

## Práctico 2: Git y GitHub

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

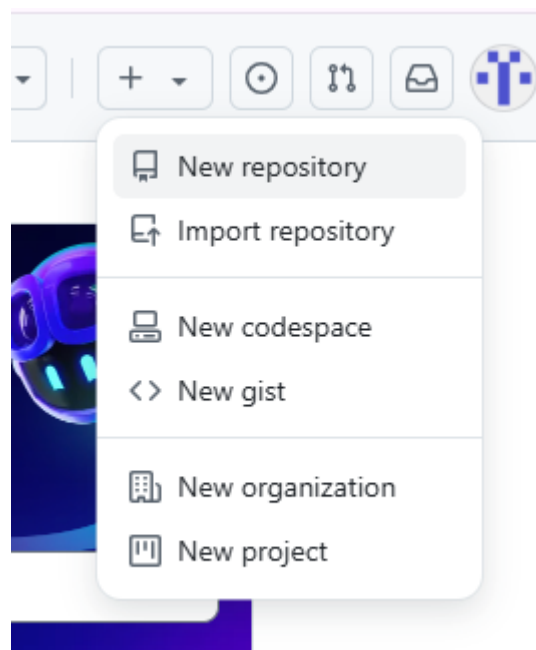
#### Respuestas

- ¿Qué es GitHub?

GitHub es una plataforma de repositorios remotos que utiliza Git para el control de versiones. Permite almacenar y gestionar código de forma eficiente, al que puedo acceder desde cualquier dispositivo con conexión a internet. Nos permite compartir código con otras personas, ir guardando las distintas modificaciones, colaborar en otros procesos, y trabajar en conjunto en un mismo proyecto, desde distintos dispositivos. Puede ser de uso público o privado

- ¿Cómo crear un repositorio en GitHub?

Inicia sesión en GitHub, despliega las opciones de “+” (Create new...) y selecciona “New Repository”.



Una vez seleccionado, se abrirá una página donde deberás completar la información del repositorio:

Nombre

Descripción (opcional)

Seleccionar si es público o privado

Indicar si necesitamos que agregue un archivo README entre otras configuraciones.

Otras configuraciones

Una vez completa toda la información, se debe dar click en “Create repository” para finalizar el proceso.

- **¿Cómo crear una rama en Git?**

Ejecutando **git branch {nombre de la rama a crear}**

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

Verifico en que rama estoy trabajando, ejecutando **git Branch**.

Una vez verificado ejecuto **git ckeckout {nombre de la rama a la que quiero cambiar}**

- **¿Cómo fusionar ramas en Git?**

Desde la rama a la que quiero que se apliquen los cambios ejecuto **git merge {nombre de la rama que se va a fusionar}**

- ¿Cómo crear un commit en Git?

Ejecutando **git commit -m "{mensaje que describa la modificación}"**

- ¿Cómo enviar un commit a GitHub?

Ingresando **git push origin {nombre de la rama}**

- ¿Qué es un repositorio remoto?

Un repositorio remoto, es un repositorio que se encuentra en la red, al que puedo acceder desde cualquier servidor.

- ¿Cómo agregar un repositorio remoto a Git?

Desde la carpeta local que deseo agregar al repositorio remoto, debo ejecutar el siguiente comando en git:

**Git init -> inicia el repositorio**

**Git add . -> Agrega archivos**

**Git commit -m "{nombre}" -> hace un commit de los cambios**

**Git remote add origin {URL} \_> Agrega el reposito remoto**

**Git push -u origin {rama} -> Sube los cambios**

- ¿Cómo empujar cambios a un repositorio remoto?

Indicando en Git:

**Git Add .**

**Git commit -m "{descripción del cambio}"**

**Git push -u origin {rama} o Git push (luego de la primera vez)**

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

Puedo utilizar los siguientes comandos:

**Git pull origin {rama}** → Aplica cambios desde el repositorio remoto

**Git pull (Si usamos -u en el push)** → Aplica cambios desde el repositorio remoto

**Git fetch** → Me muestra los cambios en el remoto, pero no los aplica

- **¿Qué es un fork de repositorio?**

Es un clon/copia de un repositorio ya subido a GitHub. Al clonarlo vamos a tener la copia/clon del repositorio en nuestra cuenta.

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

Desde el repositorio GitHub que quiero clonar, se da click en Fork



El repositorio debe ser Publico.

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

Se debe clonar el repositorio fork en nuestro repositorio local. Para eso se debe ejecutar el comando:

**git clone {url del repositorio}**

Una vez realizado eso, se debe crear una nueva rama para los cambios, navegando en el directorio del repositorio clonado:

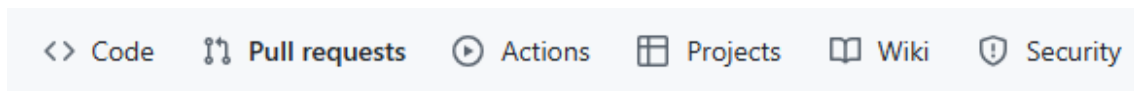
**cd <nombre repositorio>**

Se cambia a la nueva rama

**Git checkout -b [{ama}]**


Una vez realizados y subidos los cambios:

Desde el repositorio clonado, se hace clic en “Pull Requests”



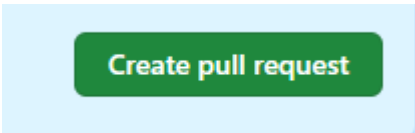
.

Hacer clic en “New pull request”



Se abrirá una página que comparará las modificaciones realizadas en nuestro repositorio con el repositorio “original”

Damos clic en “Create Pull request”



En la página que se abre, se podrá indicar el título, y la descripción de cuáles

fueron las modificaciones realizadas, explicando porque consideramos que sería mejor aplicar estos cambios sobre el repositorio “original”

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

Desde el repositorio en el que se encuentra la solicitud de extracción, se debe dar clic en la pestaña "Pull requests" para ver todas las solicitudes de extracción.

Seleccionar la solicitud de extracción que desees aceptar.

Revisar los cambios propuestos

Si aceptas los cambios, haz clic en el botón "Review changes" en la solicitud de extracción.

Selecciona "Approve" y añade un comentario si desees. Luego, haz clic en "Submit review"

Una vez aprobada, haz clic en el botón "Merge pull request".

Añade un mensaje de confirmación si es necesario y haz clic en "Confirm merge".

Después de fusionar la solicitud de extracción, puedes eliminar la rama fuente si ya no es necesaria. GitHub te dará la opción de eliminar la rama justo después de la fusión.

- **¿Qué es un etiqueta en Git?**

Las etiquetas son puntos específicos el historial de commits. Indican puntos importantes en las modificaciones que se van realizando

Las etiquetas pueden ser anotadas o ligeras.

En las etiquetas anotadas se incluyen mensaje de etiquetados, el nombre del etiquetador, la fecha y pueden ser firmadas y verificadas, las etiquetas ligeras no contienen esta información

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

Para una etiqueta ligera se ejecuta el comando:

**Git tag {nombre de la etiqueta}**

Para una etiqueta anotada se ejecuta el comando:

**Git tag -a {nombre de la etiqueta} -m “Descripción”**

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

Para subir la etiqueta a un repositorio remoto se debe ejecutar:

**git push origin {nombre de la etiqueta}**

o

**git push origin -tags** (para subir todas las etiquetas)

- **¿Qué es un historial de Git?**

El historial es el registro de todos los cambios realizados en el repositorio.

- **¿Cómo ver el historial de Git?**

Para ver el historial se debe ejecutar **git log**. Esto me mostrará todos los commits que se hayan realizado. Cada commit representa un punto específico en el desarrollo del proyecto.

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

Ejecutando **git log**, voy a ver los comandos en orden cronológico inverso, desde lo actual a lo más antiguo.

Cada entrada incluye información como:

- El identificador único del commit (SHA-1 hash)
- El autor del commit
- La fecha y hora del commit
- El mensaje del commit

```
vcalcerano@NB1007 MINGW64 ~/Documents/utn/UTN-TUPaD-P1/UTN-TUPaD-P1 (main)
$ git log
commit c08d4db5d1200368ed91685f7bfba87f5ba28e96 (HEAD -> main, origin/main, origin/HEAD)
Author: sbruselario <sbruselario@gmail.com>
Date: Mon Mar 17 22:06:33 2025 -0300

    😊 ¡Bienvenido a Programación 1!

vcalcerano@NB1007 MINGW64 ~/Documents/utn/UTN-TUPaD-P1/UTN-TUPaD-P1 (main)
$
```

Hay opciones adicionales como:

**Git log -p** → Muestra las diferencias entre los commits.

**Git log --oneline** → Muestra el historial más compacto.

**Git log -n 5** → Limita el N° de commits mostrados

**Git log --author= "Nombre del autor"** → Filtra por autor

**git log --grep="Texto de búsqueda"** → Busca por mensaje de commit

**git log --since="xxxx-xx-xx" --until="xxxx-xx-xx"** → Buscar entre fechas



**git show <hash\_parcial>** → Busca por hash

**git log -- path/al/archivo** → Busca por archivo

Estas opciones también se pueden combinar, para crear búsquedas más precisas.

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

Si se quiere eliminar todo el historial lo que debo hacer es crear una nueva rama:

**Git checkout {nueva rama}**

Añadir todos los archivos y hacer un commit inicial:

**git add .**

**git commit -m "Commit inicial"**

Eliminar la rama original y ponerle otro nombre a la nueva rama.

**git branch -D {rama}**

**git branch -m {rama}**

Por último se debe forzar el push al repositorio remoto.

**git push -f origin {rama}**

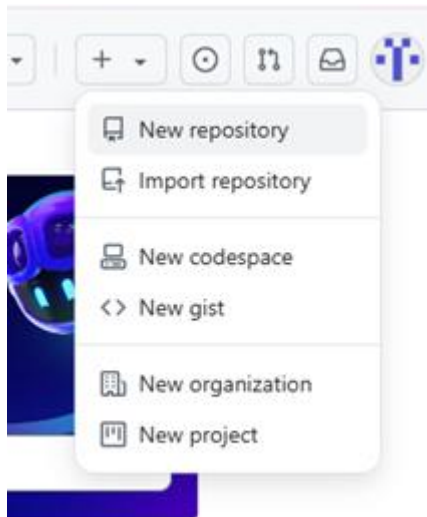
Lo que va a hacer esto es sobrescribir la rama original, con una vacía, por lo que se elimina todo el historial de commits anteriores.

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

Un repositorio privado es un repositorio al cuál solo pueden acceder las personas que estén autorizadas. Estos repositorios no son visibles si no se está autorizado.

- ¿Cómo crear un repositorio privado en GitHub?

Inicia sesión en GitHub, despliega las opciones de “+” (Create new...) y selecciona “New Repository”.



Una vez seleccionado, se abrirá una página donde deberás completar la información del repositorio:

Nombre

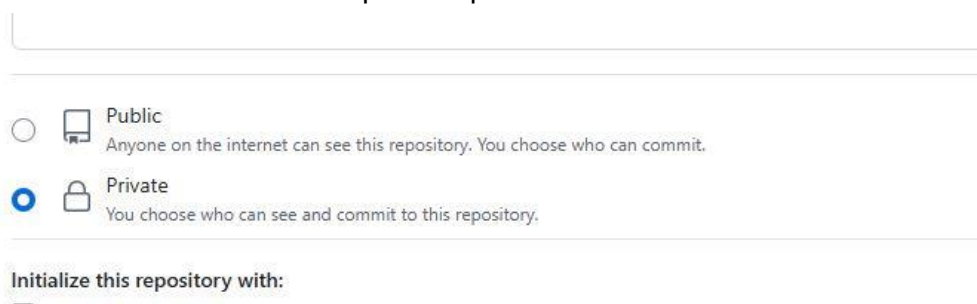
Descripción (opcional)

Seleccionar si es público o privado

Indicar si necesitamos que agregue un archivo README entre otras configuraciones.

Otras configuraciones

En estas opciones se deberá indicar que el repositorio es PRIVADO



☐ Public  
Anyone on the internet can see this repository. You choose who can commit.

☒ Private  
You choose who can see and commit to this repository.

Initialize this repository with:

Una vez completa toda la información, se debe dar click en “Create repository” para finalizar el proceso.

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

Se debe dar clic en



**Add collaborators to this repository**

Search for people using their GitHub username or email address.

Invite collaborators

Esto permite agregar a los colaboradores de GitHub a los que deseo invitar al proyecto.

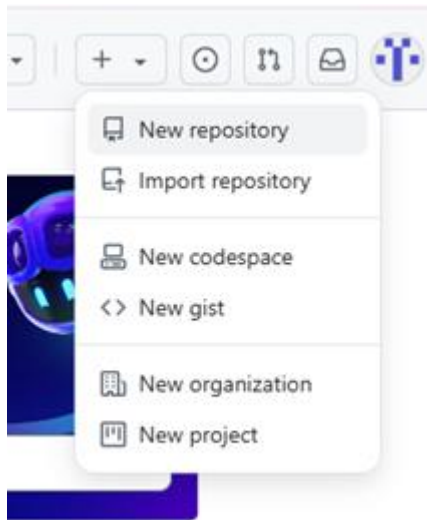
Una vez seleccionado el tipo de perfil que se quiere otorgar, se da clic en ADD para enviar la invitación.

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

Un repositorio público, es un repositorio al que puede acceder cualquier usuario GitHub. Al ser público, cualquier persona puede visualizarlo, clonarlo y sugerir cambios

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

Inicia sesión en GitHub, despliega las opciones de “+” (Create new...) y selecciona “New Repository”.



Una vez seleccionado, se abrirá una página donde deberás completar la información del repositorio:

Nombre

Descripción (opcional)

Seleccionar si es público o privado

Indicar si necesitamos que agregue un archivo README entre otras configuraciones.

Otras configuraciones

En estas opciones se deberá indicar que el repositorio es PÚBLICO



Una vez completa toda la información, se debe dar click en “Create repository” para finalizar el proceso.

- ¿Cómo compartir un repositorio público en GitHub?

Se debe acceder al repositorio, copiar la URL, y compartirla.

## 2) Realizar la siguiente actividad:

<https://github.com/VeroCal/Primer-repositorio.git>

## 3) Realizar la siguiente actividad:

<https://github.com/VeroCal/conflict-exercise.git>