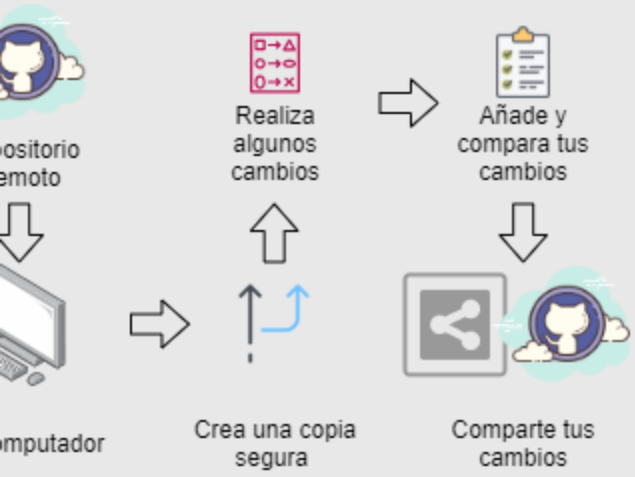
abajo colaborativo con GIT y Github

royectos de código abierto se integran para brindar un esquema de trabajo colaborativo y reproducible. En esta breve guía encontrarás comando úriles para manejar el de versones con GIT y el manejo de repositorios remotos que nos ofrece Github para el trabajo en equipo manteniendo un registro de la historia de nuestross proyectos.

ERGOSTAT

ema de control de versiones en un nte colaborativo



donde empezar?

git

formas de empezar con el control de versiones.

Desde una carpeta en tu computador

```
git init mi_carpeta/
```

e comando damos por iniciada la historia de nuestro o almacenado en una carpeta.

Desde un repositorio remoto

```
git clone <url al repostorio de git>
```

e comando traemos a nuestro computador toda la historia sitorio que estamos clonando.

ERGOSTATS

Paso a paso por el esquema

branch o version segura personal

Una vez que tenemos nuestra carpeta en la cual hemos implementado el control de versiones, lo siguiente es empezar por crear una copia segura del proyecto o **branch.**

```
git branch mi_copia_segura
```

Una branch es una copia exacta del proyecto en la cual podemos realizar modificaciones que sean luego integradas en el aspecto final de este.

Para pasar de una brach a otra usamos:

```
git checkout otroa_rama
```

Incluyendo cambios al proyecto

Los cambios que realices en un cambio en la carpeta se registraran en tres instancias:

Untracked files: Se trata de archivos que no estaban o están incluidos en la historia del proyecto.

Changes not staged for commit: Se trata de cambios que hemos realizado sobre archivos que están dentro de la historia del proyecto pero que aun están en desarrollo.

Changes staged for commit: Se trata de cambios realizado sobre archivos que están dentro de la historia del proyecto y que deseamos que se registren en la historia del proyecto.

Verificando el estado de los elementos del proyecto

El comando que debes tener presente para revisar en cual de las instancias mencionadas están los archivos sobre los que haz realizado cambios es el siguiente:

```
git status
```

¿Pero, qué es un commit?

Un commit es un punto en la historia del proyecto que quere preservar para futuras referencias. Todos aquellos cambios importantes deben registrarse con comentario para que nue compañeros puedan entender que hemos hecho.

Empezamos con añadir los archivos a la historia proyecto

Un archivo especifico en cualquier instan

```
git add my_archivo.txt
```

Solo los archivos modificados:

```
git add -u
```

Todos los archivos de la carpeta:

```
git add .
```

Registro de un estado del proyecto

```
git commit -m "Cambio import
```

Con un commit se genera un código SHA que nos permitira entre las distintas versiones de nuestro proyecto.

Compartir con tus compañeros

Git

Revisa el enlace de tu repositorio remoto.

```
git config --get remote.origin.url
```

Pon al día tu proyecto para tener la ultima versión remota:

```
git fetch
```

Actualiza tu carpeta local con la versión remota actualizada:

```
git pull origin <nombre_de_la_branch>
```

Sube tus cambios para que tus compañeros los puedan co e incluir en la versión definitiva del proyecto en un futuro.

```
git push origin <nombre_de_la_branch>
```