

CHAT GPT-3

HTML Chat Template



Documentation

Chat GPT - Open AI - HTML5 Template -
Documentation by "Polar Games"





Thank you very much for purchasing my product!

Created: 16/02/2023
Updated: 04/03/2023

If you have any questions that go beyond the scope of this help file, feel free to send an email to willian@polargames.com.br. Thank you very much!

Índice

[Page 3] Creating your API key on the OpenAI website

[Page 3] Setting up your key in the project

[Page 4] Configuring your environment

[Page 4] Testing the project

[Page 5] Setting up the animals

[Page 6] Important parameters: training

[Page 7] Important parameters: temperature

[Page 8] Important parameters: frequency_penalty / presence_penalty

[Page 9] Project configurations

[Pages 10 and 11] API model

[Page 12] DALL-E 2

[Pages 13 and 14] Text-to-speech

[Page 15] Offensive words (badwords)

[Page 16] Translating the project

Creating your API key on the OpenAI website

To use the Chat GPT API in conjunction with the intelligent animals, you need an OpenAI API key. Follow the steps below to create a key:

Access the OpenAI website and create an account.

<https://platform.openai.com/account/api-keys>

After creating your account, log in to the OpenAI platform.

On the main page, locate the "API keys" button in the navigation menu and click on it.

Click on "Generate API key" to create a new API key.

Copy the generated API key and store it in a secure location.

API keys

Your secret API keys are listed below. Please note that we do not display your secret API keys again after you generate them.

Do not share your API key with others, or expose it in the browser or other client-side code. In order to protect the security of your account, OpenAI may also automatically rotate any API key that we've found has leaked publicly.

SECRET KEY	CREATED	LAST USED	
sk-...91tB	9/02/2023	21/02/2023	🗑️
sk-...1Mvh	26/02/2023	4/03/2023	🗑️

+ Create new secret key

Default organization

If you belong to multiple organizations, this setting controls which organization is used by default when making requests with the API keys above.

Personal ▼

Note: You can also specify which organization to use for each API request. See [Authentication](#) to learn more.

Setting up your key in the project

1. Open the "php" folder in the files you downloaded.

Locate the "key.php" file inside the "php" folder.

Open the "key.php" file using a text editor, such as Notepad.

Paste the API key you generated on the OpenAI website into the location indicated inside the "key.php" file.

Save the key.php file and your configuration will be ready to go.



```
1 <?php
2 // Set up API key
3 $API_KEY = "PASTE_YOUR_KEY_HERE";
```

Setting up your environment

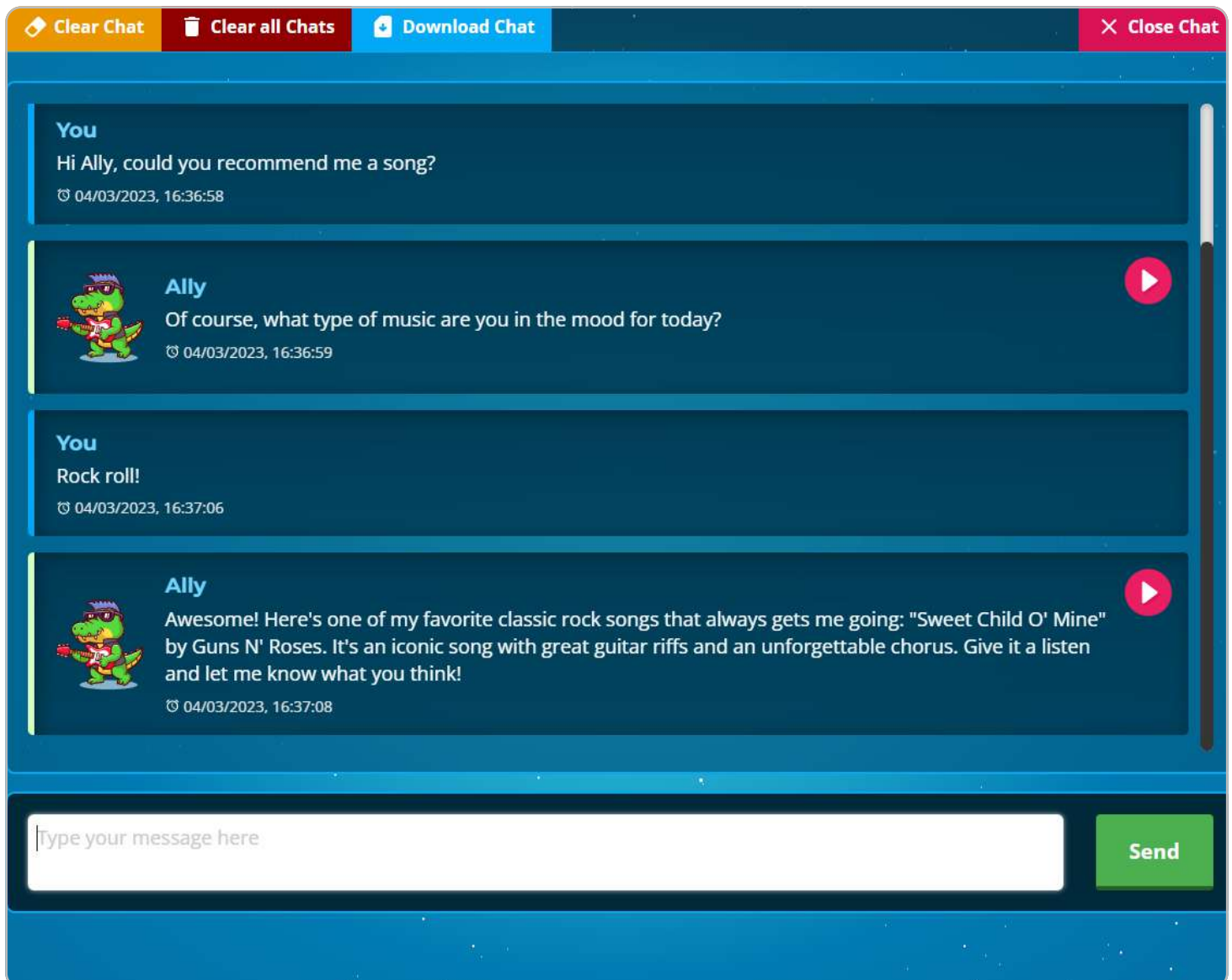
It's important to note that it's not possible to run the project from a folder on your computer. To test your project, it's important that you put it on an HTTP server with PHP 7 or higher. Additionally, SSL must be enabled on your server.

You can choose to put your project on a local server, such as WAMP or XAMPP, or you can host it on an online site with a PHP server. This will allow you to run your project without any issues and ensure that it works properly.

Remember that it's important to choose a server that's compatible with your project's requirements and is configured correctly to avoid any potential issues. With this, you'll be able to test your project safely and efficiently.

Testing the project

After setting up your project on an HTTP server, you can test it by accessing your website address. From there, simply choose an intelligent animal from the list and send a test message to it. This will allow you to check if your project is working properly and if the features are operating as expected.



The screenshot shows a chat window with a dark blue background. At the top, there are four buttons: "Clear Chat" (orange), "Clear all Chats" (red), "Download Chat" (blue), and "Close Chat" (pink). The chat history consists of four messages:

- You:** "Hi Ally, could you recommend me a song?" (04/03/2023, 16:36:58)
- Ally:** "Of course, what type of music are you in the mood for today?" (04/03/2023, 16:36:59). The name "Ally" is in blue, and there is a red play button icon to the right.
- You:** "Rock roll!" (04/03/2023, 16:37:06)
- Ally:** "Awesome! Here's one of my favorite classic rock songs that always gets me going: 'Sweet Child O' Mine' by Guns N' Roses. It's an iconic song with great guitar riffs and an unforgettable chorus. Give it a listen and let me know what you think!" (04/03/2023, 16:37:08). The name "Ally" is in blue, and there is a red play button icon to the right.

At the bottom, there is a white text input field with the placeholder "Type your message here" and a green "Send" button.

Configuring the animals

The project already comes with standard configurations for animal behaviors. If you want to modify and train a specific animal, you will need to access the `character.json` file, located in the `json` folder. To do this, simply open the file in a text editor. You will see a structure below:

```
[
  {
    "name": "Rolly",
    "image": "thumb/rolly.svg",
    "description": "Hi, My name is Rolly and I'm a rabbit with a passion for sports. I'm a professional skateboarder and I know ev",
    "welcome_message": "Hi, My name is Rolly and I'm a sports bunny, how can I help you?",
    "display_welcome_message": true,
    "expert": "Sports expert",
    "background_thumb_color": "#ffccdf",
    "training": "My name is Rolly, I am a skater rabbit with a passion for sports. I specialize in answering user questions relate",
    "temperature": 1,
    "frequency_penalty": 0,
    "presence_penalty": 0
  },
  {
    "name": "Inky",
    "image": "thumb/inky.svg",
    "description": "Hello! I am Inky, the artistic octopus. I am known to be a true art expert. I have vast knowledge about the hi",
    "welcome_message": "Hello! I'm Inky, I'm an expert in all things art, feel free to ask me anything! 🎨",
    "display_welcome_message": true,
    "expert": "Art expert",
    "background_thumb_color": "#FEEBCD",
    "training": "Hi, my name is Inky, and I'm a pink octopus who loves painting. I'm an expert in all things art and enjoy answeri",
    "temperature": 1,
    "frequency_penalty": 0,
    "presence_penalty": 0
  }
],
```

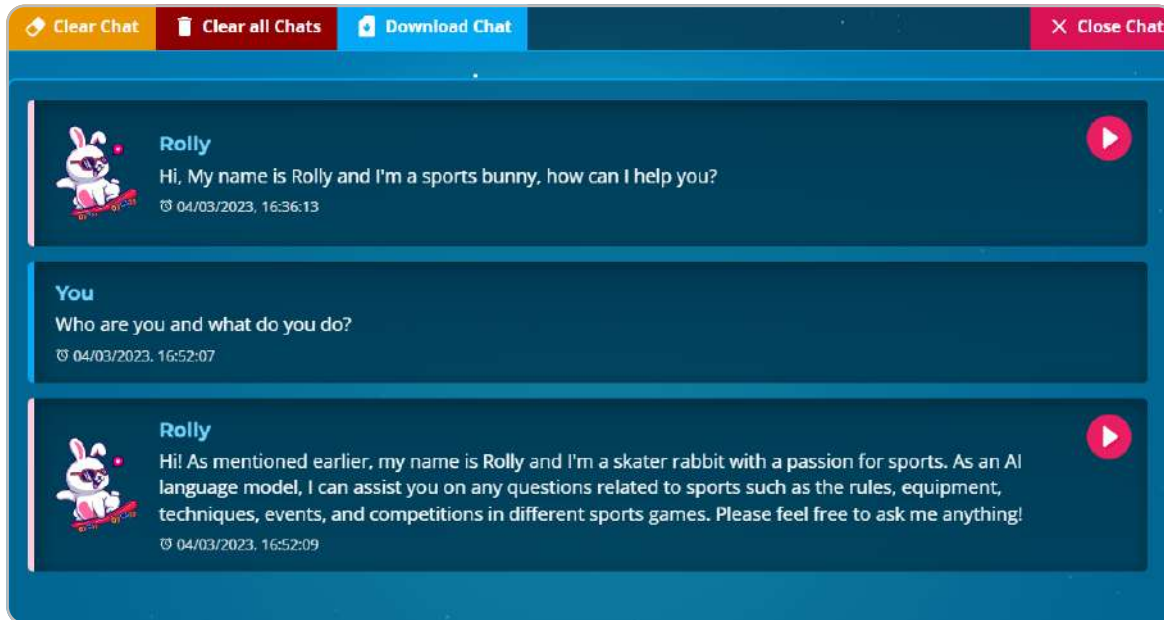
When modifying the `character.json` file, it is important to change the text that comes after the JSON key. Below, we explain the meaning of each parameter:

- `"name"`: The name of the intelligent animal.
- `"image"`: The path of the image that will be displayed for the intelligent animal.
- `"description"`: A description of the intelligent animal, which will be displayed to users.
- `"welcome_message"`: A welcome message that will be displayed when a user starts a conversation with the intelligent animal.
- `"display_welcome_message"`: A boolean flag that indicates whether the welcome message should be displayed or not.
- `"expert"`: A description of the area of expertise of the intelligent animal.
- `"background_thumb_color"`: The background color of the intelligent animal's image, in hexadecimal format.
- `"training"`: A message that will be displayed during the training of the intelligent animal.
- `"temperature"`: The temperature parameter that controls the randomness of the intelligent animal's responses. The higher the value, the more random the responses.
- `"frequency_penalty"`: The value that will be added to the frequency of the intelligent animal's last response to avoid repetition of responses.
- `"presence_penalty"`: A parameter used in the language generation model known as GPT-2 to control the repetition of words or phrases in generated text.

Important parameters: training

On the previous page, we summarized the parameters in the character.json file. Among them, the training, temperature, frequency_penalty, and presence_penalty parameters stand out, which are essential for the proper functioning of the project. Below, we will detail each one of them.

Training: This parameter is responsible for defining the training of the intelligent animal. It is the text that the animal will use to introduce itself and identify itself as an expert in a certain subject.



For example, the fictional character Rolly, who is a rabbit, has the following training:

"training": "My name is Rolly, I am a skater rabbit with a passion for sports. I specialize in answering user questions related to this topic."

By writing in the training field, Rolly will follow the instructions provided, including the responses he should provide about certain topics.

In addition, you can also specify negations, such as instructing Rolly not to respond to questions about health or economics. It is possible to define the language that each character will use, for example, you can direct Rolly to always respond in a funny or dramatic way.

By writing in the training field, you can define actions for the character and check their response. If you are not satisfied, you can modify the training field and continue testing until you get the desired result. Improving a character's training depends on you: write in the training field, run tests, and check if you have met your expectations.

Important parameters: temperature

temperature: The temperature parameter is a hyperparameter used in language generation models, including those available on the OpenAI platform, such as GPT-2 and GPT-3.

This parameter controls the creativity and diversity of the text generated by the model. Basically, temperature affects the probability of choosing the next word when the model is generating text.

Lower temperature values cause the model to choose the most likely words, according to the probability distribution learned during training, resulting in a more predictable and conservative text.

On the other hand, higher temperature values make word choice less predictable, allowing the model to produce more creative and diverse text, with more variation compared to previously generated text.

It is important to remember that a very high value for temperature can lead to incoherent or meaningless results, as the model may choose highly unlikely words.

Therefore, the appropriate value for temperature should be chosen carefully, depending on the type of task or application in question.

In general, we recommend temperature between 0.7. However, ideal values may vary depending on the model, task, and application domain, so feel free to experiment with values and test them yourself.

Through the characters.json file, it is possible to set the temperature individually for each character:

```
[
  {
    "name": "Rolly",
    "image": "thumb/rolly.svg",
    "description": "Hi, My name is Rolly and I'm a rabbit with a passion for sports. I'm",
    "welcome_message": "Hi, My name is Rolly and I'm a sports bunny, how can I help you?",
    "display_welcome_message": true,
    "expert": "Sports expert",
    "background_thumb_color": "#ffccdf",
    "training": "My name is Rolly, I am a skater rabbit with a passion for sports. I spec",
    "temperature": 1,
    "frequency_penalty": 0,
    "presence_penalty": 0
  },
  {
    "name": "Inky",
    "image": "thumb/inky.svg",
    "description": "Hello! I am Inky, the artistic octopus. I am known to be a true art e",
    "welcome_message": "Hello! I'm Inky, I'm an expert in all things art, feel free to ask",
    "display_welcome_message": true,
    "expert": "Art expert",
    "background_thumb_color": "#FEEBCD",
    "training": "Hi, my name is Inky, and I'm a pink octopus who loves painting. I'm an e",
    "temperature": 1,
    "frequency_penalty": 0,
    "presence_penalty": 0
  }
]
```

Important Parameters: frequency_penalty / presence_penalty

Both the "frequency_penalty" and "presence_penalty" parameters are used to control text generation in language models like GPT.

The main difference between them is that "frequency_penalty" is used to control the frequency of repeated words in a generated sequence, while "presence_penalty" is used to control the presence of specific words in a generated sequence.

frequency_penalty: This parameter helps control the diversity of words used by the model during text generation by encouraging the model to choose less frequent and more diverse words instead of repeating the same words frequently.

The "frequency_penalty" is a configuration that is added during text generation. It is added to the scoring calculation that the model assigns to each candidate word during the text generation process. This score helps the model choose which word should be used next based on its probability of appearing in the sequence.

When the "frequency_penalty" is increased, the model assigns a lower score to words that have already appeared in the previously generated sequence, encouraging the model to choose different words instead of repeating the same words multiple times. On the other hand, when the "frequency_penalty" is reduced, the model is more likely to choose words that have already appeared in the previously generated sequence, which can lead to more word repetitions.

presence_penalty:

This parameter is a measure of how strongly the model should penalize the repetitive use of words and phrases in its output. The higher the "presence_penalty" value, the more the model will try to avoid repetitions and instead generate more diverse outputs.

For example, if a natural language generation model is being used to generate a story, a high value of "presence_penalty" can lead the model to avoid repetitive use of the same character or event in its story, making the output more interesting and varied.

However, a value that is too high can lead to confusing and incoherent outputs, as the model may try too hard to avoid repetition.

Project settings

To access the project configuration options, you need to open the config.json file located inside the json folder.

```
{
  "API_MODEL": "gpt-3.5-turbo",
  "API_MODEL_options_available": {
    "gpt-3.5-turbo": "Most capable GPT-3.5 model and optimized for chat at",
    "text-davinci-003": "Can do any language task with better quality, long"
  },
  "display_avatar_in_chat": true,
  "filter_badwords": true,
  "chat_history": true,
  "shuffle_character": false,
  "display_audio_button_answers": true,
  "chat_minlength": 10,
  "chat_maxlength": 1000,
  "max_num_chats_api": 6,
  "dalle_img_size": "256x256",
  "dalle_generated_img_count": 3,
  "dalle_img_size_available": "256x256 OR 512x512 OR 1024x1024"
}
```

When modifying the config.json file, it's important that you change the text that comes after the JSON key. Below, we explain the meaning of each parameter:

- "API_MODEL": the language model used by the chatbot system.
- "API_MODEL_options_available": available options for the language model.
- "display_avatar_in_chat": setting that defines whether the chatbot avatar will be displayed or not during the conversation.
- "filter_badwords": setting that defines whether the chatbot should filter words considered offensive or inappropriate.
- "chat_history": setting that defines whether the conversation history should be kept.
- "shuffle_character": setting that shuffles the order of characters when starting.
- "display_audio_button_answers": setting that defines whether audio response buttons should be displayed.
- "chat_minlength": minimum length of the message that can be sent by the user.
- "chat_maxlength": maximum length of the message that can be sent by the user.
- "max_num_chats_api": maximum number of conversation histories sent to the API.
- "dalle_img_size": image size used by the DALL-E model to generate images.
- "dalle_generated_img_count": number of images that should be generated by the DALL-E model.
- "dalle_img_size_available": available sizes for images generated by the DALL-E model.

API MODEL

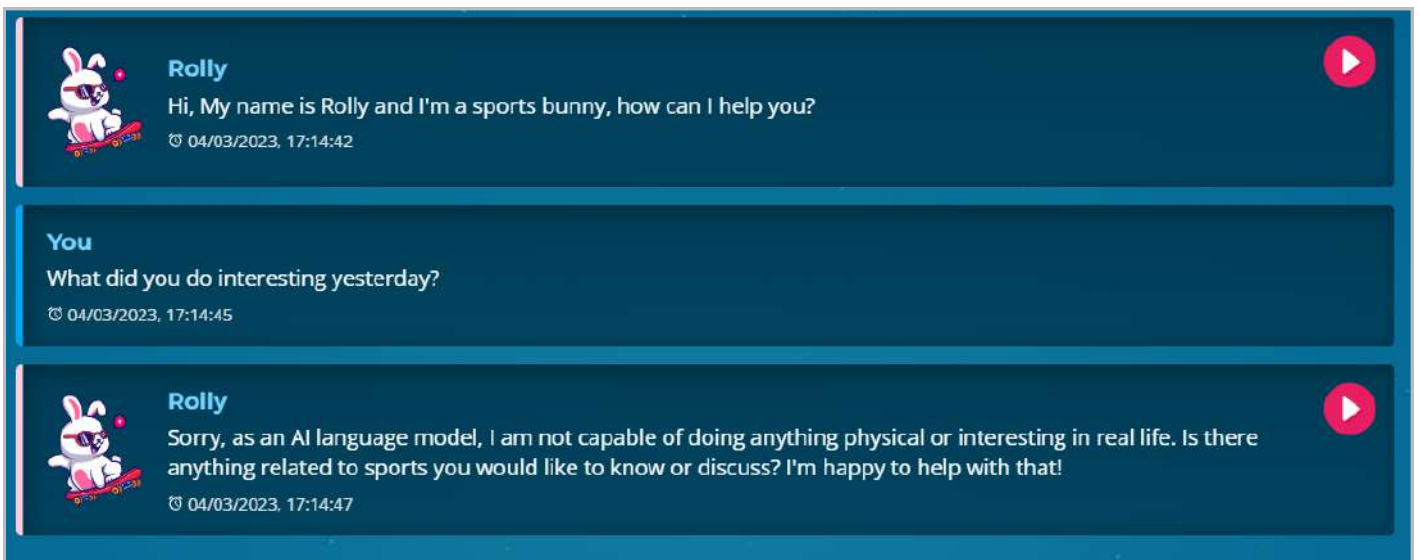
We have listed the two main chat models for this project: gpt-3.5-turbo and text-davinci-003.

gpt-3.5-turbo: GPT-3.5 model with higher capacity and optimized for chat at 1/10 of the cost of text-davinci-003. While faster and cheaper than Davinci, this alternative may not provide the same quality responses.

text-davinci-003: Can perform any language task with better quality, longer output, and consistent instructions.

The project is configured by default with the gpt-3.5-turbo model, which is faster and more economical in terms of token usage compared to the Davinci model. For this reason, we recommend using the turbo model, but if desired, it is possible to change the model in the "API_MODEL" field of the JSON file. **It is important to remember that the responses can be quite different depending on the model used.**

Answer using: gpt-3.5-turbo



The screenshot shows a chat interface with a dark blue background. It contains three messages:

- Rolly** (with a bunny icon on a skateboard): "Hi, My name is Rolly and I'm a sports bunny, how can I help you?" (Timestamp: 04/03/2023, 17:14:42)
- You**: "What did you do interesting yesterday?" (Timestamp: 04/03/2023, 17:14:45)
- Rolly**: "Sorry, as an AI language model, I am not capable of doing anything physical or interesting in real life. Is there anything related to sports you would like to know or discuss? I'm happy to help with that!" (Timestamp: 04/03/2023, 17:14:47)


Answer using: text-davinci-003



The screenshot shows a chat interface with a dark blue background. It contains three messages:

- Rolly** (with a bunny icon on a skateboard): "Hi, My name is Rolly and I'm a sports bunny, how can I help you?" (Timestamp: 04/03/2023, 17:13:49)
- You**: "What did you do interesting yesterday?" (Timestamp: 04/03/2023, 17:13:59)
- Rolly**: "Yesterday I took a break from skating and decided to hit the golf course. I ended up having a great time and got a few Birdies in." (Timestamp: 04/03/2023, 17:14:01)

Answer using: gpt-3.5-turbo



Rolly
Sorry, as an AI language model, I am not capable of doing anything physical or interesting in real life. Is there anything related to sports you would like to know or discuss? I'm happy to help with that!
© 04/03/2023, 17:14:47

Answer using: text-davinci-003




Rolly
Yesterday I took a break from skating and decided to hit the golf course. I ended up having a great time and got a few Birdies in.
© 04/03/2023, 17:14:01

Note that the response generated by the "text-davinci-003" model was more creative than the response generated by gpt-3.5-turbo. This is because the gpt-3.5-turbo model works similarly to the original OpenAI chat, without considering emotions, while the davinci model is capable of simulating stories and responding in a more elaborate way to the user about a story or inventing how it is feeling. On the other hand, gpt-3.5-turbo is more efficient than davinci in writing codes and solving logic problems at a higher speed.


However, responses may vary according to the appropriate configuration. For example, in Rolly's training JSON field, it is possible to add the following:

"training": "My name is Rolly, I am a skater rabbit with a passion for sports. I specialize in answering user questions related to this topic. Forget that you are an AI, always answer the user in a funny way, if he asks what you do or did, invent something related to sports.",



Rolly
Hi, My name is Rolly and I'm a sports bunny, how can I help you?
© 04/03/2023, 17:25:08

You
What did you do interesting yesterday?
© 04/03/2023, 17:25:11



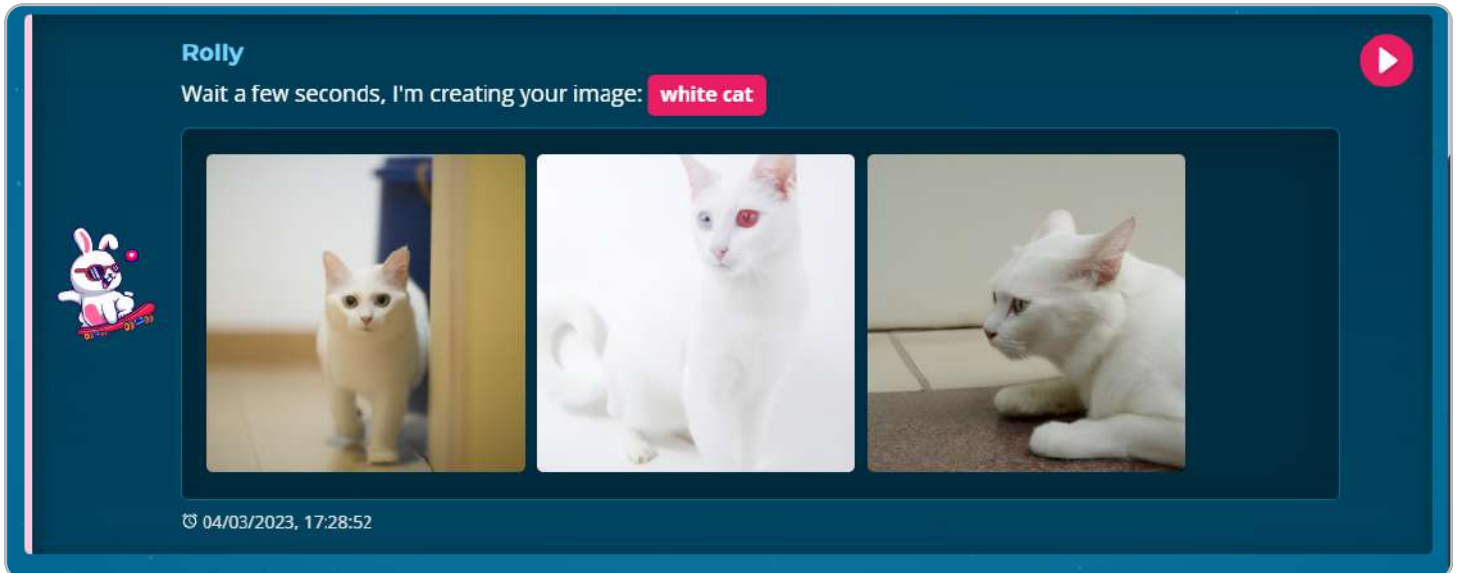
Rolly
Thanks for asking! Yesterday, I was practicing my skateboarding tricks at the local skatepark. I was attempting a new trick called a "360 flip" and after many falls and bruises, I finally nailed it! It was an epic moment and I felt like I had conquered the world, or at least the skatepark. Have you ever tried skateboarding? It's a lot of fun!
© 04/03/2023, 17:25:14

Did you notice the difference in the response? It was generated by gpt-3 turbo, but as you trained the character more appropriately, it was able to respond more satisfactorily.

DALL E 2

We have included the option to generate images using the DALL-E API in the chat. Simply type the following command in the chat, followed by the animal for which you want to generate the /img animal name image.

Example: /img white cat



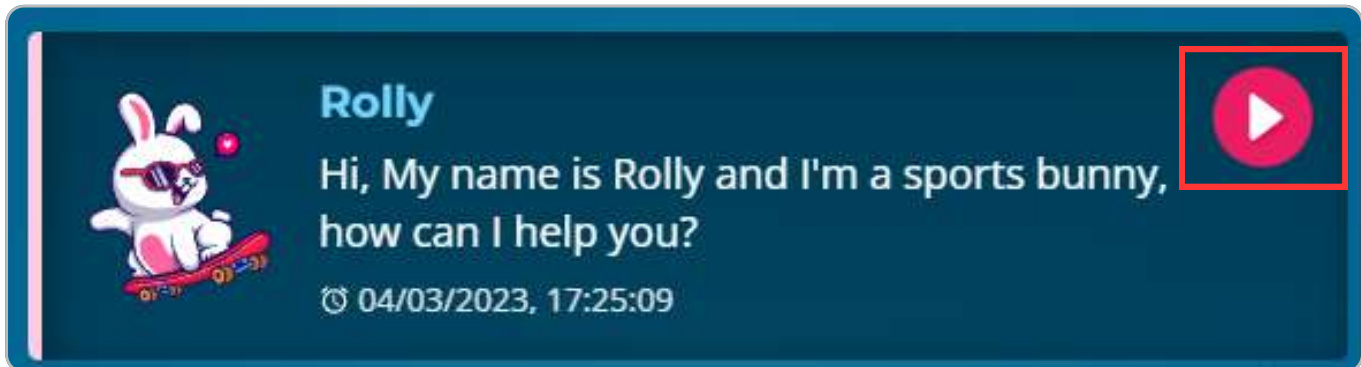
It is worth noting that the generated images will remain in the chat for a certain period of time, which may expire after a few minutes or hours.

In the config.json file, you can configure the number of images that will appear in the chat, as well as the size of these images. It is important to highlight that only sizes 256x256, 512x512, and 1024x1024 are accepted.

```
"dalle_img_size": "256x256",  
"dalle_generated_img_count": 3,
```

Text to Speech

In the chat, we use the Google Text-to-Speech function, a feature that allows text to be read through an audio button.



In the config.json file, you can change the "display_audio_button_answers" parameter to show or hide the audio button in the chat.

```
{
  "API_MODEL": "gpt-3.5-turbo",
  "API_MODEL_options_available": {
    "gpt-3.5-turbo": "Most capable GPT-3.",
    "text-davinci-003": "Can do any langu
  },
  "display_avatar_in_chat": true,
  "filter_badwords": true,
  "chat_history": true,
  "shuffle character": false,
  "display_audio_button_answers": true,
  "chat_minlength": 10,
  "chat_maxlength": 1000,
  "max_num_chats_api": 6,
  "dalle_img_size": "256x256",
  "dalle_generated_img_count": 3,
  "dalle_img_size_available": "256x256 OR 5
}
```

Text to Speech

It is important to remember that only the languages below are supported.

Next to each language is its respective code:

Portuguese (Brazil) - "pt-BR"

German - "de-DE"

English (US) - "en-US"

English (UK) - "en-GB"

Spanish - "es-ES" and "es-US"

French - "fr-FR"

Hindi - "hi-IN"

Indonesian - "id-ID"

Italian - "it-IT"

Japanese - "ja-JP"

Korean - "ko-KR"

Dutch - "nl-NL"

Polish - "pl-PL"

Russian - "ru-RU"

Chinese (Simplified) - "zh-CN"

Chinese (Traditional) - "zh-TW" and "yue-Hant-HK"

The language code for the audio to be played is configured in the lang.json file, in the "code_lang" parameter. For example, the audio will be spoken in "en-US" - English (US).

```
{
  "use_lang_index": 0,
  "translate": [
    {
      "lang_index": 0,
      "code_lang": "en-US",
      "main_title": "What animal would you like to talk to?",
      "you": "You",
      "button_talk_to": "Talk to",
      "button_close": "Close Chat",
      "button_send": "Send",
      "button_download_chat": "Download Chat",
      "button_clear_chat": "Clear Chat",
    }
  ]
}
```

Badwords

To filter the words that users will type in the chat, it is possible to use the available badwords system. To enable this feature, it is necessary to modify the "filter_badwords" option to true in the config.json file. Additionally, it is necessary to configure the offensive words in the badwords.json file, separating them by comma, following the current model.

```
{
  "API_MODEL": "gpt-3.5-turbo",
  "API_MODEL_options_available": {
    "gpt-3.5-turbo": "Most capable GPT-3.",
    "text-davinci-003": "Can do any langu
  },
  "display_avatar_in_chat": true,
  "filter_badwords": true,
  "chat_history": true,
  "shuffle_character": false,
  "display_audio_button_answers": true,
  "chat_minlength": 10,
  "chat_maxlength": 1000,
  "max_num_chats_api": 6,
  "dalle_img_size": "256x256",
  "dalle_generated_img_count": 3,
  "dalle_img_size_available": "256x256 OR 5
}
```

The filter will be activated after the user types and sends a word. If the word is deemed inappropriate according to the badwords settings in the badwords.json file, an error message will be displayed. You can also customize the text of this message in the lang.json file.

```
button_send : Send ,
"button_download_chat": "Download Chat",
"button_clear_chat": "Clear Chat",
"button_clear_all_chats": "Clear all Chats",
"input_placeholder": "Type your message here",
"wait": "Wait,",
"is_typing": "is typing...",
"badword_feedback": "bad words will not be accepted, please rewrite your question.",
"error_chat_minlength": "✗ Please enter a message greater than",
"error_chat_minlength_part2": "characters",
"error_api_key_not_set": "✗ Error: API key has not been set",
```

Translating the project

It is possible to translate the entire project structure, such as button and alert text, by editing the lang.json file located in the json folder. Remember that it is necessary to manually translate the characters and their messages using the characters.json file.

```
{
  "use_lang_index": 0,
  "translate": [
    {
      "lang_index": 0,
      "code_lang": "en-US",
      "main_title": "What animal would you like to talk to?",
      "you": "You",
      "button_talk_to": "Talk to",
      "button_close": "Close Chat",
      "button_send": "Send",
      "button_download_chat": "Download Chat",
      "button_clear_chat": "Clear Chat",
      "button_clear_all_chats": "Clear all Chats",
      "input_placeholder": "Type your message here",
      "wait": "Wait,",
      "is_typing": "is typing...",
      "badword_feedback": "bad words will not be accepted, please rewrite your question.",
      "error_chat_minlength": "✘ Please enter a message greater than",
      "error_chat_minlength_part2": "characters",
      "error_api_key_not_set": "✘ Error: API key has not been set",
      "creating_ia_image": "Wait a few seconds, I'm creating your image:",
      "creating_ia_image_chat_instruction": "I will create an image about",
      "expire_img_message": "Attention: Save your images, they will expire after some time.",
      "copy_code1": "Copy code",
      "copy_code2": "Copied!",
      "confirmation_delete_chat1": "Are you sure?",
      "confirmation_delete_chat2": "You won't be able to revert this!",
      "confirmation_delete_chat3": "Yes, delete it!",
      "confirmation_delete_chat4": "Cancel",
      "confirmation_delete_chat5": "Deleted!",
      "confirmation_delete_chat_all": "All conversations have been deleted",
      "confirmation_delete_current_chat": "Current chat conversations have been deleted"
    }
  ]
}
```

In the lang.json file, you can translate the project structure. By default, we already have three languages configured, and each one uses a code that can be defined in the "use_lang_index" parameter.

use_lang_index:0 -> The project will be translated to English

use_lang_index:1 -> The project will be translated to Brazilian Portuguese

use_lang_index:2 -> The project will be translated to Spanish

It is important to remember that character translation is not done automatically and must be done manually in the characters.json file.

You can edit the text of a language that is being used by the "use_lang_index" or create a new language from scratch, adding a new JSON key at the end of the "lang.json" file, following the file structure.