UNIVERSITAT POLITÈCNICA DE VALÈNCIA

ESCUELA TÉCNICA SUPERIOR INGENIERÍA INDUSTRIAL VALENCIA

TRABAJO FIN DE MASTER EN INGENIERÍA BIOMÉDICA

DISEÑO DE UN CHATBOT WEB PARA EL ESTUDIO DE LA OBESIDAD Y EL SOBREPESO

AUTOR: EMILIO PARDO DOMÉNECH

TUTOR: JUAN MIGUEL GARCÍA GÓMEZ

COTUTORA: SABINA ASENSIO CUESTA

Curso Académico: 2020-21

AGRADECIMIENTOS

Me gustaría dar las gracias al grupo de investigación BDSLab por permitirme colaborar con ellos durante todo este tiempo (aunque se hayan tenido que trabajar a distancia debido al coronavirus) a mis tutores Sabina Asensio Cuesta y Juan Miguel García Gómez, por darme la oportunidad de desarrollar mis capacidades como programador y darme una breve pero intensa experiencia laboral. También quiero destacar la ayuda prestada por mi compañero de proyecto Vicent Blanes Selva y a todos mis amigos programadores, un trozo de todos ellos está en este proyecto y, por supuesto, a mi familia por estar siempre ahí.

Sin todos vosotros, este proyecto no hubiera sido posible.

RESUMEN

La obesidad y el sobrepeso son uno de los mayores problemas de salud de la sociedad española actual. Estas enfermedades están asociadas a malos hábitos de vida, como el sedentarismo o pautas nutricionales incorrectas. Estas patologías tienden a presentarse junto a otras comorbilidades como la diabetes mellitus tipo 2, apnea obstructiva del sueño, hipertensión, infiltraciones grasas en el hígado, entre otras.

La obesidad ha sido estudiada desde múltiples perspectivas, pero poco se sabe sobre cómo afecta la obesidad a nivel de entorno del individuo. Es decir, puede contagiarse “la obesidad”. Para recoger información acerca de este aspecto se desarrolló en 2019 el proyecto Wakamola.

Wakamola es un chatbot desarrollado por la Universitat Politécnica de Valencia en la aplicación de mensajería Telegram. Su objetivo es alcanzar el máximo de población española adulta, pero la aplicación de mensajería Telegram se utiliza poco en España. Por ello, se ha buscado la alternativa de realizar una aplicación web con un diseño adaptado y nuevas funcionalidades añadidas.

Por un lado, se ha desarrollado una aplicación web que permite la recogida de datos del usuario mediante una encuesta guiada. Además, el usuario puede establecer relaciones con otros usuarios, de forma que puede estudiarse el entorno de cada usuario. Por otro lado, se ha realizado un estudio piloto y se han analizado los datos recopilados por Wakamola en relación a los hábitos de vida y sobrepeso, así como la opinión de los participantes sobre la usabilidad de la solución web propuesta.

RESUM

L' obesitat i el sobrepès són uns dels majors problemes de salut de la societat espanyola actual. Estes malalties estan associades a mals hàbits de vida, com el sedentarisme o pautes nutricionals incorrectes. Aquestes patologies tendeixen a presentar-se junt amb altres comorbiditats com la diabetis mellitus tipus 2, apnea obstructiva de la son, hipertensió, infiltracions grasses en el fetge, entre altres.

L'obesitat ha sigut estudiada des de múltiples perspectives, però poc es sap sobre com afecta l'obesitat a nivell d'entorn de l'individu. És a dir, pot contagiar-se "l'obesitat". Per a recollir informació sobre aquest aspecte es va desenvolupar en 2019 el projecte Wakamola.

Wakamola és un chatbot desenvolupat per la Universitat Politècnica de València en l'aplicació de missatgeria Telegram. El seu objectiu és assolir el màxim de la població espanyola adulta, però l'aplicació de missatgeria Telegram s'empra poc en Espanya. Per això, s'ha buscat una alternativa de realitzar una aplicació web amb un disseny adaptat i altres funcionalitats afegides.

Per un costat, s'ha desenvolupat una aplicació web que permet la recollida de dades de l'usuari mitjançant una enquesta guiada. A més, l'usuari pot establir relacions en altres usuaris, de forma que es pot estudiar l'entorn de cada usuari. Per altra banda, s'ha realitzat un estudi pilot i s'han analitzat les dades recopilades per Wakamola en relació amb els hàbits de vida saludable i sobrepès, aixina com l'opinió dels participants sobre la usabilitat de la solució proposta.

ABSTRACT

Obesity and overweight are one of the biggest health problems in current Spanish society. These diseases are associated with bad lifestyle habits, such as sedentary lifestyle or incorrect nutritional guidelines. These pathologies tend to present together with other comorbidities such as type 2 diabetes mellitus, obstructive sleep apnea, hypertension, fatty infiltrations in the liver, among others.

Obesity has been studied from multiple perspectives, but little is known about how obesity affects the individual's environment. In other words, "obesity" can be spread. To collect information about this aspect, the Wakamola project was developed in 2019.

Wakamola is a chatbot developed by the Polytechnic University of Valencia in the Telegram messaging application. Its objective is to reach the maximum of the adult Spanish population, but the Telegram messaging application is little used in Spain. For this reason, the alternative of creating a web application with an adapted design and new added functionalities has been sought.

On the one hand, a web application has been developed that allows the collection of user data through a guided survey. In addition, the user can establish relationships with other users, so that the environment of each user can be studied. On the other hand, a pilot study has been conducted and the data collected by Wakamola have been analyzed related to healthy lifestyle and overweight, as well as the participants opinion about the web proposed solution usability.

Índice

1 Introducción 1
1.1 Contexto social 1
1.1.1 El problema de la obesidad y el sobrepeso 1
1.1.2 Prevalencia de la obesidad y el sobrepeso en la sociedad 2
1.2 Contexto tecnológico 3
1.2.1 Paradigma de las plataformas de desarrollo de aplicaciones 3
1.2.2 Paradigma del desarrollo de aplicaciones web 5
1.2.3 Comunicación entre máquinas en una red 6
1.2.4 Chatbots orientados al ámbito de la nutrición y de la actividad física 10
1.2.5 Aplicaciones en el mercado 11
1.3 Contexto del proyecto 13
1.3.1 El proyecto Wakamola 13
1.3.2 Wakamola Telegram 14
1.3.3 Análisis del funcionamiento de Wakamola Telegram 15
2 Objetivos del proyecto 21
3 Análisis del problema 22
3.1 Estudio de mercado de las posibles plataformas en las que implantar Wakamola 23
3.2 Estudio de la viabilidad de la implantación en las diferentes plataformas 24
3.2.1 Insertar un chatbot en redes sociales 24
3.2.1.1 WhatsApp 24
3.2.1.2 Facebook 25
3.2.1.3 Instagram 25
3.2.1.4 Line 25
3.2.2 Insertar un bot en un sitio web 27
3.2.2.1 Xenioo 27
3.2.2.2 Azure 28
3.2.2.3 SnatchBot 28
4 Solución final propuesta 29
4.1 Arquitectura del nuevo Wakamola 29
4.2 Etapas de desarrollo del proyecto 31
5 Tecnologías utilizadas 32
5.1 Lenguajes utilizados 32

5.2 Librerías, frameworks, middlewares y utilidades utilizados 34
5.3 Herramientas utilizadas 36
6 Ejecución del proyecto 38
6.1 Desarrollo del Frontend 38
6.1.1 Diseño del Frontend 38
6.2 Desarrollo del Backend 44
6.2.1 Estructura de los JSON enviados 44
6.2.2 Algoritmo del Backend 55
6.2.3 Arquitectura de la base de datos 57
6.3 Implantación del software en servidor 59
7 Pruebas de usabilidad 60
7.1 Task Test 60
7.2 System Usability Scale (SUS) 61
7.3 User Experience Questionnaire (UEQ) 63
8 Resultados obtenidos de la realización de las pruebas 65
8.1 Aplicación de Wakamola 65
8.2 Task Test 71
8.3 SUS 81
8.4 UEQ 82
9 Conclusiones 84
10 Líneas futuras 85
11 Bibliografía 86

Índice de tablas

Tabla 1 Características y diferencias entre las páginas web, Web App, Progressive Web App y las aplicaciones nativas. [6] 4
Tabla 2 Descripción de los campos de la tabla Users 57
Tabla 3 Descripción de los campos de la tabla Responses 57
Tabla 4 Descripción de los campos de la tabla Relationship 57
Tabla 5 Descripción de los campos de la tabla Status 58
Tabla 6 Agrupación de las cualidades de la experiencia del usuario en el UEQ [51] 63
Tabla 7 Parejas de términos negativos y positivos utilizados para el UEQ-S 64

Índice de figuras

Figura 1 Datos del Índice de Masa Corporal separadas por sexos en España desde 1975 hasta 2014 (Di Cesare et al., 2016) 2
Figura 2 Gráfica del porcentaje de uso de cada red social en la sociedad española en 2020 (IAB Spain, 2020)..... 22
Figura 3 Diagrama de Gantt del proyecto Wakamola web 38
Figura 4 Algoritmo de las peticiones POST al Backend de Wakamola Web 56
Figura 5 Distribución por géneros participantes de la prueba..... 65
Figura 6 Distribución por estado social de los participantes de la prueba 65
Figura 7 Distribución de las edades de los participantes de la prueba..... 66
Figura 8 Distribución del peso de los participantes de la prueba 66
Figura 9 Distribución de la altura de los participantes de la prueba 67
Figura 10 Distribuciones de las respuestas sobre la dieta de los participantes de la prueba 68
Figura 11 Distribuciones sobre las respuestas de los participantes de la actividad física en días 69
Figura 12 Distribuciones de las respuestas de los participantes de la prueba sobre la actividad física en minutos dedicados a cada tipo de actividad..... 69
Figura 13 Distribución de las respuestas de los participantes sobre la cantidad de horas que permanecen sentados..... 70
Figura 14 Resultados obtenidos de las valoraciones de los usuarios a la facilidad de la tarea (1) y al tiempo empleado en completarla (2)..... 71
Figura 15 Representación de la cantidad reportada por los usuarios que han cometido al realizar la tarea..... 72
Figura 16 Representación de si los usuarios han comprendido los datos que han introducido para realizar la tarea (en este caso el nombre de usuario y la contraseña) 72
Figura 17 Resultados obtenidos de las valoraciones de los usuarios a la facilidad de la tarea (1) y al tiempo empleado en completarla (2)..... 73
Figura 18 Representación de la cantidad reportada por los usuarios que han cometido al realizar la tarea..... 74
Figura 19 Representación de si los usuarios han comprendido los datos que han introducido para realizar la tarea (en este caso el nombre de usuario y la contraseña) 74
Figura 20 Resultados obtenidos de las valoraciones de los usuarios a la facilidad de la tarea (1) y al tiempo empleado en completarla (2)..... 75
Figura 21 Representación de la cantidad reportada por los usuarios que han cometido al realizar la tarea..... 76
Figura 22 Resultados obtenidos de las valoraciones de los usuarios a la facilidad de la tarea (1) y al tiempo empleado en completarla (2)..... 77
Figura 23 Representación de la cantidad reportada por los usuarios que han cometido al realizar la tarea..... 78
Figura 24 Representación de si los usuarios han comprendido los datos que han introducido para realizar la tarea (en este caso el nombre de usuario y la contraseña) 78
Figura 25 Resultados obtenidos de las valoraciones de los usuarios a la facilidad de la tarea (1) y al tiempo empleado en completarla (2)..... 79

Figura 26 Representación de la cantidad reportada por los usuarios que han cometido al realizar la tarea 80
Figura 27 Representación de si los usuarios han comprendido los datos que han introducido para realizar la tarea (en este caso las entradas en las encuestas) 80
Figura 28 Distribuciones de las respuestas de los usuarios a las preguntas del System Usability Scale 81
Figura 29 Distribución de las puntuaciones de los usuarios a la usabilidad de Wakamola web 81
Figura 30 Representación de las respuestas al User Experience Questionnaire 82
Figura 31 Representación de las valoraciones de los usuarios de las cualidades pragmáticas, hedónicas y el atractivo general de Wakamola web 83
Figura 32 Representación de la comparación de los valores de las cualidades generales del UEQ con otros productos de la base de datos del UEQ 83

Índice de ilustraciones

Ilustración 1 Representación del Backend y del Frontend 5
Ilustración 2 Ejemplo de una petición HTTP (“Generalidades del protocolo HTTP - HTTP | MDN”, s. f.) 6
Ilustración 3 Ejemplo de una respuesta HTTP (“Generalidades del protocolo HTTP - HTTP | MDN”, s. f.) 7
Ilustración 4 Representación del funcionamiento de un socket. Fuente: Elaboración propia 8
Ilustración 5 Ejemplo de la notación de JSON 9
Ilustración 6 Ejemplo de la experiencia de usuario del chatbot sobre el COVID-19 de la OMS . 11
Ilustración 7 Visualización del usuario de la aplicación Lark 11
Ilustración 8 Ejemplo de experiencia del usuario del chatbot "Diet Planner" 12
Ilustración 9 Visualización de la aplicación Jolt.ai 12
Ilustración 10 Icono de Wakamola 13
Ilustración 11 Captura de pantalla del inicio del uso de la aplicación Wakamola 15
Ilustración 12 Captura de pantalla de la encuesta personal de la aplicación Wakamola Telegram 16
Ilustración 13 Captura de pantalla de diferentes matrices de botones de la aplicación Wakamola 17
Ilustración 14 Captura de pantalla de la encuesta de dieta de la aplicación Wakamola Telegram 17
Ilustración 15 Captura de pantalla de la encuesta de actividad física de la aplicación Wakamola Telegram 18
Ilustración 16 Captura de pantalla del Wakaestado de la aplicación Wakamola Telegram 19
Ilustración 17 Captura de pantalla de la opción de compartir de la aplicación Wakamola Telegram 19
Ilustración 18 Captura de pantalla de la Wakarred de la aplicación Wakamola Telegram 20
Ilustración 19 Captura de la página web de la Wakarred de la aplicación de Wakamola Telegram 20
Ilustración 20 Página de la aplicación de AutoResponder para WA 24
Ilustración 21 Ejemplo del desarrollo de una aplicación con Xenioo 27
Ilustración 22 Estructura de la aplicación 29

Ilustración 23 Rutas y estructuras involucradas en la comunicación de máquinas en la aplicación de Wakamola web 30
Ilustración 24 Imagen de la pantalla del inicio de la aplicación 39
Ilustración 25 Visualización de la pantalla de inicio de la aplicación en un smartphone 40
Ilustración 26 Detalla del mensaje de error que aparece en caso de no introducir correctamente los datos en la forma 41
Ilustración 27 Captura de pantalla de la aplicación una vez se ha iniciado en la pantalla de inicio con los campos laterales plegados 41
Ilustración 28 Captura de pantalla de la aplicación una vez se ha iniciado en la pantalla de inicio con los campos laterales desplegados 42
Ilustración 29 Captura de pantalla de la aplicación una vez se ha iniciado en la pantalla de inicio en un smatrphone 42
Ilustración 30 Captura de pantalla de la aplicación cunado se ha recibido una petición de amistad 43
Ilustración 31Represetnación del proceso de comunicación entre servicios durante el proceso de registrar un usuario 45
Ilustración 32 Representación del proceso de comunicación entre servicios durante el proceso de acceder un usuario 47
Ilustración 33 Representación del proceso de comunicación entre servicios para el envío de mensajes 49
Ilustración 34 Representación del proceso de comunicación entre servicios durante el proceso de realizar una petición de amistad 51
Ilustración 35 Representación del proceso de comunicación entre servicios durante el proceso de buscar un usuario 53
Ilustración 36 Diseño relacional de las diferentes tablas que componen la aplicación 58

1 Introducción

El presente proyecto ha sido fruto de la realización de una Beca de Colaboración del Ministerio de Educación, Cultura y Deporte, en el **Biomedical Data Science Laboratory (BDSLab)** de la Universitat Politécnica de València, bajo la dirección de la doctora Sabina Asensio Cuesta y la supervisión del catedrático Juan Miguel García Gómez.

El presente proyecto no ha partido de cero, pues cuando se planteó el proyecto ya existía una versión previa desarrollada por el grupo de investigación BDSLab para la plataforma de mensajería instantánea Telegram.

1.1 Contexto social

1.1.1 El problema de la obesidad y el sobrepeso

Conforme a la Organización Mundial de la Salud (OMS), el sobrepeso y la obesidad son una acumulación anormal o excesiva de grasa que puede ser perjudicial para la salud. Para determinar esta cantidad se utiliza un simple indicador denominado Índice de masa corporal (IMC) que se calcula dividiendo el peso de la persona por su altura elevada al cuadrado. Así pues, se mide en kg/m^2. En el caso de adultos se considera sobrepeso cuando el IMC es igual o superior a los 25 kg/m^2 mientras que la obesidad se produce cuando el IM es igual o superior a los 30 kg/m^2.

La causa fundamental de esta enfermedad es el desequilibrio energético entre las calorías consumidas y las gastadas, pues se consumen alimentos de alto contenido calórico y ricos en grasas y un descenso de la actividad física debido a estilos de vida más sedentarios, el uso de los medios de transporte y el aumento en la urbanización.

El tener un IMC implicado es un importante factor de riesgo de enfermedades no contagiosas como las enfermedades cardiovasculares, diabetes, trastornos del aparato locomotor (en especial la osteoartritis) y algunos cánceres. El riesgo de contraer estas enfermedades es directamente proporcional al IMC, es decir, a mayor IMC, mayores son las probabilidades de contraer alguna de estas enfermedades (OMS, 2015).

1.1.2 Prevalencia de la obesidad y el sobrepeso en la sociedad

Según la OMS, en 2016 más de 1900 millones de adultos tenían sobrepeso, de los cuales 650 millones eran obesos, esto es un 39% y un 13% respectivamente de la población mundial. Entre los años 1975 y 2016 la prevalencia mundial de la obesidad se ha casi triplicado.

En cuanto a los menores, el mismo año se calculó que había 41 millones de niños de menos de 5 años con obesidad o sobrepeso y más de 340 millones de niños y adolescentes (de 5 a 19 años) con sobrepeso u obesidad. (OMS, 2015).

En Europa, más del 50% de la población tiene sobrepeso y el 20%, obesidad. A nivel español puede observarse cómo esta tendencia al alza se mantiene como puede observarse en la figura inferior. Así, se ha pasado del 7,4% en 1987 al 17,4% en 2017, y el sobrepeso afecta ya el 37,1% de la población. Más de la mitad de los adultos (54,5%) tiene exceso de peso ("Wakamola, un bot que te ayuda a conocer mejor tus hábitos nutricionales y de actividad física a través de Telegram – Wakamola", 2019). Esto unido a un gasto asociado de entre el 0'7% y un 2'8% del presupuesto destinado a cada país (Withrow & Alter, 2011), supone que el problema de la obesidad y sus comorbilidades asociadas suponen aproximadamente más de 500 millones de euros al año a la sanidad española (en base al gasto presupuestado en la sanidad española en 2019 (*Principales resultados Estadística de Gasto Sanitario Público*, 2018)).

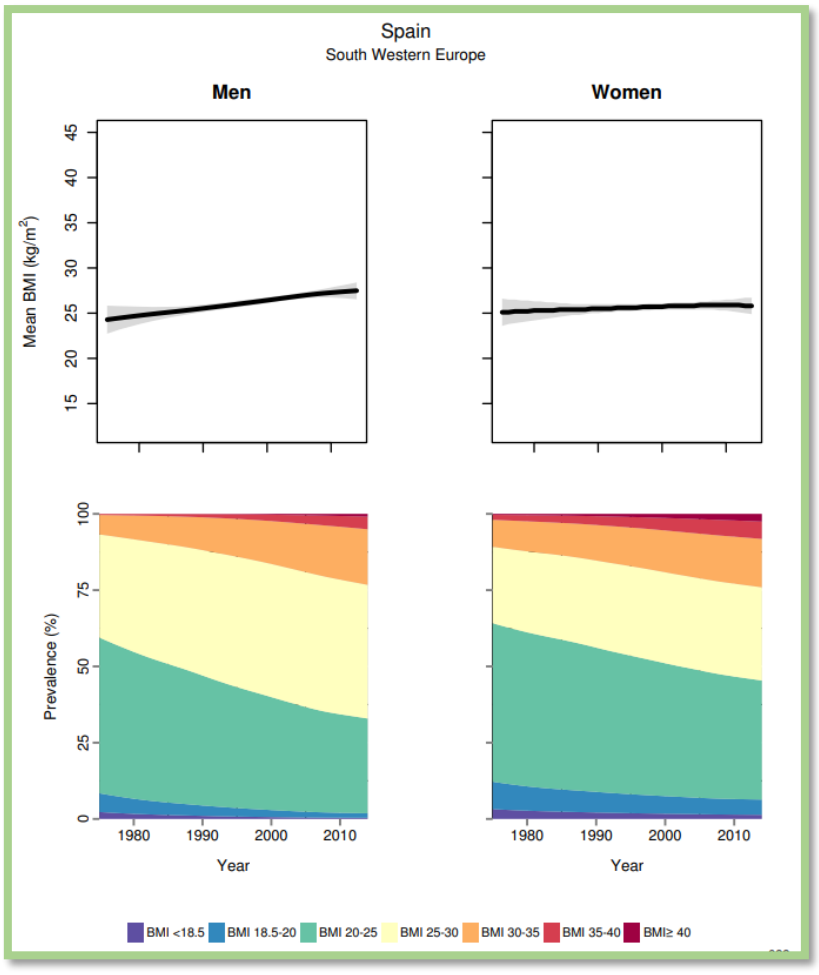

Figura 1 Datos del Índice de Masa Corporal separadas por sexos en España desde 1975 hasta 2014 (Di Cesare et al., 2016)

1.2 Contexto tecnológico

1.2.1 Paradigma de las plataformas de desarrollo de aplicaciones

Actualmente, el desarrollo de aplicaciones está orientado hacia cuatro paradigmas: las páginas web, las Web Apps, las Progressive Apps y las aplicaciones nativas.

- **Páginas web**

Las páginas web sirven para mostrar información, siempre de la misma manera. Al ser estáticas indica que su contenido no se modifica si el usuario hace clic sobre ella o pasa el cursor del ratón sobre alguna parte. No obstante, utiliza hiperenlaces para conectar diferentes sitios web o recursos del mismo sitio. Su código fuente se aloja en un ordenador diferente del usuario, el servidor web. Además, los datos generados se mantienen en dicho servidor. Tienen la ventaja de que son independientes del sistema operativo, ya que se ejecutan sobre un navegador web, que se limita a mostrar el contenido.

- **Web Apps**

Las Web Apps son una mejora de las páginas web, ya que son sitios web dinámicos, es decir tienen elementos que se modifican en función de las acciones del usuario con el sitio, por ejemplo, al pulsar una imagen o un botón, su contenido se modifica, cambiando de apariencia o el contenido mostrado. También utilizan hipervínculos para dirigir hacia otros recursos propios del mismo sitio o hacia recursos externos.

- **Progressive Web Apps**

Las Progressive Web Apps son aplicaciones que se ejecutan en un servidor, pero que tienen acceso a recursos físicos del dispositivo cliente en el que se ejecutan, como la cámara, el micrófono, el GPS, … No obstante, requieren de la instalación de una pequeña cantidad de código en el dispositivo cliente, pero la mayor parte del código es ejecutada en un servidor.

- **Aplicaciones nativas**

Las aplicaciones nativas son códigos desarrollados específicamente para un sistema operativo en concreto. Son monoplataforma, de forma que se han de adaptar el programa desarrollado para diferentes sistemas operativos. No requieren de un navegador para su ejecución ni e acceso a Internet para los recursos. Requieren de su instalación en el dispositivo de ejecución, por lo que consumen recursos del dispositivo (memoria y capacidad de procesamiento).(Aranceta-Bartrina et al., 2005)

A modo de resumen de las características de cada una de estas clases de aplicaciones se ha elaborado la tabla 1.

Tabla 1 Características y diferencias entre las páginas web, Web App, Progressive Web App y las aplicaciones nativas. [6]

Plataforma	Página Web	Web App	Progressive Web App	Aplicación Nativa
Descripción	Son sitios web estáticos	Son sitios web dinámicos que requieren de un mayor desarrollo	Páginas web que se comportan como aplicaciones nativas. Aprovechan las funcionalidades de los dispositivos en que se visualizan	Aplicaciones que se instalan en los dispositivos. Se ubican en las plataformas o markets desde las que los usuarios se las descargan.
Independiente de la plataforma del usuario	Sí	SÍ	Sí	No
Ejecución código	Servidor	Servidor	Servidor y/o dispositivo usuario	Dispositivo usuario
Almacenamiento de los datos	Servidor	Servidor	Servidor y dispositivo	Dispositivo usuario
Utilización de las funciones del dispositivo	No	No (Limitada)	Sí	Sí
Acceso al servicio	URL	URL	URL	Aplicación dispositivo
Instalación	No	No	No	Sí
Actualización	En servidor (no requiere descarga)	En servidor (no requiere descarga)	En servidor (no requiere descarga)	En dispositivo (requiere descarga)
Conexión a Internet	Sí	Sí	No siempre	No siempre

De todas estas opciones, se ha escogido elaborar una **Web App** por ser programadas en unos lenguajes de programación relativamente fáciles de aprender (HTML, CSS y JavaScript) de los cuales existen muchos recursos para aprender a utilizarlos. Además, permiten ser abiertas desde cualquier dispositivo y no requieren de una instalación.

1.2.2 Paradigma del desarrollo de aplicaciones web

En el desarrollo de aplicaciones web, tanto para las Web App como las páginas web, el proyecto se suele dividir en dos partes: el **Frontend** y el **Backend**. De esta forma, se facilita el desarrollo y, además, se pueden concentrar los esfuerzos de los equipos hacia diferentes direcciones, con diferentes problemática, pero manteniendo un objetivo en común ("¿Qué es Backend y Frontend? - Descubre Comunicación", s. f.).

- **Frontend**

El Frontend es la parte visual de la aplicación en la que interactúan los usuarios. Se conoce como el lado del cliente. En esta área se enfocan en el aspecto visual. Así se trabajar en diseñar y añadir en el código la ubicación de la información, la forma en la que se presenta, las imágenes utilizadas, las tipologías de letras y tamaños de letras utilizados, los colores de la temática, los botones, las animaciones, las transiciones entre distintos elementos, la presentación en diferentes dispositivos y tamaños de pantallas (ordenadores, tablets y smartphones), entre otros.

Así pues, esta área se ocupa de la experiencia del usuario, procurando que el sitio sea lo más atractivo, sencillo de utilizar y funcional posible. En este lado de la aplicación es común utilizar los lenguajes de programación: HTML5, CSS3, JavaScript, Jquery y Ajax.("¿Qué es Backend y Frontend? - Descubre Comunicación", s. f.)

- **Backend**

El Backend se trata del código que se ejecuta en el lado del servidor. En este lado se encargan de recibir los datos, del lado del cliente, procesarlos y elaborar una respuesta o reaccionar de forma apropiada a estos. Consta del código de la lógica de la aplicación además de una base de datos para almacenar la información generada.

Los principales lenguajes que se utilizan en este entorno son para la parte de la lógica de la aplicación: ASP.NET, PHP, Python, Ruby, Node.js, Java, mientras que para la gestión de la base de datos suele utilizarse un programas como MySQL, SQL Server, PostgreSQL, Oracle o MongoDB.("¿Qué es Backend y Frontend? - Descubre Comunicación", s. f.)

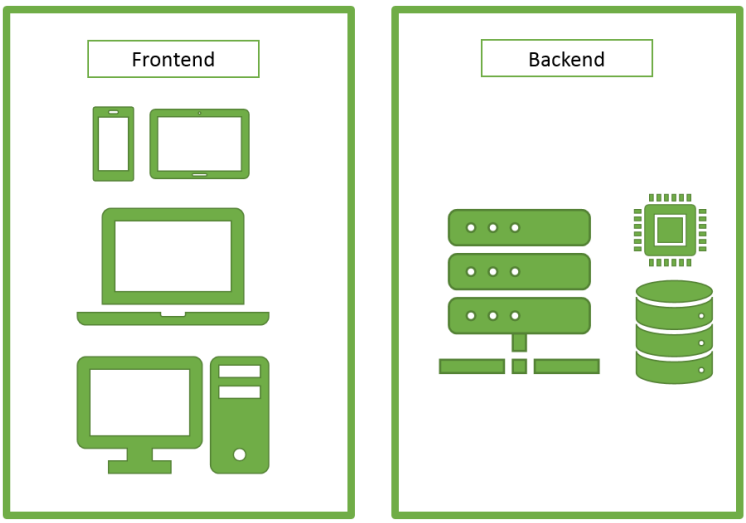

Ilustración 1 Representación del Backend y del Frontend

1.2.3 Comunicación entre máquinas en una red

La comunicación entre máquinas dentro en una red local forma es uno de los pilares tecnológicos de la sociedad actual. Sin ella no tendríamos grandes avances como Internet. Para que se establezca esta comunicación es necesario el uso de estándares por parte de todos los dispositivos implicados. Es entonces donde entran los estándares internacionales y protocolos, de los cuales existen protocolos para enviar y recibir datos, como HTTP o los sockets; estándares para la escucha de peticiones de recursos y su correcta repuesta, como las API REST, e incluso estándares para la codificación de mensajes intercambiados entre procesos como JSON, que admite el intercambio de mensajes entre destinos lenguajes de programación.

- **HTTP**

El *Hipertext Transfer Protocol* (más conocido por sus siglas HTTP) se trata de un protocolo de la capa de aplicación de estructura cliente-servidor diseñado a principios de la década de 1990. Es un protocolo utilizado para la petición de datos y recursos, como documentos HTML, sobre el protocolo TCP, encriptado TLS cualquier otro protocolo fiable. Se trata de la base para el intercambio de datos en la Web.

Su funcionamiento consiste en que el cliente realiza la petición de datos, normalmente con un navegador web, a un servidor, que le responderá si efectivamente tiene ese recurso un conjunto de ficheros que están asociados a ese recurso. El navegador mostrará la unión de los subdocumentos recibidos como respuesta, como un fichero HTML, un fichero de estilo CSS, vídeos, imágenes, scripts, …formando así la página web completa. Tanto las peticiones como las respuestas se realizan intercambiando mensajes individuales.

Las peticiones HTTP tienen una estructura formada por: un método HTTP, la dirección URL del recurso, la versión del protocolo HTTP, las cabeceras HTTP (opcionalmente) y el cuerpo del mensaje., como puede observarse en la ilustración 2.

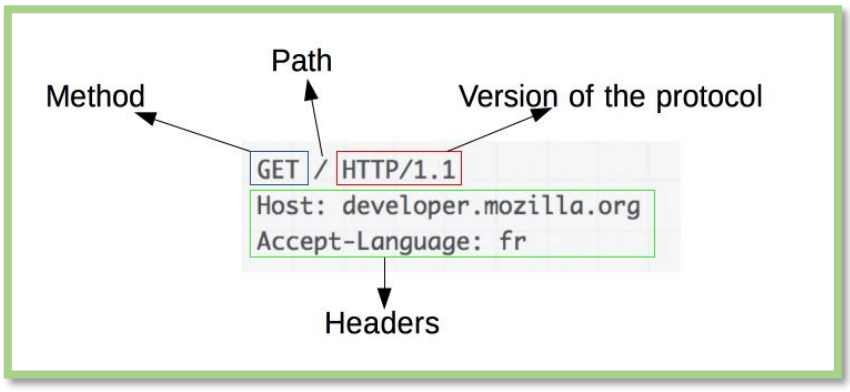

Ilustración 2 Ejemplo de una petición HTTP (“Generalidades del protocolo HTTP - HTTP | MDN”, s. f.)

El método HTTP suele ser un verbo como GET, POST, PUT o DELETE o un nombre como OPTIONS o HEAD que define la operación que el cliente quiere realizar. Los más destacados son GET, que se utiliza para obtener un recurso del servidor, y POST, que se utiliza para enviar datos al servidor.

La dirección del recurso consta de la dirección URL del recurso, sin el campo "*http://",* estando así formada por el dominio junto con el número de puerto TCP.

Las cabeceras HTTP pueden aportar información adicional a los servidores sobre el cliente, como el lenguaje aceptado.

El cuerpo del mensaje es donde, si se trata de una petición POST, se incluyen los datos que debe de utilizar el servidor.

Las respuestas a las peticiones tienen una estructura similar a las peticiones. Constan de la versión del protocolo HTTP que se está utilizando, un código de estado (indicando si la petición ha sido exitosa o no), un mensaje de estado indicando brevemente el código de estado, unas cabeceras HTTP y, si se ha solicitado, el recurso que se ha solicitado. En la ilustración 3 puede observarse un ejemplo de una respuesta HTTP ("Generalidades del protocolo HTTP - HTTP | MDN", s. f.).

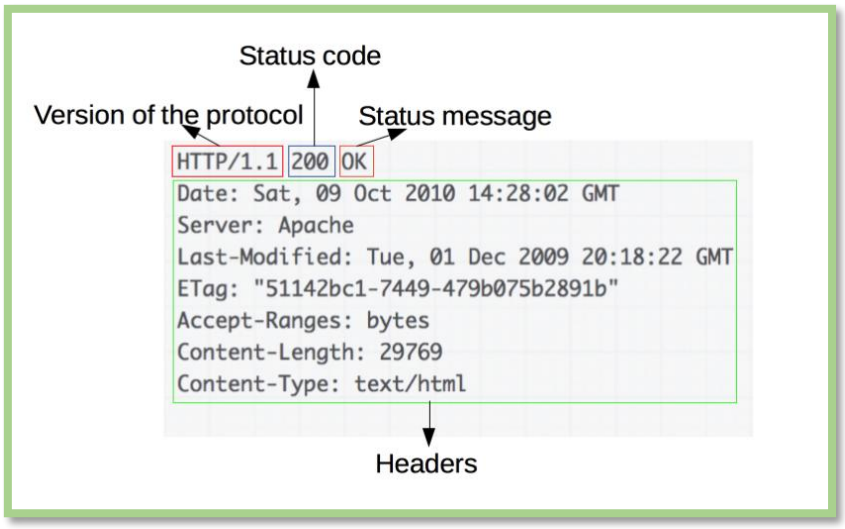

Ilustración 3 Ejemplo de una respuesta HTTP ("Generalidades del protocolo HTTP - HTTP | MDN", s. f.)

- **Sockets**

Un socket es una interfaz para enviar y recibir datos en un puerto específico estableciendo una conexión entre la misma o diferentes máquinas. Se trata de un tipo de software que actúa estableciendo un canal de comunicación de red bidireccional entre un servidor y el programa receptor del cliente. Desde las aplicaciones únicamente se tiene que llamar a una interfaz de programación de aplicaciones (API) para enviar los datos al proceso corriendo en otra máquina a través de la red. De esta forma, los sockets se simplifican el transporte de mensajes a través de la red.

Los sockets funcionan siguiendo una estructura de cliente servidor. EL servidor está esperando a que un cliente le haga una solicitud de conexión. Una vez se haya establecido la conexión (normalmente TCP) se procederá al envío como respuesta por parte del servidor del recurso solicitado por el cliente. No obstante, tanto el cliente como el servidor pueden leer o escribir en sockets que están conectados a un canal de comunicación específicos.(“¿Qué es un socket?”, s. f.) Este proceso se ha esquematizado en la ilustración 4.

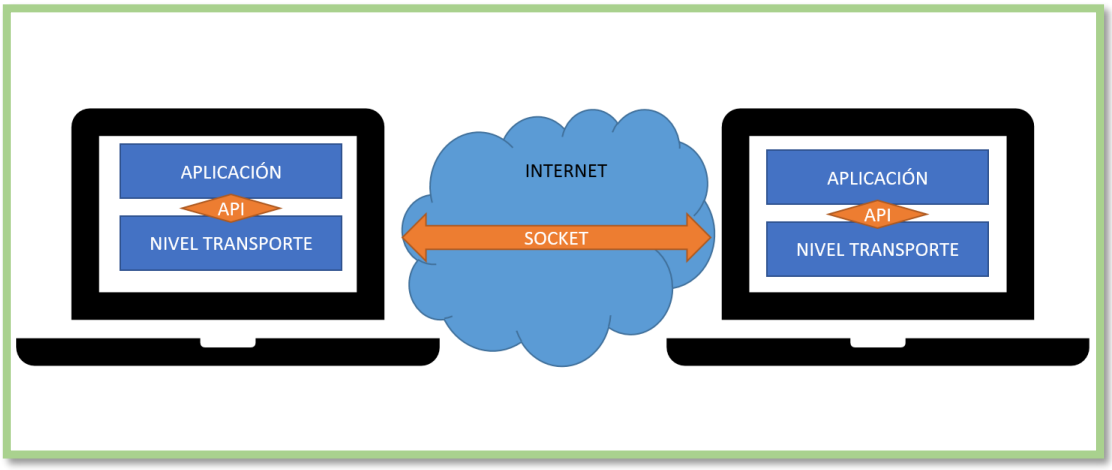

Ilustración 4 Representación del funcionamiento de un socket. Fuente: Elaboración propia

- **API REST**

Se trata de un protocolo de cliente-servidor sin estado, basado en el protocolo HTTP. El formato de intercambio de datos es normalmente JSON o XML. REST significa Representational State Transfer. Las API REST se utilizan para ofrecer datos a aplicaciones que se ejecutan en dispositivos móviles, para ofrecer datos a otros desarrolladores de nuestra aplicación en un formato ordenado que puedan ser consumidos por otras aplicaciones o sitios web.(“Arquitectura de una API REST · Desarrollo de aplicaciones web”, s. f.)

Las operaciones más importantes que se pueden realizar en la comunicación entre ambas máquinas son las peticiones: GET (obtener datos), POST (introducir datos), PUT (editar los datos ya introducidos) y DELETE (eliminar datos)(“API REST: qué es y cuáles son sus ventajas en el desarrollo de proyectos”, s. f.).

Una llamada a la API REST consta de una URL a un recurso, un método basado en HTTP (como GET; POST, PUT o DELETE) y un resultado de la operación que se enviará como respuesta. Esta podrá ser (200 = acción realizada correctamente, 400 = petición incorrecta o 500 error del lado del servidor). Estos códigos están estandarizados para las aplicaciones.(“API REST: qué es y cuáles son sus ventajas en el desarrollo de proyectos”, s. f.)

- **JSON**

JSON (JavaScript Object Notation) es un formato estándar para el intercambio de datos entre máquinas dentro de una red, de la misma forma que el estándar XML. JSON está basado en el lenguaje de programación JavaScript y en él los datos están siempre contenidos entre llaves y para caracterizar un seguir la estructura de pares nombre / valor. Este par está separado siempre dos puntos (“JSON”, s. f.). UN ejemplo de JSON puede observarse en la ilustración 5.

Ilustración 5 Ejemplo de la notación de JSON

Con esta notación puede recrearse el envío de objetos entre máquinas. Pues puede accederse a sus diferentes propiedades. Además, JSON es interpretable por diferentes lenguajes de programación, por lo que varios servicios corriendo en lenguajes diferentes que reciban el mismo JSON serán capaces de extraer sus variables.

1.2.4 Chatbots orientados al ámbito de la nutrición y de la actividad física

Los bots son programas informáticos que simulan ser personas, interactúan con los usuarios o con otras aplicaciones y cumplen con un objetivo específico por medio de algoritmos. Los **chatbots** son bots programados para entablar una conversación simulada con los usuarios que simula la interacción humana, pero siendo innecesaria su participación (Martínez Molera, s. f.).

Se utilizan chatbots en distintos sectores como los negocios, el entretenimiento y el servicio al cliente, donde las redes sociales y las plataformas de mensajería son relevantes. Existen incluso chatbots orientados al ámbito de la salud ("Chatbot as a great addition to fitness apps — Jasoren", s. f.), donde cumplen funciones, como:

- Chatbots para el entrenamiento físico
- Chatbots orientados a la nutrición
- Chatbots como ayuda y guía en la meditación
- Chatbots para la monitorización emocional
- Chatbots para la monitorización de estados de salud especiales (como chequeo de síntomas, soporte a la diabetes, …)

Se utilizan chatbots en vez de las soluciones convencionales porque admiten una navegación más sencilla que en las aplicaciones comunes. Además, permiten personalizar las respuestas en función del usuario, siendo posible configurarlos para que llamen al usuario por su nombre. De esta forma, hacen que el usuario se sienta un trato cercano, por lo que se incentiva el uso de la aplicación.

El único problema es que requieren un constante mantenimiento y mejora de sus interacciones con el usuario con tal de lograr la mejor experiencia para el usuario.

1.2.5 Aplicaciones en el mercado

En el mercado se han encontrado múltiples aplicaciones que utilizan de chatbots para recabar información sobre el estado de salud física y mental de sus usuarios. Algunos de los ejemplos encontrados son:

- **Chatbot de la OMS sobre el COVID-19**

Es un chatbot habilitado para WhatsApp por la empresa Prekelt, una ONG sudafricana. Se creó este servicio a nivel local para asegurar que le público recibiera la mejor información posible sobre el brote de COVID-19, pero ha acabado desplegándose a nivel global. Está desarrollada en tres idiomas actualmente: inglés, francés y español. Sus funciones son aportar datos actualizados y oficiales sobre la pandemia, consejos para protegerse, respuestas a preguntas frecuentes, entre otras funciones. (Miranda, 2020)

Ilustración 6 Ejemplo de la experiencia de usuario del chatbot sobre el COVID-19 de la OMS

- **Lark**

Chatbot creado por la empresa *Lark Technologies, Incorporated* que ofrece distintas soluciones personalizadas para el usuario para el cuidado y la prevención de la diabetes, el cuidado de la hipertensión y/o fomentar los comportamientos saludables como manejo del estrés y de la ansiedad, el dejar de fumar o la pérdida de peso.(“Lark Programs | A Whole Person Approach to Care | Lark Health”, s. f.)

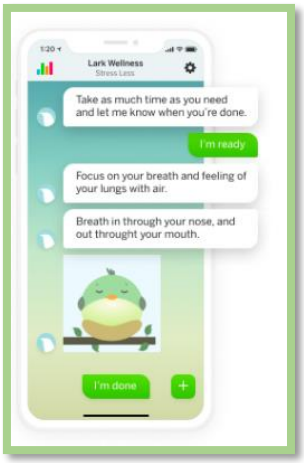

Ilustración 7 Visualización del usuario de la aplicación Lark

- **Makerobos**

Makerobos es una compañía que ofrece chatbots personalizados al cliente. Uno de sus productos es el *"Diet Planner Chatbot"* que ayuda a crear dietas basadas en los gustos de los clientes, sus presupuestos y su horario. Ofrece objetivos dietéticos y nutricionales a partir de su calculadora calórica. ("Diet Planner Chatbot - Applications, Benefits and Bot Template | Makerobos.com", s. f.)

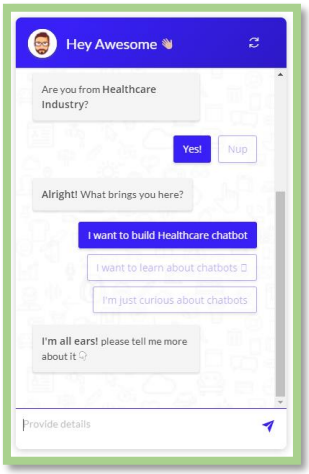

Ilustración 8 Ejemplo de experiencia del usuario del chatbot "Diet Planner"

- **Jolt.ai**

Jolt.ai es un chatbot diseñado para *Facebook Messenger* para lograr los objetivos de actividad física y de salud de sus usuarios. Tiene la capacidad de adaptar su discurso en diferentes tonos, desde animado a más calmado. Puede enviar *gifs*, *emojis* y utilizar el lenguaje natural. Puede integrarse con el *Apple Watch* o *Fitbit*. Monitoriza la actividad con chequeos y pone retos semanales. Empareja usuarios con perfiles similares para realizar una competición entre ellos para ganar más puntos al final de la semana. Los puntos conseguidos en la aplicación son canjeables por *Apple Watch*, *AirPods* o tarjetas de regalo.("Jolt.ai - The best fitness accountability partner you've ever had | Product Hunt", s. f.)

Ilustración 9 Visualización de la aplicación Jolt.ai

1.3 Contexto del proyecto

1.3.1 El proyecto Wakamola

El proyecto Wakamola surgió bajo el marco del proyecto europeo CrowdHealth, en el grupo BDSLab del instituto ITACA de la Universitat Politècnica de València (UPV). El objetivo era dar a conocer los hábitos nutricionales y de actividad física de la población. En un primer momento se diseñó e implementó como un bot de Telegram que simulaba mantener una conversación con el usuario sobre su dieta actividad física, enfermedades, edad, peso, red social, etc.(Asensio-Cuesta et al., 2021)

La información obtenida se utilizaba para crear una red de relaciones que permitiera estudiar las interacciones de los hábitos nutricionales y físicos junto con su condición física además de los de su entorno, como su red familiar, laboral y amistades

El nombre de Wakamola proviene de una encuesta que se realizó a casi 500 personas con 52 diseños propuestos por alumnos de la ETSID. Tras esta se escogieron tanto el nombre como el diseño y la temática de la aplicación ("Wakamola, un bot que te ayuda a conocer mejor tus hábitos nutricionales y de actividad física a través de Telegram – Wakamola", 2019).

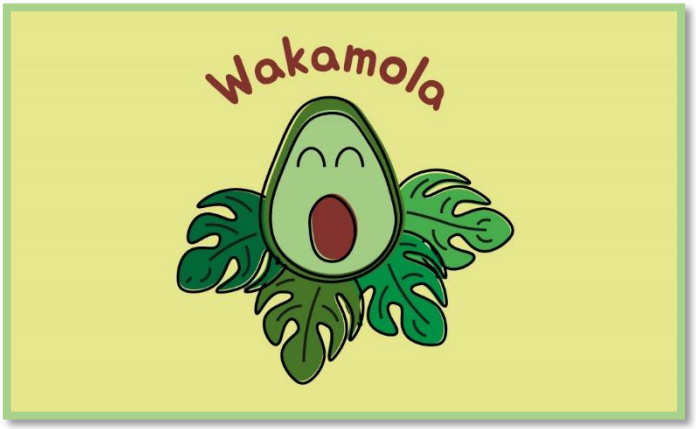

Ilustración 10 Icono de Wakamola

Wakamola permite determinar la importancia que tiene el entorno sobre cada usuario, estudiando factores como la ubicación de la vivienda y sus relaciones interpersonales (familia, amigo, trabajo) sobre el índice de masa corporal (IMC) ("Wakamola, un bot que te ayuda a conocer mejor tus hábitos nutricionales y de actividad física a través de Telegram – Wakamola", 2019), la medida utilizada por la OMS para determinar la obesidad (OMS, 2015). El análisis de datos recogidos por Wakamola servirá para elaborar actuaciones ("Wakamola, un bot que te ayuda a conocer mejor tus hábitos nutricionales y de actividad física a través de Telegram – Wakamola", 2019) con la intención de solucionar el problema del sobrepeso y obesidad que afronta la sociedad actual (Di Cesare et al., 2016).

1.3.2 Wakamola Telegram

La herramienta Wakamola funciona mediante el establecimiento de una conversación del usuario con el bot a través de la aplicación de Telegram, realizando una serie de preguntas que van que van desde la edad o el peso hasta el tipo de alimentación o veces a la semana que se hace ejercicio. Se trata de una aplicación desarrollada en los idiomas castellano, valenciano e inglés que, en función de las respuestas del usuario, se le asigna una puntuación (Wakaestado) entre 0 y 100 puntos para cada uno de los campos: Actividad física, dieta e índice de masa corporal (IMC). También se valora la presencia de enfermedades en el usuario y el Wakaestado de sus contactos en la aplicación.

Para la asignación correcta de las puntuaciones fue necesario el asesoramiento de Ana Frígola de la Facultad de Farmacia de la Universitat de València, así como de los doctores del Hospital Universitari i Politècnic La Fe: Juan Francisco Merino-Torres, Matilde Rubio Almanza, Salvador Tortajada y Ruth Vilar-Mateo.

Desde el 28 de enero de 2019 hasta finales de febrero de ese mismo año se realizó un estudio piloto de la aplicación para comprobar la viabilidad del proyecto. Este se realizó un estudio piloto en la localidad valenciana de Tavernes de la Valldigna, gracias a la colaboración del ayuntamiento de la ciudad (“Wakamola, un bot que te ayuda a conocer mejor tus hábitos nutricionales y de actividad física a través de Telegram – Wakamola”, 2019).

1.3.3 Análisis del funcionamiento de Wakamola Telegram

Para iniciar la conversación con Wakamola se tiene que haber instalado la aplicación de Telegram, en el dispositivo del usuario, ya sea un ordenador, una tablet o un smartphone. Una vez accedido a ella, se tiene que buscar el bot "WakamolaUPV", donde se muestra un mensaje aclarando la finalidad del bot y un botón de inicio. Al pulsarlo, empieza a ejecutarse el bot mostrando el menú de inicio del bot que consta de una imagen principal de bienvenida junto con una matriz (array) de botones con las diferentes opciones que puede ejecutar el usuario.

- **Personal:** Inicia la encuesta sobre los datos personales del usuario, como la edad, el peso, el estado civil y enfermedades relacionadas con al obesidad
- **Dieta:** inicia la conversación sobre la dieta del usuario
- **Actividad:** inicia la conversación sobre el nivel de actividad física del usuario
- **Wakaestado.** Únicamente funciona cuando el usuario ha completado las tres encuestas. AL pulsarlo y cumplir este requisito muestra las puntuaciones descompuestas que ha obtenido el usuario y en comparación con sus contactos. Además, ofrece una puntuación global sobre 100 llamada "Wakaestado" que da nombre a nombre botón. Es un indicador del grado de salud del usuario, así como de su implicación con la aplicación, pues se valora la cantidad de contactos que ha hecho participaren el proyecto.
- **Wakarred:** muestra enlace a otra dirección red donde se muestra un grafo de las diferentes poblaciones que se han formado al utilizar los usuarios la aplicación. Permite observar la puntuación individual y compararse con el resto de los usuarios (los otros nodos).
- **Acerca de Wakamola:** muestra un mensaje detallando los diferentes participantes del proyecto.
- **Compartir:** muestra un enlace que al enviarlo a otro usuario de Telegram le redirige a la aplicación de Wakamola, con lo que puede hacer la encuesta. Además, estos usuarios quedan conectados dentro de la aplicación, de forma que se empieza a formar una nueva población dentro de la "Wakarred".

Una vez se hace clic en cada una de las opciones muestra una imagen personalizada del campo que ha seleccionado el usuario, además de un mensaje indicando que al pulsar sobre los mensajes pueden editarse las respuestas.

Ilustración 11 Captura de pantalla del inicio del uso de la aplicación Wakamola

Así pues, cuando se pulsa en “Personal” aparece la ilustración 12 y a continuación las preguntas personales que forman un total de 16 preguntas generales que son:

1. *¿Cuánto pesas? (en kg)*
2. *¿Cuánto mides? (en cm)*
3. *En promedio, ¿Cuántas horas duermes?*
4. *En promedio, ¿Cuántos cigarrillos consumes diariamente? (cero si no eres fumador)*
5. *¿Has recibido alguna vez un diagnóstico de hipertensión o tomas medicación por eso?*
6. *¿Has recibido alguna vez un diagnóstico de diabetes o tomas medicación por eso?*
7. *¿Has recibido alguna vez un diagnóstico de colesterol elevado (hipercolesterolemia) o tomas medicación por eso?*
8. *¿Has recibido alguna vez un diagnóstico de enfermedad cardiovascular o tomas medicación por eso?*
9. *¿Con qué género te identificas?*
10. *¿Cuántos años tienes?*
11. *¿Cuál es tu nivel de estudios?*
12. *¿Cuál es tu estado civil?*
13. *¿Cuántos sois en casa?*
14. *¿Eres estudiante?*
15. *¿Eres trabajador?*
16. *¿Puedes decirme tu código postal?*

Ilustración 12 Captura de pantalla de la encuesta personal de la aplicación Wakamola Telegram

Como puede observarse en la ilustración 12, en las preguntas sobre el género (pregunta 9), el nivel de estudios (pregunta 11) y el estado civil (pregunta 12) aparece un teclado en la conversación del bot con las diferentes respuestas admitidas como correctas.

Se distinguen en este caso tres clases de respuestas: respuestas numéricas, donde se ha de introducir un valor numérico, ya sea entero o decimal, una respuesta de carácter booleano (sí o no), una respuesta de opciones delimitadas. En caso de introducir una respuesta de una clase diferente de la admitida, la aplicación lanza un mensaje de advertencia y repite al usuario la

pregunta hasta que introduce una respuesta válida. Además, muestra un mensaje aclaratorio indicando cuál es el tipo de respuesta que debe introducir.

Ilustración 13 Captura de pantalla de diferentes matrices de botones de la aplicación Wakamola

Cuando se acaba la encuesta muestra un mensaje de agradecimiento y una sugerencia de con qué parte de la aplicación seguir continuando, introduciendo datos, como puede visualizarse en la ilustración 13.

Ilustración 14 Captura de pantalla de la encuesta de dieta de la aplicación Wakamola Telegram

En la parte de dieta las preguntas consisten en responder cuántas veces al día o a la semana se consumen una serie de alimentos. El total de los alimentos de los cales se pregunta asciende a los 51 alimentos.

Ilustración 15 Captura de pantalla de la encuesta de actividad física de la aplicación Wakamola Telegram

Por último, se pueden responder las preguntas de la parte de la actividad física, correspondientes a la ilustración 15. Son:

1. *Durante los últimos 7 días ¿Cuántos días hiciste actividades físicas vigorosas como levantar objetos pesados, excavar, aeróbicos, o pedalear rápido en bicicleta; durante más de 10 minutos?*
2. *¿Cuántos minutos dedicaste en uno de esos días a esas actividades vigorosas?*
3. *Durante los últimos 7 días ¿Cuántos días hiciste actividades físicas moderada durante más de 10 minutos como como cargar objetos livianos, pedalear en bicicleta a paso regular, o jugar dobles de tenis?*
4. *¿Cuántos minutos dedicaste en uno de esos días a esas actividades moderadas?*
5. *Durante los últimos 7 días ¿Cuántos días caminaste en el trabajo, en traslados, o como recreación al menos 10 minutos continuos?*
6. *En uno de esos días normalmente ¿Cuántos minutos caminas?*
7. *Durante los últimos 7 días ¿Cuántas horas en total sueles estar sentad@?*

Ilustración 16 Captura de pantalla del Wakaestado de la aplicación Wakamola Telegram

Con las respuestas a las preguntas anteriores se obtiene el Wakaestado, que pondera el IMC, la alimentación del usuario y su actividad, como puede observarse en la ilustración 16. Asigna un valor del 1 al 100 a la nutrición del usuario, a su actividad física y a su IMC. La media de estos tres valores da lo que se conoce como el Wakaestado del usuario. Además, muestra cómo está el usuario en función de los usuarios con los que ha establecido una relación.

Ilustración 17 Captura de pantalla de la opción de compartir de la aplicación Wakamola Telegram

Esta relación entre usuarios se mantiene entre los usuarios pulsando en compartir y enviando el enlace dentro de la plataforma Telegram, como puede verse en la ilustración 17. Al abrir el enlace otro usuario, estos quedan conectados en la Wakarred.

Ilustración 18 Captura de pantalla de la Wakarred de la aplicación Wakamola Telegram

Al pulsar sobre la entrada Wakarred, se muestra un enlace, como en la ilustración 18, que al pulsarlo redirige al usuario a otra página web, donde se muestran diferentes nodos, donde cada uno de ellos representa el Wakaestado de un usuario, como en la ilustración 19. Los nodos conectados son relaciones entre usuarios, de forma que pueden observarse diferentes comunidades.

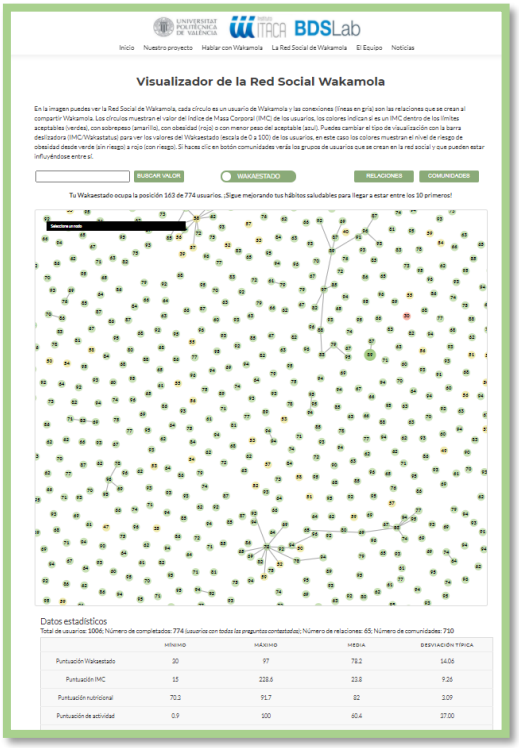

Ilustración 19 Captura de la página web de la Wakarred de la aplicación de Wakamola Telegram

2 Objetivos del proyecto

El objetivo de este Trabajo Final de Máster es desarrollar una plataforma web para recopilar datos sobre la dieta, actividad física, peso y que creen una red social que permita estudiar las causas de la obesidad en una población de forma individual y social. Para ello se parte de la aplicación Wakamola preexistente en Telegram que sirve como punto de partida y referencia para el desarrollo de la aplicación web "Wakamola web".

Los objetivos propuestos han sido:

- Crear una plataforma de visualización de Wakamola de mayor alcance que la plataforma Telegram.
- Mantener la estética y la temática de Wakamola web.
- Recopilar los datos introducidos por los usuarios.
- Almacenar los datos recogidos permanentemente en una base de datos.
- Garantizar la calidad de los datos obtenidos.
- Obtener el Wakaestado correctamente (una puntuación ponderada de la alimentación, la actividad física y el estado de salud física general del usuario).
- Crear una vía para que los usuarios puedan establecer vínculos entre sí.
- Asegurar el funcionamiento de la aplicación en el entorno de producción.
- Obtener un producto escalable y capaz de conectarse a otros entornos de visualización.
- Evaluar la experiencia de los usuarios al utilizar la aplicación.

3 Análisis del problema

Wakamola es una aplicación de chatbot ubicada en Telegram cuya finalidad es recopilar información sobre el estado físico y la alimentación de la población. La problemática que se encontraban es Telegram, la propia plataforma donde se halla ubicada Wakamola. Telegram se trata de una red social multiplataforma de baja implantación en la sociedad española, siendo superada en uso ampliamente por la aplicación WhatsApp. Como puede observarse en la figura 2, en una encuesta realizada en el 2020, se encontró que un 85% de los encuestados utilizaba WhatsApp, mientras que Telegram únicamente era utilizada por un 25% de los encuestados.

Así, para lograr alcanzar un mayor tamaño de población y poder de esta forma conseguir una mayor cantidad de datos se requiere de ampliar las plataformas en las que Wakamola pueda ponerse en funcionamiento.

No obstante, las poblaciones de estudio vienen sesgadas por el acceso a las nuevas tecnologías. Así, es poco probable que la aplicación sea utilizada por usuarios de edades muy avanzadas, debido a la propia brecha tecnológica. Por lo tanto, el usuario al que va destinado la población ya está habituado a las páginas web. Está habituado a utilizar páginas web. Aun así, se ha decidido mantener el diseño lo más sencillo posible pensando además en que el usuario pueda introducir nuevamente los datos para mantener a la aplicación actualizada de su situación personal.

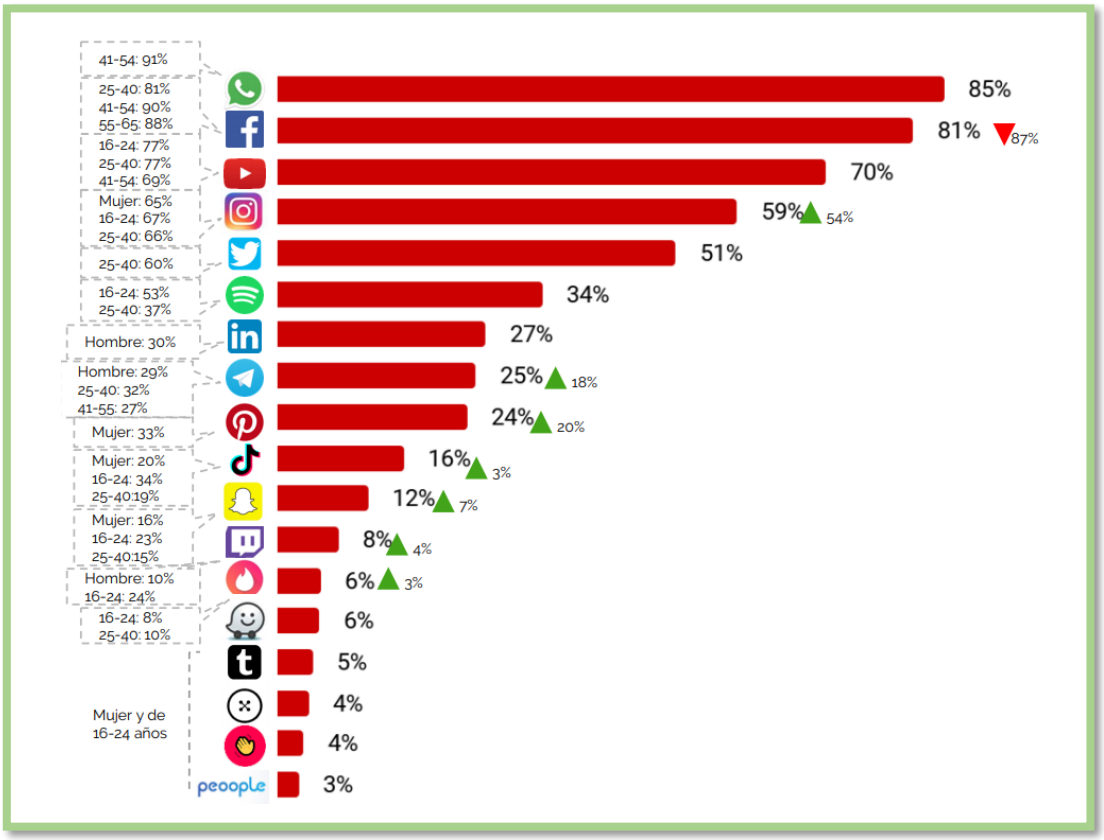

Figura 2 Gráfica del porcentaje de uso de cada red social en la sociedad española en 2020 (IAB Spain, 2020)

3.1 Estudio de mercado de las posibles plataformas en las que implantar Wakamola

Se estudió la viabilidad de la implantación de la herramienta Wakamola en diferentes redes sociales ampliamente implantadas en la sociedad española, además de considerarse otras redes ampliamente utilizadas en otros lugares del planeta. Entre estas opciones se sopesó:

- **WhatsApp:** como principal red social implantada en prácticamente todos los estratos de la sociedad española, sin importar género ni edad. El problema surge que para la elaboración de chatbots se necesita tener acceso a la API de dicha aplicación y en el caso de WhatsApp esta es de uso privado. Además, si se quiere hacer uso de ella se debe contactar con empresas que WhatsApp certifica y pagarles por la concesión del acceso a esta API. Dado el carácter de baja financiación de este proyecto, se descartó esta opción a pesar de su potencial capacidad de alcanzar a la mayor parte de la sociedad. También existen opciones menos lícitas para lograr este objetivo, pero también se descartaron dadas las características del proyecto y la política de WhatsApp con respecto a los bots.
- **Facebook Messenger:** es una de las principales plataformas en las que implantar la aplicación de Wakamola, al ser la principal red social en EEUU, además de ampliamente utilizada en España, sobre todo, por grupos de edad mayores de 30 años. Al contrario que en WhatsApp, su API es de uso público, por lo que se pueden implementar chat bots.
- **Página web:** se consideró la opción de la creación de una página web propia que accediera a la base de datos y que implementara lo mismo que se tenía en Telegram. Concretamente, como se trata de un chat, la página web debería de ser interactiva y reaccionar a las acciones del usuario, no simplemente limitarse a mostrar información siempre de la misma forma. Por ende, debería de tratarse de una página web dinámica o Web App.

3.2 Estudio de la viabilidad de la implantación en las diferentes plataformas

Previo a la toma de decisión sobre la plataforma en la que implantar la solución, se realizó un estudio preliminar de la viabilidad de la implantación de la solución final utilizando diferentes softwares ya desarrollados. Se investigó sobre la posibilidad de tanto añadir el bot a una **red social** de mayor implantación en la sociedad, como en una **página web** ya creada, con diferentes softwares comerciales que se hayan en el mercado.

3.2.1 Insertar un chatbot en redes sociales

A la hora de insertar en una de las redes sociales se pretendía utilizar una estructura de software ya desarrollada para poder cumplir con los objetivos del proyecto. Dada la gran extensión en cuanto número de usuarios, las investigaciones iniciales partieron hacia encontrar una solución ya existente para alguna de las principales redes sociales. Los principales objetivos, dado el tamaño de su población de usuarios fueron: **WhatsApp, Facebook, Instagram** y **Line.**

3.2.1.1 WhatsApp

Se descubrió que la aplicación de WhatsApp no cuenta con una forma nativa de alojar chatbots, así como sucede con Telegram. Así pues, se requiere de un proceso adicional para el desarrollo de un chatbot para esta red social utilizando un paquete SDK (Kit de Desarrollo de Software) para bots de chat, siendo necesario así tener conocimientos de programación, o bien, la utilización de programas comerciales como **AutoResponder para** WA (Hubspot, 2019) o la propia aplicación de **WhatsApp Business.**

AutoResponder para WA – Respuesta automática permite generar respuestas personalizables a las peticiones de los usuarios. De la misma manera que el bot de Telegram programado para Wakamola, esta App permite la utilización de emoticonos, pero no puede enviar imágenes ni stickers. Existen dos versiones: una gratuita y otra de pago con más funcionalidades. No obstante, únicamente permite contestar a los mensajes, pero no captar las respuestas para guardarlas en una base de datos.

Ilustración 20 Página de la aplicación de AutoResponder para WA

En caso de utilizar esta aplicación se recomienda no utilizar un número propio para elaborar el chatbot de WhatsApp ya que va a perderse. Otro problema con esta aplicación es que WhatsApp tiene algoritmos internos que detectan los mensajes que son spam. Dado que nuestro bot probablemente vaya a responder siempre de la misma manera es probable que sea baneado (Planeta Chatbot, 2017).

WhatsApp Business es la aplicación de WhatsApp con una API para obtener los datos de esta Esta aplicación no estaba cuando se realizó la investigación de la plataforma de aplicación, de forma que resultó imposible encontrar información sobre cómo hacer un bot en esta plataforma (WhatsApp Inc., 2020).

También existen empresas especializadas en crear chatbots para esta plataforma. Así, son capaces de acceder a la API de WhatsApp, ya que no existe un modo directo para acceder como si ocurre con las APIs de otras apps de mensajería como Facebook o Telegram. LE coste de implementación con una de estas compañías es de 2000€ de coste fijo de implementación junto con unos costes variables mensuales de 300€ a 2000€ al mes, de pendiendo de la cantidad de mensajes recibidos (“Agencia de Chatbots | WhatsApp | Facebook Messenger | Instagram | Alexa Skills”, s. f.). Por lo tanto, dada la falta de presupuesto para este proyecto esta opción no resulta viable.

3.2.1.2 Facebook

En cuanto a Facebook, es posible crear chatbots para esta plataforma utilizando la propia API de la aplicación. (Apcelent, 2018) Así pues, existe la opción de externalizar el proceso de la creación del bot. El precio de dicho proyecto costaría 9 $ / mes con la empresa **OctaneAI** (Octane AI, 2020) o Mobile **Monkey**, (“Chat Marketing Platform & Chatbot Pricing”, s. f.), que ofrecen soluciones para chatbots en Facebook.

No obstante, existen herramientas software que permiten crear estos chatbots de forma gratuita como **Chatfuel**. (“Tutorial: cómo construir un chatbot en Facebook Messenger | by Dimitry Kagan | Planeta Chatbot : todo sobre los Chat bots, Voice apps e Inteligencia Artificial”, s. f.) Es una herramienta freemium que permite el envío gratuito de hasta 100.000 mensajes al mes, muy superior a otras herramientas que se hayan en el mercado. No obstante, la mejora cuesta 15$ al mes, por lo que el proyecto se vuelve poco escalable dada la falta de presupuesto.

3.2.1.3 Instagram

En cuanto a Instagram, no existe una API oficial que permita la implementación fácil de chatbots (“Cómo diseñar un chatbot en Instagram | by Nutella Developer | Planeta Chatbot : todo sobre los Chat bots, Voice apps e Inteligencia Artificial”, s. f.), de la misma manera que sucedía con WhatsApp. No obstante, existen otras formas no oficiales de crear un bot para esta red social, simulando mediante código ser un usuario (“How to Build an Instagram Chatbot (using Machine Learning) - Markets and Data - YouTube”, s. f.). Se descarta esta opción, pues no se trata de una forma oficial para desarrollar un chatbot.

3.2.1.4 Line

La aplicación de Line cuenta con sus APIs abiertas para que los desarrolladores puedan elaborar sus propios chatbots. (“LINE se une a la moda de los bots abriendo su API para desarrolladores”, s. f.) No obstante, su acceso solo es gratuito durante un periodo de tiempo limitado. A pesar de que existen bots creados en Python para esta aplicación (“GitHub - line/line-bot-sdk-python:

LINE Messaging API SDK for Python", s. f.), persiste el problema con Telegram de la falta de usuarios en España, por lo que se descarta opción.

3.2.2 Insertar un bot en un sitio web

Otra posibilidad que se exploró fue la opción de desarrollar un bot con un software externo y añadirlo a una página web de diseño propio. Para ello se investigaron diferentes softwares comerciales disponibles online, con la idea de que fueran gratuitos y fueran *open source*, además de que permitieran recoger datos e introducirlos en una base de datos. Entre los programas explorados aparecen Xenioo, Azure y Snatchbot.

3.2.2.1 Xenioo

Para insertar un bot en un sitio web se probó la opción de utilizar el software Xenioo una herramienta para crear visualmente la lógica del bot y que admite su incorporación en múltiples plataformas, incluida WhatsApp ("Diseño de chatbots en páginas web | by Planeta Chatbot | Planeta Chatbot : todo sobre los Chat bots, Voice apps e Inteligencia Artificial", s. f.).

Xenioo tiene diferentes planes de pago. La versión gratuita que permite crear 2 chatbots y enviar 1000 mensajes al mes. Si estas condiciones no fueran suficientes, el Starter Plan que son unos 30 $ al mes, permitiendo ampliar la cantidad de bots posibles a crear y la cantidad de mensajes que manejan. Además, tiene la ventaja de que el mismo proyecto puede compilarse para diferentes plataformas entre las que se incluyen WhatsApp, Facebook, Telegram y Web. Sin embargo, esta aplicación no permite enviar externamente variables a una base de datos para que se puedan registrar las respuestas de los usuarios.

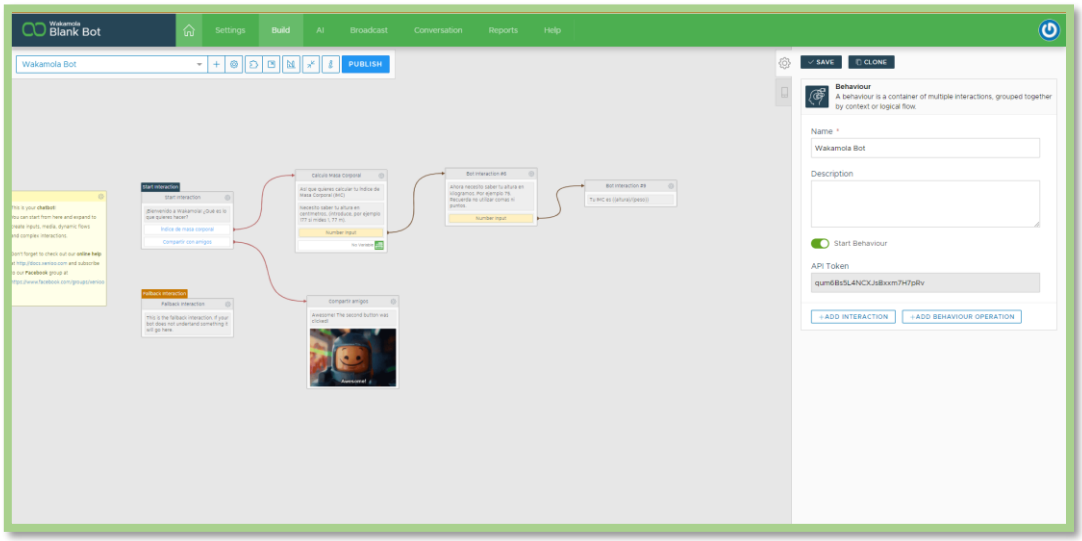

Ilustración 21 Ejemplo del desarrollo de una aplicación con Xenioo

3.2.2.2 Azure

Se exploró la opción de Azure, que permite introducir un chat Bot en una página web. Azure permite crear de un bot de forma gratuita que envíe mensajes ilimitados por canales estándar. Por canales premium cuesta 0,04€ por cada 1000 mensajes. ("Precios de Azure Bot Service | Microsoft Azure", s. f.). Además, se cobrará Azure App Service, pero también existe un plan gratuito para este servicio que se adapta a los requisitos del proyecto. Además, admite la posibilidad de escalar las capacidades del sitio web por 0.011€ / hora.

3.2.2.3 SnatchBot

Otra herramienta interesante para utilizar sería SnatchBot que permite la creación y personalización de chatbots. Permite un número ilimitado de mensajes, interacciones ilimitadas, y un número ilimitado de bots. No obstante, aparecerá en un margen la marca de agua de SnatchBot y no permite la edición del bot desde varias cuentas al mismo tiempo ("Precios de Chatbot | SnatchBot", s. f.). La versión 'Pro' tiene un coste de 30$ mínimo al mes, para enviar un total de 10000 mensajes.

4 Solución final propuesta

Al equipo Wakamola se le mostraron todas estas propuestas y, tras la deliberación, se llegó a la conclusión de que se debía de elaborar una **página web propia** que se conectara al servidor de Wakamola y que recrease, con la mayor fidelidad posible, el servicio levantado en Telegram.

Para ello se planteó que sería necesario el desarrollo de un **Frontend** capaz de conectarse al **Backend** donde ya se está ejecutando el código de Wakamola. Además, deberá de adaptarse el actual Backend al nuevo servicio Frontend para que sean capaces de intercambiar mensajes entre ellos.

4.1 Arquitectura del nuevo Wakamola

La arquitectura desarrollada parte de la existencia de un Backend funcional ya desarrollado completamente, con una API donde se conecta la aplicación de Telegram. El nuevo Frontend que se desarrollará permitirá la visualización web de un chat similar al de la aplicación de Telegram, admitiendo las mismas funcionalidades. El Backend común es el que se conectará la base de datos y realiza las manipulaciones correspondientes de los datos. Para comunicar ambas máquinas se utilizará un servicio de API REST. La idea general de la estructura del proyecto Wakamola puede observarse en la ilustración 8.

Ilustración 22 Estructura de la aplicación

Para un usuario de la aplicación Wakamola web, las rutas que seguirán los datos y se intercambiarán entre el cliente y el servicio Frontend desarrollado en Node.js mediante sockets. Este servicio de Node.js se comunicará mediante peticiones POST del protocolo HTTP con el Backend desarrollado en Python, concretamente con el framework de Flask. Ambos servicios

únicamente utilizarán peticiones POST. Puede observarse una representación de estos procesos en la ilustración15. La aplicación de Flask se encargará tanto de extraer como de introducir y modificar los datos almacenados en la base de datos de MariaDB. Además, la aplicación de Flask escuchará, por un lado, las peticiones de Wakamola Telegram y, por otro, las peticiones de Wakamola web, dado que los mensajes seguirán la misma estructura.

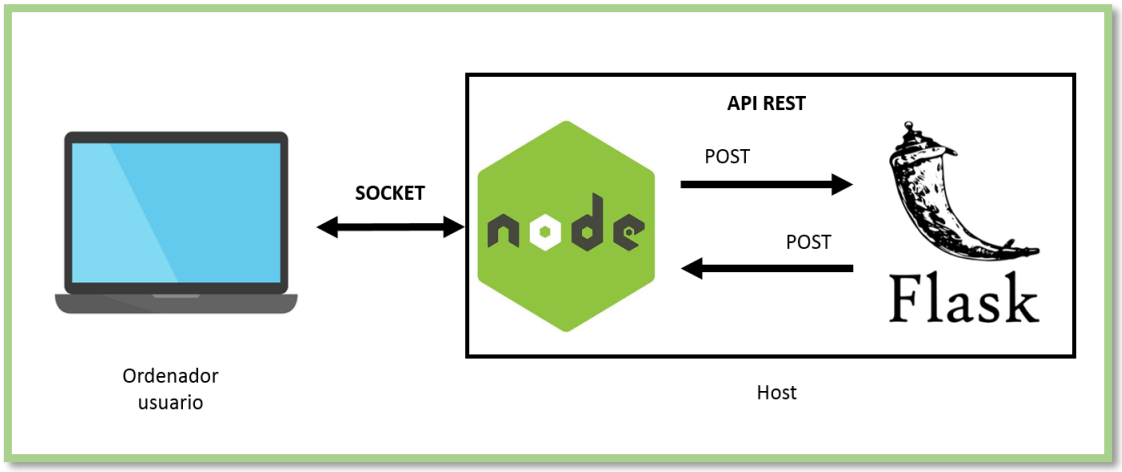

Ilustración 23 Rutas y estructuras involucradas en la comunicación de máquinas en la aplicación de Wakamola web

4.2 Etapas de desarrollo del proyecto

Dada la complejidad del proyecto, este se dividió en varias etapas.

1 **Estudio y selección de las tecnologías de implantación**
Se desarrolló una investigación sobre las diferentes posibilidades que había para el desarrollo de la nueva plataforma. Una vez investigadas, probadas y sopesadas las diferentes posibilidades, se elaboró la hoja de ruta del proyecto

2 **Desarrollo del Frontend**
Se empezó por el desarrollo de la parte visual del proyecto, por la que acceden los usuarios, mientras se estaba trabajando en la parte del Backend. Se utilizó un simulador de Backend mediante la aplicación de Postman, que permitió simular los mensajes que llegarían desde el Backend para comprobar el correcto funcionamiento una vez desarrollado este.

3 **Desarrollo del Backend**
Como no se avanzó en este campo, se tuvo que entender el código ya desarrollado para la aplicación de Wakamola Telegram y adaptarlo para que fuera capaz de recibir peticiones HTTP desde el Frontend. De nuevo, para evitar conectar ambas partes, se realizaron las pruebas simulando los mensajes con Postman. Una vez, comprobado el correcto funcionamiento se conectaron ambas partes y se testeó su funcionamiento.

4 **Implantación del software en servidor**
Dado que se desarrolló el código en una máquina distinta en entorno de desarrollo que, en el entorno de producción, pues una tenía un sistema operativo Windows y la otra, un sistema operativo Linux, se tuvo que encapsular la aplicación mediante el uso del software Docker. Este software tiene una curva de aprendizaje elevada. Se tuvieron que solucionar múltiples errores debidos a esta aplicación.

5 **Realización de pruebas**
Por último, se lanzaron unas pruebas en local con diferentes usuarios para evaluar tanto las posibilidades de adquisición de los datos de los usuarios, como de sus opiniones respecto a la aplicación, tanto de las partes estéticas, funcionales como de sus posibilidades de implantación en el mercado (aunque sea de forma no comercial al tratarse de un proyecto Creative commons).

6 **Análisis de los resultados**
Por último, se analizaron los datos obtenidos de estas pruebas realizadas a los usuarios, tanto de los datos obtenidos por la propia aplicación, como de sus opiniones del aspecto, usabilidad para determinar tanto si el trabajo realizado hasta el momento es correcto, como de posibles mejoras que pudieran realizarse.

5 Tecnologías utilizadas

En el proceso de creación de la nueva versión de Wakamola se han utilizado diferentes recursos, tanto de lenguajes de programación, como de frameworks y librerías ampliamente utilizadas hoy en día para el desarrollo web. También se han utilizado softwares ajenos para el desarrollo del código, su testeo y su despliegue.

5.1 Lenguajes utilizados

Para la elaboración del presente proyecto se han utilizado los HTML, CSS, JavaScript y Python como lenguajes de programación para la creación del proyecto.

- **HyperText Markup Language (HTML)**

HTML (lenguaje de marcado de hipertexto) es el componente de las páginas web que les aporta la estructura. Se denomina de "Hipertexto" porque utiliza enlaces para vincular las páginas web entre sí, formando entre ellas un conjunto.

El "Marcado" viene porque utiliza elementos especiales denominados etiquetas para etiquetas o marcar texto, imágenes y otro contenido. Existe una amplia cantidad de etiquetas que se caracterizan por estar contenidas entre los símbolos "<" y ">". Además, poseen la característica de que deben de cerrarse siempre para indicar donde finaliza el contenido de dicha etiqueta con el símbolo "</" y ">", de forma que se obtiene una estructura arbórea ("HTML: Lenguaje de etiquetas de hipertexto | MDN", 2020).

- **Cascading Style Sheets (CSS)**

El CSS (hojas de estilo en cascada) es el lenguaje de estilos utilizado para la presentación de documentos, tanto HTML como XML. Se trata de un archivo que contienen los elementos embellecedores del estilo de los archivos de estructura. EN estos documentos se puede indicar la posición de los elementos, la tipografía de letra utilizada, el tamaño de esta, su color. Incluso algunos elementos básicos de cambio de apariencia como cuando se sitúa el puntero del ratón por encima del elemento ("CSS | MDN", 2020).

- **JavaScript (JS)**

JavaScript es un lenguaje de programación diseñando para ser utilizado en los ordenadores de los usuarios (clientes) en la navegación web. Se trata de un lenguaje de scripting para páginas web, ligero, interpretado o compilado *just-in-time* (justo a tiempo), de un solo hilo, dinámico y con soporte para la programación orientada a objetos. Aunque se utiliza principalmente para las páginas web, también se utiliza en entornos fuera del navegador, como Node.js o Adobe Acrobat ("JavaScript | MDN", 2020). Con JavaScript vuelve las páginas web dinámicas, es decir, capaces de responder a las acciones del usuario, como pulsar un botón o escribir en un campo de texto.

- **Python**

Python es un lenguaje de programación interpretado, dinámico y multiplataforma que se caracteriza por su fácil legibilidad. Soporta tanto la programación orientada a objetos (POO), la programación imperativa y programación funcional. Lo administra la Python Software

Foundation y posee una licencia de código abierto. (“Python - Wikipedia, la enciclopedia libre”, s. f.) Se ha utilizado para el desarrollo de la parte del Backend de la aplicación.

- **Structured Query Language (SQL)**

SQL es el sublenguaje desarrollado a mediados de los años 70 para acceder y manipular las bases de datos relacionales, manejadas por los sistemas de manipulación de bases de datos relacionales (RDBMS) (Melton, Jim; Simon, 1993). Algunos de los RDBMS más conocidos son MySQL, MariaDB, SQLite, PostreSQL o SQLServer, pero para acceder, modificar, introducir o eliminar los datos que almacenan todos ellos utilizan una estructura similar de consulta, el lenguaje SQL.

Para la presente aplicación se ha utilizado para introducir y modificar los datos de forma permanente dentro del equipo (de forma masiva) en la base de datos creada en MariaDB.

5.2 Librerías, frameworks, middlewares y utilidades utilizados

Además de los lenguajes de programación anteriormente mencionados, se han utilizado **Node.js, Express, Bootstrap, jQuery, Morgan** y **Nodemon** para elaborar principalmente la parte del Frontend. Para la parte del Backend se han utilizado múltiples librerías, de las cuales destacar el uso de **Flask** para el desarrollo de servidores web en Python.

- **Node.js**

Es un entorno de tiempo de ejecución multiplataforma basado en el lenguaje de programación JavaScript. Además, posee la cualidad de que es de código abierto, es asíncrono, ya que su arquitectura de entrada salida de datos está orientada a eventos y se basa en el motor V8 de Google. Fue creado por Ryan Dahl en 2009 para crear programas de red altamente escalables, como los servidores web. Cabe destacar que, al contrario que la mayoría del código de JavaScript, no se ejecuta en el navegador, sino en el servidor.(“Node.js”, 2020).

- **Express**

Es un esquema (framework) para el desarrollo y/o la implementación de una aplicación web para Node.js. Express facilita el enrutamiento de direcciones bajo las solicitudes HTTP. Se utiliza para indicar qué documentos deben de mostrar o qué datos deben de almacenar las peticiones GET y POST (Del Ra, 2015).

- **Bootstrap**

Es un framework de código abierto para la creación de interfaces de usuario, creado para realizar un desarrollo web más rápido y sencillo. Bootstrap posee todo tipo de plantillas HTML y CSS para diversas funciones y componentes, como el sistema de cuadrícula. Además, permite diseñar una plataforma que se adapte a la pantalla del dispositivo en el que se muestre. (“¿Qué es Bootstrap? - Una Guía Completa para Principiantes”, s. f.)

- **jQuery**

Es un framework de JavaScript que se adapta al navegador web en el que se ejecuta el código por parte del usuario. Facilita el acceso a los elementos del DOM (Document Object Model), es decir, a las estructuras del código HTML para modificar sus propiedades.(“Introducción a jQuery”, s. f.)

- **Morgan**

Es un middleware que se utiliza para mostrar en la consola las peticiones de protocolo HTTP, errores, entre otras funcionalidades. (“Getting Started With morgan: the Node.js Logger Middleware | DigitalOcean”, s. f.) Resulta de gran utilidad para testear y debuggear API Rest.

- **Nodemon**

Es una utilidad para desarrolladores en JavaScript que detecta automáticamente cuando se han realizado cambios en el servidor y lo reinicia automáticamente.(“nodemon”, s. f.) Facilita el desarrollo del frontend.

- **Flask**

Flask es un web framework desarrollado en Python para la creación de páginas web. Incluye un servidor web de desarrollo para probar y visualizar el código añadido mientras se desarrolla (Domingo Múñoz, 2017). Permite el desarrollo del código ejecutado en servidor completamente en Python. Además, con la instalación de diferentes plugins y de librerías permite la gestión del acceso a la base de datos.

5.3 Herramientas utilizadas

Para el desarrollo de software, no solo son necesarios lenguajes, librerías y frameworks, sino también otras herramientas para el desarrollo más rápido de código, como los IDE, o herramientas para comprobar su funcionamiento.

- **Visual Studio Code**

Visual Studio Code es el editor de código fuente utilizado para elaborar el código en Node.JS, JavaScript, HTML y CSS. Se trata de un editor gratuito y de código abierto desarrollado por Microsoft que funciona tanto en Windows, Linux como en macOS. Tiene soporte para la depuración, control integrado de Git, resaltado de sintaxis y, además, permite se mejora continua con la instalación de plugins específicos para facilitar el desarrollo de código. [35]

- **PyCharm**

Es un IDE (Integrated Development Environment) específico para el desarrollo de código fuente en Python. En el desarrollo de esta aplicación se ha utilizado para el desarrollo de la parte del backend. Creado por JetBrains (“PyCharm: uno de los mejores IDE para Python — Escuela de Python”, s. f.)

- **Postman**

Postman es una herramienta que permite la creación de todo tipo de peticiones personalizadas del protocolo HTTP, como GET, POST, PULL, … para poner a prueba la API desarrollada de servicios REST. Las peticiones pueden ser guardadas y repetidas a posteriori. Además, permite adaptar el mensaje para que contenga tanto texto plano como JSONs.

Postman permite definir colecciones de APIs, gestionar la documentación de la API, establecer variables locales y globales, entre otras muchas funcionalidades. [37]

- **Docker**

Docker es un proyecto open source de la empresa Docker Inc. Se trata de una tecnología que permite la creación y uso de máquinas virtuales muy ligeras y modulares a las que denomina “contenedores” Estos contenedores pueden estar basados en sistemas operativos Linux o Windows. Los contenedores admiten ser copiados, implementados y movidos entre diferentes entornos de producción y desarrollo, sin importar el sistema operativo y las librerías instaladas en la máquina del desarrollador, de pruebas y de producción. Así pues, se puede desarrollar código en un ordenador con un sistema operativo, por ejemplo, Windows y en producción puede ejecutarse el mismo código en una máquina con un sistema operativo diferente, por ejemplo, Linux.

Docker utiliza el kernel de Linux y sus funcionalidades para ejecutar el código de los contenedores de forma independiente para hacer un mejor uso de la infraestructura del dispositivo en el que se ejecuta y, al mismo tiempo, conservar la seguridad que se tiene en sistemas separados.

Un contenedor se basa en una imagen de Docker, junto con las librerías y variables de entorno que necesita y el código a ejecutar. Cuando se combinan a estos tres elementos y se ejecutan es cuando se forma un contenedor de Docker. Además, permite la creación de redes de

contenedores (como si de redes de máquinas físicas se trataran9 para facilitar la conexión de aplicaciones desarrolladas en distintos entornos. [38]

- **Google Chrome**

Es un navegador web gratuito de código cerrado desarrollado por Google. Está disponible desde 2008 para Windows, macOS, Linux, Android y iOS. [39] Posee una herramienta en modo desarrollador que facilita el control sobre los elementos que componen el código en HTML, CSS y JavaScript, facilitando el desarrollo de páginas web.

- **MariaDB**

Es una base de datos relacional de código abierto. Fue creada por los creadores originales de MySQL. Se utiliza en muchos proyectos en la nube y por defecto en la mayoría de las distribuciones de Linux.(“MariaDB Foundation - MariaDB.org”, s. f.) A diferencia de SQLite, tiene una gran respuesta, es consistente y robusta.

En las primeras versiones de la aplicación se utilizó SQLite para probar las funcionalidades de almacenamiento y recuperación de la información, pero dada la futura escalabilidad de la aplicación y de que se puedan conectar otros servicios a la misma fuente de datos, se decidió cambiarla en versiones posteriores a MariaDB, por sus mejores prestaciones con grandes cantidades de datos.

6 Ejecución del proyecto

Para el desarrollo del proyecto de Wakamola web se elaboró el siguiente diagrama de Gantt con los tiempos dedicados a cada parte del proceso. Las tareas fueron adaptadas a los tiempos reales finales utilizados para desarrollar la aplicación.

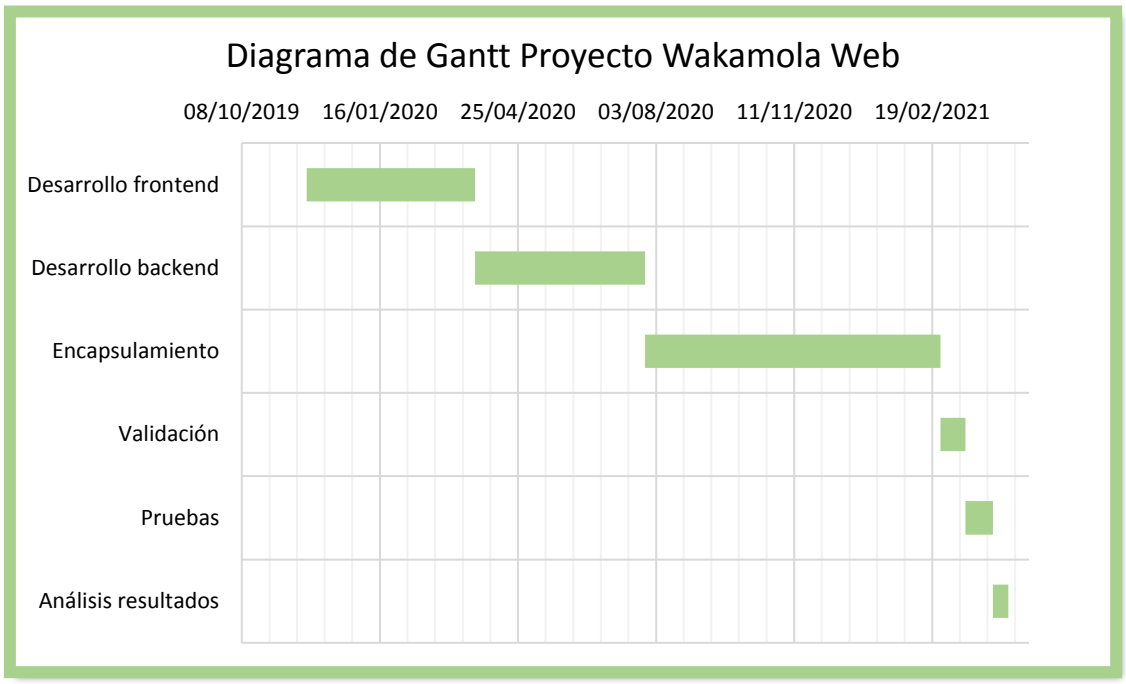

Figura 3 Diagrama de Gantt del proyecto Wakamola web

6.1 Desarrollo del Frontend

En primer lugar, se procedió a desarrollar una interfaz amigable que permitiera el envío de información desde la máquina del usuario a un servidor. Dado que el servidor Wakamola no se quería que estuviera directamente expuesto a las vulnerabilidades de la red, se creó un primer Backend en Node.js que recibía la información introducida en los dispositivos de los usuarios (ordenador, smartphone o Tablet) y enviaba sin procesar directamente al Backend. Además de contar con una forma para la realización del registro de los usuarios y de ser capaz de enviar y recibir solicitudes de amistad para codificar las relaciones entre los usuarios.

6.1.1 Diseño del Frontend

Cuando un usuario accede a la URL de Wakamola web, se le muestra la página web de inicio. Esta página tiene la apariencia de la ilustración 24. Aparece un encabezado en la parte superior junto con un pie de página indicando que la obra se encuentra bajo la protección de Creative Commons.

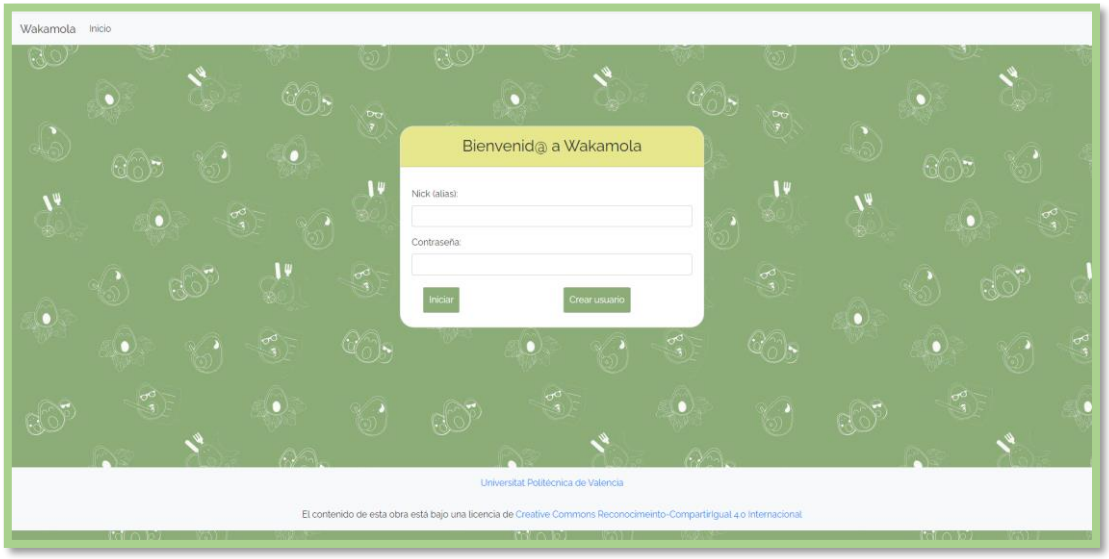

Ilustración 24 Imagen de la pantalla del inicio de la aplicación

En el centro se encuentra la forma para que los usuarios se registren o accedan. Aparece un campo para introducir el nombre, un campo para introducir la contraseña, un botón para acceder si ya se ha accedido antes a la aplicación y otro si es la primera vez que se accede para registrarse en la base de datos. La presentación se ha diseñado de forma que se adapta al dispositivo del usuario. Así, la apariencia en smartphones es distinta a la de la vista en ordenador, como puede verse en la ilustración 25, donde se observa la apariencia que tiene la aplicación al cargarse en un smartphone standard.

Ilustración 25 Visualización de la pantalla de inicio de la aplicación en un smartphone

Los usuarios han de rellenar los campos correspondientes al nombre de usuario y la contraseña. No se ha especificado ninguna clase de regla, ni para el nombre de usuario ni para la contraseña. Únicamente cuando se desea crear un nuevo usuario, se comprueba que el nombre introducido no es utilizado por ningún otro usuario. Además, para mayor seguridad, la contraseña se guarda codificada en la base de datos, de forma que ni incluso un usuario con acceso a la base de datos sería capaz de conocer la contraseña del usuario. Deben de introducirse un nombre y una contraseña y pulsar o el botón “Iniciar” si el usuario ya se había registrado previamente o “Crear usuario” si el usuario no se había registrado previamente. Esto se ha creado para tener diferentes métodos para cada proceso, por si un proceso se debe de modificar el Frontend creando, por ejemplo, un formulario distinto para registrarse y acceder.

En la forma se muestra un mensaje de error si no coinciden el nombre de usuario con la contraseña o si se quiere registrar el usuario con un nombre ya utilizado y que, por lo tanto, aparece en la base de datos. El mensaje aparece en rojo en la parte superior de la forma, como puede observarse en la ilustración 26.

Ilustración 26 Detalla del mensaje de error que aparece en caso de no introducir correctamente los datos en la forma

Una vez se ha registrado correctamente, la aplicación muestra el aspecto de la imagen 27. Aparecen varias formas: el chat en la zona izquierda y otras dos formas en el lado derecho (para el diseño en grandas pantallas y medias, como ordenadores y tablets).

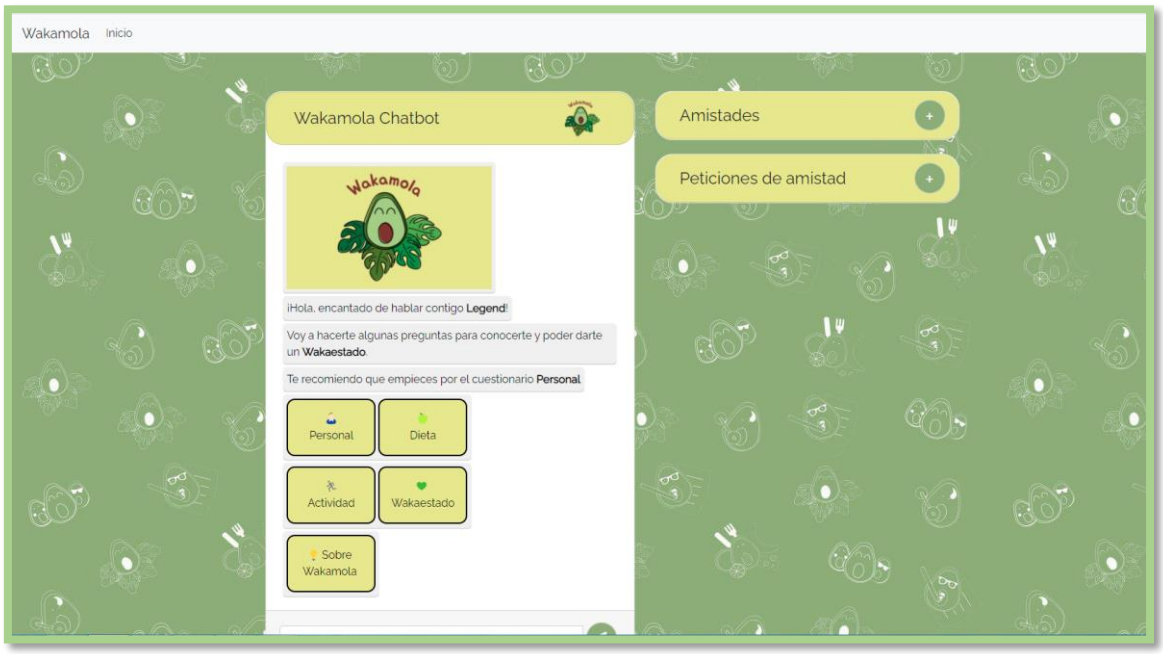

Ilustración 27 Captura de pantalla de la aplicación una vez se ha iniciado en la pantalla de inicio con los campos laterales plegados

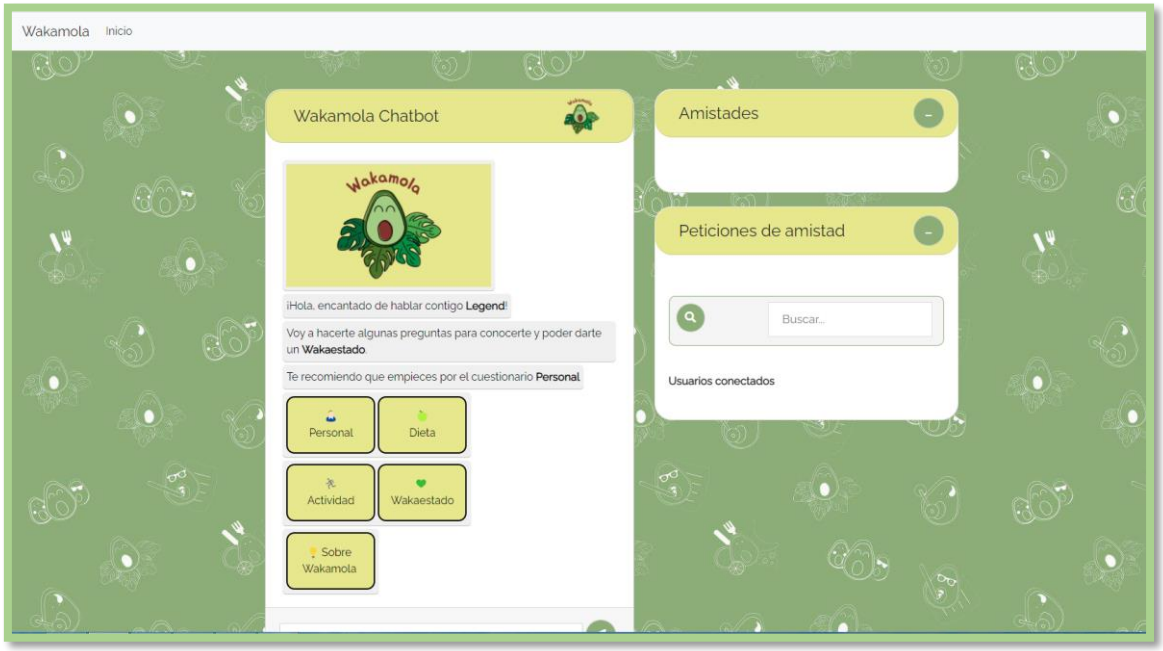

Ilustración 28 Captura de pantalla de la aplicación una vez se ha iniciado en la pantalla de inicio con los campos laterales desplegados

El diseño para smartphones de pantalla pequeña cambia. Las formas desplegables contienen, por orden, la lista de amistades del usuario en la aplicación y la forma para buscar nuevos usuarios. En el diseño para smartphones, debido a la falta de espacio, aparecen en la parte superior, como puede observarse en la ilustración 29.

Ilustración 29 Captura de pantalla de la aplicación una vez se ha iniciado en la pantalla de inicio en un smatrphone

Estas formas se muestran, en una primera instancia, plegadas dado que en el diseño para smartphones ocupaban demasiado espacio como para mostrarse desplegadas.

En la zona del chat, siempre al iniciar la aplicación se envía un mensaje especial al Backend, para que envíe no solo los mensajes de bienvenida, las imágenes y los botones de opciones del chat, sino también la lista con las amistades del usuario y las diferentes peticiones de amistad. La lista de amistades aparece en el campo “Amistades” y en el campo “Peticiones de amistad” aparecen las diferentes peticiones de amistad, junto con un botón para aceptar la petición o rechazarla. En esta forma también aparece un buscador para buscar usuarios que no están conectados y una lista de los usuarios que están conectados en el mismo momento que el usuario. S se buscan usuarios se les puede enviar una petición de amistad.

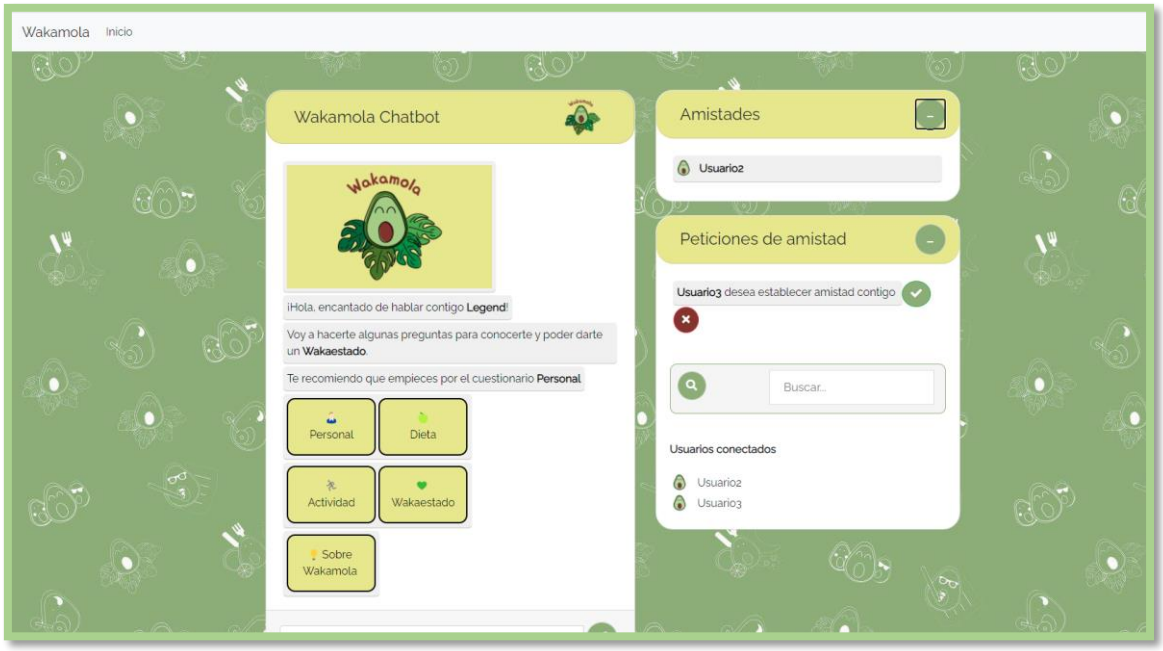

Ilustración 30 Captura de pantalla de la aplicación cunado se ha recibido una petición de amistad

6.2 Desarrollo del Backend

En un principio esta parte del proyecto escapaba a la extensión del proyecto, pero al resultar de vital importancia lograr comunicar el Backend con el Frontend se decidió llevarla a cabo. Para el desarrollo del Backend se partió de la versión original del código de Wakamola Telegram. En primer lugar, se creó una API REST con el framework Flask de Python para poder comunicarse con el servicio Frontend. La lógica que sigue la aplicación es la siguiente. Se modificaron levemente los métodos que había en el código original de Wakamola para procesar las peticiones.

A continuación, se empezaron a realizar pruebas para tanto almacenar información en la base de datos como recuperarla. Así se tuvieron que crear los métodos para registrarse, crear amistades, almacenar las respuestas, … en una primera instancia se probó con la base de datos "SQLite3". Más adelante del proyecto, tuvo que modificarse por MariaDB, una RDBMS (Relational Database Management System) más avanzada para el almacenamiento de grandes cantidades de datos, cosa que SQLite se quedará limitada en un entorno de producción, además de que permite tener la base de datos como un servicio distinto, cosa que facilita la mejora del diseño en producción. Por lo tanto, para favorecer la escalabilidad del proyecto, se tuvo que modificar levemente las consultas a la base de datos para adaptarlas del formato SQLite a MariaDB (MySQL).

6.2.1 Estructura de los JSON enviados

La comunicación entre los usuarios y el servicio de Node.js se establece mediante sockets de forma que la información se envía al servidor del Frotend de forma como si se estuviera trabajando en el mismo servidor y no requiere de estructuración desde el código desarrollado. Sin embargo, entre los servicios de Node.js y Flask sí que se tiene que crear una estructura para los mensajes entre ellos. Para ello, se decidió utilizar el estándar JSON (JavaScript Object Notation). Se tuvo que crear un mensaje establecido para cada una de las acciones que podía realizar el usuario.

Todos los mensajes de los JSON tienen los siguientes campos en común:

- **"Lang":** es el lenguaje que ha seleccionado el usuario. Actualmente únicamente está habilitado el castellano, por lo que el único valor admitido es "es".
- **"Frotend_code":** es el código del Frontend utilizado, por si utiliza otra plataforma para la visualización de Wakamola. Se ha utilizado únicamente el parámetro "5555" de momento para la versión web actual.
- **"User":** es el nombre de usuario que ha ejecutado la acción cuando se trata de un mensaje del Frontend al Backend o al que va ejecutada la acción cuando el mensaje se envía del Backend al Frontend.
- **"Type":** es la acción que se va a realizar. Puede ser perteneciente al proceso de registrar un usuario ("register"), acceder un usuario ya registrado ("register"), enviar un mensaje("send_message"), …
- **"Content":** es el contenido necesario ara que se pueda ejecutar la acción deseada.

Así, se crearon mensajes estandarizados en la notación JSON para cada una de las acciones de los usuarios:

1. Registrar un usuario
2. Acceder un usuario
3. Envío de mensajes
4. Peticiones de amistad
5. Búsqueda de usuarios

Cada mensaje enviado entre procesos tiene sus correspondientes respuestas, en función de la casuística de la información recibida. Así pues, los mensajes desarrollados para cada u a de estas acciones se desarrollan a continuación.

- **Registrar usuario**

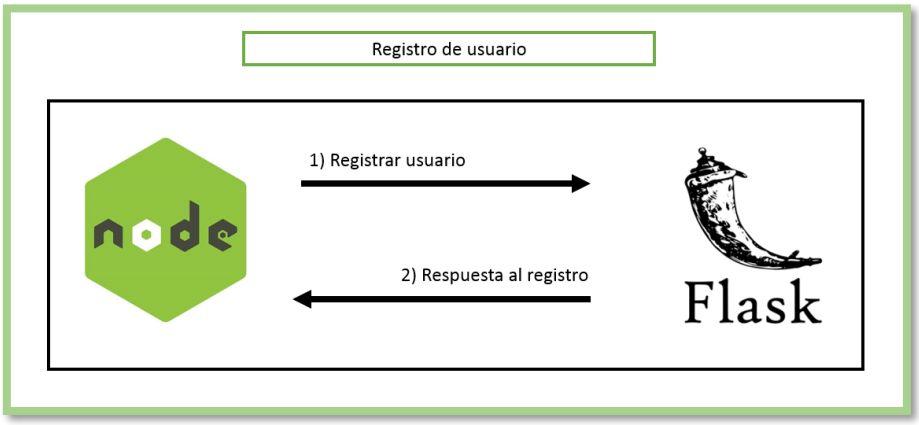

Ilustración 31Represetnación del proceso de comunicación entre servicios durante el proceso de registrar un usuario

1) Registrar usuario desde el Frontend: Cuando se pulsa el botón de registrar un usuario desde el Frontend, se envía el siguiente JSON al Backend:

```
{
    "_comment": "Este es el mensaje del frontend al backend cuando
se registra un usuario",
    "lang": "es",
    "type": "register",
    "frontend_code": "5555",
    "user": "NombreUsuario",
    "content": {
        "password": "contraseñaUsuario"
    }
}
```

2.a) Respuesta positiva: Para que el Frontend pueda mostrar las formas del chat, de las amistades y de las peticiones de amistad, se tiene que confirmar que el usuario no aparece en la base de datos. En caso de que no aparezca en la base de datos el nombre de “user”, se envía:

```
{
    "_comment": "Esta es la respuesta del backend al frontend cuando se
registra un usuario correctamente",
    "lang": "es",
    "type": "register",
    "frontend_code": "5555",
    "user": "NombreUsuario",
    "content": {
        "status": 1,
        "error_message": ""
    }
}
```

2.b) Respuesta negativa: el mensaje envía un error y un "status" o cuando el usuario aparece en la base de datos.

```
{
    "_comment": "Esta es la respuesta del backend al frontend cuando se
intenta registrar un usuario incorrectamente",
    "lang": "es",
    "type": "register",
    "frontend_code": "5555",
    "user": "NombreUsuario",
    "content": {
        "status": 0,
        "error_message": "The user introduced is not valid"
    }
}
```

- **Acceso de un usuario**

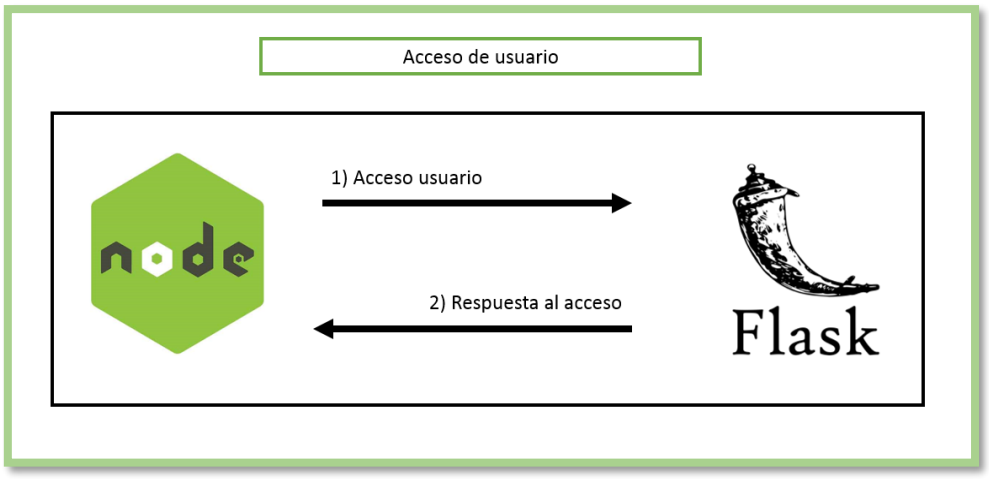

Ilustración 32 Representación del proceso de comunicación entre servicios durante el proceso de acceder un usuario

1) Inicio: Cuando el usuario pulsa sobre el botón de "Iniciar", se tiene que comprobar que tanto el nombre del usuario como la contraseña se han introducido correctamente. El mensaje enviado desde el Frontend adquiere la forma:

```
{
    "_comment": "Este es el mensaje del frontend al backend cuando
intenta acceder un usuario",
    "lang": "es",
    "type": "login",
    "frontend_code": "5555",
    "user": "NombreUsuario",
    "content": {
        "password": "contraseñaUsuario"
    }
}
```

2.a) Acceso concedido: Si el usuario aparece en la base de datos y la contraseña introducida coiniciden, se le concede acceso al usuario mediante el siguiente mensaje. Desde el Frontend se muestra tanto el chat como las formas de amistades y peticiones de amistad, tal y como se ha hecho en el caso anterior.

```
{
    "_comment": "Esta es la respuesta del backend al frontend cuando
accede un usuario correctamente",
    "lang": "es",
    "type": "login",
    "frontend_code": "5555",
    "user": "NombreUsuario",
    "content": {
        "status": 1,
        "token": "53423542323",
        "error_message": ""
    }
}
```

2.b) Acceso denegado: si el nombre del usuario introducido no aparece en la base de datos, o si no coinciden la contraseña y el nombre introducido, se procede a enviar al Frontend el mensaje:

```
{
    "_comment": "Esta es la respuesta del backend al frontend cuando
intenta acceder un usuario no registrado",
    "lang": "es",
    "type": "login",
    "frontend_code": "5555",
    "user": "NombreUsuario",
    "content": {
        "status": 0,
        "token": "",
        "error_message": "The user introduced is not valid"
    }
}
```

- **Envío de mensajes**

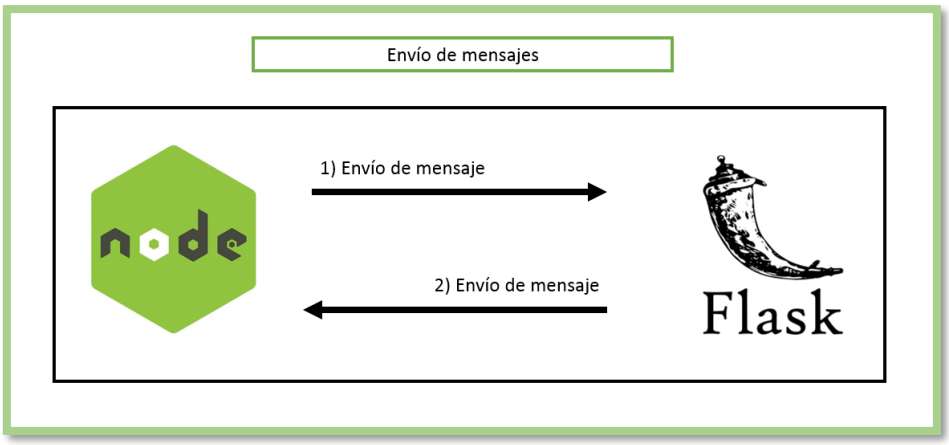

Ilustración 33 Representación del proceso de comunicación entre servicios para el envío de mensajes

1) Envío mensaje desde el Frontend: cuando el usuario pulsa el botón de enviar o Enter en el teclado, se envía un mensaje desde el servidor Frontend hasta el Backend utilizando la siguiente sintaxis de JSON:

```
{
    "_comment": "Esto es el mensaje del usuario que envía el
frontend al backend",
    "lang": "es",
    "type": "message",
    "frontend_code": "5555",
    "message_id": 7474,
    "user": {
        "user_name": "NombreUsuario",
        "token": "53423542323"
    },
    "content": "Mensaje enviado del usuario"
}
```

2) Envío de mensaje desde el Backend: Cuando el Backend acaba de procesar el mensaje que ha recibido del usuario, o cuando dentro del programa corresponda, el Backend le envía un mensaje al usuario un mensaje capaz de indicar la imagen a mostrar, una lista de mensajes, una lista de elementos que mostrar como un array o incluso, si mostrar el menú de inicio de Wakamola. Todo ello puede visualizarse ene le siguiente ejemplo:

```
{
    "_comment": "Esto es el mensaje que recibe el frontend desde el back
end",
    "lang": "es",
    "type": "message",
    "frontend_code": "5555",
    "user": "NombreUsuario",
    "content": [
        {
            "type": "image",
            "element": "NombreImagen"
        },
        {
            "type": "text",
            "element": "Mensaje a mostrar al usuario"
        },
        {
            “type”: “keyboard”,
            “element”: [
                    “elemento1”,
                     “elemento2”,
                     “elementoN”
             ]
        },
        {
            “type”: “main_menu”
        }
    ]
}
```

- **Peticiones de amistad**

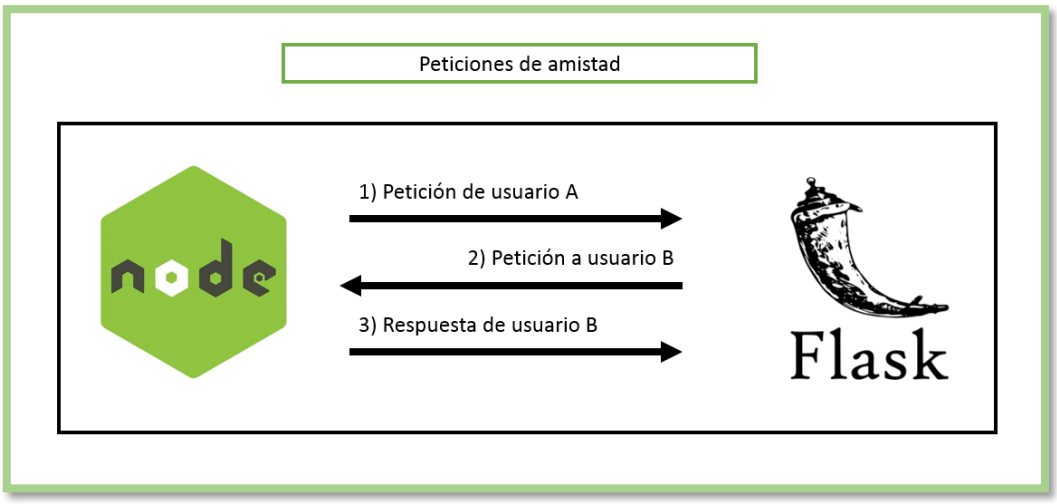

Ilustración 34 Representación del proceso de comunicación entre servicios durante el proceso de realizar una petición de amistad

1) Envío de la petición del usuario A al Backend: El procesamiento de las peticiones de amistad ocurre en tres pasos. El primero es cuando un usuario pulsa sobre e icono de otro usuario, bien sea porque está conectado o bien porque le ha aparecido su nombre en una búsqueda y le ha pulsado al botón de agregar usuario. En cualquier caso, se le envía desde el Frontend al Backend un mensaje indicando el nombre del usuario emisor (A) y el nombre del usuario que recibe la petición (B).

```
{
    "_comment": "Envío de la petición de amistad del usuario A desde el
frontend al backend",
    "lang": "es",
    "type": "send_friend_request",
    "frontend_code": "5555",
    "user": {
        "user_name": "NombreUsuarioA",
        "token": "53423542323"
    },
    "content": {
        "target_user": "NombreUsuarioB",
    }
}
```

2) Envío de la petición del usuario A al Frontend del usuario B: Bien si el usuario B está conectado en ese momento o si se conecta más tarde, el Backend le indica al Frontend que ha recibido una petición de amistad con el siguiente formato:

```
{
    "_comment": "Envío de la petición de amistad del usuario A desde el fr
ontend al backend",
    "lang": "es",
    "type": "send_friend_request",
    "frontend_code": "5555",
    "user": {
        "user_name": "NombreUsuarioA",
        "token": "53423542323"
    },
    "content": {
        "target_user": "NombreUsuarioB",
    }
}
```

3) Envío de la respuesta del usuario B al Backend: Tanto si la respuesta es positiva como negativa, el Frontend le envía el siguiente JSON al Backend, indicando en el campo "result" si se ha aceptado (1) o no (0) la petición y actuar en consecuencia. SI se acepta, se guardan como amigos. Si se rechaza, se elimina la petición.

```
{
    "_comment": "Envío de la petición de amistad del usuario A desde
el frontend al backend",
    "lang": "es",
    "type": "response_friend_request",
    "frontend_code": "5555",
    "user": {
        "user_name": "NombreUsuarioB",
        "token": "53423542323",
    },
    "content": {
        "result": 1,
        "username_asks": "NombreUsuarioA",
    }
}
```

- **Buscar un usuario**

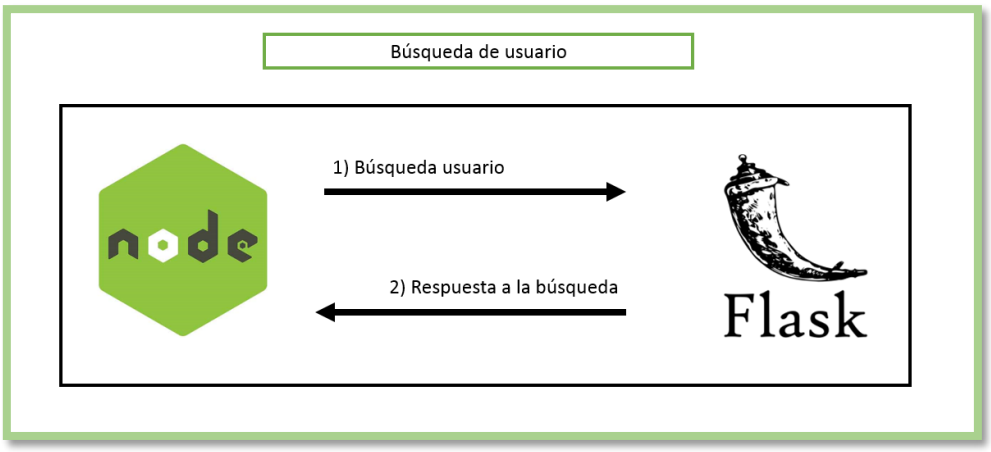

Ilustración 35 Representación del proceso de comunicación entre servicios durante el proceso de buscar un usuario

1) Envío de la petición de búsqueda de un usuario: Cuando un usuario busca el nombre de otro, se envía desde el Frontend el siguiente JSON al Backend, indicando quien realiza la búsqueda y el nombre del usuario buscado.

```
{
    "_comment": "Envío de la petición desde A para buscar usuario B en la
Base de Datos",
    "lang": "es",
    "type": "findUser",
    "frontend_code": "5555",
    "user": {
        "user_name": "NombreUsuarioA",
        "token": "53423542323"
    },
    "content": {
        "result": 0,
        "search_query": "NombreUsuarioB"
    }
}
```

2) Respuesta a la búsqueda: Tras hacer una búsqueda exacta en la base de datos, si ha aparecido el usuario buscado, se le notifica al Frontend del usuario que ha realizado la búsqueda con el siguiente JSON. Si ha aparecido el campo “msg” es verdadero (1), y falso (0) si el usuario buscado no aparece en la base de datos. El algoritmo de búsqueda es por coincidencia exacta, por lo que no da recomendaciones de nombres de usuario similares.

```
{
    "_comment": "Envío de la respuesta al frontend del usuario A de
buscar al usuario B en la Base de Datos",
    "lang": "es",
    "type": "found_users",
    "frontend_code": "5555",
    "user": {
        "user_name": "NombreUsuarioA",
    },
    "content": {
        "msg": 0,
        "search_query": "NombreUsuarioB"
    }
}
```

6.2.2 Algoritmo del Backend

El Backend es el encargado de procesar las peticiones y de realizar las transformaciones de los datos que correspondan a la petición y de modificar la base de datos en función de estos. Para ello, se sigue la siguiente lógica:

1. La **API REST** de Flask recibe una petición únicamente de tipo **POST** con el contenido del Frontend, que se han descrito en el apartado anterior
2. Esta petición es procesada por el método **"handle_message",** que discrimina en función de la propiedad **"message_type".**
3. Así se procesa de forma diferente:
    a. ***"register":*** Procederá a comprobar si el usuario está en la base de datos. Devolverá un valor positivo si no aparece o negativo en el caso contrario.
    b. ***"login":*** También procederá a comprobar si el usuario está en la base de datos y si cuadra la contraseña introducida.
    c. ***"findUser":*** Si se pulsa el botón de la lupa en "Buscar", se procederá a buscar en la base de datos si el nombre introducido aparece. Únicamente en el caso de que haya una coincidencia exacta (dado que los algoritmos de búsqueda laxa requieren de un mayor tiempo de procesamiento) se devolverá un estado positivo, de forma que le responda al Frontend que efectivamente, dicho usuario existe.
    d. ***"send_friend_request":*** En este caso, un usuario (A) desea establecer amistad con otro (B) y ha pulsado sobre el botón de "Enviar petición de amistad" en el Frontend. Se procederá a guardar en la base de datos que el usuario B ha recibido una petición del usuario A.
    e. ***"response_friend_request":*** El usuario B ha recibido la petición del usuario A y ha decidido o aceptarla o declinarla. En ambos casos, su respuesta se guarda en la base de datos.
    f. ***"message":*** Cuando desde el Frontend se envía un mensaje de este tipo, se procesan de diferente forma, en función del contenido de "**message content**".
        i. ***"start":*** Este es el mensaje que se envía desde el Frontend una vez se ha accedido a la forma del chat porque se ha verificado el acceso del usuario. Se busca si el usuario tiene amistades o peticiones de amistad pendientes y se envían al Frontend. Además, se envía la imagen y el texto de bienvenida, junto al array de botones que conforman el menú de inicio de Wakamola. Además, en la base de datos se pone el guarda que el usuario está en fase 0 (inicio). A este estado se vuelve cada vez que el usuario finalice una encuesta.
        ii. ***"personal":*** Si el usuario selecciona realizar la encuesta personal, se envía al Frontend el mensaje de inicio de la encuesta personal, con su imagen característica y su texto. Además, se cambia su estado en la base de datos a "Personal" (Codificado como 1). Este estado se mantendrá hasta que haya respondido correctamente a todas las respuestas de personal, entonces se cambiará al estado de inicio.
        iii. ***"food":*** Si el usuario selecciona realizar la encuesta sobre la dieta, se enviará la imagen y texto sobre la dieta. Se guardará en la base de datos que el usuario está en fase 2 hasta que conteste correctamente a todas

las respuestas correctamente. EN cuanto finalice se devolverá al estado 0 y se enviará a el menú de inicio.

iv. ***"activity":*** En este caso el usuario ha seleccionado completar la encuesta de actividad física. Se guardará que esté en la fase 3. Se mantendrá este estado, de nuevo, hasta que se completen todas las respuestas y se vuelva al estado inicial, enviando el menú de inicio.

v. ***"risk":*** En este caso, el usuario desea conocer su Wakaestado. En caso de que no se hayan completado toda la información, se le indicará que debe completar todas ellas, antes de poder proceder. Si ciertamente las ha completado, se le enviará un mensaje con las puntuaciones descompuestas para cada una de las encuestas, la valoración de su Wakaestado y el Wakaestado medio de sus contactos (si ha establecido amistades con toros usuarios).

vi. ***"credits":*** Como en la versión original de Wakamola, se le envía información al usuario sobre quiénes han participado en la creación del Proyecto Wakamola.

vii. Cualquier otro mensaje es la respuesta de una pregunta de la aplicación, por lo que se tiene en cuenta cuál es la respuesta que está respondiendo el usuario. Las respuestas pueden ser de respuesta de afirmación/negación, numérica entera, numérica decimal o unas opciones determinadas. La aplicación comprueba cuál es la respuesta que se está respondiendo y si coincide con la tipología de las respuestas a esa pregunta, se guarda en la base de datos dicha respuesta y se procede a cambiar en la base de datos a que el estado del usuario está en la siguiente pregunta.

A modo de aclaración, se desarrolló el gráfico lógico de la figura 4 para mayor compresión de la lógica que utiliza la aplicación de Backend.

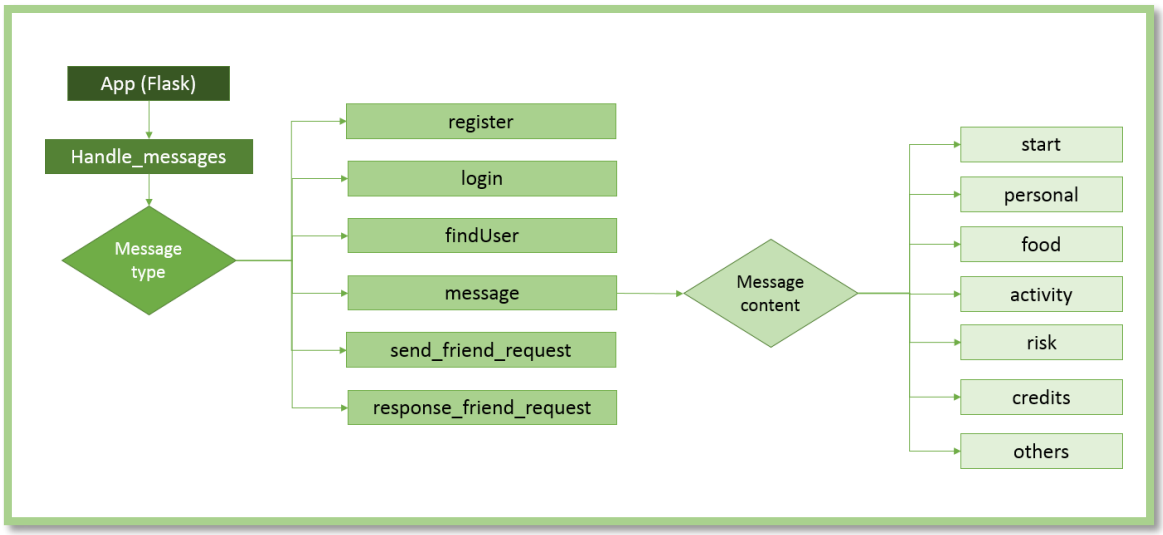

Figura 4 Algoritmo de las peticiones POST al Backend de Wakamola Web

6.2.3 Arquitectura de la base de datos

En la base de datos de la aplicación implementada en MariaDB se han diseñado las tablas: Users, Relationships, Status y Responses.

- **Users**

Users es la tabla principal en la que se registran los usuarios. Contiene información básica como el nombre de usuario, la contraseña y la fecha en la que se registró. La descripción de sus campos puede observarse en la tabla 2.

Tabla 2 Descripción de los campos de la tabla Users

id_User	**Entero**	**Clave primaria. Número identificador del usuario**
Nickname	Texto	Alias con el que se registra el usuario
Password	Texto	Contraseña del usuario
Created on	Fecha	Fecha y hora en la que se registra

- **Responses**

Responses es la tabla en la que se registran las contestaciones de los usuarios, además de cuando se realizó la respuesta del usuario, por si rellena múltiples veces las encuestas. La descripción de sus campos puede observarse en la tabla 3.

Tabla 3 Descripción de los campos de la tabla Responses

Id_User	**Entero**	**Clave ajena. Número identificador del usuario**
Phase	Entero	Estado en el que se encuentra el usuario. 1 => Personal 2 => Dieta 3 => Actividad
Question	Entero	Número de la pregunta en la que se encuentra
Answer	Texto	Respuesta a la pregunta
Timestamp	Fecha	Fecha y hora en la que se ha respondido

- **Relationship**

Relationship es la tabla en la que se registran las peticiones de amistad de los usuarios y si se han aceptado o rechazado. La descripción de sus campos puede observarse en la tabla 4.

Tabla 4 Descripción de los campos de la tabla Relationship

Active	**Entero**	**Clave ajena. Número identificador del usuario**
Passive	Entero	Encuesta de la que se ha respondido. 1 => Personal, 2 => Food y 3 => Actividad
Type	Entero	Número de la pregunta en la que se encuentra
Timestamp	Fecha	Fecha y hora en la que se ha respondido
Accepted	Boolean	Indica si se ha aceptado la petición de amistad (1) o no (0)

- **Status**

Status es la tabla que almacena el estado de cada usuario mientras completa los formularios. Permite conocer cuál es la última pregunta que ha respondido. También almacena si se han completado todas las encuestas y cuál es el último Wakaestado que ha obtenido. La descripción de sus campos puede observarse en la tabla 5.

Tabla 5 Descripción de los campos de la tabla Status

Id_User	**Entero**	**Clave ajena. Número identificador del usuario**
Phase	Entero	Estado en el que se encuentra el usuario. 0 => Inicio 1 => Personal 2 => Dieta 3 => Actividad
Question	Entero	Número de la pregunta en la que se encuentra
Completed personal	Boolean	Indica si se han respondido todas las preguntas personales (1) o no (0)
Completed food	Boolean	Indica si se han respondido todas las preguntas sobre la dieta (1) o no (0)
Completed activity	Boolean	Indica si se han respondido todas las preguntas sobre la actividad física (1) o no (0)
Last wakaestado	Decimal	Último Wakaestado obtenido (puntuación obtenida)

De esta forma, la base de datos sigue el siguiente esquema relacional que aparece en la ilustración 36, de forma que puede observarse como el campo id_User actúa como clave primaria de la tabla *Users* y como clave foránea de las tablas *Responses, Status y Relationship*.

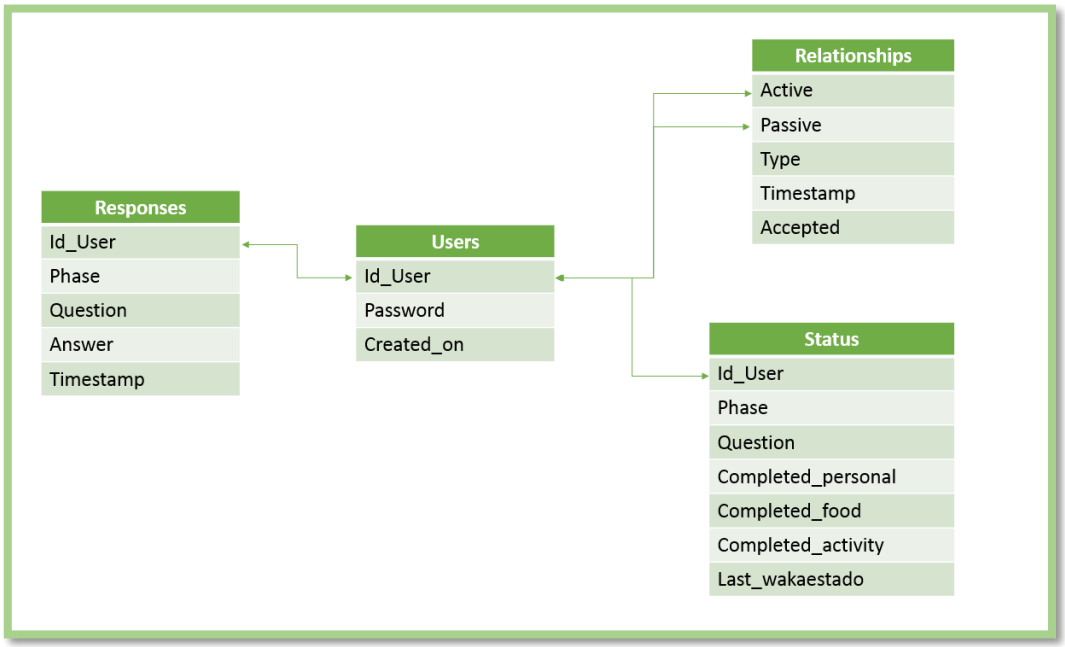

Ilustración 36 Diseño relacional de las diferentes tablas que componen la aplicación

6.3 Implantación del software en servidor

El terminal en que se realizó el desarrollo de la aplicación utilizaba un sistema operativo Windows 8.1. Sin embargo, el servidor en el que iba a realizarse las pruebas test utilizaba un sistema operativo Ubuntu (Linux).

Para lograr que el código funcionara correctamente en ambos dispositivos se optó por una opción común de entornos de producción como lo es Docker, un sistema que permite encapsular el código e introducirlo en una pseudo máquina virtual, el Docker Engine. Con Docker, se comprobó que el código funcionaba correctamente independientemente del sistema operativo de la máquina en que se ejecute.

En Docker, cada uno de los servicios (el Frontend y el Backend) se ejecutan en máquinas virtuales distintas (contenedores de Docker) que se comunican entre ellos utilizando peticiones del protocolo HTTP a sus respectivas API REST.

Para crear varios servicios y conectarlos entre si fue necesario utilizar la función Docker-Compose, diseñada para levantar simultáneamente varios contenedores al mismo tiempo y conectarlos entre ellos, como si de una red de ordenadores se tratara.

No obstante, cada vez que se crea una imagen con Docker, esto es, como una plantilla para la creación de los contendores, el contenido del proyecto de la máquina física se copia a la máquina virtual. Así, esto quiere decir que se copia el código actual del proyecto a la imagen. Resulta problemático pues se estaba utilizando la base de datos SQLite3, que permite integrar que permite incluir una base de datos integrada en la propia aplicación. Esta funcionalidad es útil cuando se trata de crear una aplicación de escritorio, pero en la solución actual resultaba problemático, pues cada vez que se reiniciaba el contenedor, se perdían los datos que esta contenía, pues se copia la base de datos vacía del código de desarrollo.

Para lograr que estos datos se mantuvieran, aunque el contenedor fuera eliminado, se tuvo que migrar el código del Backend para adaptarlo de SQLite3 a MariaDB. Dado que MariaDB funciona como otra aplicación independiente, se tuvo que crear un servicio nuevo para la base de datos y replicarlo en un contenedor de Docker. Además, al tener que crearse un contenedor de MariaDB, al eliminarse o reiniciarse el contenedor se pierden los datos contenidos en este. Para evitarlo, se utilizan los volúmenes de Docker, que permiten el almacenamiento de los datos de forma permanente en la máquina física.

A modo de resumen, la solución final en Docker levanta los siguientes servicios:

- **Frontend (Node.js):** Con el código desarrollado en Node.js, HTML, CSS y JavaScript, para la parte visual a la que acceden los usuarios.
- **Backend (Flask):** Se encarga de la parte lógica de manejar las respuestas de los usuarios y de manejar el flujo de datos hacia la base de datos.
- **Base de datos (MariaDB):** únicamente se encarga del almacenamiento de la información y de modificarla o recuperarle en función de las peticiones del Backend.

En este servidor se realizaron pequeñas pruebas con usuario fantasma para comprobar el correcto funcionamiento con los usuarios del grupo Wakamola para corregir pequeños errores que pudieran alterar la experiencia de los usuarios. Tras 7 versiones, se pasó a las pruebas de usabilidad con un pequeño grupo de voluntarios.

7 Pruebas de usabilidad

Finalmente, para comprobar la funcionalidad del software en entorno de producción se realizó un despliegue mediante Docker en un servidor Ubuntu dentro de la red de la universidad. En esta versión de prueba únicamente tuvieron acceso los miembros del equipo Wakamola, junto a un pequeño grupo de 21 voluntarios.

De este grupo de voluntarios se recogieron los datos de las diferentes preguntas que posee la aplicación. Además, se les solicitó que cumplimentaran una serie de pruebas dedicadas a evaluar la usabilidad de la aplicación y la experiencia del usuario, con tal de tener lista de puntos a mejorar que sirvan de punto de partida para las futuras versiones de Wakamola.

Para ello se emplearon las encuestas:

- **Task Test**
- **System Usability Scale (SUS)**
- **User Experience Questionnaire (UEQ)**

7.1 Task Test

El Task Test consiste en solicitar a los usuarios que completen una serie de tares y que valoren diferentes parámetros de la realización de estas, como el tiempo requerido o la cantidad de errores cometidos. Las tareas que ser les pidió que valoraran fueron:

1. **Registrarse en la aplicación**
2. **Acceder con un usuario ya registrado en la aplicación**
3. **Interactuar con el menú raíz de la aplicación**
4. **Establecer una relación de amistad con otro usuario**
5. **Introducir los datos de la encuesta**

De cada una de estas tareas se les solicitó al acabarlas que respondieran a las siguientes preguntas:

1. *¿Has realizado la tarea?*: Una pregunta explicativa, solo hay que responder sí o no
2. *La tarea ha resultado:* junto a una escala de valoración del 1 (muy difícil) al 7 (muy fácil)
3. *El tiempo necesitado para realizar esta tarea ha sido:* junto a una escala de valoración del 1 (mucho) al 7 (poco).
4. *La cantidad de errores cometidos durante la realización de la tarea asciende a:* con una escala de 0 hasta 5 o más
5. *¿Has entendido todos los datos que has tenido que introducir?:* Una pregunta explicativa que solo requiere responder sí o no
6. *Comenta cualquier posible mejora del proceso:* Una sección de comentarios donde el usuario puede añadir sus sugerencias sobre la tarea.

Las respuestas a estas preguntas se evaluaron para comprobar si se deberían de replantear la forma en que se estaba realizando estas tareas y si se deberían de replantear los procesos.

7.2 System Usability Scale (SUS)

La usabilidad es la cualidad de la página web o programa informático que son fáciles de usar y, por ello, con los que usuario se siente cómodo y satisfecho utilizándolos.

El **System Usability Scale (SUS)** se trata de una herramienta utilizada para medir la usabilidad en una amplia variedad de productos y servicios, incluyendo hardware, software, dispositivos móviles, sitios web y aplicaciones. Fue creada por John Brooke en 1968 y consiste en un cuestionario de 10 preguntas con cinco posibles respuestas, baremadas desde "Totalmente en desacuerdo" con un 1 a "Muy de acuerdo" con un 5, en una escala que va del 10 al 50. A continuación, se aplica una serie de cálculos a las respuestas obtenidas para lograr que la escala vaya del 0 al 100 (Aranceta-Bartrina et al., 2005).

El SUS se ha vuelto un estándar dentro de la industria, dado que (Aranceta-Bartrina et al., 2005):

- Es muy fácil de escalar para administrar a los participantes de las pruebas
- Ofrece resultados fiables aún con pequeñas muestras de usuarios
- Permite diferenciar correctamente entre sistemas usables y no usables.

El cálculo de la puntuación del SUS se realiza de la siguiente manera ("How To Use The System Usability Scale (SUS) To Evaluate The Usability Of Your Website - Usability Geek", s. f.):

1. A cada una de las respuestas impares se le resta 1 a la puntuación valorada por el usuario. Así, a la valoración de las respuestas 1, 3, 5, 7 y 9 se les hace respuesta - 1.
2. Para cada una de las respuestas pares, se le ha de restar a 5 la respuesta, es decir, aplicar 5 - respuesta. Esto ocurre con las respuestas de las preguntas 2, 4, 6 y 8.
3. Sumar todos los nuevos valores obtenidos
4. Multiplicar por 2'5 a la suma total.
5. La puntuación final se compara con una escala.
    - Una puntuación inferior a 51 significa que la aplicación tiene una usabilidad pobre y debe de solucionarse pronto este problema.
    - Una puntuación inferior a 68 significa que la aplicación tiene una usabilidad decente, pero se debe de invertir tiempo en mejorarla.
    - Una puntación superior a 80'3 significa que a los usuarios les ha encantado la aplicación y que la recomendarían a sus amigos.

Así pues, la forma de interpretar los resultados es normalizando las puntuaciones para obtener percentiles, de forma que no deben de interpretarse las puntuaciones como porcentajes, aunque vayan del 1 al 100.

Como ejemplo práctico, si, por ejemplo, un usuario ha respondido:

$$Respuestas = [\mathbf{5}, 3, \mathbf{4}, 2, \mathbf{3}, 1, \mathbf{2}, 3, \mathbf{4}, 5]$$

La puntuación del SUS se calcula así:

1. Se divide el vector de las respuestas según su posición en respuestas pares e impares:

$$Respuestas\ impares = [5, 4, 3, 2, 4]$$

$$Respuestas\ pares = [3, 2, 1, 3, 5]$$

2. Se obtienen los coeficientes de las respuestas impares, restándole 1 a cada respuesta:

$$Coeficientes\ impares\ =\ [(5\ -1), (4\ -1), (3\ -1), (2\ -1), (4\ -1)]\ =\ [4, 3, 2, 1, 3]$$

3. Se obtienen los coeficientes de las respuestas pares, restándole a 5 cada respuesta:

$$Coeficientes\ pares\ =\ [\ (5\ -3), (5\ -2), (5\ -1), (5\ -3), (5\ -5)]\ =\ [2, 3, 4, 2, 0]$$

4. Se calcula el sumatorio de todos los coeficientes obtenidos:

$$\sum Coeficientes\ =\ (4\ +\ 3\ +2\ +1\ +3)\ +\ (2+3\ +4\ +2\ +0) =\ 24$$

5. Se multiplica el valor obtenido por 2'5, obteniendo así su puntuación final. En este caso:

$$Puntuación\ =\ 2'5\ \sum Coeficientes\ =\ 2'5\ \cdot\ 24\ =\ 60$$

6. Al ser la puntuación inferior a 68, según el usuario se debería de trabajar más en la usabilidad de la aplicación, pero aun así se trata de una aplicación decente

Para la evaluación de Wakamola se tomaron las preguntas y se adaptaron con el nombre de la aplicación de la página web oficial del SUS ("System Usability Scale (SUS) | Usability.gov", s. f.) para la aplicación de Wakamola. Aunque el lenguaje con el que se han escrito resulte poco común, estas preguntas se modificaron dado que son preguntas validadas. Las preguntas planteadas fueron:

1. Creo que usaría Wakamola frecuentemente

2. Encuentro que Wakamola es innecesariamente complejo

3. Creo que Wakamola fue fácil de usar

4. Creo que necesitaría ayuda de una persona con conocimientos técnicos para usar Wakamola

5. Las funciones de este Wakamola están bien integradas.

6. Creo que Wakamola es muy inconsistente.

7. Imagino que la mayoría de la gente aprendería a usar ese Wakamola de forma muy rápida.

8. Encuentro que Wakamola es muy difícil de usar.

9. Me siento confiado al usar Wakamola.

10. Necesité aprender muchas cosas antes de ser capaz de usar Wakamola.

Así, a los a los usuarios de las pruebas de Wakamola se les fueron presentadas estas preguntas tras finalizar todas las tareas de testeo en forma de un formulario. Tuvieron que marcar en una casilla del 1 al 5, según si sus reacciones si estaban "Muy en desacuerdo" o "Totalmente de acuerdo" con cada una de las preguntas planteadas y, posteriormente se valoraron los resultados obtenidos.

7.3 User Experience Questionnaire (UEQ)

El User Experience Questionnaire fue desarrollado en 2005 con el objetivo de obtener un enfoque analítico basado en los datos para medir la experiencia del usuario. Con este formulario se pretende medir la experiencia del usuario al utilizar la aplicación y poder hacer conjeturas sobre las áreas de mejora del producto. El UEQ muestra un patrón de 6 cualidades de experiencia de usuario medidas para un producto evaluado. A partir de este patrón es posible hacer al menos algunas suposiciones sobre dónde buscar las mejoras. Estas cualidades medidas son:

- **Atractivo:** Mide la impresión global del producto, grado de agrado por los usuarios
- **Perspicacia:** relacionado con la familiaridad del producto y facilidad de aprendizaje de sus funcionalidades
- **Eficiencia:** Mide el esfuerzo necesario por los usuarios para resolver las tareas
- **Dependencia:** Relacionado con el sentimiento de control de las interacciones con el programa
- **Estimulación:** Mide el grado de motivación y emoción que provoca el usar el producto
- **Novedad:** Mide los grados de innovación, creatividad y captación del interés de los usuarios por el programa

Esto términos, se agrupan dentro de la dimensión del atractivo (que es considerada la condición general a evaluar). Dentro del atractivo, las cualidades se agrupan según sean pragmáticos o hedónicos, como puede observarse en la tabla 6.

Tabla 6 Agrupación de las cualidades de la experiencia del usuario en el UEQ [51]

Atractivo	
Cualidades pragmáticas	**Cualidades hedónicas**
Perspicacia	Estimulación
Eficacia	Novedad
Dependencia	

Existen dos versiones del UEQ: Una larga y otra corta, llamada UEQ-S. La versión corta está diseñada para las ocasiones en las que el usuario:

- Desea abandonar rápidamente el sitio web en el que se encuentra
- Ha realizado otro cuestionario sobre experiencia de usuario previamente a la realización de esta encuesta
- Ha de probar múltiples versiones del mismo producto o múltiples productos en una misma sesión.

En este caso, se ha aplicado la versión del UEQ-S porque el usuario previamente ha rellenado otras encuestas de usabilidad y experiencia de usuario. De esta forma se evita estresar a los usuarios y que su situación emocional sesgue los resultados. La versión del UEQ-S consta de 8 ítems, 4 hedónicos y 4 pragmáticos, clasificados por parejas de ítems, donde se presentan primero los ítems pragmáticos y luego los hedónicos. Dentro de las parejas de ítems, se muestra primero lo ítems negativos y luego los positivos. Los parejas de términos utilizados puede observarse en la tabla 7.

Tabla 7 Parejas de términos negativos y positivos utilizados para el UEQ-S

Término negativo	Término positivo
Obstructivo	Impulsor de apoyo
Complicado	Fácil
Ineficiente	Eficiente
Confuso	Claro
Aburrido	Emocionante
No interesante	Interesante
Convencional	Original
Común	Novedoso

Para valorar cada uno de estos términos se utiliza una escala del 1 al 7, que luego se normaliza, para que la escala vaya del -3 al 3. El usuario para cada una de estas variables valora si la dimensión está más cerca del término negativo, asignándole un número más bajo a dicha cualidad o un número más alto, asignándole una cualidad positiva.

El UEQ puede utilizarse para:

- **Comparar la experiencia de usuario entre productos:** Así, puede valorarse las diferencias entre la versión actual de un producto y una nueva versión, utilizando una prueba t entre los resultados obtenidos para cada una de las cualidades, valorando si los cambios en cada una de las cualidades son estadísticamente significativos.
- **Testear si un producto tiene suficiente experiencia de usuario:** Es una forma de medir si el producto cumple con las expectativas generales sobre la experiencia de usuario. Estas expectativas de los usuarios están formadas por sus experiencias con otros productos que usan con frecuencia.

Para obtener una mejor imagen de la calidad de un producto, es necesario comparar la experiencia de usuario medida del producto con los resultados de otros productos establecidos, por ejemplo, de un conjunto de datos de referencia que contenga productos típicos bastante diferentes. (“User Experience Questionnaire (UEQ)”, s. f.)

La evaluación comparativa clasifica un producto en 5 categorías (en escala):

- ***Excelente:*** en el rango del 10% de los mejores resultados.
- ***Bueno**:* el 10% de los resultados del conjunto de datos de la evaluación comparativa son mejores y el 75% de los resultados son peores.
- ***Superior a la media:*** el 25% de los resultados del conjunto de datos de referencia son mejores que el resultado del producto evaluado y el 50% de los resultados son peores.
- ***Inferior a la media:*** El 50% de los resultados de la prueba de referencia son mejores que el resultado del producto evaluado, el 25% de los resultados son peores:
- ***Malo:*** En el rango del 25% de los peores resultados.

El propio formulario del UEQ dispone de una hoja de Excel donde ya están todos los cálculos introducidos. La única operación que hay que realizar es añadir los datos en la correspondiente hoja, además de mostrar las gráficas necesarias para evaluar los resultados de las valoraciones de los usuarios de la experiencia de los usuarios.

8 Resultados obtenidos de la realización de las pruebas

8.1 Aplicación de Wakamola

De todas estas respuestas se realizó un breve análisis exploratorio de los datos recogidos. En total, participaron en las pruebas de la aplicación un total de 21 personas. La muestra de los usuarios que participaron en la prueba se distribuye con un 61’9% de participantes hombres y un 38’1% de participantes mujeres, como se representa en la figura 5.

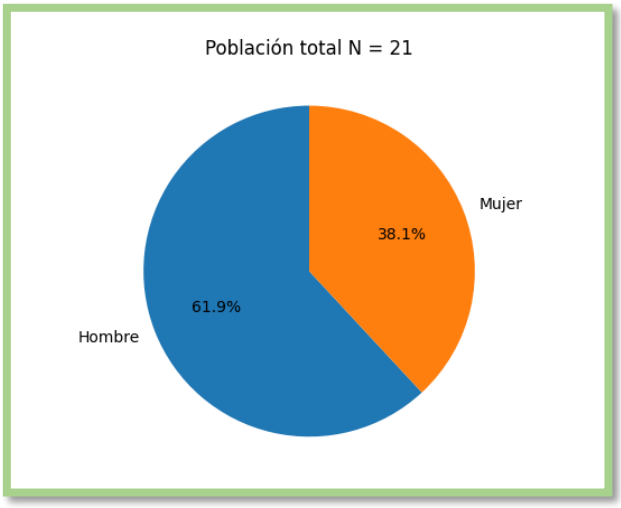

Figura 5 Distribución por géneros participantes de la prueba

Además, de esto datos un 47’6% se registró como soltero y un 52’4% se registró como en pareja o casado, como puede visualizarse en la figura 6.

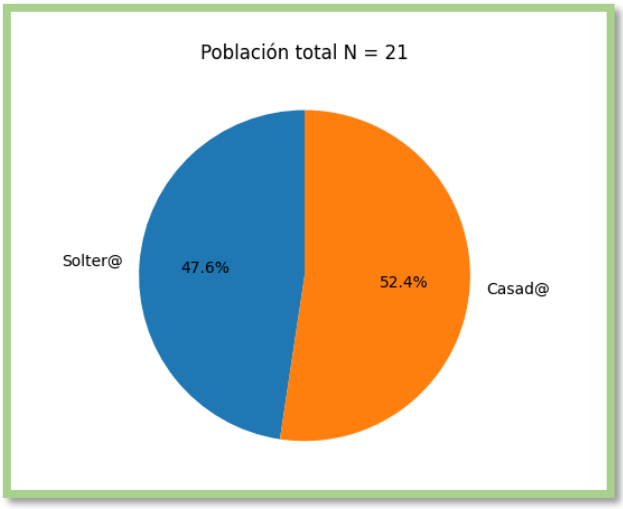

Figura 6 Distribución por estado social de los participantes de la prueba

La edad de los participantes se distribuye entre los 16 y 60 años con la mediana en los 26 años, presentando una media a los 34’33 años, y una desviación típica de 13’83 años. Se trata, como puede visualizar en el gráfico de cajas y bigotes de la figura 7, de una distribución claramente sesgada a la derecha, pues los datos parecen acumularse hacia las edades inferiores, con un coeficiente de curtosis de –1’08.

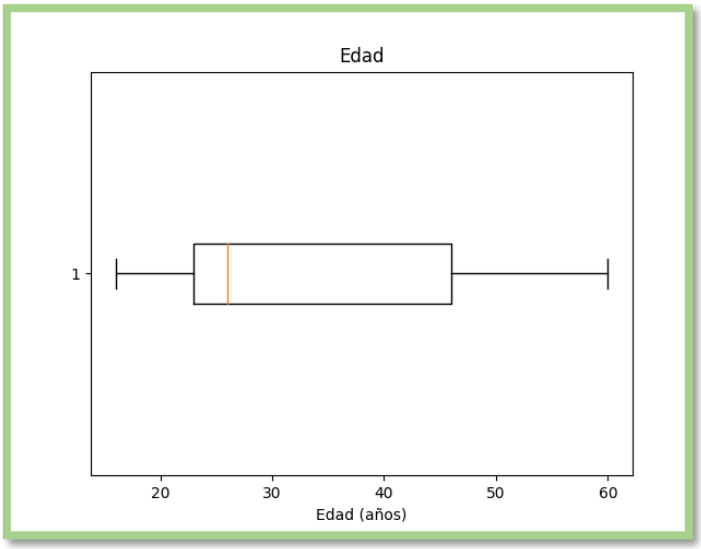

Figura 7 Distribución de las edades de los participantes de la prueba

En cuanto al peso de la muestra, esta tiene una media de 68’23 kg, con una mediana en 67 kg. El peso máximo registrado es de 93 kg y el peso mínimo ha sido de 38 kilogramos. En este caso, la distribución de los datos se ha hallado también sesgada un poco a derechas, con un coeficiente de curtosis de -0’78. Puede observarse en la figura 8.

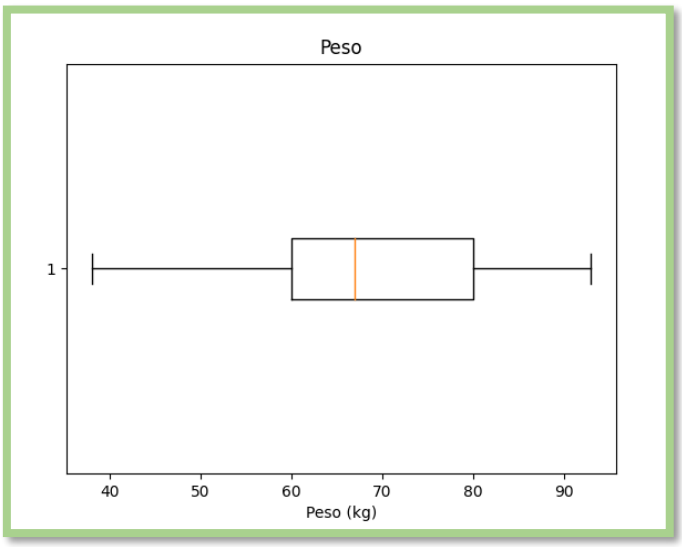

Figura 8 Distribución del peso de los participantes de la prueba

En cuanto a las alturas registradas, se ha hallado que tienen una media de 169’67 cm, con unos valores máximos y mínimos de 167 cm y 146 cm respectivamente. La mediana se encuentra en los 167 cm y la desviación típica es de 11’39 cm. Pueden visualizarse una representación de los resultados en la figura 9.

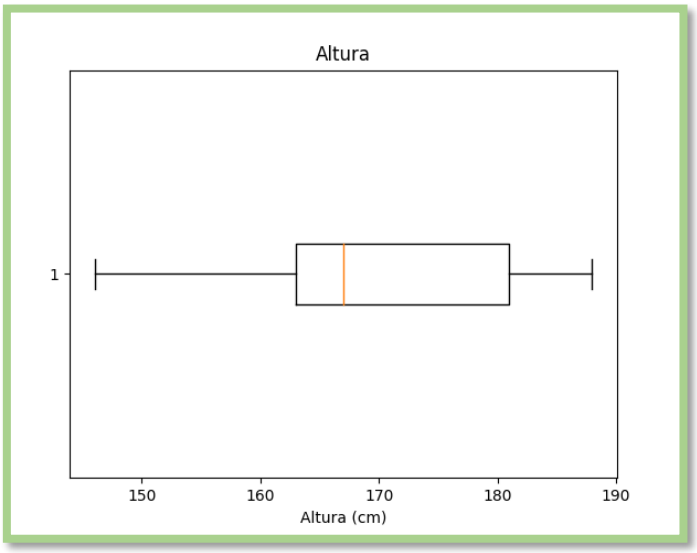

Figura 9 Distribución de la altura de los participantes de la prueba

Sobre el estado de salud de los usuarios, se recogieron datos sobre si padecía de hipertensión, diabetes, … pero de cada uno de estos grupos únicamente padecían estas patologías 1 o 2 usuarios en el caso de hipertensión.

Además, con el chatbot desarrollado se recogieron datos sobre las pautas nutricionales de 50 alimentos. En la figura 10 puede observarse los gráficos de cajas y bigotes obtenidos a partir de las respuestas de cada uno de este grupo alimentario. Cada una de las respuestas tiene una gran dispersión, y se debería de realizar un análisis exhaustivo de cada una de las variables como están relacionadas con otros parámetros estudiados, pero esta tarea escapa al alcance del presente trabajo. Cabe destacar que el alimento más consumido de media son las frutas, junto a las patatas, la carne y las verduras como las espinacas y que los alimentos menos consumidos de media son la mantequilla, los dulces, las bebidas destiladas, las galletas y otros aceites que no son de oliva. Esto apunta a que, a priori, los usuarios tienen unos buenos hábitos alimentarios según las recomendaciones de la dieta mediterránea (Neurosciences, 2009).

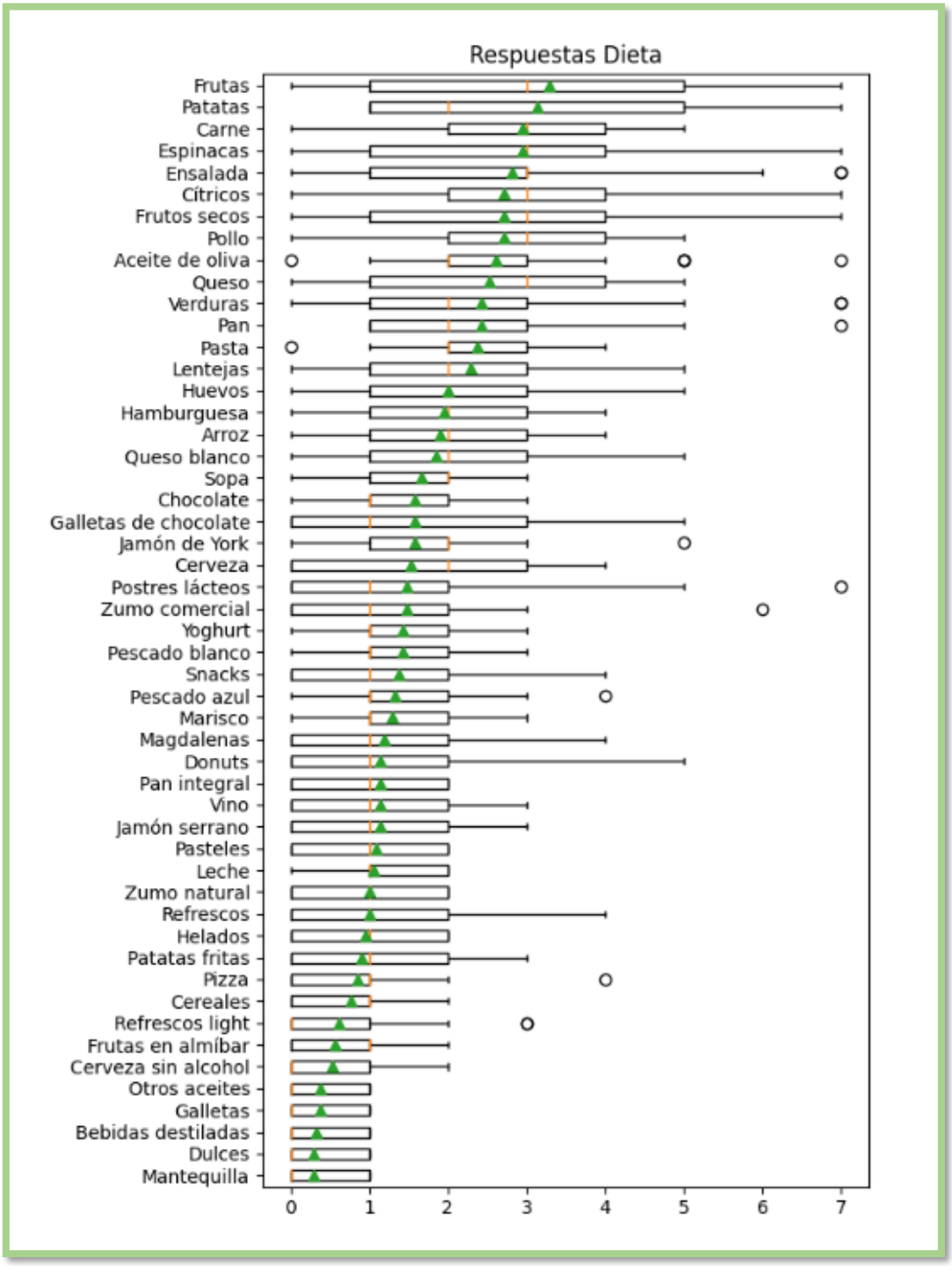

Figura 10 Distribuciones de las respuestas sobre la dieta de los participantes de la prueba

De la misma manera, con la aplicación se pudieron obtener datos sobre la actividad física de los participantes. Se obtuvo así que los usuarios realizan actividades moderadas una media de 2’29 días a la semana, actividades físicas moderadas una media de 3’10 días a la semana y salen a pasear una media de 3’24 veces por semana. Las desviaciones típicas muestrales para cada una de estas respuestas han sido del 2’33 días, 2’41 días y 2’25 días y las medianas, de 1 día, 2 días y 3 días respectivamente. Estos datos pueden visualizarse en la figura 11.

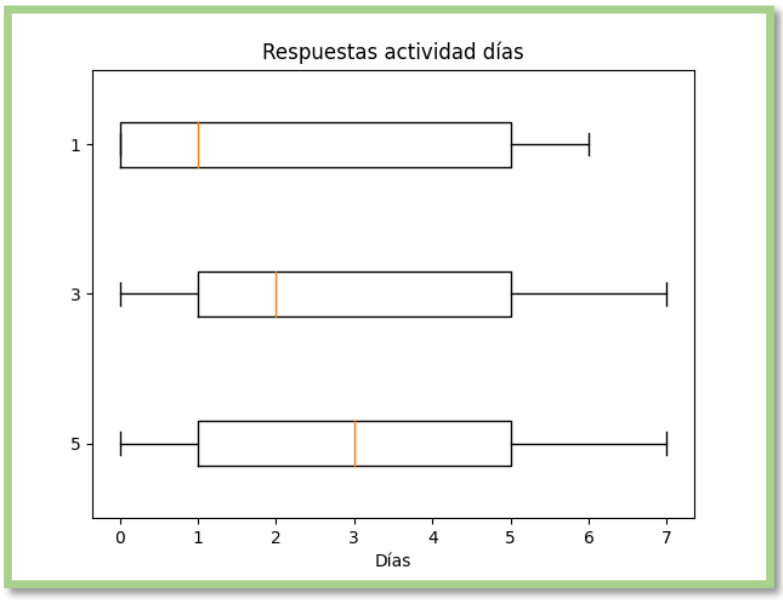

Figura 11 Distribuciones sobre las respuestas de los participantes de la actividad física en días

Por otro lado, la cantidad de minutos dedicados cada día a estas actividades ha variado bastante. Para actividades físicas moderadas ha sido una media de 45'76 minutos diarios, actividades moderadas 46'29 y actividades ligeras 43'67 minutos. Las desviaciones típicas han sido de 36'62, 34,29 y 30,40 minutos. La mediana ha sido de 60 minutos, 60 minutos y 40 minutos. Todo esto puede visualizarse en la figura 12.

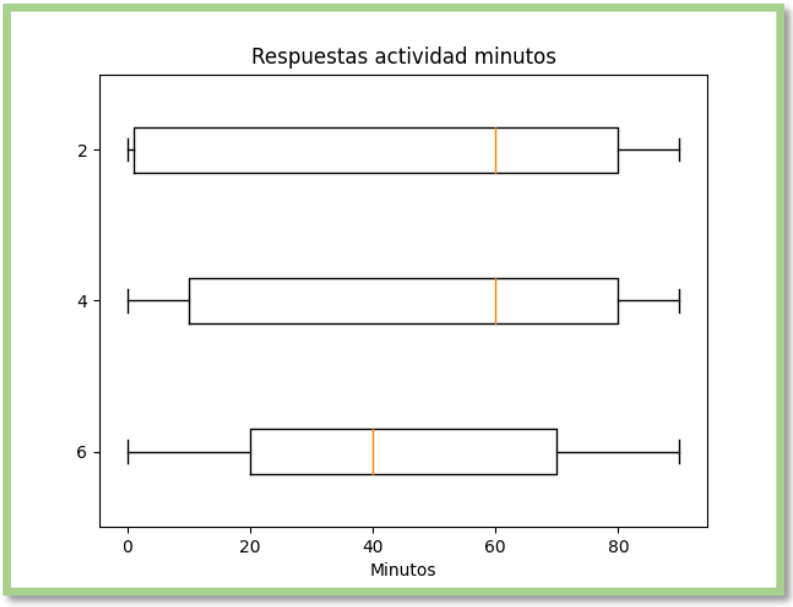

Figura 12 Distribuciones de las respuestas de los participantes de la prueba sobre la actividad física en minutos dedicados a cada tipo de actividad

Por último, se tomaron datos de las horas que pasan sentados los u usuarios. Se obtuvo que los usuarios pasan sentados una media de 6’05 horas diarias con una desviación típica de 3’73 horas. La mediana es de 6 horas al día, como se puede visualizar en la figura 13.

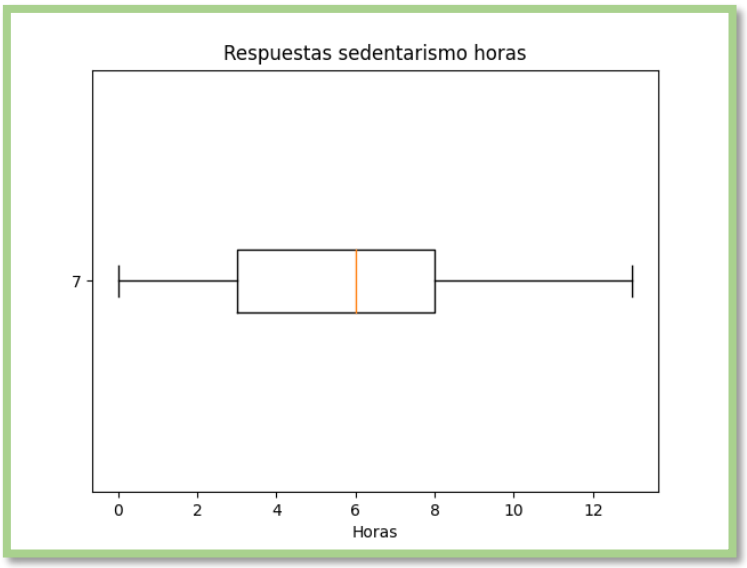

Figura 13 Distribución de las respuestas de los participantes sobre la cantidad de horas que permanecen sentados

En conclusión, se lograron adquirir una gran cantidad de datos útiles desde el punto de vista investigador, con los que poder trabajar y realizar investigaciones. Corresponde a futuros trabajos el encontrar estas relaciones entre las múltiples diferentes variables de estudio, tanto para realizar estudios poblacionales, como modelos predictivos.

8.2 Task Test

Todos los participantes de la aplicación se les explicó las tareas principales que tenían que realizar, para que pudieran valorarlas luego en el task test. Los 21 participantes de las pruebas voluntarias completaron esta parte de la prueba.

- **Tarea 1: Registrarse en la aplicación**

La primera tarea consistió en crearse un usuario en la aplicación. La mayoría de los usuarios valoraron con puntuaciones superiores a 6 tanto la facilidad de la tarea como el tiempo requerido para realizar la tarea, como puede observarse en la figura 14.

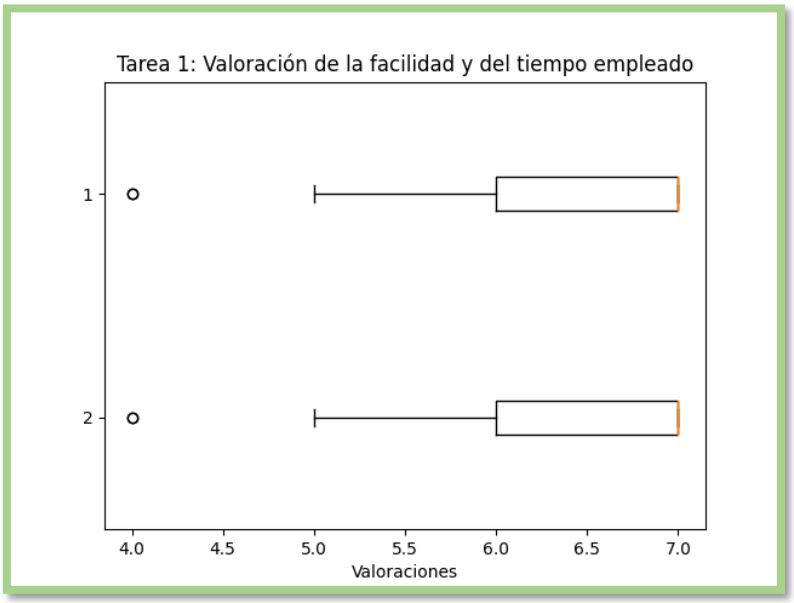

Figura 14 Resultados obtenidos de las valoraciones de los usuarios a la facilidad de la tarea (1) y al tiempo empleado en completarla (2)

La cantidad de errores que cometieron estos usuarios durante la realización de esta tarea también fue muy baja. El 66'7% de los usuarios no cometió errores en esta tarea, el 28'6% cometió un error y solo un 4'8% cometió 2 errores durante el desarrollo de esta tarea. Se considera pues, que, desde el punto de vista de los usuarios, la tarrea es intuitiva, rápida y fácil.

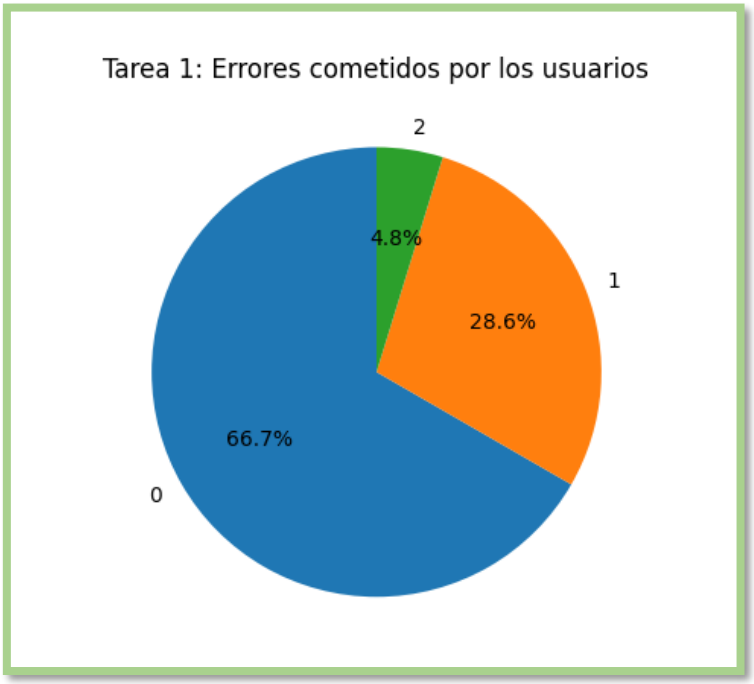

Figura 15 Representación de la cantidad reportada por los usuarios que han cometido al realizar la tarea

No obstante, un 42’9% de los usuarios no comprendían los datos que requerían la web, como puede visualizarse en la figura 16. Esto puede deberse a que la pregunta no esté bien formulada en el formulario del task test y da lugar a malinterpretaciones o a que se requiera un campo explicativo de cómo se deben de ingresar los datos.

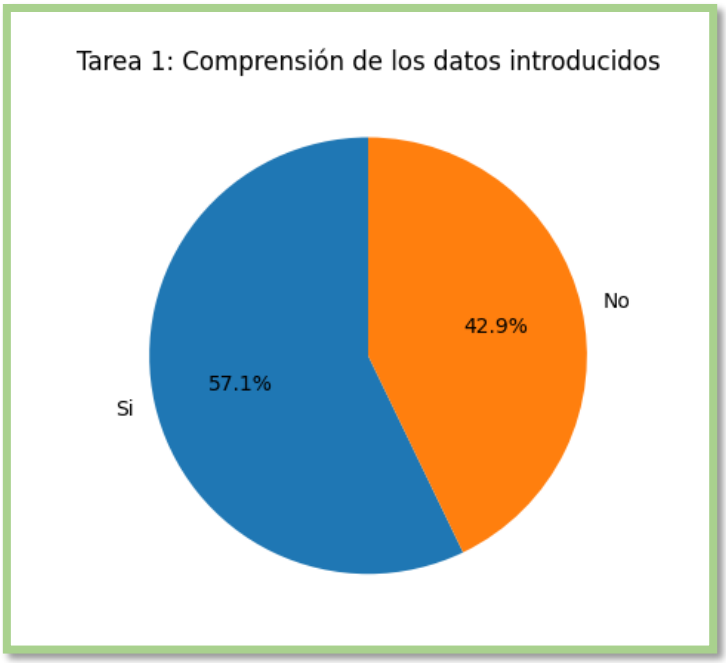

Figura 16 Representación de si los usuarios han comprendido los datos que han introducido para realizar la tarea (en este caso el nombre de usuario y la contraseña)

- **Tarea 2: Volver a acceder en la aplicación con un usuario ya registrado**

La segundad tarea consistía en cerrar la aplicación y volver a ingresar en ella, para comprobar que efectivamente el usuario ya estaba registrado en ella. De nuevo, la tarea fue valorada como fácil y que requiere poco tiempo su realización, como se desprende de la figura 17.

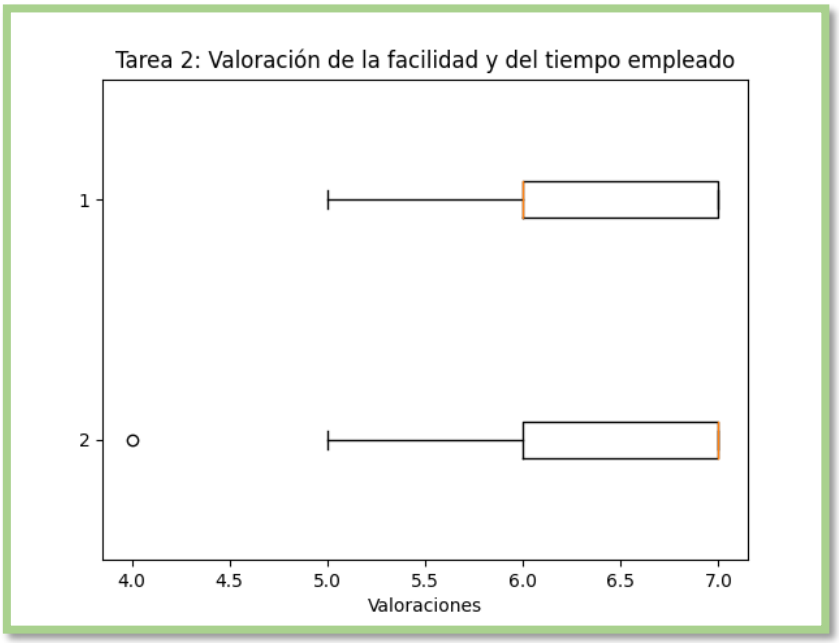

Figura 17 Resultados obtenidos de las valoraciones de los usuarios a la facilidad de la tarea (1) y al tiempo empleado en completarla (2)

En cuanto a la cantidad de fallos cometidos por los usuarios, cabe destacar que se produjeron más errores. Aunque el 66’7% de los usuarios logró entrar sin cometer errores, hubo un 23’8% que cometió un error, un 4’8% que cometió dos errores y otro 4’8% que cometió tres errores.

Simplificar esta tarea es una de las opciones que se deben de tener en cuenta para mejorar la aplicación en un futuro.

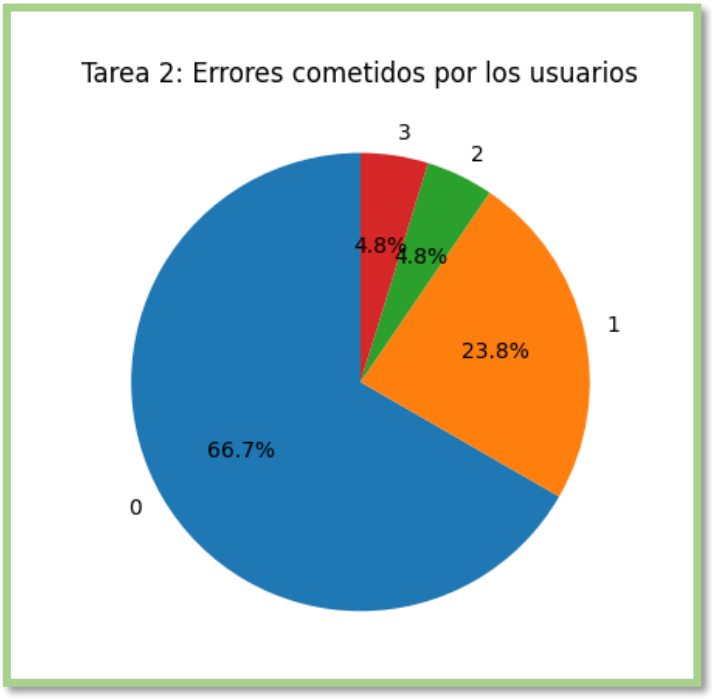

Figura 18 Representación de la cantidad reportada por los usuarios que han cometido al realizar la tarea

De nuevo, una mayoría de los usuarios comprendió los datos que había tenido que introducir (61’9%), pero aun así muchos usuarios no comprendieron los datos que tuvieron que introducir, como puede verse en la figura 19. De nuevo, puede deberse a no interpretar correctamente la pregunta o a que realmente hay que añadir aclaraciones sobre los datos.

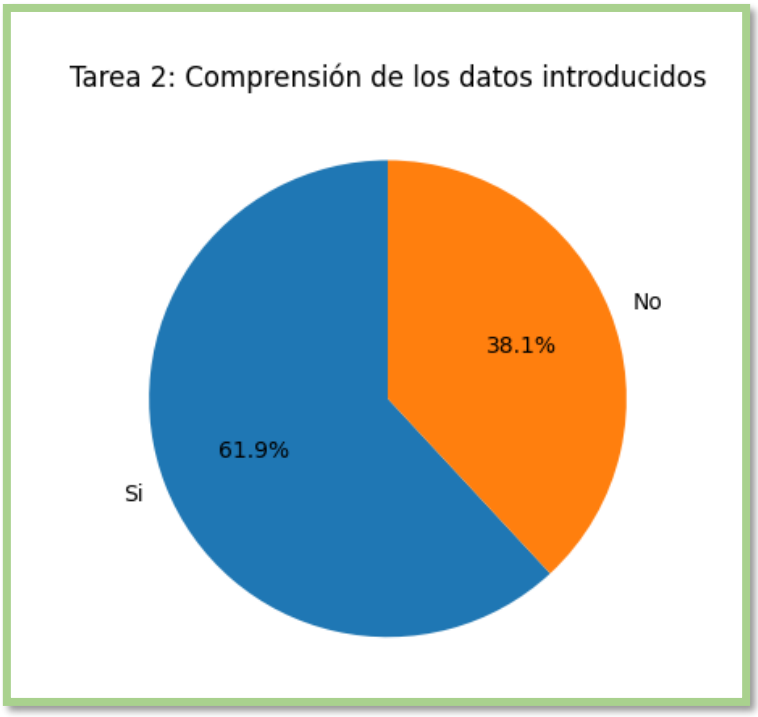

Figura 19 Representación de si los usuarios han comprendido los datos que han introducido para realizar la tarea (en este caso el nombre de usuario y la contraseña)

- **Tarea 3: Interactuar con el menú de inicio de la aplicación**

Esta tarea consistía en interactuar con el menú de inicio de la aplicación, donde debe de escogerse qué es lo que se desea responder o la actividad que se desea que realice el bot. En este caso, los usuarios la valoraron más negativamente, pues, como se puede ver en la figura 20, hay una mayor dispersión en los resultados de la facilidad de la tarea. En cambio, el tiempo empleado sigue valorándose como poco. Esto sugiere que se debe de trabajar en volver el menú de inicio de la aplicación más intuitivo para los usuarios.

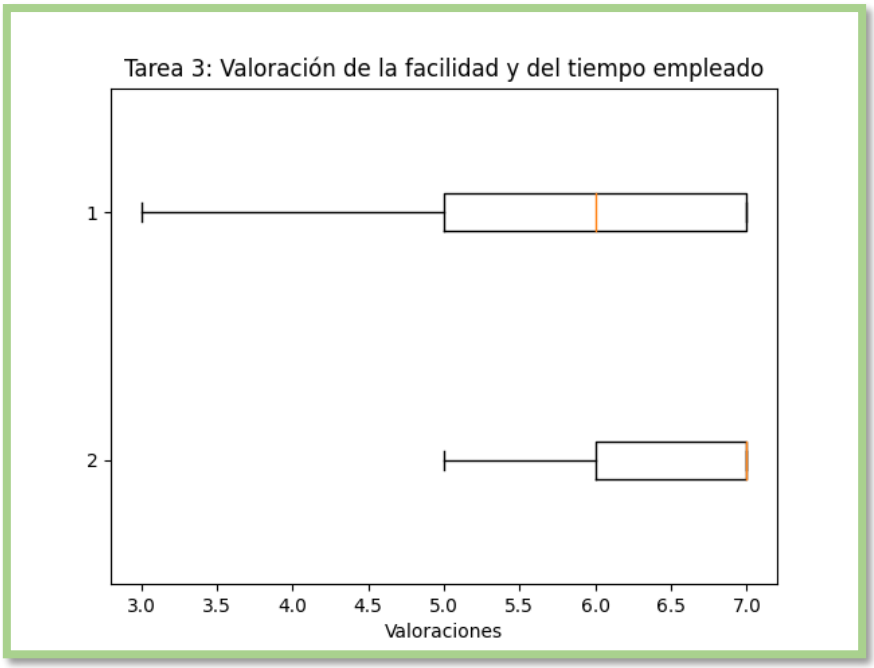

Figura 20 Resultados obtenidos de las valoraciones de los usuarios a la facilidad de la tarea (1) y al tiempo empleado en completarla (2)

En cuanto a los errores cometidos, se desprende de la figura 21 que un 71’4% de los usuarios no cometieron errores. Aun así, solo un 4’8% de los usuarios cometió un error, pero, sin embargo, un 14’3% de los usuarios cometió un error realizando la tarea y un 9’5% cometió 3 errores realizando la tarea. Así, pues estos datos confirman que el menú de inicio es una tarea que requiere de trabajo de mejora de su interacción con el usuario, volviéndolo más simple e intuitivo.

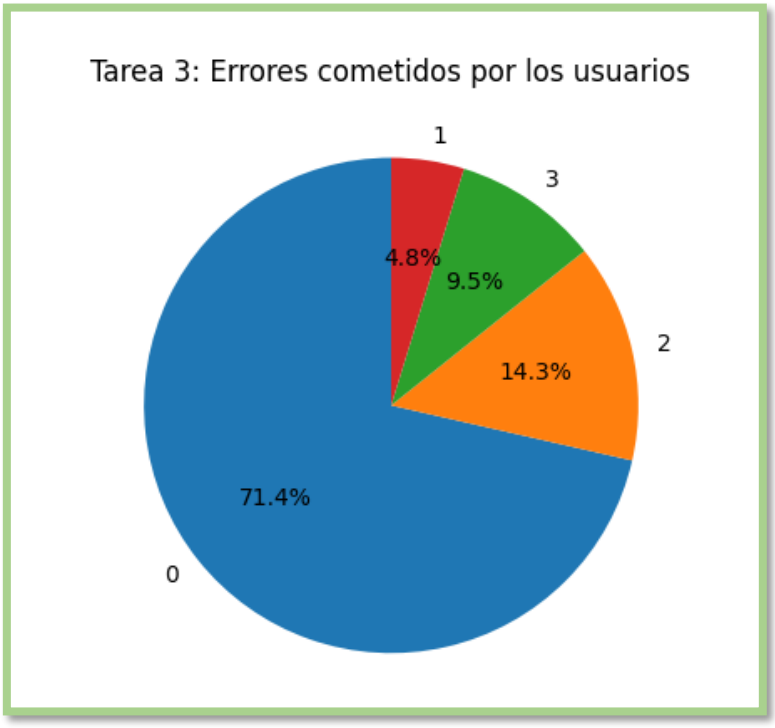

Figura 21 Representación de la cantidad reportada por los usuarios que han cometido al realizar la tarea

En este caso, dado que los usuarios no tenían que introducir datos, simplemente hacer clic sobre los iconos que mostraba la aplicación, no se realizó una evaluación de la comprensión de los datos introducidos.

- **Tarea 4: Establecer una relación de amistad con otro usuario**

La penúltima tarea consistía en establecer amistad entre los usuarios, bien entre ellos si la realizaban en grupo, o bien a un usuario predefinido si realizaban la tara de forma individual. Como se desprende de la figura 22, la mayoría de los usuarios consideró la tarea como fácil y que requería poco tiempo. No obstante, existen en ambos casos un usuario que las valora negativamente. Este caso puede ser que le supusiera alguna complicación el realizar la tarea, de ahí que la valorara negativamente.

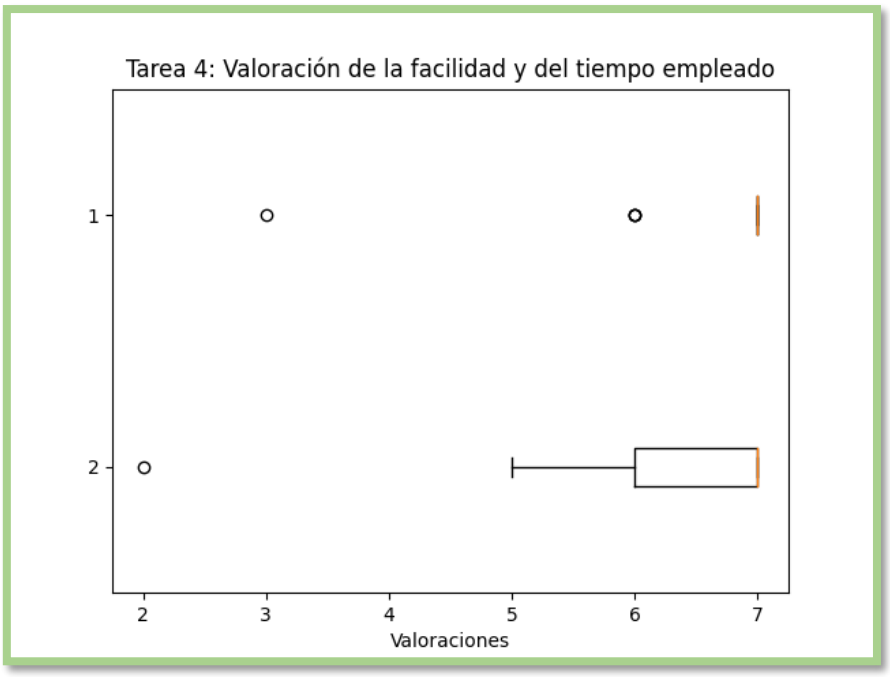

Figura 22 Resultados obtenidos de las valoraciones de los usuarios a la facilidad de la tarea (1) y al tiempo empleado en completarla (2)

De la figura 25, en cambio, se observa que, aunque un 57’1% realizó la tarea sin cometer errores, un 14’3% cometió un error, un 19’0% cometió dos errores y un 9’5% de los usuarios cometieron 5 errores o más durante el desarrollo de la tarea. Por lo tanto, se desprende que hay que invertir tiempo en rediseñar el proceso de establecer relaciones entre los usuarios para volverlo más simple e intuitivo.

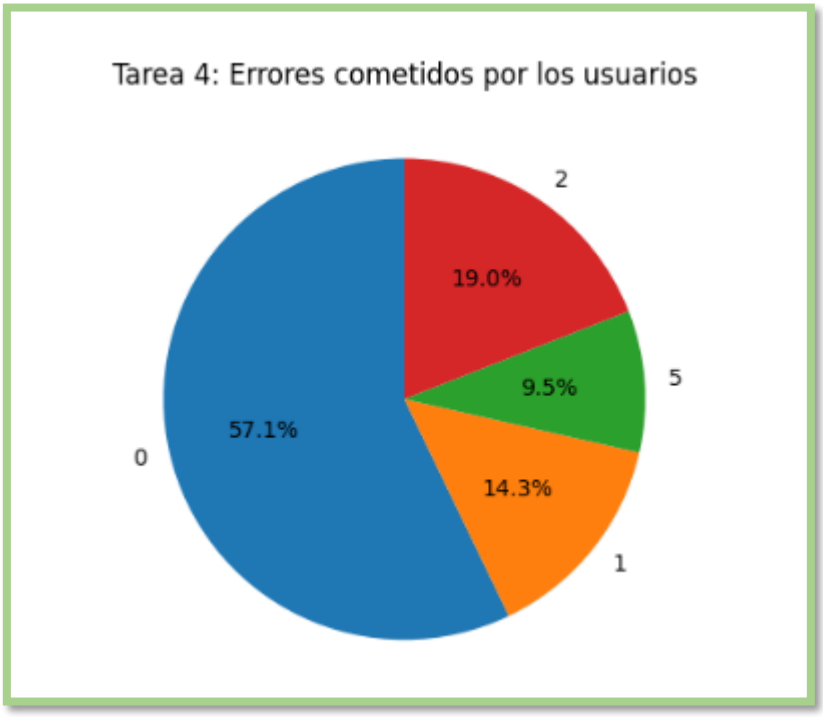

Figura 23 Representación de la cantidad reportada por los usuarios que han cometido al realizar la tarea

De esta tarea, solo un 28'6% de los usuarios no comprendió cuáles fueron los datos que tuvo que introducir, por lo que la familiaridad con los procesos de otras aplicaciones similares para establecer amistad hace que sea una tarea que no requiere explicación a los usuarios.

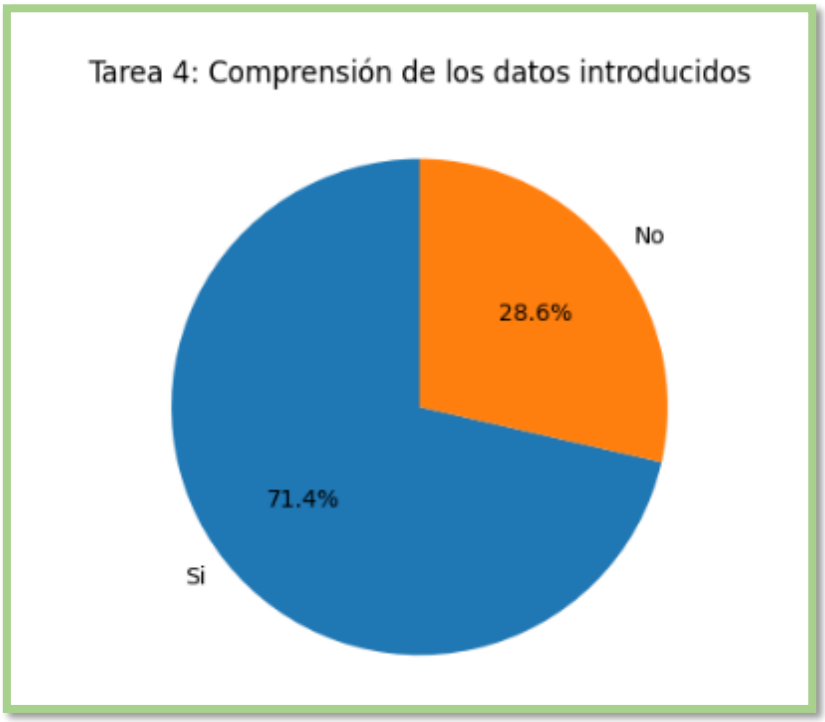

Figura 24 Representación de si los usuarios han comprendido los datos que han introducido para realizar la tarea (en este caso el nombre de usuario y la contraseña)

- **Tarea 5: Probar todos los campos de la aplicación Wakamola y rellenar toda la información**

Por último, se les solicitó a los usuarios que probaran a rellenar todos los datos en la aplicación y valoraran esta experiencia. Fue una forma de garantizar que acabaran todas las encuestas. Esta tarea fue valorada por la mayoría como fácil, pero, sin embargo, el tiempo necesario para realizarla fue valorado negativamente, con la mayoría de las puntuaciones cercanas al 1. Esto sugiere que se debería de trabajar en acortar las encuestas de Wakamola, principalmente la encuesta de alimentación que se trata, de la más larga.

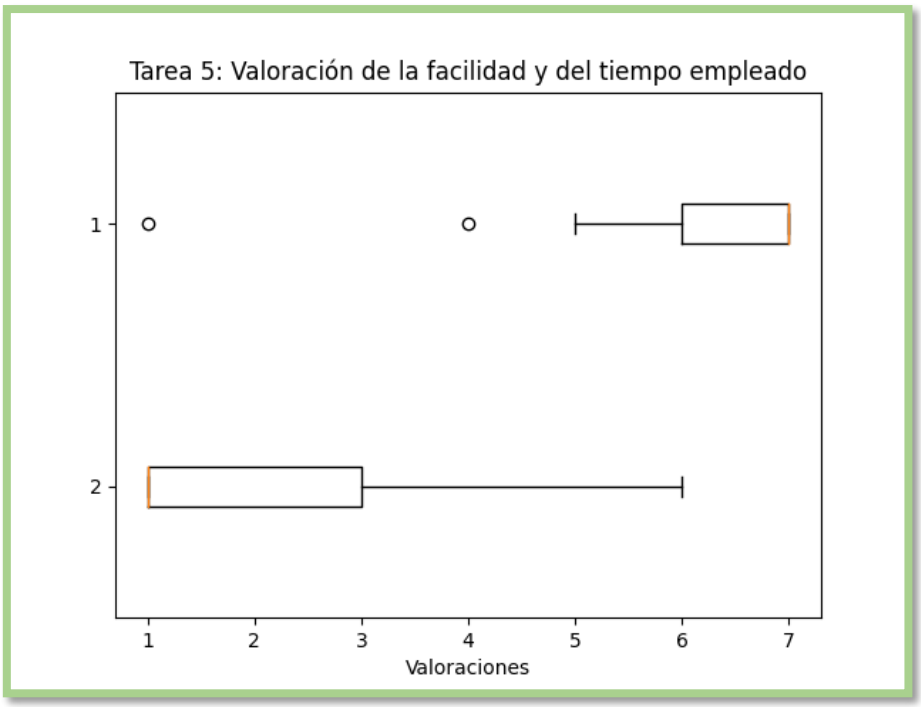

Figura 25 Resultados obtenidos de las valoraciones de los usuarios a la facilidad de la tarea (1) y al tiempo empleado en completarla (2)

La cantidad de errores cometidos en esta tarea es muy dispar. Mientras que la mayoría (un 66’7% de los usuarios no cometió ningún error, como se puede ver en la figura 26, hubo un 9’5% que cometió un error rellenando las encuestas. Un 4’8% cometió dos errores, un 14’3% cometió tres errores y un 4’8% cometió 4 errores durante el rellenado de la encuesta, de forma que habían introducido un valor que no querían introducir. Esto sugiere que se debe hacer más intuitiva la forma de responder a las preguntas, por lo que se debería de plantear otro diseño o permitir la edición de las respuestas de los usuarios para no tener que volverá repetir una parte de la encuesta debido a un error de escritura.

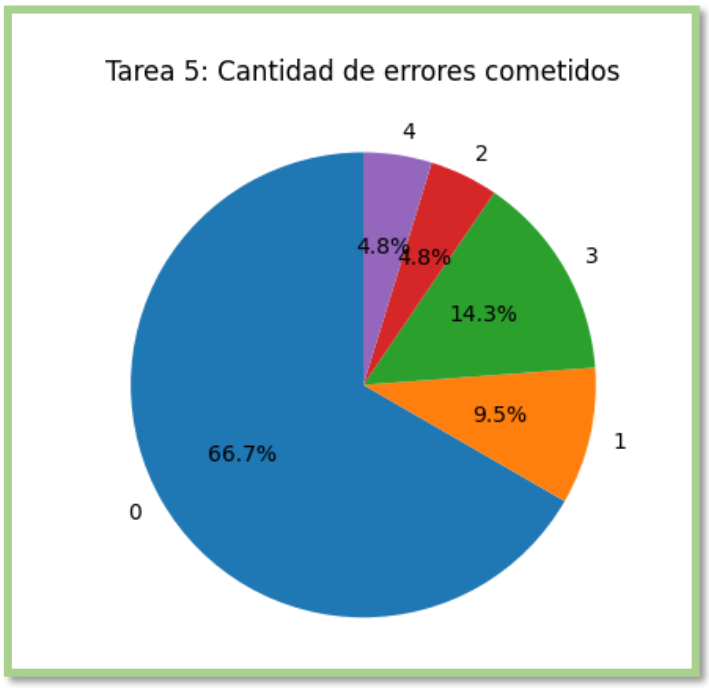

Figura 26 Representación de la cantidad reportada por los usuarios que han cometido al realizar la tarea

En este caso, un 57'1% de los usuarios no comprendió los datos que tuvo que introducir, como se desprende de la figura 27. Esto sugiere que se debería de trabajar en la claridad de las preguntas de la encuesta.

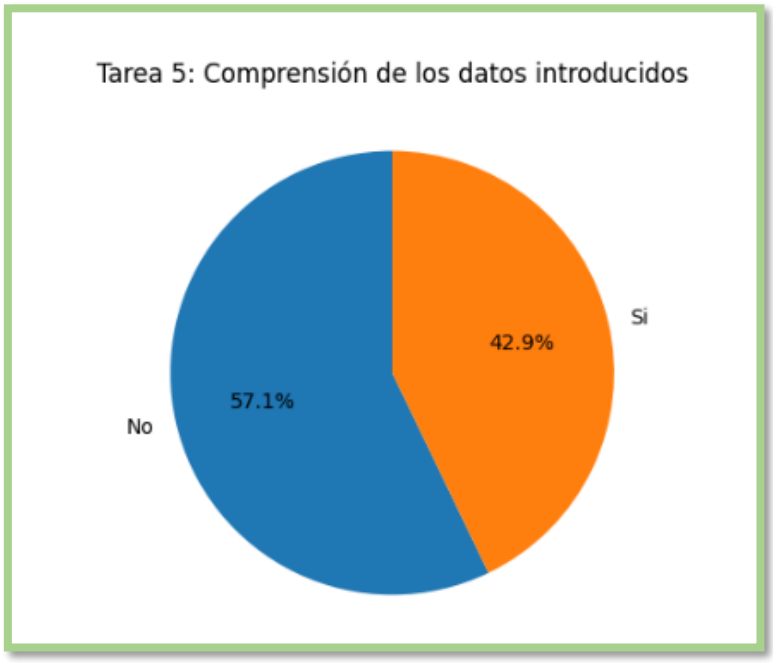

Figura 27 Representación de si los usuarios han comprendido los datos que han introducido para realizar la tarea (en este caso las entradas en las encuestas)

8.3 SUS

Tras completar todas las tareas de la aplicación se les pidió que rellenaran un formulario en Google Formular. Las respuestas obtenidas se han resumido en el contenido de la figura 28.

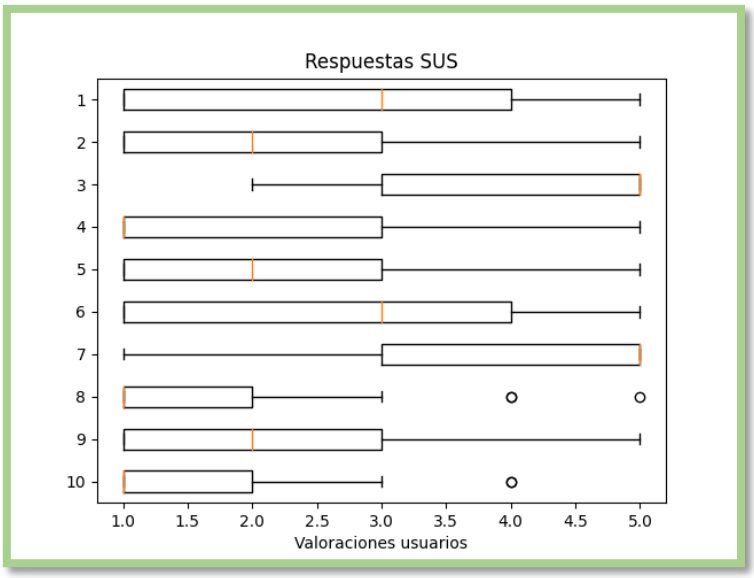

Figura 28 Distribuciones de las respuestas de los usuarios a las preguntas del System Usability Scale

De estas valoraciones, se obtuvieron las puntuaciones de los usuarios, con lo que se obtuvieron los resultados de la figura 29.

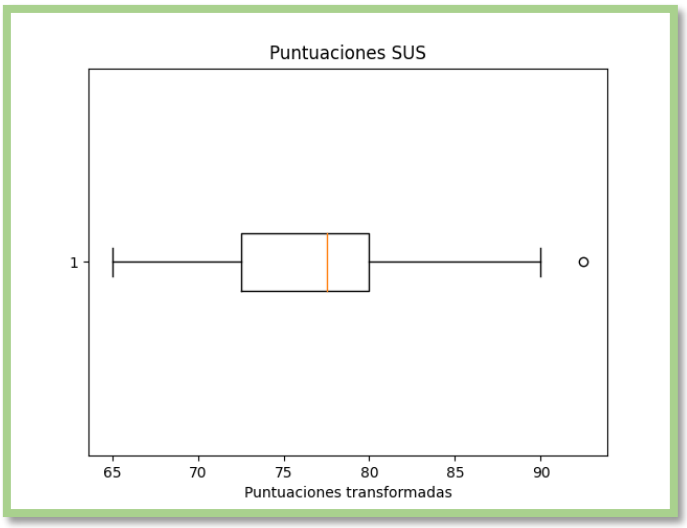

Figura 29 Distribución de las puntuaciones de los usuarios a la usabilidad de Wakamola web

Se obtuvo pues una puntuación media de 76’67, con una puntuaciones mínima y máxima de 65 y 92’5 puntos, respectivamente. La mediana se encuentra en los 77’5 puntos y la desviación estándar muestral es de 7’08 puntos. Así pues, se puede decir que las valoraciones de los usuarios, en general, se encuentran por encima de los ligeramente por debajo de los 80’3 puntos necesarios para decir que la aplicación tiene una usabilidad aceptable, pero mejorable para los usuarios de prueba. Así pues, se debe decidir si se quiere invertir más tiempo y recursos en mejorar la usabilidad de la aplicación.

8.4 UEQ

En último lugar, se les solicitó a los usuarios que completaran las preguntas correspondientes al User Experience Questionnaire. Puede observarse la representación de las respuestas obtenidas en la figura 30. Estas respuestas fueron introducidas en la herramienta que ofrece el propio UEQ para interpretar los datos.

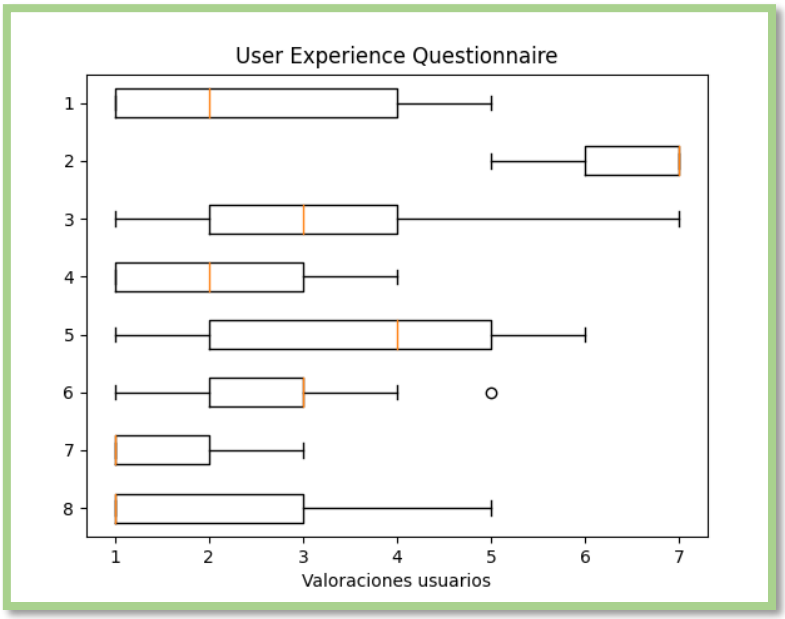

Figura 30 Representación de las respuestas al User Experience Questionnaire

La primera tabla que ofrece el UEQ es la tabla que agrupa las respuestas en función de Las 8 características. Obtienen medidas descriptivas como la media, la varianza, la desviación típica, además de los términos que se han utilizado en inglés.

De la figura 31 se desprende que los usuarios no han tenido, en general, una buena impresión de Wakamola. Tanto sus cualidades pragmáticas, hedónicas, como atractivo general no han sido valorados positivamente. Estos resultados abren el frente a una nueva orientación del trabajo, mejorar la experiencia del usuario que utiliza Wakamola, volviéndolo más innovador, práctico y dependiente.

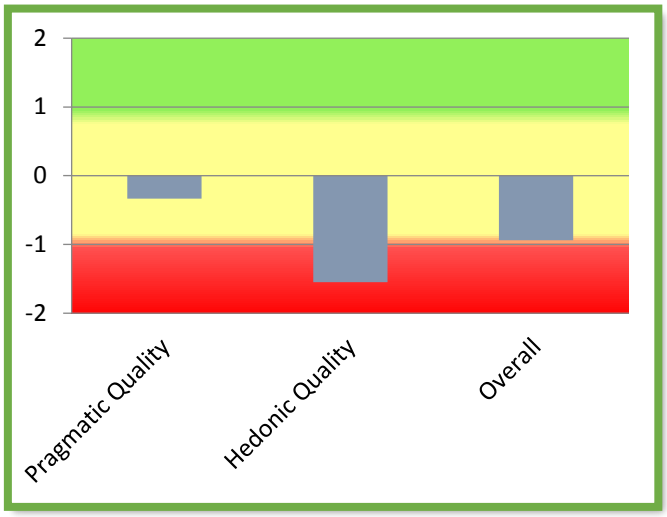

Figura 31 Representación de las valoraciones de los usuarios de las cualidades pragmáticas, hedónicas y el atractivo general de Wakamola web

El UEQ, además permite comparar los resultados obtenidos con la de otros productos que también han realizado el UEQ. Así, pues al compáralo con estos, se obtiene que Wakamola web posee unas malas cualidades pragmáticas (perspicacia y evidencia) y, sobre todo, hedónicas (dependencia, estimulación y novedad). Así, el atractivo general de la aplicación desde el punto de vista de la experiencia del usuario es muy bajo.

En conclusión, los esfuerzos actuales en la aplicación deben de orientarse no hacia mejorar la funcionalidad de Wakamola web, sino en lograr que sea más atractiva para los usuarios, para lograr una mayor comunidad de usuarios.

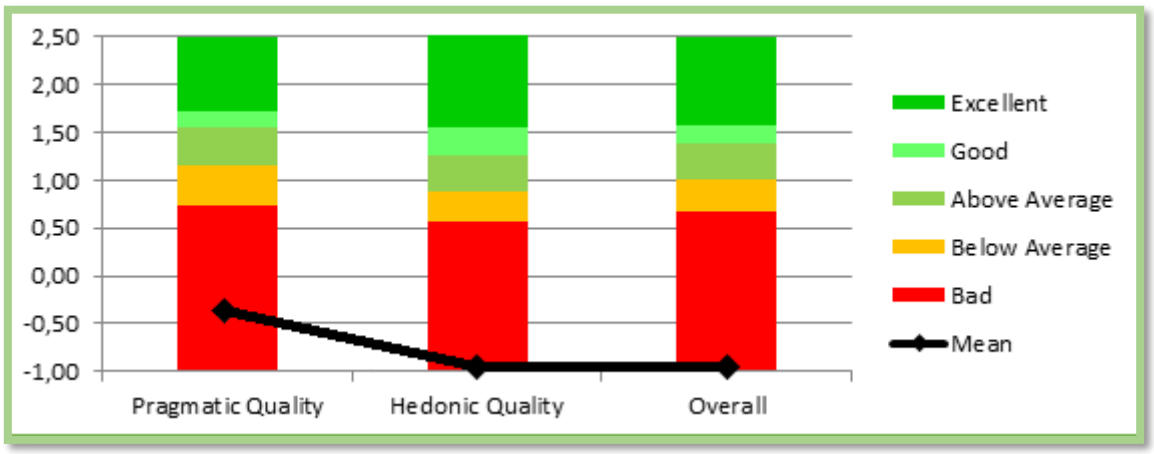

Figura 32 Representación de la comparación de los valores de las cualidades generales del UEQ con otros productos de la base de datos del UEQ

9 Conclusiones

- Se ha logrado crear una plataforma de visualización de Wakamola en un sitio web que tiene un potencial mayor alcance que la plataforma Telegram.
- Se ha logrado mantener la estética y la temática de Wakamola web, con colores verdes y el personaje original, aunque según los usuarios esta estética es mejorable.
- Se han recopilado los datos introducidos por los usuarios e interpretarlos en bajo grado y almacenarlos de forma permanente en una base de datos.
- Se ha instalado una serie de comprobaciones a los datos introducidos para garantizar que los datos obtenidos son correctos.
- Se ha creado un sistema de peticiones de amistad para que los usuarios puedan establecer vínculos entre sí.
- Se ha comprobado el correcto funcionamiento de la aplicación en el entorno de producción con pruebas, además de utilizando el software Docker que garantiza su despliegue incluso si el servidor físico actual se modifica.
- El programa al estar virtualizado y usar una API Rest estandarizada y documentada, puede ser utilizado por otros posibles futuros servicios que puedan surgir.
- Se ha evaluado correctamente la experiencia de los usuarios al utilizar Wakamola y se han determinado las diferentes futuras líneas de trabajo de Wakamola: principalmente, el atractivo general de la aplicación reducir los tiempos de realización de la encuesta.

10 Líneas futuras

- Mejorar la seguridad de la aplicación, almacenando en la base de datos los nombres de usuarios ya codificados e imponiendo restricciones a las contraseñas.
- Sustituir las conversaciones preestablecidas por un modelo entrenando de conversaciones, mejorando la interactividad con el usuario.
- Dado que en la página web no se pueden implementar notificaciones, desarrollar un sistema que envía recordatorios bien al móvil, bien al correo electrónico.
- Añadir utilidad a la aplicación, dado que únicamente se limita a mostrar una puntuación y la de los semejantes, como dar consejos personalizados para mejorar la salud del usuario, por lo que se requeriría ponerse en contacto con algún equipo médico para elaborarlos y desarrollarlos en profundidad.
- Añadir un sistema de retos y premios para aumentar el uso de la aplicación, como de apariencias para el personaje de la aplicación, insignias o competitividad para conseguir la mayor red y lograr estar entre las mejores puntuaciones dentro de la red.
- Aumentar la ciberseguridad aplicando encriptación a todos los datos de la base de datos, dado que se tratan de datos de alta privacidad (datos de la situación médica).
- Desarrollar una aplicación propia para cada plataforma de sistema operativo, principalmente para Android e iOS para lograr cubrir el problema de las notificaciones y facilitar la monitorización.
- Aplicar diferentes normativas al producto, como la ISO 27000 aplicable al almacenamiento de la información o la ISO 27999 sobre buenas prácticas de seguridad de la información en el entorno sanitario (es la versión para el entorno sanitario de la ISO/IEC 27002). (“Serie 27k”, s. f.)

11 Bibliografía

¿Qué es Backend y Frontend? - Descubre Comunicación. (s. f.). Recuperado 20 de febrero de 2021, de https://descubrecomunicacion.com/que-es-backend-y-frontend/

¿Qué es Bootstrap? - Una Guía Completa para Principiantes. (s. f.). Recuperado 8 de noviembre de 2020, de https://www.hostinger.es/tutoriales/que-es-bootstrap/

¿Qué es un socket? (s. f.). Recuperado 2 de abril de 2021, de https://www.speedcheck.org/es/wiki/socket/

Agencia de Chatbots | WhatsApp | Facebook Messenger | Instagram | Alexa Skills. (s. f.). Recuperado 20 de febrero de 2021, de https://chatbotchocolate.com/

Apcelent. (2018). How to Create a Facebook Messenger Bot with Python Flask. Recuperado 1 de noviembre de 2020, de https://dev.to/apcelent/how-to-create-a-facebook-messenger-bot-with-python-flask-50j2

API REST: qué es y cuáles son sus ventajas en el desarrollo de proyectos. (s. f.). Recuperado 8 de marzo de 2021, de https://www.bbvaapimarket.com/es/mundo-api/api-rest-que-es-y-cuales-son-sus-ventajas-en-el-desarrollo-de-proyectos/

Aranceta-Bartrina, J., Serra-Majem, L., Foz-Sala, M., Moreno-Esteban, B., Colaborativo SEEDO, G., Barbany, M., … Aranceta, J. (2005). *Prevalencia de obesidad en España * El Grupo Colaborativo Español para el Estudio de la Obesidad está formado por los siguientes miembros: J. Aranceta*. *460 Med Clin (Barc)125(12)* (Vol. 125). Recuperado de http://www.doyma.es

Arquitectura de una API REST · Desarrollo de aplicaciones web. (s. f.). Recuperado 30 de marzo de 2021, de https://juanda.gitbooks.io/webapps/content/api/arquitectura-api-rest.html

Asensio-Cuesta, S., Blanes-Selva, V., Conejero, J. A., Frigola, A., Portolés, M. G., Merino-Torres, J. F., … García-Gómez, J. M. (2021). A User-Centered Chatbot (Wakamola) to Collect Linked Data in Population Networks to Support Studies of Overweight and Obesity Causes: Design and Pilot Study. *JMIR Medical Informatics*, *9*(4), e17503. https://doi.org/10.2196/17503

Chat Marketing Platform & Chatbot Pricing. (s. f.). Recuperado 8 de noviembre de 2020, de https://mobilemonkey.com/prices

Chatbot as a great addition to fitness apps — Jasoren. (s. f.). Recuperado 15 de marzo de 2021, de https://jasoren.com/chatbot-techniques-that-can-increase-customer-engagement-for-health-and-fitness-brands/

Cómo diseñar un chatbot en Instagram | by Nutella Developer | Planeta Chatbot : todo sobre los Chat bots, Voice apps e Inteligencia Artificial. (s. f.). Recuperado 8 de noviembre de 2020, de https://planetachatbot.com/como-disenar-un-chatbot-en-instagram-478c755288c0

CSS | MDN. (2020). Recuperado 2 de noviembre de 2020, de https://developer.mozilla.org/es/docs/Web/CSS

Del Ra, W. (2015). Sams Teach Yourself Node.js in 24 Hours by George Ornbo. *ACM SIGSOFT Software Engineering Notes*. https://doi.org/10.1145/2830719.2830737

Di Cesare, M., Bentham, J., Stevens, G. A., Zhou, B., Danaei, G., Lu, Y., … Cisneros, J. Z. (2016).

Trends in adult body-mass index in 200 countries from 1975 to 2014: A pooled analysis of 1698 population-based measurement studies with 19.2 million participants. *The Lancet*, *387*(10026), 1377-1396. https://doi.org/10.1016/S0140-6736(16)30054-X

Diet Planner Chatbot - Applications, Benefits and Bot Template | Makerobos.com. (s. f.). Recuperado 5 de enero de 2021, de https://www.makerobos.com/chatbot-templates/healthcare-chatbot/diet-planner-chatbot

Diseño de chatbots en páginas web | by Planeta Chatbot | Planeta Chatbot : todo sobre los Chat bots, Voice apps e Inteligencia Artificial. (s. f.). Recuperado 10 de noviembre de 2020, de https://planetachatbot.com/diseño-chatbots-paginas-web-657074421927

Domingo Múñoz, J. (2017). Qué es Flask y ventajas que ofrece. Recuperado 14 de febrero de 2021, de https://openwebinars.net/blog/que-es-flask/

Generalidades del protocolo HTTP - HTTP | MDN. (s. f.). Recuperado 29 de marzo de 2021, de https://developer.mozilla.org/es/docs/Web/HTTP/Overview

Getting Started With morgan: the Node.js Logger Middleware | DigitalOcean. (s. f.). Recuperado 8 de noviembre de 2020, de https://www.digitalocean.com/community/tutorials/nodejs-getting-started-morgan

GitHub - line/line-bot-sdk-python: LINE Messaging API SDK for Python. (s. f.). Recuperado 8 de noviembre de 2020, de https://github.com/line/line-bot-sdk-python

How to Build an Instagram Chatbot (using Machine Learning) - Markets and Data - YouTube. (s. f.). Recuperado 8 de noviembre de 2020, de https://www.youtube.com/watch?v=Abow8NFd4UE

How To Use The System Usability Scale (SUS) To Evaluate The Usability Of Your Website - Usability Geek. (s. f.). Recuperado 20 de abril de 2021, de https://usabilitygeek.com/how-to-use-the-system-usability-scale-sus-to-evaluate-the-usability-of-your-website/

HTML: Lenguaje de etiquetas de hipertexto | MDN. (2020). Recuperado 2 de noviembre de 2020, de https://developer.mozilla.org/es/docs/Web/HTML

Hubspot. (2019). Cómo crear un bot para WhatsApp. Recuperado 1 de noviembre de 2020, de https://blog.hubspot.es/marketing/WhatsApp-bot

IAB Spain. (2020). ESTUDIO REDES SOCIALES 2020. Recuperado 19 de septiembre de 2020, de https://iabspain.es/sin-acceso/download-id/39688/

Introducción a jQuery. (s. f.). Recuperado 8 de noviembre de 2020, de https://desarrolloweb.com/articulos/introduccion-jquery.html

JavaScript | MDN. (2020). Recuperado 2 de noviembre de 2020, de https://developer.mozilla.org/es/docs/Web/JavaScript

Jolt.ai - The best fitness accountability partner you've ever had | Product Hunt. (s. f.). Recuperado 16 de marzo de 2021, de https://www.producthunt.com/posts/jolt-ai

JSON. (s. f.). Recuperado 8 de marzo de 2021, de https://www.json.org/json-es.html

Lark Programs | A Whole Person Approach to Care | Lark Health. (s. f.). Recuperado 5 de enero de 2021, de https://www.lark.com/programs/#

LINE se une a la moda de los bots abriendo su API para desarrolladores. (s. f.). Recuperado 8 de noviembre de 2020, de https://www.xatakandroid.com/comunicacion-y-mensajeria/line-

se-une-a-la-moda-de-los-bots-abriendo-su-api-para-desarrolladores

MariaDB Foundation - MariaDB.org. (s. f.). Recuperado 7 de marzo de 2021, de https://mariadb.org/

Martínez Molera, L. (s. f.). Cómo crear un bot para WhatsApp. Recuperado 20 de septiembre de 2020, de https://blog.hubspot.es/marketing/whatsapp-bot

Melton, Jim; Simon, A. R. . (1993). Understanding the New SQL: A Complete Guide. Recuperado 14 de febrero de 2021, de https://books.google.es/books?hl=es&lr=&id=ZOOMSTZ4T_QC&oi=fnd&pg=PA1&dq=sql&ots=e2I89NRdgw&sig=r6wcMzN4Ws_uzFwgX4LkAVgkYwA#v=onepage&q=sql&f=false

Miranda, L. (2020). Este chat de WhatsApp te mantiene informado sobre el coronavirus. Recuperado 3 de enero de 2021, de https://hipertextual.com/2020/03/whatsapp-coronavirus-chat-oms

Neurosciences, W. I. for. (2009). The Mediterranean Diet Pyramid. Recuperado 4 de mayo de 2021, de https://memory.ucsf.edu/sites/memory.ucsf.edu/files/MediterraneanDietHandout.pdf

Node.js. (2020). Recuperado 1 de noviembre de 2020, de https://es.wikipedia.org/wiki/Node.js

nodemon. (s. f.). Recuperado 8 de noviembre de 2020, de https://nodemon.io/

Octane AI. (2020). Octane AI | Quiz, Messenger, SMS for Shopify. Recuperado 1 de noviembre de 2020, de https://octaneai.com/

OMS. (2015). Obesidad y sobrepeso (OMS). Recuperado 9 de agosto de 2020, de http://www.who.int/mediacentre/factsheets/fs311/es/

Planeta Chatbot. (2017). Cómo crear Bots en WhatsApp. Recuperado 1 de noviembre de 2020, de https://planetachatbot.com/crear-bots-en-whatsapp-35f36d4f9928

Precios de Azure Bot Service | Microsoft Azure. (s. f.). Recuperado 10 de noviembre de 2020, de https://azure.microsoft.com/es-es/pricing/details/bot-service/

Precios de Chatbot | SnatchBot. (s. f.). Recuperado 10 de noviembre de 2020, de https://es.snatchbot.me/pricing

Principales resultados Estadística de Gasto Sanitario Público. (2018).

PyCharm: uno de los mejores IDE para Python — Escuela de Python. (s. f.). Recuperado 8 de noviembre de 2020, de https://www.escuelapython.com/pycharm-uno-de-los-mejores-ide-para-python/

Python - Wikipedia, la enciclopedia libre. (s. f.). Recuperado 8 de noviembre de 2020, de https://es.wikipedia.org/wiki/Python

Serie 27k. (s. f.). Recuperado 15 de marzo de 2021, de https://www.iso27000.es/iso27000.html

System Usability Scale (SUS) | Usability.gov. (s. f.). Recuperado 17 de febrero de 2021, de https://www.usability.gov/how-to-and-tools/methods/system-usability-scale.html

Tutorial: cómo construir un chatbot en Facebook Messenger | by Dimitry Kagan | Planeta Chatbot : todo sobre los Chat bots, Voice apps e Inteligencia Artificial. (s. f.). Recuperado

8 de noviembre de 2020, de https://planetachatbot.com/tutorial-como-construir-un-chatbot-con-facebook-messenger-de474ee93f92

User Experience Questionnaire (UEQ). (s. f.). Recuperado 29 de marzo de 2021, de https://www.ueq-online.org/

Wakamola, un bot que te ayuda a conocer mejor tus hábitos nutricionales y de actividad física a través de Telegram – Wakamola. (2019). Recuperado 24 de julio de 2020, de https://wakamola.webs.upv.es/index.php/2019/01/28/__trashed/

WhatsApp Inc. (2020). API de WhatsApp Business. Recuperado 1 de noviembre de 2020, de https://www.whatsapp.com/business/api?lang=es

Withrow, D., & Alter, D. A. (2011, febrero 1). The economic burden of obesity worldwide: A systematic review of the direct costs of obesity. *Obesity Reviews*. John Wiley & Sons, Ltd. https://doi.org/10.1111/j.1467-789X.2009.00712.x