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?
  Es una plataforma en donde los programadores pueden compartir sus repositorios de forma publica o privada.
- ¿Cómo crear un repositorio en GitHub?

  Primero se debe crear una cuenta en GitHub, debemos seleccionar + (nuevo repositorio), poner un nombre, y crear repositorio. Debemos antes inicializar proyecto con GIT, luego vincularlo al repositorio remoto y subir los archivos con GIT.

- ¿Cómo crear una rama en Git?

  Se ejecuta el comando Git Branch (Nos permitrá ver las ramas disponibles, y con * nos dice cual es la actual). Si queremos iniciar una nueva debemos poner Git Branch y el nombre de la nueva rama (este paso solo crea la rama, no quiere decir que cambie a ella). Para poder cambiar a la nueva rama se escribe Git Checkout y el nombre de la nueva rama.

  Con Git add . se guardan los cambios.

- ¿Cómo cambiar a una rama en Git?

Para cambiar de rama primero miramos cual tenemos en el repositorio con Gir Branch, luego con git checkout cambiamos de rama.

- ¿Cómo fusionar ramas en Git?
- Se utiliza Git merge para fusionar una o mas ramas dentro de la rama que está activa. Primero nos posicionamos en Git Checkout (la rama principal, donde queremos que queden los cambios). Para poder guardar los cambios de las ramas Gir merge y el nombre de la rama que queremos volcar a la rama principal.
- ¿Cómo crear un commit en Git?

  Primero debemos ver el estado del archivo con Git Status. Luego agregar los archivos a incluir en Gt add. Y al finalizar git commit -m con un mensaje descriptivo.
- ¿Cómo enviar un commit a GitHub?
  Una vez realizado el commit se ejecuta Gir push para subir a Git Hub
- ¿Qué es un repositorio remoto?

  Un repositorio remoto es el que se encuentra en el servidor central. Permite que varios colaboradores trabajen en el mismo proyecto en diferentes lugares.
- ¿Cómo agregar un repositorio remoto a Git?

  Para agregar un repositorio remoto nuevo, use el comando git remote add en el terminal, dentro del directorio donde está almacenado su repositorio.
- ¿Cómo empujar cambios a un repositorio remoto?

  Para empujar camios a un repositorio remoto se utiliza Git push
- ¿Cómo tirar de cambios de un repositorio remoto?

  Se utiliza Git pull
- ¿Qué es un fork de repositorio?

  Es una copia de un repositorio creada en una cuenta diferente permitiendo desarrollar cambios sin afectar el original. El fork se realiza generando una copia en la cuenta del usuario
- ¿Cómo crear un fork de un repositorio?

  Iniciar sesión en GitHub, buscar el repositorio, luego hacer click en Fork y se creará la copia.

- ¿Cómo enviar una solicitud de extracción (pull request) a un repositorio?
  Fork debe estar actualizado con Git pull. En Github abrir repositorio fork, seleccionar y hacer clik en compare y pull request
- ¿Cómo aceptar una solicitud de extracción?

  En el repositorio ir a la pestaña Pull requests, abrir la solicitud y revisar los cambios, luego hacer click en merge pull request y luego en confirm merge
- ¿Qué es un etiqueta en Git?

  Es un marcador en un commit especifico, generalmente utilizado para identificar versiones importantes.
- ¿Cómo crear una etiqueta en Git?

  Se utiliza Gir tag -1 – list
- ¿Cómo enviar una etiqueta a GitHub?

  Para enviar una etiqueta se puede usar Git Push
- ¿Qué es un historial de Git?

  Un historial de Git es el registro de todos los commits realizados en un repositorio.
- ¿Cómo ver el historial de Git?

  Para ver el historial Git se utiliza Git log
- ¿Cómo borrar el historial de Git?

  Se puede reiniciar el repositorio.
- ¿Qué es un repositorio privado en GitHub?

  Un repositorio privado es un repositorio que solo personas con permiso pueden ver y colaborar
- ¿Cómo crear un repositorio privado en GitHub?

  Cuando crees el repositorios debes hacer click en Private
- ¿Cómo invitar a alguien a un repositorio privado en GitHub?

  En la pestaña settings, luego collaborators y se agrega el nombre del usuario.
- ¿Qué es un repositorio público en GitHub?

  Un repositorio publico es un repositorio accesible por cualquier persona en GitHub
- ¿Cómo crear un repositorio público en GitHub?

  En la descripción se elije Public
- ¿Cómo compartir un repositorio público en GitHub?

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.