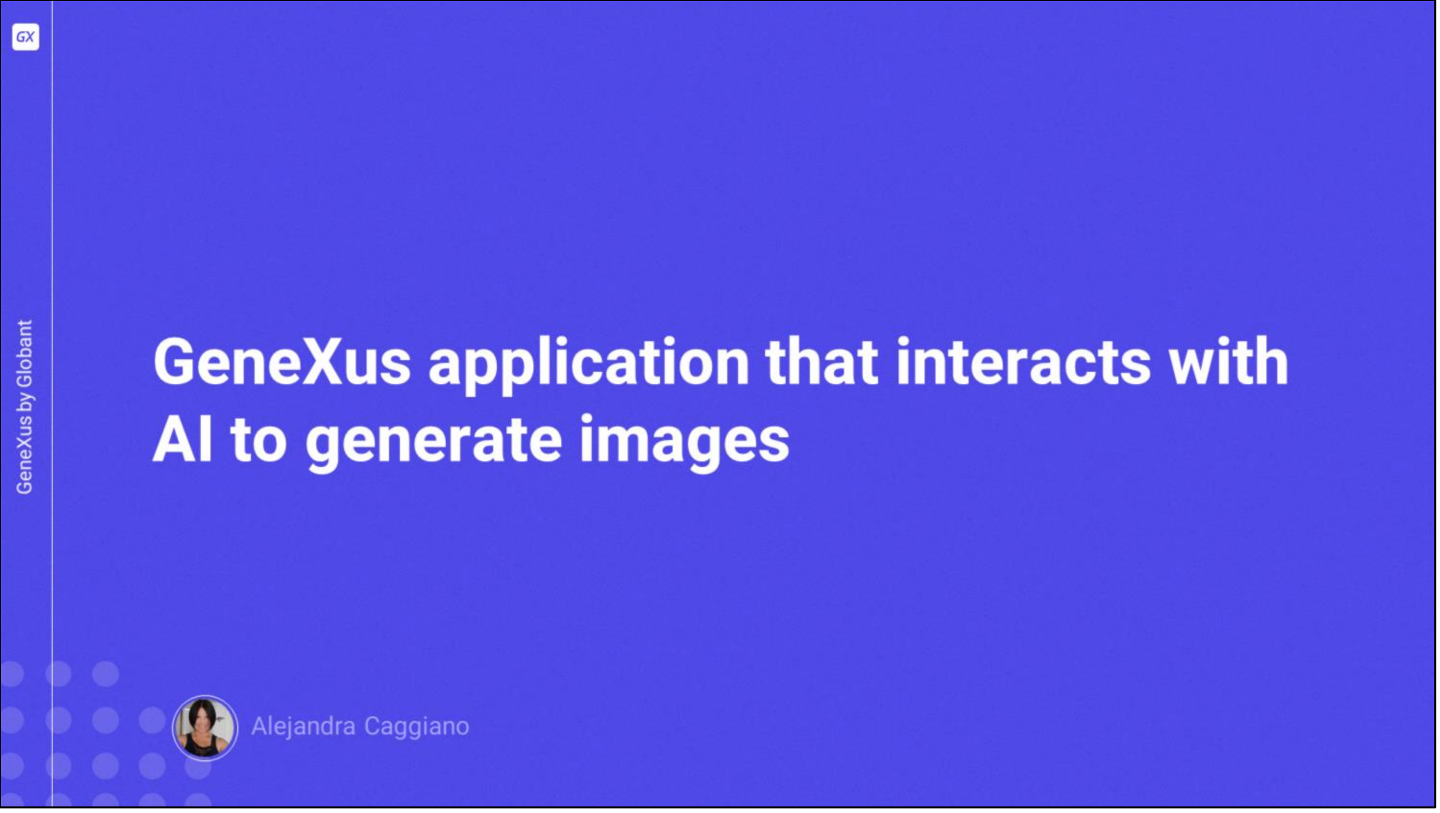
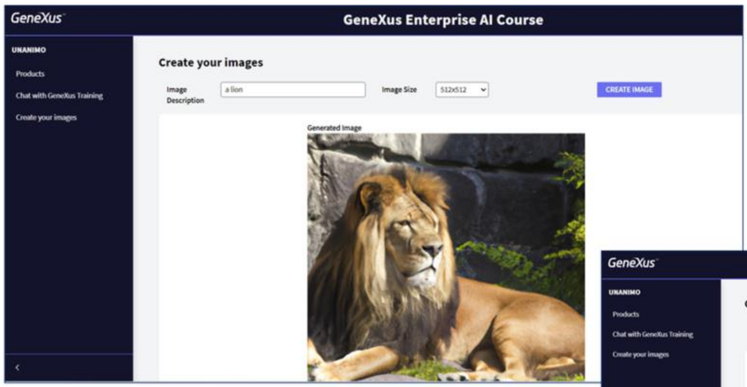


GeneXus by Globant

GeneXus application that interacts with AI to generate images

Alejandra Caggiano

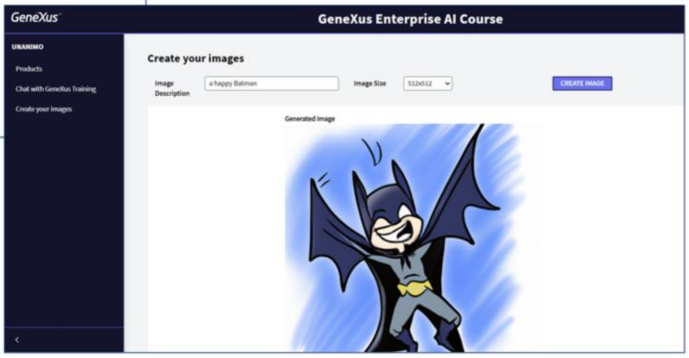
GeneXus application that interacts with AI to generate images



Create your images

Image Description: a lion

Image Size: 512x512

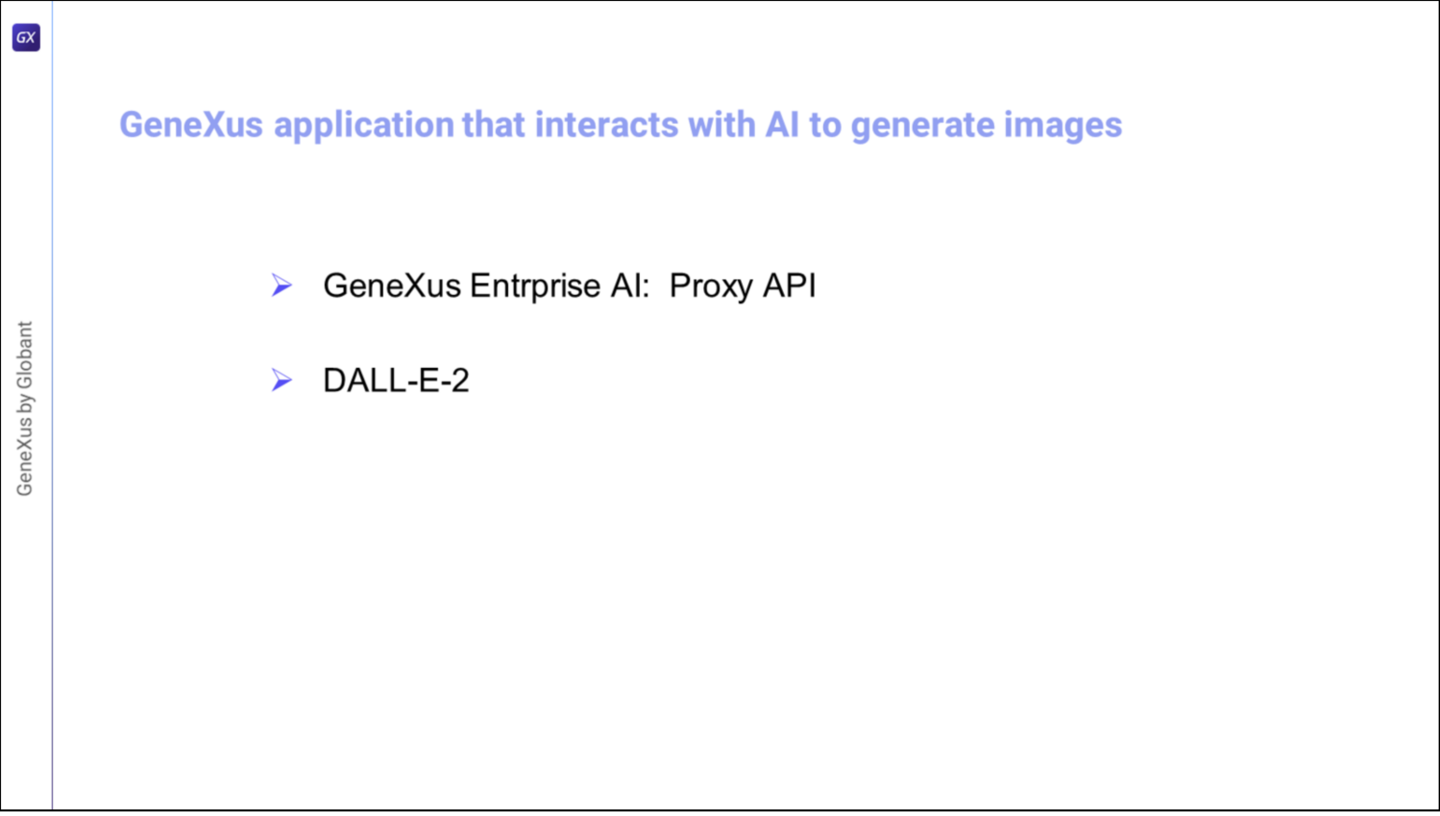


Create your images

Image Description: a happy Batman

Image Size: 512x512

We already have our GeneXus application that interacts with a Chat assistant and a RAG assistant. We are now going to enable the possibility of generating images.

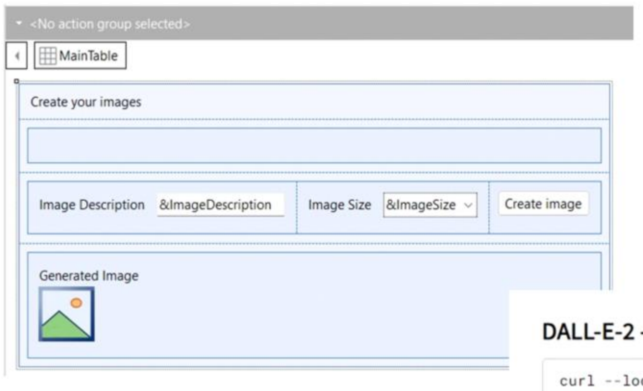


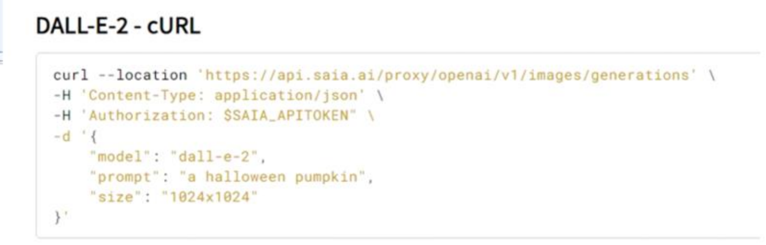
GeneXus application that interacts with AI to generate images

- GeneXus Entrprise AI: Proxy API
- DALL-E-2

For that, we are going to use the Proxy API to interact with an AI system that can generate images from a description in natural language.

GeneXus application that interacts with AI to generate images




DALL-E-2 - cURL

```
curl --location 'https://api.saia.ai/proxy/openai/v1/images/generations' \
-H 'Content-Type: application/json' \
-H 'Authorization: $SAIA_APITOKEN' \
-d '{
    "model": "dall-e-2",
    "prompt": "a halloween pumpkin",
    "size": "1024x1024"
}'
```

Let's go to GeneXus, to our knowledge base.

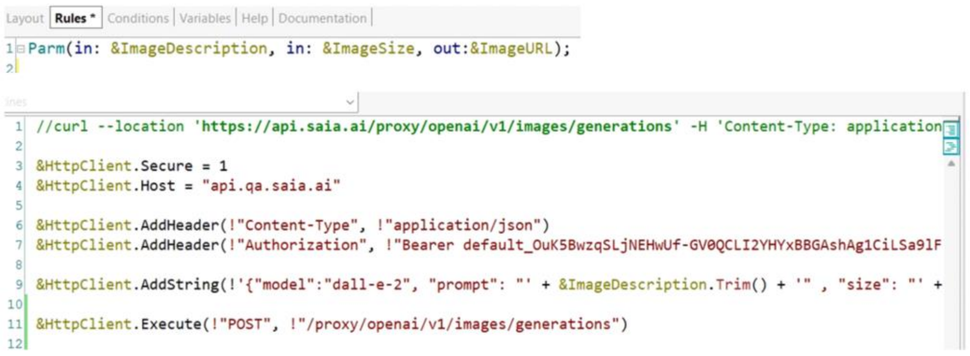
We have already created a web panel called CreateImages, where we enable the end user to enter a brief description, and select the size of the image to be generated.

The functionality that we want to achieve is the code in the event associated with the Create Image button. And for that we must create a procedure that receives the description and size of the image as input parameters and returns the URL of the generated image. From that URL we will show the image in the &GeneratedImage variable included in the web panel form.

We go to Tools / Integration application / cURL Inspector.

We name it GenerateImages and load the cURL sample, which in this case corresponds to the interaction of the Proxy API with DALL-E-2.

GeneXus application that interacts with AI to generate images



```
Parm(in: &ImageDescription, in: &ImageSize, out:&ImageURL);
```

```
//curl --location 'https://api.saia.ai/proxy/openai/v1/images/generations' -H 'Content-Type: application

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

&HttpClient.AddHeader(!"Content-Type", !"application/json")
&HttpClient.AddHeader(!"Authorization", !"Bearer default_OuK5BwzqSLjNEHwUf-GV0QCLI2YHYxBBGAshAg1CiLSa9lF

&HttpClient.AddString(!'{"model":"dall-e-2", "prompt": "' + &ImageDescription.Trim() + '" , "size": "' +

&HttpClient.Execute(!"POST", !"/proxy/openai/v1/images/generations")
```

As we already know, this generates the base procedure where we will complete the connection definition with the necessary data, according to our context.

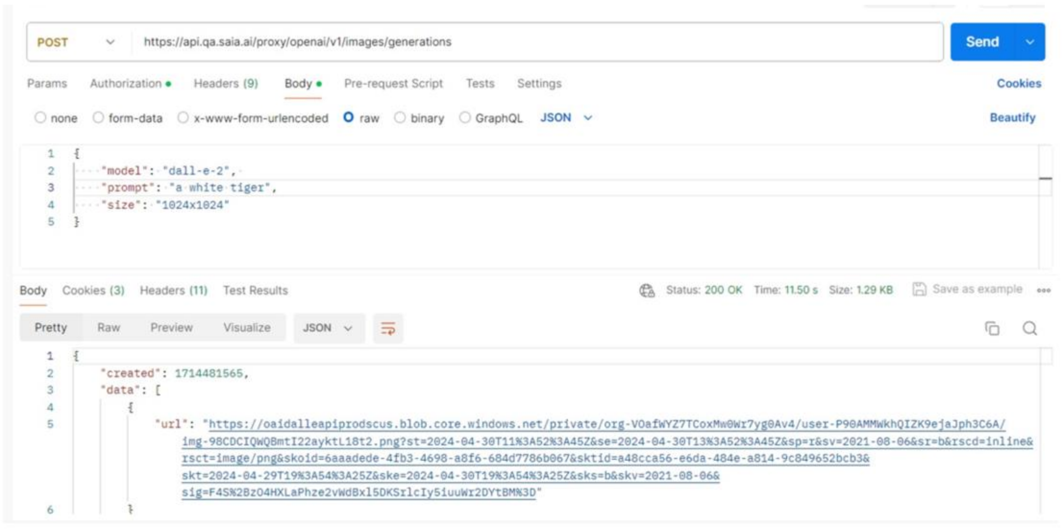
First, we declare the Parm rule to receive the description and size of the image, and return the URL of the generated image.

Let's analyze the source:

We define the HTTPS connection protocol… …define the host, the project Api Token… …and define the body of the request so that it considers the input parameters corresponding to the description and size of the image required by the end user.

Then we define the POST for its execution.

GeneXus application that interacts with AI to generate images



```
https://api.qa.saia.ai/proxy/openai/v1/images/generations
```

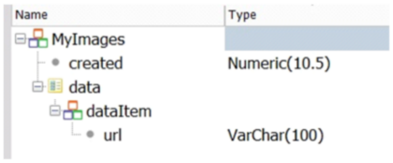
```
{
    "model": "dall-e-2",
    "prompt": "a white tiger",
    "size": "1024x1024"
}
```

```
{
    "created": 1714481565,
    "data": [
        {
            "url": "https://oaidalleapiprodscus.blob.core.windows.net/private/org-VOafWYZ7TCoxMw0Wr7yg0Av4/user-P90AMMWkhQIZK9ejaJph3C6A/
                img-98CDCIQWQBmtI22ayktL18t2.png?st=2024-04-30T11%3A52%3A45Z&se=2024-04-30T13%3A52%3A45Z&sp=r&sv=2021-08-06&sr=b&rscd=inline&
                rsct=image/png&skoid=6aaadede-4fb3-4698-a8f6-684d7786b067&sktid=a48cca56-e6da-484e-a814-9c849652bcb3&
                skt=2024-04-29T19%3A54%3A25Z&ske=2024-04-30T19%3A54%3A25Z&sks=b&skv=2021-08-06&
                sig=F4S%2Bz04HXLaPhze2vWdBxl5DKSrlcIy5iuuWr2DYtBM%3D"
        }
```

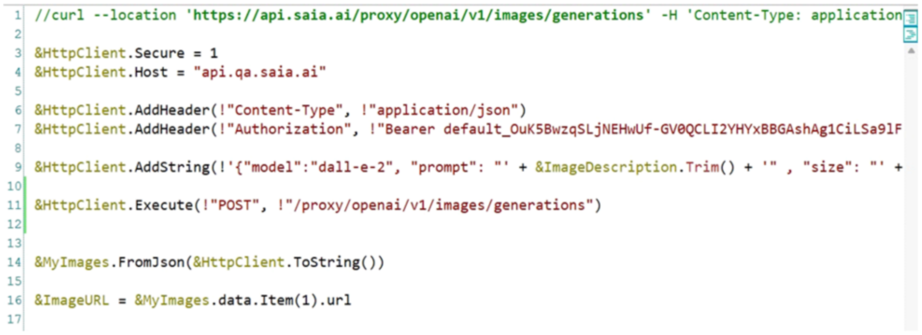
What happens to the response received?

We repeat the same process as in the previous examples. From Postman we save the request response and import the file into our knowledge base, using the option Tools / Application integration / Json Import.

GeneXus application that interacts with AI to generate images



Name	Type
MyImages	
created	Numeric(10.5)
data	
dataItem	
url	VarChar(100)



```
//curl --location 'https://api.saia.ai/proxy/openai/v1/images/generations' -H 'Content-Type: application

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

&HttpClient.AddHeader(!"Content-Type", !"application/json")
&HttpClient.AddHeader(!"Authorization", !"Bearer default_OuK5BwzqSLjNEHwUf-GV0QCLI2YHYxBBGAshAg1CiLSa9lF

&HttpClient.AddString(!'{"model":"dall-e-2", "prompt": "' + &ImageDescription.Trim() + '" , "size": "' +

&HttpClient.Execute(!"POST", !"/proxy/openai/v1/images/generations")


&MyImages.FromJson(&HttpClient.ToString())

&ImageURL = &MyImages.data.Item(1).url

```

We name it MyImages, and this generates the following structured data type:

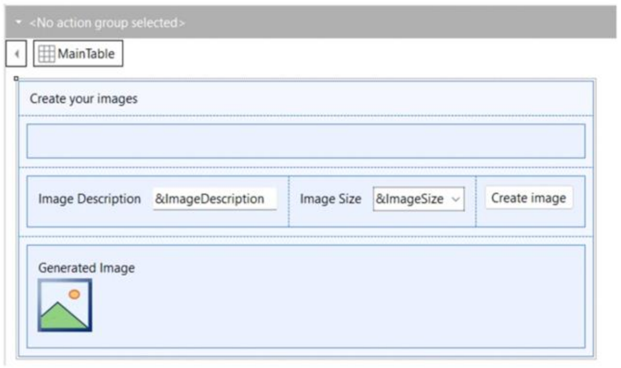
...where the URL element of the first item of the "data" collection will correspond to the URL of the generated image and will be the output parameter of our procedure.

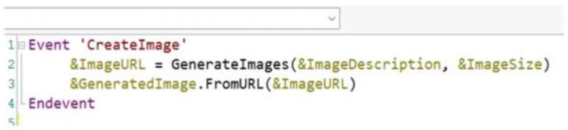
So we define the &MyImages variable, based on the SDT, and load it with the request response by applying the FromJson method, as we have already seen in the previous examples.

Finally, we load the &ImageURL variable with the value of the "url" element of the first item of the "data" collection.

GeneXus by Globant

GeneXus application that interacts with AI to generate images





```
Event 'CreateImage'
      &ImageURL = GenerateImages(&ImageDescription, &ImageSize)
      &GeneratedImage.FromURL(&ImageURL)
Endevent
```

Once this procedure has been defined, we must call it from the event associated with the web panel button.

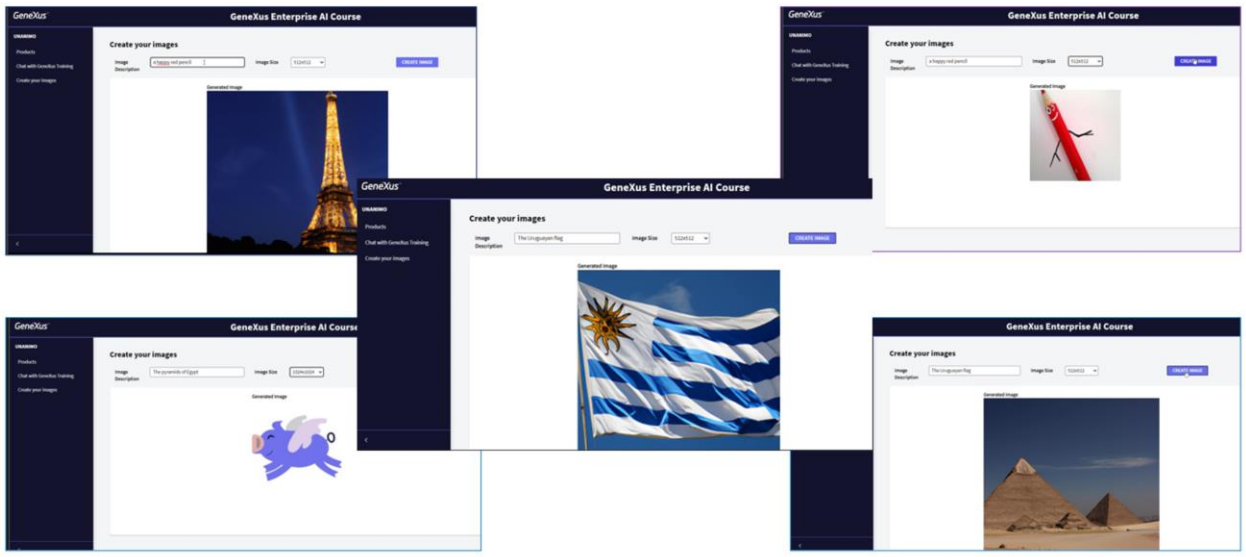
So we make the call and receive its output in the &ImageURL variable.

Finally, the &GeneratedImage variable, included in the form, is loaded with the image obtained from the URL received.

To test it, we press F5.

GeneXus by Globant

GeneXus application that interacts with AI to generate images



First, we are going to ask for an image of the Eiffel Tower at night, and we choose 512x512 as its size.

If we don't like the image we receive, we ask for another one.

Let's look at other more creative examples, such as a red pencil smiling...

Or a flying blue piggy, sized 256x256… The pyramids of Egypt... …and the flag of Uruguay.

GeneXus by Globant

GeneXus™
by Globant

training.genexus.com