UNIVERSIDAD AUTÓNOMA DE BAJA CALIFORNIA

Ingeniería en Software y Tecnologías Emergentes

Práctica 0: Gestión de Repositorios con Git y GitHub

Materia: Paradigmas de programación

ALUMNO: Daniel Rios Rodriguez

MATRÍCULA: 372800

GRUPO: 941

Ensenada, Baja California a 29 de marzo de 2024.

Introducción:

En este reporte de práctica exploraremos la gestión de repositorios utilizando Git y GitHub. Comenzaremos creando y verificando una cuenta en GitHub, instalando y verificando Git, y estableciendo el entorno de trabajo necesario. Luego, inicializaremos un repositorio local, crearemos una llave pública/privada para conexión segura con GitHub, y configuraremos un repositorio en la plataforma. Abordaremos el clonado del repositorio, la gestión de ramas para cambios eficientes, la restauración de cambios y la fusión de ramas. A través de este proceso, obtendremos una comprensión integral de cómo colaborar efectivamente en proyectos de desarrollo de software utilizando estas herramientas de control de versiones.

Desarrollo:

Crear / Verificar cuenta de Github

- Abre tu navegador web y ve a Github.com.
- Haz clic en "Sign up" si aún no tienes una cuenta. Si ya tienes una cuenta, inicia sesión.

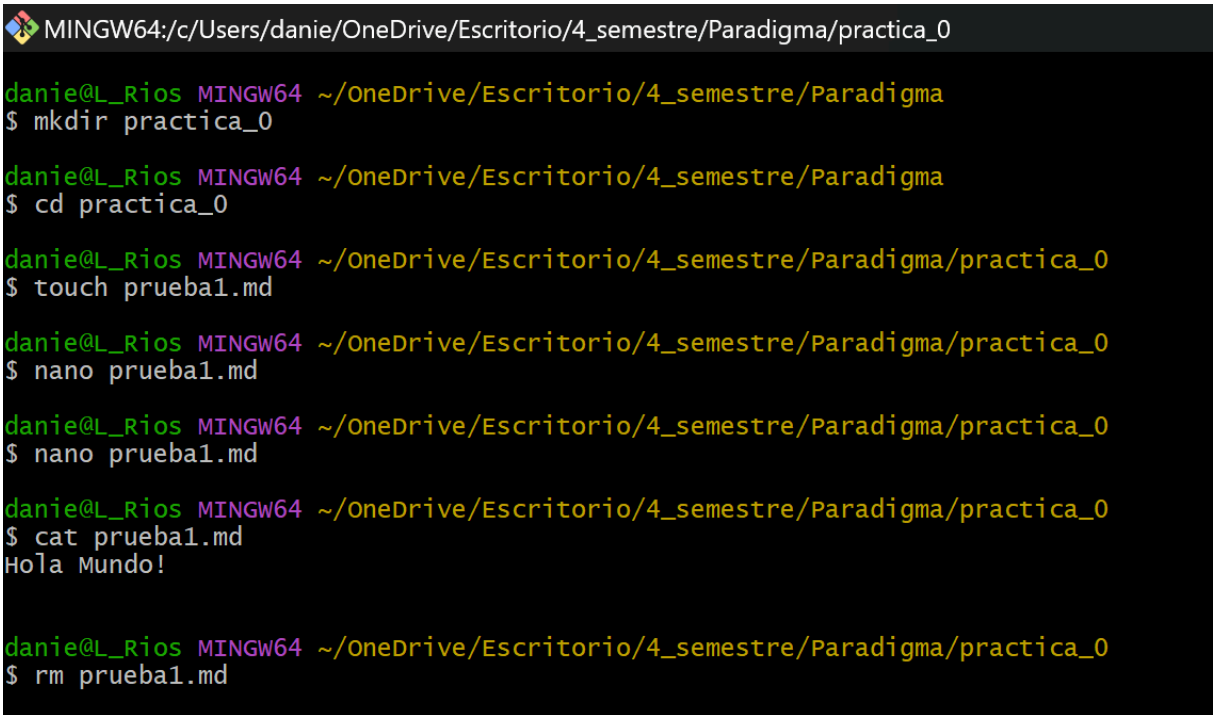
Instalar / Verificar Git

- Verifica la instalación ejecutando git --version en tu terminal.
- Si aún no tienes Git instalado, descárgalo e instálalo desde git-scm.com.

Crear folder del proyecto

- Decide dónde deseas almacenar tu proyecto en tu sistema de archivos.
- Crea una nueva carpeta para tu proyecto si aún no tienes una. ● En la terminal, en este caso bash se pueden ejecutar los siguientes comandos (ejemplo):

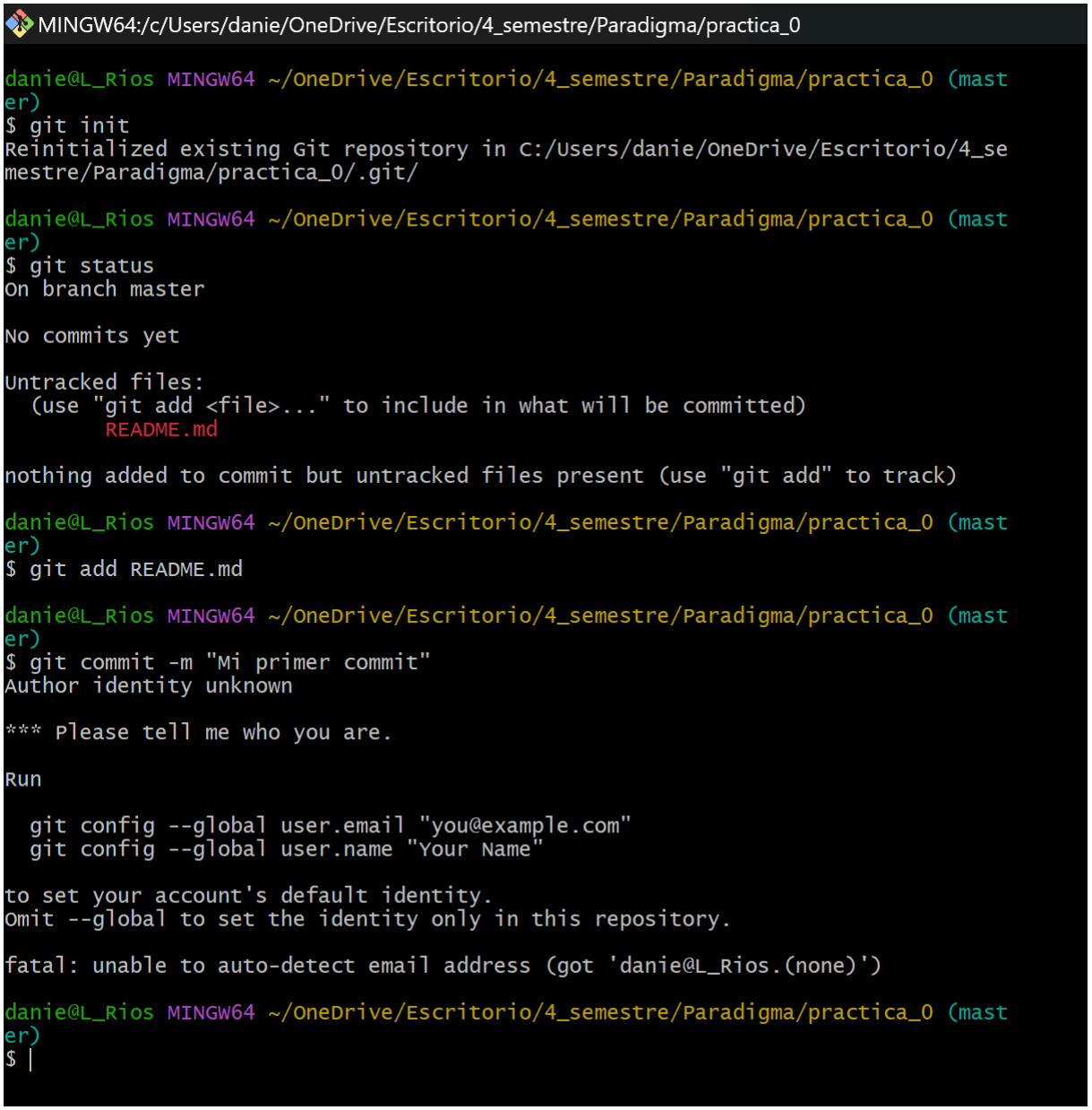
```
$mkdir Pparadigmas/practica0 // Crear carpetas

$cd Pparadigmas/practica0 // Cambiar directorio

$touch prueba.md // Crear archivo

$nano prueba.md // Abrir archivo en editor de texto $cat prueba.md

// Leer archivo en terminal $rm pureba.md // Eliminar archivo

$code README.md // Abrir archivo en VSCode
```



Inicializar repositorio

- Abre tu terminal o símbolo del sistema.
- Navega hasta la carpeta de tu proyecto usando el comando cd. ● Ejecuta git init para inicializar un nuevo repositorio Git en la carpeta.

```
$cd .. // Volver al directorio anterior (atras) $git init //
Inicializar repositorio local de git $git status // Checar el estado de
los archivos en el repositorio $git add README.md // Añadir archivo a
los archivos rastreados $git commit -m " mi primer commit" // Crear una
captura de los cambios
```



Crear llave pública / privada

- Si aún no tienes una llave SSH, genera una usando el comando ssh-keygen.
- Localizar el archivo creado y leerlo usando el comando cat.

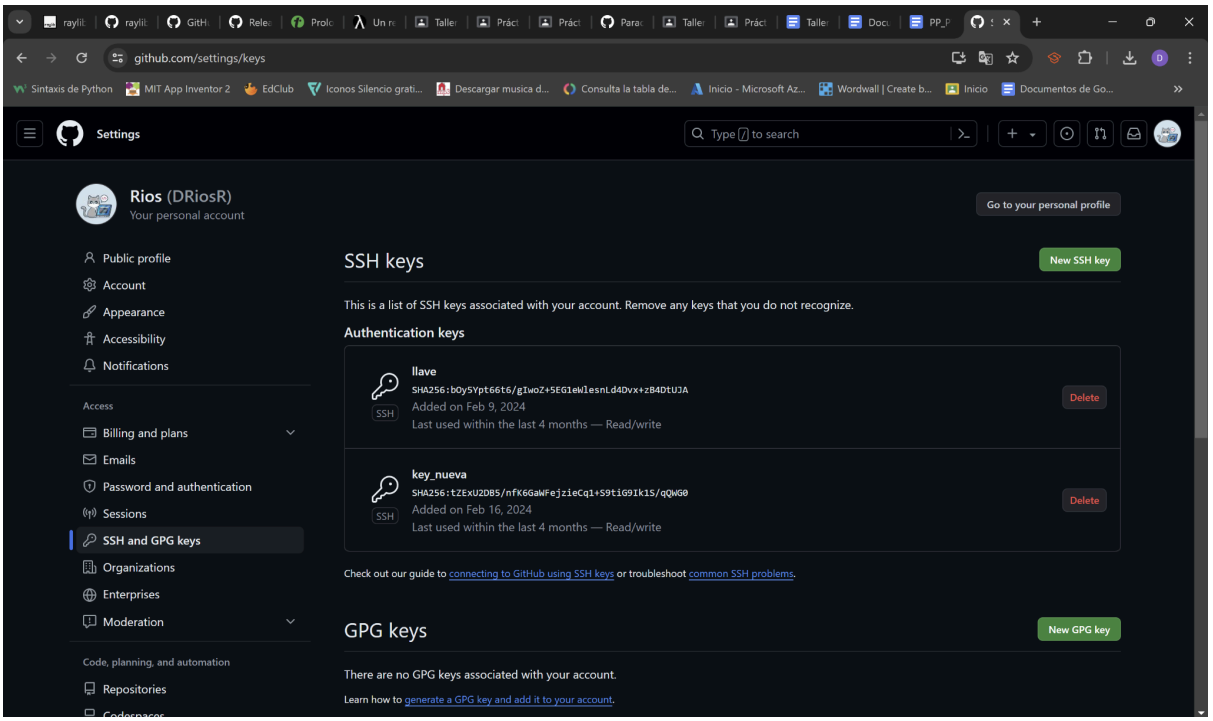
```
$ssh-keygen

$ls ~/.ssh

$cat ~/.ssh/id_ed25519.pub
```

Crear repositorio en Github

● Inicia sesión en tu cuenta de Github.
● Haz clic en el signo más en la esquina superior derecha y selecciona "New repository".
● Dale un nombre a tu repositorio, establece la visibilidad y otras configuraciones según tu preferencia.
● Haz clic en "Create repository".

Agregar llave a Github

● Copia el contenido de tu llave pública creada anteriormente.
● Ve a la configuración de tu cuenta de Github haciendo clic en tu avatar en la esquina superior derecha y seleccionando "Settings".
● En el menú de la izquierda, haz clic en "SSH and GPG keys". ● Haz clic en "New SSH key".
● Pega el contenido de tu llave pública en el campo "Key" y dale un nombre descriptivo.
● Haz clic en "Add SSH key" para guardar.



Clonar repositorio

● Ve a Github y haz clic en "Code", de ahí copia el código de SSH. ● En tu terminal, navega hasta la carpeta donde deseas clonar el repositorio. ● Ejecuta git clone seguido del código de SSH que copiaste.

```
$mkdir temp // Crear la carpeta temporal $cd temp //

Cambiar directorio a temp

$git clone git@github.com:LizBarrios/ParadigmasP.git // Clonar
repositorio

$cd ParadigmasP // Cambiar directorio
```

Crear branch

● Crea un nuevo branch usando el comando git checkout -b seguido del nombre del branch.
● Verifica que estás en el branch correcto usando git status.

```
$git checkout -b temp // Crear branch

$git status // Verificar branch
```

● Realiza cambios en tu proyecto y haz un commit.

```
$code . // Abrir proyecto en VSCode

$git add README.md // Añadir archivo a los archivos rastreados $git commit

-m "Modificar README" // Crear captura de cambios
```

● Sube tu branch a Github usando el comando git push seguido de -u origin y el nombre del branch. (-u origin solo se usa la primera vez, después solo se usa git push)

```
$git push -u origin temp // Subir branch a Github
```

Restaurar cambios

● Haz un cambio pero no lo añadas al stage.
● Usa el comando git restore seguido de un punto para eliminar los cambios. 6

```
$git restore . // Eliminar cambios
```

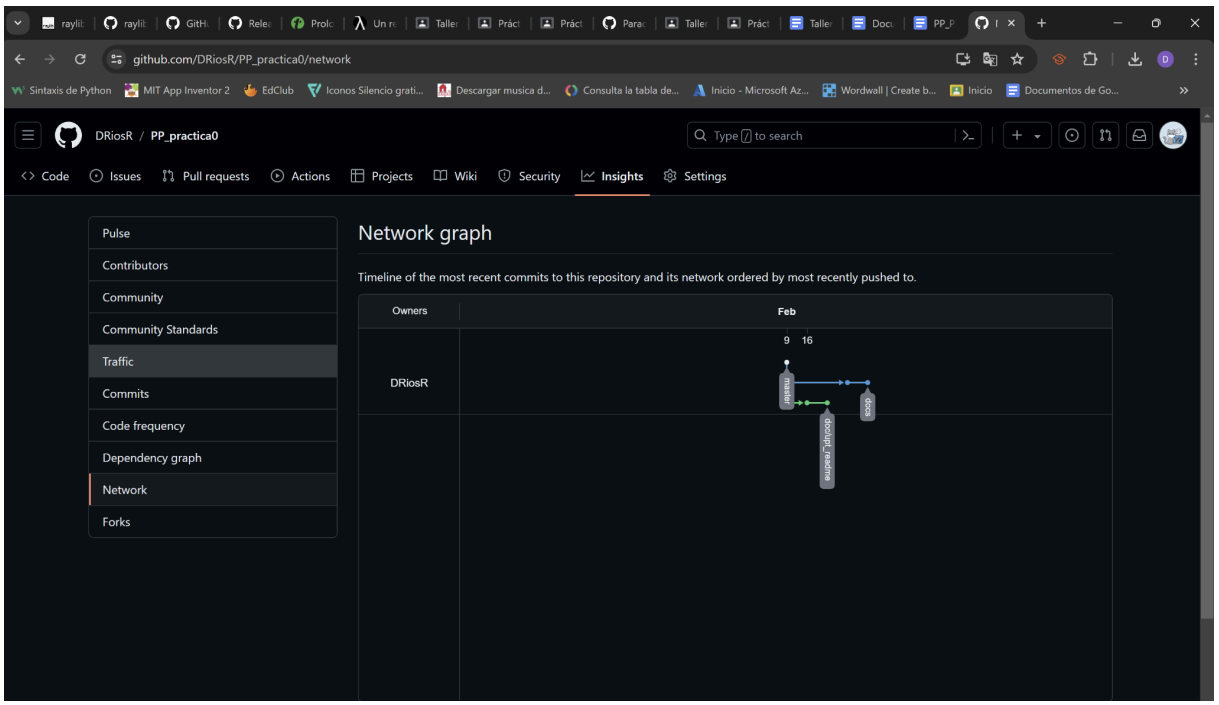
- Si añadiste el cambio, debes ejecutar lo siguiente:

```
$git restore --staged README.md
```

Fusionar branch

- Cambia al branch que deseas fusionar.
- Ejecuta git merge seguido del nombre del branch que deseas fusionar.
-

```
$git checkout main

$git merge temp
```



Repositorio de GitHub

```
git@github.com:DRiosR/PP_practica0.git
```

Conclusión:

La práctica inicial ha sido fundamental para comprender las herramientas esenciales del desarrollo colaborativo de software. Exploramos desde la creación de cuentas en GitHub hasta la gestión de ramas y la integración de cambios. Estas herramientas no solo facilitan la organización del código, sino que también fomentan una colaboración efectiva entre desarrolladores, mejorando así la eficiencia del equipo en entornos profesionales. Aunque algunos puedan preferir interfaces gráficas, personalmente encontré que la línea de comandos ofrecía una eficiencia superior en estos procesos.