**C.F.G.S DAM Y DAW .ENTORNOS DE DESARROLLO**

**TEMA 4. Control de versiones**

**PRÁCTICA DE CONTROL DE VERSIONES. (SCV)**

|  |
| --- |
| RA4. Optimiza código empleando las herramientas disponibles en el entorno de desarrollo.    f) Se ha realizado el control de versiones integrado en el entorno de desarrollo. |

En esta práctica aprenderás el concepto y el uso de un sistema de control de versiones. Debes realizar la práctica a partir de las explicaciones iniciales del profesor.

Debes entregar la práctica en formato pdf. El trabajo preferiblemente se hará en parejas pero podrá hacerse individualmente.

Esta práctica tendrá 2 puntos para el examen. Para superar la práctica debes tener una nota mínima de 1 punto. Superar la práctica exime de hacer las preguntas de esta parte en el examen final.

# PARTE 1

En tu trabajo debes entregar:

* Portada con el nombre del trabajo y autores.
* Índice con los apartados del trabajo y las páginas.
* ¿Qué es un control de versiones y para qué sirve?
* ¿Qué es git y Github?
* ¿Qué otros sistemas de control de versiones se suelen actualizar en la actualidad?
* ¿Qué son los sistemas centralizados y distribuidos? Diferencias
* Define los siguientes conceptos en el SCV (Sistema de control de versiones) git: o Repositorio

o Que incluye el fichero .git o Explicar los siguientes comandos:  Comando Add.  Comando Commit  Comando Pull.

* + Comando Push
  + Comando Status

**C.F.G.S DAM Y DAW .ENTORNOS DE DESARROLLO**

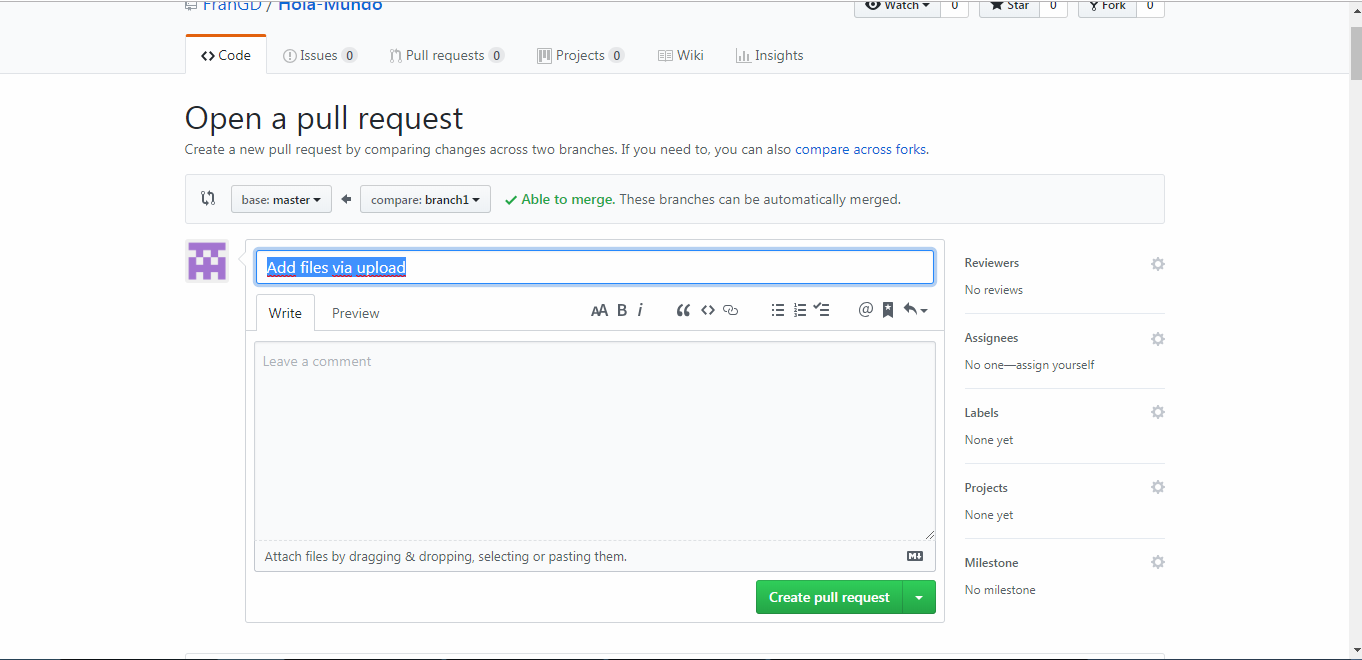
**TEMA 4. Control de versiones**

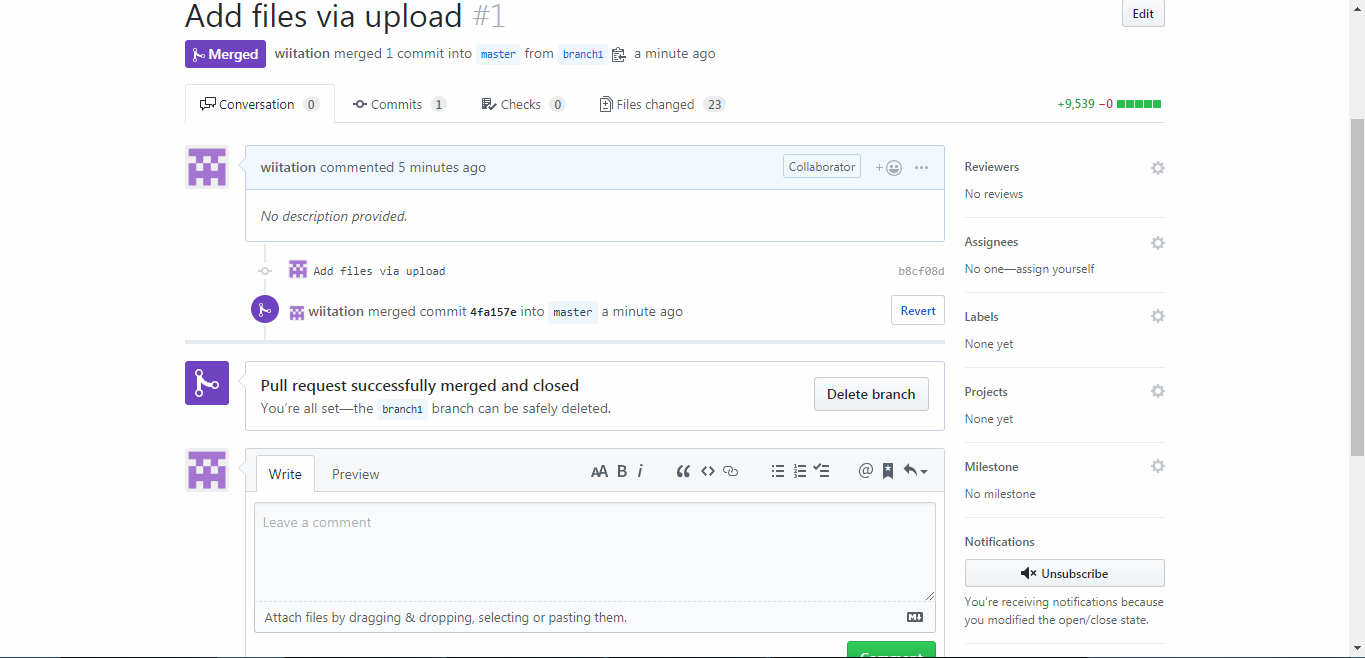
## **PARTE 2** (Para realizar esta segunda parte debes entender bien los conceptos de la primera parte)

Para que puedas tener en tu ordenador una copia de un sistema de control de versiones y puedas trabajar en paralelo con un compañero, haremos lo siguiente:

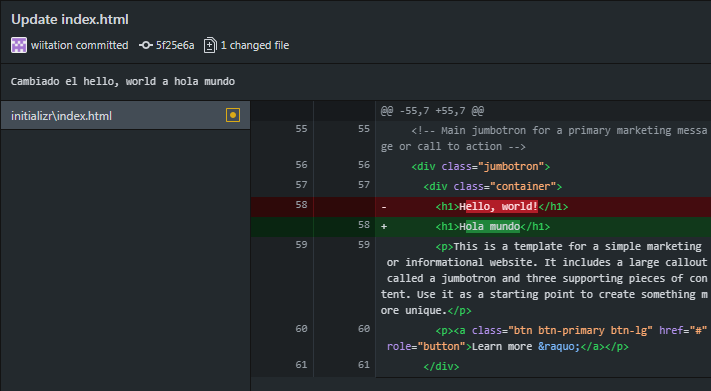
* Vamos a crear una cuenta de Github donde crearemos un repositorio. (Busca en internet como crear una cuenta y crear un repositorio).
* Crea un par de archivos de partida. Uno puede ser un documento README.txt
* Añade un colaborador a tu proyecto. Tu compañero y tu debéis tener cuentas distintas en Github
* Después de crear una cuenta en Github y un repositorio instala la aplicación github para Windows en tu ordenador.
* Realiza una copia (clone) del repositorio en tu ordenador.
* Añade carpetas con archivos y súbelos al repositorio. **¿Qué comandos debes utilizar para añadir un archivo, confirmar el cambio y subirlo al repositorio?**

Commit

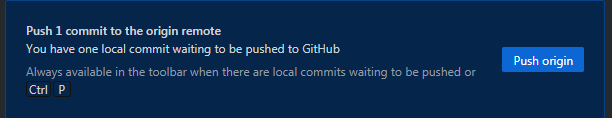




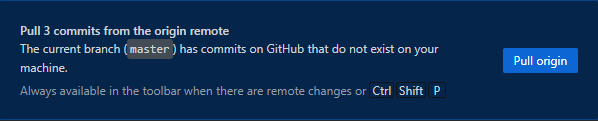
* Ambos compañeros debéis tener en vuestro ordenador una copia del proyecto en vuestro equipo local.
* Ambos compañeros debéis modificar un archivo distinto. Ve a la aplicación de github instalada. Comprueba que ha pasado.
* Realiza *commit* de ese fichero modificado para que suba al repositorio. (recuerda poner una descripción antes de hacerle commit)



* Realiza un push de ese fichero para que suba al repositorio.



* Descarga de nuevo todos las modificaciones que se han realizado en el proyecto. Debes haber recibido en tu repositorio local (clone del proyecto) los cambios que ha realizado tu compañero y tu compañero los tuyos.



* Conflictos. Prueba a modificar el mismo archivo que tu compañero en la última versión. Hazle commit y push. **¿Qué sucede? ¿Cómo lo solucionamos?**
* Se crea un conflicto entre ambos archivos con diferentes cambios.
* Añadimos los cambios al ultimo archivo subido y aceptamos cambios recientes, cambios entrantes o ambos cambios y lo subimos otra vez.

NOTA: Para esta segunda parte realiza capturas y explicaciones de lo que vas añadiendo al documento de entrega y sobre todo responde a las preguntas