

Conceptos Básicos Git Hub

Juan José Gracia Gómez

Ingeniería de Sistemas Y computación

Universidad de Cundinamarca

Profesor: William Alexander Matallana Porras

Chía

2025

Tabla de Contenido.

Introducción..	3
Objetivo.....	4
Desarrollo.....	5
Conclusión..	18
Uso de Inteligencia Artificial.....	18
Referencias.....	19

Introducción.

En el desarrollo de este proyecto, comenzaremos creando un repositorio en GitHub y un proyecto en IntelliJ. Luego, inicializaremos el repositorio localmente y realizaremos nuestro primer commit para sincronizarlo con GitHub. Durante el desarrollo, iremos realizando cambios en el código, creando ramas para nuevas características y subiéndolas al repositorio remoto. Si es necesario, revertiremos ciertos cambios para corregir errores. Al final, fusionaremos las ramas, eliminaremos las que ya no sean necesarias y nos aseguraremos de que el proyecto esté completamente actualizado y listo para su finalización.

Objetivo.

Aprender y familiarizarse con los comandos y el flujo de trabajo de Git y GitHub, a través de la creación y gestión de un repositorio, la realización de cambios en el código, la utilización de ramas, y la sincronización de los cambios entre el repositorio local y remoto.

Desarrollo.

Lo primero es crear una carpeta.



Ahora vamos a desasociar la cuenta y el usuario:

```
PS C:\Users\Juan José\Desktop\Miproyecto> git config --global --unset user.email jjgracia@ucundinamarca.edu.co
PS C:\Users\Juan José\Desktop\Miproyecto> git config --unset user.name JuanJGr
```

```
PS C:\Users\Juan José\Desktop\Miproyecto> git config list
diff.astextplain.textconv=astextplain
filter.lfs.clean=git-lfs clean -- %f
filter.lfs.smudge=git-lfs smudge -- %f
filter.lfs.process=git-lfs filter-process
filter.lfs.required=true
http.sslbackend=schannel
core.autocrlf=true
core.fscache=true
core.symlinks=false
pull.rebase=false
credential.helper=manager
credential.https://dev.azure.com.usehttppath=true
init.defaultbranch=master
core.autocrlf=true
core.repositoryformatversion=0
core.filemode=false
core.bare=false
core.logallrefupdates=true
core.symlinks=false
core.ignorecase=true
```

git config --global --unset user.name: Elimina el nombre de usuario global configurado en Git.

git config --global --unset user.email: Elimina el correo electrónico global configurado en Git.

De nuevo volvemos a vincular el correo y el nombre de usuario.

```
PS C:\Users\Juan José\Desktop\Miproyecto> git config --global user.email "jjgracia@ucundinamarca.edu.co"
PS C:\Users\Juan José\Desktop\Miproyecto> git config --global user.name "JuanJGr"
```

git config --global user.name "Tu Nombre": Configura el nombre de usuario en Git a nivel global, es decir, para todos los repositorios en tu sistema.

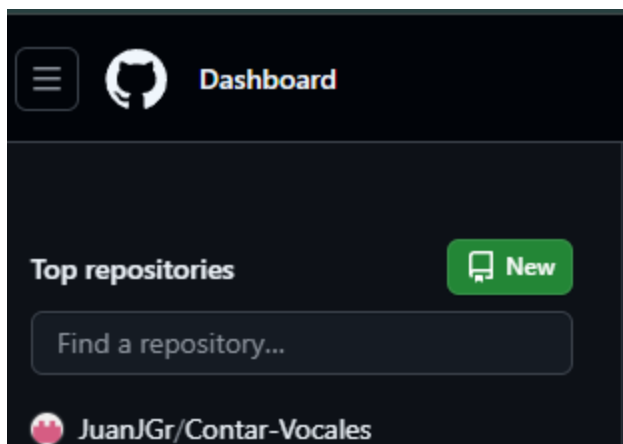
git config --global user.email "tu.email@dominio.com": Configura el correo electrónico en Git a nivel global, también para todos los repositorios en el sistema.

En la terminal de la carpeta escribimos `git config list`:

```
PS C:\Users\Juan José\Desktop\Miproyecto> git config list
diff.astextplain.textconv=astextplain
filter.lfs.clean=git-lfs clean -- %f
filter.lfs.smudge=git-lfs smudge -- %f
filter.lfs.process=git-lfs filter-process
filter.lfs.required=true
http.sslbackend=schannel
core.autocrlf=true
core.fscache=true
core.symlinks=false
pull.rebase=false
credential.helper=manager
credential.https://dev.azure.com.usehttppath=true
init.defaultbranch=master
user.name=JuanJGr
user.email=jjgracia@ucundinamrca.edu.co
core.autocrlf=true
core.repositoryformatversion=0
core.filemode=false
core.bare=false
core.logallrefupdates=true
core.symlinks=false
core.ignorecase=true
remote.origin.url=https://github.com/JuanJGr/Ejercicio-Nombre.git
remote.origin.fetch=+refs/heads/*:refs/remotes/origin/*
PS C:\Users\Juan José\Desktop\Miproyecto> |
```

git config --list: Se usa para listar todas las configuraciones actuales de Git en el sistema.

Creamos un repositorio en nuestra cuenta de GitHub.



Copiamos y pegamos en la terminal del proyecto lo que nos aparece en el repositorio de GitHub.

```
PS C:\Users\Juan José\Desktop\Mi Proyecto\Contar Vocales> echo "# Contar-Vocales" >> README.md
PS C:\Users\Juan José\Desktop\Mi Proyecto\Contar Vocales> git init
Initialized empty Git repository in C:/Users/Juan José/Desktop/Mi Proyecto/Contar Vocales/.git/
PS C:\Users\Juan José\Desktop\Mi Proyecto\Contar Vocales> git add README.md
PS C:\Users\Juan José\Desktop\Mi Proyecto\Contar Vocales> git commit -m "first commit"
[master (root-commit) f4f8002] first commit
1 file changed, 0 insertions(+), 0 deletions(-)

create mode 100644 README.md
PS C:\Users\Juan José\Desktop\Mi Proyecto\Contar Vocales> git branch -M main
PS C:\Users\Juan José\Desktop\Mi Proyecto\Contar Vocales> git remote add origin https://github.com/JuanJGr/Contar-Vocales.git
PS C:\Users\Juan José\Desktop\Mi Proyecto\Contar Vocales> git push -u origin main
Enumerating objects: 3, done.
Counting objects: 100% (3/3), done.
Writing objects: 100% (3/3), 251 bytes | 251.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
To https://github.com/JuanJGr/Contar-Vocales.git
```

Como podemos observar anteriormente hay varios comandos los cuales sirven para:

echo "# Contar-Vocales" >> README.md: Crea un archivo README.md y agrega el texto "# Contar-Vocales".

git init: Inicializa un nuevo repositorio Git en el proyecto.

git add README.md: Agrega el archivo README.md al área de preparación (staging) para el commit.

git commit -m "first commit": Crea un commit con el mensaje "first commit" que guarda el archivo en el repositorio.

git branch -M main: Renombra la rama actual a main (cambia de master a main).

`git remote add origin https://github.com/JuanJGr/Contar-Vocales.git`: Conecta el repositorio local a GitHub usando la URL del repositorio.

git push -u origin main: Sube tu rama main al repositorio remoto en GitHub.

Luego hacemos los primeros cambios al código principal y los vamos a subir desde el repositorio local al remoto en GitHub.

```
PS C:\Users\Juan José\Desktop\Mi Proyecto\Contar Vocales> git status
On branch main
Your branch is up to date with 'origin/main'.

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        .gitignore
        .idea/
        Contar Vocales.iml
        src/
```

git status: “Muestra el estado del directorio en el que estás trabajando y la instantánea preparada.” Atlassian. (s. f.).

```
PS C:\Users\Juan José\Desktop\Mi Proyecto\Contar Vocales> git add .
warning: in the working copy of '.gitignore', LF will be replaced by CRLF the next time Git touches it
warning: in the working copy of 'src/Main.java', LF will be replaced by CRLF the next time Git touches it
```

```
PS C:\Users\Juan José\Desktop\Mi Proyecto\Contar Vocales> git commit -m "Primeras Líneas De Código."
[main 30ff6f5] Primeras Líneas De Código.
7 files changed, 84 insertions(+)
create mode 100644 .gitignore
create mode 100644 .idea/.gitignore
create mode 100644 .idea/misc.xml
create mode 100644 .idea/modules.xml
create mode 100644 .idea/vcs.xml
create mode 100644 Contar Vocales.iml
create mode 100644 src/Main.java
PS C:\Users\Juan José\Desktop\Mi Proyecto\Contar Vocales> git push origin main
Enumerating objects: 12, done.
Counting objects: 100% (12/12), done.
Delta compression using up to 6 threads
```

git add .: “Mueve los cambios del directorio de trabajo al área del entorno de ensayo” Atlassian. (s. f.).

git commit -m: Crea un commit con un mensaje que describe los cambios realizados.


```
PS C:\Users\Juan José\Desktop\Mi Proyecto\Contar Vocales> git push origin main
Enumerating objects: 12, done.
Counting objects: 100% (12/12), done.
Delta compression using up to 6 threads
Compressing objects: 100% (9/9), done.
Writing objects: 100% (11/11), 1.84 KiB | 627.00 KiB/s, done.
Total 11 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
To https://github.com/JuanJGr/Contar-Vocales.git
f4f8002..30ff6f5  main -> main
```

git push origin main: Sube los cambios locales de la rama main al repositorio remoto en GitHub.

Aquí podemos ver cómo se subieron correctamente los cambios desde el local al remoto.

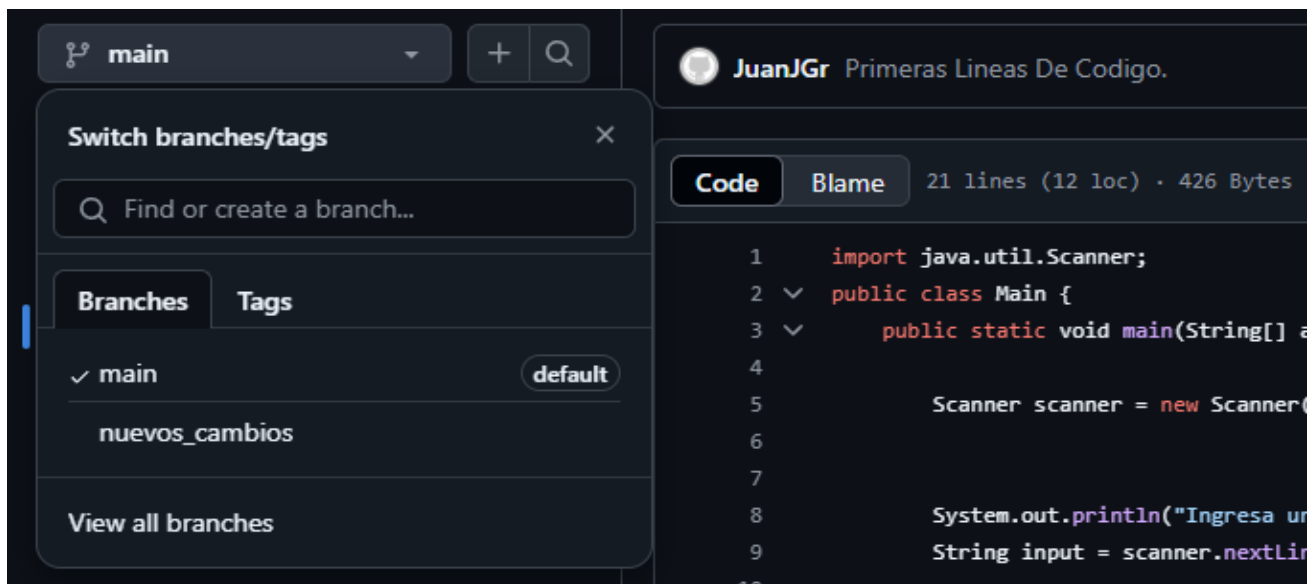
Si queremos tener los últimos cambios de alguna rama del remoto:

```
Juan José@LAPTOP-A0S7PFEG MINGW64 ~/Desktop/Mi Proyecto/Contar Vocales (main)
$ git pull origin main
From https://github.com/JuanJGr/Contar-Vocales
 * branch          main          -> FETCH_HEAD
Already up to date.
```

git pull origin main: sirve para obtener los últimos cambios de la rama main en el repositorio remoto (origin) y fusionarlos con la rama local actual.

Ahora creamos una nueva rama donde se añadirán algunas cosas al código.

```
PS C:\Users\Juan José\Desktop\Mi Proyecto\Contar Vocales> git branch
* main
PS C:\Users\Juan José\Desktop\Mi Proyecto\Contar Vocales> git switch -C "nuevos_cambios"
Switched to a new branch 'nuevos_cambios'
PS C:\Users\Juan José\Desktop\Mi Proyecto\Contar Vocales> git push origin "nuevos_cambios"
Total 0 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
remote:
remote: Create a pull request for 'nuevos_cambios' on GitHub by visiting:
remote:   https://github.com/JuanJGr/Contar-Vocales/pull/new/nuevos\_cambios
remote:
To https://github.com/JuanJGr/Contar-Vocales.git
* [new branch]   nuevos_cambios -> nuevos_cambios
```



git branch: Muestra las ramas actuales en el repositorio. El * indica que estás en la rama main.

git switch -C "nuevos_cambios": Crea una nueva rama llamada nuevos_cambios y me cambia a esa rama.

git push origin "nuevos_cambios": Sube la rama nuevos_cambios al repositorio remoto en GitHub.

Si queremos cambiar de rama:

```
Juan José@LAPTOP-A0S7PFEG MINGW64 ~/Desktop/Mi Proyecto/Contar Vocales (nuevos_cambios)
$ git switch main
Switched to branch 'main'
Your branch is up to date with 'origin/main'.
```

git switch rama: Se utiliza para cambiar a una rama existente en el repositorio local.

Para descargar actualizaciones del remoto:

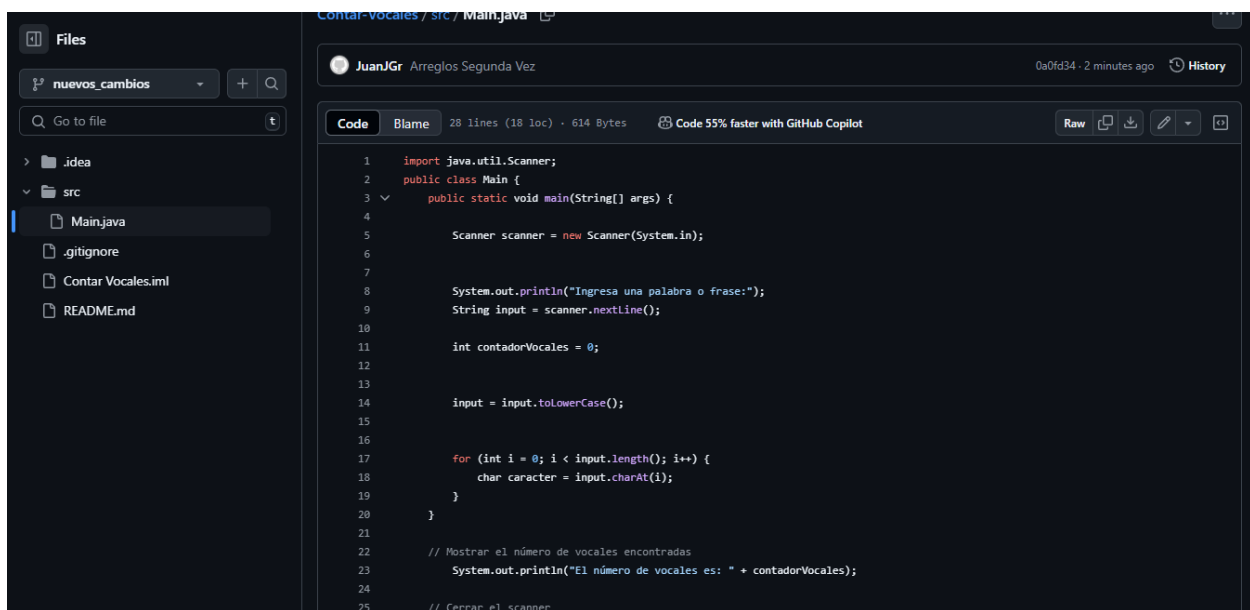
```
Juan José@LAPTOP-A0S7PFE6 MINGW64 ~/Desktop/Mi Proyecto/Contar Vocales (main)
$ git fetch --all
```

git fetch --all: Descarga las actualizaciones de todos los remotos configurados, sin fusionarlas con el trabajo actual.

Agregamos nuevas líneas de código y subimos estos cambios al remoto.

```
Terminal Local x
PS C:\Users\Juan José\Desktop\Mi Proyecto\Contar Vocales> git status
On branch nuevos_cambios
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

no changes added to commit (use "git add" and/or "git commit -a")
PS C:\Users\Juan José\Desktop\Mi Proyecto\Contar Vocales> git add .
warning: in the working copy of 'src/Main.java', LF will be replaced by CRLF the next time Git touches it
PS C:\Users\Juan José\Desktop\Mi Proyecto\Contar Vocales> git commit -m "Arreglos Segunda Vez"
[nuevos_cambios 0a0fd34] Arreglos Segunda Vez
1 file changed, 8 insertions(+), 1 deletion(-)
PS C:\Users\Juan José\Desktop\Mi Proyecto\Contar Vocales> git push origin nuevos_cambios
Enumerating objects: 7, done.
Counting objects: 100% (7/7), done.
Delta compression using up to 6 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (4/4), 451 bytes | 451.00 KiB/s, done.
Total 4 (delta 2), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (2/2), completed with 2 local objects.
To https://github.com/JuanJGr/Contar-Vocales.git
   30ff6f5..0a0fd34  nuevos_cambios -> nuevos_cambios
```



```
Contar-vocales / src / Main.java
JuanJGr Arreglos Segunda Vez 0a0fd34 - 2 minutes ago History

Code Blame 28 lines (18 loc) · 614 Bytes Code 55% faster with GitHub Copilot Raw

1 import java.util.Scanner;
2 public class Main {
3     public static void main(String[] args) {
4
5         Scanner scanner = new Scanner(System.in);
6
7
8         System.out.println("Ingresa una palabra o frase:");
9         String input = scanner.nextLine();
10
11         int contadorVocales = 0;
12
13
14         input = input.toLowerCase();
15
16
17         for (int i = 0; i < input.length(); i++) {
18             char character = input.charAt(i);
19         }
20     }
21
22     // Mostrar el número de vocales encontradas
23     System.out.println("El número de vocales es: " + contadorVocales);
24
25     // Cerrar el scanner
```

Ahora observamos que en el código hay un error por lo que tenemos que revertir los últimos cambios.

```
PS C:\Users\Juan José\Desktop\Mi Proyecto\Contar Vocales> git log
commit 0a0fd340538b93fd6b9e0da4bcb7e643de29b42e (HEAD -> nuevos_cambios, origin/nuevos_cambios)
Author: JuanJGr <jjgracia@ucundinamrca.edu.co>
Date: Thu Feb 20 07:36:06 2025 -0500

    Arreglos Segunda Vez

commit 30ff6f540982039a748acc59fce56a2a0a4f2fd4 (origin/main, main)
Author: JuanJGr <jjgracia@ucundinamrca.edu.co>
Date: Thu Feb 20 07:14:03 2025 -0500

    Primeras Lineas De Código.

commit f4f80025c7e667c47bf422580c27566aaa4c9aa9
Author: JuanJGr <jjgracia@ucundinamrca.edu.co>
Date: Thu Feb 20 06:56:35 2025 -0500

    first commit
PS C:\Users\Juan José\Desktop\Mi Proyecto\Contar Vocales> ^C
PS C:\Users\Juan José\Desktop\Mi Proyecto\Contar Vocales> git revert 0a0fd340538b93fd6b9e0da4bcb7e643de29b42e
[nuevos_cambios ee6a42c] Revert "Arreglos Segunda Vez"
1 file changed, 1 insertion(+), 8 deletions(-)
```

git log: “Permite explorar las revisiones anteriores de un proyecto. Proporciona varias opciones de formato para mostrar las instantáneas confirmadas.” Atlassian. (s. f.).

git revert 0a0fd340538b93fd6b9e0da4bcb7e643de29b42e: Este comando revierte los cambios realizados en el commit con el ID 0a0fd340538b93fd6b9e0da4bcb7e643de29b42e ("Arreglos Segunda Vez"). El comando crea un nuevo commit con el mensaje "Revert 'Arreglos Segunda Vez'". En este commit, se eliminan 8 líneas y se agrega 1 línea al archivo modificado.

Eliminamos la rama “nuevos_cambios”:

```
PS C:\Users\Juan José\Desktop\Mi Proyecto\Contar Vocales> git branch
  main
* nuevos_cambios
PS C:\Users\Juan José\Desktop\Mi Proyecto\Contar Vocales> git switch main
Switched to branch 'main'
Your branch is up to date with 'origin/main'.
PS C:\Users\Juan José\Desktop\Mi Proyecto\Contar Vocales> git branch
* main
  nuevos_cambios
PS C:\Users\Juan José\Desktop\Mi Proyecto\Contar Vocales> git branch -D "nuevos_cambios"
Deleted branch nuevos_cambios (was ee6a42c).
```

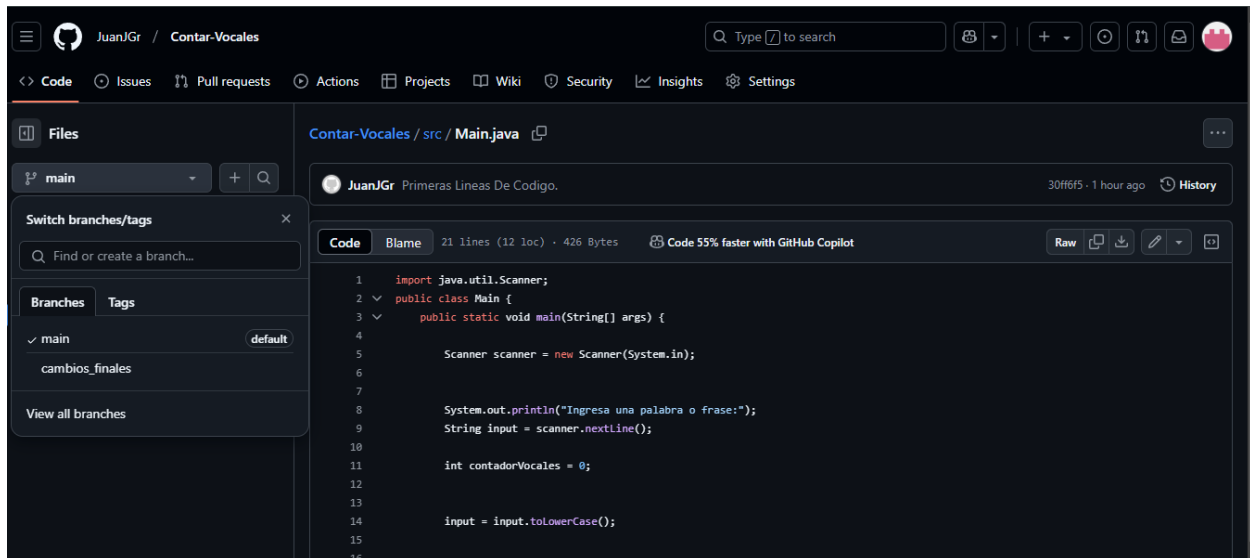
```
PS C:\Users\Juan José\Desktop\Mi Proyecto\Contar Vocales> git push origin --delete nuevos_cambios
To https://github.com/JuanJGr/Contar-Vocales.git
- [deleted]          nuevos_cambios
PS C:\Users\Juan José\Desktop\Mi Proyecto\Contar Vocales>
```

git branch -D “rama”: Elimina una rama local de forma forzada, incluso si no ha sido fusionada.

git push origin --delete <rama>: Elimina una rama remota.

Creamos una nueva rama llamada “cambios_finales” donde esta el código ya completo sin errores y todo esto lo subimos al remoto.

```
S C:\Users\Juan José\Desktop\Mi Proyecto\Contar Vocales> git add .
S C:\Users\Juan José\Desktop\Mi Proyecto\Contar Vocales> git commit -m "codigo_final"
cambios_finales da547ba] codigo_final
1 file changed, 17 insertions(+), 5 deletions(-)
S C:\Users\Juan José\Desktop\Mi Proyecto\Contar Vocales> git push origin cambios_finales
numerating objects: 7, done.
counting objects: 100% (7/7), done.
delta compression using up to 6 threads
compressing objects: 100% (3/3), done.
writing objects: 100% (4/4), 482 bytes | 482.00 KiB/s, done.
total 4 (delta 2), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (2/2), completed with 2 local objects.
To https://github.com/JuanJGr/Contar-Vocales.git
30ff6f5..da547ba  cambios_finales -> cambios_finales
```



Para ver las ramas en remoto:


```
Juan José@LAPTOP-A0S7PFE6 MINGW64 ~/Desktop/Mi Proyecto/Contar Vocales (main)
$ git branch -r
  origin/HEAD -> origin/main
  origin/cambios_finales
  origin/main
```

git branch -r: Muestra las ramas remotas de nuestro repositorio.

Fusionamos las dos ramas desde el GitHub.










Create a new pull request by comparing changes across two branches. If you need to, you can also [compare across forks](#). [Learn more about diff con](#)

base: main < compare: cambios_finales ✓ Able to merge. These branches can be automatically merged.



 Add a title

codigo_final


Add a description

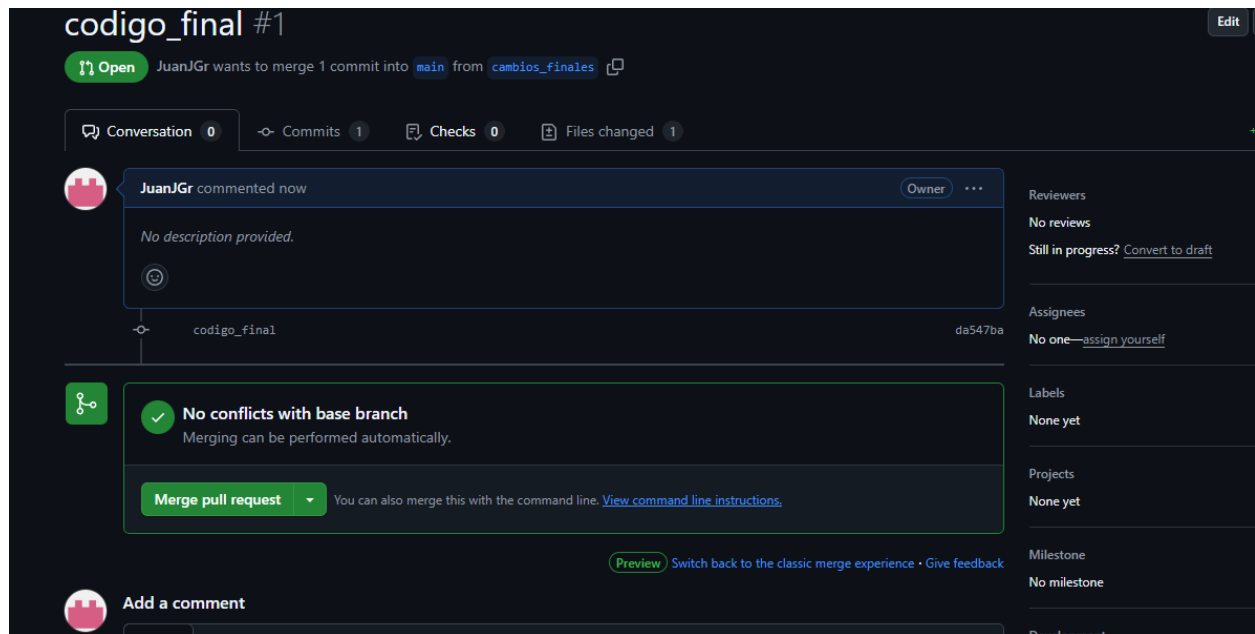
Write Preview H B I  <>  |    |  @   

Add your description here...

 Markdown is supported  Paste, drop, or click to add files

Create pull request

 Remember, contributions to this repository should follow our [GitHub Community Guidelines](#).



Para combinar ramas por comando, seria:

```
Juan José@LAPTOP-A0S7PFE6 MINGW64 ~/Desktop/Mi Proyecto/Contar Vocales (main)
$ git rebase nuevos_cambios
Current branch main is up to date.

Juan José@LAPTOP-A0S7PFE6 MINGW64 ~/Desktop/Mi Proyecto/Contar Vocales (main)
$ git merge nuevos_cambios
Already up to date.
```

git merge: Une dos ramas, creando un commit de fusión.

git rebase: Reaplica los cambios de una rama sobre otra, creando un historial más limpio lineal.

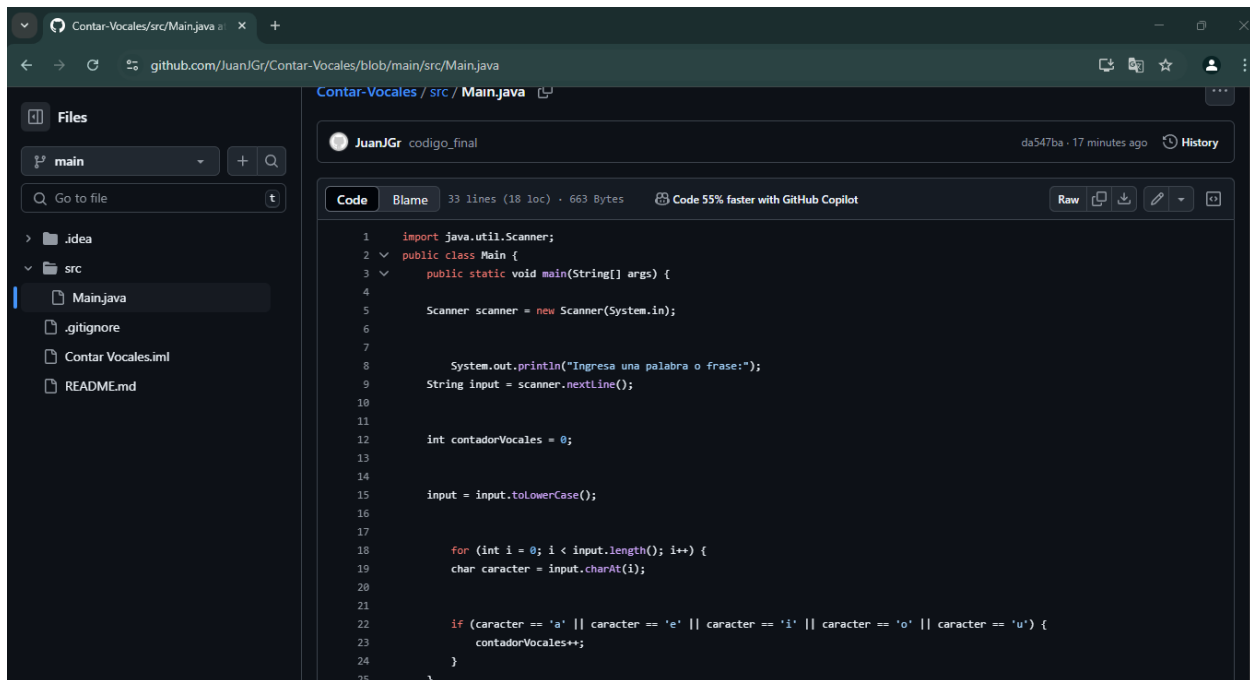
Por último, eliminamos la rama desde el local e importamos los cambios.

```

PS C:\Users\Juan José\Desktop\Mi Proyecto\Contar Vocales> git pull origin main
remote: Enumerating objects: 1, done.
remote: Counting objects: 100% (1/1), done.
remote: Total 1 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
Unpacking objects: 100% (1/1), 899 bytes | 149.00 KiB/s, done.
From https://github.com/JuanJGr/Contar-Vocales
* branch          main          -> FETCH_HEAD
   30ff6f5..5b31538  main          -> origin/main
Updating 30ff6f5..5b31538
Fast-forward
 src/Main.java | 22 ++++++++-----
 1 file changed, 17 insertions(+), 5 deletions(-)
PS C:\Users\Juan José\Desktop\Mi Proyecto\Contar Vocales> git branch -D "cambios_finales"
Deleted branch cambios_finales (was da547ba).

```

Y así es como tenemos el proyecto terminado.



```

1  import java.util.Scanner;
2  public class Main {
3      public static void main(String[] args) {
4
5          Scanner scanner = new Scanner(System.in);
6
7          System.out.println("Ingresa una palabra o frase:");
8          String input = scanner.nextLine();
9
10
11          int contadorVocales = 0;
12
13
14          input = input.toLowerCase();
15
16
17          for (int i = 0; i < input.length(); i++) {
18              char character = input.charAt(i);
19
20
21              if (character == 'a' || character == 'e' || character == 'i' || character == 'o' || character == 'u') {
22                  contadorVocales++;
23              }
24          }
25      }

```


Si queremos ver el historial de cambios:

```
Juan José@LAPTOP-A0S7PFEG MINGW64 ~/Desktop/Mi Proyecto/Contar Vocales (main)
$ git reflog
173e651 (HEAD -> main) HEAD@{0}: checkout: moving from nuevos_cambios to main
5b31538 (origin/main, origin/HEAD, r, nuevos_cambios) HEAD@{1}: checkout: moving from main to nuevos_cambios
173e651 (HEAD -> main) HEAD@{2}: commit: S
5b31538 (origin/main, origin/HEAD, r, nuevos_cambios) HEAD@{3}: checkout: moving from nuevos_cambios to main
5b31538 (origin/main, origin/HEAD, r, nuevos_cambios) HEAD@{4}: checkout: moving from main to nuevos_cambios
5b31538 (origin/main, origin/HEAD, r, nuevos_cambios) HEAD@{5}: checkout: moving from nuevos_cambios to main
5b31538 (origin/main, origin/HEAD, r, nuevos_cambios) HEAD@{6}: checkout: moving from main to nuevos_cambios
5b31538 (origin/main, origin/HEAD, r, nuevos_cambios) HEAD@{7}: pull origin main: Fast-forward
30ff6f5 HEAD@{8}: checkout: moving from cambios_finales to main
da547ba (origin/cambios_finales) HEAD@{9}: commit: codigo_final
30ff6f5 HEAD@{10}: checkout: moving from main to cambios_finales
30ff6f5 HEAD@{11}: checkout: moving from nuevos_cambios to main
eeba42c HEAD@{12}: revert: Revert "Arreglos Segunda Vez"
da0fd34 HEAD@{13}: commit: Arreglos Segunda Vez
30ff6f5 HEAD@{14}: checkout: moving from main to nuevos_cambios
30ff6f5 HEAD@{15}: commit: Primeras Líneas De Código.
...skipping...
173e651 (HEAD -> main) HEAD@{0}: checkout: moving from nuevos_cambios to main
5b31538 (origin/main, origin/HEAD, r, nuevos_cambios) HEAD@{1}: checkout: moving from main to nuevos_cambios
173e651 (HEAD -> main) HEAD@{2}: commit: S
5b31538 (origin/main, origin/HEAD, r, nuevos_cambios) HEAD@{3}: checkout: moving from nuevos_cambios to main
5b31538 (origin/main, origin/HEAD, r, nuevos_cambios) HEAD@{4}: checkout: moving from main to nuevos_cambios
5b31538 (origin/main, origin/HEAD, r, nuevos_cambios) HEAD@{5}: checkout: moving from nuevos_cambios to main
5b31538 (origin/main, origin/HEAD, r, nuevos_cambios) HEAD@{6}: checkout: moving from main to nuevos_cambios
```

git reflog: Muestra el historial de todos los cambios de referencia (como cambios de ramas) en nuestro repositorio, incluso si no están en el historial de commits.

```
Juan José@LAPTOP-A0S7PFEG MINGW64 ~/Desktop/Mi Proyecto/Contar Vocales (main)
$ git log --oneline
173e651 (HEAD -> main) S
5b31538 (origin/main, origin/HEAD, r, nuevos_cambios) Merge pull request #1 from JuanJGr/cambios_finales
da547ba (origin/cambios_finales) codigo_final
30ff6f5 Primeras Líneas De Código.
f4f8002 first commit
```

git log --oneline: Muestra el historial de commits en formato compacto.

En caso de que queramos clonar un repositorio, nos ubicamos en la terminal de la carpeta donde queramos descargarlo y escribimos el siguiente comando con la URL de dicho repositorio:

```
Instale la versión más reciente de PowerShell para obtener nuevas características y mejoras. https://aka.ms/PSWindows
PS C:\Users\Juan José\Desktop\Miproyecto> git clone https://github.com/JuanJGr/Contar-Vocales.git
Cloning into 'Contar-Vocales'...
remote: Enumerating objects: 19, done.
remote: Counting objects: 100% (19/19), done.
remote: Compressing objects: 100% (12/12), done.
remote: Total 19 (delta 2), reused 18 (delta 2), pack-reused 0 (from 0)
Receiving objects: 100% (19/19), done.
Resolving deltas: 100% (2/2), done.
PS C:\Users\Juan José\Desktop\Miproyecto>
```

En caso de que queramos asociar un archivo a un repositorio existente, añadimos el documento a la carpeta del repositorio:

```
PS C:\Users\Juan José\Desktop\Comando Basicos> git add prueba.txt
PS C:\Users\Juan José\Desktop\Comando Basicos> git commit -m "Añadir documento"
[main 672b99f] Añadir documento
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 prueba.txt
PS C:\Users\Juan José\Desktop\Comando Basicos> git push origin main
Enumerating objects: 6, done.
Counting objects: 100% (6/6), done.
Delta compression using up to 6 threads
Compressing objects: 100% (4/4), done.
Writing objects: 100% (5/5), 478 bytes | 159.00 KiB/s, done.
Total 5 (delta 1), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (1/1), done.
To https://github.com/JuanJGr/Comandos-Git.git
 492deec..672b99f  main -> main
```

The screenshot shows the GitHub interface for the repository 'Comandos-Git'. At the top, the repository name is displayed with a 'Public' badge. Below this, the main branch 'main' is selected, showing 1 branch and 0 tags. A search bar and buttons for 'Add file' and 'Code' are visible. The commit history table lists three commits:

Commit Hash	Author	Message	Time
672b99f	JuanJGr	Añadir documento	1 minute ago
492deec	JuanJGr	first commit	9 minutes ago
Initial	JuanJGr	README.md	9 minutes ago

Conclusión.

Este proyecto ha permitido adquirir una comprensión práctica y profunda de los comandos y el flujo de trabajo de Git y GitHub. A través de la creación de un repositorio, la gestión de ramas, la sincronización de cambios y la resolución de errores, he aprendido a manejar de manera efectiva el control de versiones, asegurando un desarrollo organizado y colaborativo.

Uso de Inteligencia Artificial.

El uso de inteligencia artificial que se usó para este trabajo fue un 30% para estructurar el código principal y aclarar algunos conceptos y/o comandos.

Referencias

Atlassian. (s. f.). *Comandos básicos de Git / Atlassian Git*

Tutorial. <https://www.atlassian.com/es/git/glossary#commands>