OpenAI API

+

FlutterFlow

Project Three

Image Generator

Quick Short Project

AI Visualizer is a creative tool built with FlutterFlow and powered by the OpenAI Image Generation API

Its purpose is to enable dynamic visual creation by instantly transforming user text prompts into high-quality images using a cost-efficient model.

1. Context and Overview

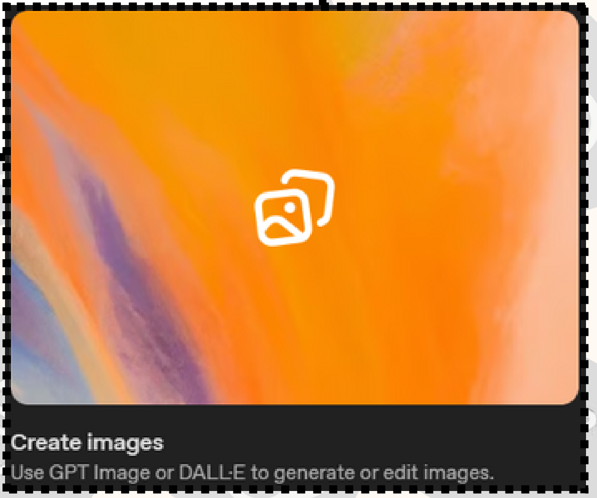
The Dynamic Image Generator is an AI-powered creative tool that turns text into stunning images instantly. It integrates FlutterFlow (low-code UI) OpenAI Image API Image Generation API (using the high-quality, cost-efficient gpt-image-1 model). Users simply type a prompt and get custom visuals with full control over size, style, and format, showcasing seamless low-code generative AI integration.

2. Inside the OpenAI documentation

Docs → Images & Vision → Create Images

gpt-image-1 model

- Supported image-generation models
- Output formats
- Model differences in cost, quality, and capabilities
- Safety limitations
- Usage examples
- cURL code for API calls



Create images

Use GPT Image or DALL·E to generate or edit images.

Specifically for this project

This model outputs high-quality images at a reasonable cost and supports customization such as size, style, and format

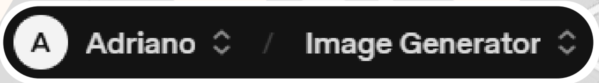
The default of image inputs utilize the gpt-image-1 model

3. Creating the Project and API Keys

In the OpenAI dashboard

Create the project named

Image Generator


A Adriano / Image Generator

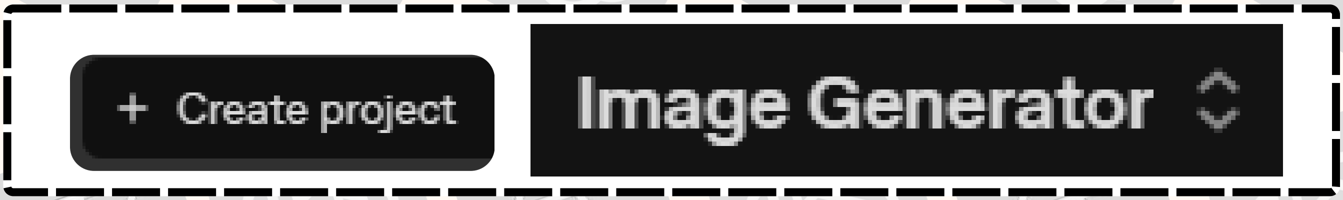
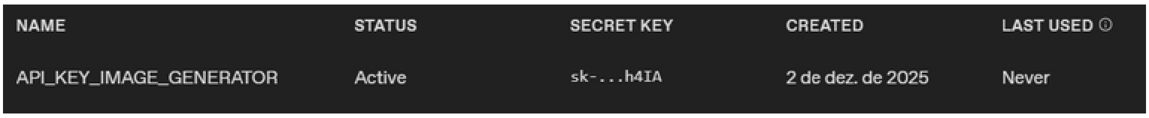

+ Create project

Image Generator

Generate an API Key

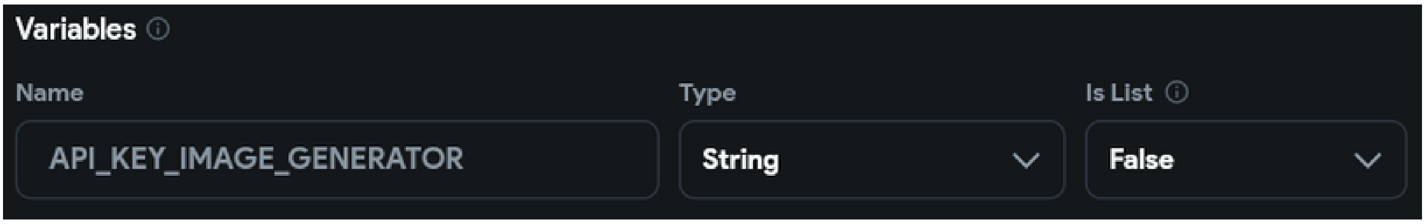
API keys

Don't Share your API Key



NAME	STATUS	SECRET KEY	CREATED	LAST USED ⓘ
API_KEY_IMAGE_GENERATOR	Active	sk-...h4IA	2 de dez. de 2025	Never

Store the key securely (Flutterflow → variables)


Variables ⓘ

Name	Type	Is List ⓘ
API_KEY_IMAGE_GENERATOR	String	False

4. Building the UI in FlutterFlow

Container 1

Title: **Image Generator Project**

Describe Image

txt_Prompt

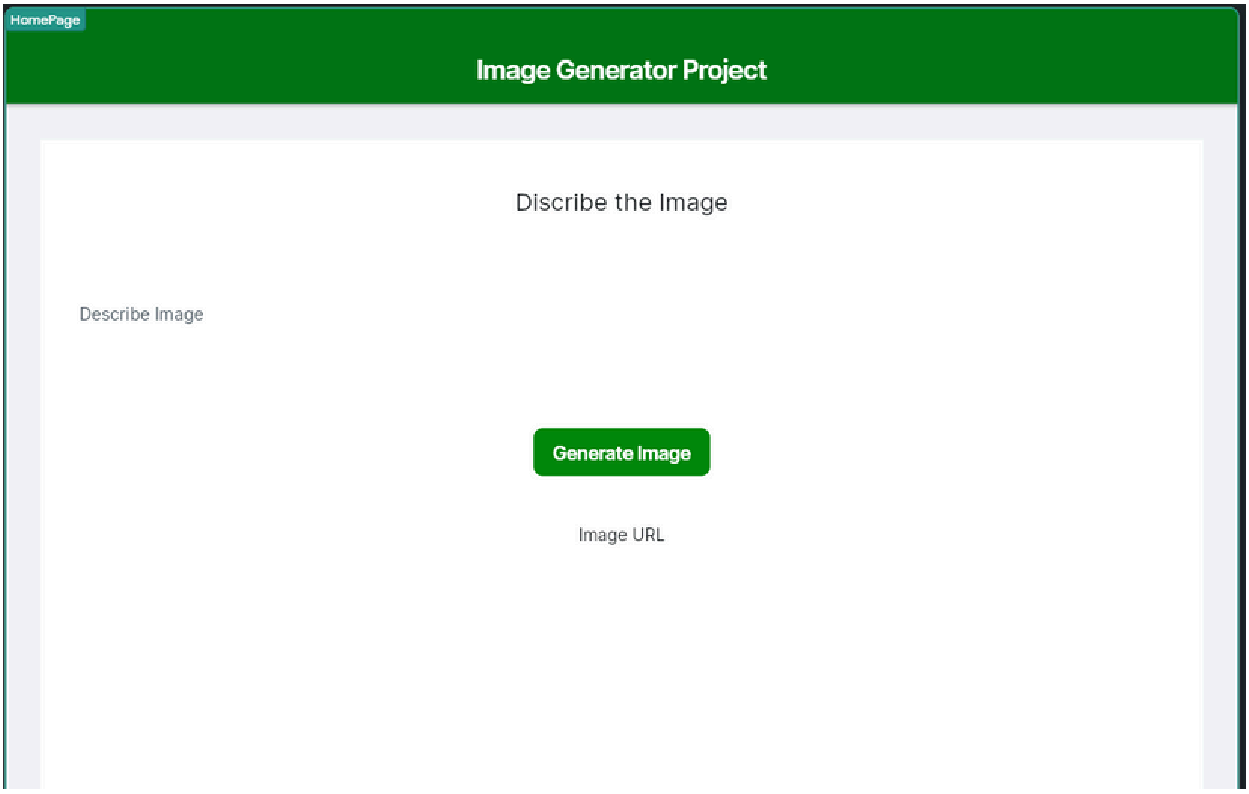
Describe Image

Button

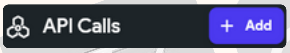
Image_URL

Answer

Preview



5. Creating the API Call In FlutterFlow


API Calls
+ Add

New API Call

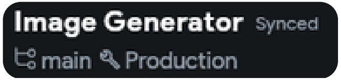


Image Generator Synced
main Production

Name: "Image Generator"

Set as POST

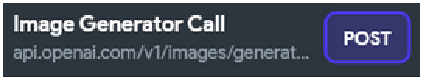
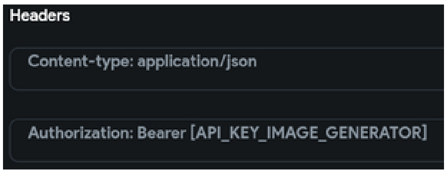


Image Generator Call
api.openai.com/v1/images/generat...
POST

Add headers:

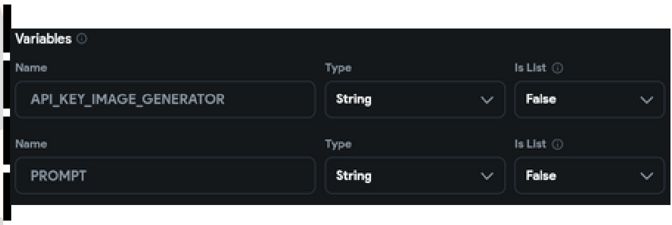
- Content-Type: application/json
- Authorization: Bearer [API_KEY_IMAGE_GENERATOR]


Headers

Content-type: application/json

Authorization: Bearer [API_KEY_IMAGE_GENERATOR]

Create variables


Variables

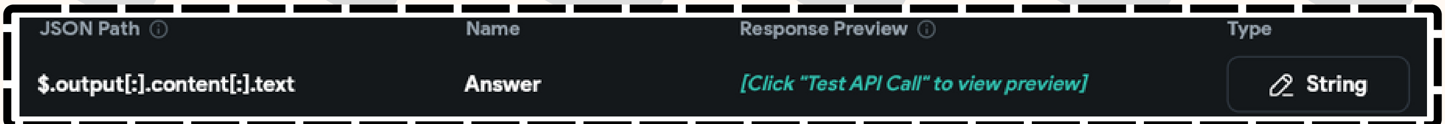
Name	Type	Is List
API_KEY_IMAGE_GENERATOR	String	False
PROMPT	String	False

- API_KEY_IMAGE_GENERATOR → string
- PROMPT → string (input from user)

Define Response Varable:

$.output[:]content[:].text

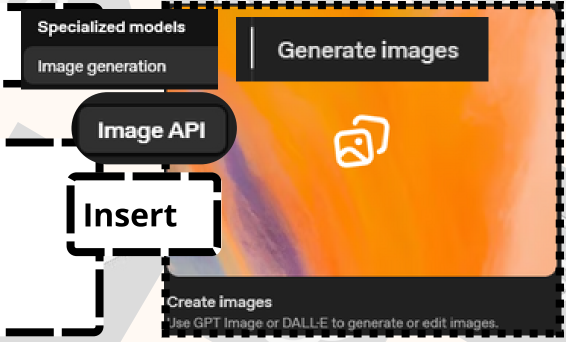
Answer = JSON Path



JSON Path	Name	Response Preview	Type
$.output[:].content[:].text	Answer	[Click "Test API Call" to view preview]	String

In JSON Body, add:

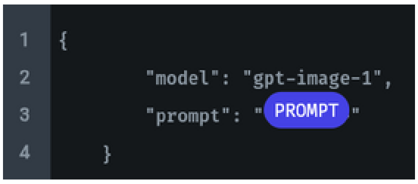
the OpenAI API **doc**



Insert

Create images
Use GPT Image or DALL·E to generate or edit images.

```
{
     "model": "gpt-image-1",
     "prompt": "[PROMPT"
  }
```



```
{
        "model": "gpt-image-1",
        "prompt": "PROMPT"
  }
```

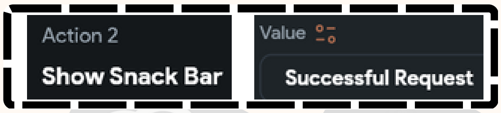
6. Applying the Image Generator API Inside the App

In the "Send Question" button:

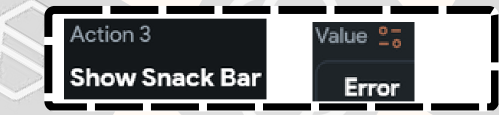
Add Action → **API Call**

Pass: API_KEY_IMAGE_GENERATOR PROMPT

On Success → Show Snackbar **“Request Successful”**

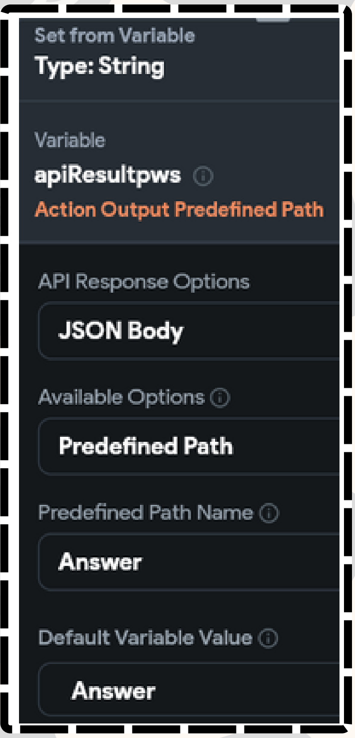

Action 2
Show Snack Bar
Value
Successful Request

On Error → Show Snackbar **“Error Detected”**


Action 3
Show Snack Bar
Value
Error

On the answer container:

- Set the text to show the API variable **Answer**


Set from Variable
Type: String

Variable
apiResultpws
Action Output Predefined Path

API Response Options
JSON Body

Available Options
Predefined Path

Predefined Path Name
Answer

Default Variable Value
Answer

Testing

Ask a Question

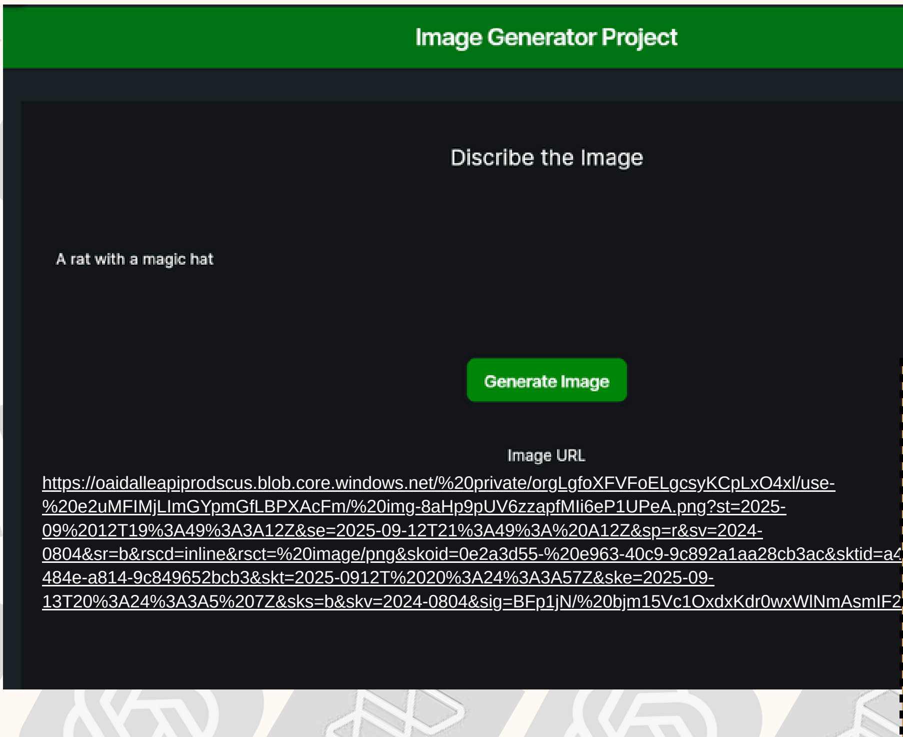


Image Generator Project

Discribe the Image

A rat with a magic hat

Generate Image

Image URL

https://oaidalleapiprodscus.blob.core.windows.net/%20private/orgLgfoXFVFoELgcsyKCpLxO4xl/use-%20e2uMFIMjLImGYpmGfLBPXAcFm/%20img-8aHp9pUV6zzapfMIi6eP1UPeA.png?st=2025-09%2012T19%3A49%3A3A12Z&se=2025-09-12T21%3A49%3A%20A12Z&sp=r&sv=2024-0804&sr=b&rscd=inline&rsct=%20image/png&skoid=0e2a3d55-%20e963-40c9-9c892a1aa28cb3ac&sktid=a4
484e-a814-9c849652bcb3&skt=2025-0912T%2020%3A24%3A3A57Z&ske=2025-09-13T20%3A24%3A3A5%207Z&sks=b&skv=2024-0804&sig=BFp1jN/%20bjm15Vc1OxdxKdr0wxWlNmAsmIF2

7. Conclusion

The Image Generator project demonstrates how to

- Integrate Flutterflow with the OpenAI Image Generation API
- Allow users to generate images by simply describing what they want
- Bind dynamic variables to API requests
- Handle responses and display generated images
- Build fast, smart, low-code creative applications

The Image Generator Project is a practical, fully functional example of how low-code development (FlutterFlow) + generative AI (OpenAI Image Generation API) instantly unlocks the ability to create dynamic, high-quality visual content without writing a single line of backend or specialized ML code, making sophisticated art generation apps accessible to anyone in minutes.

Next Project

Voice Transcription, project demonstrates in practice how to:

- Use the OpenAI Speech API to convert audio into text
- Upload audio files directly from Flutterflow
- Handle API responses and display transcriptions in a clean interface
- Understand the broader audio ecosystem including real-time, speech, and transcription capabilities

Audio Transcription

Quick Short Project

FlutterFlow

+

OpenAI API