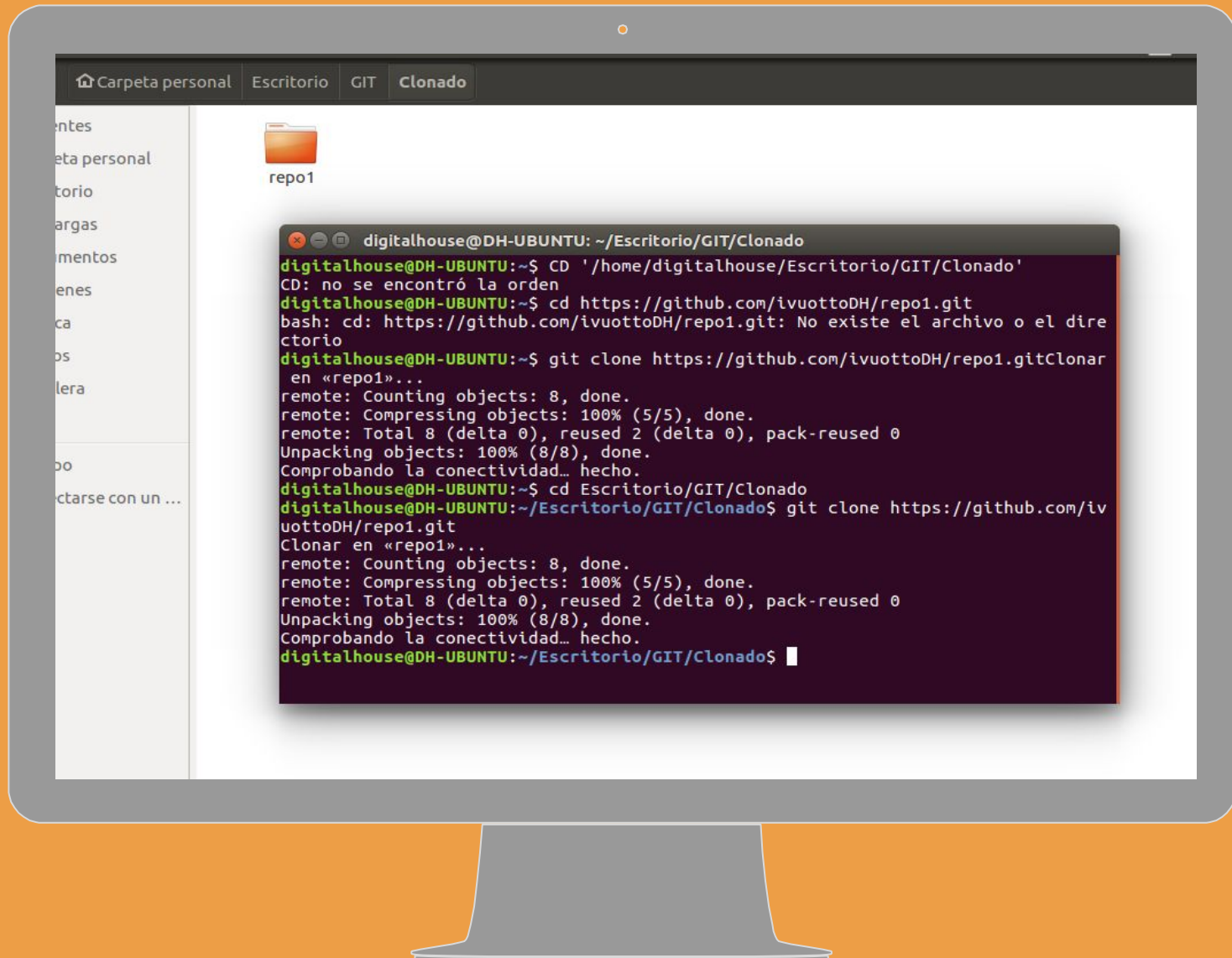


PRÁCTICA!

BAJAR GIT A NUESTRAS MÁQUINAS



CREAR REPOSITORIO LOCAL (git init)

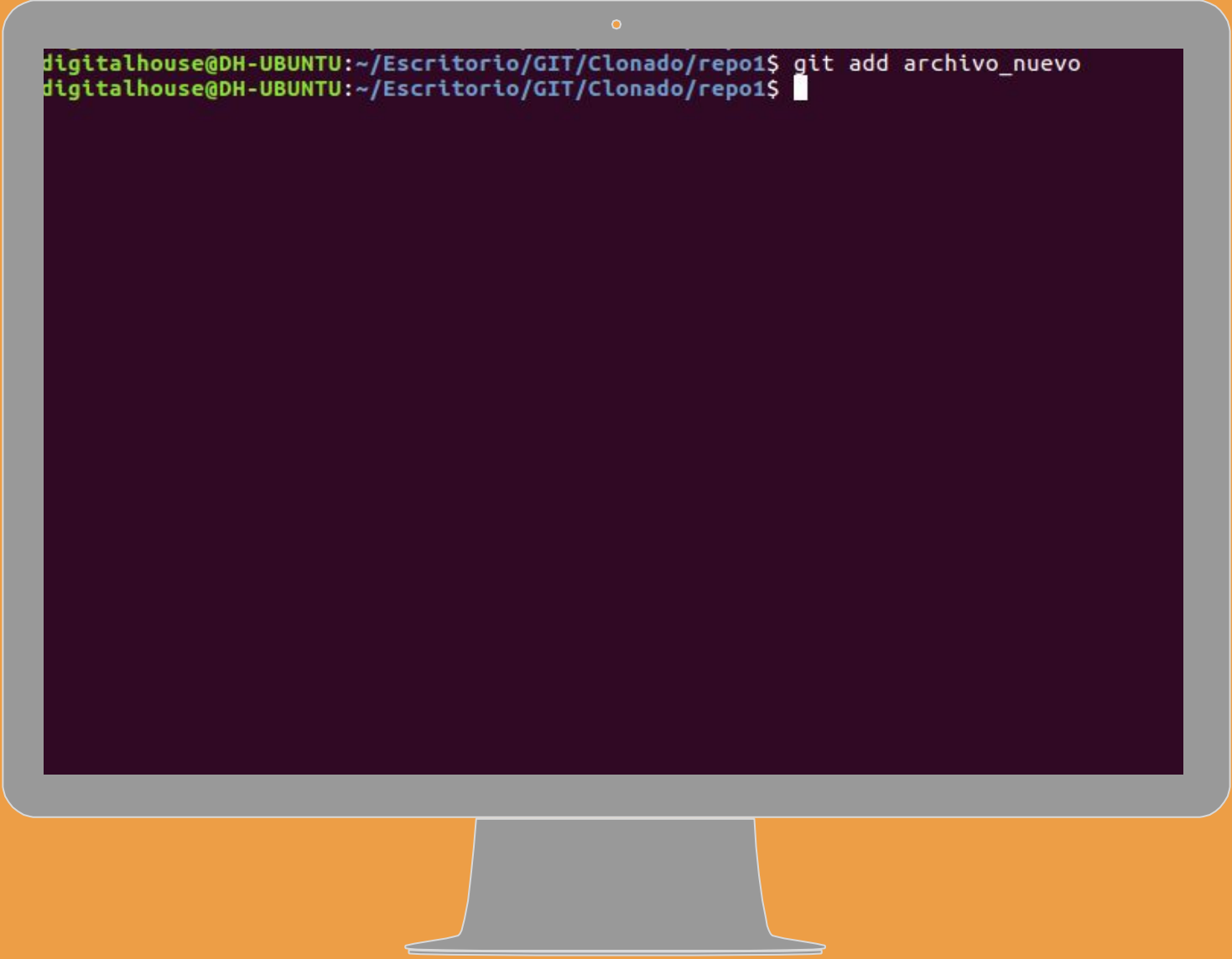


GIT INIT

Crea un repositorio local (en nuestra máquina) y nos permite comenzar a utilizar todas las funcionalidades de GIT.

Generalmente crea una carpeta oculta la cual contiene todo el repositorio y sus distintas ramificaciones.

AGREGAR ARCHIVOS (git add all)



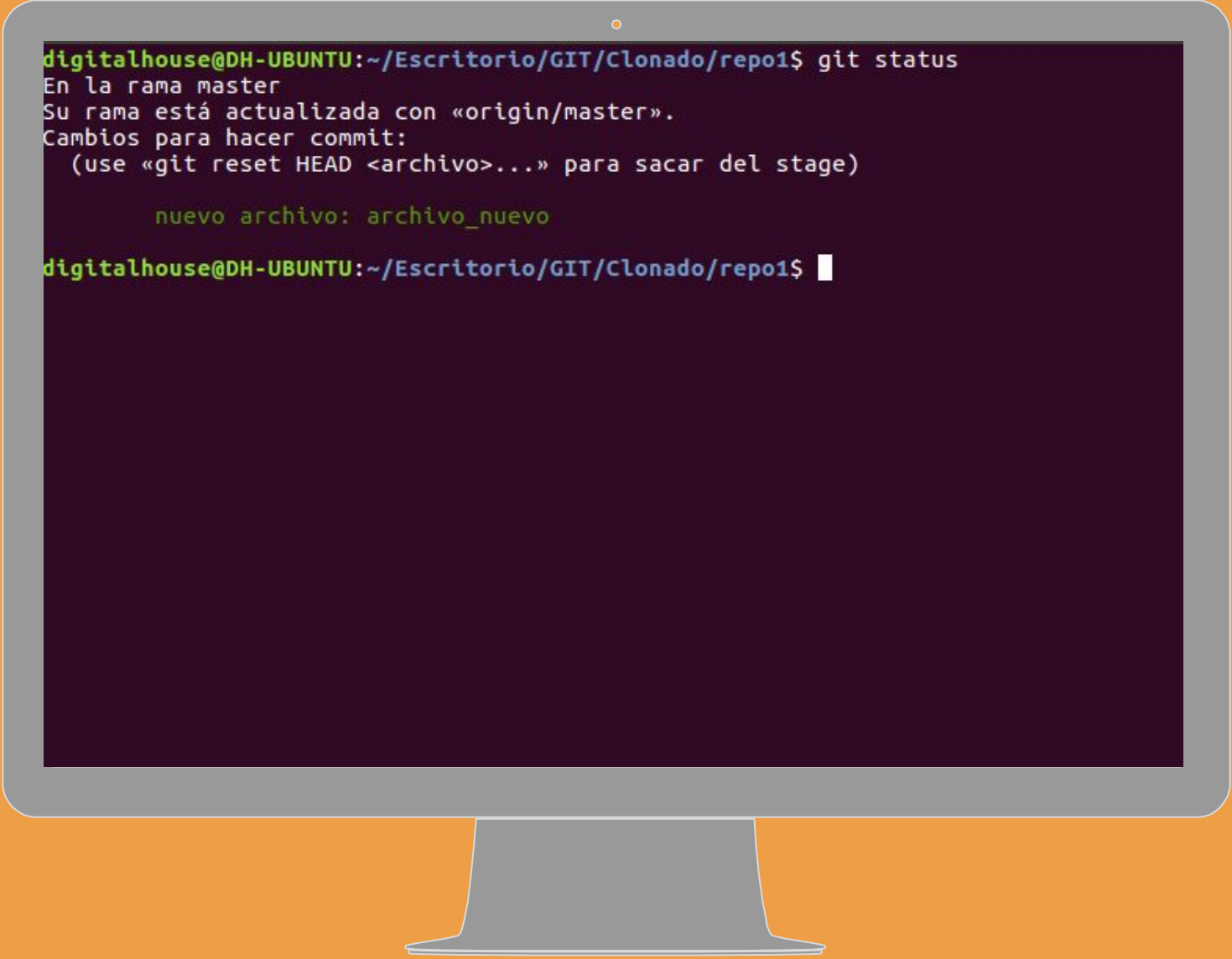
```
digitalhouse@DH-UBUNTU:~/Escritorio/GIT/Clonado/repo1$ git add archivo_nuevo
digitalhouse@DH-UBUNTU:~/Escritorio/GIT/Clonado/repo1$
```

■ GIT ADD ALL ó GIT ADD archivo.txt

git add all - agrega al repositorio *todos* los archivos que hayamos creado en nuestro proyecto.

git add archivo.txt - agrega al repositorio el archivo referenciado.

REVISAMOS LOS CAMBIOS (git status)



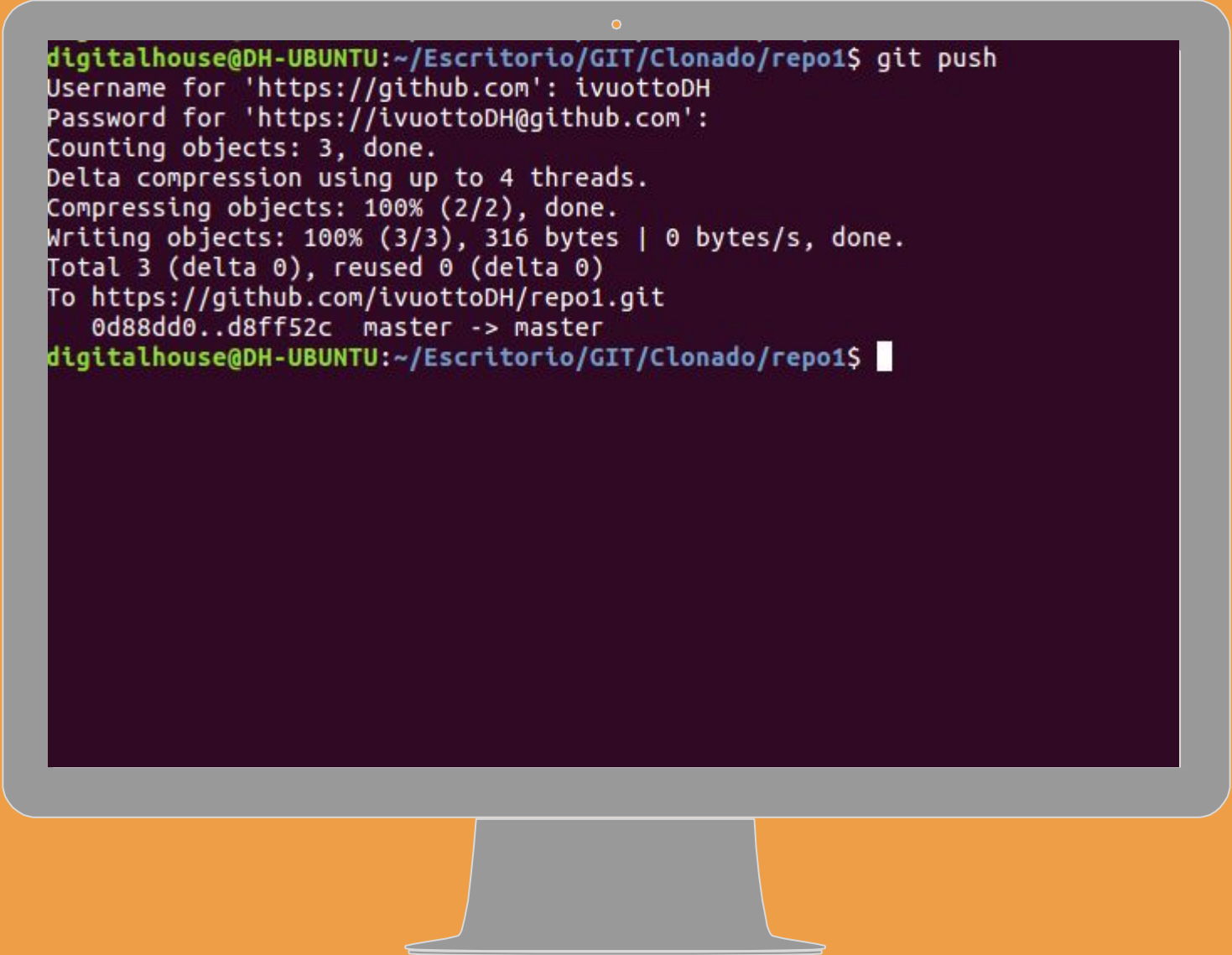
```
digitalhouse@DH-UBUNTU:~/Escritorio/GIT/Clonado/repo1$ git status
En la rama master
Su rama está actualizada con «origin/master».
Cambios para hacer commit:
  (use «git reset HEAD <archivo>...» para sacar del stage)

    nuevo archivo: archivo_nuevo
digitalhouse@DH-UBUNTU:~/Escritorio/GIT/Clonado/repo1$
```

GIT STATUS

Nos permite verificar el estado de nuestro repositorio, nos dice si hay archivos agregados al repositorio o si hay archivos que aún no han sido agregados al mismo.

"COMITEAMOS" LOS CAMBIOS (git commit)



```
digitalhouse@DH-UBUNTU:~/Escritorio/GIT/Clonado/repo1$ git push
Username for 'https://github.com': ivuottoDH
Password for 'https://ivuottoDH@github.com':
Counting objects: 3, done.
Delta compression using up to 4 threads.
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 316 bytes | 0 bytes/s, done.
Total 3 (delta 0), reused 0 (delta 0)
To https://github.com/ivuottoDH/repo1.git
   0d88dd0..d8ff52c  master -> master
digitalhouse@DH-UBUNTU:~/Escritorio/GIT/Clonado/repo1$
```


GIT COMMIT

Nos permite decir "*oficialmente*" que los archivos que agregamos con anterioridad los queremos tener preparados y listos para después subirlos al repositorio remoto (nube).

Este comando debe ir acompañado de un mensaje así:

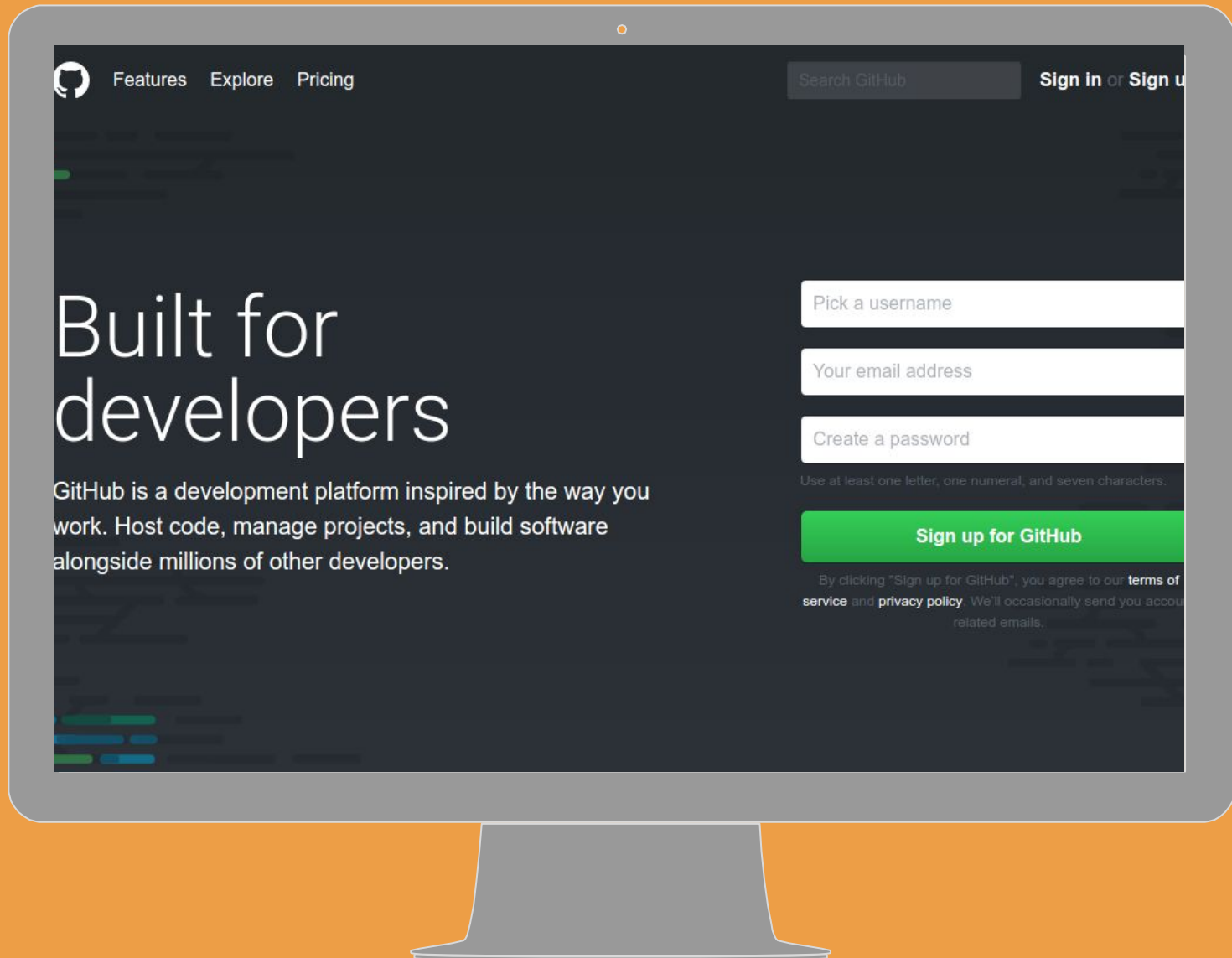
git commit -m "Primer versión de la aplicación"

ENVIAMOS LOS CAMBIOS (git push)

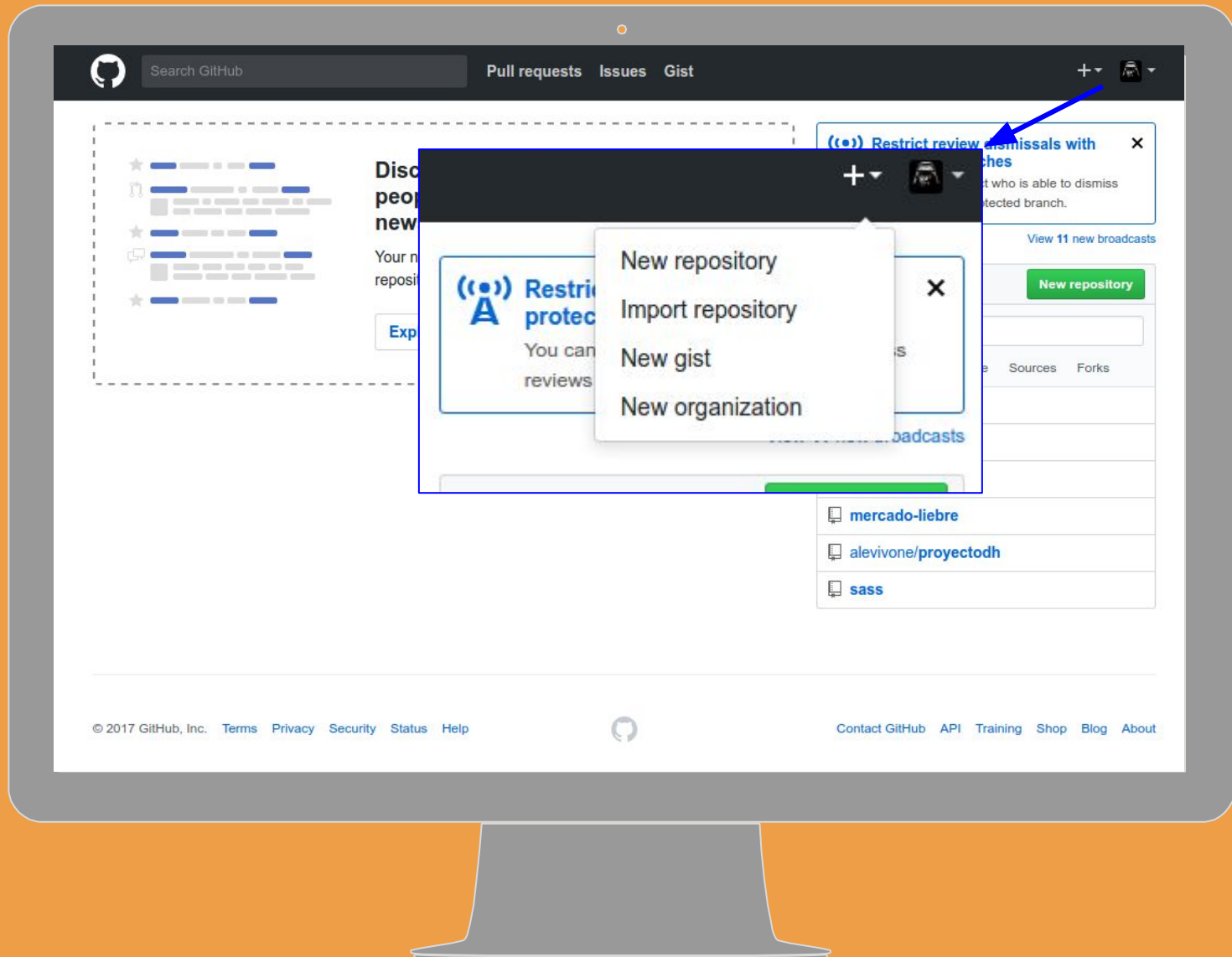
```
digitalhouse@DH-UBUNTU:~/Escritorio/GIT/Clonado/repo1$ git push
Username for 'https://github.com': ivuottoDH
Password for 'https://ivuottoDH@github.com':
Counting objects: 3, done.
Delta compression using up to 4 threads.
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 316 bytes | 0 bytes/s, done.
Total 3 (delta 0), reused 0 (delta 0)
To https://github.com/ivuottoDH/repo1.git
   0d88dd0..d8ff52c  master -> master
digitalhouse@DH-UBUNTU:~/Escritorio/GIT/Clonado/repo1$
```

**ANTES DEL
PUSH**

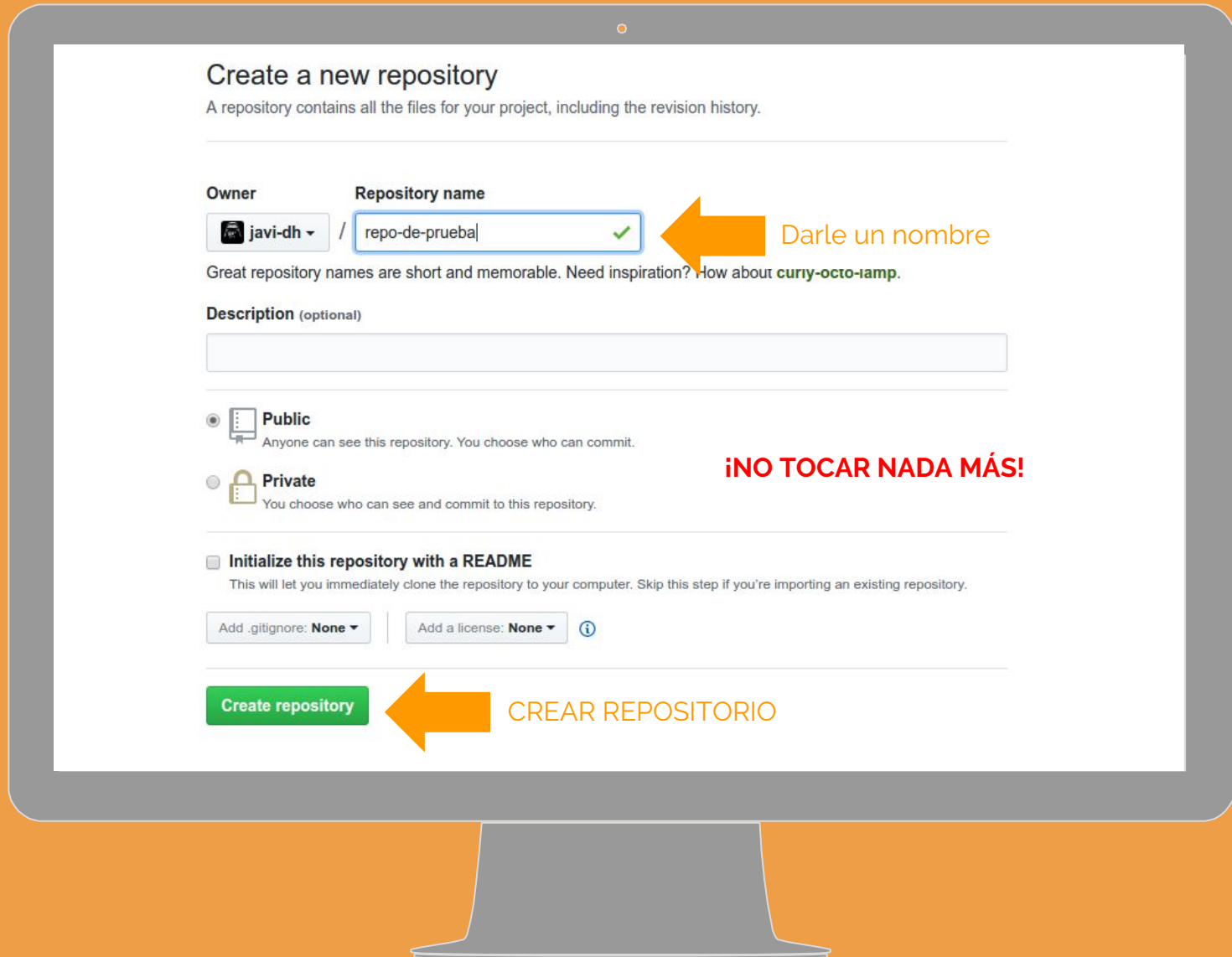
CREAR CUENTA EN GITHUB.COM



CREAR REPOSITORIO REMOTO



CREAR REPOSITORIO REMOTO



The image shows a computer monitor displaying the GitHub 'Create a new repository' page. The page has a white background with a light gray border. The title 'Create a new repository' is at the top, followed by a subtitle 'A repository contains all the files for your project, including the revision history.' Below this is a form with several sections. The 'Owner' section shows 'javi-dh' with a dropdown arrow. The 'Repository name' section shows 'repo-de-prueba' with a green checkmark. An orange arrow points to this field with the text 'Darle un nombre'. The 'Description (optional)' section has a text input field. The 'Public' and 'Private' radio buttons are visible, with 'Public' selected. An orange arrow points to the 'Private' option with the text '¡NO TOCAR NADA MÁS!'. The 'Initialize this repository with a README' checkbox is unchecked. Below this are two dropdown menus for 'Add .gitignore: None' and 'Add a license: None'. At the bottom is a green 'Create repository' button. An orange arrow points to this button with the text 'CREAR REPOSITORIO'.

Create a new repository

A repository contains all the files for your project, including the revision history.

Owner: javi-dh / Repository name: repo-de-prueba ✓

Great repository names are short and memorable. Need inspiration? How about curly-octo-lamp.

Description (optional)

☒ Public
Anyone can see this repository. You choose who can commit.

☐ Private
You choose who can see and commit to this repository.

☐ Initialize this repository with a README
This will let you immediately clone the repository to your computer. Skip this step if you're importing an existing repository.

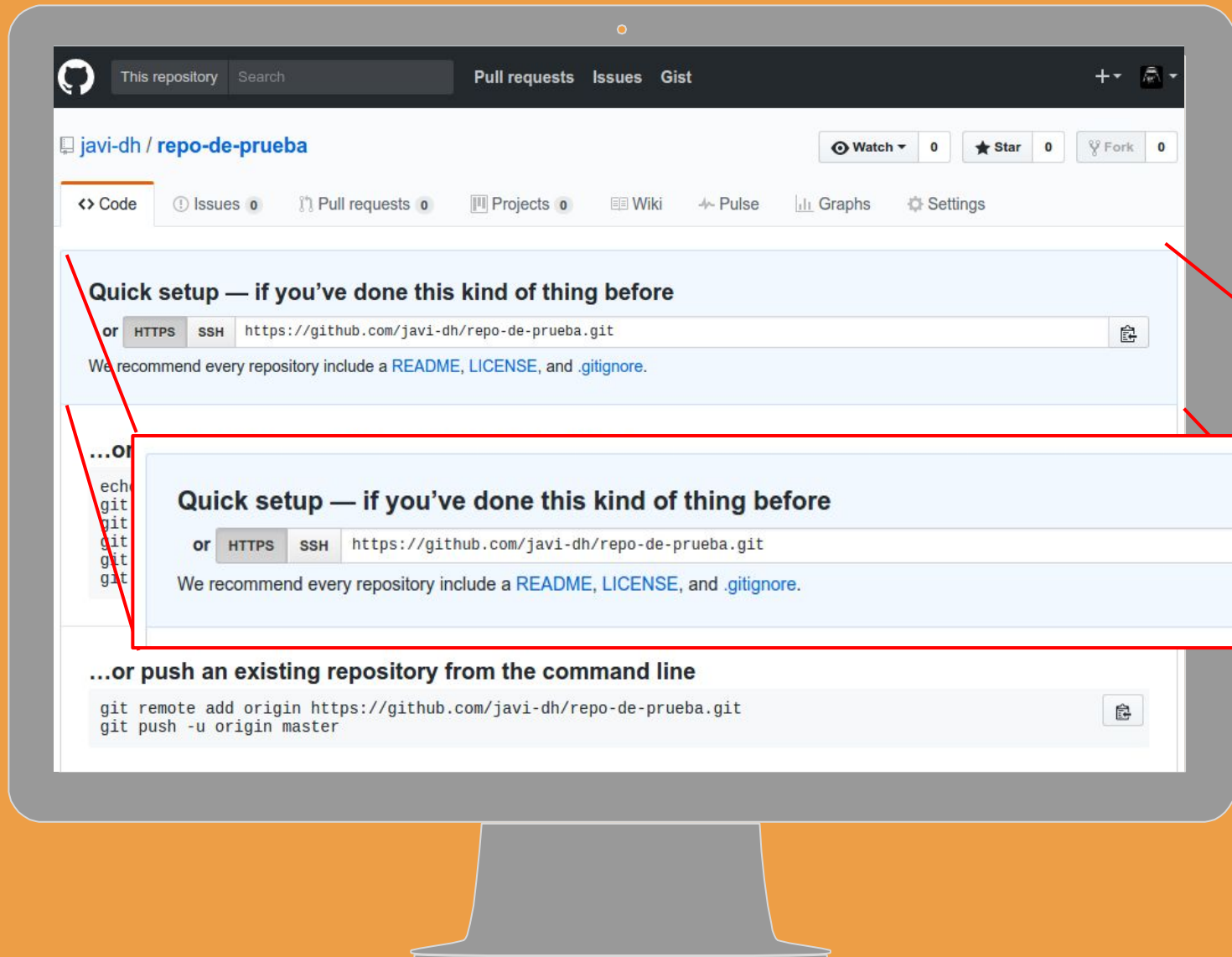
Add .gitignore: None | Add a license: None ⓘ

Create repository

Annotations:

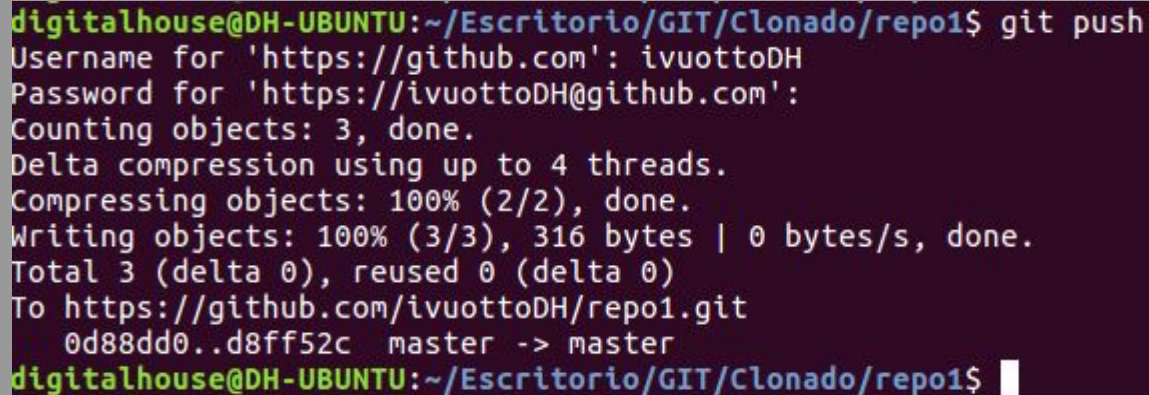
- Darle un nombre (points to Repository name)
- ¡NO TOCAR NADA MÁS! (points to Private option)
- CREAR REPOSITORIO (points to Create repository button)

TENER A LA MANO ESTA URL



SETEAMOS EL REPO REMOTO

(git remote add origin)

A terminal window with a dark purple background and a grey border, mounted on a grey stand. The terminal shows the output of a 'git push' command. The text is as follows:

```
digitalhouse@DH-UBUNTU:~/Escritorio/GIT/Clonado/repo1$ git push
Username for 'https://github.com': ivuottoDH
Password for 'https://ivuottoDH@github.com':
Counting objects: 3, done.
Delta compression using up to 4 threads.
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 316 bytes | 0 bytes/s, done.
Total 3 (delta 0), reused 0 (delta 0)
To https://github.com/ivuottoDH/repo1.git
    0d88dd0..d8ff52c  master -> master
digitalhouse@DH-UBUNTU:~/Escritorio/GIT/Clonado/repo1$
```

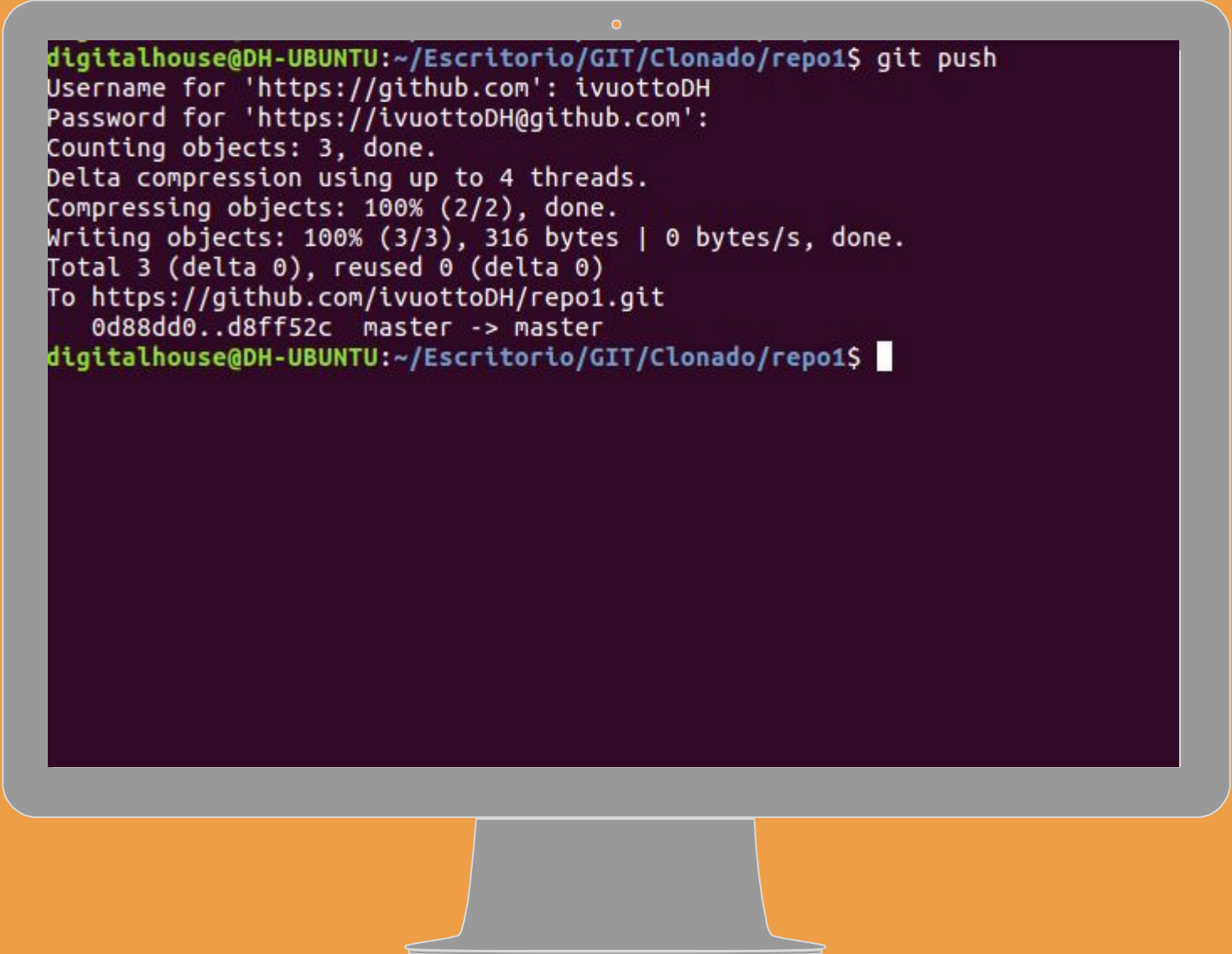
GIT REMOTE ADD ORIGIN

Nos permite decirle a nuestro repositorio LOCAL en qué repositorio REMOTO vamos a subir nuestros archivos.

Este comando debe verse así:

```
git remote add origin https://github.com/usuario/repositorio.git
```


FINALMENTE - ENVIAMOS NUESTROS ARCHIVOS (git push)



```
digitalhouse@DH-UBUNTU:~/Escritorio/GIT/Clonado/repo1$ git push
Username for 'https://github.com': ivuottoDH
Password for 'https://ivuottoDH@github.com':
Counting objects: 3, done.
Delta compression using up to 4 threads.
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 316 bytes | 0 bytes/s, done.
Total 3 (delta 0), reused 0 (delta 0)
To https://github.com/ivuottoDH/repo1.git
   0d88dd0..d8ff52c  master -> master
digitalhouse@DH-UBUNTU:~/Escritorio/GIT/Clonado/repo1$
```

GIT PUSH

Es la directriz final cuando queremos llevar todo nuestro trabajo del repositorio *local* al *remoto*, en este comando, debemos especificar desde dónde enviamos los archivos y en dónde los queremos subir.

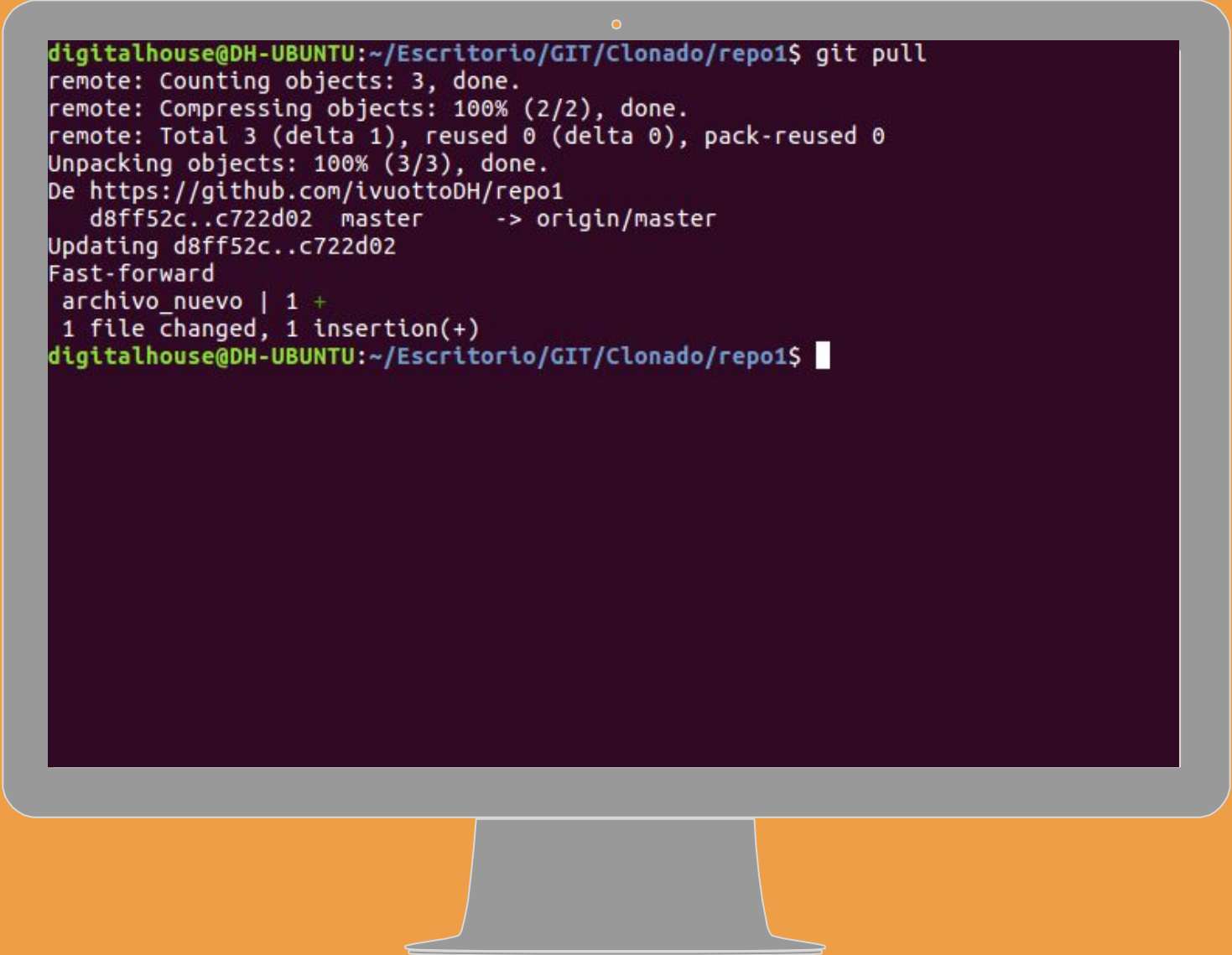
Este comando debe verse así:

git push origin master

**OK, YA TENEMOS NUESTRO
REPOSITORIO REMOTO
ACTUALIZADO!**

**¿Y SI LOS CAMBIOS LOS SUBIÓ
OTRO PERSONA AL REPOSITORIO
REMOTO?
(repositorio local desactualizado)**

ACTUALIZAR DIRECTORIO LOCAL (git pull)



```
digitalhouse@DH-UBUNTU:~/Escritorio/GIT/Clonado/repo1$ git pull
remote: Counting objects: 3, done.
remote: Compressing objects: 100% (2/2), done.
remote: Total 3 (delta 1), reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (3/3), done.
De https://github.com/ivuottoDH/repo1
   d8ff52c..c722d02  master    -> origin/master
Updating d8ff52c..c722d02
Fast-forward
  archivo_nuevo | 1 +
  1 file changed, 1 insertion(+)
digitalhouse@DH-UBUNTU:~/Escritorio/GIT/Clonado/repo1$
```

GIT PULL

Nos permite "bajar" las actualizaciones existentes en el repositorio remoto a nuestro repositorio local, para de esta manera tener en nuestra máquina, la versión más reciente de los archivos del proyecto.

¿USO LA
CONSOLA O
USO UN GUI?

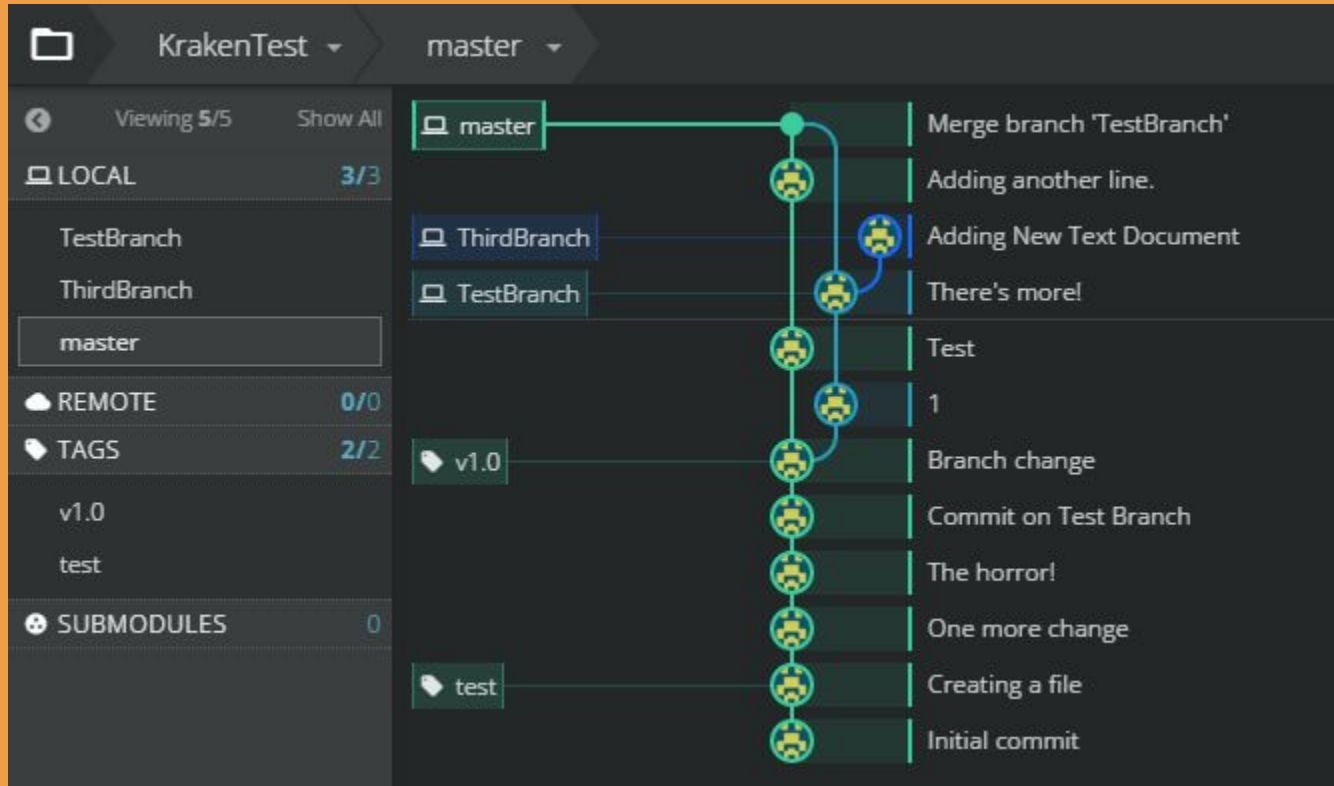


■ GUI - Graphical User Interface

Si bien, la consola es la mejor herramienta para trabajar con GIT, existen programas con una interfaz gráfica que nos permiten hacer todo este proceso mucho más visual.

ATENTIS: Muchas veces el uso de estos programas genera más conflictos que soluciones :(

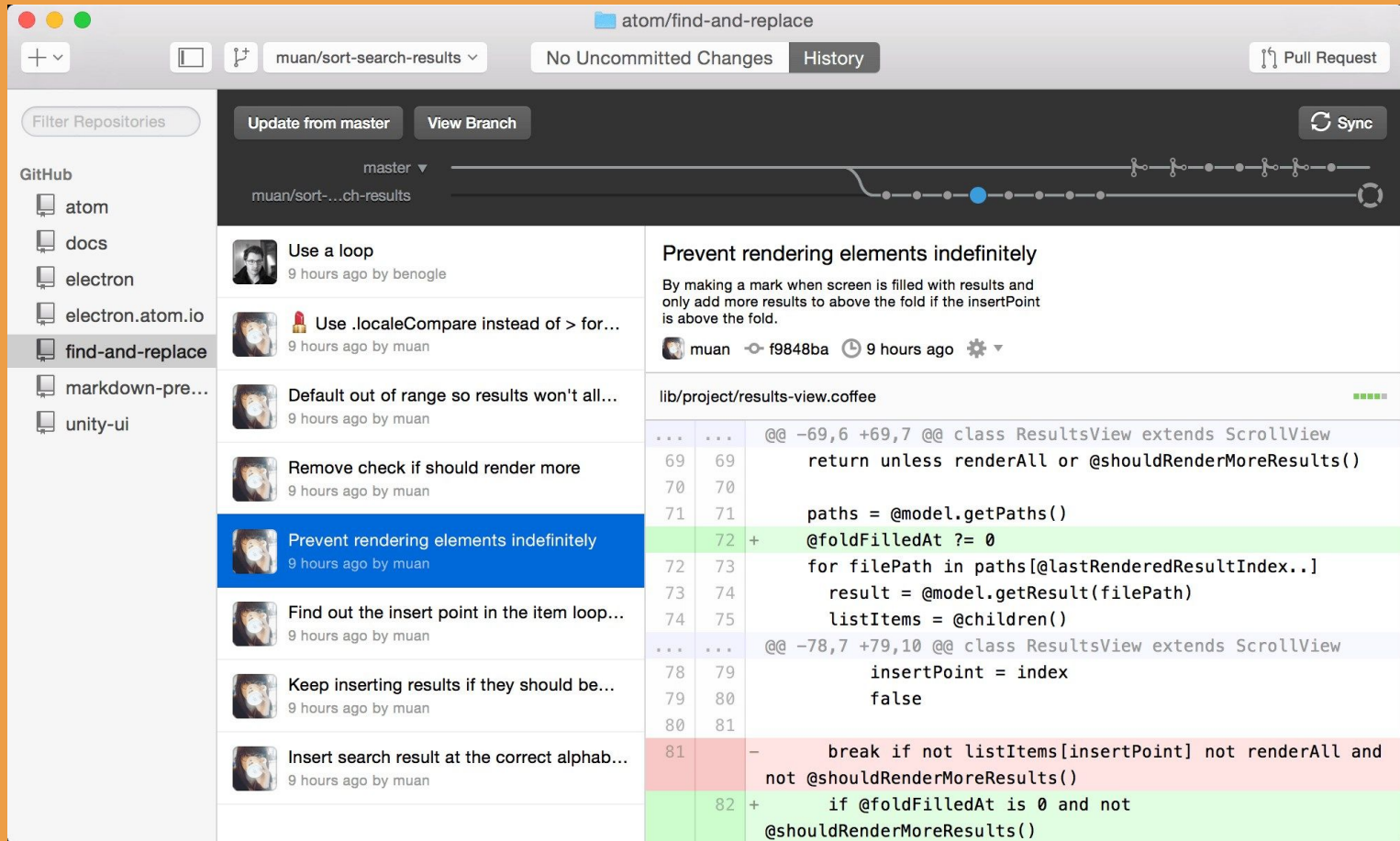
Algunos GUI para usar GIT



Git Kraken

<https://www.gitkraken.com/>

Algunos GUI para usar GIT



Github Desktop

<https://desktop.github.com/>



CONFLICTOS

CONFLICTO COMUNES

Sea que usemos la consola de comandos o un GUI, el error más frecuente es cuando dos o más usuarios **tocan el mismo archivo en la misma línea de código**.

Para evitar esto, la recomendación más eficiente es dividir bien el laburo y no trabajar al mismo tiempo varias personas sobre el mismo archivo.

GRACIAS!

Consultas?
