

Título

Manual Técnico

Breve Descripción

Manual técnico para la herramienta GITLAB dedicada al control de versiones en los distintos proyectos.

Propósito

La creación de este manual se realiza con el propósito de dar a conocer a detalle el proceso de instalación y configuración de la herramienta, así como estándares e infraestructura para el uso adecuado.

Elaborado por

IABJ

Última revisión

22/03/2021

Referencias
Versión

1.0

Válido

De: *Agosto/2020*

A: *Diciembre/2021*
Sustituye al Documento
Estado
☐ ☐ Draft

☐ ☒ Definitive

Lista de Distribución

SC Manager	RGA
Scrum Master	COMS
Software Project Manager	COMS
Technical Leader	MORR
SCM Group	BEGI, HEOX
SQA Group	RURL, TOBF
Software Engineering Group	MORR, MALZ
Test Group	ANBM, SARE, RITL, LORO
Data Analyst	MROM

Autorización
Elaborado por
IABJ
Revisado por
SAXA
**Revisado por /
Autorizado por**

1. INTRODUCCIÓN	3
1.1 GITLAB	3
2. REQUERIMIENTOS	5
3. INSTALACIÓN DE GITLAB	6
3.1 INSTALACIÓN Y CONFIGURACIÓN LAS DEPENDENCIAS NECESARIAS.....	6
3.2 AGREGAR EL REPOSITORIO DE PAQUETES DE GITLAB E INSTALE EL PAQUETE	7
3.3 BUSQUE EL NOMBRE DE HOST E INICIE SESIÓN	8
4. ACTUALIZAR VERSIÓN DE GITLAB	9
5. CONFIGURACIÓN DE GITLAB.....	9
6. BACKUP	14

1. INTRODUCCIÓN

En el presente documento se dan a conocer las instrucciones para el uso adecuado de la herramienta GitLab, la cual es requerida de alojamiento de repositorios con varias funciones de seguimientos de problemas de proyectos en Praxis.

El manual presenta una breve descripción acerca de GitLab, posteriormente se explican las áreas que lo componen, así como una serie de comandos más comunes a utilizar, después de ello se desglosan las funciones y su uso correspondiente.

Este manual está basado en la versión de Git 13.1

1.1 GITLAB

GitLab es una plataforma basada en Git que integra una gran cantidad de herramientas esenciales para el desarrollo y la implementación de software y la gestión de proyectos:

- Alojamiento de código en repositorios con control de versiones.
- Seguimiento de propuestas para nuevas implementaciones, informes de errores y comentarios con un rastreador de problemas con todas las funciones .
- Organizar y priorizar con Juntas Temáticas .
- Revisión de código en solicitudes de combinación con cambios de vista previa en vivo por rama con aplicaciones de revisión .
- Compilación, prueba e implementación con integración continua incorporada .
- Despliegue de sitios web estáticos personales y profesionales con GitLab Pages .
- Integración con Docker mediante GitLab Container Registry .
- Seguimiento del ciclo de vida del desarrollo mediante GitLab Value Stream Analytics .
- Brindar soporte con Service Desk .

Con GitLab Enterprise Edition, también puede:

- Mejorar la colaboración con:
 - Fusionar aprobaciones de solicitudes.
 - Múltiples cesionarios para problemas.
 - Múltiples foros de emisión.
- Cree relaciones formales entre problemas con problemas relacionados .
- Utilice Burndown Charts para realizar un seguimiento del progreso durante un sprint o mientras trabaja en una nueva versión de su software.
- Aproveche Elasticsearch con Advanced Global Search y Advanced Syntax Search para una búsqueda de código más rápida y avanzada en toda su instancia de GitLab.
- Autentica usuarios con Kerberos.
- Refleje un repositorio de otro lugar en su servidor local.
- Exportar problemas como CSV.
- Vea toda su canalización de CI / CD que involucra más de un proyecto con canalizaciones de múltiples proyectos .
- Bloquea archivos para evitar conflictos.
- Vea el estado actual y el estado de cada entorno de CI que se ejecuta en Kubernetes con Deploy Boards .
- Aproveche el método de entrega continua con Canary Deployments .
- Escanee su código en busca de vulnerabilidades y muéstrelas en solicitudes de combinación .

También puede integrar GitLab con numerosas aplicaciones de terceros, como Mattermost, Microsoft Teams, HipChat, Trello, Slack, Bamboo CI, Jira y muchas más.

2. REQUERIMIENTOS

Hardware

Los requisitos de la CPU dependen del número de usuarios y de la carga de trabajo esperada. Su carga de trabajo está influenciada por factores como entre otros, qué tan activos son sus usuarios, cuánta automatización usa, duplicación y tamaño de repositorio / cambio.

La siguiente es la guía de hardware de CPU mínima recomendada para un conjunto de tamaños de base de usuario de GitLab de ejemplo.

- 4 núcleos es el número mínimo recomendado de núcleos y admite hasta 500 usuarios.

La siguiente es la guía de hardware de memoria mínima recomendada para un conjunto de tamaños de base de usuario de GitLab de ejemplo.

- 4 GB de RAM es el tamaño de memoria mínimo requerido y admite hasta 500 usuarios

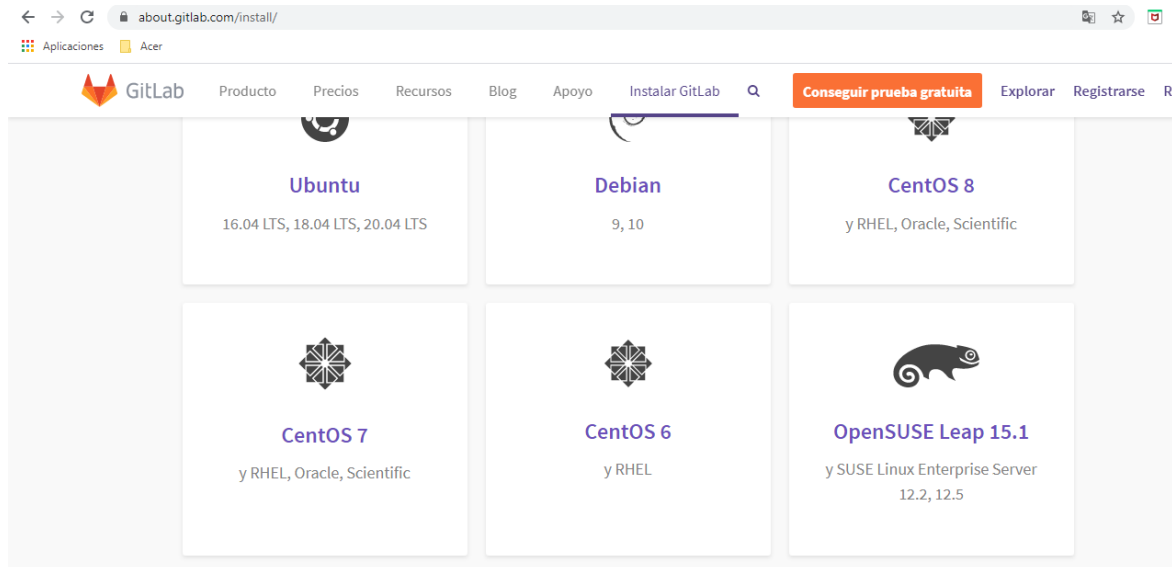
Software

- A partir de GitLab 12.2 se requiere Ruby (MRI) 2.6 en adelante.
- La versión mínima requerida de Go es 1.13
- GitLab usa webpack para compilar activos de frontend, lo que requiere una versión mínima de Node.js 10.13.0.

3. INSTALACIÓN DE GITLAB

Para instalar la herramienta tenemos que acceder a la página de descargas de GitLab y seleccionar el sistema operativo. En nuestro caso CentOS 7.

Link: <https://about.gitlab.com/install/>



Seleccionando el sistema operativo, podremos ver las instrucciones a seguir para la instalación de la herramienta, se solicita el username y password por defecto para poder acceder.

3.1 INSTALACIÓN Y CONFIGURACIÓN LAS DEPENDENCIAS NECESARIAS

En CentOS 7 (y RedHat / Oracle / Scientific Linux 7), los siguientes comandos también abrirán el acceso HTTP, HTTPS y SSH en el firewall del sistema.

```
sudo yum install -y curl policycoreutils-python openssh-server
```

```
sudo systemctl enable sshd
```

```
sudo systemctl start sshd
```

```
sudo firewall-cmd --permanent --add-service=http
```

```
sudo firewall-cmd --permanent --add-service=https
```

```
sudo systemctl reload firewalld
```

A continuación se instalará Postfix para enviar correos electrónicos de notificación. Si desea utilizar otra solución para enviar correos electrónicos omita este paso y configure un servidor SMTP externo después de que se haya instalado GitLab.

```
sudo yum install postfix  
sudo systemctl enable postfix  
sudo systemctl start postfix
```

Durante la instalación de Postfix aparecerá una pantalla de configuración, seleccione 'Sitio de Internet' y presione Intro. Use el DNS externo de su servidor para 'nombre de correo' y presione enter. Si aparecen pantallas adicionales, continúe presionando enter para aceptar los valores predeterminados.

3.2 AGREGAR EL REPOSITORIO DE PAQUETES DE GITLAB E INSTALE EL PAQUETE

Para agregar el repositorio de paquetes de GitLab.

```
curl https://packages.gitlab.com/install/repositories/gitlab/gitlab-  
ee/script.rpm.sh | sudo bash
```

A continuación, se debe añadir el paquete GitLab. Cambie `https://gitlab.example.com` a la URL en la que desea acceder a su instancia de GitLab. La instalación configurará e iniciará GitLab automáticamente en esa URL.

Para las URL's `https`, GitLab solicitará automáticamente un certificado con Let's Encrypt que requiere acceso HTTP entrante y un nombre de host válido. También puede usar su propio certificado o simplemente usar `http` `//`.

```
sudo EXTERNAL_URL="https://gitlab.example.com" yum install -y gitlab-ee
```

3.3 BUSQUE EL NOMBRE DE HOST E INICIE SESIÓN

En la primera visita será redirigido a una pantalla de restablecimiento de contraseña. Proporcione la contraseña para la cuenta de administrador inicial y será redirigido a la pantalla de inicio de sesión. Utilice el nombre de usuario de la cuenta predeterminada “root” para iniciar sesión.

Setup new password

Please set a new password before proceeding.

After a successful password update you will be redirected to login screen.

Password

Password
confirmation

Set new password

GitLab

GitLab is open source software to collaborate on code.
Sign in or browse for [public projects](#).

Sign in

root

☐ Remember me

Sign in

[Forgot your password?](#)

Did not receive confirmation email? [Send again](#)

4. ACTUALIZAR VERSIÓN DE GITLAB

Si desea hacer una copia de seguridad antes de actualizar, el siguiente comando genera una copia de seguridad de los datos “/var/opt/gitlab/backups” de forma predeterminada.

```
sudo gitlab-rake gitlab:backup:create STRATEGY=copy
```

Actualice a la última versión de GitLab (13.1)

```
sudo dnf install -y gitlab-ce
```

5. CONFIGURACIÓN DE GITLAB

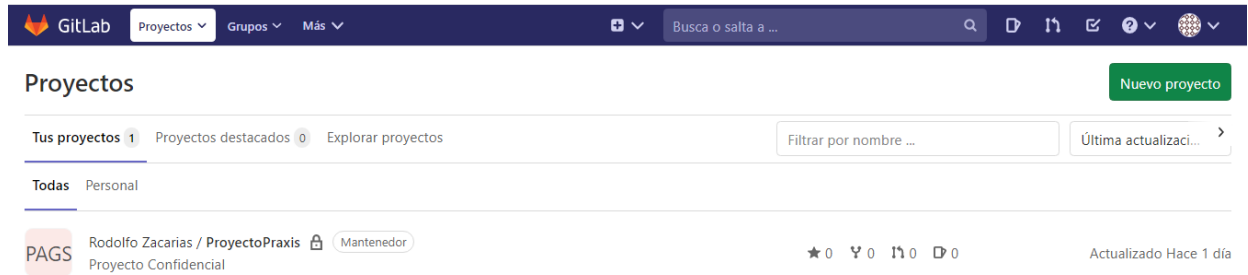
Lo primero que se tiene que hacer en GitLab es crear un nuevo proyecto. Esto lo consigues presionando el icono “+” en la barra superior. Se necesita poner el nombre del proyecto, el espacio de nombres al que pertenece y qué nivel de visibilidad debe tener. Una vez que tenga el proyecto, querrá usarlo para un repositorio local de Git. Cada proyecto se puede acceder por HTTPS o SSH, protocolos que podemos configurar en nuestro repositorio como un Git remoto. La URL la encontrará al principio de la página principal del proyecto. Para un repositorio local existente, puede crear un remoto llamado gitlab del siguiente modo:

```
$ git remote add gitlab https://server/namespace/project.gitgit config --global
```

Si no tiene copia local del repositorio, puede hacer esto:

```
$ git clone https://server/namespace/project.git
```

Una vez introducida la nueva password el sistema nos redirige nuevamente a la página de login dónde se tendrá que acceder con las nuevas credenciales.



Para trabajar en un proyecto GitLab puede añadir usuarios al proyecto en la sección usuarios de los ajustes del mismo, y asociar el usuario con un nivel de acceso.

A continuación se tendrá que rellenar el formulario con la información del usuario y éste recibirá un email con la información para acceder y una password temporal que el sistema le obligará a cambiar la primera vez que haga login en la herramienta.

Cualquier nivel de acceso tipo “Developer” o superior, permite al usuario enviar commits y ramas sin ninguna limitación.

New user

Account

Name	<input type="text" value="Rubén Aguilera"/>
	* required
Username	<input type="text" value="raguilera82"/>
	* required
Email	<input type="text" value="raguilera@autentia.com"/>
	* required

Password

Password A temporary password will be generated and sent to user.
User will be forced to change it after first sign in

Access

Projects limit	<input type="text" value="10"/>
Can create group	<input checked="" type="checkbox"/>
Admin	<input checked="" type="checkbox"/>

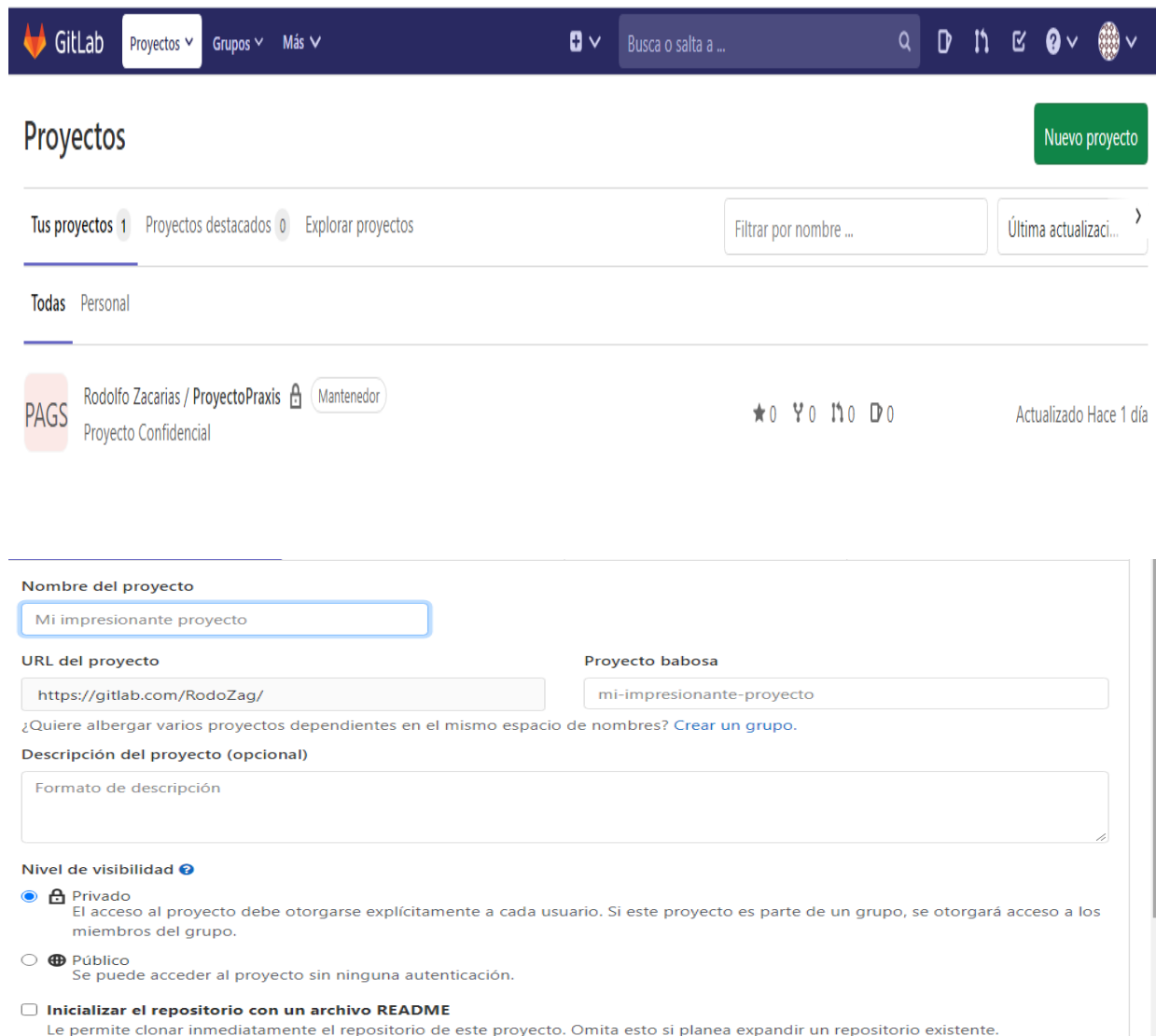
Profile

Avatar	<input type="button" value="Choose File"/> No file chosen
Skype	<input type="text"/>
Linkedin	<input type="text"/>
Twitter	<input type="text" value="@raguilera82"/>
Website	<input type="text" value="http://www.autentia.com"/>

Create user

Cancel

A continuación, para crear el primer proyecto presione el botón «New Project», donde nos solicita información como el nombre del proyecto, una descripción y de qué tipo va a ser el proyecto.



The screenshot shows the GitLab web interface for creating a new project. At the top, the GitLab logo and navigation tabs (Proyectos, Grupos, Más) are visible. A search bar and utility icons are on the right. The main heading is 'Proyectos', with a green 'Nuevo proyecto' button. Below this, there are filters for 'Tus proyectos' (1), 'Proyectos destacados' (0), and 'Explorar proyectos'. A search bar 'Filtrar por nombre ...' and a 'Última actualizaci...' dropdown are also present. Under the 'Todas' tab, a project entry for 'Rodolfo Zacarias / ProyectoPraxis' is shown, marked as 'Mantenedor' and 'Proyecto Confidencial'. It has 0 stars, 0 forks, 0 issues, and 0 discussions, and was updated 'Hace 1 día'. The form below contains the following fields and options:

- Nombre del proyecto:** A text input field containing 'Mi impresionante proyecto'.
- URL del proyecto:** A text input field containing 'https://gitlab.com/RodoZag/'.
- Proyecto babosa:** A text input field containing 'mi-impresionante-proyecto'.
- Descripción del proyecto (opcional):** A large text area with the placeholder 'Formato de descripción'.
- Nivel de visibilidad:** Two radio button options:
 - ☒ **Privado**: El acceso al proyecto debe otorgarse explícitamente a cada usuario. Si este proyecto es parte de un grupo, se otorgará acceso a los miembros del grupo.
 - ☐ **Público**: Se puede acceder al proyecto sin ninguna autenticación.
- ☐ **Inicializar el repositorio con un archivo README**: Le permite clonar inmediatamente el repositorio de este proyecto. Omita esto si planea expandir un repositorio existente.

Al presionar el botón «Create», la herramienta crea el proyecto y nos mostrará una serie de instrucciones.

SSH HTTP git@localhost:raguilera82/autentia-test.git

Este proyecto es para hacer la prueba de concepto de GitLab – [Edit](#)

Git global setup:

```
git config --global user.name "Rubén Aguilera"
git config --global user.email "raguilera@autentia.com"
```

Create Repository

```
mkdir autentia-test
cd autentia-test
git init
touch README
git add README
git commit -m 'first commit'
git remote add origin git@localhost:raguilera82/autentia-test.git
git push -u origin master
```

Existing Git Repo?

```
cd existing_git_repo
git remote add origin git@localhost:raguilera82/autentia-test.git
git push -u origin master
```

Remove project

Se mostrará un aviso que el usuario no podrá acceder al repositorio hasta que no añada su SSH Key pública para poder conectar con el repositorio sin necesidad de poner la password, se necesitará abrir la terminal y por lo siguiente:

```
$> ssh-keygen -t rsa
$> cat ~/.ssh/id_rsa.pub
```

Entonces en la herramienta, podemos presionar en el enlace que nos muestra o desde la opción de menú «Profile Settings» en la pestaña «SSH Keys», se debe pulsar en el botón «Add SSH Key» donde le damos un nombre a la clave y pegamos la clave ssh pública que es todo el contenido que se muestra en pantalla al hacer el cat.

Add an SSH Key

Paste your public key here. Read more about how to generate a key on the [SSH help page](#).

Title	<input type="text" value="SSH Key para GitLab"/>
Key	<pre>ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAQDQO+UCODY20Rk0oz8QESWzGBkfjYrgwP7pj/cx6SiqJ ocoBuinf0l3DYepTdhHyr5WcN5yCzTkPPh+hoqlzJxXal0fl6ufGoJQ3RzmvPJks6xk/Us1tKoNF6fd75 1rsTJjyFxxSjV+ u+w8MwkZdnwOwS0HrHAU2zgLgqI3hRnfreMo5wuTJU7MqYhrNxVuiVGsno0Nk7mESznsMeMkY wwWG6M7real52fKlwa/ygON2sl2qws7XFXqRNkQdLxqJ4rE7SHpEoXYa6adV8GkqEbfP9F raguilera82@xxxxx</pre>
<div><input type="button" value="Add key"/> <input type="button" value="Cancel"/></div>	

Para probar que todo ha ido bien se necesita abrir la terminal e intentar hacer un clone del proyecto.

****Ejemplo****

```
git clone git@ip_maquina_centos:raguilera82/autentia-test.git
```

Si la terminal no solicita que se introduzca una contraseña significa que la configuración SSH se ha realizado exitosamente. Ahora puede interactuar con el repositorio de la forma habitual.

6. BACKUP

Primero se tendrá que hacer los cambios de configuración necesarios para hacer backup para ello se tendrá que editar el fichero `/etc/gitlab/gitlab.rb`

```
$> sudo nano /etc/gitlab/gitlab.rb
```

Dentro de este fichero con CTRL + W buscar la palabra Backup Settings que nos mandará directamente a la sección. En esta sección se podrá observar dónde se almacenarán localmente los backups generados en la propiedad «backup_path». Por defecto, en «/var/opt/gitlab/backups»

En la propiedad «backup_keep_time» se puede definir en segundos el tiempo de vida de los ficheros de backup a fin de no llenar el disco de la máquina. Por defecto este valor se establece a 604800 segundos que son 7 días. Si se necesita que el fichero de backup se almacene automáticamente en un bucket S3 de AWS solo tiene que quitar los comentarios, las líneas que se refieren a la propiedad «backup_upload_connection» y establecer el nombre del bucket en la propiedad «backup_upload_remote_directory».

Para que los cambios de configuración tengan efecto, se tiene que ejecutar:

\$> sudo gitlab-ctl reconfigure

Hecho esto en cualquier momento que se requiera generar un fichero de backup solo se tendrá que ejecutar el siguiente comando:

\$> sudo gitlab-rake gitlab:backup:create

Al ejecutar este comando se generará el archivo y se tendrá que comprobar que se almacene en el bucket de AWS previamente configurado.

Nota: Por razones de seguridad los ficheros: /etc/gitlab/gitlab.rb y /etc/gitlab/gitlab-secrets.json no se incluyen dentro del fichero generado y tienen que ser guardados de forma independiente.

Ahora si lo que se necesita es que el backup se realice de forma automática todos los días a las 2 AM tenemos que crear un cron siguiendo estos pasos:

\$> sudo su -

\$> crontab -e

Añadir la siguiente línea:

```
0 2 * * * /opt/gitlab/bin/gitlab-rake gitlab:backup:create CRON=1
```

Una vez que se tenga guardado el fichero con el backup, se necesita ver cómo poder utilizarlo para restaurar la instancia en caso de que sea necesario.

Lo primero que necesita es que GitLab esté corriendo y al menos se haya ejecutado una vez un reconfigure.

```
$> sudo gitlab-ctl reconfigure
```

Nota: En caso de no estar corriendo no se puede restaurar el backup, así que al menos se necesitará una instancia limpia con la misma versión de GitLab corriendo.

El siguiente paso sería copiar el fichero de backup deseado en la ruta de la propiedad `backup_path` que por defecto es `/var/opt/gitlab/backups/` ahora se necesita parar los servicios «unicorn» y «sidekiq» únicamente, el resto de servicios tienen que estar corriendo.

```
$> sudo gitlab-ctl stop unicorn
```

```
$> sudo gitlab-ctl stop sidekiq
```

Para poder comprobar que solo estos servicios están parados se necesita ejecutar lo siguiente:

```
$> sudo gitlab-ctl status
```

Ahora ejecutamos el comando de restauración indicando el timestamp del fichero de backup.

```
$> sudo gitlab-rake gitlab:backup:restore  
BACKUP=1534804928_2018_08_21_11.1.4_gitlab
```

Hecho esto de forma satisfactoria es el momento de restaurar los ficheros `/etc/gitlab/gitlab.rb` y `/etc/gitlab/gitlab-secrets.json` y reiniciar.

```
$> sudo gitlab-ctl restart
```


Podemos comprobar que todos los procesos están correctamente con el comando:

```
$> sudo gitlab-rake gitlab:check SANITIZE=true
```