

Documentation	
GET STARTED	
<u>Quickstart</u>	<p><b>Quickstart</b> Get up and running with the API in a few minutes.</p> <p><b>Create an API Key</b> Please visit <a href="#">here</a> to create an API Key.</p> <p><b>Set up your API Key (recommended)</b> Instead of putting your API key directly in your code, set it up as a secret environment variable. This makes using the API easier (no more copy-pasting the key everywhere) and more secure (your key won't be accidentally saved in your code).</p> <p><b>Next Steps</b> Check out the <a href="#">Playground</a> to try out the API in your browser. Also visit <a href="#">AI-Cloud Github</a> for examples and sample applications.</p>
<u>Models</u>	<p><b>Supported Models</b></p> <p><u>Llama-3-8B</u></p> <ul style="list-style-type: none"><li>• Model ID: <b>Meta-Llama-3-8B-Instruct</b></li><li>• Developer: Meta</li><li>• Context Window: 8192 tokens</li></ul> <p><u>Mistral-7B</u></p> <ul style="list-style-type: none"><li>• Model ID: <b>Mistral-7B-Instruct</b></li><li>• Developer: Mistral</li><li>• Context Window: 8192 tokens</li></ul> <p><u>Krutrim</u></p> <ul style="list-style-type: none"><li>• Model ID: <b>Krutrim-spectre-v2</b></li><li>• Developer: Krutrim</li><li>• Context Window: 4096 tokens</li></ul> <p><i>These are Instruct-type models. They are directly accessible through the API endpoint using the model IDs mentioned above.</i></p>
FEATURES	
<u>Chat Completions</u>	<p><b>Chat Completion Models</b></p> <p><u>Request</u></p> <div>Unset</div>

```
curl https://cloud.olakrutrim.com/v1/chat/completions \
-H "Content-Type: application/json" \
-H "Authorization: Bearer <KRUTRIM_API_KEY>" \
-d '{
  "model": "Krutrim-spectre-v2",
  "messages": [
    {
      "role": "system",
      "content": "You are a helpful assistant."
    },
    {
      "role": "user",
      "content": "Hello!"
    }
  ],
  "frequency_penalty": 0,
  "logit_bias": {2435:-100, 640:-100},
  "logprobs": true,
  "top_logprobs": 2,
  "max_tokens": 256,
  "n": 1,
  "presence_penalty": 0,
  "response_format": { "type": "text" },
  "stop": null,
  "stream": false,
  "temperature": 0,
  "top_p": 1
}'
```

#### Required Parameters:

- **messages** (array): A list of messages in the conversation so far. Each message is an object with the following fields:
  - **role** (string): Can be "system", "user", or "assistant".
  - **content** (string): The text of the message.
  - **name** (string, optional): An optional name to disambiguate messages from different users with the same role.
- **model** (string): ID of the language model to use for generation. Check Model ID(s) in the **Model section**.

#### Optional Parameters:

- **frequency\_penalty** (number, defaults to 0): Controls the likelihood of repeating previous phrases. Values between -2.0 and 2.0, with positive values penalizing repetition.
- **max\_tokens** (integer, optional): Maximum number of tokens allowed in the generated response.
  - **Default:** 256
- **n** (integer, defaults to 1): How many different chat completion choices to generate (increases cost).
- **presence\_penalty** (number, defaults to 0): Similar to frequency penalty, but considers overall newness of topics. Values between -2.0 and 2.0.
- **seed** (integer, optional): Attempts to generate the same response for repeated requests with the same seed. Determinism is not guaranteed.
- **stop** (string/array, optional): Up to 4 sequences to stop generation upon encountering.
- **stream** (boolean, defaults to false): Enables sending response in chunks during generation.
- **temperature** (number, defaults to 1): Controls randomness of the response (0-2). Higher is more random.
- **top\_p** (number, defaults to 1): Sampling method focusing on high-probability tokens (0-1). Lower values consider fewer tokens.

#### Response

Unset

```
{
  "id": "chatcmpl-123",
  "object": "chat.completion",
  "created": 1702685778,
  "model": "Krutrim-spectre-v2",
  "choices": [
    {
      "index": 0,
      "message": {
        "role": "assistant",
        "content": "Hello! How can I assist you today?"
      },
      "logprobs": {
        "content": [
          {
            "token": "Hello",
            "logprob": -0.31725305,
            "bytes": [72, 101, 108, 108, 111],
            "top_logprobs": [
              {
                "token": "Hello",
                "logprob": -0.31725305,
                "bytes": [72, 101, 108, 108, 111]
              }
            ]
          }
        ]
      }
    }
  ]
}
```

```
        "token": "Hi",
        "logprob": -1.3190403,
        "bytes": [72, 105]
      }
    ]
  },
  "finish_reason": "stop"
},
"usage": {
  "prompt_tokens": 9,
  "completion_tokens": 9,
  "total_tokens": 18
},
"system_fingerprint": null
}
```

The response is a chat completion object, or a sequence of chunks if streaming is enabled. Here are the key properties:

- **id** (string): Unique identifier for the chat completion.
- **choices** (array): List of chat completion choices (can be multiple if **n** is greater than 1).
- **Logprobs**- (boolean, null, optional)
  - Default: false
- **created** (integer): Unix timestamp (seconds) of when the chat completion was generated.
- **model** (string): ID of the language model used for generation.
- **object** (string): Always "chat.completion".
- **usage** (object): Statistics related to the completion request (resource usage).

## Errors

### Understanding Error Messages

These codes show up in your response. If there's an error, you'll also get details about it. Here's a breakdown of the codes you might see:



#### Success Codes

- **200 OK:** Everything went smoothly!

#### Error Codes

- **400 Bad Request:** Your request was confusing. Double-check the format and try again.

	<ul style="list-style-type: none"> <li>• <b>404 Not Found:</b> We couldn't find what you requested. Maybe the address is wrong?</li> <li>• <b>422 Unprocessable Entity:</b> The information you gave us wasn't quite right. Make sure it's all filled in correctly.</li> <li>• <b>429 Too Many Requests:</b> You're asking too much too fast! Slow down and try again later.</li> <li>• <b>500 Internal Server Error:</b> Something went wrong on our end. Try again later, or contact us if it keeps happening.</li> <li>• <b>502 Bad Gateway:</b> We got a bad response from someone helping us. It might be temporary, so try again.</li> <li>• <b>503 Service Unavailable:</b> We're taking a break or a bit busy. Wait a bit and try again.</li> </ul>
INTEGRATIONS	
<b><u>OpenAI Compatibility</u></b>	<p><b>OpenAI Compatibility</b> You can use the official OpenAI python SDK to run inferences::</p> <pre> Python # Assume openai&gt;=1.0.0 from openai import OpenAI  # Create an OpenAI client with your KRUTIM API KEY and endpoint openai = OpenAI(     api_key="&lt;YOUR_KRUTRIM_API_KEY&gt;",     base_url="https://cloud.olakrutrim.com/v1", )  chat_completion = openai.chat.completions.create(     model="krutrim-spectre-v2",     messages=[         {"role": "system", "content": "You are a helpful assistant."},         {"role": "user", "content": "Hello"}], )  print(chat_completion.choices[0].message.content) </pre>
<b><u>Accounts</u></b>	

API keys	<b>API keys</b> API keys are required for accessing the APIs. You can manage your API keys <a href="#">here</a> .						
Rate Limits	<b>Rate Limits</b> Rate limits act as control measures to regulate how frequently a user can access our services within a specified period. <table><tr><th>RPM</th><th>RPD</th><th>TPD</th></tr><tr><td>60</td><td>600</td><td>500000</td></tr></table>	RPM	RPD	TPD	60	600	500000
RPM	RPD	TPD					
60	600	500000					
<b><u>LEGAL</u></b>							
Policies & Notices	<b>Policies &amp; Notices</b> <a href="#">Terms of Use</a>  <a href="#">Privacy Policy</a>  <a href="#">EULA</a> 