Training day 5 report

Image Generation :

- This Python script uses **Hugging Face's** stabilityai/stable-diffusion-xl-base-1.0 model to generate an image based on a user-provided prompt.
- The resulting image is saved locally and optionally opened in the default viewer.

Main Features and Flow:

Authentication

- Requires a **Hugging Face API token** (HF_API_TOKEN) for access.
- Script stops execution if the token is missing.

Dynamic Prompt Input

- Asks the user to **input a text prompt** to describe the image.
- Also asks for a **filename** to save the image (defaults to output.png if blank).

Filename Validation

- Validates the filename to ensure it **doesn't contain illegal characters** (\/:*?"<>|).

Image Generation

- Sends a **POST request** with the prompt to the Hugging Face Inference API.
- Expects an **image (binary data)** in response.
- Uses PIL.Image.open() to open the image and image.save() to save it locally.

Output Storage

- The image is stored in the **current working directory** with the provided filename.
- The **absolute path** of the saved image is displayed.

Auto-Open Feature

- Detects the OS (Windows, macOS, or Linux) and opens the image in the **default image viewer**.

Error Handling

- Handles HTTP errors, network issues, timeout, or if the response is not a valid image.

Properties of the Output Image

- **Format**: Default is PNG (output.png), but the user can specify any valid filename like .jpg, .jpeg, etc.
- **Size & Quality**: Depends on the model's response; SDXL usually produces high-quality images, typically **1024x1024 resolution**.
- **Saved Location**: In the **same folder where the script is run**.
- **Automatically opened** after generation for instant viewing.

Example output flow:

Enter your image prompt: A futuristic city at sunset

Enter filename to save image (default: output.png): sunset_city.png

Image generated and saved to:

C:\Users\YourName\Desktop\sunset_city.png

Image Modifier (Ghibli Style) :

Purpose of the Code:

- This script **takes an existing image** as input (uploaded by the user) and sends it to an AI model (like **ghibli-diffusion**, image-to-image stylization, or ControlNet) to **convert it into a Ghibli-style version**.
- The modified image is then **saved** and optionally **displayed**.

Main Features and Flow:

Input Image Upload

- Asks the user to **upload or select an image file** (like .jpg, .png).
- Validates if the file is an actual image using PIL.Image.open().

Style Prompting (Optional)

- May optionally include a **text prompt** like:
  "Convert this image into Studio Ghibli anime style"
  or
  "Ghibli-inspired transformation"
  depending on the model's API.

Image-to-Image API Request

- Sends the uploaded image to an **image-to-image transformation API**, like:
  - stabilityai/stable-diffusion-img2img
  - SG161222/Realistic_Vision_V5.1
  - Or a custom Ghibli-style model
- Uses multipart/form-data or Base64 encoding to send the image.

Modified Image Output

- Receives the **modified image** in response.
- Converts it from bytes to an image object and **saves** it locally.

Output File

- Automatically names the new file something like ghibli_output.png or appends _ghibli to the original filename.

Auto-Open Feature

- Opens the newly generated image using the default image viewer depending on OS.

Properties of Modified Image:

Property	Description
Style	Ghibli-inspired (soft lines, anime shading, fantasy palette)
Resolution	Typically 512x512 to 1024x1024 (depends on the model)
File Type	PNG or JPEG
Saved Path	Current directory with new filename
Preserves Layout	Yes, but with stylized elements (like sky, buildings, humans)
Color Palette	Lighter, dream-like tones (Ghibli aesthetics)

Example Output Flow:

```
CopyEdit
 Enter image path: myphoto.jpg
 Applying Ghibli transformation...
 Modified image saved as: myphoto_ghibli.png
 Auto-opening the image in default viewer...
```

Error Handling

- Checks for:
  - Invalid or corrupted image file
  - API errors or timeouts
  - Unsupported file formats
- Provides user-friendly messages like:
  - "❌ Please upload a valid image file"
  - "❌ Error applying style. Try a clearer input image or prompt"