Curso práctico de actualización a

PARTE 2

Julio 2019

CONTENIDO

OBJETIVO

Veremos la potencia del generador de chatbots introducido en la versión GeneXus 16.

Crearemos un chatbot para una realidad de una aplicación que brinda servicios al ciudadano. Será un chatbot tanto para la aplicación WEB como para la aplicación SD de esta realidad (lo haremos en Android por la facilidad que brinda en la prototipación).

Usaremos Watson como Natural Language Processing (NLP) Provider. Pero bien podría ser Dialogflow, sin ningún problema.

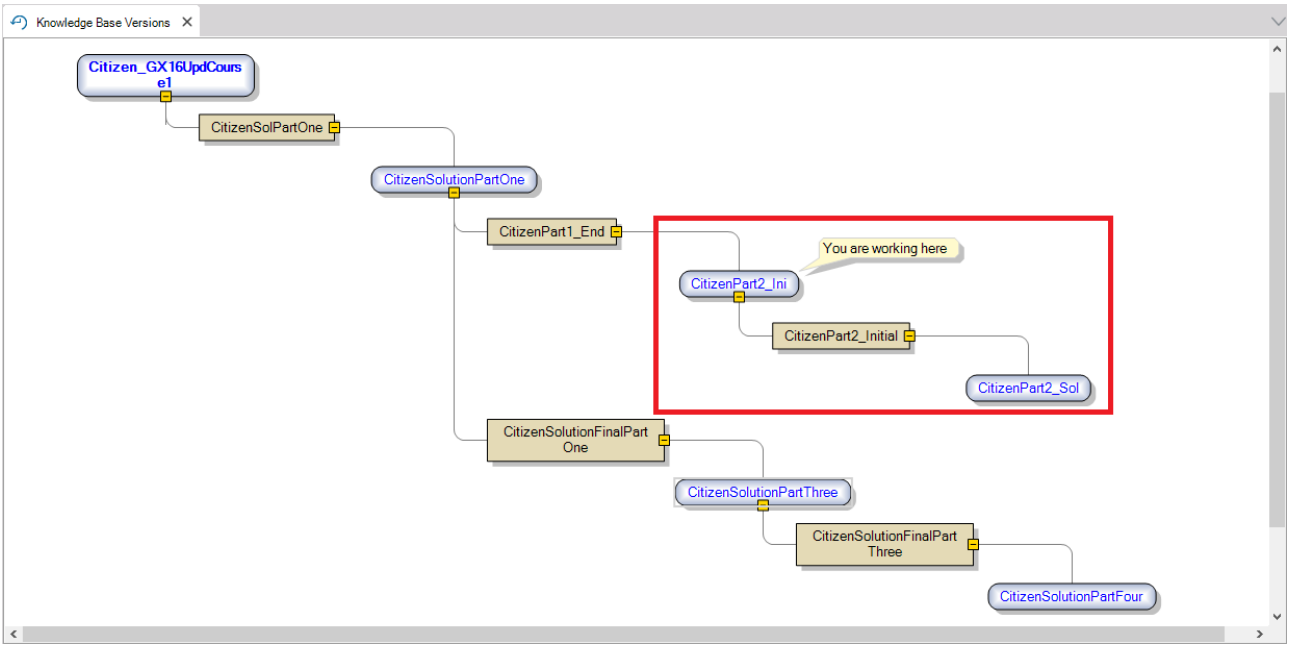
A través de nuestro chatbot los usuarios podrán:

- Hacer un reclamo por cualquier tema ocurrido en la calle; puede ser un árbol caído, una señal de tránsito que falta o deteriorada, un auto mal estacionado, etc.
- Obtener información acerca de actividades culturales que se desarrollan, o de los requisitos para trámites administrativos.

ALGÚN AJUSTE PREVIO

Obviamos la configuración de la cuenta en el NLP provider, dado que para este práctico ya se ha realizado. El instructor le brindará la información requerida cuando la necesite.

Se partirá de la versión CitizenPart2_Ini de la KB Citizen_GX16Course en http://samples.genexusserver.com/v16.



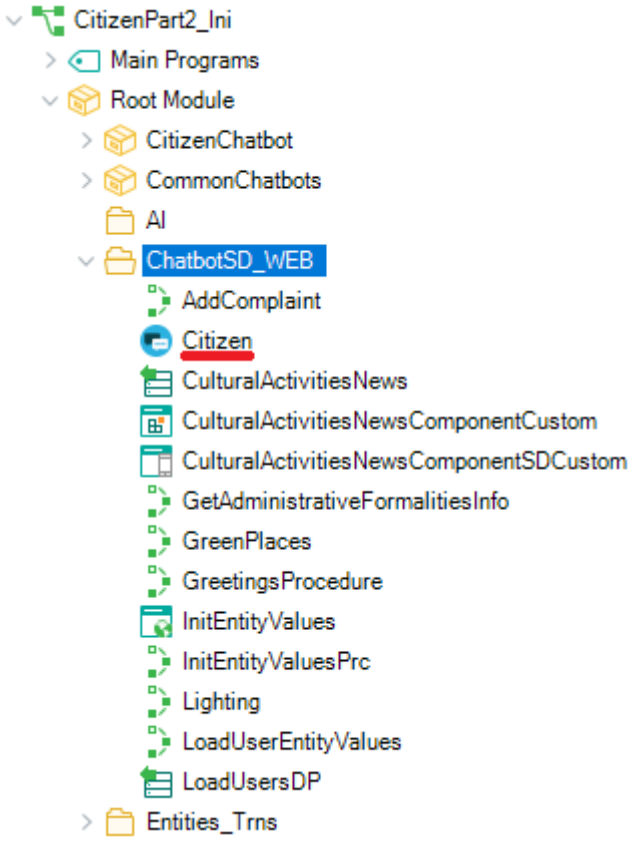
Esta versión de la KB viene con folders, objetos y módulos ya creados para no perder tiempo.

COMENCEMOS

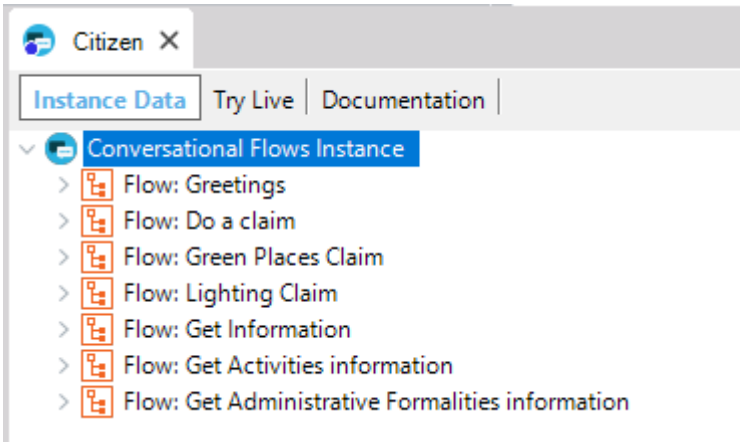
¿CUÁLES SON LOS CONCEPTOS BASE DETRÁS DE UN CHATBOT?

Un chatbot está representado por un Objeto de la KB, nuevo en GeneXus 16, llamado Conversational Flows.

En la KB, dentro del folder ChatbotSD_WEB, ya tenemos creado un objeto Conversational Flows, que nos servirá como base para construir nuevas funcionalidades para nuestra solución de chatbots.



Ábralo y verá en qué consiste el diseño de un chatbot en GeneXus:



Tiene una estructura de árbol, en donde cada nodo principal, “Flow”, representa una intención del usuario final, que nuestro chatbot va a saber contestar apropiadamente.

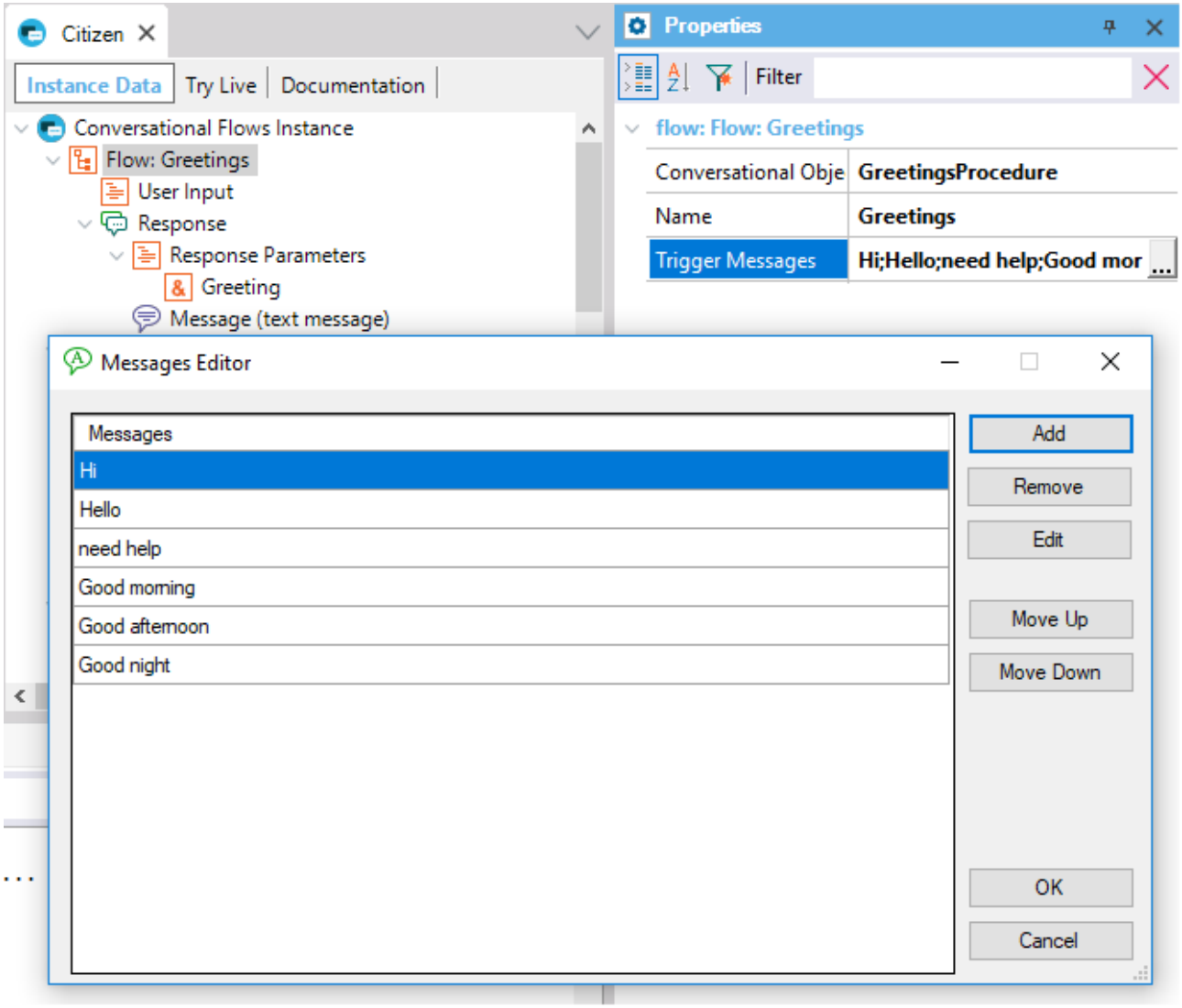
¿Qué es un *intent*? Refleja la voluntad del usuario al momento de hacer la consulta, por ejemplo “Hacer reclamo por un auto mal estacionado”.

En el diagrama del Conversational Flows Instance, el Flow tiene una relación directa con el Intent.

¿Cómo interpreta el NLP provider la intención del usuario?

Lo hace a través de los **Trigger messages** que agreguemos en el Flow; cuanto más completos, mejor podrá hacer la inferencia.

Observe en el "Flow: Greetings" la propiedad **Trigger Messages**, que tiene varias posibilidades de saludo:

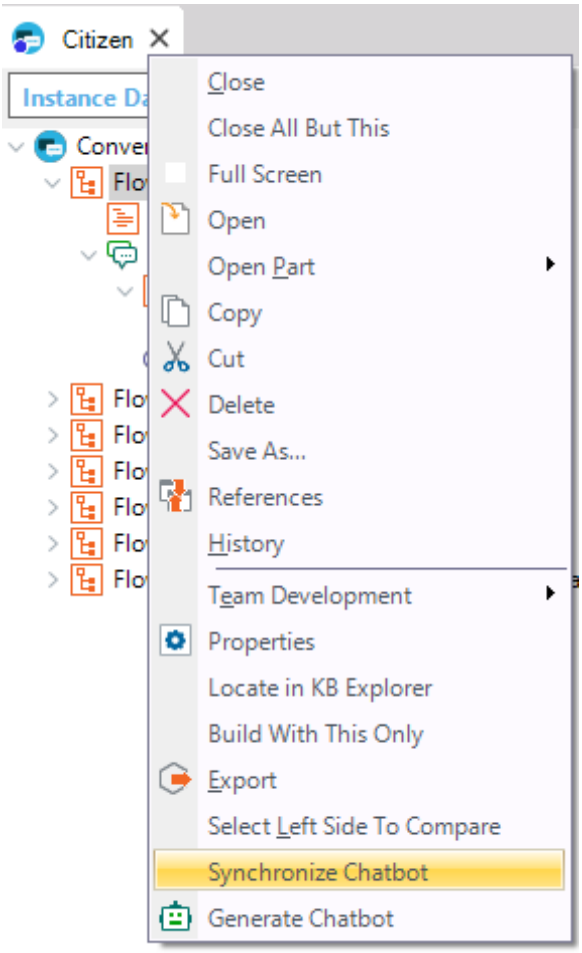


Este chatbot que tenemos en la KB aún no ha sido sincronizado con el Provider.

En el objeto Citizen, en el nodo padre "Conversational Flows Instance", configuremos en las propiedades User name y User password los valores que el instructor deberá proveerle.

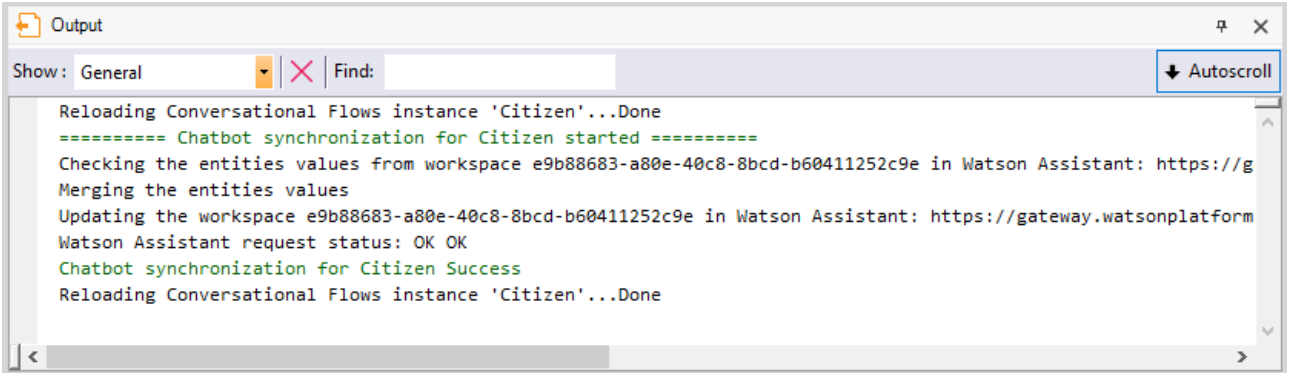
Salvemos el objeto, y veamos cómo se crea el diálogo, con sus intents y Trigger messages en Watson.

Haciendo botón derecho sobre el objeto, podemos seleccionar la opción Synchronize Chatbot:



Esta acción genera un archivo de metadata que se impacta en el NLP provider, para generar el diálogo con sus intents y entidades.

Podemos ver en el tab General del output de GeneXus, la salida de la ejecución de la opción Synchronize.



¿Qué ocurre si agregamos un Trigger message para el Flow “Greetings”?

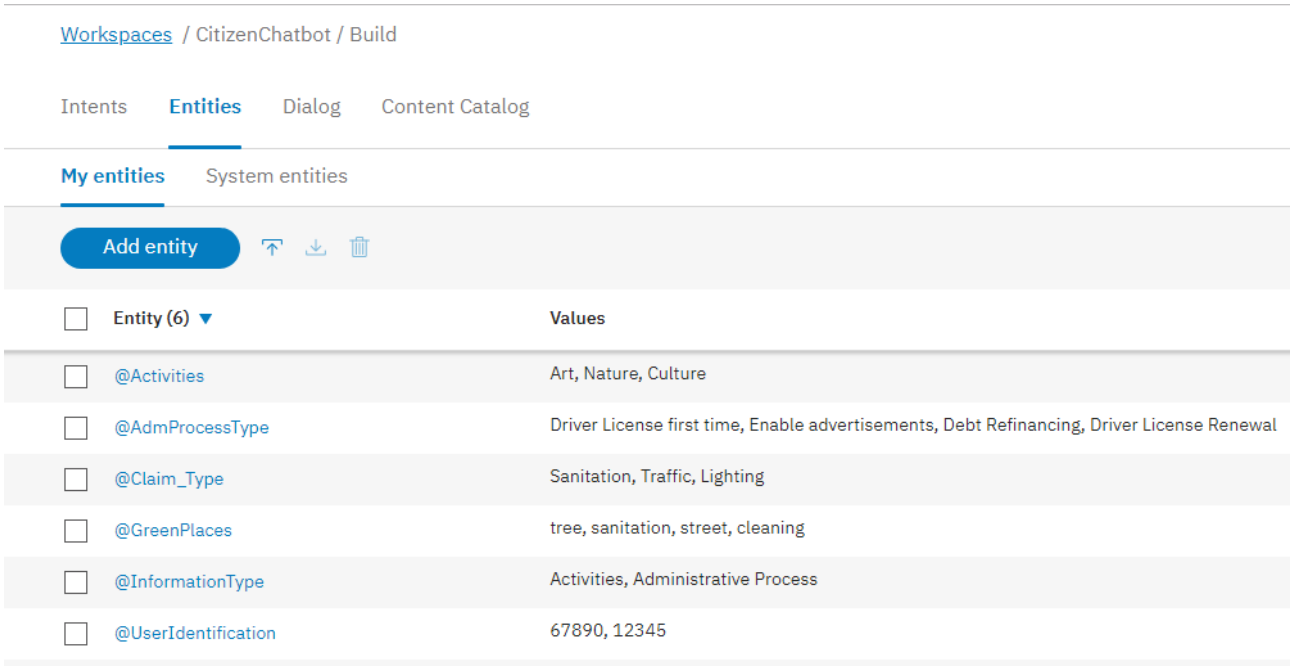
Realiza un cambio o inserta un nuevo Trigger Message en el Flow Greetings. Solo con salvar el objeto Conversational Flows se impacta el cambio en Watson.

Dada una intención del usuario se pueden distinguir diferentes objetos o tópicos dentro de esa intención, que llamamos formalmente Entidades.

Por ejemplo, dada la intención de agendar fecha y hora, una entidad de esa intención sería el **motivo** de agendar fecha, y el valor de dicha entidad por ejemplo, puede ser el **trámite de renovación** de la libreta de conducir.

Entonces, le llamamos entidad a un objeto de la consulta, que agrupa varios valores, y alguno o varios de ellos serán instanciados en la query.

Las entidades se almacenan en el NLP provider, con sus posibles valores y sinónimos.

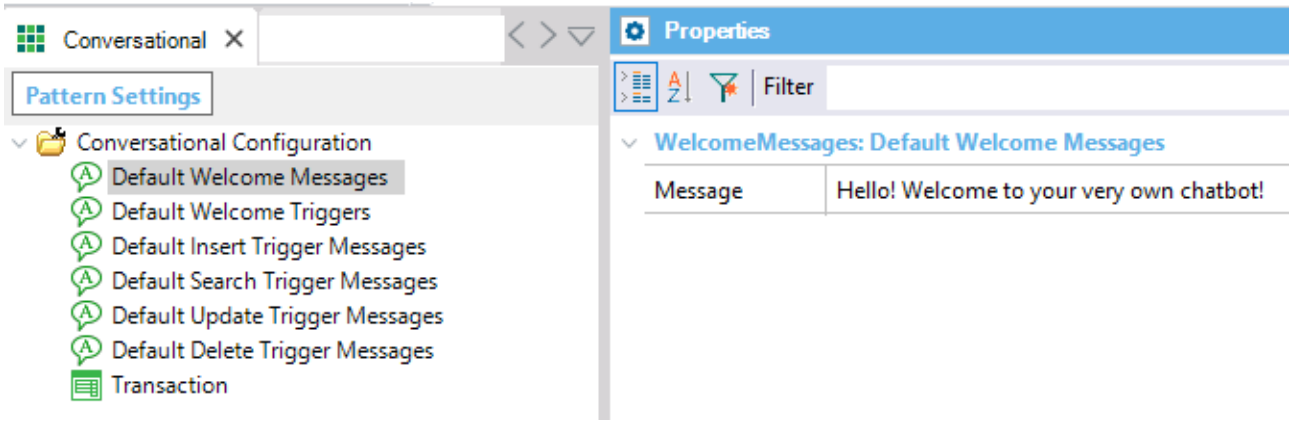


Para crear un chatbot, se necesita modelar su comportamiento (definir intents, entidades) y la respuesta que se muestra al usuario para cada uno de esos intents.

Ese modelo se hace en el objeto conversational flows (en nuestro caso, Citizen).

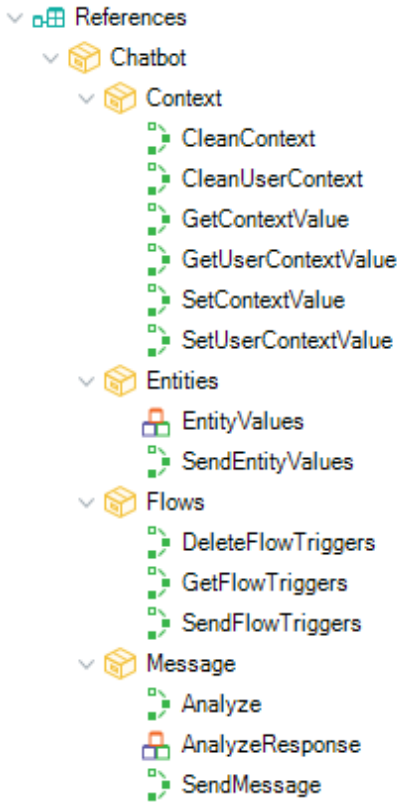
Nota: El generador de chatbots funciona como un pattern. A partir del modelo, el pattern genera automáticamente todos los objetos necesarios para resolver la conversación entre el usuario y el NLP provider.

Veamos en las Preferences, bajo Patterns, el Conversational Flows y sus propiedades.

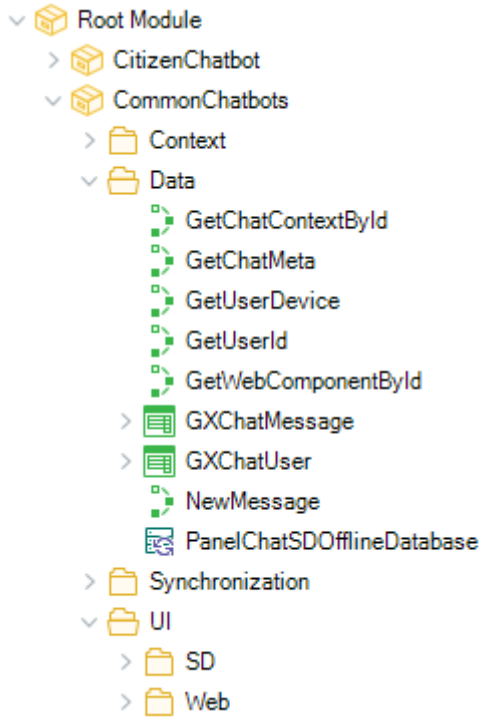


Los componentes dentro de la KB, son los siguientes:

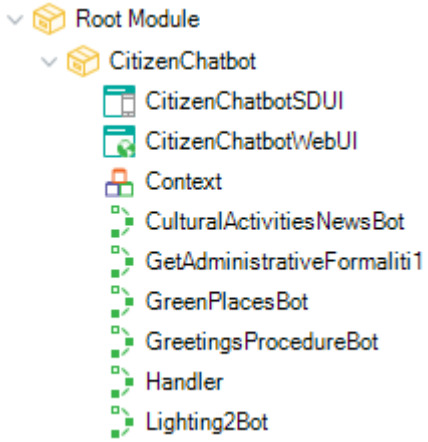
(1) Módulo Chatbots. Es un módulo externo que se instala en la KB y resuelve la comunicación con el Provider.



(2) Recursos del pattern. Son objetos de ejemplo para la creación de nuestro chatbot. Podemos modificarlos a gusto. Se encuentran en el módulo CommonChatbots.



(3) Objetos generados. A partir de nuestro modelo, se generan estos objetos en un módulo que contiene el nombre del objeto Conversational Flows (CitizenChatbot), y sirven de enlace con los objetos de nuestra KB y el servidor.



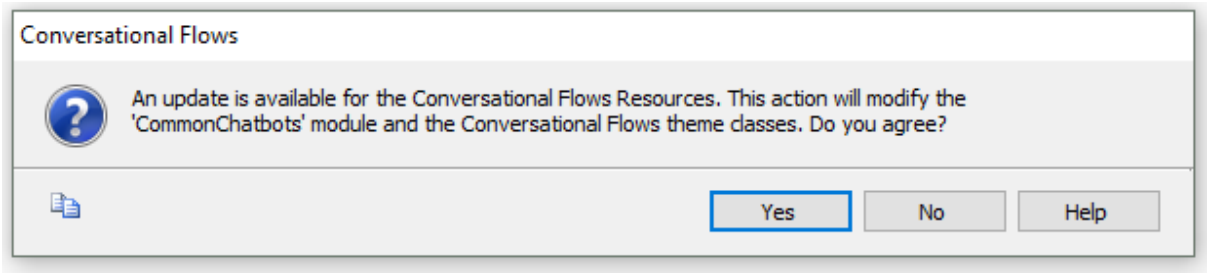
Ya entendimos los principales conceptos, y qué es el Chatbot Generator, ahora ¡empecemos a construir!

EL CHATBOT EN ACCIÓN

Hagamos botón derecho sobre el objeto Citizen, y luego Generate Chatbot.

Nota: La opción Generate Chatbot hace que se calculen los objetos que genera el pattern [3], y se importen en la KB.

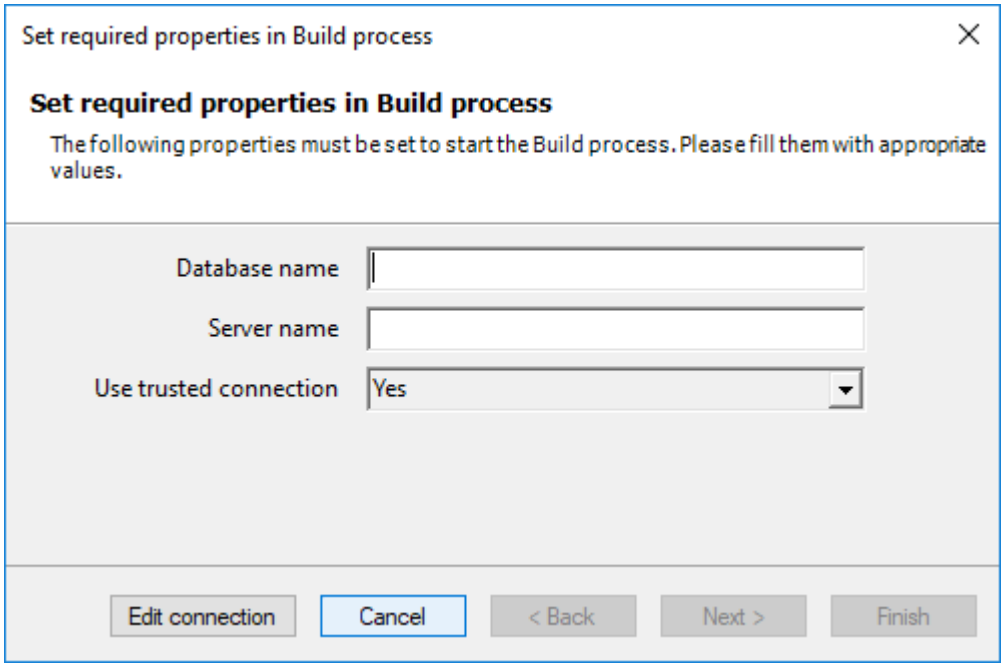
Ver el siguiente mensaje, que indica que se actualizarán los recursos[2] del generador de chatbots.



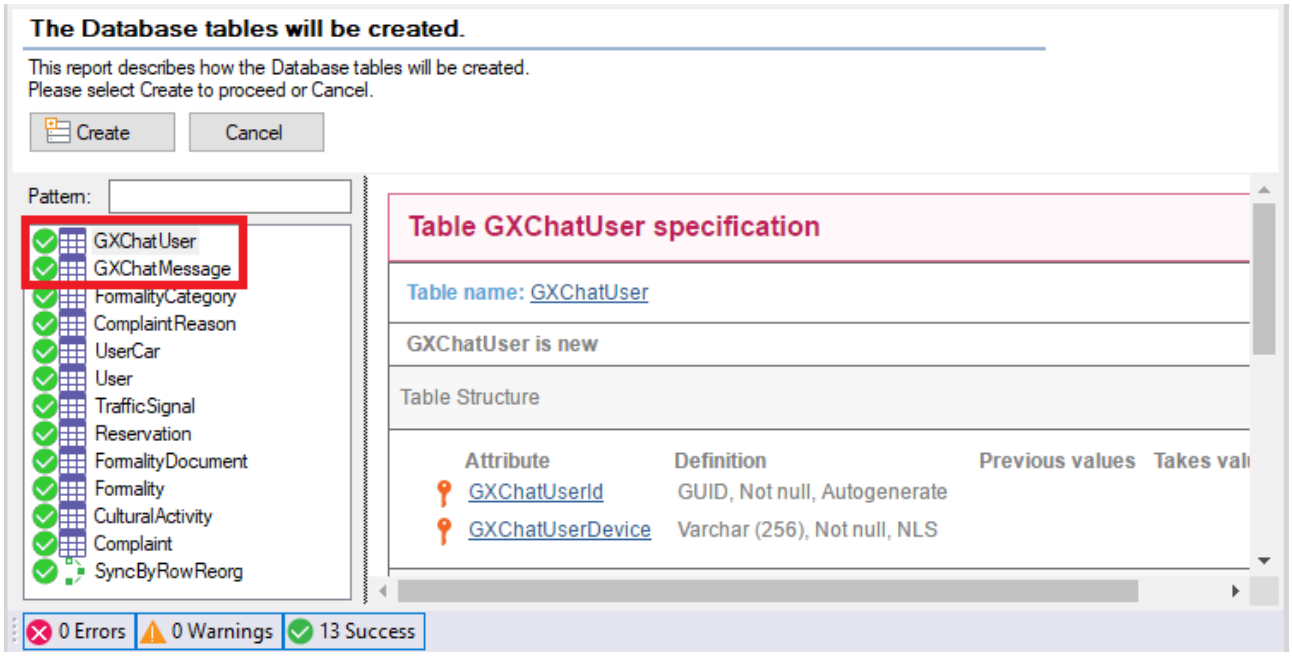
Observar en el output que se importan los recursos para Web y SD del generador de chatbots.

Hagamos un rebuild all. El build también genera los objetos del Pattern cuando es necesario (si se detectan cambios en la instancia).

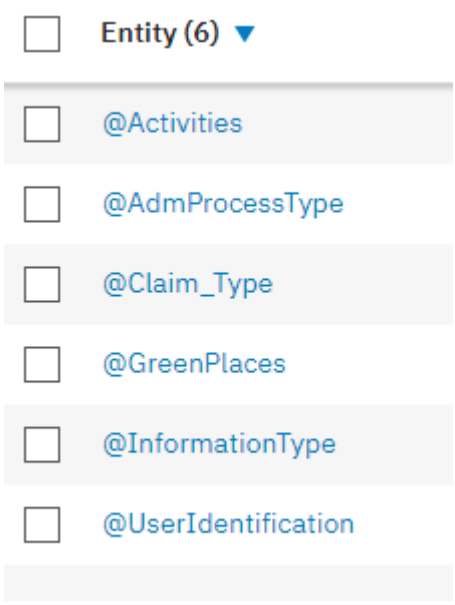
Se va a solicitar el create de la BD. Ingrese el Server de BD correspondiente al SQLServer local (estamos prototipando con un environment .Net).



Observar que hay dos tablas, llamadas GXChatMessage y GXChatUser, que son usadas por el chatbot.



Vamos a trabajar con las siguientes entidades, que puedes ver en la Interfaz de usuario para IBM watson assistant":



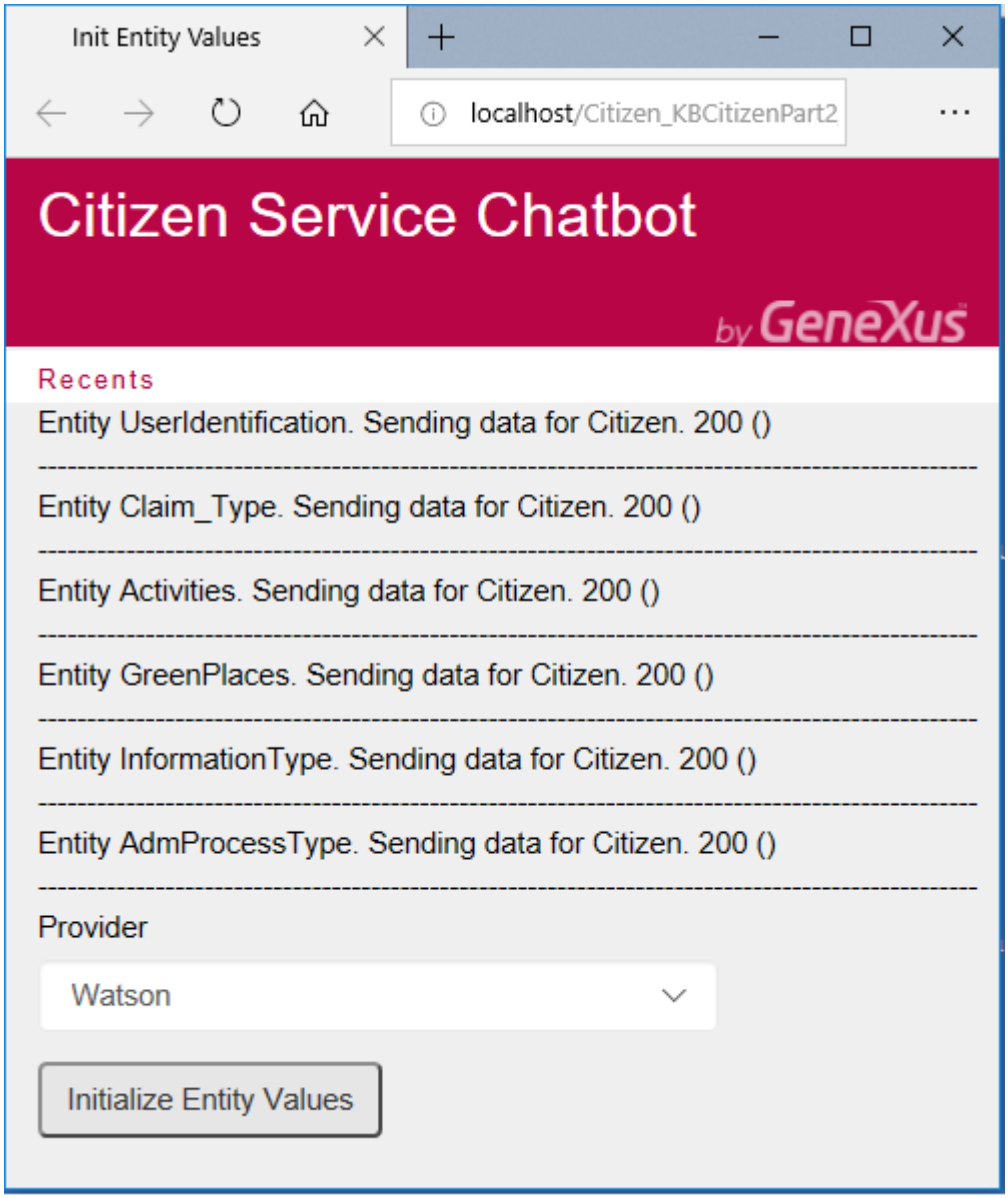
Serán cargadas en el NLP provider a través de una API provista por el módulo chatbots.

Abra el objeto InitEntityValues para conocer el método usado para subir valores y entidades a Watson (Chatbot.SendEntityValues).

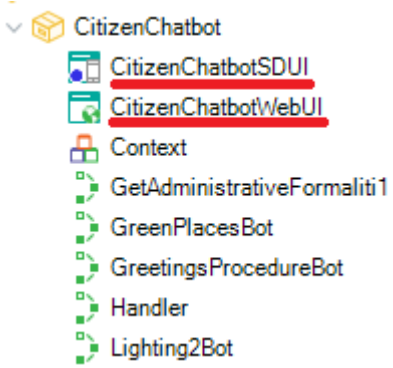
Esto siempre hace un merge con lo que tenemos en el NLP provider.

Ejecute entonces, el webpanel InitEntityValues y presione el botón Initialize Entity Values.

El resultado de la ejecución será:



Ahora ejecutemos los siguientes objetos para ver el chatbot en runtime.



Aplicación WEB: Ejecute el objeto main CitizenChatbotWebUI

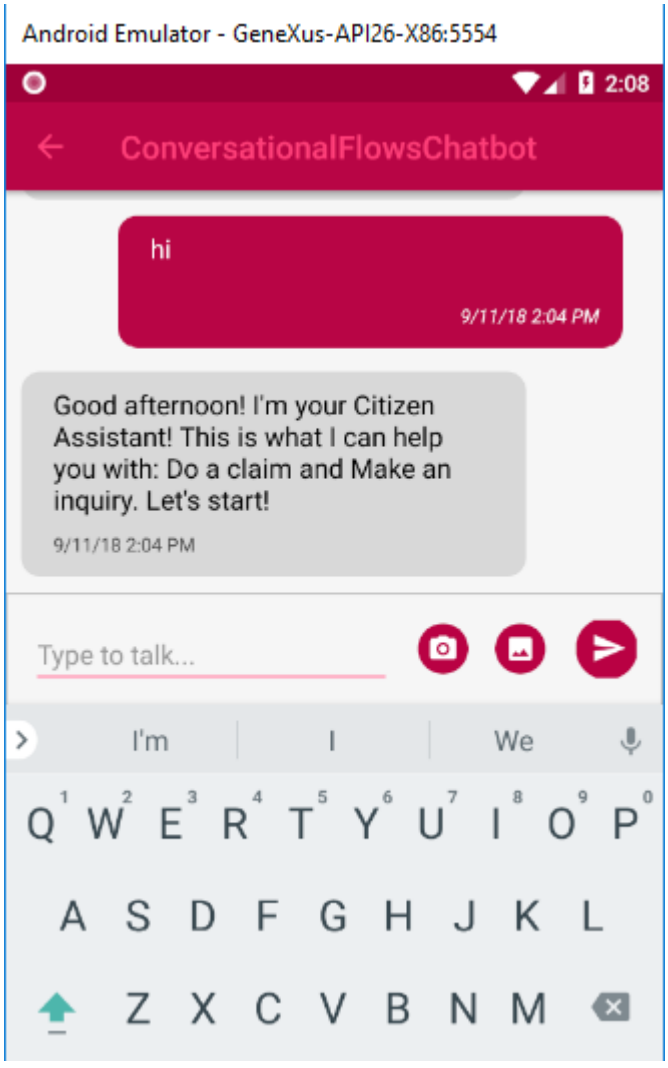
Aplicación SD: Ejecute el objeto main CitizenChatbotSDUI

Nota: Estos objetos llaman a nuestro panel principal de chatbots, pasando como parámetro el Provider y el nombre de la instancia.

```
Event 'ClientStart'
      CommonChatbots.PanelChatSD.Call(!"Citizen")
EndEvent
```

Ya puede empezar a dialogar con el chatbot Web y SD. Comience saludando al chatbot.

En SD por ejemplo:



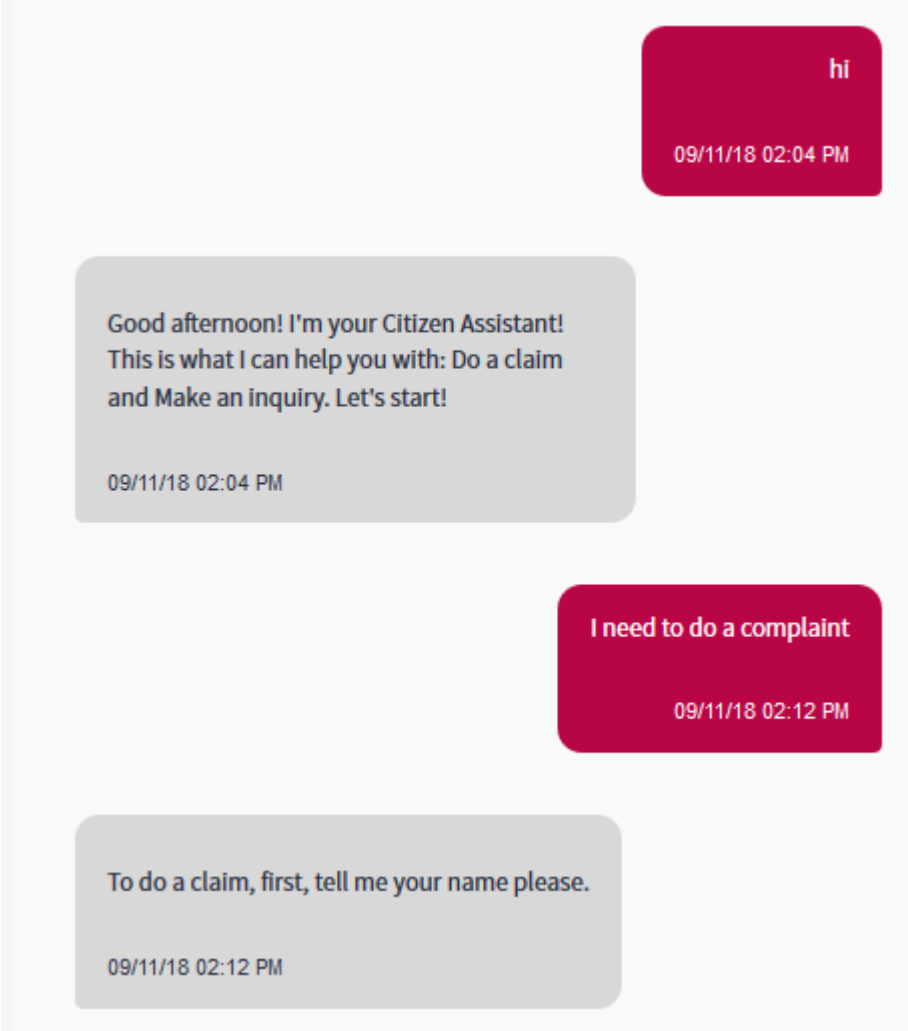
INTRODUCIENDO MEJORAS EN EL CHATBOT

Vamos a agregar intents y hacer algunas modificaciones a nuestro chatbot (Citizen).

HACER UN RECLAMO

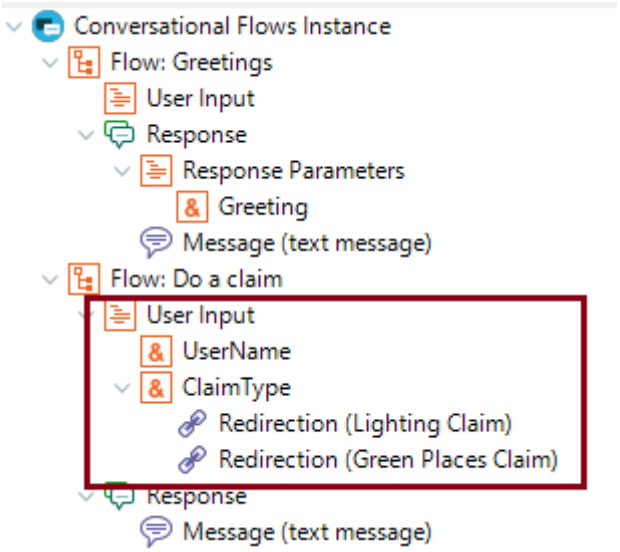
Ejecute el panel CitizenChatbotWebUI o CitizenChatbotSDUI, y salude al bot. Puede trabajar indistintamente con Web o SD.

El bot le preguntará por dónde seguir. Por el momento, ingrese “Do a claim” o un concepto similar, como en la captura siguiente:



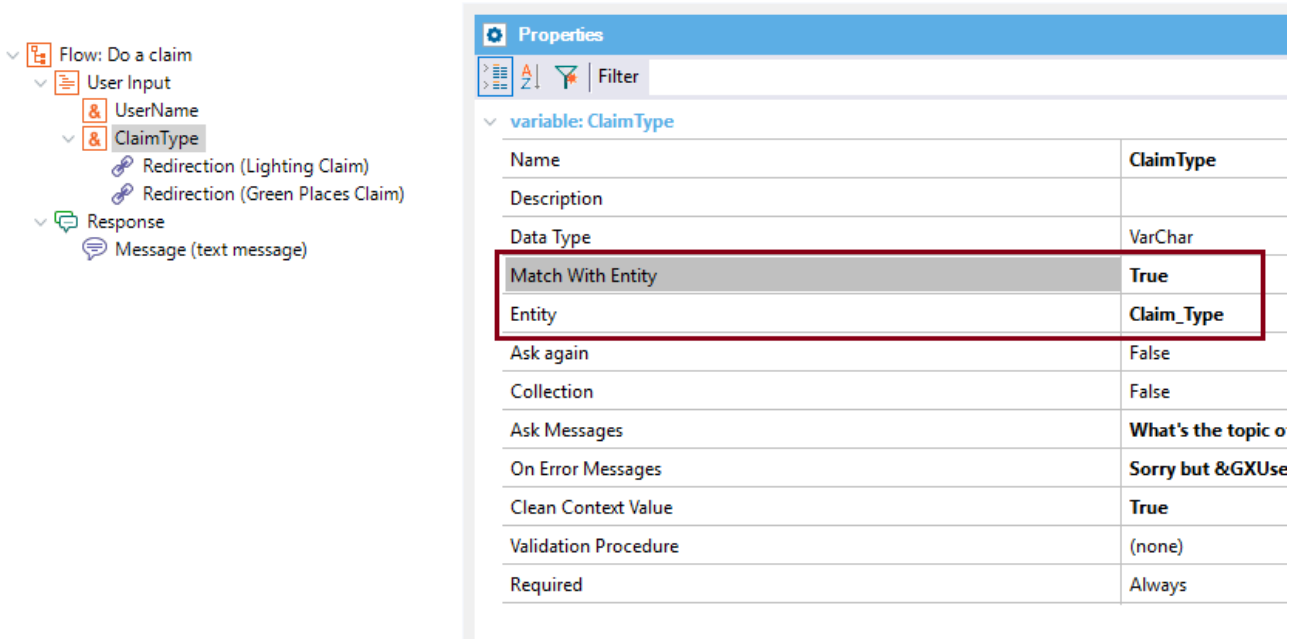
Inmediatamente se llama al Flow “Do a claim”, el cual le pedirá información para ser completado.

La información que se pide ingresar son los User Inputs que vemos en la definición del Flow.



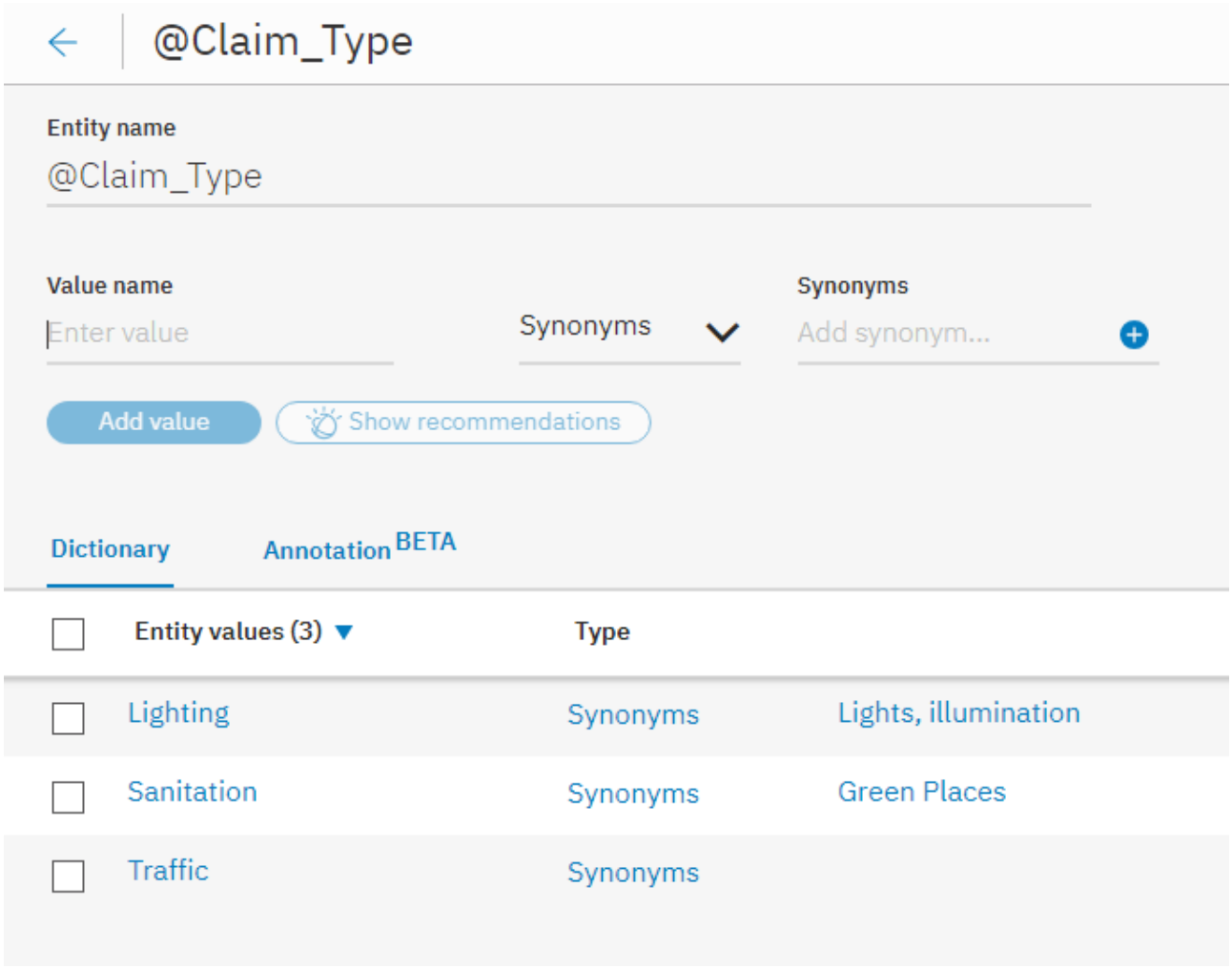
Verá que el primero es el &UserName, para el cual la propiedad Clean Context Value está apagada. Eso significa que el bot recordará ese dato en el contexto, para no volver a solicitarlo dentro de la misma sesión en ninguno de los Flows de la conversación.

Observe que el otro dato solicitado es ClaimType. Este es particular, en el sentido de que está mapeado con una entidad en Watson que lleva el mismo nombre. Eso lo notamos viendo la propiedad Match With Entity de este Input.

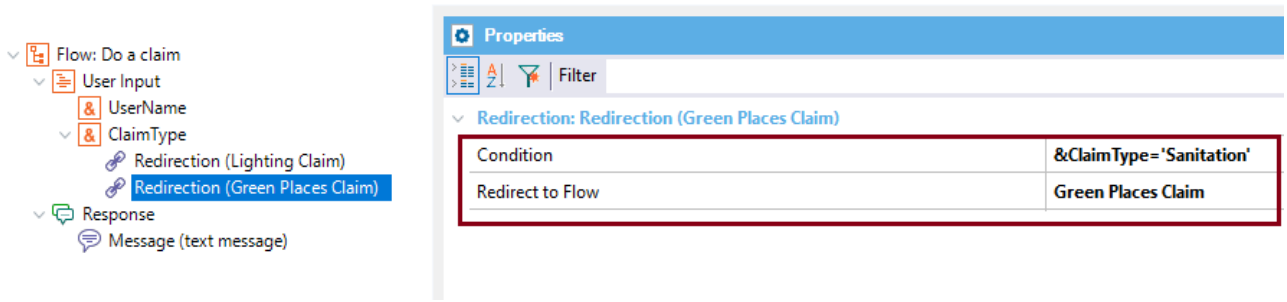


¿Qué significa que se mapea con una entidad en Watson? Que se controlará que el usuario no ingrese un dato que no se encuentre dentro de los valores (y sus sinónimos) de esta entidad. Es como un chequeo de integridad contra el Provider.

Puede ver en Watson que los valores posibles de esta entidad son los siguientes:

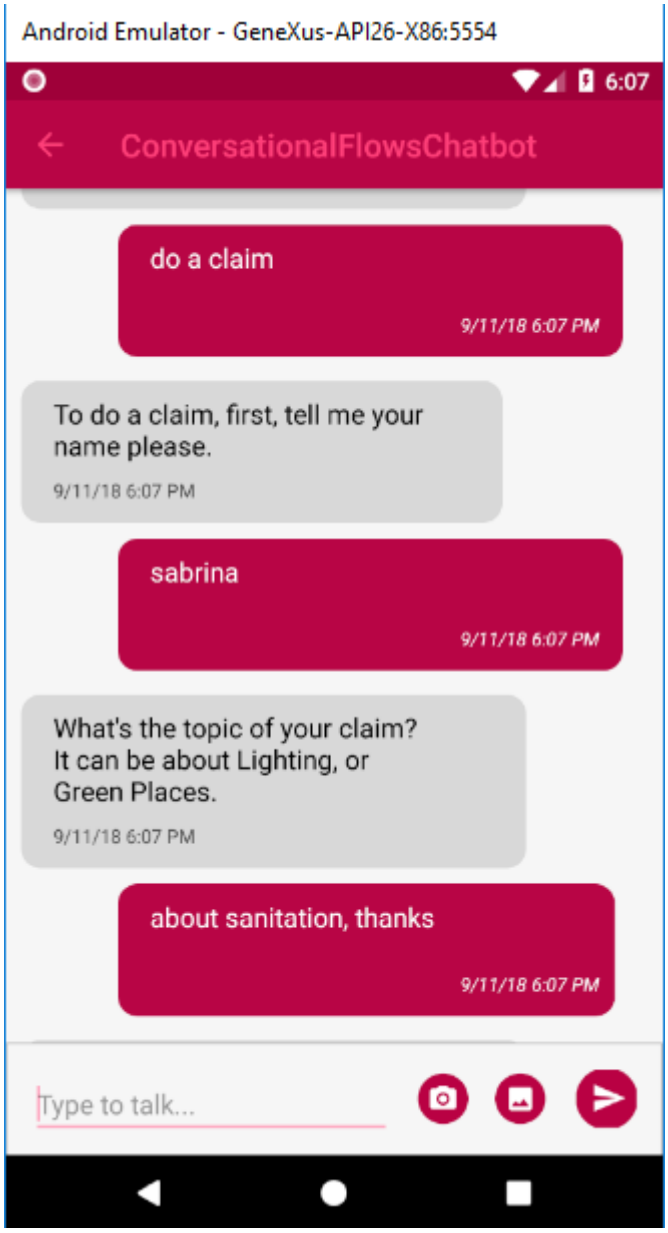


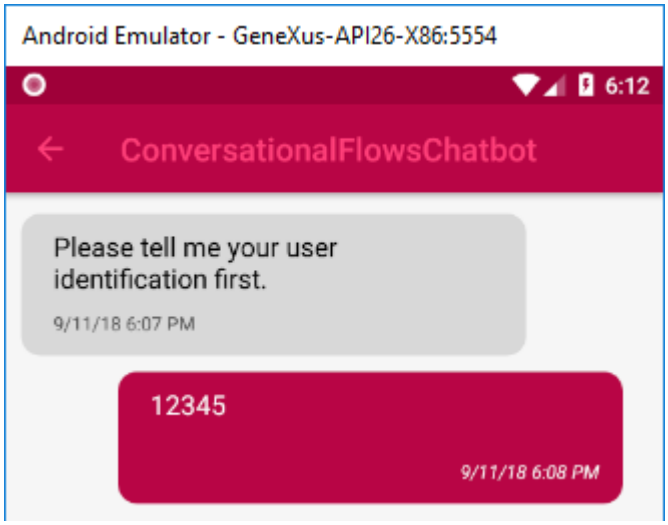
Observe que cada uno de los posibles valores de entrada redirige a otro Flow, para continuar con la conversación. Esto está dado por la propiedad Redirect to Flow, bajo un nodo Redirection.



Puede probar ejecutar cualquiera de ellos y ver cómo se desencadena el diálogo.

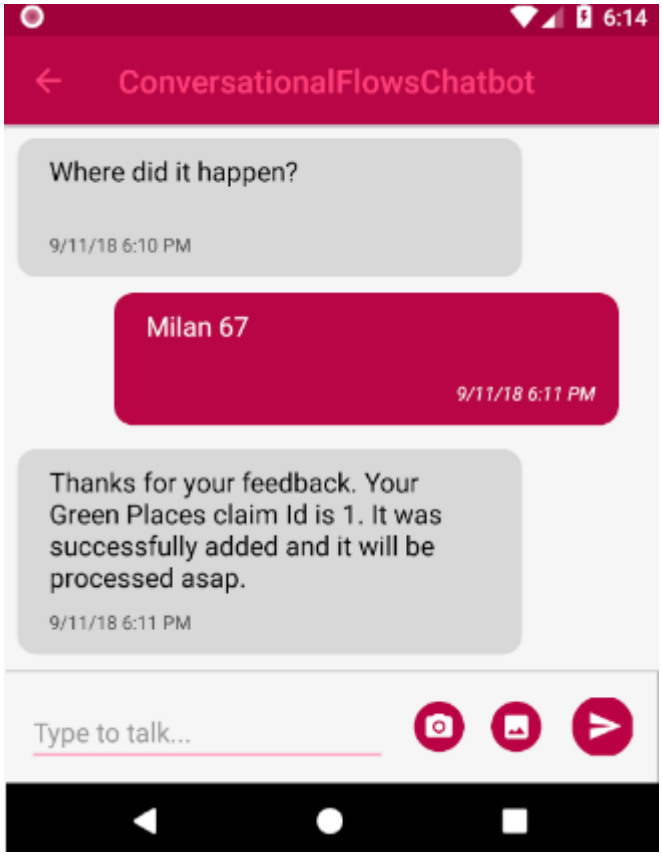
Por ejemplo, una conversación posible es esta:


Android Emulator - GeneXus-API26-X86:5554
6:07
ConversationalFlowsChatbot
do a claim
9/11/18 6:07 PM
To do a claim, first, tell me your name please.
9/11/18 6:07 PM
sabrina
9/11/18 6:07 PM
What's the topic of your claim? It can be about Lighting, or Green Places.
9/11/18 6:07 PM
about sanitation, thanks
9/11/18 6:07 PM
Type to talk...


Android Emulator - GeneXus-API26-X86:5554
6:12
ConversationalFlowsChatbot
Please tell me your user identification first.
9/11/18 6:07 PM
12345
9/11/18 6:08 PM

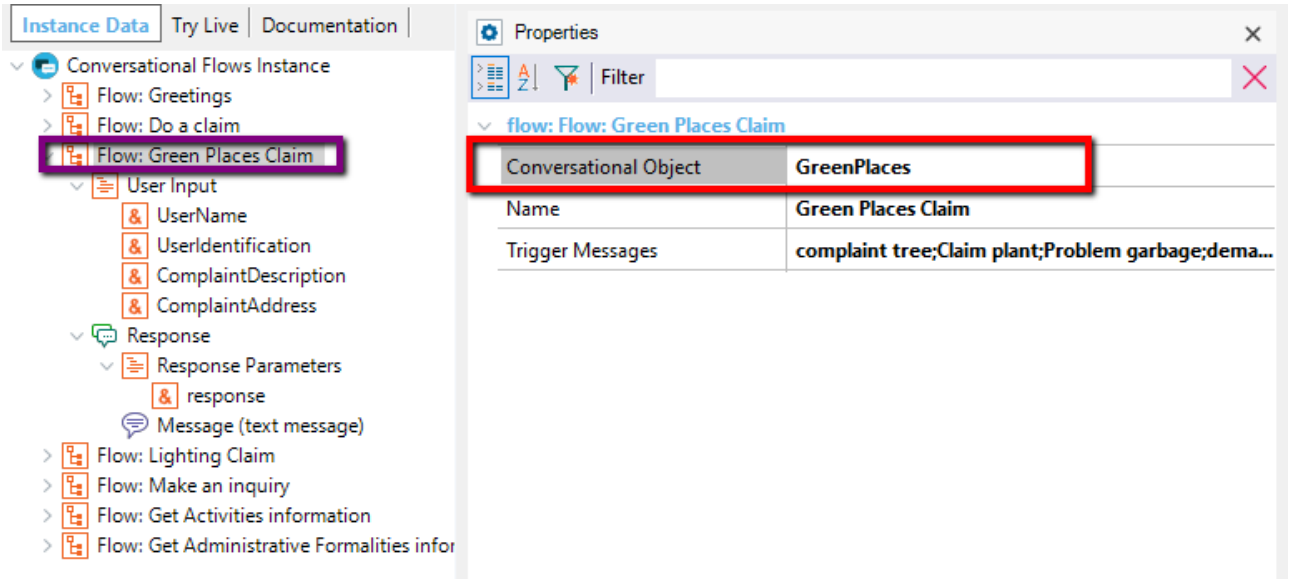
Nota: Puede buscar los usuarios de Citizen ejecutando el panel Home del backend.

Escriba por ejemplo “There is a broken pipe” y el bot le responderá:



Veamos el Flow “Green Places Claim”.

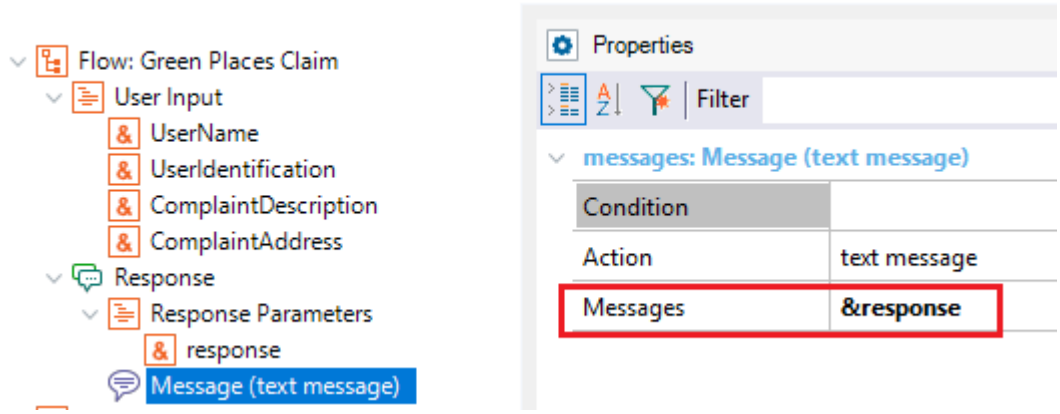
Es un flujo sencillo en donde una vez que se ingresaron los datos, el desenlace de la respuesta se produce a través de la ejecución de un procedimiento, que está dado en la propiedad “Conversational Object” del Flow. Se trata del objeto GreenPlaces.



Abra el objeto GreenPlaces, y observe que recibe como parámetro un subconjunto de los inputs solicitados al usuario y son usados para registrar la demanda del usuario.

```
parm(in:&UserIdentification, in:&ComplaintDescription,
in:&ComplaintAddress,out:&response);
```

El parámetro de salida, &response, puede verlo en la propiedad Message del nodo Response. Eso significa que la salida de este Flow es, simplemente, el response devuelto por el procedimiento GreenPlaces.



¡Así de sencillo!

Resumen: ¿Cómo se resuelve la respuesta? Cuando el usuario termina de ingresar los datos solicitados, se ejecuta el objeto dado en la propiedad Conversational Object.

Observe y pruebe:

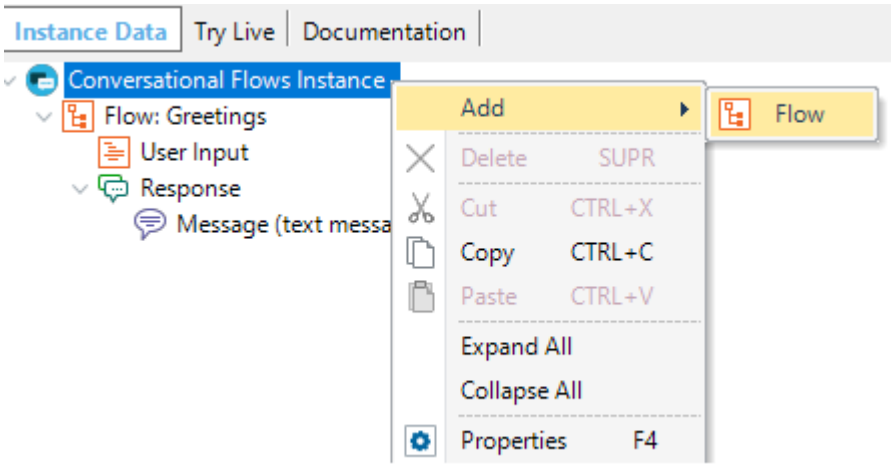
El usuario puede entrar directo al Flow “Green Places claim”, o cualquiera de los Flows de reclamo, es decir, no tiene por qué ser solicitado que ingrese el motivo del reclamo, sino que lo puede incluir directo en su consulta. Pruebe ingresar “I want to do a demand about a lighting issue”.

En la consulta puede incluir cualquiera de los inputs que mapean con una entidad, y no se le volverá a solicitar. Ejemplo “I’m user 12345 and I want to do a claim about missing lights on the street.”

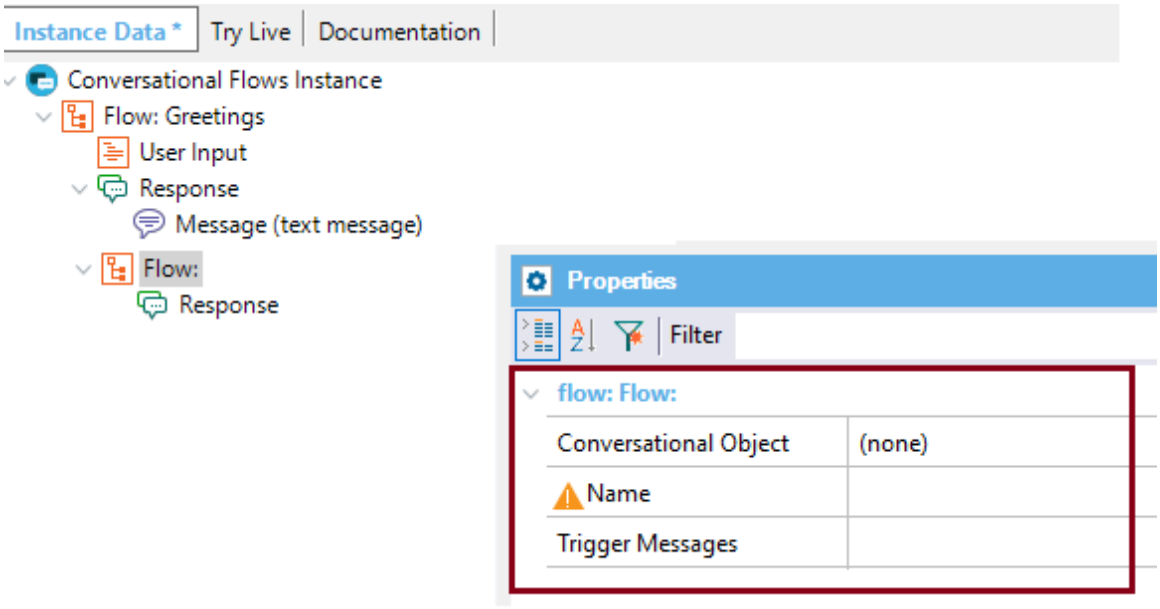
OPCIONAL: VER LAS ACTIVIDADES CULTURALES

En este punto vamos a agregar un nuevo intent a nuestro chatbot, con el fin de que el usuario pueda ver las actividades culturales donde puede participar.

Agregue un Flow llamado “Get Activities information”. Para eso, tiene que hacer botón derecho sobre el nodo padre de la instancia “Conversational Flows Instance”, y luego Add -> Flow.

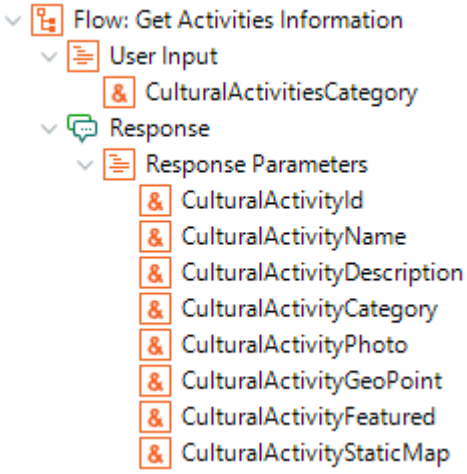


Allí tendrá para completar el Name del Flow (“Get Activities information”).



En la propiedad **Conversational Object**, configure el Data Provider "CulturalActivitiesNews".

Verá que automáticamente se completan el **User Input** y los **Response parameters**. Esto lo hace GeneXus a partir de la data del **Conversational Object** seleccionado.



Abra el Data Provider CulturalActivitiesNews y verá en qué consiste y los parámetros que recibe y retorna. Básicamente, recibe una categoría de actividad (Art, culture, nature) y devuelve las actividades según ese filtro.

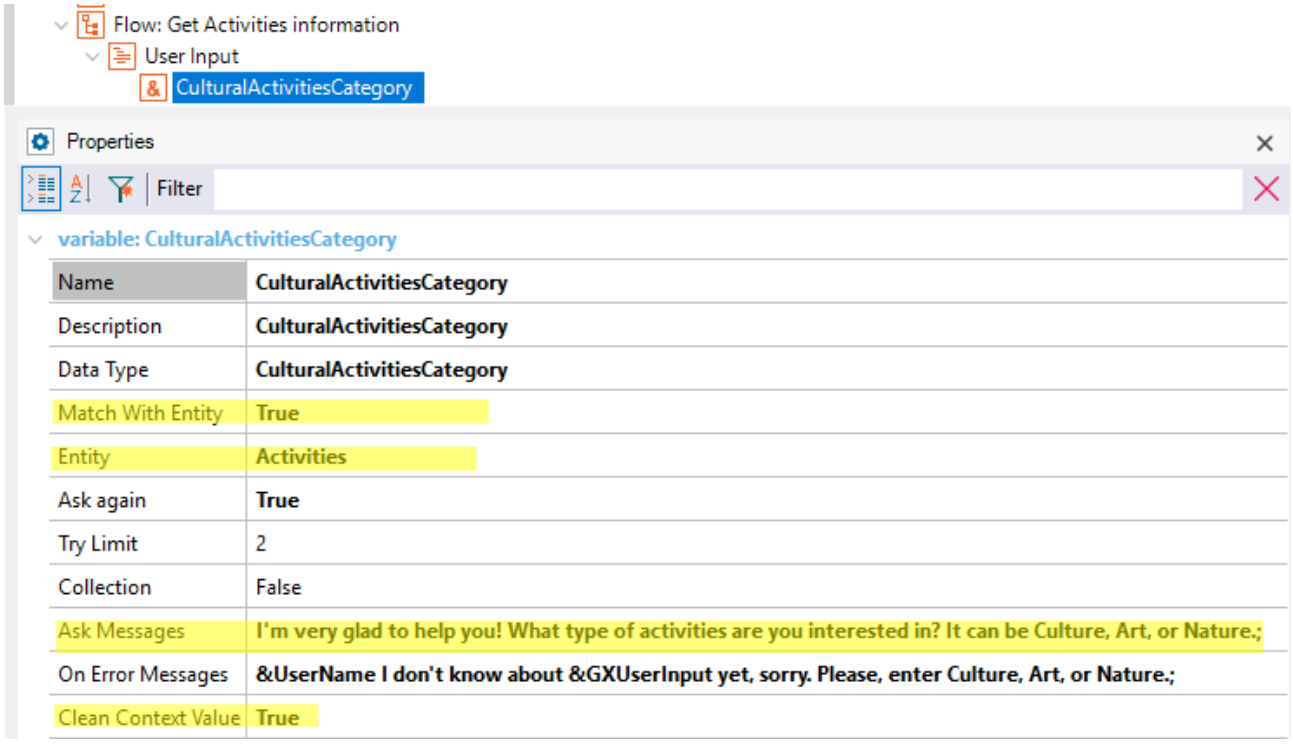
```
parm(in:&CulturalActivitiesCategory);

CulturalActivity
where CulturalActivityCategory = &CulturalActivitiesCategory
{
     CulturalActivityId = CulturalActivityId
```

```
        CulturalActivityName = CulturalActivityName
        CulturalActivityDescription = CulturalActivityDescription
        CulturalActivityCategory = CulturalActivityCategory
        CulturalActivityPhoto = CulturalActivityPhoto
}
```

Volviendo al Flow, seleccione el User Input "CulturalActivitiesCategory".

Configure las propiedades siguientes:

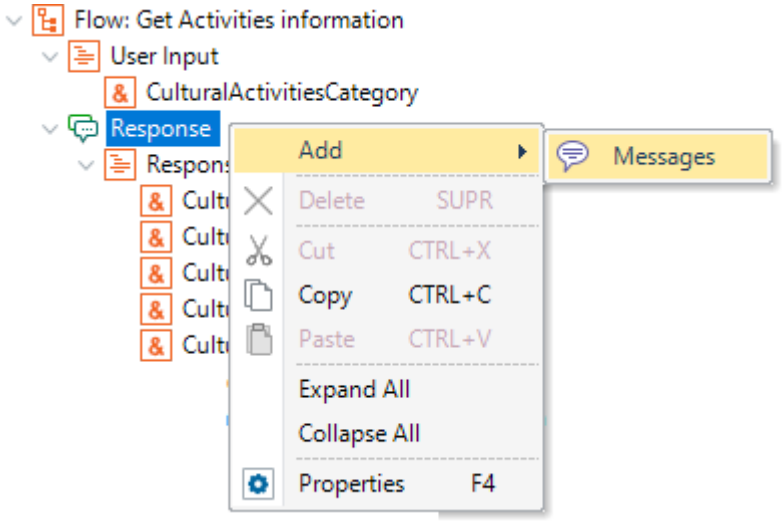
- Match with Entity = TRUE
- Entity = Activities
- Ask Messages = I'm very glad to help you! What type of activities are you interested in? It can be Culture, Art, or Nature.
- Clean Context Value = TRUE



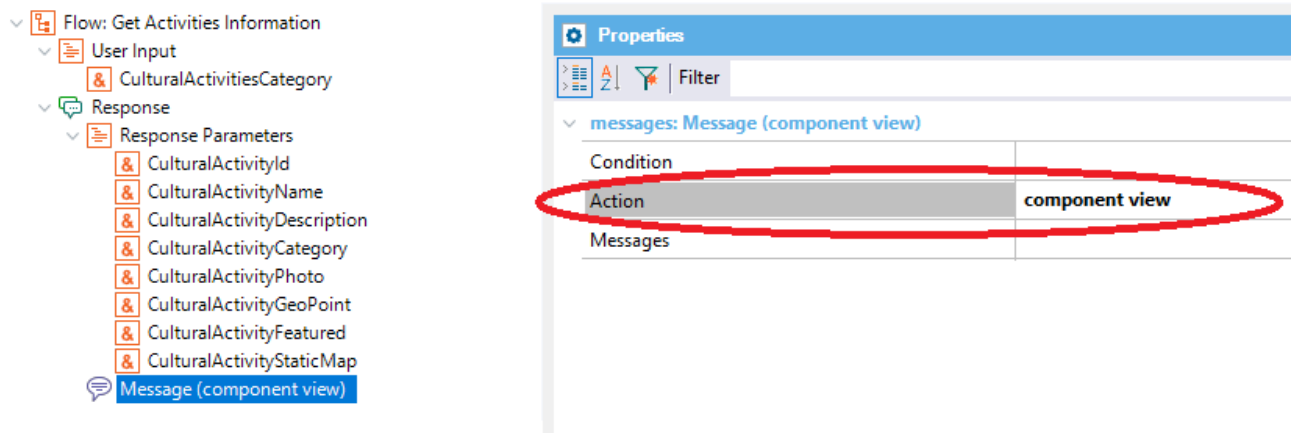
Como resultado de este Flow, vamos a mostrar al usuario un grid horizontal con las actividades.

Agregue un Message, cuya propiedad Action sea "Component View" de forma de que la salida de la respuesta sea un componente.

Para agregar el Message, debe hacer botón derecho en el nodo Response, y luego Add -> Messages.



En el nodo Message recién creado, edite la propiedad Action, y cámbiela al valor “component view”.

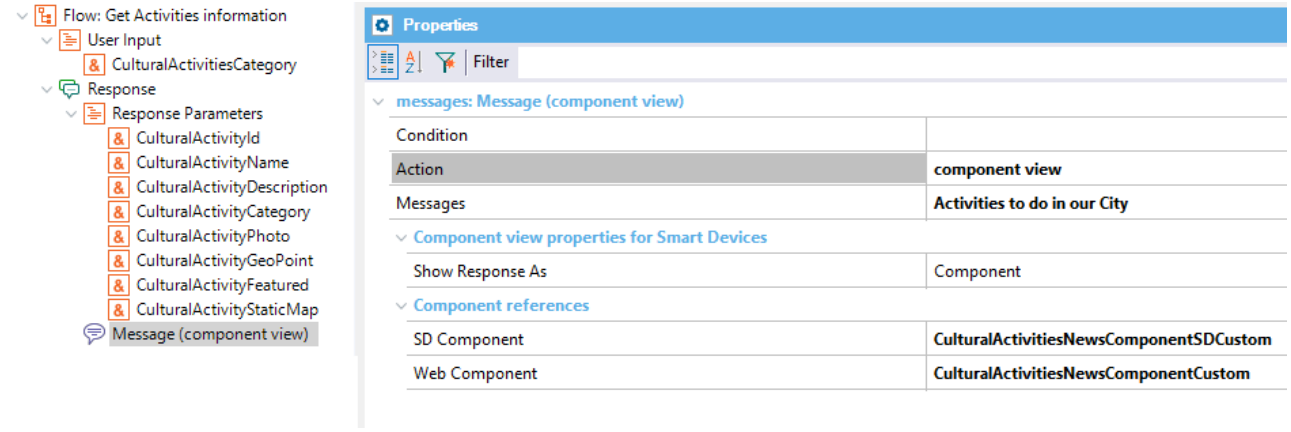


Edite la propiedad messages y escriba “Activities to do in our City”.

Ahora falta indicar cuál es el componente a mostrarse como resultado de este Flow, donde tendremos el horizontal grid que despliega los datos.

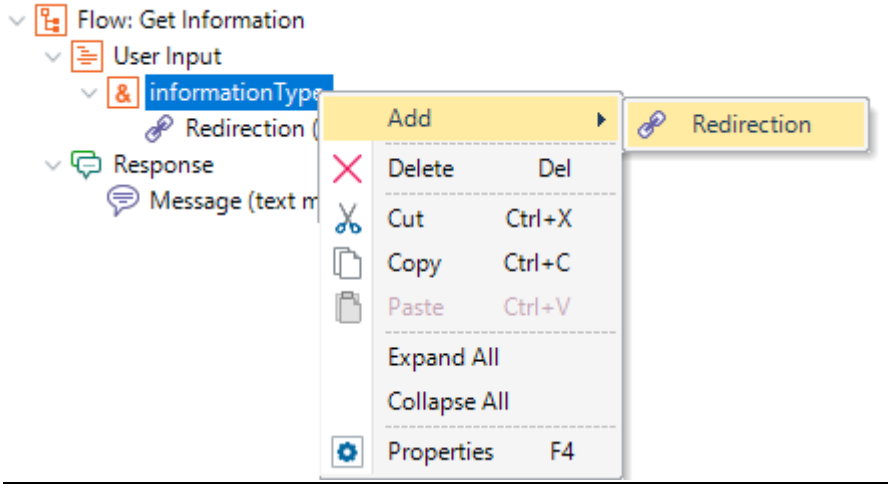
Podríamos usar el objeto generado automáticamente con este fin (indicado en la propiedad Generated web Component), pero usaremos el nuestro.

Configure la propiedad **Web Component** con el valor “CulturalActivitiesNewsComponentCustom”, y la propiedad SD components con el valor “CulturalActivitiesNewsComponentSDCustom” como se muestra en la figura:

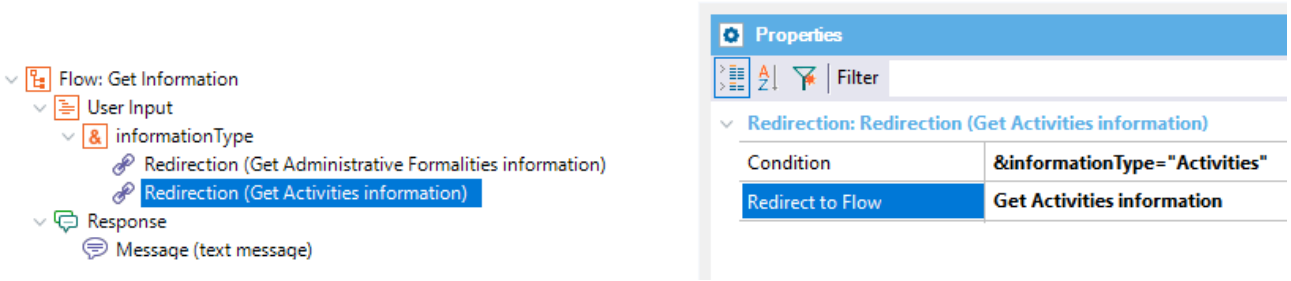


Para completar el ejercicio, editemos el Flow “Get Information”, y agreguemos un nodo Redirection que redirija al Flow que acabamos de crear.

Tiene que hacer botón derecho en el nodo informationType, del Flow “Get Information”, y luego Add -> Redirection.



Luego, complete lo siguiente (debe respetar el casing en lo ingresado en la Condition):



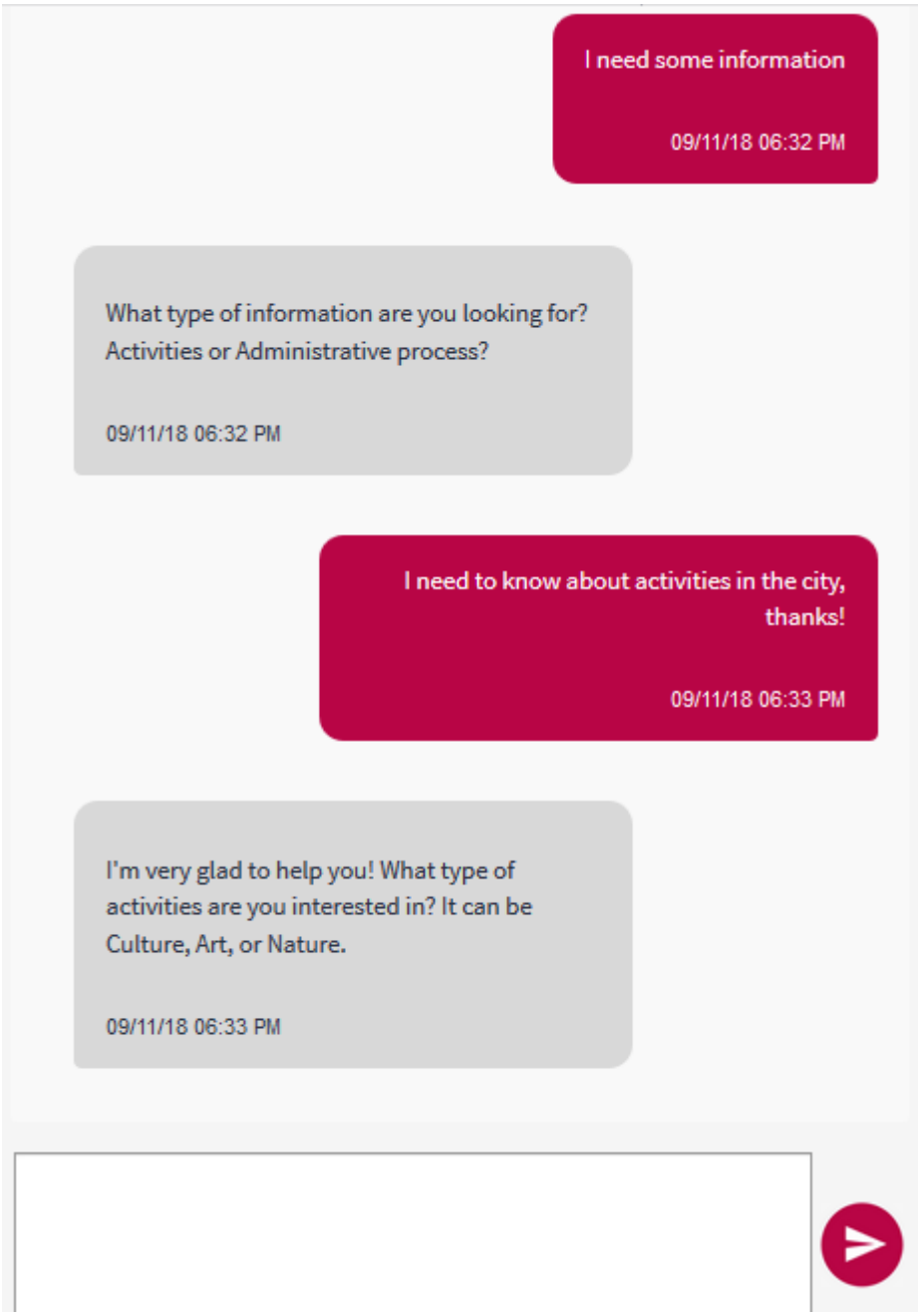
De esa forma, si el usuario entra por el Flow “Get Information”, podrá ir al Flow de “Get Activities Information” automáticamente, luego de que se le solicite la información que desea obtener, e indique que se trata de “Activities”.

¡Listo!

Para ejecutar debe hacer lo siguiente:

1. Botón derecho sobre la instancia Citizen -> Generate Chatbot
2. Run del webpanel CitizenChatbotWebUI y el panel CitizenChatbotSDUI para ver el resultado.

Nota: si no funciona con esto, como workaround modifique el objeto handler para que sea main y haga un rebuild sobre él. Luego quítele la propiedad main y vuelva a ejecutar el paso 2.

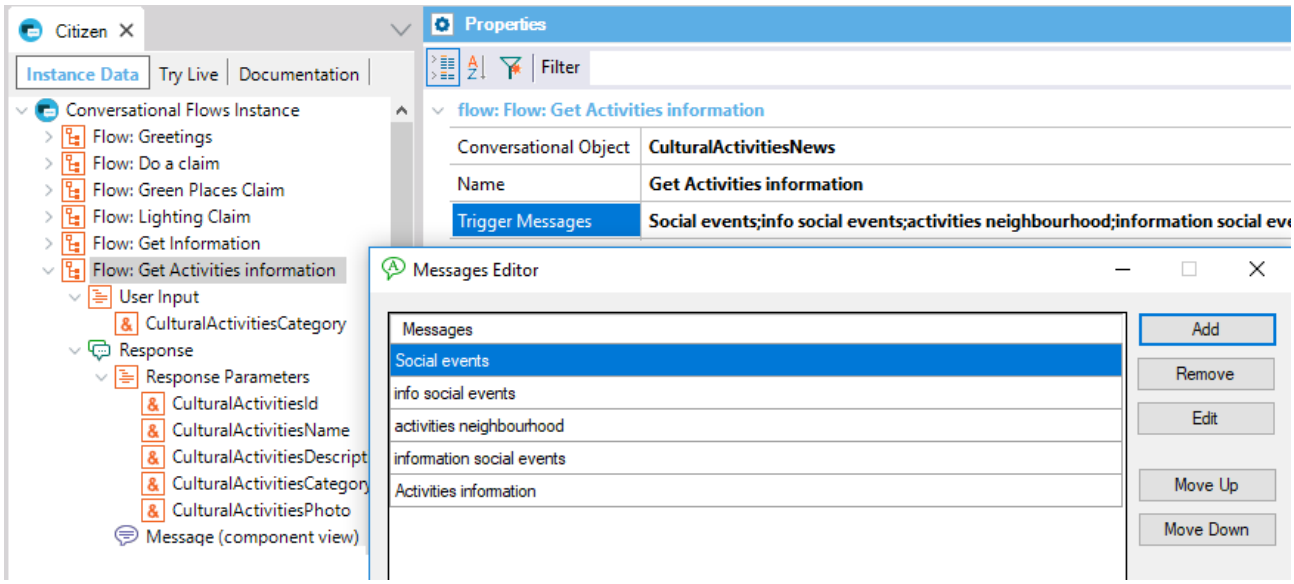


Elija alguna y verá el panel con el grid horizontal.

Mejor aún sería que se pudiera permitir que el usuario directamente indique acerca de qué temas quiere obtener información, y no tener que consultarle si no es necesario.

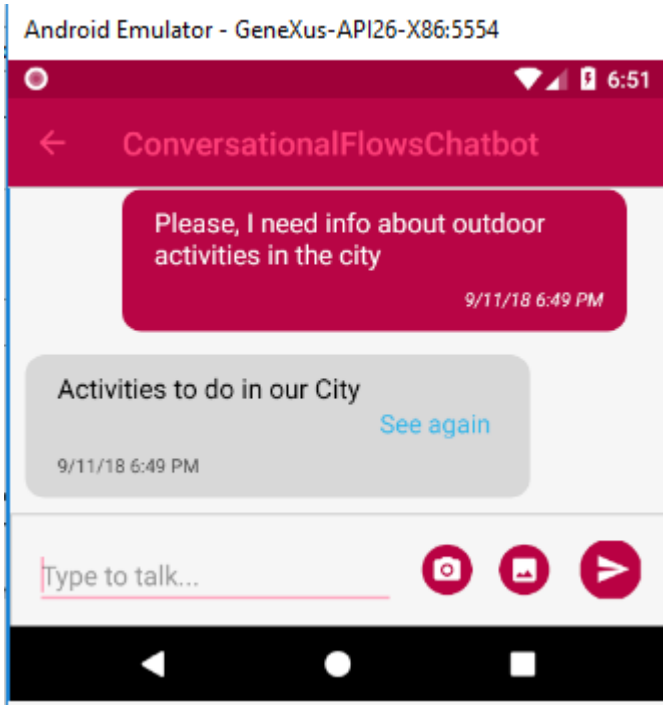
Eso significa entrenar al chatbot para permitir ingresar al Flow “Get Activities Information” de forma directa. Agreguemos algún Trigger Message (el que le parezca adecuado), para que Watson aprenda cómo puede ingresar el usuario a este flujo directamente, sin necesidad de venir desde otro.

Por ejemplo:



Lo puede probar ingresando la consulta “I need information about the outdoor activities!” (es un ejemplo, tiene libertad de usar cualquier consulta semánticamente equivalente).

Observe que en esa consulta agregamos también el tipo de actividad, que es “outdoor”.



ANEXO: TRY LIVE

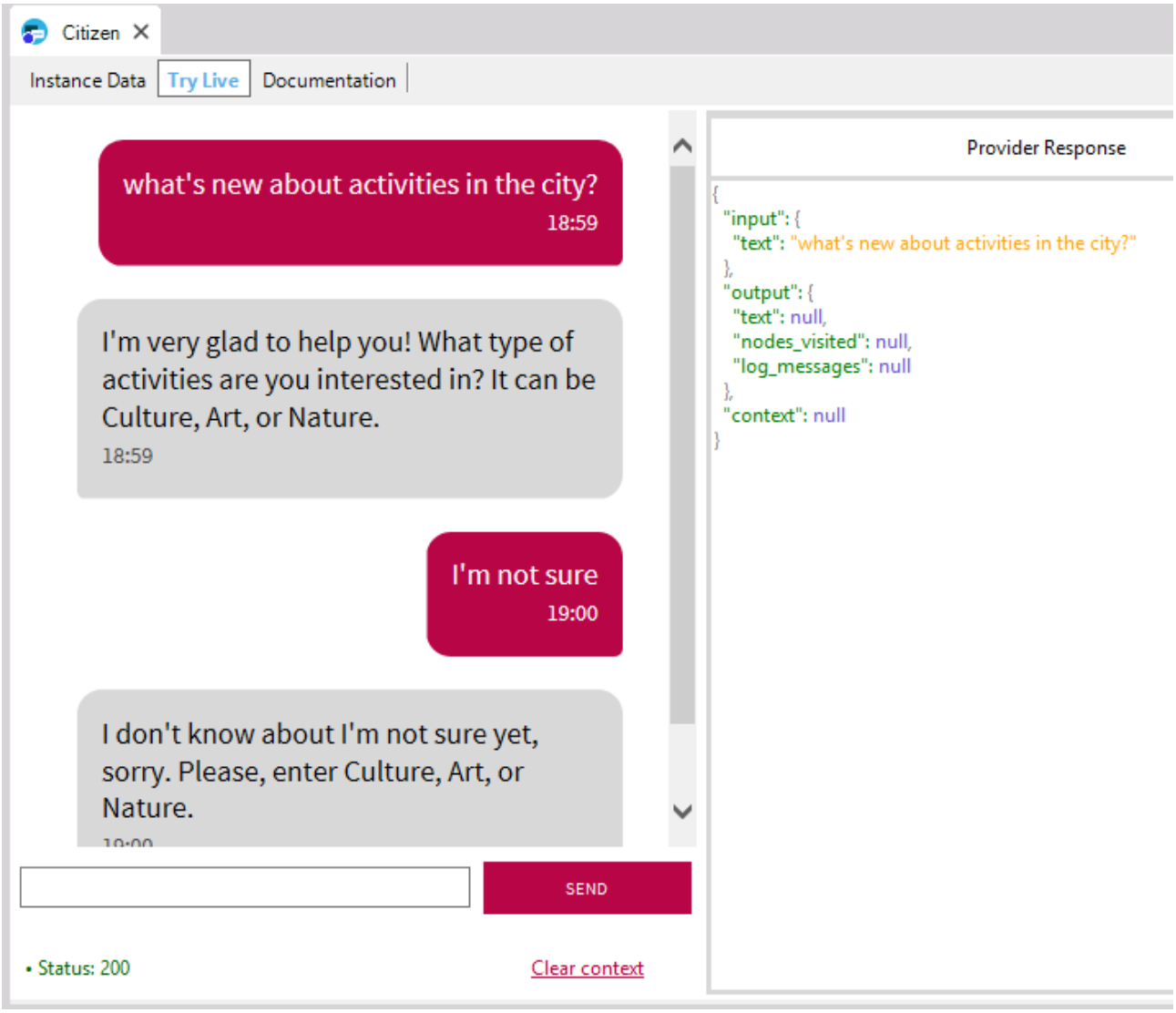
Este apartado es opcional.

Dentro de GeneXus, puede probar el diálogo de su chatbot usando la herramienta Try Live.

Try Live es un tab al lado del tab Instance Data, donde puede probar cómo responde directamente el NLP provider.

Puede probar cualquier consulta dentro de lo que hemos estado trabajando en nuestro chatbot.

Por ejemplo:



Nota:

El Try live nos permite probar el reconocimiento de los intents, y ejecución del flujo, mientras que no haya necesidad de llamar a un objeto GeneXus. Cuando es así, el resultado que veremos en pantalla es la variable de salida del procedimiento.

Recuerde: Que los chatbots se deben entrenar para ver un funcionamiento óptimo en cuanto a todas las posibilidades de consulta que realicen los usuarios.

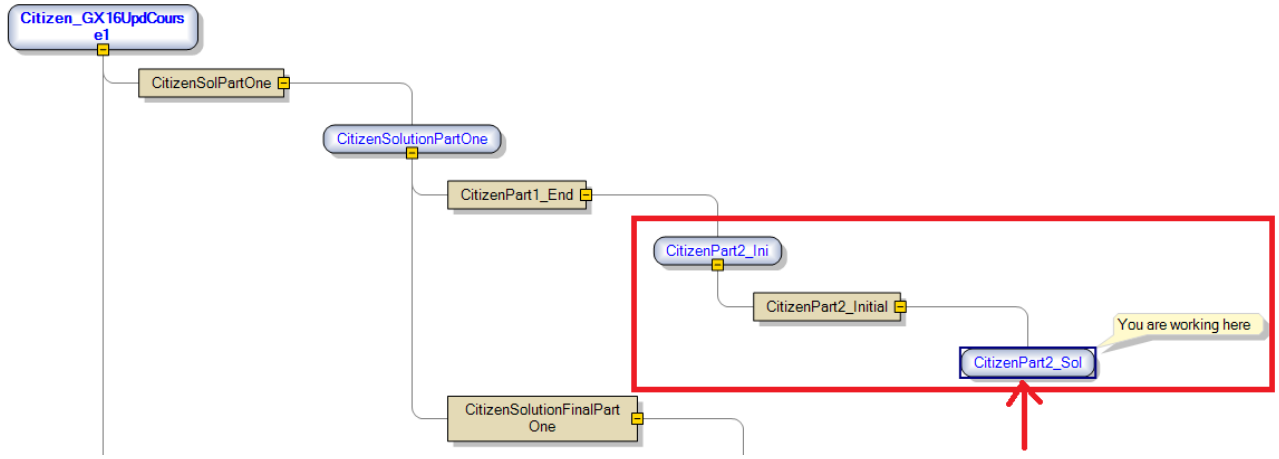
Entrenarlo significa agregar Trigger messages a los intents, para que éstos se reconozcan, reduciendo la posibilidad de ambigüedad.

DOCUMENTACIÓN

[Chatbots in GeneXus]
[Chatbot Generator]
[Chatbots Architecture]
[How to build a Chatbot using GeneXus]
[Chatbot Generator Try Live]
[IBM Watson Setup]

KB SOLUCIÓN

Podrá descargar de GeneXus Server la KB solución de este práctico para comparar resultados. Es la versión de la KB de nombre CitizenPart2_Sol.



Le hemos quitado el UserName y UserPassword. Configure los valores apropiados para poder ejecutar.