💡 Considerations

This project is specifically designed for applications that adhere to the current **.NET supported versions**. However, the **Feijuca.Auth.Api** module can be accessed from any programming language, as it processes calls against **Keycloak**.

If you intend to use only the API module (for creating users, managing groups, sessions, and tokens), you can do so without having **.NET** in your technology stack, as it is a RESTful API.

On the other hand, if your stack is built with **.NET**, you'll benefit from enhanced capabilities and integration options. As you progress through the documentation, you'll gain a better understanding of these advantages.

🔧 Prerequisites

- An instance of **Keycloak** server.

🖥️ Let's get started!

- 🔧 **Want to use Feijuca.Auth.Api?** Check out the necessary configuration steps here.
- 🏢 **Interested in the Feijuca.Auth.MultiTenancy concept?** Find the necessary configuration steps here *(Available soon)*.

Feijuca.Auth.Api 🍽️🔐

Welcome to **Feijuca.Auth.Api**, a robust API built with .NET designed to simplify the integration and management of users, groups, and permissions using Keycloak. This API enables consumers to easily implement role-based access control (RBAC) in their applications.

📚 Introduction

Feijuca.Auth.Api provides a user-friendly interface to interact with Keycloak endpoints, allowing you to:

- **Create Users**: Add new users quickly and effortlessly.
- **Manage Groups**: Create and delete groups as needed to organize your users.
- **Manage Permissions**: Add and remove permissions from groups, streamlining access control.
- **RBAC Control**: Implement efficient and scalable permission management for your application.
- **Generate JWT Tokens**: Easily generate JWT tokens for user authentication and authorization.
- **Revoke Active Sessions**: Manage user sessions by revoking active sessions when necessary.
- **Create Clients**: Set up clients for your applications, enabling secure access to your API.
- **Create Roles**: Define roles to streamline permission management and assign them to users or groups.
- **Add Roles to Groups**: Manage group permissions by adding roles to specific groups.

🚀 Features

- **Integration with Keycloak**: All operations are performed through Keycloak, ensuring security and efficiency.
- **Preconfigured Setup**: Receive a preconfigured setup and manage your calls to Keycloak easily.
- **Comprehensive Endpoints**: A variety of endpoints available for all management operations.

🛠️ Technologies Used

- .NET
- Keycloak
- RESTful API

📄 Necessary configurations

🔗 **To configure the access to the api, it is necessary do a quickly configuration .**

🔑 Configuring permissions in Keycloak for API Integration

In order for your API to perform operations such as creating users, groups, realms, clients, roles, group roles, and more in Keycloak, you need to configure the appropriate permissions in the realm. This configuration is done by granting specific permissions to the **Service Account** associated with the client your API uses. Follow the steps below:

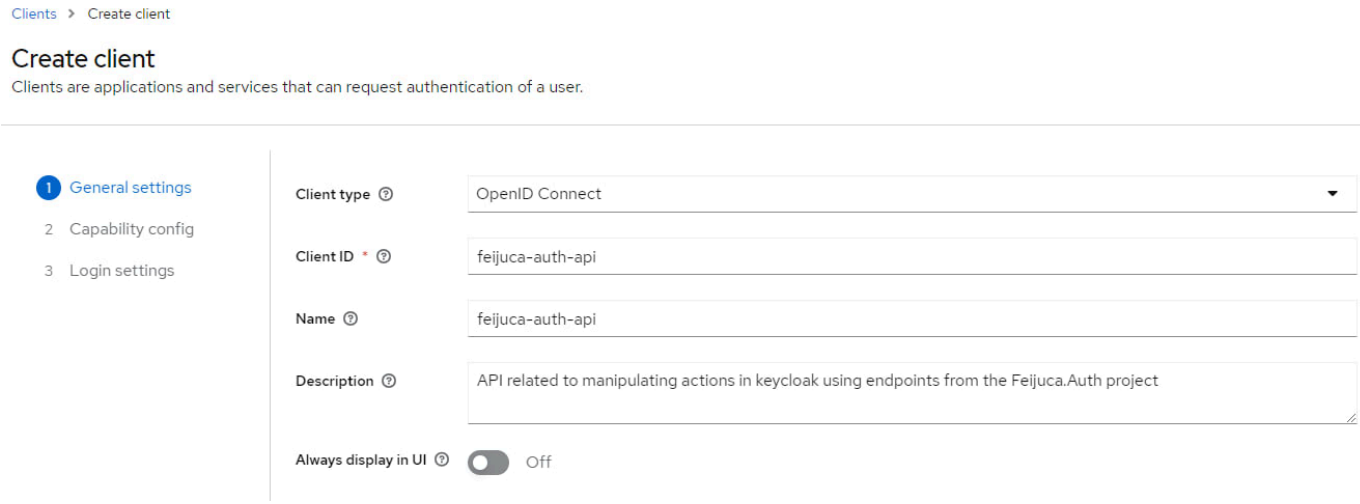
1. 🖥️ Access the Keycloak Admin console

- Log in to the Keycloak Admin Console and select the realm where you want to configure Feijuca.Auth.

---

2. 📄 Define or select the client that will represent Feijuca.Auth.

- We recommend you create a new client dedicated to handling the operations.



---

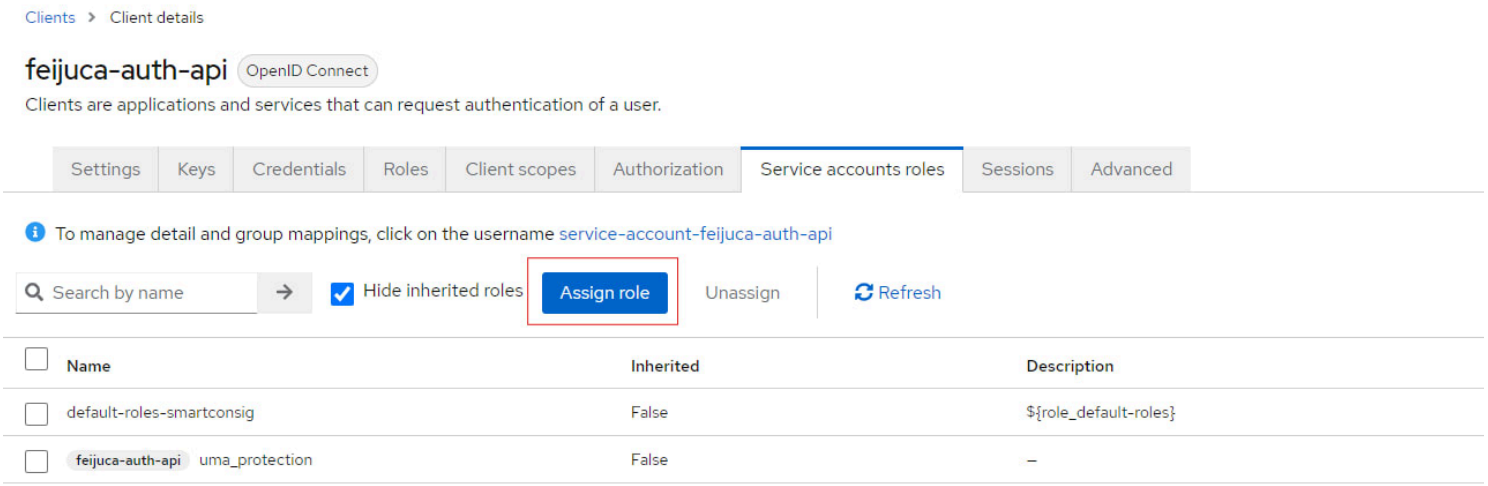
3. 🔧 Access the **Service Account Roles** Tab

- On the client page, click on the **Service Account Roles** tab to manage the permissions for the service account.

---

4. ✅ Assign the Required Roles

- In the **Service Account Roles** tab, you will see a list of available roles.
- Assign the necessary roles to the service account to allow it to perform actions like user creation, group management, realm and client creation, etc.



📜 Mandatory Roles to Assign

- **After clicking "Assign Role"**, switch the filter to **"Filter by Clients"** to ensure you're assigning the correct roles to the service account.

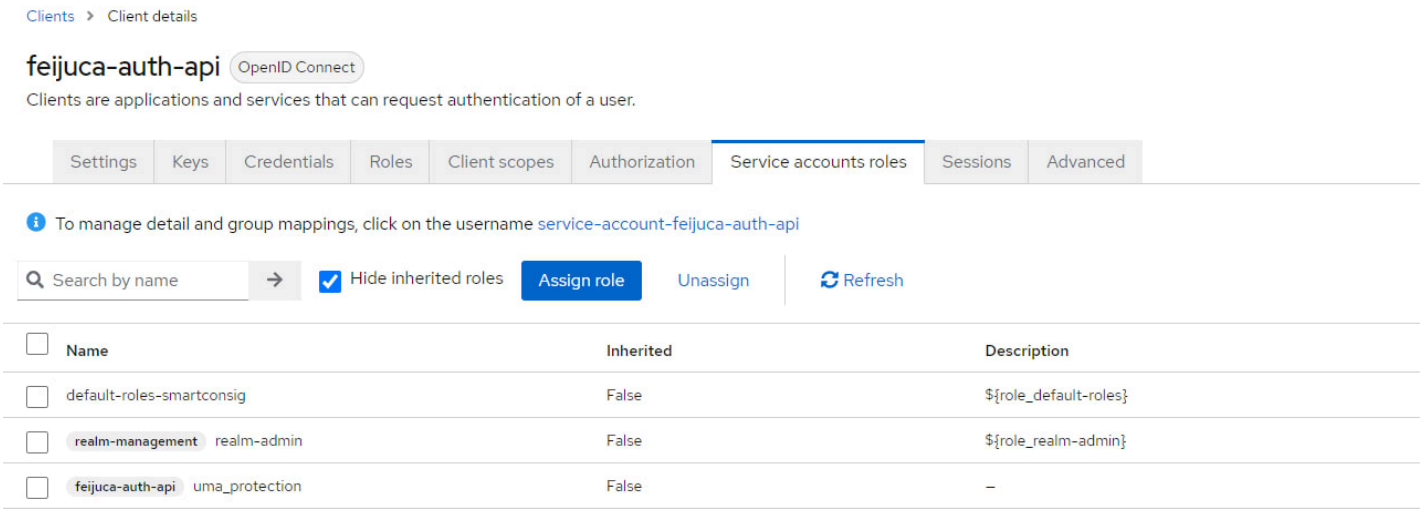
  Then, assign the following essential realm-management roles to give the service account the necessary permissions:

  - `realm-admin`: Grants full administrative access to all realm-level operations and settings.

---

5. 💾 Save the Configuration

- After finish the assignment of the required roles, the final result should be similar to:



✅ Successfully Assigned Roles to Service Account

After assigning the necessary roles to the service account, **Feijuca.Auth.Api** will be able to make authenticated requests to Keycloak's API, allowing you to manage users, groups, clients, roles, and more within the realm.

Once this is complete, you can proceed to the next step: Configuring the API.

🔑 Configuring permissions in Keycloak for API Integration

In order for your API to perform operations such as creating users, groups, realms, clients, roles, group roles, and more in Keycloak, you need to configure the appropriate permissions in the realm. This configuration is done by granting specific permissions to the **Service Account** associated with the client your API uses. Follow the steps below:

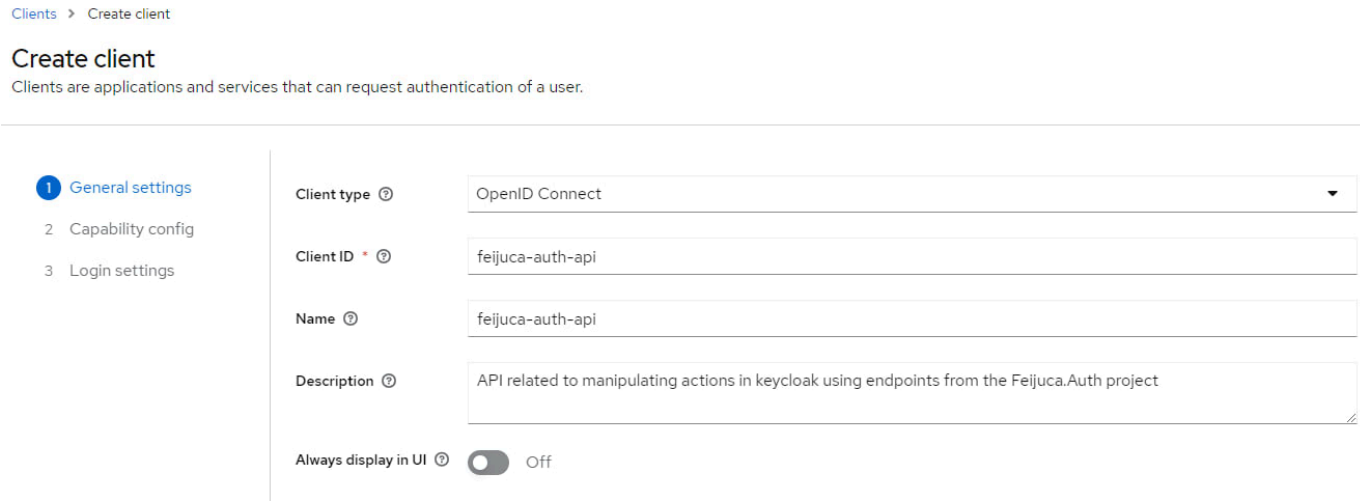
1. 🖥️ Access the Keycloak Admin console

- Log in to the Keycloak Admin Console and select the realm where you want to configure Feijuca.Auth.

---

2. 📄 Define or select the client that will represent Feijuca.Auth.

- We recommend you create a new client dedicated to handling the operations.



---

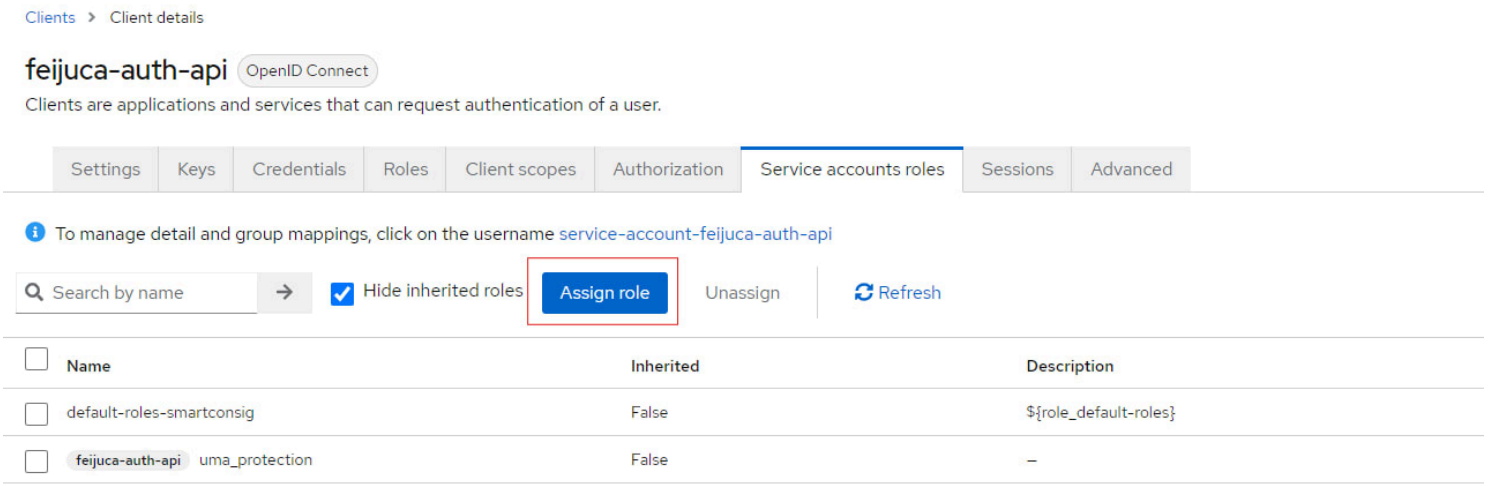
3. 🔧 Access the **Service Account Roles** Tab

- On the client page, click on the **Service Account Roles** tab to manage the permissions for the service account.

---

4. ✅ Assign the Required Roles

- In the **Service Account Roles** tab, you will see a list of available roles.
- Assign the necessary roles to the service account to allow it to perform actions like user creation, group management, realm and client creation, etc.



📜 Mandatory Roles to Assign

- **After clicking "Assign Role"**, switch the filter to **"Filter by Clients"** to ensure you're assigning the correct roles to the service account.

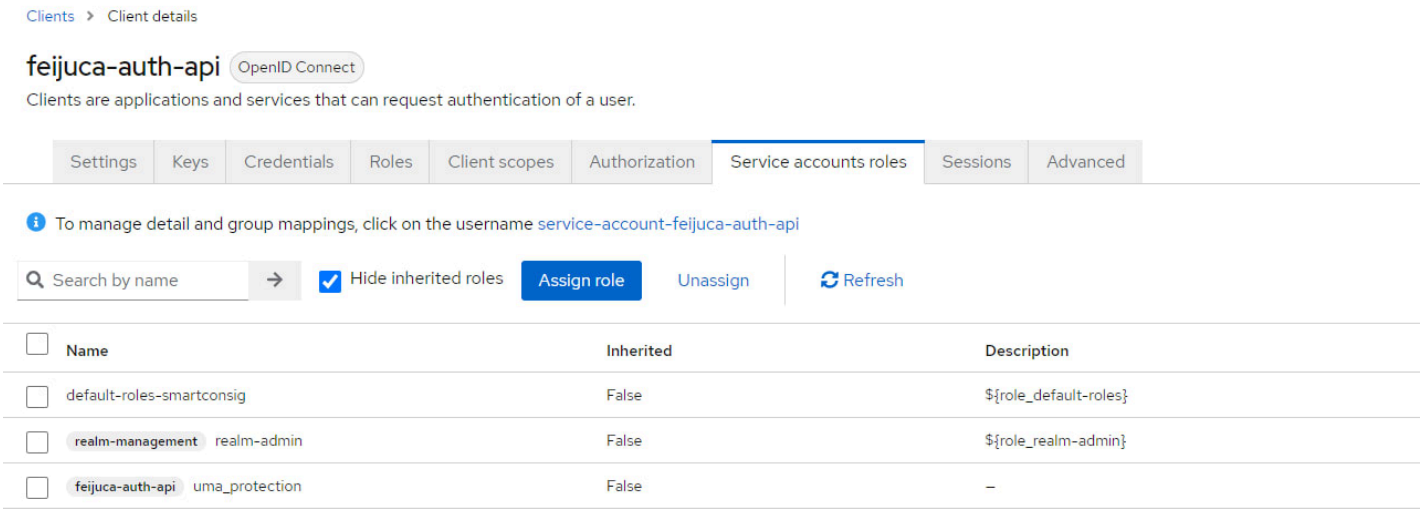
  Then, assign the following essential realm-management roles to give the service account the necessary permissions:

  - `realm-admin`: Grants full administrative access to all realm-level operations and settings.

---

5. 💾 Save the Configuration

- After finish the assignment of the required roles, the final result should be similar to:



✅ Successfully Assigned Roles to Service Account

After assigning the necessary roles to the service account, **Feijuca.Auth.Api** will be able to make authenticated requests to Keycloak's API, allowing you to manage users, groups, clients, roles, and more within the realm.

Once this is complete, you can proceed to the next step: Configuring the API.

🚀 Configuration for Keycloak Integration

To take full advantage of the various endpoints provided by **Feijuca.Auth.Api**, a quick configuration is required to input details about your Keycloak realm. These configurations are crucial because they allow **Feijuca.Auth.Api** to authenticate and retrieve permission tokens to manage users, groups, roles, and much more.

Configuration - General overview

Feijuca.Auth.Api needs some information about the realm you created in **Keycloak** in the previous step.

To store this realm information, we chose **MongoDB** as the data repository, given its flexibility and ease of configuration.

💡 **It is important to remember that this instance belongs to you** — we only expect you to define the connection string. **Feijuca.Auth.Api** only uses the data provided to authenticate with Keycloak. 🔐

Contributing with a Different Database

However, if you want to extend the project and use a different database, feel free to open a **Pull Request** to contribute your custom solution! 🚀

⚙️ Step 1: Setting up mongoDB connection string

The first configuration you need to provide is the **MongoDB connection string**. This will enable you to store and manage the Keycloak realm settings.

> **Tip**: If you don't have a MongoDB instance set up, you can create a free mongoDB server on MongoDB Atlas.

🐳 Step 2: Running Feijuca.Auth.Api with Docker

Once your MongoDB instance is ready, you can run **Feijuca.Auth.Api** using Docker. You need to pass the MongoDB connection string as an environment variable to ensure the API can communicate with the database.

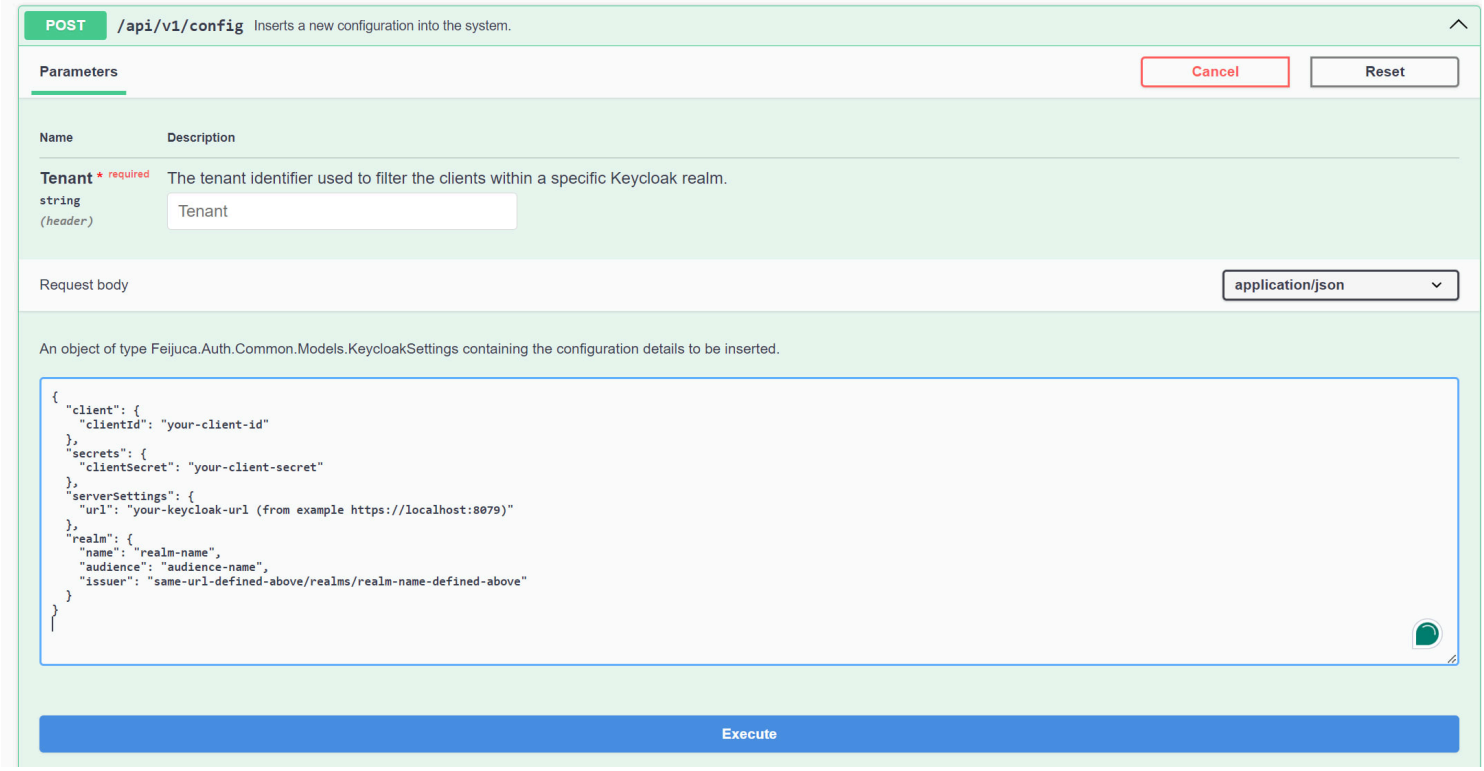
Run the following command to start the API:

```
docker run -e ConnectionString="mongodb://<username>:<password>@<host>:<port>"
coderaw/feijuca-auth-api:latest
```

🛠️ Step 3: Inserting the realm configuration

Once your Docker container is up and running with the correct configuration, you're ready to insert your Keycloak realm configuration. To insert the realm configuration, send an **HTTP POST** request to the `/api/v1/config` endpoint, with the following JSON body:

Endpoint definition



POST

/api/v1/config

Summary:

Inserts a new configuration into the system.

Header

Name	Located in	Description	Required	Schema
Tenant	header	The tenant identifier used to filter the clients within a specific Keycloak realm.	Yes	string

Body definition

Name	Located in	Description	Required	Schema
Client	body	The client id related to the client that was created to manage operations in the realm.	Yes	string
Secret	body	The client secret related to the client that was created to manage operations in the realm.	Yes	string
Server settings	body	The url where you keycloak is running.	Yes	string
Realm	body	The realm name (if you wanna use multitenancy concept, inform an array of this object), the audience name configured previosly and the issuer	Yes	object

Body example

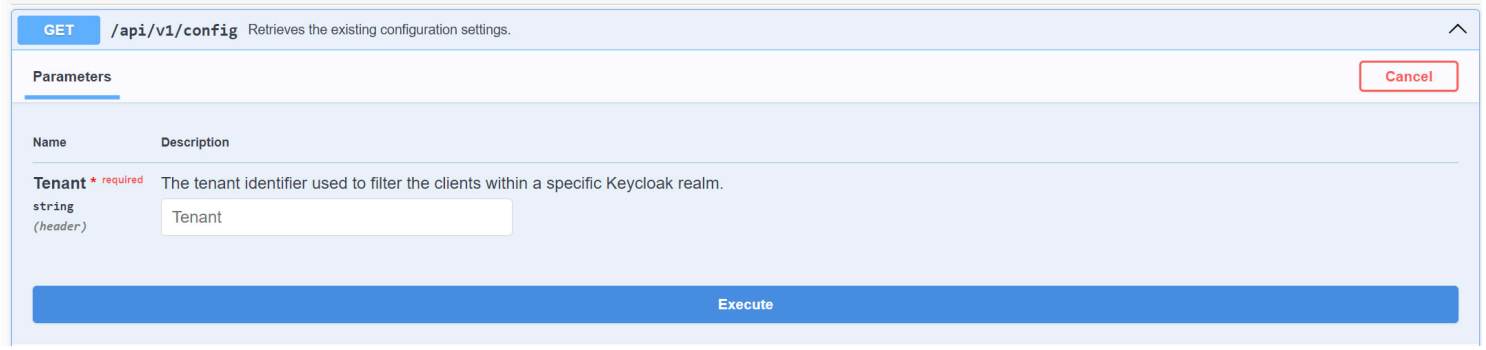
```
{
  "client": {
    "clientId": "your-client-id"
  },
  "secrets": {
    "clientSecret": "your-client-secret"
  },
  "serverSettings": {
    "url": "your-keycloak-url (from example https://localhost:8079)"
  },
  "realm": {
    "name": "realm-name",
    "audience": "audience-name",
    "issuer": "same-url-defined-above/realms/realm-name-defined-above"
  }
}
```

Responses

Code	Description
201	The configuration was successfully inserted.
400	The request was invalid or could not be processed.
500	An internal server error occurred during the processing of the request.

✅ Step 4: Confirming if your changes was applied

Endpoint definition



GET

Summary:

Retrieves the existing configuration settings.

Parameters

Name	Located in	Description	Required	Schema
Tenant	header	The tenant identifier used to filter the clients within a specific Keycloak realm.	Yes	string

Responses

Code	Description
200	The operation was successful, and the configuration settings are returned.
400	The request was invalid or could not be processed.
500	An internal server error occurred during the processing of the request.

Every configs necessary thing is done! 🔐✅

After completing the configuration above, it will be necessary only to restart your container to make it possible for the project to start again and apply the configurations. Now, you'll be ready to access all endpoints and easily manage the various instances a Keycloak realm offers. You can now begin managing users, groups, roles, and more.

👷 Ready to the next steps? Creating users.

⚙️ Endpoint specification

Method: GET

Path: /users

Summary:

Retrieves all users existing in the specified Keycloak realm.

Responses

Code	Description
200	The operation was successful, and the configuration settings are returned.
400	The request was invalid or could not be processed.
500	An internal server error occurred during the processing of the request.
401	The request lacks valid authentication credentials.
403	The request was understood, but the server is refusing to fulfill it due to insufficient permissions.

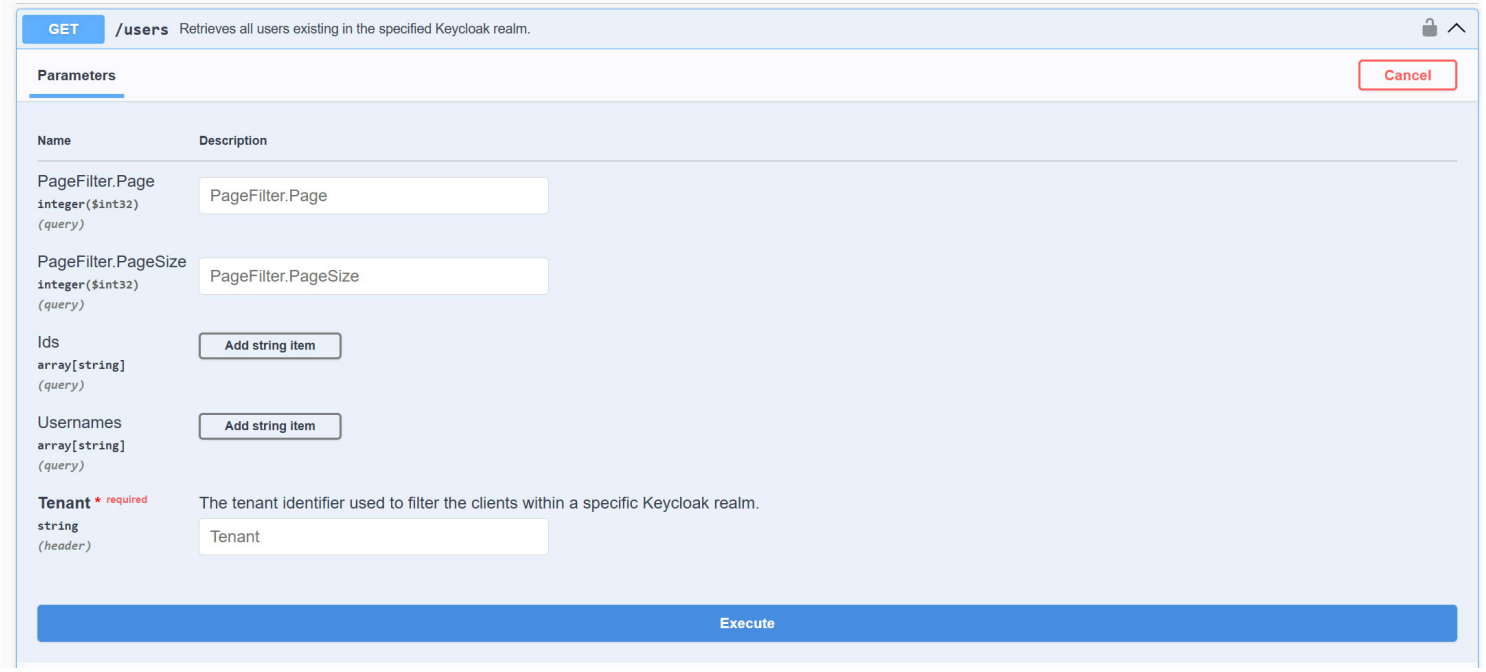
Header

Name	Located in	Description	Required	Schema
Tenant	header	The tenant identifier used to filter the clients within a specific Keycloak realm.	Yes	string

Query params

Name	Located in	Description	Required	Schema
PageFilter.Page	query	The page related to the data that is being showed.	No	int
Ids	query	The user ids that can be used to filter determinated users.	No	Guid
Usernames	query	The Usernames that can be used to filter determinated users.	No	int

Definition



📝 How to Use the Endpoint

1. **Tenant Identification**:
    - The term *tenant* in Feijuca represents the **realm name** within Keycloak where you'll be performing actions.
    - You must specify the tenant name in the **HTTP header** to proceed.
2. **Query Parameters**:
    - After setting up the header, use the available **query parameters** to customize your search based on your requirements.

🔍 Available Search Types

1. Get All Users

- Fetches all users within the realm.
- The results are provided in a paginated format, allowing you to navigate through pages to explore the user list.

2. Get Users by ID

- Enables filtering by an **array of user IDs**.
- This will search the realm and return users matching the provided IDs.

3. Get Users by Username

- Allows filtering by an **array of usernames**.

- The API will locate and return users that match the usernames provided in the request.

⚙️ Endpoint specification

Method: GET

Path: /users

Summary:

Retrieves all users existing in the specified Keycloak realm.

Responses

Code	Description
200	The operation was successful, and the configuration settings are returned.
400	The request was invalid or could not be processed.
500	An internal server error occurred during the processing of the request.
401	The request lacks valid authentication credentials.
403	The request was understood, but the server is refusing to fulfill it due to insufficient permissions.

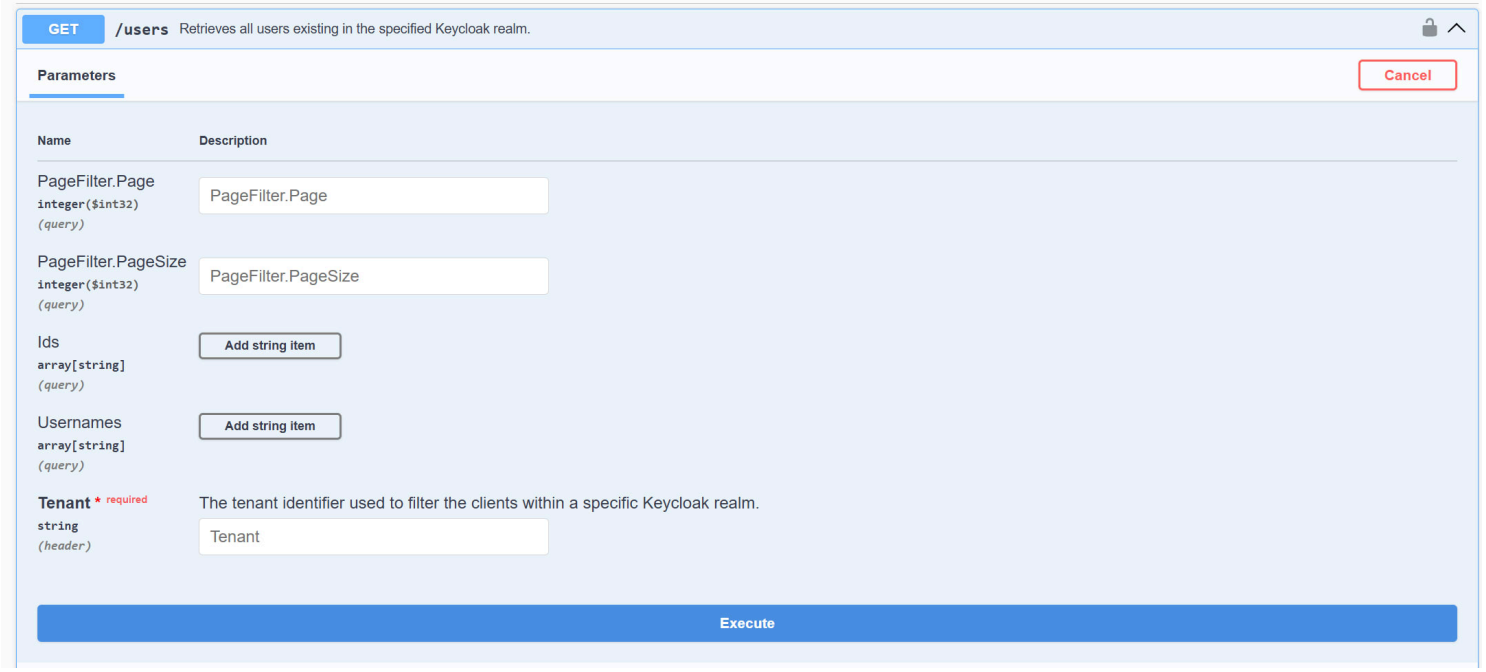
Header

Name	Located in	Description	Required	Schema
Tenant	header	The tenant identifier used to filter the clients within a specific Keycloak realm.	Yes	string

Query params

Name	Located in	Description	Required	Schema
PageFilter.Page	query	The page related to the data that is being showed.	No	int
Ids	query	The user ids that can be used to filter determinated users.	No	Guid
Usernames	query	The Usernames that can be used to filter determinated users.	No	int

Definition



📝 How to Use the Endpoint

1. **Tenant Identification**:
    - The term *tenant* in Feijuca represents the **realm name** within Keycloak where you'll be performing actions.
    - You must specify the tenant name in the **HTTP header** to proceed.
2. **Query Parameters**:
    - After setting up the header, use the available **query parameters** to customize your search based on your requirements.

🔍 Available Search Types

1. Get All Users

- Fetches all users within the realm.
- The results are provided in a paginated format, allowing you to navigate through pages to explore the user list.

2. Get Users by ID

- Enables filtering by an **array of user IDs**.
- This will search the realm and return users matching the provided IDs.

3. Get Users by Username

- Allows filtering by an **array of usernames**.

- The API will locate and return users that match the usernames provided in the request.

⚙️ Endpoint specification

Method: POST

Path: /user

Summary:

Adds a new user to the specified Keycloak realm.

Responses

Code	Description
201	The operation was successful, and the new user was created.
400	The request was invalid or could not be processed.
401	The request lacks valid authentication credentials.
403	The request was understood, but the server is refusing to fulfill it due to insufficient permissions.
500	An internal server error occurred during the processing of the request.

Header

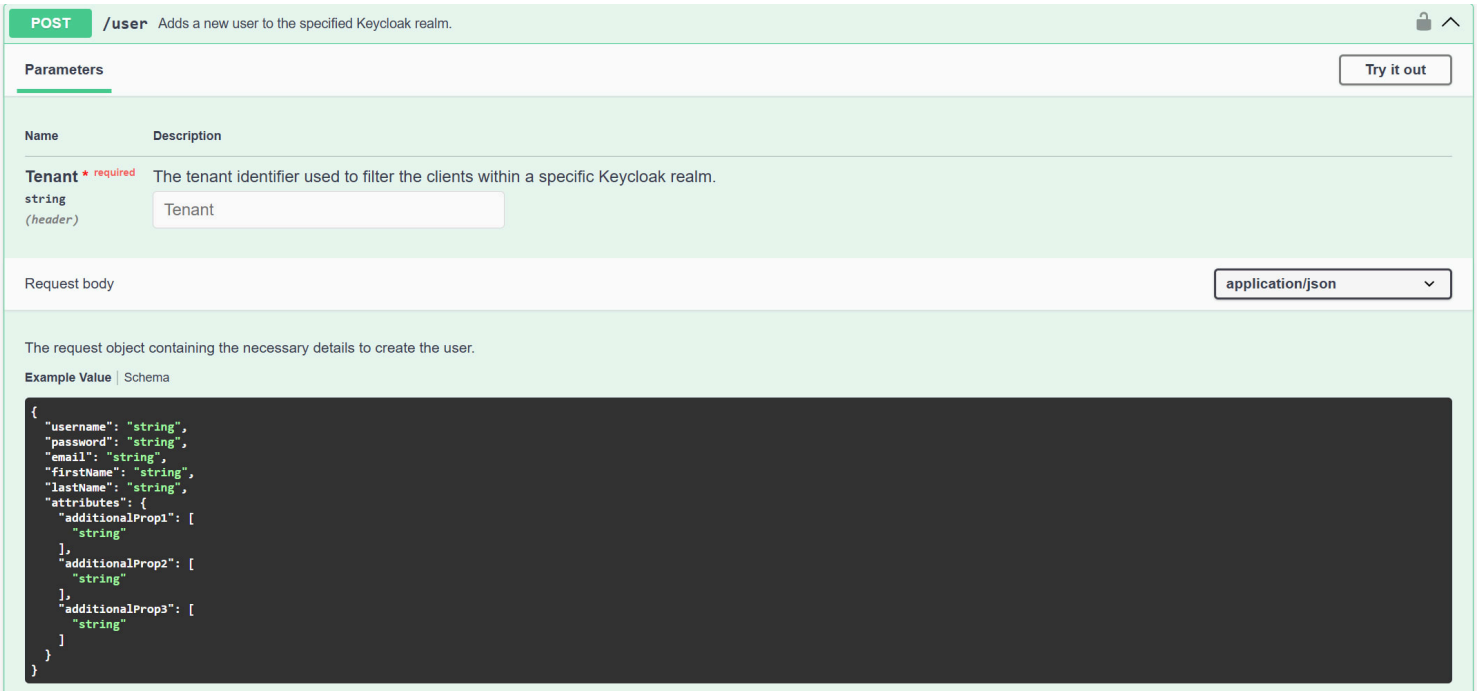
Name	Located in	Description	Required	Schema
Tenant	header	The tenant identifier used to filter the clients within a specific Keycloak realm.	Yes	string

Body definition

Name	Located in	Description	Required	Schema
Username	body	The username related to the new user that will be created.	Yes	string
Password	body	The password related to the new user that will be created.	Yes	string
Email	body	The e-mail related to the new user that will be created.	Yes	string

Name	Located in	Description	Required	Schema
FirstName	body	The first name related to the new user that will be created.	Yes	string
LastName	body	The last name related to the new user that will be created.	Yes	string
Attributes	body	The attributes that will be inserted on the new user that was created	No	Object

Definition



📝 How to Use the Endpoint

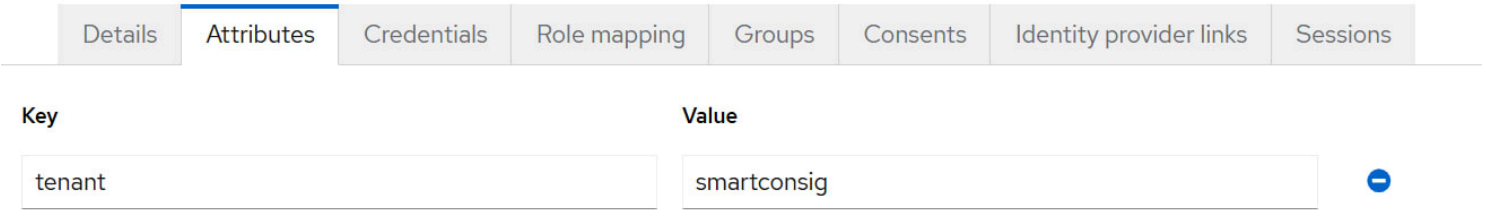
1. **Tenant Identification**:
    - The term *tenant* in Feijuca represents the **realm name** within Keycloak where you'll be performing actions.
    - You must specify the tenant name in the **HTTP header** to proceed.
2. **Body**:
    - After setting up the header, inform a valid body to insert a user. Example:

```
{
  "username": "test",
  "password": "123456",
```

```
    "email": "test@test.com",
    "firstName": "Test Firstname",
    "lastName": "Test Lastname",
    "attributes": {
        "some-attribute": [
                "attribute-value"
                ]
        }
}
```

Reminder

The attributes property is not mandatory, but if you wanna configure some attribute to your user, you can defined it using this field. An example about how attribute works, could be found below:



The JSON used to created this attribute was:

```
"attributes": {
    "tenant": [
      "smartconsig"
    ]
  }
}
```

⚙️ Endpoint specification

Method: DELETE

Path: /user

Summary:

Deletes an existing user from the specified Keycloak realm.

Responses

Code	Description
204	The user was deleted successfully.
400	The request was invalid or the user could not be found.
401	The request lacks valid authentication credentials.
403	The request was understood, but the server is refusing to fulfill it due to insufficient permissions.
500	An internal server error occurred during the processing of the request.

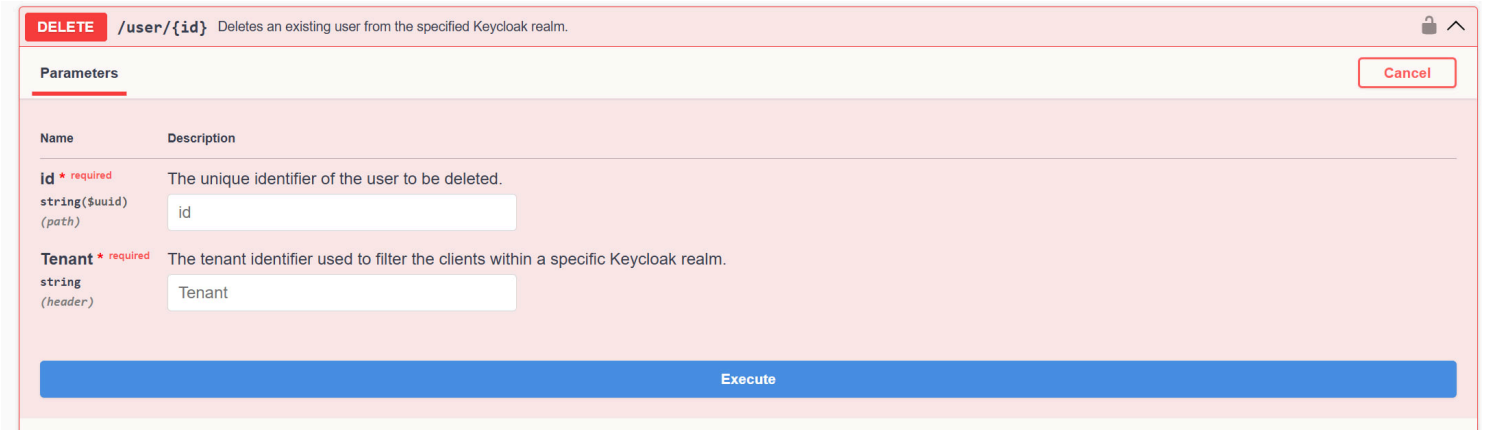
Header

Name	Located in	Description	Required	Schema
Tenant	header	The tenant identifier used to filter the clients within a specific Keycloak realm.	Yes	string

Query params

Name	Located in	Description	Required	Schema
id	query	The unique identifier of the user to be deleted.	Yes	Guid

Definition



How to Use the Endpoint

1. **Tenant Identification**:
    - The term *tenant* in Feijuca represents the **realm name** within Keycloak where you'll be performing actions.
    - You must specify the tenant name in the **HTTP header** to proceed.
2. **Query Parameters**:
    - After setting up the header, use the available **query parameters** to inform what user you want to delete.

⚙️ Endpoint specification

Method: PUT

Path: /user

Summary:

Updates an existing user in a Keycloak realm.

Responses

Code	Description
201	Authentication was successful, and the JWT token and user attributes was updated.
202	Accepted.
400	The request was invalid, such as incorrect credentials.
401	The request lacks valid authentication credentials.
403	The request was understood, but the server is refusing to fulfill it due to insufficient permissions.
500	An internal server error occurred while processing the request.

Header

Name	Located in	Description	Required	Schema
Tenant	header	The tenant identifier used to filter the clients within a specific Keycloak realm.	Yes	string

Query Parameter

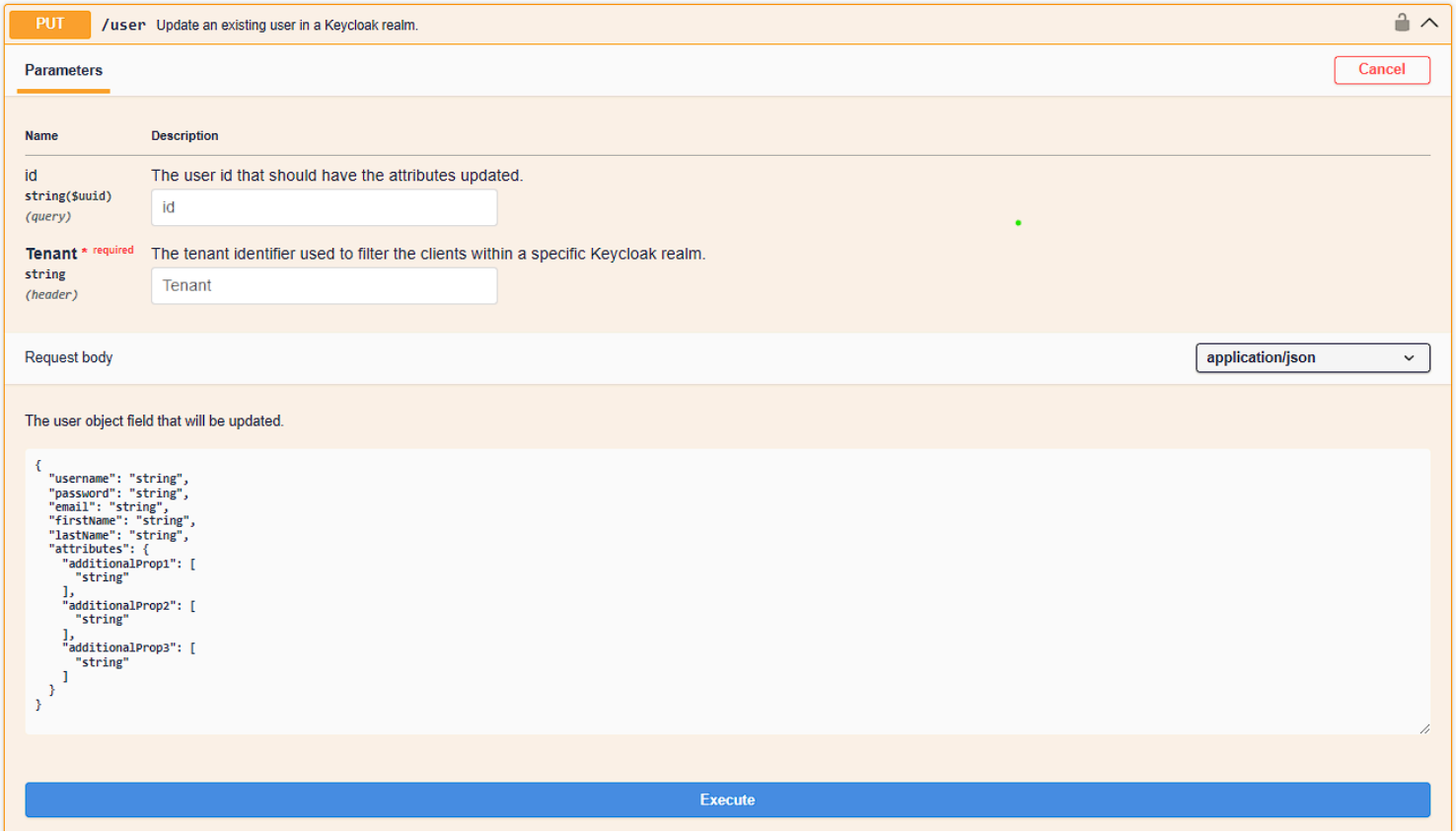
Name	Located in	Description	Required	Schema
id	query	The user id that should have the attributes updated.	Yes	Guid

Body definition

Name	Located in	Description	Required	Schema
username	body	The username of the user to be updated.	Yes	string

Name	Located in	Description	Required	Schema
password	body	The password of the user to be updated.	Yes	string
email	body	The email of the user to be updated.	Yes	string
firstName	body	The first name of the user to be updated.	Yes	string
lastName	body	The last name of the user to be updated.	Yes	string
attributes	body	The attributes of the user to be updated.	No	Object

Definition



📝 How to Use the Endpoint

1. **Tenant Identification**:
    - The term *tenant* in Feijuca represents the **realm name** within Keycloak where you'll be performing actions.
    - You must specify the tenant name in the **HTTP header** to proceed.
2. **Query Parameter**:
    - Provide the `id` of the user in the query string to specify the user to update.
3. **Body**:

- After setting up the header and query parameter, provide a valid body to update the user attributes. Example:

```
{
  "username": "updatedTest",
  "email": "updated@test.com",
  "firstName": "Updated Firstname",
  "lastName": "Updated Lastname",
  "attributes": {
    "updated-attribute": ["new-value"]
  }
}
```

⚙️ Endpoint specification

Method: POST

Path: /user/revoke-session

Summary:

Revokes all active sessions for a specified user in Keycloak.

Responses

Code	Description
202	The user sessions were revoked successfully.
400	The request was invalid or the user could not be found.
401	The request lacks valid authentication credentials.
403	The request was understood, but the server is refusing to fulfill it due to insufficient permissions.
500	An internal server error occurred during the processing of the request.

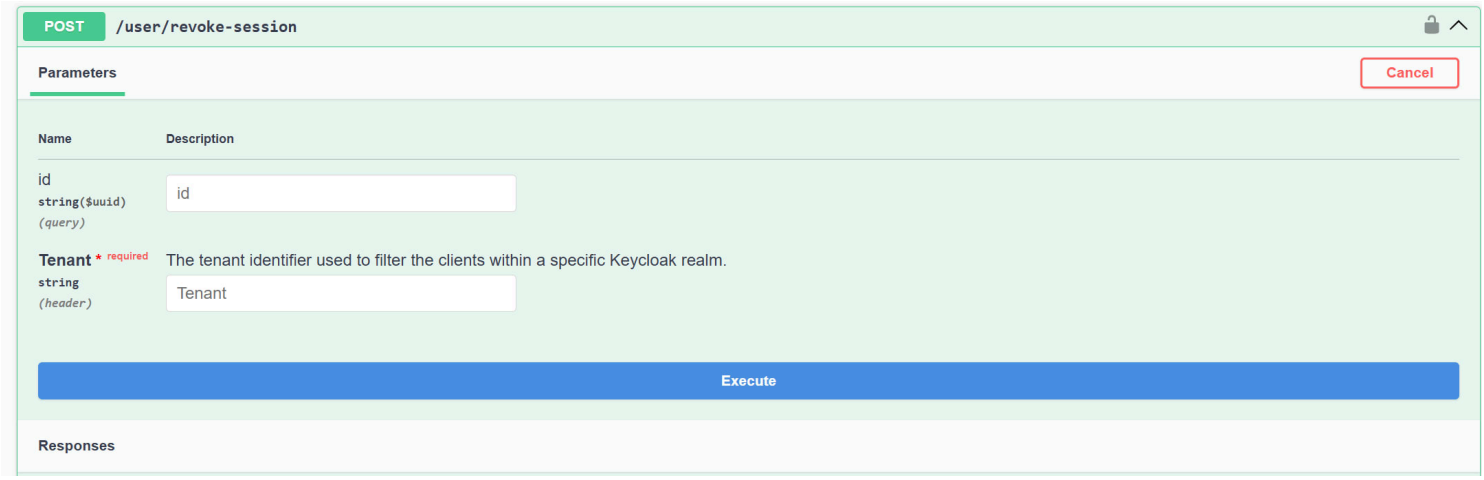
Header

Name	Located in	Description	Required	Schema
Tenant	header	The tenant identifier used to filter the clients within a specific Keycloak realm.	Yes	string

Query params

Name	Located in	Description	Required	Schema
id	query	The unique identifier of the user to be deleted.	Yes	Guid

Definition



How to Use the Endpoint

1. **Tenant Identification**:
    - The term *tenant* in Feijuca represents the **realm name** within Keycloak where you'll be performing actions.
    - You must specify the tenant name in the **HTTP header** to proceed.
2. **Query Parameters**:
    - After setting up the header, inform a valid user user id to revoke the sessions related to the provided user.

⚙️ Endpoint specification

Method: POST

Path: /user/logout

Summary:

Logs out a user and invalidates their session.

Responses

Code	Description
200	The logout was successful.
400	The request was invalid or the logout could not be processed.
500	An internal server error occurred during the processing of the request.

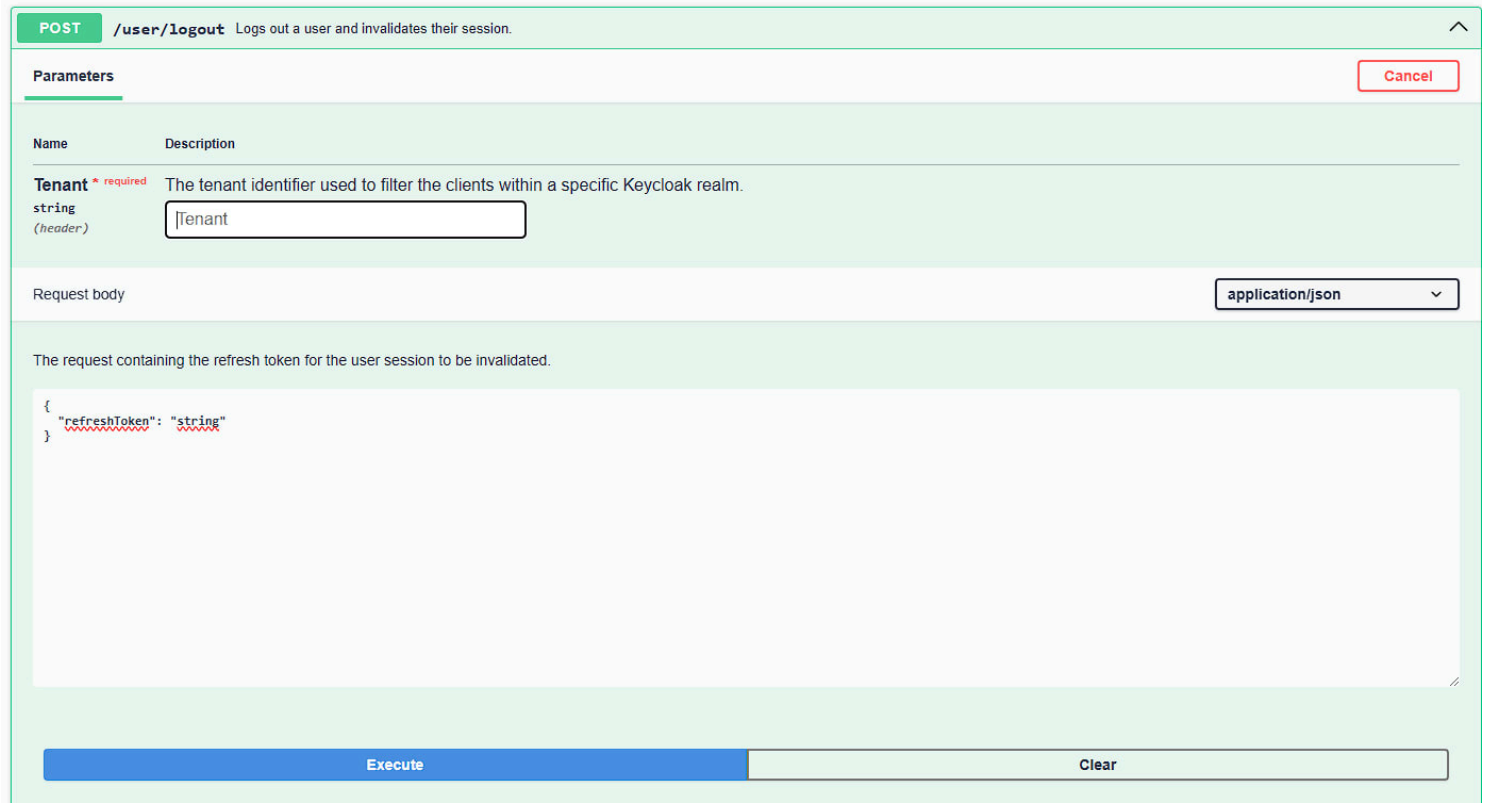
Header

Name	Located in	Description	Required	Schema
Tenant	header	The tenant identifier used to filter the clients within a specific Keycloak realm.	Yes	string

Body definition

Name	Located in	Description	Required	Schema
refreshToken	body	The refreshToken is used to identify the user session that will be invalidated.	Yes	string

Definition



How to Use the Endpoint

1. **Tenant Identification**:
    - The term *tenant* in Feijuca represents the **realm name** within Keycloak where you'll be performing actions.
    - You must specify the tenant name in the **HTTP header** to proceed.
2. **Body**:
    - After setting up the header, inform a valid body to insert a user. Example:

```
{
  "refreshToken": "your-refresh-token"
}
```

Reminder

⚙️ Endpoint specification

Method: POST

Path: /user/login

Summary:

Authenticates a user and returns a valid JWT token along with user details.

Responses

Code	Description
200	Authentication was successful, and the JWT token and user details are returned.
400	The request was invalid, such as incorrect credentials.
500	An internal server error occurred while processing the request.

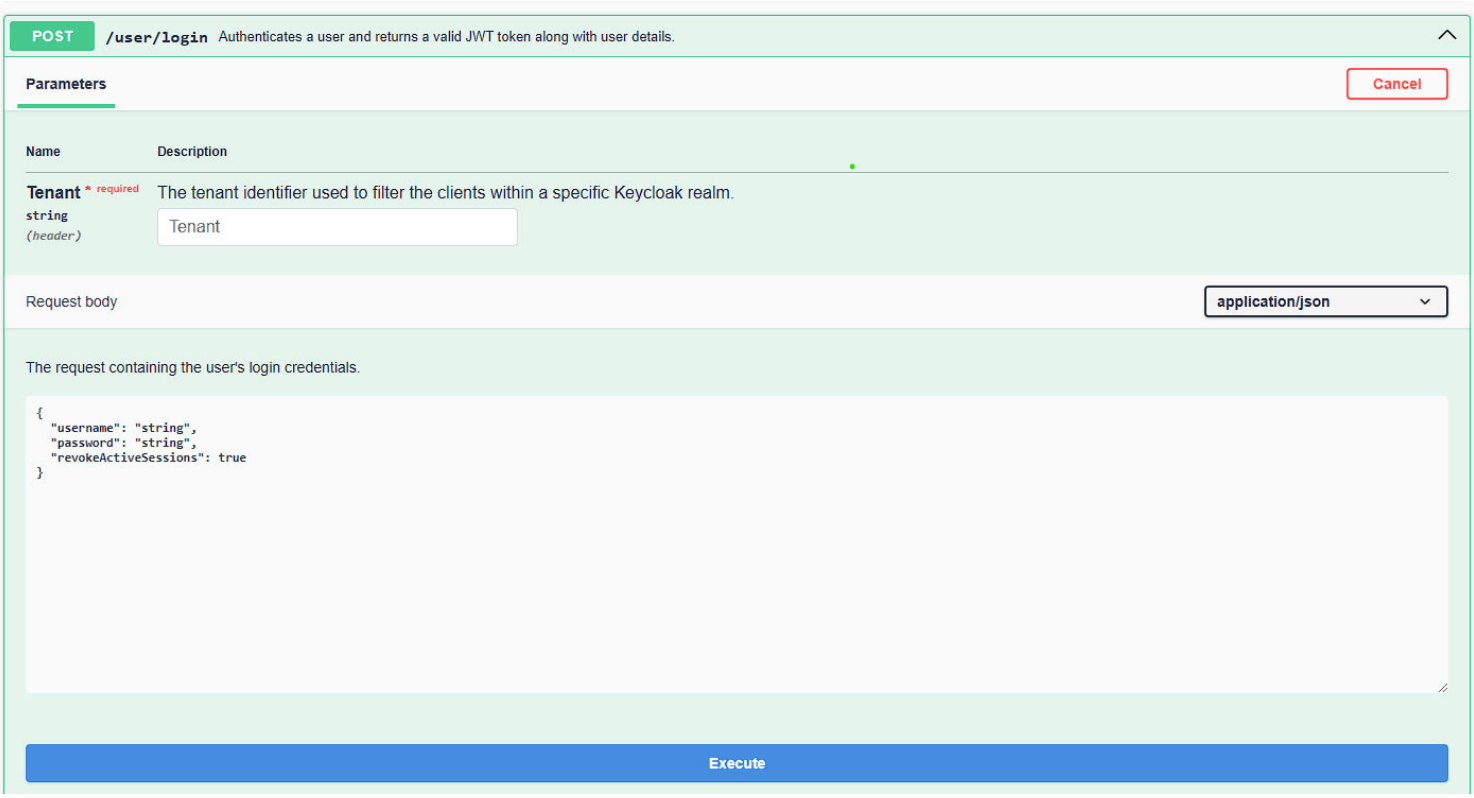
Header

Name	Located in	Description	Required	Schema
Tenant	header	The tenant identifier used to filter the clients within a specific Keycloak realm.	Yes	string

Body definition

Name	Located in	Description	Required	Schema
Username	body	The username related to the new user that will be created.	Yes	string
Password	body	The password related to the new user that will be created.	Yes	string
revokeActiveSessions	body	A boolean parameter (true or false) indicating whether the user's active sessions should be revoked during the login process.	Yes	string

Definition



How to Use the Endpoint

1. **Tenant Identification**:
    - The term *tenant* in Feijuca represents the **realm name** within Keycloak where you'll be performing actions.
    - You must specify the tenant name in the **HTTP header** to proceed.
2. **Body**:
    - After setting up the header, inform a valid body to insert a user. Example:

```
{
  "username": "teste@teste.com",
 "password": "teste12345",
 "revokeActiveSessions": true
}
```

⚙️ Endpoint specification

Method: POST

Path: /user/login

Summary:

Decodes the JWT token and returns the details related to the user logged.

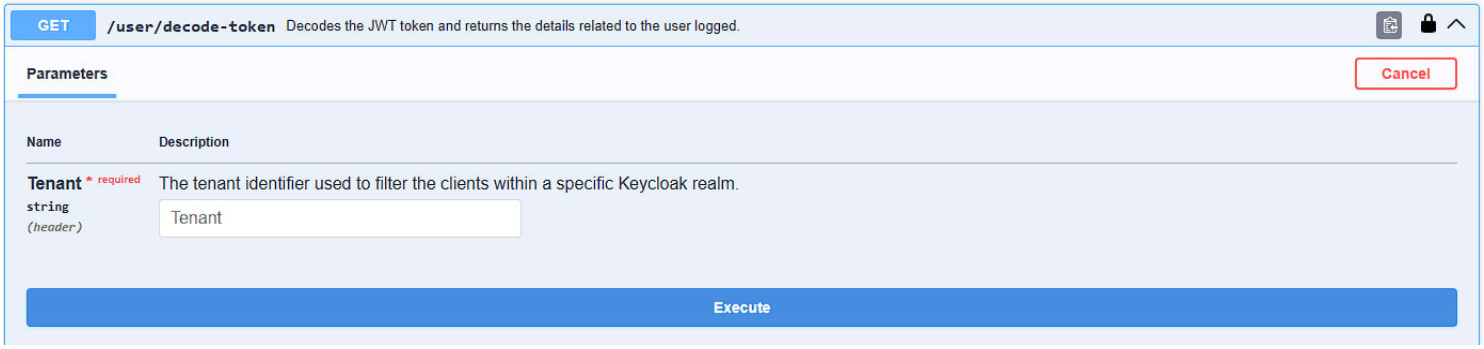
Responses

Code	Description
200	The token was successfully decoded, returning user details.
400	The request was invalid due to an issue with the token or user authentication.
401	The request lacks valid authentication credentials.
403	The request was understood, but the server is refusing to fulfill it due to insufficient permissions.
500	An internal server error occurred during the processing of the request.

Header

Name	Located in	Description	Required	Schema
Tenant	header	The tenant identifier used to filter the clients within a specific Keycloak realm.	Yes	string

Definition



📝 How to Use the Endpoint

1. **Tenant Identification**:

- The term *tenant* in Feijuca represents the **realm name** within Keycloak where you'll be performing actions.
- You must specify the tenant name in the **HTTP header** to proceed.

⚙️ Endpoint specification

Method: POST

Path: /user/refresh-token

Summary:

Refreshes a valid JWT token and returns the new token along with user details.

Responses

Code	Description
200	The refresh operation was successful, returning a new token and user details.
400	The request was invalid due to an issue with the refresh token.
500	An internal server error occurred while processing the request.

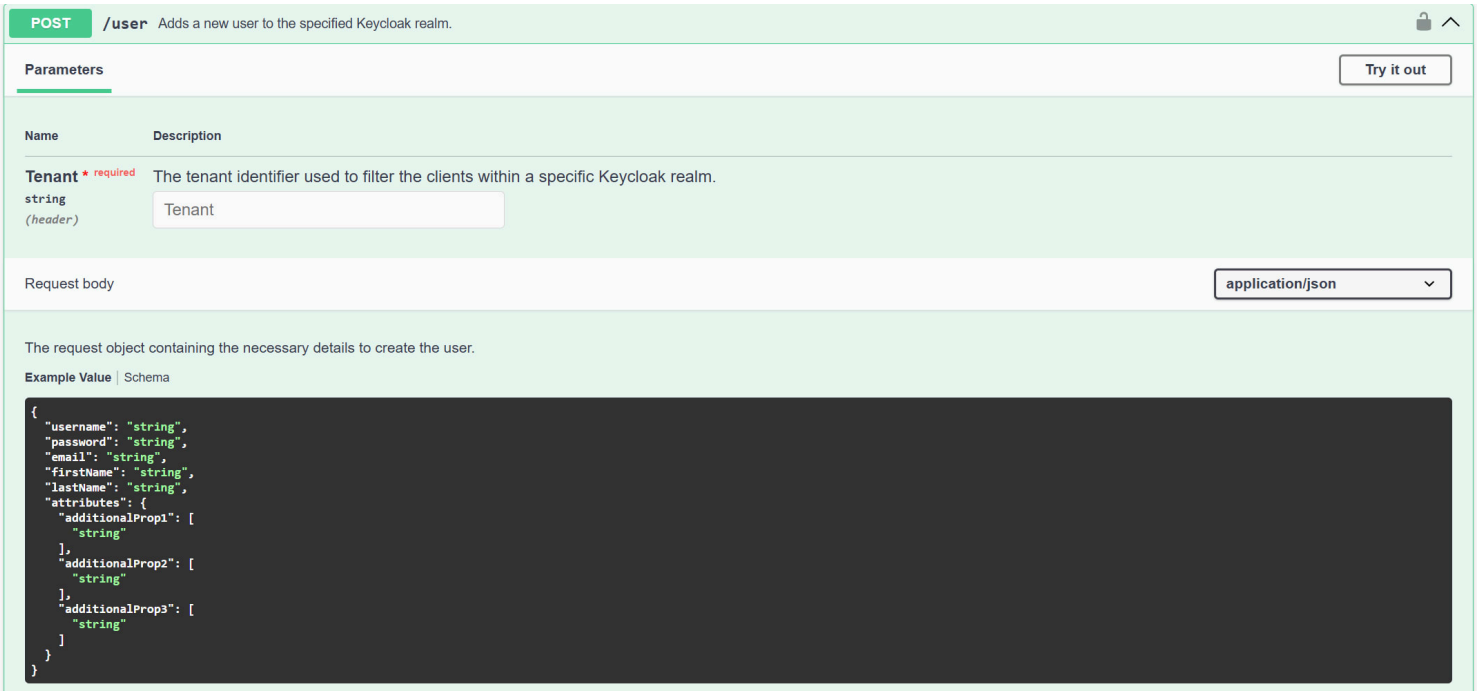
Header

Name	Located in	Description	Required	Schema
Tenant	header	The tenant identifier used to filter the clients within a specific Keycloak realm.	Yes	string

Body definition

Name	Located in	Description	Required	Schema
refreshToken	body	The refreshToken is used to identify the user session that will be invalidated.	Yes	string

Definition



📝 How to Use the Endpoint

1. **Tenant Identification**:
    - The term *tenant* in Feijuca represents the **realm name** within Keycloak where you'll be performing actions.
    - You must specify the tenant name in the **HTTP header** to proceed.
2. **Body**:
    - After setting up the header, inform a valid body to insert a user. Example:

```
{
  "refreshToken": "your-refresh-token"
}
```

⚙️ Endpoint specification

Method: GET

Path: /groups

Summary:

Returns all groups registered in the realm.

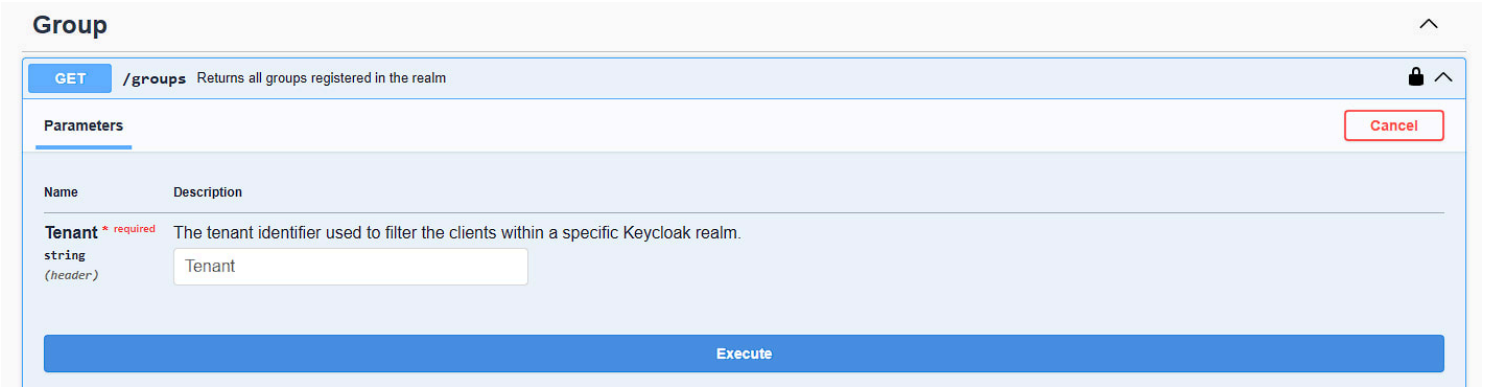
Responses

Code	Description
200	The operation was successful, and the list of groups is returned.
400	The request was invalid or could not be processed.
401	The request lacks valid authentication credentials.
403	The request was understood, but the server is refusing to fulfill it due to insufficient permissions.
500	An internal server error occurred during the processing of the request.

Header

Name	Located in	Description	Required	Schema
Tenant	header	The tenant identifier used to filter the clients within a specific Keycloak realm.	Yes	string

Definition



📝 How to Use the Endpoint

1. **Tenant Identification**:

- The term *tenant* in Feijuca represents the **realm name** within Keycloak where you'll be performing actions.
- You must specify the tenant name in the **HTTP header** to proceed.

⚙️ Endpoint specification

Method: GET

Path: /groups

Summary:

Returns all groups registered in the realm.

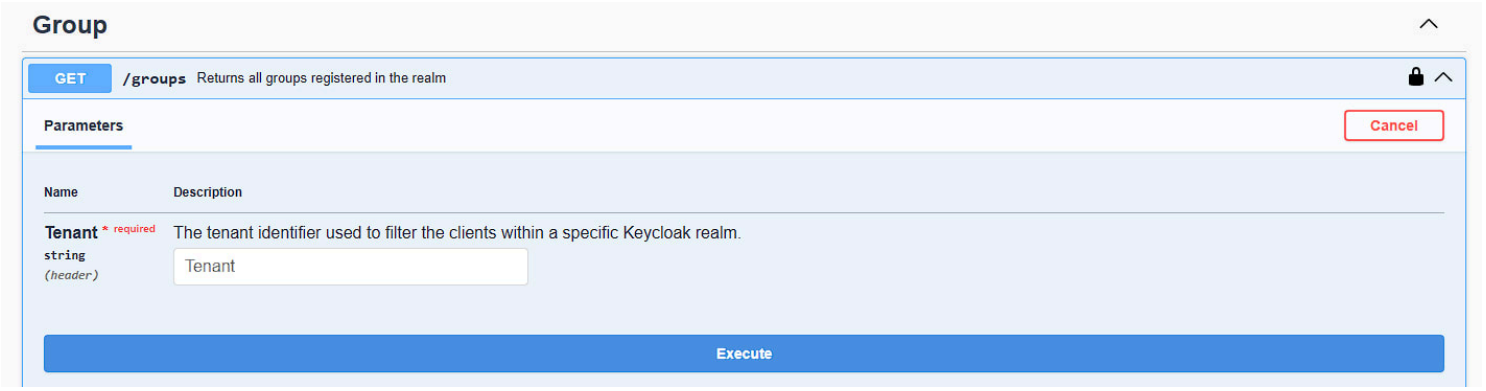
Responses

Code	Description
200	The operation was successful, and the list of groups is returned.
400	The request was invalid or could not be processed.
401	The request lacks valid authentication credentials.
403	The request was understood, but the server is refusing to fulfill it due to insufficient permissions.
500	An internal server error occurred during the processing of the request.

Header

Name	Located in	Description	Required	Schema
Tenant	header	The tenant identifier used to filter the clients within a specific Keycloak realm.	Yes	string

Definition



📝 How to Use the Endpoint

1. **Tenant Identification**:

- The term *tenant* in Feijuca represents the **realm name** within Keycloak where you'll be performing actions.
- You must specify the tenant name in the **HTTP header** to proceed.

⚙️ Endpoint specification

Method: POST

Path: /group

Summary:

Adds a new group to the specified Keycloak realm.

Responses

Code	Description
201	The group was successfully created.
400	The request was invalid or could not be processed.
401	The request lacks valid authentication credentials.
403	The request was understood, but the server is refusing to fulfill it due to insufficient permissions.
500	An internal server error occurred during the processing of the request.

Header

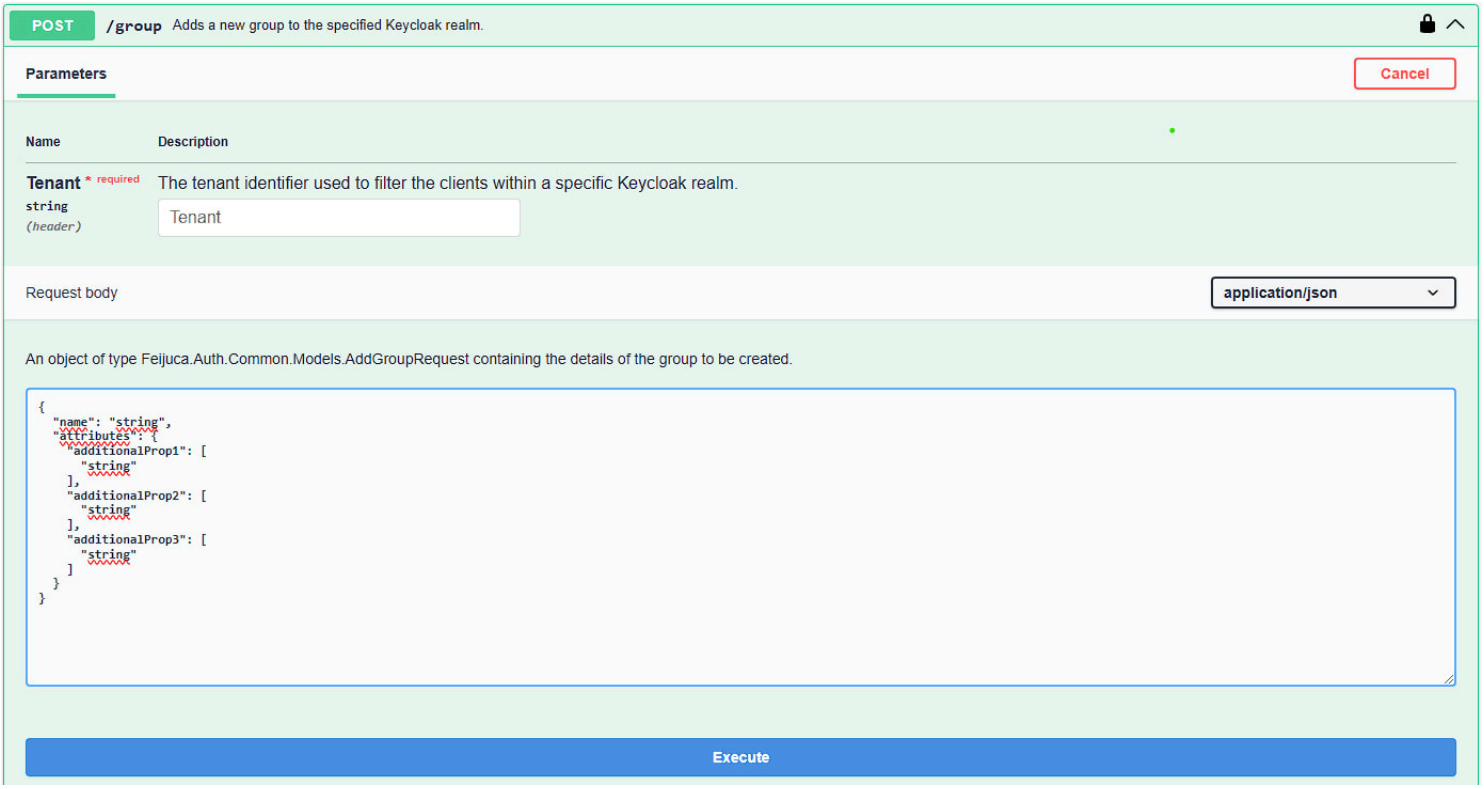
Name	Located in	Description	Required	Schema
Tenant	header	The tenant identifier used to filter the clients within a specific Keycloak realm.	Yes	string

Body definition

Name	Located in	Description	Required	Schema
Username	body	The username related to the new user that will be created.	Yes	string
Password	body	The password related to the new user that will be created.	Yes	string
Email	body	The e-mail related to the new user that will be created.	Yes	string

Name	Located in	Description	Required	Schema
FirstName	body	The first name related to the new user that will be created.	Yes	string
LastName	body	The last name related to the new user that will be created.	Yes	string
Attributes	body	The attributes that will be inserted on the new user that was created	No	Object

Definition



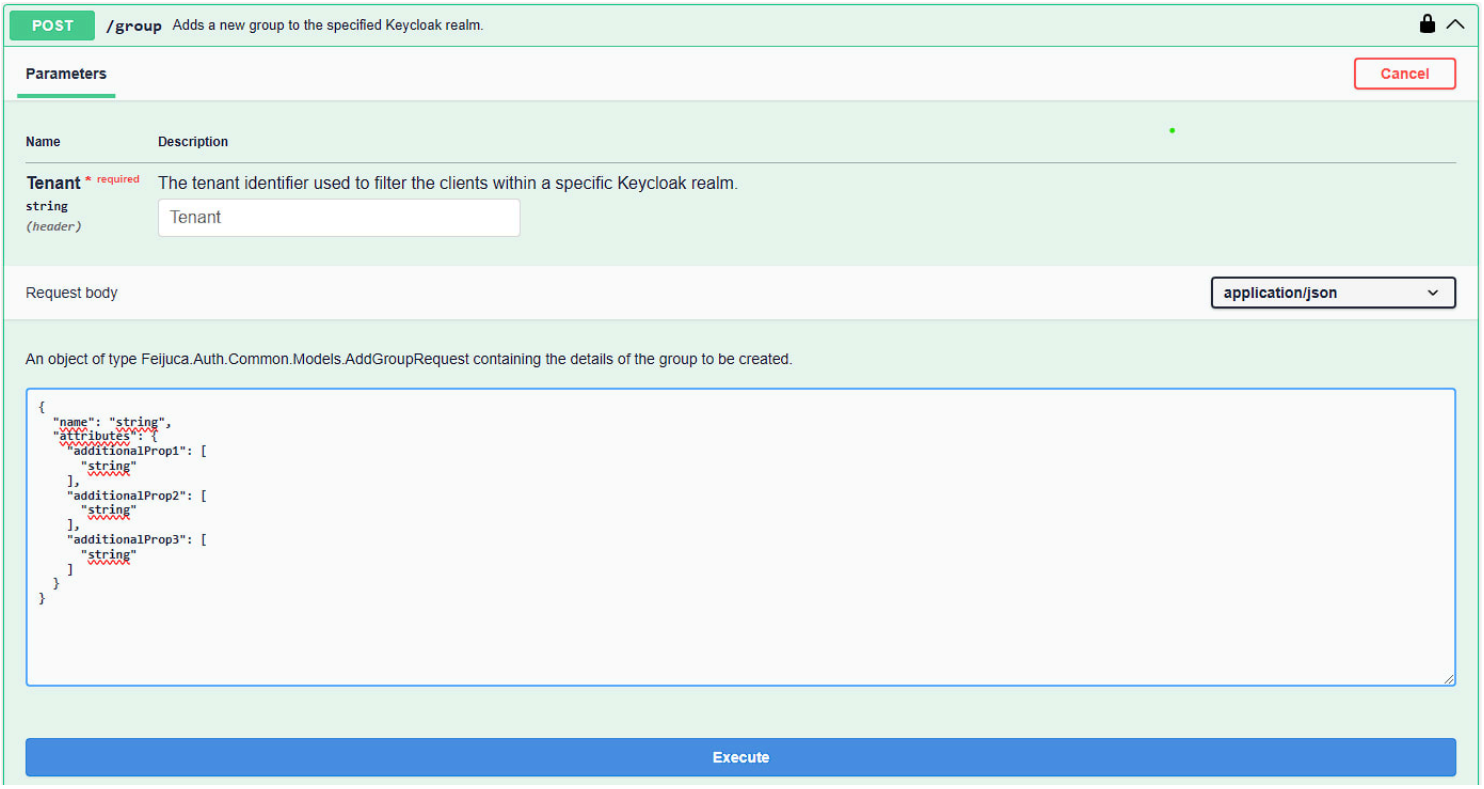
📝 How to Use the Endpoint

1. **Tenant Identification**:
    - The term *tenant* in Feijuca represents the **realm name** within Keycloak where you'll be performing actions.
    - You must specify the tenant name in the **HTTP header** to proceed.
2. **Body**:
    - After setting up the header, inform a valid body to insert a user. Example:

```
{
  "name": "Name of group",
  "attributes": {
    "some-attribute": ["attribute-value"]
  }
}
```

Reminder

The attributes property is not mandatory, but if you wanna configure some attribute to your group, you can defined it using this field. An example about how attribute works, could be found below:



The JSON used to created this attribute was:

```
"attributes": {
    "additionalProp1": [
      "test1"
    ],
    "additionalProp2": [
      "test2"
    ],
    "additionalProp3": [
      "test3"
    ]
```

```
    }
}
```

⚙ Endpoint specification

Method: POST

Path: /group/{id}

Summary:

Deletes an existing group from the specified keycloak realm.

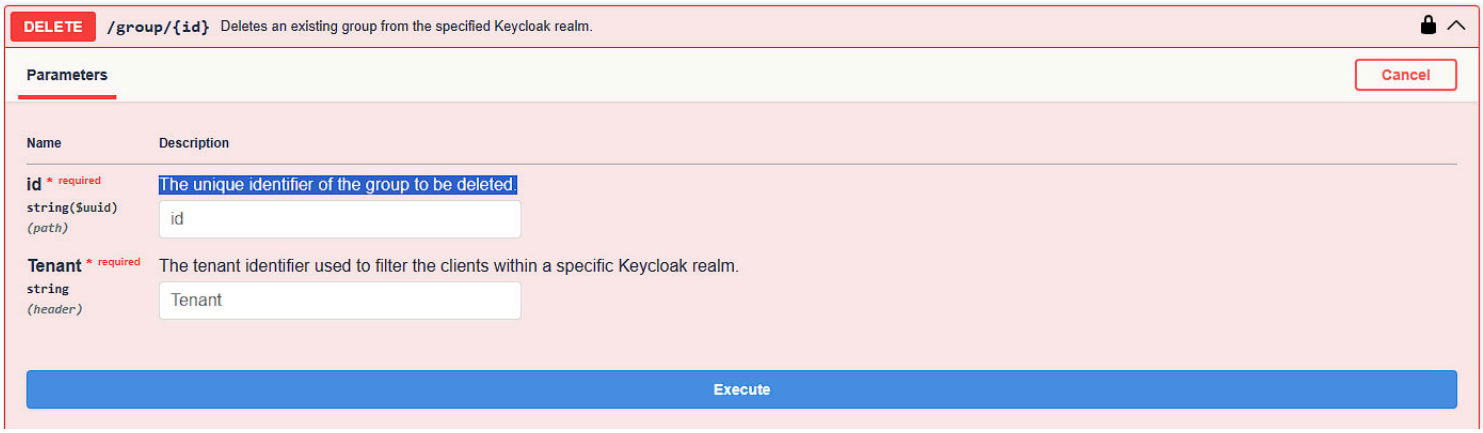
Responses

Code	Description
204	The group was successfully deleted.
400	The request was invalid or could not be processed.
401	The request lacks valid authentication credentials.
403	The request was understood, but the server is refusing to fulfill it due to insufficient permissions.
500	An internal server error occurred during the processing of the request.

Header

Name	Located in	Description	Required	Schema
id	path	The unique identifier of the group to be deleted.	Yes	Guid
Tenant	header	The tenant identifier used to filter the clients within a specific Keycloak realm.	Yes	string

Definition



📝 How to Use the Endpoint

1. **Tenant Identification**:
    - The term *tenant* in Feijuca represents the **realm name** within Keycloak where you'll be performing actions.
    - You must specify the tenant name in the **HTTP header** to proceed.
2. **Id Unique Identifier**:
    - Add the unique identifier that represents the group to be deleted.

⚙️ Endpoint specification

Method: GET

Path: /roles

Summary:

Retrieves all roles associated with clients in the specified Keycloak realm.

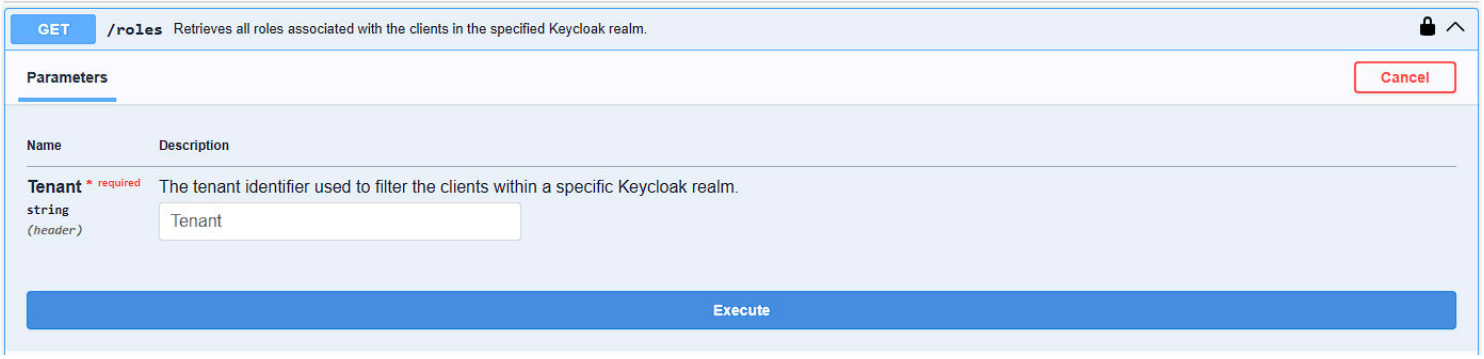
Responses

Code	Description
200	A list of roles associated with the clients in the specified realm
400	The request was invalid or could not be processed.
401	The request lacks valid authentication credentials.
403	The request was understood, but the server is refusing to fulfill it due to insufficient permissions.
500	An internal server error occurred during the processing of the request.

Header

Name	Located in	Description	Required	Schema
Tenant	header	The tenant identifier used to filter the clients within a specific Keycloak realm.	Yes	string

Definition



📝 How to Use the Endpoint

1. **Tenant Identification**:

- The term *tenant* in Feijuca represents the **realm name** within Keycloak where you'll be performing actions.
- You must specify the tenant name in the **HTTP header** to proceed.

⚙️ Endpoint specification

Method: GET

Path: /roles

Summary:

Retrieves all roles associated with clients in the specified Keycloak realm.

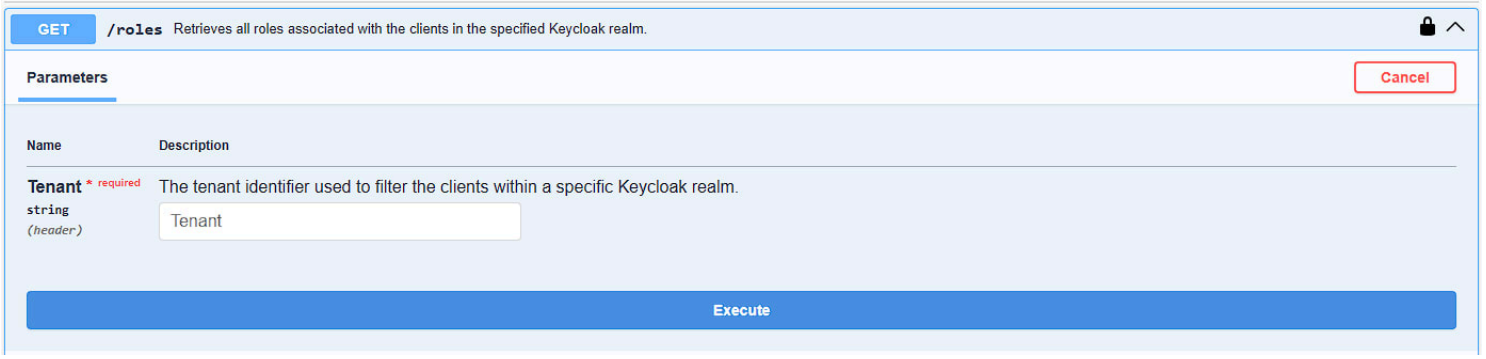
Responses

Code	Description
200	A list of roles associated with the clients in the specified realm
400	The request was invalid or could not be processed.
401	The request lacks valid authentication credentials.
403	The request was understood, but the server is refusing to fulfill it due to insufficient permissions.
500	An internal server error occurred during the processing of the request.

Header

Name	Located in	Description	Required	Schema
Tenant	header	The tenant identifier used to filter the clients within a specific Keycloak realm.	Yes	string

Definition



📝 How to Use the Endpoint

1. **Tenant Identification**:

- The term *tenant* in Feijuca represents the **realm name** within Keycloak where you'll be performing actions.
- You must specify the tenant name in the **HTTP header** to proceed.

⚙️ Endpoint specification

Method: POST

Path: /role

Summary:

Adds a new role to the client in the specified Keycloak realm.

Responses

Code	Description
201	The operation was successful, and the new role was created.
400	The request was invalid or could not be processed.
401	The request lacks valid authentication credentials.
403	The request was understood, but the server is refusing to fulfill it due to insufficient permissions.
500	An internal server error occurred during the processing of the request.

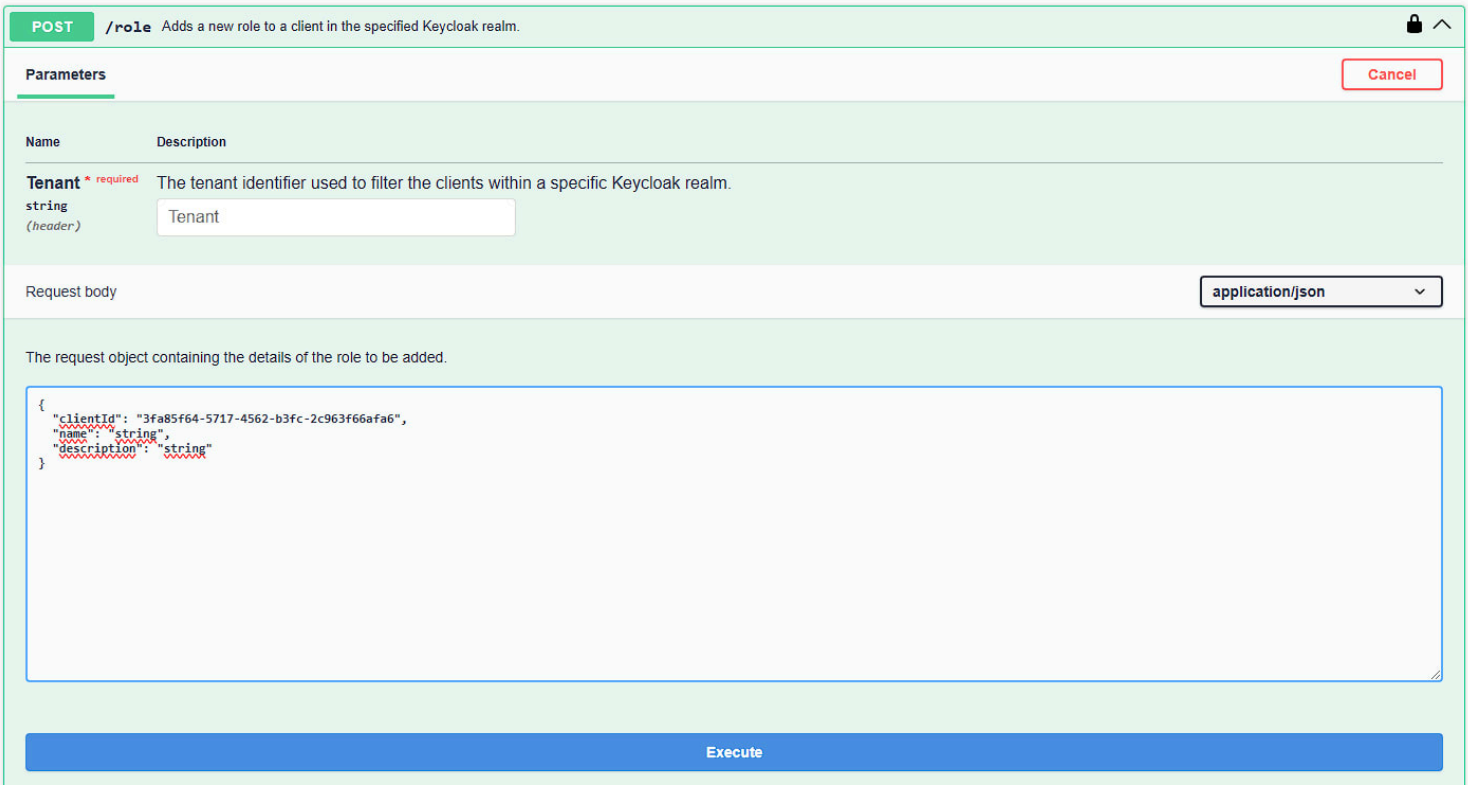
Header

Name	Located in	Description	Required	Schema
Tenant	header	The tenant identifier used to filter the clients within a specific Keycloak realm.	Yes	string

Body definition

Name	Located in	Description	Required	Schema
clientId	body	The clientId related to the unique identifier of the client that will have the new rule added.	Yes	Guid
name	body	The name is related to the rule name.	Yes	string
description	body	The description is related to information for that rule.	Yes	string

Definition



📝 How to Use the Endpoint

1. **Tenant Identification**:
    - The term *tenant* in Feijuca represents the **realm name** within Keycloak where you'll be performing actions.
    - You must specify the tenant name in the **HTTP header** to proceed.
2. **Body**:
    - After setting up the header, inform a valid body to insert a user. Example:

```
{
  "clientId": "Unique identifier of your customer(Guid)",
  "name": "test",
  "description": "description test"
}
```

⚙️ Endpoint specification

Method: GET

Path: /group/user

Summary:

Retrieves all users present in the specific group int the specified Keycloak realm.

Responses

Code	Description
200	The users in the group were successfully retrieved.
400	The request was invalid or could not be processed.
401	The request lacks valid authentication credentials.
403	The request was understood, but the server is refusing to fulfill it due to insufficient permissions.
500	An internal server error occurred during the processing of the request.

Header

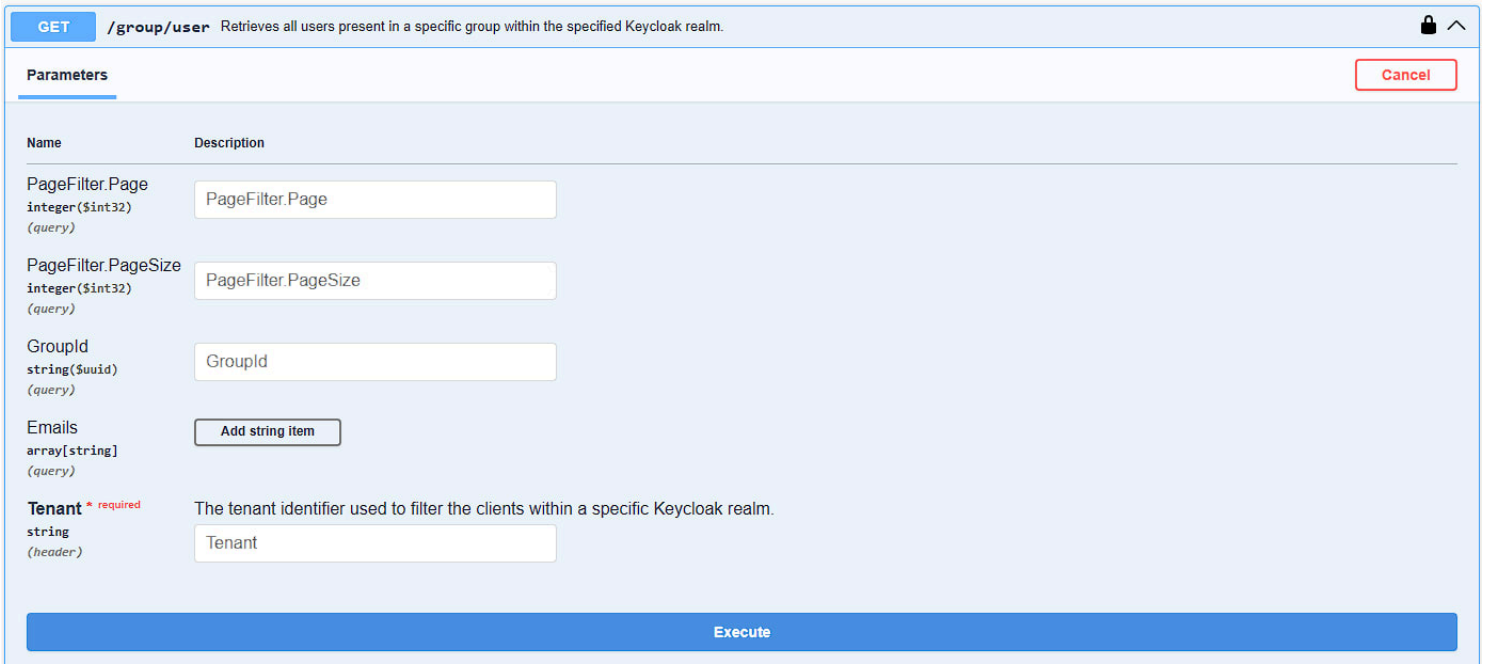
Name	Located in	Description	Required	Schema
Tenant	header	The tenant identifier used to filter the clients within a specific Keycloak realm.	Yes	string

Query params

Name	Located in	Description	Required	Schema
PageFilter.Page	query	The page related to the data that is being showed.	No	int
PageFilter.PageSize	query	Defines the page size, i.e. the maximum number of users that will be returned per page.	No	int

Name	Located in	Description	Required	Schema
GroupId	query	The group ids that can be used to filter determinated groups.	No	Guid
Emails	query	Allows you to filter users based on a list of email addresses.	No	string

Definition



📝 How to Use the Endpoint

1. **Tenant Identification**:
    - The term *tenant* in Feijuca represents the **realm name** within Keycloak where you'll be performing actions.
    - You must specify the tenant name in the **HTTP header** to proceed.
2. **Query Parameters**:
    - After setting up the header, use the available **query parameters** to customize your search based on your requirements.

🔍 Available Search Types

1. Get All Groups

- Fetches all users within the realm.

- The results are provided in a paginated format, allowing you to navigate through pages to explore the user list.

2. **Get Group by ID**

- Enables filtering by **Group IDs**.
- This will search the realm and return the group that matches the given ID.

3. **Get Users by Emails**

- Allows filtering by an **array of Emails**.
- The API will find and return users that match the user emails provided in the request.

⚙️ Endpoint specification

Method: GET

Path: /group/user

Summary:

Retrieves all users present in the specific group int the specified Keycloak realm.

Responses

Code	Description
200	The users in the group were successfully retrieved.
400	The request was invalid or could not be processed.
401	The request lacks valid authentication credentials.
403	The request was understood, but the server is refusing to fulfill it due to insufficient permissions.
500	An internal server error occurred during the processing of the request.

Header

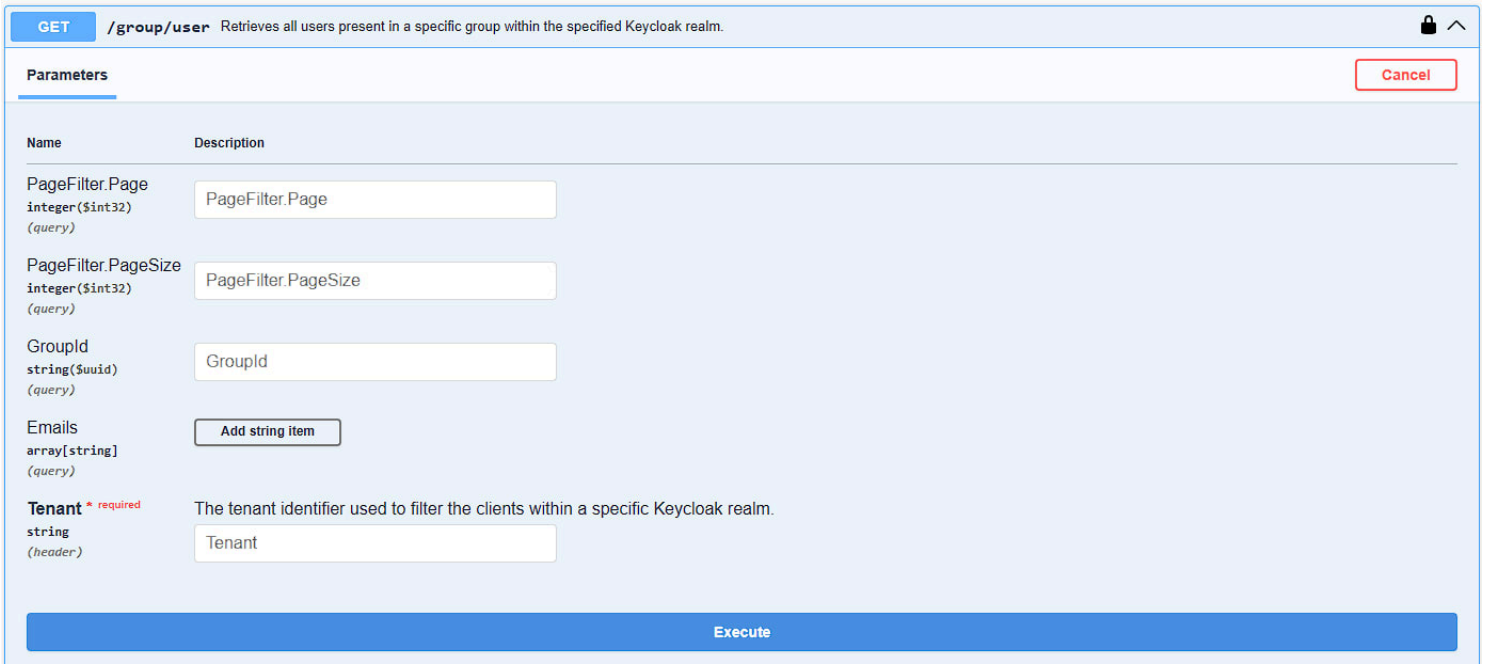
Name	Located in	Description	Required	Schema
Tenant	header	The tenant identifier used to filter the clients within a specific Keycloak realm.	Yes	string

Query params

Name	Located in	Description	Required	Schema
PageFilter.Page	query	The page related to the data that is being showed.	No	int
PageFilter.PageSize	query	Defines the page size, i.e. the maximum number of users that will be returned per page.	No	int

Name	Located in	Description	Required	Schema
GroupId	query	The group ids that can be used to filter determinated groups.	No	Guid
Emails	query	Allows you to filter users based on a list of email addresses.	No	string

Definition



📝 How to Use the Endpoint

1. **Tenant Identification**:
    - The term *tenant* in Feijuca represents the **realm name** within Keycloak where you'll be performing actions.
    - You must specify the tenant name in the **HTTP header** to proceed.
2. **Query Parameters**:
    - After setting up the header, use the available **query parameters** to customize your search based on your requirements.

🔍 Available Search Types

1. Get All Groups

- Fetches all users within the realm.

- The results are provided in a paginated format, allowing you to navigate through pages to explore the user list.

2. **Get Group by ID**

- Enables filtering by **Group IDs**.
- This will search the realm and return the group that matches the given ID.

3. **Get Users by Emails**

- Allows filtering by an **array of Emails**.
- The API will find and return users that match the user emails provided in the request.

⚙️ Endpoint specification

Method: POST

Path: /group/user

Summary:

Adds a user to the specific group in the specified Keycloak realm.

Responses

Code	Description
204	The user was successfully added to the group.
400	The request was invalid or could not be processed.
401	The request lacks valid authentication credentials.
403	The request was understood, but the server is refusing to fulfill it due to insufficient permissions.
500	An internal server error occurred during the processing of the request.

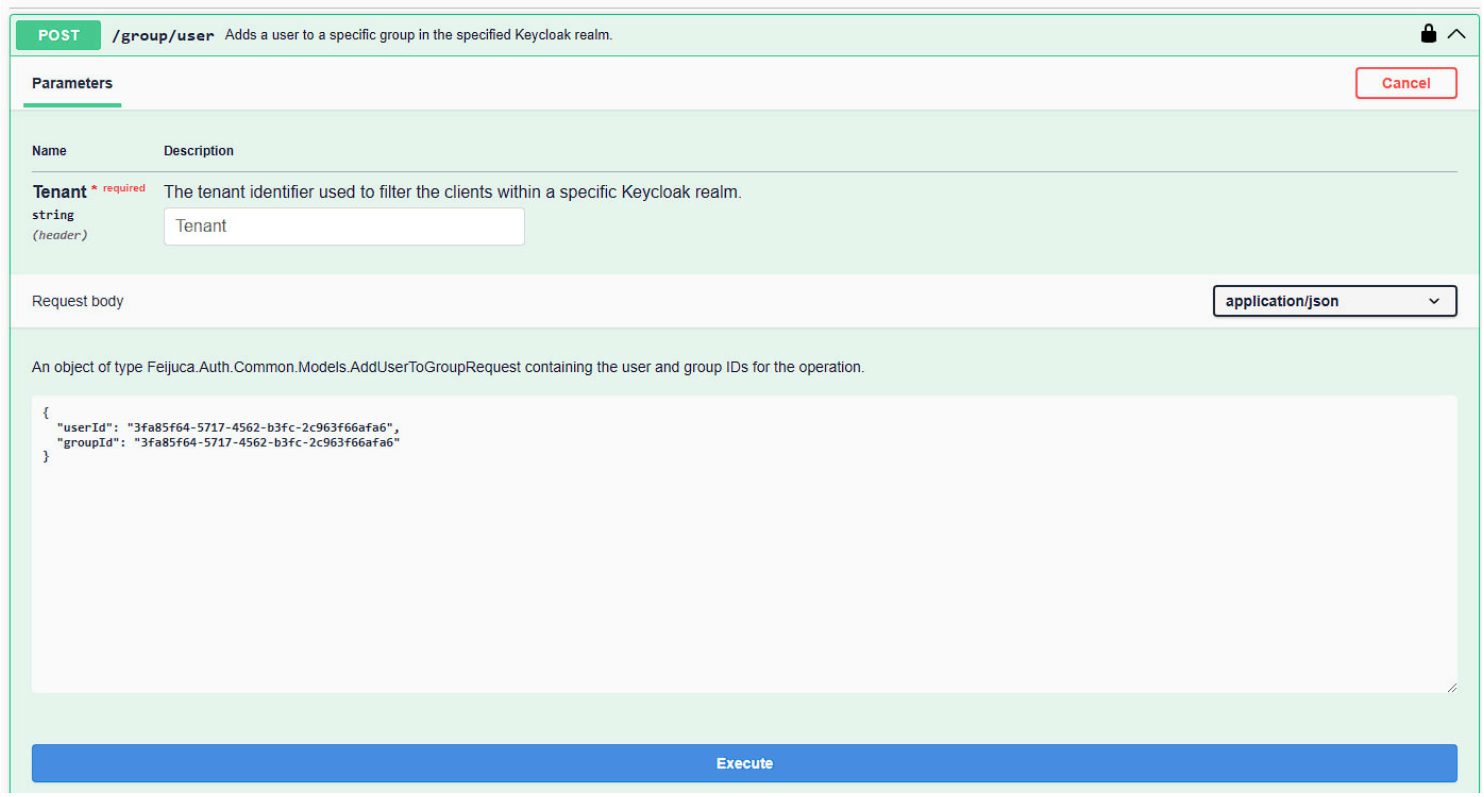
Header

Name	Located in	Description	Required	Schema
Tenant	header	The tenant identifier used to filter the clients within a specific Keycloak realm.	Yes	string

Body definition

Name	Located in	Description	Required	Schema
userId	body	The user Id related to the user who will be added to the group.	Yes	string
groupId	body	The groupId related to the group where the user will be added.	Yes	string

Definition



📝 How to Use the Endpoint

1. **Tenant Identification**:
    - The term *tenant* in Feijuca represents the **realm name** within Keycloak where you'll be performing actions.
    - You must specify the tenant name in the **HTTP header** to proceed.
2. **Body**:
    - After setting up the header, inform a valid body to insert a user. Example:

```
{
  "userId": "Unique user identifier",
  "groupId": "Unique group identifier"
}
```

⚙️ Endpoint specification

Method: DELETE

Path: /group/user

Summary:

Removes a user from a specific group within the specified Keycloak realm.

Responses

Code	Description
204	The user was successfully removed from the group.
400	The request was invalid or could not be processed.
401	The request lacks valid authentication credentials.
403	The request was understood, but the server is refusing to fulfill it due to insufficient permissions.
500	An internal server error occurred during the processing of the request.

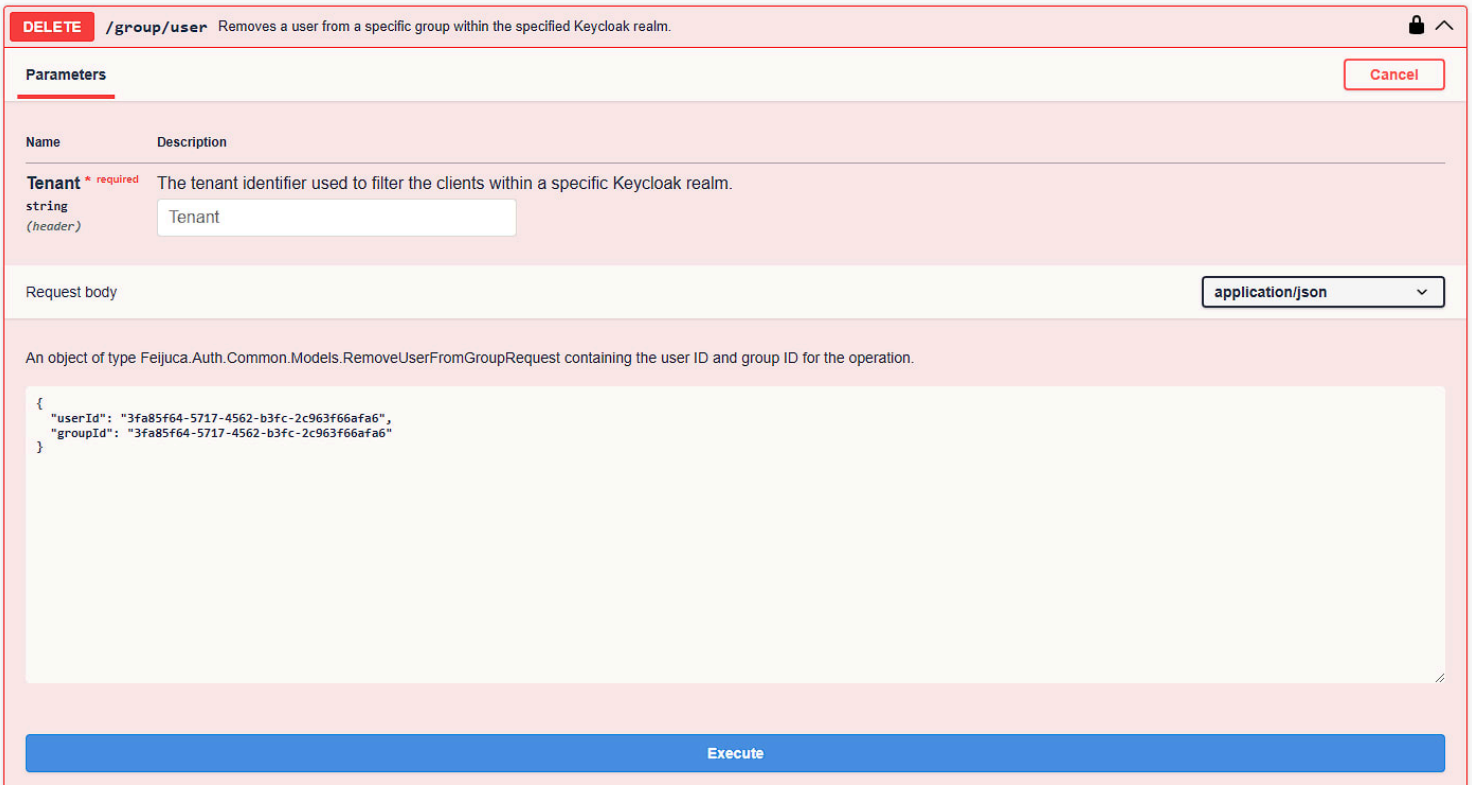
Header

Name	Located in	Description	Required	Schema
Tenant	header	The tenant identifier used to filter the clients within a specific Keycloak realm.	Yes	string

Body definition

Name	Located in	Description	Required	Schema
userId	body	The userId related to the user that will be deleted.	Yes	Guid
groupId	body	The groupId related to the group from which the user will be deleted.	Yes	Guid

Definition



How to Use the Endpoint

1. **Tenant Identification**:
    - The term *tenant* in Feijuca represents the **realm name** within Keycloak where you'll be performing actions.
    - You must specify the tenant name in the **HTTP header** to proceed.
2. **Body**:
    - After setting up the header, inform a valid body to insert a user. Example:

```
{
  "userId": "Unique user identifier",
  "groupId": "Unique group identifier"
}
```

⚙️ Endpoint specification

Method: GET

Path: /group/{id}/roles

Summary:

Retrivies the roles associated with a specific group in the specified Keycloak realm.

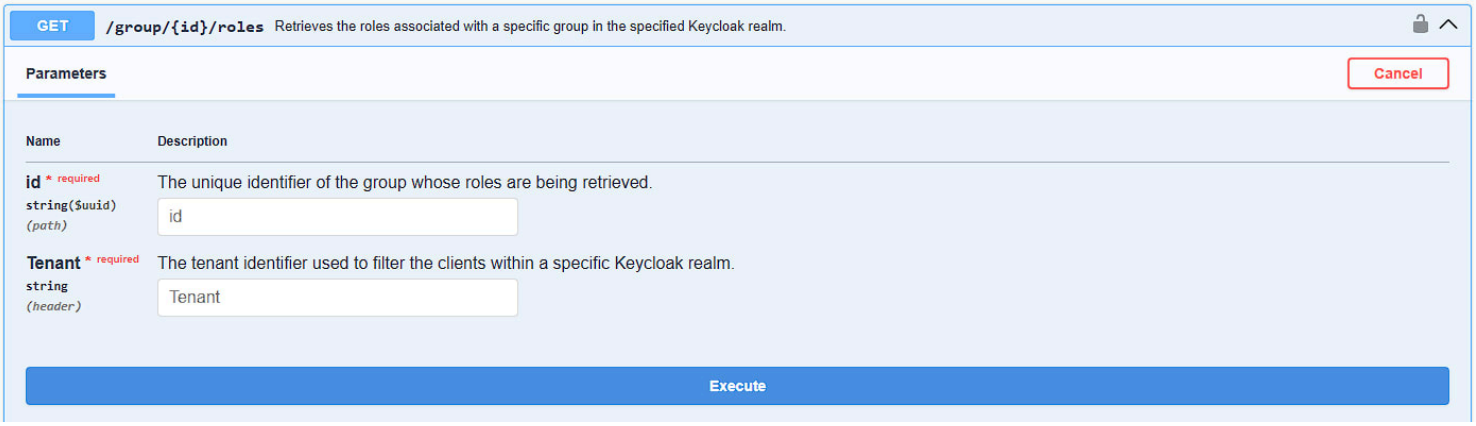
Responses

Code	Description
200	The roles associated with the group were successfully retrieved.
400	The request was invalid or could not be processed.
401	The request lacks valid authentication credentials.
403	The request was understood, but the server is refusing to fulfill it due to insufficient permissions.
500	An internal server error occurred during the processing of the request.

Header

Name	Located in	Description	Required	Schema
id	patch	The unique identifier of the group whose roles are being retrieved.	Yes	Guid
Tenant	header	The tenant identifier used to filter the clients within a specific Keycloak realm.	Yes	string

Definition



How to Use the Endpoint

1. **Tenant Identification**:
   - The term *tenant* in Feijuca represents the **realm name** within Keycloak where you'll be performing actions.
   - You must specify the tenant name in the **HTTP header** to proceed.
2. **Id Unique Identification**:
   - The term *id* represents the **unique identifier of the group** whose roles are being retrieved.
   - You must specify the Group Id in the **HTTP path** to proceed.

⚙️ Endpoint specification

Method: GET

Path: /group/{id}/roles

Summary:

Retrivies the roles associated with a specific group in the specified Keycloak realm.

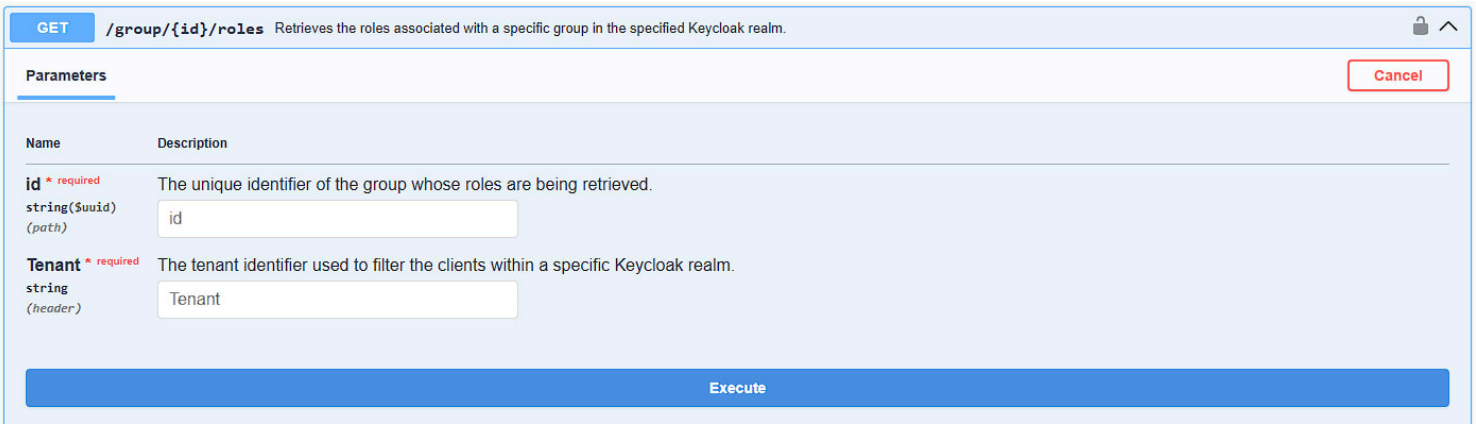
Responses

Code	Description
200	The roles associated with the group were successfully retrieved.
400	The request was invalid or could not be processed.
401	The request lacks valid authentication credentials.
403	The request was understood, but the server is refusing to fulfill it due to insufficient permissions.
500	An internal server error occurred during the processing of the request.

Header

Name	Located in	Description	Required	Schema
id	patch	The unique identifier of the group whose roles are being retrieved.	Yes	Guid
Tenant	header	The tenant identifier used to filter the clients within a specific Keycloak realm.	Yes	string

Definition



How to Use the Endpoint

1. **Tenant Identification**:
    - The term *tenant* in Feijuca represents the **realm name** within Keycloak where you'll be performing actions.
    - You must specify the tenant name in the **HTTP header** to proceed.
2. **Id Unique Identification**:
    - The term *id* represents the **unique identifier of the group** whose roles are being retrieved.
    - You must specify the Group Id in the **HTTP path** to proceed.

⚙️ Endpoint specification

Method: POST

Path: /group/{id}/role

Summary:

Adds a role to a specific group in the specified Keycloak realm.

Responses

Code	Description
201	The role was successfully added to the group.
400	The request was invalid or could not be processed.
401	The request lacks valid authentication credentials.
403	The request was understood, but the server is refusing to fulfill it due to insufficient permissions.
500	An internal server error occurred during the processing of the request.

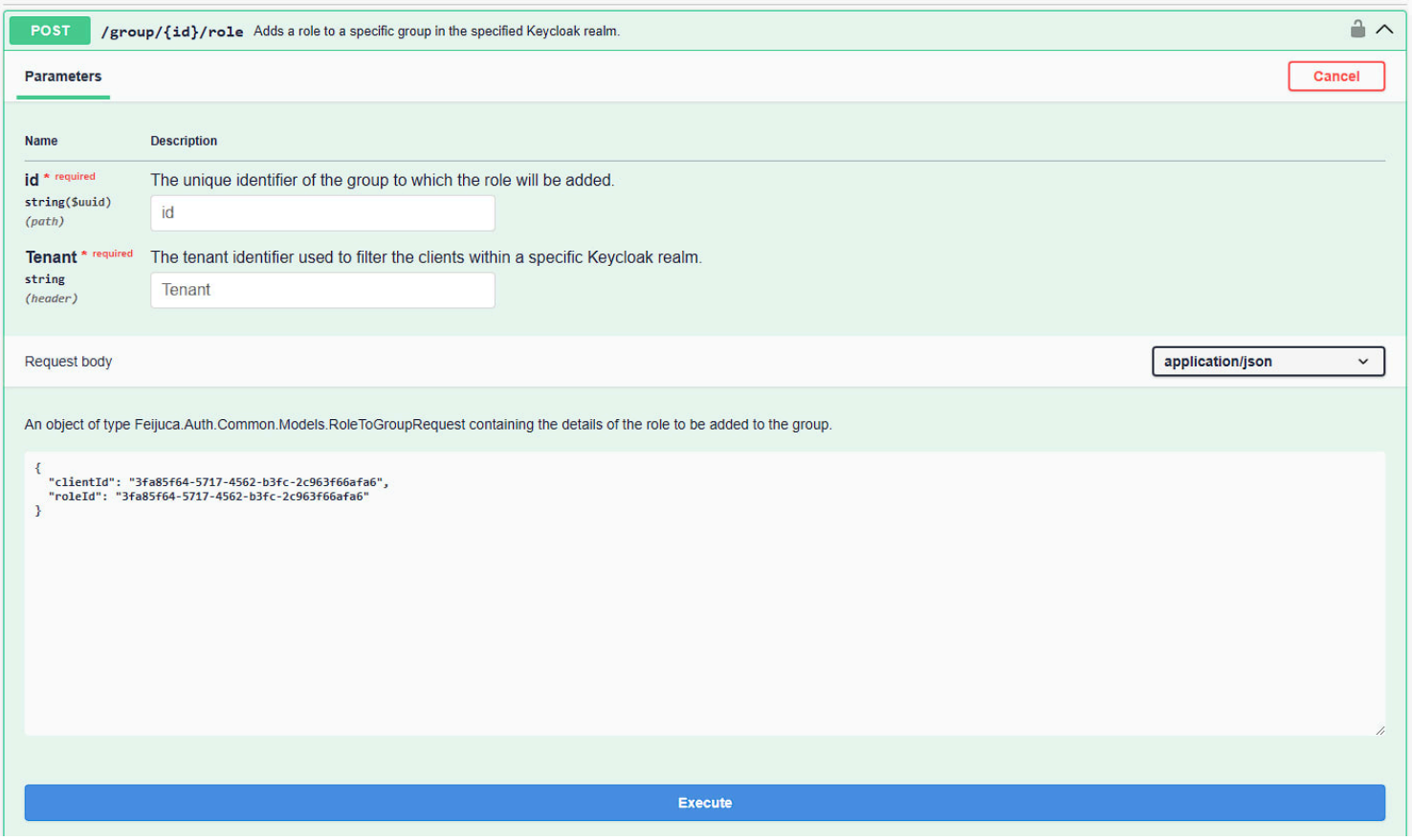
Header

Name	Located in	Description	Required	Schema
ID	patch	The unique identifier of the group to which the role will be added.	Yes	Guid
Tenant	header	The tenant identifier used to filter the clients within a specific Keycloak realm.	Yes	string

Body definition

Name	Located in	Description	Required	Schema
clientId	body	The clientId related to the unique identifier of the client that will have the new rule added.	Yes	Guid
roleId	body	The roleId related to the unique identifier of the rule to be added to the Group.	Yes	Guid

Definition



📝 How to Use the Endpoint

1. **Tenant Identification**:
    - The term *id* represents the **realm name** within Keycloak where you'll be performing actions.
    - You must specify the tenant name in the **HTTP header** to proceed.
2. **Id Unique Identification**:
    - The term *id* in Feijuca represents the **unique identifier of the group** within Keycloak where you will execute the actions.
    - You must specify the Group Id in the **HTTP path** to proceed.
3. **Body**:
    - After setting up the header, inform a valid body to insert a user. Example:

```
{
  "clientId": "Unique client identifier",
  "roleId": "Unique role identifier"
}
```

⚙️ Endpoint specification

Method: DELETE

Path: /group/{groupid}/role/{roleid}

Summary:

Remove a role from a specific group in the specified Keycloak realm.

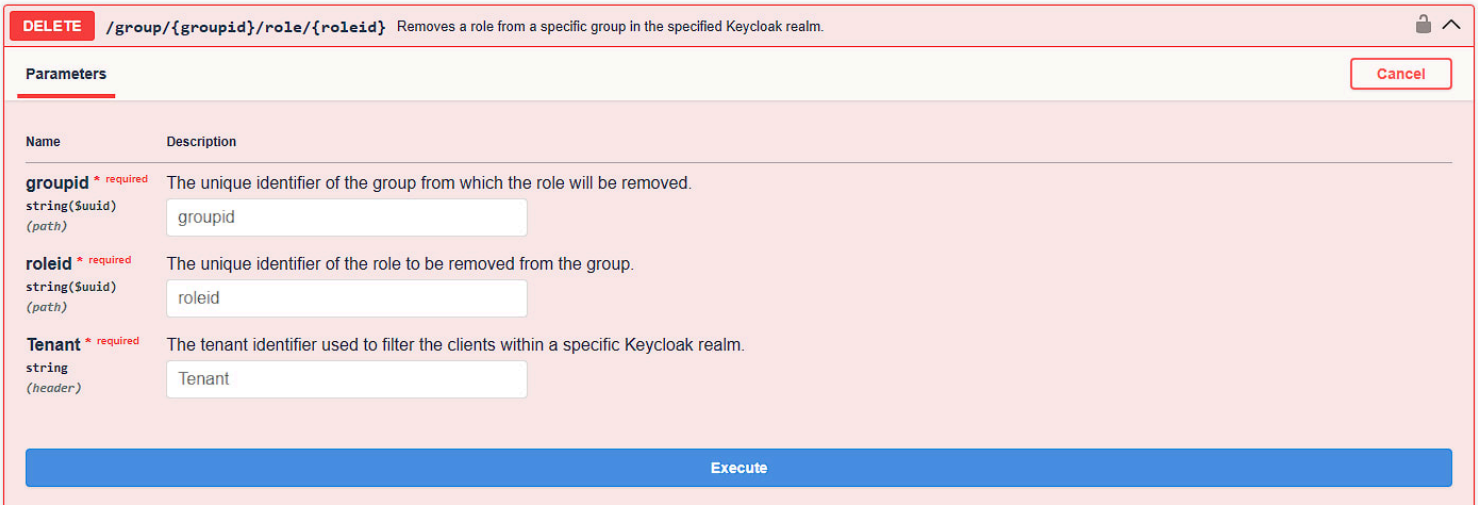
Responses

Code	Description
204	The role was successfully removed from the group.
400	The request was invalid or could not be processed.
401	The request lacks valid authentication credentials.
403	The request was understood, but the server is refusing to fulfill it due to insufficient permissions.
500	An internal server error occurred during the processing of the request.

Header

Name	Located in	Description	Required	Schema
groupid	patch	The unique identifier of the group from which the role will be removed.	Yes	Guid
roleid	patch	The unique identifier of the role to be removed from the group.	Yes	Guid
Tenant	header	The tenant identifier used to filter the clients within a specific Keycloak realm.	Yes	string

Definition



How to Use the Endpoint

1. **Tenant Identification**:
   - The term *tenant* in Feijuca represents the **realm name** within Keycloak where you'll be performing actions.
   - You must specify the tenant name in the **HTTP header** to proceed.
2. **Id Unique group identification**:
   - The term *groupid* represents the **unique identifier of the group** whose roles are being retrieved.
   - You must specify the Group Id in the **HTTP path** to proceed.
3. **Id Unique rule identification**:
   - The term *id* represents the **unique identifier of the role** whose role is being deleted from the group.
   - You must specify the Group Id in the **HTTP path** to proceed.

⚙️ Endpoint specification

Method: POST

Path: /user

Summary:

Recover all clients registered in the realm.

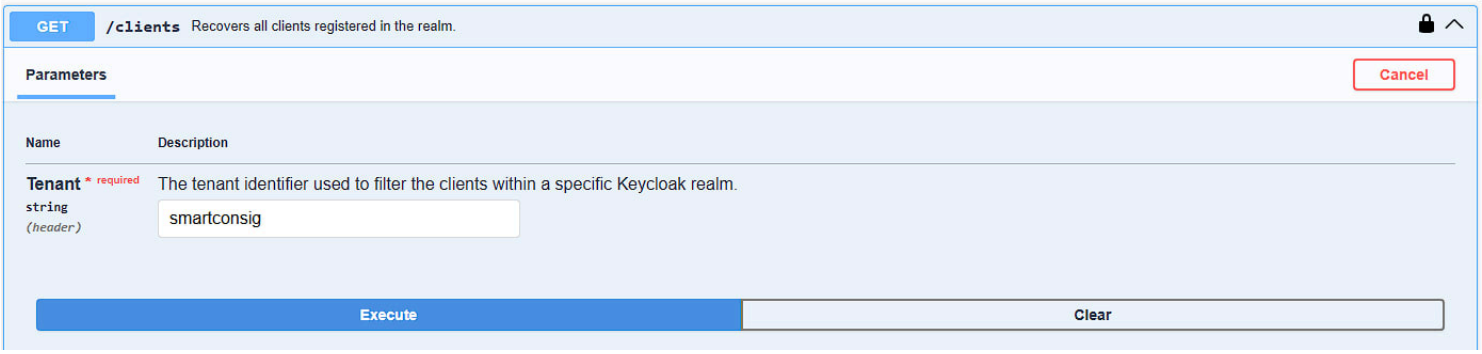
Responses

Code	Description
200	The operation was successful, and the list of clients is returned.
400	The request was invalid or could not be processed.
401	The request lacks valid authentication credentials.
403	The request was understood, but the server is refusing to fulfill it due to insufficient permissions.
500	An internal server error occurred during the processing of the request.

Header

Name	Located in	Description	Required	Schema
Tenant	header	The tenant identifier used to filter the clients within a specific Keycloak realm.	Yes	string

Definition



📝 How to Use the Endpoint

1. **Tenant Identification**:

- The term *tenant* in Feijuca represents the **realm name** within Keycloak where you'll be performing actions.
- You must specify the tenant name in the **HTTP header** to proceed.

⚙️ Endpoint specification

Method: POST

Path: /user

Summary:

Recover all clients registered in the realm.

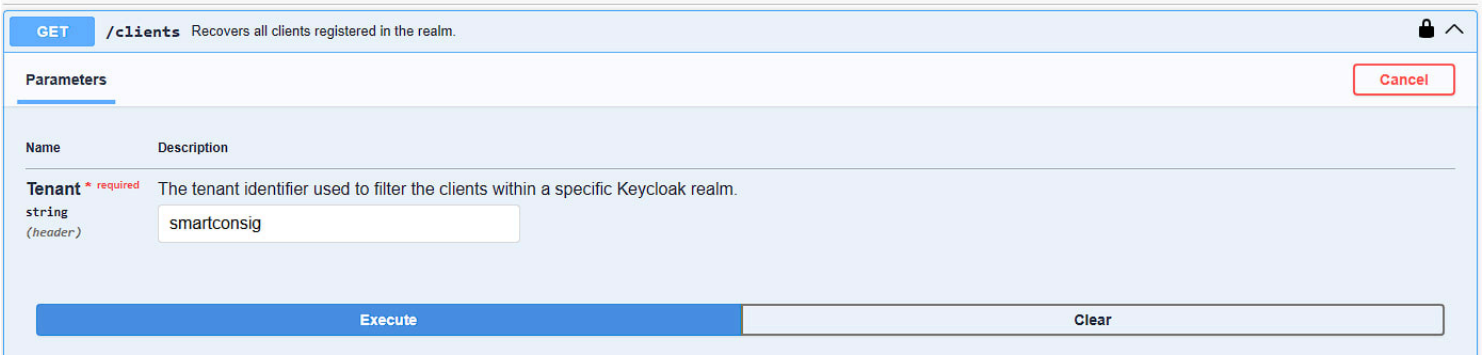
Responses

Code	Description
200	The operation was successful, and the list of clients is returned.
400	The request was invalid or could not be processed.
401	The request lacks valid authentication credentials.
403	The request was understood, but the server is refusing to fulfill it due to insufficient permissions.
500	An internal server error occurred during the processing of the request.

Header

Name	Located in	Description	Required	Schema
Tenant	header	The tenant identifier used to filter the clients within a specific Keycloak realm.	Yes	string

Definition



📝 How to Use the Endpoint

1. **Tenant Identification**:

- The term *tenant* in Feijuca represents the **realm name** within Keycloak where you'll be performing actions.
- You must specify the tenant name in the **HTTP header** to proceed.