Trabajo práctico N° 2

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

• ¿Qué es GitHub?
Github es una plataforma en línea para compartir, divulgar, colaborar o descargar repositorios de otros programadores y propios creando una comunidad util a tales fines.

• ¿Cómo crear un repositorio en GitHub?
Para crear un repositorio, luego de haberse creado una cuenta y logueado, en la esquina superior derecha vamos a “Crear nuevo” → “Nuevo repositorio” → completar los datos que me pide → click en “Crear”

• ¿Cómo crear una rama en Git?
Con el siguiente comando: git branch nombre_de_la_rama. Me creará una rama extra a la de “main”

• ¿Cómo cambiar a una rama en Git?
con el comando git checkout nombre_de_la_rama

• ¿Cómo fusionar ramas en Git?
con el comando git merge nombre_de_la_rama (Si estoy en la rama A, me fusiona rama B a rama A. Depende la rama que yo esté parado)

• ¿Cómo crear un commit en Git?
con el comando git commit -m “Descripción del cambio”. Esto hará que los cambios que yo haya hecho hasta ese momento se me guarden y sirvan como punto de retorno si quisiera

• ¿Cómo enviar un commit a GitHub?
a través del comando git push. Eso hará que desde Git envíe el código actualizado a GitHub.

• ¿Qué es un repositorio remoto?
Es un código escrito por nosotros u otro programador que está subido en una plataforma/nube de alojamiento.

• ¿Cómo agregar un repositorio remoto a Git?
Lo puedo hacer a través de una copia utilizando el comando git clone URL_del_repositorio en consola o situándome en el directorio donde quiero tener el repositorio. Ahí con el comando git remote add utilizo el argumento origin (nombre remoto por default) y URL_del_repositorio. Quedaria: “git remote add origin https://github.com/Maxi2495/primer-repo-git-utn”

• ¿Cómo empujar cambios a un repositorio remoto?
Utilizando el comando git push origin nombre_de_la_rama. Esto lo que hará es subir los cambios desde GIT hacia GitHub actualizando el repositorio remoto.

• ¿Cómo tirar de cambios de un repositorio remoto?

Lo hago a través del comando git pull. Con esto descargo un repositorio remoto y actualizo el local.

• ¿Qué es un fork de repositorio?
Un fork de repositorio es una copia exacta de un repositorio pero sin modificar el original. Una copia en nuestra cuenta. La cual puedo editar, sacar código, sumar código siempre que tenga el permiso del autor original. Sirve para colaborar.

• ¿Cómo crear un fork de un repositorio?
Se debe ir a la página del repositorio en cuestión y hacer click en el botón “fork”

• ¿Cómo enviar una solicitud de extracción (pull request) a un repositorio?
A través del repositorio en GitHub. Voy al repositorio del autor al cual le hice un fork y le solicito un pull request. Se debe seleccionar el botón “pull request”, completar con información necesaria para comentar los cambios y enviar la solicitud.

• ¿Cómo aceptar una solicitud de extracción?
Esto sirve para aceptar una solicitud “pull request” cuando alguien hizo un fork de nuestro repositorio, realizó algunos cambios y quiere subirlo. Para ello debemos hacer click dentro de GitHub en “pull requests” → hacer click en la solicitud que se quiera revisar → hacer click en “archivos cambiado” → revisar los cambios para ver si están bien → seleccionar “aprobar” y enviar una realimentación al editor.

• ¿Qué es una etiqueta en Git?
Es una “marca” que nos sirve para identificar diferentes puntos en el historial de modificaciones. Las versiones (V1.01) son una etiqueta. Hay dos tipos: ligeras y anotadas. Las ligeras son similares a una rama que no cambia. Es una “señal” hacia un commit específico. Por otra parte, las anotadas se guardan en Git como un objeto. Tienen datos del editor (nombre, correo, fecha)

• ¿Cómo crear una etiqueta en Git?
Para crear una etiqueta anotada se puede ejecutar el comando git tag -a v1.4 -m ‘Nueva version 1.4”. “-a” indica que será una etiqueta anotada. v1.4 es la versión(puedo poner otra). con “-m” indico un mensaje, en este caso que la versión es nueva y es la 1.4.
Para crear una ligera ejecutar git tag v1.4.

• ¿Cómo enviar una etiqueta a GitHub?
Agregando “--tag” a un push. El siguiente es un ejemplo: git push origin main –tag.
“origin” es el nombre del remoto que viene en default. “main” hace referencia a la rama principal y “–tag” es el comando necesario para que me guarde una etiqueta.

• ¿Qué es un historial de Git?
Es un listado de los cambios que se han hecho en el repositorio a lo largo del tiempo.

• ¿Cómo ver el historial de Git?
A través del comando git log. En él puedo ver los cambios que se hicieron, quien los hizo y en qué momento.

• ¿Cómo buscar en el historial de Git?
Hay varias opciones para hacer este. Por mencionar dos si ejecuto el comando git log –after= “fecha” me mostrará los cambios que se han realizado despues de una fecha determinada. Otro ejemplo es utilizando el comando git log –author=”nombre_del_autor” me mostrará los cambios que hizo determinado autor.

• ¿Cómo borrar el historial de Git?
Se puede borrar la carpeta “git” que se crea oculta cuando uno realiza un “git init”.

• ¿Qué es un repositorio privado en GitHub?
Es un espacio alojado en nube con código pero al cual solo se le permite el acceso a ciertas personas. No es de acceso público.

• ¿Cómo crear un repositorio privado en GitHub?
Al momento de crear un repositorio desde la pagina de GitHub, seleccionar la opción “Privado” (tiene un dibujo de un candado cerrado).

• ¿Cómo invitar a alguien a un repositorio privado en GitHub?
Ingresado a mi repositorio en Github, voy a la opción “settings/configuración” → clickeo en el panel izquierdo “colaboradores” → confirmo con contraseña y elijo la opción “buscar gente/add people” → elijo el usuario a quien quiero compartirle el repositorio.

• ¿Qué es un repositorio público en GitHub?
Es código alojado en la nube/Github disponible para cualquier persona que desee.

• ¿Cómo crear un repositorio público en GitHub?
En la pantalla de Github, en la esquina superior derecha clickeo en “create new” → “new repository” → Completar los datos y elegir la opción “public”

• ¿Cómo compartir un repositorio público en GitHub?
Para hacerlo puedo pasar la URL de mi repositorio a quien desee compartirlo o en la pagina de mi repositorio ir a settings/configuración → en el panel izquierdo ir a collaborators → confirmar contraseña y elegir los usuarios que quisiera compartir.

2) Realizar la siguiente actividad:

RESULTADO: https://github.com/Maxi2495/UTN-TUPaD-P1/tree/master

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

• Agregando un Archivo
o Crea un archivo simple, por ejemplo, "mi-archivo.txt".
o Realiza los comandos git add . y git commit -m "Agregando mi-archivo.txt" en la línea de comandos.

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

3) Realizar la siguiente actividad:

RESULTADO: https://github.com/Maxi2495/ejercicio-conflictivo

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 estés 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