



Use Oracle Identity Cloud Service's Software Development Kit (SDK) for Authentication in Python Web Applications



Before You Begin

This 15 minutes tutorial shows you how to use Oracle Identity Cloud Service's Software Development Kit (SDK) for the Python programming language to enable a sample web application to authenticate with Oracle Identity Cloud Service.

Series

This is a tutorial in the **Use Oracle Identity Cloud Service's Software Development Kit (SDK)** series. Read them in any order.

- [Use Oracle Identity Cloud Service's Software Development Kit \(SDK\) for Authentication in Java Web Applications](#)
- [Use Oracle Identity Cloud Service's Software Development Kit \(SDK\) for Authentication in Node.js Web Applications](#)
- [Use Oracle Identity Cloud Service's Software Development Kit \(SDK\) for Authentication in Python Web Applications](#)
- [Use Oracle Identity Cloud Service's Software Development Kit \(SDK\) for Authentication in .NET Web Applications](#)

Background

Oracle Identity Cloud Service provides a Software Development Kit (SDK) that you download from the console, and you can use to integrate Python web applications with Oracle Identity Cloud Service.

The Python SDK is available as two python files `IdcsClient.py` and `Constants.py`, which must be included in the web application.

To help you understand how to use the SDK, this tutorial uses a sample web application as a reference.

Important: The sample web application isn't meant to be published

to production and isn't concerned about the language's specific best practices, such as data handling, patterns, security, and so on. The sole purpose of the sample web application is to address the recommended approach to integrate Oracle Identity Cloud Service and a custom application using the SDK.


What Do You Need?

- A basic knowledge of Django framework and Python programming language to understand the code logic presented in this tutorial.
- To install [Python version 2.7.x](#). You also need the Python binary folder added to your local path. Then install [Django](#) framework, by running the following command line `pip install Django`

Note: Depending on the operational system you use the python installation may not come with pip. You need to install it separately.

- To download the [Python sample web application](#) as zip file, and extract its content to the `c:\temp` folder of your desktop.
- Access to an instance of Oracle Identity Cloud Service, and rights to download the SDK from the console and to add a confidential application.

1 Download the SDK in the Sample Web Application

1. In the Identity Cloud Service console, expand the **Navigation Drawer** , click **Settings**, and then click **Downloads**. The list of files to download appears.
2. Click **Download** to download the **Identity Cloud Service SDK for Python** SDK file, and save the zip file.
3. Open the SDK zip file and extract the files under the `src` folder into the sample web application source code's `c:\temp\python\sampleapp\` folder. The source code structure of the sample web application must be similar to the one below.

```
c:\temp\python\sampleapp\  
    __pycache__  
    migrations\  
    static\  
    templates\  

```

```
__init__.py
admin.py
apps.py
Constants.py
IdcsClient.py
models.py
tests.py
urls.py
views.py
```


4. Extract both `README.txt` and `requirements.txt` files to a temporary folder and run the following command line to install the required third-party libraries.

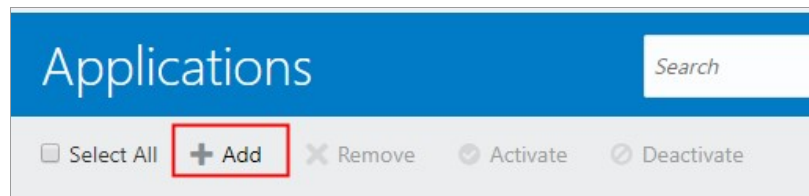
```
pip install -r requirements.txt
```

Note: If you're behind a proxy, then set up pip's proxy before running the command line.

2 Register the Sample Web Application with Oracle Identity Cloud Service


The sample web application needs a Client ID and Secret to establish communication with Oracle Identity Cloud Service. You also need to configure API permissions that must be granted to the sample web application. This section explains how to register the sample web application with Oracle Identity Cloud Service.

1. In the Identity Cloud Service console, expand the **Navigation Drawer** , click **Applications**.
2. In the **Applications** page, click **Add**.



[Description of this image](#)

3. In the **Add Application** chooser dialog, click **Confidential Application**.
4. Populate the **Details** pane as follows, and then click **Next**.

- Grant the client access to Identity Cloud Service Admin APIs.
-  Add
- App Roles
- No data to display.

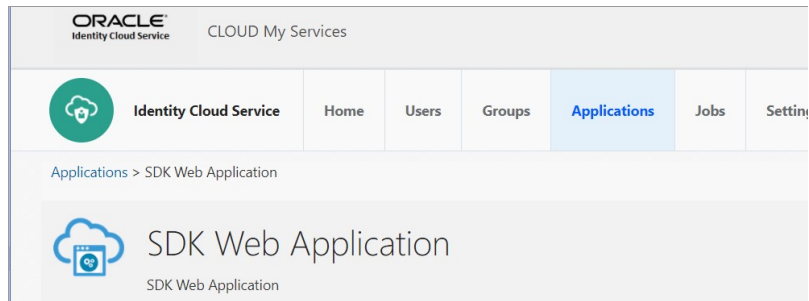
7. In the **Add App Role** dialog window, select **Authenticator Client** and **Me** in the list, and then click **Add**.
8. Click **Next** in the **Client** pane and in the following panes until you reach the last pane. Then click **Finish**.
9. In the **Application Added** dialog box, make a note of the **Client ID** and **Client Secret** values (because your web application needs these values to integrate with Oracle Identity Cloud Service), and then click **Close**.





Description of this image

10. To activate the application, click **Activate**.



Description of this image

11. In the **Activate Application?** dialog box, click **Activate Application**.

The success message **The SDK Web Application application has been activated.** appears.

12. In the Identity Cloud Service console, click the user name at the top-right of the screen, and click **Sign Out**.

3 Update the Sample Web Application

In this section, you update the sample application code to make it use Oracle Identity Cloud Service's SDK for Python programming language.

1. Populate the values of the Client ID , Client Secret , and Oracle Identity Cloud Service's base URL in the following format: **https://<domain>**

Note: It is important to update the fields above accordingly before continuing to the next sections.

2. Update the `c:\temp\python\config.json` file with the following content, and then save the file.

```
{
  "ClientId" : "clientid",
  "ClientSecret" : "clientsecret",
```

```
"BaseUrl" : "baseurl",
"AudienceServiceUrl" : "baseurl",
"scope" : "urn:opc:idm:t.user.me openid",
"TokenIssuer" : "https://identity.oraclecloud.com",
"redirectURL": "http://localhost:8000/callback",
"logoutSufix":"/oauth2/v1/userlogout",
"LogLevel":"INFO",
"ConsoleLog":"True"
}
```

4 Run the Sample Web Application

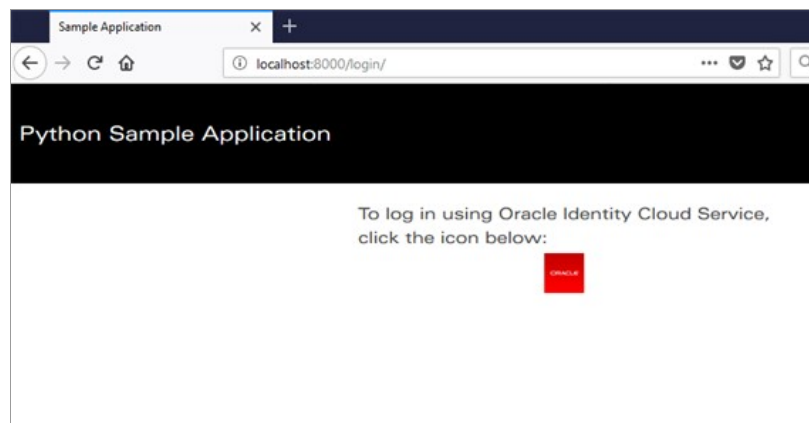
In this section of the tutorial, you prepare, run, and test the sample web application.

1. Open a command prompt, navigate to the `c:\temp\python` folder, and enter the following command lines:
2. In the command prompt, run the command line `python.exe manage.py migrate` to prepare the sample application, then run the command line `python manage.py runserver` to start the sample application.

The python server uses the 8000 port. Make sure the port isn't in use by another service.

Note: Make sure to copy the Python SDK files to the source code folder before running the previous command lines. See [Download the SDK in the Sample Web Application](#).

3. Open a browser window, access the `http://localhost:8000` URL, and click **Log in**.
4. In the **Login** page, click the **Oracle** red icon.



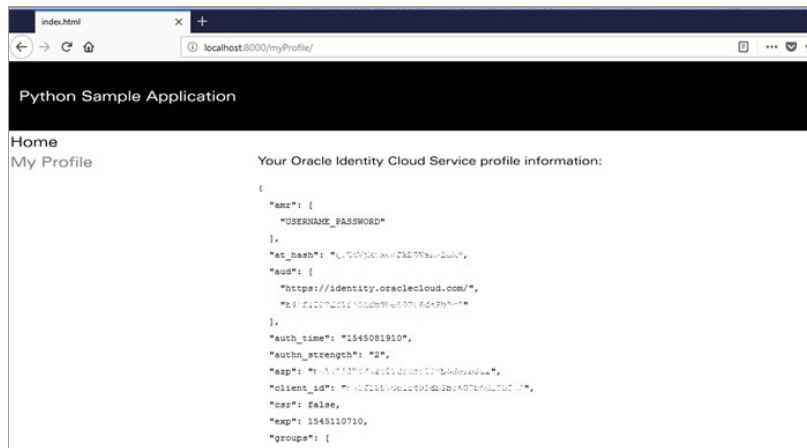


Description of this image

5. In the Oracle Identity Cloud Service **Sign In** page, sign in using your Oracle Identity Cloud Service credentials.

After you sign in to Oracle Identity Cloud Service successfully, the browser is redirected to the **/home** page. The name of the logged-in user appears at the top-right side of the page.

6. In the left menu, click **My Profile**.
7. Verify that information associated with your profile appears in the center of the page.



Description of this image

8. Click **Log Out** on the upper-right corner. The sample application finalizes the user session and redirects the browser to Oracle Identity Cloud Service's logout URL.

After Oracle Identity Cloud Service logs the user off, it redirects the user browser to the sample application index page. This behavior happens because the sample application adds two parameters `post_logout_redirect_uri` and `id_token_hint` to the Oracle Identity Cloud Service logout URL, as per below:

```
https://../oauth2/v1/userlogout?post_logout_red:
```

The `post_logout_redirect_uri` parameter value must match the **Post Logout Redirect URL** parameter value you set during [Register the Sample Web Application with Oracle Identity Cloud Service](#)



Want to Learn More?

- [Configure Federated Single Sign-On \(Federated SSO\) between Oracle Identity Cloud Service and a custom application using OAuth 2.0 and OpenID Connect.](#)
- [Administering Oracle Identity Cloud Service guide: Use Case: Adding Applications](#)

[About Oracle](#) | [Contact Us](#) | [Legal Notices](#) | [Terms of Use](#) | [Your Privacy Rights](#) |

Copyright © 2018, Oracle and/or its affiliates. All rights reserved.