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 basada en la nube donde se puede almacenar código, en un repositorio remoto, que utiliza Git para el control de versiones. Permite almacenar, colaborar y gestionar proyectos trabajando en equipo o en proyectos de código abierto.

- ¿Cómo crear un repositorio en GitHub?

  Teniendo una cuenta en GitHub y una sesión iniciada, haciendo clic en el icono "+" en la parte superior derecha de la pantalla y seleccionando "New repository".

  Luego, ingresando un nombre para el repositorio. Se puede elegir que el repositorio sea público o privado.

  Y por último haciendo clic en "Create repository".

- ¿Cómo crear una rama en Git?

  Dentro de un proyecto, en una terminal, se ejecuta el comando:

  git branch ejemplo-de-nombre

  donde, después del comando branch, se escribe el nombre de la rama a crear, en este caso “ejemplo-de-nombre”

- ¿Cómo cambiar a una rama en Git?

  Ejecutando el comando:

  git checkout nueva-rama

- ¿Cómo fusionar ramas en Git?

  Primero debemos cambiar a la rama a la que queremos fusionar los cambios (Ejemplo)

  git checkout main

  Luego ejecutar el comando

  git merge nueva-rama.

  Todos lo cambios realizados en “nueva-rama” se integrarán a “main”.

- ¿Cómo crear un commit en Git?

  Se ejecuta el comando:

  git add .

  para agregar archivos y carpetas al escenario de git.

  Luego se ejecuta el comando:

  git commit -m “ejemplo de mensaje” (Dentro de las comillas puede escribirse un mensaje identificar que cambios se hicieron)

- ¿Cómo enviar un commit a GitHub?

  Se ejecuta el comando:

  git push origin main

  donde “origin” en el nombre, por defecto, del repositorio remoto y “main” es la rama que se está enviando al repositorio remoto.

- ¿Qué es un repositorio remoto?

  Es una versión de un proyecto almacenada en un servidor en línea, permitiendo que varias personas trabajen en el mismo código, sin estar conectadas a redes privadas.

- ¿Cómo agregar un repositorio remoto a Git?

  Se ejecuta el comando:

  git remote add origin URL-del-repositorio

- ¿Cómo empujar cambios a un repositorio remoto?

  Se ejecuta el comando:

  git push origin nueva-rama

- ¿Cómo tirar de cambios de un repositorio remoto?

  Se ejecuta el comando:

  git push origin nueva-rama

- ¿Qué es un fork de repositorio?

  Es una copia de un repositorio que se encuentra en otra cuenta de GitHub. Se utiliza para hacer modificaciones en un proyecto sin afectar el original. Los forks permiten contribuir a proyectos abiertos y enviar cambios a través de pull requests.

- ¿Cómo crear un fork de un repositorio?

  En GitHub, yendo al repositorio que se quiere realizar el fork, haciendo click en Fork (en la parte superior derecha de la pantalla)

- ¿Cómo enviar una solicitud de extracción (pull request) a un repositorio?
  Una vez realizados los cambios en el fork propio, haciendo click en Pullrequest (en la parte superior de la pantalla) y luego haciendo click en “create a pull request”.

- ¿Cómo aceptar una solicitud de extracción?

  En un repositorio en GitHub, abrir la pestaña Pull requests, seleccionar la solicitud pendiente. Revisar los cambios y hacer click en Merge pull request.

  Confirma la fusión haciendo clic en Confirm merge.

- ¿Qué es un etiqueta en Git?

  Una etiqueta (tag) es una referencia a un punto específico del historial del proyecto, usada generalmente para marcar versiones estables o lanzamientos (ej. v1 o v1.0).

- ¿Cómo crear una etiqueta en Git?

  Se ejecuta el comando:

  git tag v1.0

- ¿Cómo enviar una etiqueta a GitHub?

  Se ejecuta el comando:

  git push origin v1.0

  o si se quiere enviar todas las etiquetas de una sola vez:
  git push origin –tags

- ¿Qué es un historial de Git?

  El historial de Git es un registro de todos los commits realizados en el repositorio, permitiendo ver qué cambios se hicieron y quién los hizo.

- ¿Cómo ver el historial de Git?

  Se ejecuta el comando:

  git log

- ¿Cómo buscar en el historial de Git?

  Se puede buscar por palabras clave en los commits:

  git log --grep="palabra clave"

  O se puede buscar en los cambios de archivos:

  git log -S "texto a buscar"

  - ¿Cómo borrar el historial de Git?

    Se ejecutan los comandos:

    rm -rf .git

    git init

    git add .

    git commit -m "Nuevo inicio del repositorio"

    Esto borra todo el historial sin afectar los archivos actuales

- ¿Qué es un repositorio privado en GitHub?

  Es un repositorio que solo puede ver el usuario que lo creo, y por los usuarios a los que este pudo darle permiso.

- ¿Cómo crear un repositorio privado en GitHub?

  Haciendo click en el icono "+" en la parte superior derecha de la pantalla y seleccionando "New repository".

  Luego, ingresando un nombre para el repositorio.

  Elegir la opción “Private”.

  Y por último haciendo clic en "Create repository".

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

  Abrir el repositorio.

  Settings \ Collaborators

  Add people (ingresar usuario)

  Por último click en “Invite”

- ¿Qué es un repositorio público en GitHub?

  Es un repositorio que puede ser visto por cualquier usuario de GitHub. Generalmente, se usa para proyectos abiertos.

- ¿Cómo crear un repositorio público en GitHub?

  Haciendo click en el icono "+" en la parte superior derecha de la pantalla y seleccionando "New repository".

  Luego, ingresando un nombre para el repositorio.

  Elegir la opción “Public”.

  Y por último haciendo clic en "Create repository".

- ¿Cómo compartir un repositorio público en GitHub?

  Abrir un repositorio.

  Hacer click en el botón verde “< > Code”

  Hacer click en el botón de copiar URL

2) Realizar la siguiente actividad:

- Crear un repositorio.
    - Dale un nombre al repositorio.
    - Elije el repositorio sea público.
    - Inicializa el repositorio con un archivo.

    Logueado con mi usuario de GitHub, voy a la parte superior de la pantalla al icono “+” “New repository”, en “Repository name” escribo “Unidad_2- - Ejercicio_2” y en “Description” escribo “Repositorio de Programación I - Ejercicio 2 - Unidad 2 - TUPaD-UTN”
    Dejo seleccionada la opción “Public”
    En “Initialize this repository with:” tildo la opción “Add a README file”
    Por último hago click en "Create repository".

- 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.

    Copio la URL del repositorio con el botón “Code”
    Abro una terminal en mi computadora y ejecuto el comando:
    git clone https://github.com/MolinariP/MolinariP-Unidad_2-Ejercicio_2.git
    Creo un archivo txt con el nombre “Primer-archivo” con el texto “Este es el primer archivo”
    Desde la consola, ejecuto los comandos:
    git add .
    git commit -m “Agrego Primer-archivo.txt”

- Sube los cambios al repositorio en GitHub con git push origin main (o el nombre de la rama correspondiente).

    En la consola ejecuto el comando:
    git push origin main

- Creando Branchs

  - Crear una Branch
  - Realizar cambios o agregar un archivo
  - Subir la Branch

En la consola ejecuto los comandos:
git branch cambio

git checkout cambio

Al archivo txt le agrego el texto " del ejercicio 2 de la unidad 2"

En la consola ejecuto los comandos:
git add .

git commit -m "Agrego texto al archive Primer-archivo.txt"

Subo la nueva rama con el comando:

git push origin cambio

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).

  https://github.com/MolinariP/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.