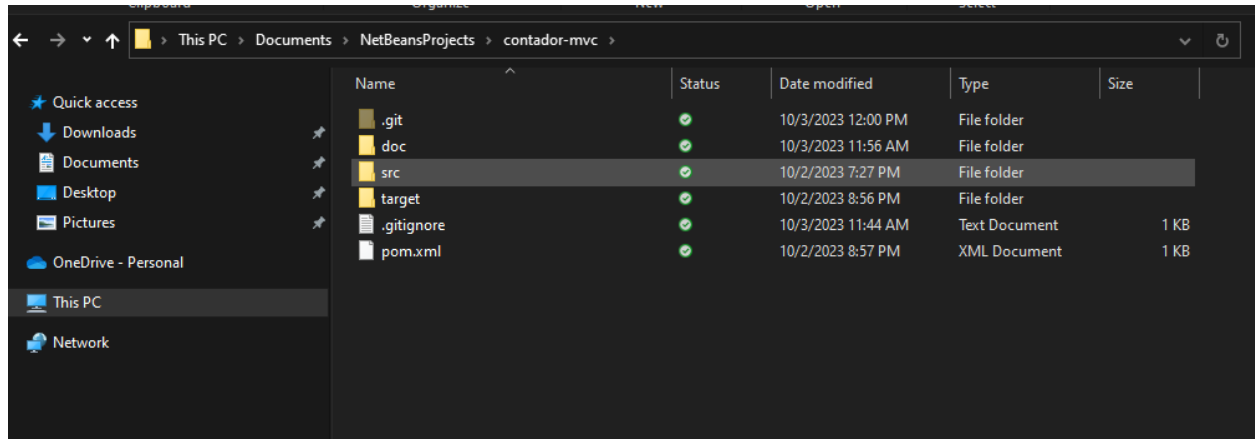
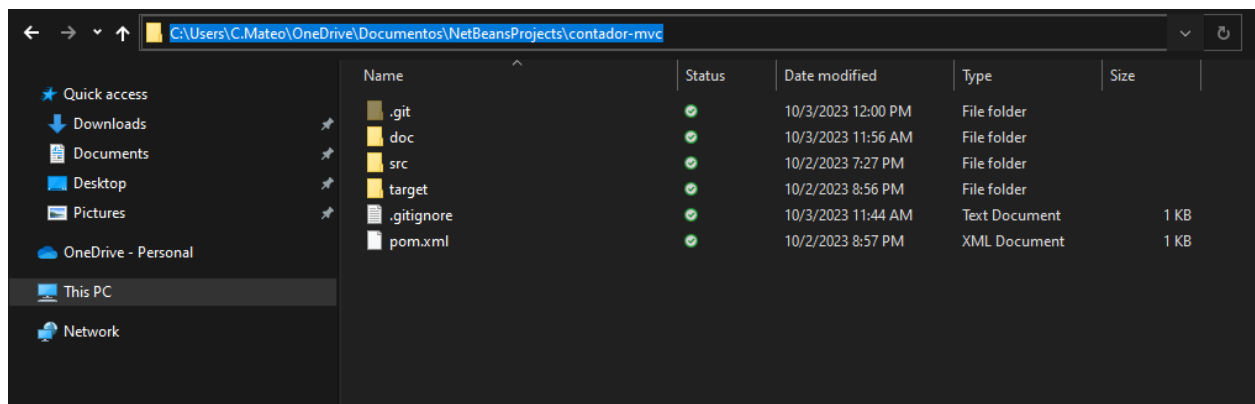


Trabajo colaborativo con Git y Github.

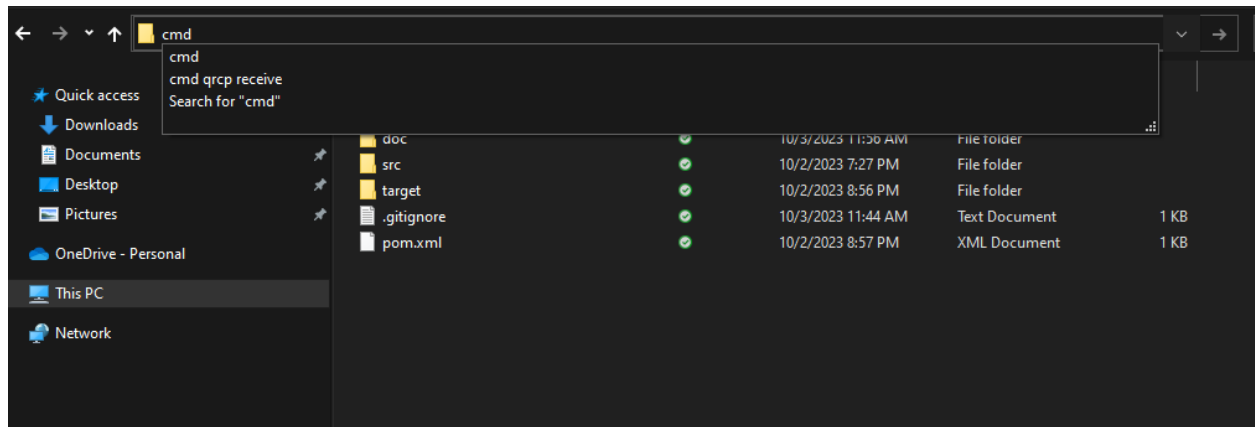
Creación de ramas



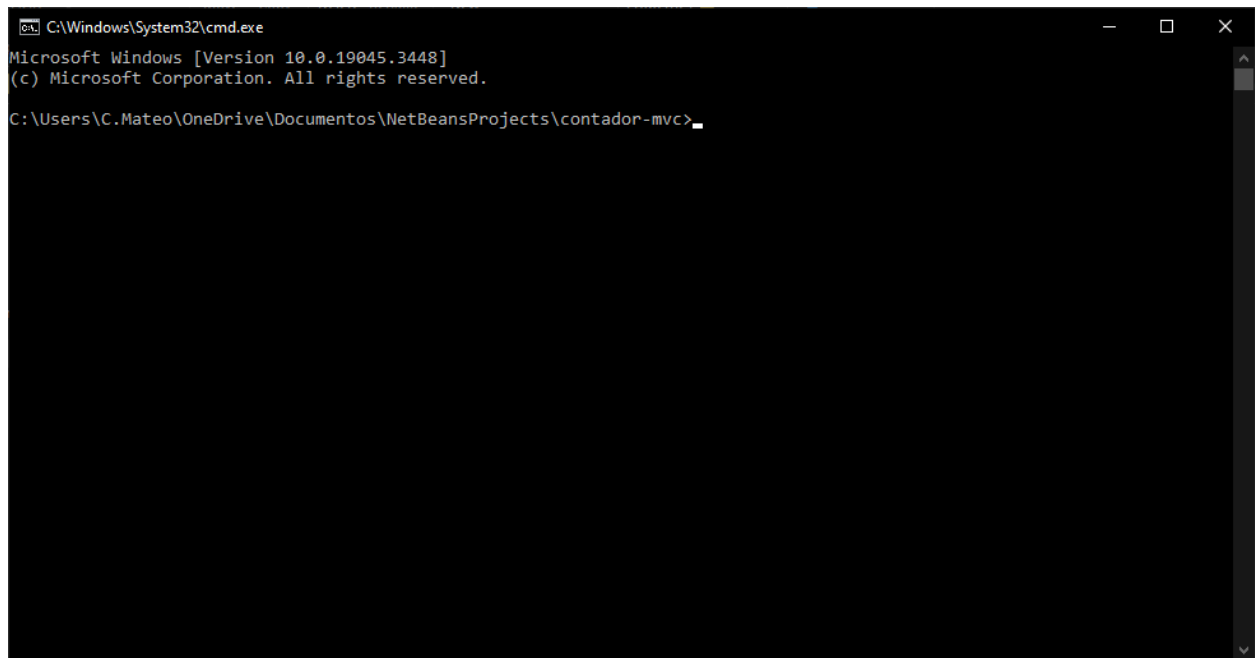
Abrimos el explorador de archivo y nos ubicamos en la carpeta raíz del proyecto.



Seleccionamos la barra de direcciones y eliminamos la ruta.



Escribimos “cmd” y oprimimos ENTER.



A continuación se abre una consola de windows en el directorio raíz del proyecto.

```
C:\Windows\System32\cmd.exe

C:\Users\C.Mateo\OneDrive\Documentos\NetBeansProjects\contador-mvc>git branch mateo

C:\Users\C.Mateo\OneDrive\Documentos\NetBeansProjects\contador-mvc>git branch
dev
* main
  mateo

C:\Users\C.Mateo\OneDrive\Documentos\NetBeansProjects\contador-mvc>git checkout mateo
Switched to branch 'mateo'

C:\Users\C.Mateo\OneDrive\Documentos\NetBeansProjects\contador-mvc>git branch
dev
  main
* mateo

C:\Users\C.Mateo\OneDrive\Documentos\NetBeansProjects\contador-mvc>
```

Escribimos los siguientes comando:

git branch <nombre-rama>

El comando `git branch <nombre-rama>` permite crear una rama (branch) en el proyecto. Las ramas nos permiten tener versiones paralelas de desarrollo, en la que podemos agregar nuevas característica, modificar existentes, eliminarlas o corregir errores del proyecto. Cada rama es independiente y trabaja sobre su propio historial de cambios, esto permite colaborar en el proyecto sin afectar el trabajo de otros miembros del equipo de desarrollo.

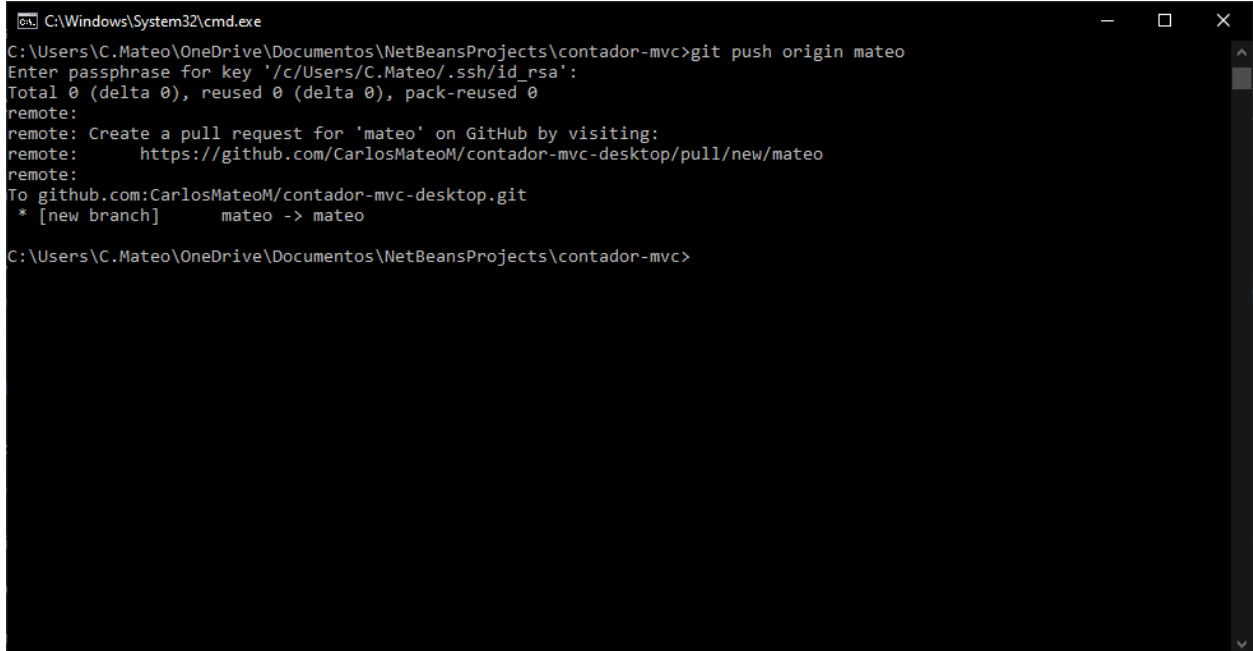
git branch

Este comando lista las ramas del proyecto. En este ejemplo se ha ejecutado para confirmar que se ha creado correctamente la nueva rama, podemos que existen 3:

- **main** : La rama 'main' es la rama principal del proyecto, donde reside la versión más estable, revisada y probada. Los cambios realizados en las diferentes ramas creadas a lo largo del proyecto se integran finalmente en la rama principal.
- **dev** : La rama 'dev' es la rama previa a la rama principal 'main'. Aquí se integran las nuevas modificaciones o correcciones con el propósito de revisarlas y aprobarlas antes de pasarlas a la rama principal. Funciona como un entorno de prueba para el proyecto.
- **mateo**: La rama 'mateo' se ha creado con el fin de ejemplificar la creación de ramas en git.

git checkout <nombre-rama>

El comando 'git checkout <nombre-rama>' permite cambiar de una rama a otra en git.

A screenshot of a Windows command prompt window titled 'C:\Windows\System32\cmd.exe'. The window shows the execution of the command 'git push origin mateo' in a directory 'C:\Users\C.Mateo\OneDrive\Documentos\NetBeansProjects\contador-mvc'. The output shows a successful push to the 'mateo' branch on the remote repository 'github.com:CarlosMateoM/contador-mvc-desktop.git'. It also prompts for a passphrase for an SSH key and provides a URL to create a pull request on GitHub. The prompt ends with 'C:\Users\C.Mateo\OneDrive\Documentos\NetBeansProjects\contador-mvc>'.

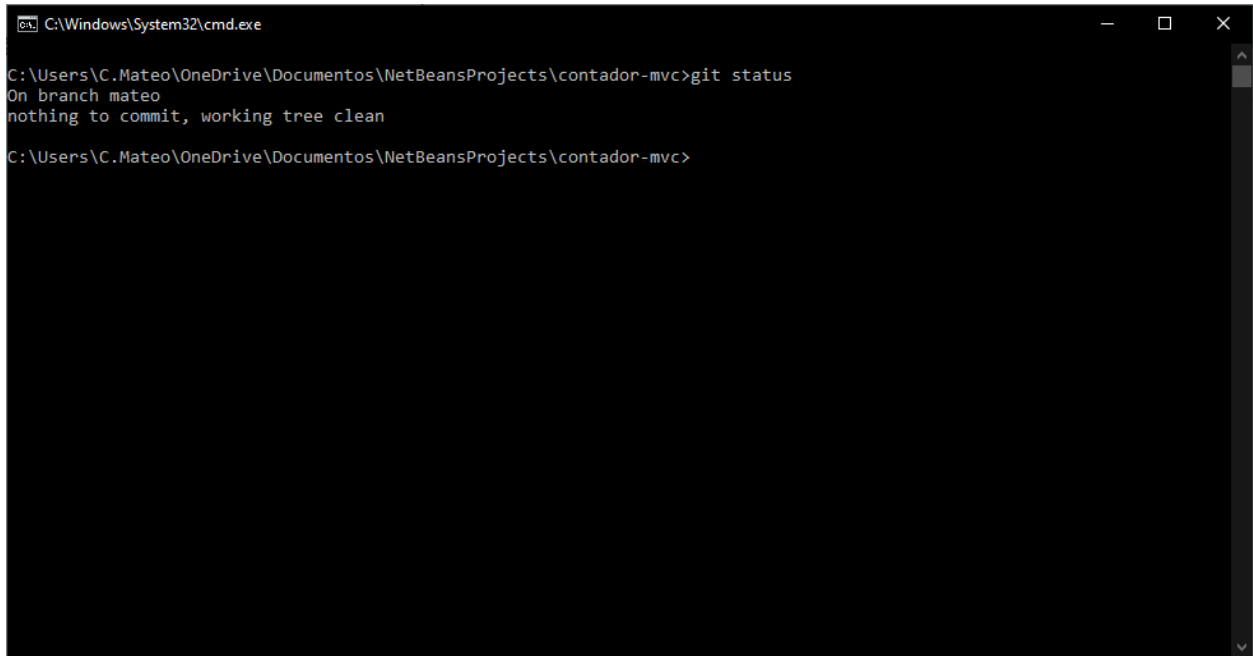
```
C:\Windows\System32\cmd.exe
C:\Users\C.Mateo\OneDrive\Documentos\NetBeansProjects\contador-mvc>git push origin mateo
Enter passphrase for key '/c/Users/C.Mateo/.ssh/id_rsa':
Total 0 (delta 0), reused 0 (delta 0), pack-reused 0
remote:
remote: Create a pull request for 'mateo' on GitHub by visiting:
remote:   https://github.com/CarlosMateoM/contador-mvc-desktop/pull/new/mateo
remote:
To github.com:CarlosMateoM/contador-mvc-desktop.git
 * [new branch]      mateo -> mateo
C:\Users\C.Mateo\OneDrive\Documentos\NetBeansProjects\contador-mvc>
```

git push origin <nombre-rama>

El comando git push origin <nombre-rama> se utiliza para subir una rama específica desde tu repositorio local al repositorio remoto "origin" en Git. Esto es útil cuando deseas compartir una rama que has creado localmente con otros colaboradores o almacenar una copia de esa rama en el repositorio remoto.

Con este último comando la rama está lista para trabajar en las nuevas características o modificaciones que se desea hacer en el proyecto.

Primer commit en la rama de trabajo

A screenshot of a Windows Command Prompt window. The title bar at the top reads 'C:\Windows\System32\cmd.exe'. The command prompt shows the following text:

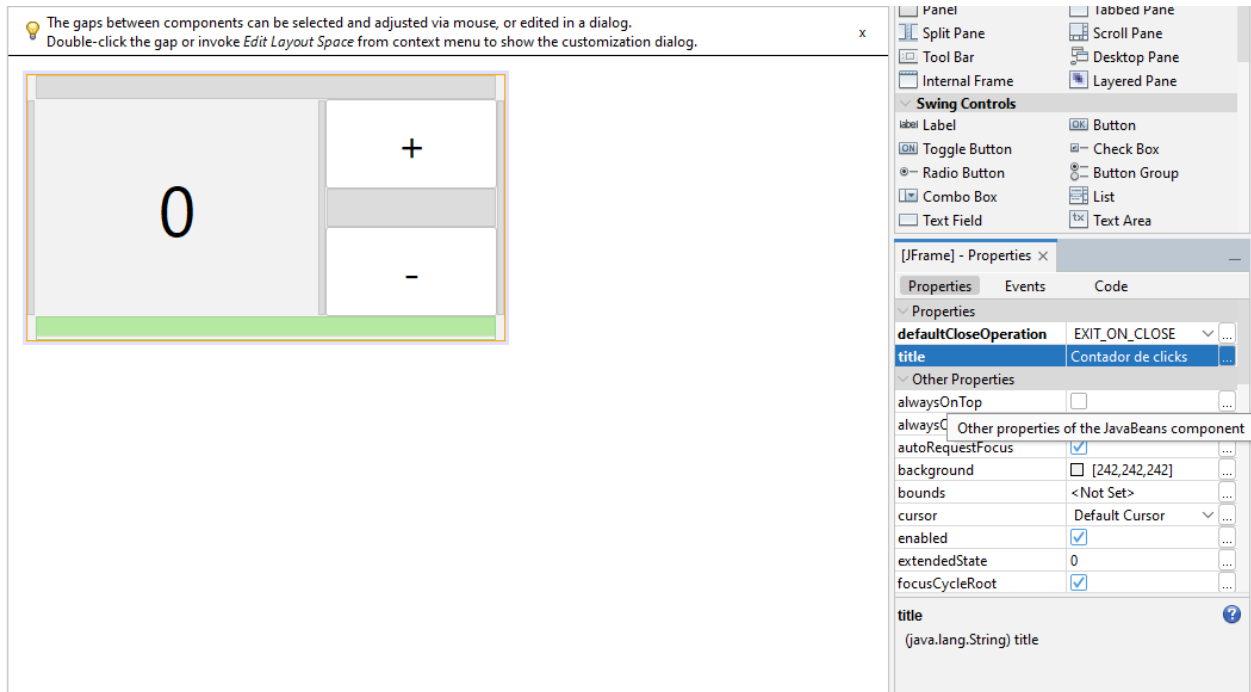
```
C:\Users\C.Mateo\OneDrive\Documentos\NetBeansProjects\contador-mvc>git status
On branch mateo
nothing to commit, working tree clean

C:\Users\C.Mateo\OneDrive\Documentos\NetBeansProjects\contador-mvc>
```

git status

El comando `git status` es una herramienta para obtener información sobre el estado actual de tu repositorio local. Cuando ejecutas `git status`, Git te proporciona detalles sobre los archivos en tu directorio de trabajo y su relación con el sistema de control de versiones.

Esta es la rama recién creada, no hemos realizado ningún cambio sobre los archivos de nuestro repositorio, por lo que la salida del comando anterior dice que no hay cambios por comprometer.

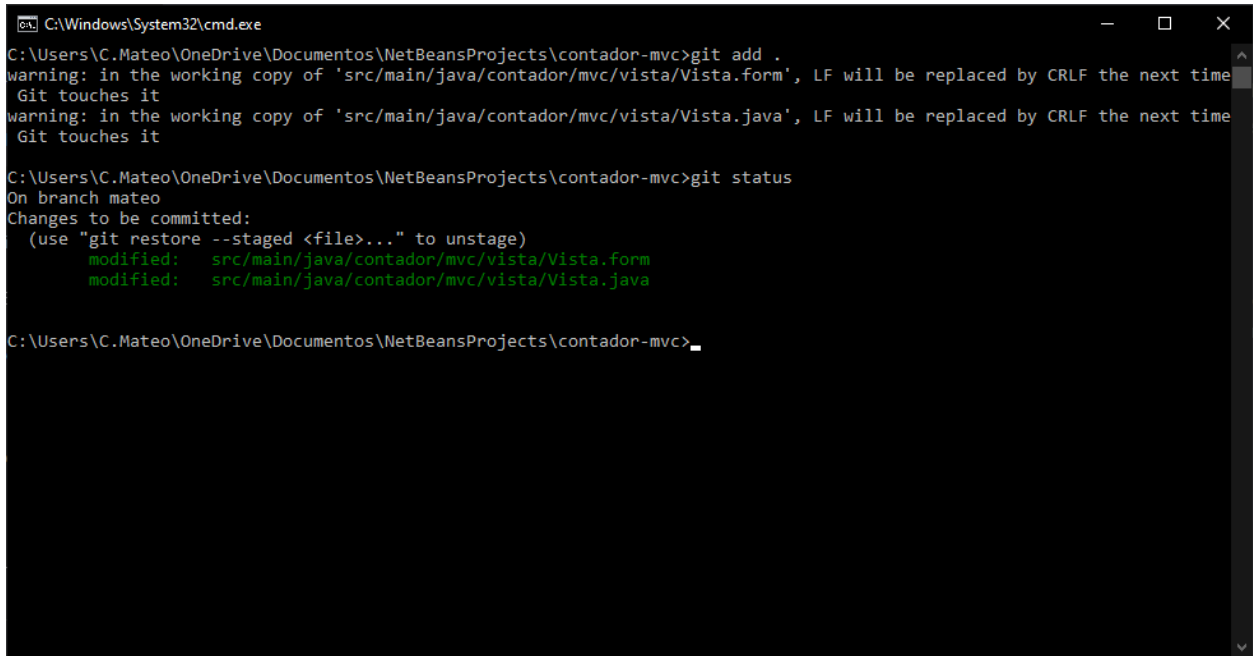


A continuación, modificaremos el título de JFrame del proyecto de “Contador” a “Contador de clicks”

```
C:\Windows\System32\cmd.exe

C:\Users\C.Mateo\OneDrive\Documentos\NetBeansProjects\contador-mvc>git status
On branch mateo
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   src/main/java/contador/mvc/vista/Vista.form
        modified:   src/main/java/contador/mvc/vista/Vista.java
no changes added to commit (use "git add" and/or "git commit -a")
C:\Users\C.Mateo\OneDrive\Documentos\NetBeansProjects\contador-mvc>
```

Ejecutamos el comando 'git status' el cual nos notifica que hemos modificado los archivos 'vista.form' y 'vista.java'.

A screenshot of a Windows command prompt window titled 'C:\Windows\System32\cmd.exe'. The window shows the execution of 'git add .' followed by two warnings about LF to CRLF conversion for 'Vista.form' and 'Vista.java'. Then 'git status' is run, showing the current branch as 'mateo' and listing two modified files: 'src/main/java/contador/mvc/vista/Vista.form' and 'src/main/java/contador/mvc/vista/Vista.java'. The prompt ends with 'C:\Users\C.Mateo\OneDrive\Documentos\NetBeansProjects\contador-mvc>'.

```
C:\Windows\System32\cmd.exe
C:\Users\C.Mateo\OneDrive\Documentos\NetBeansProjects\contador-mvc>git add .
warning: in the working copy of 'src/main/java/contador/mvc/vista/Vista.form', LF will be replaced by CRLF the next time
  Git touches it
warning: in the working copy of 'src/main/java/contador/mvc/vista/Vista.java', LF will be replaced by CRLF the next time
  Git touches it

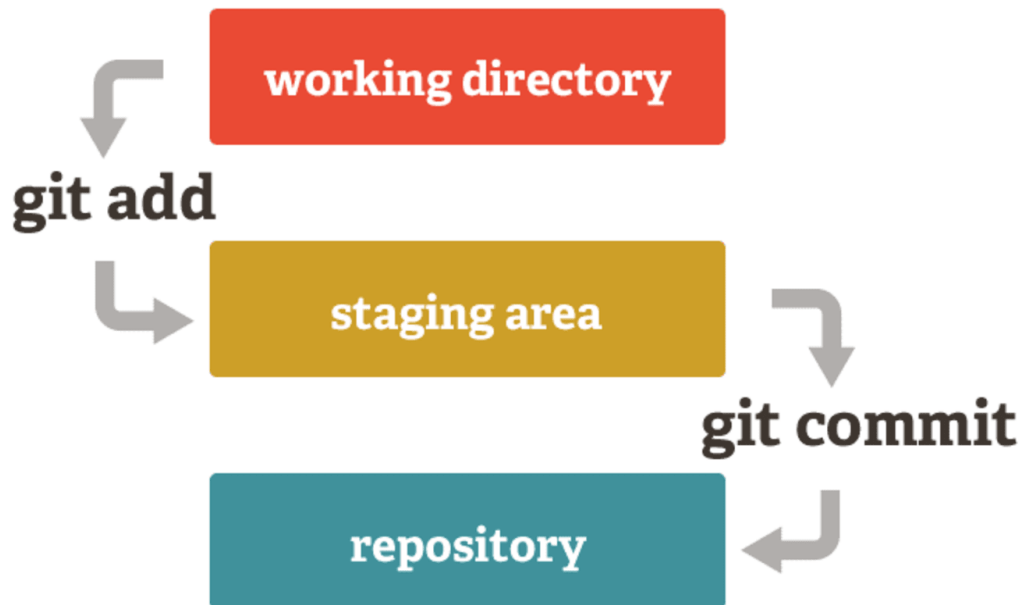
C:\Users\C.Mateo\OneDrive\Documentos\NetBeansProjects\contador-mvc>git status
On branch mateo
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        modified:   src/main/java/contador/mvc/vista/Vista.form
        modified:   src/main/java/contador/mvc/vista/Vista.java

C:\Users\C.Mateo\OneDrive\Documentos\NetBeansProjects\contador-mvc>
```

git add .

El comando git add . se utiliza para agregar todos los archivos y cambios en el directorio de trabajo al área de preparación (staging area) de Git. Esto significa que todos los archivos modificados y nuevos archivos que aún no se han confirmado se incluirán en el próximo commit que realices

Staging area



El "staging area" (área de preparación) en Git es una parte fundamental del flujo de trabajo de control de versiones. Es un espacio intermedio donde se almacenan temporalmente los cambios antes de que se confirmen en una nueva versión (commit) del proyecto. Esta área permite a los usuarios seleccionar y revisar los cambios que se incluirán en el próximo commit, lo que proporciona un alto grado de control sobre el historial del proyecto. El "staging area" facilita la separación y revisión de los cambios antes de confirmarlos, lo que es beneficioso para mantener un historial de versiones organizado y lógico en proyectos de desarrollo de software y colaborativos.

Al ejecutar nuevamente el comando 'git status' muestra los archivos modificados resaltados en verde para indicar que se encuentran en el 'staging area' listos para ser agregados en el próximo commit al historial de versiones de git.


```
C:\Windows\System32\cmd.exe
C:\Users\C.Mateo\OneDrive\Documentos\NetBeansProjects\contador-mvc>git add .
warning: in the working copy of 'src/main/java/contador/mvc/vista/Vista.form', LF will be replaced by CRLF the next time
Git touches it
warning: in the working copy of 'src/main/java/contador/mvc/vista/Vista.java', LF will be replaced by CRLF the next time
Git touches it

C:\Users\C.Mateo\OneDrive\Documentos\NetBeansProjects\contador-mvc>git status
On branch mateo
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        modified:   src/main/java/contador/mvc/vista/Vista.form
        modified:   src/main/java/contador/mvc/vista/Vista.java

C:\Users\C.Mateo\OneDrive\Documentos\NetBeansProjects\contador-mvc>git commit -m "cambio titulo jframe"
[mateo 2fb7a4a] cambio titulo jframe
 2 files changed, 2 insertions(+), 2 deletions(-)

C:\Users\C.Mateo\OneDrive\Documentos\NetBeansProjects\contador-mvc>
```

git commit -m “mensaje confirmación”

El comando `git commit -m "mensaje"` se utiliza para crear una nueva confirmación (commit) en Git con un mensaje de confirmación asociado. La confirmación representa un punto en el historial de versiones de tu proyecto y almacena un conjunto de cambios específicos. El mensaje de confirmación te permite describir brevemente los cambios realizados en esa confirmación.

```
C:\Windows\System32\cmd.exe
warning: in the working copy of 'src/main/java/contador/mvc/vista/Vista.form', LF will be replaced by CRLF the next time
Git touches it
warning: in the working copy of 'src/main/java/contador/mvc/vista/Vista.java', LF will be replaced by CRLF the next time
Git touches it

C:\Users\C.Mateo\OneDrive\Documentos\NetBeansProjects\contador-mvc>git status
On branch mateo
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        modified:   src/main/java/contador/mvc/vista/Vista.form
        modified:   src/main/java/contador/mvc/vista/Vista.java

C:\Users\C.Mateo\OneDrive\Documentos\NetBeansProjects\contador-mvc>git commit -m "cambio titulo jframe"
[mateo 2fb7a4a] cambio titulo jframe
 2 files changed, 2 insertions(+), 2 deletions(-)

C:\Users\C.Mateo\OneDrive\Documentos\NetBeansProjects\contador-mvc>git push origin mateo
Enter passphrase for key '/c/Users/C.Mateo/.ssh/id_rsa':
Enumerating objects: 19, done.
Counting objects: 100% (19/19), done.
Delta compression using up to 2 threads
Compressing objects: 100% (6/6), done.
Writing objects: 100% (10/10), 654 bytes | 327.00 KiB/s, done.
Total 10 (delta 4), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (4/4), completed with 4 local objects.
To github.com:CarlosMateoM/contador-mvc-desktop.git
   a57cce0..2fb7a4a  mateo -> mateo

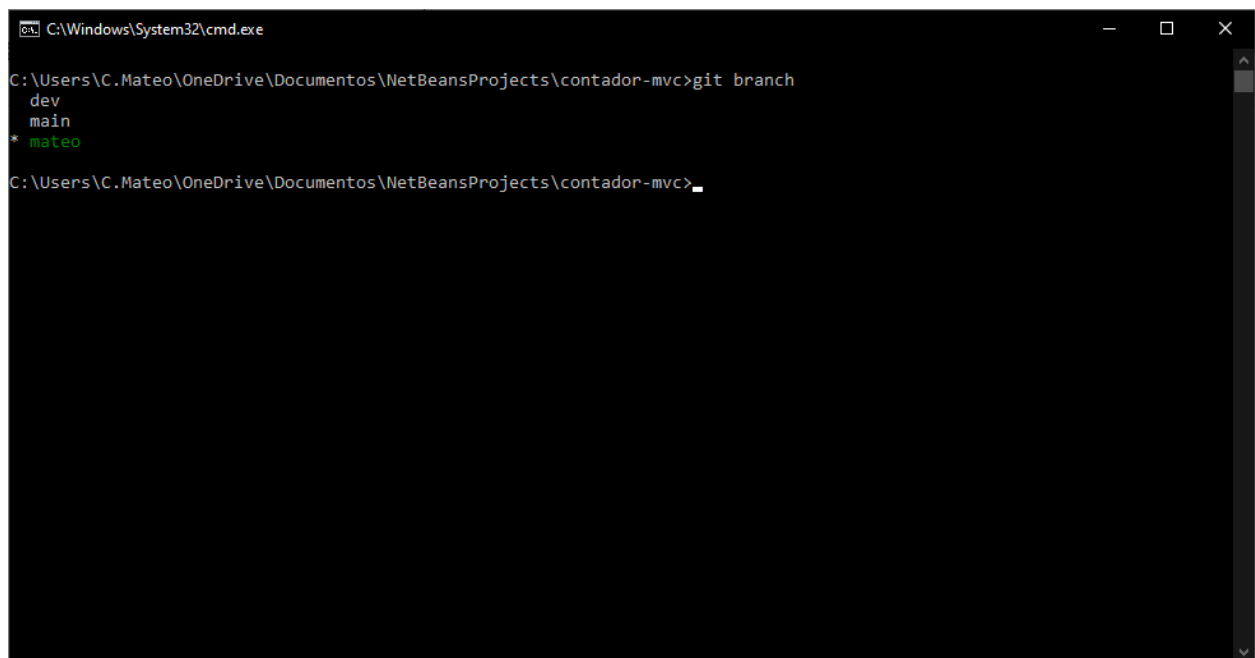
C:\Users\C.Mateo\OneDrive\Documentos\NetBeansProjects\contador-mvc>
```

git push origin <rama>

El comando `git push origin <rama>` se utiliza para enviar (push) una rama específica desde tu repositorio local al repositorio remoto llamado "origin" en Git.

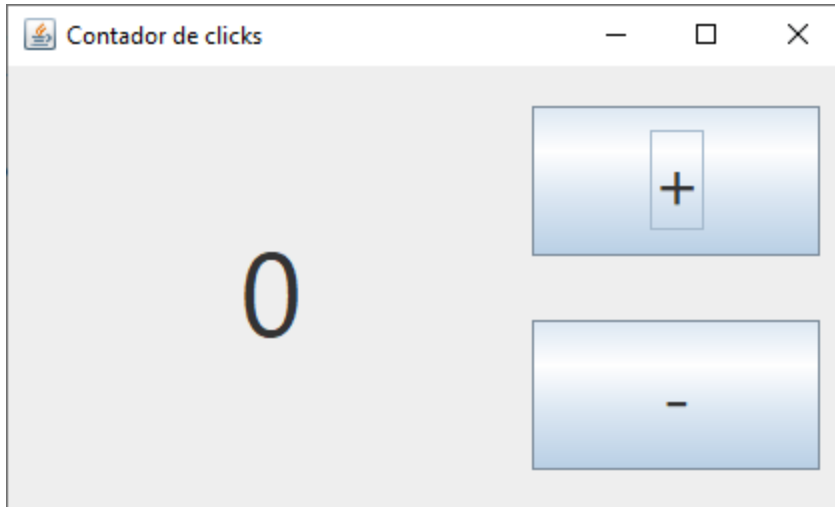
En ese caso hemos enviado nuestros cambios a la rama 'mateo', la cual hemos creado para este ejemplo.

Cambiando de ramas.



```
C:\Windows\System32\cmd.exe
C:\Users\C.Mateo\OneDrive\Documentos\NetBeansProjects\contador-mvc>git branch
dev
main
* mateo
C:\Users\C.Mateo\OneDrive\Documentos\NetBeansProjects\contador-mvc>
```

Con el comando 'git branch' se verifica en qué rama del proyecto se está situado. Hasta este punto se encuentra en la rama 'mateo', en la cual se ha trabajado y se ha modificado el proyecto.



Se puede observar que el JFrame conserva el título 'Contador de clicks'.

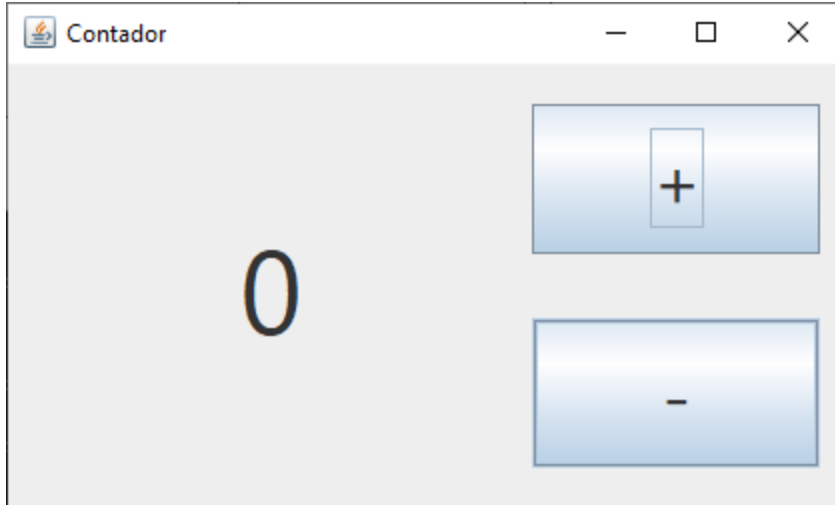
```
C:\Windows\System32\cmd.exe

C:\Users\C.Mateo\OneDrive\Documentos\NetBeansProjects\contador-mvc>git branch
  dev
  main
* mateo

C:\Users\C.Mateo\OneDrive\Documentos\NetBeansProjects\contador-mvc>git checkout dev
Switched to branch 'dev'

C:\Users\C.Mateo\OneDrive\Documentos\NetBeansProjects\contador-mvc>
```

'git checkout dev' cambia la rama actual 'mateo' a 'dev', en esta última no se ha modificado el proyecto, por lo que es posible verificar si aún tiene el antiguo título.



Cuando ejecutas el programa, puedes notar que la ventana (JFrame) tiene un título diferente al que se encuentra en la rama 'mateo'. Esto ocurre porque hemos cambiado de rama y, por lo tanto, de versión en el código. Cada rama representa una línea de desarrollo independiente, y en nuestro caso, tenemos una rama previa a la principal llamada 'dev'. Por lo tanto, los cambios que hacemos en una rama, como 'mateo', pueden ser distintos de los de otras ramas.

En el futuro, cuando hayamos implementado nuevas características y las hayamos revisado y probado en sus respectivas ramas (como 'mateo' y 'dev'), estas ramas se fusionarán. La fusión significa que los cambios de una rama se incorporarán en otra, en este caso, en la rama principal. Una vez que hayamos fusionado las características probadas en la rama 'mateo' con la rama 'dev' y finalmente con la rama principal, habremos completado el ciclo de vida de la rama 'mateo'.

En resumen, el uso de ramas nos permite trabajar en diferentes versiones de nuestro proyecto de forma independiente, lo que puede resultar en diferencias en el comportamiento, como el título de la ventana en este caso. Sin embargo, al final, fusionaremos las mejoras y cambios en la rama principal, pasando por la rama 'dev', para tener una versión unificada y actualizada de nuestro proyecto.