Alumno: Vapore, Bruno Giuliano
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 diseñada como una comunidad, similar a una red social, donde se puede crear y compartir los repositorios de código fuente en forma pública o privada. Permite el trabajo colaborativo entre los programadores, editando código y permitiendo hacer un seguimiento de los cambios. A su vez, puede asociarse con el repositorio local de Git.

- ¿Cómo crear un repositorio en GitHub?

  Desde el Dashboard en GitHub se puede crear un nuevo repositorio con “New”, se le asigna un nombre, una descripción, se determina si es público o privado, y tiene la opción de agregar un archivo README.

- ¿Cómo crear una rama en Git?

  Desde la consola se escribe el comando git branch “nombre de la rama”

- ¿Cómo cambiar a una rama en Git?

  Desde la consola se escribe el comando git checkout “nombre de la rama”

- ¿Cómo fusionar ramas en Git?

Desde la consola me posiciono en la rama que uso de base para hacer la fusión, donde quedarán guardados los cambios, y con el comando git merge “nombre de la rama” la fusiono, vuelva la rama nombrada a donde estoy posicionado.

- ¿Cómo crear un commit en Git?

  Desde la consola se escribe el comando git commit –m “mensaje descriptivo”

- ¿Cómo enviar un commit a GitHub?

  Desde la consola se escribe git push, o si es la primera vez hay que usar git push –u origin master/main

- ¿Qué es un repositorio remoto?

  Es un repositorio propio, de otro usuario o empresa que se comparte por redes privadas o se sube a internet. Por lo general se sube a la plataforma más utilizada (GitHub) aunque existen otras plataformas. Esto permite trabajar colaborativamente en diferentes proyectos o continuar el propio proyecto desde otro lugar.

- ¿Cómo agregar un repositorio remoto a Git?

  Desde la consola se escribe el comando git clone “URL del repositorio”

- ¿Cómo empujar cambios a un repositorio remoto?

  Desde la consola se escribe el comando git push, una vez comiteados los cambios con git add . y git commit

- ¿Cómo tirar de cambios de un repositorio remoto?

  Desde la consola se escribe el comando git pull

- ¿Qué es un fork de repositorio?

  Es una copia (o bifurcación) del repositorio original de otro usuario en GitHub, para agregarlo como repositorio propio en la cuenta personal. Permite trabajar con el código sin afectar el repositorio original, salvo que realice un pull request donde el otro usuario podrá aceptar o rechazar tales cambios sugeridos.

- ¿Cómo crear un fork de un repositorio?

  Desde el repositorio en GitHub, voy al recuadro de fork (a la derecha del nombre del repositorio) y hago click en “Create a new fork”, donde podré darle un nombre y una descripción.

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

  Desde el repositorio en GitHub, voy a la pestaña de pull request y hago click en New pull request. Desde ahí elijo el base repository y head repository para fusionar antes de hacer click en Create pull request.

- ¿Cómo aceptar una solicitud de extracción?

  Desde el repositorio en GitHub, voy a la pestaña de pull request y hago click en el request que quiero analizar y fusionar, luego click en Merge pull request y luego Confirm merge

- ¿Qué es una etiqueta en Git?

  Es una referencia que sirve para administrar el repositorio, permitiendo marcar puntos importantes en el historial del repositorio, por ejemplo marcar versiones de lanzamiento.

- ¿Cómo crear una etiqueta en Git?

  Desde la consola se escribe el comando git tag “nombre de la etiqueta”

- ¿Cómo enviar una etiqueta a GitHub?

  Desde la consola se escribe el comando git push origin “nombre de la etiqueta”

- ¿Qué es un historial de Git?

  Es un listado de los cambios que se han hecho en el repositorio a través del commit, en orden cronológico, indica el autor y la fecha de cada commit.

- ¿Cómo ver el historial de Git?

  Desde la consola se escribe el comando git log para ver los commits. También se puede usar el comando git reflog para ver el historial del repositorio.

- ¿Cómo buscar en el historial de Git?

  Desde la consola, una vez ingresado el comando git log, puedo buscar el identificador alfanumérico (hash) del commit. Se podría agregar el subcomando - - since y - - until para buscar por fechas determinadas; - - author= para buscar por autor, entre otras posibilidades. También podría buscar el contenido de un commit, una vez ingresado git reflog, con el comando git show y el hash.

- ¿Cómo borrar el historial de Git?

  No se podría borrar, en principio, el historial del repositorio (que se visualiza con git reflog), pero si regresar en el tiempo. Desde la consola se escribe el comando git reset y los subcomandos - -soft, - - mixed y - - hard, según si quiero retroceder el puntero HEAD al commit especificado sin modificar el Stage (modo suave); o si quiero incluir también el Stage (modo mixto); o si quiero modificar de forma permanente los archivos y carpetas del proyecto (modo duro). Aunque siempre podré ver el historial completo con el comando git reflog.

- ¿Qué es un repositorio privado en GitHub?

  Es un repositorio remoto al cual solo tienen acceso las personas autorizadas.

- ¿Cómo crear un repositorio privado en GitHub?

  Desde el Dashboard en GitHub se crea un nuevo repositorio con “New” y se selecciona la opción “Private”.

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

  Desde la pestaña “Settings” del repositorio, selecciono “Collaborators” y luego se hace click en “Add people” para ingresar el usuario de GitHub que va a tener acceso al repositorio. Finalmente, una vez seleccionado el nivel de acceso, se hace click en “Add”.

- ¿Qué es un repositorio público en GitHub?

  Es un repositorio remoto en el que toda la comunidad de GitHub y otras personas en Internet pueden tener acceso al código.

- ¿Cómo crear un repositorio público en GitHub?

  Desde el Dashboard en GitHub se crea un nuevo repositorio con “New” y se selecciona la opción “Public”.

- ¿Cómo compartir un repositorio público en GitHub?

  Desde el repositorio que quiero compartir hago click en “<> Code” y desde la pestaña local copio la url del repositorio para enviársela a quien quiera por cualquier medio.

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

URL del repositorio: https://github.com/BrunoGVapore/TP2-Actividad2.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.

URL del repositorio:https://github.com/BrunoGVapore/conflict-exercise.git