

## Práctico 2: Git y GitHub

**Objetivos:** El estudiante desarrollará competencias para trabajar con Git y GitHub, aplicando conceptos fundamentales de control de versiones, colaboración en proyectos y resolución de conflictos, en un entorno simulado y guiado.

**Alumno:** Matias Mansilla

**Comisión:** M2025-3

### Actividades

1) Contestar las siguientes preguntas utilizando las guías y documentación proporcionada.

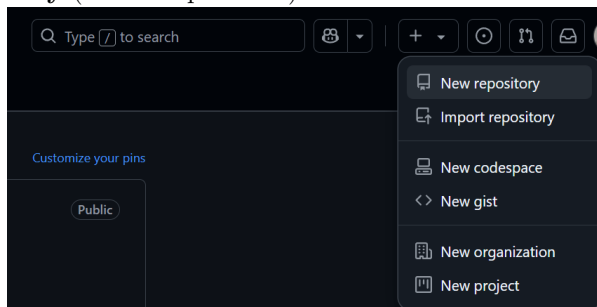
#### ¿Qué es GitHub?

GitHub es una plataforma en línea de desarrollo colaborativo que permite a programadores y equipos trabajar juntos en proyectos de software. Está basada en Git, un sistema de control de versiones que facilita gestionar cambios en el código, colaborar con otros y mantener un historial organizado de los avances.

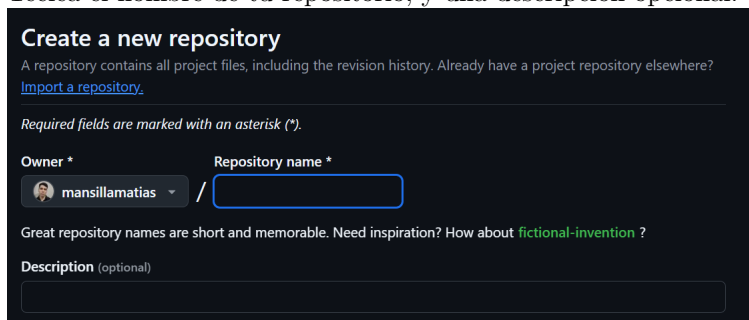
#### ¿Cómo crear un repositorio en GitHub?

Creación de un repositorio a partir de la interfaz de usuario web

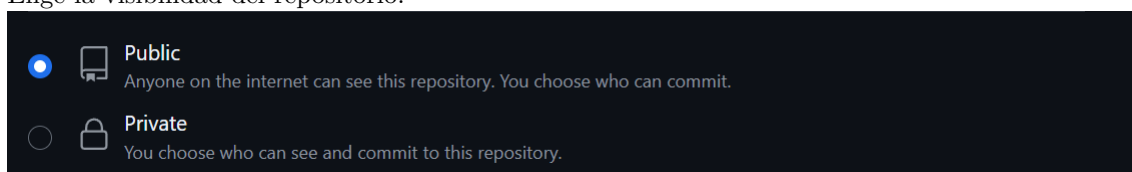
1. En la esquina superior derecha de cualquier página, selecciona + y luego haz clic en **New repository** (Nuevo repositorio).



2. Teclea el nombre de tu repositorio, y una descripción opcional.

A screenshot of the 'Create a new repository' form in GitHub. The form has a dark background with white text. At the top, it says 'Create a new repository' followed by a description: 'A repository contains all project files, including the revision history. Already have a project repository elsewhere? [Import a repository.](#)'. Below this, it says 'Required fields are marked with an asterisk (\*)'. There are two main input fields: 'Owner \*' and 'Repository name \*'. The 'Owner \*' field has a dropdown menu showing 'mansillamatias'. The 'Repository name \*' field is an empty text box. Below these fields, there is a hint: 'Great repository names are short and memorable. Need inspiration? How about [fictional-invention](#) ?'. At the bottom, there is a 'Description (optional)' field, which is also empty.

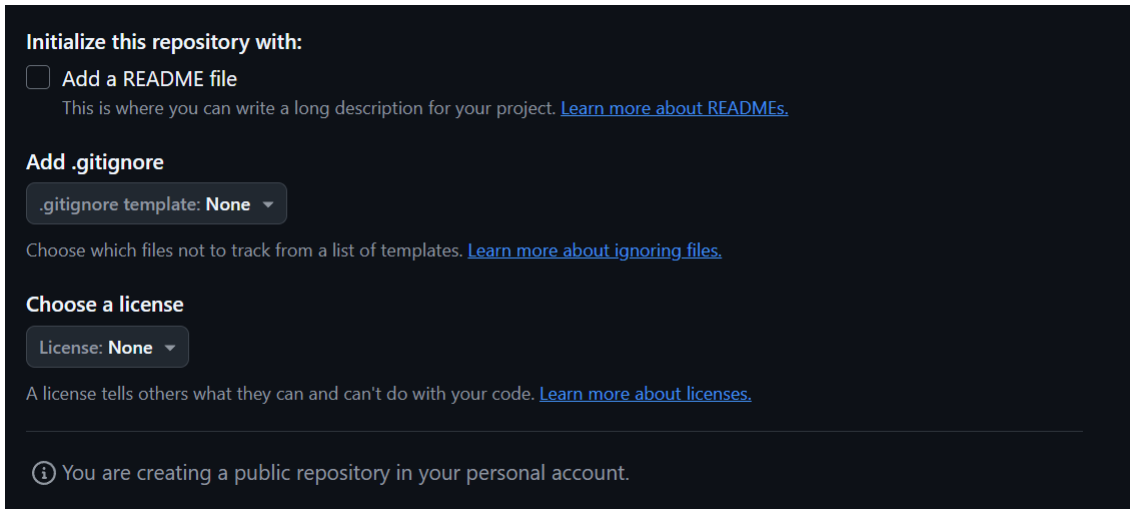
3. Elige la visibilidad del repositorio.

A screenshot of the repository visibility selection options in GitHub. There are two radio button options. The first option is 'Public', which is selected (the radio button is filled with a blue dot). Below it, it says 'Anyone on the internet can see this repository. You choose who can commit.' The second option is 'Private', which is not selected (the radio button is empty). Below it, it says 'You choose who can see and commit to this repository.'

4. De manera opcional se puede:

- Crear un README, que es un documento que describe tu proyecto.

- Crear un archivo .gitignore, que es un conjunto de reglas de omisión.
- Elegir agregar una licencia de software a tu proyecto.



**Initialize this repository with:**

☐ Add a README file  
This is where you can write a long description for your project. [Learn more about READMEs.](#)

**Add .gitignore**


.gitignore template: **None** ▾

Choose which files not to track from a list of templates. [Learn more about ignoring files.](#)


**Choose a license**

License: **None** ▾

A license tells others what they can and can't do with your code. [Learn more about licenses.](#)

 You are creating a public repository in your personal account.

5. Finalmente, haga clic en **Create repository** (Crear repositorio).



### ¿Cómo crear una rama en Git?

Para crear una rama en Git, desde la consola de Git Bash, utilizamos el comando:

```
git branch nombre-rama
```

Donde “nombre-rama” será el nombre que deseamos nombrar la nueva rama.

### ¿Cómo cambiar a una rama en Git?

Para cambiar de rama utilizamos el comando:

```
git checkout rama
```

Donde “rama” es el nombre de la rama a la cual queremos cambiar.

### ¿Cómo fusionar ramas en Git?

Se fusionan las ramas con el comando:

```
git merge rama
```

Este comando fusiona la rama que pusimos en el código, con la rama actual en la cual estamos trabajando. Por ejemplo, si estamos trabajando en la rama “main” y hacemos un merge con la rama “sub”, el contenido de la rama “sub” pasaría a estar dentro del contenido de la rama “main”.

### ¿Cómo crear un commit en Git?

Se crea un commit con el comando:

```
git commit -m "mensaje"
```

Esto confirma los cambios hechos en el proyecto o repositorio, en “mensaje” va el comentario o título que se quiera poner al cambio realizado.

## ¿Cómo enviar un commit a GitHub?

Se utiliza el comando:

```
git push
```

Tener en cuenta que la primera vez que se va a hacer este “push” se debe utilizar el comando:

```
git push -u origin main
```

el comando -u crea la referencia a cual branch (en este ejemplo siendo la rama main) queremos realizar el push, una vez utilizado, se puede trabajar solamente con git push sin necesidad de especificar nuevamente todo.

## ¿Qué es un repositorio remoto?

Un repositorio remoto es una versión de un repositorio local que está almacenada en un servidor en línea, como GitHub, GitLab u otra plataforma similar. Sirve para trabajar de manera colaborativa y compartir los cambios realizados en el código o proyecto con otros miembros del equipo.

## ¿Cómo agregar un repositorio remoto a Git?

Para agregar un repositorio remoto a Git, sigue estos pasos:

1. **Obtén la URL del repositorio remoto:** Ve a tu cuenta en una plataforma como GitHub, GitLab o Bitbucket y copia la URL del repositorio que quieres conectar (debe terminar en .git).
2. **Abre tu terminal en el proyecto local:** Ve a la carpeta de tu proyecto y abre la terminal.
3. **Conecta el repositorio remoto:** Usa el siguiente comando en tu terminal:

```
git remote add origin [URL]
```

Sustituye [URL] con la URL del repositorio remoto que copiaste antes.

## ¿Cómo empujar cambios a un repositorio remoto?

Para empujar tus cambios a un repositorio remoto en Git, sigue estos pasos:

1. **Asegúrate de que los cambios están confirmados:**

- Primero, agrega los cambios al área de stage si no lo has hecho:

```
git add .
```

- Luego, confirma los cambios con un mensaje descriptivo:

```
git commit -m "Descripción de los cambios realizados"
```

2. **Empuja los cambios al repositorio remoto:**

- Si es la primera vez que empujas, utiliza este comando para establecer la rama remota:

```
git push -u origin main
```

- En adelante, simplemente usa:

```
git push
```

Esto subirá los cambios de la rama local a la rama principal (main) del repositorio remoto.

## ¿Cómo tirar de cambios de un repositorio remoto?

Para tirar de cambios de un repositorio remoto en Git, sigue estos pasos:

### 1. Asegúrate de estar en la rama correcta:

- Antes de realizar el pull, verifica en qué rama estás trabajando:

```
git branch
```

- Cambia a la rama deseada si es necesario:

```
git checkout nombre-rama
```

### 2. Usa el comando:

```
git pull
```

Esto sincronizará tu rama local con la versión más reciente en el repositorio remoto.

Si necesitas especificar la rama remota, puedes usar:

```
git pull origin nombre-rama
```

Este comando combinará los cambios en tu rama local automáticamente, y si hay conflictos, Git te pedirá que los resuelvas antes de completar el proceso.

## ¿Qué es un fork de repositorio?

Un fork de un repositorio es una copia independiente del repositorio original que se crea para poder trabajar en él sin afectar el proyecto original. Es muy común en plataformas como GitHub cuando quieres colaborar con un proyecto, pero no tienes permisos directos para modificar el código.

Con un fork, puedes:

- Realizar modificaciones en tu copia del repositorio.
- Probar nuevas características sin riesgo de cambiar el original.
- Contribuir al proyecto original enviando tus cambios mediante una pull request.

En esencia, un fork te permite tomar un proyecto existente y hacer una versión personalizada, que luego puedes compartir o integrar al repositorio original según las necesidades del desarrollo.

## ¿Cómo crear un fork de un repositorio?

Para crear un fork de un repositorio, se siguen los siguientes pasos:

1. **Ve al repositorio original:** Abre el repositorio que deseas bifurcar en GitHub.
2. **Haz clic en el botón “Fork”:** Este botón se encuentra en la esquina superior derecha de la página del repositorio. Al hacer clic, se creará automáticamente una copia del repositorio en tu cuenta.
3. **Personaliza tu fork (opcional):** Cuando inicies el proceso de bifurcación, puedes elegir un nombre para tu fork o dejar el nombre original. También puedes añadir una descripción personalizada.

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

Enviar una solicitud de extracción (pull request) es fundamental para proponer cambios a un repositorio. Aquí tienes los pasos básicos:

### 1. Crear una rama para los cambios:

- Asegúrate de estar en tu fork del repositorio.
- Crea una nueva rama para trabajar en los cambios:

```
git checkout -b nombre-de-rama
```

- Haz los cambios necesarios en tu proyecto y confírmalos:

```
git add .  
git commit -m "Descripción de los cambios"
```

## 2. Sube la rama a tu fork:

Sube la nueva rama al repositorio remoto:

```
git push origin nombre-de-rama
```

## 3. Crea la pull request:

- Ve a la plataforma donde se encuentra el repositorio (por ejemplo, GitHub).
- Navega a tu fork y haz clic en el botón que aparece al subir la rama, normalmente dice “Compare & pull request”.
- Asegúrate de que la rama de tu fork esté configurada para fusionarse con la rama correspondiente del repositorio original, como main.
- Agrega un título y descripción clara sobre los cambios que realizaste.
- Haz clic en “Create pull request”.

## 4. Espera la revisión:

- El mantenedor del repositorio original revisará tu solicitud de extracción.
- Podrían solicitar cambios o aprobarla y fusionarla directamente.

## ¿Cómo aceptar una solicitud de extracción?

Aceptar una solicitud de extracción (pull request) implica revisar los cambios propuestos y fusionarlos al repositorio si son adecuados. Aquí están los pasos principales:

1. **Abre el repositorio en GitHub:** Ve a la pestaña de **Pull Requests** en el repositorio.
2. **Selecciona la solicitud de extracción:** Haz clic en la solicitud de extracción que deseas revisar.
3. **Revisa los cambios propuestos:** Puedes comparar los cambios con el código actual utilizando la interfaz de comparación que proporciona GitHub. Si tienes comentarios o sugerencias, puedes escribirlos en el área de discusión.
4. **Aprueba los cambios (si son correctos):** Usa el botón **Approve** o simplemente escribe un comentario indicando que los cambios están listos para fusionarse.
5. **Fusiona los cambios:** Haz clic en el botón **Merge pull request**. En la ventana de confirmación, selecciona **Confirm Merge** para completar la fusión.
6. **Opcional: elimina la rama:** Si la rama creada para la solicitud ya no es necesaria, puedes eliminarla después de la fusión haciendo clic en **Delete branch**.

## ¿Qué es un etiqueta en Git?

En Git, una etiqueta (tag) se utiliza para marcar puntos específicos en el historial de commits de un proyecto, generalmente para identificar versiones importantes, como lanzamientos de software. Por ejemplo, puedes etiquetar un commit con “v1.0” para indicar que es la primera versión estable de tu proyecto. A diferencia de las ramas, las etiquetas son inmutables, es decir, no cambian con el tiempo.

Existen dos tipos principales de etiquetas en Git:

- **Etiquetas livianas (lightweight tags):** Son como un marcador simple, sin información adicional.
- **Etiquetas anotadas (annotated tags):** Incluyen un mensaje, fecha, y la información del autor, y suelen ser más utilizadas porque contienen metadatos adicionales.

## ¿Cómo crear una etiqueta en Git?

Para crear una etiqueta en Git, puedes seguir estos pasos:

1. **Etiqueta liviana (simple marcador):**

Para crear una etiqueta básica sin descripción adicional, usa este comando:

```
git tag nombre-etiqueta
```

Sustituye nombre-etiqueta con el nombre que quieras asignar, por ejemplo, v1.0.

2. **Etiqueta anotada (con metadatos):**

Si necesitas agregar detalles como un mensaje, autor o fecha, utiliza el siguiente comando:

```
git tag -a nombre-etiqueta -m "Mensaje descriptivo"
```

Por ejemplo:

```
git tag -a v1.0 -m "Versión estable del proyecto"
```

## ¿Cómo enviar una etiqueta a GitHub?

Una vez creada la etiqueta, sube la etiqueta al repositorio remoto:

```
git push origin nombre-etiqueta
```

Si quieres empujar todas las etiquetas creadas de una vez, utiliza:

```
git push --tags
```

## ¿Qué es un historial de Git?

Un historial de Git es el registro completo de todos los commits realizados en un proyecto. Este historial incluye información esencial, como:

- **Autor:** quién hizo cada cambio.
- **Fecha y hora:** cuándo se realizó el commit.
- **Mensaje:** la descripción o título que se asignó al cambio.

## ¿Cómo ver el historial de Git?

Puedes ver el historial de tu proyecto usando el comando:

```
git log
```

Este comando mostrará todos los commits en orden cronológico inverso, desde el más reciente hacia el más antiguo.

## ¿Cómo buscar en el historial de Git?

Para buscar en el historial de Git y encontrar información específica, puedes utilizar los siguientes comandos:

1. **Ver todo el historial:**

Usa el comando básico:

```
git log
```

Esto mostrará el historial completo de commits en orden cronológico inverso.

2. **Buscar un commit con una palabra clave o patrón:**

Usa el comando:

```
git log --grep="palabra clave"
```

Por ejemplo, para buscar commits relacionados con "bugfix":

```
git log --grep="bugfix"
```

### 3. Mostrar un historial más compacto:

Si quieres un resumen más breve:

```
git log --oneline
```

### 4. Buscar un cambio específico en un archivo:

```
git log -- nombre-del-archivo
```

### 5. Buscar contenido o código específico en el historial:

Si quieres encontrar un cambio relacionado con un fragmento específico de código, usa:

```
git log -S "fragmento de código"
```

### 6. Ver diferencias específicas en los commits:

```
git log -p
```

## ¿Cómo borrar el historial de Git?

Borrar el historial de Git implica remover los commits existentes de un repositorio. Ten en cuenta que este proceso es irreversible y puede causar problemas si se está colaborando con otras personas en un proyecto.

Borra el historial en el repositorio local ejecutando estos comandos:

```
rm -rf .git  
git init
```

Esto eliminará por completo el historial y reiniciará el repositorio Git local.

## ¿Qué es un repositorio privado en GitHub?

Un repositorio privado en GitHub es un repositorio que no es visible públicamente. Solo las personas a las que tú les otorgues acceso podrán verlo y trabajar en él.

## ¿Cómo crear un repositorio privado en GitHub?

Al crear un repositorio, selecciona **Private** en la configuración de visibilidad para que tu repositorio no sea público.

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

Para invitar a alguien a un repositorio privado en GitHub sigue estos pasos:

### 1. Abre el repositorio privado:

Ve al repositorio al que deseas invitar colaboradores.

### 2. Accede a la configuración de colaboradores:

- Haz clic en el botón **Settings** (Configuración) en la parte superior del repositorio.
- En el menú de la izquierda, busca la sección **Collaborators and teams** o simplemente **Manage access** (Administrar acceso).

### 3. Invita al colaborador:

- Haz clic en el botón **Invite collaborators** (Invitar colaboradores).

- Ingresa el nombre de usuario de GitHub o la dirección de correo electrónico de la persona que deseas invitar.
  - Selecciona el nombre cuando aparezca en los resultados y haz clic en **Add**.
4. **Otorga permisos:**  
Puedes ajustar los permisos del colaborador, como si tendrá acceso de solo lectura o permisos para editar el repositorio.
  5. **Confirma la invitación:**  
El colaborador recibirá una notificación y deberá aceptar la invitación para obtener acceso al repositorio.

### ¿Qué es un repositorio público en GitHub?

Un repositorio público en GitHub es un repositorio que está disponible para que cualquier persona en Internet lo vea y acceda. Esto significa que el código y los archivos del proyecto son completamente abiertos y visibles, permitiendo a otros usuarios explorar, descargar e incluso contribuir (si se les da permiso) al proyecto.

### ¿Cómo crear un repositorio público en GitHub?

Es la opción por defecto al crear un repositorio, en la sección de visibilidad, selecciona **Public** (Público).

### ¿Cómo compartir un repositorio público en GitHub?

Para compartir un repositorio público en GitHub con otros, debes obtener la URL del repositorio:

1. Ve a la página principal de tu repositorio en GitHub.
2. Haz clic en el botón **Code** (ubicado cerca del área superior derecha).
3. Copia la URL HTTPS, SSH o GitHub CLI del repositorio.

**2)** El ejercicio de la actividad 2 se encuentra realizado en el siguiente repositorio:

<https://github.com/mansillamatias/UTN-TUPaD-Repositorio-TP2>

**3)** El ejercicio de la actividad 3 se encuentra realizado en el siguiente repositorio:

<https://github.com/mansillamatias/UTN-TUPaD-Conflict-Exercise-TP2>