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?
  - GitHub es una plataforma tipo red social donde puedes subir a un repositorio tu código de forma pública o privada.
- ¿Cómo crear un repositorio en GitHub?
  - Se pueden crear desde la página de GitHub, en la parte superior derecha hay un botón (+), que nos permitirá crear un repositorio público o privado en Github.
- ¿Cómo crear una rama en Git?
  - Para crear una rama se usa el comando ***git branch 'Nombre de la rama'***, si solo usamos el comando ***git branch*** nos mostrará las ramas que existen.
- ¿Cómo cambiar a una rama en Git?
  - Para cambiar de rama se usa el comando ***git checkout 'Nombre de la rama a la que queremos movernos'***
- ¿Cómo fusionar ramas en Git?
  - Para fusionar ramas primero nos posicionamos en la rama donde queremos fusionar los cambios y luego usar el comando ***git merge 'Nombre de la rama que queremos fusionar'*** si hay conflictos en la fusion git nos pedirá resolverlos y en cuyo caso se resuelvan o no haya conflictos se generará con commit nuevo con la fusión.

- ¿Cómo crear un commit en Git?
  - Para crear un commit se usa el comando ***git commit -m 'Mensaje para marcar el comit'***
- ¿Cómo enviar un commit a GitHub?
  - Para subir el commit al repositorio de GitHub se usa el comando ***git push -u origin master*** (la primera vez, luego con solamente ***git push***)
- ¿Qué es un repositorio remoto?
  - Un repositorio remoto es un espacio en un servidor en línea donde puedo subir mis archivos de código, compartirlos con otras personas que también podrían descargarlo y modificarlo.
- ¿Cómo agregar un repositorio remoto a Git?
  - Para agregar un repositorio remoto al entorno local se usa el comando ***git remote add origin 'url del repositorio'***
- ¿Cómo empujar cambios a un repositorio remoto?
  - Para empujar/subir los cambios a un repositorio remoto se usa ***git push.***
- ¿Cómo tirar de cambios de un repositorio remoto?
  - Para traer los commits de un repositorio remoto al local usamos el comando ***git pull.***
- ¿Qué es un fork de repositorio?
  - Un Fork es copiar un repositorio de otra persona y pegarlo en tu perfil, para así realizar cambios sin afectar al repositorio de esa persona.
- ¿Cómo crear un fork de un repositorio?
  - En la página de GitHub cuando entras al proyecto de alguien mas podes hacer click a un boton arriba a la derecha con el símbolo de una bifurcación.

- ¿Cómo enviar una solicitud de extracción (pull request) a un repositorio?
  - Una vez hecho el push de los commits a GitHub, en la plataforma seleccionamos el botón 'New pull request' para enviar la solicitud.
- ¿Cómo aceptar una solicitud de extracción?
  - En la solicitud de pull request si la fusión de las ramas no genera conflictos te aparece un botón que dice 'Merge pull request' el cual al clickear procede a realizar el merge de ambas ramas.
- ¿Qué es un etiqueta en Git?
  - Una etiqueta es como un marcador que apunta hacia un commit concreto en el historial del repositorio, es inmutable y se suele utilizar para identificar versiones.
- ¿Cómo crear una etiqueta en Git?
  - Para crear una etiqueta se usa el comando ***git tag 'Nombre de la etiqueta'***
- ¿Cómo enviar una etiqueta a GitHub?
  - Para subir una etiqueta a GitHub se usa el comando ***git push origin tag 'Nombre de la etiqueta'***
- ¿Qué es un historial de Git?
  - Es un registro cronológico de todos los commits que se hicieron en el repositorio: Quien los hizo, qué cambios se hicieron, cuando se hicieron y porque se hicieron (si dejaron algún mensaje en el commit)
- ¿Cómo ver el historial de Git?
  - Con el comando ***git log*** podemos ver el historial de commits de la rama actual.
- ¿Cómo buscar en el historial de Git?
  - Con el comando ***git log - -oneline*** podemos ver los commits linea por linea, con el comando ***git log - -all*** podemos ver el historial de commits de todas las ramas, con el comando ***git log - -graph*** obtenemos un gráfico ASCII que muestra la estructura de ramas y fusiones, también está el comando ***git log - -decorate*** que nos devuelve las referencias (ramas, etiquetas, etc) que apuntan a cada commit, que también se puede usar en conjunto con los anteriores comandos (online, all, graph) para una mejor visualización.
- ¿Cómo borrar el historial de Git?
  - Para borrar el historial se usa el comando git reset, que nos permite regresar al commit que deseamos volver y borrar aquellos que se hayan hecho posterior a ese commit, tiene tres modos (soft, mix(por defecto), hard) donde cada una va a tratar de manera diferente esos cambios producidos después al commit que queremos llegar.
- ¿Qué es un repositorio privado en GitHub?
  - Es un espacio en un servidor en línea para almacenar archivos de código, pero el acceso al repositorio está solo permitido para el propietario o para las personas que este autorice.
- ¿Cómo crear un repositorio privado en GitHub?

- En la sección '**Your Repositories**' arriba a la derecha hay un botón '**New**' para crear repositorios y en las opciones de creación podemos seleccionar que sea privado.

- ¿Cómo invitar a alguien a un repositorio privado en GitHub?
  - Navegando en el repositorio en la parte superior ir a **'Settings'**, en el menú de la izquierda buscar '**Collaborators**' una vez ahí clickear el botón '**Add people**' y escribir el nombre de usuario de la persona que queramos invitar al repositorio.
- ¿Qué es un repositorio público en GitHub?
  - Es un espacio en un servidor en línea para almacenar archivos de código y al ser público, tiene acceso a él cualquier persona.
- ¿Cómo crear un repositorio público en GitHub?
  - En la sección '**Your Repositories**' arriba a la derecha hay un botón '**New**' para crear repositorios y en las opciones de creación podemos seleccionar que sea público.
- ¿Cómo compartir un repositorio público en GitHub?
  - Un repositorio público dada su característica de 'público' puede ser accedido por cualquier persona, también se puede compartir la **url** del repositorio y se puede invitar a colaborar en el mismo desde las opción '**add people**' como ya se mencionó antes.

2) Realizar la siguiente actividad:

- Crear un repositorio.
  - Dale un nombre al repositorio.
  - Elige que 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

Actividad Nro 2 = https://github.com/lonker96/Primer-Repo

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.

Actividad Nro3 = https://github.com/lonker96/conflict-exercise