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?

Son carpetas virtuales donde se guarda todo el historial del código

- ¿Cómo crear un repositorio en GitHub?

En vc podes crear repositorio remoto con la extensión de github

- ¿Cómo crear una rama en Git?

Con git Branch y el nombre para la rama nueva

- ¿Cómo cambiar a una rama en Git?

Con git checkout y el nombre de la rama

- ¿Cómo fusionar ramas en Git?

Con git merge y el nombre de la rama a la que se desea tomar los cambios

- ¿Cómo crear un commit en Git?

Git comit -m “”

- ¿Cómo enviar un commit a GitHub?

Con Git comit -m “”y enter

- ¿Qué es un repositorio remoto?

Un control de versiones remoto

- ¿Cómo agregar un repositorio remoto a Git?

Sincronizando el repositorio local con uno ya existente o creando un nuevo repositorio

- ¿Cómo empujar cambios a un repositorio remoto?

Con git push

- ¿Cómo tirar de cambios de un repositorio remoto?

Con git pull

- ¿Qué es un fork de repositorio?

Es clonar el repositorio de otra persona en tu repositorio en git hub

- ¿Cómo crear un fork de un repositorio?

Vas al repositorio ageno apretas en fork y listo

- ¿Cómo enviar una solicitud de extracción (pull request) a un repositorio?

git push -u origin nueva-rama

- ¿Cómo aceptar una solicitud de extracción?

busca la solicitud de extracción que deseas aceptar y haz el merge pull request

- ¿Qué es un etiqueta en Git?

Permiten etiquetar commits como versiones específicas del software

- ¿Cómo crear una etiqueta en Git?

git tag -a

- ¿Cómo enviar una etiqueta a GitHub?

git push origin --tags

- ¿Qué es un historial de Git?

Es el registro de todos los cambios realizados en un proyecto

- ¿Cómo ver el historial de Git?

Git log

- ¿Cómo buscar en el historial de Git?

A través de comando como git log --author="Nombre"/git log --since="2023-01-01"

- ¿Cómo borrar el historial de Git?

git reset HEAD~n

- ¿Qué es un repositorio privado en GitHub?

No esta disponible para otros usuarios

- ¿Cómo crear un repositorio privado en GitHub?

Se crea un repositorio y se coloca como privado

- ¿Cómo invitar a alguien a un repositorio privado en GitHub?

Se agrega como colaborador

- ¿Qué es un repositorio público en GitHub?

Un repositorio en donde cualquier persona puede proponer cambios

- ¿Cómo crear un repositorio público en GitHub?

Se crea un repositorio y se coloca como publico

- ¿Cómo compartir un repositorio público en GitHub?

Mediante el link del repositorio

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 " https://github.com/mangellopezdeblasis/conflict-exercise.git 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.

https://github.com/mangellopezdeblasis/conflict-exercise.git