

Git and GitHub

Nicolás García Peñaloza

INCORPORACIÓN EN R Y PYTHON (COMING SOON TO STATA).

10 DE MARZO, 2025

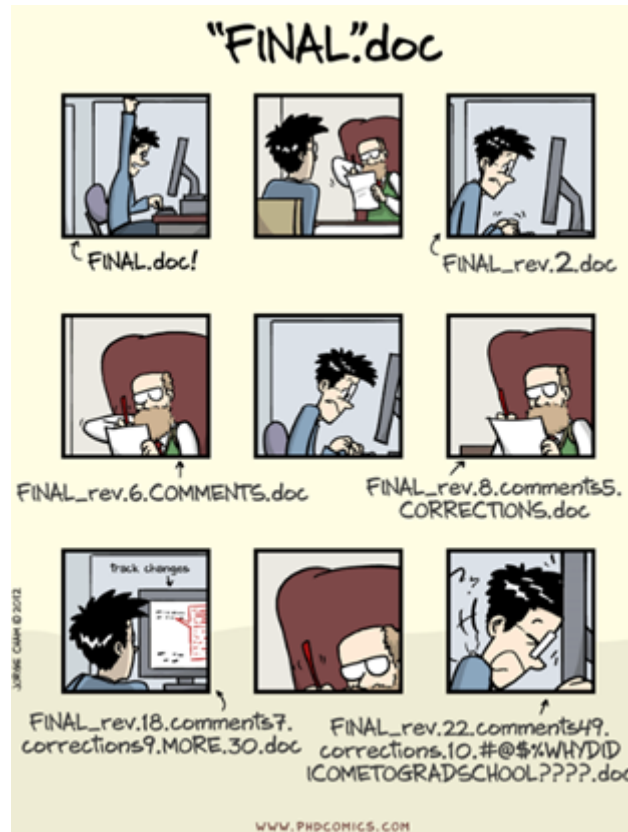
Resumen

El desarrollo de proyectos en análisis de datos exige no solo el dominio de herramientas de vanguardia, sino también la aplicación rigurosa de buenas prácticas en medición. La calidad y el profesionalismo en este campo emergen de una sinergia entre el poder computacional, la organización meticulosa y una sólida gobernanza de datos. Así, el análisis de los datos deja de ser solo técnica y se convierte en una forma de pensamiento estratégico.

Índice

1. Git and GitHub	2
2. R en Rstudio	4
3. Python	14

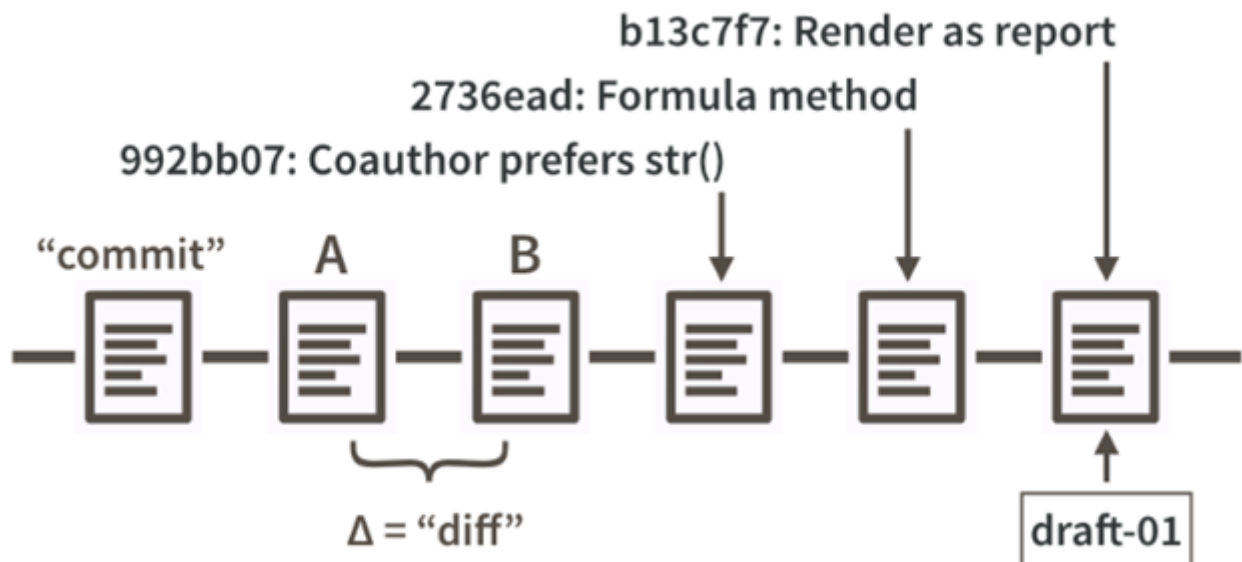
1. Git and GitHub



El desarrollo de proyectos que implican la programación de aplicaciones, software o análisis de datos requiere un control riguroso de las versiones generadas. Esto garantiza una adecuada estructuración y documentación del código, además de facilitar el trabajo colaborativo. Para ello, es fundamental incorporar un sistema de control de versiones estándar y moderno. [Git](#), un sistema de control de versiones distribuido, estándar en el mundo del desarrollo, es una excelente opción que permite gestionar de manera eficiente la evolución de un conjunto de archivos, llamado repositorio, de forma ordenada y coherente.

[Git](#) es un proyecto de código abierto robusto, mantenido de manera activa, desarrollado por Linus Torvalds, el creador del kernel de Linux. Su principal función es almacenar el historial de versiones de un proyecto, lo que facilita tanto el seguimiento de cambios como la colaboración entre equipos.

Para comprender el funcionamiento de Git, es esencial familiarizarse con tres conceptos clave: repositorios, commits y diffs. Los repositorios son los contenedores donde se almacena toda la información del proyecto. Los commits son puntos específicos del historial que documentan los cambios realizados, junto con una descripción clara de los mismos. Por último, las diffs muestran las diferencias entre versiones, destacando los cambios realizados de una versión a otra.



Ahora, exploremos [GitHub](#), una plataforma que actúa tanto como red social para desarrolladores como un servicio de gestión de proyectos en la nube. GitHub permite almacenar y gestionar proyectos de manera remota, con la opción de hacerlos públicos o privados según lo decida el contribuyente. Es comparable a servicios de almacenamiento como Dropbox, pero específicamente diseñado para el desarrollo y la colaboración en proyectos de programación, con funcionalidades mucho más avanzadas.

GitHub utiliza el sistema de control de versiones de Git, lo que significa que puedes trasladar tu proyecto desde un entorno local a la nube sin perder la capacidad de gestionar versiones, colaborar con otros y realizar un seguimiento detallado de los cambios.

Este servicio ofrece una versión gratuita que incluye almacenamiento de hasta 1 GB para [repositorios](#), lo que lo convierte en una excelente opción para proyectos pequeños y medianos. ¿Ahora Git y GitHub son lo mismo? No. Aunque están estrechamente relacionados, no son lo mismo. Git es la herramienta que gestiona versiones localmente, mientras que GitHub es una plataforma que extiende sus funcionalidades al entorno colaborativo en la nube. En conjunto, forman una poderosa combinación para el desarrollo moderno.

Git	GitHub
Es un software.	Es un servicio.
Es un sistema de control de versiones para gestionar el historial del código fuente.	Es un servicio de alojamiento para repositorios Git.
Es mantenido por linux.	Es mantenido por Microsoft.
Se instala localmente en el sistema.	Está alojado en la web.
No tiene ninguna función de gestión de usuarios.	Incorpora una función de gestión de usuarios.
Open-source.	Incluye un nivel gratuito y un nivel de pago por uso.
Tiene una configuración mínima de herramientas externas.	Tiene un mercado activo para la integración de herramientas.

Un vez tengamos instalados los programas llámese [R \(1\)](#), [Python \(1,2,3,4,5\)](#) y sus respectivas IDE con las que se sienta a gusto para trabajar, [debemos instalar Git](#) y crear una cuenta en [GitHub](#). En el terminal del sistema operativo podemos actualizar Git con (git update) para ver la versión puede escribir (git - -version)

Recuerde: Inicialmente instalamos Git y debemos tener una cuenta en GitHub

ver estos videos para entender de fondo GIT: [Link](#), [Link 2](#), [Link 3](#)

1. Abrimos en el equipo el simbolo del sistema (cmd y/o terminal) o buscamos en el equipo *git bash*

o *git CMD* una vez se abrió alguno de los dos escribimos *git --version* para conocer la versión con la que contamos, luego introducimos a Git:

```
git config - -global user.name "NicolasGP01"
```

```
git config - -global user.email "nicolasgp0109@gmail.com"
```

```
git config - -global - -list
```

```
git config - -global user.name
```

```
git config - -global user.email
```

**Utilizamos generalmente las credenciales del nombre y correo que tenemos en GitHub*

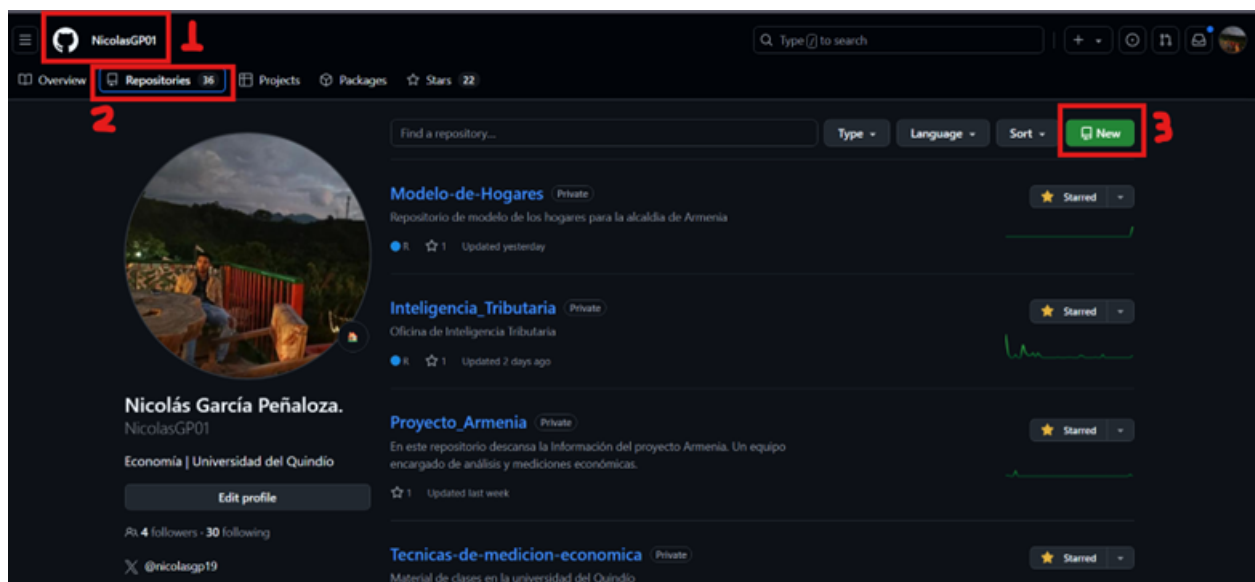
2. R en Rstudio

Cómo lo integramos con R en Rstudio: En este caso debemos contar con R y Rstudio instalados en nuestros equipos. [URL](#)

Otra alternativa para establecer nuestros alias en GIT es desde Rstudio con el paquete "usethis" que puede establecer su nombre de usuario Git y correo electrónico desde dentro de R:

```
1 library(usethis)
2 usethis::use_git_config(user.name = "NicolasGP01" , user.email = "
  nicolasgp0109@gmail.com")
```

Creando la copia del repositorio GITHUB



Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere? [Import a repository.](#)

Required fields are marked with an asterisk ().*

Repository template

No template

Start your repository with a template repository's contents.

Owner * NicolasGP01

Repository name * Fondo_de_Inversion Fondo_de_Inversion is available

Great repository names are short and memorable. Need inspiration? How about [scaling-garbanzo](#)?

Description (optional)

Este repositorio contiene el estudio para un fondo de inversión

Public Anyone on the internet can see this repository. You choose who can commit.

Private You choose who can see and commit to this repository.

Initialize this repository with:

☒ Add a README file

This is where you can write a long description for your project. [Learn more about READMEs.](#)

Add .gitignore

.gitignore template: None

.gitignore template

Filter...

OpenCart

OracleForms

Packer

Perl

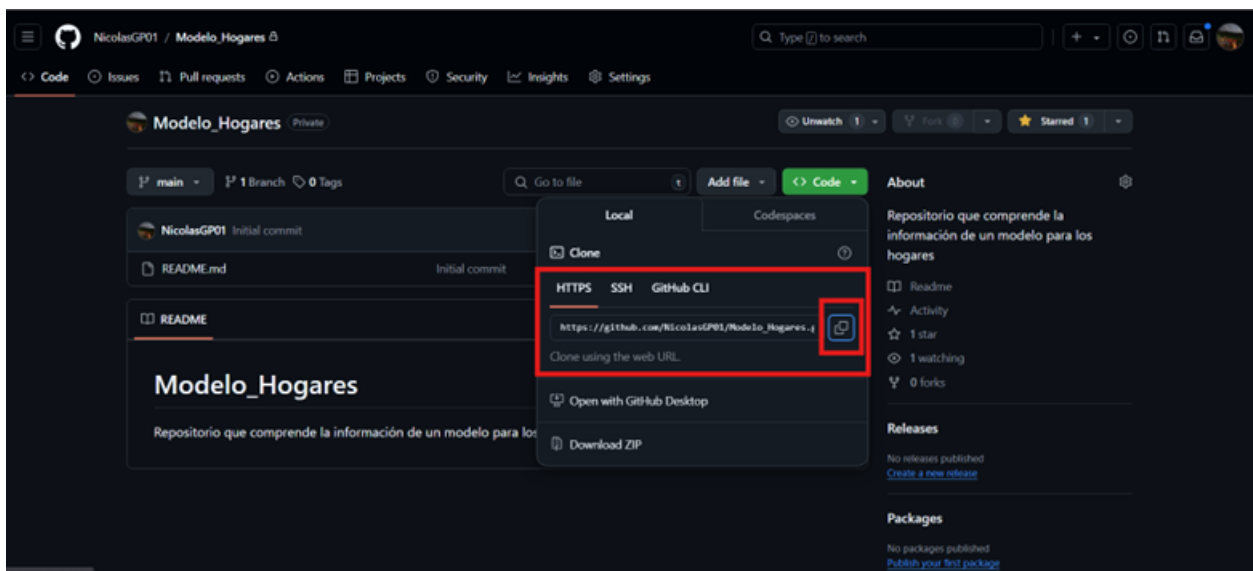
Phalcon

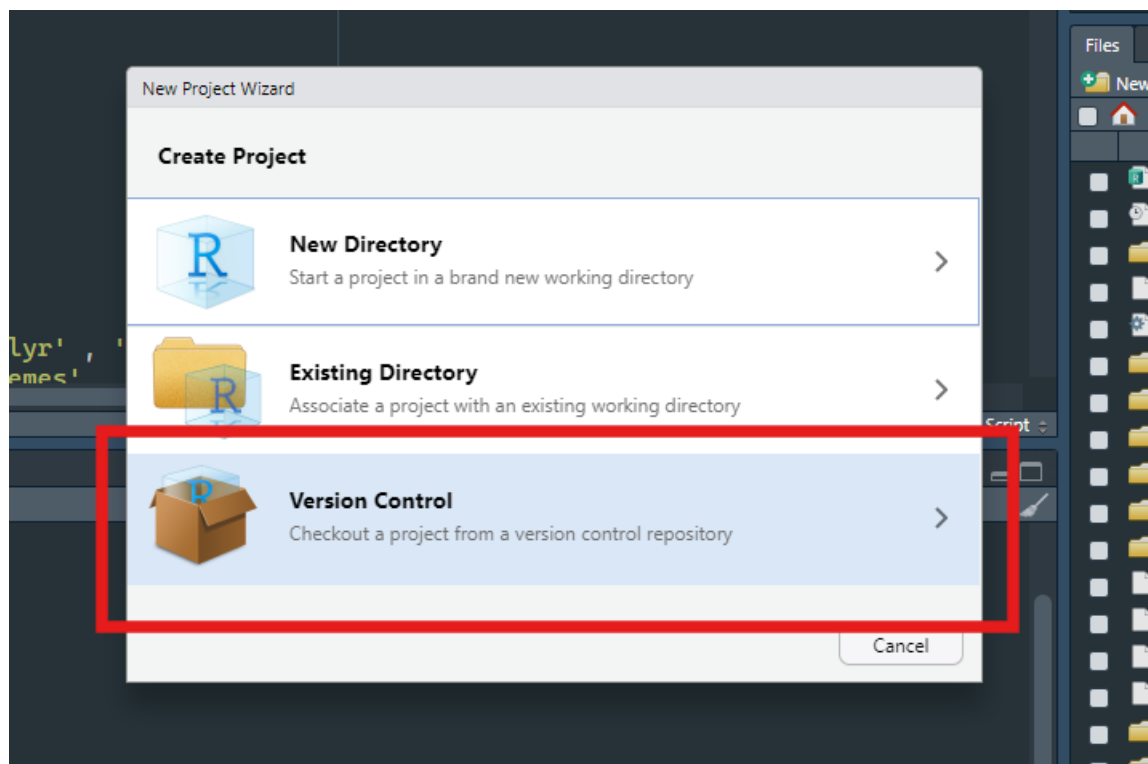
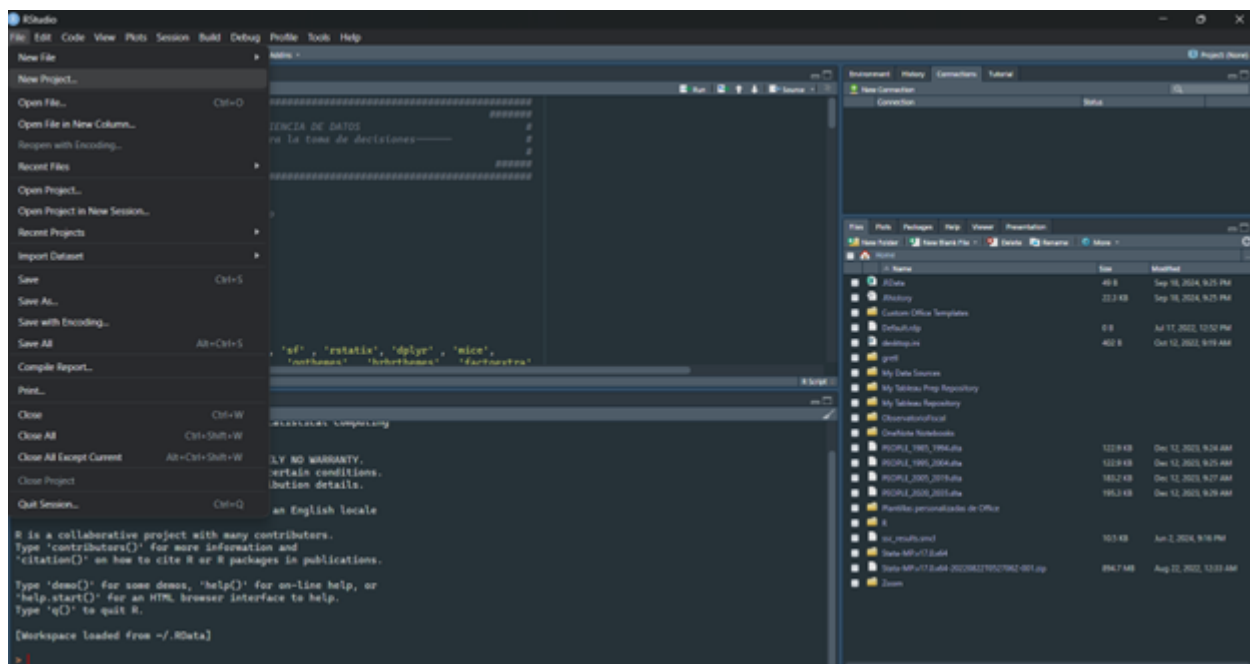
PlayFramework

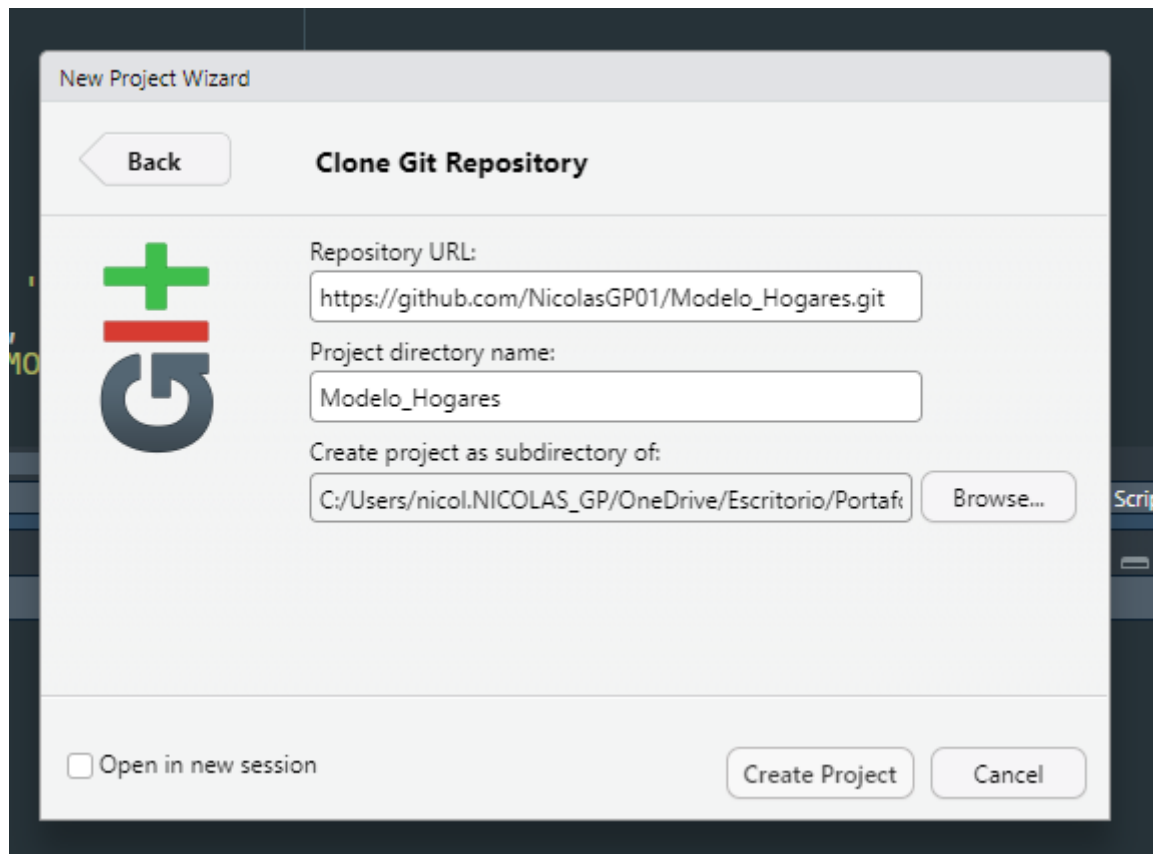
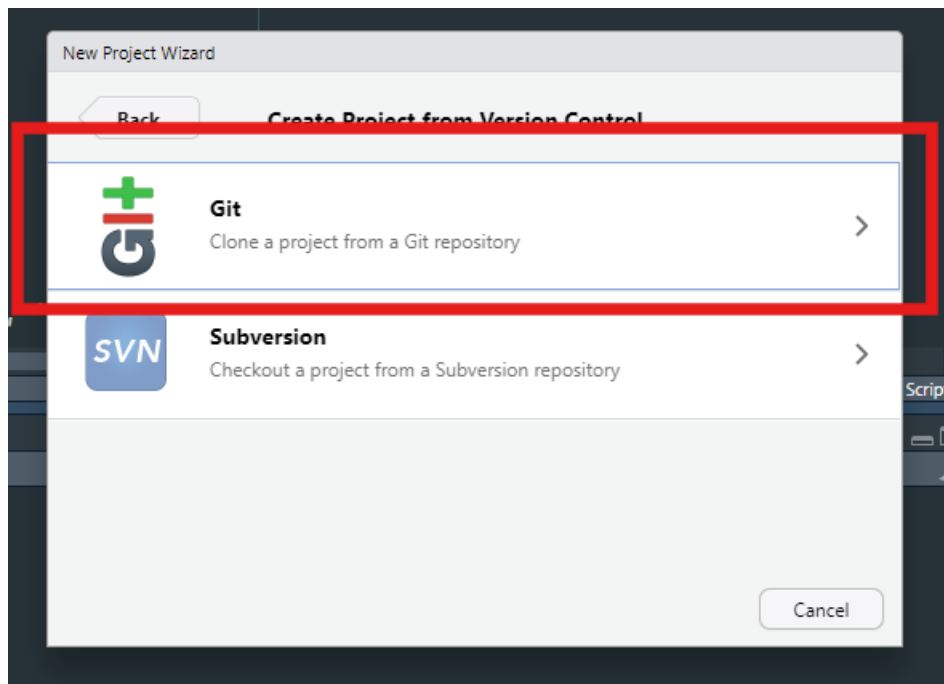
Plone

Prestashop

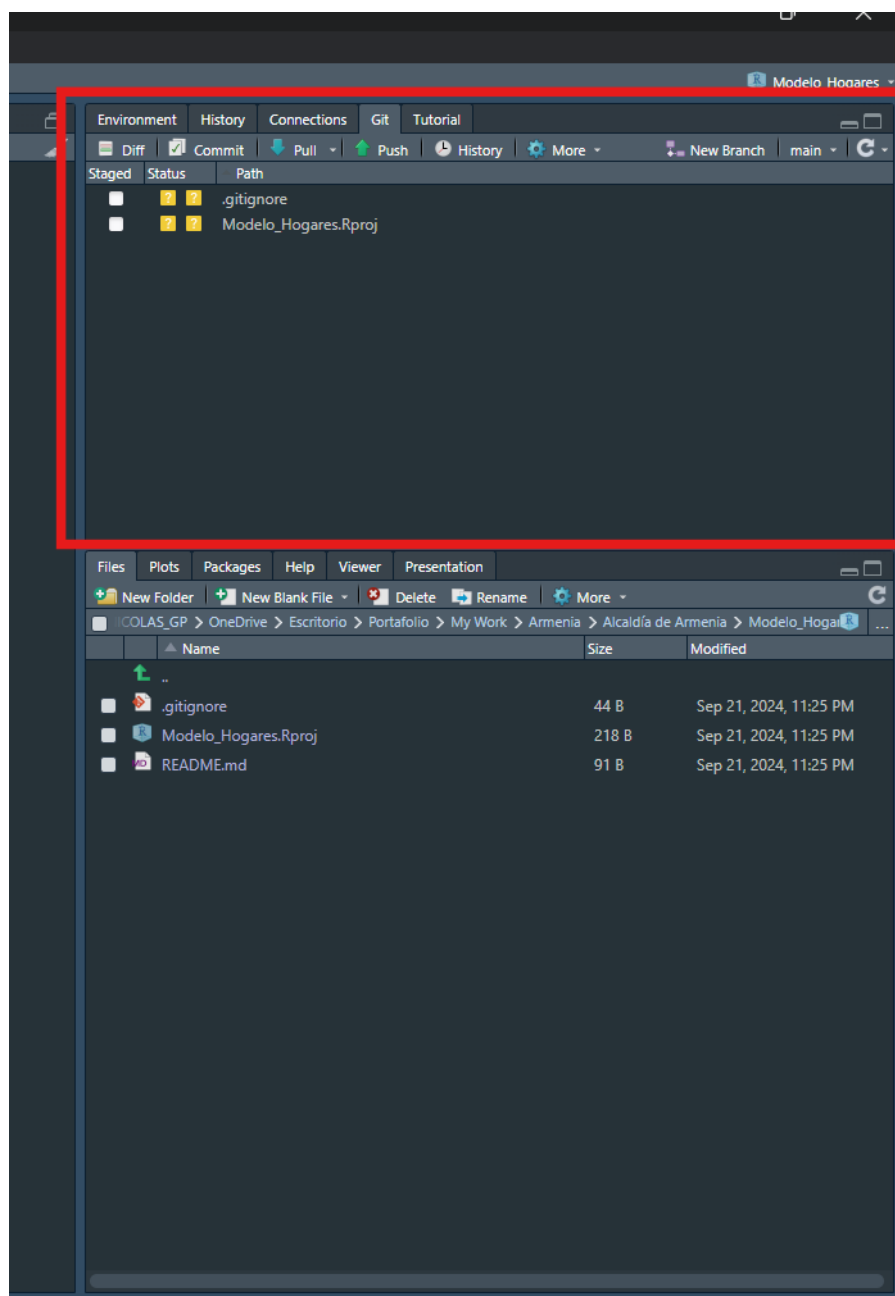
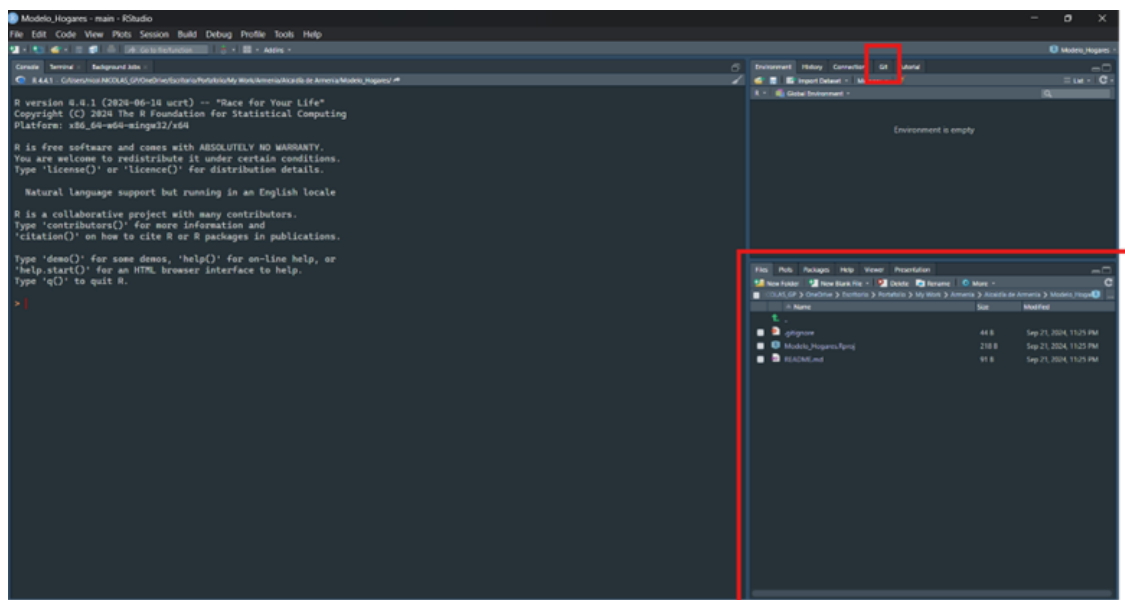
Create repository

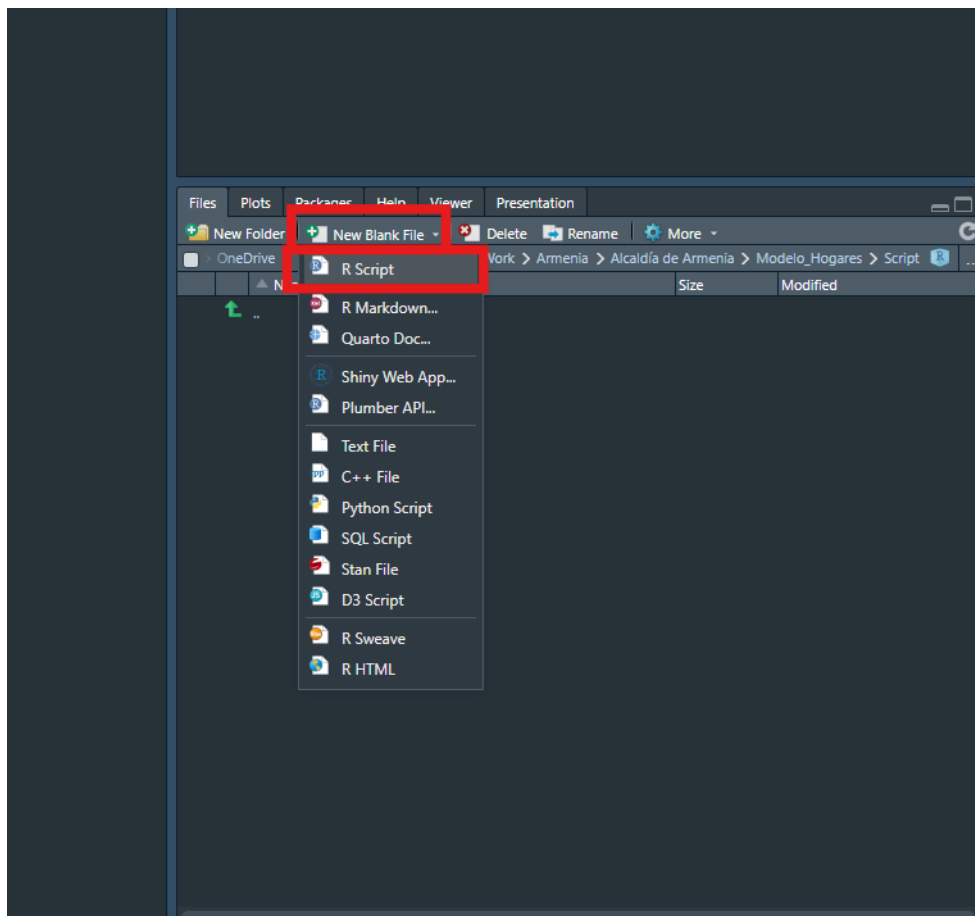
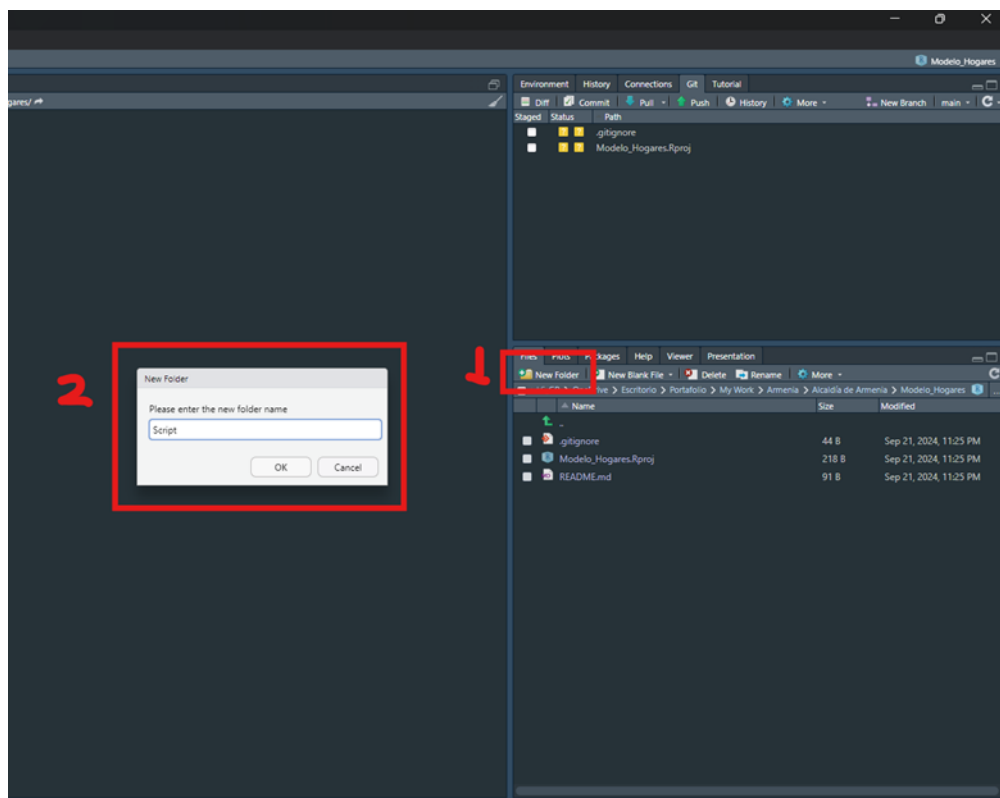


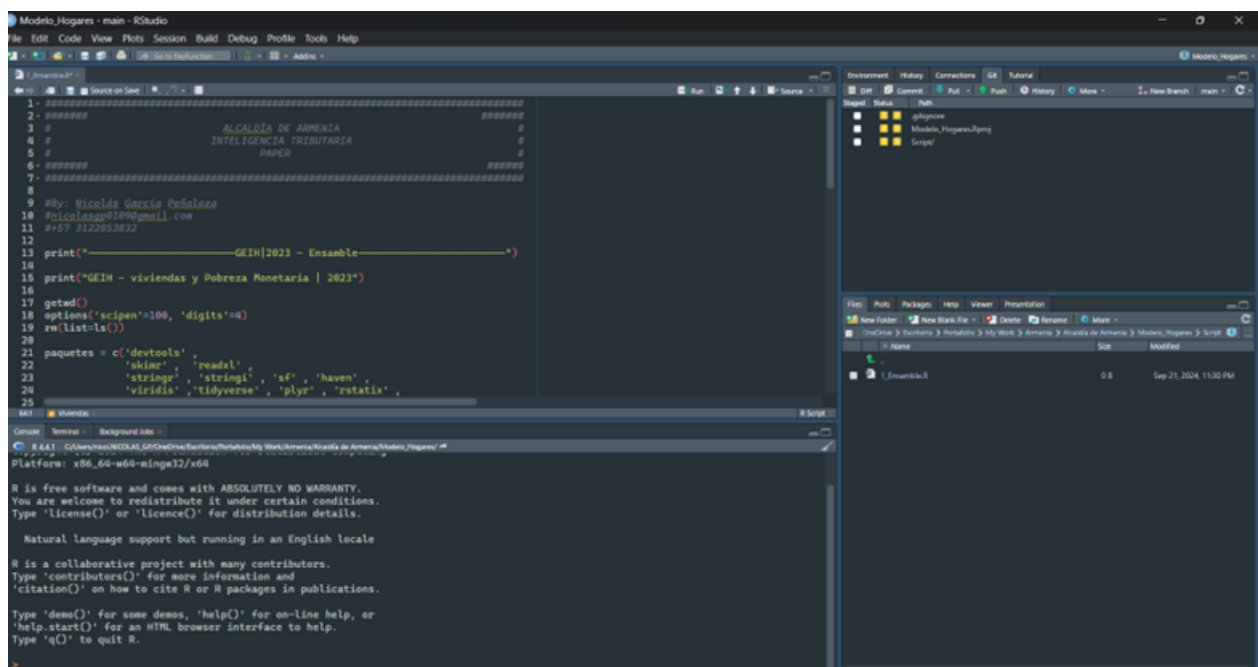
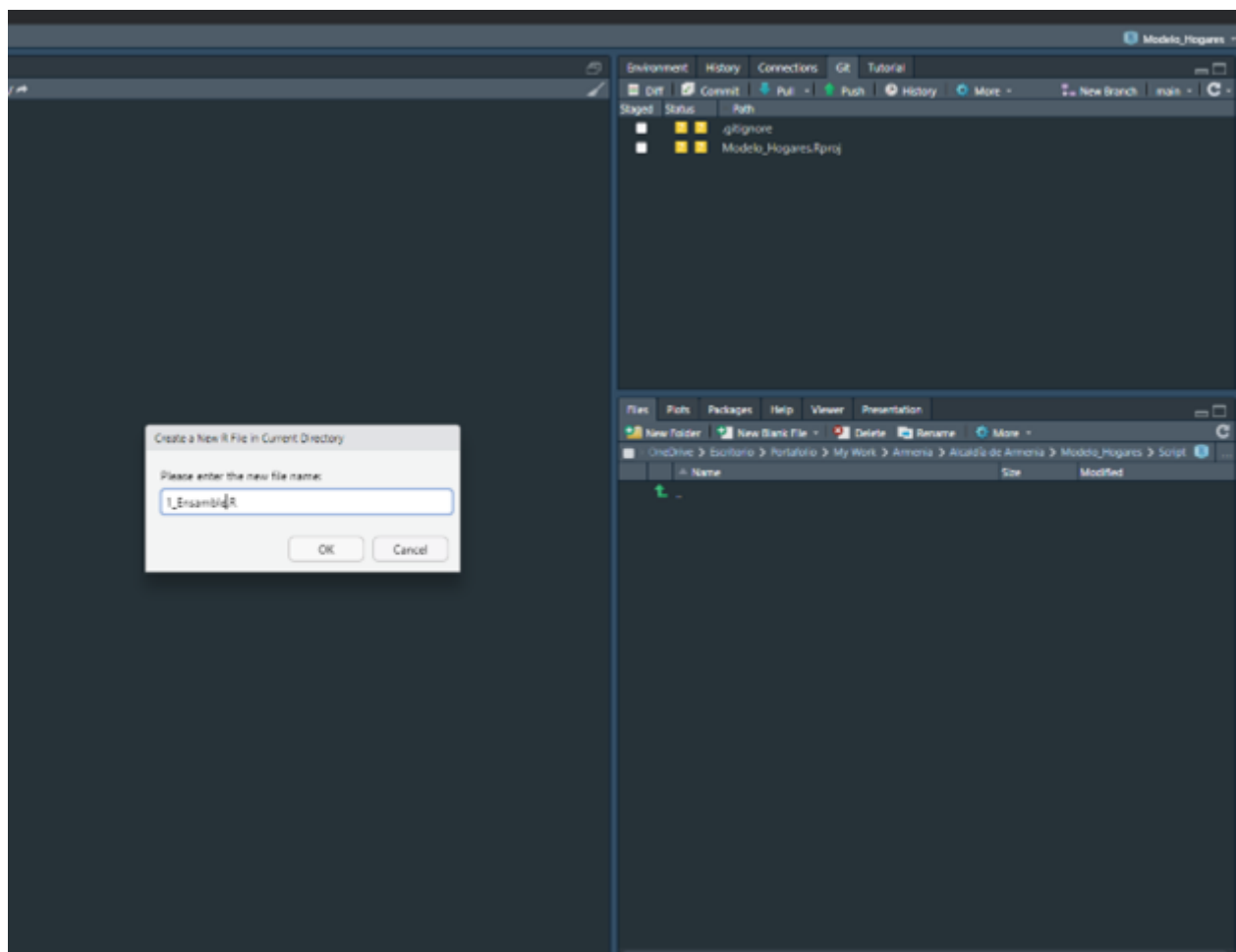


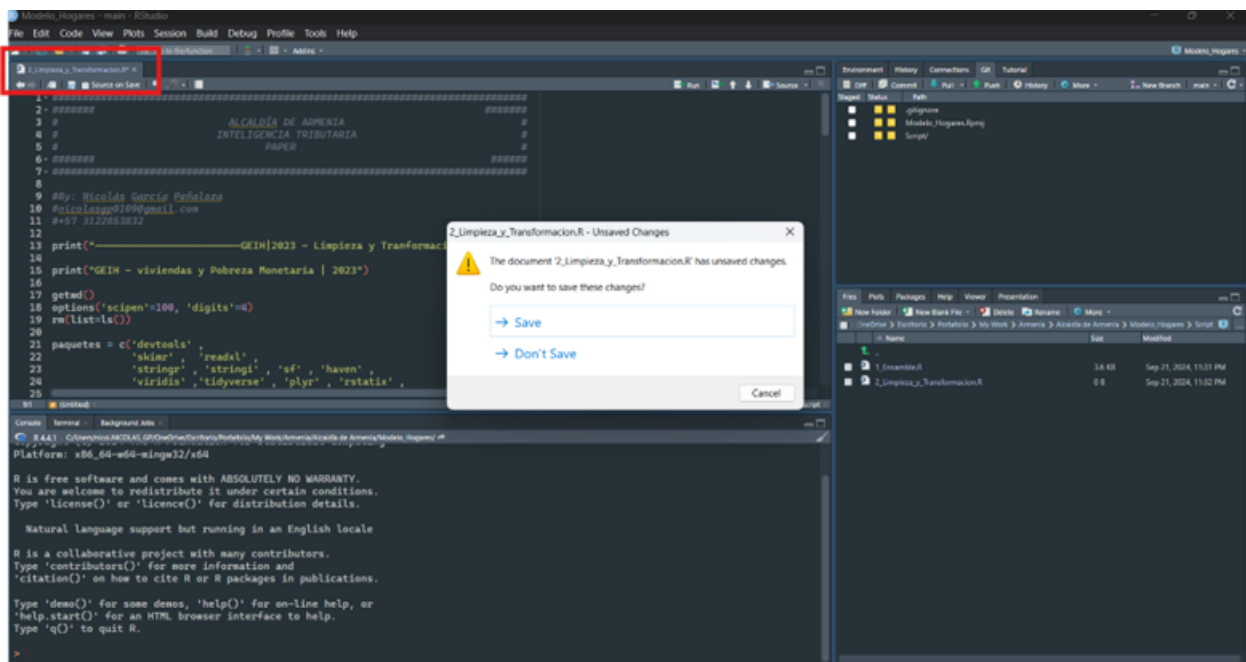


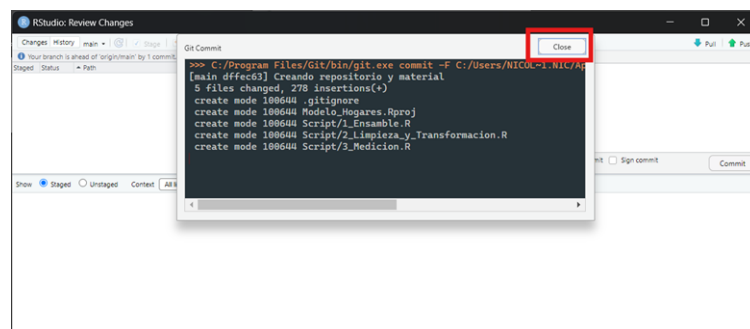
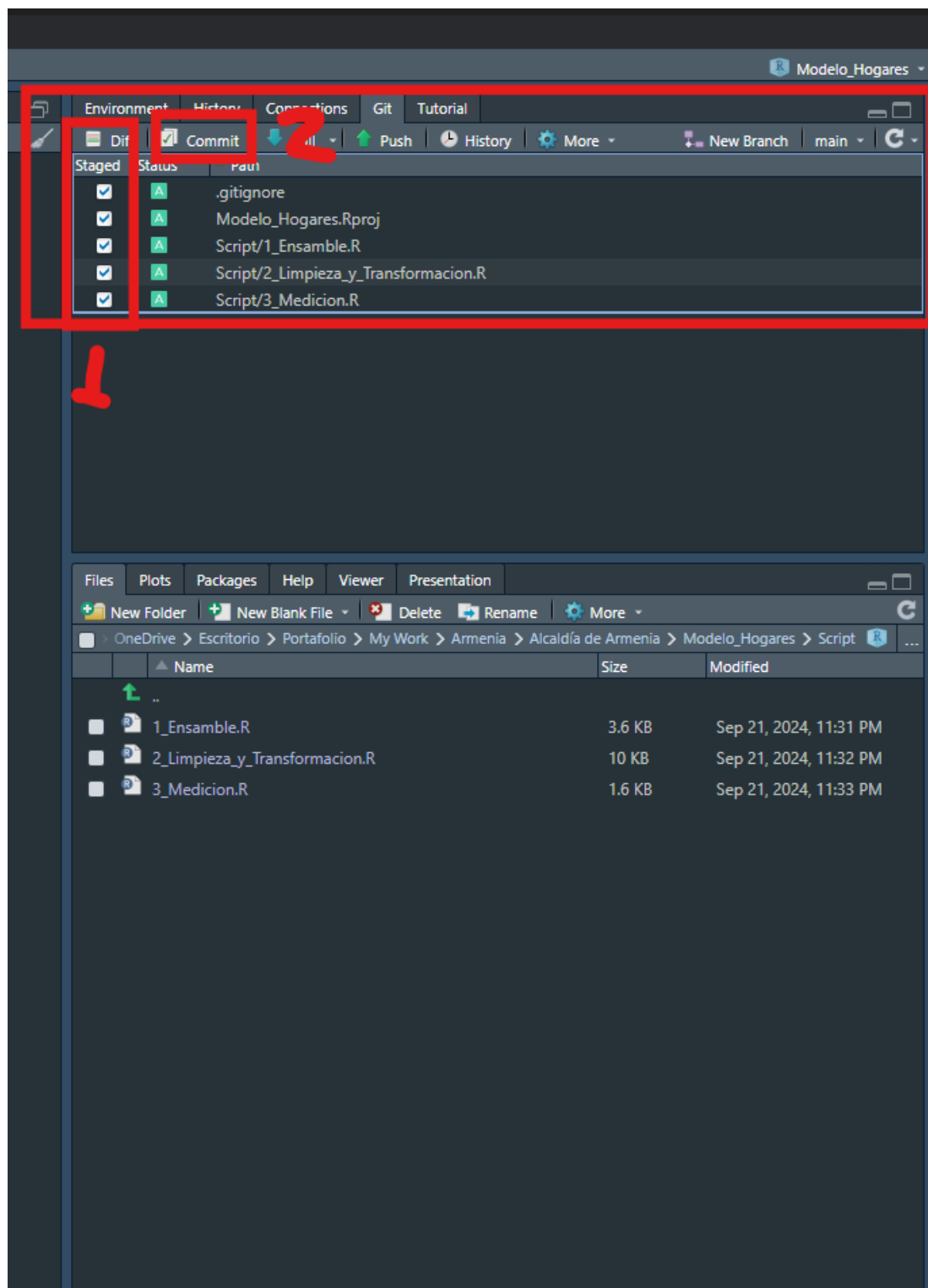
Inicialmente nos puede pedir autenticación

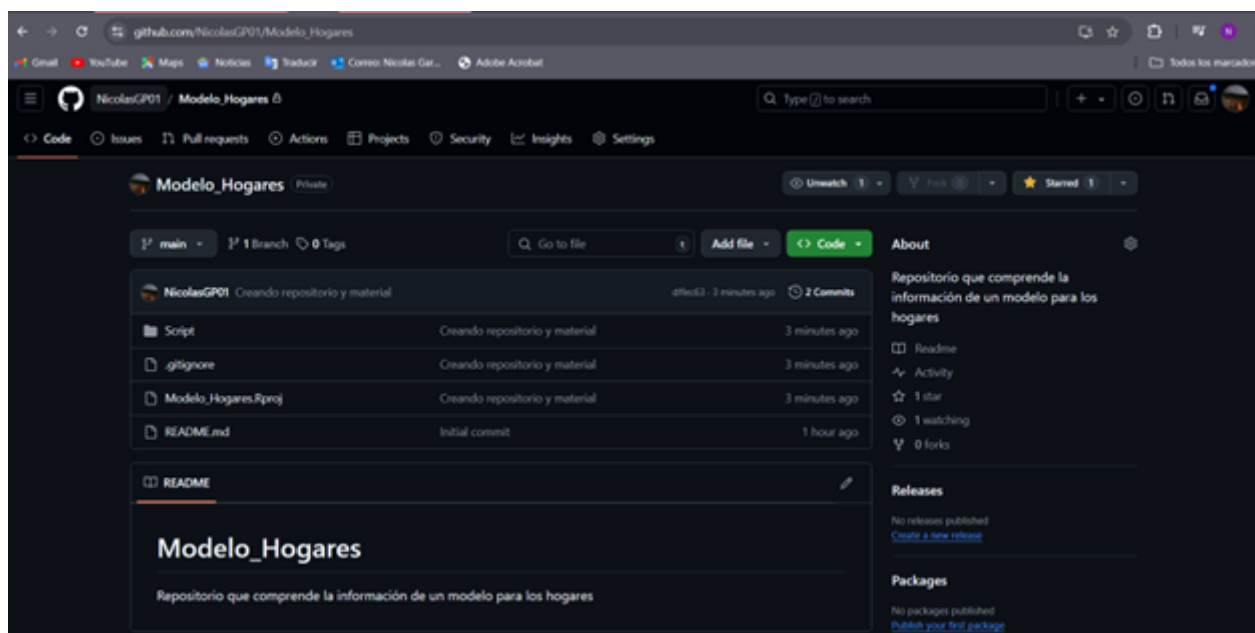
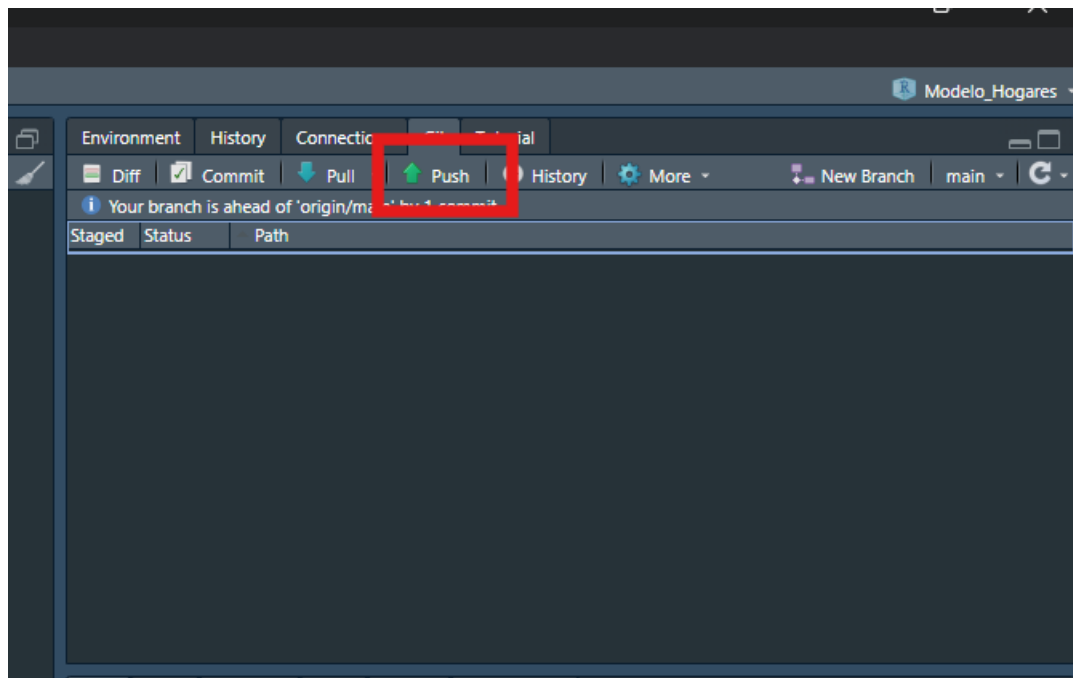


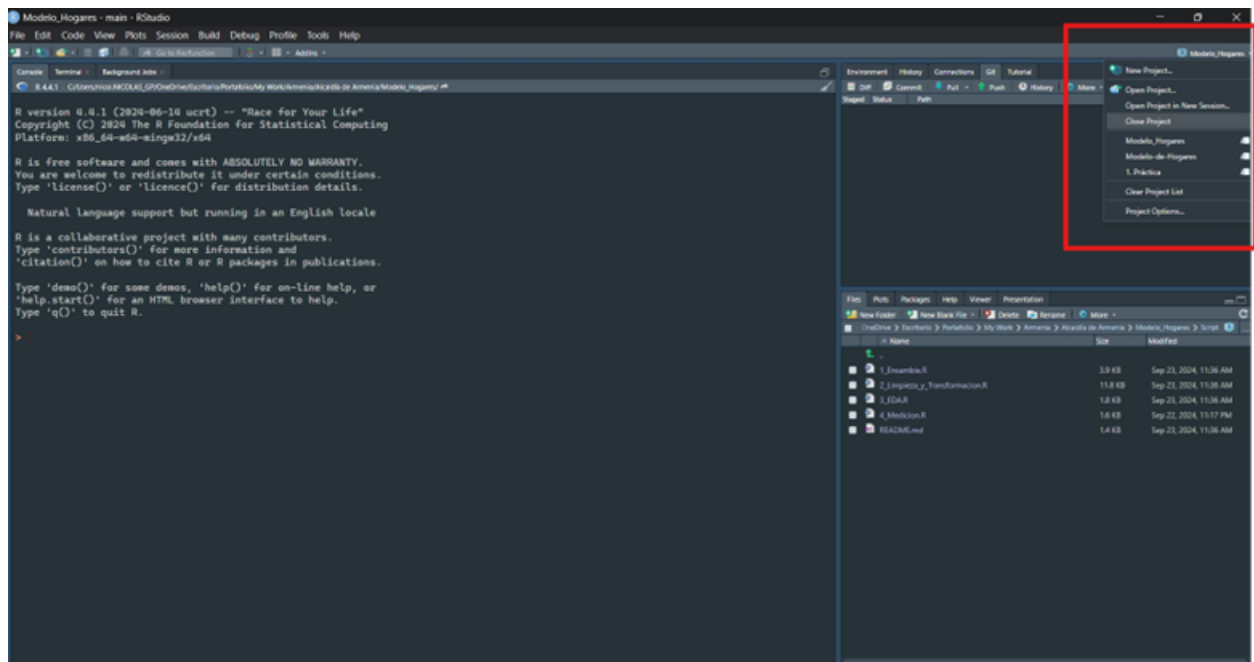








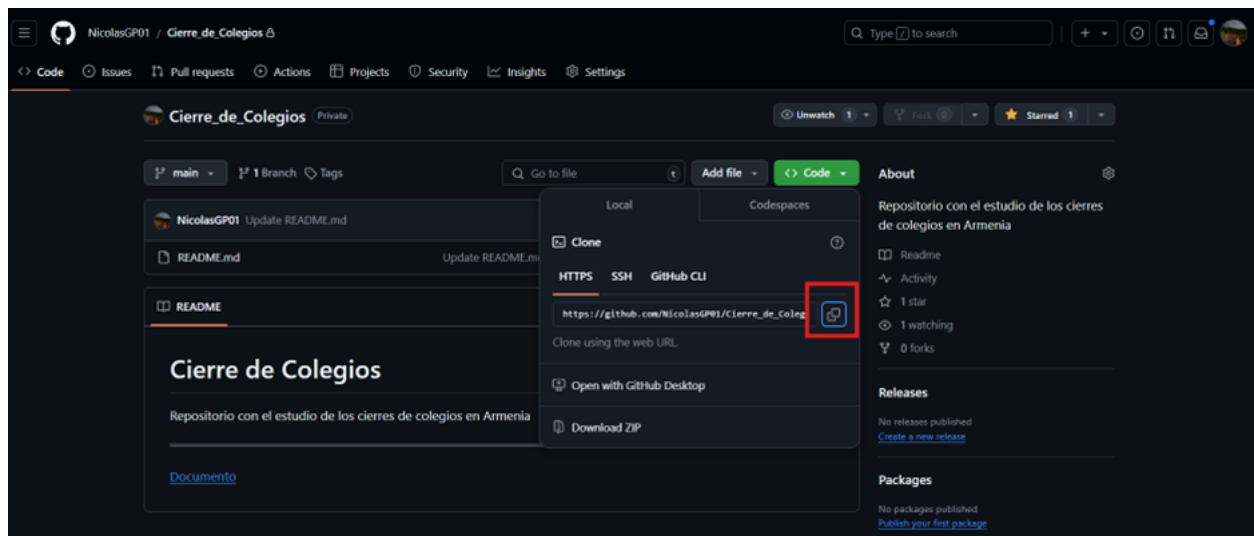


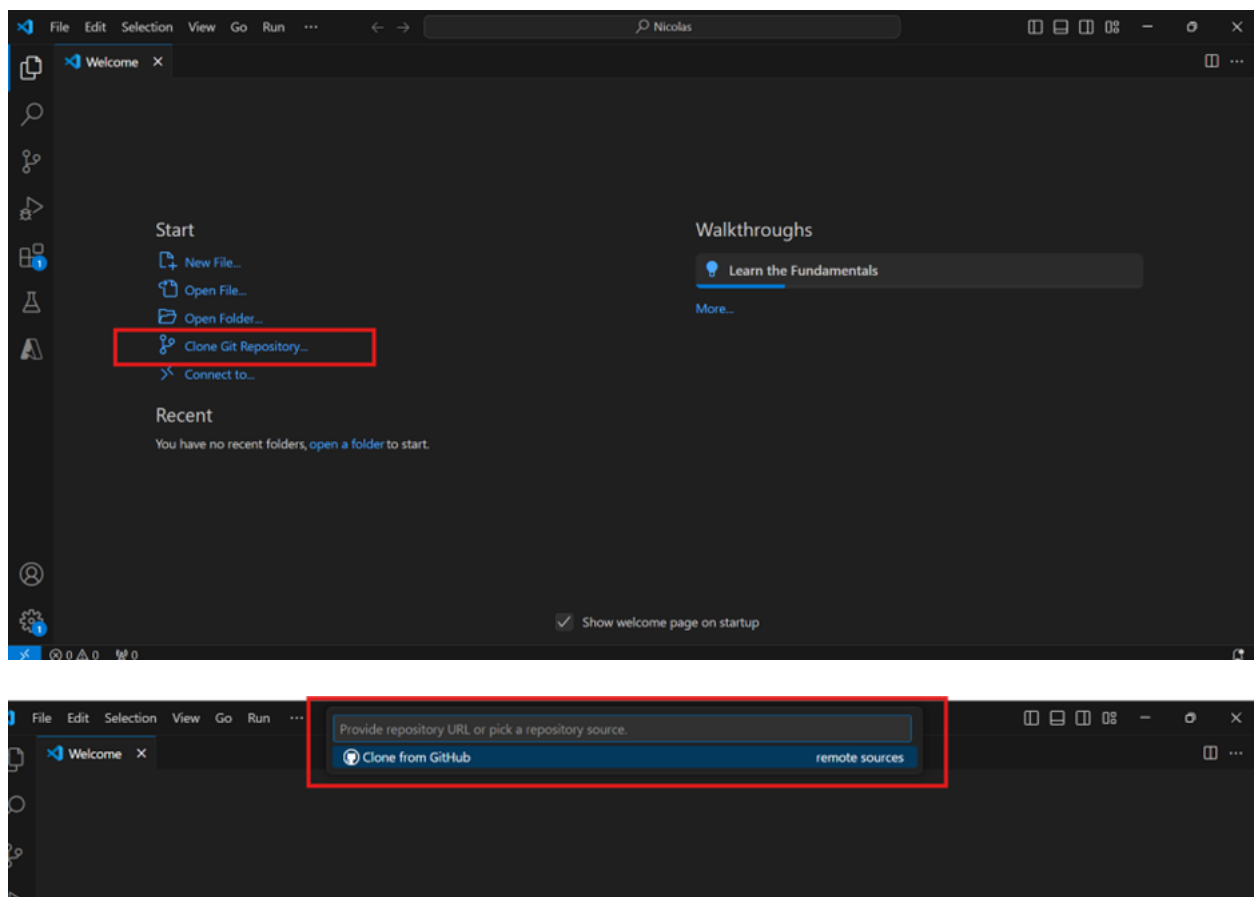


3. Python

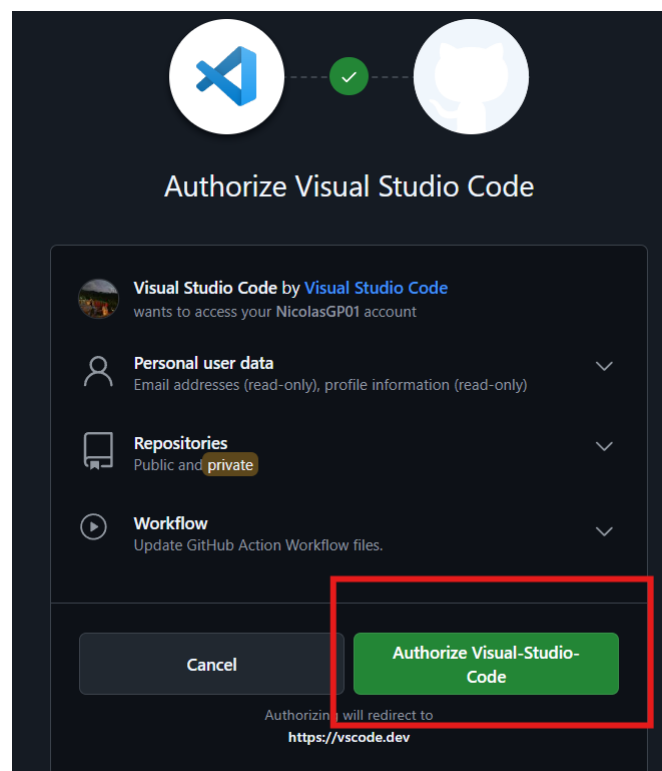
En este caso desarrollaremos nuestro código de **Python** en **Visual Studio Code**.

**Recuerde ya haber establecido el nombre y correo en GIT si apenas va empezar a utilizar este control de versiones con Python*

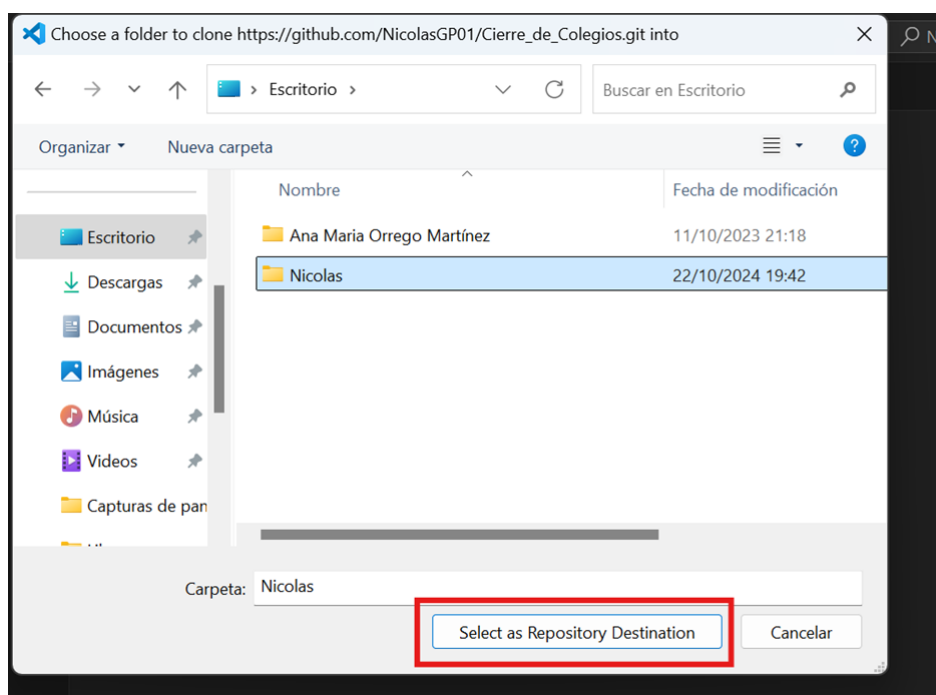
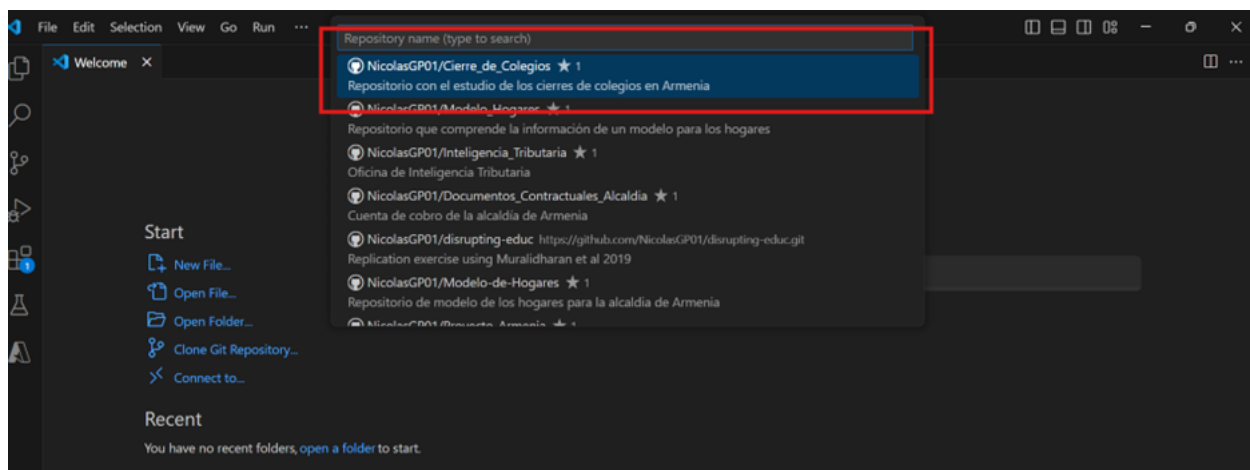




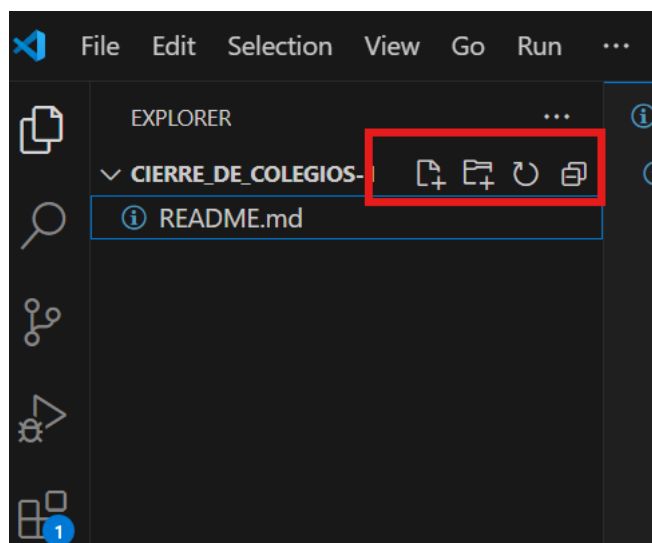
Le damos permitir

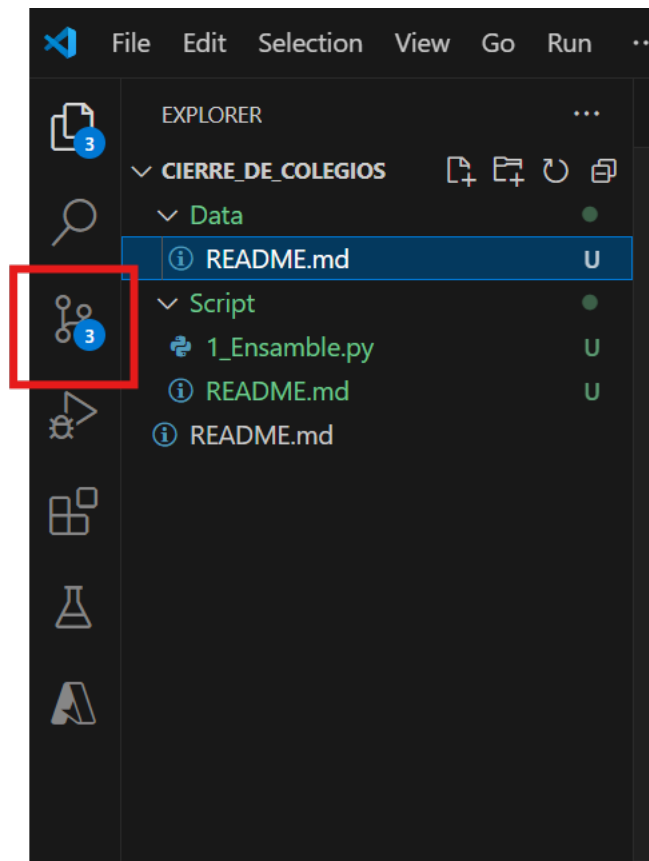


Elegimos el repositorio el cual vamos a trabajar

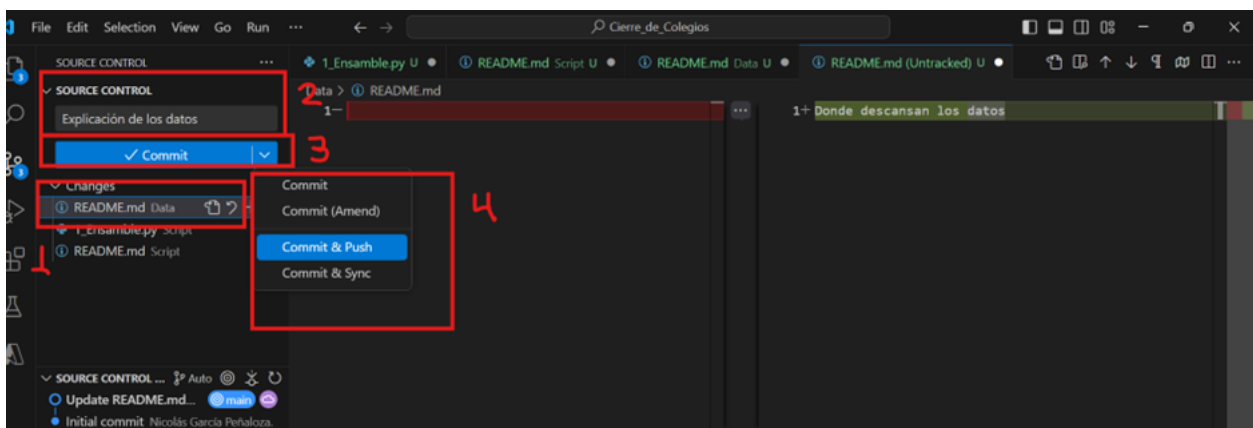
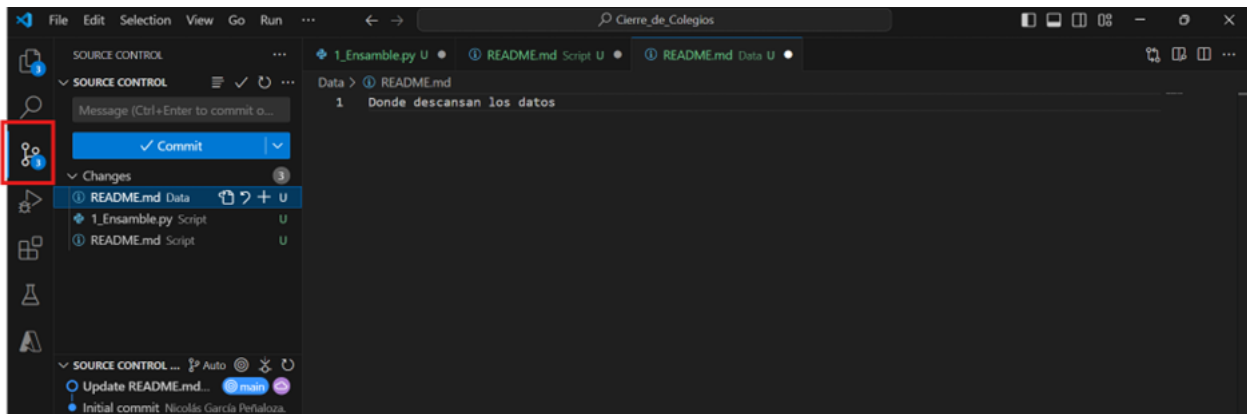


Le damos en verificar en la opción de su gusto Luego abrimos allí mismo el repositorio Podemos elegir abrir una carpeta o un script

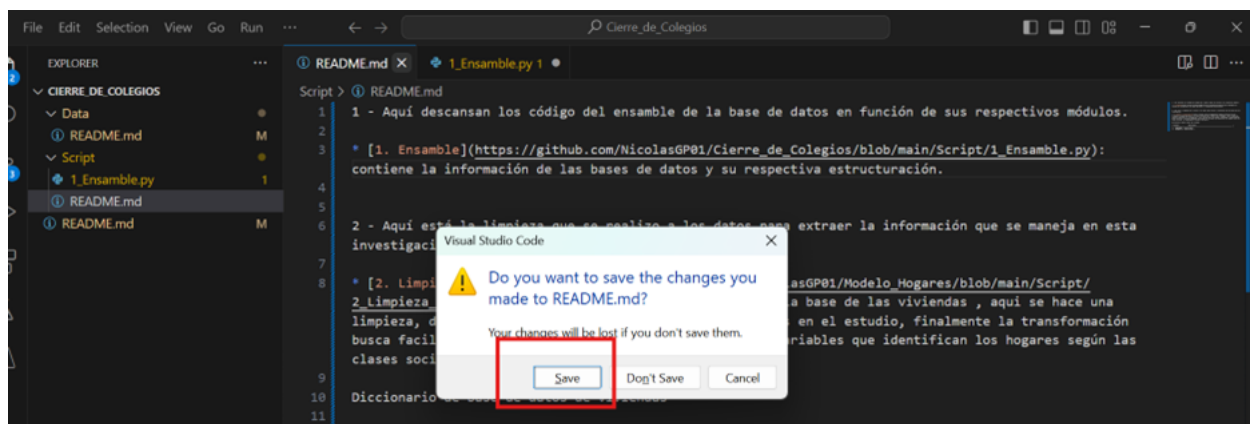
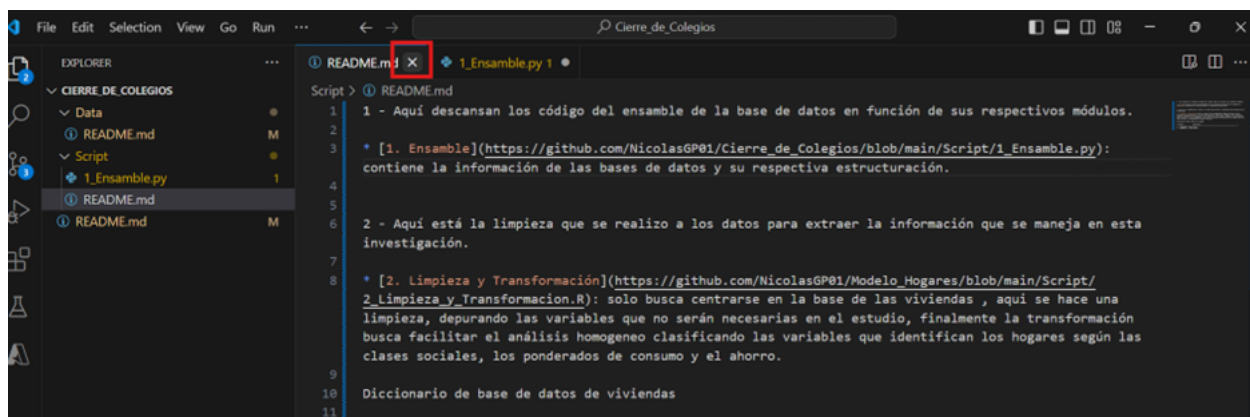




Enseguida de la U en verde del archivo debemos tener el símbolo + para que se puedan ejecutar los cambios



Una vez ya guardado y enviado para ejecutar los cambios debemos cerrar los archivos y guardarlos



Si despues vamos a abrir y clonar otro repositorio debemos escribir GitHub en la barra de busqueda.

Tengamos en cuenta que por el tiempo y las actualizaciones los pasos pueden cambiar.

