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 un controlador de versiones de nuestros proyectos basado en Git. Los programadores podemos almacenar, compartir y gestionar nuestro código.

- ¿Cómo crear un repositorio en GitHub?

  Ir a GitHub, iniciar sesión

  En la esquina superior derecha, hacer click en el botón + y seleccionar Nuevo Repositorio

  Elegir un nombre para el repositorio y si va a ser publico o privado

  Hacer clic en Crear Repositorio

- ¿Cómo crear una rama en Git?

  Con el comando de git branch ‘Nombre de la rama’

- ¿Cómo cambiar a una rama en Git?

  Con git checkout, puedes moverte entre ramas por github

- ¿Cómo fusionar ramas en Git?

  Si trabajaste en una rama y querés unir los cambios a la principal (main), podés hacer un merge con git checkout main y lurgo git merge mi-rama

- ¿Cómo crear un commit en Git?

  Con los comandos de git add . agrego los archivos al área de preparación y luego con git commit -m 'Mensaje del commit' creao el commit con un mensaje

- ¿Cómo enviar un commit a GitHub?

  Para conectar la primera vez con el repositorio de github debemos utilizar el comando de git push -u origin main, si no es la primera vez solo con git push

- ¿Qué es un repositorio remoto?

  Es un repositorio almacenado en GitHub o en otro servidor, permitiendo trabajar en equipo

- ¿Cómo agregar un repositorio remoto a Git?

  Utilizo el comando git remote add origin 'link al repositorio' puedo agregar un repositorio remoto a git

- ¿Cómo empujar cambios a un repositorio remoto?

  Para subir cambios utilizo git push origin main

- ¿Cómo tirar de cambios de un repositorio remoto?

  Con el comando git pull origin main

- ¿Qué es un fork de repositorio?

  Es una copia de un repositorio remoto de otra persona en mi cuenta de github, permitiéndome modificarlo si afectar el original.

- ¿Cómo crear un fork de un repositorio?

  Ingresando al repositorio que deseo hacer el fork

  Arriba a la derecha, hacer click en el btn fork

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

  Hacer cambios en el fork

Subir los cambios con git push

Ir al repositorio original en github y hacer click en ‘New pull request”.

Seleccionar la rama y enviar la solicitud

- ¿Cómo aceptar una solicitud de extracción?

  Ir a la pestaña de pull request en github

  Revisar los cambios

  Hacer click en ‘merge pull request’

  Confirmar con confirm merge

- ¿Qué es un etiqueta en Git?

  Un tag o etiqueta que se usa para marcar versiones específicas del código por ej: v1.0.2

- ¿Cómo crear una etiqueta en Git?

  Git tag -a v1.0.2 -m ‘Version 1.0.2’

- ¿Cómo enviar una etiqueta a GitHub?

  Con git push origin v1.0.2 o para enviarlas a todas con git push --tags

- ¿Qué es un historial de Git?

  Muestra todos los commits hechos en el proyecto

- ¿Cómo ver el historial de Git?

  Con git log o git log –oneline para ver resumen

- ¿Cómo buscar en el historial de Git?

  Con git log –grep=’palabra-clave’ para buscar por mensaje

  Y con git log –author=’nombre’

- ¿Cómo borrar el historial de Git?

  Podemos eliminar la carpeta .git, pero no se recomienda hacerlo en proyectos compartidos o con el comando rm -rf .git

- ¿Qué es un repositorio privado en GitHub?

  En un repositorio privado solo puede acceder el dueño y las personas invitadas

- ¿Cómo crear un repositorio privado en GitHub?

  Ir a GitHub y hacer clic en "+" → "New repository"

  Poner un nombre y elegir "Private"

  Hacer clic en "Create repository"

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

  Ir al repositorio de github

  Ir a settings y entrar a manage Access

  Hacer clic en "Invite a collaborator" y escribir su usuario

  Enviar la invitación

- ¿Qué es un repositorio público en GitHub?

  Un repositorio público es visible para cualquiera, pero solo los colaboradores pueden modificarlo

- ¿Cómo crear un repositorio público en GitHub?

  Ir a GitHub y hacer clic en "+" → "Nuevo Repositorio"

- ¿Cómo compartir un repositorio público en GitHub?

  Con la URL de 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

Link al repositorio de la actividad 2:

https://github.com/SebaGossos/actividad-2.git

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.

Link al repositorio de la actividad 3:
https://github.com/SebaGossos/conflict-exercise.git