**Meme\_maker**

import os, random﻿from io import BytesIO﻿from datetime import datetime﻿import requests﻿from PIL import Image, ImageDraw, ImageFont﻿from openai import OpenAI﻿# ---- API KEY ----﻿API\_KEY = "sk-proj-FhbxWNZEkxsgwTG6wpYY8J9TXBZO297clP5xIZyR2tDHYjRBFTuusa3svld3rWwroZOJLqipzIT3BlbkFJJVl6KAx3Vyv52LxacXMzwY6J9\_ul7eIkp8L2VmKEfZIumRsBK6\_knwiIyCssNhi0Sgt2fN2nMA"  # Your OpenAI API key﻿if not API\_KEY.startswith("sk-"):﻿    raise SystemExit("Please set your OpenAI API key in API\_KEY variable.")﻿client = OpenAI(api\_key=API\_KEY)﻿# ---- Background Images ----﻿backgrounds = [﻿    "https://images.unsplash.com/photo-1501594907352-04cda38ebc29",﻿    "https://images.unsplash.com/photo-1502672260266-1c1ef2d93688",﻿    "https://images.unsplash.com/photo-1500530855697-b586d89ba3ee",﻿    "https://images.unsplash.com/photo-1519682337058-a94d519337bc",﻿    "https://images.unsplash.com/photo-1520974749533-6d1fda1b2d74",﻿    "https://images.unsplash.com/photo-1560807707-8cc77767d783",﻿    "https://images.unsplash.com/photo-1602524812292-92d4f0e3f70c"﻿]﻿# ---- Font Loader ----﻿def get\_font(size):﻿    for path in ["/system/fonts/Roboto-Bold.ttf", "/system/fonts/Roboto-Regular.ttf"]:﻿        if os.path.exists(path):﻿            return ImageFont.truetype(path, size)﻿    cache\_dir = "/storage/emulated/0/Pydroid3/deps"﻿    os.makedirs(cache\_dir, exist\_ok=True)﻿    font\_path = os.path.join(cache\_dir, "DejaVuSans-Bold.ttf")﻿    if not os.path.exists(font\_path):﻿        r = requests.get(﻿            "https://github.com/dejavu-fonts/dejavu-fonts/blob/master/ttf/DejaVuSans-Bold.ttf?raw=1",﻿            timeout=30﻿        )﻿        with open(font\_path, "wb") as f:﻿            f.write(r.content)﻿    return ImageFont.truetype(font\_path, size)﻿# ---- AI Caption Generator ----﻿def generate\_captions(topic):﻿    prompt = (﻿        f"Create a funny meme about '{topic}' in exactly two lines.\n"﻿        "Do NOT use 'Top:', 'Bottom:', numbers, or any labels.\n"﻿        "Just write two short, punchy lines (under 8 words each).\n"﻿        "Line 1 is the top caption, line 2 is the bottom caption."﻿    )﻿    resp = client.chat.completions.create(﻿        model="gpt-4o-mini",﻿        messages=[{"role": "user", "content": prompt}],﻿        temperature=0.9,﻿    )﻿    lines = [line.strip() for line in resp.choices[0].message.content.split("\n") if line.strip()]﻿    clean\_lines = []﻿    for line in lines:﻿        line = line.replace("Top:", "").replace("Bottom:", "")﻿        line = line.lstrip("0123456789.-: ").strip()﻿        clean\_lines.append(line)﻿    top = clean\_lines[0] if len(clean\_lines) > 0 else ""﻿    bottom = clean\_lines[1] if len(clean\_lines) > 1 else ""﻿    return top, bottom﻿# ---- Text Drawing ----﻿def draw\_text\_centered(draw, text, y, font, image\_width):﻿    words = text.split()﻿    lines = []﻿    line = ""﻿    for word in words:﻿        test\_line = line + (" " if line else "") + word﻿        if draw.textbbox((0, 0), test\_line, font=font)[2] < image\_width \* 0.9:﻿            line = test\_line﻿        else:﻿            lines.append(line)﻿            line = word﻿    if line:﻿        lines.append(line)﻿    for line in lines:﻿        w = draw.textbbox((0, 0), line, font=font)[2]﻿        x = (image\_width - w) // 2﻿        draw.text(﻿            (x, y),﻿            line,﻿            font=font,﻿            fill="white",﻿            stroke\_width=3,﻿            stroke\_fill="black",﻿        )﻿        y += font.size + 5﻿    return y﻿# ---- Main ----﻿topic = input("Enter meme topic: ").strip()﻿top\_text, bottom\_text = generate\_captions(topic)﻿# Pick two random backgrounds﻿url1 = random.choice(backgrounds)﻿url2 = random.choice(backgrounds)﻿img1\_data = requests.get(url1, timeout=30).content﻿img2\_data = requests.get(url2, timeout=30).content﻿img1 = Image.open(BytesIO(img1\_data)).convert("RGB").resize((800, 800))﻿img2 = Image.open(BytesIO(img2\_data)).convert("RGB").resize((800, 800))﻿# Create new blank image with double height﻿final\_img = Image.new("RGB", (800, 1600))﻿final\_img.paste(img1, (0, 0))﻿final\_img.paste(img2, (0, 800))﻿# Draw text﻿draw = ImageDraw.Draw(final\_img)﻿font\_size = 60﻿font = get\_font(font\_size)﻿# Top caption on first image﻿draw\_text\_centered(draw, top\_text, 20, font, 800)﻿# Bottom caption at top of second image﻿draw\_text\_centered(draw, bottom\_text, 800 + 20, font, 800)﻿# Save file﻿timestamp = datetime.now().strftime("%Y%m%d\_%H%M%S")﻿out\_path = f"/storage/emulated/0/Download/meme\_{timestamp}.jpg"﻿final\_img.save(out\_path, format="JPEG", quality=95)﻿print("✅ Meme saved as:", out\_path)