



MANUAL DE USUARIO GIT

UNIVERSIDAD DE CUNDINAMARCA

FACULTAD DE INGENIERIA

INGENIERIA DE SISTEMA

EXTENSIÓN CHIA

LINEA DE PROFUNDIZACIÓN III

RAUL ALEXANDER SUESCUN PEREZ

SEPTIEMBRE 02 DE 2025

TABLA DE CONTENIDO

INTRODUCCIÓN.....	4
OBJETIVOS	5
CONFIGURACION DE GIT.....	6
Git config –list.....	6
Git config --global user.email	6
Git config – -global user.name	7
Git config - - global - -unset user.email	7
Git config - - global - -unset user.name.....	8
INICIALIZACIÓN Y CLONACIÓN	8
Git init.....	8
Git clone.....	9
TRABAJO CON CAMBIOS	9
Git status	9
Git add	10
Git commit -m “ ”	10
SINCRONIZACIÓN CON REMOTO	11
Git push origin rama	11
Git pull origin rama	11
MANEJO DE RAMAS	12
Git push origin - - delete rama	12
Git branch -D rama.....	12
Git swtich -c rama.....	13
Git switch rama	13
Git Branch -r.....	14
INFORMACIÓN E HISTORIAL	14
Git remote -v	14
Git log.....	15
Git reflog	15
Git log --online.....	16
OPERACIONES AVANZADAS.....	16
Git fetch - -all.....	16



Git merge	17
Git revert	17
CONCLUSIONES	18
REFERENCIAS BIBLIOGRAFICAS.....	19

INTRODUCCIÓN

Este manual de usuario ha sido creado con el propósito fundamental de servir como una guía clara, concisa y práctica para estudiantes de todos los niveles que deseen dominar el uso de Git, el sistema de control de versiones más popular en la industria del software. Se ha realizado la compilación y organización metódica de los comandos esenciales, agrupándolos por funcionalidad para facilitar su consulta rápida y su aprendizaje progresivo. La estructura está diseñada para que, desde el momento de la configuración inicial hasta la ejecución de operaciones avanzadas, cualquier usuario pueda encontrar la información precisa que necesita para trabajar de manera eficiente y sin contratiempos.

La creación de este compendio nace de la necesidad de contar con una referencia confiable y unificada que elimine la ambigüedad y proporcione descripciones directas de la utilidad de cada instrucción. Más que una simple lista, este manual busca contextualizar cada comando dentro del flujo de trabajo real de un proyecto. Es ideal que esta herramienta se convierta en un acompañante indispensable en el día a día, potenciando la productividad y fortaleciendo la comprensión sobre el robusto ecosistema de Git.

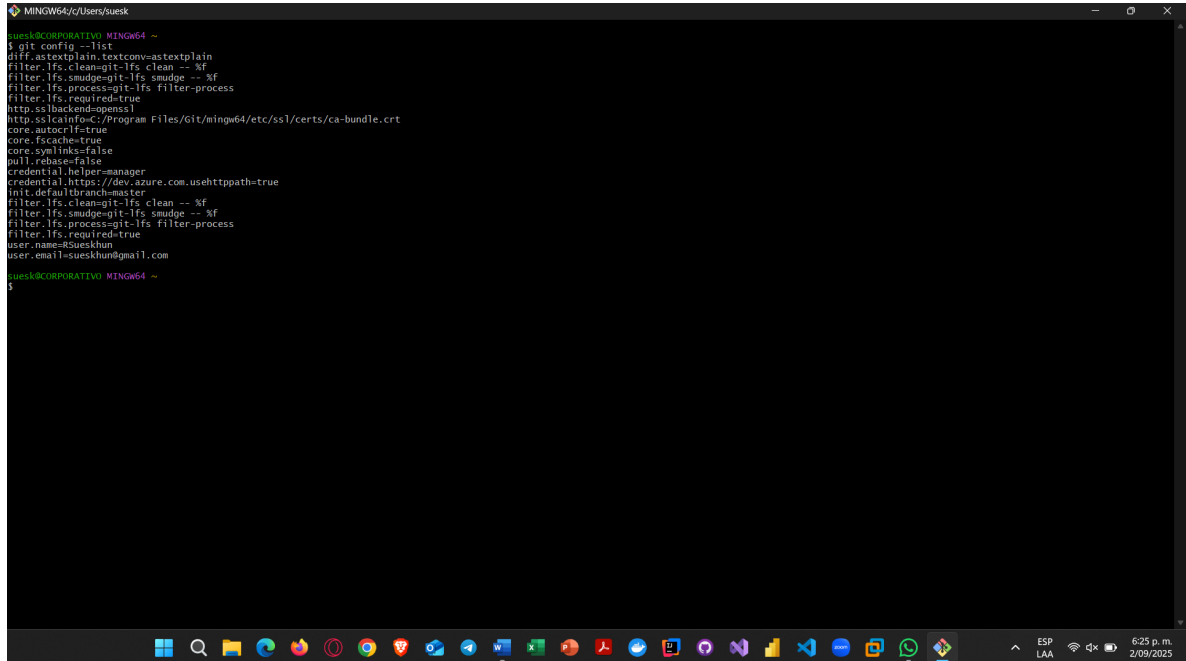
OBJETIVOS

1. Reducir la complejidad inicial asociada con Git, desglosando comandos críticos en descripciones sencillas y accesibles.
2. Promover las buenas prácticas y un flujo de trabajo estandarizado, educando sobre su aplicación correcta dentro de un flujo de trabajo colaborativo moderno.
3. Fomentar la autonomía y la resolución de problemas al proporcionar una explicación clara de comandos para revertir cambios, recuperar estados anteriores, gestionar ramas o sincronizar con repositorios remotos.

CONFIGURACION DE GIT

Git config --list

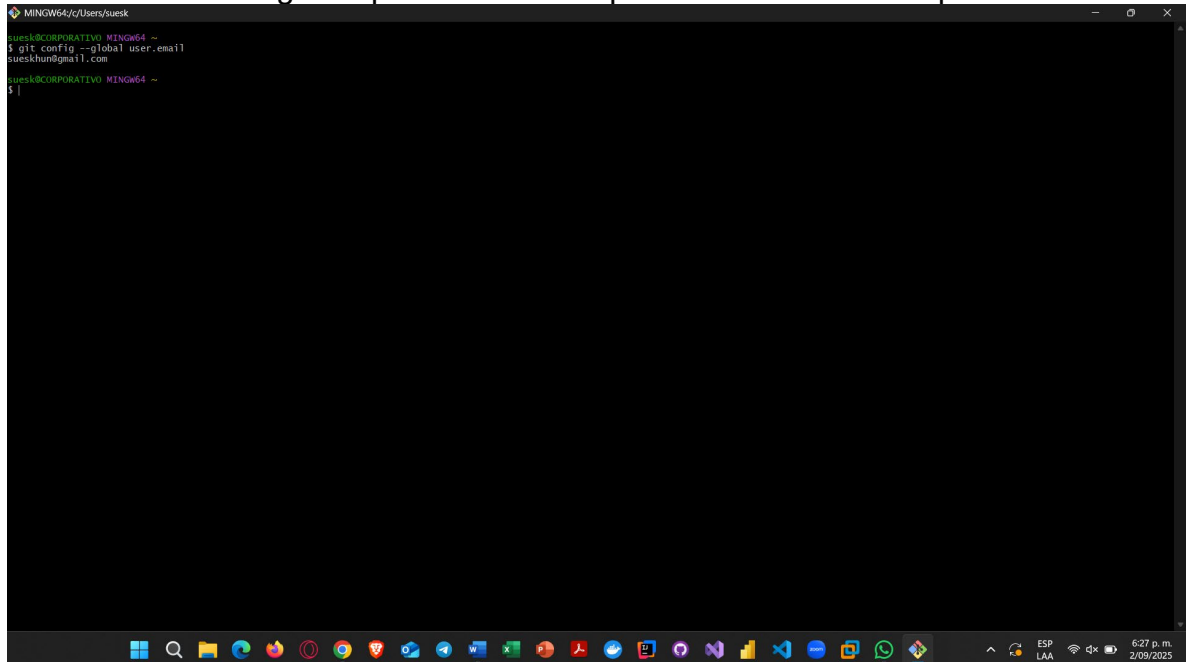
Muestra todas las configuraciones de Git actualmente establecidas en el sistema.



```
sueskh@CORPORATIVO MINGW64 ~  
$ 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=openssl  
http.sslcainfo=C:/Program Files/Git/mingw64/etc/ssl/certs/ca-bundle.crt  
core.autocrlf=true  
core.fsckcache=true  
core.symbols=false  
pull.rebase=false  
credential.helper=manager  
credential.https://dev.azure.com.usehttppath=true  
init.defaultbranch=master  
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  
user.name=Sueskhun  
user.email=sueskhun@gmail.com  
$
```

Git config --global user.email

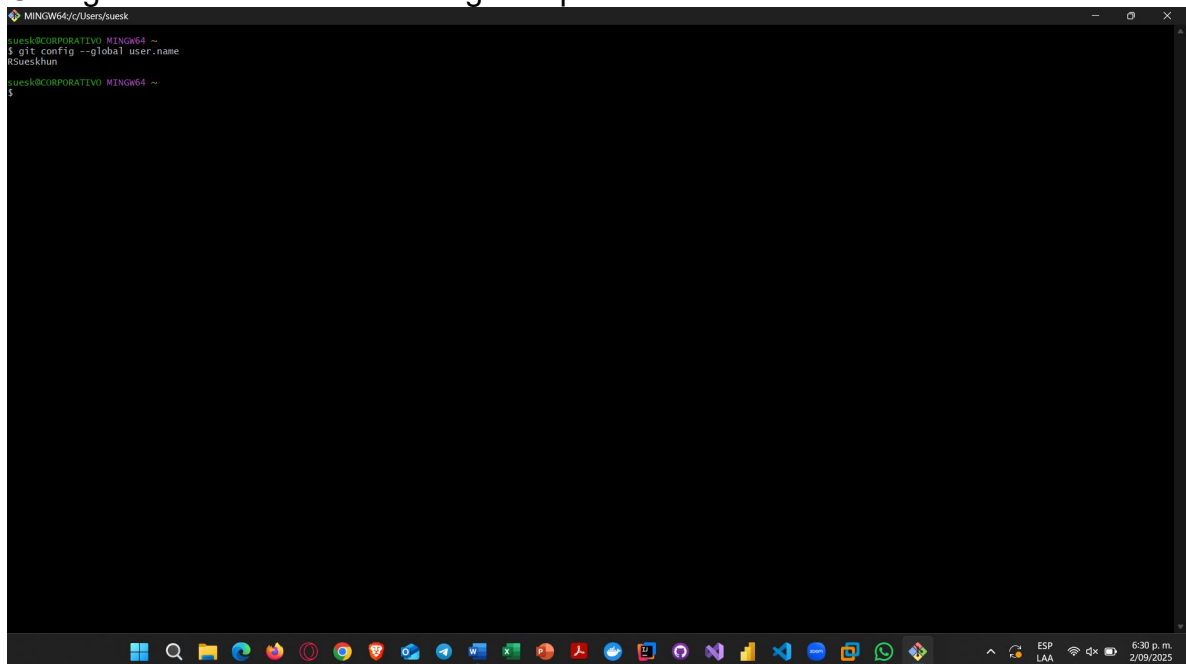
Establece el email global para todos los repositorios Git en la máquina.



```
sueskh@CORPORATIVO MINGW64 ~  
$ git config --global user.email  
sueskhun@gmail.com  
$
```

Git config --global user.name

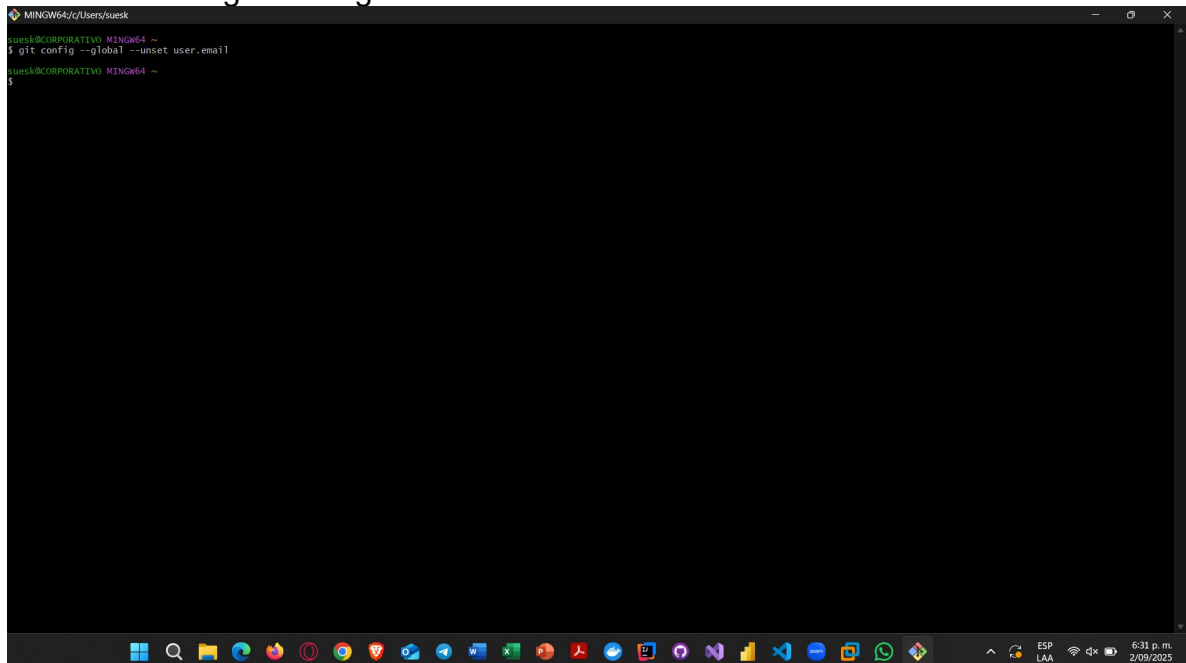
Configura el nombre de usuario global para todos los commits.



```
MINGW64/c/Users/suesk
suesk@CORPORATIVO MINGW64 ~
$ git config --global user.name
sueskhun
suesk@CORPORATIVO MINGW64 ~
$
```

Git config --global --unset user.email

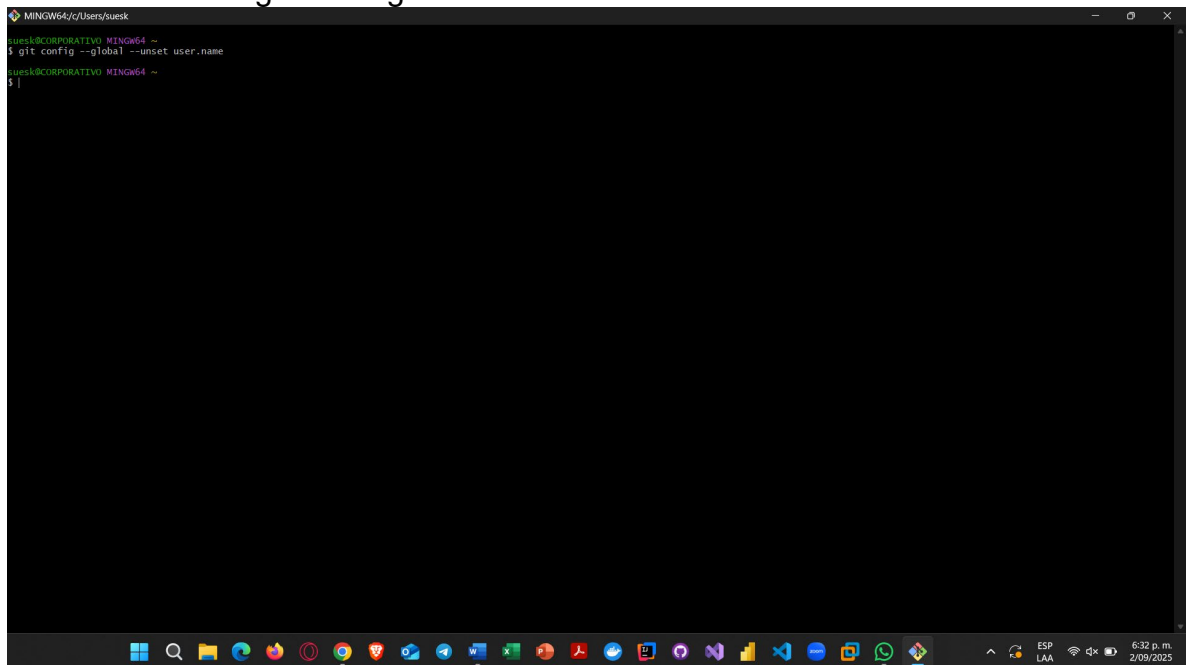
Elimina la configuración global del email del usuario.



```
MINGW64/c/Users/suesk
suesk@CORPORATIVO MINGW64 ~
$ git config --global --unset user.email
suesk@CORPORATIVO MINGW64 ~
$
```

Git config - - global - -unset user.name

Remueve la configuración global del nombre de usuario.

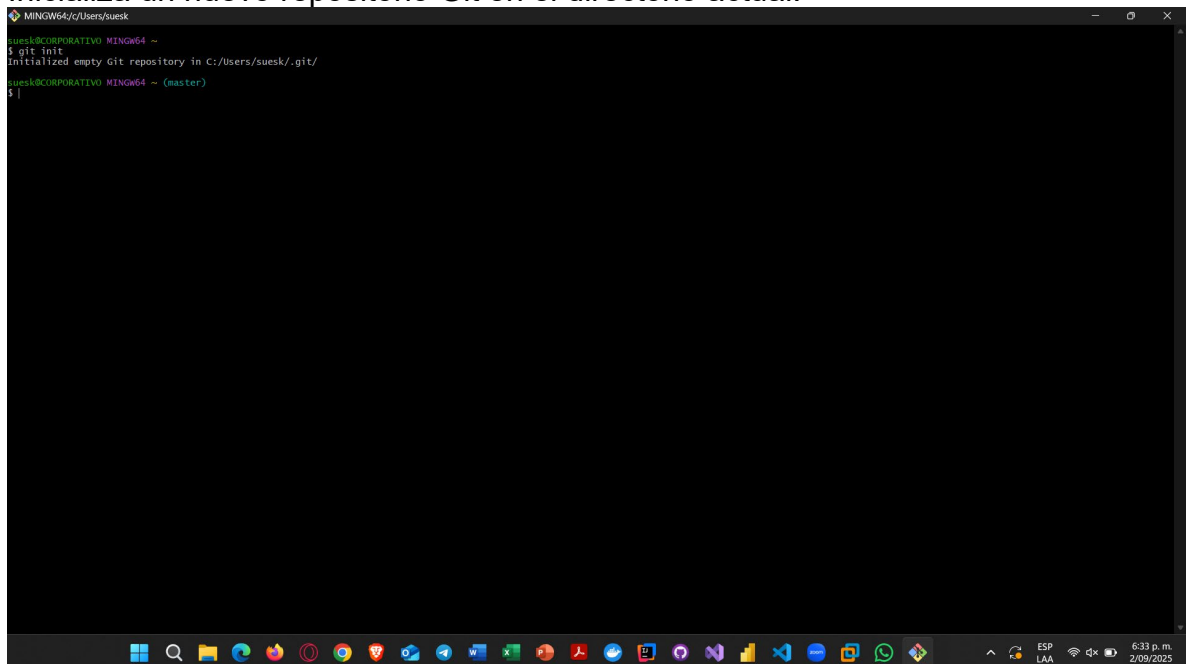


```
MINGW64/c/Users/suesk
suesk@CORPORATIVO MINGW64 ~
$ git config --global --unset user.name
suesk@CORPORATIVO MINGW64 ~
$ |
```

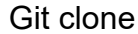
INICIALIZACIÓN Y CLONACIÓN

Git init

Inicializa un nuevo repositorio Git en el directorio actual.



```
MINGW64/c/Users/suesk
suesk@CORPORATIVO MINGW64 ~
$ git init
Initialized empty Git repository in C:/Users/suesk/.git/
suesk@CORPORATIVO MINGW64 ~ (master)
$ |
```

Copia un repositorio remoto completo a la máquina local. Para completar el proceso se debe especificar la fuente del repositorio a clonar de lo contrario solo mostrará los complementos que acompañan este comando.

```
MINGW64/c/Users/juesh/
juesh@KORPORATIVU MINGW64 ~ (master)
$ git clone
fatal: You must specify a repository to clone.

usage: git clone [-options] [-] <repo> [<dir>]

  -v, --[no-]verbose          be more verbose
  -q, --[no-]quiet            be more quiet
  --[no-]progress            force progress reporting
  --[no-]reject-shallow       don't clone shallow repository
  -n, --no-checkout           don't create a checkout
                             opposite of --no-checkout
  --checkout                 create a bare repository
  --[no-]bare                create a mirror repository (implies --bare)
  -l, --[no-]local            to clone from a local repository
  --no-hardlinks              don't use local hardlinks, always copy
  --hardlinks                opposite of --no-hardlinks
  -s, --[no-]shared           setup as shared repository
  --[no-]recurse-submodules[=<pathspec>]
                             initialize submodules in the clone
  --[no-]recursive ...       alias of --recurse-submodules
  -j, --[no-]jobs <n>         number of submodules cloned in parallel
  --[no-]template <template-directory>
                             directory from which templates will be used
  --[no-]reference <repo>    reference repository
  --[no-]reference-if-able <repo>
                             reference repository
  --[no-]dissociate           use --reference only while cloning
  -o, --[no-]origin <name>   use <name> instead of 'origin' to track upstream
  -b, --[no-]branch <branch> checkout <branch> instead of the remote's HEAD
  -u, --[no-]upload-pack <path>
                             path to git-upload-pack on the remote
  --[no-]depth <depth>       create a shallow clone of that depth
  --[no-]shallow-since <time>
                             create a shallow clone since a specific time
  --[no-]shallow-exclude <revision>
                             deepen history of shallow clone, excluding rev
  --[no-]single-branch        clone only one branch, HEAD or --branch
  --no-tags                   don't clone any tags, and make later fetches not to follow them
  --tags                      opposite of --no-tags
  --[no-]shallow-submodules   any cloned submodules will be shallow
  --[no-]separate-git-dir <gitdir>
                             separate git dir from working tree
  --[no-]ref-format <format> specify the reference format to use
  -c, --[no-]config <key=value>
                             set config inside the new repository
  --[no-]server-option <server-specific>
                             option to transmit
  -4, --ipv4                  use IPv4 addresses only
  -6, --ipv6                  use IPv6 addresses only
  --[no-]filter <args>        object filtering
  --[no-]also-filter-submodules
```

TRABAJO CON CAMBIOS

Git status

Muestra el estado actual del repositorio.

The screenshot shows an IDE with a project named "Manual de Usuario GIT". The file explorer on the left shows the project structure, including a "src" directory with a "Main.java" file. The editor window displays the code for "Main.java":

```

public class Main {
    public static void main(String[] args) {
        // To Run code, press [Mayús] [F10] or click the ▶ icon in the gutter.
        // Press [Alt] [Enter] with your caret at the highlighted text to see how IntelliJ IDEA suggests
        // fixing it.
        System.out.println("Hello and welcome!");
    }
}

```

The terminal window at the bottom shows the output of the command `git status`:

```

PS C:\Users\suesk\IdeaProjects\Manual de Usuario GIT> git status
On branch main

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)

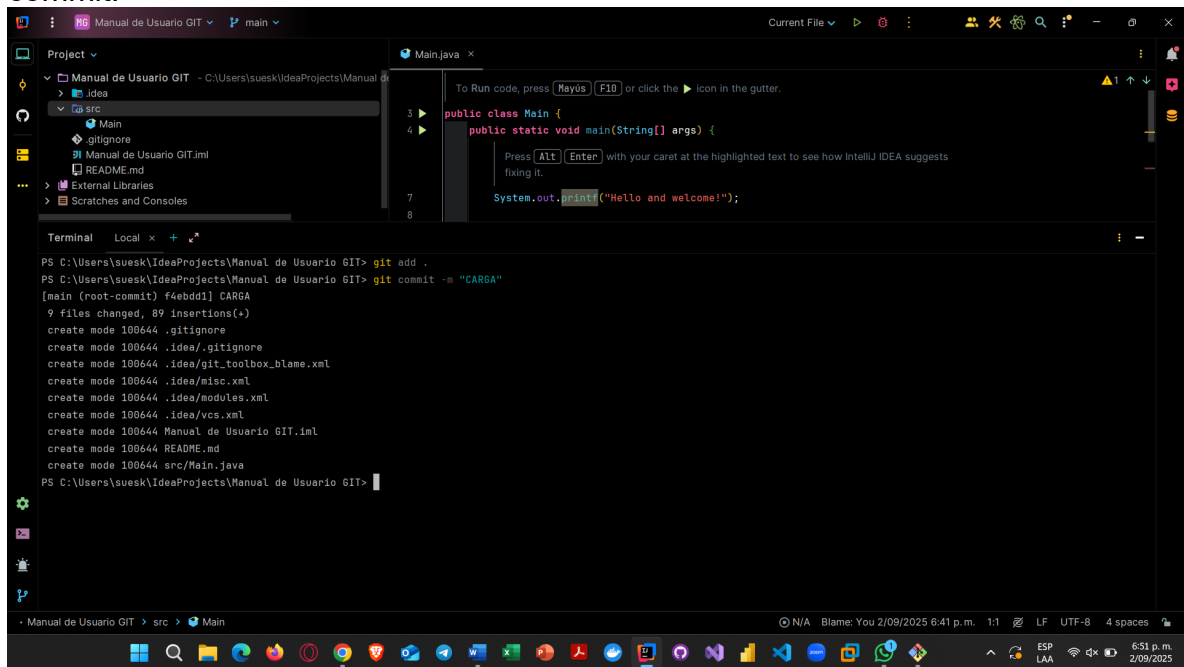
    new file:   .gitignore
    new file:   .idea/.gitignore
    new file:   .idea/git_toolbox_blame.xml
    new file:   .idea/misc.xml
    new file:   .idea/modules.xml
    new file:   .idea/vcs.xml
    new file:   Manual de Usuario GIT.iml
    new file:   README.md
    new file:   src/Main.java

```

The status bar at the bottom indicates the current file is "Main.java", the project is "Manual de Usuario GIT", and the branch is "main".

Git add .

Agrega todos los archivos modificados al área de staging para el próximo commit.

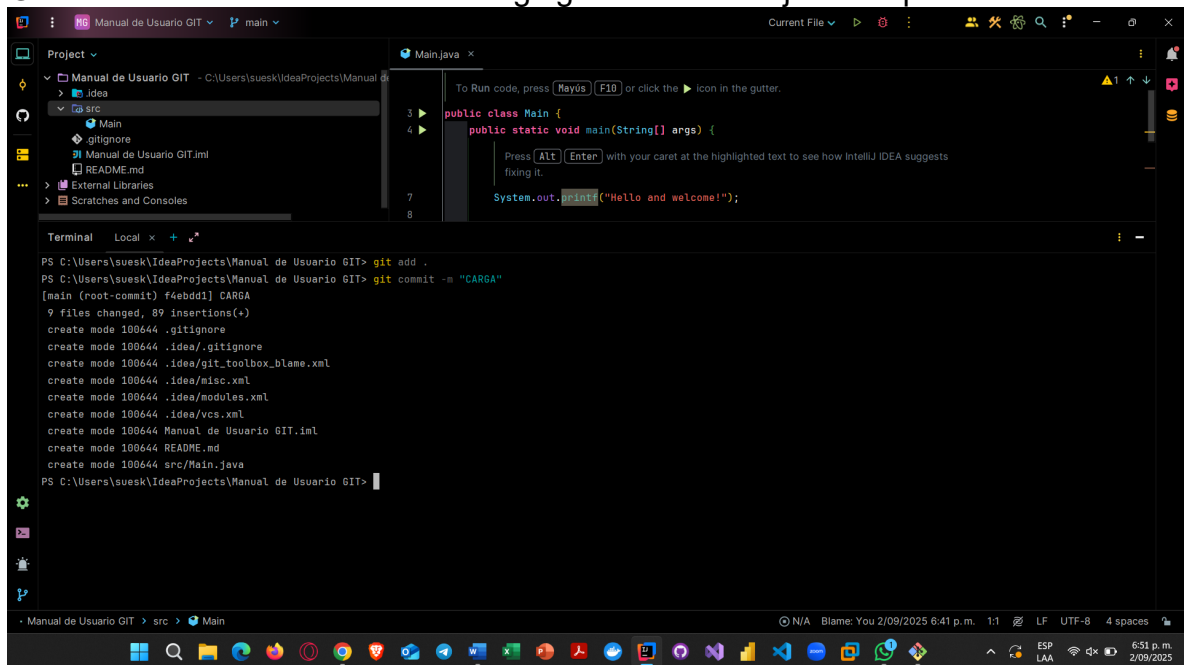


The screenshot shows the IntelliJ IDEA interface. The Project view on the left shows the file structure of 'Manual de Usuario GIT'. The Main.java file is open in the editor, showing a simple Java class with a main method. The Terminal window at the bottom shows the execution of the following commands:

```
PS C:\Users\suesk\IdeaProjects\Manual de Usuario GIT> git add .
PS C:\Users\suesk\IdeaProjects\Manual de Usuario GIT> git commit -m "CARGA"
[main (root-commit) f4ebdd1] CARGA
9 files changed, 89 insertions(+)
create mode 100644 .gitignore
create mode 100644 .idea/.gitignore
create mode 100644 .idea/git_toolbox_blame.xml
create mode 100644 .idea/misc.xml
create mode 100644 .idea/modules.xml
create mode 100644 .idea/vcs.xml
create mode 100644 Manual de Usuario GIT.iml
create mode 100644 README.md
create mode 100644 src/Main.java
PS C:\Users\suesk\IdeaProjects\Manual de Usuario GIT>
```

Git commit -m “ ”

Guarda los cambios del área de staging con un mensaje descriptivo.



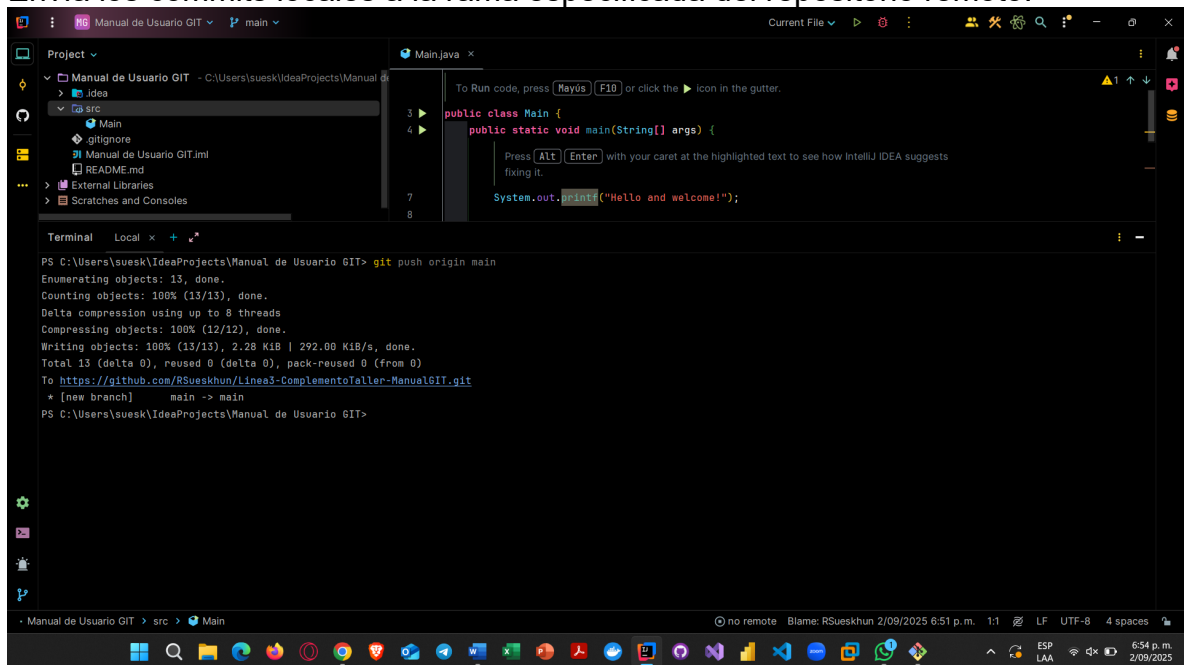
The screenshot shows the IntelliJ IDEA interface, similar to the previous one. The Terminal window at the bottom shows the execution of the following commands:

```
PS C:\Users\suesk\IdeaProjects\Manual de Usuario GIT> git add .
PS C:\Users\suesk\IdeaProjects\Manual de Usuario GIT> git commit -m "CARGA"
[main (root-commit) f4ebdd1] CARGA
9 files changed, 89 insertions(+)
create mode 100644 .gitignore
create mode 100644 .idea/.gitignore
create mode 100644 .idea/git_toolbox_blame.xml
create mode 100644 .idea/misc.xml
create mode 100644 .idea/modules.xml
create mode 100644 .idea/vcs.xml
create mode 100644 Manual de Usuario GIT.iml
create mode 100644 README.md
create mode 100644 src/Main.java
PS C:\Users\suesk\IdeaProjects\Manual de Usuario GIT>
```

SINCRONIZACIÓN CON REMOTO

Git push origin rama

Envía los commits locales a la rama especificada del repositorio remoto.



The screenshot shows the IntelliJ IDEA interface. The Project view on the left displays the file structure of the 'Manual de Usuario GIT' project. The main editor shows the 'Main.java' file with the following code:

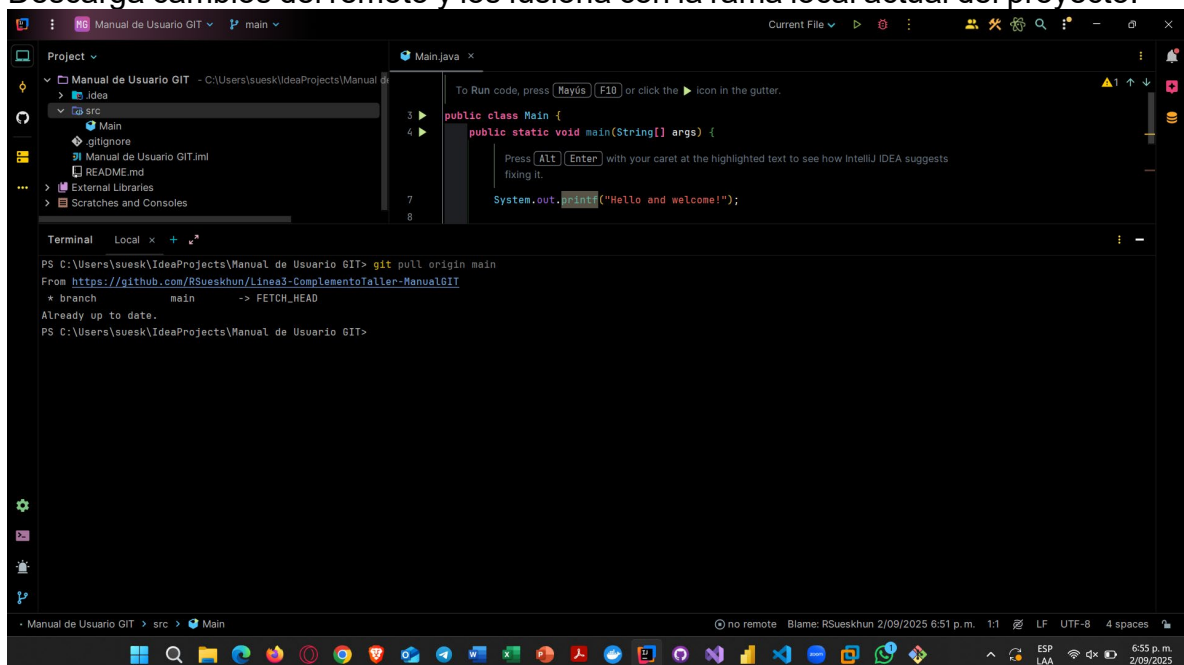
```
public class Main {  
    public static void main(String[] args) {  
        System.out.printf("Hello and welcome!");  
    }  
}
```

The Terminal window at the bottom shows the execution of the 'git push origin main' command, which was successful. The output indicates that 13 objects were enumerated, 100% of the objects were counted, and the push was completed to the 'main' branch of the remote repository.

```
PS C:\Users\suesk\IdeaProjects\Manual de Usuario GIT> git push origin main  
Enumerating objects: 13, done.  
Counting objects: 100% (13/13), done.  
Delta compression using up to 8 threads  
Compressing objects: 100% (12/12), done.  
Writing objects: 100% (13/13), 2.28 KiB | 292.00 KiB/s, done.  
Total 13 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)  
To https://github.com/RSueskhun/Linea3-ComplementoTaller-ManualGIT.git  
* [new branch]    main -> main  
PS C:\Users\suesk\IdeaProjects\Manual de Usuario GIT>
```

Git pull origin rama

Descarga cambios del remoto y los fusiona con la rama local actual del proyecto.



The screenshot shows the IntelliJ IDEA interface. The Project view on the left displays the file structure of the 'Manual de Usuario GIT' project. The main editor shows the 'Main.java' file with the following code:

```
public class Main {  
    public static void main(String[] args) {  
        System.out.printf("Hello and welcome!");  
    }  
}
```

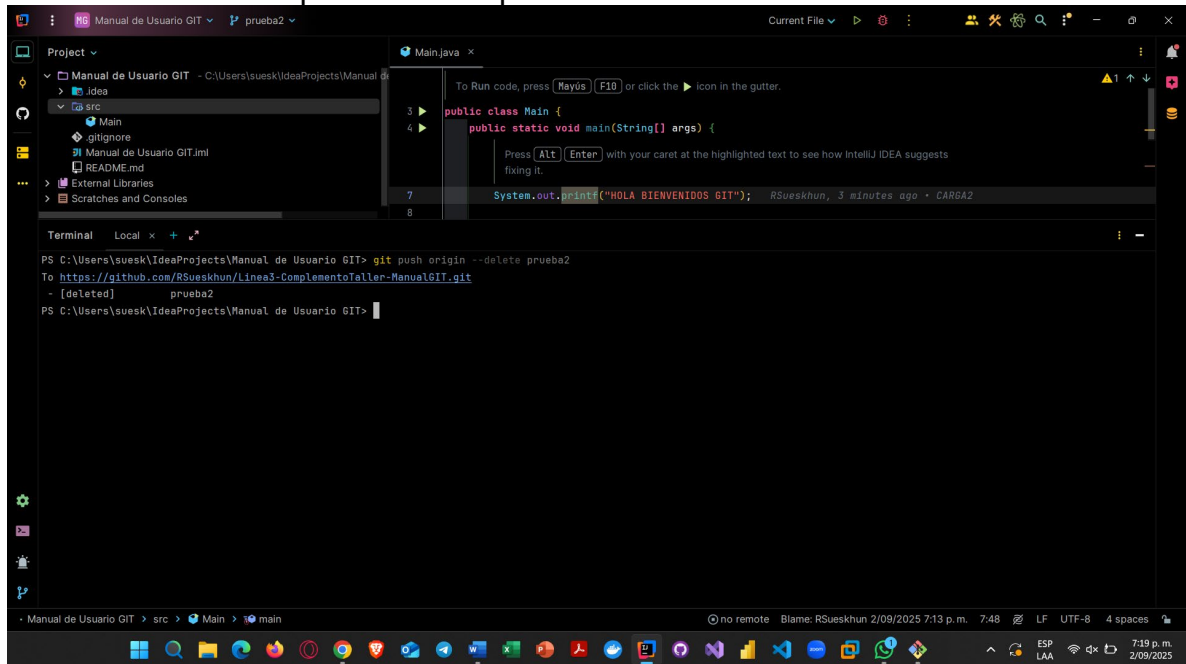
The Terminal window at the bottom shows the execution of the 'git pull origin main' command, which was successful. The output indicates that the local branch is already up to date with the remote branch.

```
PS C:\Users\suesk\IdeaProjects\Manual de Usuario GIT> git pull origin main  
From https://github.com/RSueskhun/Linea3-ComplementoTaller-ManualGIT  
* branch          main      -> FETCH_HEAD  
Already up to date.  
PS C:\Users\suesk\IdeaProjects\Manual de Usuario GIT>
```

MANEJO DE RAMAS

Git push origin --delete rama

Elimina una rama específica del repositorio remoto.

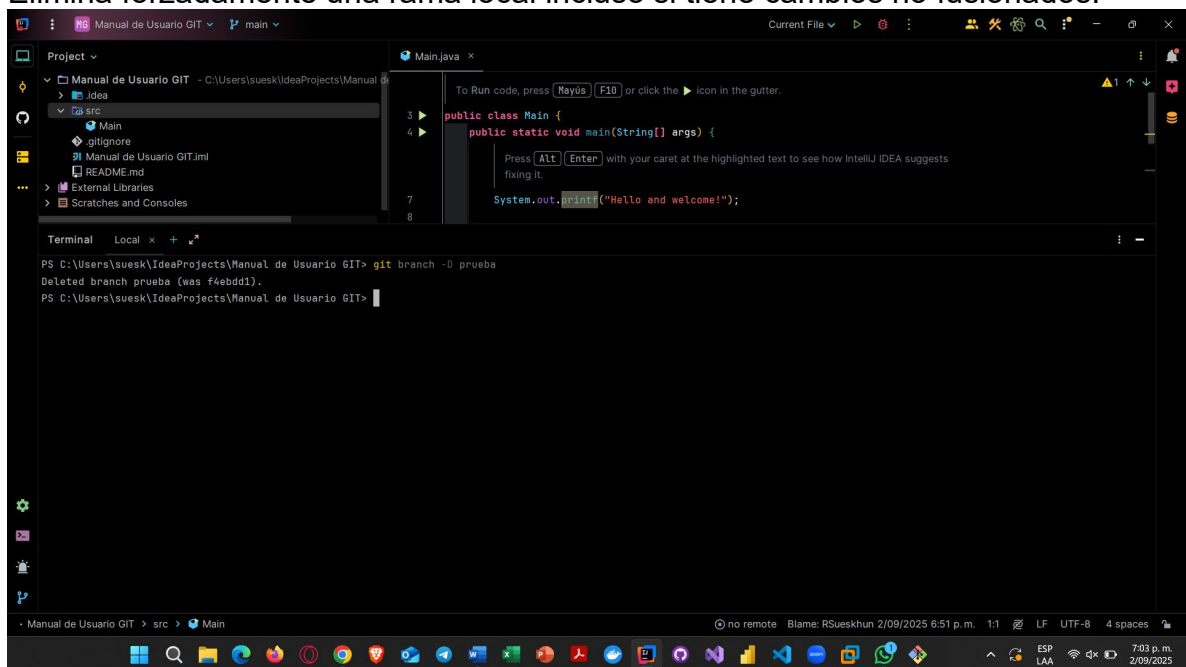


The screenshot shows the IntelliJ IDEA interface. The Project view on the left shows the 'Manual de Usuario GIT' project. The Main.java file is open in the editor, showing a Java class with a main method. The terminal at the bottom displays the command 'git push origin --delete prueba2' and its output, indicating that the remote branch 'prueba2' has been successfully deleted.

```
PS C:\Users\suesk\IdeaProjects\Manual de Usuario GIT> git push origin --delete prueba2
To https://github.com/RSueskhun/linea3-ComplementoTaller-ManualGIT.git
- [deleted]           prueba2
PS C:\Users\suesk\IdeaProjects\Manual de Usuario GIT>
```

Git branch -D rama

Elimina forzosamente una rama local incluso si tiene cambios no fusionados.

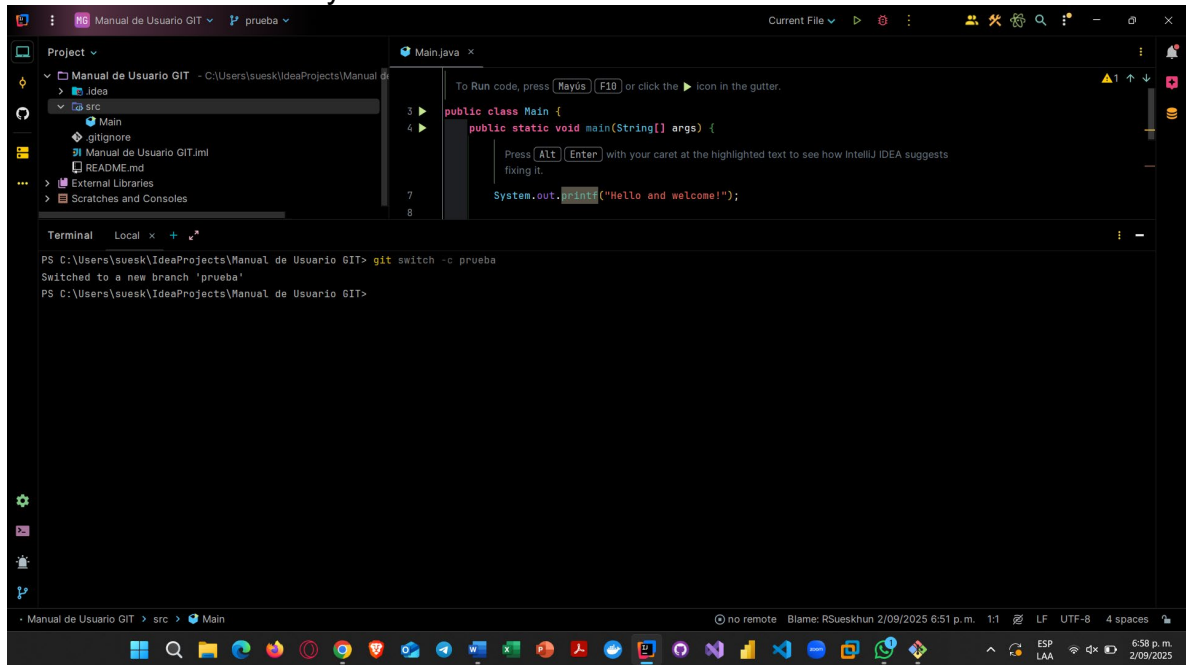


The screenshot shows the IntelliJ IDEA interface. The Project view on the left shows the 'Manual de Usuario GIT' project. The Main.java file is open in the editor, showing a Java class with a main method. The terminal at the bottom displays the command 'git branch -D prueba' and its output, indicating that the local branch 'prueba' has been successfully deleted.

```
PS C:\Users\suesk\IdeaProjects\Manual de Usuario GIT> git branch -D prueba
Deleted branch prueba (was f4ebdd1).
PS C:\Users\suesk\IdeaProjects\Manual de Usuario GIT>
```

Git switch -c rama

Crea una nueva rama y cambia a ella inmediatamente.



The screenshot shows the IntelliJ IDEA interface. The Project view on the left shows the file structure of the 'Manual de Usuario GIT' project. The main editor displays the 'Main.java' file with the following code:

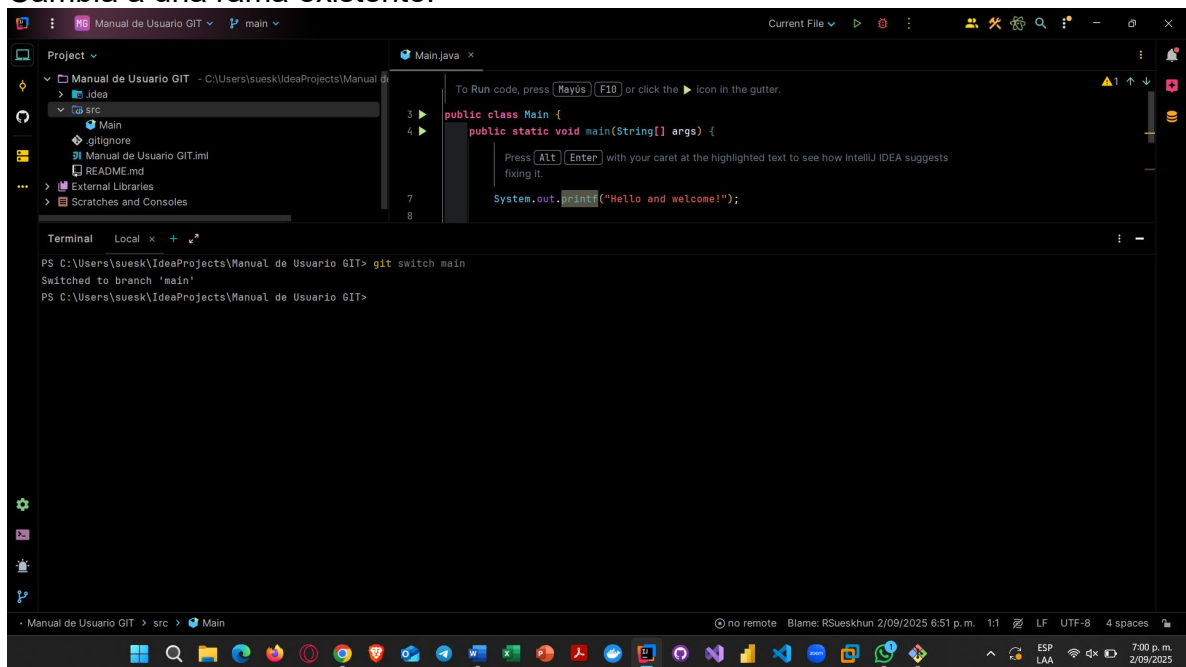
```
public class Main {  
    public static void main(String[] args) {  
        System.out.println("Hello and welcome!");  
    }  
}
```

The Terminal at the bottom shows the execution of the 'git switch -c prueba' command, which successfully switches to the new branch 'prueba'.

```
PS C:\Users\suesk\IdeaProjects\Manual de Usuario GIT> git switch -c prueba  
Switched to a new branch 'prueba'  
PS C:\Users\suesk\IdeaProjects\Manual de Usuario GIT>
```

Git switch rama

Cambia a una rama existente.



The screenshot shows the IntelliJ IDEA interface. The Project view on the left shows the file structure of the 'Manual de Usuario GIT' project. The main editor displays the 'Main.java' file with the following code:

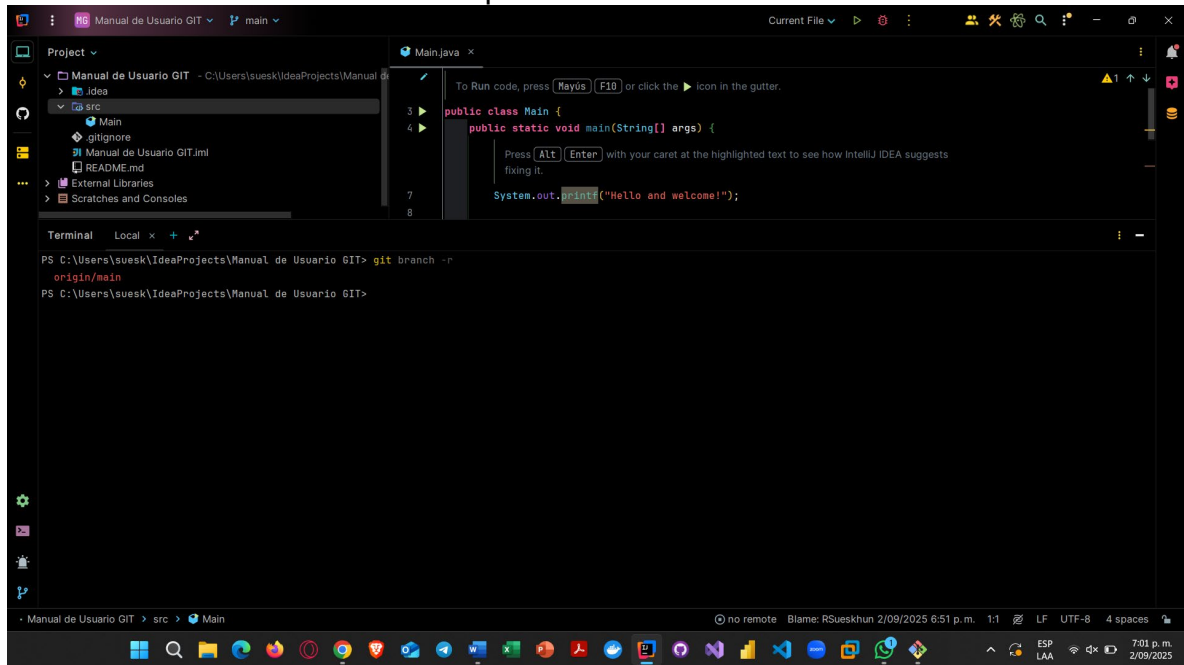
```
public class Main {  
    public static void main(String[] args) {  
        System.out.println("Hello and welcome!");  
    }  
}
```

The Terminal at the bottom shows the execution of the 'git switch main' command, which successfully switches to the existing branch 'main'.

```
PS C:\Users\suesk\IdeaProjects\Manual de Usuario GIT> git switch main  
Switched to branch 'main'  
PS C:\Users\suesk\IdeaProjects\Manual de Usuario GIT>
```

Git Branch -r

Lista todas las ramas remotas disponibles.



The screenshot shows the IntelliJ IDEA interface. The Project view on the left displays the file structure of the 'Manual de Usuario GIT' project. The main editor shows a Java file 'Main.java' with the following code:

```
public class Main {  
    public static void main(String[] args) {  
        System.out.printf("Hello and welcome!");  
    }  
}
```

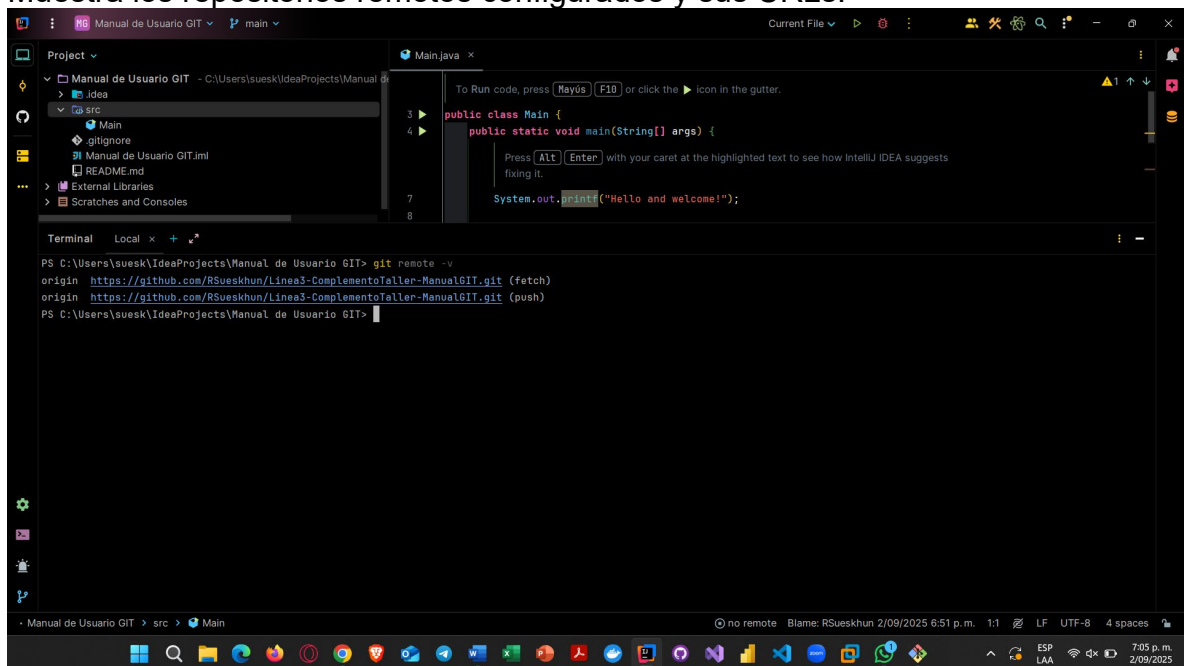
The Terminal at the bottom shows the command 'git branch -r' being executed, which lists the remote branches:

```
PS C:\Users\suesk\IdeaProjects\Manual de Usuario GIT> git branch -r  
origin/main  
PS C:\Users\suesk\IdeaProjects\Manual de Usuario GIT>
```

INFORMACIÓN E HISTORIAL

Git remote -v

Muestra los repositorios remotos configurados y sus URLs.



The screenshot shows the IntelliJ IDEA interface. The Project view on the left displays the file structure of the 'Manual de Usuario GIT' project. The main editor shows a Java file 'Main.java' with the following code:

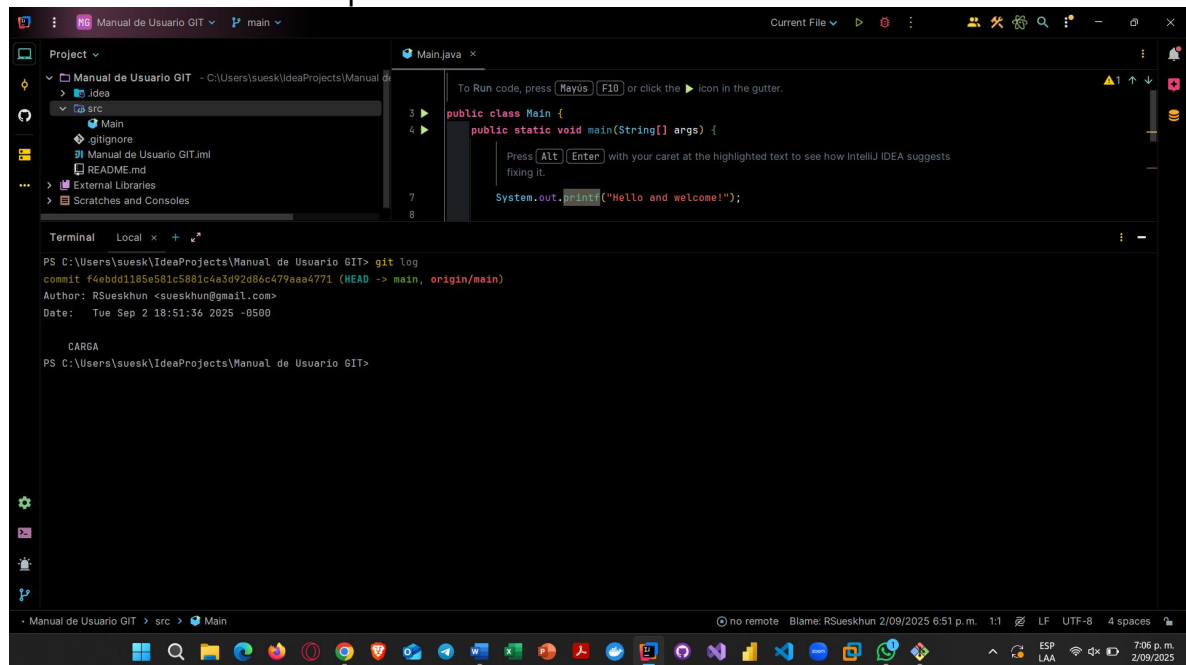
```
public class Main {  
    public static void main(String[] args) {  
        System.out.printf("Hello and welcome!");  
    }  
}
```

The Terminal at the bottom shows the command 'git remote -v' being executed, which lists the remote repositories and their URLs:

```
PS C:\Users\suesk\IdeaProjects\Manual de Usuario GIT> git remote -v  
origin https://github.com/RSueskhun/Linea3-ComplementoTaller-ManualGIT.git (fetch)  
origin https://github.com/RSueskhun/Linea3-ComplementoTaller-ManualGIT.git (push)  
PS C:\Users\suesk\IdeaProjects\Manual de Usuario GIT>
```

Git log

Muestra el historial completo de commits con detalles extensos.



The screenshot shows the IntelliJ IDEA interface. The Project view on the left displays the file structure of the 'Manual de Usuario GIT' project. The main editor shows the 'Main.java' file with a simple Java class. The terminal at the bottom displays the output of the 'git log' command, showing a single commit with the hash 'f4ebdd1185e581c5881c4a3d92d86c479aaa4771' and the author 'RSueskhun'.

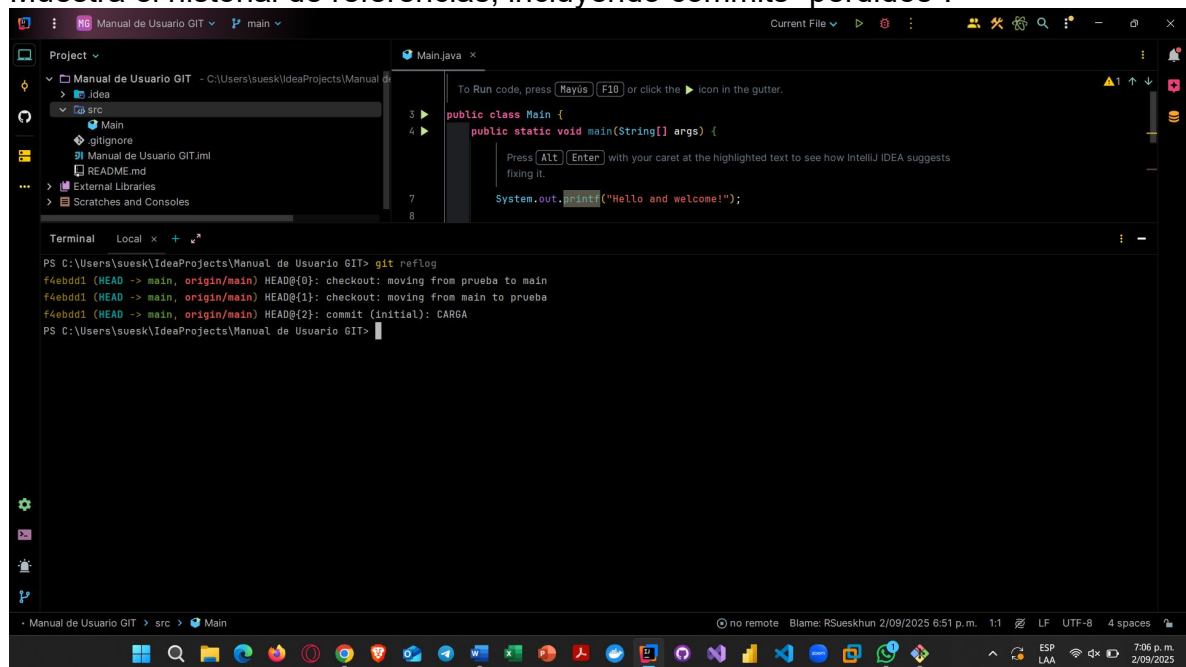
```
PS C:\Users\suesk\IdeaProjects\Manual de Usuario GIT> git log
commit f4ebdd1185e581c5881c4a3d92d86c479aaa4771 (HEAD -> main, origin/main)
Author: RSueskhun <sueskhun@gmail.com>
Date: Tue Sep 2 18:51:36 2025 -0500

CARGA

PS C:\Users\suesk\IdeaProjects\Manual de Usuario GIT>
```

Git relog

Muestra el historial de referencias, incluyendo commits "perdidos".



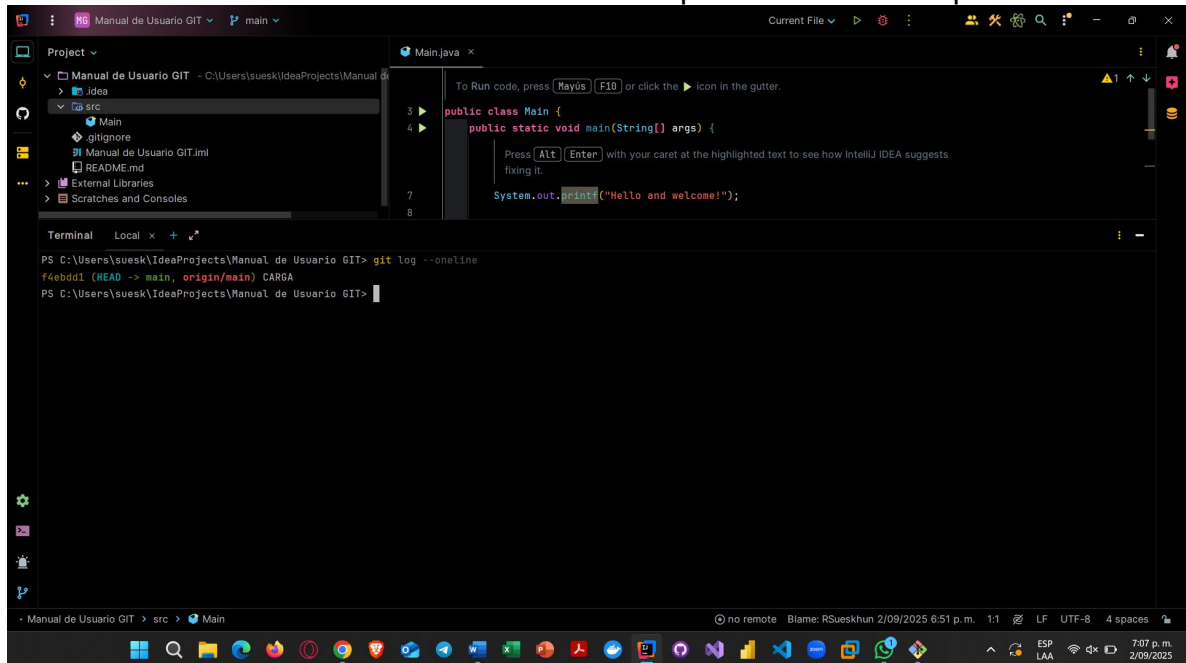
The screenshot shows the IntelliJ IDEA interface. The Project view on the left displays the file structure of the 'Manual de Usuario GIT' project. The main editor shows the 'Main.java' file with a simple Java class. The terminal at the bottom displays the output of the 'git relog' command, showing the history of the 'HEAD' pointer, including checkout operations and a commit.

```
PS C:\Users\suesk\IdeaProjects\Manual de Usuario GIT> git relog
f4ebdd1 (HEAD -> main, origin/main) HEAD@{0}: checkout: moving from prueba to main
f4ebdd1 (HEAD -> main, origin/main) HEAD@{1}: checkout: moving from main to prueba
f4ebdd1 (HEAD -> main, origin/main) HEAD@{2}: commit (initial): CARGA

PS C:\Users\suesk\IdeaProjects\Manual de Usuario GIT>
```

Git log --oneline

Muestra el historial de commits en formato compacto de una línea por commit.



The screenshot shows the IntelliJ IDEA interface. The Project view on the left displays the project structure. The main editor shows a Java file named Main.java with the following code:

```
public class Main {  
    public static void main(String[] args) {  
        System.out.println("Hello and welcome!");  
    }  
}
```

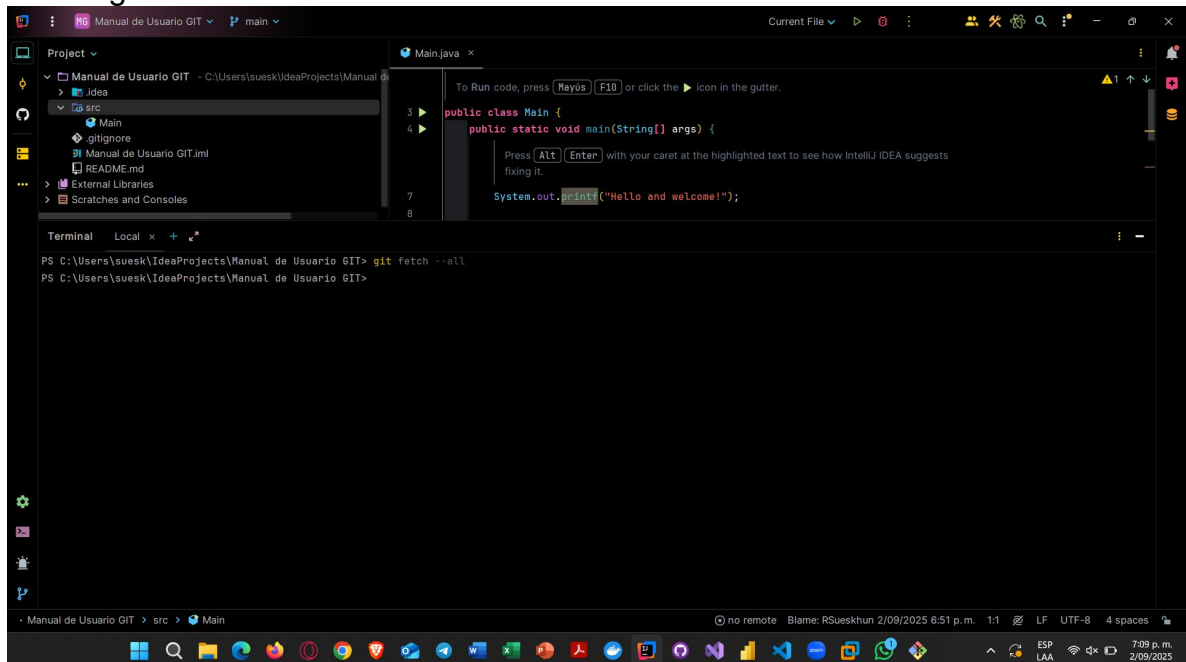
The Terminal window at the bottom shows the execution of the command `git log --oneline`. The output is:

```
PS C:\Users\suesk\IdeaProjects\Manual de Usuario GIT> git log --oneline  
f4ebdd1 (HEAD -> main, origin/main) CARGA  
PS C:\Users\suesk\IdeaProjects\Manual de Usuario GIT>
```

OPERACIONES AVANZADAS

Git fetch - -all

Descarga todos los cambios del remoto sin fusionarlos automáticamente.



The screenshot shows the IntelliJ IDEA interface. The Project view on the left displays the project structure. The main editor shows a Java file named Main.java with the following code:

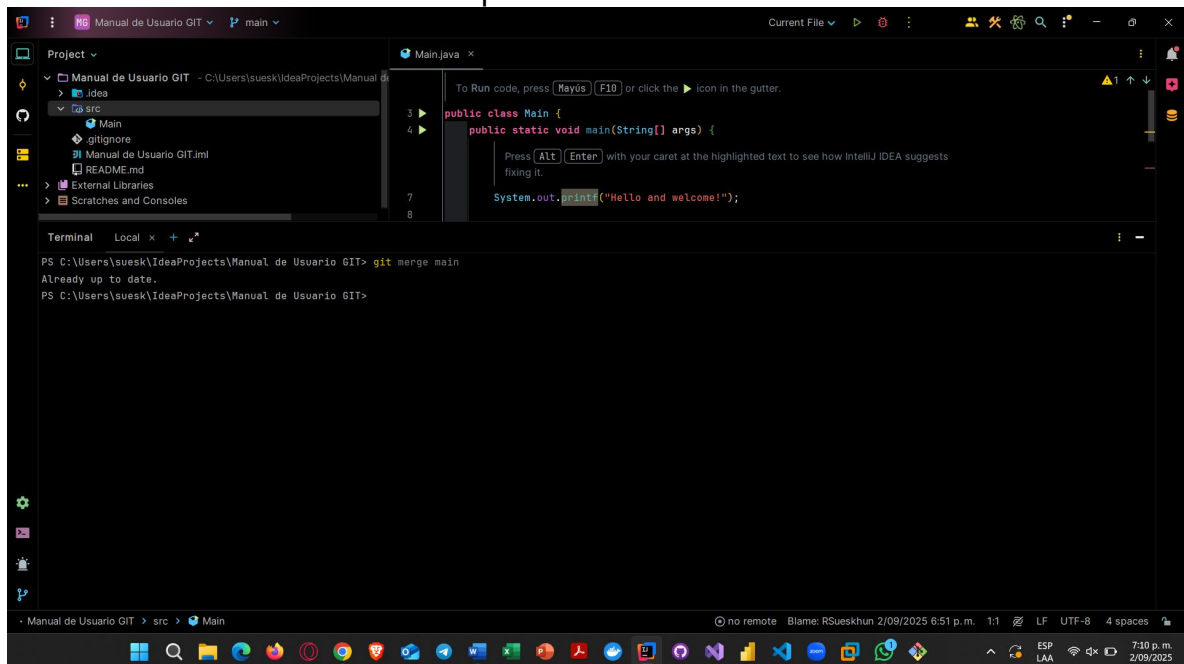
```
public class Main {  
    public static void main(String[] args) {  
        System.out.println("Hello and welcome!");  
    }  
}
```

The Terminal window at the bottom shows the execution of the command `git fetch --all`. The output is:

```
PS C:\Users\suesk\IdeaProjects\Manual de Usuario GIT> git fetch --all  
PS C:\Users\suesk\IdeaProjects\Manual de Usuario GIT>
```


Git merge

Fusiona cambios de una rama específica a la rama actual.

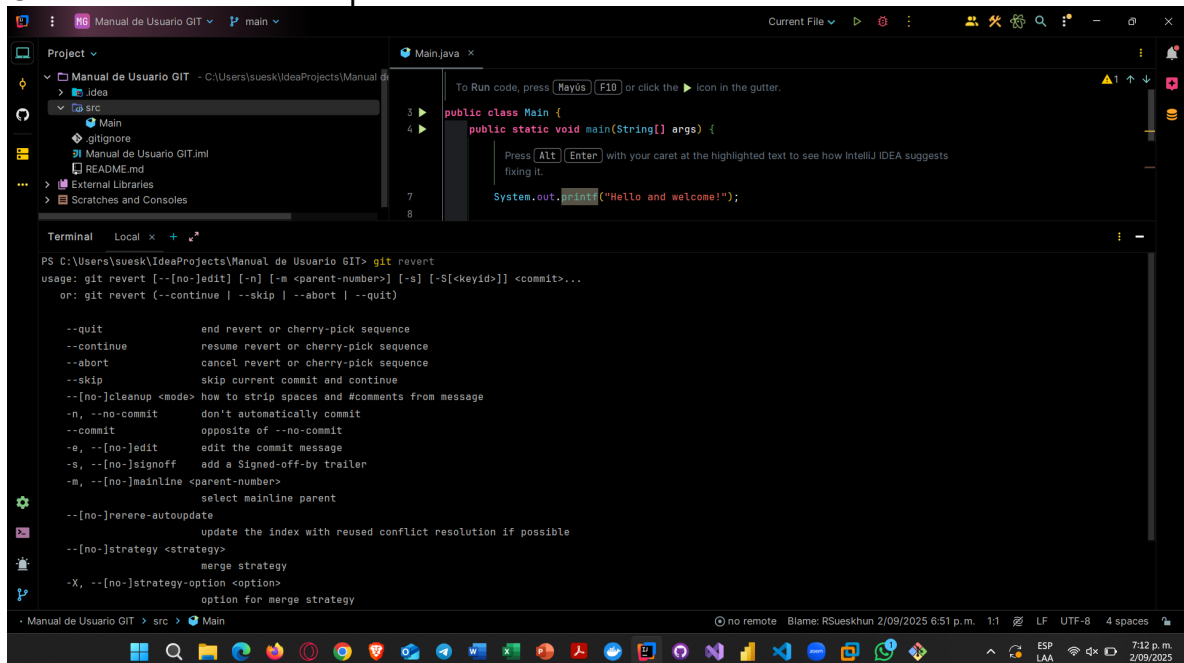


The screenshot shows the IntelliJ IDEA interface. The Project view on the left shows the 'Manual de Usuario GIT' project with a 'src' directory containing 'Main.java'. The Main.java file is open in the editor, showing a Java class with a main method. The terminal at the bottom shows the command 'git merge main' being executed, with the output 'Already up to date.'.

```
PS C:\Users\suesk\IdeaProjects\Manual de Usuario GIT> git merge main
Already up to date.
PS C:\Users\suesk\IdeaProjects\Manual de Usuario GIT>
```

Git revert

Crea un nuevo commit que deshace los cambios de un commit anterior.



The screenshot shows the IntelliJ IDEA interface. The Project view on the left shows the 'Manual de Usuario GIT' project with a 'src' directory containing 'Main.java'. The Main.java file is open in the editor, showing a Java class with a main method. The terminal at the bottom shows the command 'git revert' being executed, with the output 'usage: git revert [--no-edit] [-n] [-m <parent-number>] [-s] [--<keyid>] <commit>...'.

```
PS C:\Users\suesk\IdeaProjects\Manual de Usuario GIT> git revert
usage: git revert [--no-edit] [-n] [-m <parent-number>] [-s] [--<keyid>] <commit>...
or: git revert [--continue | --skip | --abort | --quit]

--quit          end revert or cherry-pick sequence
--continue      resume revert or cherry-pick sequence
--abort         cancel revert or cherry-pick sequence
--skip         skip current commit and continue
--[no-]cleanup <mode> how to strip spaces and #comments from message
-n, --no-commit don't automatically commit
--commit       opposite of --no-commit
-e, --[no-]edit edit the commit message
-s, --[no-]signoff add a Signed-off-by trailer
-m, --[no-]mainline <parent-number>
                select mainline parent
--[no-]rerere-autoupdate
                update the index with reused conflict resolution if possible
--[no-]strategy <strategy>
                merge strategy
-X, --[no-]strategy-option <option>
                option for merge strategy
```

CONCLUSIONES

1. Se logró desmitificar Git al presentar una estructura categorizada y de fácil acceso, que descompone comandos complejos en instrucciones sencillas. Esto reduce significativamente la barrera de entrada para nuevos desarrolladores y permite una consulta intuitiva, cumpliendo con el objetivo de hacer la tecnología accesible y menos intimidante.
2. El manual funciona como un pilar para la estandarización de procedimientos dentro de los equipos de desarrollo, al ofrecer definiciones claras y un lenguaje común. Al educar sobre el uso correcto de comandos críticos, se promueve un flujo de trabajo coherente y se minimizan los errores derivados de la malinterpretación o el uso incorrecto de la herramienta.
3. Se fortalece la autonomía del usuario al equiparlo con el conocimiento necesario para diagnosticar y resolver incidencias frecuentes por su propia cuenta.
4. La guía trasciende ser una simple lista de comandos al contextualizar su aplicación en escenarios reales, explicando no solo el "cómo" sino también el "cuándo" y el "porqué". Esto fomenta las buenas prácticas desde el inicio, ayudando a crear un historial de proyectos más limpio, comprensible y colaborativo.
5. El manual se consolida como una herramienta de productividad esencial que optimiza el tiempo y el esfuerzo invertido en el control de versiones. Al centralizar la información clave de forma clara y concisa, se elimina la necesidad de búsquedas fragmentadas en fuentes diversas, permitiendo a los desarrolladores mantener el flujo de trabajo y concentrarse en la resolución de problemas de valor.

REFERENCIAS BIBLIOGRAFICAS

- Chacon, S., & Straub, B. (2014). Pro Git (2da ed.). Apress. <https://git-scm.com/book/en/v2>
- Software Freedom Conservancy. (2022). Git reference manual. Git. <https://git-scm.com/docs>