1. ¿Qué es git?

Es un sistema de control de versiones distribuido, lo que significa que un clon local del proyecto es un repositorio de control de versiones completo. Estos repositorios locales plenamente funcionales permiten trabajar sin conexión o de forma remota con facilidad.

1. ¿Qué es github?

Github es un portal creado para alojar el código de las aplicaciones de cualquier desarrollador, y que fue comprada por Microsoft [en junio del 2018](https://www.xataka.com/aplicaciones/oficial-microsoft-compra-github-7-500-millones-dolares). La plataforma está creada para que los desarrolladores suban el código de sus aplicaciones y herramientas, y que como usuario no sólo puedas descargarte la aplicación, sino también entrar a su perfil para leer sobre ella o colaborar con su desarrollo.

1. ¿Qué son los repositorios locales en git?

Un repositorio local en Git contiene todo el historial del proyecto, incluyendo todas las versiones de los archivos, los mensajes de los commits, y las ramas. Esto significa que puedes acceder a todas las versiones anteriores y hacer cambios en el proyecto sin estar conectado a un servidor remoto.

1. ¿Qué son los repositorios remotos en git?

Un repositorio remoto es una versión del proyecto que está alojado en un servidor externo o en la nube, en lugar de en tu máquina local. Los repositorios remotos permiten colaborar con otros desarrolladores y sincronizar el trabajo a través de la red.

1. ¿Como se crean los repositorios remotos en git?

Para agregar un repositorio remoto nuevo, use el comando git remote add en el terminal, dentro del directorio donde está almacenado su repositorio.

El comando git remote add toma dos argumentos: Un nombre remoto, por ejemplo, origin y una dirección URL remota, por ejemplo, https://github.com/OWNER/REPOSITORY.git

Por ejemplo:



1. ¿Como se crean los repositorios locales en git?

* Para crear un nuevo repositorio Git local, simplemente debe navegar hasta la carpeta donde desea crear el repositorio y ejecutar el siguiente comando en la línea de comandos:

git init

* Este comando creará un nuevo repositorio Git en la carpeta actual. Puede verificar que el repositorio se ha creado correctamente escribiendo el siguiente comando:

git status

* Ahora que ha creado un repositorio Git, puede añadir archivos y hacer cambios en esos archivos. Para hacerlo, debe crear archivos nuevos o modificar archivos existentes y luego ejecutar el siguiente comando:

git add archivo.txt

* Este comando añadirá el archivo especificado al índice de Git para realizar un seguimiento de los cambios que ha realizado. Si ha realizado cambios en varios archivos, puede ejecutar el siguiente comando para añadir todos los archivos al índice:

git add .

* Para confirmar los cambios realizados, debe realizar confirmaciones en el repositorio. Para hacerlo, debe ejecutar el siguiente comando:

git commit -m "Mensaje de confirmación"

* Este comando confirmará los cambios realizados en el repositorio y creará una nueva revisión en el historial de Git. También puede crear ramas en su repositorio local mediante el siguiente comando:

git branch nombre\_rama

* Este comando creará una nueva rama en su repositorio local. Puede cambiar entre ramas utilizando el siguiente comando:

git checkout nombre\_rama

* Este comando cambiará su repositorio local a la rama especificada.

1. ¿Que son las ramas(branches) en git?

Es una línea separada de desarrollo dentro de un proyecto. Las ramas te permiten trabajar en diferentes características, correcciones de errores, o experimentos de manera aislada sin afectar el código principal del proyecto. Cada rama puede tener su propio historial de commits, y puedes fusionar ramas entre sí cuando lo necesites.

1. Indique como se utilizan los siguientes comandos.

* Clone: se utiliza para crear una copia exacta (o "clon") de un repositorio remoto en tu máquina local. Es una de las formas más comunes de empezar a trabajar con un proyecto existente.
* Checkout: se utiliza para cambiar entre ramas, restaurar archivos a un estado específico, o incluso crear nuevas ramas. Es una herramienta muy versátil que se emplea principalmente para moverse dentro del historial de un proyecto o para trabajar en diferentes líneas de desarrollo.
* Pull: Se utiliza para actualizar tu repositorio local con los cambios realizados en un repositorio remoto. Combina dos comandos en uno: git fetch y git merge. Primero, recupera (fetch) los últimos cambios desde el repositorio remoto, y luego los fusiona (merge) en tu rama actual.
* Commit: se utiliza para guardar los cambios en el repositorio local. Un commit en Git es una instantánea de los archivos que están en el área de preparación (staging area). Estos commits forman el historial del proyecto, permitiendo que puedas volver a estados anteriores si es necesario.
* Push: se utiliza para enviar los commits de tu repositorio local a un repositorio remoto. Es una forma de compartir tus cambios con otros desarrolladores o de actualizar el código en un servidor remoto.
* Stash: se utiliza para guardar temporalmente los cambios que has realizado en tu área de trabajo y en el área de preparación (staging area) sin comprometerlos (commit). Esto es útil cuando necesitas cambiar de rama o realizar otra tarea, pero no estás listo para hacer un commit.