



Desarrollo de aplicaciones Cliente - Servidor

Trabajo Práctico Número 1

Grupo N°: 3

Integrantes:

- Acevedo Ariel - acevedoariel.ea1@gmail.com
- Acosta Gaston - gasteac@gmail.com
- Ramirez Joaquin - joaquin.ramirez170998@gmail.com
- Ruiz Franco - ruizfranco2812@gmail.com
- Romero Diego - diegoromeroofre@gmail.com
- Sosa Diego - dhsosa98@gmail.com
- Vilalta Tomas - tomasfedericovilalta@gmail.com

-2022-

Índice

Actividad 1: Informe de investigación de Sistema de Control de Versiones	2
ACTIVIDAD 1	2
Sistemas de Control de Versiones Locales	3
Sistemas de Control de Versiones Distribuidos	4
Sistemas de Control de Versiones Centralizados	5
Algunos ejemplos de SCV descentralizados	6
Git	6
Mercurial	7
Bitkeeper	7
Algunos ejemplos de SCV centralizados	7
CVS: Concurrent Versions System	7
Subversion (SVN)	8
Perforce	8
Cuadro comparativo, VCS centralizados y descentralizados	9
Cuadro comparativo, plataformas comerciales de SCV	9
ACTIVIDAD 2:	10
Mercurial	10
Mercurial vs GIT	11
Ventajas y desventajas de Mercurial	12
Implementación con un servidor gratuito	13
Primeros pasos	13
Primer commit	15
Primer push	16
Creación de ramas	17
Merge entre ramas	19
Conflictos entre ramas	20
ACTIVIDAD 3	23
Enlaces a repositorios	45

Actividad 1: Informe de investigación de Sistema de Control de Versiones

- Investigar y elaborar un breve informe de sistemas de control de versiones disponibles en el mercado, tanto del tipo centralizado como descentralizado (entre 5 y 8, ejemplo git, mercurial, svn, cvs, bitkeeper, etc). Indicar los siguientes ítems:
 1. Tipos de versionado soportados.
 2. Licencia
 3. Costo (gratuito / propietario)
 4. Quien lo mantiene.
 5. Plataformas soportadas (Windows, Unix, etc)
 6. Extras
 1. Elaborar un cuadro comparativo que resuma los puntos antes mencionados
 2. Realizar el mismo cuadro con plataformas comerciales de sistemas de control de versiones (entre 5 y 8, por ejemplo Atlassian, Github, etc) agregando la columna sistemas de control de versiones que soporta mencionadas en el punto anterior. Además mencionar que herramientas adicionales incluyen (por ejemplo wiki, herramientas de gestión de proyectos, etc).

ACTIVIDAD 1

Antes de empezar a describir brevemente algunos sistemas de control de versiones, vale preguntarnos, **¿que es un sistema de control de versiones?**

Un VCS nos permite guardar un registro de las modificaciones que realizamos sobre un fichero o conjunto de ficheros a lo largo del tiempo de tal manera que sea posible recuperar versiones específicas más adelante. Habitualmente se utiliza en entornos de desarrollo de software, pero puede resultar de gran utilidad para cualquier persona que necesite un control robusto sobre la tarea que está realizando.

La mayoría de la gente utilizamos algún sistema de control de versiones sin ser conscientes de ello. El ejemplo más claro es al editar cualquier tipo de fichero en servicios de almacenamiento en “la nube” como Google Drive o Dropbox.

Pero, ¿por qué es importante utilizar un sistema de control de versiones?

Un software de control de versiones es una valiosa herramienta con numerosos beneficios para un flujo de trabajo de equipos de software de colaboración. Cualquier proyecto de software que tiene más de un desarrollador manteniendo archivos de código fuente debe usar un VCS. Además, los proyectos mantenidos por una sola persona se beneficiarán enormemente de su uso. Se puede decir que no hay una razón válida para privarse del uso de un VCS en cualquier proyecto moderno de desarrollo de software.

Existen 3 tipos de VSC, los distribuidos, los centralizados y los locales.

- Sistemas de Control de Versiones Locales

Un método de control de versiones, usado por muchas personas, es copiar los archivos a otro directorio (quizás indicando la fecha y hora en que lo hicieron, si son ingeniosos). Este método es muy común porque es muy sencillo, pero también es tremendamente propenso a errores. Es fácil olvidar en qué directorio te encuentras y guardar accidentalmente en el archivo equivocado o sobrescribir archivos que no querías.

Para afrontar este problema los programadores desarrollaron hace tiempo VCS locales que contenían una simple base de datos, en la que se llevaba el registro de todos los cambios realizados a los archivos.

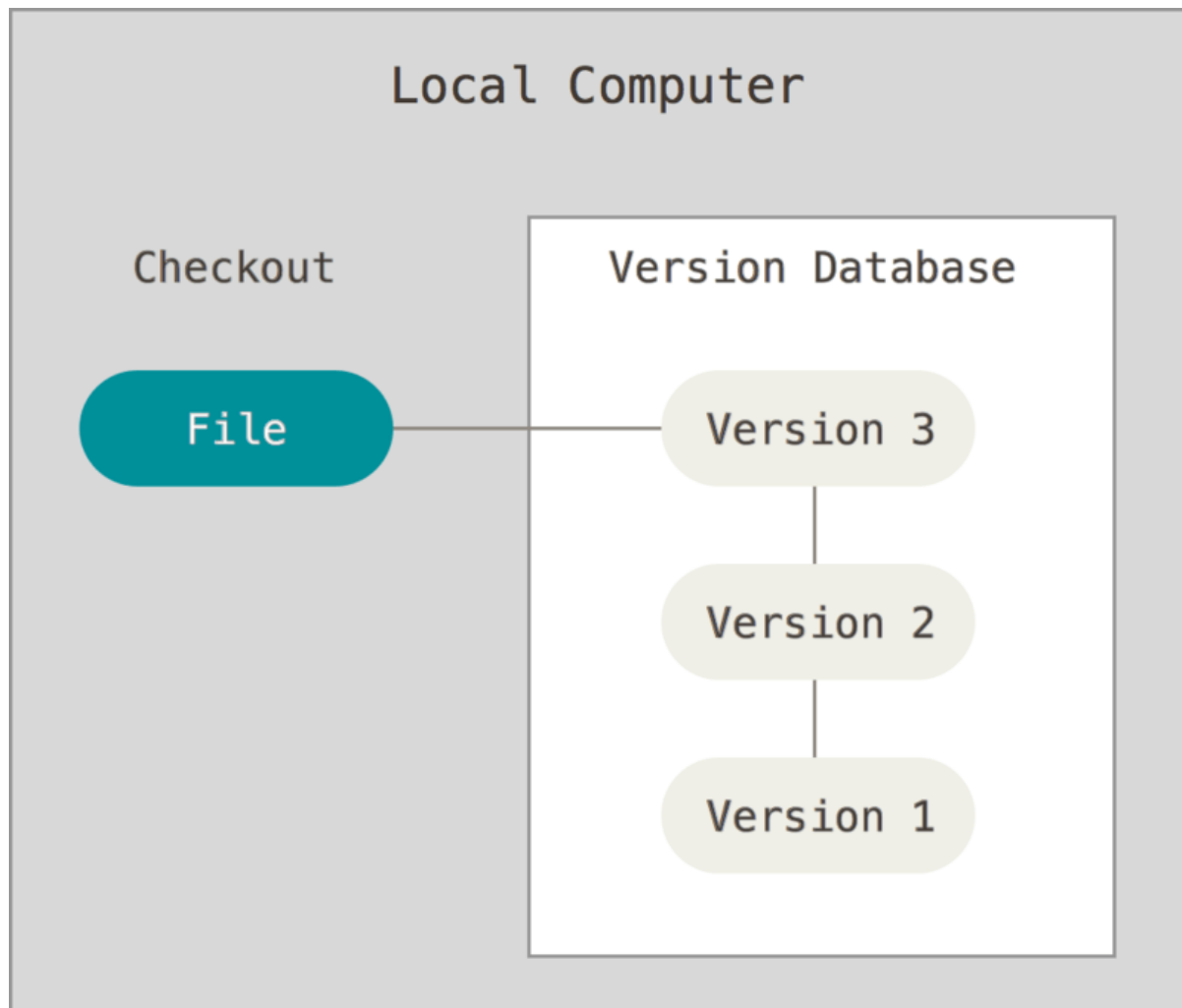


Imagen: Control de versiones local

- Sistemas de Control de Versiones Distribuidos

En un DVCS (como Git, Mercurial, Bazaar o Darcs), los clientes no solo descargan la última copia instantánea de los archivos, sino que se replica completamente el repositorio. De esta manera, si un servidor deja de funcionar y estos sistemas estaban colaborando a través de él, cualquiera de los repositorios disponibles en los clientes puede ser copiado al servidor con el fin de restaurarlo. Cada clon es realmente una copia completa de todos los datos.

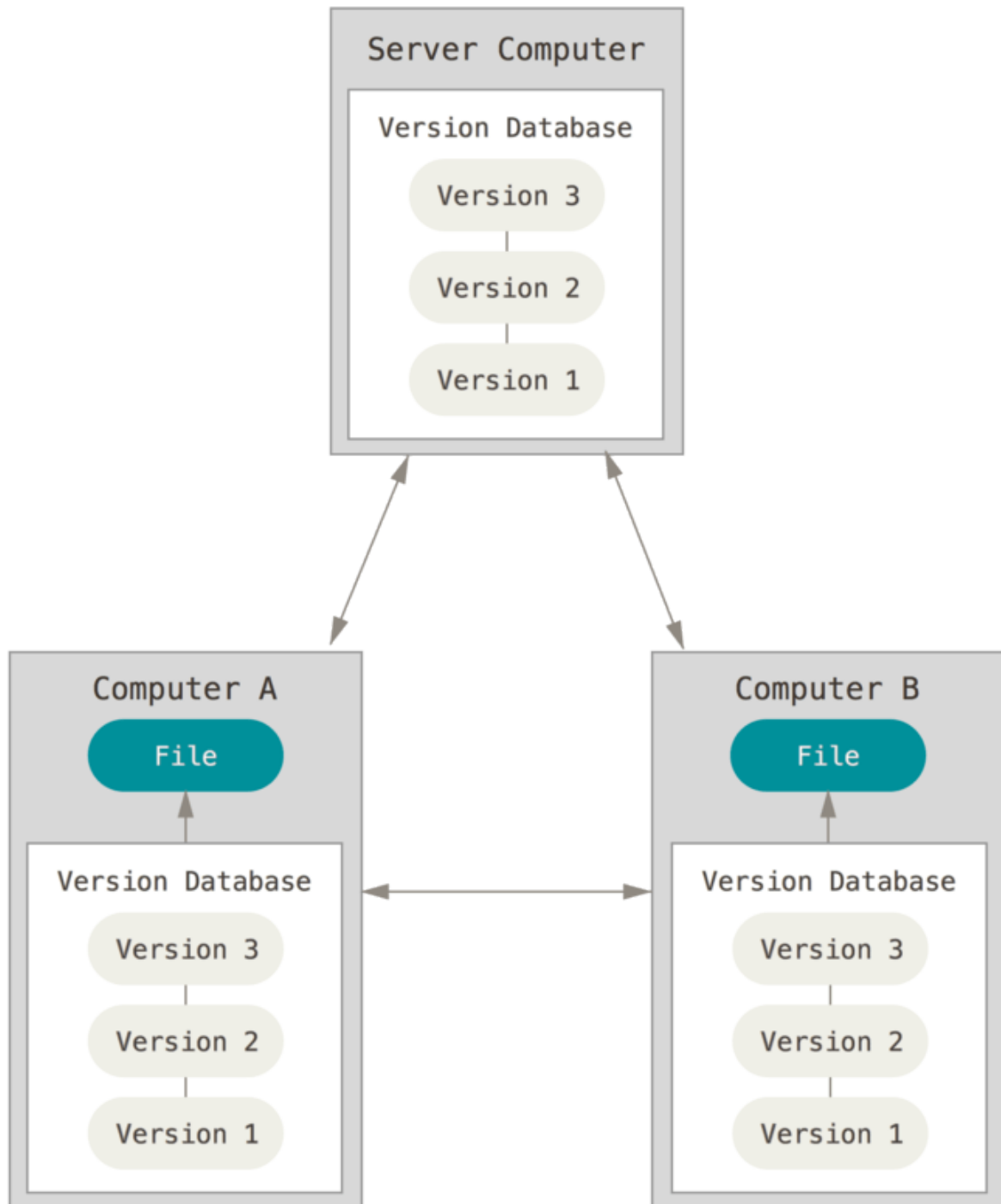


Imagen: Control de versiones distribuido

- Sistemas de Control de Versiones Centralizados

El siguiente gran problema con el que se encuentran las personas es que necesitan colaborar con desarrolladores en otros sistemas. Los sistemas de Control de Versiones Centralizados (CVCS por sus siglas en inglés) fueron desarrollados para solucionar este

problema. Estos sistemas, como CVS, Subversion y Perforce, tienen un único servidor que contiene todos los archivos versionados y varios clientes que descargan los archivos desde ese lugar central. Este ha sido el estándar para el control de versiones por muchos años.

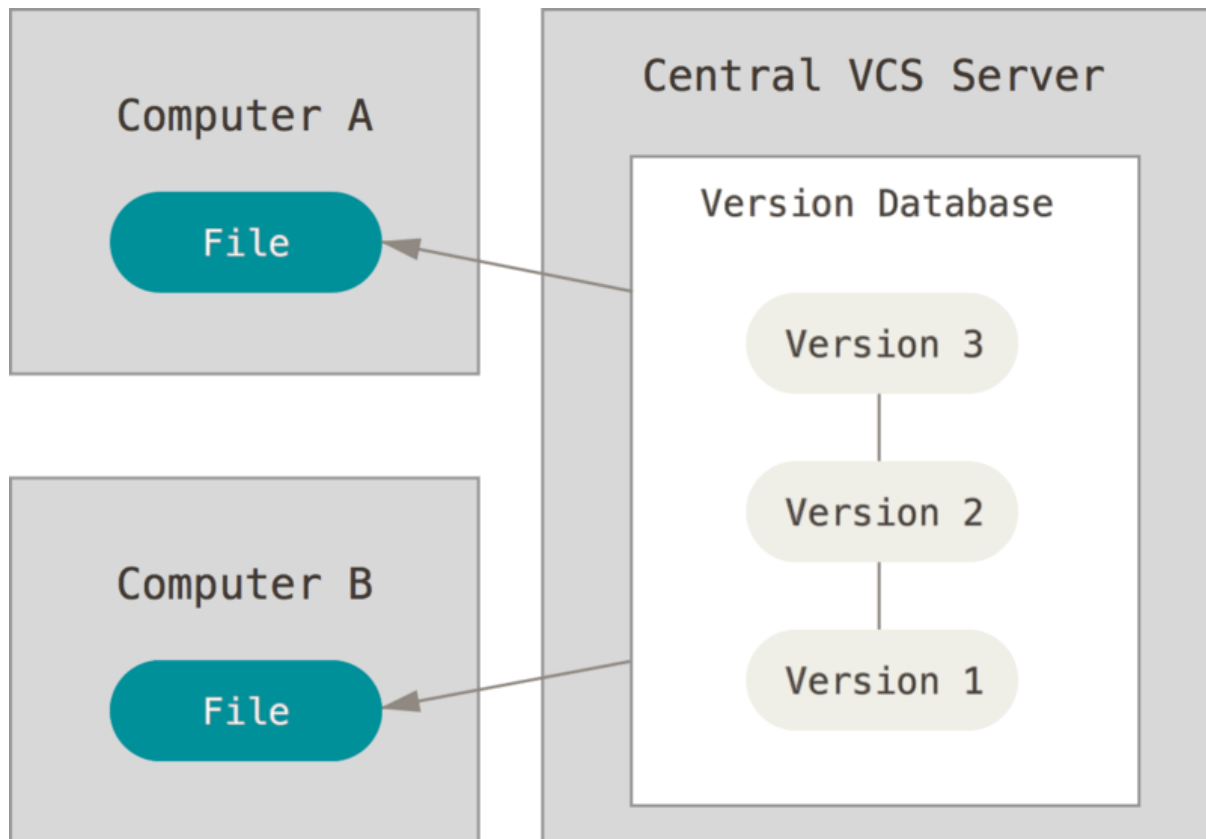


Imagen: Control de versiones centralizado

Algunos ejemplos de SCV descentralizados

- **Git**

Es una de las mejores herramientas de control de versiones disponible en el mercado actual. Es un modelo de repositorio distribuido compatible con sistemas y protocolos existentes como HTTP, FTP, SSH y es capaz de manejar eficientemente proyectos pequeños a grandes.

- 1) **Tipos de versionado soportados:** Arquitectura distribuida.
- 2) **Licencia:** Licencia Pública General de GNU (GNU GPL v2)
- 3) **Costo:** herramienta gratuita (software libre).
- 4) **Quien lo mantiene:** Junio Hamano y comunidad.
- 5) **Plataformas soportadas:** Microsoft Windows, Linux.

- **Mercurial**

Es una herramienta distribuida de control de versiones que está escrita en Python, C y Rust, y destinada a desarrolladores de software. Los sistemas operativos que admite son similares a Unix, Windows y macOS. Tiene un alto rendimiento y escalabilidad con capacidades avanzadas de ramificación y fusión y un desarrollo colaborativo totalmente distribuido. Además, posee una interfaz web integrada.

- 1) **Tipos de versionado soportados:** Arquitectura distribuida.
- 2) **Licencia:** GNU GPL versión 2.
- 3) **Costo:** Es una herramienta gratuita.
- 4) **Quien lo mantiene:** Mantenido por la comunidad.
- 5) **Plataformas soportadas:** GNU/linux, Windows, Mac OS X y la mayoría de otros sistemas tipo Unix.

- **Bitkeeper**

BitKeeper es un sistemas de control de versiones distribuido para el código fuente de los programas producidos a partir de BitMover Inc.

- 1) **Tipos de versionado soportados:** Arquitectura distribuida.
- 2) **Licencia:** Apache 2 a partir de la versión 7.2ce.
- 3) **Costo:** Open source gratuito.
- 4) **Quien lo mantiene:** BitMover Inc.
- 5) **Plataformas soportadas:** Unix y Windows.

Algunos ejemplos de SCV centralizados

- **CVS: Concurrent Versions System**

Concurrent Versions System o simplemente CVS, es una aplicación informática que implementa un sistema de control de versiones: mantiene el registro de todo el trabajo y los cambios en los ficheros (código fuente principalmente, en un único archivo para cada fichero correspondiente), que forman un proyecto de programa y permite que distintos desarrolladores, potencialmente situados a gran distancia, colaboren.

- 1) **Tipos de versionado soportados:** Arquitectura centralizada.
- 2) **Licencia:** GNU General Public License, versión 1.0 o posterior.
- 3) **Costo:** Es un software de uso libre.
- 4) **Quien lo mantiene:** The CVS Team
- 5) **Plataformas soportadas:** Unix y Windows.

- **Subversion (SVN)**

Apache Subversion (abreviado frecuentemente como SVN, por el comando svn) es una herramienta de control de versiones open source programada en C, basada en un repositorio cuyo funcionamiento se asemeja enormemente al de un sistema de ficheros.

Utiliza el concepto de revisión para guardar los cambios producidos en el repositorio. Entre dos revisiones solo guarda el conjunto de modificaciones, optimizando así al máximo el uso de espacio en disco. SVN permite al usuario crear, copiar y borrar carpetas con la misma flexibilidad con la que lo haría si estuviese en su disco duro local. Dada su flexibilidad, es necesaria la aplicación de buenas prácticas para llevar a cabo una correcta gestión de las versiones del software generado.

- 1) **Tipos de versionado soportados:** Arquitectura centralizada.
- 2) **Licencia:** Apache/BSD.
- 3) **Costo:** Es un software de uso libre.
- 4) **Quien lo mantiene:** Apache Software Foundation
- 5) **Plataformas soportadas:** Windows, Mac OS X, Linux, etc (multiplataformas).

- **Perforce**

Sistema de control de versiones comercial, propietario y centralizado desarrollado por Perforce Software, Inc.

- 1) **Tipos de versionado soportados:** Arquitectura centralizada.
- 2) **Licencia:** Propietaria.
- 3) **Costo:** Propietario.
- 4) **Quien lo mantiene:** Perforce Software, Inc.
- 5) **Plataformas soportadas:** Windows, Mac OS X, Linux, Solaris, FreeBSD, etc.

EXTRAS

Cuadro comparativo, VCS centralizados y descentralizados

Tipo de versionado	DESCENTRALIZADOS			CENTRALIZADOS		
VSC	GIT	MERCURIAL	BITKEEPER	CVS	SUBVERSION	PERFORCE
Licencia	GNU GPL v2	GNU GPL v2	Apache 2	GNU General Public License, versión 1.0 o posterior	Apache/BSD	Propietaria
Costo	Gratuito	Gratuito	Gratuito	Gratuito	Gratuito	Propietario
Mantenimiento	Junio Hamano y comunidad	Mantenido por la comunidad	BitMover Inc.	The CVS Team	Apache Software Foundation	Perforce Software, Inc.
Plataformas soportadas	Microsoft Windows, Linux	GNU/Linux, Windows, Mac OS X y la mayoría de otros sistemas tipo Unix	Linux, Windows, OS X, BSD, Solaris	Unix y Windows	Windows, Mac OS X, Linux, etc (multiplataformas)	Windows, Mac OS X, Linux, Solaris, FreeBSD, etc.

Cuadro comparativo, plataformas comerciales de SCV

Tipo de versionado	DESCENTRALIZADOS			CENTRALIZADOS		
Plataformas	GitHub	SourceForge	BitBucket	FusionForge	CloudForge	Assembla
Licencia	Licencia MIT	GPL	Comercial	GNU General Public License, versión 1.0 o posterior	Propietario	Creative Commons
Costo	Open Source - con opciones de licencias	Open Source	Gratuito	Gratuito	Con Suscripción	Propietario
Mantenimiento	Microsoft	Geeknet	Atlassian	The CVS Team	CollabNet	Perforce Software, Inc.
Plataformas soportadas	Microsoft Windows, Linux	Linux, Windows, OS X, BSD, Solaris	Linux, Windows, MacOS	Unix y Windows	Windows, Mac OS X, Linux, etc (multiplataformas)	Windows, Mac OS X, Linux, Solaris, FreeBSD, etc.
VSC soportadas	GIT	CVS, SVN, Git, Mercurial	Mercurial y Git	CVS	Git, SVN	Perforce, Git, SVN

ACTIVIDAD 2:

Análisis y utilización de un Sistema de Control de Versiones Centralizado

- Investigar un SCV Centralizado o Descentralizado (distinto de git) y explicar las principales características brevemente.
- Enumerar ventajas y desventajas, y comparación con SCV Descentralizados (cuadro comparativo).
- Seleccionar un servidor que se encuentre en la nube/web gratuito para realizar un ejemplo.
- Realizar un ejemplo iniciar el repositorio, clonarlo, modificarlo y generar conflictos, crear ramas y realizar merge de las mismas con el trunk principal, en un pequeño equipo por lo menos 3 miembros del grupo.
- Utilizar de ser necesario una herramienta cliente (gráfico o consola) o IDE.
- Documentar el ejemplo con capturas de commits de los miembros del equipo sobre un mismo archivo y otro ejemplo de branch y merge.

Mercurial

Mercurial es un sistema de control de versiones multiplataforma, para desarrolladores de software. Está implementado principalmente haciendo uso del lenguaje de programación Python, pero incluye una implementación binaria de diff escrita en C. Rust también se utiliza para mejorar el rendimiento. Mercurial fue escrito originalmente para funcionar sobre GNU/Linux. Ha sido adaptado para Windows, Mac OS X y la mayoría de otros sistemas tipo Unix. Mercurial es, sobre todo, un programa para la línea de comandos. Todas las operaciones de Mercurial se invocan como opciones dadas a su programa motor, hg.

Arquitectura distribuida

Mercurial da a cada desarrollador una copia local de todo el historial de desarrollo. De esta forma funciona independientemente del acceso a la red o de un servidor central. La confirmación, el branching y el merging son rápidos y económicos.

Rápido

Las implementaciones y estructuras de datos de Mercurial están diseñadas para ser rápidas. Puedes generar diffs entre revisiones o volver atrás en el tiempo en cuestión de segundos. Por esto, Mercurial es idóneo para proyectos grandes.

Platform independent

Mercurial se escribió pensando en la independencia de la plataforma. Como resultado, las versiones binarias están disponibles en todas las plataformas principales.

Extensible

La funcionalidad de Mercurial puede ampliarse con extensiones, ya sea activando las oficiales, descargando algunas desde la wiki, o escribiendo las tuyas.

Fácil de usar

Mercurial presenta un conjunto de comandos consistente en el que la mayoría de los usuarios se sienten como en casa. Las acciones potencialmente peligrosas están disponibles a través de las extensiones que tienes que habilitar, por lo que la interfaz básica es fácil de usar, fácil de aprender y difícil de romper.

Open Source

Mercurial es software gratis, licenciado bajo los términos de GNU General Public License Version 2 or any later version.

Mercurial vs GIT

VENTAJAS

- + Git es más rápido que Mercurial para operaciones de red
- + El enfoque de GIT para las branches es más poderoso
- + Los complementos se pueden escribir en muchos idiomas.
- + Index, una potente herramienta de staging
- + Puede convertir el repositorio GIT a Mercurial y viceversa sin pérdida de datos

DESVENTAJAS

- Comandos de checkout complicados
- Git no rastrea los cambios de nombre
- Git es más lento en Windows que Mercurial
- Comandos grandes: tediosos de ingresar
- Los complementos tienen limitaciones
- Rebase puede ser destructivo, los accidentes ocurren y, cuando lo hacen, pueden afectar a todo el equipo.

Ventajas y desventajas de Mercurial

VENTAJAS

- + Más fácil de aprender que Git
- + La usabilidad de GUI es mejor que Git
- + La ayuda a través de HG Help es mejor que la ayuda de GIT
- + Ramas visualmente más fáciles de entender
- + No se puede reescribir el historial: solo leer el historial (a diferencia de GIT)

DESVENTAJAS

- Los complementos deben estar escritos en Python
- Combinar características y funcionalidad es problemático cuando se usan diferentes extensiones
- No se puede convertir un Mercurial Repository a GIT y luego volver atrás sin pérdida de datos
- El comando de revertir solo deshará el último commit, se deben usar extensiones adicionales para más.
- Sin sparse-checkouts: gran limitación para proyectos grandes

Implementación con un servidor gratuito

Primeros pasos

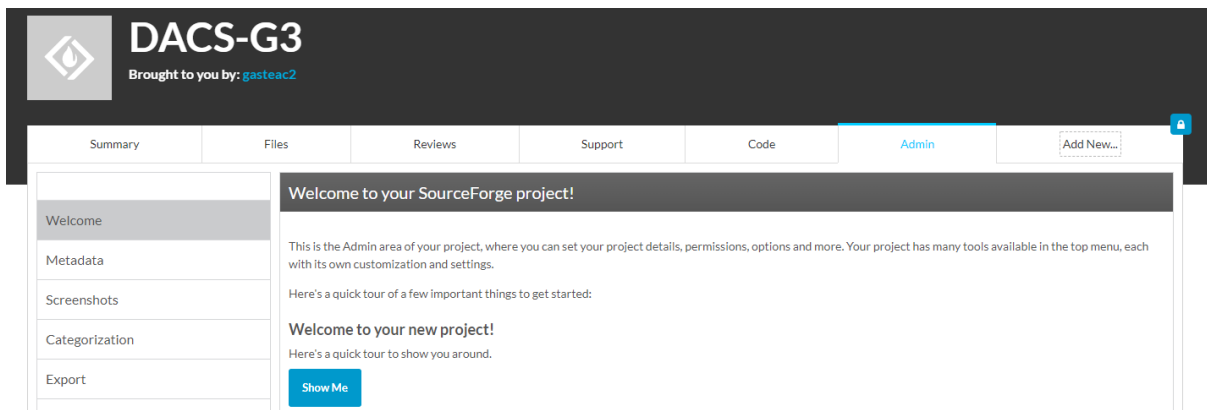
Primero, descargamos TortoiseHg de la página de Mercurial

<https://www.mercurial-scm.org>



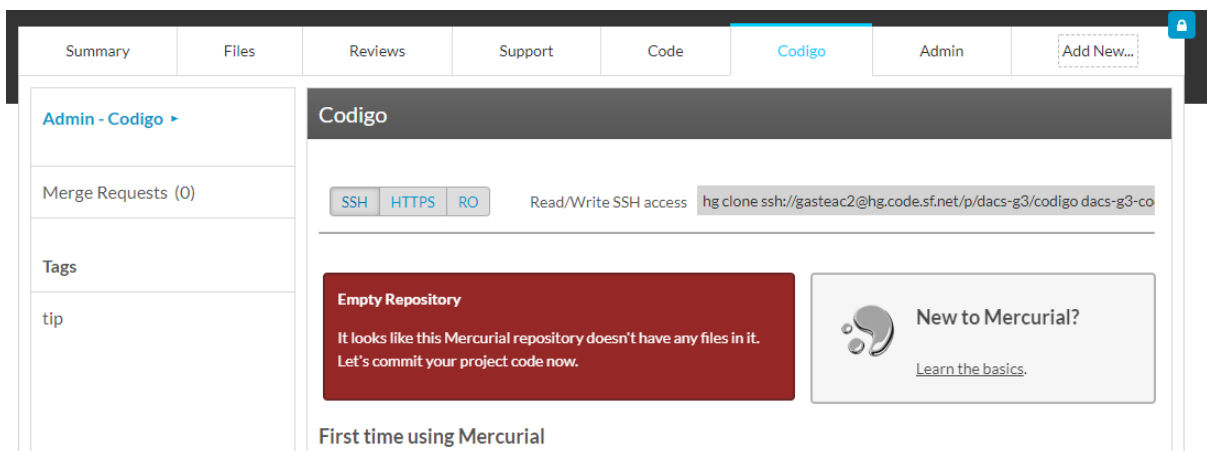
The screenshot shows the Mercurial website. At the top is a navigation bar with links: About, Guide, Download, Extensions, News/Wiki, and a search box labeled 'Search the Wiki'. Below the navigation bar, on the left, is the text 'Work easier Work faster' followed by a paragraph: 'Mercurial is a free, distributed source control management tool. It efficiently handles projects of any size and offers'. On the right, there is a blue button that says 'Download now' with a downward arrow, and below it, 'Mercurial 6.1' and 'TortoiseHg 6.1 MSI installer -'. Further down, it says 'Another OS?' and 'Get Mercurial for:' followed by links for 'Mac OS X', 'Windows', and 'other'.

Ahora creamos un proyecto nuevo en SourceForge



The screenshot shows the SourceForge project page for 'DACS-G3'. The page has a dark header with the project name and 'Brought to you by: gasteac2'. Below the header is a navigation bar with tabs: Summary, Files, Reviews, Support, Code, Admin, and a button 'Add New...'. The 'Admin' tab is selected. On the left, there is a sidebar with links: Welcome, Metadata, Screenshots, Categorization, Export, and a button 'Show Me'. The main content area has a welcome message: 'Welcome to your SourceForge project!' and 'Welcome to your new project!' with a 'Show Me' button.

Creamos un nuevo repositorio “Mercurial” y nos dirigimos a él

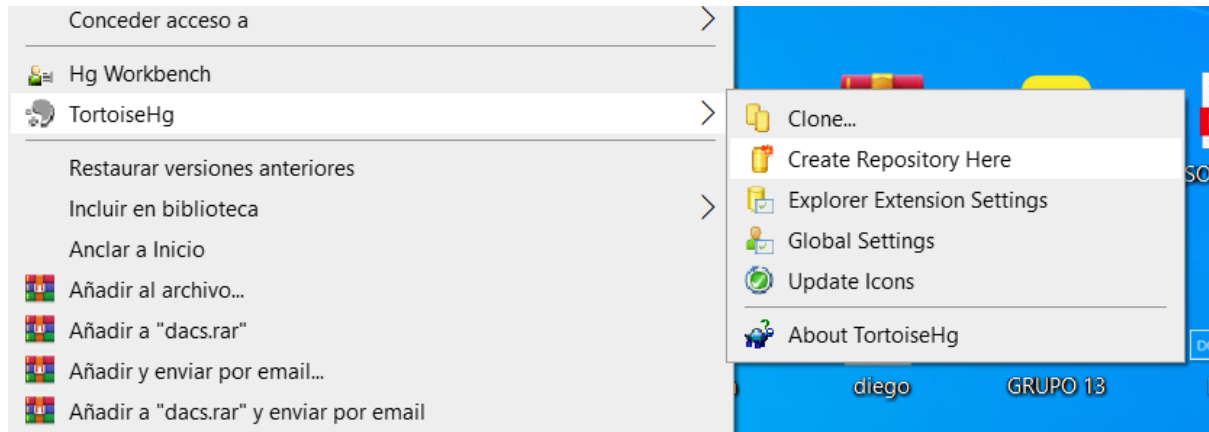


The screenshot shows the SourceForge project page for 'DACS-G3', specifically the 'Codigo' (Code) tab. The navigation bar has tabs: Summary, Files, Reviews, Support, Code, Codigo, Admin, and a button 'Add New...'. The 'Codigo' tab is selected. On the left, there is a sidebar with links: Admin - Codigo, Merge Requests (0), Tags, and a tip. The main content area has a header 'Codigo' and a section for 'SSH', 'HTTPS', and 'RO' access. It shows the command 'hg clone ssh://gasteac2@hg.code.sf.net/p/dacs-g3/codigo dacs-g3-co'. Below this, there is a red box that says 'Empty Repository' and 'It looks like this Mercurial repository doesn't have any files in it. Let's commit your project code now.' To the right of this box is a box that says 'New to Mercurial?' and 'Learn the basics.' At the bottom, it says 'First time using Mercurial'.

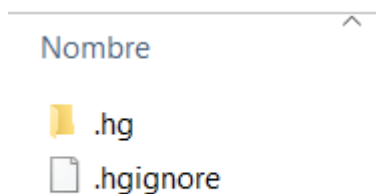
Configuramos el username en mercurial con la siguiente línea de código

hg config --edit

Creamos una carpeta que va a ser nuestro repositorio



Se nos crean los archivos necesarios de Mercurial



Vinculamos nuestro repositorio con SourceForge

First time using Mercurial

```
hg clone ssh://gasteac2@hg.code.sf.net/p/dacs-g3/g3-dacs dacs-g3-g3-dacs
cd dacs-g3-g3-dacs
touch README
hg add
hg commit -m 'Initial commit'
hg push
```

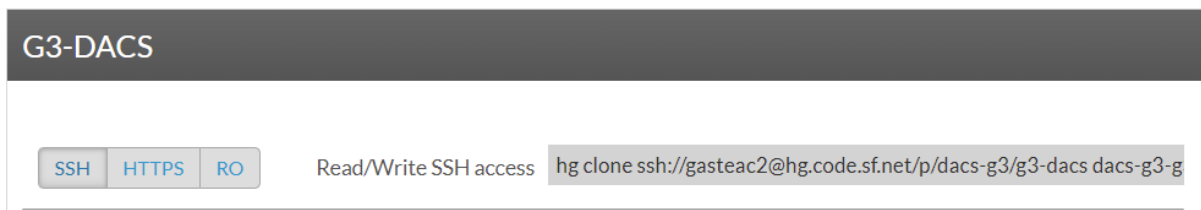
Existing Repository

Add the new repository as a remote in .hg/hgrc like this:


```
[paths]
default = ssh://gasteac2@hg.code.sf.net/p/dacs-g3/g3-dacs
```

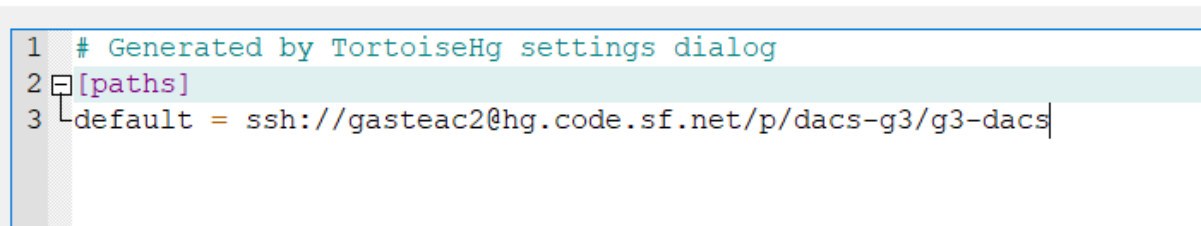
Now you can push your code to the repository.

Como ya tenemos una carpeta creada, vamos a utilizar el repositorio existente



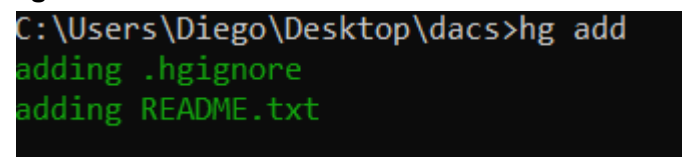
Editamos el .hg/hgrc y ponemos la URL del servidor

 C:\Users\Diego\Desktop\dacs\.hg\hgrc



Creamos archivos y los agregamos al repo con

hg add

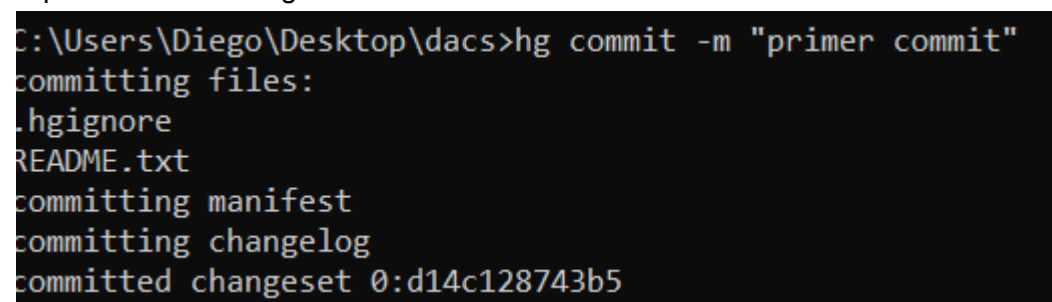


Primer commit

Procedemos a realizar un commit con tortoiseHg

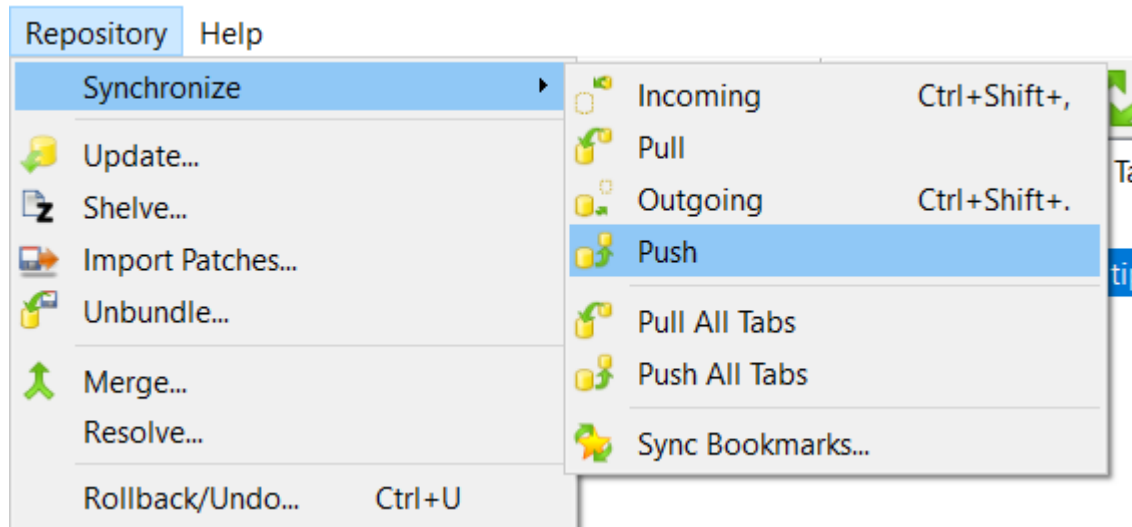


O podemos utilizar hg commit -m "comentario"



Primer push

Luego el push, podemos hacerlo con tortoiseHg





O por consola

hg push (hg push -f si queremos forzarlo)

```
C:\Users\Diego\Desktop\dacs>hg push
pushing to ssh://gasteac2@hg.code.sf.net/p/dacs-g3/g3-dacs
remote: Keyboard-interactive authentication prompts from server
remote: End of keyboard-interactive prompts from server
searching for changes
1 changesets found
uncompressed size of bundle content:
    247 (changelog)
    223 (manifests)
    121 .hgignore
    138 README.txt
remote: adding changesets
remote: adding manifests
remote: adding file changes
remote: added 1 changesets with 2 changes to 2 files
remote: <Repository /hg/p/dacs-g3/g3-dacs> refresh queued.
```

lo vemos reflejado en la página de SourceForge

File	Date
 .hgignore	4 minutes ago
 README.txt	4 minutes ago

Error común al hacer push

Ahora bien, qué pasa si un compañero sube un archivo actualizado, por ejemplo un .txt al repositorio remoto, y otro compañero que NO tiene ese último archivo actualizado, también quiere hacer un push al servidor (sin antes haber hecho un pull) ? Lo que sucede es que Mercurial nos dice que hay cosas en nuestro repositorio remoto que no están en nuestro repositorio local, o hacemos un pull antes, o descartamos cambios.

```
% hg update --config ui.merge=internal:merge --rev tip --clean
0 files updated, 0 files merged, 0 files removed, 0 files unresolved
[command completed successfully Sun Apr 17 21:58:13 2022]
% hg push ssh://gasteac2@hg.code.sf.net/p/dacs-g3/codigo
pushing to ssh://gasteac2@hg.code.sf.net/p/dacs-g3/codigo
remote: Keyboard-interactive authentication prompts from server:
remote: End of keyboard-interactive prompts from server
searching for changes
remote has heads on branch 'default' that are not known locally: 67de8ff95efd
abort: push creates new remote head bcad086f5d85
hint: pull and merge or see 'hg help push' for details about pushing new heads
[command returned code 255 Sun Apr 17 22:15:13 2022]
dacs-g3-codigo%
```

Creación de ramas




hg branch rama

```
C:\Users\Diego\Desktop\dacs>hg branch rama
marked working directory as branch rama
(branches are permanent and global, did you want a bookmark?)
```

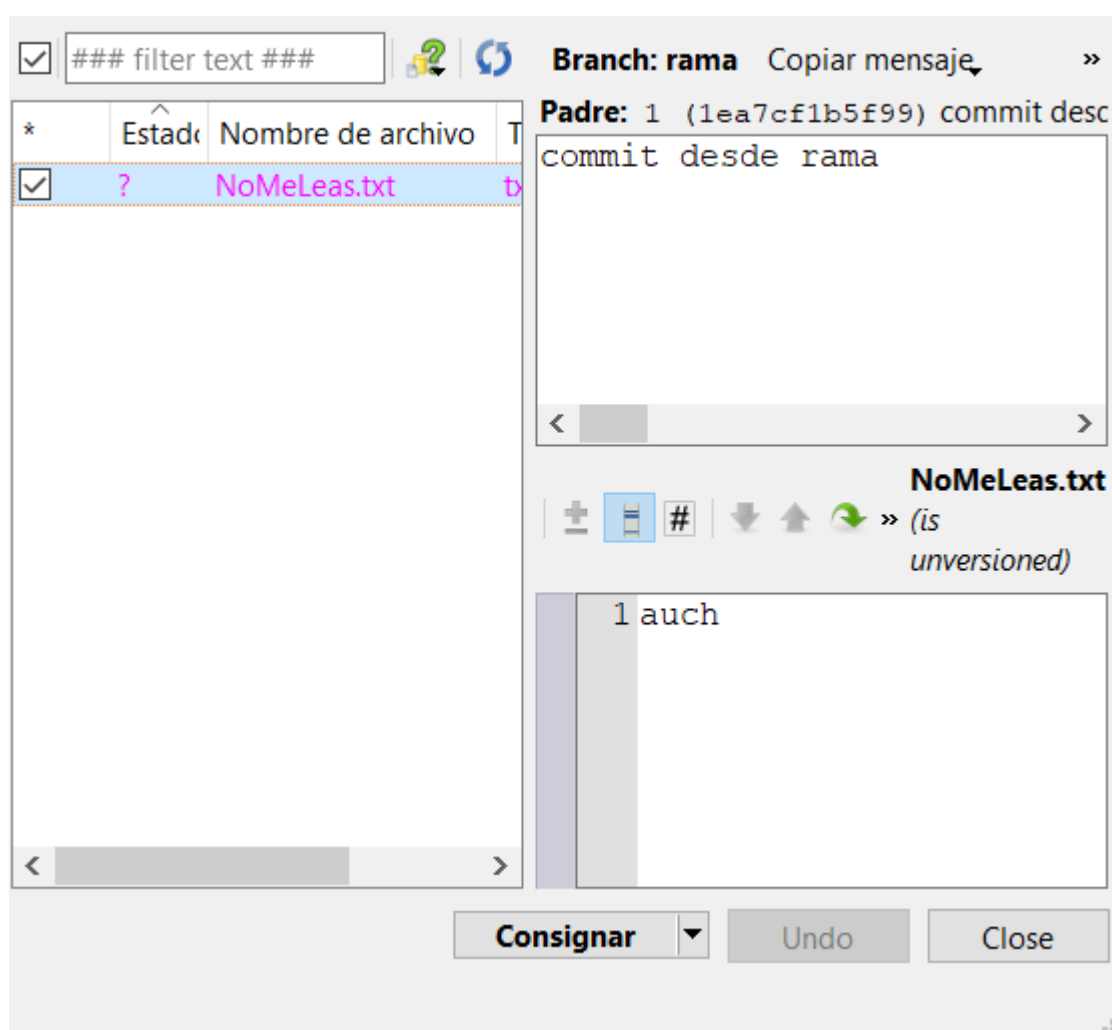
Estamos parados en esa rama nueva, pero la rama no se crea hasta que se commitea algo. Debemos cambiar algo o crear algo, luego hacer el commit.

Creamos otro .txt

hg commit -m 'algo'

Gráfico	Rev	Rama	Descripción	Autho	Age	
	1+	rama	★ Directorio de trabajo ★	Diego	ahora	
	1	rama	rama tip commit desd...	Gast...	3 se...	ti
	0	default	default primer commit	Gast...	6 mi...	

✓ dacs - commit








y luego el push de la siguiente manera, como diciéndole que pushee una nueva ramita

hg push --new-branch

Y aparece nuestra nueva rama

Branches
default
rama

Merge entre ramas

Branches	
default	 .hgignore
rama	 NoMeLeas.txt
	 README.txt
Merge Requests (0)	
Branches	
default	 .hgignore
rama	 README.txt
	Read Me

Primero nos paramos en la rama a la cual queremos traer los cambios de otra rama, con **hg update default**

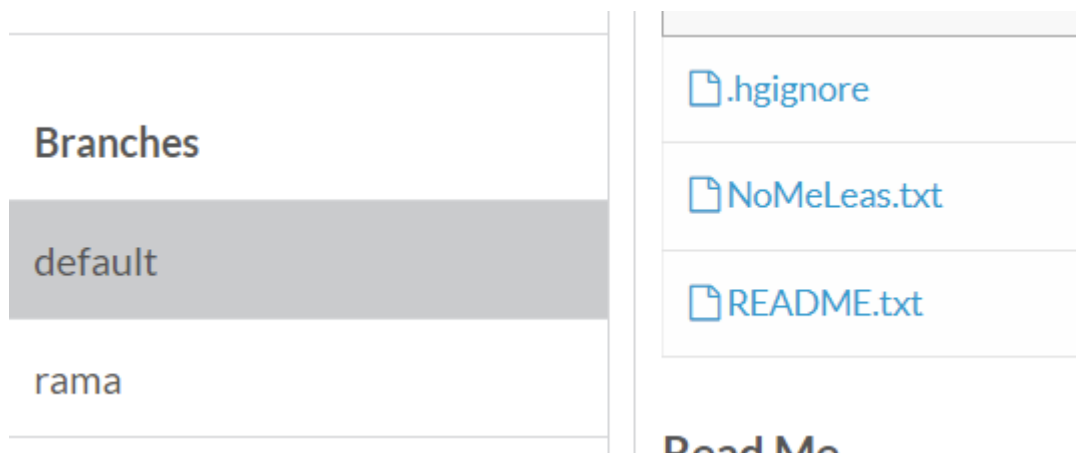
y le decimos que haga merge con cierta rama

hg merge rama

```
C:\Users\Diego\Desktop\dacs>hg update default
resolving manifests
removing NoMeLeas.txt
0 files updated, 0 files merged, 1 files removed, 0 files unresolved

C:\Users\Diego\Desktop\dacs>hg merge rama
resolving manifests
getting NoMeLeas.txt
1 files updated, 0 files merged, 0 files removed, 0 files unresolved
(branch merge, don't forget to commit)
```

Luego hacemos commit y push y vemos que se actualiza la rama default.

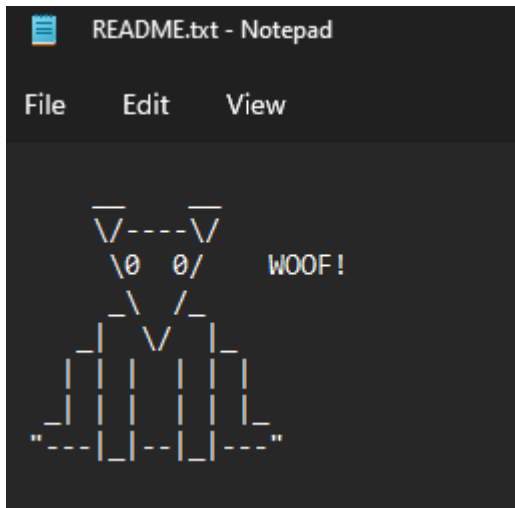


Si queremos ver los conflictos entre dos ramas se utiliza una tool de mercurial -t :merge

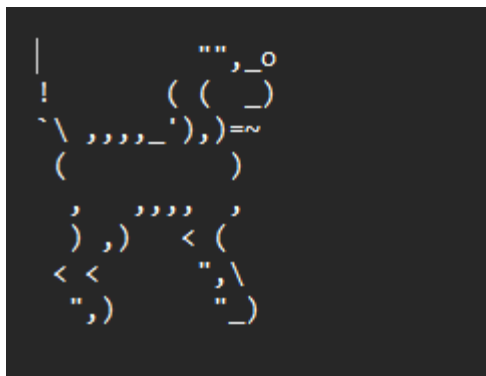
Conflictos entre ramas

En esta sección vamos a crear una rama con el comando `hg branch`, en la cual vamos a crear un `readme` diferente al de la rama original (en el mismo archivo, en las mismas líneas) entonces cuando intentemos hacer un merge de estas ramas (default y duke) nos debería mostrar el conflicto que hay en este archivo, entre esas líneas.

En la rama duke, nuestro `.txt` luce de la siguiente manera



En la rama default, nuestro archivo luce así



y si hacemos merge, con la siguiente línea de código: **`hg merge duke -t :merge`**

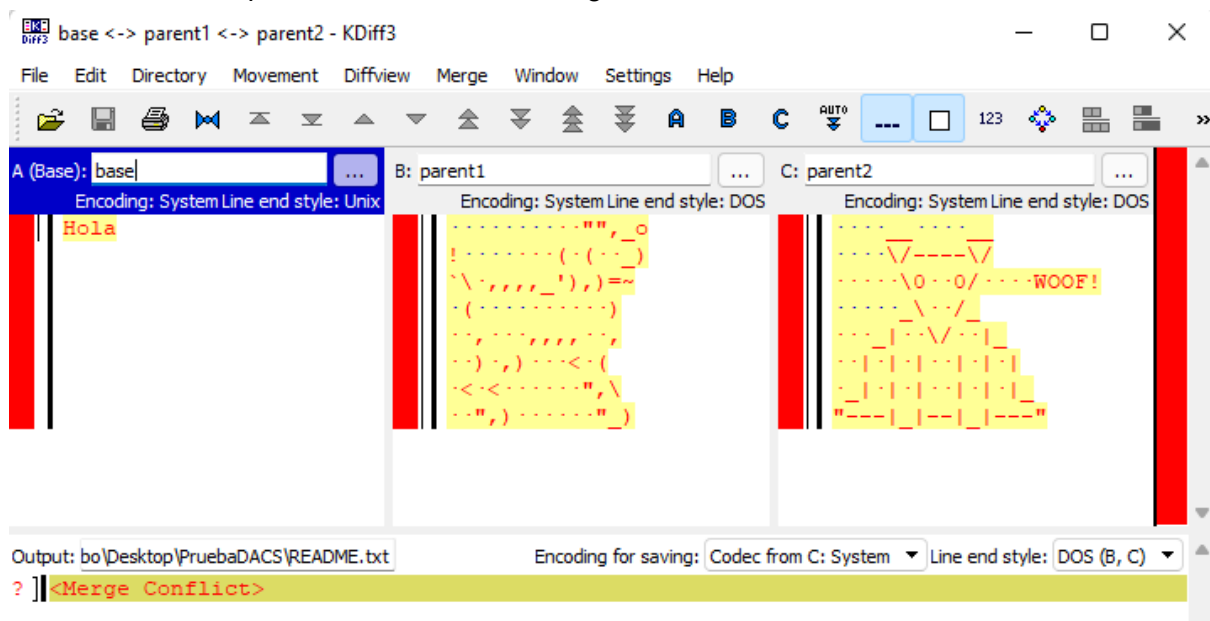
```
PS C:\Users\Lobo\Desktop\PruebaDACS> hg merge duke -t :merge
resolving manifests
merging README.txt
warning: conflicts while merging README.txt! (edit, then use 'hg resolve --mark')
0 files updated, 0 files merged, 0 files removed, 1 files unresolved
use 'hg resolve' to retry unresolved file merges or 'hg merge --abort' to abandon
PS C:\Users\Lobo\Desktop\PruebaDACS> |
```

y podemos ver los conflictos, utilizando: **hg diff**

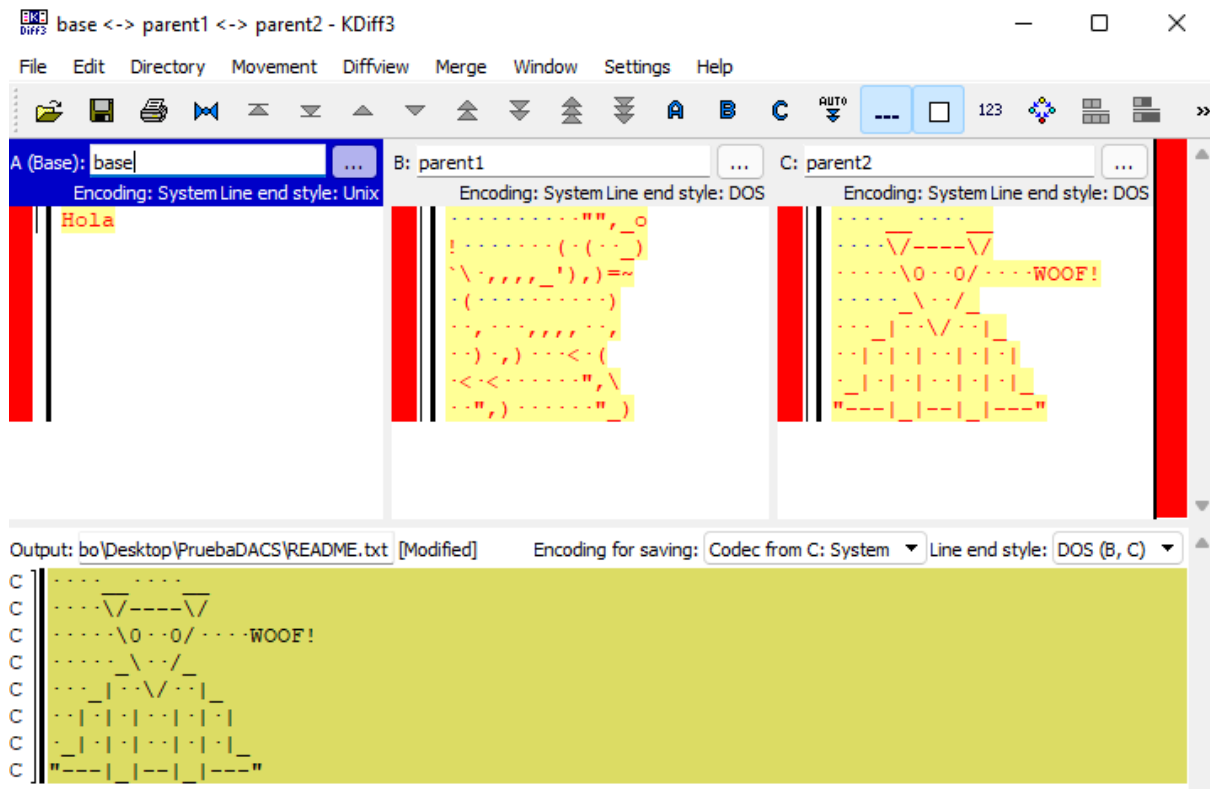
```
PS C:\Users\Lobo\Desktop\PruebaDACs> hg diff
diff -r c5de3509584b README.txt
--- a/README.txt      Fri Apr 22 11:28:40 2022 -0300
+++ b/README.txt      Fri Apr 22 11:44:16 2022 -0300
@@ -1,3 +1,4 @@
+<<<<<< working copy
+
+      "","_o
+      !      ( ( _ )
+      \ , , , , _ ' , ) = ~
@@ -5,4 +6,12 @@
+
+      ' , ) ' ' ' ' < ( '
+      < <      " , \
-      " , )      " _ )
\ No newline at end of file
+      " , )      " _ ) =====
+
+      --      --
+      \ / ---- \ /
+      \ 0   0 /      WOOF!
+      _ \   / _
+      - |   \ /   | -
+      | | | | | | |
+      _ | | | | | | _
+      + " --- | _ | -- | _ | --- " >>>>>> merge rev

PS C:\Users\Lobo\Desktop\PruebaDACs> |
```

Para solucionarlo, podemos utilizar tortoiseHg con la herramienta kdiff3



Y ahora seleccionamos la versión que queremos utilizar, en nuestro caso, la B



Seleccionamos guardar, y finalmente commiteamos la resolución de estos conflictos.

ACTIVIDAD 3

Actividad práctica sobre Git y Github

Utilizando Git por línea de comandos o desde la Web de Github (según corresponda) realizar el siguiente ejercicio (ir evidenciando documentando los pasos ver nota al final):

1. Un miembro del equipo va a clonar el siguiente [repositorio](#) y va a crear una rama para el grupo (la misma va a tener la forma GX/principal donde X es el número de grupo).

```
PS D:\Github\GIT> git clone https://github.com/FRRe-DACS/2022-TP1-GIT
Cloning into '2022-TP1-GIT'...
remote: Enumerating objects: 453, done.
remote: Counting objects: 100% (453/453), done.
remote: Compressing objects: 100% (357/357), done.
remote: Total 453 (delta 89), reused 440 (delta 84), pack-reused 0
Receiving objects: 100% (453/453), 582.10 KiB | 409.00 KiB/s, done.
Resolving deltas: 100% (89/89), done.
PS D:\Github\GIT>
```

Acá se puede observar cómo se clona el repositorio asignado en la actividad
Ya nos encontramos con el repositorio:

 2022-TP1-GIT	17/4/2022 20:06	Carpeta de archivos
--	-----------------	---------------------

Esta es la carpeta obtenida en nuestra pc.

```
PS D:\Github\GIT\2022-TP1-GIT> git checkout -b G3/Principal
Switched to a new branch 'G3/Principal'
PS D:\Github\GIT\2022-TP1-GIT> |
```

```
PS D:\Github\GIT\2022-TP1-GIT> git push -u origin G3/Principal
Total 0 (delta 0), reused 0 (delta 0), pack-reused 0
remote:
remote: Create a pull request for 'G3/Principal' on GitHub by visiting:
remote:      https://github.com/FRRe-DACS/2022-TP1-GIT/pull/new/G3/Principal
remote:
To https://github.com/FRRe-DACS/2022-TP1-GIT
 * [new branch]      G3/Principal -> G3/Principal
branch 'G3/Principal' set up to track 'origin/G3/Principal'.
PS D:\Github\GIT\2022-TP1-GIT> |
```

```

PS D:\GitHub\GIT\2022-TP1-GIT> mkdir grupo3

Directory: D:\GitHub\GIT\2022-TP1-GIT

Mode                LastWriteTime         Length Name
----                -
d----             17/4/2022   20:16             grupo3

PS D:\GitHub\GIT\2022-TP1-GIT> cd .\grupo3\

```

```

PS D:\GitHub\GIT\2022-TP1-GIT\grupo3> npm init
This utility will walk you through creating a package.json file.
It only covers the most common items, and tries to guess sensible defaults.

See 'npm help init' for definitive documentation on these fields
and exactly what they do.

Use 'npm install <pkg>' afterwards to install a package and
save it as a dependency in the package.json file.

Press ^C at any time to quit.
package name: (grupo3)
version: (1.0.0)
description: tp1-G3 proyecto de node
entry point: (index.js)
test command:
git repository:
keywords:
author: los frigolinas
license: (ISC)
About to write to D:\GitHub\GIT\2022-TP1-GIT\grupo3\package.json:

{
  "name": "grupo3",
  "version": "1.0.0",
  "description": "tp1-G3 proyecto de node",
  "main": "index.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "author": "los frigolinas",
  "license": "ISC"
}

Is this OK? (yes)
PS D:\GitHub\GIT\2022-TP1-GIT\grupo3> ls

```

Acá se puede observar la rama del grupo creada

2. En su repositorio local el usuario va a crear un va a crear una carpeta de grupo (grupoX) y dentro de la misma va a crear un proyecto en Node.js. Commitear los cambios en el repositorio y subir la rama al servidor remoto. Les dejo un link de ayuda:

- Link 1
- Link 2

```
PS D:\GitHub\GIT\2022-TP1-GIT\grupo3> git add .
warning: LF will be replaced by CRLF in grupo3/package.json.
The file will have its original line endings in your working directory
PS D:\GitHub\GIT\2022-TP1-GIT\grupo3> git commit -m 'first commit Grupo3'
[G3/Principal 715d905] first commit Grupo3
1 file changed, 11 insertions(+)
create mode 100644 grupo3/package.json
PS D:\GitHub\GIT\2022-TP1-GIT\grupo3> git push origin G3/Principal
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 8 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (4/4), 495 bytes | 495.00 KiB/s, done.
Total 4 (delta 0), reused 0 (delta 0), pack-reused 0
To https://github.com/FRRe-DACS/2022-TP1-GIT
901f51a..715d905 G3/Principal -> G3/Principal
PS D:\GitHub\GIT\2022-TP1-GIT\grupo3> |
```

Luego de crear la carpeta y el archivo de manera local, se agregó al repositorio dichos archivos.

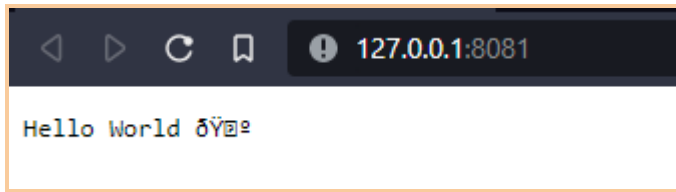
```
var http = require("http");

http
  .createServer(function (request, response) {
    response.writeHead(200, { "Content-Type": "text/plain" });
    response.end("Hello World 🐱\n");
  })
  .listen(8081);

// Console shows:
console.log("Server running at http://127.0.0.1:8081/");
```

Este es el código dentro del archivo tp1.js

```
PS D:\GitHub\GIT\2022-TP1-GIT\grupo3> node .\tp1.js
Server running at http://127.0.0.1:8081/
```



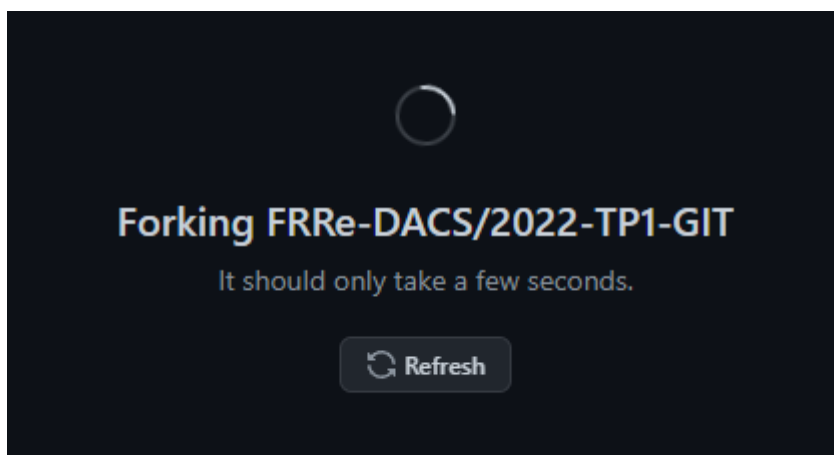
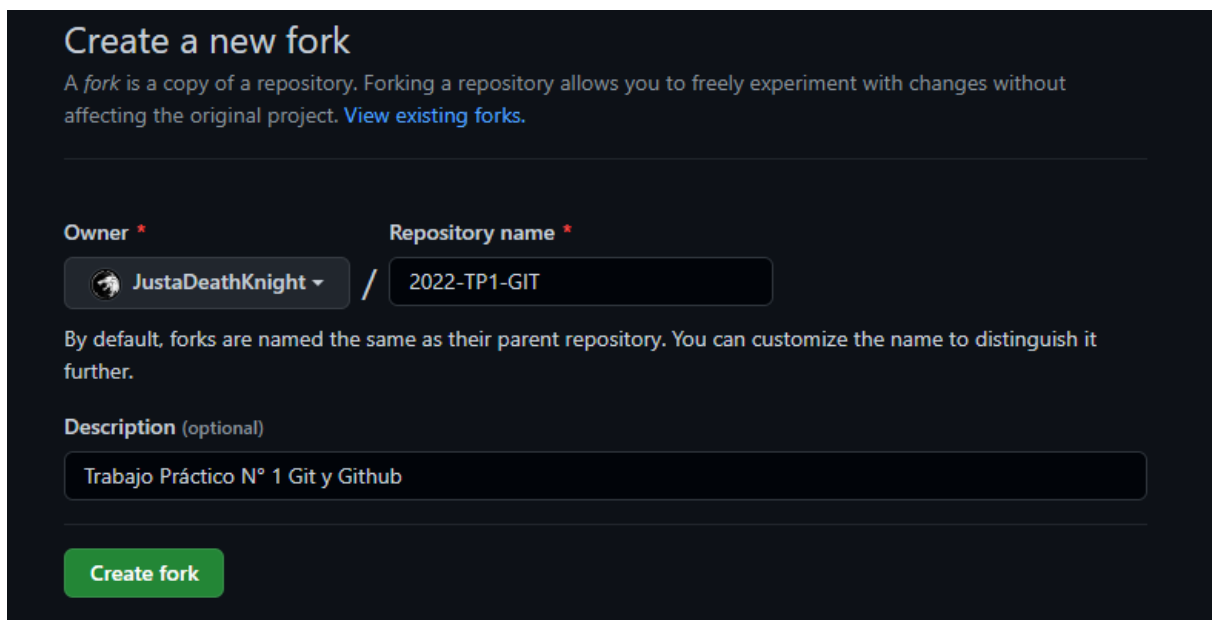
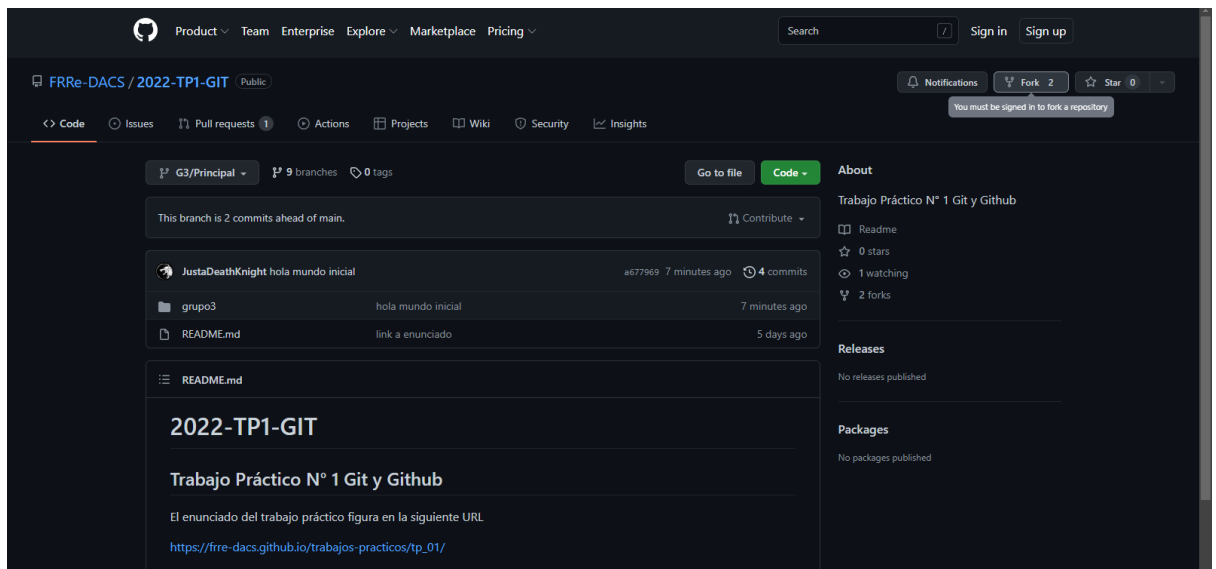
Nombre	Fecha de modificación	Tipo	Tamaño
package.json	17/4/2022 20:35	Archivo de origen ...	1 KB
package-lock.json	17/4/2022 20:32	Archivo de origen ...	1 KB
tp1.js	17/4/2022 20:26	JSFile	1 KB

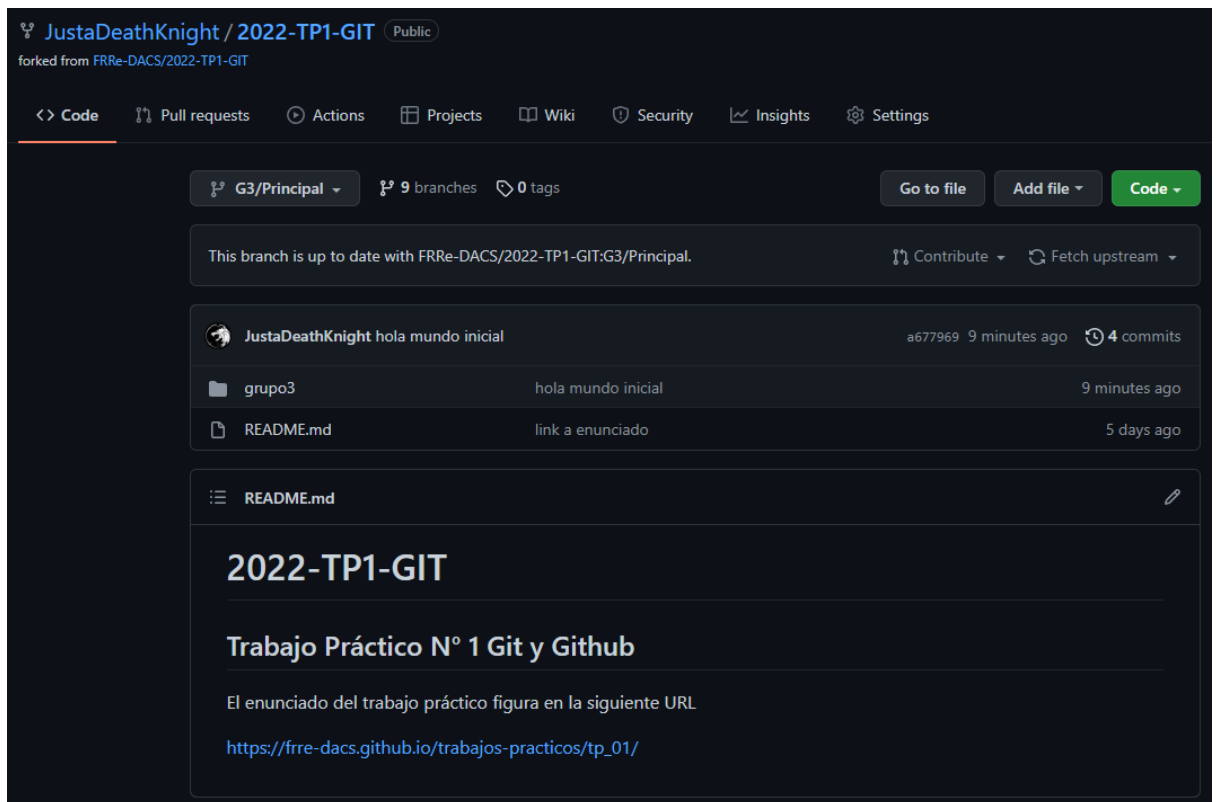
Los archivos en la pc

```
PS D:\GitHub\GIT\2022-TP1-GIT\grupo3> git add .
warning: LF will be replaced by CRLF in grupo3/package.json.
The file will have its original line endings in your working directory
warning: LF will be replaced by CRLF in grupo3/package-lock.json.
The file will have its original line endings in your working directory
PS D:\GitHub\GIT\2022-TP1-GIT\grupo3> git commit -m 'hola mundo inicial'
[G3/Principal a677969] hola mundo inicial
3 files changed, 28 insertions(+), 1 deletion(-)
create mode 100644 grupo3/package-lock.json
create mode 100644 grupo3/tp1.js
PS D:\GitHub\GIT\2022-TP1-GIT\grupo3> git push origin G3/Principal
Enumerating objects: 9, done.
Counting objects: 100% (9/9), done.
Delta compression using up to 8 threads
Compressing objects: 100% (6/6), done.
Writing objects: 100% (6/6), 790 bytes | 790.00 KiB/s, done.
Total 6 (delta 1), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
To https://github.com/FRRRe-DACS/2022-TP1-GIT
715d905..a677969 G3/Principal -> G3/Principal
PS D:\GitHub\GIT\2022-TP1-GIT\grupo3> |
```

- Una vez creada la rama del grupo en el servidor uno de los miembros del grupo va a hacer un fork de la rama. Clona el fork, va a insertar una función que imprime en un label una entrada de pantalla, commit.> push y pull request al repositorio del grupo.

Ahora se procede a demostrar el fork realizado por Ariel Acevedo.





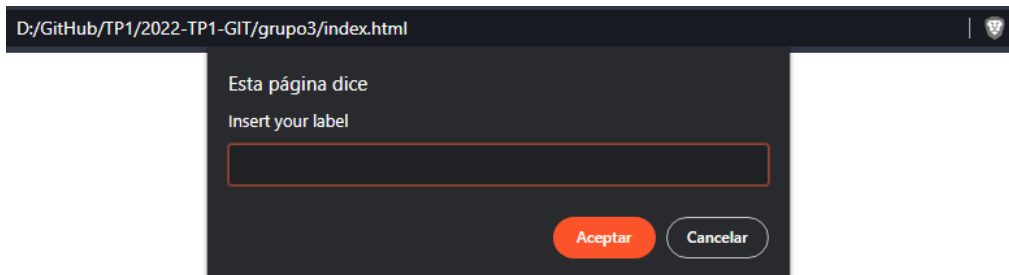
Clono el repositorio a una nueva carpeta local

```
PS D:\GitHub\TP1> git clone https://github.com/JustaDeathKnight/2022-TP1-GIT.git
Cloning into '2022-TP1-GIT'...
remote: Enumerating objects: 463, done.
remote: Counting objects: 100% (463/463), done.
remote: Compressing objects: 100% (365/365), done.
remote: Total 463 (delta 90), reused 450 (delta 85), pack-reused 0
Receiving objects: 100% (463/463), 583.27 KiB | 2.11 MiB/s, done.
Resolving deltas: 100% (90/90), done.
PS D:\GitHub\TP1> |
```

Luego me muevo a la rama del grupo

```
PS D:\GitHub\TP1\2022-TP1-GIT> git checkout G3/Principal
Switched to a new branch 'G3/Principal'
branch 'G3/Principal' set up to track 'origin/G3/Principal'.
PS D:\GitHub\TP1\2022-TP1-GIT> |
```

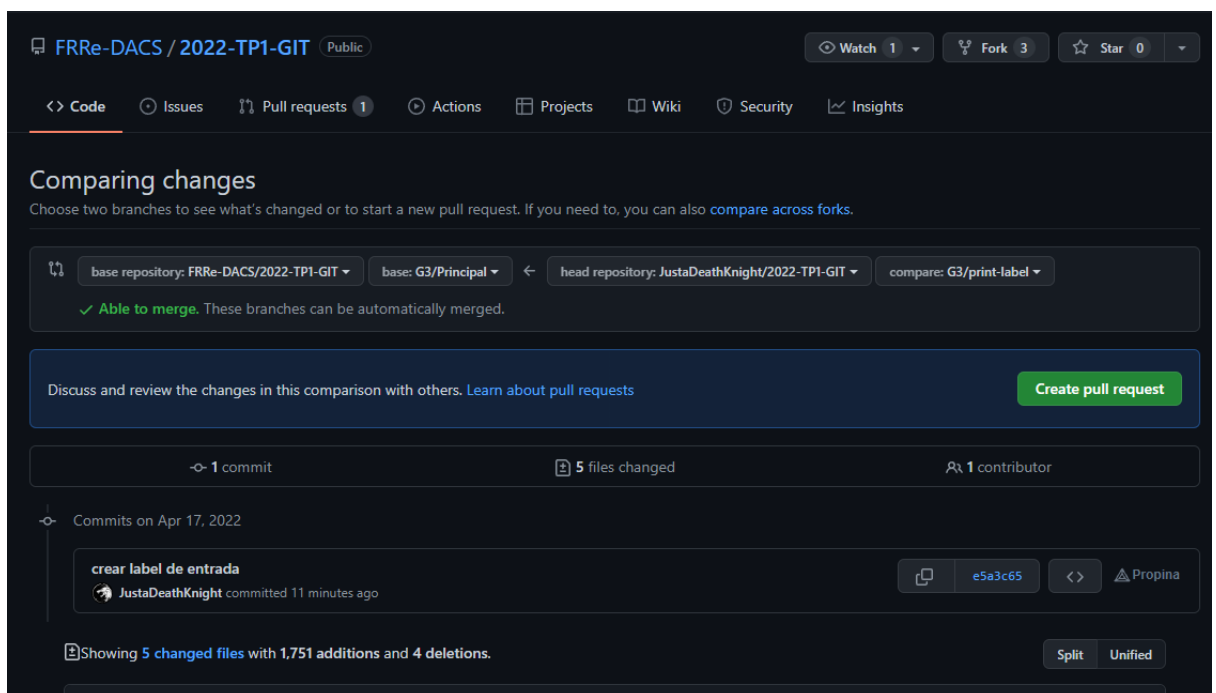
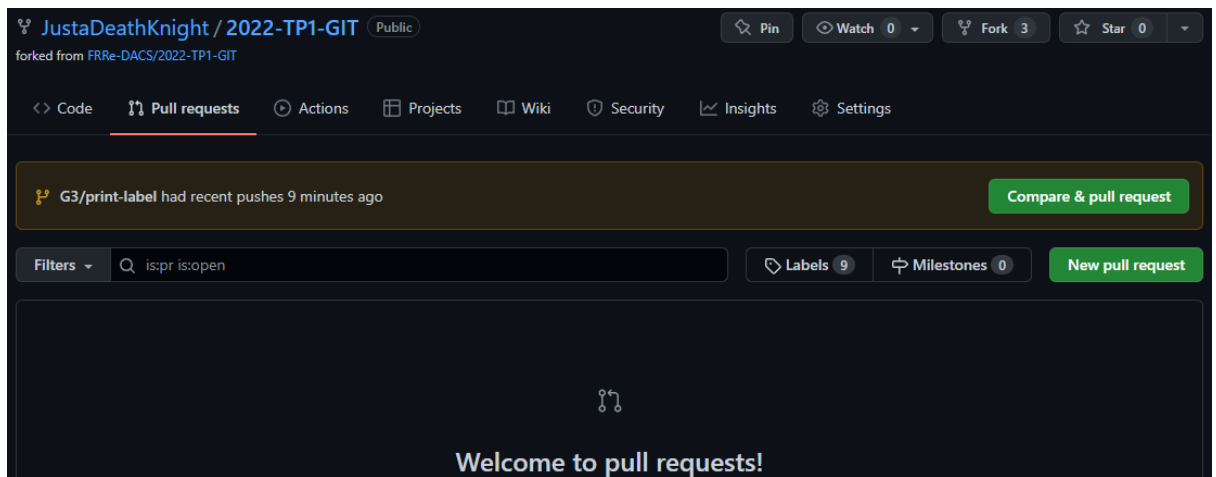
Pruebo la nueva funcionalidad

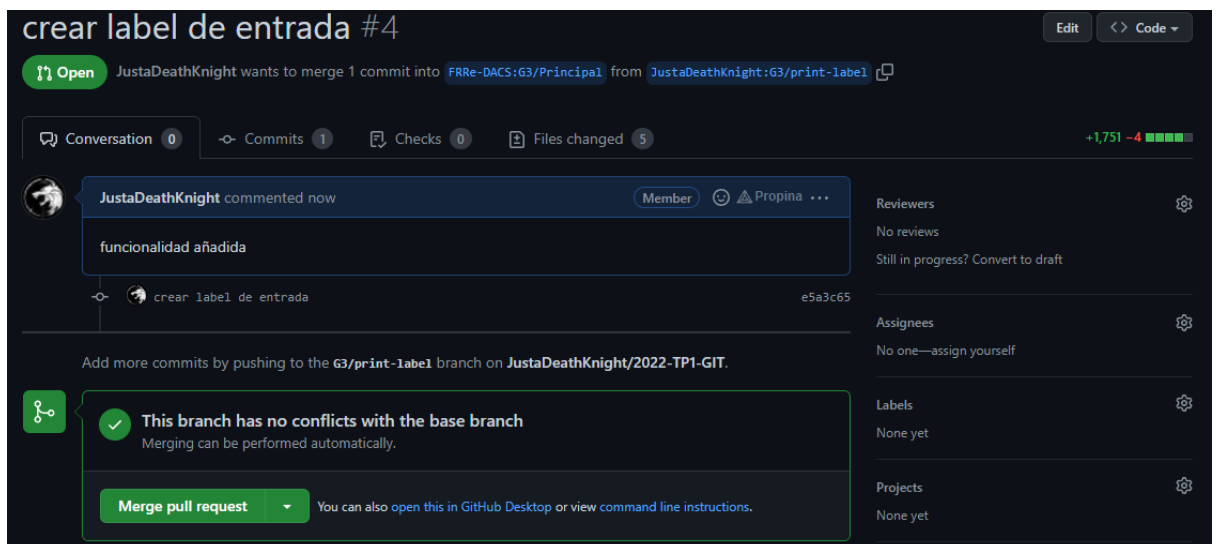
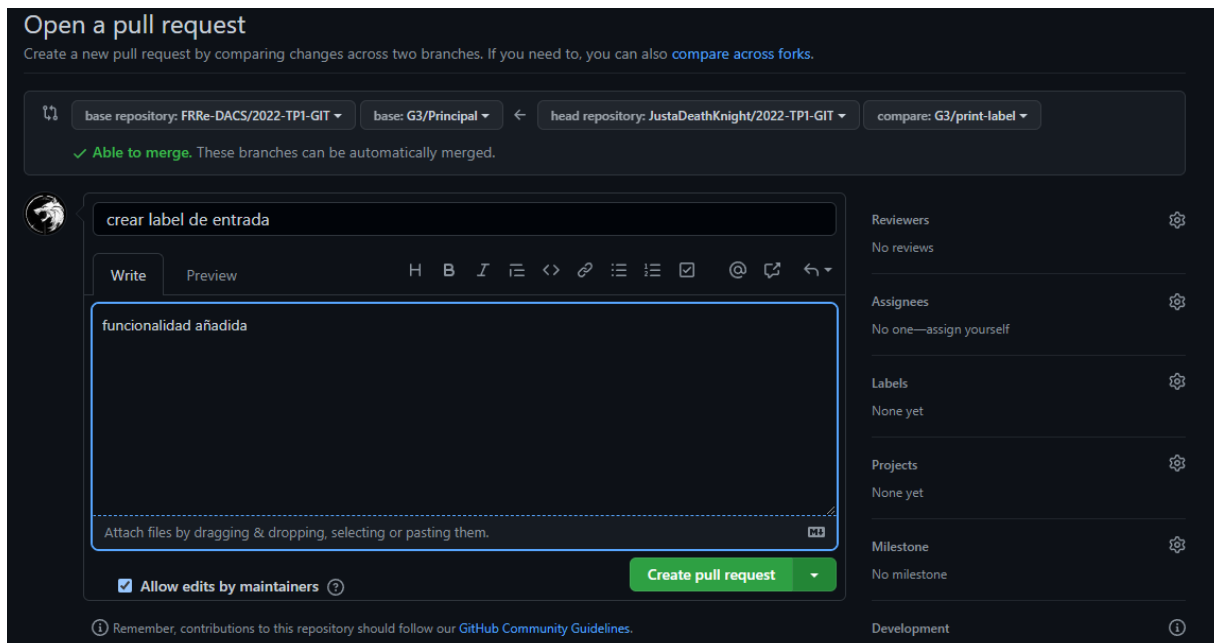


```
PS D:\GitHub\TP1\2022-TP1-GIT\grupo3> git add .
PS D:\GitHub\TP1\2022-TP1-GIT\grupo3> git status
On branch G3/print-label
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        new file:   .gitignore
        new file:   index.html
        new file:   main.js
        modified:  package-lock.json
        modified:  package.json

PS D:\GitHub\TP1\2022-TP1-GIT\grupo3> |
```

```
PS D:\GitHub\TP1\2022-TP1-GIT\grupo3> git commit -m 'crear label de entrada'
[G3/print-label e5a3c65] crear label de entrada
 5 files changed, 1751 insertions(+), 4 deletions(-)
 create mode 100644 grupo3/.gitignore
 create mode 100644 grupo3/index.html
 create mode 100644 grupo3/main.js
PS D:\GitHub\TP1\2022-TP1-GIT\grupo3> git push origin G3/print-label
Enumerating objects: 12, done.
Counting objects: 100% (12/12), done.
Delta compression using up to 8 threads
Compressing objects: 100% (7/7), done.
Writing objects: 100% (8/8), 11.97 KiB | 11.97 MiB/s, done.
Total 8 (delta 1), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
remote:
remote: Create a pull request for 'G3/print-label' on GitHub by visiting:
remote:   https://github.com/JustaDeathKnight/2022-TP1-GIT/pull/new/G3/print-label
remote:
To https://github.com/JustaDeathKnight/2022-TP1-GIT.git
 * [new branch]      G3/print-label -> G3/print-label
PS D:\GitHub\TP1\2022-TP1-GIT\grupo3> |
```





4. Los demás miembros del grupo: Clonar el repositorio y tomar la rama del grupo. A partir de la rama del grupo, crean una rama personal (gXiniciales grupo X e 2 iniciales) donde realizar una modificación en código (insertar una función que transforme el formato de un texto, que calcule una suma y la muestre en pantalla, etc) y realizar un commit y push, (Generar un conflicto y resolverlo). Ponerse de acuerdo en el grupo.

Diego Sosa

```
C:\Users\Diego\TP1-CS>git clone https://github.com/FRRe-DACS/2022-TP1-GIT
Cloning into '2022-TP1-GIT'...
remote: Enumerating objects: 472, done.
remote: Counting objects: 100% (472/472), done.
remote: Compressing objects: 100% (372/372), done.
Receiving objects: 93% (439/472) 548 (delta 86), pack-reused 0Receiving objects: 92% (435/472)
Receiving objects: 100% (472/472), 595.82 KiB | 1.83 MiB/s, done.
Resolving deltas: 100% (91/91), done.
```

```
C:\Users\Diego\TP1-CS>cd 2022-TP1-GIT

C:\Users\Diego\TP1-CS\2022-TP1-GIT>git checkout G3/Principal
Switched to a new branch 'G3/Principal'
Branch 'G3/Principal' set up to track remote branch 'G3/Principal' from 'origin'.

C:\Users\Diego\TP1-CS\2022-TP1-GIT>git checkout -b G3DS
Switched to a new branch 'G3DS'

C:\Users\Diego\TP1-CS\2022-TP1-GIT>git push origin G3DS
Total 0 (delta 0), reused 0 (delta 0), pack-reused 0
remote:
remote: Create a pull request for 'G3DS' on GitHub by visiting:
remote:   https://github.com/FRRe-DACS/2022-TP1-GIT/pull/new/G3DS
remote:
To https://github.com/FRRe-DACS/2022-TP1-GIT
 * [new branch]      G3DS -> G3DS

C:\Users\Diego\TP1-CS\2022-TP1-GIT>
```

```
const label = prompt('Insert your label')
document.getElementById('id-label').innerHTML += label ? label.toUpperCase() : 'NO LABEL'
```

El texto mostrado ahora es en formato UpperCase.

```
C:\Users\Diego\TP1-CS\2022-TP1-GIT>git add .

C:\Users\Diego\TP1-CS\2022-TP1-GIT>git commit -m "Transform label text to UpperCase"
[G3DS a06a1a1] Transform label text to UpperCase
1 file changed, 1 insertion(+), 1 deletion(-)

C:\Users\Diego\TP1-CS\2022-TP1-GIT>git push origin G3DS
Enumerating objects: 7, done.
Counting objects: 100% (7/7), done.
Delta compression using up to 8 threads
Compressing objects: 100% (4/4), done.
Writing objects: 100% (4/4), 415 bytes | 415.00 KiB/s, done.
Total 4 (delta 2), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (2/2), completed with 2 local objects.
To https://github.com/FRRe-DACS/2022-TP1-GIT
 fe94280..a06a1a1  G3DS -> G3DS

C:\Users\Diego\TP1-CS\2022-TP1-GIT>
```

Acevedo Ariel

```
PS D:\GitHub\Ariel> git clone https://github.com/FRRe-DACS/2022-TP1-GIT
Cloning into '2022-TP1-GIT'...
remote: Enumerating objects: 472, done.
remote: Counting objects: 100% (472/472), done.
remote: Compressing objects: 100% (372/372), done.
remote: Total 472 (delta 91), reused 458 (delta 86), pack-reused 0
Receiving objects: 100% (472/472), 595.82 KiB | 1.74 MiB/s, done.
Resolving deltas: 100% (91/91), done.
PS D:\GitHub\Ariel> |
```

```

PS D:\GitHub\Ariel> cd .\2022-TP1-GIT\
PS D:\GitHub\Ariel\2022-TP1-GIT> git checkout G3/Principal
Switched to a new branch 'G3/Principal'
branch 'G3/Principal' set up to track 'origin/G3/Principal'.
PS D:\GitHub\Ariel\2022-TP1-GIT> |

```

```

PS D:\GitHub\Ariel\2022-TP1-GIT> git checkout -b G3AA
Switched to a new branch 'G3AA'
PS D:\GitHub\Ariel\2022-TP1-GIT> git push origin G3AA
Total 0 (delta 0), reused 0 (delta 0), pack-reused 0
remote:
remote: Create a pull request for 'G3AA' on GitHub by visiting:
remote:      https://github.com/FRRe-DACS/2022-TP1-GIT/pull/new/G3AA
remote:
To https://github.com/FRRe-DACS/2022-TP1-GIT
 * [new branch]      G3AA -> G3AA
PS D:\GitHub\Ariel\2022-TP1-GIT> |

```

```

D: > GitHub > Ariel > 2022-TP1-GIT > grupo3 > JS main.js > ...
1  const label = prompt("Ingresa algun label");
2
3  document.getElementById("id-label").innerHTML += label
4  ? label.normalize()
5  : "No ingresaste nada bro";
6

```

```

PS D:\GitHub\Ariel\2022-TP1-GIT\grupo3> git add .
PS D:\GitHub\Ariel\2022-TP1-GIT\grupo3> git status
On branch G3AA
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        modified:   main.js

PS D:\GitHub\Ariel\2022-TP1-GIT\grupo3> git commit -m 'nomalizado de texto y cambio a textos'
[G3AA 50451a0] nomalizado de texto y cambio a textos
 1 file changed, 4 insertions(+), 2 deletions(-)
PS D:\GitHub\Ariel\2022-TP1-GIT\grupo3> |

```

```

PS D:\GitHub\Ariel\2022-TP1-GIT\grupo3> git push origin G3AA
Enumerating objects: 7, done.
Counting objects: 100% (7/7), done.
Delta compression using up to 8 threads
Compressing objects: 100% (4/4), done.
Writing objects: 100% (4/4), 478 bytes | 478.00 KiB/s, done.
Total 4 (delta 1), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
To https://github.com/FRRe-DACS/2022-TP1-GIT
    fe94280..50451a0  G3AA -> G3AA
PS D:\GitHub\Ariel\2022-TP1-GIT\grupo3> |

```

Joaquin Ramirez

```

PS C:\cliente-servidor-git> git clone https://github.com/FRRe-DACS/2022-TP1-GIT
Cloning into '2022-TP1-GIT'...
remote: Enumerating objects: 472, done.
remote: Counting objects: 100% (472/472), done.
remote: Compressing objects: 100% (372/372), done.
remote: Total 472 (delta 91), reused 458 (delta 86), pack-reused 0
Receiving objects: 92% (435/472), 595.82 KiB | 1.36 MiB/s, done.
Receiving objects: 100% (472/472), 595.82 KiB | 1.36 MiB/s, done.
Resolving deltas: 100% (91/91), done.

```

```

PS C:\cliente-servidor-git> cd 2022-TP1-GIT
PS C:\cliente-servidor-git\2022-TP1-GIT> git checkout G3/Principal
Switched to a new branch 'G3/Principal'
branch 'G3/Principal' set up to track 'origin/G3/Principal'.

PS C:\cliente-servidor-git\2022-TP1-GIT> git checkout -b G3JR
Switched to a new branch 'G3JR'
PS C:\cliente-servidor-git\2022-TP1-GIT> git push origin G3JR
Total 0 (delta 0), reused 0 (delta 0), pack-reused 0
remote:
remote: Create a pull request for 'G3JR' on GitHub by visiting:
remote:   https://github.com/FRRe-DACS/2022-TP1-GIT/pull/new/G3JR
remote:
To https://github.com/FRRe-DACS/2022-TP1-GIT
 * [new branch]      G3JR -> G3JR

```

```

1  const label = prompt('Insert your label')
2
3  document.getElementById('id-label').innerHTML += label ? label.toLowerCase() : "no label"

```

```

nothing to commit, working tree clean
PS C:\cliente-servidor-git\2022-TP1-GIT> git status
On branch G3JR
Your branch is up to date with 'origin/G3JR'.

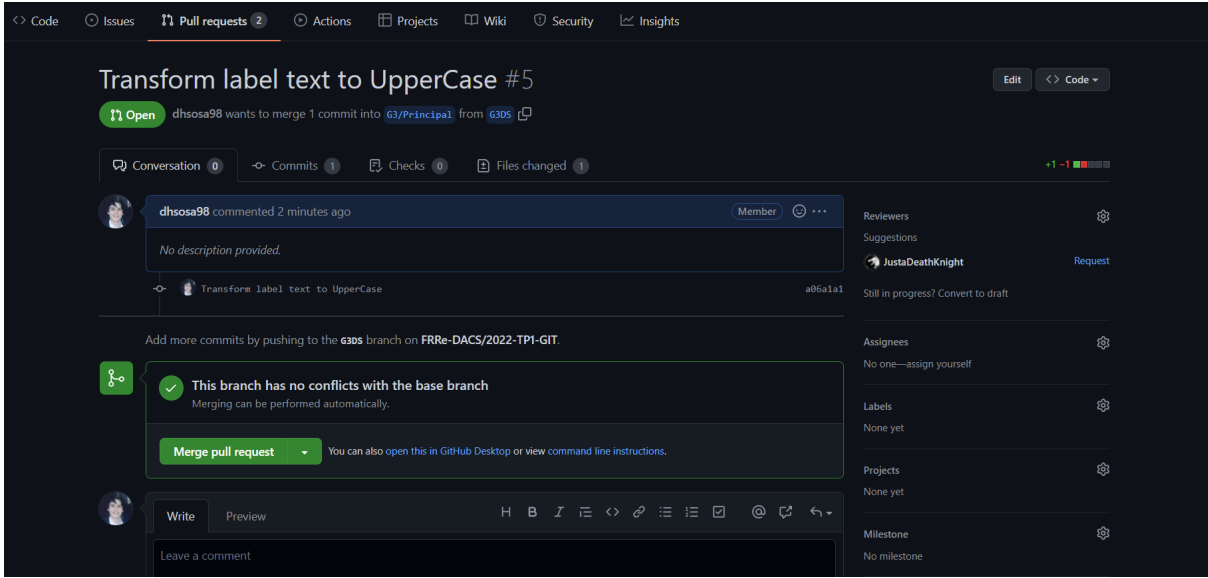
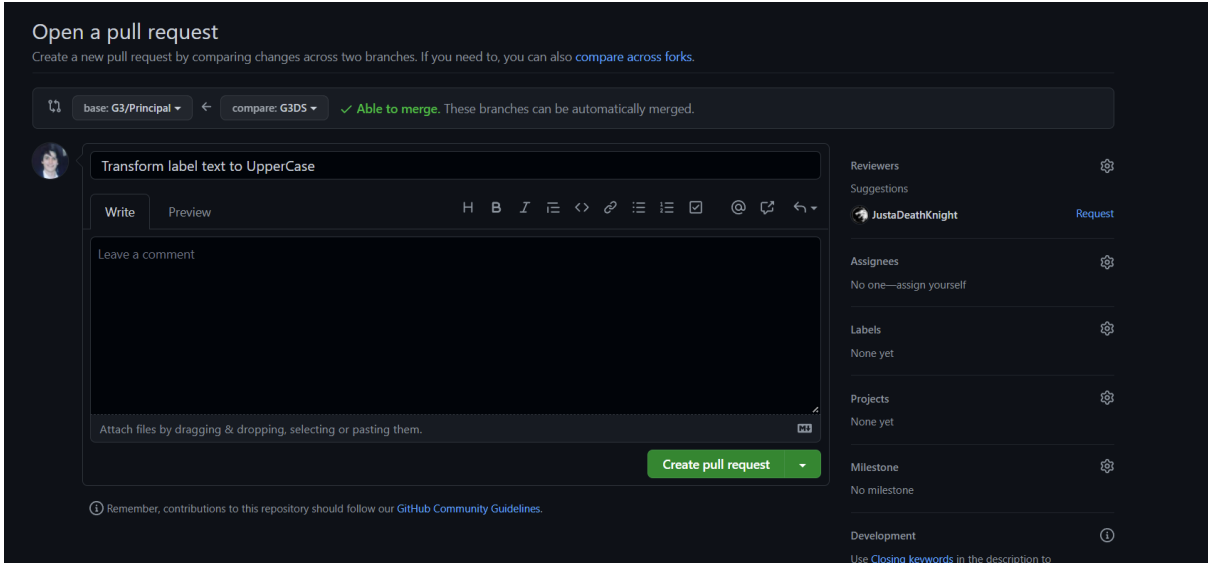
nothing to commit, working tree clean
PS C:\cliente-servidor-git\2022-TP1-GIT> git add .
PS C:\cliente-servidor-git\2022-TP1-GIT> git commit -m "label en minuscula"
[G3JR 207f4bd] label en minuscula
1 file changed, 1 insertion(+), 1 deletion(-)
PS C:\cliente-servidor-git\2022-TP1-GIT> git push origin G3JR
Enumerating objects: 7, done.
Counting objects: 100% (7/7), done.
Delta compression using up to 4 threads
Compressing objects: 100% (4/4), done.
Writing objects: 100% (4/4), 408 bytes | 204.00 KiB/s, done.
Total 4 (delta 2), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (2/2), completed with 2 local objects.
To https://github.com/FRRe-DACS/2022-TP1-GIT
fe94280..207f4bd  G3JR -> G3JR

```

5. Realizar un pull request de la rama personal a la principal de grupo.

PR Diego Sosa

The screenshot shows the GitHub 'Comparing changes' page. At the top, there are navigation tabs: Code, Issues, Pull requests (selected), Actions, Projects, Wiki, Security, and Insights. The main heading is 'Comparing changes', followed by a subtext: 'Choose two branches to see what's changed or to start a new pull request. If you need to, you can also [compare across forks](#).' Below this, there are two dropdown menus: 'base: G3/Principal' and 'compare: G3DS'. To the right of these is a green checkmark and the text 'Able to merge. These branches can be automatically merged.' Below the dropdowns is a blue button that says 'Create pull request'. Underneath the button, there is a summary bar showing '1 commit', '1 file changed', and '1 contributor'. At the bottom, there is a section for 'Commits on Apr 17, 2022' which lists a single commit: 'Transform label text to UpperCase' by user 'dhsosa98', committed 4 minutes ago. To the right of the commit title is a copy icon, the commit hash 'a06a1a1', and a code icon.



PR Joaquin Ramirez

This screenshot shows a GitHub Pull Request (PR) titled "label en minuscula #6". The PR is open and is being reviewed by JustaDeathKnight. The PR description indicates that the code has been changed. A green checkmark indicates that the branch has no conflicts with the base branch, and merging can be performed automatically. The PR is being merged into the G3JR branch on the FRRe-DACS/2022-TP1-GIT repository. The PR is created by JoaquinRamirez98, who is a member of the repository. The PR is currently in progress, and the reviewer JustaDeathKnight has requested changes. The PR is currently in progress, and the reviewer JustaDeathKnight has requested changes.

PR Acevedo Ariel

This screenshot shows a GitHub Pull Request (PR) titled "nominalizado de texto y cambio a textos". The PR is open and is being reviewed by JustaDeathKnight. The PR description indicates that the code has been changed. A green checkmark indicates that the branch has no conflicts with the base branch, and merging can be performed automatically. The PR is being merged into the G3JR branch on the FRRe-DACS/2022-TP1-GIT repository. The PR is created by JoaquinRamirez98, who is a member of the repository. The PR is currently in progress, and the reviewer JustaDeathKnight has requested changes. The PR is currently in progress, and the reviewer JustaDeathKnight has requested changes.

nomalizado de texto y cambio a textos #7

Open JustaDeathKnight wants to merge 1 commit into `63/Principal` from `63AA`

Conversation 0 Commits 1 Checks 0 Files changed 1

JustaDeathKnight commented now Member Propina ...

No description provided.

nomalizado de texto y cambio a textos 50451a0

Add more commits by pushing to the `63AA` branch on `FRRe-DACS/2022-TP1-GIT`.

This branch has no conflicts with the base branch
Merging can be performed automatically.

Merge pull request

You can also [open this in GitHub Desktop](#) or view [command line instructions](#).

label en minuscula #6

Merged JoaquinRamirez98 merged 1 commit into `63/Principal` from `63R` now

Conversation 0 Commits 1 Checks 0 Files changed 1 +1 -1

JoaquinRamirez98 commented 8 minutes ago Member Propina ...

codigo cambiado

label en minuscula 287f4bd

JoaquinRamirez98 merged commit 5275abf into `63/Principal` now Revert

Pull request successfully merged and closed
You're all set—the `63R` branch can be safely deleted. Delete branch

Write Preview H B I

Leave a comment

Reviewers
No reviews

Assignees
No one—assign yourself

Labels
None yet

Projects
None yet

Milestone
No milestone


Development
Successfully merging this pull request may close these issues.

Joaquin aceptó su PR


Transform label text to UpperCase #5

[Open](#) dhsosa98 wants to merge 1 commit into [G3/Principal](#) from [G3DS](#)



Conversation 0 Commits 1 Checks 0 Files changed 1

 dhsosa98 commented 40 minutes ago Member ⋮


No description provided.

 Transform label text to UpperCase a06a1a1

Add more commits by pushing to the [G3DS](#) branch on [FRRe-DACS/2022-TP1-GIT](#).

  **This branch has conflicts that must be resolved**
Use the [web editor](#) or the [command line](#) to resolve conflicts.
Conflicting files
grupo3/main.js

[Merge pull request](#) ⌵ You can also open this in [GitHub Desktop](#) or view [command line instructions](#).

Reviewers
Suggestions
 JustaDeathKnight [Request](#)

Still in progress? Convert to draft

Assignees
No one—assign yourself

Labels
None yet

Projects
None yet


Milestone
No milestone

nomalizado de texto y cambio a textos #7


[Open](#) JustaDeathKnight wants to merge 1 commit into [G3/Principal](#) from [G3AA](#)

Status: Open


Conversation 0 Commits 1 Checks 0 Files changed 1

 JustaDeathKnight commented 3 minutes ago Member 😊 ⚠️ Propina ⋮

No description provided.

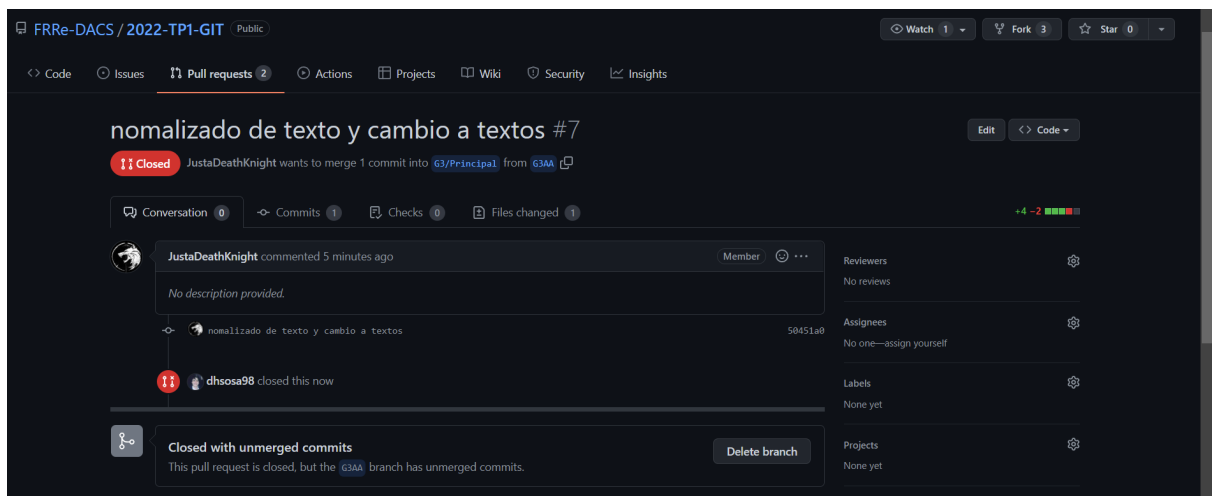
 nomalizado de texto y cambio a textos 50451a0

Add more commits by pushing to the [G3AA](#) branch on [FRRe-DACS/2022-TP1-GIT](#).

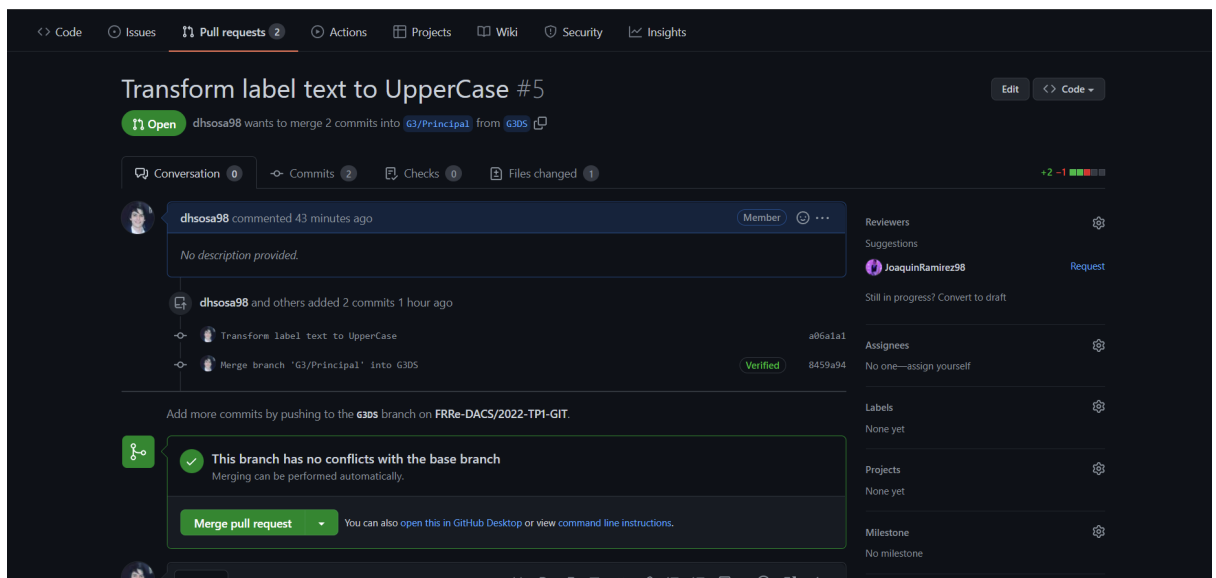
  **This branch has conflicts that must be resolved**
Use the [web editor](#) or the [command line](#) to resolve conflicts.
Conflicting files
grupo3/main.js

[Merge pull request](#) ⌵ You can also open this in [GitHub Desktop](#) or view [command line instructions](#).

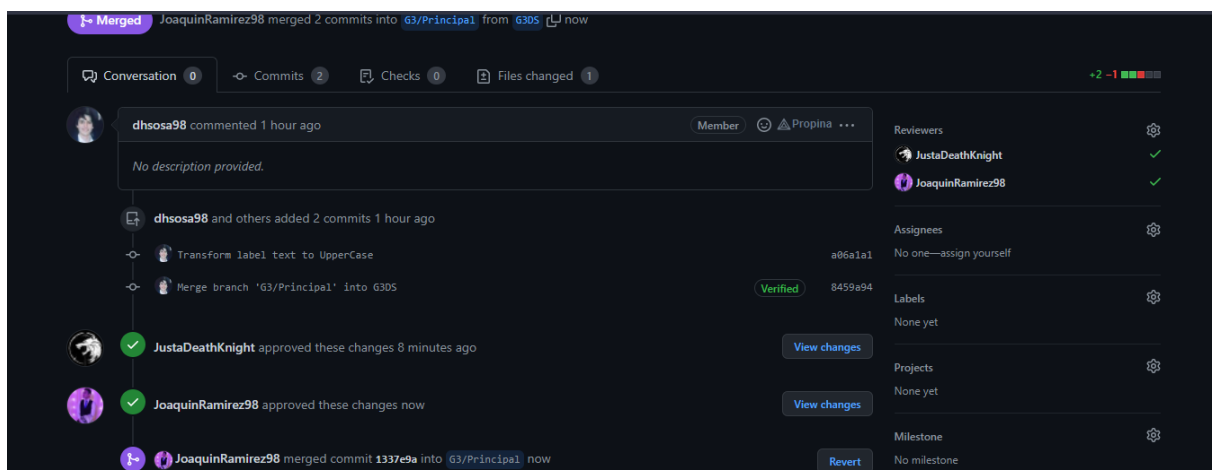
Ahora los otros PR tienen problemas



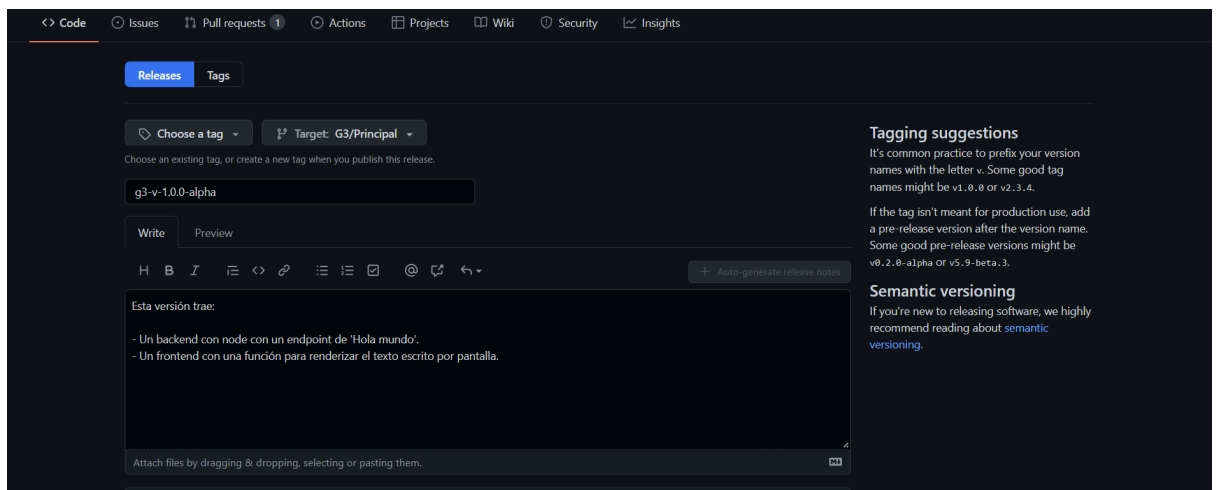
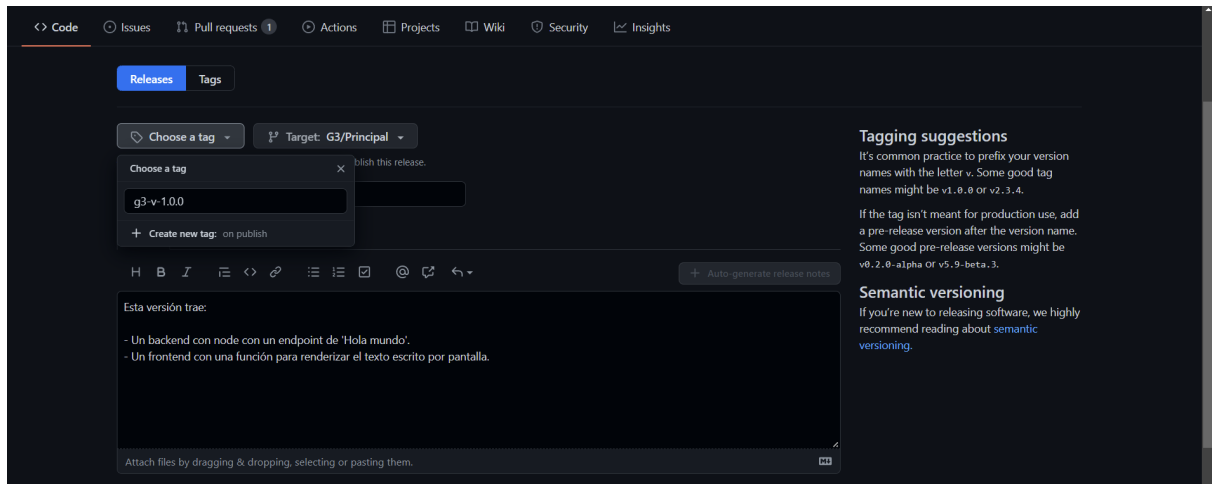
Decidimos cerrar el PR de Ariel



Por otro lado, resolvemos los conflictos del PR de Diego y procedimos a aceptarlo



6. Aceptar / confirmar los pull request en la web, obtener la funcionalidad completa del programa. Generar un tag para la versión con el nombre gX-V-1.0.0 X número de grupo (por línea de comando) y subir al repositorio remoto.



7. Realizar un cambio en el programa sobre la rama principal del grupo y subir el cambio (que introduce un error al programa).

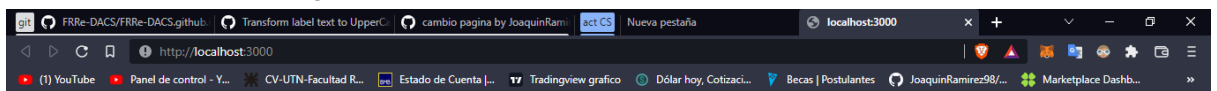
```
1  const label = prompt('Insert your label')
2
3  document.getElementById('id-label').innerHTML + label ? label.toUpperCase() : 'NO LABEL'
4
```

```

PS C:\cliente-servidor-git\2022-TP1-GIT> git checkout G3/Principal
Switched to a new branch 'G3/Principal'
branch 'G3/Principal' set up to track 'origin/G3/Principal'.
PS C:\cliente-servidor-git\2022-TP1-GIT> git pull
Already up to date.
PS C:\cliente-servidor-git\2022-TP1-GIT> git add .
PS C:\cliente-servidor-git\2022-TP1-GIT> git commit -m "cambios codigo"
PS C:\cliente-servidor-git\2022-TP1-GIT> git commit -m "cambios codigo"
[G3/Principal 9b39800] cambios codigo
1 file changed, 1 insertion(+), 1 deletion(-)
PS C:\cliente-servidor-git\2022-TP1-GIT> git push
Enumerating objects: 7, done.
Counting objects: 100% (7/7), done.
Delta compression using up to 4 threads
Compressing objects: 100% (4/4), done.
Writing objects: 100% (4/4), 374 bytes | 187.00 KiB/s, done.
Total 4 (delta 2), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (2/2), completed with 2 local objects.
To https://github.com/FRRe-DACS/2022-TP1-GIT
1337e9a..9b39800 G3/Principal -> G3/Principal

```

No devuelve nada la pagina



8. Revertir los cambios al commit del tag creado anteriormente y subir los cambios a la rama principal.

```

PS C:\Users\Diego\TP1-CS\2022-TP1-GIT> git tag
g3-v-1.0.0

```

Acá podemos observar nuestro tag creado anteriormente

```

C:\Users\Diego\TP1-CS\2022-TP1-GIT> git reset --hard g3-v-1.0.0
HEAD is now at 1337e9a Merge pull request #5 from FRRe-DACS/G3DS

```

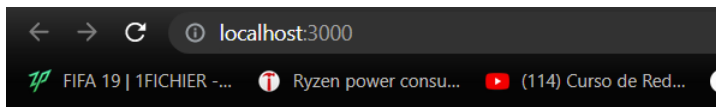
```

C:\Users\Diego\TP1-CS\2022-TP1-GIT> git push origin +G3/Principal
Total 0 (delta 0), reused 0 (delta 0), pack-reused 0
To https://github.com/FRRe-DACS/2022-TP1-GIT
+ 9b39800...1337e9a G3/Principal -> G3/Principal (forced update)

C:\Users\Diego\TP1-CS\2022-TP1-GIT>

```

Funciona nuevamente



PRUEBA

9. A partir de la rama principal (del grupo) crear una rama de test (para cambios futuros) introducir un par de commits que tengan nuevas funcionalidades y llevar a la rama principal solo el commit que se agregó anterior al head de la rama test.

Creamos la rama test

```
C:\Users\Diego\TP1-CS\2022-TP1-GIT>git checkout -b G3/test
Switched to a new branch 'G3/test'

C:\Users\Diego\TP1-CS\2022-TP1-GIT>git push origin G3/test
Total 0 (delta 0), reused 0 (delta 0), pack-reused 0
remote:
remote: Create a pull request for 'G3/test' on GitHub by visiting:
remote:   https://github.com/FRRe-DACS/2022-TP1-GIT/pull/new/G3/test
remote:
To https://github.com/FRRe-DACS/2022-TP1-GIT
 * [new branch]      G3/test -> G3/test

C:\Users\Diego\TP1-CS\2022-TP1-GIT>
```

Primer commit para sumar dos números ingresados por pantalla

```
PS C:\Users\Diego\TP1-CS\2022-TP1-GIT> git add .
PS C:\Users\Diego\TP1-CS\2022-TP1-GIT> git commit -m "Sum two numbers"
[G3/test 2351d79] Sum two numbers
 2 files changed, 12 insertions(+), 2 deletions(-)
PS C:\Users\Diego\TP1-CS\2022-TP1-GIT> git push origin G3/test
Enumerating objects: 9, done.
Counting objects: 100% (9/9), done.
Delta compression using up to 8 threads
Compressing objects: 100% (5/5), done.
Writing objects: 100% (5/5), 644 bytes | 644.00 KiB/s, done.
Total 5 (delta 2), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (2/2), completed with 2 local objects.
To https://github.com/FRRe-DACS/2022-TP1-GIT
 1337e9a..2351d79  G3/test -> G3/test
```

Segundo Commit para restar los dos números ingresados por pantalla

```
C:\Users\Diego\TP1-CS\2022-TP1-GIT>git add .

C:\Users\Diego\TP1-CS\2022-TP1-GIT>git commit -m "Sustract two numbers"
[G3/test bfb95e1] Sustract two numbers
 2 files changed, 3 insertions(+)

C:\Users\Diego\TP1-CS\2022-TP1-GIT>git push origin G3/test
Enumerating objects: 9, done.
Counting objects: 100% (9/9), done.
Delta compression using up to 8 threads
Compressing objects: 100% (5/5), done.
Writing objects: 100% (5/5), 476 bytes | 476.00 KiB/s, done.
Total 5 (delta 3), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (3/3), completed with 3 local objects.
To https://github.com/FRRe-DACS/2022-TP1-GIT
   2351d79..bfb95e1  G3/test -> G3/test

C:\Users\Diego\TP1-CS\2022-TP1-GIT>
```

Mediante el comando git log vemos el hash del commit requerido

```
C:\Users\Diego\TP1-CS\2022-TP1-GIT>git log
commit bfb95e13624e04177298cc314e51ad316952cbc5 (HEAD -> G3/test, origin/G3/test)
Author: Diego Sosa <dhsosa98@gmail.com>
Date:   Sun Apr 17 23:24:58 2022 -0300

    Sustract two numbers

commit 2351d794e040b670c408be50ece9af44190f425d
Author: Diego Sosa <dhsosa98@gmail.com>
Date:   Sun Apr 17 23:21:35 2022 -0300

    Sum two numbers

commit 1337e9adeb8290a99f3ddccd8088c2336113ec01 (tag: g3-v-1.0.0, origin/G3/Principal, G3/Principal)
Merge: 5275abf 8459a94
Author: Ramirez Joaquin <101899509+JoaquinRamirez98@users.noreply.github.com>
Date:   Sun Apr 17 22:44:41 2022 -0300

    Merge pull request #5 from FRRe-DACS/G3DS

    Transform label text to UpperCase
```

Y con el comando cherry-pick obtenemos los cambios de ese commit para devolver a la rama principal a ese estado.

```
C:\Users\Diego\TP1-CS\2022-TP1-GIT>git checkout G3/Principal
Switched to branch 'G3/Principal'
Your branch is up to date with 'origin/G3/Principal'.

C:\Users\Diego\TP1-CS\2022-TP1-GIT>git cherry-pick 2351d794e040b670c408be50ece9af44190f425d
[G3/Principal b5d28f3] Sum two numbers
Date: Sun Apr 17 23:21:35 2022 -0300
2 files changed, 12 insertions(+), 2 deletions(-)

C:\Users\Diego\TP1-CS\2022-TP1-GIT>git push origin G3/Principal
Enumerating objects: 9, done.
Counting objects: 100% (9/9), done.
Delta compression using up to 8 threads
Compressing objects: 100% (5/5), done.
Writing objects: 100% (5/5), 649 bytes | 649.00 KiB/s, done.
Total 5 (delta 2), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (2/2), completed with 2 local objects.
To https://github.com/FRRe-DACS/2022-TP1-GIT
1337e9a..b5d28f3 G3/Principal -> G3/Principal
```

Enlaces a repositorios

- **SourceForge:** [DACS-G3](#)
- **Git:** [G3/Principal](#)