**LAB: Create Google Authenthentication on Website**

**Get your Google API client ID**

To enable Sign In With Google on your website, you first need to set up your Google API client ID. To do so, complete the following steps:

1. Open the **Credentials** page of the [Google APIs console](https://console.developers.google.com/apis).
2. Create or select a Google APIs project. If you already have a project for the Sign In With Google button or Google One Tap, use the existing project and the web client ID. When creating production applications, [multiple projects](https://developers.google.com/identity/protocols/oauth2/policies#separate-projects) may be necessary, repeat the remaining steps of this section for each project you manage.
3. Click **Create credentials > OAuth client ID** and for **Application type** select **Web application** to create a new client ID. To use an existing client ID select one of type **Web application**.
4. Add the URI of your website to **Authorized JavaScript origins**. The URI includes the scheme and fully qualified hostname only. For example, https://www.example.com.

**Key Point:** For local tests or development add both **http://localhost** and **http://localhost:<port\_number>Key Point:** Google One Tap can only be displayed in HTTPS domains.

1. Optionally, credentials may be returned using a redirect to an endpoint you host rather than through a JavaScript callback. If this is the case, add your redirect URIs to **Authorized redirect URIs**. Redirect URIs include the scheme, fully qualified hostname, and path and must comply with [Redirect URI validation rules](https://developers.google.com/identity/protocols/oauth2/web-server#uri-validation). For example, https://www.example.com/auth-receiver.

**Key Point:** When testing using http and localhost set the [Referrer-Policy](https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/Referrer-Policy) header in your web app to **Referrer-Policy: no-referrer-when-downgrade**.

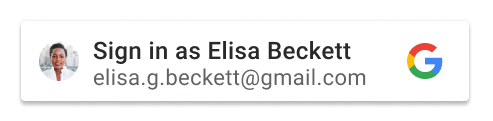
**Configure your OAuth Consent Screen**

Both Sign In With Google and One Tap authentication include a consent screen which tells users the application requesting access to their data, what kind of data they are asked for and the terms that apply.

1. Open the [OAuth consent screen](https://console.developers.google.com/apis/credentials/consent) page of the APIs & Services section of the Google Developer Console.
2. If prompted, select the project you just created.
3. On the "OAuth consent screen" page, fill out the form and click the "Save" button.
   1. **Application name:** The name of the application asking for consent. The name should accurately reflect your application and be consistent with the application name users see elsewhere.
   2. **Application logo:** This image is shown on the consent screen to help users to recognize your app. The logo is shown on Sign In With Google consent screen and on [account settings](https://myaccount.google.com/permissions), but is not shown on One Tap dialog.
   3. **Support email:** Shown on the consent screen for user support and to G Suite administrators evaluating access to your application for their users. This email address is shown to users on the Sign In With Google consent screen when the user clicks the application name.
   4. **Scopes for Google APIs:** Scopes allow your application to access your user's private data. For the authentication, default scope (email, profile, openid) is sufficient, you don't need to add any sensitive scopes. It is generally a best practice to request scopes incrementally, at the time access is required, rather than up front. [Learn more](https://developers.google.com/identity/protocols/oauth2/scopes)
   5. **Authorized domains:** To protect you and your users, Google only allows applications that authenticate using OAuth to use Authorized Domains. Your applications' links must be hosted on Authorized Domains. [Learn more](https://support.google.com/cloud/answer/6158849#authorized-domains).
   6. **Application Homepage link:** Shown on Sign In With Google consent screen and One-Tap GDPR compliant disclaimer information under the "Continue as" button. Must be hosted on an Authorized Domain.
   7. **Application Privacy Policy link:** Shown on Sign In With Google consent screen and One-Tap GDPR compliant disclaimer information under the "Continue as" button. Must be hosted on an Authorized Domain.
   8. **Application Terms of Service link (Optional):** Shown on Sign In With Google consent screen and One-Tap GDPR compliant disclaimer information under the "Continue as" button. Must be hosted on an Authorized Domain.
4. Check "Verification Status", if your application needs verification then click the "Submit For Verification" button to submit your application for verification. Refer to [OAuth verification requirements](https://support.google.com/cloud/answer/9110914) for details.

**Display the Sign In With Google button**

Add a Sign In With Google button to your site to enable users to sign-up or sign-in to your web app. Use either HTML or JavaScript to render the button and attributes to customize the button shape, size, text, and theme.



After a user selects a Google Account and provides their consent, Google shares the user profile using a JSON Web Token (JWT). For an overview of the steps involved during sign-in and user experience see [How it works](https://developers.google.com/identity/gsi/web/guides/overview#how_it_works). [Understand the personalized button](https://developers.google.com/identity/gsi/web/guides/personalized-button) reviews the different conditions and states affecting how the button is displayed to users.

**Key Point:** Our [Code Generator](https://developers.google.com/identity/gsi/web/tools/configurator) helps you to interactively design your Sign In With Google button and generates the code needed to add it to your web page.

## Prerequisites

Complete the following before adding the button to your login page:

* Follow the steps described in [Setup](https://developers.google.com/identity/gsi/web/guides/get-google-api-clientid) to configure your OAuth Consent Screen and to obtain a Client ID.
* [Load the client library](https://developers.google.com/identity/gsi/web/guides/client-library).

## Button rendering

To display the Sign In With Google button, you may choose either HTML or JavaScript to render the button on your login page:

Render the sign-in button using HTML, returning the JWT to your platform's login endpoint.

<html>  
  <body>  
    <script src="https://accounts.google.com/gsi/client" async></script>  
    <div id="g\_id\_onload"  
        data-client\_id="YOUR\_GOOGLE\_CLIENT\_ID"  
        data-login\_uri="https://your.domain/your\_login\_endpoint"  
        data-auto\_prompt="false">  
    </div>  
    <div class="g\_id\_signin"  
        data-type="standard"  
        data-size="large"  
        data-theme="outline"  
        data-text="sign\_in\_with"  
        data-shape="rectangular"  
        data-logo\_alignment="left">  
    </div>  
  <body>  
</html>

An element with a g\_id\_signin class renders as a Sign In With Google button. The button is customized by the parameters provided in the data attributes.

To display a Sign In With Google button and Google One Tap on the same page, remove data-auto\_prompt="false". This is recommended to reduce friction and improve sign-in rates.