This guide shows how to enable an existing web application for OpenID Connect (OIDC) with Identity Platform. This includes accepting OIDC tokens from identity providers (IdP), verifying their contents, and producing a lightweight JWT that you can use in your application to verify authentication and perform authorization.

To enable OIDC for your web app, you'll need the following:

- Access to an application that you want to enable for OIDC sign-on.
- A Google Cloud project for which you're a Project Owner, with billing enabled (/billing/docs/how-to/modify-project#enable_billing_for_a_project) for the project.
- Identity Platform enabled (/identity-platform/docs/quickstart-cicp) for the project.

1. Go to the **Identity Providers** page in the Cloud Console.
   Go to the Identity Providers page (https://console.cloud.google.com/customer-identity/providers)
2. Click **Add A Provider**.
3. On the **Select a provider** drop-down list, select **OpenID Connect**.
4. Next to **Enabled**, click the button to enable the provider.
5. Under **Configure OIDC connection**, enter the following details:
   a. A **Display Name**. Note that an identifier for the provider will be automatically generated. This identifier will begin with `oidc`. You will need to reference the identifier when signing in a user as described below.
   b. Enter the **ClientID** from your OIDC identity provider.

c. Enter the **Issuer URL** from your OIDC identity provider. The OIDC issuer is used to determine the discovery document as described in the discovery specification for OpenID Connect (https://openid.net/specs/openid-connect-discovery-1_0.html#ProviderConfig). This is used to construct the OAuth endpoints and retrieve the public keys that are used for verification, like verifying signatures for issued ID tokens.

6. Make sure that the provisioned Identity Platform OAuth callback URL for your project is configured at your OIDC provider as the redirect URL. This is usually of the form of: `https://[AUTH_DOMAIN]/__/auth/handler`

Note that if your OIDC identity provider doesn't comply with the OpenID Connect specification for discovery, it won't work with the Identity Platform v2. We recommend discussing adherence to the OIDC specification with your identity provider. The Identity Platform client SDK provides an API that handles the entire OIDC handshake and returns an ID token issued by Identity Platform.

If you're signing in users to `https://www.example.com/login`, you will need to add `www.example.com` to the list of authorized domains in Identity Platform. To do this:

1. Go to the **Identity Platform Settings** page in the Cloud Console.
   Go to the Identity Platform Settings page (https://console.cloud.google.com/customer-identity/settings)
2. Under **Authorized** Domains, click **Add Domain**.
3. Add any domains that should be allowed to authenticate your end users.

Identity Platform is now configured to trust assertions from your OIDC identity provider.

Next, you'll configure your web application to support OIDC SSO flows via the client SDK. For more information about the Web SDK, see the docs (https://firebase.google.com/docs/auth/web/start).

To add the SDK to your application, follow the steps below:

1. Go to the **Identity Platform Users** page in the Cloud Console.
   Go to the Identity Platform Users page (https://console.cloud.google.com/customer-identity/users)

2. On the top right, click application **setup details**.
3. Copy the application setup details into your web application. For example:

You can use the following methods to sign in with an OIDC provider to Identity Platform:

- Sign in using the `id_token` implicit OAuth flow. This flow will take care of the OAuth handshake.
- Sign in using the OIDC provider's ID token. This method assumes that the OIDC ID token is already available.

For more information, see `firebase.auth.OAuthProvider` (https://firebase.google.com/docs/reference/js/firebase.auth.OAuthProvider).

To sign in with an OIDC provider, you need to instantiate an `OAuthProvider` instance using the provider ID that is used to configure the provider in the Cloud Console:

You will also need to add your website domain to the list of authorized domains in the Cloud Console to securely sign in users with an OIDC provider that uses the `id_token` OAuth flow. For example, if you are signing in users to `https://www.example.com/login`, you will need to add `www.example.com` to the list of authorized domains.

When you use the full OAuth flow, you can configure sign in to use a popup or a redirect flow:

- To sign in with a popup-window, call `signInWithPopup`:

- To sign in by redirecting to the sign-in page, call `signInWithRedirect`:

After the user completes sign-in and returns to the your application, you can get the sign-in result by calling `getRedirectResult`:

At the end of the flow, you can get the returned OIDC ID token from the `result.credential.idToken` field of the returned OAuthCredential.

To sign in with an OIDC ID token directly, you'll first initialize an OAuthCredential:

Next, sign in directly with this credential using `signInWithCredential`:

You can also link an OIDC provider to an existing user using `linkWithPopup`, `linkWithRedirect`, or `linkWithCredential`. For example, you can link multiple providers to the same user, so they can sign in with either method:

Sensitive operations like updating email or adding or modifying a password will require recent sign-in using `reauthenticateWithPopup`, `reauthenticateWithRedirect`, or `reauthenticateWithCredential` APIs:

When you create a project, Identity Platform will provision a unique subdomain for your project powered by Firebase Hosting, like `https://my-app-12345.firebaseapp.com`. This will also be used as the redirect mechanism for all OAuth, OIDC, and SAML based sign-in operations. Your project name at the FirebaseApp domain will be automatically whitelisted for all supported OAuth/OIDC/SAML providers. This means that users will see that URL while signing in to Google, before redirecting back to the application.

To customize the URL to show a domain that you own, follow the steps below:

1. Use the Firebase Console to connect a custom domain (https://firebase.google.com/docs/hosting/custom-domain) to your project.
2. Use the Cloud Console to add your custom domain `auth.custom.domain.com` to the list of authorized domains.
3. In your identity provider's configuration, update the callback URL from `https://my-app-12345.firebaseapp.com/__/auth/handler` to `https://auth.custom.domain.com/__/auth/handler`.
4. When you initialize your Auth instance, update the authDomain to the custom domain:

By completing this guide, your web app is now configured for using OIDC with Identity Platform, including sign-in flow and a custom redirect domain.