GESTIÓN DE USUARIOS CON KEYCLOAK

1.Objetivos

Los objetivos que se quieren cumplir en este proyecto son sencillos. Básicamente queremos usar un servidor IAM (Administración de identidades y accesos) como es keycloak, para gestionar los usuarios y sus permisos en nuestros distintos servicios. Para hacerlo más sencillo y familiar he elegido muchos servicios que usamos día a día en nuestro instituto para gestionar distintas tareas y que pueden llegar también a ser útiles en un entorno laboral. Tales como:

- Redmine
- Wordpress
- Moodle

Estos son los servicios que tengo pensado implementar.

2. Escenario

La idea era instalarlo en servidores docker, y aunque durante mucho tiempo he estado gestionando todo en docker para diferentes pruebas no fui capaz de sacarlo hacia adelante, así que todos los servicios están en local.

Todos los servicios estarán conectados con nuestro servidor Keycloak obviamente para poder realizar las comprobaciones del proyecto. Y se accederá en local a todos los servicios.

3. Fundamentos teóricos y conceptos

- **Servidor IAM:** Servidor de gestión de accesos e identidades. En nuestro caso usaremos Keycloak, el cual nos permitirá realizar acciones tales como:
    - Inicio de sesión único.
    - Creación de usuarios y roles.
    - Cambio de contraseñas.
    - Monitorización de accesos.
    - Creación de usuarios temporales.
    - Gestionar los permisos de dichos usuarios.
    - Múltiple factor de autenticación.
    - Revocación de accesos.
    - Gestionar olvido de contraseñas.
    - Validación de cuentas.
    - Conexión con sistema LDAP.
    - Configuración en alta disponibilidad.
- **SAML2:** Es lo que nos permitirá realizar un inicio de sesión único. Crearemos los usuarios y roles en keycloak, sin embargo, tendremos SAML como intermediario para realizar el inicio de sesión único. Es un protocolo de autenticación en el que los Proveedores de identidad (IdP) intercambian documentos XML que permiten a un usuario final acceder a un Proveedor de servicios.

- **Oauth2:** Se trata de un protocolo para pasar la autorización de un servicio a otro sin compartir las credenciales de usuario reales, como un nombre de usuario y contraseña. Con esta herramienta un usuario puede iniciar sesión en una plataforma y luego estar autorizado para realizar acciones y ver datos en otra plataforma. Utiliza tokens de acceso. Un Token de acceso es un dato que representa la autorización para acceder a los recursos en nombre del usuario final.
- **OpenID:** Otro intermediario como SAML2, el que probablemente usemos, ya que tiene mayor documentación de soporte en instalaciones en todos los sistemas. Es un protocolo de identidad que utiliza los mecanismos de autorización y autenticación de OAuth 2.0. Realiza la autenticación del usuario, el consentimiento del usuario y la emisión de tokens.
- **Token de ejecución:** Está diseñado principalmente como un medio para conceder acceso a un conjunto de recursos, por ejemplo, API remotas o datos de usuario.
- **OTP: E**s una contraseña que pierde su validez después de su uso, de ahí su denominación. Por lo general, se emplea como parte de una autenticación de doble factor. Nosotros usaremos este servicio para autentificarnos de forma segura y usaremos FreeOTP como aplicación que proporcione los códigos

4.Instalaciones y configuraciones

4.1 Keycloak

- Debemos instalar dependencias como java:

```
apt-get install default-jdk -y
```

- Descargamos keycloak

```
wget https://github.com/keycloak/keycloak/releases/download/15.0.2/keycloak-15.0.2.tar.gz
```

- Lo descomprimimos y movemos a /opt

```
tar -xvzf keycloak-15.0.2.tar.gz
mv keycloak-15.0.2 /opt/keycloak
```

- Añadimos usuarios y grupos necesarios.

```
groupadd keycloak
useradd -r -g keycloak -d /opt/keycloak -s /sbin/nologin keycloak
```

- Cambiamos los permisos del servicio.

```
chown -R keycloak: /opt/keycloak
chmod o+x /opt/keycloak/bin/
```

- Procedemos a habilitar los ficheros de configuración:

```
cp /opt/keycloak/docs/contrib/scripts/systemd/wildfly.conf
/etc/keycloak/keycloak.conf
cp /opt/keycloak/docs/contrib/scripts/systemd/launch.sh /opt/keycloak/bin/
```

- Cambiamos los permisos.

```
chown keycloak: /opt/keycloak/bin/launch.sh
```

- Ahora entramos en el fichero “/opt/keycloak/bin/launch.sh” que debe quedar de la siguiente forma:

```
#!/bin/bash
if [ "x$WILDFLY_HOME" = "x" ]; then
WILDFLY_HOME="/opt/keycloak"
fi
if [[ "$1" == "domain" ]]; then
$WILDFLY_HOME/bin/domain.sh -c $2 -b $3
else
$WILDFLY_HOME/bin/standalone.sh -c $2 -b $3
fi
```

- Ahora crearemos un servicio para que se ejecute nuestro servidor keycloak.

```
cp /opt/keycloak/docs/contrib/scripts/systemd/wildfly.service
/etc/systemd/system/keycloak.service
```

- Debe quedar de la siguiente forma:

```
[Unit]
Description=The Keycloak Server
After=syslog.target network.target
Before=httpd.service
[Service]
Environment=LAUNCH_JBOSS_IN_BACKGROUND=1
EnvironmentFile=/etc/keycloak/keycloak.conf
User=keycloak
Group=keycloak
LimitNOFILE=102642
PIDFile=/var/run/keycloak/keycloak.pid
ExecStart=/opt/keycloak/bin/launch.sh $WILDFLY_MODE $WILDFLY_CONFIG
$WILDFLY_BIND
StandardOutput=null
[Install]
WantedBy=multi-user.target
```

- Creamos un usuario administrador para poder loggearnos:

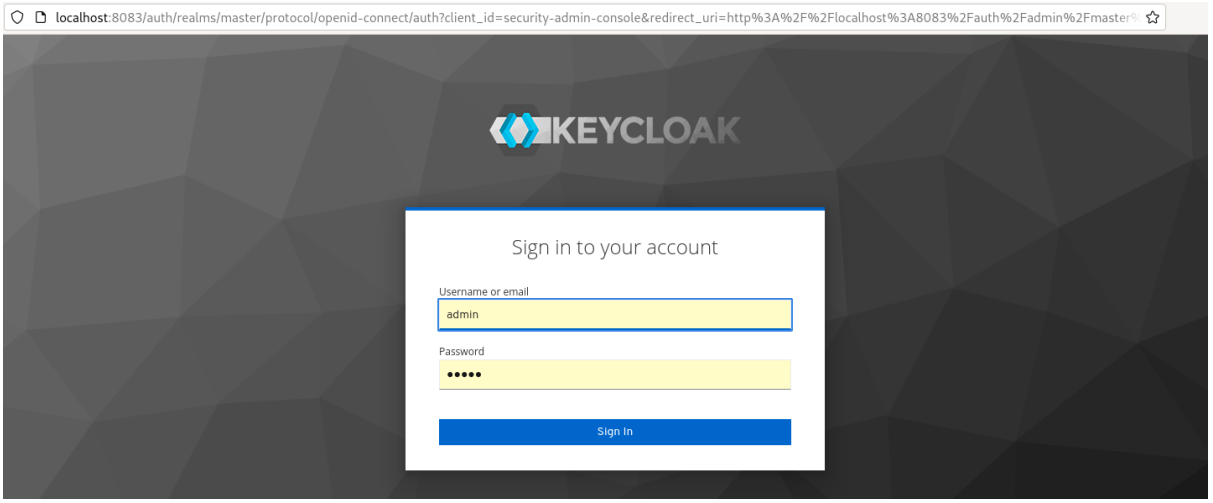
```
/opt/keycloak/bin/add-user-keycloak.sh -u admin
```

- Para no tener problemas debemos deshabilitar el requerimiento de SSL.

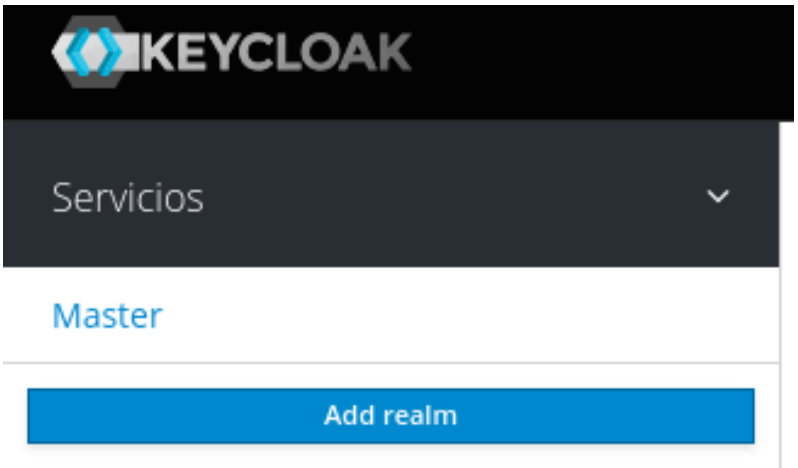
/opt/keycloak/bin/kcadm.sh config credentials --server http://localhost:8080/auth --realm master --user admin

/opt/keycloak/bin/kcadm.sh update realms/master -s sslRequired=NONE

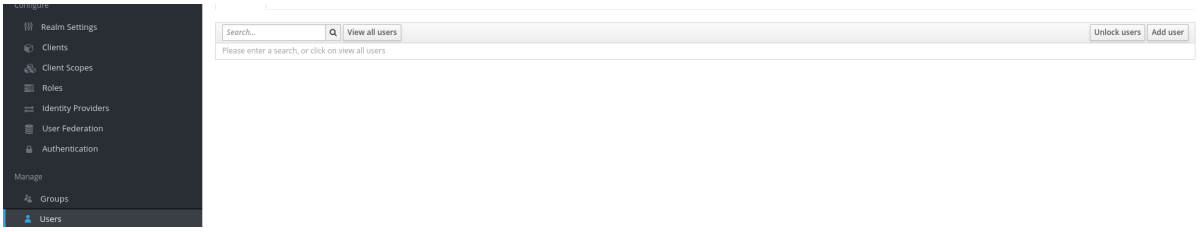
- Y ya podremos acceder a la página de administración web.



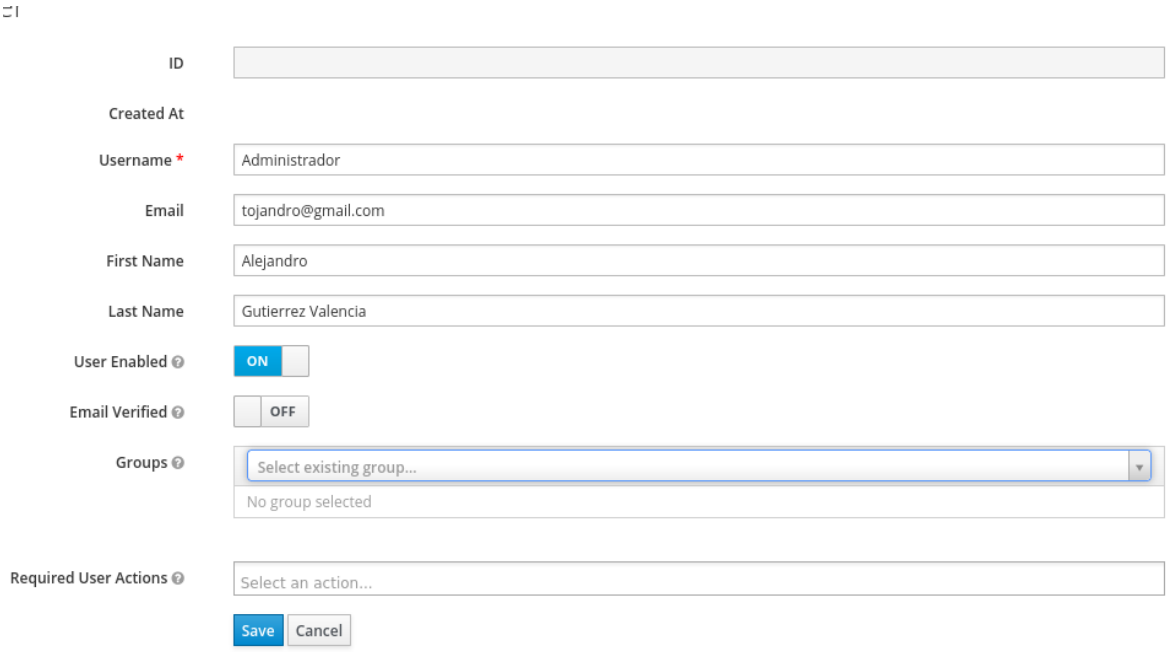
- Vamos a entrar y configurar un nuevo Realm, los realm son dominios dentro de nuestro servidor donde manejaremos los distintos clientes y usuarios.



- Dentro del realm que hemos creado llamado “Servicios” vamos a crear un usuario. Vamos a la pestaña de “users” y le damos a “add user”.



- Lo configuramos a nuestro gusto, en mi caso crearé uno que nos sirva como administrador.

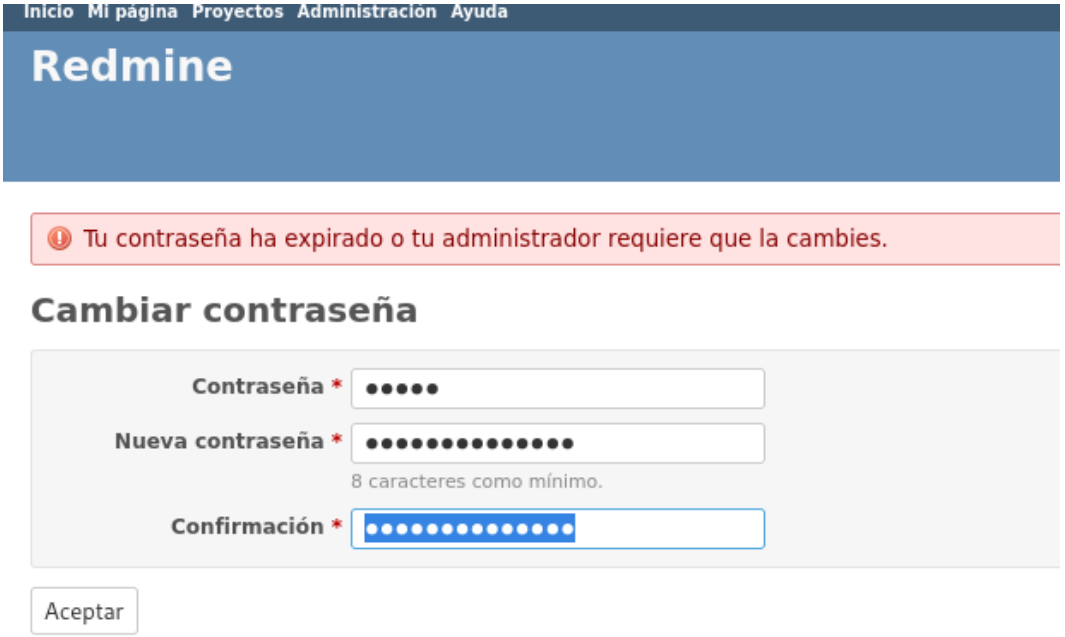


- Una vez creado entramos y le ponemos una contraseña. Desde el apartado “Credentials”, desactivamos la opción de “temporal”, añadimos una contraseña y guardamos.

4.2 Redmine

- Instalación paso a paso, primero entramos con admin/admin tal como está configurado:

- Seguidamente se nos pedirá que cambiemos la contraseña:



4.3 Instalación de plugin OpenID en Redmine

- Por ahora entraremos manualmente en el contenedor y lo instalaremos, más adelante crearemos una receta ansible que haga estos pasos sola.

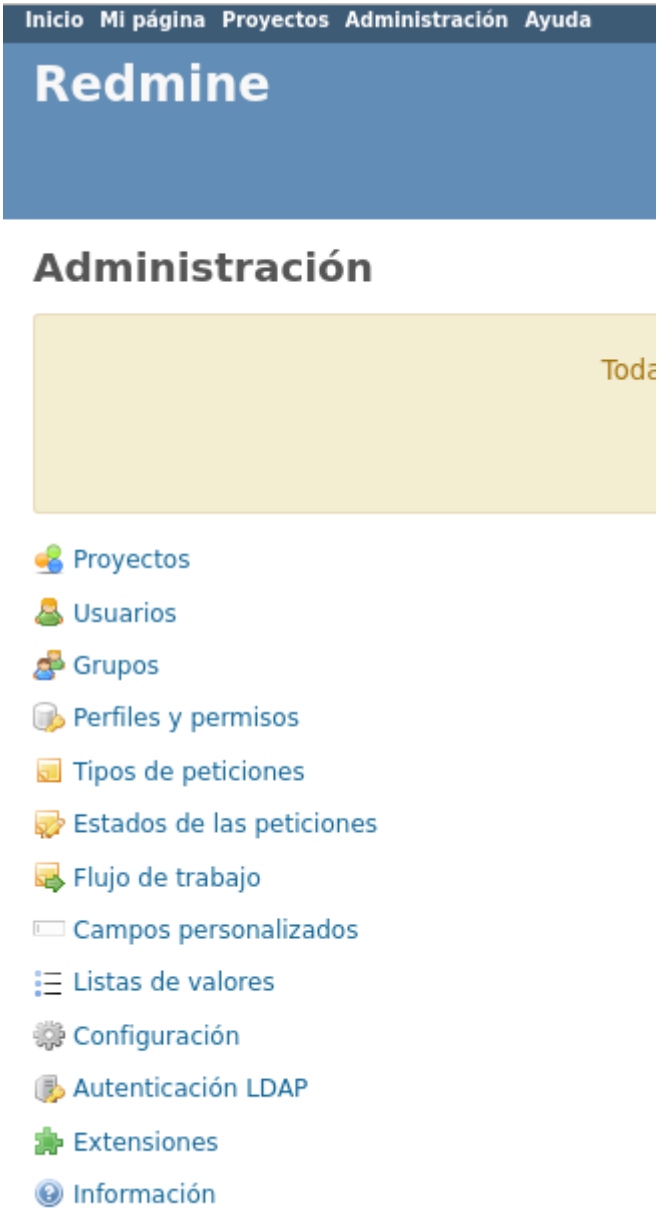
- Clonamos el plugin en la carpeta correspondiente.

```
root@30f00b33dd75:/usr/src/redmine/plugins# git clone
https://github.com/devopskube/redmine_openid_connect
Cloning into 'redmine_oidc'...
remote: Enumerating objects: 486, done.
remote: Counting objects: 100% (46/46), done.
remote: Compressing objects: 100% (26/26), done.
remote: Total 486 (delta 28), reused 20 (delta 20), pack-reused 440
Receiving objects: 100% (486/486), 93.33 KiB | 1.35 MiB/s, done.
Resolving deltas: 100% (197/197), done.
```

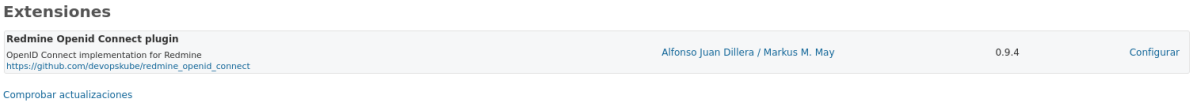
- Una vez dentro realizamos las siguientes acciones:

```
bundle install
bundle exec rake redmine:plugins:migrate RAILS_ENV=production
```

- Reiniciamos el servidor y nos dirigimos a Administración > Extensiones.



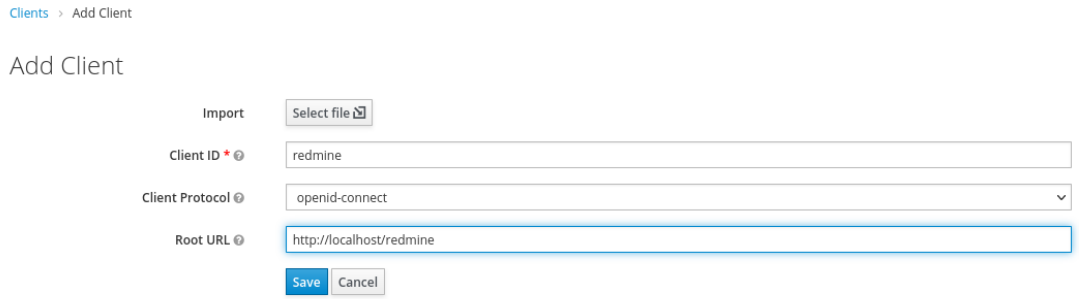
- Y ya tendremos nuestro plugin instalado.



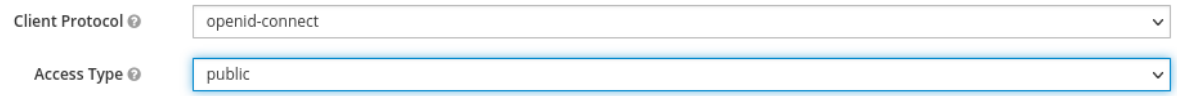
4.3 Creación de cliente redmine en keycloak.

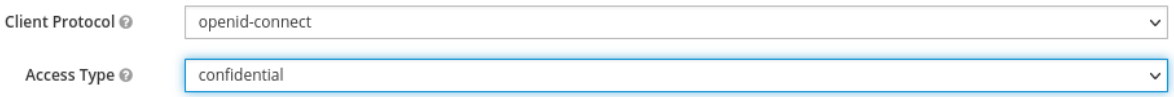
- Para configurar cada servicio necesitamos crear un cliente en nuestro realm. Para ello obviamente, iremos a la pestaña de “clients” y crearemos uno nuevo.

- Empezaremos con esta simple configuración, el nombre del cliente, el protocolo “OpenId” y la URL de nuestro servicio.

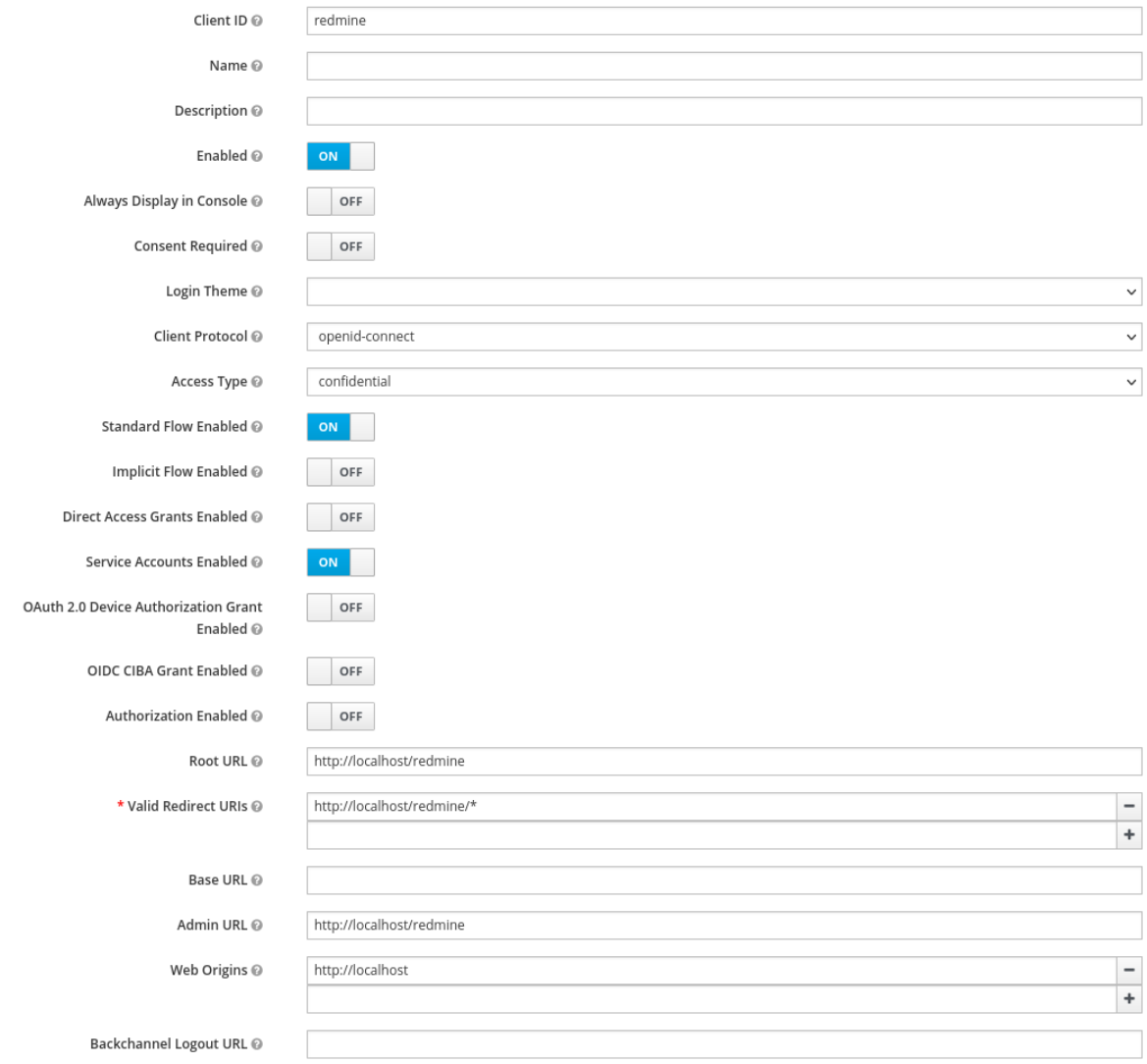


- Una vez guardemos, nos aparecerá una pestaña con este cliente y una configuración mucho más amplia, en primer lugar cambiaremos el protocolo de público a confidencial.

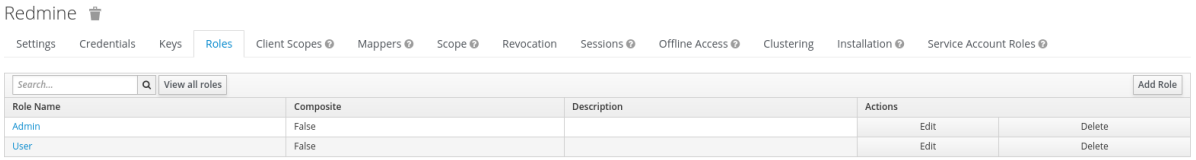




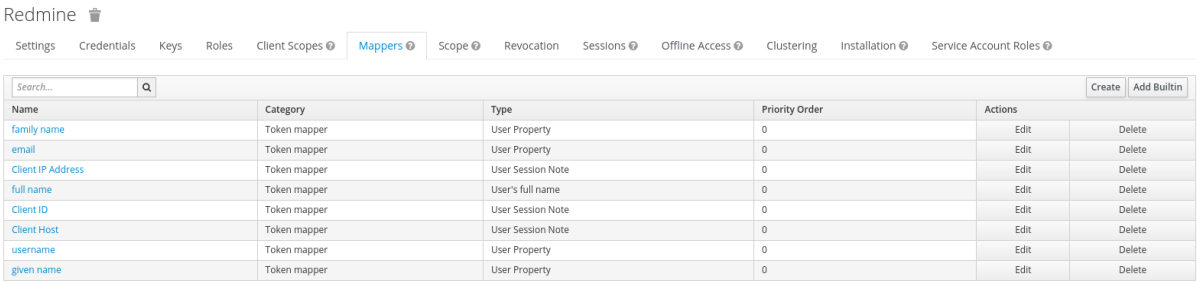
- Esta es la configuración principal:



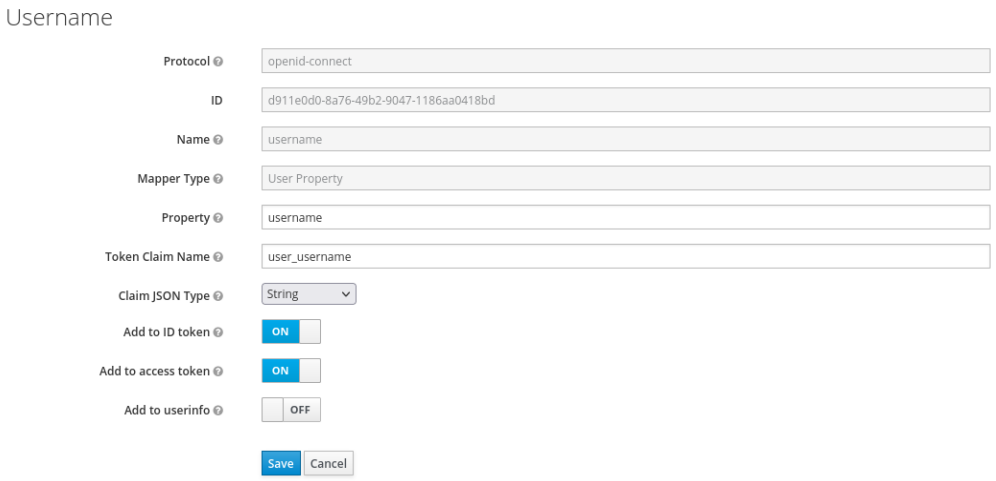
- En la pestaña roles crearemos dos nuevos “Admin” y “User”.



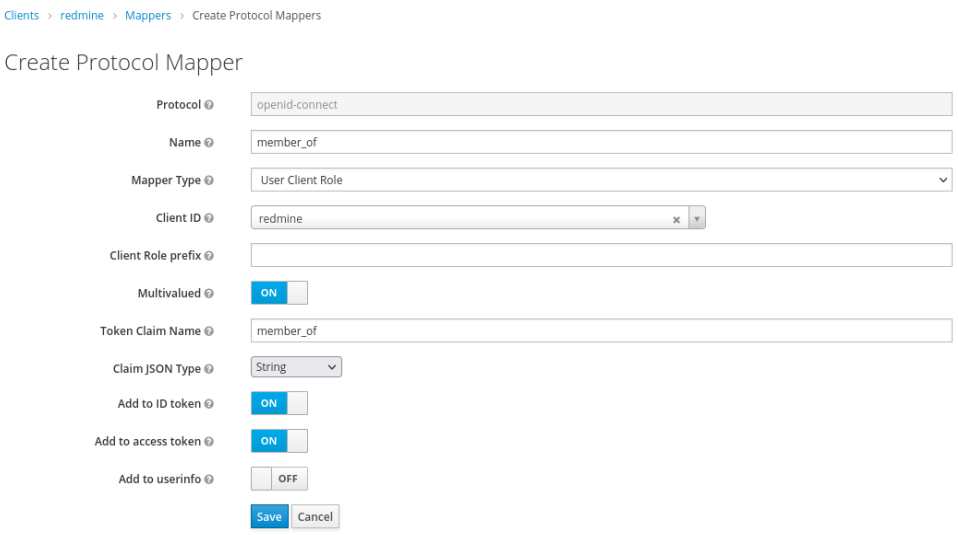
- En “Mappers” debemos añadir todos estos pre-construidos por defecto en keycloak.



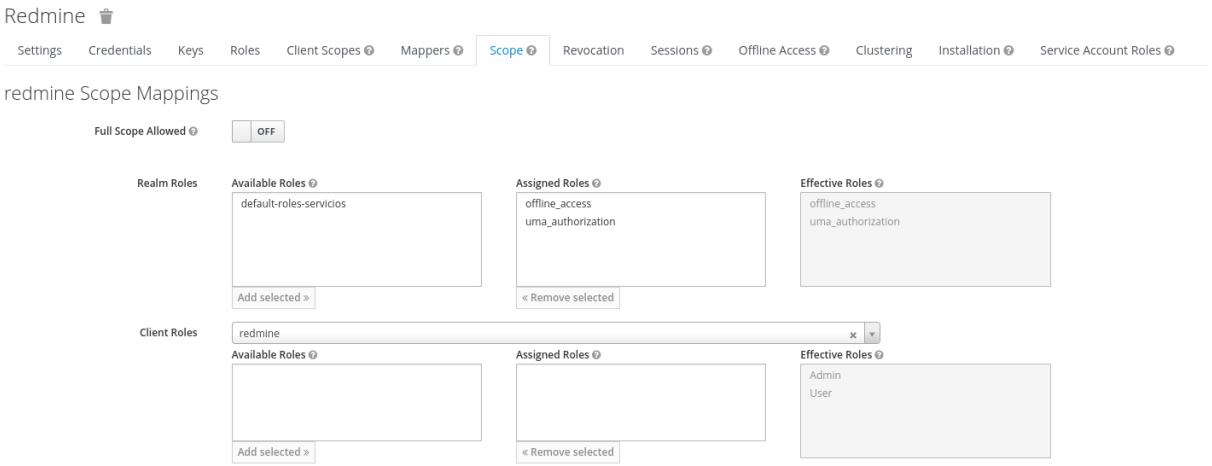
- Solo debemos cambiar el mapper “username” que normalmente viene con el campo “token claim name” como “preferred_username”, debemos cambiar a “user_username”



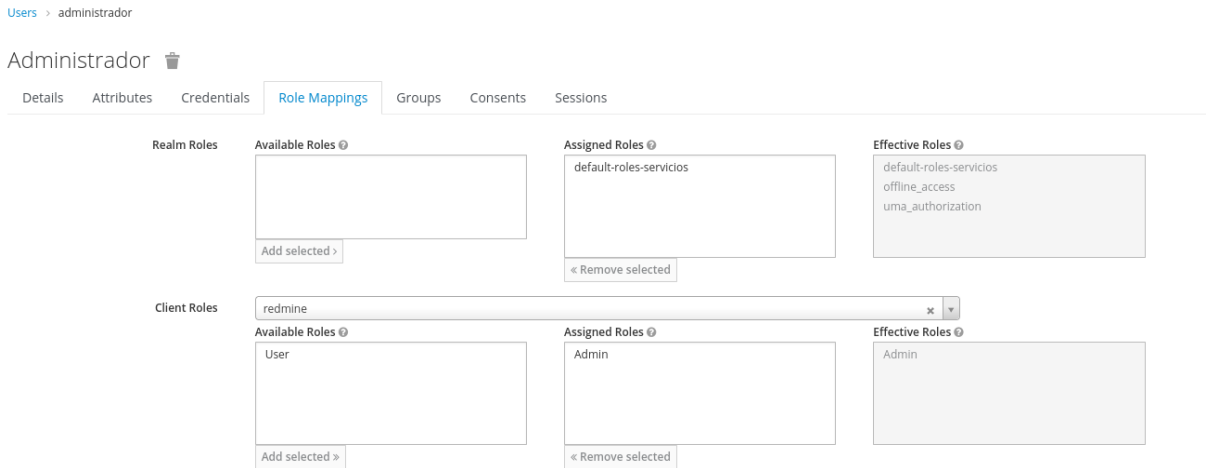
- Y crear uno nuevo llamado “member of”



- En scope debemos añadir “offline_access” y “uma_authorization” al “realm roles”

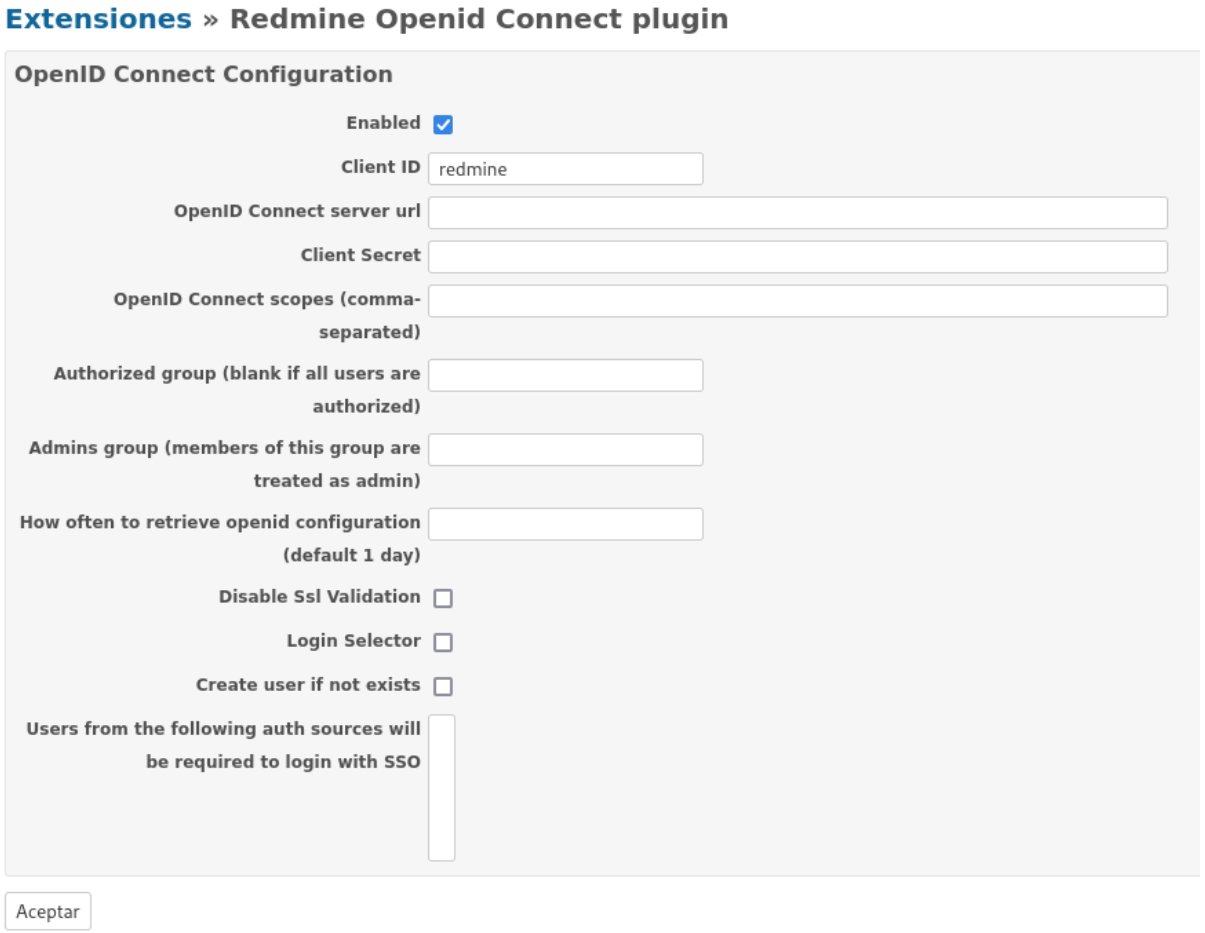


- Ya hemos terminado con el cliente, ahora para añadir a nuestro usuario Administrador entramos al usuario creado en “Role Mappings”, en el desplegable de “Client Roles” ponemos nuestro cliente y seleccionamos en este caso el rol de administrador.

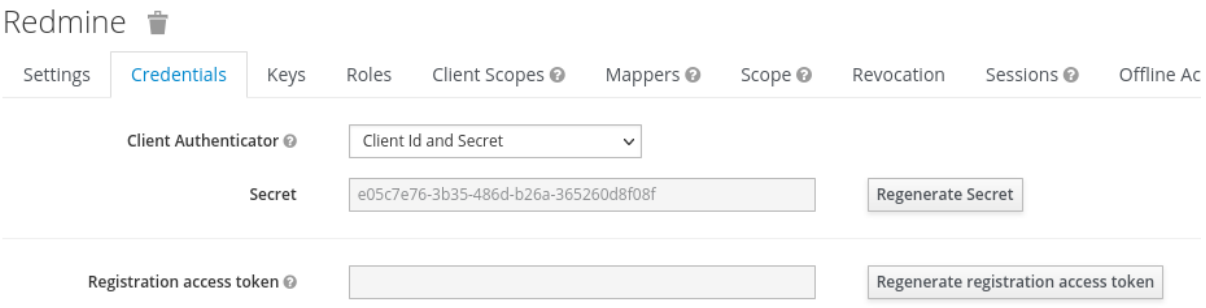


4.4 Configuración de OpenId Generic en Redmine

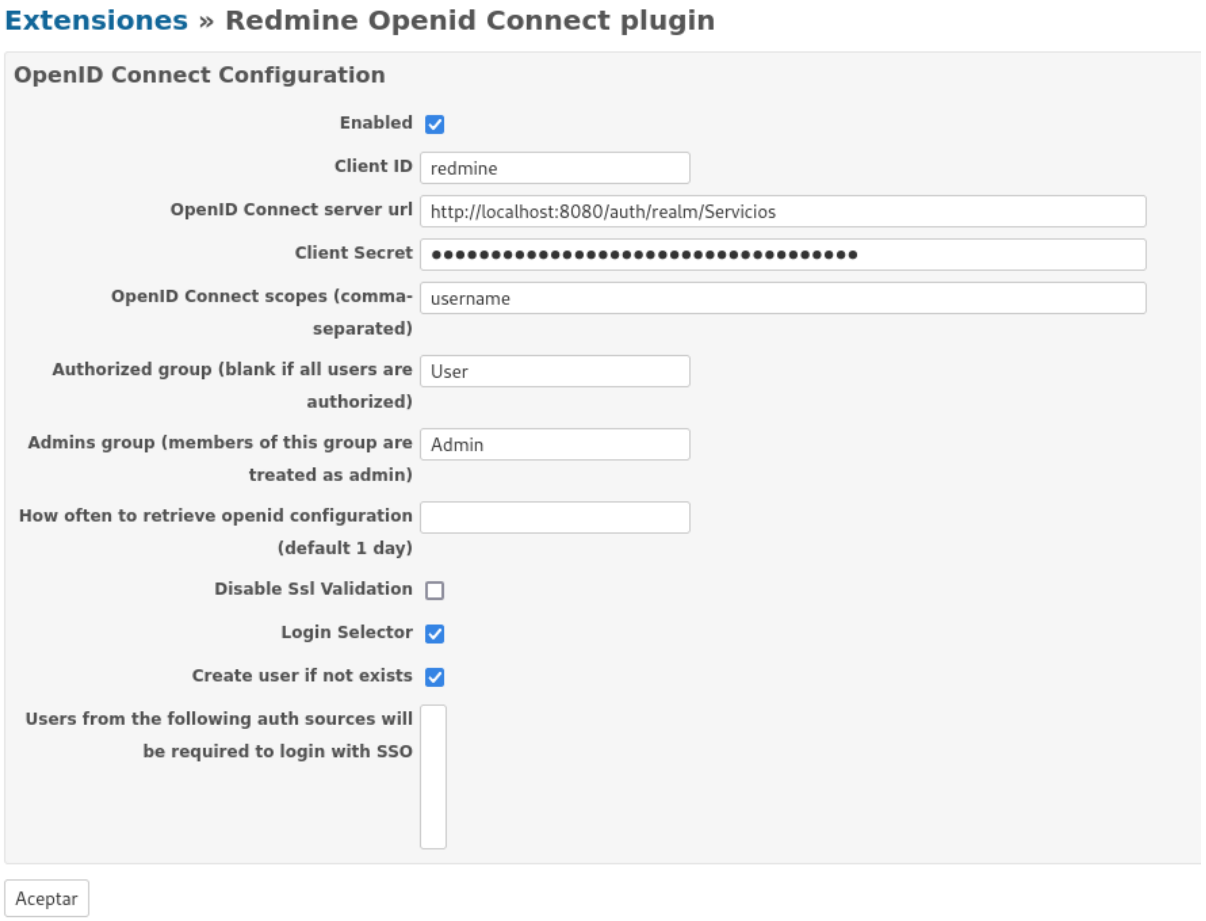
- Vamos a configurar nuestro OpenId con el plugin qué instalamos anteriormente. Entramos y aparecerá esta página:



- Empecemos poco a poco, en primer lugar necesitamos el “Client Secret”, esto es una clave que sacamos de nuestro cliente en keycloak, en el apartado “Credentials”

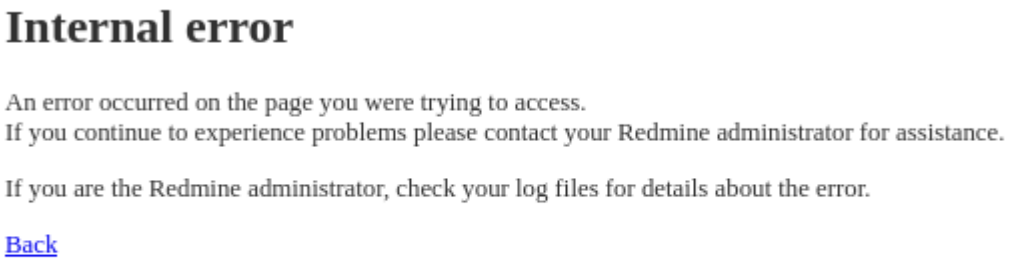


- Lo pegamos en “client secret” y rellenamos los demás campos, en “server url” la dirección del servidor keycloak, en “authorized group” uno para los usuarios normales a los que calificamos como rol en keycloak por “User” y en el otro a los administradores, a quienes calificamos como “Admin”



- Una vez hecho esto salimos y vemos que nos aparece la opción de loggearnos con SSO.

- Sin embargo, hay un problema y no puedo loggearme.



Internal error

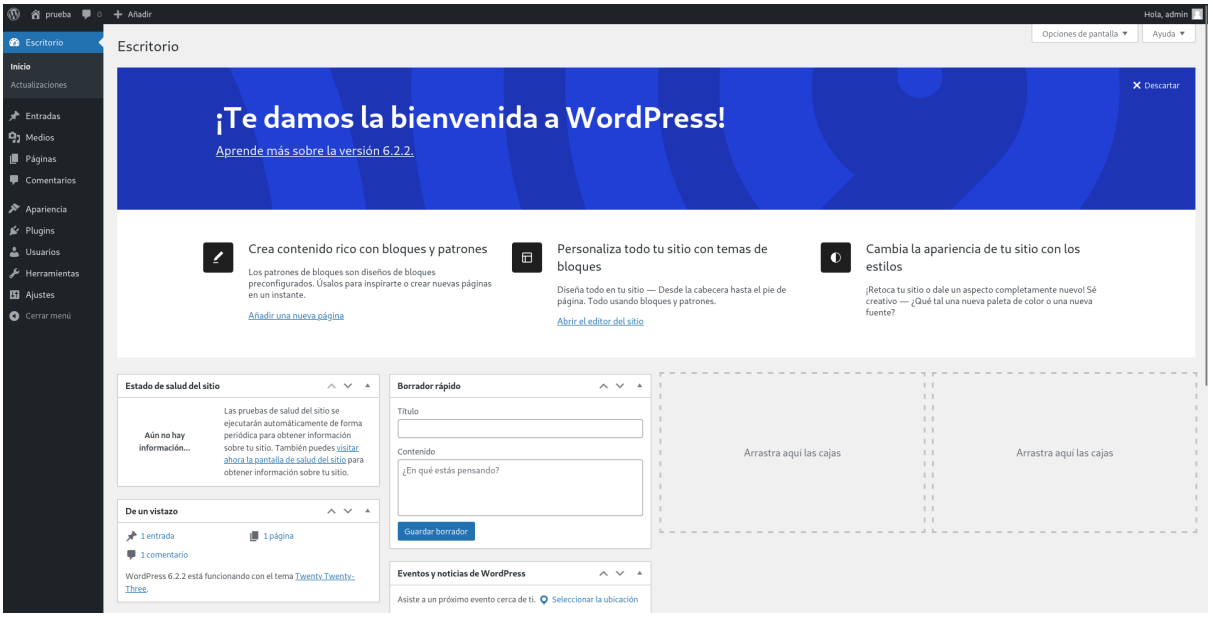
An error occurred on the page you were trying to access.
If you continue to experience problems please contact your Redmine administrator for assistance.

If you are the Redmine administrator, check your log files for details about the error.

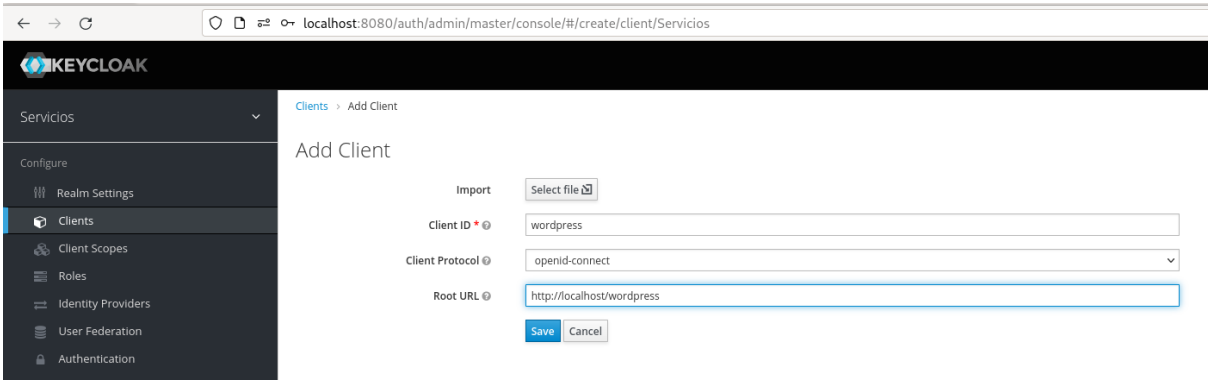
Back

4.5 Cliente wordpress en keycloak

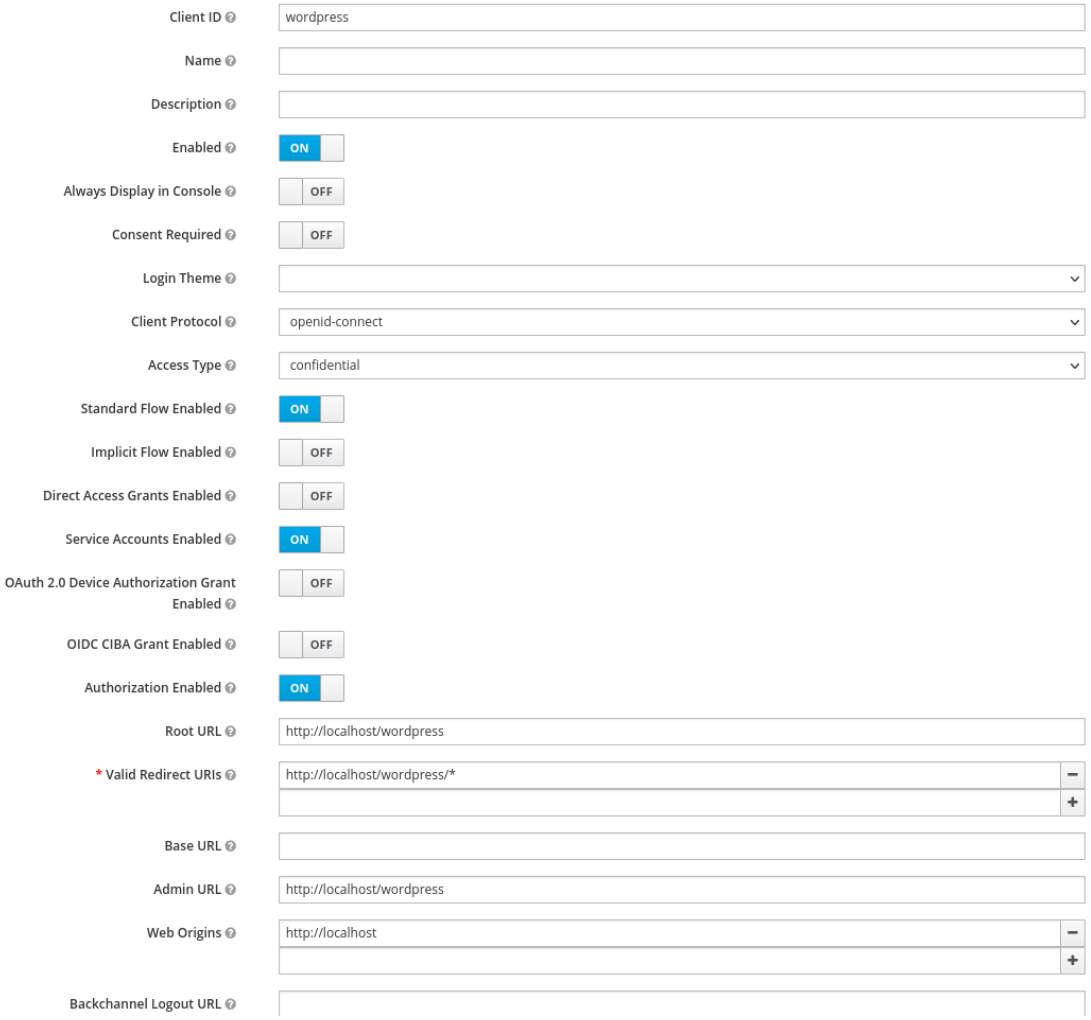
- Primero instalaremos un servidor wordpress normal y corriente.



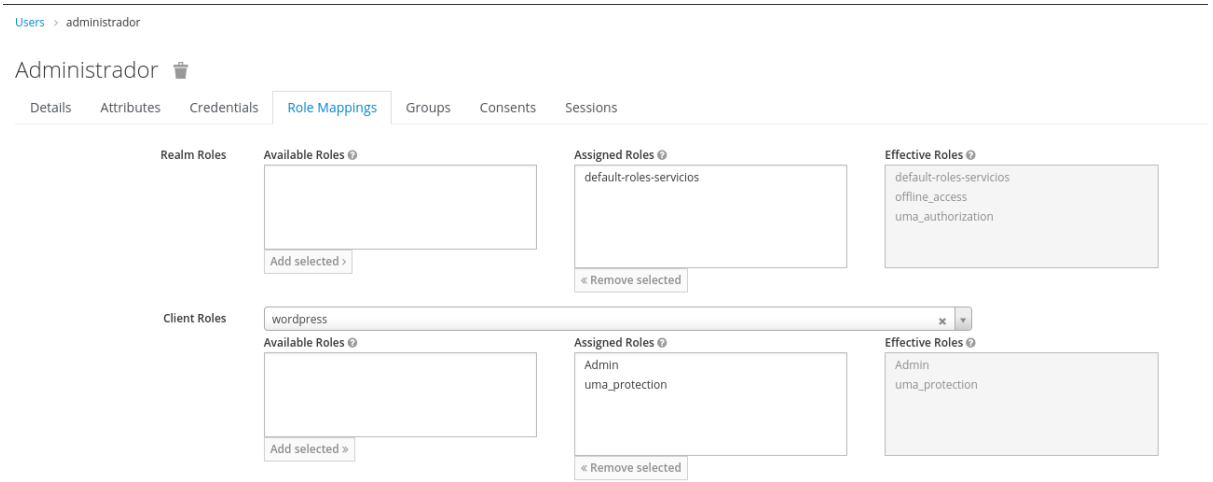
- Una vez instalado vamos a crear el cliente de keycloak:



- Esta será la configuración principal:

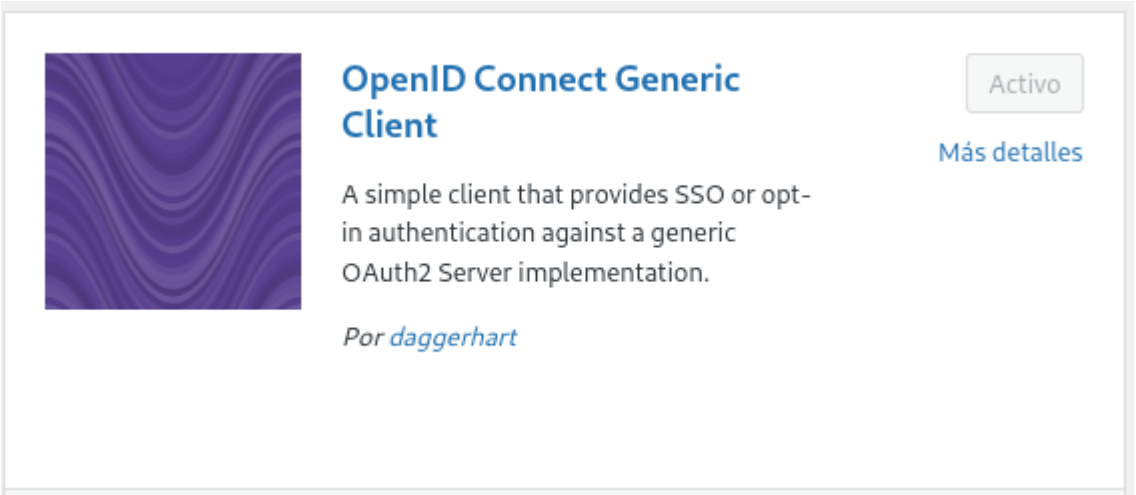


- Nuevamente crearemos un rol Admin y User, aunque aquí no nos servirá tanto como en redmine. Ya qué como veremos más adelante, no podremos especificar qué usuarios serán administradores y cuáles no en wordpress. Sin embargo, sí que podemos añadir el usuario al cliente wordpress.

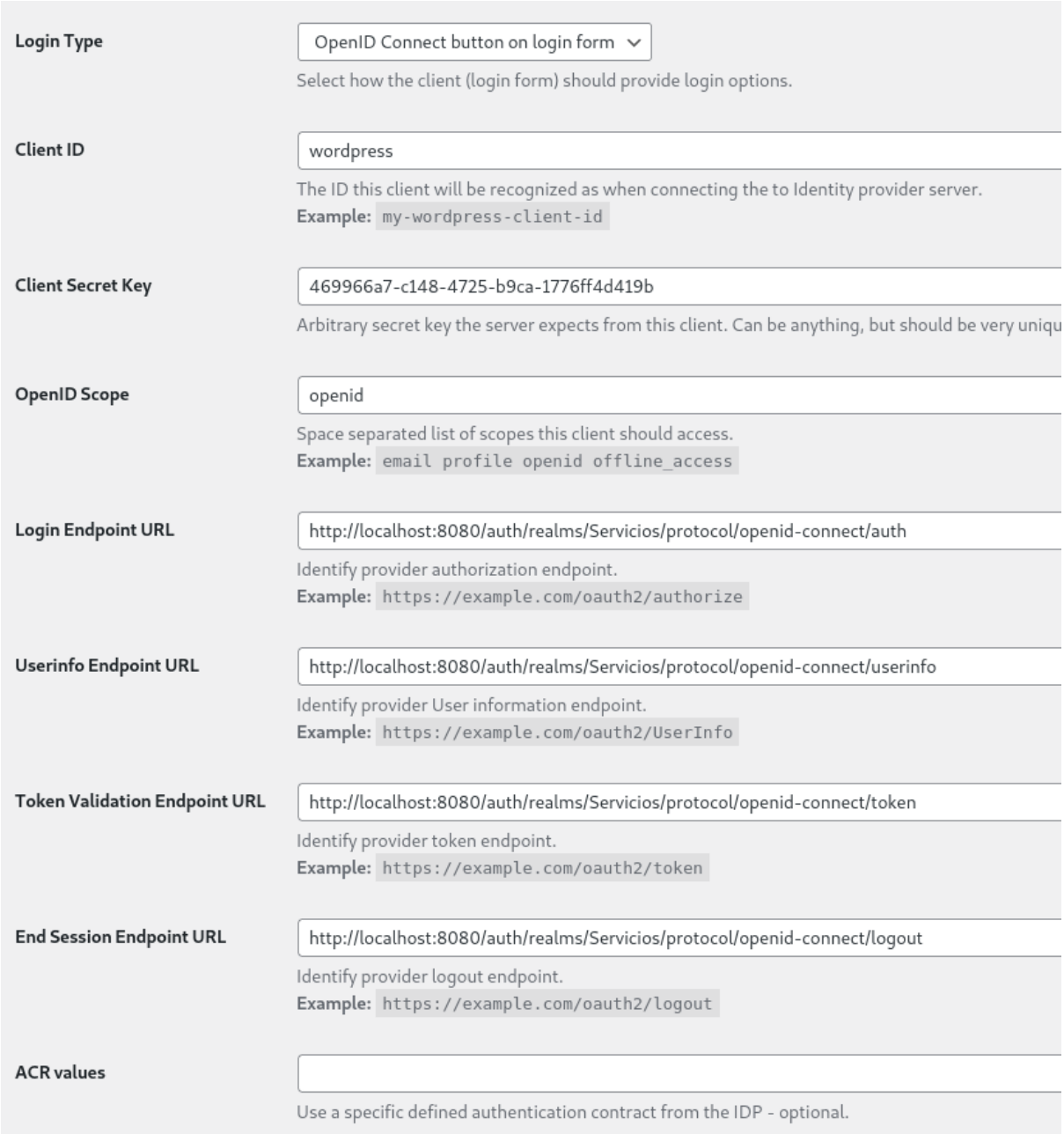


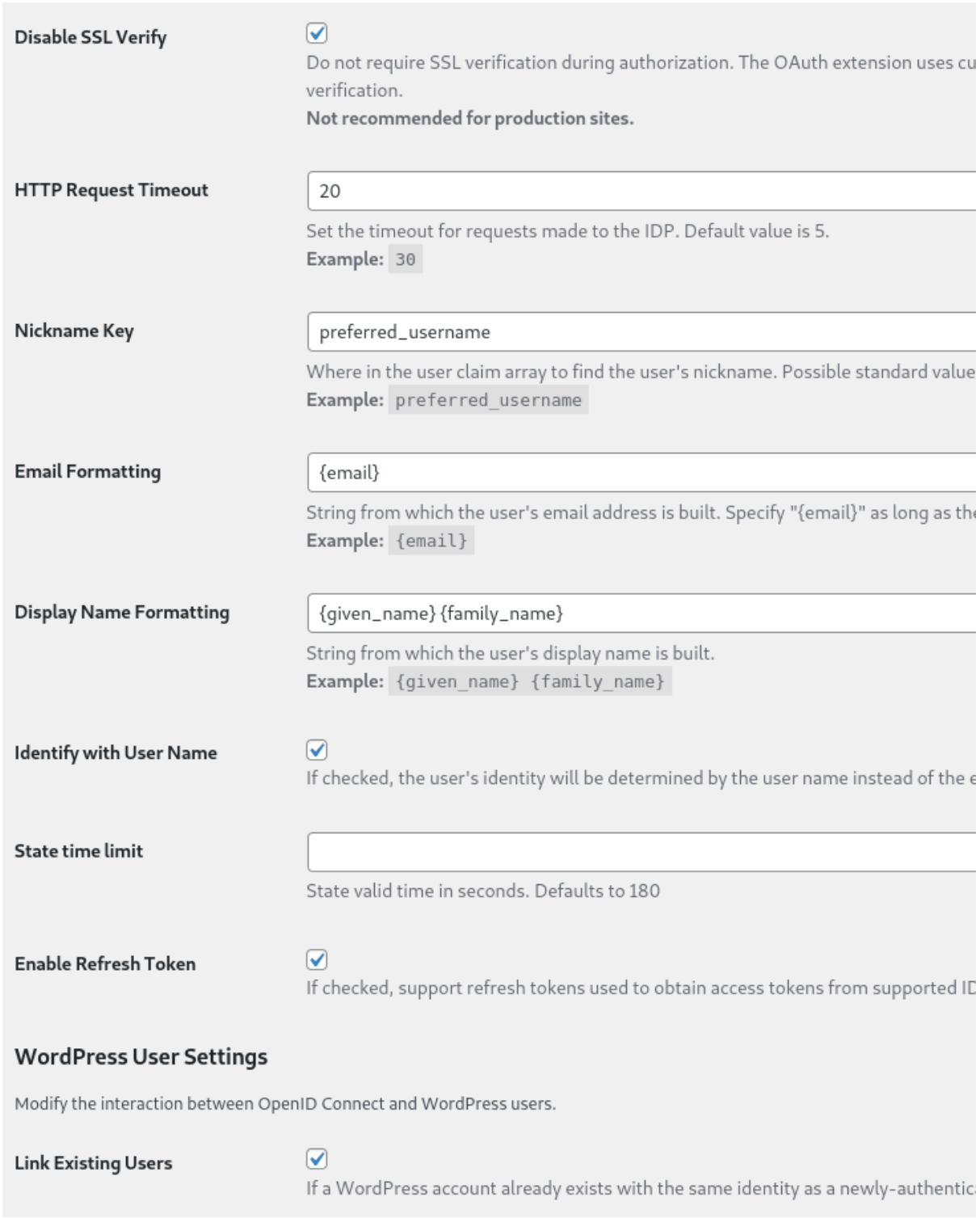
4.6 Instalación y configuración del plugin OpenId en wordpress

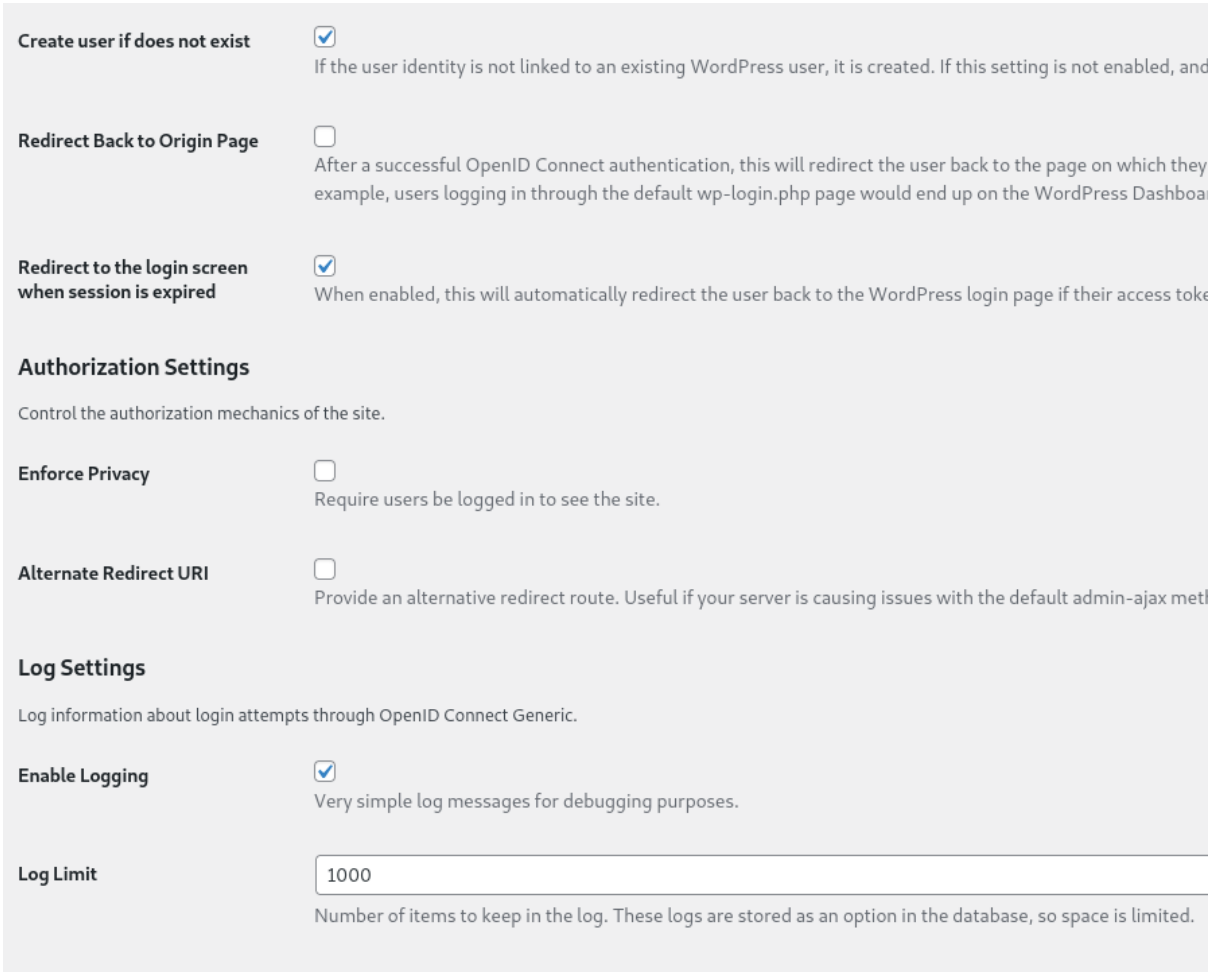
- Ya hemos configurado el cliente de keycloak, ahora vamos a instalar el plugin de OpenId en wordpress que nos permitirá conectar nuestro keycloak. Para ello vamos a “plugins > añadir nuevo” y buscamos e instalamos el siguiente plugin:



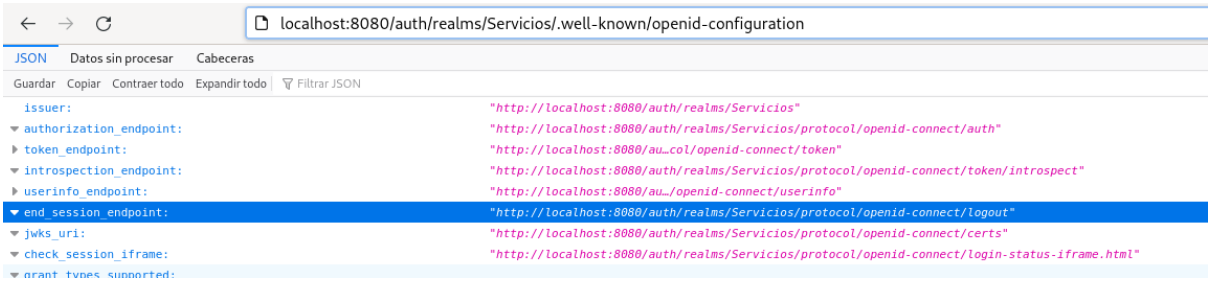
- Una vez activado nos vamos a “Ajustes > OpenId Connect Client” y ahí podremos configurar nuestro OpenId para que se conecte a keycloak. Vamos a ver la configuración:




Disable SSL Verify
Do not require SSL verification during authorization. The OAuth extension uses cu
verification.
Not recommended for production sites.
HTTP Request Timeout
20
Set the timeout for requests made to the IDP. Default value is 5.
Example: 30
Nickname Key
preferred_username
Where in the user claim array to find the user's nickname. Possible standard value
Example: preferred_username
Email Formatting
{email}
String from which the user's email address is built. Specify "{email}" as long as th
Example: {email}
Display Name Formatting
{given_name} {family_name}
String from which the user's display name is built.
Example: {given_name} {family_name}
Identify with User Name
If checked, the user's identity will be determined by the user name instead of the e
State time limit
State valid time in seconds. Defaults to 180
Enable Refresh Token
If checked, support refresh tokens used to obtain access tokens from supported ID
WordPress User Settings
Modify the interaction between OpenID Connect and WordPress users.
Link Existing Users
If a WordPress account already exists with the same identity as a newly-authentic

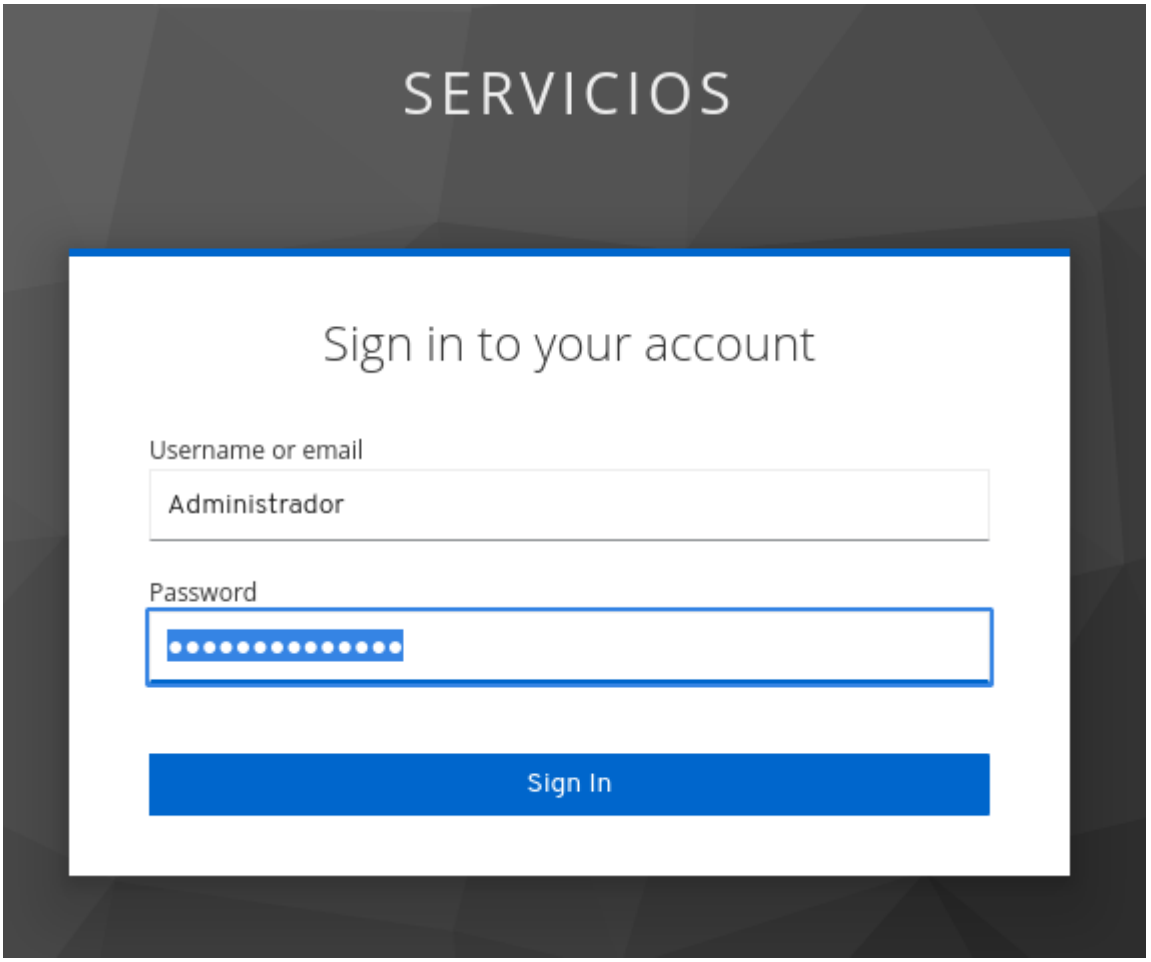


- Recordemos que el “cliente secret” lo sacamos del cliente que hemos creado en nuestro realm de keycloak en el apartado “credentials” y que las URL las sacaremos de “http://localhost:8080/auth/realms/Servicios/.well-known/openid-configuration”

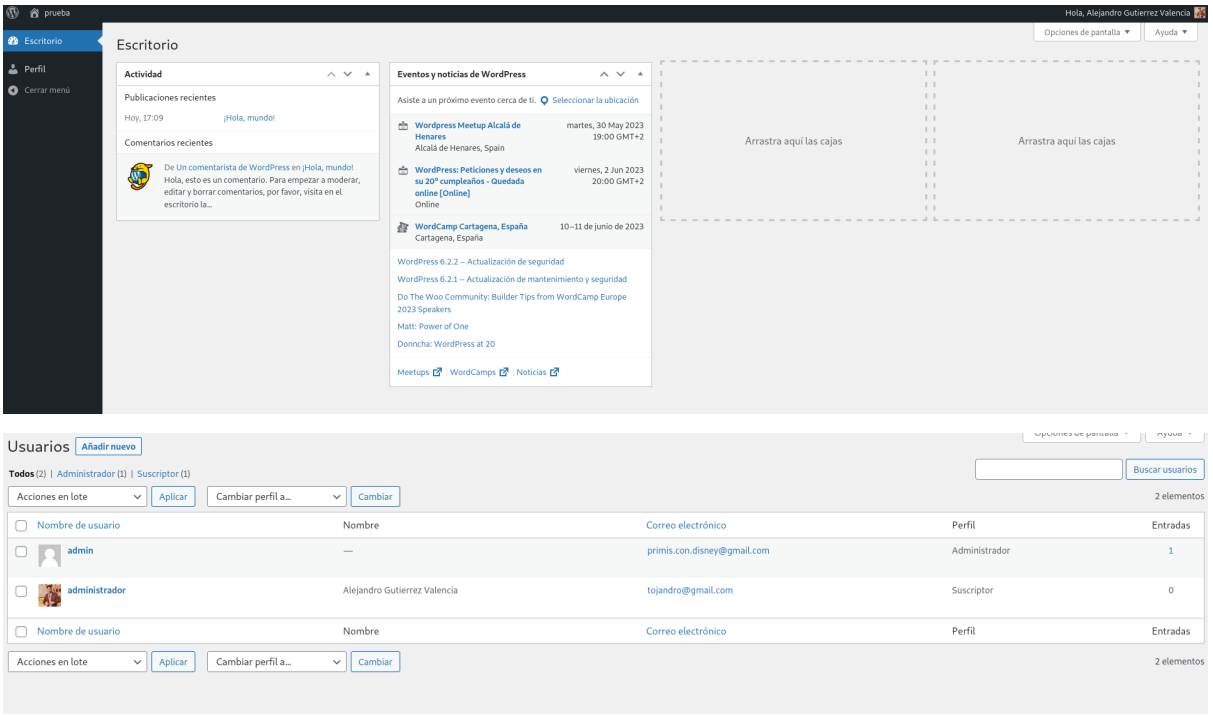


- Una vez hecho esto podemos salir y loggearnos con la opción de OpenId

- Se nos abrirá una pestaña de keycloak para introducir nuestras credenciales:



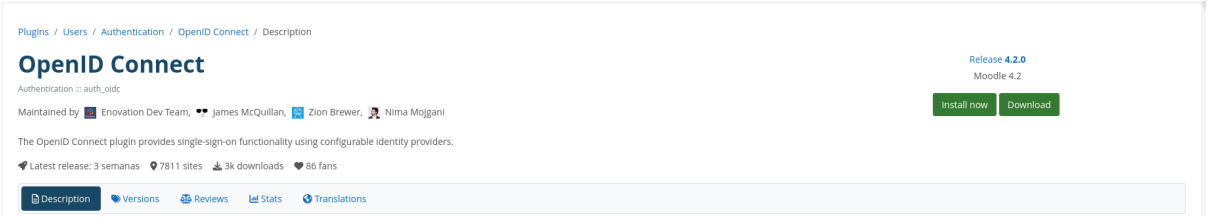
- Ahora ya estamos dentro. Y podemos ver que se ha creado un usuario nuevo.



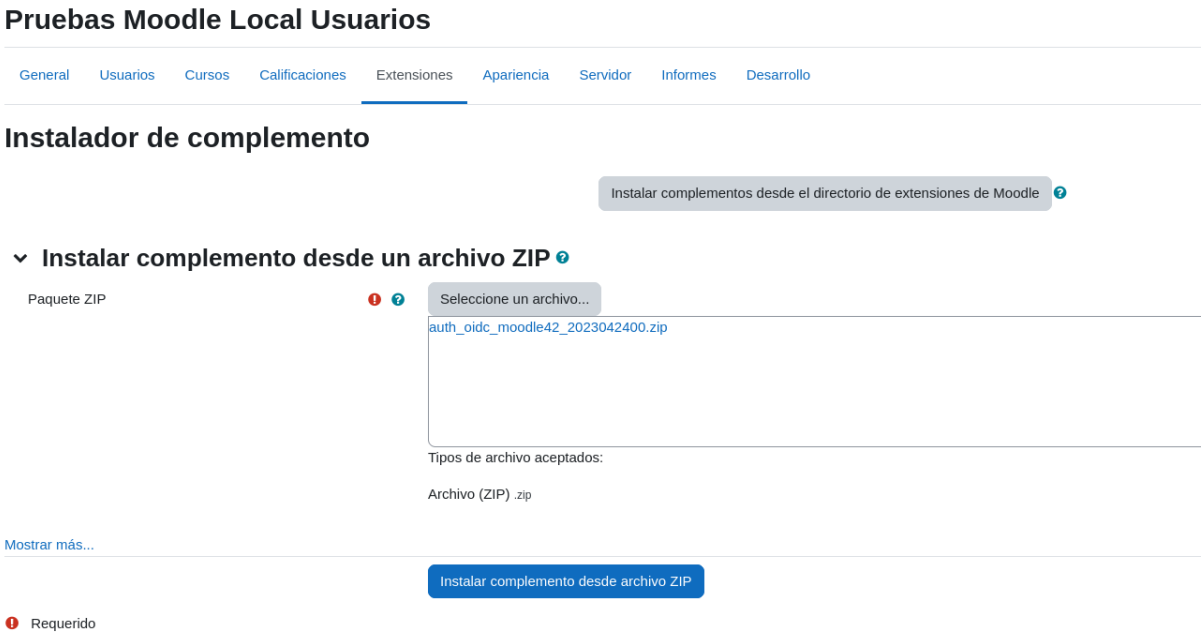
- Aunque vemos que no es administrador. Para que lo fuese deberíamos entrar como un usuario administrador y añadir este usuario específico como administrador.

4.7 Instalación y configuración de OpenID en moodle.

- Descargamos el plugin de [la página oficial de moodle]

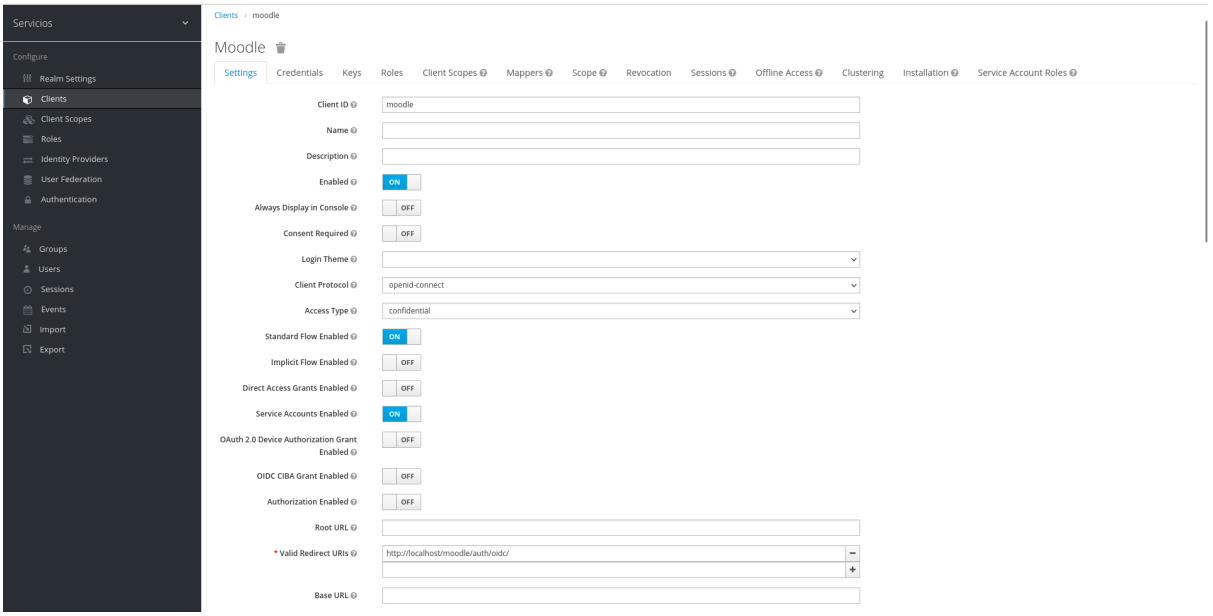


- Lo instalamos en nuestro sitio moodle, para ello nos dirigimos a “Zona de administración > Extensiones > Instalar complemento” y arrastramos nuestro plugin aquí.

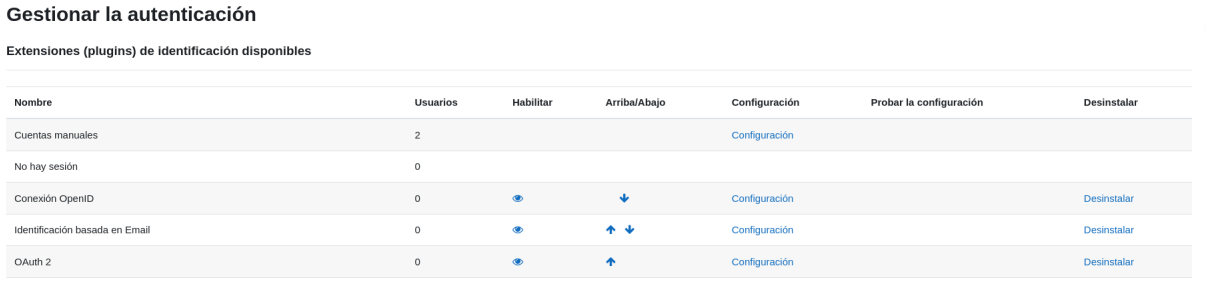


- Ahora tendremos que realizar la configuración de nuestro OpenID, rellenando los datos tal como lo hicimos en los otros dos servicios.

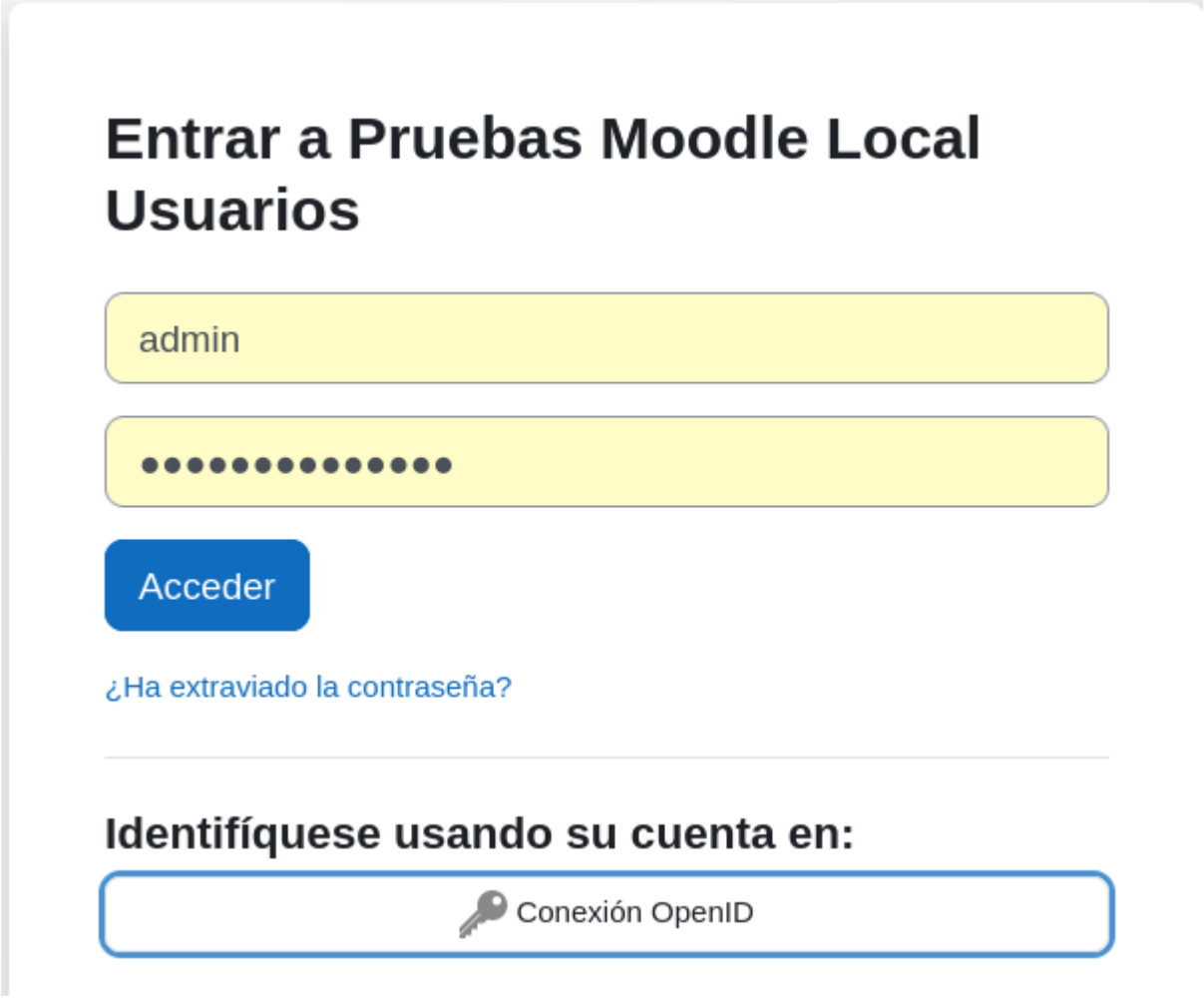
- Por supuesto hemos creado un cliente nuevo en keycloak.



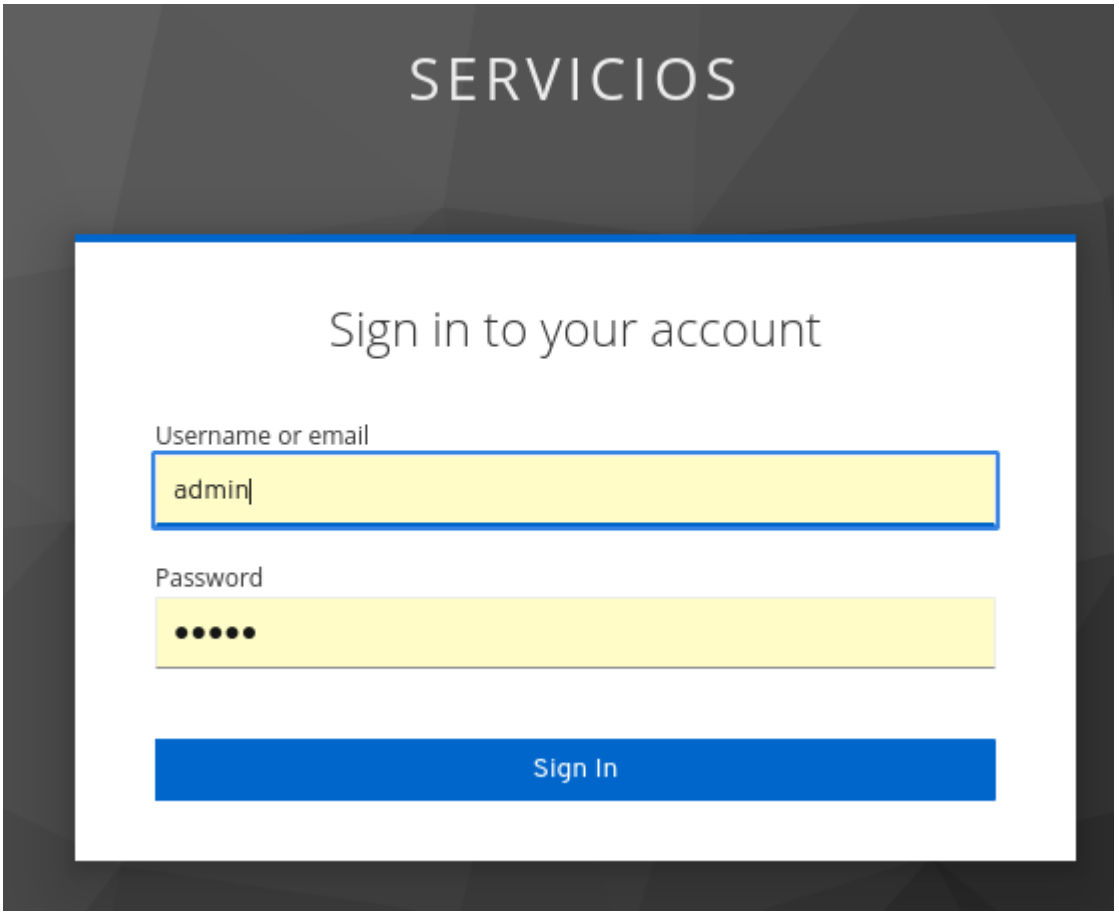
- Nos dirigimos de nuevo a extensiones y esta vez a gestionar autenticación, y pondremos OpenID como nuestra primera opción al loggearnos.



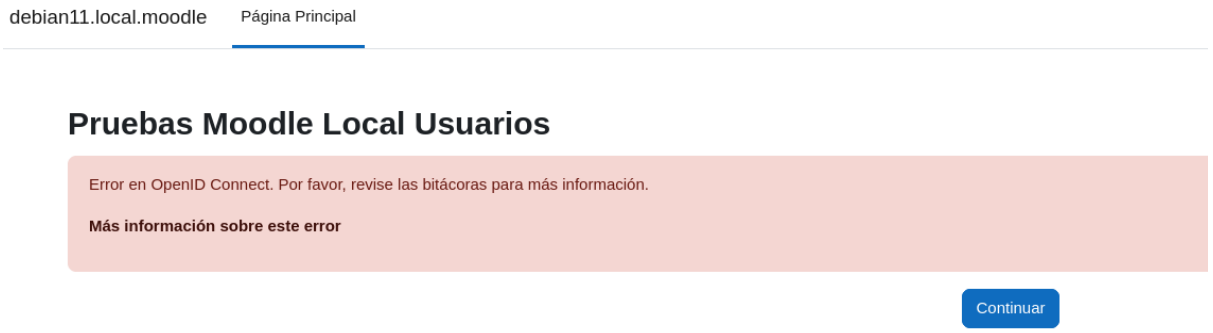
- Ahora nos desloggeamos y tratamos de volver a loggearnos con OpenID.



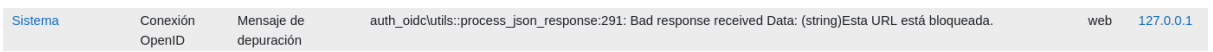
- Aparecerá la página de keycloak.



- Aunque vemos que no es posible la conexión.



- Si miramos el log encontramos este error.

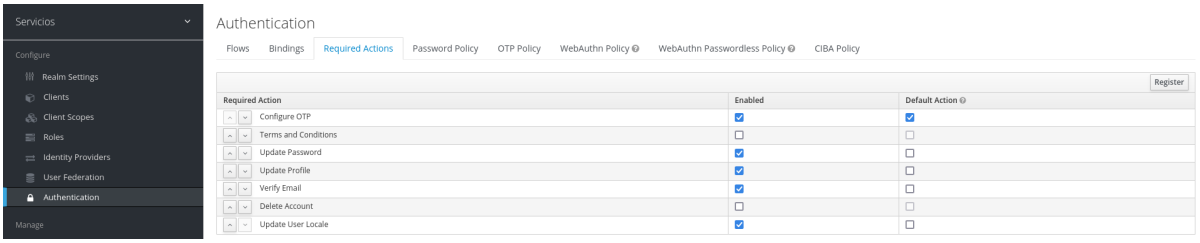


- El error es debido a una de las URL de redirección de keycloak, la cual no existe en este servicio, esto se debe a que el plugin de OpenID para Moodle está pensado para implantarse con Microsoft 365.

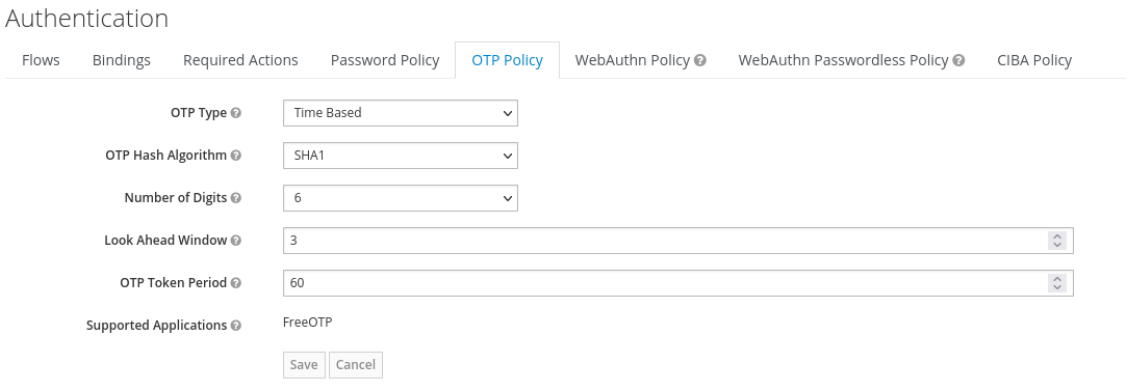
4.8 Implementación de factor de doble verificación

4.8.1 Configuración FreeOTP en keycloak

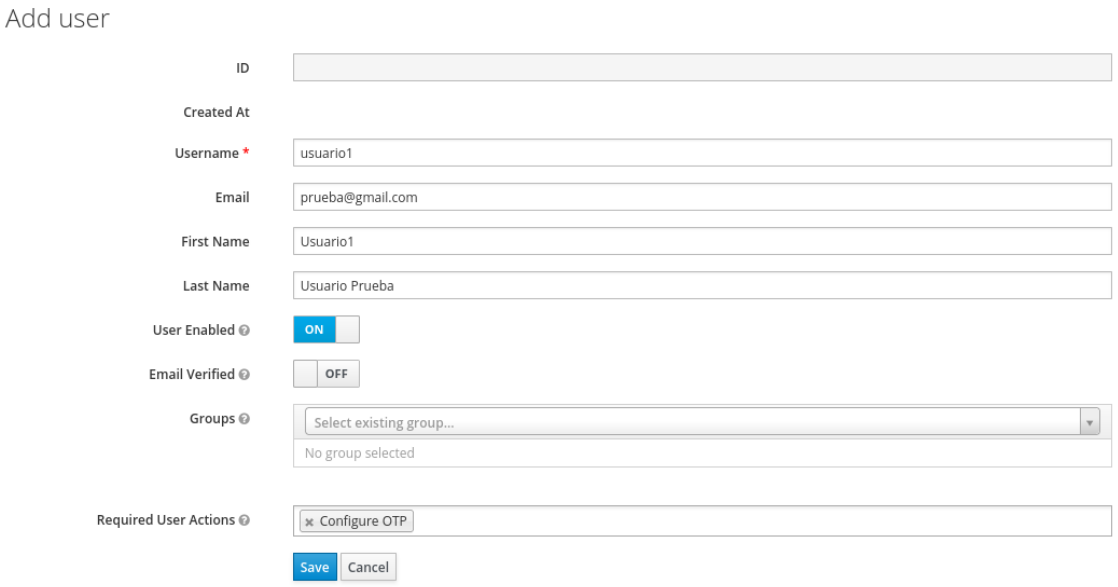
- Primero realizamos la configuración keycloak. Para ello nos vamos al panel de administración y nos dirigimos a “Athentication > Required Actions” y en “Configure OTP” seleccionamos las dos casillas, para habilitarlo y para añadir las opciones por defecto.



- Ahora configuraremos a nuestro gusto esta herramienta cambiando a la pestaña “OTP Policy”, en mi caso está basado en el tiempo, el token dura 60 segundos se podrá usar en 3 pestañas a la vez y serán 6 dígitos. Por supuesto el algoritmo de encriptación estará basado en SHA1.



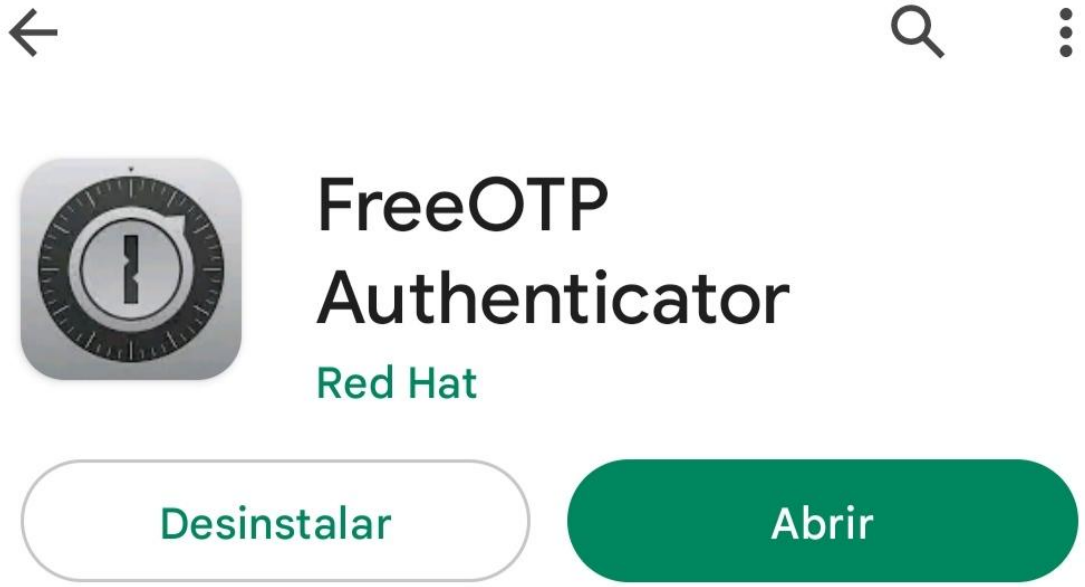
- Ahora debemos implementar esta funcionalidad en nuestros usuarios, vamos a crear uno nuevo para ver cómo se configuraría. Como vemos en “Required User Actions” hemos añadido la opción Configure OTP.



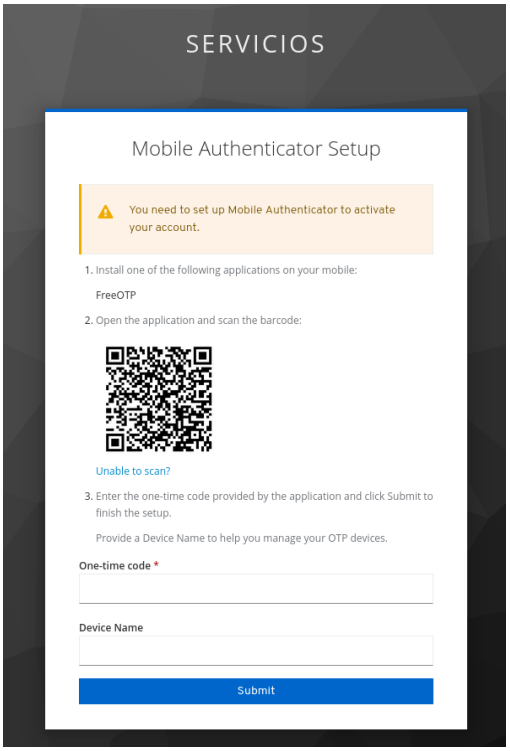
- Tenemos nuestro usuario configurado con OTP, ahora debemos instalar nuestra aplicación de verificación y ver una prueba.

4.8.2 Instalación de aplicación de verificación y prueba

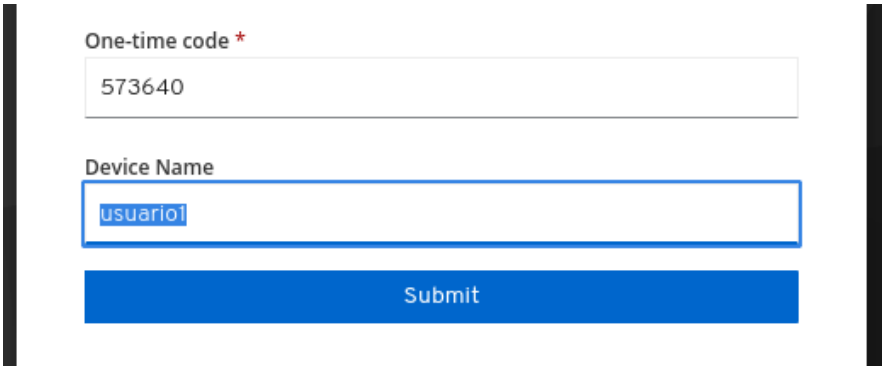
- Debemos instalar una aplicación la cual nos servirá para obtener el código de verificación correspondiente. Se llama “FreeOTP Authentificator”.



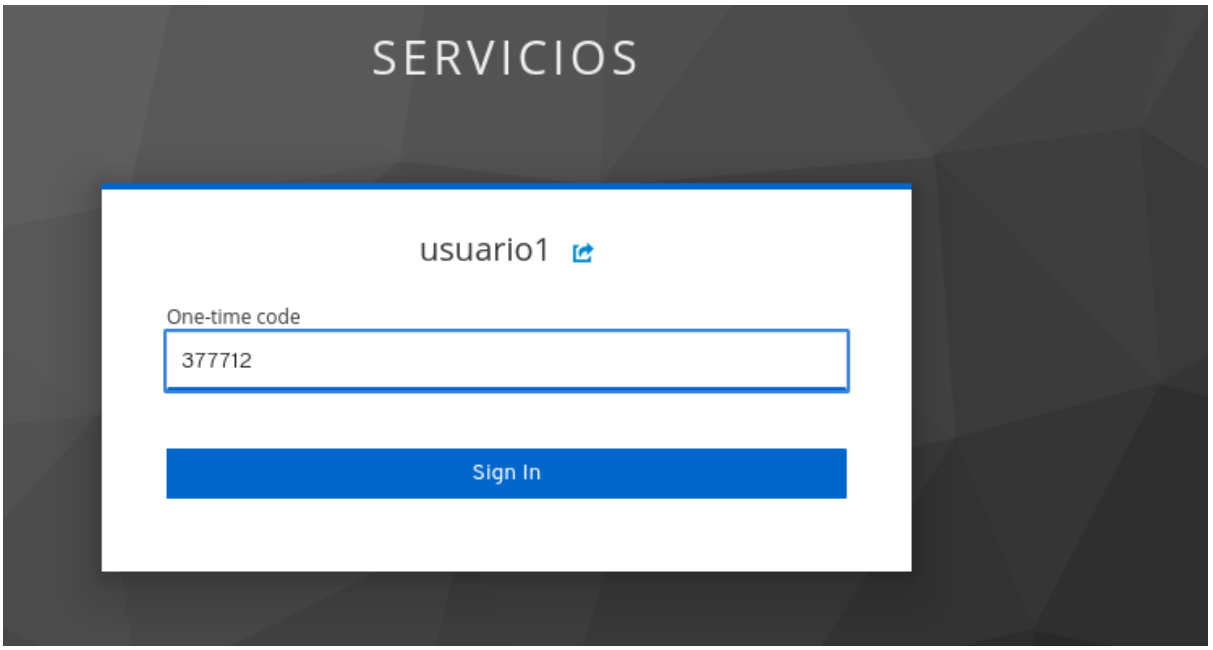
- Tenemos un problema y es que por seguridad la aplicación no permite realizar capturas de pantalla dentro de la misma, sin embargo el proceso es muy sencillo.



- Con la aplicación pulsaremos en un icono “+” y escaneamos el código qr qué aparece y nos aparecerá directamente un código, lo escribimos y ya estamos loggeados.



- Si nos conectamos de nuevo no tendremos que escanear el código, simplemente entramos en nuestra aplicación y tapeamos el código que habrá cambiado.



5. Conclusiones y propuestas

Sin duda queda mucho por explorar en todas las herramientas, apenas he rascado la superficie de las posibilidades técnicamente de Keycloak y sus funcionalidades.

En conclusión. Keycloak es un servicio más complejo de lo que imaginaba al comenzar que, para exprimir su funcionalidad, depende de otras muchas pequeñas piezas que van dando forma poco a poco a lo que queremos conseguir. Que es: tener una gestión de usuarios y accesos centralizada de nuestros diferentes servicios.

Para realizar esta labor hay que profundizar en las diferentes herramientas intermedias de las que hemos hablado como: SAML2, OpenId, Oauth2. Todas tienen sus ventajas e inconvenientes.

6. Bibliografía

6.1 Keycloak

https://youtu.be/W38qJEodeKs?list=PL4bT56Uw3S4wEZ0Sp7jrGAX8DMS-MKowg

https://www.enmilocalfunciona.io/acelerando-los-desarrollos-con-contenedores-keycloak/

https://www.enmilocalfunciona.io/acelerando-los-desarrollos-con-contenedores-keycloak-parte-2/

https://blog.desdelinux.net/keycloak-una-solucion-de-gestion-de-acceso-e-identidad-de-codigo-abierto/

https://www.chakray.com/es/como-utilizar-keycloak-proveedor-identidades-wso2-api-manager/

https://apim.docs.wso2.com/en/latest/administer/key-managers/configure-keycloak-connector/

https://cloudinfrastructureservices.co.uk/install-keycloak-sso-on-ubuntu-20-04/

6.2 REDMINE

https://stackoverflow.com/questions/75984580/redmine-openid-conncet-with-keycloak

https://labarta.es/como-instalar-redmine-usando-docker-compose/

https://github.com/Contargo/redmine_oidc

https://github.com/devopskube/redmine_openid_connect

http://devopsku.be/setup/redmine-keycloak/

https://chachocool.com/como-instalar-redmine-en-debian-11-bullseye/

6.3 MOODLE

https://stackoverflow.com/questions/69490574/how-to-connect-moodle-with-keycloak

https://plugins.miniorange.com/moodle-saml-single-sign-on-sso-for-keycloak

https://moodle.org/plugins/auth_oidc

https://docs.moodle.org/310/en/Microsoft_365#Plugin_Installation

6.4 WORDPRESS

https://westergaard.eu/2018/05/sso-lets-talk-about-single-sign-on-for-wordpress-and-gitlab-using-keycloak-redhat-sso-featuring-special-guest-duo-security-for-2fa/

https://dev.to/vishalraj82/using-https-in-docker-for-local-development-nc7

https://www.rubenortiz.es/2022/02/18/habilitar-https-en-wordpress-local-con-docker/

6.5 OAUTH2

https://linuxhint.com/oauth-linux-installation/

https://www.albertcoronado.com/2020/09/24/login-en-tus-aplicaciones-de-microservicios-con-oauth2-proxy/

6.6 OPENID CONNECT

https://www.youtube.com/watch?v=nNVlewjKQEQ

6.7 Factor de doble verificación OTP.

https://ultimatesecurity.pro/post/2fa/

https://mailchimp.com/es/help/set-up-a-two-factor-authentication-app-at-login/

https://repositorio.unican.es/xmlui/bitstream/handle/10902/26830/444290.pdf?sequence=1