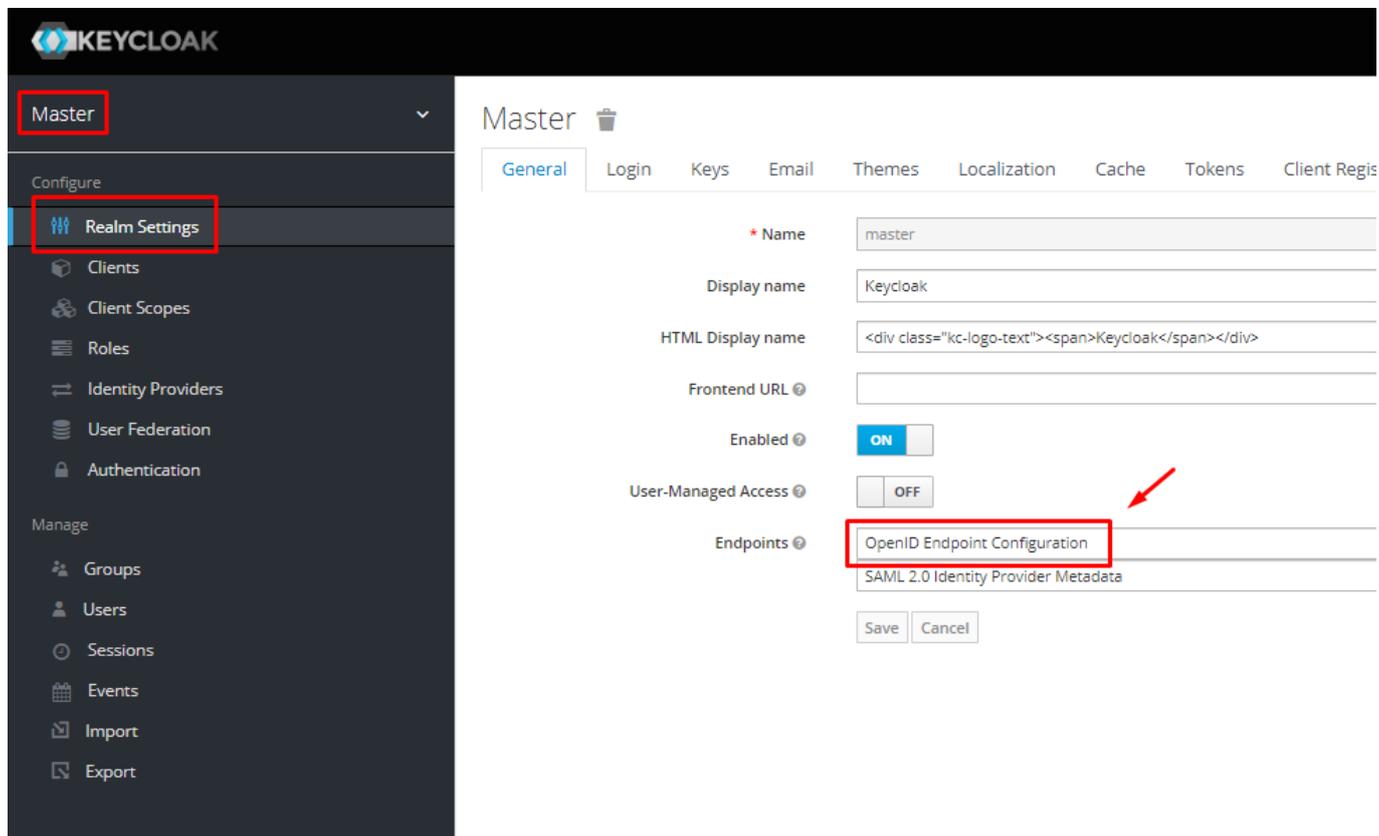


_1. Conocer los endpoint de **OpenID OIDC** en **Keycloak**, para esto seleccionamos el realm «**master**», Configure «**Realm Settings**» y en la pestaña «**General**» damos clic donde dice «**OpenID Endpoint Configuration**»



The screenshot shows the Keycloak Admin Console interface. On the left sidebar, the 'Master' realm is selected, and 'Realm Settings' is highlighted under the 'Configure' section. The main content area shows the configuration for the 'master' realm in the 'General' tab. The 'Endpoints' section is expanded, and 'OpenID Endpoint Configuration' is highlighted with a red box and a red arrow pointing to it. Other visible fields include 'Name' (master), 'Display name' (Keycloak), 'HTML Display name' (HTML template), 'Frontend URL', 'Enabled' (ON), and 'User-Managed Access' (OFF).

Se abrirá una URL

<http://localhost:8080/realms/master/.well-known/openid-configuration>

```
1 // 20220708233925
2 // http://localhost:8080/realms/master/.well-known/openid-configuration
3
4 {
5   "issuer": "http://localhost:8080/realms/master",
6   "authorization_endpoint": "http://localhost:8080/realms/master/protocol/openid-connect/auth",
7   "token_endpoint": "http://localhost:8080/realms/master/protocol/openid-connect/token",
8   "introspection_endpoint": "http://localhost:8080/realms/master/protocol/openid-connect/token/introspect",
9   "userinfo_endpoint": "http://localhost:8080/realms/master/protocol/openid-connect/userinfo",
10  "end_session_endpoint": "http://localhost:8080/realms/master/protocol/openid-connect/logout",
11  "frontchannel_logout_session_supported": true,
12  "frontchannel_logout_supported": true,
13  "jwks_uri": "http://localhost:8080/realms/master/protocol/openid-connect/certs",
14  "check_session_iframe": "http://localhost:8080/realms/master/protocol/openid-connect/login-status-iframe.html",
15  "grant_types_supported": [
16    "authorization_code",
17    "implicit",
18    "refresh_token",
19    "password",
20    "client_credentials",
21    "urn:ietf:params:oauth:grant-type:device_code",
22    "urn:openid:params:grant-type:ciba"
23  ],
24  "acr_values_supported": [
25    "0",
26    "1"
27  ],
28  "response_types_supported": [
29    "code",
30    "none",
31    "id_token",
32    "token"
33  ]
34 }
```

_2. Nos vamos a **Postman** y consultamos el token, donde:

- **Token method:** POST
- **Token endpoint:**
http://localhost:8080/realms/master/protocol/openid-connect/token
- **x-www-form-urlencoded**
 - **grant_type:** password
 - **client_id:** admin-cli (es la aplicación cliente registrada en keycloak, se puede crear otra aplicación)
 - **username:** juliopari (previamente creado)
 - **password:** 123456 (previamente creado)

Keycloak / token

POST http://localhost:8080/realms/master/protocol/openid-connect/token

Params Authorization Headers (9) **Body** Pre-request Script Tests Settings Cookies

none form-data **x-www-form-urlencoded** raw binary GraphQL

KEY	VALUE	DESCRIPTION
<input checked="" type="checkbox"/> Content-Type	application/x-www-form-urlencoded	
<input checked="" type="checkbox"/> grant_type	password	
<input checked="" type="checkbox"/> client_id	admin-cli	
<input checked="" type="checkbox"/> username	julio pari	
<input checked="" type="checkbox"/> password	123456	

Body Cookies Headers (11) Test Results Status: 200 OK Time: 76 ms Size: 2.46 KB Save Response

Pretty Raw Preview Visualize JSON

```
1  
2  
3  
4  
5  
6  
7  
8  
9  
10  
"access_token": "eyJhbGciOiJIUzI1NiIsInR5cCI6IWRX",  
"refresh_token": "eyJhbGciOiJIUzI1NiIsInR5cCI6IWRX",  
"expires_in": 60,  
"refresh_expires_in": 1800,  
"token_type": "Bearer",  
"not-before-policy": 0,  
"session_state": "4fe38be-3438-492e-93f6-584f078de48f",  
"scope": "email profile"
```

_3. Copiamos el valor de «**access_token**» que es un **JSON Web Token** y nos vamos a **jwt.io** para abrirlo

