**Set up sign-up and sign-in with an Oracle Cloud account using Azure Active Directory B2C**

***Before you begin***, use the selector above to choose the type of policy you’re configuring. Azure AD B2C offers two methods of defining how users interact with your applications: through predefined [user flows](https://docs.microsoft.com/en-us/azure/active-directory-b2c/user-flow-overview), or through fully configurable [custom policies](https://docs.microsoft.com/en-us/azure/active-directory-b2c/custom-policy-overview). The steps required in this article are different for each method.

## Prerequisites

* [Create a user flow](https://docs.microsoft.com/en-us/azure/active-directory-b2c/tutorial-create-user-flows) to enable users to sign up and sign into your application.
* If you haven't already done so, [register a web application](https://docs.microsoft.com/en-us/azure/active-directory-b2c/tutorial-register-applications), and [enable ID token implicit grant](https://docs.microsoft.com/en-us/azure/active-directory-b2c/tutorial-register-applications#enable-id-token-implicit-grant).

## Create an app in the Oracle IDCS developer dashboard.

To enable sign-in for users with an Oracle IDCS account in Azure Active Directory B2C (Azure AD B2C), you need to create an application in [Oracle IDCS Cloud](https://cloud.oracle.com) . For more information, see [Create Oracle Identity cloud services](https://docs.oracle.com/en-us/iaas/Content/Identity/Concepts/federation.htm). If you don't already have an Oracle account, you can sign up at <https://Cloud.oracle.com/>.

1. Sign in to the [Oracle Cloud Developer Console](https://cloud.oracle.com) with your Oracle cloud account credentials.
2. From the Dashboard, select **Login with Oracle** **Cloud Account**
3. Create an **Oracle Identity Cloud Service application** to define the OAuth resource and OAuth client relationships. This Oracle Identity Cloud Service application can be configured as an OAuth resource server for accessing data resources using OAuth. Please provide below information.

Graphical user interface, text, application, email

Description automatically generated

1. Click on Oracle Identity Cloud Service URL and login with Oracle Cloud Account.

Graphical user interface, application

Description automatically generated

1. After Successful Login, Create Application.
2. In the Oracle Identity Cloud Service console, select the **Application tab**, click Add, and then in the Add Application window, select Trusted Application.

Graphical user interface

Description automatically generated with medium confidence

1. On the Details page of the Add Trusted Application wizard, give the new application a name. If you want, set other values such as description and tags. Click Next.
2. On the Client page, select Configure this application as a client now.
3. Additional options appear on the page. Set the following values:
4. Allowed Grant Types: Client Credentials and JWT Assertion.
5. Redirect URL: Enter a Redirect URL for the user to be sent after authentication, such as the Oracle SaaS application home page.

(https://<tenantname>.b2clogin.com/<tenantName>.onmicrosoft.com/oauth2/authresp)

Graphical user interface, text, application, email

Description automatically generated

1. Client Type: Select the Confidential option.
2. On the Resources page, select Configure this application as a resource server now.
3. Additional options appear on the page.
4. Set the following values:
5. Primary Audience: <your Oracle Applications base address>.
6. Allowed Scopes: Click Add and create a scope with the value /. Check the Requires Consent box.
7. Click Next, and on the Authorization page, click Finish to save the application.
8. An “Application Added” notification is shown. **Make a copy of the Client ID and Client Secret:** you’ll need to provide them to Oracle Support later.
9. If you need them later, **the Client ID and Client Secret** also appear on the Configuration tab in the Details section for the application.

Graphical user interface, application, Teams

Description automatically generated

1. With the app created and saved, select the Configuration tab, and expand the Client Configuration section. In the Accessing APIs from Other Applications section, under Allowed Scopes, click Add. In the Add Scope dialog, add one or more Resources. You can check the box for a resource to add all its scopes or click the right arrow for a given resource to select individual scopes. You can check the box for the whole app to add all resources. Click the Add button. Click Save to save your changes.
2. As you add scopes, they are listed by application and allowed scope in the Allowed Scopes area. You can select a scope in this area and click the Remove button to remove it.
3. To activate the application, from the Oracle Identity Cloud Service console, select Applications, and select the application. Click the Activate button to the right of the application name.

Graphical user interface, application, Teams

Description automatically generated

## Create Oracle identity provider.

1. In your Azure AD B2C tenant, select new OpenID Connect Provider using **Identity Provider under Manage menu.**
2. Provide the name (say OracleIDCS) , IDCS metadata URL, client id and client secret with required information.
3. Select **Save**.

A screenshot of a computer

Description automatically generated with medium confidence

## Create Oracle identity provider.

1. In your Azure AD B2C tenant, select new OpenID Connect Provider using **Identity Provider under Manage menu.**
2. Provide the name (say **OracleIDCS**) , IDCS metadata URL, client id and client secret with required information.
3. Select **Save**.

**Add Oracle identity provider to Custom Policy**

## Create a policy key

You need to store the client secret that you previously recorded in your Azure AD B2C tenant.

1. Sign in to the [Azure portal](https://portal.azure.com/).
2. Make sure you're using the directory that contains your Azure AD B2C tenant by selecting the **Directory + subscription** filter in the top menu and choosing the directory that contains your tenant.
3. Choose **All services** in the top-left corner of the Azure portal, and then search for and select **Azure AD B2C**.
4. On the Overview page, select **Identity Experience Framework**.
5. Select **Policy Keys** and then select **Add**.
6. For **Options**, choose Manual.
7. Enter a **Name** for the policy key. For example, Oracle Secret. The prefix B2C\_1A\_ is added automatically to the name of your key.
8. In **Secret**, enter your client secret that you previously recorded. (Step 20 of IDCS)
9. For **Key usage**, select Signature.
10. Click **Create**.

## Configure Oracle as an identity provider.

To enable users to sign in using a Oracle account, you need to define the account as a claims provider. that Azure AD B2C can communicate with through an endpoint. The endpoint provides a set of claims that are used by Azure AD B2C to verify that a specific user has authenticated.

You can define an Oracle account as a claims provider by adding it to the **ClaimsProviders** element in the extension file of your policy.

1. Open the TrustFrameworkExtensions.xml.
2. Find the **ClaimsProviders** element. If it does not exist, add it under the root element.
3. Add a new **ClaimsProvider** as follows:

XMLCopy

<ClaimsProvider>

      <DisplayName>Login using Oracle Cloud IDCS</DisplayName>

  <TechnicalProfiles>

    <TechnicalProfile Id="OracleClientLogin-OpenIdConnect">

      <DisplayName>OracleClient Employee</DisplayName>

      <Description>Login with your Oracle account</Description>

      <Protocol Name="OpenIdConnect"/>

      <Metadata>

        <Item Key="METADATA">[https://idcs-xxxx.identity.oraclecloud.com/.well-known/openid-configuration</Item](https://idcs-xxxx.identity.oraclecloud.com/.well-known/openid-configuration%3c/Item)>

        <Item Key="client\_id">Oracle Client ID for App</Item>

        <Item Key="response\_types">code</Item>

        <Item Key="scope">openid profile</Item>

        <Item Key="response\_mode">form\_post</Item>

        <Item Key="HttpBinding">POST</Item>

        <Item Key="UsePolicyInRedirectUri">false</Item>

      </Metadata>

      <CryptographicKeys>

        <Key Id="client\_secret" StorageReferenceId="B2C\_1A\_OracleAppSecret"/>

      </CryptographicKeys>

      <OutputClaims>

        <OutputClaims>

        <OutputClaim ClaimTypeReferenceId="issuerUserId" PartnerClaimType="oid" DefaultValue="xxx"/>

        <OutputClaim ClaimTypeReferenceId="tenantId" PartnerClaimType="tid" DefaultValue="xxx"/>

        <OutputClaim ClaimTypeReferenceId="givenName" PartnerClaimType="given\_name" DefaultValue="xxx"/>

        <OutputClaim ClaimTypeReferenceId="surName" PartnerClaimType="family\_name" DefaultValue="xx" />

        <OutputClaim ClaimTypeReferenceId="displayName" PartnerClaimType="name" DefaultValue="xx" />

         <OutputClaim ClaimTypeReferenceId="email" PartnerClaimType="name" DefaultValue=[xxx](mailto:oraclejit@gmail.com)/>

        <OutputClaim ClaimTypeReferenceId="objectId" PartnerClaimType="sub" DefaultValue="xx"/>

        <OutputClaim ClaimTypeReferenceId="authenticationSource" DefaultValue="socialIdpAuthentication" AlwaysUseDefaultValue="true" />

        <OutputClaim ClaimTypeReferenceId="identityProvider" PartnerClaimType="iss" DefaultValue="xxx"/>

      </OutputClaims>

      <OutputClaimsTransformations>

        <OutputClaimsTransformation ReferenceId="CreateRandomUPNUserName"/>

        <OutputClaimsTransformation ReferenceId="CreateUserPrincipalName"/>

        <OutputClaimsTransformation ReferenceId="CreateAlternativeSecurityId"/>

        <OutputClaimsTransformation ReferenceId="CreateSubjectClaimFromAlternativeSecurityId"/>

      </OutputClaimsTransformations>

      <UseTechnicalProfileForSessionManagement ReferenceId="SM-SocialLogin"/>

    </TechnicalProfile>

  </TechnicalProfiles>

</ClaimsProvider>

1. Set **client\_id** to the application ID from the application registration.
2. Save the file.

## Add a user journey

At this point, the identity provider has been set up, but it's not yet available in any of the sign-in pages. If you don't have your own custom user journey, create a duplicate of an existing template user journey, otherwise continue to the next step.

1. Open the TrustFrameworkBase.xml file from the starter pack.
2. Find and copy the entire contents of the **UserJourney** element that includes Id="SignUpOrSignIn".
3. Open the TrustFrameworkExtensions.xml and find the **UserJourneys** element. If the element doesn't exist, add one.
4. Paste the entire content of the **UserJourney** element that you copied as a child of the **UserJourneys** element.
5. Rename the Id of the user journey. For example, Id="CustomSignUpSignIn".

## Add the identity provider to a user journey

Now that you have a user journey, add the new identity provider to the user journey. You first add a sign-in button, then link the button to an action. The action is the technical profile you created earlier.

1. Find the orchestration step element that includes Type="CombinedSignInAndSignUp", or Type="ClaimsProviderSelection" in the user journey. It's usually the first orchestration step. The **ClaimsProviderSelections** element contains a list of identity providers that a user can sign in with. The order of the elements controls the order of the sign-in buttons presented to the user. Add a **ClaimsProviderSelection** XML element. Set the value of **TargetClaimsExchangeId** to a friendly name.
2. In the next orchestration step, add a **ClaimsExchange** element. Set the **Id** to the value of the target claims exchange Id. Update the value of **TechnicalProfileReferenceId** to the Id of the technical profile you created earlier.

The following XML demonstrates the first two orchestration steps of a user journey with the identity provider:

XMLCopy

<OrchestrationStep Order="1" Type="CombinedSignInAndSignUp" ContentDefinitionReferenceId="api.signuporsignin">

<ClaimsProviderSelections>

...

<ClaimsProviderSelection TargetClaimsExchangeId="OracleExchange" />

</ClaimsProviderSelections>

...

</OrchestrationStep>

<OrchestrationStep Order="2" Type="ClaimsExchange">

...

<ClaimsExchanges>

<ClaimsExchange Id="OracleExchange" TechnicalProfileReferenceId="OracleClientLogin-OpenIdConnect" />

</ClaimsExchanges>

</OrchestrationStep>

## Configure the relying party policy

The relying party policy, for example [SignUpSignIn.xml](https://github.com/Azure-Samples/active-directory-b2c-custom-policy-starterpack/blob/master/SocialAndLocalAccounts/SignUpOrSignin.xml), specifies the user journey which Azure AD B2C will execute. Find the **DefaultUserJourney** element within [relying party](https://docs.microsoft.com/en-us/azure/active-directory-b2c/relyingparty). Update the **ReferenceId** to match the user journey ID, in which you added the identity provider.

In the following example, for the CustomSignUpOrSignIn user journey, the **ReferenceId** is set to CustomSignUpOrSignIn:

XMLCopy

<RelyingParty>

<DefaultUserJourney ReferenceId="CustomSignUpSignIn" />

...

</RelyingParty>

### Upload the custom policy

1. Sign in to the [Azure portal](https://portal.azure.com/).
2. Select the **Directory + Subscription** icon in the portal toolbar, and then select the directory that contains your Azure AD B2C tenant.
3. In the Azure portal, search for and select **Azure AD B2C**.
4. Under **Policies**, select **Identity Experience Framework**.
5. Select **Upload Custom Policy**, and then upload the two policy files that you changed, in the following order: the extension policy, for example TrustFrameworkExtensions.xml, then the relying party policy, such as SignUpSignIn.xml.

## Test your custom policy.

1. Select your relying party policy, for example B2C\_1A\_signup\_signin.
2. For **Application**, select a web application that you [previously registered](https://docs.microsoft.com/en-us/azure/active-directory-b2c/tutorial-register-applications). The **Reply URL** should show https://jwt.ms.
3. Select the **Run now** button.
4. From the sign-up or sign-in page, select **Oracle** to sign in with Oracle account.

If the sign-in process is successful, your browser is redirected to https://jwt.ms, which displays the contents of the token returned by Azure AD B2C.