### Documentation: GitLab OAuth MCP Tool

#### ****Introduction****

This document describes the GitLab OAuth MCP Tool, an MCP server that provides two tools for GitLab OAuth:

1. Generating an authorization URL for user login.
2. Exchanging an authorization code for an access token.

The tool integrates with the Model Context Protocol (MCP) and enables seamless OAuth-based authentication for applications using GitLab.

### ****Prerequisites****

* Python 3.10 or higher.
* MCP SDK installed (mcp[cli]).
* Required Python libraries: requests, dotenv.
* GitLab OAuth credentials:
  + GITLAB\_CLIENT\_ID
  + GITLAB\_CLIENT\_SECRET
  + GITLAB\_REDIRECT\_URI

### ****Setup****

#### 1. ****Environment Setup****

Create a new project directory and initialize it:

uv init gitlab\_oauth

cd gitlab\_oauth

uv venv

.venv\Scripts\activate

uv add mcp[cli] requests python-dotenv

#### 2. ****Environment Variables****

Create a .env file in the project directory with the following content:

GITLAB\_CLIENT\_ID=your\_client\_id

GITLAB\_CLIENT\_SECRET=your\_client\_secret

GITLAB\_REDIRECT\_URI=your\_redirect\_uri

Replace your\_client\_id, your\_client\_secret, and your\_redirect\_uri with your GitLab credentials.

#### 3. ****Code File****

Save the following code as gitlab\_oauth.py:

[Refer to the complete code in the previous section.]

### ****How It Works****

#### 1. ****Authorization URL Generation****

The get\_authorization\_url tool generates a URL that users can visit to log in with GitLab. The URL includes:

* Client ID
* Redirect URI
* State parameter (to prevent CSRF attacks)
* Scope (read\_user by default)

#### 2. ****Access Token Exchange****

The exchange\_code\_for\_token tool exchanges the authorization code (provided by GitLab after user login) for an access token. This token can be used to access GitLab APIs.

### ****Using the MCP Tools****

#### ****Run the Server****

Start the server with:

uv run gitlab\_oauth.py

#### ****Generate Authorization URL****

Use the get\_authorization\_url tool with a state parameter:

mcp-cli call get\_authorization\_url --state=your\_state\_value

* Open the returned URL in a browser.
* Log in with GitLab and authorize the application.
* Note the code parameter in the redirect URI.

#### ****Exchange Code for Token****

Use the exchange\_code\_for\_token tool with the authorization code:

mcp-cli call exchange\_code\_for\_token --code=your\_authorization\_code

The response will include:

* Access token
* Token type
* Expiration time

### ****Configuration for Claude for Desktop****

To integrate the tool with Claude for Desktop:

Locate the configuration file:

* + macOS: ~/Library/Application Support/Claude/claude\_desktop\_config.json
  + Windows: %APPDATA%\Claude\claude\_desktop\_config.json

Add the following configuration:

{

"mcpServers": {

"gitlab\_oauth": {

"command": "uv",

"args": [

"--directory",

"/ABSOLUTE/PATH/TO/gitlab\_oauth",

"run",

"gitlab\_oauth.py"

]

}

}

}

Restart Claude for Desktop.

Verify the gitlab\_oauth tools are available under the hammer icon.

### ****Error Handling****

* **Log File**: Errors and server activities are logged in mcp\_server.log.
* **Common Issues**:
  + **Invalid Redirect URI**: Ensure the GITLAB\_REDIRECT\_URI matches the URI configured in GitLab.
  + **Missing Environment Variables**: Confirm the .env file is correctly set up.
  + **Network Errors**: Check the internet connection and GitLab availability.

### ****Customization****

To expand or modify the tool:

1. Change the scopes in the get\_authorization\_url function to access different GitLab APIs.
2. Add more tools for token refresh, user information, or repository access using GitLab's API.

### ****Conclusion****

This GitLab OAuth MCP Tool simplifies the process of integrating GitLab authentication into your applications. It provides an easy-to-use interface for generating OAuth authorization URLs and obtaining access tokens.