PRESNETACIÓN

Nombres:

Enmanuel Eduardo

Apellidos:

Heredia del rosario

Matrícula:

2021-1938

Asignatura:

Programación tres

Docente:

Kely Tejada

DESARROLLA EL SIGUIENTE CUESTIONARIO

1. ¿QUÉ ES GIT?

Git es un sistema de control de versiones distribuido diseñado para rastrear cambios en archivos y coordinar el trabajo de equipos de desarrollo de software. 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.

Permite a los desarrolladores trabajar en un mismo proyecto de forma colaborativa y mantener un historial detallado de todos los cambios realizados en el código a lo largo del tiempo. Git es ampliamente utilizado en la industria del desarrollo de software debido a su eficiencia, escalabilidad y capacidad para trabajar tanto en entornos locales como remotos. Los desarrolladores confirman su trabajo localmente y, a continuación, sincronizan su copia del repositorio con la copia en el servidor. Este paradigma es distinto del control de versiones centralizado, donde los clientes deben sincronizar el código con un servidor antes de crear nuevas versiones.

2. ¿PARA QUÉ FUNCIONA EL COMANDO GIT INIT?

El comando Git init se utiliza para crear un nuevo repositorio Git en un directorio vacío o en uno existente que aún no está siendo rastreado por Git. Creando de esta manera una estructura de directorios y archivos ocultos en el directorio de trabajo del proyecto, que permite a Git rastrear los cambios en los archivos que agregas al repositorio.

Git init es un comando que se utiliza una sola vez durante la configuración inicial de un repositorio nuevo. Al ejecutar este comando, se creará un nuevo subdirectorio. git en tu directorio de trabajo actual. También se creará una nueva rama principal.

3. ¿QUÉ ES UNA RAMA EN GIT?

Una rama en Git es una línea independiente de desarrollo que permite a los desarrolladores trabajar en nuevas características, solucionar problemas o realizar cambios en el código sin afectar directamente la rama principal del proyecto, conocida como "rama principal" o "rama maestra". Las ramas facilitan la colaboración en el desarrollo, ya que cada miembro del equipo puede trabajar en su propia rama y luego fusionar sus cambios en la rama principal una vez que estén listos y probados.

En Git, las ramas son parte del proceso de desarrollo diario. Las ramas de Git son un puntero eficaz para las instantáneas de tus cambios. Cuando quieres añadir una nueva función o solucionar un error, independientemente de su tamaño, generas una nueva rama para alojar estos cambios.

4. ¿CÓMO SABER EN QUÉ RAMA ESTOY EN GIT?

Con el comando git Branch, el cual despliega un listado en consola con tadas las ramas creadas resaltando en la que estas colocado, ademad de que el git Bash, que es la consola de git, te muestra en que rama estas directamente.

5. ¿QUIÉN CREÓ GIT?

Git fue creado por Linus Torvalds en 2005. Linus es conocido por ser el creador del kernel de Linux, y desarrolló Git para ayudar a la comunidad de desarrollo de Linux a gestionar el gran volumen de contribuciones y cambios que recibía el proyecto.

6. ¿CUÁLES SON LOS COMANDOS MÁS ESENCIALES DE GIT?

- **git init:** Inicializa un nuevo repositorio Git.
- **git add:** Agrega cambios al área de preparación (staging area) para ser incluidos en el próximo commit.
- **git commit:** Crea un nuevo commit con los cambios agregados al área de preparación.
- **git status:** Muestra el estado actual del repositorio, incluyendo los archivos modificados y pendientes de commit.
- **git log:** Muestra el historial de commits realizados en el repositorio.
- **git branch:** Muestra una lista de ramas y resalta la rama actual.
- **git checkout:** Cambia de rama o restaura archivos a versiones anteriores.
- **git merge:** Fusiona cambios de una rama en otra.
- **git pull:** Descarga cambios del repositorio remoto y los fusiona con la rama actual.
- **git push:** Envía cambios locales al repositorio remoto.

7. ¿QUÉ ES GIT FLOW?

Git Flow es un conjunto de prácticas y reglas para el uso de Git en un flujo de trabajo específico que facilita la colaboración y gestión del desarrollo de software. Fue popularizado por Vincent Driessen en su publicación "A successful Git branching model". Git Flow define dos ramas principales: `master` y `develop`, y propone el uso de ramas adicionales para características (`feature`), versiones (`release`), correcciones (`hotfix`) y soporte (`support`). Esta estructura de ramas proporciona un flujo de trabajo ordenado y bien definido para el desarrollo de software.

Gitflow es un modelo alternativo de creación de ramas en Git en el que se utilizan ramas de función y varias ramas principales. Fue Vincent Driessen en nvie quien lo publicó por primera vez y quien lo popularizó.

8. ¿QUÉ ES TRUNK BASED DEVELOPMENT?

Trunk Based Development es un enfoque de desarrollo de software en el que todos los cambios se realizan directamente en la rama principal del repositorio, como `master` o `main`, en lugar de utilizar ramas de características a largo plazo. Los desarrolladores trabajan en cambios pequeños y frecuentes, realizando commits directamente a la rama principal después de una revisión de código adecuada. El objetivo de TBD es minimizar la complejidad del desarrollo y reducir el tiempo entre la implementación y la puesta en producción de nuevas características o correcciones de errores. Requiere prácticas sólidas de pruebas automáticas y revisiones de código para garantizar la estabilidad y calidad del código en la rama principal. Es una práctica de gestión de control de versiones en la que los desarrolladores fusionan pequeñas actualizaciones de forma frecuente en un "tronco" o rama principal (main).