# **Trabajo: Configuración de Django con django-allauth y despliegue en Google Cloud**

## **1. Configuración en Django**

### **Archivo settings.py**

En este archivo, se definen las configuraciones esenciales para integrar django-allauth y permitir el inicio de sesión mediante cuentas de Google.

### **Ajustes importantes en settings.py:**

INSTALLED\_APPS = [

#\*\*\*OTRAS APLICACIONES\*\*\*

'django.contrib.sites',

'allauth',

'allauth.account',

'allauth.socialaccount',

'allauth.socialaccount.providers.google',

#\*\*\*OTRAS APLICACIONES\*\*\*

]

AUTHENTICATION\_BACKENDS = [

'django.contrib.auth.backends.ModelBackend',

'allauth.account.auth\_backends.AuthenticationBackend',

]

MIDDLEWARE = [

#\*\*\*OTROS MIDDLEWARE\*\*\*

"allauth.account.middleware.AccountMiddleware",

#\*\*\*OTROS MIDDLEWARE\*\*\*

]

SITE\_ID = 1

### **Explicación de la configuración:**

* **INSTALLED\_APPS:** Aquí se incluyen las aplicaciones necesarias para django-allauth y la integración con proveedores de cuentas sociales como Google.
* **AUTHENTICATION\_BACKENDS:** Define que el sistema de autenticación será manejado por el backend de django-allauth, además del backend tradicional de Django.
* **MIDDLEWARE:** Incluye el middleware necesario para gestionar la sesión de los usuarios que inician sesión con redes sociales.
* **SITE\_ID:** Es necesario para identificar el sitio web dentro de la base de datos de django-allauth. Este valor se debe ajustar según la configuración de tu entorno.

### **Archivo urls.py**

En este archivo, se configurarán las URLs para permitir la navegación en la aplicación, incluyendo la ruta para gestionar las autenticaciones.

### **Configuración de urls.py:**

from django.conf import settings

from django.conf.urls.static import static

urlpatterns = [

#\*\*\*OTRAS RUTAS\*\*\*

# Rutas de autenticación proporcionadas por django-allauth

path('accounts/', include('allauth.urls')),

# Incluir las URLs de allauth para login, logout, registro, etc.

] + static(settings.MEDIA\_URL, document\_root=settings.MEDIA\_ROOT)

# Configuración para archivos estáticos y multimedia

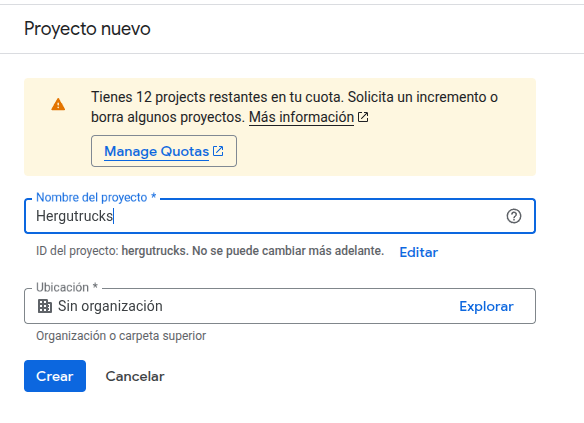
### **Explicación de la configuración:**

* **path('accounts/', include('allauth.urls')):** Esto incluye todas las rutas necesarias para manejar el inicio de sesión, registro, y el inicio de sesión social mediante Google.

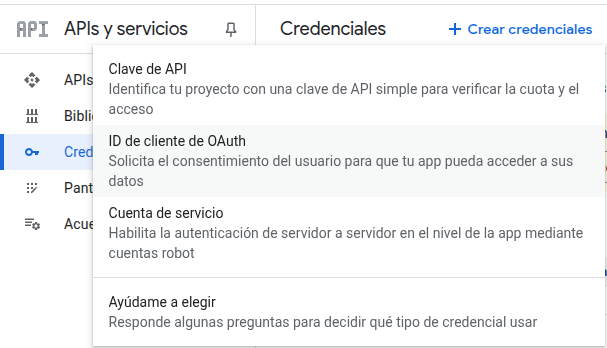
## **2. Configuración en Google Cloud**

Ahora procederemos a crear nuestro proyecto en Google para obtener las claves *client ID* y *secret key*, las cuales utilizaremos más adelante en el archivo *settings.py*. Para ello, utilizaremos el siguiente enlace: [Google Developers Console](https://console.developers.google.com/).

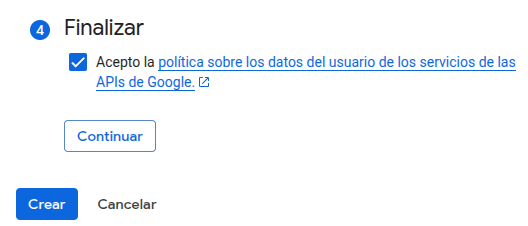
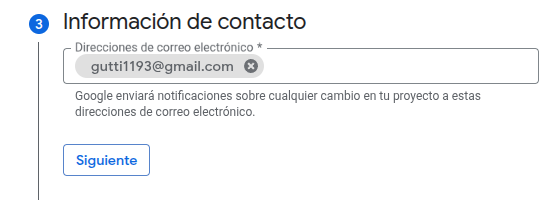
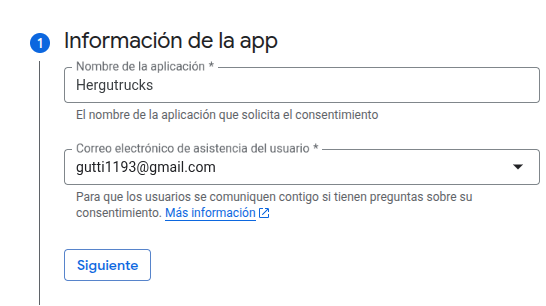
Creo un proyecto nuevo:



Una vez creado me muevo al apartado de credenciales y agregó una nueva credencial. En el desplegable que se muestra, seleccione la opción de ID de cliente 0Auth:



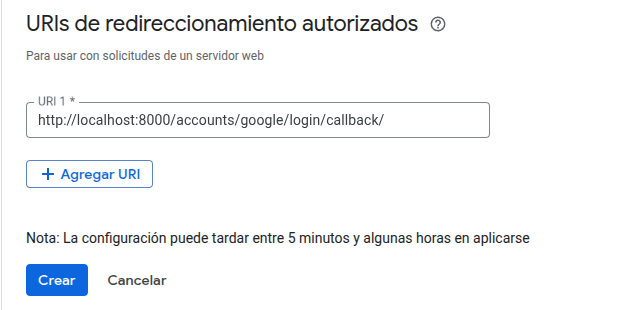
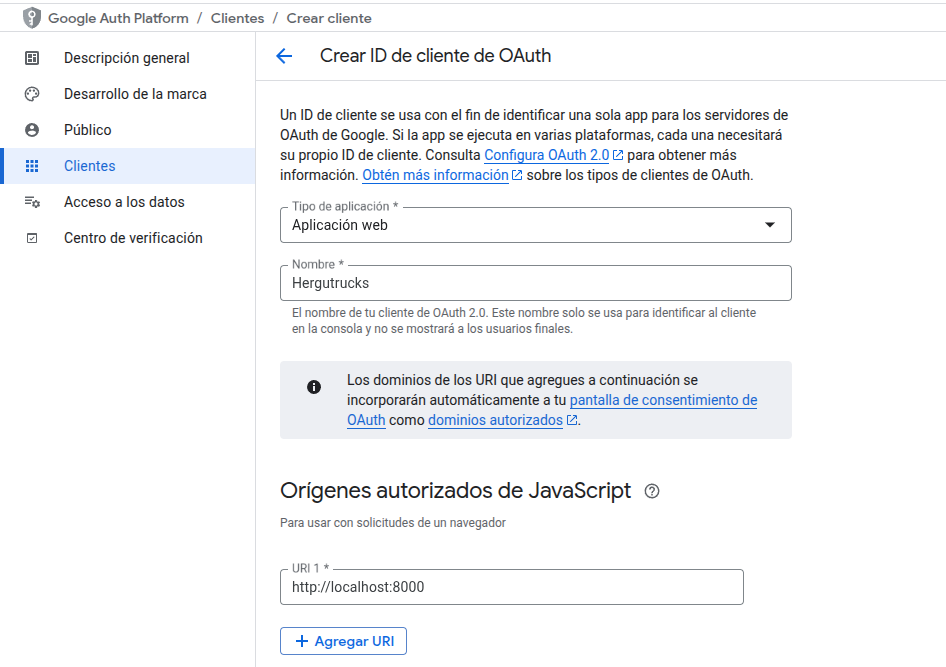
Antes de seguir, nos pedirá que configuremos la pantalla de consentimiento. En el clicamos y le damos a comenzar. Ahora solo debemos rellenar los campos del formulario que nos aparece



Una vez realizados los anteriores pasos, podremos crear el cliente de 0Auth



y ahora rellenamos de nuevo con los datos necesarios:



Le damos a crear y veremos que ya aparecer en nuestro listado personal:



Volvemos a nuestra aplicación y agregamos lo siguiente:

SOCIALACCOUNT\_PROVIDERS = {

'google': {

# For each OAuth based provider, either add a ``SocialApp``

# (``socialaccount`` app) containing the required client

# credentials, or list them here:

'APP': {

'client\_id': '302447481722-hd1cgjkfq40p47dqoofir6if6ld1iu8t.apps.googleusercontent.com',

'secret': 'GOCSPX-vk\_jH4ssdSMUH6KfTDn7hgeOd\_Nh',

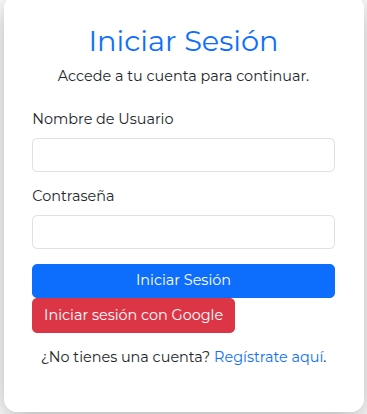
'key': ''

}

}

}

Una vez realizado todo esto, podemos comprobar el resultado en nuestra aplicación:



Al clicar sobre la opción de google, seremos redireccionados y podremos escoger nuestro correo:

