# `GPT4VisionAPI` Documentation

Table of Contents

## Introduction<a name="introduction"></a>

Welcome to the documentation for the `GPT4VisionAPI` module! This module is a powerful wrapper for the OpenAI GPT-4 Vision model. It allows you to interact with the model to generate descriptions or answers related to images. This documentation will provide you with comprehensive information on how to use this module effectively.

## Installation<a name="installation"></a>

Before you start using the `GPT4VisionAPI` module, make sure you have the required dependencies installed. You can install them using the following commands:

```bash
pip3 install --upgrade swarms
```

## Module Overview<a name="module-overview"></a>

The `GPT4VisionAPI` module serves as a bridge between your application and the OpenAI GPT-4 Vision model. It allows you to send requests to the model and retrieve responses related to images. Here are some key features and functionality provided by this module:

- Encoding images to base64 format.
- Running the GPT-4 Vision model with specified tasks and images.
- Customization options such as setting the OpenAI API key and maximum token limit.

## Class: GPT4VisionAPI<a name="class-gpt4visionapi"></a>

The `GPT4VisionAPI` class is the core component of this module. It encapsulates the functionality required to interact with the GPT-4 Vision model. Below, we'll dive into the class in detail.

### Initialization<a name="initialization"></a>

When initializing the `GPT4VisionAPI` class, you have the option to provide the OpenAI API key and set the maximum token limit. Here are the parameters and their descriptions:

Parameter	Type	Default Value	Description
openai_api_key	str	`OPENAI_API_KEY` environment variable (if available)	The OpenAI API key. If not provided, it defaults to the `OPENAI_API_KEY` environment variable.
max_tokens	int	300	The maximum number of tokens to generate in the model's response.

Here's how you can initialize the `GPT4VisionAPI` class:

```python
from swarm_models import GPT4VisionAPI

# Initialize with default API key and max_tokens
api = GPT4VisionAPI()

# Initialize with custom API key and max_tokens
custom_api_key = "your_custom_api_key"
api = GPT4VisionAPI(openai_api_key=custom_api_key, max_tokens=500)
```

### Methods<a name="methods"></a>

#### encode_image<a name="encode_image"></a>

This method allows you to encode an image from a URL to base64 format. It's a utility function used internally by the module.

```python
def encode_image(img: str) -> str:
    """
    Encode image to base64.

    Parameters:
    - img (str): URL of the image to encode.

    Returns:
    str: Base64 encoded image.
    """
```

#### run<a name="run"></a>

The `run` method is the primary way to interact with the GPT-4 Vision model. It sends a request to the model with a task and an image URL, and it returns the model's response.

```python
def run(task: str, img: str) -> str:
    """
    Run the GPT-4 Vision model.

    Parameters:
    - task (str): The task or question related to the image.
    - img (str): URL of the image to analyze.

    Returns:
    str: The model's response.
    """
```

__call__<a name="__call__"></a>

The `__call__` method is a convenient way to run the GPT-4 Vision model. It has the same functionality as the `run` method.

```python
def __call__(task: str, img: str) -> str:
    """
        Run the GPT-4 Vision model (callable).

        Parameters:
        - task (str): The task or question related to the image.
        - img
    (str): URL of the image to analyze.

        Returns:
        str: The model's response.
    """
```

## Examples<a name="examples"></a>

Let's explore some usage examples of the `GPT4VisionAPI` module to better understand how to use it effectively.

### Example 1: Basic Usage<a name="example-1-basic-usage"></a>

In this example, we'll use the module with the default API key and maximum tokens to analyze an image.

```python
from swarm_models import GPT4VisionAPI

# Initialize with default API key and max_tokens
api = GPT4VisionAPI()

# Define the task and image URL
task = "What is the color of the object?"
img = "https://i.imgur.com/2M2ZGwC.jpeg"
```

```
# Run the GPT-4 Vision model
response = api.run(task, img)

# Print the model's response
print(response)
```

### Example 2: Custom API Key<a name="example-2-custom-api-key"></a>

If you have a custom API key, you can initialize the module with it as shown in this example.

```python
from swarm_models import GPT4VisionAPI

# Initialize with custom API key and max_tokens
custom_api_key = "your_custom_api_key"
api = GPT4VisionAPI(openai_api_key=custom_api_key, max_tokens=500)

# Define the task and image URL
task = "What is the object in the image?"
img = "https://i.imgur.com/3T3ZHwD.jpeg"

# Run the GPT-4 Vision model
response = api.run(task, img)
```

```python
# Print the model's response

print(response)
```

### Example 3: Adjusting Maximum Tokens<a name="example-3-adjusting-maximum-tokens"></a>

You can also customize the maximum token limit when initializing the module. In this example, we set it to 1000 tokens.

```python
from swarm_models import GPT4VisionAPI


# Initialize with default API key and custom max_tokens

api = GPT4VisionAPI(max_tokens=1000)


# Define the task and image URL

task = "Describe the scene in the image."

img = "https://i.imgur.com/4P4ZRxU.jpeg"


# Run the GPT-4 Vision model

response = api.run(task, img)


# Print the model's response

print(response)
```

## Additional Information<a name="additional-information"></a>

- If you encounter any errors or issues with the module, make sure to check your API key and internet connectivity.
- It's recommended to handle exceptions when using the module to gracefully handle errors.
- You can further customize the module to fit your specific use case by modifying the code as needed.

## References<a name="references"></a>

- [OpenAI API Documentation](https://beta.openai.com/docs/)

This documentation provides a comprehensive guide on how to use the `GPT4VisionAPI` module effectively. It covers initialization, methods, usage examples, and additional information to ensure a smooth experience when working with the GPT-4 Vision model.