Google Nest Camera [Internal] API

TamirMayer Follow 10 min read · Feb 12, 2024

GitHub - TamirMa/google-nest-telegram-sync

Contribute to TamirMa/google-nest-telegram-sync development by creating an account on GitHub.

github.com

It all started a year ago when I bought the well-recommended **Google Nest Doorbell Camera** and mounted it on my front door.

I think I made the right choice. Well, almost. I'm using my own SmartHome module for controlling all of my devices, including the lights, Bosch Dishwasher, AEG Oven, and ACs, but unfortunately, Google wasn't an easy plug-and-play device.

All I wanted was a way to save my Front Door Camera video clips forever. In today's world data is not something we should delete. Storage is almost free.

So why is Google allowing me to save **ONLY 3 hours** of video clips??

I bet that's because they want you to pay $6 a month for saving your clips for the last 30 days. So I paid.

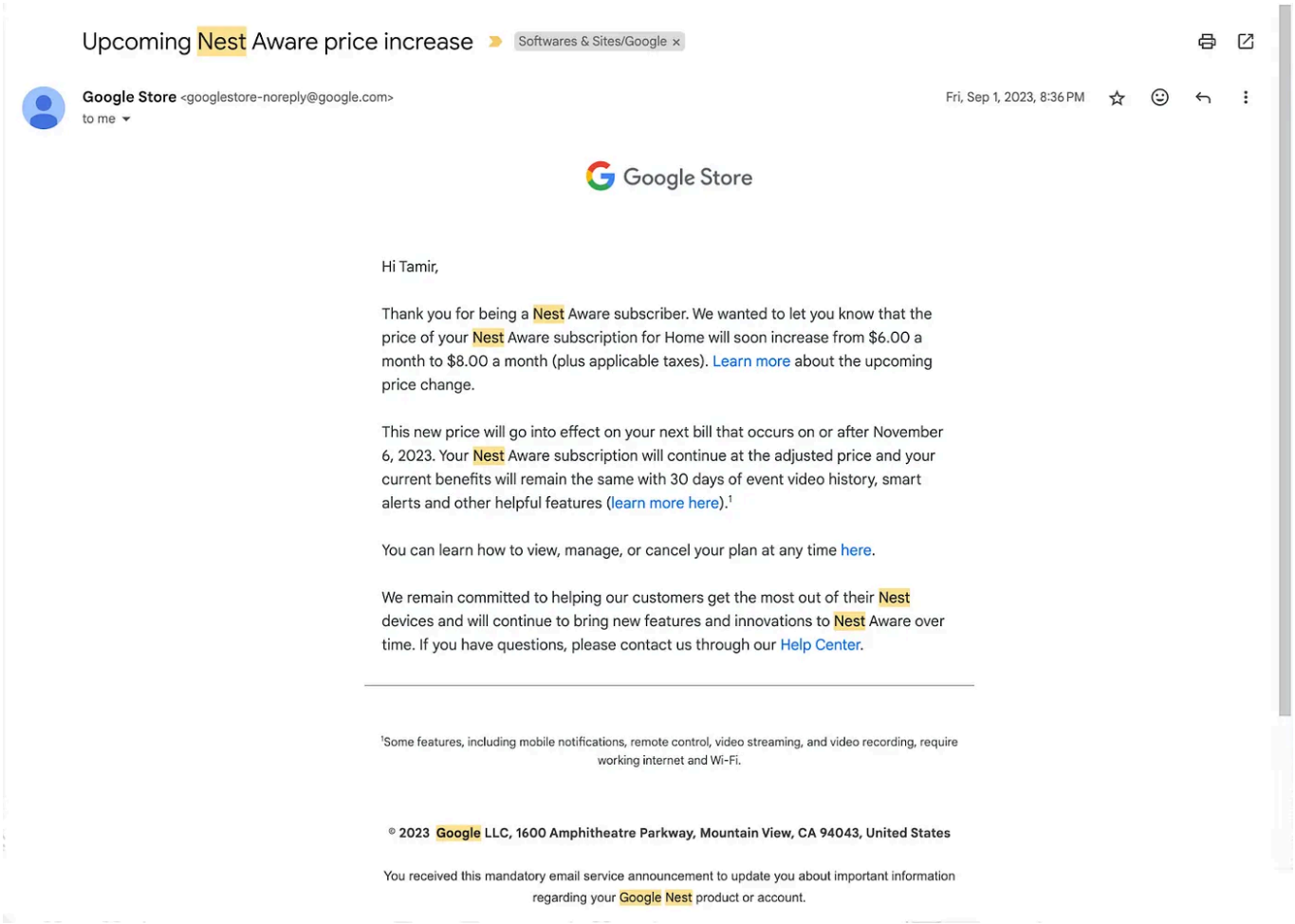
I convinced myself that $6 a month is not that much, and I will also get face detection!! A face detection that couldn't distinguish my wife's face from her 15 years younger sister.

I was feeling trapped. A simple device in today's SmartHome environment should simply give an API and let me choose what I want to do with the data. If a cloud service is what I need, let me pay for it. If an internal link is what I prefer, give me an API for that.

But Google chose the other way.

I was paying Google almost a year of monthly subscriptions (**equivalent to half of what I paid for the camera itself**)

Until they decided to raise the monthly price. $8. Same features.



Upcoming Nest Aware price increase Softwares & Sites/Google ×

Google Store <googlestore-noreply@google.com> Fri, Sep 1, 2023, 8:36 PM
to me

Google Store

Hi Tamir,

Thank you for being a Nest Aware subscriber. We wanted to let you know that the price of your Nest Aware subscription for Home will soon increase from $6.00 a month to $8.00 a month (plus applicable taxes). Learn more about the upcoming price change.

This new price will go into effect on your next bill that occurs on or after November 6, 2023. Your Nest Aware subscription will continue at the adjusted price and your current benefits will remain the same with 30 days of event video history, smart alerts and other helpful features (learn more here).[1]

You can learn how to view, manage, or cancel your plan at any time here.

We remain committed to helping our customers get the most out of their Nest devices and will continue to bring new features and innovations to Nest Aware over time. If you have questions, please contact us through our Help Center.

[1]Some features, including mobile notifications, remote control, video streaming, and video recording, require working internet and Wi-Fi.

© 2023 Google LLC, 1600 Amphitheatre Parkway, Mountain View, CA 94043, United States

You received this mandatory email service announcement to update you about important information regarding your Google Nest product or account.

Now I was pissed off. That feeling of me and all the other users having to pay for something so simple, without any other choice was making me crazy.

And that's when I decided I was going to find a way to save my Camera clips for free.

Google Nest Doorbell -> Telegram

Let's go!

The most important thing to do when starting a research project is to make sure you are not doing something already found online. On each step I made

on my project I was searching extensively if there was already something someone solved for me.

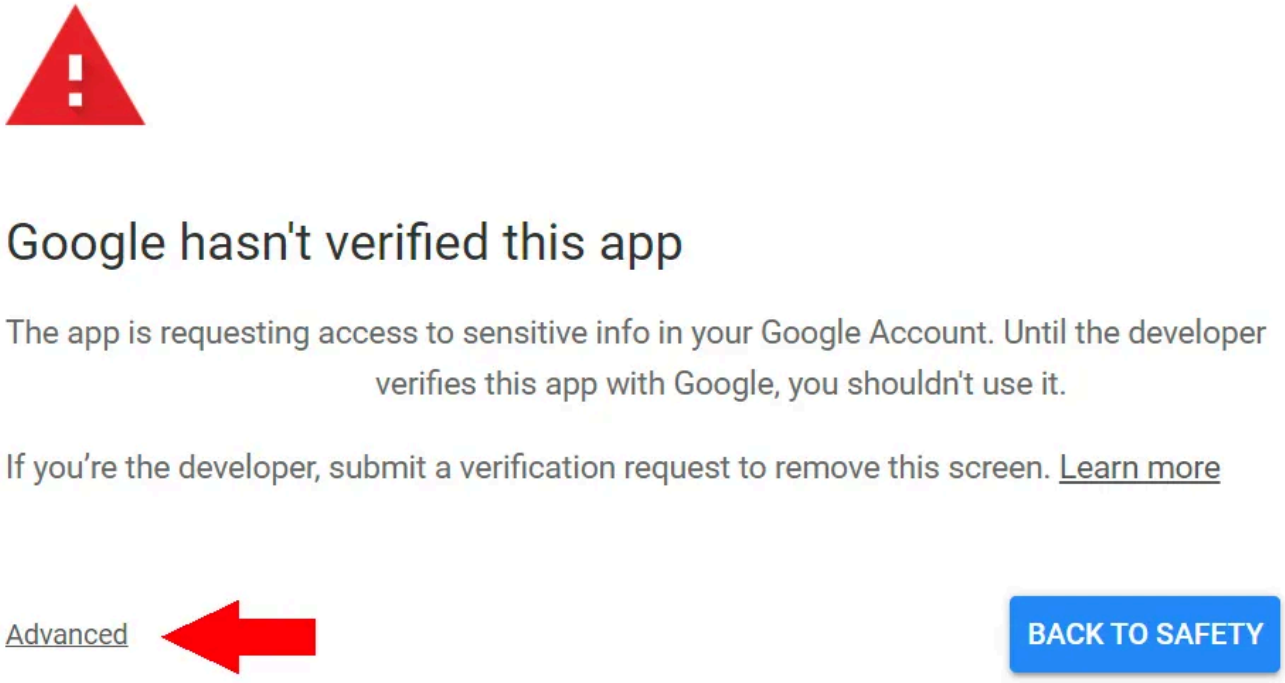
I found something. At least I thought I found it.

It's called "Smart Device Management API"

That is the API being used in Home Assistant (the well-beloved Smart Home system), Python script you can find on GitHub (I love Python!), and even Google's official developers website!

All you need to do to get access to this wonderful API is to follow these steps:

- Start your project on GCP. **WHAT THE ....?**
- Create your project on the **Google Nest** ***Device Access Console***
- **Pay $5 to start that project!**
- It feels like I'm giving it to charity.
  A charity for people to learn how to write Smart Home APIs correctly.
- Manually create user credentials for the project and copy it **locally**.
- Link my Google account while Google warns me I'm doing something wrong



- Extract the token from the URL parameters while I was redirected to google.com!!

https://www.google.com/?state=9pcGXnCebzRMvZQmLmJxbv7BJBLCfa&code=4%2F0AfJohCkAh2OUBkCOLVZIUhqFMtueBhPwKxeoYttuDQv0X6WoKgx1v7raid_GMpOzPAmk6Q&scope=https%3A%2F%2Fwww.googleapis.com%2Fauth%2Fsdm.service+https%3A%2F%2Fwww.googleapis.com%2Fauth%2Fpubsub

- Continue the process locally on your Python script to get a genuine **refresh_token.**

Simple as that!

Thank you, Google "Register for Device Access"!

I consider myself a technological person, and this one was intense. The number of ticks I missed along the way, scopes I forgot to add, redirect_uri

mismatches, figuring out what I should do with that token in the URL params, finding the right Python script for supporting this torture… It was too much. I think it took me 2 days. And I had to have a week off between those days.

Those were not 2 fun days as you can imagine.

Why can't Google provide a service for generating refresh_tokens? Do they need to raise their GCP usage for some reason?? Nah..

But now. Finally. Let's explore the API!

Simple API for Simple People

Looking on Google's official developers website was very promising:

- Get my devices
- Subscribe to events
- Access a live stream

Let's check it.

Getting my devices worked great!

```
name: enterprises/32d0f7a4-62a1-4873-813f-9367169c9a72/devices/AVPHwEtaettICkn7l
parentRelations:
- displayName: Front door
parent: enterprises/32d0f7a4-62a1-4873-813f-9367169c9a72/structures/AVPHwEvEcNRv
traits:
sdm.devices.traits.CameraImage:
maxImageResolution:
height: 1200
width: 1920
sdm.devices.traits.CameraLiveStream:
```

```
audioCodecs:
- OPUS
supportedProtocols:
- WEB_RTC
videoCodecs:
- H264
sdm.devices.traits.CameraMotion: {}
sdm.devices.traits.CameraPerson: {}
sdm.devices.traits.DoorbellChime: {}
sdm.devices.traits.Info:
customName: Front door doorbell
type: sdm.devices.types.DOORBELL
```

Okay but where can I get my recent video clips?

Maybe if we subscribe to the events I will get the videos.

It's called Google Pub/Sub

Subscribing to the events was easy using the python-google-nest-sdm module. Thank god people are writing such complicated modules just to spread good out there.

All you need to do for subscribing to events

1. Go to Device Access Console
2. On your GCP project, create a subscription
3. Pray
4. Pray harder

I was using the simple command

```
google_nest -project_id "31e0a7a9-61a8-4873-823f-9375169d9d72" \
        -client_id "43732173191-xxxxyyyyyyzzzzzzz.apps.googleusercontent.com" \
        -client_secret "GOCSPX-V937Nz_NAqyasd83SDAi3-1H-7r" \
        -v subscribe front_doorbell_subscription
```

But nothing happened.

So I was praying even harder. Created the subscription one more time, refreshed the token, set the parameters again, walking back and forth in front of my camera to generate events (**My neighbors must think I'm crazy, good thing they don't have a Doorbell Camera**)

And then… an EVENT!!


```
{
  "eventId": "7c9af6a3-xxxx-yyyy-zzzz-75daa4a2e1f5",
  "timestamp": "2019-01-01T00:00:01Z",
  "resourceUpdate": {
    "name": "enterprises/31e0a7a9-61a8-4873-823f-9375169d9d72/devices/AVPHwEtaet
    "events": {
      "sdm.devices.events.CameraClipPreview.ClipPreview": {
        "eventSessionId": "armFIzqWyT15KwoKWig…",
        "previewUrl": "https://nest-camera-frontend.googleapis.com/frontend/encr
      }
    }
  },
  "userId": "AVPHwEtaettICk…",
  "resourceGroup": [
    "enterprises/31e0a7a9-61a8-4873-823f-9375169d9d72/devices/AVPHwEtaettICk…."
  ]
}
```


Look at it. How beautiful it is. Let's download the video clip 🙂

So I downloaded the clip in “previewUrl” using the same token we have used so far. You will be amazed I was just getting the preview clip. 10 frames! That’s what Google gave me.

But wait. Where can I get the full video?

Can I use the SessionID? Should I wait for another event? Is there an API call for getting the recent videos?

Nope.
Nothing!
No documentation for full videos, leaving me just the option to create my own WebRTC stream each time I get a notification.

Create a video Stream — Good luck with that

It’s too frustrating to write about all my efforts in recording the Google stream using Python’s aiortc, and the complicated un-documented Google’s SDP offering mechanism which is described without an example for streaming it.

When I realized creating a video stream would block my Google Home from saving the video in the app, plus the fact that I was not getting all the motion events but only some part of it, I decided to look for another solution.

Why can’t I find a solution online?

Am I the only person who wants to back up its Google Nest Doorbell videos outside the Google Home app?

Probably not. So let’s figure out how to do that!

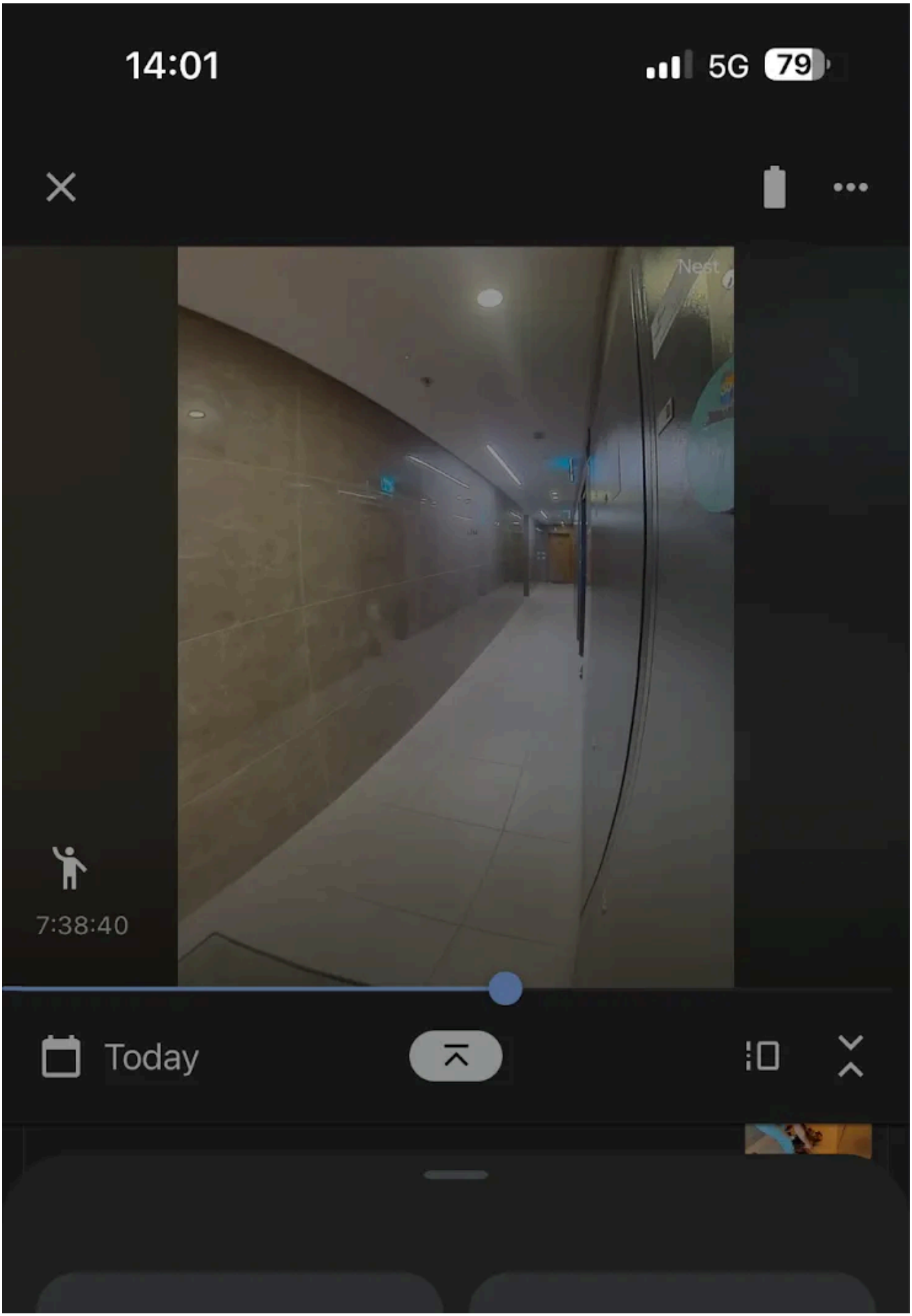
A final search through the internet, including some late-night talking with ChatGPT, Google Bard (not leaking any internal information), Bing.com's subset of GPT 4, github.com, Nothing...

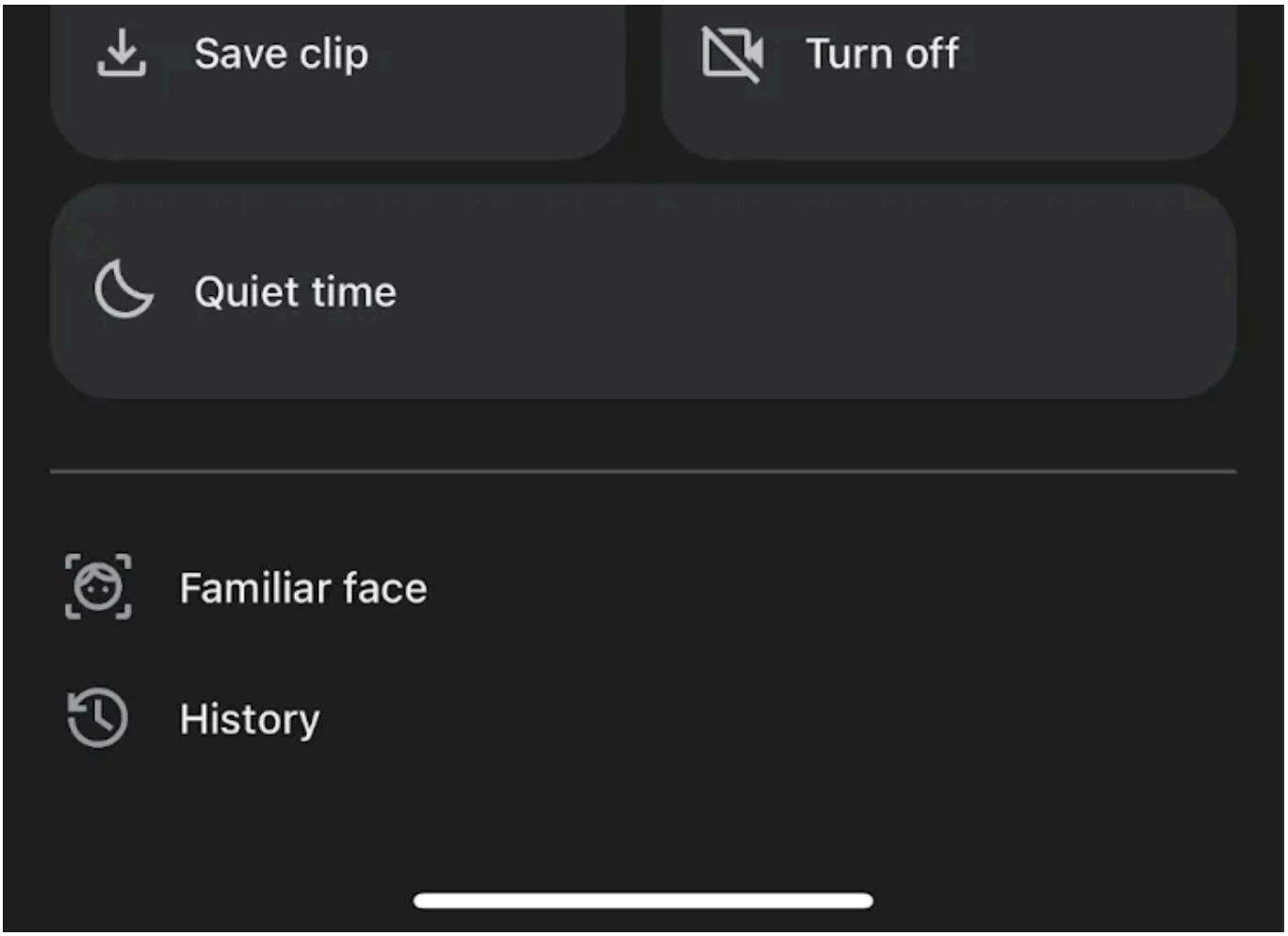
Let's Start

And we are off to a great start 🙂

The only question I keep in mind is what does the Google Home app do? How does it access the "History" videos in the app?

The main thing I wanted to achieve was to implement the "Save Clip" option from the app.


14:01
5G
79
Nest
7:38:40
Today



All this project was for implementing the “Save clip” button in a Python script

When I was trying to mitmproxy my iPhone and see the requests I was not surprised Google made it extra hard and verified the RootCA is only Google’s.

A short search online suggested installing an Android virtual device using “Android Studio” and intercepting the app of the virtual device.

I found the following blog which was almost great for describing how to set up your environment: *capturing-grpc-requests-in-mobile-applications*

The only part I was skipping was setting up the app proxy.

I was using another script to intercept the app called “frida-interception-and-unpinning”.

- Clone the git locally
- Edit the config file
- Add the following conscrypt-bypass I found here:
- ./android/consrypt-bypass.js

```
Java.perform(function() {
    var TrustManagerImpl = Java.use('com.android.org.conscrypt.TrustManagerImpl'
    var ArrayList = Java.use("java.util.ArrayList");
    TrustManagerImpl.verifyChain.implementation = function(untrustedChain, trust
    host, clientAuth, ocspData, tlsSctData) {
        console.log("[+] Bypassing TrustManagerImpl->verifyChain()");
        return untrustedChain;
    }
    TrustManagerImpl.checkTrustedRecursive.implementation = function(certs, host
    trustAnchorChain, used) {
        console.log("[+] Bypassing TrustManagerImpl->checkTrustedRecursive()");
        return ArrayList.$new();
    };
});
```

Finally, you should have the following setup installed:

- Android studio with a Pixel device, Android 9 (Pie) with Google APIs enabled
- ADB is installed on your computer
- mitmweb is up and running
- The mitmproxy certificate should be installed on your Android device
- frida server running on your device as root

```
generic_arm64:/ # /data/local/tmp/frida-server-16.1.11-android-arm64
```

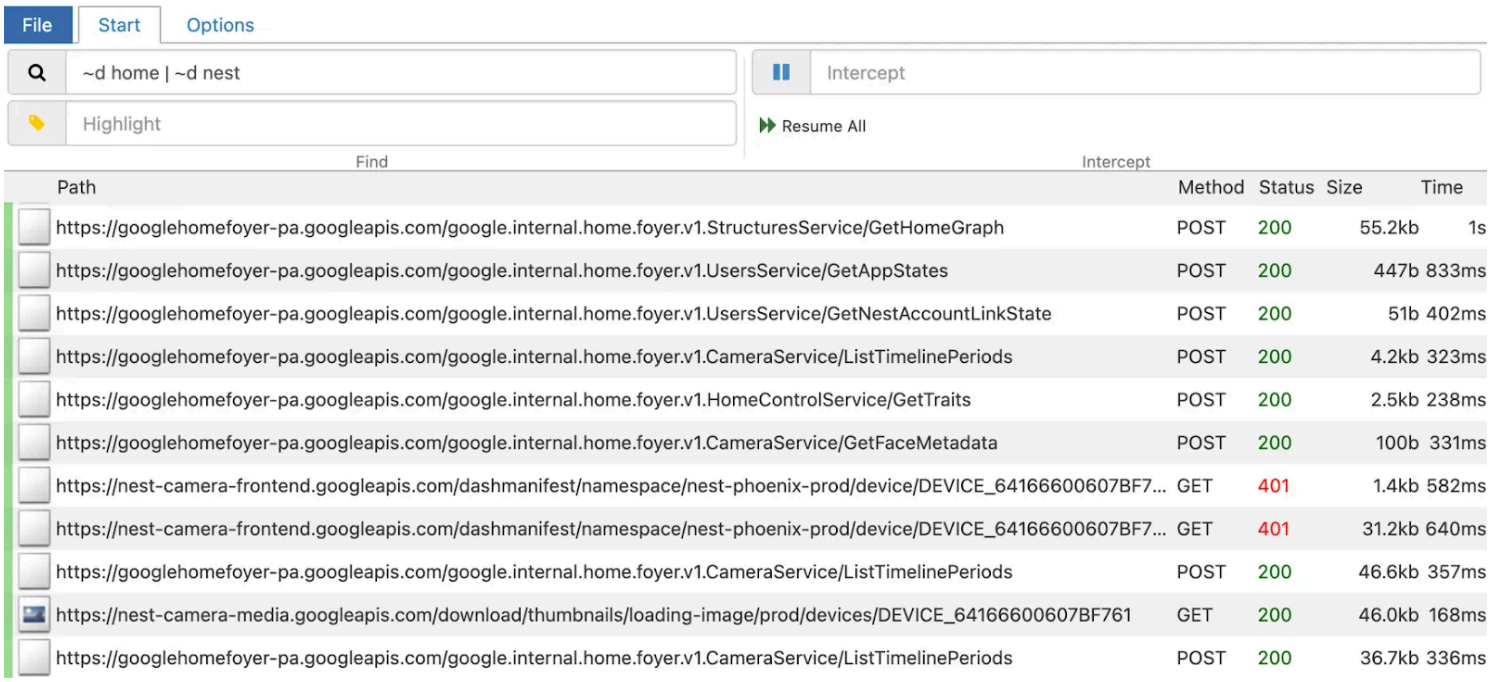
- The APK installed on your device

```
adb install APK_PATH
```

- Run the frida scripts for intercepting the HTTPS

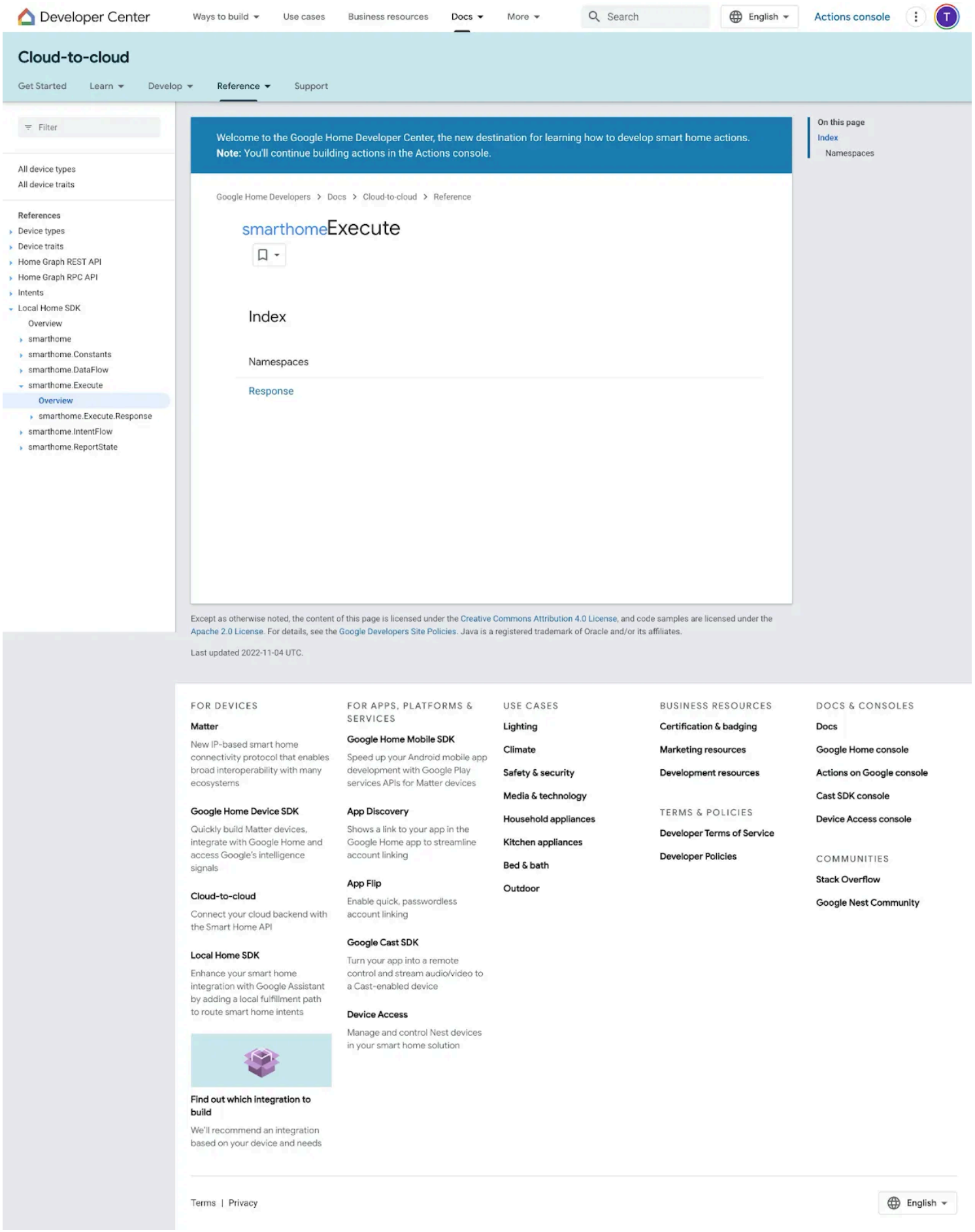
```
frida -U \
  -l ./config.js \
  -l ./native-connect-hook.js \
  -l ./android/android-proxy-override.js \
  -l ./android/android-system-certificate-injection.js \
  -l ./android/android-certificate-unpinning.js \
  -l ./android/android-certificate-unpinning-fallback.js \
  -l ./android/consrypt-bypass.js \
  -f com.google.android.apps.chromecast.app
```

OMG! I'm seeing the (decrypted) protocol!!!



HomeGraph

What's homegraph? Well, it seems like an almost public API for controlling your Google Home devices. A quick look here shows how much Google wants me to use this API:


Developer Center
Ways to build
Use cases
Business resources
Docs
More
Search
English
Actions console
Cloud-to-cloud
Get Started
Learn
Develop
Reference
Support
Filter
All device types
All device traits
References
Device types
Device traits
Home Graph REST API
Home Graph RPC API
Intents
Local Home SDK
Overview
smarthome
smarthome.Constants
smarthome.DataFlow
smarthome.Execute
Overview
smarthome.Execute.Response
smarthome.IntentFlow
smarthome.ReportState
Welcome to the Google Home Developer Center, the new destination for learning how to develop smart home actions.
Note: You'll continue building actions in the Actions console.
Google Home Developers > Docs > Cloud-to-cloud > Reference
smarthomeExecute
Index
Namespaces
Response
On this page
Index
Namespaces
Except as otherwise noted, the content of this page is licensed under the Creative Commons Attribution 4.0 License, and code samples are licensed under the Apache 2.0 License. For details, see the Google Developers Site Policies. Java is a registered trademark of Oracle and/or its affiliates.
Last updated 2022-11-04 UTC.
FOR DEVICES
Matter
New IP-based smart home connectivity protocol that enables broad interoperability with many ecosystems
Google Home Device SDK
Quickly build Matter devices, integrate with Google Home and access Google's intelligence signals
Cloud-to-cloud
Connect your cloud backend with the Smart Home API
Local Home SDK
Enhance your smart home integration with Google Assistant by adding a local fulfillment path to route smart home intents
Find out which integration to build
We'll recommend an integration based on your device and needs
FOR APPS, PLATFORMS & SERVICES
Google Home Mobile SDK
Speed up your Android mobile app development with Google Play services APIs for Matter devices
App Discovery
Shows a link to your app in the Google Home app to streamline account linking
App Flip
Enable quick, passwordless account linking
Google Cast SDK
Turn your app into a remote control and stream audio/video to a Cast-enabled device
Device Access
Manage and control Nest devices in your smart home solution
USE CASES
Lighting
Climate
Safety & security
Media & technology
Household appliances
Kitchen appliances
Bed & bath
Outdoor
BUSINESS RESOURCES
Certification & badging
Marketing resources
Development resources
TERMS & POLICIES
Developer Terms of Service
Developer Policies
DOCS & CONSOLES
Docs
Google Home console
Actions on Google console
Cast SDK console
Device Access console
COMMUNITIES
Stack Overflow
Google Nest Community
Terms | Privacy
English

> *Every low-effort documentation like this just adds to my motivation to offer a better experience to Google Nest Doorbell.*

So what's now?

It seems like homegraph is a semi-public API for contacting your home devices.

To use this API I found the awesome Python module "*glocaltokens*"!

A simple "pip install glocaltokens" would get you a lifetime of opportunities.

But first, a token. The easiest way to create a master token is by using the following docker:

```
$ docker run --rm -it breph/ha-google-home_get-token
```

I would recommend using a one-time password for your Google account (don't forget to use two-way authentication before trying to create one). And make sure to handle it carefully and securely.

The docker will create a "master_token" for you. This is what you need for using glocaltokens!

```
from glocaltokens.client import GLocalAuthenticationTokens

client = GLocalAuthenticationTokens(
  master_token="<The master token you created>"
```

```
    username="<YOUR_GOOGLE_USERNAME>", # your gmail address
    password="FAKE_PASSWORD" # won't be used
)

homegraph_response = client.get_homegraph()
# This one will list all your home devices
# One of them would be your Nest Camera
homegraph_response.home.devices
```

AS SIMPLE AS THAT!

How can we get the video clips?

As I was exploring the requests being created by my Google Home app, I saw that the list of commands being sent is much longer than the command documented on Google's developers website.

The name "google.**internal**.home.foyer.v1.CameraService" would suggest Google doesn't want me to know about it.

And then, **Protobuf. Oh no.**

I couldn't express how much I hate Protobuf. I think that's the cruelest way to send data to other people. It might be data-efficient, but is it time-efficient when developing a program? I bet it's not. Especially when you don't have the original .proto files.

I was looking for all the non-protobufs requests over there, and found two important ones:

- https://nest-camera-frontend.googleapis.com/dashmanifest/namespace/nest-phoenix-

prod/device/DEVICE_64166600...?start_time=2024-02-08T10%3A14%3A37.480Z&end_time=2024-02-08T22%3A14%3A37.480Z&types=1&types=2&types=4&variant=2

Which returned a list of events my device recorded in the time frame requested. This is an XML format. I would take XML over Protobuf any day.

- https://nest-camera-frontend.googleapis.com/mp4clip/namespace/nest-phoenix-prod/device/DEVICE_64166600...?start_time=1707363180876&end_time=1707368757371

Which returns a FULL QUALITY VIDEO for a given time frame.

Bingo!

The last piece of the puzzle was how to create an access token for these requests. glocaltokens's client.get_access_token() was not giving me a valid one.

I found a Google service for checking access tokens, and handled the access token used in my mitmproxy session:

```
curl "https://oauth2.googleapis.com/tokeninfo?access_token=ACCESS_TOKEN"
```

And received the following response:

```
{
  "azp": "498579633514-vipn56bj423ufpht3gm6bd23fjl74a06.apps.googleusercontent.c
  "aud": "498579633514-vipn56bj423ufpht3gm6bd23fjl74a06.apps.googleusercontent.c
  "scope": "https://www.googleapis.com/auth/nest-account",
  "exp": "1707660265",
  "expires_in": "3593",
  "access_type": "offline"
}
```


A little guess I made is that "*https://www.googleapis.com/auth/nest-account*" is the service I had to use while getting a dedicated Nest access_token instead of the one glocaltokens were using for homegraph ("*https://www.google.com/accounts/OAuthLogin*"). So I created a modified get_access_token call using the new service..

And...

Then...

I GOT A PROPER GOOGLE NEST ACCESS TOKEN!

This is it!

- Get a master token
- Use glocaltokens to get an access_token for the service "*https://www.googleapis.com/auth/nest-account*"
- Request the events you want to get for a given time frame:
- *https://nest-camera-frontend.googleapis.com/dashmanifest/namespace/nest-phoenix-prod/device/{DEVICE_ID}?start_time={ISO_FORMAT_8601_Z}&end_time={ISO_FORMAT_8601_Z}&types=1&types=2&types=4&variant=2*

- Parse the XML and its events (“Periods”)
- *https://nest-camera-frontend.googleapis.com/mp4clip/namespace/nest-phoenix-prod/device/{DEVICE_ID}?start_time={microsec_timestamp}&end_time={microsec_timestamp}*
- **Enjoy with your newly created mp4 clips!**

Saving the videos

For the last part, I needed a place to save all those video clips. Although having a well-paid Google Drive plan, I decided to keep my storage for other things.

As I’m a true fan of Telegram, I thought it would be nice to get all those videos in a Telegram channel.

So, a quick Telegram Bot creation (using Bot Father), a quick Telegram Channel creation, and a short Python script for initializing the bot.

And — that’s it.

Here’s a shortcut for you to do it too: https://github.com/TamirMa/google-nest-telegram-sync/blob/main/README.md#usage

A Final Word

This was not easy, but seeing how it all worked out was fun!
I do think Google should reconsider its Nest Cameras API strategy. In today's world, especially in IOT, there's no room for hidden APIs to block your users from using such a basic feature as downloading a video clip captured by your camera.

If you made it here, I hope you enjoyed reading my blog, and please feel free to contact me with any questions!

See you next time :)

Google Nest IoT Research Google Nest

Written by TamirMayer

3 followers · 3 following

https://www.linkedin.com/in/tamir-mayer/

Follow

Responses (2)

Karthikeyan Mahadevan

What are your thoughts?

Casey L Chadwell
Mar 30

This is great, I experienced some of the same challenges and this helped me a lot!

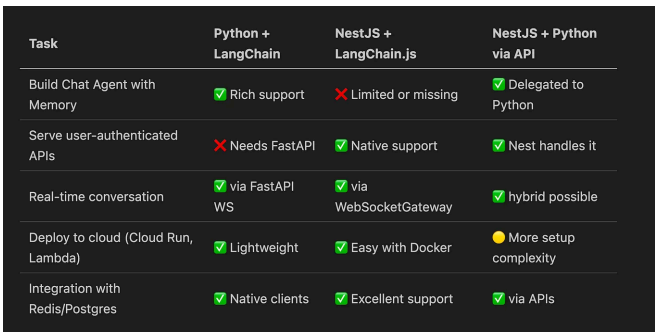
3 Reply

Beyond Cradle
Sep 25, 2024

This is Brilliant Tamir

Reply

Recommended from Medium



Task	Python + LangChain	NestJS + LangChain.js	NestJS + Python via API
Build Chat Agent with Memory	✅ Rich support	❌ Limited or missing	✅ Delegated to Python
Serve user-authenticated APIs	❌ Needs FastAPI	✅ Native support	✅ Nest handles it
Real-time conversation	✅ via FastAPI WS	✅ via WebSocketGateway	✅ hybrid possible
Deploy to cloud (Cloud Run, Lambda)	✅ Lightweight	✅ Easy with Docker	🟡 More setup complexity
Integration with Redis/Postgres	✅ Native clients	✅ Excellent support	✅ via APIs

Pradeep Gudipati

Ravi Savaliya

🧠 Python vs NestJS for LangChain: A Performance & Developer...

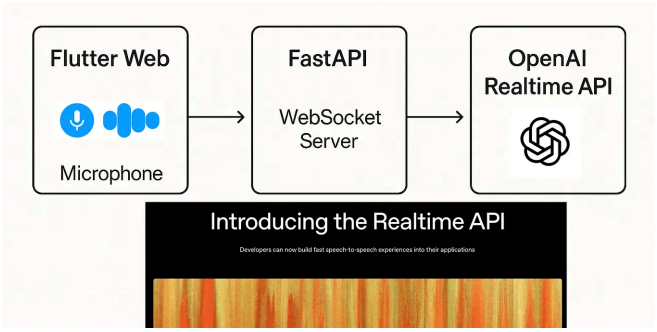
As AI systems become more complex and personalized, developers are increasingly...

May 6

What's New in Flutter 3.35—Real Use Cases, Migration Guide &...

Flutter 3.35 is here, and it's more than just another version bump. This release brings...

Aug 18 29 1

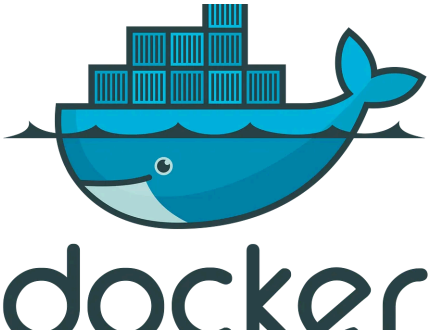


CellCS

OpenAI Realtime API: Practical Guide with Real Case Example 1/2

1 OpenAI Realtime API

Apr 28 23

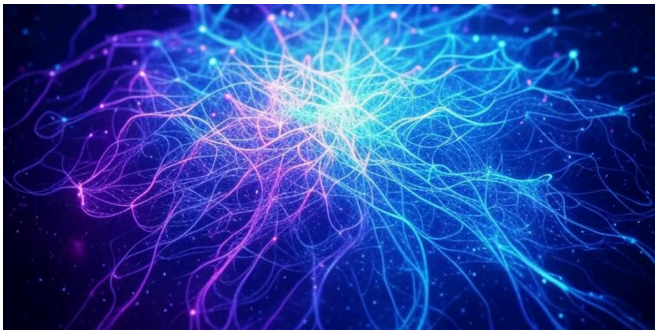


Abhinav

Docker Is Dead—And It's About Time

Docker changed the game when it launched in 2013, making containers accessible and...

Jun 8 5.4K 137



Sajith K

Using PostgreSQL with LangGraph for State Management and Vector...

LangGraph is a powerful framework for building stateful, conversational AI...

Alex Fuentes

The Future is Speech-to-Speech: How S2S Models Are...

The Traditional Voice Agent Pipeline: A Beautiful Mess

Apr 27 63 2 Aug 4 2

See more recommendations