UNIVERSITAT POLITÈCNICA DE VALÈNCIA

ESCOLA POLITÈCNICA SUPERIOR DE GANDIA

Grado en Tecnologías Interactivas

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

ESCOLA POLITÈCNICA SUPERIOR DE GANDIA

“VIHRTUAL-APP: Un chatbot para la divulgación médica del VIH”

TRABAJO FINAL DE GRADO

Autor/a:
Joan Ciprià Moreno Teodoro

Tutor/a:
Víctor Sánchez Anguix
Juan Miguel Alberola Oltra
Félix Gutiérrez Rodero

GANDIA, 2021

Resumen

La inteligencia artificial (IA) está transformando el mundo actual y la manera en la cual nos comunicamos con las máquinas. El lenguaje natural se está revelando como una forma muy óptima y competente de crear interfaces personalizadas que permitan a cada usuario interactuar utilizando sus propias palabras. Ante este nuevo escenario empezamos a ver como se popularizan los chatbots: programas diseñados para interactuar con los usuarios utilizando lenguaje natural. En el ámbito sanitario esto no ha sido una excepción y se siguen estos avances con gran interés en busca de asistentes virtuales que permitan mejorar y automatizar ciertos procesos médicos. En España, por ejemplo, en el año 2019 se notificaron 2.698 nuevos diagnósticos de VIH, de los cuales el 45.9 % presentaron diagnóstico tardío. Bajo este contexto surge VIHrtual-App, un proyecto que busca ayudar a detener la transmisión e informar ofreciendo un chatbot al público donde puedan obtener información veraz y relevante. En este documento se describe todo el proceso de estudio previo, diseño e implentación. Finalmente, se realiza una validación de usabilidad con usuarios reales, obteniendo unos resultados satisfactorios.

Palabras clave: chatbots, virus de la inmunodeficiencia humana, lenguaje natural, inteligencia artificial, salud.

Abstract

Artificial intelligence (AI) is transforming today's world and the way we communicate with machines. Natural language is proving to be very optimal to create customized interfaces which allow each user to interact with machines using their own words. In this new scenario, chatbots (programs designed to interact with users using natural language are growing in popularity. In health scope these advances are gathering interest interest to search virtual assistants to improve and automate certain medical processes. On the other hand, in 2019 2.698 new HIV diagnoses were reported in Spain, 45.9 % of them presented late diagnosis. With this context, VIHrtual-App is a project that seeks to help stop transmission and inform by offering a chatbot to the public where they can obtain truthful and relevant information. This document describes the whole process of preliminary study, design and implementation. Finally, a usability validation is carried out with real users, obtaining satisfactory results.

Keywords: chatbots, human immunodeficiency virus, natural language, artificial intelligence, health.

Índice general

Índice de figuras

Índice de tablas

Capítulo 1

Introducción

1.1. Introducción

La inteligencia artificial (IA) está transformando el mundo actual y la manera en la cual nos comunicamos con las máquinas, popularizándose cada vez más los asistentes y sistemas basados en voz. Esto es debido a que el lenguaje natural se está revelando como una forma muy óptima y competente de crear interfaces personalizadas que permitan a cada usuario interactuar utilizando sus propias palabras [1] . Ante este nuevo escenario empezamos a ver como se popularizan los chatbots: programas diseñados para interactuar con los usuarios utilizando lenguaje natural (por voz o texto) con el objetivo de hacer creer al usuario está hablando con una persona real.

En el ámbito sanitario esto no ha sido una excepción y se siguen estos avances con gran interés en busca de asistentes virtuales que permitan mejorar y automatizar ciertos procesos médicos [2]. No sólo se pretende obtener un beneficio inmediato al resolver las dudas más comunes y automatizar ciertas consultas, si no que a largo plazo esta alfabetización y acceso libre a la información puede contribuir a una sociedad más formada y consciente que mejore de manera activa su salud.

Por otro lado, en España, el virus de la inmunodeficiencia humana (VIH) continua representando un problema. Según la última estimación publicada por el Plan Nacional de Sida, en España viven un total de 151.387 personas con VIH. Sólo en el año 2019 se notificaron 2.698 nuevos diagnósticos de VIH, de los cuales el 45.9 % presentaron diagnóstico tardío [3]. Bajo este contexto surge este trabajo de final de grado. Un proyecto que busca ayudar a detener la transmisión e informar ofreciendo un servicio al público donde poder obtener información veraz y relevante. Un lugar dónde todas aquellas personas puedan resolver las dudas más comunes sobre el VIH y aprendan a tomar medidas de prevención. En definitiva, facilitar la mayor información posible a los usuarios para que estén protegidos y sean conscientes de la problemática actual. En este trabajo de final de grado se lleva a cabo el desarrollo de esta idea diseñando e implementando un chatbot completamente funcional que es accesible vía web.

Este proyecto está enmarcado dentro de VIHrtual-App, un proyecto de investigación con la colaboración de la Universitat Politècnica de València (UPV), la Fundación FISABIO y la Unidad de Enfermedades Infecciosas del Hospital General de Elche. La participación por parte de los colaboradores del hospital ha sido clave en el asesoramiento médico y veracidad de la información.

El presente documento describe todo el proceso que se ha seguido durante todo el desarrollo. Desde el estudio previo hasta su despliegue y validación.

1.2. Objetivos del proyecto

El objetivo principal de este proyecto es el desarrollo de un asistente virtual para la prevención del VIH. Los usuarios podrán interactuar con él mediante una interfaz conversacional y resolver sus

dudas. Para llevar a cabo esta tarea es necesario cumplir una serie de objetivos secundarios:

- Revisar el estado del arte sobre la creación de chatbots, obteniendo información sobre las técnicas actuales de diseño e implementación.
- Estudiar las características sociales en el diseño de la interacción entre humanos y chatbots.
- Analizar implementaciones similares de chatbots aplicados al campo de la salud.
- Detectar los requisitos del proyecto y analizar distintas herramientas para el desarrollo del asistente.
- Diseñar e implementar el servicio.
- Validar el proyecto con usuarios reales.

1.3. Estructura del documento

Capítulo I: Introducción
En el primer capítulo se realiza una breve introducción al proyecto, cuales son sus objetivos y la motivación que ha empujado a su desarrollo. Asimismo, también se describe la metodología de trabajo empleada durante toda su realización y una breve descripción de la estructura del documento.

Capítulo II: Marco Teórico
En este capítulo se pretende dar a conocer la base teórica sobre las cual se basa el proyecto: se describe el estado actual del arte, indaga en los aspectos a considerar en el diseño del chatbot y se analizan las distintas herramientas disponibles para la implementación.

Capítulo III: Desarrollo de la propuesta
En el tercer capítulo se muestra todo el proceso del desarrollo de la propuesta, desde el análisis de requisitos hasta el diseño e implementación del sistema.

Capítulo IV: Pruebas de validación
En este capítulo se realiza una evaluación final del sistema para comprobar que todos los requisitos iniciales planteados hayan sido satisfechos.

Capítulo V: Conclusiones
En sexto capítulo se realiza una valoración final del proyecto teniendo en cuenta las pruebas de validación.

Capítulo VI: Referencias
En este último capítulo se listan todas las referencias consultadas y citadas durante el desarrollo de este trabajo.

Capítulo 2

Metodología

En este capítulo se describe la metodología se trabajo seguida, tanto desde un punto de vista de gestión de proyecto como desde el punto de vista técnico.

2.1. Scrum

Durante todo el desarrollo se ha empleado la metodología ágil *Scrum*. Este método se caracteriza porque se realizan entregas parciales y regulares del producto final priorizando aquellas tareas que sean de mayor importancia para el resultado deseado. El desarrollo se ejecuta en ciclos temporales cortos y de duración fija (*Sprints*) y al finalizar deben proporcionar un incremento del producto que sea susceptible de ser entregado. Tras esto, se realiza una reunión con el *Product Owner* para evaluar el resultado *Sprint review* y se planifica la siguiente iteración (*Sprint planning*).

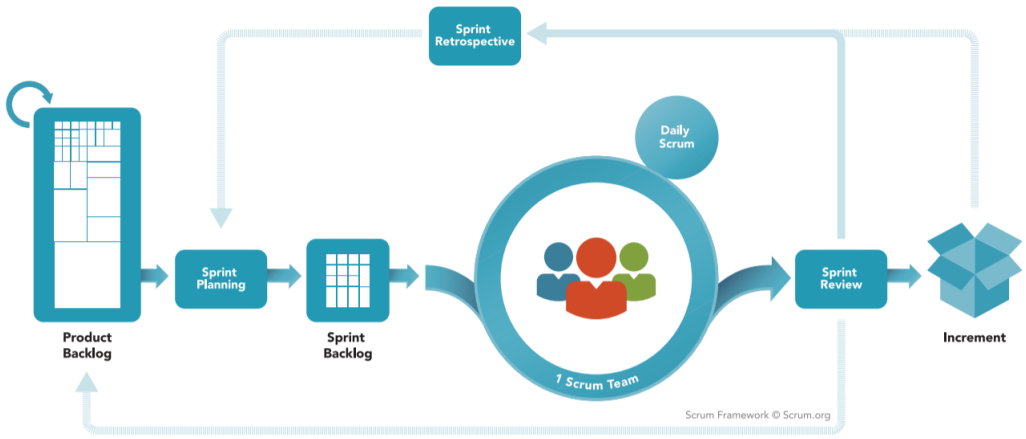

Figura 2.1: Metodología Scrum. Fuente: scrum.org/resources/scaling-scrum

Dada la naturaleza del proyecto y la poca experiencia previa en el desarrollo de chatbots esta forma de trabajar aporta la suficiente flexibilidad para subsanar los posibles errores o problemas que surjan durante el proceso de diseño o desarrollo. Por otra lado, una parte crucial en la construcción de cualquier chatbot es poder empezar a probar el servicio con usuarios reales cuando antes, de manera que se pueda ir incorporando a la base de datos toda la información recopilada y el desarrollo se pueda beneficiar de este *feedback*. Es en este punto donde la entrega de valor en períodos cortos de tiempo de *Scrum* es idónea para el proyecto, ya que estas iteraciones posibilitan tener una producto mínimo viable (*PMI*) cuanto antes.

En este proyecto los *Sprints* han sido de 2 semanas naturales de duración, realizando el día 14 una reunión con el *Product Owner* (los tutores del proyecto) para valorar el progreso realizado y determi-

nar las próximos pasos a seguir.

Por otra parte, también se ha mantenido un contacto constante a través de reuniones *online* y correos electrónicos con los investigadores del Hospital General de Elche para contar en todo momento con su visto bueno en temas médicos y de diseño.

En los documentos anexos a este trabajo se pueden encontrar las actas de todas estas reuniones con la fecha, los participantes y los temas tratados.

2.2. Control de versiones

Durante el desarrollo se ha utilizado *Git* como herramienta de control versiones, manteniendo así un control sobre los cambios realizados en el código. Se trabajó con dos repositorios, uno para el *frontend* [4] y otro para el *backend* [5] (ver sección 4.2.1), ambos con 2 ramas distintas: *master* y *develop*. Durante el trascurso del *sprint* se trabajaba sobre *develop* y en una máquina local. Al finalizar, si todo funcionaba correctamente, se realizaba un *merge* a *master* para incorporar a la rama estable los últimos cambios.

Esta metodología de trabajo era crucial, ya que, durante una gran parte del desarrollo se tuvo a disposición un servidor de producción donde funcionaba el chatbot (ver sección 4.3.4). Este servidor estaba siempre actualizado con la última versión de la rama *master*, y por tanto, era muy importante que fuera lo más estable posible.

2.3. Roadmap

Siguiendo la metodología *Scrum* se propone una ruta de desarrollo (ver Figura 2.2) centrada en la obtención de un producto mínimo viable lo antes posible. Para ello primeramente se realizan los diseños del chatbot, se organiza la información recibida y se trabaja para la obtención de un *PMI* en versión web. A partir de ese punto es se publica una versión de prueba del *bot* para empezar con la obtención de datos reales y realizar distintas iteraciones sobre el producto hasta conseguir el resultado deseado. Una vez se alcanza el nivel de madurez requerido, la web se empaqueta como *app* para móvil y se publica.

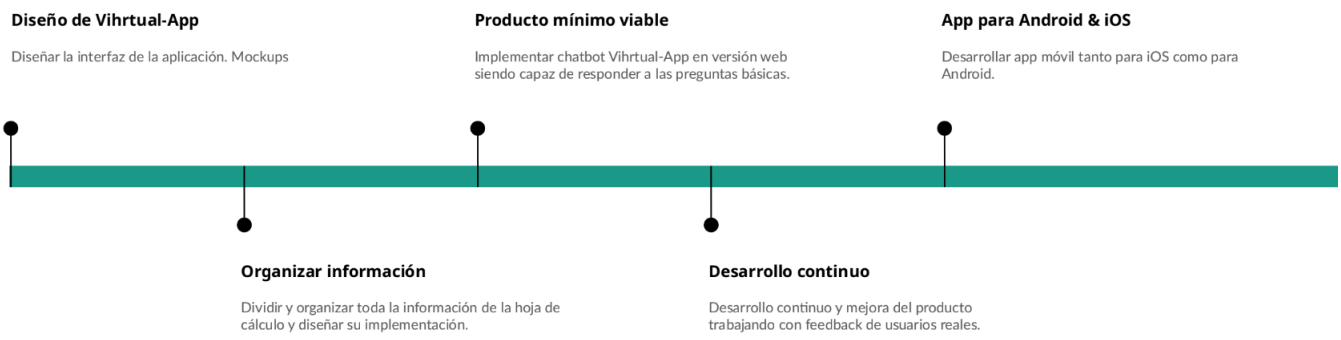

Figura 2.2: *Roadmap* propuesto para el desarrollo de VIHrtual-App

Capítulo 3

Marco Teórico

En este capítulo se realiza una breve descripción del estado actual de la interacción hombre máquina, la interpretación del lenguaje natural y diferentes técnicas para aplicar estos principios a la creación de chatbots. Por otra parte, también se exponen distintas características sociales y de interacción a tener en cuenta en el diseño de un chatbot. Por último, se revisan varios casos de asistentes conversacionales orientados al campo de la salud y se exponen distintas herramientas disponibles para realizar una implementación.

3.1. Interacción hombre-máquina y lenguaje natural

El lenguaje es la capacidad que tiene el ser humano para expresarse y comunicarse a través de diversos sistemas de signos orales, escritos o gestuales. Con la aparición de nuevas tecnologías y la creciente capacidad de computación se han abierto nuevos campos de investigación de técnicas para el procesamiento de este lenguaje natural (*NLP: Natural language processing*) con el propósito de utilizarlo como medio de interacción entre el hombre y máquina. En los últimos años estos esfuerzos se han materializado en el desarrollo de varios sistemas capaces de entender, hasta cierto punto, las intenciones expresadas por cualquier persona mediante su propia lengua para luego actuar en consecuencia. Si bien estos sistemas utilizan los últimos avances ofrecidos por la inteligencia artificial la realidad es que se encuentran aún en una etapa muy temprana de desarrollo. Por tanto, constituyen un punto de partida y sientan las bases del futuro de la interacción entre usuarios y dispositivos, estableciendo un escenario totalmente nuevo donde ya no será necesario utilizar ratón, teclado, pulsaciones o cualquier otro método tradicional de entrada. De hecho, las clásicas interfaces ya están dejando paso a un nuevo paradigma de interacción mucho más intuitivo dónde no es necesario hacer clicks ni teclear para realizar ciertas operaciones [6, 7].

Por ejemplo, el asistente virtual *Alexa* permite reproducir música con sólo decir *Alexa, reproduce la lista relax de Spotify.* O *Siri*, que puede encontrar la gasolinera más próxima a tu ubicación con un *Oye Sire, indícame como ir a la gasolinera más cercana.* Aunque esto pueda parecer simple, ciertos procesos más complejos que antes podían resultar algo tediosos como reservar un vuelo, ahora ya son posible con un simple *Google, resérvame un vuelo a París*, sin necesidad de navegar a través distintas webs ni rellenar formularios.

3.2. Chatbots

Un chatbot (*chatter-bot*) o asistente virtual se puede definir como una interfaz (programa) diseñado para que los usuarios puedan interactuar con sus dispositivos haciendo uso de lenguaje natural, ya sea por voz o texto. El objetivo principal es siempre transmitir la sensación al usuario que se encuentra dialogando con una persona real y mantener esta ilusión el mayor tiempo posible. Para ello es crucial que los chatbots sean capaces de demostrar entendimiento, respondan con coherencia y resuelvan las

tareas / problemas que los usuarios les planteen.

3.2.1. Clasificación

En los últimos años, con la popularización de los chatbots y la democratización de sus tecnologías se han creado multitud de *bots* con propósitos muy distintos. Esto dificulta realizar una clasificación precisa dada la alta subjetividad respecto al ámbito al cual puede pertenecer un chabot. Sin embargo, es posible realizar una clasificación general en función de los siguientes criterios: Objetivo, modo de interacción, técnica de diseño y ámbito [8, 9].

Objetivo (*Goal*)

Según el objetivo para el cual han sido diseñados los chatbots pueden clasificarse en:

- *Task oriented*: Su objetivo es ayudar a los usuarios a realizar una tarea muy concreta [8]. En consecuencia, están diseñados para desenvolverse en contextos bastante específicos como reservar un vuelo o anotar un evento en la agenda. En la mayoría de los casos estas acciones han sido establecidas previamente por lo que el flujo de la conversación y todos los posibles caminos están ya previstos [9].
- *Non-task oriented / Conversacionales*: No tienen ningún propósito en particular y su función se limita a mantener una conversación con el usuario. Por ende, su propósito es asemejarse lo máximo a una conversación real respondiendo y siguiendo adecuadamente al interlocutor.
- *Informativos*: Son *bots* cuya única tarea es ofrecer al usuario la información que requiera. Normalmente esta información es extraída mediante algoritmos de *string matching* de una base de datos u otra fuente de datos estática. Un ejemplo claro de aplicación son los asistentes que resuelven las dudas más frecuentes (*FAQS*) [9].

Modo de interacción

Dependiendo de como el usuario interactúe con el asistente se puede clasificar el chatbot como basado en texto, basado en voz o en respuestas predefinidas. Los siguientes modos no son incompatibles y muchos *bots* combinan ambos.

- *Voz*: Aquellos chatbots cuyo método de entrada es la voz. El asistente registra el conjunto de sonidos emitido por el usuario, lo traduce a texto e interpreta su significado. Este es el caso de los famosos asistentes del hogar como *Google Home* o *Alexa* dónde por la naturaleza de la aplicación el soporte de voz resulta ser lo más conveniente.
- *Texto*: Los *bots* basados en texto aceptan la escritura libre de cualquier mensaje y suelen encontrarse en dispositivos donde escribir resulta cómodo como ordenadores y móviles.
- *Respuestas predefinidas*: Dentro de los chatbots basados en texto se encuentran aquellos con los que se interactúa mediante el uso de respuestas predefinidas, normalmente haciendo uso de botones de respuesta rápida. Estos asistentes tienen el flujo de conversación estrictamente definido y por tanto el usuario sólo puede recorrer un número limitado de caminos previamente definidos.

Técnica de diseño

Para que un chatbot responda de forma adecuada es crucial reconocer que intenciones tiene el usuario cunado se expresa. Para ello, anteriormente se utilizaban algoritmos simples basados en reglas donde la interacción se limitaba a un simple patrón de pregunta-respuesta. Sin embargo, hoy en día con la aparición de nuevas técnicas es posible crear sistemas más complejos que nos permitan implementar

patrones de conversación mucho más sofisticados, ofreciendo al usuario final una experiencia mucho más natural y humana. Actualmente existen múltiples técnicas de diseño para el desarrollo de chatbots:

- *Parsing*: Es un método que toma como valores de entrada el texto proporcionado por el usuario y extrae la información más significativa [8]. Para ello, transforma el texto en un conjunto más sencillo de guardar y manipular. Actualmente, existen técnicas más avanzadas de *parsing* que convierten el texto en una representación numérica entendible por los ordenadores.

- *Pattern matching*: Es la técnica más empleada en el desarrollo de la mayoría de chatbots, sobretodo en los chatbots destinados a responder preguntas (*QA chatbots*) [8, 6]. Este método toma el *input* generado por el usuario e intenta clasificarlo como uno de los patrones que ya tiene definidos. Una vez clasificado como *pattern*, devuelve la respuesta asociada (*template*). Estas relaciones *pattern-response* se construyen a mano y constituyen uno de los principales inconveientes de estos sistemas [8]. Aunque las técnicas de *pattern matching* han estado presentes desde los primeros chatbots, han ido evolucionando hasta incluir cierta noción del contexto conversacional en sus algoritmos [8].

- *AIML*: AIML (*Artificial Intelligence Mark-up Language*) es un lenguaje de marcas derivado de XML diseñado para representar el conocimiento que se le da a los chatbots [6]. AIML contiene los denominados *AIML objects*, los cuales estan compuestos por *topics* o *categories*. Estos últimos son la unidad de información más importante de AIML. Están compuestos por dos elementos: *pattern* y *template*. El *pattern* representa el *input* del usuario y *template* la respuesta asociada a esa entrada. El propósito de estos objetos es modelar el flujo conversacional.

- *Chatscript*: Es una herramienta *opensource* que combina un motor de comprensión del lenguaje natural y un sistema de control de diálogo. Esta diseñada para construir chatbots que puedan mantener conversaciones interactivas guardando el estado del usuario entre conversaciones [8]. Para ello, utiliza un sistema de ficheros con reglas, las cuales se utilizan para definir el flujo del diálogo.

- *Ontologías*: Las ontologías se utilizan en los chatbots para sustituir el conocimiento elaborado a mano, por el conocimiento ontológico [8]. La mayor ventaja de esta técnica es la capacidad de los chatbots de establecer relaciones entre los conceptos de un dominio o área de conocimiento, pudiendo llegar a crear nuevos razonamientos [8].

- *Markov Chain Model*: *Markov Chain Model* es un modelo de probabilidad diseñado para predecir el siguiente estado en base al estado actual [8]. Esto implica que, cada vez que el chatbot responde, construye una nueva frase en base al modelo probabilístico. Este sistema es una simplificación de un proceso más complejo de toma de decisiones, y por tanto, no funciona bien para emular conversaciones complejas. Sin embargo, es bastante popular para chatbots orientados al entretenimiento, donde basándose en el *input* del usuario pueden imitar conversaciones simples [8].

- *Redes neuronales*: Hoy en día, con los últimos avances en *machine learning*, y en especial, con las redes neuronales, ha sido posible desarrollar chatbots más inteligentes [8]. La principal diferencia con los algoritmos basados en reglas es la capacidad de estos sistemas para aprender y mejorar. De hecho, en el campo del *deep learning* se han desarollado sistemas neuronales capaces de aprender sin de datos sin etiquetar y sin supervisión. Esto ha provocado un creciente desarrollo de redes neuronales aplicadas al procesamiento del lenguaje natural, especialmente la red neuronal recurrente (RNN) y las redes de memoria a largo plazo (LSTM) [8].

Ámbito

En función del conocimiento que posea el chatbot y los campos que intente abarcar se pueden definir como de ámbito abierto o cerrado.

- *Ámbito abierto*: Asistentes que no tienen una área de conocimiento en específico, tienen un carácter más general y suelen estar hechos para el entretenimiento.
- *Ámbito cerrado*: Centrados en resolver dudas o realizar tareas dentro de un ámbito muy específico. Suelen fallar al responder sobre temas más generales.

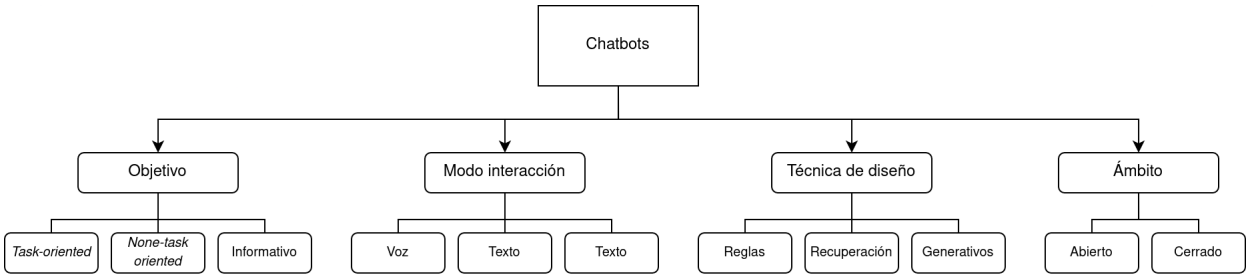

Figura 3.1: Clasificación de chatbots

3.2.2. Interacción y características sociales

Como ya se ha mencionado, es de vital importancia que las conversaciones entre el *bot* y el usuario sean lo más semejantes posibles a una entre humanos. Para ello no deben de transmitir una sensación robótica, si no que deben ser fluidas y lo más naturales posibles. Para ello, un chatbot debe de integrar una serie de rasgos que garanticen esta experiencia.

Estas características sociales han sido extraídas del artículo *How should my chatbot interact? A survey on social characteristics in human-chatbot interaction design* [7] de Ana Paula Chaves y Marco Aurelio Gerosa. En él, se hace un repaso a la literatura actual sobre cómo debería actuar un chatbot, que rasgos de personalidad debería tener y diferentes comportamientos observados. A continuación se describen aquellos aspectos que se han considerado más relevantes.

Proactividad

La proactividad es la capacidad de un sistema de poder actuar autónomamente en nombre del usuario. Un comportamiento proactivo aporta iniciativa a una conversación, contribuyendo a sostener una conversación más natural. En un ámbito más práctico, los chatbots se pueden mostrar proactividad cuando sugieren nuevos temas, realizan preguntas y ofrecen información adicional.

Conciencia

La conciencia es la capacidad de un chatbot para demostrar atención y entendimiento en una conversación. Para ello es crucial que sepa interpretar el contexto, evaluando cada frase como parte de un conjunto. Cuando un chatbot no interpreta correctamente el significado o la intención de un mensaje, su credibilidad y humanidad se ve comprometida.

Comunicabilidad

La comunicabilidad, en el contexto de los chatbots, es la capacidad de transmitir al usuario qué funcionalidades tiene. El uso de elementos visuales como respuestas rápidas e imágenes pueden ayudar a mostrar al usuario qué tareas puede realizar el chatbot, y por tanto, reducir el número de errores de entendimiento. De hecho, un estudio realizado sobre chatbots relacionados con las noticias confirmó que el 63 % de las conversaciones con los chatbots se iniciaban haciendo click sobre un elemento mostrado en el mensaje inicial.

Por otra parte, el chatbot debe aclarar el objetivo del chatbot a través de un mensaje inicial y estar preparado para responder a preguntas como *¿Qué puedes hacer?* o *¿En qué me puedes ayudar?*.

Controlar daños

Todo asistente conversacional, por muy bien diseñado que esté, se enfrentará tarde o temprano a una situación no contemplada previamente. El control de daños (damage control) son las diferentes técnicas que se pueden implementar para tratar de recuperarse en situaciones conflictivas o donde el chatbot falle. Por ejemplo, es común que un chatbot erre en una conversación debido a la falta de conocimiento lingüístico, o por enfrentarse a una pregunta que se encuentra fuera de su ámbito de conocimiento. Por tanto, un asistente debe ser capaz de reconocer estas situaciones, responder admitiendo su error e intentar reconducir la conversación.

Además, también debe detectar conductas abusivas y responder adecuadamente ante insultos y vejaciones, intentando evitar que el usuario continue con su conducta.

Rigurosidad

La sigurosisdad es la habilidad de un chatbot de ser preciso en la forma que utiliza el lenguaje. El lenguaje utilizado por un asistente debe ser consistente y no combinar varios estilos. Por ejemplo, muchos usuarios pueden ver extraño el uso combinado de emojis con un estilo muy formal.

3.3. Chatbots en el campo de la salud

Muchas de las tecnologías mencionadas anteriormente se han empezado a introducir en el campo de la medicina y de la salud en busca de soluciones que mejoren la vida de los pacientes. Aprovechando los últimos avances en interacción entre personas y máquinas se están desarrollando propuestas de asistentes virtuales que ayuden en el día a día a pacientes, familiares o incluso personas mayores. La idea es siempre mejorar la atención y automatizar ciertos procesos médicos [2] bastante comunes y rutinarios como puedan ser responder a ciertas consultas, realizar el seguimiento de una medicación o incluso detectar indicios de enfermedad.

Dependiendo de la intención y problema que pretendan solucionar en el campo se la salud se diferencian 4 grandes categorías: educación, *coaching*, prevención y diagnóstico.

Educación

Los chatbots educativos son aquellos orientados a ayudar a los usuarios a entender conceptos médicos y resolver todas aquellas dudas que puedan tener. Esta tarea la pueden realizar proporcionando información relevante sobre cierta enfermedad, resolviendo dudas técnicas sobre un análisis o ayudando a pacientes y familiares a comprender diagnósticos y tratamientos.

Por ejemplo, un asistente virtual en este contexto podría resultar útil para concienciar y advertir a la población sobre los riesgos de una determinada enfermedad. Este es el caso de IRA [10], un chatbot creado en India que, al igual que VIHrtual-App tiene por objetivo ayudar con en el grave problema padece el país con el VIH ofreciendo información y motivando a la gente a ser conscientes y responsables (ver Figura 3.2).

Dado IRA comparte objetivos con VIHrtual-App, merece especial consideración hacer una breve descripción de sus principales características. Primeramente, y diferencia de VIHrtual-App, IRA no está diseñada para interactuar utilizando el idioma propio del país, si no que ha sido programado en

inglés. Por tanto, su público o alcance es mucho mayor, pues cualquier usuario del mundo puede acceder e informarse. Por contra, presenta poca tolerancia a los errores gramaticales y no detecta muchas expresiones informales. También se ha observado que no es capaz de reconocer distintas variantes de una misma pregunta. Por ejemplo, en las pruebas realizadas, IRA identificó correctamente la expresión *Risk of anal sex*, pero falló en reconocer otras expresiones como *Is anal sex dangerous?* o *Why is anal sex dangerous?*.

En este sentido, en el diseño propuesto para VIHrtual-App se incluye como requisito que el asistente robusto, sea capaz de detectar diferentes variaciones de una misma pregunta y entienda tanto un registro formal como informal.

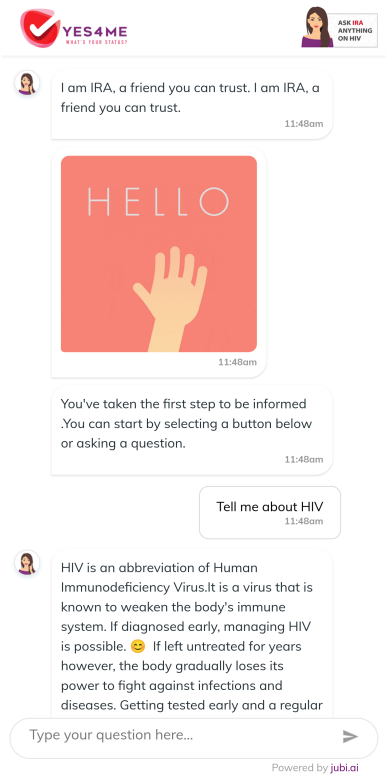

Figura 3.2: Captura de pantalla de IRA

Coaching

La segunda categoría hace referencia al término *coach* (entrenador personal). Son chatbots similares a los anteriores pues están orientados también a formar al usuario pero, si en el anterior grupo la posición del asistente era bastante pasiva, aquí es todo lo contrario. Este tipo de asistentes toman una actitud de liderazgo para influir y mejorar directamente en la salud del usuario.

De esta manera, un chatbot podría enviar notificaciones al móvil de un paciente para recordarle cuándo debe tomar la medicación y así trabajar para introducir nuevos hábitos y comportamientos. También podría aconsejar mejores hábitos de vida y proponer objetivos diarios al usuario, trabajando codo con codo para mejorar su salud.

Un buen ejemplo de esta aplicación es Florence [11] (ver Figura 3.3), un asistente que pretende ser un enfermero personal que recuerda a los usuarios cuándo deben tomarse la medicación y lleva un seguimiento del peso, presión arterial y periodo.

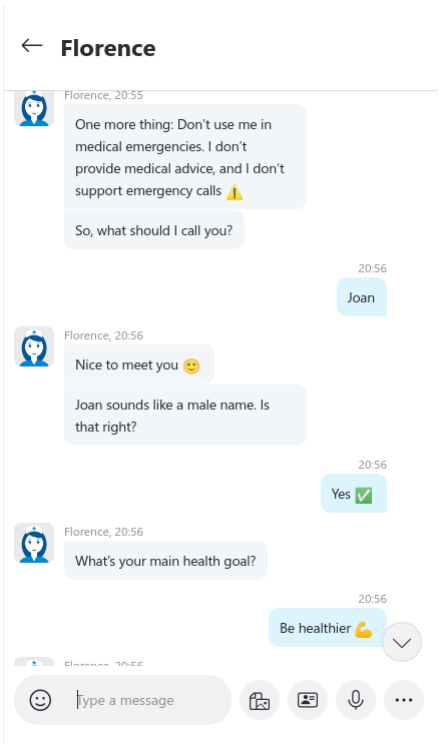

Figura 3.3: Captura de pantalla de Florence

Prevención

En la categoría de prevención se engloban todos aquellos chatbots orientados a mantener una relación continua entre médico y paciente. A través de conversaciones rutinarias un asistente virtual puede detectar patrones habituales ya conocidos y anticiparse a cualquier problema que pueda derivar en una posible enfermedad. Por ejemplo, mediante técnicas más avanzadas es posible detectar principios de demencia [12] o actitudes depresivas para la prevención de suicidios.

Diagnóstico

Por último, el cuarto grupo son aquellos chatbots que en base a unas indicaciones sintomatológicas es capaz de sugerir un diagnóstico. Esta clase de *bots* suelen utilizarse un ambiente clínico como una herramienta más para los médicos y que, por ejemplo, haciendo uso de grandes bases de datos y estadísticas sugieran al médico posibles diagnósticos.

Por otra parte también existen aplicaciones en este sentido donde se usa directamente por usuarios fuera del ámbito médico, y por tanto, suelen ser diagnósticos más sencillos. En esta dirección trabaja *Wakamola* [13] (ver Figura 3.4), un chatbot que, a través de una serie de preguntas recoge información sobre la dieta, actividad física, edad, peso etc. En función de las respuestas del usuario se realiza un cálculo y se muestra una puntuación (*Wakaestado*) entre 0 y 100.

3.4. Herramientas para construir chatbots

A la hora de desarrollar un chatbot existen diferentes herramientas que facilitan el uso e implementación de varias de las técnicas descritas en la sección 3.2.1. A continuación se realiza un análisis y comparativa de las alternativas más relevantes para construir chatbots.

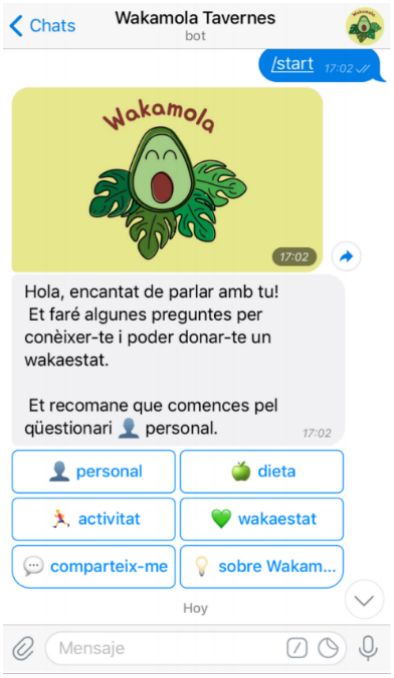

Figura 3.4: Captura de pantalla de Wakamola

3.4.1. Rasa

Rasa [14] es un *framework* gratuito y *opensource* que permite construir sistemas conversacionales inteligentes. Esta herramienta proporciona tres grandes funcionalidades para la construcción de un chatbot: comprensión del lenguaje natural (*NLU*), control del flujo los diálogos mediante algoritmos de *machine learning* e integración con distintos canales (*API REST*, aplicaciones de mensajería instantánea etc).

Rasa hace una fuerte apuesta por el desarrollo basado en conversaciones reales (*CDD: Conversation driven development*) y para ello ofrece un producto gratuito llamado *Rasa X*. Aunque no es *opensource* sí es gratuito y de libre uso facilitando la recogida y posterior etiquetado de las conversaciones.

3.4.2. Amazon Lex

Amazon Lex [15] es un servicio para crear agentes conversacionales que interactúen mediante voz o texto. Esta tecnología es la misma que se encuentra detrás de *Alexa*, por tanto este servicio pone a disposición de los desarrolladores el propio motor del popular asistente de Amazon.

Las principales características que ofrece son comprensión del lenguaje natural (NLU), reconocimiento automático de voz (ASR) e integración con distintos canales (aplicaciones de mensajería y web). El uso de la voz y la potencia de motor permite construir experiencias muy atractivas pero por contra, el servicio hace uso de los sistemas en la nube y cobra por cada solicitud de texto o voz que se realice (3.25€ por cada 1000 peticiones de voz y 0.60€ por cada 1000 de texto).

3.4.3. Microsoft Bot Framework

Bot Framework es el producto ofrecido por *Microsoft* para construir *bots* conversacionales. Este servicio ofrece un conjunto de herramientas que interactúan su propia plataforma de computación en la nube *Azure*. Por tanto, se trata de un servicio de pago que requiere de conexión para funcionar.

Sin embargo, cuenta con un plan gratuito y sin límite de mensajes, siendo la única limitación los canales por los cuales se puede acceder al chatbot. Los canales disponibles gratuitamente son *Skype*, *Slack*, *Facebook*, *Cortana* y *MS Teams*. Si se desea conectarse al chatbot a través de otros canales, como por ejemplo una web, es necesario recurrir al pago del servicio (0.42€ por cada 1000 mensajes).

3.4.4. Botkit

Botkit [16] es una herramienta *opensource* para crear chatbots que puedan integrarse en aplicaciones web o en las principales plataformas de mensajería instantánea. Actualmente forma parte del conjunto de herramientas de *Microsoft Bot Framework* por lo que ofrece soporte para hacer uso de *LUIS.ai* [17].

Por otra parte, *Botkit* también ofrece una herramienta adicional denominada *Botkit CMS*. Con ella, es posible diseñar, construir y gestionar diálogos de manera gráfica, permitiéndonos así crear interacciones más complejas donde el *bot* pueda realizar preguntas o realizar distintas acciones. Junto con *Rasa*, ambas son las dos grandes alternativas de código abierto para el desarrollo de chatbots. En comparación, *Botkit* tiene un desarrollo bastante menos activo, siendo su última versión de hace aproximadamente un año. En cambio, *Rasa* lanza versiones menores cada pocos meses y posee una comunidad más activa.

3.4.5. Comparativa

En la tabla 3.1 se puede observar una comparativa con las principales características de los *frameworks* descritos anteriormente. Tras valorar las distintas alternativas para construir chatbots se escoge a *Rasa* como plataforma sobre la cual se realizará el desarrollo. Las razones detrás de esta elección son las siguientes:

- *Rasa* cuenta con todas las herramientas necesarias para la correcta interpretación de las intenciones del usuario mediante técnicas de *NLU* y noción del contexto.
- Es *opensource*, gratuito y a diferencia de otros competidores, no se basa en ningún servicio en la nube propietario. Esto permite realizar el despliegue completo de la plataforma en cualquier máquina.
- En consecuencia, se obtiene independencia de otros servicios y control total sobre los datos recopilados de conversaciones, pues estas no estarán en manos de terceros.
- La plataforma ofrece una herramienta adicional para el aprendizaje supervisado llamada *Rasa X*. Esta permite realizar fácilmente la revisión y etiquetado de las conversaciones por lo que resulta idónea para el desarrollo.

	Microsoft	Botkit	Amazon Lex	Rasa
Basado en la nube	Sí	No	Sí	No
NLU	Sí	Sí	Sí	Sí
Soporte mensajería	Sí	Sí	Sí	Sí
Coste	Pago	Gratuito	Pago	Gratuito

Tabla 3.1: Comparativa de *frameworks*

Capítulo 4

Desarrollo de la Propuesta

En el siguiente capítulo se desarrolla la propuesta de este trabajo de final de grado. Primeramente, se realiza un análisis de requisitos y se expone el diseño de la propuesta. Una vez la propuesta esta definida, se detalla todo el proceso de implementación seguido.

4.1. Análisis de requisitos

Como requisito fundamental por parte de los colaboradores de la Unidad de Enfermedades Infecciosas del Hospital General de Elche se propuso que el chatbot fuera capaz de responder a un mínimo de preguntas que bajo su criterio médico consideraron básicas para que el servicio contara con una base suficiente de conocimiento. Este listado fue entregado en una hoja de cálculo que se puede encontrar en los documentos anexos a este trabajo. El fichero consta de 131 preguntas con sus correspondientes respuestas y cubre las preguntas más frecuentes que suelen realizarse sobre el virus de la inmunodeficiencia humana (VIH).

Aunque inicialmente se partirá de esta base, se requiere que el sistema sea fácilmente escalable, de manera que a lo largo de un futuro desarrollo se pueda ir incorporando nueva información sin que suponga un problema. Por otra parte, una vez el *bot* esté accesible al público y los usuarios interactúen con él se desea poder recopilar toda la información para posteriormente revisarla e incorporarla al *bot*.

Junto a todos estos datos será también necesario incorporar todas aquellas expresiones auxiliares que suelen acompañar las conversaciones: afirmaciones, negaciones, saludos, despedidas, agradecimientos etc. Además, habrá que añadir un subconjunto de preguntas / respuestas básicas sobre el chatbot y otras cuestiones básicas que serán necesarias para dotar al asistente de una personalidad.

En conjunto se pretende que el chatbot se parezca lo máximo posible a una persona real, que demuestre estar versado en el tema sobre el que informa, que preste atención e interés en la conversación y que sea capaz de seguir la conversación. Todo esto es crucial a la hora de mantener una conversación con sentido y fluida que, en la medida de lo posible no haga sentir al usuario que está hablando con un robot. Desde el punto de vista técnico esto requerirá incorporar técnicas de comprensión del lenguaje natural (NLU) y métodos de toma de decisiones en función del contexto. Por ende, es requisito fundamental que el *bot* sea capaz de comprender las intenciones del usuario y responder en consecuencia.

Adicionalmente, y como parte de la aplicación se desea que exista un apartado donde los usuarios puedan consultar de manera más extensa información acerca del VIH. La presentación de esta información debe seguir un formato clásico y una estructura jerarquizada.

En lo referente a su distribución / publicación, al ser un servicio informativo y de concienciación se debe buscar la mejor solución para que sea fácilmente accesible. Por último, y no menos importante se debe de considerar también la problemática de la recogida de datos pues en una conversación especialmente sobre VIH se pueden introducir datos especialmente sensibles. Los usuarios deberán ser

conscientes de como se van a tratar sus datos y aceptar los términos y condiciones.

Este análisis se presenta y acuerda en una reunión con los colaboradores la Unidad de Enfermedades Infecciosas del Hospital General de Elche para obtener su aprobación. Tanto las notas de la reunión como las diapositivas se pueden encontrar en los documentos anexos a este trabajo.

4.2. Diseño

Después de analizar detenidamente los requisitos se procede a detallar la solución propuesta para el desarrollo de VIHrtual-App.

4.2.1. Arquitectura de la aplicación

Debido a los altos requerimientos computacionales y de espacio, *Rasa* se basa en una arquitectura tradicional cliente-servidor. Para ello, pone a disposición de los desarrolladores un servidor ya integrado dentro del propio *framework* con el cual es posible publicar el chatbot. Este *backend* expone una *API* conversacional que puede ser consumida a través de diferentes canales, como pueda ser una web o una plataforma de mensajería instantánea.

Teniendo en cuenta el carácter del servicio, se decide que el chatbot se ponga a disposición del público a través de una web de libre acceso y sin requerir registro. El objetivo es facilitar a los usuarios el acceso para que cualquier persona pueda acceder al servicio sin necesidad de introducir ningún dato personal ni instalar ninguna aplicación en su terminal. Por tanto, la web será el cliente (frontend) que ofrecerá la interfaz conversacional y establecerá comunicación con el servidor (*backend*) de *Rasa*.

Adicionalmente, y para algunas funciones más avanzadas, es necesario el uso de un segundo servidor denominado *Rasa Action Server*. Este servicio permite la ejecución de funciones *Python* y se utiliza para realizar tratamientos adicionales a los datos. En este caso, este servicio ha permitido implementar el mensajes de bienvenida que muestra el chatbot al inicio.

Tal y como se puede observar en la Figura 4.1, estos tres elementos (web, servidor conversacional *Rasa* y *Rasa Action Server*) conforman la arquitectura completa de la aplicación. La web establecerá comunicación mediante el protocolo *HTTP* con los servidores de *Rasa* y podrá ser publicada a través de un servidor web, o empaquetada en forma de aplicación móvil.

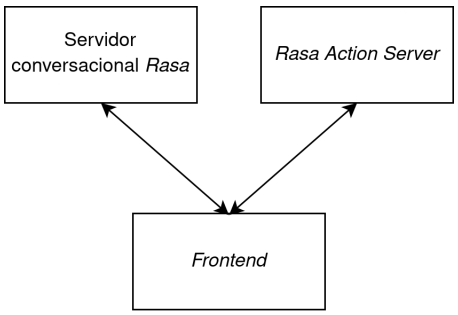

Figura 4.1: Esquema de la arquitectura de Vihrtual-App

4.2.2. Mockups

Tras el análisis de requisitos se empieza a trabajar en el diseño de la aplicación. Teniendo en cuenta que la web será accesible tanto por dispositivos móviles como por ordenadores se realizan dos diseños distintos: una versión móvil y otra de escritorio. Ambas versiones siguen los mismos principios pero

tratan de adaptarse al tamaño de pantalla y hacer un uso adecuado del espacio.

Como fuente veraz de información el servicio debe transmitir fiabilidad y confianza, por esta razón se opta por un estilo sobrio y sencillo donde diferentes tonos de azul predominan [18]. Con ello se espera que transmita una sensación de paz y seguridad al usuario.

La aplicación se compone de tres pantallas distintas: selección del asistente, chat y consulta de información. En la Figura 4.2 podemos observar estas vistas junto con el diagrama de navegación de la aplicación.

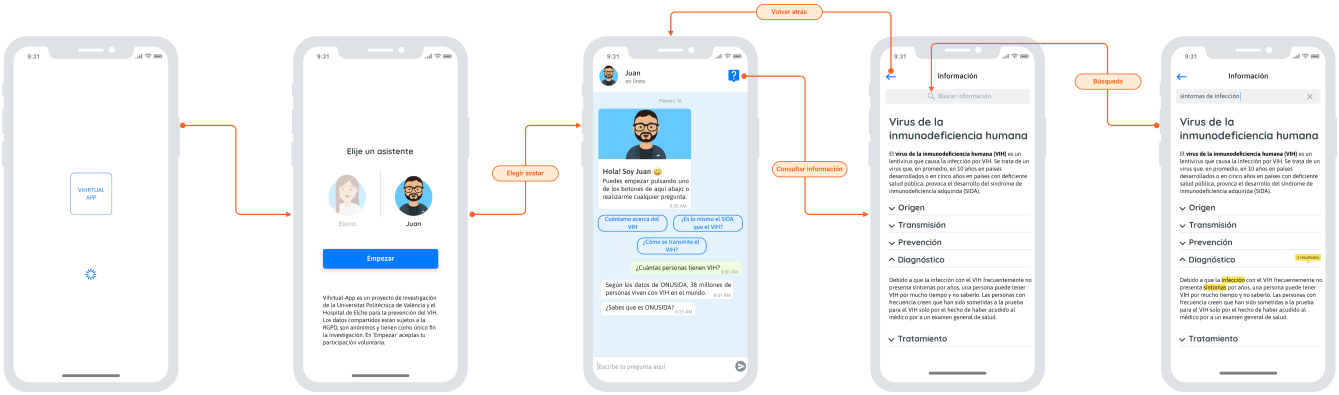

Figura 4.2: Diagrama de navegación de la aplicación. Versión móvil

Selección del asistente

En busca de establecer un primer vínculo entre el chatbot y el usuario se dota al sistema de dos personalidades distintas para humanizar la experiencia. En la primera pantalla a través de dos avatares (Juan y Elena) (ver Figura 4.3 y 4.4) el usuario puede eligir con cual de estas dos identidades desea mantener una conversación y así empezar a crear la relación. Además de elegir con quién se siente más cómodo hablando, también lee el aviso sobre el tratamiento de sus datos. Tras ello, puede pulsar "*Empezar*"para aceptar los términos y proceder al chat.

Chat

Desde un principio se tiene claro que el diseño debe girar alrededor de la interacción con el *bot*, y por tanto, el chat debe ser el elemento principal de la aplicación. Por ello, en la versión móvil se opta por una disposición donde prácticamente todo el espacio es ocupado por la conversación (ver Figura 4.5). En cambio, en la versión de escritorio el chat ocupa la mitad derecha de la pantalla, mostrando en la parte izquierda la imagen del avatar escogido con un pequeño texto informativo (ver Figura 4.6). El estilo del chat sigue los patrones de diseño típicos y más o menos estandarizados de aplicaciones de mensajería instantánea como *WhatsApp* o *Telegram*. El motivo detrás de esta decisión es facilitar la usabilidad, ya que prácticamente la totalidad de los usuarios están familiarizados con este tipo de interfaces.

Tal y como se puede observar en las Figuras 4.5 y 4.6, en la cabecera superior encontramos los elementos habituales: la imagen de perfil de con quién se está hablando y un pequeño texto que nos indica el estado: *en línea* o *escribiendo*. Ambos elementos juegan una papel importante en la simulación: mientras la conversación transcurre el usuario ve constantemente el avatar del asistente e inconscientemente realiza una asociación entre los mensajes y la imagen de la persona que esta viendo. Por otra parte, aunque el usuario sepa que es una simulación observar como el chatbot va cambiando su estado según esté escribiendo o no, vuelve a reforzar la idea debido al paralelismo con las aplicaciones

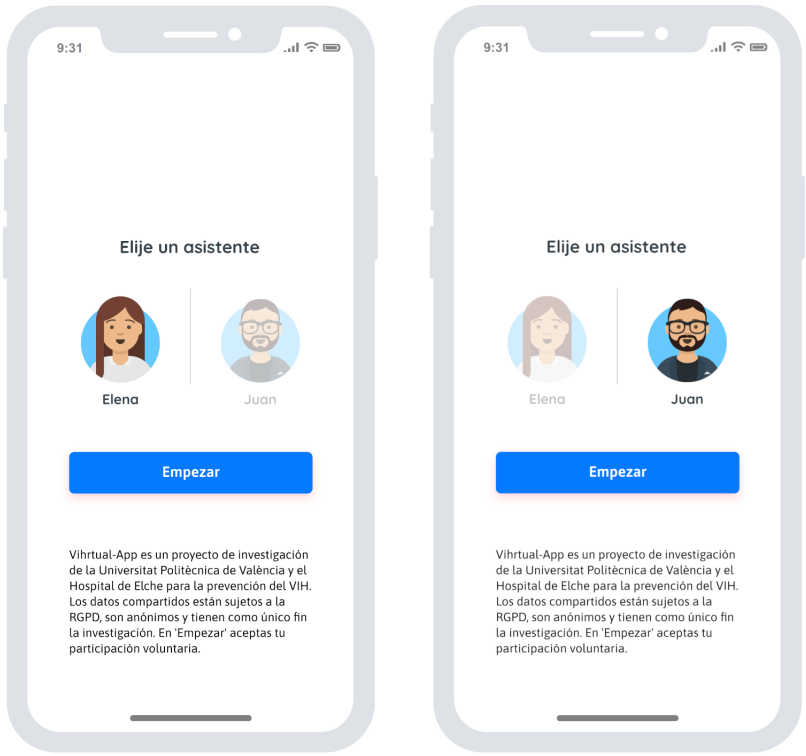

Figura 4.3: Selección del asistente. Versión móvil

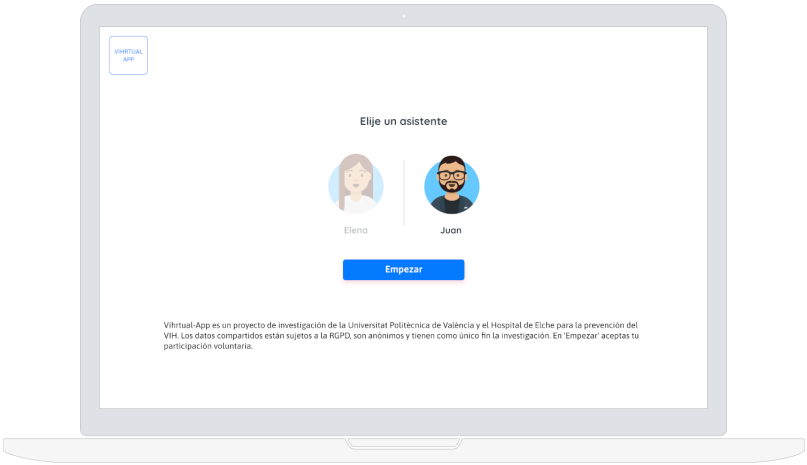

Figura 4.4: Selección del asistente. Versión de escritorio

anteriormente mencionadas.

Por último, a través del icono de la esquina superior derecha se accede a consultar la información adicional.

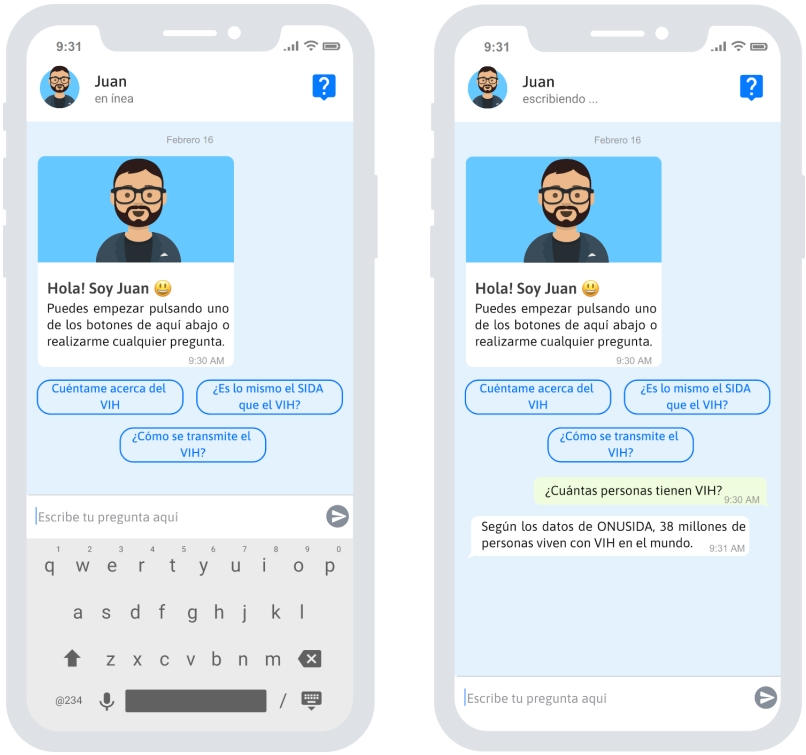

Figura 4.5: Chat. Versión móvil

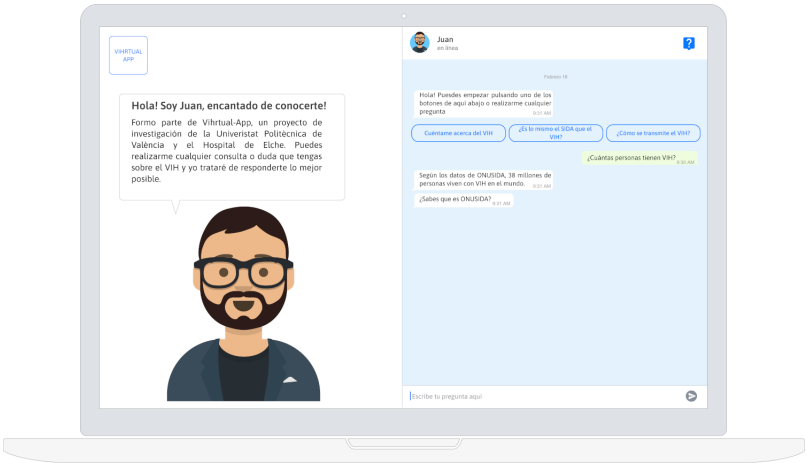

Figura 4.6: Chat. Versión de escritorio

Consulta de información

La pantalla de información contiene una breve introducción al VIH y una serie de temas anidados que se pueden ir desplegando y consultando. Además, en la parte superior existe una barra de búsqueda que permite encontrar rápidamente ciertos términos o palabras clave. Si existen resultados que coincidan con la búsqueda se indica con un pequeño indicador amarillo sobre el desplegable y resaltando el término en cuestión (ver Figura 4.7 y 4.8).

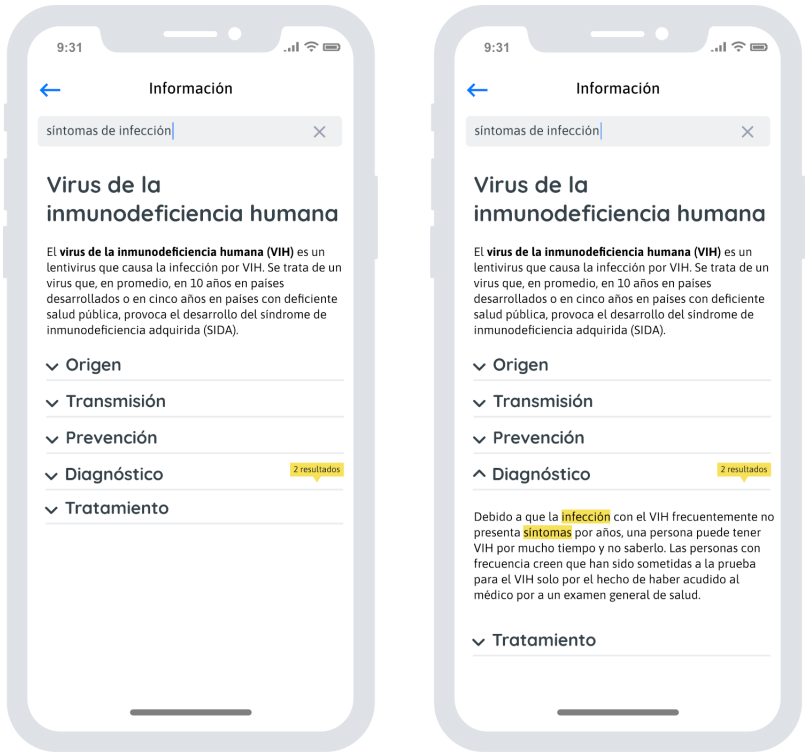

Figura 4.7: Búsqueda de información. Versión móvil

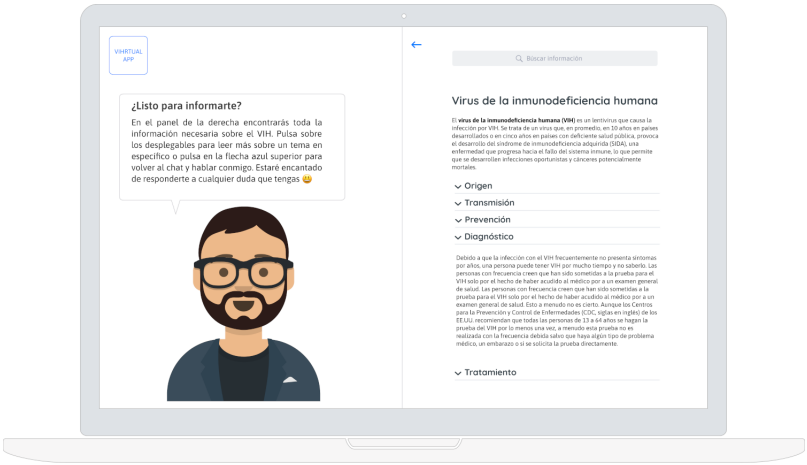

Figura 4.8: Búsqueda de información. Versión de escritorio

Estos *mockups* se envían con un pequeño vídeo y encuesta a los colaboradores la Unidad de Enfermedades Infecciosas del Hospital General de Elche para obtener su *feedback* y aprobación. Tanto sus indicaciones como la versión interactiva pueden encontrarse en los documentos anexos a este trabajo.

4.3. Implementación

Siguiendo la planificación expuesta en la sección 2.3, la implementación se realiza de manera progresiva, centrando inicialmente los esfuerzos en el desarrollo de un prototipo. El objetivo es disponer lo antes posible de una versión de prueba del chatbot, publicarla y probarla con un pequeño subconjunto de personas. Esto permite recopilar conversaciones reales de los usuarios y orientar el desarrollo hacia las necesidades detectadas. Durante este proceso, el asistente se irá nutriendo de información y mejorando hasta alcanzar un estado de madurez suficiente. Por tanto, tal y como se puede observar en la Figura 4.9, a lo largo del desarrollo la aplicación irá evolucionando desde el prototipo inicial, pasando por una fase *beta* hasta ser considerada como estable.

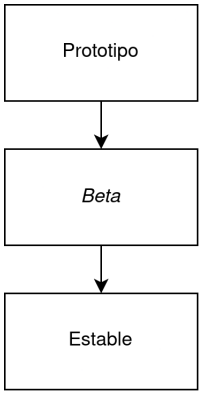

Figura 4.9: Evolución de la aplicación

A continuación, se procede a detallar los distintos pasos seguidos para llevar a cabo este proceso.

4.3.1. Conjunto de entrenamiento inicial

El conjunto de entrenamiento o *dataset* hace referencia a los datos que se utilizan para entrenar un sistema conversacional en la tarea de reconocer las intenciones expresadas por los usuarios y en responder adecuadamente. En *Rasa*, estos datos se estructuran mediante una serie de archivos en formato *YAML* (*YAML Ain't Markup Language*) que contienen las distintas entidades que representan una conversación. Así, los *intents* (intenciones) definen las diferentes preguntas que los usuarios pueden realizar, las respuestas contienen los mensajes que el chatbot devolverá y las *stories* (historias) establecen los patrones de conversación. Con todos estos datos, posteriormente el asistente trata de identificar correctamente las intenciones expresadas por los usuarios y responder en consecuencia, según le haya sido indicado.

Para que el asistente pueda mantener conversaciones lo más amplias y variadas posibles, se han añadido al conjunto de entrenamiento *intents*, respuestas y *stories* pertenecientes a 3 áreas distintas: conocimiento médico, expresiones básicas y conversación informal. Los datos médicos están orientados a responder todas las dudas médicas sobre el VIH y han sido extraídos de la hoja de cálculo facilitada por parte de la Unidad de Enfermedades Infecciosas del Hospital General de Elche. Este documento consta de 130 preguntas con sus correspondientes respuestas y contiene todo el conocimiento médico que inicialmente tendrá el chatbot.

A parte de todas las cuestiones relacionadas con el VIH, el asistente debe ser también capaz de entender y desenvolverse con una serie de expresiones básicas que suelen formar parte un diálogo. Las conversaciones suelen empezar con un *Hola* o *¡Buenos días!* y suelen ser correspondidas con otro saludo, seguidas de un *¿Cómo estás?* o *¿Cómo va?* y sus correspondientes respuestas. Durante el transcurso, también pueden aparecer otras expresiones como afirmaciones o negaciones (*¡Por supuesto!*, *Claro*, *Para nada* etc) o peticiones como *¿Me puedes ayudar?* o *Necesito tu ayuda*. Finalmente, se suele usar un *Adiós* o *Hasta luego* para despedirse y finalizar la conversación. Todas estas expresiones y muchas otras aparecen tarde o temprano en una conversación, y por tanto, se han incorporado el mayor numero posible de ellas al conjunto de entrenamiento para que el asistente las entienda y pueda hacer uso de ellas. En la la Figura 4.10 se muestra un ejemplo de una pequeña conversación donde se utilizan varias de estas expresiones.

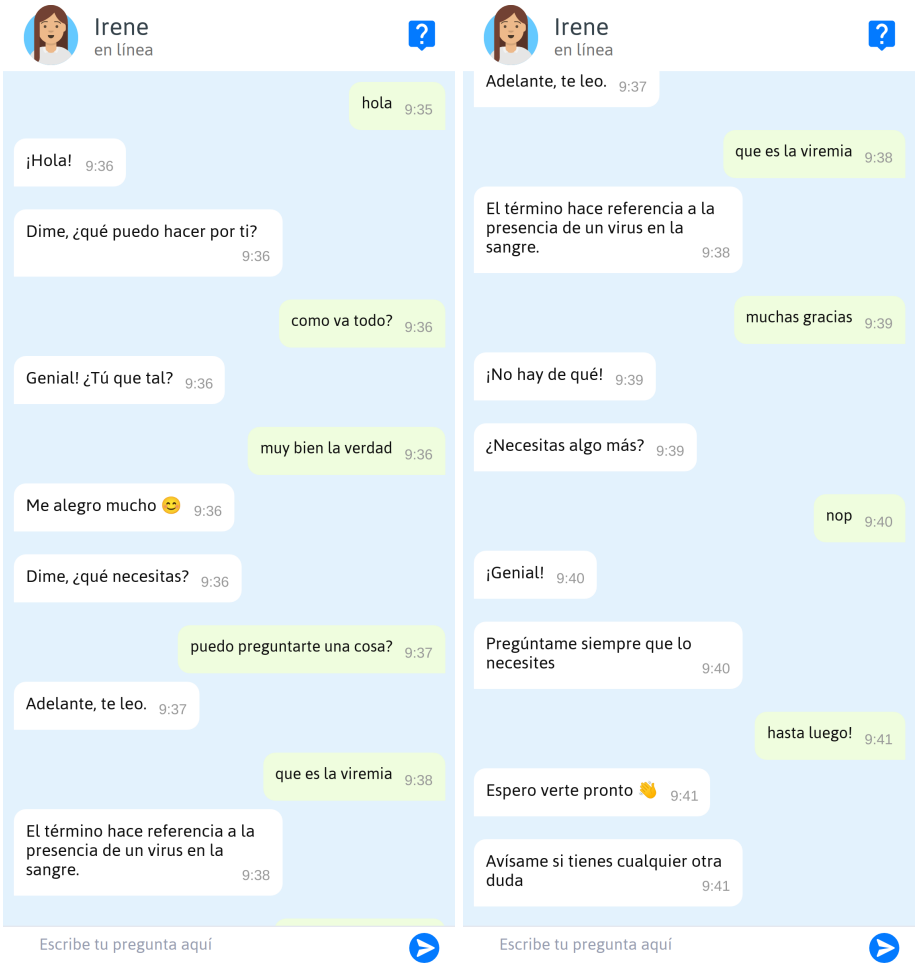

Figura 4.10: Ejemplo de una pequeña conversación

Además, también es común que los usuarios intenten mantener un poco de conversación informal con el bot. Algunos de los ejemplos más repetidos como *¿Cómo te llamas?*, *Quién te ha creado?* o *¿Qué tiempo hace?* han sido también incorporados al conjunto. A continuación se detalla para cada entidad el proceso que se ha seguido para incorporar la información al *dataset*.

Intents

Cada vez que escribimos un mensaje expresamos una intención. Los *intents* representan y definen estas intenciones recopilando una gran cantidad de ejemplos reales. Es importante tener en cuenta que una determinada intención puede ser expresada de distintas maneras. Pongamos por caso la pregunta

¿Existe cura para el VIH? extraída de la hoja de cálculo. La cuestión expresa la intención del usuario de querer saber si existe algún tratamiento que cure completamente el VIH. Sin embargo, esta es simplemente una de las múltiples formas que existen de preguntarlo, pues sería igualmente posible formular *¿Hay cura para el VIH?* o *¿El VIH tiene cura?* expresando la misma intención.

Por tanto, un usuario puede expresar la misma intención a través de diferentes expresiones. Por ello, cada pregunta o expresión a la cuál que se quiera poder responder, se debe analizar para identificar la intención, y en base a esta, elaborar una serie de variaciones que garanticen un amplio espectro de reconocimiento. Estas variaciones consisten en:

- Reformulaciones parciales o completas de las preguntas.
- Variantes con faltas de ortografía.
- Variantes con errores de escritura.

En consecuencia, y siguiendo con el ejemplo anterior, para *¿Existe cura para el VIH?* se pueden obtener diferentes variaciones como las siguientes:

```
- intent: cura_vih
  examples: |
    - ¿Existe cura para el VIH?
    - existe ya cura para el vih?
    - han inventado alguna cura para el vih?
    - que cura hay para el vih?
    - se puede sanar el vih?
    - es posible expulsar el virus del cuerpo?
    - es posible exulsar el vih?
    - tiene cura el sida?
    - como se cura elsida?
    - puede alguienr ecuperarse completamente del vih?
    - ...
```

Es importante considerar que, para que posteriormente el modelo generado distinga de forma fiable una intención de otra, los ejemplos deben ser distintos entre los *intents*. Es decir, no se debe utilizar el mismo ejemplo de entrenamiento para dos intenciones diferentes. Si los ejemplos de entrenamiento resultan demasiado similares, se produce una confusión de intenciones [19].

En esta fase inicial de la implementación, y siguiendo las recomendaciones de *Rasa*, se generan manualmente 30 variaciones por cada intención. En total, se implementan 167 *intents*, (133 acerca del VIH, 23 expresiones básicas y 11 de cháchara). Con esto se pretende garantizar unos resultados suficientemente satisfactorios para lanzar un primer prototipo. En las siguientes fases del desarrollo el número de variaciones aumentará con la recogida de datos reales (ver sección 4.3.5).

Respuestas

Las respuestas ofrecidas por el asistente son la pieza clave para formar y transmitir personalidad. En VIHrtual-App se ha querido crear y dotar al *bot* de una personalidad propia implementando los rasgos comentados en la sección 3.2.2 *Interacción y características sociales*.

Como ejemplo, a continuación se puede observar las posibles respuestas que el chatbot puede dar a un usuario que le pregunte *¿Cómo estas?*. Al responder el chatbot escoge al azar una de las variantes, añadiendo cierto dinamismo a las contestaciones para intentar no transmitir la sensación robótica de los mensajes predefinidos. Además, los textos han sido construidas utilizando varios de los recursos mencionados anteriormente, como el uso de buenos modales, el tono algo informal y la proactividad.

```
utter_estado:
  - text: Genial! ¿Tú que tal?
  - text: Genial, la verdad. Muchas gracias por preguntar.
  - text: Hoy me encuentro genial, ¡gracias por preguntar!
  - text: Hoy me siento especialmente bien. ¿Y tú que tal?
  - ...
```

En el caso de las información médica, se detecta que las respuestas proporcionadas por el hospital son, en muchos casos, difíciles de entender por parte de los usuarios. Algunas utilizan un lenguaje muy técnico y otras un registro demasiado formal que dificulta que el usuario establezca un vínculo con el *bot*. Además, en muchos casos la longitud del mensaje es excesiva, pudiendo provocar rechazo y saturación [7]. En consecuencia, y por la propia naturaleza del medio, es recomendable sintetizar y aportar sólo la información relevante.

Por tanto, es necesario aplicar distintas técnicas para dinamizar las respuestas y no aborrecer al usuario. La información debe ser concreta, responder directamente a la pregunta y en la medida de lo posible intentar captar su intención. Esta tarea se realiza mediante la introducción de modificaciones tales como:

- División de las respuestas más largas en varios mensajes.
- Ligeros cambios para variar el registro por otro menos formal.
- Supresión de afirmaciones y negaciones rotundas para adaptar las respuestas a una mayor variedad de preguntas.
- Hacer uso de algunos emoticonos para amenizar a ciertas expresiones o datos.

Dado el carácter informativo del servicio estas alteraciones se introducen intentando encontrar un equilibrio entre la calidad de la información y la síntesis. Como ejemplo, en la Figura 4.11 se puede observar la respuesta original a la pregunta *¿Qué es la Infección aguda por el VIH?* y su versión final tras las modificaciones.

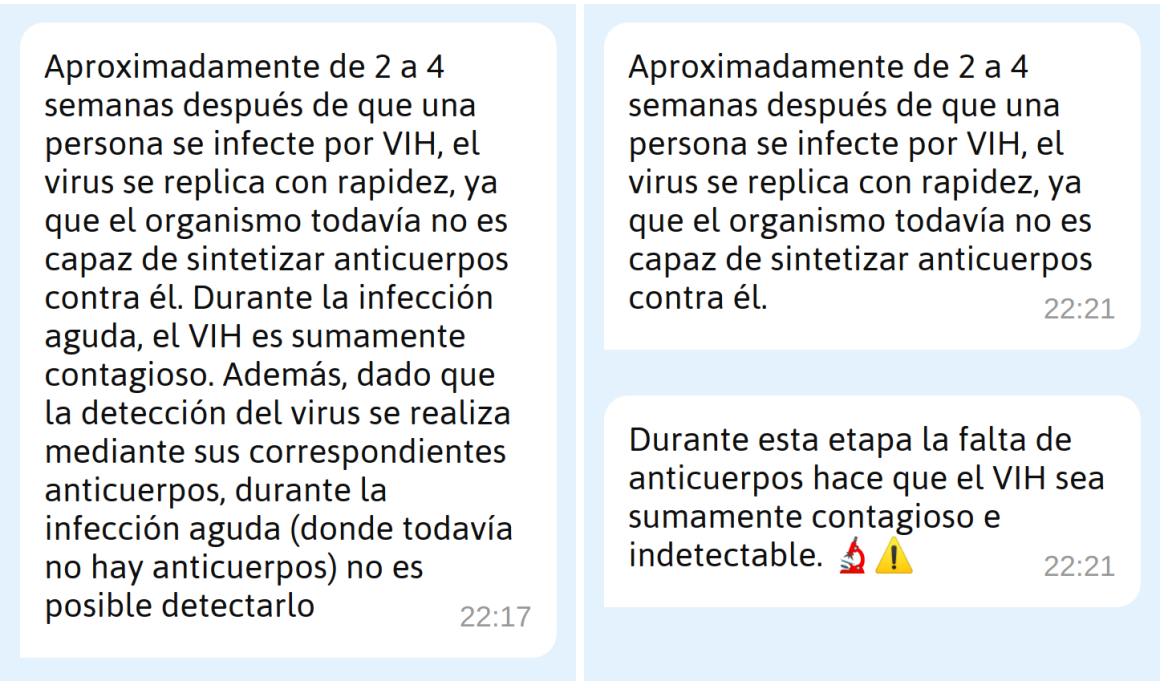

Figura 4.11: Ejemplo de transformación de una respuesta

Stories

Una vez el conjunto de entrenamiento cuenta con la información de los *intents* para identificar las expresiones y sus correspondientes respuestas, es necesario indicarle como debe utilizarlas. Para ello, se incorpora también a los datos de entrenamiento estructuras o patrones de conversación denominadas *stories*. Las historias o *stories* son representaciones de conversaciones entre usuarios y el chatbot. Estos datos utilizan un formato específico en el que la información introducida por el usuario se expresa como *intents* y las respuestas como acciones del asistente [20]. Por tanto, los *intents* permiten identificar los mensajes de entrada, las respuestas contienen la información, y las *stories* aportan el contexto y la lógica necesaria para responder coherentemente.

```
- story: Pregunta por la cura + casos que se han curado + paciente berlin
  steps:
    - intent: cura_vih
    - action: utter_cura_vih
    - intent: faq/casos_cura
    - action: utter_faq
    - intent: faq/paciente_berlin
    - action: utter_faq

- story: Agradecer + necesita algo más
  steps:
    - intent: agradecer
    - action: utter_no_hay_de_que
    - action: utter_algo_mas
    - intent: afirmar
    - action: utter_ofrecer_ayuda
```

Tras procesar toda esta información y realizar el entrenamiento, *Rasa* genera un modelo con el cual el asistente es capaz de analizar las preguntas realizadas por los usuarios, asociarlas a uno de los *intents* que ya tenga definidos (en base a una probabilidad) y ofrecer la respuesta correspondiente según le haya sido indicado.

4.3.2. Servidor (*Backend*)

Una vez el entrenamiento ha finalizado el siguiente paso es hacer accesible el modelo a través de una *API*. Tal y como se ha expuesto en la sección 4.2.1, *Rasa* pone a disposición los servidores ya integrados dentro del propio *framework* con los que se puede publicar el modelo. Simplemente es necesario configurar los puertos por los cuáles se consumirá el servicio y ejecutar los procesos correspondientes.

4.3.3. Vista (*Frontend*)

Debido a su sencillez, la aplicación web se ha construido sin hacer uso de ningún *framework* web, simplemente utilizando *HTML*, *CSS* y *Javascript*. La interfaz sigue los diseños expuestos en la sección 4.2.2, implementando un diseño *responsive*. Para ello, se ha utilizado *Flexbox*, un método de diseño unidimensional para colocar elementos en filas o columnas. Cuando el tamaño de pantalla de un dispositivo varía, los distintos elementos se flexionan para llenar el espacio adicional o se encogen para encajar en espacios más pequeños [21].

Por otro lado, la mayor parte de lógica de la aplicación ha sido implementada utilizando la librería *Chatbot-Widget* [22]. *Chatbot-Widget* es una librería *opensource* que facilita la implementación y conexión de un chat a la *API REST* del servidor de *Rasa*. Con ella se realizan las diferentes llamadas

HTTP entre el servidor y la web, enviando los mensajes del usuario y recibiendo las respuestas ofrecidas por el modelo. Además, también facilita la renderización del texto y otros elementos en el chat.

Durante el desarrollo, se detectó que la librería presentaba ciertos errores y limitaciones que afectaban al caso de uso específico de VIHrtual-App. Por ello, se realizaron las siguientes modificaciones en la librería:

- Corrección de errores en la lógica de llamada-respuesta con el servidor.
- Modificación de las funciones de renderización de texto, añadiendo nuevos elementos visuales y soporte para respuestas con múltiples mensajes.
- Reescritura de la mayor parte del código CSS utilizando flexbox.

Tras su publicación, se corrigieron varios errores que reportaron los usuarios. En la Figura 4.12 se puede observar el resultado final de la aplicación.

Figura 4.12: Diferentes vistas de la aplicación

4.3.4. Despliegue del prototipo

Una vez implementados el servidor y el *frontend* web, es necesario publicar una versión de prueba del servicio para poder empezar con la fase de obtención de datos de conversaciones reales. Para ello, la Universitat Politècnica de València pone a disposición del proyecto una maquina virtual con *Ubuntu 18.04* donde poder realizar el despliegue. Mediante acceso *SSH* se instalan todas las librerías necesarias (*Apache*, *Python*, *Rasa*, *Spacy*, etc) y se ponen en marcha tanto la web como el servidor.

Una vez el sistema está funcionando, se realiza una pequeña prueba con *Rasa X* para asegurarse que la recogida de datos funciona correctamente. Tras ello, se solicita a la UPV la apertura de los puertos 80, 5055 y 5005 para autorizar el tráfico *HTTP* (ver Figura 4.13), y así, permitir que usuarios externos a la universidad puedan acceder al servicio. De esta manera, el servicio se encuentra disponible desde *http://vihrtualapp.gti-ia.upv.es* y se puede empezar con la recogida de datos.

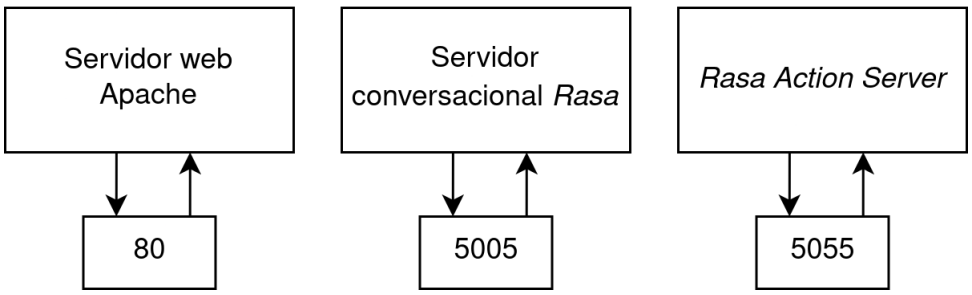

Figura 4.13: Diagrama redirección de puertos y servicios

4.3.5. Desarrollo basado en conversaciones

Uno de los mayores problemas que se suelen presentar durante el desarrollo de un chatbot es que inicialmente, el *dataset* suele estar creado por los desarrolladores, y por tanto, sesgado y alejado de situaciones reales. El desarrollo basado en conversaciones (*Conversation-Driven Development*) es el proceso de escuchar a los usuarios y utilizar esa información para mejorar el asistente [23]. Hacer el chatbot robusto puede ser un auténtico reto pues los usuarios siempre introducirán datos que inicialmente no estaban contemplados en el conjunto de entrenamiento. En consecuencia, este enfoque es la mejor manera de solucionar esta problemática, pues toda nueva información se incorpora al *dataset* y así el modelo se nutre de situaciones y expresiones reales. Esta estrategia está especialmente recomendada por los desarrolladores de *Rasa* [19].

Tras publicar la primera versión de prueba del chatbot el proyecto se encuentra listo para entrar en esta fase de desarrollo centrada en la recogida de datos. Para ello, durante varias semanas se comparte el chatbot, se revisan las conversaciones obtenidas, y se anotan, clasifican e incorporaran al *dataset* todos los datos recopilados. Tras realizar todos estos cambios sobre el conjunto de entrenamiento se ejecutan los *tests* correspondientes (ver sección 4.3.7), se corrigen posibles errores y se vuelve a compartir el chatbot para una nueva iteración (ver Figura 4.14).

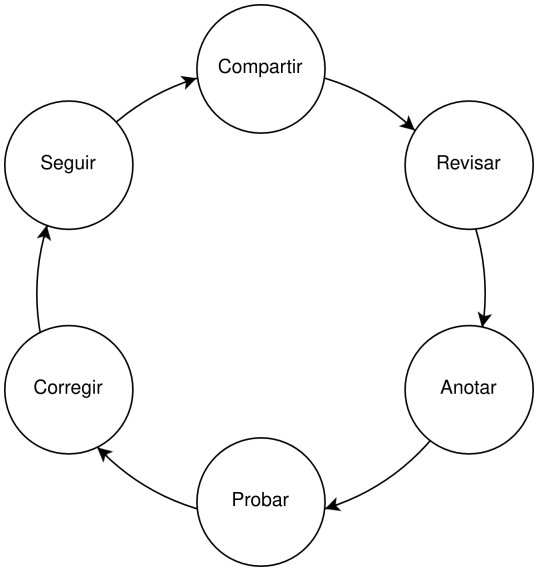

Figura 4.14: Ciclo del desarrollo basado en conversaciones

Para esta nueva fase del desarrollo se hizo uso del servicio *Rasa X*, el cuál facilita la consulta y corrección de todas las conversaciones recogidas. En la Figura 4.15 se puede ver una captura de pantalla del panel de control de *Rasa X*. En la parte izquierda, se pueden consultar todas las conversaciones que el chatbot ha mantenido con los usuarios y, en la derecha, revisar que predicciones de *intents* ha realizado y aplicar las correcciones necesarias. Por otra parte, en la Figura 4.16 se puede observar el

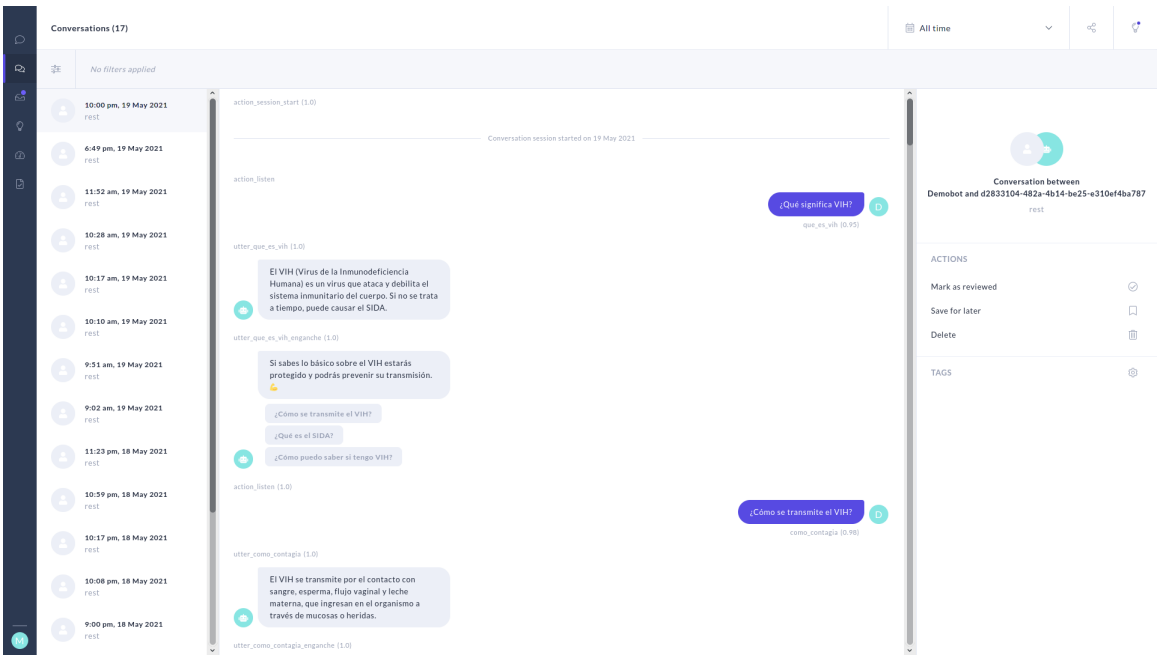

Figura 4.15: Ejemplo de una de las conversaciones recogidas

la herramienta de etiquetado de los datos *NLU* recibidos.

Durante todo este proceso se recogieron y analizaron aproximadamente 60 conversaciones de las cuáles se pudo extraer e incorporar a la base de conocimiento la siguiente información:

- 40 preguntas nuevas (con sus correspondientes respuestas) que inicialmente no estaban previstas.
- 232 variantes o expresiones.

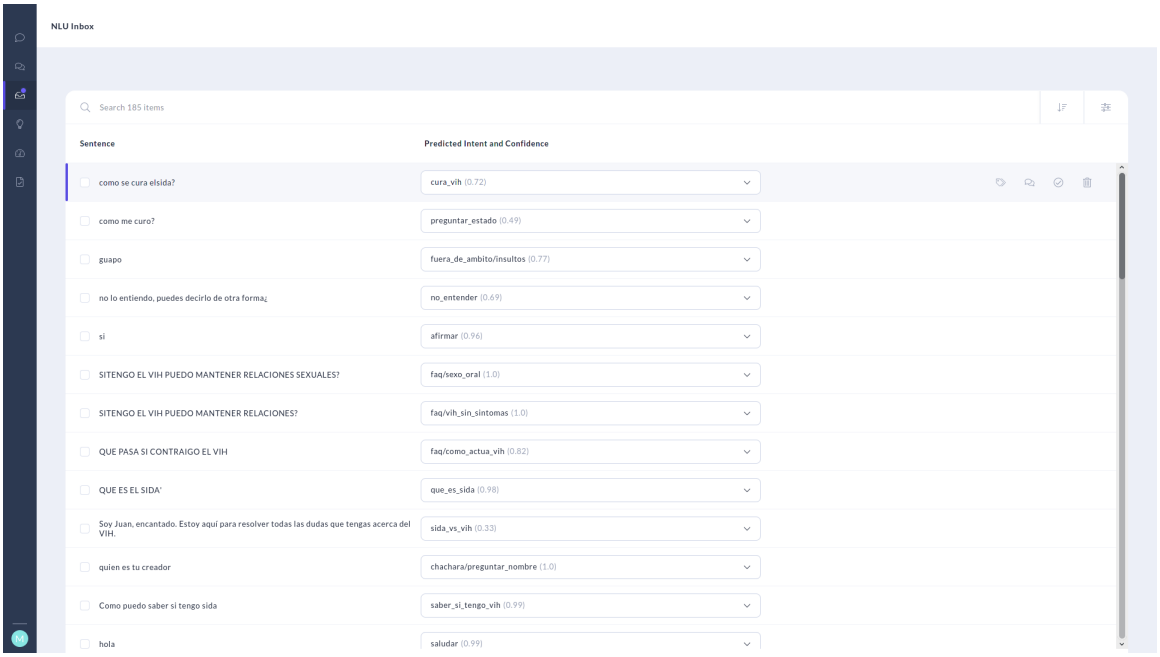

Figura 4.16: Datos NLU recopilados y su etiquetado

4.3.6. Control de daños

A continuación se describe la implementación de diferentes técnicas utilizadas en *VIHrtual-App* para el control de daños, siguiendo las recomendaciones expuestas en la sección 3.2.2.

Falta de conocimiento

Durante el devenir de una conversación es bastante probable que un usuario realice una pregunta que esté fuera del ámbito de conocimiento del chatbot, bien sea por desconocimiento o por poner a prueba el *bot*. Cuando un asistente no responde adecuadamente a este tipo de preguntas puede crear un sentimiento de decepción en el usuario o incluso alentarle a continuar con el comportamiento abusivo [7].

Por tanto, todos los casos detectados durante la recogida de información se han recopilado y agrupado en varios *intents* para poder identificar correctamente estas situaciones y ofrecer una respuesta adecuada, intentado siempre reconducir la situación. En la Figura 4.17 se puede observar como el asistente reconoce la intención del usuario como fuera del ámbito, le responde disculpándose e indicando que no puede ayudarle en esa tarea y le sugiere algunas preguntas a través de unos botones de respuesta rápida.

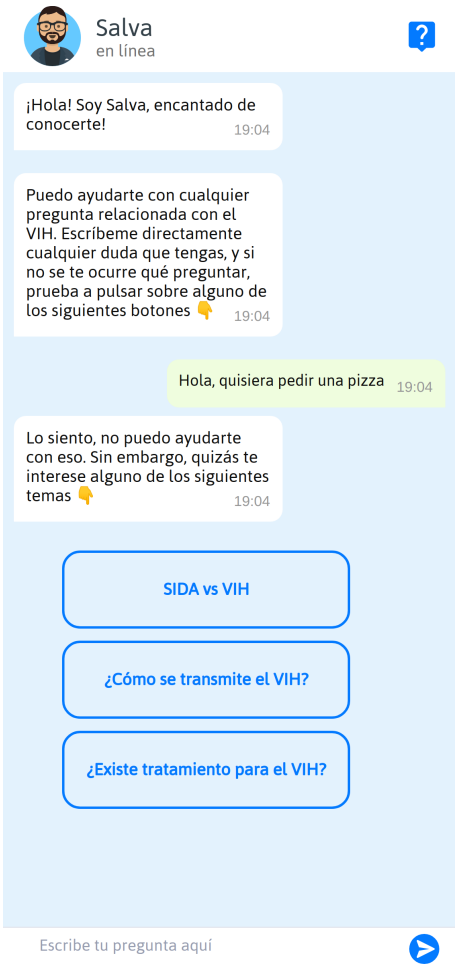

Figura 4.17: Ejemplo de manejo de mensajes fuera de ámbito

Conductas abusivas

Por otra arte, los chatbots suelen ser más objeto de insultos y vejaciones de lo que un humano en una conversación real seria [7]. En este caso, la estrategia es idéntica, se recogen los datos necesarios para reconocer las situaciones y se responde para evitar que el usuario continue con su comportamiento. En la Figura 4.18 se puede ver como responde el *bot* ante un usuario poco amigable.

Fallback

Por último, y como último recurso, cuando el chabot no es capaz de identificar con suficiente confianza un *intent*, muestra un mensaje disculpándose y preguntando amablemente si el usuario es capaz de reformular la pregunta. De esta manera, se intenta forzar al usuario a expresar su intención en otras palabras para poder tener una nueva oportunidad de reconocimiento. En la figura 4.18 se puede observar un caso como el expuesto donde *bot* es capaz de recuperase ante un fallo inicial.

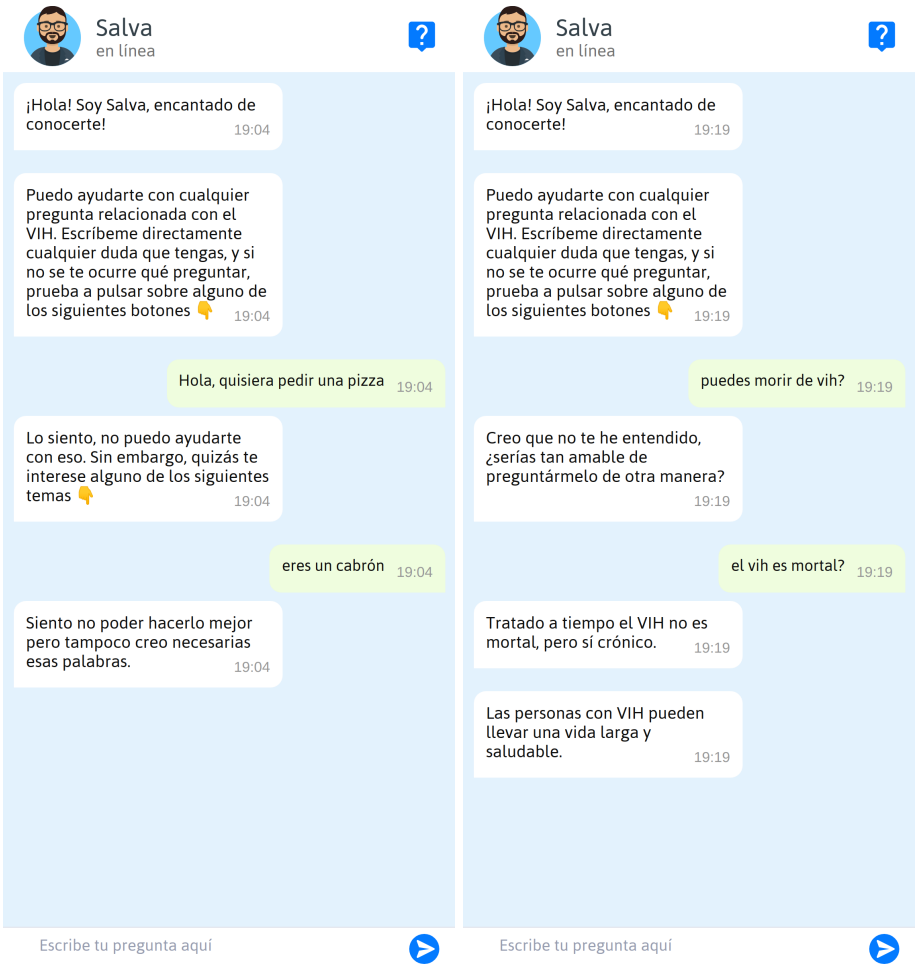

Figura 4.18: Ejemplo de recuperación ante insultos y identificación de *intent*

4.3.7. Tests

Tal y como se ha expuesto anteriormente en la sección 4.3.5, durante el desarrollo basado en conversaciones una parte importante del ciclo cosiste en realizar frecuentemente *tests* automáticos. Tradicionalmente, en el mundo del *software* se entiende por *test* una prueba que verifica que una función devuelve el valor esperado. En *Rasa*, los *test* miden si los modelos generados hacen las predicciones correctas [24].

A medida que el desarrollo avanza, se van incluyendo cada vez más *stories*. Por ello, para evitar inconsistencias es importante asegurarse de que con cada nueva funcionalidad añadida, el modelo continua comportándose como esperamos. Para este cometido *Rasa* ofrece un formato específico con el que escribir tests y así poder asegurarse periódicamente que el modelo no presenta errores.

Las *test stories* son un formato modificado de *stories* que incluyen también los mensajes de los usuarios. De esta manera, los tests aseguran que, ante un determinado *input*, el modelo se comportará de la manera esperada. Siguiendo con el ejemplo visto anteriormente, para la *story* donde el usuario le da las gracias al chatbot y este le pregunta si necesita algo más, se obtienen el siguiente test.

```
- story: Agradecer + necesita algo más
  steps:
  - intent: agradecer
    user: |-
      muchas gracias!
  - action: utter_no_hay_de_que
  - action: utter_algo_mas
  - intent: afirmar
    user: |-
      siii
  - action: utter_ofrecer_ayuda
```

La estrategia seguida para la implementación de los tests en VIHrtual-App ha sido implementar, para cada *story*, un test. De esta manera, cada nuevo patrón de conversación introducido en el modelo dispone de su correspondiente prueba. Esto ha permitido evitar regresiones durante todo el desarrollo.

4.3.8. Distribución

Adicionalmente, y una vez el chatbot sea considerado estable, se publicará para aquellos usuarios que lo deseen una aplicación web empaquetada para *Android* y para *iOS*. Esto permitirá que aquellos usuarios interesados puedan instalar el servicio desde las tiendas de aplicaciones habituales en sus dispositivos móviles.

Aunque aún no está disponible al público, este empaquetado se ha realizado utilizando *Capacitor* [25], una herramienta que permite crear binarios para plataformas móviles.

Capítulo 5

Pruebas de validación

En esta sección se muestran las distintas pruebas de validación que se han realizado a la aplicación. Por una parte se ha realizado una evaluación del modelo NLU para comprobar como se comporta el modelo, y por otra, un estudio de usabilidad con *feedback* de usuarios reales.

5.1. Evaluación del modelo NLU

Una vez el chatbot está publicado, este recibe y procesa información que no forma parte de su conjunto de entrenamiento. Para observar como se comportará el modelo bajo estas circunstancias es posible evaluar el modelo de comprensión del lenguaje natural (NLU). Para ello, previamente se divide el *dataset* en una proporción de 80/20 de datos de entrenamiento y de test respectivamente. Tras esta división, se realiza un entrenamiento con los datos destinados al entrenamiento y se prueba el modelo con los datos de test. En esta prueba, *Rasa* comprobará si el modelo identifica correctamente los *intents*.

Tras realizar el test, se obtiene un informe con los resultados. El informe consta de un archivo en formato *JSON* y una matriz de confusión de *intents*. En la Figura 5.1 se puede observar la matriz de confusión obtenida tras el test. Idealmente, si no se produjera ninguna confusión entre *intents*, debería ser una matriz diagonal. En este caso, aunque gran parte de los datos se concentran en la diagonal (indicando la correcta identificación de un *intent*), si que podemos observar algo de dispersión. Por ejemplo, es significativa la confusión que se produce entre el *intent chachara* y *fuera de ámbito*. Esta confusión se puede deber a la similitud de los datos entre ambos intents, ya que tanto la conversación informal como las expresiones fuera del ámbito del chatbot pueden tener características similares.

Por otra parte, también podemos observar como muchos datos son identificados erróneamente como *faq*. El término *faq* es el *intent* donde se agrupan la mayor parte de *subintents* relacionados con el VIH. Por tanto, es donde más variedad de datos se concentra, dando lugar a las confusiones que muestra la matriz. Por tanto, la evaluación del modelo NLU ha servido para revisar el rendimiento del modelo e identificar que *intents* han dado problemas de confusión. Tras su detección, se ha procedido a la revisión de la calidad de los datos del conjunto de entrenamiento y se han realizado las modificaciones oportunas.

5.2. Evaluación de usabilidad

Con la finalidad de validar el correcto funcionamiento del asistente virtual, se ha realizado una prueba de usabilidad con usuarios reales. De esta manera, se podrían detectar posibles errores y corregirlos. Los participantes fueron mayoritariamente estudiantes de la UPV, captados a través de una invitación por email enviada por los tutores de este trabajo. El correo incluía un breve texto indicando a los participantes que probaran durante unos minutos el chatbot y que, posteriormente, realizaran un cuestionario facilitado a través de un link. Todos los participantes que realizaron el cuestionario han

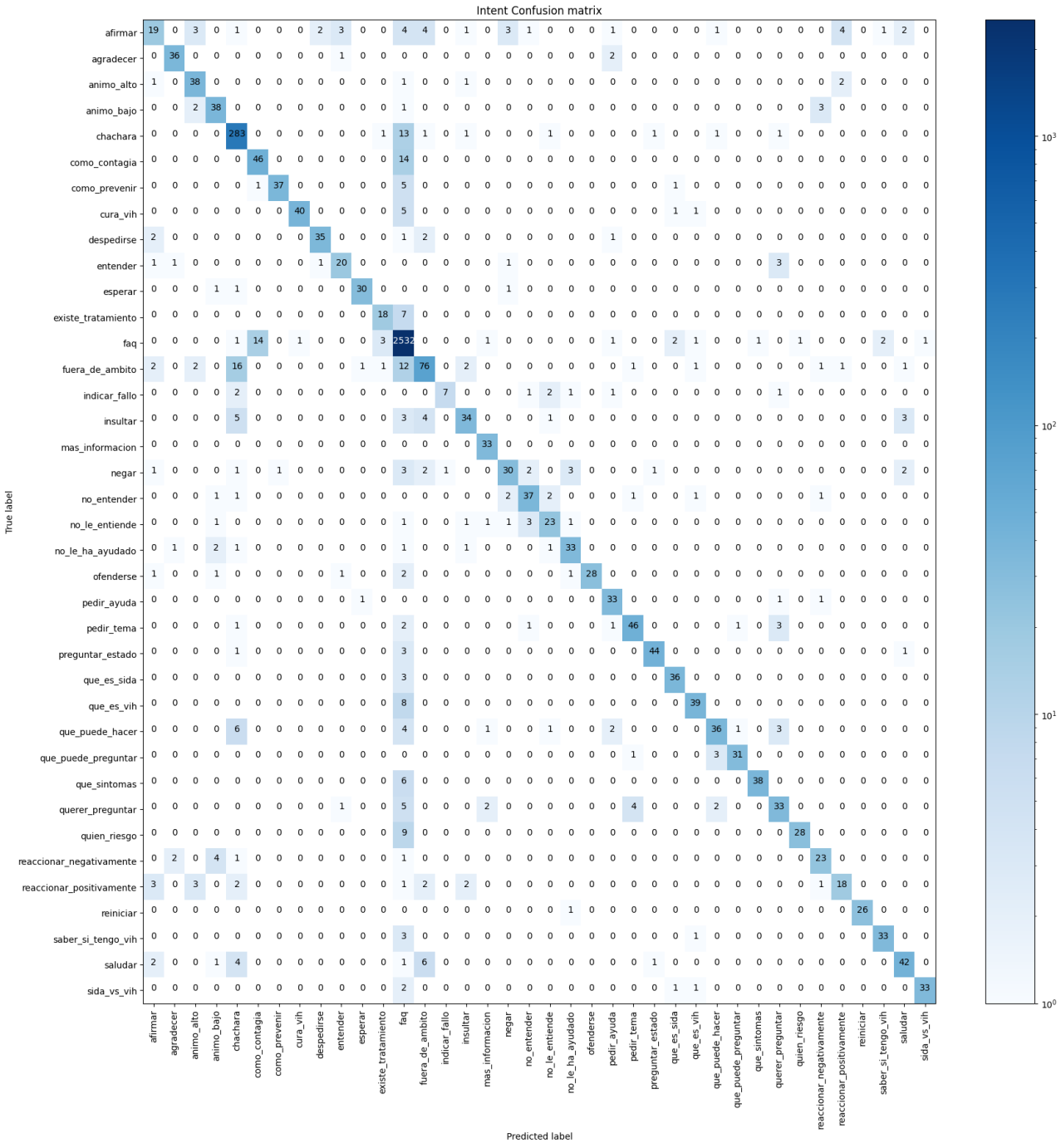

Figura 5.1: Matriz de confusión de *intents*

sido incluidos en los resultados. En total, 12 adultos participaron en la prueba durante un periodo de 7 días.

Para este estudio se han utilizado los cuestionarios validados *System Usability Scale* (SUS) [26] y *Chatbot Usability Questionnaire* (CUQ) [27]. SUS es un cuestionario genérico diseñado para obtener una evaluación general y rápida sobre la usabildiad de una determinada aplicación [26]. Se compone de 10 afirmaciones sobre las cuales los usuarios deben valorar en una escala del 1 al 5 su total disconformidad (1) o total conformidad (5). Se utilizó una versión del SUS traducida al español y validada [28].

Por otro lado, CUQ es un cuestionario de usabilidad específico que evalúa la personalidad, la inteligencia, el entendimiento, la navegación y el manejo de errores de un chatbot [27]. CUQ está diseñado para ser comparable al SUS (utiliza la misma escala de valoración) pero utilizando 16 afirmaciones específicas para chatbots. Para su uso, se realizó una traducción lo más fiel posible al español.

Ambos cuestionarios fueron realizados a través de herramienta *Google Forms* y en los anexos a este documento se puede encontrar un documento con las preguntas y respuestas obtenidas.

5.2.1. Cálculo de puntuaciones

Una vez recogidos todos los datos, se utilizó la hoja de cálculo de puntuación SUS [29] para calcular las puntuaciones sobre 100. La fórmula para este cálculo se muestra en la ecuación 5.1.

$$\overline{SUS} = \frac{1}{n}\sum_{i=1}^{n} norm. \sum_{j=1}^{m} \begin{cases} q_{i,j} - 1, q_{i,j} mod 2 > 0 \\ 5 - q_{i,j,otherwise.} \end{cases} \tag{5.1}$$

donde n=número de sujetos (cuestionarios), m=10 (número de preguntas), $q_{i,j}$=puntuación individual por pregunta por participante y norm=2.5.

Las puntuaciones del CUQ se calcularon sobre 160 utilizando la hoja de cálculo de puntuación CUQ [30]. Posteriormente, los resultados se normalizaron para dar una puntuación sobre 100, permitiendo así la comparación con SUS. La fórmula para el cálculo se muestra en la ecuación 5.2.

$$CUQ = \frac{(\sum_{i=1}^{m} 2p_i - 1) - 8) + 40 - (\sum_{n=i}^{m} 2p_i)}{64} \times 100 \tag{5.2}$$

donde m = 16 (número de preguntas) y p_i = puntuación de cada pregunta individual por participante [27].

5.2.2. Resultados

A continuación se exponen los resultados obtenidos de los 12 participantes que evaluaron la usabilidad de Vihrtual-App.

System Usability Scale (SUS)

En la Figura 5.2 se muestra un diagrama de *box and whisker* con las puntuaciones que recibió la aplicación en el cuestionario SUS. La puntuación media fue de 85.625 sobre 100. La puntuación máxima obtenida fue de 97.5 y la mínima de 65.0. La puntuación media de VIHrtual-App sitúa al chatbot en el rango del percentil 96 al 100, equivalente al *Grado A+* [31]. Si se observan los resultados desglosados por prueba y participante (ver documentos anexos a este trabajo), se puede comprobar como, al igual que en la valoración global, todos los participantes arrojaron una puntuación en SUS

superior a la CUQ. Esto puede deberse a que SUS pueda ser menos efectivo para evaluar la usabilidad de un chatbot, pues sus preguntas son más genéricas y están orientadas a medir aspectos más tradicionales del *UX*.

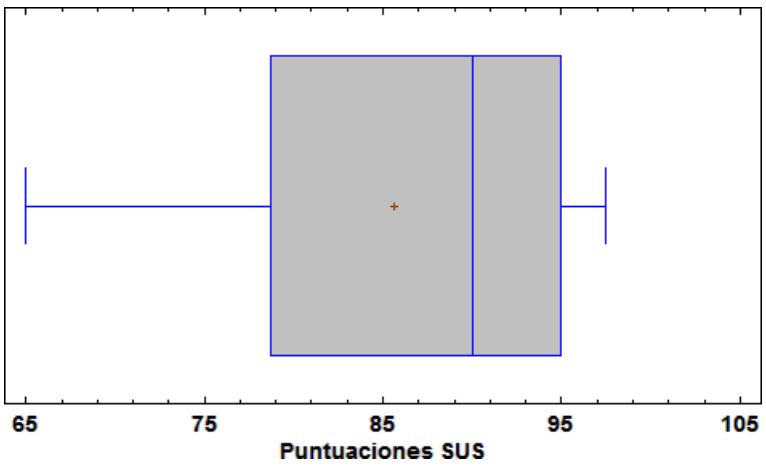

Figura 5.2: Puntuaciones SUS de Vihrtual-App

Chatbot Usability Questionnaire (CUQ)

En la Figura 5.3 se muestra un diagrama de *box and whisker* con las puntuaciones que recibió el chatbot en el cuestionario CUQ. La puntuación media fue de 80.342. La puntuación máxima obtenida fue de 95.3 y la mínima de 65.6. Tal y como se ha mencionado anteriormente, la puntuación media de CUQ es inferior a la SUS, y por tanto, sitúa al chatbot en el rango del percentil 90 al 95, equivalente al *Grado A*.

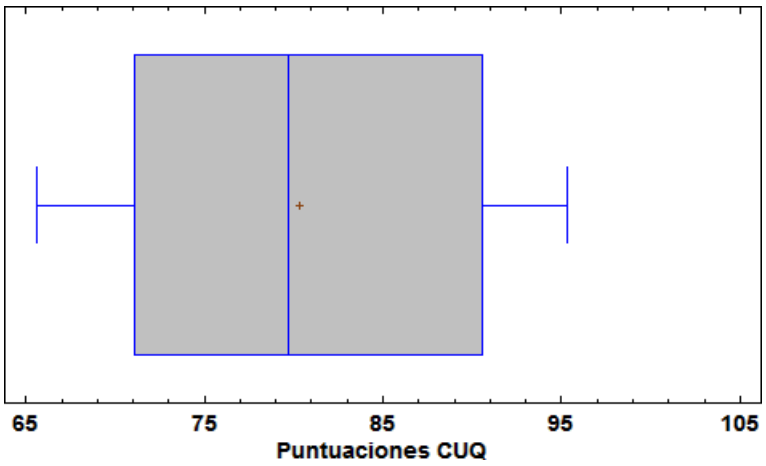

Figura 5.3: Puntuaciones CUQ de Vihrtual-App

Por su parte, CUQ está orientado específicamente a chatbots y permite obtener detalles muy relevantes sobre ciertos aspectos de VIHrtual-App. En este sentido, ha sido notable la valoración positiva sobre la personalidad del chatbot. Como se puede observar en la Figura 5.4, en una escala del 1 (Totalmente en desacuerdo) al 5 (Totalmente de acuerdo) un 83.4 % (66.7 % + 16.7 %) de los participantes expresó estar muy conforme con la afirmación *La personalidad del chatbot era real y cautivadora.*

Por otra parte, el estudio también ha revelado varios aspectos donde se necesita mejorar la experiencia de usuario. Por ejemplo, es significativo observar en la Figura 5.5 las distntas valoraciones obtenidas sobre las afirmaciones *El chatbot falló en reconocer muchas de mis entradas* y *El chatbot me*

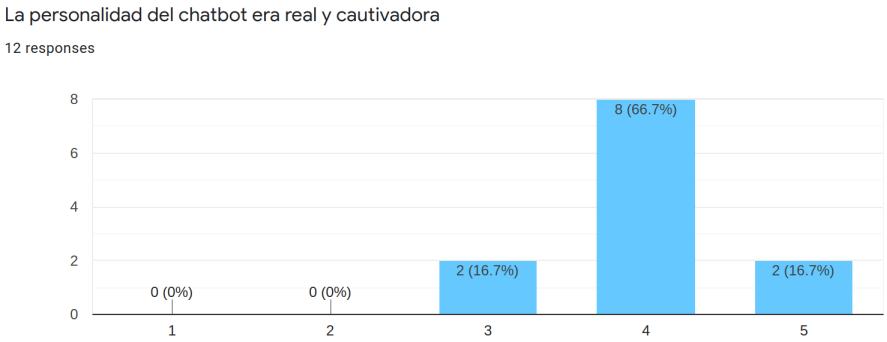

Figura 5.4: Valoración de respuesta

entendió bien.

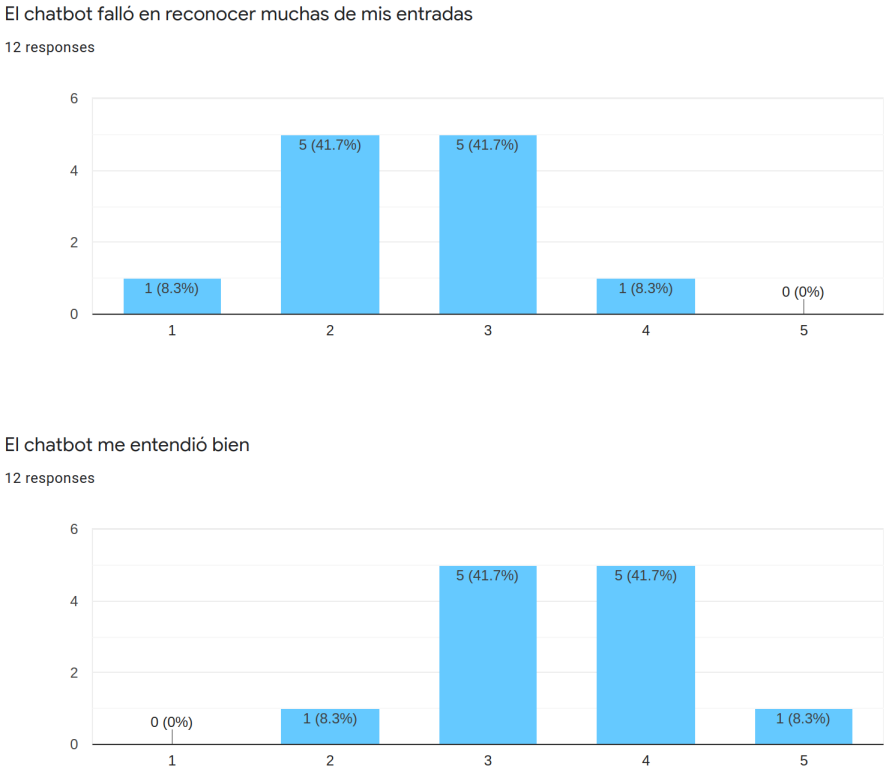

Figura 5.5: Valoración de respuestas

Un 41.7 % de los encuestados valoró con un 3 y un 8.3 % con un 4 su experiencia respecto a que el chatbot no fuera capaz de reconocer muchas de las entradas. Por tanto, al menos la mitad de los encuestados (41.7 % + 8.3 %) manifiestan que el asistente tiene aún mucho margen de mejora respecto al reconocimiento. De igual manera, se han obtenido los mismos resultados (pero invertidos) para *El chatbot me entendió bien.* De nuevo, al menos un 50 % de los participantes expresó que el asistente no entendió bien en algún momento lo que querían expresar. Este hecho se ha podido corroborar analizando varias de las conversaciones recopiladas durante la prueba. En ellas, se puede observar como estos usuarios han realizado distintas preguntas que el chatbot, por falta de conocimiento, no ha sido capaz de responder o ha ofrecido una respuesta equivocada.

En general, los resultados del estudio de usabilidad han sido satisfactorios y el cuestionario CUQ ha permitido detectar aquellos aspectos de VIHrtual-App que necesitan mejorar. En especial, el reconocimiento de las entradas debe mejorarse y la base de conocimiento debe ampliarse.

Capítulo 6

Conclusiones

Durante la realización de este trabajo se han alcanzado con éxitos los objetivos planteados inicialmente. Se ha explorado el estado actual de creación de chatbots y se han estudiado distintos principios de diseño para aplicarlos al proyecto. Posteriormente, todo este conocimiento adquirido se ha plasmado en el diseño e implementación del asistente virtual, implementando todas las funcionalidades inicialmente previstas.

Tras su publicación, se han podido recopilar las conversaciones mantenidas por los usuarios con el servicio, de manera que ha sido posible trabajar en base a esta información para ir corrigiendo y mejorando el servicio. Este modo de trabajo ha producido una buena sinergia con los colaboradores de la Unidad de Enfermedades Infecciosas del Hospital General de Elche, que en última instancia, ha repercutido positivamente en el desarrollo. Gracias a la recogida de datos y su colaboración, ha sido posible recopilar y responder hasta 40 nuevas preguntas que inicialmente no estaban previstas, añadiendo una notable mejora a la base de conocimiento del chatbot.

Por otra parte, también se ha realizado una evaluación de usabilidad con 12 participantes que respondieron a los cuestionarios validados SUS y CUQ. Ambas pruebas arrojaron unos resultados de 85.625 y 80.342 sobre 100 respectivamente. Estas valoraciones positivas corroboran, en gran parte, muchas de las características sociales implementadas en el diseño de interacción. Por otro lado, también han servido para mostrar aquellos puntos donde es necesario continuar trabajando para mejorar la experiencia. En especial, el reconocimiento de entradas tiene margen de mejora y la cantidad de preguntas que puede responder el chatbot se debe aumentar.

Es importante considerar que, sólo 12 usuarios participaron en la prueba, y de ellos, la mayor parte fueron estudiantes. Por tanto, el análisis de los resultados debe realizarse teniendo en cuenta este contexto. Sería necesario un estudio más extenso donde, tanto la cantidad de encuestados como la variedad de sus perfiles fuera mayor. En este caso, gran parte de los participantes eran jóvenes y tenían estudios tecnológicos, por lo que los resultados obtenidos pueden estar algo sesgados.

En general, los resultados obtenidos han sido satisfactorios, y se espera que continuando con desarrollo basado en conversaciones se corrijan todos aquellos aspectos que son susceptibles de mejorar. En un futuro, la intención es que el chatbot sea también capaz de responder a otros temas relacionados con el VIH como puedan ser las las enfermedades de transmisión sexual.

A nivel personal, este trabajo de final de grado ha supuesto todo un impulso a la formación recibida en el grado. El reto de diseñar, dirigir e implementar una idea ha sido toda una experiencia. Durante todo el desarrollo ha sido necesario un aprendizaje permanente para poder resolver los problemas inicialmente planteados, superar las dificultades y lidiar con la planificación del proyecto. Igualmente, trabajar con un grupo de profesionales de un ámbito distinto ha supuesto una mejora de mis capacidades comunicativas, y sobretodo, una experiencia muy enriquecedora. Este trabajo ha supuesto para mi toda una introducción en el mundo del desarrollo de chatbots, y llevar a cabo un proyecto como

este me ha otorgado las aptitudes necesarias para orientar mi futura vida profesional hacia el sector de la inteligencia artificial.

Capítulo 7

Referencias

[1] Wlodek Zadrozny y col. "Natural Language Dialogue for Personalized Interaction". En: *Commun. ACM* 43 (ago. de 2000), págs. 116-120. DOI: 10.1145/345124.345164.

[2] Joao Luis Montenegro, Cristiano André da Costa y Rodrigo Righi. "Survey of Conversational Agents in Health". En: *Expert Systems with Applications* 129 (abr. de 2019). DOI: 10.1016/j.eswa.2019.03.054.

[3] Gobierno de España. Ministerio de Sanidad. *Vigilancia epidemiológica del VIH y SIDA en España*. 2019. URL: https://www.mscbs.gob.es/ciudadanos/enfLesiones/enfTransmisibles/sida/vigilancia/Informe_VIH_SIDA_20201130.pdf.

[4] *Repositorio app Vihrtual-App*. URL: https://github.com/joancipria/VihrtualApp-app.

[5] *Repositorio servidor Vihrtual-App*. URL: https://github.com/joancipria/VihrtualApp.

[6] Sameera Abdul-Kader y John Woods. "Survey on Chatbot Design Techniques in Speech Conversation Systems". En: (jul. de 2015).

[7] Ana Paula Chaves Steinmacher y Marco Aurelio Gerosa. "How Should My Chatbot Interact? A Survey on Social Characteristics in Human–Chatbot Interaction Design". En: *International Journal of Human-Computer Interaction* 37 (nov. de 2020), págs. 1-30. DOI: 10.1080/10447318.2020.1841438.

[8] Shafquat Hussain, Omid Sianaki y Nedal Ababneh. "A Survey on Conversational Agents/Chatbots Classification and Design Techniques". En: mar. de 2019, págs. 946-956. ISBN: 978-3-319-98284-7. DOI: 10.1007/978-3-030-15035-8_93.

[9] Ketakee Nimavat y Tushar Champaneria. "Chatbots: An overview. Types, Architecture, Tools and Future Possibilities". En: oct. de 2017.

[10] *IRA Chatbot*. URL: https://yes4me.net/IRA.

[11] *Florence Chatbot*. URL: https://florence.chat/.

[12] Hiroki Tanaka y col. "Detecting Dementia Through Interactive Computer Avatars". En: *IEEE Journal of Translational Engineering in Health and Medicine* 5 (2017), págs. 1-11. DOI: 10.1109/JTEHM.2017.2752152.

[13] *Wakamola Chatbot*. URL: http://wakamola.webs.upv.es/.

[14] *Rasa*. URL: https://rasa.com/.

[15] *Amazon Lex*. URL: https://aws.amazon.com/lex/.

[16] *Botkit*. URL: https://www.botkit.ai/.

[17] *LUIS.ai*. URL: https://www.luis.ai/.

[18] Eva Heller. *Psicología del color: Cómo actúan los colores sobre los sentimientos y la razón*. 2004.

[19] Karen White. *10 Best Practices for Designing NLU Training Data*. 2020. URL: https://blog.rasa.com/10-best-practices-for-designing-nlu-training-data/.

[20] Justina Petraityte. *Designing Rasa training stories*. 2019. URL: https://blog.rasa.com/designing-rasa-training-stories/.

[21] *Flexbox*. URL: https://developer.mozilla.org/en-US/docs/Learn/CSS/CSS_layout/Flexbox.

[22] *Chatbot-Widget*. URL: https://github.com/JiteshGaikwad/Chatbot-Widget.

[23] Alan Nichol. *Conversation-Driven Development*. 2020. URL: https://blog.rasa.com/conversation-driven-development-a-better-approach-to-building-ai-assistants/.

[24] Karen White. *Write Tests! How to Make Automated Testing Part of Your Rasa Dev Workflow*. 2020. URL: https://blog.rasa.com/rasa-automated-tests/.

[25] *Capacitor*. URL: https://capacitorjs.com/.

[26] John Brooke. "SUS: A quick and dirty usability scale". En: (1996).

[27] Samuel Holmes y col. "Usability Testing of a Healthcare Chatbot: Can We Use Conventional Methods to Assess Conversational User Interfaces?" En: *Proceedings of the 31st European Conference on Cognitive Ergonomics*. ECCE 2019. BELFAST, United Kingdom: Association for Computing Machinery, 2019, 207–214. ISBN: 9781450371667. DOI: 10.1145/3335082.3335094. URL: https://doi.org/10.1145/3335082.3335094.

[28] Sevilla-Gonzalez MDR. Moreno Loaeza L. Lazaro-Carrera LS. Bourguet Ramirez B. Vázquez Rodríguez A. Peralta-Pedrero ML. Almeda-Valdes P. "Spanish Version of the System Usability Scale for the Assessment of Electronic Tools: Development and Validation". En: *JMIR Hum Factors* (2020), págs. 1-11. DOI: 10.1109/JTEHM.2017.2752152.

[29] T. Tullis y B. Albert. *Measuring the User Experience: SUS Calculation Tool*. 2018.

[30] Samuel Holmes. Anne Moorhead. Raymond Bond. Huiru Zheng. Vivien Coates. y Michael Mctear. *The Chatbot Usability Questionnaire (CUQ)*. 2019. URL: https://www.ulster.ac.uk/research/topic/computer-science/artificial-intelligence/projects/cuq.

[31] James R. Lewis y Jeff Sauro. "Item Benchmarks for the System Usability Scale". En: *J. Usability Studies* 13.3 (mayo de 2018), 158–167. ISSN: 1931-3357.