### **Facundo Rios**

811874

Com3

### **Trabajo con Ramas**

**Realizar todos los pasos e ir pegando los resultados en un documento en Google drive accesible por todos los miembros del equipo.**

**Al finalizar el práctico copiar el enlace al documento y al github creado en un Notepad o Word que se subirá a la tarea.**

# **Parte 1: Gestión de Ramas y Fusiones Básicas**

## **Título: Creación y Fusión de Ramas en Git**

### **Objetivos:**

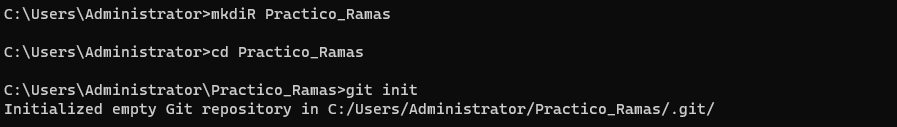
* Aprender a crear ramas en Git.
* Entender cómo fusionar ramas en un repositorio.

### **Pasos:**

1. **Inicializar un Repositorio**

Crea un directorio para tu proyecto e inicializa un repositorio Git:

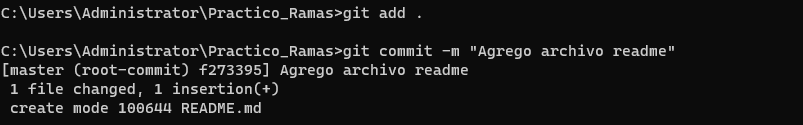
mkdir mi\_proyecto  
cd mi\_proyecto  
git init



1. **Crear un Archivo Inicial**

Crea un archivo README.md con contenido básico:

echo "Mi Proyecto" > README.md  
git add README.md  
git commit -m "Agregar archivo README.md"

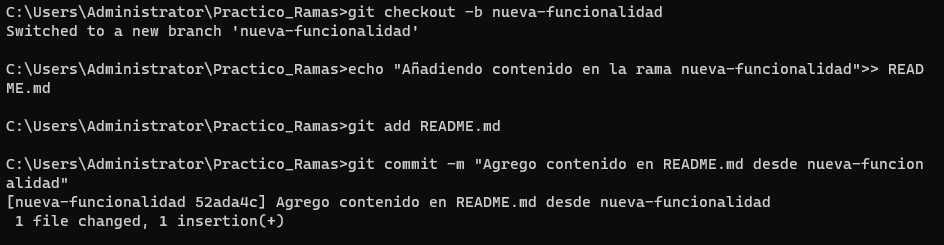
1. **Crear y Cambiar a una Nueva Rama**

Crea una nueva rama llamada nueva-funcionalidad y cámbiate a ella:

git checkout -b nueva-funcionalidad

1. **Modificar el Archivo en la Nueva Rama**

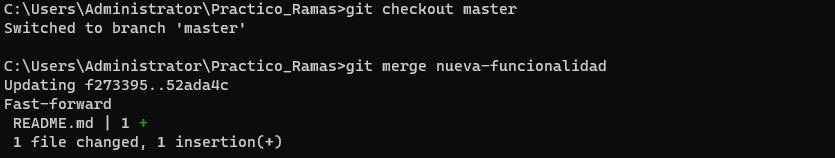
Edita README.md y agrega contenido en la nueva rama:

echo "Añadiendo contenido en la rama nueva-funcionalidad." >> README.md  
git add README.md  
git commit -m "Agregar contenido en README.md desde nueva-funcionalidad"  


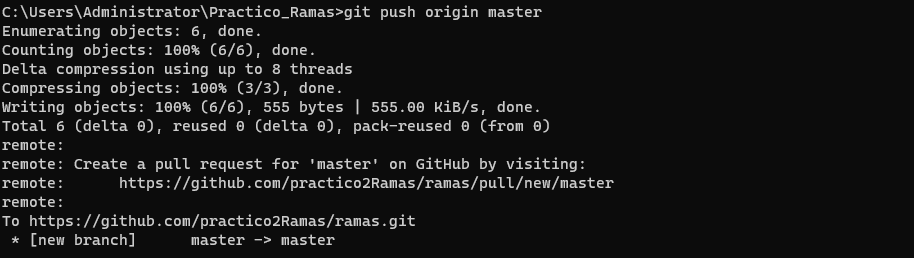
1. **Volver a la Rama Principal y Fusionar**

Cambia a la rama main y fusiona nueva-funcionalidad:

git checkout main  
git merge nueva-funcionalidad

git remote add origin https://github.com/ClaudiaRepository/PruebaGit  
git push origin main ( o master)  






# **Parte 2: Manejo de Conflictos en Git**

## **Título: Resolución de Conflictos en Fusiones de Git**

### **Objetivos:**

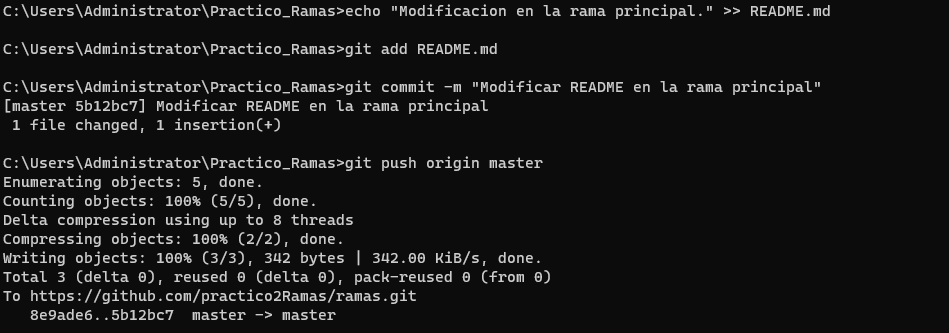
* Comprender cómo ocurren los conflictos en Git.
* Aprender a resolver conflictos y continuar con el flujo de trabajo.

### **Pasos:**

1. **Crear un Conflicto Potencial**

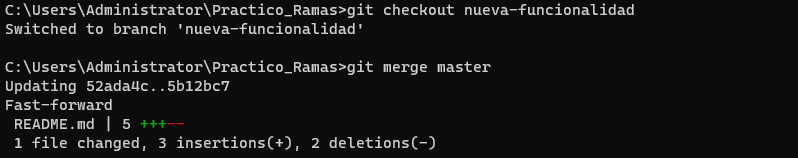
En la rama main, edita README.md para agregar contenido:

echo "Modificación en la rama principal." >> README.md  
git add README.md  
git commit -m "Modificar README.md en la rama principal"  
git push origin main



1. **Intentar Fusionar y Generar un Conflicto**

Cambia a la rama nueva-funcionalidad e intenta fusionarla con main:

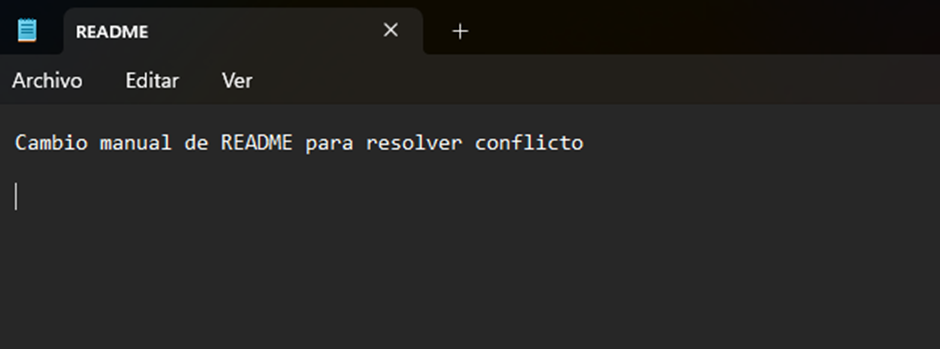
git checkout nueva-funcionalidad  
 git merge main  


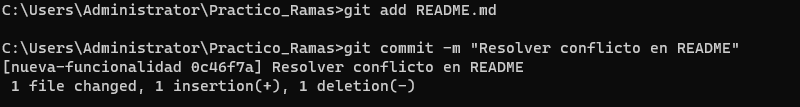
Deberías encontrarte con un conflicto en README.md.

1. **Resolver el Conflicto**

Abre README.md y edita las secciones marcadas para resolver el conflicto.

Después de resolverlo, confirma los cambios:

git add README.md  
git commit -m "Resolver conflicto en README.md"  




1. **Fusionar la Rama y Subir los Cambios**

Cambia a la rama main, fusiona nuevamente nueva-funcionalidad y sube los cambios:

git checkout main  
git merge nueva-funcionalidad  
git push origin main

# **Parte 3: Colaboración en GitHub con Organizaciones**

## **Título: Colaboración y Gestión de Proyectos en Organizaciones de GitHub**

### **Objetivos:**

* Aprender a crear y gestionar una organización en GitHub.
* Colaborar en proyectos compartidos utilizando herramientas de GitHub.

### **Pasos:**

1. **Crear una Organización en GitHub**

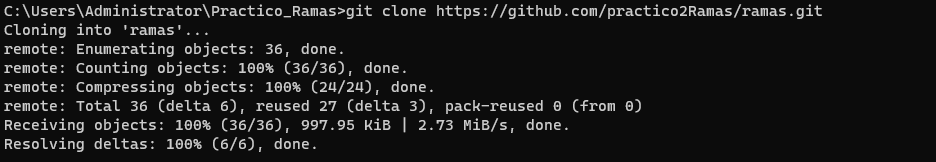
Ve a [GitHub Organizations](https://github.com/account/organizations) y crea una organización nueva. Añade miembros a la organización.

1. **Crear un Repositorio dentro de la Organización**

Crea un nuevo repositorio bajo la organización y otorga permisos a los miembros. Invitar a colaboradores y asignar permisos(En la pestaña Settings Buscar Manage Access primero Invite a Collaborator y luego con click derecho darle permisos ).

1. **Clonar el Repositorio a la Máquina Local**

Los miembros pueden clonar el repositorio a sus máquinas locales:

git clone <https://github.com/organizacion/nombre-repositorio.git>cd nombre-repositorio  




1. **Colaborar en el Repositorio**

Los miembros pueden crear ramas, hacer commits y enviar pull requests para colaborar en el proyecto.

1. **Revisar y Fusionar Cambios**

Utiliza GitHub para fusionar cambios en la rama principal del repositorio.

1. **Investigar más funciones**

Investiga alguna función vista en github y no vista en el práctico.

En esta parte voy a introducir otros comandos de git que son utilizados pero que no vimos en el practico

* *git log*: Muestra el historial de commits.
* *git diff*: Muestra las diferencias entre los archivos en el área de trabajo y el área de preparación.
* *git reset* <archivo>: Quita archivos del área de preparación.

