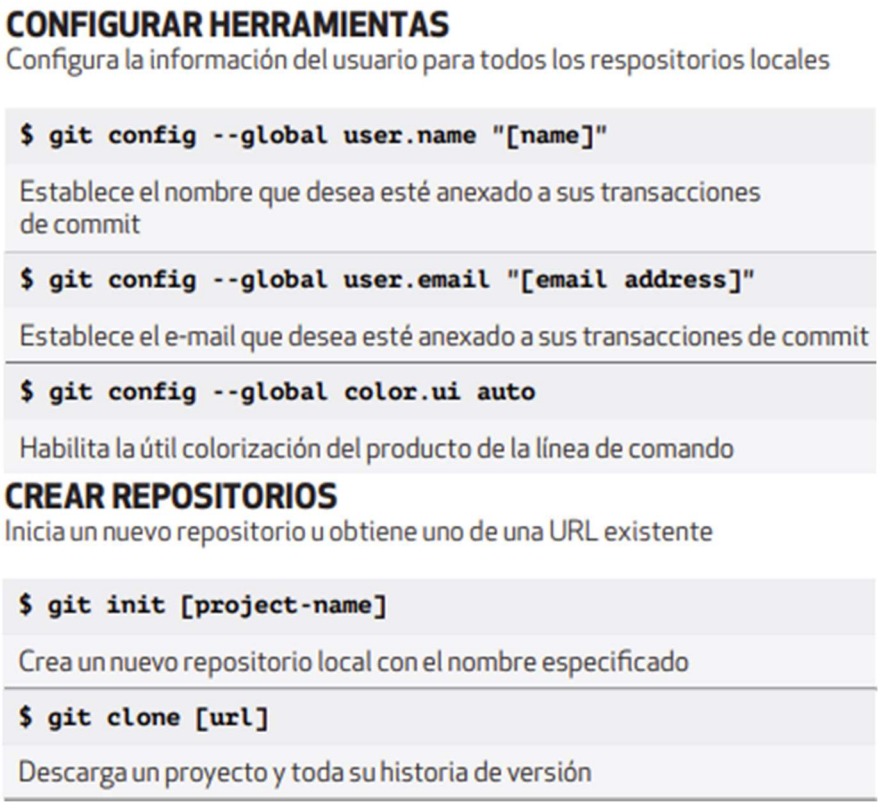
Introducción a Git.

¿Qué es? Git es un sistema de control de versiones que realiza un seguimiento de los cambios en los archivos. Git es especialmente útil cuando un grupo de personas y tú estáis haciendo cambios en los mismos archivos al mismo tiempo.

Normalmente, para hacerlo en un flujo de trabajo basado en Git, harías lo siguiente:

- **Crear una rama** a partir de la copia principal de archivos en los que tú (y tus colaboradores) estén trabajando.
- **Realizar modificaciones** en los archivos de forma independiente y segura en tu propia rama personal.
- Dejar que Git **fusione mediante combinación** y de forma inteligente los cambios específicos en la copia principal de archivos, de modo que los cambios no afecten a las actualizaciones de otras personas.
- Dejar que Git **realice un seguimiento** de tus cambios y los de otras personas, por lo que todos siguen trabajando en la versión más actualizada del proyecto.



EFECTUAR CAMBIOS

Revisa las ediciones y elabora una transacción de commit

$ git status
Enumera todos los archivos nuevos o modificados que se deben confirmar
$ git diff
Muestra las diferencias de archivos que no se han enviado aún al área de espera
$ git add [file]
Toma una instantánea del archivo para preparar la versión
$ git diff --staged
Muestra las diferencias del archivo entre el área de espera y la última versión del archivo
$ git reset [file]
Mueve el archivo del área de espera, pero preserva su contenido
$ git commit -m "[descriptive message]"
Registra las instantáneas del archivo permanentemente en el historial de versión

CAMBIOS GRUPALES

Nombra una serie de commits y combina esfuerzos ya culminados

$ git branch
Enumera todas las ramas en el repositorio actual
$ git branch [branch-name]
Crea una nueva rama
$ git checkout [branch-name]
Cambia a la rama especificada y actualiza el directorio activo
$ git merge [branch]
Combina el historial de la rama especificada con la rama actual
$ git branch -d [branch-name]
Borra la rama especificada

REPASAR HISTORIAL

Navega e inspecciona la evolución de los archivos de proyecto

`$ git log`

Enumera el historial de la versión para la rama actual

`$ git log --follow [file]`

Enumera el historial de versión para el archivo, incluidos los cambios de nombre

`$ git diff [first-branch]...[second-branch]`

Muestra las diferencias de contenido entre dos ramas

`$ git show [commit]`

Produce metadatos y cambios de contenido del commit especificado

REHACER COMMITS

Borra errores y elabora historial de reemplazo

`$ git reset [commit]`

Deshace todos los commits después de `[commit]`, preservando los cambios localmente

`$ git reset --hard [commit]`

Desecha todo el historial y regresa al commit especificado

SINCRONIZAR CAMBIOS

Registrar un marcador de repositorio e intercambiar historial de versión

`$ git fetch [bookmark]`

Descarga todo el historial del marcador del repositorio

`$ git merge [bookmark]/[branch]`

Combina la rama del marcador con la rama local actual

`$ git push [alias] [branch]`

Carga todos los commits de la rama local al GitHub

`$ git pull`

Descarga el historial del marcador e incorpora cambios