Documentation:

Theory:

Diffusion Modelling, inspired by non-equilibrium thermodynamics, (spreading data throughout the dataset, refining it over steps to obtain a high quality result).
This takes place over a couple of steps involving Gaussian distribution and neural networks.

The essential idea is to systematically and slowly destroy structure in a data distribution through an iterative forward diffusion process and then with the help of neural networks we restore structure in the data(reverse diffusion). That is how we obtain a traceable general model of the data.

Forward Process: Noise to the image is added in each iterative step(follows Normal Distribution).

Reverse Diffusion Process: Using a Neural network, the model gradually removes noise over a number of iterations.

Steps Involved:

- **Image Generation**: For Image Generation, I implemented the below models:

  - **SDXL Base 1.0:** (https://huggingface.co/docs/diffusers/en/using-diffusers/sdx). SDXL is based on Stable Diffusion model with a larger U-Net and 2 text encoders to increase the number of hyperparameters. It has an added refiner model which adds additional refinement details.
  - **SDXL Turbo:** (https://huggingface.co/stabilityai/sdxl-turbo). It is a refined version of SDXL 1.0, based on ADD(Adversarial Diffusion Distillation), with which we can sample image diffusion models at high image quality
  - **Runway-ML:** (https://huggingface.co/runwayml/stable-diffusion-v1-5). This is a re-initialized Stable Diffusion 1.5 checkpoint with the weights of Stable-Diffusion-1.2 checkpoint and further fine-tuned.
  - **DreamLike PhotoReal 2.0:** (https://huggingface.co/dreamlike-art/dreamlike-photoreal-2.0). This model is based on Stable Diffusion 1.5, made and modified by dreamlike.art .

- I tried a few other implementations as well(not mentioned in the notebook), and found **DreamLike performing the best overall, with SDXL Base 1.0 doing slightly better in some use cases**.

- **Image Upscaling**: For Image Upscaling,I tried using the following implementations:

- **OpenCV's cv2**: (Not mentioned in the notebook): After implementing this, we found that it was just resizing the image to 2048x2048 scales with loss in quality, due to interpolation(filling the missing nodes based on neighbouring nodes).
- **Swin2SR x4 Scaler:** (https://huggingface.co/caidas/swin2SR-realworld-sr-x4-64-bsrgan-psnr). This model is based on the Swin2SR research paper: (https://arxiv.org/abs/2209.11345).

❖ Problems Faced:

- Running out of GPU resources in most pipelines:
  To fix this error, we enable cpu offloading by removing: **pipe.to("cuda")** and replacing it with **pipe.enable_model_cpu_offload(). Shouldn't happen on Google Colab with A100 Gpu.**

- Image Resizing Issues:
  From our model, we get our image in the format of 768x768 pixels, after upscaling it was 1512x1512. Hence, we resize our initial image to 1024x1024 pixels.
  From Hereon, after scaling the size is 2052x2052. So, for the final image we resize it 2048x2048. (a/c to documentation).

Observation:
From our findings, we observed DreamLike to give the most photorealistic images (SDXL Base in a few cases), with efficient time-resource usage. For further upscaling, without loss in quality, Swin2SR is used and we obtain our final image of 2048x2048 pixels from a given prompt.

- SDXL Base 1.0 had a lot of errors in the images it was forming. SDXL Turbo was making images with a bluish-green tint to them. Runway-ML was slightly better but unable to form faces and other complex features. DreamLike also has slight issues while forming faces but performs well the majority of the time.
- Swin2SR is slightly slow but successfully upscales our image, (~a few pixels greater) which is then resized to our final requirements(2048x2048).
- Added a few lines of code(commented), which can be used to save the images, if required.

NOTE: Please Give in the prompt at line 7.