HIGA_TP-02

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

¿Qué es GitHub?

Es una plataforma en donde podemos guardar, gestionar y compartir repositorios hechos con Git. Su uso facilita el trabajo en equipo.

¿Cómo crear un repositorio en GitHub?

Dado por entendido ya se cuenta con una cuenta usuario, se puede crear un repositorio de dos maneras:

- haciendo click en el **botón verde** en el tab Repositories de tu cuenta GitHub.
- en el **+** que figura en el margen superior derecho de la cuenta

Deberemos luego darle nombre, descripción, visibilidad (privado/público) y algunas opciones más.

¿Cómo crear una rama en Git?

Para crear un branch (rama) en Git, deberemos usar el comando `git branch <nombreDeLaRama>`

Un branch permite trabajar en cambios sin afectar al proyecto principal.

¿Cómo cambiar a una rama en Git?

Para de la rama main a un branch, usamos el comando `git checkout <nombreDeLaRama>`

¿Cómo fusionar ramas en Git?

Para fusionar ramas utilizamos el comando `git branch <nombreDeLaRama>`

¿Cómo crear un commit en Git?

Primero, debemos añadir los archivos para el commit `git add .`

Luego, hacemos el commit mediante `git commit -m "Texto que describa brevemente el commit"`

¿Cómo enviar un commit a GitHub?

Una vez hecho el commit, se utiliza el comando `git push origin main` para subirlo a GitHub

¿Qué es un repositorio remoto?

Es la versión del proyecto que está alojada en la nube (GitHub en nuestro caso). La versión que está en nuestra PC sería la local.

¿Cómo agregar un repositorio remoto a Git?

Para agregar un repositorio remoto a Git, podemos usar

- `git clone url` para clonar un repositorio remoto
- `git remote add origin url` para añadir un nuevo repositorio remoto

¿Cómo empujar cambios a un repositorio remoto?

Utilizamos el comando `git push -u origin master`

Luego de la 1era vez, podemos utilizar `git push` o `git push --tags` (sube cambios incluyendo los tags)

¿Cómo tirar de cambios de un repositorio remoto?

Para tirar de cambios, utilizamos `git pull origin master` o `git pull` (si usamos -u durante el pull)

¿Qué es un fork de repositorio?

Es una copia independiente de un repositorio, que se crea en tu cuenta. Podemos trabajar sobre él, sin afectar al proyecto principal.

¿Cómo crear un fork de un repositorio?

En la esquina superior derecha de la página, haga clic en **Fork**.

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

Dentro del fork -> Contribute -> Open pull request

¿Cómo aceptar una solicitud de extracción?

- Dentro del repositorio en GitHub, click en **Pull Requests**
- Seleccionar el pull request a aceptar
- Click en **Review changes** para revisar los cambios
- Click en **Submit review** para confirmar

¿Qué es un etiqueta en Git?

Los tags se utilizan para marcar determinados puntos en el historial de commits. Son generalmente usados para marcar momentos clave del proyecto.

¿Cómo crear una etiqueta en Git?

Para crear un tag:

`git tag <nombredeltag>`

`git tag -a <nombredeltag> -m <mensaje>` crear tag con mensaje

¿Cómo enviar una etiqueta a GitHub?

`git push origin --tags` sube todos los tags (en el caso de haber varios)

`git push origin <nombredeltag>`

¿Qué es un historial de Git?

Es el registro completo de todos los cambios en el repositorio.

Permite rastrear cambios, saber quien los hizo, y revertir versiones anteriores

¿Cómo ver el historial de Git?

`git log` muestra sólo los commits confirmados

`git reflog` muestra todo (inclusive commits no confirmados, resets, reverts, checkouts, etc

¿Cómo buscar en el historial de Git?

`git log --grep="texto a buscar"`

`git log -S "texto a buscar"` <- por palabra clave

¿Cómo borrar el historial de Git?

```
git checkout --orphan nueva-rama
git add .
git branch -D main
git branch -m main
git push --force
```

¿Qué es un repositorio privado en GitHub?

Es un repositorio visible solo para el creador y los usuarios que han sido invitados al mismo.

¿Cómo crear un repositorio privado en GitHub?

Al momento de crear el repositorio, elegir la opción **Private**.

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

Dentro del repositorio: Settings -> Collaborators -> Buscar usuario, agregarlo y asignarle un rol

¿Qué es un repositorio público en GitHub?

Es un repositorio de libre accesso. El código y el resto es visible para quien lo desee.

La edición está restringida a usuarios con permiso. Alternativamente, se puede hacer fork y enviar un pull request.

¿Cómo crear un repositorio público en GitHub?

Al momento de crear el repositorio, elegir la opción **Public**.

¿Cómo compartir un repositorio público en GitHub?

Compartir la URL del mismo (por ej, https://github.com/emegiga/UTN-TUPaD-P1)

2) Realizar la siguiente actividad:

https://github.com/emegiga/prog1_tp2_ej2

- **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:

https://github.com/emegiga/prog1_tp2_ej3

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. """