**Práctico 2: Git y GitHub**

**Alumno: DAVID GUTIERREZ**

*Actividades*

* 1. Contestar las siguientes preguntas utilizando las guías y documentación proporcionada (Desarrollar las respuestas)

* + - ¿Qué es GitHub?

GitHub es una plataforma de desarrollo colaborativo que utiliza Git como sistema de control de versiones. Permite a los usuarios almacenar, gestionar y seguir el historial de proyectos de software.

* + - ¿Cómo crear un repositorio en GitHub?
    1. Inicia sesión en GitHub
    2. Haz clic en el botón "nuevo repositorio"
    3. Introduce un nombre (público o privado)
    4. Haz clic en "Crear repositorio"
    - ¿Cómo crear una rama en Git?

para crear una rama en Git se usa

git branch rama\_x

* + - ¿Cómo cambiar a una rama en Git?

Para cambiar a una rama git checkout rama\_x

* + - ¿Cómo fusionar ramas en Git?

Para fusionar una rama en la rama en la que estás actualmente, primero hay que estar en la rama de destino:

git checkout rama\_destino

Luego

git merge rama-a-fusionar

* + - ¿Cómo crear un commit en Git?

Para crear un commit, primero hay que subir al área de preparación (staging area). Utiliza:

git add . o git add archivoasubir.txt

Luego, crea el commit con

git commit -m "Mensaje del commit"

* + - ¿Cómo enviar un commit a GitHub?

Para enviar tus commits al repositorio remoto se utiliza el siguiente comando, siempre y cuando ya este cargado al staging area

git push origin nombre-de-la-rama

* + - ¿Qué es un repositorio remoto?

Un repositorio remoto es una versión de tu proyecto que se almacena en la nube, como GitHub

* + - ¿Cómo agregar un repositorio remoto a Git?

Para agregar un repositorio remoto se ejecuta el siguiente comando:

git remote add nombre\_remoto URL-del-repositorio

* + - ¿Cómo empujar cambios a un repositorio remoto?

Para empujar (o subir) tus cambios al repositorio remoto se utiliza

git push origin nombre-de-la-rama

Esto actualiza el repositorio remoto con tu trabajo local en la rama especificada

* + - ¿Cómo tirar de cambios de un repositorio remoto?

Para obtener los cambios realizados en un repositorio remoto, utiliza

git pull origin nombre-de-la-rama

Esto fusionará los cambios remotos en tu rama local actual

* + - ¿Qué es un fork de repositorio?

una copia personal de un proyecto existente en GitHub. Permite a los usuarios modificar el proyecto sin afectar el repositorio original

* + - ¿Cómo crear un fork de un repositorio?

Hacer clic en el botón "Fork" en la esquina superior derecha de la página del repositorio. Esto creará una copia del repositorio en mi cuenta de GitHub

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

Para enviar una solicitud de extracción:

1. Ve a tu repositorio forked.
2. Haz clic en "Pull requests".
3. Luego, selecciona "New pull request".
4. Elige la rama que deseas fusionar y llena la información requerida.
5. Haz clic en "Create pull request".
   * + ¿Cómo aceptar una solicitud de extracción?

Para aceptar una solicitud de extracción en un repositorio

1. Ve a la sección de "Pull requests" del repositorio.
2. Selecciona la solicitud que deseas revisar.
3. Revisa los cambios y si todo está correcto, haz clic en "Merge pull request".
   * + ¿Qué es un etiqueta en Git?

Una etiqueta en Git es un marcador que se utiliza para señalar un punto específico en el historial del proyecto, normalmente para identificar versiones o entregas importantes

* + - ¿Cómo crear una etiqueta en Git?

Para crear una etiqueta

git tag -a nombre-etiqueta -m "Mensaje descriptivo"

Esto crea una etiqueta anotada con un mensaje

* + - ¿Cómo enviar una etiqueta a GitHub?

Para enviar una etiqueta a GitHub se utiliza

git push origin nombre-etiqueta

Esto subirá la etiqueta creada a tu repositorio remoto

* + - ¿Qué es un historial de Git?

El historial de Git es una lista de todos los commits realizados en un repositorio, que incluye información como los mensajes de commit, la fecha, el autor y los cambios realizados.

* + - ¿Cómo ver el historial de Git?

Para ver el historial de commits en un repositorio, ejecuta:

git log

Esto mostrará una lista detallada de todos los commits en la rama actual

* + - ¿Cómo buscar en el historial de Git?

Para buscar en el historial se puede usar

git log --grep="texto-buscado"

Esto filtrará los commits que contienen el texto especificado en sus mensajes

* + - ¿Cómo borrar el historial de Git?

Borrar el historial de Git de un repositorio

* Hacer un nuevo commit vacío:

git checkout --orphan nueva-rama

git commit -m "Inicializando nuevo historial"

* Eliminar la rama antigua:

git branch -D main

* Renombrar la nueva rama:

git branch -m main

* Forzar el push al repositorio remoto:

git push -f origin main

* + - ¿Qué es un repositorio privado en GitHub?

Es un repositorio no es accesible al público.

* + - ¿Cómo crear un repositorio privado en GitHub?

1. Inicia sesión en tu cuenta de GitHub.
2. Haz clic en el botón "+" "New repository".
3. introduce un nombre
4. Selecciona la opción "Private".
5. Haz clic en "Create repository".
   * + ¿Cómo invitar a alguien a un repositorio privado en GitHub?
6. Ir a la página del repositorio privado.
7. Haz clic en "Settings" (configuración).
8. Seleccionar "Manage access"
9. Click en "Invite teams or people".
10. Ingresa el nombre de usuario o correo electrónico de la persona que deseas invitar y presiona "Invite".
    * + ¿Qué es un repositorio público en GitHub?

Puede ser visto y accedido por cualquier persona en Internet.

* + - ¿Cómo crear un repositorio público en GitHub?

Asegúrarse de seleccionar la opción "Public" al crear un repositorio o desde setings

* + - ¿Cómo compartir un repositorio público en GitHub?

Copia la URL del repositorio desde la barra de direcciones de tu navegador.

* 1. Realizar la siguiente actividad:

* + - Crear un repositorio.

* + - * + Dale un nombre al repositorio. o 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

o Crear una Branch o Realizar cambios o agregar un archivo o 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.