

INFORME MICROPROYECTO GIT



Universidad
del Cauca

PRESENTADO POR:

Angee Daniela Quintero Gutierrez

Manuel Alejandro Macías

ASIGNATURA:

Laboratorio II de Sistemas Telemáticos

PRESENTADO A:

Fulvio Yesid Vivas

UNIVERSIDAD DEL CAUCA

Facultad de Ingeniería Electrónica y Telecomunicaciones

PROGRAMA: Tecnología en Telemática

Popayán

15/05/202

Índice:

1. INTRODUCCION
2. OBJETIVOS
3. MARCO TEORICO
4. PROCESOS Y DESARROLLO DEL PROYECTO
5. ANALISIS DE RESULTADOS
6. CONCLUSIONES
7. BIBLIOGRAFIAS

OBJETIVOS:

- Crear un repositorio público en GitHub llamado "TicTacToe" para alojar un proyecto de juego de "Tres en línea" (Tic-Tac-Toe) y utilizar Git para gestionar las versiones del proyecto.
- Realizar un commit inicial con el mensaje "Initial Commit" y hacer un push del repositorio local al remoto para establecer la base del proyecto.
- Agregar un archivo "README.md" que explique el propósito y funcionamiento del proyecto, incluyendo una imagen de ejemplo del juego.
- Adaptar el proyecto para que los mensajes del juego estén en español, garantizando una experiencia de usuario coherente y localizada.
- Realizar pruebas exhaustivas del juego para asegurarse de que funcione correctamente en todos los casos posibles, incluyendo el inicio del juego por parte del usuario, del computador, la victoria del usuario, la victoria del computador y el empate.
- Actualizar el repositorio local y remoto con los cambios realizados para reflejar la traducción del juego al español y la corrección de errores, utilizando el mensaje de commit "Actualización a idioma español".
- Modificar el código en el archivo "game.jsp" para ampliar el ancho del borde de la tabla (atributo "border" del objeto table) y para permitir jugar al "Cuatro en raya" (atributo "GRID_SIZE" de la clase GameBean).
- Actualizar el repositorio local y remoto con las modificaciones realizadas en el juego, utilizando el mensaje de commit "Modificación ancho de borde y juego Cuatro en raya".

Introducción:

La gestión eficiente del código fuente es esencial en el desarrollo de software, y una de las herramientas más populares para este propósito es Git. Git es un sistema de control de versiones ampliamente utilizado que permite a los desarrolladores rastrear, administrar y colaborar en proyectos de software de manera efectiva. En este informe, exploraremos la integración de Git con Eclipse IDE, una de los entornos de desarrollo integrado más utilizados en la industria del software.

En este informe, examinaremos cómo utilizar Git en combinación con Eclipse IDE, destacando las ventajas de esta integración y proporcionando ejemplos prácticos para ilustrar su uso. Además, discutiremos las mejores prácticas para trabajar con Git en Eclipse y cómo aprovechar al máximo esta poderosa combinación de herramientas. A medida que avanzamos en la era de la programación colaborativa y el desarrollo ágil de software, la integración de Git y Eclipse IDE se convierte en un recurso invaluable para los desarrolladores. Este informe proporcionará una visión general de esta integración y brindará información esencial para aquellos que deseen optimizar su flujo de trabajo y mejorar la colaboración en proyectos de desarrollo de software.

Marco teórico:

Git:

Git es un sistema de control de versiones distribuido ampliamente utilizado en el desarrollo de software. Proporciona una forma eficiente de rastrear cambios en el código fuente a lo largo del tiempo y facilita la colaboración entre equipos de desarrollo. En un contexto teórico, Git se basa en los siguientes conceptos clave:

1. **Repositorio:** Un repositorio Git es una base de datos que almacena todos los cambios y versiones del código fuente. Puede ser local o remoto, lo que permite a los desarrolladores trabajar de manera descentralizada y colaborativa.
2. **Commit:** Un commit es una instantánea de un conjunto de cambios en el código fuente. Cada commit se asocia con un mensaje descriptivo que explica los cambios realizados. Esto facilita la comprensión de la evolución del código.
3. **Rama (Branch):** En Git, se pueden crear ramas para trabajar en paralelo en diferentes características o problemas del proyecto. Las ramas permiten el desarrollo aislado y la posterior fusión de cambios.
4. **Fusión (Merge):** La fusión es el proceso de combinar los cambios de una rama en otra. Permite incorporar nuevas características o correcciones de errores en la rama principal o en otras ramas.
5. **Conflicto:** Los conflictos surgen cuando Git no puede fusionar automáticamente los cambios de dos ramas. Los desarrolladores deben resolver estos conflictos manualmente.
6. **Orígenes Remotos:** Los repositorios remotos son copias de un repositorio Git alojado en servidores. Los desarrolladores pueden colaborar, sincronizar y compartir cambios a través de repositorios remotos.

Eclipse IDE:

Eclipse Integrated Development Environment (IDE) es un entorno de desarrollo integrado de código abierto ampliamente utilizado. Ofrece un conjunto de herramientas y características que simplifican el proceso de desarrollo de software. En un marco teórico, Eclipse IDE se basa en los siguientes conceptos clave:

1. **Entorno de Desarrollo:** Eclipse proporciona una interfaz unificada que incluye un editor de código, depurador, gestión de proyectos y herramientas de desarrollo específicas del lenguaje.
2. **Perspectivas (Perspectives):** Eclipse organiza las vistas y las herramientas en perspectivas específicas según el tipo de desarrollo. Cada perspectiva se adapta a un flujo de trabajo particular, como desarrollo de Java, desarrollo web, o modelado.
3. **Vistas (Views):** Las vistas en Eclipse son ventanas que muestran información o proporcionan acceso a herramientas específicas. Las vistas pueden personalizarse según las necesidades del desarrollador.
4. **Editor de Código:** Eclipse ofrece un potente editor de código con resaltado de sintaxis, auto-completado y herramientas de refactorización que facilitan la escritura y edición de código.
5. **Depuración (Debugging):** Eclipse incluye herramientas de depuración que permiten a los desarrolladores rastrear y solucionar problemas en sus aplicaciones paso a paso.
6. **Extensiones (Plugins):** Eclipse es altamente extensible a través de plugins, lo que significa que los desarrolladores pueden personalizar su entorno de desarrollo agregando funcionalidades específicas.

Ubuntu:

Ubuntu puede ser utilizado como un ejemplo de un sistema operativo de código abierto que se basa en la filosofía de la colaboración y el compartir. Este enfoque contrasta con el de los sistemas operativos de código cerrado, como Windows y macOS, que son desarrollados y mantenidos por empresas privadas.

Ubuntu también puede ser utilizado como un ejemplo de cómo la tecnología puede ser utilizada para promover la inclusión y la accesibilidad. Al ser un sistema operativo libre y gratuito, Ubuntu está disponible para todos los usuarios, independientemente de su situación económica. Además, su diseño intuitivo y su fácil manejo lo hacen accesible para usuarios de todos los niveles de experiencia.

A continuación, se presentan algunos de los principios de Ubuntu que pueden ser relevantes para un marco teórico de un informe:

1. **Comunidad:** Ubuntu es un proyecto comunitario en el que participan personas de todo el mundo. Este enfoque colaborativo permite que el sistema operativo sea desarrollado y mejorado de manera rápida y eficiente.
2. **Autonomía:** Ubuntu es un sistema operativo libre y gratuito, lo que significa que los usuarios tienen el control sobre su propio software. Esto permite que los usuarios personalicen y adapten Ubuntu a sus necesidades específicas.
3. **Inclusión:** Ubuntu está diseñado para ser accesible para todos los usuarios, independientemente de su nivel de experiencia o situación económica.

PROCESOS Y DESARROLLO DEL PROYECTO:

1. Comenzamos creando una maquina virtual en Virtual Box junto a la imagen ISO del Ubuntu.

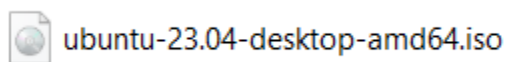


Imagen 1.

2. Después de hacer la instalación del Ubuntu comenzamos la instalación del Git desde la terminal con el comando “**sudo apt install git**”, que puede ser observado en la siguiente imagen, junto a su resultado:

```
manuel@ubuntu:~$ sudo apt install git
[sudo] password for arjun:
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  git-man liberror-perl
Suggested packages:
  git-daemon-run | git-daemon-sysvinit git-doc git-el git-email git-gui gitk
  gitweb git-cvs git-mediawiki git-svn
The following NEW packages will be installed:
  git git-man liberror-perl
0 upgraded, 3 newly installed, 0 to remove and 155 not upgraded.
Need to get 5,468 kB of archives.
After this operation, 38.4 MB of additional disk space will be used.
Do you want to continue? [Y/n]
```

Imagen 2.

- Procedemos a ingresar la letra “Y “para iniciar la instalación y después de haber terminado, confirmamos con el comando “git - -version” que se halla instalado correctamente y con que versión contamos.

```
Setting up git-man (1:2.25.1-1ubuntu3.1) ...
Setting up git (1:2.25.1-1ubuntu3.1) ...
Processing triggers for man-db (2.9.1-1) ...
manuel@ubuntu:~$git --version
git version 2.25.1
manuel@ubuntu:~$
```

Imagen 3.

3. Se crea una cuenta GITHUB para la creación de los repositorios, donde se va a subir de forma remota los archivos que se creen desde el Ubuntu.

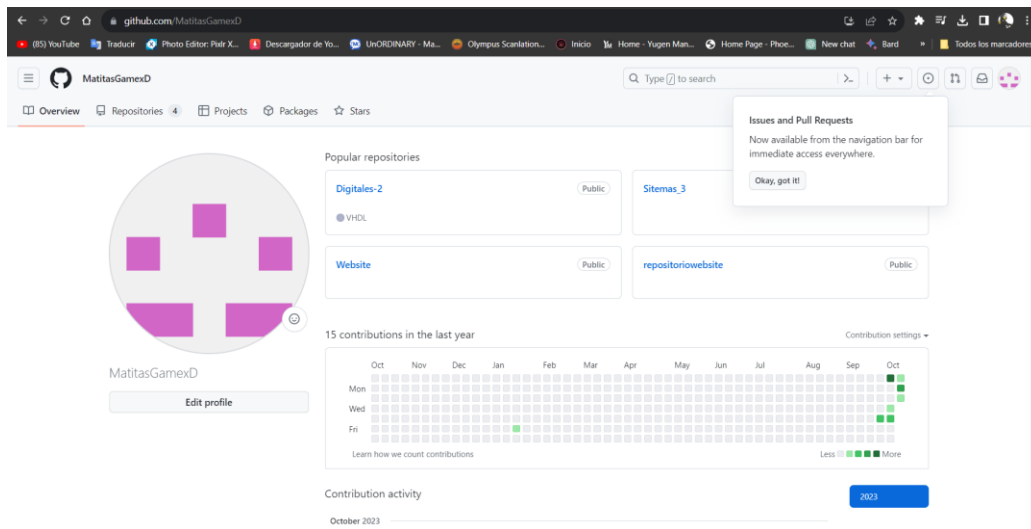


Imagen 4.

- Se procede con la creación de un repositorio llamado “repositoriowebiste “.

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 ().*

Owner * MatitasGameXD / Repository name *

Great repository names are short and memorable. Need inspiration? How about [laughing-octo-bassoon](#) ?

Description (optional)

repositoriowebiste Public
Updated 3 hours ago

☒ **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

- Después comenzamos con la creación de un archivo tipo “txt” para subir a nuestro GitHub de forma remota desde nuestro Ubuntu juntos a sus COMMIT correspondientes.

```
Initialized empty Git repository in /home/manuel/Desktop/Triki/.git/
manuel@ubuntu:~/Desktop/Triki$ git status
On branch master

No commits yet

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        hello.txt

nothing added to commit but untracked files present (use "git add" to track)
manuel@ubuntu:~/Desktop/Triki$ git add hello.txt
manuel@ubuntu:~/Desktop/Triki$ git add hello.txt
manuel@ubuntu:~/Desktop/Triki$ git status
On branch master

No commits yet

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

Imagen 5.

```
manuel@ubuntu:~/Desktop/Triki$ git config --global user.name
manuel
manuel@ubuntu:~/Desktop/Triki$ git config --global user.email
manucomacias23@gmail.com
manuel@ubuntu:~/Desktop/Triki$ git commit -m "Primer commit"
[master (root-commit) 3a387a5] Primer commit
 1 file changed, 1 insertion(+)
 create mode 100644 hello.txt
manuel@ubuntu:~/Desktop/Triki$
```

Imagen 6.

- Se procedió con la configuración remota, para establecer una conexión con el Git instalado desde el Ubuntu y el GIT HUB , para la subida del archivo , esto se logra con el comando “git remote add origin “junto a la dirección que establezca la pagina web de GIT HUB , que en este caso seria “git remote add origin <https://github.com/MatitasGamexD/TRiki.git> “

```
manuel@ubuntu:~/Desktop/Triki$ git remote add origin https://github.com/MatitasGamexD/TRiki.git
```

- Ya habiendo establecido la conexión correctamente se procede a la subida de los archivos con el comando “git push -u origin main” o “git push -u origin master”

6. Para lograr una conexión exitosa con el GIT HUB se debe de generar un **TOKEN** el cual genera un tipo de contraseña la cual nos permitirá establecer la conexión para así subir los archivos que deseemos a nuestro repositorio en GIT HUB.

NOTA: si no se hace este proceso no se logrará la conexión deseada y no se podrá realizar el trabajo deseado.

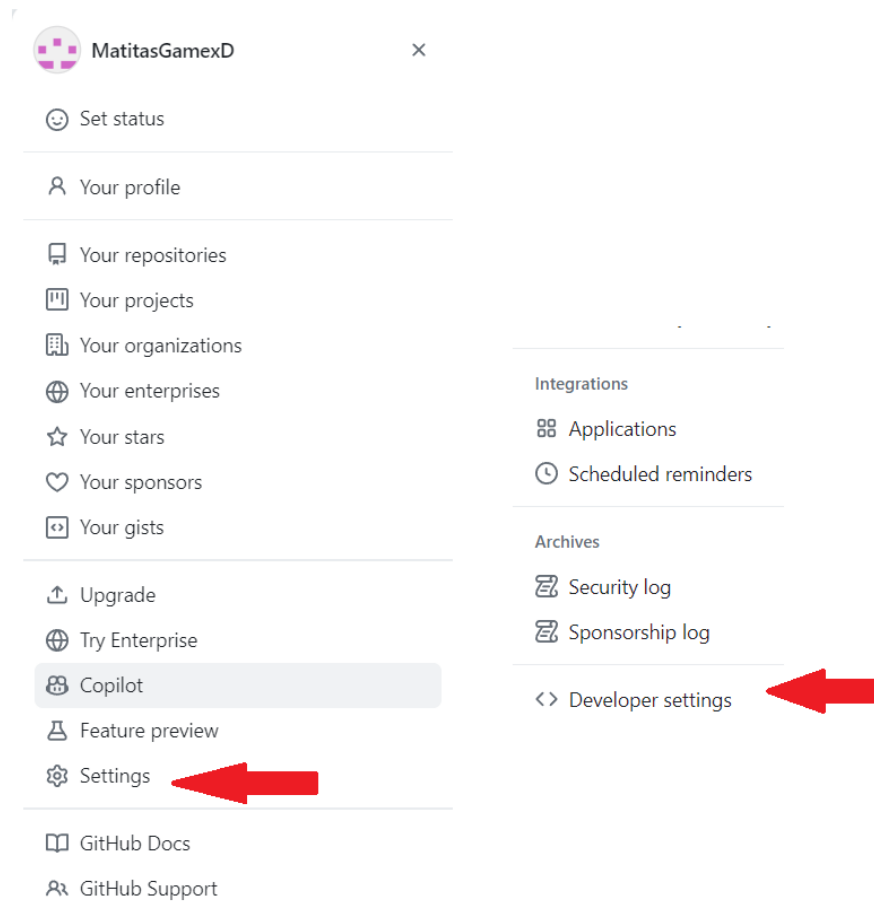
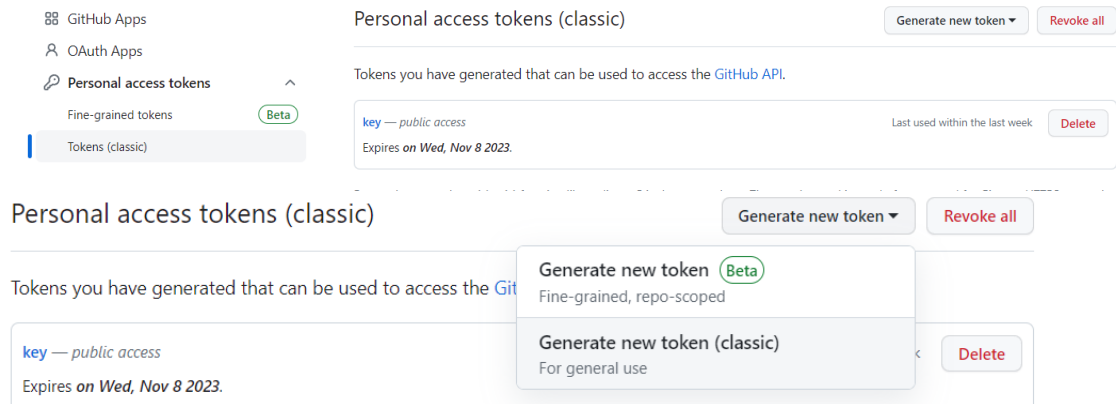


Imagen 7.

- Como se observa en la imagen 7. Procedemos a ir a la opción Settings , ya habiendo seleccionado la opción bajamos en la pagina hasta encontrar la opción “Developer settings”
- Ya dentro de la opción “Developer settings” elegimos la opción de crear un TOKEN (classic) y procedemos a generarlo como se muestra a continuación:



7. Ya habiendo realizado la creación del TOKEN procedemos con la subida del archivo a nuestro repositorio.

Nota: Se ingresa con el usuario de la cuenta de nuestro Git Hub y con el TOKEN que nos halla dado el Git Hub , el cual equivale a la contraseña pedida en este proceso.

```
manuel@ubuntu:~/Desktop/Triki$ git push -u origin master
Username for 'https://github.com': MatitasGamexD
Password for 'https://MatitasGamexD@github.com':
remote: Support for password authentication was removed on August 13, 2021.
remote: Please see https://docs.github.com/en/get-started/getting-started-with-g
it/about-remote-repositories#cloning-with-https-urls for information on currentl
y recommended modes of authentication.
fatal: Authentication failed for 'https://github.com/MatitasGamexD/TRiki.git/'
```

Imagen 8.

- Como se puede apreciar en la Figura 8. No se puede hacer una conexión remota desde el Git Ubuntu por problemas de conexión con el servidor. Este error persistió después de varios intentos y diferentes formas de conexión.

```
manuel@ubuntu:~/Desktop/Triki$ git push -u origin main
Username for 'https://github.com': MatitasGamexD
Password for 'https://MatitasGamexD@github.com':
remote: Support for password authentication was removed on August 13, 2021.
remote: Please see https://docs.github.com/en/get-started/getting-started-with-g
it/about-remote-repositories#cloning-with-https-urls for information on currentl
y recommended modes of authentication.
fatal: Authentication failed for 'https://github.com/MatitasGamexD/TRiki.git/'
manuel@ubuntu:~/Desktop/Triki$ git remote add origin https://github.com/MatitasG
amexD/TRiki.git
error: remote origin already exists.
manuel@ubuntu:~/Desktop/Triki$ https://github.com/MatitasGamexD/TRiki.git
bash: https://github.com/MatitasGamexD/TRiki.git: No such file or directory
manuel@ubuntu:~/Desktop/Triki$ git branch -M main
manuel@ubuntu:~/Desktop/Triki$ git branch -M main
manuel@ubuntu:~/Desktop/Triki$ git remote add origin https://github.com/MatitasG
amexD/TRiki.git
error: remote origin already exists.
manuel@ubuntu:~/Desktop/Triki$ git push -u origin main
Username for 'https://github.com': MatitasGamexD
Password for 'https://MatitasGamexD@github.com':
remote: Support for password authentication was removed on August 13, 2021.
remote: Please see https://docs.github.com/en/get-started/getting-started-with-g
it/about-remote-repositories#cloning-with-https-urls for information on currentl
y recommended modes of authentication.
fatal: Authentication failed for 'https://github.com/MatitasGamexD/TRiki.git/'
manuel@ubuntu:~/Desktop/Triki$ ubuntu
ubuntu: command not found
manuel@ubuntu:~/Desktop/Triki$
```

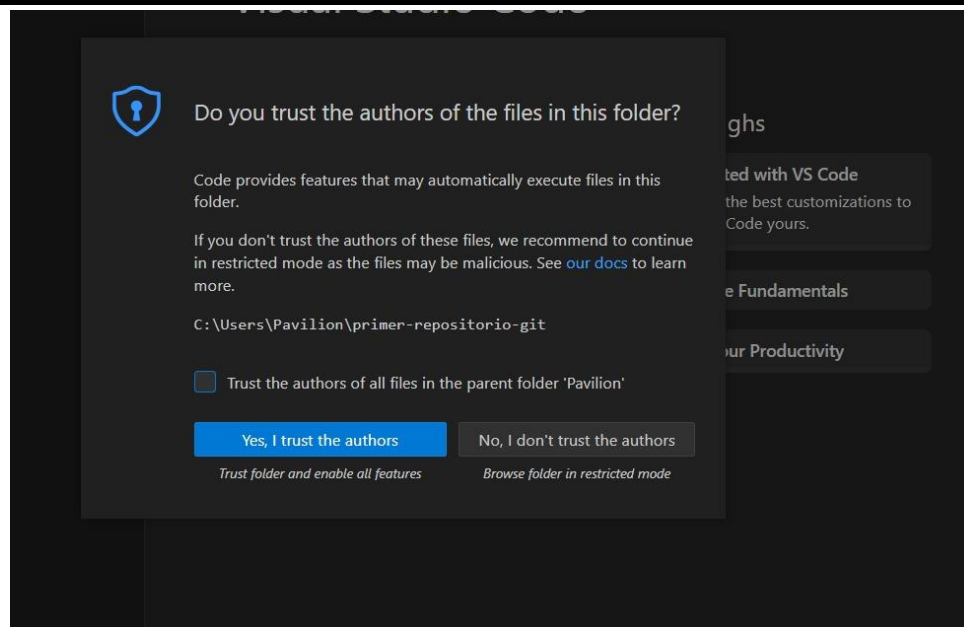
Imagen 9.

8. Por lo anterior se decidió elaborar el proyecto desde un Git diferente el cual no está relacionado con el servidor UBUNTU, Se realiza lo mismo que en anterior Git, como se demostrara a continuación :

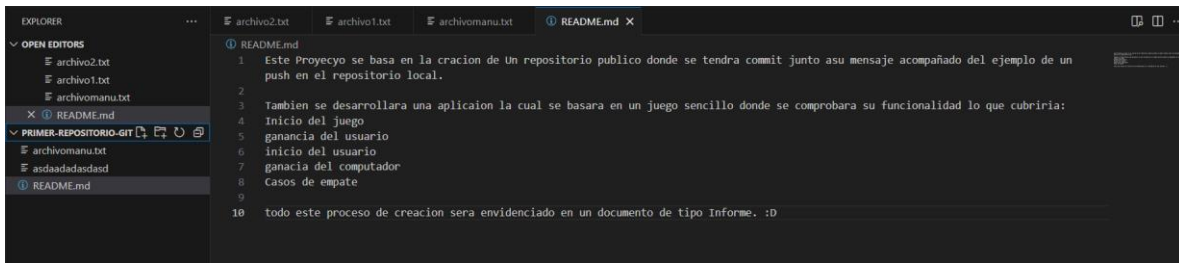
```
Pavilion@DESKTOP-DM01USE MINGW64 ~  
$ git config --global user.name "Manuel Macias"  
Pavilion@DESKTOP-DM01USE MINGW64 ~  
$ git config --global user.email manucomacias23@gmail.com  
Pavilion@DESKTOP-DM01USE MINGW64 ~  
$ git config --global core.editor "code --wait"  
  
Pavilion@DESKTOP-DM01USE MINGW64 ~  
$ pwd  
/c/Users/Pavilion  
Pavilion@DESKTOP-DM01USE MINGW64 ~  
$ mkdir primer-repositorio-git  
Pavilion@DESKTOP-DM01USE MINGW64 ~  
$ cd primer-repositorio-git  
Pavilion@DESKTOP-DM01USE MINGW64 ~/primer-repositorio-git  
$ pwd  
/c/Users/Pavilion/primer-repositorio-git
```

- Se hace la configuración del GIT y la creación de un repositorio para la creación y manejo de archivos junto con el uso de un editor de textos, que en este caso será Visual Studio Code.

```
Pavilion@DESKTOP-DM01USE MINGW64 ~/primer-repositorio-git (main)  
$ ls -a  
./ ../ .git/  
Pavilion@DESKTOP-DM01USE MINGW64 ~/primer-repositorio-git (main)  
$ code .
```



9. Procedemos a la creación del archivo README.md donde se pondrá la explicación breve del proyecto a realizar.



10. Procedemos con la configuración para agregar el archivo README.md y cualquier cambio que haya hecho:

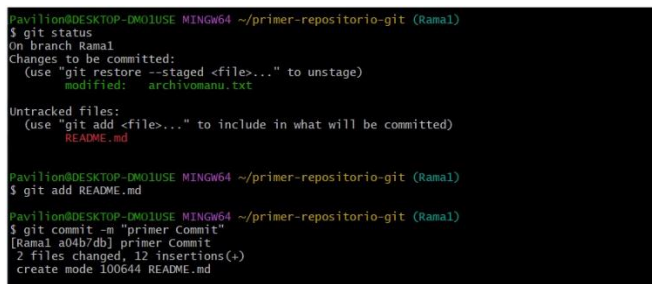


Imagen 10.

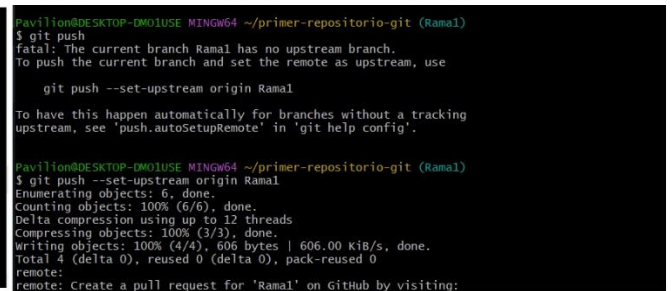


Imagen 11.

- Dentro de la configuración se realizó un “commit” junto a sus mensaje de “Primer Commit” para ser mostrado en el repositorio de Git HUB junto a las actualizaciones de los cambios hechos , esto gracias al comando “git push” o “git push - -set -upstream origin”

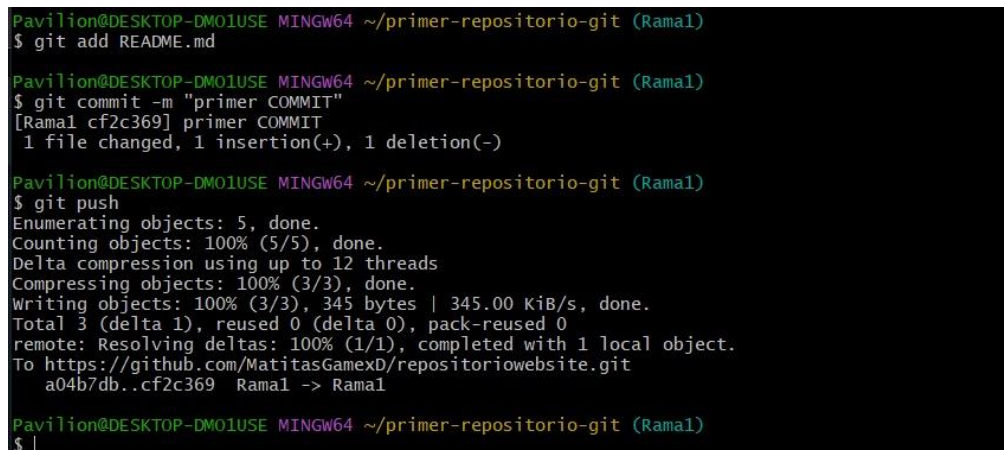


Imagen 12.

11. Por ultimo accedemos a nuestro repositorio en Git Hub y confirmamos la subida de nuestros archivos, con las actualizaciones que hayamos realizado en estos:

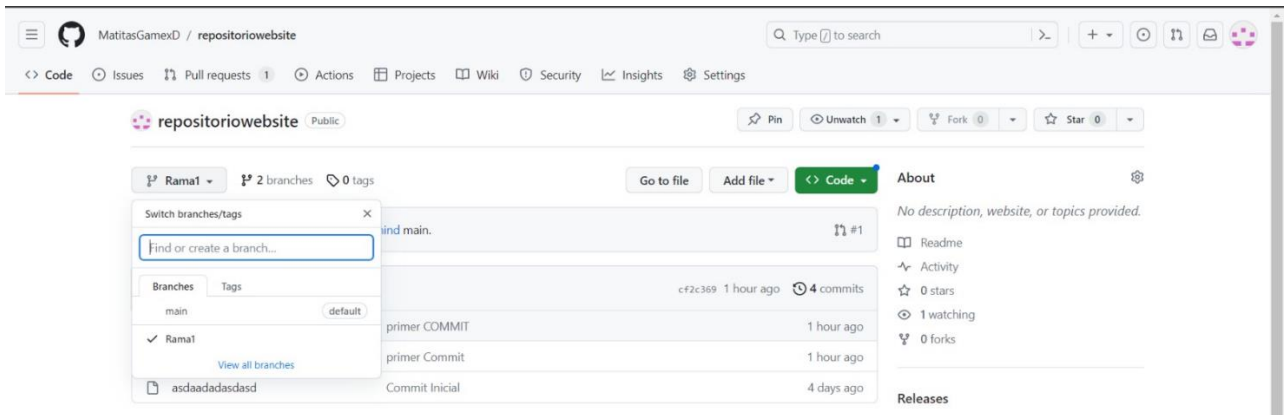


Imagen 13.

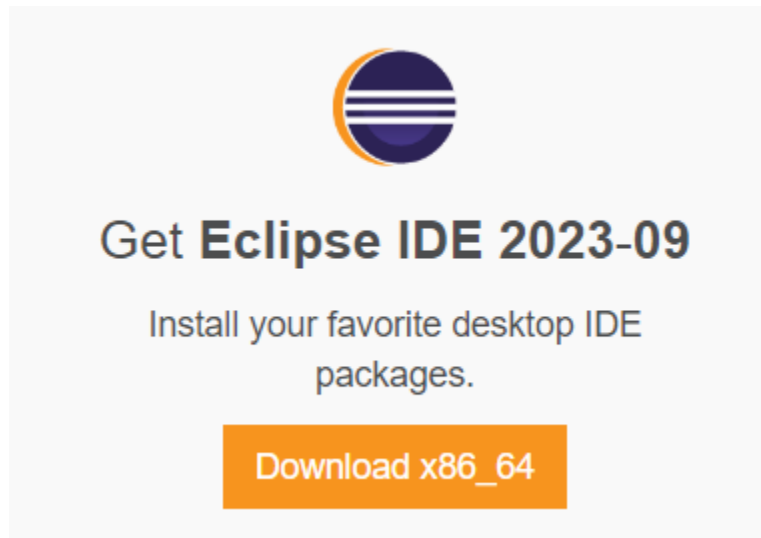
gitfv primer COMMIT		cf2c369	1 hour ago	🕒 4 commits
📄	README.md	primer COMMIT	1 hour ago	
📄	archivomanu.txt	primer Commit	1 hour ago	
📄	asdaadadasdasd	Commit Inicial	4 days ago	

Imagen 14.



Imagen 15.

12. Ya teniendo nuestro repositorio accesible en Git Hub procedemos a la creación de nuestro aplicativo web , el cual , se basara en Eclipse IDE .



- Se debe de descargar las extensiones o paquetes que requerimos para nuestro proyecto las cuales serían:



Imagen 16.



Imagen 17.

- Lo anterior con el objetivo de crear tanto las clases necesarias para el funcionamiento de nuestra aplicación (Imagen 16) y de nuestra página web (Imagen 17).

13. Ya habiendo echo la instalación, se procede a la creación de un Dynamic Web Project en Eclipse llamado TicTacToe.

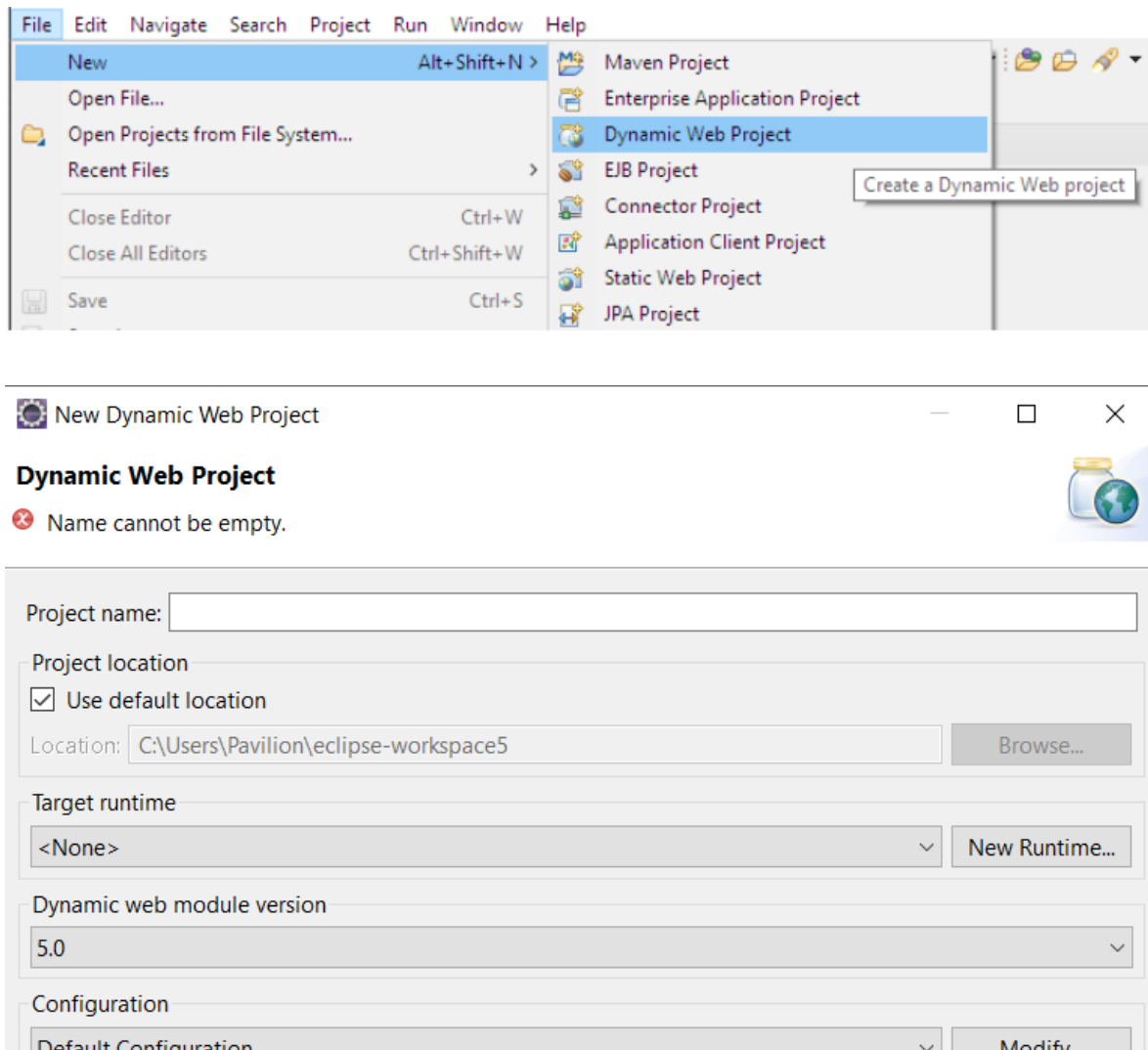
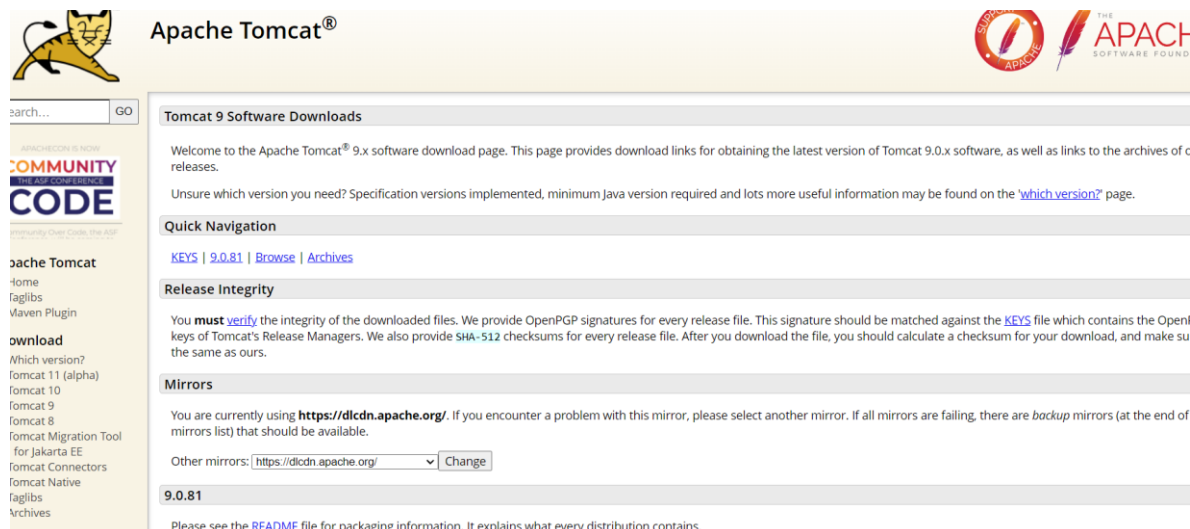


Imagen 18.

- Para que nuestra aplicación funcione de la forma correcta se necesitara de un compilador, el cual se tendrá que descargar para así poder utilizarlo, a continuación, se mostrara el proceso de instalación del compilador:




Paso 1:



The screenshot shows the Apache Tomcat 9.0.81 download page. The page has a header with the Apache Tomcat logo and the Apache Software Foundation logo. Below the header, there is a search bar and a navigation menu. The main content area is titled "Tomcat 9 Software Downloads" and contains a welcome message, a "Quick Navigation" section with links to KEYS, 9.0.81, Browse, and Archives, a "Release Integrity" section explaining the importance of verifying file integrity, and a "Mirrors" section listing available mirrors. The page also includes a sidebar with links to the Apache Tomcat community, download page, and archives.

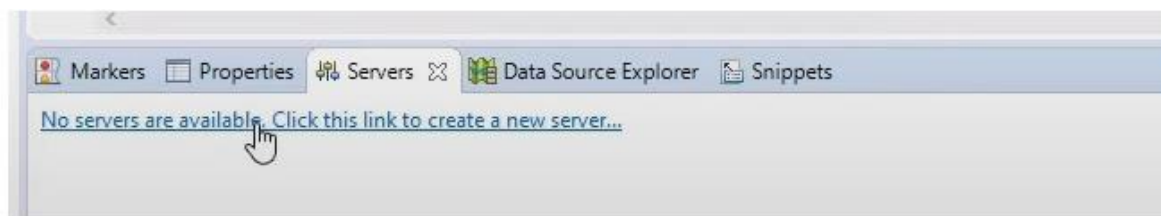
Ingresamos a la página web “ <https://tomcat.apache.org/download-90.cgi> “ y seleccionamos el archivo “ 64-bit Windows zip (pgp, sha512) “ para su descarga.

Paso 2 :

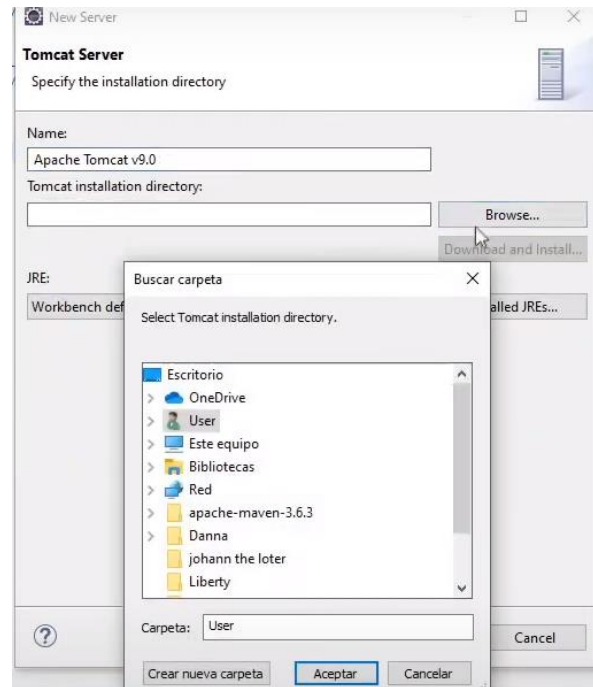
	apache-tomcat-9.0.81	9/10/2023 9:04 p. m.	Carpeta de archivos	
	eclipse	7/09/2023 2:25 p. m.	Carpeta de archivos	
	apache-tomcat-9.0.81-windows-x64.zip	10/10/2023 4:02 p. m.	Archivo WinRAR ZIP	13.037 KB

Nos dirigimos a la carpeta donde hayamos descargado el archivo y lo descomprimos, así dejándonos una carpeta.

Paso 3:

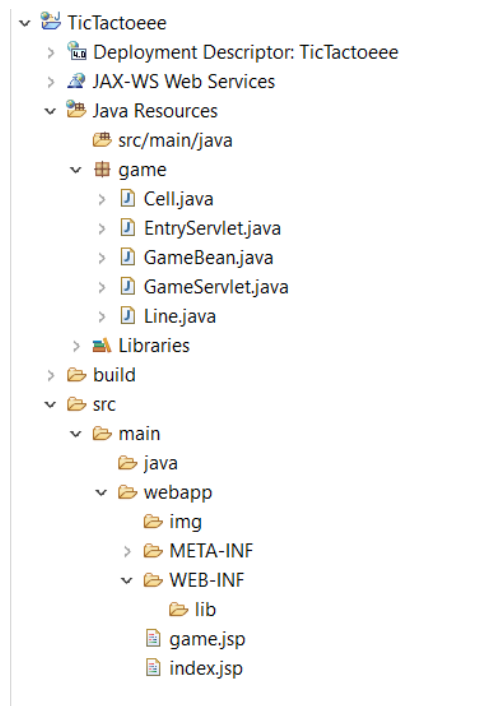


Después volveremos a Eclipse y agregaremos el compilador.

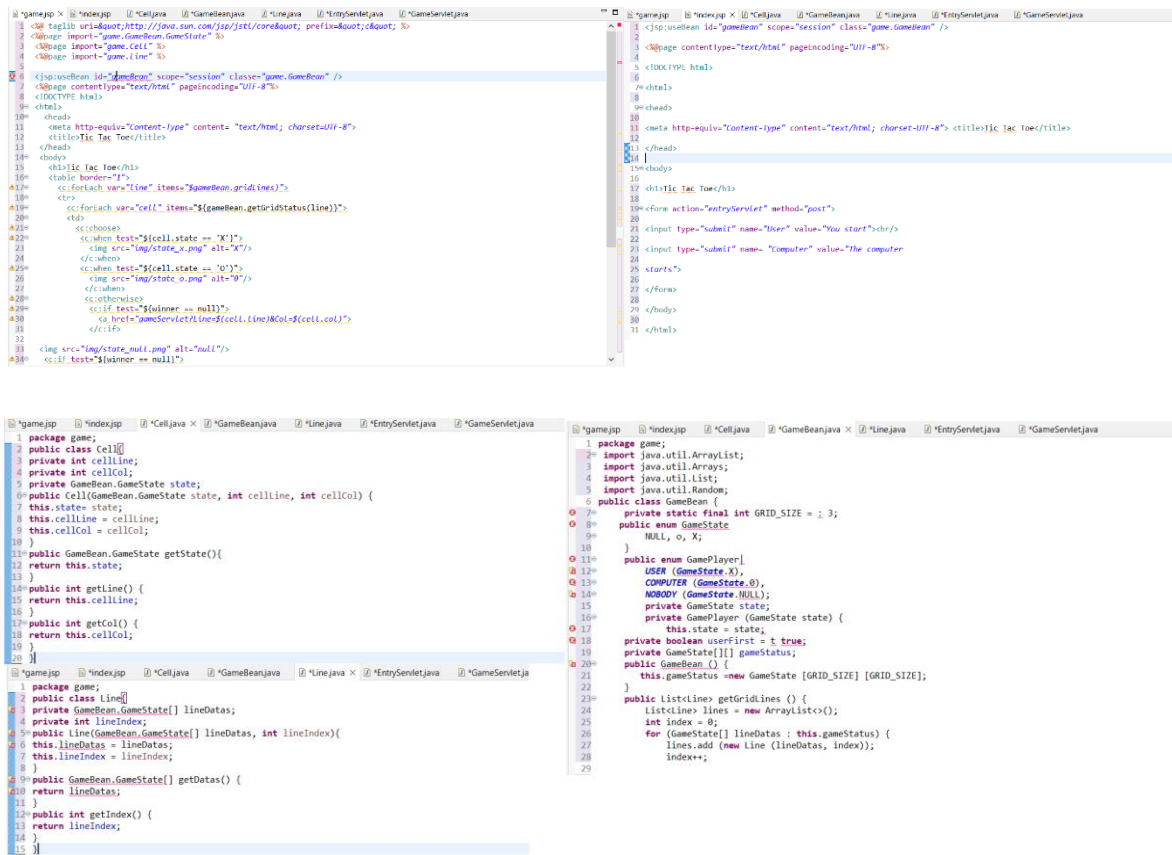


Para agregarlo seleccionaremos la carpeta en nuestro dispositivo donde hayamos descomprimido el archivo ZIP del compilador.

14. Ya habiendo realizado todo lo anterior, procedemos con la creación de nuestras clases y paquetes, donde contendremos el funcionamiento de nuestra aplicación web.



15. Se comienza agregar el código correspondiente en cada clase , para el funcionamiento de nuestra aplicación web.



- Como se puede observar el código presenta varios errores, lo cual impide el funcionamiento de nuestra aplicación, provocando que no se logre el objetivo de correrlo.

16. Un ejemplo de lo que se quería mostrar seria lo siguiente:



Imagen 19.

- En la imagen 19. Se demuestra que se realizó un nuevo archivo para demostrar que con las instalaciones anteriores ya mostradas si se logra crear una página web que muestre lo que deseamos en este caso un simple “Hola mundo”.

Análisis de Resultados:

1. A través de la realización del proyecto se lograron algunos de los objetivos propuestos para este proyecto, pero no de la forma dada o acordada por nuestro Maestro a cargo, Esto a razón de que se presentaban errores que se salían de nuestro control humano.
2. Se logra apreciar que se recurre a soluciones diferentes a las dadas en clase para la realización de los objetivos del proyecto las cuales tienen un resultado coherente con nuestros objetivos a conseguir.

Conclusiones:

- Creación del Repositorio en GitHub: Se ha creado con éxito un repositorio público en GitHub con el nombre "TicTacToe". Este paso es fundamental para gestionar eficientemente las versiones de la aplicación y permitir la colaboración en el proyecto.
- Gestión de Versiones con Git: Se ha utilizado Git para gestionar las versiones de la aplicación. Se realizó un commit inicial con el mensaje "Initial Commit" y se hizo un push del repositorio local al remoto. Esto garantiza un control adecuado de los cambios en el proyecto.
- Documentación del Proyecto: Se agregó un archivo README.md que proporciona una descripción detallada del proyecto y muestra una imagen de ejemplo. Esto facilita la comprensión del propósito del proyecto y su funcionamiento.
- Localización en español: Se modificó el proyecto para que los mensajes del juego estén en español. Esto mejora la usabilidad y accesibilidad del juego para los usuarios que hablan español.
- Actualización de Repositorio: Se actualizaron tanto el repositorio local como el remoto con los cambios realizados para la localización en español, utilizando el mensaje de commit "Actualización a idioma español". Esto mantiene el historial de versiones coherente y bien documentado.

BIBLIOGRAFIAS:

- [Microproyecto 3](#)
- [INTALACION GIT](#)
- [Apache Tomcat®](#)
- [Eclipse IDE](#)
- [GIT HUB](#)
- [UBUNTU](#)
- [REPOSITORIO GIT HUB](#)