



TECNOLÓGICO NACIONAL DE MÉXICO
INSTITUTO TECNOLÓGICO DE APIZACO

**“SISTEMA DE ADMINISTRACIÓN DE TICKETS PARA
MESA DE SERVICIO (HELP DESK)” MÓDULO 3: DISEÑO Y
CREACIÓN DE APLICACIÓN MÓVIL PARA EL REGISTRO
DE TICKETS.**

TITULACIÓN INTEGRAL INFORME DE RESIDENCIA PROFESIONAL

**PARA OBTENER EL TÍTULO PROFESIONAL DE:
INGENIERO EN TECNOLOGÍAS DE LA INFORMACIÓN Y
COMUNICACIONES**

**PRESENTA:
DAVID ALFONSO RODRÍGUEZ BÁEZ**

**NOMBRE DEL ASESOR:
MTRO. HIGINIO NAVA BAUTISTA**



APIZACO, TLAX., MARZO 2024



Apizaco, Tlaxcala, **11/octubre /2023**

OFICIO No: SC/OPV/196/23

ASUNTO: Liberación de proyecto
para titulación integral

MARTÍN ROJAS RAMÍREZ
JEFE DEL DEPTO. DE DIVISIÓN DE ESTUDIOS PROFESIONALES
P R E S E N T E.

POR ESTE MEDIO LE INFORMO QUE HA SIDO LIBERADO EL SIGUIENTE PROYECTO PARA LA TITULACIÓN INTEGRAL.

NOMBRE DEL EGRESADO:	DAVID ALFONSO RODRÍGUEZ BÁEZ
CARRERA:	INGENIERÍA EN TECNOLOGÍAS DE LA INFORMACIÓN Y COMUNICACIONES
NO. DE CONTROL:	17371060
NOMBRE DEL PROYECTO:	"SISTEMA DE ADMINISTRACIÓN DE TICKETS PARA MESA DE SERVICIO (HELP DESK)" MÓDULO 3: DISEÑO Y CREACIÓN DE APLICACIÓN MÓVIL PARA EL REGISTRO DE TICKETS.
PRODUCTO U OPCIÓN:	TITULACIÓN INTEGRAL INFORME DE RESIDENCIA PROFESIONAL

AGRADEZCO DE ANTEMANO SU VALIOSO APOYO EN ESTA IMPORTANTE ACTIVIDAD PARA LA FORMACIÓN PROFESIONAL DE NUESTROS EGRESADOS.

ATENTAMENTE

*Excelencia en Educación Tecnológica®
Pensar para Servir, Servir para Triunfar®*

ALAN AUGUSTO GALLEGOS CUELLAR
JEFE DEL DEPTO. DE SISTEMAS Y COMPUTACIÓN

c.c.p. División de Estudios Profesionales
c.c.p. Depto. de Servicios Escolares
c.c.p. Archivo
AAGC/mrol*



SECRETARÍA DE EDUCACIÓN PÚBLICA
INSTITUTO TECNOLÓGICO NACIONAL
DE MÉXICO
INSTITUTO TECNOLÓGICO DE APIZACO
**DEPARTAMENTO DE
SISTEMAS Y COMPUTACIÓN**

HIGINIO NAVA BAUTISTA
ASESOR INTERNO





Apizaco, Tlax., 26/febrero/2024
No. OFICIO: DEP-CT/212/2024

ASUNTO: AUTORIZACIÓN DE IMPRESIÓN

DAVID ALFONSO RODRIGUEZ BAEZ
EGRESADO DE LA CARRERA DE
INGENIERÍA EN TECNOLOGÍAS DE LA INFORMACIÓN Y COMUNICACIONES
CON NÚMERO DE CONTROL: 17371060
P R E S E N T E

Por este medio, me permito informar a usted que por aprobación de la comisión revisora asignada para valorar el trabajo que mediante la opción **TITULACIÓN INTEGRAL INFORME DE RESIDENCIA PROFESIONAL** cuyo tema es: **"SISTEMA DE ADMINISTRACIÓN DE TICKETS PARA MESA DE SERVICIO (HELP DESK) MÓDULO 3: DISEÑO Y CREACIÓN DE APLICACIÓN MÓVIL PARA EL REGISTRO DE TICKETS"**, conforme a lo establecido en el procedimiento para la obtención del Título Profesional en el Instituto Tecnológico, la División de Estudios Profesionales a mi cargo le otorga:

AUTORIZACIÓN DE IMPRESION

Debiendo entregar **3** empastados del mismo, apropiadamente encuadernados, a la brevedad, para presentar su acto de recepción profesional.

Sin otro particular por el momento, le envió un cordial saludo.

ATENTAMENTE

*Excelencia en Educación Tecnológica®
Pensar para Servir, Servir para Triunfar®*


MARTÍN ROJAS RAMÍREZ
JEFE DE LA DIVISIÓN DE ESTUDIOS PROFESIONALES



c.p. archivo



Av. Instituto Tecnológico No. 418, San Andrés Ahuashuatepec, Municipio de Tzompantepec,
Tlaxcala, Mex. C.P. 90491 Tel. (241) 4172010 Ext. 304, e-mail:
dep_titulacion@apizaco.tecnm.mx



Agradecimientos

Me gustaría expresar mi más sincero agradecimiento a las siguientes personas que han desempeñado un papel fundamental en la realización de mi persona y formación de ingeniería profesional:

En primer lugar, estoy agradecido con mi padre, Alfredo Rodríguez García, por su inquebrantable apoyo a lo largo de mi trayectoria académica y por los sacrificios que ha hecho para guiar mi éxito. Tu sabiduría me ha convertido en la persona que hoy soy.

También extiendo mi agradecimiento a mi madre, Zenaida Báez Hernández, por su amor, paciencia y comprensión sin límites. Su constante tranquilidad y palabras de aliento me han dado la fuerza y la determinación para superar los desafíos. Su apoyo inquebrantable ha sido una fuerza impulsora detrás de mis logros, y siempre estaré agradecido por su presencia en mi vida.

Estoy en deuda con mi asesor escolar, Higinio Nava Bautista, por su invaluable guía, experiencia y tutoría. Sus comentarios perspicaces, críticas constructivas y dedicación inquebrantable han sido fundamentales para dar forma a la dirección del proyecto. Su aliento y confianza en mis habilidades me han motivado a ir más allá de mis límites y luchar por la excelencia.

Además, me gustaría expresar mi gratitud a mi asesor profesional, Lic. Francisco Flores Avendaño, por sus profundos conocimientos, experiencia técnica y valiosas sugerencias. Su disposición a compartir su conocimiento y experiencia ha sido fundamental para dar forma al curso de mi investigación.

Apreciaré sus contribuciones para la vida personal y profesional.

Índice

Agradecimientos.....	1
Resumen	2
Capítulo I: Generalidades del proyecto	3
1.1 Introducción.....	3
1.2 Datos de la empresa.....	4
1.2.1 Descripción de la Empresa	4
1.2.2 Misión	4
1.2.3 Visión	5
1.2.4 Áreas de trabajo del estudiante	5
1.2.5 Nombre del proyecto:	5
1.3 Problema a resolver	5
1.4 Objetivo general	6
1.4.1 Objetivos específicos	6
1.5 Justificación	7
Capítulo II: Marco teórico	7
2.1 Bases teóricas:.....	7
2.2 Antecedentes	8
2.3 Competencias en el mercado	9
2.3.1 OsTicket	9
2.3.2 Freshdesk	11
2.3.3 UVDesk	13
2.3.4 Remedy Service Desk de BMC	15
2.3.5 OTRS	18

2.3.6 ngDesk	20
2.4 Metodologías	23
2.4.1 Buenas practicas	23
2.4.2 ISO 20000	24
2.4.3 ITIL 4	26
Capítulo III: Planeación.....	28
3.1 Explicación y entendimiento del modelo de negocio	28
3.2 Metodología Agile	28
3.3 Desarrollo basado en componentes.....	28
3.4 Integración continua y entrega continua	28
3.5 Versionado de código y control de versiones.....	29
3.6 Requerimientos funcionales	29
3.6.1 Acceso de usuarios al sistema:.....	29
3.6.2 Gestión de SLA.....	30
3.6.3 Proceso para levantamiento de Tickets en plataforma / app móvil.....	30
3.6.4 Atención del Ticket	30
3.7 Requerimientos no funcionales	31
3.8 Historias de usuario	33
3.8.1 Registro de usuarios.....	33
3.8.2 Levantamiento de ticket.....	34
3.8.3 Atención del ticket	36
3.9 Diagramas BPM-Ticketing.....	38
3.9.1 Diagrama BPM-Ticketing para el levantamiento y creación de tickets.	38
3.9.2 Diagrama BPM-Ticketing para el seguimiento y atención de tickets.	39
Capítulo IV: Desarrollo (diseño, maquetación y codificación).	40

4.1 Diseño y maquetación de la interfaz (Front-End).....	40
4.1.1 Inicio de sesión.....	40
4.1.2 Crear cuenta.....	41
4.1.3 Levantamiento de incidencias	42
4.1.4 Menú-sesión-incidencia	43
4.1.5 Captura de incidencia.....	44
4.1.6 Levantamiento de incidencia.....	45
4.1.7 Generar ticket	46
4.1.8 Menú-sesión-Generar ticket	47
4.1.9 Chat	48
4.1.10 Menú – Sesión - Chat	49
4.1.11 Tickets enviados	50
4.1.12 Menú-tickets-enviados	51
4.2 Codificación	52
4.2.1 Codificación lógica de rutas de la aplicación móvil (Back-End).	52
4.2.2 Importación de componentes en la aplicación móvil (Back-End).	53
4.2.3 Codificación lógica de servicios en la aplicación móvil (Back-End).	54
4.2.4 Codificación lógica del componente Inicio de sesión (Back-End).	55
4.2.5 Codificación de la interfaz (Front-End).	56
4.2.6 Codificación lógica del componente Crear cuenta (Back-End).	57
4.2.7 Codificación de la interfaz (Front-End).	58
4.2.8 Codificación lógica del componente Incidencia (Back-End).	59
4.2.9 Codificación de la interfaz (Front-End).	60
4.2.10 Codificación lógica del componente Chat (Back-End).	61
4.2.11 Codificación de la interfaz Chat (Front-End).	62

4.2.12 Codificación de la interfaz Generar tickets (Front-End).....	63
4.2.13 Codificación de la interfaz Tickets enviados (Front-End).	64
Capítulo V: Resultados	65
5.1 Aplicación finalizada	65
5.1.1 Inicio de sesión	65
5.1.2 Crear cuenta	66
5.1.3 Incidencia	67
5.1.4 Captura de incidencia.....	68
5.1.5 Levantamiento de incidencia	69
5.1.6 Redacción de incidencia	70
5.1.7 Menú-sesión-incidencia	71
5.1.8 Generar ticket.....	72
5.1.9 Menú-sesión-Generar ticket.....	73
5.1.10 Interfaz del Chat.....	74
5.1.11 Respuestas con Inteligencia Artificial mediante ChatGPT	75
5.1.12 Menú – Sesión - Chat.....	76
5.1.13 Tickets enviados.....	77
5.1.14 Menú – Sesión – Tickets enviados.....	78
Capítulo VI: Conclusiones.....	79
Capítulo VII: competencias desarrolladas	80
Capítulo VIII: Glosario de términos	81
Capítulo IX: Referencias	83

Índice de ilustraciones

Ilustración 1 Diagrama BPM-Ticketing para el levantamiento y creación de tickets.....	38
Ilustración 2 Diagrama BPM-Ticketing para el seguimiento y atención de tickets.	39
Ilustración 3 Inicio de sesión	40
Ilustración 4 Crear cuenta.....	41
Ilustración 5 Levantamiento de incidencias	42
Ilustración 6 Menú-sesión-incidencia.....	43
Ilustración 7 Captura de incidencia	44
Ilustración 8 Levantamiento de incidencia	45
Ilustración 9 Generar ticket	46
Ilustración 10 Menú-sesión-Generar ticket	47
Ilustración 11 Chat.....	48
Ilustración 12 Menú – Sesión - Chat	49
Ilustración 13 Tickets enviados	50
Ilustración 14 Menú-tickets-enviados	51
Ilustración 15 Codificación lógica de rutas de la aplicación móvil (Back-End).	52
Ilustración 16 Importación de componentes en la aplicación móvil (Back-End).....	53
Ilustración 17 Codificación lógica de servicios en la aplicación móvil (Back-End).....	54
Ilustración 18 Codificación lógica del componente Inicio de sesión (Back-End).	55
Ilustración 19 Codificación de la interfaz (Front-End).	56
Ilustración 20 Codificación lógica del componente Crear cuenta (Back-End).	57
Ilustración 21 Codificación de la interfaz (Front-End).	58
Ilustración 22 Codificación lógica del componente Incidencia (Back-End).	59
Ilustración 23 Codificación de la interfaz (Front-End).	60
Ilustración 24 Codificación lógica del componente Chat (Back-End).	61
Ilustración 25 Codificación de la interfaz Chat (Front-End).	62
Ilustración 26 Codificación de la interfaz Generar tickets (Front-End).	63
Ilustración 27 Codificación de la interfaz Tickets enviados (Front-End).....	64
Ilustración 28 Inicio de sesión	65
Ilustración 29 Crear cuenta.....	66

Ilustración 30 Incidencia	67
Ilustración 31 Captura de incidencia	68
Ilustración 32 Levantamiento de incidencia	69
Ilustración 33 Redacción de incidencia	70
Ilustración 34 Menú-sesión-incidencia.....	71
Ilustración 35 Generar ticket	72
Ilustración 36 Menú-sesión-Generar ticket	73
Ilustración 37 Interfaz del Chat	74
Ilustración 38 Respuestas con Inteligencia Artificial mediante ChatGPT	75
Ilustración 39 Menú – Sesión - Chat	76
Ilustración 40 Tickets enviados	77
Ilustración 41 Menú – Sesión – Tickets enviados	78

Resumen

El presente trabajo expone la creación de un software de administración de tickets denominado mesas de servicios el cual es capaz de agilizar la solución de incidencias y/o problemas dentro de una empresa y dar una solución en el menor tiempo posible para, poder cubrir las necesidades del cliente/empresa, para ello este sistema estará enfocado a la generación de tickets para poder dar una solución pronta y eficaz a cualquier problema presentado dentro de su empresa o negocio.

Capítulo I: Generalidades del proyecto

1.1 Introducción

Mesa de ayuda o “Help Desk” por su nombre en inglés, el cual está enfocada a todos los tipos de comercio, empresa u organización con el fin de brindar una forma más completa de poder atender las necesidades o problemas de una manera más eficaz y rápida posible presentando una solución mediante metodologías e investigaciones.

Las herramientas "Help Desk" se han convertido en una parte integral de las operaciones de soporte y gestión de servicios en muchas organizaciones. Estas herramientas surgieron en respuesta a la necesidad de gestionar eficientemente las solicitudes y problemas de los usuarios, y han evolucionado a lo largo del tiempo para adaptarse a las demandas cambiantes del entorno tecnológico.

Muchas herramientas de soporte técnico ofrecen portales de autoservicio para clientes, lo que les permite crear y rastrear sus propios tickets de soporte, acceder a artículos de la base de conocimientos y encontrar respuestas a problemas comunes. Esta funcionalidad de autoservicio ayuda a mejorar la satisfacción del cliente y reduce la carga de trabajo de los equipos de soporte.

1.2 Datos de la empresa



Nombre: Soft-TI Apizaco

Giro, ramo o sector: Consultoría

Dirección: BLVD 16 de septiembre 506 interior 4, Apizaco Tlaxcala.

RFC: FOAF700917QTA

Nombre del titular de la empresa: Lic. Francisco Flores Avendaño “director general”.

Teléfono: 241- 107- 3452

1.2.1 Descripción de la Empresa

Sof-Ti es una empresa dedicada a proveer consultoría TI (Tecnología de la Información), servicios tecnológicos, integrales y desarrollo de sistemas de información, somos un grupo de consultores especialistas que generamos valor a la empresa con la elaboración de soluciones de TI enfocadas a cumplir las expectativas de nuestros clientes.

1.2.2 Misión

Fortalecer y empoderar a la empresa a través del uso adecuado de las tecnologías de la información.

1.2.3 Visión

Ser una empresa líder en transformación digital e innovadora dedicada a fortalecer las estrategias tecnológicas de las empresas.

1.2.4 Áreas de trabajo del estudiante

Desarrollador de aplicaciones móviles.

1.2.5 Nombre del proyecto:

“Sistema de Administración de Tickets para Mesa de Servicio (Help Desk)”

Módulo 3: diseño y creación de aplicación móvil para el registro de tickets.

1.3 Problema a resolver

Actualmente en el mercado existen muchos sistemas de mesas de servicio el cual tienen algunas limitantes como lo son los altos costos de licenciamiento o la falta de componentes que hagan más fácil el proceso de atención y levantamiento de un ticket ya que una mesa de servicio es un sistema que es diseñado y utilizado específicamente en empresas u organizaciones, por esa razón no hay un sistema en el mercado que se integre o se adapte a cualquier sistema de negocio, empresa u organización, y lo más importante, un sistema que sea portable, móvil con funcionalidades nativas que le den un valor agregado frente a la competencia existente.

1.4 Objetivo general

Diseñar e implementar una herramienta para la gestión y administración de tickets bajo el marco de referencia ITIL y mejora continua, dando seguimiento al flujo de trabajo de cualquier tipo de organización que permita medir y configurar de manera adecuada los niveles de servicio establecidos con el propósito de poder aumentar los índices de calidad del servicio.

1.4.1 Objetivos específicos

- Diseñar y crear el Back-end de la aplicación móvil.
- Diseñar y crear el Front-end de la aplicación móvil.
- Construcción de componentes nativos para la compatibilidad de los dispositivos móviles.
- Gestión y administración de base de datos de usuarios y perfiles.
- Reducir tiempos en el levantamiento de incidencias.
- Reducir tiempos en la creación del ticket.
- Facilitar la comunicación para la atención del ticket.
- Reducir el tiempo para el seguimiento del ticket.
- Diseñar y crear una arquitectura eficaz y evolutiva.
- Diseñar y crear una interfaz fluida y evolutiva.
- Diseñar y crear una aplicación funcional para cualquier dispositivo móvil.

1.5 Justificación

Basado en las buenas prácticas para la gestión de servicios de tecnologías de información orientadas a conseguir servicios más efectivos y de mejor calidad como los son (ISO 20000 e ITIL), se crea un espacio en IT el cual surge este proyecto el cual requiere una atención más estricta y específica hacia los aspectos que se incluirán en nuestro marco de trabajo.

Capítulo II: Marco teórico

2.1 Bases teóricas:

En el presente apartado se concentran investigaciones previas, conceptos y consideraciones teóricas que sustentan y respaldan este proyecto.

A continuación, el objetivo del estudio del mercado para el desarrollo del proyecto.

1. **Enfoque:** El estudio de mercado se centra en comprender el entorno y las características del mercado en el que se introducirá el producto o servicio. Analiza a los clientes potenciales, la competencia, las tendencias del mercado, los hábitos de consumo y otros factores relevantes.
2. **Objetivos:** El objetivo principal del estudio de mercado es identificar y evaluar la demanda del producto o servicio, conocer las necesidades y preferencias del público objetivo, y determinar la viabilidad comercial del proyecto. Proporciona información para tomar decisiones estratégicas de marketing, como segmentación de mercado, posicionamiento y estrategias de precios.
3. **Datos recopilados:** El estudio de mercado implica recopilar y analizar datos cualitativos y cuantitativos relacionados con el mercado objetivo. Esto puede incluir encuestas, entrevistas, análisis de datos demográficos, investigaciones de mercado existentes y análisis de la competencia.

2.2 Antecedentes

Los antecedentes de las herramientas "Help Desk" se remontan a las décadas de 1980 y 1990, cuando las empresas comenzaron a darse cuenta de la importancia de brindar un soporte técnico efectivo a sus usuarios. Inicialmente, el soporte se gestionaba a través de llamadas telefónicas, lo que resultaba en una falta de organización y dificultades para realizar un seguimiento adecuado de los problemas.

Con el crecimiento de las organizaciones y la creciente complejidad de los sistemas tecnológicos, surgió la necesidad de herramientas más estructuradas y automatizadas para gestionar el soporte técnico. Esto llevó al desarrollo de las primeras soluciones de "Help Desk" como sistemas basados en tickets, que permitían a los usuarios enviar solicitudes de soporte y recibir respuestas a través de una plataforma centralizada.

Con el tiempo, estas herramientas evolucionaron para incluir características como la asignación automática de tickets, la priorización de problemas, el seguimiento del estado de las solicitudes y la generación de informes. Además, se incorporaron funcionalidades de gestión de conocimientos, permitiendo a los usuarios acceder a bases de conocimientos y documentos técnicos para resolver problemas comunes por sí mismos.

Con el advenimiento de Internet, las herramientas de "Help Desk" también se trasladaron al entorno en línea, lo que facilitó aún más el acceso y la colaboración entre los usuarios y los equipos de soporte técnico. Se desarrollaron soluciones basadas en la web que permitían a los usuarios enviar solicitudes de soporte a través de formularios en línea y recibir actualizaciones en tiempo real sobre el estado de sus tickets.

2.3 Competencias en el mercado

2.3.1 OsTicket

OsTicket es un sistema de gestión de tickets de código abierto que se utiliza ampliamente para brindar soporte al cliente y administrar solicitudes de ayuda en muchas organizaciones.

OsTicket proporciona una interfaz fácil de usar para que los agentes de soporte puedan recibir, organizar y responder a los tickets de manera eficiente. Los usuarios pueden enviar sus consultas a través de diferentes canales, como correo electrónico, formularios web o incluso Twitter, y el sistema los convierte en tickets que se pueden asignar a los agentes correspondientes.

Características:

Algunas de las características comunes de OsTicket incluyen:

- **Sistema de tickets:** permite la creación, seguimiento y gestión de tickets de soporte.
- **Automatización de procesos:** se pueden configurar respuestas automáticas, reglas de enrutamiento y notificaciones para mejorar la eficiencia.
- **Base de conocimientos:** una base de datos de artículos y preguntas frecuentes que los usuarios pueden consultar antes de enviar un ticket.
- **Integraciones:** OsTicket se puede integrar con otras herramientas y servicios, como sistemas de chat en vivo, correo electrónico, CRM, entre otros.
- **Informes y análisis:** proporciona datos y métricas para evaluar el rendimiento y la productividad del equipo de soporte.

Cabe destacar que OsTicket es una solución de código abierto, lo que significa que su código fuente está disponible para que los usuarios lo modifiquen y personalicen según sus necesidades. También existe una versión comercial llamada "osTicket Plus" que ofrece características adicionales y soporte técnico.

Limitaciones:

1. **Personalización limitada:** Aunque OsTicket permite cierta personalización a través de su código fuente abierto, realizar modificaciones más avanzadas puede requerir conocimientos técnicos especializados. Esto puede dificultar la adaptación de la herramienta a necesidades específicas de la organización.
2. **Curva de aprendizaje inicial:** Para los nuevos usuarios y administradores, OsTicket puede tener una curva de aprendizaje inicial, especialmente si no están familiarizados con el sistema o no tienen experiencia previa en la gestión de un sistema de tickets. Requiere tiempo y esfuerzo para comprender todas las características y funcionalidades disponibles.
3. **Limitaciones de escalabilidad:** Aunque OsTicket es adecuado para pequeñas y medianas empresas, puede tener limitaciones en términos de escalabilidad para grandes organizaciones con altos volúmenes de tickets y múltiples equipos de soporte. En tales casos, es posible que se requieran soluciones más robustas y escalables.
4. **Soporte técnico:** Si bien OsTicket cuenta con una comunidad activa de usuarios y desarrolladores que brindan soporte a través de foros y recursos en línea, la disponibilidad de soporte técnico oficial puede ser limitada, especialmente en comparación con soluciones comerciales que ofrecen asistencia dedicada.
5. **Actualizaciones y mantenimiento:** Mantener OsTicket actualizado puede requerir tiempo y esfuerzo, especialmente si se han realizado modificaciones personalizadas en el código fuente. Además, como es una herramienta de código abierto, es responsabilidad del usuario asegurarse de aplicar las actualizaciones y parches de seguridad necesarios.

Versión móvil:

OsTicket en sí mismo no ofrece una aplicación móvil oficial para dispositivos móviles.

Sin embargo, dado que OsTicket es una solución basada en la web, se puede acceder a ella a través de un navegador móvil en dispositivos móviles, lo que permite una experiencia similar a la de una aplicación.

2.3.2 Freshdesk

Freshdesk es una popular plataforma de servicio al cliente en la nube que ofrece una amplia gama de características para la gestión de tickets y la atención al cliente.

Características:

Sistema de tickets: permite la creación, seguimiento y gestión de tickets de soporte de manera centralizada.

Automatización de procesos: se pueden establecer reglas y flujos de trabajo automatizados para agilizar las respuestas y asignaciones de tickets, así como para enviar notificaciones a los clientes y agentes.

Canales de soporte omnicanal: Freshdesk admite múltiples canales de comunicación, como correo electrónico, chat en vivo, teléfono, redes sociales y portal de autoservicio. Esto brinda a los clientes opciones para comunicarse según sus preferencias:

- **Base de conocimientos:** permite crear una base de datos de artículos, preguntas frecuentes y soluciones que los clientes pueden buscar antes de presentar un ticket. Esto ayuda a reducir la carga de trabajo del equipo de soporte al proporcionar respuestas autoayuda.

- **Automatización de respuestas:** se pueden configurar respuestas automáticas y respuestas predefinidas para consultas comunes, lo que ahorra tiempo y mejora la eficiencia del equipo de soporte.
- **Informes y análisis:** Freshdesk ofrece informes y análisis detallados para evaluar el rendimiento del equipo de soporte, medir la satisfacción del cliente y obtener información valiosa sobre patrones de soporte.
- **Integraciones:** Freshdesk se integra con una amplia gama de herramientas y servicios populares, como CRM, chat en vivo, automatización de marketing y más, lo que permite una mayor funcionalidad y sincronización de datos.

Limitaciones:

Costo: Freshdesk ofrece diferentes planes de precios, incluido un plan gratuito con funciones limitadas y planes de pago con características adicionales. Para algunas organizaciones, especialmente aquellas con presupuestos limitados, el costo de los planes de pago puede resultar un obstáculo.

1. **Complejidad inicial:** Al igual que con cualquier sistema de gestión de tickets, puede haber una curva de aprendizaje inicial para familiarizarse con la interfaz y las funcionalidades de Freshdesk. Esto puede requerir tiempo y capacitación adicional para los nuevos usuarios y administradores.
2. **Personalización limitada en planes más bajos:** Algunas características avanzadas y personalizaciones más específicas pueden estar disponibles solo en planes de pago más altos. Esto puede ser una limitación si una organización requiere una personalización más profunda para adaptarse a sus necesidades específicas.
3. **Dependencia de la conexión a Internet:** Freshdesk es una solución basada en la nube, lo que significa que requiere una conexión a Internet confiable para acceder y utilizar la plataforma. Si hay interrupciones en la conectividad o problemas con el proveedor de servicios en la nube, esto puede afectar la disponibilidad y el rendimiento de Freshdesk.

4. **Soporte técnico limitado en planes gratuitos:** En el plan gratuito de Freshdesk, el soporte técnico puede estar limitado o no estar disponible. Esto significa que puede haber una falta de asistencia directa de los representantes de Freshdesk si se encuentran problemas o dificultades técnicas.
5. **Restricciones de almacenamiento de datos:** Dependiendo del plan elegido, Freshdesk puede tener limitaciones en el almacenamiento de datos, como la cantidad de tickets almacenados o el espacio para adjuntos. Esto puede requerir una gestión cuidadosa de los datos almacenados y una posible necesidad de eliminar o archivar tickets antiguos.

Versión móvil:

Freshdesk ofrece aplicaciones móviles oficiales que se pueden instalar en dispositivos móviles con los sistemas operativos iOS (iPhone/iPad) y Android. Estas aplicaciones están diseñadas específicamente para brindar una experiencia optimizada en dispositivos móviles y facilitar la gestión de tickets y la interacción con los clientes mientras estás en movimiento.

2.3.3 UVDesk

UVDesk es una plataforma de código abierto para la gestión de tickets y el soporte al cliente. Proporciona una solución integral para administrar las solicitudes de soporte y brindar un servicio eficiente a los clientes.

Características:

Algunas características clave de UVDesk incluyen:

- **Sistema de tickets:** permite la creación, seguimiento y gestión de tickets de soporte de manera centralizada.

- **Automatización de procesos:** se pueden establecer reglas y flujos de trabajo automatizados para asignar tickets, enviar notificaciones y mejorar la eficiencia en la gestión de solicitudes.
- **Colaboración en equipo:** UVDesk facilita la colaboración entre los miembros del equipo de soporte, lo que les permite trabajar juntos en la resolución de tickets y compartir información relevante.
- **Canales de comunicación múltiples:** admite diferentes canales de comunicación, como correo electrónico, formularios web, chat en vivo y redes sociales, lo que permite a los clientes comunicarse a través de su canal preferido.
- **Base de conocimientos:** UVDesk ofrece una base de conocimientos donde se pueden almacenar artículos, preguntas frecuentes y soluciones para que los clientes puedan buscar respuestas antes de crear un ticket.
- **Personalización:** UVDesk es una plataforma de código abierto, lo que permite personalizar y adaptar la herramienta según las necesidades específicas de la organización.

Limitaciones:

1. **Curva de aprendizaje inicial:** Para los nuevos usuarios y administradores, UVDesk puede tener una curva de aprendizaje inicial, especialmente si no están familiarizados con el sistema o no tienen experiencia previa en la gestión de un sistema de tickets. Puede requerir tiempo y esfuerzo para comprender todas las características y funcionalidades disponibles.
2. **Personalización y desarrollo técnico:** Si bien UVDesk es una plataforma de código abierto y ofrece la posibilidad de personalizar y adaptar la herramienta, realizar modificaciones más avanzadas o implementar nuevas características puede requerir conocimientos técnicos especializados en desarrollo web.
3. **Soporte técnico:** A diferencia de las soluciones comerciales que suelen ofrecer soporte técnico dedicado, el soporte para UVDesk puede ser

limitado, ya que depende en gran medida de la comunidad de usuarios y desarrolladores que brindan asistencia a través de foros y recursos en línea.

4. **Actualizaciones y mantenimiento:** Al ser una plataforma de código abierto, es responsabilidad del usuario realizar actualizaciones y aplicar parches de seguridad. Esto implica gestionar adecuadamente las actualizaciones y mantener la plataforma actualizada, lo que puede requerir tiempo y esfuerzos adicionales.
5. **Limitaciones de escalabilidad:** Si bien UVDesk puede ser adecuado para pequeñas y medianas empresas, puede tener limitaciones en términos de escalabilidad para organizaciones más grandes o con altos volúmenes de tickets. Esto puede afectar el rendimiento y la capacidad de gestión eficiente de la herramienta en escenarios de mayor carga de trabajo.

Versión móvil:

Actualmente, UVDesk no ofrece una aplicación móvil oficial para dispositivos móviles.

Sin embargo, dado que UVDesk es una plataforma basada en la web, se puede acceder a ella a través de un navegador móvil en dispositivos móviles.

2.3.4 Remedy Service Desk de BMC

Remedy Service Desk es una solución popular de gestión de servicios y tickets de BMC, una empresa reconocida en el ámbito de la gestión de servicios de TI.

Remedy Service Desk ofrece una amplia gama de características para el manejo eficiente de tickets y la prestación de servicios de soporte.

Características:

Algunas de las características clave de Remedy Service Desk incluyen:

- **Sistema de tickets:** permite la creación, seguimiento y gestión de tickets de soporte de manera centralizada. Los tickets se pueden asignar a grupos o agentes específicos para su resolución.
- **Automatización de procesos:** se pueden establecer reglas y flujos de trabajo automatizados para acelerar la gestión de tickets, la asignación y la respuesta a las solicitudes de soporte.
- **Catálogo de servicios:** permite definir y gestionar un catálogo de servicios, lo que facilita a los usuarios solicitar los servicios o productos necesarios y a los agentes proporcionar la asistencia correspondiente.
- **Base de conocimientos:** brinda una base de conocimientos centralizada donde se pueden almacenar artículos, preguntas frecuentes y soluciones para que los usuarios finales encuentren respuestas a sus preguntas antes de presentar un ticket.
- **Paneles de control y análisis:** ofrece paneles de control personalizables y herramientas de informes para obtener una visibilidad en tiempo real sobre los indicadores clave de rendimiento (KPI) y el desempeño del equipo de soporte.
- **Integraciones:** Remedy Service Desk se integra con otras herramientas y sistemas, como herramientas de monitoreo, gestión de activos, gestión de cambios y más, para proporcionar una experiencia más completa y una gestión más eficiente.

Limitaciones:

1. **Complejidad y curva de aprendizaje:** Remedy Service Desk es una herramienta sofisticada con muchas características y opciones de configuración. Esto puede resultar en una curva de aprendizaje pronunciada para los nuevos usuarios y administradores. Requiere tiempo y

esfuerzo para familiarizarse con la interfaz y las funcionalidades, y puede requerir capacitación adicional.

2. **Costo:** Remedy Service Desk es una solución de software empresarial y, como tal, puede tener un costo significativo, especialmente para organizaciones más pequeñas o con presupuestos limitados. Los costos pueden incluir licencias, mantenimiento, soporte y posibles integraciones adicionales.
3. **Requisitos de infraestructura:** Para implementar y ejecutar Remedy Service Desk de manera eficiente, puede ser necesario contar con una infraestructura de TI sólida y bien configurada. Esto puede implicar la necesidad de servidores dedicados, capacidad de almacenamiento, recursos de red y personal de TI experimentado.
4. **Personalización y configuración compleja:** Aunque la flexibilidad de Remedy Service Desk permite una amplia personalización y configuración, esto puede requerir habilidades técnicas avanzadas y conocimientos especializados en la herramienta. La implementación y personalización más complejas pueden requerir la asistencia de expertos o consultores de BMC.
5. **Dependencia del soporte de BMC:** Como Remedy Service Desk es una solución propietaria de BMC, la asistencia y el soporte técnico dependen de la disponibilidad y calidad del soporte ofrecido por BMC. Esto puede ser una preocupación si se experimentan problemas críticos o si se necesita un soporte rápido y confiable.
6. **Complejidad en actualizaciones y migraciones:** Las actualizaciones de software y las migraciones a nuevas versiones de Remedy Service Desk pueden ser complejas y requieren una planificación cuidadosa. Las organizaciones deben considerar el impacto de las actualizaciones en la personalización existente, los flujos de trabajo y las integraciones personalizadas.

Versión móvil:

Remedy Service Desk de BMC ofrece una aplicación móvil oficial llamada "BMC Helix Remedy" que se puede instalar en dispositivos móviles con los sistemas operativos iOS (iPhone/iPad) y Android. Esta aplicación móvil permite acceder y gestionar los tickets y servicios desde dispositivos móviles.

2.3.5 OTRS

OTRS (Open-source Ticket Request System) es una plataforma de gestión de tickets de código abierto que se utiliza ampliamente para la administración de servicios de atención al cliente y soporte técnico.

OTRS ofrece una variedad de características y funcionalidades para la gestión eficiente de tickets y la automatización de procesos.

Características:

Algunas de las características clave de OTRS incluyen:

- **Sistema de tickets:** permite la creación, seguimiento y gestión de tickets de soporte de manera centralizada. Los tickets pueden clasificarse, priorizarse y asignarse a agentes o grupos específicos.
- **Automatización de procesos:** OTRS ofrece potentes capacidades de automatización que permiten establecer reglas y flujos de trabajo para la gestión de tickets, la asignación automática, las respuestas automáticas y otras acciones relacionadas con el ciclo de vida del ticket.
- **Gestión de conocimientos:** cuenta con una base de conocimientos integrada donde se pueden almacenar artículos, soluciones y respuestas a preguntas frecuentes para facilitar la resolución rápida de problemas y permitir a los usuarios buscar información antes de crear un ticket.

- **Portal de autoservicio:** proporciona un portal de autoservicio para que los usuarios finales puedan enviar y rastrear sus propios tickets, así como acceder a la base de conocimientos y otros recursos útiles.
- **Integraciones:** OTRS se puede integrar con otras herramientas y sistemas, como sistemas de monitoreo, bases de datos externas, herramientas de chat en vivo y más, para una gestión más completa y eficiente de los servicios de soporte.
- **Informes y análisis:** OTRS ofrece funciones de informes y análisis para medir y evaluar el rendimiento del equipo de soporte, identificar tendencias, generar métricas y obtener una visión general de la carga de trabajo y el desempeño del servicio.

Limitaciones:

1. **Curva de aprendizaje:** OTRS puede tener una curva de aprendizaje pronunciada, especialmente para los nuevos usuarios y administradores sin experiencia previa en la gestión de un sistema de tickets. La configuración inicial y la comprensión de todas las características y funcionalidades pueden requerir tiempo y esfuerzo.
2. **Interfaz de usuario:** Algunos usuarios pueden considerar que la interfaz de usuario de OTRS es menos intuitiva y visualmente atractiva en comparación con otras herramientas de gestión de tickets. Esto puede afectar la usabilidad y la experiencia del usuario.
3. **Personalización y desarrollo técnico:** La personalización avanzada de OTRS o la implementación de nuevas características pueden requerir habilidades técnicas y conocimientos especializados en desarrollo web. Realizar modificaciones más complejas o personalizaciones específicas puede requerir recursos adicionales.
4. **Documentación y soporte:** Aunque OTRS tiene documentación y una comunidad de usuarios activa, algunos usuarios pueden encontrar que la documentación oficial no es tan exhaustiva o detallada como desearían.

Además, el soporte técnico dedicado puede requerir una suscripción adicional o una inversión adicional en servicios de soporte.

5. **Actualizaciones y mantenimiento:** Al ser una solución de código abierto, es responsabilidad del usuario realizar actualizaciones y aplicar parches de seguridad. Esto puede requerir tiempo y esfuerzo adicional para mantener el sistema actualizado y garantizar la seguridad.
6. **Escalabilidad:** Aunque OTRS es adecuado para organizaciones de diferentes tamaños, algunos usuarios pueden encontrar limitaciones en términos de escalabilidad en escenarios de alto volumen de tickets o en organizaciones más grandes. El rendimiento y la capacidad de gestión pueden verse afectados en tales casos.

Versión móvil:

OTRS ofrece una versión móvil para dispositivos móviles llamada "OTRS Mobile App". Esta aplicación móvil está disponible tanto para iOS (iPhone/iPad) como para Android y permite a los usuarios acceder y gestionar los tickets de OTRS desde sus dispositivos móviles.

2.3.6 ngDesk

ngDesk es una plataforma de gestión de tickets y servicio al cliente que permite a las organizaciones administrar y responder a las solicitudes de soporte de manera eficiente.

Características:

Algunas de las características clave de ngDesk incluyen:

- **Sistema de tickets:** ngDesk ofrece un sistema de tickets centralizado que permite a los usuarios crear, rastrear y gestionar tickets de soporte de

manera organizada. Los tickets se pueden asignar a agentes específicos, priorizarse y seguir su progreso hasta su resolución.

- **Automatización de procesos:** la herramienta cuenta con capacidades de automatización que permiten la configuración de reglas y flujos de trabajo para agilizar la gestión de tickets. Esto incluye la asignación automática de tickets, respuestas automáticas y acciones predefinidas basadas en ciertos criterios.
- **Portal de autoservicio:** ngDesk ofrece un portal de autoservicio para que los usuarios finales puedan presentar sus solicitudes de soporte, buscar respuestas en la base de conocimientos y realizar un seguimiento del progreso de sus tickets.
- **Gestión de la base de conocimientos:** la herramienta proporciona una base de conocimientos donde se pueden almacenar artículos y soluciones para consultas frecuentes. Esto ayuda a los usuarios a encontrar respuestas rápidas a sus problemas antes de presentar un ticket.
- **Chat en vivo:** ngDesk cuenta con una función de chat en vivo que permite a los usuarios comunicarse directamente con los agentes de soporte en tiempo real para resolver problemas de manera instantánea.
- **Informes y análisis:** la herramienta ofrece herramientas de generación de informes y análisis para medir el rendimiento del equipo de soporte, evaluar la satisfacción del cliente y obtener información sobre las tendencias y los patrones de soporte.

Limitaciones:

1. **Interfaz de usuario limitada:** Algunos usuarios pueden considerar que la interfaz de usuario de ngDesk es menos intuitiva o menos atractiva visualmente en comparación con otras herramientas de help desk. Esto puede afectar la experiencia del usuario y requerir un período de adaptación.

2. **Personalización limitada:** ngDesk puede tener limitaciones en cuanto a la personalización y adaptabilidad a las necesidades específicas de una organización. Algunas características y funcionalidades pueden no ser totalmente personalizables o requerir conocimientos técnicos más avanzados.
3. **Escalabilidad limitada:** Para organizaciones con grandes volúmenes de tickets y equipos de soporte extensos, ngDesk puede presentar limitaciones en cuanto a su escalabilidad. Esto puede resultar en una experiencia más lenta o dificultades para gestionar grandes volúmenes de solicitudes de soporte.
4. **Integraciones limitadas:** A diferencia de otras herramientas de help desk más establecidas, ngDesk puede tener menos integraciones con otras aplicaciones o sistemas empresariales. Esto puede limitar su capacidad para conectarse y sincronizarse con otras herramientas o plataformas utilizadas en la organización.
5. **Soporte técnico limitado:** Al ser una herramienta menos conocida en comparación con algunas soluciones de help desk más populares, es posible que ngDesk no cuente con una amplia base de usuarios o una comunidad de soporte tan activa. Esto puede resultar en un soporte técnico limitado o tiempos de respuesta más largos para resolver problemas o consultas.
6. **Costos adicionales:** Aunque ngDesk ofrece una versión gratuita con funcionalidad básica, algunas características más avanzadas o personalizaciones pueden requerir una actualización a un plan de pago. Los costos adicionales pueden ser un factor a considerar, especialmente para organizaciones con presupuestos limitados

Versión móvil:

ngDesk no cuenta con una aplicación móvil dedicada para dispositivos móviles. Sin embargo, ngDesk es accesible a través de navegadores web en dispositivos móviles.

2.4 Metodologías

2.4.1 Buenas practicas

La gestión de servicios de tecnologías de información (ITSM, por sus siglas en inglés) se refiere a las prácticas y enfoques utilizados para diseñar, entregar, administrar y mejorar los servicios de TI dentro de una organización. El objetivo principal de la gestión de servicios de TI es garantizar que los servicios de tecnologías de información sean efectivos, eficientes y de alta calidad, alineados con las necesidades y objetivos del negocio.

Las buenas prácticas para lograr servicios más efectivos y de mejor calidad aplicadas en la creación y desarrollo de este proyecto son las siguientes:

- **Enfoque centrado en el cliente:** El cliente debe ser el foco principal en la entrega de servicios de TI. Comprende las necesidades y expectativas de los clientes y diseña servicios que satisfagan sus requerimientos. Mantén una comunicación abierta con los clientes para obtener retroalimentación y mejorar continuamente los servicios.
- **Procesos claros:** Define procesos bien definidos y documentados para todas las actividades relacionadas con la gestión de servicios de TI. Esto incluye la gestión de incidentes, gestión de cambios, gestión de problemas, gestión de configuraciones, entre otros. Los procesos deben ser claros, comprensibles y seguidos por todo el equipo.
- **Aplicación de las mejores prácticas de la industria:** Adopta marcos y estándares reconocidos en la gestión de servicios de TI, como ITIL (IT

Infrastructure Library) u otros marcos similares. Estos marcos proporcionan pautas y buenas prácticas para gestionar eficazmente los servicios de TI.

- **Acuerdos de nivel de servicio (SLA):** Define y acuerda los niveles de servicio con los clientes internos o externos. Los SLA establecen las expectativas de los clientes en términos de calidad, tiempo de respuesta, disponibilidad, capacidad y otros aspectos clave del servicio.
- **Colaboración y el trabajo en equipo:** Promueve la colaboración entre los equipos de TI y otras áreas de la organización. La gestión de servicios de TI implica trabajar en estrecha colaboración con los usuarios finales, los proveedores y otras partes interesadas para garantizar la entrega exitosa de los servicios.
- **Mejoras continuas con una arquitectura evolutiva:** Establece un ciclo de mejora continua para identificar áreas de mejora en los servicios de TI. Realiza revisiones periódicas, recopila datos y métricas, e implementa acciones correctivas y preventivas para optimizar los servicios y garantizar la satisfacción del cliente.
- **Personal adecuado:** Contar con personal capacitado y con las habilidades adecuadas para gestionar los servicios de TI de manera efectiva. Proporciona oportunidades de capacitación y desarrollo profesional para mantener actualizado al equipo.

2.4.2 ISO 20000

La norma ISO/IEC 20000, también conocida como ISO 20000, es un estándar internacional que establece los requisitos para implementar y gestionar un sistema de gestión de servicios de tecnología de la información (TI) de calidad. Fue publicada por la Organización Internacional de Normalización (ISO) y la Comisión Electrotécnica Internacional (IEC) con el objetivo de mejorar la calidad de los servicios de TI y garantizar la satisfacción del cliente.

La norma ISO 20000 se basa en el ciclo de mejora continua de la calidad conocido como PDCA (Planificar, Hacer, Verificar, Actuar) y se compone de dos partes principales:

ISO/IEC 20000-1: Requisitos del sistema de gestión de servicios de TI: Esta parte establece los requisitos que una organización debe cumplir para implementar un sistema de gestión de servicios de TI efectivo. Incluye aspectos como la planificación y diseño del servicio, la gestión de proveedores, la gestión del cambio, la gestión de incidentes y problemas, la gestión del nivel de servicio, entre otros.

ISO/IEC 20000-2: Código de práctica para la gestión de servicios de TI: Esta parte proporciona orientación y recomendaciones para la implementación de los requisitos establecidos en la parte 1. Ofrece ejemplos, casos de estudio y mejores prácticas para ayudar a las organizaciones a gestionar sus servicios de TI de manera efectiva.

La implementación de la norma ISO 20000 tiene varios beneficios, incluyendo:

Mejora de la calidad de los servicios de TI: Ayuda a las organizaciones a establecer y mantener altos estándares de calidad en la prestación de servicios de TI, lo que se traduce en una mayor satisfacción del cliente.

Mayor eficiencia operativa: La norma fomenta la estandarización de los procesos y la optimización de las operaciones de TI, lo que conduce a una mayor eficiencia y reducción de costos.

Gestión proactiva de los servicios: Proporciona un marco para la gestión proactiva de incidentes, problemas y cambios, lo que ayuda a prevenir interrupciones en los servicios y minimiza el impacto de los incidentes.

Cumplimiento de requisitos legales y regulatorios: Ayuda a las organizaciones a cumplir con los requisitos legales y regulatorios relacionados con la prestación de servicios de TI.

Mayor confianza del cliente: La certificación en ISO 20000 demuestra el compromiso de una organización con la calidad y la mejora continua de sus servicios de TI, lo que genera confianza en los clientes y socios comerciales.

2.4.3 ITIL 4

ITIL 4 es la versión más reciente del marco de trabajo ITIL para la gestión de servicios de tecnologías de información (ITSM). Fue lanzado en 2019 y representa una evolución significativa en comparación con las versiones anteriores de ITIL. Esta nueva versión se ha actualizado para abordar los desafíos y demandas actuales del entorno empresarial digital y ágil.

A continuación, se presentan algunos aspectos clave que se tomaron en cuenta para el desarrollo del presente proyecto de ITIL 4:

- **Enfoque en el valor:** ITIL 4 pone un mayor énfasis en la creación y entrega de valor para el cliente. Reconoce la importancia de comprender las necesidades y expectativas del cliente y la necesidad de adaptarse a los cambios rápidos en el mercado y la tecnología para proporcionar servicios valiosos y relevantes.
- **Sistema de valor del servicio:** ITIL 4 introduce el concepto del Sistema de Valor del Servicio (Service Value System, SVS), que es un enfoque holístico para la gestión de servicios de TI. El SVS proporciona una estructura para la coordinación y la integración de todos los componentes, actividades y procesos involucrados en la provisión de servicios de TI.
- **Cuatro dimensiones de la gestión del servicio:** ITIL 4 presenta cuatro dimensiones clave que deben tenerse en cuenta al diseñar e implementar servicios de TI: organización y personas, información y tecnología, socios y proveedores, y valor y resultados. Estas dimensiones ayudan a tener una visión integral de los diferentes aspectos que influyen en la gestión exitosa de los servicios.

- **El enfoque de flujo de valor:** ITIL 4 adopta un enfoque de flujo de valor, inspirado en enfoques ágiles y DevOps. Se centra en la entrega continua de valor a través de la colaboración y la automatización, eliminando silos y mejorando la eficiencia y la calidad en todo el ciclo de vida del servicio.
- **Prácticas de gestión de servicios:** ITIL 4 introduce nuevas prácticas y actualiza algunas de las prácticas existentes. Algunas de las nuevas prácticas incluyen la gestión del valor, la gestión de riesgos, la gestión de la configuración, la gestión del conocimiento y la gestión de la continuidad del servicio. Estas prácticas ofrecen orientación detallada sobre cómo realizar actividades y procesos específicos en la gestión de servicios de TI.
- **Integración con enfoques ágiles y DevOps:** ITIL 4 reconoce la importancia de los enfoques ágiles y DevOps en la entrega de servicios de TI rápidos y de alta calidad. Se alinea con estos enfoques y ofrece orientación sobre cómo integrar las prácticas y los principios ágiles y DevOps dentro del marco de trabajo de ITIL.
- **ITIL 4 como una base sólida para la gestión de servicios de TI:** En un entorno empresarial cada vez más digital y ágil. Su enfoque centrado en el valor, la adaptabilidad y la integración con enfoques ágiles y DevOps lo convierten en una herramienta relevante y actualizada para las organizaciones que buscan mejorar la calidad y eficiencia de sus servicios de TI.

Capítulo III: Planeación

3.1 Explicación y entendimiento del modelo de negocio

En base a la solución propuesta para el desarrollo de nuestro sistema web se planteó el modelo de negocio y creación del primer boceto de diagrama UML con la finalidad de poder entender el modelo de negocio y el enfoque que se le daría a este proyecto.

3.2 Metodología Agile

Metodología Agile: El enfoque ágil es ampliamente utilizado en el desarrollo de software y se adapta bien a proyectos individuales. Puedes adoptar metodologías ágiles como Scrum o Kanban para dividir tu proyecto en tareas más pequeñas y manejables. Estas metodologías te permitirán iterar rápidamente, recibir retroalimentación y ajustar tu desarrollo según sea necesario.

3.3 Desarrollo basado en componentes

Desarrollo basado en componentes: Ionic con Angular se basa en el concepto de componentes reutilizables. Puedes aprovechar este enfoque al máximo desarrollando tu aplicación mediante la creación de componentes bien definidos y reutilizables. Esto te permitirá ahorrar tiempo y esfuerzo a medida que avanza el desarrollo de tu aplicación, ya que podrás utilizar y personalizar los componentes en diferentes partes de la app.

3.4 Integración continua y entrega continua

Integración continua y entrega continua (Continuous Integration and Continuous Delivery, CI/CD): Establecer un flujo de trabajo de CI/CD puede ser beneficioso, incluso en un proyecto individual. Puedes utilizar herramientas como Jenkins o Travis CI para automatizar las pruebas, compilación y entrega de tu aplicación. Esto garantizará que los cambios se integren y entreguen de manera constante, lo que facilitará el seguimiento y la corrección de errores.

3.5 Versionado de código y control de versiones

Versionado de código y control de versiones: Es importante utilizar un sistema de control de versiones, como Git, para mantener un registro de los cambios realizados en tu código. Utiliza ramas (branch) para trabajar en nuevas funcionalidades o correcciones de errores sin afectar la versión principal. Esto facilitará la colaboración y permitirá un mejor manejo de las diferentes etapas de desarrollo.

3.6 Requerimientos funcionales

3.6.1 Acceso de usuarios al sistema:

- El sistema debe reflejar el nombre y logo de la empresa.
- El sistema permitirá ingresar al sistema por medio de un “Login” (usuario y contraseña).
- En el proceso de registro de un usuario nuevo, el sistema debe solicitar los siguientes datos: Nombre de usuario, correo, contraseña.
- Al terminar el proceso de registro, el sistema enviará un correo de confirmación.

3.6.2 Gestión de SLA

- El sistema debe permitir la gestión de SLA proporcionados por la empresa
- El sistema permitirá agregar SLA
- El sistema permitirá la eliminación de SLA
- El sistema permitirá la edición de SLA
- El sistema mostrará el listado de SLA para seleccionar el deseado y atender algún ticket en específico

3.6.3 Proceso para levantamiento de Tickets en plataforma / app móvil

- El sistema permitirá crear el ticket por medio de un botón.
- Al crear un ticket necesito que me permita agregar el origen.
- Al crear un ticket necesito que me permita agregar el destinatario.
- Al crear un ticket necesito que me permita agregar el asunto.
- Al crear un ticket necesito que me permita agregar descripción de la incidencia.
- Al crear un ticket necesito que me permita agregar la evidencia de la incidencia.
- El sistema permitirá enviar el ticket por medio de un botón.
- El sistema debe permitir la adición de imágenes, archivos para complementar la descripción de la incidencia.

3.6.4 Atención del Ticket

- Al realizar la atención de un ticket, el sistema debe permitir el envío de mensajes entre agente/cliente.
- El sistema administrará las tareas como son los tickets y se canalizan hacia un área en específico si es necesario.

- El sistema debe permitir a los usuarios autorizados cambiar el estado de atención del Ticket (Abierto, pendiente, espera, cerrado).
- El sistema permitirá la revisión del estatus de sus Tickets creados.
- El sistema tendrá funciones administrativas.
- Administra las tareas como son los Tickets y se canalizan hacia un área en específico si es necesario.
- Catálogos unificados.
- El sistema mostrará una opción de que el cliente u/o usuario encuentre específicamente lo que necesita dentro de nuestro sistema.
- Gestión unificada de tareas.
- Se puede crear un ticket y ver el estatus de revisión de algún otro ticket
- Automatización inteligente.
- Si se requiere alguna atención de ticket básico como cambio de contraseña y búsqueda de usuario no es necesario levantar un ticket lo puede hacer solo desde la plataforma.
- Administración de datos.
- Niveles de prioridad y/o autorización.

3.7 Requerimientos no funcionales

Requerimientos no funcionales del producto		
No.	Categoría	Descripción
1	Seguridad	Al terminar el registro, se deberá enviar un correo de validación para el nuevo administrador.
2	Seguridad	Los permisos de acceso al sistema podrán ser cambiados solamente por el administrador de acceso a datos.
3	Seguridad	Todos los sistemas deben respaldarse cada 24 horas. Los respaldos deben ser almacenados en una localidad segura ubicada en un edificio distinto al que reside el sistema.
4	Seguridad	Todas las comunicaciones externas entre servidores de datos, aplicación y cliente del sistema deben estar encriptadas utilizando el algoritmo RSA.
5	Eficiencia	El sistema debe ser capaz de procesar N datos por segundo.
6	Eficiencia	Toda funcionalidad del sistema y transacción de negocio

		debe responder al usuario en menos de 5 segundos.
7	Eficiencia	Los datos modificados en la base de datos deben ser actualizados para todos los usuarios que acceden en menos de 2 segundos.
8	Usabilidad	Para hacer uso del sistema web el usuario debe contar con conexión a internet.
9	Usabilidad	El sistema debe contar con manuales de usuario estructurados adecuadamente.
10	Usabilidad	El sistema debe proporcionar mensajes de error que sean informativos y orientados a usuario final.
11	Disponibilidad	El sistema debe tener una disponibilidad del 99,99% de las veces en que un usuario intente accederlo.
12	Disponibilidad	El tiempo para iniciar o reiniciar el sistema no podrá ser mayor a 5 minutos.
13	Disponibilidad	La tasa de tiempos de falla del sistema no podrá ser mayor al 0,5% del tiempo de operación total.
14	Disponibilidad	El promedio de duración de fallas no podrá ser mayor a 15 minutos.
15	Disponibilidad	El promedio de duración de fallas no podrá ser mayor a 15 minutos.
16	Disponibilidad	La probabilidad de falla del Sistema no podrá ser mayor a 0,05.
17	Organizacionales	El procedimiento de desarrollo de software debe desarrollarse bajo el marco de referencia ITIL y mejora continua.
18	Organizacionales	El sistema debe ser desarrollado utilizando las herramientas Visual Studio y Android Studio bajo el lenguaje de programación java, Frame Angular más Ionic.

3.8 Historias de usuario

3.8.1 Registro de usuarios

N° 10	Nombre de historia: registro de usuarios		Sprint:	Tiempo estimado:
Módulo	Acceso de usuarios	Nivel de prioridad	Alto	
Dependencia		Programador asignado	David Alfonso Rodríguez Báez	
Descripción				
Usuario (Como)	Usuario			
Objetivo (Quiero)	Registrar datos (correo electrónico, nombre, apellidos, teléfono, dirección, Id de trabajador o cliente, departamento o rol)			
Beneficio (Para poder)	Ingresar al sistema			
Proceso	Desde la página mostrar opción de registro en caso de no contar con cuenta			
Ponderación	5			

N° 11	Nombre de historia: correo de confirmación - usuario		Sprint:	Tiempo estimado:
Módulo	Acceso de usuarios	Nivel de prioridad	Alto	
Dependencia	Historia 10	Programador asignado	Luis Fernando mendieta	
Descripción				
Usuario (Como)	Usuario			
Objetivo (Quiero)	Recibir correo de confirmación y alta de registro como usuario			
Beneficio (Para poder)	Saber que el alta ha sido correcta y poder ingresar al sistema con el usuario y contraseña			
Proceso	Al terminar el registro, el botón debe agregar al usuario y enviar correo de confirmación			
Ponderación	34			

N° 12	Nombre de historia: recuperar contraseña		Sprint:	Tiempo estimado:
Módulo	Acceso de usuarios	Nivel de prioridad	Medio	
Dependencia		Programador asignado	Luis Fernando Mendieta	
Descripción				
Usuario (Como)	Agente, usuario			
Objetivo (Quiero)	Recuperar la contraseña del usuario			
Beneficio (Para poder)	Recuperar y seguir utilizando el sistema			
Proceso	Dentro del login, agregar opción de recuperación de contraseña, al dar click se enviará un correo registrado con un link en el que pueda modificar la contraseña, confirmando la contraseña con un captcha			
Ponderación	34			

3.8.2 Levantamiento de ticket

N° 13	Nombre de historia: creación de ticket		Sprint:	Tiempo estimado:
Módulo	Levantamiento del Ticket	Nivel de prioridad	Alto	
Dependencia		Programador asignado	David Alfonso Rodríguez Báez	
Descripción				
Usuario (Como)	Administrador, agente y usuario			
Objetivo (Quiero)	Que el sistema me permita crear tickets			
Beneficio (Para poder)	Indicar la incidencia			
Proceso	El sistema debe otorgar la opción de "levantar ticket" por medio de un botón que redirige al usuario a un formulario.			
Ponderación	3			

N° 15	Nombre de historia: llenar ticket		Sprint:	Tiempo estimado:
Módulo	Levantamiento del Ticket	Nivel de prioridad	Alto	
Dependencia	Historia 14	Programador asignado	David Alfonso Rodríguez Báez	
Descripción				
Usuario (Como)	Administrador, agente y usuario			
Objetivo (Quiero)	Que el sistema permita añadir descripción de la incidencia con adición de imágenes, archivos para complementar la descripción de la incidencia.			
Beneficio (Para poder)	Describir y explicar la incidencia a detalle			
Proceso	El sistema debe otorgar las opciones de "destinatario (correo)", "asunto (descripción de incidencia)", y/o adjuntar archivos.			
Ponderación	3			

N° 18	Nombre de historia:	Sprint:	Tiempo estimado:
Módulo	Levantamiento del Ticket	Nivel de prioridad	Alto
Dependencia	Historia 14	Programador asignado	David Alfonso Rodríguez Báez
Descripción			
Usuario (Como)	Usuario		
Objetivo (Quiero)	Que el sistema ofrezca una lista desplegable con las areas de soporte y seleccionar la opción requerida		
Beneficio (Para poder)	Para indicar el área a la corresponde la incidencia		
Proceso	El sistema debe ofrecer una lista desplegable con las áreas de soporte de la empresa y el usuario podrá seleccionar la opción requerida		
Ponderación			

3.8.3 Atención del ticket

N° 21	Nombre de historia: buzón para recepción de Tickets		Sprint:	Tiempo estimado:
Módulo	Atención del Ticket	Nivel de prioridad	Alto	
Dependencia		Programador asignado	David Alfonso Rodríguez Báez	
Descripción				
Usuario (Como)	Mesa de Servicio, agente			
Objetivo (Quiero)	Recibir Tickets levantados			
Beneficio (Para poder)	Dar mejor atención y no dejar Tickets sin abrir			
Proceso	Crear buzón en el panel del agente y de la mesa de servicio, donde recibirá notificación de nuevo Ticket en el buzón			
Ponderación				

N° 22	Nombre de historia: información del Ticket		Sprint:	Tiempo estimado:
Módulo	Atención del Ticket	Nivel de prioridad	Alto	
Dependencia	Historia 21	Programador asignado	David Alfonso Rodríguez Báez	
Descripción				
Usuario (Como)	Mesa de Servicio, agente			
Objetivo (Quiero)	Ver descripción de la incidencia/problema al abrir el Ticket			
Beneficio (Para poder)	Analizar y dar una solución eficaz al Ticket			
Proceso	Abrir el buzón de tickets y al seleccionar el Ticket, se dará mostrará la información que regsitró el cliente			
Ponderación				

N° 23	Nombre de historia: buzón de mensajes		Sprint:	Tiempo estimado:
Módulo	Atención del Ticket	Nivel de prioridad	Medio	
Dependencia	Historia 21	Programador asignado	David Alfonso Rodríguez Báez	
Descripción				
Usuario (Como)	Mesa de Servicio, agente			
Objetivo (Quiero)	Tener buzón de mensajes de atención de Ticket (mensajes del cliente)			
Beneficio (Para poder)	No perder ningún mensaje del cliente			
Proceso	Crear botón que despliegue últimos mensajes. En ambos paneles estará un botón con icono alusivo a mensajes			
Ponderación				

N° 25	Nombre de historia: chat		Sprint:	Tiempo estimado:
Módulo	Atención del Ticket	Nivel de prioridad	Alto	
Dependencia		Programador asignado	David Alfonso Rodríguez Báez	
Descripción				
Usuario (Como)	Mesa de servicio, agente, cliente			
Objetivo (Quiero)	Mantener comunicación en el transcurso de solución del Ticket			
Beneficio (Para poder)	Mejorar la comunicación			
Proceso	Dar seguimiento ante cualquier novedad del ticket en atención.			
Ponderación				

3.9 Diagramas BPM-Ticketing

3.9.1 Diagrama BPM-Ticketing para el levantamiento y creación de tickets.

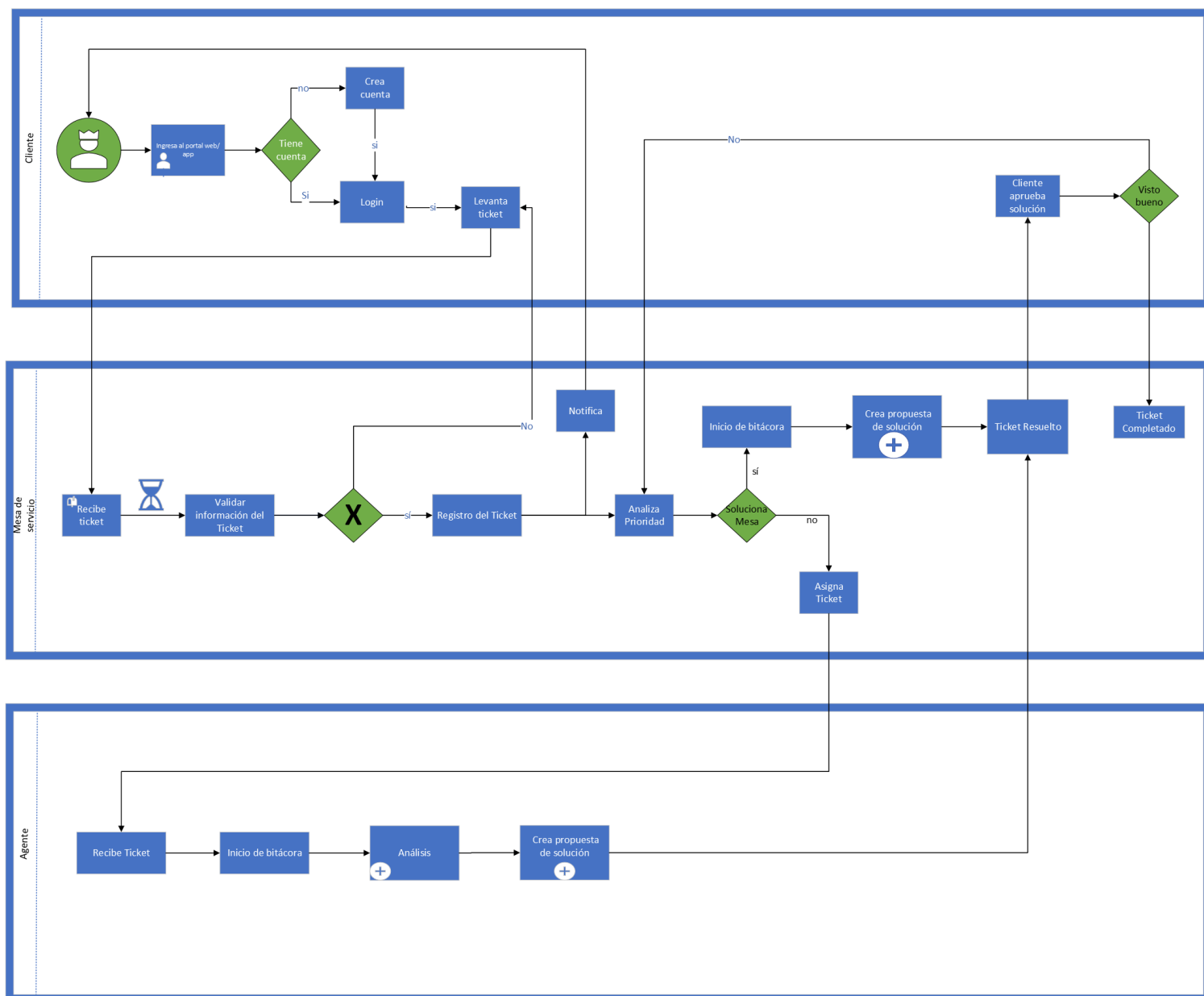


Ilustración 1 Diagrama BPM-Ticketing para el levantamiento y creación de tickets.

3.9.2 Diagrama BPM-Ticketing para el seguimiento y atención de tickets.

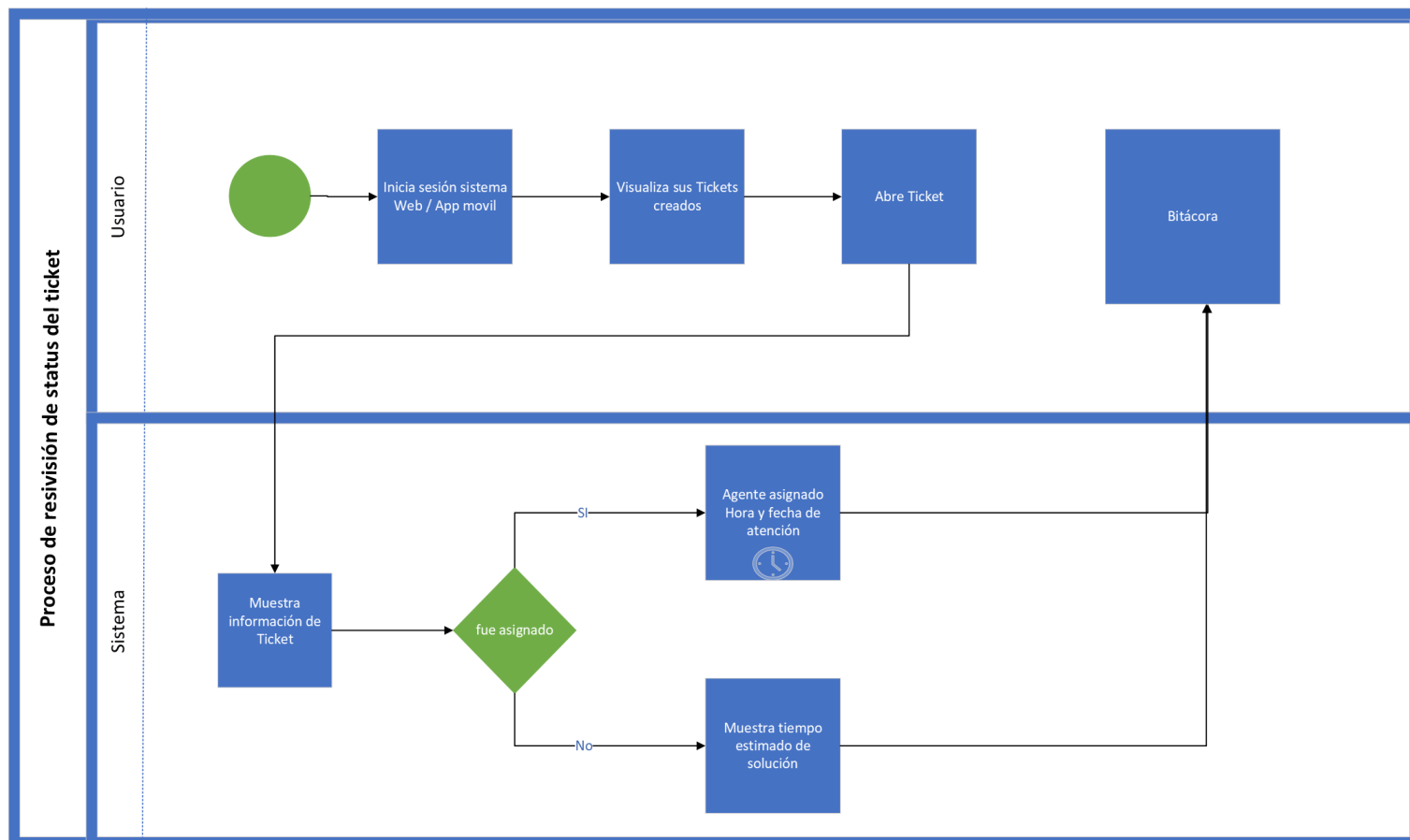


Ilustración 2 Diagrama BPM-Ticketing para el seguimiento y atención de tickets.

Capítulo IV: Desarrollo (diseño, maquetación y codificación).

4.1 Diseño y maquetación de la interfaz (Front-End).

4.1.1 Inicio de sesión

Inicio de sesión

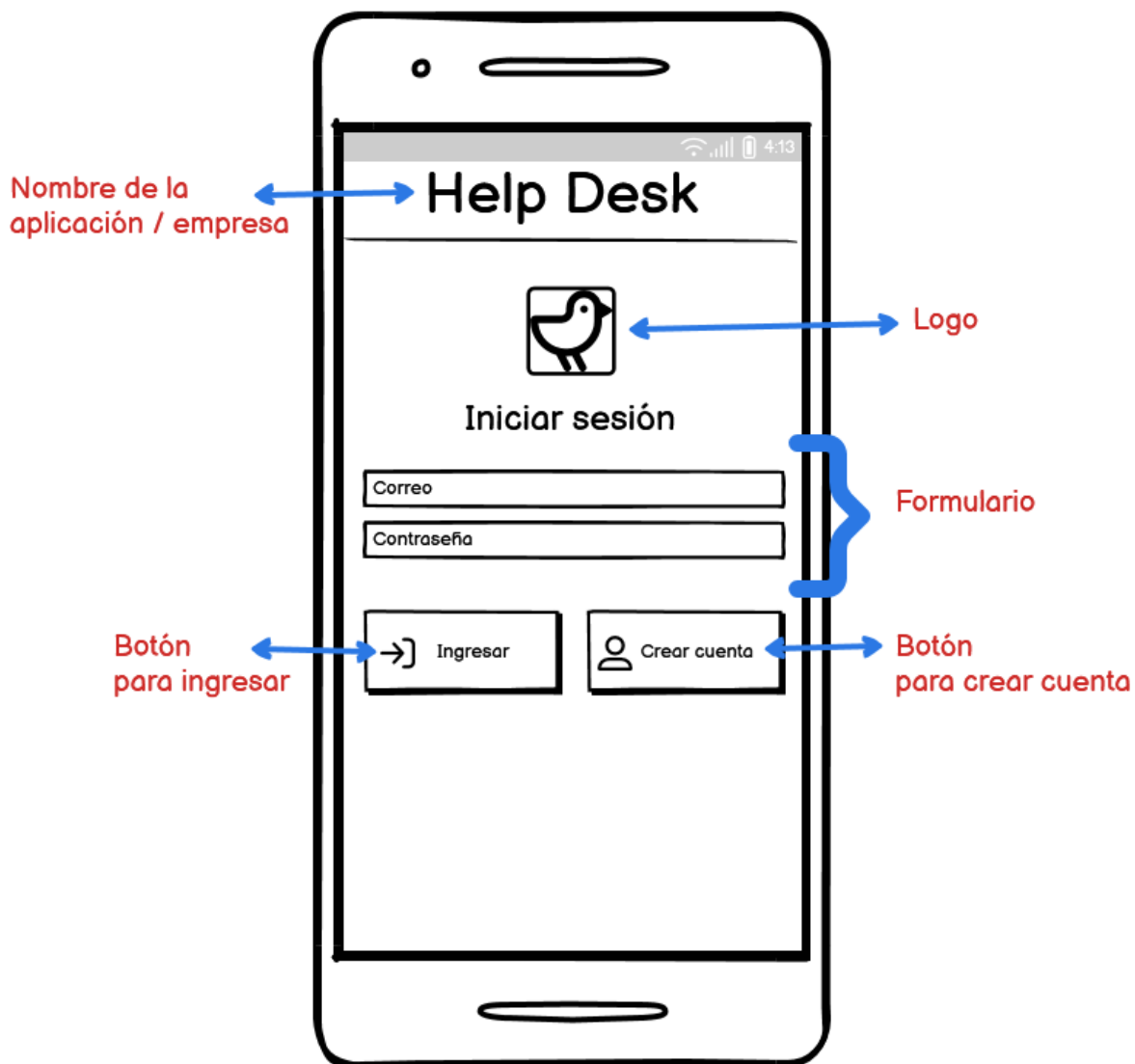


Ilustración 3 Inicio de sesión

4.1.2 Crear cuenta

Crear cuenta

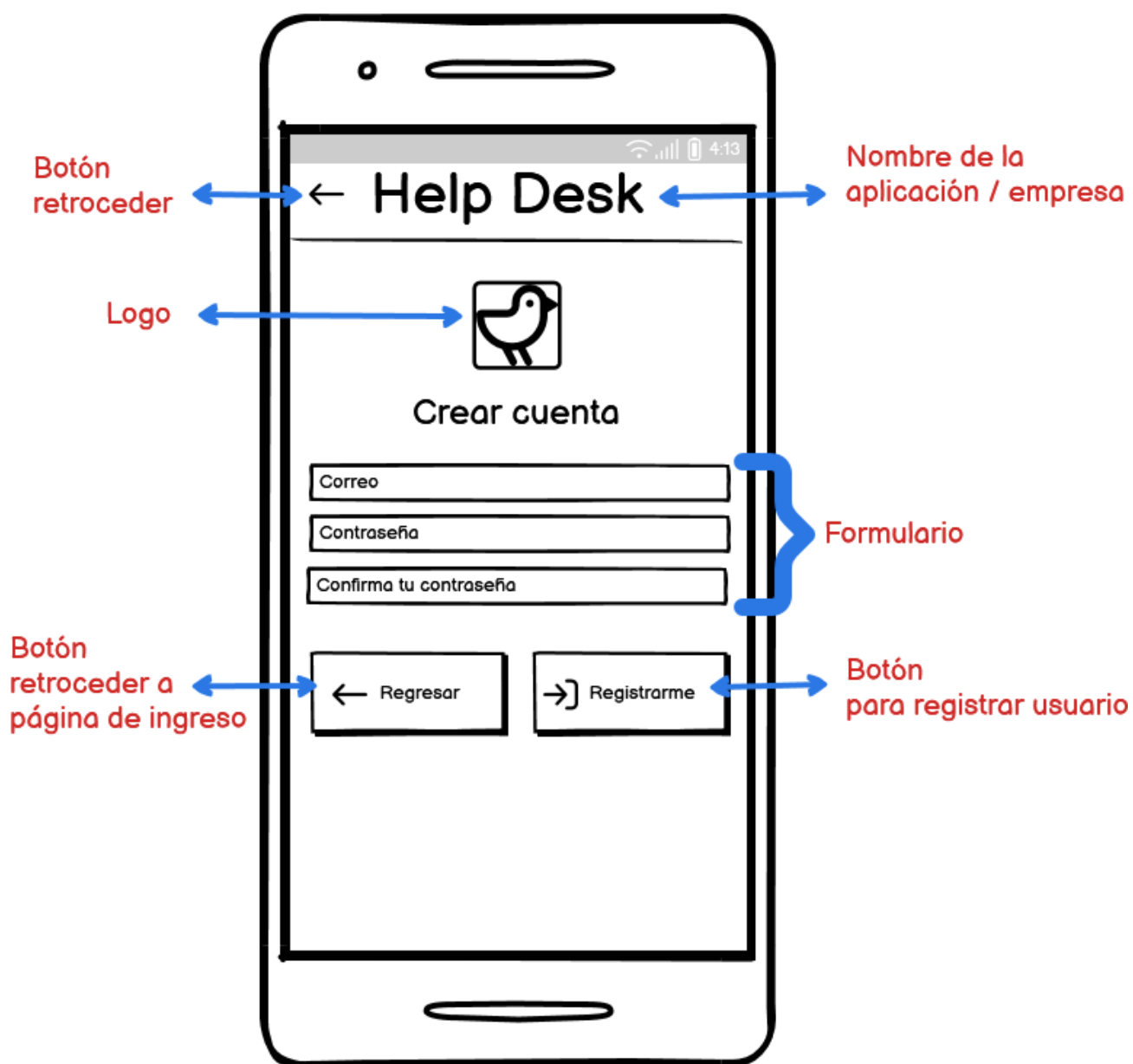


Ilustración 4 Crear cuenta

4.1.3 Levantamiento de incidencias

Captura y levantamiento de incidencias

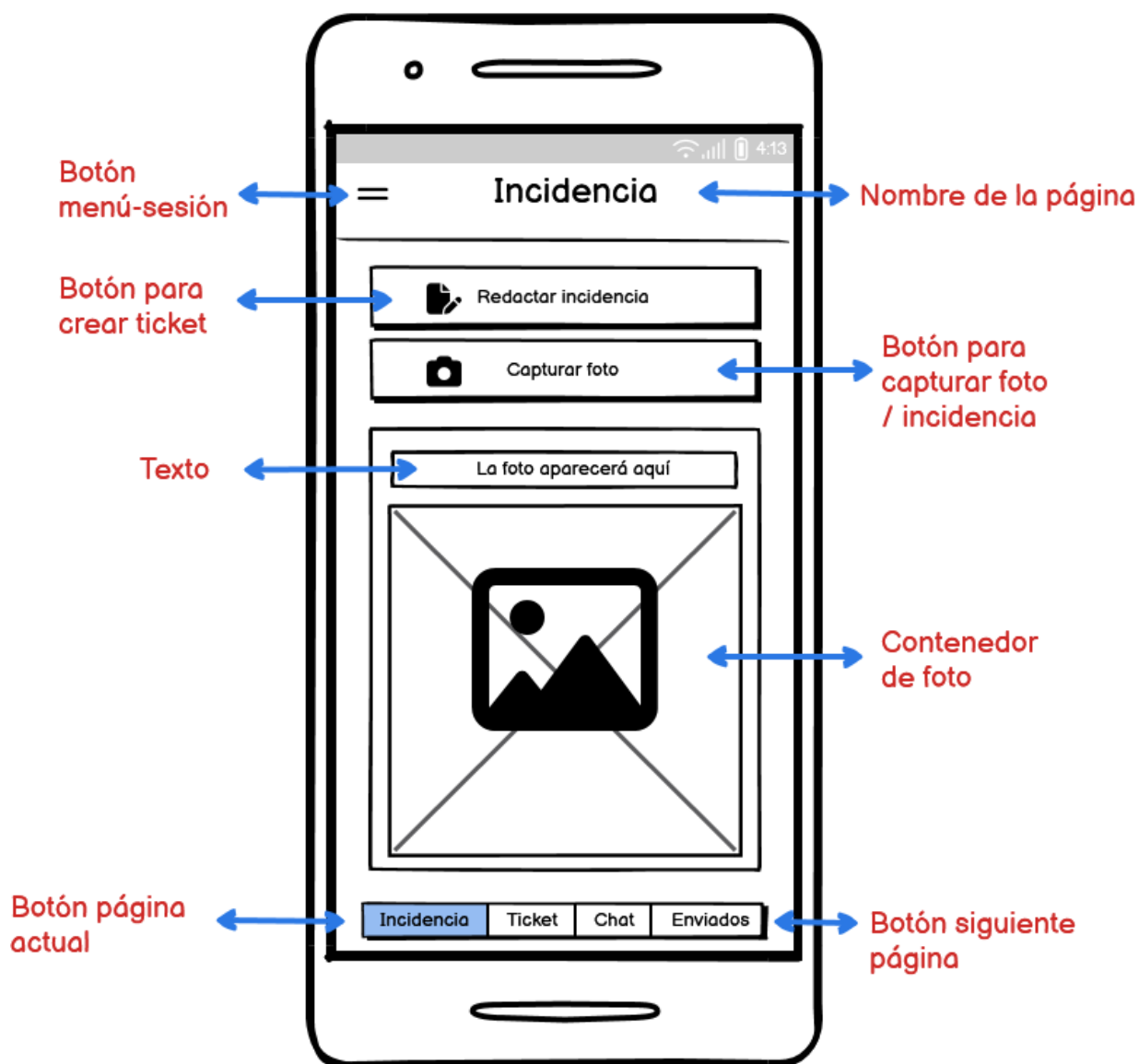


Ilustración 5 Levantamiento de incidencias

4.1.4 Menú-sesión-incidencia

Menú-sesión-Incidencia

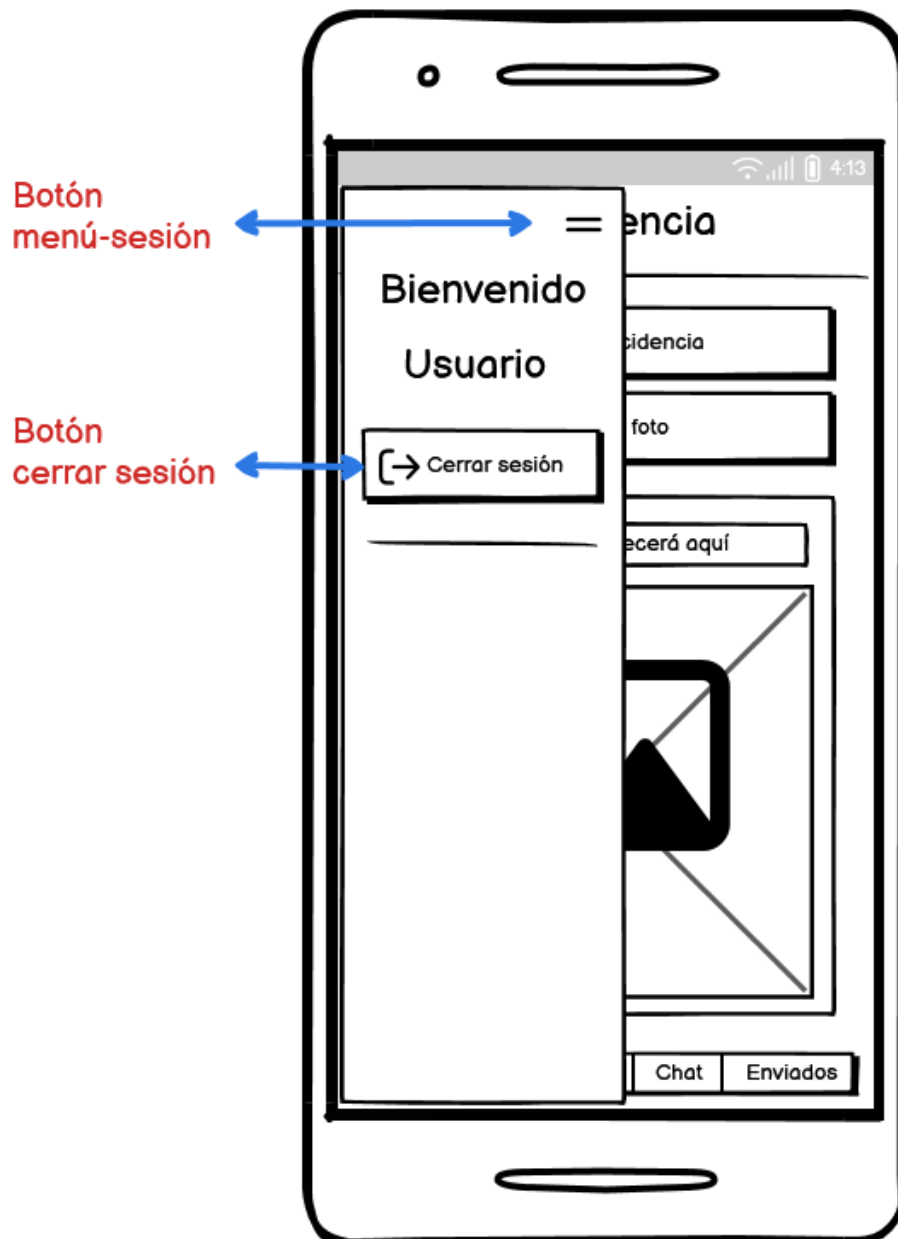


Ilustración 6 Menú-sesión-incidencia

4.1.5 Captura de incidencia

Captura de incidencia

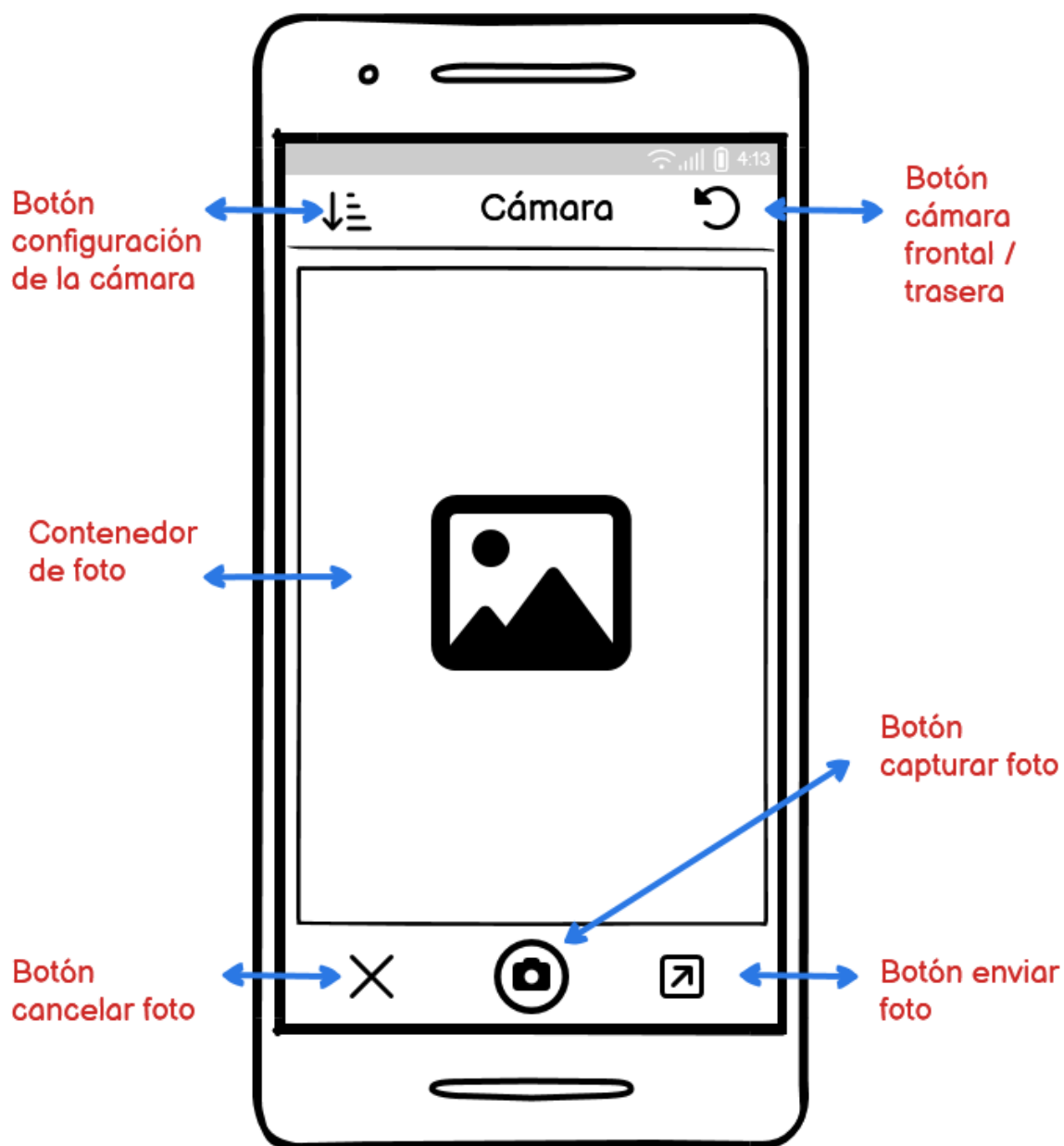


Ilustración 7 Captura de incidencia

4.1.6 Levantamiento de incidencia

Redactar incidencia

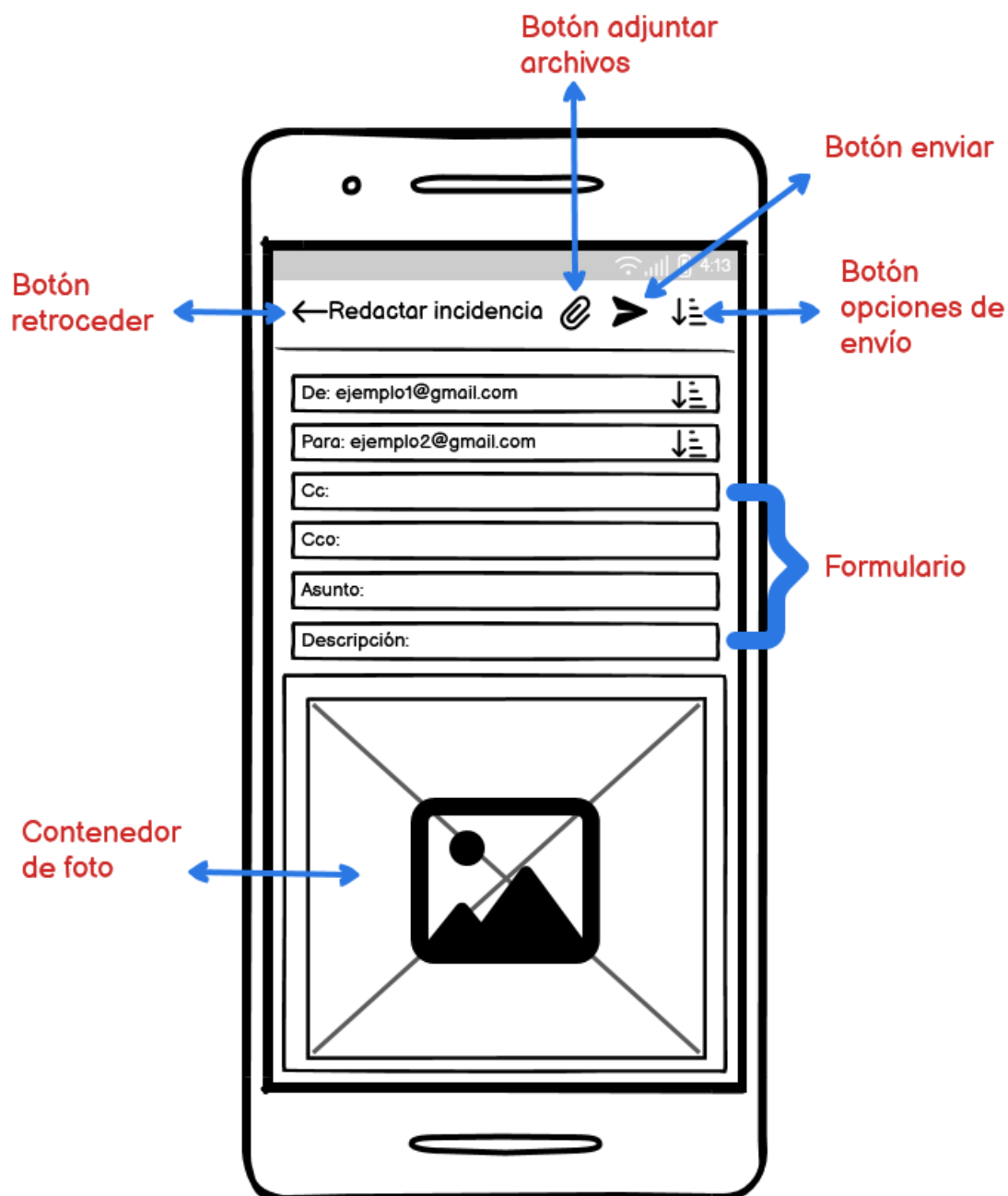


Ilustración 8 Levantamiento de incidencia

4.1.7 Generar ticket

Generar ticket

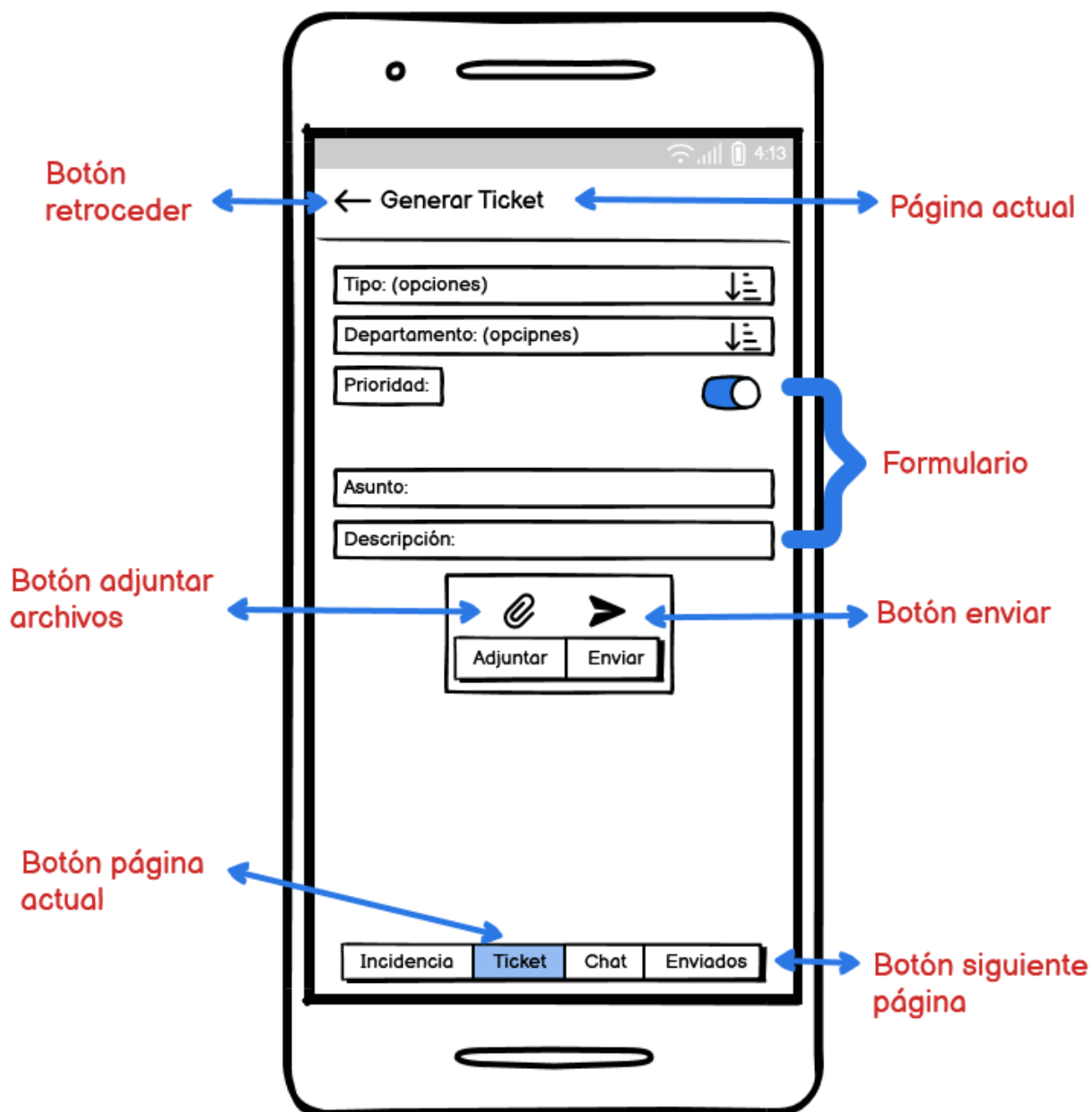


Ilustración 9 Generar ticket

4.1.8 Menú-sesión-Generar ticket

Menú-sesión-Generar ticket

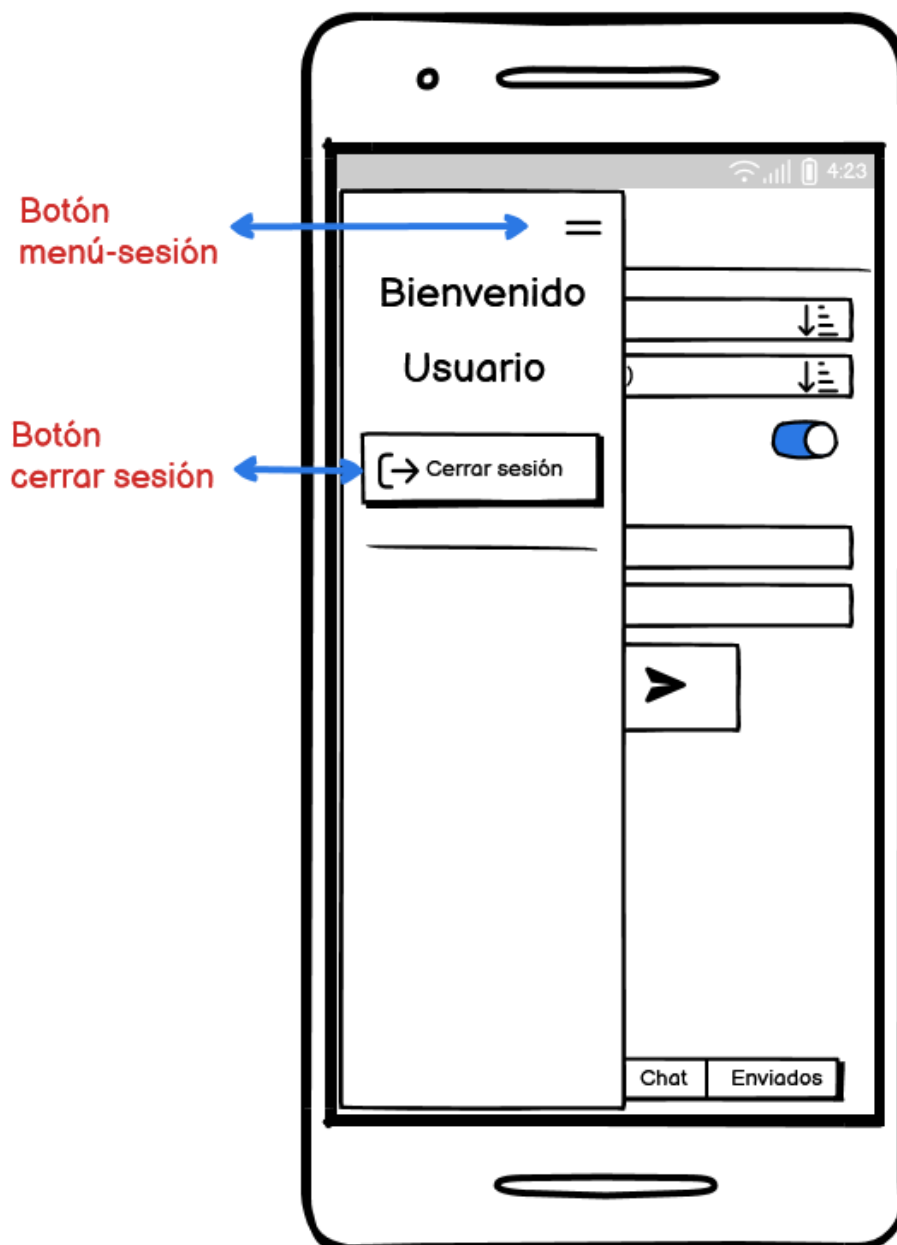


Ilustración 10 Menú-sesión-Generar ticket

4.1.9 Chat

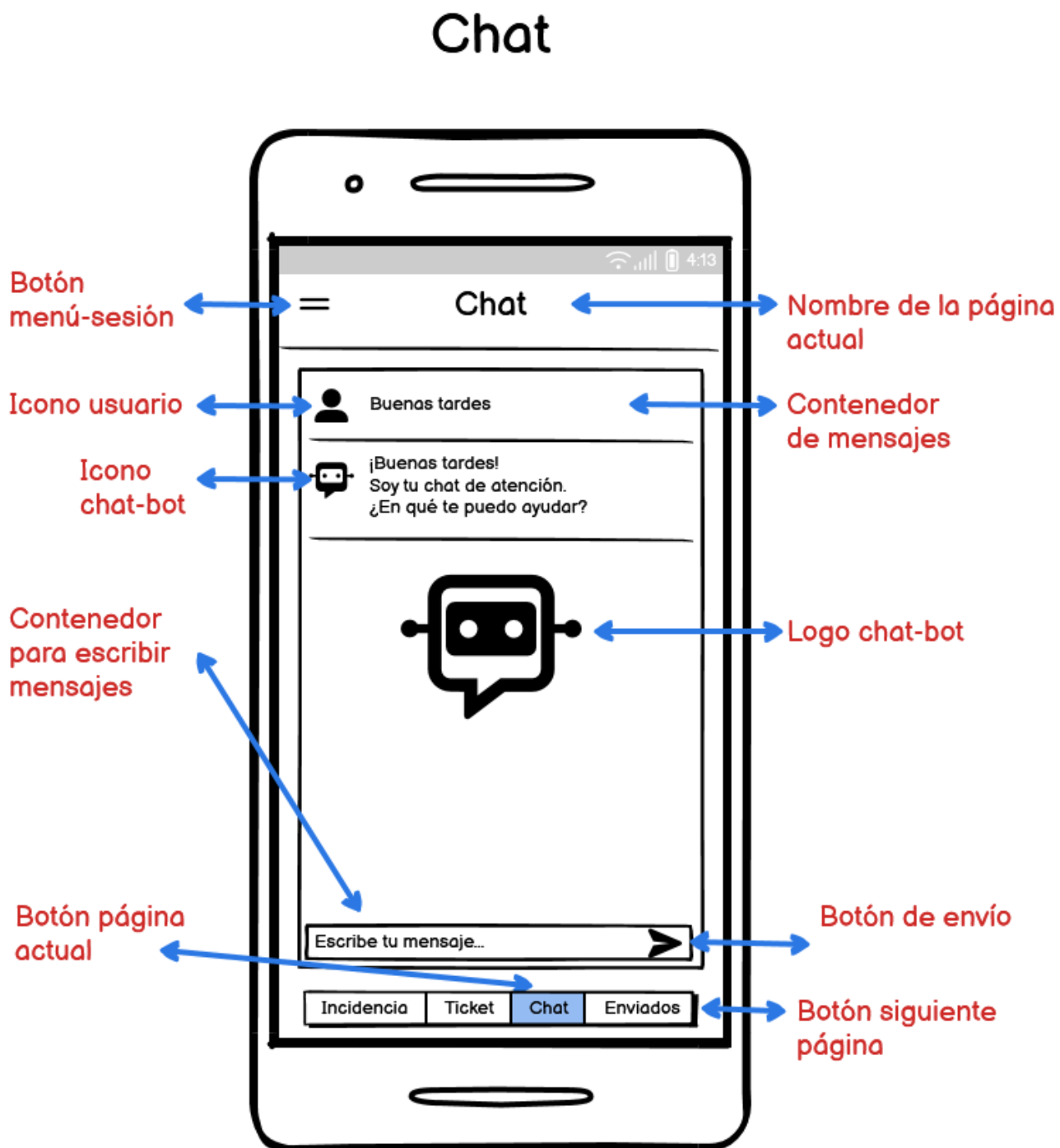


Ilustración 11 Chat

4.1.10 Menú – Sesión - Chat

Menú-sesión-Chat

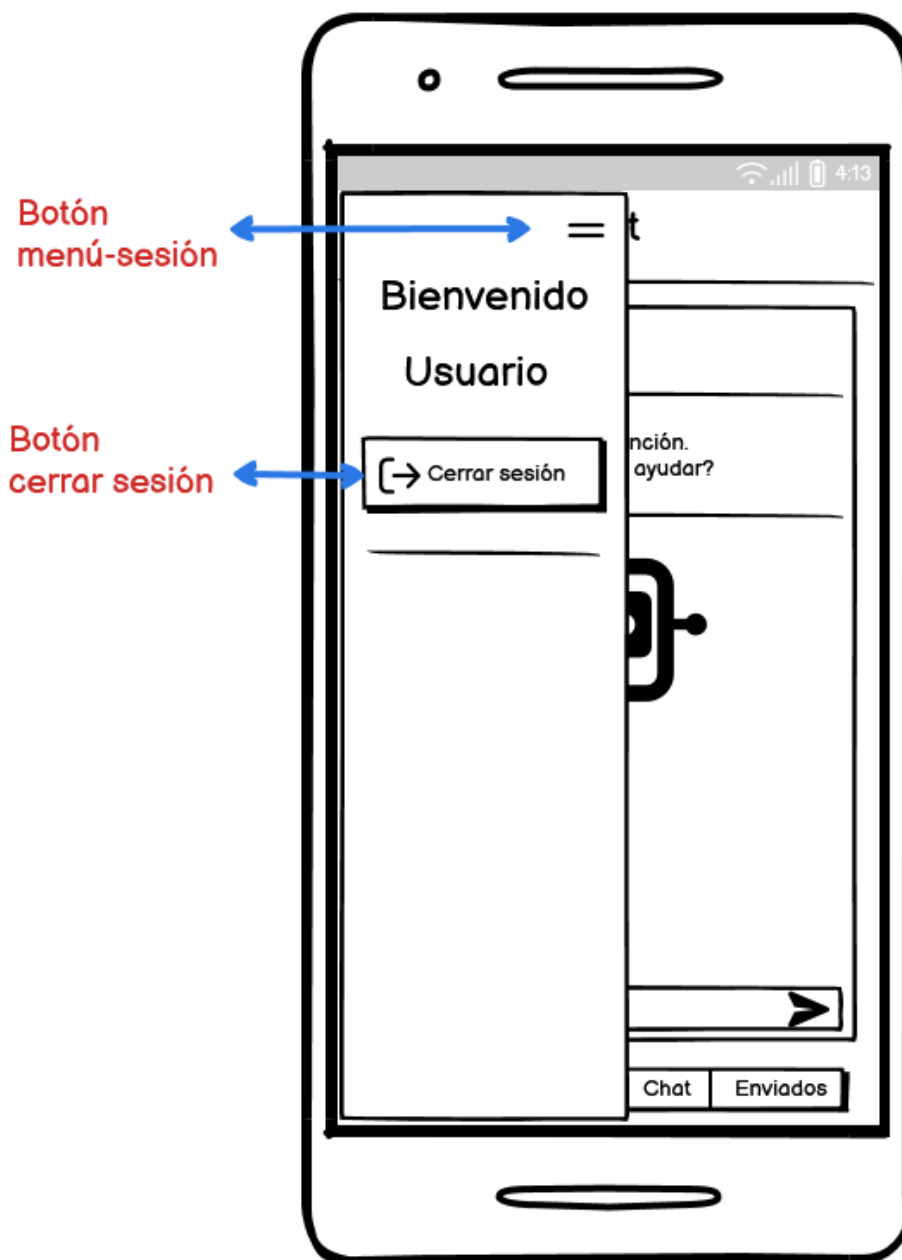


Ilustración 12 Menú – Sesión - Chat

4.1.11 Tickets enviados

Tickets enviados

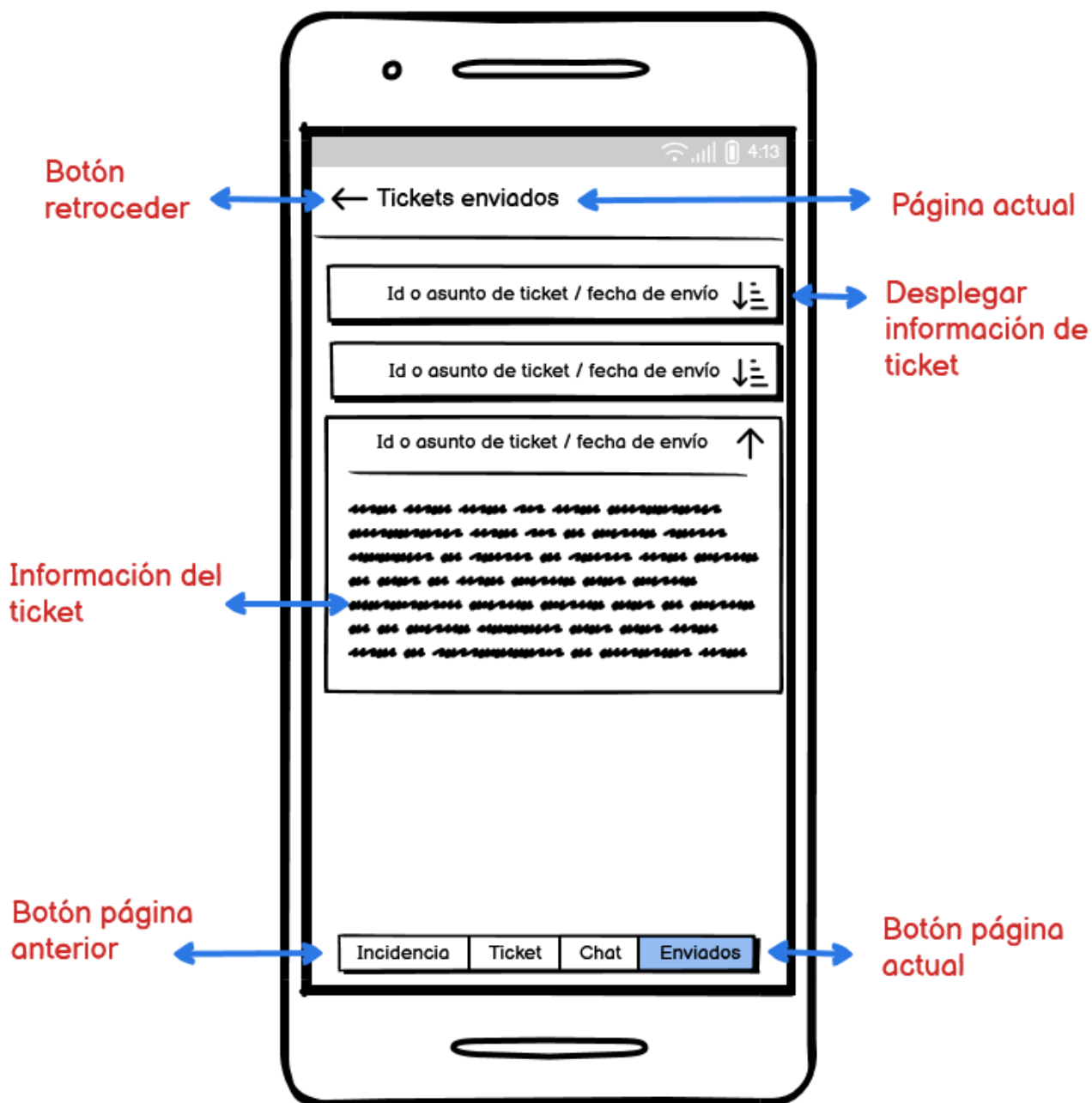


Ilustración 13 Tickets enviados

4.1.12 Menú-tickets-enviados

Menú-sesión-Tickets enviados

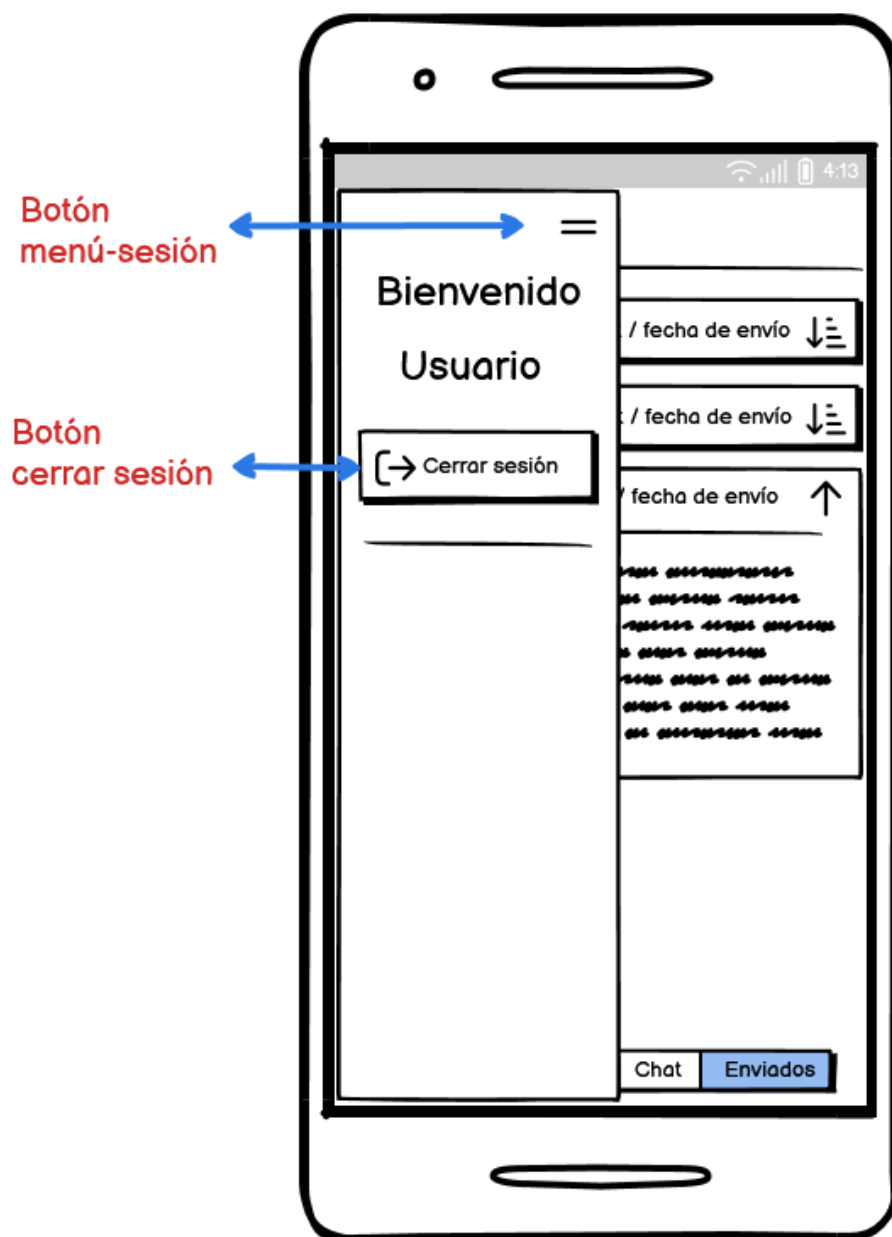


Ilustración 14 Menú-tickets-enviados

4.2 Codificación

Se definen y se crean los componentes a partir de la maquetaación.

4.2.1 Codificación lógica de rutas de la aplicación móvil (Back-End).

```
1 import { NgModule } from '@angular/core';
2 import { PreloadAllModules, RouterModule, Routes } from '@angular/router';
3 import { AuthGuard, redirectLoggedInTo, redirectUnauthorizedTo } from '@angular/fire/auth-guard';
4
5 const redirectUnauthorizedToLogin = () => redirectUnauthorizedTo(['login']);
6 const redirectLoggedInToTabs = () => redirectLoggedInTo(['tabs']);
7
8 const routes: Routes = [
9   {
10    path: '',
11    redirectTo: 'login',
12    pathMatch: 'full'
13  },
14  {
15    path: 'tabs',
16    canActivate: [AuthGuard],
17    data: { AuthGuardPipe : redirectUnauthorizedToLogin },
18    loadChildren: () => import('./tabs/tabs.module').then( m => m.TabsPageModule)
19  },
20  {
21    path: 'login',
22    loadChildren: () => import('./login/login.module').then( m => m.LoginPageModule)
23  },
24  {
25    path: 'register',
26    loadChildren: () => import('./register/register.module').then( m => m.RegisterPageModule)
27  },
28  {
29    path: 'home',
30    canActivate: [AuthGuard],
31    data: { AuthGuardPipe : redirectUnauthorizedToLogin },
32    loadChildren: () => import('./home/home.module').then( m => m.HomePageModule)
33  },
34  {
35    path: 'ticket',
36    canActivate: [AuthGuard],
37    data: { AuthGuardPipe : redirectUnauthorizedToLogin },
38    loadChildren: () => import('./ticket/ticket.module').then( m => m.TicketPageModule)
39  },
40  {
41    path: 'chat',
42    canActivate: [AuthGuard],
43    data: { AuthGuardPipe : redirectUnauthorizedToLogin },
44    loadChildren: () => import('./chat/chat.module').then( m => m.ChatPageModule)
45  },
46  {
47    path: 'formato',
48    canActivate: [AuthGuard],
49    data: { AuthGuardPipe : redirectUnauthorizedToLogin },
50    loadChildren: () => import('./formato/formato.module').then( m => m.FormatoPageModule)
51  },
52 ];
```

Ilustración 15 Codificación lógica de rutas de la aplicación móvil (Back-End).

4.2.2 Importación de componentes en la aplicación móvil (Back-End).

```
1  import { NgModule } from '@angular/core';
2  import { BrowserModule } from '@angular/platform-browser';
3  import { RouteReuseStrategy } from '@angular/router';
4
5  import { IonicModule, IonicRouteStrategy } from '@ionic/angular';
6  import { AuthGuardModule } from '@angular/fire/auth-guard';
7
8  import { AppComponent } from '../app.component';
9  import { AppRoutingModuleModule } from '../app-routing.module';
10 import { initializeApp, provideFirebaseApp } from '@angular/fire/app';
11 import { environment } from '../environments/environment';
12 import { provideAuth, getAuth } from '@angular/fire/auth';
13 import { provideFirestore, getFirestore } from '@angular/fire/firestore';
14
15 import { EmailComposer } from '@awesome-cordova-plugins/email-composer/ngx';
16
17 import { HttpClientModule } from '@angular/common/http'
18
19 @NgModule({
20   declarations:
21     [
22       AppComponent
23     ],
```

```
25   imports:
26     [
27       BrowserModule,
28       IonicModule.forRoot({ mode: 'md' }),
29       AppRoutingModuleModule,
30       provideFirebaseApp(() => initializeApp(environment.firebase)),
31       provideAuth(() => getAuth()),
32       provideFirestore(() => getFirestore()),
33       AuthGuardModule,
34       HttpClientModule
35     ],
36
37   providers:
38     [
39       EmailComposer,
40       {
41         provide: RouteReuseStrategy,
42         useClass: IonicRouteStrategy
43       }],
44   bootstrap:
45     [
46       AppComponent
47     ],
48 })
49 export class AppModule { }
```

Ilustración 16 Importación de componentes en la aplicación móvil (Back-End).

4.2.3 Codificación lógica de servicios en la aplicación móvil (Back-End).

```
import { Injectable } from '@angular/core';
import { Auth,
  signInWithEmailAndPassword,
  createUserWithEmailAndPassword,
  signOut,
  authState } from '@angular/fire/auth';

@Injectable({
  providedIn: 'root'
})
export class AuthService {

  authState$ = authState(this.afAuth);

  constructor(
    private afAuth: Auth
  ) { }

  async register(email: any, password: any) {
    const user = await createUserWithEmailAndPassword(this.afAuth, email, password);
    return await signInWithEmailAndPassword(this.afAuth, email, password);
  }

  login(email: any, password: any) {
    return signInWithEmailAndPassword(this.afAuth, email, password);
  }

  logout(){
    return signOut(this.afAuth)
  }
}
```

```
1  import { HttpClient } from '@angular/common/http';
2  import { Injectable } from '@angular/core';
3  import { environment } from 'src/environments/environment';
4
5  @Injectable({
6    providedIn: 'root'
7  })
8  export class OpenaiService {
9
10     constructor(private http: HttpClient) {}
11
12     sendQuestion(prompt: string){
13       return this.http.post(environment.baseUrl, {prompt});
14     }
15   }
16
```

Ilustración 17 Codificación lógica de servicios en la aplicación móvil (Back-End).

4.2.4 Codificación lógica del componente Inicio de sesión (Back-End).

```
1  import { Component, OnInit } from '@angular/core';
2  import { FormBuilder, Validators } from '@angular/forms';
3
4  import { AuthService } from '../services/auth.service';
5  import { Router } from '@angular/router';
6
7  @Component({
8    selector: 'app-login',
9    templateUrl: './login.page.html',
10   styleUrls: ['./login.page.scss'],
11 })
12 export class LoginPage implements OnInit {
13
14   form = this.FormBuilder.group({
15     email: ['', [Validators.email, Validators.required]],
16     password: ['', [Validators.required]],
17   });
18
19   constructor(
20     private FormBuilder: FormBuilder,
21     private auth: AuthService,
22     private router: Router
23   ) { }
24
25   ngOnInit() {
26   }
27
28
29   login() {
30     if (this.form.valid) {
31       const { email, password } = this.form.getRawValue();
32       this.auth.login( email, password )
33         .then(() => {
34           this.router.navigate(['/tabs']);
35         })
36         .catch(error => {
37           console.error(error);
38         });
39     } else {
40       this.form.markAllAsTouched();
41     }
42   }
43 }
```

Ilustración 18 Codificación lógica del componente Inicio de sesión (Back-End).

4.2.5 Codificación de la interfaz (Front-End).

```
1  <ion-header [translucent]="true">
2    <ion-toolbar color="primary">
3      <ion-title class="ion-text-center">
4        <strong>Help Desk</strong>
5      </ion-title>
6    </ion-toolbar>
7  </ion-header>
8
9  <ion-content class="ion-padding">
10
11    <div class="logo">
12      <ion-avatar style="margin: 0 auto;">
13        <ion-img src="/assets/logo.png"/>
14      </ion-avatar>
15      <h4>Iniciar sesión</h4>
16    </div>
17
18    <form [formGroup]="form" validate (ngSubmit)="login()">
19      <ion-list>
20        <ion-item>
21          <ion-label position="floating">Correo</ion-label>
22          <ion-input type="email" errorText="Invalid email" formControlName="email"></ion-input>
23        </ion-item>
24
25        <ion-item>
26          <ion-label position="floating">Contraseña</ion-label>
27          <ion-input type="password" formControlName="password"></ion-input>
28        </ion-item>
29      </ion-list>
30    </form>
31
32    <ion-row>
33      <ion-col>
34        <ion-button type="submit" expand="block">
35          <ion-icon slot="start" name="log-in-outline"></ion-icon>
36          Ingresar
37        </ion-button>
38      </ion-col>
39      <ion-col>
40        <ion-button routerLink="/register" type="button" expand="block">
41          <ion-icon slot="start" name="person-add-outline"></ion-icon>
42          Crear cuenta
43        </ion-button>
44      </ion-col>
45    </ion-row>
46  </ion-content>
```

Ilustración 19 Codificación de la interfaz (Front-End).

4.2.6 Codificación lógica del componente Crear cuenta (Back-End).

```
1  import { Component, OnInit } from '@angular/core';
2  import { FormBuilder, Validators } from '@angular/forms';
3
4  import { AuthService } from '../services/auth.service'
5  import { Router } from '@angular/router';
6
7  @Component({
8    selector: 'app-register',
9    templateUrl: './register.page.html',
10   styleUrls: ['./register.page.scss'],
11 })
12 export class RegisterPage implements OnInit {
13
14   form = this.FormBuilder.group({
15     email: ['', [Validators.email, Validators.required]],
16     password: ['', [Validators.required]],
17     confirmPassword: ['', [Validators.required]],
18   });
19
20   constructor(
21     private FormBuilder: FormBuilder,
22     private auth: AuthService,
23     private router: Router
24   ) { }
25
26   ngOnInit() {
27   }
28
29
30   register() {
31     if (this.form.valid) {
32       const { email, password } = this.form.getRawValue();
33       this.auth.register( email, password )
34         .then(() => {
35           this.router.navigate(['/tabs']);
36         })
37         .catch(error => {
38           console.error(error);
39         });
40     } else {
41       this.form.markAllAsTouched();
42     }
43   }
44 }
45
```

Ilustración 20 Codificación lógica del componente Crear cuenta (Back-End).

4.2.7 Codificación de la interfaz (Front-End).

```
1  <ion-header [translucent]="true">
2    <ion-toolbar color="primary">
3      <ion-buttons slot="start">
4        <ion-back-button></ion-back-button>
5      </ion-buttons>
6      <ion-title>
7        <strong>Help Desk</strong>
8      </ion-title>
9    </ion-toolbar>
10 </ion-header>
11
12 <ion-content class="ion-padding">
13
14   <div class="logo">
15     <ion-avatar style="margin: 0 auto;">
16       <ion-img src="/assets/logo.png"/>
17     </ion-avatar >
18     <h4>Crear cuenta</h4>
19   </div>
20
21   <form [formGroup]="form" novalidate (ngSubmit)="register()">
22     <ion-list>
23       <ion-item>
24         <ion-label position="floating">Correo</ion-label>
25         <ion-input type="email" formControlName="email"></ion-input>
26       </ion-item>
27       <ion-item>
28         <ion-label position="floating">Contraseña</ion-label>
29         <ion-input type="password" formControlName="password"></ion-input>
30       </ion-item>
31
32       <ion-item>
33         <ion-label position="floating">Confirma tu contraseña</ion-label>
34         <ion-input type="password" formControlName="confirmPassword"></ion-input>
35       </ion-item>
36     </ion-list>
37
38     <ion-row>
39       <ion-col>
40         <ion-button routerLink="/login" routerDirection="back" type="button" expand="block">
41           <ion-icon slot="start" name="arrow-back-outline"></ion-icon>
42           Regresar
43         </ion-button>
44       </ion-col>
45       <ion-col>
46         <ion-button type="submit" expand="block">
47           <ion-icon slot="start" name="log-in-outline"></ion-icon>
48           Registrarme
49         </ion-button>
50       </ion-col>
51     </ion-row>
52   </form>
53 </ion-content>
```

Ilustración 21 Codificación de la interfaz (Front-End).

4.2.8 Codificación lógica del componente Incidencia (Back-End).

```
import { Component, OnInit } from '@angular/core';

import { EmailComposerOptions } from '@awesome-cordova-plugins/email-composer/ngx';
import { EmailComposer } from '@awesome-cordova-plugins/email-composer/ngx';
import { Camera, CameraResultType, CameraSource } from '@capacitor/camera';

@Component({
  selector: 'app-home',
  templateUrl: './home.page.html',
  styleUrls: ['./home.page.scss'],
})
export class HomePage {

  hasAccount = false;
  currentImage?: null | string;
  imageData?: null | string;

  constructor(private emailComposer: EmailComposer) {}

  async captureImage() {
    const image = await Camera.getPhoto({
      quality: 90,
      allowEditing: false,
      resultType: CameraResultType.Base64,
      source: CameraSource.Camera, // Camera, Photos or Prompt!
    });
    this.imageData = image.base64String;
    this.currentImage = `data:image/jpeg;base64,${image.base64String}`;
  }
}
```

```
33   async openEmail() {
34     const email: EmailComposerOptions = {
35       to: ' ',
36       cc: ' ',
37       attachments: [`base64:image.jpg://${this.imageData}`],
38       subject: 'Incidencia ',
39       body: ' Hola!... esto es una prueba. ',
40     };
41
42     await this.emailComposer.open(email);
43   }
44
45 }
46
```

Ilustración 22 Codificación lógica del componente Incidencia (Back-End).

4.2.9 Codificación de la interfaz (Front-End).

```
1  <ion-header [translucent]="true">
2    <ion-toolbar color="primary">
3      <ion-buttons slot="start">
4        <ion-menu-button></ion-menu-button>
5      </ion-buttons>
6      <ion-title>
7        <strong>Incidencia</strong>
8      </ion-title>
9    </ion-toolbar>
10 </ion-header>
11
12 <ion-content class="ion-padding">
13
14   <ion-button fill="clear" expand="block" (click)="openEmail()">
15     Redactar incidencia
16     <ion-icon slot="start" name="create"></ion-icon>
17   </ion-button>
18
19   <ion-button fill="clear" expand="block" (click)="captureImage()">
20     Capturar foto
21     <ion-icon slot="start" name="camera"></ion-icon>
22   </ion-button>
23
24   <ion-card>
25
26     <ion-card-header>
27       "La foto aparecerá aquí, posteriormente genera tu ticket"
28     </ion-card-header>
29
30     <ion-card-content>
31       <img [src]="currentImage" *ngIf="currentImage" />
32     </ion-card-content>
33
34   </ion-card>
35
36 </ion-content>
37
38
```

Ilustración 23 Codificación de la interfaz (Front-End).

4.2.10 Codificación lógica del componente Chat (Back-End).

```
1 import { Component, ViewChild } from '@angular/core';
2 import { FormControl, FormGroup, Validators } from '@angular/forms';
3 import { Message } from '../models/message.model';
4 import { OpenaiService } from '../services/openai.service';
5 import { IonContent } from '@ionic/angular';
6 import { CustomValidators } from '../utils/custom-validators';
7
8 @Component({
9   selector: 'app-chat',
10  templateUrl: './chat.page.html',
11  styleUrls: ['./chat.page.scss'],
12 })
13 export class ChatPage {
14
15   @ViewChild(IonContent, {static: false}) content!: IonContent
16
17   messages: Message[] = [];
18
19   form = new FormGroup({
20     prompt: new FormControl('', [Validators.required, CustomValidators.nowhiteSpace])
21   })
22
23   loading: boolean = false;
24
25   constructor(
26     private openAi: OpenaiService
27   ) { }
28
29   submit() {
30     if(this.form.valid){
31
```

```
34     //Mensaje del usuario
35     let userMsg: Message = { sender: 'me', content: prompt }
36     this.messages.push(userMsg);
37
38     //Mensaje del usuario
39     let botMsg: Message = { sender: 'bot', content: '' }
40     this.messages.push(botMsg);
41
42     this.scrollToBottom();
43     this.form.reset();
44     this.form.disable();
45
46     this.loading = true;
47
48     this.openAi.sendQuestion(prompt).subscribe({
49       next: (res: any) => {
50         this.loading = false;
51         this.typeText(res.bot)
52         this.form.enable();
53       }, error: (error: any) => {
54         console.log(error);
55       }
56     })
57   }
58 }
59
60 typeText(text: string) {
61
62   let textIndex = 0;
```

Ilustración 24 Codificación lógica del componente Chat (Back-End).

4.2.11 Codificación de la interfaz Chat (Front-End).

```
1 <ion-header [translucent]="true">
2   <ion-toolbar color="primary">
3     <ion-buttons slot="start">
4       <ion-menu-button></ion-menu-button>
5     </ion-buttons>
6     <ion-title>
7       <strong>Chat</strong>
8     </ion-title>
9   </ion-toolbar>
10 </ion-header>
11
12 <ion-content scrollEvents="true">
13
14   <div *ngIf="!messages.length" class="empty">
15     <ion-icon src="assets/bot.svg"></ion-icon>
16     <h4>Hola, soy tu chat de atención.</h4>
17   </div>
18
19   <ion-item lines="none" [ngClass]="{'bot' : m.sender! == 'me'}" class="chat-item" *ngFor="let m of messages;
20     <ion-badge slot="start" [color]="m.sender == 'me' ? 'success' : 'primary'" mode="ios">
21     <ion-icon class="chat-icon" [src]="m.sender == 'me' ? 'assets/user.svg' : 'assets/bot.svg'"></ion-icon>
22   </ion-badge>
23   <ion-label class="ion-text-wrap">
24     <p class="w-space">{{m.content}}</p>
25     <ion-spinner *ngIf = "loading && (i+1 == messages.length)" name="dots" color="tertiary">
26     </ion-spinner>
27   </ion-label>
28 </ion-item>
29
30 </ion-content>
31
32 <ion-footer class="ion-padding">
33
34   <form [formGroup]="form">
35
36     <ion-item lines="none" class="text-box-item">
37
38       <ion-textarea
39         formControlName="prompt"
40         auto-grow="true"
41         placeholder="Escribe tu mensaje...">
42       </ion-textarea>
43
44       <ion-buttons slot="end">
45         <ion-button (click)="submit()" [disabled]="form.invalid">
46           <ion-icon name="send" class="submit-icon" color="primary"></ion-icon>
47         </ion-button>
48       </ion-buttons>
49     </ion-item>
50
51   </form>
52 </ion-footer>
53
54 </ion-footer>
```

Ilustración 25 Codificación de la interfaz Chat (Front-End).

4.2.12 Codificación de la interfaz Generar tickets (Front-End).

```
1  <ion-header [translucent]="true">
2    <ion-toolbar color="primary">
3      <ion-buttons slot="start">
4        <ion-menu-button></ion-menu-button>
5      </ion-buttons>
6      <ion-title>
7        <strong>Generar ticket</strong>
8      </ion-title>
9    </ion-toolbar>
10 </ion-header>
11
12
13 <ion-content>
14   <ion-card>
15     <ion-list>
16       <ion-item>
17         <ion-select aria-label="Tipo" interface="action-sheet" placeholder="Tipo">
18           <ion-select-option value="solicitud">Solicitud</ion-select-option>
19           <ion-select-option value="asistencia">Asistencia</ion-select-option>
20           <ion-select-option value="problema">Problema</ion-select-option>
21           <ion-select-option value="servicio">Servicio</ion-select-option>
22         </ion-select>
23       </ion-item>
24
25       <ion-item>
26         <ion-select aria-label="Depart" interface="action-sheet" placeholder="Departamento">
27           <ion-select-option value="administracion">Administracion</ion-select-option>
28           <ion-select-option value="marketing">Marketing</ion-select-option>
29           <ion-select-option value="ventas">Ventas</ion-select-option>
30           <ion-select-option value="finanzas">Finanzas</ion-select-option>
31         </ion-select>
32       </ion-item>
33
34       <ion-item>
35         <ion-toggle [enableOnOffLabels]="true">Prioridad</ion-toggle>
36       </ion-item>
37
38       <ion-item>
39         <ion-input label="Asunto" label-placement="floating" fill="solid"></ion-input>
40       </ion-item>
41
42       <ion-item>
43         <ion-input label="Descripción" label-placement="floating" fill="solid"></ion-input>
44       </ion-item>
45
46       <ion-button type="button" fill="clear" expand="block">
47         Adjuntar
48         <ion-icon slot="end" name="document-attach"></ion-icon>
49       </ion-button>
50
51       <ion-button type="submit" fill="clear" expand="block">
52         Enviar
53         <ion-icon slot="end" name="send"></ion-icon>
54       </ion-button>
55
56     </ion-list>
57   </ion-card>
58 </ion-content>
```

Ilustración 26 Codificación de la interfaz Generar tickets (Front-End).

4.2.13 Codificación de la interfaz Tickets enviados (Front-End).

```
1  <ion-header [translucent]="true">
2    <ion-toolbar color="primary">
3      <ion-buttons slot="start">
4        <ion-menu-button></ion-menu-button>
5      </ion-buttons>
6      <ion-title>
7        <strong>Tickets enviados</strong>
8      </ion-title>
9    </ion-toolbar>
10 </ion-header>
11
12 <ion-content>
13
14   <ion-card>
15
16     <ion-accordion-group>
17
18       <ion-accordion value="first">
19         <ion-item slot="header" color="light">
20           <ion-label>
21             <strong> Solicitud
22           </strong>
23         </ion-label>
24         <ion-datetime-button datetime="datetime"></ion-datetime-button>
25         <ion-modal [keepContentsMounted]="true">
26           <ng-template>
27             <ion-datetime id="datetime"></ion-datetime>
28           </ng-template>
29         </ion-modal>
30       </ion-item>
31
32       <div class="ion-padding" slot="content">Tipo:</div>
33       <div class="ion-padding" slot="content">Departamento:</div>
34       <div class="ion-padding" slot="content">Prioridad:</div>
35       <div class="ion-padding" slot="content">Asunto:</div>
36       <div class="ion-padding" slot="content">Descripción:</div>
37     </ion-accordion>
38   </ion-accordion-group>
39 </ion-card>
40
41 </ion-content>
```

Ilustración 27 Codificación de la interfaz Tickets enviados (Front-End).

Capítulo V: Resultados

5.1 Aplicación finalizada

5.1.1 Inicio de sesión

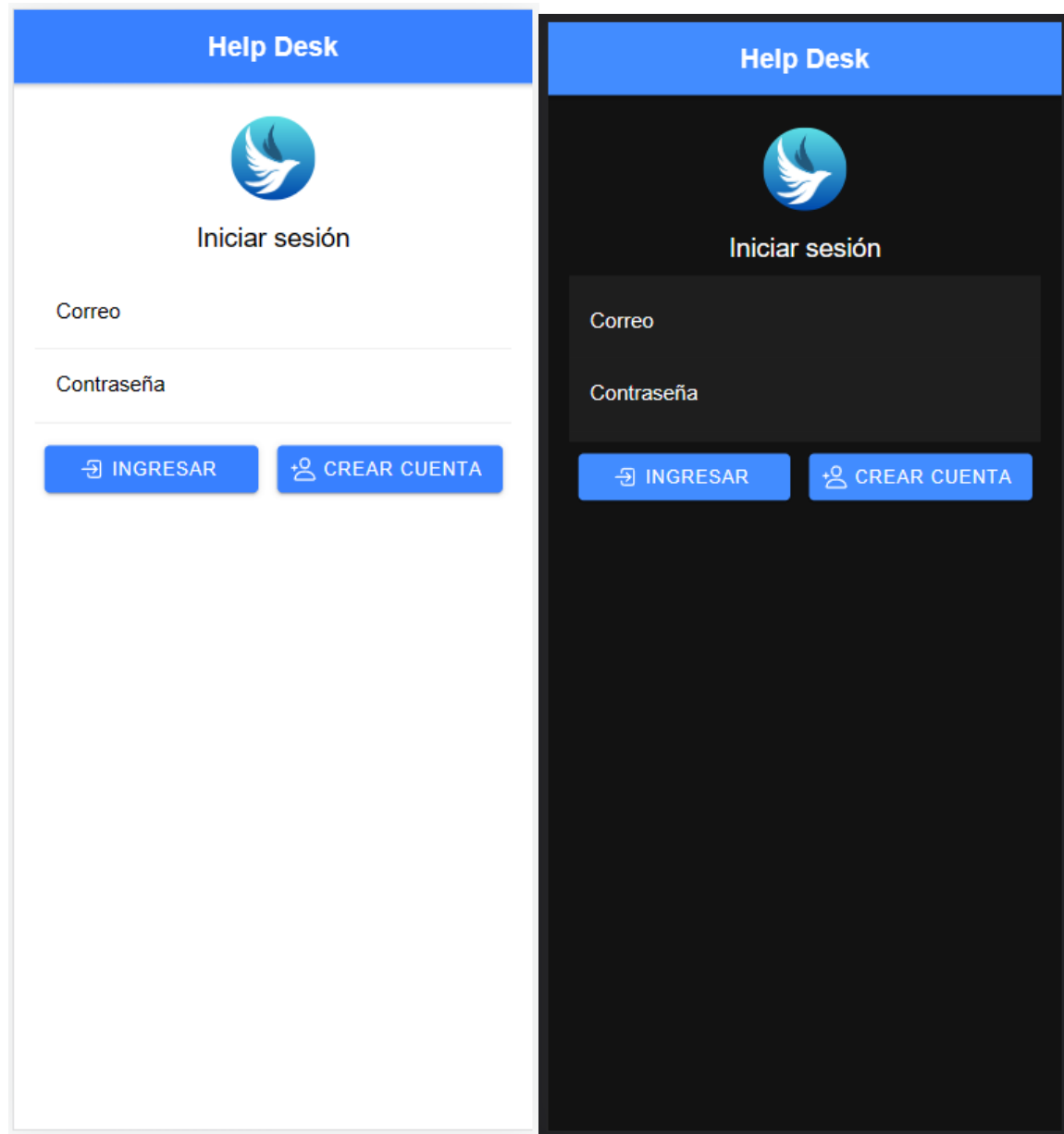


Ilustración 28 Inicio de sesión

5.1.2 Crear cuenta

← Help Desk



Crear cuenta

Correo

Contraseña

Confirma tu contraseña

← REGRESAR → REGISTRARME

Ilustración 29 Crear cuenta

5.1.3 Incidencia

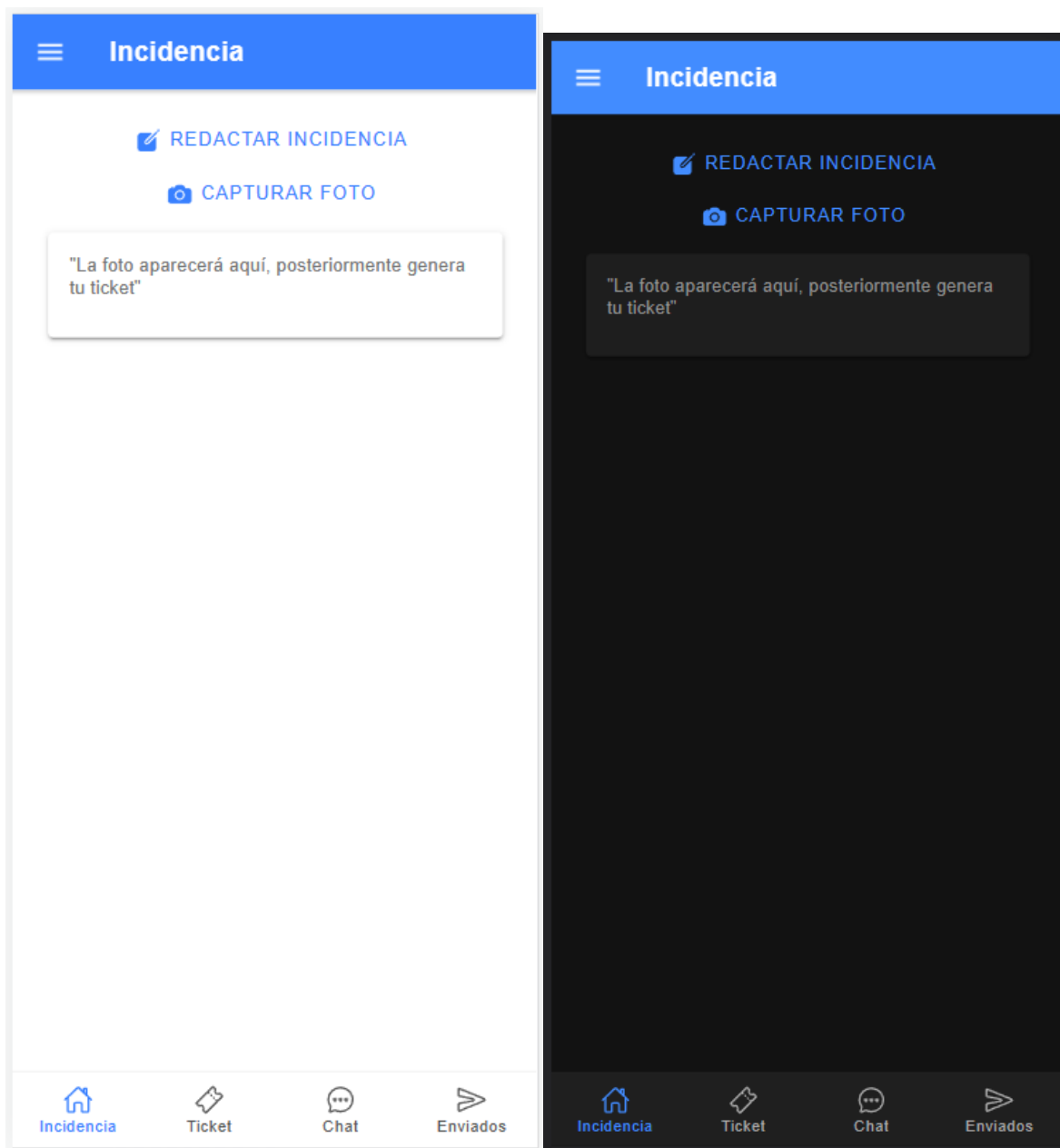


Ilustración 30 Incidencia

5.1.4 Captura de incidencia

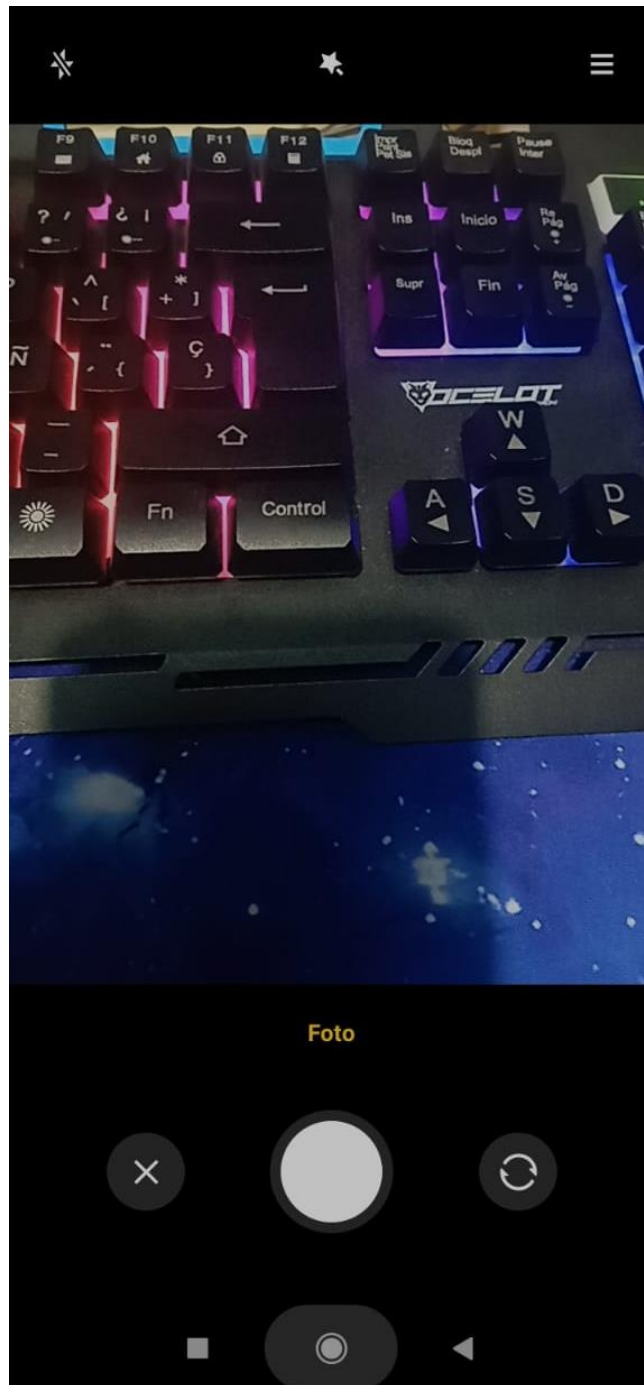


Ilustración 31 Captura de incidencia

5.1.5 Levantamiento de incidencia

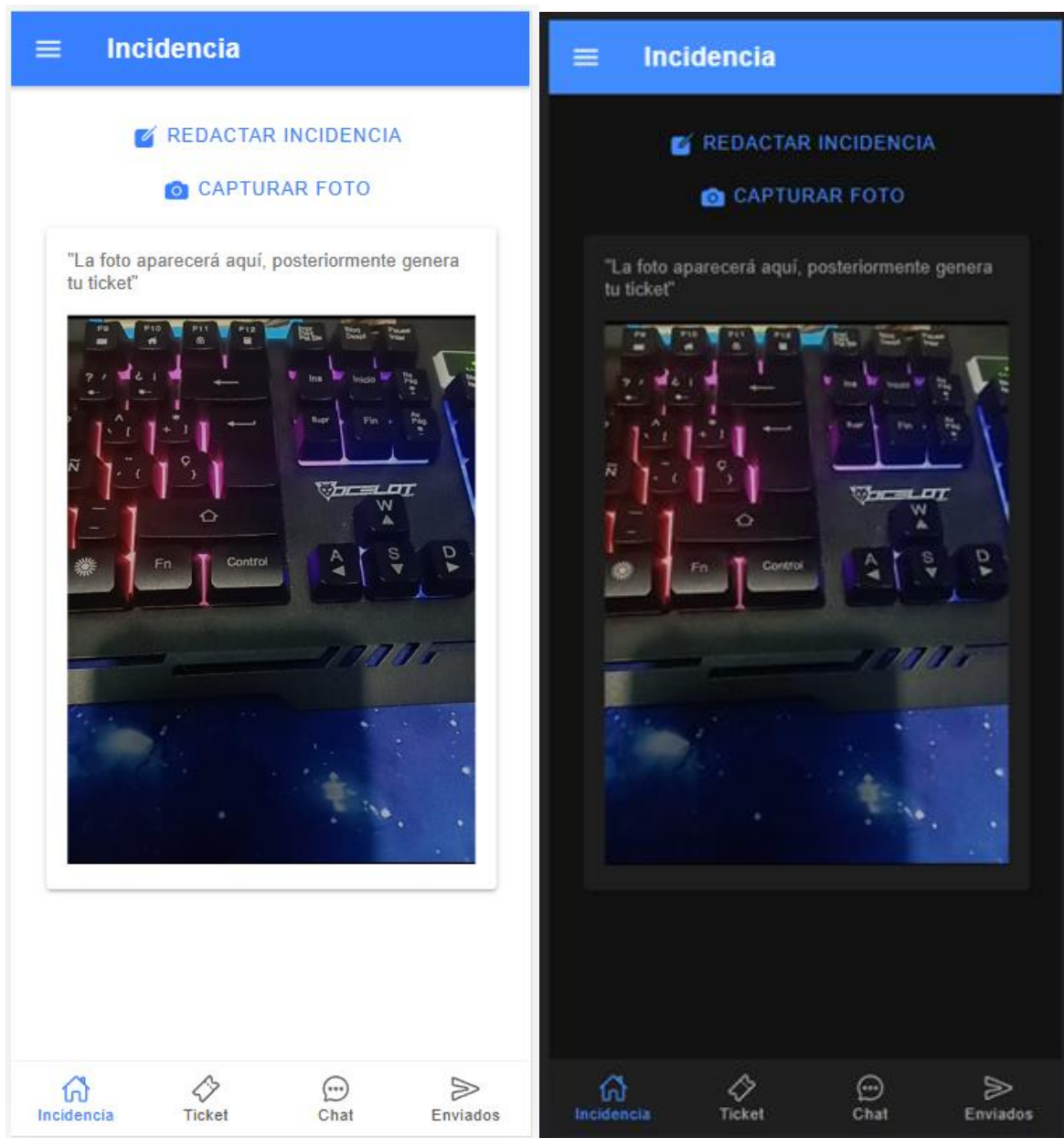


Ilustración 32 Levantamiento de incidencia

5.1.6 Redacción de incidencia

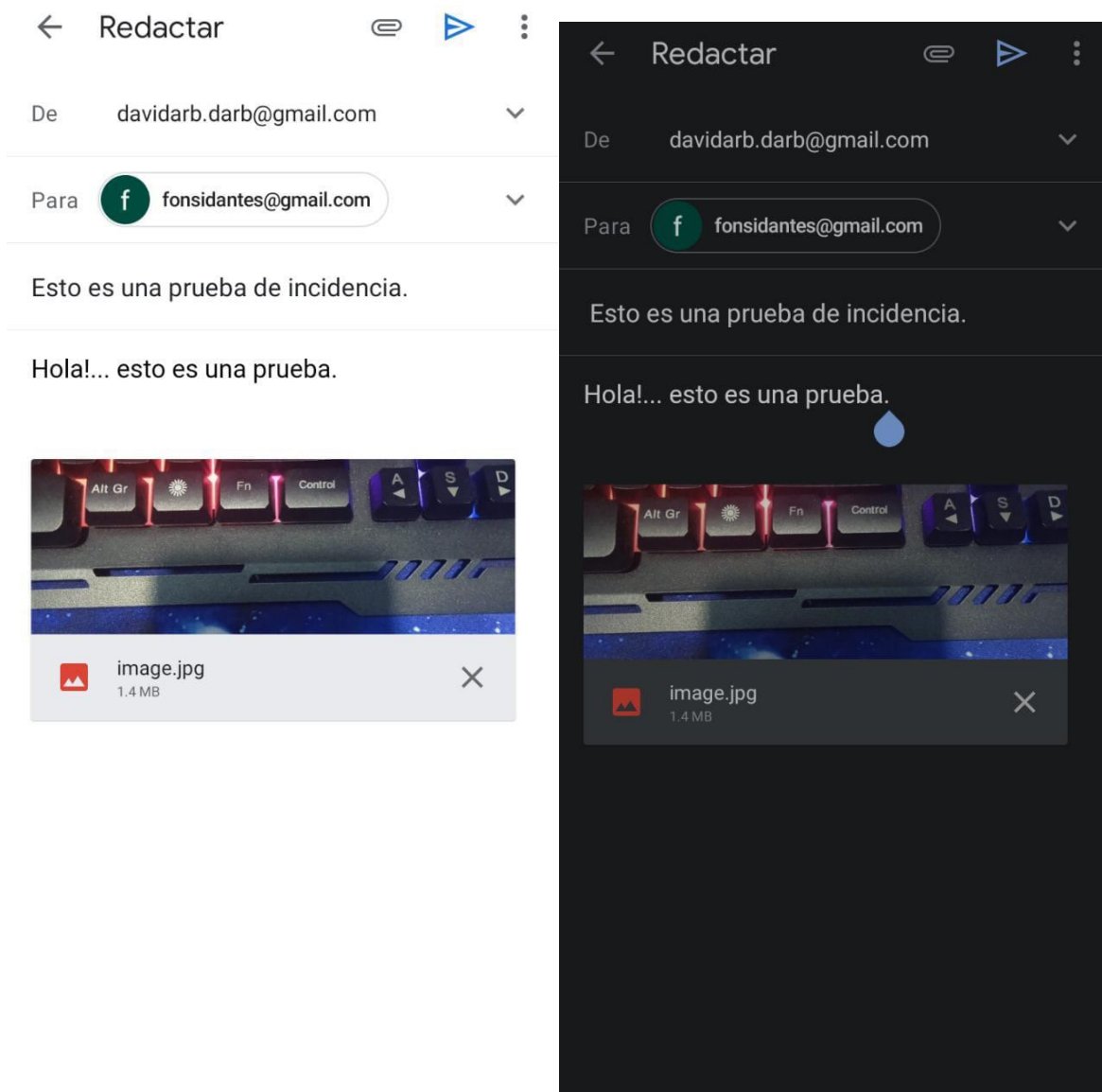


Ilustración 33 Redacción de incidencia

5.1.7 Menú-sesión-incidencia

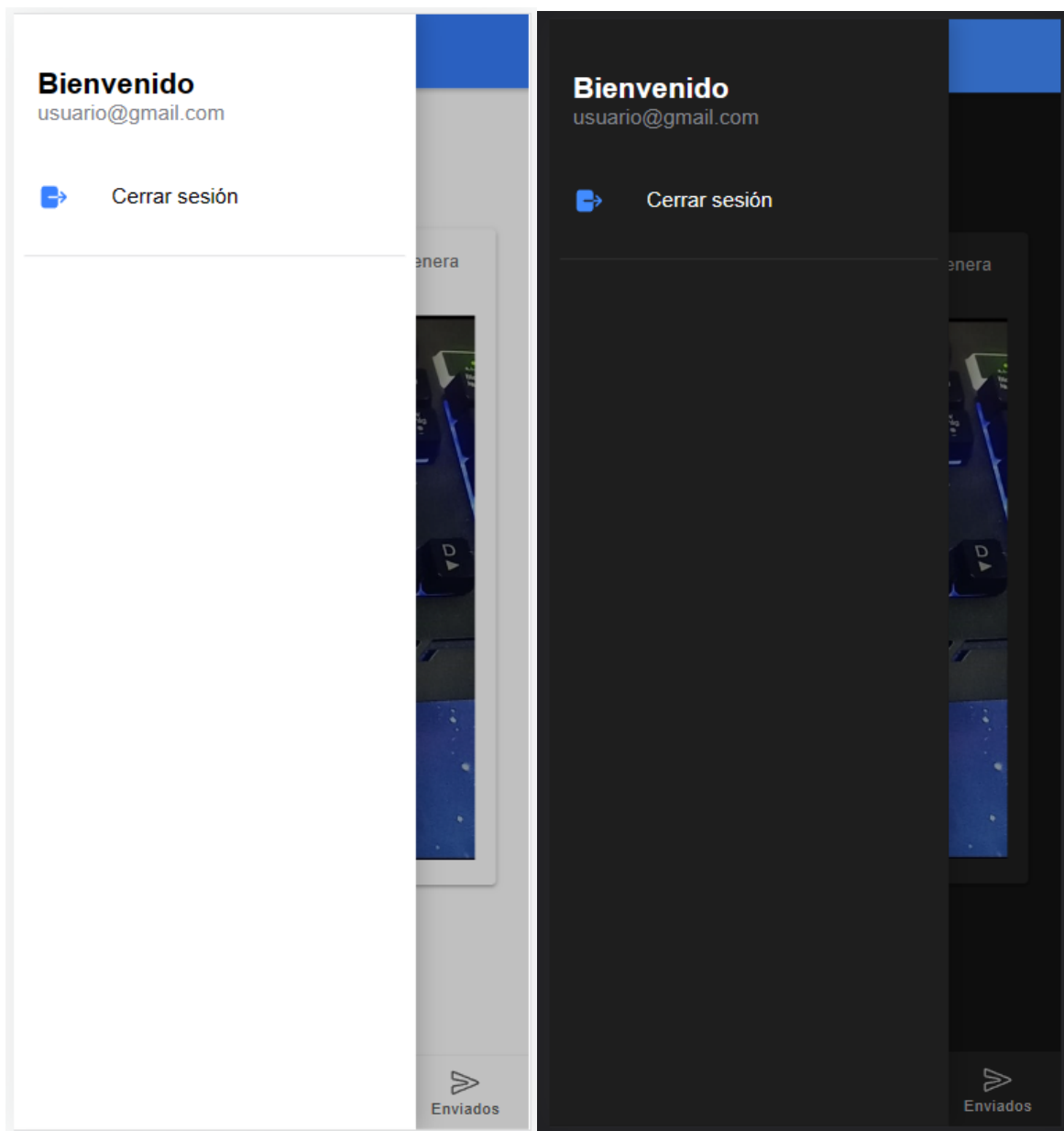


Ilustración 34 Menú-sesión-incidencia

5.1.8 Generar ticket

The image displays two versions of a mobile application interface for generating a ticket, side-by-side. The left version is in light mode, and the right version is in dark mode. Both screens have a blue header bar with a hamburger menu icon and the text "Generar ticket".

The form on the screen includes the following fields:

- Problema**: A dropdown menu.
- Ventas**: A dropdown menu.
- Prioridad**: A text field with a toggle switch turned on (indicated by a blue checkmark).
- Asunto**: A text field containing the text "Base de datos".
- Descripción**: A text field containing the text "La base de datos no responde."

Below the form, there are two buttons: "ADJUNTAR" with a paperclip icon and "ENVIAR" with a right-pointing arrow icon.

At the bottom of the screen is a navigation bar with four icons and labels: "Incidencia" (house icon), "Ticket" (ticket icon), "Chat" (speech bubble icon), and "Enviados" (right-pointing arrow icon).

Ilustración 35 Generar ticket

5.1.9 Menú-sesión-Generar ticket

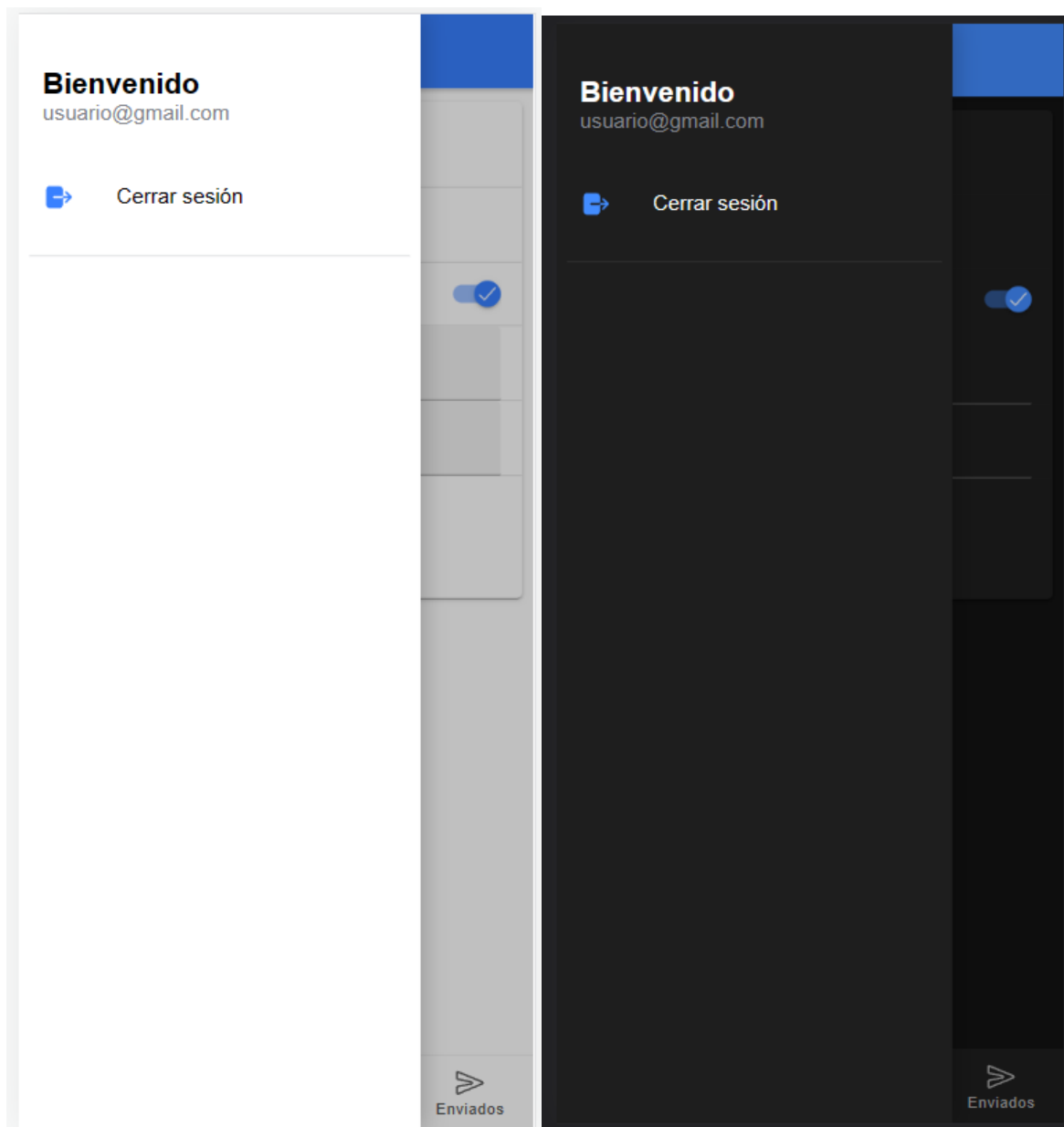


Ilustración 36 Menú-sesión-Generar ticket

5.1.10 Interfaz del Chat

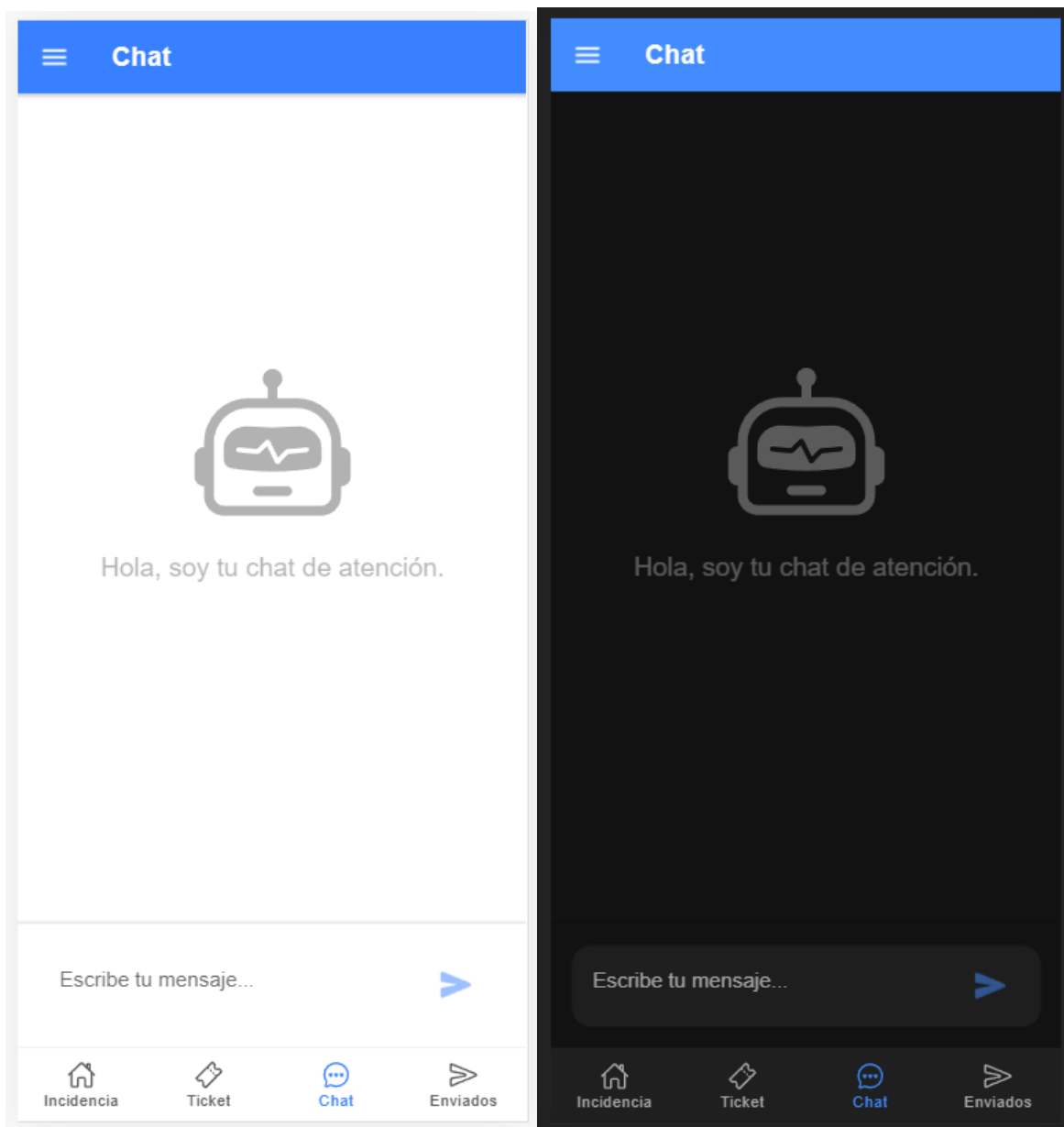


Ilustración 37 Interfaz del Chat

5.1.11 Respuestas con Inteligencia Artificial mediante ChatGPT

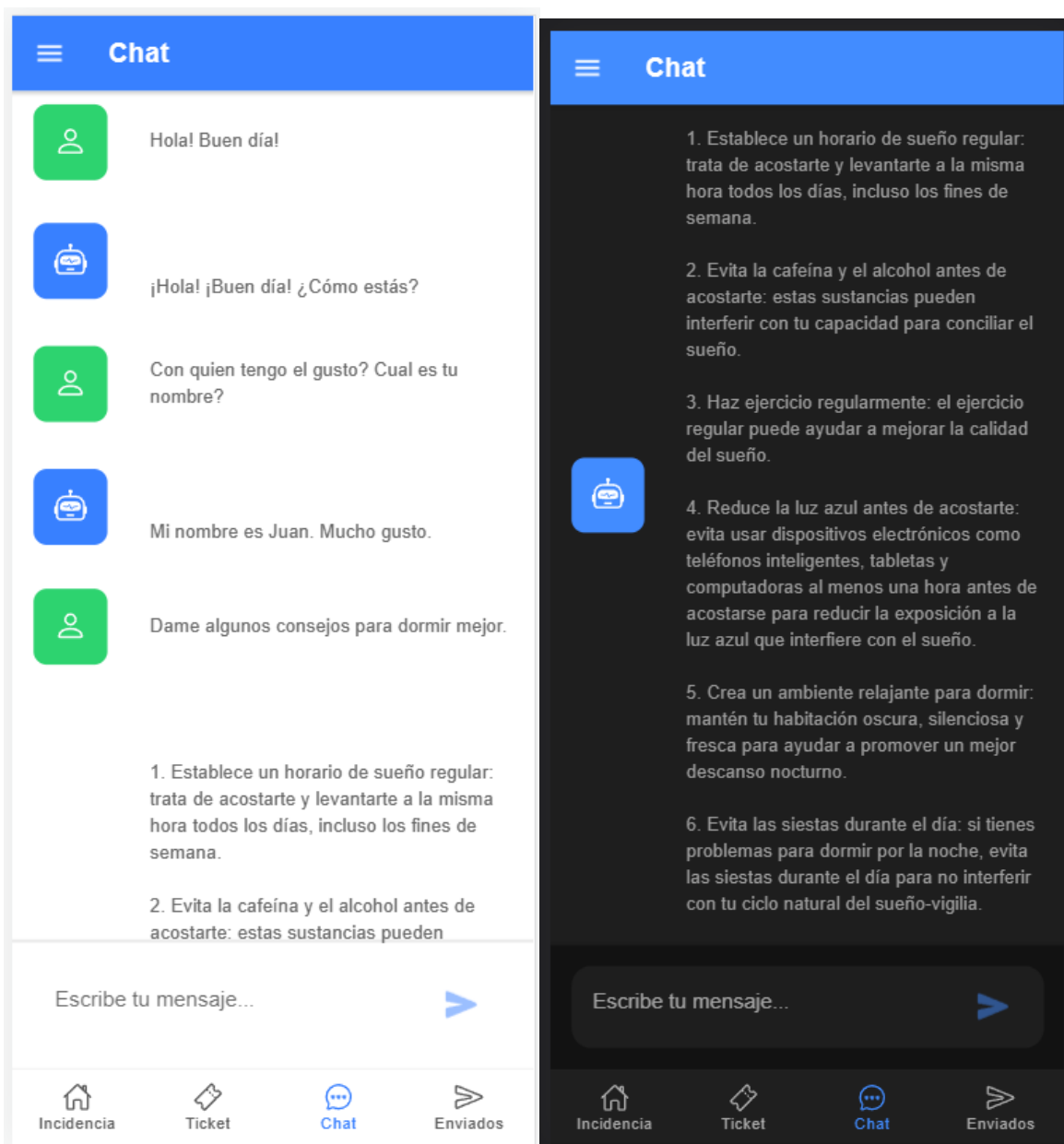


Ilustración 38 Respuestas con Inteligencia Artificial mediante ChatGPT

5.1.12 Menú – Sesión - Chat

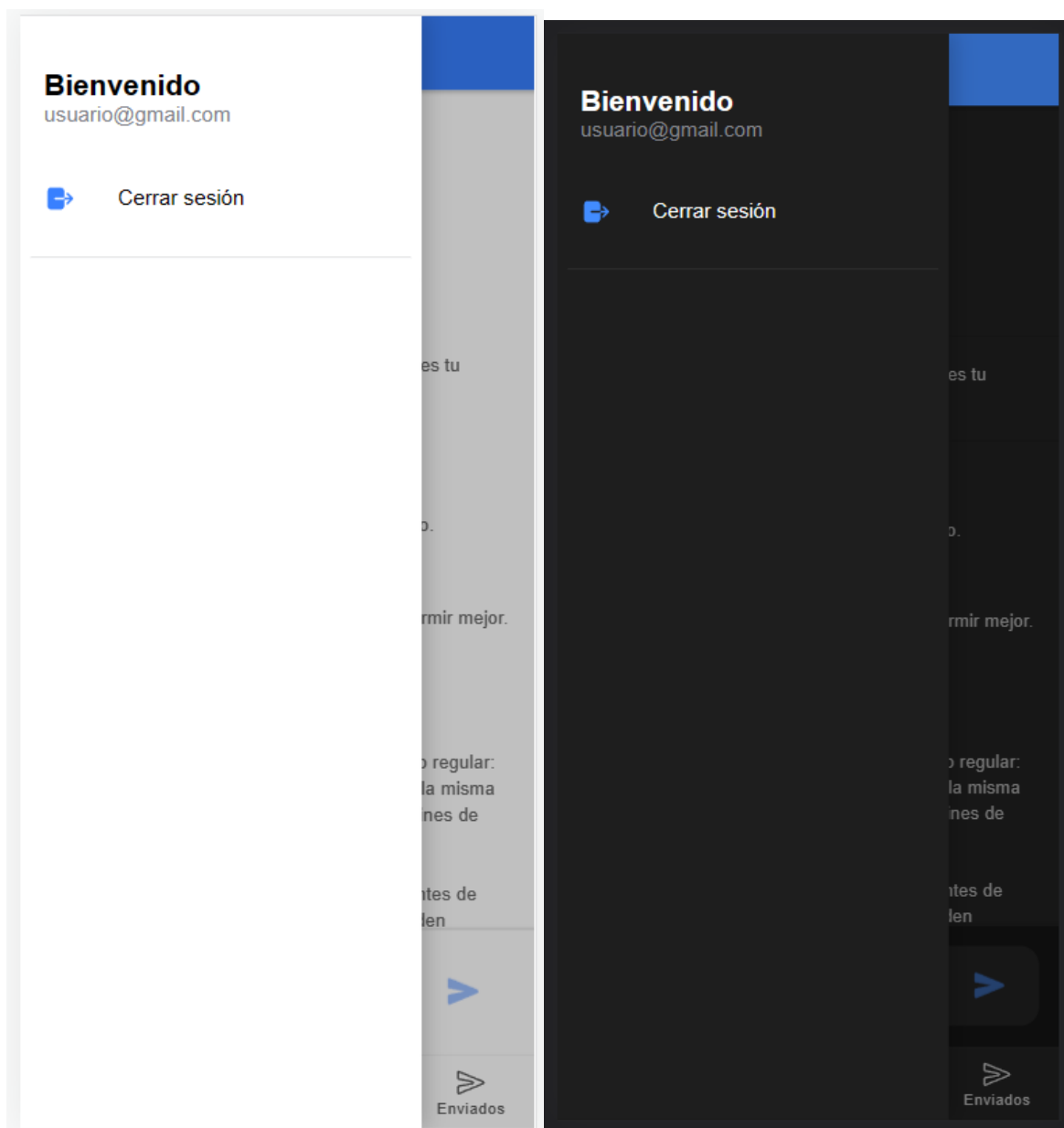


Ilustración 39 Menú – Sesión - Chat

5.1.13 Tickets enviados

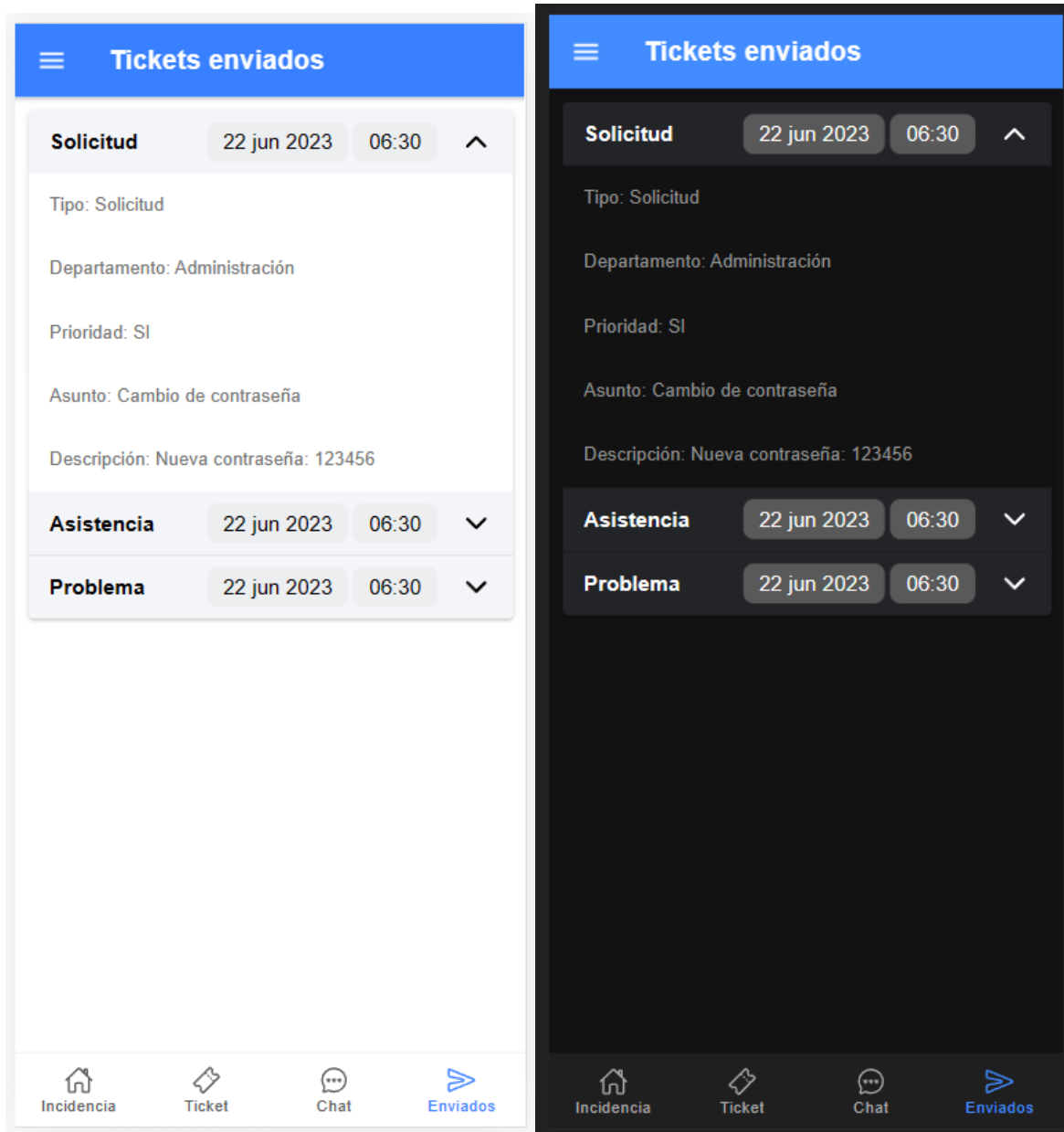


Ilustración 40 Tickets enviados

5.1.14 Menú – Sesión – Tickets enviados

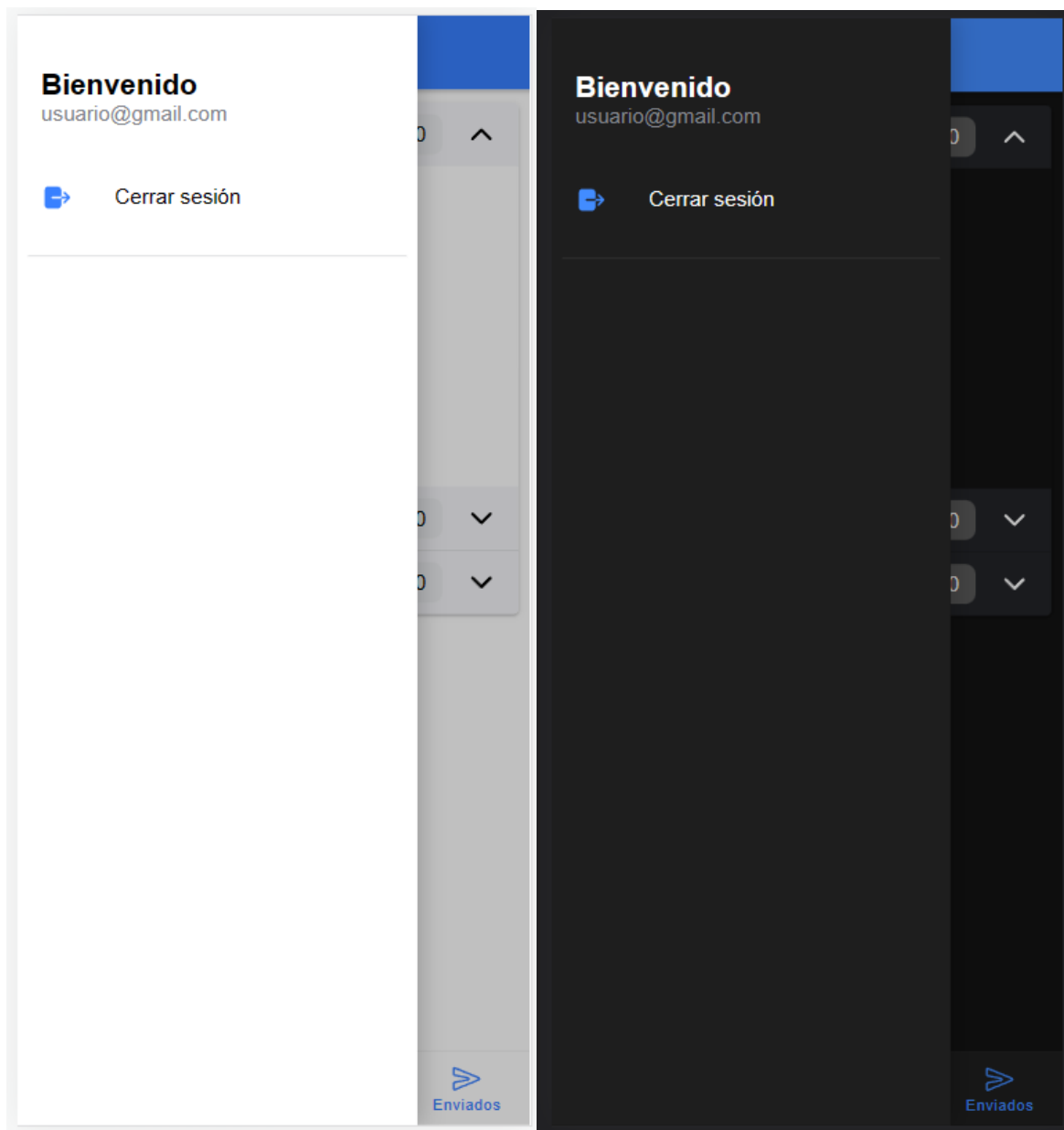


Ilustración 41 Menú – Sesión – Tickets enviados

Capítulo VI: Conclusiones

Las tecnologías van aumentando día con día a medida que actualmente ya contamos con sistemas para todo lo que se nos pueda ofrecer a si mismo ya hay varias herramientas de ayuda disponibles lo cuales son los "Help Desk" el cual le permiten al usuario el poder identificar un problema de algún sistema y poder reportarlo para su pronta atención y dar solución a los problemas suscitados dentro de una empresa u organización.

En la actualidad, las herramientas de "Help Desk" han evolucionado para incluir características avanzadas como la automatización de tareas, la integración con otros sistemas de gestión, el análisis de datos y la inteligencia artificial. Estas herramientas desempeñan un papel fundamental en la optimización de los procesos de soporte técnico, mejorando la satisfacción del usuario y permitiendo a las organizaciones gestionar eficientemente grandes volúmenes de solicitudes.

En resumen, las herramientas "Help Desk" se remontan a la necesidad de gestionar eficientemente el soporte técnico en las organizaciones. Han evolucionado desde sistemas basados en tickets hasta soluciones en línea con características avanzadas, mejorando la productividad y la calidad del servicio de soporte técnico.

Capítulo VII: competencias desarrolladas

1. Desarrollo de aplicaciones móviles: Al trabajar con Ionic y Angular, adquisición de experiencia en el desarrollo de aplicaciones móviles multiplataforma.
2. Creación de interfaces de usuario atractivas y receptivas, implementar funcionalidades móviles y optimizar el rendimiento de las aplicaciones.
3. Metodologías ágiles: Aprendizaje al trabajar con metodologías ágiles, a colaborar en equipos ágiles, participar en sprints, realizar seguimiento de tareas, participar en reuniones diarias y adaptarse a los cambios rápidos en los requisitos del proyecto. También adquisición de habilidades de comunicación efectiva y trabajo en equipo.
4. Gestión de servicios de TI (ITSM): Al familiarizarse con ITIL 4 y las prácticas de gestión de servicios de TI. Entender cómo se administran los servicios de TI en una organización. Esto implica comprender los conceptos de estrategia de servicios, diseño, transición, operación y mejora continua. También se puede aprender sobre la gestión de incidentes, problemas, cambios y configuraciones, así como la importancia de los SLA y la entrega de valor al cliente.
5. Cumplimiento de estándares ISO: Al tener conocimiento sobre ISO 20000. Comprender los requisitos y las mejores prácticas para la gestión de servicios de TI y trabajar en consonancia con los estándares internacionales. Esto implica el conocimiento sobre la estructura y los procesos establecidos por la norma, así como la implementación de controles para garantizar la calidad y la mejora continua.
6. Versionado de código y control de versiones: Es importante utilizar un sistema de control de versiones, como Git, para mantener un registro de los cambios realizados en tu código. Utiliza ramas (branches) para trabajar en nuevas funcionalidades o correcciones de errores sin afectar la versión principal. Esto facilitará la colaboración y permitirá un mejor manejo de las diferentes etapas de desarrollo.

Capítulo VIII: Glosario de términos

Aplicación móvil: aquella herramienta que los usuarios pueden utilizar accediendo a un dispositivo inteligente móvil a través de componentes nativos e internet.

Caso de uso: Un caso de uso es la descripción de una acción o actividad. Un diagrama de caso de uso es una descripción de las actividades que deberá realizar alguien o algo para llevar a cabo algún proceso.

Caso del negocio: Un caso de negocio es una herramienta de soporte y planeación que proyecta resultados financieros probables, así como otras consecuencias de negocio derivadas de una acción.

GitHub: (plataforma de desarrollo colaborativo) para alojar proyectos utilizando el sistema de control de versiones Git.

Historias de usuario: Una historia de usuario es una representación de un requisito escrito en una o dos frases utilizando el lenguaje común del usuario.

ITIL: Las siglas ITIL significan Information Technology Infrastructure Library, que traduciríamos literalmente como Biblioteca de Infraestructura de Tecnologías de Información. ITIL es una guía de buenas prácticas para la gestión de servicios de tecnologías de la información (TI)

Login: Combinación de números y letras que sirve para identificar a un usuario dentro del sistema.

Mesa de servicio: Un Help Desk es un equipo centralizado dentro de una empresa que atiende a empleados o clientes en masa, utilizando un producto de software para organizar las conversaciones.

Metodología ágil: Aquellas que permiten adaptar la forma de trabajo a las condiciones del proyecto, consiguiendo flexibilidad e inmediatez en la respuesta para amoldar el proyecto y su desarrollo a las circunstancias específicas del entorno.

Modelado flujo del negocio: Diagrama de flujo de negocios es sobre los eventos de transacción, incluidas las normas y reglamentos, mientras el diagrama de flujo del sistema es satisfacer al primero para informar y sistematizar todo, o parte de al menos el diagrama de flujo.

Normas ISO: ISO (International Standardization Organization) es la organización que se encarga de la creación de normas de fabricación, comercio y comunicación que tienen un alcance internacional.

Open Source: Es un modelo de desarrollo de software basado en la colaboración abierta.

Requerimiento: un requerimiento es una exigencia que tiene un software para poder funcionar de manera correcta.

SCRUM: Scrum es un marco que permite el trabajo colaborativo entre equipos. Al igual que un equipo de rugby (de donde proviene su nombre) cuando entrena para un gran partido, scrum anima a los equipos a aprender a través de las experiencias, a autoorganizarse mientras aborda un problema y a reflexionar sobre sus victorias y derrotas para mejorar continuamente.

Servidor: Un servidor es un conjunto de computadoras capaces de atender las peticiones de un cliente y devolverle una respuesta en concordancia.

SLA: "Service Level Agreement", en español (Acuerdo de nivel de servicio o Garantía de nivel de servicio).

Spring: Se trata de un miniproyecto de no más de un mes (ciclos de ejecución muy cortos -entre una y cuatro semanas), cuyo objetivo es conseguir un incremento de valor en el producto que estamos construyendo

Ticket de servicio: Un ticket de soporte técnico es una forma de hacernos llegar por escrito una consulta o de reportar un problema relacionado con cualquiera de nuestros servicios.

Usuario: un usuario es una persona que utiliza una computadora o un servicio de red.

Capítulo IX: Referencias

Instituto Nacional de Estadística y Geografía(INEGI). (s/f). *Demografía de los Negocios (DN) 2019 - 2021*. Org.mx. Recuperado el 21 de junio de 2023, de <https://www.inegi.org.mx/programas/dn/2021/>

Atlassian. (s/f). *Jira Service Management*. Atlassian. Recuperado el 21 de junio de 2023, de <https://www.atlassian.com/software/jira/service>

En, R., & Hurtado -Código, D. M. (s/f). *PROPUESTA DE UN SISTEMA DE INFORMACIÓN “MESA DE AYUDA” PARA*. Edu.co. Recuperado el 21 de junio de 2023, de <https://alejandria.poligran.edu.co/bitstream/handle>

HIXSA. (2022, octubre 7). *Help Desk y Service Desk Antecedentes históricos (Primera parte)*. HIXSA Blog | Sobre el mercado de ITSM, RPA y Gestión documental. <https://blog.hixsa.com/help-desk-y-service-desk-antecedentes-historicos-primera-parte/>

Hurtado, A. M. Z. (1606413300000). *Evolución del Intelligent Service Desk de TI*. Linkedin.com. <https://es.linkedin.com/pulse/evoluci%C3%B3n-del-intelligent-service-desk-de-ti-zegarra-hurtado>

Instituto Nacional de Estadística y Geografía(INEGI). (s/f). *Demografía de los Negocios (DN) 2019 - 2021*. Org.mx. Recuperado el 21 de junio de 2023, de <https://www.inegi.org.mx/programas/dn/2021/>

Navas Gómez, T. I., & Flórez Moreno, D. S. (2014). *Metodología para la implementación de una mesa de servicio usando buenas prácticas*. Universidad Autónoma de Bucaramanga UNAB.

((OTRS)) *Community Edition*. (2021, septiembre 2). OTRS; OTRS Group.

<https://otrs.com/es/soluciones-de-software-otrs/otrs/otrs-community-edition/>

Rodríguez Gallardo, J. A., López de la Madrid, M. C., & Espinoza de los Monteros

Cárdenas, A. (2018). Study of the implementation of Help Desk software in an institution of higher education. *PAAKAT: Revista de Tecnología y Sociedad*, 8(14), 1–20. <https://doi.org/10.32870/pk.a8n14.298>

Una Revisión de la Mesa de Ayuda TI de Sociedad Portuaria Puerto Bahía: Un Paso para la Automatización. (s/f). Edu.co. Recuperado el 21 de junio de 2023, de

<https://revistascientificas.cuc.edu.co/CESTA/article/view/4270/427>

Zendesk: Software de servicio de atención al cliente y CRM de ventas. (2020, octubre 15).

Zendesk. <https://www.zendesk.com.mx/>