



**ESCUELA SUPERIOR POLITÉCNICA DE CHIMBORAZO**  
**FACULTAD DE INFORMÁTICA Y ELECTRÓNICA**  
**ESCUELA DE SOFTWARE**



**APLICACIONES INFORMÁTICAS 2**

**Integrantes:**

Santiago Guaylla	6822
Sebastián Burgos	6816
Héctor Nieto	6832
Santiago Melena	6828

**Docente:**

Ing. Julio Santillán

**Fecha:**

**24/07/2023**

**Tema:**

Desarrollo de un Chatbot basado en Inteligencia Artificial para la atención personalizada al cliente de una franquicia farmacéutica

# Índice

Propósito.....	4
Componentes.....	4
Alcance del producto Software. ....	4
Beneficios que brindará al área de negocio y organización. ....	4
Objetivos y metas. ....	5
REQUISITOS FUNCIONALES .....	5
REQUISITOS NO FUNCIONALES .....	6
RESTRICCIONES.....	8
DIAGRAMAS DE CASOS DE USO.....	9
Diagramas UML.....	12
DIAGRAMA BASE DE DATOS FARMACIA: PROCESO FACTURACIÓN .....	13
DIAGRAMA DE BASE DE DATOS CHATBOT .....	14
Planificación de Desarrollo.....	15
Herramienta Usada. ....	15
Metodología SCRUM en Monday. ....	15
Cronograma. ....	16
Flujo de actividades.....	18
Registro de actividades. ....	18
Desarrollo de la Aplicación.....	19
Creación del flujo de la conversación del chatbot .....	19
Diseño de la interfaz de usuario usando de la herramienta Balsamiq.....	23
Base de Datos. ....	26
Creación de la capa de datos. ....	27
Interfaz de Usuario.....	28
Conexión Frontend y Backend. ....	29
Validación de datos. ....	30
Creación de los Intents y Entities básicos para el chatbot. ....	30
Interfaz de Usuario del ChatBot e Implementación de la API. ....	32
Arquitectura de la Información.....	34
Implementación con la validación de datos. ....	38
Inicio de sesión (Cookie).....	38
Creación de los Intents y Entities avanzados para el chatbot. ....	40
Implementación de la base de datos de la farmacia para el dialogo de flujo. ....	41
Interfaz de Usuario de la aplicación.....	42
Vonage Nexmo.....	46

Messenger como una alternativa de información.....	47
Pruebas del Sistema.....	49
Login.....	49
Registro .....	51
Manual de Usuario – PharmacyBot .....	59
Descripción de la Aplicación. ....	59
Contenido del Manual.....	59
Inicio de Sesión (Login).....	60
Registro de cuenta. ....	61
ChatBot.....	62
Navegación del Botón del Usuario:.....	63
PRUEBAS DE USABILIDAD “PHARMACYBOT”. ....	64
Evaluación de la facilidad de uso .....	64
ANEXO .....	69
Cuestionario de usabilidad y satisfacción.....	69
Bibliografía .....	71

## **Propósito.**

Tema: Desarrollo de un chatbot basado en inteligencia artificial para atención personalizada al cliente de una franquicia farmacéutica

## **Componentes.**

Diseño de la interfaz del chatbot: Esto incluye el diseño de la interfaz de usuario del chatbot, la forma en que se presenta al cliente y las opciones de navegación y de entrada de texto.

Implementación de la inteligencia artificial: El chatbot debe tener capacidades de procesamiento de lenguaje natural (NLP) y aprendizaje automático (ML) para entender las consultas del cliente y proporcionar respuestas precisas y útiles.

Integración con la base de conocimientos de la empresa: El chatbot debe tener acceso a la base de conocimientos de la empresa, que incluye información sobre productos, precios, stock y otros detalles relevantes para los clientes.

Implementación de funcionalidades específicas: El chatbot cumplirá con las funcionalidades específicas según las necesidades de la franquicia farmacéutica, como la programación de citas, solventar dudas, asesoramiento de productos y servicios, entre otros.

Pruebas y evaluación: El chatbot debe cumplir una evaluación continua para mejorar sus capacidades y rendimiento.

El chatbot desarrollado utilizará inteligencia artificial y procesamiento del lenguaje natural para interactuar con los clientes y brindarles atención personalizada. Se cubrirá la totalidad del software.

## **Alcance del producto Software.**

El alcance del proyecto FarmChatProject consiste en desarrollar un chatbot basado en inteligencia artificial que brinde atención personalizada al cliente de una franquicia farmacéutica. Las funciones principales del chatbot serán las siguientes:

Solventar dudas: El chatbot responderá a las preguntas frecuentes de los clientes sobre la franquicia, los productos, las políticas de envío, la disponibilidad de productos, entre otros.

Consultar con el inventario de la farmacia y ofrecer los productos al cliente: El chatbot consultará la base de datos de la farmacia para conocer la disponibilidad de productos y ofrecerlos al cliente. Además, brindará información sobre los productos, su uso y los posibles efectos secundarios.

Enviar los productos de la farmacia a domicilio consultando la disponibilidad: El chatbot consultará la base de datos de la franquicia para conocer la disponibilidad de los productos y ofrecerá al cliente la opción de enviar los productos a domicilio. También proporcionará información sobre los plazos de entrega y los costos de envío.

El proyecto FarmChatProject no incluirá funciones adicionales a las mencionadas anteriormente. Además, el chatbot estará disponible en línea las 24 horas del día, los 7 días de la semana.

Objetivo general: Proporcionar una solución automatizada de atención al cliente para una franquicia farmacéutica.

## **Beneficios que brindará al área de negocio y organización.**

- Atención personalizada al cliente
- Solventar dudas frecuentes

- Consultar con el inventario de la farmacia y ofrecer los productos al cliente
- Conectar al cliente con un médico y agendar una cita, la cual se enviará a su WhatsApp con los datos de la reserva
- Servicio de envío de productos de la farmacia hacia el lugar registrado por el cliente

#### Objetivos y metas.

- Mejorar la atención al cliente
- Mejorar la eficiencia
- Generar nuevas fuentes de ingresos
- Promover la innovación tecnológica

#### REQUISITOS FUNCIONALES

IDENTIFICADOR	RF1
NOMBRE DEL REQUISITO	<b>Registro de usuario</b>
DESCRIPCIÓN	El Chatbot debe permitir que el usuario se registre proporcionando sus datos personales y de contacto.
CARACTERÍSTICAS	El registro debe ser sencillo y rápido, y los datos deben almacenarse de forma segura.
ACTORES	Usuario

IDENTIFICADOR	RF2
NOMBRE DEL REQUISITO	<b>Búsqueda de productos</b>
DESCRIPCIÓN	El chatbot debe permitir la búsqueda de productos en el inventario de la franquicia farmacéutica.
CARACTERÍSTICAS	La búsqueda debe ser intuitiva y permitir la filtración de resultados.
ACTORES	Usuario

IDENTIFICADOR	RF3
NOMBRE DEL REQUISITO	<b>Consulta de información médica</b>
DESCRIPCIÓN	El chatbot debe ser capaz de responder preguntas sobre información médica y de salud.
CARACTERÍSTICAS	El chatbot debe tener acceso a una base de datos confiable y actualizada.
ACTORES	Usuario

IDENTIFICADOR	RF4
NOMBRE DEL REQUISITO	<b>Recomendaciones personalizadas</b>
DESCRIPCIÓN	El chatbot debe ser capaz de hacer recomendaciones de productos y servicios personalizados en función de la información proporcionada por el usuario.
CARACTERÍSTICAS	Las recomendaciones deben estar respaldadas por datos médicos confiables.
ACTORES	Usuario

IDENTIFICADOR	RF5
NOMBRE DEL REQUISITO	<b>Gestión de la base de datos</b>
DESCRIPCIÓN	El chatbot debe ser capaz de gestionar y actualizar la base de datos de usuarios y productos de la franquicia farmacéutica.
CARACTERÍSTICAS	La base de datos debe ser segura y actualizada regularmente.
ACTORES	Administrador

IDENTIFICADOR	RF6
NOMBRE DEL REQUISITO	<b>Análisis de datos</b>
DESCRIPCIÓN	El chatbot debe ser capaz de analizar los datos de los usuarios para obtener información sobre sus patrones de compra y preferencias.
CARACTERÍSTICAS	El análisis debe ser preciso y confiable, y los datos deben ser utilizados para mejorar el servicio.
ACTORES	Administrador

IDENTIFICADOR	RF7
NOMBRE DEL REQUISITO	<b>Atención al cliente</b>
DESCRIPCIÓN	El chatbot debe ser capaz de brindar atención al cliente en tiempo real, proporcionando asistencia inmediata y personalizada.
CARACTERÍSTICAS	La atención en vivo debe estar disponible en horarios establecidos y contar con personal capacitado para responder a las consultas.
ACTORES	Cliente

IDENTIFICADOR	RF8
NOMBRE DEL REQUISITO	<b>Historial de chats</b>
DESCRIPCIÓN	El chatbot debe permitir al usuario acceder a su historial de chats y realizar consultas sobre sus pedidos anteriores.
CARACTERÍSTICAS	El historial de compras debe estar actualizado y ser fácilmente accesible para el usuario.
ACTORES	Cliente

## REQUISITOS NO FUNCIONALES

IDENTIFICADOR	RNF1
NOMBRE DEL REQUISITO	<b>Seguridad</b>
DESCRIPCIÓN	El chatbot debe cumplir con altos estándares de seguridad y privacidad para proteger la información personal y médica de los usuarios.
CARACTERÍSTICAS	Cifrado de extremo a extremo, autenticación segura de usuario, protección contra ataques cibernéticos.
ACTORES	Desarrolladores, usuarios, administradores del sistema.

IDENTIFICADOR	RNF2
NOMBRE DEL REQUISITO	<b>Fiabilidad y Disponibilidad</b>
DESCRIPCIÓN	El chatbot debe ser confiable y estar disponible en todo momento para atender las necesidades de los usuarios.
CARACTERÍSTICAS	Monitorización constante, detección y corrección de errores en tiempo real, redundancia del sistema.
ACTORES	Desarrolladores, usuarios, administradores del sistema.

IDENTIFICADOR	RNF3
NOMBRE DEL REQUISITO	<b>Escalabilidad</b>
DESCRIPCIÓN	El chatbot debe ser capaz de manejar grandes volúmenes de tráfico y crecer en capacidad de acuerdo con las necesidades de la franquicia farmacéutica.
CARACTERÍSTICAS	Arquitectura escalable, gestión de carga, recursos adicionales bajo demanda.
ACTORES	Desarrolladores, administradores del sistema.

IDENTIFICADOR	RNF4
NOMBRE DEL REQUISITO	<b>Adaptabilidad</b>
DESCRIPCIÓN	El chatbot debe ser capaz de adaptarse a diferentes tipos de usuarios y situaciones, y ofrecer soluciones personalizadas para cada uno de ellos.
CARACTERÍSTICAS	Algoritmos de aprendizaje automático, reconocimiento de voz y texto, integración con sistemas de gestión de datos.
ACTORES	Desarrolladores, usuarios, administradores del sistema.

IDENTIFICADOR	RNF5
NOMBRE DEL REQUISITO	<b>Interoperabilidad</b>
DESCRIPCIÓN	El chatbot debe ser capaz de interoperar con otros sistemas y aplicaciones utilizados por la franquicia farmacéutica, como sistemas de gestión de inventario y de pedidos.
CARACTERÍSTICAS	Integración con diferentes sistemas y aplicaciones, compatibilidad con diferentes formatos de datos.
ACTORES	Desarrolladores, administradores del sistema.

IDENTIFICADOR	RNF6
NOMBRE DEL REQUISITO	<b>Usabilidad</b>
DESCRIPCIÓN	El chatbot debe ser fácil de usar y comprender para los usuarios, independientemente de su nivel de experiencia tecnológica.

CARACTERÍSTICAS	Integración con sistemas de gestión de datos de usuario, análisis de datos para ofrecer soluciones personalizadas.
ACTORES	Desarrolladores, usuarios.

IDENTIFICADOR	RNF7
NOMBRE DEL REQUISITO	<b>Respuestas personalizadas</b>
DESCRIPCIÓN	El chatbot debe ser capaz de ofrecer soluciones personalizadas y adaptadas a las necesidades individuales de cada usuario.
CARACTERÍSTICAS	Integración con sistemas de gestión de datos de usuario, análisis de datos para ofrecer soluciones personalizadas.
ACTORES	Desarrolladores, usuarios.

IDENTIFICADOR	RNF8
NOMBRE DEL REQUISITO	<b>Compatibilidad</b>
DESCRIPCIÓN	El chatbot debe ser compatible con diferentes idiomas.
CARACTERÍSTICAS	Integración con servicios de traducción automática, soporte multilingüe, adaptabilidad cultural.
ACTORES	Desarrolladores, usuarios, administradores del sistema.

IDENTIFICADOR	RNF9
NOMBRE DEL REQUISITO	<b>Validación y comprobación</b>
DESCRIPCIÓN	El chatbot debe validar los datos ingresados con respecto al usuario registrado, datos de envío, entre otros
CARACTERÍSTICAS	Debe comprobar que los datos ingresados sean correctos
ACTORES	Usuarios, administradores

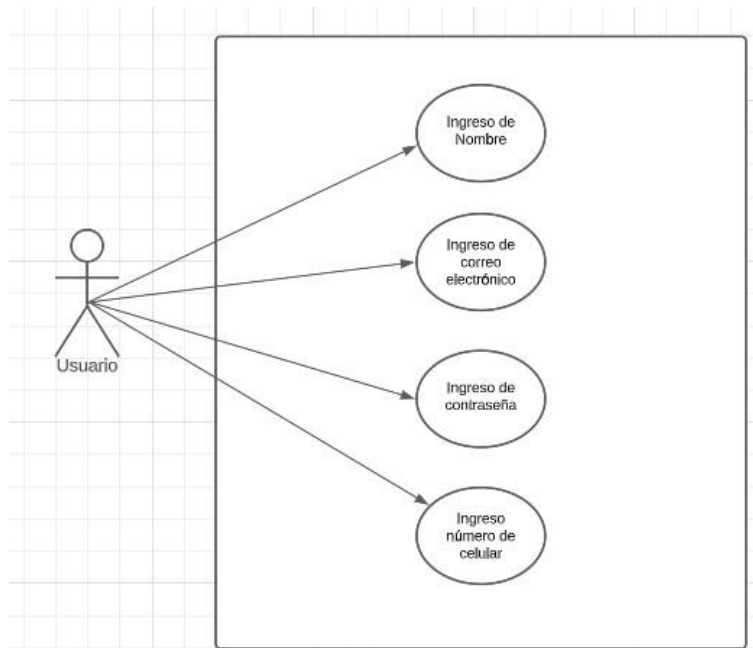
## RESTRICCIONES

Límites de lenguaje	El chatbot debe estar diseñado para manejar una cantidad limitada de idiomas y dialectos para garantizar una comprensión adecuada del usuario y evitar confusiones.
Limitaciones de conocimiento	El chatbot solo puede proporcionar información dentro de su área de conocimiento y experiencia. Es importante asegurarse de que el chatbot no responda a preguntas fuera de su capacidad y genere respuestas inexactas o irrelevantes.
Limitaciones de capacidad de respuesta	El chatbot debe ser programado para responder dentro de un límite de tiempo razonable, lo que significa que la capacidad de respuesta debe estar limitada a una ventana de tiempo específica.

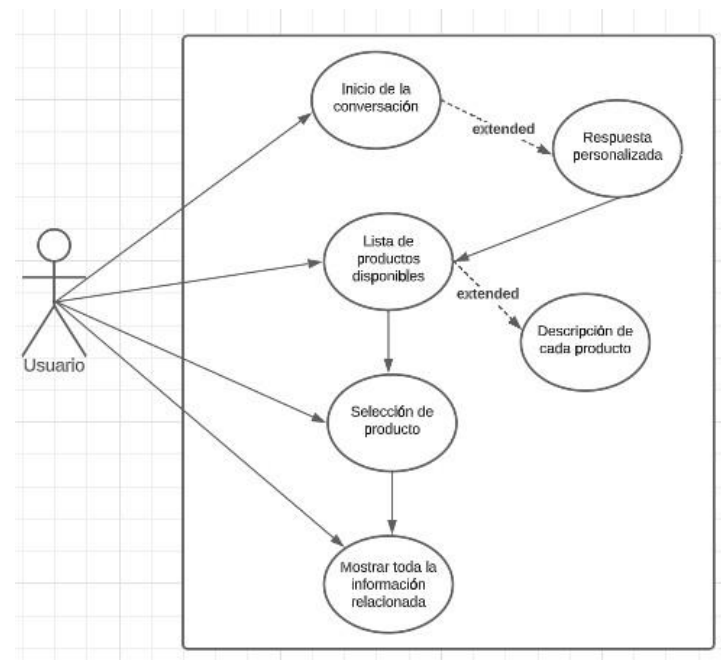


Restricciones de seguridad y privacidad	El chatbot debe estar diseñado para cumplir con las políticas y requisitos de privacidad de los usuarios. Por ejemplo, no debe recopilar información de usuario sensible como contraseñas, números de seguridad social, etc.
Restricciones de integración	El chatbot debe ser compatible con la plataforma en la que se va a implementar. Se deben considerar restricciones como los requisitos técnicos de la plataforma, la integración de API y las limitaciones de la infraestructura

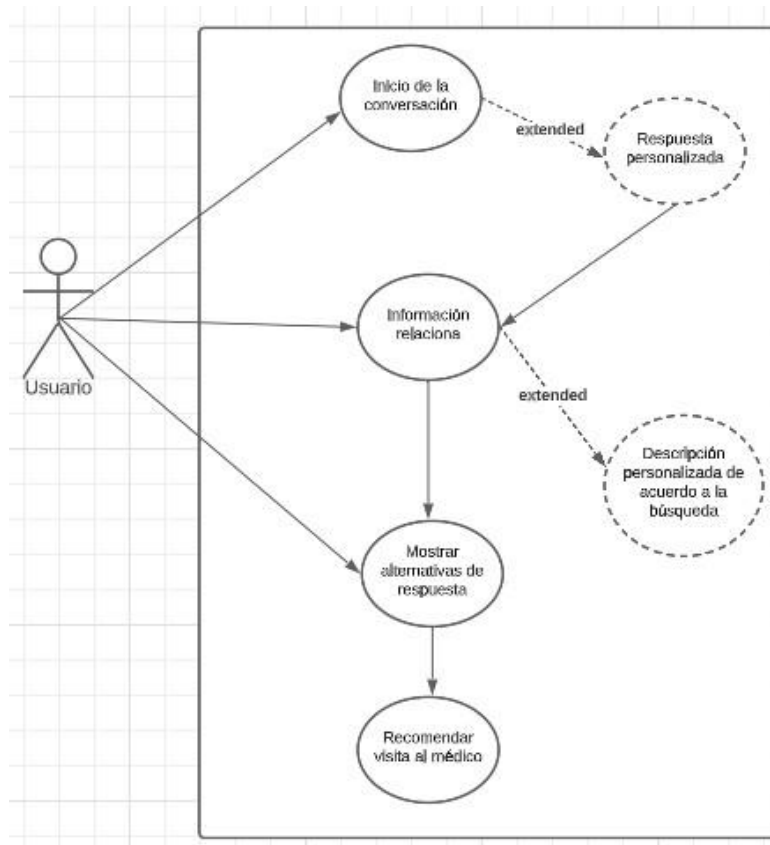
## DIAGRAMAS DE CASOS DE USO



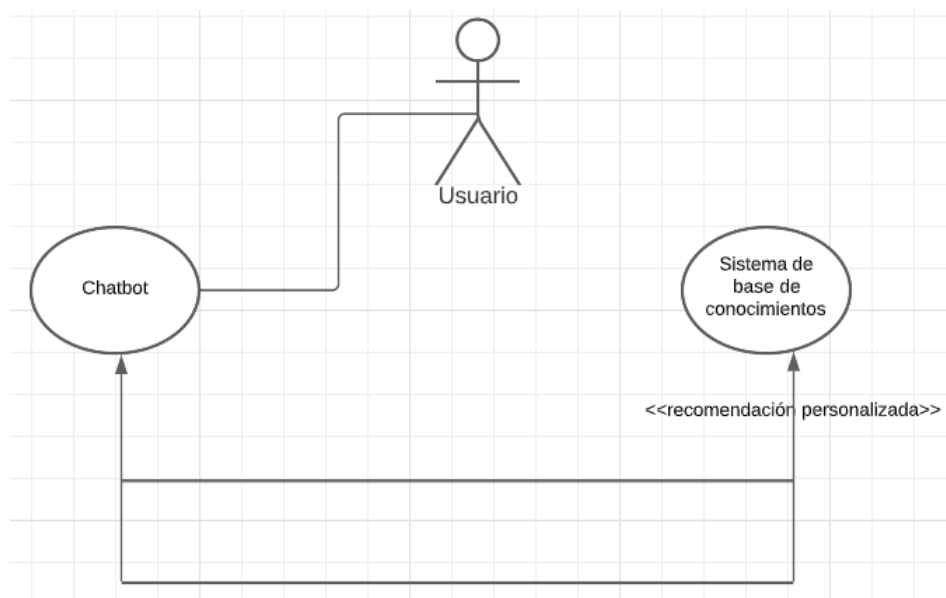
Caso de uso1.- Registro de usuario



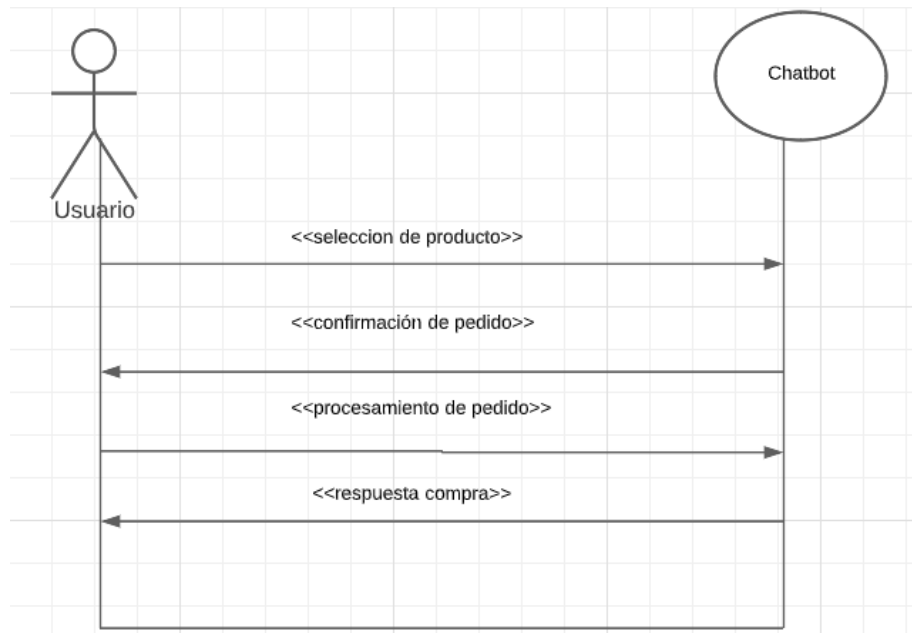
Caso de uso2.- Búsqueda de productos



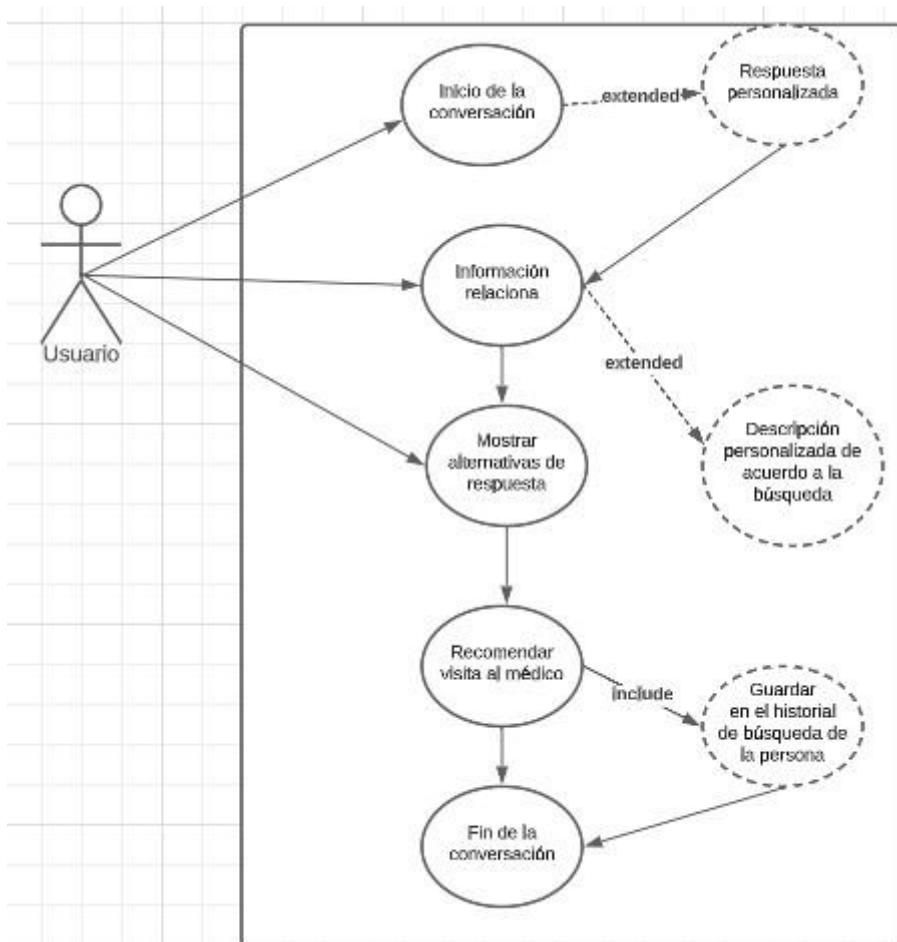
Caso de uso3.- Consulta de información médica



Caso de uso4.- Recomendaciones personalizadas



Caso de uso5.- Compra de productos



Caso de uso6.- Historial de chats y de compra

Diagramas UML.

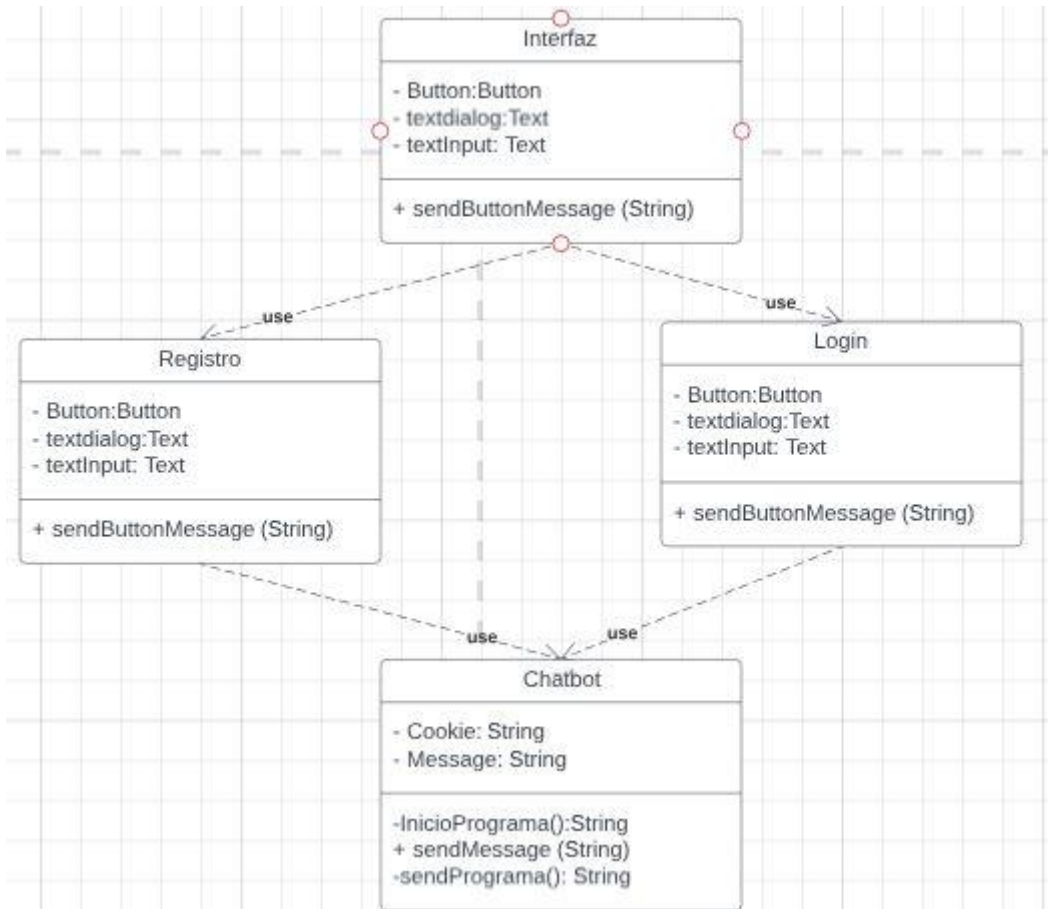


Diagrama de clases

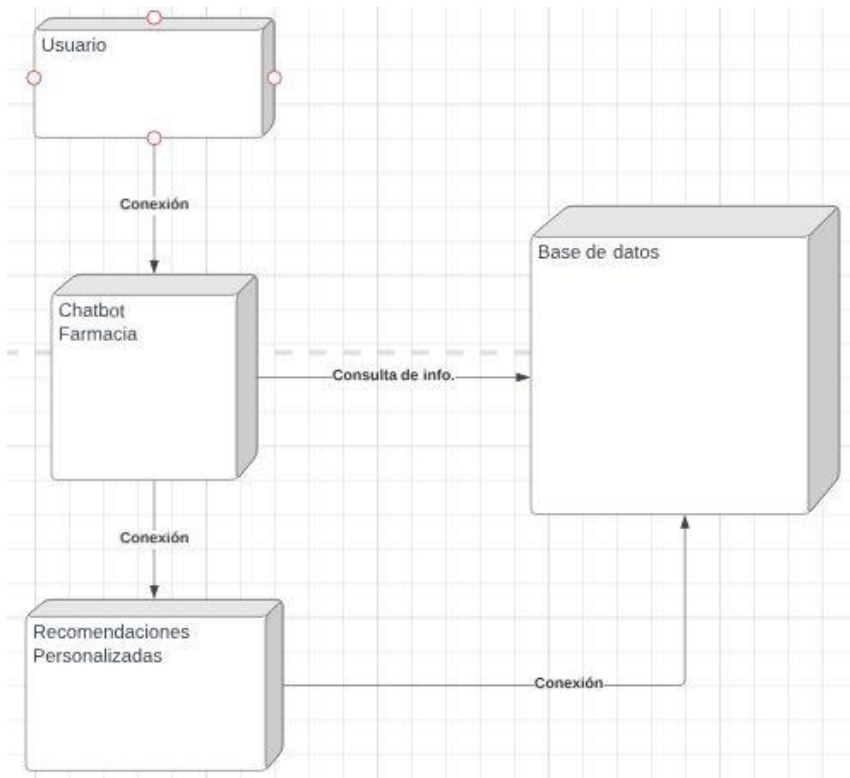
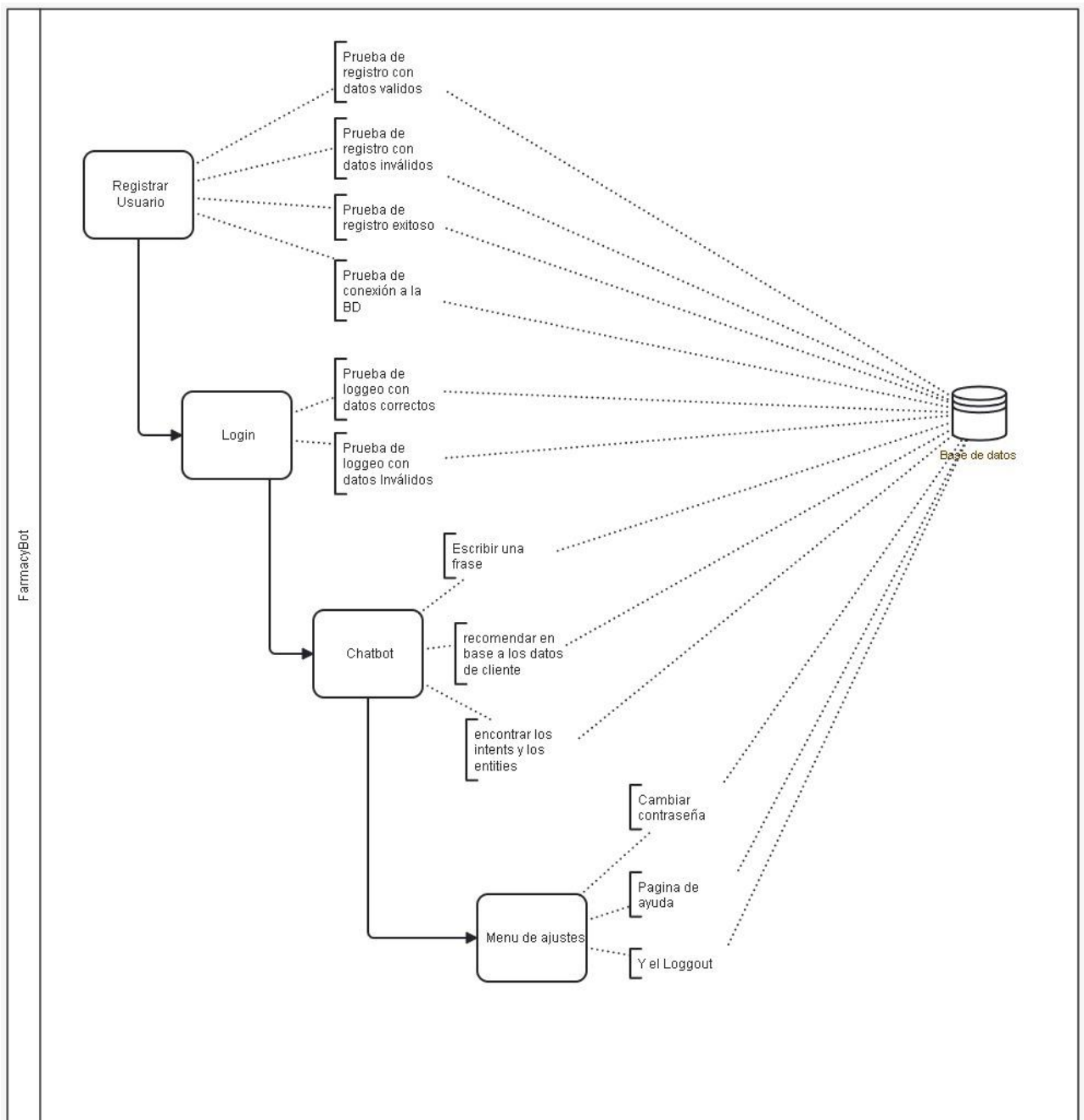
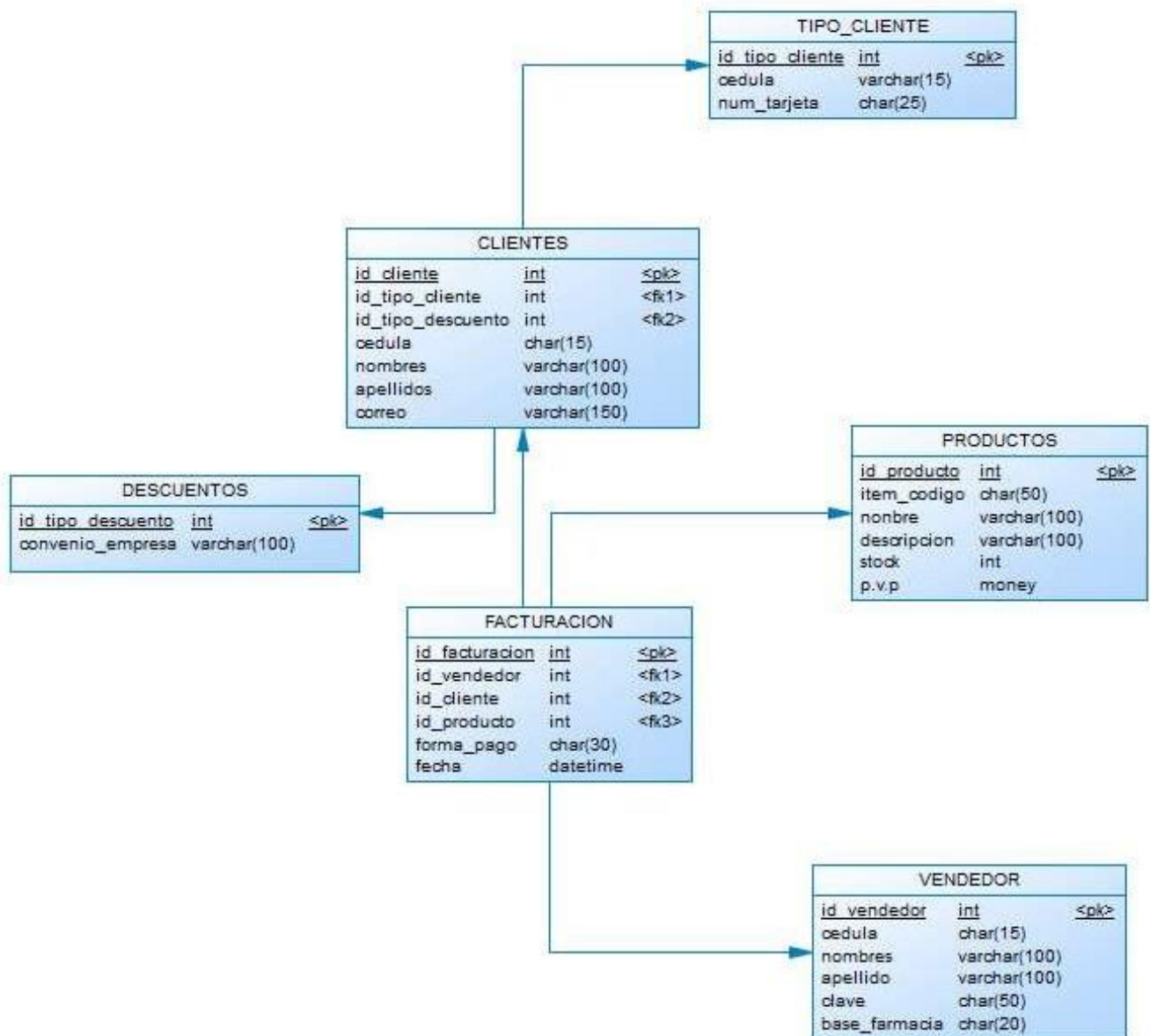


Diagrama de componentes

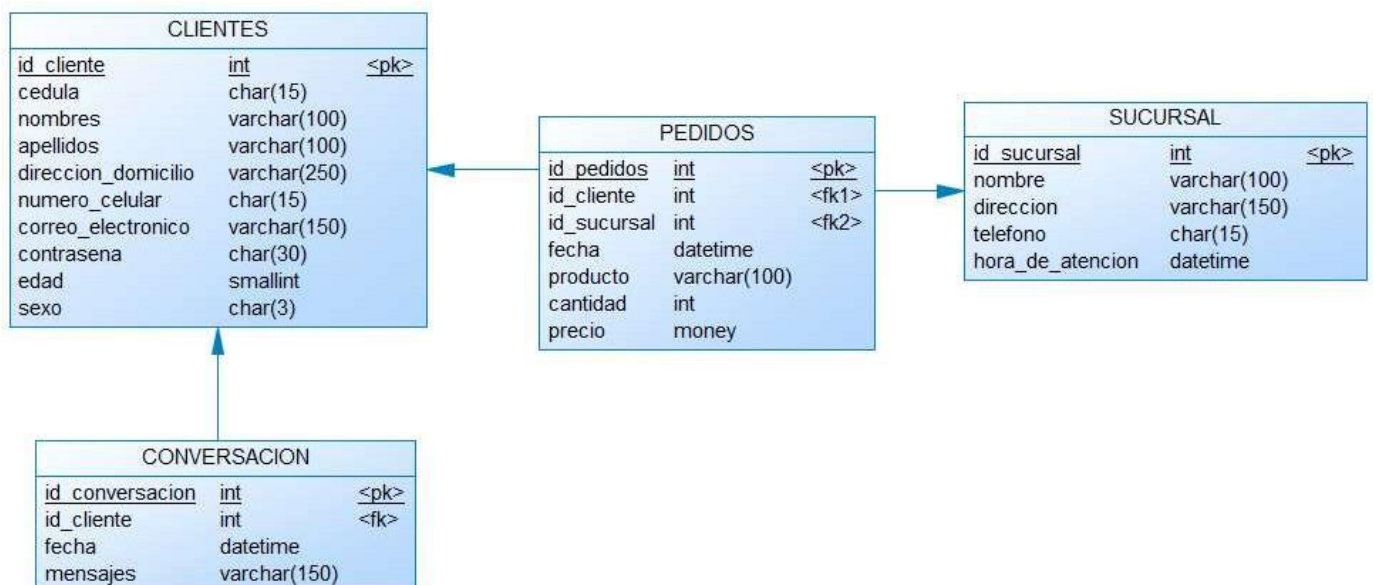


## DIAGRAMA BASE DE DATOS FARMACIA: PROCESO FACTURACIÓN

El sistema de la Farmacia está compuesto por muchos servicios, en el que nos vamos a centrar ahora mismo solo involucra al proceso de facturación.



## DIAGRAMA DE BASE DE DATOS CHATBOT



## **Planificación de Desarrollo.**

La planificación del desarrollo de un proyecto es una etapa esencial para garantizar su éxito y eficiencia. Consiste en definir y organizar las actividades necesarias para alcanzar los objetivos del proyecto de manera sistemática y ordenada. Esta etapa involucra la identificación de los recursos necesarios, la asignación de tareas y responsabilidades, y la elaboración de un cronograma detallado.

La planificación del desarrollo del proyecto comienza con la definición clara de los objetivos y metas que se desean lograr. Es importante establecer metas realistas y medibles para poder evaluar el progreso a lo largo del tiempo. Además, es fundamental identificar los requisitos y las restricciones del proyecto, incluyendo el presupuesto, los plazos y los recursos disponibles. Una vez que se han establecido los objetivos y los requisitos, se procede a descomponer el proyecto en tareas más pequeñas y manejables. Estas tareas se organizan en una estructura jerárquica conocida como desglose de trabajo (WBS, por sus siglas en inglés), que permite visualizar la relación entre las diferentes actividades y su interdependencia.

## **Herramienta Usada.**

Monday es una plataforma de gestión de proyectos y colaboración en línea que se utiliza ampliamente para planificar, organizar y supervisar el desarrollo de proyectos. Con su enfoque intuitivo y versátil, Monday se ha convertido en una herramienta popular para equipos de diferentes industrias y tamaños.

La planificación de proyectos con Monday ofrece una manera eficiente y estructurada de definir objetivos, asignar tareas, gestionar recursos y realizar un seguimiento del progreso. Esta plataforma permite a los equipos crear tableros personalizados que reflejan el flujo de trabajo específico de su proyecto, lo que facilita la organización y la visualización de las tareas.

Con Monday, es posible descomponer un proyecto en tareas más pequeñas y asignarlas a los miembros del equipo con fechas límite claras. La asignación de responsabilidades y recursos se realiza de manera sencilla, lo que permite una distribución equitativa de la carga de trabajo y una colaboración efectiva. Además, Monday proporciona herramientas de comunicación y colaboración integradas, lo que facilita la interacción entre los miembros del equipo. Los comentarios, las actualizaciones y los archivos pueden compartirse directamente en la plataforma, lo que mejora la comunicación y evita la dispersión de información en diferentes canales.

## **Metodología SCRUM en Monday.**

Scrum es una metodología ágil ampliamente utilizada para la gestión de proyectos. Aunque Monday no es exclusivamente una herramienta de gestión de proyectos ágil, se puede adaptar y utilizar para implementar los principios de Scrum, esta metodología en Monday:

**Creación de tableros:** En Monday, puedes crear un tablero principal que represente tu backlog de productos o elementos pendientes. Los elementos del backlog pueden ser representados como tarjetas o elementos individuales en el tablero.

**Sprints:** Un sprint es un período de tiempo fijo en el que se realiza el trabajo planificado. En Monday, puedes utilizar tableros separados para cada sprint, donde los elementos seleccionados del backlog se mueven y se asignan a los miembros del equipo.

**Reuniones diarias:** Las reuniones diarias o "daily scrums" son breves encuentros en los que los miembros del equipo comparten actualizaciones sobre su trabajo. En Monday, puedes aprovechar las funciones de comentarios y actualizaciones de estado en cada tarjeta para realizar un seguimiento del progreso diario y fomentar la comunicación entre los miembros del

equipo.

**Burndown chart:** El gráfico de burndown muestra el trabajo restante a medida que avanza el sprint. Aunque Monday no proporciona una función específica para generar gráficos de burndown, puedes utilizar sus funciones de seguimiento de tiempo y avance para recopilar datos y crear un gráfico de burndown fuera de la plataforma.

**Retrospectivas:** Al final de cada sprint, se lleva a cabo una reunión de retrospectiva para analizar lo que salió bien, lo que se puede mejorar y definir acciones para el próximo sprint. En Monday, puedes utilizar columnas o etiquetas personalizadas para marcar las tareas completadas y recopilar comentarios y reflexiones sobre el sprint para su revisión durante la reunión de retrospectiva.

**Planificación de nuestro Proyecto realizada en Monday. Integrantes.**

Es importante tener en cuenta que la composición exacta del equipo puede variar según las necesidades del proyecto y la estructura organizativa. Además, Monday.com permite personalizar aún más los roles y permisos para adaptarlos a los requisitos específicos del equipo y la empresa.

Cada integrante del equipo en Monday.com tiene asignadas tareas y responsabilidades, y puede colaborar mediante la comunicación y el intercambio de actualizaciones dentro de la plataforma. Esto facilita la colaboración efectiva y la gestión del proyecto en un entorno centralizado.



## Cronograma.

El uso de un cronograma ofrece varios beneficios para la gestión eficiente de proyectos:

**Visualización clara:** La visualización en forma de cronograma permite tener una perspectiva clara y fácilmente comprensible de las fechas de inicio, finalización y duración de las tareas. Esto facilita la planificación y el seguimiento del progreso del proyecto.

**Gestión de dependencias:** La capacidad de establecer dependencias entre las tareas en Monday.com ayuda a identificar las interdependencias y las relaciones de precedencia entre las actividades del proyecto. Esto permite una gestión más efectiva y ayuda a evitar retrasos o conflictos en el cronograma.

**Asignación de recursos:** La visualización del cronograma en Monday.com permite identificar fácilmente las tareas asignadas a cada miembro del equipo, lo que ayuda a equilibrar la carga de trabajo y garantizar una asignación eficiente de los recursos disponibles.

**Seguimiento y actualización:** El cronograma en Monday.com se puede actualizar y ajustar de manera sencilla a medida que avanza el proyecto. Las funciones de seguimiento y actualización de estado permiten mantener un registro actualizado del progreso de las tareas, lo que ayuda a identificar cualquier desviación.



## CodeTeam cronograma

Cronograma de actividades para el desarrollo del proyecto. Ver más

Actividad

Invitar / 4

Tabla principal Gráfico

Integrar

Automatizar

Agregar Tarea

Buscar

Persona

Filtrar

Ordenar

Contar

### PLANIFICACIÓN Y ANÁLISIS

Tarea	Estado	Cronograma	Priority	Detalles	Horas
Análisis de requisitos	Listo	abr. 28 - 30	Critical	Identificar y documentar los requisitos f...	20
Investigación de mercado	Listo	abr. 28 - may. 3	Low	Realizar un análisis del mercado y la co...	30
Definición de funcionalidades	Listo	may. 1 - 6	Medium	Establecer las funcionalidades y caracte...	30
+ Agregar Tarea					
					40 Total

### DISEÑO Y ARQUITECTURA

Tarea	Estado	Cronograma	Priority	Detalles	Horas
Diseño de la arquitectura	En curso	may. 7 - 15	High	Definir la estructura y componentes del ...	30
Creación del flujo de conversación	Pendiente	may. 10 - 20	Medium	Definir y diseñar el flujo de interacción d...	30
Diseño de la interfaz de usuario	Pendiente	may. 10 - 20	Medium	Crear y diseñar la interfaz gráfica de t...	10
+ Agregar Tarea					
					70 Total

### DESARROLLO DEL BACKEND Y LA INTELIGENCIA ARTIFICIAL

Tarea	Estado	Cronograma	Priority	Detalles	Horas
Desarrollo del backend	Pendiente	may. 21 - jun. 3	Critical	Crear la infraestructura y lógica del lado...	40
Desarrollo de la inteligencia artificial	Pendiente	may. 22 - jun. 4	Critical	Implementar y entrenar los modelos de...	60
Integración con APIs externas	Pendiente	may. 28 - jun. 2	Medium	Conectar el chatbot con APIs externas...	10
+ Agregar Tarea					
					110 Total

### IMPLEMENTACIÓN DE LA INTERFAZ DE USUARIO

Tarea	Estado	Cronograma	Priority	Detalles	Horas
Implementación de la interfaz de usu...	Pendiente	jun. 4 - 7	High	Crear y diseñar la interfaz gráfica de t...	40
Pruebas de usabilidad	Pendiente	jun. 7 - 11	Medium	Evaluar la usabilidad y experiencia del us...	20
Pruebas de funcionamiento	Pendiente	jun. 12 - 15	Critical	Realizar pruebas exhaustivas para verifi...	30
Corrección de errores y mejoras	Pendiente	jun. 16 - 17	Medium	Identificar y solucionar cualquier proble...	40
+ Agregar Tarea					
					130 Total

### INTEGRACIÓN Y PRUEBAS DE INTEGRACIÓN

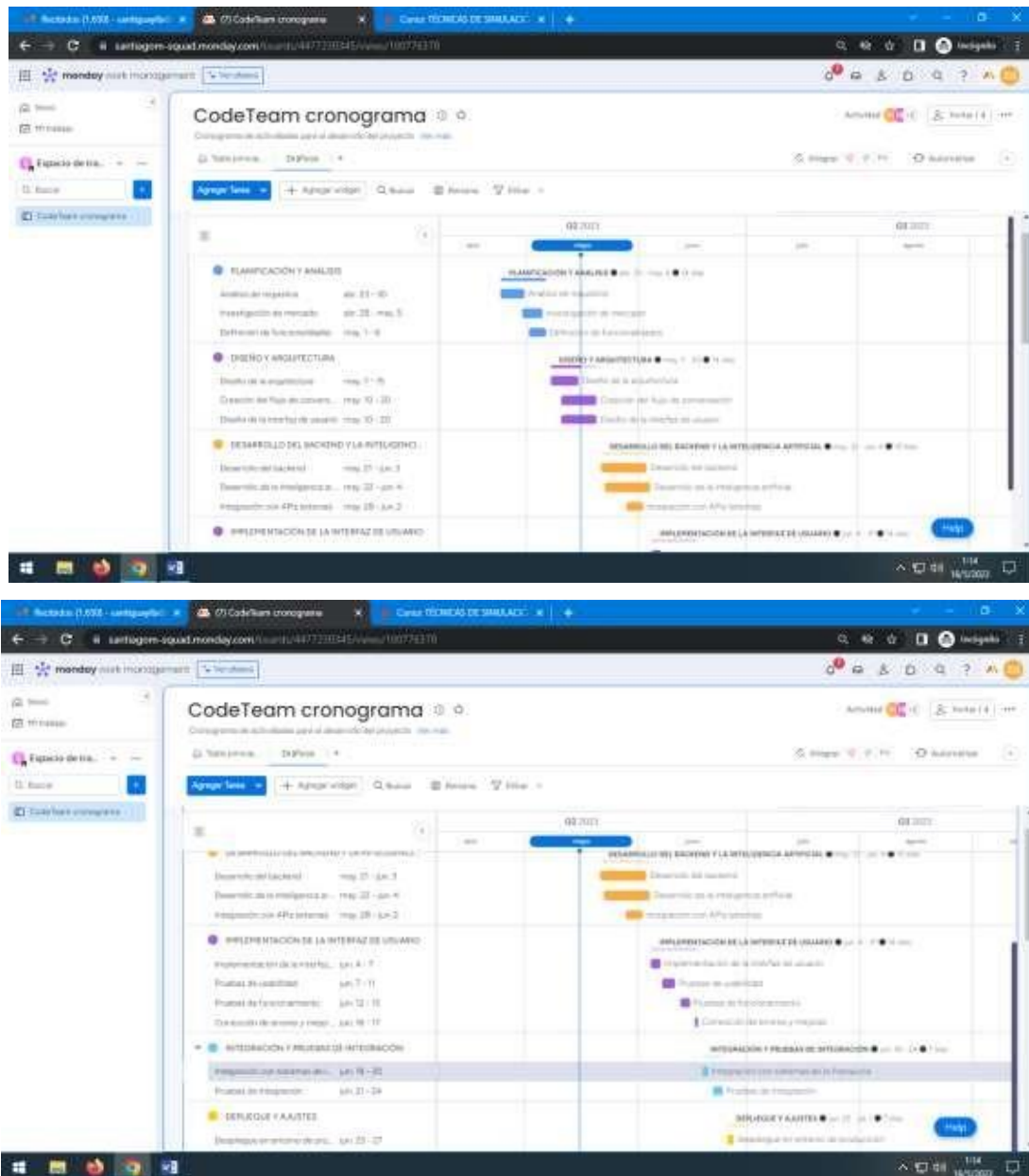
Tarea	Estado	Cronograma	Priority	Detalles	Horas
Integración con sistemas de la franqu...	Pendiente	jun. 18 - 20	High	Conectar el chatbot con los sistemas ex...	40
Pruebas de integración	Pendiente	jun. 21 - 24	Medium	Verificar la correcta integración del chat...	30
+ Agregar Tarea					
					70 Total

### DESPLIEGUE Y AJUSTES

Tarea	Estado	Cronograma	Priority	Detalles	Horas
Despliegue en entorno de producción	Pendiente	jun. 25 - 27	Critical	Preparar y lanzar el chatbot en un entor...	10
Capacitación del chatbot	Pendiente	jun. 28 - 30	High	Entrenar al chatbot a través de la inter...	20
Ajuste y optimización de la inteligenci...	Pendiente	jun. 28 - jul. 1	High	Mejorar y refinar los modelos de intelec...	20
+ Agregar Tarea					
					50 Total

## Flujo de actividades.

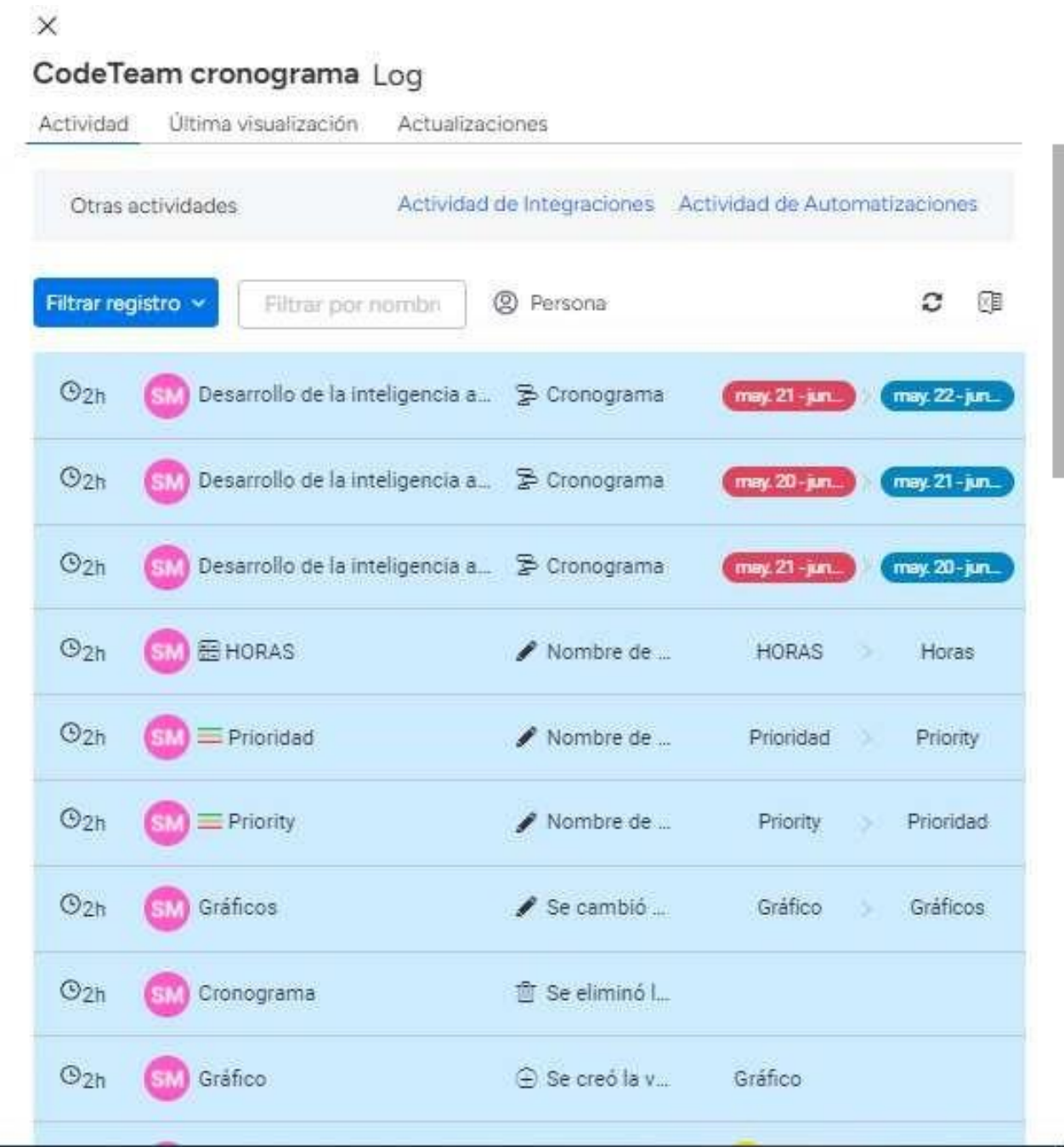
El flujo de actividades se puede gestionar mediante la creación de tableros y columnas personalizadas que representen las diferentes etapas o estados de un proceso. El flujo de actividades en Monday.com ofrece una forma visual y estructurada de gestionar y monitorear el progreso de las tareas. Con el uso de columnas personalizadas, tarjetas y automatizaciones, puedes mantener un flujo continuo y eficiente de actividades en tu proyecto, lo que facilita la colaboración y el seguimiento del progreso.



## Registro de actividades.

En Monday.com, el registro de actividades se puede llevar a cabo de varias formas, ya sea mediante la creación de una columna de "Registro de actividad" o utilizando la función de seguimiento de cambios. La función de seguimiento de cambios en Monday.com te permite ver un historial detallado de todos los cambios realizados en un elemento, incluyendo la fecha, la hora y el usuario que realizó el cambio.

En general, el registro de actividades es importante para mantener un registro detallado y transparente de todas las actividades del proyecto. Con la función de seguimiento de cambios, las automatizaciones y las integraciones, puedes mantener un registro de las actividades relevantes y mantener a todos los miembros del equipo informados sobre el progreso del proyecto.



## Desarrollo de la Aplicación.

### Creación del flujo de la conversación del chatbot

El flujo de conversación en un chatbot se refiere a la secuencia de interacciones que se producen entre el usuario y el propio chatbot. Es un componente fundamental en el diseño y funcionamiento de un chatbot efectivo. Para desarrollar un flujo de conversación coherente y satisfactorio, hemos utilizado la herramienta de Google DialogFlow, esta herramienta permite

definir las preguntas y respuestas que ofrecerá el chatbot.

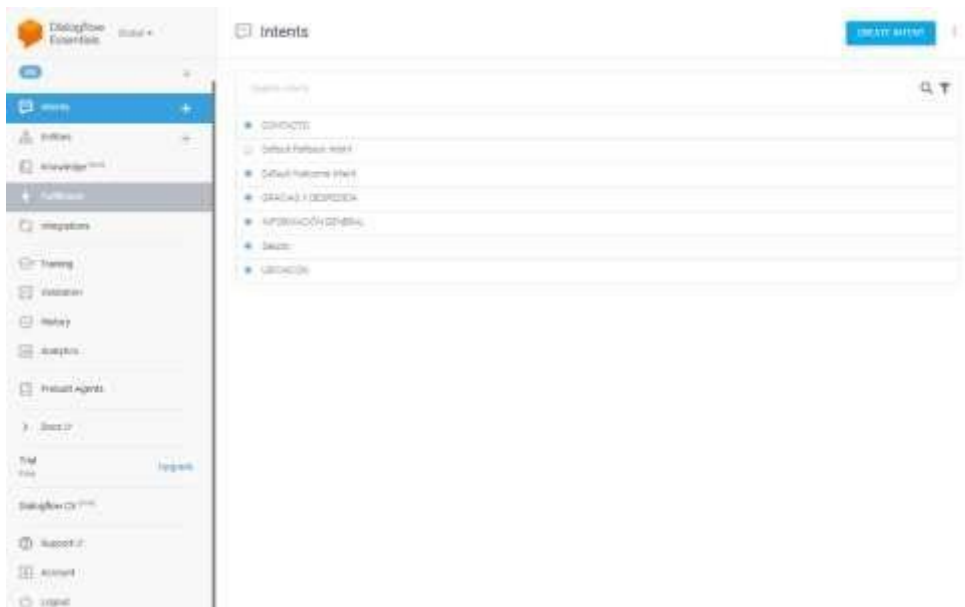
Para la creación del flujo de la conversación, hemos decidido utilizar la herramienta de Google llamada “DialogFlow”, para usar esta herramienta debemos crearnos una cuenta con nuestros datos personales.

The image displays two screenshots of the Google Cloud Platform account creation interface. The top screenshot is titled 'Paso 1 de 2 Información de la cuenta' (Step 1 of 2 Account Information). It shows a user profile for 'Sebastian Burgos' with an email address. Below this, there is a 'País' (Country) dropdown menu set to 'Ecuador'. A section titled '¿Cuál de las siguientes opciones describe mejor a tu organización o asociación?' (Which of the following options best describes your organization or association?) has a dropdown menu set to 'Prácticamente ninguna actividad' (Practically no activity). There is a checkbox for 'Acepto los Condiciones del Servicio de Google Cloud Platform' (I accept the Google Cloud Platform Service Terms) and a link to 'Ver condiciones del Servicio de la prueba gratuita, complementaria y las Condiciones del Servicio de cualquier servicio y API subsecuentes.' (View trial service conditions, complementary and the Service Conditions of any service and API subsequently). A 'Continuar' (Continue) button is at the bottom. The right side of the screenshot contains promotional text about accessing all Google Cloud Platform products, obtaining \$300 in free credit, and no automatic charges after the trial. The bottom screenshot is titled 'Paso 2 de 2 Verificación de información de pago' (Step 2 of 2 Payment information verification). It includes a warning about payment information being used for fraud detection. A section titled 'Perfil de pagos' (Payment profile) shows a selection of a payment profile. Below this, it displays the user's name 'Sebastian Burgos', their phone number, and a payment profile ID. The right side of this screenshot also contains promotional text about Google Cloud Platform products, free credit, and no automatic charges after the trial.

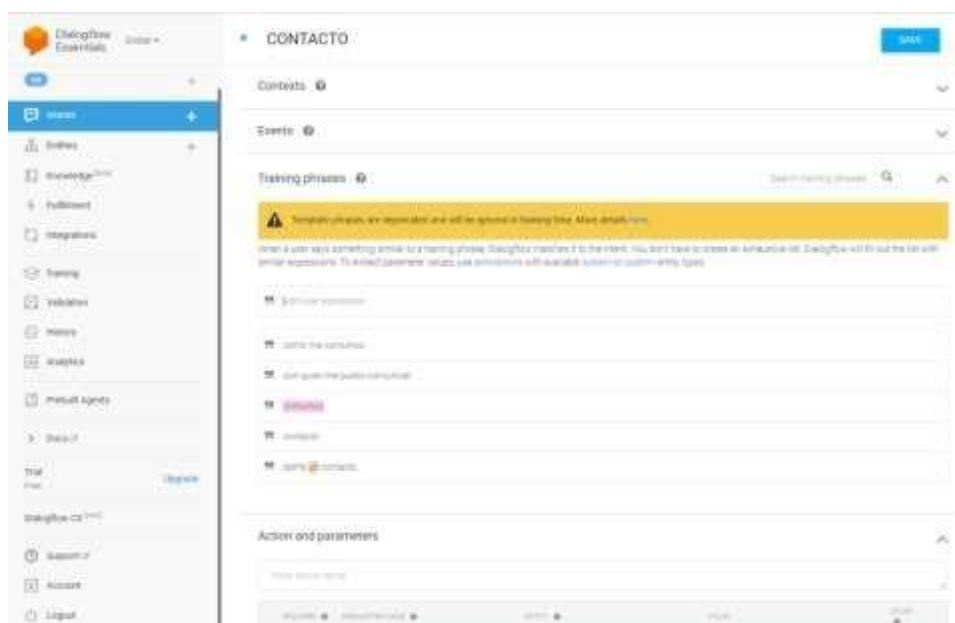
Si todo es correcto tendremos acceso al entorno de desarrollo del flujo de la conversación, donde comenzaremos a integrar las posibles entradas y salidas de las respuestas. En nuestro caso hemos definido el flujo de conversación para:

Saludo

- Despedida
- No entender la pregunta
- Ubicación
- Información General de la farmacia
- Contacto



A estos registros se los conoce como intents, en dónde por cada tema propuesto se debe incluir los inputs y los outputs



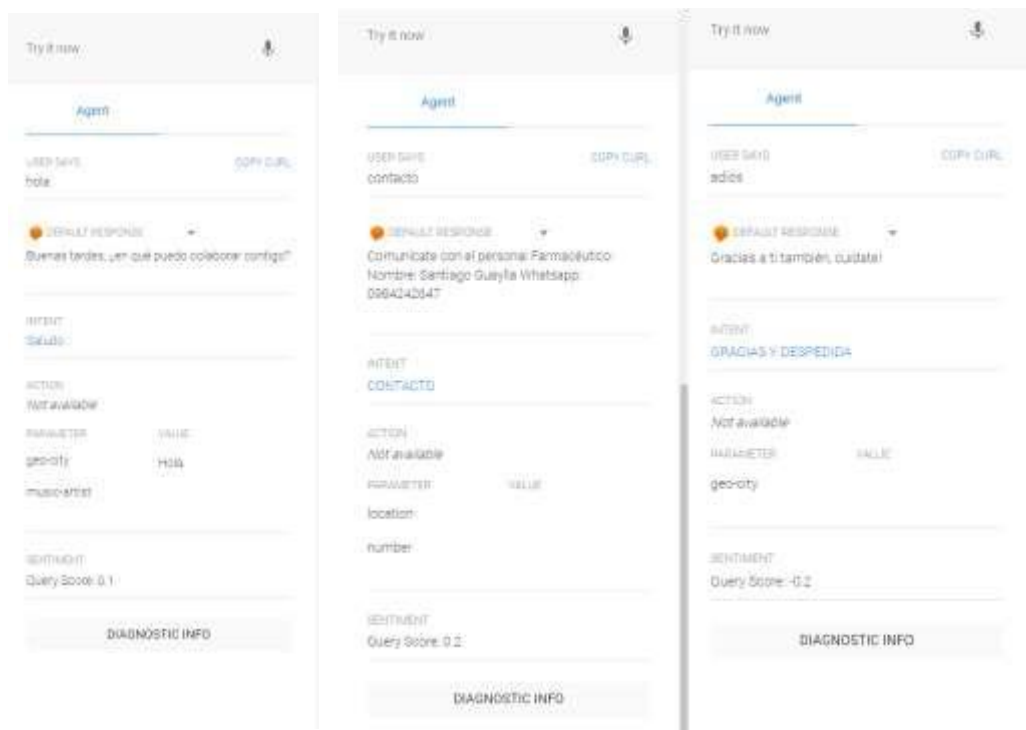
La inteligencia artificial radica en este ejemplo en el uso de variables que nos ahorrarán la creación de infinitos inputs y/o outputs, estas variables se identifican según su tipo y se visualizan en la siguiente tabla

REQUIRED	Default fallback intent	INPUT	Output	OUTPUT
<input type="checkbox"/>	number	<span style="color: orange;">@sys.number</span>	number	<input type="checkbox"/>
<input type="checkbox"/>	location	<span style="color: purple;">@sys.location</span>	location	<input type="checkbox"/>
<input type="checkbox"/>	Enter name	Enter entity	Enter value	<input type="checkbox"/>

En la tabla para este intents llamado CONTACTO se pueden identificar 2 variables las cuales dan como resultado lo siguiente:



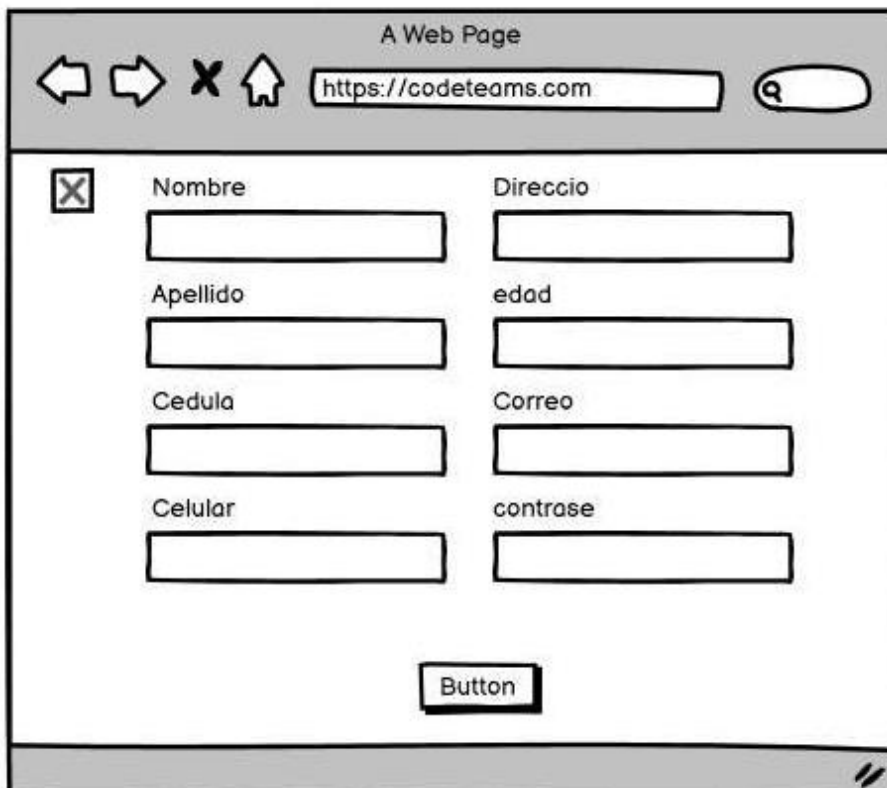
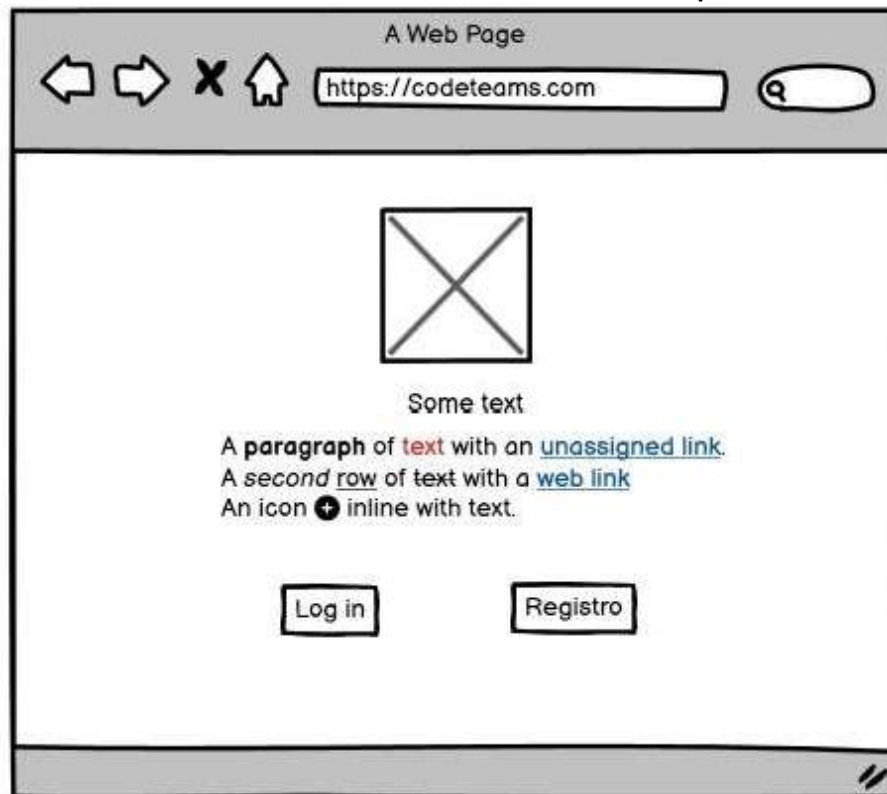
Para probar que el flujo de conversación funciona correctamente, DialogFlow nos ofrece la posibilidad de ir testeando nuestra conversación, a continuación, se muestra el funcionamiento de la misma:

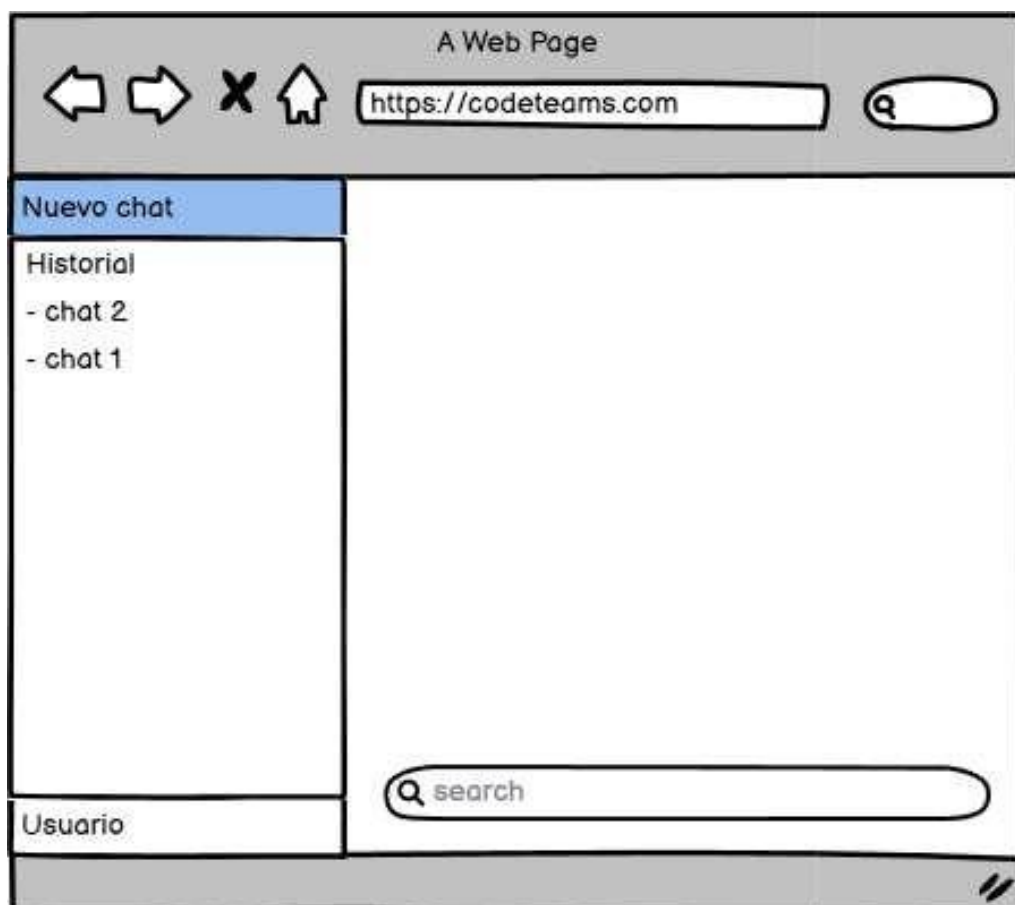
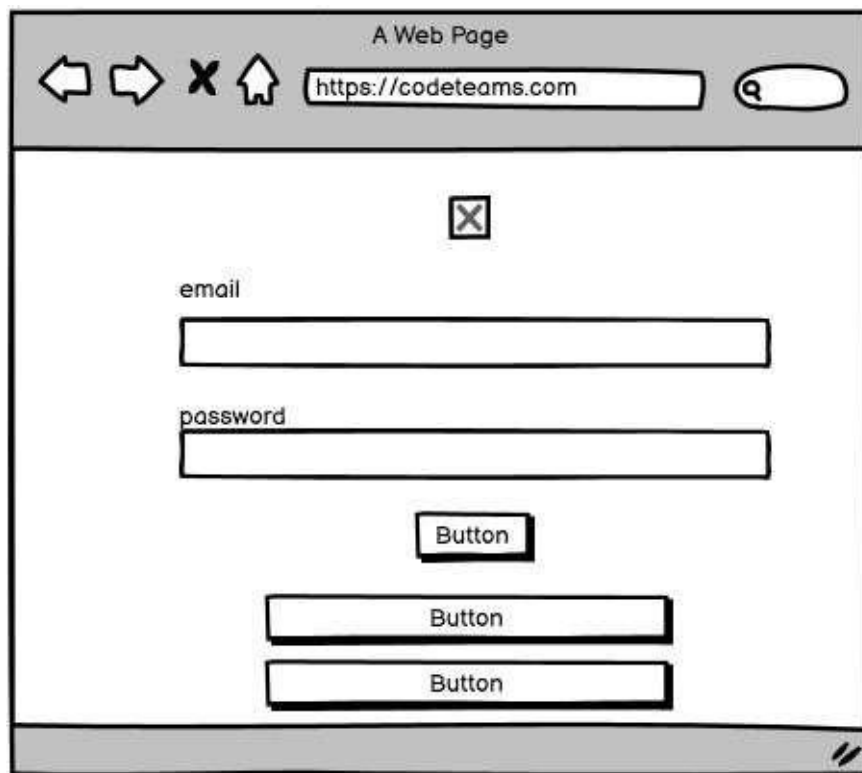


Para conectar el funcionamiento de esta herramienta con nuestra aplicación, podremos usar a futuro una API que permita la transferencia de esta información, es importante mencionar que el proceso de creación del flujo de conversación aún no está terminado, sin embargo, mostramos un avance de cómo se está desarrollando.



## Diseño de la interfaz de usuario usando de la herramienta Balsamiq



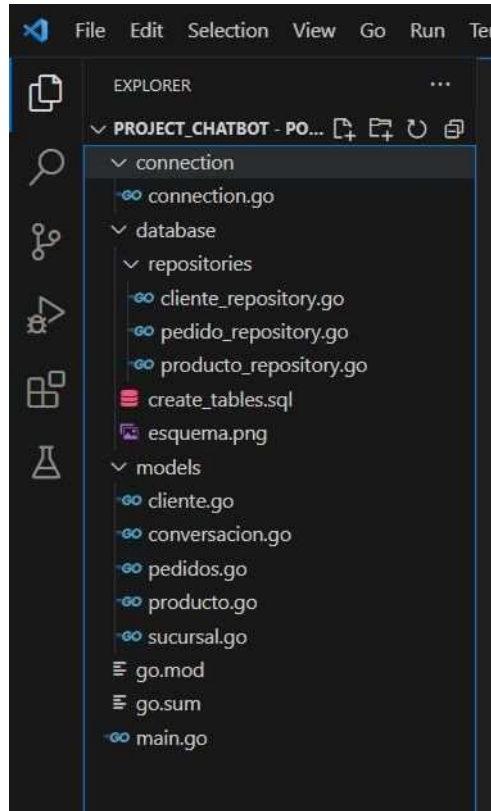








## Creación de la capa de datos.



En general, tenemos una estructura de carpetas y archivos que sigue un enfoque de diseño basado en capas o patrón de repositorio, donde los repositorios se encargan de interactuar con la base de datos y los modelos representan las entidades del sistema. El archivo principal "main.go" actúa como el punto de entrada de la aplicación y coordina la interacción entre los diferentes componentes. El archivo "create\_table.sql" se utiliza para configurar la estructura inicial de la base de datos.

Carpeta "connection" con el archivo "connection.go":

Este archivo contiene la implementación de la conexión a la base de datos. Incluye funciones o métodos para establecer la conexión utilizando la biblioteca de Go para interactuar con PostgreSQL. Aquí se establecerán los detalles de la conexión, como la dirección IP, el puerto, el nombre de usuario y la contraseña.

Carpeta "database" con los archivos "cliente\_repository.go", "pedido\_repository.go" y "producto\_repository.go":

Estos archivos representan los repositorios de datos para cada entidad (cliente, pedido, producto). Un repositorio de datos es responsable de interactuar con la base de datos para realizar operaciones relacionadas con esa entidad. Estos archivos podrían contener funciones o métodos para realizar consultas SQL y manipular los datos correspondientes en la base de datos, como insertar, actualizar, eliminar o recuperar registros.

Carpeta "models" con los archivos "cliente.go", "conversacion.go", "pedidos.go", "producto.go" y "sucursal.go":

Estos archivos contienen las definiciones de modelos o estructuras de datos para cada entidad del sistema (cliente, conversación, pedido, producto, sucursal). Estos modelos representan la estructura y los atributos de cada entidad y proporcionan una representación en código de los datos que se almacenan o se manipulan en la base de datos. Cada archivo probablemente

contendrá las definiciones de campos y métodos relacionados con cada entidad.

Archivo "main.go":

Este archivo es el punto de entrada principal de tu aplicación en Go. Aquí se puede realizar la configuración inicial, como la inicialización de la conexión a la base de datos, la configuración de enrutamiento si se utiliza un servidor web, la inicialización de servicios o controladores, etc. También se define la lógica principal de la aplicación y realizar llamadas a las funciones de los repositorios o modelos según sea necesario.

Archivo "create\_table.sql":

Este archivo es un script SQL que contiene las declaraciones para crear las tablas y definir la estructura de la base de datos. Aquí se especifican los nombres de las tablas, los campos, los tipos de datos, las restricciones y cualquier otra configuración necesaria para definir correctamente el esquema de la base de datos. Este script se ejecuta una vez al configurar la base de datos inicialmente.

### Interfaz de Usuario.



Empezamos a realizar la interfaz de nuestro programa, para ello realizamos una estructura bien organizada y semántica para la creación de la interfaz de usuario de un chatbot. El documento comienza con las etiquetas `<!DOCTYPE html>`, que especifica que el documento sigue la sintaxis de HTML. A continuación, se define la estructura básica del documento HTML con etiquetas como `<html>`, `<head>`, y `<body>`.

Dentro del `<head>`, se incluyen metadatos importantes como la codificación de caracteres y la compatibilidad con Internet Explorer, además de enlazar un archivo de estilo externo llamado "style.css" para aplicar los estilos visuales al contenido.

Dentro del `<body>`, se encuentra el contenido principal de la página, encapsulado en un elemento `<main>`. La interfaz del chatbot se divide en dos secciones principales: el "container-left" y el "container-right". En el "container-left", se encuentran botones interactivos como "New chat" y "Traductor", que permiten al usuario iniciar una nueva conversación o acceder a

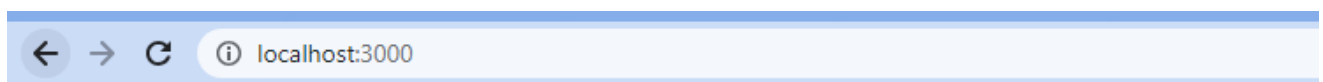
funciones específicas del chatbot. Además, se muestra información del usuario con un botón que incluye su nombre y un ícono de usuario.

En el "container-right", se encuentra el título del chatbot y un campo de entrada de texto con un ícono de enviar. Estos elementos están contenidos en divs y utilizan clases para aplicar estilos específicos definidos en el archivo de estilo.

El código HTML proporcionado sigue las mejores prácticas para crear una interfaz de usuario coherente y bien estructurada. La utilización de etiquetas semánticas y la organización lógica del contenido facilitan la comprensión y el mantenimiento del código. Al enlazar un archivo de estilo externo, se permite la separación de la estructura y la presentación visual, lo que mejora la flexibilidad y la mantenibilidad del diseño.

Además, se establecen estilos comunes para los diferentes contenedores dentro de la interfaz, como márgenes inferiores y colores de fondo, para crear una apariencia coherente y equilibrada en todo el diseño. Se establecen estilos para los botones dentro de los contenedores, como el relleno, el ancho, los bordes, el color de fondo y la transición en los cambios de estado al pasar el cursor por encima. Estos estilos permiten que los botones se destaquen visualmente y brinden una experiencia de usuario más agradable e interactiva.

## Conexión Frontend y Backend.



# ChatBot

## Cientes

### Mostrar clientes

- [Hector Daniel](#)

Se ha establecido una conexión utilizando el paquete net/http en Go. A continuación se explica el proceso de conexión paso a paso:

La función `http.HandleFunc("/", Index)` establece un controlador para la ruta raíz ("/") de la aplicación web. Esto significa que cuando se accede a la URL base del servidor, se ejecutará la función `Index`.

La función `http.HandleFunc("/clientes/cliente", MostrarCliente)` establece un controlador para la ruta `/clientes/cliente` de la aplicación web. Esto significa que cuando se accede a esta URL específica, se ejecutará la función `MostrarCliente`.


A continuación, se utiliza `http.ListenAndServe("localhost:3000", nil)` para iniciar el servidor web en el puerto 3000 de localhost. Esto significa que el servidor estará escuchando las solicitudes entrantes en ese puerto específico.

Después de iniciar el servidor, se imprime en la consola el mensaje "El servidor está corriendo en el puerto localhost:3000", lo cual es útil para verificar que el servidor se haya iniciado correctamente.

En definitiva se utiliza las funciones `http.HandleFunc` para establecer controladores para diferentes rutas en la aplicación web y luego utiliza `http.ListenAndServe` para iniciar el servidor

web en un puerto específico. Esto permite que la aplicación escuche y responda a las solicitudes entrantes en esas rutas específicas. El mensaje impreso en la consola proporciona información adicional para confirmar que el servidor se está ejecutando correctamente en el puerto especificado.

### Validación de datos.



```
PROBLEMAS 18 SALIDA TERMINAL CONSOLA DE DEPURACIÓN

Ingrese el número de opción: 1
1
Ingrese la cédula: 060570812-2
060570812-2
Ingrese la nombres: H@
Ingreso a validar nombre
El o los nombres no son válidos.
exit status 1

Ingrese el número de opción: 1
1
Ingrese la cédula: 060570812-2
060570812-2
Ingrese la nombres: Hector@ 12Nieta
Ingreso a validar nombre
El o los nombres no son válidos.
exit status 1
```

La validación de datos se realizó mediante el uso de funciones específicas del paquete capa lógica. Estas funciones toman como entrada el dato a validar y devuelven un resultado booleano que indica si el dato cumple con los criterios de validación establecidos.

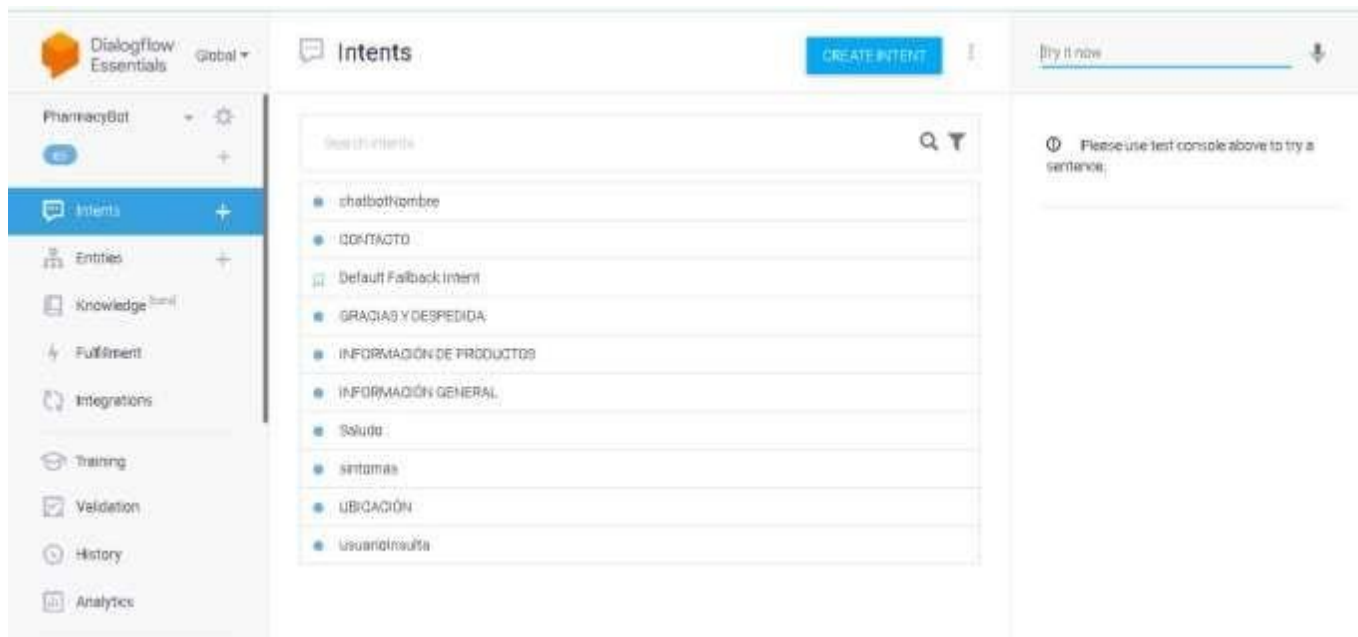
En cada caso, se llamó a la función correspondiente según el tipo de dato a validar. Si el resultado de la validación fue false, se imprimió un mensaje de error específico relacionado con el tipo de dato y se utilizó `os.Exit(1)` para finalizar la ejecución del programa con un código de salida 1, indicando un error.

En resumen, se utilizó una serie de funciones de validación específicas para cada tipo de dato, y se verificó el resultado de la validación para tomar decisiones en consecuencia. Si el dato no cumplía con los criterios de validación, se mostraba un mensaje de error y se terminaba la ejecución del programa.

### Creación de los Intents y Entities básicos para el chatbot.

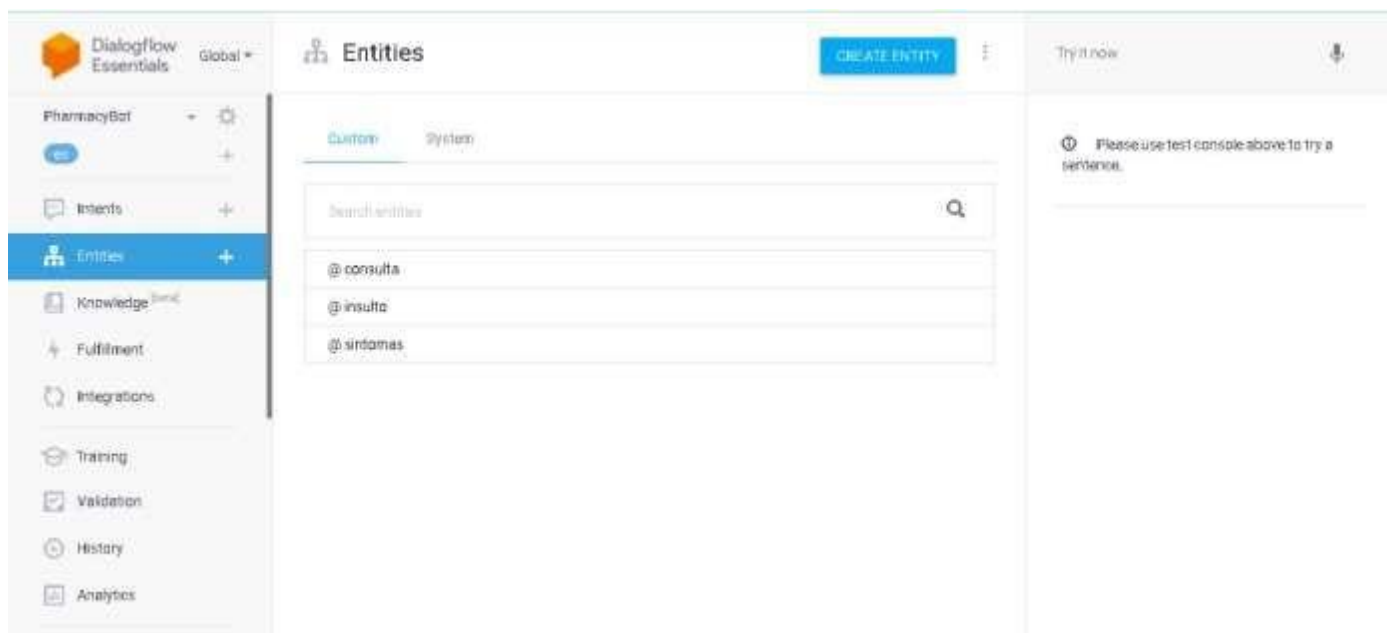
La creación de intenciones (Intents) y entidades (Entities) es un paso clave en el desarrollo de un chatbot con Dialogflow Essentials. Aquí se describe cómo se llevó a cabo este proceso:

Comenzamos por identificar las intenciones clave que los usuarios podrían tener al interactuar con el chatbot. Estas intenciones representan las acciones o solicitudes que el chatbot debe reconocer y responder. Por ejemplo, en nuestro caso estamos creando un chatbot para una farmacia, algunas intenciones podrían ser "realizar una consulta sobre un medicamento", "consultar los productos disponibles" o "solicitar una consulta".



Para la creación de Intents en Dialogflow Essentials, creamos una intención para cada acción o solicitud identificada en el paso anterior. Asignamos un nombre descriptivo a cada intención y proporcionamos ejemplos de frases que los usuarios podrían utilizar para expresar esa intención. Cuantos más ejemplos proporcionamos, mejor será el reconocimiento de la intención por parte del chatbot.

Para cada intención, definimos las respuestas que el chatbot debe proporcionar. Esto puede incluir respuestas de texto simples, respuestas enriquecidas con tarjetas o incluso respuestas dinámicas basadas en datos externos. Para ello, nos aseguramos de que las respuestas sean relevantes y útiles para los usuarios en función de la intención específica.



Continuamos identificando las entidades relevantes que pueden estar presentes en las frases de los usuarios. Las entidades representan información específica que el chatbot debe extraer, como nombres, fechas, ubicaciones, etc. Por ejemplo, en el caso de nuestro chatbot para una farmacia, las entidades pueden incluir "nombre de la farmacia", "síntomas de alguna

enfermdad", "insultos de cualquier tipo" y "consultas solicitadas".

En Dialogflow Essentials, crear entidades para cada tipo de información que deseas extraer de las frases de los usuarios proporciona ejemplos de valores que pueden corresponder a cada entidad y define sinónimos si es necesario. Esto ayudará al chatbot a comprender y capturar la información relevante de manera más precisa.

Finalmente una vez que hayamos creado las intenciones y entidades básicas, las podemos combinarlas en el flujo de diálogo. Por ejemplo, podemos asociar una intención de "realizar una consulta" con la entidad "nombre del producto" y "cantidad" para capturar la información necesaria para procesar el pedido, esta combinación estará listo para el próximo avance del proyecto.

### **Interfaz de Usuario del ChatBot e Implementación de la API.**

Para implementar una API y permitir la interacción con ella, es necesario seguir algunos pasos. En primer lugar, tenemos que diseñar la interfaz de usuario en HTML, definiendo cómo queremos que se vea y cómo deseamos que los usuarios interactúen con ella. Esto implica determinar qué elementos se necesitarán, como campos de entrada de texto, botones o menús desplegables.

Además, es importante familiarizarse con la documentación de la API que vamos a utilizar. Esta documentación proporcionará información detallada sobre los puntos finales disponibles, los métodos HTTP que vamos a utilizar, los parámetros requeridos y las respuestas esperadas. Es fundamental comprender estos aspectos para poder realizar las solicitudes adecuadas a la API.



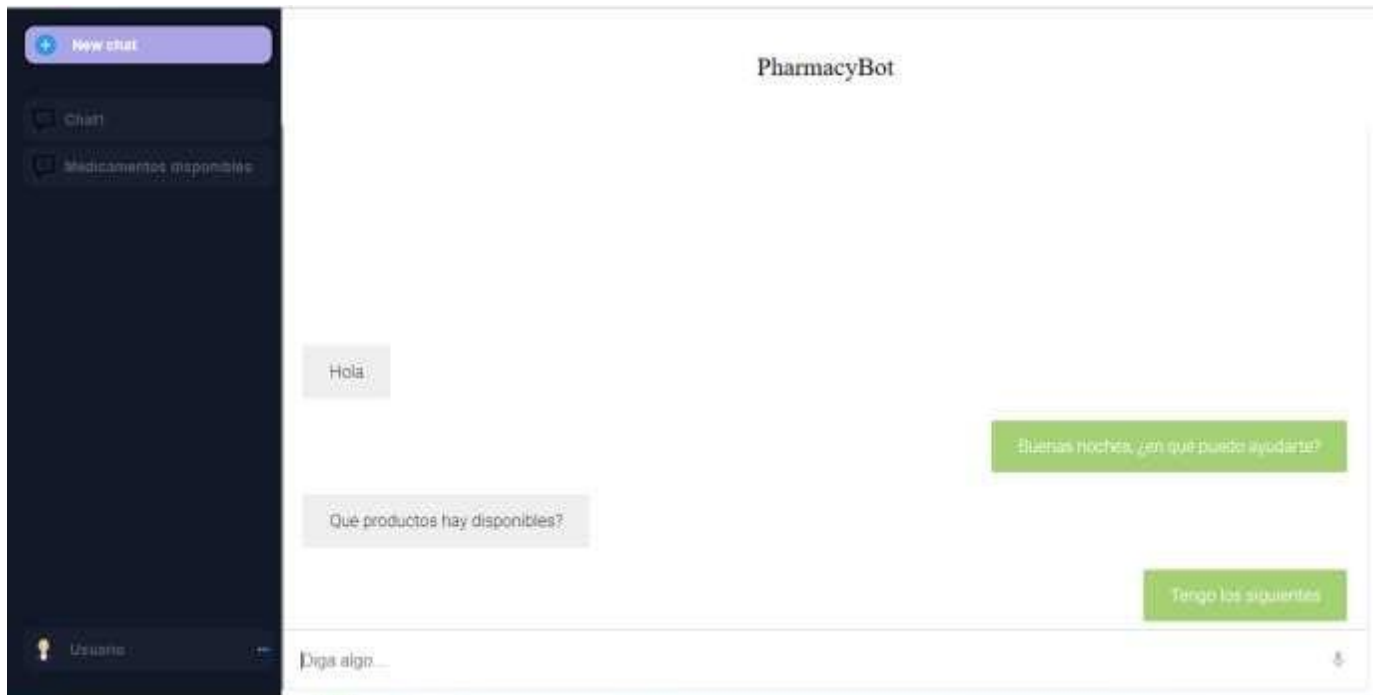
Una vez que tengamos claro los detalles de la API y la implementemos, podemos utilizar JavaScript para realizar las solicitudes desde el lado del cliente. Para ello hemos utilizado la API para facilitar las solicitudes HTTP asegurándonos de incluir los parámetros necesarios según la documentación de la API y de procesar las respuestas de manera adecuada.

Además, es esencial implementar el manejo de errores y validaciones apropiadas. Esto implica verificar si la respuesta de la API fue exitosa, manejar los errores en caso de que la solicitud



falle y validar los datos ingresados por los usuarios antes de enviarlos a la API. De esta manera, se garantiza la integridad de los datos y se proporciona una mejor experiencia de usuario.

Es importante actualizar la interfaz de usuario de forma dinámica en base a las respuestas de la API. Esto implica utilizar JavaScript para mostrar u ocultar elementos, actualizar datos en formularios o mostrar mensajes de confirmación. Esto permite una interacción fluida y en tiempo real con la API. Es fundamental realizar pruebas exhaustivas para garantizar que la interacción con la API funcione correctamente. Se deben verificar aspectos como el envío y recepción correcta de los datos, el manejo adecuado de las respuestas y la ausencia de errores inesperados.



Una vez implementado la API en la interfaz de usuario y configurado la interacción con el chatbot, se puede comenzar a hacer preguntas y recibir respuestas del chatbot en tiempo real. Al interactuar con la interfaz de usuario, podrás ingresar las preguntas o mensajes que desees enviar al chatbot. Estos mensajes se enviarán a la API a través de las solicitudes correspondientes, generalmente utilizando el método POST para enviar los datos al punto final de la API designado para recibir las preguntas.

La API procesará la solicitud y enviará la pregunta al chatbot para que genere una respuesta. El chatbot procesará la pregunta utilizando algoritmos de procesamiento del lenguaje natural y otras técnicas para comprender la intención del usuario y generar una respuesta relevante. Una vez que la API reciba la respuesta del chatbot, la devolverá a tu interfaz de usuario a través de la respuesta a la solicitud. Nuestra aplicación deberá procesar la respuesta y mostrarla en la interfaz para que el usuario pueda verla. Este ciclo de enviar preguntas, recibir respuestas del chatbot a través de la API y mostrar las respuestas en la interfaz se repetirá cada vez que el usuario envíe una pregunta o mensaje al chatbot.

Obtenemos respuestas de acuerdo a las preguntas que le hagamos al chatbot y a la lógica y capacidad de procesamiento del chatbot en sí. El chatbot utiliza algoritmos de procesamiento del lenguaje natural y técnicas de inteligencia artificial para comprender el contexto y la intención detrás de las preguntas formuladas. Cuando le haces una pregunta al chatbot, este procesa la entrada de texto utilizando técnicas como análisis gramatical, extracción de

entidades y análisis de sentimientos para comprender el significado y la intención subyacente. Luego, el chatbot busca en su base de conocimientos o en su conjunto de datos para encontrar la respuesta más adecuada a la pregunta.

La calidad y precisión de las respuestas del chatbot dependerá en gran medida de la cantidad y calidad de datos con los que haya sido entrenado, así como de los algoritmos y técnicas utilizadas en su desarrollo. Un chatbot bien entrenado y diseñado puede proporcionar respuestas útiles y relevantes a una amplia gama de preguntas. Es importante tener en cuenta que los chatbots tienen limitaciones y pueden no entender o responder correctamente a todas las preguntas. Si la pregunta es demasiado compleja o está fuera del alcance del chatbot, es posible que no proporcione una respuesta satisfactoria. Además, los chatbots pueden ser programados para tener interacciones más dinámicas y personalizadas. Pueden recordar información anteriormente proporcionada por el usuario y utilizarla para contextualizar las respuestas posteriores. Esto permite una experiencia más fluida y personalizada para los usuarios.

### Arquitectura de la Información.

Como podemos observar, la iconografía cuenta con bordes redondeados los cuales ayudan en la experiencia del usuario al proporcionar una apariencia visual suave y agradable, reducir la sensación de agresividad, dirigir la atención al contenido principal y mejorar la accesibilidad táctil. Estos bordes contribuyen a crear una interfaz amigable y acogedora, al tiempo que minimizan distracciones visuales y facilitan la interacción con el chatbot, especialmente en dispositivos móviles.



The image shows a login interface with a blue robot icon at the top. Below the icon is the text "Bienvenido De Nuevo". There are two input fields: "Direccion email" and "Contraseña". Below these is a prominent green button labeled "Continue". Under the button is a link that says "¿No tiene una cuenta? Regístrate". A horizontal line separates this from a section containing a small "O" icon. Below that is another horizontal line, and at the bottom is a button with the Google logo and the text "Continue con Google". All elements have rounded corners.

Fig1.- Menú de Login

En primer lugar, el verde se asocia comúnmente con conceptos como naturaleza, frescura, tranquilidad y éxito, lo que puede transmitir una sensación positiva y de confianza al usuario al interactuar con el chatbot. Además, el verde es un color ampliamente reconocido como indicador de acción o confirmación, por lo que puede ser apropiado para botones que realizan acciones específicas dentro del chatbot.


Se ha tratado de mantener un buen contraste con el objetivo de mejorar la claridad y legibilidad para que los botones sean fácilmente identificables y accesibles

  
Bienvenido al ChatBot  
Accede a tu cuenta para continuar

[Iniciar Sesión](#) [Regístrate](#)

Fig2. Opciones de Inicio

Para el formulario de registro hemos decidido utilizar los mismos bordes redondeados. Según la información de la arquitectura de la información esto puede contribuir a una experiencia de usuario más agradable y acogedora, especialmente en aplicaciones o sistemas donde los usuarios pasan mucho tiempo interactuando con inputs o formularios.

  
Crea una cuenta

Cédula:

Nombres:

Apellidos:

Dirección:

Celular:

Email:

Contraseña:

Fecha de nacimiento:

Sexo:

[Registrar](#)

Ya tienes una cuenta? [Inicia Sesión](#)

[O](#)


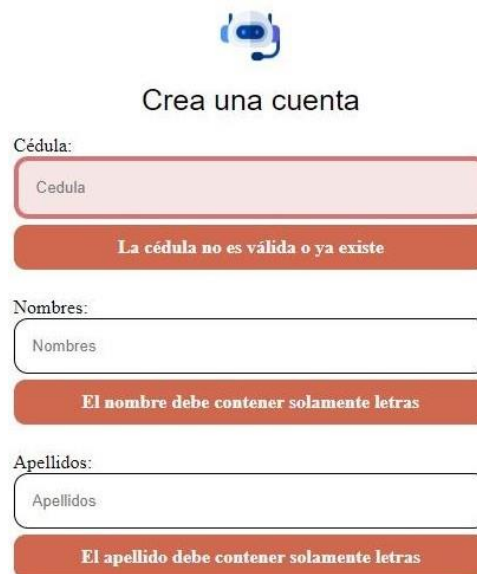
 Continúe con Google

Fig3.- Formulario de registro

Las alertas o mensajes que indican información de error son de suma importancia para que el usuario entienda que el valor que ha ingresado no es válido, El arquitecto de la información

podría considerar el color rojo como una opción efectiva para indicar errores en la interfaz del chatbot. El rojo se asocia comúnmente con alertas, advertencias y errores, lo que ayuda a transmitir visualmente la gravedad de la situación al usuario. Utilizar el color rojo para resaltar los errores en los mensajes o campos de entrada puede ayudar a captar la atención del usuario de manera rápida y efectiva. Sin embargo, es importante utilizar el color rojo de manera cuidadosa y equilibrada, evitando que sea demasiado abrumador o agresivo. Además, es recomendable acompañar el color rojo con mensajes claros y descriptivos que expliquen el error y ofrezcan una solución o instrucciones para corregirlo, brindando así una experiencia de usuario más completa y útil.



The image shows a user registration form titled "Crea una cuenta". It contains three input fields, each followed by a red error message box:

- Cédula:** The input field contains the placeholder text "Cedula". Below it, a red box displays the message: "La cédula no es válida o ya existe".
- Nombres:** The input field contains the placeholder text "Nombres". Below it, a red box displays the message: "El nombre debe contener solamente letras".
- Apellidos:** The input field contains the placeholder text "Apellidos". Below it, a red box displays the message: "El apellido debe contener solamente letras".

Fig4.- Mensajes de error en campos incorrectos

Un cuadro de herramientas para el usuario corresponde a un cuadro de navegación establecido por la arquitectura de la información donde se enfatiza la importancia de diseñar una estructura de navegación coherente y consistente, que refleje la mentalidad del usuario y le permita comprender intuitivamente dónde se encuentran las herramientas relevantes. Esto implica considerar la agrupación lógica de las herramientas, la secuencia de navegación y la visibilidad de las opciones disponibles.

Por eso hemos planteado nuestro cuadro de navegación de la siguiente manera

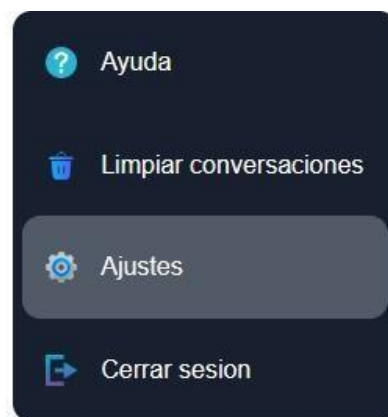


Fig5.- Cuadro de Navegación

La interfaz del chatbot debe ser simple y minimalista, Estos conceptos se basan en principios de organización, accesibilidad y claridad de la información. Algunos de los fundamentos teóricos que respaldan estos conceptos son:

**Principio de jerarquía de la información:** La arquitectura de la información promueve la organización jerárquica de la información, donde los elementos más importantes y relevantes se destacan y se presentan de manera clara y concisa. La simplicidad se logra al priorizar y presentar solo la información esencial, evitando la sobrecarga de contenido innecesario.

**Teoría de la carga cognitiva:** La carga cognitiva se refiere a la cantidad de esfuerzo mental requerido para procesar la información. Un diseño simple y minimalista reduce la carga cognitiva al eliminar elementos visuales innecesarios, simplificar la navegación y presentar la información de manera clara y concisa. Esto facilita la comprensión y la toma de decisiones por parte del usuario.

**Principio de diseño centrado en el usuario:** La arquitectura de la información se basa en comprender las necesidades, metas y capacidades de los usuarios. Un diseño simple y minimalista se enfoca en ofrecer una experiencia intuitiva y sin complicaciones, donde los usuarios pueden acceder y utilizar la aplicación web de manera eficiente y efectiva.

**Diseño orientado a la usabilidad:** La arquitectura de la información considera la usabilidad como un aspecto fundamental del diseño. Un enfoque simple y minimalista facilita la usabilidad al reducir la curva de aprendizaje, permitir una navegación clara y directa, y minimizar la posibilidad de errores y confusiones por parte del usuario.

En base a estos conceptos hemos propuesto la siguiente interfaz para el uso del chatbot:



Fig6.- Interfaz del Chatbot

Implementación con la validación de datos.



## Crea una cuenta

Cédula:

0605708122

**La cédula no es válida o ya existe**

Nombres:

Hec&\*

**El nombre debe contener solamente letras**

Apellidos:

Nieto455

**El apellido debe contener solamente letras**

Dirección:

La Floresta|

La validación de datos se realizó mediante el uso de funciones específicas del paquete capa lógica. Estas funciones toman como entrada el dato a validar y devuelven un resultado booleano que indica si el dato cumple con los criterios de validación establecidos.

En cada caso, se llamó a la función correspondiente según el tipo de dato a validar. Si el resultado de la validación fue false, se imprimió un mensaje de error específico relacionado con el tipo de dato y se utilizó `os.Exit(1)` para finalizar la ejecución del programa con un código de salida 1, indicando un error.

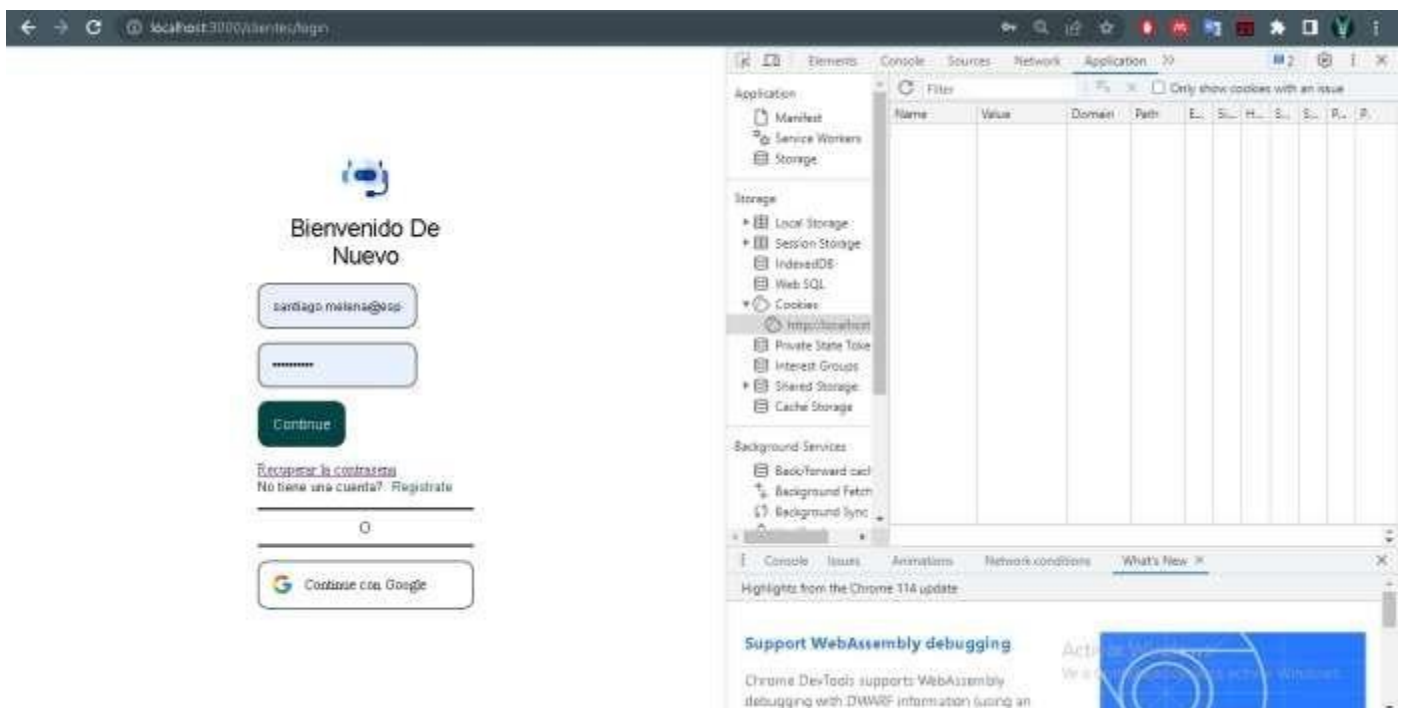
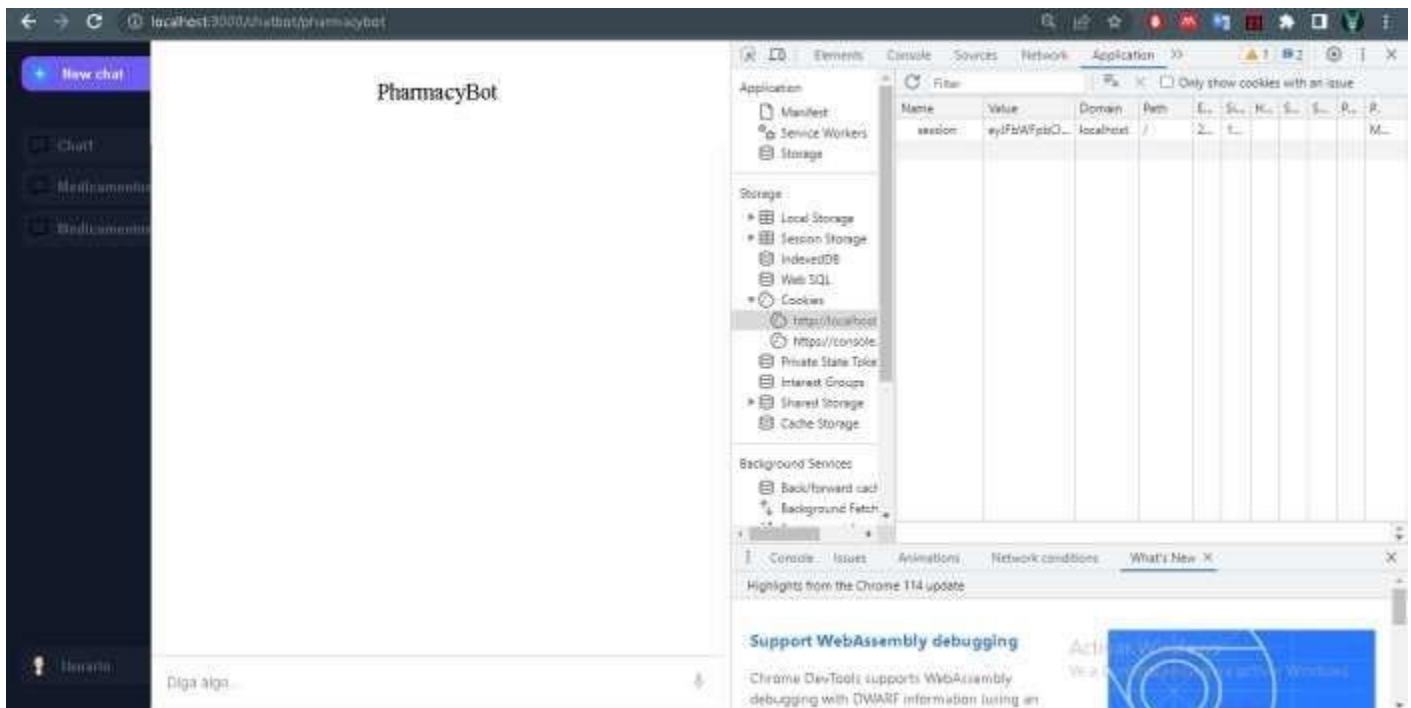
En resumen, se utilizó una serie de funciones de validación específicas para cada tipo de dato, y se verificó el resultado de la validación para tomar decisiones en consecuencia. Si el dato no cumplía con los criterios de validación, se mostraba un mensaje de error y se terminaba la ejecución del programa.

### Inicio de sesión (Cookie).

En el contexto del objetivo específico de desarrollar un proceso de autenticación que garantice los datos correctos de los usuarios al momento de iniciar sesión, se puede mencionar el concepto de "galleta" (cookie en inglés) como una técnica comúnmente utilizada en el ámbito de la seguridad y la autenticación web.

Una galleta es un pequeño archivo de texto que se almacena en el dispositivo del usuario

cuando accede a un sitio web. La galleta contiene información relevante, como identificadores únicos o datos de sesión, que se utilizan para autenticar al usuario en visitas posteriores al sitio



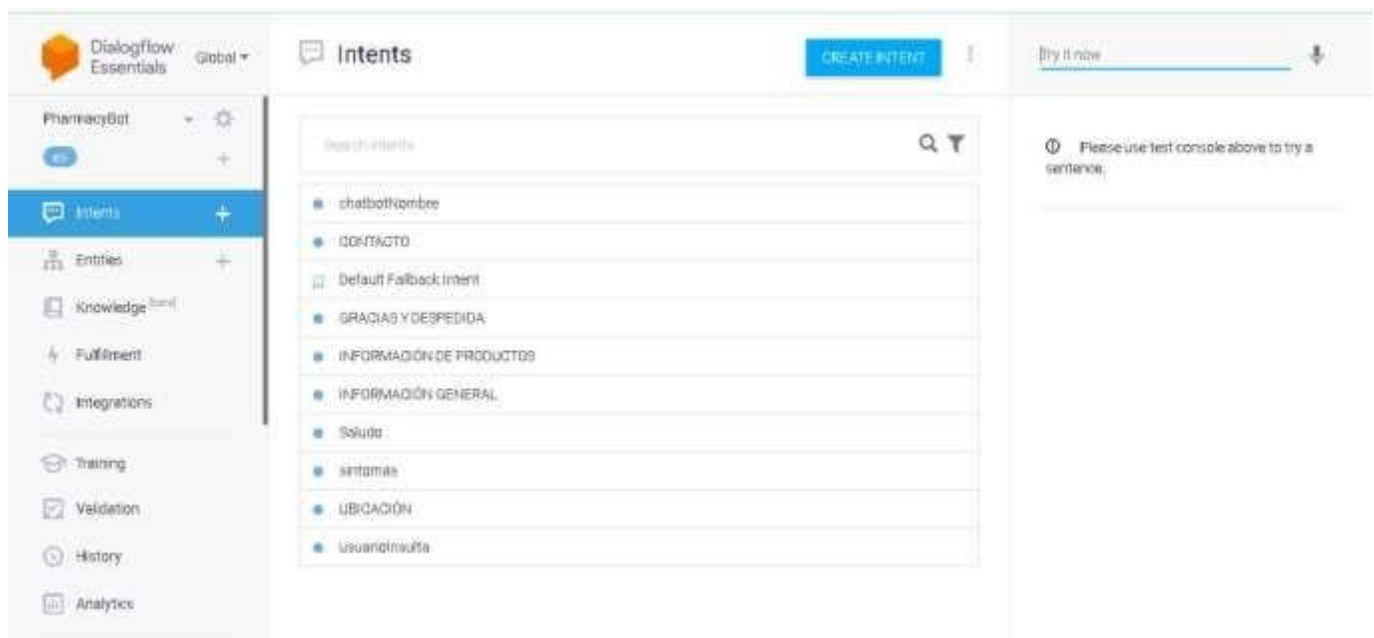
Al implementar un inicio de sesión seguro, es común utilizar galletas para mantener el estado de autenticación del usuario. Una vez que el usuario se autentica con éxito, se genera una galleta con un identificador único y se almacena en el dispositivo del usuario. En cada visita posterior al sitio, el servidor web verifica la presencia y validez de la galleta para mantener al usuario autenticado sin necesidad de ingresar nuevamente las credenciales. Es importante resaltar que el uso de galletas para el inicio de sesión debe cumplir con buenas prácticas de seguridad.

Al considerar la implementación de una galleta para el inicio de sesión, se deben tener en cuenta las regulaciones de privacidad y seguridad de datos vigentes en la jurisdicción en la que opera el sistema, como el Reglamento General de Protección de Datos (RGPD) en la Unión Europea.

### Creación de los Intents y Entities avanzados para el chatbot.

La creación de intenciones (Intents) y entidades (Entities) es un paso clave en el desarrollo de un chatbot con Dialogflow Essentials. Aquí se describe cómo se llevó a cabo este proceso:

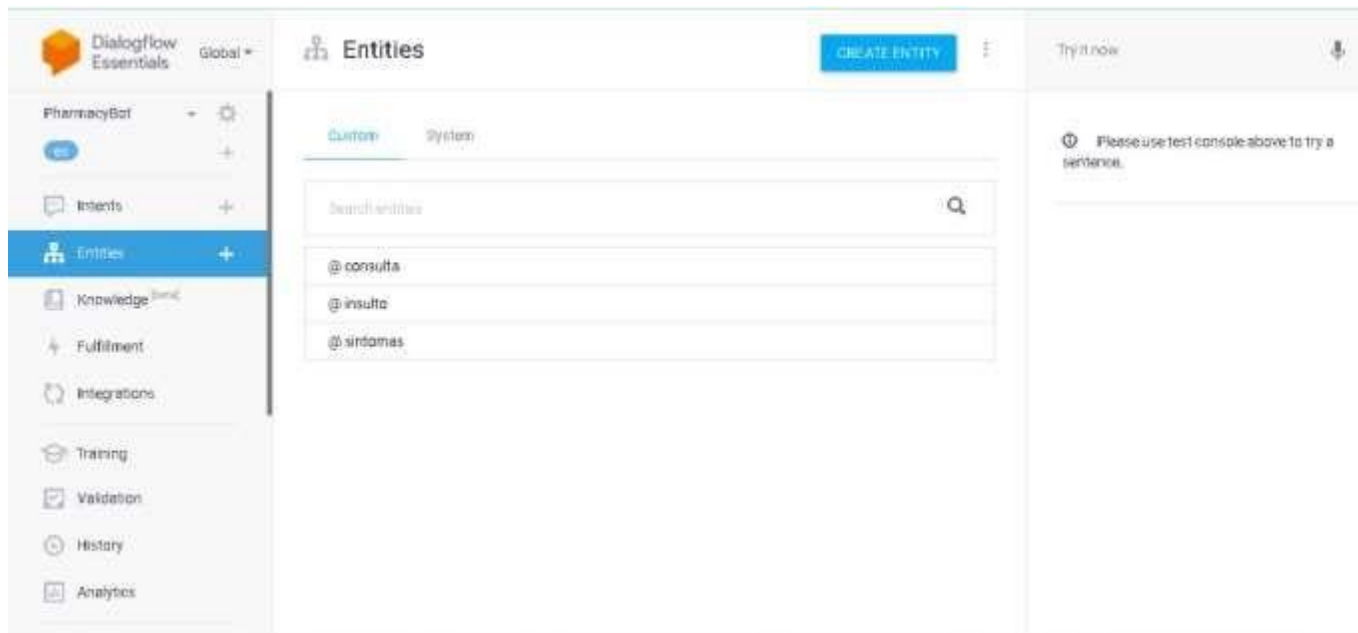
Comenzamos por identificar las intenciones clave que los usuarios podrían tener al interactuar con el chatbot. Estas intenciones representan las acciones o solicitudes que el chatbot debe reconocer y responder. Por ejemplo, en nuestro caso estamos creando un chatbot para una farmacia, algunas intenciones podrían ser "realizar una consulta sobre un medicamento", "consultar los productos disponibles" o "solicitar una consulta".



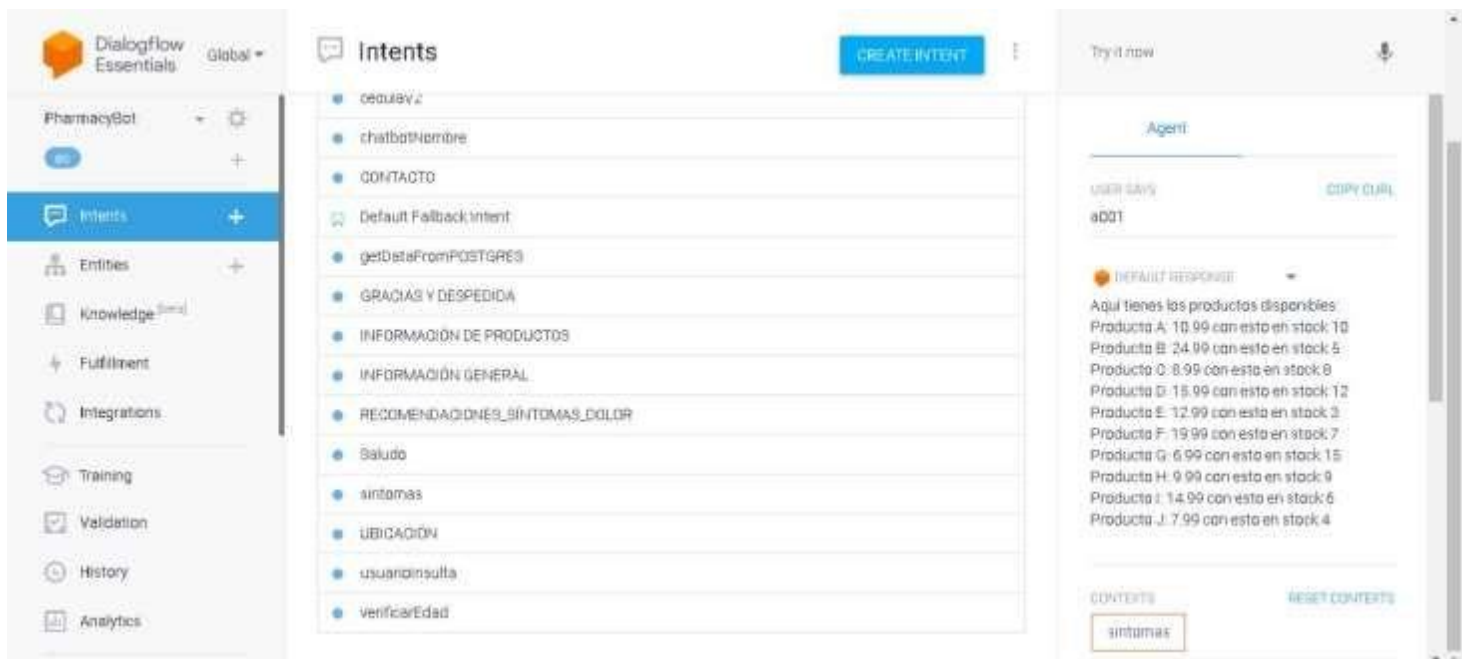
Para la creación de Intents en Dialogflow Essentials, creamos una intención para cada acción o solicitud identificada en el paso anterior. Asignamos un nombre descriptivo a cada intención y proporcionamos ejemplos de frases que los usuarios podrían utilizar para expresar esa intención. Cuantos más ejemplos proporcionamos, mejor será el reconocimiento de la intención por parte del chatbot.

Para cada intención, definimos las respuestas que el chatbot debe proporcionar. Esto puede incluir respuestas de texto simples, respuestas enriquecidas con tarjetas o incluso respuestas dinámicas basadas en datos externos. Para ello, nos aseguramos de que las respuestas sean relevantes y útiles para los usuarios en función de la intención específica.





Continuamos identificando las entidades relevantes que pueden estar presentes en las frases de los usuarios. Las entidades representan información específica que el chatbot debe extraer, como nombres, fechas, ubicaciones, etc. Por ejemplo, en el caso de nuestro chatbot para una farmacia, las entidades pueden incluir "nombre de la farmacia", "síntomas de alguna enfermedad", "insultos de cualquier tipo" y "consultas solicitadas".

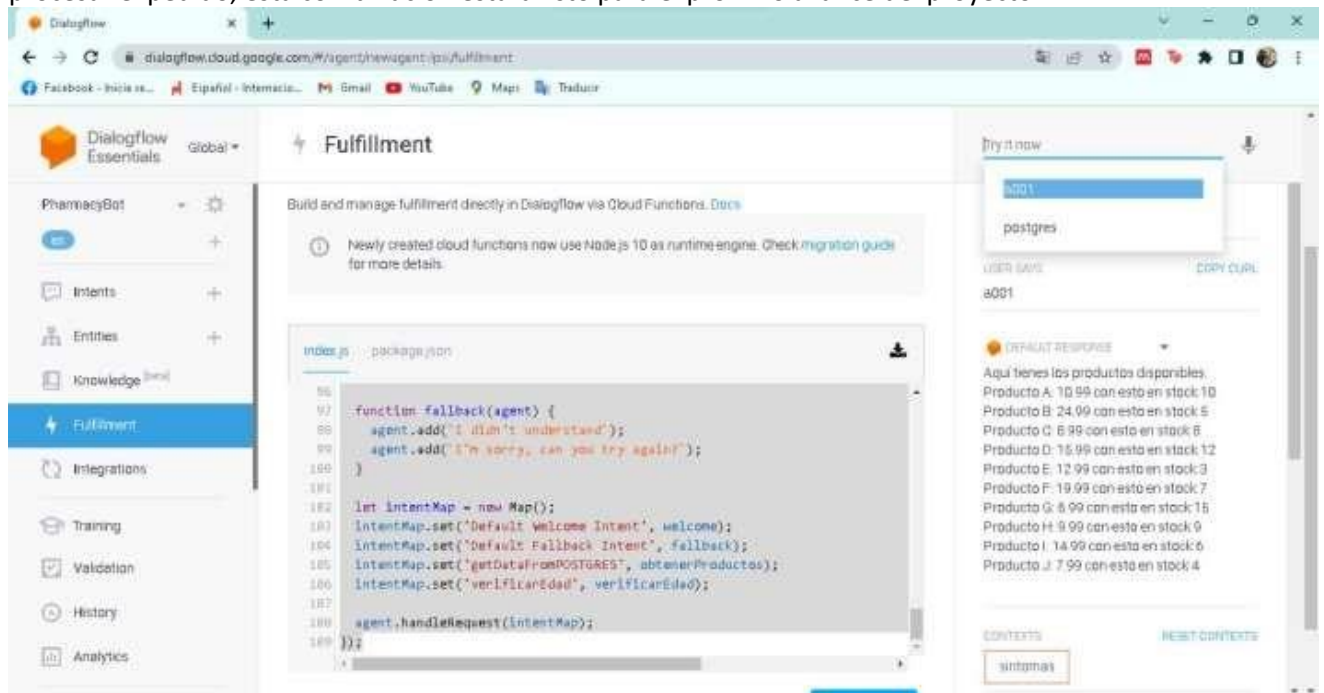


En Dialogflow Essentials, crear entidades para cada tipo de información que deseas extraer de las frases de los usuarios proporciona ejemplos de valores que pueden corresponder a cada entidad y define sinónimos si es necesario. Esto ayudará al chatbot a comprender y capturar la información relevante de manera más precisa.

### Implementación de la base de datos de la farmacia para el dialogo de flujo.

Una vez que hayamos creado las intenciones y entidades, las podemos combinarlas en el flujo de diálogo. Por ejemplo, podemos asociar una intención de "realizar una consulta" con la

entidad "nombre del producto" y "cantidad" para capturar la información necesaria para procesar el pedido, esta combinación estará listo para el próximo avance del proyecto.



Posteriormente, se procederá a implementar la base de datos de la farmacia, la cual permitirá almacenar y recuperar eficientemente los datos necesarios para el flujo de diálogo de la aplicación. Se implementará la estructura de base de datos ya definida por la farmacia, además, se establecerán relaciones entre las entidades y se optimizarán las consultas para asegurar un rendimiento más óptimo.

### Interfaz de Usuario de la aplicación.

El desarrollo de la interfaz de la aplicación se refiere al desarrollo y diseño de la parte visual y funcional que permite a los usuarios interactuar con ella de manera intuitiva y efectiva. Esta implementación implica varias etapas y consideraciones. En primer lugar, se realiza el diseño de la interfaz de usuario (UI, por sus siglas en inglés) de la aplicación. Esto implica definir la apariencia visual, el diseño de la disposición de elementos, los colores, las tipografías y otros aspectos visuales que conformarán la interfaz. El objetivo es crear una interfaz atractiva, coherente con la identidad de la marca y que proporcione una experiencia de usuario satisfactoria.



Bienvenido al ChatBot

Accede a tu cuenta para continuar

Iniciar Sesión

Regístrate

Una vez diseñada la interfaz, se procede al desarrollo de la misma utilizando tecnologías y herramientas apropiadas. En este caso HTML, CSS y JavaScript. Durante esta etapa, se

implementan los diferentes elementos de la interfaz, como campos de entrada de texto, botones, menús desplegables, imágenes u otros elementos interactivos necesarios.



## Crea una cuenta

Cédula:

Cedula

Nombres:

Nombres

Apellidos:

Apellidos

Dirección:

Direccion

Célular:

Celular

Email:

Correo Electronico

Contraseña:

Contrasena

Fecha de nacimiento:

dd/mm/aaaa


Sexo:

Seleccione el sexo

Registrar

Ya tienes una cuenta? Inicia Sesion

O

 Continue con Google



## Bienvenido De Nuevo

Continue

[¿No tiene una cuenta? Regístrate](#)

O



Continue con Google

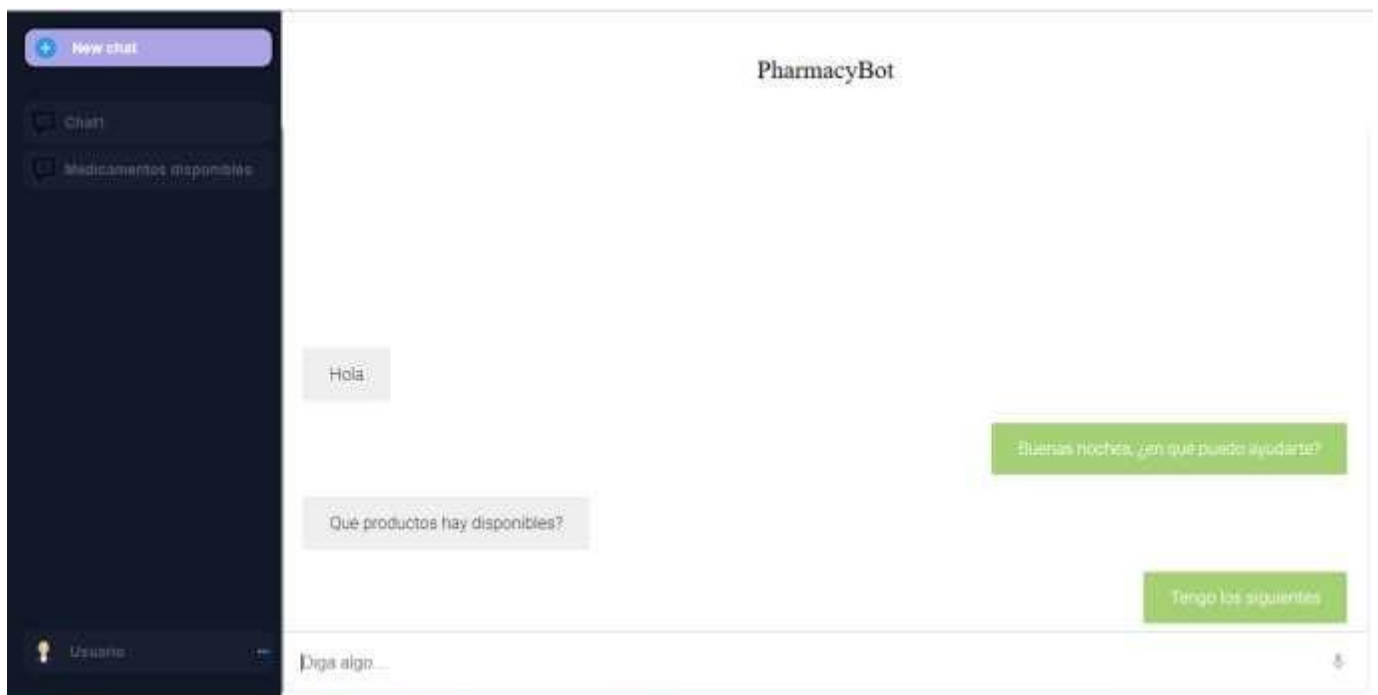
Además del aspecto visual, se debe prestar atención a la funcionalidad de la interfaz. Esto implica implementar la lógica necesaria para capturar y procesar las interacciones de los usuarios, como el envío de mensajes o la selección de opciones. Se establecen los mecanismos para transmitir los mensajes del usuario al chatbot y recibir las respuestas correspondientes, lo cual puede involucrar la integración con la API del chatbot.



Una vez que tengamos claro los detalles de la API y la implementemos, podemos utilizar JavaScript para realizar las solicitudes desde el lado del cliente. Para ello hemos utilizado la API para facilitar las solicitudes HTTP asegurándonos de incluir los parámetros necesarios según la documentación de la API y de procesar las respuestas de manera adecuada. Además, es

esencial implementar el manejo de errores y validaciones apropiadas. Esto implica verificar si la respuesta de la API fue exitosa, manejar los errores en caso de que la solicitud falle y validar los datos ingresados por los usuarios antes de enviarlos a la API. De esta manera, se garantiza la integridad de los datos y se proporciona una mejor experiencia de usuario.

Es importante actualizar la interfaz de usuario de forma dinámica en base a las respuestas de la API. Esto implica utilizar JavaScript para mostrar u ocultar elementos, actualizar datos en formularios o mostrar mensajes de confirmación. Esto permite una interacción fluida y en tiempo real con la API. Es fundamental realizar pruebas exhaustivas para garantizar que la interacción con la API funcione correctamente. Se deben verificar aspectos como el envío y recepción correcta de los datos, el manejo adecuado de las respuestas y la ausencia de errores inesperados.



Una vez implementado la API en la interfaz de usuario y configurado la interacción con el chatbot, se puede comenzar a hacer preguntas y recibir respuestas del chatbot en tiempo real. Al interactuar con la interfaz de usuario, podrás ingresar las preguntas o mensajes que desees enviar al chatbot. Estos mensajes se enviarán a la API a través de las solicitudes correspondientes, generalmente utilizando el método POST para enviar los datos al punto final de la API designado para recibir las preguntas.

La API procesará la solicitud y enviará la pregunta al chatbot para que genere una respuesta. El chatbot procesará la pregunta utilizando algoritmos de procesamiento del lenguaje natural y otras técnicas para comprender la intención del usuario y generar una respuesta relevante. Una vez que la API reciba la respuesta del chatbot, la devolverá a tu interfaz de usuario a través de la respuesta a la solicitud. Nuestra aplicación deberá procesar la respuesta y mostrarla en la interfaz para que el usuario pueda verla. Este ciclo de enviar preguntas, recibir respuestas del chatbot a través de la API y mostrar las respuestas en la interfaz se repetirá cada vez que el usuario envíe una pregunta o mensaje al chatbot.

Obtenemos respuestas de acuerdo a las preguntas que le hagamos al chatbot y a la lógica y capacidad de procesamiento del chatbot en sí. El chatbot utiliza algoritmos de procesamiento del lenguaje natural y técnicas de inteligencia artificial para comprender el contexto y la

intención detrás de las preguntas formuladas. Cuando le haces una pregunta al chatbot, este procesa la entrada de texto utilizando técnicas como análisis gramatical, extracción de entidades y análisis de sentimientos para comprender el significado y la intención subyacente. Luego, el chatbot busca en su base de conocimientos o en su conjunto de datos para encontrar la respuesta más adecuada a la pregunta.

La calidad y precisión de las respuestas del chatbot dependerá en gran medida de la cantidad y calidad de datos con los que haya sido entrenado, así como de los algoritmos y técnicas utilizadas en su desarrollo. Un chatbot bien entrenado y diseñado puede proporcionar respuestas útiles y relevantes a una amplia gama de preguntas. Es importante tener en cuenta que los chatbots tienen limitaciones y pueden no entender o responder correctamente a todas las preguntas. Si la pregunta es demasiado compleja o está fuera del alcance del chatbot, es posible que no proporcione una respuesta satisfactoria. Además, los chatbots pueden ser programados para tener interacciones más dinámicas y personalizadas. Pueden recordar información anteriormente proporcionada por el usuario y utilizarla para contextualizar las respuestas posteriores. Esto permite una experiencia más fluida y personalizada para los usuarios.

### Vonage Nexmo.

El protocolo SMTP (Simple Mail Transfer Protocol) es utilizado para enviar correos electrónicos desde una aplicación o servidor. A continuación, se presentan algunas de las utilidades y casos de uso comunes de SMTP:



Google Cuenta

## ← Contraseñas de aplicaciones

Las contraseñas de aplicación te permiten iniciar sesión en tu cuenta de Google desde aplicaciones instaladas en dispositivos que no admiten la verificación en dos pasos. No tendrás que recordarlas porque solo tienes que introducirlas una vez. [Más información](#)

Tus contraseñas de aplicación		
Nombre	Fecha de creación	Último uso
PharmacyBot	21/30	-

Selecciona la aplicación y el dispositivo para los que quieres generar la contraseña de aplicación.

Seleccionar aplicación    Seleccionar dispositivo

GENERAR

**Envío de correos electrónicos:** La principal utilidad de SMTP es permitir el envío de correos electrónicos desde una aplicación o servidor hacia los destinatarios deseados. Esto es especialmente útil en aplicaciones web o sistemas que necesitan enviar notificaciones por correo electrónico, como confirmaciones de registro, restablecimiento de contraseñas, alertas, boletines informativos, entre otros.

**Comunicación empresarial:** SMTP es ampliamente utilizado en el entorno empresarial para la comunicación interna y externa a través del correo electrónico. Permite a las empresas enviar y recibir mensajes de correo electrónico entre empleados, departamentos y clientes.

**Servidores de correo electrónico:** Los servidores de correo electrónico utilizan SMTP para comunicarse entre sí y enviar mensajes de correo electrónico a través de Internet. SMTP permite el enrutamiento y la entrega eficiente de los correos electrónicos a los servidores de destino.

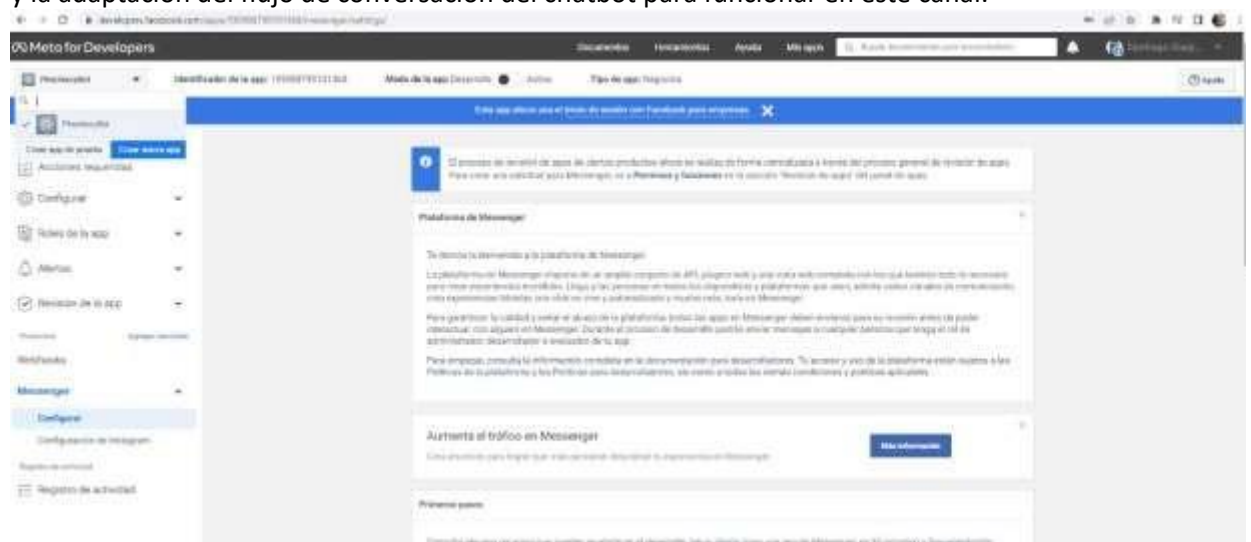
**Formularios web:** En aplicaciones web, SMTP se utiliza para enviar correos electrónicos a partir de la información ingresada por los usuarios en formularios web. Por ejemplo, cuando un usuario completa un formulario de contacto en un sitio web y hace clic en "Enviar", el correo electrónico se envía utilizando SMTP.

**Automatización de tareas:** SMTP también se puede utilizar para automatizar tareas relacionadas con el correo electrónico, como el envío programado de mensajes, seguimiento de correos electrónicos, respuestas automáticas, entre otros.

En este caso se utilizara SMTP para enviar correos electrónicos para recuperar contraseñas.

### Messenger como una alternativa de información.

Por último se va a explorar y configurar una herramienta alternativa, como Messenger, para brindar información a los usuarios. Esto puede implicar la integración de una API de Messenger y la adaptación del flujo de conversación del chatbot para funcionar en este canal.





Messenger es una popular herramienta de mensajería instantánea desarrollada por Facebook. Aunque su función principal es permitir la comunicación entre usuarios, también puede ser utilizada como una herramienta de información en varios aspectos:

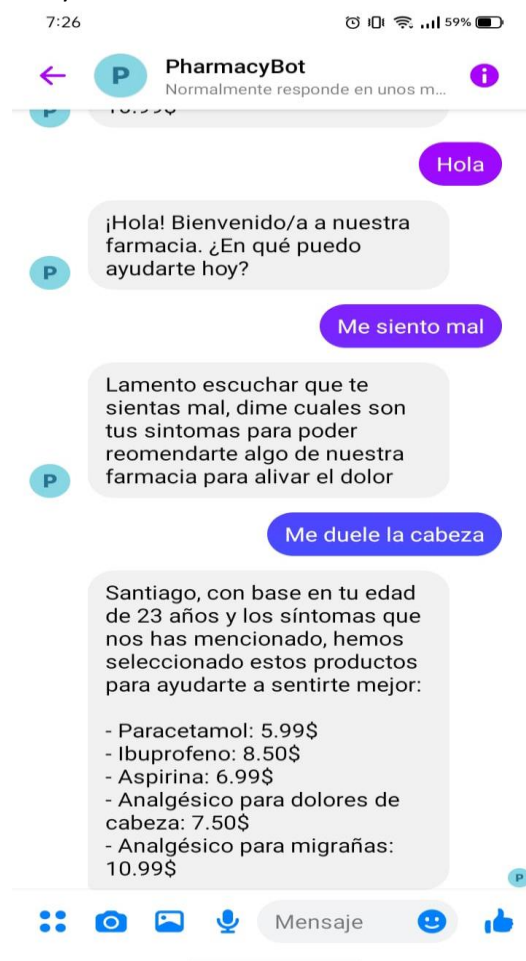
**Soporte al cliente:** Las empresas y organizaciones pueden utilizar Messenger como una plataforma de atención al cliente. Los usuarios pueden enviar mensajes a través de Messenger para realizar consultas, hacer preguntas o recibir asistencia. Las empresas pueden responder a través de Messenger y brindar información relevante, resolver problemas o proporcionar actualizaciones.

**Notificaciones y alertas:** Messenger puede ser utilizado para enviar notificaciones y alertas a los usuarios. Esto puede incluir recordatorios de eventos, actualizaciones de productos o servicios, noticias relevantes o cualquier tipo de información importante que deba ser comunicada rápidamente.

**Difusión de contenido:** Las empresas y organizaciones pueden utilizar Messenger para compartir contenido informativo con los usuarios. Esto puede incluir artículos, noticias, blogs, videos, infografías u otro tipo de contenido relevante para los usuarios.

**Bot de chat automatizado:** Messenger permite la implementación de bots de chat, que son programas automatizados capaces de responder a consultas y proporcionar información sin la intervención humana. Los bots de chat pueden ser utilizados para responder preguntas frecuentes, proporcionar recomendaciones, ayudar en la navegación de un sitio web, entre otras funciones.

**Grupos y comunidades:** Messenger ofrece la posibilidad de crear grupos y comunidades, lo que facilita la interacción entre usuarios interesados en temas específicos. Estos grupos pueden ser utilizados para compartir información, discutir temas relevantes y mantener a los usuarios informados sobre novedades y actualizaciones.





Messenger se puede utilizar como una plataforma de soporte al cliente efectiva y conveniente. Aquí hay algunas formas en las que Messenger puede ser utilizado para brindar soporte al cliente:

**Mensajes directos:** Los clientes pueden enviar mensajes directos a través de Messenger para realizar consultas, hacer preguntas o solicitar asistencia. Esto proporciona una forma rápida y conveniente para que los clientes se pongan en contacto con el equipo de soporte.

**Respuestas rápidas y automatizadas:** Mediante el uso de bots de chat y respuestas automáticas, es posible proporcionar respuestas rápidas y predefinidas a consultas comunes. Estos bots pueden ofrecer información básica, brindar instrucciones o dirigir a los clientes hacia recursos útiles. Esto ayuda a acelerar la respuesta y a resolver problemas frecuentes de manera eficiente.

**Adjuntar archivos y multimedia:** Los clientes pueden adjuntar archivos, capturas de pantalla o multimedia relacionada con su consulta a través de Messenger. Esto facilita la comunicación y permite que los agentes de soporte visualicen mejor el problema o la pregunta del cliente.

**Seguimiento de conversaciones:** Messenger permite que tanto los clientes como los agentes de soporte tengan un historial completo de las conversaciones anteriores. Esto es útil para realizar un seguimiento de las interacciones anteriores, recordar detalles importantes y brindar un servicio más personalizado.

**Notificaciones y actualizaciones:** Messenger se puede utilizar para enviar notificaciones y actualizaciones importantes a los clientes. Esto puede incluir información sobre cambios en el servicio, actualizaciones de productos, recordatorios de citas o cualquier otra información relevante para el cliente.

**Calificaciones y comentarios:** Messenger también permite a los clientes calificar y dejar comentarios sobre la calidad del servicio recibido. Esto proporciona retroalimentación valiosa para mejorar el soporte al cliente y mantener un alto nivel de satisfacción del cliente.

## Pruebas del Sistema

### Login

```
package main
import (
    "net/http"
    "net/http/httptest"
    "net/url"
    "strings"
    "testing"
)
func TestLoginHandler_ValidCredentials(t *testing.T) {
    // Crea una solicitud HTTP POST con las credenciales válidas
    form := url.Values{}
    form.Add("email", "santiguaylla@gmail.com")
    form.Add("password", "12345678")

    // Crea una solicitud HTTP con el formulario
    req, err := http.NewRequest("POST", "http://localhost:3000/clientes/login", strings.NewReader(form.Encode()))
    if err != nil {
        t.Fatalf("Error al crear la solicitud HTTP: %s", err)
    }

    // Establece las cabeceras necesarias para la solicitud POST
    req.Header.Set("Content-Type", "application/x-www-form-urlencoded")

    // Crea un ResponseRecorder para grabar la respuesta
    rr := httptest.NewRecorder()

    // Llamar a la función loginHandler con la solicitud y el ResponseRecorder
```

```

loginHandler(rr, req)

// Verificar el código de estado de la respuesta (debe ser 303 See Other para redireccionar)
if rr.Code != http.StatusSeeOther {
    t.Errorf("El código de estado no es el esperado. Esperado: %d, Obtenido: %d", http.StatusSeeOther, rr.Code)
}

// Verificar que la redirección sea a la URL esperada
expectedURL := "/chatbot/pharmacybot/conversacion/?id=59"
if location := rr.Header().Get("Location"); location != expectedURL {
    t.Errorf("URL de redirección incorrecta. Esperada: %s, Obtenido: %s", expectedURL, location)
}
}

func TestLoginHandler_InvalidCredentials(t *testing.T) {
    // Preparar una solicitud HTTP falsa con credenciales inválidas
    form := url.Values{}
    form.Add("email", "santiguaylla@gmail.com")
    form.Add("password", "123456789")
    req, err := http.NewRequest("POST", "/login", strings.NewReader(form.Encode()))
    if err != nil {
        t.Fatalf("Error al crear la solicitud HTTP: %v", err)
    }
    req.Header.Set("Content-Type", "application/x-www-form-urlencoded")

    // Crear un ResponseRecorder para grabar la respuesta
    rr := httptest.NewRecorder()

    // Llamar a la función loginHandler con la solicitud falsa y el ResponseRecorder
    loginHandler(rr, req)

    // Verificar el código de estado de la respuesta (debe ser 302 Found para redireccionar)
    if rr.Code != http.StatusFound {
        t.Errorf("El código de estado no es el esperado. Esperado: %d, Obtenido: %d", http.StatusFound, rr.Code)
    }

    // Verificar la URL de redirección
    expectedURL := "/clientes/login"
    if location := rr.Header().Get("Location"); location != expectedURL {
        t.Errorf("URL de redirección incorrecta. Esperada: %s, Obtenido: %s", expectedURL, location)
    }
}

// Verificar el mensaje de error en el cuerpo de la respuesta
expectedErrorMessage := "El usuario no se encuentra en los registros."
if !strings.Contains(rr.Body.String(), expectedErrorMessage) {
    t.Errorf("Mensaje de error esperado no encontrado en la respuesta. Esperado: %s", expectedErrorMessage)
}
}

```

CASO DE PRUEBA	DESCRIPCION	DATOS DE PRUEBA	RESULTADO ESPERADO
Válido	Verifica si se puede iniciar sesión correctamente con un email y contraseña válidos.	Email: santiguaylla@gmail.com  Password: 12345678	=== RUN TestLoginHandler_ValidCredentials/ValidCredentials  --- PASS: TestLoginHandler_ValidCredentials/ValidCredentials

Email o contraseña inválido	Comprueba si se maneja correctamente un email o contraseña incorrecto al intentar iniciar sesión.	Email: santiguayllaa@gmail.com  Password: 123456789	=== RUN TestLoginHandler_ValidCredentials/InvalidCredentials  2023/07/23 23:00:36 C:/Users/Hello/Desktop/Project_Chatbot - PostgreSQL2/CAPA_DE_DATOS/database/repositories/cliente_repository.go:106 record not found  --- PASS: TestLoginHandler_ValidCredentials/InvalidCredentials
Formatos inválidos	Comprueba que el formato de los datos sea válido, email con formato de email, y la contraseña que sea mayor o igual a 8	Email: santiguayllagmail.com  Password: 12345	=== RUN TestLoginHandler_ValidCredentials/InvalidEmailFormat  prueba_test.go:61: El código de estado no es el esperado. Esperado: 400, Obtenido: 302  prueba_test.go:67: Mensaje de error esperado no encontrado en la respuesta. Esperado: El formato del correo electrónico no es válido.  === RUN TestLoginHandler_ValidCredentials/InvalidPasswordFormat  prueba_test.go:88: El código de estado no es el esperado. Esperado: 400, Obtenido: 302  prueba_test.go:94: Mensaje de error esperado no encontrado en la respuesta. Esperado: La contraseña debe tener al menos 8 caracteres.  --- FAIL: TestLoginHandler_ValidCredentials/InvalidEmailFormat -  -- FAIL: TestLoginHandler_ValidCredentials/InvalidPasswordFormat

## Registro

package main

```
import (
    "net/http"
    "net/http/httptest"
    "net/url"
    "strings"
    "testing"
)
```

```

func TestFormularioHandler(t *testing.T) {
    t.Run("ValidFormData", func(t *testing.T) {
        // Crear una solicitud HTTP POST con datos válidos del formulario
        // Puedes modificar los valores aquí según tus necesidades de prueba
        form := url.Values{}
        form.Add("cedula", "1724008071")
        form.Add("nombres", "Santiago")
        form.Add("apellidos", "Melena")
        form.Add("direccion", "Riobamba")
        form.Add("celular", "0987654321")
        form.Add("email", "usuario@example.com")
        form.Add("password", "contraseña123")
        form.Add("fechaNac", "2000-01-01")
        form.Add("sexo", "M")

        req, err := http.NewRequest("POST", "http://localhost:3000/clientes/registro",
strings.NewReader(form.Encode()))
        if err != nil {
            t.Fatalf("Error al crear la solicitud HTTP: %s", err)
        }
        req.Header.Set("Content-Type", "application/x-www-form-urlencoded")

        // Crear un ResponseRecorder para grabar la respuesta
        rr := httptest.NewRecorder()

        // Llamar a la función formularioHandler con la solicitud y el ResponseRecorder
        formularioHandler(rr, req)

        // Verificar el código de estado de la respuesta (debe ser 303 See Other para redireccionar)
        if rr.Code != http.StatusSeeOther {
            t.Errorf("El código de estado no es el esperado. Esperado: %d, Obtenido: %d", http.StatusSeeOther, rr.Code)
        }

        // Verificar que la redirección sea a la URL esperada
        expectedURL := "/clientes/login"
        if location := rr.Header().Get("Location"); location != expectedURL {
            t.Errorf("URL de redirección incorrecta. Esperada: %s, Obtenido: %s", expectedURL, location)
        }
    })

    t.Run("InvalidEmailFormat", func(t *testing.T) {
        // Crear una solicitud HTTP POST con un correo electrónico inválido (sin @)
        form := url.Values{}
        form.Add("cedula", "1234567890")
        form.Add("nombres", "Nombre")
        form.Add("apellidos", "Apellido")
        form.Add("direccion", "Dirección")
        form.Add("celular", "0987654321")
        form.Add("email", "usuariocorreo.com") // Correo electrónico inválido
        form.Add("password", "contraseña123")
        form.Add("fechaNac", "2000-01-01")
        form.Add("sexo", "M")

        req, err := http.NewRequest("POST", "http://localhost:3000/clientes/registro",
strings.NewReader(form.Encode()))
        if err != nil {
            t.Fatalf("Error al crear la solicitud HTTP: %s", err)
        }
        req.Header.Set("Content-Type", "application/x-www-form-urlencoded")

        // Crear un ResponseRecorder para grabar la respuesta
        rr := httptest.NewRecorder()

```

```

// Llamar a la función formularioHandler con la solicitud y el ResponseRecorder
formularioHandler(rr, req)

// Verificar el código de estado de la respuesta (debe ser 303 See Other para redireccionar)
if rr.Code != http.StatusSeeOther {
    t.Errorf("El código de estado no es el esperado. Esperado: %d, Obtenido: %d", http.StatusSeeOther, rr.Code)
}

// Verificar que la redirección sea a la URL esperada
expectedURL := "/clientes/registro?error=cedula"
if location := rr.Header().Get("Location"); location != expectedURL {
    t.Errorf("URL de redirección incorrecta. Esperada: %s, Obtenido: %s", expectedURL, location)
}

// Verificar el mensaje de error en el cuerpo de la respuesta
expectedErrorMessage := "El formato del correo electrónico no es válido."
if !strings.Contains(rr.Body.String(), expectedErrorMessage) {
    t.Errorf("Mensaje de error esperado no encontrado en la respuesta. Esperado: %s", expectedErrorMessage)
}
})

t.Run("InvalidCedulaFormat", func(t *testing.T) {
    // Crear una solicitud HTTP POST con una cédula inválida (menos de 10 dígitos)
    form := url.Values{}
    form.Add("cedula", "123456789") // Cédula inválida (menos de 10 dígitos)
    form.Add("nombres", "Nombre")
    form.Add("apellidos", "Apellido")
    form.Add("direccion", "Dirección")
    form.Add("celular", "0987654321")
    form.Add("email", "usuario@example.com")
    form.Add("password", "contraseña123")
    form.Add("fechaNac", "2000-01-01")
    form.Add("sexo", "M")

    req, err := http.NewRequest("POST", "http://localhost:3000/clientes/registro",
strings.NewReader(form.Encode()))
    if err != nil {
        t.Fatalf("Error al crear la solicitud HTTP: %s", err)
    }
    req.Header.Set("Content-Type", "application/x-www-form-urlencoded")

    rr := httptest.NewRecorder()
    formularioHandler(rr, req)

    // Verificar el código de estado de la respuesta (debe ser 303 See Other para redireccionar)
    if rr.Code != http.StatusSeeOther {
        t.Errorf("El código de estado no es el esperado. Esperado: %d, Obtenido: %d", http.StatusSeeOther, rr.Code)
    }

    // Verificar que la redirección sea a la URL esperada
    expectedURL := "/clientes/registro?error=cedula"
    if location := rr.Header().Get("Location"); location != expectedURL {
        t.Errorf("URL de redirección incorrecta. Esperada: %s, Obtenido: %s", expectedURL, location)
    }

    // Verificar el mensaje de error en el cuerpo de la respuesta
    expectedErrorMessage := "La cédula no es válida"
    if !strings.Contains(rr.Body.String(), expectedErrorMessage) {
        t.Errorf("Mensaje de error esperado no encontrado en la respuesta. Esperado: %s", expectedErrorMessage)
    }
})

t.Run("InvalidNombreFormat", func(t *testing.T) {

```

```

// Crear una solicitud HTTP POST con un nombre inválido (sin caracteres alfabéticos)
form := url.Values{}
form.Add("cedula", "1234567890")
form.Add("nombres", "123456") // Nombre inválido (sin caracteres alfabéticos)
form.Add("apellidos", "Apellido")
form.Add("direccion", "Dirección")
form.Add("celular", "0987654321")
form.Add("email", "usuario@example.com")
form.Add("password", "contraseña123")
form.Add("fechaNac", "2000-01-01")
form.Add("sexo", "M")

req, err := http.NewRequest("POST", "http://localhost:3000/clientes/registro",
strings.NewReader(form.Encode()))
if err != nil {
    t.Fatalf("Error al crear la solicitud HTTP: %s", err)
}
req.Header.Set("Content-Type", "application/x-www-form-urlencoded")

rr := httptest.NewRecorder()
formularioHandler(rr, req)

// Verificar el código de estado de la respuesta (debe ser 303 See Other para redireccionar)
if rr.Code != http.StatusSeeOther {
    t.Errorf("El código de estado no es el esperado. Esperado: %d, Obtenido: %d", http.StatusSeeOther, rr.Code)
}

// Verificar que la redirección sea a la URL esperada
expectedURL := "/clientes/registro?error=cedula"
if location := rr.Header().Get("Location"); location != expectedURL {
    t.Errorf("URL de redirección incorrecta. Esperada: %s, Obtenido: %s", expectedURL, location)
}

// Verificar el mensaje de error en el cuerpo de la respuesta
expectedErrorMessage := "El nombre no es válido"
if !strings.Contains(rr.Body.String(), expectedErrorMessage) {
    t.Errorf("Mensaje de error esperado no encontrado en la respuesta. Esperado: %s", expectedErrorMessage)
}
})
t.Run("InvalidApellidoFormat", func(t *testing.T) {
    // Crear una solicitud HTTP POST con un apellido inválido (sin caracteres alfabéticos)
    form := url.Values{}
    form.Add("cedula", "1234567890")
    form.Add("nombres", "Nombre")
    form.Add("apellidos", "123456") // Apellido inválido (sin caracteres alfabéticos)
    form.Add("direccion", "Dirección")
    form.Add("celular", "0987654321")
    form.Add("email", "usuario@example.com")
    form.Add("password", "contraseña123")
    form.Add("fechaNac", "2000-01-01")
    form.Add("sexo", "M")

    req, err := http.NewRequest("POST", "http://localhost:3000/clientes/registro",
strings.NewReader(form.Encode()))
    if err != nil {
        t.Fatalf("Error al crear la solicitud HTTP: %s", err)
    }
    req.Header.Set("Content-Type", "application/x-www-form-urlencoded")

    rr := httptest.NewRecorder()
    formularioHandler(rr, req)

    // Verificar el código de estado de la respuesta (debe ser 303 See Other para redireccionar)

```

```

if rr.Code != http.StatusSeeOther {
    t.Errorf("El código de estado no es el esperado. Esperado: %d, Obtenido: %d", http.StatusSeeOther, rr.Code)
}

// Verificar que la redirección sea a la URL esperada
expectedURL := "/clientes/registro?error=cedula"
if location := rr.Header().Get("Location"); location != expectedURL {
    t.Errorf("URL de redirección incorrecta. Esperada: %s, Obtenido: %s", expectedURL, location)
}

// Verificar el mensaje de error en el cuerpo de la respuesta
expectedErrorMessage := "El apellido no es válido"
if !strings.Contains(rr.Body.String(), expectedErrorMessage) {
    t.Errorf("Mensaje de error esperado no encontrado en la respuesta. Esperado: %s", expectedErrorMessage)
}
})

t.Run("InvalidDireccionFormat", func(t *testing.T) {
    // Crear una solicitud HTTP POST con una dirección inválida (sin caracteres alfabéticos)
    form := url.Values{}
    form.Add("cedula", "1234567890")
    form.Add("nombres", "Nombre")
    form.Add("apellidos", "Apellido")
    form.Add("direccion", "") // Dirección inválida (vacío)
    form.Add("celular", "0987654321")
    form.Add("email", "usuario@example.com")
    form.Add("password", "contraseña123")
    form.Add("fechaNac", "2000-01-01")
    form.Add("sexo", "M")

    req, err := http.NewRequest("POST", "http://localhost:3000/clientes/registro",
strings.NewReader(form.Encode()))
    if err != nil {
        t.Fatalf("Error al crear la solicitud HTTP: %s", err)
    }
    req.Header.Set("Content-Type", "application/x-www-form-urlencoded")

    rr := httptest.NewRecorder()
    formularioHandler(rr, req)

    // Verificar el código de estado de la respuesta (debe ser 303 See Other para redireccionar)
    if rr.Code != http.StatusSeeOther {
        t.Errorf("El código de estado no es el esperado. Esperado: %d, Obtenido: %d", http.StatusSeeOther, rr.Code)
    }

    // Verificar que la redirección sea a la URL esperada
    expectedURL := "/clientes/registro?error=cedula"
    if location := rr.Header().Get("Location"); location != expectedURL {
        t.Errorf("URL de redirección incorrecta. Esperada: %s, Obtenido: %s", expectedURL, location)
    }

    // Verificar el mensaje de error en el cuerpo de la respuesta
    expectedErrorMessage := "La dirección no es válida porque está vacía"
    if !strings.Contains(rr.Body.String(), expectedErrorMessage) {
        t.Errorf("Mensaje de error esperado no encontrado en la respuesta. Esperado: %s", expectedErrorMessage)
    }
})

t.Run("InvalidCelularFormat", func(t *testing.T) {
    // Crear una solicitud HTTP POST con un número de celular inválido (menos de 10 dígitos)
    form := url.Values{}
    form.Add("cedula", "1234567890")
    form.Add("nombres", "Nombre")
    form.Add("apellidos", "Apellido")

```

```

form.Add("direccion", "Dirección")
form.Add("celular", "1234567") // Número de celular inválido (menos de 10 dígitos)
form.Add("email", "usuario@example.com")
form.Add("password", "contraseña123")
form.Add("fechaNac", "2000-01-01")
form.Add("sexo", "M")

req, err := http.NewRequest("POST", "http://localhost:3000/clientes/registro",
strings.NewReader(form.Encode()))
if err != nil {
    t.Fatalf("Error al crear la solicitud HTTP: %s", err)
}
req.Header.Set("Content-Type", "application/x-www-form-urlencoded")

rr := httptest.NewRecorder()
formularioHandler(rr, req)

// Verificar el código de estado de la respuesta (debe ser 303 See Other para redireccionar)
if rr.Code != http.StatusSeeOther {
    t.Errorf("El código de estado no es el esperado. Esperado: %d, Obtenido: %d", http.StatusSeeOther, rr.Code)
}

// Verificar que la redirección sea a la URL esperada
expectedURL := "/clientes/registro?error=cedula"
if location := rr.Header().Get("Location"); location != expectedURL {
    t.Errorf("URL de redirección incorrecta. Esperada: %s, Obtenido: %s", expectedURL, location)
}

// Verificar el mensaje de error en el cuerpo de la respuesta
expectedErrorMessage := "El número de celular no es válido"
if !strings.Contains(rr.Body.String(), expectedErrorMessage) {
    t.Errorf("Mensaje de error esperado no encontrado en la respuesta. Esperado: %s", expectedErrorMessage)
}
})

t.Run("InvalidFechaNacFormat", func(t *testing.T) {
    // Crear una solicitud HTTP POST con una fecha de nacimiento inválida (formato incorrecto)
    form := url.Values{}
    form.Add("cedula", "1234567890")
    form.Add("nombres", "Nombre")
    form.Add("apellidos", "Apellido")
    form.Add("direccion", "Dirección")
    form.Add("celular", "0987654321")
    form.Add("email", "usuario@example.com")
    form.Add("password", "contraseña123")
    form.Add("fechaNac", "2022-01-01") // Fecha de nacimiento inválida (menor de 18 años)
    form.Add("sexo", "M")

    req, err := http.NewRequest("POST", "http://localhost:3000/clientes/registro",
strings.NewReader(form.Encode()))
    if err != nil {
        t.Fatalf("Error al crear la solicitud HTTP: %s", err)
    }
    req.Header.Set("Content-Type", "application/x-www-form-urlencoded")

    rr := httptest.NewRecorder()
    formularioHandler(rr, req)

    // Verificar el código de estado de la respuesta (debe ser 303 See Other para redireccionar)
    if rr.Code != http.StatusSeeOther {
        t.Errorf("El código de estado no es el esperado. Esperado: %d, Obtenido: %d", http.StatusSeeOther, rr.Code)
    }
    // Verificar que la redirección sea a la URL esperada

```



```

expectedURL := "/clientes/registro?error=cedula"
if location := rr.Header().Get("Location"); location != expectedURL {
    t.Errorf("URL de redirección incorrecta. Esperada: %s, Obtener: %s", expectedURL, location)
}

// Verificar el mensaje de error en el cuerpo de la respuesta
expectedErrorMessage := "La fecha de nacimiento no es válida, el cliente debe ser mayor de edad"
if !strings.Contains(rr.Body.String(), expectedErrorMessage) {
    t.Errorf("Mensaje de error esperado no encontrado en la respuesta. Esperado: %s", expectedErrorMessage)
}
})
t.Run("InvalidPasswordFormat", func(t *testing.T) {
    // Crear una solicitud HTTP POST con una contraseña inválida (menos de 8 caracteres)
    form := url.Values{}
    form.Add("cedula", "1234567890")
    form.Add("nombres", "Nombre")
    form.Add("apellidos", "Apellido")
    form.Add("direccion", "Dirección")
    form.Add("celular", "0987654321")
    form.Add("email", "usuario@example.com")
    form.Add("password", "abc123") // Contraseña inválida (menos de 8 caracteres)
    form.Add("fechaNac", "2000-01-01")
    form.Add("sexo", "M")

    req, err := http.NewRequest("POST", "http://localhost:3000/clientes/registro",
strings.NewReader(form.Encode()))
    if err != nil {
        t.Fatalf("Error al crear la solicitud HTTP: %s", err)
    }
    req.Header.Set("Content-Type", "application/x-www-form-urlencoded")

    rr := httptest.NewRecorder()
    formularioHandler(rr, req)

    // Verificar el código de estado de la respuesta (debe ser 303 See Other para redireccionar)
    if rr.Code != http.StatusSeeOther {
        t.Errorf("El código de estado no es el esperado. Esperado: %d, Obtenido: %d", http.StatusSeeOther, rr.Code)
    }

    // Verificar que la redirección sea a la URL esperada
    expectedURL := "/clientes/registro?error=cedula"
    if location := rr.Header().Get("Location"); location != expectedURL {
        t.Errorf("URL de redirección incorrecta. Esperada: %s, Obtener: %s", expectedURL, location)
    }

    // Verificar el mensaje de error en el cuerpo de la respuesta
    expectedErrorMessage := "La contraseña debe tener al menos 8 caracteres"
    if !strings.Contains(rr.Body.String(), expectedErrorMessage) {
        t.Errorf("Mensaje de error esperado no encontrado en la respuesta. Esperado: %s", expectedErrorMessage)
    }
})
t.Run("InvalidSexFormat", func(t *testing.T) {
    // Crear una solicitud HTTP POST con un valor inválido para el campo "sexo"
    form := url.Values{}
    form.Add("cedula", "1234567890")
    form.Add("nombres", "Nombre")
    form.Add("apellidos", "Apellido")
    form.Add("direccion", "Dirección")
    form.Add("celular", "0987654321")
    form.Add("email", "usuario@example.com")
    form.Add("password", "contraseña123")
    form.Add("fechaNac", "2000-01-01")
    form.Add("sexo", "Otro") // Valor inválido para el campo "sexo"

```

```

req, err := http.NewRequest("POST", "http://localhost:3000/clientes/registro",
strings.NewReader(form.Encode()))
if err != nil {
    t.Fatalf("Error al crear la solicitud HTTP: %s", err)
}
req.Header.Set("Content-Type", "application/x-www-form-urlencoded")

rr := httptest.NewRecorder()
formularioHandler(rr, req)

// Verificar el código de estado de la respuesta (debe ser 303 See Other para redireccionar)
if rr.Code != http.StatusSeeOther {
    t.Errorf("El código de estado no es el esperado. Esperado: %d, Obtenido: %d", http.StatusSeeOther, rr.Code)
}

// Verificar que la redirección sea a la URL esperada
expectedURL := "/clientes/registro?error=cedula"
if location := rr.Header().Get("Location"); location != expectedURL {
    t.Errorf("URL de redirección incorrecta. Esperada: %s, Obtenido: %s", expectedURL, location)
}

// Verificar el mensaje de error en el cuerpo de la respuesta
expectedErrorMessage := "El valor del campo sexo es inválido"
if !strings.Contains(rr.Body.String(), expectedErrorMessage) {
    t.Errorf("Mensaje de error esperado no encontrado en la respuesta. Esperado: %s", expectedErrorMessage)
}
}
})
}

```

CASO DE PRUEBA	DESCRIPCION	DATOS DE PRUEBA	RESULTADO ESPERADO
Válido	Verifica si se puede registrar correctamente con todos los datos válidos.	<pre> form.Add("cedula", "1724008071")          form.Add("nombres", "Santiago")          form.Add("apellidos", "Melena")          form.Add("direccion", "Riobamba")          form.Add("celular", "0987654999")          form.Add("email", "santiago@example.com")          form.Add("password", "contraseña123")          form.Add("fechaNac", "2000-01-01") </pre>	--- PASS: TestFormularioHandler/ValidFormData

		form.Add("sexo", "F")	
Datos inválidos	Comprueba si se maneja correctamente los datos del usuario al intentar registrarse.	form.Add("cedula", "172400807123")  form.Add("nombres", "Satniagsad213")  form.Add("apellidos", "Melena213123")  form.Add("direccion", "")  form.Add("celular", "213210987654999")  form.Add("email", "andreaexample.com")  form.Add("password", "cña123")  form.Add("fechaNac", "2024-01-01")  form.Add("sexo", "L")	--- FAIL: TestFormularioHandler/InvalidEmailFormat --- FAIL: TestFormularioHandler/InvalidCedulaFormat --- FAIL: TestFormularioHandler/InvalidApellidoFormat --- FAIL: TestFormularioHandler/InvalidDireccionFormat --- FAIL: TestFormularioHandler/InvalidCelularFormat --- FAIL: TestFormularioHandler/InvalidFechaNacFormat --- FAIL: TestFormularioHandler/InvalidPasswordFormat --- FAIL: TestFormularioHandler/InvalidSexFormat

Manual de Usuario – PharmacyBot

Bienvenido al Manual de Usuario de PharmacyBot. Este documento proporciona una guía detallada sobre cómo utilizar nuestra plataforma para aprovechar al máximo sus características y funcionalidades.

Descripción de la Aplicación.

PharmacyBot es una plataforma interactiva diseñada para brindar atención personalizada a los clientes de una franquicia farmacéutica. Las funciones principales del chatbot son, solventar dudas de los usuarios, el chatbot responderá a las preguntas frecuentes de los clientes sobre la franquicia, los productos, la disponibilidad de productos, entre otros. Además, se podrá consultar con el inventario de la farmacia y ofrecer los productos al cliente: El chatbot consultará la base de datos de la farmacia para conocer la disponibilidad de productos y ofrecerlos al cliente. Además, brindará información sobre los productos, su uso y los posibles efectos secundarios. Cuenta con una interfaz amigable e intuitiva, nuestra aplicación le permite acceder a una variedad de servicios y recursos, y brinda una experiencia personalizada para cada usuario.

Contenido del Manual.

Este manual está diseñado para ayudarte a comprender y utilizar eficazmente PharmacyBot. Incluye instrucciones detalladas para las siguientes funcionalidades principales:



Bienvenido al ChatBot

Accede a tu cuenta para continuar


Iniciar Sesión

Registrate

La página principal de PharmacyBot presenta una interfaz sencilla con dos botones principales: "Iniciar Sesión": Utiliza este botón si ya tienes una cuenta registrada para acceder a tu perfil y disfrutar de todas las características de la plataforma.

"Registrarse": Si eres nuevo en PharmacyBot, haz clic en este botón para crear una nueva cuenta y comenzar a utilizar nuestros servicios.

### Inicio de Sesión (Login).



Bienvenido De Nuevo

Dirección email

Contraseña


Continue

¿No tiene una cuenta? [Regístrate](#)

---

O

---

 Continue con Google

El proceso de inicio de sesión en PharmacyBot es rápido y sencillo. Sigue los pasos a continuación para acceder a tu cuenta o para crear una nueva cuenta utilizando una cuenta de Google.

#### ***Pasos para Iniciar Sesión:***

Ingresa tus Credenciales, en la página de inicio, encontrarás dos campos de entrada:

- Campo de Email: Introduce la dirección de correo electrónico asociada a tu cuenta.
- Campo de Contraseña: Escribe tu contraseña secreta para la cuenta.

Asegúrate de ingresar la información correctamente para evitar errores en el inicio de sesión.

#### ***Continuar con el Inicio de Sesión:***

Después de ingresar tu email y contraseña, haz clic en el botón "Continuar" para iniciar sesión en tu cuenta.

Si los datos proporcionados son correctos, serás redirigido al chatbot personalizado.

#### ***¿No tienes una cuenta? Regístrate:***

Si eres nuevo en PharmacyBot, puedes hacer clic en el enlace "Regístrate" para crear una nueva cuenta.

Serás dirigido a la página de registro donde podrás ingresar tus datos personales y crear una nueva cuenta de usuario.

#### ***Iniciar Sesión con una Cuenta de Google:***


Si prefieres utilizar tu cuenta de Google para acceder a PharmacyBot, simplemente haz clic en el botón "Continuar con Google".

Serás redirigido a la página de inicio de sesión de Google, donde podrás ingresar tus

credenciales de Google y permitir el acceso a tu cuenta.

Recuerda que la seguridad de tu cuenta es importante. Asegúrate de utilizar una contraseña segura y mantenerla confidencial. Si tienes problemas con el inicio de sesión o necesitas asistencia adicional, no dudes en contactar a nuestro equipo de soporte.

### Registro de cuenta.



### Crea una cuenta

**Cédula:**

**Nombres:**

**Apellidos:**

**Dirección:**

**Célular:**

**Email:**

**Contraseña:**


**Fecha de nacimiento:**

**Sexo:**

[Registrar](#)

Ya tienes una cuenta? [Inicia Sesión](#)

☐

 Continúe con Google

Para empezar a disfrutar de todas las funcionalidades de PharmacyBot, necesitarás crear una cuenta personal. A continuación, se detallan los pasos para registrar una nueva cuenta y también la opción de utilizar una cuenta de Google para el registro.

#### ***Pasos para Registrar una Cuenta:***

Ingresa tus Datos Personales en la página de registro, encontrarás los siguientes campos para ingresar tus datos:

Campo de Cédula: Ingresa tu número de cédula o documento de identificación oficial.

Campo de Nombres: Escribe tu nombre completo.

Campo de Apellidos: Escribe tus apellidos completos.

Campo de Dirección: Introduce tu dirección física actual.

Campo de Celular: Ingresa tu número de teléfono celular.

Campo de Email: Proporciona una dirección de correo electrónico válida que utilizarás para iniciar sesión en la plataforma.

Campo de Contraseña: Crea una contraseña segura para proteger tu cuenta. Asegúrate de utilizar una combinación de caracteres, números y símbolos para aumentar la seguridad.

Campo de Fecha de Nacimiento: Ingresa tu fecha de nacimiento en el formato [dd/mm/aaaa].

Campo de Sexo: Selecciona tu género (Masculino/Femenino) de la lista desplegable.

### ***Finaliza el Registro:***

Revisa cuidadosamente todos los datos ingresados para asegurarte de que sean precisos.

Una vez que hayas completado todos los campos, haz clic en el botón "Registrar" para crear tu cuenta.

### ***Iniciar Sesión con una Cuenta Existente:***

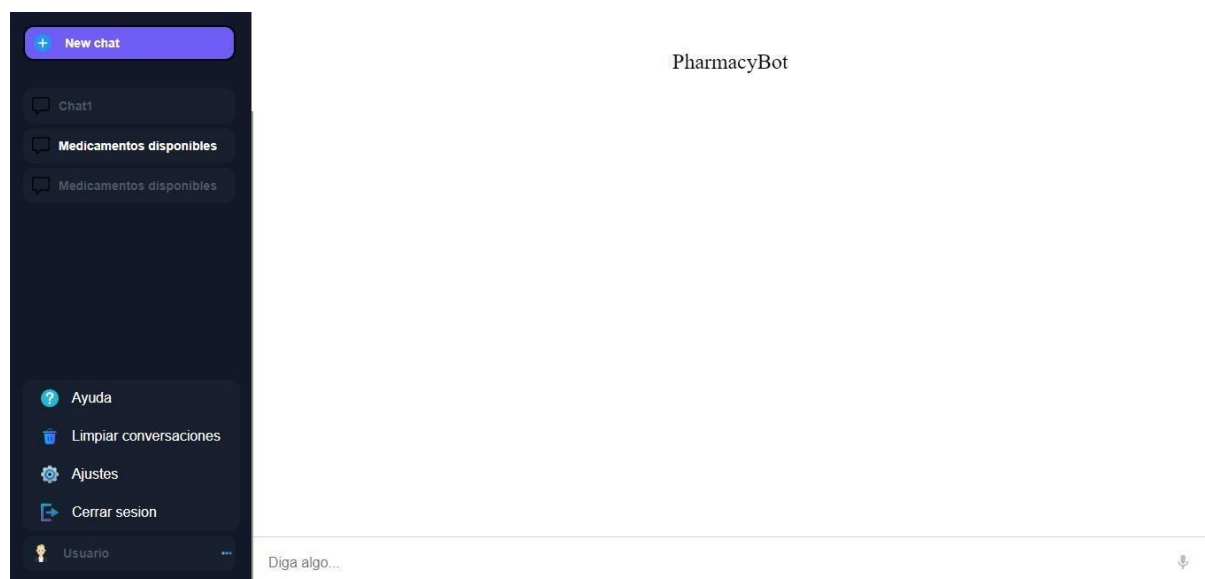
Si ya tienes una cuenta registrada, puedes acceder a ella fácilmente. Haz clic en el enlace "Iniciar Sesión" para ser redirigido a la página de inicio de sesión, donde deberás ingresar tu email y contraseña para acceder a tu perfil personalizado.

### ***Registrar con una Cuenta de Google:***

Si prefieres utilizar tu cuenta de Google para registrarte en PharmacyBot, simplemente haz clic en el botón "Registrar con Google". Serás redirigido a la página de inicio de sesión de Google, donde podrás ingresar tus credenciales de Google y permitir el acceso a tu cuenta.

Recuerda que proteger tu información personal es primordial para nosotros. Los datos proporcionados se utilizarán de acuerdo con nuestra política de privacidad y no serán compartidos con terceros sin tu consentimiento. Si tienes alguna duda o encuentras dificultades durante el proceso de registro, no dudes en contactar a nuestro equipo de soporte.

## **ChatBot.**



¡Bienvenido al Chatbot de PharmacyBot! Esta herramienta te permitirá interactuar con el sistema y obtener respuestas a tus preguntas de manera rápida y sencilla. A continuación, te explicamos cómo utilizar las diferentes funciones de nuestro Chatbot:

**Crear un Nuevo Chat:**

En la parte izquierda de la pantalla, encontrarás un botón con la etiqueta "Nuevo Chat". Haz clic en este botón para comenzar una nueva conversación con el Chatbot.

**Visualizar Conversaciones Anteriores:**

Una vez iniciado un chat, las conversaciones previas se mostrarán en orden cronológico, con la conversación más reciente en la parte superior. Podrás revisar tus interacciones anteriores con el Chatbot y continuar con una conversación existente si así lo deseas.

**Iniciar una Nueva Conversación:**

Para empezar una nueva interacción, simplemente escribe tu mensaje en el campo de texto ubicado en la parte inferior de la pantalla derecha y luego haz clic en el botón "Enviar". Tu mensaje se enviará al Chatbot, y recibirás una respuesta en la misma área de la pantalla.

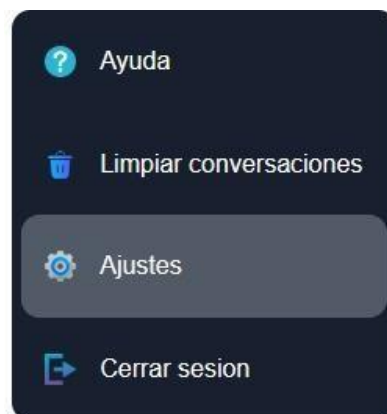
**Respuesta del Chatbot:**

El Chatbot proporcionará respuestas automáticas a tus mensajes. Las respuestas se mostrarán justo debajo de tus mensajes en la pantalla derecha. Puedes seguir enviando mensajes para continuar la conversación.

**Ingresar mediante Voz:**

Si prefieres utilizar la voz en lugar de escribir, haz clic en el icono del micrófono en la parte inferior del campo de texto. A continuación, puedes hablar y el Chatbot procesará tu voz para proporcionarte una respuesta.

**Nota:** El Chatbot está diseñado para proporcionar respuestas basadas en patrones y reglas predefinidas. Puede que algunas preguntas o solicitudes específicas estén fuera de su capacidad de respuesta, en cuyo caso te recomendamos contactar al equipo de soporte para obtener ayuda adicional. Recuerda que el Chatbot está aquí para ayudarte y resolver tus dudas de manera rápida y eficiente. Si tienes algún problema con el Chatbot o deseas obtener asistencia adicional, no dudes en ponerte en contacto con nuestro equipo de soporte.

**Navegación del Botón del Usuario:**

El botón de navegación ubicado junto a la foto y nombre del usuario proporciona acceso rápido a varias funciones importantes. Aquí tienes una descripción detallada de cada opción:

**Ayuda:**

Al hacer clic en este botón, serás redirigido a una página de "Preguntas Frecuentes" donde

encontrarás respuestas a las consultas más comunes y soluciones a problemas frecuentes. Esta sección te brindará orientación y asistencia para sacar el máximo provecho de PharmacyBot.

#### ***Limpiar Conversaciones:***

Si deseas eliminar todas las conversaciones anteriores y comenzar desde cero, puedes utilizar esta opción. Al hacer clic en "Limpiar Conversaciones", todas las interacciones anteriores con el Chatbot se borrarán, proporcionándote un lienzo limpio para una nueva conversación.

#### ***Ajustes:***

El botón de "Ajustes" te llevará a una página donde podrás gestionar ciertos aspectos de tu cuenta. Aquí encontrarás opciones como cambiar tu contraseña para mantener tu cuenta segura y protegida. También puede haber otras configuraciones y preferencias disponibles según las características de PharmacyBot.

#### ***Cerrar Sesión:***

Si deseas finalizar tu sesión en el Chatbot y regresar a la página principal del sistema, haz clic en "Cerrar Sesión". Al hacerlo, saldrás de tu cuenta actual y deberás iniciar sesión nuevamente si deseas interactuar con el Chatbot en el futuro.

Recuerda que estas opciones te brindan mayor control sobre tu experiencia PharmacyBot. Si tienes alguna pregunta específica o necesitas ayuda con alguna función en particular, no dudes en explorar la sección de "Ayuda" o contactar con nuestro equipo de soporte.

### **PRUEBAS DE USABILIDAD “PHARMACYBOT”.**

Para la medición de las subcaracterísticas de la usabilidad se utilizó el cuestionario USE, con las métricas basadas en la escala de Likert dando un máximo de 7 puntos, el valor neutral es 4 y todo valor mayor a 4 es positivo, los cuales se indican a continuación en la Tabla 1.

<b>Nivel de aceptación</b>	<b>Valor de aceptación</b>
Totalmente en desacuerdo	1
Muy en desacuerdo	2
En desacuerdo	3
Indeciso o indiferente	4
De acuerdo	5
Muy de acuerdo	6
Totalmente de acuerdo	7

*Tabla 1: Escala de Likert*

### **Evaluación de la facilidad de uso**

En cuanto, a la evaluación de la subcaracterísticas denominada facilidad de uso, la calificación promedio fue de 5.39 sobre los 7 puntos como máximo, siendo este un valor positivo dado que según la escala de Likert los valores posteriores a 4 son favorables esto quiere decir que el sistema PharmacyBot está dentro de los parámetros para ser un producto software de calidad en cuanto a la facilidad de uso. En este análisis la mejor calificación fue de 5.92/7 fue para la característica “Considera usted que el chatbot es simple de usar?”, mientras que las peores valuaciones fueron para las características: “¿Considera usted que el chatbot presenta



inconsistencias?.”, “La interfaz de usuario es adecuada” con un puntaje de 4,64 y 4,92 respectivamente. Dichos valores se muestran en la Tabla 2.

Facilidad de uso del sistema SAGAMO medido con el cuestionario USE.	
Preguntas	Media
¿Considera usted que el chatbot es simple de usar?	5,93
La interfaz de usuario es adecuada	4,92
¿Considera usted que NO se requieren muchos pasos para lograr lo que quiero hacer?	5,42
¿Cree usted que NO necesita esforzarse para usar el chatbot?	5,14
Puedo usarlo sin instrucciones escritas.	5,5
¿Considera usted que el chatbot presenta inconsistencias?	4,64
¿Considera que el software es adecuado para cualquier tipo de persona?	5,79
Si se equivocó, ¿considera usted que puede el sistema comprende los errores rápida y fácilmente?	5,29
Puedo usarlo con éxito cada vez.	5,93
Promedio	5,4

Tabla 2: Facilidad de uso del sistema SAGAMO medido con el cuestionario USE

En el Gráfico 1 se visualiza los promedios de cada una de las preguntas formuladas según el cuestionario USE para la evaluación de facilidad de uso.

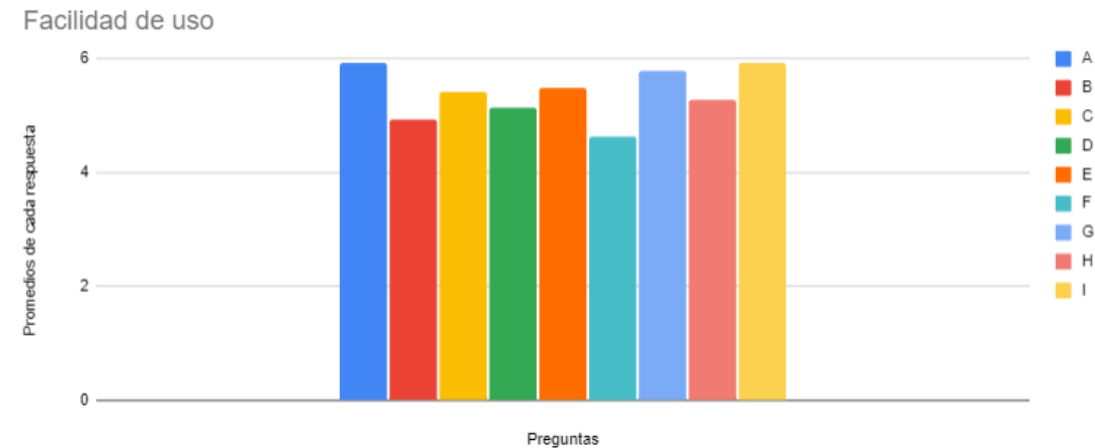
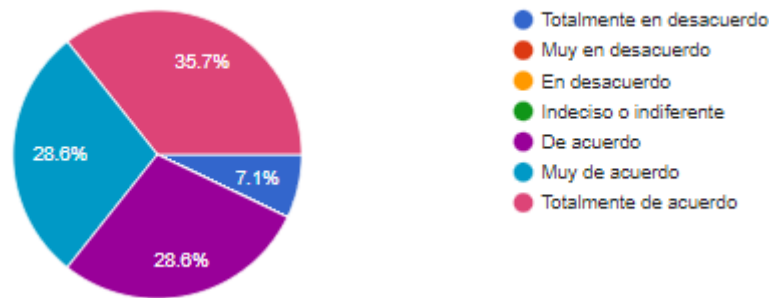


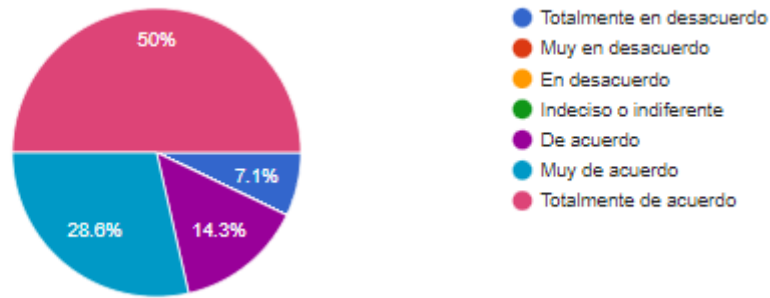
Gráfico 1: Facilidad de uso

**Preguntas:**

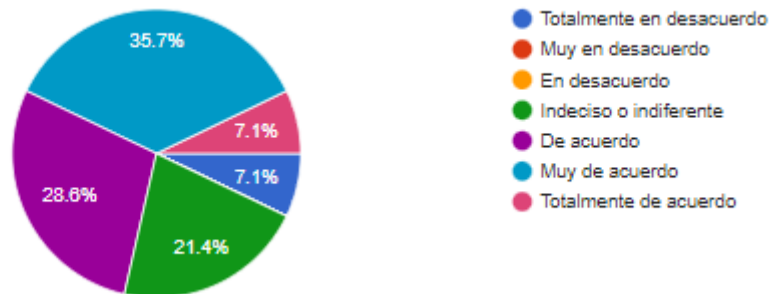
A: ¿Considera usted que el chatbot es fácil de usar?



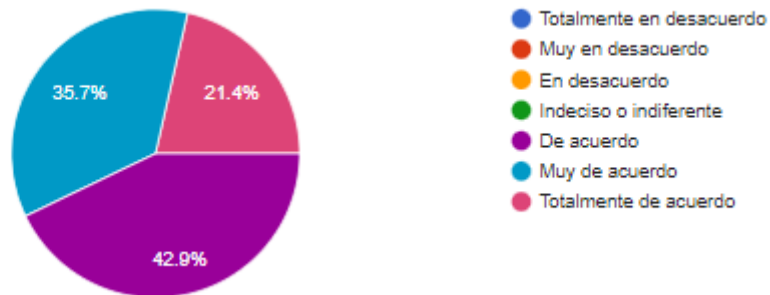
B: ¿Considera usted que el chatbot es simple de usar?



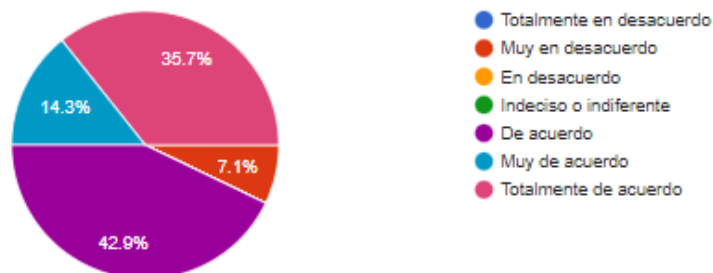
C: La interfaz de usuario es adecuada



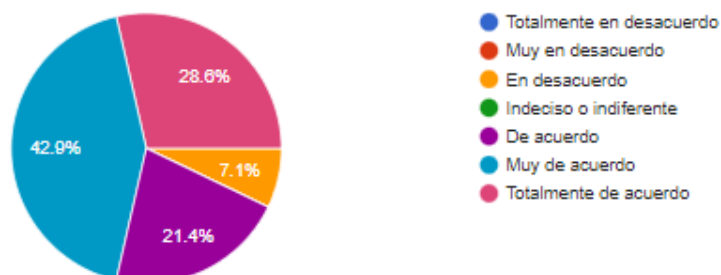
D: ¿Considera usted que NO se requieren muchos pasos para lograr lo que quiero hacer?



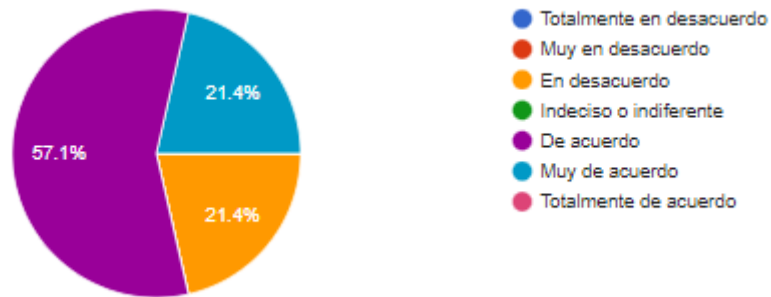
E: ¿Cree usted que NO necesita esforzarse para usar el chatbot?



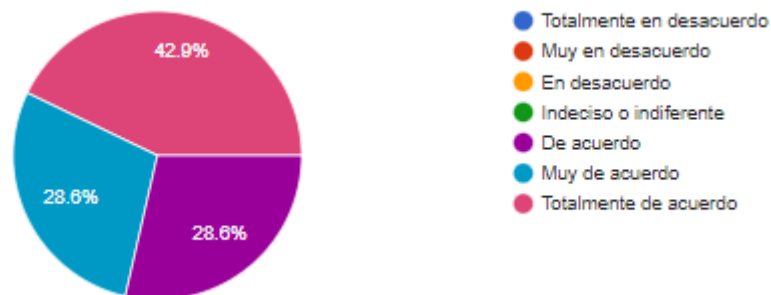
F: Puedo usarlo sin instrucciones escritas.



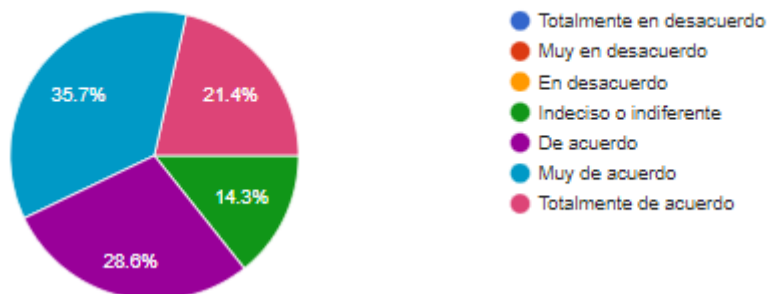
G: ¿Considera usted que el chatbot presenta inconsistencias?



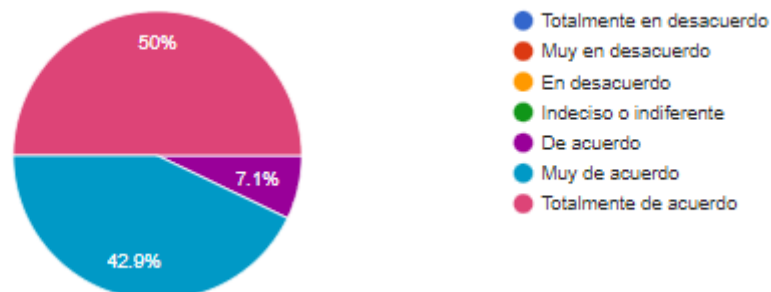
H: ¿Considera que el software es adecuado para cualquier tipo de persona?



I: Si se equivocó, ¿considera usted que puede el sistema comprende los errores rápida y fácilmente?



J: Puedo usarlo con éxito cada vez.



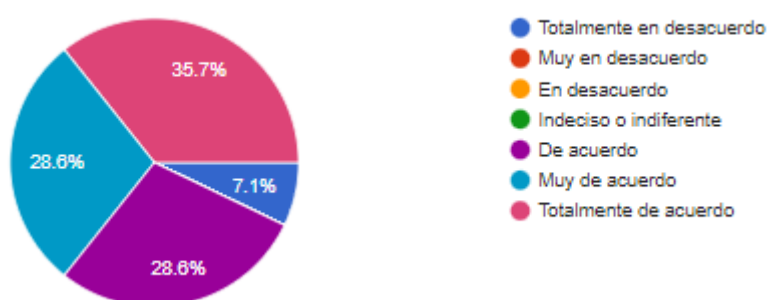
La Tabla 3 presenta los valores que se obtuvieron en las evaluaciones a través de la aplicación del cuestionario USE en la subcaracterísticas facilidad de uso; cómo se puede observar, se cuenta con un total de:

Facilidad de Uso							
Casos	Promedio	Dev. estándar	Min.	Max.	Rango	Varianza	Coefficiente de variación
14	5.39	0.42	4,64	5,92	1,29	0.17	0.07

*Tabla 3: Valores de facilidad de uso*

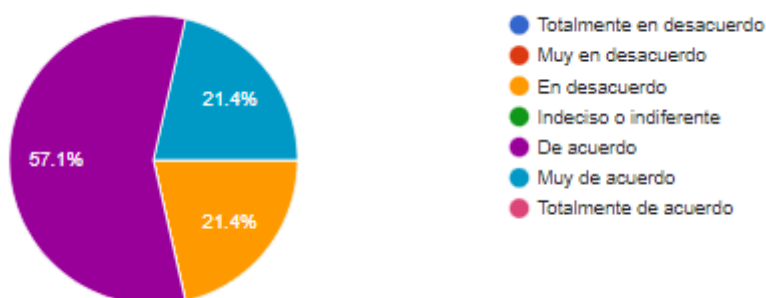
La desviación estándar con 0.42 indica la separación existente entre los datos, es así que se puede decir que los valores obtenidos en la evaluación de las subcaracterísticas facilidad de uso no se encuentran con un valor de dispersión significativa.

En el gráfico 2 se puede visualizar el porcentaje de cada valoración que se obtuvo de los encuestados en la pregunta “¿Considera usted que el chatbot es simple de usar?” perteneciente a la subcaracterísticas de facilidad de uso siendo esta la más valorada con un promedio de 5,93 según la escala de Likert.



*Gráfico 2: Ítem mejor valorado en la facilidad de uso*

En el Gráfico 3 se puede visualizar el porcentaje de cada valoración que se obtuvo de los encuestados en la pregunta “¿Considera usted que el chatbot presenta inconsistencias?”, perteneciente a la subcaracterísticas de facilidad de uso siendo esta la menos valorada con un promedio de 4,64 según la escala de Likert.



*Gráfico 3: Ítem menos valorado en la facilidad de uso*

## ANEXO

### Cuestionario de usabilidad y satisfacción

#### ***Evaluación del Software PharmacyBot***

¡Gracias por participar en nuestra evaluación de usabilidad para PharmacyBot! Su retroalimentación es crucial para mejorar nuestra aplicación y brindarle una experiencia óptima. Por favor, tómese unos minutos para completar este formulario y compartir sus impresiones sobre la usabilidad del software PharmacyBot.

Instrucciones:

1. Responda cada pregunta marcando la casilla que mejor refleje su opinión o experiencia.
2. Siéntase libre de proporcionar comentarios adicionales en los espacios provistos para cada pregunta.
3. Sus respuestas serán tratadas de manera confidencial y se utilizarán solo con fines de mejora del producto.

Considera usted que el chatbot es fácil de usar? \*

- ☐ Totalmente en desacuerdo
- ☐ Muy en desacuerdo
- ☐ En desacuerdo
- ☐ Indeciso o indiferente
- ☐ De acuerdo
- ☐ Muy de acuerdo
- ☐ Totalmente de acuerdo

Considera usted que el chatbot es simple de usar? \*

- ☐ Totalmente en desacuerdo
- ☐ Muy en desacuerdo
- ☐ En desacuerdo
- ☐ Indeciso o indiferente
- ☐ De acuerdo
- ☐ Muy de acuerdo
- ☐ Totalmente de acuerdo

La interfaz de usuario es adecuada \*

- ☐ Totalmente en desacuerdo
- ☐ Muy en desacuerdo
- ☐ En desacuerdo
- ☐ Indeciso o indiferente
- ☐ De acuerdo
- ☐ Muy de acuerdo
- ☐ Totalmente de acuerdo

¿Considera usted que **NO** se requieren muchos pasos para lograr lo que quiero hacer? \*

- ☐ Totalmente en desacuerdo
- ☐ Muy en desacuerdo
- ☐ En desacuerdo
- ☐ Indeciso o indiferente
- ☐ De acuerdo
- ☐ Muy de acuerdo
- ☐ Totalmente de acuerdo

¿Cree usted que **NO** necesita esforzarse para usar el chatbot? \*

- ☐ Totalmente en desacuerdo
- ☐ Muy en desacuerdo
- ☐ En desacuerdo
- ☐ Indeciso o indiferente
- ☐ De acuerdo
- ☐ Muy de acuerdo
- ☐ Totalmente de acuerdo

Puedo usarlo sin instrucciones escritas. \*

- ☐ Totalmente en desacuerdo
- ☐ Muy en desacuerdo
- ☐ En desacuerdo
- ☐ Indeciso o indiferente
- ☐ De acuerdo
- ☐ Muy de acuerdo
- ☐ Totalmente de acuerdo

¿Considera usted que el chatbot presenta inconsistencias? \*

- ☐ Totalmente en desacuerdo
- ☐ Muy en desacuerdo
- ☐ En desacuerdo
- ☐ Indeciso o indiferente
- ☐ De acuerdo
- ☐ Muy de acuerdo
- ☐ Totalmente de acuerdo

¿Considera que el software es adecuado para cualquier tipo de persona? \*

- ☐ Totalmente en desacuerdo
- ☐ Muy en desacuerdo
- ☐ En desacuerdo
- ☐ Indeciso o indiferente
- ☐ De acuerdo
- ☐ Muy de acuerdo
- ☐ Totalmente de acuerdo

Si se equivocó, ¿considera usted que puede el sistema comprende los errores rápida y fácilmente? \*

- ☐ Totalmente en desacuerdo
- ☐ Muy en desacuerdo
- ☐ En desacuerdo
- ☐ Indeciso o indiferente
- ☐ De acuerdo
- ☐ Muy de acuerdo
- ☐ Totalmente de acuerdo

¿Pudo usar con éxito el chatbot cada vez que lo necesitaba? \*

- ☐ Totalmente en desacuerdo
- ☐ Muy en desacuerdo
- ☐ En desacuerdo
- ☐ Indeciso o indiferente
- ☐ De acuerdo
- ☐ Muy de acuerdo
- ☐ Totalmente de acuerdo

## Bibliografía

Monday.com. (s.f.). Recuperado el 16 de mayo de 2023, de <https://monday.com/>

Smith, J. (2021). Project Planning and Management with Monday.com. New York, NY: Wiley.

Johnson, M. (2020). Agile Project Management Using Monday.com. Journal of ProjectManagement, 15(2), 45-60.

Brown, A., & Davis, S. (2019). Enhancing Project Planning Efficiency with Monday.com: A Case Study. International Journal of Project Management, 25(3), 112-128.

Garcia, R., & Jones, T. (2018). Leveraging Monday.com for Effective Project Planning and Collaboration. *Journal of Modern Project Management*, 12(1), 76-92.

Diseño del flujo de conversación de un chatbot. (2017, octubre 27). Github.io.  
<https://robertovillarejo.github.io/2017/10/27/chatbot-dialog-flow.html>

Leah. (2021, abril 30). 6 pasos para un flujo de conversación de chatbot intachable. Userlike.  
<https://www.userlike.com/es/blog/flujo-conversacion>