

How to Create a Local YouTube Front-End, Quick & Dirty

written by Traveler

You need:

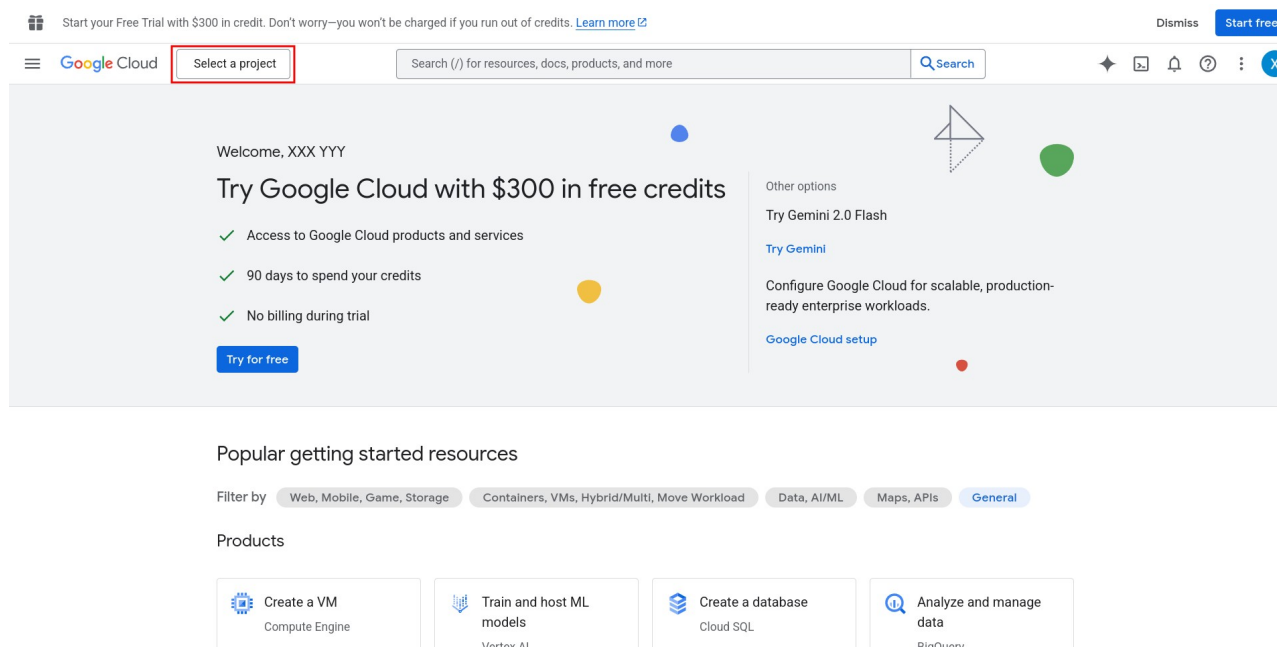
- Gmail account
- API key for YouTube Data API v3
- Python
- Linux (though it could, in theory, work with Windows too)
- Cloned the repository from <https://github.com/EdgeOfAssembly/LocalTube> into LocalTube directory somewhere (I presume you have already done it, otherwise you would not be reading this. Duh!)

First, I assume you already have a Gmail account created.

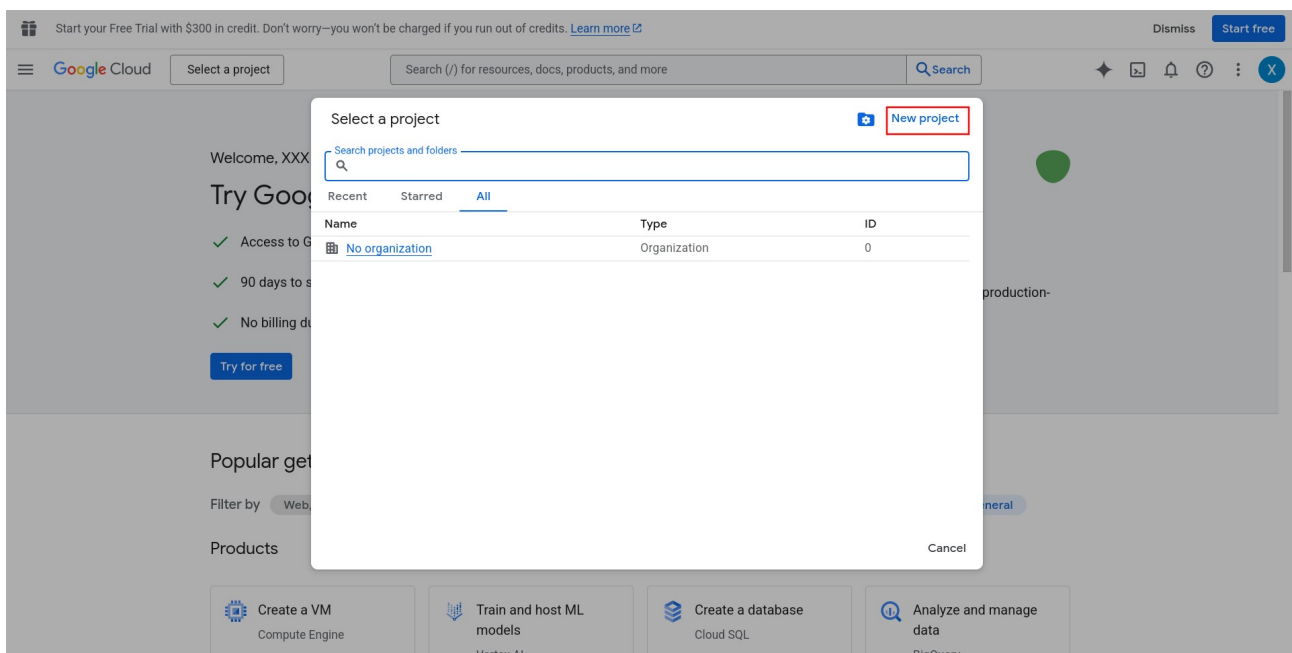
The next step is to create a project (because we can't have an API key without a project).

Go to: console.cloud.google.com and log in with your Gmail account email and password.

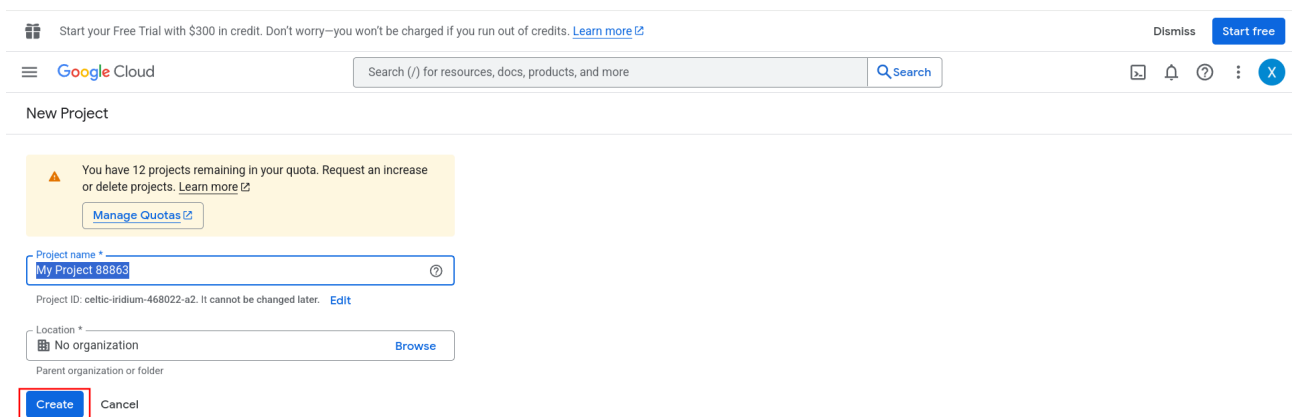
You should see something like this after you agree to the Terms of Service. Click "Select a project."



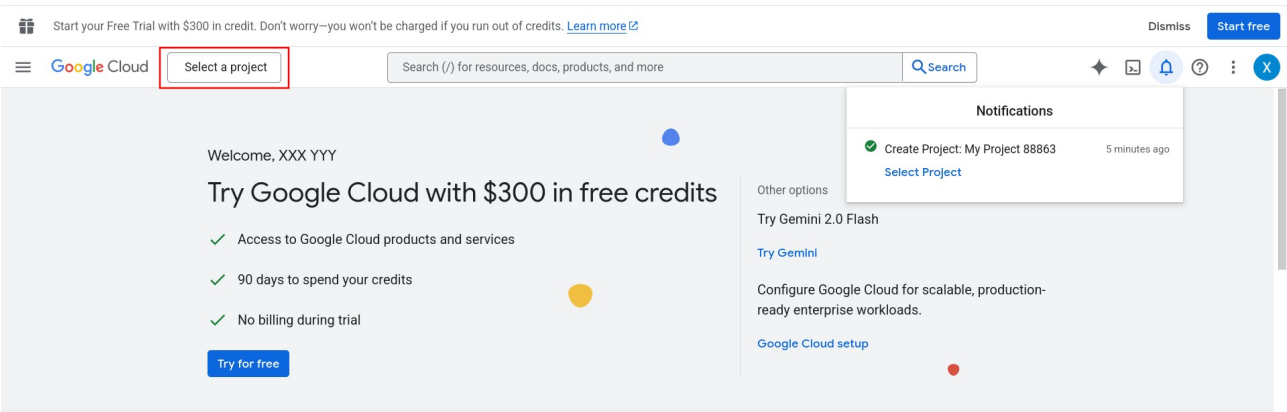
Click "New project."



Use the given project name or enter anything you want, then click "Create."




Click "Select a Project" again.





Popular getting started resources


Filter by [Web, Mobile, Game, Storage](#) [Containers, VMs, Hybrid/Multi, Move Workload](#) [Data, AI/ML](#) [Maps, APIs](#) [General](#)

Products

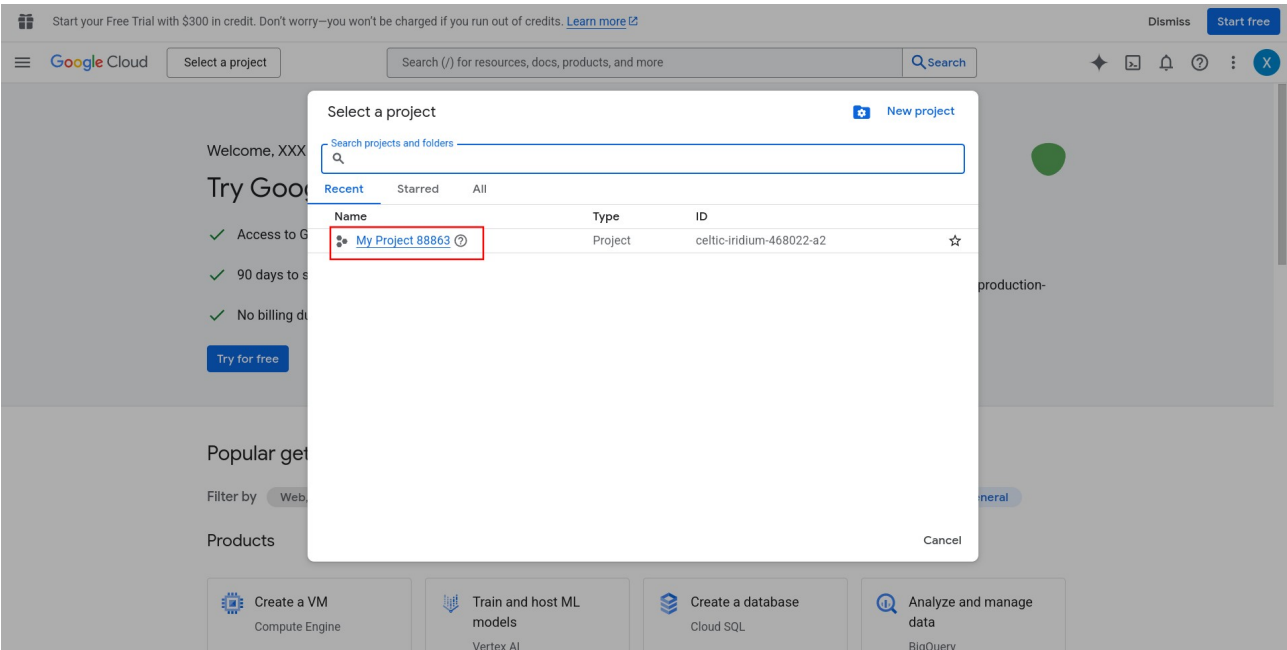
 **Create a VM**
Compute Engine

 **Train and host ML models**
Vertex AI

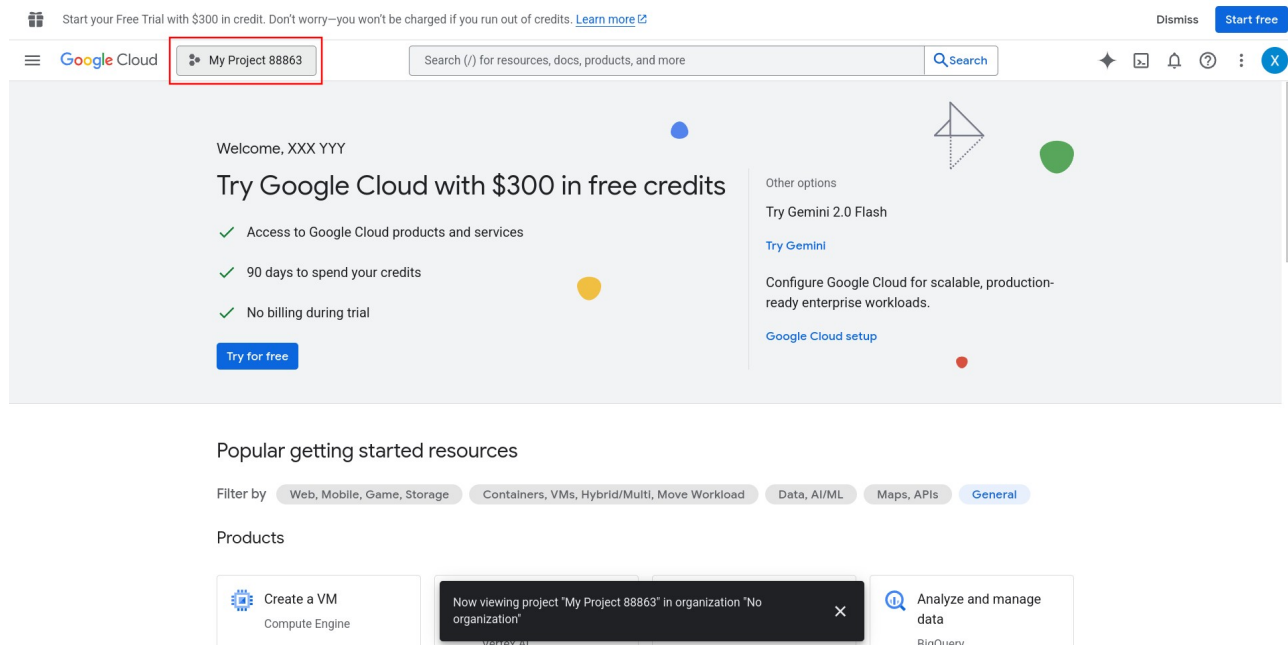
 **Create a database**
Cloud SQL

 **Analyze and manage data**
BigQuery

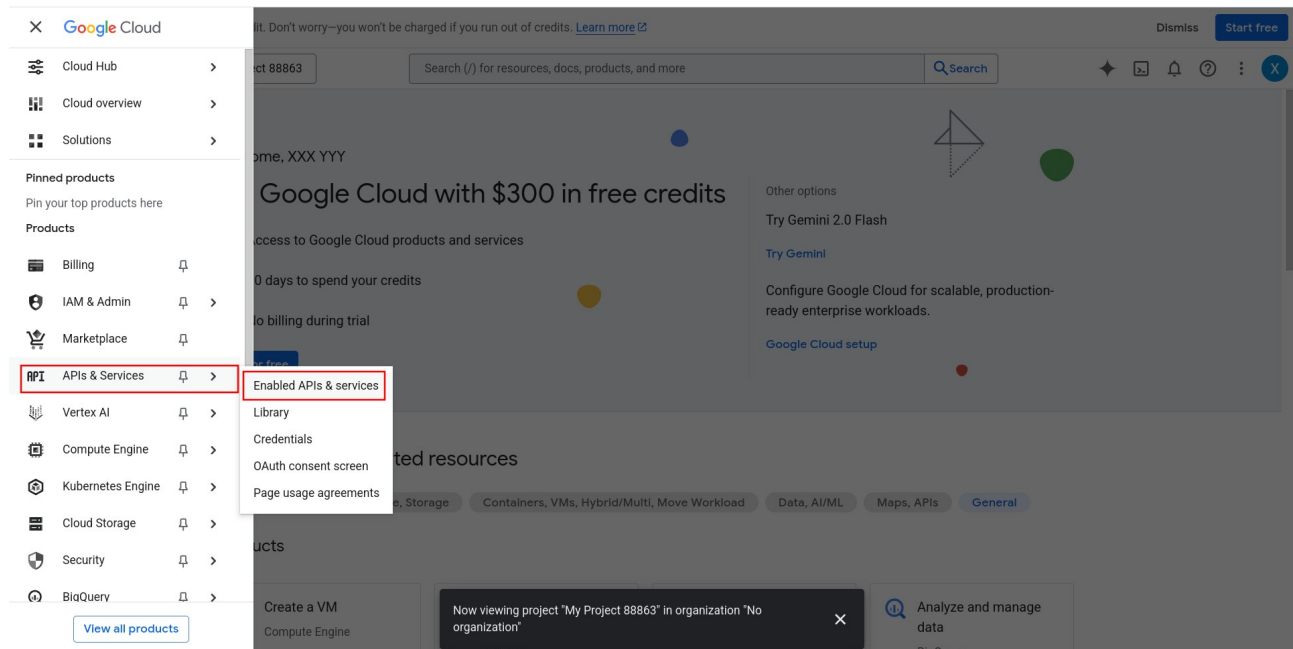
Click your project name.



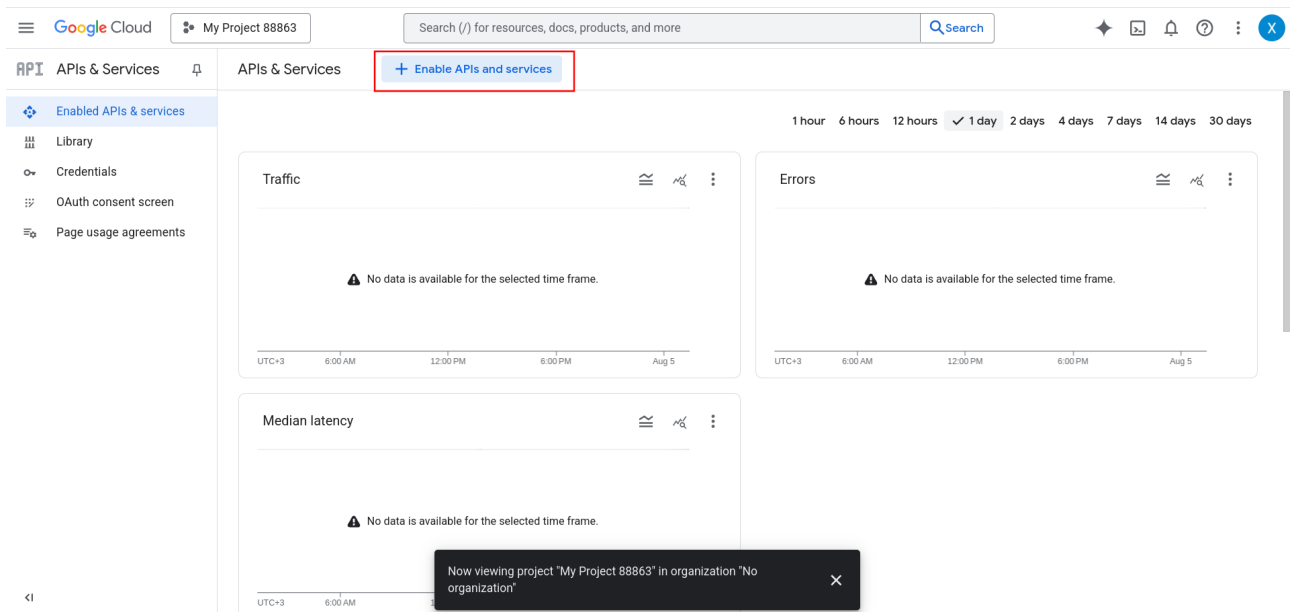
Your project is now selected.



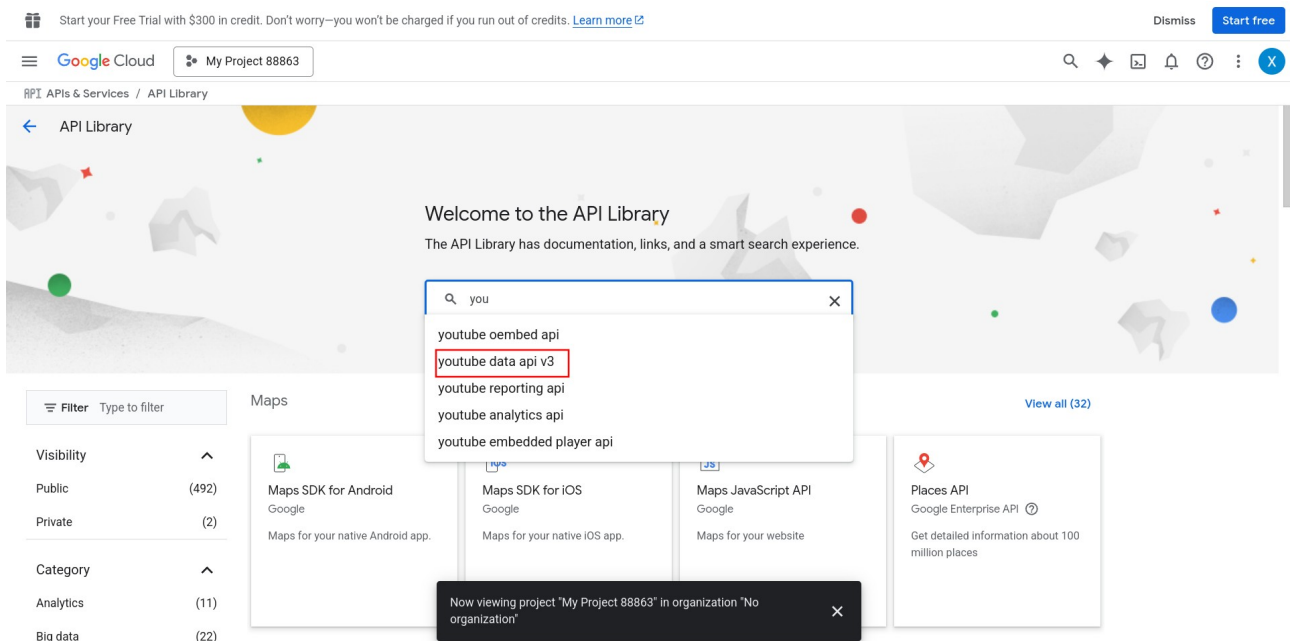
Next, you need to enable the YouTube Data API v3 and get an API key. From the left, open the menu, select "APIs & Services," and then click "Enable APIs & Services."



Then click "Enable APIs and Services" again.



In the search bar, type "you" and select "YouTube Data API v3."



Click the "YouTube Data API v3."

Start your Free Trial with \$300 in credit. Don't worry—you won't be charged if you run out of credits. [Learn more](#)

Dismiss [Start free](#)

Google Cloud My Project 88863

APIs & Services / API Library / Browse

API Library

Search youtube data api v3

API Library > "youtube data api v3"

Filter Type to filter


1 result

Visibility ^

Public (1)

Category ^

YouTube (1)

 **YouTube Data API v3**
Google
The YouTube Data API v3 is an API that provides access to YouTube data, such as videos, playlists, and channels.

Now viewing project "My Project 88863" in organization "No organization"


Click "Enable."

Start your Free Trial with \$300 in credit. Don't worry—you won't be charged if you run out of credits. [Learn more](#)

Dismiss [Start free](#)

Google Cloud My Project 88863

Product details

 **YouTube Data API v3**
[Google](#)

The YouTube Data API v3 is an API that provides access to YouTube data, such as videos, playlists,...

[Enable](#) [Try this API](#)

[Overview](#) [Documentation](#) [Support](#) [Related Products](#)

Overview

The YouTube Data API v3 is an API that provides access to YouTube data, such as videos, playlists, and channels.

About Google

Google's mission is to organize the world's information and make it universally accessible and useful. Through products and platforms like Search, Maps, Gmail, Android, Google Play, Chrome and YouTube, Google plays a meaningful role in the daily lives of billions of people.

Additional details

Type: [SaaS & APIs](#)
Last product update: 7/22/22
Category: [YouTube](#)
Service name: youtube.googleapis.com

Now viewing project "My Project 88863" in organization "No organization"

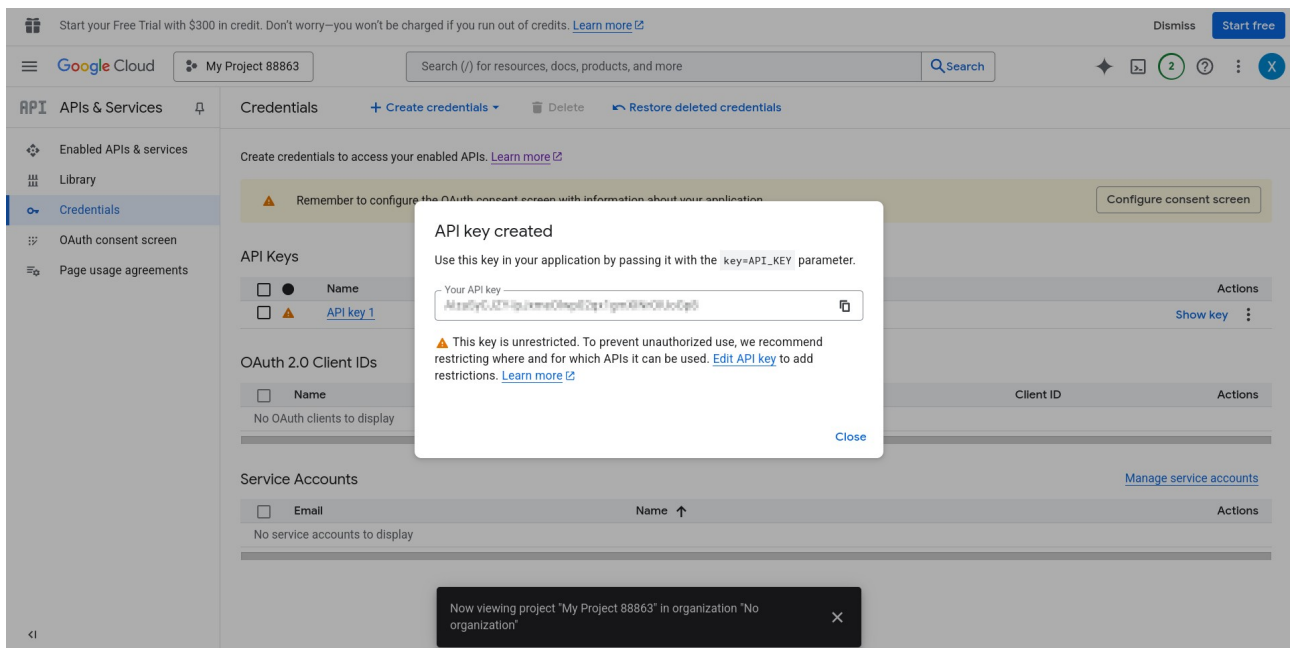
It should show something like this now.

The screenshot shows the Google Cloud console interface. At the top, there's a header with a trial offer and a search bar. The left sidebar shows navigation options like 'Enabled APIs & services', 'Library', 'Credentials', 'OAuth consent screen', and 'Page usage agreements'. The main content area is titled 'API/Service Details' and shows the 'YouTube Data API v3' is enabled. A red box highlights the API details section, which includes the service name 'youtube.googleapis.com', type 'Public API', status 'Enabled', and links for documentation, exploration, and support. Below this, there are tabs for 'Metrics', 'Quotas & System Limits', and 'Credentials'. The 'Metrics' tab is active, showing a graph of traffic by response code. A notification at the bottom states: 'Now viewing project "My Project 88863" in organization "No organization"'.

So, the API is now enabled. The only thing needed now is the API key. Click "Credentials," then "Create Credentials," and then "API key."

The screenshot shows the 'Credentials' page in the Google Cloud console. The left sidebar has 'Credentials' selected. The main content area shows options to 'Create credentials to a...' with a dropdown menu open. The dropdown menu lists four options: 'API key' (Identifies your project using a simple API key to check quota and access), 'OAuth client ID' (Requests user consent so your app can access the user's data), 'Service account' (Enables server-to-server, app-level authentication using robot accounts), and 'Help me choose' (Asks a few questions to help you decide which type of credential to use). Below the dropdown, there are sections for 'API Keys', 'OAuth 2.0 Client IDs', and 'Service Accounts', each with a table to manage them. A notification at the bottom states: 'Now viewing project "My Project 88863" in organization "No organization"'.

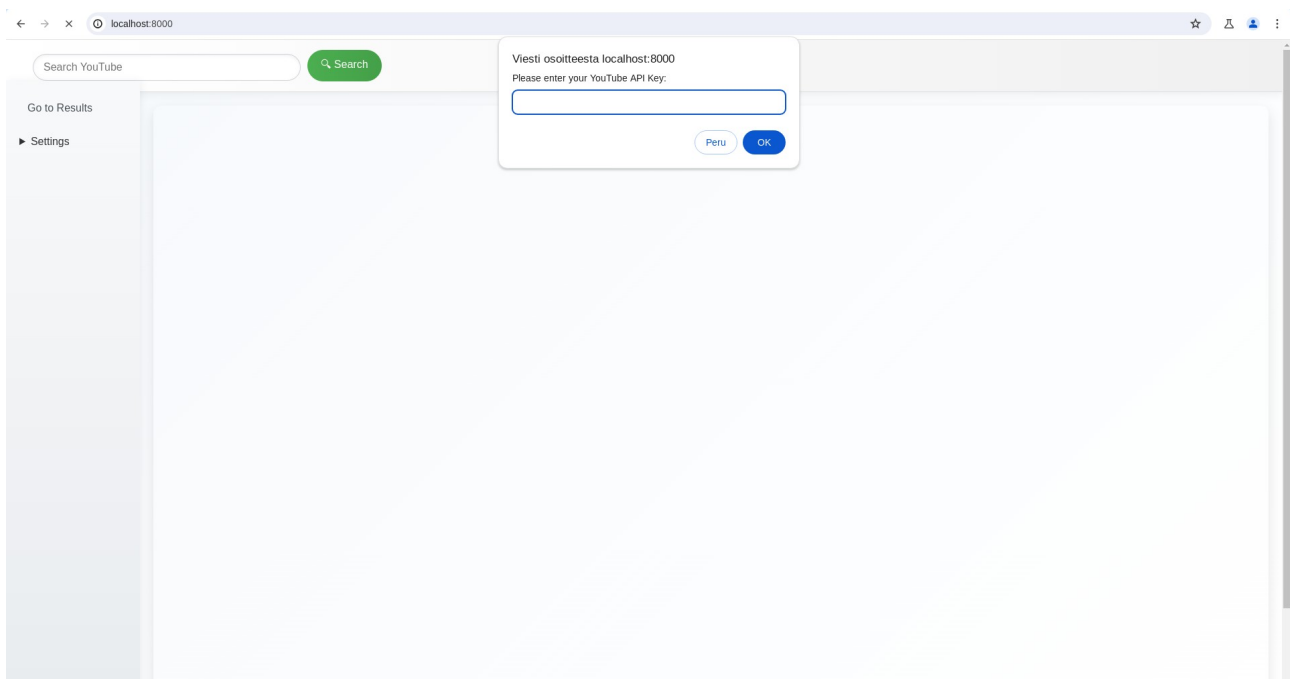
Copy and paste the generated API key somewhere safe, then log off.



Start the web server...

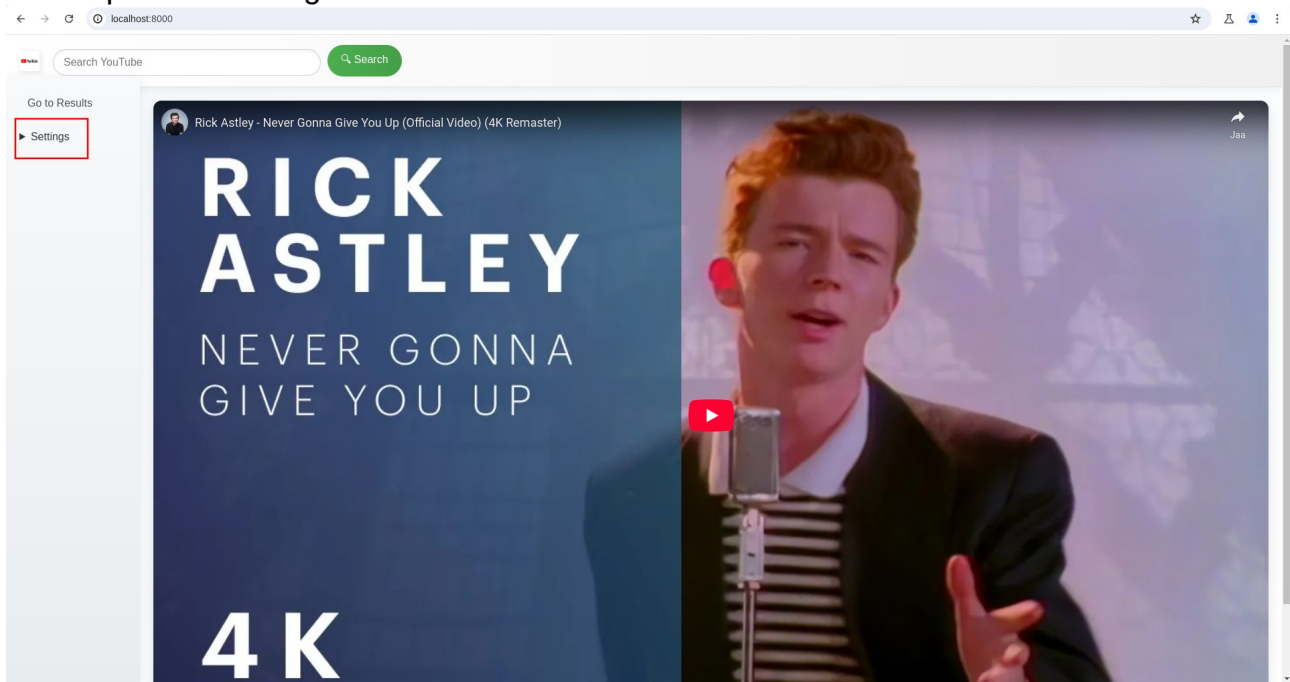
```
/tmp/LocalTube $ python -m http.server 8000
Serving HTTP on 0.0.0.0 port 8000 (http://0.0.0.0:8000/) ...
```

Then open <http://localhost:8000> in your browser. If it works, you should see something like this.

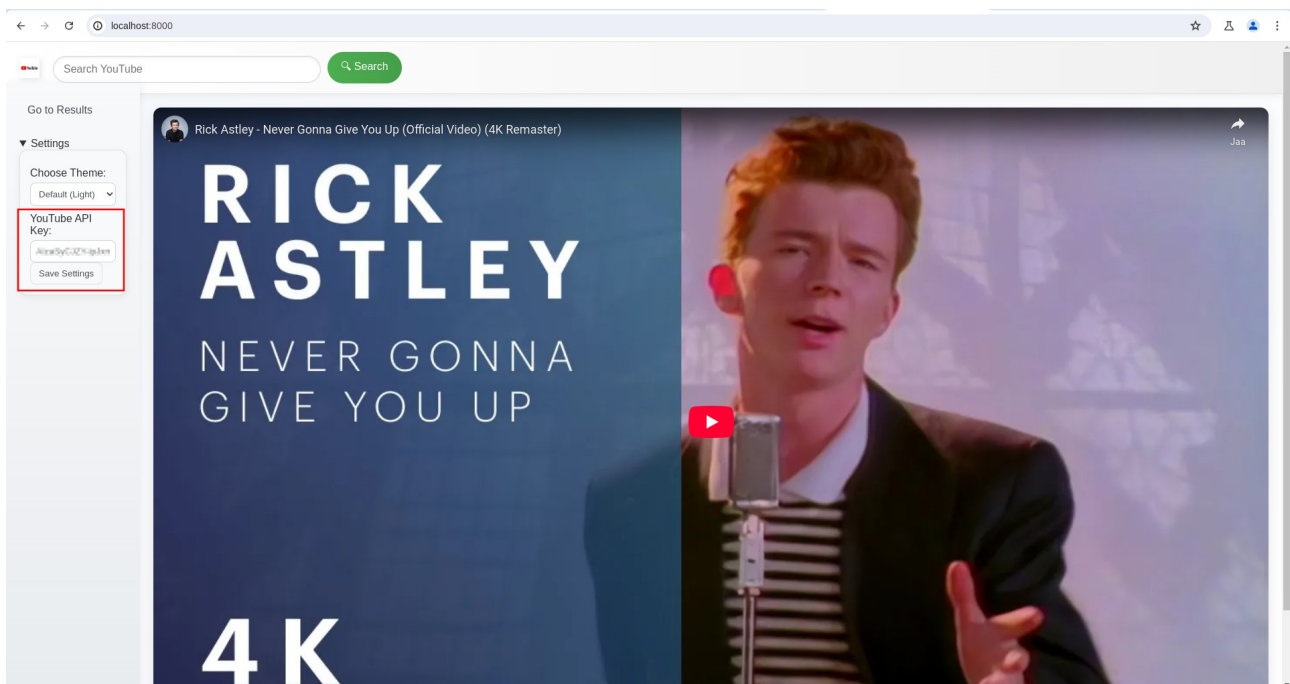


Copy-paste your YouTube API key here

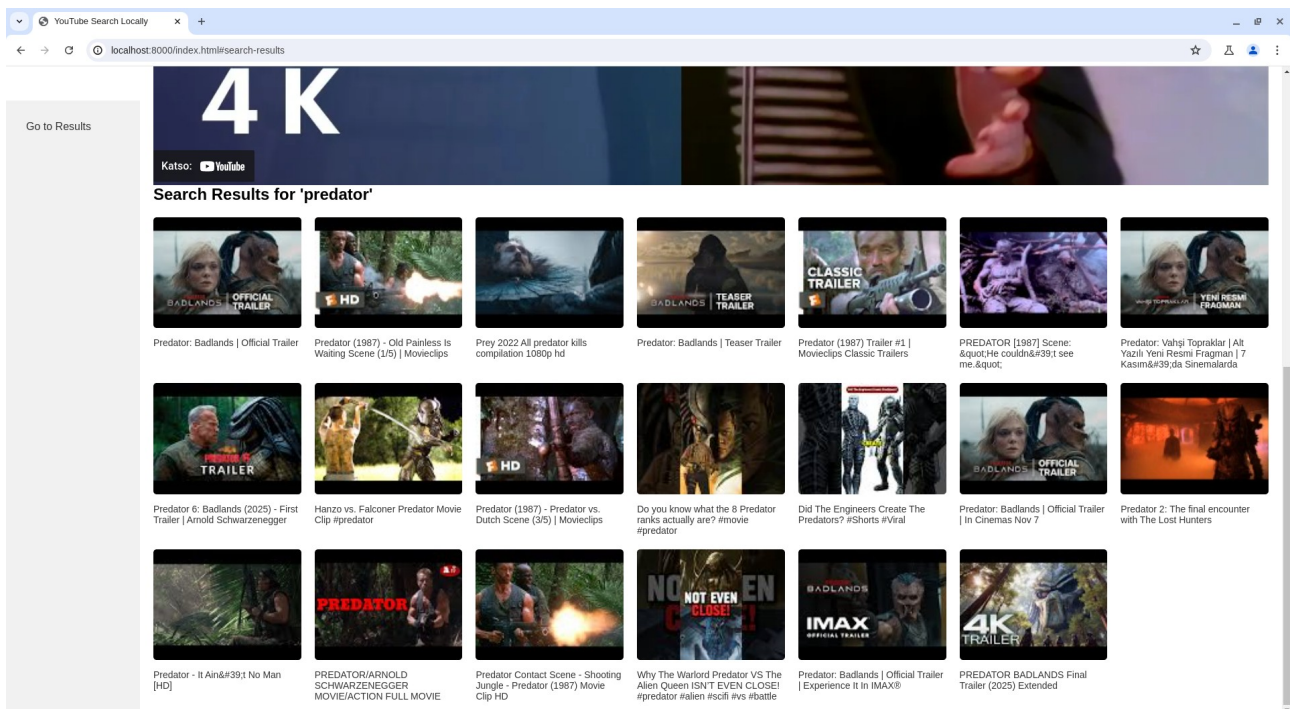
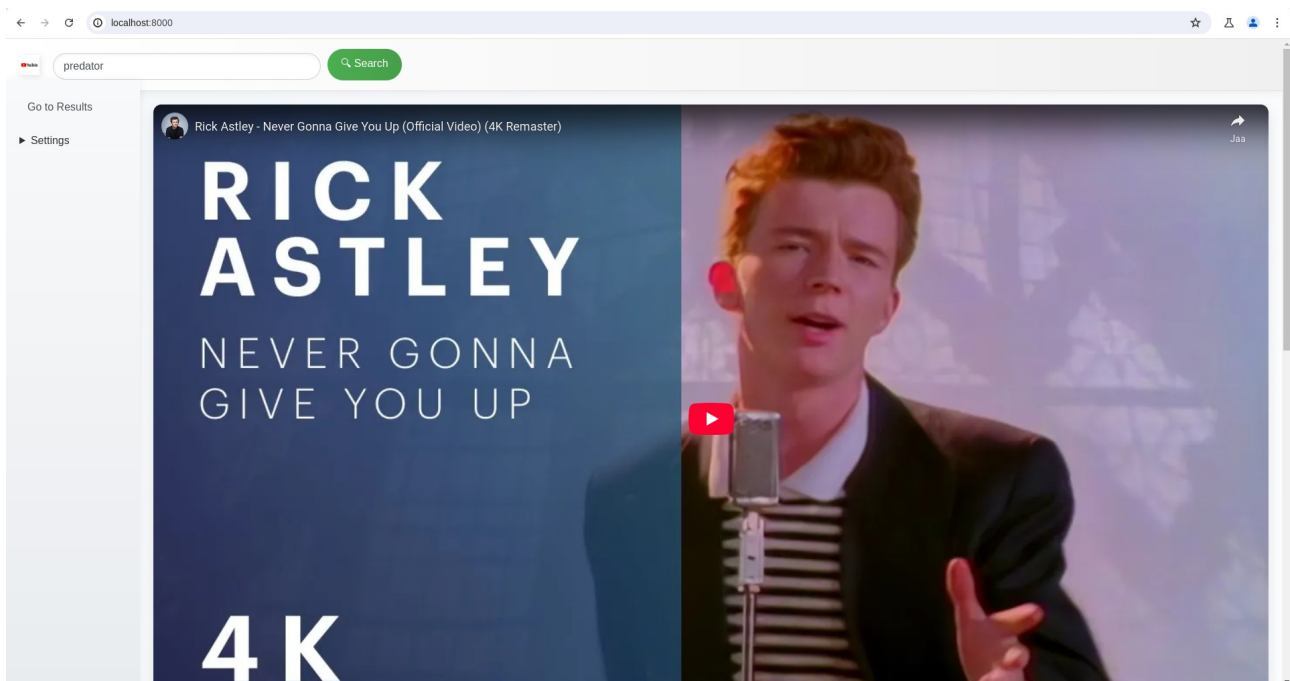
Then open the Settings from left ...



And make sure your YouTube API key is there. Then click Save Settings.



Next, you can let Ricky boy sing or search for, let's say, "Predator" (manliest movie ever made) in the search bar and then click "Go to Results."



Congrats, you now have a local YouTube front-end without Google Analytics or other user-tracking nonsense. Of course, the actual search results and video stream still come from YouTube. If you're adventurous enough, you could even put this on a remote VPS server. The catch? Your free API key allows you to make about 100 requests per day, but it should be more than enough for personal use. BTW, the search results are capped (for now) to 20 (hey! this is quick & dirty remember?) but I'm sure you can "hack" it to give more if you want.

Enabling Optional Video Downloading in LocalTube

This optional feature lets you download videos (e.g., in 1080p or higher) through LocalTube, in addition to streaming. Follow these steps to set it up:

1. Prerequisites

- Ensure Python 3.x is installed (check with: `python3 --version`).
- Ensure pip (Python package installer) is available.

2. Create a Virtual Environment

- Open a terminal in the LocalTube directory and run:

```
bash
python3 -m venv venv
```

3. Activate the Virtual Environment

- Activate it by running:

```
bash
source venv/bin/activate
```
- Your terminal prompt will change (e.g., `(venv)` appears).

4. Install Required Packages

- Install the necessary packages using the provided `requirements.txt` file:

```
bash
pip install -r requirements.txt
```
- This installs `flask` and `yt-dlp`.

5. Run the Flask Application

- Start the Flask server with:

```
bash
python3 app.py
```
- The server typically runs on `http://localhost:8000`.

6. Access LocalTube with Downloading

- Open your browser and go to `http://localhost:8000` (or the address shown in the Flask output).
- You can now stream and download videos.

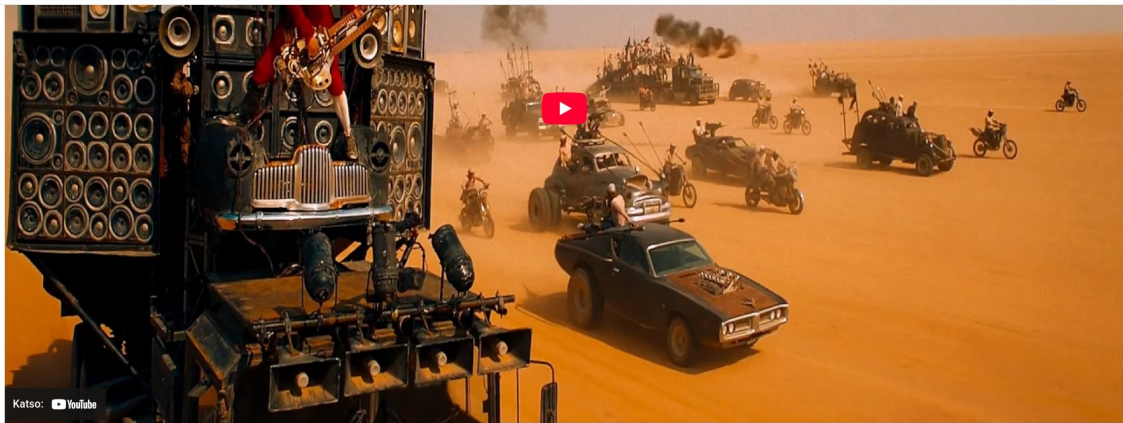
7. Deactivate the Environment

- When finished, deactivate the virtual environment by running:

```
bash
deactivate
```

Note This setup is optional and only required if you want to download videos. For basic streaming, you can simply use `python3 -m http.server 8000` without these steps.

Go to Results



Katso: YouTube

Download

: 31.3% (Speed: 2.86 MiB/s, ETA: 01:15)

Search Results for 'Mad Max'



Mad Max: Fury Road - Official Main Trailer [HD]



Mad Max: Fury Road (2015) - The chase begins (1/10) (slightly edited) [4K]



Mad Max - The Nightrider [HD]



🔪 Mad Max 2&3's Most BRUTAL (and Hilarious) Fail Ever! #WastelandChaos



MAD MAX Full Movie 2025: Furiosa Road | Superhero FXL Fantasy Movies 2025 in English (Game Movie)



Official Trailer: Mad Max (1979)



FURIOSA: A MAD MAX SAGA | OFFICIAL TRAILER #1