Project Documentation

Project Title: v i d e o u p l o a d f r o m o n e d r i v e t o y o u t u b e

Source code : https://github.com/SANKETHSB/one_drive_upload

Name : **Sanketh S B**
Email :sanketh.opqtech@gmail.com

Abstract

The **Video Upload from OneDrive to YouTube** project is designed to streamline the process of transferring video files from OneDrive to YouTube. This automation tool utilizes Microsoft Graph API to interact with OneDrive, download video files, and then upload them to YouTube using the YouTube Data API.

The system authenticates users through Microsoft and Google OAuth, ensuring secure access to both platforms. It first scans a specified OneDrive folder for video files, downloads them locally, and then uploads them to YouTube with customizable metadata like title, description, and privacy settings. Additionally, the system includes error handling and logging mechanisms to monitor the process and provide feedback to users.

The project eliminates the need for manual file transfer between platforms, making it ideal for users who frequently upload content to YouTube from OneDrive, such as content creators, educators, and organizations.

Table of Contents

1. Pre-requisites

Before setting up the script, ensure that the following are available:

- **Python 3.x** installed on your machine.

- **An Azure Active Directory (AAD) application** for Microsoft API access.

- **A Google Cloud Platform (GCP) project** with YouTube Data API enabled.

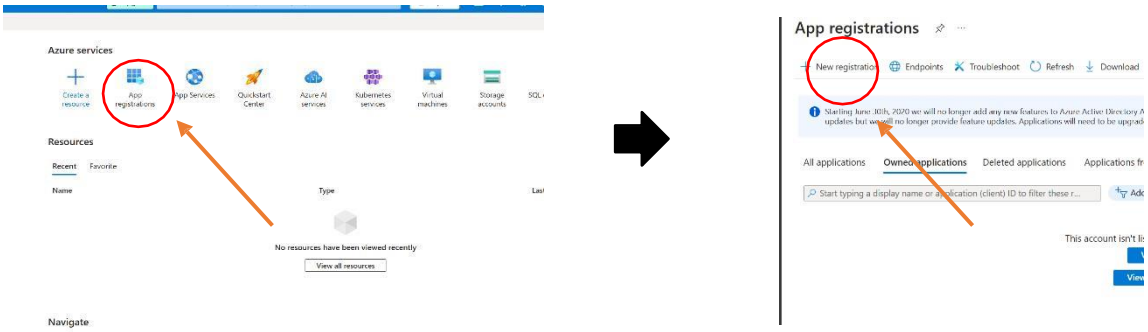
GOOGLE CLOUD CONSOLE

2. Setup Instructions

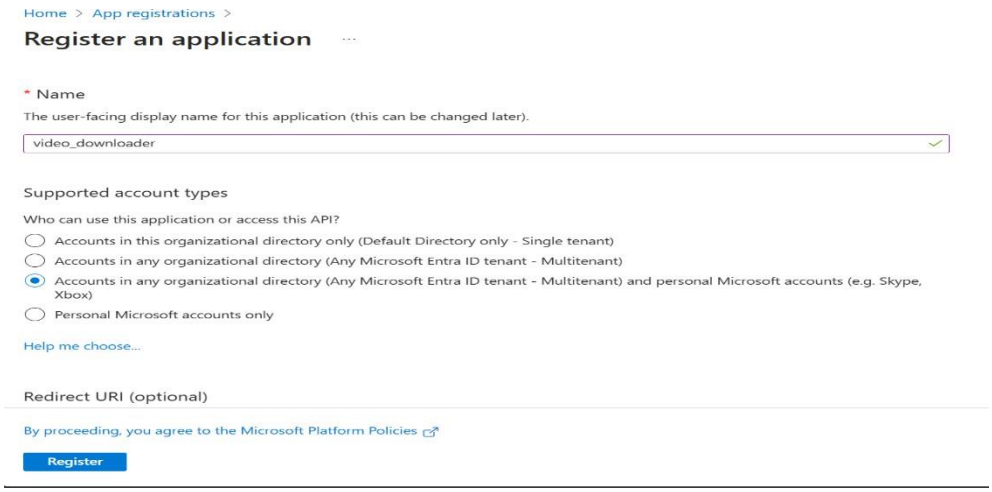
Microsoft API Configuration

To download files from OneDrive, you need to register an application in **Azure Active Directory** (AAD) and configure the Microsoft Graph API.

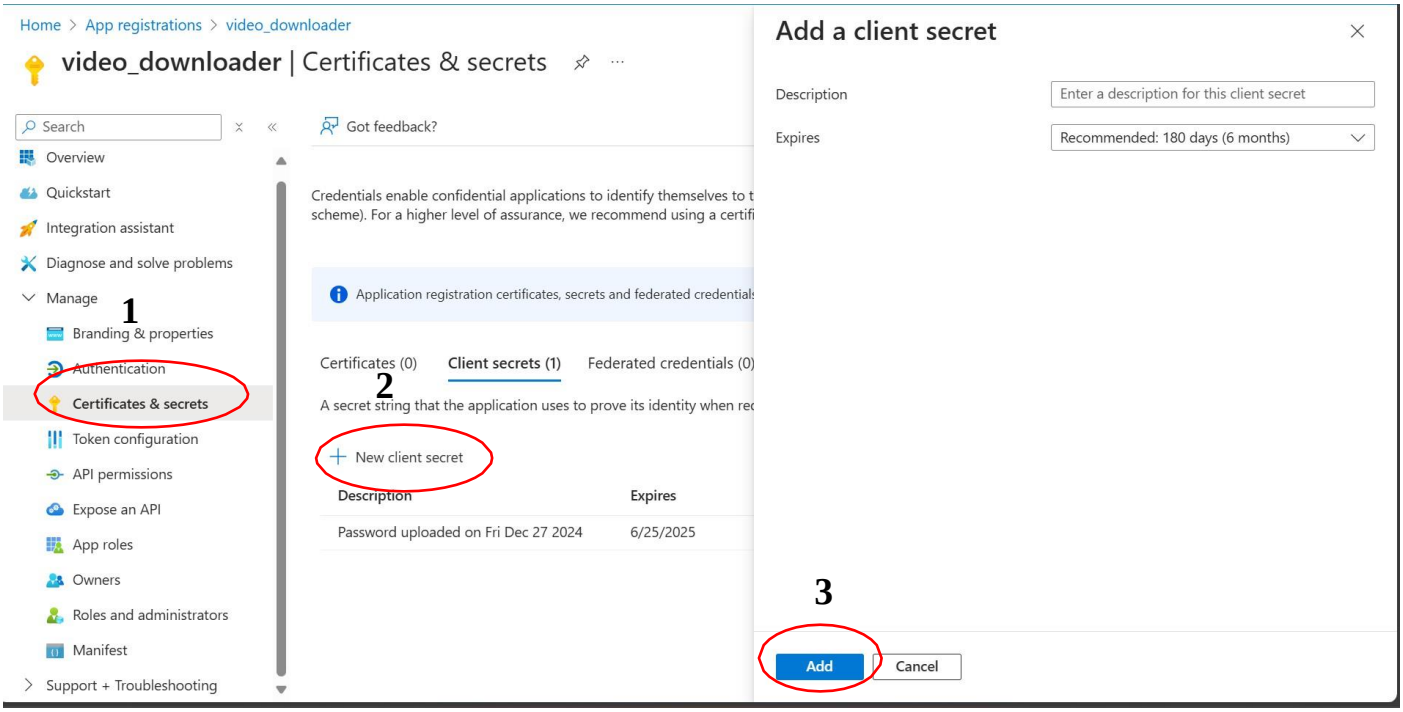
1. **Register your application in Azure Active Directory**:
    - Go to Azure Portal and sign in.
    - Navigate to **Azure Active Directory** > **App registrations** > **New registration**.

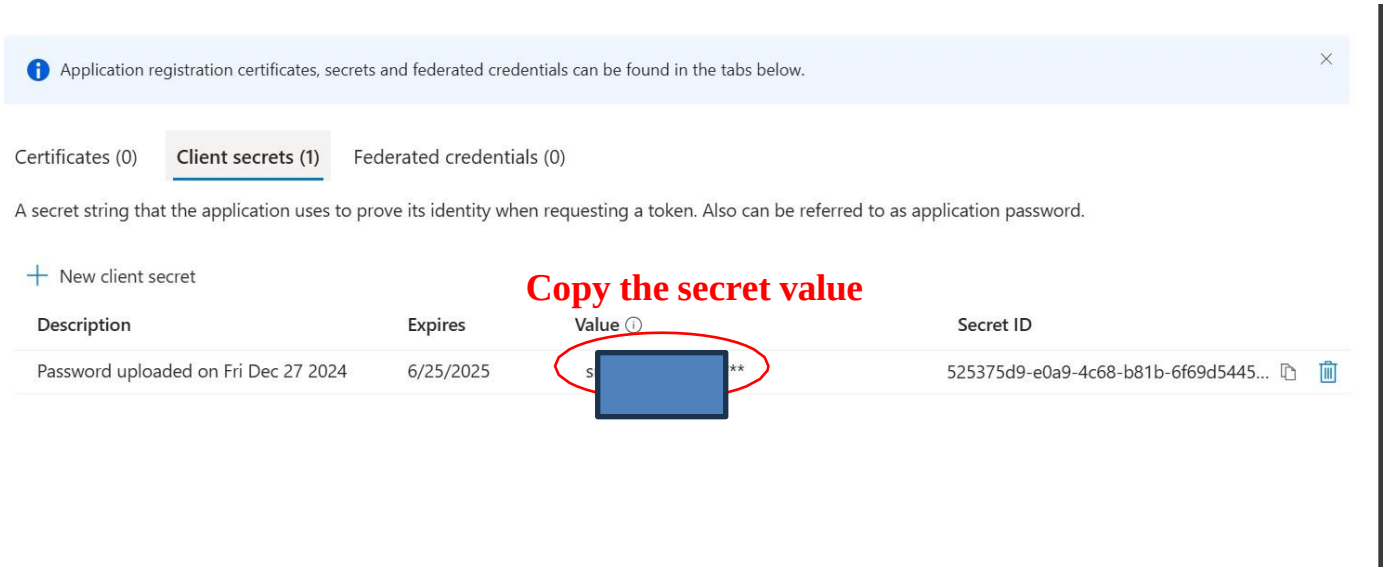


    - Provide an application name (e.g., Video Downloader).
    - Choose **Accounts in any organizational directory and personal Microsoft accounts** for the supported account type.

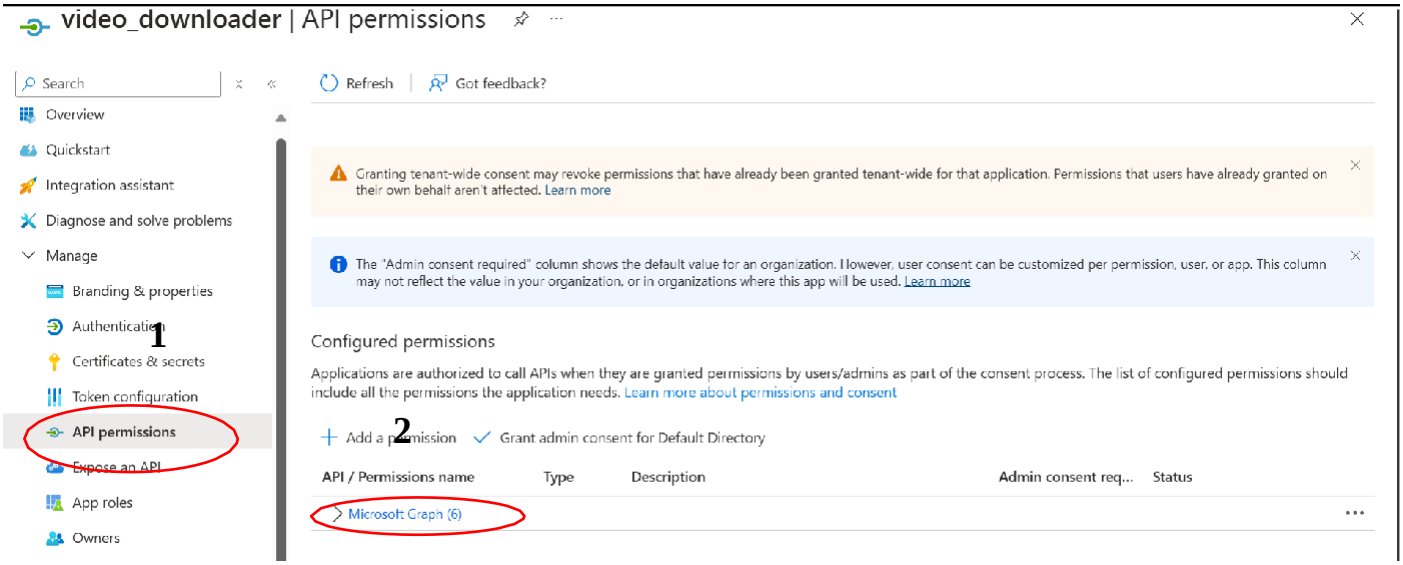


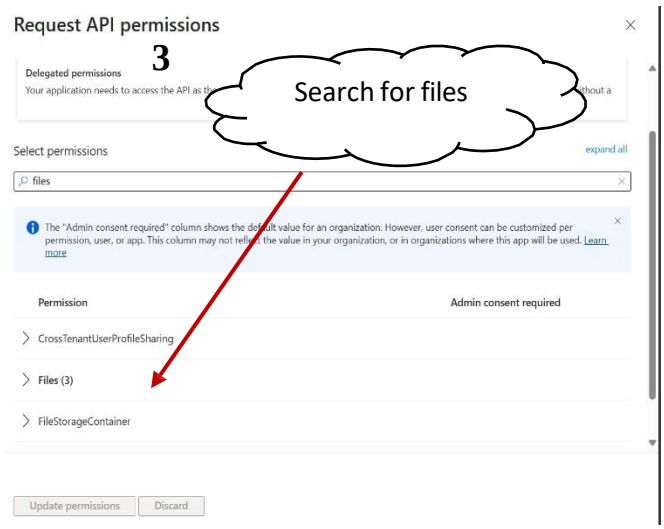
- After registering, go to **Certificates & secrets** > **New client secret**. Save the **client secret** value.

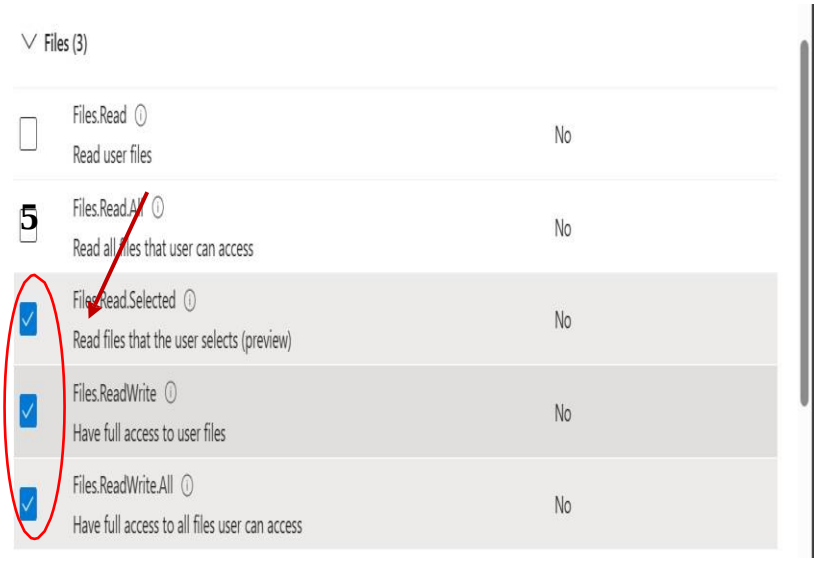


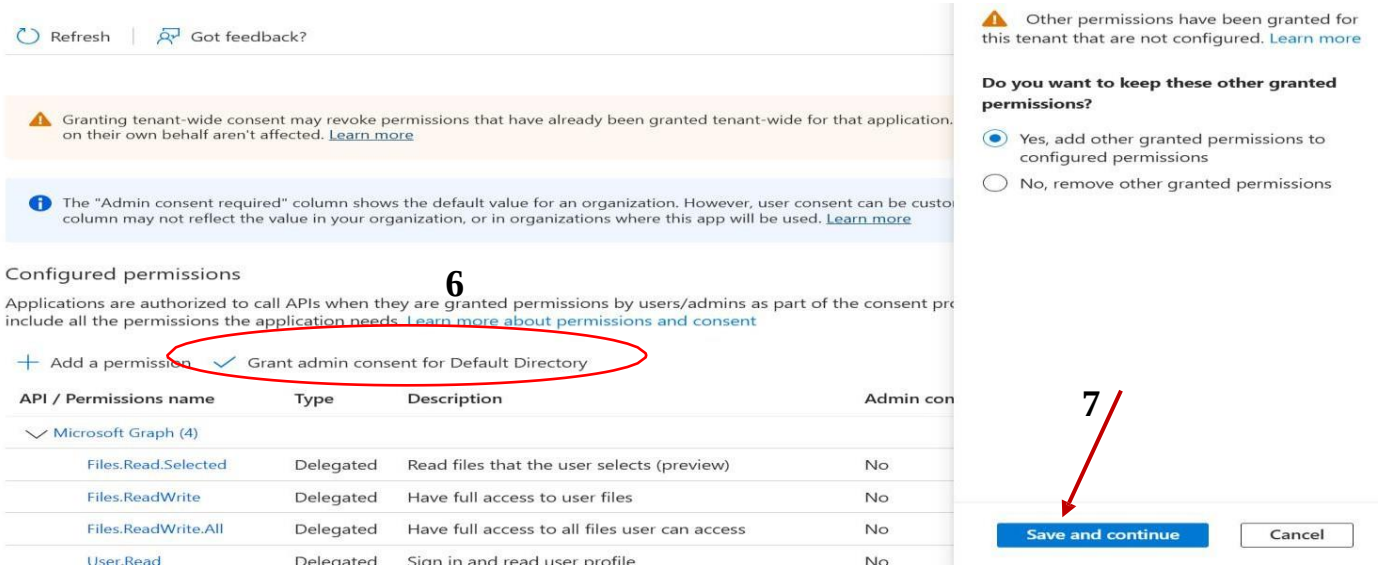


- Copy and update the secret value in the .env file with a name(**CLEINT_SECRET**)
- Under **API Permissions**, add the following Microsoft Graph permissions:
  - **Files.ReadWrite.All**
  - **User.Read**
  - **Files.ReadWrite**
  - **Files.Read**

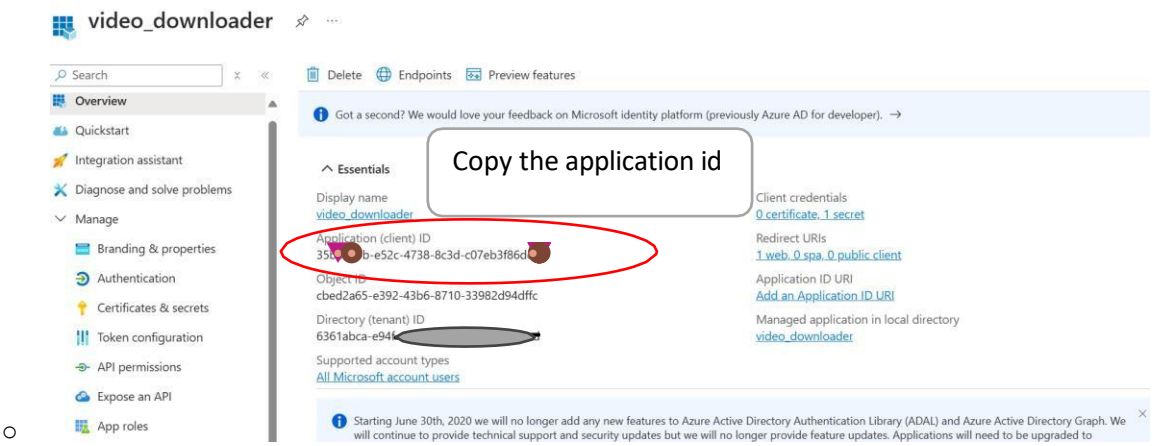

video_downloader | API permissions
Search
Overview
Quickstart
Integration assistant
Diagnose and solve problems
Manage
Branding & properties
Authentication
Certificates & secrets
Token configuration
API permissions
Expose an API
App roles
Owners
1
Refresh
Got feedback?
Granting tenant-wide consent may revoke permissions that have already been granted tenant-wide for that application. Permissions that users have already granted on their own behalf aren't affected. Learn more
The "Admin consent required" column shows the default value for an organization. However, user consent can be customized per permission, user, or app. This column may not reflect the value in your organization, or in organizations where this app will be used. Learn more
Configured permissions
Applications are authorized to call APIs when they are granted permissions by users/admins as part of the consent process. The list of configured permissions should include all the permissions the application needs. Learn more about permissions and consent
Add a permission
2
Grant admin consent for Default Directory
API / Permissions name
Type
Description
Admin consent req...
Status
Microsoft Graph (6)


Request API permissions
3
Search for files
Delegated permissions
Select permissions
expand all
files
Permission
Admin consent required
CrossTenantUserProfileSharing
Files (3)
FileStorageContainer
Update permissions
Discard


Files (3)
Files.Read
Read user files
No
Files.Read.All
Read all files that user can access
No
5
Files.Read.Selected
Read files that the user selects (preview)
No
Files.ReadWrite
Have full access to user files
No
Files.ReadWrite.All
Have full access to all files user can access
No


Refresh
Got feedback?
Granting tenant-wide consent may revoke permissions that have already been granted tenant-wide for that application. on their own behalf aren't affected. Learn more
The "Admin consent required" column shows the default value for an organization. However, user consent can be custo column may not reflect the value in your organization, or in organizations where this app will be used. Learn more
Configured permissions
6
Applications are authorized to call APIs when they are granted permissions by users/admins as part of the consent pr include all the permissions the application needs. Learn more about permissions and consent
Add a permission
Grant admin consent for Default Directory
API / Permissions name
Type
Description
Admin con
Microsoft Graph (4)
Files.Read.Selected
Delegated
Read files that the user selects (preview)
No
Files.ReadWrite
Delegated
Have full access to user files
No
Files.ReadWrite.All
Delegated
Have full access to all files user can access
No
User.Read
Delegated
Sign in and read user profile
No
Other permissions have been granted for this tenant that are not configured. Learn more
Do you want to keep these other granted permissions?
Yes, add other granted permissions to configured permissions
No, remove other granted permissions
7
Save and continue
Cancel

- Make a note of the **Application (client) ID** and update in the .env with the name (**APPLICATION_ID)**

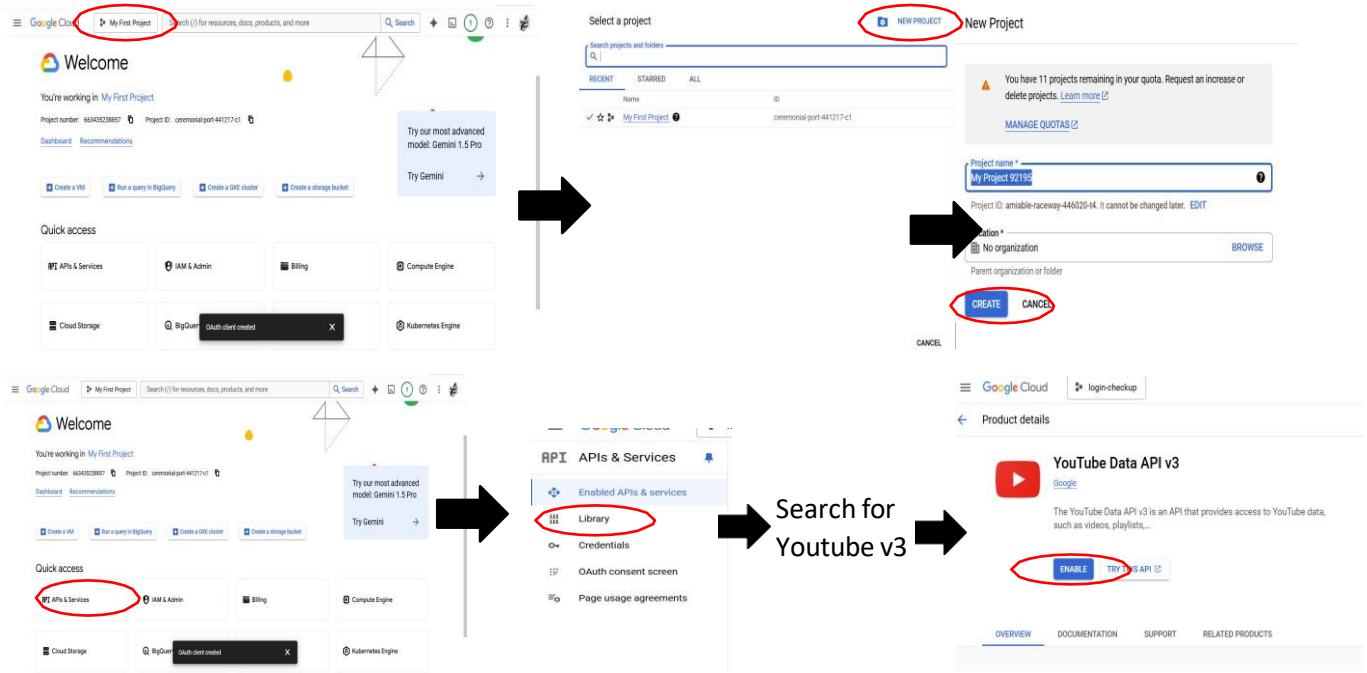


2. **Create the onedrive_token.json file**:
    - This file will store the access token and refresh token. The script will create it after the first authentication.

Google API Configuration

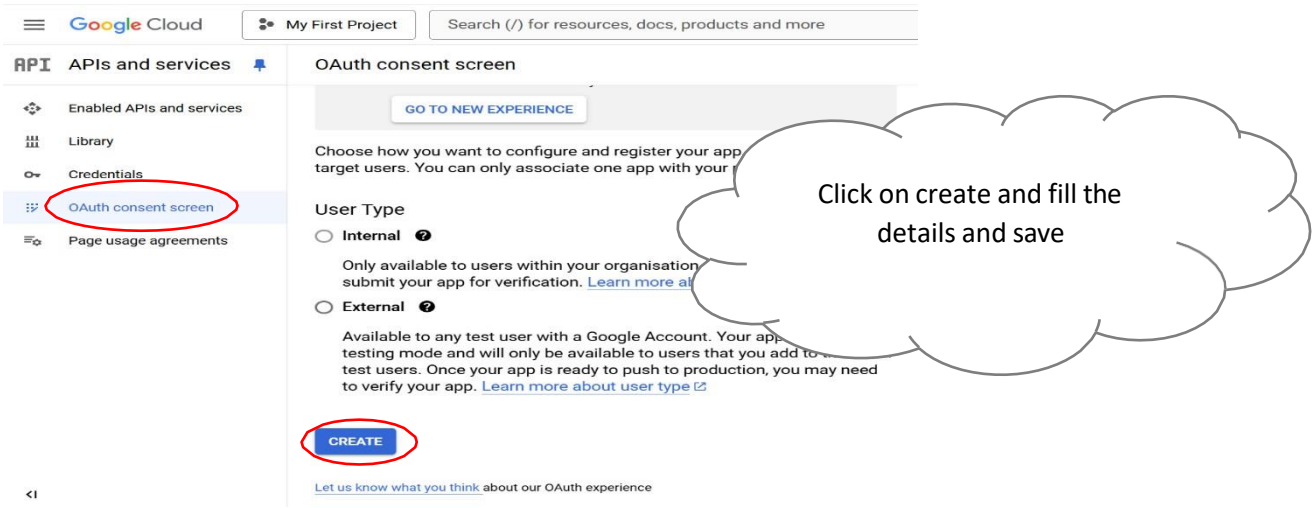
To upload videos to YouTube, you must set up OAuth 2.0 credentials on **Google Cloud Console**.

1. **Create a project in Google Cloud Console**:
    - Go to Google Cloud Console.
    - Create a new project and enable the **YouTube Data API v3** for the project.

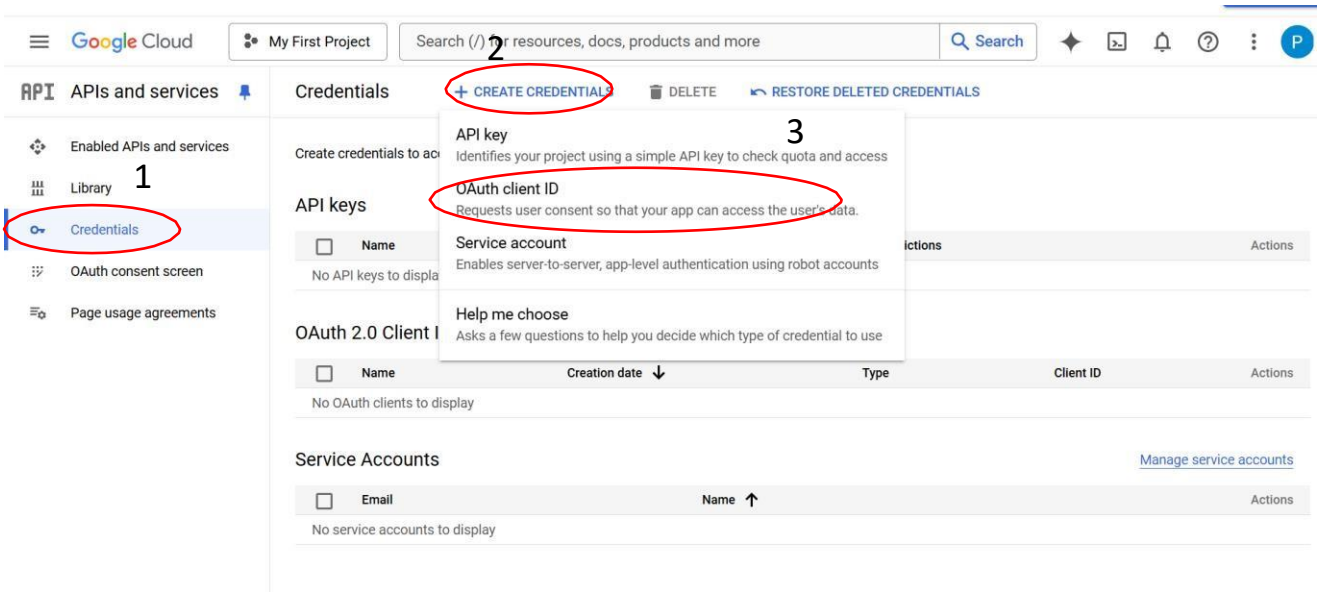


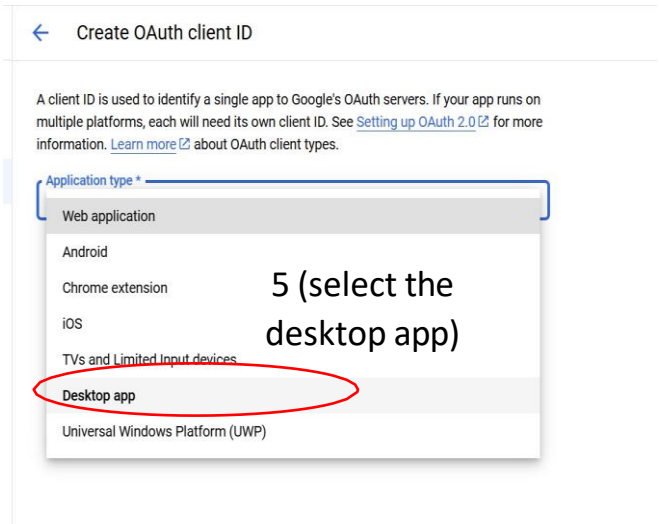
2. Generate OAuth 2.0 credentials:

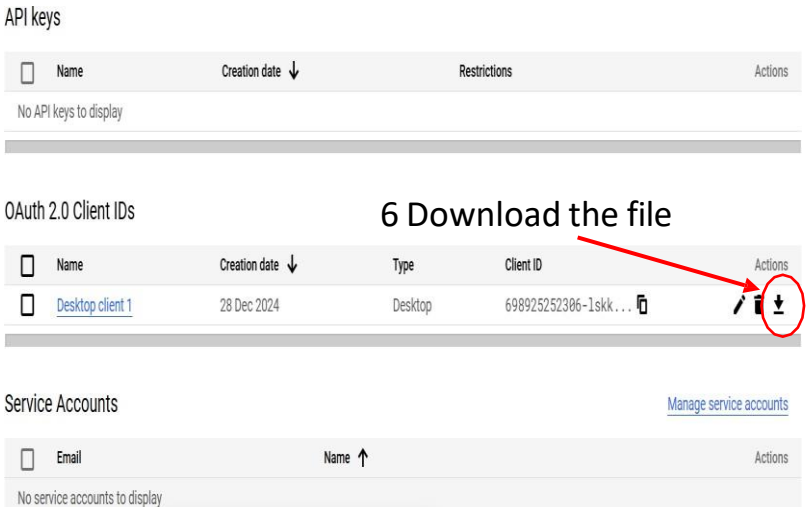
- In the **oAuth consent screen** . click on **External > fill the app name (Video_uploader)>select the user support email and fill the developer contact information (use the same email id)>clikc on save and continue > click on the add or remove scope> search for Scope : https://www.googleapis.com/auth/youtube.upload > click on save and continue> add the test user> Cick on save**



- In the **Credentials** section, click **Create credentials** > **OAuth 2.0 Client IDs**
- Choose **Desktop App** as the application type.







- Download the **client_secrets.json** file (if the file name is downloaded with a another name rename to client_secrets.json) and save it in your project directory.

3. **Generate the youtube_token.json file**:
   - After the first authentication, the script will generate the youtube_token.json file, which will store the access token and refresh token for future API calls.

4. **Environment Setup**
   Create a `.env` file in the project root directory to store sensitive data such as your API keys and secrets.
   Example `.env` file:

```
APPLICATION_ID=your-application-id-here
CLIENT_SECRET=your-client-secret-here
```

3. Installing Dependencies

The script requires several Python packages for functionality. Install them using pip:

pip install msal httpx google-auth google-auth-oauthlib google-api-python-client python-dotenv

4. Script Breakdown

Authentication Functions

- Microsoft Authentication: The script uses the msal library to authenticate with the Microsoft Graph API. The authentication flow follows OAuth2, and the token is stored in the onedrive_token.json file. If the token is expired or absent, the script requests a new token.

Python **def get_access_token(application_id, client_secret, scopes):**

Authentication flow for MS Graph API

- YouTube Authentication: The google-auth and google-auth-oauthlib libraries handle the OAuth2 flow for authenticating with the YouTube API. Tokens are stored in the youtube_token.json file. If the token expires, it will be refreshed.

def authenticate_youtube():

Handles OAuth2 authentication for YouTube

File Download Functionality

The download_file function retrieves files from OneDrive. It sends a request to the Microsoft Graph API and checks for a 302 redirect response. If a file is found, it is downloaded to the specified directory.

def download_file(headers, file_id, file_name):

Download video file from OneDrive

YouTube Upload Functionality

After downloading the videos from OneDrive, the script uploads them to YouTube using the upload_video function. It uses the YouTube Data API v3 and uploads the file with a title, description, and category.

def upload_video(youtube, video_file, title, description, category="22", privacy="private"):

 # Upload video to YouTube

Logging and Error Handling

The script includes logging of actions (such as video uploads and file downloads) with timestamps for each step. The logs are written to the console.

import time

def log_action(message):

timestamp = time.strftime('%Y-%m-%d %H:%M:%S', time.localtime())

print(f"[{timestamp}] {message}")

5. Running the Script

After setting up the dependencies and configurations, you can run the script using the following command:

python app.py

The script performs the following steps:

1. **Authenticates with Microsoft Graph API to fetch files from OneDrive.**
2. **Downloads video files from OneDrive to the downloads folder.**
3. **Uploads videos to YouTube.**
4. **Cleans up the downloads folder after uploading.**

6. Common Issues and Troubleshooting

Microsoft Graph Authentication Fails

- Ensure that the Application ID and Client Secret in the .env file are correct.
- Make sure that the required permissions (Files.ReadWrite.All, User.Read) are granted in Azure AD.

YouTube Upload Fails

- Ensure that the OAuth2 credentials (client secrets) are correctly configured.
- Make sure that the YouTube Data API v3 is enabled in Google Cloud Console.

Missing downloads Folder

- If the downloads folder does not exist, the script will automatically create it, but ensure that the script has proper permissions to write files.

7. Additional Notes

- Token Expiry: The script handles token expiry by checking if the access token is valid. If expired, it will automatically refresh the token.
- Error Handling: The script will print error messages if any part of the process fails (e.g., API call failures, file not found).
- Logs: Every action (e.g., file download, video upload) is logged with a timestamp to help track the progress.

Conclusion

This script provides a comprehensive solution for automating the process of downloading video files from OneDrive and uploading them to YouTube. It leverages the Microsoft Graph API for secure file access from OneDrive and the YouTube Data API for video uploads, ensuring seamless integration between the two platforms. With built-in authentication flows for both APIs, token management, and error handling, the script minimizes manual intervention.

By following the detailed setup instructions and ensuring proper API configurations and dependencies, you can easily configure and run the script. It also logs every action with timestamps, making it easier to track progress and debug issues if they arise.

This solution can be particularly useful for users who need to manage and upload content from OneDrive to YouTube regularly, automating a tedious process and ensuring both security and efficiency throughout.

You can access the full source code and instructions for setting up this script on GitHub:

GitHub Repository - https://github.com/SANKETHSB/one_drive_upload

Or

GitHub link - https://github.com/SANKETHSB/one_drive_upload