```
Start coding or generate with AI.
```

```
━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 363.4/363.4 MB 4.4 MB/s eta 0:00:00
━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 13.8/13.8 MB 63.9 MB/s eta 0:00:00
━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 24.6/24.6 MB 35.1 MB/s eta 0:00:00
━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 883.7/883.7 kB 42.0 MB/s eta 0:00:00
━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 664.8/664.8 MB 850.0 kB/s eta 0:00:00
━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 211.5/211.5 MB 6.3 MB/s eta 0:00:00
━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 56.3/56.3 MB 9.8 MB/s eta 0:00:00
━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 127.9/127.9 MB 9.1 MB/s eta 0:00:00
━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 207.5/207.5 MB 5.1 MB/s eta 0:00:00
━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 21.1/21.1 MB 47.4 MB/s eta 0:00:00
```

```python
from transformers import pipeline

# Load sentiment classifier model (free + small)
classifier = pipeline("text-classification", model="distilbert-base-uncased-finetuned-sst-2-english")
```

```
/usr/local/lib/python3.11/dist-packages/huggingface_hub/utils/_auth.py:94: UserWarning:
The secret `HF_TOKEN` does not exist in your Colab secrets.
To authenticate with the Hugging Face Hub, create a token in your settings tab (https://huggingface.co/settings/tokens), set it as s
You will be able to reuse this secret in all of your notebooks.
Please note that authentication is recommended but still optional to access public models or datasets.
  warnings.warn(
config.json: 100%                                   629/629 [00:00<00:00, 13.6kB/s]
model.safetensors: 100%                             268M/268M [00:03<00:00, 89.6MB/s]
tokenizer_config.json: 100%                         48.0/48.0 [00:00<00:00, 2.28kB/s]
vocab.txt: 100%                                     232k/232k [00:00<00:00, 4.40MB/s]
Device set to use cpu
```

```python
def classify_content(title, desc):
    # Combine title and description
    text = title + " " + desc

    # Run the text through the classifier
    result = classifier(text)[0]  # returns label + score

    # Map sentiment label to Mindful/Mindless
    label = result['label']  # 'POSITIVE' or 'NEGATIVE'
    return "Mindful" if label == "POSITIVE" else "Mindless"
```

```python
# 🔍 Sample video to classify
test_title = "10 YouTube Shorts That Will Blow Your Mind"
test_desc = "A compilation of fun, viral, trending short videos."

# Run classification
output = classify_content(test_title, test_desc)

# Show result
print(f"🎬 Title: {test_title}")
print(f"📝 Description: {test_desc}")
print(f"✅ Classification: {output}")
```

```
🎬 Title: 10 YouTube Shorts That Will Blow Your Mind
📝 Description: A compilation of fun, viral, trending short videos.
✅ Classification: Mindful
```

```python
from googleapiclient.discovery import build

# Initialize YouTube API client
def get_youtube_service(api_key):
    return build('youtube', 'v3', developerKey=api_key)

# Fetch video titles and descriptions by search query
def get_videos(query, api_key, max_results=5):
    youtube = get_youtube_service(api_key)
    request = youtube.search().list(
        q=query,
        part='snippet',
        type='video',
        maxResults=max_results
    )
    response = request.execute()
    videos = []
```

```
    for item in response['items']:
        title = item['snippet']['title']
        desc = item['snippet']['description']
        videos.append((title, desc))
    return videos
```

```
youtube_api_key = "AIzaSyD3hBJrfsttsrs6zWHt5rasWpnfitJk27E"
```

```
# 🔍 Search YouTube + classify each result
query = "deep work productivity"

videos = get_videos(query, youtube_api_key, max_results=5)

for i, (title, desc) in enumerate(videos, 1):
    label = classify_content(title, desc)
    print(f"🎬 {i}. {title}")
    print(f"📝 {desc}")
    print(f"🧠 Classified as: {label}")
    print("-" * 60)
```

```
🎬 1. Success in a distracted world: DEEP WORK by Cal Newport
📝 1-Page PDF Summary: https://lozeron-academy-llc.kit.com/deepwork Book Link: http://amzn.to/29sgNW7 Join the Productivity ...
🧠 Classified as: Mindful
------------------------------------------------------------
🎬 2. The Deep Work Routine That Changed My Life
📝 I've tried all the productivity hacks. Here's how you take back control of your attention. Become future-proof (2 weekly letters
🧠 Classified as: Mindless
------------------------------------------------------------
🎬 3. Music for Work — Limitless Productivity Radio
📝 This radio is here to make your day more productive. It plays a carefully selected mix of deep future garage and soothing chills
🧠 Classified as: Mindful
------------------------------------------------------------
🎬 4. Avoiding Distractions &amp;amp; Doing Deep Work | Dr. Cal Newport &amp;amp; Dr. Andrew Huberman
📝 Dr. Cal Newport and Dr. Andrew Huberman discuss the role of technology, social media, and internet usage in our lives, ...
🧠 Classified as: Mindful
------------------------------------------------------------
🎬 5. Deep Work Music — Maximum Productivity and Concentration Mix
📝 Welcome to our carefully crafted electronic music mix, designed to elevate focus and productivity. Featuring deep and dark Futur
🧠 Classified as: Mindful
------------------------------------------------------------
```

```
import pandas as pd
from datetime import datetime

def log_results(videos_with_labels, filename="mindfulfeed_log.csv"):
    rows = []
    for title, desc, label in videos_with_labels:
        rows.append({
            "Timestamp": datetime.now().strftime("%Y-%m-%d %H:%M:%S"),
            "Title": title,
            "Description": desc,
            "Label": label
        })
    df = pd.DataFrame(rows)
    df.to_csv(filename, mode='a', index=False, header=not pd.io.common.file_exists(filename))
    print(f"✅ Logged {len(rows)} entries to {filename}")
```

```
# Classify and log results
query = "focus tips"
videos = get_videos(query, youtube_api_key, max_results=5)

results = []
for title, desc in videos:
    label = classify_content(title, desc)
    print(f"🎬 {title} → 🧠 {label}")
    results.append((title, desc, label))

# Save to CSV
log_results(results)
```

```
🎬 5 Tips to Quickly Improve Focus &amp;amp; Concentration → 🧠 Mindful
🎬 6 ADHD techniques to help you FOCUS → 🧠 Mindful
🎬 Neuroscientist: How To Boost Your Focus PERMANENTLY in Minutes → 🧠 Mindful
🎬 How to Focus While Studying → 🧠 Mindless
🎬 Can&amp;#39;t focus? The SECRET to study with LASER FOCUS → 🧠 Mindful
```

```
✅ Logged 5 entries to mindfulfeed_log.csv
```

```
# Classify and log results
query = "focus tips"
videos = get_videos(query, youtube_api_key, max_results=5)

results = []
for title, desc in videos:
    label = classify_content(title, desc)
    print(f"🎬 {title} → 🧠 {label}")
    results.append((title, desc, label))

# Save to CSV
log_results(results)
```

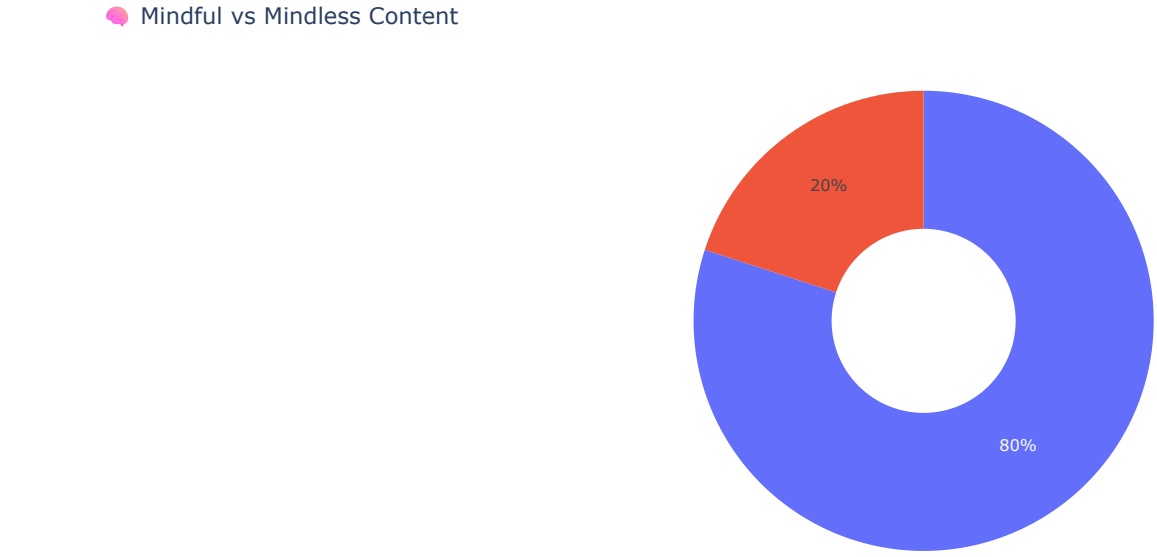
```
🎬 5 Tips to Quickly Improve Focus &amp;amp; Concentration → 🧠 Mindful
🎬 6 ADHD techniques to help you FOCUS → 🧠 Mindful
🎬 Neuroscientist: How To Boost Your Focus PERMANENTLY in Minutes → 🧠 Mindful
🎬 How to Focus While Studying → 🧠 Mindless
🎬 Can&amp;#39;t focus? The SECRET to study with LASER FOCUS → 🧠 Mindful
✅ Logged 5 entries to mindfulfeed_log.csv
```

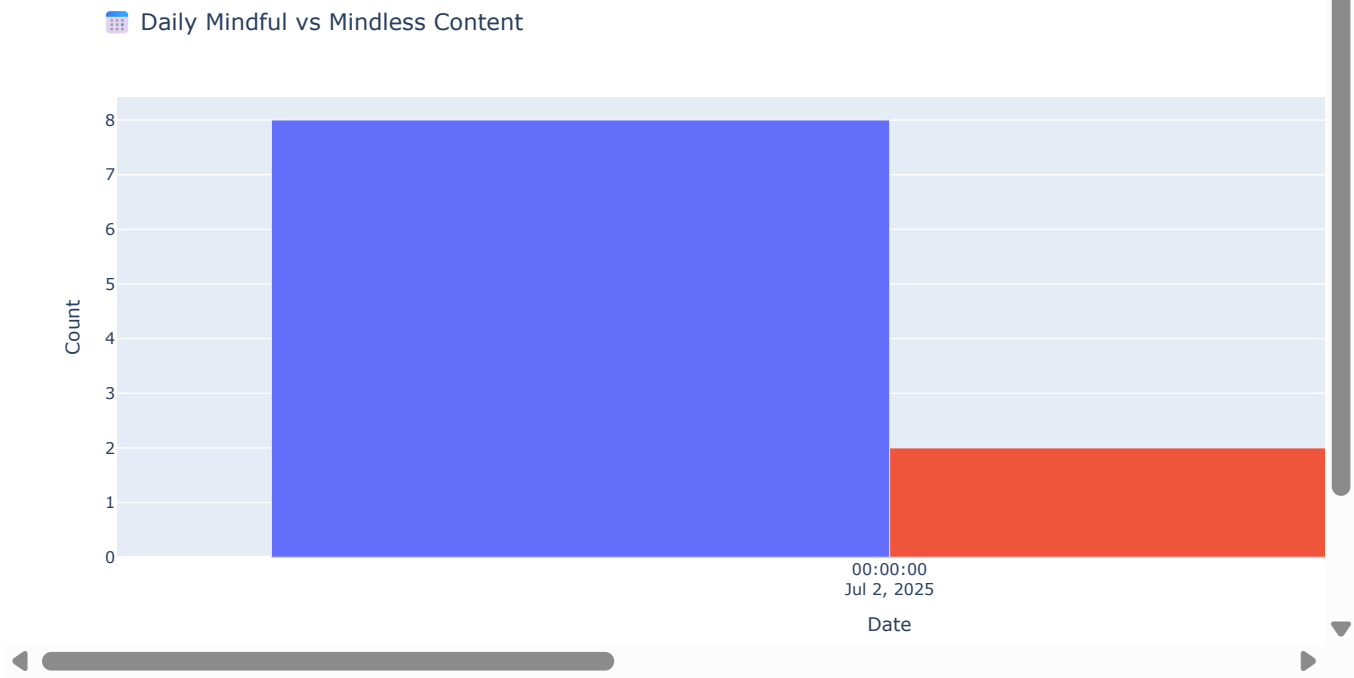
```
def check_daily_limit(filename="mindfulfeed_log.csv", limit=3):
    df = pd.read_csv(filename)
    df['Timestamp'] = pd.to_datetime(df['Timestamp'])
    today = pd.Timestamp.now().normalize()
    today_logs = df[df['Timestamp'].dt.date == today.date()]
    mindless_count = (today_logs['Label'] == "Mindless").sum()

    if mindless_count >= limit:
        print(f"🚨 You've watched {mindless_count} mindless videos today. Consider taking a break!")
    else:
        print(f"✅ You're within your mindful limit: {mindless_count}/{limit}")
```

```
Start coding or generate with AI.
```

```
import plotly.express as px

def visualize_logs(filename="mindfulfeed_log.csv"):
    try:
        df = pd.read_csv(filename)
    except FileNotFoundError:
        print("⚠️ No logs found yet.")
        return

    if df.empty:
        print("🟡 Log is empty.")
        return

    # --- Pie Chart: Mindful vs Mindless ---
    pie_fig = px.pie(
        df,
        names='Label',
        title='🧠 Mindful vs Mindless Content',
        hole=0.4,
        color_discrete_map={"Mindful": "green", "Mindless": "red"}
    )
    pie_fig.show()

    # --- Bar Chart: Content Over Time ---
    df['Timestamp'] = pd.to_datetime(df['Timestamp'])
    df['Date'] = df['Timestamp'].dt.date

    bar_data = df.groupby(['Date', 'Label']).size().unstack(fill_value=0)
    bar_fig = px.bar(
        bar_data,
        barmode='group',
        title='📅 Daily Mindful vs Mindless Content',
        labels={'value': 'Count', 'Date': 'Date'}
    )
    bar_fig.show()
```

```
visualize_logs()
```






```
def mindful_goal_tracker(filename="mindfulfeed_log.csv", goal=5):
    df = pd.read_csv(filename)
    df['Timestamp'] = pd.to_datetime(df['Timestamp'])
    today = pd.Timestamp.now().normalize()
    today_logs = df[df['Timestamp'].dt.date == today.date()]
    mindful_count = (today_logs['Label'] == "Mindful").sum()

    if mindful_count >= goal:
        print(f"🌟 Great job! You hit your mindful goal: {mindful_count}/{goal}")
    else:
        print(f"💡 You've watched {mindful_count}/{goal} mindful videos today. Keep going!")
```