

Resolución Práctico 2: Git y GitHub

Alumno: Grassi, Pablo Javier

Actividades

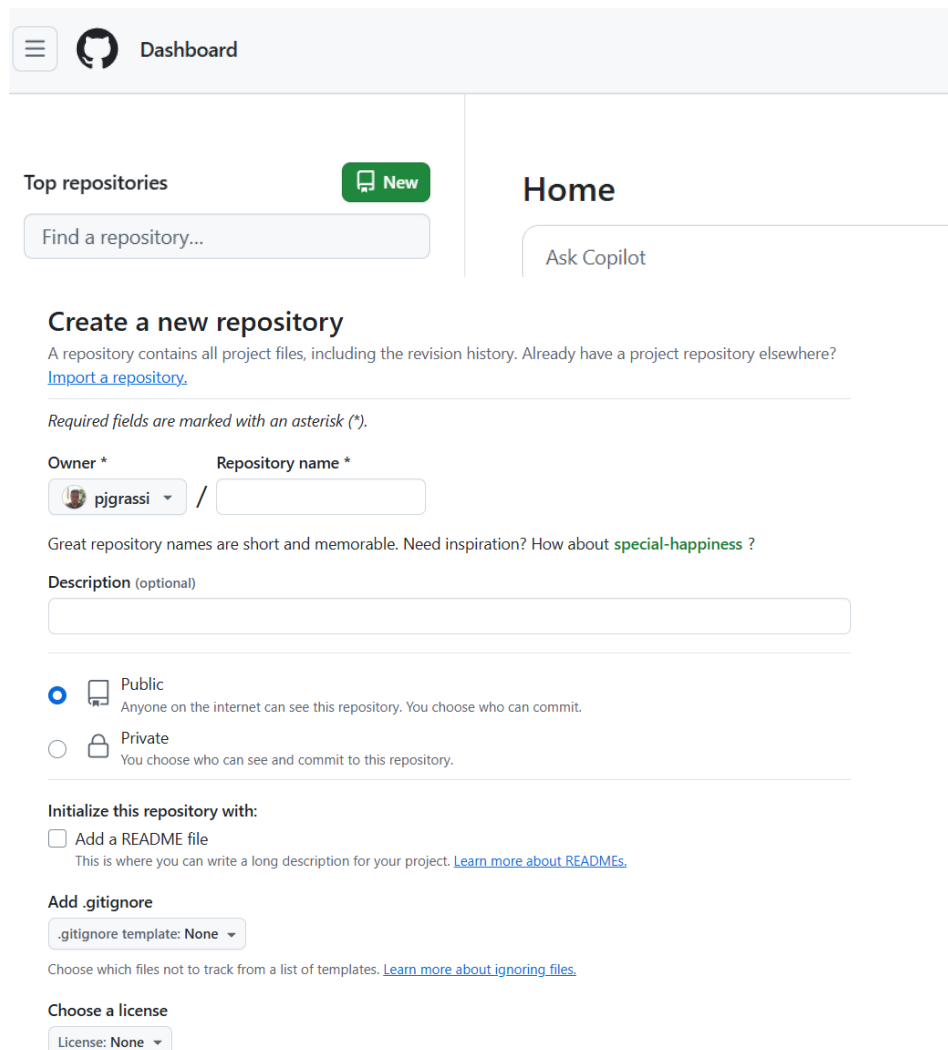
- 1) Contestar las siguientes preguntas utilizando las guías y documentación proporcionada (Desarrollar las respuestas) :

- ¿Qué es GitHub?

Es una plataforma que permite a los desarrolladores subir sus proyectos, registrar cambios, permite colaboración, como así también compartir los mismos. Es de código abierto y gratuito.

- ¿Cómo crear un repositorio en GitHub?

Primero que nada se debe crear y configurar la nueva cuenta en GitHub. Luego en el Dashboard se realiza click en el botón NEW. Luego se debe configurar el nuevo repositorio (nombre, descripción, publico o privado, etc...). Finalmente se puede ingresar al mismo y obtener la url para luego en la aplicación que usemos realizar el git clone y descargarlo a nuestra pc, para luego trabajar en el.



Dashboard

Top repositories [New](#)

Find a repository...

Home


Ask Copilot

Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere? [Import a repository.](#)


Required fields are marked with an asterisk (*).


Owner * Repository name *

 pjgrassi /

Great repository names are short and memorable. Need inspiration? How about [special-happiness](#) ?

Description (optional)

☒  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:

☒ 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](#)

- ¿Cómo crear una rama en Git?

Para crear una rama en git se utiliza el comando: `git branch nuevaRama`

- ¿Cómo cambiar a una rama en Git?

Para cambiar y trabajar sobre una rama se utiliza el comando: `git checkout nuevaRama`.

- ¿Cómo fusionar ramas en Git?

Para fusionar dos ramas se utiliza el siguiente comando: `git merge`. Por ejemplo tenemos la rama main, y la nuevaRama, primero vamos a la rama main con `git checkout main`, y luego la fusionamos con la nuevaRama con `git merge nuevaRama`.

- ¿Cómo crear un commit en Git?

Un commit es para registrar cambios que se hayan realizado en el proyecto, la idea es realizarlos para que quede registro de los mismos y luego poder movernos y regresar atrás si es necesario. Una vez realizados los cambios se procede con el comando `git add nombreDelArchivo` o `git add .` para agregar todos los archivos, luego se procede con el commit con el comando `git commit -m <mensaje de referencia del commit>`.

- ¿Cómo enviar un commit a GitHub?

Para subir finalmente los cambios a GitHub se debe utilizar el comando `git push origin <nombre de la branch>`.

- ¿Qué es un repositorio remoto?

Son versiones de los proyectos que se encuentran en la red, ósea remotos. Los mismos se pueden o no modificar, compartir y trabajar.

- ¿Cómo agregar un repositorio remoto a Git?

Se puede agregar un repositorio remoto con el comando `git remote add <nombre> <url>`.

- ¿Cómo empujar cambios a un repositorio remoto?

Para empujar los cambios se realiza con el comando `git push origin <nombre de la rama>`.

- ¿Cómo tirar de cambios de un repositorio remoto?

Para tirar los cambio en mi repositorio local se procede con el comando `git pull origin <nombre de la rama>`.

- ¿Qué es un fork de repositorio?

El fork lo que nos permite es ir a una url de algún repositorio que nos compartieron o encontramos para realizar una copia del mismo en nuestra cuenta de GitHub y así puedes trabajar sobre el sin modificar el original, digamos trabajando sobre nuestra propia copia.

- ¿Cómo crear un fork de un repositorio?

Una vez que se accede al link del repositorio de GitHub se hace click sobre el icono de fork que se encuentra arriba a la derecha. Esto nos permitirá realizar la copia en nuestra cuenta de dicho repo.

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

El pull request es enviar una solicitud de que se suban los cambios realizados a un repositorio, el autor del mismo es quien evalúa si lo acepta o no dependiendo de su análisis. En la página de GitHub ya dentro del repo en la ventana arriba a la izquierda se encuentran los botones de pull request, te permite agregar la descripción necesaria para poder especificar los cambios introducidos, también se ven aquellos archivos que se modifican.

- ¿Cómo aceptar una solicitud de extracción?

El usuario autor puede aceptar con merge pull request si así lo considera de acuerdo a su análisis previo de los cambios detallados por el solicitante.

- ¿Qué es una etiqueta en Git?

Una etiqueta sirve para marcar puntos específicos y relevantes de un proyecto como la versión V1.0 por ejemplo.

- ¿Cómo crear una etiqueta en Git?

Se pueden crear etiquetas con el comando `git tag V1.0`

- ¿Cómo enviar una etiqueta a GitHub?

Una vez creada la etiqueta para enviarla se procede con el comando `git push origin V1.0` o sino se reemplaza el nombre de la etiqueta por `--tags` y envía todas las etiquetas juntas.

- ¿Qué es un historial de Git?

El historial es el registro de los cambios realizados, ósea de los commit enviados con sus datos.

- ¿Cómo ver el historial de Git?

Para ver el historial se usa el comando `git log`.

- ¿Cómo buscar en el historial de Git?

Para buscar en el historial, según nombre de archivos se usa `git log --<nombre del archivo>`, según palabra clave se usa `git log --grep=<palabra clave>`, según fecha se usa `git log --since=<fecha desde> --until=<fecha hasta>` y según autor se usa `git log --author=<nombre del autor>`.

- ¿Cómo borrar el historial de Git?

Para borrar el historial se utiliza el comando `git reset`.

- ¿Qué es un repositorio privado en GitHub?

Es aquel repositorio donde solo puede ingresar aquel que está explícitamente autorizado por el autor.

- ¿Cómo crear un repositorio privado en GitHub?

En el proceso de creación del repositorio existe la opción de publico o privado, es allí donde se elige privado en este caso.

Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere? [Import a repository.](#)



Required fields are marked with an asterisk (*).

Owner * Repository name *

 pjgrassi /

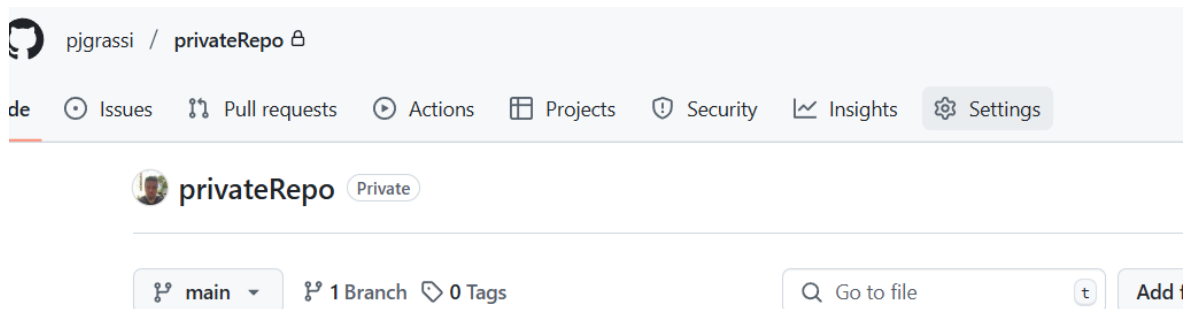
Great repository names are short and memorable. Need inspiration? How about **redesigned-engine** ?

Description (optional)

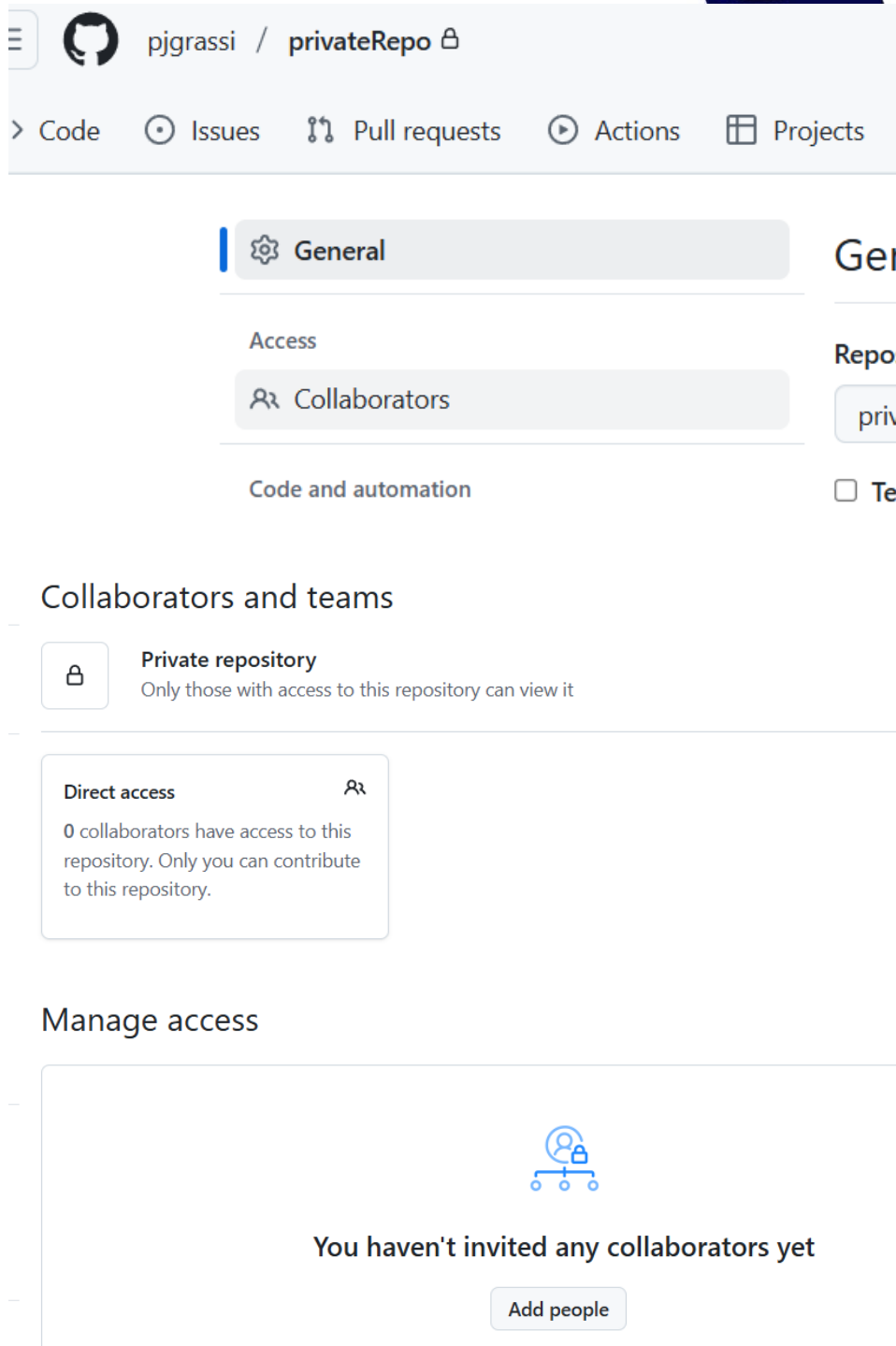
- ☐  Public
Anyone on the internet can see this repository. You choose who can commit.
- ☒  Private
You choose who can see and commit to this repository.

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

Ingresando al repo privado que se creo previamente, se hace click en settings, luego a la izquierda ingresar a Collaborators, y luego ir a Add People.



The screenshot shows the GitHub interface for a repository named 'privateRepo' owned by 'pjgrassi'. The repository is marked as 'Private'. The navigation bar includes links for Issues, Pull requests, Actions, Projects, Security, Insights, and Settings. Below the navigation bar, there are statistics for the repository: 1 Branch and 0 Tags. A search bar labeled 'Go to file' is visible, along with an 'Add' button.



The screenshot shows the GitHub repository settings page for a repository named 'privateRepo' owned by 'pjgrassi'. The page has a navigation bar with links for Code, Issues, Pull requests, Actions, and Projects. Below this, there are tabs for General, Access, and Code and automation. The 'Access' tab is selected, showing the 'Collaborators' section. It indicates that the repository is private and that 0 collaborators have access. A message states: '0 collaborators have access to this repository. Only you can contribute to this repository.' There is a button to 'Add people' and a 'Manage visibility' button.

- ¿Qué es un repositorio publico de GitHub?

Es un tipo de repositorio que permite el acceso de todos al mismos, digamos es de publico acceso a todo el mundo.

- ¿Cómo crear un repositorio público en GitHub?


En el proceso de creación del repositorio existe la opción de publico o privado, es allí donde se elige publico en este caso.

Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere? [Import a repository.](#)



Required fields are marked with an asterisk ().*

Owner * Repository name *

 pjgrassi /

Great repository names are short and memorable. Need inspiration? How about [special-happiness](#) ?

Description (optional)

- ☒  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:

- ☐ 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 ▾

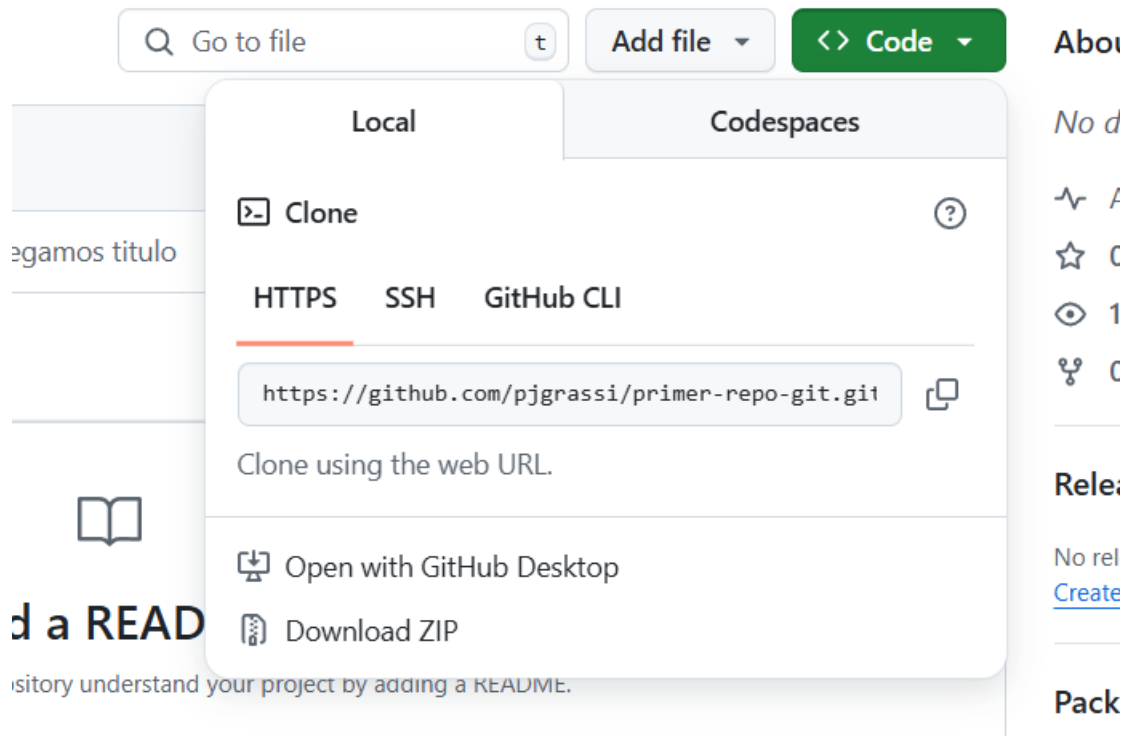
Choose which files not to track from a list of templates. [Learn more about ignoring files.](#)

Choose a license

License: None ▾

- ¿Cómo compartir un repositorio público en GitHub?

En la pgina del repo se elige el botón Code donde se despliega un menú donde se encuentra la url para compartir.



2) Realizar la siguiente actividad:

Link del repositorio del ejercicio: https://github.com/pjgrassi/TP2-GIT_GITHUB

- 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:

Link del repositorio del ejercicio: <https://github.com/pjgrassi/conflict-exercise>

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.