Git y Github

1. Introducción

Git es un software de control de versiones ó CVS por sus siglas en Ingles. Este tipo de software están diseñados para porder llevar un contro de los cambios en un proyecto en el que esté involucrado el desarrollo de algun tipo de software.

Un ejemplo en el que git es usado bastante es cuando tenemos varias versiones de un proyecto:

- tesis.docx
- tesis-terminada.docx
- tesis-terminada-final.docx
- tesis-terminada-fianl-definitiva.docx

En lugar de tener tantos archivos diferentes para llevar un control de los cambios, con git tendriamos un solo archivo y este mismo se encargaria de guardar registros de los cambios que se hagan en el proyecto. Aunque estos ejemplos son con archivos de word, git funciona mejor con archivos de texto por ejemplo; .txt, .csv, .c, .java, .tex, etc. Para git es mas dificil especificar los cambios para archivos empaquetados o binarios como .exe, .docx, .pptx, .jpeg, etc.

Otra gran ventaja de git es que permite a muchas personas colaborar en un solo proyecto, de esta forma se puede agilizar el proceso de desarrollo. Para colaborar se pueden utilizar herramientas que implementan el sistema de git como GitHub, GitLab, Bitbucket, etc.

2. Instalar Git

Para poder empezar a usar git tenemos que instalarlo, algunos sistemas como Mac o Linux lo podrían tener ya instalado, Windows no trae git instalado así que siempre es necesario instalarlo. Para revisar si ya estan instalados en estos sistemas es necesario abrir el terminal. En linux el terminal se abre presionando las teclas ctrl+alt+t y en Mac se puede abrir el terminal presionando las teclas cmd+espacio y luego ecribiendo la palabra terminal. Una vez abierto el terminal tenemos que escribir el commando:

Listing 1: Comando para revisar versión de Git

```
git --version
```

Si lo tenemos instalado deberiamos ver una linea parecida a la siguiente:

Listing 2: Versión de Git

```
git version 2.24.3
```

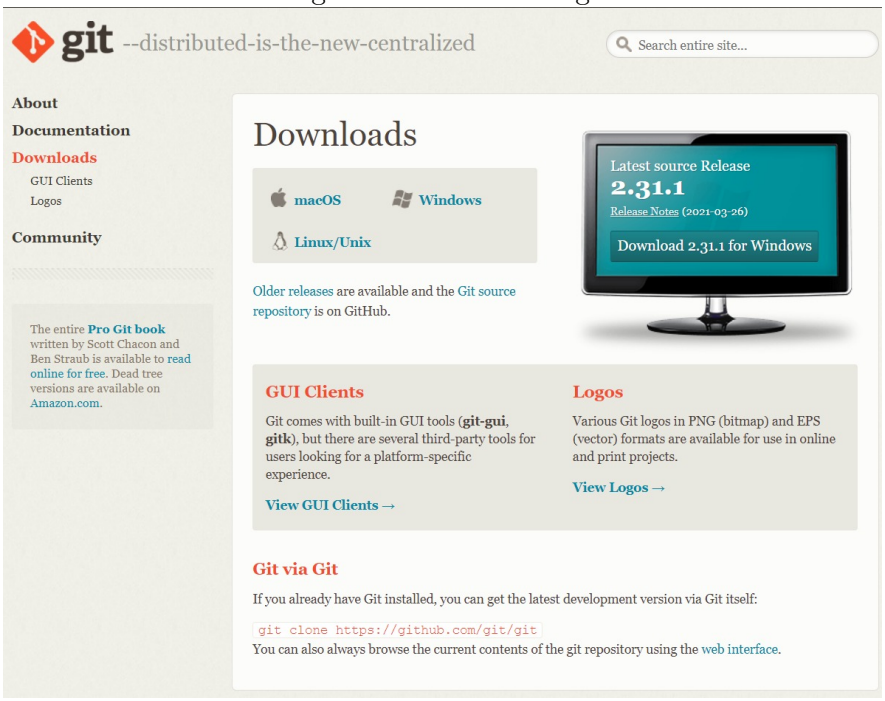
Si no lo tenemos instalado veremos algo parecido a esto:

Listing 3: Git no instalado

```
Unknown command: git
```

En caso que git no este instalado tendremos que ir al sito web https://git-scm.com/downloads y darle click al botón de descargar. En la figura 1 se muestra la pagina donde descargar git.

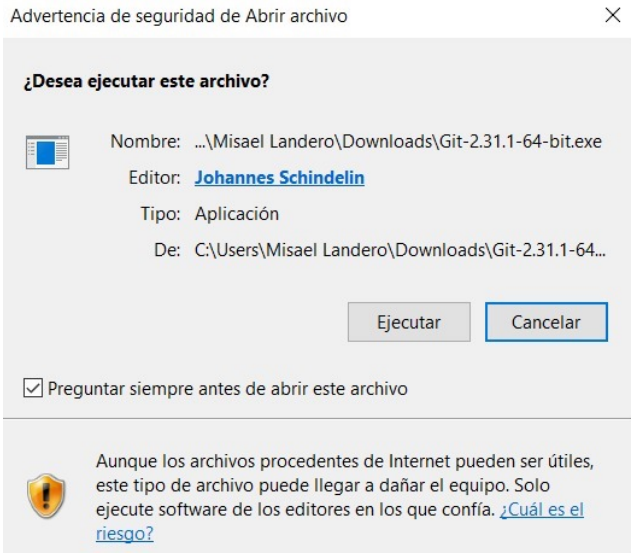
Figura 1: Sitio web de git



2.1. Windows

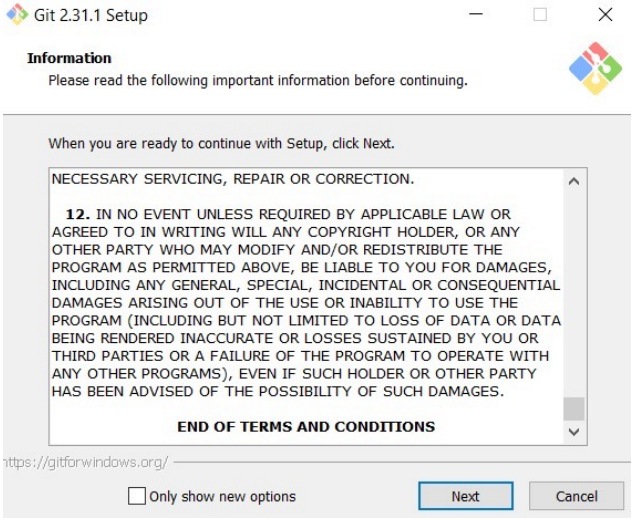
Una vez descargado el git hay que ejecutar el programa. Aparecera la siguiente ventana y hay que darle click en el botón Ejecutar:

Figura 2: ejecutar programa git



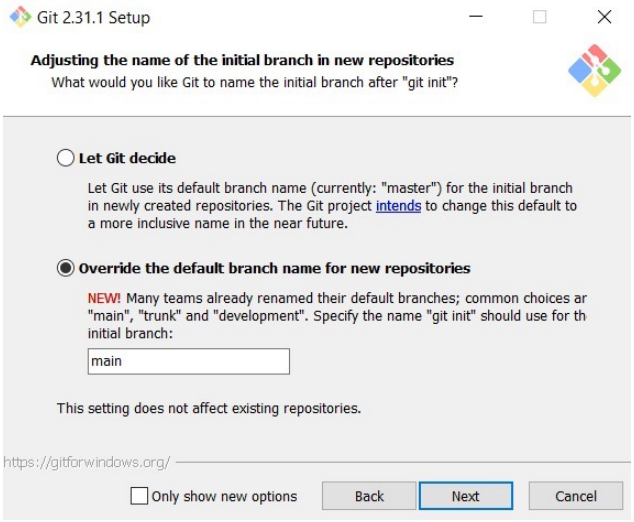
Luego se arbrira la ventana de instalación, en la cual le daremos siguiente a la mayoría de las opciones.

Figura 3: ejecutar programa git



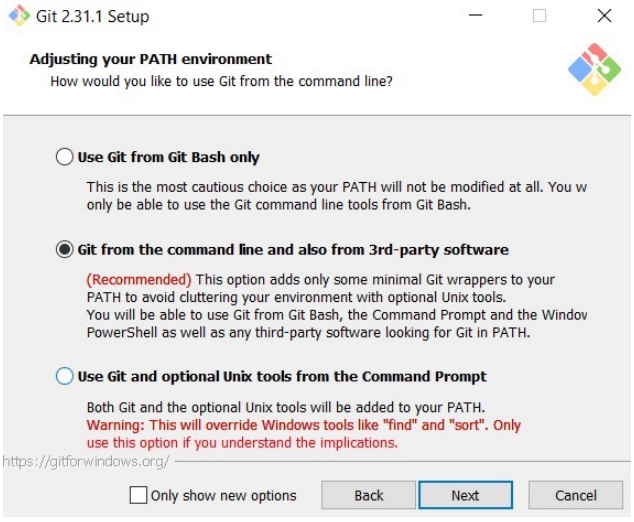
Luego de darle click a siguiente una cuantas veces llegaremos a una opción par definir el nombre por defecto de la rama principal, este lo cambiaremos a main en lugar de master ya que GitHub cambió su configuración para que la rama principal se llame main. De esta forma nos ahorraremos cambiarle el nombre a la rama maualmente.

Figura 4: ejecutar programa git



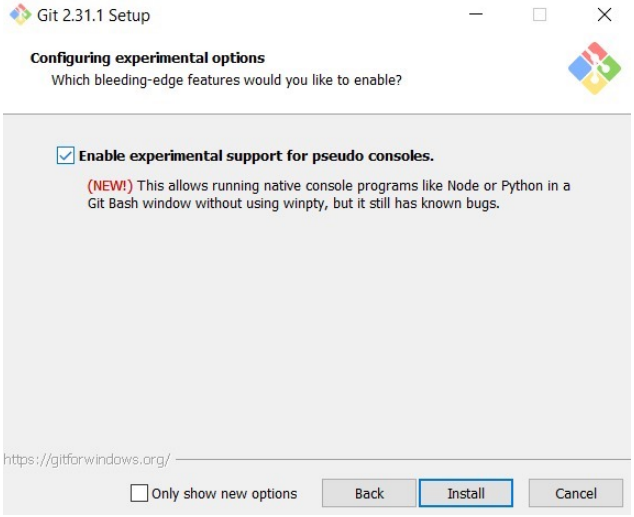
La siguiente opción que tendremos que modificar es desde donde podemos ejecutar git, eligiremos la opción de ejecutarlo desde la linea de comando y desde software de terceros (esto nos facilitará usarlo).

Figura 5: ejecutar programa git



La última opción que tenemos que agregar es el de usar pseudo consolas, esto ayuda a poder ejecutar comandos de python o node desde la consoloa de git.

Figura 6: ejecutar programa git



Despues de seleccionar la opción anterior le damos click en el botón Instalar y con esto ya tendremos git instalado.

2.2. Mac

La instalación de git en mac es mucho mas sencilla, pero antes de poder instalarlo tendremos que instalar un manejador de paquetes. El manejador que paquetes que instalaremos se llama HomeBrew, esta dispobible en el siguiente enlace: https://brew.sh/.

Para instalar HomeBrew tendremos que abrir el terminal de Mac y escribir el siguiente comando:

Listing 4: Comando para instalar HomeBrew

```
/bin/bash -c "$(curl -fsSL https://raw.githubusercontent.com/
   Homebrew/install/HEAD/install.sh)"
```

Una vez instalado HomeBrew, instalar git es sencillo. Siempre en el terminal escribiremos el siguiente comando:

Listing 5: Comando para instalar git en Mac

```
brew install git
```

2.3. Linux

Por último, instalar git en Linux es mas facil que en los sistemas operativos anteriores. Primero abriremos el termial y luego dependiendo de la distribución de Linux que usemos escribiremos el siguiente comando:

2.3.1. Ubuntu/Debian

Listing 6: Comando para instalar git en Ubuntu

```
apt-get install git
```

2.3.2. Fedora

Listing 7: Comando para instalar git hasta Fedora21

```
yum install git
```

Listing 8: Comando para instalar git en Fedora22 y superiores

```
dnf install git
```

Para otras distribuciones de linux revisar el siguiente enlace: https://git-scm.com/download/linux

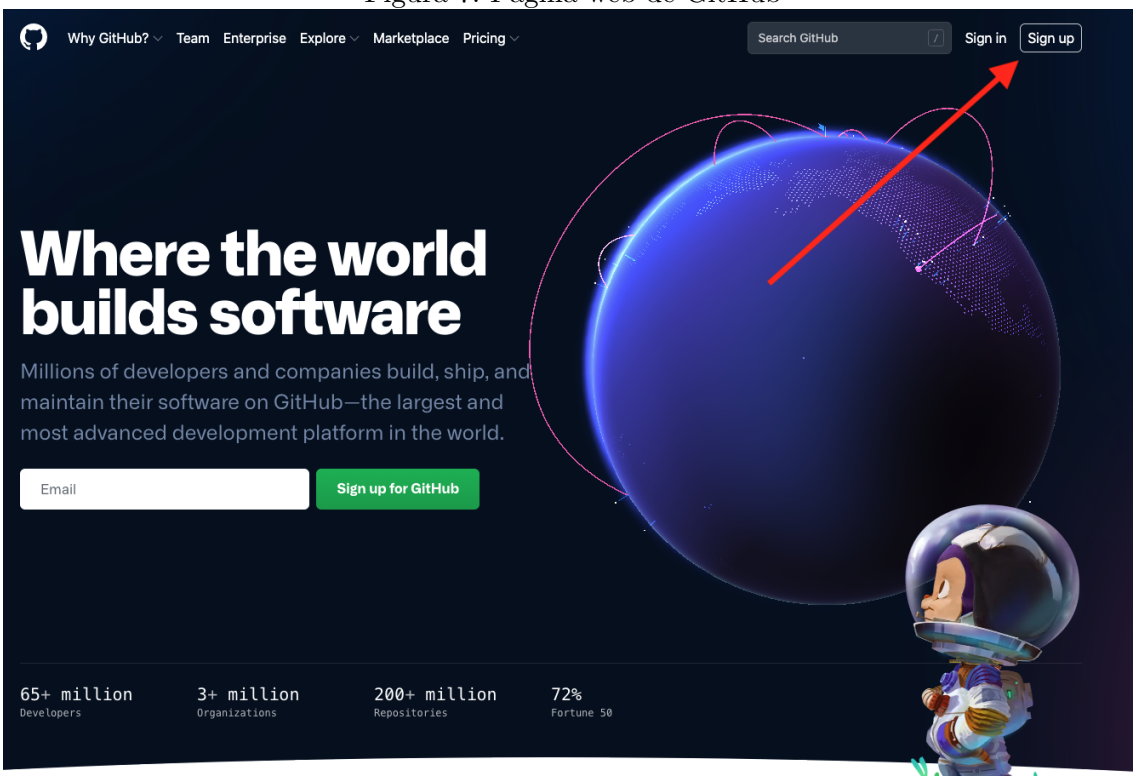
3. GitHub

Github es una plataforma que implemeta Git y hace mas facil poder colaborar con varias personas en proyectos de codigo a traves de la web. GitHub es relativamente sencillo de usar, pero tiene funcionalidades bien robustas para programadores experimentados.

3.1. Registrarse en GitHub

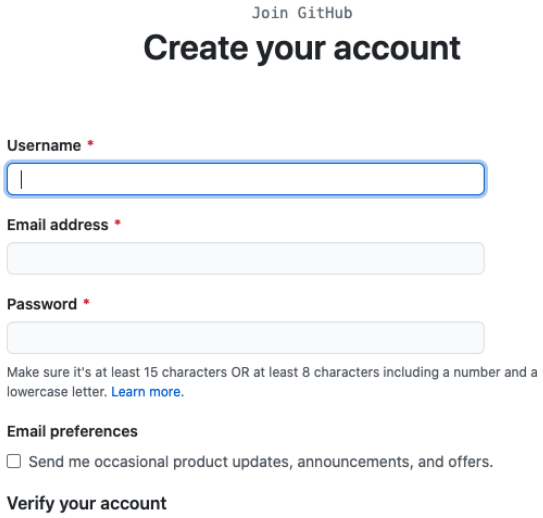
Para empezar a usar GitHub primero hay que crear un perfil en, para esto hay que ir a la dirección web https://github.com/ y hacer click en el botón de la esquina superios izquierda "Sign Up".

Figura 7: Página web de GitHub



Una vez hayamos dado click en el botón SignUp entratemos a una página donde tendremos que poner nuestros datos para crear la cuenta.

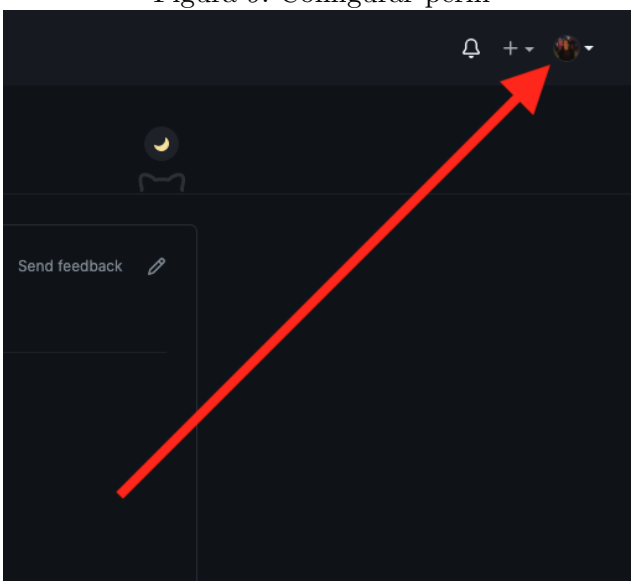
Figura 8: Registrarse en GitHub



3.2. Datos de perfil

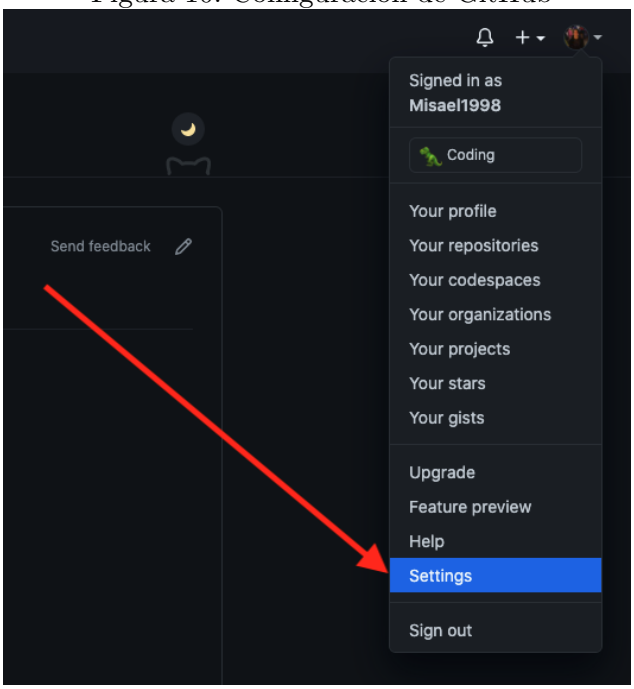
Cuando ya tengamos la cuenta deberiamos modificar nuestros datos de perfil ya que GitHun no añade datos como nombre o foto de perfil, si no que hay que ponerlos manualmente. Para agregar estos debemos buscar en la esquina superior derecha un circulo donde debería estar la foto por defecto para nuestro perfil y le damos click.

Figura 9: Configurar perfil



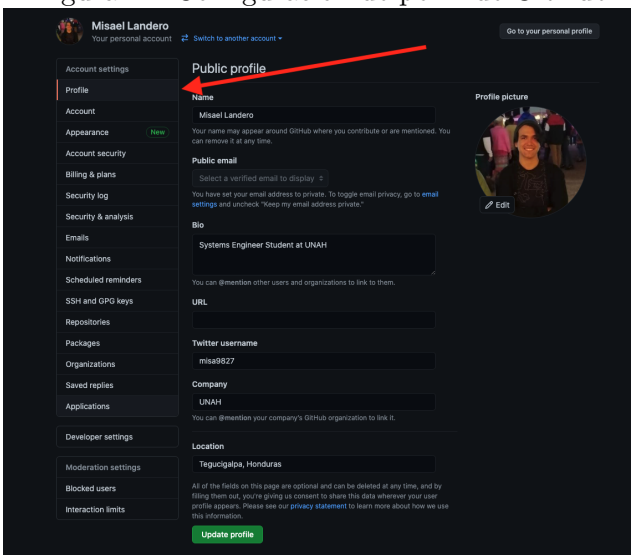
Luego de dar click se desplegara un menú y buscamos la opción de configuración ó settings.

Figura 10: Configuración de GitHub



Esto no llevara a la página de configuración de GitHub, en la barra lateral izquierda buscaremos la opción de perfil para poder empezar a agregar nuestros datos.

Figura 11: Configuración de perfil de GitHub

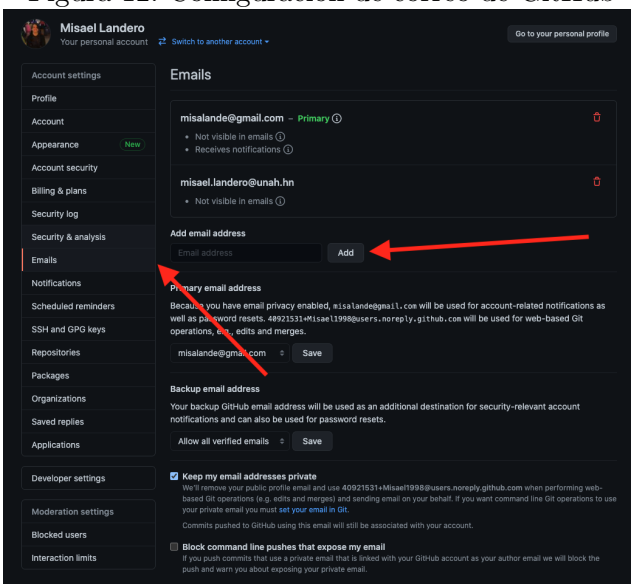


Aquí agregaremos información pública como el nombre, foto de perfil, cuenta de twitter, etc.

Si tenemos un correo de una universidad/institucional lo podemos agregar como correo adicional y esto desbloqueará una plan que normalmente es pagado. Para agregar un correo alternativo buscamos en la barra lateral izquierda la opción de correo ó emails y le damos click. Allí podemos

agregar correos adicionales y elegir la visibilidad de estos mismos en nuestro perfil.

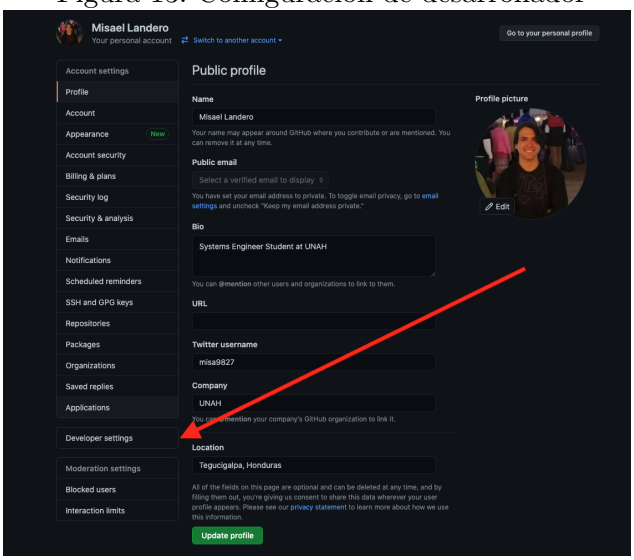
Figura 12: Configuración de correo de GitHub



3.3. Tokens de autenticación

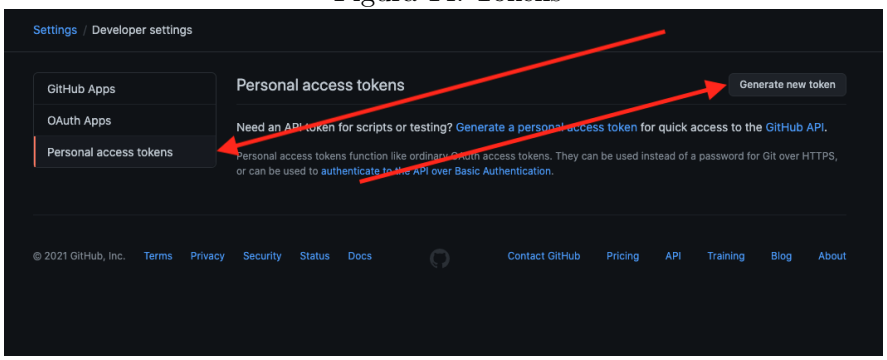
La última configuración que hay que hacer en GitHub es la de el token de autenticación, esto nos permitira subir nuestros repositorios y cambios hechos en nuestras compuradoreas. Anteriormente GitHub usaba correo y contraseña para autenticación, pero lo están cambiando a tokens con el proposito de hacer más segura la comunicación. Existen otras técnicas como SSH pero los tokens son más faciles de usar. Para empezar a usar los tokens tenemos que entrar en las herramientas de desarrollador que están en la barra lateral izquierda.

Figura 13: Configuración de desarrollador



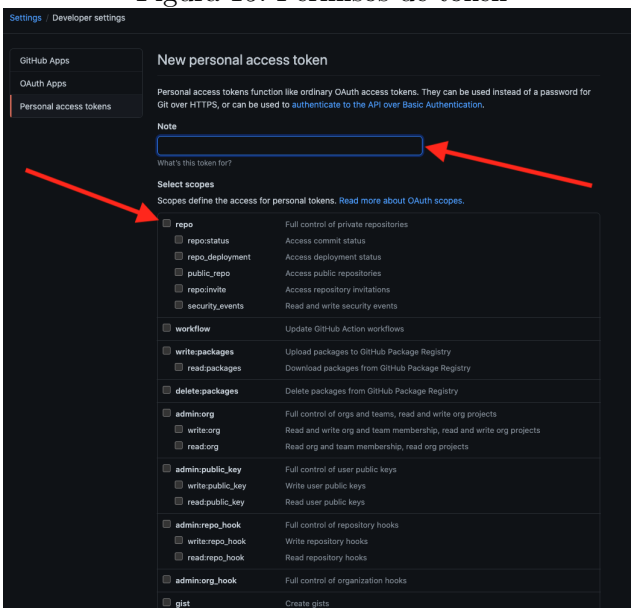
Luego buscamos la opción de tokens personales en la barra lateral izquierda y le damos click. Luego le damos click al botón de generar token para crear nuestro primer token.

Figura 14: Tokens



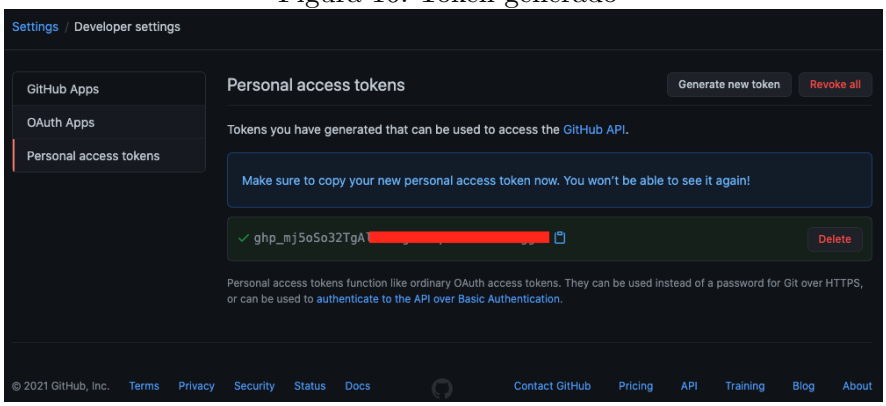
Luego hay que seleccionar lor permisos que tendrá el token, para efectos de simplicidad solo seleccionaremos repo. Junto con esto podemos escribir una breve descripción del proposito del token.

Figura 15: Permisos de token



Una vez creado el token tenemos que asegurarnos de copiarlo y guardarlo en un lugar seguro, ya que este token es el acceso a los repositorios en los que estamos trabajando. En caso de perder el token se puede simplemente borrar y crear uno nuevo, esta es la ventaja de usar tokens ya que no perdemos acceso a nuestra cuenta en GitHub y no configuramos llaves SSH desde cero.

Figura 16: Token generado



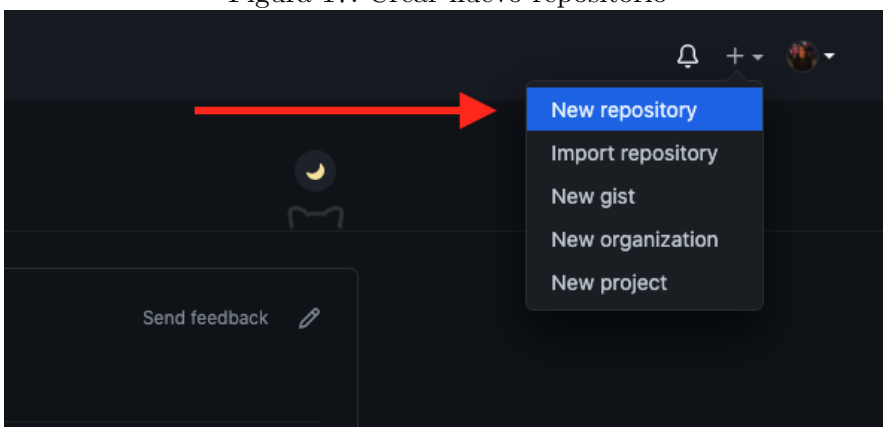
4. Repositorios

Un repositorio es donde guardamos el proyecto, se podría decir que es una convención que se usa para indicar que una carpeta o directorio contiene cierto tipo de archivos o proyectos. Para que Git empiece a llevar un registro de los cambios en el proyecto lo primero que tenemos que hacer es inicializar un repositorio, hay muchas formas para inicializar un repositorio y cada una puede ser util en diferentes ocaciones pero al final todas hacen lo mismo. Una de las formas mas convenientes es crearlo en GitHub y despues descargarlo, de esta forma podemos definir varias configuraciones del repositorio sin tener que usar el terminal o crear ciertos archivos directamente.

4.1. Crear repositorio en GitHub

Para crear el repositorio en GitHub tenemos que buscar el botón [+] en la esquina superior derecha, luego le damos click y seleccionamos nuevo repositorio.

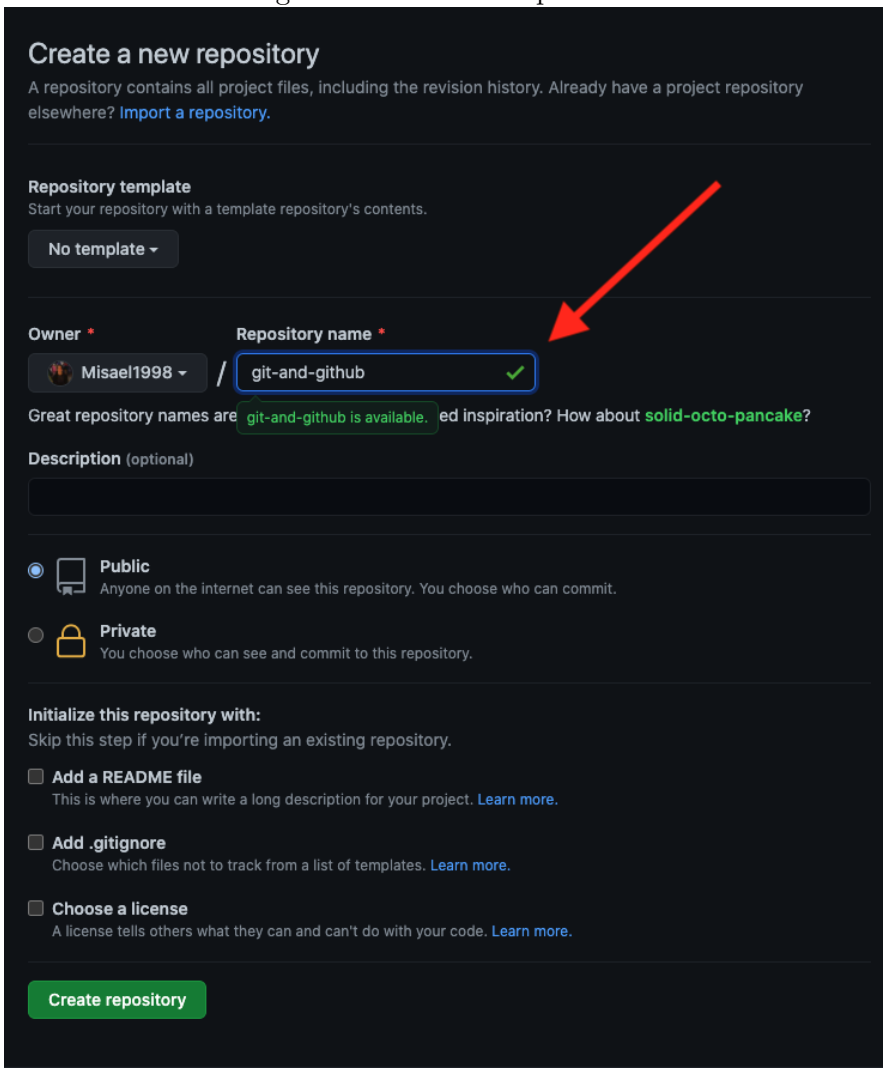
Figura 17: Crear nuevo repositorio



Esto nos llevará a un menú de configuración para el repositorio, lo primero que haremos sera nombrarlo. Para nombrar un repositorio tenemos que seguir ciertas restricciones como, solo letras

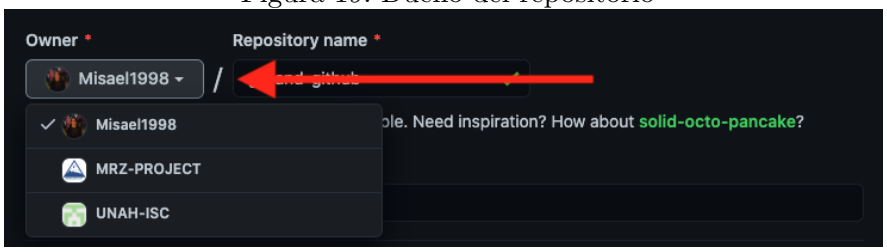
minusculas, no se aceptan espacios, el nombre tiene que ser único dentro de nuestros repositorios, etc.

Figura 18: Nombrar repositorio



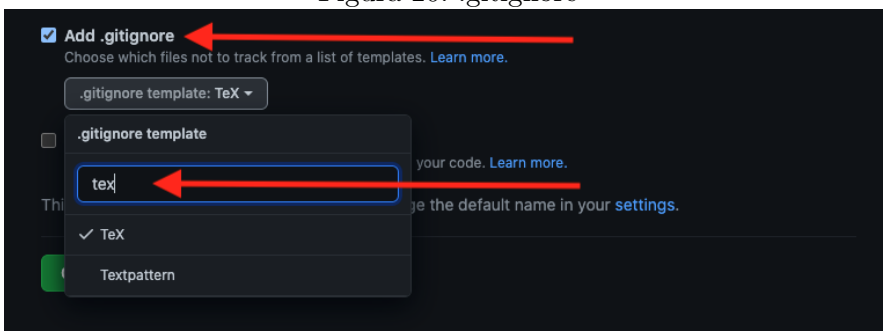
Si tenemos multiples usuarios ó formamos parte de varias organizzaciones tendremos que especificar quien es el dueño del repositorio (las organizaciones las explicare mas adelante), esta opción la encontramos al lado izquierdo del nombre del repositorio.

Figura 19: Dueño del repositorio



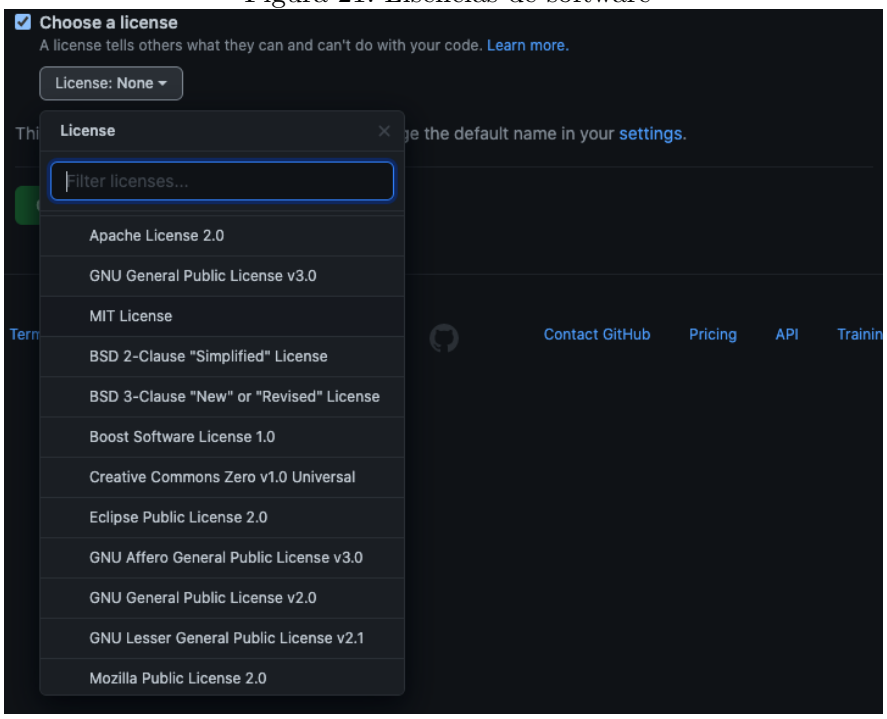
Luego de esto tenemos la opción de agregar un gitignore, esto sirve para excluir archivos de git i.e. decirle a git cuales archivos no queremos que registre cambios. Para el .gitignore podemos elegir una plantilla para un tipo específico de proyecto, por ejemplo si estamos trabajando en un LaTex podemos buscar la plantilla tex y esto exluira los archivos .log, .dvi, etc. Así git solo se interesa en llevar cambios de los archivos que necesitamos.

Figura 20: .gitignore



La siguiente opción que podemos agregar es el uso de una lisencia de software, igual que con la plantilla de .gitignore podemos solo buscar el tipo de lisencia que queremos agregar y marcarla. Aquí podemos agregar de una vez si estamos usando GPL, APACHE, MIT u otras, claro que si cambiamos de lisencia se puede modificar mas adelante.

Figura 21: Lisencias de software



5. Commits

6. Branches

7. GitHub y Organizaciones