Contents

[**Introduction** 3](#_Toc157609702)

[**Project Overview** 3](#_Toc157609703)

[The Generative AI project aims to generate, alter Images using generative artificial intelligence models for image2image and text2image generation. The web application allows users to utilize different generative AI capabilities through an intuitive interface. 3](#_Toc157609704)

[**Objectives** 3](#_Toc157609705)

[**Technologies Used** 3](#_Toc157609706)

[**Programming Languages** 3](#_Toc157609707)

[**Libraries and Frameworks** 3](#_Toc157609708)

[**Project Setup** 4](#_Toc157609709)

[**Installation Instructions** 4](#_Toc157609710)

[**Code Overview** 4](#_Toc157609711)

[**Directory Structure** 4](#_Toc157609712)

[**Main Code Files** 5](#_Toc157609713)

[**Methodology** 5](#_Toc157609714)

[**Text to Image Generation** 5](#_Toc157609715)

[**Image to Image Generation** 5](#_Toc157609716)

[**From Canny Image Generation** 5](#_Toc157609717)

[**From Depth Image Generation** 5](#_Toc157609718)

[**From Segmentation Image Generation** 5](#_Toc157609719)

[**Usage Guide** 6](#_Toc157609720)

[**Running the Code** 6](#_Toc157609721)

[**Interacting with the Interface** 6](#_Toc157609722)

[**Keyboard Shortcuts** 6](#_Toc157609723)

[**System Diagrams** 6](#_Toc157609724)

[**Architecture Diagram** 7](#_Toc157609725)

[**Results and Visualization** 7](#_Toc157609726)

[**Sample working Frame** 7](#_Toc157609727)

[**Performance Evaluation** 7](#_Toc157609728)

[**Quality of Generation** 7](#_Toc157609729)

[**Future Enhancements** 7](#_Toc157609730)

[**Ideas for Improvement** 7](#_Toc157609731)

[**Feature Additions** 8](#_Toc157609732)

[**Conclusion** 8](#_Toc157609733)

[**Project Summary** 8](#_Toc157609734)

[**Achievements**. 8](#_Toc157609735)

[**Closing Remarks** 8](#_Toc157609736)

[**Appendix** 8](#_Toc157609737)

[**Installation Troubleshooting** 8](#_Toc157609738)

[**Glossary of Terms** 8](#_Toc157609739)

[**References** 8](#_Toc157609740)

**Stable Diffusion (Img2Img, Txt2Img) Project Documentation**

**Introduction**

## **Project Overview**

### The Generative AI project aims to generate, alter Images using generative artificial intelligence models for image2image and text2image generation. The web application allows users to utilize different generative AI capabilities through an intuitive interface.

### **Objectives**

The key features of the project include:

* **Text-to-Image generation** - Users can provide a text prompt and generate a corresponding image through AI models like Stable Diffusion. Options for faster low-quality generation or slower high-quality generation.
* **Image-to-Image generation** - Users can upload an image and choose transforms like depth estimation, Canny edge detection, segmentation, and then provide a text prompt to generate a modified image.
  1. **Segment Anything Module** - Performs object detection and segmentation on uploaded images using SAM model. Detected objects are provided as options for image-to-image generation.
  2. **GroundingDINO Module** - Detects and labels objects in images to enable segmentation and region-based image editing.

The web interface allows intuitive interaction with the AI models through text prompts or uploaded images. Additional features like text-to-video can also be incorporated.

**Technologies Used**

## **Programming Languages**

* Python (For backend and GenAI)
* Flask (For server-side scripting and acting as controller between Front-End and Back-End)
* HTML, Tailwind-CSS, JavaScript (For Front-End)

## **Libraries and Frameworks**

* Flask
* PyTorch
* GroundingDino
* Segment\_anything (SAM)
* Diffusers
* Transformers
* Urllib
* NumPy
* Glob
* Opencv-python (cv2)
* os
* Random
* Pillow (PIL)

**Project Setup**

## **Installation Instructions**

* Install Python (version 3.9.x or higher) from [Python Official Website](https://www.python.org/downloads/).
* Install required packages using pip:

**bashCopy code:**

pip3 install torch torchvision torchaudio --index-url https://download.pytorch.org/whl/cu121

and

pip install -U diffusers transformers accelerate controlnet\_aux opencv-contrib-python flask

* + or can visit PyTorch website for latest code: [PyTorch Official Website](https://pytorch.org/).

**Code Overview**

## **Directory Structure**

* **GroundingDINO**/: git cloned directory as it is yet not available in pypi.
* **static/**: Directory containing static code for web app.
* **ControlNet\_Image/**: Directory containing where ControlNet themed image will be saved (Canny, Depth, Segmented) after conversion from app.py.
* **images/**: Directory containing all model’s generated output images will be saved to display on web app.
* **user\_uploads**: Directory where user’s uploaded image will be saved from img2img module.
* **png, svg images**: Static images displayed on UI.
* **templates/**: Directory which contains all html code files of webapp.
* **img2img.html**: This is the page for img2img module.
* **text2img.html**: This is the page for text2img module.
* **app.py**: Main code file to run the web application (Flask code).
* **Run\_Img2Img\_Inference\_Canny.py**: Script for canny-based generation in img2img module.
* **Run\_Img2Img\_Inference\_Depth.py**: Script for depth-based generation in img2img module.
* **Run\_text2Img\_Inference\_Fast.py**: Script for fast but less precise generation in text2img module.
* **Run\_text2Img\_Inference\_Quality.py**: Script for slow but most precise generation in text2img module.
* **SD\_SAM.py**: Script for segmentation generation in img2img module.
* **SDXL-Turbo.ipynb**: Jupyter Notebook containing all practice scripts and in their basic form (non-deployable).

## **Main Code Files**

* **app.py**: Contains the main logic for connecting scripts and to run the web application.

**Methodology**

## **Text to Image Generation**

The project processes user’s input (query) and generates an image which is then displayed on UI and can be downloaded.

## **Image to Image Generation**

The project processes user’s input (query) and uploaded image then generates an image as per his/her instructions which is then displayed on UI and can be downloaded.

### **From Canny Image Generation**

User have to select (check) the canny option box, then upload the image. Then, he will be rendered sketched image on UI and then has to provide a prompt (input or query or text) that he wants to generate on that canny image.

### **From Depth Image Generation**

User have to select (check) the depth option box, then upload the image. Then, he will be rendered depthified image on UI and then has to provide a prompt (input or query or text) that he wants to generate on that depth image.

### **From Segmentation Image Generation**

User have to select (check) the segmentation option box, then upload the image. Then, he will be rendered segmented image on UI along with the classes detected on his uploaded image and then he has to first select one of the class (that he wants to change) and then provide a prompt (input or query or text) that he wants to generate on that segmented image (his selected class will be replaced by the generated object).

**Note:-> For Image-to-Image module, user have to provide a short query describing features of the required object. More precise the query, better the results. Also, App can replace any object using its shape so user cannot change the dimensions of the object like we cannot make a detected person with a cat or any other object as both have different dimensions but segmentation Option might do a little better on such case than other canny or depth options.**

**Usage Guide**

## **Running the Code**

1. Clone the repository and navigate to the project directory.
2. Install all dependencies.
3. Run the following command to execute the code:

**Run with simple code (terminal command):** python app.py

## **Interacting with the Interface**

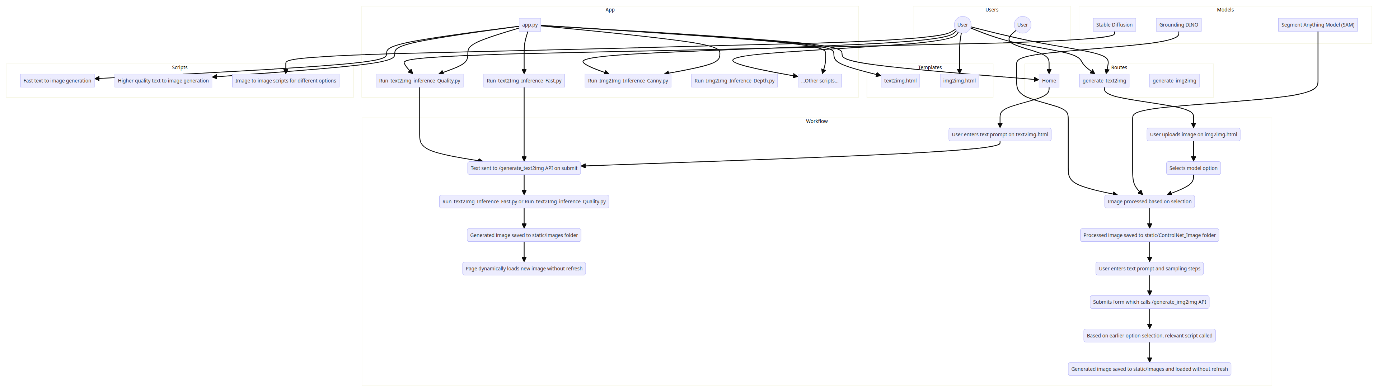
* Use the interface and follow pages.

## **Keyboard Shortcuts**

* Click on cross button “x” on top right to close the webapp.

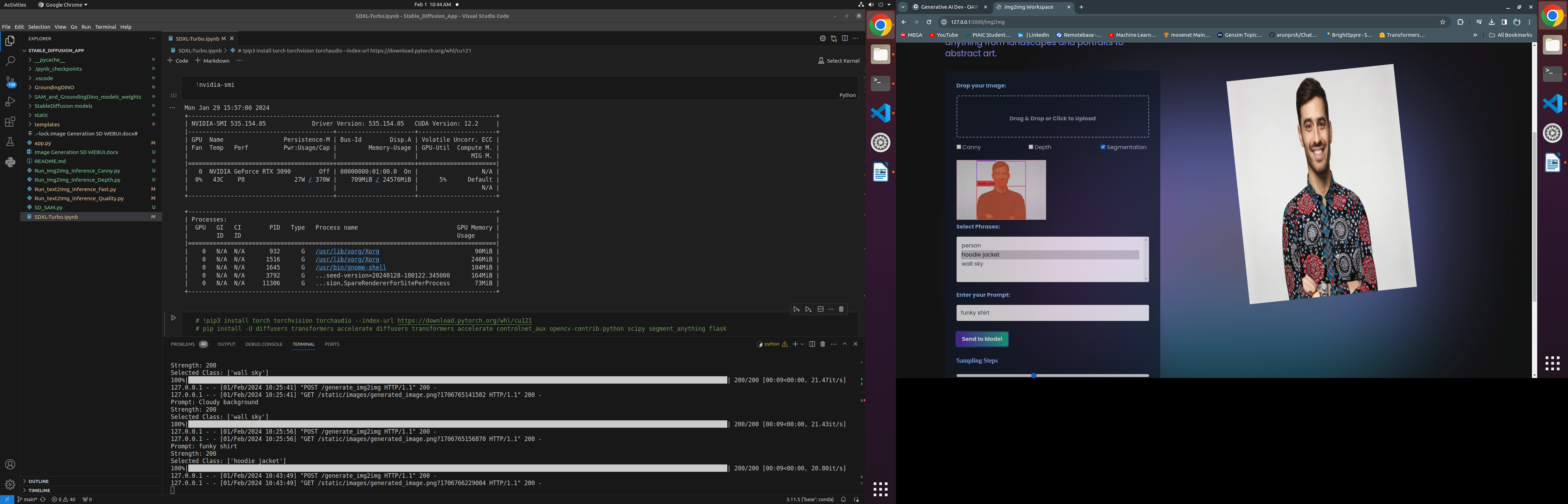
**System Diagrams**

## **Architecture Diagram**



**Results and Visualization**

## **Sample working Frame**



**Performance Evaluation**

## **Quality of Generation**

* User’s prompt highly affects the overall quality of generation.

**Future Enhancements**

## **Ideas for Improvement**

* Enhance prompt quality for better generations (one idea is to train a model such as BERT transformer and evaluate user’s prompts just like passwords are done weal, better, good).
* Update the model’s regularly if a newer version is available.
* Increase segmentation by adding custom (user pointed) selection area on the image. Idea is just like s24 ultra’s app do (grants user’s access to tap and select what to alter).



* Include Image upscaler module to provide user upto 4k resolution image as output reason is SD models can generate or alter upto 512 x 512 images.

## **Feature Additions**

* Add interactive and user-friendly UI.
* Implement support for multi user engagement capability.
* Create an additional page that will be used to display user’s generations on his/her consent.
* Add session handling and hence, user login for multi user capability.
* Make UI responsive to all types of displays.

**Conclusion**

## **Project Summary**

This project successfully implements Generative AI techniques to modify and speedup content generation. The project demonstrates the potential of Generative AI.

## **Achievements**.

* It makes editing and images generation 60% faster and cost effective using AI.

## **Closing Remarks**

The project showcases the power of combining computer vision and machine learning to provide image generation for content creation.

**Appendix**

## **Installation Troubleshooting**

* If facing issues during installation, refer to the Troubleshooting section in the README.

## **Glossary of Terms**

* Definitions of key terms used throughout the documentation. (To be included later)

## **References**

* List of resources, tutorials, and references used during the project. (To be included later)