UNIVERSIDAD DE SAN CARLOS DE GUATEMALA
FACULTAD DE INGENIERIA

AMABILI ROSISABEL AGUILÓN GARCÍA
CARNÉ: 3277836211101

HOMEWORK 1

¿Qué es GIT?

Es un sistema que permite el control de versiones de softwares, es moderno y cuenta con un amplio campo de apoyo para dar la introducción a nuevos usuarios. Este sistema se diferencia de los sistemas antiguos debido al uso de una arquitecta estructurada lo que provoca que se le denomine DVCS (sistema de control de versiones distribuido, por sus siglas en inglés), esto permite la creación de copias de trabajo para cada desarrollador, lo que genera un repositorio que puede almacenar el historial completo de todos los cambios.

Control de versiones con GIT

Este sistema permite registrar los cambios realizados en uno o varios archivos durante el tiempo que ha sido modificado, para poder recuperar versiones específicas más adelante. Lo que implica que si algún archivo se pierda o arruina será posible la recuperación de forma más fácil.

Existen distintas versiones como el Sistema de Control de Versiones Locales, Sistema de Control de Versiones Centralizados y Sistema de Control de Versiones Distribuidos. El primero es copiar los archivos a otro directorio, a pesar de ser un método común por ser sencillo, es propenso a errores; el segundo sirve para solucionar el problema de la colaboración entre desarrolladores; el tercero ofrece replicar completamente un repositorio, para poder recuperarlo en caso de dejar de funcionar.

Estados de archivos en GIT

GIT presenta tres estados principales en los que se pueden encontrar tus archivos: **confirmado (committed), modificado (modified), y preparado (staged)**. El estado confirmado significa que los datos están almacenados de forma segura en la base de datos de GIT; el estado modificado representa a los archivos que se han modificado, pero todavía no ha sido confirmado a la base de datos de GIT; el estado preparado significa que se ha marcado un archivo modificado en su versión actual para que vaya en su próxima confirmación.

Configurar un repositorio

Primero se define repositorio de GIT como un almacenamiento virtual de un proyecto, lo que permite guardar versiones del código a las que puedes acceder cuando sea necesario.

Para crear el nuevo repositorio, se debe utilizar el comando `git init`, este comando se utiliza solo una vez durante la configuración inicial. Al ejecutar `git init`, se creará un nuevo subdirectorio .git en el directorio de trabajo actual y también una nueva rama principal.

Comandos en GIT

A continuación, se presenta un listado de comandos utilizados en GIT:

- **git add:** mueve los cambios del directorio de trabajo al área del entorno de ensayo.
- **rama de git:** permite crear entornos de desarrollo aislados en un solo repositorio.
- **git checkout:** además de extraer las confirmaciones y las revisiones de archivos antiguas, git checkout también sirve para navegar por las ramas existentes.
- **git clean:** elimina los archivos sin seguimiento del directorio de trabajo.
- **git clone:** crea una copia de un repositorio de Git existente.
- **git commit:** confirma la instantánea preparada en el historial del proyecto.
- **git commit –amend:** permite modificar la confirmación más reciente.
- **git config:** este comando va bien para establecer las opciones de configuración para instalar Git.
- **git fetch:** con este comando, se descarga una rama de otro repositorio junto con todas sus confirmaciones y archivos asociados.
- **git init:** inicializa un nuevo repositorio de Git.
- **git log:** permite explorar las revisiones anteriores de un proyecto.
- **Git merge:** es una forma eficaz de integrar los cambios de ramas divergentes.
- **git pull:** este comando es la versión automatizada de git fetch. Descarga una rama de un repositorio remoto e inmediatamente la fusiona en la rama actual.
- **git push:** permite mover una o varias ramas a otro repositorio, lo que es una buena forma de publicar contribuciones.
- **git rebase:** un cambio de base con git rebase permite mover las ramas, lo que ayuda a evitar confirmaciones de fusión innecesarias.
- **git rebase -i:** la marca -i se usa para iniciar una sesión de cambio de base interactivo.
- **git reflog:** Git realiza el seguimiento de las actualizaciones en el extremo de las ramas mediante un mecanismo llamado registro de referencia o reflog.
- **git remote:** en lugar de pasar la URL completa a los comandos fetch, pull y push, permite usar un atajo más significativo.
- **git reset:** el restablecimiento permite limpiar o eliminar por completo los cambios que no se han enviado a un repositorio público.
- **git revert:** permite deshacer una instantánea confirmada.
- **git status:** muestra el estado del directorio en el que estás trabajando y la instantánea preparada.