

Zoe Chew

1K Followers · About

Follow

Sign in


Get started




In 5 mins: Set up Google login to sign up users on Django



Zoe Chew · Jul 27, 2019 · 6 min read

Note: This article is part of my [toolkit newsletters](#)  where I share resources about building things.

I work with Python and Django as my tech stacks in building [web apps](#) 

Today, I want to share about building user registration through Google OAuth. I also vlog this in [video](#) 

I will cover the basic setup and keep the context specific to Django app & social login with Google. Therefore, this post is ideal for any beginners.



1. How to sign up users on your site ❤️

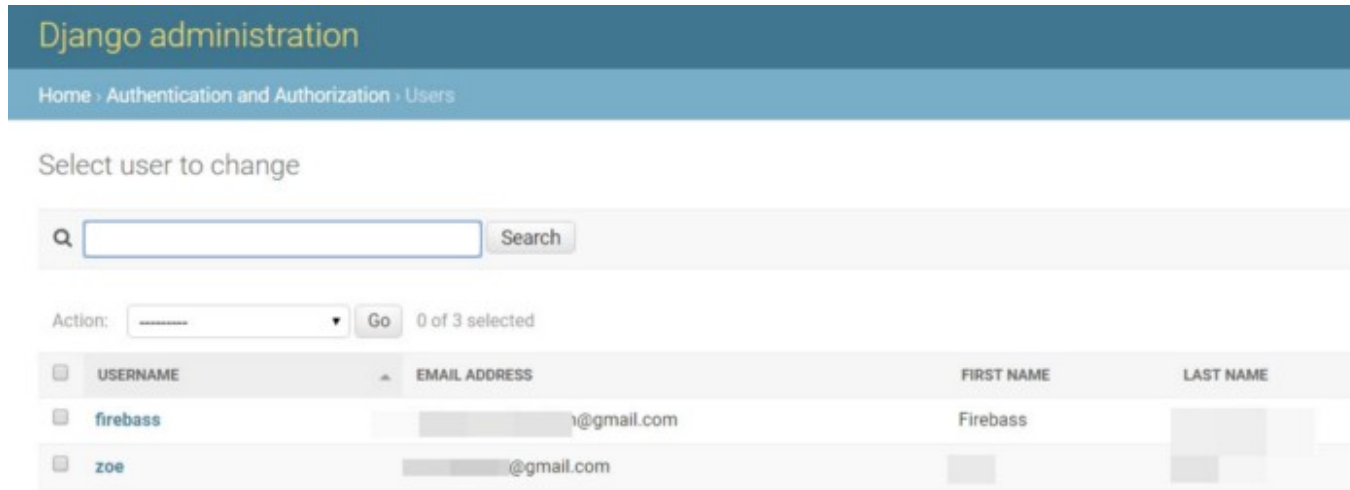
Users registration is part of the customer onboarding process for tech-based products.

If you're a beginner in programming, building a custom login might be complex to start.

Luckily, Django comes with a third-party `django-allauth` package that helps you set up social login in few minutes.

Google login is fast because it doesn't require to create a new User account.

Once users access your app via Gmail, you'll be able to see their email addresses in your backend:



2. Create a django project 🐱💕

Before that happens, let's create a new directory on desktop and call it "google_login".

In the new directory, you will need to set up virtual environment, install django, create django project (my_project) and create an app (social_app).

On terminal, use these command lines to proceed:

```
virtualenv .  
./Scripts/activate  
pip install django  
django-admin.py startproject my_project .  
python manage.py startapp social_app  
python manage.py runserver  
python manage.py migrate  
python manage.py runserver
```

To check if your setup works, go to (<http://127.0.0.1:8000/>) in your browser. You should see the default Django page:

django

[View release notes](#) for Django 2.2



The install worked successfully! Congratulations!

You are seeing this page because `DEBUG=True` is in your settings file and you have not configured any URLs.

**Django Documentation**

Topics, references, & how-to's

**Tutorial: A Polling App**

Get started with Django

**Django Community**

Connect, get help, or contribute

3. Install django-allauth

The `django-allauth` package will handle authentication that has to do with user's social account access.

Quit your server with CTRL+C in the terminal to continue next step.

On terminal, use this command to install:

```
pip install django-allauth
```

4. Configure settings.py

Next, you'll need to configure `settings.py` file in the `my_project` folder.

- First, update the `INSTALLED_APPS` section:

```
INSTALLED_APPS = [  
    'django.contrib.admin',  
    'django.contrib.auth',  
    'django.contrib.contenttypes',  
    'django.contrib.sessions',  
    'django.contrib.messages',  
    'django.contrib.staticfiles',  
    'django.contrib.sites',    # <--  
    'social_app',    # <--  
  
    'allauth',    # <--  
    'allauth.account',    # <--  
    'allauth.socialaccount',    # <--  
    'allauth.socialaccount.providers.google',    # <--  
]
```

- At the bottom of `settings.py` we need to specify the `allauth` backend:

```
AUTHENTICATION_BACKENDS = (  
    'django.contrib.auth.backends.ModelBackend',  
    'allauth.account.auth_backends.AuthenticationBackend',  
)
```

- Continue to add `site_id` and specify redirect URL upon successful Google login:

```
SITE_ID = 1
```

```
LOGIN_REDIRECT_URL = '/'
```

- Finally, enable email scope to receive user's email addresses after successful social login:

```
SOCIALACCOUNT_PROVIDERS = {  
    'google': {  
        'SCOPE': [  
            'profile',  
            'email',  
        ],  
        'AUTH_PARAMS': {  
            'access_type': 'online',  
        }  
    }  
}
```

5. Create templates folder

You'll need to display a homepage with a link that shows "Sign in with Google".

- Set up the Django template folders following the tree level below. You'll also need to create a new html file `index.html`:

```
— social_app
  — __pycache__
  — migrations
  — templates # <-- New Folder
    — social_app # <-- New Folder
      — index.html # <-- New File
```

- Edit `index.html` file by adding necessary title, links and tags:

```
{% load socialaccount %}
<html>
  <head>
    <title>Google Registration</title>
  </head>
  <body>
    <h1>My Google Login Project</h1>
    {% if user.is_authenticated %}
    <p>Welcome, {{ user.username }} !</p>

    {% else %}
    <h1>My Google Login Project</h1>
    <a href="{% provider_login_url 'google' %}">Login with Google</a>
```



```
{% endif %}  
  
</body>  
</html>
```

6. Configure urls.py

Go to `urls.py` file of your `my_project` directory.

- Add the allauth urls and specify “include” on top of import:

```
from django.contrib import admin  
from django.urls import path, include # <--  
  
urlpatterns = [  
    path('admin/', admin.site.urls),  
    path('accounts/', include('allauth.urls')), # <--  
]
```

- Configure Django templates and link to the urls:

```
from django.contrib import admin  
from django.urls import path, include  
from django.views.generic import TemplateView # <--
```

```
urlpatterns = [  
    path('',  
    TemplateView.as_view(template_name="social_app/index.html")), # <--  
    path('admin/', admin.site.urls),  
    path('accounts/', include('allauth.urls')),  
]
```

- Now, make all the necessary migrations for the changes you've made:

```
(my_project) $ python manage.py makemigrations  
(my_project) $ python manage.py migrate
```

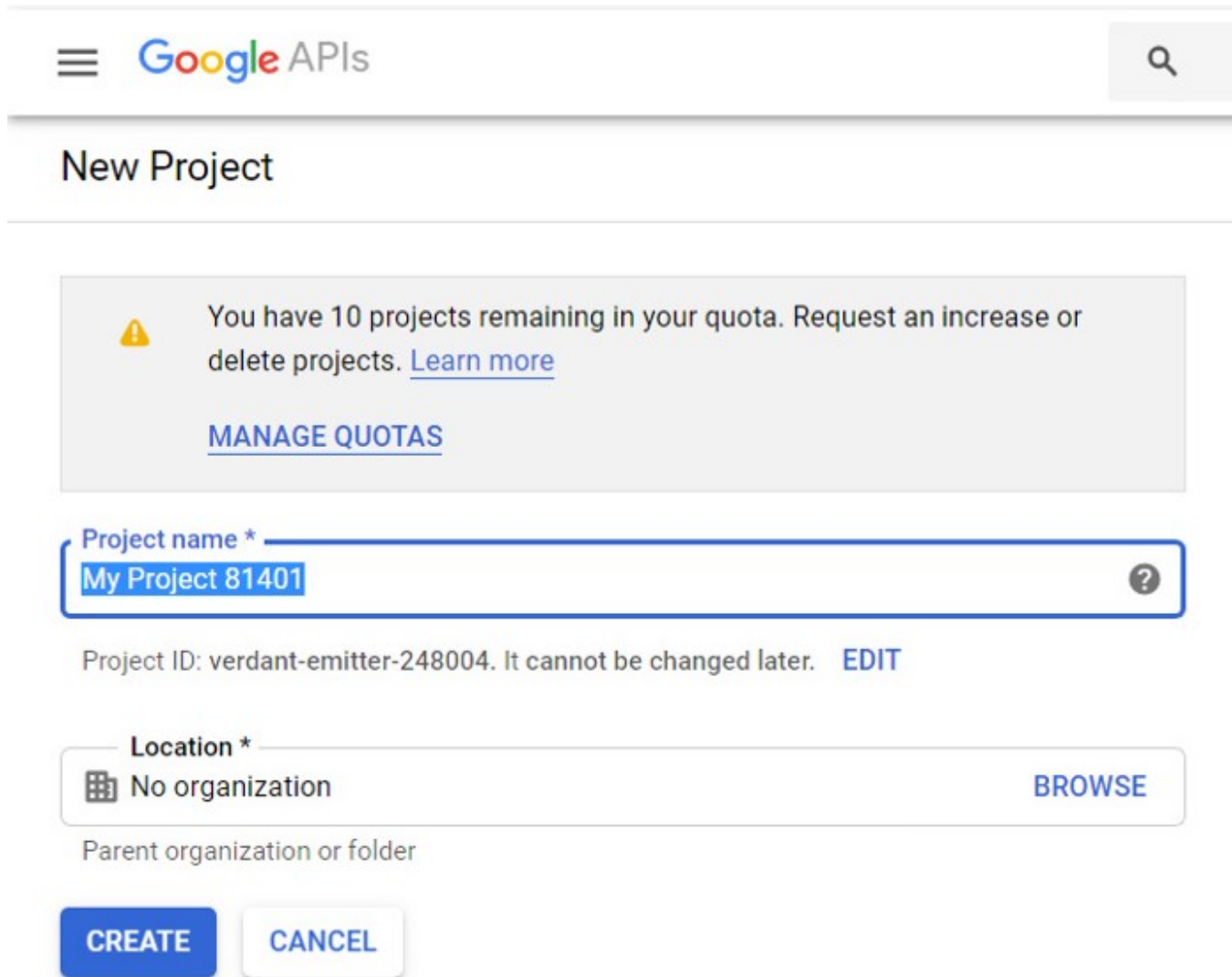
7. Google API console

To add Google login on your app, you'll need to set up OAuth application via [Google Developers Console](#)

Getting started

- Go to Dashboard, create a NEW PROJECT
- Name your new project, preferably your website or app name. User will be able to see this project name when we redirect them to Google login page.

- Click “CREATE” to proceed.



The screenshot shows the 'New Project' page in the Google APIs console. At the top, there's a header with the Google APIs logo and a search icon. Below the header, the title 'New Project' is displayed. A warning box indicates that the user has 10 projects remaining in their quota and provides a link to 'Learn more' and a 'MANAGE QUOTAS' link. The 'Project name' field is highlighted with a blue border and contains the text 'My Project 81401'. Below this, the 'Project ID' is shown as 'verdant-emitter-248004' with a note that it cannot be changed later and an 'EDIT' link. The 'Location' field is also highlighted with a blue border and contains the text 'No organization'. To the right of the 'Location' field is a 'BROWSE' button. Below the 'Location' field, the text 'Parent organization or folder' is visible. At the bottom, there are two buttons: 'CREATE' (blue) and 'CANCEL' (white).

Google APIs

New Project

You have 10 projects remaining in your quota. Request an increase or delete projects. [Learn more](#)

[MANAGE QUOTAS](#)

Project name *
My Project 81401

Project ID: verdant-emitter-248004. It cannot be changed later. [EDIT](#)

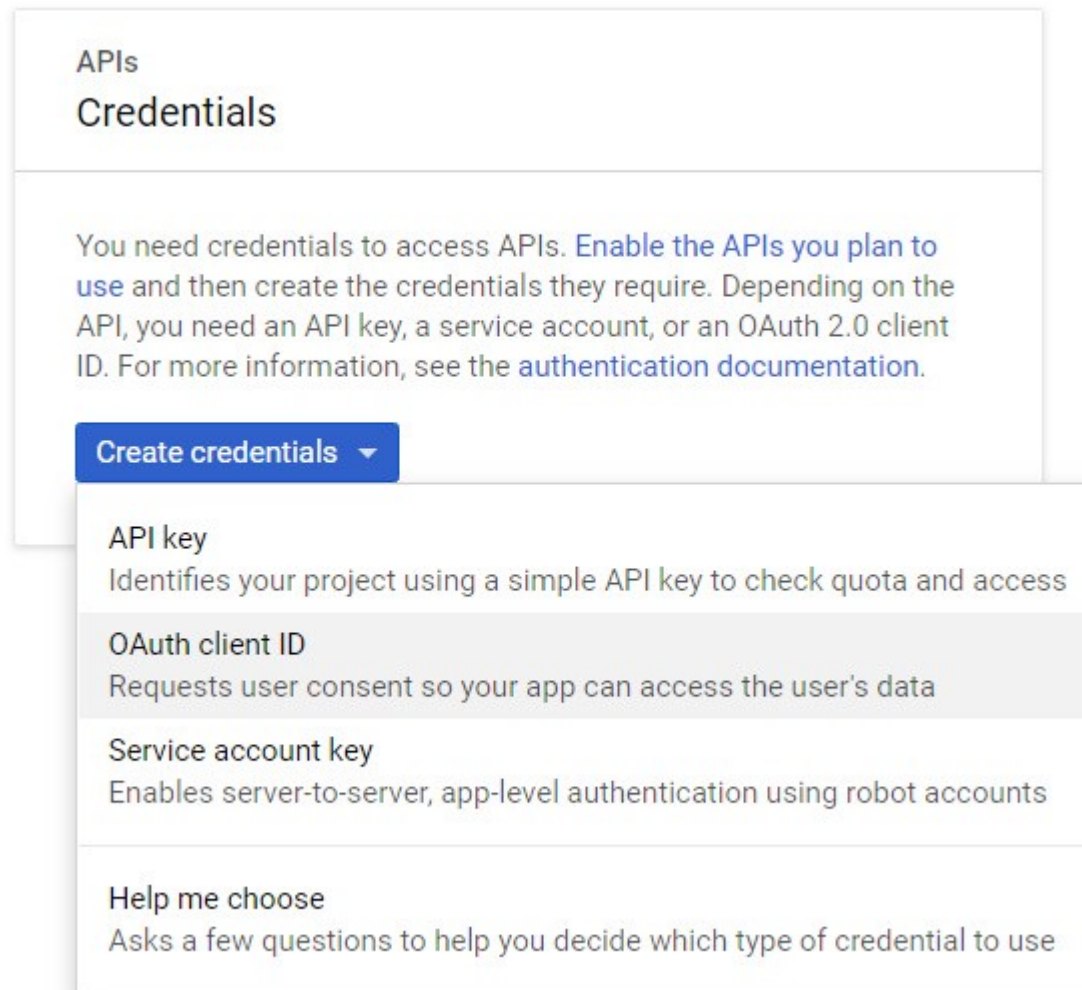
Location *
No organization [BROWSE](#)

Parent organization or folder

[CREATE](#) [CANCEL](#)

APIs Credentials

- Back to “Dashboard”, go to “Credentials” on left panel.
- Create credentials. On the dropdown, choose “OAuth Client ID” option.



OAuth consent screen

- Make sure you fill out the “OAuth Consent Screen” form.
- You’ll only need to provide “Application name”, “Email” and click “SAVE”.

Credentials

Credentials

OAuth consent screen

Domain verification

Before your users authenticate, this consent screen will allow them to choose whether they want to grant access to their private data, as well as give them a link to your terms of service and privacy policy. This page configures the consent screen for all applications in this project.

Verification status

Not published

Application name ?

The name of the app asking for consent


Application logo ?

An image on the consent screen that will help users recognize your app



Create OAuth client ID

- Now, you can create your OAuth Client ID by filling out these details;
- Authorized Javascript origins (<http://127.0.0.1:8000>)
- Authorized redirect URL
(<http://127.0.0.1:8000/accounts/google/login/callback/>)

 Create OAuth client ID

For applications that use the OAuth 2.0 protocol to call Google APIs, you can use an OAuth 2.0 client ID to generate an access token. The token contains a unique identifier. See [Setting up OAuth 2.0](#) for more information.

Application type


☒ Web application

☐ Android [Learn more](#)

☐ Chrome App [Learn more](#)

☐ iOS [Learn more](#)

☐ Other

Name 

Web client 1


Restrictions

Enter JavaScript origins, redirect URIs, or both [Learn More](#)

Origins and redirect domains must be added to the list of Authorized Domains in the [OAuth consent settings](#).

Authorized JavaScript origins

For use with requests from a browser. This is the origin URI of the client application. It can't contain a wildcard (https://*.example.com) or a path (<https://example.com/subdir>). If you're using a nonstandard port, you must include it in the origin URI.

<http://127.0.0.1:8000> 

<https://www.example.com>

Type in the domain and press Enter to add it

Type in the domain and press Enter to add it

Authorized redirect URIs

For use with requests from a web server. This is the path in your application that users are redirected to after they have authenticated with Google. The path will be appended with the authorization code for access. Must have a protocol. Cannot contain URL fragments or relative paths. Cannot be a public IP address.

`http://127.0.0.1:8000/accounts/google/login/callback/`

Type in the domain and press Enter to add it

Create

Cancel

Obtain OAuth client


- Once you click “CREATE”, you will be able to obtain your “client ID” and “client Secret”.
- You’ll need this information to proceed the next steps

OAuth client

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

i OAuth is limited to 100 [sensitive scope logins](#) until the [OAuth consent screen](#) is published. This may require a verification process that can take several days.

Here is your client ID

`971090730061-io0rr3o99h9u33naaq86telc7u591t4o.apps.googleusercontent.com` 

Here is your client secret

NwWNDtF4AAm9QTQU_0NkBWP7



OK

8. Django admin

We'll need to configure Django admin. To access the admin panel, you'll need to create a superuser to login. Type this command on terminal:

```
(my_project) $ python manage.py createsuperuser
```

You're required to provide "username", "email" and "password" in the terminal. Once you're done, proceed to start the server:

```
(my_project) $ python manage.py runserver
```

Go to (<http://127.0.0.1:8000/admin>) to access your admin page. Make sure you provide the credentials to login.

After successful login, your admin page should look like this:

Django administration

Site administration

ACCOUNTS

Email addresses

+ Add

✎ Change

AUTHENTICATION AND AUTHORIZATION

Groups

+ Add

✎ Change

Users

+ Add

✎ Change

SITES

Sites

+ Add

✎ Change

SOCIAL ACCOUNTS

Social accounts

+ Add

✎ Change

Social application tokens

+ Add

✎ Change

Social applications

+ Add

✎ Change

Add a site

On the SITES section, click “sites” and fill out the details and click “Save”:

- **Domain name:** 127.0.0.1:8000
- **Display name:** 127.0.0.1:8000

Add social applications


Back to admin homepage, under “SOCIAL ACCOUNTS” section, click “Social applications” to fill out these settings:

- **Provider:** Google
- **Name:** Google API
- **Client id:** (refer step 7, your OAuth details)
- **Secret key:** (refer step 7, your OAuth details)

9. See the result 🙌


You’re almost done! Now navigate to (<http://127.0.0.1:8000/>) to see the “Sign in with Google” page.

Try to access the Google login, you should be able to see the redirect screen:


 Sign in with Google

Choose an account


to continue to **Newapp Google Login**




Whizz Zoe
[redacted]@gmail.com



Firebass Music Nation
[redacted]@gmail.com



Zoe Chew
[redacted]@gmail.com



Use another account

To continue, Google will share your name, email address, language preference, and profile picture with Newapp Google Login.

After successfully signed in with Google, you should be able to see your Google account name on the homepage:

My Google Login Project

Welcome, Sarah !

Try to sign in with another Google account via (<http://127.0.0.1:8000/accounts/google/login/>), you'll see the user name changed accordingly:

My Google Login Project

Welcome, Zoe!

10. Where to find me? 🙌

[1] If you love this article & feeling generous, send me a coffee? ☕

[2] Find me on Personal Site / Twitter / LinkedIn 🔥

[3] In case you miss out, I can send my upcoming resources to your inbox —
Click here to join my newsletter ➡

[Python](#)

[Django](#)

[API](#)

[Web Development](#)

[User Experience](#)

[About](#)

[Help](#)

[Legal](#)