**TITLE**

**Summary**

Se está utilizando Rasa con NLU (Lenguaje natural de entendimiento). Es un framework que utiliza IA y funciona como un SaaS. Se puede tener todo en un servidor local. Necesita de visual C++ Build para poder funcionar. Se compone de RASA NLU, el cual es el encargado de interpretar los mensajes que le llegan de un usuario y RASA CORE el cual se encarga de predecir cuál es la mejor acción y así responder al usuario. Está compuesto de lo siguiente:

* Intent: Es la intención de lo que quiere hacer el cliente. por ejemplo reservar un tiquete.
* Entity: Es la extracción de lo importante en el mensaje, por ejemplo si dicen que quieren reserva en un X restaurante a las 10 pm, el entity sería el nombre y la hora.
* Stories: Define la interacción entre el usuario y el bot ya predefinida. Los stories tienen la siguiente nomenclatura:
  + Nombre del archivo
  + Entrada del usuario
  + Acción por parte del bot
* Actions: Las acciones permiten preguntar al usuario por más informacion para que todo sea más preciso.

**Configuration**

Se deben crear los siguientes archivos.

* NLU training(nlu.md): Son Datos que permiten entrenar el bot para que funcione mejor.
* Stories: Son conversaciones entre el usuario y el bot.
* Domain: Es el que contiene todo de lo que se compone el bot(intent, actions, entities, etc.)

Ya no se utiliza run\_evaluation. En vez de eso se debe hacer uso de este import:

from rasa\_nlu.test import run\_evaluation

**Comandos a utilizar para crear el bot rápidamente**

* Crear un entorno virtual.
* Ejecutar:

make install\_dependencies

Este comando instalará todo lo necesario **dentro del entorno virtual**. Además, se instalan las dependencias de spacy y los paquetes en español

* Ejecutar:

make init

Este comando crea el proyecto.

* Para correr rasa, ejecutar:

`rasa` shell

Para salirse de rasa, ingresar el comando /stop en el prompt de rasa

* Para crear un story es necesario actualizar también el archivo nlu.md y el archivo domain.yml. Poniendo los intent y las acciones a realizar. También es importante que cada acción lleve antepuesta la palabra utter\_

**Integración con otras aplicaciones(Facebook, Slack, Telegram.)**

El primer paso para realizar la integración es descargar y ejecutar [ngRok](https://dashboard.ngrok.com/get-started). Este programa nos permite realizar un túnel entre la web y nuestra máquina local, para que las API's tengan acceso a nuestro dispositivo. Basta con seguir el tutorial que aparece en la página. Para correrlo hay que *setear* el token personal que aparece en la página web una vez nos registramos.

ngrok authtoken 1Q6eHyI0d3A3er5R3qokE8qh6Tn\_3AD9oMxkYWYMK91WYmM1n

Después se procede a ejecutar el programa corriendo en el puerto que queramos:

ngrok http 5005

Si todo sale correctamente, aparecerá en la terminal un mensaje con nuestro nombre de usuario.

<http://localhost:4040/inspect/http> Consola de ngrok

Para configurar las API's basta con modificar el archivo credentials.yml y poner los token y el nobmre de nuestras apps creadas en cada página.

**Slack:**

<https://app.slack.com/client/TMUMESR3P/CMUMETVPF/details/info>

<https://rasa.com/docs/rasa/user-guide/messaging-and-voice-channels/> <https://rasa.com/docs/rasa/user-guide/connectors/telegram/> <https://rasa.com/docs/rasa/user-guide/connectors/facebook-messenger/>

<https://6b7b6b0e.ngrok.io/webhooks/slack/webhook> <https://6b7b6b0e.ngrok.io/webhooks/facebook/webhook> <https://6b7b6b0e.ngrok.io/webhooks/telegram/webhook>

**Telegram**

t.me/clarencebot Token: 930390880:AAF26zqGT8KmjtygdGHlGSwGVNtfaztHdgw