Realice un Informe paso a paso de la configuración

de GitHub para su proyecto de Fin de Ciclo

Asignatura:

OBJETOS Y ABSTRACCION DE DATOS

Carrera:

Sistemas de Información

Nivel:

Tercer nivel

Paralelo:

“B”

Periodo:

Octubre del 2023 hasta enero 2024

Estudiante:

Quimi Ramírez Iván Josué

Docente:

Ing. Vargas Nolivos Hernan Patricio

Portoviejo-Manabí- Ecuador

INDICE

1. INTRODUCCIÓN

La plataforma GitHub ha emergido como un punto central para el desarrollo colaborativo de proyectos, y la acción de subir un repositorio o proyecto en este entorno se presenta como un paso esencial, especialmente en el contexto del proyecto de medio ciclo. Este proceso no solo simplifica la gestión del código, sino que también facilita la colaboración entre los miembros del equipo, proporcionando una plataforma robusta para el control de versiones y la coordinación eficiente.

En esta guía detallada, se explorarán los pasos prácticos y esenciales necesarios para llevar a cabo la carga exitosa del proyecto en GitHub. Desde la creación de un nuevo repositorio hasta la sincronización eficiente de los cambios realizados, cada paso será abordado para garantizar una experiencia fluida y optimizada en el desarrollo del proyecto de medio ciclo.

Al seguir estos pasos, aquellos que participan en el proyecto no solo estarán compartiendo su trabajo de manera efectiva, sino que también aprovecharán las características poderosas de GitHub, tales como el seguimiento de problemas, solicitudes de extracción y colaboración sin problemas. La preparación será clave para maximizar el potencial del proyecto de medio ciclo mediante la implementación efectiva de estas prácticas recomendadas en la plataforma GitHub.

2. OBJETIVOS

2.1. OBJETIVO GENERAL:

Realizar la carga exitosa de un proyecto propio en GitHub mediante comandos Git, garantizando una efectiva administración del código y facilitando la colaboración con otros desarrolladores para fomentar la revisión y mejora conjunta.

2.2. OBJETIVOS ESPECÍFICOS:

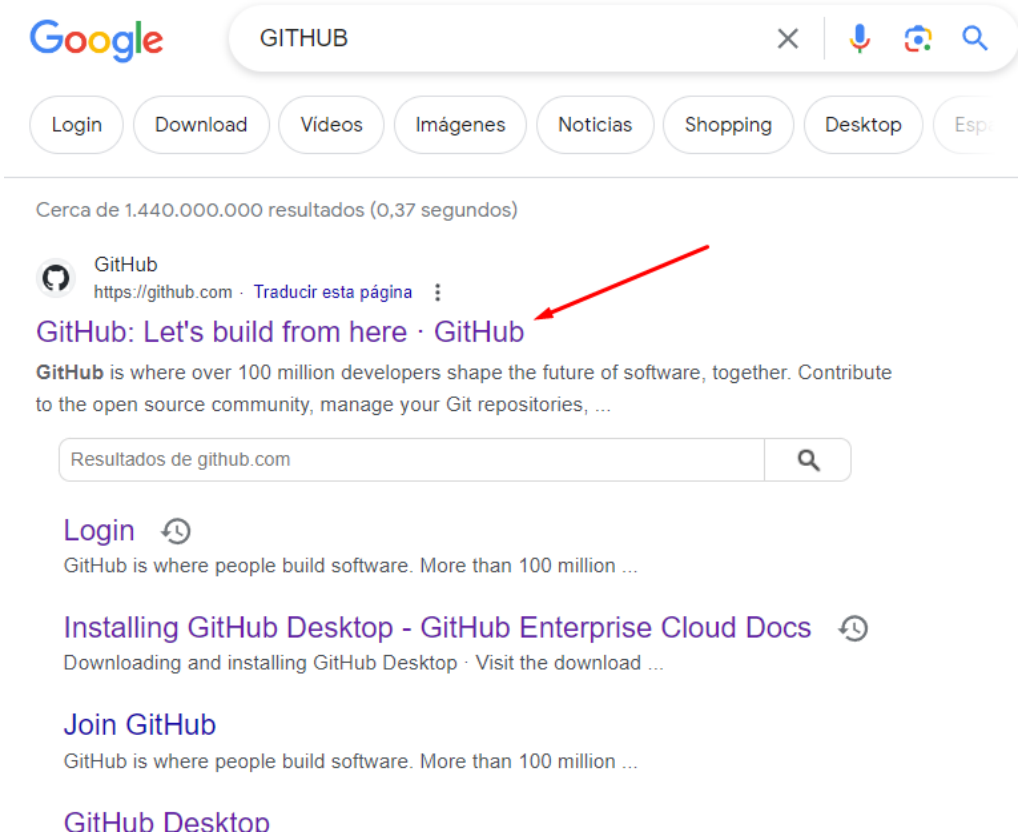
Crear un repositorio en GitHub que represente de manera precisa la estructura y contenido del proyecto desarrollado, brindando un entorno propicio para la colaboración.

Utilizar comandos Git para realizar operaciones fundamentales, como agregar, confirmar y enviar cambios al repositorio remoto, asegurando un seguimiento adecuado del historial de versiones y proporcionando transparencia en el proceso de desarrollo.

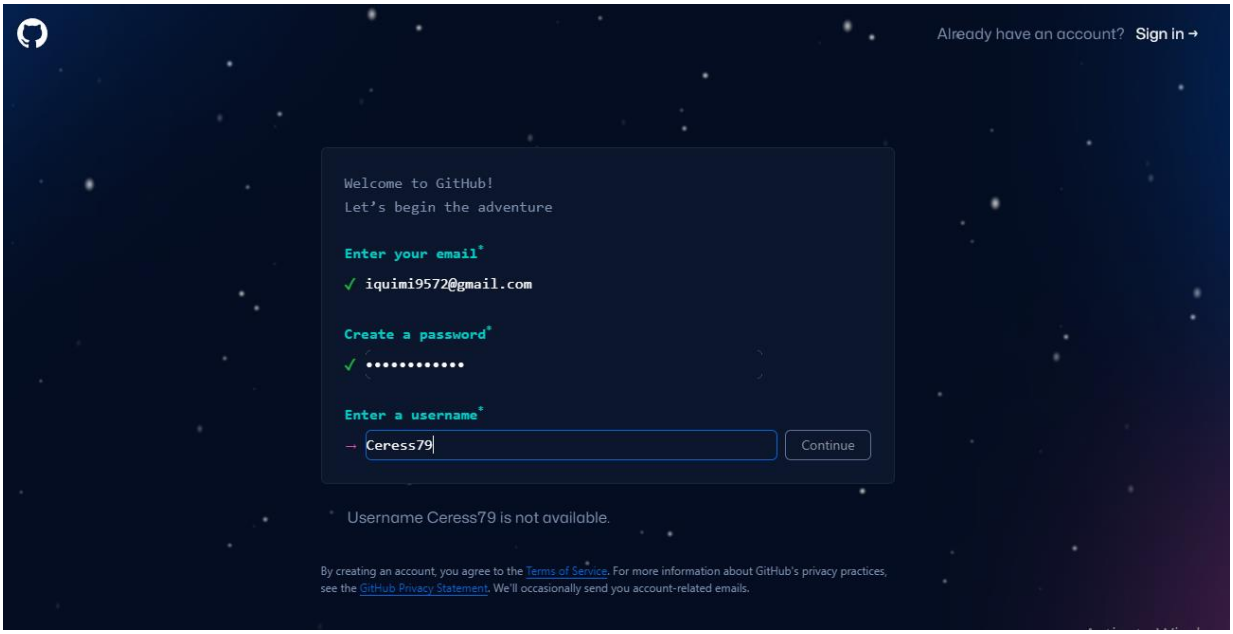
Implementar y gestionar ramas (branches) de forma estratégica para facilitar el desarrollo de nuevas características y asegurar una integración suave del código principal, permitiendo a otros programadores opinar y contribuir eficientemente al proyecto.

3. CREACION DE LA CUENTA EN GITHUB

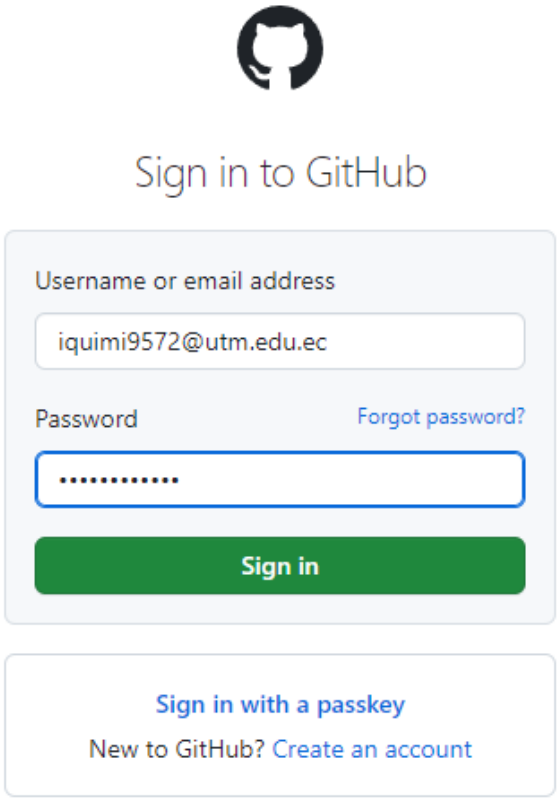
Se debe entrar al buscador de un navegador, y buscar “gituhub”, entrar al primer link que aparece.



Una vez dentro de la página principal, se le da al botón de crear cuenta, y se procede a completar los datos solicitados para crear esta misma, los datos que se ingresara son relacionado con la Universidad Técnica de Manabí, el correo institucional, el nombre de usuario personalizado:

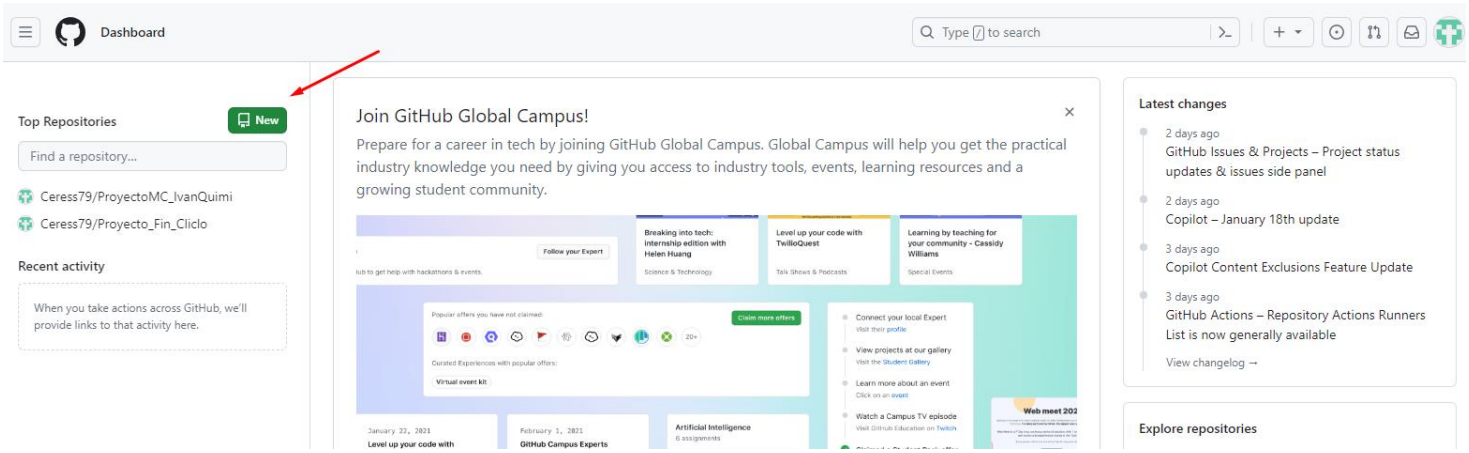


Una vez creada la cuenta, iniciamos sesión de no haberse iniciado automáticamente.

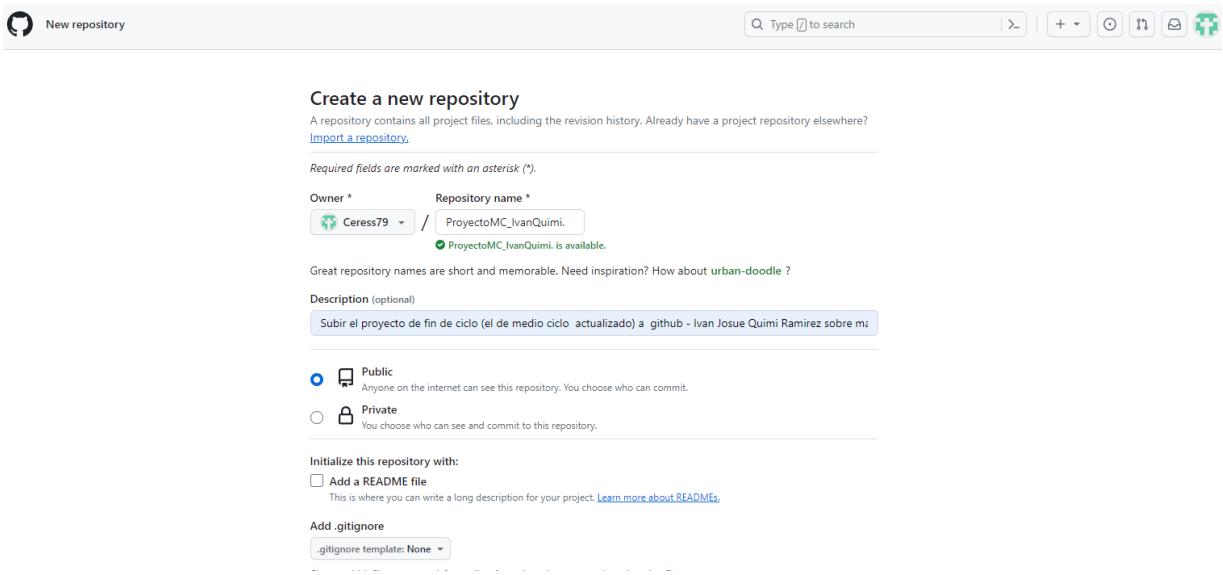


4. CREAR REPOSITORIO EN GITHUB

Posteriormente, entramos a la página principal. En esta se tiene muchas opciones, pero la que se interesa es el botón verde de la izquierda llamado “NEW”, se le da clic para crear un nuevo repositorio, como se ve en la imagen:

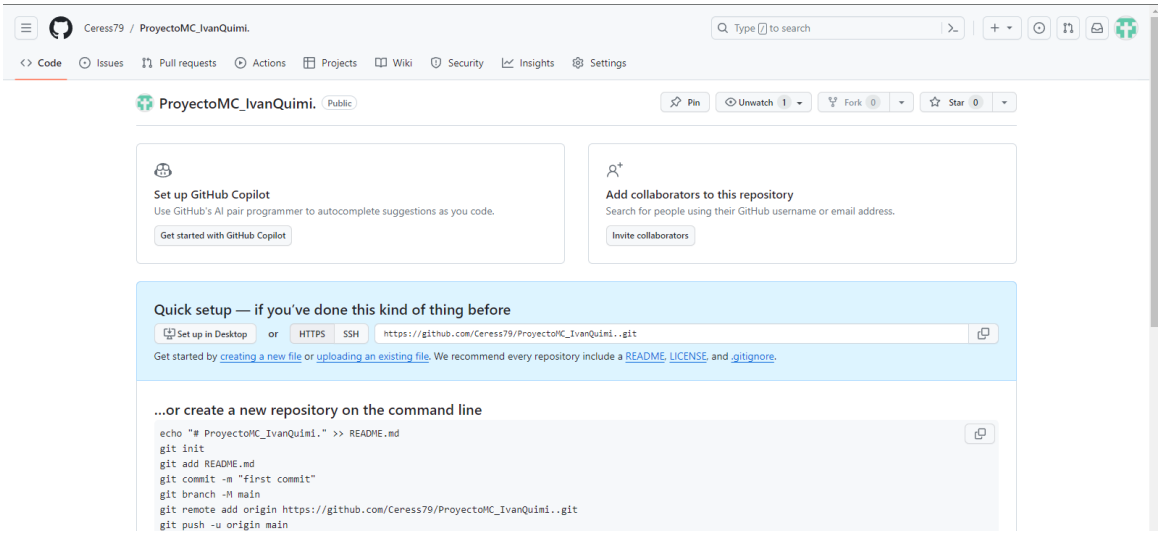


Una vez



seleccionado el botón "New", se abrirá una nueva página que permite la creación de un nuevo repositorio. Para este informe, el nombre asignado al repositorio es "ProyectoMC_IvanQuimi". Se proporcionó una descripción concisa para identificar el propósito del repositorio. Posteriormente, se procedió a seleccionar el botón de "Crear Repositorio", dando inicio al proceso de creación del mismo. Este procedimiento garantiza la organización y gestión eficiente del proyecto denominado "ProyectoMC_IvanQuimi" en el entorno de desarrollo correspondiente.

Tras completar el proceso de creación del repositorio, se visualiza una página que confirma el éxito de la operación. En esta pantalla, se puede apreciar la información detallada del repositorio recién creado, confirmando su nombre como "ProyectoMC_IvanQuimi" y mostrando la descripción asociada. Este paso culmina con éxito al haber creado el repositorio de manera satisfactoria, estableciendo así una base sólida para la gestión y colaboración en el desarrollo del proyecto mencionado.:

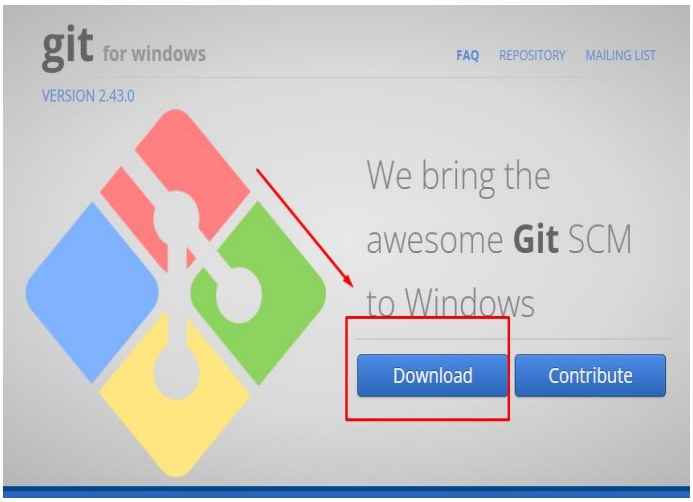


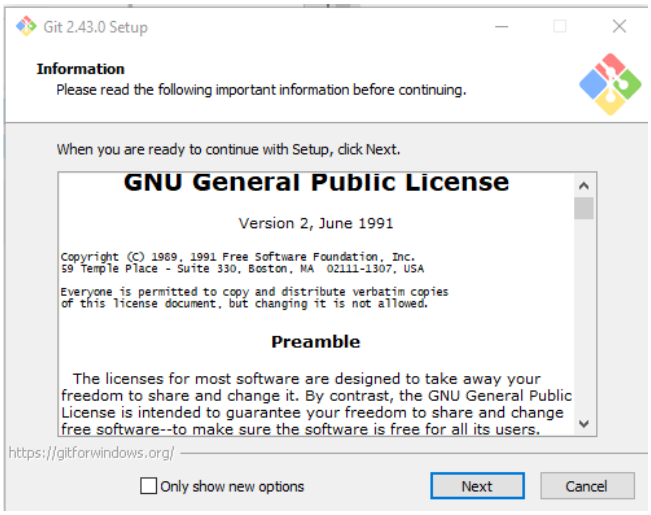
5. INSTALACIÓN DE GIT

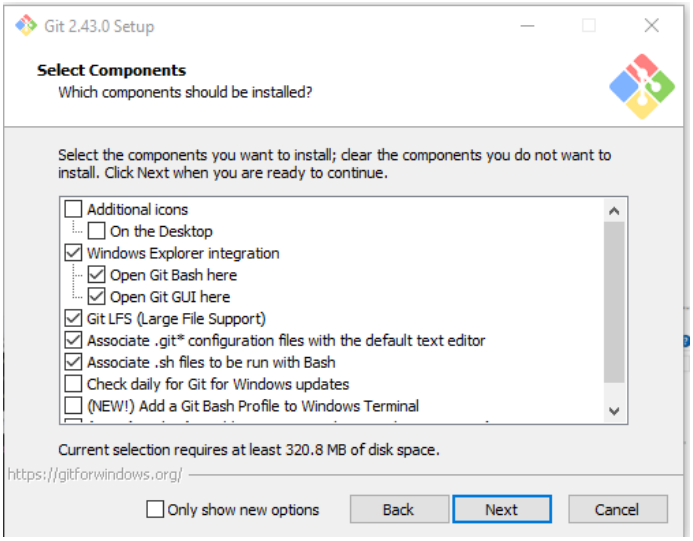
El siguiente paso consiste en la instalación de Git en el ordenador. Para llevar a cabo este proceso, se debe seguir los siguientes pasos:

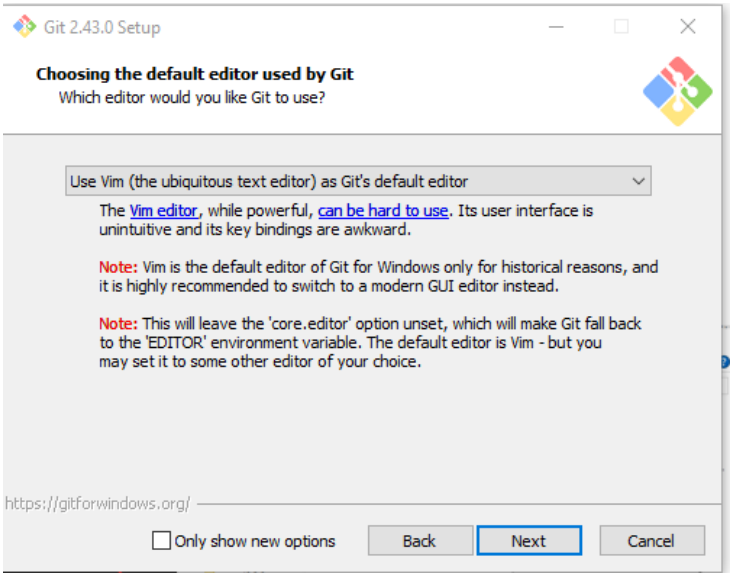
- Descargue la versión de Git para Windows desde https://gitforwindows.org/
- Ejecute el archivo de instalación como administrador.
- Siga las opciones predeterminadas durante la instalación, seleccionando "Use Git from the command line" y "Checkout Windows-style, commit Unix-style line endings."
- Finalice la instalación presionando "Siguiente" y luego "Finalizar."

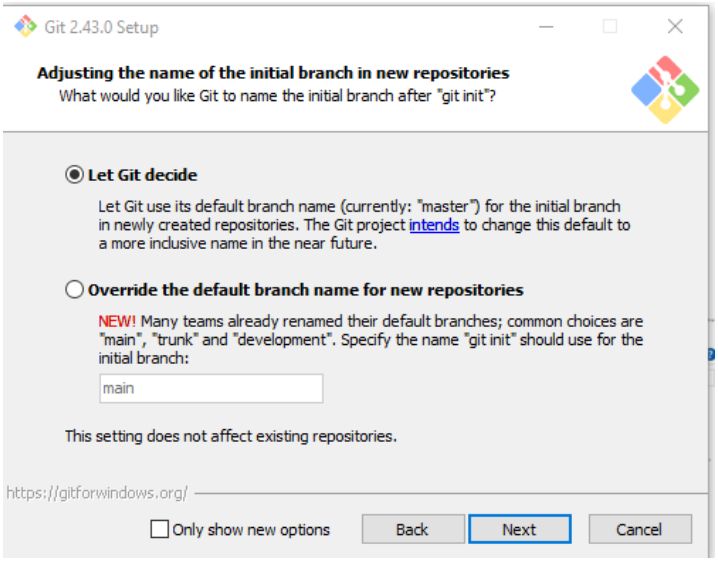
.

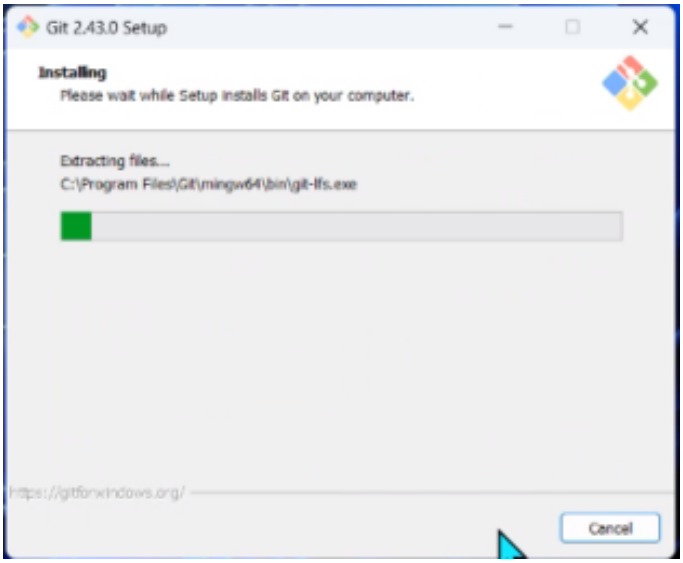

git for windows
FAQ
REPOSITORY
MAILING LIST
VERSION 2.43.0
We bring the awesome Git SCM to Windows
Download
Contribute


Git 2.43.0 Setup
Information
Please read the following important information before continuing.
When you are ready to continue with Setup, click Next.
GNU General Public License
Version 2, June 1991
Copyright (C) 1989, 1991 Free Software Foundation, Inc.
59 Temple Place - Suite 330, Boston, MA 02111-1307, USA
Everyone is permitted to copy and distribute verbatim copies
of this license document, but changing it is not allowed.
Preamble
The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change free software--to make sure the software is free for all its users.
https://gitforwindows.org/
Only show new options
Next
Cancel


Git 2.43.0 Setup
Select Components
Which components should be installed?
Select the components you want to install; clear the components you do not want to install. Click Next when you are ready to continue.
Additional icons
On the Desktop
Windows Explorer integration
Open Git Bash here
Open Git GUI here
Git LFS (Large File Support)
Associate .git* configuration files with the default text editor
Associate .sh files to be run with Bash
Check daily for Git for Windows updates
(NEW!) Add a Git Bash Profile to Windows Terminal
Current selection requires at least 320.8 MB of disk space.
https://gitforwindows.org/
Only show new options
Back
Next
Cancel


Git 2.43.0 Setup
Choosing the default editor used by Git
Which editor would you like Git to use?
Use Vim (the ubiquitous text editor) as Git's default editor
The Vim editor, while powerful, can be hard to use. Its user interface is unintuitive and its key bindings are awkward.
Note: Vim is the default editor of Git for Windows only for historical reasons, and it is highly recommended to switch to a modern GUI editor instead.
Note: This will leave the 'core.editor' option unset, which will make Git fall back to the 'EDITOR' environment variable. The default editor is Vim - but you may set it to some other editor of your choice.
https://gitforwindows.org/
Only show new options
Back
Next
Cancel


Git 2.43.0 Setup
Adjusting the name of the initial branch in new repositories
What would you like Git to name the initial branch after "git init"?
Let Git decide
Let Git use its default branch name (currently: "master") for the initial branch in newly created repositories. The Git project intends to change this default to a more inclusive name in the near future.
Override the default branch name for new repositories
NEW! Many teams already renamed their default branches; common choices are "main", "trunk" and "development". Specify the name "git init" should use for the initial branch:
main
This setting does not affect existing repositories.
https://gitforwindows.org/
Only show new options
Back
Next
Cancel


Git 2.43.0 Setup
Installing
Please wait while Setup installs Git on your computer.
Extracting files...
C:\Program Files\Git\mingw64\bin\git-lfs.exe
Cancel

6. Crear EL REPOSITORIO LOCAL

6.1.RUTA DEL ARCHIVO JAVA

La creación de un repositorio en Git representa el proceso inicial de configuración del entorno de trabajo destinado al seguimiento de versiones en un proyecto. Mediante la utilización de comandos específicos en la terminal de Git BASH, se procede a la instauración de un repositorio local. Para comenzar el proceso de creación de un repositorio en Git, el usuario debe primero dirigirse a la ruta donde se encuentra ubicado el proyecto en la terminal de Git BASH. Esta ubicación específica se refleja en la línea de comandos, como se ilustra en el siguiente ejemplo:

```
LA MAMALONA@DESKTOP-12FVJ5J MINGW64 ~/Documents/NetBeansProjects /proyectoMC_IvanQuimi
```

6.2.GIT INIT, GIT ADD, GIT COMMIT -M

Una vez que la terminal está ubicada en la ruta del proyecto, se pueden ejecutar los comandos de Git, como git init, git add ., y git commit, para establecer y gestionar el repositorio local en ese directorio específico. Estos comandos permitirán la captura y seguimiento de versiones en el proyecto, brindando así un control efectivo sobre el desarrollo del software en cuestión.

```
LA MAMALONA@DESKTOP-12FVJ5J MINGW64 ~/Documents/NetBeansProjects/proye
toMC_IvanQuimi (maing)
$ git init
Initialized empty Git repository in C:/Users/LA MAMALONA/Documents/Net
cts/proyecto/ProyectoMC_IvanQuimi/.git/
```

```
git init // Iniciar un nuevo repositorio Git local en el directorio actual
```

```
git add .// Agregar todos los archivos al área de preparación (staging)
```

```
LA MAMALONA@DESKTOP-12FVJ5J MINGW64 ~/Documents/NetBeansProjects/proyecto/Proyec
toMC_IvanQuimi (master)
$ Git add .
```

```
git commit -m // Confirmar los cambios en el repositorio local con un mensaje descriptivo
```

```
LA MAMALONA@DESKTOP-12FVJ5J MINGW64 ~/Documents/NetBeansProjects/proye
cto/ProyectoMC_IvanQuimi (master)
$ git commit -m "Primer Commit"
[master (root-commit) 3f1bc9b] Primer Commit
 9 files changed, 193 insertions(+)
 create mode 100644 pom.xml
 create mode 100644 src/main/java/com/mycompany/proyectomc_ivanquimi/M
etodos.java
 create mode 100644 src/main/java/com/mycompany/proyectomc_ivanquimi/P
royectoMC_IvanQuimi.java
 create mode 100644 target/classes/.netbeans_automatic_build
 create mode 100644 target/classes/com/mycompany/proyectomc_ivanquimi/
Metodos.class
 create mode 100644 target/classes/com/mycompany/proyectomc_ivanquimi/
ProyectoMC_IvanQuimi.class
 create mode 100644 target/maven-status/maven-compiler-plugin/compile/
default-compile/createdFiles.lst
 create mode 100644 target/maven-status/maven-compiler-plugin/compile/
default-compile/inputFiles.lst
 create mode 100644 target/test-classes/.netbeans_automatic_build
```

6.3. GIT CONFIG --GLOBAL

Además de la configuración del repositorio local, es esencial establecer la identidad del usuario para asociar correctamente los cambios en el historial del repositorio. Los comandos git config proporcionados configuran globalmente la dirección de correo electrónico y el nombre de usuario para todas las operaciones de Git en el sistema. A continuación, se explica cada línea de código:

```
git config --global user.email "iquimi9572@utm.edu.ec"
```

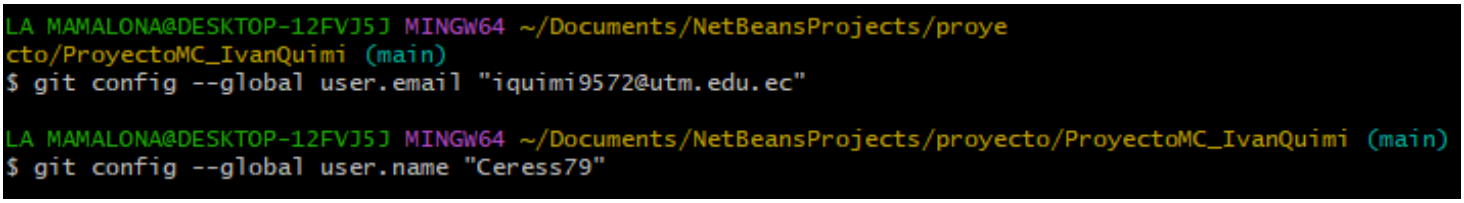
6.4.GIT CONFIG --GLOBAL

Este comando establece globalmente la dirección de correo electrónico asociada con las operaciones de Git. Es fundamental utilizar la dirección de correo electrónico asociada a la cuenta de Git para garantizar la correcta atribución de los cambios realizados en el repositorio.

```
git config --global user.name "Ceress79"
```

Similarmente, esta instrucción establece globalmente el nombre de usuario asociado con las operaciones de Git. El nombre de usuario se utiliza para identificar al autor de los commits en el historial del repositorio.

Ambas configuraciones son cruciales para asegurar que los commits reflejen adecuadamente la autoría de los cambios en el repositorio. Con estas configuraciones, se establece una conexión entre los cambios realizados en el repositorio y la identidad del usuario que los efectuó.



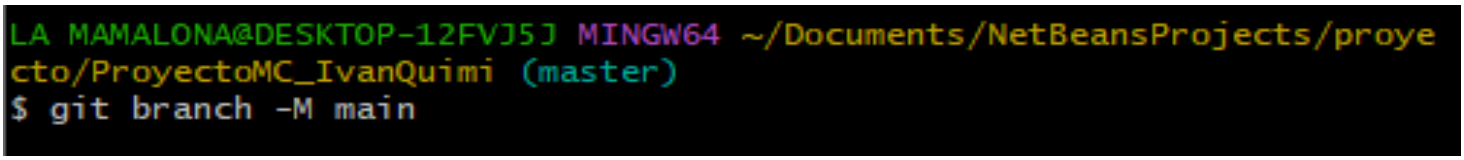
6.5.GIT BRANCH -M MAIN

Después de haber configurado la identidad del usuario, se procede a utilizar el siguiente comando para cambiar el nombre de la rama principal del repositorio a "main". Este paso es significativo para alinearse con las prácticas actuales y reflejar el uso de la rama principal como el punto central del desarrollo:

```
git branch -M main
```

En este comando:

- git branch se utiliza para manipular ramas en el repositorio.
- -M se emplea para renombrar la rama actualmente activa.
- main es el nuevo nombre asignado a la rama principal.



6.6.GIT REMOTE ADD ORIGIN

El siguiente paso es vincular el repositorio local con el repositorio remoto en GitHub. Esto se realiza mediante el comando git remote add origin, donde "origin" es un alias que representa la ubicación remota del repositorio. Aquí está el comando formalizado:

```
git remote add origin https://github.com/Ceress79/ProyectoMC_IvanQuimi.git
```

```
LA MAMALONA@DESKTOP-12FVJ5J MINGW64 ~/Documents/NetBeansProjects/proye
cto/ProyectoMC_IvanQuimi (main)
$ git remote add origin https://github.com/Ceress79/ProyectoMC_IvanQui
mi.git
```

En este comando:

git remote add origin: Establece un enlace entre el repositorio local y el repositorio remoto en GitHub, utilizando "origin" como nombre de referencia para la ubicación remota.

https://github.com/Ceress79/ProyectoMC_IvanQuimi.git: Es la URL del repositorio remoto. Esta dirección debe coincidir con la ubicación del repositorio en GitHub.

Con este comando, se configura la conexión remota, lo que permitirá la sincronización entre el repositorio local y el remoto

6.7.GIT PUSH -U ORIGIN MAIN

A continuación, se ejecuta el comando git push -u origin main. Este comando tiene como propósito principal enviar los cambios realizados en la rama principal del repositorio local al repositorio remoto en GitHub.

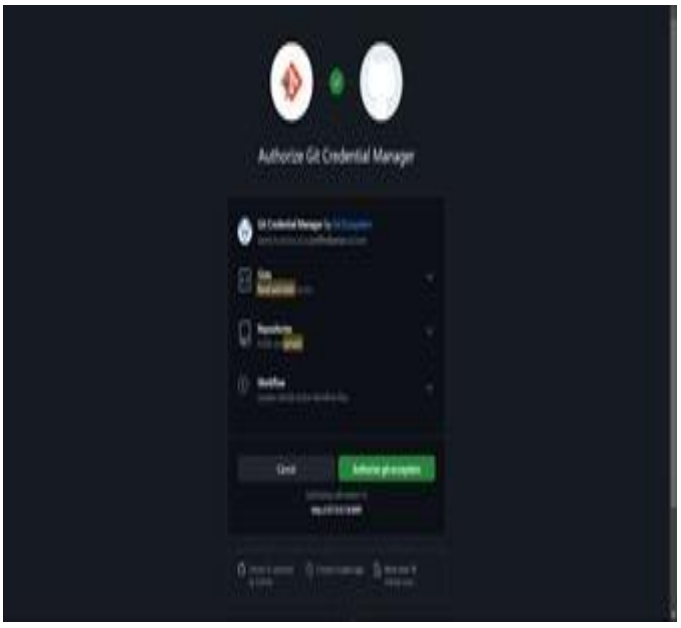
```
git push -u origin main
```

```
LA MAMALONA@DESKTOP-12FVJ5J MINGW64 ~/Documents/NetBeansProjects/proye
cto/ProyectoMC_IvanQuimi (main)
$ git push -u origin main
Enumerating objects: 26, done.
Counting objects: 100% (26/26), done.
Delta compression using up to 8 threads
Compressing objects: 100% (15/15), done.
Writing objects: 100% (26/26), 6.20 KiB | 907.00 KiB/s, done.
Total 26 (delta 1), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (1/1), done.
To https://github.com/Ceress79/ProyectoMC_IvanQuimi.git
 * [new branch]      main -> main
branch 'main' set up to track 'origin/main'.
```

- git push: Envía los cambios locales al repositorio remoto en GitHub.
- -u origin main: Establece la rama local "main" como la rama predeterminada para el repositorio remoto llamado "origin". Esto simplifica futuros comandos git push y git pull, ya que la opción -u configura la relación predeterminada entre la rama local y la remota, eliminando la necesidad de especificar la rama y la ubicación cada vez que se realice una operación.

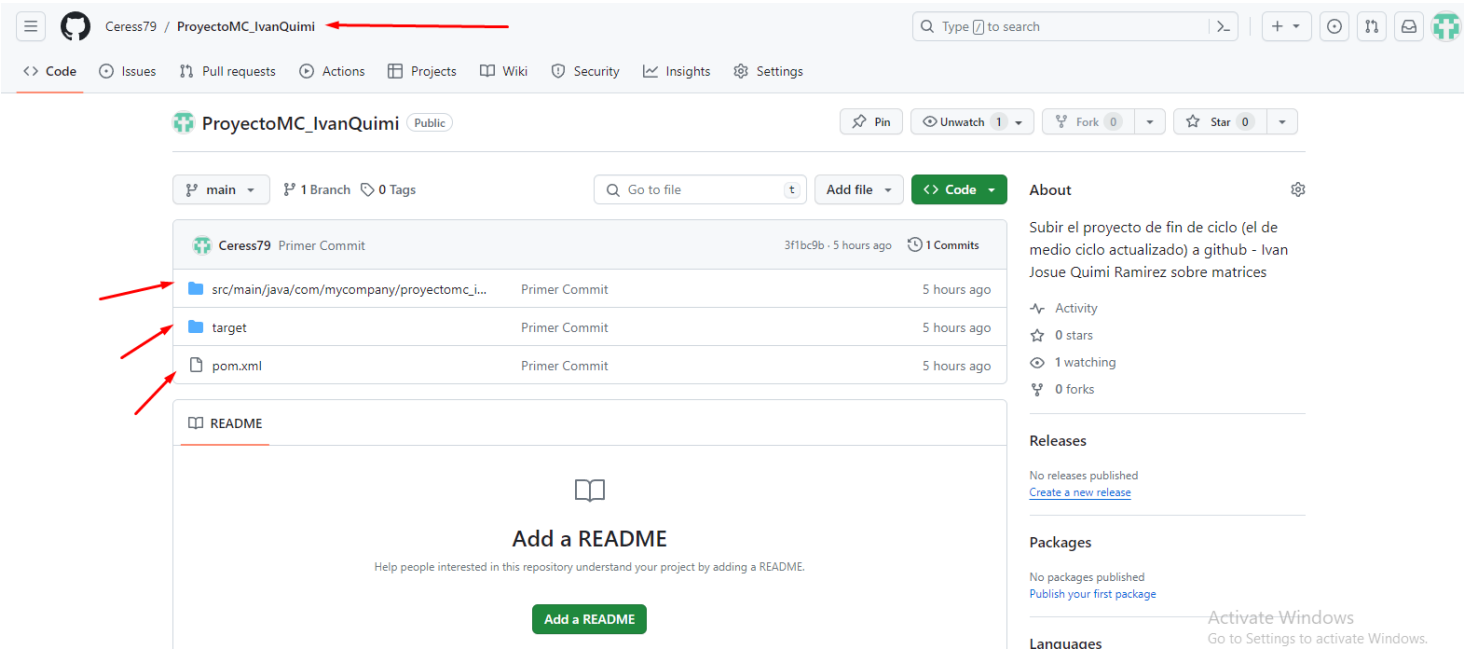
6.8.SINCRONIZAR Y AUTORIZAR CON LA CUENTA

Después de realizar el git push, se establece una conexión entre el repositorio local y el remoto en GitHub. Este proceso implica la transferencia de cambios y la actualización de la rama principal. La concesión de permisos y la sincronización de datos con la cuenta de GitHub confirman que los archivos se han cargado correctamente en el repositorio remoto.



7. VERIFICACION DE LOS ARCHIVOS SUBIDOS

Finalmente, para comprobar que los archivos se subieron exitosamente, se debe de ir a la página de GitHub y abrir el repositorio, donde se debe de visualizar todos los archivos del proyecto de medio ciclo subidos:



Link del repositorio de GITHUB:

https://github.com/Ceress79/ProyectoMC_IvanQuimi.git