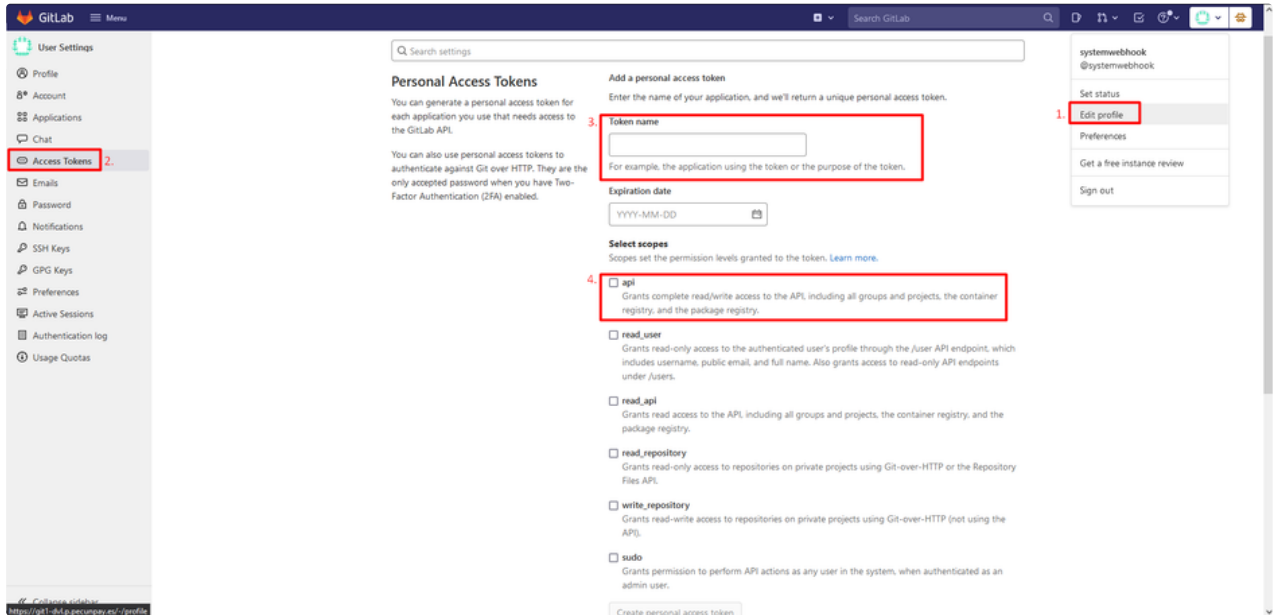


Configuración Webhook

Para poder realizar esto hay que seguir varios pasos:

Los pasos del 1 al 3 sólo hay que hacerlos en la configuración inicial.

1. Configurar un usuario con permisos para que pueda acceder a todos los proyectos en **Gitlab** y otro usuario en el **Jenkins** para que pueda ejecutar los jobs.
2. Crear un token para dicho usuario en **Gitlab** para que pueda acceder al API.
 - a. Para ello tenemos que loguearnos/impersonar al usuario y dentro de su perfil, ir a **Edit Profile > Access Tokens** . Ahí creamos un token con el nombre que queramos y permisos de **api** , le damos a crear token y lo guardamos, ya que sólo se mostrará una vez. En el pantallazo se muestra el orden que hay que seguir.



3. En el servidor de **Jenkins** hay que realizar la siguiente configuración, habiendo instalado previamente el plugin de Gitlab para Jenkins.
 - a. Nos vamos al apartado de **Manage Jenkins > Configure System > Gitlab** , activamos **Enable authentication for 'project' endpoint** y configuramos los datos de conexión.
 - b. En la parte de credenciales, en **API Token for accessing Gitlab** tenemos que crear una credencial con el Token que hemos creado anteriormente.
 - c. Le damos a **Test Connection** y pone que está correcto, le damos a guardar para terminar este paso.

Gitlab

☒ Enable authentication for '/project' end-point

GitLab connections

Connection name

A name for the connection

git1-dvl.p.pecunpay.es

Gitlab host URL

The complete URL to the Gitlab server (e.g. http://gitlab.mydomain.com)

https://git1-dvl.p.pecunpay.es

Credentials

API Token for accessing Gitlab

GitLab API token (API Token Gitlab)

+ Add

Advanced...

Test Connection

Esta parte hay que hacerlo por cada proyecto que queramos configurar.

1. Ahora que configurar el job de **Jenkins** para que acepte las llamadas por webhook.
 - a. Creamos un job nuevo, ya sea Freestyle o Pipeline.
 - b. La parte que hay que configurar es **Build Triggers** marcando la opción de **Build when a change is pushed to GitLab**. Para nuestro caso hemos marcado que se active el job cuando hagan un push al código sólo para la branch de **develop**.
 - c. Para que sólo afecte a una branch, en la parte de **Advanced**, tenemos que indicarlo así.

☒ Build when a change is pushed to GitLab. GitLab webhook URL: https://jenkinsconstruccion1-dvl.p.pecunpay.es:8443/project/dvl-webhook/webhook-test

Enabled GitLab triggers

☒ Push Events

☐ Push Events in case of branch delete

☐ Opened Merge Request Events

☐ Build only if new commits were pushed to Merge Request ?

☐ Accepted Merge Request Events

☐ Closed Merge Request Events

Rebuild open Merge Requests

Never

● Filter branches by name ?

Include

develop,

⚠ Following patterns don't match any branch in source repository:

Exclude

2. En la parte del proyecto en **Gitlab**, tenemos que ir a **Settings > Integrations** y buscar Jenkins. Lo activamos y configuramos los datos que nos piden, tales como la URL del servidor de Jenkins, el nombre del job, que no puede ir con espacios y tiene que ir de la siguiente forma si está dentro de una carpeta: `<carpeta>/<job>` y el usuario que hemos creado al principio. Le damos a **Test settings** y comprobamos que conecta correctamente.