Datos Cuenta GIT

Password: m-oF__Mf2.44
Email: juanlarrea@outlook.de
name : yamek53

Instalacion

1. Instalar Git
2. Comandos en Terminal Git Bash

juanl@DESKTOP-FLC7NNL MINGW64 ~
$ config --global user.email juanlarrea@outlook.de

juanl@DESKTOP-FLC7NNL MINGW64 ~
$ git confi --global user.name “yamek53”

juanl@DESKTOP-FLC7NNL MINGW64 ~
$ git config --global core.editor "code --wait"

juanl@DESKTOP-FLC7NNL MINGW64 ~
$ git config --global -e

juanl@DESKTOP-FLC7NNL MINGW64 ~
$ git config --global core.autocrlf true

juanl@DESKTOP-FLC7NNL MINGW64 ~
$ git config -h

Flujo de trabajo GIT

Etapas del proceso (Computador >> Stage >> Commit >> Server):

1. Stage: Agregar archivos a la etapa Stage para Verificar los cambios que queremos seleccionar
2. Commit: Realizamos los cambios
3. Server: Pasar los archivos a la nube
4. Si queremos eliminar un archivo de nuestro computador tendremos que realizar la misma operacion para que quede registrado en la nube (servidor)

Cada vez que realicemos un cambio en un archivo pasaremos las modificaciones a la etapa Stage. No pasamos los archivos en si, si no las modificaciones.

Comandos básicos

ls para ver las carpetas y directorios
pwd ruta actual en mi pc
cd directorio para moverse entre directorios
cd .. para subir una posicion en el arbol
mkdir nombreCarpeta para crear un nuevodirectorio

.git el punto sirve para ocultar un archivo (Podemos mostrarlo en el explorador de archivos activando la casilla correspondiente)
git status muestra el estado actual del repositorio
git add agregar archivos a stage
git init Inicia el repositorio local

Agregar archivos a Stage

git add nombreArchivo.extension Agrega un archivo especifico
git add *.ext Agrega todos los archivos con esa extension
git add . Agrega todos los archivos del listado Untracked Files (Mala práctica)
git add archivo1.ext archivo2.ext Agrega varios archivos

Comprometer archivos (Commit)

Comprometemos los cambios del archivo una vez se han verificado en Stage. Hay dos maneras de hacerlo:

Git commit -m "comentario con las modificaciones realizadas"
git commit Se abre la ventana de edicion de commit en el editor y podemos comentar los cambios.

Finalmente tenemos que guardar y cerrar la ventana.

Eliminar archivos

rm nombreArchivo.ext Eliminar archivo

El archivo ha sido eliminado pero hay que agregar el cambio a la etapa Stage:

git add nombreArchivo.ext Archivo eliminado en la etapa de Stage
git commit -m "Comentario" Los cambios han sido comprometidos. Archivo eliminado

Restaurar archivos

git restore –-stage nombreArchivo.ext Restaura los cambios en la etapa de Stage si el archivo no fue eliminado
git restore nombreArchivo.ext Restaura el archivo eliminado en la etapa de stage

Mover o renombrar archivos

mv nombreArchivo.ext nuevoNombre.ext Para cambiar nombre de archivo
git add nombreArchivo.ext nuevoNombre.ext Para comprometer los cambios
git commit -m "Archivo renombrado"
git mv nombreArchivo.ext nuevoNombre.ext Agrega los cambios directamente a Stage

Ignorar archivos

Podemos ignorar archivos para que no sean incluidos nunca en nuestro repositorio. Estos podrian ser por ejemplo archivos de configuracion especificos para nuestra computadora. Por ejemplo, variables de entorno.

Creamos un archivo .env donde guardamos nuestras variables de entorno como por ejempo datos de acceso a un servidor:

Password = 123456
user = Juan

Creamos un archivo .gitignore y especificamos los archivo y las rutas que no quiero agregar a mi repositorio. Por ejemplo:

.env
node_modules/

Agregamos el archivo y lo comprometemos:

git add .gitignore
git commit -m "agregando archivo gitignore"

GIT Status mejorado

git status -s

Claves:

M Rojo Archivo modificado
M Verde Agregado a Stage
?? Rojo GIT no esta haciendo el seguimiento. El archivo no se ha agregado todavia a Stage.
A Verde El archivo ha sido agregado a Stage

Sie se encuentran en verde estan listos para comprometer (commit)

GIT git diff

git diff Para ver los cambios que vamos a realizar
En rojo se encuentra lo que habia antes de los cambios y en verde lo que se ha cambiado
@@ -1 +1,3 @@ Indicación de las lineas que se estan modificando

Letra "q" para salir

git diff --staged Para ver los cambios que se encuentran en la etapa de Stage

GIT git log

git log --oneline Muestra el historial de cambios con un identificador y el comentario

Letra "q" para salir

Ramas o branches

Podemos crear una rama o bifurcacion independiente de la principal para realizar modificaciones.

1. Verificar en que rama nos encontramos (main): **git branch**
2. Crear una rama: **git checkout -b rama-b**
3. Verificar en que rama nos encontramos (main): **git branch**
4. Podemos revisar el historia de cambios: **git log --online**
5. Vemos que se ha creado la rama en la que nos encontramos: **(HEAD -> rama-b)**
6. Para mostrar el contenido del archivo: **cat archivo.ext**
7. Para moverse a otra rama: **git checkout main**

Despues podemos agregar los cambios a la rama principal (main).

En la rama donde queremos incluir los cambios (main) escribimos **git merge** seguido de larama que queremos incorporar (rama-b)

git merge rama-b

De esta manera pueden estar trabajndo varias personas en sus correspondientes ramas a la vez y en un mismo proyecto, sin alterar la rama principal.

Conectar con GITHUB

1. Crear una cuenta
2. Crear un nuevo repositorio (boton new)
3. Configurar repositorio: Nombre de repositorio, Public, etc...
4. Conectar con el servidor remoto: **git remote add origin url del repositorio**
5. Subir archivos al servidor cuando la rama principal no esta creada: **git push -u origin main**
6. Subir archivos al servidor si la rama ya esta creada: **git push**
7. Introducir nombre de usuario de GITHUB
8. Introducir password: KEY de GITHUB
9. ENTER >> Se suben los archivos
10. Cuando queremos crear una rama nueva en el repositorio: **git push -u origin nombre Rama**

Conseguir KEY de Github

1. Perfil de Github
2. Settings
3. Developer settings
4. Personal Acces Token
5. Generate new token
6. Introducir nombre del token
7. Indicar tiempo expiracion del token

8. Indicar alcance del token (todas)
9. Generate token

Otros comandos

git remote [-v | --verbose]
git remote add [-t <branch>] [-m <master>] [-f] [--tags | --no-tags] [--mirror=<fetch|push>] <name> <url>
git remote rename [--[no-]progress] <old> <new>
git remote remove <name>
git remote set-head <name> (-a | --auto | -d | --delete | <branch>)
git remote [-v | --verbose] show [-n] <name>
git remote prune [-n | --dry-run] <name>
git remote [-v | --verbose] update [-p | --prune] [(<group> | <remote>)...]
git remote set-branches [--add] <name> <branch>...
git remote get-url [--push] [--all] <name>
git remote set-url [--push] <name> <newurl> [<oldurl>]
git remote set-url --delete <name> <url>
-v, --[no-]verbose be verbose; must be placed before a subcommand

git clone https://github.com/Yamek53/pruebas-odoo
`cd repositorio`

ssh-keygen -t ed25519 -C "juanlarrea@outlook.de"

Clave ssh

ssh-ed25519 AAAAC3NzaC1lZDI1NTE5AAAAIMdZwpAhlfKtijKHqlYkvPsbeo6jQajVd5+Gfxd2PgAl juanlarrea@outlook.de