Práctico 2: Git y GitHub

Estudiante: Spital, Emilce

1- Git es un sistema de control de versiones distribuido, de código abierto, creado por Linus Torvalds en 2005. Es un servicio que permite a los desarrolladores almacenar y gestionar el código de sus proyectos, realizar un seguimiento de los cambios, colaborar con otros desarrolladores y facilitar la revisión de código.

2- Para crear un repositorio en GitHub debemos seguir los siguientes pasos:

- Crea una cuenta en GitHub
- Inicia sesión en tu cuenta de GitHub.
- Haz clic en el botón "+" (más) en la esquina superior derecha y selecciona "New repository".
- Completa los detalles del nuevo repositorio:
- Repository name: Elige un nombre descriptivo para tu proyecto.
- Description (opcional): Agrega una breve descripción de tu repositorio.
- Public o Private: Selecciona si quieres que tu repositorio sea público (visible para todos) o privado (solo accesible para colaboradores que invites).
- Haz clic en el botón "Create repository".
- Una vez creado, GitHub te proporcionará una URL para acceder a tu repositorio

3- Para crear una rama debemos estar en repositorio local (en la linea de comandos) y escribir **git branch <nombre_de_la_rama>.** Reemplazamos <nombre_de_la_rama> con el nombre que queremos darle a la nueva rama (por ejemplo, proyecto nuevo). Este comando crea una nueva rama, pero no te cambia a ella.

4- Para movernos a una rama existente, utilizamos el comando **git checkout <**nombre de la rama**>.** Tambien puedemos crear y cambiar a una nueva rama al mismo tiempo con el comando **git checkout -b**

5- Para combinar los cambios de una rama en otra debemos:
Cambiar a la rama en la que queremos integrar los cambios: por ejemplo **git checkout main,** ejecutar el comando git merge seguido del nombre de la rama que quieres fusionar: **git merge “nuevo-proyecto”.**

6- Para crear un commit:
git add <nombre_del_archivo>
Para un archivo específico **git add .** Para agregar todos los cambios Ejecutamos el comando git commit con un mensaje descriptivo:
git commit -m "Descripción breve y clara de los cambios realizados"

7- Para enviar los commits locales al repositorio remoto en GitHub, utilizamos el comando git push: **git push origin <nombre_de_la_rama>.**

8- Un repositorio remoto es una versión de tu repositorio que está alojada en un servidor en línea (como GitHub).

GitHub es una plataforma en línea que permite almacenar proyectos que usan el sistema de control de versiones Git. Es el servicio más popular del mundo para compartir y colaborar en proyectos de software, tanto de código abierto como privados. Almacena los repositorios en la nube, lo que facilita el trabajo colaborativo entre desarrolladores ubicados en diferentes partes del mundo.

9- Debemos ir a la página del repositorio en GitHub y copiar la URL del repositorio (normalmente termina en .git).
En el repositorio local, utilizamos el comando **git remote add.**

10- Para enviar tus commits locales a una rama específica del repositorio remoto: **git push origin <nombre_de_la_rama>**

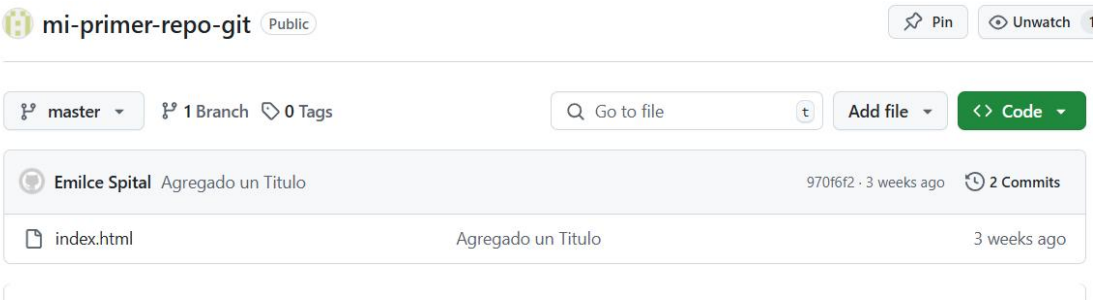
11- Para descargar los cambios del repositorio remoto y fusionarlos con la rama local actual, utilizamos el comando git pull:
Nos aseguramos de estar en la rama local donde queremos recibir los cambios: **git checkout main**
Ejecutamos el comando git pull
git pull origin <nombre_de_la_rama_remota>

12- Un fork de repositorio es como crear una copia personal de un repositorio que pertenece a otra persona o a una organización. La razón más común para hacer un fork es poder realizar cambios en el código de un proyecto al que no tienes permisos de escritura directos. Puedes modificar tu copia (tu fork) y luego proponer tus cambios al propietario del repositorio original a través de un mecanismo llamado "Pull Request" (solicitud de extracción).
13- · ¿Cómo crear un fork de un repositorio?
Abrir la pagina https://github.com/usuario/repositorio).
Buscar el botón "Fork" y hacer clic
Al hacer clic en este botón, GitHub comenzará el proceso de creación de una copia del repositorio en tu propia cuenta de GitHub.
Selecciona tu cuenta personal.
Espera a que se cree el fork: GitHub tardará unos segundos en copiar todo el repositorio (el código, el historial de commits, las ramas, etc.) a tu cuenta.

14- · Para clonar el fork a la computadora usamos:
Bash
git clone https://github.com/tu_usuario/repositorio.gitcd repositorio

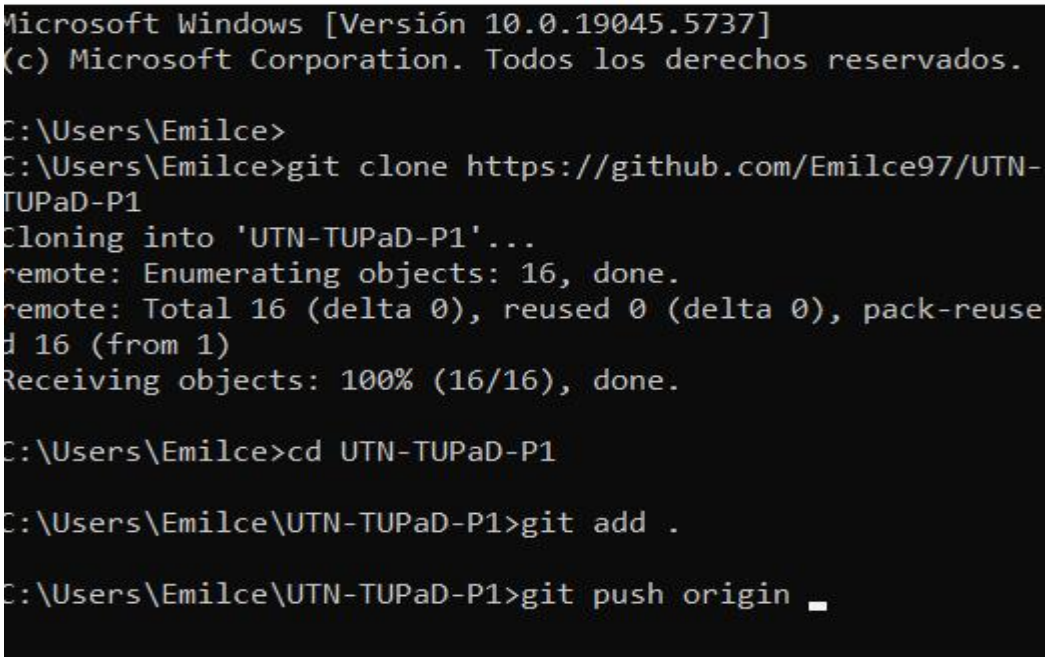
15- En GitHub, desde tu fork, ir a la pestaña "Pull requests" y crear una nueva, comparando tu rama con la del repositorio original.
16- En GitHub, en la pestaña "Pull requests" del repositorio original, revisar la solicitud y hacer clic en "Merge pull request".
17- Un punto de referencia estático en la historia del repositorio, usado comúnmente para versiones.
18- Git tag <nombre_etiqueta> (ligera) o git tag -a <nombre_etiqueta> -m "<mensaje>" (anotada).

19- Git push origin <nombre_etiqueta> o git push origin --tags (para todas).
20- Un registro cronológico de todos los cambios (commits) en un repositorio.
21- Git log (detallado), git log --oneline (compacto), git log --graph --oneline --decorate --all (gráfico).
22- Git log --author="<autor>", git log --grep="<palabra>", git log --since="<fecha>", git log <archivo>.
23- Generalmente se evita en repositorios compartidos. Para local: git reset --hard HEAD~<n> (con precaución). Para remoto, requiere git push --force (¡riesgoso!). Se prefiere git revert.
24- Un repositorio cuyo contenido solo es visible para el propietario y los colaboradores invitados.
25- Al crear un nuevo repositorio, seleccionar la opción "Private".
26- En la pestaña "Settings" del repositorio, ir a "Collaborators" y añadir personas por su nombre de usuario o correo electrónico.
27- Un repositorio cuyo contenido es visible para cualquier persona en internet.
28- Al crear un nuevo repositorio, seleccionar la opción "Public".

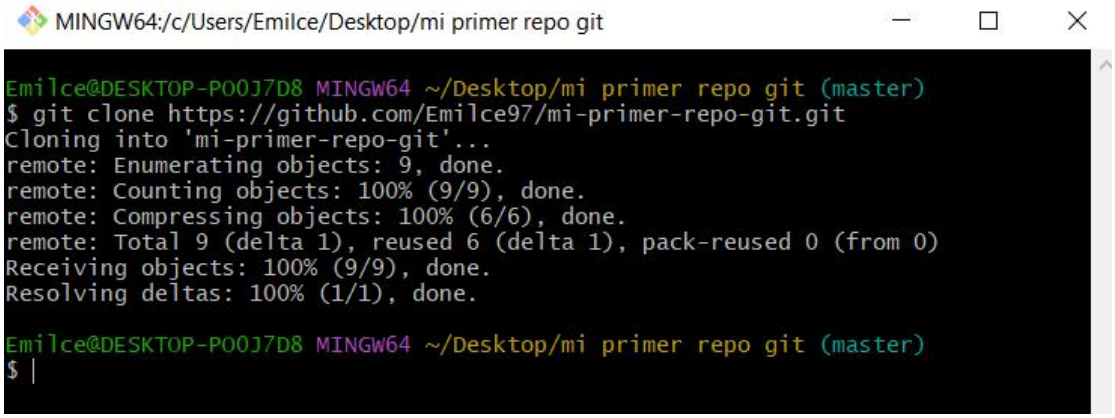


29- Copiando y compartiendo la URL del repositorio.

Actividad 3. Crear un repositorio

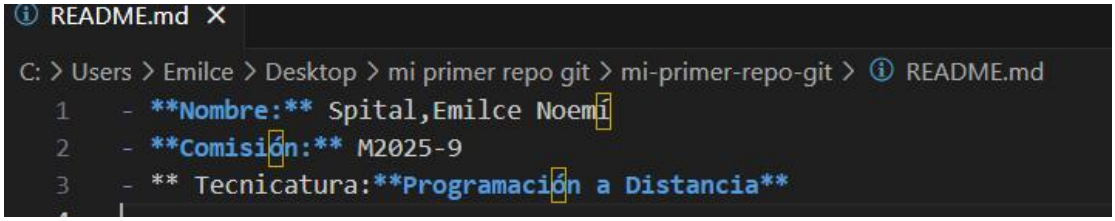


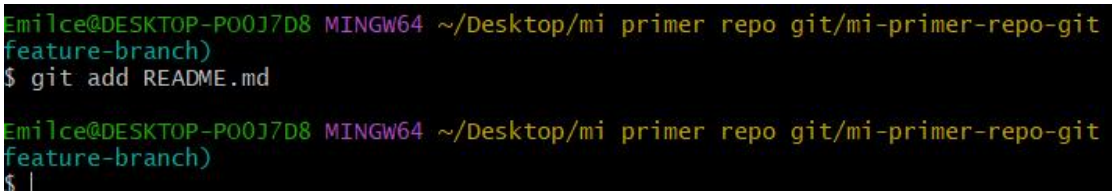
Clonar una URL



Crear una nueva rama:

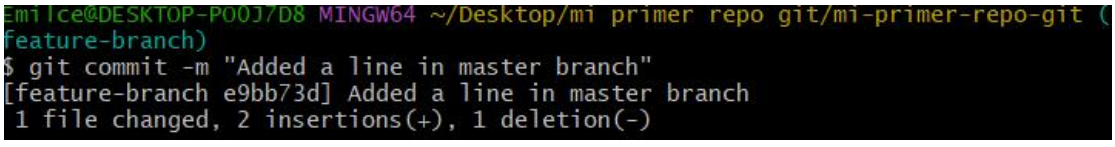
Guardar los cambios



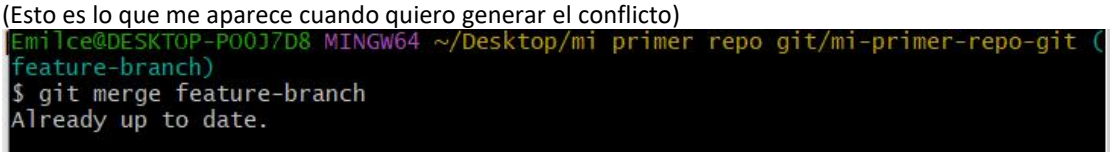


Volver a la rama master

Conflicto



(Esto es lo que me aparece cuando quiero generar el conflicto)



Cambios guardados en el repositorio .