Curso de Git

Comandos básicos

Comandos de terminal

Pwd = te dice en que carpeta estas ubicado

Cd / = para moverte de carpeta

Cd . = es la carpeta en la que estas

Cd .. = te devuelve a la carpeta anterior

Con tabulación podes rellenar el nombre dependiendo de los archivos que tenga la carpeta ej: Us + TAB = User

Mkdir “nombre” = crea una carpeta con el nombre que indicaste

Touch vacio.txt = crea un archivo vacio

Cat vacio.txt = para ver el contenido del archivo

Ls = te muestra los archivos que tiene la carpeta

Ls -al = te muestra todos lso archivos incluso los ocultos

Ls l = te muestra todo pero no los ocultos

Ls -a = te muestra todo pero no en una lista

History = para ver los comandos que realizaste

¡+numero ej: ¡54 = ejecuta el código numero 54 del history

Rm vacio.txt = para borrar el archivo

--help = es para ver como funcionan todos los comandos

Comandos para github

Git init = Crea en la carpeta el repositorio

Git clone url = trae una copia del repositorio en línea

Git add = es para agregar archivos para mandar al repositorio, arranca el archivo

Git add . = agrega todos los archivos

Git rm = es para quitar el archivo

git rm --cached: Elimina los archivos de nuestro repositorio local y del área de staging, pero los mantiene en nuestro disco duro. Básicamente le dice a Git que deje de trackear el historial de cambios de estos archivos, por lo que pasaran a un estado untracked.

git rm --force: Elimina los archivos de Git y del disco duro. Git siempre guarda todo, por lo que podemos acceder al registro de la existencia de los archivos, de modo que podremos recuperarlos si es necesario (pero debemos usar comandos más avanzados).

Git commit -m “mensaje”= es el que envía los últimos cambios al archivo al repositorio

Git commit -am “mensaje”= agrega los archivos previamente creados al staggin yenvía los últimos cambios al archivo al repositorio

Git status = ver el estado del repositorio

Git show = te muestra todos los cambios hechos(diff te muestra los cambios que hubo entre los commits)

Git log archivo.ej = te muestra todos los cambios del archivo(los números son el nombre, y después en que rama lo haces y el commit que hiciste)

Git diff +numerodecommit + el numero de commit que queres comparar = te devuelve la diferencia que hubo

Git push = enviar hacia otro repositorio remoto

Git fetch = trae una actualización que hicieron en el repositorio remoto

git merge = une la actualización que traje con el fetch con la que ya tenia

Git pull = trae los archivos del repositorio y los fuciona con los que tengo

Git reset numerodecommit –hard = Vuelve todo a como estaba en ese commit

Git reset numerodecommit –soft = Vuelve todo a como estaba pero lo que esta en stagin sigue estando ahí

git reset HEAD: Este es el comando para sacar archivos del área de staging. No para borrarlos ni nada de eso, solo para que los últimos cambios de estos archivos no se envíen al último commit, a menos que cambiemos de opinión y los incluyamos de nuevo en staging con git add, por supuesto

Git checkout = para traer

Git branch = trae todas las ramas que existen

Git branch “nombre” = crea una rama con el nombre

git remote add origin master “link” = para conectar el repositorio local con el remoto

git push origin master

Git config = trae todas las configuraciones que trae

Git config –list= ves las configuraciones por defecto

Git config –list -- --show-origin = donde estan las configuraciones guardadas

Git config –global usar.name “nombre” = cambiar el nombre

Git config –global usar.email “nombre” = cambiar el email

Teoria

Git es para usar de manera local y github es para hacer trabajos en red

Diferencias entre archivos txt (texto plano) y binario(texto y mas cosas)

Staging : Es la memoria ram, estado temporal donde agregas archivos cuando , si un archivo esta aca se dice que esta trackeado (cuando le das add)

Branch = las ramas se crean para modificar el código y que no se modifique en la rama principal(master)

Las ramas mas comunes para crear son development(que es la parte experimental que luego vas a fucionar) y la hotfix(son los cambios que tienen que salir ya mismo)

Merge se usa para mezclar ramas