# Chat Completions

POST https://api.sarvam.ai/v1/chat/completions

Content-Type: application/json

Calls Sarvam LLM API to get the chat completion. Supported model(s): `sarvam-m`.

Reference: https://docs.sarvam.ai/api-reference-docs/chat/completions

## OpenAPI Specification

```yaml

openapi: 3.1.1

info:

title: Chat Completions

version: endpoint\_chat.completions

paths:

/v1/chat/completions:

post:

operationId: completions

summary: Chat Completions

description: >-

Calls Sarvam LLM API to get the chat completion. Supported model(s):

`sarvam-m`.

tags:

- - subpackage\_chat

parameters:

- name: api-subscription-key

in: header

required: true

schema:

type: string

responses:

'200':

description: Successful Response

content:

application/json:

schema:

$ref: '#/components/schemas/CreateChatCompletionResponse'

'400':

description: Bad Request

content: {}

'403':

description: Forbidden

content: {}

'422':

description: Unprocessable Entity

content: {}

'429':

description: Quota Exceeded

content: {}

'500':

description: Internal Server Error

content: {}

requestBody:

content:

application/json:

schema:

$ref: '#/components/schemas/CreateChatCompletionRequest'

components:

schemas:

ChatCompletionRequestSystemMessage:

type: object

properties:

content:

type: string

role:

type: string

enum:

- type: stringLiteral

value: system

required:

- content

- role

ChatCompletionRequestUserMessage:

type: object

properties:

content:

type: string

role:

type: string

enum:

- type: stringLiteral

value: user

required:

- content

- role

ChatCompletionRequestAssistantMessage:

type: object

properties:

content:

type: string

role:

type: string

enum:

- type: stringLiteral

value: assistant

required:

- content

- role

ChatCompletionRequestMessage:

oneOf:

- $ref: '#/components/schemas/ChatCompletionRequestSystemMessage'

- $ref: '#/components/schemas/ChatCompletionRequestUserMessage'

- $ref: '#/components/schemas/ChatCompletionRequestAssistantMessage'

SarvamModelIds:

type: string

enum:

- value: sarvam-m

ReasoningEffort:

type: string

enum:

- value: low

- value: medium

- value: high

StopConfiguration:

oneOf:

- type: string

- type: array

items:

type: string

CreateChatCompletionRequest:

type: object

properties:

messages:

type: array

items:

$ref: '#/components/schemas/ChatCompletionRequestMessage'

model:

$ref: '#/components/schemas/SarvamModelIds'

temperature:

type:

- number

- 'null'

format: double

top\_p:

type:

- number

- 'null'

format: double

reasoning\_effort:

oneOf:

- $ref: '#/components/schemas/ReasoningEffort'

- type: 'null'

max\_tokens:

type:

- integer

- 'null'

stream:

type:

- boolean

- 'null'

stop:

oneOf:

- $ref: '#/components/schemas/StopConfiguration'

- type: 'null'

'n':

type:

- integer

- 'null'

seed:

type:

- integer

- 'null'

frequency\_penalty:

type:

- number

- 'null'

format: double

presence\_penalty:

type:

- number

- 'null'

format: double

wiki\_grounding:

type:

- boolean

- 'null'

required:

- messages

- model

FinishReason:

type: string

enum:

- value: stop

- value: length

- value: tool\_calls

- value: content\_filter

- value: function\_call

Role:

type: string

enum:

- value: assistant

ChatCompletionResponseMessage:

type: object

properties:

content:

type: string

role:

$ref: '#/components/schemas/Role'

required:

- content

- role

Choice:

type: object

properties:

finish\_reason:

$ref: '#/components/schemas/FinishReason'

index:

type: integer

message:

$ref: '#/components/schemas/ChatCompletionResponseMessage'

required:

- finish\_reason

- index

- message

CompletionUsage:

type: object

properties:

completion\_tokens:

type: integer

prompt\_tokens:

type: integer

total\_tokens:

type: integer

required:

- completion\_tokens

- prompt\_tokens

- total\_tokens

CreateChatCompletionResponse:

type: object

properties:

id:

type: string

choices:

type: array

items:

$ref: '#/components/schemas/Choice'

created:

type: integer

model:

type: string

object:

type: string

enum:

- type: stringLiteral

value: chat.completion

usage:

oneOf:

- $ref: '#/components/schemas/CompletionUsage'

- type: 'null'

required:

- id

- choices

- created

- model

- object

```

## SDK Code Examples

```python

from sarvamai import SarvamAI

client = SarvamAI(

api\_subscription\_key="YOUR\_API\_SUBSCRIPTION\_KEY",

)

client.chat.completions(

messages=[{"content": "content", "role": "assistant"}],

)

```

```typescript

import { SarvamAIClient } from "sarvamai";

const client = new SarvamAIClient({ apiSubscriptionKey: "YOUR\_API\_SUBSCRIPTION\_KEY" });

await client.chat.completions({

messages: [{

role: "assistant",

content: "content"

}]

});

```

```go

package main

import (

"fmt"

"strings"

"net/http"

"io"

)

func main() {

url := "https://api.sarvam.ai/v1/chat/completions"

payload := strings.NewReader("{\n \"messages\": [\n {\n \"content\": \"foo\",\n \"role\": \"foo\"\n }\n ],\n \"model\": \"sarvam-m\"\n}")

req, \_ := http.NewRequest("POST", url, payload)

req.Header.Add("api-subscription-key", "<apiKey>")

req.Header.Add("Content-Type", "application/json")

res, \_ := http.DefaultClient.Do(req)

defer res.Body.Close()

body, \_ := io.ReadAll(res.Body)

fmt.Println(res)

fmt.Println(string(body))

}

```

```ruby

require 'uri'

require 'net/http'

url = URI("https://api.sarvam.ai/v1/chat/completions")

http = Net::HTTP.new(url.host, url.port)

http.use\_ssl = true

request = Net::HTTP::Post.new(url)

request["api-subscription-key"] = '<apiKey>'

request["Content-Type"] = 'application/json'

request.body = "{\n \"messages\": [\n {\n \"content\": \"foo\",\n \"role\": \"foo\"\n }\n ],\n \"model\": \"sarvam-m\"\n}"

response = http.request(request)

puts response.read\_body

```

```java

HttpResponse<String> response = Unirest.post("https://api.sarvam.ai/v1/chat/completions")

.header("api-subscription-key", "<apiKey>")

.header("Content-Type", "application/json")

.body("{\n \"messages\": [\n {\n \"content\": \"foo\",\n \"role\": \"foo\"\n }\n ],\n \"model\": \"sarvam-m\"\n}")

.asString();

```

```php

<?php

$client = new \GuzzleHttp\Client();

$response = $client->request('POST', 'https://api.sarvam.ai/v1/chat/completions', [

'body' => '{

"messages": [

{

"content": "foo",

"role": "foo"

}

],

"model": "sarvam-m"

}',

'headers' => [

'Content-Type' => 'application/json',

'api-subscription-key' => '<apiKey>',

],

]);

echo $response->getBody();

```

```csharp

var client = new RestClient("https://api.sarvam.ai/v1/chat/completions");

var request = new RestRequest(Method.POST);

request.AddHeader("api-subscription-key", "<apiKey>");

request.AddHeader("Content-Type", "application/json");

request.AddParameter("application/json", "{\n \"messages\": [\n {\n \"content\": \"foo\",\n \"role\": \"foo\"\n }\n ],\n \"model\": \"sarvam-m\"\n}", ParameterType.RequestBody);

IRestResponse response = client.Execute(request);

```

```swift

import Foundation

let headers = [

"api-subscription-key": "<apiKey>",

"Content-Type": "application/json"

]

let parameters = [

"messages": [

[

"content": "foo",

"role": "foo"

]

],

"model": "sarvam-m"

] as [String : Any]

let postData = JSONSerialization.data(withJSONObject: parameters, options: [])

let request = NSMutableURLRequest(url: NSURL(string: "https://api.sarvam.ai/v1/chat/completions")! as URL,

cachePolicy: .useProtocolCachePolicy,

timeoutInterval: 10.0)

request.httpMethod = "POST"

request.allHTTPHeaderFields = headers

request.httpBody = postData as Data

let session = URLSession.shared

let dataTask = session.dataTask(with: request as URLRequest, completionHandler: { (data, response, error) -> Void in

if (error != nil) {

print(error as Any)

} else {

let httpResponse = response as? HTTPURLResponse

print(httpResponse)

}

})

dataTask.resume()

```