Folder Structure

viral-clip-generator/ │ ├── app.py ├── utils.py ├── requirements.txt ├── README.md ├── .gitignore ├── templates/ │ ├── index.html │ └── clips.html └── static/ ├── uploads/ (empty folder) └── clips/ (empty folder)

app.py

```
from flask import Flask, render_template, request, send_from_directory
import os, tempfile
from utils import download_youtube, generate_subtitles, save_clip
from moviepy.editor import VideoFileClip

app = Flask(__name__)
UPLOAD_FOLDER = "static/uploads"
CLIP_FOLDER = "static/clips"
os.makedirs(UPLOAD_FOLDER, exist_ok=True)
os.makedirs(CLIP_FOLDER, exist_ok=True)

@app.route("/", methods=["GET","POST"])
def index():
    if request.method=="POST":
        file = request.files.get("video")
        url = request.form.get("url")
        top = int(request.form.get("top",3))
        vertical = request.form.get("vertical")=="on"
        subs = request.form.get("subs")=="on"

        if url:
            video_path = download_youtube(url)
        elif file:
            video_path = os.path.join(UPLOAD_FOLDER,file.filename)
            file.save(video_path)
        else:
            return render_template("index.html", error="Please provide a video
or URL")

        audio_path = tempfile.mktemp(suffix=".wav")
        clip = VideoFileClip(video_path)
        clip.audio.write_audiofile(audio_path, logger=None)
        clip.close()
        subtitles = generate_subtitles(audio_path) if subs else []

        duration = VideoFileClip(video_path).duration
        step = max(int(duration/top),60)
        saved_files=[]
        for i in range(top):
            start=i*step
            end=min(start+60,duration)
            outp=os.path.join(CLIP_FOLDER,f"clip_{i+1}.mp4")
            save_clip(video_path,start,end,outp,subtitles,vertical)
            saved_files.append(f"clips/clip_{i+1}.mp4")

        return render_template("clips.html", files=saved_files)
    return render_template("index.html")

@app.route("/clips/<path:filename>")
def download_clip(filename):
    return send_from_directory(CLIP_FOLDER, filename)

if __name__=="__main__":
    app.run(debug=True)
```

---

utils.py

```
import os, subprocess
from moviepy.editor import VideoFileClip, TextClip, CompositeVideoClip
import whisper

def download_youtube(url, out_path="static/uploads/video.mp4"):
    cmd = ["yt-dlp", "-f", "bestvideo+bestaudio/best", "-o", out_path, url]
    subprocess.check_call(cmd)
    return out_path

def generate_subtitles(audio_path):
    model = whisper.load_model("small")
    result = model.transcribe(audio_path)
    return result["segments"]

def save_clip(video_path, start, end, out_path, subtitles=[], vertical=False):
    clip = VideoFileClip(video_path).subclip(start,end)
    if vertical:
        w,h = clip.size
        ratio=9/16
        new_width=int(h*ratio)
        x1=(w-new_width)//2
        clip = clip.crop(x1=x1,x2=x1+new_width).resize(height=1080)
    txt_clips=[]
    for seg in subtitles:
        if seg["start"]>=end or seg["end"]<=start: continue
        t_start=max(seg["start"],start)-start
        t_end=min(seg["end"],end)-start
        caption = TextClip(seg["text"],fontsize=40,color="white",
                           stroke_color="black",stroke_width=2,size=clip.size,

method="caption").set_start(t_start).set_end(t_end).set_position(("center","bottom"))
        txt_clips.append(caption)
    final = CompositeVideoClip([clip]+txt_clips)
    final.write_videofile(out_path, codec="libx264", audio_codec="aac",
logger=None)
    clip.close()
    final.close()
```

templates/index.html

```
<!doctype html>
<html>
<head><title>Viral Clip Generator</title></head>
<body>
<h1>Viral Clip Generator</h1>
<form method="POST" enctype="multipart/form-data">
    <label>YouTube URL (optional):</label><br>
    <input type="text" name="url" placeholder="Paste URL"><br><br>
    <label>Upload Video:</label><br>
    <input type="file" name="video"><br><br>
    <label>Number of Clips:</label>
    <input type="number" name="top" value="3"><br><br>
    <input type="checkbox" name="subs"> Add Subtitles<br>
    <input type="checkbox" name="vertical"> Vertical 9:16<br><br>
    <button type="submit">Generate Clips</button>
</form>
</body>
</html>
```

templates/clips.html

```html
<!doctype html>
<html>
<head><title>Generated Clips</title></head>
<body>
<h1>Generated Clips</h1>
<ul>
{% for file in files %}
    <li><a href="{{ file }}">{{ file }}</a></li>
{% endfor %}
</ul>
<a href="/">Go Back</a>
</body>
</html>
```

---

requirements.txt

```
flask
moviepy
whisper
numpy
opencv-python
pytesseract
yt-dlp
```

.gitignore

```
__pycache__/
static/uploads/
static/clips/
*.pyc
```

README.md

```
# Viral Clip Generator

A GitHub-ready Flask web app to generate viral video clips from YouTube or local
videos.

## Features
- YouTube URL input
- Local video upload
- Multi-clip extraction
- Subtitles auto-generate (Whisper)
- Vertical crop 9:16 for TikTok/Shorts
- Download-ready clips

## Setup

1. Clone repo:
   git clone <repo-url>
   cd viral-clip-generator

2. Install dependencies:
   pip install -r requirements.txt

3. Run the app:
   python app.py

4. Open browser:
   http://127.0.0.1:5000/
```