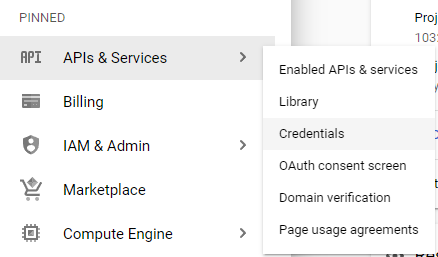
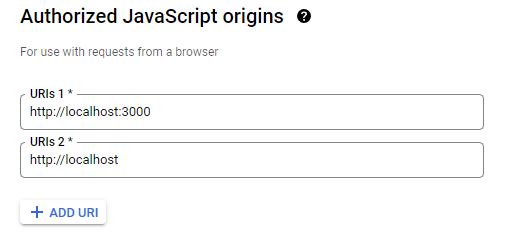
**Google oauth**

**Getting google clientId:**

To setup our authorization we first need to get a clientId from google. To get this clientId follow the steps below:

1. Visit [console.cloud.google.com](https://www.google.com/url?sa=t&rct=j&q=&esrc=s&source=web&cd=&ved=2ahUKEwirjNSVipn6AhUWHOwKHZeXB9gQFnoECBQQAQ&url=https%3A%2F%2Fconsole.cloud.google.com%2F%3Fhl%3Dnl&usg=AOvVaw3Idd3F2VEXRsUcGJxJoOdW)
2. After this create or login with your google account
3. Navigate to the dashboard and click Create Project
4. Give your project an appropriate name and then click create
5. After this Hover over APIs and Services and click on credentials: 
6. Then click Create credentials at the top and select OAuth client ID
7. You will then need to configure your consent screen, first of select usertype external
8. Then fill in the app information: App name, email. For now we will skip the app domain setup. For developer contact information fill in your own email and click save and continue.
9. For now we will skip the scopes, because we will only be getting the basic user info required for login
10. For Test user add your own email, and all other emails of people who will be testing your website
11. After saving this go back to the credential screen again and click oAuth client ID again
12. For application type select the type that best suits your product in my case this will be a web application
13. Then if you are using react or any other kind of javascript framework you need to add a javascript origin. For react if you run npm start it will run on localhost:3000 by default So let’s add <http://localhost:3000> and <http://localhost> to the javascript origins
14. After this click create, and if you did everything correctly you just got your ClientId and secret. In this case we will only be using the client ID as we only need simple user data

**Adding google oauth to react:**

We will be adding google oauth to react by using a npm package. In your react project run

“npm install @react-oauth/google@latest” this will add all things you need to your application.  
  
Alright, after installing this package create a new react component that looks similar to this:

import React from "react";

import { GoogleLogin, GoogleOAuthProvider } from "@react-oauth/google";

const SignUpPage = () => {

  return (

  );

};

export default SignUpPage;

Make sure you also use the GoogleLogin and GoogleOAuthProvider at the top of you file.

Then in the return add a GoogleOAuthProvider and add your clientId.

<GoogleOAuthProvider clientId="CLIENT\_ID\_HERE">

    </GoogleOAuthProvider>

After this we will be adding a GoogleLogin button, but let’s first go over some of the properties for this button:

* buttonText, here you can add text that will show up on the button
* onSuccess, Here you will be adding a method that handles the response when login is successful
* onFailure, Here you will be adding a method that handles the response when login fails
* isSignedIn, When this is true it will remember the user that signed in
* useOneTap, This will make login easier for the user. The user can login with one tap and don’t have to fill in their password and email
* auto\_select, This makes google automatically try to select the user that want to login.

So after knowing what the properties do we will be adding this <GoogleLogin/>, we will add this google login inside the GoogleOAuthProvider like this:

 <GoogleOAuthProvider clientId="">

      <GoogleLogin

        buttonText="Sign In with Google"

        onSuccess={(response) => {

          console.log(response);

        }}

        onFailure={(response) => {

          console.log(response);

        }}

        isSignedIn={true}

        useOneTap

        auto\_select

      />

    </GoogleOAuthProvider>

Alright, now add this component to a different component and after pressing the login button you should be able to see some credential and userId being logged in the console.  
This credential is encoded as JWT format. You can decode it with this method:

  parseJwt(token) {

    var base64Url = token.split(".")[1];

    var base64 = base64Url.replace(/-/g, "+").replace(/\_/g, "/");

    var jsonPayload = decodeURIComponent(

      window

        .atob(base64)

        .split("")

        .map(function (c) {

          return "%" + ("00" + c.charCodeAt(0).toString(16)).slice(-2);

        })

        .join("")

    );

    return JSON.parse(jsonPayload);

  }