Php Laravel Socialite And Android Google Sign In Operation

1. Install Socialite: `composer require laravel/socialite`

2. Add following codes in `config/services.php`

```
'google' => [
    'client_id' => env('GOOGLE_CLIENT_ID'), // Your Google Client ID
    'client_secret' => env('GOOGLE_CLIENT_SECRET'), // Your Google Client Secret
    'redirect' => 'http://www.codesenior.com',
    ],
```

And add GOOGLE_CLIENT_ID and GOOGLE_CLIENT_SECRET variables in .env file:

```
GOOGLE_CLIENT_ID=692373818685-1s057a8mja62g3i7cmj88v2spt3d8b8e.apps.googleusercontent.com
GOOGLE_CLIENT_SECRET=c-4CsKAagTYHVyPKbGVcbAsr
```

3. Create a controller and add login function:

```
public function login( Request $request ) {
    $googleAuthCode = $request->input( 'googleAuthCode' );
    $accessTokenResponse= Socialite::driver('google')->getAccessTokenResponse($googleAuthCode);
    $accessToken=$accessTokenResponse["access_token"];
    $expiresIn=$accessTokenResponse["expires_in"];
    $idToken=$accessTokenResponse["id_token"];

$refreshToken=isset($accessTokenResponse["refresh_token"])?$accessTokenResponse["refresh_token"]:"";
    $tokenType=$accessTokenResponse["token_type"];
    $user = Socialite::driver('google')->userFromToken($accessToken);


}
```

At Line 1, googleAuthCode parameter comes from Android app:

```
@Override
protected void onCreate(@Nullable Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_google_sign_in);
    SignInButton btnSignIn = findViewById(R.id.sign_in_button);
    btnSignIn.setSize(SignInButton.SIZE_STANDARD);
    GoogleSignInOptions gso = new
GoogleSignInOptions.Builder(GoogleSignInOptions.DEFAULT_SIGN_IN)
    .requestServerAuthCode("692373818685-
1s057a8mja62g3i7cmj88v2spt3d8b8e.apps.googleusercontent.com")
    .requestEmail()
    .build();
    mGoogleSignInClient = GoogleSignIn.getClient(this, gso);
    btnSignIn.setOnClickListener(this);
}
@Override
protected void onStart() {
    super.onStart();
    // Check for existing Google Sign In account, if the user is already signed in
    // the GoogleSignInAccount will be non-null.
    GoogleSignInAccount account = GoogleSignIn.getLastSignedInAccount(this);
    updateUI(account);
}
private void updateUI(GoogleSignInAccount account) {
    if (account != null) {
        try {
            String email = account.getEmail();
            String fullName = account.getDisplayName();
            String authCode = account.getServerAuthCode();
            authenticate(email, authCode, fullName);

```

```
            } catch (Exception e) {
                if (e.getMessage() != null)
                Log.e(TAG, e.getMessage());
                Toast.makeText(getApplicationContext(), getString(R.string.unhandled_error),
Toast.LENGTH_SHORT).show();
            }
        }

}
private void authenticate(String email, String googleAuthCode, String fullName) throws
IOException {
        Retrofit retrofit = ApiClient.getClient(Config.REST_API);
        TokenService service = retrofit.create(TokenService.class);
        service.getOath(new Token(email, googleAuthCode, fullName)).enqueue(new
Callback<TokenResponse>() {
            @Override
            public void onResponse(Call<TokenResponse> call, Response<TokenResponse> response) {
                TokenResponse tokenResponse = response.body();
                if (tokenResponse == null) {
                    Toast.makeText(getApplicationContext(), getString(R.string.unhandled_error),
Toast.LENGTH_SHORT).show();
                } else {
                    SharedPreferencesUtil.write(getApplicationContext(), "access_token",
tokenResponse.getData());
                    startActivity(new Intent(GoogleSignActivity.this, MainActivity.class));
                    finish();
                }
            }
            @Override
            public void onFailure(Call<TokenResponse> call, Throwable t) {
                Toast.makeText(getApplicationContext(), getString(R.string.unhandled_error),
Toast.LENGTH_SHORT).show();
            }
        });
}
private void signIn() {
        Intent signInIntent = mGoogleSignInClient.getSignInIntent();
        startActivityForResult(signInIntent, RC_SIGN_IN);
}

@Override
public void onActivityResult(int requestCode, int resultCode, Intent data) {
        super.onActivityResult(requestCode, resultCode, data);

        // Result returned from launching the Intent from GoogleSignInClient.getSignInIntent(...);
        if (requestCode == RC_SIGN_IN) {
            // The Task returned from this call is always completed, no need to attach
            // a listener.
            Task<GoogleSignInAccount> task = GoogleSignIn.getSignedInAccountFromIntent(data);
            handleSignInResult(task);
        }
}

private void handleSignInResult(Task<GoogleSignInAccount> completedTask) {
        try {
            GoogleSignInAccount account = completedTask.getResult(ApiException.class);

            // Signed in successfully, show authenticated UI.
            updateUI(account);
        } catch (ApiException e) {
            // The ApiException status code indicates the detailed failure reason.
            // Please refer to the GoogleSignInStatusCodes class reference for more information.
            Log.w(TAG, "signInResult:failed code=" + e.getStatusCode());
            updateUI(null);
        }
}
```

For more information about Android configuration, read offical documentation: https://developers.google.com/identity/sign-

in/android/start-integrating

At line 2. **getAccessTokenResponse()** function will return access token info without error, but we should add **redirect url** into Authorised redirect URIs place in Web client (Auto-created for Google Sign-in) API, where you can see in https://console.developers.google.com page.

At Line 8, we access google user detailed information. Please note that, we should enable **Google Plus API** in https://console.developers.google.com page.