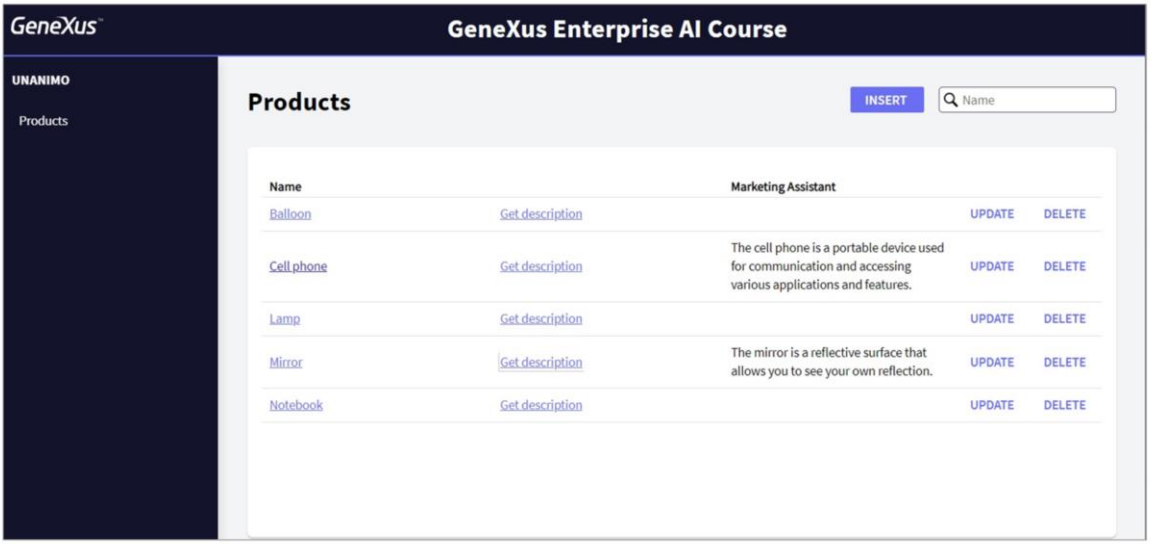
GeneXus by Globant

Aplicación GeneXus que interactúa con un Chat Assistant

Alejandra Caggiano

Llegados a este punto, ya sabemos cómo interactuar con GeneXus Enterprise AI, conocemos su Backoffice y su Frontend, sabemos cómo crear asistentes y cómo testearlos, ya sea editando el prompt, a través del Playground o vía api.

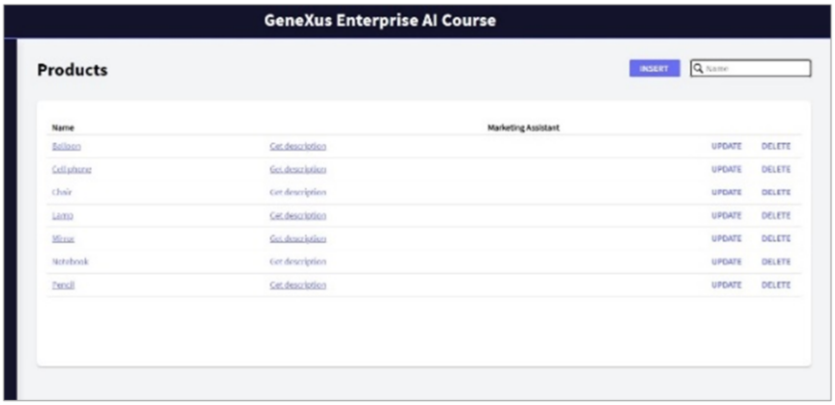
Aplicación GeneXus que interactúa con un Chat Assistant



Vamos a ver ahora un ejemplo de uso para interactuar con nuestros asistentes desde una base de conocimiento GeneXus. El objetivo es poder interactuar con nuestro MarketingAssistant para que devuelva la descripción formal del producto indicado por el usuario.

Aplicación GeneXus que interactúa con un Chat Assistant

Product	Product
ProductId	Numeric(4.0)
ProductName	Character(20)



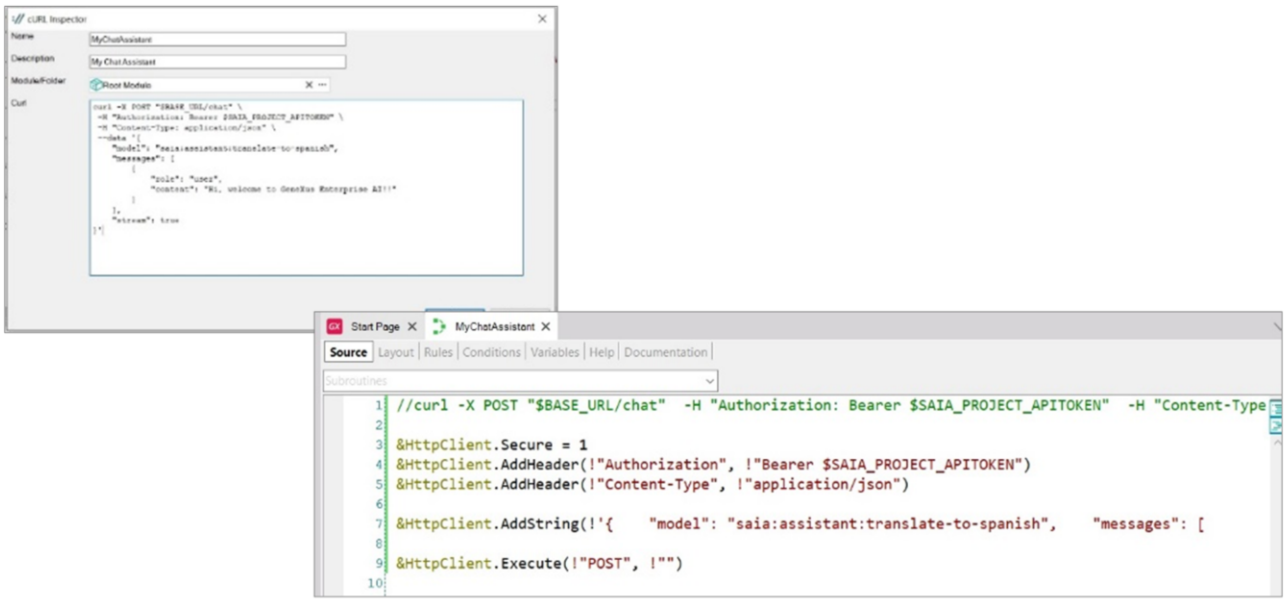
Bien. En nuestra KB tenemos la transacción Prducto, muy simple, con Id y Nombre a la que le hemos aplicado el pattern Work With for Web.

Nuestra idea es poder solicitar la descripción del producto desde la pantalla principal generada por la aplicación de este pattern.

Para eso en la instancia del pattern aplicado, hemos definido dos variables:

- La variable &Description que es sobre la cual el usuario dará click, y tiene el texto “Get description”.
- Y la variable &AssistantDescription que es la que recibirá la descripción devuelta por el asistente.

Aplicación GeneXus que interactúa con un Chat Assistant



Bien. El objetivo ahora es definir el procedimiento que establecerá la interacción con el asistente MarketingAssistant y devolverá entonces la descripción buscada.

Si bien podríamos crear un procedimiento en blanco y comenzar a definir los datos de conexión mediante una variable de tipo HttpClient, lo que haremos es darle a GeneXus el cURL sample que necesitamos para hacer el POST y GeneXus nos devolverá la estructura base para esta definición.

Vamos entonces al menú Tool / Appliction Integration, y elegimos cURL Inspector. Ponemos como nombre MyChatAssistant, y pegamos el sample, disponible, como ya sabemos, en la documentación técnica de GeneXus Enterprise AI.

https://wiki.genexus.com/enterprise-ai/wiki?8,Table+of+contents%3AEnterprise+AI,

Presionamos Ok, y vemos en la ventana KB Explorer que se creó el procedimiento de nombre MyChatAssistant, que ya nos ofrece la base de lo que necesitamos.

Vemos que se ha definido una variable &HttpClient basada en el tipo de dato del mismo nombre. Este tipo de dato permite crear un request, una solicitud, la envía a una URL y lee los resultados.

Aplicación GeneXus que interactúa con un Chat Assistant

```
//curl -X POST "$BASE_URL/chat"  -H "Authorization: Bearer $SAIA_PROJECT_APITOKEN"  -H "Content-Type: application,

&HttpClient.Secure = 1
&HttpClient.Host = "api.qa.saia.ai"

&HttpClient.AddHeader(!"Authorization", !"Bearer default_OuK5BwzqSLjNEHwUf-GV0QCLI2YHYxBBGAshAg1CiLSa9lFgeZKcQr5S
&HttpClient.AddHeader(!"Content-Type", !"application/json")

&HttpClient.AddString(!'{"model":"saia:assistant:MarketingAssistant","messages": [{ "role":"user", "content":"' +
```

¿Qué debemos hacer nosotros ahora?

Sustituir los parámetros con la información de nuestro contexto.

Lo primero que debemos especificar es si la llamada se realizará mediante un protocolo seguro o no.

Para eso debemos declarar el método Secure. El valor 0, que es el valor predeterminado, indica que se utilizará el protocolo HTTP, y el valor 1 indica que se utilizará el protocolo HTTPS. Indicamos entonces el valor 1.

Bien. Declaramos luego el Project api token, que previamente, como ya sabemos, lo copiamos desde la plataforma.

Vamos a concentrarnos ahora en el Body, el cuerpo de la solicitud.

Lo que haremos es similar a lo que hicimos desde Postman cuando testeamos al asistente vía api.

En primer lugar, indicamos el nombre del asistente que en este caso es MarketingAssistan.

Pensemos ahora lo siguiente. Nuestro asistente debe devolver la descripción del producto que el usuario indique, por lo tanto, el producto

debe ser recibido por parámetro en este procedimiento.

Así que declaramos la regla Parm para que el procedimiento reciba el nombre del producto y devuelva la descripción.

Volvemos al source, y en el grupo Messages vemos el “content” que, como ya sabemos, corresponde al input del usuario. Lo modificamos para quede parametrizado y tome en cada caso el nombre del producto que se recibe por parámetro.

Indicamos también la revisión del asistente que queremos utilizar.

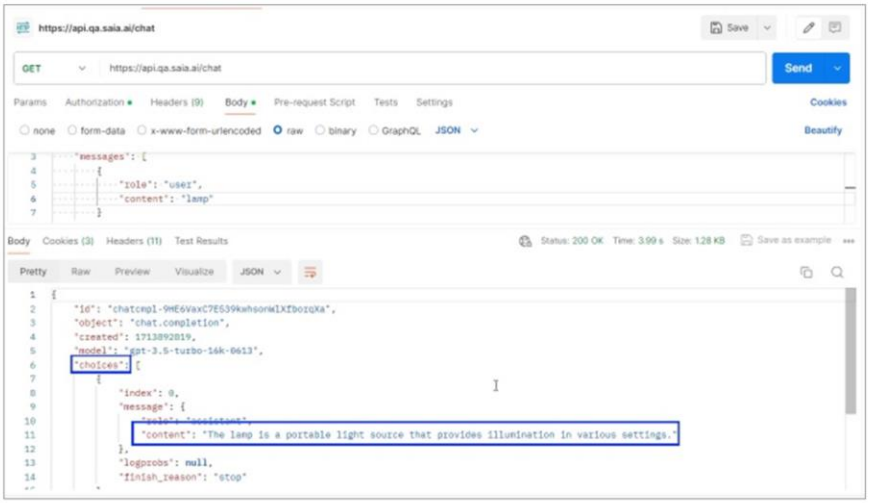
Aplicación GeneXus que interactúa con un Chat Assistant

```
//curl -X POST "$BASE_URL/chat"  -H "Authorization: Bearer $SAIA_PROJECT_APITOKEN"  -H "Content-Type: application/json"  --data '{    "model": "saia:assista

&HttpClient.Secure = 1
&HttpClient.Host = "api.qa.saia.ai"

&HttpClient.AddHeader(!"Authorization", !"Bearer default_OuK5BwzqSLjNEHwUf-GV0QCLI2YHYxBBGAshAg1CiLSa9lFgeZKcQr5S0xsehbjbg1RDbe6IcOOGbJWeIaaGbiRimSgtX
&HttpClient.AddHeader(!"Content-Type", !"application/json")

&HttpClient.AddString(!'{"model":"saia:assistant:MarketingAssistant","messages": [{ "role":"user", "content":"' + &ProductName.Trim() + '"} ],"revisionId":2

&HttpClient.Execute(!"POST", !"/chat")
```

Debemos ahora definir la ejecución del POST, y para eso debemos completar el path con /chat.

Bien. Ya tenemos definida la ejecución de la consulta. ¿Qué debemos hacer ahora? Recibir la respuesta del asistente y devolver solamente la descripción del producto.

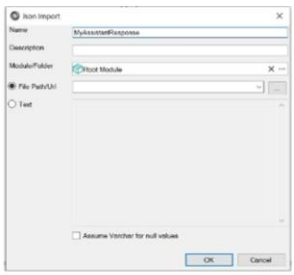
Aplicación GeneXus que interactúa con un Chat Assistant

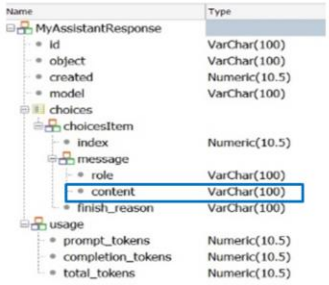


Si observamos en Postman la ejecución de la consulta y la respuesta recibida, vemos que tiene la forma de una estructura, donde la descripción que nos interesa queda guardada en el item “content”, del primer elemento de la colección Choices.

La salvamos entonces como json.

Aplicación GeneXus que interactúa con un Chat Assistant






```
//curl -X POST "$BASE_URL/chat"  -H "Authorization: Bearer $SAIA_PROJECT_APITOKEN"  -H "Content-Type: application/json"  --data '{    "model": "saia:assista

&HttpClient.Secure = 1
&HttpClient.Host = "api.qa.saia.ai"

&HttpClient.AddHeader(!"Authorization", !"Bearer default_OuK5BwzqSLjNEHwUf-GV0QCLI2YHYxBBGAshAg1CiLSa9lFgeZKcQr5S0xsehbjbg1RDbe6IcOOGbJWeIaaGbiRimSgtX
&HttpClient.AddHeader(!"Content-Type", !"application/json")

&HttpClient.AddString(!'{"model":"saia:assistant:MarketingAssistant","messages": [{ "role":"user", "content":"' + &ProductName.Trim() + '"} ],"revisionId":2

&HttpClient.Execute(!"POST", !"/chat")

&MyAssistantResponse.FromJson(&HttpClient.ToString())

&MyChoises = &MyAssistantResponse.choices

&ProductDescription = &MyChoises.Item(1).message.content
```

Volvemos a nuestra KB e importamos este archivo json para que GeneXus cree automáticamente el tipo de dato estructurado que lo represente. Para eso vamos a Tools / Application Integration / Json import.

Le ponemos como nombre MyAssistantResponse, y cargamos el archivo.

Presionamos Ok y vemos creado el sdt correspondiente. Observemos su estructura. Tenemos acá el item que estamos buscando.

Definimos entonces la variable &MyAssistantResponse basada en este mismo tipo de dato….. y la cargamos con la respuesta, aplicando el método FromJson que requiere a su vez de un string.

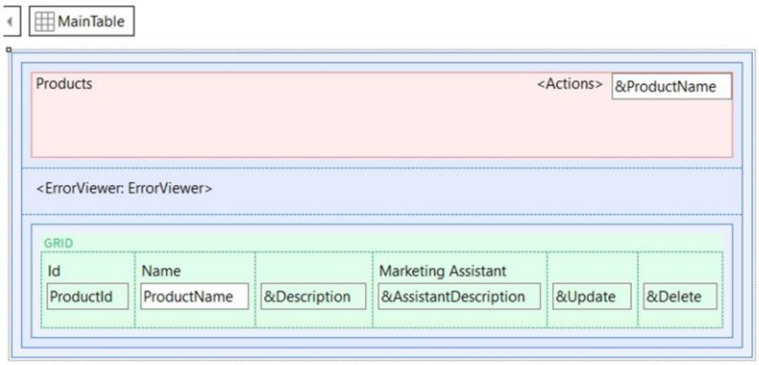
Observemos nuevamente el dato estructurado. Para llegar al item “content” debemos recuperar la colección Choices.

Así que definimos la variable &MyChoices basada en el tipo de dato &MyAssistantResponse.choicesItem.

Debe ser una colección, así que marcamos la casilla IsCollection. Cargamos la colección

Finalmente, en la variable &ProductDescription, que es la variable de retorno del procedimiento, cargamos el valor de “content” del item 1 de la colección, que es donde se guarda la respuesta que buscamos del asistente.

Aplicación GeneXus que interactúa con un Chat Assistant



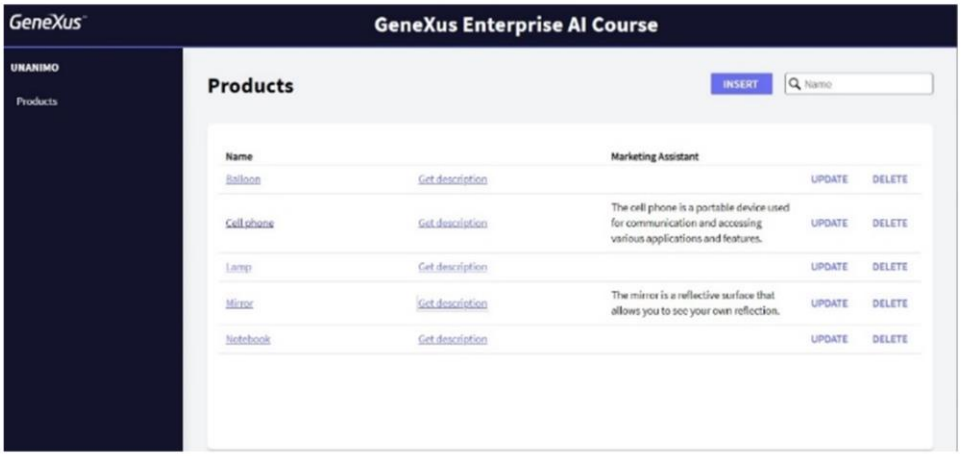

```
Event &Description.click()
    &AssistantDescription = MyChatAssistant(ProductName)
endevent
```

Bien. ¿Qué debemos hacer ahora? Llamar a este procedimiento desde el web panel WWProduct.

Para eso, en la solapa de eventos, definimos el evento Click aplicado a la variable &Description.

Para probar presionamos F5.

Aplicación GeneXus que interactúa con un Chat Assistant



Ejecutamos el web panel. Elegimos un producto y presionamos Get description.

Vemos entonces la respuesta devuelta por el asistente.

GeneXus by Globant

GeneXus™
by Globant

training.genexus.com