

# Auxiliar 1

IN3501 Tecnologías de Información para la Comunicación y Gestión

#### Auxiliares:

Pablo Hernandez – p.hernandez.m@ug.uchile.cl Javiera Ovalle – javiera.ovallet@gmail.com Jorge Pinto – jorgepintoriveros@gmail.com

#### Instalaciones Previas

Dudas o problemas con Django o Git

#### Git - Github

Git es un software de control de versiones para el desarrollo de otros software. Su propósito es llevar los cambios en los distintos archivos de un software y coordinar el trabajo para que varias personas puedan trabajar en un mismo proyecto.

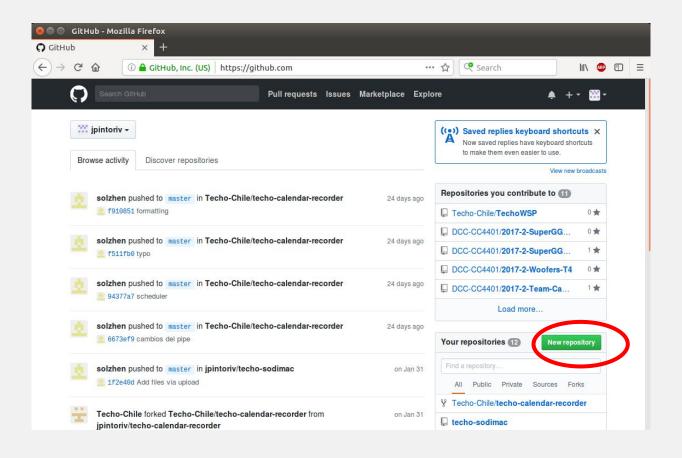
Github es una plataforma que trabaja en base a git.

### Crear un repositorio en Github

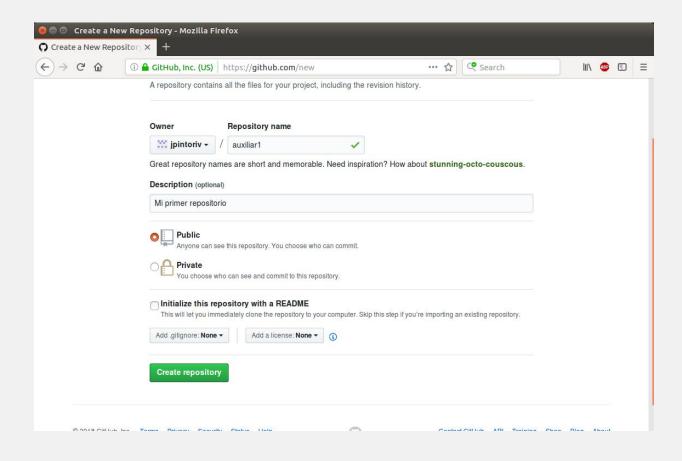
Existe una forma clásica, pero en el curso usaremos un Classroom

Un classroom es una plataforma de Github que nos ayudará a simplificar el trabajo en grupos y la revisión de las tareas por parte de los ayudantes.

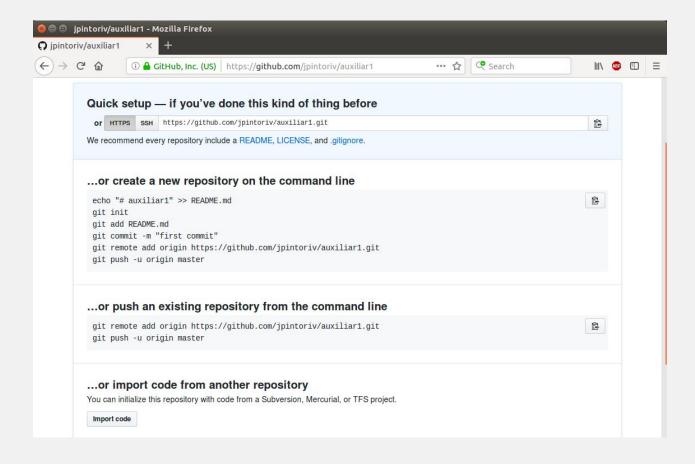
#### Forma Clasica



### Forma Clasica

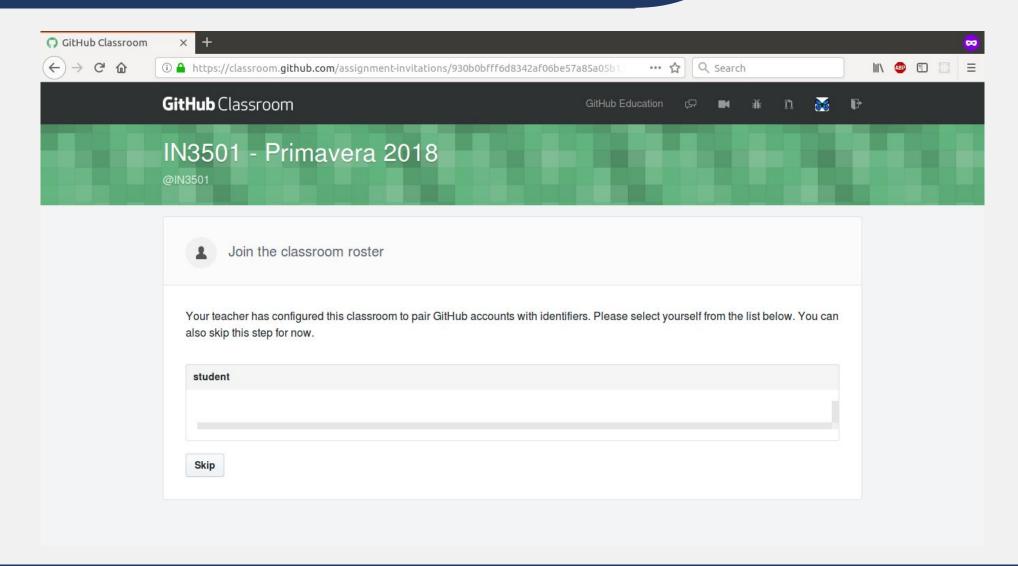


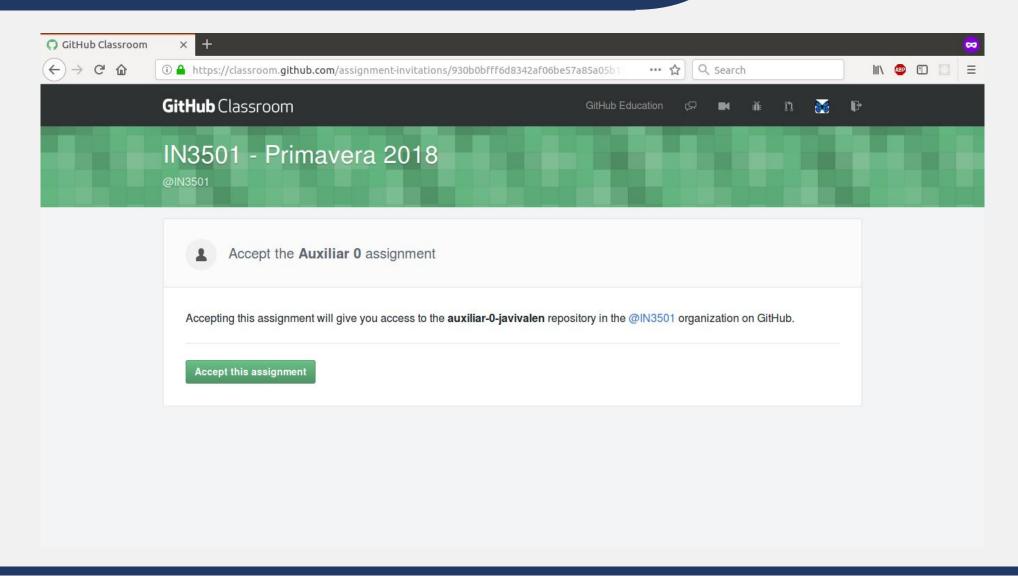
#### Forma Clasica

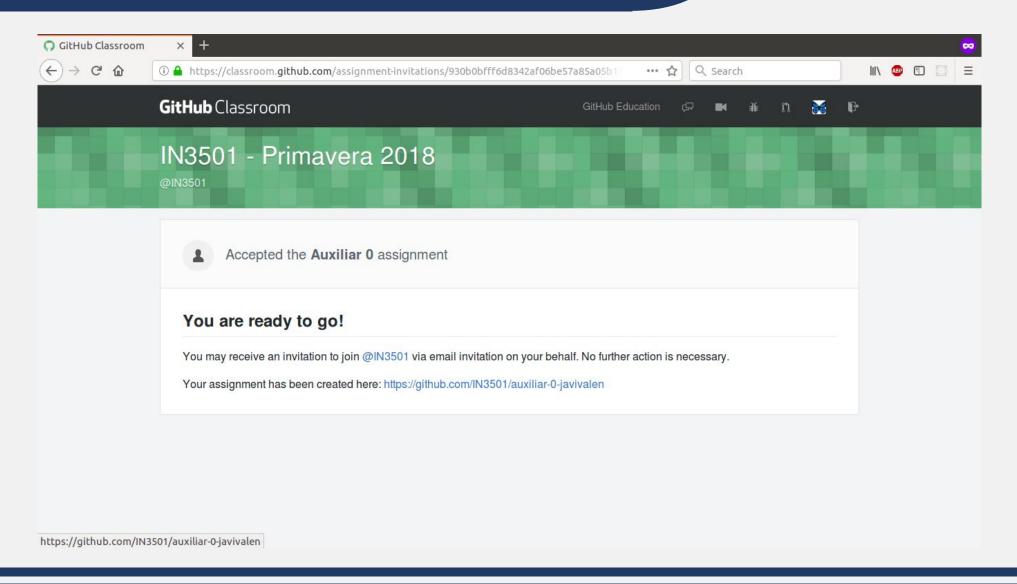


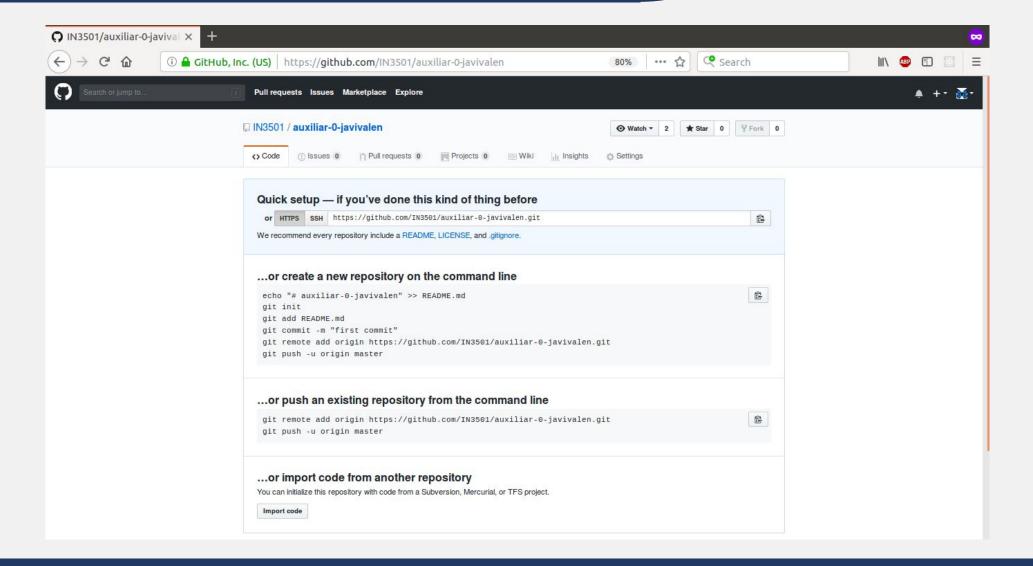
Deben apretar el link que se les entregará según la tarea o el desafio, el link para esta auxiliar es el siguiente:

https://classroom.github.com/a/YAz46U\_i





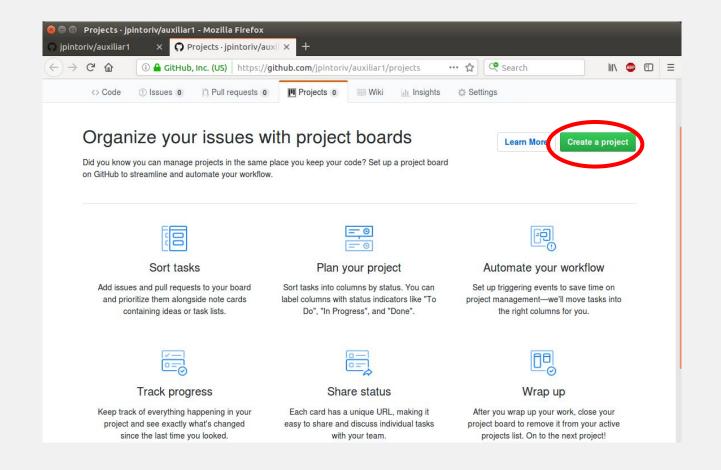




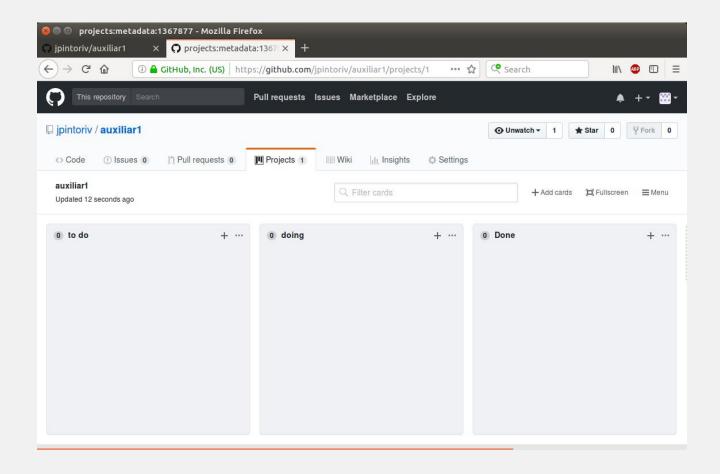
## Trabajo en equipo

- 1. Todo el equipo saber que está haciendo el resto
- 2. Todo el equipo debe manejar el mismo código
- 3. Todo el equipo debe saber que es lo que falta por hacer
- 4. Todo el equipo debe saber lo que ya está hecho

## Creacion de un tablero (board)



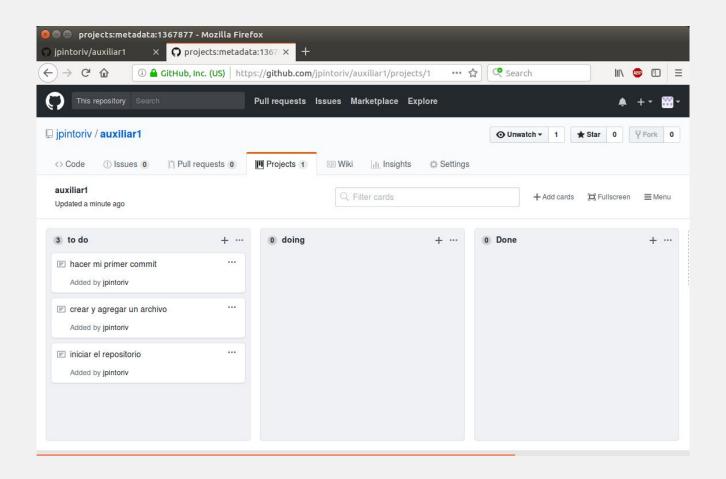
## Creacion de un tablero (board)



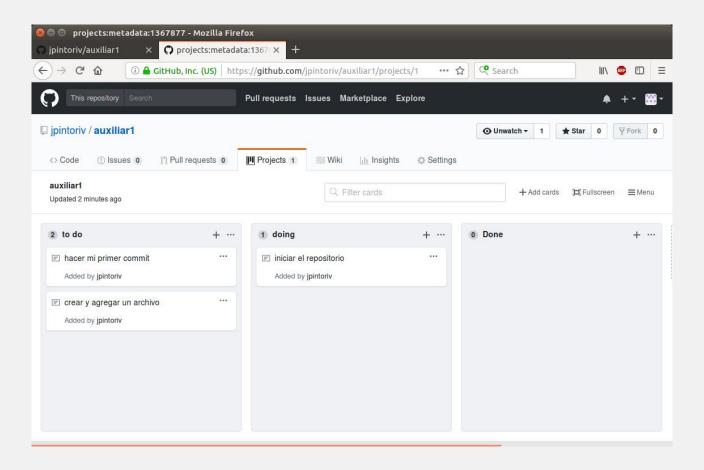
#### Primeros Pasos en Github

- 1. Iniciar repositorio en nuestro computador
- 2. Crear y agregar un archivo
- 3. Realizar nuestro primer commit

#### Primeros Pasos en Github



## Iniciar el repositorio



## Iniciar el repositorio

Es importante iniciar el repositorio en nuestro computador, para esto debemos escoger un directorio y luego ejecutar el comando:

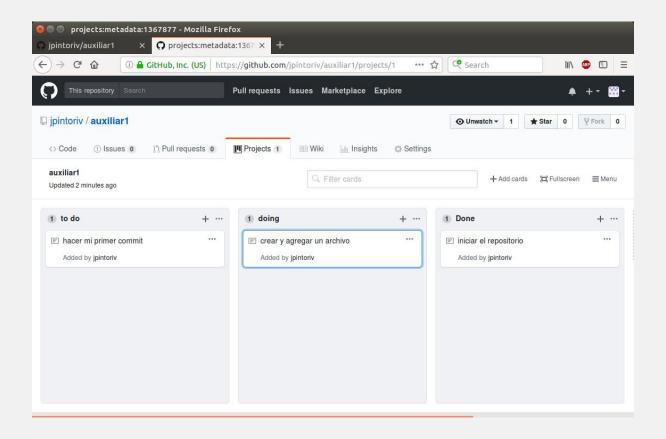
#### git init

Este comando creará un repositorio vacío con el que luego trabajaremos.

## Iniciar el repositorio

```
gorge@jp: ~/Escritorio/auxiliar1
gorge@jp:~/Escritorio/auxiliar1$ git init
Initialized empty Git repository in /home/gorge/Escritorio/auxiliar1/.git/gorge@jp:~/Escritorio/auxiliar1$
```

## Crear y agregar un archivo



## Crear y agregar un archivo

Creamos un archivo suma.py, el cual solamente es una función que se encarga de sumar 2 números:

def suma(a,b): return a + b

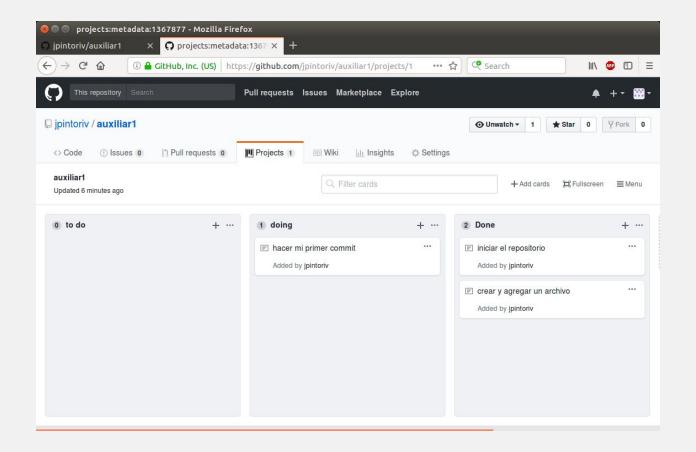
Para agregar una archivo a un futuro commit, debemos usar el comando:

git add nombre\_archivo

Importante: cada vez que se cambia un archivo, si deseamos incluir estos cambios se debe ejecutar agregar el archivo

## Crear y agregar un archivo

```
gorge@jp: ~/Escritorio/auxiliar1
gorge@jp:~/Escritorio/auxiliar1$ git init
Initialized empty Git repository in /home/gorge/Escritorio/auxiliar1/.git/
gorge@jp:~/Escritorio/auxiliar1$ ls
suma.py
gorge@jp:~/Escritorio/auxiliar1$ git add suma.py
gorge@jp:~/Escritorio/auxiliar1$
```



Un commit corresponde a un conjunto de cambios que yo realice en un repositorio. Por ejemplo si yo modifique 3 archivos, primero debo usar el comando add para agregar esos 3 archivos al commit y luego ejecutar el commit.

Si no se ejecuta el commit, no ocurre ningún cambio en el repositorio.

#### git commit -m "información del commit"

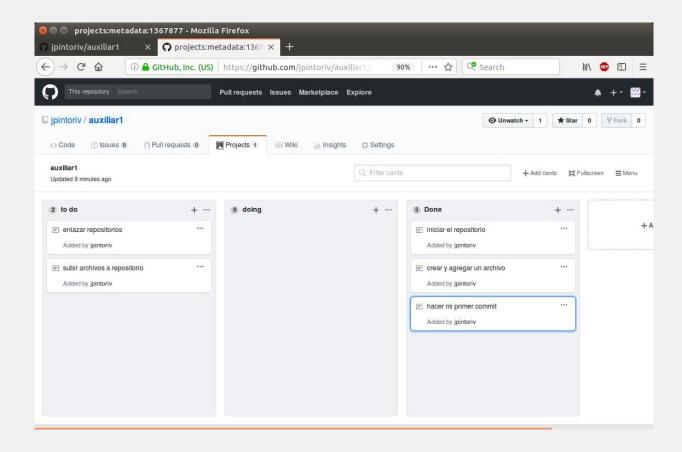
se recomienda usar la **flag -m** seguida de un comentario, este comentario tiene relación con el commit, por ejemplo, si mi commit tiene relacion con algun error puedo agregar un comentario como: "corrección error numero 17"

```
gorge@jp:~/Escritorio/auxiliar1$ git init
Initialized empty Git repository in /home/gorge/Escritorio/auxiliar1/.git/
gorge@jp:~/Escritorio/auxiliar1$ ls
suma.py
gorge@jp:~/Escritorio/auxiliar1$ git add suma.py
gorge@jp:~/Escritorio/auxiliar1$ git commit -m "mi primer commit"
[master (root-commit) 69f201c] mi primer commit
    1 file changed, 5 insertions(+)
    create mode 100644 suma.py
```

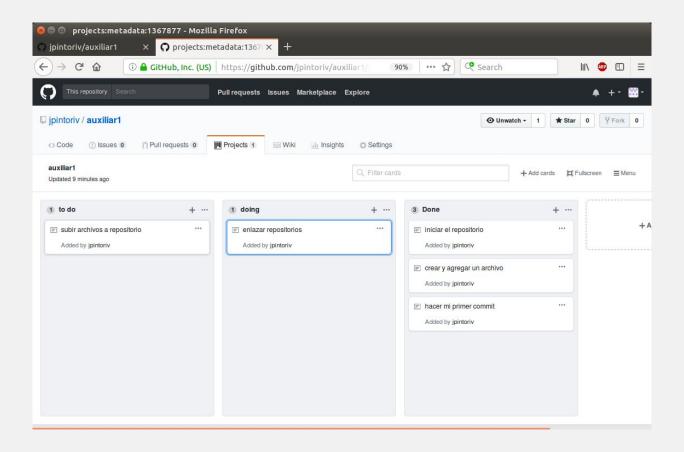
Hasta ahora solo hemos trabajado en un repositorio local (en nuestro computador), no hemos enlazado el repositorio creado con nuestro repositorio local.

Lo anterior quiere decir que los cambios por ahora son solo cambios locales.

#### **Nuevas Tareas**



## Enlazar repositorios



### Enlazar repositorios

Para clonar un repositorio en mi computador yo debo usar el comando:

git clone direccion\_del\_repositorio

La direccion del repositorio en github suele ser: github.com/nombre\_usuario/nombre\_repositorio.git

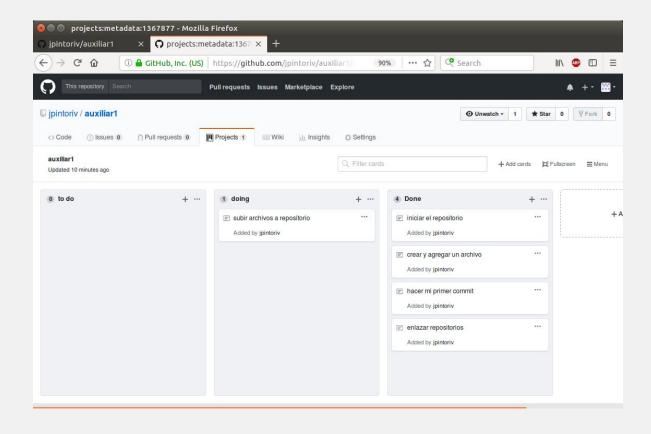
Como es un repositorio vacío, debemos usar el siguiente comando:

git remote add origin direccion\_del\_repositorio

## Enlazar repositorios

```
gorge@jp: ~
ggorge@jp:~$
gorge@jp:~$ git remote add origin https://github.com/jpintoriv/auxiliar1.git
```

## Subir archivos a repositorio



### Subir archivos a repositorio

Para "subir" los archivos al repositorio, se ocupa el comando:

#### git push

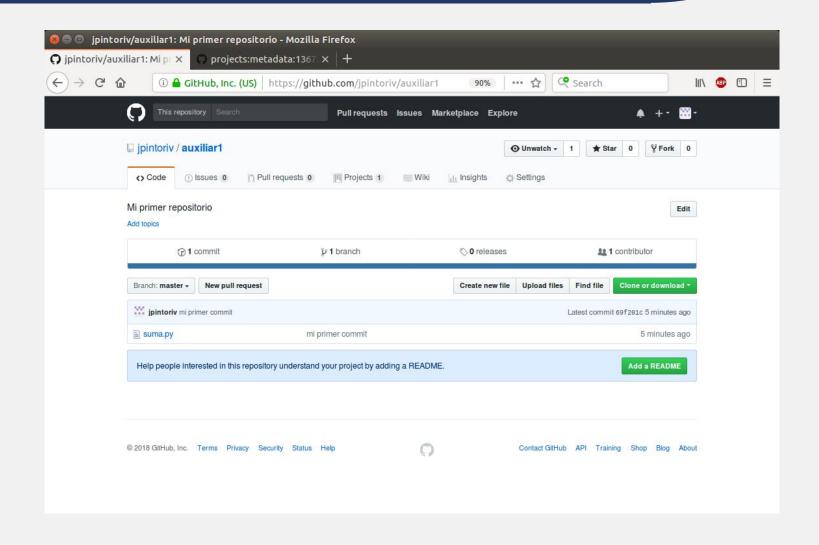
En caso de que nuestro nuestro repositorio local esté desactualizado con respecto al repositorio online debemos usar:

#### git pull

Como es un repositorio vacío, en este caso se debe ocupar el comando

git push -u origin master

#### Estado actual



#### Estado actual

Para saber en qué estado se encuentra nuestro repositorio local se usa el comando:

#### git status

Este nos indica que archivos no hemos agregado, los archivos que estan actualizados y los desactualizados con respecto al repositorio.

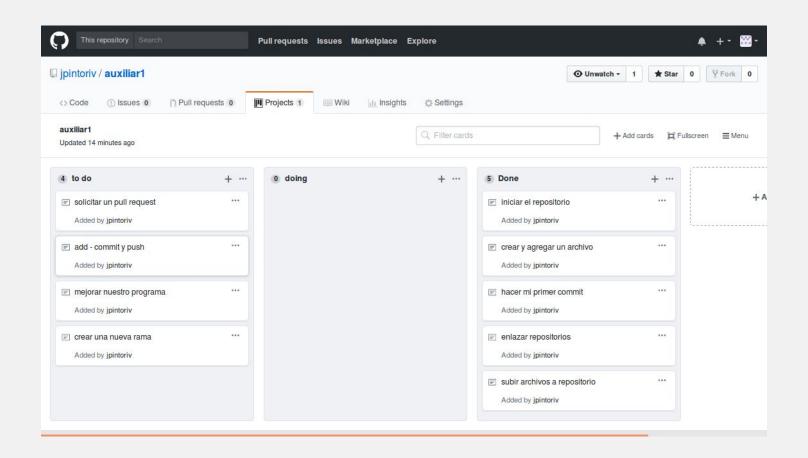
## Nuevas Tareas (branch)

Git trabaja con un sistema de ramas, en las cuales permite realizar trabajo en paralelo sin que interfieran las ramas.

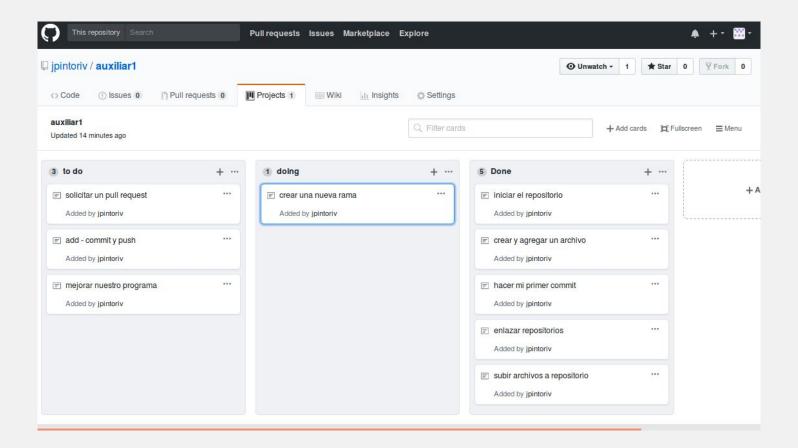
Para el trabajo en equipo, cada vez que alguno de ustedes asuma una tarea se recomienda:

- 1. Crear una nueva rama
- 2. Trabajar en esa rama hasta que la tarea está terminada
- 3. Cuando la tarea está terminada solicitar un pull request
- 4. Esperar a que el resto del equipo lea el pull request y luego aceptarlo

#### **Nuevas Tareas**



#### Crear una nueva rama



#### Crear una nueva rama

Para listar la lista de ramas, se usa el comando:

git branch

Para cambiar de rama, se usa el comando:

git checkout nombre\_de\_rama

Para crear una rama y además cambiarse a esta se usa el comando:

git checkout -b nombre\_rama\_nueva

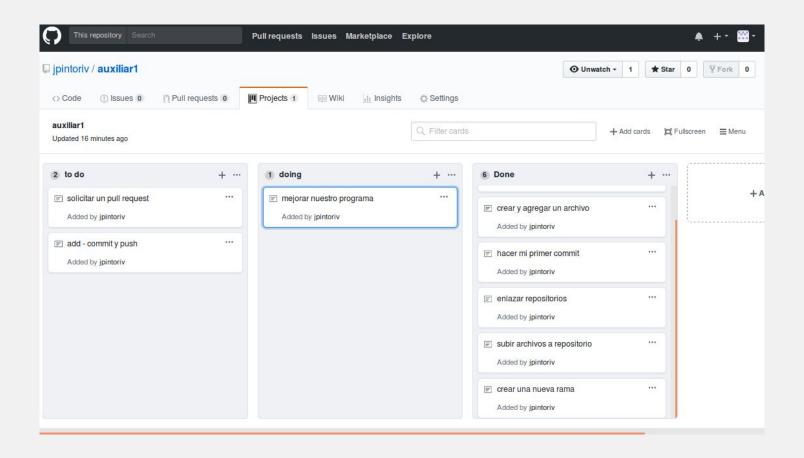
#### Crear una nueva rama

```
gorge@jp:~/Escritorio/auxiliar1$ git branch

* master
gorge@jp:~/Escritorio/auxiliar1$ git checkout -b nueva_rama
Switched to a new branch 'nueva_rama'
gorge@jp:~/Escritorio/auxiliar1$ git branch
    master

* nueva_rama
gorge@jp:~/Escritorio/auxiliar1$
```

# Mejorar nuestro programa

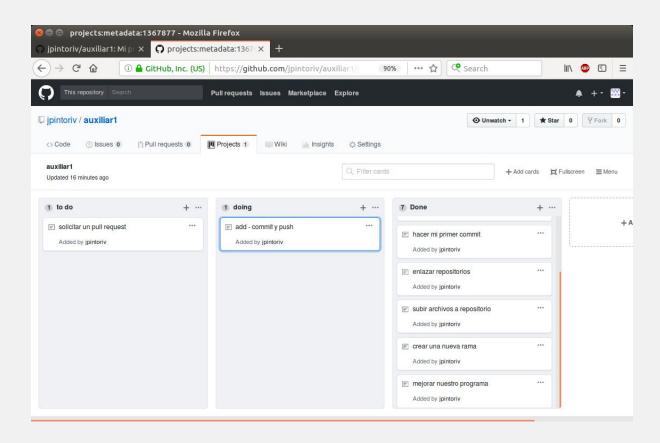


## Mejorar nuestro programa

Supongamos que ahora nos piden un programa que pueda sumar 3 números en lugar de dos, entonces nuestro programa quedaría ahora así:

> def suma(a,b,c): return a + b + c

# Add, commit y push



## Add, commit y push

Simple:

git add suma.py

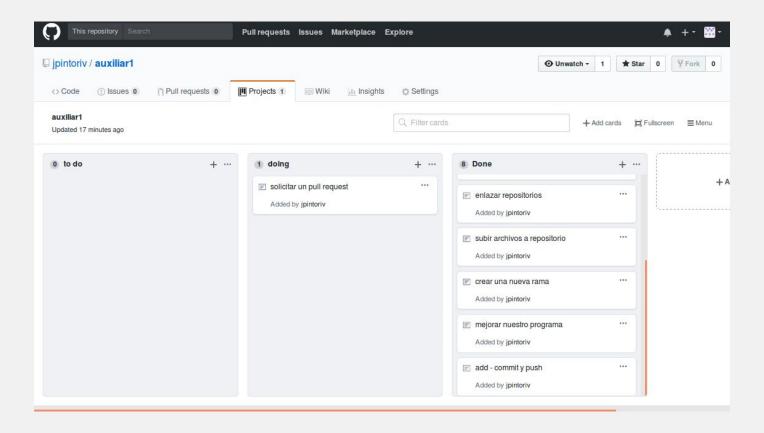
git commit -m "se puede sumar 3 numeros"

git push

Pero, como nuestra rama es nueva, debemos usar un comando diferente:

git push --set-upstream origin nombre\_rama\_nueva

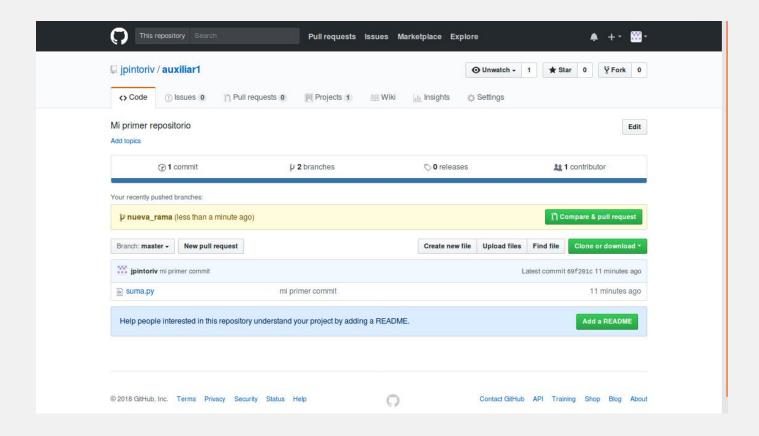
## Add, commit y push

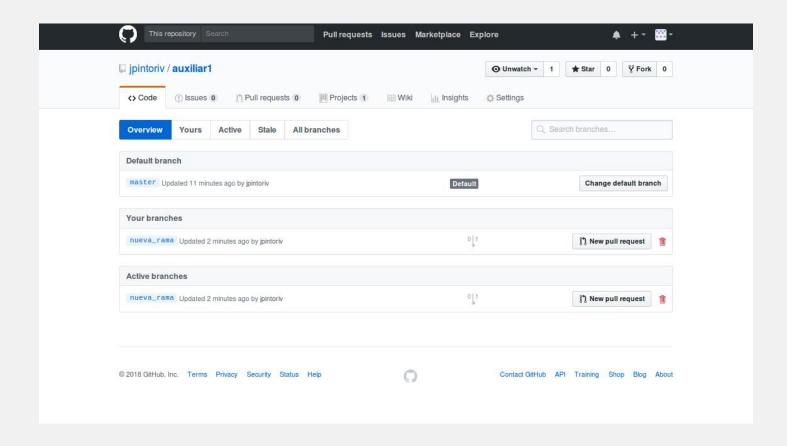


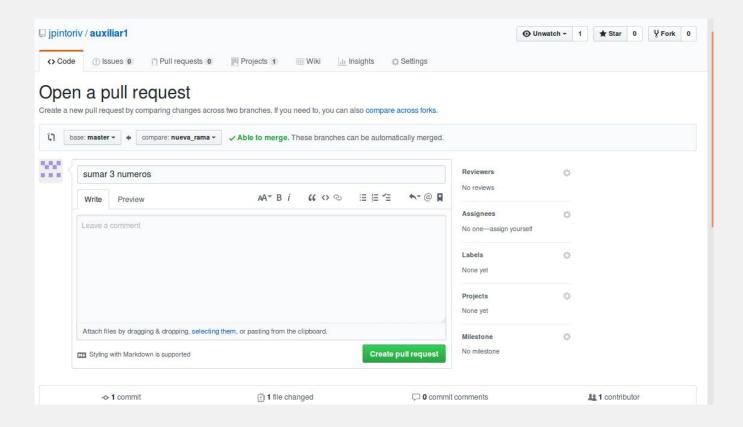
Un pull request corresponde a combinar (merge en lenguaje de git) dos ramas.

Se realiza un pull request en lugar de un merge para que el resto del equipo lea el código y lo acepte

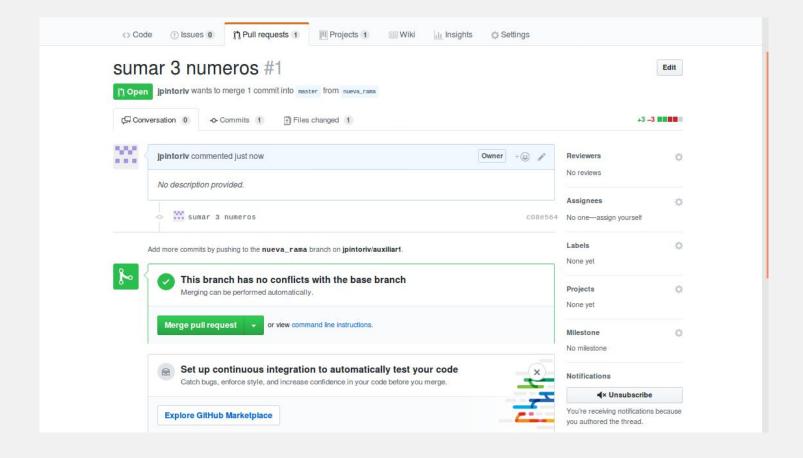
Un **merge** consiste en una mezcla de dos ramas. Git es bastante inteligente y sabe como mezclar dos ramas preservando el ultimo codigo que



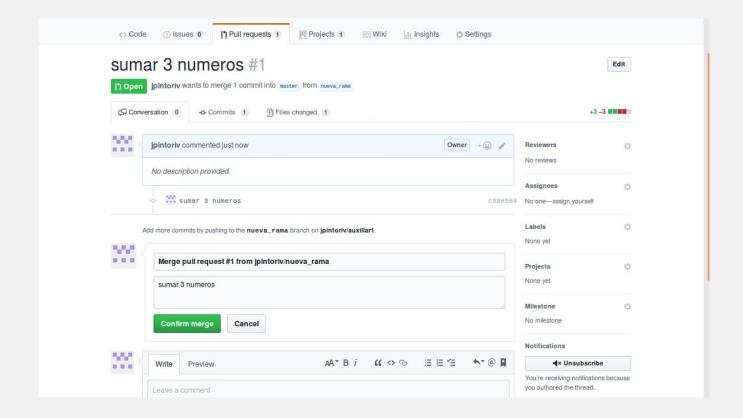




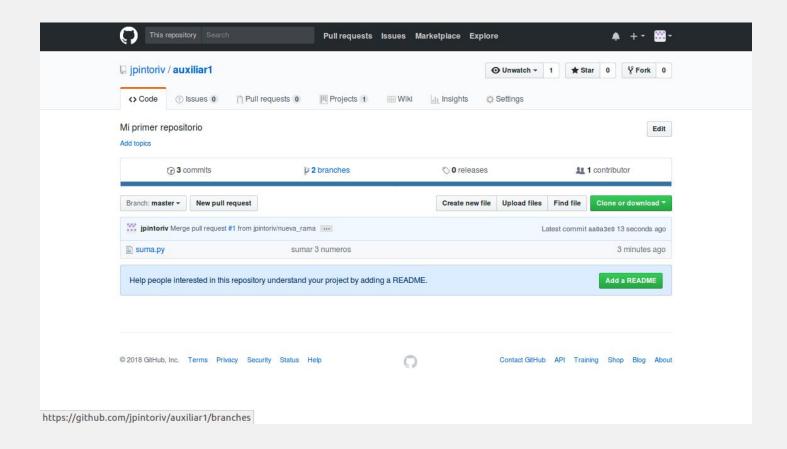
## Aceptar Pull Request



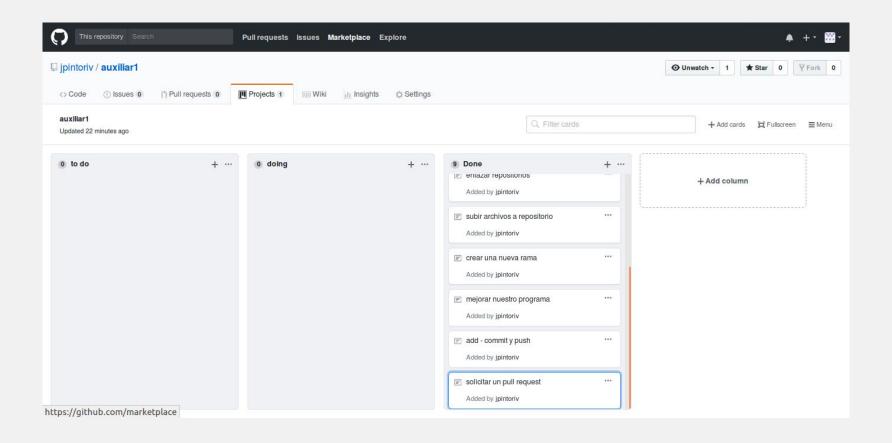
# Aceptar Pull Request - Merge



# Revisar Pagina de Inicio



#### Todas las tareas terminadas



#### Desafío 0

Subir captura de pantalla que demuestre que instaló de manera correcta git Crear una cuenta en github y subir nombre de usuario Subir dirección de repositorio que cumpla con lo enseñado en este tutorial:

- Debe contener 2 ramas
- La rama principal (master) debe estar "mergeada" con la rama secundaria
- Debe tener al menos 2 commits

El nombre de usuario y la dirección del repositorio la pueden comentar en la tarea o pueden subir un archivo que la contenga.

Fecha de Entrega: MIÉRCOLES 26 - 23:59 por Ucursos



# Auxiliar 1

IN3501 Tecnologías de Información para la Comunicación y Gestión

#### Auxiliares:

Pablo Hernandez – p.hernandez.m@ug.uchile.cl Javiera Ovalle – javiera.ovallet@gmail.com Jorge Pinto – jorgepintoriveros@gmail.com