



Watson Natural Language Classifier

Hands-On Workshop

Laboratorio:

Sombrero seleccionador con Watson NLC

Tabla de Contenido

Introducción.....	3
Creación del clasificador Watson NLC.....	4
Utilización del clasificador NLC.....	9
Desarrollo de una aplicación Node-RED.....	9
Apéndice A: Versión mejorada de la aplicación Node-RED	15
Apéndice B: Despliegue de la aplicación mejorada en Java	19

Introducción

En este ejercicio crearemos:

- Un clasificador con el Watson Natural Language Classifier
- Una aplicación sencilla que lo utilice

Este ejercicio está basado en una idea de Ryan Anderson y consiste en crear un clasificador que simule el sombrero seleccionador que vimos en las películas y libros de Harry Potter.

Para aquellos que no las hayan visto o los hayan leído, el primer año en Hogwarts, el colegio de magia, los niños son asignados a uno de cuatro grupos o casas en función de sus cualidades. De esta asignación se encarga un sombrero parlante que el niño se pone en la cabeza. Las opciones posibles en función de esas cualidades son:

- Gryffindor (el valiente y audaz)
- Hufflepuff (el leal y confiable)
- Ravenclaw (el ingenioso)
- Slytherin (el ambicioso y a menudo desagradable)

Para entrenar al clasificador utilizaremos un documento CSV en el que aparece una colección de palabras (adjetivos y nombres) y su categorización en una o más clases. Cada casa de Hogwarts se corresponde con una clase del clasificador.

El CSV está basado en el creado por Ryan Anderson que ha sido traducido y adaptado al español.

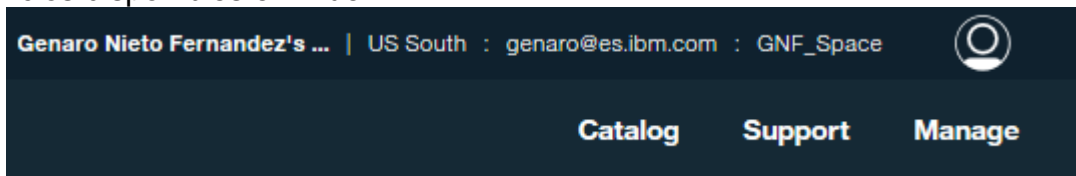
Creación del clasificador Watson NLC

En primer lugar vamos a crear una instancia del servicio Watson Natural Language Classifier.

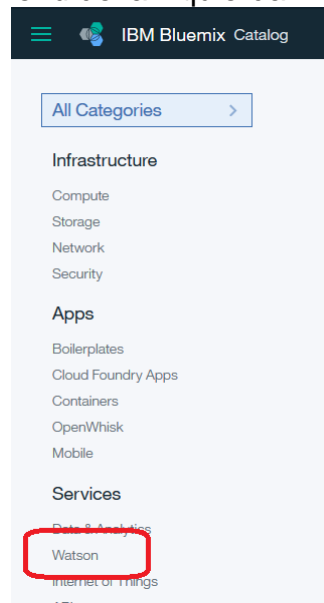
1. Navegamos a la página <http://bluemix.net>. Pulsa Log In e introduce tu usuario y contraseña.



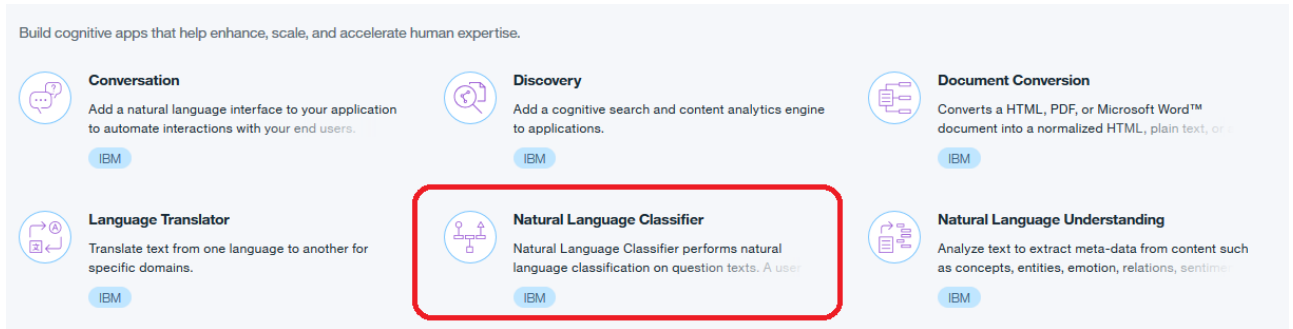
2. Pulsamos Catalog en la esquina superior derecha para acceder a la colección de servicios disponibles en Bluemix.



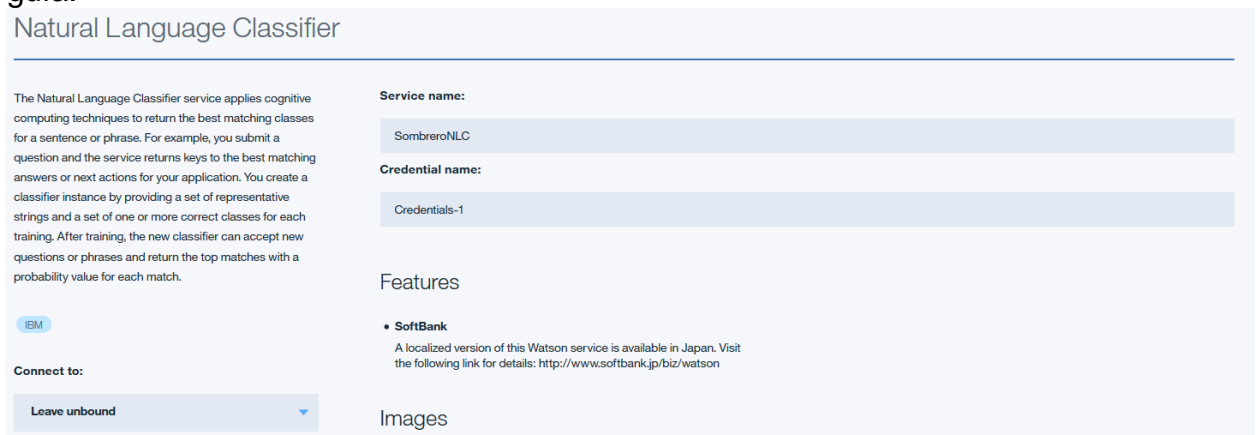
3. En el catálogo buscamos el servicio Natural Language Classifier dentro de la sección de servicios de Watson. La forma más rápida de encontrarlo es seleccionando Watson en el menú de la izquierda.



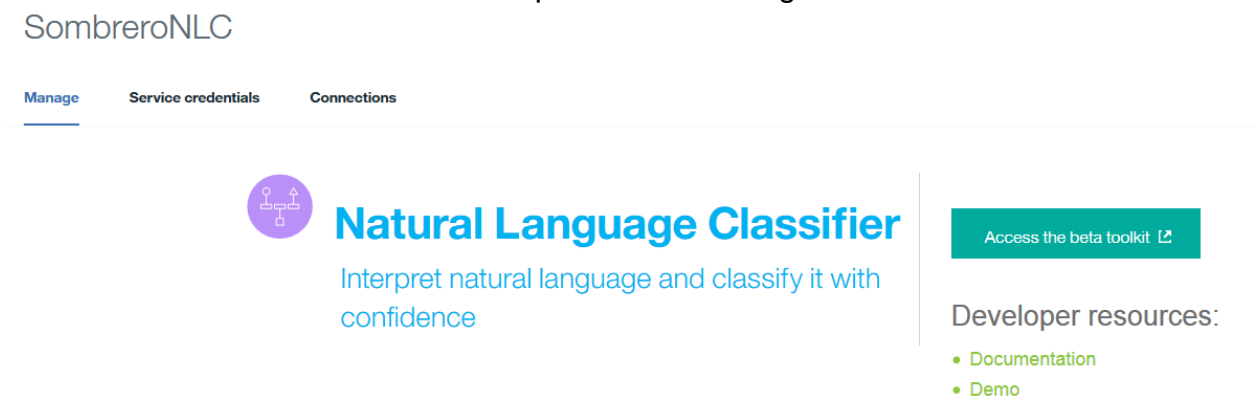
4. Una vez localizado lo seleccionamos.



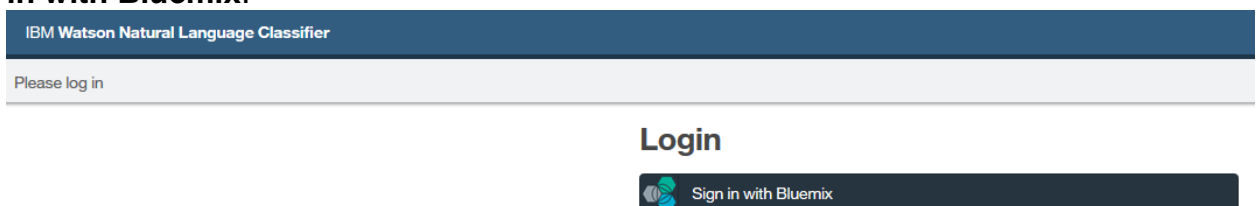
5. Damos un nombre a nuestra instancia del servicio o dejamos el que se nos proporciona por defecto. Pulsamos **Create** en la parte inferior derecha. **IMPORTANTE:** Debemos indicar un nombre único. No copiar el indicado en esta guía.



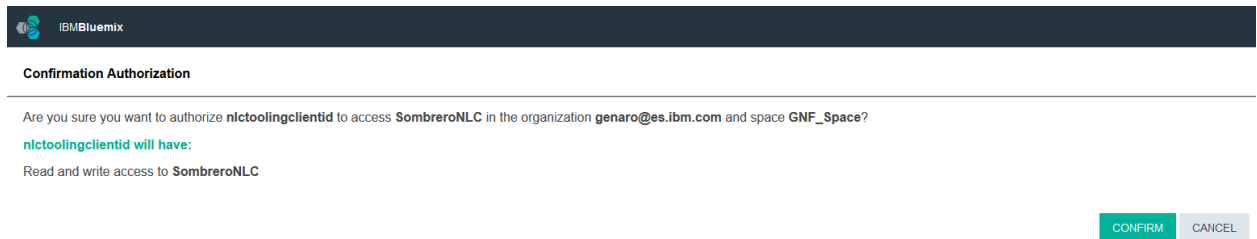
6. Nos aparece la página del servicio que acabamos de crear. Pulsamos **Access the beta toolkit** para utilizar una aplicación que nos facilita las operaciones con el servicio. Se nos abrirá en una nueva pestaña del navegador.



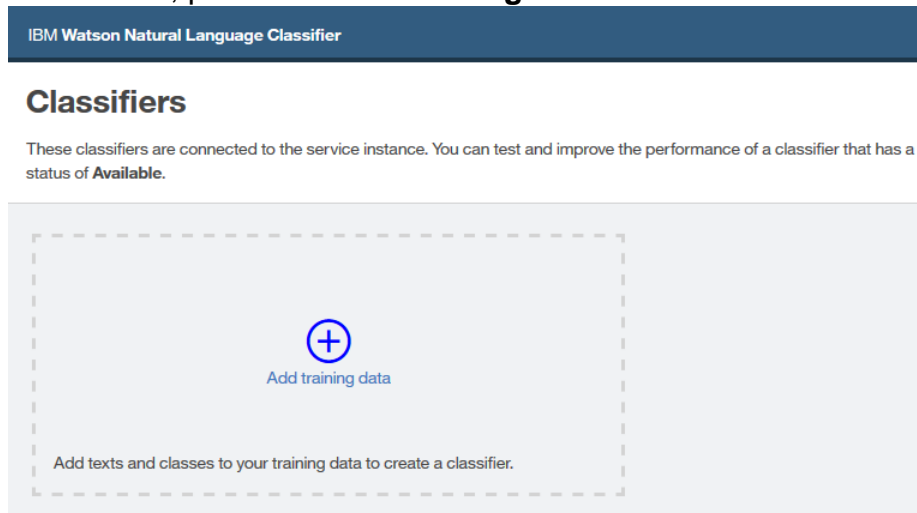
7. Necesitamos dar acceso a esta aplicación a nuestro servicio. Pulsamos sobre **Sign in with Bluemix**.



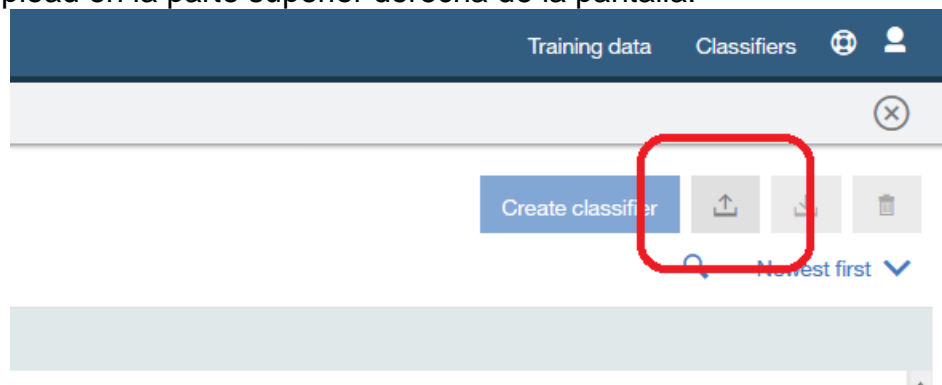
8. Y confirmamos el acceso.



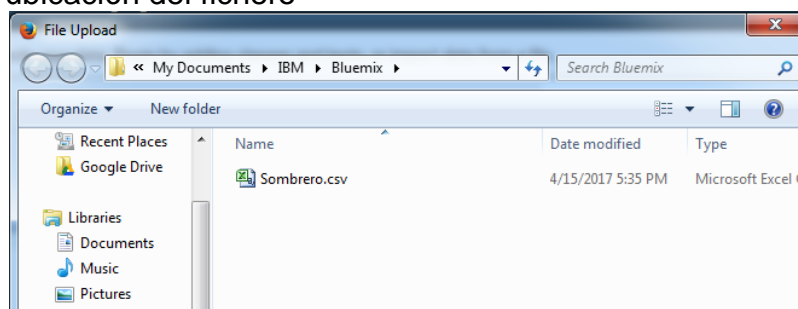
9. Ya en la herramienta, pulsamos **Add training data**.



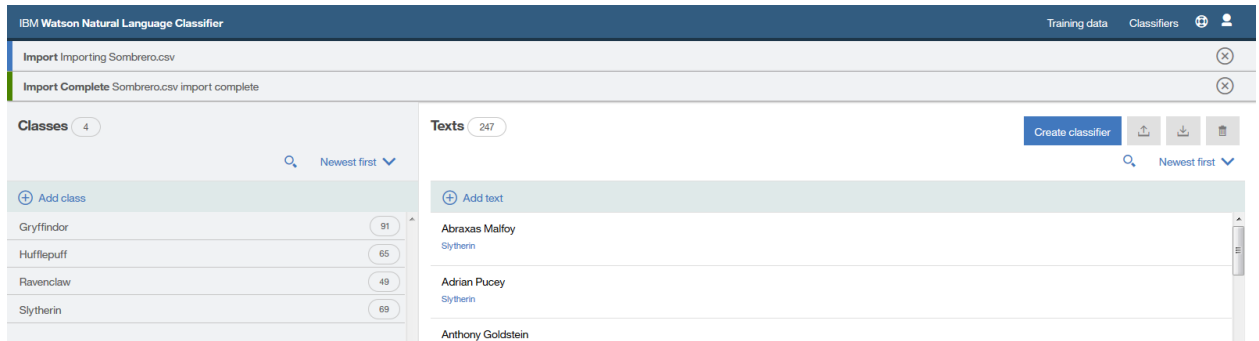
10. En la siguiente página, podemos introducir la información de forma manual. Pero nosotros vamos a utilizar el fichero CSV que ya tenemos preparado. Pulsamos el botón Upload en la parte superior derecha de la pantalla.



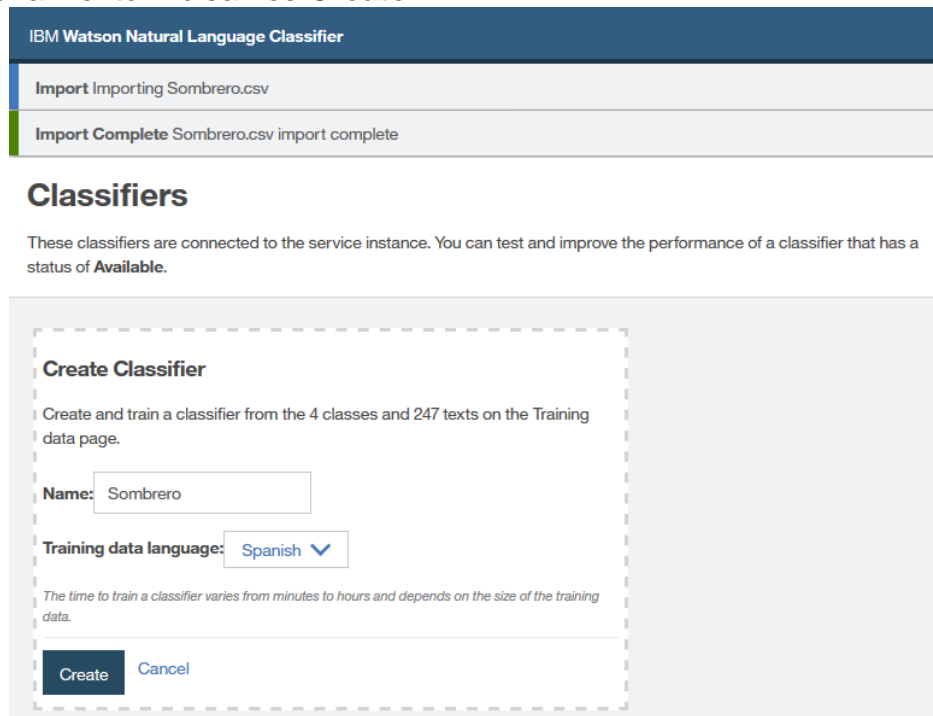
11. Indicamos la ubicación del fichero



12. Una vez tenemos cargada la información pulsamos **Create classifier**.



13. Damos un nombre a nuestro clasificador y seleccionamos **Spanish** en el idioma del entrenamiento. Pulsamos **Create**.



14. El proceso de clasificación comienza y tardará unos minutos.



Classifiers

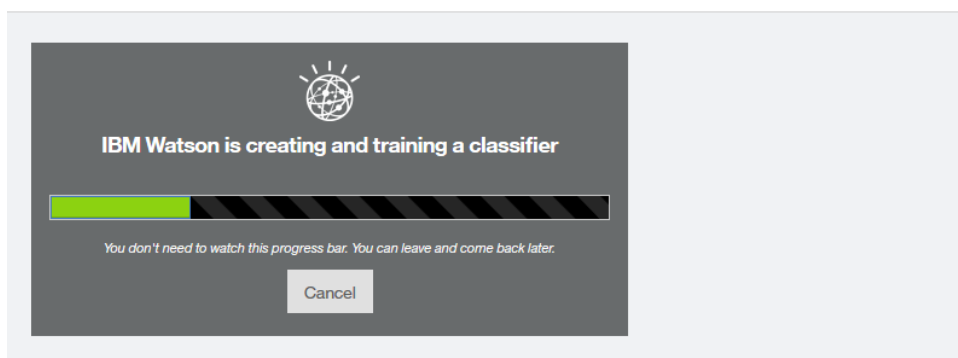
These classifiers are connected to the service instance. You can test and improve the performance of a classifier that has a status of **Available**.

15. Podemos refrescar la página para ver la evolución del proceso.

IBM Watson Natural Language Classifier	
Import	Importing Sombrero.csv
Import Complete	Sombrero.csv import complete
Training Requested	Request to train was submitted
Training Data Validated	Starting to train classifier...

Classifiers

These classifiers are connected to the service instance. You can test and improve the performance of a classifier that has a status of **Available**.



16. Cuando el proceso finalice se nos proporcionará un identificador para nuestro clasificador.

IBM Watson Natural Language Classifier

Classifiers

These classifiers are connected to the service instance. You can test and improve the performance of a classifier that has a status of **Available**.

Sombrero

Created Apr 15, 2017 1:18:22 AM
Classifier ID: 90e7acx197-nlc-37603

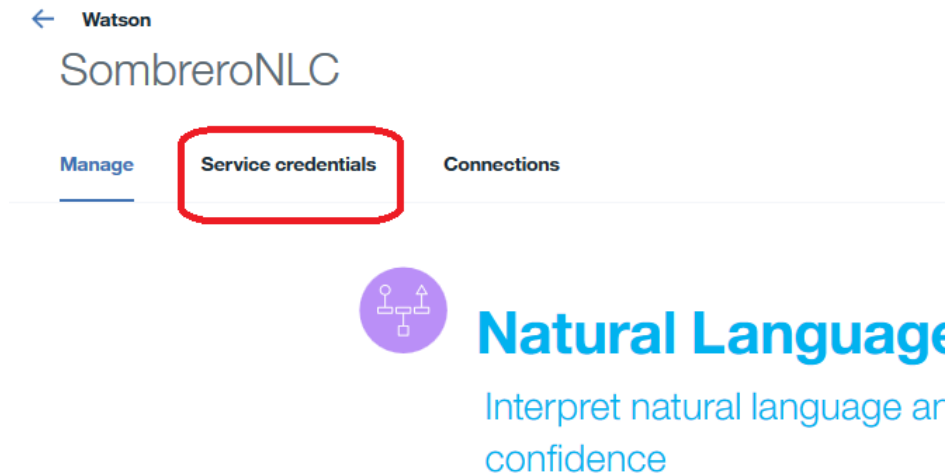
Available

→

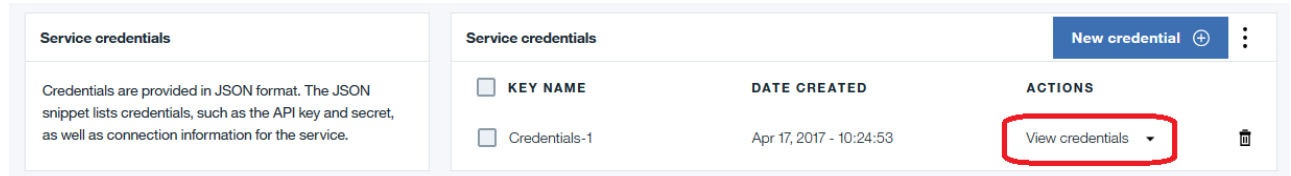
Utilización del clasificador NLC

Ahora que tenemos el clasificador, ya podemos utilizarlo. Para ello, además del identificador del clasificador, necesitaremos conocer las credenciales del servicio NLC que hemos creado.

1. Volvemos a la pestaña del navegador en el que tenemos la consola de administración de Bluemix. Pulsamos en **Service Credentials**.



2. Seleccionamos **View Credentials**.

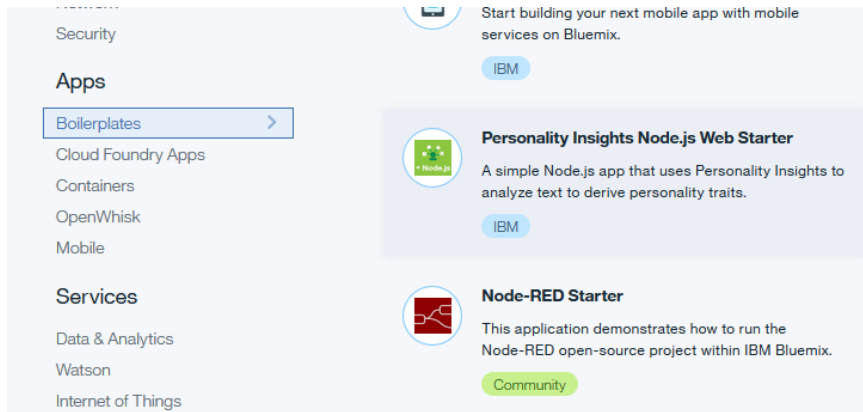


3. Más adelante necesitaremos ese usuario / contraseña que es único para cada instancia.



Desarrollo de una aplicación Node-RED

1. En la consola de administración de Bluemix volvemos a la sección del catálogo, buscamos en el apartado de Boilerplates el icono rojo correspondiente a Node-RED y lo seleccionamos.



2. Proporcionamos un nombre único a la aplicación y pulsamos **Create**.

Create a Cloud Foundry App

Node-RED Starter

This application demonstrates how to run the Node-RED open-source project within IBM Bluemix.

Community

[View Docs](#)

VERSION 0.6.0
TYPE Boilerplate
REGION US South

App name:
SombbreroNLC

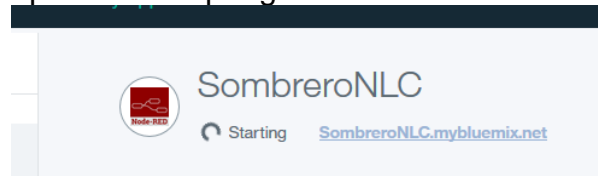
Host name:
SombbreroNLC

Domain:
mybluemix.net

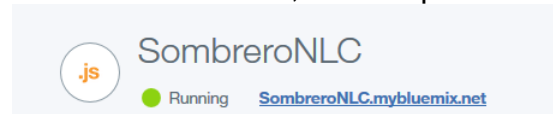
Selected Plan:
SDK for Node.js™
Default

Cloudant NoSQL DB
Lite

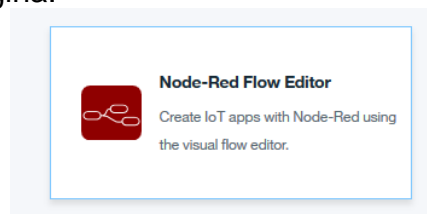
3. Esperamos a que la aplicación se ponga en marcha.



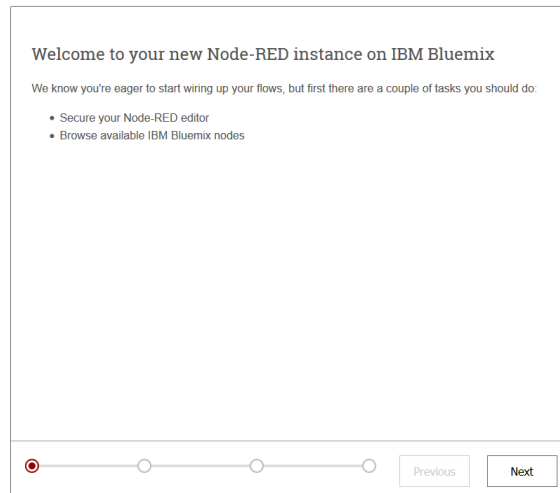
4. Una vez que la aplicación está en marcha, nos desplazamos al final de la página.



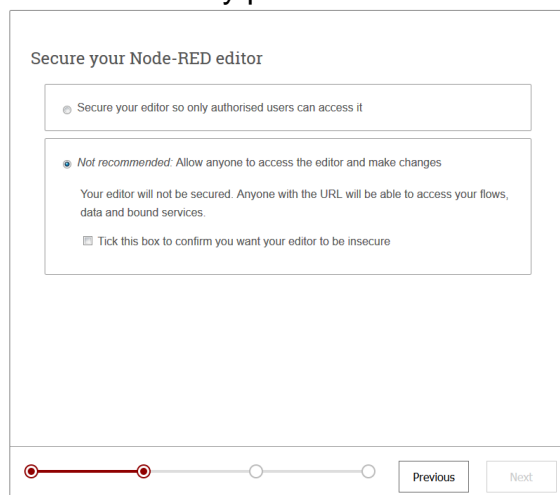
5. A continuación abrimos el editor de aplicaciones pulsando sobre el icono que está al final de esta misma página.



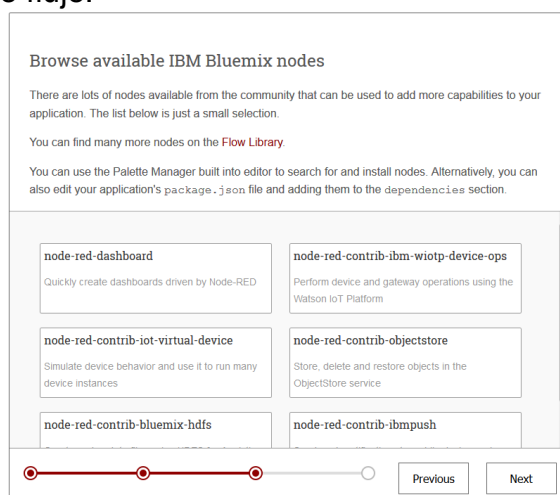
6. Los pasos de configuración que vienen a continuación son nuevos. Antiguamente pasábamos directamente al editor de flujos Node-RED. Ahora es necesario realizar una serie de pasos previos. Pulsamos **Next**.



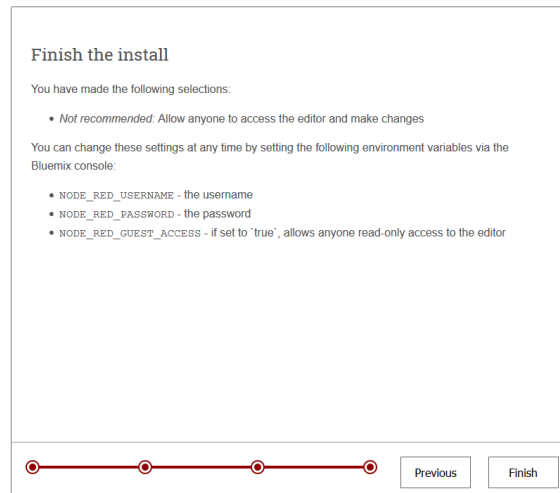
7. Aunque es posible (y recomendable) proteger nuestro editor con usuario y contraseña, para este ejercicio seleccionaremos la opción que no los utiliza. Marcamos la casilla de confirmación y pulsamos **Next**.



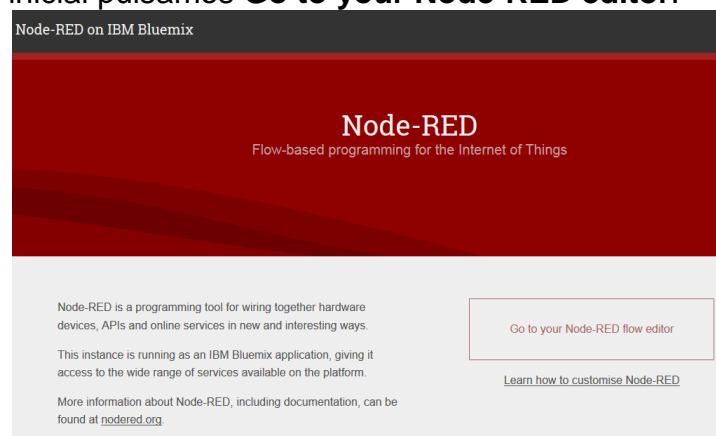
8. Pulsamos **Next** en la siguiente pantalla porque no vamos a utilizar nodos adicionales en nuestro flujo.



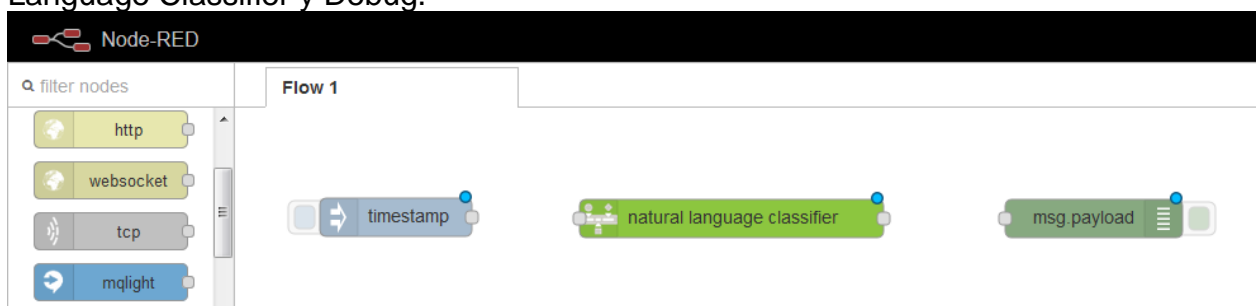
9. Pulsamos **Finish** para finalizar el proceso inicial de configuración de la aplicación.



10. Ya en la página inicial pulsamos **Go to your Node-RED editor**.



11. Añadimos los nodos a nuestra zona de trabajo. Los nodos se añaden seleccionándolos y arrastrándolos. Son tres nodos: Inject, Watson Natural Language Classifier y Debug.



12. Conectamos los nodos pulsando el ratón sobre el conector de la izquierda y arrastrándolo hasta el destino en la derecha.



13. Cuando tengamos los nodos conectados pasamos a configurarlos. Para configurar un nodo es necesario pulsar dos veces sobre él. Primero configuramos el nodo Inject a la izquierda. Seleccionamos que la 'payload', la carga del mensaje que va a crearse, va a ser una cadena de texto e introducimos un texto. Por ejemplo: "Soy

*una persona inteligente y me gusta el color dorado. También soy una persona trabajadora. Y bastante curioso.”. Pulsamos **Done**.*

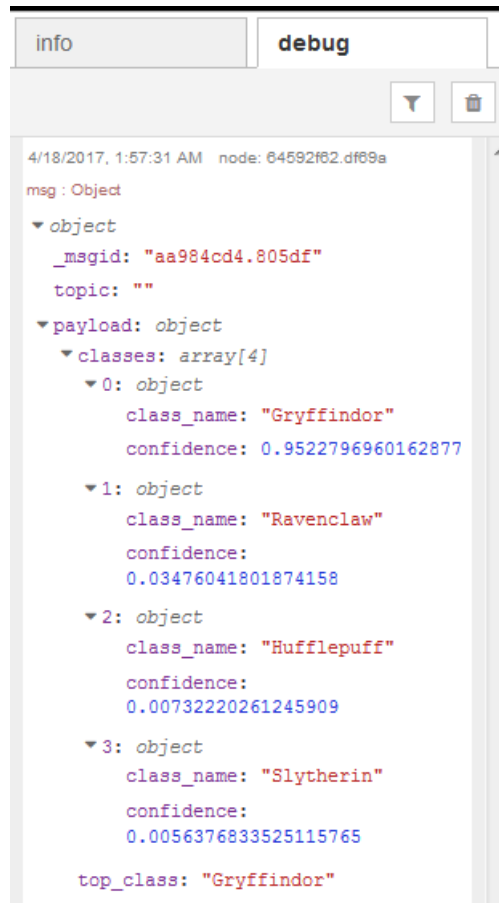
14. Ahora configuramos el nodo de Watson NLC. Aquí copiaremos los credenciales que obtuvimos en un paso anterior. Pulsamos **Done**. También introduciremos el indicador del clasificador que habíamos generado.

15. Por último configuramos el nodo Debug. Seleccionamos la opción que permite mostrar todo el contenido del mensaje que viaja por nuestro flujo al llegar a ese punto. Pulsamos **Done**.

16. Ahora publicaremos los cambios en nuestro flujo pulsando sobre el botón **Deploy** en la parte superior derecha.



17. Ahora ya podemos pulsar sobre el botón del nodo Inject a la izquierda y ver el resultado. El resultado aparecerá en la pestaña debug en la parte derecha de la pantalla.



18. El NLC ha analizado las frases, ha encontrado las palabras claves y ha emitido su veredicto.

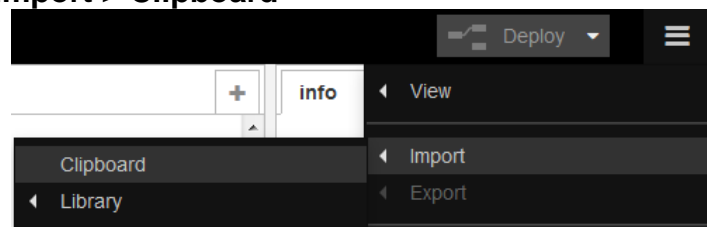
Apéndice A: Versión mejorada de la aplicación Node-RED

Vamos a importar una versión mejorada de la aplicación. Esta versión incluye un interfaz web. La página inicial consta de un formulario en el que podemos introducir la frase a analizar. En la segunda página nos aparece el resultado del análisis de la frase.

1. Node-RED permite importar y exportar flujos en formato JSON. Para importar el flujo debemos copiar en el portapapeles el texto que aparece a continuación.

```
[{"id":"de37a40f.60b94","type":"watson-natural-language-classifier","z":"bcf34d98.35945","name":"Hogwarts","mode":"classify","language":"en","classifier":"90e7acx197-nlc-37603","x":371,"y":201,"wires":[["8078c276.06ac18","8d0e71d1.54856"]]}, {"id":"8078c276.06ac18","type":"debug","z":"bcf34d98.35945","name":"","active":false,"console":"false","complete":"false","x":564,"y":200,"wires":[]}, {"id":"e434a2d2.4293f8","type":"httpin","z":"bcf34d98.35945","name":"Sombrero","url":"/sombrero","method":"get","swaggerDoc":"","x":115,"y":140,"wires":[["6cba4171.655a4"]]}, {"id":"58ca3848.7dbb48","type":"httpresponse","z":"bcf34d98.35945","name":"Response","x":781,"y":139,"wires":[]}, {"id":"6cba4171.655a4","type":"template","z":"bcf34d98.35945","name":"Formulario","field":"payload","fieldType":"msg","format":"handlebars","syntax":"mustache","template":"<html>\n  <body>\n    <h1>Sombrero Seleccionador</h1>\n    <form action='./consulta'>\n      <input type='text' name='frase'>\n      <input type='submit' value='Enviar' name='send'>\n    </form>\n  </body>\n</html>","x":430,"y":141,"wires":[["58ca3848.7dbb48"]]}, {"id":"bc346126.8e5f2","type":"httpin","z":"bcf34d98.35945","name":"Consulta","url":"/consulta","method":"get","swaggerDoc":"","x":115,"y":207,"wires":[["702e57d8.ade95"]]}, {"id":"702e57d8.ade95","type":"function","z":"bcf34d98.35945","name":"Preparar consulta","func":"msg.payload = msg.req.query.frase\nreturn msg;","outputs":1,"noerr":0,"x":244,"y":287,"wires":[["de37a40f.60b94"]]}, {"id":"8d0e71d1.54856","type":"template","z":"bcf34d98.35945","name":"Preparar resultado","field":"payload","fieldType":"msg","format":"handlebars","syntax":"mustache","template":"<html>\n  <body>\n    <table>\n      <tr><td colspan=2><h2>{{payload.top_class}}</h2></td></tr>\n      <tr><td colspan=2>{{#payload.classes}}\n      <tr>\n      <td><strong>{{class_name}}</strong></td>\n      <td>{{confidence}}</td>\n      </tr>\n    </tr>\n    </tr>\n  </table>\n  </body>\n</html>","x":547,"y":284,"wires":[["58ca3848.7dbb48"]]}]
```

2. Volvemos al editor de Node-RED y en el menú de la esquina superior derecha seleccionamos **Import > Clipboard**



3. Y pegamos el texto que habíamos seleccionado previamente. Marcamos la opción **New Flow** y pulsamos **Import**.

Import nodes

```

,"syntax":"mustache","template":"<html>\n  <body>\n    <table>\n      <tr><td colspan=2><h2>{{payload.top_class}}</h2></td></tr>\n      {{#payload.classes}}\n        <tr>\n          <td><strong>\n            {{class_name}}</strong></td>\n          <td>{{confidence}}</td>\n        </tr>\n      {{/payload.classes}}\n    </table>\n  </body>\n</html>","x":547,"y":284,"wires":[["58ca3848.7dbb48"]]

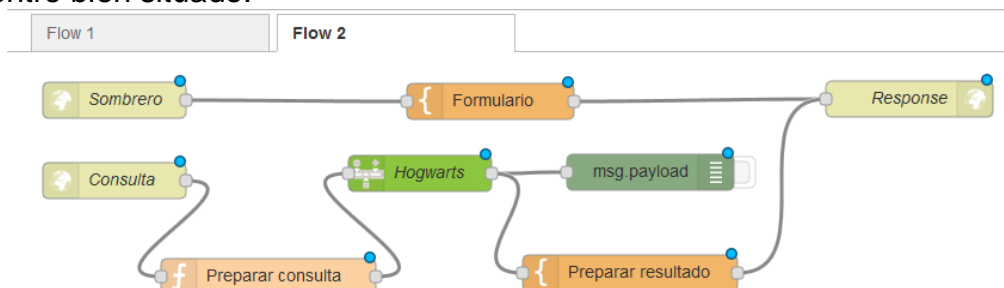
```

Import to

current flow
new flow

Cancel
Import

- Situamos el flujo en pantalla con el ratón y pulsamos el botón de éste cuando se encuentre bien situado.



- Pulsamos el botón **Deploy** para desplegar los cambios.



- En este flujo se han añadido en la parte izquierda dos nodos HTTP Request que atienden las peticiones a las dos páginas web que hemos mencionado antes en /sombrero y /consulta respectivamente.

Flow 1
Flow 2

Edit http in node

Delete
Cancel
Done

Method

GET

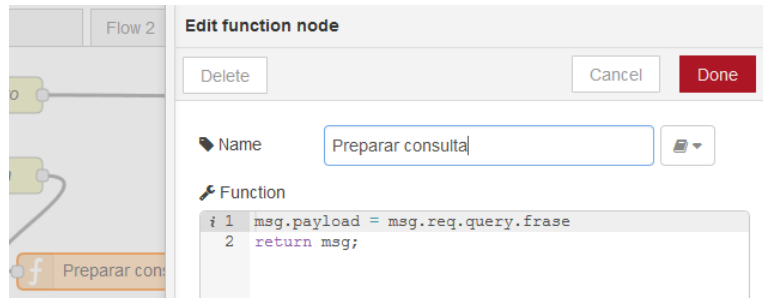
URL

/sombrero

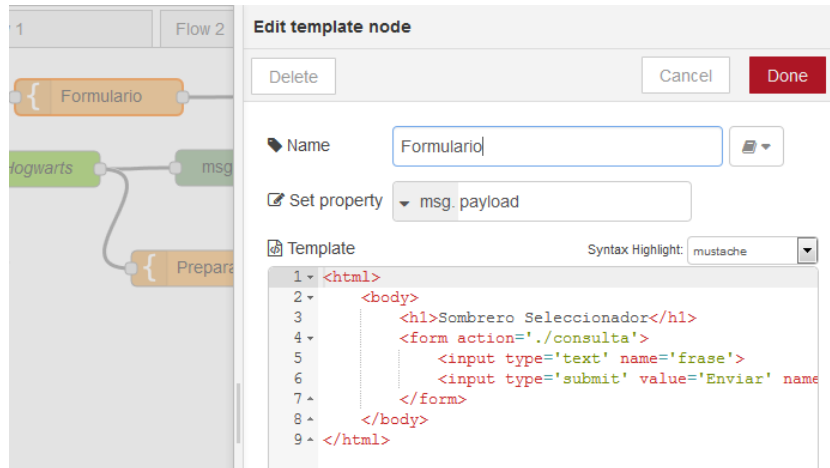
Name

Sombrero

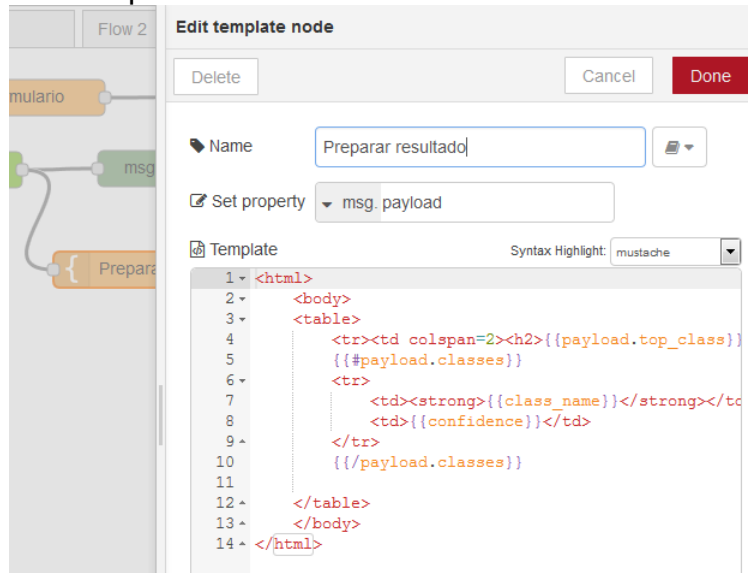
- Un nodo que recupera la consulta para que sea recibida por el siguiente nodo en el flujo.



8. Y dos nodos HTTP Response que construyen el formulario para realizar la consulta...



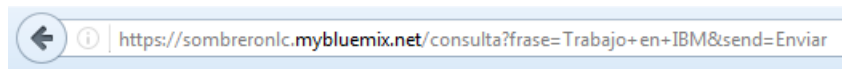
9. ...y la página con la respuesta recibida del clasificador.



10. Invocamos nuestra aplicación añadiendo al dominio de nuestra aplicación el sufijo /sombrero nos aparecerá un sencillo formulario para introducir la consulta.



11. Y cuando enviamos la consulta recibimos como respuesta los datos generados por el clasificador.



Ravenclaw

Ravenclaw 0.5574542074474105

Gryffindor 0.2922823953113878

Slytherin 0.13522162179790095

Hufflepuff 0.015041775443300832

Apéndice B: Despliegue de la aplicación mejorada en Java

En este caso vamos a ver cómo podemos desplegar una aplicación Java que realiza exactamente lo mismo que la que hemos visto en Node-RED.

El objetivo es ver cómo se despliegan aplicaciones utilizando la línea de comandos de Cloud Foundry. Aunque al principio la consola de Bluemix puede parecer más sencilla de utilizar, a la larga, es más cómodo realizar muchas de las operaciones desde una línea de comandos. Especialmente cuando es necesario repetirlos o ejecutarlos de forma automatizada.

Para ello debemos tener instalada la línea de comandos Cloud Foundry. Esta aplicación la podemos descargar desde <https://github.com/cloudfoundry/cli/releases>.

1. Abrimos una ventana de comandos y tecleamos la siguiente instrucción para conectarnos a Bluemix:

```
cf login -a api.ng.bluemix.net
```

2. A medida que nos lo solicite introduciremos nuestro usuario, contraseña y el espacio al que nos queremos conectar.

```
[genaro@localhost ~]$ cf login -a api.ng.bluemix.net
API endpoint: api.ng.bluemix.net

Email> genaro@es.ibm.com

Password>
Authenticating...
OK

Targeted org genaro@es.ibm.com

Select a space (or press enter to skip):
1. GNF_Space
2. Hacks
3. Demos

Space> 3
Targeted space Demos

API endpoint: https://api.ng.bluemix.net (API version: 2.54.0)
User: genaro@es.ibm.com
Org: genaro@es.ibm.com
Space: Demos
[genaro@localhost ~]$
```

3. A continuación introducimos el comando que sube la aplicación. Debemos indicar el nombre de la aplicación (que debe ser único) y la ubicación del fichero WAR con el código.

```
cf push NLCHat -p /home/genaro/NLCHat.war
```

4. Comienza el proceso de despliegue de la aplicación en Bluemix.

```
Creating app NLCHat in org genaro@es.ibm.com / space Demos as genaro@es.ibm.com...
OK

Creating route nlchat.mybluemix.net...
OK

Binding nlchat.mybluemix.net to NLCHat...
OK

Uploading NLCHat...
Uploading app files from: /tmp/unzipped-app541938483
Uploading 4.2K, 11 files
Done uploading
OK

Starting app NLCHat in org genaro@es.ibm.com / space Demos as genaro@es.ibm.com...
Downloading dotnet-core_v1_0_10-20170124-1145...
Downloading liberty-for-java_v3_5-20161114-1152...
Downloading xpages_buildpack...
Downloading binary_buildpack...
Downloading staticfile_buildpack...
Downloaded staticfile_buildpack
Downloading liberty-for-java_v3_7-20170118-2046...
```

5. Una vez haya finalizado la instalación veremos la siguiente información en pantalla:

```
Uploading droplet...
Uploaded droplet (131.8M)
Uploading complete
Destroying container
Successfully destroyed container

0 of 1 instances running, 1 starting
0 of 1 instances running, 1 starting
0 of 1 instances running, 1 starting
0 of 1 instances running, 1 starting
0 of 1 instances running, 1 starting
0 of 1 instances running, 1 starting
0 of 1 instances running, 1 starting
1 of 1 instances running

App started

OK

App NLCHat was started using this command `.liberty/initial_startup.rb`

Showing health and status for app NLCHat in org genaro@es.ibm.com / space Demos as genaro@es.ibm.com...
OK

requested state: started
instances: 1/1
usage: 1G x 1 instances
urls: nlchat.mybluemix.net
last uploaded: Tue Apr 18 22:33:53 UTC 2017
stack: cflinuxfs2
buildpack: Liberty for Java(TM) (WAR, liberty-17.0.0_1, buildpack-v3.8-20170308-1507, ibmjdk-1.8.0_20170215, env)

#0 state since cpu memory disk details
#0 running 2017-04-19 12:35:38 AM 60.1% 127.3M of 1G 184.9M of 1G
[genaro@localhost ~]$
```

6. Vemos que, por defecto, se ha iniciado una instancia de la aplicación con 1GB de memoria. Ahora vamos a conectar nuestra instancia del servicio Watson.

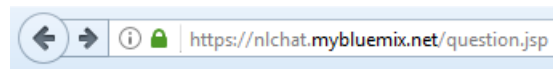
```
cf bind-service NLCHat SobreroNLC
```

```
[genaro@localhost ~]$ cf bind-service NLCHat SombreroNLC
Binding service SombreroNLC to app NLCHat in org genaro@es.ibm.com / space GNF_Space as genaro@es.ibm.com...
OK
TIP: Use 'cf restage NLCHat' to ensure your env variable changes take effect
```

7. Y, como nos sugiere la CLI, reiniciamos la aplicación para que surjan efecto los cambios:

```
[genaro@localhost ~]$ cf restage NLCHat
Restaging app NLCHat in org genaro@es.ibm.com / space GNF_Space as genaro@es.ibm.com...
Downloading php_buildpack...
Downloading staticfile_buildpack...
Downloading liberty-for-java...
Downloading sdk-for-nodejs_v3_10-20170119-1146...
Downloading binary_buildpack...
Downloaded dotnet-core_v1_0_10-20170124-1145
Downloading liberty-for-java_v3_4_1-20161030-2241...
Downloaded liberty-for-java
```

8. Abrimos un navegador y cargamos la aplicación:



Sombrero Seleccionador

Trabajo en IBM

9. Y ahora podemos detener la aplicación:

```
cf stop NLCHat
```

```
[genaro@localhost ~]$ cf stop NLCHat
Stopping app NLCHat in org genaro@es.ibm.com / space Demos as genaro@es.ibm.com...
OK
[genaro@localhost ~]$ █
```

10. Y desinstalarla si fuese necesario:

```
cf delete NLCHat -r
```

```
[genaro@localhost ~]$ cf delete NLCHat -r
Really delete the app NLCHat?> yes
Deleting app NLCHat in org genaro@es.ibm.com / space Demos as genaro@es.ibm.com...
OK
[genaro@localhost ~]$ █
```