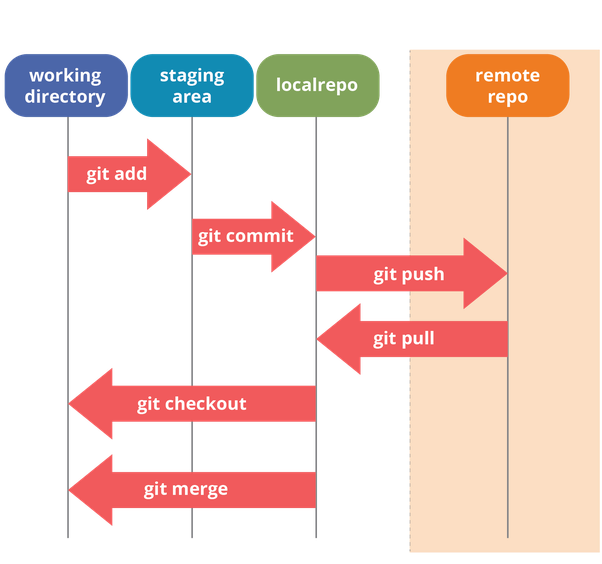
TP2\_GIT



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? Git es software de control de versiones que maneja la organización de las distintas versiones de una misma aplicación cuando tienen varios archivos de código fuente.
* ¿Que es github? GitHub es una plataforma colaborativa que sirve para guardar y desarrollar proyectos de software editando el código fuente.
* ¿Que es gitlab? GitLab es similar a GitHub, con la adición de herramientas como seguimiento de error de código y wikis.
* Diferencias  
  GitLab, a diferencia de GitHub, permite hasta 3 colaboradores en un repositorio. Además de esto, GitHub da la opción de leer o escribir en un repositorio, mientras que GitLab permite dar acceso a partes específicas de un proyecto. GitLab también permite clasificar un proyecto como “en desarrollo” y permite añadir adjuntos en un problema. GitHub además permite hasta 1GB de espacio por repositorio, junto con un límite de 100MB por archivo, mientras que GitLab permite hasta 10GB.
* ¿Que proyectos colocarias en github y cuales en gitlab?¿Porque?  
  GitLab parece tener una mayor utilidad para proyectos más complejos y desarrollados, en los que varia gente está trabajando, y en las que se requieren varias ramas de desarrollo al mismo tiempo, con mayor división de tareas. GitHub, al ofrecer menor acceso de capacidad y colaboradores, se ve mejor destinado a proyectos más pequeños, o a borradores y pruebas en menor escala de proyectos mayores.
* ¿Cuales son sus principales comandos?¿ como se usan?  
  **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.

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.