COMANDOS GIT

Git init

* Crea el área de ensayo y el repositorio local.
* La forma sencilla de hacerlo es ir a la carpeta del proyecto y dar click derecho y hacer git bash here y ahí ejecutar el comando.

Git add

* **Git add(+ nombre de archivo y extensión)** le dice a git que dé seguimiento al archivo especificado, o sea, lo pone en el área de ensayo (staging área)
* **Git add .** agrega todos los archivos del proyecto al staging área.

Git commit

* Respalda los archivos que están en seguimiento en el área de ensayo finalmente, llevándolos al repositorio local, dándoles un código de referencia.
* **Git commit –m** Permite respaldar los archivos y agregar una descripción de ese respaldo en texto. Ejemplo: git commit –m “Version 1.4 se agregaron los botones enviar y borrar”
* **Git commit -am** Permite hacer el add y el commit con mensaje en el mismo comando. Ej. Git commit –am ¨ver 1.1 agregado el título y la barra de navegación¨
* **Git commit --amend** Abre el editor VIM que nos permite, entre otras cosas, editar el mensaje de un commit. Primero se introduce **i:** y se le da enter, esto permite editar, hecho esto, podemos borrar el mensaje que estaba con supr, luego, con Esc se sale del modo edición, otra vez se introduce **i:** y enter para volver al modo edición y se escribe el nuevo mensaje, se le da enter y luego, con Esc se sale otra vez del modo edición. Cuando está todo listo, se escribe **:wq** y se le da enter, asi se guardan cambios y se sale del editor VIM

Git status

* **Git status –s** nos muestra la lista de archivos y directorios en la carpeta del proyecto con sus respectivos status de seguimiento. El status se representa con símbolos o letras antes de cada nombre de archivo.
* **??** El archivo o directorio no está bajo seguimiento (no fue movido al staging área)
* **A** Archivo o directorio en seguimiento
* **M** Archivo modificado no respaldado y no en staging área.

Git log

* **Git log –oneline** me muestra todas las versiones de los archivos que ya están en el repositorio ya que el comando **status** se les hace el **commit**. También los muestra con su respectivo código que git les otorga automáticamente y que se usa para volver a una versión anterior.

Git reset

* **Git reset –hard (mas código de la versión del archivo)**. Con este comando se regresa a la versión del archivo que se especifica. Ej. Git reset –hard e343j32m
* **Git reset (mas nombre de archivo y extensión)** quita el archivo del staging área conservando su contenido.

Git push

* **Git push** envía los cambios a remoto

Git pull

* **Git pull** Actualiza en local desde remoto.

Git tag

* **Git tag (nombre que le queremos dar)–m** crea una tag del proyecto con un mensaje. Ej Git tag 19-07-2022v1 –m “Primer versión del proyecto”

Git clone

* **Git clone (+ url del proyecto)** obtiene una copia del proyecto. La url se obtiene en github en el botón CODE. También se puede descargar el proyecto en zip desde el mismo lugar.