UNIVERSITAT POLITÈCNICA DE VALÈNCIA

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Escuela Politécnica Superior de Gandia

Diseño e implementación de un asistente virtual de información turística para la ciudad de Gandia

Trabajo Fin de Grado

Grado en Tecnologías Interactivas

AUTOR/A: Hernandez Ramirez, Juan Carlos

Tutor/a: Sánchez Anguix, Víctor

Cotutor/a: Alberola Oltra, Juan Miguel

Director/a Experimental: MORENO TEODORO, JOAN CIPRIA

CURSO ACADÉMICO: 2022/2023

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Escuela Politécnica Superior de Gandia

Grado en Tecnologías Interactivas

UNIVERSITAT
POLITÈCNICA
DE VALÈNCIA

ESCUELA POLITECNICA
SUPERIOR DE GANDIA

“Diseño e implementación de un asistente virtual de información turística para la ciudad de Gandía“

Trabajo Fin de Grado

Autor: Hernandez Ramirez, Juan Carlos
Tutor: Sánchez Anguix, Víctor
Cotutor: Alberola Oltra, Juan Miguel
Director Experimental: Moreno Teodoro, Joan Cipria
Curso académico: 2023

Resumen

El objetivo de este trabajo final de grado es el desarrollar un Chatbot en Rasa para mejorar la experiencia turística en Gandía. El Chatbot está diseñado para interactuar con los turistas y proporcionarles información útil sobre la ciudad, lugares turísticos, eventos, gastronomía etc. El Chatbot será capaz de responder las preguntas frecuentes de los turistas sobre las mejores rutas para recorrer la ciudad, los horarios de apertura de los lugares turísticos, las opciones de transporte disponibles… La idea es proporcionar un guía turístico personalizado y accesible en cualquier momento y lugar. Para este desarrollo se utilizará Rasa, una herramienta utilizada para desarrollar Chatbots basada en inteligencia artificial y aprendizaje automático. Con lo que se pretende conseguir un lenguaje que permita interactuar con las personas de forma completamente natural.

Palabras clave: asistente virtual, chatbot, turismo inteligente, inteligencia artificial; procesamiento del lenguaje natural

Abstract

The objective of this final degree work is to develop a Chatbot in Rasa to improve the tourist experience in Gandía. The Chatbot is designed to interact with tourists and provide them with useful information about the city, tourist attractions, events, gastronomy, etc. The Chatbot will be able to answer tourists' frequently asked questions about the best routes to take around the city, opening hours of tourist sites, available transportation options... The idea is to provide a personalized and accessible tourist guide at any time and place. Rasa, a tool used to develop Chatbots based on artificial intelligence and machine learning, will be used for this development. The aim is to achieve a language that allows interacting with people in a completely natural way.

Keywords: virtual assistant, chatbot, smart tourism, artificial intelligence, natural language processing.

Índice general

Índice de figuras

1. Introducción

La forma en la que nos comunicamos con la tecnología ha evolucionado a lo largo de los años, empezando con dispositivos muy primitivos que simplemente almacenaban información hasta hoy en día que podemos comunicarnos con una IA capaz de entender el lenguaje humano, como es el caso del chatbots.

El desarrollo de los primeros chatbots se remonta a la década de los 60 y surge como parte de la investigación para determinar los límites de la IA. Sus primeros usos tenían como objetivo mantener conversaciones terapéuticas con los usuarios a partir de patrones y esquemas muy simples. Un caso que fue muy llamativo, fue el de Chatterbot, un juego multiusuario de calabozos y escenarios simulados que te hacía creer que estaba interactuando con otros usuarios cuando en realidad estabas manteniendo conversaciones con una inteligencia artificial. Este juego fue muy exitoso porque ningún usuario conocía de la existencia de la IA y todos se pensaban que estaban jugando online.

A medida que la tecnología ha ido avanzando los chatbots se han vuelto mucho más potentes y capaces de ayudarnos en cualquier tipo de situación como puede ser el ya conocido por todo el mundo Chat GPT. Un chatbot capaz de responder a todo tipo de preguntas y realizar casi cualquier tarea, desde pedirle que nos escriba una receta para un postre, hasta pedirle que nos redacte la tesis de un doctorado. Tal ha sido el impacto de Chat GPT que muchos se han alarmado por el potencial que tiene para sustituir muchos de los puestos de trabajo de la actualidad como son: creadores de contenido (publicidad,periodismo…), trabajos tecnológicos (programadores, ingenieros software…), trabajadores industriales etc.

Sin embargo, lejos de la realidad a Chat GPT le quedan unas cuantas tareas pendientes antes de poder sustituirnos y es que aún carece de algunas capacidades humanas como son la capacidad de raciocinio y pensamiento crítico, algo fundamental para tener un buen desempeño en el mundo laboral.

Los chatbots siempre han tenido como objetivo el dar soporte al ser humano en distintas áreas: hospitalidad, turismo, psicología… Y en los últimos años, nos hemos dado cuenta que pueden brindar una gran ayuda para informar a los usuarios en un gran cantidad de temas. A partir de aquí surge la idea de desarrollar un chatbot para el turismo Gandia. Desde el ayuntamiento de Gandia hemos recibido parte de las preguntas más frecuentes que se realizan en la oficina de turismo y esta ha sido la base para desarrollar el proyecto.

1.2 Objetivo del proyecto

Este proyecto tiene como objetivo desarrollar un chabot inteligente, capaz de proporcionar información útil sobre el turismo en Gandía. Para llegar a este objetivo, se requiere del análisis de algunos puntos previos

- Estudio de las distintas tecnologías que existen en la actualidad para el desarrollo de los chatbots
- Estudio de la interacción que se produce entre el ser humano y una máquina

- Búsqueda de los chatbots que existen para el turismo en la actualidad
- Definición de los objetivos claros y realistas a los que se pretende llegar.
- Diseño e implementación del chatbot
- Validación del chatbot con personas reales.

2. Marco teórico

En este punto se realiza un estudio de las distintas tecnologías que existen en la actualidad para el desarrollo de los chatbots, además también de realizar un pequeño análisis de cómo es la interacción entre la persona y la máquina. Para finalizar se exponen los distintos chatbots que existen en el sector del turismo.

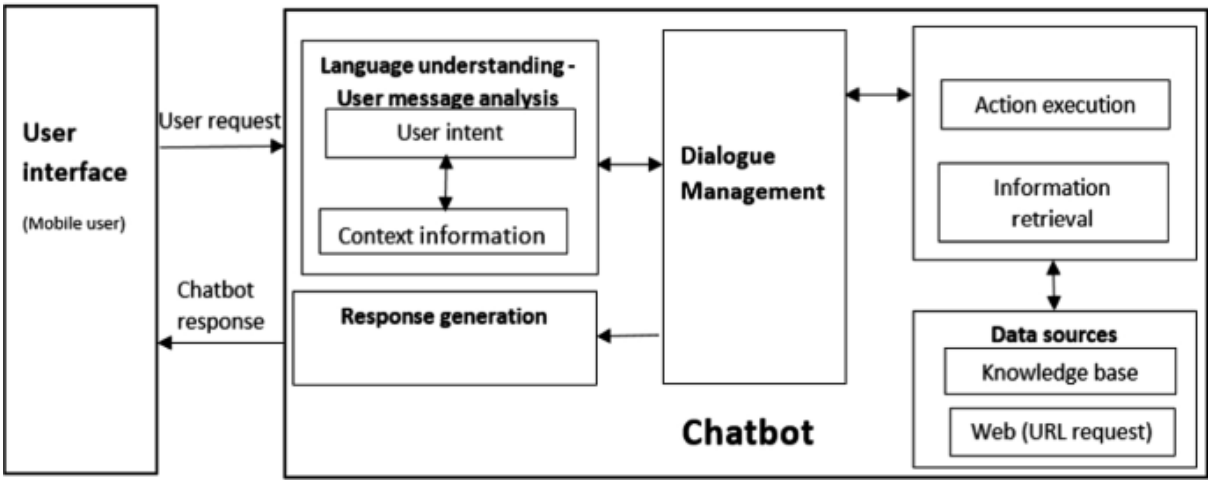
2.1 Chatbots

Un chatbot (o también conocido como asistente virtual) es un programa software diseñado con el fin de simular una conversación humana con usuarios reales. Para ello utiliza algoritmos de inteligencia artificial y técnicas de procesamiento de lenguaje, consiguiendo interpretar las respuestas del usuario y proporcionando respuestas adecuadas y coherentes.

2.1.1 Diseño y funcionamiento de un chatbot

El diseño e implementación de un chatbot involucra diversas técnicas. Entender cuál es el objetivo final del chatbot es clave para determinar qué plataformas, herramientas y algoritmos se utilizaran para su desarrollo.

El primer paso para desarrollar un sistema es dividir cada uno de sus pequeños desarrollos en estándares para conseguir un desarrollo modular.



F.1: Esquema funcionamiento chatbot

El proceso se inicia con la petición del usuario, por ejemplo. ¿Dónde puedo alquilar un coche?. Esta petición puede provenir desde diferentes orígenes según donde esté programada su interfaz.

Una vez el chatbot reciba la petición, este debe interpretar el texto recibido y extraer toda la información asociada que pueda haber. En este caso tenemos: Intent: “alquilar”, entities: ‘[vehículo: “coche”]’

A continuación, el chatbot buscará según su base de conocimiento, cuál es la mejor respuesta que se adapta según la entrada de datos recibida. Determinado si tiene toda la información necesaria para proporcionar una respuesta o requiere de algún tipo de aclaración.

Cuando la fase de entendimiento ha finalizado el chatbot decidirá cuál es la respuesta correcta para el usuario según su database o pudiendo ser necesaria realizar una petición a un servicio externo mediante APIs o similar. Ej: www.rentacar.com

Antes de enviar la respuesta al usuario, esta debe ser procesada mediante componentes de generación de respuesta utilizando NLG (Natural Language Generation) para proporcionar una respuesta lo más natural posible según el modelo con el que haya sido entrenado el chatbot. : Aquí puedes alquilar un coche: www.rentacar.com

2.1.2 Clasificación de los chatbots

Los chatbots se pueden clasificar según diferentes parámetros: su ámbito de conocimiento, los servicios que proporciona, objetivos que pretende conseguir y según el método utilizado para su entrenamiento.

Ámbito de conocimiento

Según el ámbito de conocimiento que pueden abarcar los chatbots se clasifican en 2 categorías:

- Dominio abierto: el chatbot puede hablar de casi cualquier tema y responder apropiadamente.

- Dominio cerrado: el conocimiento del chatbot está centrado en un tema en concreto y puede fallar ante otro tipo de preguntas.

Servicio proporcionado

Se considera la proximidad que tiene el chatbot con el usuario.

- Interpersonal: Se entiende a los chatbots que proporcionan servicios de reserva de hoteles, restaurantes etc. No pretenden ser ‘amigos’ de los usuarios y normalmente se utilizan para responder a FAQs (Frequently Asked Questions). Simplemente interpretan la información y se la proporcionan al usuario.

- Intrapersonal: Estos chatbots existen dentro del dominio personal del usuario en aplicaciones como Telegram, Slack o WhatsApp. Buscan ser amigos de los usuarios y entender al humano como si lo hiciera una persona.

- Inter-agent: Son chabots denominados ‘omnipresentes’ debido a su naturaleza de uso. Actúan como intermediarios entre el usuario y un agente. Un claro ejemplo de esto son Cortana o Alexa.

Objetivos

Su clasificación se basa según cual es el objetivo principal del chatbot.

- Informativa: Chatbots diseñados para proporcionar información estática como es el caso de FAQ chatbots.

- Chat-based/Conversational: Chabots que pretende entender a los usuarios en casi cualquier ámbito y emitir una respuesta coherente al respecto.

- Task-based: Chabots que tiene como objetivo desarrollar una tarea concreta como puede ser la realizar la reserva de un hotel o hacer un pedido de comida.

Modelo de entrenamiento

Esta clasificación tiene en cuenta el método utilizado para el entrenamiento y generación de respuesta. Existen 3 modelos utilizados para ello:

- Rule-based: Se basan en la elección de una respuesta a partir de la entrada del usuario según unas reglas previamente establecidas. No genera nuevas tipo de respuestas ya que estas son estáticas y a cuanta mayor sea el número de reglas que tiene el chatbot para entender al usuario mejor será la respuesta obtenida. Sin embargo estos chatbots presentan varias inconsistencias a la hora de manejar los errores gramaticales, además que solo tiene en cuenta la última respuesta introducida por el usuario.

- Retrieval-based: Modelo mucho más flexible que analiza la pregunta del usuario antes de enviar una respuesta, esto lo hace mediante uso de unos índices que recupera a partir de las conversaciones previas que haya podido tener el chatbot.

- Generative-model: Este tipo de modelo genera respuestas a partir de los mensajes actuales y los anteriores, con lo que consigue asemejarse más a lo que sería una conversación real. Para ello hace uso de algoritmos de machine learning y técnicas de deep learning. Sin embargo tienen un alto coste de creación y entrenamiento de modelos.

2.1.3 Interacción humano-chatbot

A la hora de determinar el éxito de un chatbot, es crucial que este sea capaz de transmitir humanidad al usuario, que el usuario sea capaz de sentirse cómodo a la hora interaccionar como si de una persona se tratase. Es por ello que la interacción humano-chatbot siempre ha sido un foco de estudio a la hora de desarrollar un chatbot. Existen numerosas características que nos definen como humanos a la hora de tener una interacción, aquí se van a resumir algunas de las más importantes:

Proactividad

El chatbot debe ser capaz de invitar al usuario a hablar. Esto lo puedo conseguir proponiendo temas o preguntas a los que esté puedo responder. Esta es una característica

que aporta más humanidad a la conversación y por tanto proporciona una mejor sensación de comida al usuario.

Capacidad de entendimiento

Una característica fundamental a la hora de desarrollar el chatbot es la capacidad de entendimiento de la que dispone.Este debe ser capaz de entender el contexto en el que transcurre la conversación y otorgar una respuesta coherente según convenga. Por ejemplo, si al bot le pregunta. ¿Dónde puedo alquilar? Este debe tiene que ser capaz de entender si en algún momento anterior se habló de algún vehículo en particular, o por el contrario el chabot tiene que realizar otra pregunta para averiguar qué vehículo debe alquilar el usuario.

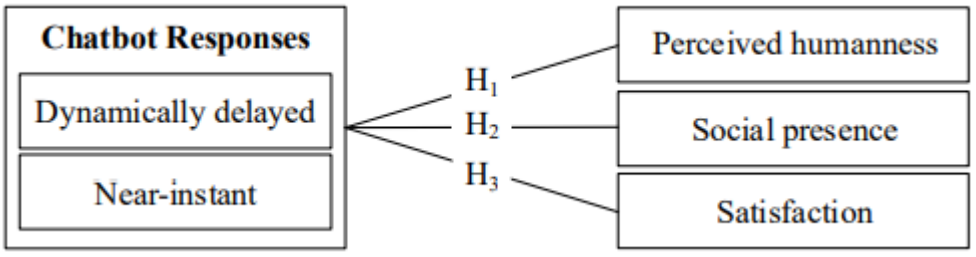
Robustez

El control de errores es algo que tiene que estar definido en cualquier software, y un chatbot no debería ser menos. Al igual que un ser humano no es capaz de responder a todas las preguntas que le hacen, el chatbot tampoco debería ser capaz de hacerlo. Para esto existen algunos elementos de control como son los fallback, que nos permiten controlar este tipo de preguntas mostrando una respuesta similar: Lo siento, todavía no soy capaz de responder a este tipo de preguntas.

Cabe destacar que no siempre este es el camino correcto ya que si el usuario nos formular la pregunta de diferente manera, quizás el chatbot si que es capaz de entender la pregunta. Por lo que conviene tener distintos métodos de control de respuestas antes de devolver un fallback.

Tiempo de respuesta

Faster is not always Better es un artículo escrito por KIT (Karlsruhe Institute of Technology), en el que se argumenta que responder a un usuario de manera inmediata no siempre es la mejor solución si lo que buscamos es emular una conversación real.



F.2: Efectos de aplicar delay en un chatbot

En este estudio se encontró que los usuarios tenían un mayor nivel de satisfacción, mejor percepción de humanidad y mejor presencia social, si el chatbot forzaba cierto delay en sus respuestas. Si el chabot simulaba estar leyenda las respuestas como si de una persona se tratase y a partir de aquí pensar cuál es la mejor respuesta que se puede dar, los resultados obtenidos apuntaban a que el

2.2 Tecnologías

Son varios los factores a tener en cuenta antes de elegir una plataforma en la que desarrollar un chatbot, es por ello que en esta sección se hará un breve análisis de las distintas plataformas que existen hoy en día para desarrollar un chatbot y cuales son sus principales características.

Rasa

Rasa es un framework utilizado para la creación de chatbots y asistentes virtuales desarrollado en Python. Rasa es conocido por su gran flexibilidad y eficacia ya que permite una gran personalización al ser de código abierto. Además es compatible con múltiples lenguajes y permite la integración de varios canales: web, email, api rest…Esta es una gran elección siempre y cuando tengas conocimientos básicos de programación ya que su curva de aprendizaje inicial es un poco elevada.

Rasa cuenta con una versión gratuita y otra versión empresarial de pago (Rasa X) que nos aporta características adicionales, como son el soporte de una página web para la puesta en producción del chatbot y una interfaz que nos permite visualizar las conversaciones desarrolladas previas que se han tenido con el chatbot.

Botkit

Bootkit es conocida por ser un framework open source centrado en el código de fácil desarrollo. Botkit cuenta con una serie de herramientas y recursos predefinidos que hacen muy fácil la creación del chatbot para diversas plataformas (Facebook,Slack, Microsoft Teams…). Además cuenta con una gran comunidad activa y gran número de plugins que permiten ampliar las funcionalidades del chatbot según tus necesidades.

Bootkit cuenta también con una herramienta denominada Botkit CMS que permite el desarrollo de un chatbot mediante interfaz gráfica, haciendo mucho más accesible la creación de chatbots para todo el mundo. Bootkit es gratuito pero cuenta con un coste adicional por parte de algunos servicios o complementos.

Dialogflow

Plataforma desarrollada por google para la creación de chatbots basados en la nube.

Dialogflow es capaz de interpretar el contexto de las conversaciones con lo que se consigue una interacción más realista ante el usuario. Permite la integración con un gran número de canales, incluyendo: Google Assistant, Amazon Alexa, Facebook…Dialogflow puede utilizar múltiples tipos de entradas ya sea en texto o voz y así mismo permite la respuesta en texto o mediante una voz sintética.

Dialogflow cuenta con un plan gratuito pero muy limitado y planes adicionales de pago que ofrecen mayor número de características.

Microsoft Bot Framework:

Microsoft Bot framework es una plataforma de desarrollo de chatbots que utiliza LUIS (Language Understanding Intelligent Service) para el entendimiento del lenguaje natural. Destaca por disponer de herramientas propias desarrolladas por Microsoft, que permite diseñar, construir y conectar diferentes chatbots mediante la nube de

Azure.También cuenta con funcionalidades como el reconocimiento de voz y la integración de IA lo que permite tener un gran desempeño.

Microsoft Bot Framework permite el desarrollo de un chatbot de forma completamente gratuita pero su despliegue en Azure tiene costos asociados.

2.3 Comparativa de chabots

Como se puede observar en la figura 3 de análisis de características, la elección del software que utilicemos estará muy determinado según el caso de uso en el que nos encontremos y el tipo de chatbot que queremos desarrollar. En este caso nos hemos decantado por Rasa, debido a las siguientes razones:

- Open source y gratuito, por lo que no necesitamos de contemplar ningún coste adicional si queremos realizar el despliegue del chatbot.
- Aunque la versión de Rasa X Enterprise es de pago,Rasa cuenta con múltiples ayudas que nos facilitan la puesta en producción y mantenimiento. Al ser de código abierto, nosotros mismos podemos replicar las funcionalidades de Rasa X mediante un desarrollo adicional.
- Cuenta con herramientas como NLU (Natural Language Understanding) para mejorar la interpretación de los mensajes del usuario.
- Nos permite una control total sobre las funcionalidades que le podemos poner al bot

Plataforma	Open source	NLU	Integración en canales	Personalización	Documentación/Comunidad	Pla de pago
Rasa	Sí	Sí	Slac,Facebook, WEB	Alta al ser código abierto	La versión gratuita de Rasa (Rasa X)dejo de tener soporte en el 2022	Versión graituta y versión de pago (Rasa X Enterprise)
Botkit	Sí	Sí (mediante plugins)	Slack,Microsoft, Messenger	Alta al ser código abierto	Amplia documunetación y comunidad activa	Bootkit gratuito pero tiene costes adicionales según loser servicios
Dialogflow	No	Sí	Amazon Alexa, GoogleAssistant, Facebook	Limitada por el servicio de Google	Extensa comunidad soportada por Google	Plan grauito pero muy limitado y plan de pago con mayor capacidad y caracteristiacs
Microsoft Bot Framework	No	No (cuenta con LUIS)	Gran capcaidad de integración mediante Azure	Amplia red de servicios soportado por Microsoft	Extensa comunidad soportada por Microsoft	Su dessarrollo es gratuito pero su despligue en Azure tiene costes asociados

F.3: Comparativa de softwares para desarrollar un chatbot

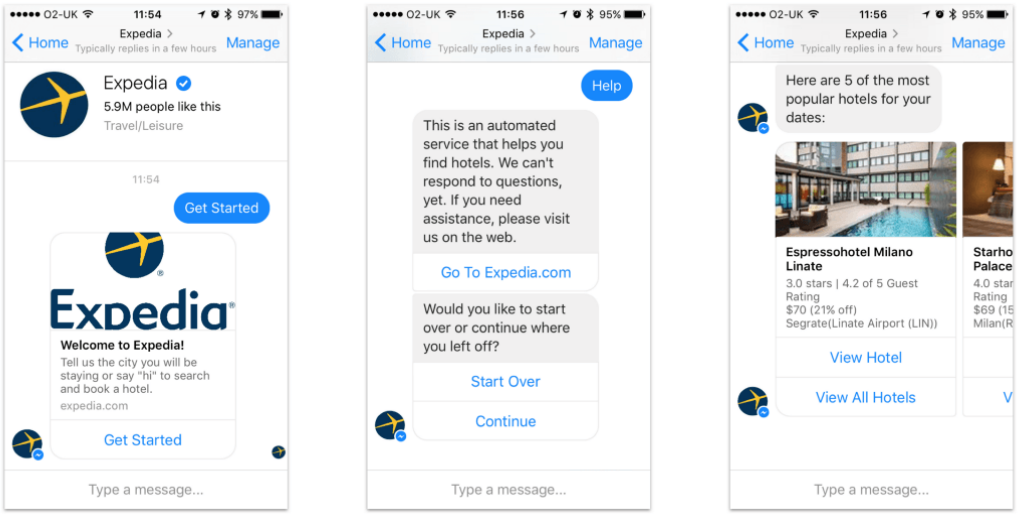
2.4 Chatbots de turismo

Recientemente se ha observado una clara tendencia alcista en el uso de los chatbots en casi todos los sectores de asistencia personal, y no es menos el sector del turismo. Muchas son las empresas turísticas que se decantan por un chatbot a la hora de proporcionar una mejor atención personalizada a los usuarios (aerolíneas, hoteles, agencias de viajes etc.) y razones no les falta. Algunas de ellas son: disponibilidad completa de 24h al día los 7 días de la semana, motivación (no se cansan y no necesitan de estímulos), permiten asistencia a un gran número de usuarios simultáneamente y sin esperas, gran personalización en el tipo de respuesta/servicio que se quiere proporcionar…

Es por ello que muchas empresas conocidas dentro del sector turístico han desarrollado sus propios chatbots:

Expedia Facebook Messenger Bot

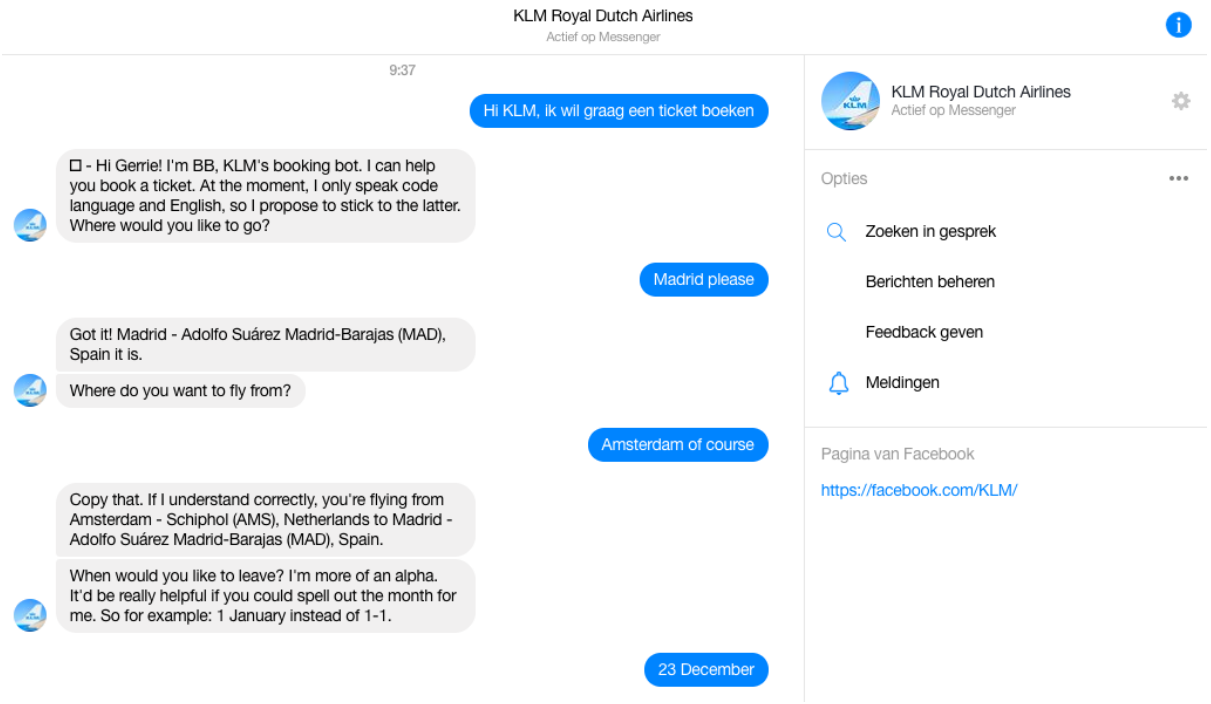
Un chatbot desarrollado en Facebook Messenger, que permite a los usuarios realizar reservas de hoteles. Los usuarios tienen que indicar una localidad de destino y las fechas en las que desean realizar el viaje y estos recibirán recomendaciones de reservas de hoteles que se pueden realizar directamente desde la plataforma. Destaca por poder proporcionar respuestas personalizadas sobre casi cualquier sitio del mundo.



F.4: Expedia chatbot

BlueBot KLM

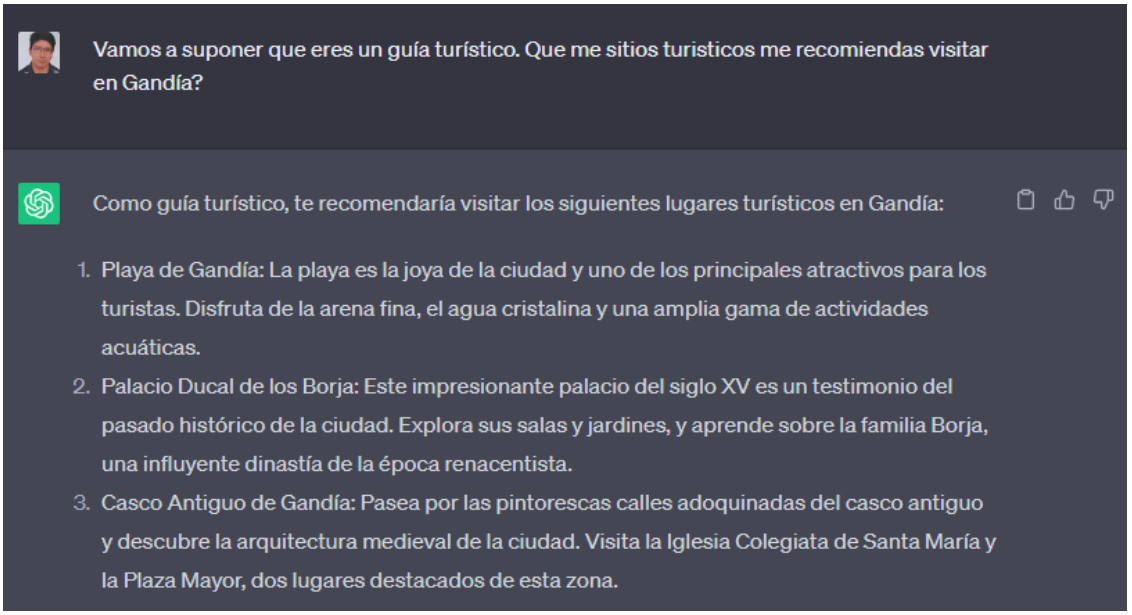
KLM, una aerolínea de los Países Bajos ha desarrollado BlueBot, un chatbot capaz de realizar reservas de vuelos y proporcionar una asistencia completa durante todo el proceso, proporcionado las mejores opciones según cuales sean tus necesidades. Cuenta con una integración con Google Home lo que la hace mucho más funcional. Además de tener asistencia incluso para ayudarnos a hacer la maleta que nos hará distintas recomendaciones según cual sea nuestro destino.



F.5 BlueBot KLM

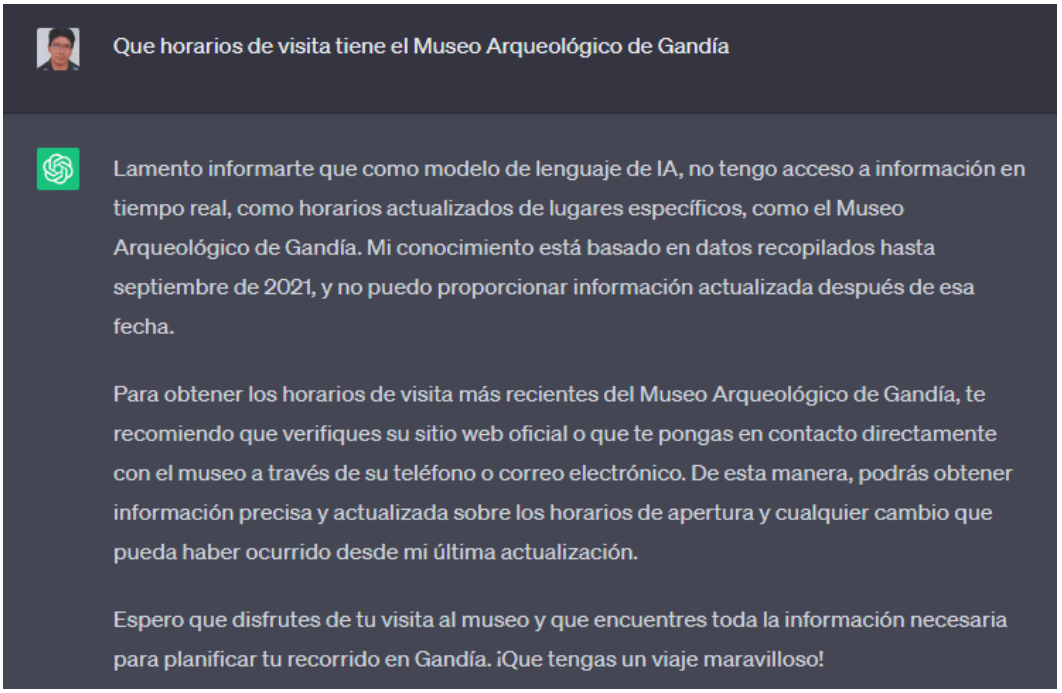
ChatGPT

Como ya se ha mencionado anteriormente, ChatGPT puede hacer todo tipo de tareas, y como no podía ser menos también puede ser un gran guía turístico. Basta con proporcionarle un par de indicaciones básicas y este nos proporcionará información muy interesante respecto a lo que podemos hacer en Gandía.



F.6: Chat GPT: Ejemplo de uso

Como se puede observar en este ejemplo, las respuestas son bastante acertadas y nos proporcionan una descripción muy detallada de los sitios que podemos visitar. Sin embargo no todo es perfecto como podemos ver a continuación:



F.7: Chat GPT: Ejemplo de uso 2

Su mayor limitación está presente a la hora de acceder a información actual. Chat GPT solo dispone de información hasta septiembre de 2021, por lo que cualquier pregunta que haga referencia a un periodo posterior a este, no sabrá responderlo. En este caso estamos preguntando por horarios de visita y esto puede variar mucho dependiendo de diversos factores, es por ello que aquí Chat GPT pierde gran parte de su fiabilidad.

3. Propuesta

En este capítulo se va a explicar cuál ha sido el proceso que se ha seguido para el desarrollo de este chatbot. Como primer punto se analizará los requisitos de la propuesta para posteriormente detallar el procedimiento que se ha seguido en cada una de sus fases de desarrollo.

3.1 Requisitos

La principal necesidad que pretende cumplir este chatbot es la de asistir a los trabajadores del sector turístico de Gandía. Durante el verano Gandía recibe un gran número de turistas que a menudo necesitan una pequeña guía para poder orientarse en Gandía o simplemente necesitan saber que actividades turísticas se pueden hacer aquí.

Es por ello que surge la necesidad de disponer de un asistente virtual que permita tener una asistencia completa las 24h del día, que sea accesible desde cualquier lugar y que permita tener actualizado a todos los usuarios de las últimas novedades sobre el turismo de Gandia.

Partimos de un documento que nos ha proporcionado el ayuntamiento de Gandia. Este documento consta de 59 preguntas y respuestas, donde aparecen las preguntas que mayor se realizan a los servicios turísticos.Conjuntamente a estas preguntas también será necesario incorporar expresiones auxiliares que el usuario pudiera hacer para comunicarse con el chabot tales como: saludar, afirmar, negar, agradecer, despedirse etc. De igual manera es necesarios añadir algunas preguntas adicionales que puedan responder información básica sobre el chatbot y otro tipo de cuestiones que ayudarán al chatbot a tener una personalidad más humana.

En cuanto a los requisitos técnicos, serán necesarios incorporar técnicas de comprensión del lenguaje (NLU) y métodos que ayuden a la toma de decisiones en función del contexto. De igual manera será importante disponer de una buena interfaz que permita interactuar con el usuarios de la manera más intuitiva posible, siendo primordial que sea accesible desde cualquier dispositivo móvil o portátil.

A sí mismos de cara a poder realizar mejoras sobre el chatbot sería interesante disponer de una interfaz que nos permita analizar las conversaciones que ha tenido el chatbot con los usuarios, tanto si han sido exitosas como si no lo han sido.

El objetivo final del chatbot es conseguir tener una interacción lo más personal posible, demostrando interés en la conversación, dominio sobre los temas que está hablando y una gran capacidad de entendimiento en lo que al usuario se refiere. Todo esto ayudará a que se produzca una conversación con sentido y fluida que parezca no proceder de un robot.

3.2 Diseño

Una vez se han definido los requisitos, se procede a explicar la solución propuesta para el desarrollo del chatbot.

3.2.1 Arquitectura de la aplicación

Como es habitual en este tipo de desarrollos, se seguirá la arquitectura de cliente-servidor, teniendo en cuenta que Rasa tiene altos costos computacionales y de almacenamiento, en el servidor se alojan todos los servicios necesarios para el correcto funcionamiento de Rasa y de la web. El back-end estará compuesto por los siguientes elementos:

- Base datos: La BD que almacenará todas las conversaciones para su posterior análisis
  .
- Server API REST: El servidor API REST será el encargado de comunicarse con la base de datos, tanto para almacenar las conversaciones como para acceder a su historial. Con el uso de una API conseguimos tener una capa adicional que proporcionará mayor seguridad y escalabilidad a la parte del back-end.

- Rasa conversational: El servidor de Rasa será el encargado de interpretar los mensajes de los usuarios (NLU) y proporcionar una respuesta coherente en cuestión.

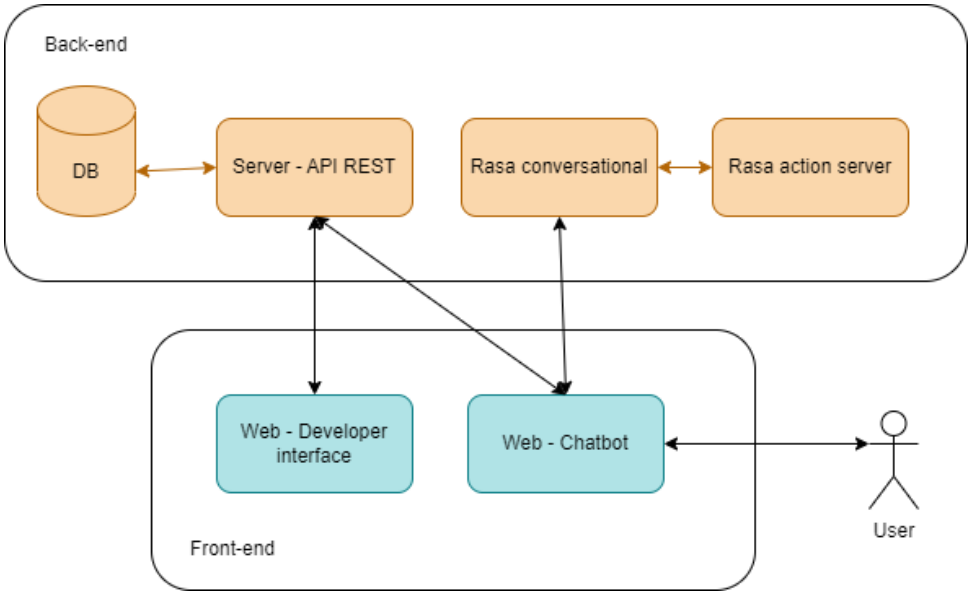
- Rasa action server: El servidor de acciones de Rasa tiene como objetivo proporcionar funciones adicionales al funcionamiento estándar de Rasa, como puede ser dar la bienvenida al usuario o proporcionar alguna respuesta más compleja ya sea porque depende de otro tipo servicios.

En cuanto a la parte del front - end, en una primera instancia se pretendía utilizar un servicio proporcionado por Rasa, denominada Rasa X. Este servicio nos proporciona una web completamente funcional que nos permite acceder al histórico de las conversaciones, pudiendo incluso añadir nuevas historias al chatbot desde la web.Sin embargo, este servicio con el paso del tiempo se ha vuelto solo de uso empresarial, por lo que dejaba de ser un servicio gratuito.

Como alternativa, se planteó realizar una web para los desarrolladores de Rasa que nos permite emular las funciones de este servicio web. A lo que se ha denominado Conversa-Rasa. (este desarrollo es una continuación del desarrollo inicial realizado por Joan Cipria) Con lo que se distinguen las siguientes partes. Web - chatbot y web - developer interface. Se hace esta distinción ya que la web para mantener conversaciones con el chatbot es de dominio público y cualquier persona podrá acceder a esta web.

- Web - Chatbot: Página web a la que accede el usuario para realizar las preguntas al chatbot. Esta web tendrá que ser lo más amigable e intuitiva posible ya que será la que tiene contacto con el usuario.

- Web - Developer interface: Esta interfaz es la que accede el desarrollador para analizar las preguntas del chatbot. Esta parte contará con un login de contraseña simple y una web desde la que se podrá ver el historial de conversaciones.

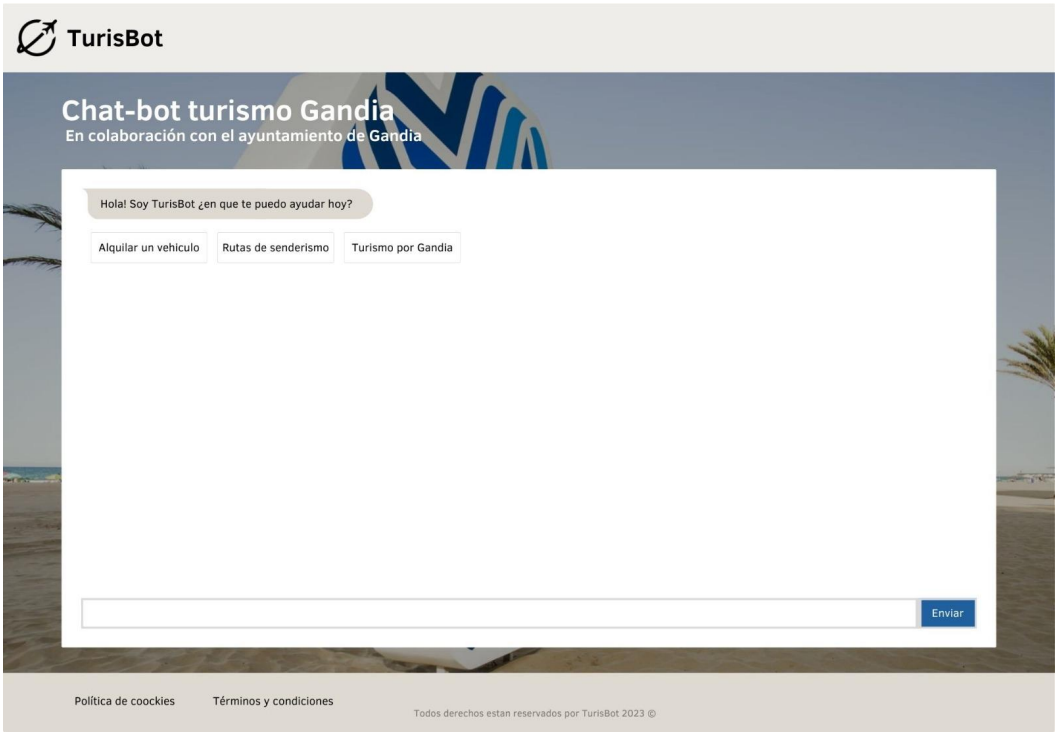


F.8: Diagrama funcionamiento

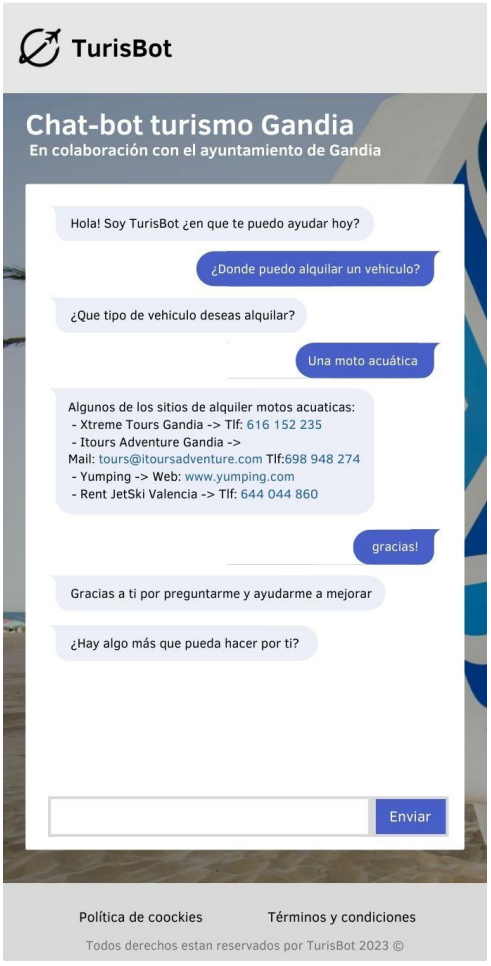
3.2.2 Mockups

Web - Chatbot

Como ya se ha comentado anteriormente el principal objetivo del chatbot es que sea lo más accesible para el usuario, tanto de un dispositivo móvil como desde un ordenador, por lo que a continuación se presentan los 2 diseños iniciales de la web del chatbot.



F.9 Mockup PC: Web - Chatbot

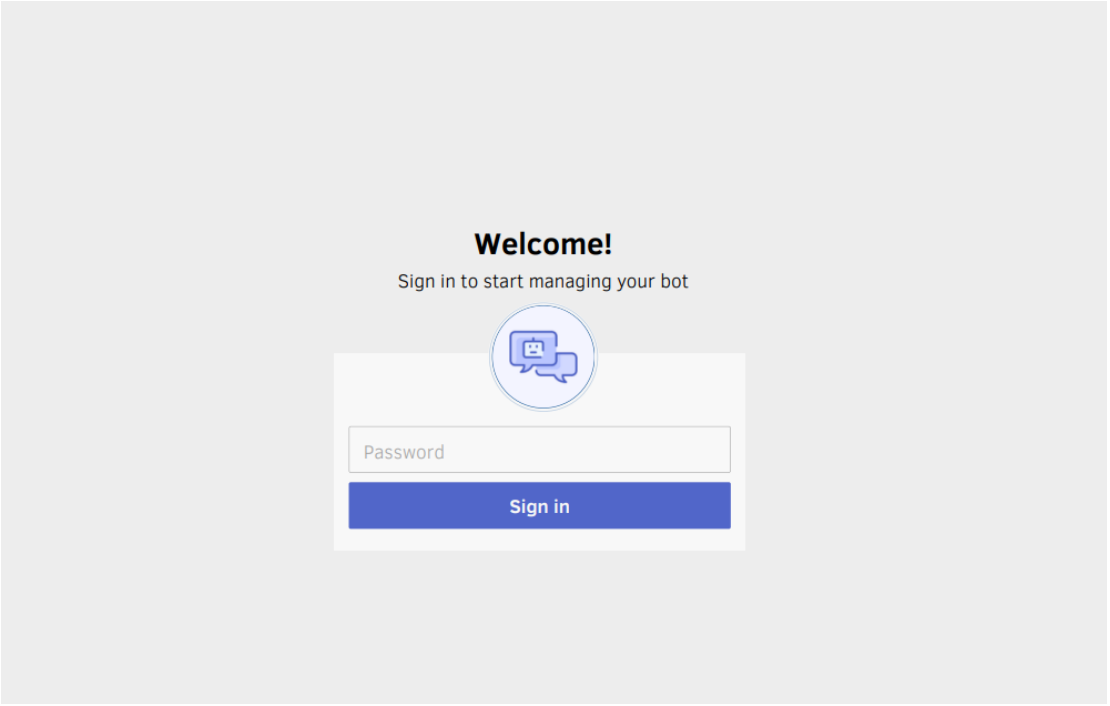


F.10 Mockup móvil: Web - chatbot

Como se puede observar en la figura 8, nada más entrar en la página web el chatbot te dará la bienvenida y te proporcionará 3 de las preguntas más frecuentes que realizan los turistas a modo de botones. Con esto damos una primer acercamiento al usuario, en caso de que no sepa cómo iniciar la interacción con el chatbot.

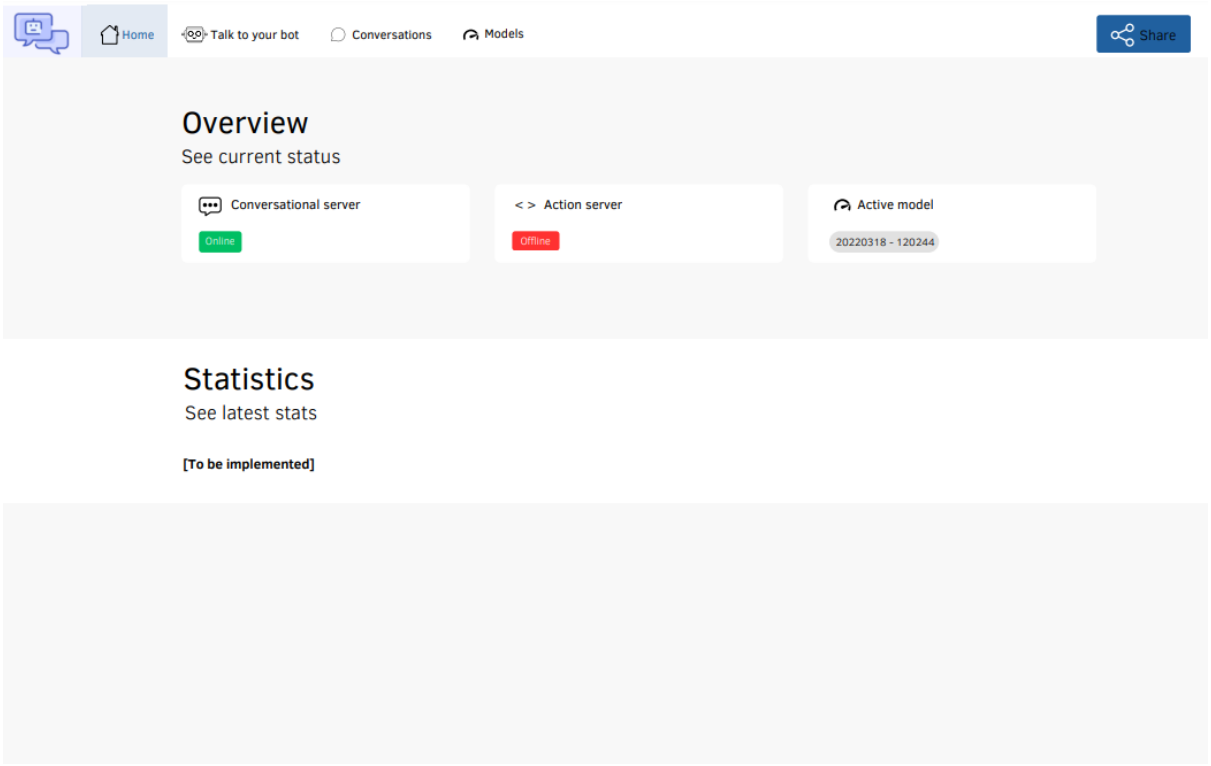
Web - Developer interface

Por otra parte también es necesario disponer de una web que nos permite acceder a las conversaciones del chatbot, teniendo un registro de todas las conversaciones que se han producido. Por lo que se ha planteado el siguiente diseño:



F.11: Mockup Welcome - Web Developer interface

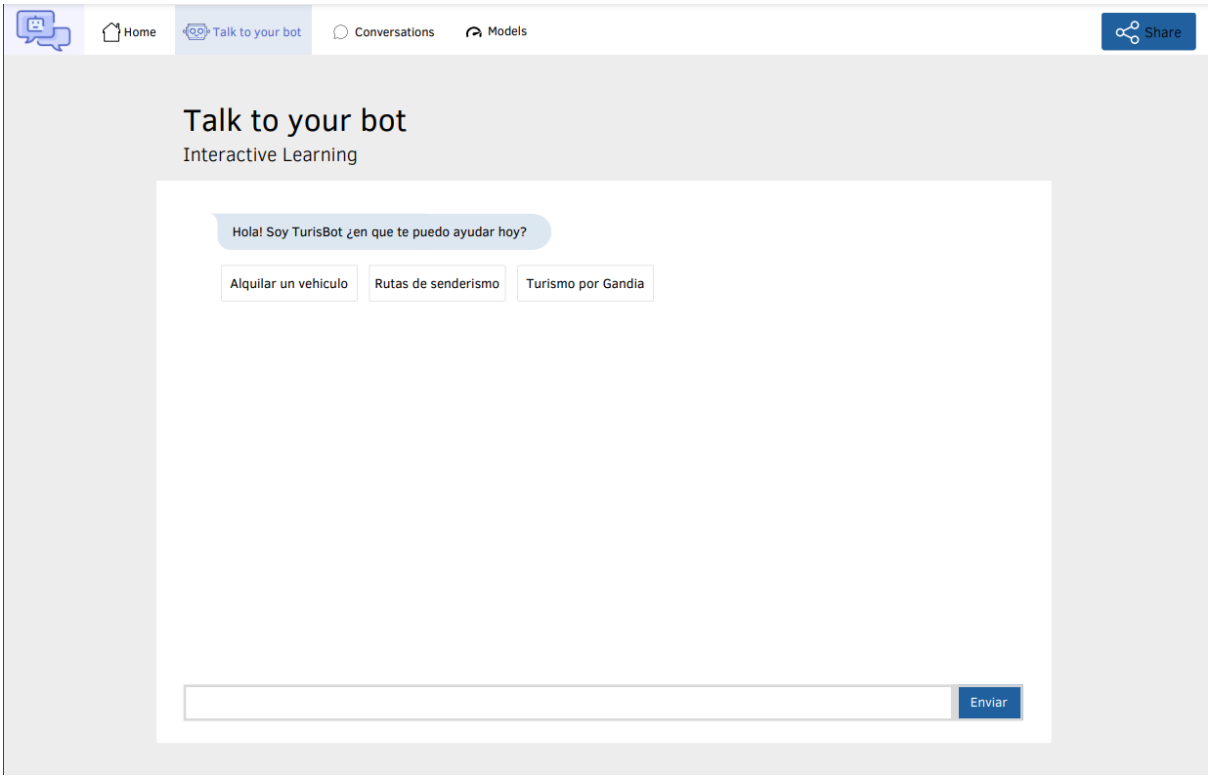
La página para acceder al panel de control del desarrollador solo contará con un campo de contraseña que se deberá rellenar para acceder. Si el servidor detecta que no tienes ninguna instalación de Rasa en el servidor, este te pedirá que establezcas una contraseña para este servicio y acto seguido te llevará a la página de Home



F.12: Mockup Home - Web Developer interface

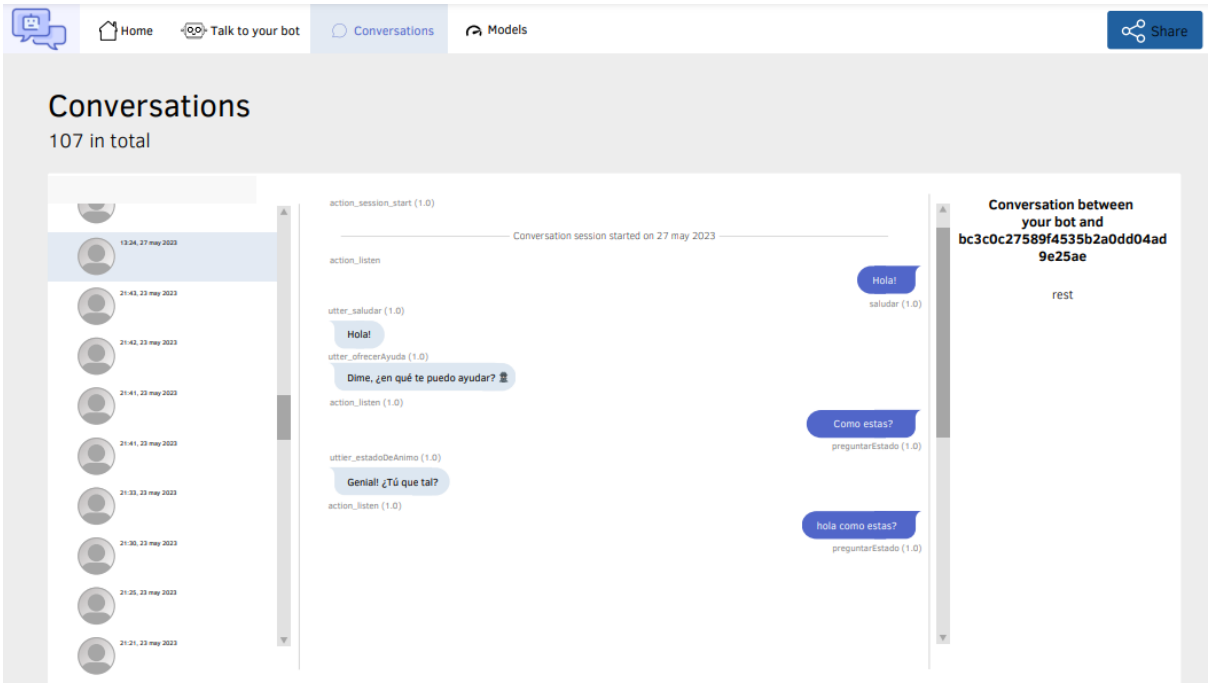
Esta página pretende mostrar un resumen del estado actual del chatbot indicando el estado actual del chatbot, si tiene los servicios iniciados (conversational y actions) e indicar el modelo con el que se está trabajando. Además en futuros desarrollos está previsto que muestre estadísticas comparando los distintos modelos o similares, con el fin de poder mejorar el desempeño del chatbot.

Esta web contará con un nav-bar desde el que podremos acceder a las distintas secciones de nuestra web. En la esquina superior derecha contamos con un botón share, que nos llevará a un página para poder compartir nuestro chatbot con cualquier usuario.



F.13: Mockup Talk to your bot - Web Developer interface

La sección Talk to your bot, pretende ser una sección para que los desarrolladores puedan interactuar con el bot, sin necesidad de salir del apartado de devolper interface.



F.14: Mockup Conversation- Web Developer interface

Y finalmente tendremos la página de conversations que nos permitirá analizar cada una de las conversaciones que ha tenido el chatbot y evaluar si estas han tomado el camino esperado o no. Además, cada una de las respuestas proporcionadas por el chatbot presenta un indicador de fiabilidad similar a: uttier_estadoDeAnimo (1.0) Esto nos indica cuál ha sido la fiabilidad con la que el robot a proporcionado esta respuesta, siendo 1.0 el 100% de fiabilidad.

Cabe destacar que la parte de web para el desarrollador está pensado para que sea usada por cualquier usuario que tenga un chatbot en Rasa, por lo que el desarrollo se ha realizado en inglés.

3.3 Implementación

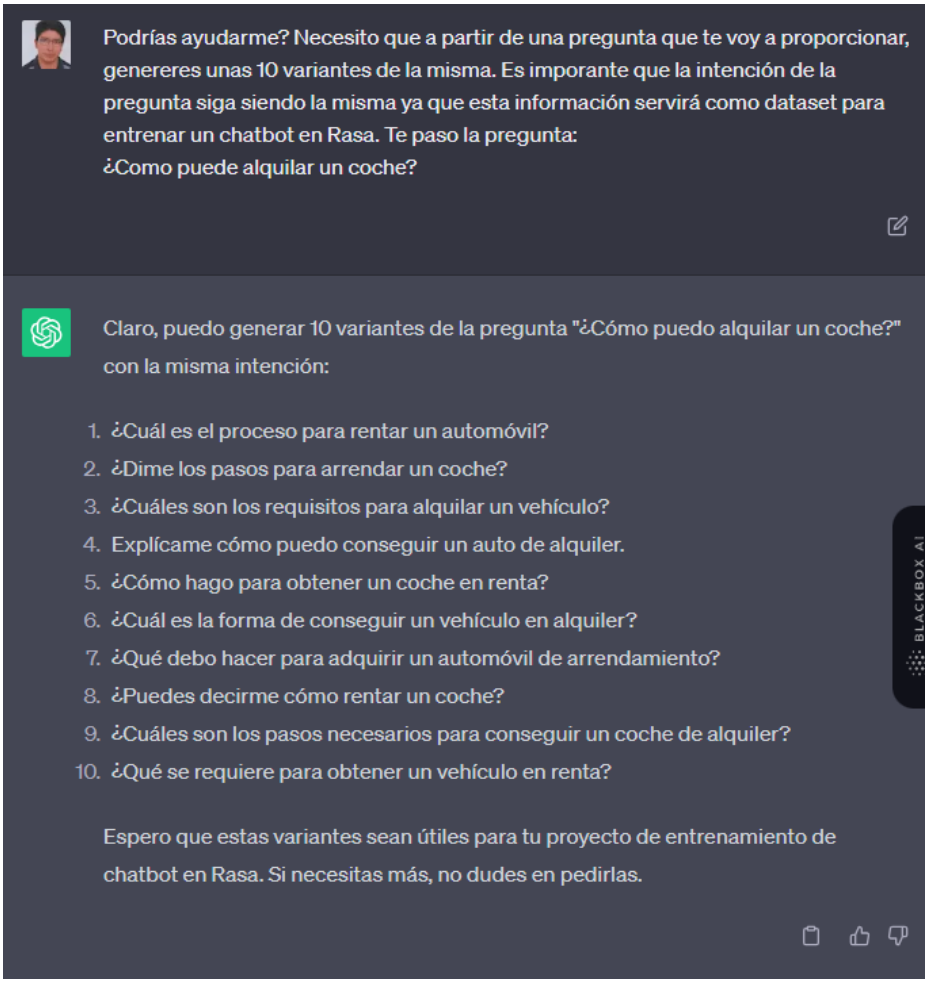
La implementación de este proyecto se inicia con el desarrollo del chatbot en rasa, ya que el objetivo es disponer de un prototipo mínimo viable lo antes posible. Con esto se busca exponer al chatbot a los usuarios en el menor tiempo posible, con lo que se podremos realizar las mejoras que sean necesarias.

3.4.1 Rasa

Rasa es una herramienta que nos permite desarrollar un chatbot con apenas unas cuantas líneas de código, pero antes de comenzar con este desarrollo son necesarios el entendimiento de algunos conceptos de cómo funciona Rasa.

Dataset

El dataset representa el conjunto de preguntas y respuestas con las que puede interactuar el chatbot. Es importante partir de un dataset inicial con información sobre algunas interacciones básicas como son saludar, despedirse, agradecer, negar etc También se ha de tener en cuenta el ámbito de uso del chatbot, que en este caso será el sector turístico, para poder crear una clasificación de intentes que nos permita un fácil mantenimiento y futuro desarrollo. En este caso, se ha distribuido el dataset en 4 ámbitos. interacción básica, preguntas sobre turismo, preguntas sobre servicios y preguntas relacionadas con la playa de Gandia. Esta clasificación se ha creado teniendo en cuenta el volumen de preguntas de cada uno de estos ámbitos y la posible relación entre las preguntas.
Cabe destacar que cuantas más versiones tengas de una pregunta, mayor será la fiabilidad de Rasa para identificar este intent, por lo que con el objetivo de mejorar la clasificación de las preguntas, se ha hecho uso de ChatGPT, que dada una pregunta este nos generaba varias versiones de la misma, obteniendo así un dataset inicial mayor.



F15. Generación de dataset con ChatGPT

Aunque se ha intentado ser lo más rigurosos posibles a la hora de formularle la pregunta a ChatGPT, este aún presenta sus limitaciones. Como se puede observar en la mayoría de preguntas de ChatGPT utiliza un lenguaje poco habitual, haciendo prácticamente inutil este dataset. Aun con todo, se ha partido de estas preguntas y se les ha realizado las modificaciones pertinentes para que estas preguntas sean válidas y se asemejan más a la realidad.

Intents

Los *intents*, como bien su nombre indica, son intenciones que el usuarios desea hacer mediante la interacción con el chatbot. Pongamos un ejemplo, si el usuario quiere alquilar un coche, esto se considera un *intent,* una intención que el usuario pretende hacer. Es importante agrupar todos las intenciones de un usuario para evitar replicar preguntas. Por ejemplo si un usuario quiere alquilar un vehículo. Ej: *¿Dónde puedo alquilar un coche?, Quiero alquilar una moto acuática, ¿donde puedo hacerlo ?* Ambas preguntas representan el mismo *intent*, la intención de alquilar un vehículo, por lo que es importante saber identificar este tipo de *intents*. Más adelante se hablará de las entidades, que nos permitirá distinguir entre sí lo que queremos alquilar es una moto acuática, o un coche.

```
- intent: preguntarPorAlquilerVehiculo
```

```
  examples: |
    - ¿Dónde puedo alquilar un coche?
    - Quiero alquilar un coche.
    - Me gustaría saber si tienen coches disponibles para
alquilar.
    - ¿Dónde puedo encontrar coches disponibles para alquilar?
…
```

Response

La definición de las respuestas es clave a la hora de determinar la sensación que proporcionan al usuario. Un gran abanico de respuestas, como por ejemplo a la hora de saludar, denota una mayor sensación de humanidad.

```
  utter_saberMas:
  - text: ¿Te gustaría saber alguna cosa más?
  - text: ¿Te gustaría preguntarme alguna cosa más?
  - text: ¿Qué más te gustaría preguntarme?
```

Stories/Rules

La manera que tiene Rasa de relacionar las preguntas y las respuestas es mediante la definición de las stories y las rules. Ambas sirven para definir, que dada una pregunta que respuesta se tiene que proporcionar. Su diferencia radica, en que una rule siempre devolverá la misma respuesta mientras que una story no. Las rules se utilizan cuando sabemos que la respuesta siempre es la misma independientemente del contexto en el que nos encontremos.¿Qué horario tiene el servicio de socorrismo?
En cambio si lo que queremos es que se tenga en cuenta el contexto de la conversación, se tendría que definir mediante una storie.

```
- story: alquilar un vehiculo moto sin saber
  steps:
  - intent: preguntarPorAlquiler
  - action: utter_responderAlquiler
  - intent: responderVehiculo
    entities:
    - vehiculo:
  - slot_was_set:
    - vehiculo: moto
  - action: utter_responderAlquilerMoto
```

En el ejemplo superior, se está realizando una pregunta. ¿Dónde puedo alquilar?
A lo que nosotros tendremos que responder en cuestión dependiendo si sabemos el vehículo al que se refiere o no. Por lo que en este caso, existen varias stories para llegar a la misma respuesta final.

Entidades

Las entidades existen para proporcionar un poco más de dinamismo tanto a las preguntas como a las respuestas. Partiendo del ejemplo anterior, *¿Donde puedo alquilar un coche?, ¿Donde puedo alquilar una moto?* Ambas preguntas presentan la misma intención pero claramente la respuesta proporcionada ha de ser diferente en cada caso. Para ello se ha utilizado las entidades. Un tipo de variables que podemos definir dentro de nuestro chatbot que podrán almacenar un número de valores previamente definido. En este caso, nuestra entidad será vehículo. Cuando se define una entidad es importante definir el tipo de valores que esta puede tomar.

```
entities:
  - vehiculo


slots:
  vehiculo:
    type: categorical
    values:
      - "bici"
      - "coche"
      - "moto"
      - "moto acuatica"
...
```

Una vez se ha definido una entidad que permite almacenar distintos valores, se ha de definir esta entidad en el intent

```
- intent: preguntarPorAlquilerVehiculo
  examples: |
    - ¿Dónde puedo alquilar un
[coche]{"entity":"vehiculo","value":"coche"}?
      - ¿Dónde puedo alquilar una
[bicicleta]{"entity":"vehiculo","value":"bici"}?
```

Seguidamente a la hora de proporcionar la respuesta, en las stories, se puede definir que respuesta se tiene que proporcionar en cada caso.

```
- story: alquilar un vehiculo coche
  steps:
  - intent: preguntarPorAlquilerVehiculo
    entities:
    - vehiculo
  - slot_was_set:
    - vehiculo: coche
  - action: utter_responderAlquilerCoche
```

Mediante la llamada slot_was_set, comprobamos que valor ha sido almacenado en la entidad vehículo y proporcionamos una respuesta coherente en cuestión. Del mismo modo se puede definir una respuesta genérica en caso de que el valor de la entidad no sea ninguno de los que hemos definido.

Rasa actions

Rasa presenta sus propias limitaciones a la hora de acceder a otro tipo de recursos. Pongamos el ejemplo de la siguiente pregunta. *¿Qué tiempo hará mañana?* En este caso, para responder la pregunta necesitamos acceder a un servicio que nos diga la previsión de mañana mediante un servicio API o similar. Para este tipo de recursos Rasa permite definir una serie de acciones que podrán ser utilizadas como si de una respuesta se tratase. Estas acciones se definen en un servidor independiente a Rasa y su desarrollo se realiza en Python, por lo que permite acceder a casi cualquier recurso que necesitemos.

3.4.2 Back-end

Una vez se ha desarrollado completamente el modelo en Rasa, el siguiente paso, consiste en la implementación del servicio en una página web que nos permita mantener una conversación con el chatbot. Para ello es necesario el desarrollo de una API que nos permita realizar esta comunicación. De igual manera si lo que pretendemos es disponer de una web para los desarrolladores que nos permita analizar las conversaciones que se han ido registrando, también es necesario el desarrollo de un servicio que nos permita acceder y almacenar toda esta información.

API Rest

Se ha optado por la creación de un servicio API Rest que nos permita realizar todas las consultas necesarias para acceder al registro de conversaciones del chatbot y el guardado de conversaciones del mismo. Para la implementación de esta API se ha utilizado Flask un framework desarrollado en Python, que nos permite crear un servicio web de manera rápida y sencilla con apenas unas cuantas líneas de código.

Mediante el servicio API también se crearán las configuraciones iniciales necesarias para el servidor, como por ejemplo es el caso de la base de datos. La primera vez que accedas al servicio, este comprobará mediante la llamada /firstrun si existe una base de datos creada previamente. En caso negativo, la web te pedirá un ruta y esta será donde se crean almacenará toda el registro de las conversaciones con el chatbot.

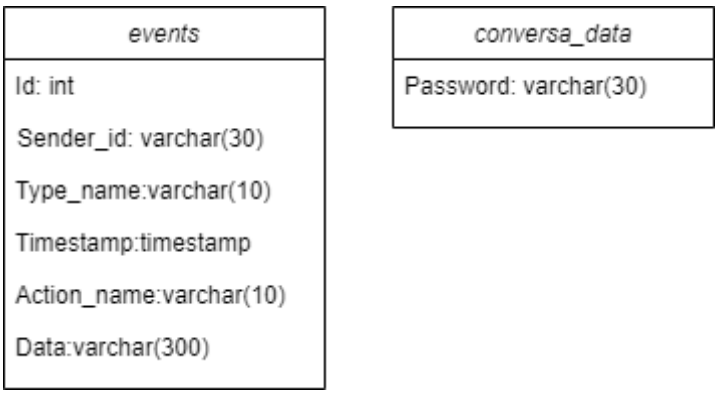
Ruta	Resultado	Definición
/conversations	JSON: [conversation]	Devuelve todos las conversaciones del chatbot
/conversation/:id	JSON: conversation	Devuelva una conversación según el id
/firstrun	boolean	Comprueba si es la primera vez que se inicia el servicio en el servidor

/checkpath/:path	boolean	Comprueba si la ruta de la base de datos ya ha sido creado
/install	-	Crear la nueva base de datos en el servidor
/login/:password	boolean	Verifica la contraseña del usuario

F.16: Rutas API Rest

Base de datos

Siguiendo con el planteamiento que se ha comentado anteriormente, la base de datos solo consta de 2 tablas. Una que guardará la contraseña para acceder al registro de las conversaciones en el servidor y otra que almacenará el registro de todas las conversaciones que ha tenido el chatbot.



F.17: Esquema base de datos

El sistema de gestión de base de datos utilizado ha sido SQLite, una versión reducida de SQL que nos permite gestionar una base de datos de una manera muy simple.

3.4.3 Front-end

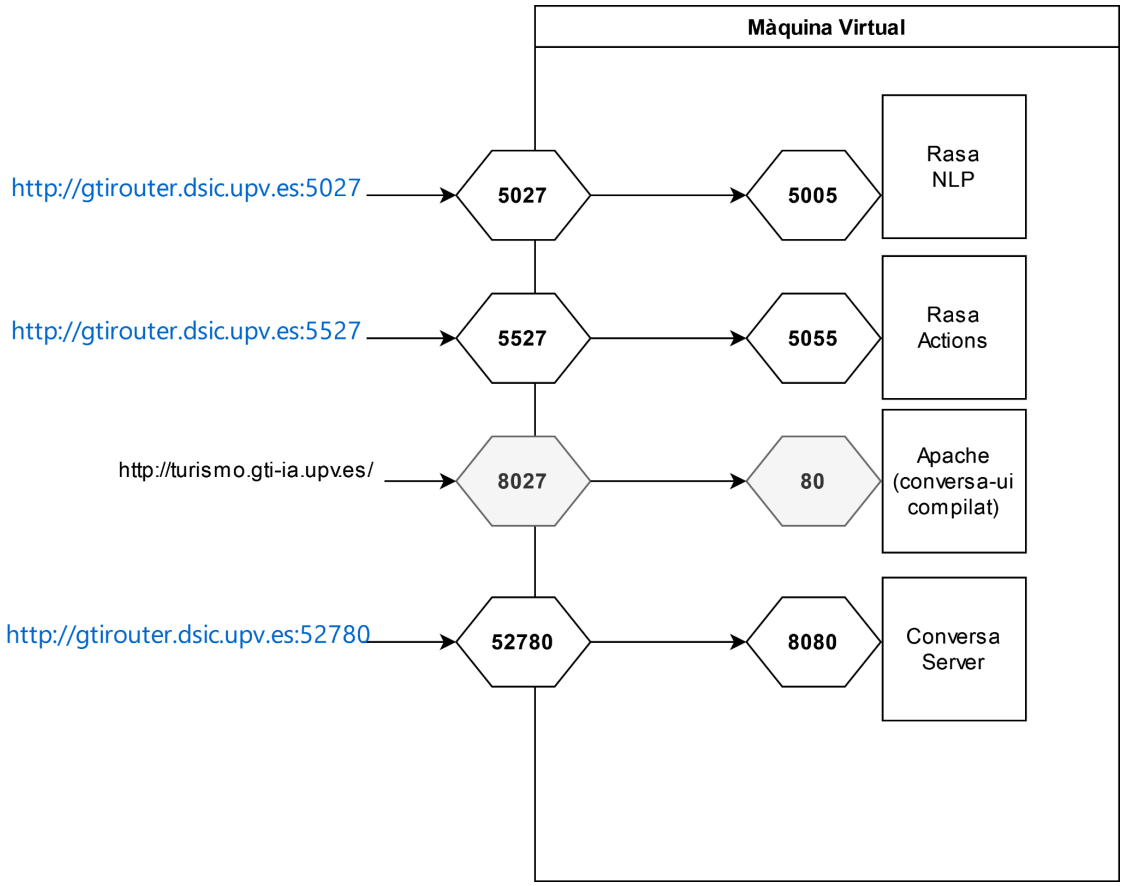
Para el desarrollo de la parte del front-end se ha utilizado Vue. Un framework desarrollado en JavaScript utilizado para la creación de interfaces. La principal ventaja que presenta este framework es la posibilidad de reutilizar componentes a la largo de toda la estructura de la página, siendo posible completar estos componentes mediante HTML,CSS y JavaScript. La interfaz seguirá los diseños y el funcionamiento explicados en la sección 3.2.2, implementando un diseño responsive aunque está previsto que su acceso sea principalmente desde un ordenador.

Desde la parte de front-end se realizará la llamada a los distintos servicios API Rest desarrollados anteriormente mediante un servicio *HHTTP* que comunicará el servidor la página web. Además todas las páginas de la parte *developer interface* tendrán una redirección automática hacia el login para evitar que usuarios no deseados entren a este sección.

3.4.4 Despliegue del prototipo

Llegados a este punto, solo queda el despliegue del prototipo en un entorno real. Para ello la Universidad Politécnica de Valencia nos ha facilitado una máquina virtual, donde podremos realizar un primer despliegue. Esta es un máquina Ubuntu, a la cual se ha accedido mediante SSHH para poder realizar la instalación de todas las librerías necesarias.

Ante las necesidades que presenta nuestro prototipo se ha tenido que solicatar que nos abran los puertos 80 (página web), 5005 (rasa conversational), 5055 (Rasa actions), 8080 (Conversa-server)



F.18: Diagrama de redirección de puertos

3.4.5 Control de daños

Finalmente se comenta las prácticas que se han seguido para evitar el mal funcionamiento del chatbot.

Fallback

Es muy probable que en algún momento de la conversación el chatbot, este no sepa cómo responder a una determinada pregunta del usuario, ya sea porque esta no está formulada correctamente o bien porque el chatbot simplemente desconoce esta pregunta.

Es por ello que en Rasa existe un margen denominado *fallback* el cual puedes establecer entre 0 y 1, y si la pregunta del usuario presenta un valor de entendimiento de Rasa por debajo de este valor, Rasa devolverá esta respuesta predefinida.

F.19: Control de errores

Cabe mencionar que esto no siempre será lo mejor ya que quizás el usuario no ha formulado bien la pregunta o la puede formular de otra manera para que el chatbot sea capaz de interpretarla. Por lo que en la mayoría de ocasiones se pedirá al usuario si nos puede formular la pregunta de otra manera.

Tests

Rasa también cuenta con una herramienta de tests automáticos que nos permiten comprobar si nuestro modelo entrenado es capaz de seguir una conversación de la manera en la que nosotros esperamos que lo haga. Para ello se utiliza un lenguaje muy similar al de las stories:

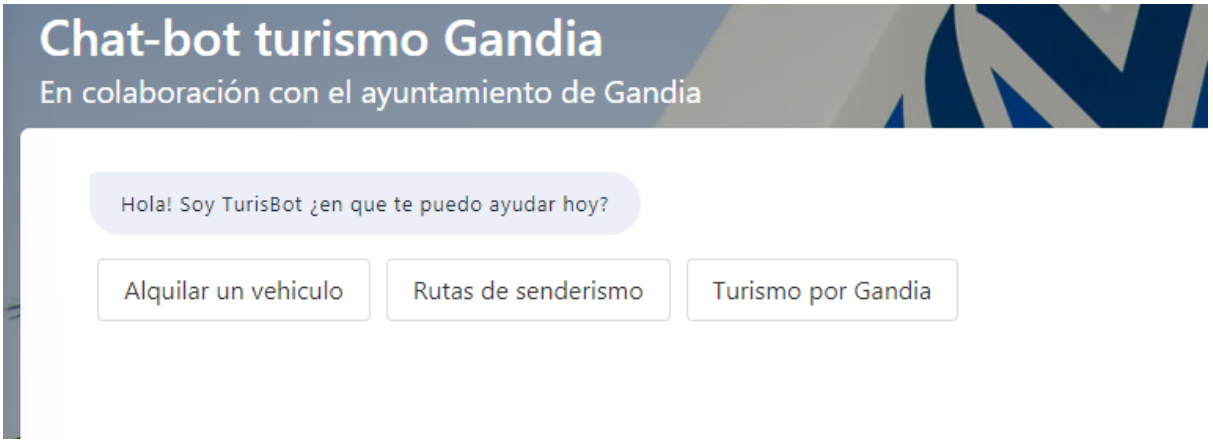
```
- story: saludar + como estas
steps:
- intent: saludar
user: |-
Hola!
- action: utter_saludar
- action: utter_como_estas
- intent: mostrar_bienestar
user: |-
muy bien gracias!
```

```
- action: utter_mostrar_alegria
- action: Ofrecer servicios
```

Este tipo de tests se suele utilizar mayoritariamente tras el entrenamiento de cada modelo, ya sea bien porque se han añadido nuevas stories al dataset o porque se han modificado parámetros preestablecidos para el entrenamiento. Con lo que conseguimos saber si nuestro nuevo modelo es funcional o no.

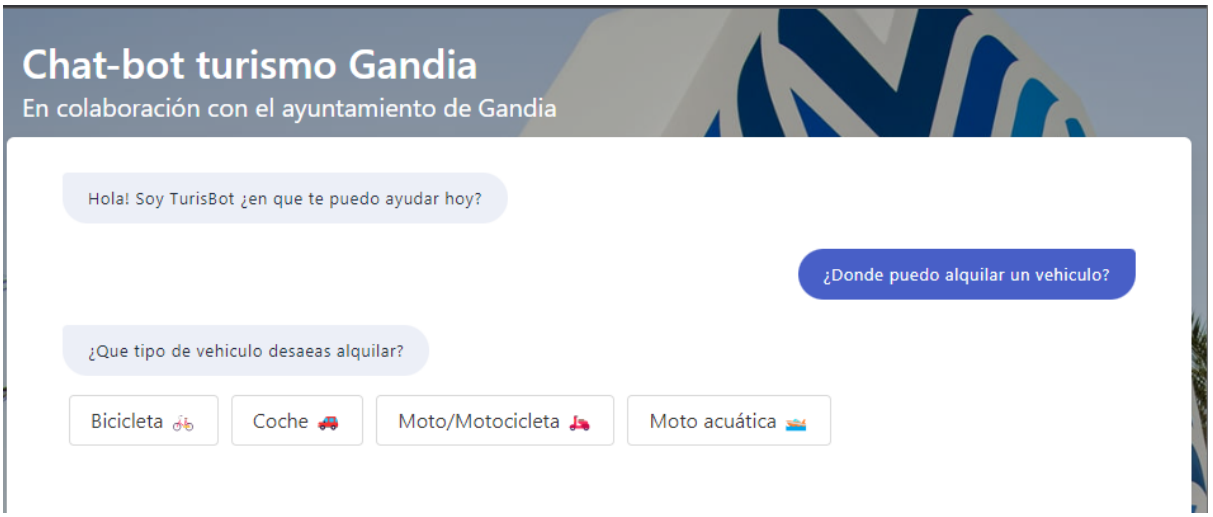
Saber llevar una conversación

Una de las principales características del ser humano a la hora de mantener una conversación, es la capacidad que tiene este para poder iniciar la misma. Esto la gran mayoría de veces lo hacemos de manera subconsciente, pero a la hora de desarrollar un chatbot es algo importante a tener en cuenta. Es por ello que en todo momento se ha buscado que el chatbot incentive al usuario a mantener una conversación. Esto se ha conseguido mediante la implementación de los botones que se muestran al iniciar la conversación con el chatbot.



F.20: Iniciar conversación

De igual manera estos botones nos pueden servir para dirigir la conversación del usuario al punto que queremos. Como por ejemplo si el usuario pregunta, *¿Donde puedo alquilar un vehículo?*. Esta respuesta puede estar predefinida por nuestra parte, y así evitamos que el usuario introduzca otro tipo de respuesta que nosotros no hemos contemplado.



F.21: Control de respuestas

Así mismo también es importante, después de cada interacción con el usuario proporcionar una respuesta amable que incentive al usuario a mantener la conversación.

4. Conclusiones

Este ha sido un proyecto muy exigente en lo que se refiere a utilización de nuevas tecnologías, ya que en su mayor medida, se han utilizado tecnologías que no se han visto durante la carrera. Esto ha hecho un proyecto mucho más ambicioso y por tanto de mayor dificultad. Sin embargo se han conseguido gran parte de los objetivos iniciales sobre este proyecto, sobre todo en lo que respecta al desarrollo de la parte software.

Las mayores carencias de este proyecto están presentes en la parte de testeo del prototipo, ya que debido a diversos inconvenientes, no se ha podido completar esta fase, habiendolo probado solo los usuarios más cercanos de mi entorno. Aún así esta pequeña fase ha sido suficiente para proporcionar un pequeño feedback y realizar las mejoras pertinentes en lo que respecta a la manera de comunicarse con el chatbot.

Desde un punto de vista global, el resultado del proyecto ha sido muy satisfactorio sobre todo en lo que respecta al conocimiento adquirido en este proceso. Debido a la gran diversidad de tecnologías utilizadas y la complejidad del mismo, ha requerido de una formación constante durante todas las fases del proyecto. Esto ha acabado proporcionándome aptitudes y conocimientos que sin duda me ayudaran en mi futuro profesional.

Las líneas de mejora que presenta este proyecto, están claras, sobre todo en lo que respecta a la fase de testeo. Este ha sido un proyecto en colaboración con grado de turismo y el ayuntamiento de Gandia, por lo que todavía queda un gran camino antes de que este llegue a ser un producto final, pero creo que este prototipo presenta una clara visión del potencial de este proyecto.

5. Referencias

1. Dias e Cordeiro, I., & da Silva Batista, I. M. (2020). La experiencia del usuario en el proceso de adquirir información para planear el viaje el caso del chatbot de Kayak. *Estudios y perspectivas en turismo*, *29*(3), 792-816. URL: http://www.scielo.org.ar/scielo.php?pid=S1851-17322020000300792&script=sci_arttext&tlng=en

2. Adamopoulou, E., & Moussiades, L. (2020). An overview of chatbot technology. In *Artificial Intelligence Applications and Innovations: 16th IFIP WG 12.5 International Conference, AIAI 2020, Neos Marmaras, Greece, June 5–7, 2020, Proceedings, Part II 16* (pp. 373-383). Springer International Publishing. URL: https://link.springer.com/chapter/10.1007/978-3-030-49186-4_31

3. Gnewuch, U., Morana, S., Adam, M., & Maedche, A. (2018). Faster is not always better: understanding the effect of dynamic response delays in human-chatbot interaction. URL: https://web.archive.org/web/20200322152504id_/https://aisel.aisnet.org/cgi/viewcontent.cgi?article=1112&context=ecis2018_rp

4. GitHub Chabot turismo Gandia URL: https://github.com/JuanC99/ChatBotTurismoGandia

5. GitHub Web Conversa: https://github.com/conversa-rasa/conversa-ui/tree/develop_interaccion_chatBot

6. GitHub Conversa server: https://github.com/conversa-rasa/conversa-server

7. Moreno Teodoro, JC. (2021). VIHRTUAL-APP: un chatbot para la divulgación médica del VIH. Universitat Politècnica de València. URL:

   http://hdl.handle.net/10251/171268

8. Perez-Soler, S., Juarez-Puerta, S., Guerra, E., & de Lara, J. (2021). Choosing a chatbot development tool. *IEEE Software*, *38*(4), 94-103. URL: https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=9364349