Implementing Single Sign-On (SSO) in CodeIgniter 4

League OAuth2 client with Google and Facebook

Step 1: Set Up CodeIgniter 4 Project

Install CodeIgniter 4

```
composer create-project codeigniter4/appstarter codeigniter4-sso
```

Configure Environment Variables

- Rename the env file to .env and set your environment variables.
- Configure the database settings in .env:

```
database.default.hostname = localhost
database.default.database = your_database_name
database.default.username = your_username
database.default.password = your_password
database.default.DBDriver = MySQLi
```

- Or set your database configuration in App/Config/Database.php

Step 2: Set Up Database and Users Table

Create a Migration for Users Table

Create a migration file:

```
php spark make:migration CreateUsersTable
```

Define the users table in the migration file:

```
public function up()
{
   $this->forge->addField([
      'id' => ['type' => 'INT', 'auto_increment' => true, 'unsigned' => true],
      'provider' => ['type' => 'VARCHAR', 'constraint' => 50],
      'provider_id' => ['type' => 'VARCHAR', 'constraint' => 100],
      'email' => ['type' => 'VARCHAR', 'constraint' => 100],
      'name' => ['type' => 'VARCHAR', 'constraint' => 100],
      'created_at' => ['type' => 'DATETIME', 'null' => true],
      'updated_at' => ['type' => 'DATETIME', 'null' => true],
   ]);
   $this->forge->addKey('id', true);
   $this->forge->createTable('users');
}
```

```
public function down()
{
  $this->forge->dropTable('users');
}
```

Run the Migration

```
php spark migrate
```

Step 3: Install League OAuth2 Client

Install the League OAuth2 client and providers for Google and Facebook:

```
composer require league/oauth2-client league/oauth2-google league/oauth2-facebook
```

Step 4: Set Up Google and Facebook Apps

Create Google OAuth Credentials:

a. Go to the Google Developers Console.
b. Create a new project or select an existing one.
c. Go to OAuth consent screen, fill in the details, and save.
d. Under Credentials, create an OAuth 2.0 Client ID and select Web application as the application type.
e. Add the Authorized redirect URI as http://localhost:8080/auth/callback/google.

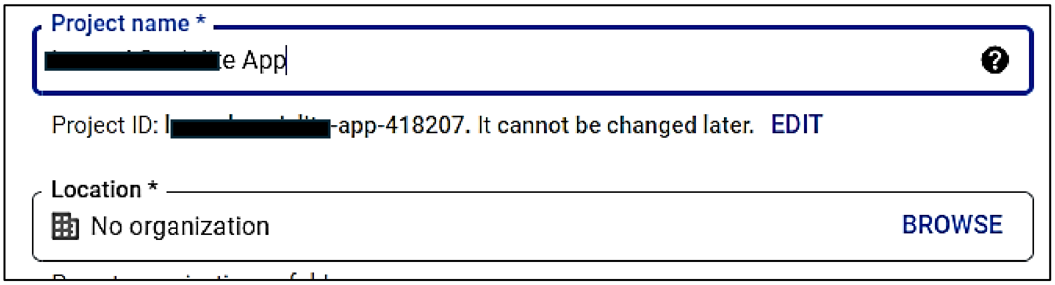
Create Facebook App Credentials:

a. Go to the Facebook Developer Console.
b. Create a new app and select For Everything Else.
c. Go to Settings > Basic and fill in your application details.
d. Go to Add Product and enable Facebook Login.
e. In Facebook Login > Settings, add http://localhost:8080/auth/callback/facebook as the Valid OAuth Redirect URI.

Create app for Google and Facebook

For Google:

a. Login to https://console.developers.google.com/ and create a new project



Project name *

[redacted]e App

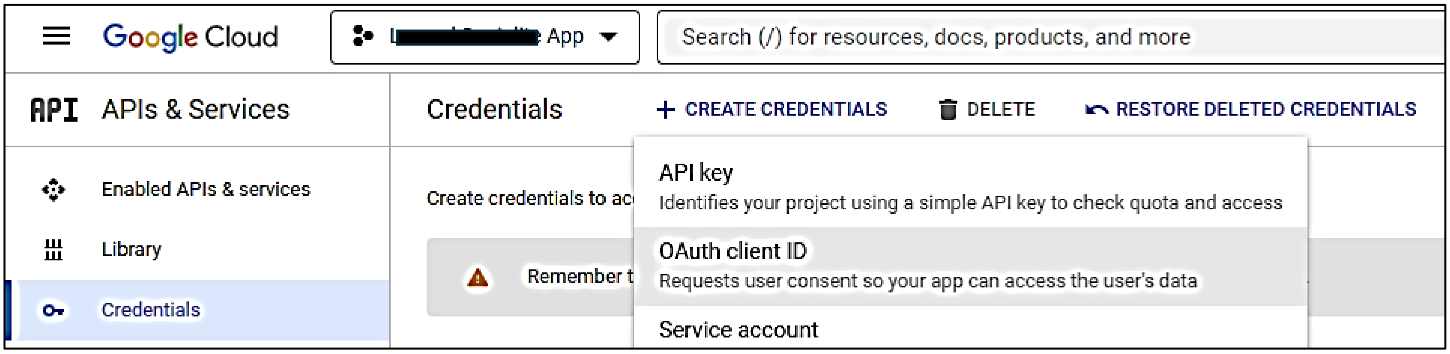
Project ID: [redacted]-app-418207. It cannot be changed later. EDIT

Location *

No organization BROWSE

b. Once created, navigate to that project then go into “Credentials” (click API Overview->Credentials)

c. Click “create credentials”→OAuth Client ID



Google Cloud | [redacted] App ▾ | Search (/) for resources, docs, products, and more

API APIs & Services

Enabled APIs & services

Library

Credentials

Credentials + CREATE CREDENTIALS DELETE RESTORE DELETED CREDENTIALS

Create credentials to ac

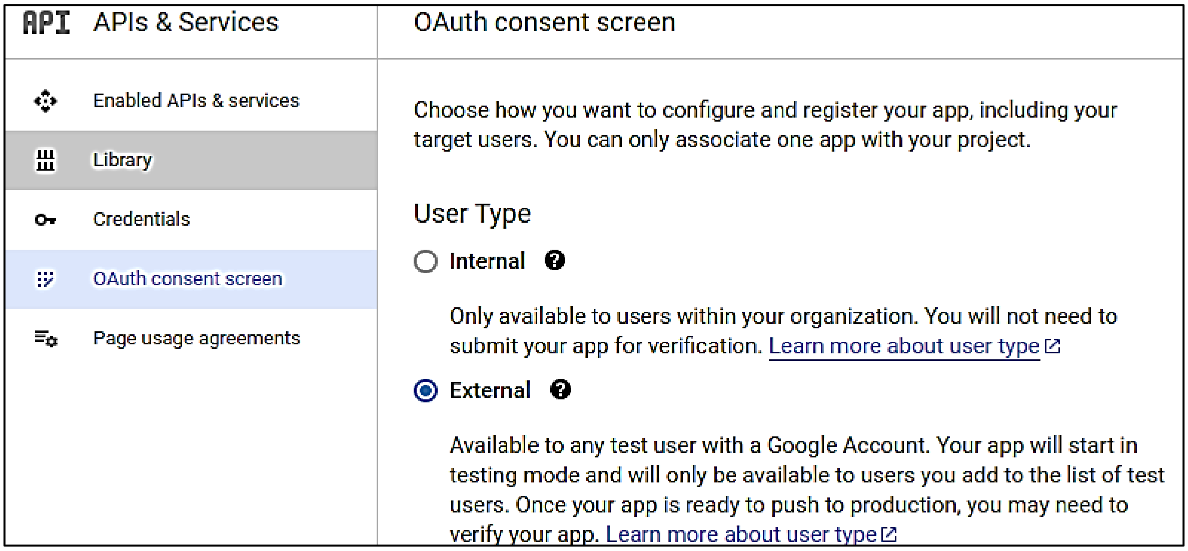
Remember t

API key
Identifies your project using a simple API key to check quota and access

OAuth client ID
Requests user consent so your app can access the user's data

Service account

d. Click “configure consent screen” then select “External” and click create.



API APIs & Services

Enabled APIs & services

Library

Credentials

OAuth consent screen

Page usage agreements

OAuth consent screen

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

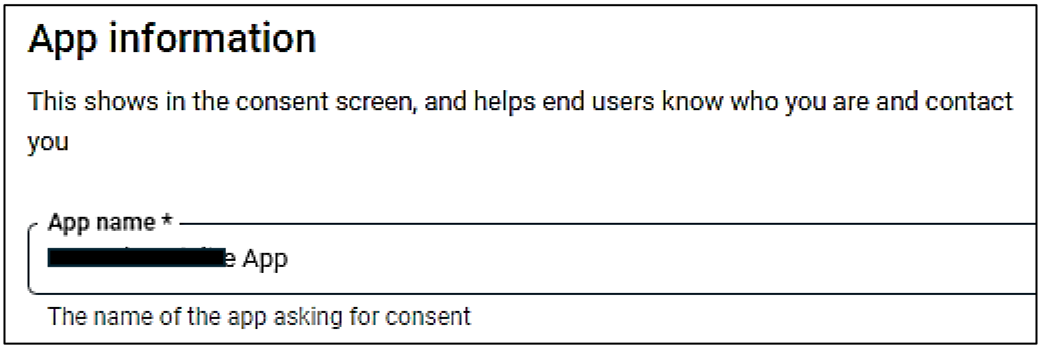
○ Internal

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

● 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 about user type

e. Fill in App Name and Support Email then click **Save and Continue**



App information

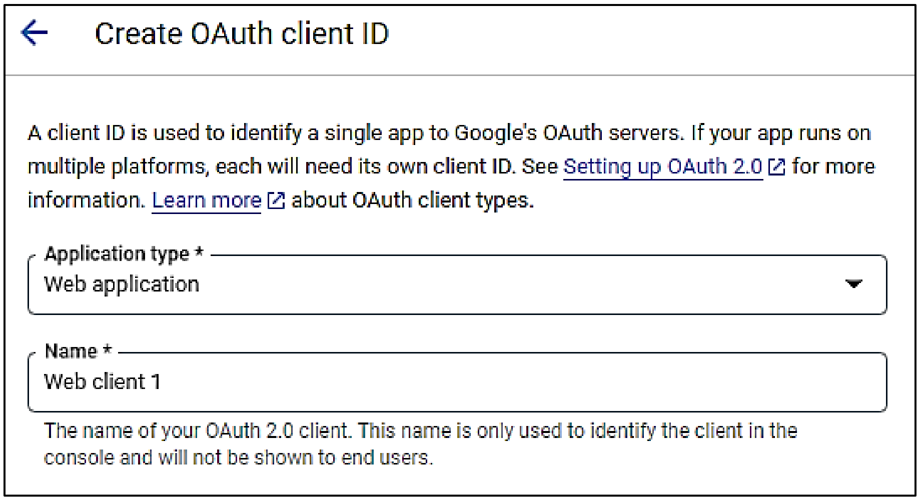
This shows in the consent screen, and helps end users know who you are and contact you

App name *

[redacted] App

The name of the app asking for consent

f. Click back to credentials→create credentials→oauth client id→select application type and add redirect urls then click Create



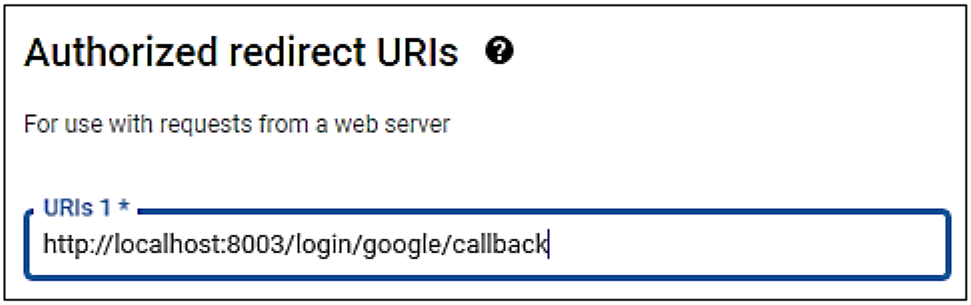
Create OAuth client ID

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. Learn more about OAuth client types.

Application type *: Web application

Name *: Web client 1

The name of your OAuth 2.0 client. This name is only used to identify the client in the console and will not be shown to end users.

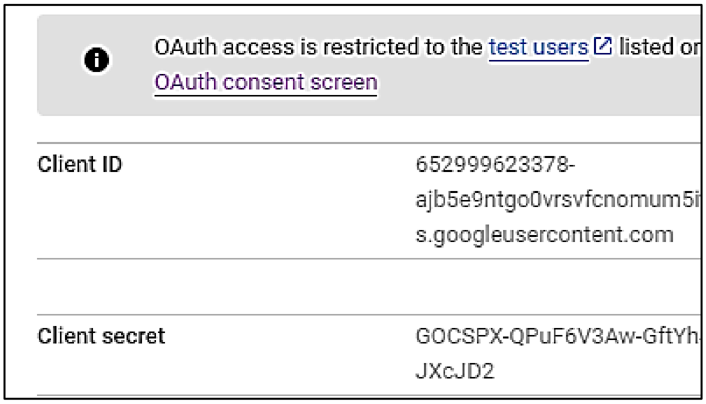


Authorized redirect URIs

For use with requests from a web server

URIs 1 *: http://localhost:8003/login/google/callback

g. Take note of Client ID and Secret



OAuth access is restricted to the test users listed or OAuth consent screen

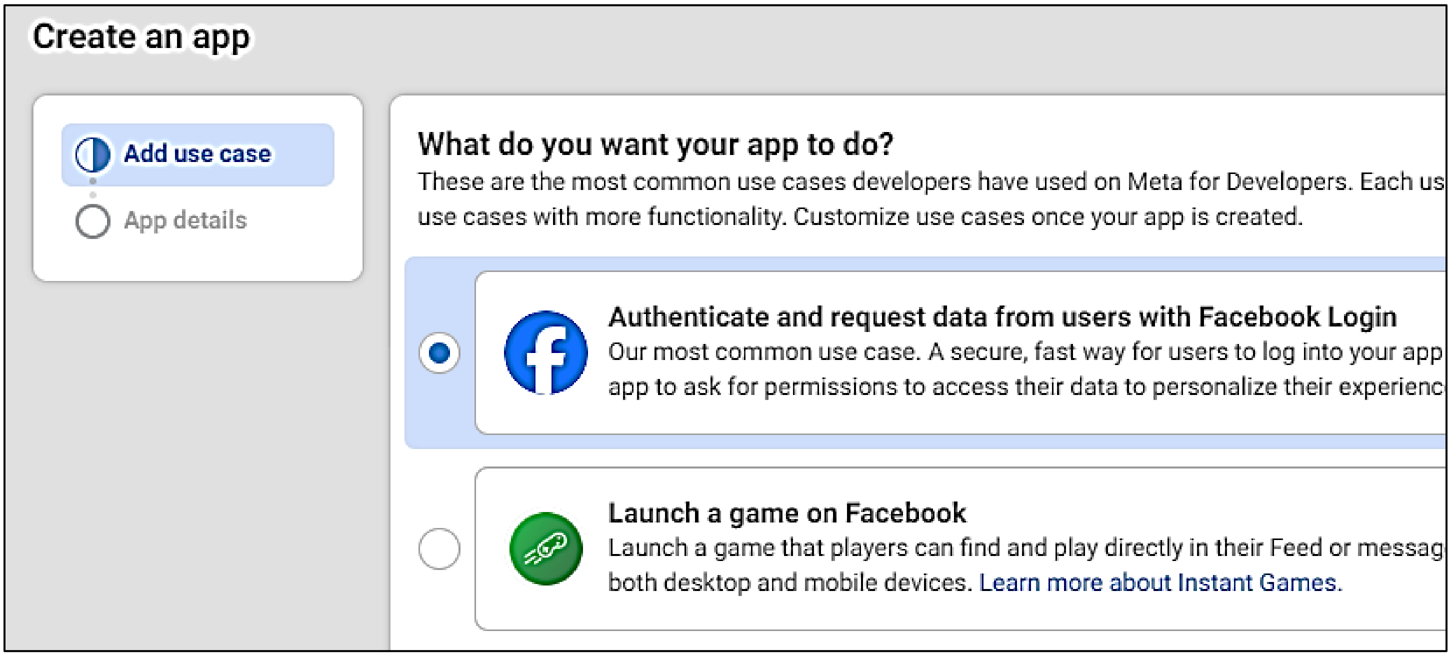
Client ID	652999623378- ajb5e9ntgo0vrsvfcnomum5i s.googleusercontent.com
Client secret	GOCSPX-QPuF6V3Aw-GftYh JXcJD2

h. Note of your client id and secret

```
GOOGLE_CLIENT_ID="65xxxxxx3378-ajb5e9ntaf0mhnn.apps.googleusercontent.com"
GOOGLE_CLIENT_SECRET="GOCSPX-QPxxxxxw-GftYxxxxxUJ9a-JXcJD2"
```

For Facebook

a. Navigate to https://developers.facebook.com/apps and click ‘create app’ → and select facebook login



Create an app

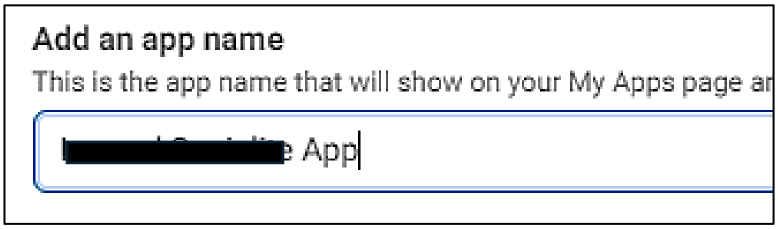
What do you want your app to do?

These are the most common use cases developers have used on Meta for Developers. Each us
use cases with more functionality. Customize use cases once your app is created.

● **Authenticate and request data from users with Facebook Login**
Our most common use case. A secure, fast way for users to log into your app
app to ask for permissions to access their data to personalize their experienc

○ **Launch a game on Facebook**
Launch a game that players can find and play directly in their Feed or messag
both desktop and mobile devices. Learn more about Instant Games.

b. Add an app name then click ‘Create App’ (facebook login will popup)

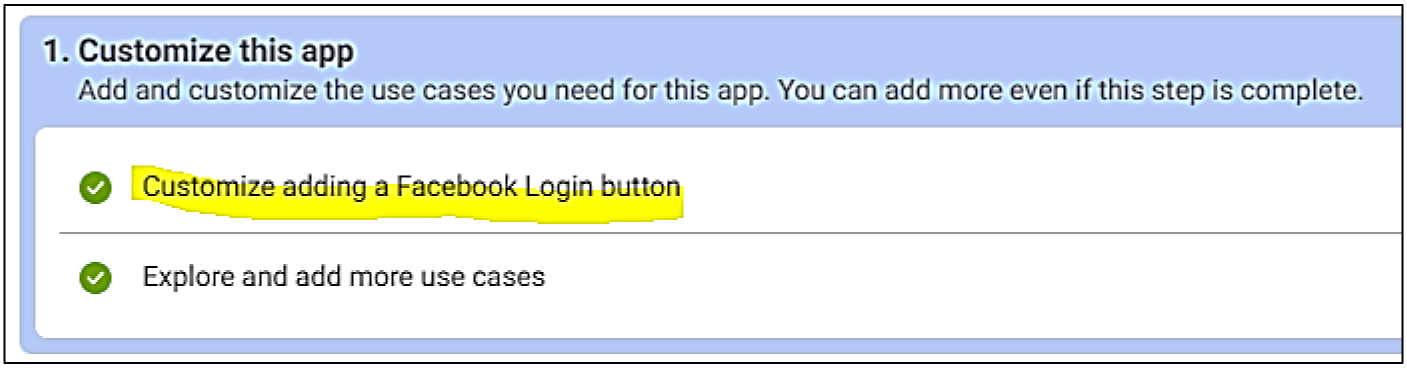


Add an app name

This is the app name that will show on your My Apps page ar

[redacted] App

c. Look for setting up facebook login as follows:



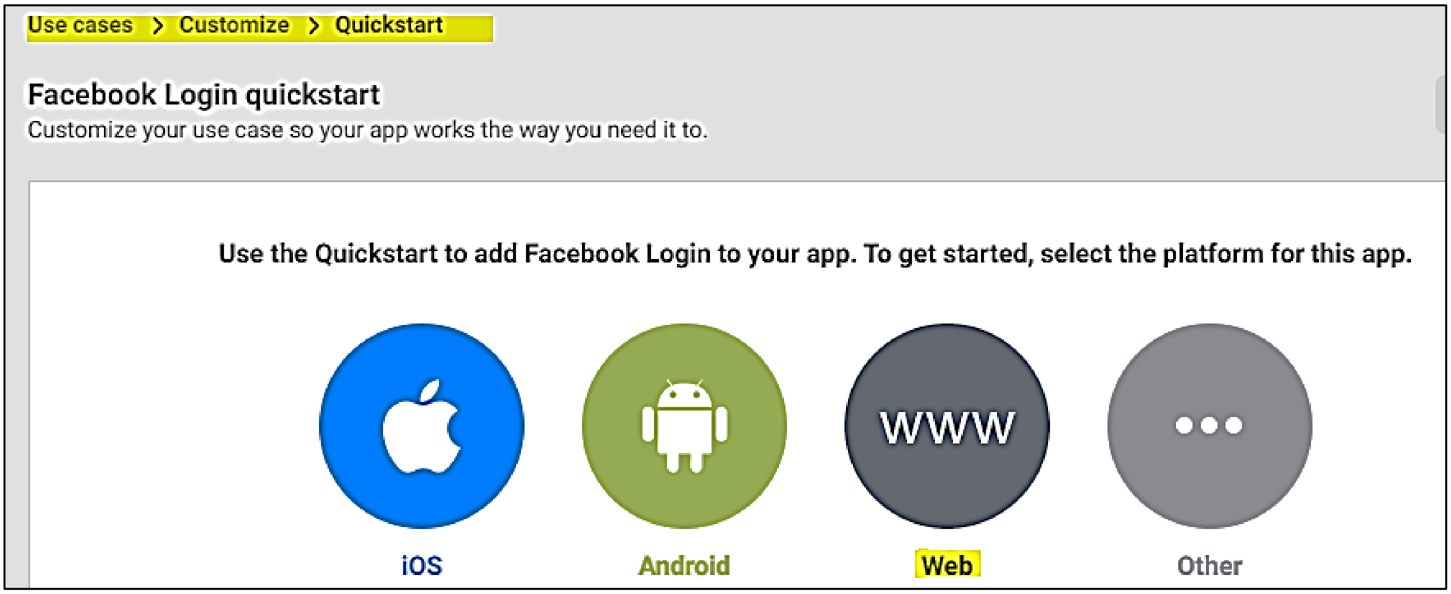
1. Customize this app

Add and customize the use cases you need for this app. You can add more even if this step is complete.

✓ Customize adding a Facebook Login button

✓ Explore and add more use cases

d. Look for login using WWW



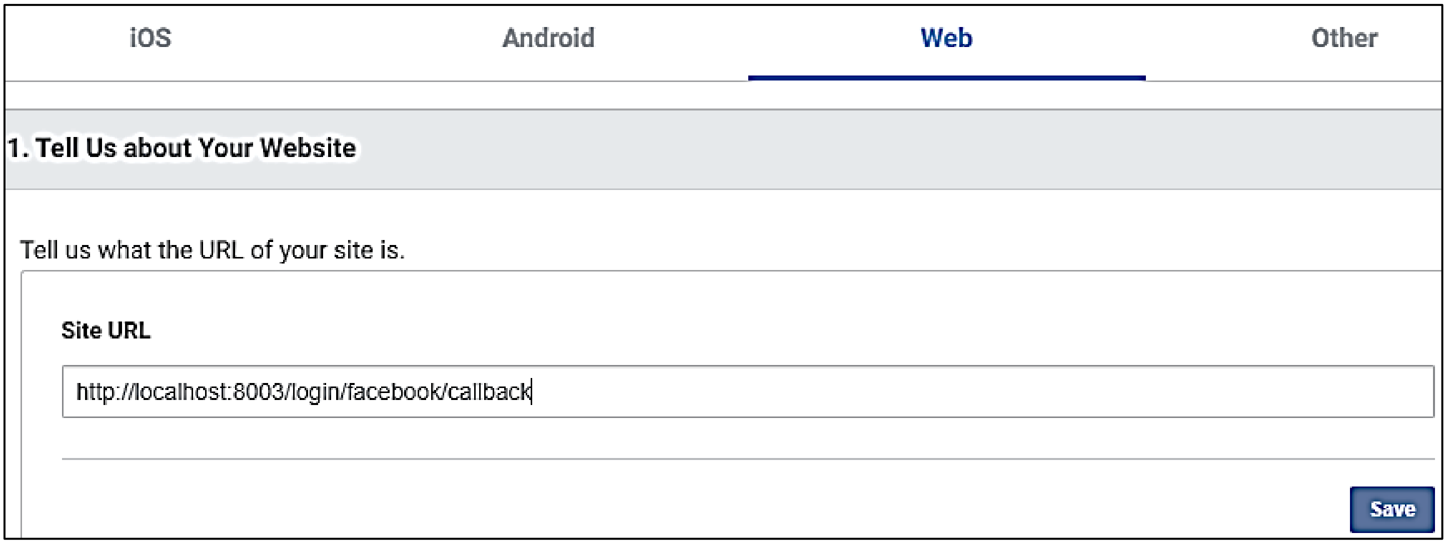
Use cases > Customize > Quickstart

Facebook Login quickstart

Customize your use case so your app works the way you need it to.

Use the Quickstart to add Facebook Login to your app. To get started, select the platform for this app.

e. Enter site URL and click Save then click the Continue button



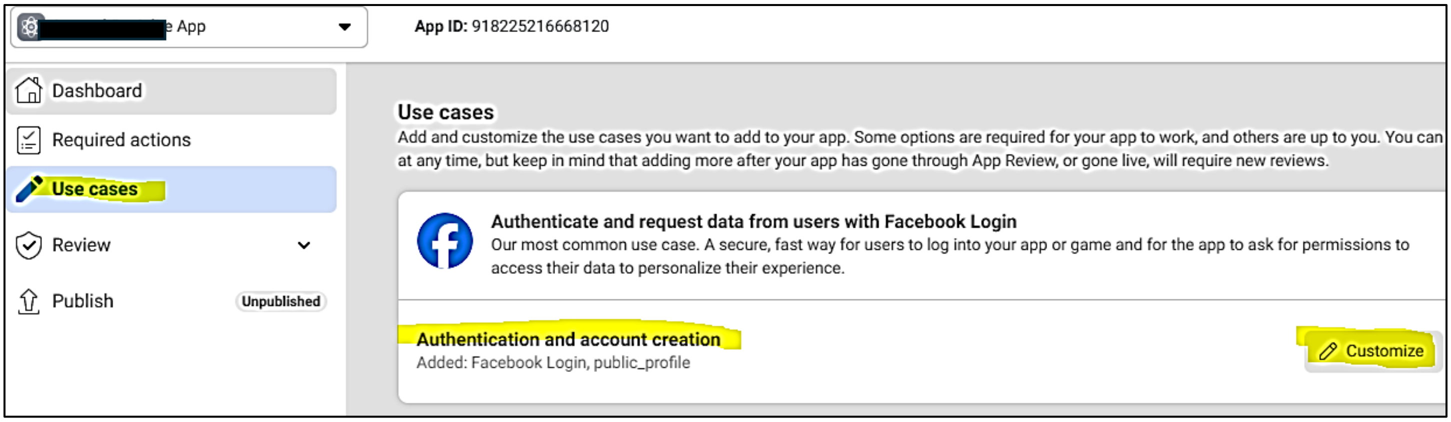
1. Tell Us about Your Website

Tell us what the URL of your site is.

Site URL

http://localhost:8003/login/facebook/callback

f. From there, you can navigate back to the Facebook App Dashboard→Use Case



[redacted] e App

App ID: 918225216668120

Use cases

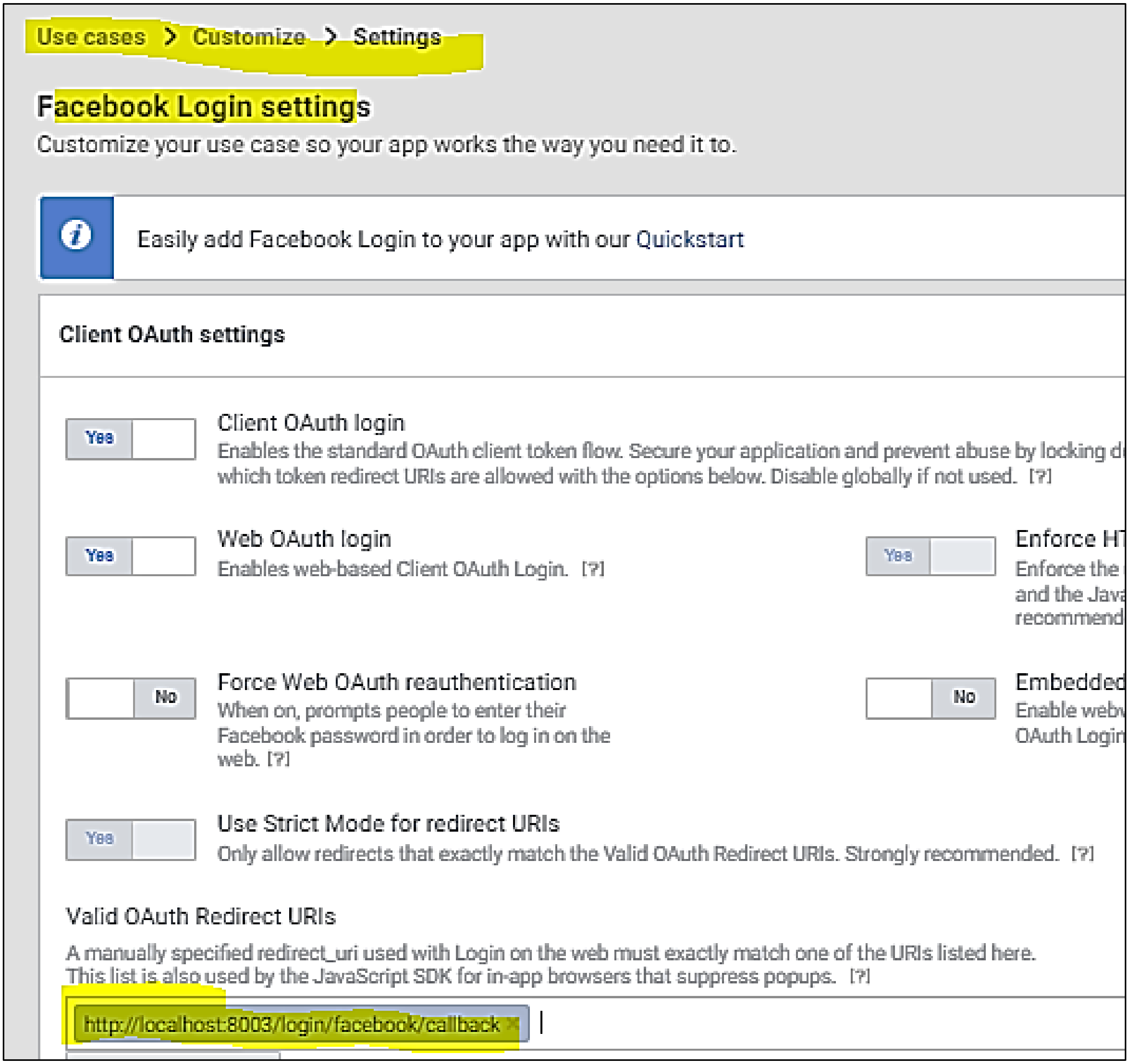
Add and customize the use cases you want to add to your app. Some options are required for your app to work, and others are up to you. You can at any time, but keep in mind that adding more after your app has gone through App Review, or gone live, will require new reviews.

Authenticate and request data from users with Facebook Login

Our most common use case. A secure, fast way for users to log into your app or game and for the app to ask for permissions to access their data to personalize their experience.

Authentication and account creation

Added: Facebook Login, public_profile



Use cases > Customize > Settings

Facebook Login settings

Customize your use case so your app works the way you need it to.

Easily add Facebook Login to your app with our Quickstart

Client OAuth settings

Yes — Client OAuth login
Enables the standard OAuth client token flow. Secure your application and prevent abuse by locking d
which token redirect URIs are allowed with the options below. Disable globally if not used. [?]

Yes — Web OAuth login
Enables web-based Client OAuth Login. [?]

Yes — Enforce HT
Enforce the
and the Java
recommend

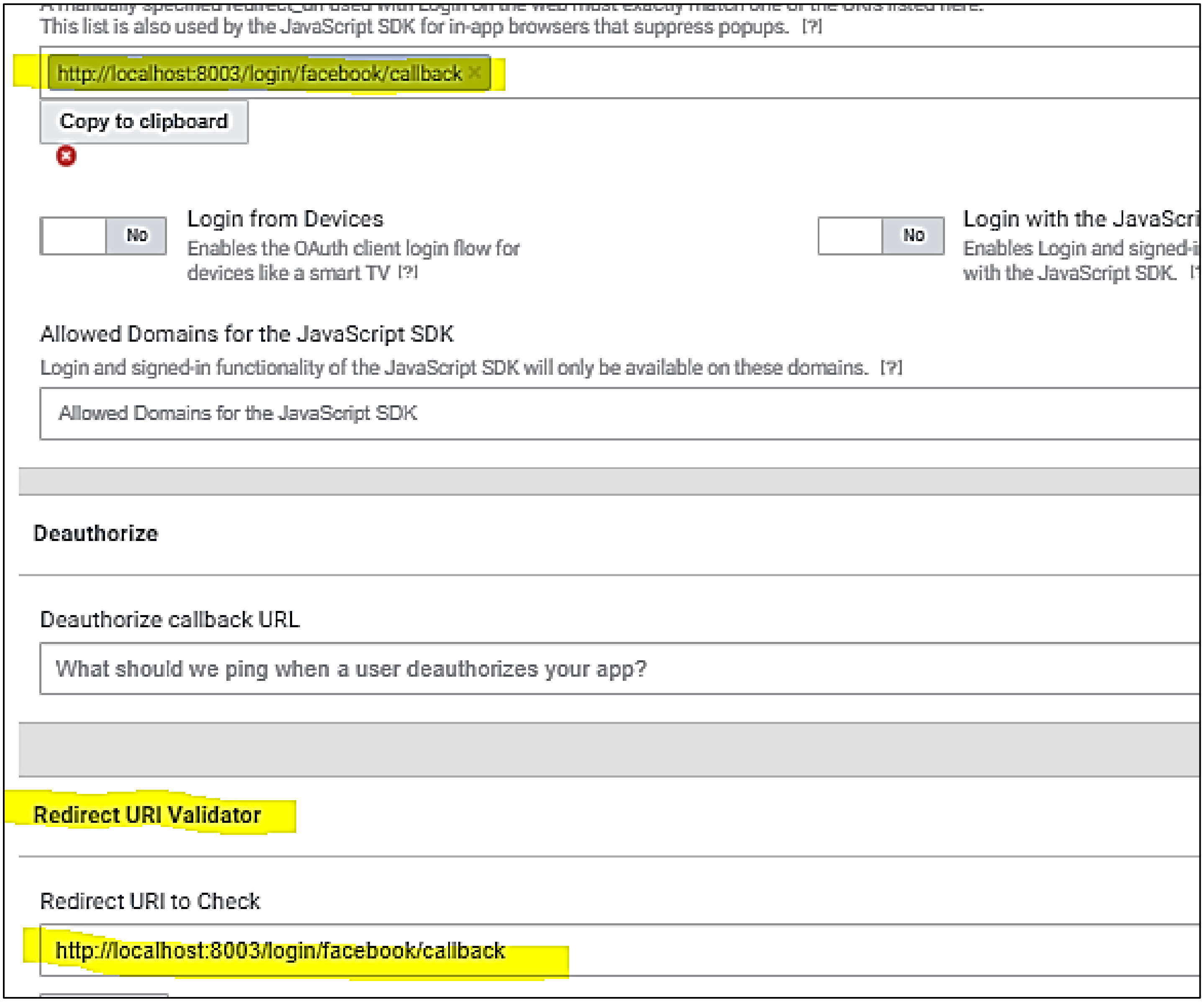
No — Force Web OAuth reauthentication
When on, prompts people to enter their Facebook password in order to log in on the web. [?]

No — Embedded
Enable web
OAuth Login

Yes — Use Strict Mode for redirect URIs
Only allow redirects that exactly match the Valid OAuth Redirect URIs. Strongly recommended. [?]

Valid OAuth Redirect URIs
A manually specified redirect_uri used with Login on the web must exactly match one of the URIs listed here.
This list is also used by the JavaScript SDK for in-app browsers that suppress popups. [?]

http://localhost:8003/login/facebook/callback ×



A manually specified redirect_uri used with Login on the web must exactly match one of the URIs listed here. This list is also used by the JavaScript SDK for in-app browsers that suppress popups. [?]

http://localhost:8003/login/facebook/callback ×

Copy to clipboard

No Login from Devices
Enables the OAuth client login flow for devices like a smart TV [?]

No Login with the JavaScri
Enables Login and signed-i
with the JavaScript SDK. [

Allowed Domains for the JavaScript SDK
Login and signed-in functionality of the JavaScript SDK will only be available on these domains. [?]

Allowed Domains for the JavaScript SDK

Deauthorize

Deauthorize callback URL

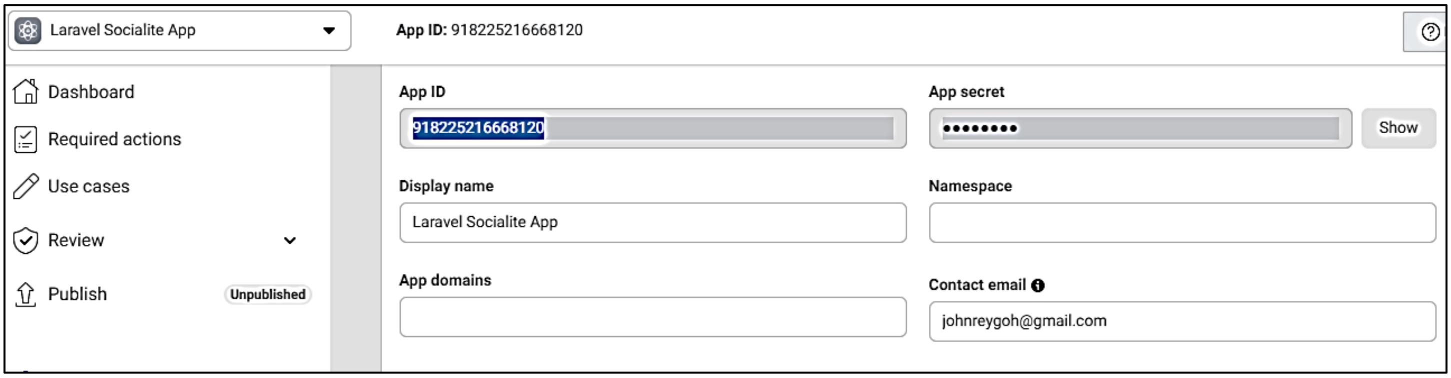
What should we ping when a user deauthorizes your app?

Redirect URI Validator

Redirect URI to Check

http://localhost:8003/login/facebook/callback

g. Go back to facebook app dashboard→app settings→basic→get app id and app secret



Laravel Socialite App

App ID: 918225216668120

App ID
918225216668120

App secret
••••••••
Show

Display name
Laravel Socialite App

Namespace

App domains

Contact email
johnreygoh@gmail.com

h. Note your Facebook Client ID and Client Secret

```
FACEBOOK_CLIENT_ID="91822xxxxx8120"
FACEBOOK_CLIENT_SECRET="1e857b59xxxxxxxxxx17c30f80fde2783e0"
```

Step 5: Configure Environment Variables for Google and Facebook

Add your Google and Facebook credentials to the .env file:

```
GOOGLE_CLIENT_ID=your_google_client_id
GOOGLE_CLIENT_SECRET=your_google_client_secret
GOOGLE_REDIRECT_URI=http://localhost:8080/auth/callback/google

FACEBOOK_CLIENT_ID=your_facebook_client_id
FACEBOOK_CLIENT_SECRET=your_facebook_client_secret
FACEBOOK_REDIRECT_URI=http://localhost:8080/auth/callback/facebook
```

Step 6: Create OAuth Controllers and Logic

Create User Model

Create app/Models/UserModel.php to manage user data:

```
namespace App\Models;

use CodeIgniter\Model;

class UserModel extends Model
{
    protected $table = 'users';
    protected $primaryKey = 'id';
    protected $allowedFields = ['provider', 'provider_id', 'email', 'name'];
}
```

Create Auth Controller

Create app/Controllers/AuthController.php to handle authentication:

```
namespace App\Controllers;

use App\Models\UserModel;
use CodeIgniter\Controller;
use League\OAuth2\Client\Provider\Google;
use League\OAuth2\Client\Provider\Facebook;

class AuthController extends Controller
{
    public function redirect($provider)
    {
        if ($provider === 'google') {
            $providerInstance = new Google([
                'clientId'     => getenv('GOOGLE_CLIENT_ID'),
                'clientSecret' => getenv('GOOGLE_CLIENT_SECRET'),
            'redirectUri'  => getenv('GOOGLE_REDIRECT_URI'),
        ]);
    } elseif ($provider === 'facebook') {
        $providerInstance = new Facebook([
            'clientId'     => getenv('FACEBOOK_CLIENT_ID'),
            'clientSecret' => getenv('FACEBOOK_CLIENT_SECRET'),
            'redirectUri'  => getenv('FACEBOOK_REDIRECT_URI'),
        ]);
    } else {
        return redirect()->back()->with('error', 'Unsupported provider');
    }

    $authorizationUrl = $providerInstance->getAuthorizationUrl();
    session()->set('oauth2state', $providerInstance->getState());

    return redirect()->to($authorizationUrl);
}

public function callback($provider)
{
    $userModel = new UserModel();

    if ($provider === 'google') {
        $providerInstance = new Google([
            'clientId'     => getenv('GOOGLE_CLIENT_ID'),
            'clientSecret' => getenv('GOOGLE_CLIENT_SECRET'),
            'redirectUri'  => getenv('GOOGLE_REDIRECT_URI'),
        ]);
    } elseif ($provider === 'facebook') {
        $providerInstance = new Facebook([
            'clientId'     => getenv('FACEBOOK_CLIENT_ID'),
            'clientSecret' => getenv('FACEBOOK_CLIENT_SECRET'),
            'redirectUri'  => getenv('FACEBOOK_REDIRECT_URI'),
        ]);
    } else {
        return redirect()->back()->with('error', 'Unsupported provider');
    }

    if (session('oauth2state') !== $this->request->getGet('state')) {
        session()->remove('oauth2state');
        return redirect()->to('/')->with('error', 'Invalid state');
    }

    try {
        $token = $providerInstance->getAccessToken('authorization_code', [
        'code' => $this->request->getGet('code')
      ]);

      $user = $providerInstance->getResourceOwner($token);

      $userData = [
        'provider'    => $provider,
        'provider_id' => $user->getId(),
        'email'      => $user->getEmail(),
        'name'       => $user->getName(),
      ];

      $existingUser = $userModel->where('provider', $provider)
                   ->where('provider_id', $user->getId())
                   ->first();

      if (!$existingUser) {
        $userModel->insert($userData);
      } else {
        $userData['id'] = $existingUser['id'];
      }

      session()->set('logged_in_user', $userData);

      return redirect()->to('/dashboard')->with('message', 'Logged in successfully!');
    } catch (\Exception $e) {
      return redirect()->to('/')->with('error', 'Failed to get user details.');
    }
  }
}
```

Set Up Routes

In app/Config/Routes.php, add routes for redirecting and handling callbacks:

```
$routes->get('auth/redirect/(:segment)', 'AuthController::redirect/$1');
$routes->get('auth/callback/(:segment)', 'AuthController::callback/$1');
```

Step 7: Create the Frontend Links

In your view, create login buttons for Google and Facebook:

```
<a href="<?= base_url('auth/redirect/google') ?>">Login with Google</a>
<a href="<?= base_url('auth/redirect/facebook') ?>">Login with Facebook</a>
```

Step 8: Test the SSO Flow

Start CodeIgniter:

```
php spark serve
```

Visit the Login Page:

- ✓ Click on the Login with Google or Login with Facebook button.
- ✓ You’ll be redirected to the Google or Facebook login page.
- ✓ After successfully logging in, you’ll be redirected back to your app and should see a success message or be redirected to your dashboard.