UST GENCYS 2025 CTF WRITEUP

TEAM NAME : GITCYCOOS
**FINAL RANK : 29th

Team Member 1 : KOTTALI ROHAN KARTHIK
Team Member 2 : MOHAMMED SHEDABED

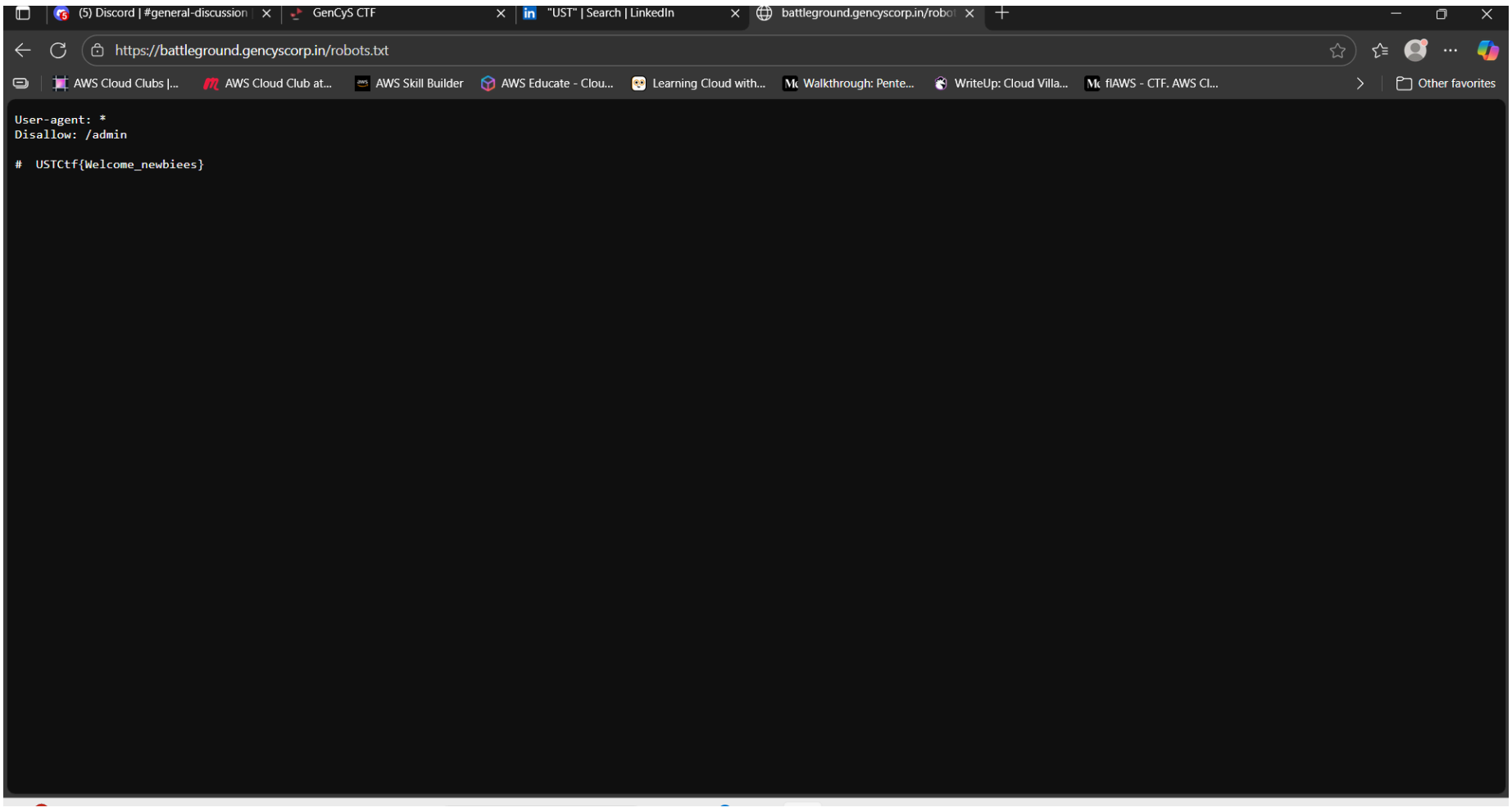
MISC

Challenge name : Welcome

Points - 5

solution: Testing with robots.txt

https://battleground.gencyscorp.in/robots.txt
In concept of Web Application Vulnerability Testing, Testing with robots.txt is the Basic Fundamental Concept
Hence when tested with robots.txt were in developers miss to hide the path /robots.txt



Final Flag : USTCtf{Welcome_newbiees}

OSINT

Challenge Name - Hidden in Plain Sight

Points : 100

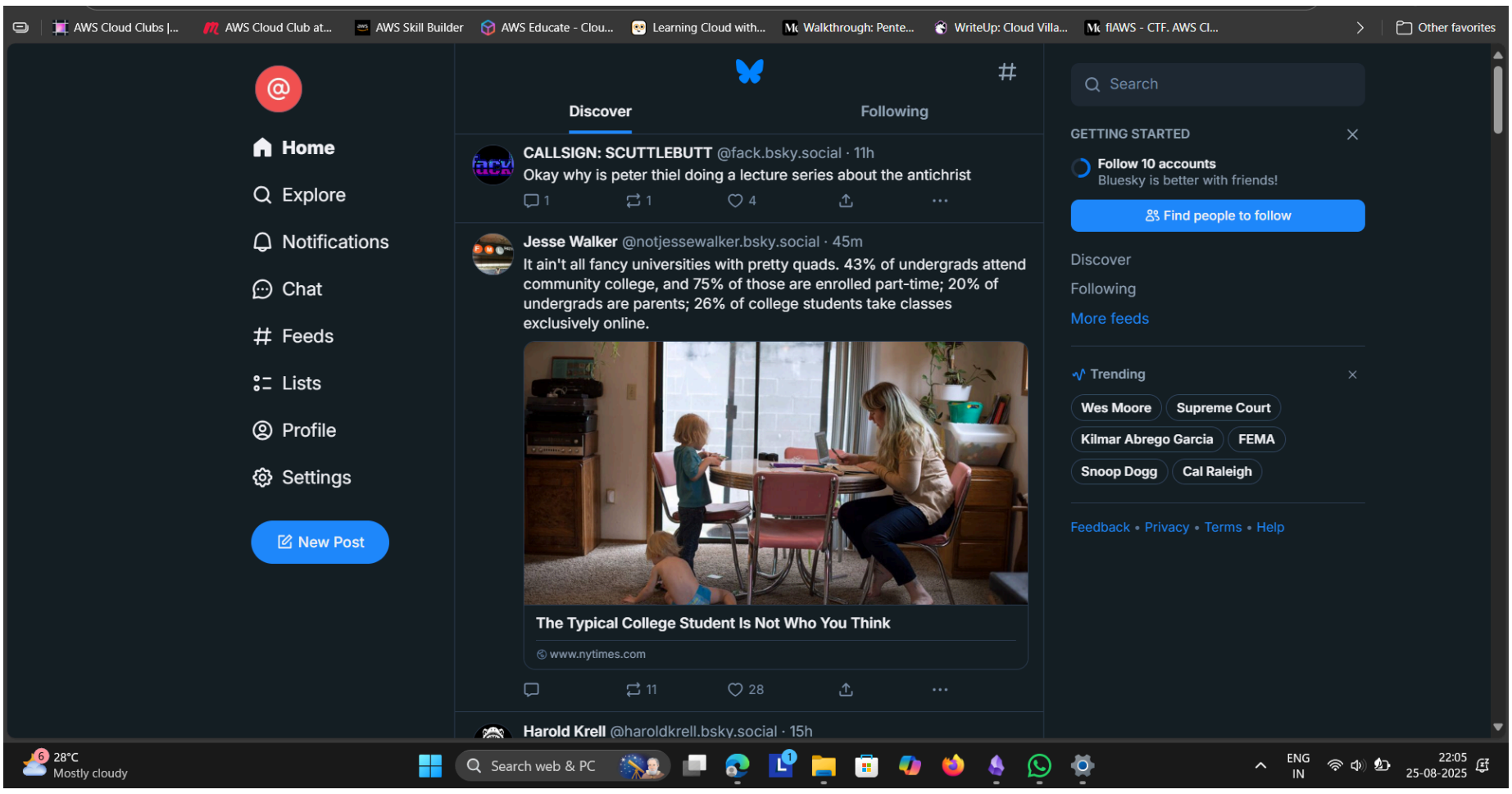
Description: Hidden in Plain Sight The mysterious group GenCyS left behind a picture. Hidden within the details is a clue to their last dinner and it is their online identity. Find it — then use that social media, there's something interesting waiting for you. Sometimes, the place you eat becomes the name you wear online. Don't just think of the big social media platforms. There are newer ones rising in the cloud too. they gave me a osint challenge where the image has a letter saying pikbest

solution:
when looking the image in detail we can see a Name highlighting "Pikbest"
A hint given that it is related to last dinner of gencys and to a social media platform but a newer one which is growing in cloud

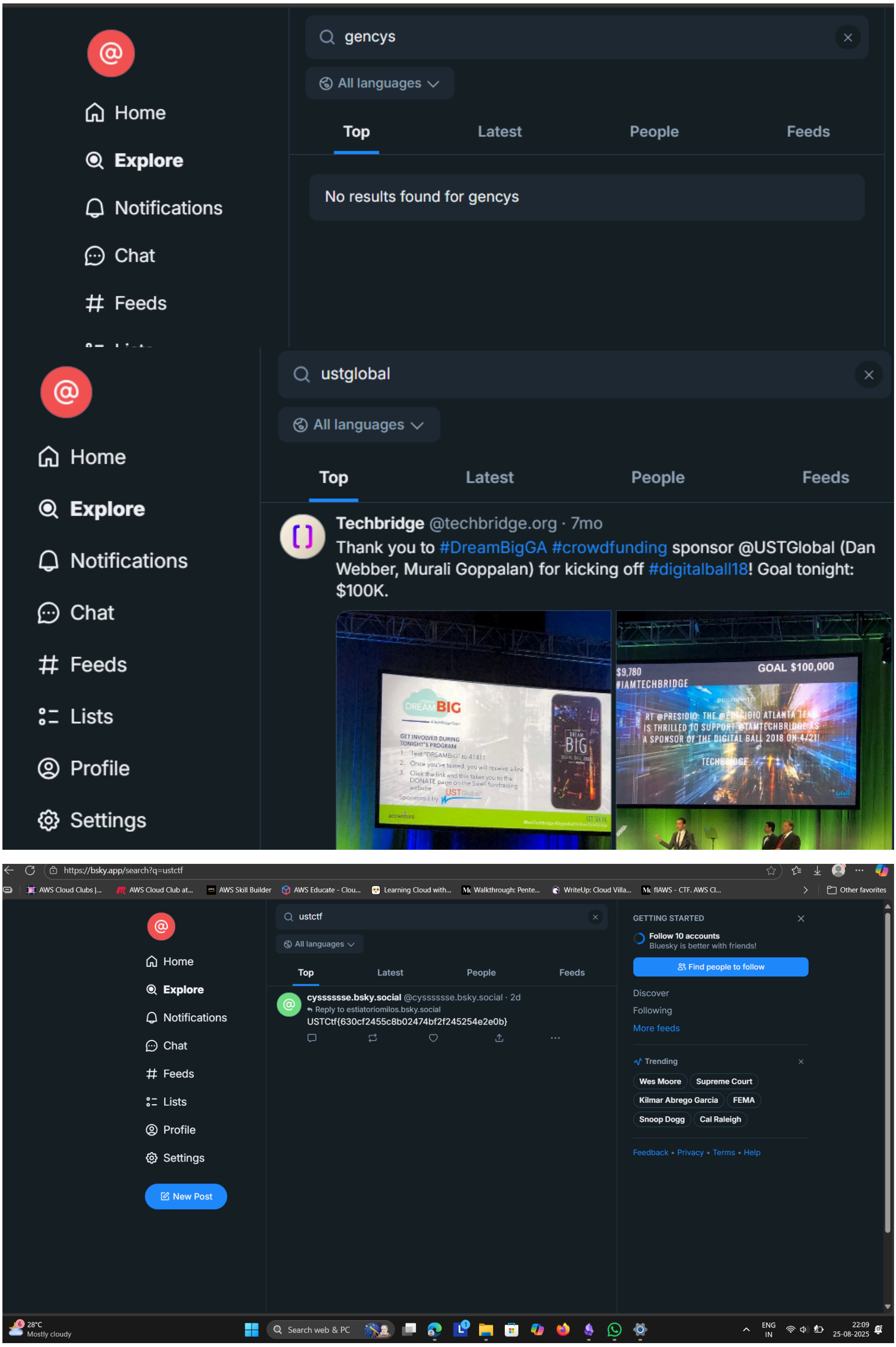
We tried with other social media platform but could not find a lead , we remember using BlueSky platform in our last ctf

Bluesky is an American **microblogging social** media **service**

so we checked with keywords



Searching with keywords :
gencys
ustglobal
ustctf



Bingo! here is the flag for this challenge with the keyword "ustctf"

Final Flag : USTCtf{630cf2455c8b02474bf2f245254e2e0b}

Challenge Name : Where am I ?

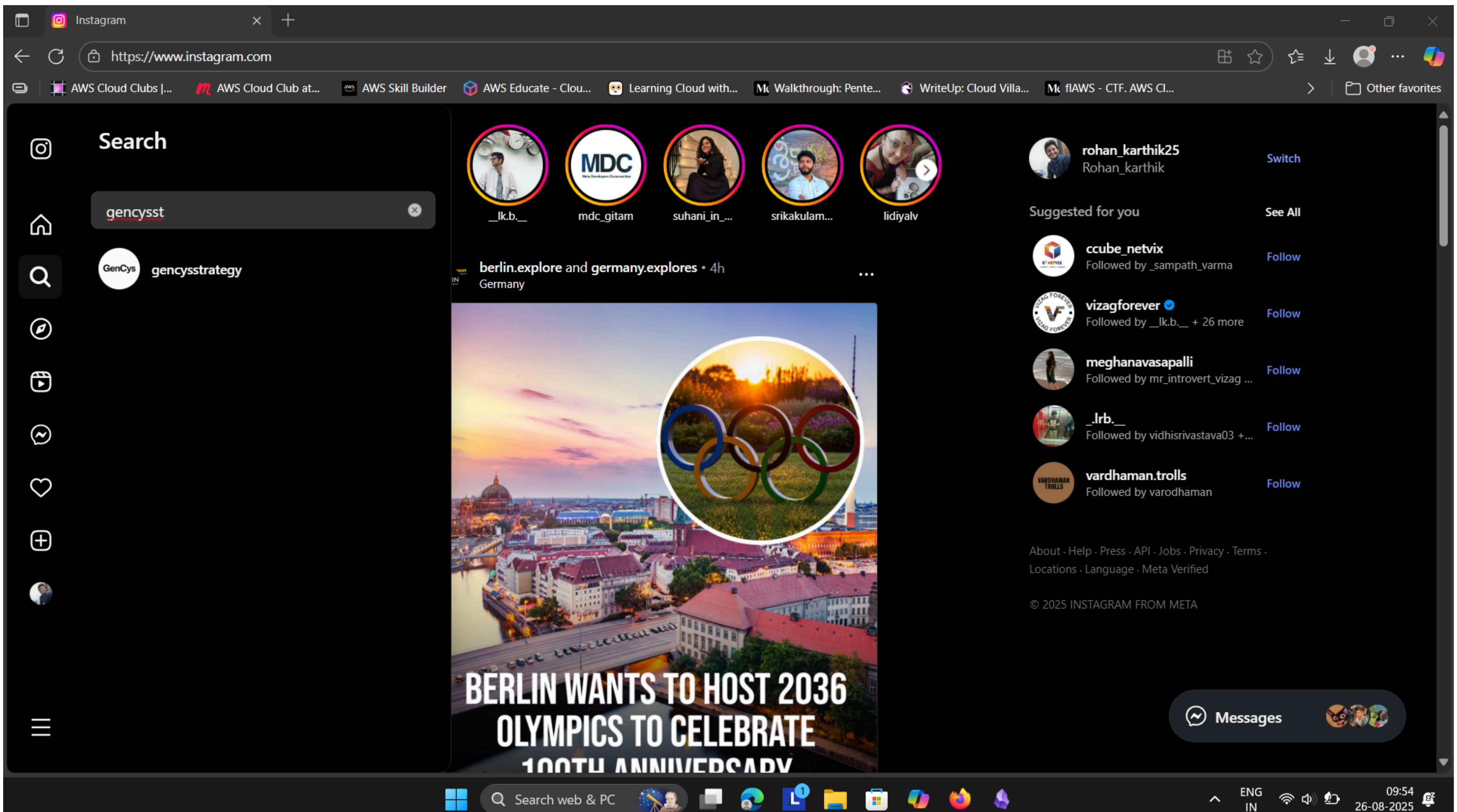
Points : 300

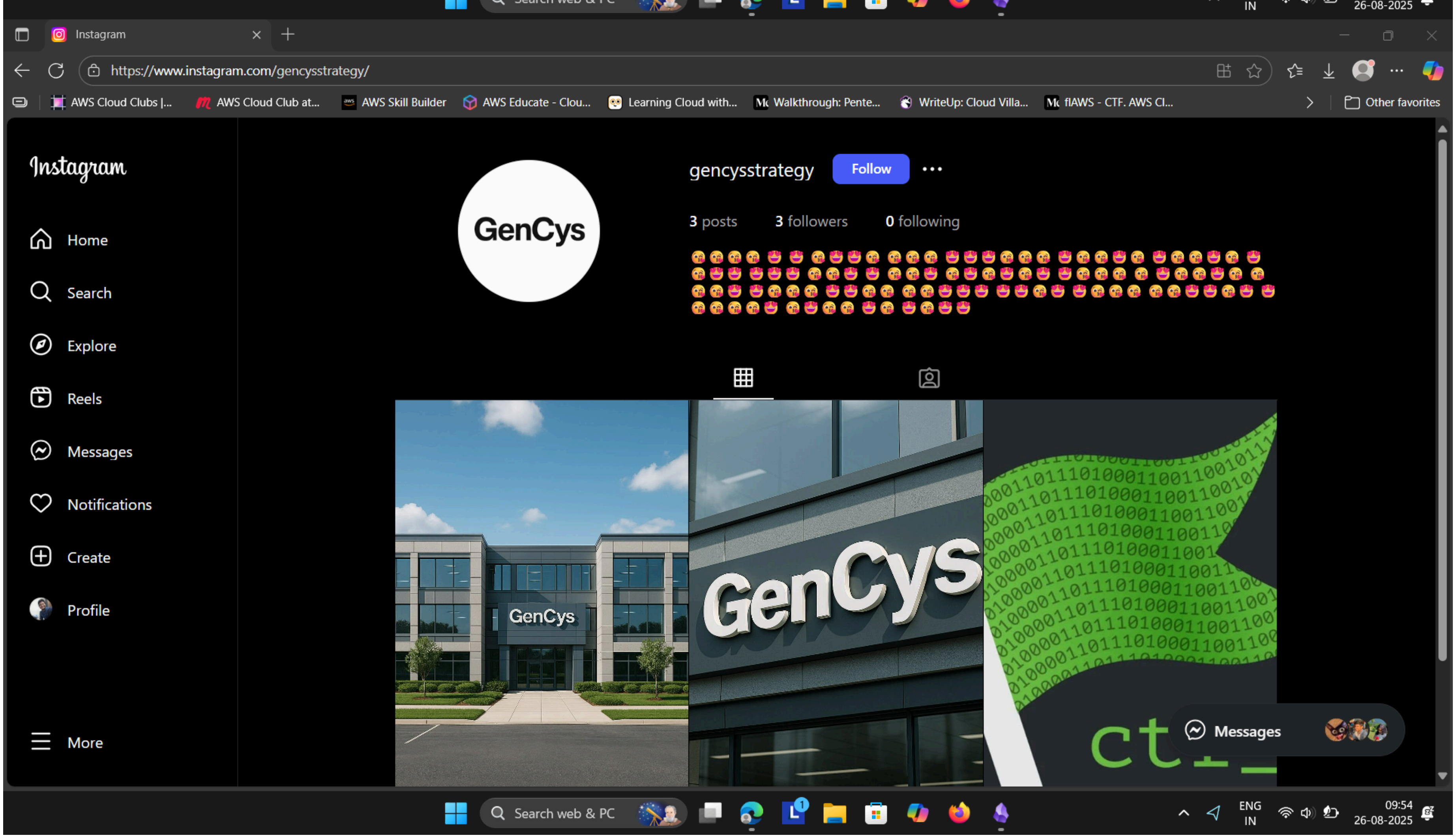
Description: Where Am I? osint A data leak by employee has been associated with gencys corperation With a online post , need to check with digital footprints and get the hidden flag

solution:

There was a Hint given with a keyword "gencysstatergy"
When searching this keyword in several social media platform
1.linkedIn
2.BlueSky
3.Instagram

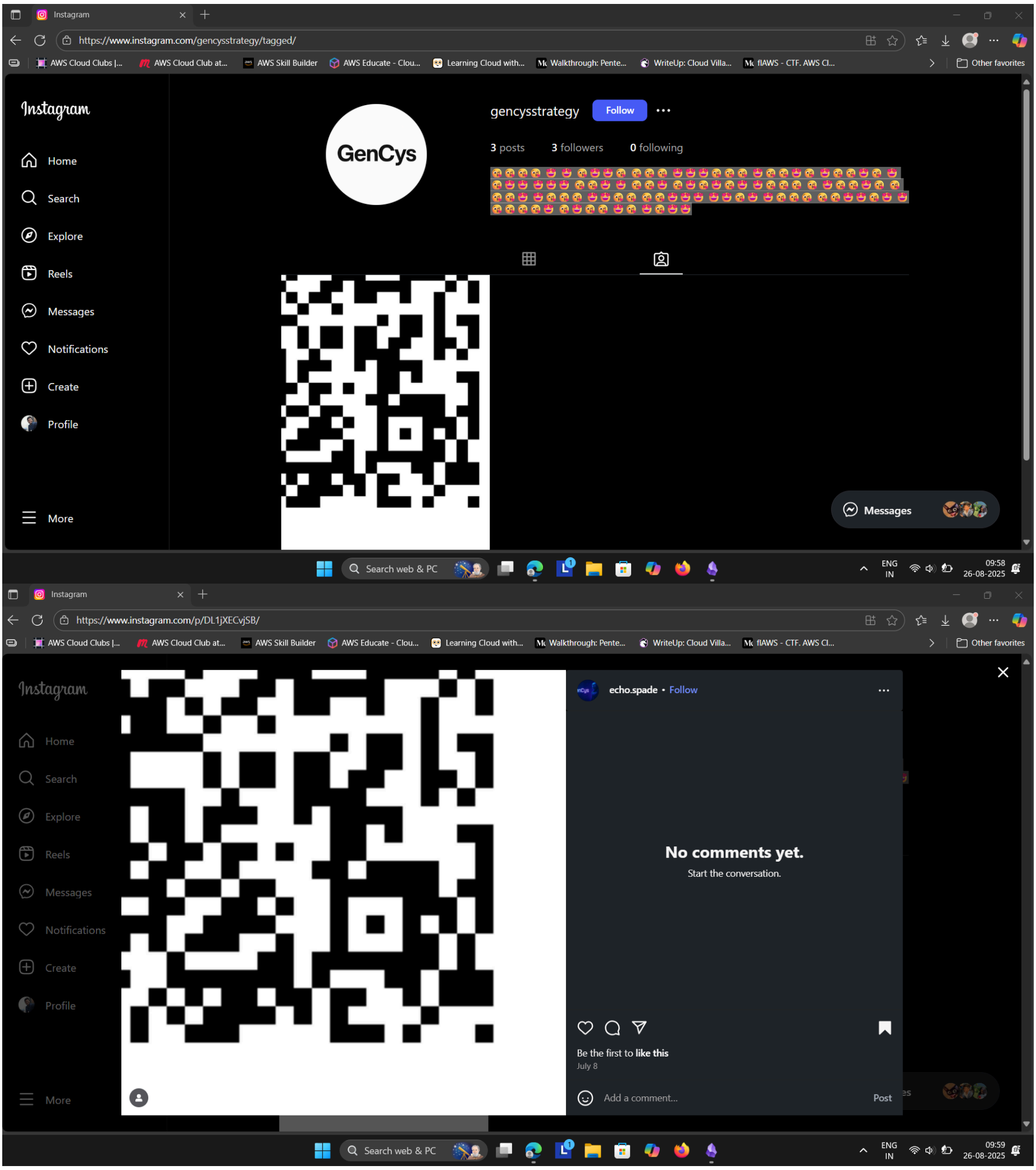
With Instagram we got a lead with "gencysstrategy"





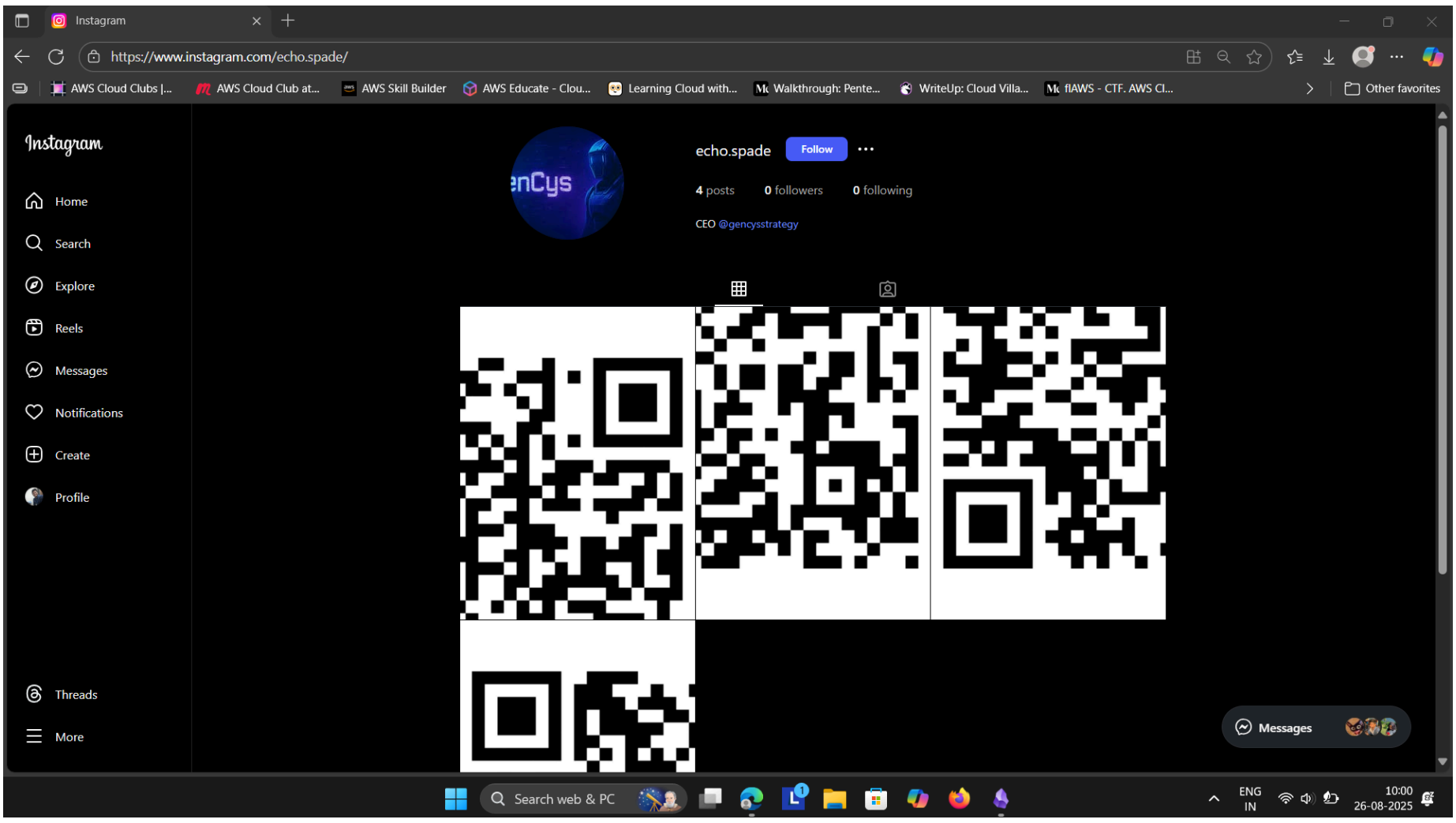
When seen into the account some suspicious emojis

😘😘😘😘 😍 😍 😘😍😍😘 😘😘😘 😍😍😍😘😘😘 😍😘😘😍😘 😍😘😘😍😘 😍😘😍😍 😍😍😍 😘😘😍
😍 😘😘😍 😘😍😘😍😘😍 😍😘😘😘 😘 😍😘😘😍😘 😘😘😘😍 😍😘😘😘 😍😍😘😘 😘😘😍😍😍 😍😍
😘😍 😍😘😘😘 😘😘😍😍😘😍 😍😘😘😘😘😍 😘😍😘😘 😍😘 😍😘😍😍



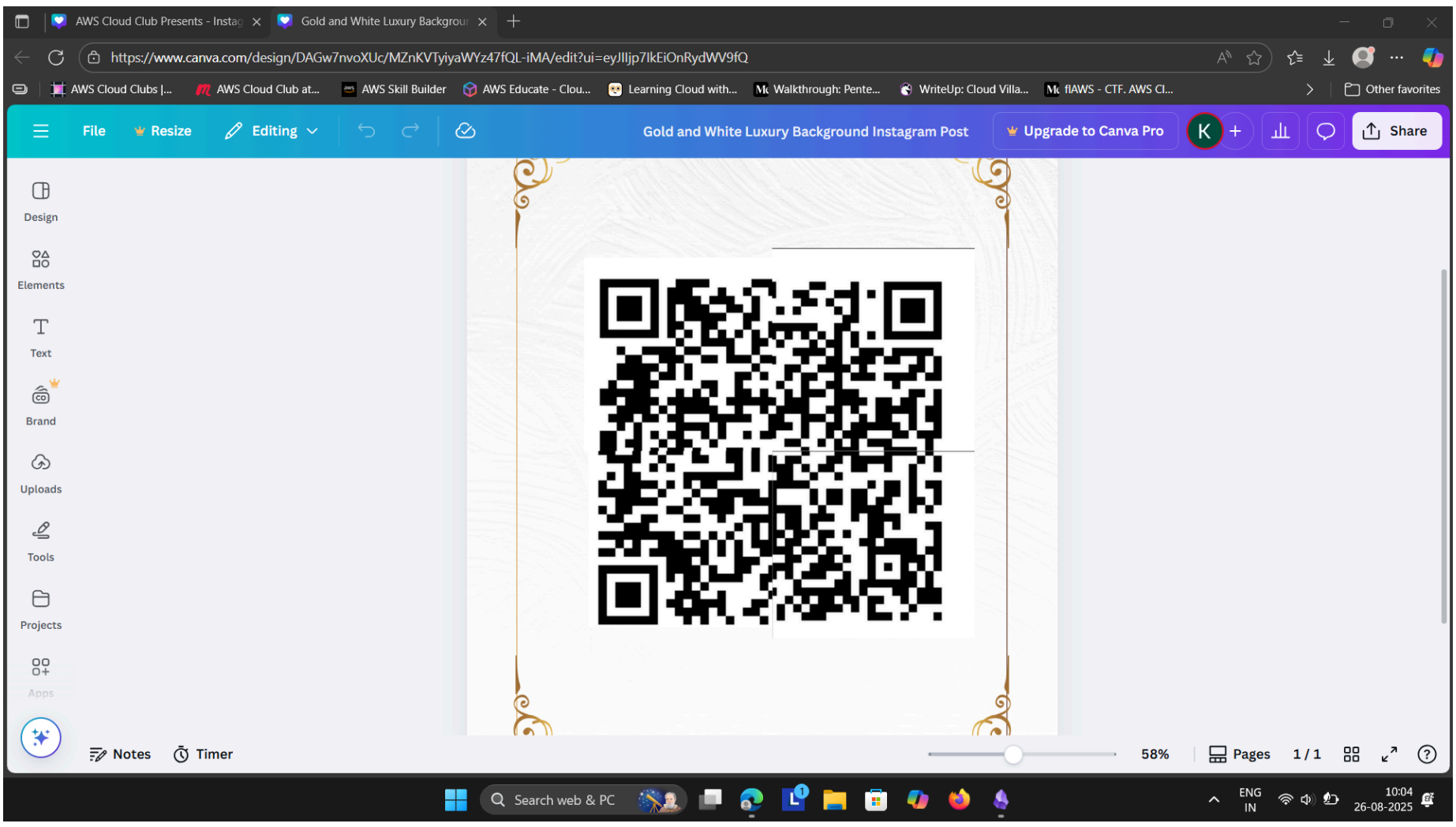
In tagged section we can see a QR which can say it is lead

the account was: echo.spade (https://www.instagram.com/echo.spade/)



Yup Hurray we have the qr's and also mentions the other first account of emojis CEO https://www.instagram.com/gencysstrategy/

Coping all the qr's into a platform like canva to form a single qr to scan for further extraction



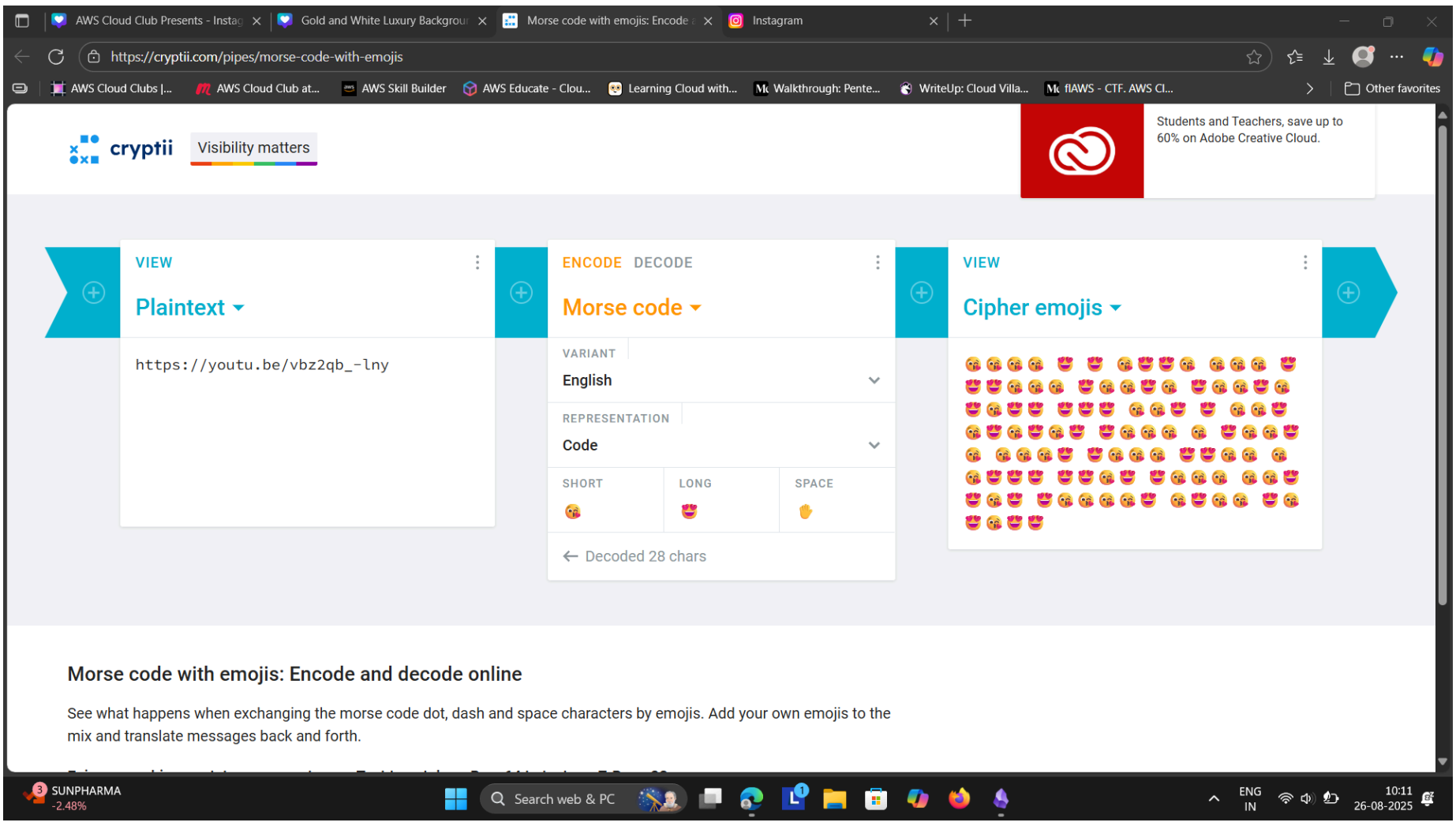
Upon scanning the qr we can got a link Morse code with emojis: Encode and decode online - cryptii

we can see it is a Morse code with emojis encoder and decoder

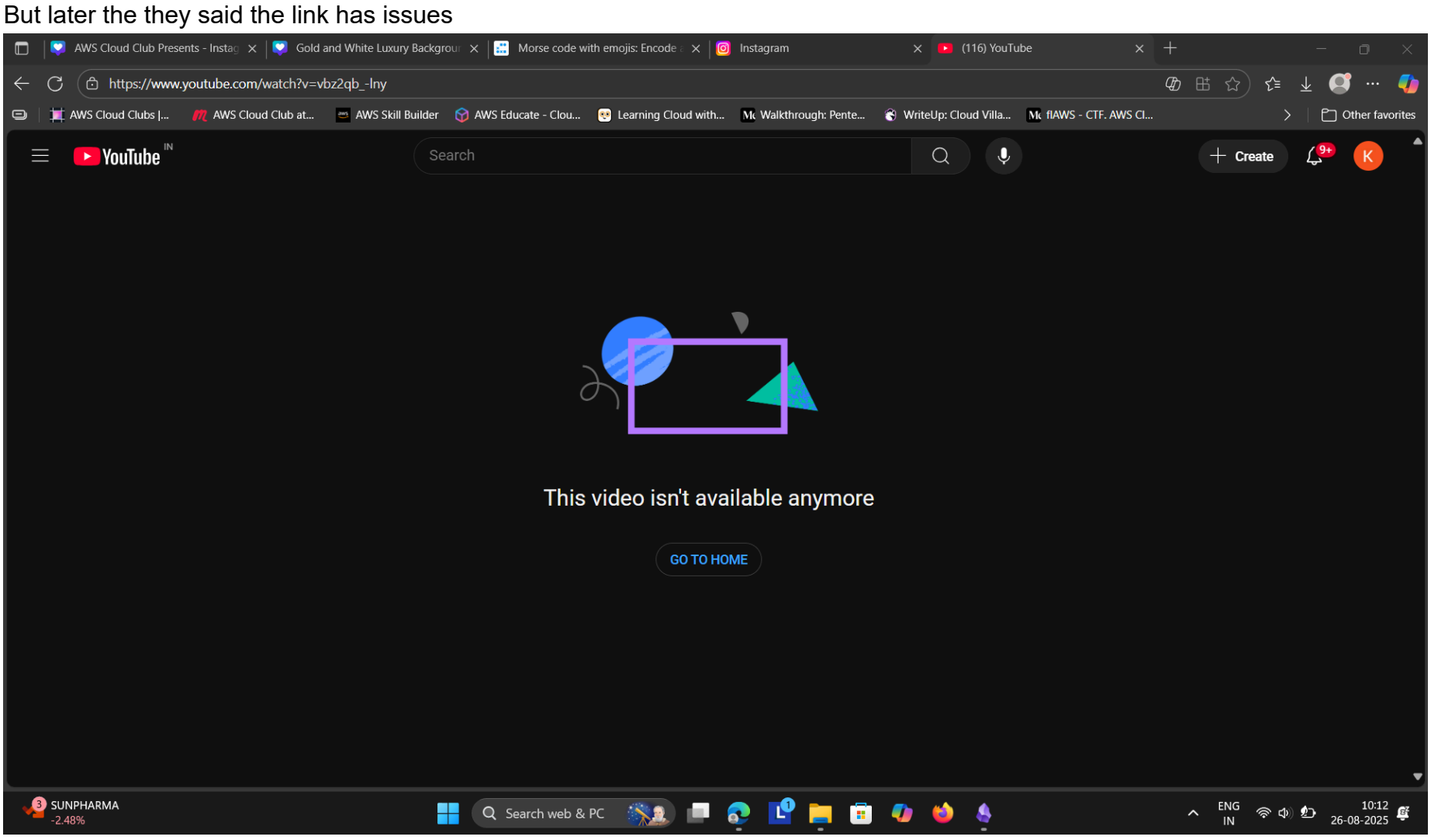
we remembered of getting a emojis in the insta account of gencysstratergy account :
Instagram

😘😘😘😘😍😍😘😍😍😘😘😘😘😍😍😍😘😘😘😍😘😘😍😘😍😘😘😍😘😍😘😍😍😍😍😘😘😍
😍😘😘😍😘😍😘😍😘😍😍😘😘😘😘😍😘😘😍😘😘😘😘😍😍😘😘😘😍😍😘😘😘😘😍😍😍😍😍
😘😍😍😘😘😘😘😘😍😍😘😍😍😘😘😘😘😍😘😍😍😘😘😍😘😍😘😍😍

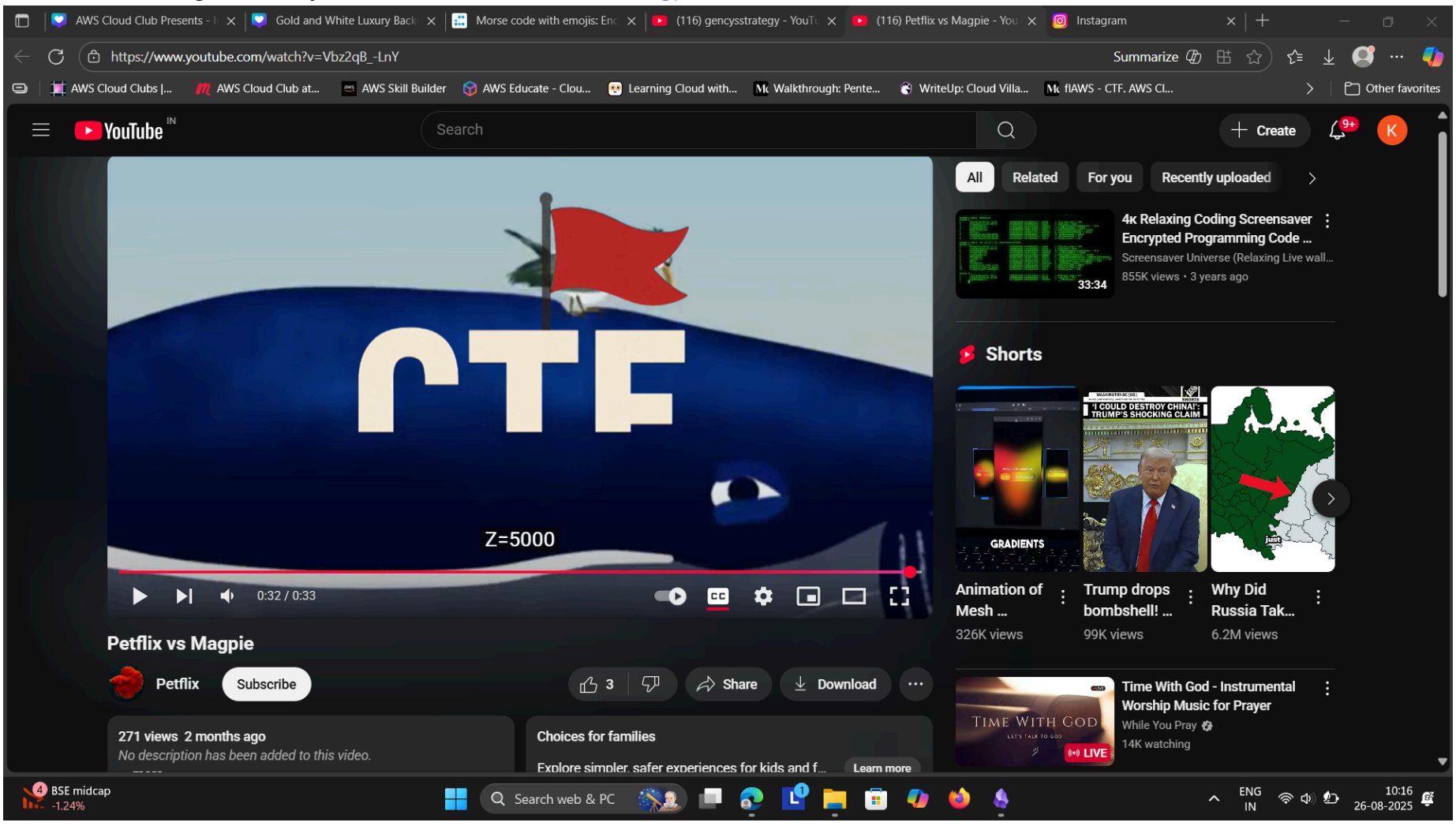
trying with these can result us a youtube link : https://youtu.be/vbz2qb_-lny



But later the they said the link has issues

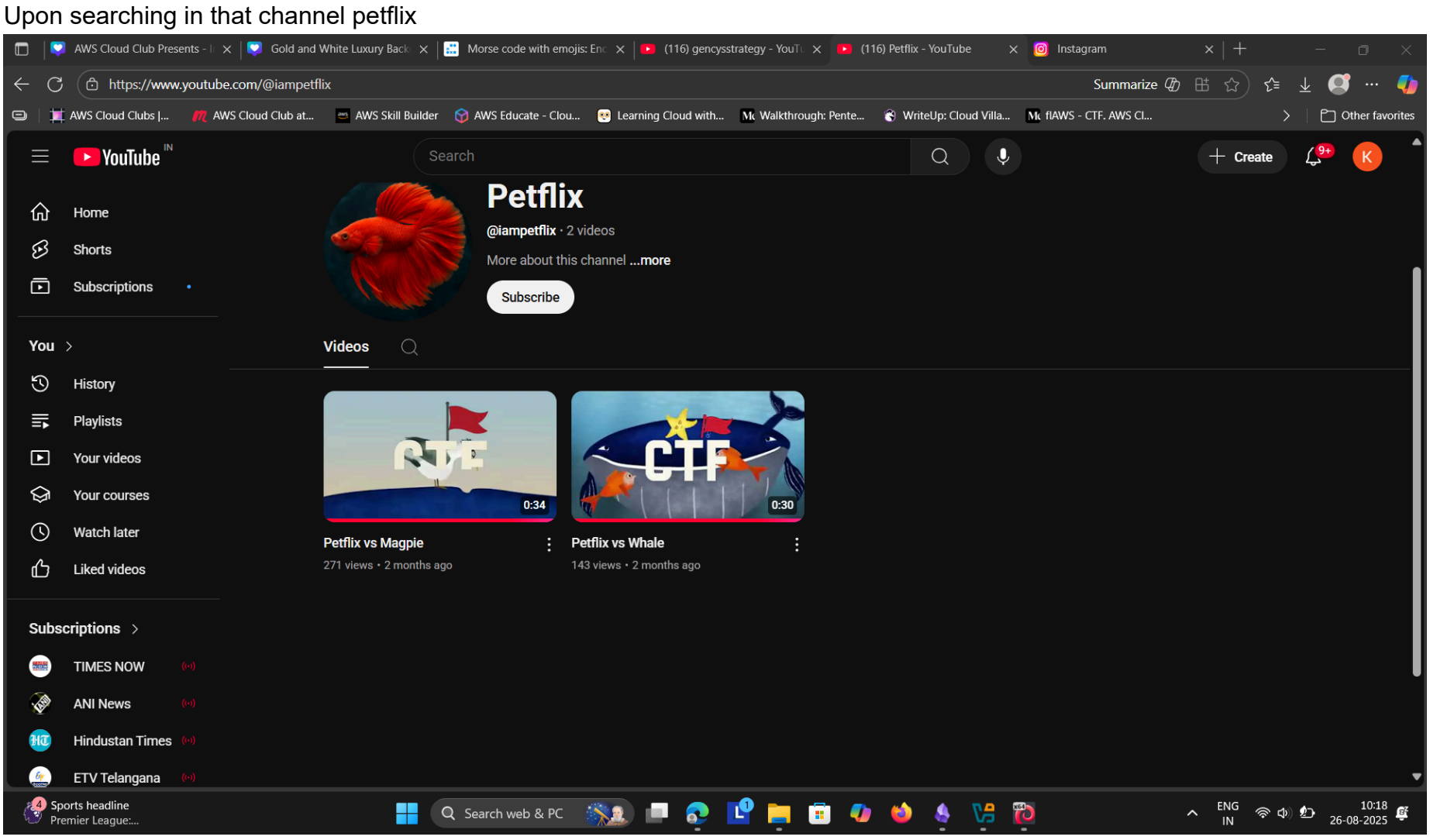


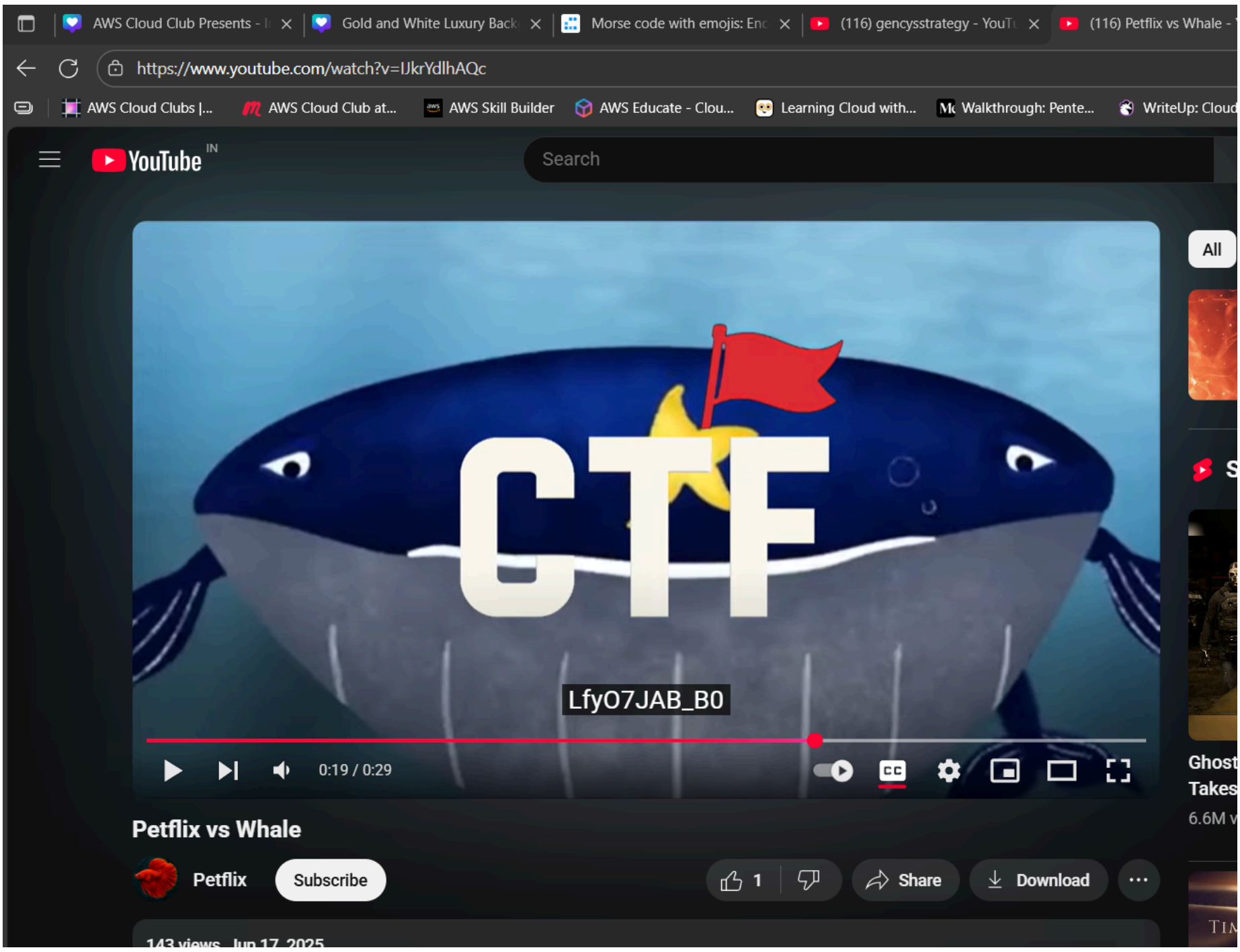
Later a Hint was given to a youtube channel : [Petflix vs Magpie]



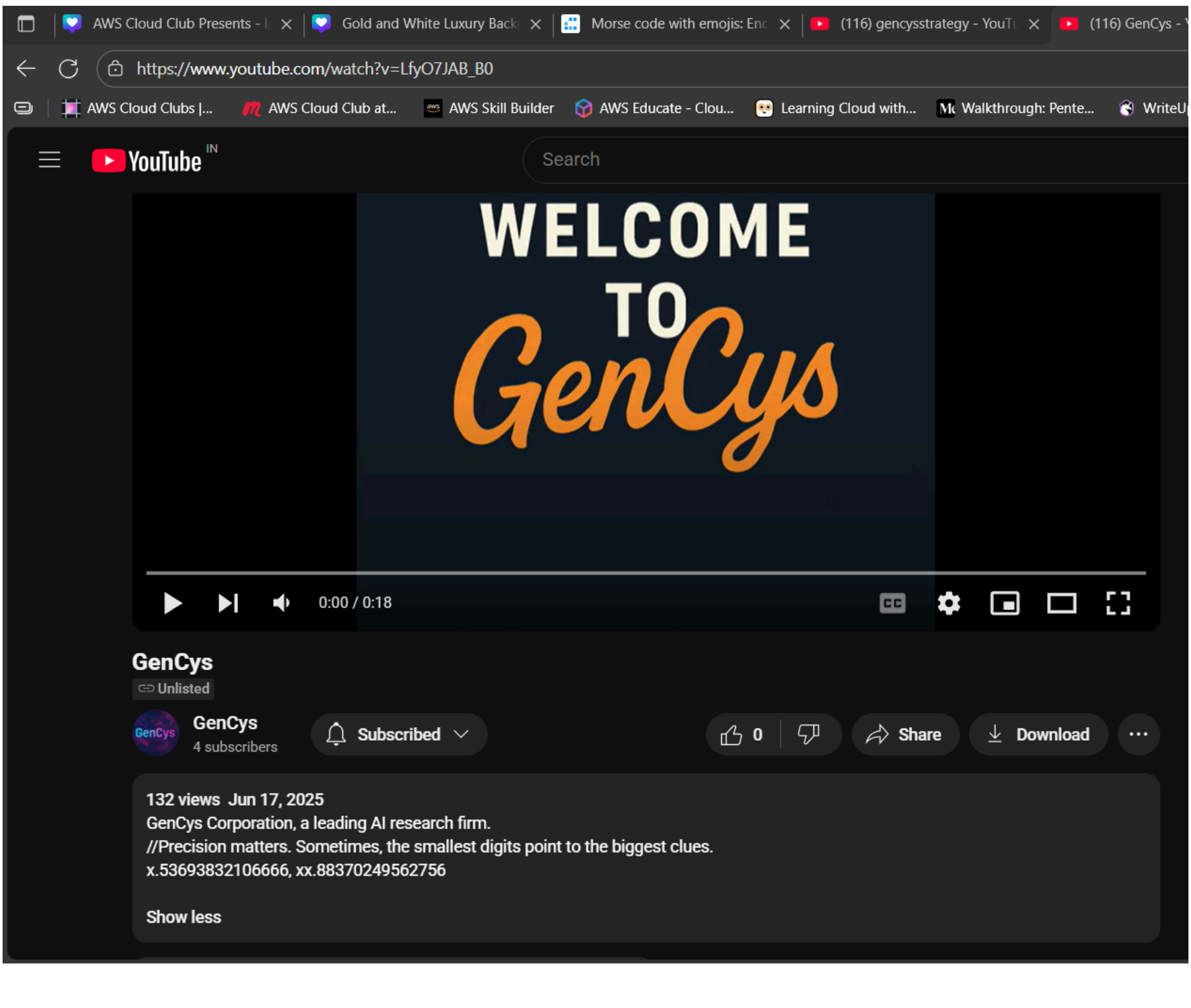
We have seen "Z=5000" in Caption

Upon searching in that channel petflix



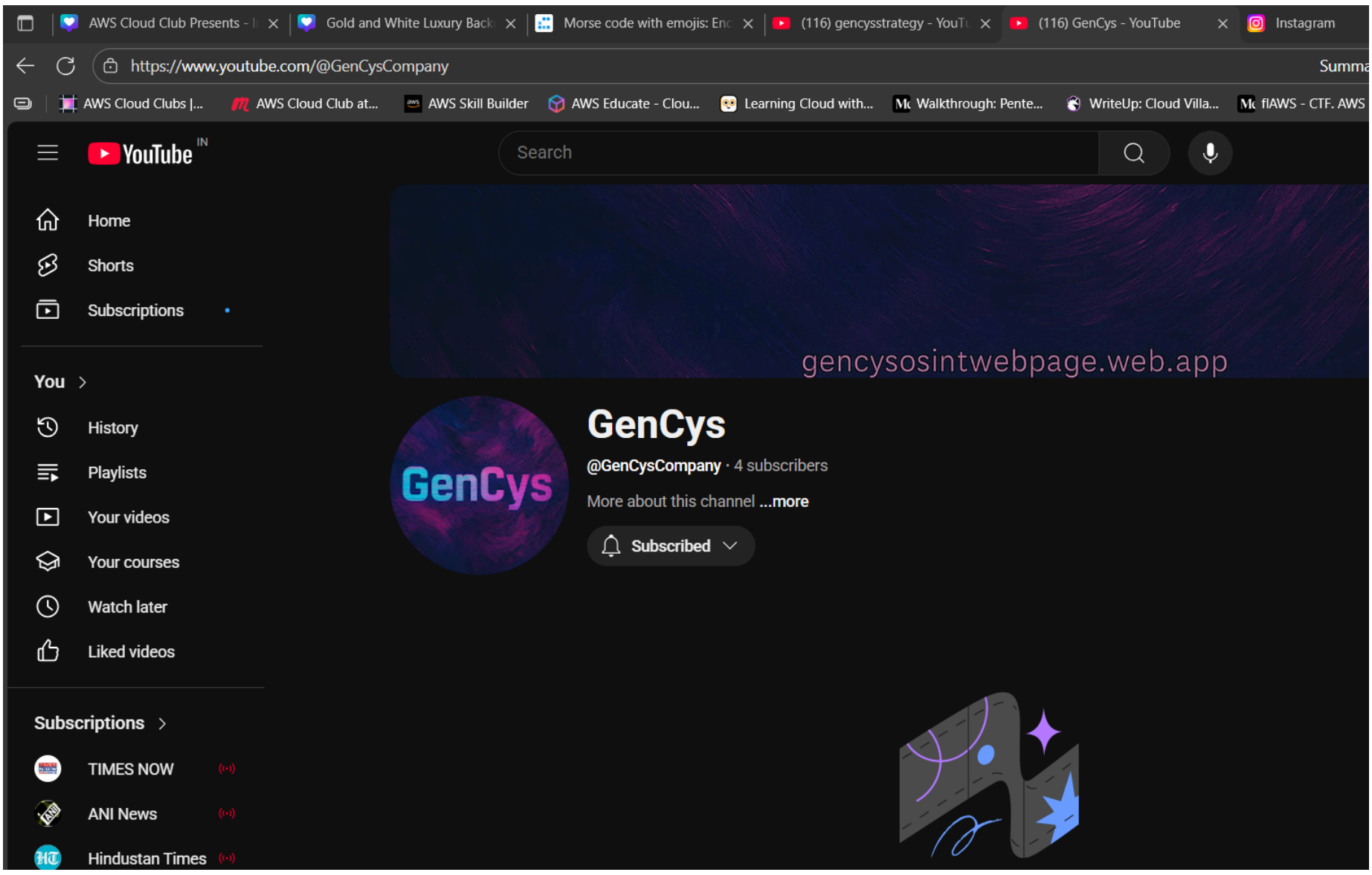


In this second video we can see a code in captions : LfyO7JAB_B0
when copying and trying that to check if it a youtube link value we are redirected to below link
[(116) GenCys - YouTube]

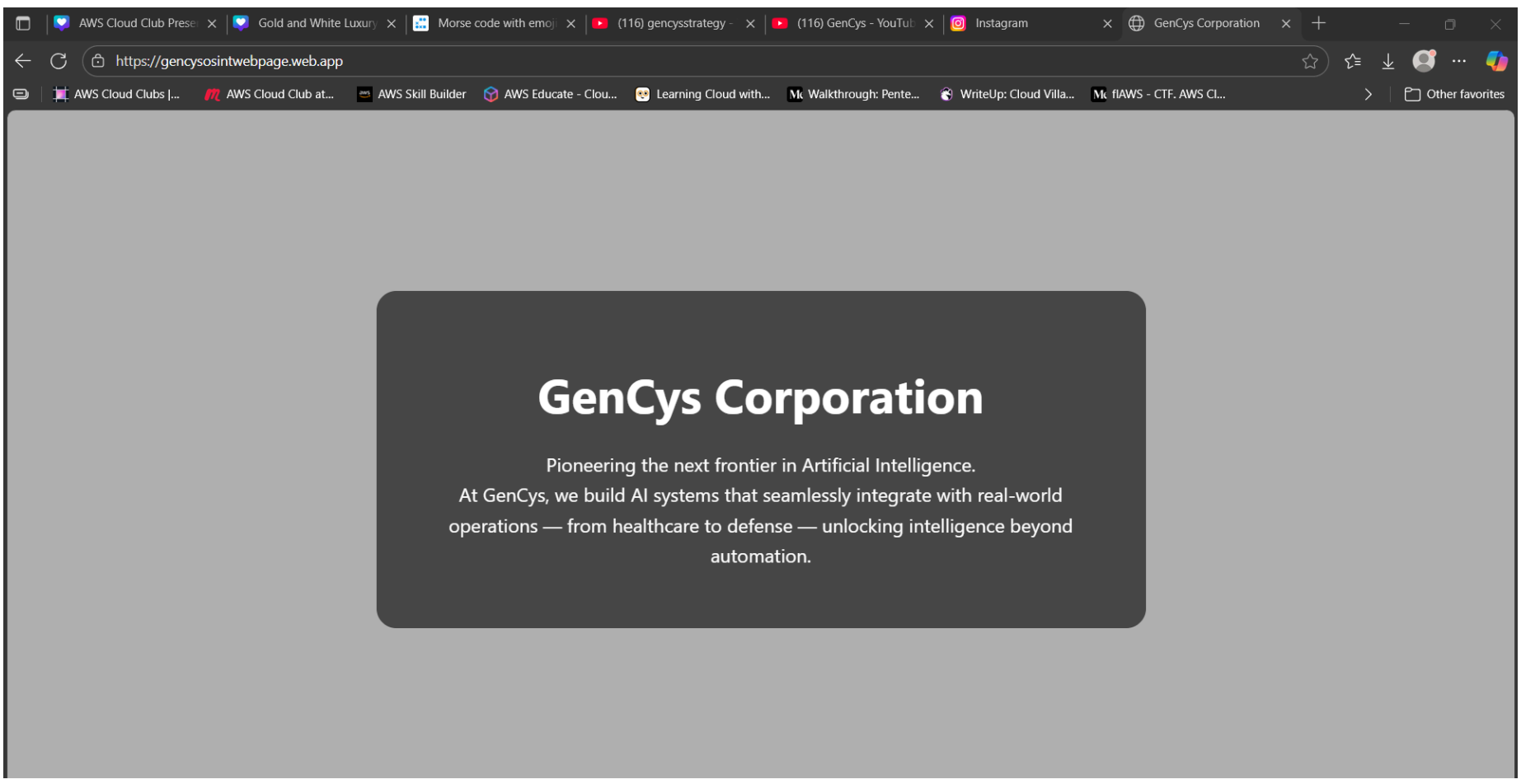


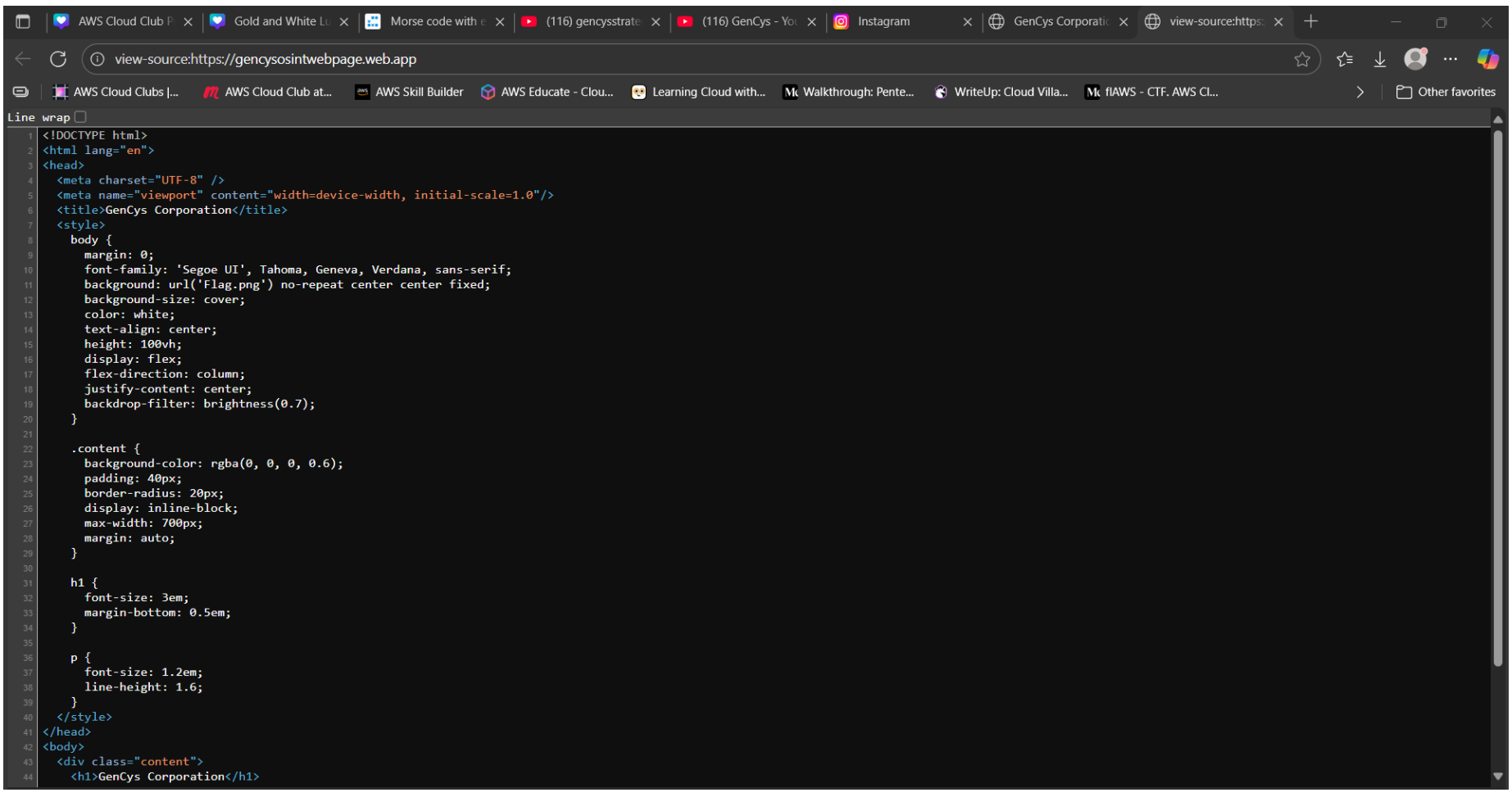
we can see a small info in the description
GenCys Corporation, a leading AI research firm. //Precision matters. Sometimes, the smallest digits point to the biggest clues.
x.53693832106666, xx.88370249562756

these "x.53693832106666, xx.88370249562756" look like coordinates



when checking with that channel we got a web link "gencysosintwebpage.web.app"



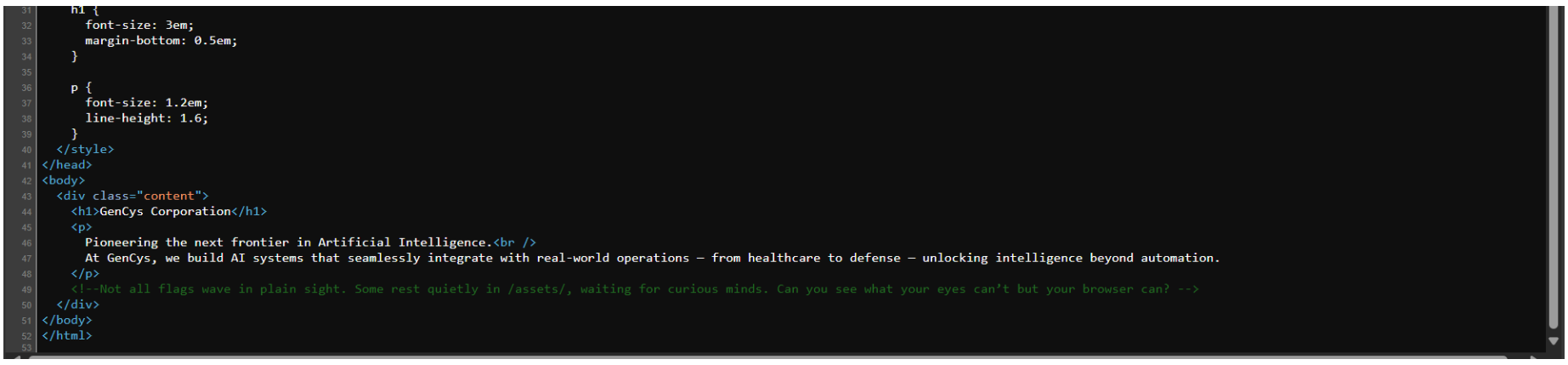


After checing the page source code we can see a lead stating that

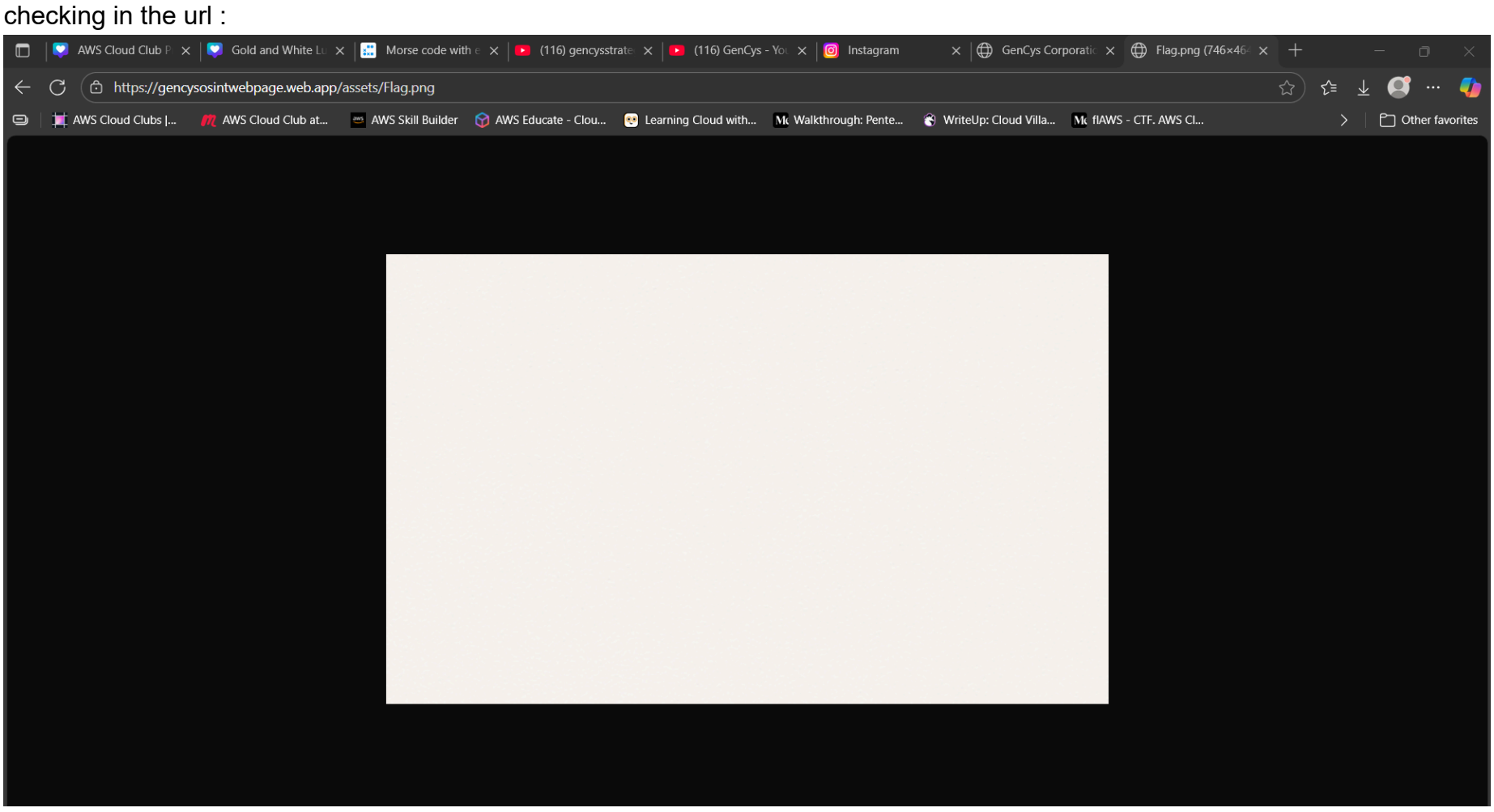
background: url('Flag.png') no-repeat center center fixed;

and

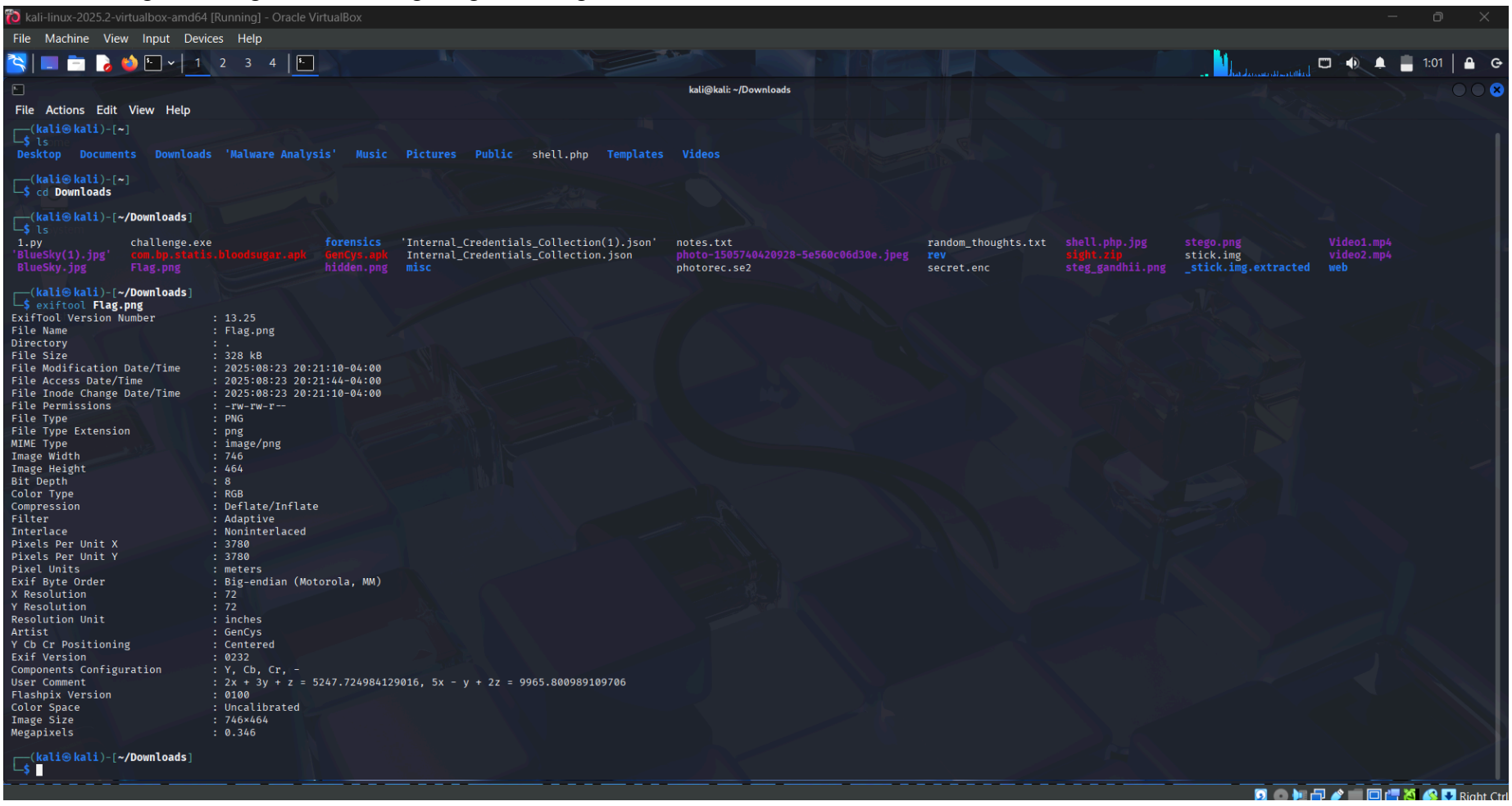
""



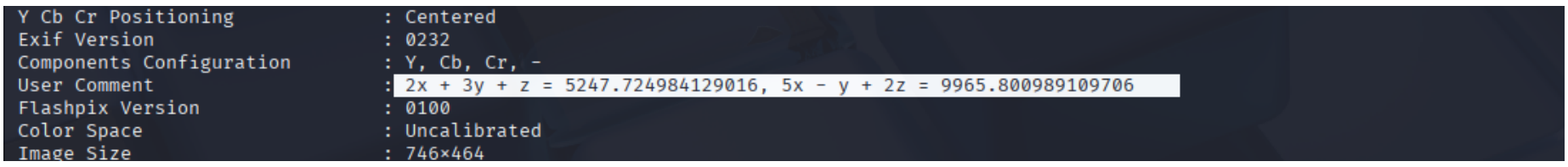
checking in the url :



Downloading the image and investigating the image with exiftool



2x + 3y + z = 5247.724984129016, 5x - y + 2z = 9965.800989109706



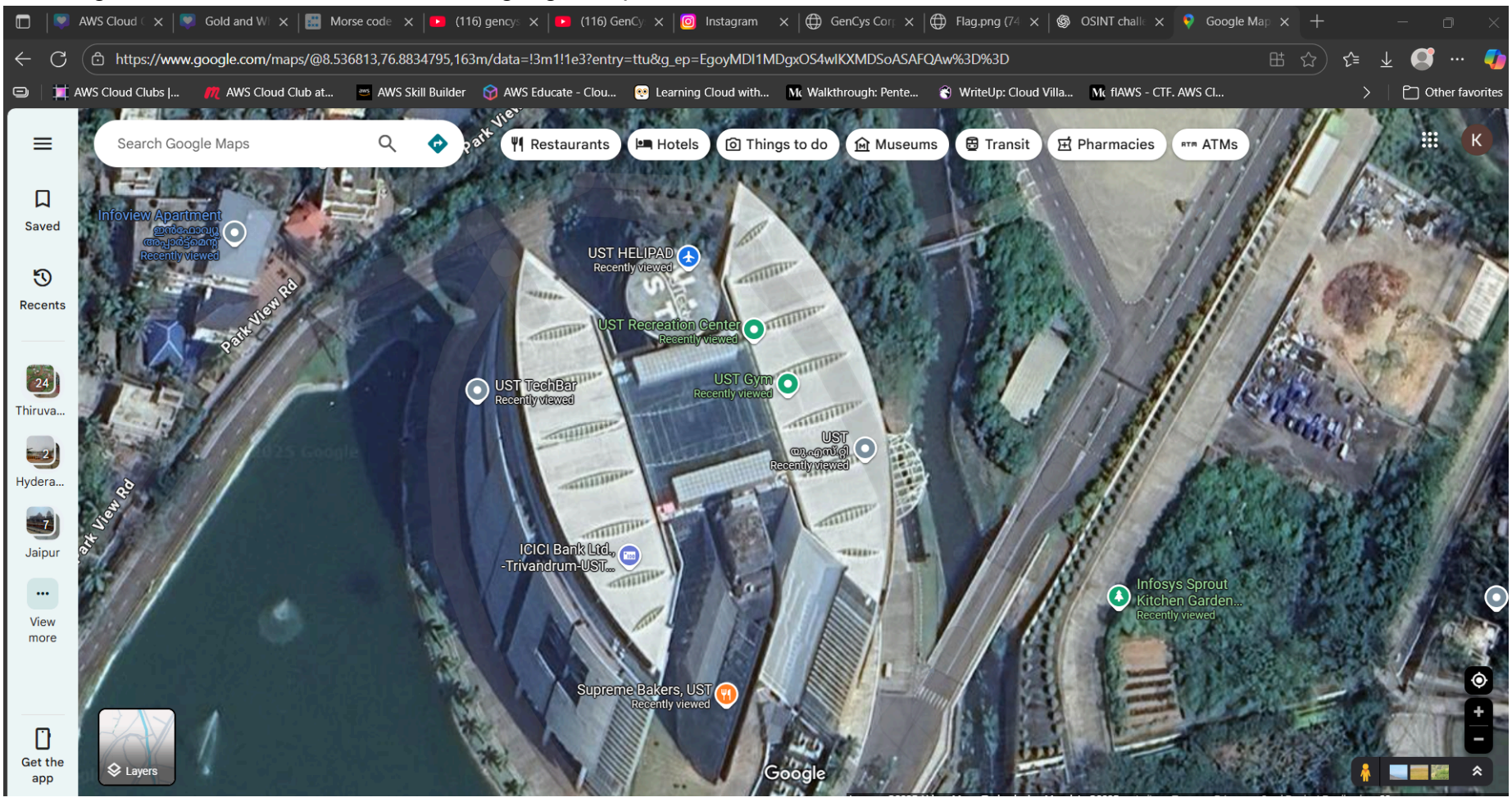
we remember a value prev Z = 5000 https://www.youtube.com/watch?v=Vbz2qB_-LnY

2x + 3y + 5000 = 5247.72498412901
5x - y + 2(5000) = 9965.800989109706
Solving the equation

8.53693832106666
76.88370249562756
this is the coordinates we got
according to these x.53693832106666, xx.88370249562756

x.53693832106666 is 8.53693832106666
xx.88370249562756 is 76.88370249562756

checking with this directs to UST Global in google maps



UST

Write a review

II : 9th Floor, UST Campus, TechnoPark Phase II, Electronics Technology Parks SEZ, Kulathoor, Thiruvananthapuram, Kerala 695583, India

4.6 ★★★★½ 3,524 reviews ⓘ

Mohd Ashraf Hisham
5 reviews · 28 photos

★★★★★ Edited 3 days ago **NEW**

Really loved visiting this place! Peaceful and quiet. Also stumbled across something curious here — felt almost like a min CTF puzzle: Z2l0aHViLmNvbS9Ob3RTb0lubm9jZW50

upon decoding the this CTF Puzzle Z2l0aHViLmNvbS9Ob3RTb0lubm9jZW50

Decode from Base64 format

Simply enter your data then push the decode button.

Z2l0aHViLmNvbS9Ob3RTb0lubm9jZW50

For encoded binaries (like images, documents, etc.) use the file upload form a little further down on this page.

UTF-8 Source character set.

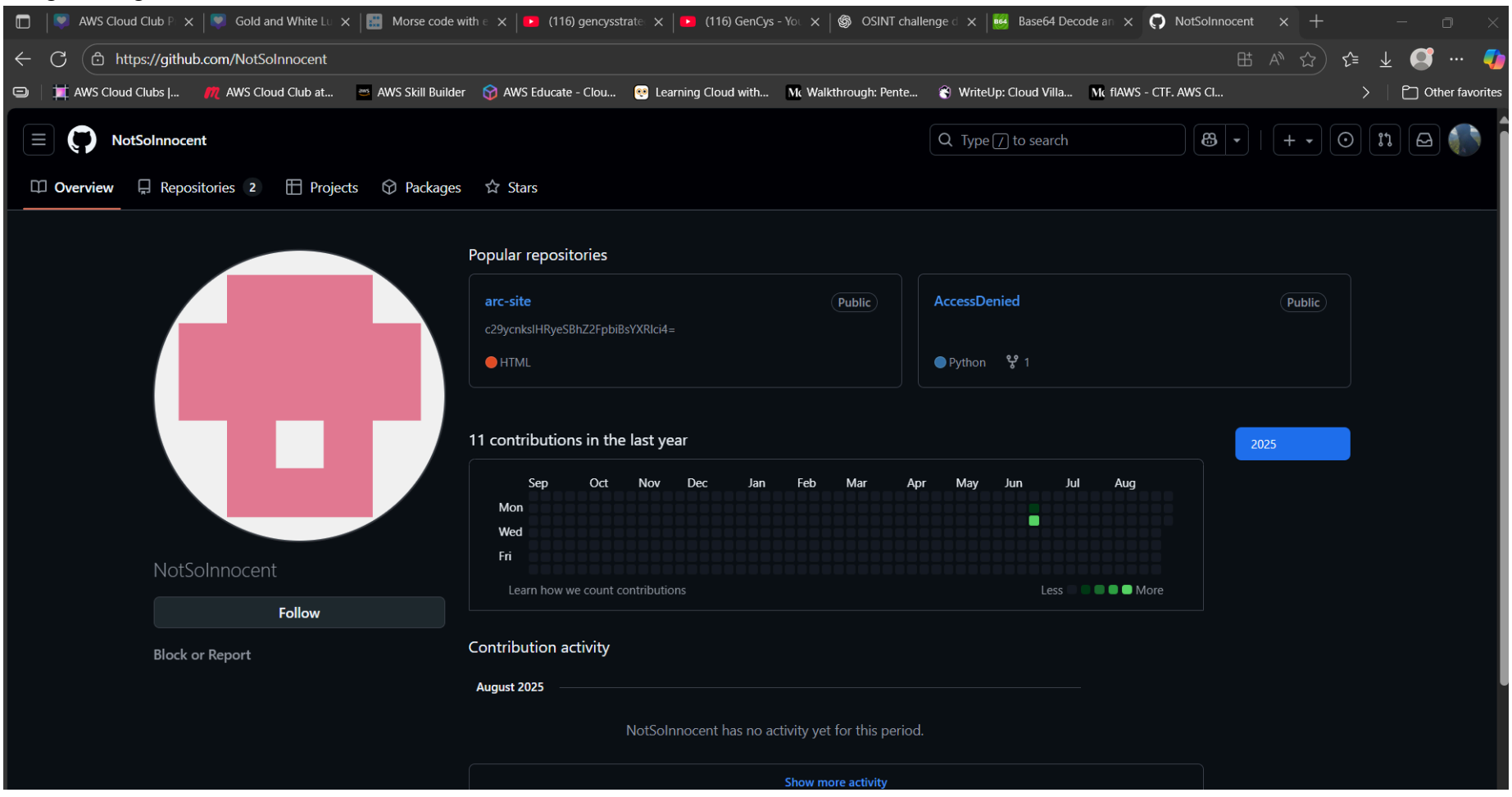
Decode each line separately (useful for when you have multiple entries).

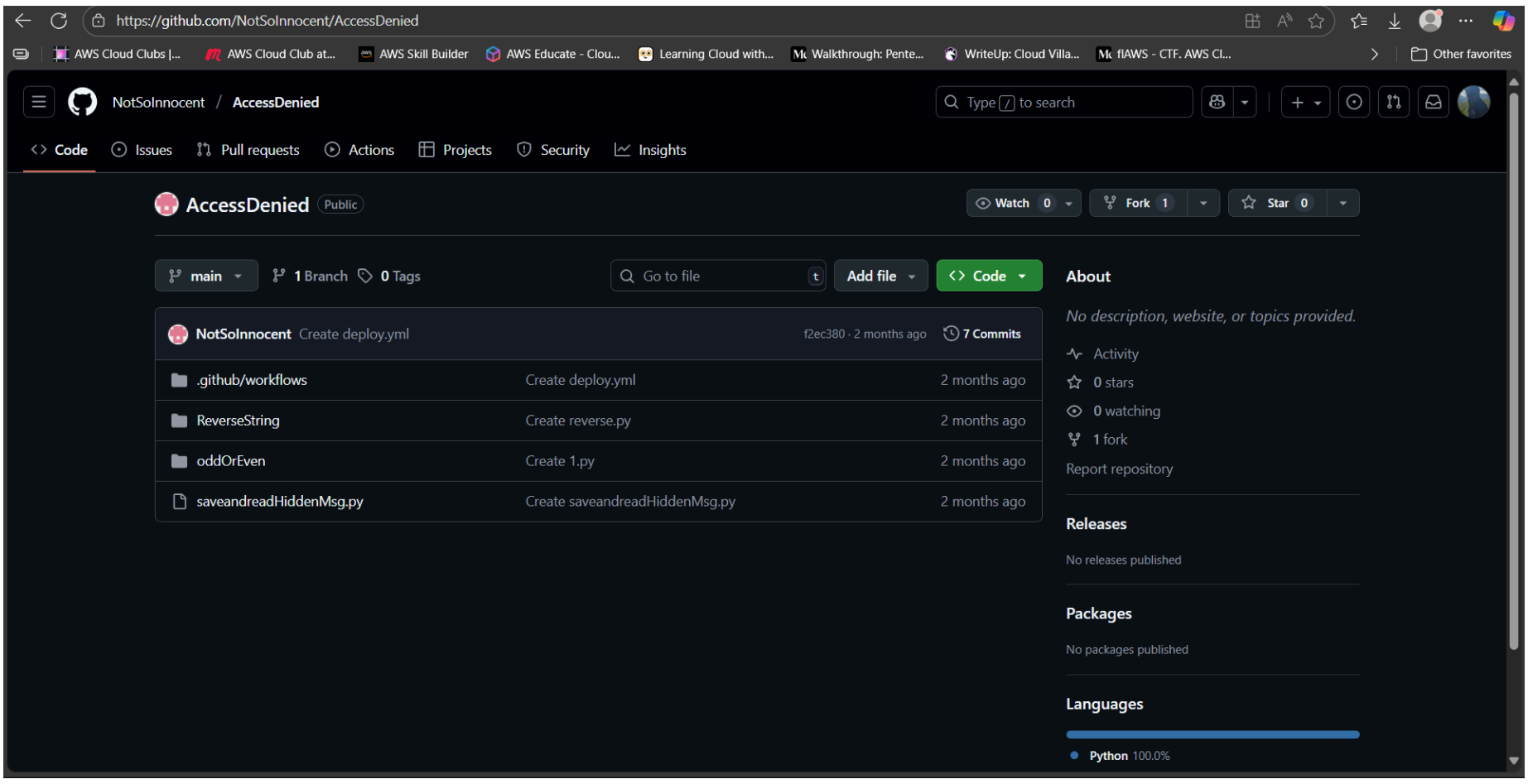
Live mode OFF Decodes in real-time as you type or paste (supports only the UTF-8 character set).

DECODE Decodes your data into the area below.

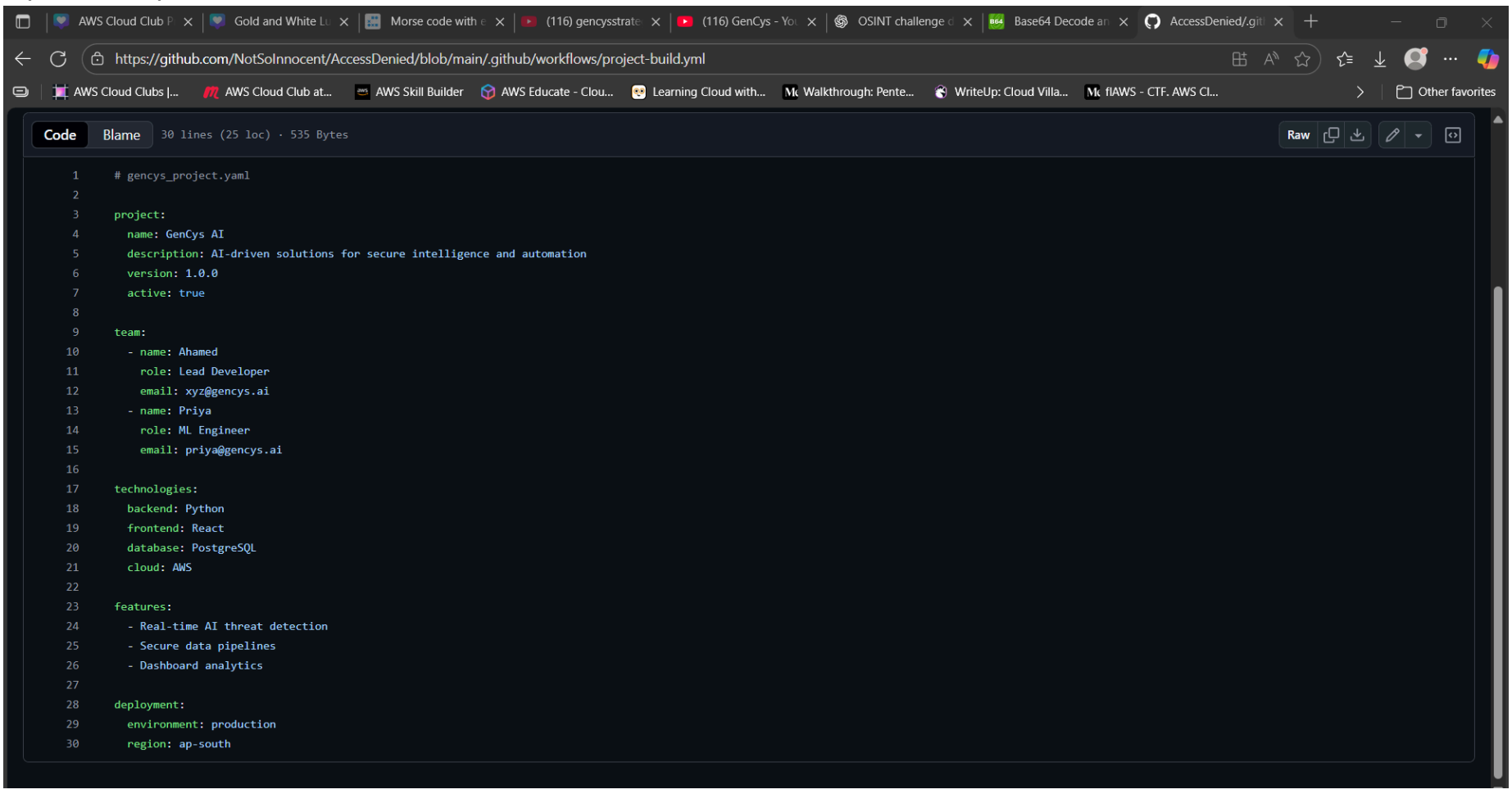
github.com/NotSoInnocent

we got the github account which is /NotSoInnocent

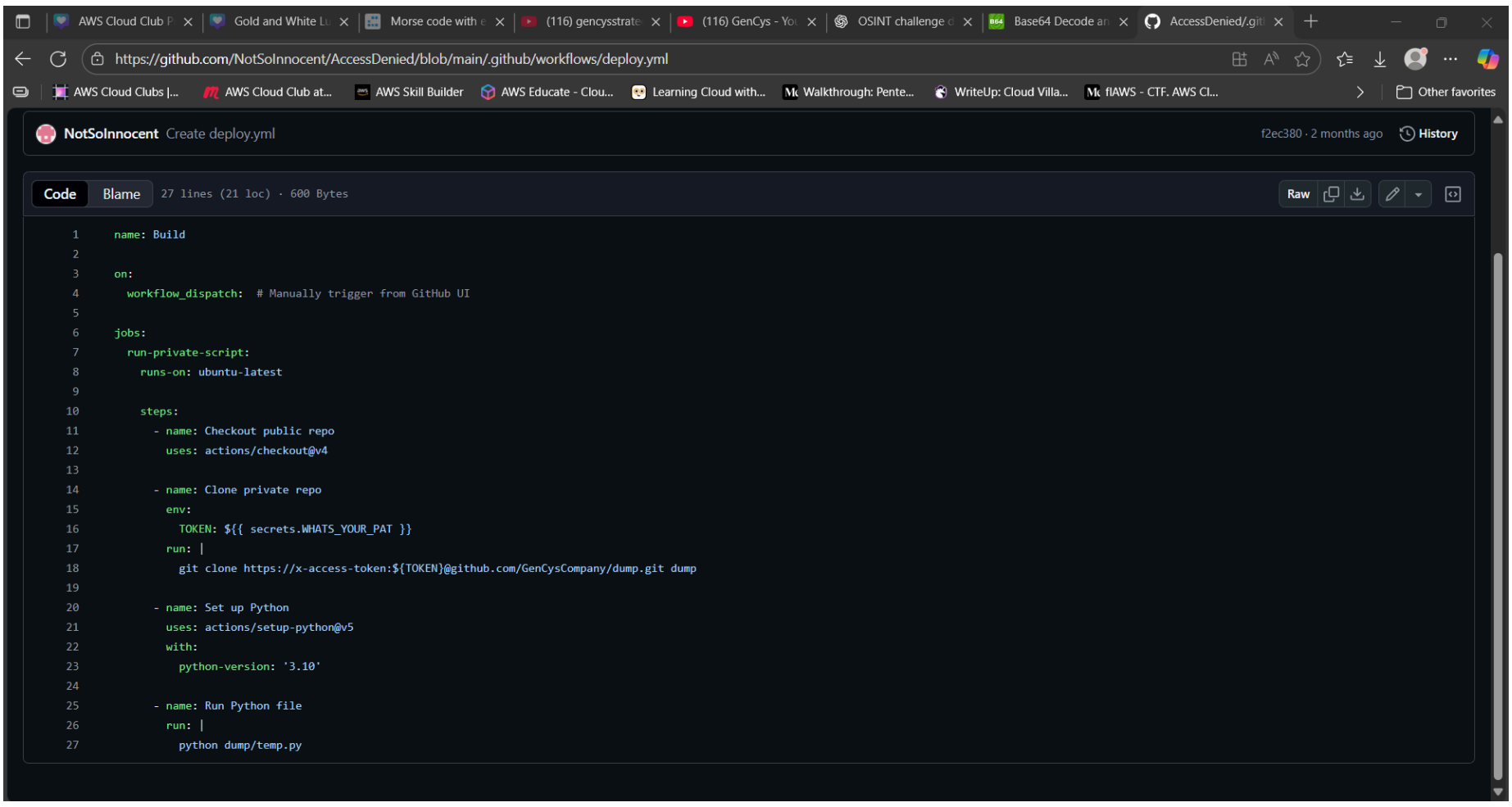




A potential repo [NotSoInnocent/AccessDenied]



[AccessDenied/.github/workflows/project-build.yml at main · NotSoInnocent/AccessDenied]

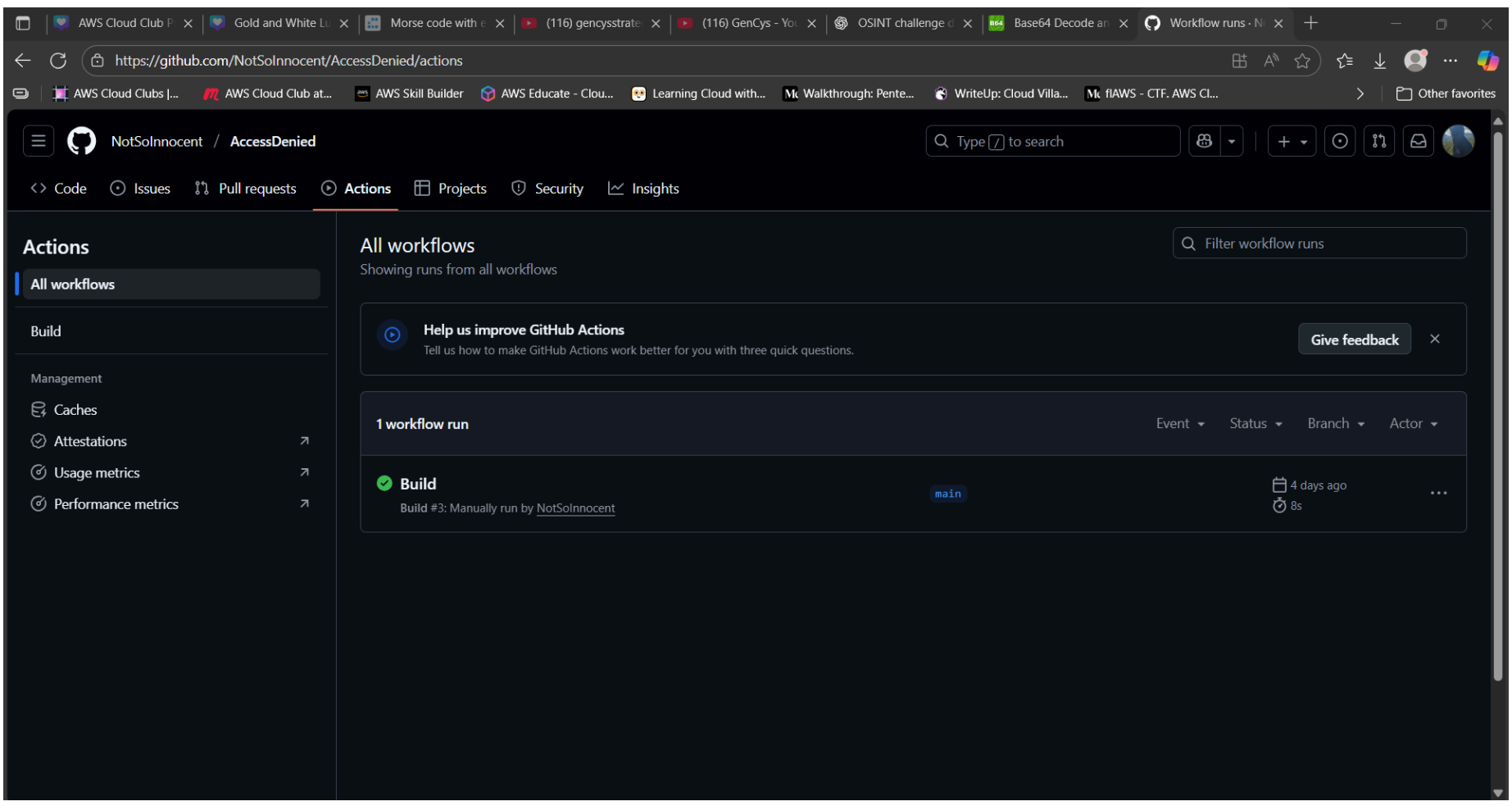


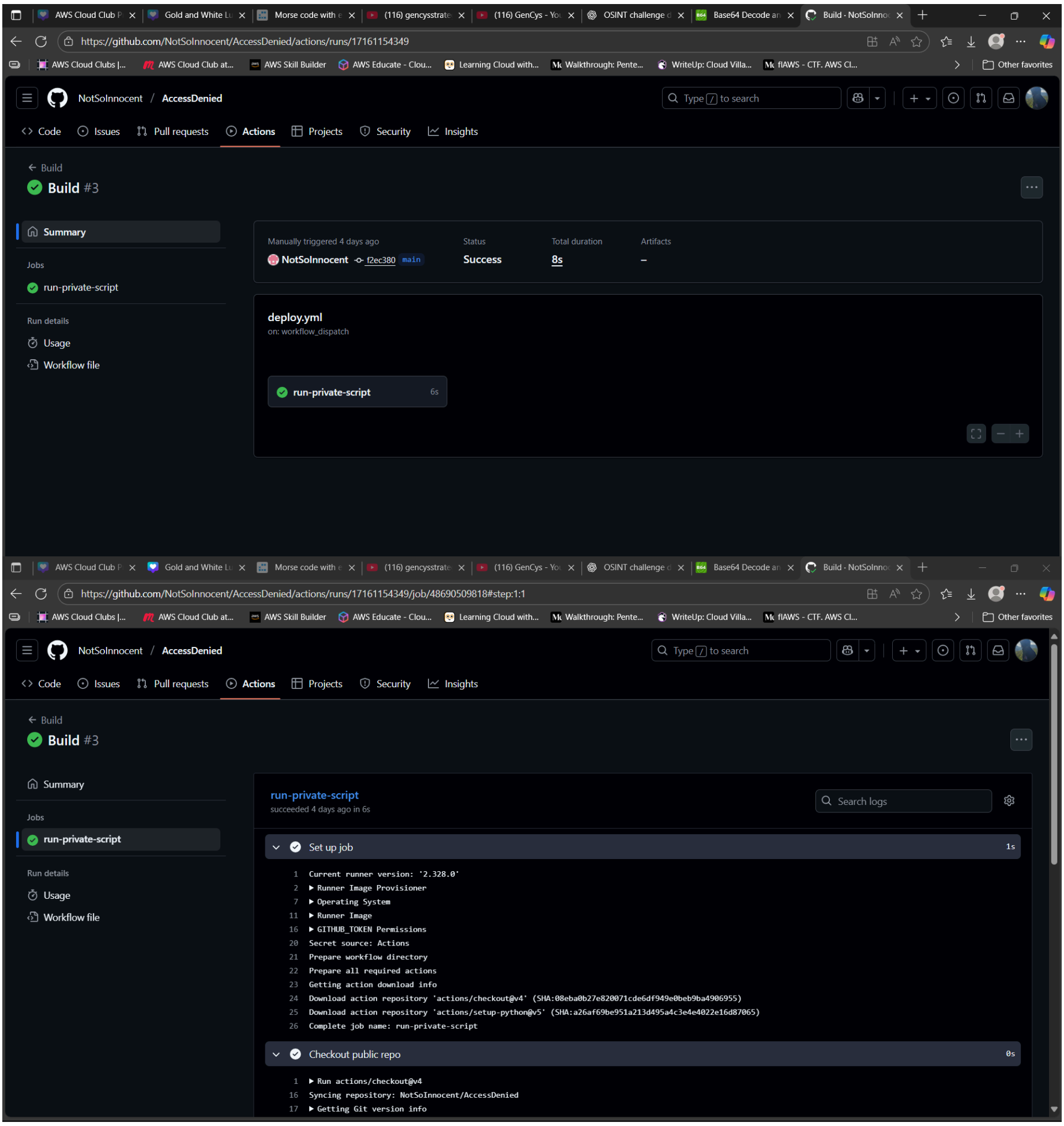
[AccessDenied/.github/workflows/deploy.yml at main · NotSoInnocent/AccessDenied]

we can see in the deploy.yml there are some steps given one among them was :
name: Checkout public repo
uses: actions/checkout@v4

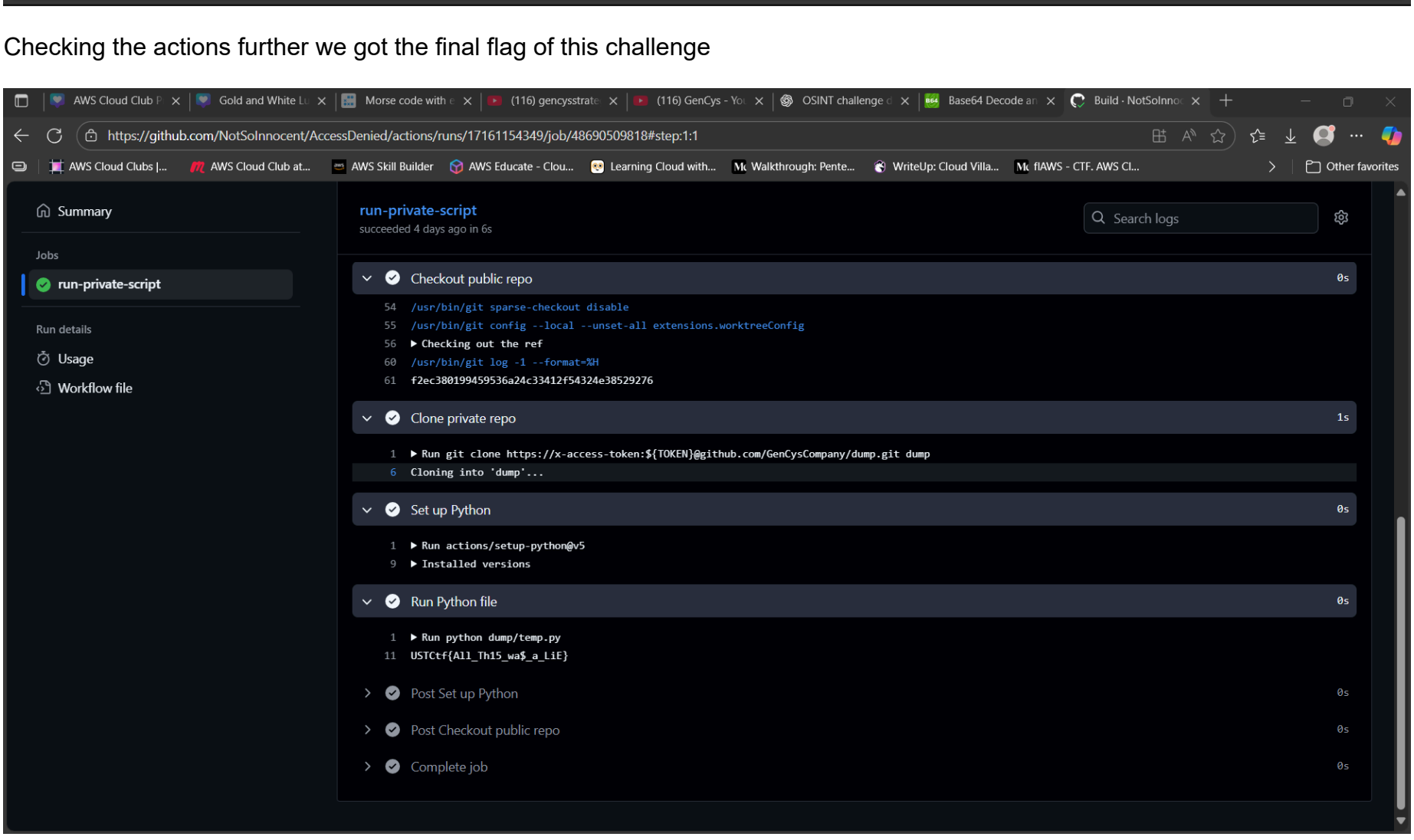
So we checked the actions

[Workflow runs · NotSoInnocent/AccessDenied]





Checking the actions further we got the final flag of this challenge



```
Run Python file                                                    0s
 1  ▶ Run python dump/temp.py
11  USTCtf{All_Th15_wa$_a_LiE}
```

Final Flag : "USTCtf{All_Th15_wa$____a_LiE}"

Physical OSINT

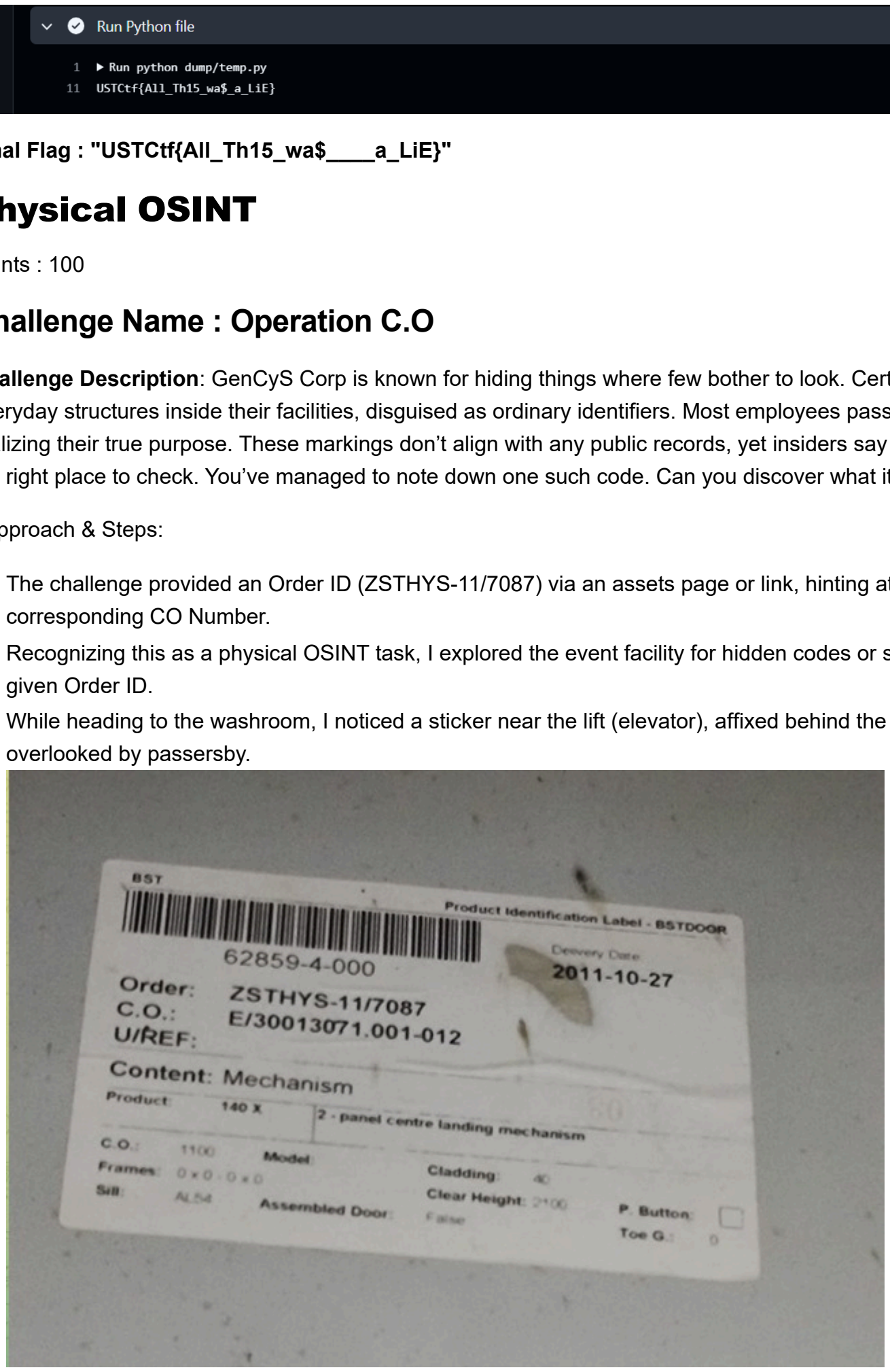
Points : 100

Challenge Name : Operation C.O

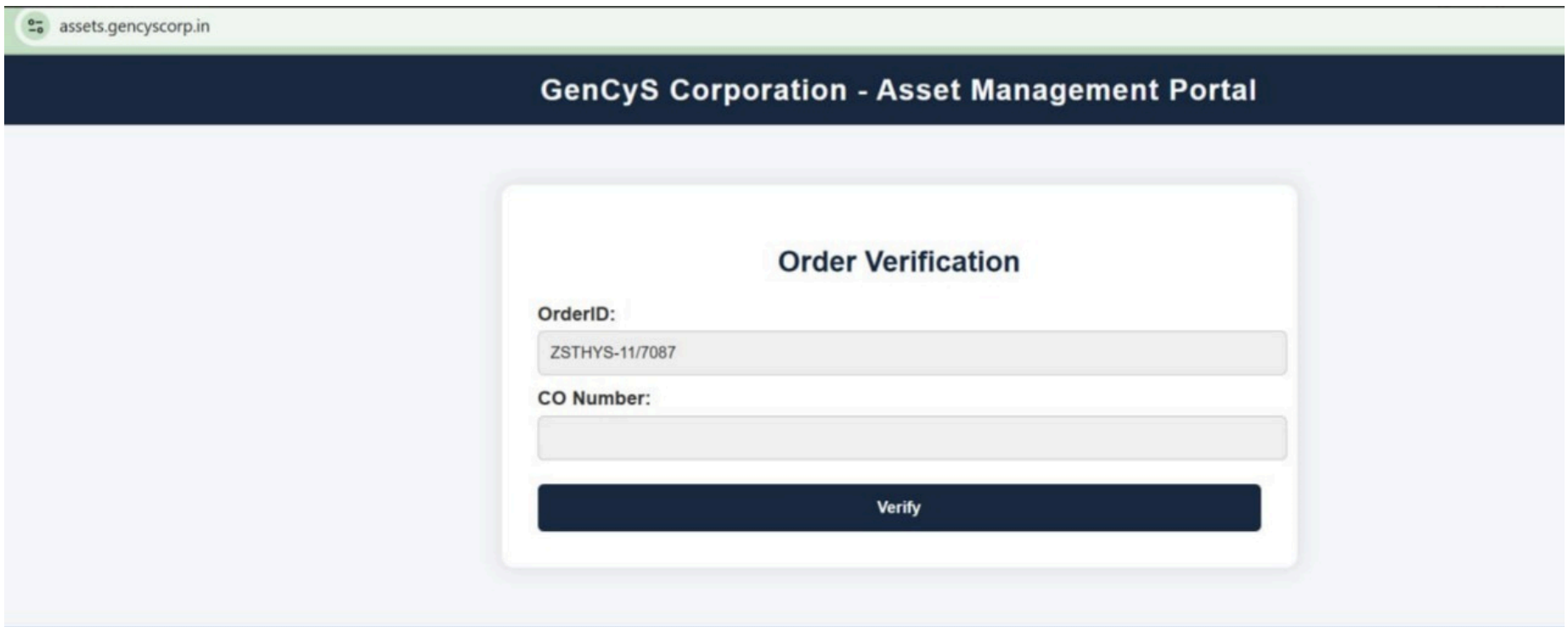
Challenge Description: GenCyS Corp is known for hiding things where few bother to look. Certain codes are attached to everyday structures inside their facilities, disguised as ordinary identifiers. Most employees pass by them daily without ever realizing their true purpose. These markings don't align with any public records, yet insiders say they can be "verified" if you know the right place to check. You've managed to note down one such code. Can you discover what it unlocks?

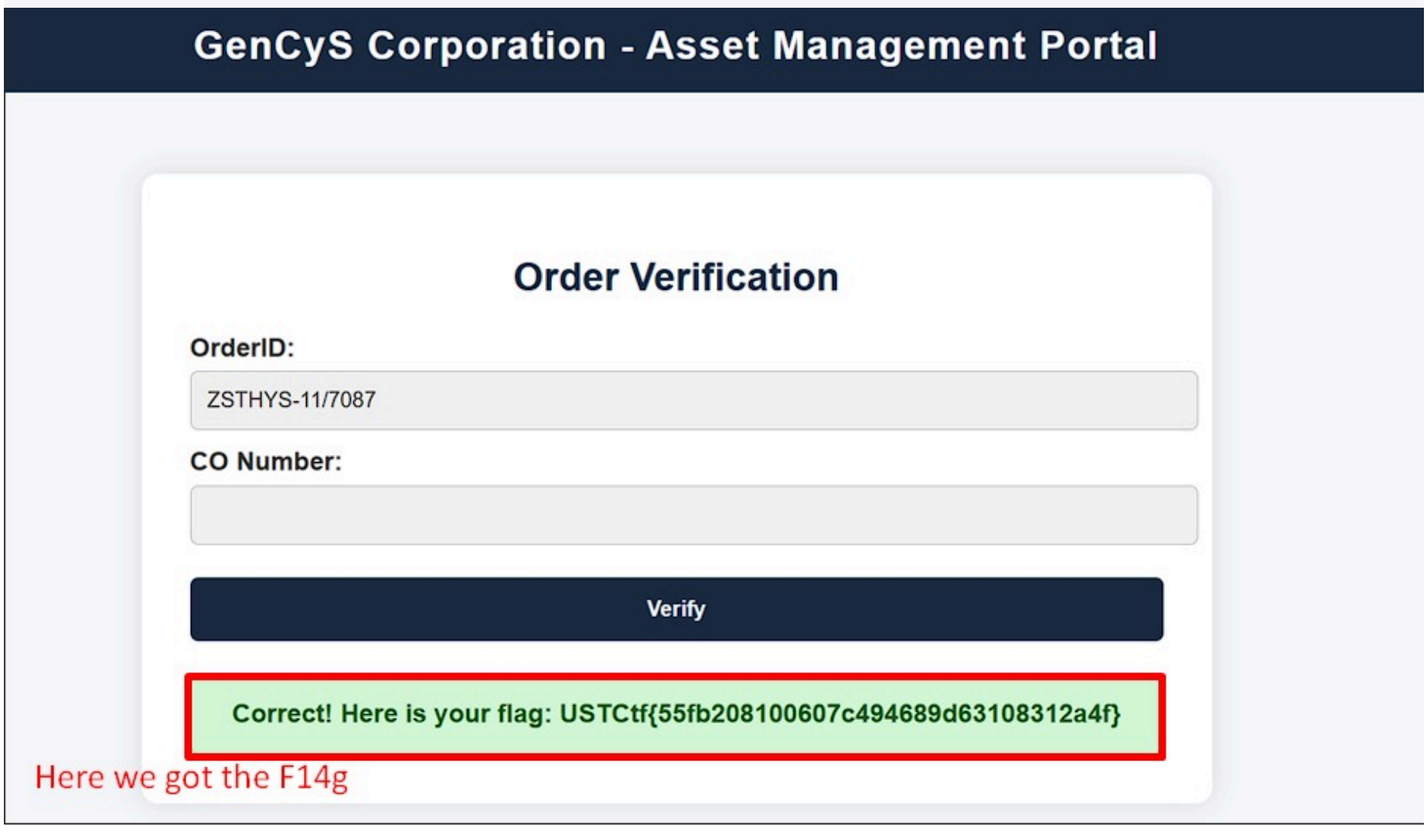
• Approach & Steps:

1. The challenge provided an Order ID (ZSTHYS-11/7087) via an assets page or link, hinting at a verification process requiring a corresponding CO Number.
2. Recognizing this as a physical OSINT task, I explored the event facility for hidden codes or stickers that might match the given Order ID.
3. While heading to the washroom, I noticed a sticker near the lift (elevator), affixed behind the glass— a location easily overlooked by passersby.



4. Inspected the sticker closely and found it contained a CO Number that directly corresponded to the provided Order ID.
5. Returned to the challenge's verification page (likely an online asset portal).





6. Entered the discovered CO Number into the input field and submitted the form via the "Verify" button.
7. The system confirmed the match, displaying a success message and revealing the flag.

Final Flag : USTCtf{55fb208100607c4946889d63108312a4f}

**Stegnography

Challenge Name : Silent Shades

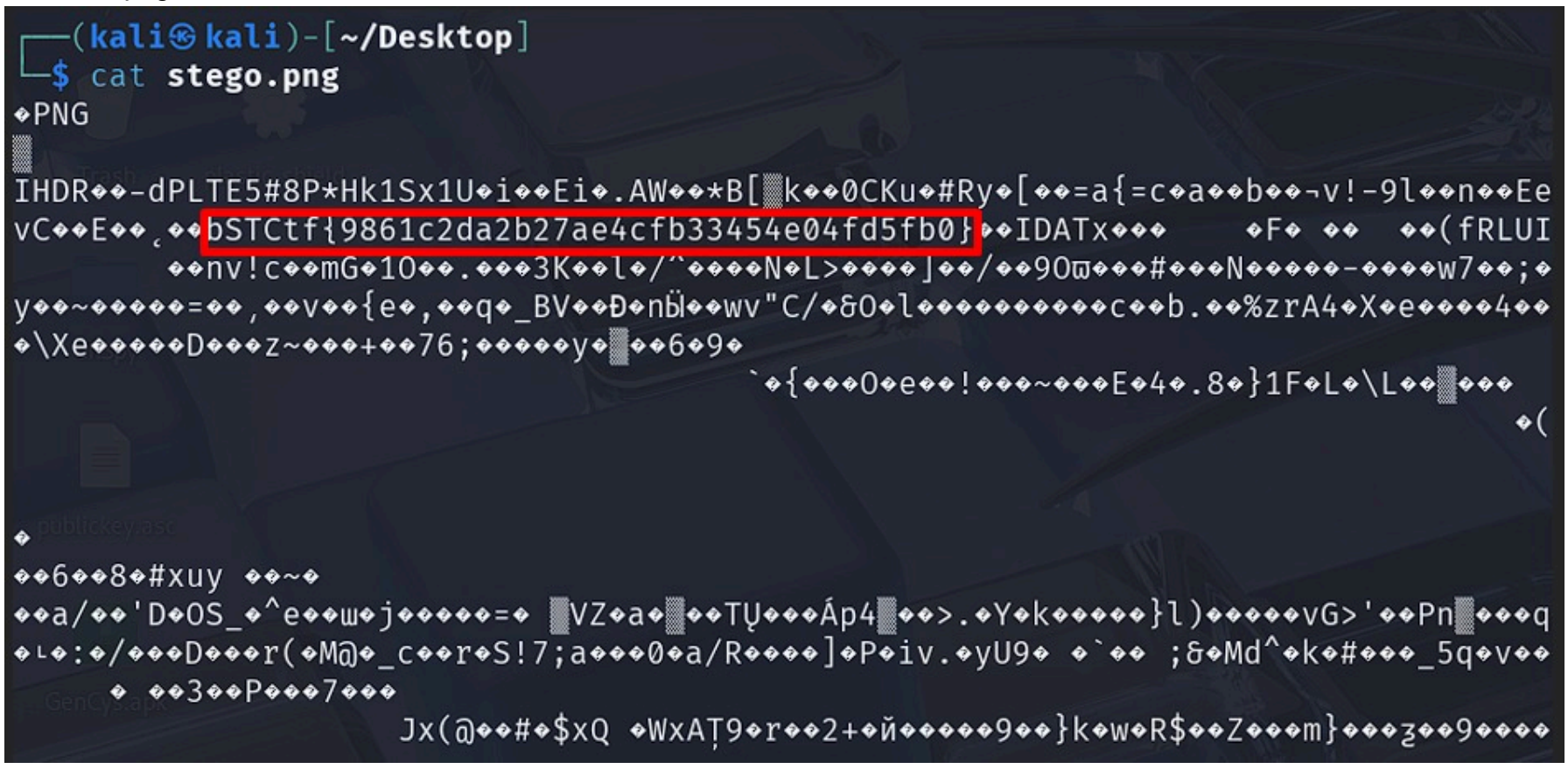
Points :

Description: An ordinary PNG image was found on an employee's workstation. It appeared harmless, but accompanying notes indicated the use of "non-visual storage techniques" to embed hidden data. The goal was to extract the concealed secret from the file.

• Approach & Steps:

1. Downloaded the image from: https://gencysctf.blob.core.windows.net/gencys/stego.png and saved it locally as ustctf.png.
2. Performed initial analysis using standard Kali Linux tools:
   binwalk to scan for embedded files.
   steghide to check for hidden data requiring a password.
   exiftool to examine metadata.
   strings to extract printable strings. No immediate leads were found.
3. Suspecting a simple embedding method, I viewed the raw file contents using the cat command:
   text

cat ustctf.png



4. Scrolling through the output revealed a readable string containing the flag, likely appended or hidden in non-image data sections. (Note from hints: "Stegongraphy just save it and Cat" confirms this low-tech approach.)

• Flag Obtained: bSTCtf{9861c2da2b27ae4cfb33454e04fd5fb0}
• But as for the format:-USTCtf{}
• So I inputed the in brackets string in it: USTCtf{9861c2da2b27ae4cfb33454e04fd5fb0}
Final Flag : USTCtf{9861c2da2b27ae4cfb33454e04fd5fb0}

Web

Challenge Name : Inference Override – 1

Points : 100

Challenge Description: The challenge involved an e-commerce-like site with unfinished authentication flows. Hints pointed to tier-based access (e.g., "Gold members get special treatment"), suggesting privilege escalation vulnerabilities.
• Approach & Steps:

1. Opened the challenge link, which redirected to: https://inference.gencyscorp.in/login.php.
2. Checked robots.txt for disallowed paths, revealing entries like
   • /admin-backup/
   • /dev-notes/
   • /checkout-old/
   • /beta-test/
   • /internal.php
   • /flag.txt # Nope, not here ;) (fake flag)
   • /gold-access-panel/
   • /support.php
3. Tested directories: At /internal.php, discovered a JSON file Internal_Credentials_Collection.json.
4. Now gave that json file to chatgpt it gave me an endpoint
5. API endpoint: https://inference.gencyscorp.in/api/creds.php, which returned credentials:

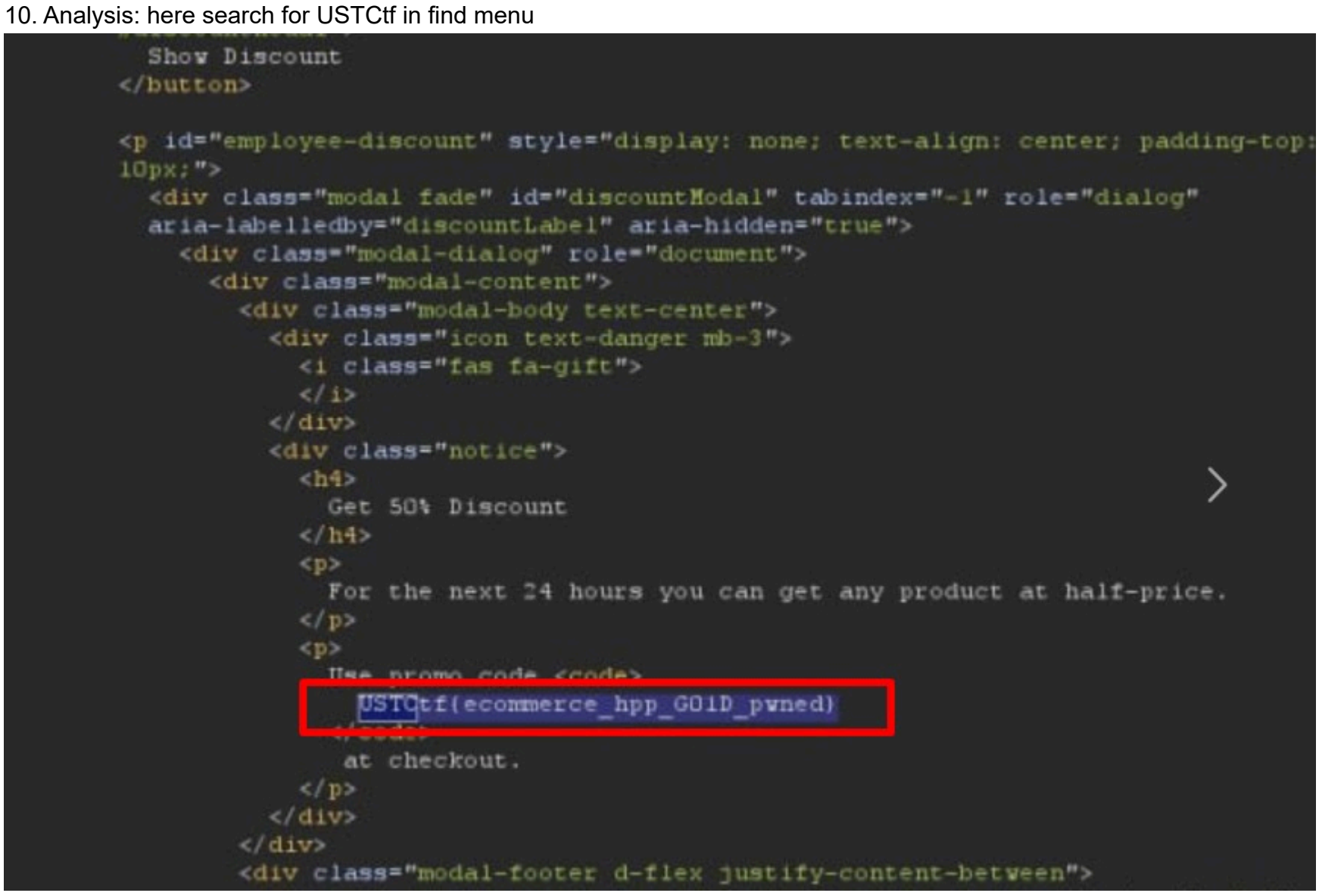
6. Logged in with these credentials, gaining access as a default-tier user. Inspected source code and cookies but found no escalation paths.
7. Used Burp Suite to intercept requests. Visiting /deals.php showed:
   text
   GET /deals.php HTTP/2
   Response: "Welcome johndoe (Tier: default)".
   As we got to know from robot.txt that only gold had access so I tried with tier=gold as Chatgpt gave that suggestion.
8. Attempted parameter tampering: Added ?tier=gold to the GET request, but it failed.

9. Switched to POST method via Burp Repeater:
   text
   POST /deals.php?tier=gold HTTP/2

```
POST /deals.php?tier=gold HTTP/2
Host: inference.gencyscorp.in
```

The server accepted this, treating the request as from a Gold-tier user and revealing the flag.

10. Analysis: here search for USTCtf in find menu



Final Flag : USTCtf{ecommerce_hpp_G01D_pwned}

Cryptography

Challenge name : Silent Sentinal
Points : 100
we got the secret.enc file in the challenge
then i ran this script :

lyric.py

```python
# Fix the ciphertext slice to exclude the newline before the '#'
data = open("secret.enc", "rb").read()
hash_pos = data.find(b"#")
ct = data[:hash_pos]
# Drop any trailing whitespace/newlines from ciphertext
ct = ct.rstrip(b"\r\n\t ")
print("Ciphertext length after trim:", len(ct))

from Crypto.Cipher import AES

def pkcs7_unpad(data, block_size=16):
    if not data or len(data) % block_size != 0:
        return None
    pad_len = data[-1]
    if pad_len < 1 or pad_len > block_size:
        return None
    return data[:-pad_len]

found = None
info = None

for i in range(10000):
    pin = f"{i:04d}"
dskey = (pin + "0"*12).encode()
    # ECB
    if len(ct) % 16 == 0:
        cipher = AES.new(key, AES.MODE_ECB)
        pt = cipher.decrypt(ct)
        unp = pkcs7_unpad(pt, 16) or pt
        s = unp.decode("utf-8", errors="ignore")
        if "{" in s and "}" in s and any(tag in s.lower() for tag in ["flag", "ctf"]):
            found = unp
            info = ("ECB", pin)
            break
    # CBC with zero IV
    cipher = AES.new(key, AES.MODE_CBC, iv=b"\x00"*16)
    if len(ct) % 16 == 0:
        pt = cipher.decrypt(ct)
        unp = pkcs7_unpad(pt, 16) or pt
        s = unp.decode("utf-8", errors="ignore")
        if "{" in s and "}" in s and any(tag in s.lower() for tag in ["flag", "ctf"]):
            found = unp
            info = ("CBC-iv0", pin)
            break

```

```
~/workspace$
~/workspace$ python lyric.py
Ciphertext length after trim: 48
=== FLAG FOUND ===
Mode: CBC-iv0 PIN: 7352

USTCtf{01eb61687e16324487eca30736cf4d6d}
Traceback (m                    ast):
  File "/hom                    e/lyric.py", line 54, in <module>
    with open(out_path, "w", encoding="utf-8") as f:
FileNotFoundError: [Errno 2] No such file or directory: '/mnt/data/decrypted_flag.txt'
~/workspace$
```

Ask Assistant

Final flag : USTCtf{01eb61687e16324487eca30736cf4d6d}

silent sentinal

Final Flag : USTCtf{De_flag}

Forensics

Challenge Name : Out of sight

Points : 100
Firstly there is a zip file given which contains 2 vidoes one is a 30 min another is 2 min.
So the 30 minute video contains 13 group of different frequencies at different time lines
. So we need to trim the complete vidoe and only leave the 13 groups of freq.
Now we have to convert it to the mp3 file . Now uploading the mp2 file in a DTMF online converter we get this
107#71#66#73#116#73#70#69#116#120#72#66#102#79#86#88#81#104#51#68#77#84#97#73#57#89#86#74#115#112#102#109#110#99#110#106#111#48#121#97#43#90#69#108#71#122#100#117#80#111#76#122#72#99#121#111#90#115#102#51#99#66#57#80

So this is an encrypted text on decrypting this we get a string which is
kGBItIFEtxHBfOVXQh3DMTaI9YVJspfmncnjo0ya+ZElGzduPoLzHcyoZsf3cB9P

Looking into the video carefully it asks us to decrypt using AES256
so we have a long string and we need a 32 bit length key to decrypt

So looking into the video file we need to binwalk both the files we get a yaffs1.bin
so we need to use this command to extract the data dd if=Video1.mp4 of=yaffs1.bin bs=1 skip=1994075

so doing all these further we get a key which looks like this 238729475239785987324958732893 21
so it is a 32 bit length one so using it as a key for the AES256 we got half flag USTCtf{445eb8b34

after running the script
import re
from Crypto.Cipher import AES
import base64

--Read the yaffs1.bin
with open("yaffs1.bin", "rb") as f:
data = f.read()

-- Extract ASCII strings that might be keys
candidates = re.findall(rb"[A-Za-z0-9+/=]{8,}", data)

print(f"Found {len(candidates)} candidate strings")

-- Try each candidate as AES key (16, 24, 32 bytes after padding/trimming)
for cand in candidates:
try:
key = cand.decode()
except:
continue

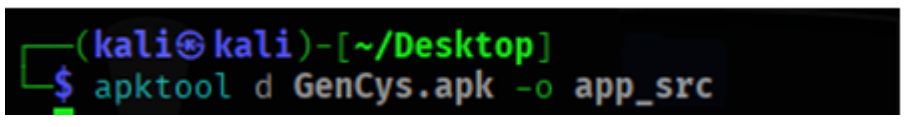
```
for size in [16, 24, 32]:
    k = key.encode()[:size].ljust(size, b"\0")  # normalize key size
    try:
        cipher = AES.new(k, AES.MODE_ECB)
        -- Try decrypting first 32 bytes of data
        decrypted = cipher.decrypt(data[:32])
        if decrypted.isascii():
            print(f"[+] Possible key: {key} (len {len(key)}) → {decrypted}")
    except Exception as e:
        pass
```

Final flag: USTCtf{445eb8b348dbaef887011882b56df69e}

Mobile

Challenge Name : The Untold
Points : 50

We are given an APK file (GenCys.apk) For mobile forensics analysis i used a tool called "apktool" after this i extracted the Main activity.java file over here you can clearly see the links from which the app is calling the data from i started visiting each website i got a morse code audio , i used a morse decoder online, and was able to extract the text SUO1VB0 on reversing the text turns out out to be 0BV1OUS since this is not the actuall flag i tested the images which was given i got this QR code from image 2.png This is what i received upon scanning the QR code it traversed to a website, instantly i viewed the source page of the site where i was able to find a base64 code, which wa eventually decoded.



**

MainActivity.java

```java
package com.example.geocysmusic;

import android.net.Uri;
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.ImageView;
import android.widget.TextView;
import androidx.appcompat.app.AppCompatActivity;
import androidx.media3.common.MediaItem;
import androidx.media3.container.NalUnitUtil;
import androidx.media3.datasource.okhttp.OkHttpDataSource;
import androidx.media3.exoplayer.ExoPlayer;
import androidx.media3.exoplayer.source.DefaultMediaSourceFactory;
import com.squareup.picasso.OkHttp3Downloader;
import com.squareup.picasso.Picasso;
import com.squareup.picasso.RequestCreator;
import java.util.List;
import kotlin.Metadata;
import kotlin.collections.CollectionsKt;
import kotlin.jvm.internal.Intrinsics;
import okhttp3.CertificatePinner;
import okhttp3.OkHttpClient;

/* compiled from: MainActivity.kt */
@Metadata(d1 = {"\u0000T\n\u0002\u0018\u0002\n\u0002\u0018\u0002\n\u0002\b\u0003\n\u0002\u0018\u0002\n\u0000\n\u0002\u0018\u0002\n\u0000\n\u0002\u0018\u0002\n\u0000\n\u0002\u0018\u0002\n\u0002\b\u0002
/* loaded from: classes4.dex */
public final class MainActivity extends AppCompatActivity {
    public static final int $stable = 8;
    private Button btnNext;
    private Button btnPlayPause;
    private Button btnPrev;
    private int currentIndex;
    private boolean isPlaying;
    private ExoPlayer player;
    private ImageView songImage;
    private TextView songTitle;
    private final List<String> songUrls = CollectionsKt.listOf((Object[]) new String[]{"https://geocysmusicplayer.web.app/Media/1.mp3", "https://geocysmusicplayer.web.app/Media/2.mp3", "https://geocys
```

```
HttpClient;

MainActivity.kt */
\u0002T\n\u0002\u0018\u0002\n\u0002\u0018\u0002\n\u0002\b\u0003\n\u0002\u0018\u0002\n\u0000\n\u0002\u0018\u0002\n\u0000\n\u0002\u0018\u0002\n\u0000\n\u0002\u0018\u0002\n\u0002\b\u0002\u0018 \n\u0002\u0010\u000e\n\u0002\n\u0000\n\u0002\u0010\u0008\n\u0000\n\u0002\u0010\u0002\n\u0000\n\u0002\u0010
lasses4.dex */
 MainActivity extends AppCompatActivity {
 final int $stable = 0;
 btnNext;
 btnPlayPause;
 btnPrev;
 currentIndex;
 isPlaying;
 player;
 songImage;
 songTitle;
 List<String> songUrls = CollectionsKt.listOf((Object[]) new String[]{"https://geocysmusicplayer.web.app/Media/1.mp3", "https://geocysmusicplayer.web.app/Media/2.mp3", "https://geocysmusicplayer.web.app/Media/3.mp3", "https://geocysmusicplayer.web.app/Media/4.mp3"});
 List<String> songImages = CollectionsKt.listOf((Object[]) new String[]{"https://geocysmusicplayer.web.app/Media/1.png", "https://geocysmusicplayer.web.app/Media/2.png", "https://geocysmusicplayer.web.app/Media/3.png", "https://geocysmusicplayer.web.app/Media/4.png"});
 List<String> songNames = CollectionsKt.listOf((Object[]) new String[]{"ギャレージ", "王のセト", "ゼロ下6", "ヒント"});

 Access modifiers changed from: protected */
```

gencysmusicplayer.web.app/Media/3.mp3

morsecode.world/international/decoder/audio-decoder-adaptive.html

MorseCode.World | Shop | International | American | Labs | More | SCPhillips.com

Morse Code Adaptive Audio Decoder

International Morse Decoders | Audio Decoder | Audio Decoder (Expert) | Gaze Decoder

Morse Decoder

This is an experimental tool for listening to, analysing and decoding International Morse code. No information from the microphone is transmitted to the server, but the connection to the server is encrypted nonetheless.

If you cannot produce your own Morse code sounds then try using my Morse code translator to play or download some.

Alphabet to decode into

Latin

All these alphabets can be sent in Morse using standard timing. The "Latin" alphabet is e.g. "ABC".

Use the microphone:

Listen | Stop

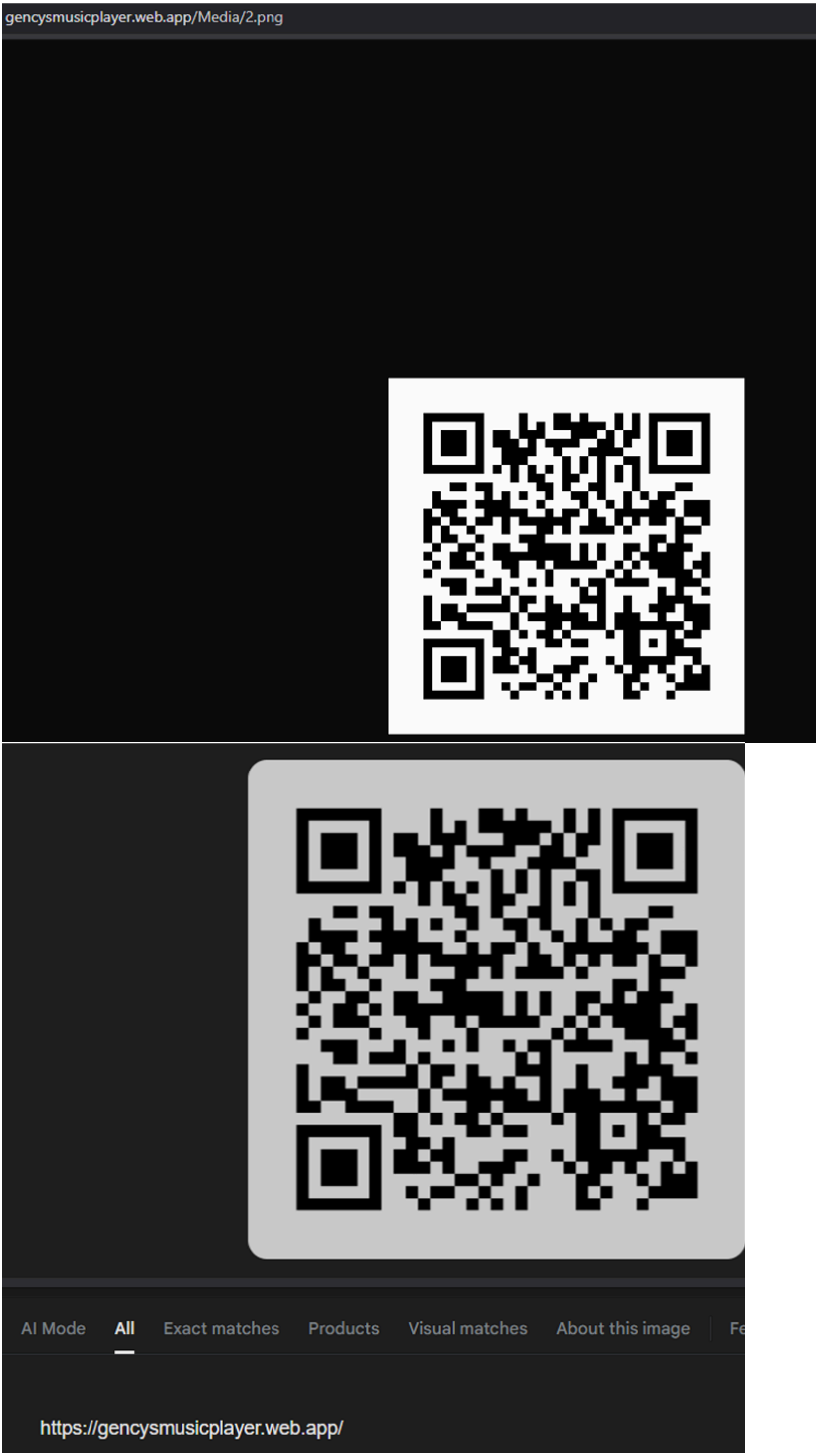
Or analyse an audio file containing Morse code:

Upload | Play | Stop

Filename: "3.mp3"

SUO1VB0

gencysmusicplayer.web.app/Media/2.png



AI Mode All Exact matches Products Visual matches About this image Fe

https://gencysmusicplayer.web.app/

```
        padding: 40px 60px;
        border-radius: 16px;
        box-shadow: 0 10px 30px rgba(0, 0, 0, 0.1);
        text-align: center;
        max-width: 400px;
    }
    h2 {
        margin-bottom: 16px;
        font-size: 24px;
        color: #222;
    }
    p {
        font-size: 18px;
        color: #444;
    }
    code {
        background-color: #f4f4f4;
        padding: 2px 6px;
        border-radius: 4px;
        font-family: monospace;
        color: #333;
    }
  </style>
head>
ody>
  <div class="container">
      <h2>Welcome to the Challenge</h2>
       <!-- VVNUQ3Rme24wdF9zMF8qKioqKioqXzNuYzBkMW5nfQ== -->
      <p>Try searching <code>/flag.txt</code></p>
  </div>


body>
```

```
VVNUQ3Rme24wdF9zMF8qKioqKioqXzNuYzBkMW5nfQ==
```

For encoded binaries (like images, documents, etc.) use the file upload form a

UTF-8 Source character set.

Decode each line separately (useful for when you have multiple entries).

Live mode OFF Decodes in real-time as you type or paste (supports onl

DECODE Decodes your data into the area below.

```
USTCtf{n0t_s0_*******_3nc0d1ng}
```

From the mobile SF we submitted the flag

**Final Flag : USTCtf{n0t_s0_0BV1OUS_3nc0d1ng}