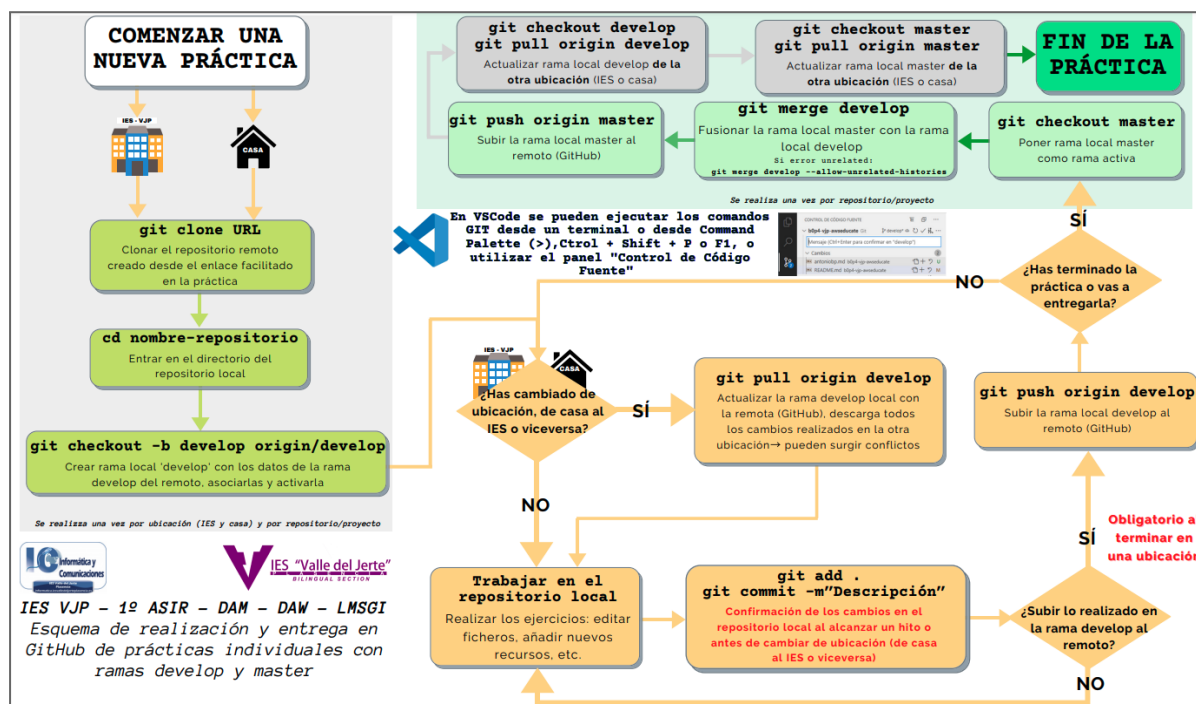


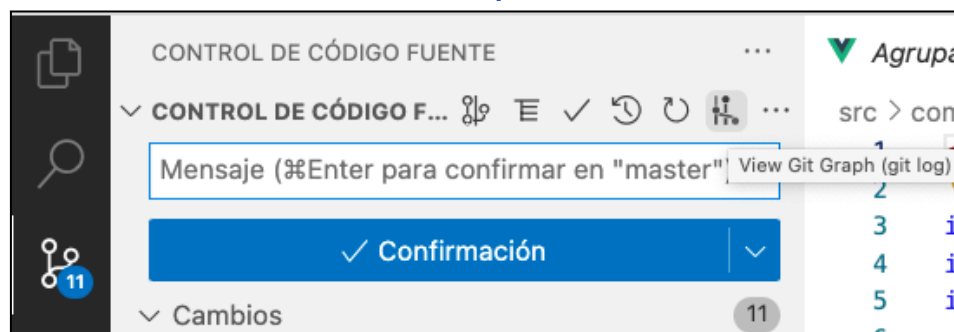
Bloque 1 - Práctica 4 - Repositorio local conectado a remoto en GitHub desde GUI VSCode

En esta práctica aplicaremos el [protocolo de realización y entrega de prácticas](#), pero sin necesidad de comandos, [desde la interfaz gráfica de VSCode](#).

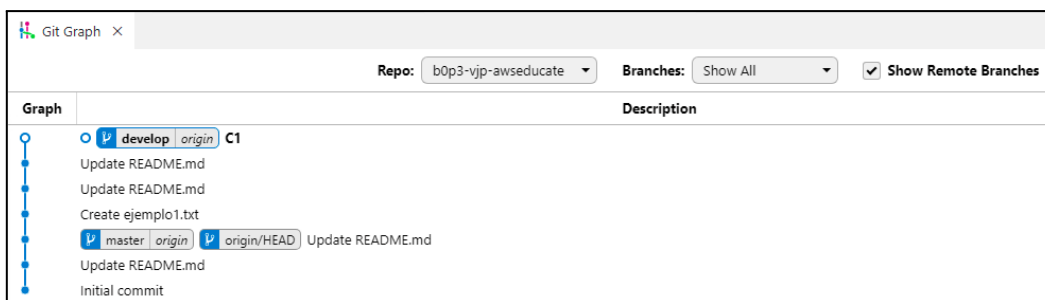


Los siguientes ejercicios se realizarán en el IES

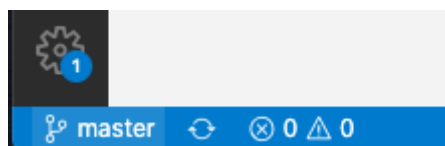
1. Instala la extensión de VSCode [Git Graph](#)



Al pulsar en los commits y demás elementos podemos ver información en más detalle, comparar contenidos de distintos commit, etc.

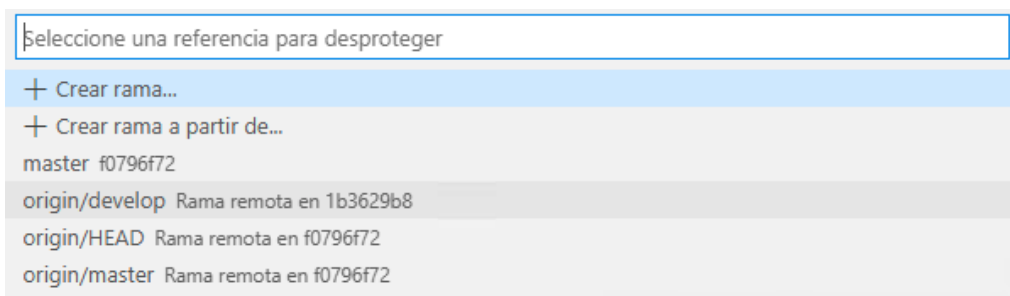


2. Pulsa en el enlace [b1p4](#) para crear un nuevo repositorio en GitHub.
3. Abre “Command Palette” (Ctrl + shift + p o F1) y escribe “Git Clone”, sigue los pasos indicados en la [diapositiva](#) para clonar el repositorio b1p4 desde GitHub en el directorio LMSGI.
4. ¿Qué ficheros hay en Working Directory? ¿cuál es la rama activa? ¿Cuántas ramas tiene?
5. Crea una rama local de nombre develop con los datos de la rama develop del remoto, asócialas y **activa** la rama develop local → pulsa en el nombre de la rama en la barra de estado, cuidado si hay más de un repositorio abierto en VSCode, la barra de estado se aplicaría al repositorio seleccionado con el ratón. *También se puede crear desde Command Palette con `git checkout` o desde el terminal con el comando `git checkout -b develop origin/develop`*



Al pulsar sobre la rama, o si utilizamos directamente “Command Palette” → “Git checkout”, pulsamos sobre el remoto que queremos asociar a uno local con el mismo nombre y activarlo, en nuestro caso “origin/develop”.

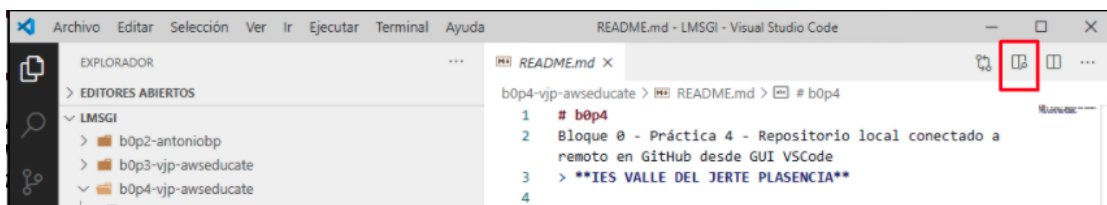
Nota: la opción “Crear rama a partir de ...” crea una local con los datos de una remota pero no las enlaza



6. Observa las ramas ejecutando `git branch -avv` y comprueba que la activa es `develop`, tiene que aparecer con un `*`, y que `develop` está enlazada con `origin/develop` y `master` con `origin/master`

```
$ git branch -avv
* develop          1b3629b [origin/develop] Update ejemplo1.txt
master            f0796f7 [origin/master] Create ejemplo1.txt
remotes/origin/HEAD -> origin/master
remotes/origin/develop 1b3629b Update ejemplo1.txt
remotes/origin/master f0796f7 Create ejemplo1.txt
```

7. ¿Qué ficheros hay en Working Directory cuando la rama activa es `develop`?
8. Abre el fichero `README.md` y pulsa en la opción “Abrir vista previa en el lateral” (ver cuadrado rojo en la imagen), observa como se aplican las marcas Markdown



Opcional - investiga por qué las notas `[^1]` en Markdown no funcionan en GitHub

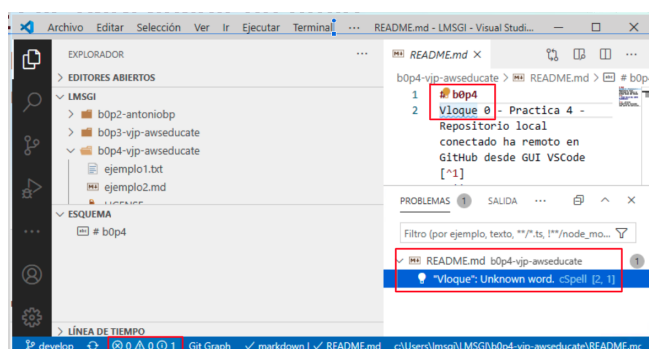
9. **Desde GitHub (puedes pulsar . para editarlo en la versión Web de VSCode o utilizar el editor interno), en la rama `develop`, añade el texto “1 - Desde IES - pruebas” al fichero `ejemplo1.txt`, guarda los cambios y cófirmalos en la rama `develop`**
10. **En VSCode**, actualiza tu copia local con los cambios que se han realizado en GitHub.
Puedes pulsar en la barra de estado, en el icono “Sincronizar cambios” que está a la derecha del nombre de la rama



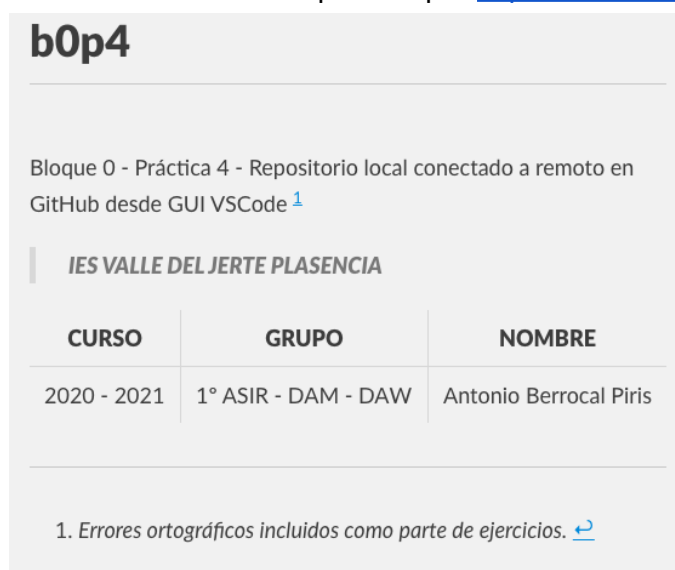
11. En **VSCode**, corrige los tres errores ortográficos del fichero `README.md` y añade una tabla realizada con Markdown con el curso, grupo y tu nombre. Guarda el fichero.


Información sobre Markdown:

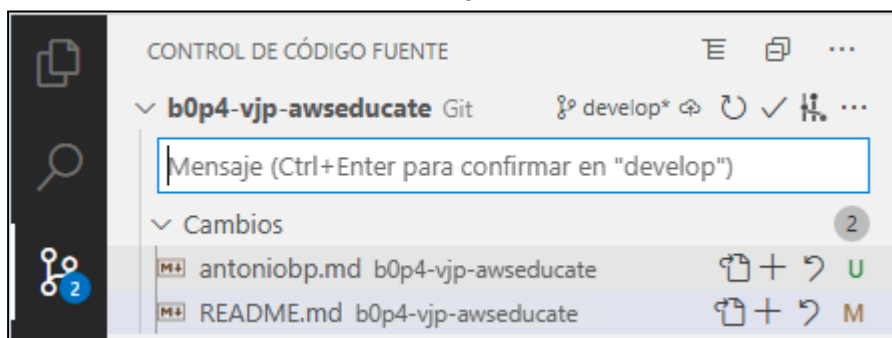
- [Probar MD online y ejemplo](#)
- [Más información](#)
- [Editor README](#)



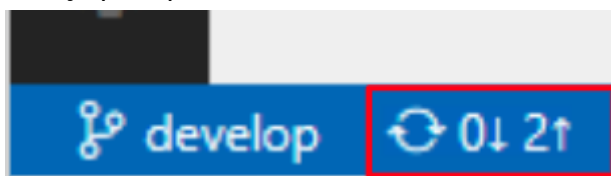
Prueba en <https://stackedit.io/app#fnref1> el contenido del fichero *README.md*
 Aspecto del fichero *README.md* interpretado por <https://stackedit.io/app#fnref1>




12. Observa cómo VSCode indica en el propio fichero *README.md* todos los cambios realizados. Pulsa en los símbolos (triángulo rojo, líneas verdes o azules) para ver detalle de cada cambio y las opciones que ofrece (deshacer, pasar a staged area, ...)
13. Comprueba pulsando sobre el icono  en la parte superior derecha del fichero *README.md* los cambios que has realizado
14. Elimina el fichero *prueba.txt*
15. Desde el panel de “Control de Código Fuente”:



- Comprueba los cambios realizados (U, M, D)
 - Pulsa sobre el fichero README.md para ver los cambios pendiente de confirmación
 - Pasa a “Staged area” todos los ficheros modificados, añadidos o eliminados, pulsa el símbolo + junto al nombre de cada fichero.
 - Confirma los cambios pulsando ✓ o escribiendo en el cuadro de texto y pulsando “Ctrl + Enter”
16. Crea un directorio de nombre img en el proyecto y añade un nuevo fichero con el logo de los CCFF, lo puedes encontrar en http://informatica.iesvalledeljertepласencia.es/?page_id=611. Si no lo descargas directamente al directorio del proyecto puedes arrastrarlo y soltarlo en el panel “Explorer”, se realizará una copia del fichero de forma automática.
17. Haz un commit que recoja estos cambios.
18. **Fusiona** la rama local master con develop. Sigue los pasos indicados en la [diapositiva](#).
19. Desde el panel Explorer → Timeline (Línea de tiempo) observa la historia de confirmaciones y pulsa en alguna para ver qué fue lo que cambió en el proyecto.
20. **Publica** la rama develop y master en el remoto en GitHub (ver [diapositiva](#)). Comprueba en GitHub que todos los cambios se han subido. *Puedes pulsar en el icono marcado en rojo para publicar los cambios en GitHub.*




21. Utiliza la cuenta de GitHub para almacenar la configuración de VSCode. Puedes encontrar información en la [diapositiva](#)
-  Bloque 1: Configuración del entorno, VSCode - GitHub - Git, protocolo de entre...

En este punto todos los cambios realizados en el IES están sincronizados con el remoto en GitHub

Realiza los siguientes ejercicios en CASA

Información sobre Markdown:

- [Documentación](#)
 - [Probar MD online y ejemplo](#)
 - [Más información](#)
22. Sincroniza VSCode de casa con el del IES para mantener la misma configuración (temas, extensiones, etc). Puedes encontrar información en la diapositiva  Bloque 1: Configuración del entorno, VSCode - GitHub - Git, proto...
23. Abre "Command Palette" (Ctrl + shift + p o F1) y escribe "Git Clone", sigue los pasos indicados en la [diapositiva](#) para clonar el repositorio b1p4 desde GitHub.
24. Crea una rama local de nombre develop con los datos de la rama develop del remoto, asócialas y **activa la rama develop** local
25. **Opcional** - Prueba alguna extensión para trabajar con MD, como por ejemplo [Markdown All in One](#) o [Markdown PDF](#)
26. Añade un enlace utilizando [Markdown](#) para mostrar el logo de los CCFF en el fichero README.md y guarda el fichero.
Aspecto del fichero README.md interpretado por <https://stackedit.io/app>

b0p4

Bloque 0 - Práctica 4 - Repositorio local conectado a remoto en GitHub desde GUI VSCode ¹

IES VALLE DEL JERTE PLASENCIA

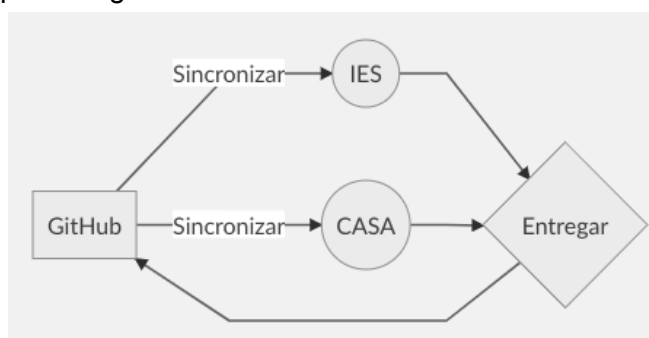
CURSO	GRUPO	NOMBRE
2020 - 2021	1º ASIR - DAM - DAW	Antonio Berrocal Piris



1. Errores ortográficos incluidos como parte de ejercicios. [↩](#)

27. Crea un nuevo fichero de nombre `principal.md`. Utiliza Markdown para añadir:

- Un encabezado de primer nivel con el texto “IES Valle del Jerte - Plasencia”
- Un encabezado de segundo nivel con el texto “LMSGI”, que es una **abreviatura** de “Lenguaje de Marcas y Sistemas de Gestión de Información”
- Un párrafo con el texto “Esquema resumen entrega de prácticas”
- El esquema siguiente



- Un párrafo con el texto, “Ejemplo de fórmulas en Markdown”

- La siguiente fórmula ([en formato LaTeX para utilizar en Markdown](#))

$$x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

Guarda el fichero.

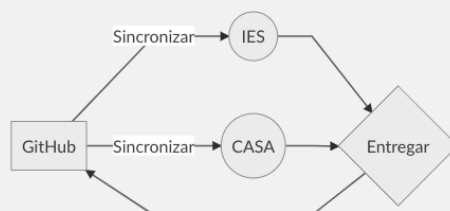
Prueba en <https://stackedit.io/app> el contenido del fichero principal.md

Aspecto del fichero principal.md interpretado por <https://stackedit.io/app>

IES Valle del Jerte - Plasencia

LMSGI

Esquema resumen entrega de prácticas



Ejemplo de fórmulas en Markdown

$$x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

28. Desde el panel de “Control de Código Fuente”, **en un solo paso**, registra los cambios en “Staged Area” y confírmalos.
29. Aquí **termina la práctica en casa**:
 - Fusiona la rama local master con develop. Sigue los pasos indicados en la [diapositiva](#).
 - Sube **ambas ramas al remoto**. Sigue los pasos indicados en la [diapositiva](#).
30. Comprueba en GitHub que la rama master se ha fusionado con develop, por lo tanto tiene que tener todos los cambios.
31. Observa en GitHub la opción Insights/Network para ver una representación gráfica de las ramas y commits.

A realizar en el IES

32. Actualiza el repositorio local con los cambios realizados en casa, tanto la rama master como la develop

33. **Opcional** - Prueba las diferentes funciones de la extensión Git Graph
34. **Opcional** - Abre la versión Web de VSCode, la conectada con GitHub (recuerda, pulsa . sobre un proyecto en la web de GitHub) o <https://vscode.dev/>, sincrónizalo con tu VSCode utilizando la cuenta de GitHub y edita un proyecto. Observa que no todas las extensiones están disponibles.
35. **Opcional** - Examina los [atajos de teclado de GitHub](#) y relacionalos con lo estudiado.
36. **Opcional** - El artículo "[El Gobierno publica un código incompleto y confuso de la 'app' Radar Covid](#)" dice *"Lo más importante es que nos falta el histórico. Es una parte esencial. No sabemos cuándo empezó, como se ha ido construyendo, sobre qué bases ni qué decisiones se han ido tomando"*, **contrasta esta noticia** clonando en el escritorio de tu ordenador el código de la app [Covid19Radar](#), utiliza Explorer → Timeline y la extensión Git Graph para revisar la historia de la app.
37. **Opcional** - Observa la historia de la app [COVID italiana](#).

Parte de esta práctica se ha realizado en el aula y parte en casa, aunque ambas partes estaban diferenciadas. En el resto de prácticas, no existirá esta diferenciación. No olvides seguir el protocolo definido en las diapositivas

A partir de aquí, en las distintas prácticas puedes utilizar uno u otro método, comandos o GUI VSCode, o los dos a la vez.

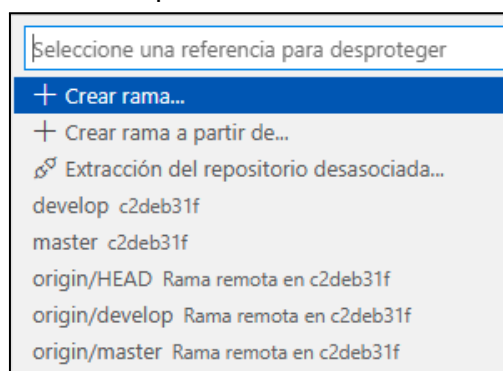
38. **Opcional** - Instala y prueba la extensión VSCode [Git History](#)
39. **Opcional** - Transforma de forma automática código Markdown en HTML y viceversa y a PDF
40. **Opcional** - Busca y analiza extensiones VSCode para facilitar el trabajo con Markdown
41. **Opcional** - Para qué podríamos utilizar <https://github.com/features/codespaces>
42. **Opcional** - Analiza CodeBerg (de Gitea), SourceHut.
43. **Opcional** - Desarrollo de nuestra propia plataforma de desarrollo basada en opensource como Gitea o GitLab Community Edition

TRABAJO BÁSICO EN UN PROYECTO COMPARTIDO A REALIZAR DESDE VSCODE

44. Clona el repositorio https://github.com/LMSGI-VJP/b1p4_compartido-grupo.git

45. Crea una rama local de nombre develop con los datos de la rama develop del remoto, asócialas y **activa** la rama develop local. Se hace según el protocolo de clases.

46. Crea una rama con tu nombre y **actívala**. Pulsa en la barra de estado en el nombre de la rama activa y selecciona la opción “Crear rama”



47. Crea un fichero de texto con tu nombre conteniendo tu nombre y el grupo al que perteneces. Confirma los cambios realizados en tu rama, en el mensaje del commit pon C0.

48. Al fichero de texto anterior añade una línea con el texto “Línea 1” y confirma los cambios añadiendo el mensaje C1 al commit.

49. Repite el ejercicio anterior 3 veces cambiando el número 1 por 2, 3 y 4.

50. Sube tu rama al remoto.



51. Conecta en GitHub a Insights - Network y observa el gráfico (https://github.com/LMSGI-VJP/b1p4_compartido-grupo/network), pulsa en los commits para ver los detalles.

52. Observa en VSCode con la utilidad Git Graph cómo evoluciona el repositorio.

53. Fusiona la rama develop con tu rama (recuerda, primero activa develop, actualiza con el remoto por si hay cambios pendientes y, por último, desde la paleta de comandos indica `git merge`). Sincroniza con el servidor remoto para que tus cambios locales pasen al servidor. Conecta a GitHub y observa el contenido de la rama develop.

LOS SIGUIENTES EJERCICIOS SE REALIZARÁN CUANDO SE HAYAN REALIZADO LOS ANTERIORES POR TODO EL ALUMNADO

54. Sincroniza con el servidor y observa en VSCode - GitGraph la evolución de las ramas.

55. Conecta en GitHub a Insights - Network y observa el gráfico (https://github.com/LMSGI-VJP/b1p4_compartido-grupo/network), pulsa en los commits para ver los detalles.

56. Fusiona tu rama con la rama develop remota, recuerda: pon tu rama como activa y desde la paleta de comandos ejecuta `git merge` y selecciona la rama develop. ¿Qué hay en tu rama?

57. Desde tu rama, modifica el fichero README.md añadiendo tu nombre y grupo a la tabla. Confirma los cambios poniendo en el mensaje CF.

58. Sube los cambios realizados al remoto.

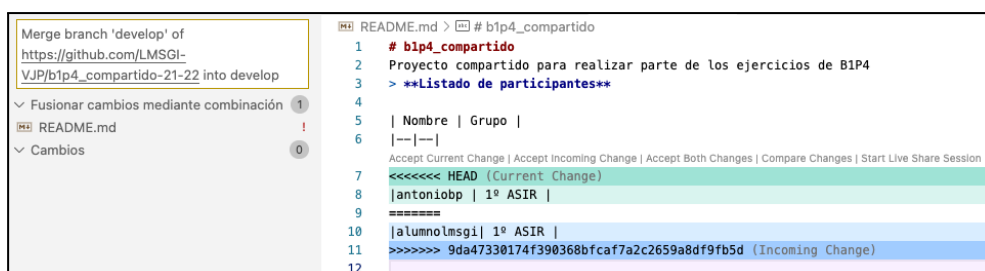
LOS SIGUIENTES EJERCICIOS SE REALIZARÁN CUANDO SE HAYAN REALIZADO LOS ANTERIORES POR TODO EL ALUMNADO

59. Cambia en **GitHub** a la rama de un compañero o de una compañera y observa el contenido del fichero README.md

60. Cambia a la rama develop y actualiza los cambios remotos.

DAM

61. Fusiona la rama develop con tu rama (recuerda, activa develop y desde la paleta de comandos `git merge` y selecciona tu rama). Es probable que surja un conflicto por haberse modificado el mismo fichero desde otra rama. Acepta todos los cambios y realiza un nuevo commit para que se solucione el conflicto. Ahora tienes en tu repositorio local las últimas actualizaciones, pero pueden surgir más conflictos.



62. Sincroniza la rama develop con el remoto. Puede surgir de nuevo un conflicto, si es así, resuélvelo aceptando todos los cambios. Repite estos pasos hasta que no haya conflictos.

63. Observa en VSCode - GitGraph la evolución del repositorio.

EL SIGUIENTE EJERCICIO LO REALIZA EL PROFESOR EN SU PUESTO

64. Fusiona la rama master con develop (recuerda, rama master activa y después git merge). Es probable que muestre un mensaje de error por ramas no relacionadas, si es así, se soluciona desde la línea de comandos con `git merge develop --allow-unrelated-histories -m"comentario"`



A REALIZAR POR EL ALUMNADO

65. Actualiza la rama develop y master con los últimos cambios.

66. Ejecuta `git config -e` para ver la configuración y la unión con el remoto.

67. Activa la rama master y ejecuta el comando `git blame README.md`