**GIT Y GITHUB**

git config –list –show-origin para que me muestre la cofiguracion del git

para salir de ahí presionamos q

protocol de comunicacion ssh entre git y github

ssh-keygen -t rsa -b 4096 -C “correo electronico”

.ssh/id\_rsa

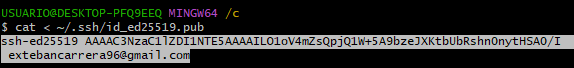
Y buscarlo en el explorador de archivos

Escriba ls -al ~/.ssh para ver si hay claves SSH existentes.

ssh-keygen -t rsa -b 4096 -C “[your\_email@example.com](mailto:your_email@example.com)”

luego vamos a git hub y en setting nos vamos a ssh and gpc keys

copiamos la clave del portapapeles o con un block de notas la abrimos



Para crear una carpeta en git colocamos el comando

mkdir nombreCarpeta

para clonar ese repositorio creado en github en mi pc colocamos el comando

git clone y la llave ssh de github del repositorio creado

el comando git status nos sirve para que nos muestre los cambios que estamos realizando

después utilizamos el comando git commit -m “una breve descripción del cambio que hemos hecho”

y con git commit -am ya se guardan los cambios esto es de manera local

cuando creamos un archivo ya sea un codigo de manera local en el computador tenemos los siguientes comandos para poderlo subir a github.

Git add nombreArchivo.extension, git commit -am “descripción del archivo”, git push origin main

Para actualizer cambios de github a mi computadora realizamos los siguientes comandos:

Primero para descargar los metadatos colocamos git fetch, y para actualizar colocamos git pull origin main.

Forma de actualizar datos de forma asíncrona

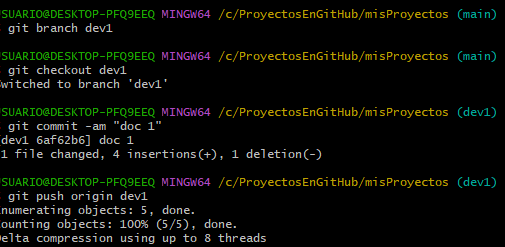
Git Branch nombrecualquiera = > se crea una línea nueva y no necesariamente ahí que descargar todo el main

Para moverme a esa línea hago el siguiente comando: git checkout nombreLinea

Hacemos el commit

Git commit -am “nombreCualquiera”

Git push origin nombreBrancj



Para regresarme al main hacemos lo siguiente

Git chekout main

Con git checkout -B nombreBranch => lo creamos y lo movemos a ese Branch de una sola línea

GIT MERGE Y PULL REQUEST

Colocar lo del main a un Branch es:

Git merge main nombreBranch

**HISTORIA DE UN REPOSITORIO**

Git reset –hard \_> nos devuelve al ultimo commit del codigo que hicimos

Git checkout HEAD – main.c -> me hace el reset de un solo archivo no de todos

Git log -> historial de lo que se ha hecho

Git log –oneline -> información mas reducida

Git log –oneline –graph-> mirar la info un poco mas bonita

Comandos

ls -l nos da mas detallado lo de las descripciones de cada elemento o item

pwd describir la carpeta o imprimir donde estoy parado actualmente

con touch podemos crear un tipo de archivo ejemplo archivo.txt

para abrir un .txt podemos usar el editor de texto nano seguido del nombre del archivo

con cat hacemos un output sobre algún contenido

con archivo.txt **>** archivo2.txt -> estamos sobre escribiendo lo de el primer archivo al segundo

con archivo.txt **>>** archivo2.txt -> estamos añadiendo lo del primer archivo al segundo sin sobre escribirlo

mkdir carpeta -> crear carpetas

cp archivoOriginal.txt copia

cp archivo\*.txt copiarEnCarpeta -> cp es para copiar

mv -> para mover un archivo o tambn para cambiarle el nombre creando uno nuevo

rm-> remove

rm -rf para borrar una carpeta o directorio tener cuidado

este comando es para escribir un nombre o correo si no lo tenemos:

git config –global user.name “nombre”

git config –global user.email “el correo”

Estado 1: **git init:** crear. git es con: git init

**Estado 2: git add:** git add –all para agregar o guardar cambios de todos los archivos al mismo tiempo

Estado 3: **git commit: con el comando git commit debemos guardar de manera manual**

Escape :wq y enter

Si no es git commit -m ”el comentario que queramos proporcionar”

**Git log:** con el comando git log : vemos el identificador único de nuestros archivos y el autor fecha y demás.