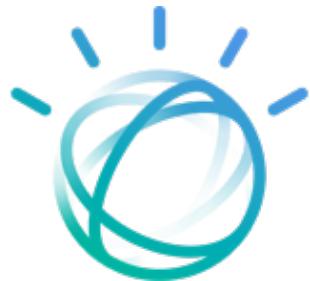


# Breaking Cloud

## Cognitive

Lab 3: Entrena tu primer chatbot



## Tabla de contenidos

---

<b>Introducción al laboratorio .....</b>	3
1- Creamos el servicio de Watson Conversation .....	4
2- Definiendo la conversación en lenguaje natural.....	6
3- Tu chatbot en telegram .....	24



## Introducción al laboratorio

En este laboratorio, adquirirás los conocimientos necesarios para utilizar Watson AI para crear un chatbot en lenguaje natural. Para ello, utilizaremos el servicio de **Watson Conversation** para crear las intenciones, entidades y el flujo de la conversación. Además, utilizaremos Node-RED para crear nuestro chatbot en telegram.

### Requisitos:

- Tener cuenta de IBM Cloud
- Acceso a Internet
- Cuenta en telegram
- Descargar el contenido del laboratorio de  
<https://github.com/mariaborbones/BreakingCloudCognitive>

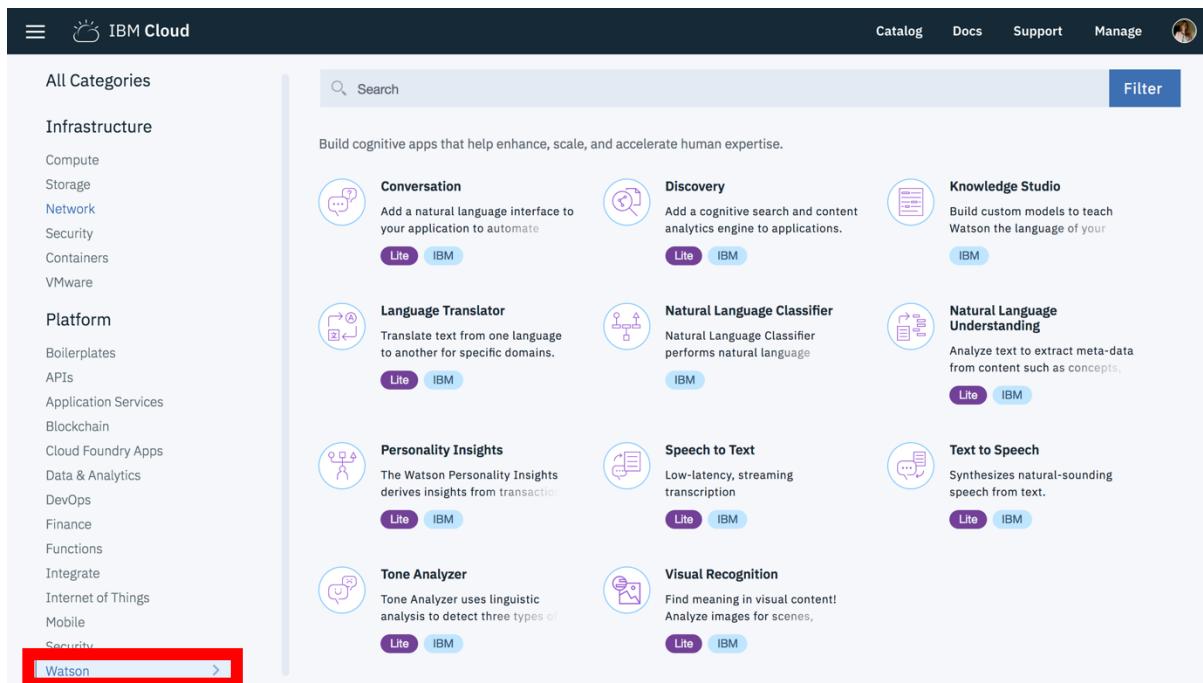
## 1- Creamos el servicio de Watson Conversation

Para crear el servicio de Watson Conversation, necesitamos acceder al catálogo de **IBM Cloud** desde la siguiente URL: [bluemix.net](https://bluemix.net)

Una vez hemos accedido a **IBM Cloud**, hacemos click en catálogo (barra superior) como se muestra en la imagen:

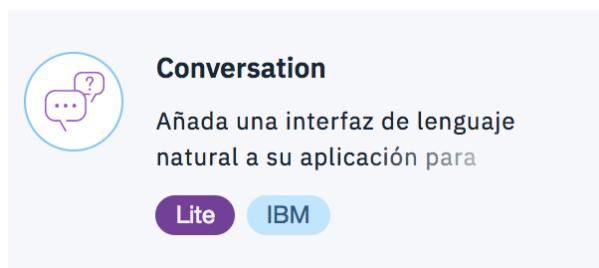


En el menú de la izquierda, donde se muestran todos los servicios de IBM Cloud, buscamos la categoría **Watson** debajo de **Plataforma** como se muestra en la imagen y hacemos click en ella:



Service	Description	Options
Conversation	Add a natural language interface to your application to automate	Lite, IBM
Discovery	Add a cognitive search and content analytics engine to applications.	Lite, IBM
Knowledge Studio	Build custom models to teach Watson the language of your	IBM
Language Translator	Translate text from one language to another for specific domains.	Lite, IBM
Natural Language Classifier	Natural Language Classifier performs natural language	IBM
Natural Language Understanding	Analyze text to extract meta-data from content such as concepts.	Lite, IBM
Personality Insights	The Watson Personality Insights derives insights from transaction	Lite, IBM
Speech to Text	Low-latency, streaming transcription	Lite, IBM
Text to Speech	Synthesizes natural-sounding speech from text.	Lite, IBM
Tone Analyzer	Tone Analyzer uses linguistic analysis to detect three types of	Lite, IBM
Visual Recognition	Find meaning in visual content! Analyze images for scenes,	Lite, IBM

En este caso, entre todo el conjunto de servicios, vamos a elegir desplegar Conversation, así que lo buscamos y hacemos click sobre el mismo:



**Conversation**  
Añada una interfaz de lenguaje natural a su aplicación para

Lite IBM

En la siguiente pantalla, debemos asignarle un nombre al servicio (por ejemplo: lab3), una región (recomendable Alemania), nuestra organización y el espacio de trabajo donde queremos desplegarlo. Hacemos click en **crear**.

IBM Cloud Catalog Docs Support Manage

Ver todo Conversation

Añada una interfaz de lenguaje natural a su aplicación para automatizar interacciones con los usuarios finales. Las aplicaciones comunes incluyen agentes virtuales y bots de conversación que se pueden integrar y comunicar en cualquier canal o dispositivo. Prepare el servicio Watson Conversation a través de una aplicación web fácil de utilizar, diseñado para que pueda crear rápidamente flujos de conversación natural entre las aplicaciones y los usuarios, y desplegar soluciones escalables y económicas.

Nombre del servicio: Conversation-lab3

Seleccione una región/ubicación de despliegue: Alemania

Elija una organización: maria.borbones@es.ibm.com

Elija un espacio: cognitive

Imágenes

Pulse una imagen para aumentar y ver capturas de pantalla, diapositivas o videos. Las capturas de pantalla muestran la interfaz de usuario para el servicio una vez que se haya suministrado.

Lite IBM Ver documentos

AUTOR IBM PUBLICADO 02/03/2018

¿Necesita ayuda? Póngase en contacto con el equipo de ventas de IBM Cloud

Estimar coste mensual Calculadora de costes

Crear

*¡Enhorabuena! Has completado la primera parte del laboratorio. Ya sabes cómo desplegar un servicio de Watson en IBM Cloud.*

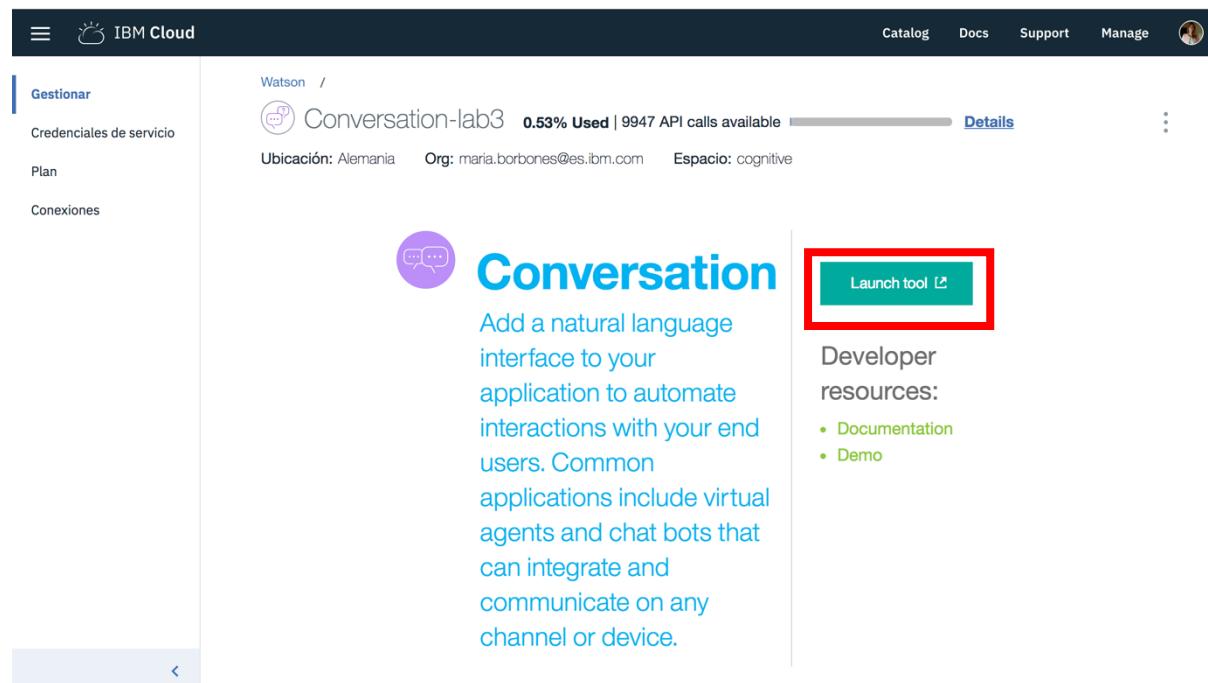
## 2- Definiendo la conversación en lenguaje natural

**Watson Conversation** permite añadir una interfaz en lenguaje natural a las aplicaciones para automatizar interacciones con los usuarios finales.

El uso más común es en agentes virtuales y bots de conversación que se pueden integrar y comunicar con cualquier canal o dispositivo.

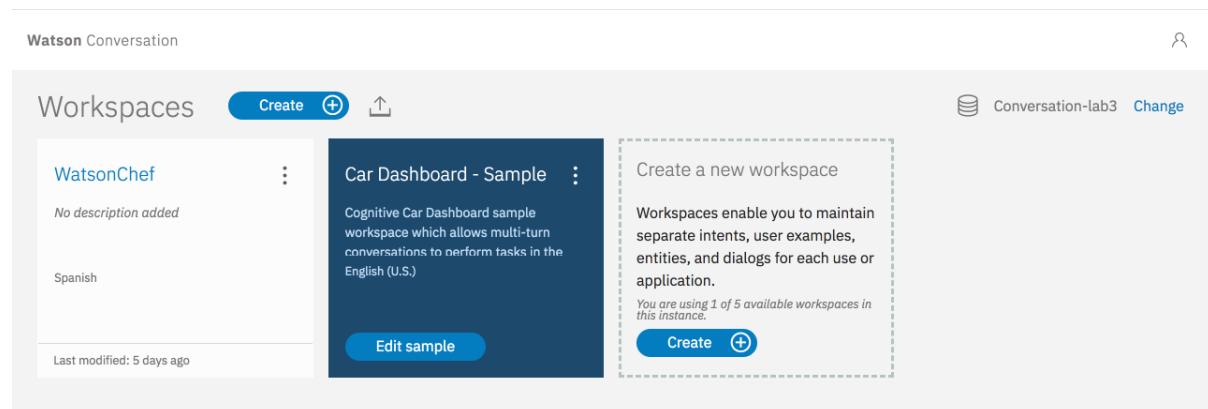
En este laboratorio vamos a darle voz a chefito, un chabot de recetas que nos va a aconsejar qué receta cocinar. Además, este laboratorio nos va a servir para entender cómo funciona el entrenamiento de Watson Conversation y darnos cuenta que es muy importante involucrar a un SME en el diseño de la conversación

Primero, debemos acceder a la interfaz de usuario de conversation. Para ello, hacemos click en **Launch tool** como se muestra en la imagen:



The screenshot shows the IBM Cloud interface for the Watson Conversation service. On the left, there's a sidebar with options like 'Gestionar', 'Credenciales de servicio', 'Plan', and 'Conexiones'. The main area displays a summary for 'Conversation-lab3': '0.53% Used | 9947 API calls available', 'Ubicación: Alemania', 'Org: maria.borbones@es.ibm.com', and 'Espacio: cognitive'. Below this, there's a large central box with the title 'Conversation' and a description: 'Add a natural language interface to your application to automate interactions with your end users. Common applications include virtual agents and chat bots that can integrate and communicate on any channel or device.' To the right of this box is a 'Launch tool' button, which is highlighted with a red rectangle. Further down, there's a section titled 'Developer resources:' with links to 'Documentation' and 'Demo'.

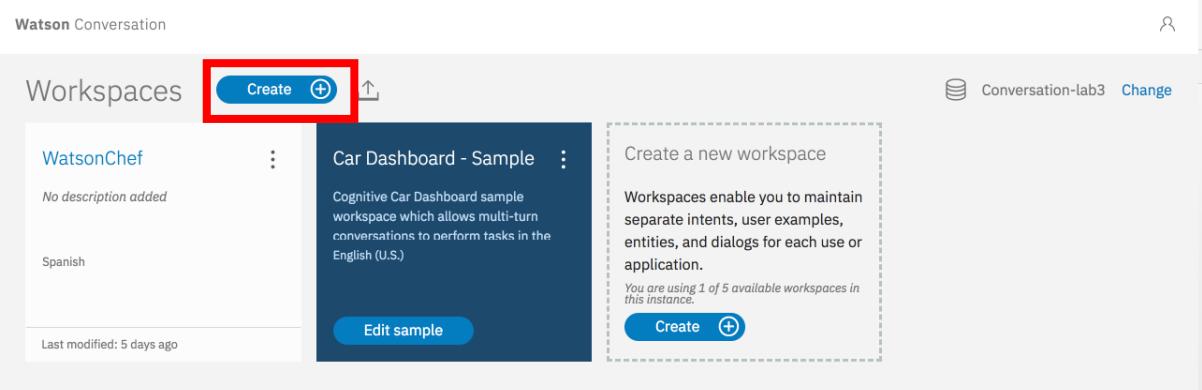
Y accedemos a la consola de Watson Conversation



The screenshot shows the Watson Conversation workspace management interface. At the top, it says 'Watson Conversation'. Below that, there's a 'Workspaces' section with a 'Create' button and a plus sign. It lists two workspaces: 'WatsonChef' (with a description 'No description added') and 'Car Dashboard - Sample' (with a description 'Cognitive Car Dashboard sample workspace which allows multi-turn conversations to perform tasks in the English (U.S.)'). The 'Car Dashboard - Sample' workspace has an 'Edit sample' button. To the right, there's a panel for creating a new workspace, which includes a 'Create' button with a plus sign. The panel also states 'You are using 1 of 5 available workspaces in this instance.'

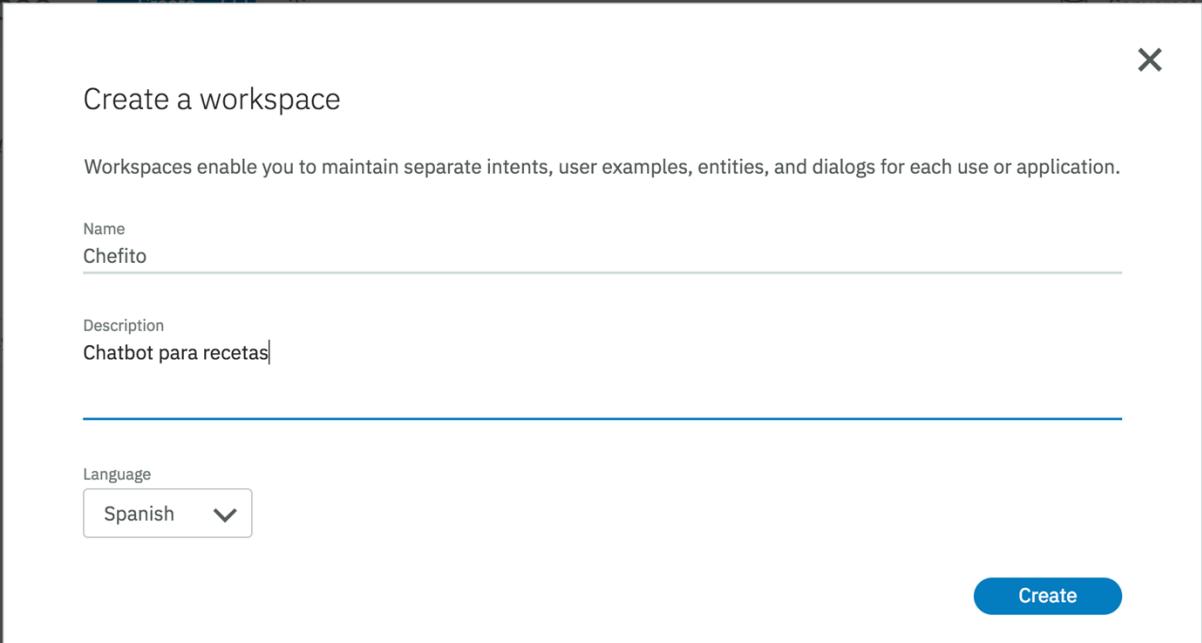
Lo primero que necesitamos es crear un workspace o espacio de trabajo. Cada workspace se define para una conversación independiente y para un idioma específico.

Para crear nuestra primera conversación hacemos clic en **create +**



The screenshot shows the 'Watson Conversation' interface. At the top, there's a navigation bar with a user icon. Below it, the title 'Workspaces' is displayed, followed by a 'Create +' button with a plus sign, which is highlighted with a red box. To the right of the button, there's a small upward arrow icon. On the far right of the header, there are icons for 'Conversation-lab3' and 'Change'. The main area lists workspaces: 'WatsonChef' (No description added, Spanish, last modified 5 days ago) and 'Car Dashboard - Sample' (Cognitive Car Dashboard sample workspace which allows multi-turn conversations to perform tasks in the English (U.S.)). Below these, a dashed box contains instructions: 'Create a new workspace', 'Workspaces enable you to maintain separate intents, user examples, entities, and dialogs for each use or application.', and a note: 'You are using 1 of 5 available workspaces in this instance.' At the bottom of this section is another 'Create +' button.

Y elegimos el nombre para nuestro workspace (por ejemplo: chefito), de manera opcional le podemos asignar una descripción y elegimos el idioma con el que vamos a trabajar, en nuestro caso español



The screenshot shows a modal dialog titled 'Create a workspace'. The text inside says: 'Workspaces enable you to maintain separate intents, user examples, entities, and dialogs for each use or application.' Below this, there are input fields: 'Name' (set to 'Chefito'), 'Description' (set to 'Chatbot para recetas'), and a 'Language' dropdown menu set to 'Spanish'. At the bottom right of the dialog is a large blue 'Create' button.

Ahora ya tenemos nuestro workspace y podemos empezar a trabajar. Pero antes, vamos a introducirnos algunos conceptos para que entendáis como funciona la definición de una conversación en lenguaje natural con **Watson Conversation**

## Cuando nos referimos a intenciones ...



### Definición

Una intención (intent) es un objetivo expresado como una entrada de un usuario, tal como, responder a una pregunta o procesar el pago de una factura.

Watson Conversations reconoce la intención y elige el flujo correcto en la conversación.



### Planificar

Necesitamos saber para planificar las intenciones qué es lo que el cliente/usuario quiere hacer y qué debe gestionar la aplicación.

Las intenciones que se identifiquen para la aplicación determinarán el flujo que necesitaremos crear para el diálogo. También determinará a qué sistemas necesitaremos conectarnos



### Identificar

- Recoge todas las posibles preguntas de usuarios reales.
- Ten en cuenta que se puede formular la misma pregunta de diferentes maneras:
  - Dime la previsión meteorológica
  - ¿Está lloviendo?
  - ¿Qué tiempo hace?
- Ordena todos tus ejemplos en categorías basándote en las capacidades que quieras que tenga tu aplicación.
- Actualiza tus intenciones y ejemplos todas las veces que necesites

## Cuando nos referimos a entidades...



### Definición

Una entidad (entity) representa una clase de objeto o tipo de dato que es relevante para el propósito del usuario.

**Watson conversations** reconoce las entidades mencionadas en la conversación con el usuario y es capaz de ejecutar la acción correcta.



### Planificar

Una entidad provee un contexto específico para una intención determinada.

Si la intención representa un verbo (algo que el usuario quiere hacer), la entidad representa un nombre (el objeto o contexto para la acción).



### Ejemplos

- Imagina que tu aplicación es un panel de control de un coche cognitivo que permite encender y apagar los accesorios. Considera las siguientes enunciados:
  - Apaga los faros del coche
  - Apaga la radio
  - Para el aire acondicionado
- Todas estas peticiones representan la misma intención (apagar algo) que se puede representar por #apagar
- Ahora necesitamos representar las entidades que queremos apagar
  - Faros
  - Radio
  - Aire acondicionado
- Para cada valor, además se pueden especificar sinónimos. Por ejemplo, podemos usar luces en vez de faros.

Y por último utilizamos las entidades e intenciones para definir el diálogo



El diálogo utiliza las entidades y las intenciones identificadas en la conversación con el usuario para obtener la información necesaria y proveer la mejor respuesta.

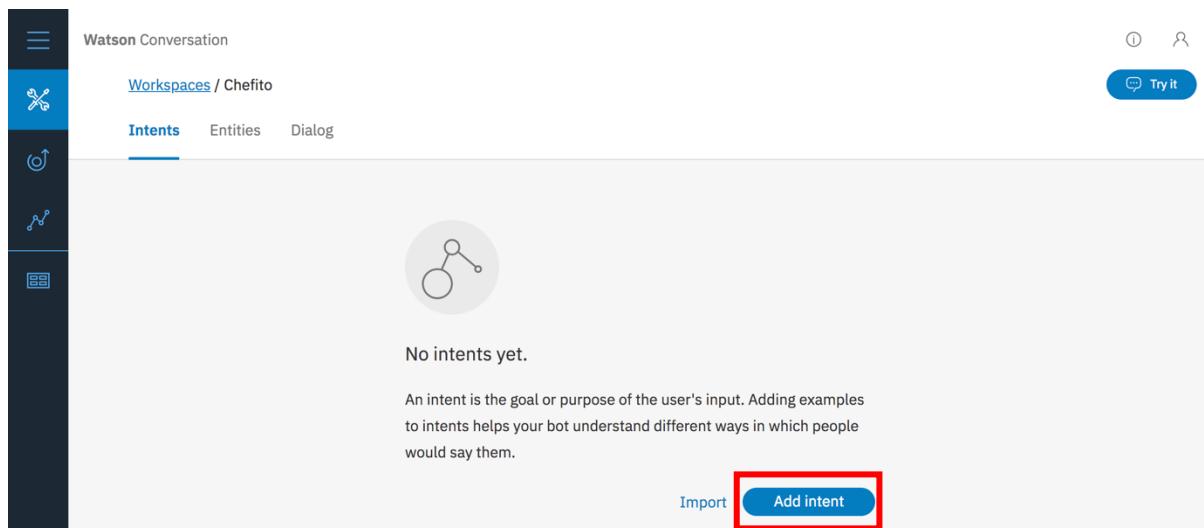
El diálogo se representa gráficamente como un árbol y cada una de las ramas procesa una intención de las previamente definidas



#### ¿Cómo funciona?

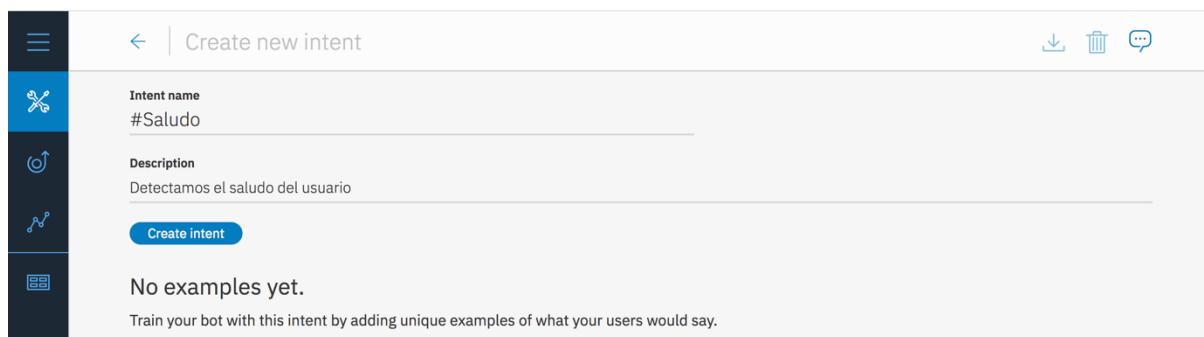
- Elige la primera intención desde la cuál quieres crear una rama del diálogo.
- Determina la respuesta que quieras proveer para esta intención. Puede ser una respuesta simple o una llamada a backend.
- Identifica la información que necesitas para poder dar la respuesta correcta. Por ejemplo, si quieras dar información sobre el pronóstico del tiempo, necesitarás saber la localización y la fecha.
- Define la estructura de la información de contexto.
- Construye la secuencia de diálogo correcta para que capture la información que necesitas. Piensa las posibles entradas de los usuarios e identifica si necesitas información adicional.
- Prueba tu diálogo con diferentes entradas

Lo primero que vamos a hacer es crear las intenciones. Vamos a crear primero la intención de Saludo, así que hacemos click **Add intent**



The screenshot shows the Watson Conversation interface. On the left is a sidebar with icons for Workspaces, Intents, Entities, and Dialog. The main area has tabs for 'Workspaces / Chefito', 'Intents' (which is active), 'Entities', and 'Dialog'. Below the tabs is a circular icon with a tree-like structure. The text 'No intents yet.' is displayed. A descriptive paragraph explains what an intent is: 'An intent is the goal or purpose of the user's input. Adding examples to intents helps your bot understand different ways in which people would say them.' At the bottom right, there are 'Import' and 'Add intent' buttons, with 'Add intent' being highlighted by a red box.

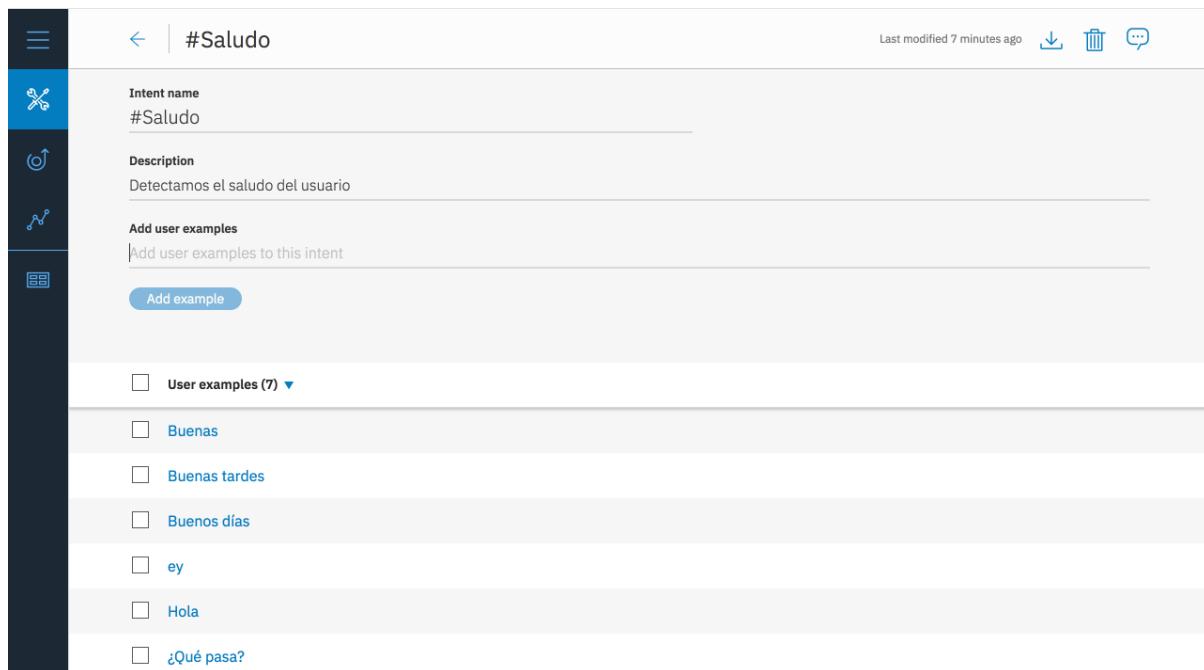
La vamos a llamar **Saludo** y podemos añadir una descripción de forma opcional. Le damos click a **Create intent**



The screenshot shows the 'Create new intent' dialog. On the left is a sidebar with icons for Workspaces, Intents, Entities, and Dialog. The main area has a back arrow and the text 'Create new intent'. Below is a form with 'Intent name' set to '#Saludo' and 'Description' set to 'Detectamos el saludo del usuario'. At the bottom is a 'Create intent' button. Below the form, it says 'No examples yet.' and 'Train your bot with this intent by adding unique examples of what your users would say.'

Y vamos a darle ejemplos a Watson de oraciones que debe detectar como el saludo del usuario como por ejemplo: (hacemos click en **Add example** para añadir los ejemplos)

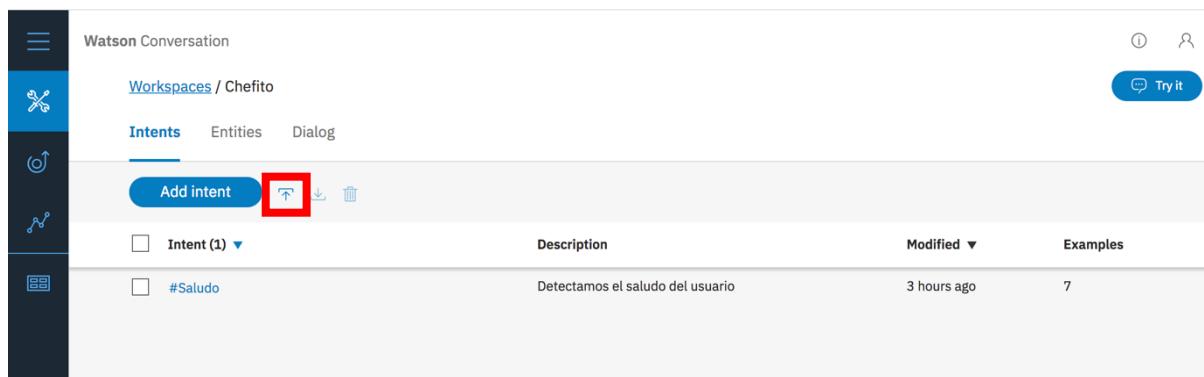
- Buenas
- Buenas tardes
- Buenos días
- Ey
- Hola
- ¿Qué pasa?
- ¿Qué tal?



The screenshot shows the Watson Assistant interface. On the left is a sidebar with icons for intents, entities, dialog, and workspace. The main area is titled '#Saludo'. It shows the intent name '#Saludo' and a description 'Detectamos el saludo del usuario'. Below this, there's a section for 'Add user examples' with a button 'Add example'. A list of user examples is shown, each with a checkbox:
 

- [User examples \(7\) ▾](#)
- [Buenas](#)
- [Buenas tardes](#)
- [Buenos días](#)
- [ey](#)
- [Hola](#)
- [¿Qué pasa?](#)

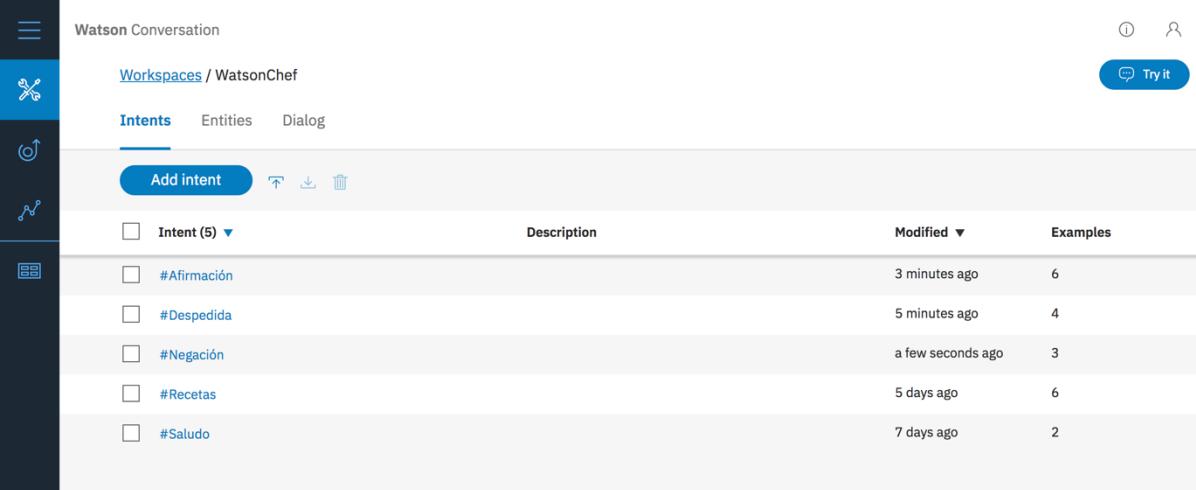
Para salir simplemente hacemos click en la flecha  y se guardan los cambios de manera automática. Ahora vamos a importar el resto de intenciones, así que hacemos click en el icono de importar que se muestra en la imagen:



Intent (1) ▾	Description	Modified ▾	Examples
<a href="#">#Saludo</a>	Detectamos el saludo del usuario	3 hours ago	7

Y accedemos al directorio que hemos descomprimido en nuestro escritorio en laboratorio 2, a la carpeta lab3 en concreto al directorio workspace. Seleccionamos el fichero intents.csv y hacemos click en **Import**

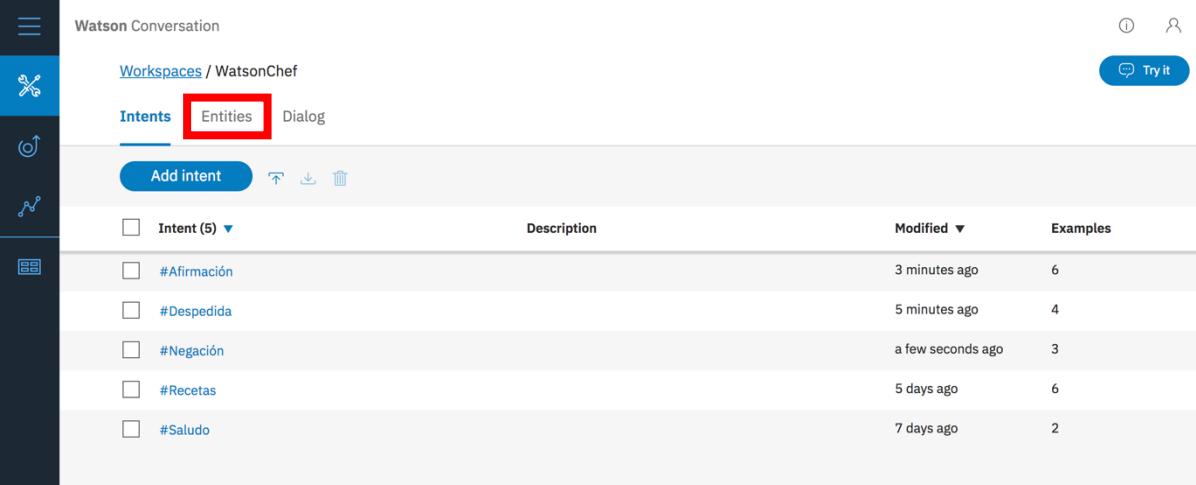
Una vez importados nos aparecen el resto de intenciones que podremos utilizar en el flujo de nuestra conversación:



The screenshot shows the Watson Conversation interface with the 'Intents' tab selected. There are five intents listed:

Intent	Description	Modified	Examples
#Afirmación		3 minutes ago	6
#Despedida		5 minutes ago	4
#Negación		a few seconds ago	3
#Recetas		5 days ago	6
#Saludo		7 days ago	2

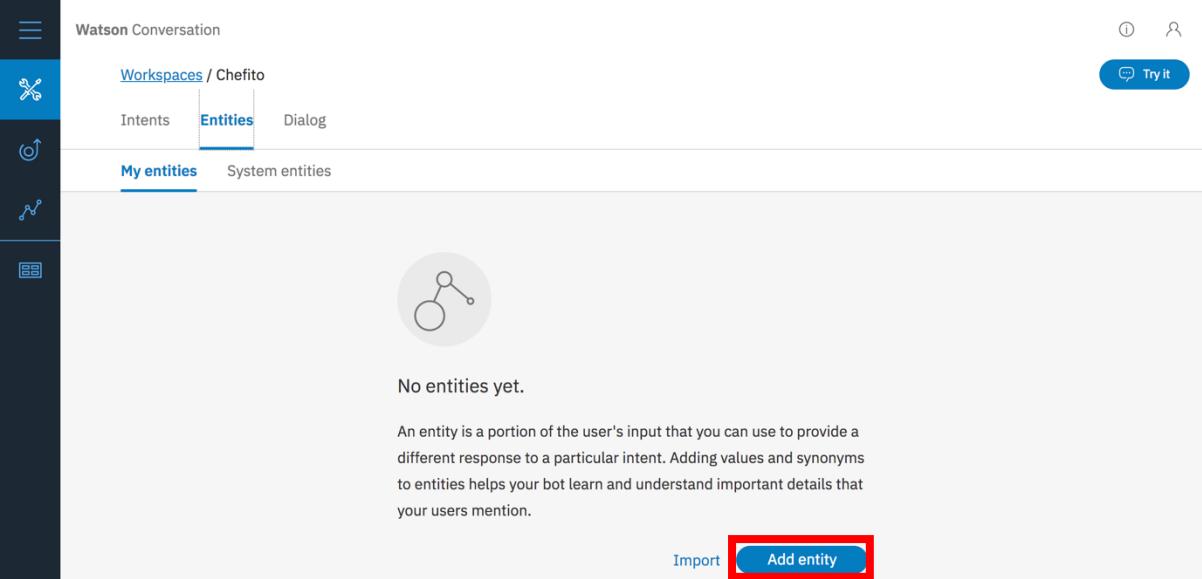
Ahora vamos a añadir las entidades, para eso hacemos click en entities como se muestra en la imagen:



The screenshot shows the Watson Conversation interface with the 'Entities' tab highlighted by a red box. There are five entities listed:

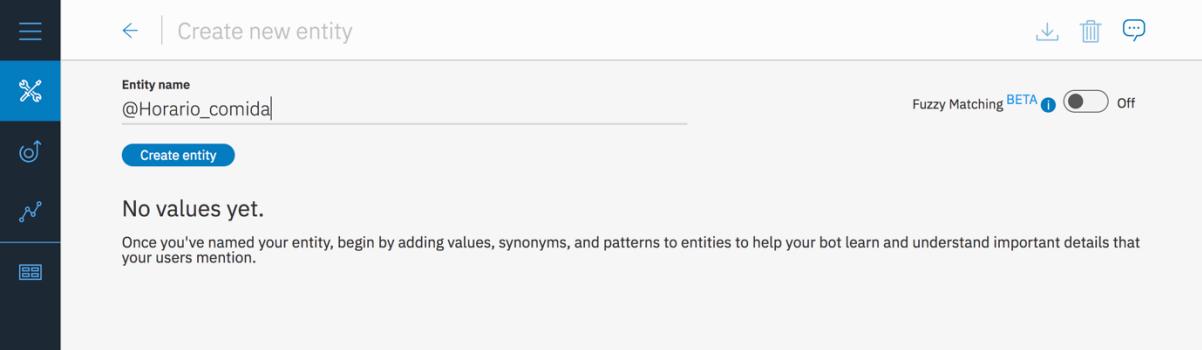
Entity	Description	Modified	Examples
#Afirmación		3 minutes ago	6
#Despedida		5 minutes ago	4
#Negación		a few seconds ago	3
#Recetas		5 days ago	6
#Saludo		7 days ago	2

Y hacemos click en **Add entity**



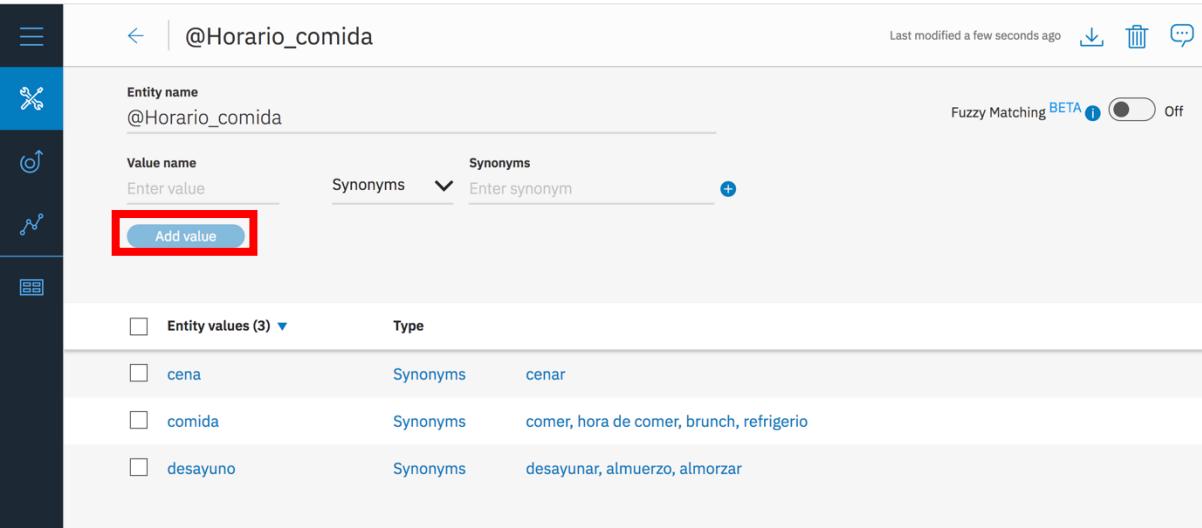
The screenshot shows the Watson Conversation interface under the 'Entities' tab. A large message says 'No entities yet.' Below it, a text block explains what entities are and how they help a bot learn. At the bottom right, there are 'Import' and 'Add entity' buttons, with 'Add entity' being highlighted by a red box.

Y como nombre le establecemos @Horario\_comida y hacemos click en **Create entity**



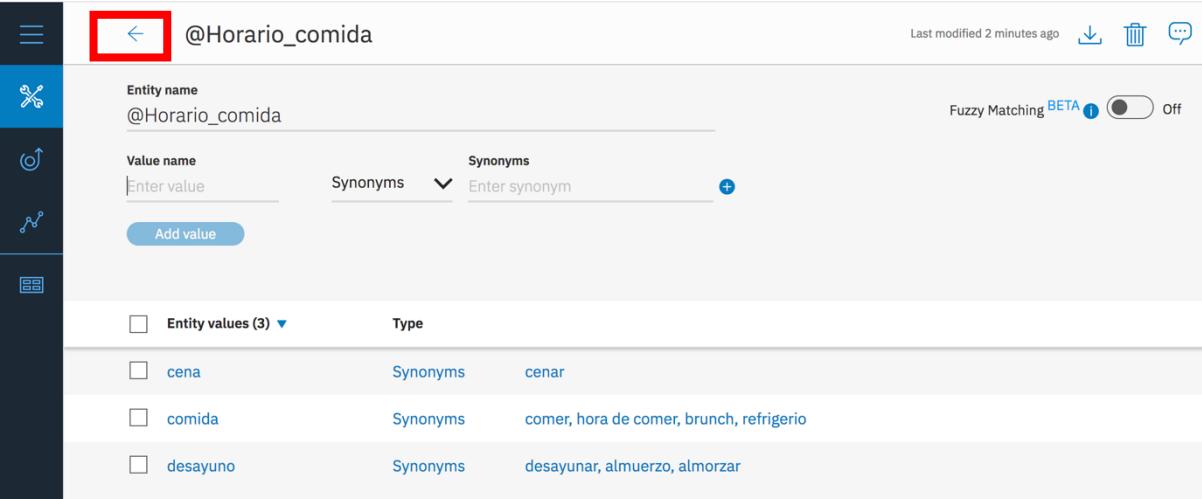
The screenshot shows the 'Create new entity' dialog. The entity name field contains '@Horario\_comida'. Below it, a 'Create entity' button is visible. A note says 'No values yet.' and provides instructions for adding values, synonyms, and patterns.

Una vez creada, añadimos valores para la entidad. En este caso van a ser los tres horarios de comida posible: desayuno, comida y cena (podemos añadir también sinónimos). Para añadir cada valor, hacemos click en **Add value**

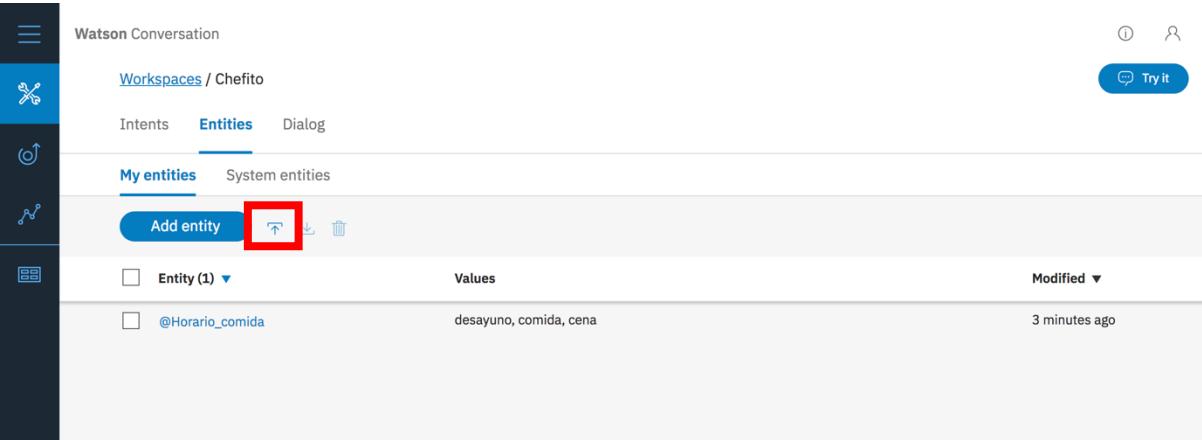


The screenshot shows the entity details page for '@Horario\_comida'. It lists three entity values: 'cena', 'comida', and 'desayuno', each associated with its type ('Synonyms') and examples ('cena', 'comer, hora de comer, brunch, refrigerio', 'desayunar, almuerzo, almorzar'). The 'Add value' button is highlighted with a red box.

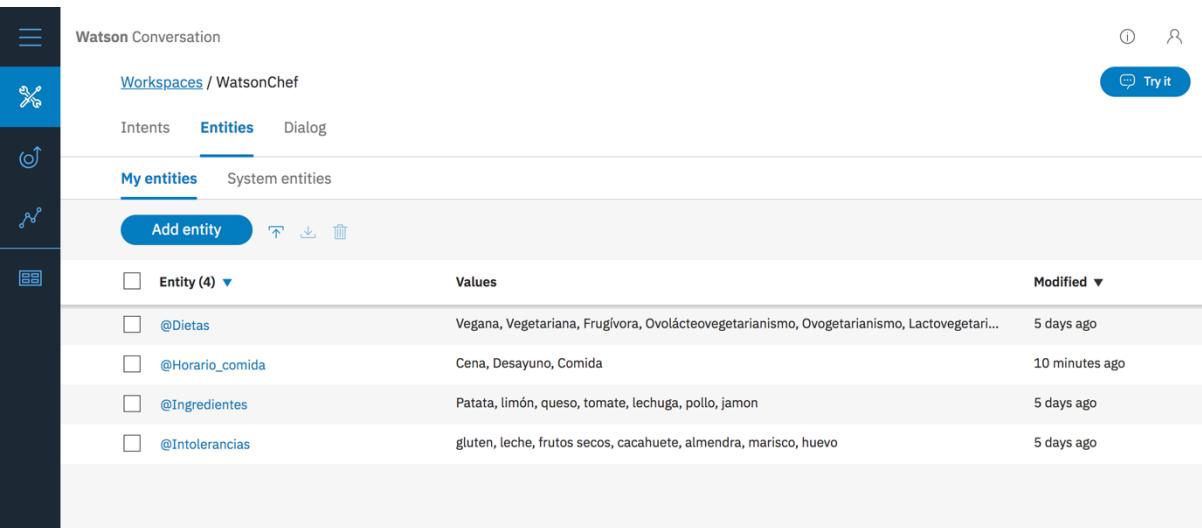
Para salir hacemos click de nuevo  y se guardan los cambios de manera automática



Importamos el resto de entidades haciendo click en el icono 'Importar'



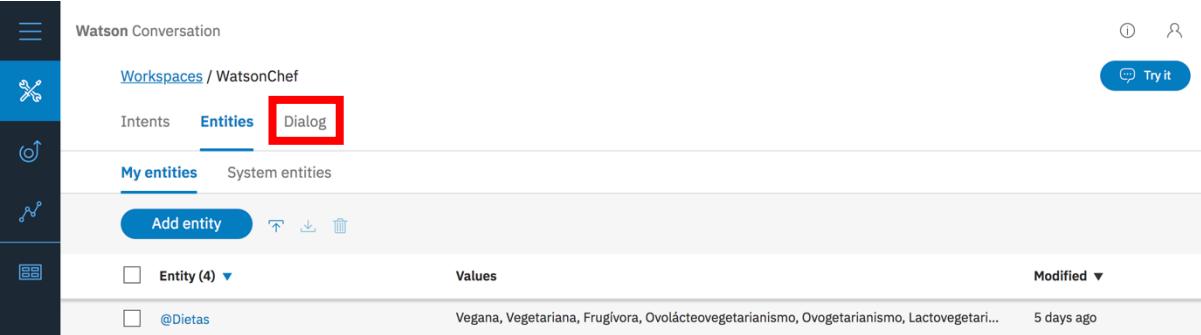
Y buscamos el archivo '**entities.csv**' dentro de la carpeta que hemos descargado en el laboratorio 2, en concreto en la carpeta workspaces del lab3.



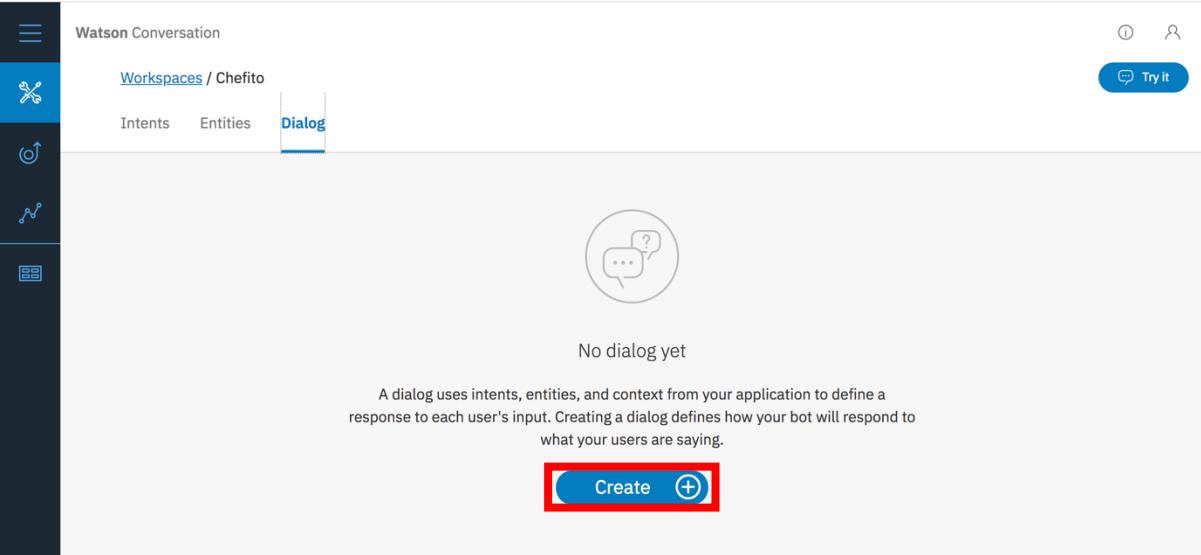
Como resultado, tenemos todas las entidades disponibles para definir la conversación.

*Copyright IBM Corporation 2013-2017. All rights reserved.*

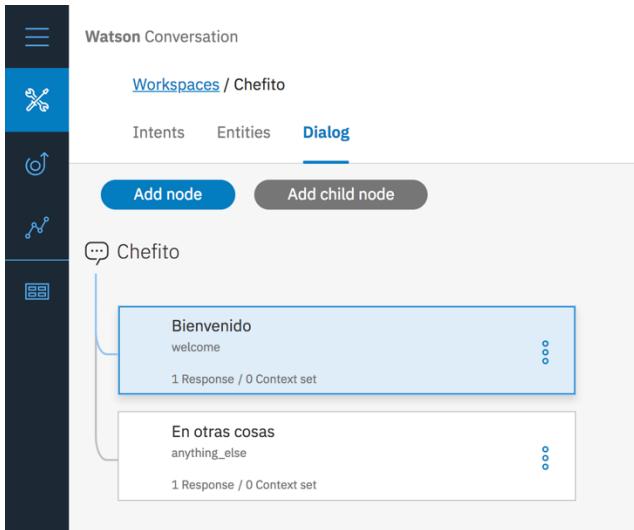
Ahora llega el turno de diseñar nuestra conversación. Hacemos click en la pestaña **Dialog** para empezar a trabajar



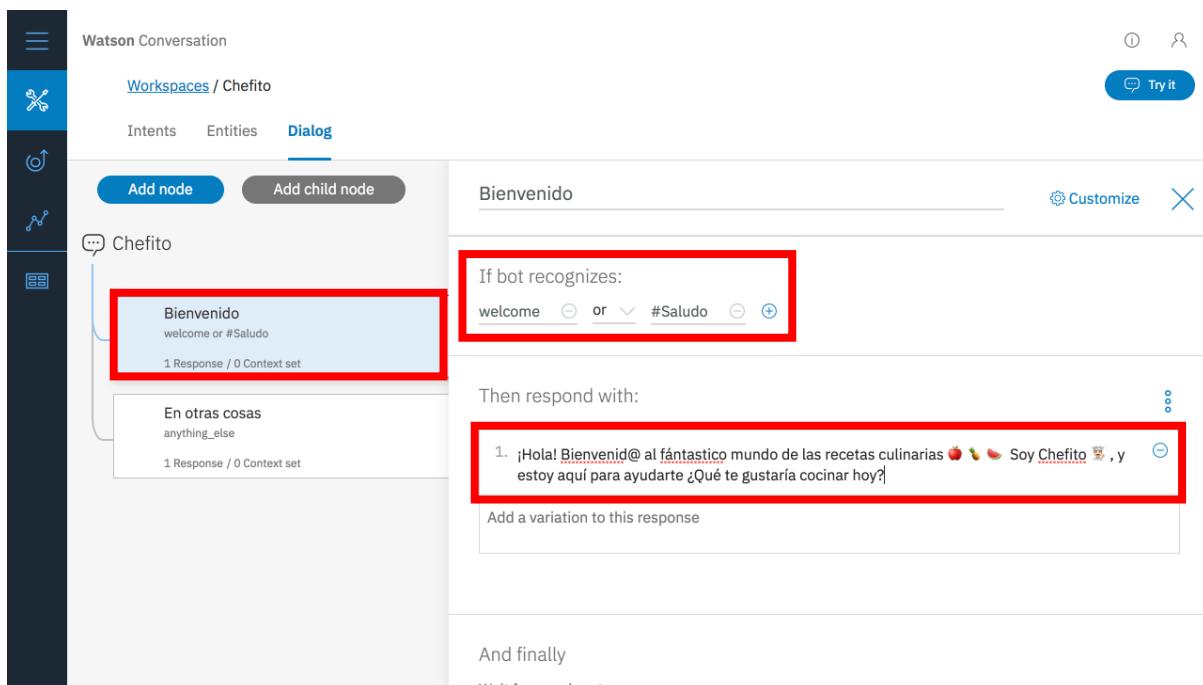
Hacemos click en **create +**



Y vemos como automáticamente se crean dos nodos más, **Bienvenido** y **En otras cosas**. El nodo **Bienvenido** es utilizado para iniciar la conversación desde el chatbot y el nodo **En otras cosas** es ejecutado cuando no se detecta ninguna entidad o intención, como nodo por defecto.



Definimos primero cómo vamos a saludar a los usuarios, para ello hacemos click en el nodo **Bienvenido** para editar el contenido



The screenshot shows the Watson Conversation interface. On the left, there's a sidebar with icons for workspace management, nodes, entities, intents, and dialog. The main area is titled 'Watson Conversation' and shows a workspace named 'Workspaces / Chefito'. Under the 'Dialog' tab, a node named 'Bienvenido' is selected. A red box highlights the condition section: 'If bot recognizes: welcome or #Saludo'. Another red box highlights the response section: 'Then respond with: ¡Hola! Bienvenid@ al fántastico mundo de las recetas culinarias 🍎🍍🍉 Soy Chefito 🍔, y estoy aquí para ayudarte ¿Qué te gustaría cocinar hoy?'. Below these, there's a field for 'Add a variation to this response'.

Lo primero es especificar la condición que debe reconocer el bot para ejecutar este nodo. En este caso, por defecto hay una acción específica para dar la bienvenida (**welcome**) pero vamos a añadirle que sea capaz de responder al usuario con este mismo nodo cuando reciba un saludo. Para eso, añadimos como condición alternativa (OR) que se detecte la intención **#Saludo** como se muestra en la imagen anterior.

En segundo lugar debemos establecer la respuesta que queremos dar al usuario cuando nos salude o queramos saludarle, así que en **Then respond with** añadimos la respuesta:

*¡Hola! Bienvenid@ al fántastico mundo de las recetas culinarias 🍎🍍🍉 Soy Chefito 🍔, y estoy aquí para ayudarte ¿Qué te gustaría cocinar hoy?*

Además el campo **Add a variation to this response** nos permite crear respuestas alternativas, para que el bot use diferentes respuestas para el mismo momento de la conversación.

Finalmente, dedicamos que acción llevar a cabo después de enviar nuestra respuesta. Tenemos tres posibilidades:

- ✓ Esperar la respuesta del usuario
- ✓ Saltarnos la respuesta del usuario
- ✓ Saltar a otro nodo

En nuestro caso en **Add finally** vamos a dejarlo con el valor por defecto **Wait for user input** para esperar la respuesta del usuario.

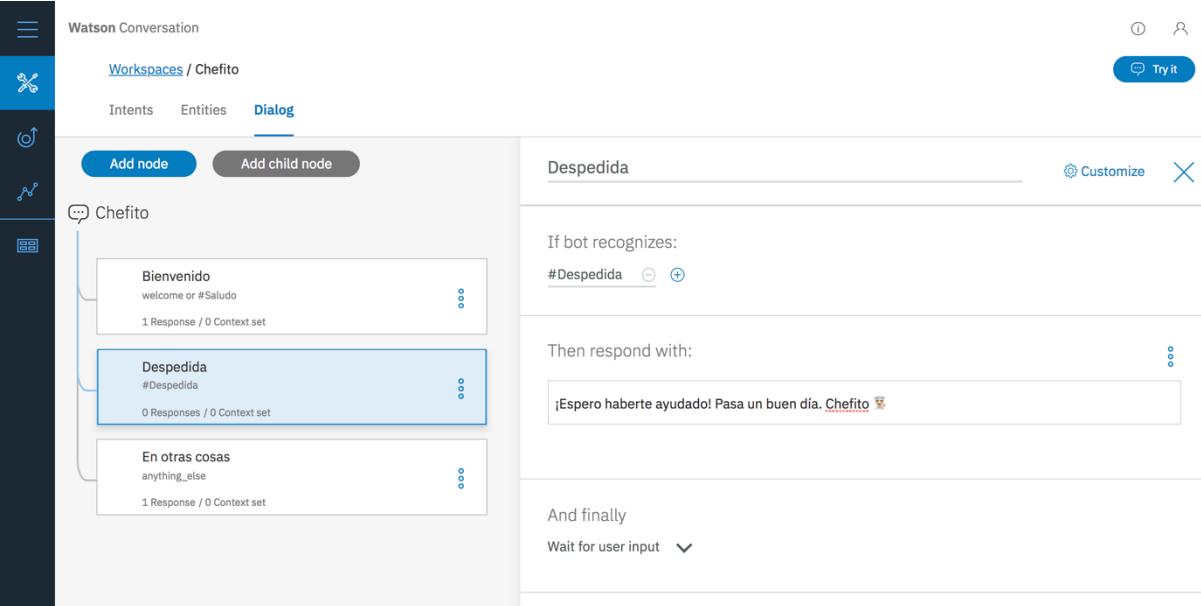
Una vez editado el contenido, hacemos click en la **X** y los cambios se guardan automáticamente.



The screenshot shows the Watson Conversation interface. On the left is a sidebar with icons for workspace management, intents, entities, and dialog. The main area is titled 'Watson Conversation' under 'Workspaces / WatsonChef'. It has tabs for 'Intents', 'Entities', and 'Dialog', with 'Dialog' selected. A button bar at the top right includes 'Try it' and 'Customize'. Below the tabs, there are buttons for 'Add node' and 'Add child node'. A list of nodes on the left includes 'Bienvenido' (selected), 'WatsonChef', and others like 'Despedida' and 'En otras cosas'. The 'Bienvenido' node's details panel shows 'Bienvenido' with 'welcome or #Saludo' as its condition and '1 Response / 0 Context set' as its response. To the right of the node list is a large text input field containing the message '¡Espero haberte ayudado! Pasa un buen día. Chefito' followed by a small chef emoji.

Ahora vamos a repetir el proceso, pero vamos a crear un nodo para detectar cuando el usuario se está despidiendo. Repite los pasos anteriores, pero esta vez crea un nodo cuya condición sea detectar la intención **#Despedida** y devuelva la siguiente respuesta:

*¡Espero haberte ayudado! Pasa un buen día. Chefito* 



This screenshot shows the same Watson Conversation interface as the previous one, but with a different dialog structure. The 'Bienvenido' node from the previous screenshot is now part of a larger tree under a 'Despedida' parent node. The 'Despedida' node has a condition of '#Despedida' and a response of '¡Espero haberte ayudado! Pasa un buen día. Chefito' with a chef emoji. The 'Bienvenido' node remains in the tree, and a new node 'En otras cosas' with condition 'anything\_else' is also present. The 'Despedida' node is highlighted with a red box around its delete icon.

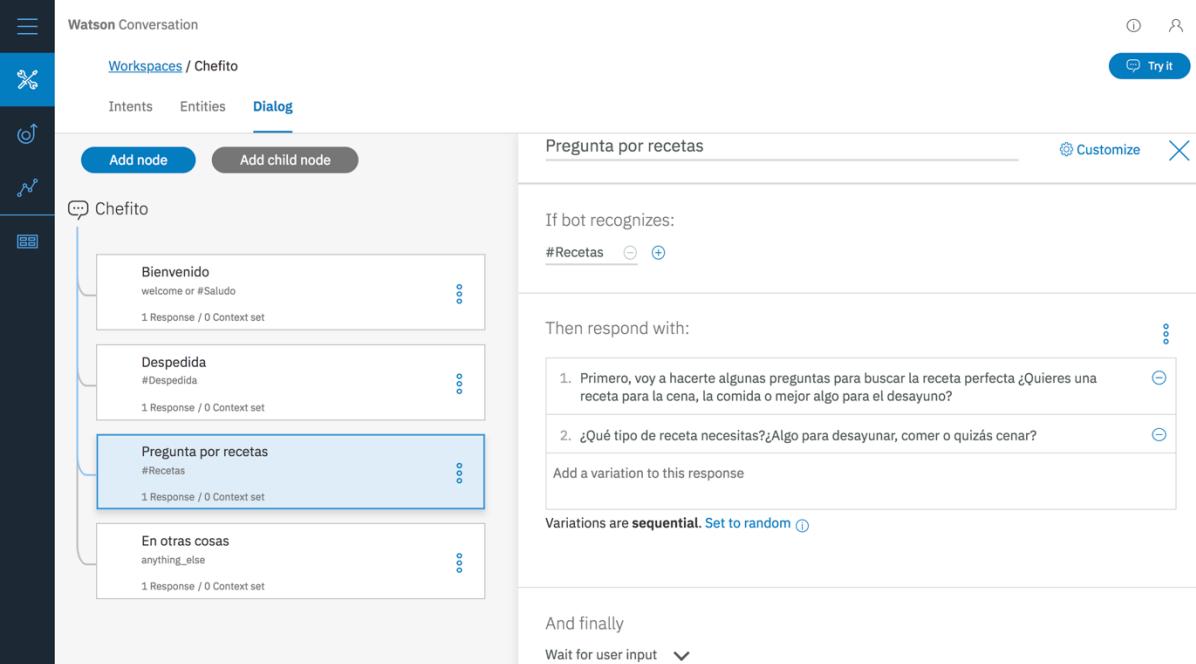
¡Perfecto! Ya tenemos controlado el saludo y la despedida del usuario con nuestro bot. Ahora vamos a crear el contenido de la conversación, una pequeña píldora de cómo se entrena Watson para que podáis crear vuestros propios bots.

Hacemos click en **Add node** y vamos a crear un nodo para responder al usuario cuando nos pregunte por una receta o qué puede comer. De nombre elegimos **pregunta por recetas** y como condición vamos a especificar que detecte la intención **#Recetas** que incluía ejemplos como ¿Qué puedo comer?, Quiero cocinar...

Como respuesta vamos a poner dos alternativas:

- 1- *Primero, voy a hacerte algunas preguntas para buscar la receta perfecta ¿Quieres una receta para la cena, la comida o mejor algo para el desayuno?*
- 2- *¿Qué tipo de receta necesitas? ¿Algo para desayunar, comer o quizás cenar?*

Y como acción dejamos que espere que le responda el usuario como se muestra en la imagen



Pregunta por recetas

If bot recognizes:  
#Recetas

Then respond with:

1. Primero, voy a hacerte algunas preguntas para buscar la receta perfecta ¿Quieres una receta para la cena, la comida o mejor algo para el desayuno?
2. ¿Qué tipo de receta necesitas? ¿Algo para desayunar, comer o quizás cenar?

Add a variation to this response

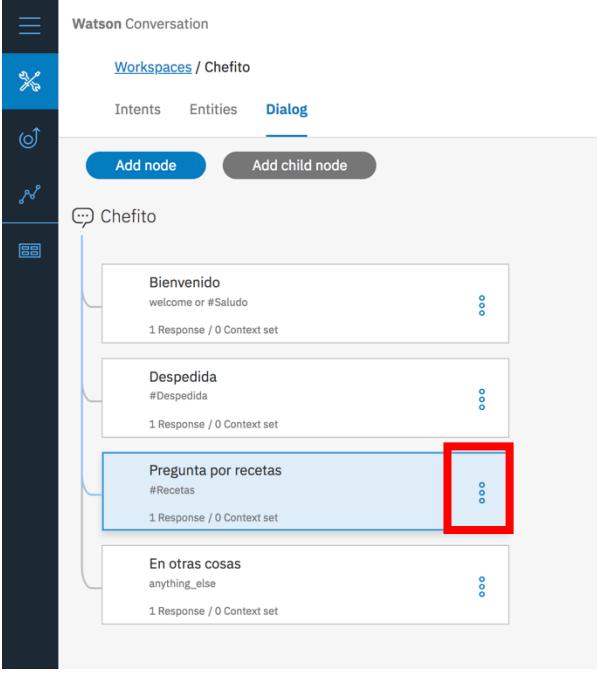
Variations are sequential. Set to random ⓘ

And finally

Wait for user input ▾

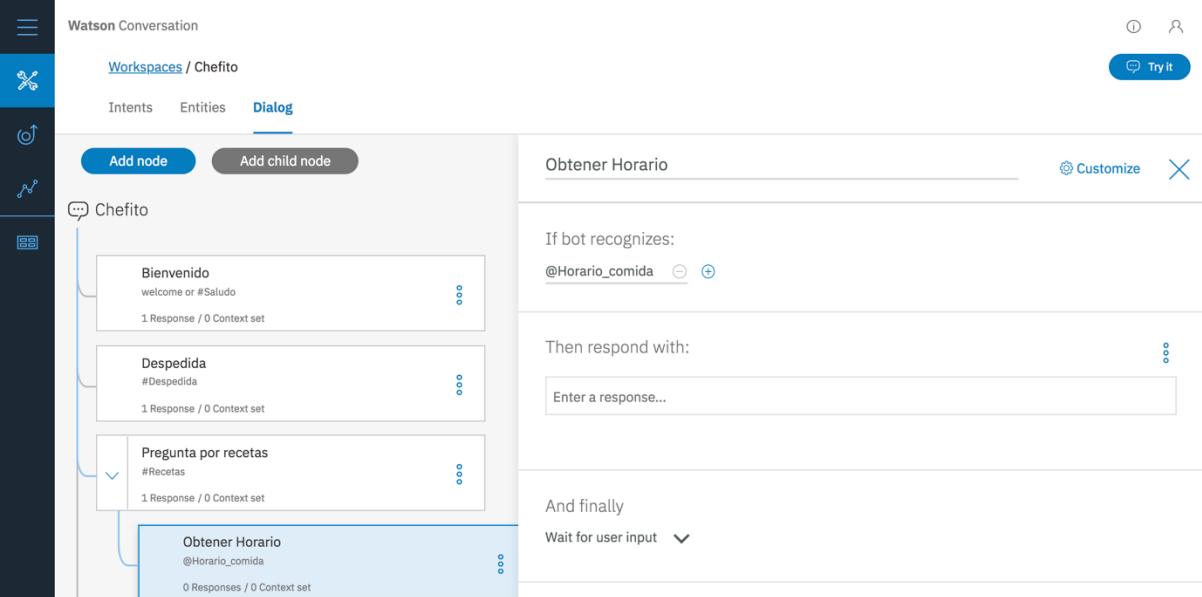
Hacemos click en la X para que se guarden los cambios de manera automática.

Ahora vamos a agregar un nodo hijo al nodo que acabamos de crear para distinguir si la receta que quiere el usuario es para el desayuno, la comida o la cena. Hacemos click en los tres puntitos del nodo **Pregunta por recetas**



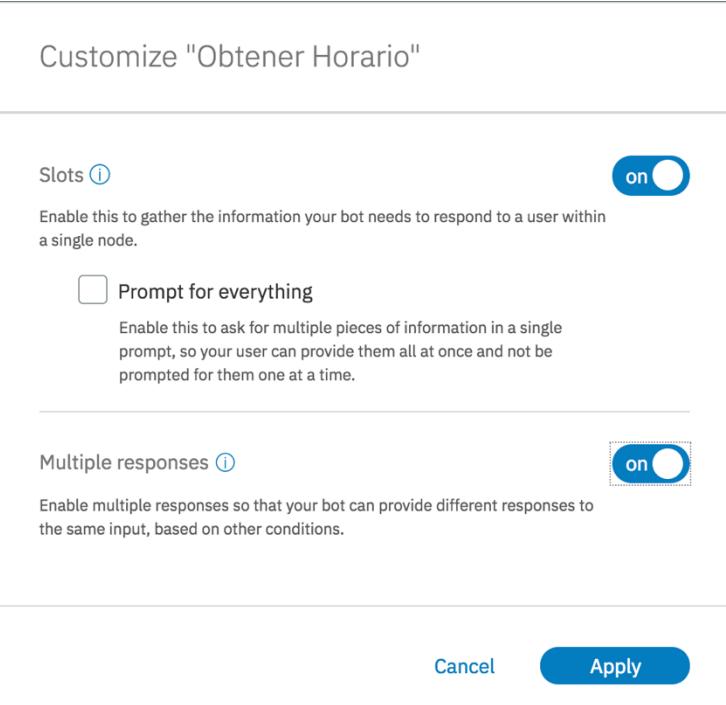
Y hacemos click en **Add child Node**

Para el nuevo nodo elegimos como nombre **Obtener Horario** y la condición va a ser detectar la entidad **@Horario\_comida**



The screenshot shows the Watson Conversation interface. On the left, there's a sidebar with icons for navigation and a main area titled "Watson Conversation" showing "Workspaces / Chefito". Below this are tabs for "Intents", "Entities", and "Dialog". Under "Dialog", there are buttons for "Add node" and "Add child node". The main workspace area displays a tree structure for a "Chefito" workspace. Nodes include "Bienvenido" (welcome or Saludo), "Despedida" (#Despedida), "Pregunta por recetas" (#Recetas), and "Obtener Horario" (@Horario\_comida). The "Obtener Horario" node is currently selected. To the right of the tree, a configuration panel for this node is shown. It has sections for "If bot recognizes:" (with a slot entry for "@Horario\_comida"), "Then respond with:" (a text input field labeled "Enter a response..."), and "And finally" (a section for "Wait for user input").

Vamos a aprovechar en este nodo otra capacidad de Watson Conversation que son los **SLOTS**. Los **SLOTS** nos van a permitir recoger diferentes datos del usuario a partir de un único nodo. Para ello hacemos click en [Customize](#) :



The screenshot shows the "Customize "Obtener Horario"" dialog. It contains two main sections: "Slots" and "Multiple responses".

- Slots**: A toggle switch is set to "on". Description: "Enable this to gather the information your bot needs to respond to a user within a single node." There is also a checkbox for "Prompt for everything" which is unchecked.
- Multiple responses**: A toggle switch is set to "on". Description: "Enable multiple responses so that your bot can provide different responses to the same input, based on other conditions."

At the bottom of the dialog are "Cancel" and "Apply" buttons.

Y activamos tanto los Slots como las respuestas múltiples, que nos va a permitir definir una respuesta diferente en función de la información proporcionada por el usuario. Para guardar hacemos click en [Apply](#).

Obtener Horario

If bot recognizes:

@Horario\_comida   - +

Then check for:

Check for	Save it as	If not present, ask	Type
1 @Horario_comida	\$Horario_comida	Enter a prompt	Optional <span style="border: 1px solid #ccc; padding: 2px;"> </span> <span style="border: 1px solid #ccc; padding: 2px;">-</span> <span style="border: 1px solid #ccc; padding: 2px;">X</span>

+ Add slot

Podemos observar que la interfaz ha cambiado ligeramente para añadir los slots y las múltiples respuestas. Primero vamos a configurar un slot para recoger en una variable la información de horario de comida que nos proporciona el usuario, que nos podría servir en futuras interacciones con el usuario.

En este laboratorio sólo vamos a hacerlo con una variable, pero podríamos guardar en el contexto con variables la información referente a otras entidades o intenciones detectadas para usarlas durante el flujo de la conversación o para interactuar con sistemas terceros.

Agregamos en la sección **Check for** la entidad @Horario\_comida y automáticamente se completa el campo **Save it as** que nos va a permitir guardar el valor de dicha entidad en la variable de contexto \$Horario\_comida

Obtener Horario

If bot recognizes:

@Horario\_comida   - +

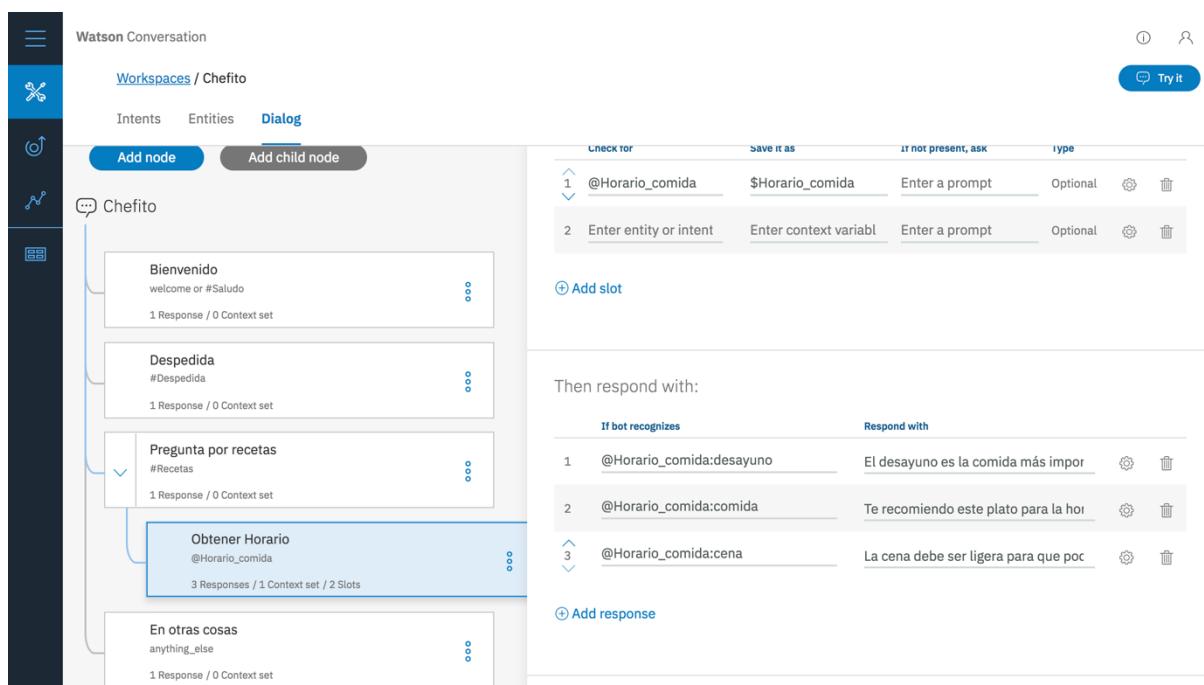
Then check for:

Check for	Save it as	If not present, ask	Type
1 @Horario_comida	\$Horario_comida	Enter a prompt	Optional <span style="border: 1px solid #ccc; padding: 2px;"> </span> <span style="border: 1px solid #ccc; padding: 2px;">-</span> <span style="border: 1px solid #ccc; padding: 2px;">X</span>

+ Add slot

Ahora vamos a la sección **Then respond with** donde vamos a responder al usuario con una receta diferente dependiendo de si está buscando una receta para el desayuno, la comida o la cena.

If bot recognizes	Respond with
@Horario_comida:desayuno	El desayuno es la comida más importante del día así que te recomiendo esta receta para que empieces bien el día <a href="https://danzadefogones.com/tostadas-francesas-french-toast-veganas/">https://danzadefogones.com/tostadas-francesas-french-toast-veganas/</a>
@Horario_comida:comida	Te recomiendo este plato para la hora de comer y que te permite tener una dieta sana y equilibrada <a href="https://danzadefogones.com/curry-de-verduras/">https://danzadefogones.com/curry-de-verduras/</a>
@Horario_comida:cena	La cena debe ser ligera para que podamos tener un sueño profundo. Te recomiendo esta receta <a href="https://danzadefogones.com/rollos-tempah-marinado/">https://danzadefogones.com/rollos-tempah-marinado/</a>



Check for	Save it as	If not present, ask	Type
1 @Horario_comida	\$Horario_comida	Enter a prompt	Optional
2 Enter entity or intent	Enter context variabl	Enter a prompt	Optional

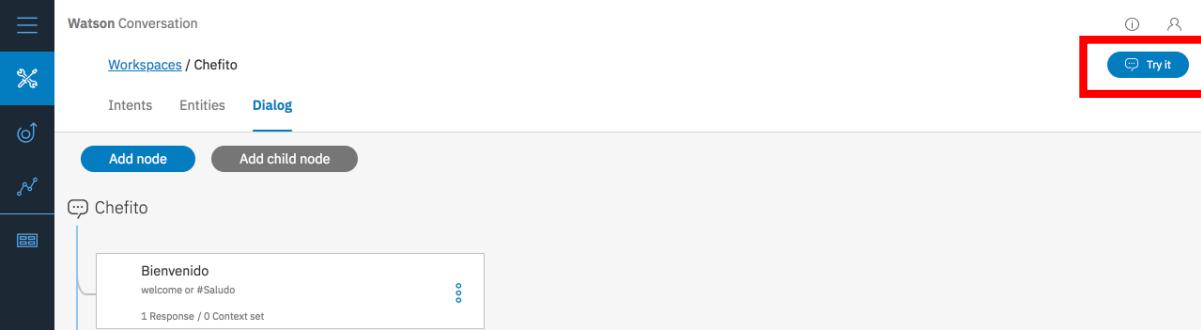
Then respond with:

If bot recognizes	Respond with
1 @Horario_comida:desayuno	El desayuno es la comida más impor
2 @Horario_comida:comida	Te recomiendo este plato para la hor
3 @Horario_comida:cena	La cena debe ser ligera para que poc

Dejamos de nuevo que el nodo espere la respuesta del usuario y cerramos la configuración del nodo.

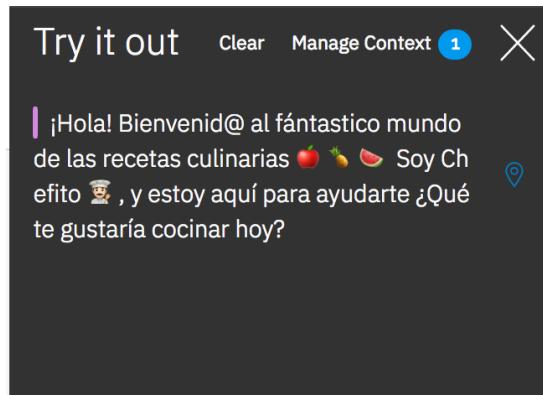
Ya tenemos una conversación definida para nuestro bot. Tened en cuenta que es un ejercicio de laboratorio corto y sencillo para mostrar la fase de entrenamiento de un bot, por lo que una conversación real para que funcione correctamente debería tener más nodos, definir correctamente el flujo de todos los nodos y almacenar las variables de contexto necesarias.

¡Toca probar nuestro bot! Desde la propia interfaz del servicio de conversation somos capaces de probar la conversación, así que hacemos click en el botón **Try it**



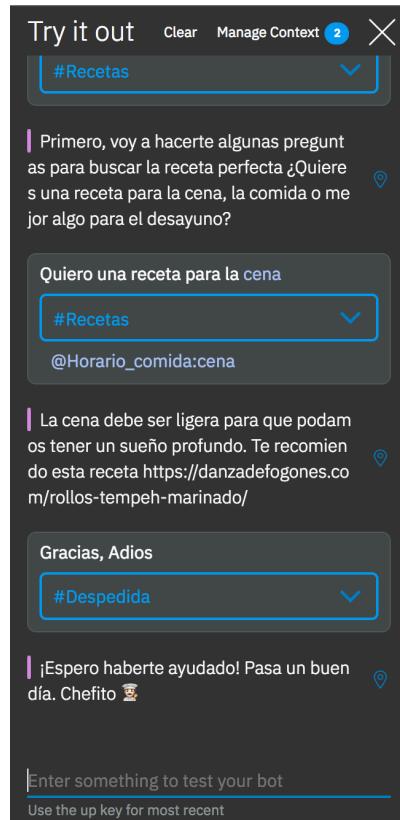
The screenshot shows the Watson Conversation interface. On the left is a sidebar with icons for workspace management, intents, entities, and dialog. The main area shows a workspace named 'Workspaces / Chefito'. Under the 'Dialog' tab, there's a node labeled 'Bienvenido' with the sub-node 'welcome or #Saludo'. Below the nodes, there's a message box containing the greeting '¡Hola! Bienvenid@ al fántastico mundo de las recetas culinarias' followed by some emojis. The 'Try it' button is highlighted with a red box.

Y se nos abre un panel que inicia con la frase de saludo que hemos implementado



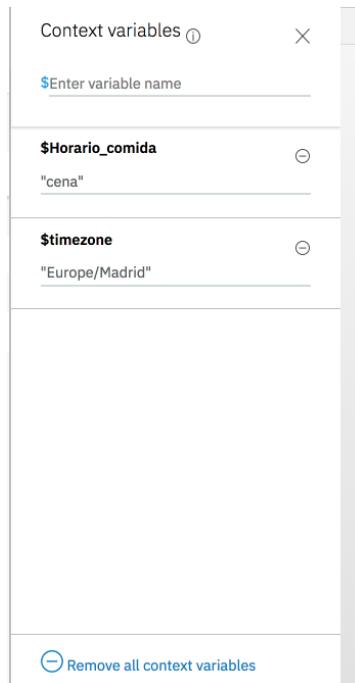
Probamos la conversación para ver la navegación por los diferentes nodos (ten en cuenta que sólo hemos entrenado a nuestra conversación para que sea capaz de seguir el hilo de la conversación definido)





Podemos ver como en cada interacción con el usuario es capaz de detectar las entidades e intenciones en la conversación y responder con las respuestas que hemos definido en el diálogo.

Además, si hacemos click en [Manage Context](#) observamos las variables que se han ido almacenando, en concreto la variable \$horario\_comida que contiene el valor introducido en la conversación, en el caso del ejemplo, es cena.



The dialog shows the following context variables:

- \$Horario\_comida: "cena"
- \$timezone: "Europe/Madrid"

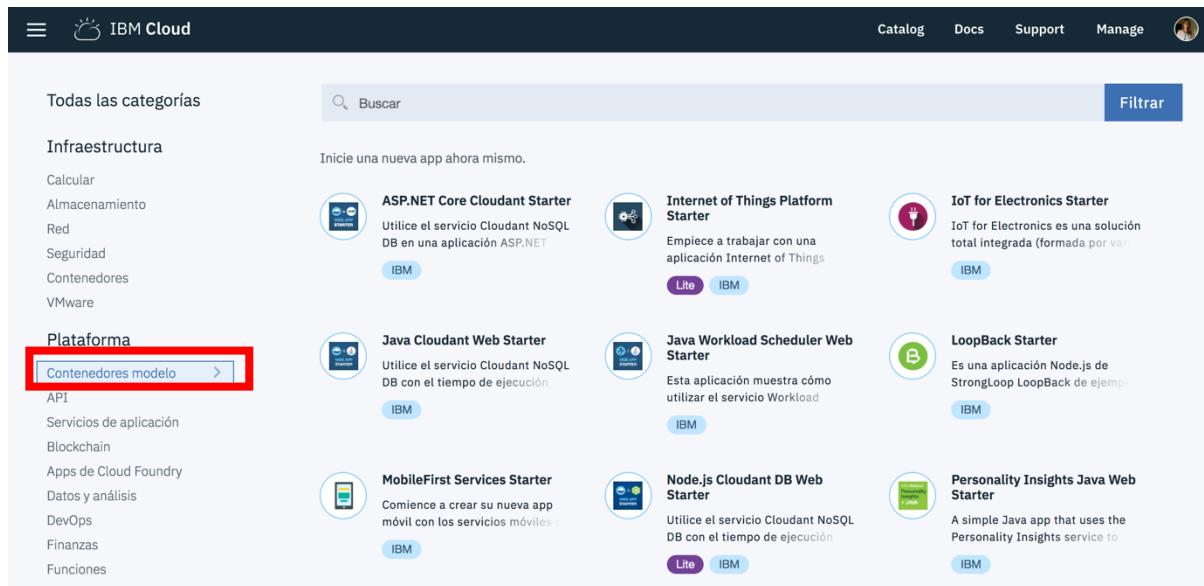
At the bottom, there is a button: [\(-\) Remove all context variables](#)

*¡Enhorabuena! Has completado la segunda parte del laboratorio. Ya sabes como crear tu propio bot con IBM Watson Conversation.*

### 3- Tu chatbot en telegram

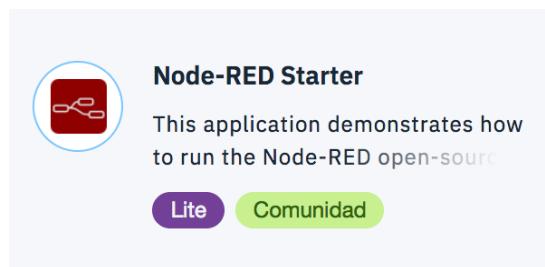
Ahora que ya tenemos definida la conversación vamos a exponerla para que los usuarios puedan acceder a ella a través de telegram.

Primero vamos a desplegar NodeRED en nuestra cuenta de IBM Cloud. Accedemos de nuevo a [bluemix.net](#) y vamos directos al catálogo y hacemos click en **Contenedores Modelo**

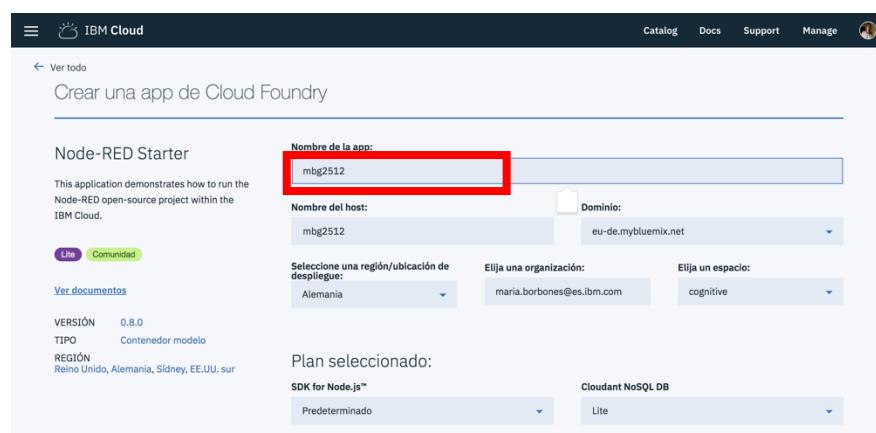


The screenshot shows the IBM Cloud Catalog interface. On the left, there's a sidebar with categories like Infraestructura, Plataforma, and Contenedores modelo. The 'Contenedores modelo' option is highlighted with a red box. The main area displays various application starters, including ASP.NET Core Cloudant Starter, Internet of Things Platform Starter, IoT for Electronics Starter, Java Cloudant Web Starter, Java Workload Scheduler Web Starter, LoopBack Starter, MobileFirst Services Starter, Node.js Cloudant DB Web Starter, and Personality Insights Java Web Starter. Each starter has a brief description, an icon, and two buttons: 'Lite' and 'IBM'.

buscamos Node-RED Starter y hacemos click sobre él

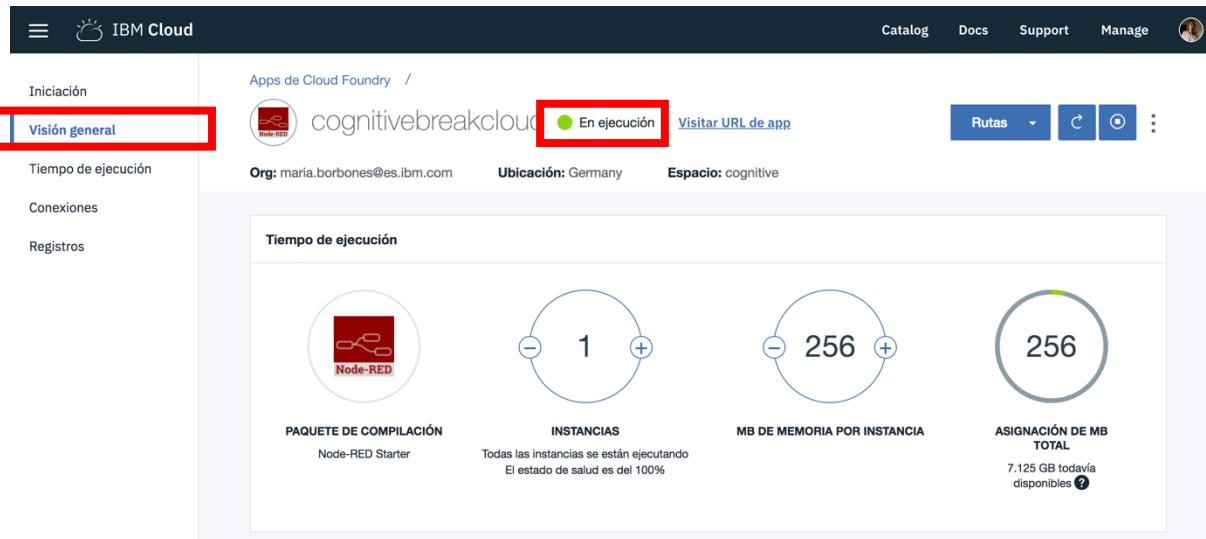


Asignamos un nombre que debe ser único (por ejemplo: iniciales y fecha de nacimiento) y dejamos el resto de valores que vienen por defecto

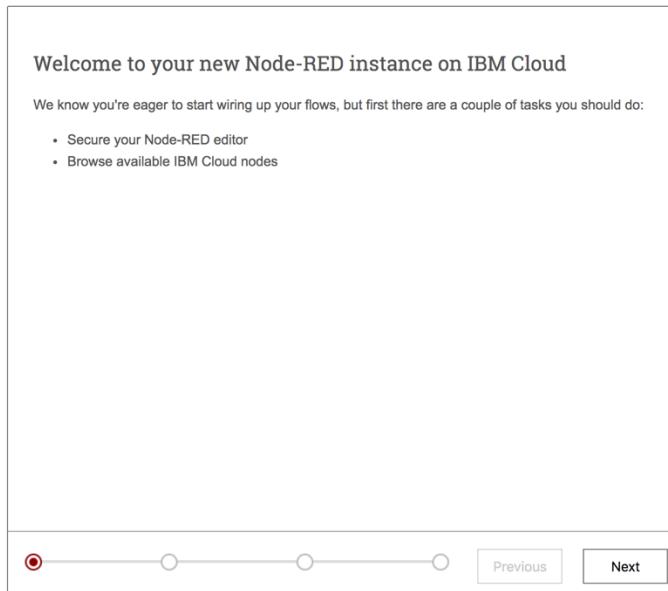


The screenshot shows the 'Crear una app de Cloud Foundry' (Create a Cloud Foundry app) form. The 'Nombre de la app:' field is filled with 'mgb2512' and is highlighted with a red box. Other fields include 'Nombre del host:' (mgb2512), 'Dominio:' (eu-de.mybluemix.net), 'Selección una región/ubicación de despliegue:' (Alemania), 'Eliga una organización:' (maria.borbones@es.ibm.com), 'Eliga un espacio:' (cognitive), and 'Plan seleccionado:' (Cloudant NoSQL DB Lite). There are also sections for 'VERSIÓN' (0.8.0), 'TIPO' (Contenedor modelo), and 'REGION' (Reino Unido, Alemania, Sidney, EE.UU, sur).

Hacemos click en crear y esperamos unos minutos a que la aplicación se haya desplegado por completo. Podemos ver el estado de despliegue de la aplicación si accedemos al tab de **Visión general**



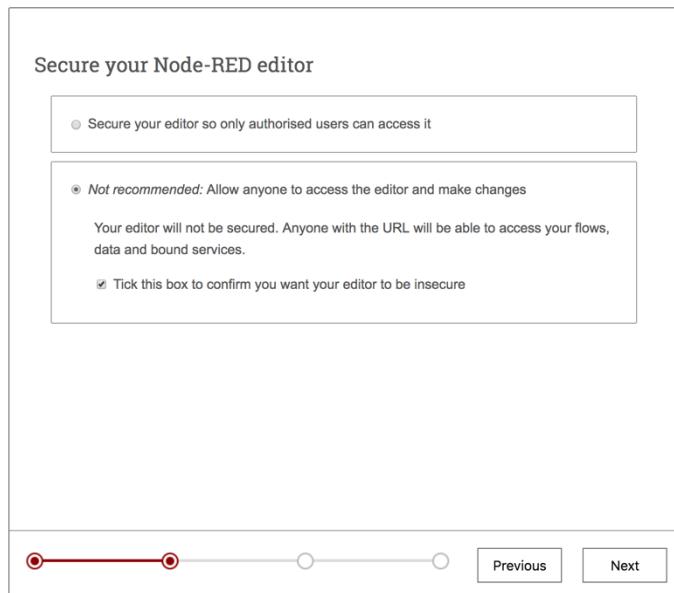
Una vez esté en ejecución, hacemos click en [Visitar URL de app](#) para acceder a la consola de Node-RED.



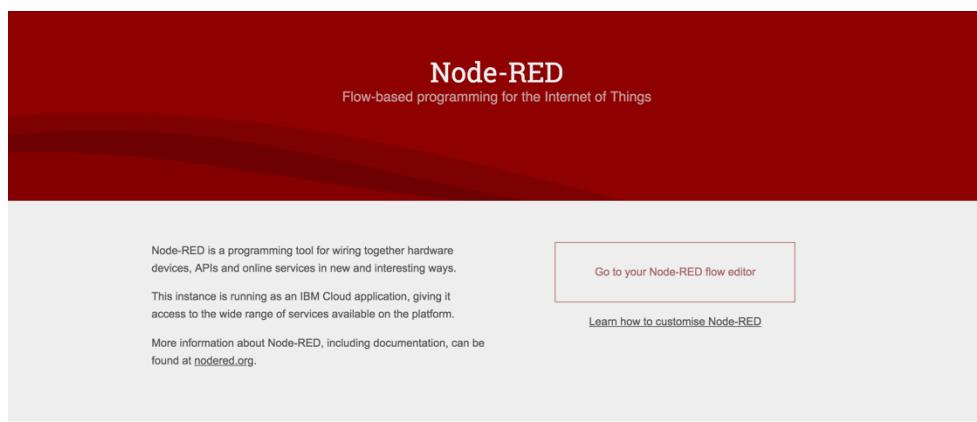
Hacemos click en Next y seleccionamos

- Not recommended: Allow anyone to access the editor and make changes

Y habilitamos el tick para confirmar que quieres utilizar el editor de forma no segura y hacemos click en Next



Hacemos click en siguiente hasta que lleguemos a la pantalla principal de Node-RED



Una vez allí, hacemos click en **Go to your Node-RED flow editor**

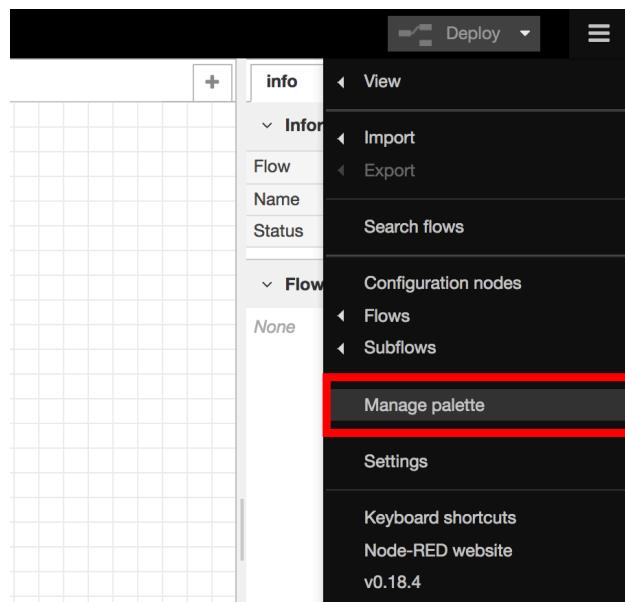
Node-RED es un entorno de desarrollo basado en flujos que permite crear lógica de aplicación de manera gráfica. Fue creado por IBM Research y es muy útil para crear prototipos de aplicaciones. Puedes encontrar más información en:

<https://nodered.org/>

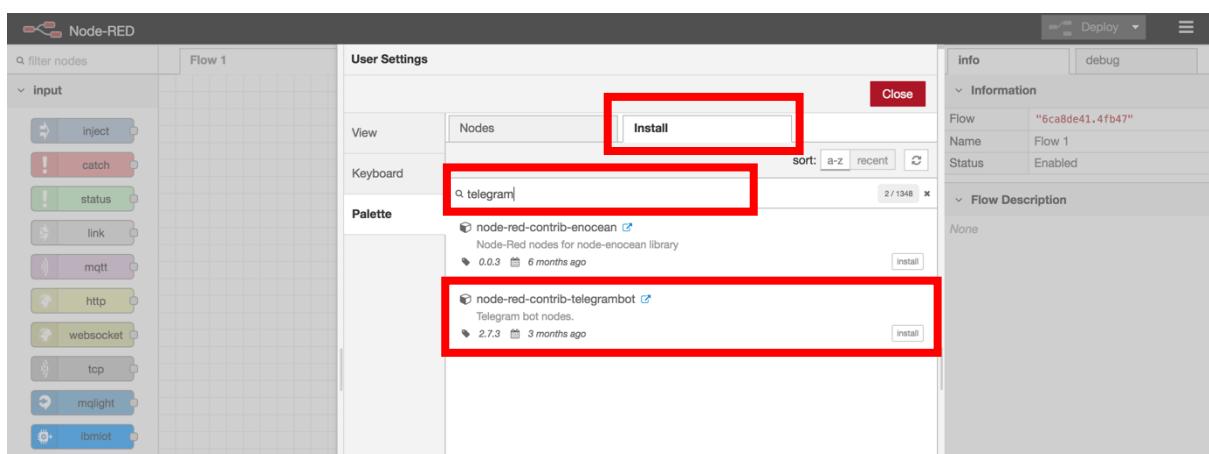
Una estamos en la consola de Node-RED, vamos a importar el flujo necesario para conectar nuestro chatbot con telegram. Hacemos click en las tres barritas de menú que podemos encontrar en la parte superior derecha de la página como se muestra en la imagen



Y seleccionamos **Manage palette** para añadir los nodos de telegram



En la ventana que aparece hacemos click en el tab **Install** y buscamos **telegram**



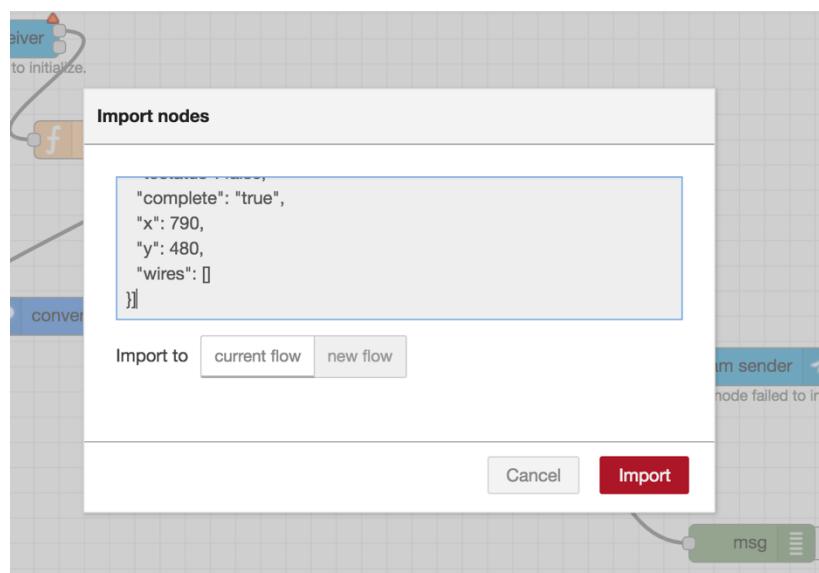
Instalamos el paquete **node-red-contrib-tegrambot**

Una vez instalado vamos a importar el flujo completo de Node-RED. Accedemos al directorio que nos hemos bajado en el laboratorio 2, pero esta vez a la carpeta del lab3, en concreto a la carpeta flujos. En un editor de texto abrimos el fichero **flujo\_conversation.json** y copiamos todo el contenido.

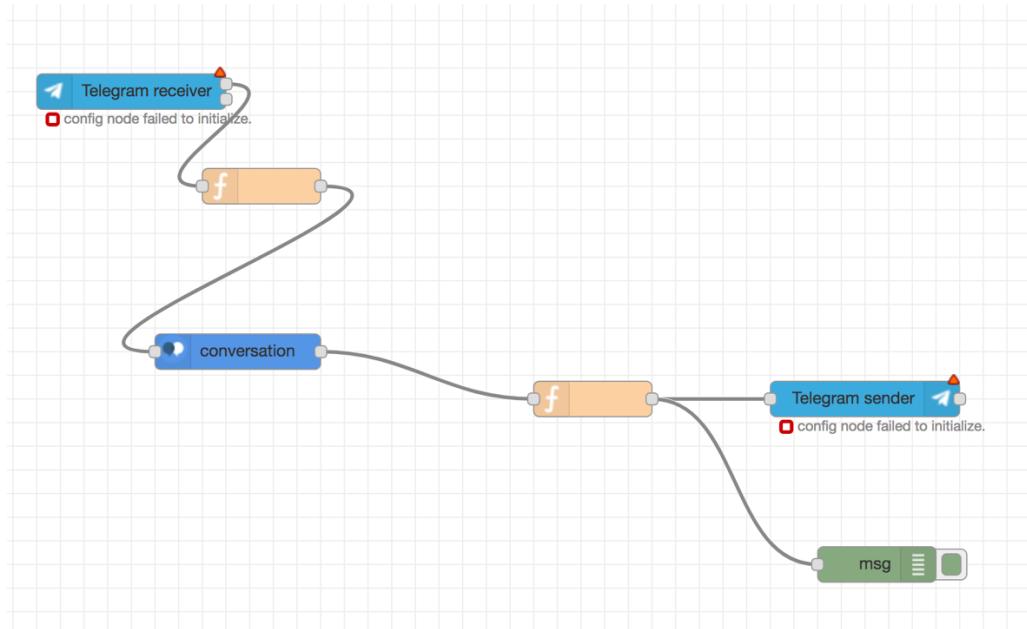
Después volvemos a nuestro Node-RED y volvemos a hacer click en las 3 barritas que se encuentran en la parte superior derecha de la pantalla.



Pero esta vez seleccionamos **Import > Clipboard** y pegamos el contenido de nuestro flujo en el cuadro de texto

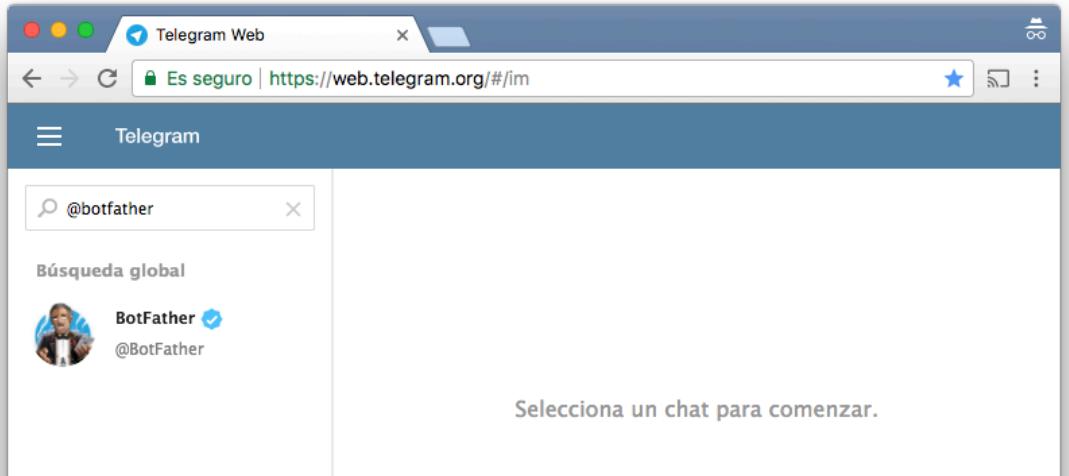


Una vez pegado, hacemos click en Import y se nos importará un flujo como el mostrado en la siguiente imagen

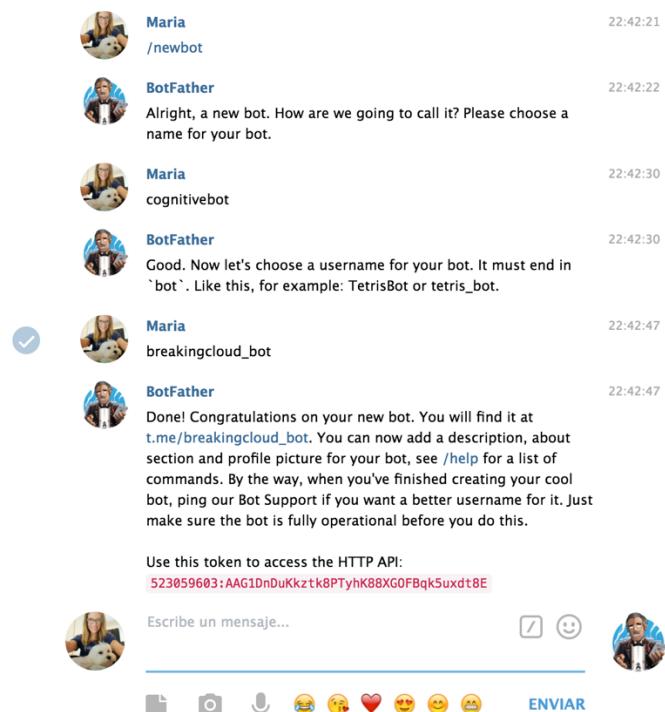


¡Perfecto! Ahora tienes tu flujo preparado para ser configurado. Pero antes...debemos crear nuestro chatbot en telegram.

Para ello, vamos a telegram <https://web.telegram.org>, buscamos el usuario **@botfather** como se muestra en la imagen e iniciamos una conversación con él



Para crear nuestro bot debemos mandarle el comando **/newbot** en la conversación y darle un nombre a nuestro bot como se muestra en la imagen:

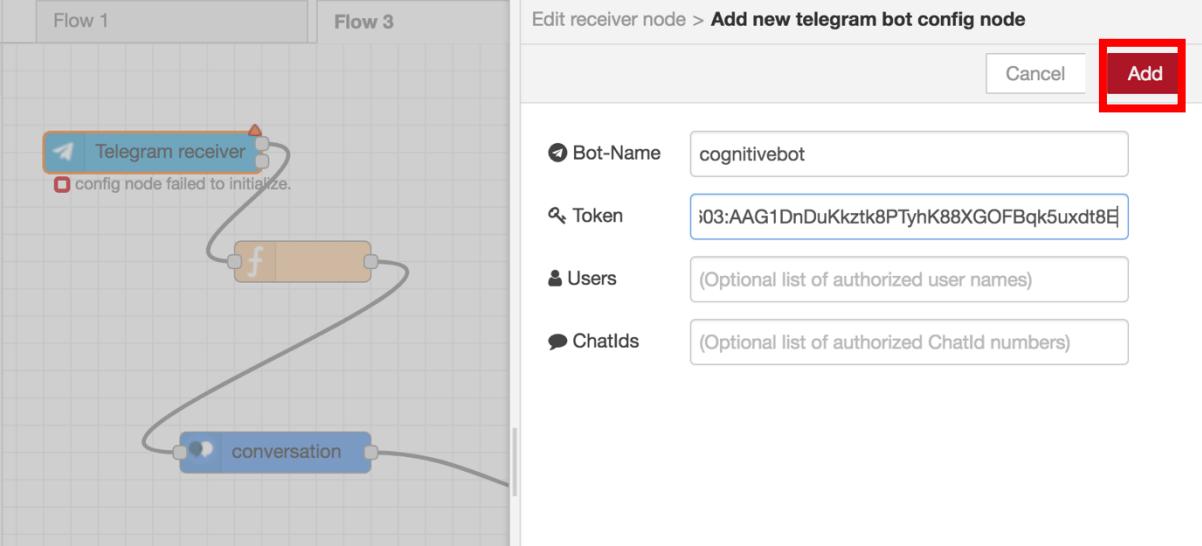


Una vez definido nuestro bot, nos proporciona un token que debemos guardar para configurar en Node-RED.

Volvemos a node-RED, hacemos doble click en el nodo **Telegram receiver**, y en el menú hacemos click en el lápiz para configurar un nuevo bot:

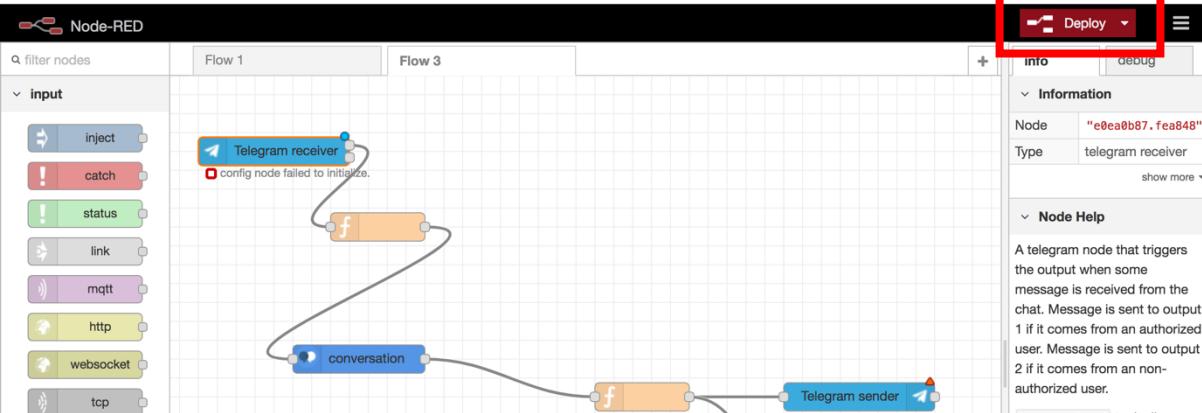


Ponemos el nombre de nuestro bot , el token que nos había generado telegram y hacemos click en **Add**



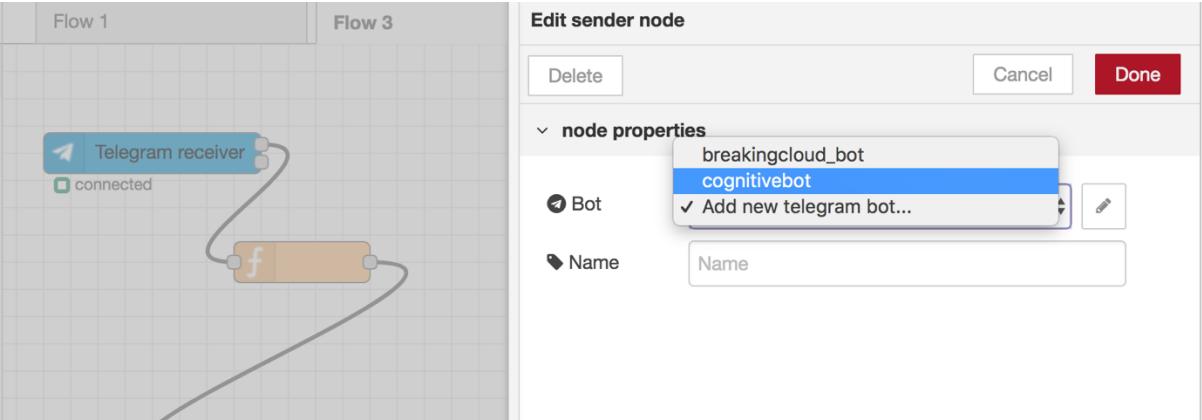
The screenshot shows the Node-RED interface with two flows. Flow 1 has a 'Telegram receiver' node followed by a 'conversation' node and a function node labeled 'f'. Flow 3 is partially visible. On the right, a configuration dialog for the 'Telegram receiver' node is open, showing fields for Bot-Name (cognitivebot), Token (a long string of characters), Users (optional list of authorized user names), and ChatIds (optional list of authorized ChatId numbers). The 'Add' button in the top right of the dialog is highlighted with a red box.

Una vez añadido hacemos click en **Done** para guardar los cambios. Y en el editor hacemos click en **Deploy** para guardar los cambios y desplegar el flujo.



The screenshot shows the Node-RED interface with the configured flow. The 'Deploy' button in the top right toolbar is highlighted with a red box. The right sidebar shows the 'Info' tab with details about the 'Telegram receiver' node, including its ID ('e0ea0b87fea848'), type ('telegram receiver'), and a description of its function as a telegram node that triggers output based on message source.

Repetimos el proceso con el nodo **Telegram Sender**, pero como nuestro bot ya está configurado sólo necesitamos seleccionarlo en el listado desplegable y hacemos click en **Done**



The screenshot shows the Node-RED interface with Flow 1. The 'Telegram receiver' node now has a green 'connected' status. The 'Edit sender node' dialog is open, showing the 'node properties' section where 'cognitivebot' is selected from a dropdown menu. The 'Done' button in the top right of the dialog is highlighted with a red box.

En el editor de nuevo hacemos click en **Deploy** para guardar los cambios.

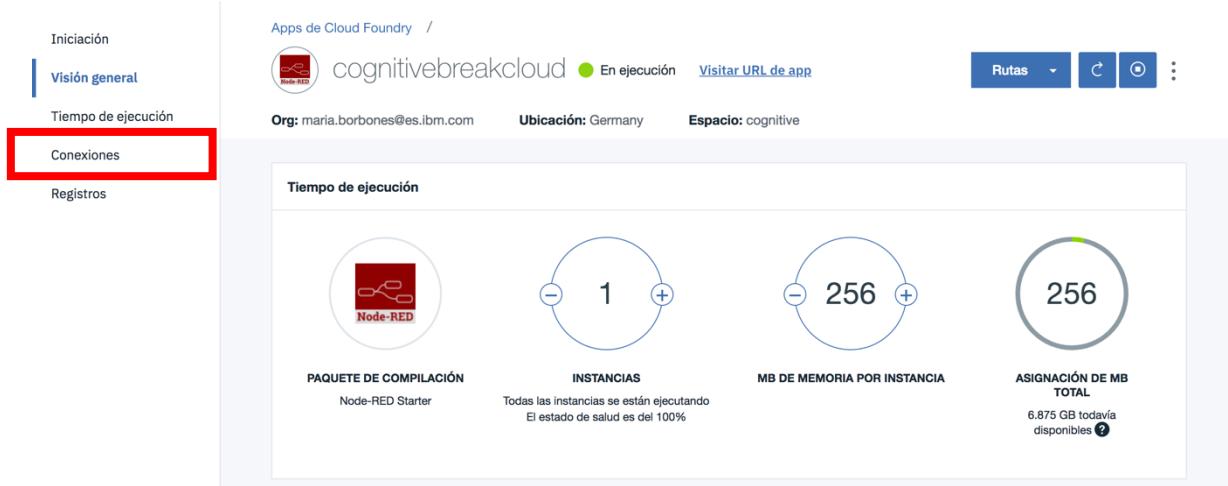
Ahora ya sólo queda configurar nuestro nodo de Conversation. Pero antes necesitamos hacer binding de nuestro Node-RED con nuestro servicio de Conversation.

Volvemos a la consola de IBM Cloud en [bluemix.net](#) y hacemos click en el icono de IBM Cloud para acceder al Dashboard



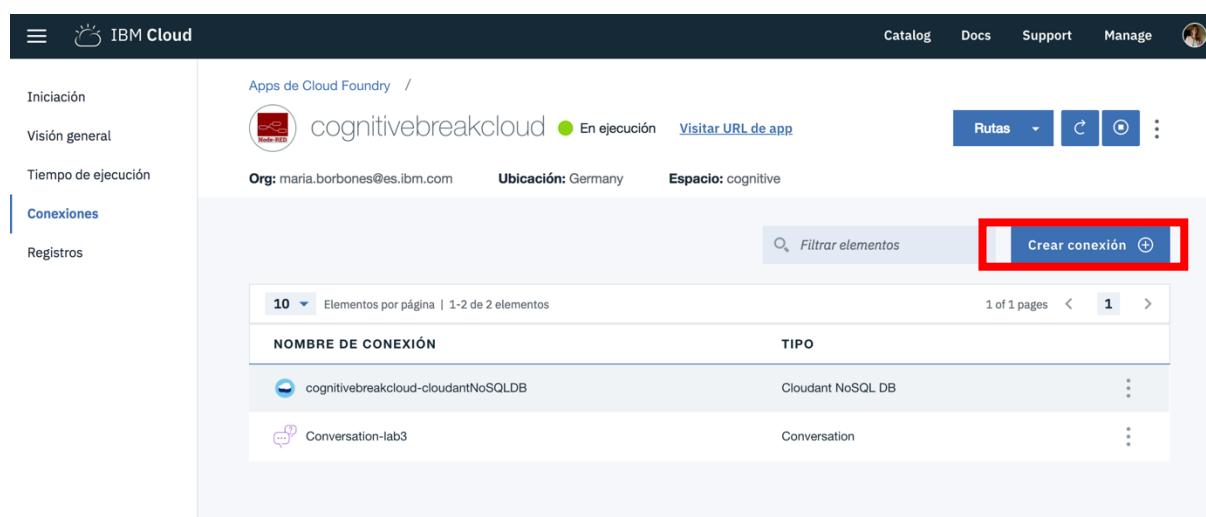
Una vez en el dashboard, buscamos nuestra aplicación de Node-RED (estará debajo de la categoría Apps de Cloud Foundry) y hacemos click sobre el nombre de la aplicación para acceder a su panel de control

En el menú lateral, buscamos la opción **Conexiones** y accedemos a él



Hacemos click sobre el botón **Crear conexión**

Hacemos click sobre el botón **Crear conexión**



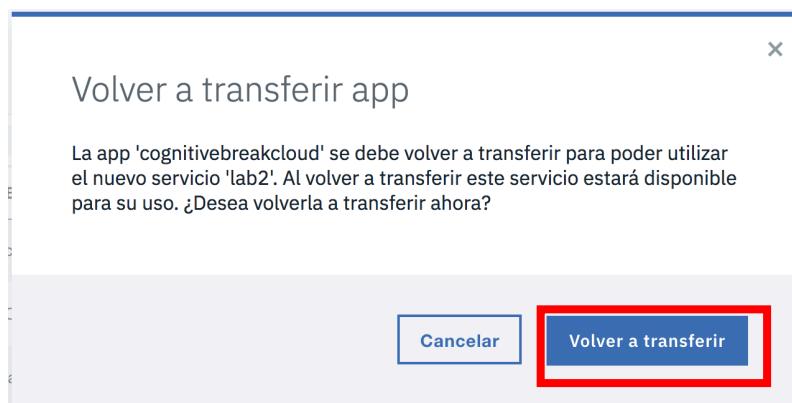
NOMBRE DE CONEXIÓN	TIPO
cognitivebreakcloud-cloudantNoSQLDB	Cloudant NoSQL DB
Conversation-lab3	Conversation

Y en la nueva página buscamos nuestro servicio de conversation y hacemos click sobre el botón **connect** (aparece al pasar el cursor por encima del servicio) para que conecte nuestro Node-RED con el servicio.



SERVICIOS	GRUPO DE RECURSOS	PLAN	OFERTA DE SERVICIOS
aleatoriorandom-cloudantNoSQLDB	--	Lite	Cloudant NoSQL DB
cloud-object-storage-WDP	MariaResources	Lite	Cloud Object Storage
Discovery-lab2	--	Lite	Discovery
lab2	--	Lite	Natural Language Understanding

Nos aparecerá una alerta preguntándonos si queremos volver a transferir y le decimos que sí. Este proceso tardará unos minutos, ya que IBM Cloud va a parar nuestra aplicación, configurar lo necesario para poder hacer uso de Conversation desde Node-RED y volver a arrancarla.



Esperamos de nuevo que nuestra aplicación se esté ejecutando. Podemos ver el estado de despliegue en la misma página y una vez arrancada podemos acceder a NodeRED desde el link de [Visitar URL de app](#)

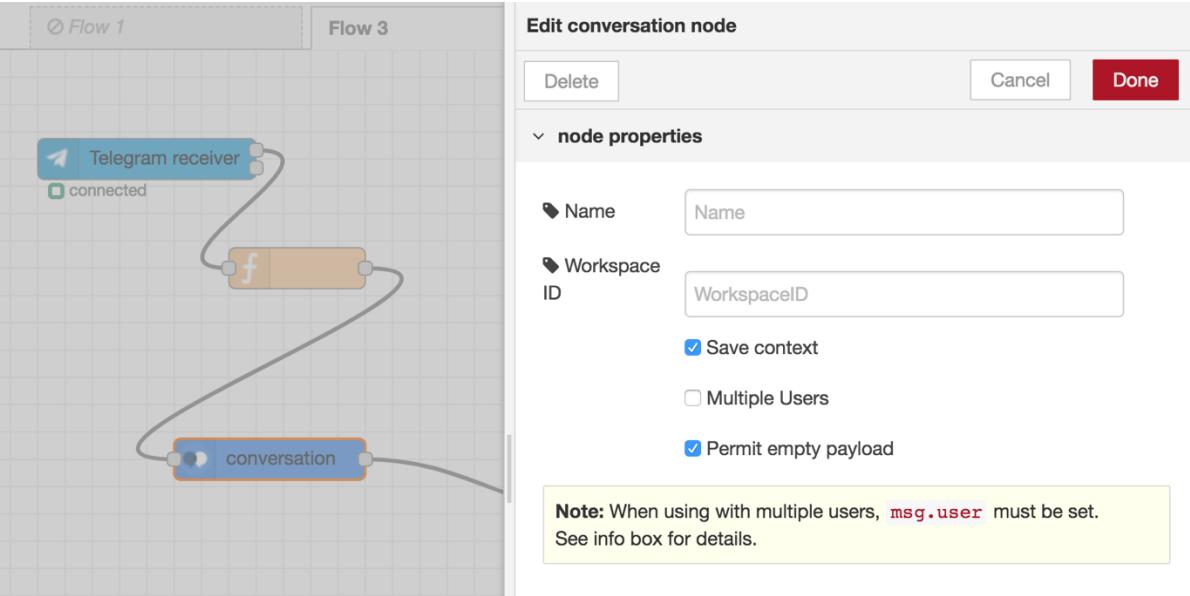


Apps de Cloud Foundry /

cognitivebreakcloud  Volviendo a transferir [Visitar URL de app](#) Rutas

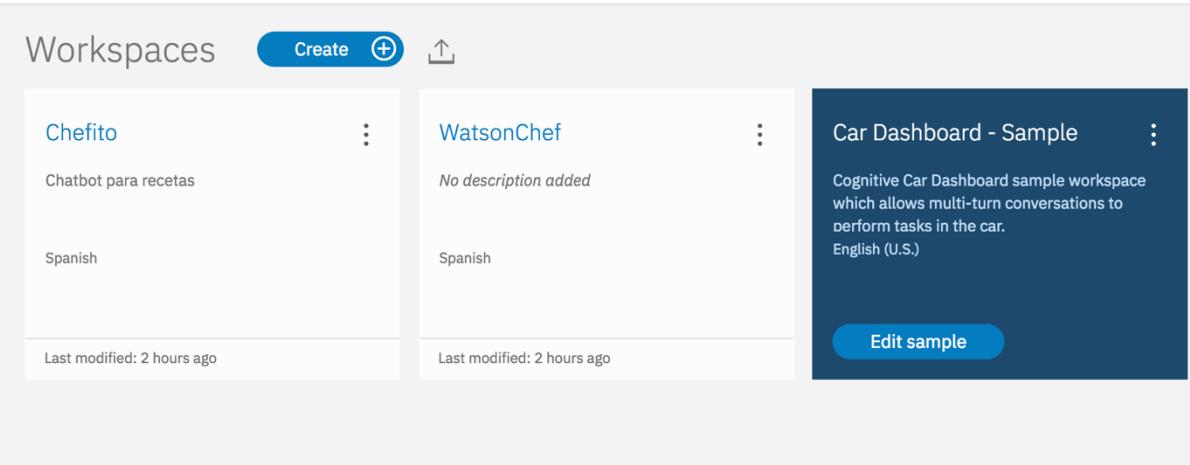
Org: maria.borbones@es.ibm.com Ubicación: Germany Espacio: cognitive

Cuando estemos de nuevo en nuestro Node-RED, hacemos doble click en el nodo de conversation para acceder a su configuración



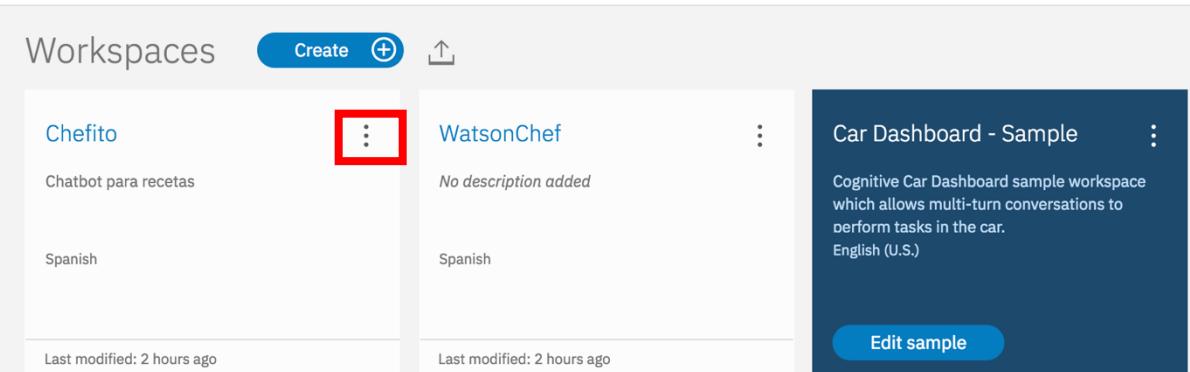
The screenshot shows the Watson Assistant interface. On the left, there's a flow diagram titled 'Flow 1' and 'Flow 3'. It includes nodes like 'Telegram receiver' (connected), a function node with an 'f' icon, and a 'conversation' node. A callout from the 'conversation' node points to the right, opening the 'Edit conversation node' dialog box. This dialog has tabs for 'Delete', 'Cancel', and 'Done'. Under 'node properties', there are fields for 'Name' (empty), 'Workspace ID' (empty), and checkboxes for 'Save context' (checked), 'Multiple Users' (unchecked), and 'Permit empty payload' (checked). A note at the bottom says: 'Note: When using with multiple users, `msg.user` must be set. See info box for details.'

Y simplemente lo que necesitamos es el Workspace ID de nuestra conversación. Para obtenerlo, volvemos al servicio de Conversation y en la página de bienvenida recogemos las credenciales.



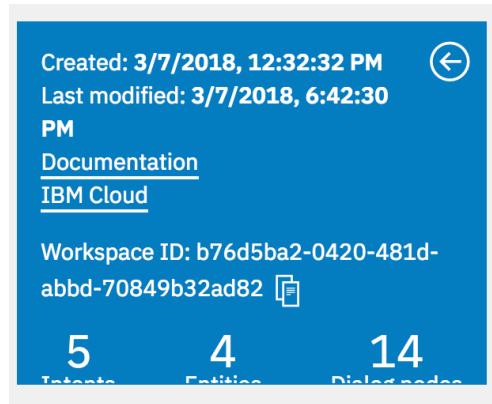
The screenshot shows the 'Workspaces' page. It lists three workspaces: 'Chefito' (Chatbot para recetas, Spanish, last modified 2 hours ago), 'WatsonChef' (No description added, Spanish, last modified 2 hours ago), and 'Car Dashboard - Sample' (Cognitive Car Dashboard sample workspace, English (U.S.), last modified 2 hours ago). There is a 'Create' button and an upward arrow icon at the top.

Hacemos click en los tres puntos del workspace que queremos configurar



This screenshot is similar to the previous one, showing the 'Workspaces' page. The 'Chefito' workspace now has a red box around its three-dot menu icon. The other workspaces ('WatsonChef' and 'Car Dashboard - Sample') also have their own three-dot menu icons.

Y en el desplegable hacemos click en View details para copiar el **Workspace ID**



Volvemos a Node-RED y pegamos el identificador en el campo workspace ID del nodo de Conversation

Hacemos click en **Done** para guardar los cambios. Y en el editor hacemos click en **Deploy** para guardar los cambios y desplegar el flujo.

¡Y ha llegado el momento más esperado! Vamos a probar nuestro bot. Así que volvemos a telegram y buscamos nuestro bot por su nombre



Y le escribimos hola para comprobar que funciona correctamente:

CO cognitivebot 20:36:52

¡Hola! Bienvenid@ al fántastico mundo de las recetas culinarias 🍎  
🍍🍉🍍 Soy Chefito 🍔🍕🍔, y estoy aquí para ayudarte. ¿Qué te gustaría cocinar hoy?

Escribe un mensaje... 

         ENVIAR

*¡Enhorabuena! Has completado el laboratorio y ya eres todo un experto en chatbots*

