

Trabajo Práctico final



Integrantes: Leandro Casarin
Marcelo Rubini
Mariano Varela

Trabajo Práctico final

[¿Qué es Git?](#)

[¿Que es Gitlab?](#)

[¿Porque Gitlab?](#)

[Funcionalidades](#)

[Aplicabilidad](#)

[Instalación y configuración](#)

[Dependencias](#)

[Ruby](#)

[Usuario git del sistema](#)

[Base de datos](#)

[Gitlab](#)

[Apache](#)

[Modo de uso](#)

[Gestión de usuarios](#)

[Poryectos](#)

[Grupos](#)

[Permisos](#)

[El Muro](#)

[Wiki](#)

[Issues](#)

[Errores](#)

¿Qué es Git?

Es un software de control de versiones no centralizado orientado al mantenimiento de versiones de aplicaciones donde estas tienen un gran número de archivos de código fuente.

¿Que es Gitlab?

Es un repositorio de gestión de proyectos mediante una interfaz web.

Nos permite gestionar los grupos, personas y permisos a las mismas que estén involucrada en el proyecto o los proyectos, además de poder hacer un seguimiento del estado actual e histórico del mismo a través de las modificaciones que haya ido sufriendo.

¿Porque Gitlab?

Por brindar la posibilidad de no depender de la aplicación alojada en la nube, sino que podemos accederla de forma local e independiente de la misma.

Funcionalidades

- Logueo
- Diferentes tipos de acceso o permisos
- Seguimiento de incidencias
- Revision de código
- Gestión de equipos grupos
- Capacidad para importar repositorios
- Interfaz web
- Copias de seguridad

Aplicabilidad

1- Máquina de uso diario como servidor

Al tener el servidor alojado localmente no requiere de conexión a internet para trabajar.

La contra es que no es flexible a la hora de trabajar en equipo.

2- Máquina de asignada a trabajar como servidor de la aplicación o servicios de máquinas

virtuales para que alojen a la misma

Donde podemos acceder trabajar con el repositorio en cualquier momento y nos brinda un mayor ancho de banda

Instalación y configuración

Componentes de la aplicación

- Dependencias
- Ruby
- Usuario git del sistema
- Gitlab shell
- Base de datos
- Gitlab
- Apache

Dependencias

```
apt-get update -y
```

```
apt-get upgrade -y
```

Vim para editar las configuraciones(opcional, se podría optar por otro editor)

```
sudo apt-get install -y vim
```

Las librerías necesarias para la instalación

```
sudo apt-get install -y build-essential zlib1g-dev libyaml-dev libssl-dev libgdbm-dev libreadline-dev
```

```
libncurses5-dev libffi-dev curl git-core openssh-server redis-server checkinstall libxml2-dev libxslt-dev
```

```
libcurl4-openssl-dev libicu-dev
```

Python

```
sudo apt-get install -y python
```

Asegurarse de que tenemos Python 2.5 o mayor

```
python - version
```

Sino instalar la versión

```
sudo apt-get install python2.7
```

Ruby

Si ruby 1.8 esta instalado eliminarlo

```
sudo apt-get remove -y ruby1.8
```

Descargar Ruby y compilarlo

```
mkdir /tmp/ruby && cd /tmp/ruby
```

```
curl --progress http://ftp.ruby-lang.org/pub/ruby/1.9/ruby-1.9.3-p392.tar.gz | tar xz
```

```
cd ruby-1.9.3-p392
```

```
./configure
```

```
make
```

```
sudo make install
```

Instalar gema bundler

```
sudo gem install Bundler - no-ri - no-rdoc
```

Usuario git del sistema

Crear usuario git

```
sudo adduser --disabled-login --gecos 'GitLab' git
```

Ir al directorio de inicio y clonar gitlab shell

```
cd /home/git
```

```
sudo -u git -H git clone https://github.com/gitlabhq/gitlab-shell.git
```

```
cd gitlab-shell
```

Cambiar a la versión a utilizar

```
sudo-u git-H git checkout v1.4.0
```

```
sudo-u git-H cp config.yml.example config.yml
```

Editar la configuracion con vim reemplazando gitlab_url por “mi_dominio.com”

```
sudo -u git -H vim config.yml
```

Setup

```
sudo -u git -H ./bin/install
```

Base de datos

Instalando las librerias necesarias

```
sudo apt-get install-y mysql-server mysql-client libmysqlclient-dev
```

Cuando nos pida ingresar una contraseña lo hacemos y la recordamos

Ingresar a mysql para crear un usuario para gitlab

```
mysql-u root-p
```

```
CREATE USER 'gitlab' @ 'localhost' IDENTIFICADO POR '$ password';
```

Crear la base para gitlab

```
CREATE DATABASE IF NOT EXISTS `gitlabhq_production` DEFAULT CHARACTER SET `utf8` COLLATE `utf8_unicode_ci`;
```

Darle al usuario los permisos necesarios

```
GRANT SELECT, LOCK TABLES, INSERT, UPDATE, DELETE, CREATE, DROP INDEX, ALTER en `gitlabhq_production` * TO  
'gitlab' @ 'localhost';
```

Salir de la base

```
\q
```

Gitlab

Instalar gitlab en el directorio indicado

```
cd /home/git
```

Clonar el repositorio

```
sudo -u git -H git clone https://github.com/gitlabhq/gitlabhq.git gitlab
```

```
cd /home/git/gitlab
```

```
sudo gem install mysql2
```

Checkout de una versión estable

```
sudo -u git -H git checkout 5-3-stable
```

Copiar configuracion de ejemplo

```
sudo -u git -H cp config/gitlab.yml.example config/gitlab.yml
```

Cambiar la configuracion con el dominio que queremos direccionar

```
sudo -u git -H vim config/gitlab.yml
```

Otorgar permisos en directorios

```
sudo chown -R git log/
```

```
sudo chown -R git tmp/
```

```
sudo chmod -R u+rwX log/
```

```
sudo chmod -R u+rwX tmp/
```

Crear directorio con sus permisos

```
sudo -u git -H mkdir /home/git/gitlab-satellites
```

```
sudo -u git -H mkdir tmp/pids/
```

```
sudo -u git -H mkdir tmp/sockets/
```

```
sudo chmod -R u+rwX tmp/pids/
```

```
sudo chmod -R u+rwX tmp/sockets/
```

```
sudo -u git -H mkdir public/uploads
```

```
sudo chmod -R u+rwX public/uploads
```

Copiar configuración de puma.rb de ejemplo

```
sudo -u git -H cp config/puma.rb.example config/puma.rb
```

Configuración global del usuario git. El email debe estar configurado de acuerdo al gitlab.yml

```
sudo-u git-H git config - user.name global "GitLab"
```

```
sudo-u git-H git config - global de user.email "gitlab @ localhost"
```

Configurar opciones de gitlab.db

```
sudo -u git cp config/database.yml.mysql config/database.yml
```

Asegurarse de cambiar usuario y contraseña en database.yml

```
sudo -u git -H vim config/database.yml
```

Darle los permisos de lectura solo a git

```
sudo -u git -H chmod o-rwx config/database.yml
```

Instalando las gemas necesarias

```
cd /home/git/gitlab
```

```
sudo gem install charlock_holmes --version '0.6.9.4'
```

```
sudo -u git -H bundle install --deployment --without development test postgres unicorn aws
```

Iniciar la base de datos

```
sudo -u git -H bundle exec rake gitlab:setup RAILS_ENV=production
```

Cuando nos pregunte si queremos crear la base de datos tipeamos “yes”

Instalar el script de inicio

```
sudo cp lib/support/init.d/gitlab /etc/init.d/gitlab
```

```
sudo chmod +x /etc/init.d/gitlab
```

Apache

Instalar librerías necesarias

```
sudo apt-get install apache2-prefork-dev libapr1-dev libaprutil1-dev
```

Instalar la gema passenger

```
cd /home/git/gitlab/
```

```
sudo gem install passenger
```

```
sudo passenger-install-apache2-module
```

Ahora es necesario instalar gitlab y apache

```
sudo a2ensite gitlab
```

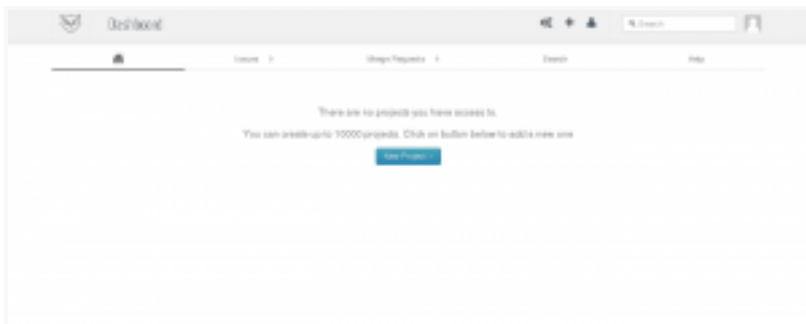
```
sudo service gitlab restart
```

```
sudo service apache2 restart
```

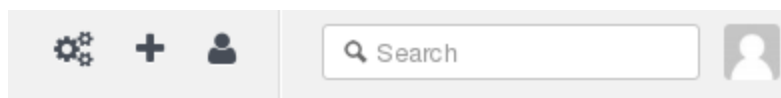
Levantar la aplicación en el puerto configurado

Modo de uso


La pantalla principal o *Dashboard* muestra toda la actividad del usuario y sus proyectos.







Empezando por la izquierda tenemos ; el icono área de administración(admin area), icono para crear un nuevo proyecto, icono para ver el perfil del usuario, búsqueda y ícono que despliega un menú con las opciones de ver el perfil de usuario y salir del sistema.





La pantalla del perfil de usuario esta dividida en varios apartados que se muestran en la parte superior, el primer apartado por la izquierda (símbolo de una casa) nos permite añadir información del usuario, cuenta de usuario(*Account*) dónde podemos cambiar la contraseña y el nombre de usuario, clave de *SSH*, Diseño (*Desing*) donde podemos aplicar diversos temas a la cuenta y por último el historial del usuario. Para cambiar la contraseña nos vamos al apartado *Account* dónde vemos la opción *Password*, escribimos la contraseña dos veces y pulsamos en el botón *Save password*.

 Profile





 Account SSH Keys  Design History

Private token keep it secret!

Private token used to access application resources without authentication.
It can be used for atom feed or API

Reset

QcYktVnjBLMukuHTVpkG

Password

After successful password update you will be redirected to login page where you should login with new password

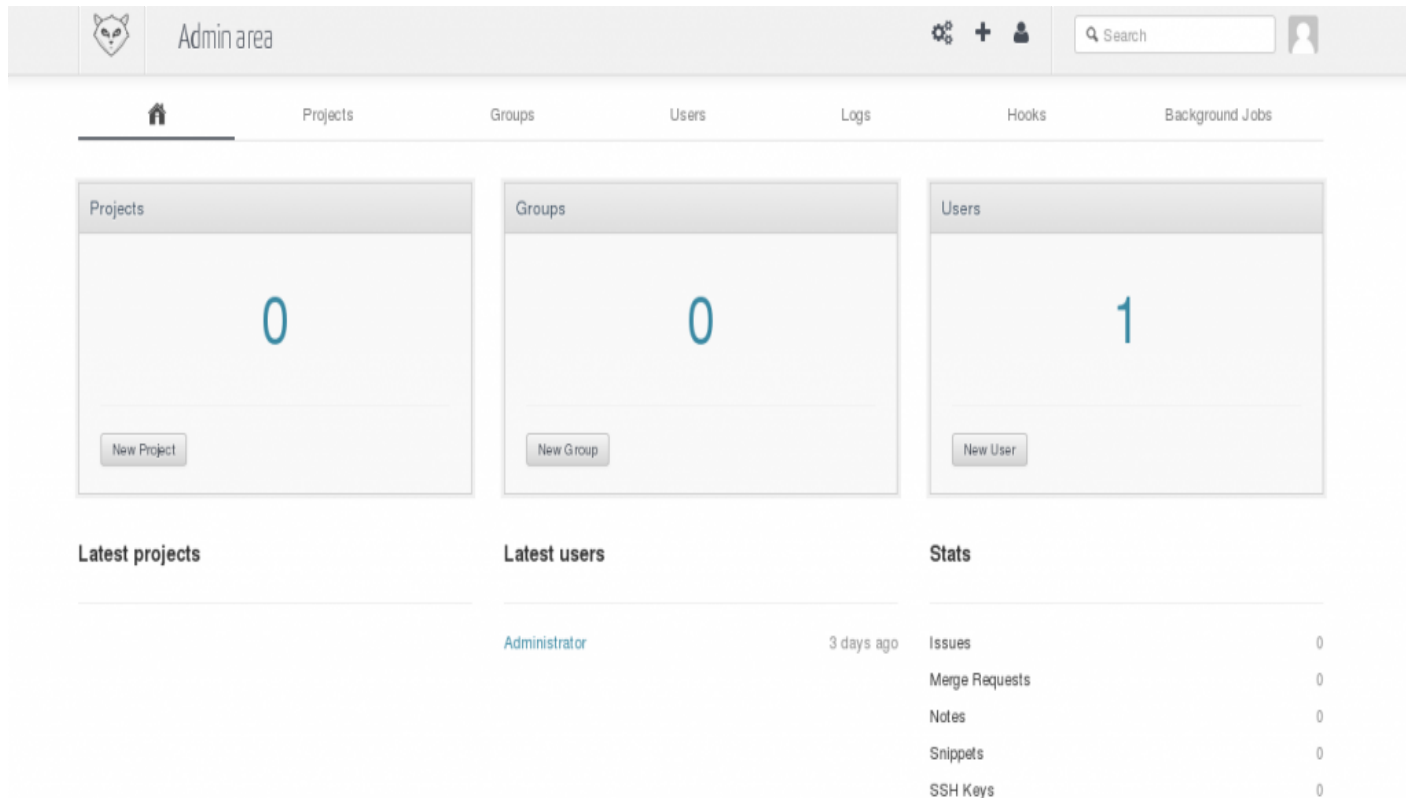
Password

Password confirmation

Save password

Gestión de usuarios

Es recomendable no usar el usuario administrador para los proyectos, para eso podemos crear un nuevo usuario para gestionar los repositorios. Para crear una usuario se debe realizar desde la cuenta administrador. Nos situamos en el área de administración del usuario administrador.



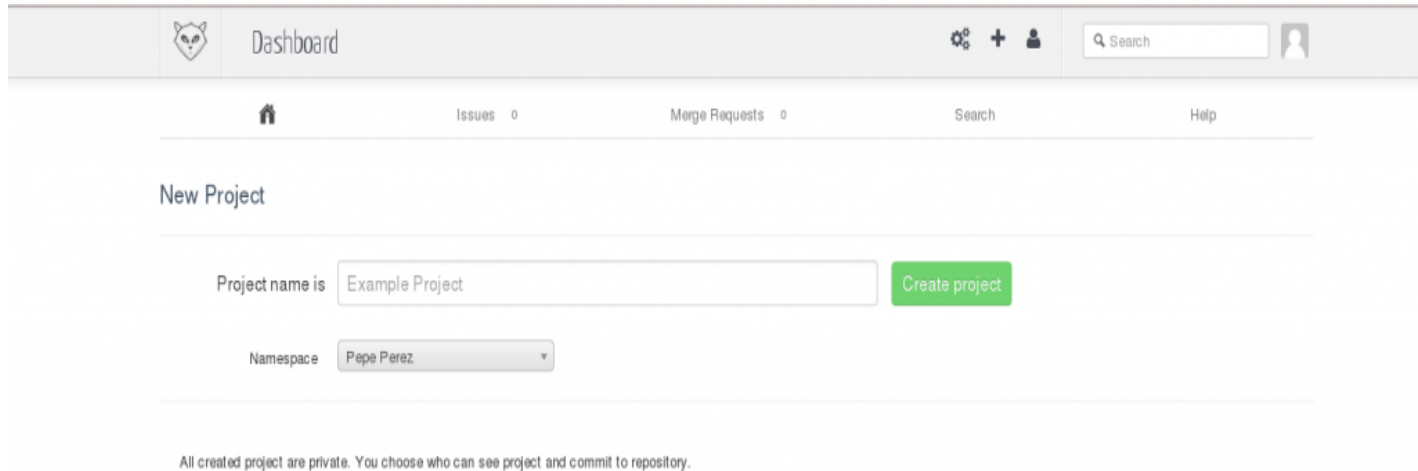
Hacemos esto para que nadie que sepa con qué usuario trabajamos en un repositorio pueda entrar a nuestro perfil y revisar nuestros proyectos si es que no lo deseamos.

En esta área podemos crear o administrar usuarios, proyectos y grupos. Para crear un usuario nuevo, pulsamos en el botón “*New User*” donde se muestra una nueva pantalla donde introduciremos los datos del nuevo usuario, marcar la opción “*Administrator*” para que el nuevo usuario tenga el rol de administrador, el nuevo usuario tendrá un límite de 10 proyectos que se puede modificar escogiendo el valor apropiado en el apartado “*Project limit*” pulsando en el botón “*Save*”.

Proyectos

Para crear un proyecto, si estamos en el apartado *User*, pulsamos en el apartado *Projects*. Otra forma de llegar es *Admin area->Projects*.

En la siguiente ventana se pulsa en *New Projects*, para crear un proyecto.

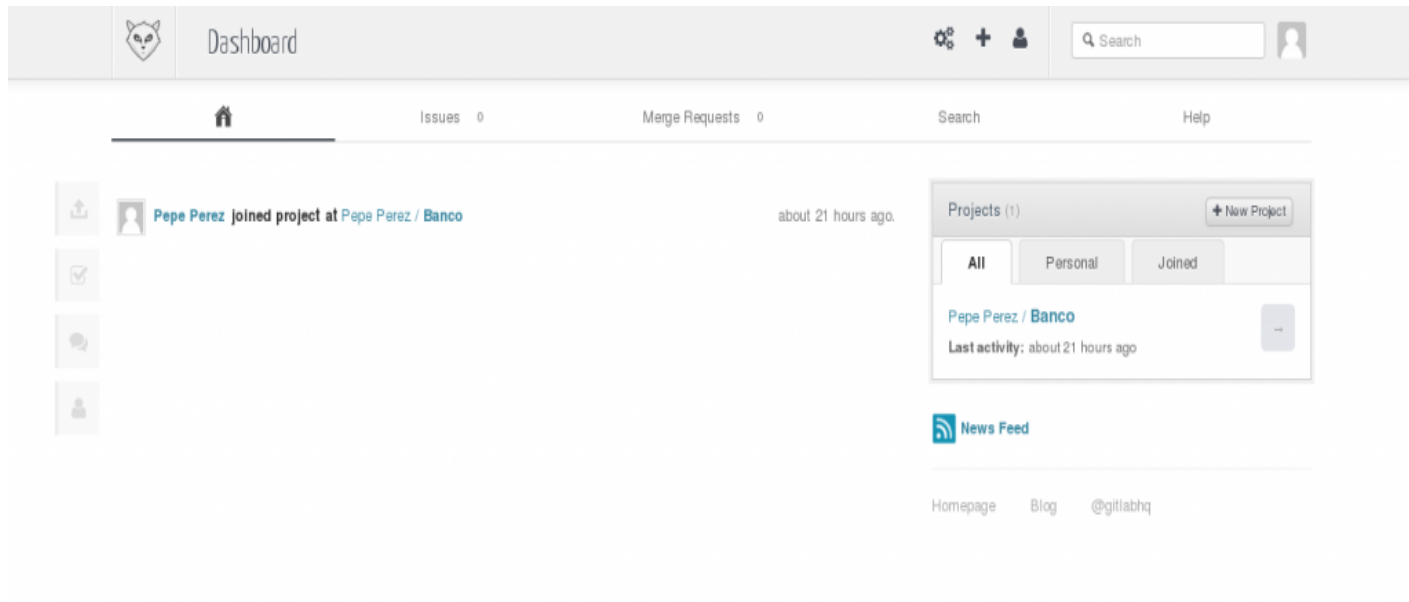


The screenshot shows the 'New Project' form in the GitLab dashboard. At the top, there is a header bar with the GitLab logo, the word 'Dashboard', and navigation links for 'Issues', 'Merge Requests', 'Search', and 'Help'. Below the header, the 'New Project' section contains a text input field for 'Project name' with the value 'Example Project', a green 'Create project' button, and a dropdown menu for 'Namespace' with the value 'Pepe Perez'. A note at the bottom states: 'All created project are private. You choose who can see project and commit to repository.'

Seleccionamos el nombre de proyecto (*Project name*) y el espacio de nombre (*Namespace*) que especifica el nombre del usuario que se ha creado anteriormente. Esto último, indica que el proyecto creado depende de ese usuario, pulsamos en *Create project*.

GitLab trabaja como un repositorio remoto donde podremos administrar y compartir los proyecto, pero trabajaremos con un repositorio local, que estará en nuestro ordenador donde vamos a trabajar , realizamos “commit” y después subiremos los cambios mediante el comando *push* de *Git*.

Ahora en el *DashBoard* aparece más contenido, a la izquierda veremos un historial que registrara cualquier actividad del usuario *Pepe* y sus proyectos. En la parte derecha veremos los proyectos del usuario, también tenemos un *RSS(New Feed)* y con un cliente *RSS* podremos estar al tanto de la actividad del historial si tener que conectarnos a *GitLab*.




Grupos

Cuando tenemos múltiples proyectos en *GitLab* podemos agruparlos. Los grupos facilitan la administración de varios proyectos.

Podemos crear un grupo con nombre Personal (por ejemplo). Nos situamos en la área de administración, vemos un apartado *Groups* y pulsamos en el botón *New Group*.



Escribimos el nombre deseado para el grupo



Projects

Groups

Users

Logs


Hooks

Background Jobs

Group: Personal


Group

Name:Personal

 Rename

Path:/home/git/repositories/personal

Owner:Pape Perez

 Change owner

Group is empty

Move projects to group

You can move only projects with existing repos
Group projects will be moved in group directory and will not be accessible by old path

Projects

Select projects

Add

En la ventana anterior vemos la información del grupo creado y solo necesitamos añadir proyectos al grupo. Escribimos en la parte inferior el nombre del proyecto creado y lo añadimos.

Si borramos un grupo que contenía proyectos, estos se borrarán también. Si los queremos conservar primero movemos los proyectos a otro grupo y luego sí borramos el grupo.

Permisos

GitLab nos brinda una serie de permisos a los usuarios que cuando son asignados a un proyecto tenemos elegir, estos permisos se engloban en diferentes *tipos de accesos*, dependiendo del tipo que se asigne al usuario podrá realizar diferentes tareas en un proyecto.

Para ver la categoría de un usuario para un proyecto determinado nos vamos al área de administración del usuario, pulsamos en el apartado *User* y se mostrará un listado de los usuarios. Pulsamos en el nombre del usuario (columna *Name*), y aparecerá la información del

usuario.

Add User to Projects

Projects	Project Access:
<input type="text" value="Select projects"/>	<input type="text" value="Guest"/>
<input type="button" value="Add"/>	Read more about project permissions here

Owner of groups:

Name
Personal

Authorized Projects:

Name	Project Access		
Personal / Banco	Master	<input type="button" value="Edit Access"/>	<input type="button" value="Remove from team"/>

En esta ventana se muestra información del usuario ,en la parte inferior, podemos asignar el usuario a un proyecto y el tipo de acceso al proyecto(permisos),e información del grupo al que pertenece. Por último, los proyectos que tiene asignados y el tipo de acceso, si pulsamos en botón *Edit Access*, se puede modificar el tipo de acceso del usuario a ese proyecto.

El Muro

Dentro del Proyecto hay una sección denominado *Wall* donde se pueden escribir mensajes que leerán todos los usuario del proyecto. Escribimos un mensaje y pulsamos en el botón añadir (Add comment).

Home Files Commits Network Issues 1 Merge Requests 0 Wall Wiki

Comments are parsed with [GitLab Flavored Markdown](#).

NOTIFY VIA EMAIL: ☒ Project team ATTACHMENT: File name... Any file up to 10 MB Choose File ...

Pepe Perez 1 day ago
probando el muro

rocio 1 day ago
probando el muro

Wiki

Si queremos escribir un documento explicando como se instala nuestra aplicación o un tutorial de la aplicación, una opción es utilizar el apartado *Wiki* de *GitLab* que nos permite escribir un documento, para acceder a la wiki de un proyecto.

Si deseamos escribir un documento en la wiki con diferentes formatos, GitLab proporciona una sistema de marcas que añade formatos al texto y cambiar el aspecto del texto en la Wiki.

Wiki prueba

Pages History Edit

Podemos escribir con formato, escribir en *cursiva*, **negrita**

Título

Una lista

- paso 1
- paso 2
- paso 3

También podemos poner links [rooteando](#).

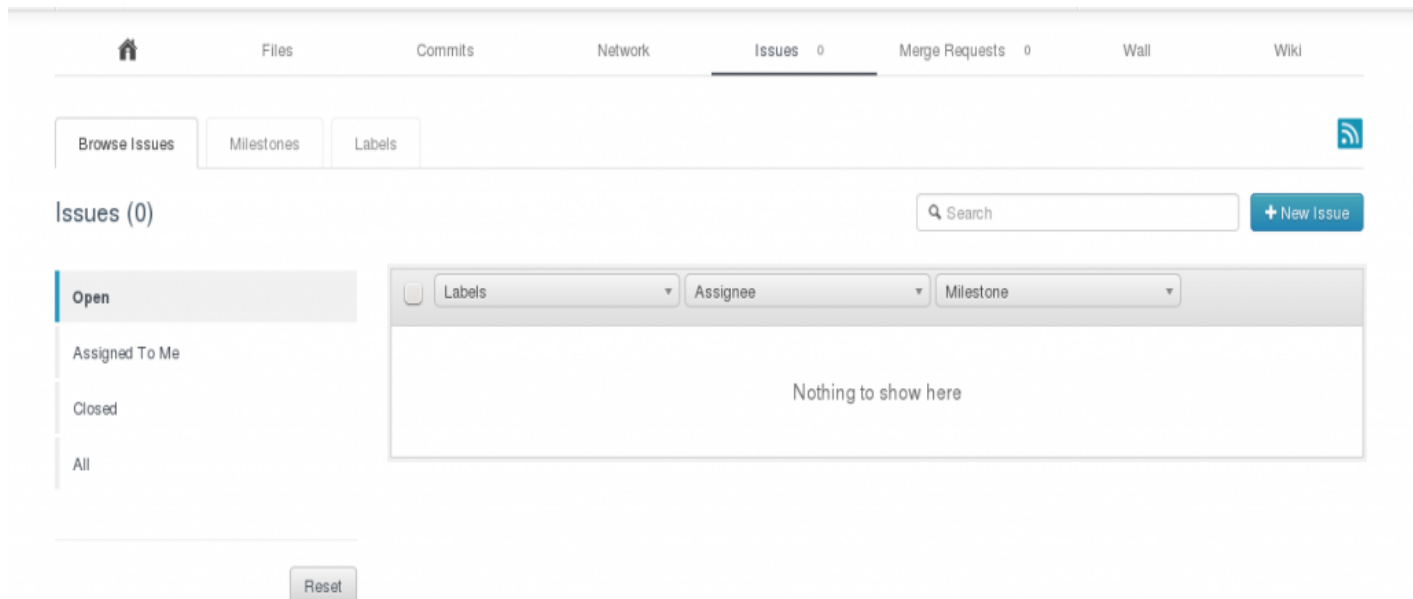
Last edited by Pepe Perez, 1 minute ago

Issues

También podemos generar issues de tareas por hacer en el proyecto y repartirlas entre los

usuarios del mismo.

Dentro del proyecto vamos a la sección Issues.



Para crear una nueva seleccionamos New Issue.

Le damos un nombre en el campo Subject, se lo asignamos a un usuario en particular (opcional) y podemos dar detalles de lo que se desee sobre esa tarea. Además podemos agregar etiquetas que se refieran a dicha tarea.

A dicha tarea podemos editarla, cerrarla en caso de que haya sido completada o agregar un comentario en la parte inferior, para que lo vean los demás usuarios. También podemos adjuntar un archivo a la issue seleccionando Choose file.

New Issue

Subject *

Assign to Select a user

Milestone Select milestone

Labels

Separate with comma.

Details

Issues are parsed with [GitLab Flavored Markdown](#).

Submit new issue

Cancel

Errores

Problema con gema pg version 0.15.1

Al correr el comando:

```
sudo -u git -H bundle install --deployment --without development test mysql
```

que nos permite instalar las gemas necesarias para bindear la aplicacion con la base de datos podemos llegar a encontrarnos con que la gema pg no se encuentra correctamente instalada.

Con lo cual lo primero que estariamos tentados a hacer es instalarla a mano, pero no. La solucion indicada ante este conflicto es antes de ejecutar el comando mostrado anteriormente instalemos la libreria libpq-dev que permite instalar pq:

```
gem install libpq-dev
```

Al finalizar dicha instalacion ejecutamos el comando:

```
sudo -u git -H bundle install --deployment --without development test mysql
```

Y la gema pq se deberá instalar sin problemas junto a las demás.