

## PRÁCTICA 1 DE PAT

1. Creación de Fork
2. Prueba de los Comandos
  - 2.1. Git clone

```
● @allendeyarza → /workspaces/p1 (main) $ git clone https://github.com/gitt-3-pat/p1
Cloning into 'p1'...
remote: Enumerating objects: 6, done.
remote: Counting objects: 100% (1/1), done.
remote: Total 6 (delta 0), reused 0 (delta 0), pack-reused 5
Receiving objects: 100% (6/6), done.
```

Git clone lo utilizamos para clonar un repositorio git existente (desde el servidor Git, Git Hub) a mi máquina local. El comando tiene como argumento la URL del repositorio que estamos clonando.

### 2.2. Git Status

```
● @allendeyarza → /workspaces/p1 (main) $ git status
On branch main
Your branch is up to date with 'origin/main'.

Untracked files:
  (use "git add <file>..." to include in what will be committed)
  p1/

nothing added to commit but untracked files present (use "git add" to track)
```

El comando Git Status informa sobre la situación de mi rama actual, en este caso main, con relación al repositorio remoto (origin). Nos responde On branch main que quiere decir que nos encontramos en la rama main del repositorio local. Además nos informa de que nuestra rama local está actualizada respecto a la del repositorio. Esto es lógico puesto que todavía no hemos realizado ninguna actualización.

### 2.3. Git add .

```
● @allendeyarza → /workspaces/p1 (main) $ git add .
warning: adding embedded git repository: p1
hint: You've added another git repository inside your current repository.
hint: Clones of the outer repository will not contain the contents of
hint: the embedded repository and will not know how to obtain it.
hint: If you meant to add a submodule, use:
hint:   git submodule add <url> p1
hint: If you added this path by mistake, you can remove it from the
hint: index with:
hint:   git rm --cached p1
hint: See "git help submodule" for more information.
```

El comando Git add se utiliza para indicar que los cambios o archivos nuevos se lleven al área de preparación (listos para ser incluidos en el próximo commit), en este caso con el .

indicamos que se realice sobre todos los archivos y directorios (si no quisiéramos todos, especificaremos cuales).

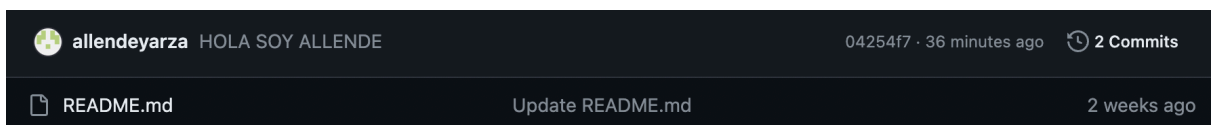
#### 2.4. Git commit

```
● @allendeyarza → /workspaces/p1 (main) $ git commit -m "HOLA SOY ALLENDE"
[main 04254f7] HOLA SOY ALLENDE
1 file changed, 1 insertion(+)
create mode 160000 p1
```

Se utiliza para realizar un commit en mi repositorio git, es decir, guardar los cambios realizados en mi proyecto en el historial de versiones. Los cambios se indican con un mensaje.

#### 2.5. Git push

```
● @allendeyarza → /workspaces/p1 (main) $ git push
Enumerating objects: 3, done.
Counting objects: 100% (3/3), done.
Delta compression using up to 2 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (2/2), 291 bytes | 291.00 KiB/s, done.
Total 2 (delta 0), reused 0 (delta 0), pack-reused 0
To https://github.com/allendeyarza/p1
07720b5..04254f7  main -> main
```



Con Git push enviamos los commits locales al repositorio remoto (se ve en la segunda imagen como ha llegado el commit). Actualizamos el historial del repositorio remoto con mis cambios locales.

#### 2.8. Git Checkout

```
● @allendeyarza → /workspaces/p1 (main) $ git checkout -b feature/1
Switched to a new branch 'feature/1'
● @allendeyarza → /workspaces/p1 (feature/1) $ git checkout main
Switched to branch 'main'
Your branch is up to date with 'origin/main'.
```

Creamos y nos movemos a una nueva rama que hemos denominado feature/1. El -b lo que hace es crear una nueva y movernos a ella en una única acción. Con checkout main lo que hacemos es movernos de nuevo a la rama main.