

API de Asistente



Alejandra Caggiano

Dentro del conjunto de APIs disponibles, GeneXus Enterprise AI ofrece una que permite la creación de nuevos asistentes, la modificación de sus definiciones y la recuperación de su información.

API de Asistente

Variables genéricas

Variable	Description
\$BASE_URL	The base URL for your GeneXus Enterprise AI installation.
\$SAIA_APITOKEN	An API token generated for each project.

Métodos disponibles

Method	Path	Description
GET	/assistant/{id}	Gets assistant data
POST	/assistant	Creates a new assistant
PUT	/assistant/{id}	Updates an assistant
DELETE	/assistant/{id}	Deletes an assistant
POST	/assistant/text/begin	Begins a text conversation with the GeneXus Enterprise AI Assistant
POST	/assistant/text	Sends a text prompt to the GeneXus Enterprise AI Assistant
POST	/assistant/chat	Sends a chat request to the GeneXus Enterprise AI Assistant
GET	/assistant/request/{id}/status	Retrieves the status of a request
POST	/assistant/request/{id}/cancel	Cancels a request

Para utilizar esta API debemos considerar las variables genéricas que ya conocemos: \$Base_URL y \$SAIA_APIToken

Y se requiere también de un API token de GeneXus Enterprise AI relacionado con el alcance de la organización.

Los métodos disponibles para esta API son los siguientes.

A modo de ejemplo, vamos a probar el método POST que permite crear un nuevo asistente.

API de Asistente: POST assistant

cURL Sample

```
curl -X POST "$BASE_URL/v1/assistant" \  
-H "Authorization: Bearer $$AIA_PROJECT_API_TOKEN" \  
-H "accept: application/json" \  
-d '{  
  "type": "chat",  
  "name": "TestAssistant",  
  "prompt": "translate the following text to Esperanto"  
}'
```

<https://api.qa.saia.ai/v1/assistant>

El cURL sample correspondiente, que recordemos se encuentran disponibles en la documentación técnica de GeneXus Enterprise AI, indica que la url debe tener la siguiente forma:

Por lo tanto, en nuestro ejemplo, la url es la siguiente: <https://api.qa.saia.ai/v1/assistant> donde esta parte corresponde al contenido de la variable BASE_UR: aplicada nuestro contexto.

API de Asistente: POST assistant

The screenshot displays the Genexus Enterprise AI interface. On the left, a sidebar shows navigation options like Dashboard, Assistants, and Requests. The main area shows a table of 'Project Api Tokens' for the 'TrainingProject'. The table has columns for Name, Description, and Status. Below the table, there's a 'MyFrenchAssistant' section with a prompt in English and its translation in French.

On the right, a Postman interface shows a POST request to 'https://api.qa.saila.ai/v1/assistant'. The request body is a JSON object:

```
1 {
2   "type": "chat",
3   "name": "MyFrenchAssistant",
4   "prompt": "The user proposes a small paragraph, and the expected response is its translation into French. For example, for the input 'My name is Alejandra and I live in Uruguay. We are in autumn.', the expected response is 'Je m'appelle Alejandra et je vis en Uruguay. C'est l'automne.'"}

```

The response body is also shown in JSON format:

```
1 {
2   "assistantDescription": "MyFrenchAssistant",
3   "assistantId": "910018b0-c704-4a18-92b0-34cb24de5ccf",
4   "assistantName": "MyFrenchAssistant",
5   "assistantPriority": 0,
6   "assistantStatus": 1,
7   "assistantType": "TextPromptAssistant",
8   "intents": [
9     {
10      "assistantIntentDefaultRevision": "1",
11      "assistantIntentDescription": "Default",
12      "assistantIntentId": "01e2270a-5d70-427e-8492-9ceca1283a06",
13      "assistantIntentName": "Default",

```

Bien. Ingresamos entonces a Postman y declaramos el POST.

Vemos que el tipo de autorización es Bearer y se requiere de un Project api token. Así que, desde la plataforma, vamos a la opción Api tokens y copiamos el default.

Volvemos entonces a Postman, y definimos la autorización requerida. En type ponemos Bearer token, y pegamos el token.

Para definir el cuerpo necesitamos indicar el tipo de asistente, su nombre y prompt.

Nuestra intención es crear un asistente para chat, que nos devuelva la traducción al francés de un pequeño párrafo ingresado por el usuario.

Su nombre será "MyFrenchAssistant" y el prompt dirá que se trata de un asistente personal. El usuario propone un pequeño párrafo y la respuesta esperada es su traducción al francés, Por ejemplo, para el input "Me llamo Alejandra y vivo en Uruguay. Estamos en otoño.", la respuesta esperada es "Je m'appelle Alejandra et je vis en Uruguay. C'est l'automne."

Así que en la solapa Body, elegimos Raw, Jason y definimos la estructura.

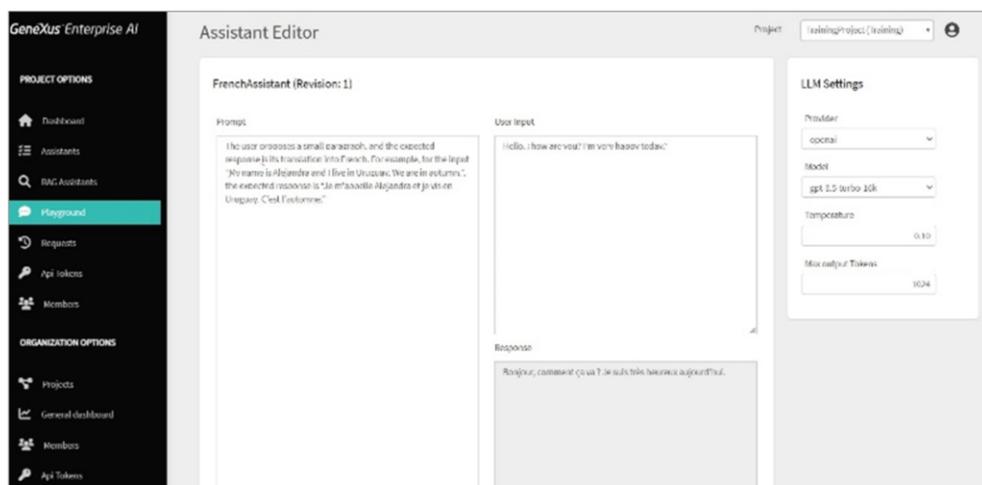
El tipo de asistente es Chat, el nombre es MyFrenchAssistant, y pegamos el

prmt que mencionamos anteriormente.

Presionamos Send, y vemos la respuesta con la creación del asistente.

Entre toda la información que se nos brinda, vemos que se creó la revisión 1, o sea, la primera versión de la definición del asistente, (recordemos que un asistente puede tener varias revisiones), y vamos a tomar especial cuenta en el Id, ya que es un parámetro necesario en la ejecución de otros métodos de la API, como ser GET, PUT y DELETE.

Testeando el Asistente: Edit prompt



Bien, lo que queremos ahora es probarlo. Y para eso tenemos varias opciones.

Una opción es ingresar directamente a la plataforma y verificar que el asistente MyFrenchAssistant está definido.

Podemos ver el prompt definido, ingresar un input y testearlo.

Por ejemplo, vamos a decir "Hola, ¿cómo estás? Estoy muy contenta hoy." Presionamos Test y vemos su traducción al francés.

Testeando el asistente: Playground



Muy bien. Vamos a testarlo también desde el Playground

Seleccionamos el asistente, creamos un nuevo chat, e indicamos esta vez que Uruguay es un país de América del Sur. Su capital es Montevideo.”

Y vemos su traducción.

Testeando el asistente: Postman API Platform

cURL Sample

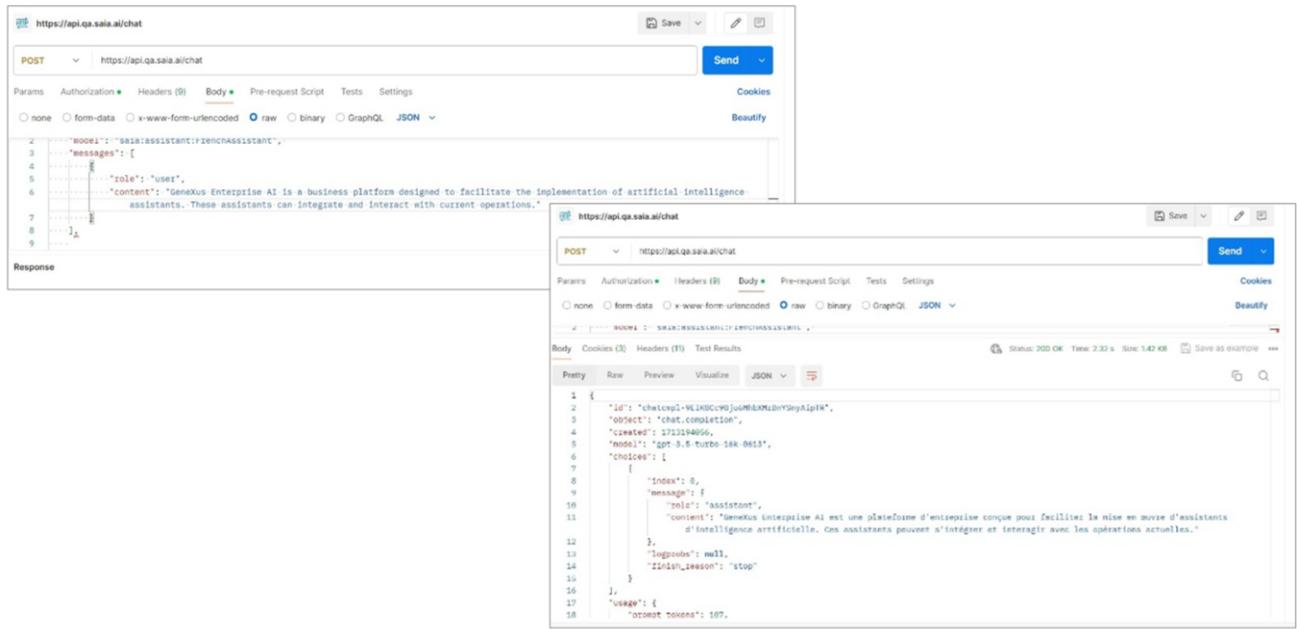
```
curl -X POST "$BASE_URL/chat" \  
-H "Authorization: Bearer $$SAIA_PROJECT_API_TOKEN" \  
-H "Content-Type: application/json" \  
--data '{  
  "model": "saia:assistant:translate-to-spanish",  
  "messages": [  
    {  
      "role": "user",  
      "content": "Hi, welcome to GeneXus Enterprise AI!!"  
    }  
  ],  
  "stream": true  
'
```

Chat API

<https://api.qa.saia.ai/chat>

Perfecto. Nos queda testearlo también vía API, y para eso vamos a utilizar la API para chat. El cURL sample es el que estamos viendo y que ya hemos utilizado anteriormente. Vamos entonces a definir un POST, y la url será la siguiente: <https://api.qa.saia.ai/chat>

Testeando el asistente: Postman API Platform



Volvemos a Postman y definimos el POST.

La autorización es Bearer y necesitamos un Project api token. Así que vamos a la plataforma, API tokens, copiamos el Default, y lo pegamos.

Vamos a Body, Raw, Json y definimos el cuerpo de la solicitud. Recordemos que "Model" corresponde al tipo de asistente, seguido de su nombre.

Así que indicamos el type "assistant" y el nombre "MyFrenchAssistant".

Recordemos también que Messages define un mensaje que se desea agregar, donde "content" corresponde al input del usuario.

Vamos a indicar como input que "GeneXus Enterprise AI es una plataforma de negocios diseñada para facilitar la implementación de asistentes de inteligencia artificial. Estos asistentes pueden integrarse e interactuar con las operaciones actuales."

De ser necesario podríamos indicar otros parámetros, como puede ser determinar la revisión del asistente que se desea utilizar. En este caso tenemos una sola y es la activa por defecto, así que no es necesario indicarla.

Presionamos Send, y vemos la respuesta.

API de Asistente: DELETE assistant

cURL Sample

```
curl -X DELETE "$BASE_URL/v1/assistant/{id}" \  
-H "Authorization: Bearer $SAIA_PROJECT_API_TOKEN" \  
-H "accept: application/json"
```

```
"assistantId": "be96910c-d008-4697-846d-38accdf220e8",
```

<https://api.qa.saia.ai/v1/assistant/be96910c-d008-4697-846d-38accdf220e8>

Ahora bien. Para finalizar, vamos a eliminar el asistente que hemos creado y testeado a través de todas las opciones.

Para eso vamos a utilizar el método Delete, que requiere del AssistantId como parámetro.

El Id de nuestro asistente que ahora queremos eliminar es el que estamos viendo:

Por lo tanto, la url necesaria es la siguiente:

<https://api.qa.saia.ai/v1/assistant/be96910c-d008-4697-846d-38accdf220e8>

También necesitamos un Project api token para la autorización.

API de Asistente: DELETE assistant

The image shows two overlapping screenshots. The top one is a Postman interface for a DELETE request to `https://api.qa.saka.ai/v1/assistant/be96910c-d008-4697-846d-38accdf220e8`. The Authorization tab is selected, showing a Bearer Token type and a token value: `default_OuK5BwzqSLNEHwUF-GV0DQCLIZYfYxY8BGAeNjgTCLSa8FgeZKcOrSS0rsvhehojogIRDpeSlcOQObWelaGcBRmSgIX0A8fKlogwW0NkRv3Uw00P1-J7UBWDC0X-DXrP2uRkJILUx0`. The bottom screenshot is the Genexus Assistant interface, showing a table of assistants. The table has columns for Assistant Name, Type, Active Revisions, and Last Update. Two assistants are listed: 'NubeInglésAsist' and 'NubeEspañolAsist', both of type 'FrontendChat'.

Assistant Name	Type	Active Revisions	Last Update
NubeInglésAsist	FrontendChat	1	02/17/24 07:03:49
NubeEspañolAsist	FrontendChat	1	02/15/24 16:12:45

Vamos a Postman y definimos la solicitud. Presionamos Send.

No hay errores, y si volvemos a la plataforma, podemos comprobar que el asistente fue eliminado.

GeneXus[™]
by **Globant**

training.genexus.com