**REPOSITORIOS Y CONTROL DE VERSIONES.**

**¿Qué es un repositorio?**

En un lenguaje natural podríamos referirnos a un repositorio como el lugar donde almacenaremos y gestionaremos información, puede ser simplemente un sistema de archivos en un disco duro, una Base de Datos o por ejemplo podríamos decir que una Biblioteca es un gran repositorio, porque allí se almacenan libros y artículos de interés, la parte de gestión es cuando la biblioteca es administrada y sus libros actualizados.

En nuestra área los repositorios serán esos lugares dentro de nuestro equipo o en la nube donde como se mencionó anteriormente almacenaremos información (Documentos, Código fuente) y dicha información será gestionada por miembros de nuestro equipo de trabajo, para esto haremos uso de herramientas que nos faciliten el control de versiones.

**¿Y que es un sistema de Control de Versiones?**

El control de versiones es un sistema que registra los cambios realizados sobre un archivo o conjunto de archivos a lo largo del tiempo, de esta manera se puede recuperar versiones específicas más adelante.

Sabemos que una versión representa los diferentes estados o actualizaciones de algo, si en un grupo de trabajo creamos un documento tenemos la versión inicial, pero si después de un tiempo ese documento es modificado por alguno de los integrantes del equipo, pero se requiere tener un histórico del documento original, pues ya podemos estar hablando de una segunda versión y así sucesivamente.

Ahora, un Sistema de Control de Versiones permite la gestión de los cambios que se realicen sobre determinado elemento (en nuestro ejemplo el documento), así podremos administrar quien realizo que cambio, cuando lo realizo, que modifico, y tener el documento actualizado sin que se pierdan las versiones anteriores.

Existen muchas herramientas como Subversión, Mercurial, GIT entre otras que nos brindarán el entorno necesario para construir nuestros repositorios.

En el siguiente enlace se profundiza un poco más sobre este tema: <https://git-scm.com/book/es/v1/Empezando-Acerca-del-control-de-versiones>

**GIT**

Tomando la definición de Wikipedia “Git es un software de control de versiones diseñado por Linus Torvalds, pensando en la eficiencia y la confiabilidad del mantenimiento de versiones de aplicaciones cuando éstas tienen un gran número de archivos de código fuente. Su propósito es llevar registro de los cambios en archivos de computadora y coordinar el trabajo que varias personas realizan sobre archivos compartidos”

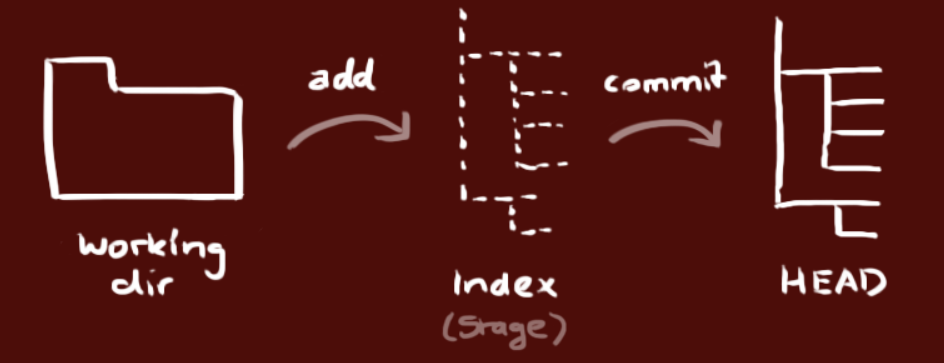
**Conceptos Básicos.**

Cuando trabajamos con GIT a nivel general tendremos un esquema de tres “Arboles” que representan el estado o flujo de trabajo de nuestra información, estos son el **directorio de trabajo**, el **index** y el **HEAD**

**Directorio de trabajo:** Este es el espacio inicial donde se encuentran los archivos que queremos gestionar en el repositorio.

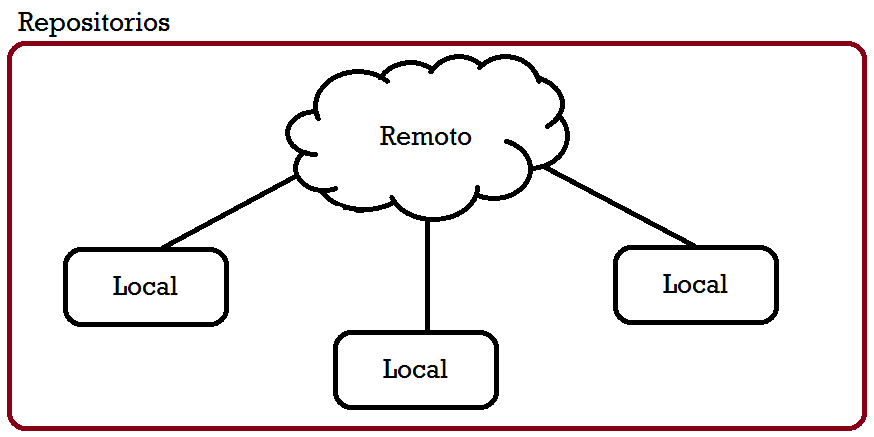
**Index:** representa un lugar o zona intermedia donde se van a alojar los archivos antes de ser cargados en el repositorio local en su totalidad.

**HEAD:** Este es el lugar o zona donde finalmente serán alojados los archivos del repositorio local antes de pasar al repositorio remoto.



Como vemos en la gráfica se pueden evidenciar los bloques o zonas por los que pasa la información, es importante tener claro este flujo de trabajo en el que desde nuestro ambiente local donde está la información originalmente debemos agregar (add) los archivos al index para luego mediante un commit ser enviados al head y desde allí tenerlos listos para poderlos almacenar en un repositorio remoto.

Basado en lo anterior cuando usamos una herramienta de control de versiones como GIT internamente vamos a tener un repositorio local donde trabajamos directamente con la información y un repositorio remoto donde finalmente va a estar alojada dicha información, de esa manera podemos trabajar varias personas en un mismo repositorio donde cada uno tendrá su propio ambiente local (trabajando con el mecanismo explicado anteriormente) para al final apuntar hacia un único ambiente remoto.



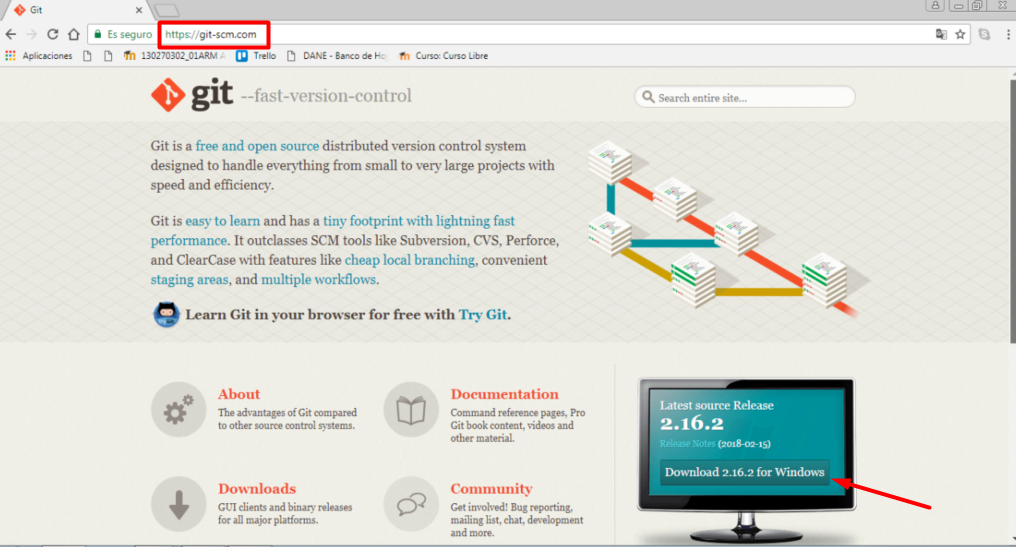
Así cada miembro del equipo tendrá inicialmente la versión original del sistema o información pudiendo realizar modificaciones en su ambiente local que luego serán sincronizadas para el resto del equipo de trabajo al subirlas (commit and push) al ambiente remoto compartiendo los cambios a todos los usuarios.

**GITHUB**

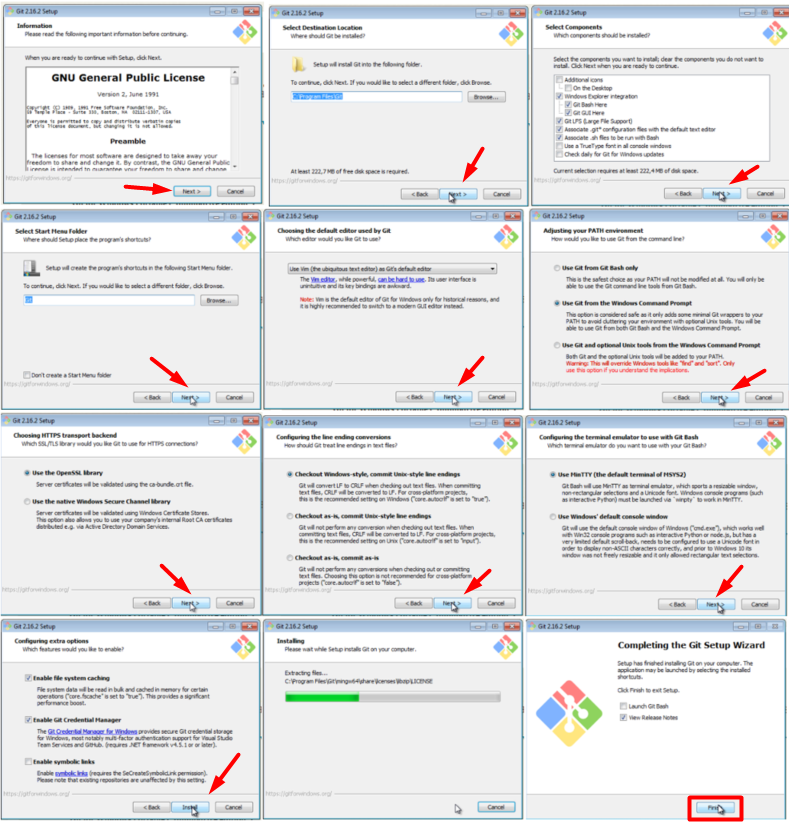
GitHub es un espacio gratuito para alojar proyectos en la nube utilizando el sistema de control de versiones Git, el código se almacena de forma pública, aunque también se puede hacer de forma privada mediante opciones de pago.

**DESCARGA E INSTALACIÓN DE GIT**

Inicialmente desde la página oficial <https://git-scm.com/>



Al descargar el instalador damos siguiente siguiente y listo.

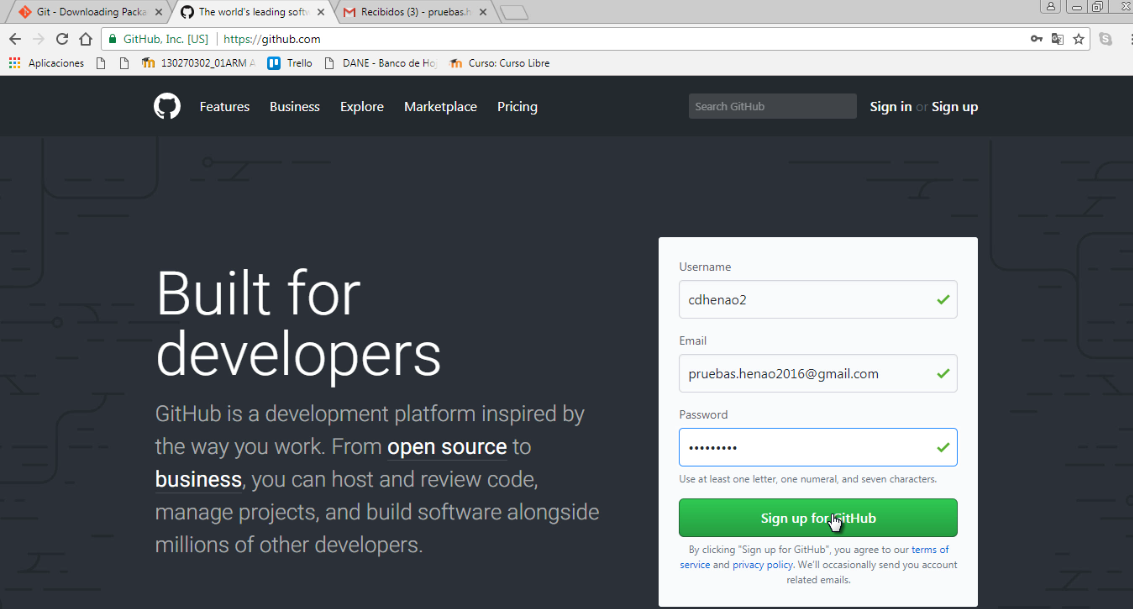


Después de esto procedemos a verificar la instalación buscando en inicio la consola de git

|  |  |
| --- | --- |
|  | Seleccionamos Git Bash y en la consola escribimos  ***git --version*** |

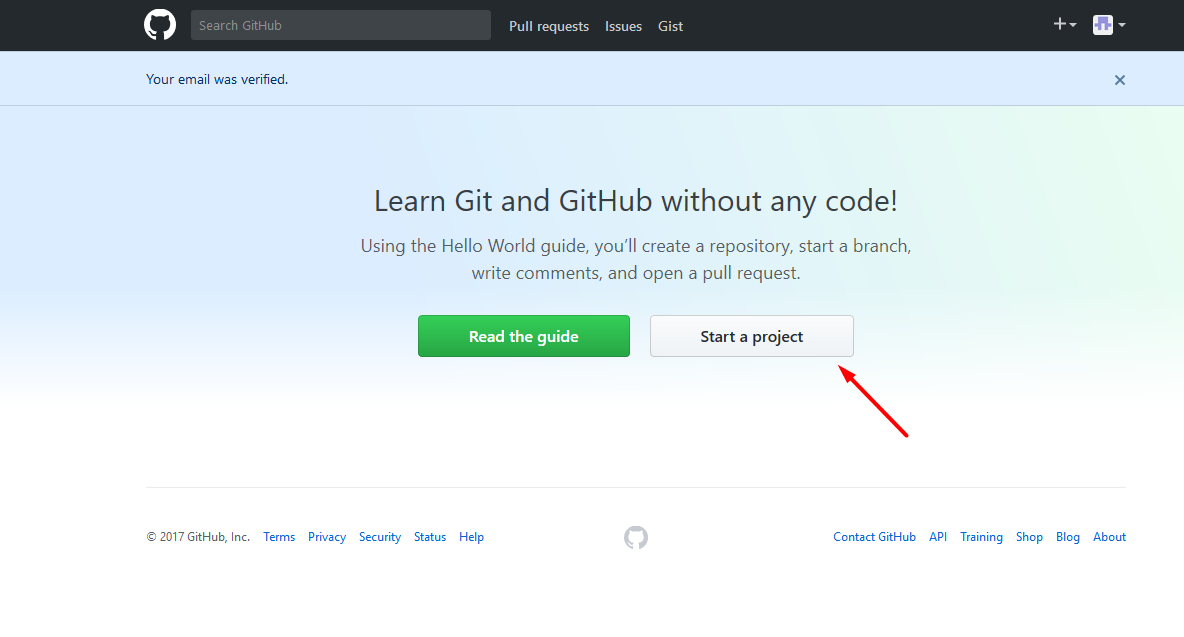
**CREAR REPOSITORIO EN GITHUB.**

Para crear un repositorio Inicialmente debemos tener una cuenta en el sitio oficial de GitHub <https://github.com/>

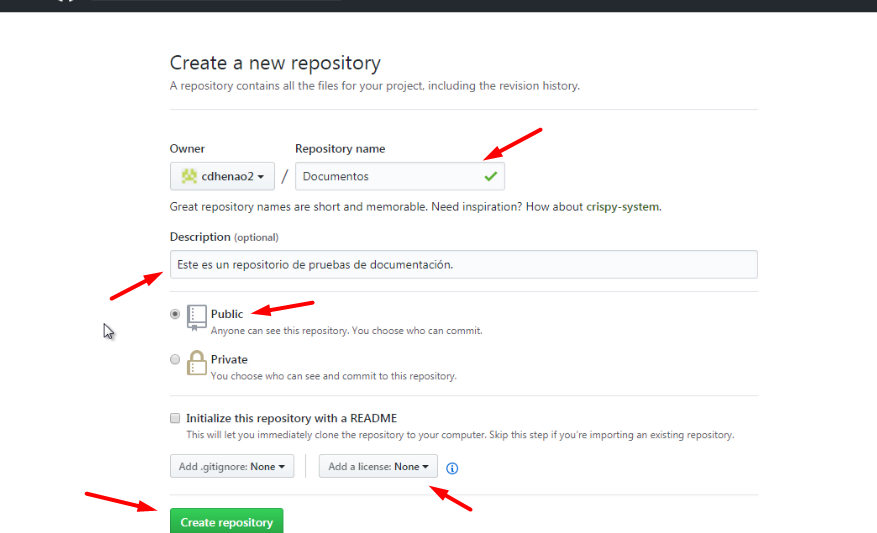


Después del registro podremos confirmar en nuestro correo electrónico el enlace para iniciar el proceso, cuando inicia damos continuar en las configuraciones.

Si deseamos podemos crear un repositorio directamente ingresando al botón Start a project

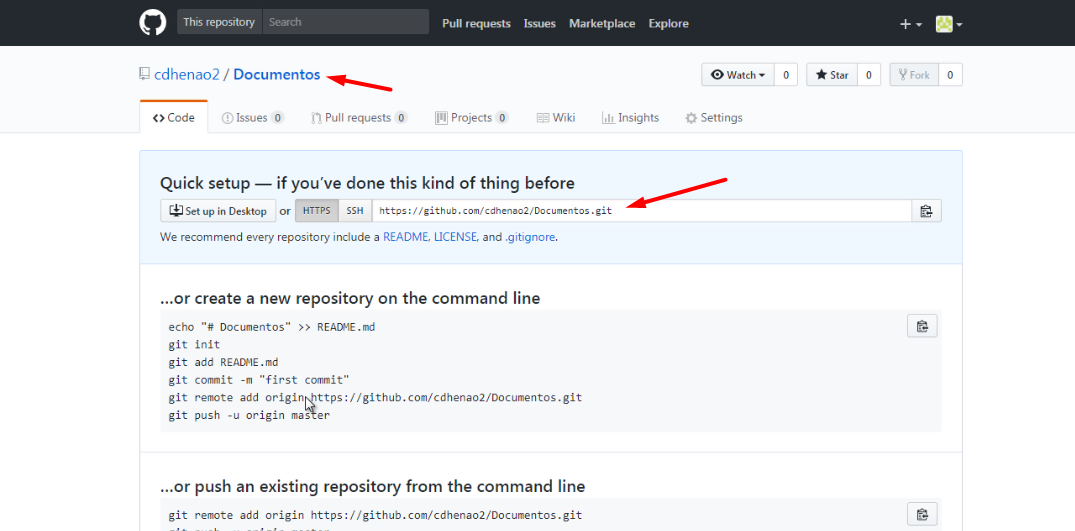


Al hacerlo diligenciamos el formulario para la creación de nuestro repositorio.

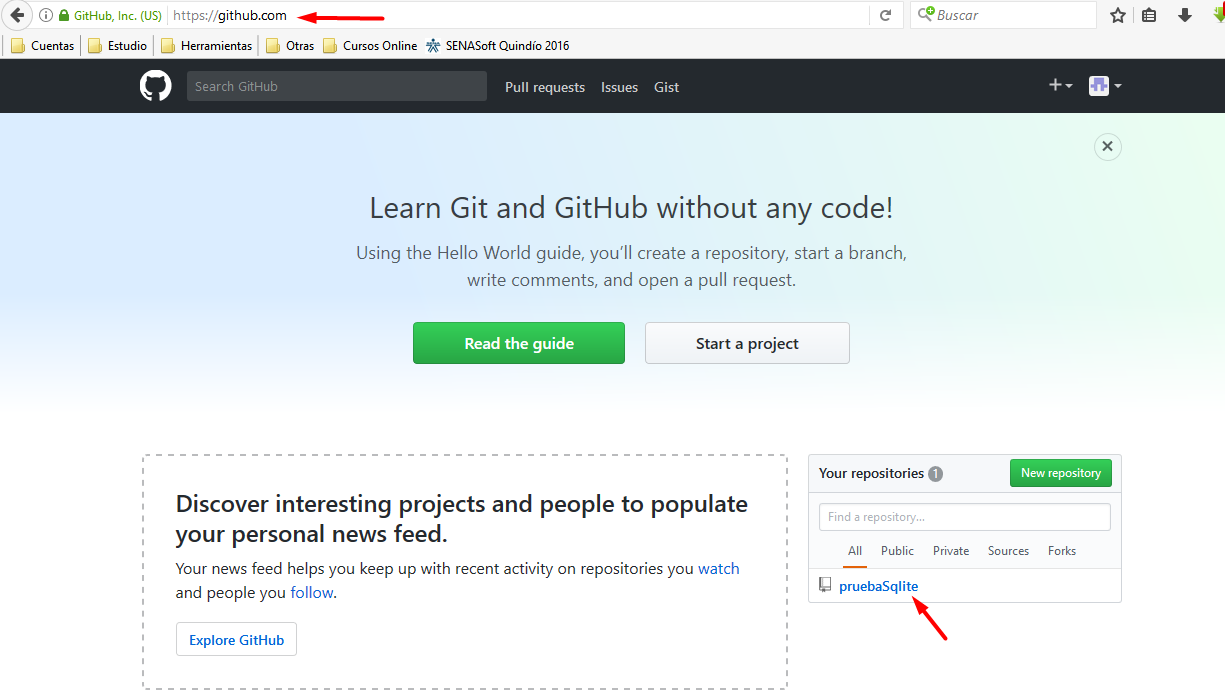


Se debe diligenciar el nombre del repositorio, descripción, lo definimos como público y de forma opcional podemos establecer el tipo de licencia de uso.

Al final se presenta la página con la dirección de nuestro repositorio, se debe tener en cuenta que en este momento lo que tenemos es el espacio para subir nuestros proyectos.



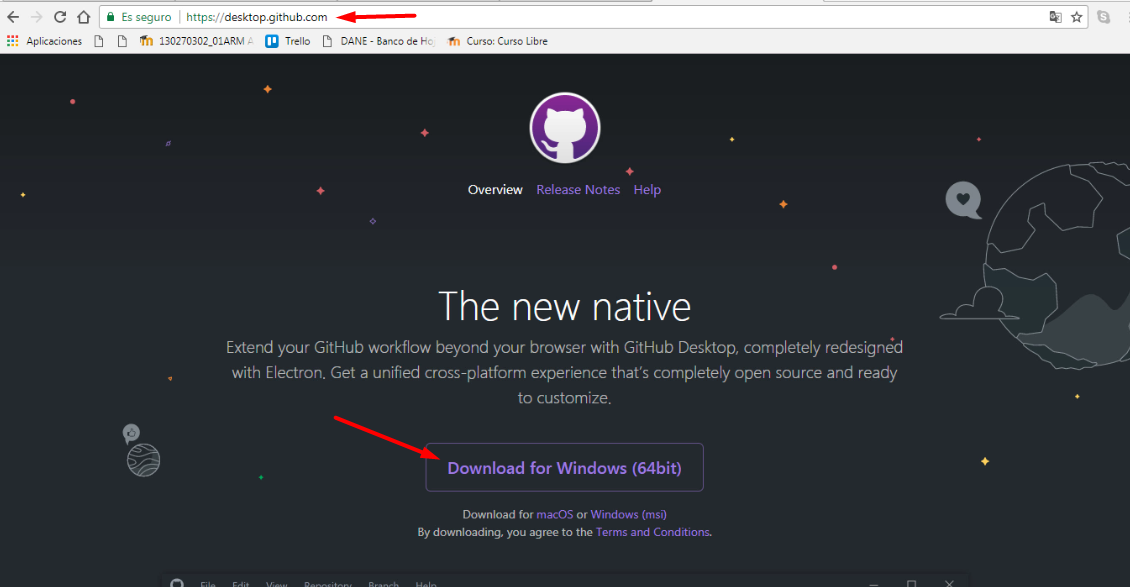
Al ingresar nuevamente a la página oficial podremos ver nuestro repositorio creado.



**INSTALAR CLIENTE GITHUB**

Después de haber creado nuestra cuenta, podemos descargar el cliente GitHub para tenerlo en instalado en nuestro pc, en este caso descargamos la versión para Windows desde la página oficial.

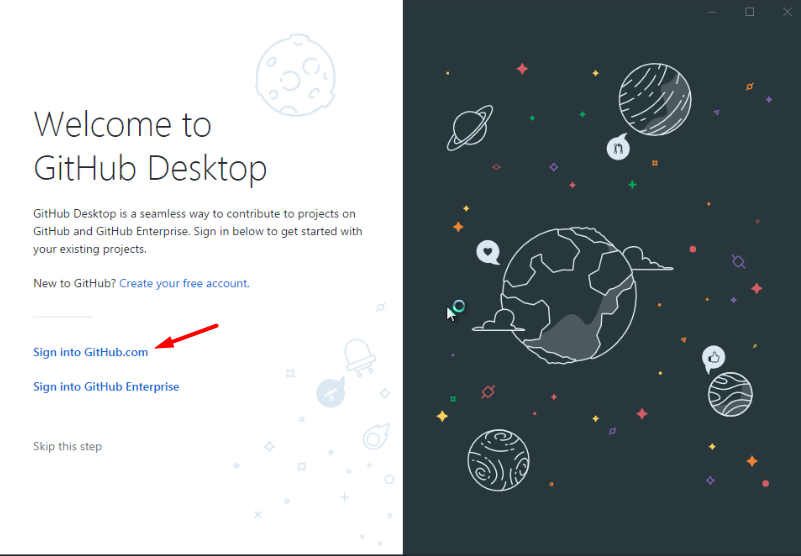
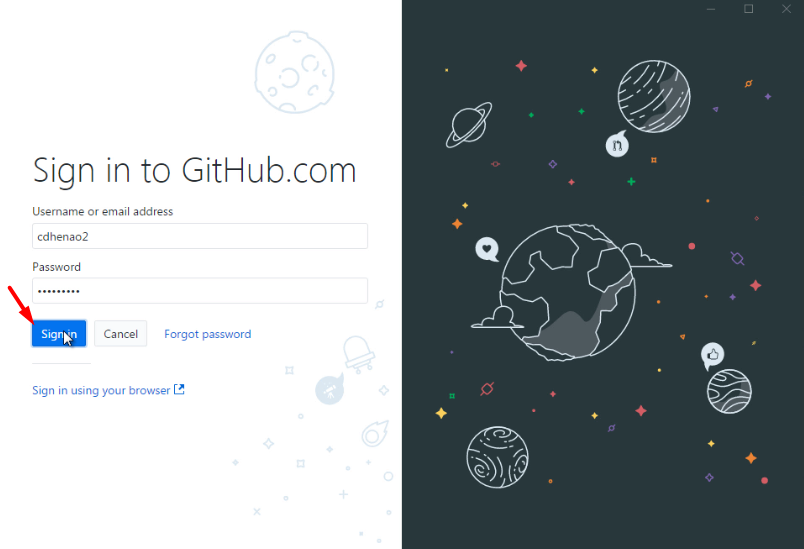
<https://desktop.github.com/>



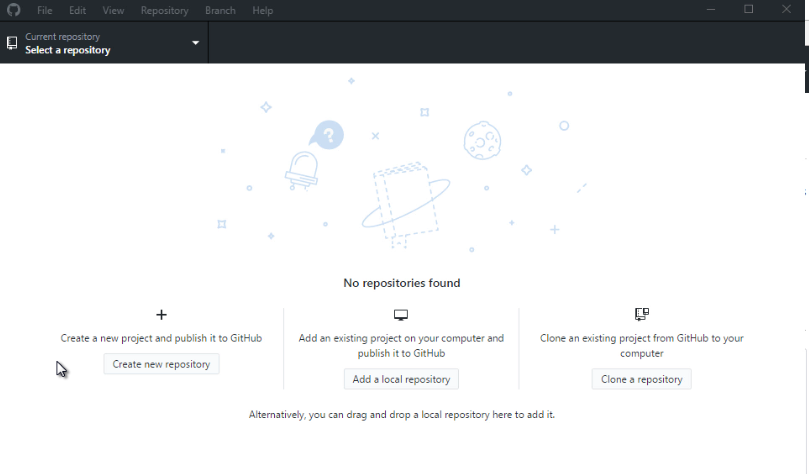
Al hacerlo se descarga el instalador, posteriormente lo ejecutamos y seguimos el paso a paso.

(El proceso de instalación no presenta mayor grado de dificultad ni demora, sin embargo en caso de producirse algún fallo se deberá intentar nuevamente.)

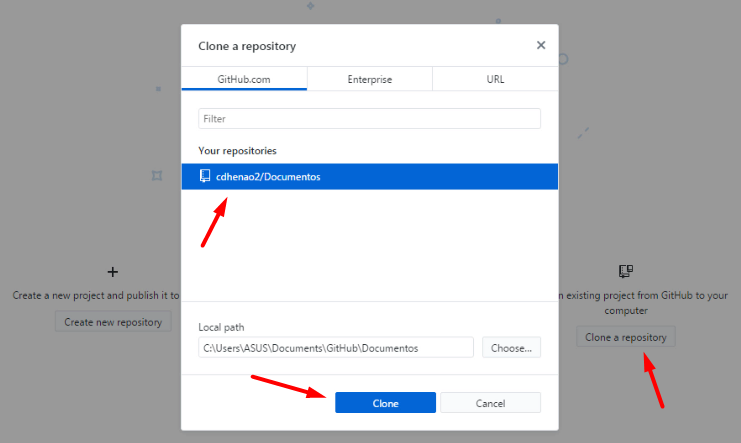
Inicialmente se presenta una ventana donde podemos realizar la vinculación con nuestra cuenta gitHub o crear una nueva.

Luego de vincular nuestra cuenta se presenta la ventana principal del sistema, desde aquí podremos realizar 3 acciones: Crear un nuevo repositorio, agregar un repositorio existente o clonar un repositorio.



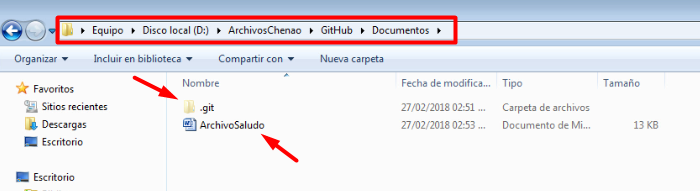
En nuestro caso vamos a clonar el repositorio creado en GitHub con anterioridad



En este proceso podemos definir cuál va a ser la ruta donde queremos que se almacenen nuestro repositorio y desde allí gestionar la información que queremos cargar a GitHub.

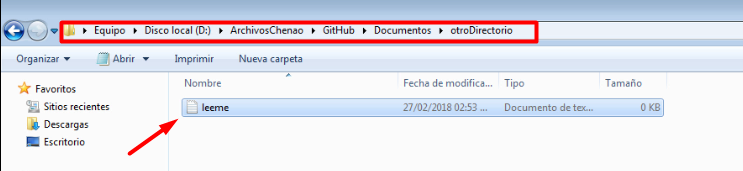
**GESTIONAR REPOSITORIO DE DOCUMENTACIÓN.**

Después de haber creado el repositorio remoto en GitHub y ya teniendo nuestro ambiente local mediante el cliente de escritorio ya podemos empezar a gestionar la información que queremos almacenar.

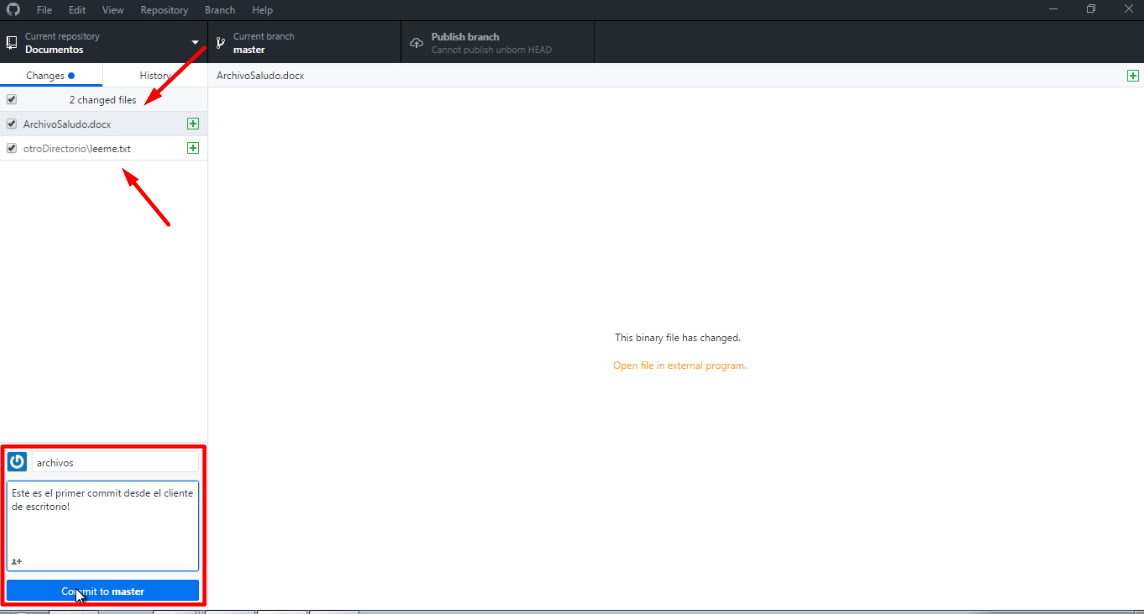


La ruta corresponde a la ruta que definimos al momento de clonar el repositorio, la carpeta .git nos indica que ese directorio está emparentado con el repositorio remoto y por ultimo tenemos un documento que vamos a gestionar.

Allí podemos agregar diferentes elementos o archivos a gestionar

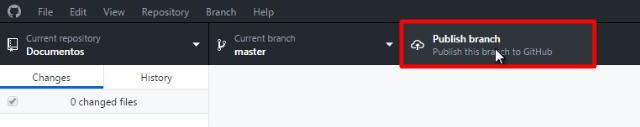


Después de esto regresamos al cliente GitHub que instalamos en nuestro equipo y veremos que ya se encuentran vinculados los archivos en la herramienta.

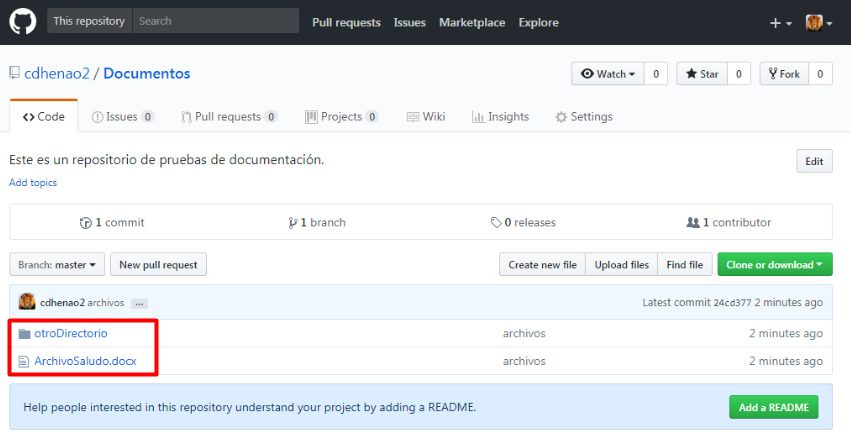


Luego de esto llenamos el recuadro inferior donde indicamos el mensaje del commit que realizaremos a nuestro repositorio local

Acto seguido presionamos la opción de **Publish branch**



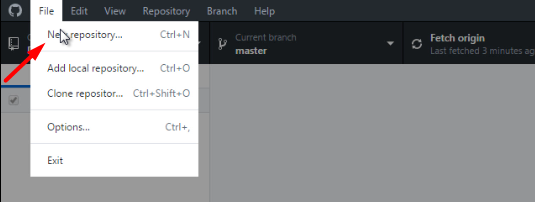
Después de esto buscamos en nuestra cuenta de gitHub el repositorio que vinculamos al cliente de escritorio y allí se verán alojados los archivos.



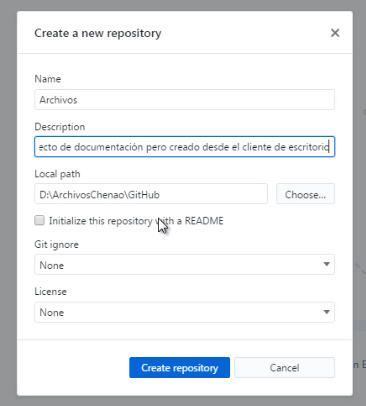
Si modificamos el contenido de los archivos automáticamente la herramienta identifica los cambios.

**CREAR NUEVO REPOSITORIO**

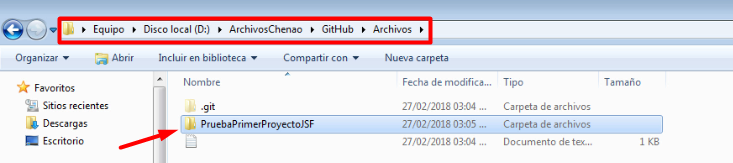
En el proceso anterior vinculamos a nuestro ambiente local un repositorio remoto que teníamos alojado en nuestra cuenta de GitHub, ahora vamos a hacer el proceso contrario, crearemos un repositorio local que será subido a nuestra cuenta GitHub, para eso vamos a file y lo creamos.



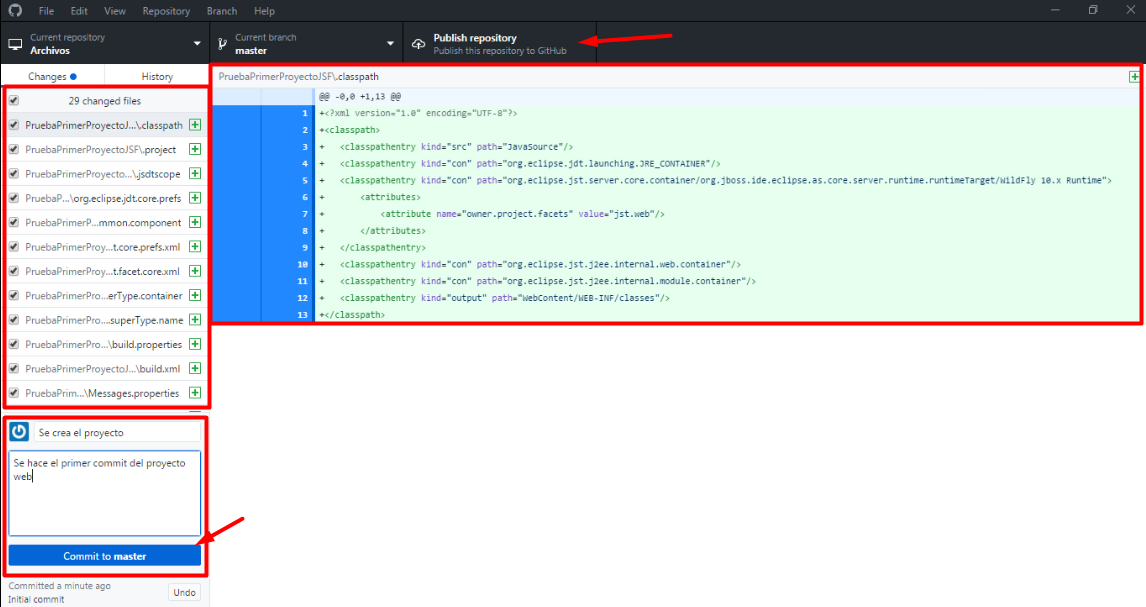
Al hacerlo se carga una ventana donde diligenciamos la información básica como nombre y descripción, así como las opciones de licencia de uso.



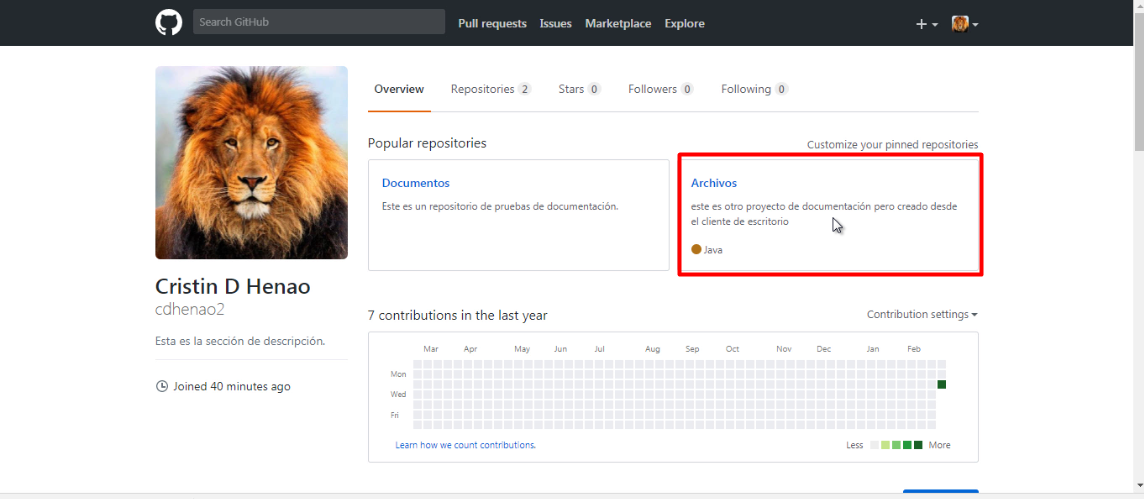
Al presionar el botón “Create repository” se genera una carpeta igual la que generamos al momento de clonar, allí podemos alojar diferentes archivos, en este caso vinculamos un proyecto web.



Cuando pasamos al cliente GitHub local veremos todos los archivos listos para subir.



Cuando verificamos el repositorio remoto ya tendremos nuestro proyecto vinculado.



**VINCULAR ECLIPSE CON GITHUB**

En los procesos anteriores trabajamos con el cliente GitHub de escritorio, como vimos este nos facilita bastante el proceso a la hora de vincular los elementos al entorno remoto pero ahora vamos a trabajar directamente desde eclipse para gestionar nuestros proyectos java.

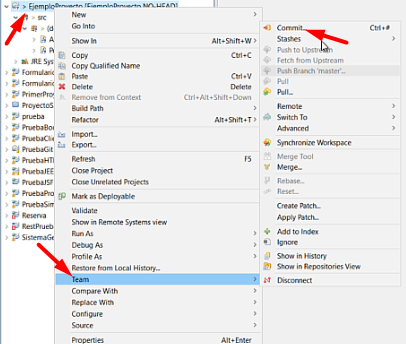
Inicialmente crearemos un nuevo repositorio, como convención podemos llamarlo con el mismo nombre del proyecto al que vamos a asociar.

|  |  |
| --- | --- |
|  |  |

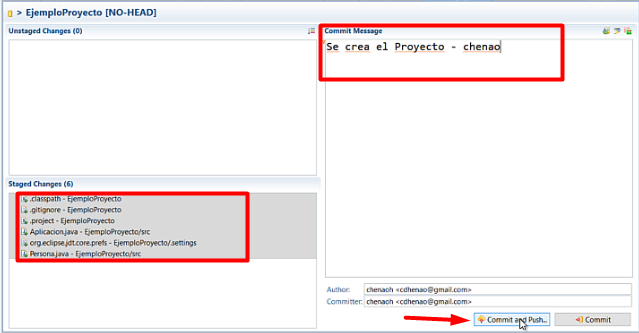
Después de haber creado el repositorio en gitHub vamos a eclipse y presionamos clic derecho en el proyecto a vincular, seleccionamos Team/Shared Project

|  |  |
| --- | --- |
|  | Al hacerlo se carga la ventana en la que definimos donde se almacenará nuestro repositorio local. |

Luego de esto nuevamente damos clic derecho al proyecto, seleccionamos Team/Commit

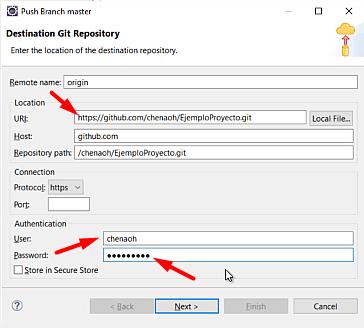


Al hacerlo se carga una nueva ventana donde seleccionamos los elementos que queremos vincular al repositorio y agregamos el mensaje del commit a enviar.

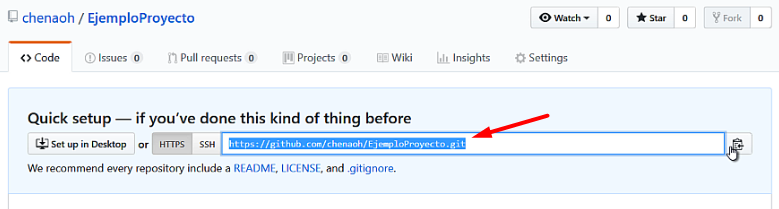


Presionamos el botón Commit and Push para vincular el proyecto y los cambios en nuestro repositorio Local y con el Push al repositorio remoto.

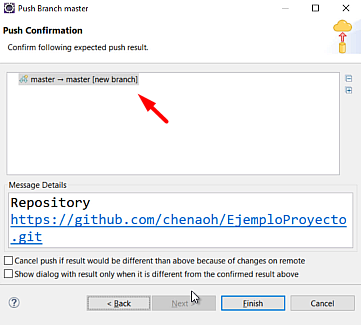
Al hacerlo se carga una ventana donde ingresamos la url del repositorio y las credenciales de ingreso



La url la obtenemos del repositorio remoto que creamos.



Al darle Next nos logueamos nuevamente y al final se presenta la ventana de confirmación indicando que ya ha sido cargado el proyecto en el repositorio.



Al revisar el repositorio encontramos el proyecto vinculado

