🎁 Installing Required Packages

```
pip install requests pillow
```

This command (run in the terminal or command prompt) installs:

- `requests`: for making HTTP requests.
- `pillow`: a Python imaging library used to open, manipulate, and save image files.

📦 Importing Libraries

```
import requests
from PIL import Image
from io import BytesIO
import time
```

- `requests`: used to send requests to Hugging Face's API.
- `PIL.Image`: used to display and save the generated image.
- `BytesIO`: allows you to treat image bytes like a file (needed to open image from the API response).
- `time`: used to pause if the model is still loading.

🔐 Set API Token

```
API_TOKEN = "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"
headers = {
    "Authorization": f"Bearer {API_TOKEN}"
}
```

- `API_TOKEN`: your Hugging Face personal access token — required to access private or rate-limited models.
- `headers`: sets the authorization in the HTTP request so the API knows it's you.

🖊️ Prompt Input

```
prompt = input("Enter a prompt to generate an image: ")
```

- Asks the user to type a **text description** (prompt), which the model will use to generate the image.

🌐 API URL Setup

```
api_url = "https://api-inference.huggingface.co/models/black-
forest-labs/FLUX.1-dev"
```

- This is the endpoint for the **FLUX.1-dev model** hosted on Hugging Face.
- This specific model turns text prompts into images.

📡 Send API Request + Wait if Model is Loading

```
response = requests.post(api_url, headers=headers,
json={"inputs": prompt})
while response.status_code == 503:
    print("Model is loading... waiting 10 seconds.")
    time.sleep(10)
    response = requests.post(api_url, headers=headers,
json={"inputs": prompt})
```

- Sends the text prompt to the model using a POST request.
- If the model is **still loading (503 response)**, waits 10 seconds and retries.
- Keeps checking until the model is ready.

🖼️ Process API Response

```
if response.status_code == 200:
    image = Image.open(BytesIO(response.content))
    image.show()
    image.save("generated_image.png")
    print("✅ Image saved as 'generated_image.png'")
```

- If the response is successful (`200 OK`):
  - Converts the returned image bytes into an image using `BytesIO` and `Image.open`.
  - `image.show()`: opens the image using the system's default image viewer.
  - `image.save("generated_image.png")`: saves the image to your working directory.
  - Prints a success message.

❌ Handle Errors

```
else:
    print("❌ Error:", response.status_code)
    print(response.text)
```

- If the API request fails (not `200 OK`), prints the HTTP status code and error message from the API.

Let me know if you'd like to improve this code — for example, by saving images with unique filenames or adding error logging.