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) :

- ¿Qué es GitHub? Git hub es una herramienta para el control de versiones, sirve para poder ir moviendote en el tiempo dentro de tu proyecto, viendo cada modificacion o implementación a tu proyecto.
- ¿Cómo crear un repositorio en GitHub? Para poder crear tu repositorio tendras que hacer una cuenta de Github donde ahi almacenaras todos tus repositorios, una vez creada debes ir a repositorios y apretar el botón de NEW, de esa forma agregaras un nuevo repositorio.
- ¿Cómo crear una rama en Git? Para crear una rama en git debes aplicar el comando de git branch nombre-rama asi crearas una nueva rama.
- ¿Cómo cambiar a una rama en Git? Para cambiar de rama lo que tendrás que hacer es aplicar el comando git checkout nombre-rama asi te cambiaras de ramas.
- ¿Cómo fusionar ramas en Git? Para fusionar ramas tendrás que aplicar el comando de git merge y el nombre de la rama que quieres fusionar a la rama en la que estas situado.
- ¿Cómo crear un commit en Git? Para crear un commit tendrás que realizar el siguiente comando commit -m "Texto de commit", con esto agregaras un commit en tu rama para saber que modificación realizaste.
- ¿Cómo enviar un commit a GitHub? Para enviar un commit a GitHub lo que tendrás que realizar sera un git push origin nombre-rama, esto lo que hace es "empujar" el archivo de tu rama hacia el repositorio remoto de tu GitHub.
- ¿Qué es un repositorio remoto? Los repositorios remotos son versiones de tu proyecto que están hospedadas en Internet.
- ¿Cómo agregar un repositorio remoto a Git? Para agregar un repositorio remoto tienes que utilizar un comando git remote add y a continuación un nombre remoto y después la url del repositorio remoto.
- ¿Cómo empujar cambios a un repositorio remoto? Con el comando git push nombre-repo nombre-rama empujas los cambios al repositorio
- ¿Cómo tirar de cambios de un repositorio remoto? Con el comando git revert deshaces los cambios del repositorio remoto.
- ¿Qué es un fork de repositorio? El fork lo que hace es realizar una copia de un repositorio de GitHub.
- ¿Cómo crear un fork de un repositorio? Debes ir al repositorio requerido y luego en la parte de arriba te aparece fork le haces click y automaticamente te lo copia.

- ¿Cómo enviar una solicitud de extracción (pull request) a un repositorio? Para enviar una solicitud de incorporación de cambios (pull request) se debe indicar la rama y repositorio de origen y destino. Esto permite que un desarrollador incorpore los cambios en su proyecto.
- ¿Cómo aceptar una solicitud de extracción? Primero tienes que entrar al repositorio y ir a la sección de pull request, luego dentro veras todas las solicitudes de extracción, y podras fusionar con la rama principal si es que lo deseas asi.
- ¿Qué es un etiqueta en Git? En Git una etiqueta o tag sirve básicamente como una rama firmada que no permuta, es decir, siempre se mantiene inalterable
- ¿Cómo crear una etiqueta en Git? Para crear una etiqueta en git debes poner el comando git tag nombre-tag
- ¿Cómo enviar una etiqueta a GitHub? Para enviar la etiqueta utilizamos el comando de git push origin tag-name
- ¿Qué es un historial de Git? Cada rama tiene un historial de commits, para listar los commits como vista del historial de una rama, puedes usar el comando git log
- ¿Cómo ver el historial de Git? Con el comando git log --oneline veras los commits por linea o git log --decorate --all --graph --oneline lo veras de forma graficada
- ¿Cómo buscar en el historial de Git?
  - ¿Cómo borrar el historial de Git? Primero se realiza el check-out en una rama temporal, despuesse agrega todos los archivos, se confirman los cambios en el historial de confirmaciones. Después eliminas la rama principaly cambias el nombre de la rama temporal a principal y actualizas nuestro repositorio Git.
- ¿Qué es un repositorio privado en GitHub? Los repositorios privados solo son repositorios accesibles para usted, las personas con las que comparte acceso explícitamente y, en el caso de los repositorios de la organización, para ciertos miembros de la misma .
- ¿Cómo crear un repositorio privado en GitHub? Seria lo mismo que uno publico solamente que a la hora de realizar la creación de este se elija la opción de que sea privado
- ¿Cómo invitar a alguien a un repositorio privado en GitHub? Una vez dentro del repositorio vas a configuraciones y seleccionas la opción de colaboradores.
- ¿Qué es un repositorio público en GitHub? Los repositorios públicos de GitHub son accesibles para todos en internet.
- ¿Cómo crear un repositorio público en GitHub? Vas a la sección de repositorios y agregas uno y le colocas la opción de que sea publico
- ¿Cómo compartir un repositorio público en GitHub? Para invitar a personas vas ajustes dentro del repositorio y luego a colaboradores, ahi podras a invitar a las personas que quieras.

2) Realizar la siguiente actividad:

- Crear un repositorio.
  - Dale un nombre al repositorio.
  - Elije el repositorio sea público.
  - Inicializa el repositorio con un archivo.
- Agregando un Archivo
  - Crea un archivo simple, por ejemplo, "mi-archivo.txt".
  - Realiza los comandos git add . y git commit -m "Agregando mi-archivo.txt" en la línea de comandos.
  - Sube los cambios al repositorio en GitHub con git push origin main (o el nombre de la rama correspondiente).

- Creando Branchs

    - Crear una Branch
    - Realizar cambios o agregar un archivo
    - Subir la Branch

3) Realizar la siguiente actividad:

Paso 1: Crear un repositorio en GitHub

- Ve a GitHub e inicia sesión en tu cuenta.
- Haz clic en el botón "New" o "Create repository" para crear un nuevo repositorio.
- Asigna un nombre al repositorio, por ejemplo, conflict-exercise.
- Opcionalmente, añade una descripción.
- Marca la opción "Initialize this repository with a README".
- Haz clic en "Create repository".

Paso 2: Clonar el repositorio a tu máquina local

- Copia la URL del repositorio (usualmente algo como https://github.com/tuusuario/conflict-exercise.git).
- Abre la terminal o línea de comandos en tu máquina.
- Clona el repositorio usando el comando:

```
git clone https://github.com/tuusuario/conflict-exercise.git
```

- Entra en el directorio del repositorio:

```
cd conflict-exercise
```

Paso 3: Crear una nueva rama y editar un archivo

- Crea una nueva rama llamada feature-branch:

```
git checkout -b feature-branch
```

- Abre el archivo README.md en un editor de texto y añade una línea nueva, por ejemplo:

Este es un cambio en la feature branch.

- Guarda los cambios y haz un commit:

```
git add README.md
```

```
git commit -m "Added a line in feature-branch"
```

Paso 4: Volver a la rama principal y editar el mismo archivo

- Cambia de vuelta a la rama principal (main):

```
git checkout main
```

- Edita el archivo README.md de nuevo, añadiendo una línea diferente:

Este es un cambio en la main branch.

- Guarda los cambios y haz un commit:

```
git add README.md
```

```
git commit -m "Added a line in main branch"
```

Paso 5: Hacer un merge y generar un conflicto

- Intenta hacer un merge de la feature-branch en la rama main:

```
git merge feature-branch
```

- Se generará un conflicto porque ambos cambios afectan la misma línea del archivo README.md.

Paso 6: Resolver el conflicto

- Abre el archivo README.md en tu editor de texto. Verás algo similar a esto:

```
<<<<<<< HEAD
```

Este es un cambio en la main branch.

```
=======
```

Este es un cambio en la feature branch.

```
>>>>>>> feature-branch
```

- Decide cómo resolver el conflicto. Puedes mantener ambos cambios, elegir uno de ellos, o fusionar los contenidos de alguna manera.
- Edita el archivo para resolver el conflicto y guarda los cambios(Se debe borrar lo marcado en verde en el archivo donde estes solucionando el conflicto. Y se debe borrar la parte del texto que no se quiera dejar).
- Añade el archivo resuelto y completa el merge:

```
git add README.md
```

```
git commit -m "Resolved merge conflict"
```

Paso 7: Subir los cambios a GitHub

- Sube los cambios de la rama main al repositorio remoto en GitHub:

```
git push origin main
```

- También sube la feature-branch si deseas:

```
git push origin feature-branch
```

Paso 8: Verificar en GitHub

- Ve a tu repositorio en GitHub y revisa el archivo README.md para confirmar que los cambios se han subido correctamente.
- Puedes revisar el historial de commits para ver el conflicto y su resolución.