```
# INSTALL REQUIRED PACKAGES
# requests and pandas are Python libraries, to install via pip, Python's package manager
# requests is for making HTTP requests (like fetching web pages)
# pandas is for working with structured data (tables, CSVs, etc)
# youtube-comment-downloader is a package for downloading Youtube comments without using the Youtube API
# quiet makes the output cleaner
# each of the import lines imports a Python module or library

!pip install --quiet youtube-comment-downloader pandas requests

import pandas as pd
import re
import requests
import json
from youtube_comment_downloader import YoutubeCommentDownloader

# SET PREFERENCE FOR ENGLISH LANGUAGE COMMENTS
class EnglishCommentDownloader(YoutubeCommentDownloader):
    def __init__(self):
        super().__init__()
        self.session.headers.update({
            'Accept-Language': 'en-US,en;q=0.9'
        })

# EXTRACT VIDEO ID FROM URL
def extract_video_id(url):
    match = re.search(r"v=([^&]+)", url)
    return match.group(1) if match else None

# GET PUBLISH DATE FROM VIDEO PAGE HTML
def get_publish_date(video_url):
    headers = {"User-Agent": "Mozilla/5.0 (Windows NT 10.0; Win64; x64)"}
    response = requests.get(video_url, headers=headers)
```

```
    html = response.text

    match = re.search(r'ytInitialPlayerResponse\s*=\s*({.+?});', html)
    if not match:
        return None

    data_json = match.group(1)
    data = json.loads(data_json)
    publish_date = data.get("microformat", {}).get("playerMicroformatRenderer", {}).get("publishDate", None)
    return publish_date

# SUPPLY VIDEO LINKS AND GIVE THEM NAMES
# Add any additional (name, url) pairs desired
videos = [
("Swedish Trans Experience", "https://www.youtube.com/watch?v=k6NYnGpBJbs"),
("MCC Brussels", "https://www.youtube.com/watch?v=t_6lYfTOJkQ"),
("DW Report Pope", "https://www.youtube.com/watch?v=KdCJXf3ctnE"),
("DW Report UK", "https://www.youtube.com/watch?v=ta0H-ESCinw"),

]

downloader = EnglishCommentDownloader()

# COLLECT ALL COMMENTS FROM THESE VIDEOS
all_comments = []

for video_name, video_url in videos:
    print(f"Processing: {video_name} ({video_url})")
    video_id = extract_video_id(video_url)
    if not video_id:
        print("Invalid URL, skipping.")
        continue
```

```
    # GET PUBLISH DATE
    publish_date = get_publish_date(video_url) or "Unknown"
    print(f"Publish date: {publish_date}")

    # SCRAPE COMMENTS
    try:
        comments = downloader.get_comments_from_url(video_url)
        for comment in comments:
            all_comments.append({
                'video_name': video_name,
                'video_url': video_url,
                'video_publish_date': publish_date,
                'author': comment.get('author'),
                'comment': comment.get('text'),
                'time': comment.get('time'),
                'likes': comment.get('votes'),
            })
    except Exception as e:
        print(f"Error fetching comments for this video: {e}")
        continue

# SAVE ALL COMMENTS TO A SINGLE CSV FILE
combined_df = pd.DataFrame(all_comments)
csv_filename = 'all_youtube_comments.csv'
combined_df.to_csv(csv_filename, index=False)
print(f"Combined CSV saved: {csv_filename} with {len(combined_df)} comments.")

# DOWNLOAD CSV FROM GOOGLE COLAB
from google.colab import files
files.download(csv_filename)
```