Adding Google Authentication to Laravel 8 Website

Kenpachi Zaraki

Jun 8, 2021

With social authentication, users do not have to fill out forms in order to register on your website; they also do not need to remember another password in other to access their accounts on your website. This article will provide a step-by-step process to add Google authentication to your website

Assumptions:

This tutorial assumes that you have PHP and Composer installed on your computer. It also uses XAMPP's Apache and MySQL installations. It is therefore assumed that the reader is familiar with XAMPP. Visit the Composer and XAMPP websites to download them.

1. Create a new Laravel project

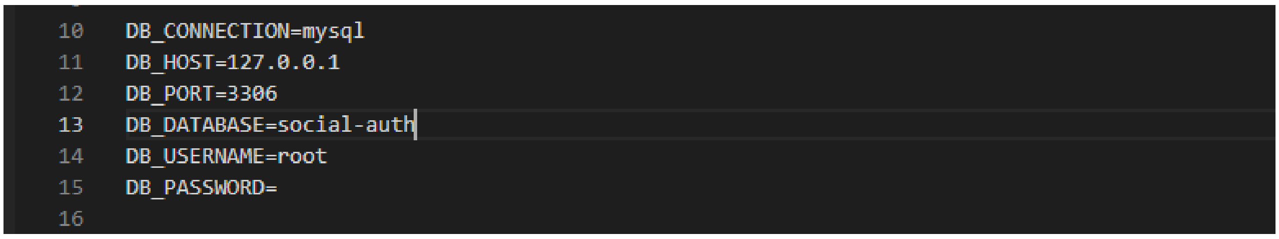
Run the following on your command line to create a new Laravel project called social-auth

```
composer create-project laravel/laravel social-auth
```

2. Configure your database setup

You can do this in two steps:

1. Create a new database called “” in phpMyAdmin
2. Edit your file with the appropriate details as shown below



```
DB_CONNECTION=mysql
DB_HOST=127.0.0.1
DB_PORT=3306
DB_DATABASE=social-auth
DB_USERNAME=root
DB_PASSWORD=
```

3. Setup your basic authentication system with Laravel Breeze

Laravel 8 provides three free authentication starter packages: Laravel Breeze, Laravel Jetstream, and Laravel Fortify. For this tutorial, we will be using Laravel Breeze. Feel free to use whichever you want. It should not make a difference for what we need to do here.

To setup Laravel Breeze, run the following lines **in the order they appear** on your command line:

```
composer require laravel/breeze --devphp artisan breeze:installnpm installnpm run devphp artisan migrate
```

4. Install Laravel Socialite and add providers and aliases in your config file

Run the following line on your command line to install Laravel Socialite

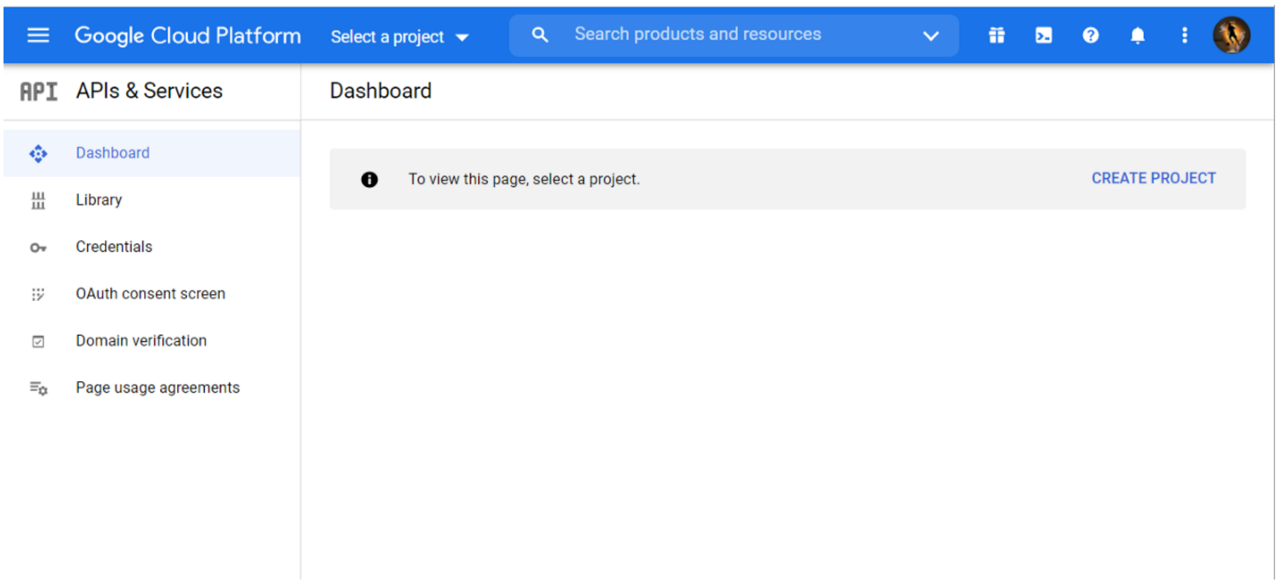
```
composer require laravel/socialite
```

In your add Socialite to your and arrays thus:

```
........'providers' => [    ....    .... Laravel\Socialite\SocialiteServiceProvider::class,],'aliases' => [    ....    .... 'Socialite' => Laravel\Socialite\Facades\Socialite::class,],........
```

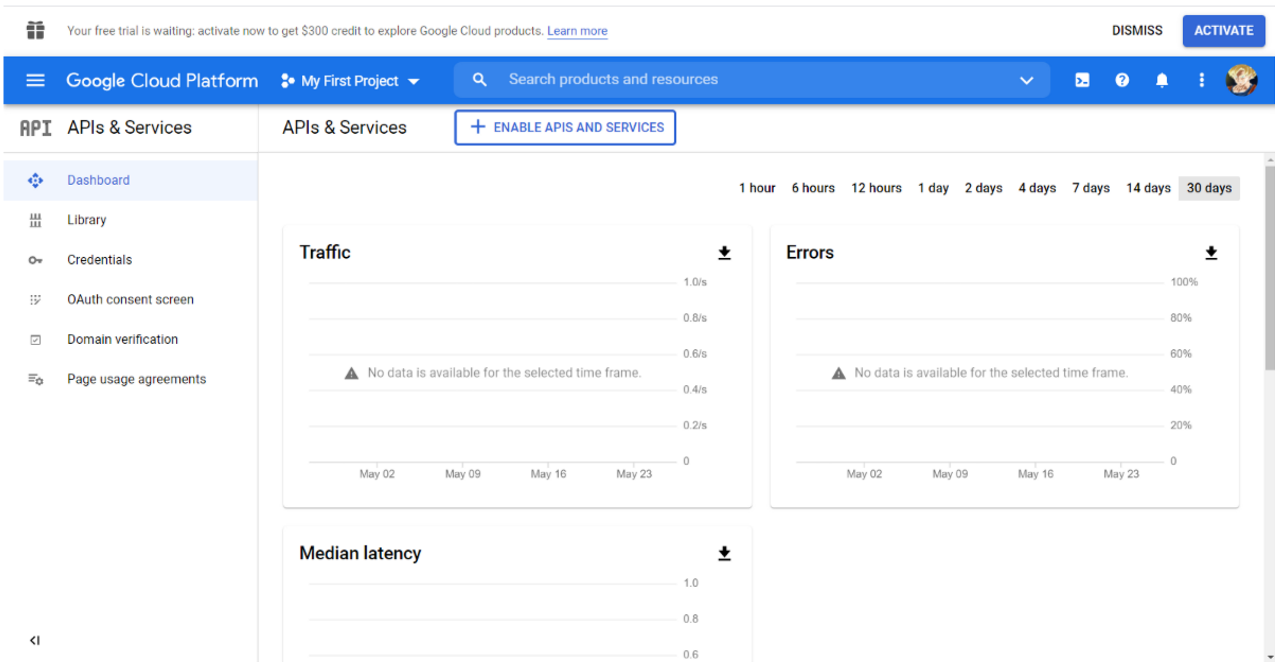
5. Create a new Google app

Follow this link to create a new Google app. It will take you to a page that looks like the one shown below if you are doing this for the first time :



To view this page, select a project.

If you have done something like this before, the page will look something like this. The rest of the tutorial is based on the assumption that you are doing this for the first time.



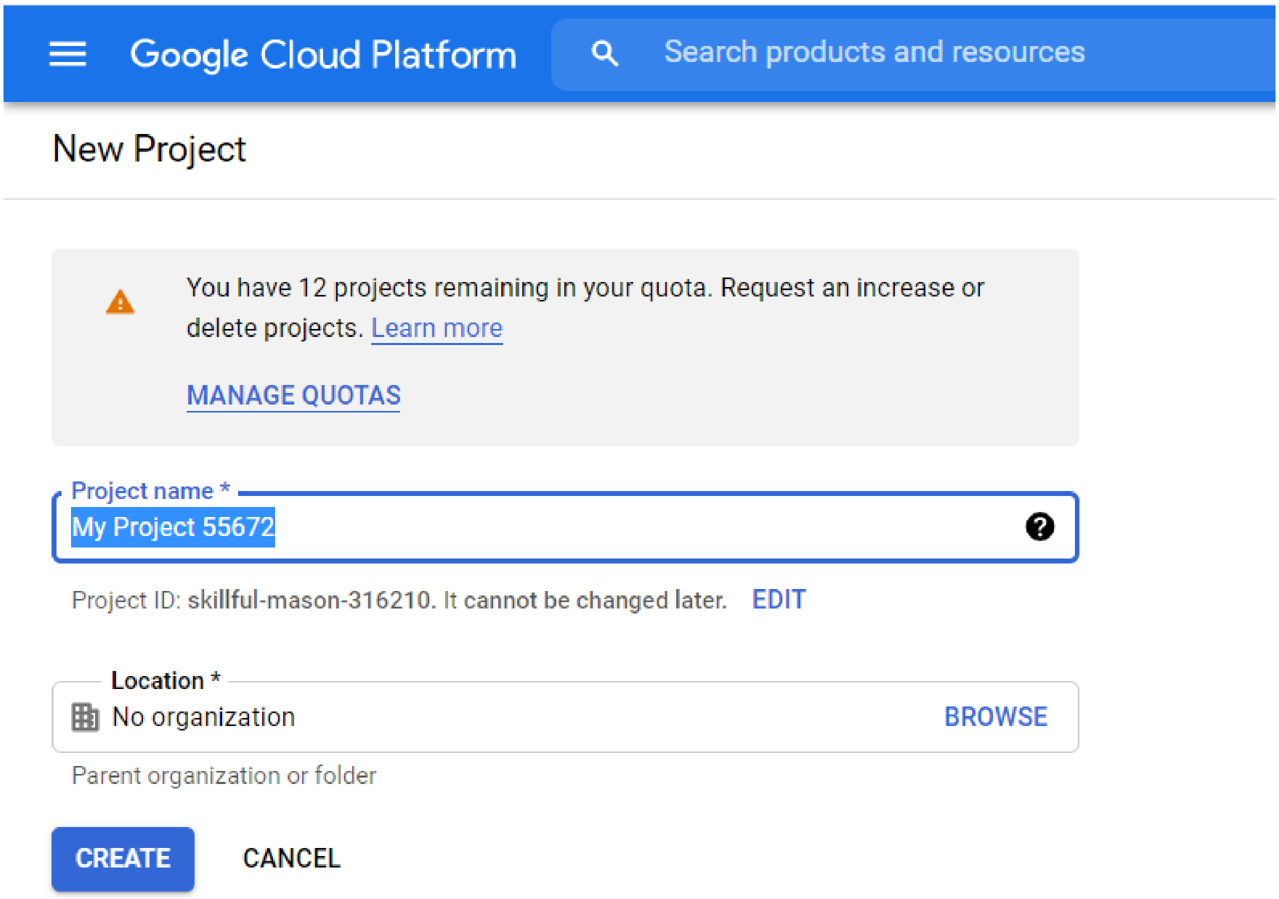
Your free trial is waiting: activate now to get $300 credit to explore Google Cloud products. Learn more

No data is available for the selected time frame.

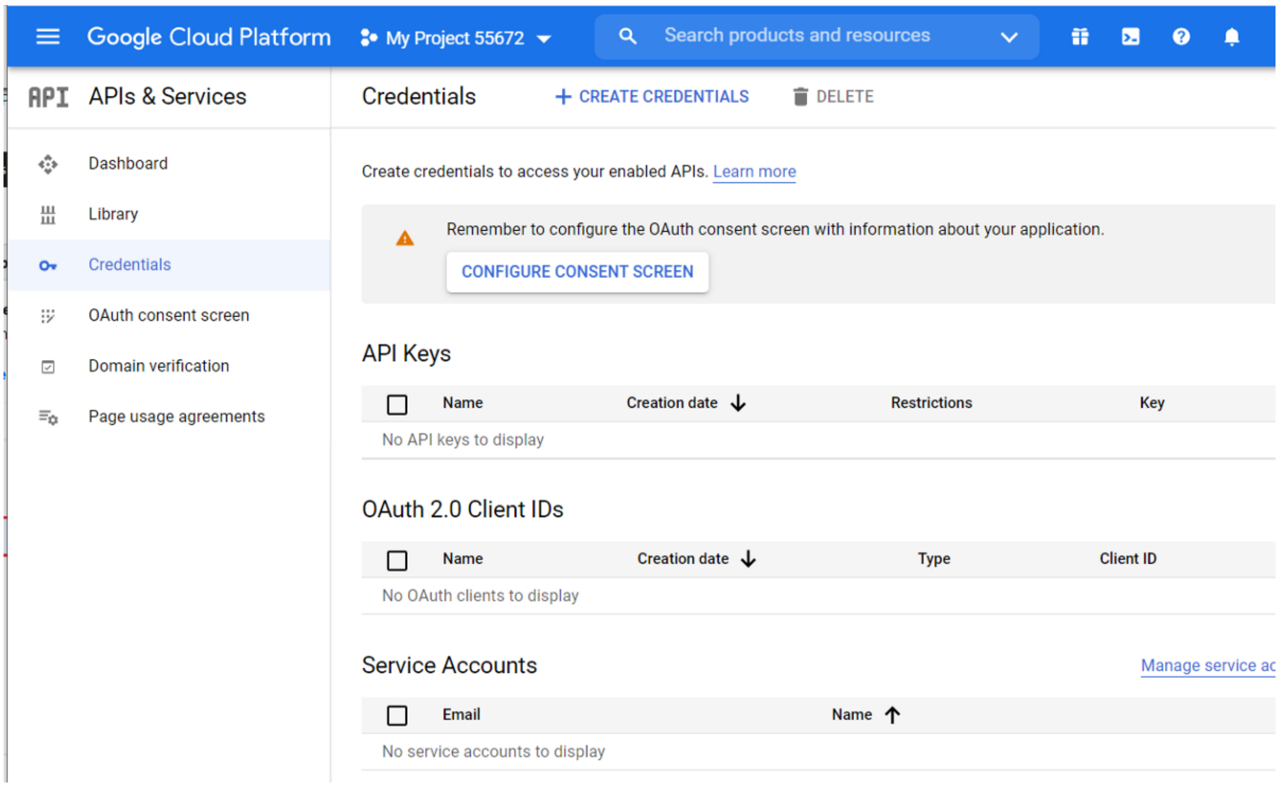
No data is available for the selected time frame.

When on the page do the following:

1. Click the “CREATE PROJECT” link. Provide required details for the resulting page.


Google Cloud Platform
Search products and resources
New Project
You have 12 projects remaining in your quota. Request an increase or delete projects. Learn more
MANAGE QUOTAS
Project name *
My Project 55672
Project ID: skillful-mason-316210. It cannot be changed later. EDIT
Location *
No organization BROWSE
Parent organization or folder
CREATE CANCEL

2. Click the Credentials option on the sidebar


Google Cloud Platform My Project 55672 Search products and resources
API APIs & Services
Credentials + CREATE CREDENTIALS DELETE
Dashboard
Library
Credentials
OAuth consent screen
Domain verification
Page usage agreements
Create credentials to access your enabled APIs. Learn more
Remember to configure the OAuth consent screen with information about your application.
CONFIGURE CONSENT SCREEN

API Keys

Name	Creation date ↓	Restrictions	Key
No API keys to display			

OAuth 2.0 Client IDs

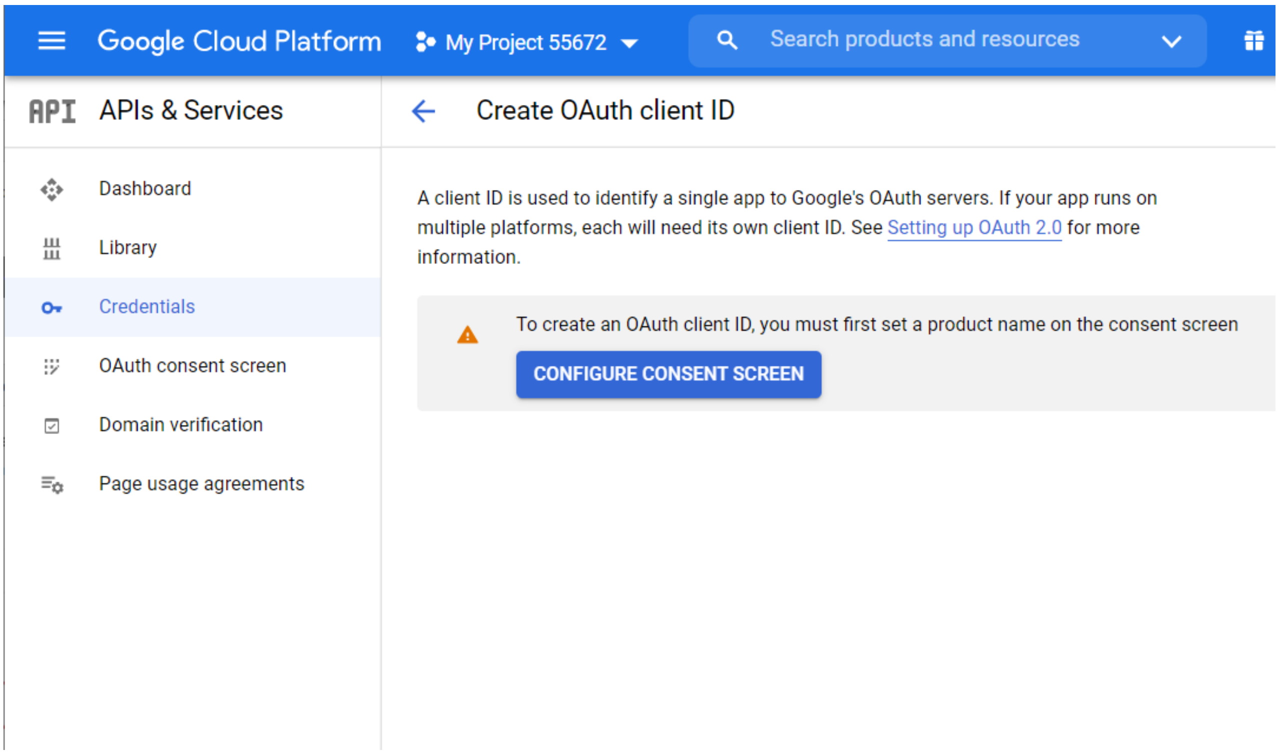
Name	Creation date ↓	Type	Client ID
No OAuth clients to display			

Service Accounts
Manage service ac

Email	Name ↑
No service accounts to display	

3. Click the “**CREATE CREDENTIALS**” item on the navbar and select the “**OAuth Client ID**” option from the resulting dropdown widget.

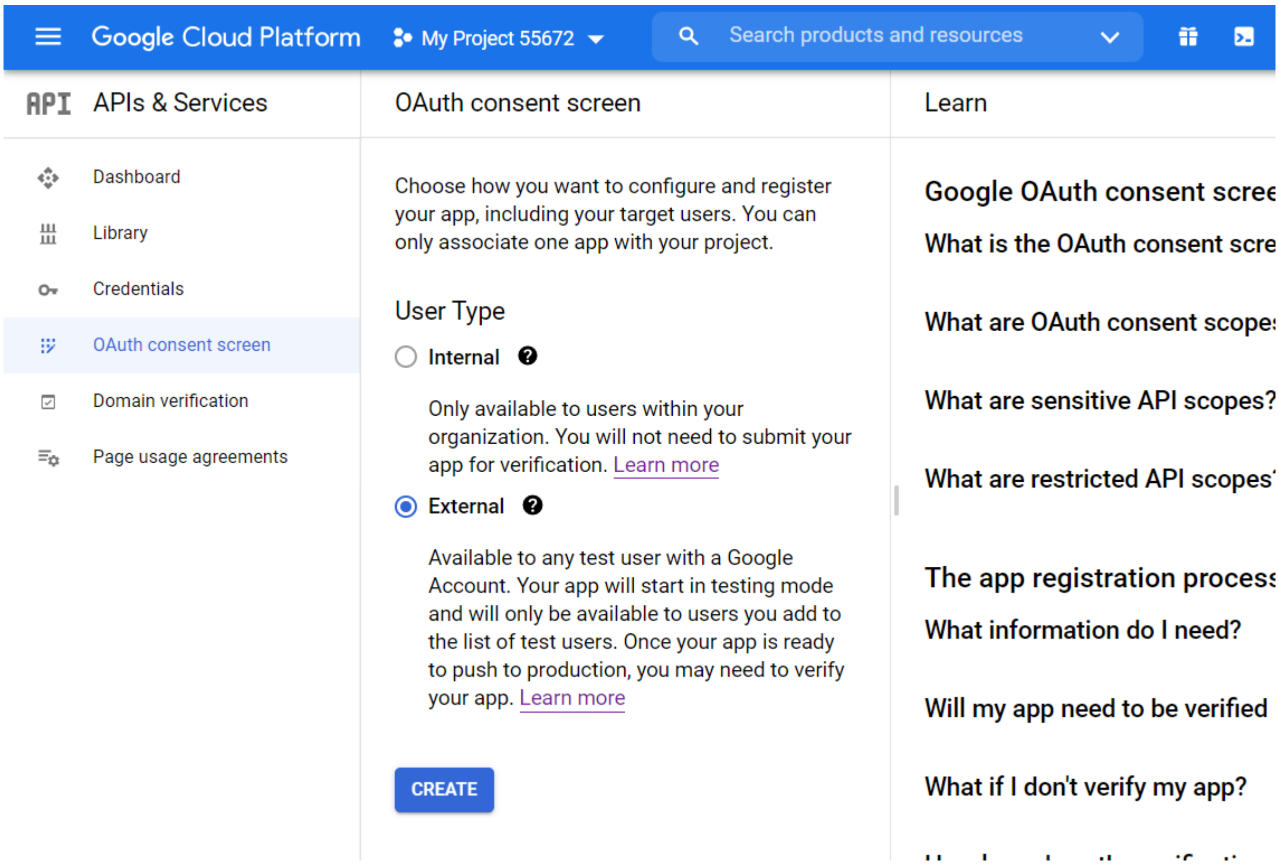
4. Click the “**CONFIGURE CONSENT SCREEN**” button.



A client ID is used to identify a single app to Google's OAuth servers. If your app runs on multiple platforms, each will need its own client ID. See Setting up OAuth 2.0 for more information.

To create an OAuth client ID, you must first set a product name on the consent screen

5. For this case, I’ll pick the “**External**” option. Feel free to pick whichever works for you.



Choose how you want to configure and register your app, including your target users. You can only associate one app with your project.

User Type

Internal

Only available to users within your organization. You will not need to submit your app for verification. Learn more

[✓ External]

Available to any test user with a Google Account. Your app will start in testing mode and will only be available to users you add to the list of test users. Once your app is ready to push to production, you may need to verify your app. Learn more

Google OAuth consent scree

What is the OAuth consent scre

What are OAuth consent scope

What are sensitive API scopes?

What are restricted API scopes

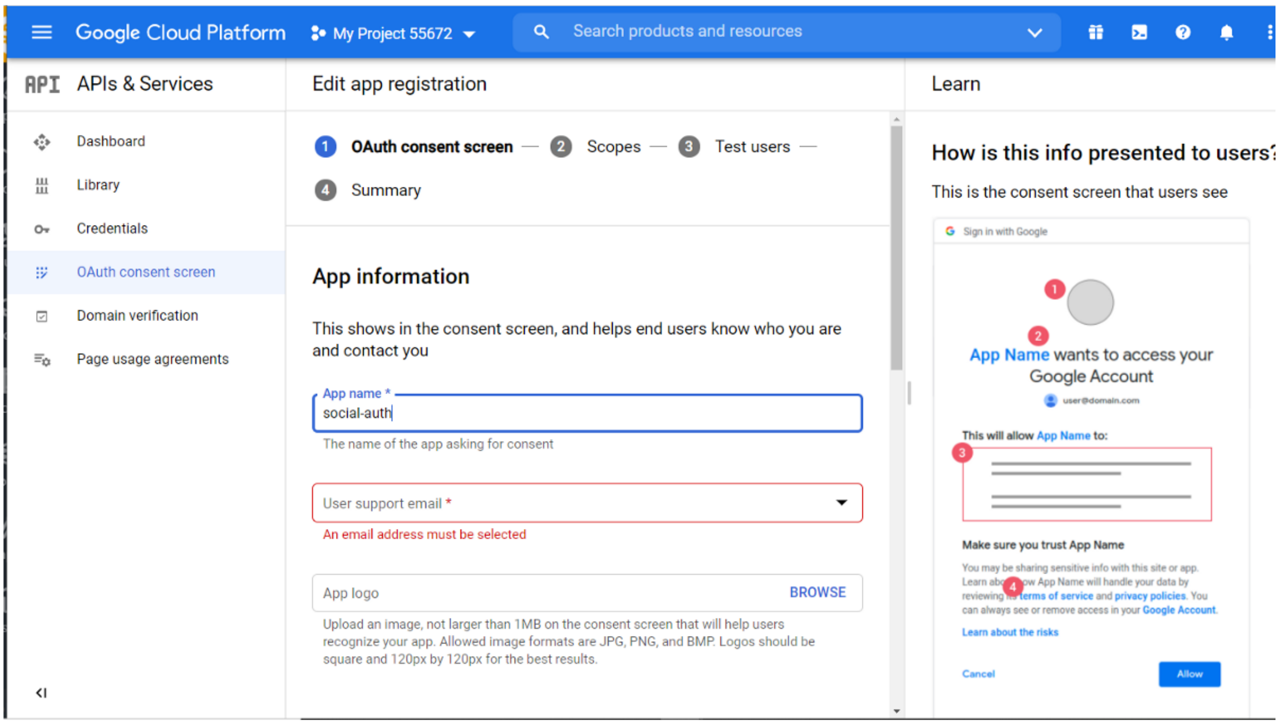
The app registration process

What information do I need?

Will my app need to be verified

What if I don't verify my app?

6. Fill the next form with the appropriate data and click the "SAVE AND CONTINUE" button



App information

This shows in the consent screen, and helps end users know who you are and contact you

App name * social-auth

The name of the app asking for consent

User support email *

An email address must be selected

App logo BROWSE

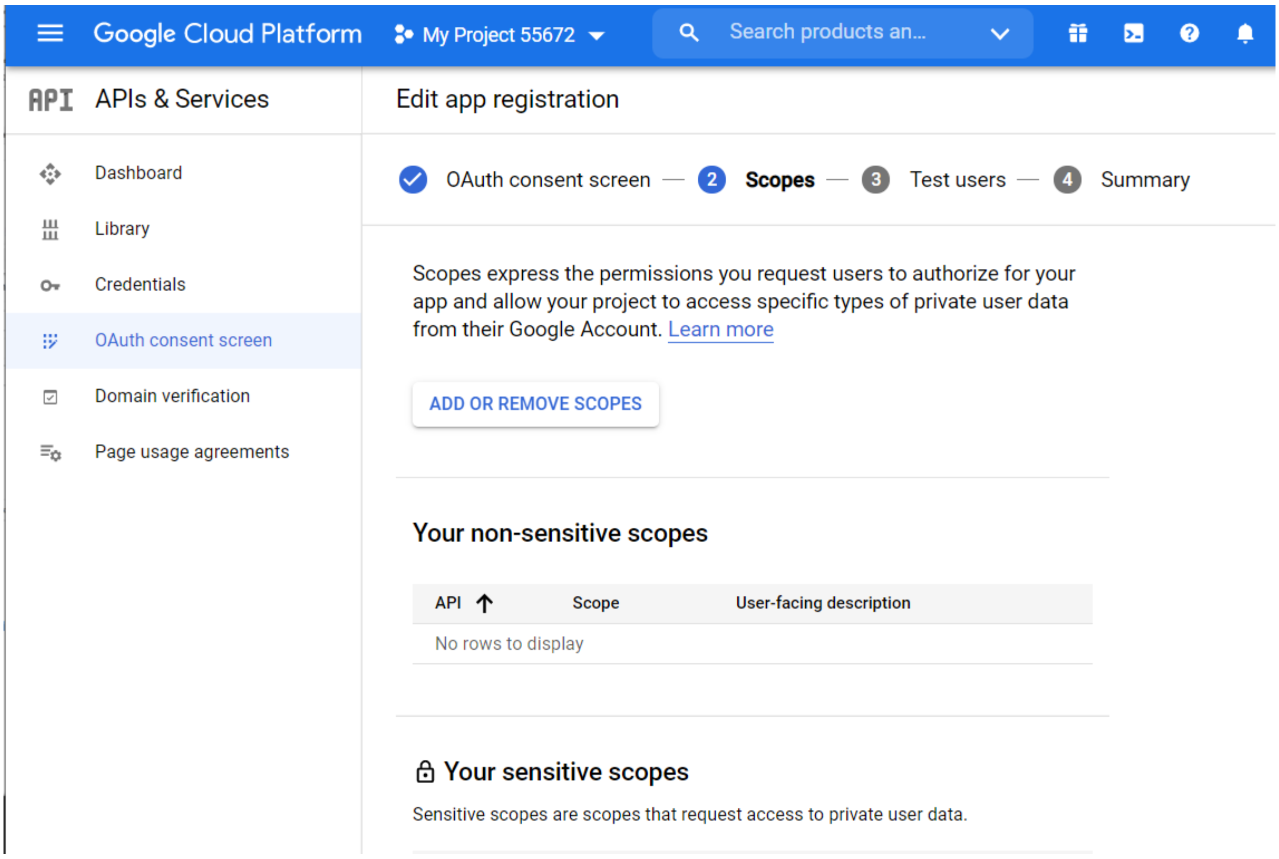
Upload an image, not larger than 1MB on the consent screen that will help users recognize your app. Allowed image formats are JPG, PNG, and BMP. Logos should be square and 120px by 120px for the best results.

How is this info presented to users?

This is the consent screen that users see

You may be sharing sensitive info with this site or app. Learn about how App Name will handle your data by reviewing its terms of service and privacy policies. You can always see or remove access in your Google Account.

7. Click the "**ADD OR REMOVE SCOPES**" button to add permissions users should authorize in the login process.



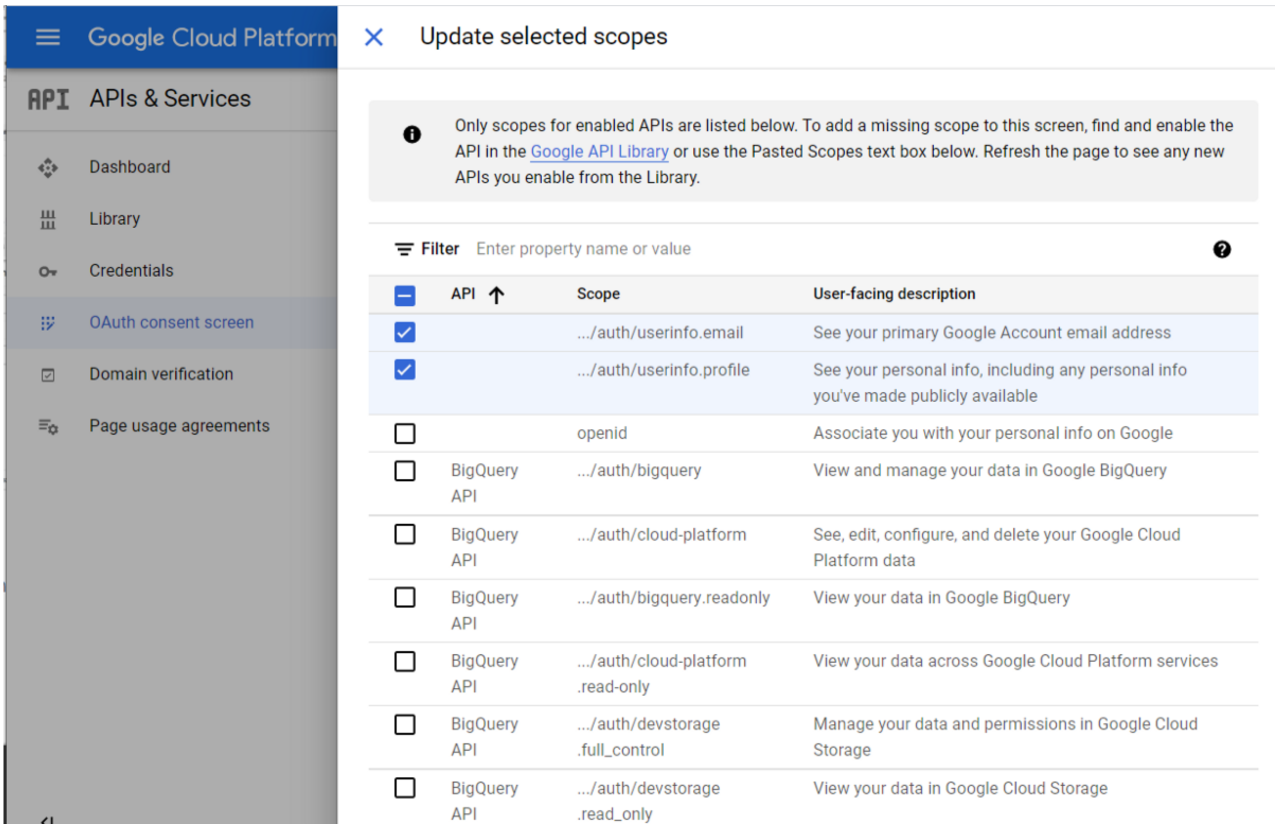
Scopes express the permissions you request users to authorize for your app and allow your project to access specific types of private user data from their Google Account. Learn more

Your non-sensitive scopes

API ↑	Scope	User-facing description
No rows to display		

Your sensitive scopes

Sensitive scopes are scopes that request access to private user data.



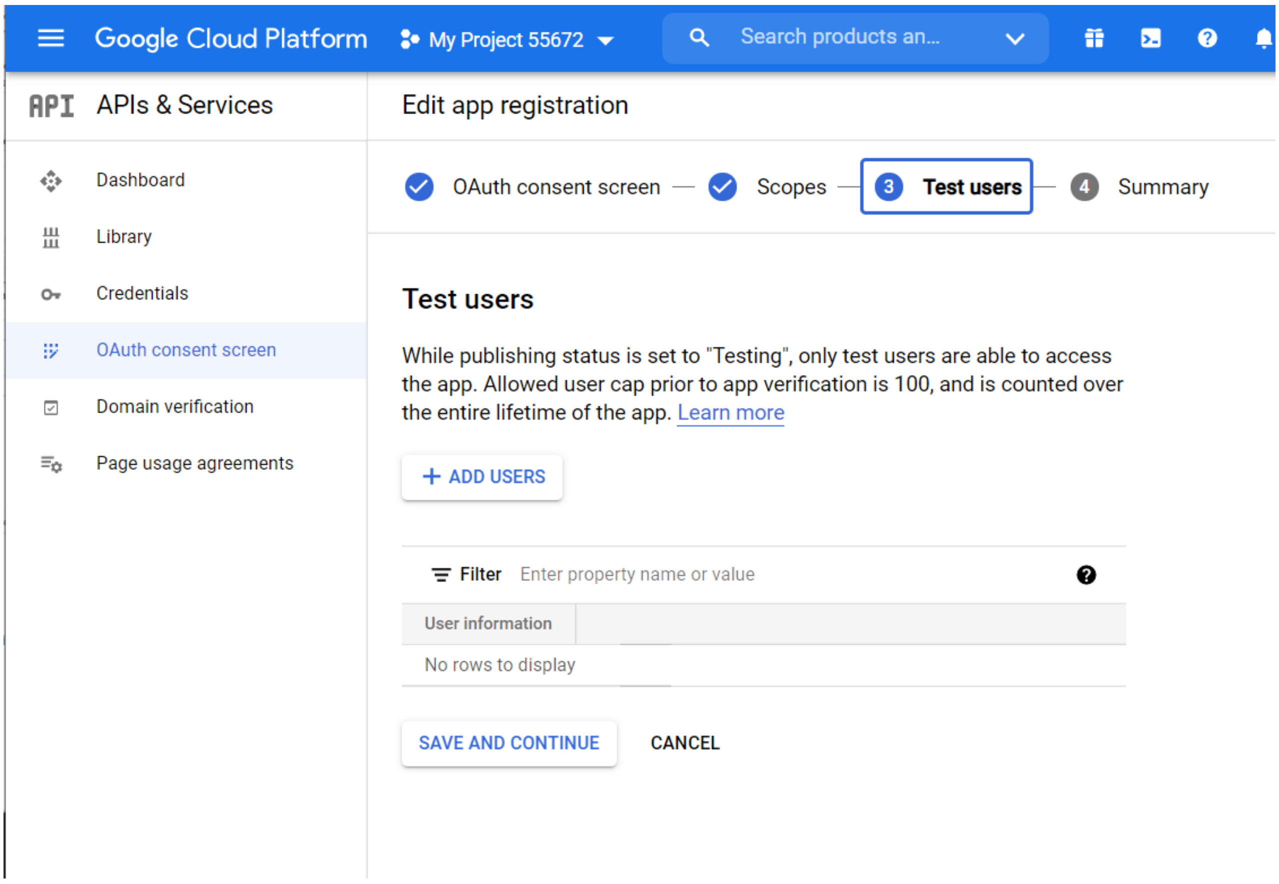
Only scopes for enabled APIs are listed below. To add a missing scope to this screen, find and enable the API in the Google API Library or use the Pasted Scopes text box below. Refresh the page to see any new APIs you enable from the Library.

Filter Enter property name or value

	API	Scope	User-facing description
☑		.../auth/userinfo.email	See your primary Google Account email address
☑		.../auth/userinfo.profile	See your personal info, including any personal info you've made publicly available
☐		openid	Associate you with your personal info on Google
☐	BigQuery API	.../auth/bigquery	View and manage your data in Google BigQuery
☐	BigQuery API	.../auth/cloud-platform	See, edit, configure, and delete your Google Cloud Platform data
☐	BigQuery API	.../auth/bigquery.readonly	View your data in Google BigQuery
☐	BigQuery API	.../auth/cloud-platform .read-only	View your data across Google Cloud Platform services
☐	BigQuery API	.../auth/devstorage .full_control	Manage your data and permissions in Google Cloud Storage
☐	BigQuery API	.../auth/devstorage .read_only	View your data in Google Cloud Storage

After picking your scopes, scroll down to the bottom of the page and click the “**UPDATE**” button.

8. Add test users on the next page using their email



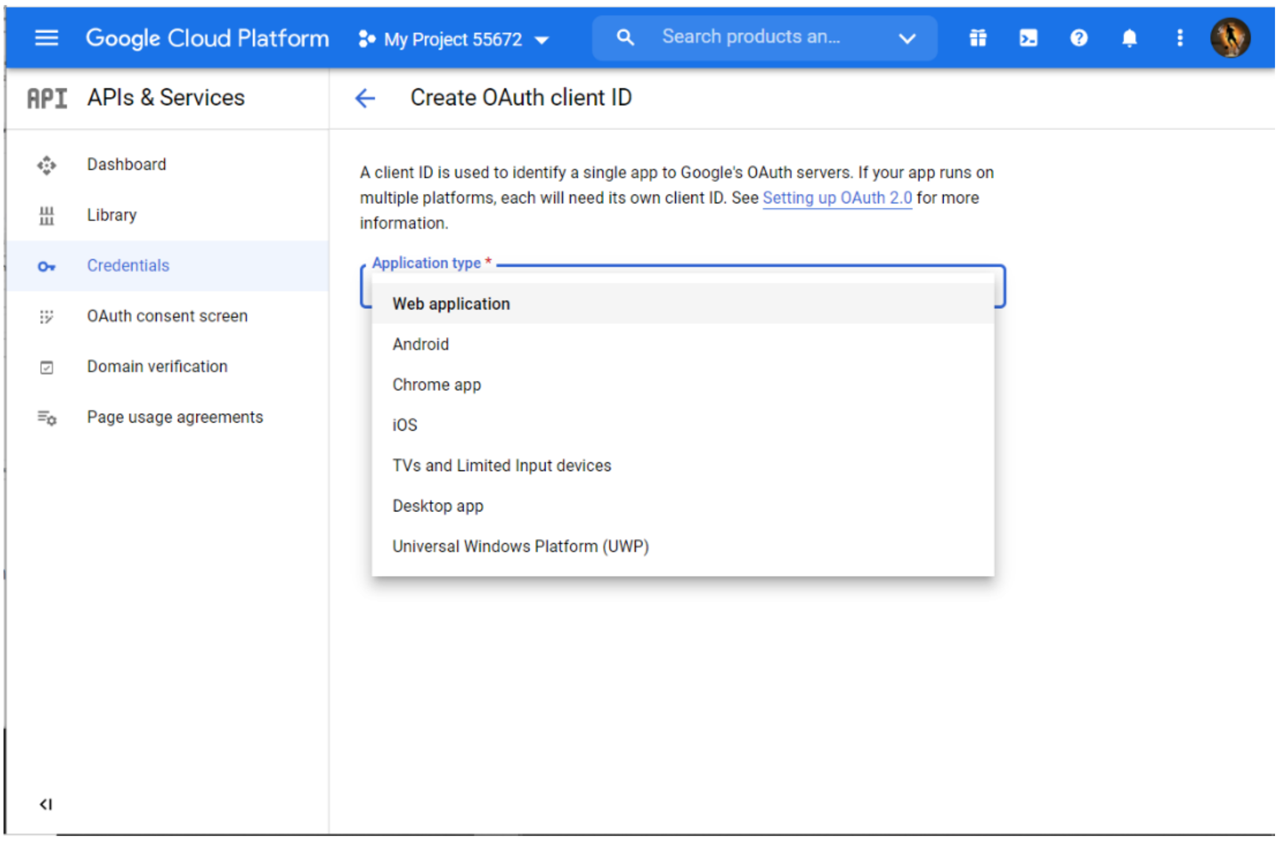
Test users

While publishing status is set to "Testing", only test users are able to access the app. Allowed user cap prior to app verification is 100, and is counted over the entire lifetime of the app. Learn more

On the summary page that appears after this one, click the “RETURN TO DASHBOARD” button.

9. Go back to the credentials page and create new credentials as explained in **step 3.**

10. Provide details as shown in the screenshots below and click the “**CREATE**” button.

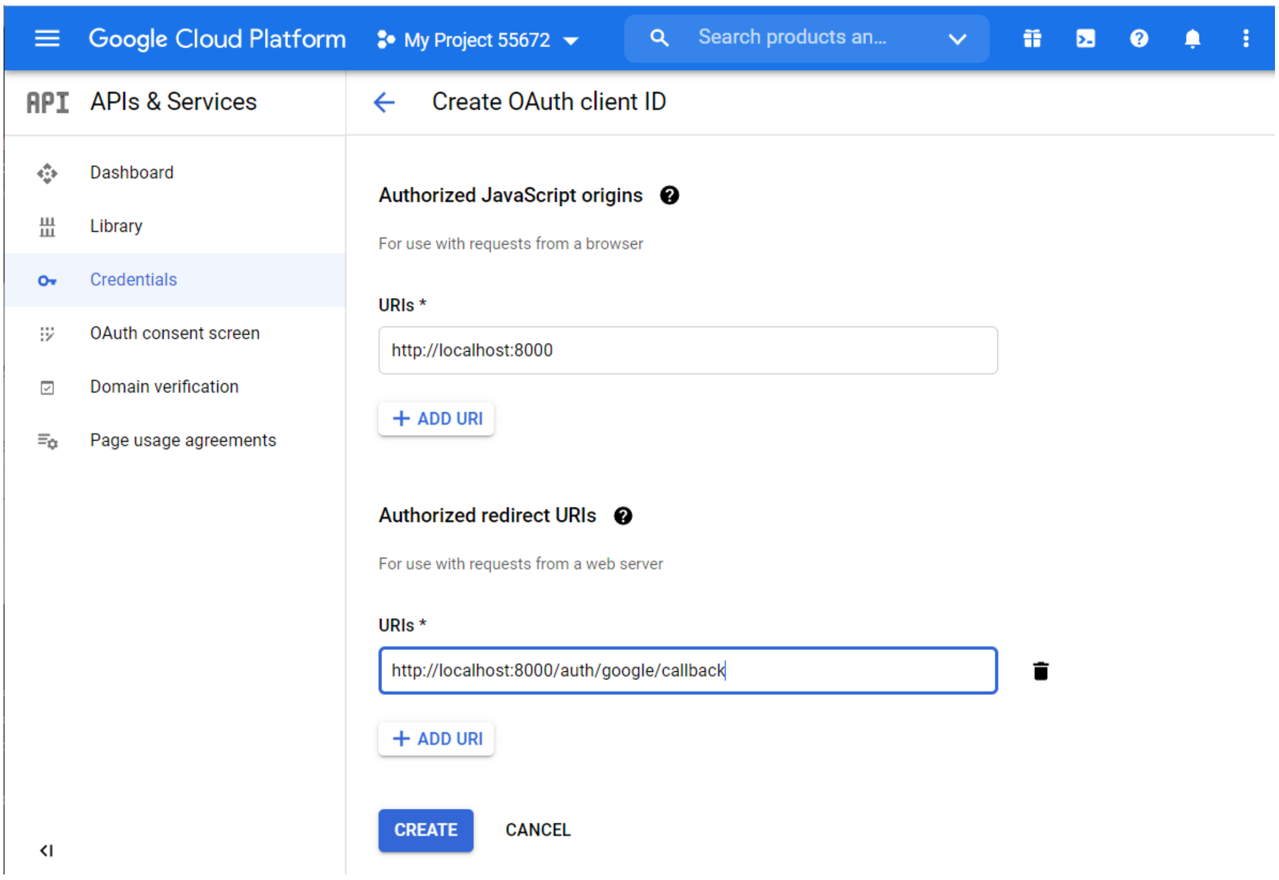


Create OAuth client ID

A client ID is used to identify a single app to Google's OAuth servers. If your app runs on multiple platforms, each will need its own client ID. See Setting up OAuth 2.0 for more information.

Application type *

- Web application
- Android
- Chrome app
- iOS
- TVs and Limited Input devices
- Desktop app
- Universal Windows Platform (UWP)



Create OAuth client ID

Authorized JavaScript origins

For use with requests from a browser

URIs *

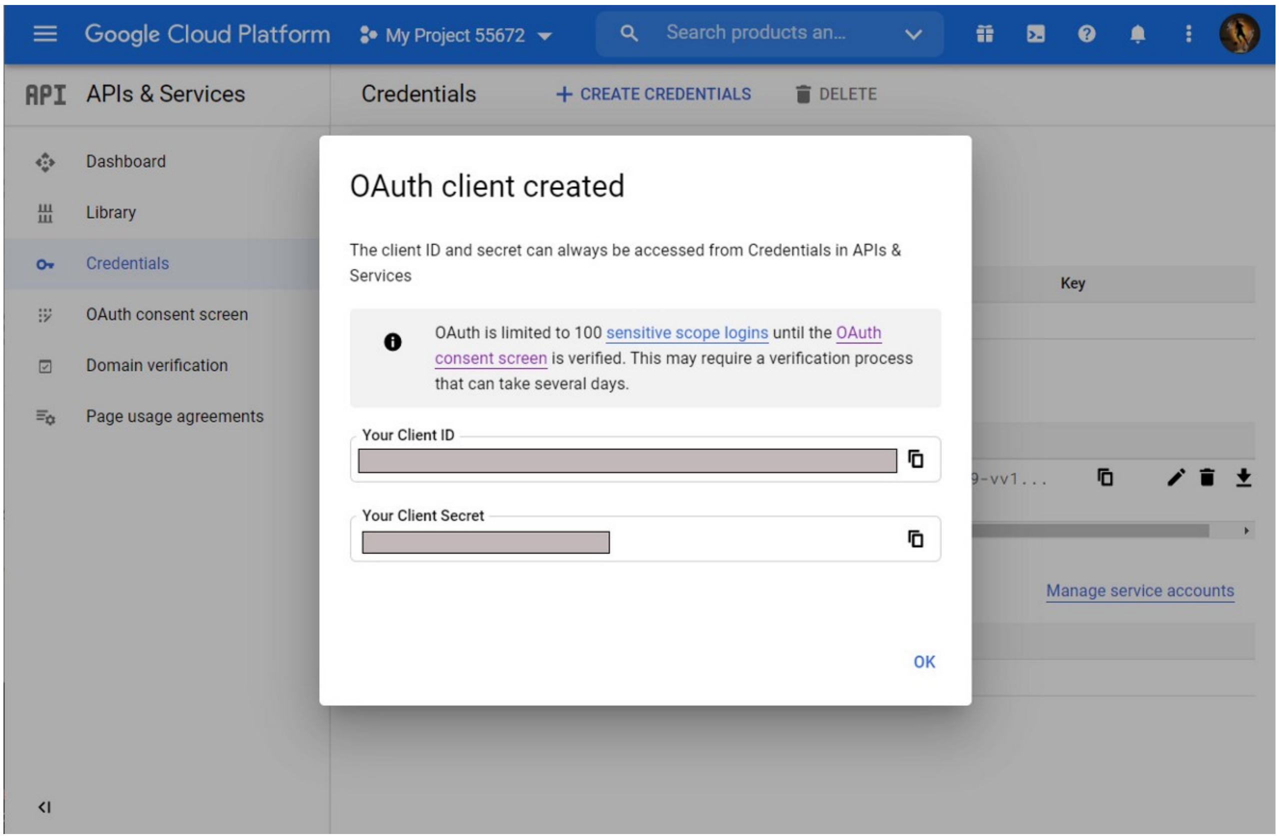
http://localhost:8000

Authorized redirect URIs

For use with requests from a web server

URIs *

http://localhost:8000/auth/google/callback



OAuth client created

The client ID and secret can always be accessed from Credentials in APIs & Services

OAuth is limited to 100 sensitive scope logins until the OAuth consent screen is verified. This may require a verification process that can take several days.

Your Client ID [redacted]

Your Client Secret [redacted]

OK

Add a Google entry to your file.

```
return [    ....    ....    ....    'google' => [        'client_id' =>
env('GOOGLE_CLIENT_ID'),        'client_secret' => env('GOOGLE_CLIENT_SECRET',
'redirect' =>   env('GOOGLE_CLIENT_REDIERECT',    ],]
```

Add the following to your **.** file

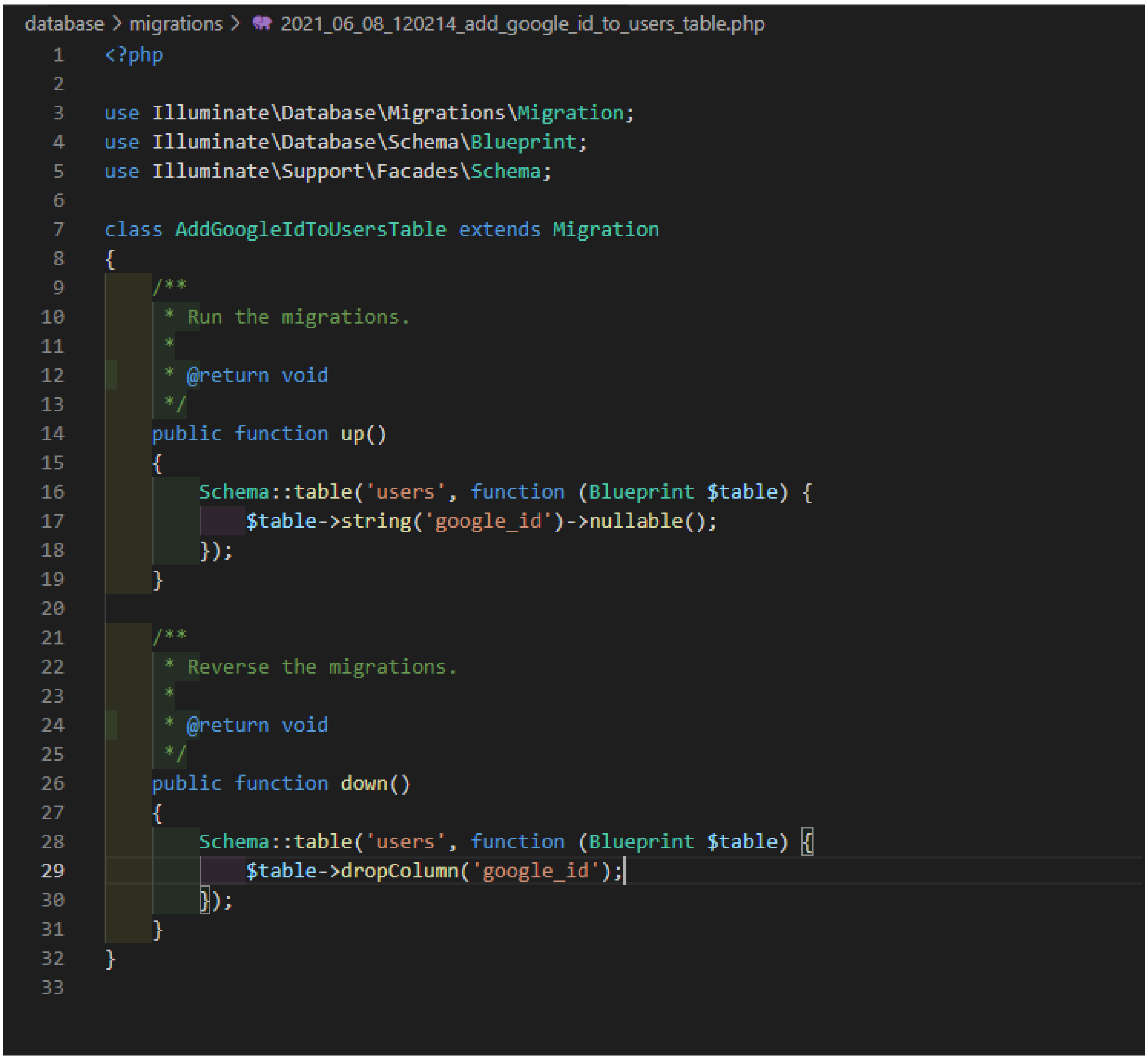
```
GOOGLE_CLIENT_ID="Your Google app id"GOOGLE_CLIENT_SECRET="Your Google client
secret"GOOGLE_CLIENT_REDIERECT="http://localhost:8000/auth/google/callback"
```

6. Add a“google_id” column to your “users” table

Run the following line in your command line to create the migration file to add a *google_id* column the users table

```
php artisan make:migration add_google_id_to_users_table
```

Make the following changes to the file created.


database > migrations > 2021_06_08_120214_add_google_id_to_users_table.php

```php
<?php

use Illuminate\Database\Migrations\Migration;
use Illuminate\Database\Schema\Blueprint;
use Illuminate\Support\Facades\Schema;

class AddGoogleIdToUsersTable extends Migration
{
    /**
     * Run the migrations.
     *
     * @return void
     */
    public function up()
    {
        Schema::table('users', function (Blueprint $table) {
            $table->string('google_id')->nullable();
        });
    }

    /**
     * Reverse the migrations.
     *
     * @return void
     */
    public function down()
    {
        Schema::table('users', function (Blueprint $table) {
            $table->dropColumn('google_id');
        });
    }
}
```

Run the following line on your command line to update your database

```
php artisan migrate:fresh
```

7. Setup your Google authentication routes

Add the following lines to your file.

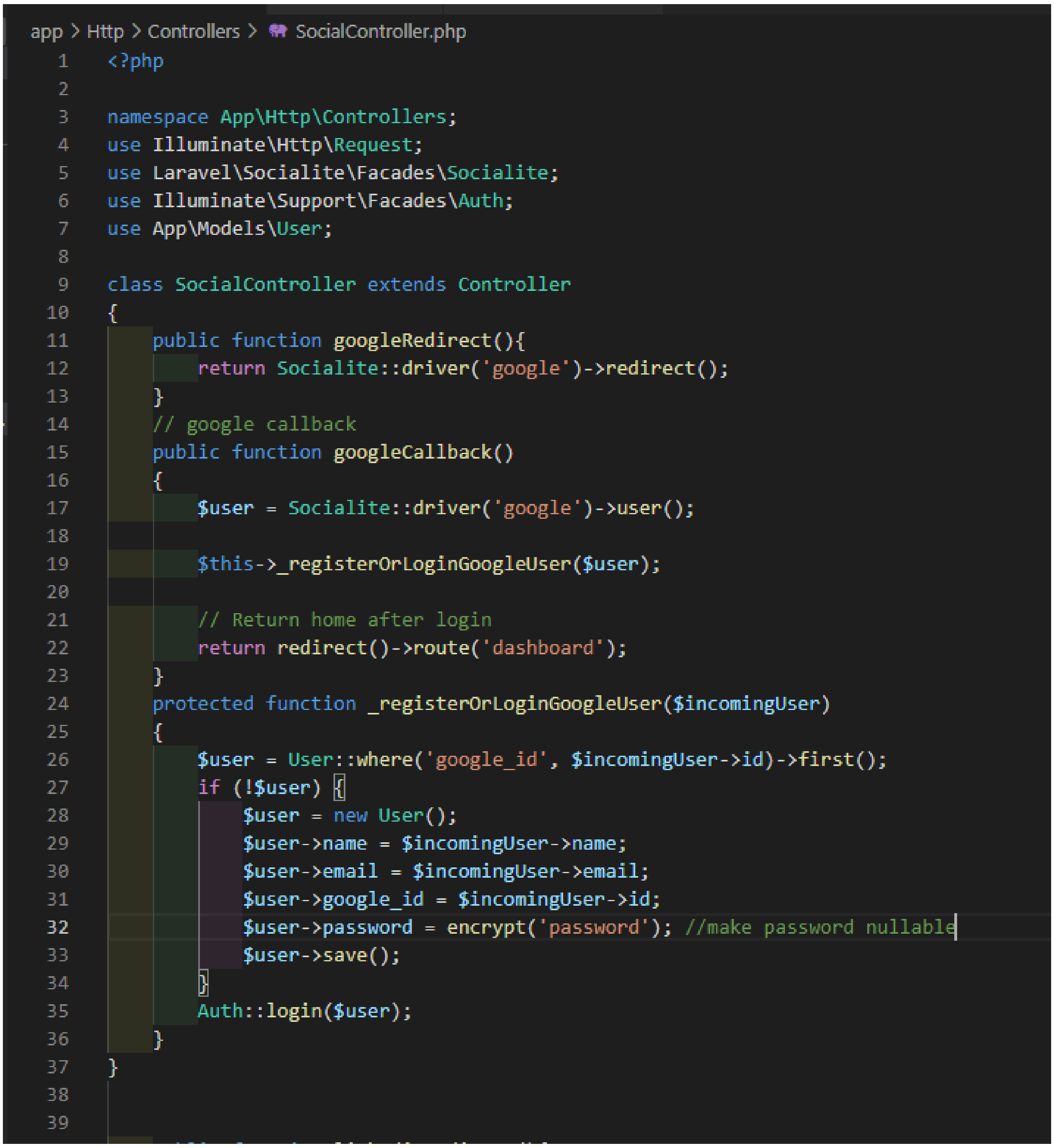
```
use App\Http\Controllers\SocialController;//Google Authentication
RoutesRoute::get('auth/google', [SocialController::class,
'googleRedirect']);Route::get('auth/google/callback', [SocialController::class,
'googleLoginOrRegister']);
```

8. Create and configure controller to handle the routes

Run the following line on your command line to create a controller to handle social authentication.

```
php artisan make:controller SocialController
```

Define *googleRedirect()* and *googleCallback()* as shown below:



```php
<?php

namespace App\Http\Controllers;
use Illuminate\Http\Request;
use Laravel\Socialite\Facades\Socialite;
use Illuminate\Support\Facades\Auth;
use App\Models\User;

class SocialController extends Controller
{
    public function googleRedirect(){
        return Socialite::driver('google')->redirect();
    }
    // google callback
    public function googleCallback()
    {
        $user = Socialite::driver('google')->user();

        $this->_registerOrLoginGoogleUser($user);

        // Return home after login
        return redirect()->route('dashboard');
    }
    protected function _registerOrLoginGoogleUser($incomingUser)
    {
        $user = User::where('google_id', $incomingUser->id)->first();
        if (!$user) {
            $user = new User();
            $user->name = $incomingUser->name;
            $user->email = $incomingUser->email;
            $user->google_id = $incomingUser->id;
            $user->password = encrypt('password'); //make password nullable
            $user->save();
        }
        Auth::login($user);
    }
}


```

9. Add Google login button to your registration and login pages

Add a copy of the link below to both the *register.blade.php* and *login.blade.php* in the **resources\views\auth** folder

```
<div class="flex items-center justify-center mt-4">  <a href="{{
route('login.google') }}" class="underline text-sm text-gray-600 hover:text-gray-
900">    Login With Google  </a></div>
```