Estudiante: Etchegoyen Gabriel

Práctico 2: Git y GitHub

Objetivo:

El estudiante desarrollará competencias para trabajar con Git y GitHub, aplicando conceptos fundamentales de control de versiones, colaboración en proyectos y resolución de conflictos, en un entorno simulado y guiado.

Resultados de aprendizaje:

1. Comprender los conceptos básicos de Git y GitHub: Identificar y explicar los principales términos y procesos asociados con Git y GitHub, como repositorios, ramas, commits, forks, etiquetas y repositorios remotos.
2. Manejar comandos esenciales de Git: Ejecutar comandos básicos para crear, modificar, fusionar y gestionar ramas, commits y repositorios, tanto en local como en remoto.
3. Aplicar técnicas de colaboración en GitHub: Configurar y utilizar repositorios remotos, realizar forks, y gestionar pull requests para facilitar el trabajo colaborativo.
4. Resolver conflictos en un entorno de control de versiones: Identificar, analizar y solucionar conflictos de merge generados en un flujo de trabajo con múltiples ramas.

Actividades
1) Contestar las siguientes preguntas utilizando las guías y documentación proporcionada (Desarrollar las respuestas)

1. ¿Qué es GitHub?
GitHub es una plataforma en línea basada en Git que permite alojar repositorios de código, colaborar en proyectos, gestionar versiones y facilitar el trabajo en equipo mediante herramientas como pull requests, issues y forks.

2. ¿Cómo crear un repositorio en GitHub?
Inicia sesión en GitHub o crear una cuenta si no tenemos una.
Hacer clic en "New" o "Create repository".
Asignar un nombre, descripción (opcional) y elegir su visibilidad (público/privado).
Opcionalmente, marca "Initialize this repository with a README".(recomendado).
Hacer clic en "Create repository".

3. ¿Cómo crear una rama en Git?
Utilizamos el comando - git branch nombre de la rama a crear -.

4.¿Cómo cambiar a una rama en Git?
Utilizando el comando - git checkout nombre de la rama a la que deseamos movernos - .

5.¿Cómo fusionar ramas en Git?
Primero nos posicionamos en la rama de nuestro repositorio (- git checkout rama destino -) que recibirá los cambios y luego mediante el comando - git merge rama a fusionar - haremos la fusión entre ambas.

6. ¿Cómo crear un commit en Git?
Primero agregamos los cambios mediante el comando - git add . - , luego mediante el comando - git commit -m “descripción de commit o cambios”

7. ¿Cómo enviar un commit a GitHub?

Una vez realizado el commit utilizamos el comando -git push origin rama destino - para enviarlo a nuestro repositorio remoto previamente creado.

8. ¿Qué es un repositorio remoto?
Un repositorio remoto es una versión de tu proyecto que está alojada en un servidor (como GitHub) y que puede ser compartida con otros colaboradores.

9. ¿Cómo agregar un repositorio remoto a Git?
Para agregar un repositorio remoto a Git, se utiliza el comando git remote add seguido de un nombre (generalmente "origin") y la URL del repositorio remoto. Esto establece una conexión entre tu repositorio local y el repositorio en el servidor remoto.

10. ¿Cómo empujar cambios a un repositorio remoto?
Para enviar cambios locales a un repositorio remoto, se usa el comando git push seguido del nombre del remoto (usualmente "origin") y el nombre de la rama. Esto sube tus commits locales al repositorio remoto.

11. ¿Cómo tirar de cambios de un repositorio remoto?
Para obtener los últimos cambios de un repositorio remoto y fusionarlos con tu rama local, se utiliza el comando git pull. Esto combina automáticamente los cambios remotos con tu versión local.

12. ¿Qué es un fork de repositorio?
Un fork es una copia personal de un repositorio ajeno que se aloja en tu propia cuenta de GitHub. Permite hacer cambios libremente sin afectar el proyecto original, y es el primer paso para contribuir a proyectos de código abierto.

13. ¿Cómo crear un fork de un repositorio?
Para crear un fork, navega al repositorio original en GitHub y haz clic en el botón "Fork" en la esquina superior derecha. Esto creará una copia idéntica en tu cuenta personal de GitHub.

14. ¿Cómo enviar una solicitud de extracción (pull request) a un repositorio?
Después de hacer cambios en tu fork y subirlos a GitHub, puedes enviar un pull request desde la página de tu fork haciendo clic en "Pull request", seleccionando las ramas adecuadas y describiendo tus cambios. Esto notifica a los mantenedores del proyecto original sobre tus modificaciones.

15. ¿Cómo aceptar una solicitud de extracción?
Para aceptar un pull request, los mantenedores del repositorio original deben revisar los cambios en la pestaña "Pull requests" y hacer clic en "Merge pull request" si están de acuerdo con las modificaciones propuestas.

16. ¿Qué es una etiqueta en Git?
Una etiqueta en Git es una marca estática que se usa para señalar puntos importantes en el historial, generalmente versiones de lanzamiento. A diferencia de las ramas, las etiquetas no cambian una vez creadas.

17. ¿Cómo crear una etiqueta en Git?
Para crear una etiqueta, se usa el comando git tag con un nombre de versión y opcionalmente un mensaje descriptivo. Las etiquetas anotadas son las más completas ya que almacenan información adicional.

18. ¿Cómo enviar una etiqueta a GitHub?
Las etiquetas locales no se comparten automáticamente con el repositorio remoto. Para enviar una etiqueta específica o todas las etiquetas a GitHub, se deben usar comandos push especiales.

19. ¿Qué es un historial de Git?
El historial de Git es el registro completo y ordenado de todos los commits realizados en un repositorio, mostrando la evolución del proyecto a lo largo del tiempo con información sobre autores, fechas y cambios.

20. ¿Cómo ver el historial de Git?
Para visualizar el historial se usa el comando git log, que muestra los commits en orden cronológico inverso. Existen múltiples opciones para personalizar el formato de visualización.

21. ¿Cómo buscar en el historial de Git?
Git ofrece varias formas de buscar en el historial, ya sea por mensaje de commit, autor, fecha o contenido de los cambios, utilizando diferentes parámetros con el comando git log.

22. ¿Cómo borrar el historial de Git?
Borrar el historial no es una operación común y requiere crear una nueva rama sin historial, eliminar la rama principal original y forzar la actualización del repositorio remoto mediante el comando - git push –force - . Esto debe hacerse con precaución.

23. ¿Qué es un repositorio privado en GitHub?
Un repositorio privado en GitHub es aquel que solo es visible y accesible para los propietarios y colaboradores explícitamente invitados, manteniendo el código confidencial.

24. ¿Cómo crear un repositorio privado en GitHub?
Al crear un nuevo repositorio en GitHub, se puede seleccionar la opción "Private" en la configuración de visibilidad. Los repositorios privados pueden requerir una suscripción paga dependiendo del plan.

25. ¿Cómo invitar a alguien a un repositorio privado en GitHub?
Desde la configuración del repositorio, en la sección "Collaborators", se pueden agregar usuarios ingresando sus nombres o correos electrónicos. Los invitados recibirán una notificación para aceptar la colaboración.

26. ¿Qué es un repositorio público en GitHub?
Un repositorio público es visible para cualquier usuario de internet, aunque solo los colaboradores pueden hacer cambios directos. Es la opción predilecta para proyectos de código abierto.

27. ¿Cómo crear un repositorio público en GitHub?
Al crear un nuevo repositorio, seleccionamos la opción "Public" en la configuración de visibilidad. Los repositorios públicos son gratuitos y no tienen límite de colaboradores.

28. ¿Cómo compartir un repositorio público en GitHub?
Los repositorios públicos pueden compartirse simplemente enviando el enlace URL del repositorio. GitHub también ofrece botones para compartir directamente en redes sociales o mediante enlaces especiales.