NUS Generative AI: Fundamentals to Advanced Techniques

Week 10: Capstone Project Part 3.2

Done by: A Alkaff Ahamed

A Alkaff Ahamed
17 June 2025

Assignment Instructions

Learning Outcome Addressed

- Develop effective prompts to generate high-quality, contextually-relevant images
- Utilise APIs for seamless integration and adapt models for specific use cases

Objective

Learn to use image generation APIs programmatically through Replicate.com and OpenAI's DALL-E API.

Prerequisites

- Python 3.8+
- GitHub account
- Text editor or IDE
- Terminal/Command Prompt
- Free Replicate.com account
- OpenAI account (optional - requires credit card)

Tasks

- **Part 1: Replicate.com Setup (5 minutes)**
  1. Visit replicate.com and sign in with GitHub, Get your API token and use
- **Part 2: Basic Stable Diffusion Implementation (7 minutes)**
  1. Create a new Python file named `sd_generate.py`
- **Part 3: Experimenting with Prompts (8 minutes)**
  1. Modify your script to try these variations
  2. Create your own prompt variations
  3. Experiment with different seeds for the same prompt
- **Extension: DALL-E API (Optional)**
  1. Visit platform.openai.com, study the Images API documentation, use it

Submission Requirements

Create a document containing:

1. Screenshots of at least 3 generated images
2. The prompts used for each image
3. Parameter variations tried (steps, seeds)
4. Brief comparison of results with different parameters

Tips

- Always check the model's current name on Replicate.com as it may change
- Start with simple prompts and gradually add complexity
- Keep track of seeds that produce good results
- Monitor your API usage on both platforms

Common Issues

- API token not set correctly: Check environment variable
- Model name changed: Verify current model name on Replicate.com
- Rate limits: Space out your requests
- TypeError: Check parameter types (seeds must be integers)

Note on API Changes

APIs evolve regularly. Always refer to:

- com model documentation
- OpenAI API documentation for the most current implementation details.

Estimated time: 60-90 minutes

Submission Instructions:

- Select the **Start Assignment** button at the top right of this page.
- Upload your answers in the form of a Word or PDF file.
- Select the **Submit Assignment** button to submit your responses.

This is a graded and counts towards programme completion. You may attempt this assignment only once.

Assignment Answers

Part 1: Replicate.com Setup

1. Visit replicate.com and sign in with GitHub
2. Get your API token:
    - Go to Account Settings
    - Copy your API token

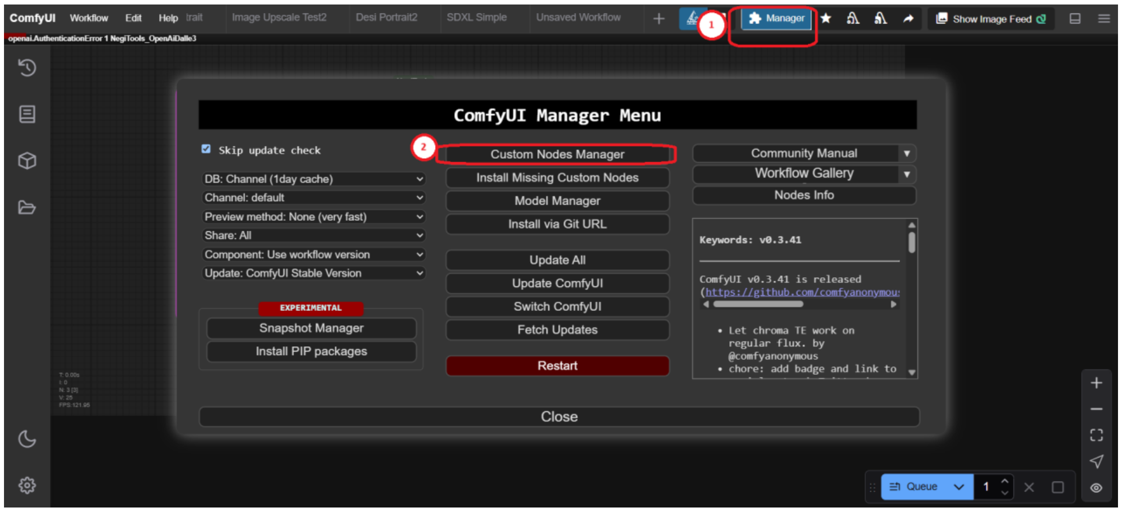
In your terminal:

```
export REPLICATE_API_TOKEN=your_token_here
```

Install the Python client:

```
pip install replicate
```

At the time of assignment, replicate.com is no longer free. I have decided to run the prompts from my local setup using ComfyUI. The instructions below are for downloading and setting up ComfyUI:

1. Download the windows release (`ComfyUI_windows_portable_nvidia.7z`) from https://github.com/comfyanonymous/ComfyUI
2. Extract the file and use any terminal/git to `cd ComfyUI_windows_portable\ComfyUI\custom_nodes`
3. Clone the ComfyUI Manager's GitHub Repository inside the `custom_nodes` folder
   `git clone https://github.com/ltdrdata/ComfyUI-Manager comfyui-manager`
4. Start ComfyUI by double click either the `run_cpu.bat` or `run_nvidia_gpu.bat`, it will open in your browser
5. (Optional, only required if using OpenAPI) Go to `Manager > Custom Nodes Manager` and search for "ComfyUI-NegiTools", then install it
   

   openai.AuthenticationError 1 NegiTools_OpenAiDalle3

   [✓] Skip update check

   DB: Channel (1day cache)
   Channel: default
   Preview method: None (very fast)
   Share: All
   Component: Use workflow version
   Update: ComfyUI Stable Version

   ```
   Keywords: v0.3.41
   ComfyUI v0.3.41 is released
   (https://github.com/comfyanonymou
     • Let chroma TE work on
       regular flux. by
       @comfyanonymous
     • chore: add badge and link to
   ```
6. (Optional, only required if using OpenAPI) Edit the `run_cpu.bat` or `run_nvidia_gpu.bat` by adding the OpenAPI key:
   ```
   set OPENAI_API_KEY=sk-xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx  # ADD THIS LINE
   .\python_embeded\python.exe -s ComfyUI\main.py ...
   pause
   ```
7. Restart ComfyUI and refresh the browser

Part 2: Basic Stable Diffusion Implementation

1. Create a new Python file named `sd_generate.py`
2. Basic implementation:

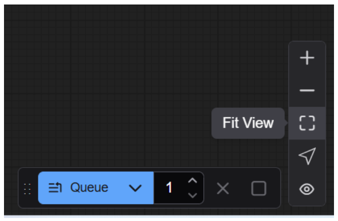
```
import replicate
import os

# The prompt for image generation
prompt = "a white siamese cat"

# Run the model
output = replicate.run(
"stability-ai/stable-diffusion-3.5-medium",
    input={
        "prompt": prompt,
        "seed": 42,
        "steps": 30,
        "format": "png"
    }
)

print(f"Image URL: {output[0]}")
```

3. Try different parameters:
   - Adjust steps (20-50)
   - Try different seeds
   - Test various output formats

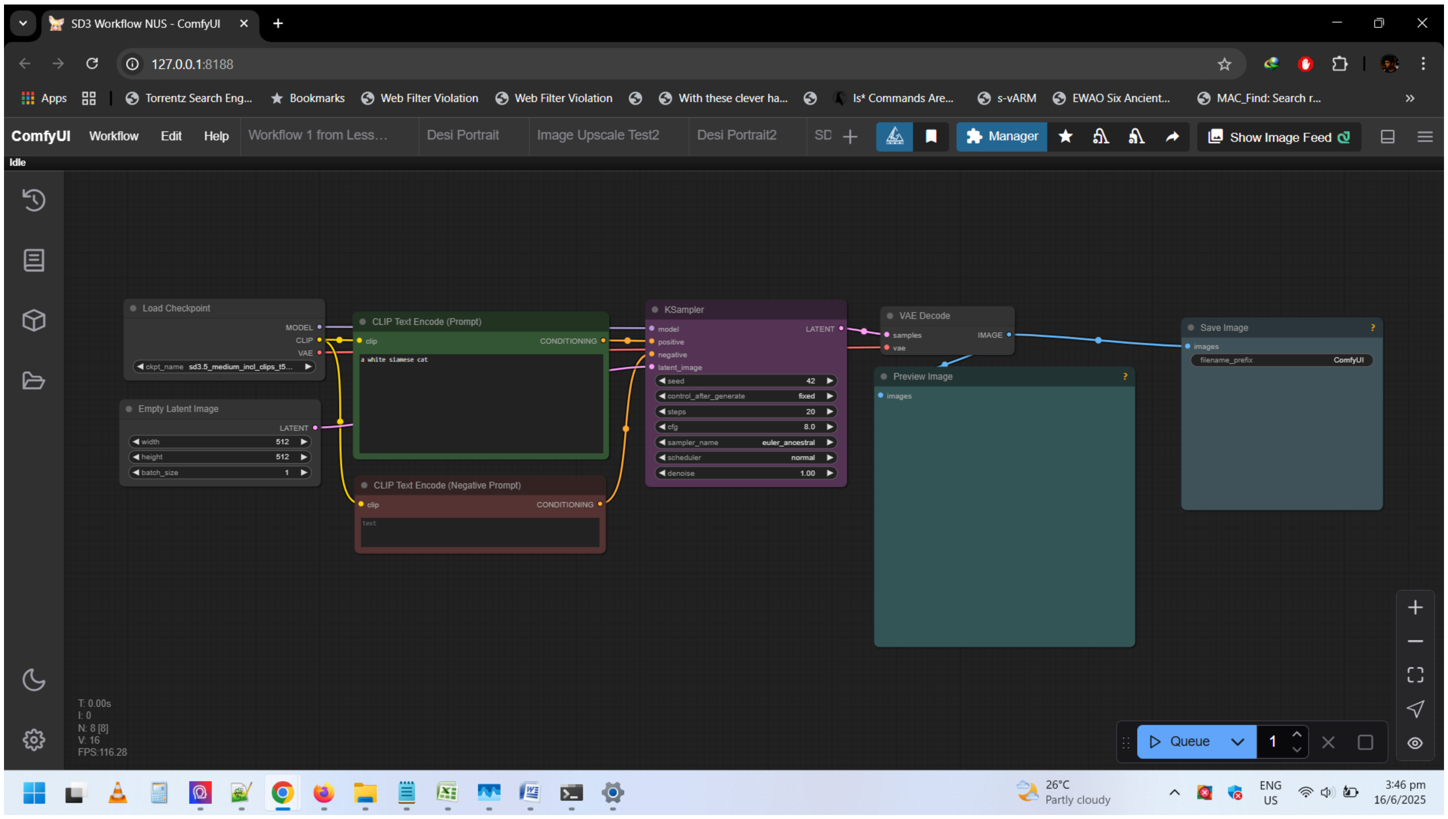
Setup ComfyUI

At the time of assignment, replicate.com is no longer free. I have decided to run the prompts from my local setup using ComfyUI. The instructions below are for downloading the model and setting up the workflow for running Stable Diffusion 3.5 model locally:

1. Download this file (`sd3.5_medium_incl_clips_t5xxlfp8scaled.safetensors`) from here https://huggingface.co/Comfy-Org/stable-diffusion-3.5-fp8/tree/main
2. Once downloaded, put the file inside this directory: `ComfyUI_windows_portable\ComfyUI\models\checkpoints`
3. Drag and Drop the `SD3 Workflow NUS.json` file into the ComfyUI workspace
4. If the workspace looks empty, click the "Fit View" option to bring the zoom to the workflow



5. Mouse Wheel for zoom, Drag for moving around in the workspace
6. Adjust the positive and negative prompts and KSampler values accordingly in their respective nodes
7. Then click "Queue", the program will run
8. Generated images will be stored in `ComfyUI_windows_portable\ComfyUI\output` directory



ckpt_name sd3.5_medium_incl_clips_t5...

a white siamese cat

width 512
height 512
batch_size 1

seed 42
control_after_generate fixed
steps 20
cfg 8.0
sampler_name euler_ancestral
scheduler normal
denoise 1.00

filename_prefix ComfyUI

Figure 1: Screenshot showing the loaded workflow for Stable Diffusion 3.5

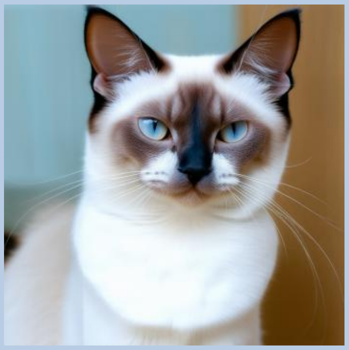
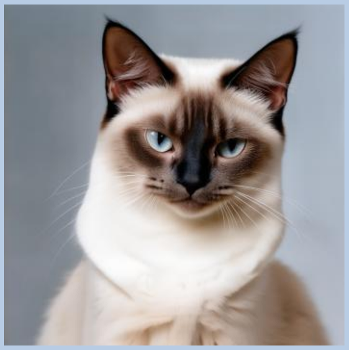
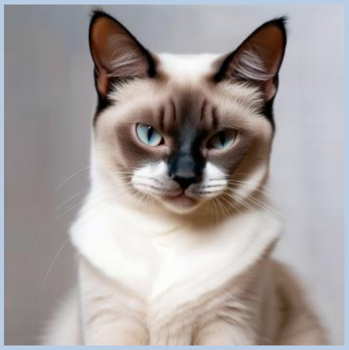
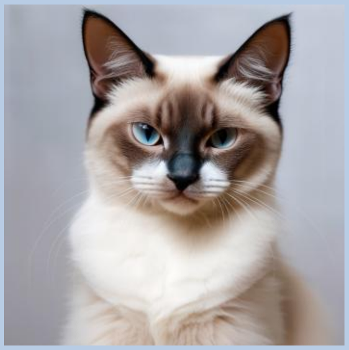
Nodes Explained:

- **Empty Latent Image**: This is where you define the image size to generate
- **CLIP Text Encode (Prompt)**: This is where the main prompt is entered to describe the image you want to generate.
- **CLIP Text Encode (Negative Prompt)**: This is where you enter negative prompts to specify elements you want the model to avoid (e.g., "blurry", "distorted").
- **KSampler**: This node allows you to configure the generation parameters, such as: Seed, Sampling steps, Sampler method (e.g., euler_ancestral), CFG (guidance strength)
- **Save Image**: This is where you define the output file name and location for the generated image.

Varying the Steps 20 to 50

Following settings were used:

- **Prompt**: a white siamese cat
- **Negative Prompt**: -
- **Seed**: 42
- **CFG**: 7.5
- **Sampler Name**: euler_ancestral (Euler A)
- **Scheduler**: Normal
- **Steps**: 20, 30, 40, 50
- **Image Size**: 768x768

Steps = 20	Steps = 30	Steps = 40	Steps = 50
			

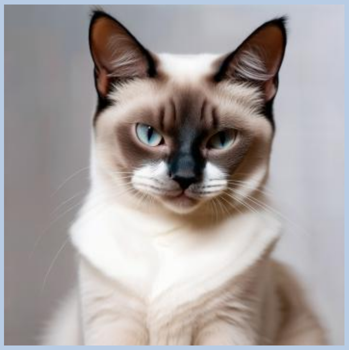
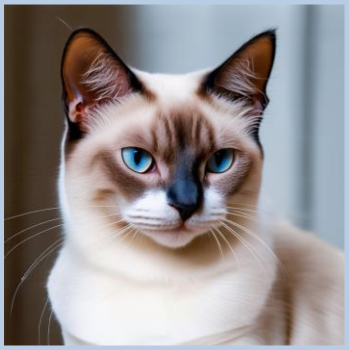
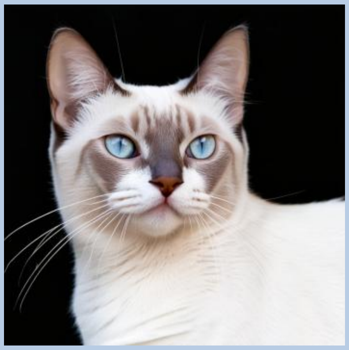
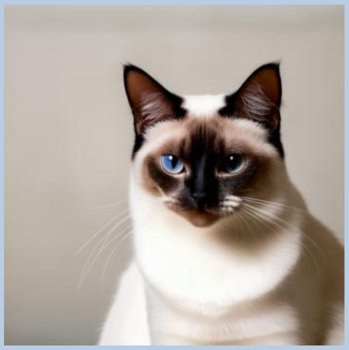
Observations:

- **20 steps**: Basic structure visible but blurry and underdeveloped
- **30 steps**: Better detail and form, slight improvement in facial features
- **40 steps**: Clear eyes, realistic texture, symmetrical structure — best overall
- **50 steps**: Slight refinements but no significant visual gain over 40
- **Conclusion**: Use 40 steps as it is the optimum

Varying the Seeds

Following settings were used:

- **Prompt**: a white siamese cat
- **Negative Prompt**: -
- **Seed**: 42, 43, 44, 45
- **CFG**: 7.5
- **Sampler Name**: euler_ancestral (Euler A)
- **Scheduler**: Normal
- **Steps**: 40
- **Image Size**: 768x768

Seed = 42	Seed = 43	Seed = 44	Seed = 45
			

Observations

- Changing the seed value results in visually unique outputs even when all other parameters remain constant.
- This demonstrates how seeds are useful for exploring diverse variations of a concept or prompt.
- For consistent results or reproducibility, keep seed fixed. For variety, change the seed.

Part 3: Experimenting with Prompts

1. Modify your script to try these variations:

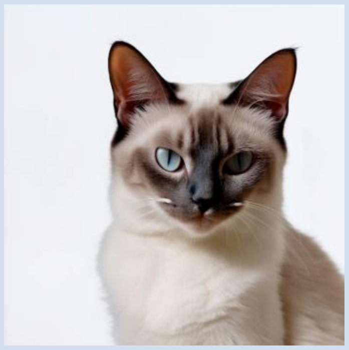
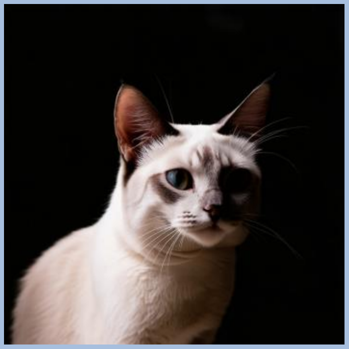
```
prompts = [
"a white siamese cat, studio lighting",
    "a white siamese cat, dramatic lighting, dark background",
    "a white siamese cat, outdoor setting, natural light"
]

for prompt in prompts:
    output = replicate.run(
        "stability-ai/stable-diffusion-3.5-medium",
        input={
            "prompt": prompt,
            "seed": 42,
            "steps": 30,
            "format": "png"
        }
    )
    print(f"Prompt: {prompt}")
    print(f"Image URL: {output[0]}\n")
```

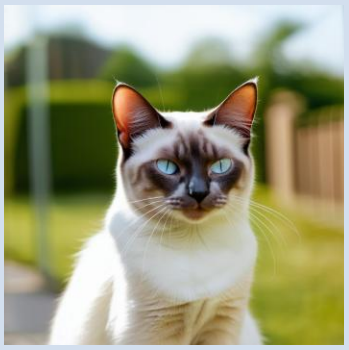
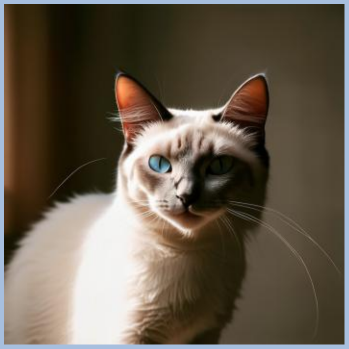
2. Create your own prompt variations
3. Experiment with different seeds for the same prompt

Prompt Variations

Following settings were used:

- **Prompts:** [
  "a white siamese cat, studio lighting",
  "a white siamese cat, dramatic lighting, dark background",
  "a white siamese cat, outdoor setting, natural light",
  " a white siamese cat, cinematic lighting, vintage film style",
  ]
- **Negative Prompt**: -
- **Seed**: 42
- **CFG**: 7.5
- **Sampler Name**: euler_ancestral (Euler A)
- **Scheduler**: Normal
- **Steps**: 40
- **Image Size**: 768x768

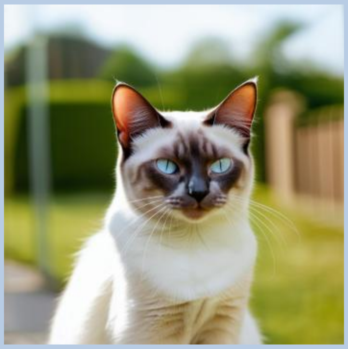
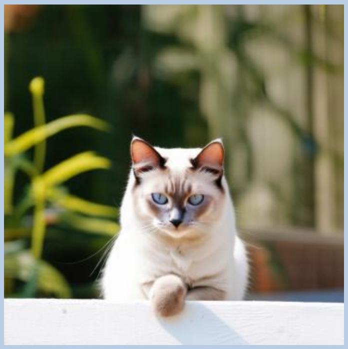
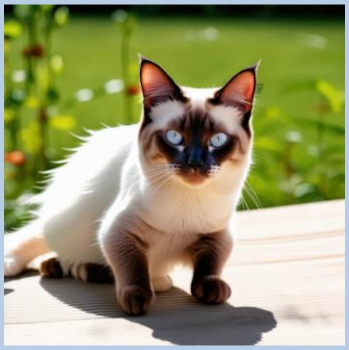
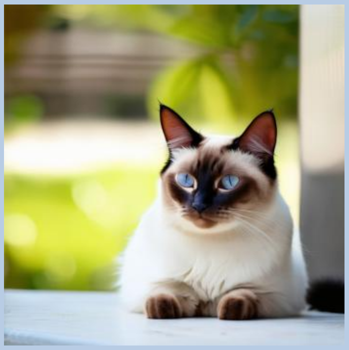
	**Prompt 1**: a white siamese cat, studio lighting **Observation**: Very standard and ideal for catalog-style clarity.
	**Prompt 2**: a white siamese cat, dramatic lighting, dark background **Observation**: Prominent cinematic feel emphasizes cat's facial structure

	Prompt 3**:** **a white siamese cat, outdoor setting, natural light** Observation**: Realistic and vivid, background complements subject.**
	**Prompt 4**: a white siamese cat, cinematic lighting, vintage film style **Observation**: Stylized and artistic, evokes a nostalgic or editorial quality.

Seed Variations

Following settings were used:

- **Prompts:** a white siamese cat, outdoor setting, natural light
- **Negative Prompt**: -
- **Seed**: 42, 43, 44, 45
- **CFG**: 7.5
- **Sampler Name**: euler_ancestral (Euler A)
- **Scheduler**: Normal
- **Steps**: 40
- **Image Size**: 768x768

Seed = 42	Seed = 43	Seed = 44	Seed = 45
			

Observations

- Changing the seed value results in visually unique outputs even when all other parameters remain constant.
- This demonstrates how seeds are useful for exploring diverse variations of a concept or prompt.
- For consistent results or reproducibility, keep seed fixed. For variety, change the seed.

Extension: DALL-E API (Optional)

1. Visit platform.openai.com
2. Study the Images API documentation
3. Basic implementation:

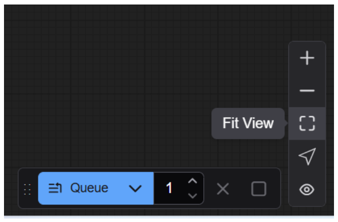
```
from openai import OpenAI
client = OpenAI()

response = client.images.generate(
model="dall-e-3",
    prompt="a white siamese cat",
    size="1024x1024",
    quality="standard",
    n=1,
)

print(response.data[0].url)
```

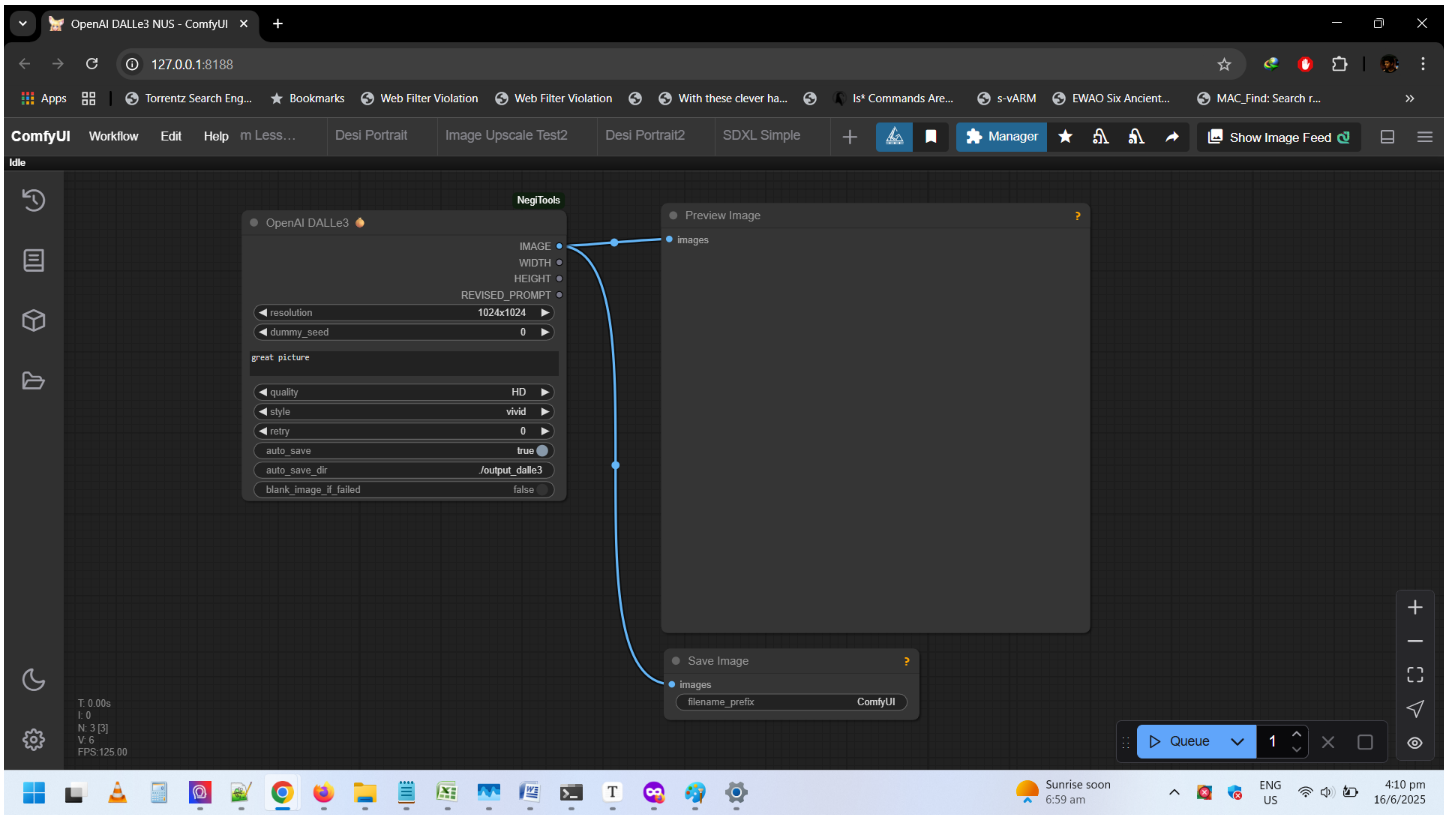
ComfyUI can be setup to connect with OpenAI using OpenAI API Key.

In the Part 1 instructions, step 6 and 7 are required for running this section.

After OpenAI API is properly setup, follow these instructions:

1. Drag and drop the `OpenAI DALLe3 NUS.json` file to the workspace
2. If the workspace looks empty, click the "Fit View" option to bring the zoom to the workflow





OpenAI DALLe3
resolution 1024x1024
dummy_seed 0
great picture
quality HD
style vivid
retry 0
auto_save true
auto_save_dir ./output_dalle3
blank_image_if_failed false

Save Image
filename_prefix ComfyUI

Figure 2: Screenshot showing the loaded workflow for OpenAI DALLe 3

Nodes Explained:

- **OpenAI DALLe3**: This is where you set the prompt and other input parameters for generating the image
- **Save Image**: This is where you define the output file name and location for the generated image.