Procedimiento para el Control de Versiones Git-lab

EL control de versiones GitLab es una plataforma de colaboración y comunicación.

Se prevé que la implementación de GitLab para la elaboración colaborativa de los sistemas de información y aplicaciones web del MPPEUCT, permitirá mejorar los procesos de trabajo y el producto final. En este sentido, la herramienta permitirá obtener un producto de calidad, enriquecido por la participación eficaz de múltiples actores. El modo de funcionamiento a través de un documento distribuido y almacenado en repositorios con acceso al que van a intervenir en su elaboración, facilitará obtener la trazabilidad completa del código fuente, desde el inicio del proyecto hasta su versión final.

La intervención sobre el mismo, pasa de ser secuencial mediada por una lógica de revisión y corrección a través del control de cambios, a ser una intervención colaborativa a través de la producción de múltiples versiones de un mismo proyecto. Todos los usuarios que serán parte de este control de versiones podrán acceder a los cambios, para visualizar datos, quién intervino, en qué momento y qué lugar. Esto permite trabajar en un mismo archivo a distancia, en momentos simultáneos o distintos. Los aportes de cada desarrollador quedan plasmados en versiones diferentes. Nada se elimina ni se pierde. Todos los datos, la información y las ideas pueden reutilizarse como un disparador para un nuevo aporte.

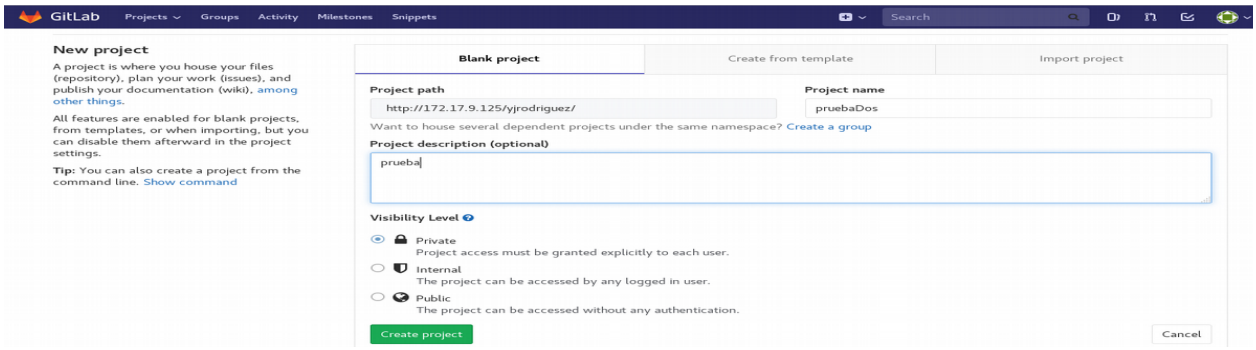
Para hacer uso del **control de versiones GitLab**, los usuarios deberán tener instalado en sus maquinas git, siguiendo los siguientes pasos:

- ✔ Comando para instalar **git aptitude install git**.
- ✔ Verificar que tenga instalado **SSH**.
- ✔ Cada usuario del sistema ha de generar su clave o llave pública SSH, en su maquinas de trabajo para la autenticación del mismo.
- ✔ Verificar que no tengas una llave pública ya creada, aplicando lo siguiente:

```
yurodriguez@mcti-v082-i201:/home$ cd ~/.ssh
yurodriguez@mcti-v082-i201:~/.ssh$ ls
id_rsa  id_rsa.pub
```

En caso de no poseer llave pública ejecutar el siguiente comando para generarla ssh-keygen en la carpeta **.ssh**

- ✔ Deberán enviar a la coordinación de seguridad de informática el listados de los usuarios que tendrán acceso al control de versiones, los mismos deberán indicar su llave públicas generadas en su maquina de trabajo.
- ✔ Una vez creado el usuario estos podrán hacer uso del mismo, la coordinación automatización de procesos se encargara de crear sus repositorio (proyecto) mediante la interfaz gráfica del control de versiones, estos proyectos deberán ser creado de manera privada con la finalidad de darles los permisos correspondiente a los usuarios que podrán colaborar en el mismo.



- ✔ Cada repositorio (proyecto) se manejara por ramas esto simplemente es un apuntador móvil apuntando a una de esas confirmaciones, por defecto cuando se crea un repositorio (proyecto), git genera una rama por defecto llamada máster, el cual sera usada por la coordinación de seguridad de informática.
- ✔ Dentro de los repositorios (proyectos) se crearan varias ramas quedando de esta manera:
- ◆ Rama usuario: Esta rama es la que usara cada colaborador (desarrollador) en el repositorio (proyecto), asignado por el coordinación de automatización de procesos.
- ◆ Rama desarrollo: Esta rama es la usada por la coordinación de automatización de procesos donde unificara la rama de cada colaborador (programador), que ya se encuentre estable
- ◆ Rama calidad (pre-producción): Esta rama es la usada por la coordinación de seguridad de informática, donde unificara la rama ya estable de desarrollo, para validar que el proyecto este estable en el ambiente de pre-produccion.
- ◆ Rama máster : Esta sera usada por la coordinación de seguridad de informática, donde se hará una unificación con calidad (pre-producción), una vez que el mismo se encuentre estable y este validado por parte de la coordinación antes mencionada y la coordinación de proyectos.

La coordinación de automatización de procesos una vez creado su repositorio (proyecto) mediante la interfaz gráfica, podrá configurar su usuario git de la siguiente manera:

```
git config --global user.name "Yurasmy Rodriguez"
git config --global user.email "yjrodriguez@mppeuct.gob.ve"
```

ejecutar el siguiente comando desde consola para clonar el proyecto, hacerlo con su usuario, no usar root en este caso.

```
yurodriguez@mcti-v082-i201:/var/www/html$ git clone git@gitlab.mppeuct.gob.ve:yjrodriguez/pruebaDos.git
```

Ya clonado el repositorio (proyecto) , se debe ingresar a la carpeta **cd pruebaDos/**, se se crea un archivo dentro de la misma.

```
yurodriguez@mcti-v082-i201:/var/www/html/pruebaDos$ touch README.md
```

Se agrega el archivo y se realiza un comentario.

```
yurodriguez@mcti-v082-i201:/var/www/html/pruebaDos$ git add README.md
yurodriguez@mcti-v082-i201:/var/www/html/pruebaDos$ git commit -m "add README"
```

Se sube a la rama máster el archivo que se crea.

```
yurodriguez@mcti-v082-i201:/var/www/html/pruebaDos$ git push -u origin master
Contando objetos: 3, listo.
Escribiendo objetos: 100% (3/3), 219 bytes | 219.00 KiB/s, listo.
Total 3 (delta 0), reused 0 (delta 0)
To 172.17.9.125:yjrodriguez/pruebaDos.git
 * [new branch]      master -> master
Rama 'master' configurada para hacer seguimiento a la rama remota 'master' de '
yurodriguez@mcti-v082-i201:/var/www/html/pruebaDos$ git branch
* master
```

Otra forma de agregar git a un repositorio (proyecto), ya creado en tu local.

```
cd existing_folder
git init
git remote add origin git@gitlab.mppeuct.gob.ve:yjrodriguez/pruebaDos.git
git add .
git commit -m "Initial commit"
git push -u origin master
```

La coordinación de automatización de procesos creara su rama de desarrollo en su repositorio (proyecto) y las ramas necesaria de los desarrolladores que colaboraran dentro del mismo.

```
yurodriguez@mcti-v082-i201:/var/www/html/pruebaDos$ git checkout -b desarrollo
Cambiado a nueva rama 'desarrollo'
yurodriguez@mcti-v082-i201:/var/www/html/pruebaDos$ git branch
* desarrollo
  master
yurodriguez@mcti-v082-i201:/var/www/html/pruebaDos$ git push -u origin desarrollo
Total 0 (delta 0), reused 0 (delta 0)
remote:
remote: To create a merge request for desarrollo, visit:
remote:   http://gitlab.mppeuct.gob.ve/yjrodriguez/pruebaDos/merge_requests/new?merge_request%5Bsource_branch%5D=desarrollo
remote:
To 172.17.9.125:yjrodriguez/pruebaDos.git
 * [new branch]      desarrollo -> desarrollo
Rama 'desarrollo' configurada para hacer seguimiento a la rama remota 'desarrollo' de 'origin'.
```

```
yurodriguez@mcti-v082-i201:/var/www/html/pruebaDos$ git checkout -b jurbina
Cambiado a nueva rama 'jurbina'
yurodriguez@mcti-v082-i201:/var/www/html/pruebaDos$ git push -u origin jurbina
Total 0 (delta 0), reused 0 (delta 0)
remote:
remote: To create a merge request for jurbina, visit:
remote:   http://gitlab.mppeuct.gob.ve/yjrodriguez/pruebaDos/merge_requests/new?merge_request%5Bsource_branch%5D=jurbina
remote:
To 172.17.9.125:yjrodriguez/pruebaDos.git
 * [new branch]      jurbina -> jurbina
Rama 'jurbina' configurada para hacer seguimiento a la rama remota 'jurbina' de 'origin'.
```

La coordinación de automatización de procesos hará la unificación de las ramas de sus colaboradores (desarrolladores) en su rama principal (desarrollo); una vez estable el proyecto, se procede a actualizar la rama de calidad(pre-producción), con la finalidad de hacerle las pruebas al proyecto.

```
yurodriguez@mcti-v082-i201:/var/www/html/pruebaDos$ git branch
* calidad
  desarrollo
  jurbina
  master
  mzalazar
yurodriguez@mcti-v082-i201:/var/www/html/pruebaDos$ git merge desarrollo
Ya está actualizado.
yurodriguez@mcti-v082-i201:/var/www/html/pruebaDos$ git push origin calidad
Everything up-to-date
```

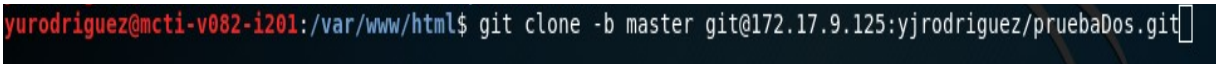
La coordinación de servidores se encargara únicamente de clonar el proyecto en el servidor de calidad (pre-producción) realizando el siguiente comando.

```
yurodriguez@mcti-v082-i201:/var/www/html$ git clone -b calidad git@172.17.9.125:yjrodriguez/pruebaDos.git
```

Luego la coordinación de seguridad de la información procede a realizar los análisis de vulnerabilidades como (caja negra, caja blanca), conjuntamente con la coordinación de proyectos quien evaluara la funcionalidad del aplicativo.

Una vez el proyecto ya estable, en el servidor de calidad (pre-producción), la coordinación de seguridad se encara de actualizar la rama máster (producción);

Indicándole a la coordinación de servidores que pueden proceder a clonar el proyecto en el servidor de de producción.



En caso de que el proyecto ya se encuentre en el servidor de producción y se necesita hacer una modificación ya antes realizada y validada por la coordinaciones pertinentes, la coordinación de servidores solo ejecutara el siguiente comando desde el proyecto