Open in Colab

```
[1] from google.colab import drive
    drive.mount('/content/drive')
```

```
Mounted at /content/drive
```

```
[2] !pip install --upgrade diffusers transformers -q
```

```
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 3.6/3.6 MB 50.3 MB/s eta 0:00:00
```

```
[3] from pathlib import Path
    import tqdm
    import torch
    import pandas as pd
    import numpy as np
    from diffusers import StableDiffusionPipeline
    from transformers import pipeline, set_seed
    import matplotlib.pyplot as plt
    import cv2
```

```
[4] class CFG:
        device = "cuda"
        seed = 42
        generator = torch.Generator(device).manual_seed(seed)
        image_gen_steps = 35
        image_gen_model_id = "stabilityai/stable-diffusion-2"
        image_gen_size = (400,400)
        image_gen_guidance_scale = 9
        prompt_gen_model_id = "gpt2"
        prompt_dataset_size = 6
        prompt_max_length = 12
```

```
[5] image_gen_model = StableDiffusionPipeline.from_pretrained(
        CFG.image_gen_model_id, torch_dtype=torch.float16,
        revision="fp16", use_auth_token='hf_[redacted]', guidance_scale=9
    )
    image_gen_model = image_gen_model.to(CFG.device)
```

```
/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 secret in your Google Colab and restart your session.
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(
model_index.json: 100% 511/511 [00:00<00:00, 14.9kB/s]
/usr/local/lib/python3.11/dist-packages/diffusers/pipelines/pipeline_loading_utils.py:285: FutureWarning: You are loading the variant fp16 from stabilityai/stable-diffusion-2 via `rev
 The Diffusers team and community would be very grateful if you could open an issue: https://github.com/huggingface/diffusers/issues/new with the title `stabilityai/stable-diffusion-2
  warnings.warn(
Fetching 12 files: 100% 12/12 [00:17<00:00,  1.71s/it]
config.json: 100% 624/624 [00:00<00:00, 16.9kB/s]
config.json: 100% 900/900 [00:00<00:00, 12.1kB/s]
pytorch_model.bin: 100% 681M/681M [00:10<00:00, 115MB/s]
vocab.json: 100% 1.06M/1.06M [00:00<00:00, 5.97MB/s]
special_tokens_map.json: 100% 460/460 [00:00<00:00, 6.45kB/s]
scheduler_config.json: 100% 345/345 [00:00<00:00, 12.0kB/s]
tokenizer_config.json: 100% 815/815 [00:00<00:00, 12.5kB/s]
merges.txt: 100% 525k/525k [00:00<00:00, 4.26MB/s]
diffusion_pytorch_model.bin: 100% 1.73G/1.73G [00:17<00:00, 53.0MB/s]
diffusion_pytorch_model.bin: 100% 167M/167M [00:02<00:00, 70.7MB/s]
config.json: 100% 602/602 [00:00<00:00, 18.7kB/s]
Keyword arguments {'use_auth_token': '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', 'guidance_scale': 9} are not expected by StableDiffusionPipeline and will be ignored.
Loading pipeline components...: 100% 5/5 [00:02<00:00,  1.32it/s]
An error occured while trying to fetch /root/.cache/huggingface/hub/models--stabilityai--stable-diffusion-2/snapshots/d75b612d366d802b1753960de862a9270c8d55f1/vae: Error no file name
Defaulting to unsafe serialization. Pass `allow_pickle=False` to raise an error instead.
An error occured while trying to fetch /root/.cache/huggingface/hub/models--stabilityai--stable-diffusion-2/snapshots/d75b612d366d802b1753960de862a9270c8d55f1/unet: Error no file nam
Defaulting to unsafe serialization. Pass `allow_pickle=False` to raise an error instead.
```

```
[6] def generate_image(prompt, model):
        image = model(
            prompt, num_inference_steps=CFG.image_gen_steps,
            generator=CFG.generator,
            guidance_scale=CFG.image_gen_guidance_scale
        ).images[0]

        image = image.resize(CFG.image_gen_size)
        return image
```

```
[6]
```

2nd part

```
!pip install flask pyngrok

from flask import Flask
from pyngrok import ngrok

from flask import render_template_string, send_from_directory,request
import os


app = Flask(__name__)
IMAGE_FOLDER = "/content/drive/MyDrive/GENAI/genimage"
app.config["UPLOAD_FOLDER"] = IMAGE_FOLDER

# Function to execute after form submission
def execute_function(input_text):
    print(f"Function executed with input: {input_text}")
    gen_image=generate_image( input_text, image_gen_model)
    image_path = f"{app.config['UPLOAD_FOLDER']}/gen_image.png"
    cv2.imwrite(image_path,np_image)


html_content = """
<!DOCTYPE html>
<html>
<head>
    <title>Flask in Google Colab</title>
    <style>
        *{
            margin:0px;
            padding:0px;
        }


        body { font-family: Arial, sans-serif; text-align:center;height:100vh;width:100vw }
        button { padding: 10px 20px; font-size: 18px; margin: 10px; }
        #imageContainer { display: none; margin-top: 20px; }

        .contain{
            background: rgb(238,174,202);
            background: linear-gradient(90deg, rgba(238,174,202,1) 0%, rgba(148,187,233,1) 100%);
            width:100%;
            height:100%;
        }

        #headid{
            text-shadow: 2px 2px 5px red;
            color:white;

            padding:25px;
        }




        #inputid{
            width:350px;
            height:43px;
            border-radius:10px;

        }

        input[type=text] {
        width: 100%;
        padding: 12px 20px;
        margin: 8px 0;
        box-sizing: border-box;
        }

        #buttonid{
        color:white;
        background-color: blue;
        border-radius:10px;
        border:blue;
        }

        #loader_image{
          position:relative;
        }

        #loader {
          position:absolute;
          top:2%;
          left:45%;
          display: none;
          margin-top: 140px;
        }

        #displayImage{
            border:solid black 2px;
            border-radius:10px;
            margin:25px;
        }

    </style>

    <script>
        function showLoader() {
            document.getElementById('loader').style.display = 'block';
            document.getElementById('image-display').style.display = 'none';
        }

        function hideLoader() {
            document.getElementById('loader').style.display = 'none';
            document.getElementById('image-display').style.display = 'block';
        }
    </script>
</head>
<body>
<div class="contain">
    <h1 id="headid">Text to Image Generator</h1>

    <form method="post" action="/" onsubmit="showLoader()">
        <input id="inputid" type="text" name="user_input" placeholder="  Enter prompt here" required>
        <button id="buttonid" type="submit">Generate</button>
    </form>

    <div id="loader_image">

        <div id="loader">
            <svg width="180" height="180" viewBox="0 0 100 100">
                <circle cx="50" cy="50" r="40" stroke="white" stroke-width="12" fill="none" stroke-dasharray="200" stroke-dashoffset="200">
                    <animate attributeName="stroke-dashoffset" from="200" to="0" dur="2s" repeatCount="indefinite" />
                </circle>
            </svg>
        </div>


        {% if user_text %}
        <img id="displayImage" src="/get_image/gen_image.png" alt="Sample Image"  onload="hideLoader()" >
        {% endif %}
    </div>
    </form>
</div>
    <script>
    document.getElementById('loader').style.display = 'none';
    </script>

</body>
</html>
"""

# Define route
@app.route('/', methods=['GET', 'POST'])
def home():
    user_text = ""
    if request.method == 'POST':
        user_text = request.form.get('user_input', "")
        execute_function(user_text)  # Execute function on submit
    return render_template_string(html_content, user_text=user_text)


@app.route('/get_image/<filename>')

def get_image(filename):
    return send_from_directory(app.config["UPLOAD_FOLDER"], filename)

if __name__ == '__main__':
  ngrok.set_auth_token("[redacted]")
  ngrok_tunnel = ngrok.connect(5000)
  print('Public URL:', ngrok_tunnel.public_url)
  app.run()
```

```
Requirement already satisfied: flask in /usr/local/lib/python3.11/dist-packages (3.1.0)
Collecting pyngrok
  Downloading pyngrok-7.2.4-py3-none-any.whl.metadata (8.7 kB)
Requirement already satisfied: Werkzeug>=3.1 in /usr/local/lib/python3.11/dist-packages (from flask) (3.1.3)
Requirement already satisfied: Jinja2>=3.1.2 in /usr/local/lib/python3.11/dist-packages (from flask) (3.1.6)
Requirement already satisfied: itsdangerous>=2.2 in /usr/local/lib/python3.11/dist-packages (from flask) (2.2.0)
Requirement already satisfied: click>=8.1.3 in /usr/local/lib/python3.11/dist-packages (from flask) (8.1.8)
Requirement already satisfied: blinker>=1.9 in /usr/local/lib/python3.11/dist-packages (from flask) (1.9.0)
Requirement already satisfied: PyYAML>=5.1 in /usr/local/lib/python3.11/dist-packages (from pyngrok) (6.0.2)
Requirement already satisfied: MarkupSafe>=2.0 in /usr/local/lib/python3.11/dist-packages (from Jinja2>=3.1.2->flask) (3.0.2)
Downloading pyngrok-7.2.4-py3-none-any.whl (23 kB)
Installing collected packages: pyngrok
Successfully installed pyngrok-7.2.4
Public URL: https://5c1b-35-227-172-145.ngrok-free.app
 * Serving Flask app '__main__'
 * Debug mode: off
INFO:werkzeug:WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
 * Running on http://127.0.0.1:5000
INFO:werkzeug:Press CTRL+C to quit
```

https://5c19-35-227-172-145.ngrok-free.app

Text to Image Generator

Astronaut on horse

Generate

https://5c19-35-227-172-145.ngrok-free.app

Text to Image Generator

Enter prompt here

Generate