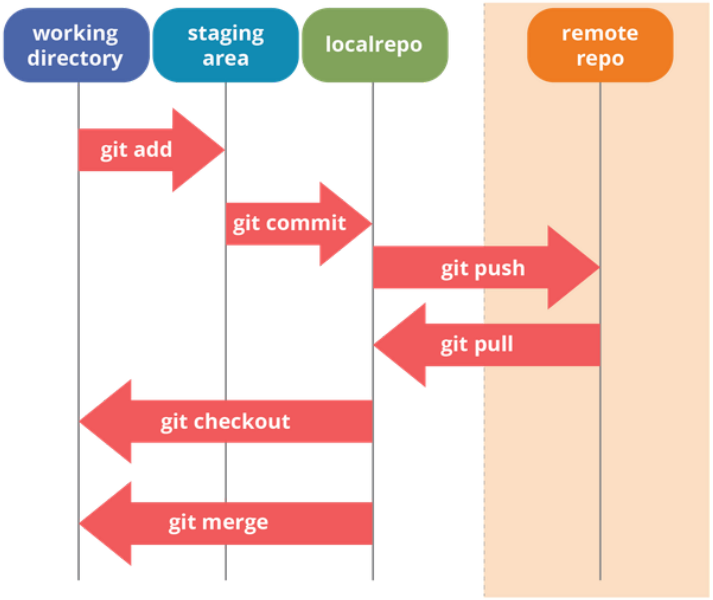
TP2_GIT



Abri una cuenta el gitlab o github.
Genera un proyecto tp_Python.
Instalarlo en tu pc git.
Haz una carpeta tp_Python en tu disco local o servidor local .
Coloca solo este archivo alli (Ahora, antes de modificarlo).
Inicializa git.
coloca tus datos.
Agrega este archivo.
Haz un commit con tu nombre como comentario.
Haz un push a tu servidor. Con el archivo antes de modificarlo.

Modifica este archivo con lo que se te pide debajo.

- ¿Que es git?
- ¿Que es github?
- ¿Que es gitlab?
- Diferencias
- ¿Como se usa y cual es su propósito general?
- ¿Quienes pueden usar sus datos?
- ¿Que proyectos colocarias en github y cuales en gitlab?¿Porque?
- ¿Cuales son sus principales comandos?¿ como se usan?
-

Guarda este archivo con los datos que agregaste.
Actualiza este archivo.
Haz un commit con tu nombre como comentario.
Haz un push a tu servidor. Con el archivo ya modificarlo.
Renombra este archivo.
Haz un pull y merge donde bajaras el archivo con el nombre viejo.

Elijan un proyecto en github o gitlab entre los alumnos.
Suban todos su aporte a ese proyecto y bajen el de los demas, asi pueden tener en local el trabajo de todos.
Editen un archivo con las mejores datos de todos los tp.
Subanlo y hagan un pull request.
Borren sus aportes personales en lo local y en sus proyectos y actualícelo con el archivo generado y mejorado por todo el grupo.

- ¿Que utilidad encontras git?
- ¿Que ventajas y desventajas tiene usar git como sistema de seguridad?
- ¿Que ventajas y desventajas tiene usar git como sistema de trabajo en grupo?

Nota: Ten en cuenta el espacio en disco
Cualquier consulta comuniquense a
cursos.agt@gmail.com o https://github.com/CursosAGT

Desarrollo

- ¿Que es git?
- Git es un sistema de control de versiones creado por Linus Torvalds (el creador del kernel de Linux). Git funciona como una forma de controlar los cambios en tu programa, proyecto o código en el tiempo. Además Git sirve para poder trabajar en equipo sobre un mismo programa de forma remota, así como también permite crear ramas paralelas a tu proyecto, para luego poder fusionarlas todas en su versión final.
- ¿Que es github?
- Github es una empresa que, mediante su página web, ofrece una plataforma de desarrollo colaborativo que usa el sistema de control de versiones de Git. Github ofrece los servicios de git de forma que es más fácil para un desarrollador usarlos, además de que permite el almacenamiento de los repositorios de git en la nube. También permite tener proyectos y repositorios privados, solo que estos requieren de una suscripción.
- ¿Que es gitlab?
- Gitlab (al igual que Github) es una plataforma de desarrollo colaborativo, desarrollada en github, que usa el sistema de controles de versiones de Git, y que te permite guardar repositorios en la nube.
- Diferencias
- Sin embargo Gitlab no es unicamente un clon de Github, ya que poseen varias diferencias. Entre ellas están:

-Github tiene un máximo de 3 colaboradores por repositorio, mientras gitlab te permite tener ilimitados.

-Mientras Github solo te da la opción de lectura o escritura en un repositorio, gitlab te permite proporcionar acceso a diferentes personas, a editar o leer unicamente ciertas partes del código.

-GitHub tiene una comunidad de usuarios y una cantidad de proyectos muy superior a GitLab. Mientras que en GitLab hay poco mas de 100.000 proyectos, en GitHub superan los 35.000.000.
- GitHub ofrece hasta 1 GB de espacio por repositorio de manera gratuita, además cada archivo que subamos al repositorio no debe pasar de 100 MB. En caso de superar estos límites, la plataforma nos avisará. Por otra parte, GitLab, ofrece una mayor capacidad de almacenamiento, siendo el límite máximo por repositorio de 10 GB.

- ¿Como se usa y cual es su propósito general?
-
- ¿Quienes pueden usar sus datos?
-
- ¿Que proyectos colocarías en github y cuales en gitlab?¿Porque?
    - Yo a github llevaría proyectos sencillos, cortos, pequeños y que no necesiten de muchas personas (como un libro por ejemplo), debido a las limitaciones previamente mencionadas en las diferencias.
    - Mientras tanto en gitlab haría software más complejo, que esté dividido en partes. Como por ejemplo un sistema operativo o un videojuego, ya que ambos están divididos en varías partes (como todo lo gráfico, las entradas y salidas de información, etc.). Esto es porque se puede aprovechar el sinfín de personas que pueden agregar y modificar para dividirse las tareas y hacer el trabajo de forma más eficiente y rápida. También es aprovechado el amplio espacio que proporciona en la nube, que ayuda a poder desarrollar proyectos más grandes.
- ¿Cuales son sus principales comandos?¿ como se usan?
- Los principales comandos de Git son :
- git help

Muestra una lista con los comandos más utilizados en GIT.

git init

Podemos ejecutar ese comando para crear localmente un repositorio con GIT y así utilizar todo el funcionamiento que GIT ofrece. Basta con estar ubicados dentro de la carpeta donde tenemos nuestro proyecto y ejecutar el comando. Cuando agreguemos archivos y un commit, se va a crear el branch master por defecto.

git add + path

Agrega al repositorio los archivos que indiquemos.

git add -A

Agrega al repositorio TODOS los archivos y carpetas que estén en nuestro proyecto, los cuales GIT no está siguiendo.

git commit -m "mensaje" + archivos

Hace commit a los archivos que indiquemos, de esta manera quedan guardados nuestras modificaciones.

git commit -am "mensaje"

Hace commit de los archivos que han sido modificados y GIT los está siguiendo.

git checkout -b NombreDeBranch

Crea un nuevo branch y automaticamente GIT se cambia al branch creado, clonando el branch desde donde ejecutamos el comando.

git branch

Nos muestra una lista de los branches que existen en nuestro repositorio.

git checkout NombreDeBranch

Sirve para moverse entre branches, en este caso vamos al branch que indicamos en el comando.

git merge NombreDeBranch

Hace un merge entre dos branches, en este caso la dirección del merge sería entre el branch que indiquemos en el comando, y el branch donde estémos ubicados.

git status

Nos indica el estado del repositorio, por ejemplo cuales están modificados, cuales no están siendo seguidos por GIT, entre otras características.

git clone URL/name.git NombreProyecto

Clona un proyecto de git en la carpeta NombreProyecto.

git push origin NombreDeBranch

Luego de que hicimos un git commit, si estamos trabajando remotamente, este comando va a subir los archivos al repositorio remoto, específicamente al branch que indiquemos.

git pull origin NombreDeBranch

Hace una actualización en nuestro branch local, desde un branch remoto que indicamos en el comando.