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 en la web dirigida a compartir repositorios en la nube producidos por programadores usando la tecnología Git.

- ¿Cómo crear un repositorio en GitHub?

  La creación de un repositorio en github se realiza a través de la plataforma Github.com; A partir de que este creado nuestro usuario, se puede crear un nuevo repositorio usando el botón crear nuevo repositorio

- ¿Cómo crear una rama en Git?

  La creación de una rama en Git se realiza en la Bash a través del comando git Branch “nombre de la rama”.

- ¿Cómo cambiar a una rama en Git?

  El cambio de ramas en Git se realiza a través del comando Git checkout “nombde de la rama a la que nos dirigimos”

- ¿Cómo fusionar ramas en Git?

  Para fusionar ramas en Git se utiliza el comando merge. Esto puede traer conflictos entre piezas de código que correspondan al mismo archivo y que sean diferentes. Deberemos resolver los conflictos llegando a un código final que puede optar por uno

de los dos en conflicto o bien llegar a un resultado final que tome un poco de cada uno

- ¿Cómo crear un commit en Git?

  Para crear un commit en Git, se utiliza primero el comando git add . y luego el comando git commit -m “un mensaje que individualice el commit”

- ¿Cómo enviar un commit a GitHub?

  Para enviar un commit a un repositorio remoto en github se utiliza el comando remote para conectar con el repositorio remoto y luego el comando push para enviarle archivos a ese repositorio

- ¿Qué es un repositorio remoto?

  Un repositorio remoto es aquel que esta alojado en la nube

- ¿Cómo agregar un repositorio remoto a Git?

  Se realiza a través del comando git remote add “dirección url del repo”

- ¿Cómo empujar cambios a un repositorio remoto?

  A través del comando push

- ¿Cómo tirar de cambios de un repositorio remoto?

  A través del comando pull

- ¿Qué es un fork de repositorio?

  Un fork es una bifurcación del código. Se produce cuando realizamos una copia de un repositorio ajeno en nuestra cuenta

- ¿Cómo crear un fork de un repositorio?

  Se crea apretando el botón fork en la web de github

- ¿Cómo enviar una solicitud de extracción (pull request) a un repositorio?
  Primero se crea una rama o una bifurcación del código sobre el que se quiere realizar. Luego se solicita el pull request con el botón cresar pull request
- ¿Cómo aceptar una solicitud de extracción?

  Desde la pagina del repositorio donde se produjo la pull request, en la pestaña Pull Request se encuentra el listado de pull requests, allí uno puede ingresar a cada una, revisar el código y aceptarlo
- ¿Qué es un etiqueta en Git?

  Una etiqueta o tag es una marca que se realiza sobre proyecto en git y sirve para identificar momentos, versiones, hacer referencias
- ¿Cómo crear una etiqueta en Git?

  Se crea usando el comando git tag “nombre de la etiqueta”
- ¿Cómo enviar una etiqueta a GitHub?

  Se envia con el comando git push
- ¿Qué es un historial de Git?

  Es un listado de todos los commits realizados en un repositorio
- ¿Cómo ver el historial de Git?

  Con el comando git log
- ¿Cómo buscar en el historial de Git?

  Se busca con el comando git log. Se puede buscar por el mensaje del commit, por el autor, por fecha, etc
- ¿Cómo borrar el historial de Git?

  Se borra con el comando orphan
- ¿Qué es un repositorio privado en GitHub?

  Es aquel al que solo el autor tiene acceso
- ¿Cómo crear un repositorio privado en GitHub?

  Con el botón crear repositorio y en la configuración de repo elegir privado
- ¿Cómo invitar a alguien a un repositorio privado en GitHub?

  Dentro del repositorio, en la configuración, en el apartado de acceso, se oprime el botón de dar acceso
- ¿Qué es un repositorio público en GitHub?

  Es aquel que cualquiera puede ver y acceder
- ¿Cómo crear un repositorio público en GitHub?

En el panel de creación del repositorio, con el botón de visibilidad, se puede elegir la opción Publico

- ¿Cómo compartir un repositorio público en GitHub?

  Con la URL 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 "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.