Práctico 2: Git y GitHub
Alumno: Angel Fernando TORRES

- **¿Qué es GitHub?**

  Es una plataforma donde se suben proyectos elaborados de manera colaborativa utilizando el software de control de versiones Git.

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

  Haciendo login con nuestro usuario en https://github.com/new y siguiendo las instrucciones. Una vez hecho, clickear en el botón verde “Crear repositorio”.

- **¿Cómo crear una rama en Git?**

  Con el comando *git branch nombre-de-la-nueva-rama.*

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

  Con el comando *git checkout nombre-de-la-nueva-rama.*

- **¿Cómo fusionar ramas en Git?**

  Con el comando *git merge nombre-de-la-nueva-rama* estando previamente en la rama principal a la que se va a fusionar.

- **¿Cómo crear un commit en Git?**

  Con el comando *git commit* (es recomendable escribir *-m* para agregar un comentario).

- **¿Cómo enviar un commit a GitHub?**

  Con el comando *git push origin main.*

- **¿Qué es un repositorio remoto?**

  Es una copia de un repositorio local, subido a internet.

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

  Teniendo inicializado un repositorio local en nuestra PC, con el comando *git remote add origin [URL del repositorio remoto].*

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

  Con el comando *git push origin main.*

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

  Con el comando *git pull origin nombre-de-la-rama.*

- **¿Qué es un fork de repositorio?**

  Es una copia de un repositorio de otra persona a partir de la cual podemos trabajar y modificar a nuestro gusto sin afectar al repositorio original.

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

  Ingresando al link *https://github.com/[usuario]/[repositorio]/fork* y seguir las instrucciones. Una vez realizado, clickear en el botón verde *“Fork”.*

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

Una vez hecho un *fork* de un repositorio y habiendo subido nuestros cambios mediante *push,* ingresar a *https://github.com/[nuestro-usuario/nuestro-fork/pulls* y clickear en *"New pull request".* Luego, clickear el botón verde "*Create pull request*".

- **¿Cómo aceptar una solicitud de extracción?**

Ingresando a *https://github.com/[nuestro-usuario/nuestro-repositorio/pulls* y después en cada solicitud, clickear el botón verde "*Merge pull request*".

- **¿Qué es un etiqueta en Git?**

Es un señalador en el historial de commits. Sirven para filtrar y organizar mejor un repositorio.

- **¿Cómo crear una etiqueta en Git?**

Con el comando *git tag nombre-de-la-etiqueta*

- **¿Cómo enviar una etiqueta a GitHub?**

Con el comando *git push origin nombre-de-la-etiqueta*

- **¿Qué es un historial de Git?**

Es el registro de todos los cambios que se realizaron en un repositorio. Contiene datos de las personas que lo modificaron, en qué fecha y hora lo hicieron, y los mensajes agregados en cada commit.

- **¿Cómo ver el historial de Git?**

Con el comando *git log.*

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

Con los comandos *git log --grep="Palabra clave"* ó buscando el autor: *git log --author="Nombre del autor".*

- **¿Cómo borrar el historial de Git?**

Existe varias maneras de realizarlo. Entre ellas, el comando *git rebase -i HEAD~n* donde *n* es la cantidad de commits que se desean modificar/eliminar. Luego se coloca *drop* en los commits a eliminar.

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

Es un repositorio en el cual sólo pueden acceder personas autorizadas.

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

Es igual a crear cualquier repositorio, pero se debe colocar la opción de *radioButton* (con el candado de imagen) "*Private*".

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

Clickeando "*Add people*" en el siguiente link: https://github.com/[nuestro usuario]/[nuestro repositorio]/settings/access

- **¿Qué es un repositorio público en GitHub?**

Es un repositorio en el cual cualquier persona con acceso a internet puede ingresar a verlo.

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

Contestado en la segunda pregunta. Se debe colocar "*Public*" al momento de hacerlo.

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

Mientras la visibilidad del repositorio sea pública, con compartir el link es suficiente: https://github.com/[nuestro usuario]/[nuestro repositorio]

2) **Realizar la siguiente actividad:**

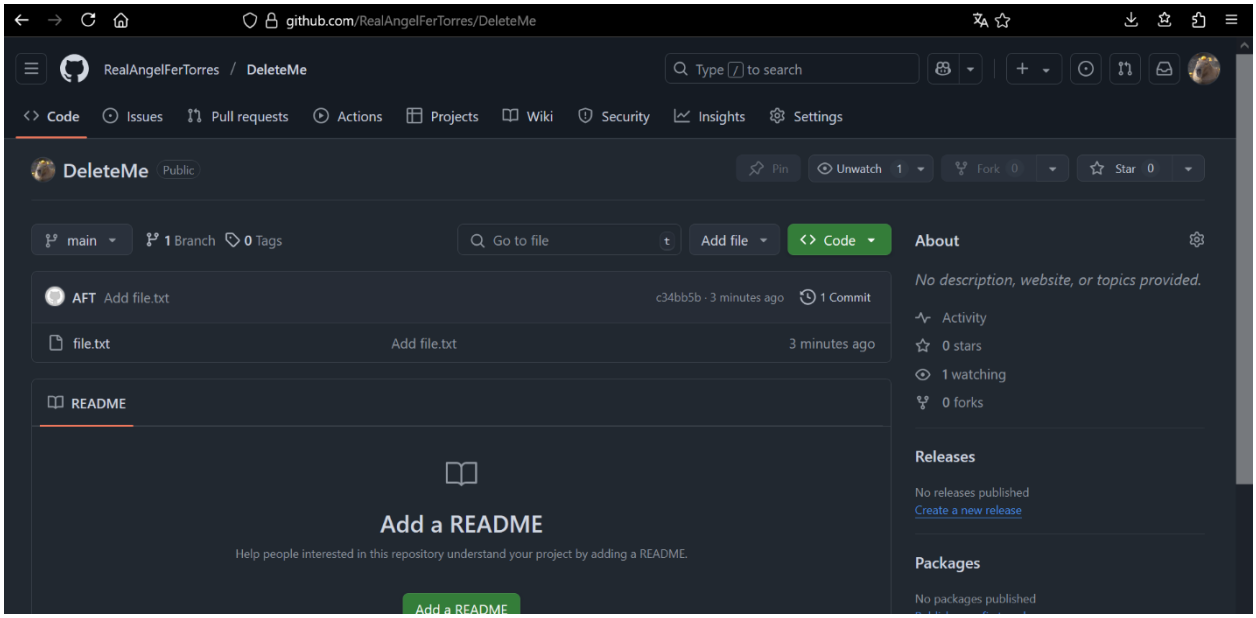
- **Crear un repositorio.**

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

  https://github.com/RealAngelFerTorres/DeleteMe

- **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).**



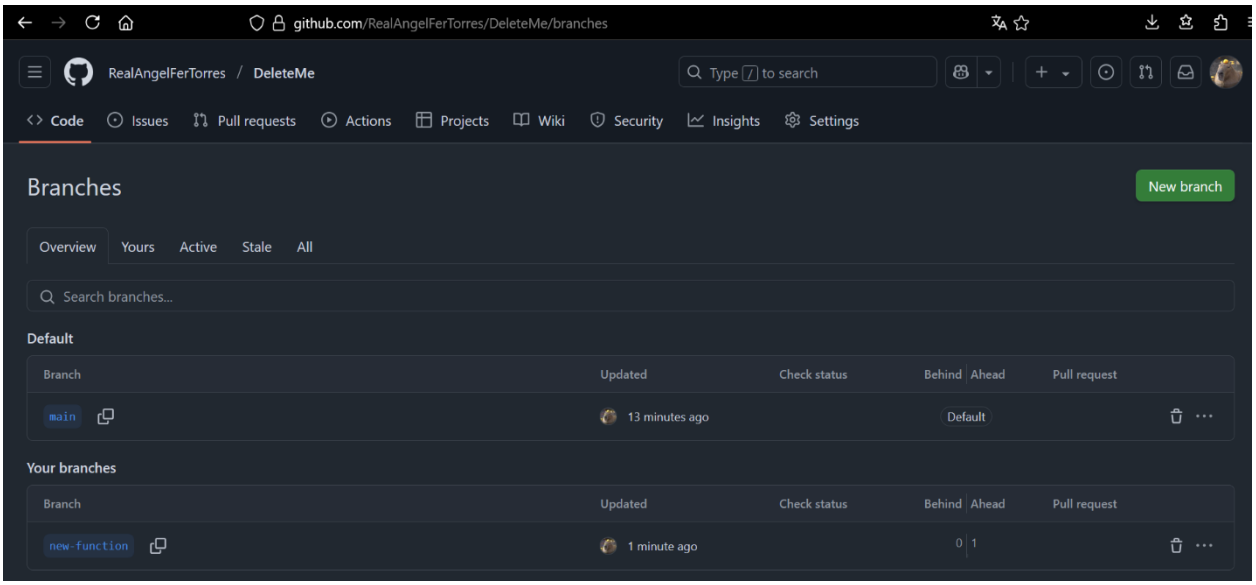

```
PS H:\Varios\Estudio\UTN - TUP\Programacion I\Ex2> git init
Initialized empty Git repository in H:/Varios/Estudio/UTN - TUP/Programacion I/Ex2/.git/
PS H:\Varios\Estudio\UTN - TUP\Programacion I\Ex2> git add .
PS H:\Varios\Estudio\UTN - TUP\Programacion I\Ex2> git commit -m "Add file.txt"
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 file.txt
On branch master
nothing to commit, working tree clean
```

```
PS H:\Varios\Estudio\UTN - TUP\Programacion I\Ex2> git push --force
Enumerating objects: 3, done.
Counting objects: 100% (3/3), done.
Writing objects: 100% (3/3), 216 bytes | 216.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
To https://github.com/RealAngelFerTorres/DeleteMe.git
 + 87e34d5...c34bb5b main -> main (forced update)
```

- **Creando Branchs**

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

https://github.com/RealAngelFerTorres/DeleteMe/tree/new-function



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.**

https://github.com/RealAngelFerTorres/conflict-exercise#