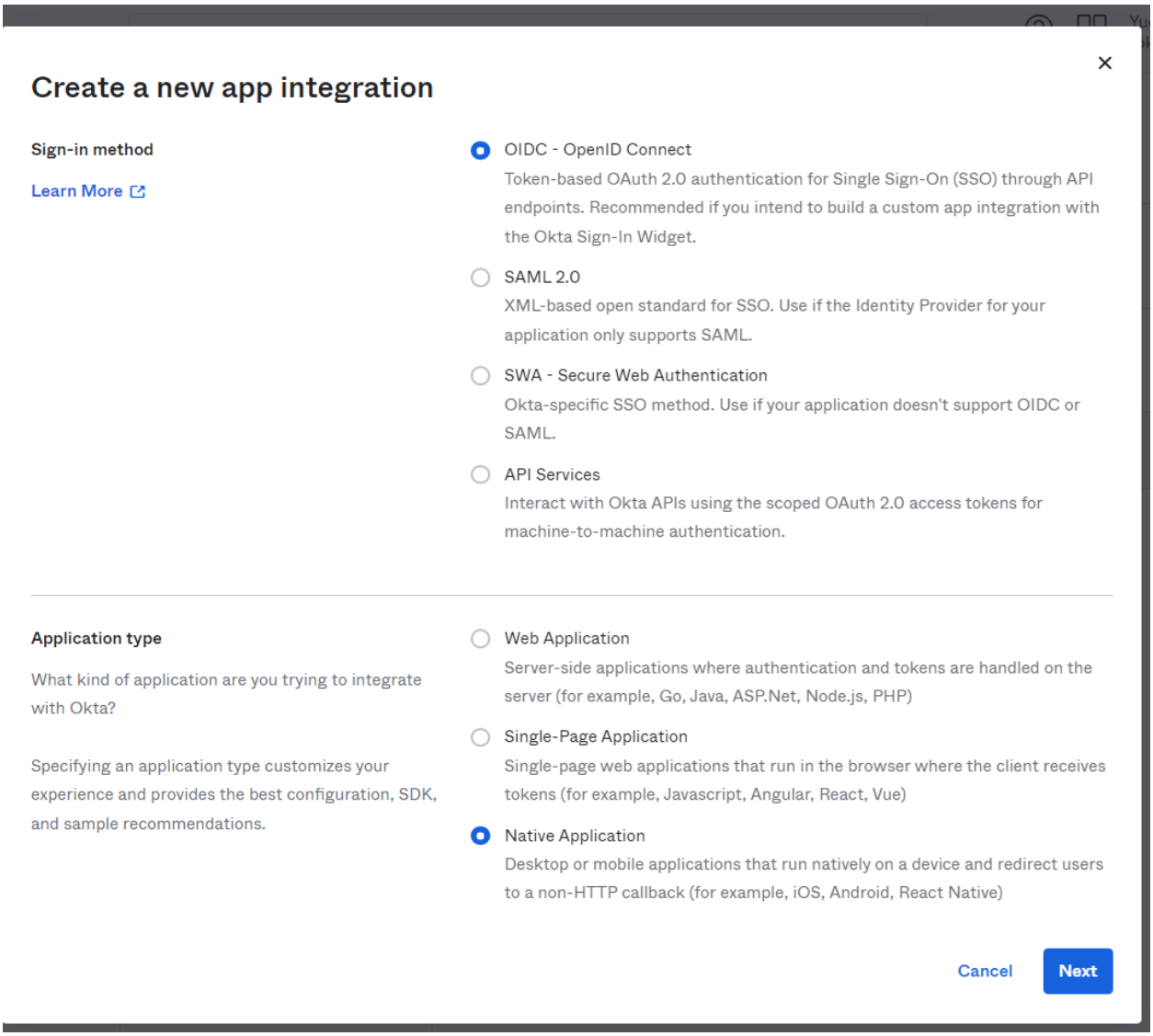
Setup of the OKTA OIDC platform on Landslide for User Authentication.

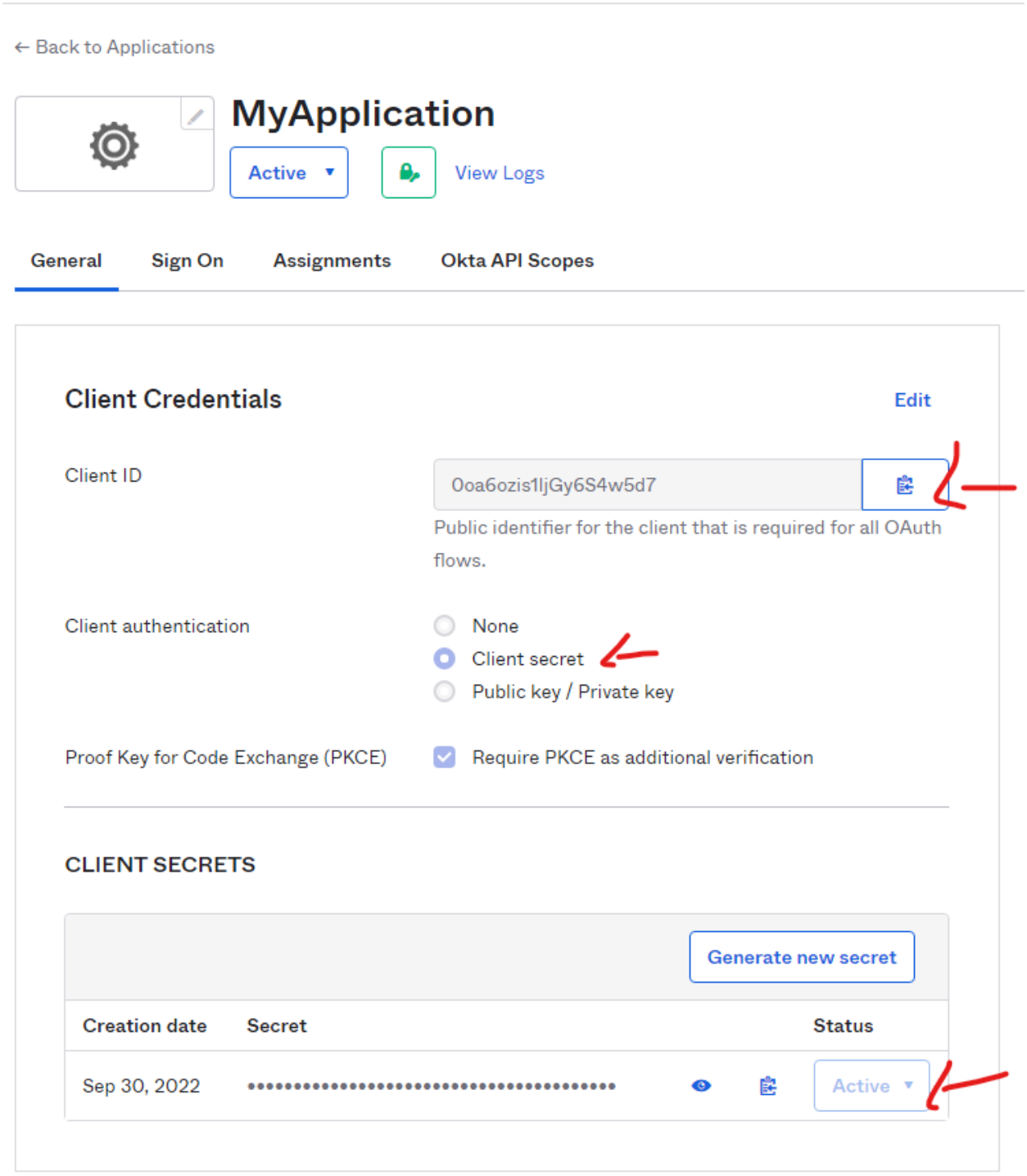
See Admin -> Server Settings (User Authentication)

#1. Create a new application with OIDC and native application type:



In this example, MyApplication is used as the application name to store the Client ID and Client Secrets :

Select **Client secret** for Client authentication :



Select **Resource Owner Password** for Grant Type :

General Settings

Edit

APPLICATION

App integration name	MyApplication
Application type	Native
Grant type	Client acting on behalf of a user

- [x] Authorization Code
- [ ] Interaction Code
- [x] Refresh Token
- [x] Resource Owner Password
- [ ] SAML 2.0 Assertion
- [ ] Device Authorization
- [ ] Token Exchange
- [ ] Implicit (hybrid)

Refresh Token

Refresh token behavior

- ( ) Rotate token after every use
- (•) Use persistent token

USER CONSENT

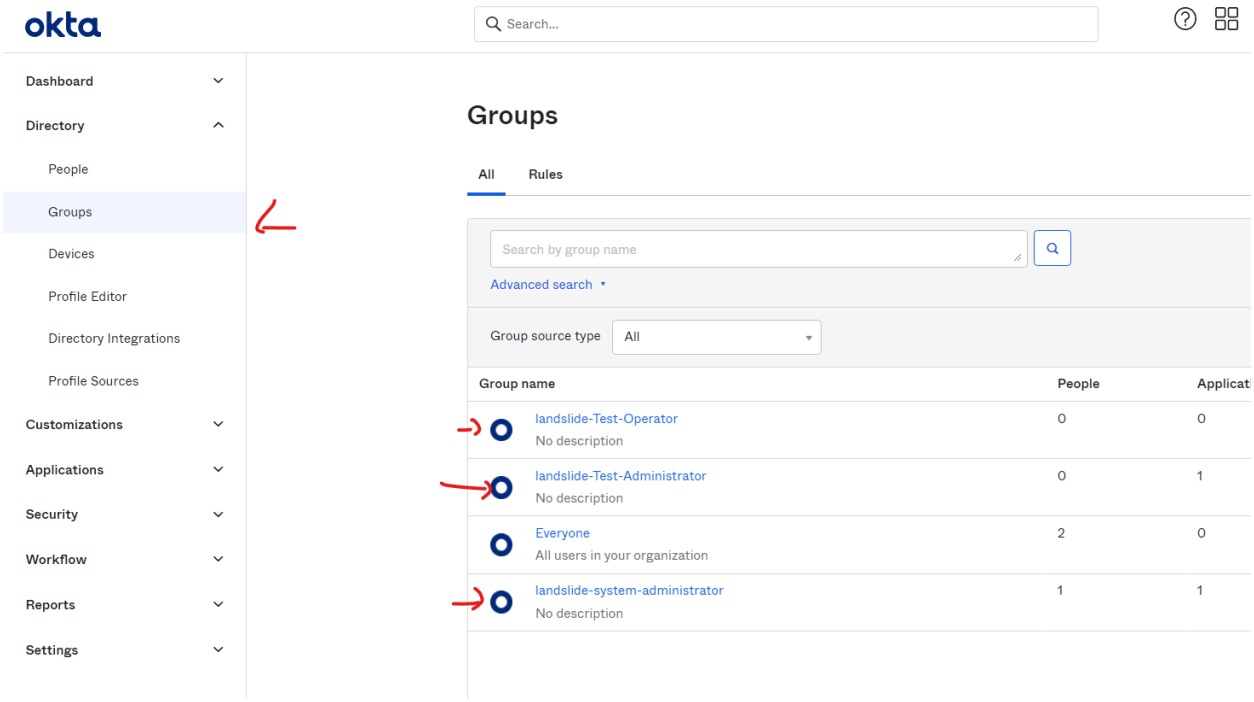
User consent

- [x] Require consent

Terms of Service URI

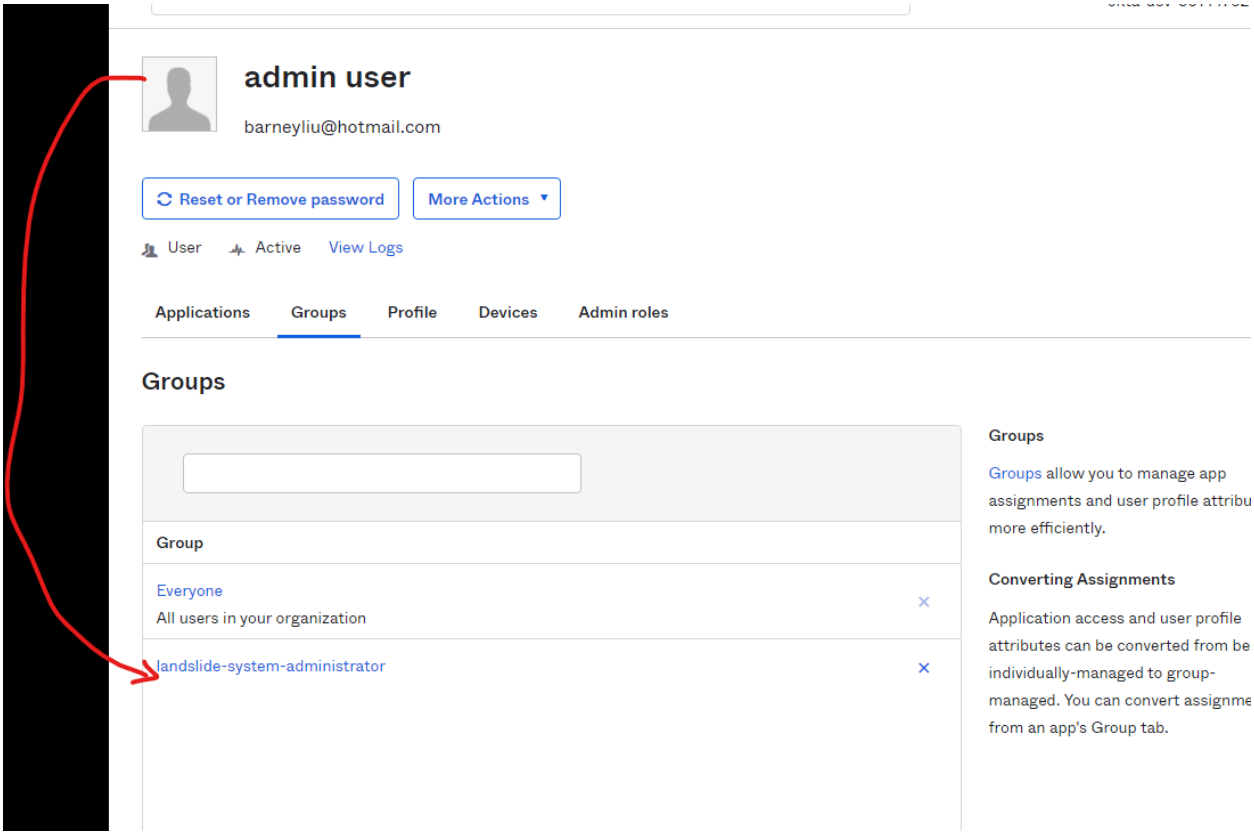
Policy URI

#2 Create the following groups in the OKTA platform



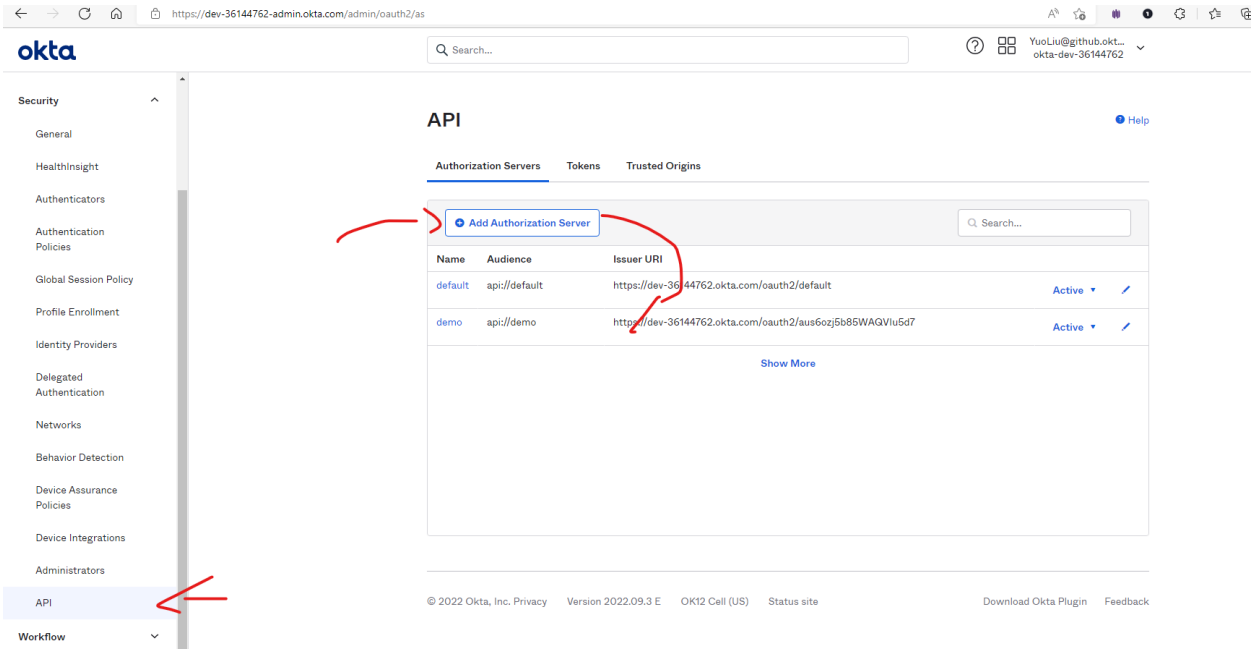
And assign users to the corresponding groups.

For instance, admin user is assigned to the landslide-system- administrator :

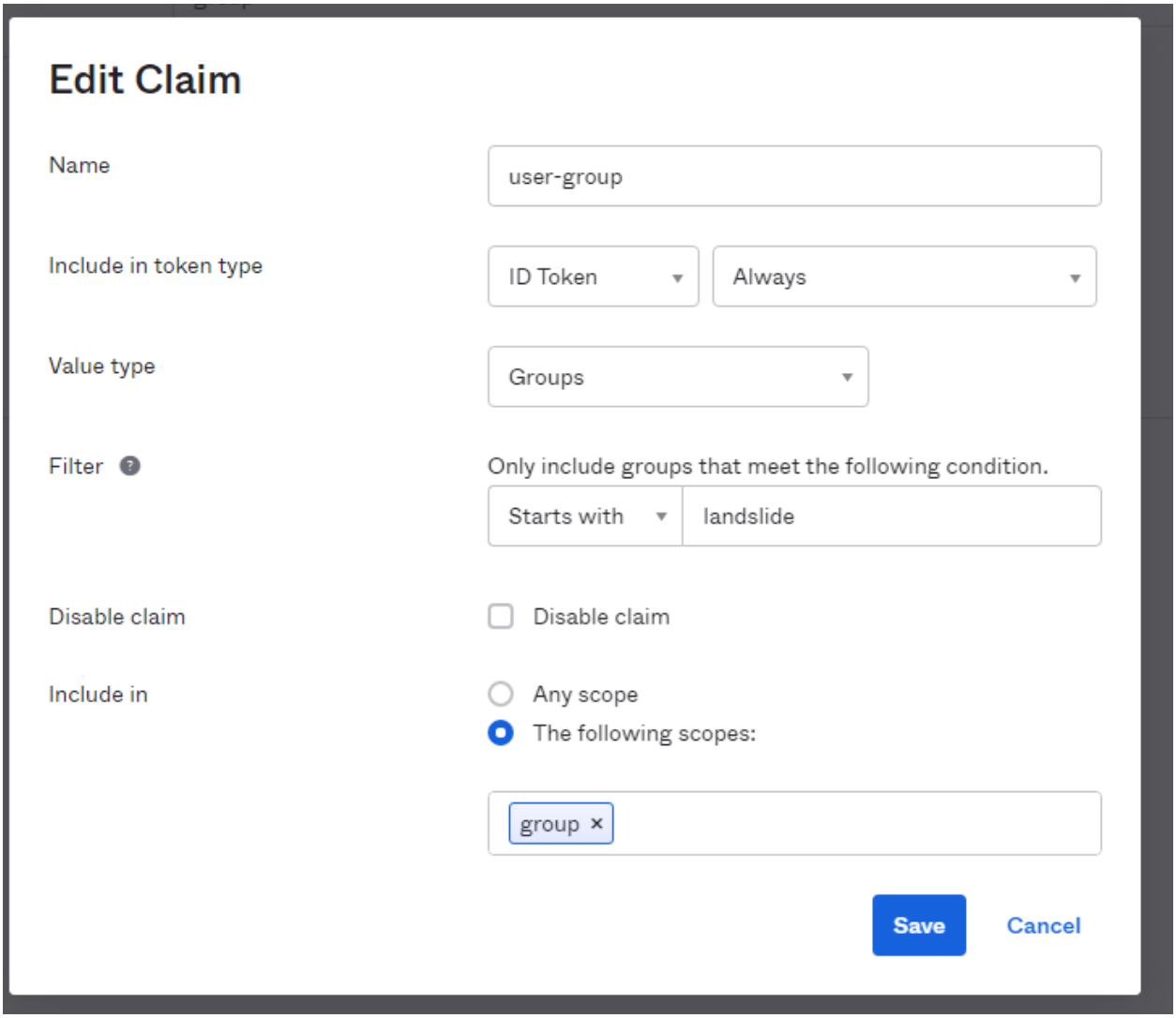


Add MyApplication to the Added group.

Enter Security/API to create a new Authorization Servers. You can use the default one as shown below:



Enter the new added auth server, add a new claim to allow Landslide to retrieve the information of user group :



Validation:

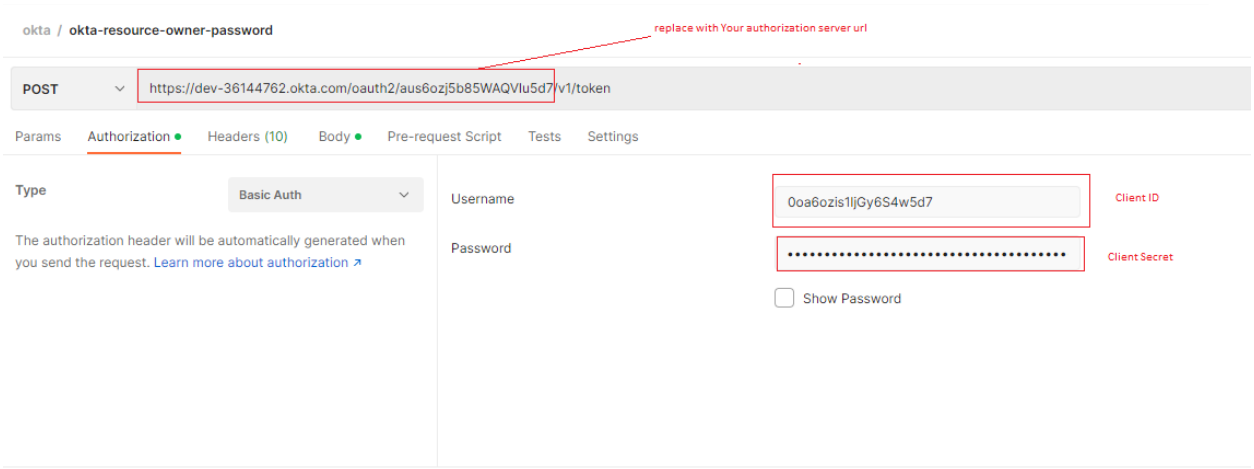
After the above configuration is done, you can use the following postman collection to validate the configuration :

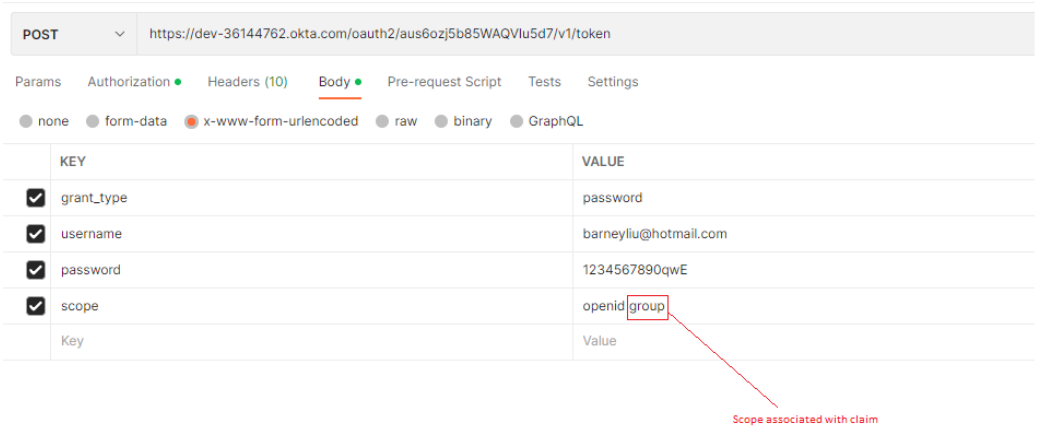

```
{
	"info": {
		"_postman_id": "6005697d-bc93-4c56-9934-9eb92f47ba70",
		"name": "okta",
		"schema": "https://schema.getpostman.com/json/collection/v2.1.0/collection.json"
	},
	"item": [
		{
			"name": "okta-resource-owner-password",
			"request": {
				"auth": {
					"type": "basic",
					"basic": [
						{
							"key": "password",
							"value": "tgp5a7ybAvduO69whB5rsUsoy6YVrNZRWYA20ZDY",
							"type": "string"
						},
						{
							"key": "username",
```

```
                "value": "0oa6ozis1ljGy6S4w5d7",
                "type": "string"
            }
        ]
    },
    "method": "POST",
    "header": [],
    "body": {
        "mode": "urlencoded",
        "urlencoded": [
            {
                "key": "grant_type",
                "value": "password",
                "type": "default"
            },
            {
                "key": "username",
                "value": "barneyliu@hotmail.com",
                "type": "default"
            },
            {
                "key": "password",
                "value": "1234567890qwE",
```

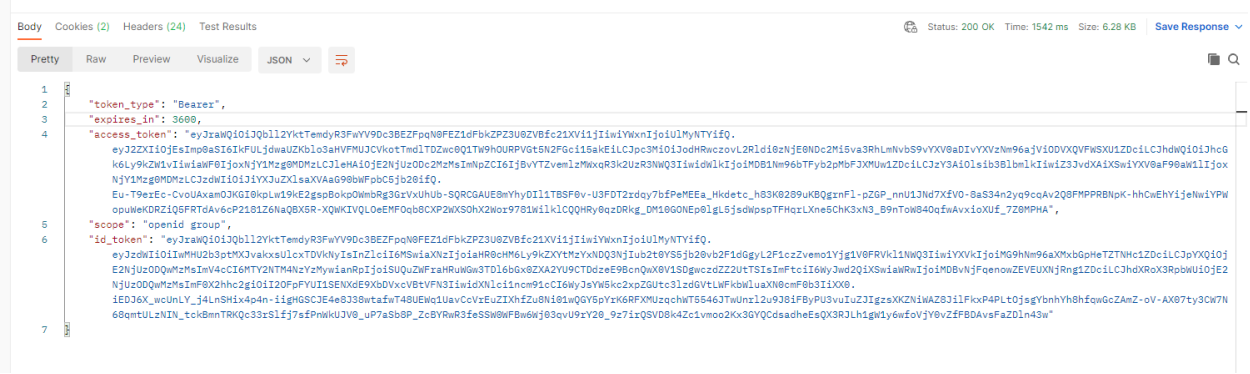
```
                    "type": "default"
                },
                {
                    "key": "scope",
                    "value": "openid group",
                    "type": "default"
                }
            ]
        },
        "url": {
            "raw": "https://dev-36144762.okta.com/oauth2/aus6ozj5b85WAQVIu5d7/v1/token",
            "protocol": "https",
            "host": [
                "dev-36144762",
                "okta",
                "com"
            ],
            "path": [
                "oauth2",
                "aus6ozj5b85WAQVIu5d7",
                "v1",
                "token"
            ]
```

```
                    }
                },
                "response": []
            }
        ]
}
```


okta / okta-resource-owner-password
replace with Your authorization server url
POST
https://dev-36144762.okta.com/oauth2/aus6ozj5b85WAQVIu5d7/v1/token
Params
Authorization
Headers (10)
Body
Pre-request Script
Tests
Settings
Type
Basic Auth
The authorization header will be automatically generated when you send the request. Learn more about authorization ↗
Username
0oa6ozis1ljGy6S4w5d7
Client ID
Password
Client Secret
Show Password


POST
https://dev-36144762.okta.com/oauth2/aus6ozj5b85WAQVIu5d7/v1/token
Params
Authorization
Headers (10)
Body
Pre-request Script
Tests
Settings
none
form-data
x-www-form-urlencoded
raw
binary
GraphQL
KEY
VALUE
grant_type
password
username
barneyliu@hotmail.com
password
1234567890qwE
scope
openid group
Key
Value
Scope associated with claim

Result:

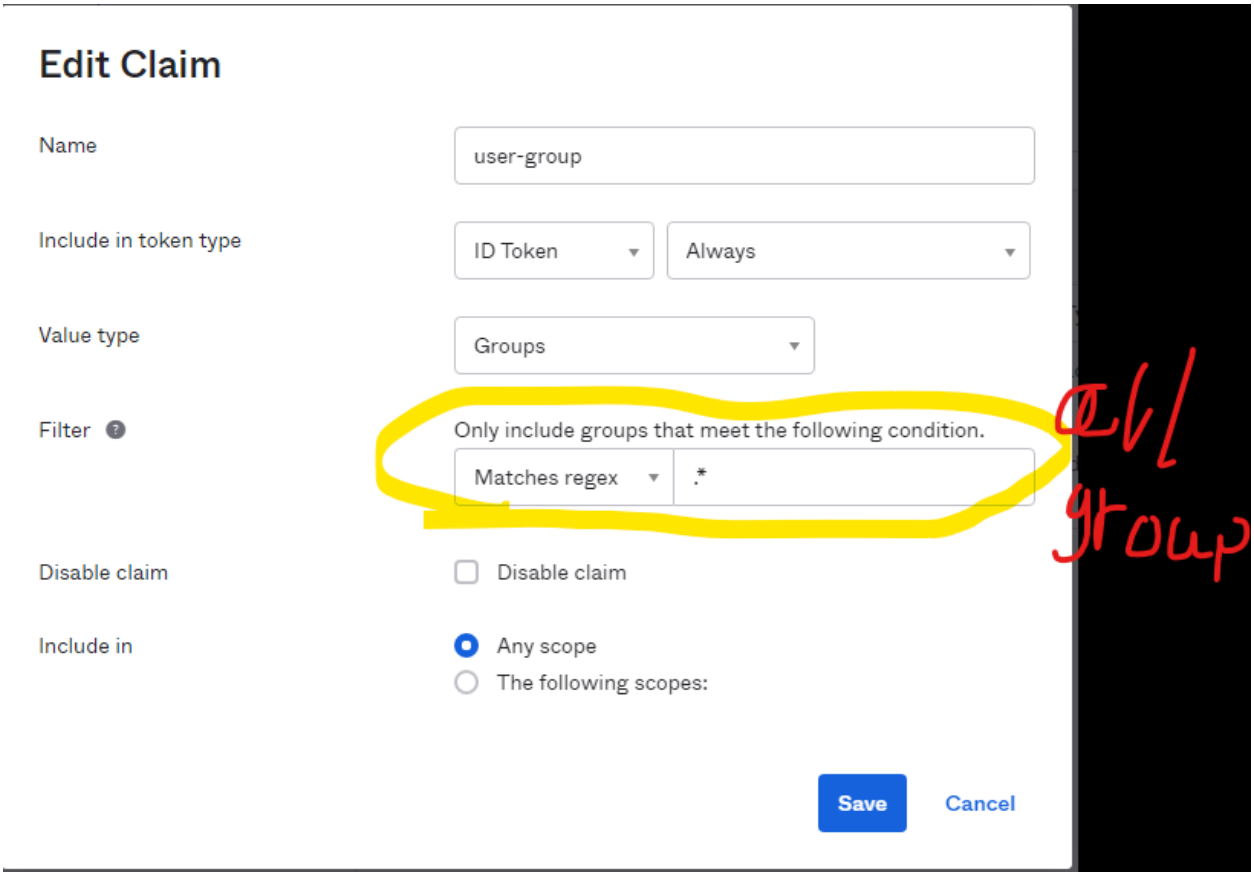


In this example, we put the claim in the ID_TOKEN, if we decode the ID_TOKEN, we get the user-group claim information.

Another way to configure the OKTA Authentication server is to return the user group for landslide application.

Okta allows several ways to define a claim, we recommend using regex.

Example: define the claim by using regex



the response will return the user-group


```
"user-group": [
  "Everyone",
  "landslide-system-administrator"
]
```


as