

## ACTIVIDAD PRÁCTICA - GIT EN ECLIPSE

***Por Pedro Berrueco***

Trabajo propuesto como  
cumplimiento de los requisitos  
para la actividad práctica de la  
unidad 3.

Modulo de Entornos de  
Desarrollo.

Curso en Desarrollo de  
Aplicaciones Multiplataforma  
Online.

Enero 2023

## Índice

ACTIVIDAD PRÁCTICA - GIT EN ECLIPSE .....	- 1 -
Descripción de la actividad Práctica .....	- 3 -
Introducción.....	- 3 -
Ejercicio 1.....	- 4 -
Ejercicio 2.....	- 4 -
Ejercicio 3.....	- 6 -
Ejercicio 4.....	- 6 -
Ejercicio 5.....	- 8 -
Ejercicio 6.....	- 10 -
Ejercicio 7.....	- 11 -
Ejercicio 8.....	- 13 -
Ejercicio 9.....	- 15 -
Ejercicio 10.....	- 15 -
Conclusiones .....	- 17 -

## Descripción de la actividad Práctica

En esta actividad debes crear un documento profesional tal y como se describe en el entregable. Debes seguir los pasos en orden, no puedes pasar al siguiente sin haber terminado el anterior. En cada apartado debes incluir capturas de pantalla de los pasos que has seguido, junto con la explicación de los pasos que has dado.

## Introducción

El objetivo de esta práctica es conocer si el alumno ha adquirido los conocimientos necesarios para poder trabajar con Git, con la resolución de los ejercicios se van probando los distintos commando de Git y se ponen en práctica de una manera muy gráfica.

El hecho de utilizarlos desde Eclipse genera que además se practique con el complemento eGit de Eclipse que es muy utilizado en el mundo del Desarrollo.

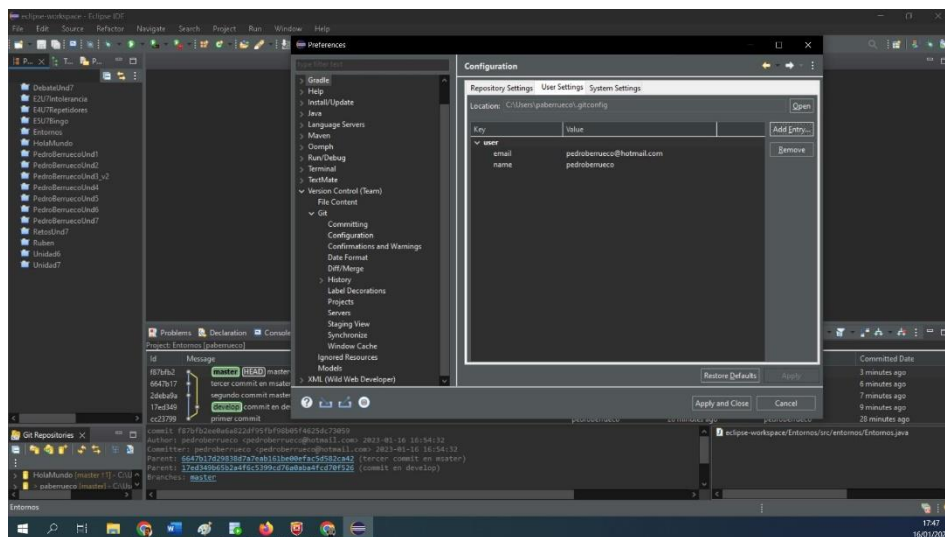
Por otro lado con la realización de esta práctica también se adquieren conocimientos del uso de metodología GitFlow que nos ayudará a entender la forma de trabajar con git en ramas y repositorios compartidos en GitHub u otros sistemas de almacenamiento de repositorio online. Con cada paso se va poniendo en práctica lo aprendido en la Unidad.

## Ejercicio 1

### Configuración de git en Eclipse:

Una vez instalado git en el ordenador, desde Eclipse se ha configurado el usuario de git para poder hacer uso de sus funciones desde el IDE de Desarrollo.

Para ello nos vamos a “Window/Preferences/Version Control(Team)/Git/Configuration” entramos en la pestaña de “User Settings” y añadimos user.name y user.email. como vemos en la imagen siguiente.



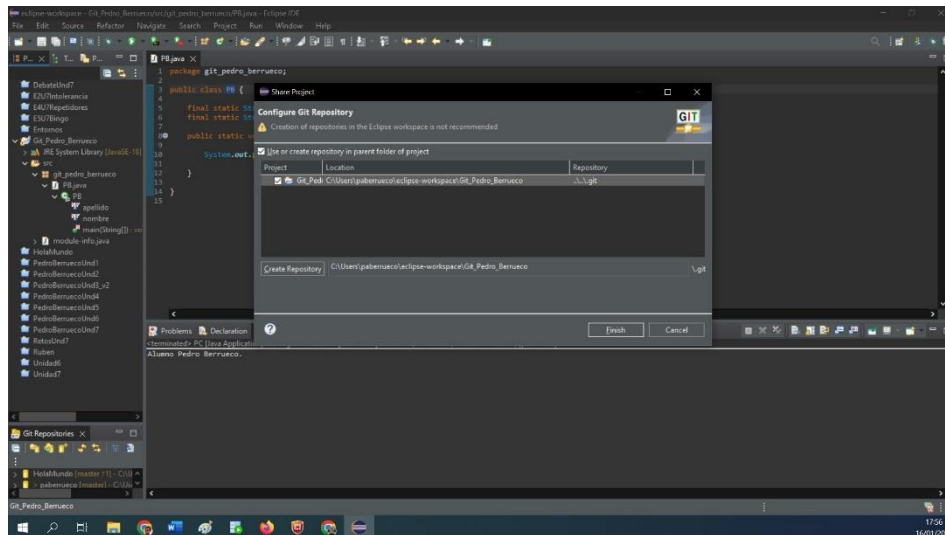
## Ejercicio 2

### Creación de un repositorio:

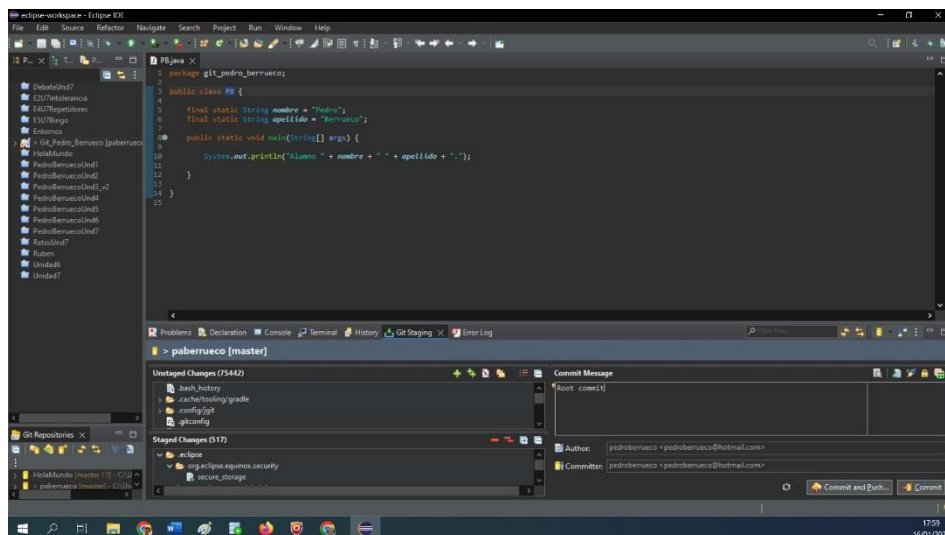
Se crea un Proyecto en Eclipse con el nombre “Git\_Pedro\_Berruenco”, en ese Proyecto se crea una clase con las siglas PB y dentro de crea el método main con un println dónde se muestra la frase “Alumno Pedro Berruenco”. Entonces nos situamos en dicho Proyecto con botón derecho y sobre marcamos sobre la opción “Team/Share Project...” seleccionamos el Proyecto y pinchamos en Finalizar,

## ACTIVIDAD PRACTICA – Pedro Berrueto

entonces ya estaremos trabajando con el Sistema de ramas de Git en dicho Proyecto. Esto sería como ejecutar un “git init” por terminal.



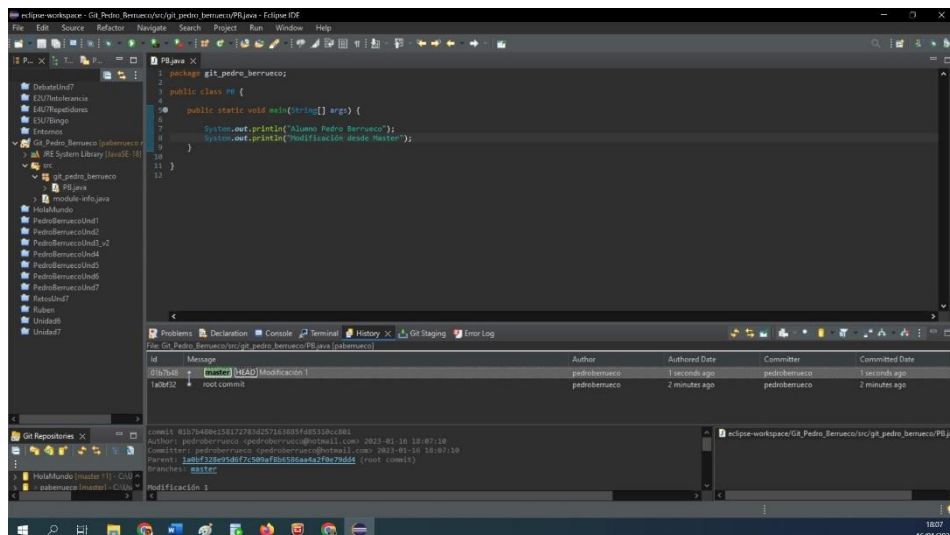
Para ejecutar Commit se debe Seleccionar nuevamente la pestaña “Team/commit...” se marca el + para agregar los ficheros modificados, esto sería igual que ejecutar el comando “git add <archivo>”.



## Ejercicio 3

### Segundo Commit en Master:

Se introduce una nueva instrucción en el fichero “System.out.println(“Modificación desde Master”);” se crea un segundo commit siguiendo los mismos pasos que en el ejercicio anterior con el comentario “Modificación 1”. Como se puede ver en la siguiente imagen.

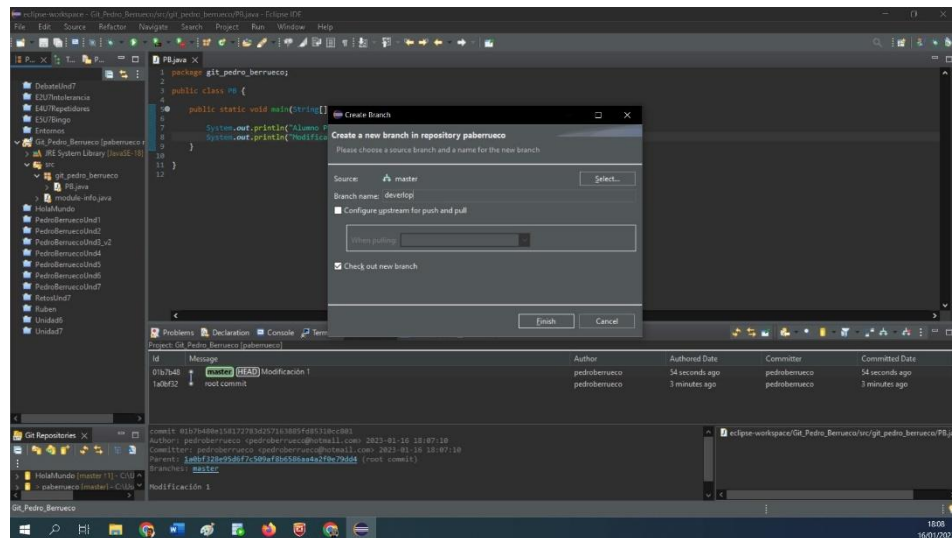


## Ejercicio 4

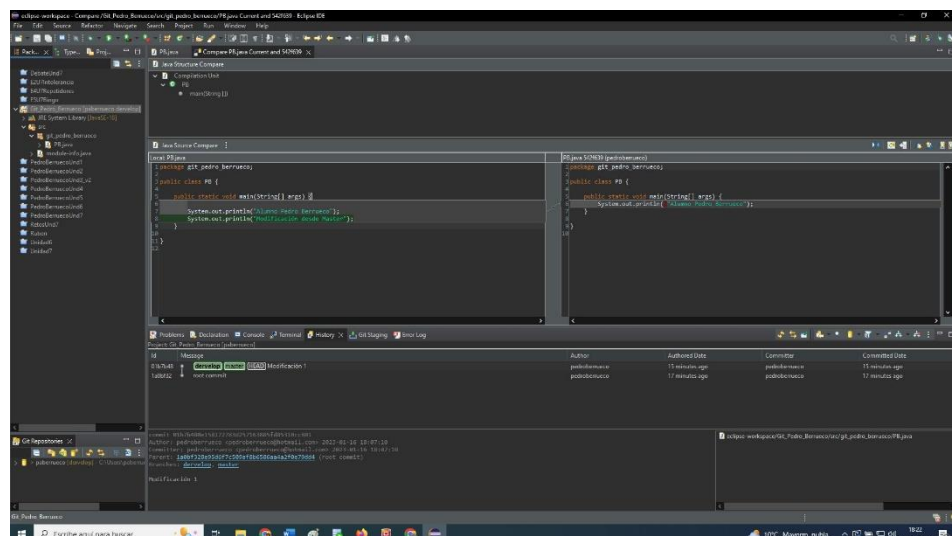
### Crear rama Develop y commit:

Para crear una rama develop se ha seleccionado la opción “Team/Switch to/New Branch” y se ha escrito el nombre de la nueva rama, esto equivale al commando “git branch -m <nombrerama>”.

## ACTIVIDAD PRACTICA – Pedro Berrueto

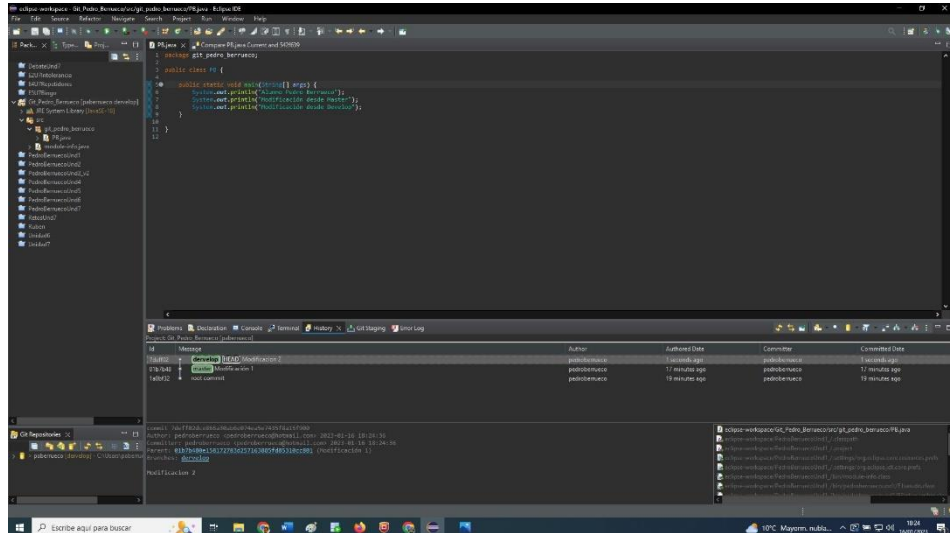


A continuación se procede a modificar nuevamente el fichero añadiendo la línea “System.out.println(“Modificación desde Develop”);”, entonces se ha creado un nuevo commit, desde e menu “Team/commit...” y se ha marcado con el mensaje “Modificación 2”.



## ACTIVIDAD PRACTICA – Pedro Berruero

Abajo podemos ver el resultado final con la cabeza situada en la rama develop.



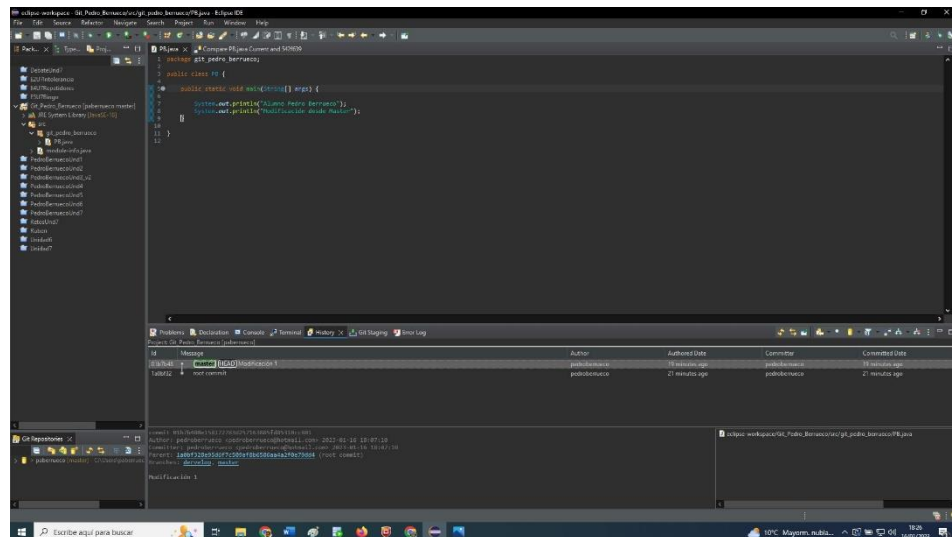
## Ejercicio 5

### Nuevo Commit desde Master:

Ahora nos posicionaremos en la rama master nuevamente, para eso seleccionamos la opción “Team/Switch to” y elegimos la opción master, esto equivaldría al comando “git checkout master”, de esta forma estaríamos retrocediendo en el tiempo en nuestro Proyecto y los archivos y las ramas volverían al punto anterior donde hicimos el último commit en esta rama, por lo que los cambios posteriores en develop habrían “desaparecido” a simple vista. Para ver todos los cambios, incluso los posteriores debemos seleccionar la opción a “Change which commits to show...” que equivaldría a realizar un “git log --oneline --all”. Ahora crearemos un nuevo cambio en la rama master añadiendo la línea “System.out.println(“Modificación desde Master - Cambiado”);” y haremos un commit con el mensaje “Modificación 3”. Podemos ver cómo quedaría en la imagen siguiente.



## ACTIVIDAD PRACTICA – Pedro Berruero



```

1 package git.pedro.berruero;
2
3 public class PedroBerruero {
4
5     public static void main(String[] args) {
6
7         System.out.println("Alonso Pedro Berruero");
8         System.out.println("MultiFisica de Pedro Berruero");
9     }
10 }
11
12

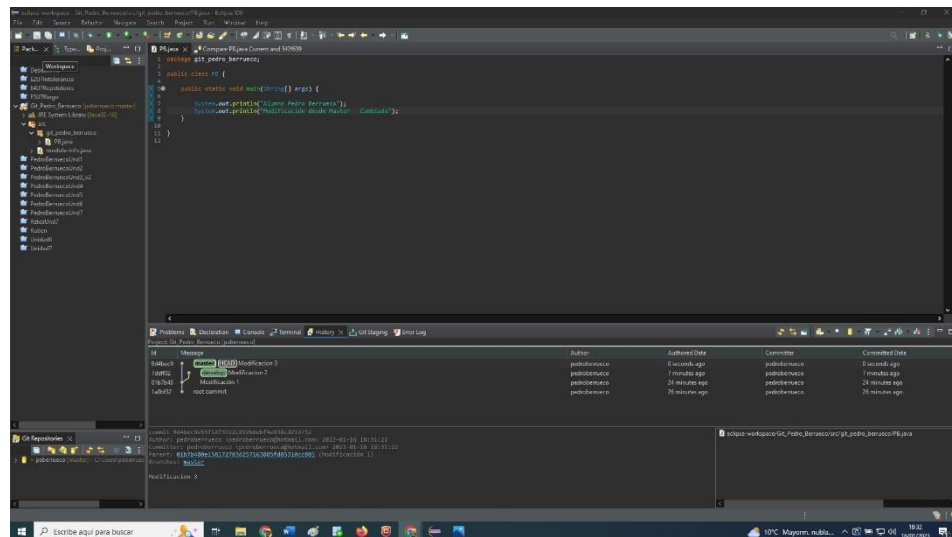
```

Console Output:

```

C:\Users\pedro\workspace\git.pedro.berruero> java -cp . PedroBerruero
Alonso Pedro Berruero
MultiFisica de Pedro Berruero

```



```

1 package git.pedro.berruero;
2
3 public class PedroBerruero {
4
5     public static void main(String[] args) {
6
7         System.out.println("Alonso Pedro Berruero");
8         System.out.println("MultiFisica de Pedro Berruero");
9     }
10 }
11
12

```

Console Output:

```

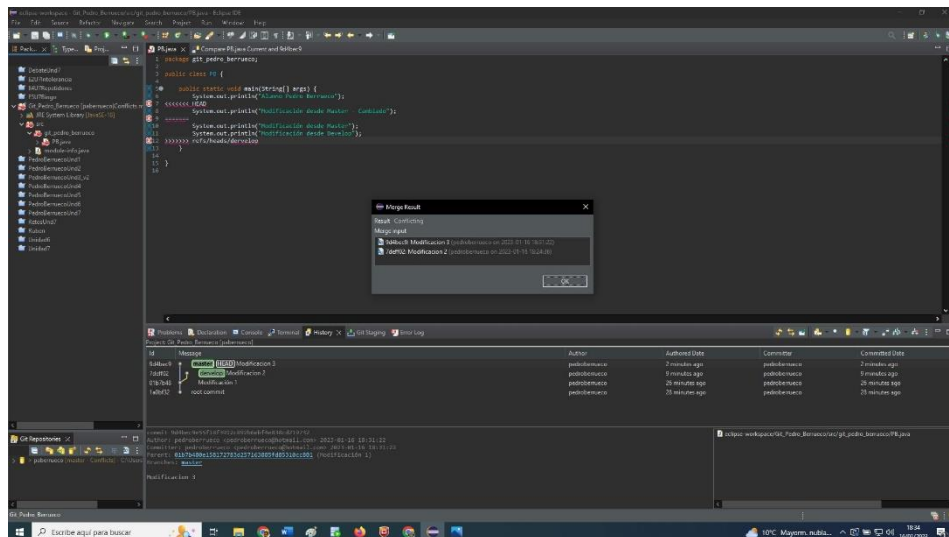
C:\Users\pedro\workspace\git.pedro.berruero> java -cp . PedroBerruero
Alonso Pedro Berruero
MultiFisica de Pedro Berruero

```

## Ejercicio 6

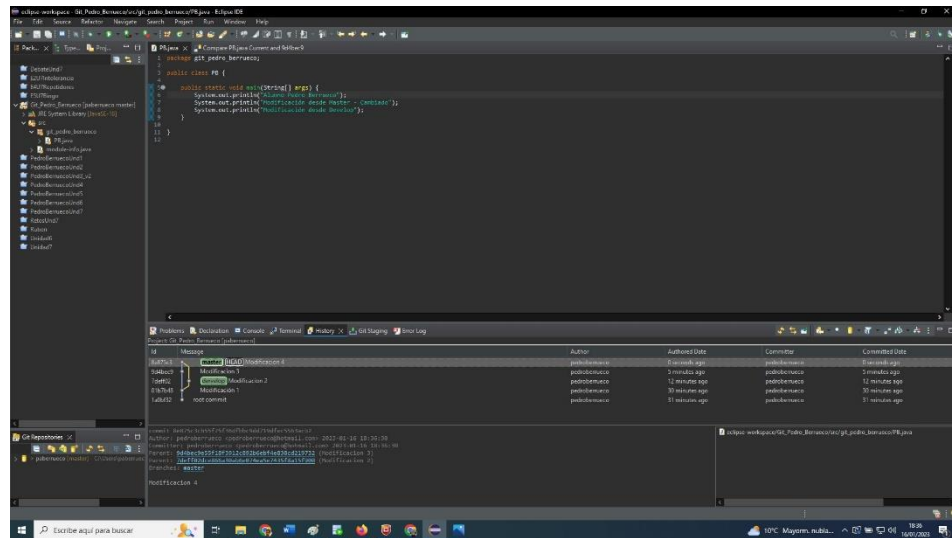
### Fusión ramas:

En este ejercicio se va a provocar un conflicto entre ramas, para ello se va a realizar un mergeo de la rama develop a la rama master, para ellos nos situamos en la rama master y seleccionamos la opción “Team/Merge...”. Esto equivaldría al commando “git merge <rama origen>”. Al pulsar sobre Merge nos aparecerá una Ventana indicando que se produce un conflicto ya que se ambas ramas han sido evolucionadas y tienen modificaciones propias que la otra no tiene, se debe resolver manualmente dichos conflictos.



Se solventa el conflicto como se indica en el enunciado y se realiza un nuevo commit de con el mensaje “Modificación 4” y ya se puede ver la fusión de ramas correctamente.

## ACTIVIDAD PRACTICA – Pedro Berruero

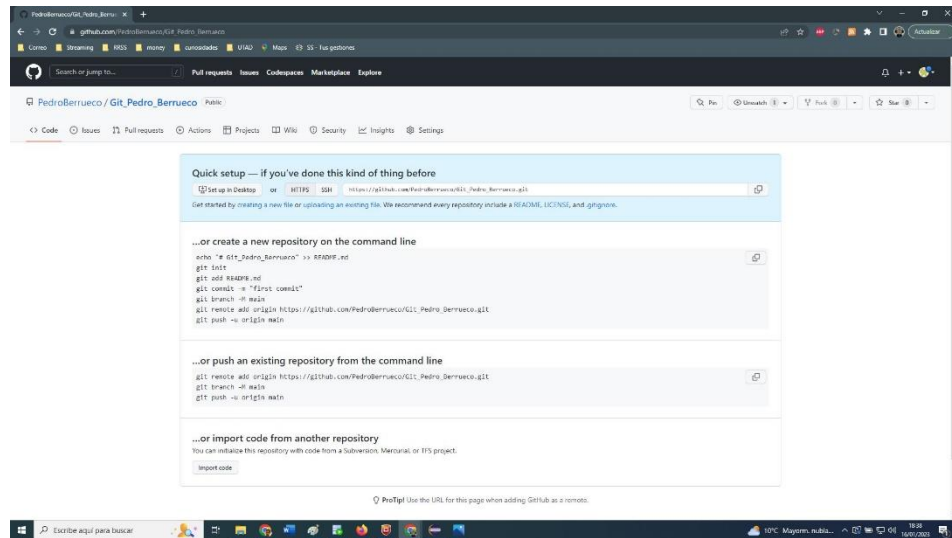


## Ejercicio 7

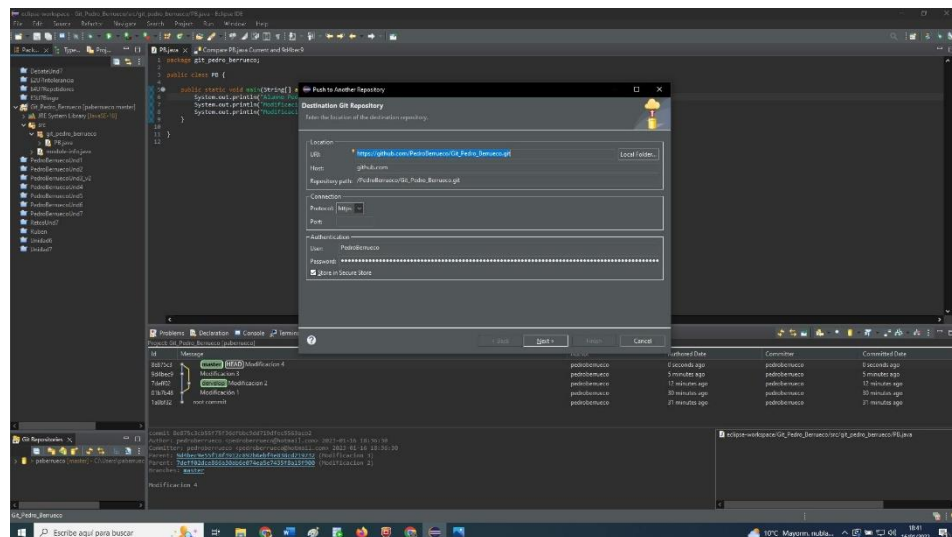
**Push a Github:**

Se crea un repositorio nuevo en GitHub con la idea de poder subir nuestro repositorio local de git, lo cual nos puede servir para compartir nuestro Proyecto o para mantener una copia de seguridad.

## ACTIVIDAD PRACTICA – Pedro Berrueto

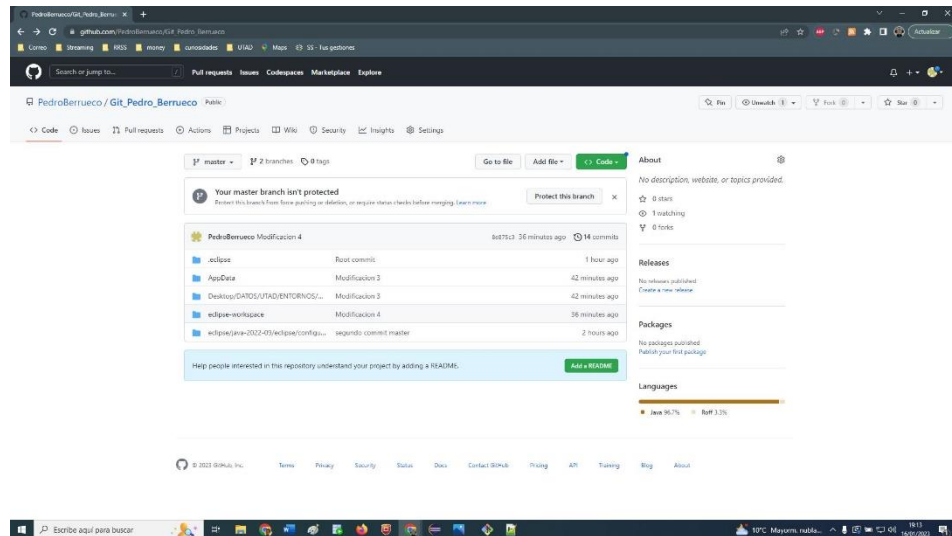


Ejecutaremos el commando “git remote add origin <URL del Proyecto github>” y a continuación “git push -u origin <rama>” que en nuestro eclipse equivaldría a “Team/Push”, la primera vez habrá que configurar el usuario y la contraseña que será el token que hayamos generado previamente.



Realizamos el push con cada una de las ramas para que se nos suban ambas a GitHub, para ello debemos posicionarnos sobre la rama en cuestión que se vaya a subir.

## ACTIVIDAD PRACTICA – Pedro Berrueco

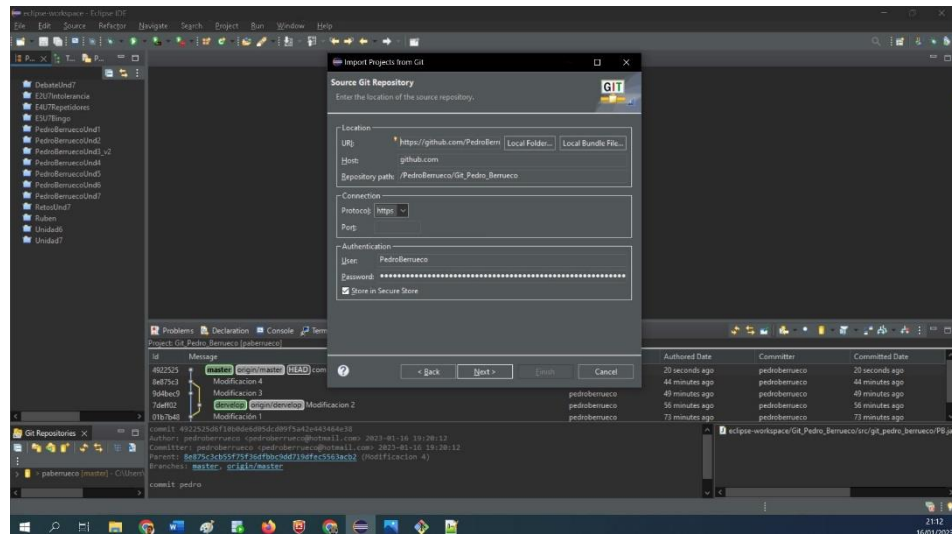


## Ejercicio 8

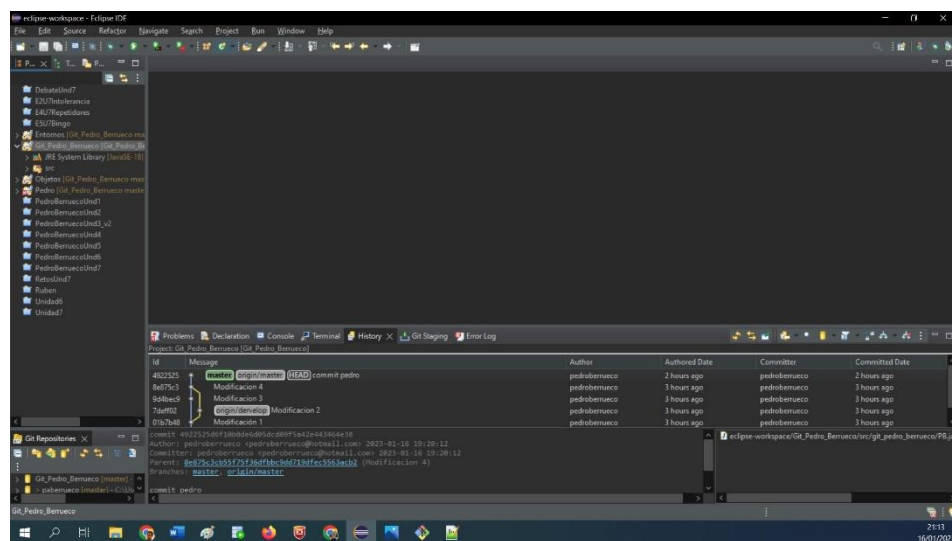
### Borrado del repositorio local y clone:

En este apartado se pretende demostrar que una vez subido nuestro Código a github este puede servirnos de copia de seguridad, para ello se elimina el Proyecto de Eclipse con botón derecho delete, y marcando la opción “Delete project content on disk” para que no quede la copia en el discoduro, despues emulariamos el commando “git pull” que aja el contenido de la rama master de nuestro repositorio en github, para ello seleccionamos la opción “File/Import Git/Project from Git,”.

## ACTIVIDAD PRACTICA – Pedro Berrueto



Aquí Podemos ver cómo se ha recuperado el Proyecto en Eclipse.

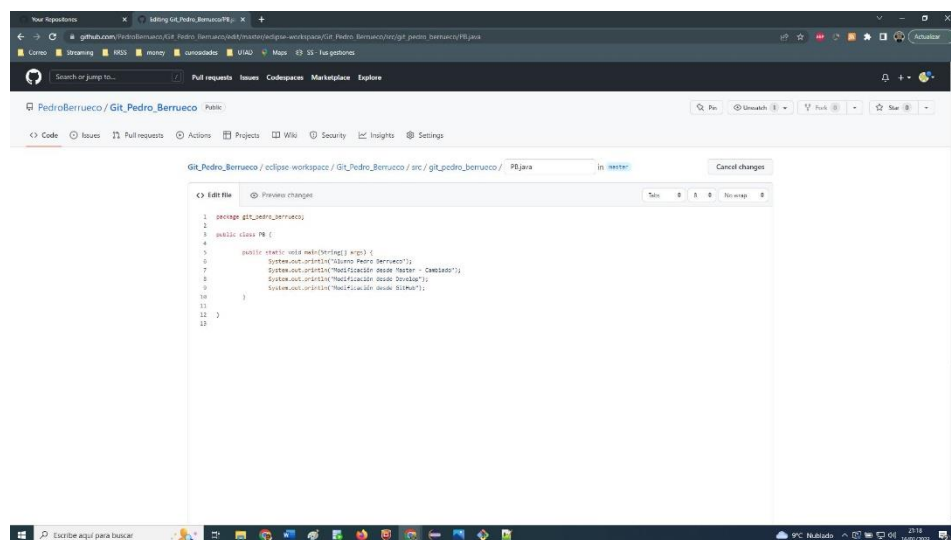


## Ejercicio 9

### Commit desde Github:

Desde Github también Podemos modificar los ficheros y ejecutar commit, esta información se actualizará en nuestro Proyecto en local.

Para demostrarlo accedemos al repositorio en GitHub, entramos en la carpeta src y luego en la clase creada anteriormente, pinchamos el icono de un lápiz ("Edit this file"), y editamos el Código creando una nueva línea "Modificación desde GitHub". En la parte inferior de esta misma pantalla tenemos la zona de "Commit changes" donde Podemos escribir un comentario, en este caso "Modificación 5" y luego pulsar en el botón "Commit changes".

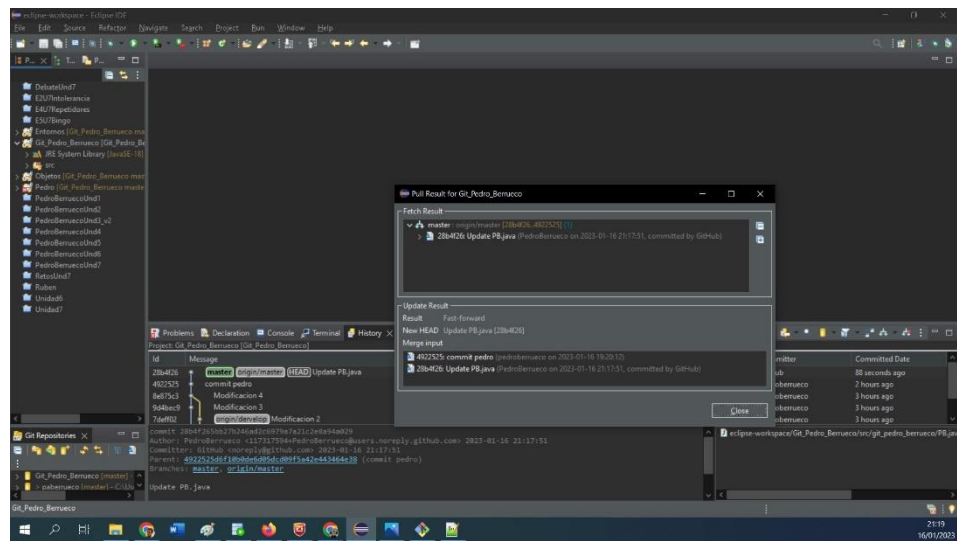


## Ejercicio 10

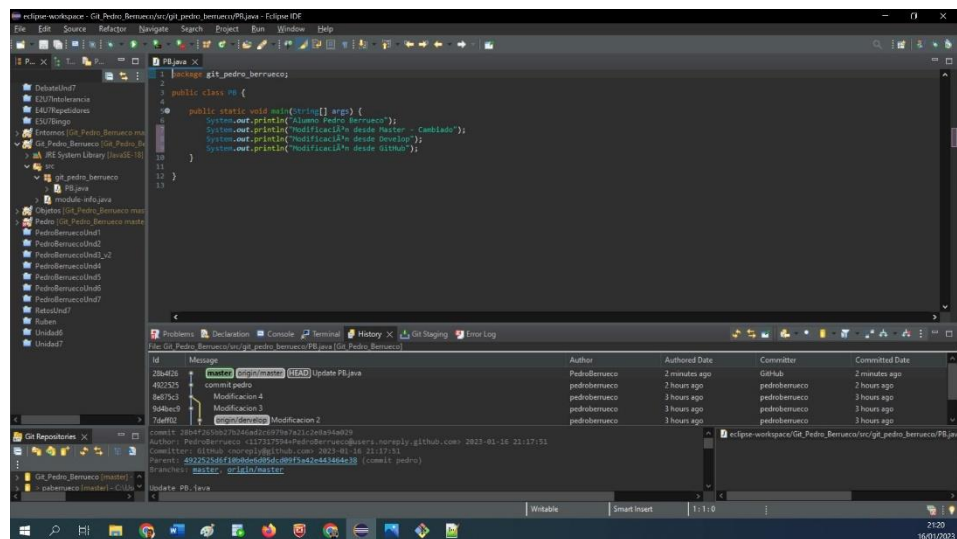
### Pull desde Eclipse:

Si realizamos un nuevo pull desde eclipse veremos como se clonan los cambios en nuestro repositorio local como se puede ver en la imagen siguiente.

## ACTIVIDAD PRACTICA – Pedro Berrueto



Y Aqui vemos graficamente como han quedado la ramas igualando master/origin.





## Conclusiones

Después de trabajar con git, gitflow, github y egit de eclipse durante esta práctica, he podido llegar a varias conclusiones. Primero, he aprendido acerca de la importancia de la gestión de versiones. Esto es muy importante para los desarrolladores de software, ya que les permite tener un control sobre el código, permitiendo a los desarrolladores regresar a versiones anteriores, sin perder el trabajo realizado.

Además, he aprendido acerca de la herramienta git, que es una excelente manera de controlar versiones. Utilizar git me ha permitido trabajar con ramas, realizar commit, hacer merge, resolver conflictos, y subir y actualizar el código desde github. Estas son todas funciones importantes para el trabajo de un desarrollador de software, por lo que el conocimiento de git es una habilidad importante para cualquier profesional de la programación.

También he aprendido acerca de gitflow, que es una metodología para trabajar con git. Esto me ha permitido organizar mi código de la mejor manera, para que todos los desarrolladores que trabajen en el mismo proyecto puedan compartir y trabajar en el mismo código. Además, me ha permitido entender cómo el trabajo de un equipo de desarrolladores puede ser más eficiente a través del uso de git.

Por último, he aprendido acerca de github y egit de eclipse. Esto me ha permitido entender mejor cómo trabajar con código remoto, así como mejorar la productividad al trabajar con el código. Estas herramientas me han permitido compartir mi código con otros desarrolladores, hacer seguimiento de los cambios realizados, y mantener controlado el flujo de trabajo de mi equipo.

En conclusión, esta práctica con git, gitflow, github y egit de eclipse me ha permitido entender mejor la gestión y control de versiones, así como mejorar mi productividad al trabajar con el código de un equipo de desarrolladores. Esto me ha ayudado mucho a mejorar mis habilidades como desarrollador de software, y espero seguir aprendiendo y mejorando mi trabajo con estas herramientas.

