**GMAIL CLEAN UP AUTOMATION USING PYTHON - SOP\_USPJ001**

# **Project Overview**

This Python project is built using the Gmail API, allowing users to automatically clean up unwanted promotional, social, and subscription emails from specific senders or by keyword. It moves them to trash and immediately empties it — all without storing any credentials, ensuring privacy and security.

# **Gmail MANUAL CLEANUP**

To Search Filters to Find Large Emails and moved to a label, and delete 1-50 mails alone in a shot and repeat the process and it takes time, when the size of the mail is large (OLD METHOD NOT EFFICIENT)

**🔍 Filter Examples to Use in the Search Bar:**

|  |  |
| --- | --- |
| **Search Query** | **What It Does** |
| larger:10M | Emails larger than 10 MB |
| larger:5M has:attachment | Emails over 5 MB with attachments |
| older\_than:1y | Emails older than 1 year |
| from:(newsletters@example.com) | Emails from specific senders |

NOTE: created a label "unsubscirbe" there are around 4000 mails in that label unsubscribe

**✅ Step-by-step to delete all emails in the “unsubscribe” label:**

1. **Go to Gmail** on your desktop.
2. In the left sidebar, click on the label **unsubscribe**.

If you don’t see it, click **“More”** to expand the label list.

1. At the top left of the email list, click the checkbox to **select all emails on the page** (usually 50).
2. You’ll now see a message like:

**“Select all 50 conversations in this label”**

Click on the link that says:

**“Select all 4,000 conversations in unsubscribe”**

1. Click the **🗑️ Trash icon** at the top to delete all of them.

**✅ Final Step: Empty the Trash to Free Up Space**

1. In the left sidebar, scroll down and click **Trash**.
2. At the top, click **“Empty Trash now”**.
3. Confirm the deletion.

# **USING PYTHON SCRIPT\_ GOOGLE API python client**

To use **Google APIs** in **Python**, you typically use **Google’s official Python client library** called:

**google-api-python-client**

**✅ Google API**

**Google APIs** are sets of restful web services provided by Google for accessing services like:

* Gmail
* Google Drive
* Google Sheets
* Google Calendar
* YouTube
* Google Maps
* and many more.

They allow external programs (like Python scripts) to interact with Google services **programmatically**.

### **API Required for GMAIL CLEANUP**

**Gmail API**

### **Use Cases**:

* Search and delete specific emails (e.g., Promotions, Unsubscribed).
* Empty trash/spam automatically.
* Apply/remove labels.

### **Permissions (Scopes)**:

* https://www.googleapis.com/auth/gmail.modify (to read and delete emails)
* https://www.googleapis.com/auth/gmail.labels (for label management)

**✅ What Is Needed in Python?**

To use a Google API in Python, you need:

#### **STEP 1: Install Google API Python Client**

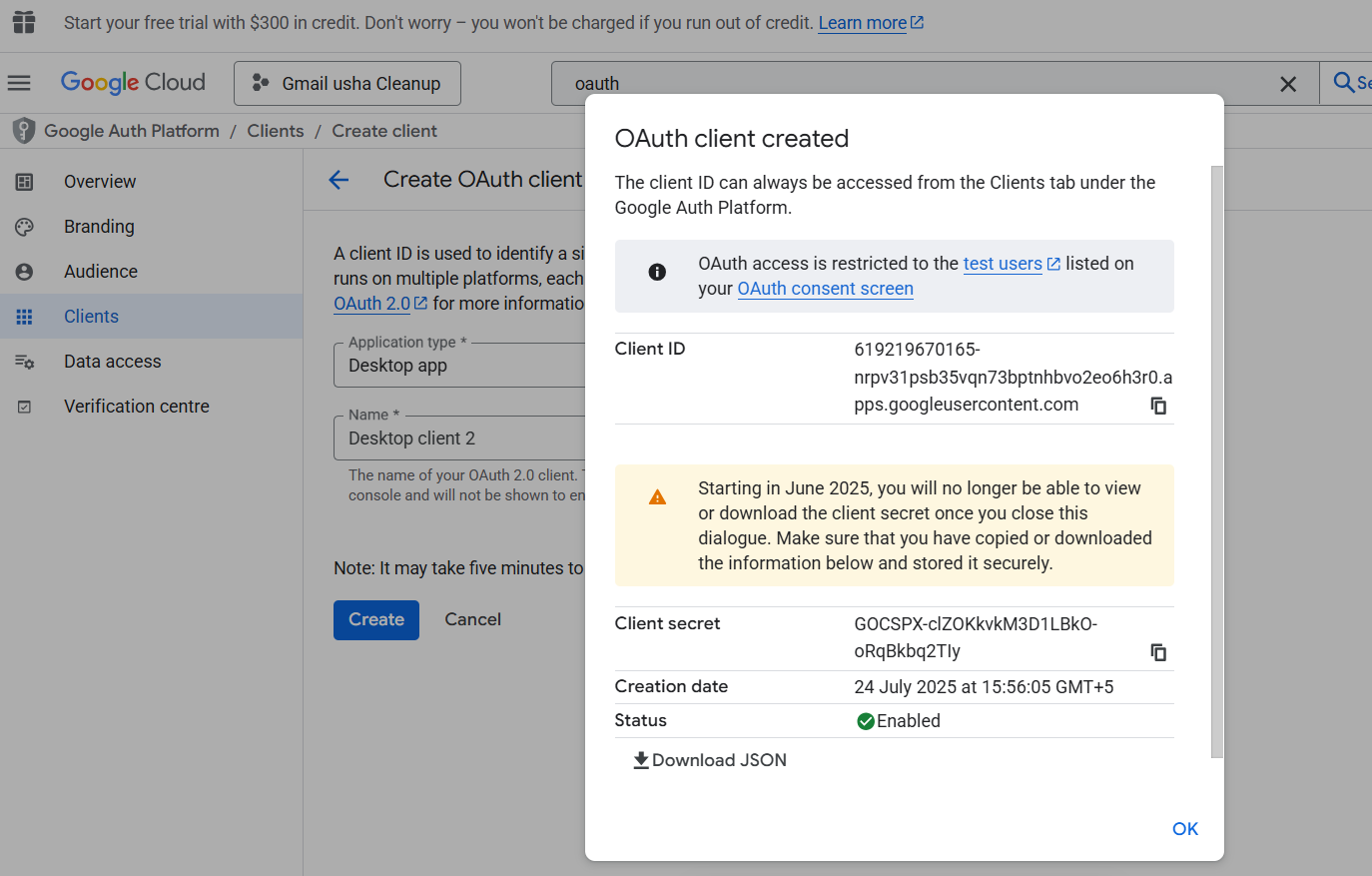
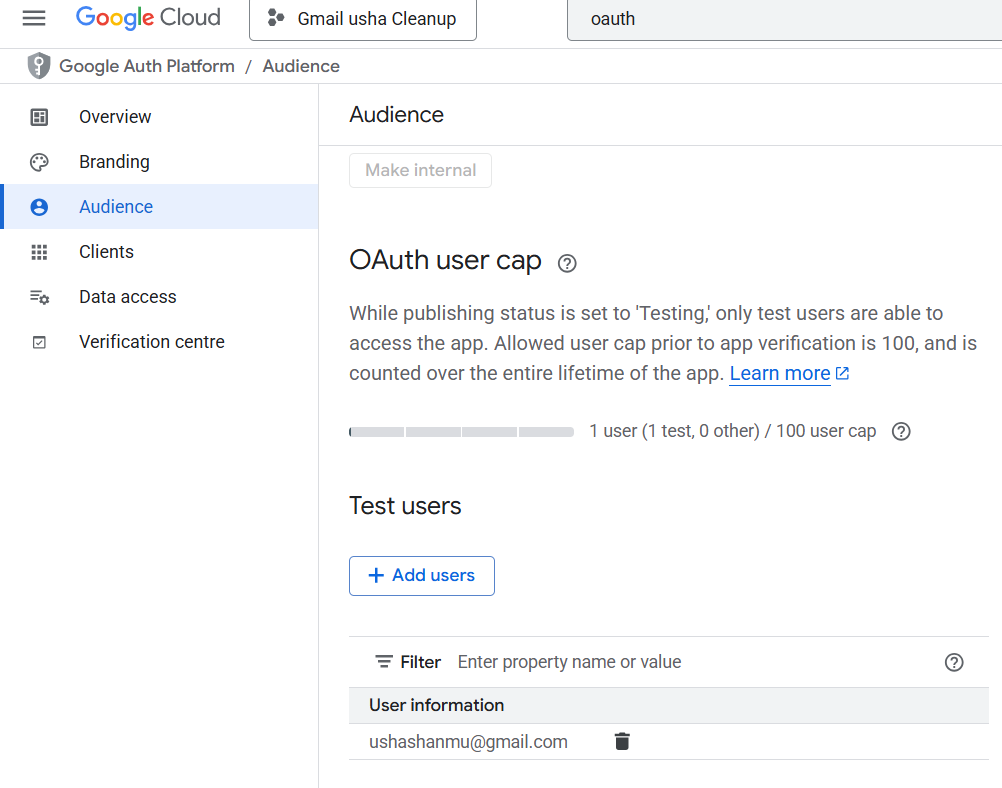
Open terminal or command prompt and run:

**install --upgrade google-api-python-client google-auth-httplib2 google-auth-oauthlib**

These handle authentication (OAuth2), HTTP requests, and interaction with various Google services.

#### **STEP2: Create a Google Cloud Project and Enable Gmail API**

1. Go to Google Cloud Console (<https://console.cloud.google.com/welcome?inv=1&invt=Ab3gFw&project=gmail-cleanup-466516>)
2. Click **"Select a project"** → **"New Project"** → Give it a name → Create
3. Go to Gmail API → Click **Enable**
4. Go to **APIs & Services > OAuth consent screen > Clients**
   * click **Create client --** Choose **"External"**,
   * Add app name (e.g., "Gmail Script"), your email, Download JSON file → rename it to credentials.json, save it in Downloads
   * Click ok
5. Go to **Audience > Test users** 
   * Click Add users – Give your mail id

# **STEP 3: PYTHON SCRIPT (#GMAIL CLEANUP SCRIPT)**

from google\_auth\_oauthlib.flow import InstalledAppFlow

from googleapiclient.discovery import build

SCOPES = ['https://www.googleapis.com/auth/gmail.modify']

def gmail\_auth():

flow = InstalledAppFlow.from\_client\_secrets\_file('credentials.json', SCOPES) *#Loads OAuth 2.0 client configuration from credentials.json. Prepares the flow to initiate user authorization using the SCOPES (permissions, e.g., Gmail read/write)*

return build('gmail', 'v1', credentials=flow.run\_local\_server(port=0))

def clean\_gmail(service, queries): *#Defines a function named clean\_gmail that takes two arguments:1. service: the authenticated Gmail API object.; 2. queries: a list of Gmail search terms (like "unsubscribe" or "from:@amazon.in").*

for query in queries*: Loops through each search query (e.g., "unsubscribe") from the queries list.*

messages = service.users().messages().list(userId='me', q=query).execute().get('messages', []) *#Calls the Gmail API to search for messages matching the current query. userId='me': means it refers to the currently logged-in user. q=query: uses Gmail's powerful search operators to filter emails (like Gmail’s web search). execute() actually sends the request and gets the response and retrieves the list of messages from the API response. If no messages are found, it returns an empty list ([]).*

print(f"Cleaning: '{query}' → {len(messages)} found") *Shows on the screen how many messages matched this query. Example: Cleaning: 'unsubscribe' → 40 found.*

for msg in messages:

service.users().messages().trash(userId='me', id=msg['id']).execute() *msg['id'] gives the unique ID of that email. The Gmail API moves it to Trash using trash(...). .execute() actually performs the action.*

print(f"✅ All emails for query '{query}' have been moved to Trash.\n") *# Confirms success of that specific query*

print("✅ All query-based emails have been successfully moved to Trash.\n") #*Confirms everything completed* ***before*** *moving to empty\_trash(service)*

def empty\_trash(service): *# Defines a function empty\_trash that uses the Gmail API to permanently delete all emails in Trash.*

service.users().messages().emptyTrash(userId='me').execute() *# Gmail API to* ***empty the entire Trash folder***.

print("Trash emptied. No Credentials Stored.") *# Confirms to the user that the trash has been emptied*

if \_\_name\_\_ == "\_\_main\_\_": ***# only run the following code if the script is executed directly, not if it's imported from another script***

queries = [

"unsubscribe", "opt out", "from:@zomato.com", "from:@amazon.in", "from:@myntra.com",

"from:@zepto.app", "from:@swiggy.in", "from:@linkedin.com", "from:@facebookmail.com",

"from:@instagram.com", "livecareer", "inflix", "from:@bookmyshow.com", "from:@freepik.com",

"customercare", "curvette", "easykids". “horoscope”

] *# This is a list of* ***search filters****. Each item is like typing* *a search into Gmail (e.g., from:@amazon.in shows all Amazon mails).*

service = gmail\_auth() *# Authenticates Gmail (you wrote this earlier) and return the Gmail API service object.*

clean\_gmail(service, queries) *# Searches & moves matched emails to Trash*

empty\_trash(service) *# calling the Function empty\_trash which will permanently deletes emails in Trash*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*