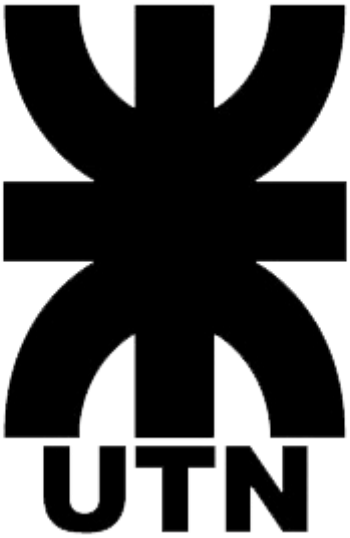
UNIVERSIDAD TECNOLOGICA NACIONAL UTN

TECNICATURA UNIVERSITARIA EN PROGRAMACION A DISTANCIA



PROGRAMACION I

Trabajo Práctico N.º 2: Git y GitHub

Estudiante: Martinez Juan.

Docente a cargo: Bruselario, Sebastián.

Tutor a cargo: Carbonari, Verónica.

Fecha de entrega: 30/03/2025

Actividades

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

• ¿Qué es GitHub?

• ¿Cómo crear un repositorio en GitHub?

• ¿Cómo crear una rama en Git?

• ¿Cómo cambiar a una rama en Git?

• ¿Cómo fusionar ramas en Git?

• ¿Cómo crear un commit en Git?

• ¿Cómo enviar un commit a GitHub?

• ¿Qué es un repositorio remoto?

• ¿Cómo agregar un repositorio remoto a Git?

• ¿Cómo empujar cambios a un repositorio remoto?

• ¿Cómo tirar de cambios de un repositorio remoto?

• ¿Qué es un fork de repositorio?

• ¿Cómo crear un fork de un repositorio?

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

• ¿Cómo aceptar una solicitud de extracción?

• ¿Qué es un etiqueta en Git?

• ¿Cómo crear una etiqueta en Git?

• ¿Cómo enviar una etiqueta a GitHub?

• ¿Qué es un historial de Git?

• ¿Cómo ver el historial de Git?

• ¿Cómo buscar en el historial de Git?

• ¿Cómo borrar el historial de Git?

• ¿Qué es un repositorio privado en GitHub?

• ¿Cómo crear un repositorio privado en GitHub?

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

• ¿Qué es un repositorio público en GitHub?

• ¿Cómo crear un repositorio público en GitHub?

• ¿Cómo compartir un repositorio público en GitHub?

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

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.

RESPUESTA:

1) **¿Qué es Github?**

GitHub es una plataforma de desarrollo colaborativo para alojar proyectos utilizando el sistema de control de versiones Git. Está basada en la nube donde puedes almacenar, compartir y trabajar junto con otros usuarios para escribir código.

2) **¿Cómo crear una rama en Git?**

Utilizando el comando: git branch nombre-rama, en donde “nombre-rama” será el nombre que nosotros le daremos a esa nueva rama.

Por ejemplo.

Git branch repositorio1

3) **¿Cómo cambiar a una rama en Git?**

Para cambiar a otra rama existente, haremos el comando:

Git checkout rama, en donde “rama” es el nombre de la rama que nosotros queremos llegar a utilizar.

Por ejemplo, si yo estoy trabajando en la rama “main” y quiero cambiar la rama “secundario”, entonces el comando utilizado seria:

Git checkout secundario

4) **¿Cómo fusionar ramas en Git?**

Para fusionar ramas existe el comando:

Git merge rama, lo que hace es unir dos ramas, en este caso la rama que nosotros especificaremos en “rama”, junto con la rama en donde actualmente estemos trabajando.

5) **¿Cómo crear un commit en Git?**

Para crear un commit y hacer que nuestros cambios queden guardados en el repositorio debemos utilizar el comando:

Git commit -m “mensaje”, en donde dentro de las comillas iría el mensaje que nosotros queramos. Vale aclarar que antes de hacer el Git commit, primero debemos hacer un git add . (esto lo que hará es añadir cuales son los cambios que se van a incluir).

6) **¿Cómo enviar un commit a GitHub?**

Utilizando el comando Git push, esto hará que se envíen los cambios al repositorio remoto-

7) **¿Qué es un repositorio remoto?**

Los repositorios remotos son versiones de tu proyecto que están alojadas en Internet o en cualquier otra red. Puedes tener varios de ellos, y en cada uno tendrás generalmente permisos de solo lectura o de lectura y escritura. Colaborar con otras personas implica gestionar estos repositorios remotos enviando y trayendo datos de ellos cada vez que necesites compartir tu trabajo.

8) **¿Cómo agregar un repositorio remoto a Git?**

Para agregar un repositorio remoto nuevo a git, usaremos el comando git remote add en el terminal, dentro del directorio donde está almacenado su repositorio.

9) **Cómo empujar cambios a un repositorio remoto?**

Utilizaremos el comando Git push, seguido de esto especificaremos el nombre del remoto y el nombre de la rama.

Por ejemplo:

Git push origin main

10) **¿Cómo tirar de cambios de un repositorio remoto?**

Utilizando el comando Git pull, se comporta de la misma manera que el anterior pero viceversa, es decir, para traer cambios del repositorio remoto a nuestro local.

Por ejemplo:

Git pull origin main

11) ¿Qué es un fork de repositorio?

Un fork es una copia de un repositorio creada en una cuenta diferente, permitiendo desarrollar cambios sin afectare el original. A diferencia de clonar, que descarga el repositorio localmente, el fork se realiza generando una copia en la cuenta del usuario.

12) ¿Cómo crear un fork de un repositorio?

Primero necesitamos tener una cuenta de GitHub, una vez hecho esto realizaremos los siguientes pasos:

- Iniciamos sesión en GitHub.
- Luego ingresaremos a un repositorio que queramos copiar.
- Una vez hecho esto, le daremos click en el botón de fork en la parte superior derecha, esto lo que hará es llevarnos a una pantalla de personalización de este nuevo fork, en donde podremos cambiar el nombre, la descripción, etc.
- Por último, terminaremos con la personalización haciendo click en “crear fork” y listo, ya tendríamos nuestra copia del repositorio en nuestra cuenta de GitHub.

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

Hacer un pull request requiere hacer varios pasos a seguir:

1. Debemos hacer un fork a un repositorio que queramos modificar.
2. Luego clonaremos ese repositorio a nuestra computadora local.
3. Crearemos una nueva Branch en donde trabajaremos separado del main.
4. Realizaremos cambios al repositorio y luego lo guardaremos.
5. Subimos los cambios a nuestro repositorio de GitHub.
6. Por ultimo, dentro de nuestro repositorio haremos click en “Compare and Pull Request”, creando una solicitud.

14) ¿Cómo aceptar una solicitud de extracción?

Para esto, debemos ir a nuestros repositorios de GitHub e ir a la pestaña de Pull Request, luego revisaremos la o las solicitudes en donde veremos los cambios propuestos, una vez esto hecho seleccionaremos la opción “Approve” para aprobar los cambios.

15)¿Qué es una etiqueta en Git?

Se consideran referencias que apuntan a puntos concretos dentro del historial de git, utilizados generalmente para marcas puntos importantes durante el desarrollo.

16)¿Cómo crear una etiqueta en Git?

Utilizando el comando Git tag nombre, en donde “nombre” será el cómo lo llamaremos.

17)¿Cómo enviar una etiqueta a GitHub?

Similar a empujar cambios a un repositorio remoto, utilizaremos el comando: git push origin “etiqueta”, en donde dentro de las comillas ira el nombre de la etiqueta que queramos enviar.

18)¿Qué es un historial de Git?

Git representa el historial de una manera fundamentalmente diferente de los sistemas de controles de versiones centralizados, Git almacena el historial como un gráfico de instantáneas de todo el repositorio. Estas instantáneas, llamadas confirmaciones en Git, pueden tener varios elementos primarios, creando un historial que parece un gráfico en lugar de una línea recta.

19) ¿Cómo ver el historial de Git?

Utilizando el comando Git log, este mostrara una línea de todas las confirmaciones en la rama o Branch en el que se encuentre.

20) ¿Cómo buscar en el historial de Git?

Si queremos buscar algo en específico en el historial de Git, podemos usar el comando Git log --grep= ”texto”, en donde lo que está entre comillas iría la palabra en especifico que queremos encontrar en el historial.

21) ¿Cómo borrar el historial de Git?

El comando git reset deshace cambios en un repositorio de Git. Se utiliza para revertir una rama a un estado anterior y es útil para limpiar o eliminar cambios que no se han enviado a un repositorio público.

22) ¿Qué es un repositorio privado en GitHub?

Es un lugar donde puedes almacenar el código, los archivos y el historial de revisiones de cada archivo. Los repositorios pueden contar con múltiples colaboradores. Al ser privado, solo vos, las personas con las que compartís el acceso explícitamente y, para los repositorios de organizaciones o algunos miembros de la organización pueden acceder a los repositorios privados.

23) ¿Cómo crear un repositorio privado en GitHub?

Una vez tengamos nuestro repositorio local y tengamos una cuenta de GitHub, vamos a ir a nuestro perfil de nuestra cuenta y en la parte superior derecha daremos click en “crear nuevo repositorio”. A continuación personalizaremos nuestro repositorio con un nombre, descripción, etc. Y luego configuraremos la accesibilidad, este puede ser privado o público. Elegimos la opción de “privado” y ya tendremos nuestro repositorio privado. Por último subimos el contenido de nuestro repositorio local a GitHub.

24) ¿Cómo invitar a alguien a un repositorio privado en GitHub?

Para esto necesitamos el o los nombres de usuario de las personas que queremos compartirles nuestro repositorio privado. Luego de esto, iremos a donde hemos creado nuestro repositorio en GitHub y clickearemos en el icono del engranaje para acceder a la configuración. Iremos a la sección de “acceso” y luego en “colaboradores”, desde acá podremos agregar los nombres de usuarios que queramos que tengan acceso a nuestro repositorio.

25) ¿Qué es un repositorio público en GitHub?

Al igual que los privados, tiene la misma funcionalidad, sumado a que los repositorios públicos son accesibles para todo el mundo en Internet.

26) ¿Cómo crear un repositorio público en GitHub?

Mismo caso al crear un repositorio privado. Una vez tengamos nuestro repositorio local y tengamos una cuenta de GitHub, vamos a ir a nuestro perfil

de nuestra cuenta y en la parte superior derecha daremos click en “crear nuevo repositorio”. A continuación personalizaremos nuestro repositorio con un nombre, descripción, etc. Y luego configuraremos la accesibilidad, este puede ser privado o público. Elegimos la opción de “publico” y ya tendremos nuestro repositorio privado. Por último subimos el contenido de nuestro repositorio local a GitHub.

27) ¿Cómo compartir un repositorio público en GitHub?

De la misma forma que en el caso del repositorio privado, podemos agregar usuarios con los que queramos compartirles nuestro repositorio. Pero la forma más conveniente en este caso sería compartir el link del repositorio de GitHub con solo copiar el link de la página.

Ejercicio 02:

https://github.com/mz-juan/TP2Ejercicio02.git

Ejercicio 03:

https://github.com/mz-juan/conflict-exercise