

Watson APIs

Agenda

1. Pre-requisitos
2. Asistente Virtual para una Tienda Departamental.....4
3. Dish Analyzer App.....18
4. Reclutamiento inteligente.....29
5. Análisis de reviews de un e-commerce.....32

Pre-requisitos

- Cuenta de IBM Cloud
- Editor de código(<https://code.visualstudio.com/download>)
- Node.js (<https://nodejs.org/en>)

Watson Assistant

1. Prework:

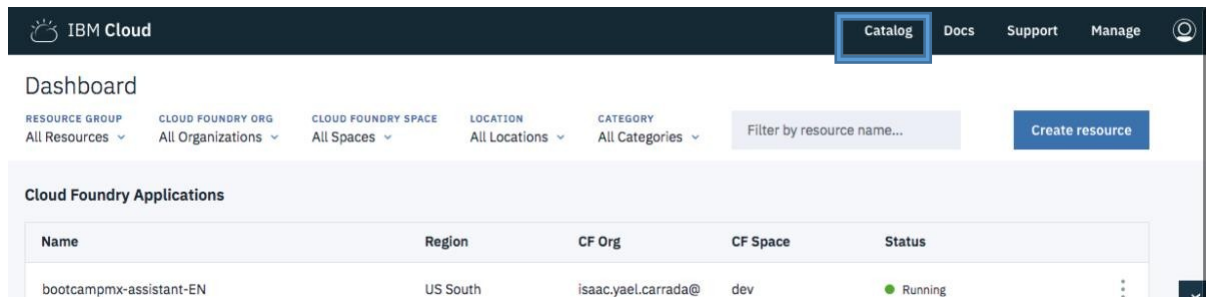
- 1.1 Descargar e instalar Node js
- 1.2 Descargar e instalar su editor de texto favorito
- 1.3 Descargar este proyecto: <https://github.com/ibmdbgmx/bootcamp-asistant>

2.1 Try out

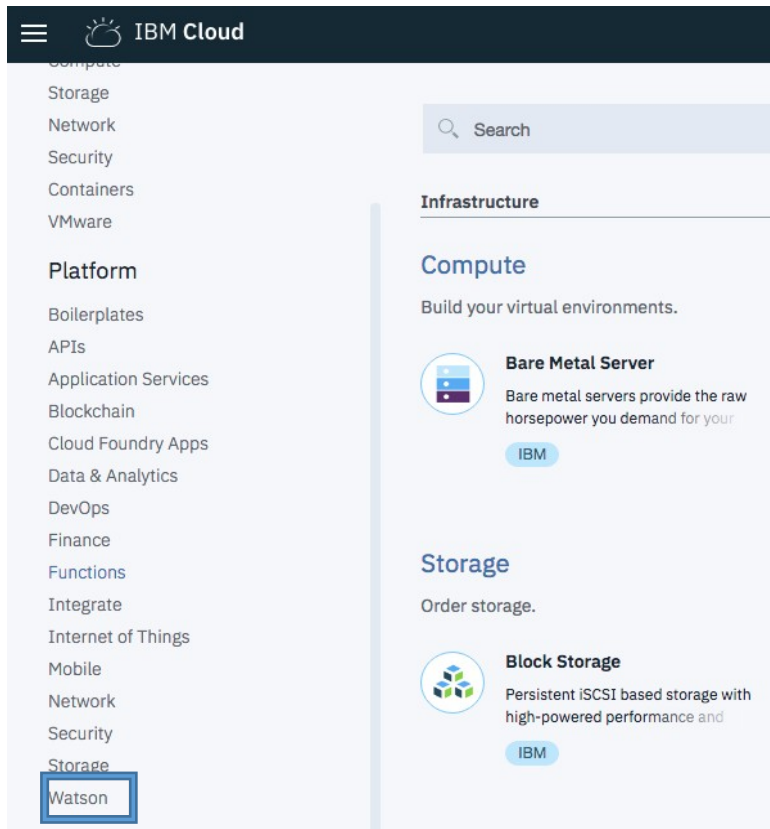
- 1. Ingresar a la URL: bootcamp-asistant-thankful-quoll.mybluemix.net
- 2. Explicación de la aplicación

3.1 Creación de servicios.

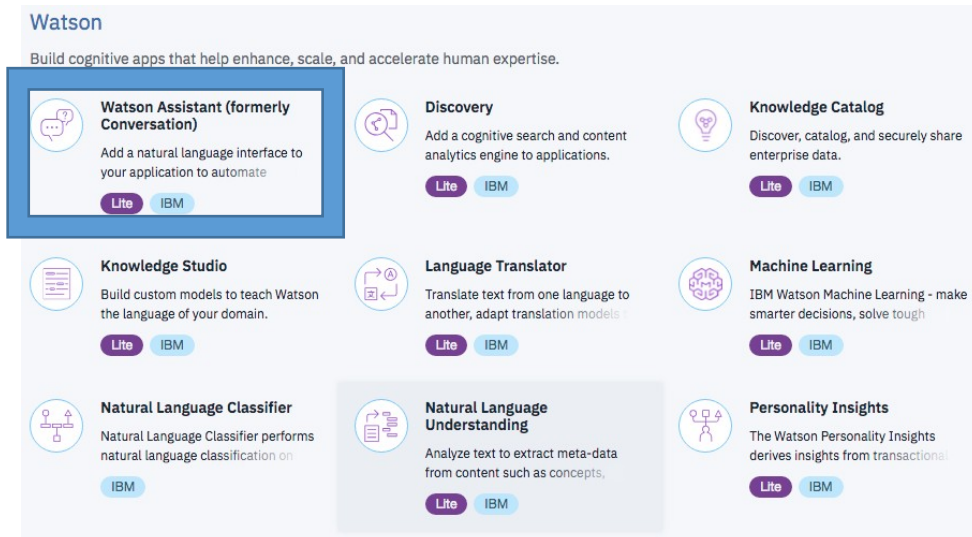
- 1. Entrar al portal de IBM Cloud (<http://console.bluemix.net>)
- 2. Ir al catalogo



- 3. Seleccionar Watson del menú en la categoría platform



4. Seleccionar Assistant



5. Nombrar app y crear servicio

View all

Watson Assistant (formerly Conversation)

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. Train Watson Assistant service through an easy-to-use web application, designed so you can quickly build natural conversation flows between your apps and users, and deploy scalable, cost effective solutions.

Service name:
Watson Assistant (formerly Conversation)-oc

Choose a region/location to deploy in:
US South

Choose an organization:
isaac.yael.carrada@ibm.com

Choose a space:
dev

Images

Click an image to enlarge and view screen captures, slides, or videos. Screen caps show the user interface for the service after it has been provisioned.

[View Docs](#) [Terms](#)

AUTHOR IBM
PUBLISHED 05/22/2018

[Need Help?](#)
[Contact IBM Cloud Sales](#)

[Estimate Monthly Cost](#)
[Cost Calculator](#)

[Create](#)

- Una vez que se haya abierto el servicio nos vamos a la opcion que dice credenciales (en caso de que no nos lleve directamente al servicio una vez creado dar click en el ícono IBM Cloud)

Manage

Service credentials

Plan

Connections

Watson / Assistant : Marina 1.86% Used | 9814 Api calls available [Details](#)

Location: US South Org: isaac.yael.carrada@ibm.com Space: dev

[Get started with the console](#) [Plan free Upgrade](#)

- En caso de ya tener creadas credenciales hacer click en **view credentials**. En caso contrario hacer click en **New credential**

Manage

Service credentials

Connections

Watson / Assistant : Marina 1.86% Used | 9814 Api calls available [Details](#)

Location: US South Org: isaac.yael.carrada@ibm.com Space: dev

Service credentials

Credentials are provided in JSON format. The JSON snippet lists credentials, such as the API key and secret, as well as connection information for the service. [Learn More](#)

Service credentials

[New credential](#)

10 Items per page | 1-1 of 1 items 1 of 1 pages < 1 >

KEY NAME	DATE CREATED	ACTIONS
<input type="checkbox"/> conversation_tooling_key1519939857718	Mar 1, 2018 - 04:31:06	View credentials

8. Copiamos estas credenciales y las guardamos (en mi caso en Notas)

```
June 25, 2018 at 4:28 PM

{
  "url": "https://gateway.watsonplatform.net/conversation/
api",
  "username": "7b90f738-683e-4179-83e2-804b032b9af1",
  "password": "aKiXcFvjkbMt"
}
```

Regresamos a la pestaña que se llama Manage y damos click en **Launch Tool** para entrenar nuestro servicio

Watson / Assistant : Marina 1.86% Used | 9814 Api calls available [Details](#)

Location: US South Org: isaac.yael.carrada@ibm.com Space: dev

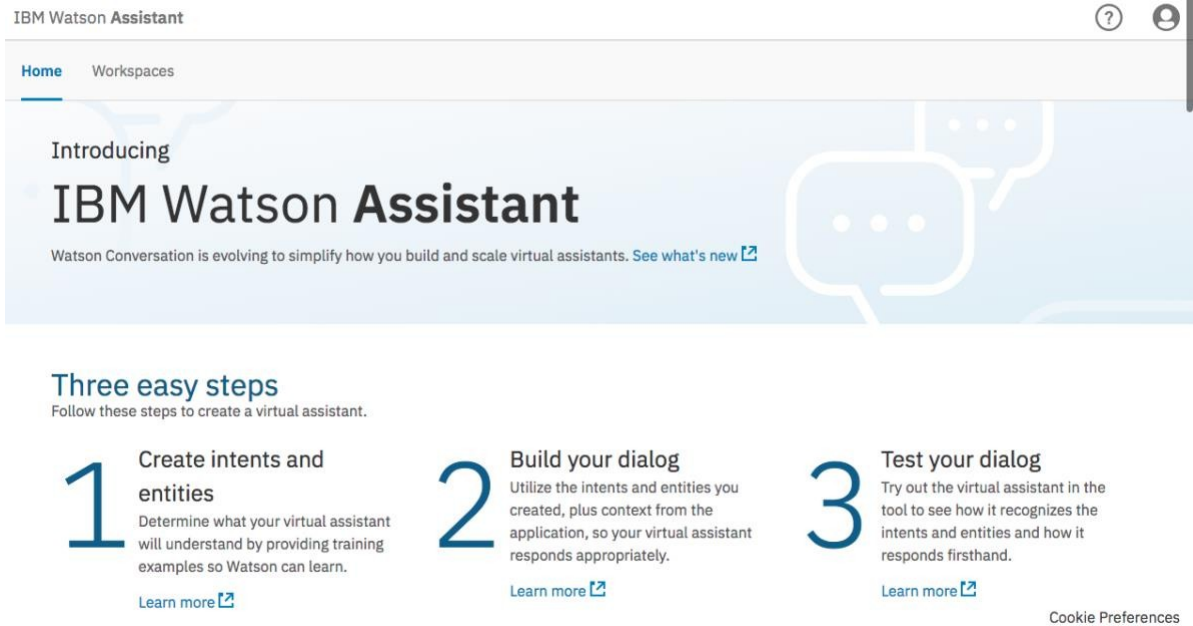
Get started with the service. Plan: free [Upgrade](#)

Launch tool [Getting started tutorial](#) [API reference](#)

Credentials [Show](#) [Configure credentials](#)

```
{
  "url": "https://gateway.watsonplatform.net/conversation/api",
  "username": ".....",
  "password": "....."
}
```

9. Una vez que hayamos lanzado la herramienta veremos una landing page muy parecida a esta:



10. Nos movemos a la ventana llamada **Workspaces**. Cada workspace representa a una instancia de un chatbot diferente.

11. Para crear o entrenar un chatbot utilizaremos los siguientes botones:



12. Creamos un nuevo workspace y le damos el nombre, descripción e idioma que queramos

×

Create a workspace

Workspaces enable you to maintain separate intents, user examples, entities, and dialogs for each use or application.

Name

Katy

Description

Chatbot asistente de seguros

Language

Spanish ▼

Create

13. Nos vamos a la pestaña Content Catalog

Category	Description	Intents	
Bancario	Transacciones básicas para un caso de uso bancario.	13	+ Add to workspace
Control del Bot	Funciones que permiten la navegación dentro de una conversación.	9	+ Add to workspace
Atención al cliente	Comprender y ayudar a los clientes con información sobre ellos y su negocio.	18	+ Add to workspace
Comercio electrónico	Pago, facturación y tareas básicas de administración para pedidos.	14	+ Add to workspace
General	Temas generales de conversación que la mayoría de los usuarios preguntan.	10	+ Add to workspace
Telecomunicaciones	Preguntas y problemas relacionados con el servicio, dispositivo y plan de telefonía de un usuario.	21	+ Add to workspace
Utilidades	Ayuda a un usuario con emergencias de servicios públicos y su servicio de utilidad.	10	+ Add to workspace

14. Dentro de esta pestaña podemos encontrar varios Intents ya entrenados, seleccionamos **atención al cliente** dándole click a **Add to workspace**

15. Regresamos a los intents

16. Navegamos por los intents para ver qué tipo de preguntas son las que podemos hacer

17. Creamos un nuevo intent que compete un producto que querramos comprar, para esto damos click en **Add Intent**

Intent name: #comprarProducto

Description: comprar un producto

Add user examples

Add user examples to this intent

Add example

☐ User examples (10) ▼

- ☐ Quiero comprar una cámara
- ☐ Quiero comprar un telefono
- ☐ Tienes camaras de seguridad?
- ☐ Tienes discos duros?


Llenamos este Intent con enunciados de cómo nuestro usuario final preguntaría respecto a nuestros productos (tienda de equipos de tecnología)

18. Pasamos a nuestra tab **Entities**. En esta creamos una entidad dando click en el botón **Add entity**

[Workspaces](#) / Katy / Build

Intents **Entities** Dialog Content Catalog

My entities System entities



No entities yet.

An entity is a portion of the user's input that you can use to provide a different response to a particular intent. Adding values and synonyms to entities helps your virtual assistant learn and understand important details that your users mention.

[Import](#) [Add entity](#)

19. Creamos una entity para los productos que podemos comprar

[←](#) | @producto

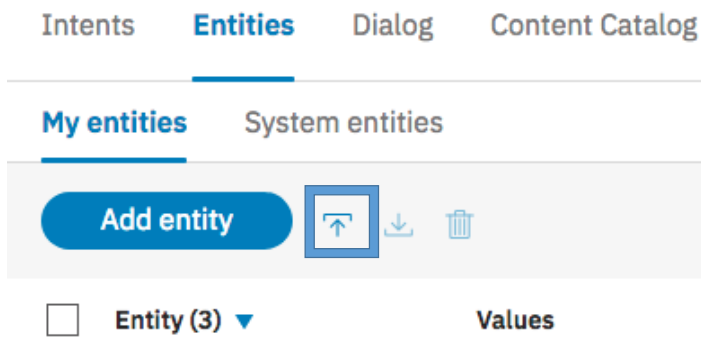
Entity name
@producto

Value name
Enter value [Add value](#)

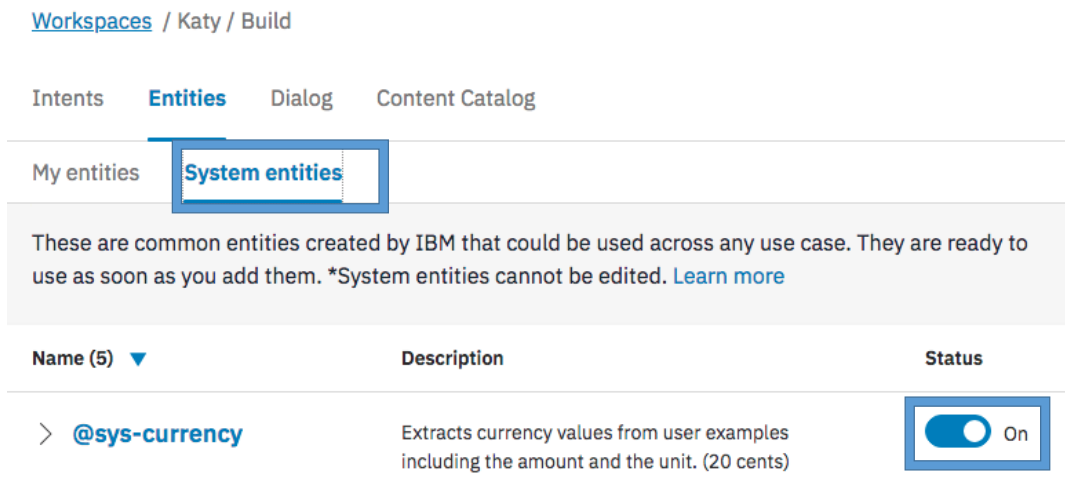
Synonyms [Add synonym...](#) [+](#)

<input type="checkbox"/> Entity values (9) ▼	Type
<input type="checkbox"/> antena	Synonyms
<input type="checkbox"/> camara	Synonyms
<input type="checkbox"/> computadora	Synonyms
<input type="checkbox"/> conector	Synonyms
<input type="checkbox"/> dron	Synonyms
<input type="checkbox"/> pantalla	Synonyms

20. De igual forma importamos los archivos que viene, en la misma carpeta training de nombre **specaudifonos.csv** y **tamañoaudifonos.csv**



21. Nos vamos a system entities y activamos **@sys_currency**

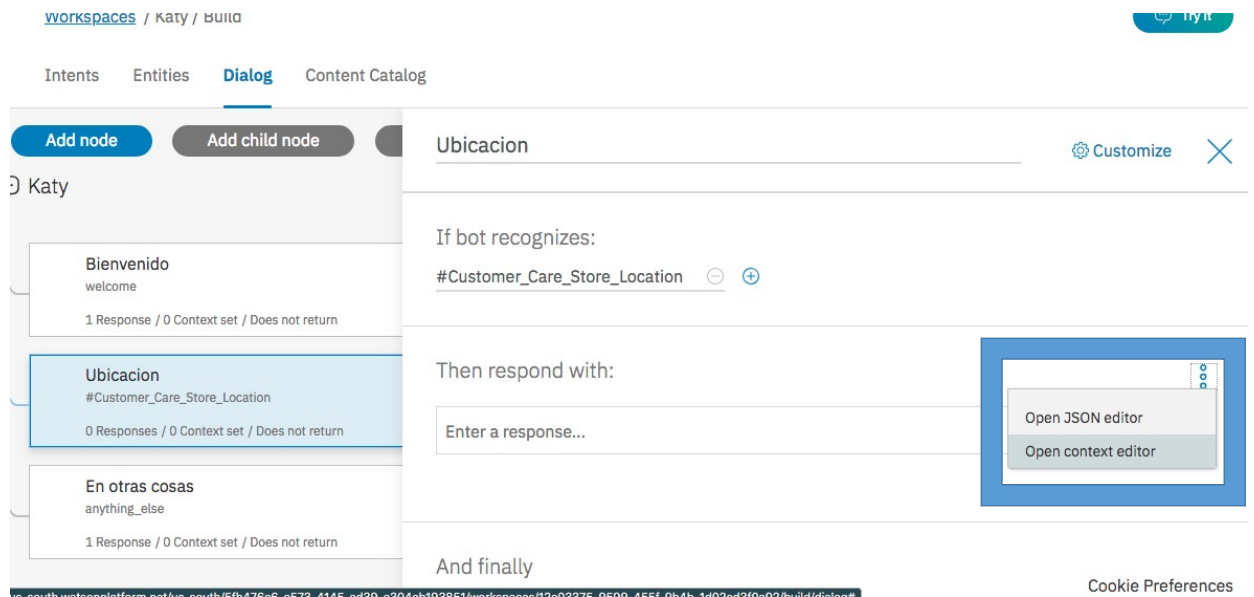


22. Posteriormente nos pasamos a nuestra ventana **Diálogo** y damos click en **Create Dialogo**:

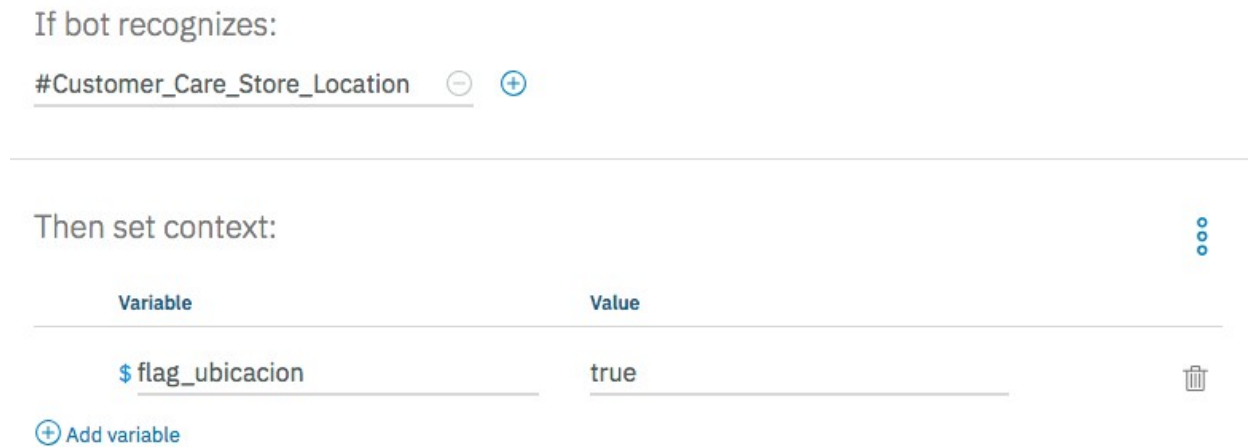
23. Damos click al nodo creado y le ponemos el nombre de “Ubicación”

24. En el condicional seleccionamos el intent **#Customer_Care_Store_Location**

25. Abrimos el editor de contexto



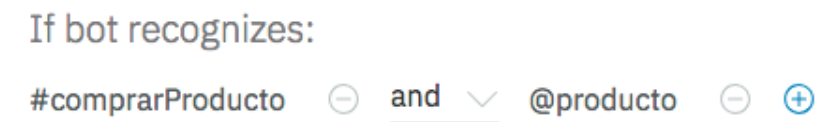
26. En este ponemos la variable de contexto \$bandera_ubicacion y el valor True



27. Damos una respuesta “Le envío la ubicación”

28. Creamos un nodo más, esta vez que se llame Comprar producto


29. Seleccionamos el intent comprarProducto y la entidad producto



30. Seleccionamos el boton customize y activamos **Multiple Responses** y damos click en Apply

Customize "Comprar producto"


[Customize node](#)
[Digressions](#)

Slots ⓘ 

Enable this to gather the information your virtual assistant needs to respond to a user within a single node.

☐ Prompt for everything

Enable this to ask for multiple pieces of information in a single prompt, so your user can provide them all at once and not be prompted for them one at a time.

Multiple responses ⓘ 

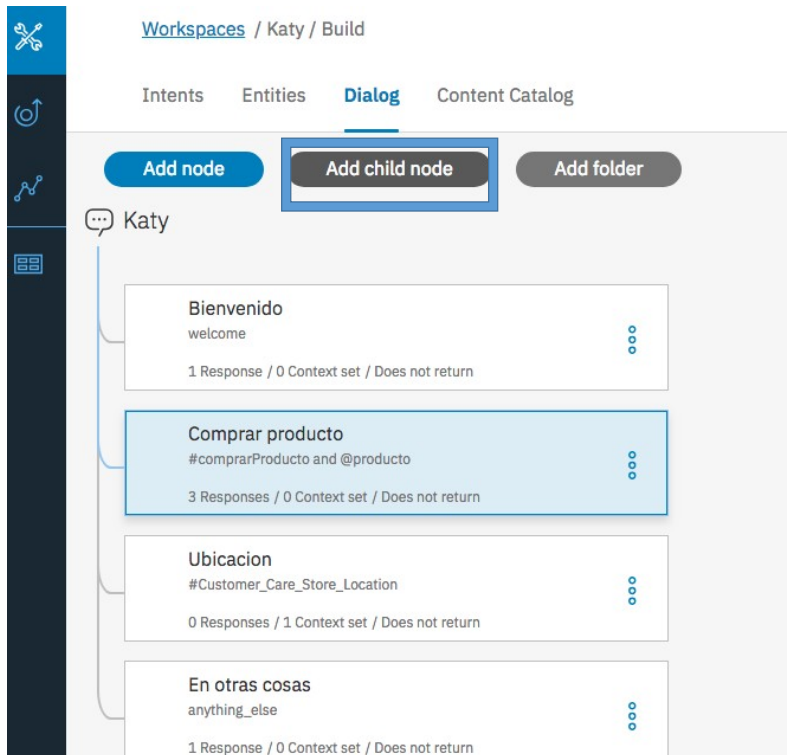
Enable multiple responses so that your virtual assistant can provide different responses to the same input, based on other conditions.

[Cancel](#)
[Apply](#)

31. De lado izquierdo seleccionamos producto, colocamos un ":" y después de esto seleccionamos a lo que tiene que ser igual el producto:

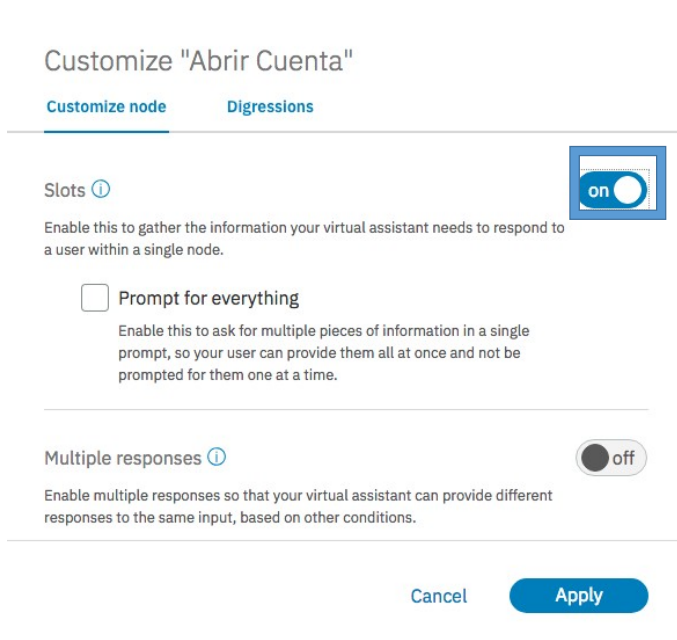
	If bot recognizes	Respond with
1	@producto:resistencia	La resistencia cuesta
2	@producto:camara	La camara cuesta
3	@producto:usb	El @producto cuesta

32. Añadimos un nodo hijo de este nodo, de la siguiente forma:



33. Dentro de este nuevo nodo le ponemos el nombre **@producto:audifonos**. De la misma forma poniendo como condición que se haya seleccionado la entidad producto con el valor de cámara

34. Seleccionamos customizar y seleccionamos **slots**



35. Una vez que ya tenemos la funcionalidad de los slots los llenamos de la siguiente manera:



If bot recognizes:

@producto:audifonos

Then check for:

Manage handlers

	Check for	Save it as	If not present, ask	Type		
1	@sys-currency	\$currency	¿De que precio quien	Required		
2	@tamañoAudifonos	\$tipo	¿De que tipo? ¿On ea	Required		
3	@specAudifonos	\$spec	¿Con cable o inalámte	Required		

36. Para la respuesta:

Then respond with:

1. Perfecto, en breve te diré que audífonos cuestan \$currency, son \$tipo y \$spec

Add a variation to this response

37. Una vez que tengamos este nodo nos vamos al padre (comprar producto) y cambiamos la respuesta de **Wait for user input** a **Jump to** y seleccionamos el hijo y después **If bot recognizes condition**:

Select a destination node (origin: Comprar producto)

bienvenido
welcome
1 Response / 0 Context set / Does not return

Comprar producto
#comprarProducto and @producto
2 Responses / 0 Context set / Does not return

Audifonos
@producto:audifonos
1 Response / 3 Context set / 3 S

Ubicacion
#Customer_Care_Store_Location
0 Responses / 1 Context set / Does not return

And finally

Wait for user input

Wait for user input

Skip user input

Jump to...

Jump to and...

Wait for user input

If bot recognizes (condition)

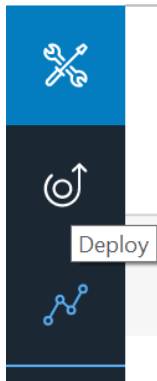
Respond

nt-us-south.watsonplatform.net/us-south/5fb476c6-c573-4145-ad39-e304eb193851/workspaces/72c03375-9599-455f-9b4b-1d02ed3f0e92/build/dialog#

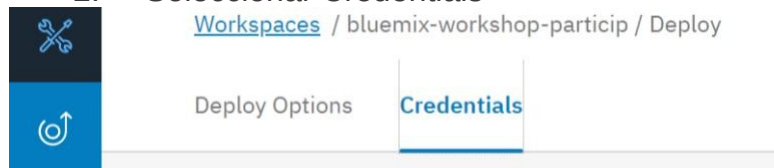
38. Una vez hecho esto ya podemos pasar a la siguiente sección

4 Configurar app nodejs con workspace

1. Hacer click en el ícono Deploy



2. Seleccionar Credentials



3. Aquí podemos encontrar el WorkspaceID, Username y Password, que utilizaremos en el siguiente paso

4. En el archivo app.js, para las variables workspaceID, username y password utilizar estos valores

```
10
11 var wconv_version_date = '2018-02-16';
12 var wconv_api_version = 'v1';
13 var wconv_workspaceId = 'reemplazar_con_ID_workspace_watson_conversation';
14 var wconv_username = 'reemplazar_con_usuario_watson_conversation';
15 var wconv_password = 'reemplazar_con_contraseña_watson_conversation';
16
```

5.5 Correr localmente

1. Abrir la línea de comandos
2. Ir al directorio en el que se encuentra nuestro proyecto
3. Ingresar: npm install
4. Ingresar: node "app.js"
5. Corre en localhost

5.6 Variables de contexto

2. La parte del código en la que está configurada la dirección del mapa es:

```
var additionalText = "<br><iframe " +
  "src=\"https://www.google.com/maps/embed?pb=!1m18!1m12!1m3!1d482173.93142824515!2d-99.39010785\" +
  "856533!3d19.23953607409557!2m3!1f0!2f0!3f0!3m2!1i1024!2i768!4f13.1!3m3!1m2!1s0x85d200dc6d9c3dbb\" +
  "%3A0xe146f685b596d5c7!2sIBM!5e0!3m2!1sen!2smx!4v1521645985106\" +
  "width=\\\"700\\\" height=\\\"400\\\" frameborder=\\\"0\\\"
  \"style=\\\"border:0\\\" allowfullscreen></iframe>\";
delete response.context.bandera_ubicacion;
callback(res, username, message, response, additionalText)
```

3. La función de respuesta con la bandera ubicación se encuentra definida en la siguiente parte


```
    }, function(err, response) {  
        ..... session.context = response.context;  
        .....  
        ..... var additionalText = "";  
        ..... if(response.context.bandera_ubicacion !== undefined)
```

5.7 Correr localmente de nuevo para notar el cambio

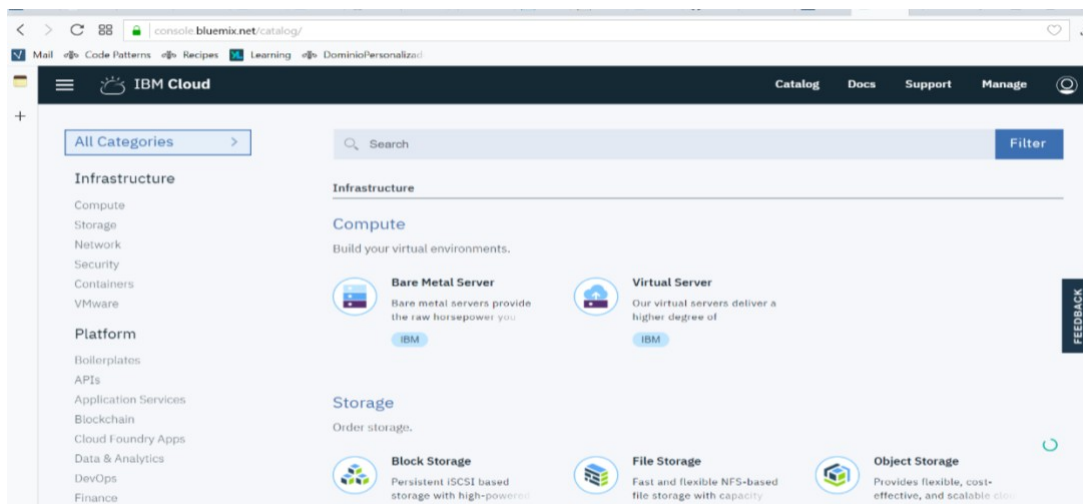
Visual Recognition

Pre-requisitos

- Servicio de Watson Visual Recognition
- Imágenes de entrenamiento(incluidas en el repositorio)
- Clonar Repositorio de Github (<https://github.com/ibmdbgmx/bootcampMx-VisualRecognition>)

1.Crear un Servicios

Crearemos los servicios de IBM

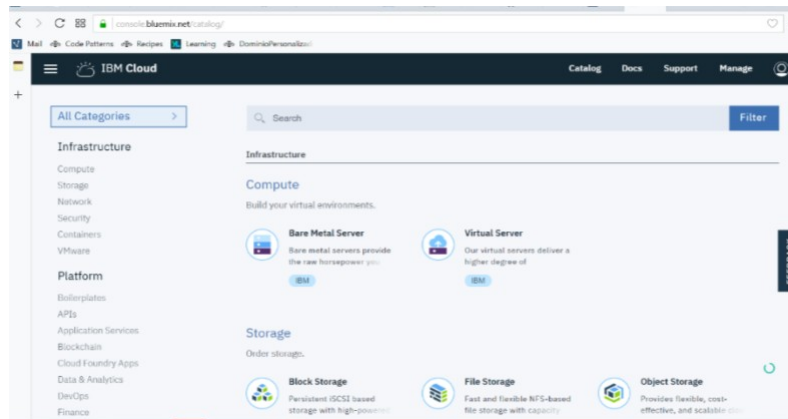


En la búsqueda ingresamos “Visual Recognition” y lo seleccionamos

- 1.-Hacer click en el botón “crear”



Debemos repetir el proceso con el siguiente servicio.



En la búsqueda ingresamos “Cloud Object Storage” y lo seleccionamos



2.-Dirijirse a “Herramienta de lanzamiento” en el servicio de Visual Recognition

Herramienta de lanzamiento

2.Crear modelo

Dentro del panel de Visual Recognition, realizar los siguientes pasos:

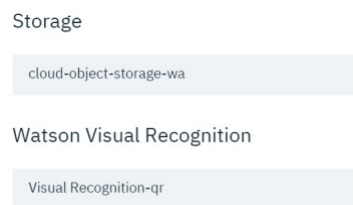
1. Hacemos clic en “Create Model”



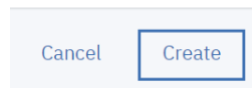
2. Ingresamos un nombre.



3. Seleccionamos los servicios anteriormente creados.



4. Hacemos clic en el botón “create”



a. Agregar archivos al proyecto y entrenamiento

1. Cambiamos el nombre del Modelo (ej. “Food Dish”)
2. Descargue los archivos .zip (NiceFood y UglyFood)del repositorio

3. Agregar los archivos .Zip(NiceFood y UglyFood) en el área de “upload to Project”

1. Upload to project

To add files to your project,
drop .zip files here or

[Browse](#)

2. Add from project

Drag .zip files from your project to the
training area to add them to your model.

0 selected

There are no .zip files in your project.

4. Seleccionamos ambos archivos y los agregamos al modelo

2 selected [Add to model](#)

<input checked="" type="checkbox"/>	NiceFood.zip 11 Jun 2018, 10:33:55 am 219.1 KB
<input checked="" type="checkbox"/>	UglyFood.zip 11 Jun 2018, 10:33:54 am 133.81 KB

5. Una vez listo, podremos entrenar al modelo

✓ Model is ready to train.

[Train Model](#) ⓘ

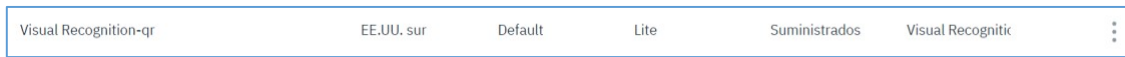
6. Una vez finalizado el entrenamiento podremos probarlo con el archivo “uglyFoodtest.jpeg” dentro de la carpeta del proyecto.

✓ **Training successful** Your model training is successful. Click [here](#) to view and test your model.

Obtenga las credenciales del servicio

Abra una nueva pestaña en su navegador para acceder a la nube de IBM en:
<https://bluemix.net>

1. En el panel de control, haga clic en el servicio de Reconocimiento Visual (antes creado)



2. Haga clic en Credenciales del servicio en el menú del lado izquierdo
3. Haga clic en ver credenciales
4. Copia las credenciales para que podamos usarlas más tarde. Las credenciales deben verse así:

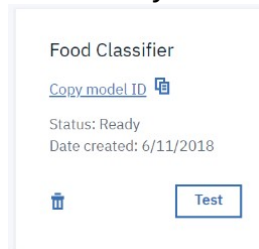
```
{
  "apikey": "iVgsY-vQXk2yHmF_AKge3IVAtb_PvqZeUeAQppjyq6BY",
  "iam_apikey_description": "Auto generated apikey during resource-key operation for Instance - crn:v1:bluemix:public:watson-vision-combined:us-south:a/f8aeed1e1657afae00ed1530658d8202:15122523-3a1c-4380-93eb-4ecee95123f9::",
  "iam_apikey_name": "auto-generated-apikey-301f7da2-3197-4939-b853-8161bb2a128"
```

Ahora puede cerrar esta pestaña del navegador

Puede probar las otras dos imágenes borrando el resultado e intente de nuevo. Más imágenes por clase mejoraría la precisión.

Probando la app

1. Configurar las credenciales en el archivo “app.js”
 - a. Copiamos el “apikey” en la variable “iam_apikey” de las credenciales de Visual Recognition.
 - b. Copiamos el model ID del modelo creado y lo sustituimos en la variable “classifier_ids”



2. Abrir terminal y correr el siguiente comando (dentro de la carpeta del proyecto)

node app

3. Dirigirse al localhost en el puerto “8080 ”
4. Selecciona un archivo

Dish Analyzer



5. Clic en el botón “Analyze Custom Classifier” para utilizar el modelo que acabamos de crear
6. Hacer clic en el botón “Analyze genera classifier” para utilizar el clasificador por defecto de Watson Visual Recognition

Visual recognition 2: Usando Clases negativas y las imágenes correctas.

Dentro del panel de Visual Recognition, realizar los siguientes pasos:

5. Hacemos clic en “Create Model”



6. Ingresamos un nombre.

Define project details

Name

AquíIngresarNombre

82

7. Seleccionamos los servicios anteriormente creados.

Storage

cloud-object-storage-wa

Watson Visual Recognition

Visual Recognition-qr

8. Hacemos clic en el botón “create”

Cancel Create

b. Agregar archivos al proyecto y entrenamiento

7. Cambiamos el nombre del Modelo (ej. “Manzanas”)

8. Descargue los archivos .zip (Manzanas y bananas)del repositorio

9. Agregar los archivos .Zip() en el área de “upload to Project”

1. Upload to project

To add files to your project,
drop .zip files here or

Browse

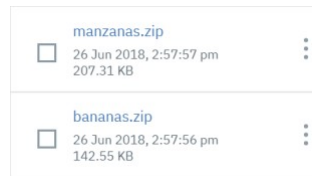
2. Add from project

Drag .zip files from your project to the
training area to add them to your model.

0 selected

There are no .zip files in your project.

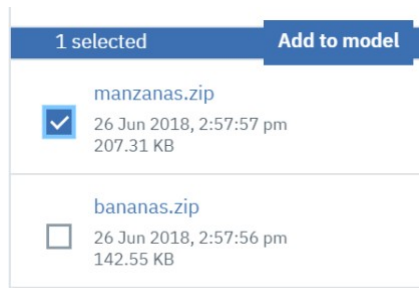
10. Seleccionamos ambos archivos y los agregamos al modelo



11. Arrastramos “bananas” hacia los negativos

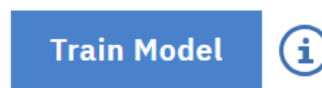


12. Seleccionamos Manzanas y le damos en “add to model”

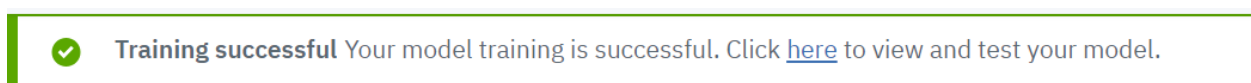


13. Una vez listo, podremos entrenar al modelo

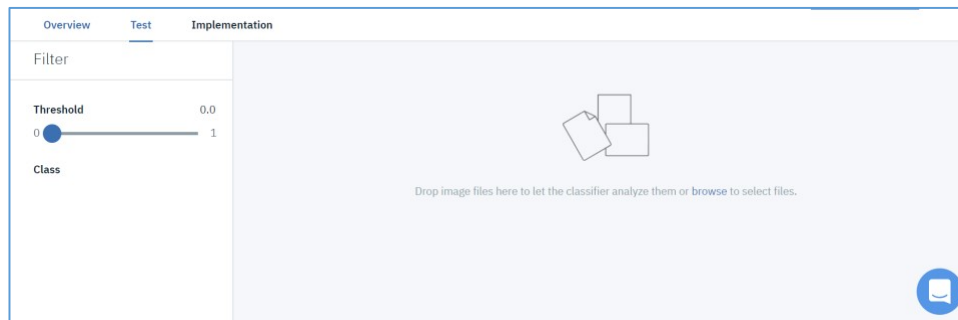
✓ Model is ready to train.



14. Una vez finalizado el entrenamiento podremos probarlo en el botón “here”.



15. Nos dirigimos a la pestaña de “test” y simplemente arrastramos una imagen.

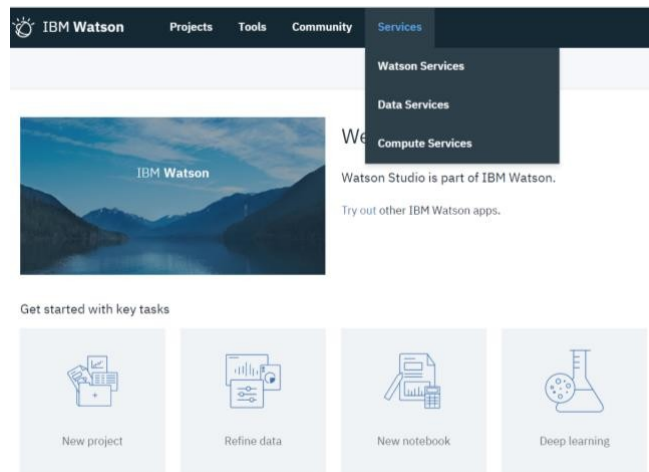


Visual recognition 3: Probando modelos beta

1. Ingresar a la plataforma:

<https://dataplatform.ibm.com>

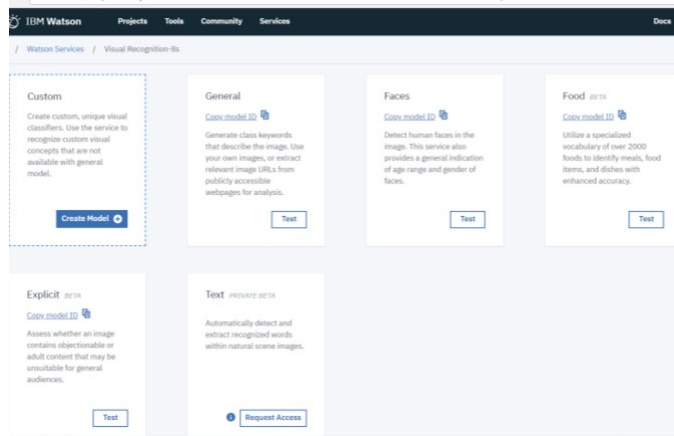
2. Nos dirigimos a la pestaña de “Services” y “Watson Services”



- En la sección de “Visual recognition “ seleccionamos el servicio que creamos en las partes anteriores y hacemos click en “Launch Tool”

Visual Recognition (4)		
NAME	TOOL	ACTIONS
Prueba#123ErrorVR	Launch tool	⋮
Visual Recognition-8s	Launch tool	⋮
Visual Recognition-qr	Launch tool	⋮
watson-vision-combined-r-watson-vis-1521576934869	Launch tool	⋮

- Probaremos cada una de las funciones beta de “Visual Recognition” con una imagen referente a



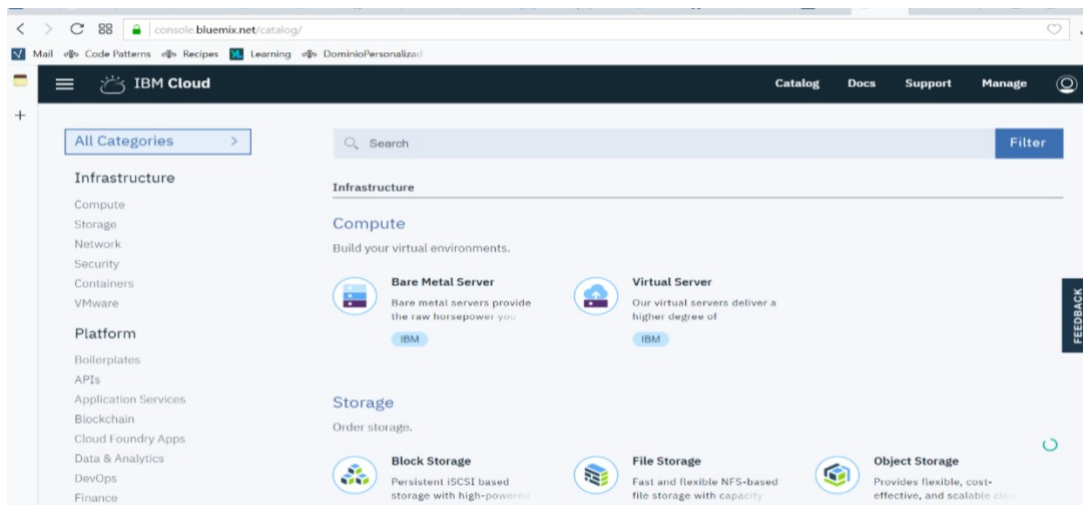
Personality Insights

Pre-requisitos

- Clonar Repositorio de Github(<https://github.com/ibmdbgmx/bootcampMx-PersonalityInsights>)

1.Crear un Servicios

Crearemos los servicios de IBM



En la búsqueda ingresamos “Personality insights ” y lo seleccionamos



1.-Hacer click en el botón “crear”



2.-Ingresar a la pestaña de “Service Credentials”

4.-Crear una credencial nueva



5.-configurar las credenciales en el archivo “app.js” (username y password)

```
username:
password:
```

6.-Abrir terminal y correr el siguiente comando

```
node app
```

7.- Dirigirse al localhost en el puerto “8080”

8.-Responder el cuestionario o utilizar texto de prueba en el archivo “ Respuestas.txt”, dentro del repositorio.

Un formulario de cuestionario con un fondo oscuro y texto blanco. El logo de Creative Commons está en la esquina superior izquierda. El formulario contiene siete preguntas con campos de entrada de texto correspondientes: "Captura tu nombre", "Captura tu edad", "Si pudieras tener un superpoder, cual seria y por que ?", "Que opinas de la importancia de la honestidad?", "Si pudieras regresar el tiempo, a que punto de tu vida lo harias ?", y "Cual ha sido tu mayor logro ?".

Personality Insights 2 : Entendiendo la respuesta

En el mismo repositorio, en la raíz, podemos encontrar un archivo llamado “response.js”

1.-Abrimos el archivo y lo corremos en la consola con el siguiente comando:

```
node response
```

En la consola podremos ver la respuesta completa del análisis de Personality insights, incluyendo algunas recomendaciones por parte del servicio.



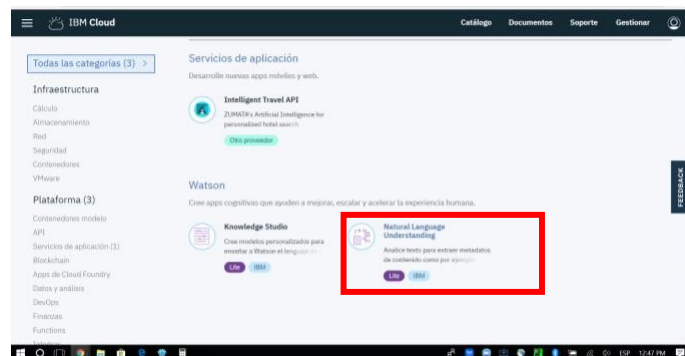
```
Command Prompt
{"score": 0.5
},
{
  "consumption_preference_id": "consumption_preferences_music_classical",
  "name": "Preferencias lectoras",
  "score": 1
}
},
{
  "consumption_preference_category_id": "consumption_preferences_reading",
  "name": "Preferencias lectoras",
  "consumption_preferences": [
    {
      "consumption_preference_id": "consumption_preferences_read_frequency",
      "name": "Es probable que lea con frecuencia",
      "score": 1
    },
    {
      "consumption_preference_id": "consumption_preferences_books_entertainment_magazines",
      "name": "Es probable que lea revistas de entretenimiento",
      "score": 1
    },
    {
      "consumption_preference_id": "consumption_preferences_books_non-fiction",
      "name": "Es probable que lea libros que no sean de ficción",
      "score": 1
    },
    {
      "consumption_preference_id": "consumption_preferences_books_financial_investing",
      "name": "Es probable que lea libros sobre inversión financiera",
      "score": 0
    },
    {
      "consumption_preference_id": "consumption_preferences_books_autobiographies",
      "name": "Es probable que lea libros autobiográficos",
      "score": 0
    }
  ]
},
{
  "consumption_preference_category_id": "consumption_preferences_volunteering",
  "name": "Preferencias de voluntariado",
  "consumption_preferences": [
    {
      "consumption_preference_id": "consumption_preferences_volunteer",
      "name": "Es probable que realice tareas de voluntario para obras sociales",
      "score": 0
    }
  ]
}
```

Natural Language Understanding

Pre-requisitos

- Links de productos de E-commerce(amazon.com)
 - o <https://amzn.to/2tIr2S8>
 - o <https://amzn.to/2IGBANa>
 - o <https://amzn.to/2KzIjom>
- Clonar Repositorio de Github(<https://github.com/ibmdbgmx/bootcampmx-NLU>)

1.Crear un Servicios



Crearemos los servicios de IBM

En la búsqueda ingresamos “Natural Language Understanding” y lo seleccionamos



1.-Hacer click en el botón “crear”



2.-Repetiremos el proceso de crear servicio con “cloudant”

3.-Ingresar a la pestaña de “Service Credentials” de cada uno de los servicios.

4.-Copiamos las credenciales de cada servicio.

a.En el caso de que **no** exista ninguna credencial haremos click en el botón “new credential”

New credential ⊕

5.-Pegar la URL de cloudant (credenciales de servicio) en el archivo “cloudant.js” dentro de la carpeta “lib”

```
var cloudantURL =
```

6.Pegar el nombre de usuario y la contraseña del servicio de NLU (credenciales de servicio) en el archivo “Watson.js” dentro de “lib”

```
'username':  
'password':
```

7.-Abrir terminal y correr el siguiente comando

```
node app
```

8.- Dirigirse al localhost en el puerto “4000”

