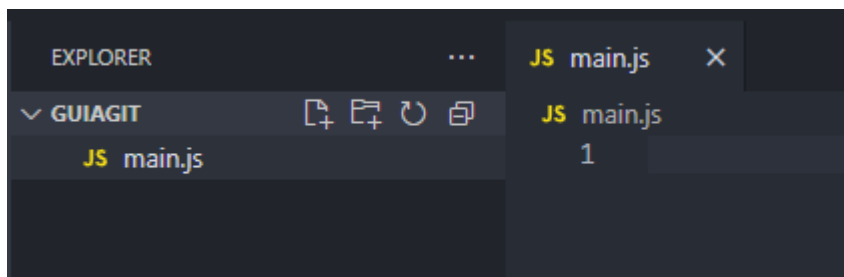


FUSIÓN DE RAMAS Y RESOLUCIÓN DE CONFLICTOS

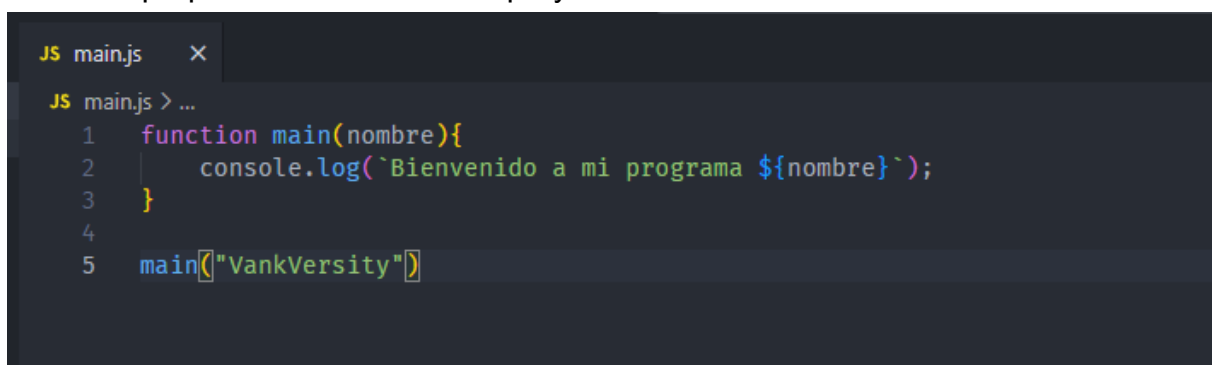
Usando merge para fusionar dos ramas en un proyecto javascript. Antes de empezar a trabajar por primera vez en su equipo, no olvide los comandos de configuración inicial.

1. FUSIÓN EXITOSA CON FAST-FORWARD

Creamos el proyecto javascript



Agregamos una pequeña funcionalidad al proyecto



En este punto, teniendo el siguiente código, iniciamos nuestro repositorio con git init y creamos nuestro primer commit ya que tenemos el layout inicial del proyecto:

```
• PS D:\vankversity\guiagit> git init
  Reinitialized existing Git repository in D:/vankversity/guiagit/.git/
• PS D:\vankversity\guiagit> git add .
• PS D:\vankversity\guiagit> git commit -m "commit inicial"
[master (root-commit) 6abcc3d] commit inicial
 1 file changed, 5 insertions(+)
 create mode 100644 main.js
• PS D:\vankversity\guiagit> █
```

En este punto nuestro historial se encuentra así:



Ahora, agregaremos funcionalidad a nuestro proyecto, agregamos una nueva función llamada despedir y hacemos un nuevo commit.

```
JS main.js M X
JS main.js > ...
1  function main(nombre){
2      console.log(`Bienvenido a mi programa ${nombre}`);
3  }
4
5  function saludaEstudiante(){
6      console.log(`hola estudiante ${nombre} `);
7  }
8
9  main("VankVersity")
10 saludaEstudiante("VankVersity")
11
12 |

● PS D:\vankversity\guiagit> git add .
● PS D:\vankversity\guiagit> git commit -m "AGREGADO funcion saludar estudiante"
[master d6dcc43] AGREGADO funcion saludar estudiante
1 file changed, 7 insertions(+), 1 deletion(-)
○ PS D:\vankversity\guiagit> 
```

Nuestro historial ahora se ve así:



Ahora pensamos en que queremos experimentar para crear una funcionalidad que salude a los profesores y no lo queremos hacer directamente en la rama `master`. Para esto creamos la rama `feature-profesores` usando el comando `git checkout -b feature-profesores` el cual crea la rama `feature-profesores` y efectúa el cambio a esta rama inmediatamente.

```
● PS D:\vankversity\guiagit> git checkout -b feature-profesores
Switched to a new branch 'feature-profesores'
○ PS D:\vankversity\guiagit> 
```

Hacemos git branch y vemos las dos ramas en nuestro repositorio. Estamos actualmente en la rama feature-profesores ,lo sabemos porque es la rama que tiene el *

```
PS D:\vankversity\guiagit> git branch
* feature-profesores
  master
PS D:\vankversity\guiagit>
```

Ahora nuestro historial luce así



Acá, el commit AGREGADO método saludarEstudiante es el ancestro que tienen en común las dos ramas.

Ahora que ya estamos en la nueva rama feature-profesores, agreguemos el método saludarProfesor y hacemos commit

```
JS main.js M X
JS main.js > ...
1  function main(nombre){
2      console.log(`Bienvenido a mi programa ${nombre}`);
3  }
4
5  function saludaEstudiante(nombre){
6      console.log(`hola estudiante ${nombre} `);
7  }
8
9  function saludaProfesor(nombre){
10     console.log(`hola profesor ${nombre} `);
11 }
12
13 main("VankVersity")
14 saludaEstudiante("VankVersity")
15 saludaProfesor("German")
16
```

```

• PS D:\vankversity\guiagit> git add .
• PS D:\vankversity\guiagit> git commit -m "AGREGADO metodo saludarProfesor"
[feature-profesores bd913d6] AGREGADO metodo saludarProfesor
1 file changed, 6 insertions(+), 2 deletions(-)
• PS D:\vankversity\guiagit>

```

Ahora nuestro historial luce así:



Si consultamos los commits en feature-profesor, veremos

```

• PS D:\vankversity\guiagit> git log --oneline
bd913d6 (HEAD -> feature-profesores) AGREGADO metodo saludarProfesor
d6dcc43 (master) AGREGADO funcion saludar estudiante
6abcc3d commit inicial
• PS D:\vankversity\guiagit>

```

Acá vemos que la rama feature-profesores contiene los commits de la master más el nuevo commit AGREGADO método saludarProfesor.

Ahora que la nueva funcionalidad saludarProfesor está lista, es hora de fusionarla con la rama master. Para esto nos cambiamos a la rama master usando git switch master

```

• PS D:\vankversity\guiagit> git switch master
Switched to branch 'master'
• PS D:\vankversity\guiagit>

```

y fusionamos la rama feature-profesores con nuestra rama master, para esto hacemos git merge feature-profesores

```

• PS D:\vankversity\guiagit> git merge feature-profesores
Updating d6dcc43..bd913d6
Fast-forward
 main.js | 8 ++++++--
1 file changed, 6 insertions(+), 2 deletions(-)
• PS D:\vankversity\guiagit>

```

Acá observamos que la acción está marcada como **Fast-forward**, esto ocurre cuando la rama que se está fusionando no tiene cambios divergentes respecto a la rama de destino, y la rama de destino puede avanzar directamente al commit de la rama fusionada sin hacer un **commit de fusión** (ver más abajo). En este caso, la rama feature-profesores no tiene cambios divergentes respecto a master, esto es, que la rama master no tiene cambios desde que se creó la rama feature-profesores.

Ahora nuestro historial se ve:



Observemos que ahora el commit AGREGADO método saludarProfesor hace parte de la rama master, ha ocurrido una fusión.

Ahora podemos eliminar la rama feature-profesor con el comando `git branch -d feature-profesores` ya que no la necesitamos más

```
PS D:\vankversity\guiagit> git branch -d feature-profesores
Deleted branch feature-profesores (was bd913d6).
PS D:\vankversity\guiagit>
```

Ahora nuestro historial luce así:

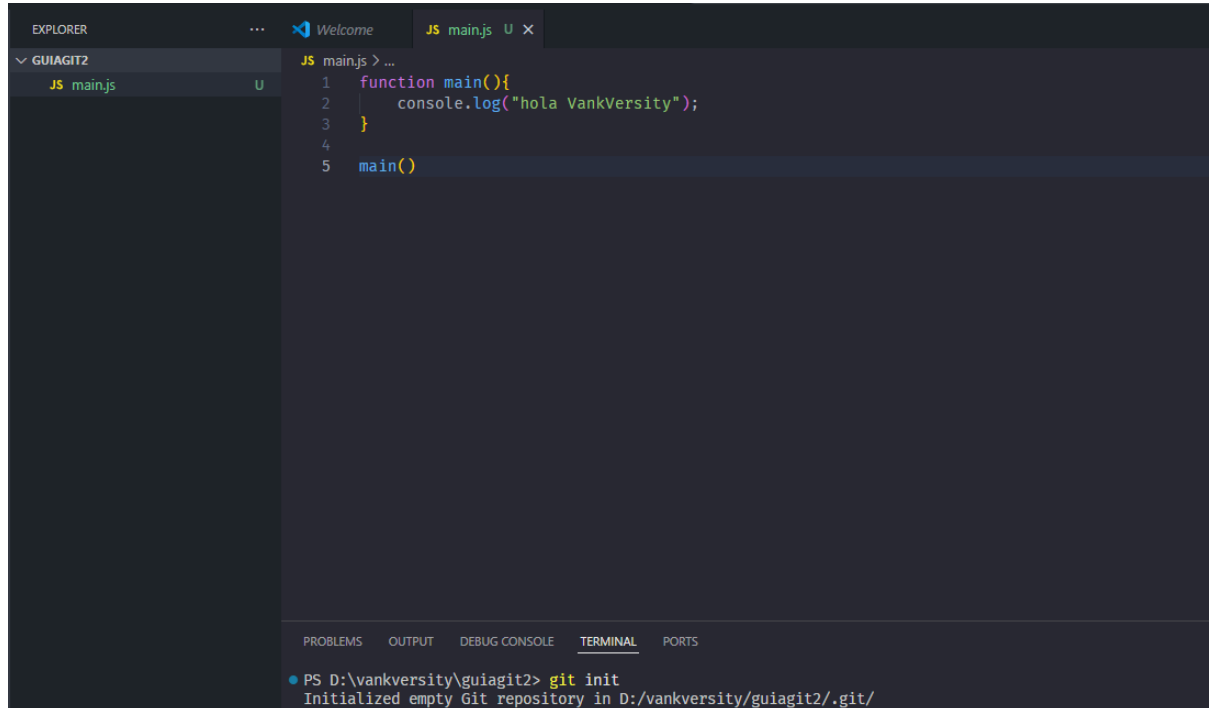


Hemos hecho exitosamente nuestra primera fusión con `git merge` mediante un fast-forward

2. FUSIÓN EXITOSA CON COMMIT DE FUSIÓN

Cuando no ocurre un fast-forward al usar git merge, se creará un commit de fusión.

Creamos un proyecto javascript con el método main. Llevamos nuestra consola a la carpeta del proyecto e inicializamos el repositorio:



The screenshot shows the Visual Studio Code interface. On the left, the Explorer pane shows a folder named 'GUIAGIT2' containing a file 'main.js'. The main editor displays the content of 'main.js', which is a JavaScript function named 'main' that logs 'hola VankVersity' to the console and then calls itself. The terminal at the bottom shows the command 'git init' being executed, resulting in the message 'Initialized empty Git repository in D:/vankversity/guiagit2/.git/'.

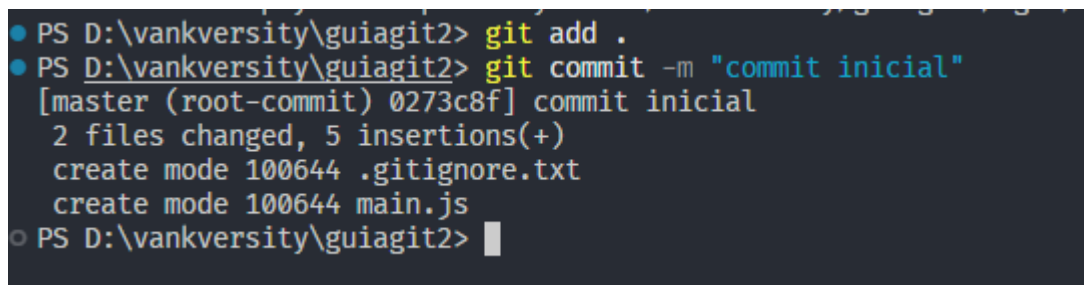
```
function main(){
  console.log("hola VankVersity");
}

main()
```

```
PS D:\vankversity\guiagit2> git init
Initialized empty Git repository in D:/vankversity/guiagit2/.git/
```

Agregamos .gitignore respectivo y hacemos commit.

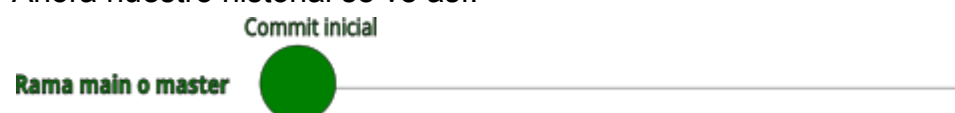
Nombre	Fecha de modificación	Tipo	Tamaño
.git	5/03/2025 1:58 a. m.	Carpeta de archivos	
main	5/03/2025 1:58 a. m.	Archivo de origen ...	1 KB
.gitignore	5/03/2025 2:00 a. m.	Documento de tex...	0 KB



The screenshot shows a terminal window with the following commands and output:

```
PS D:\vankversity\guiagit2> git add .
PS D:\vankversity\guiagit2> git commit -m "commit inicial"
[master (root-commit) 0273c8f] commit inicial
2 files changed, 5 insertions(+)
create mode 100644 .gitignore.txt
create mode 100644 main.js
PS D:\vankversity\guiagit2>
```

Ahora nuestro historial se ve así:



Trabajamos en una nueva característica de nuestro proyecto, para esto creamos la rama feature-encendido y nos cambiamos a ella, usamos git checkout -b feature-encendido

```
PS D:\vankversity\guiagit2> git checkout -b feature-encendido
Switched to a new branch 'feature-encendido'
PS D:\vankversity\guiagit2>
```

Ahora nuestro historial se ve así:



Agregamos el método encenderCoche a la clase y hacemos commit:

```
JS main.js > ...
1 function main(){
2   console.log("hola VankVersity");
3 }
4
5 function encenderCoche(){
6   console.log("Coche encendido");
7 }
8
9
10 encenderCoche()
11 main()
```

```
PS D:\vankversity\guiagit2> git add .
PS D:\vankversity\guiagit2> git commit -m "AGREGADO metodo encenderCoche"
[feature-encendido cb038fc] AGREGADO metodo encenderCoche
1 file changed, 6 insertions(+)
PS D:\vankversity\guiagit2>
```

Ahora nuestro historial se ve así:



Si hacemos git log vemos:

```
PS D:\vankversity\guiagit2> git log
commit cb038fcfe0a411c7eb974fddb5580f6d7184ecb2 (HEAD -> feature-encendido)
Author: Stiven <grstiven1004@gmail.com>
Date:   Wed Mar 5 02:04:10 2025 -0500

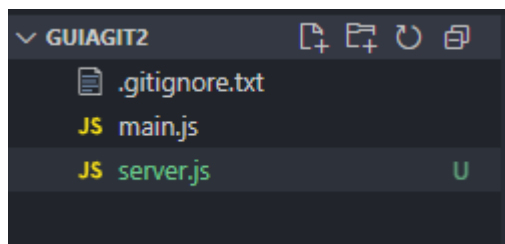
    AGREGADO metodo encenderCoche

commit 0273c8f03792b344a5bc764468c1fd1e06465b5a (master)
Author: Stiven <grstiven1004@gmail.com>
Date:   Wed Mar 5 02:01:04 2025 -0500

    commit inicial
```

Acá, en la rama feature-encendido tenemos el Commit Inicial que es commit ancestro común de las dos ramas(donde ocurrió la bifurcación) más el commit propio de la rama que acabamos de hacer: AGREGADO método encenderCoche

Ahora cambiemonos a la rama master y agreguemos un archivo nuevo Server y hagamos commit:



```
PS D:\vankversity\guiagit2> git switch master
Switched to branch 'master'
PS D:\vankversity\guiagit2> git add .
PS D:\vankversity\guiagit2> git commit -m "AGREGADO archivo server"
[master 1a1b9e6] AGREGADO archivo server
1 file changed, 0 insertions(+), 0 deletions(-)
create mode 100644 server.js
PS D:\vankversity\guiagit2>
```

Ahora nuestro historial luce así:



Cambiemonos nuevamente a la rama feature-encendido y agreguemos un nuevo método apagarCoche

```
JS main.js M X JS server.js
JS main.js > ...
1  function main(){
2      console.log("hola VankVersity");
3  }
4
5  function encenderCoche(){
6      console.log("Coche encendido");
7  }
8
9  function apagarCoche(){
10     console.log("Coche apagado");
11 }
12
13 encenderCoche()
14 apagarCoche()
15 main()
16 |
```

```
● PS D:\vankversity\guiagit2> git add .
● PS D:\vankversity\guiagit2> git commit -m "AGREGADO metodo apagarCoche"
[feature-encendido 2e52382] AGREGADO metodo apagarCoche
 1 file changed, 5 insertions(+), 1 deletion(-)
○ PS D:\vankversity\guiagit2> |
```



En este punto, las dos ramas tienen una divergencia, ya que hay cambios en master que no están en feature-encendido, por lo tanto al hacer git merge se generará un commit de fusión.

Nos cambiamos de nuevo a la rama master y hacemos git merge para fusionar los cambios de feature-encendido en master

```
PS D:\vankversity\guiagit2> git switch master
Switched to branch 'master'
PS D:\vankversity\guiagit2> git merge feature-encendido
```

se nos abrirá un editor para establecer el mensaje del commit y confirmar la fusión. Escribimos el mensaje, en este caso es Fusionado rama master con feature-encendido. Para este editor hacemos Ctrl + O y Enter, luego hacemos Ctrl + X

```
Fusionado rama master con feature-encendido
# Por favor ingresa un mensaje de commit que explique por qué es necesaria esta fusión,
# especialmente si esto fusiona un upstream actualizado en una rama de tópico.
#
# Lines starting with '#' will be ignored, and an empty message aborts
# the commit.
```

Vemos que tenemos un mensaje que nos indica que la fusión fue exitosa

```
• PS D:\vankversity\guiagit2> git merge feature-encendido
Merge made by the 'ort' strategy.
  main.js | 12 ++++++++--
  1 file changed, 11 insertions(+), 1 deletion(-)
○ PS D:\vankversity\guiagit2> █
```

Ahora el historial de cambios se ve así:



Acá vemos que la fusión de los cambios en feature-encendido con la rama master ha sido exitosa, también observamos que hay un commit de fusión(hay commit de fusión cuando no es posible un fast-forward) señalado con color violeta.

Si hacemos git log:

```

commit 0273c8f03792b344a5bc764468c1fd1e06465b5a
Author: Stiven <grstiven1004@gmail.com>
commit bfb4160124a88e64e0de626d9d772a55ad687a7 (HEAD -> master)
Merge: 1a1b9e6 2e52382
Author: Stiven <grstiven1004@gmail.com>
Date: Wed Mar 5 02:12:31 2025 -0500

    Fusonado rama master con feature-encendido

commit 2e5238289aa96a3c2ed806c0ea13f845baa4af6c (feature-encendido)
Author: Stiven <grstiven1004@gmail.com>
Date: Wed Mar 5 02:10:01 2025 -0500

    AGREGADO metodo apagarCoche

commit 1a1b9e6e320248e9480343168df9a2dd9a5e5f35
Author: Stiven <grstiven1004@gmail.com>
Date: Wed Mar 5 02:07:31 2025 -0500

```

Observamos que en el comit de fusión, en Merge: 1a1b9e6 2e52382, indica que la fusión se ha llevado a cabo con:

2e52382(último commit en master antes de la fusión)

d0a1466 (último commit en feature-encendido)

También, si nos fijamos en

```

commit 2e5238289aa96a3c2ed806c0ea13f845baa4af6c (feature-encendido)
Author: Stiven <grstiven1004@gmail.com>
Date: Wed Mar 5 02:10:01 2025 -0500

    AGREGADO metodo apagarCoche

```

quiere decir que este fue el último commit en la rama feature-encendido antes de la fusión

También en

```

commit bfb4160124a88e64e0de626d9d772a55ad687a7 (HEAD -> master)

```

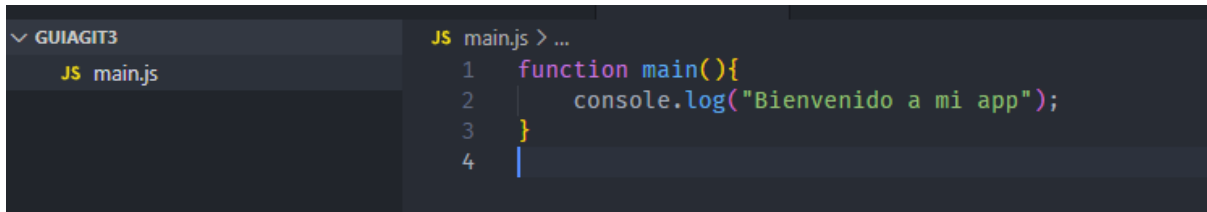
con HEAD Git nos indica que ese es el commit actual sobre el que estamos trabajando, el cual está en la rama master y su HASH es bfb4160124a88e64e0de626d9d772a55ad687a7

De esta manera hemos hecho exitosamente nuestra primera fusión usando un commit de fusión

3. FUSIÓN EXITOSA RESOLVIENDO CONFLICTOS

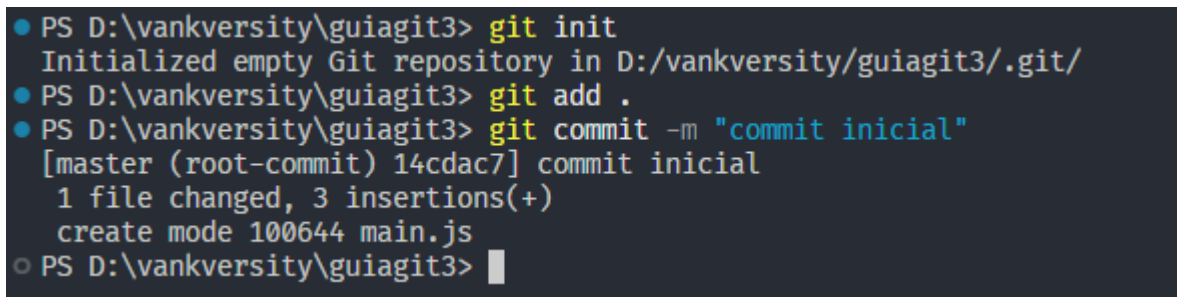
En ocasiones, al intentar fusionar dos ramas en Git, se presentan conflictos. Estos conflictos se presentan porque ambas ramas a fusionar contienen cambios en las mismas líneas de código. En este caso Git no puede decidir cuáles líneas dejar y cuáles desechar, por lo tanto debemos resolver el conflicto manualmente.

Creemos un proyecto javascript y un método main.



```
JS main.js > ...
1  function main(){
2      console.log("Bienvenido a mi app");
3  }
4  |
```

No olvidemos agregar el respectivo .gitignore al proyecto. Luego, inicializamos el repositorio con *git init* y hacemos el primer commit.

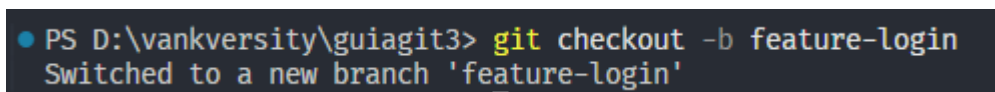


```
PS D:\vankversity\guiagit3> git init
Initialized empty Git repository in D:/vankversity/guiagit3/.git/
PS D:\vankversity\guiagit3> git add .
PS D:\vankversity\guiagit3> git commit -m "commit inicial"
[master (root-commit) 14cdac7] commit inicial
1 file changed, 3 insertions(+)
create mode 100644 main.js
PS D:\vankversity\guiagit3> |
```

Ahora nuestro historial se ve así:

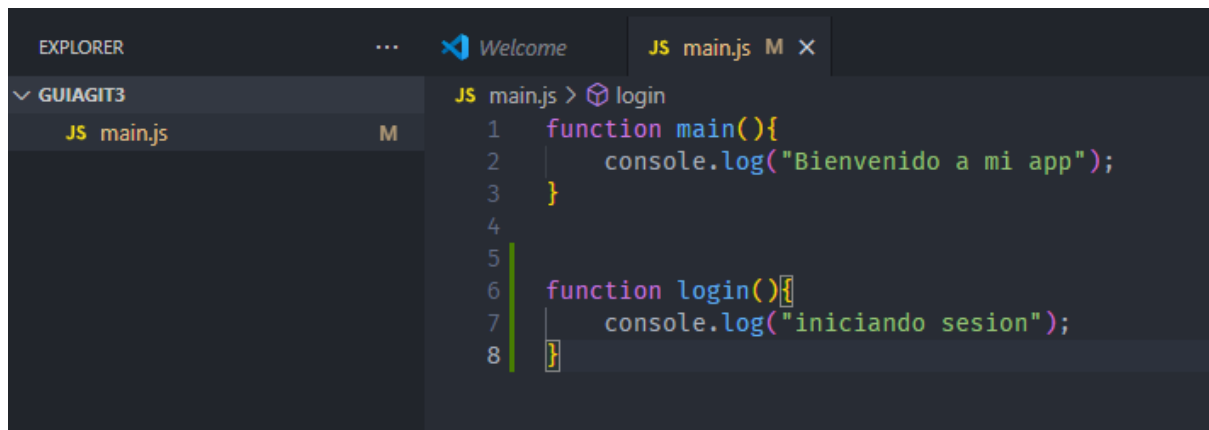


Creemos una rama feature-login para implementar una función de login en el proyecto. La creamos y nos cambiamos a ella:



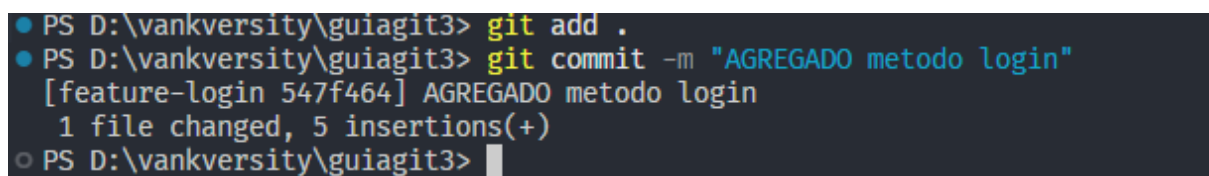
```
PS D:\vankversity\guiagit3> git checkout -b feature-login
Switched to a new branch 'feature-login'
PS D:\vankversity\guiagit3> |
```

Agregamos el método login



```
EXPLORER  ...  Welcome  JS main.js M X
v GUIAGIT3
  JS main.js M
JS main.js > login
1  function main(){
2      console.log("Bienvenido a mi app");
3  }
4
5
6  function login(){
7      console.log("iniciando sesion");
8  }
```

Hacemos commit:

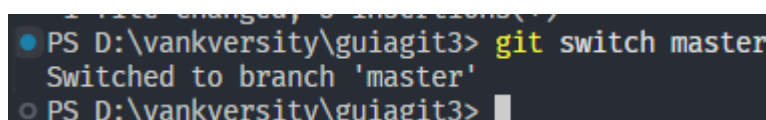


```
PS D:\vankversity\guiagit3> git add .
PS D:\vankversity\guiagit3> git commit -m "AGREGADO metodo login"
[feature-login 547f464] AGREGADO metodo login
1 file changed, 5 insertions(+)
PS D:\vankversity\guiagit3>
```

Ahora nuestro historial luce así:



Ahora volvamos a nuestra rama master



```
PS D:\vankversity\guiagit3> git switch master
Switched to branch 'master'
PS D:\vankversity\guiagit3>
```

Agreguemos el siguiente comentario:

```
JS main.js > ...
1  function main(){
2      console.log("Bienvenido a mi app");
3  }
4  /**
5   *
6   * Este comentario sobrescribe líneas que se encuentran en la
7   * rama de feature-login
8   *
9   */
```

NOTAR: Este comentario se encuentra ocupando líneas que fueron incluidas en el último commit de la rama feature-login, lo cual generará un conflicto cuando se trate de fusionar las ramas.

Ahora hagamos commit para confirmar este cambio:

```
PS D:\vankversity\guiagit3> git switch master
Switched to branch 'master'
PS D:\vankversity\guiagit3> git add .
PS D:\vankversity\guiagit3> git commit -m "AGREGADO comentario"
[master 4ce4713] AGREGADO comentario
1 file changed, 6 insertions(+)
PS D:\vankversity\guiagit3>
```

Ahora nuestro historial luce así:



Es hora de hacer nuestra fusión usando `git merge feature-login` para fusionar los cambios de feature-login en master.

Esto nos generará un conflicto

```
PS D:\vankversity\guiagit3> git merge feature-login
Auto-merging main.js
CONFLICT (content): Merge conflict in main.js
Automatic merge failed; fix conflicts and then commit the result.
PS D:\vankversity\guiagit3>
```

Si miramos el editor de código:

```
JS main.js > ...
1  function main(){
2      console.log("Bienvenido a mi app");
3  }
4  Accept Current Change | Accept Incoming Change | Accept Both Changes | Compare Changes
5  <<<<<<< HEAD (Current Change)
6  /**
7   * Este comentario sobrescribe lineas que se encuentran en la
8   * rama de feature-login
9   *
10  */
11  =====
12  >>>>>>> feature-login (Incoming Change)
13
14  function login(){
15      console.log("iniciando sesion");
16  }
17
18
```

Acá, la siguiente zona nos marca el conflicto, donde se encuentra el código incluido en el último commit de la rama feature-login y el código incluido en el último commit de la rama master

```
Accept Current Change | Accept Incoming Change | Accept Both Changes | Compare Changes
<<<<<<< HEAD (Current Change)
/**
 *
 * Este comentario sobrescribe lineas que se encuentran en la
 * rama de feature-login
 *
 */
=====
>>>>>>> feature-login (Incoming Change)
```

Acá debemos decidir cómo resolver el conflicto, tenemos tres opciones:

1. Dejar solo los cambios de la rama master (el comentario)
2. Dejar solo los cambios de la rama feature-login (el método login)
3. Dejar ambos cambios (dejar tanto el comentario como el método)

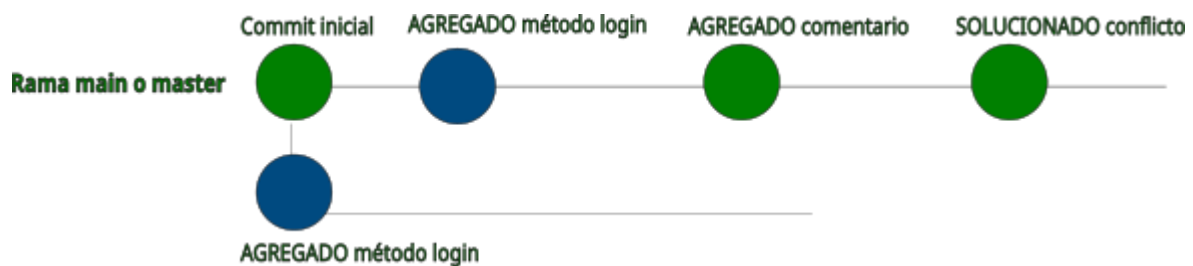
En este caso escojamos la opción 2. Procedemos entonces a editar el archivo y guardarlo, quedando el archivo así:

```
JS main.js > ...
1  function main(){
2      console.log("Bienvenido a mi app");
3  }
4
5
6  function login(){
7      console.log("iniciando sesion");
8  }
9
```

Ahora hacemos el commit de fusión y resolución del conflicto

```
PS D:\vankversity\guiagit3> git add .
PS D:\vankversity\guiagit3> git commit -m "solucionando conflictos"
[master f5ee131] solucionando conflictos
PS D:\vankversity\guiagit3>
```

Ahora nuestro historial se encuentra así:



La fusión se ha llevado a cabo exitosamente y se ha agregado el commit de fusión y resolución de conflicto *SOLUCIONADO conflicto*

Si hacemos git log en master, podemos ver el historial después de la fusión, así como los hash de los commits últimos de cada rama involucrados en la fusión.


```

● PS D:\vankversity\guiagit3> git log
commit f5ee13138bf30202d7e023590b4ae05381d54bef (HEAD -> master)
Merge: 4ce4713 547f464
Author: Stiven <grstiven1004@gmail.com>
Date: Wed Mar 5 02:39:18 2025 -0500

    solucionando conflictos

commit 4ce4713dd1ec2b9ccd3282c0878eeddd3f0f4f8a
Author: Stiven <grstiven1004@gmail.com>
Date: Wed Mar 5 02:34:52 2025 -0500

    AGREGADO comentario

commit 547f4645a9d720b8446aed30b6752cb4201854ed (feature-login)
Author: Stiven <grstiven1004@gmail.com>
Date: Wed Mar 5 02:29:05 2025 -0500

    AGREGADO metodo login

commit 14cdac7556893bdba104cc5234e1a20b9cbd4873
Author: Stiven <grstiven1004@gmail.com>
Date: Wed Mar 5 02:25:39 2025 -0500

    commit inicial
○ PS D:\vankversity\guiagit3>

```

EJERCICIOS PROPUESTOS

Ejercicio 1: Fast Forward Merge

Objetivo: Realizar un merge sin conflictos cuando la rama principal no tiene cambios nuevos.

Instrucciones:

1. Crea un nuevo repositorio en Git.
2. Agrega un archivo Javascript con un mensaje en `main()`.
3. Crea y cambia a una nueva rama (`feature-ff`).
4. Modifica el archivo Javascript en la nueva rama agregando un método adicional.
5. Realiza un commit en esta rama.
6. Vuelve a `main` y fusiona la rama con `git merge feature-ff`.
7. Observa cómo Git hace un Fast Forward sin generar un commit de merge.

Ejercicio 2: Merge con cambios divergentes (sin conflictos)

Objetivo: Realizar un merge cuando ambas ramas han cambiado diferentes líneas del archivo sin generar conflictos.

Instrucciones:

1. Asegúrate de estar en main y crea una nueva rama (feature-no-conflict).
2. Modifica el archivo Javascript agregando una nueva línea de código.
3. Realiza un commit en la rama.
4. Cambia a main y modifica el mismo archivo, pero en una línea distinta.
5. Realiza un commit en main.
6. Intenta fusionar ambas ramas y verifica que Git lo hace automáticamente sin conflictos.

Ejercicio 3: Merge con conflicto en el mismo archivo

Objetivo: Manejar un conflicto de fusión cuando ambas ramas han cambiado la misma línea.

Instrucciones:

1. Asegúrate de estar en main y crea una nueva rama (feature-conflict).
2. Modifica una línea de código en el archivo Javascript y realiza un commit en la rama.
3. Cambia de vuelta a main y edita la misma línea del archivo.
4. Realiza un commit en main.
5. Intenta hacer un merge y observa que Git muestra un conflicto.
6. Edita manualmente el archivo para resolver el conflicto.
7. Usa `git add` para marcar el archivo como resuelto y confirma el merge con un commit.

Ejercicio 4: Merge con conflictos en múltiples archivos

Objetivo: Resolver conflictos en más de un archivo.

Instrucciones:

1. Asegúrate de estar en main y crea una nueva rama (feature-multiple-conflicts).

2. **Modifica dos archivos Javascript distintos y haz un commit en la nueva rama.**
3. **Cambia a main y edita los mismos archivos en líneas diferentes.**
4. **Realiza un commit en main.**
5. **Intenta hacer un merge y revisa los conflictos en ambos archivos.**
6. **Edita los archivos para resolver los conflictos.**
7. **Usa `git add` para marcar los archivos como resueltos y confirma el merge con un commit.**

Ejercicio 5: Trabajar un proyecto Javascript de su elección, realizando múltiples commits y creando y fusionando varias ramas. Esto con el fin de ganar práctica en la implementación de ramas, resolución de conflictos y manejo de Git en general.