Estudiante:
Anthony Volquez Peña

Matrícula:

2022-0264

Profesor:

Kelyn Tejada

Materia:

Programacion lll

Tema:

Git

Desarrolla el siguiente Cuestionario

1-Que es Git?

Git es un software de control de versiones diseñado por Linus Torvalds, pensando en la eficiencia, la confiabilidad y compatibilidad del mantenimiento de versiones de aplicaciones cuando estas tienen un gran número de archivos de código fuente. Su propósito es llevar registro de los cambios en archivos de computadora incluyendo coordinar el trabajo que varias personas realizan sobre archivos compartidos en un repositorio de código.

2-Para que funciona el comando Git init?

El comando git init crea un nuevo repositorio de Git. Puede utilizarse para convertir un proyecto existente y sin versión en un repositorio de Git, o para inicializar un nuevo repositorio vacío. La mayoría de los demás comandos de Git no se encuentran disponibles fuera de un repositorio inicializado, por lo que este suele ser el primer comando que se ejecuta en un proyecto nuevo.

3-Que es una rama?

Las ramas en git son una división del estado del código, esto permite crear nuevos caminos a favor de la evolución del código.

4-Como saber es que rama estoy?

Para saber qué ramas están disponibles y cuál es el nombre de la rama actual, ejecuta git Branch.

5-Quien creo git?

Linus Torvalds

6-Cuales son los comandos más esenciales de Git?

1. **Git clone**

Git clone es un comando para descargarte el código fuente existente desde un repositorio remoto (como Github, por ejemplo). En otras palabras, Git clone básicamente realiza una copia idéntica de la última versión de un proyecto en un repositorio y la guarda en tu ordenador.

2. **Git branch**

Las ramas (branch) son altamente importantes en el mundo de Git. Usando ramas, varios desarrolladores pueden trabajar en paralelo en el mismo proyecto simultáneamente. Podemos usar el comando git branch para crearlas, listarlas y eliminarlas.

3. Git checkout

Este es también uno de los comandos más utilizados en Git. Para trabajar en una rama, primero tienes que cambiarte a ella. Usaremos git checkout principalmente para cambiarte de una rama a otra. También lo podemos usar para chequear archivos y commits.

4. Git status

El comando de git status nos da toda la información necesaria sobre la rama actual.

5. Git add

Cuando creamos, modificamos o eliminamos un archivo, estos cambios suceden en local y no se incluirán en el siguiente commit (a menos que cambiemos la configuración).

6. Git commit

Git commit es como establecer un punto de control en el proceso de desarrollo al cual puedes volver más tarde si es necesario.

También necesitamos escribir un mensaje corto para explicar qué hemos desarrollado o modificado en el código fuente.

7. Git push

Git push envía tus commits al repositorio remoto.

8. Git pull

El comando git pull se utiliza para recibir actualizaciones del repositorio remoto. Este comando es una combinación del git fetch y del git merge lo cual significa que cundo usemos el git pull recogeremos actualizaciones del repositorio remoto (git fetch) e inmediatamente aplicamos estos últimos cambios en local (git merge).

9. Git revert

A veces, necesitaremos deshacer los cambios que hemos hecho. Hay varias maneras para deshacer nuestros cambios en local y/o en remoto (dependiendo de lo que necesitemos), pero necesitaremos utilizar cuidadosamente estos comandos para evitar borrados no deseados.

10. Git merge

Git merge básicamente integra las características de tu rama con todos los commits realizados a las ramas dev (o master). Es importante que recuerdes que tienes que estar en esa rama específica que quieres fusionar con tu rama de características.

7-Que es git Flow?

GitFlow es un modelo de ramificación y flujo de trabajo para el sistema de control de versiones Git, propuesto por Vincent Driessen en 2010. Este modelo ha sido ampliamente adoptado y se ha convertido en una metodología popular para organizar y gestionar el desarrollo de proyectos de software.

El objetivo principal de GitFlow es proporcionar una estructura clara y bien definida para el flujo de trabajo en equipos de desarrollo colaborativo. Propone un conjunto de reglas y convenciones sobre cómo deben gestionarse las ramas y cómo deben integrarse las diferentes características y versiones del software.

El flujo de trabajo de GitFlow se basa en dos ramas principales:

- **Master:** Esta rama representa siempre la versión estable y desplegable del software. Aquí se encuentran los commits que han sido probados y considerados listos para ser lanzados en producción.

- **Develop:** Es la rama de integración donde se unen las diferentes características o desarrollos individuales que se van a incluir en la próxima versión. A medida que se desarrollan nuevas características, se unen a esta rama para ser probadas en conjunto antes de pasar a la rama master.

8-Que es trunk based development ?

Es una estrategia de desarrollo de software que se enfoca en mantener una rama principal (trunk) estable y en constante evolución, siendo la rama principal la principal línea de desarrollo del proyecto. Es una práctica que se opone al desarrollo en ramas largas y aisladas, donde los cambios se mantienen separados por períodos prolongados antes de ser integrados en la rama principal.