

UNIVERSIDAD TÉCNICA DE AMBATO

FACULTAD DE INGENIERÍA EN SISTEMAS, ELECTRÓNICA E INDUSTRIAL

CARRERA DE TECNOLOGÌAS DE LA INFORMACIÒN

Tema:

HELPDESK USANDO HERRAMIENTAS OPENSOURCE PARA LA GESTIÓN DE ASISTENCIAS TÉCNICAS A CLIENTES EN LA EMPRESA CONNECTASERVICES

Trabajo de titulación modalidad Proyecto de Investigación, presentado previo a la obtención del título de Ingeniero en Tecnologías de la Información.

ÁREA: Base de datos y Sistemas informáticos

LÍNEA DE INVESTIGACIÓN: Tecnologías de la información y Sistemas de control.

AUTOR: Patricio Xavier Moscoso Collantes

TUTOR: Ing. Franklin Oswaldo Mayorga Mayorga, Mg

Ambato - Ecuador

febrero – 2025

APROBACIÓN DEL TUTOR

En calidad de tutor del trabajo de titulación con el tema: HELPDESK USANDO HERRAMIENTAS OPENSOURCE PARA LA GESTIÓN DE ASISTENCIAS TÉCNICAS A CLIENTES EN LA EMPRESA CONNECTASERVICES, desarrollado bajo la modalidad Proyecto de Investigación por el señor Patricio Xavier Moscoso Collantes, estudiante de la Carrera de Ingeniería en Tecnologías de la Información, de la Facultad de Ingeniería en Sistemas, Electrónica e Industrial, de la Universidad Técnica de Ambato, me permito indicar que el estudiante ha sido tutorado durante todo el desarrollo del trabajo hasta su conclusión, de acuerdo a lo dispuesto en el Artículo 17 del Reglamento para la Titulación de Grado en la Universidad Técnica de Ambato y el numeral 6.3 del instructivo del reglamento referido.

Ambato, febrero 2025.



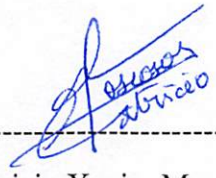
Ing. Franklin Oswaldo Mayorga Mayorga, Mg.

TUTOR

AUTORÍA

El presente trabajo de titulación con el tema: HELPDESK USANDO HERRAMIENTAS OPENSOURCE PARA LA GESTIÓN DE ASISTENCIAS TÉCNICAS A CLIENTES EN LA EMPRESA CONNECTASERVICES es absolutamente original, auténtico y personal y ha observado los preceptos establecidos en la Disposición General Quinta del Reglamento para la Titulación de Grado en la Universidad Técnica de Ambato. En tal virtud, el contenido, efectos legales y académicos que se desprenden del mismo son de exclusiva responsabilidad del autor.

Ambato, febrero 2025



Patricio Xavier Moscoso Collantes

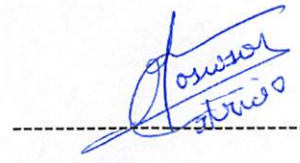
C.C. 1804262507

AUTOR

DERECHOS DE AUTOR

Autorizo a la Universidad Técnica de Ambato para que reproduzca total o parcialmente este trabajo de titulación dentro de las regulaciones legales e institucionales correspondientes. Además, cedo todos mis derechos de autor a favor de la institución con el propósito de su difusión pública, por lo tanto, autorizo su publicación en el repositorio virtual institucional como un documento disponible para la lectura y uso con fines académicos e investigativos de acuerdo con la Disposición General Cuarta del Reglamento para la Titulación de Grado en la Universidad Técnica de Ambato.

Ambato, febrero 2025



Patricio Xavier Moscoso Collantes

C.C. 1804262507

AUTOR

APROBACIÓN DEL TRIBUNAL DE GRADO

En calidad de par calificador del informe final del trabajo de titulación presentado por el señor Patricio Xavier Moscoso Collantes, estudiante de la Carrera de Tecnologías de la Información, de la Facultad de Ingeniería en Sistemas, Electrónica e Industrial, bajo la Modalidad Proyecto de Investigación, titulado HELPDESK USANDO HERRAMIENTAS OPENSOURCE PARA LA GESTIÓN DE ASISTENCIAS TÉCNICAS A CLIENTES EN LA EMPRESA CONNECTASERVICES, nos permitimos informar que el trabajo ha sido revisado y calificado de acuerdo al Artículo 19 del Reglamento para la Titulación de Grado en la Universidad Técnica de Ambato y el numeral 6.4 del instructivo del reglamento referido. Para cuya constancia suscribimos, conjuntamente con el señor presidente del Tribunal.

Ambato, febrero 2025



Firmado electrónicamente por:
FRANKLIN OSWALDO MAYORGA MAYORGA

Ing. Franklin Oswaldo Mayorga Mayorga, Mg.
PRESIDENTE DEL TRIBUNAL



Firmado electrónicamente por:
DAVID OMAR GUEVARA AULESTIA

Ing. David Guevara Aulestia, Mg.
PROFESOR CALIFICADOR



Firmado electrónicamente por:
HERNAN FABRICIO NARANJO AVALOS

Ing. Hernán Naranjo Avalos, Mg.
PROFESOR CALIFICADOR

DEDICATORIA

El presente trabajo es fruto de un grupo de personas que influenciaron mi ser, a ustedes se lo dedico, a todos y cada uno. Ambas familias, tanto de sangre como externa a ella, con el más profundo amor que puedo otorgar.

A mi familia de sangre, que en los tiempos más sombríos supo brindarme una luz en la cual acogerme. Cada consejo, cada enseñanza, cada enfado y cada alegría han hecho de mi lo que soy hoy en día. Principalmente a ustedes, a quienes están en vida y quienes viven celestialmente, es dirigido mi trabajo. No será el único, juntos atesoraremos muchos logros más. De eso estén seguros, jamás me rendiré, aún si atentase contra mi propia vida.

A mi familia externa, quienes forman una lista interminable de anécdotas inolvidables. Esta aventura no hubiese sido igual de no ser por cada ocurrencia y consejo, ayudándome a seguir adelante, proyectándome una sonrisa o escuchando mis problemas para aliviar mi conciencia. Los amo y por ello son parte de este gran logro. Siempre estaré en deuda con ustedes.

AGRADECIMIENTO

No hay palabras que puedan expresar lo agradecido que me siento ante su apoyo incondicional y su paciencia inquebrantable. A mis Padres, Abuelos y hermano, son a ustedes a quienes más atesoro en este mundo.

A mis amigos, que gracias a esos momentos tan especiales en noches estrelladas y desfogues de carisma junto con su entusiasmo y humor, con algunas lágrimas de por medio, agradezco de su compañía como parte de este gran pilar de la vida.

A la empresa ConnectaServices por darme esta oportunidad de aportar con mi granito de arena, además de ofrecerme conocimiento y experiencia para mi trayectoria profesional.

Al Ing. Franklin Mayorga por ayudarme en la realización de este proyecto, dispuesto a ofrecer su ayuda y aclarar mis dudas cuando más lo necesitaba.

ÍNDICE GENERAL DE CONTENIDOS

PORTADA I

APROBACIÓN DEL TUTOR II

AUTORÍA III

DERECHOS DE AUTOR IV

APROBACIÓN DEL TRIBUNAL DE GRADO V

DEDICATORIA VI

AGRADECIMIENTO VII

ÍNDICE GENERAL DE CONTENIDOS VIII

ÍNDICE DE TABLAS XI

ÍNDICE DE FIGURAS XVI

ÍNDICE DE ANEXOS XVIII

RESUMEN EJECUTIVO XIX

ABSTRACT XX

CAPÍTULO I. MARCO TEÓRICO 1

1.1 Tema de investigación 1

1.1.1 Planteamiento del problema 1

1.2 Antecedentes investigativos 2

1.3 Fundamentación teórica 4

1.3.1 Tecnologías de la información (TI) 4

1.3.2 Software 4

1.3.3 Desarrollo de Software 5

1.3.4 Framework 6

1.3.5 Metodologías de Desarrollo 6

1.3.6 Base de datos 7

1.3.7 Helpdesk 7

1.3.8 Administración de Empresas 8

1.3.9 Servicios Empresariales 8

1.3.10 Servicio al cliente 9
1.3.11 Asistencia técnica 10
1.4 Objetivos 11
1.4.1 Objetivo general 11
1.4.2 Objetivos específicos 11
CAPÍTULO II. METODOLOGÍA 12
2.1 Materiales 12
2.2 Métodos 17
2.2.1 Modalidad de la investigación 17
2.2.2 Investigación de campo 17
2.2.3 Investigación bibliográfica – documental 18
2.2.4 Población y muestra 18
2.2.5 Recolección de información 20
2.2.6 Procesamiento y análisis de datos 28
CAPÍTULO III. RESULTADOS Y DISCUSIÓN 31
3.1 Análisis y discusión de los resultados 31
3.1.1 Proceso identificado para la asistencia técnica 31
3.1.2 Elección de herramienta openSource para desarrollo Frontend 32
3.1.3 Elección de herramienta openSource para desarrollo Backend 35
3.1.4 Elección del gestor para Base de datos openSource 38
3.1.5 Metodologías de desarrollo para aplicaciones web 41
3.2 Desarrollo de la propuesta 43
3.2.1 Fase 1: Exploración 43
3.2.2 Fase 2: Planificación del proyecto 50
3.2.3 Fase 3: Diseño 52
3.2.4 Fase 4: Codificación 61
3.2.5 Fase 5: Pruebas 172
3.2.6 Implantación 175
CAPÍTULO IV. CONCLUSIONES Y RECOMENDACIONES 179
4.1 Conclusiones 179
4.2 Recomendaciones 180

REFERENCIAS BIBLIOGRÁFICAS **181**

ANEXOS **184**

ÍNDICE DE TABLAS

Tabla 1. Ficha De Observación Para Asistentes Técnicos 15

Tabla 2. Ficha De Observación Para Agentes De Venta 16

Tabla 3. Población Del Trabajo Investigativo. 18

Tabla 4. Muestra Seleccionada. 19

Tabla 5. Datos Recolectados 21

Tabla 6. Valores Reemplazados En La Formula. 22

Tabla 7. Interpretación Del Coeficiente Alfa De Cronbach. 22

Tabla 8. Descripción De Procesos 32

Tabla 9. Versiones De Sencha Ext Js 34

Tabla 10. Empresas Que Utilizan Sencha Ext Js Para Su Desarrollo Web 35

Tabla 11. Versiones De Node Js 37

Tabla 12. Empresas Que Utilizan Node Js En Desarrollo Web 38

Tabla 13. Versiones De Mysql 40

Tabla 14. Empresas Que Usan Mysql 41

Tabla 15. Comparativa De Metodologías De Desarrollo Ágil 42

Tabla 16. Roles Del Sistema 44

Tabla 17. Historia De Usuario 1 – Acceso Al Sistema 44

Tabla 18. Historia De Usuario 2 – Recuperar Contraseña 45

Tabla 19. Historia De Usuario 3 – Administración De Técnicos 45

Tabla 20. Historia De Usuario 4 – Administrador De Usuarios 45

Tabla 21. Historia De Usuario 5 – Administración De Roles 45

Tabla 22. Historia De Usuario 6 – Administración De Departamentos 46

Tabla 23. Historia De Usuario 7 – Administración De Planes 46

Tabla 24. Historia De Usuario 8 – Administración De Sistemas 46

Tabla 25. Historia De Usuario 9 – Cerrar Sesión 46
Tabla 26. Historia De Usuario 10 – Administración De Estados Para Tickets 47
Tabla 27. Historia De Usuario 11 – Administración De Prioridades Para Tickets.. 47
Tabla 28. Historia De Usuario 12 – Crear Ticket 47
Tabla 29. Historia De Usuario 13 – Editar Ticket 47
Tabla 30. Historia De Usuario 14 – Traspasar Ticket 48
Tabla 31. Historia De Usuario 15 – Cerrar Ticket 48
Tabla 32. Historia De Usuario 16 – Reaperturar Ticket 48
Tabla 33. Historia De Usuario 17 – Cambio De Estado Del Ticket 48
Tabla 34. Historia De Usuario 18 – Visualización De Tickets 49
Tabla 35. Historia De Usuario 19 – Mensajería Instantánea Para Chats 49
Tabla 36. Historia De Usuario 20 – Notificación De Tickets 49
Tabla 37. Historia De Usuario 21 – Tutoriales 49
Tabla 38. Historia De Usuario 22 – Reportes 50
Tabla 39. Estimación De Historias De Usuario 50
Tabla 40. Plan De Entrega 51
Tabla 41. Equipo De Trabajo 52
Tabla 42. Acceso Al Sistema 52
Tabla 43. Recuperar Contraseña 52
Tabla 44. Administración De Técnicos 53
Tabla 45. Administración De Usuarios 53
Tabla 46. Administración De Roles 53
Tabla 47. Administración De Departamentos 53
Tabla 48. Administración De Planes 53
Tabla 49. Administración De Sistemas 53
Tabla 50. Cerrar Sesión 54

Tabla 51. Administración De Estados Para Tickets 54
Tabla 52. Administración De Prioridades Para Tickets 54
Tabla 53. Crear Ticket 54
Tabla 54. Editar Ticket 54
Tabla 55. Traspasar Ticket 54
Tabla 56. Cerrar Ticket 55
Tabla 57. Reaperturar Ticket 55
Tabla 58. Cambio De Estado Del Ticket 55
Tabla 59. Visualización De Tickets 55
Tabla 60. Mensajería Instantánea Para Chats 55
Tabla 61. Notificación De Tickets 55
Tabla 62. Tutoriales 56
Tabla 63. Reportes 56
Tabla 64. Resumen Iteración 1 61
Tabla 65. Tarea 1 – Acceso Al Sistema 62
Tabla 66. Tarea 2 – Recuperar Contraseña 65
Tabla 67. Tarea 3 – Administración De Técnicos 68
Tabla 68. Tarea 4 – Administración De Usuarios 76
Tabla 69. Tarea 5 – Administración De Roles 89
Tabla 70. Resumen Iteración 2 93
Tabla 71. Tarea 6 – Administración De Departamentos 93
Tabla 72. Tarea 7 – Administración De Planes 98
Tabla 73. Tarea 8 – Administración De Sistemas 102
Tabla 74. Tarea 9 – Cerrar Sesión 106
Tabla 75. Resumen Iteración 3 109
Tabla 76. Tarea 10 – Administración De Estados Para Tickets 109

Tabla 77. Tarea 11 – Administración De Prioridades Para Tickets 115
Tabla 78. Tarea 12 – Crear Ticket 120
Tabla 79. Tarea 13 – Editar Ticket 126
Tabla 80. Resumen Iteración 4 134
Tabla 81. Tarea 14 – Traspasar Ticket 134
Tabla 82. Tarea 15 – Cerrar Ticket 135
Tabla 83. Tarea 16 – Reaperturar Ticket 137
Tabla 84. Tarea 17 – Cambio De Estado Del Ticket 141
Tabla 85. Resumen Iteración 5 145
Tabla 86. Tarea 18 – Visualización De Tickets 145
Tabla 87. Tarea 19 – Mensajería Instantánea Para Chats 157
Tabla 88. Tarea 20 – Notificación De Tickets 167
Tabla 89. Tarea 21 – Tutoriales 168
Tabla 90. Tarea 22 – Reportes 169
Tabla 91. Pruebas De Aceptación Para Iteración 1 172
Tabla 92. Pruebas De Aceptación Para Iteración 2 172
Tabla 93. Pruebas De Aceptación Para Iteración 3 172
Tabla 94. Pruebas De Aceptación Para Iteración 4 173
Tabla 95. Pruebas De Aceptación Para Iteración 5 173
Tabla 96. Pruebas De Aceptación Para Actividades Fallidas De Iteración 5 173
Tabla 97. Prueba De Usabilidad 1 174
Tabla 98. Prueba De Usabilidad 2 174
Tabla 99. Prueba De Usabilidad 3 174
Tabla 100. Prueba De Usabilidad 4 175
Tabla 101. Prueba De Usabilidad 5 175
Tabla 102. Cronograma De Capacitaciones 177

Tabla 103. Descripción De Procesos 196

ÍNDICE DE FIGURAS

figura 1. Pregunta 1 22
Figura 2. Pregunta 2 23
Figura 3. Pregunta 3 23
Figura 4. Pregunta 4 24
Figura 5. Pregunta 5 24
Figura 6. Pregunta 6 25
Figura 7. Pregunta 7 26
Figura 8. Pregunta 8 26
Figura 9. Pregunta 9 27
Figura 10. Pregunta 10 28
Figura 11. Estadísticas De Respuestas De La Encuesta 28
Figura 12. Pasos Realizados Para Asistencia Técnica 29
Figura 13. Diagrama De Procesos Optimizado 31
Figura 14. Modelo Entidad Relación 56
Figura 15. Acceso Al Sistema 58
Figura 16. Pantalla Principal 59
Figura 17. Módulos Administrativos 59
Figura 18. Módulo De Inicio 60
Figura 19. Contenido Del Ticket 61
Figura 20. Inicio De Sesión 62
Figura 21. Recuperar Contraseña 66
Figura 22. Administración De Técnicos 69
Figura 23. Administración De Usuarios 77
Figura 24. Administración De Roles 89

Figura 25. Administración De Departamentos 94
Figura 26. Administración De Planes 98
Figura 27. Administración De Sistemas 102
Figura 28. Botón De Cierre De Sesión 107
Figura 29. Administración De Estados Para Tickets 110
Figura 30. Administración De Prioridades Para Tickets 115
Figura 31. Creación Del Ticket 120
Figura 32. Detalle Ticket 126
Figura 33. Cerrar Ticket 135
Figura 34. Reaperturar Ticket 138
Figura 35. Formulario Para Envio De Mensajes 142
Figura 36. Visualización De Tickets 146
Figura 37. Manuales De Usuario 168
Figura 38. Reportes 170
Figura 39. Estructura De Documentos Pdf 171
Figura 40. Subdominio Creado Para El Aplicativo 176
Figura 41. Creación Del Servicio Node 176
Figura 42. Creación De La Base De Datos 177
Figura 43. Sistema En Producción 177

ÍNDICE DE ANEXOS

Anexo a. Fichas de observación a asistentes técnicos **184**

Anexo b. Anexo b. Fichas de observación a agentes de venta **192**

Anexo c. Diagrama de procesos para soporte técnico **196**

Anexo d. Formato para historias de usuario **197**

Anexo e. Manual de usuario **198**

RESUMEN EJECUTIVO

Los servicios de atención al cliente son fundamentales en el desarrollo de las empresas, ofrecen la satisfacción a los consumidores y hacen del crecimiento un proceso natural con el prestigio que se obtiene del mismo. Por lo que en la empresa ConnectaServices se estableció un sistema el cual facilite la organización de acontecimientos notificados.

El presente trabajo de investigación se centra en brindar una solución mediante la implantación de un sistema Helpdesk que permitió la gestión de los incidentes registrados. Se efectuó un análisis a los procesos de atención realizados por los agentes técnicos y se procedió a realizar encuestas aplicadas a los clientes de la empresa, consiguiendo una comprensión más precisa en el procedimiento de atención técnica e identificar sus posibles mejoras.

Se utilizó el framework Ext JS como herramienta para desarrollo del cliente junto con Node.js como herramienta para la recepción y tratado de datos, las mismas que fueron seleccionadas debido a su gran documentación y excelente rendimiento. Además, se aplicó la metodología ágil Extreme Programming (XP) permitiendo organizar el desarrollo y receptar cambios al final de cada iteración.

La solución presenta una gran capacidad para el manejo de incidencias, permitiendo una respuesta óptima gracias a la sección de manuales para usuarios y gestionando de forma adecuada el historial de los problemas registrados, mediante una interfaz amigable de chat en tiempo real que presenta el proceso manejado por cada incidente.

Palabras clave: Helpdesk, Ext JS, Node.js, Extreme Programming, gestión de incidencias técnicas, atención al cliente.

ABSTRACT

Customer services are fundamental in the development of companies, they offer satisfaction to consumers and make growth a natural process with the prestige that comes from it. Therefore, ConnectaServices established a system which facilitates the organisation of notified events.

This research work focuses on providing a solution through the implementation of a Helpdesk system that allowed the management of recorded incidents. The service processes carried out by the technical agents were analysed and surveys of the company's customers were carried out, achieving a more precise understanding of the technical service procedure and identifying possible improvements.

The Ext JS framework was used as a tool for client development along with Node.js as a tool for receiving and processing data, which were selected due to their great documentation and excellent performance. In addition, the agile methodology Extreme Programming (XP) was applied, allowing to organise the development and receive changes at the end of each iteration.

The solution presents a great capacity for incident handling, allowing an optimal response thanks to the user manual section and adequately managing the history of registered problems, through a friendly real-time chat interface that presents the process handled by each incident.

Keywords: Helpdesk, Ext JS, Node.js, Extreme Programming, management of recorded incidents, customer services.

CAPÍTULO I. MARCO TEÓRICO

1.1 Tema de investigación

HELPDESK USANDO HERRAMIENTAS OPENSOURCE PARA LA GESTIÓN DE ASISTENCIAS TÉCNICAS A CLIENTES EN LA EMPRESA CONNECTASERVICES

1.1.1 Planteamiento del problema

Tras la pandemia del Covid-19 muchas empresas en el mundo recurrieron a herramientas tecnológicas innovadoras que permitieran a los clientes obtener una atención adecuada de forma remota, eficiente y eficaz [1]. Las compañías optaron por implementar soluciones informáticas apresuradas para satisfacer las necesidades de los consumidores recurriendo a incrementar su visibilidad en redes sociales o brindando alternativas desarrolladas por programadores externos [2]. Hoy en día el incremento de sistemas de soporte ha tomado un rol importante dentro de las organizaciones y sus operaciones de asistencia técnica [3].

Como resultado, esta tecnología ha comenzado a implementarse en pequeñas y grandes empresas. En países de Latinoamérica existen negocios con apropiada gestión de TI (Tecnologías de la Información) que contribuyen con un 60% a la economía mundial debido a estrategias para resolución de problemas en el área de Tecnologías de la Información. Caso contrario a como se demuestra en la institución de educación superior en el Centro Universitario del Sur en Guadalajara - México donde el 45.33% de una muestra de 150 personas denotaron desconocimiento para requerir servicios de atención [4].

En Ecuador las instituciones han optado por modernizar los sistemas de atención al cliente mediante aplicativos HelpDesk ya que permiten mejorar sus tiempos de respuesta [5]. Tal es el caso de La Cruz Roja Ecuatoriana la cual implemento un servicio alojado en la nube para la atención al cliente, permitiendo optimizar el sistema brindando una respuesta eficaz ante la acumulación de peticiones por parte de los pacientes [6]. Seguidamente, permite simplificar la administración de grandes

volúmenes de datos como fue la Universidad Católica de Cotopaxi Extensión “La Maná” cuya investigación evitó la sobrecarga de trabajo y redujo la dificultad de uso hacia el personal secretario [5].

En Connectaservices, dado al ser una empresa que ofrece soluciones tecnológicas a terceros, se encuentra en constante crecimiento y presenta problemas de asistencia técnica hacia los clientes, debido a la perdida de registros de los mismos con forme al paso del tiempo, ocasionando la acumulación de quejas y entregando soluciones después de mucho tiempo.

1.2 Antecedentes investigativos

En la investigación de Christian Rivera [7] se detalla el desarrollo de un sistema HelpDesk mediante la programación por capas utilizando las tecnologías de ASP.NET Core para la Cooperativa de Ahorro y Crédito 9 de Octubre Ltda. Su metodología posee dos modalidades: bibliográfica – documental y de campo; la recolección de información se efectuó mediante una encuesta hacia una población de 20 personas en las cuales se encontraba el personal administrativo, operativo y el gerente. Tras evaluar los datos (los cuales presentaban la ineficiencia de un sistema HelpDesk obsoleto) se propuso la metodología de desarrollo XP (Extreme Programming) debido a la sofisticación, calidad y facilidad de mantenimiento a largo plazo. Los resultados obtenidos fueron satisfactorios al presentar un aplicativo optimizado para las diferentes agencias en las ciudades de Quito, Latacunga, Ambato, Mulalillo y la Mana en los tiempos acordados.

En el proyecto de titulación de Nahomy Reyes, et al. [8] define el desarrollo de un prototipo de un sistema para gestión de incidencias de servicio al cliente para la empresa de mantenimiento y servicios de equipos informáticos Remys S.A. La metodología de investigación exploratoria fue aplicada en el proyecto para la revisión de investigaciones en el área de soporte técnico sobre los procesos que se manejaban, tomando como población a toda el área, se aplicó el uso de entrevistas para la recolección de información. El desarrollo del proyecto aplicó la Metodología en Cascada debido a que cada fase se presentaba de forma secuencial facilitando la gestión del desarrollo. Se concluyó en la correcta implementación de un sistema el

cual permitía ajustarse a los requerimientos de la empresa junto con la facilidad de ejecución y control hacia incidencias reportadas por usuarios.

Según Aleksandra Revina, et al. [9] se realiza la implementación de servicio HelpDesk o mesa de ayuda para la empresa MachalaGPS. En la investigación se aplicó la metodología proyectiva debido a que la empresa propiciaba un entorno distinto. La población establecida del proyecto fue un total de 5 personas, conformando la totalidad de la empresa, analizando los datos con entrevistas y encuestas a todos los participantes. Una vez realizado el análisis de los datos, la metodología de desarrollo ITIL se implementó en la investigación ya que en las primeras fases de desarrollo permitía el rediseño de la Mesa de Ayuda además de ofrecer la capacidad para definir requisitos, acuerdos y proveedores para su implementación en determinadas fases. El resultado del proyecto fue el correcto control de dispositivos GPS (Global Positioning System) en mal estado para su reparación, aportando beneficios a todo el personal en la empresa como visualización y detalle de tickets, atención rápida e información inmediata.

En la investigación de Yashraj Chanchad, et al. [10] tiene como objetivo el desarrollar un sistema HelpDesk implementando una esquematización de tickets para maximizar la eficiencia y satisfacción de los usuarios. En su investigación no se menciona el tipo de metodología que se utilizó para el desarrollo de la investigación, sin embargo, muestra las bases para la categorización de tickets, la planificación de la base de datos y diseños definidos para software HelpDesk. En su conclusión redacta que el sistema provee de una herramienta simple pero poderosa al ser un sistema integrado con Python, Django y SQLite permitiendo implementarse en cualquier organización o institución.

En el paper elaborado por Didi Kurnaedi, et al. [11] posee el objetivo de desarrollar un HelpDesk para el reporte de quejas y el monitoreo de sistemas de la compañía mencionada en el título. La metodología aplicada en la investigación es experimental y sus métodos de recolección de datos son la observación y la entrevista hacia el departamento de TI de la empresa. La investigación concluye con la implementación del HelpDesk en la empresa evitando el uso de papel, integrando los datos de los tickets

en la base de datos y facilitando el uso hacia usuarios y agentes de atención los cuales pueden realizar reportes o exportar datos en Excel.

1.3 Fundamentación teórica

1.3.1 Tecnologías de la información (TI)

Las Tecnologías de la Información se refiere a todo uso de tecnologías para uso empresarial. Incluye cualquier acción relacionada a la creación de redes para empresas, guardar y proteger información, administrar bases de datos, ayudar a empleados o clientes, creación/mantenimiento de aplicaciones para múltiples plataformas o cualquier otra operación relacionada con la manipulación y seguridad de información empresarial [12].

Las tecnologías de la información pueden obtener 3 enfoques: el procesamiento de la información mediante técnicas, el uso de modelos matemáticos y estadísticos para la toma de decisiones o la simulación del pensamiento mediante programas informáticos [12].

También se le conoce como Tecnologías de la información a todo uso de un computador, dispositivo de almacenamiento, red, infraestructura y proceso para crear, procesar, almacenar, proteger e intercambiar toda forma de datos electrónicos [13].

1.3.2 Software

Software es todo componente lógico no tangible que posee un sistema informático. Los componentes lógicos hacen referencia a los procesos, reglas, documentación y datos asociados de un programa para cumplir una tarea especifica. Es un elemento el cual posee las siguientes características:

- El software se desarrolla, no se fabrica en un sentido clásico.
- No se desgasta con el tiempo.
- El software se construye a medida.
- Es creado en base a la lógica puramente.

El software puede presentarse como una solución factible para muchas empresas y negocios ya que debido a ella se constituyen la suficiencia, exactitud y oportunidad de la información otorgada permitiendo el crecimiento de desarrolladores en los últimos años, sin embargo, un software puede presentar múltiples problemas a la hora de no definir correctamente su planificación como puede ser:

- La falta de productividad.
- Calidad inaceptable.
- No existe tiempo suficiente para la recolección de datos en el desarrollo del software.
- El mantenimiento puede resultar costoso o no se consideró dentro de la fase de desarrollo [14].

1.3.3 Desarrollo de Software

El Desarrollo de Software es la agrupación de actividades centradas en el diseño, elaboración, implementación y soporte de software. Las actividades son clasificadas en seis fases los cuales son:

- Análisis de requerimientos: se centra en todas las características a integrar para el proyecto.
- Diseño: define la arquitectura del proyecto.
- Desarrollo: fase en la cual se procede a ejecutarse el proyecto.
- Pruebas: se comprueban los resultados de la fase de desarrollo.
- Implementación y Mantenimiento: se centra en las futuras actualizaciones y correcciones.

También se encuentran los modelos y metodologías de desarrollo para facilitar el proceso de planificación y desarrollo, de las cuales se dividen en dos grupos:

- Las metodologías ágiles los cuales gestionan un proyecto en múltiples fases como es el caso de Scrum la cual ofrece flexibilidad al presentar iteraciones que se subdividen en tareas las cuales producen un resultado concreto.

- Las metodologías tradicionales las cuales siguen un proceso ordenado como puede ser la metodología en cascada que se centra en el ordenamiento y seguimiento de sus fases aunque no ofrece flexibilidad en caso de realizarse cambios a posterior [13].

1.3.4 Framework

Un Framework es una estructura o conjunto de herramientas predefinidas que los programadores utilizan para facilitar y acelerar el proceso de desarrollo. Permite mantener un código limpio y la capacidad de reutilizar secciones de estos. Los Frameworks ayudan a colaborar en equipo, reducir tareas repetitivas y garantizar la consistencia en los proyectos [15].

1.3.5 Metodologías de Desarrollo

Una metodología de desarrollo de software es una pauta de trabajo que se utiliza para organizar, planificar y controlar el proceso del desarrollo de un sistema de información. La importancia de una metodología dentro de un proyecto es fundamental debido a que la metodología aplicada ayuda a establecer plazos de tiempo, presupuestos, actividades y tiempos de revisión.

a. Tipos de metodologías

Metodologías Ágiles: Son aquellas que se basan en procesos iterativos y de corta duración, donde es posible realizar cambios dentro de los procesos.

Metodologías Tradicionales: Se basan en un control de procesos donde se establece las actividades involucradas, el alcance del proyecto, los entregables a generar y las herramientas junto con el seguimiento que utilizaran.

Metodologías Hibridas: Son aquellas que combinan las mejores prácticas de las metodologías tradicionales y ágiles, combinando así sus ventajas [16].

1.3.6 Base de datos

Una base de datos es una recopilación sistemática de datos almacenados electrónicamente. Los datos almacenados pueden ser de cualquier tipo de dato ya sea palabras, números, imágenes, videos o archivos. El objetivo principal que poseen es el ofrecer seguridad, integridad y escalado eficiente de los datos almacenados [17].

a. Sistema Gestor de base de datos

Un gestor de base de datos, también conocido como Sistema de Gestión de Bases de Datos (SGBD), es un software que permite la creación, gestión y administración de bases de datos. Su principal función es organizar los datos de manera estructurada para facilitar su acceso, manipulación y mantenimiento. Los SGBD proporcionan una interfaz que permite a los usuarios realizar operaciones como la inserción, actualización, eliminación y consulta de datos, mientras que aseguran la integridad y seguridad de la información almacenada [18].

1.3.7 Helpdesk

Helpdesk es un sistema el cual brinda asistencia hacia los usuarios por parte de un departamento de TI. La asistencia recibida se efectúa cuando se presenta un evento en los servicios tecnológicos o en la infraestructura de la empresa, permitiendo llevar un registro y control ayudando al manejo rápido de los mismos. También tiene la capacidad de realizar atenciones telefónicas y responder de forma remota o presencial para resolver los requerimientos.

El sistema está formado por un conjunto de recursos que pueden estar formados por recursos tecnológicos y humanos que trabajan relacionalmente estableciéndose en áreas las cuales pueden definirse en: soporte de hardware, soporte de software, soporte de redes, soporte de seguridad y soporte en tareas de usuario.

Es una herramienta fundamental que permite a un negocio contar con un ecosistema tecnológico e infraestructura amplia para la gestión de servicios facilitando el manejo de procesos y enfrentándose a cualquier evento u obstáculo presente [19].

1.3.8 Administración de Empresas

La Administración de empresas es toda planeación, organización, dirección y control de actividades en una empresa. Toda empresa, sin importar su tamaño, necesita de ella debido a las condiciones que se generan permitan el crecimiento y existencia de una institución. En pocas palabras, es el rumbo racional de toda actividad dentro de una institución.

Es imprescindible para la existencia, supervivencia y éxito de la organización el contar con una buena planificación, más aún cuando el desarrollo de las empresas se centra en el tener una comunicación de punta permitiendo la interacción eficaz entre una empresa y sus consumidores.

Las empresas buscan presentar su atención y dirigir sus recursos en función hacia la optimización de los siguientes parámetros:

- La percepción del producto o servicio.
- La fabricación, operación y prestación de servicios.
- Construcción de soluciones para prestar operación y mantenimiento [20].

1.3.9 Servicios Empresariales

Los Servicios Empresariales son todos los beneficios que se ofrecen al cliente, sea de forma remota o presencial mediante un equipo capacitado. Estos poseen procesos internos y metodologías los cuales se centran en corregir errores sin la intervención del cliente, cada proceso puede poseer múltiples subprocesos añadiendo un valor agregado [21].

Los procesos dentro del servicio empresarial son las acciones que se realizan de forma natural o por intervención humana hacia determinados recursos con el fin de obtener ciertos resultados, poseen un inicio, una ejecución y un fin dentro de la cotidianidad como son en la preparación de alimentos o en la asignación de tareas. En el ámbito empresarial, se define a los procesos como la forma eficiente de aprovechar los recursos de una empresa, acortando tiempos y evitando accidentes [21].

Las metodologías dentro del ámbito empresarial es una estructura el cual permite el estudio de los procesos y procedimientos que se llevan a cabo, realizando un análisis de su eficiencia y eficacia para llegar al objetivo propuesto con resultados significativos.

Al tratarse de un producto intangible, se mide con características cualitativas como la confianza del cliente o prestigio institucional frente a propuestas similares [21].

1.3.10 Servicio al cliente

El servicio al cliente es un proceso el cual ocurre antes, durante y después de conseguir un producto o de utilizar un beneficio. El responder cualquier inquietud del cliente aun cuando no la ha solicitado. Involucra aspectos como la amabilidad, calidad y calidez otorgando una ventaja competitiva en el mercado. El buen servicio al cliente refleja el compromiso de una institución hacia el público general. En la atención al cliente se radica en los siguientes aspectos:

- El valor agregado de una buena atención al cliente permite destacar de entre una mayor demanda entre los productos ofertados.
- La competencia se nutre de buen equipamiento en calidad y precio, por ello es mejor brindar una diferenciación sustancial.
- Los clientes buscan un servicio rápido, cómodo y agradable.
- Permite mejorar la reputación de un negocio mediante el habla y recomendaciones, incluso retornando a requerir más bienes.

Para medir la calidad de la atención de los clientes se utilizan modelos de evaluación al cliente, uno de los más aplicados es el modelo SERVQUAL el cual propone cinco indicadores para demostrar la calidad en los servicios:

- Aspectos tangibles: apariencia de las instalaciones.
- Fiabilidad: prestación de un servicio fiable y preciso.
- Capacidad de respuesta: disposición del personal para un servicio rápido.
- Seguridad: inspira credibilidad y confianza mediante los conocimientos, atención y habilidad del personal.
- Empatía: atención individualizada.

Mediante estos parámetros se define una calificación la cual permite obtener las expectativas y sentimientos de un cliente, los ámbitos en los cuales nuestro servicio no cumple con las expectativas, facilitar la toma de medidas correctivas y por último el conocer las deficiencias en la calidad de nuestros servicios [22].

1.3.11 Asistencia técnica

La asistencia técnica es proveer recursos tecnológicos, conocimientos técnicos o gestión para implementar proyectos. Su principal objetivo es maximizar la calidad de todos los aspectos anteriores junto con el impacto hacia el consumidor aumentando la sostenibilidad financiera, endureciendo la resiliencia organizacional, aumentando la capacidad de inversión y obteniendo resultados satisfactorios. Existen tres grupos de personas que se manejan en la asistencia técnica:

- Demandantes: son organizaciones que necesitan información específica para transitar a sistemas de gestión sostenibles ya sea en gobernanza o planificación.
- Ofertantes: son organizaciones que financian la asistencia técnica u ofrecen servicios de asesoría, aquellos que financian se refiere a subcontratar a terceros para realizar asistencias o apoyan de forma económica directamente hacia el departamento encargado.
- Constructores del ecosistema: pueden ser demandantes u ofertantes que promueven indirectamente a la asistencia técnica con la generación de nuevo conocimiento.

También se define como una forma de compartir conocimientos, instrucciones, formación de habilidades, datos técnicos y servicios de consultoría permitiendo el fortalecimiento de las capacidades. Es una herramienta clave que permite mejorar el diseño de políticas y proyectos, mejora las habilidades de una institución además de llevarlo por el camino del desarrollo [23].

1.4 Objetivos

1.4.1 Objetivo general

Implantar un HelpDesk usando herramientas openSource para la gestión de asistencias técnicas a clientes en la empresa ConnectaServices.

1.4.2 Objetivos específicos

- Determinar los procesos involucrados en la gestión de asistencias técnicas a clientes en la empresa.
- Determinar la mejor herramienta openSource para desarrollo de Sistemas web.
- Desarrollar un Sistema web HelpDesk usando herramientas openSource para la gestión de asistencias técnicas a clientes en la empresa ConnectaServices.

CAPÍTULO II. METODOLOGÍA

2.1 Materiales

Los materiales aplicados en esta investigación para la recolección de la información fueron una encuesta para los clientes de la empresa y fichas de observación dedicadas a las acciones realizadas por el personal de la empresa (asistentes técnicos y departamento de ventas).

Encuesta aplicada a los clientes

La encuesta aplicada a los clientes tuvo como objetivo el obtener información sobre las opiniones de los clientes de la empresa ConnectaServices al momento de requerir el soporte técnico para implantar un sistema HelpDesk que permita gestionar las asistencias técnicas de los clientes en la empresa ConnectaServices.

El cuestionario posee diez preguntas las cuales se detallan a continuación:

1) ¿En qué rango de edad se encuentra?

a. Menor de 18 años
b. 18 a 30 años
c. 30 a 50 años
d. Mayor a 50 años

2) ¿Qué medio utiliza usted para reportar sus solicitudes?

a. Llamada
b. Videollamada
c. Mensaje por WhatsApp
d. Correo Electrónico

3) ¿Qué tan frecuente se comunica con el canal de soporte?

a. Una vez al mes
b. Una vez a la semana

c. Mas de dos veces por semana

d. Todos los días

4) ¿En cuánto tiempo suelen responder a sus incidencias una vez enviados a través de los medios mencionados?

a. Inmediata

b. 5 minutos

c. 30 minutos o mas

d. Nunca

5) ¿Qué tan satisfecho está con la atención recibida al momento de informar su caso?

a. Muy insatisfecho

b. Insatisfecho

c. Satisfecho

d. Muy satisfecho

6) ¿En qué tiempo es resuelto un problema notificado?

a. Menos de 24 horas

b. Menos de 48 horas

c. Mas de 48 horas

d. Nunca

7) ¿Todos sus reclamos han sido solucionados?

a. Si

b. No

8) ¿Cómo califica las soluciones brindadas a sus notificaciones?

a. Totalmente mala

b. Ineficiente

c. Buena

d. Excelente

9) ¿Cuál es su conocimiento sobre el proceso interno que se realiza al comunicar una petición?

a. Muy bajo

b. Bajo

c. Alto

d. Muy Alto

10) ¿Estaría de acuerdo con el desarrollo de un sistema informático que permita ingresar sus propios tickets y que estos sean gestionados de forma adecuada?

a. Si

b. No

Ficha de observación (Asistentes técnicos)

El objetivo de la ficha de observación, como se muestra en la Tabla 1, tuvo como objetivo el determinar los procesos involucrados en la gestión de asistencias técnicas a clientes en la empresa.

Tabla 1. Ficha de observación para asistentes técnicos

Ficha de observación				
Fecha y hora de la observación: **Tema:** **Duración total del soporte:** **Objetivo:**				
Indicadores	**Observación**		**Novedades**	
Recepción de la solicitud				
¿Cómo se recibe la solicitud? (ticket, llamada, correo)				
¿Se solicita información adicional al cliente?				
Tiempo entre recepción e inicio del proceso	Inmediata ☐	15 min ☐	30 min ☐	Más de 30 min ☐
Duración (De la fase)				
Identificación y diagnóstico del problema				
Acciones del asistente para identificar el problema				
¿Se realiza alguna verificación inicial?				
Comunicación con el cliente para confirmar detalles del problema.				
Duración (De la fase)				
Resolución del problema				
Acciones realizadas por el asistente técnico.				
Uso de herramientas o software específico.				
Verificación final con el cliente para asegurar la resolución.				
Duración (De la fase)				
Seguimiento y cierre				
¿Se ofrece alguna indicación adicional al cliente para prevenir futuros problemas?				
Registro en la plataforma de gestión de tickets (si aplica).				
Tiempo total del proceso de atención				

Indicadores	Observación	Novedades
Duración (De la fase)		
Interacción con el cliente		
Descripción de los intercambios con el cliente durante el soporte.		
Canales utilizados para comunicarse con el cliente (llamada, correo, chat, etc.).		
Notas adicionales		
Notas		

Ficha de observación (Departamento de ventas)

El objetivo de la ficha de observación para el departamento de ventas tuvo como objetivo el determinar los procesos involucrados en la gestión de asistencias técnicas a clientes en la empresa por parte del departamento de ventas.

Tabla 2. Ficha de observación para agentes de venta

Ficha de observación			
Fecha y hora de la observación: **Tema:** **Duración total del soporte:** **Objetivo:**			
Indicadores	**Observación**		**Novedades**
Recepción de la solicitud			
¿Cómo se recibe la solicitud? (ticket, llamada, correo, etc.)			
¿Cuál es la actitud del agente?			
¿Se solicita información adicional al cliente?			
¿El agente identifica rápidamente las necesidades del cliente?	Si ☐	No ☐	
Duración (De la fase)			
Presentación del producto / servicio			
¿Cómo presenta el agente el producto o servicio?			
¿Se mencionan características?			

Indicadores	Observación	Novedades
¿Se realizan demostraciones?		
¿Qué herramientas se utilizó para presentar el producto? ¿Fueron suficientes?		
Duración (De la fase)		
Objeciones		
¿El cliente presenta dudas?		
¿El agente responde las dudas con certeza?		
¿El agente se muestra positivo y proactivo ante las objeciones?		
Duración (De la fase)		
Cierre y Seguimiento		
¿Se llego a algún acuerdo?		
¿Qué decisión toma el cliente?		
¿Se ofrece alguna indicación adicional al cliente?		
¿Se acuerda una acción futura?		
Tiempo total del proceso de atención		
Actitud del cliente durante el soporte		
Duración (De la fase)		
Notas adicionales		
Notas		

2.2 Métodos

2.2.1 Modalidad de la investigación

2.2.2 Investigación de campo

La investigación de campo fue aplicada al momento en que el investigador interactuó directamente en el lugar (la empresa) donde se produce una problemática analizando sus causas y efectos.

2.2.3 Investigación bibliográfica – documental

La Investigación bibliográfica – documental fue aplicada debido a la obtención de información similar a la problemática y sus soluciones, propuestas que ayudaron a establecerse como las bases del proyecto investigativo.

2.2.4 Población y muestra

La población que se seleccionó para el proyecto de investigación se centra en el personal que realiza la atención al cliente, dicha población estuvo compuesta por un gerente, una administradora de proyectos, dos programadores, una secretaria que realiza la atención hacia la sección de ventas y 1.026 clientes.

Tabla 3. Población del trabajo investigativo.

Población	Número	Porcentaje
Gerente	1	0,10%
Project Manager	1	0,10%
Programadores Junior	2	0,19%
Secretaria	1	0,10%
Clientes	1016	99,51%
Total	1021	100,00%

La población en cuanto a la selección de clientes fue superior a cien elementos, por ende se realizó una muestra significativa.

Muestra

El cálculo de la muestra se realizó de forma probabilística sin tomar en cuenta el tamaño de la población total. La fórmula utilizada es la población infinita en la variable cualitativa debido a que la población puede crecer con el paso del tiempo, además de seleccionar aleatoriamente los participantes. Se define mediante:

$$n = \frac{Z^2 PQ}{e^2}$$

En donde:

- n = tamaño de la muestra
- Z = nivel de confianza del 95% equivalente a 1,96
- P = probabilidad de éxito con el valor de 0,5
- Q = probabilidad de fracaso con el valor de 0.5
- e = error estándar con el valor del 31% que equivale a 0.31

Sustituyendo los datos, obtenemos la siguiente cifra:

$$n = \frac{(1.96)^2(0.5)(0.5)}{(0.31)^2} = 9.98$$

El tamaño corresponde a 10 personas en el grupo de clientes. Por lo tanto se trabajará con todos los integrantes de la empresa (Gerente, Project Manager, Programadores, Secretaria) y con una muestra de 10 clientes seleccionados al azar.

Tabla 4. Muestra seleccionada.

Población	**Número**	**Porcentaje**
Gerente	1	6,66%
Project Manager	1	6,66%
Programadores Junior	2	13,35%
Secretaria	1	6,66%
Clientes	10	66,67%
Total	15	100%

2.2.5 Recolección de información

La recolección de la información se realizó mediante la aplicación de una encuesta dirigida a los clientes de la empresa ConnectaServices, además de aplicar fichas de observación hacia los asistentes técnicos de la empresa y agentes de ventas para poder extraer los procesos involucrados.

Resultado de las fichas de Observación

A continuación se presentan los resultados obtenidos de las fichas de observación aplicadas a los asistentes técnicos y a los agentes de venta.

a. Asistentes técnicos

Se obtuvieron 5 fichas de observación realizadas a los asistentes técnicos, las cuales se muestran en el **Anexo A**.

Las 5 fichas de observación presentan patrones generales al momento de realizar la atención técnica solicitada por el cliente. El primer paso que se realizó es el requerir el RUC del cliente para verificar si este se encuentra asociado a algún sistema o servicio de la empresa. Posteriormente se procede a revisar el sistema del cliente para verificar porque se produce y encontrar la solución al mismo, cabe mencionar que en esta fase el soporte se alarga indefinidamente ya que depende enteramente del asistente, también se encontró que si una anomalía no se puede solucionar, el soporte se traspasa al líder del área para que este sea quien lo resuelva. Finalmente se notifica al cliente la solución propuesta donde puede corroborar su solución.

b. Agentes de venta

Se obtuvieron 3 fichas de observación realizadas a los agentes de venta, los cuales se muestran en el **Anexo B**.

Las 3 fichas de observación demuestran que los soportes recibidos son a través de múltiples medios pero el patrón que se presenta se denota como el proceso general de cómo se realizan los procesos de asistencia, con la diferencia que en la fase de solución

a una anomalía, se cambia por una videoconferencia con el cliente para presentar el producto o servicio, mencionando a detalle todo lo que se ofrece.

c. Encuesta realizada a los clientes de la empresa

Para realizar la validación de la encuesta, se aplicó el coeficiente Alfa de Cronbach el cual es una herramienta estadística psicométrica que permite verificar la veracidad y eficiencia de una encuesta realizada con escalas de medición. La aplicación se efectúa mediante la siguiente fórmula:

$$\alpha = \frac{k}{k-1}(1 - \frac{\sum S_i^2}{S_t^2})$$

Donde:

k = El número de ítems en el cuestionario (preguntas)

$\sum S_i^2$ = Sumatoria de la varianza de los ítems.

S_t^2 = Varianza de la suma de los ítems.

Con los datos obtenidos, se procedió a realizar una matriz en Excel como se muestra en la Tabla 5 con 10 clientes seleccionados de los 20 participantes para realizar los cálculos necesarios para ser aplicados en la formula.

Tabla 5. Datos recolectados

Sujeto	item 1	item 2	item 3	item 4	item 5	item 6	item 8	item 9	Totales
sujeto 1	3	4	4	4	2	3	2	1	23
sujeto 5	3	4	4	4	3	3	3	2	26
sujeto 12	3	3	4	4	3	3	1	1	22
sujeto 13	3	2	2	2	1	3	1	2	16
sujeto 15	4	3	4	4	3	3	3	2	26
sujeto 16	2	3	2	3	2	2	1	2	17
sujeto 17	3	3	4	4	2	2	2	1	21
sujeto 18	4	3	4	4	2	3	1	2	23
sujeto 19	2	2	2	3	1	2	1	2	15
sujeto 20	4	3	4	4	2	3	2	1	23
Varianzas	0.5444	0.4444	0.9333	0.4889	0.5444	0.2333	0.6778	0.2667	15.5111

En la fórmula de Alfa de Cronbach se reemplazó los valores calculando de la siguiente manera:

$$\alpha = \frac{k}{k-1}\left(1 - \frac{\sum S_i^2}{S_t^2}\right)$$

Tabla 6.Valores reemplazados en la formula.

$\sum S_i^2$	**Sumatoria Varianzas**	4.1333
S_t^2	**Varianza total**	15.5111
k	**Numero de items**	8
$\frac{k}{k-1}$	**Calculo A**	1.1429
$(1 - \frac{\sum S_i^2}{S_t^2})$	**Calculo B**	0.7335
α	**α de Cronbach**	0.8383

Tabla 7.Interpretación del Coeficiente Alfa de Cronbach.

Item	**Valor**
Mayor a 0.9	Excelente
0.80 - 0.89	Buena
0.70 - 0.79	Aceptable
Menor a 0.69	Deficiente

Como se muestra en la Tabla 6, el valor obtenido de los datos es 0.8383 lo que al momento de verificar en la Tabla 7 se encuentra en el rango de coeficiente Bueno demostrando la confiabilidad de los datos obtenidos.

Resultados de la encuesta

Primera Pregunta: ¿En qué rango de edad se encuentra?

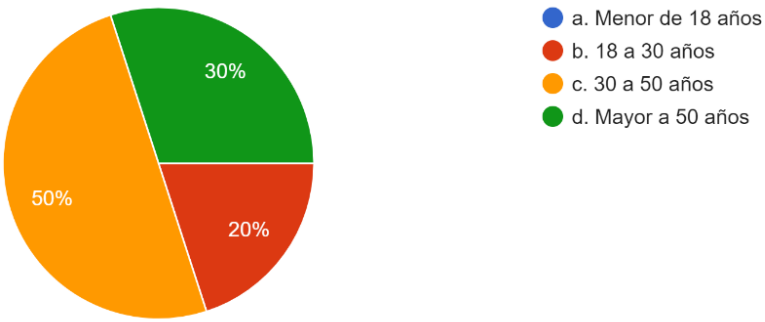


Figura 1. Pregunta 1

Análisis e interpretación de resultados

En la Figura1 se muestra que el 50% de los participantes se encuentra entre los 30 y 50 años, lo cual, junto con el 20% siendo menores de 30 años, los datos muestran que la mayoría de los participantes son adultos los cuales muestran experiencia en el uso de sistemas informáticos.

Segunda Pregunta: ¿Qué medio utiliza usted para reportar sus solicitudes?

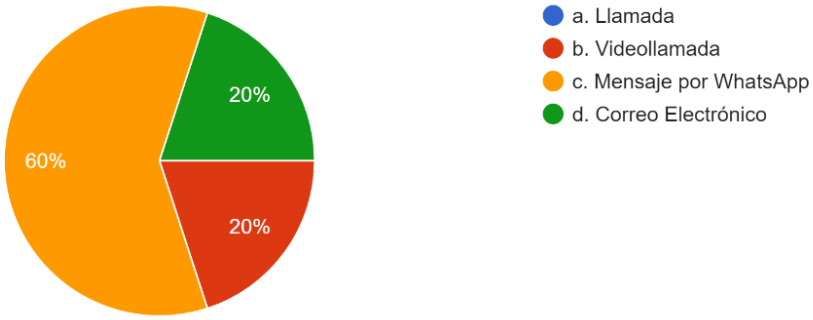


Figura 2. Pregunta 2

Análisis e interpretación de los resultados

En la Figura 2 se muestra que el 60% de las peticiones de los participantes se realizan mediante mensajes de WhatsApp, mientras que el 40% lo realiza por medios distintos, como videollamadas o correos electrónicos. Esta diferencia verifica que no se organiza apropiadamente el registro de soportes técnicos ya que mayormente se encuentran registrados en múltiples entornos.

Pregunta 3: ¿Qué tan frecuente se comunica con el canal de soporte?

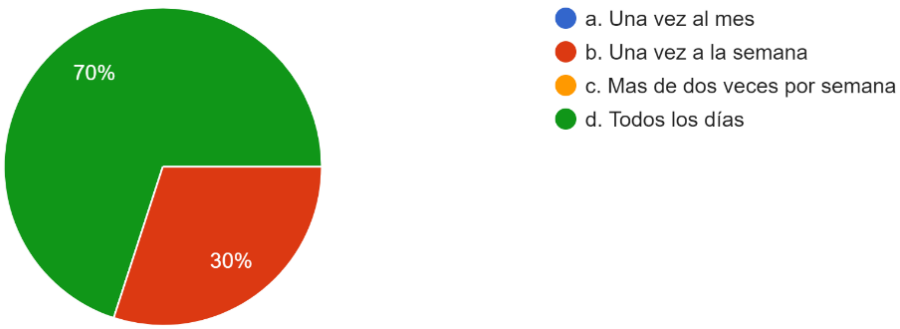


Figura 3. Pregunta 3

En la Figura 3 se presenta que el 70% de los participantes utilizan el canal de soporte todos los días y el resto lo realiza mínimo una vez a la semana. Estos datos indican que el soporte realizado no se efectúa de forma correcta ya que la mayoría de los clientes reportan incidencias más de una vez a la semana.

Pregunta 4: ¿En cuánto tiempo suelen responder a sus incidencias una vez enviados a través de los medios mencionados?

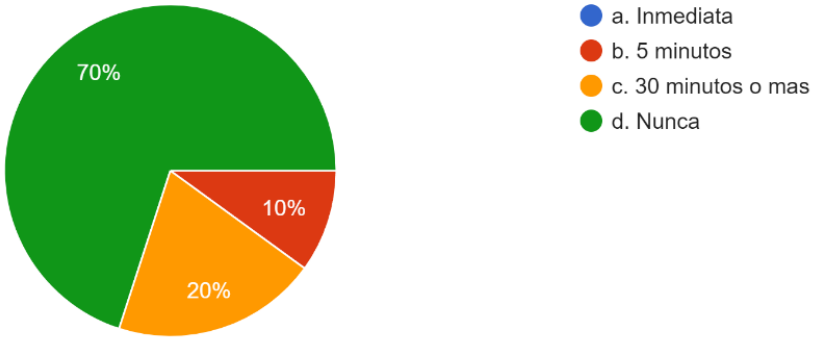


Figura 4. Pregunta 4

Análisis e interpretación de los resultados

Como se evidencia en la Figura 4, el 70% de los participantes reporta que las incidencias no se responden, mientras que el 30% son respondidos en más de 5 minutos. Los datos demuestran que los asistentes no están pendientes en la mayoría de las incidencias.

Pregunta 5: ¿Qué tan satisfecho está con la atención recibida al momento de informar su caso?

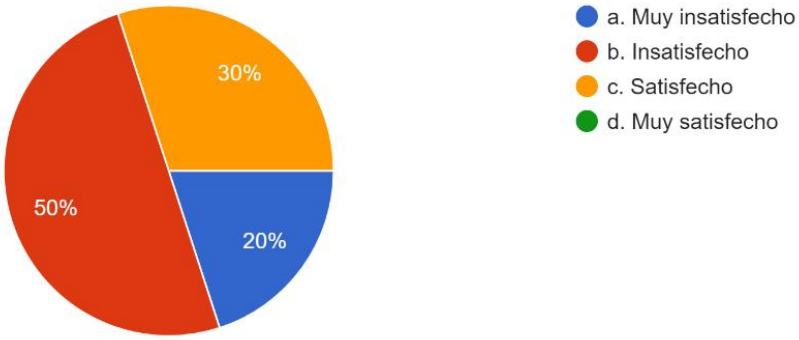


Figura 5. Pregunta 5

Análisis e interpretación de los resultados

Como se evidencia en la Figura 5, el 50% de los participantes muestran descontento en cuanto a las soluciones por parte de los técnicos indicando que no son acertadas. Además, el 20% de los participantes los cuales se encuentran muy insatisfechos con los resultados por el equipo de soporte indicando soluciones aún menos efectivas. Por otro lado, el 30% restante reporta sentirse satisfecho con las respuestas brindadas demostrando que pocas contribuciones son adecuadas para los clientes.

Pregunta 6: ¿En qué tiempo es resuelto un problema notificado?

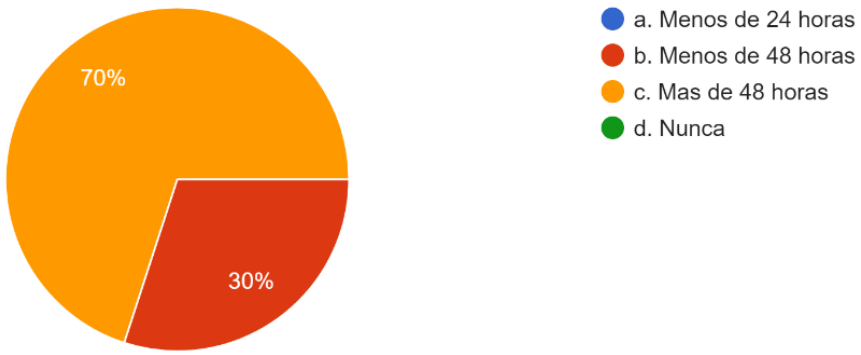


Figura 6. Pregunta 6

Análisis e interpretación de resultados

En la Figura 6 se evidencia que el 70% de los soportes notificados por los participantes son solucionados en más de 48 horas, mientras que el 30% restante son solucionados en menos de 48 horas. Estos datos evidencian que los tickets notificados toman demasiado tiempo en resolverse, lo cual sugiere la necesidad de agilizar el proceso de resolución de las novedades reportadas por los clientes.

Pregunta 7: ¿Todos sus reclamos han sido solucionados?

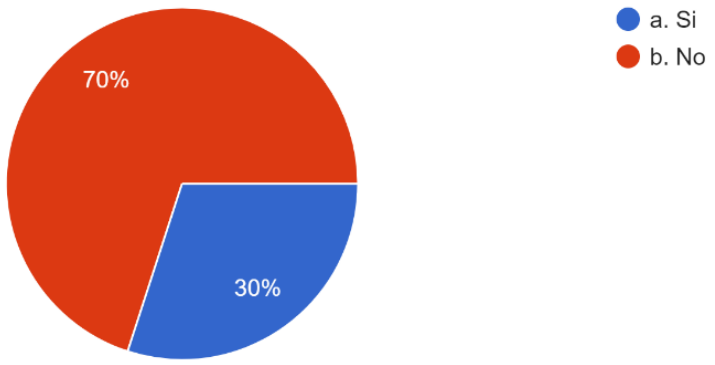


Figura 7. Pregunta 7

Análisis e interpretación de resultados

Como se muestra en la Figura 7, el 70% de los participantes reporta que sus reclamos no han sido resueltos, mientras que el 30% notificó que si fueron resueltos. Los datos recolectados indican que los soportes registrados no son solucionados, lo que a su vez sugiere la necesidad de mejorar el proceso de recepción y resolución de tickets mediante un sistema informático.

Pregunta 8: ¿Cómo califica las soluciones brindadas a sus notificaciones?

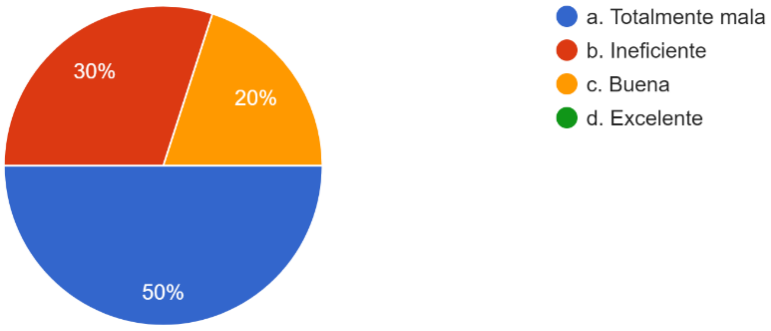


Figura 8. Pregunta 8

Análisis e interpretación de resultados

En la Figura 8 se presenta que el 50% de los participantes posee soluciones infructíferas una vez reportadas sus incidencias mientras que el 30% expresa soluciones ineficientes. Sin embargo, el 20% restante posee soluciones buenas a sus soportes. Los datos evidencian que la forma de manejar soportes necesita ser mejorada abordando los puntos débiles de lo que la provoca.

Pregunta 9: ¿Cuál es su conocimiento sobre el proceso interno que se realiza al comunicar una petición?

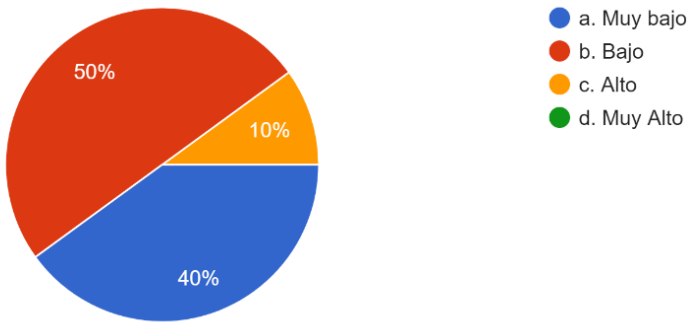


Figura 9. Pregunta 9

Análisis e interpretación de resultados

En la Figura 9 se puede observar que el 50% de los participantes no poseen ningún conocimiento sobre el proceso interno al momento de solucionar un soporte, mientras que el 40% posee conocimiento mínimo sobre el proceso de solución de un ticket. Sin embargo, el 10% conoce el proceso que se lleva a cabo al solucionar una petición. Los datos recolectados muestran que la empresa no se comunica eficientemente con los clientes en el proceso de resolución de un problema, por lo que un sistema gestor de asistencias técnicas permitiría verificar en tiempo real como se aborda un ticket.

Pregunta 10: ¿Estaría de acuerdo con el desarrollo de un sistema informático que permita ingresar sus propios tickets y que estos sean gestionados de forma adecuada?

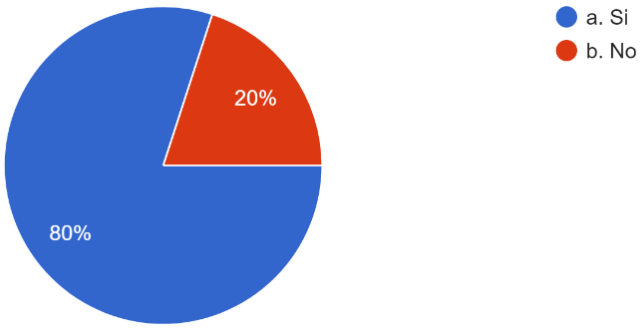


Figura 10. Pregunta 10

Análisis e interpretación de resultados

En la Figura 10, el 80% está de acuerdo con la implementación de un sistema informático para la gestión de sus inconvenientes mientras que el 20% no está de acuerdo con la implementación del sistema. El porcentaje que se muestra es un indicativo de la aceptación por parte de los clientes al desarrollo de un sistema que permita ingresar y administras los tickets ingresados.

2.2.6 Procesamiento y análisis de datos

Para el análisis estadístico de datos obtenidos mediante la encuesta a los clientes de la empresa se procedió a representar los valores mediante el Figura11. En cuanto a los procedimientos obtenidos de la empresa mediante las fichas de observación son representados en la Figura 1.

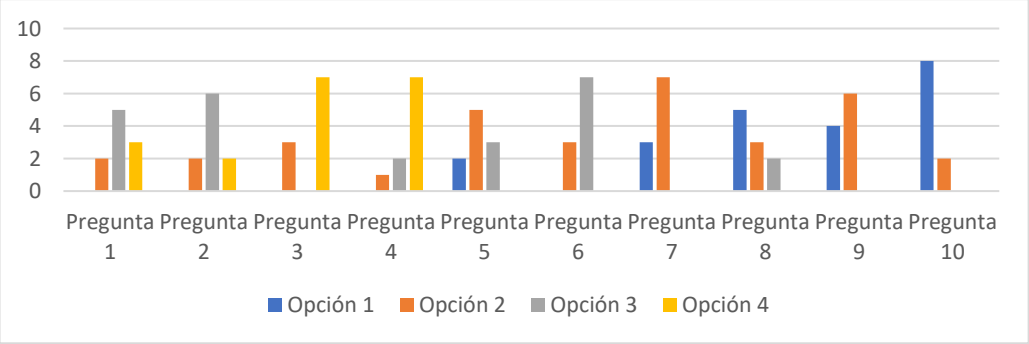


Figura 11. Estadísticas de respuestas de la encuesta

En los datos recopilados en la Figura 11 se puede evidenciar la frecuencia de las respuestas obtenidas denotando los siguientes resultados:

- La mayoría de los registros de los soportes se realiza mayoritariamente en WhatsApp y estos se pierden mientras la cantidad de mensajes aumenta considerablemente en el transcurso de las semanas.

- Las soluciones otorgadas por parte de la empresa no brindan la satisfacción al cliente ya que toman demasiado tiempo en responderse, además de no ser registrados adecuadamente al momento de manejar los soportes.

- La mayoría de los clientes están de acuerdo en la implantación de un sistema informático que les permita verificar sus soportes registrados además de ellos ingresarlos manualmente.

Mediante las fichas de observación se obtuvieron los siguientes pasos para la asistencia técnica de clientes.

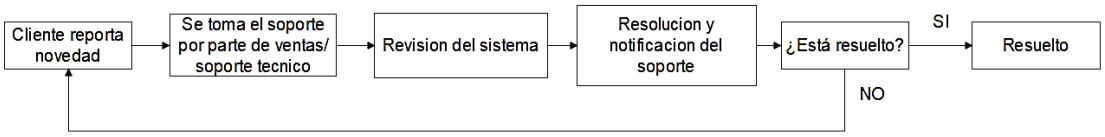


Figura 12. Pasos realizados para asistencia técnica

En base a los datos recolectaos en las fichas de observación se obtuvieron los siguientes puntos:

- El proceso de recibir una novedad puede ser efectuado por parte del departamento de ventas o los programadores, son estos quienes definen el proceso que se lleva a cabo para resolver lo que el cliente reporta.

- Los programadores y el departamento de ventas pueden traspasar el soporte a otro departamento en caso de que no consigan una solución, al momento de resolverse la petición, el soporte es devuelto nuevamente al responsable inicial para notificar al cliente.

- El soporte puede traspasarse las veces que sean necesarias y este se registra en un sistema propio de la empresa, cada que se necesite ser traspasado a otra persona.

- Tanto los clientes como los asistentes técnicos pueden ingresar el ticket de soporte. El ticket al momento de registrarse guarda la persona de quien ingresó y el tipo de sistema utilizado es un sistema Web.

- En el proceso de revisar el sistema de un cliente, en caso de que no se encuentre la causa, se solicita conferencia con el cliente para poder revisar el sistema propio del cliente de forma remota.

CAPÍTULO III. RESULTADOS Y DISCUSIÓN

3.1 Análisis y discusión de los resultados

3.1.1 Proceso identificado para la asistencia técnica

Con los datos obtenidos en el Capítulo 2, se puede expresar el proceso llevado a cabo por los asistentes técnicos al momento de realizar un soporte técnico, este proceso es descrito mediante un diagrama de procesos representado en el **Anexo C**. Es a partir del diagrama de procesos existente con el cual se realiza una optimización del proceso en un nuevo diagrama, el cual se manifiesta en la Figura 13.

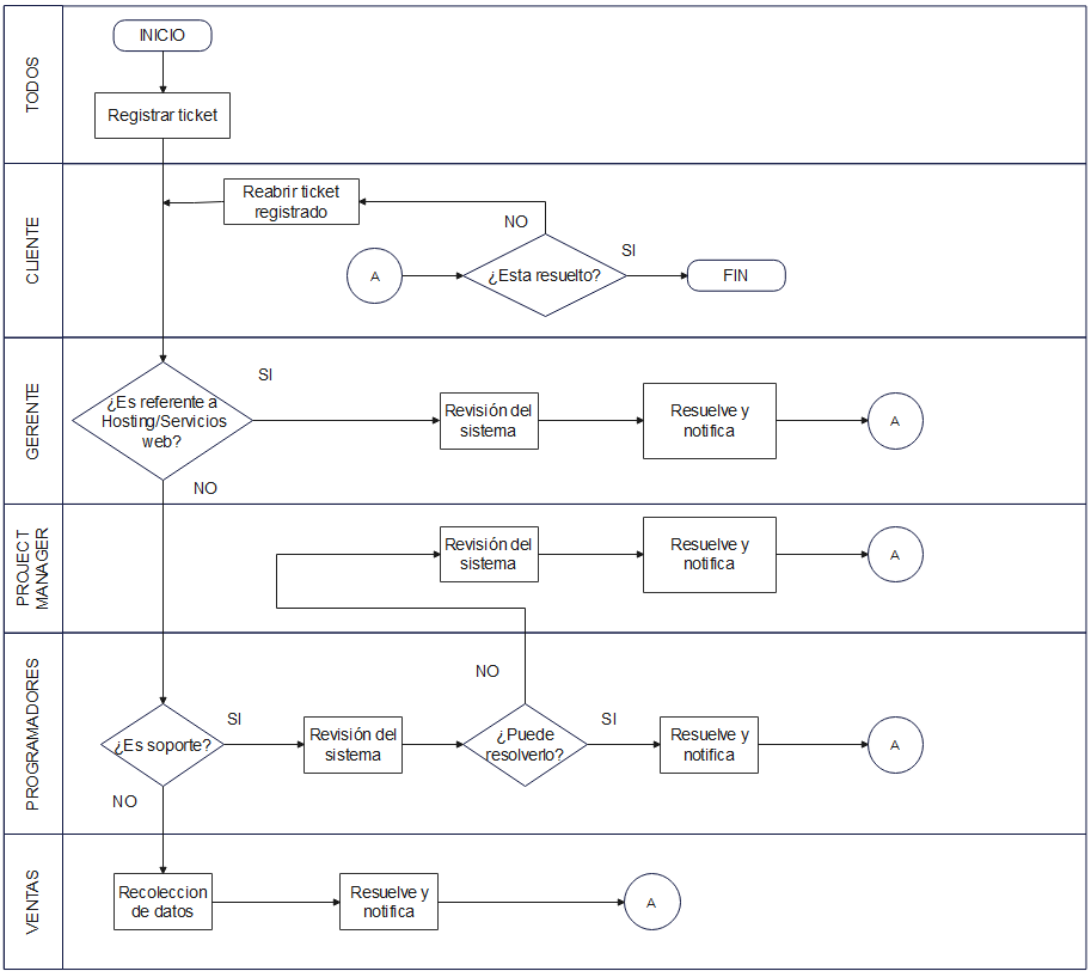


Figura 13. Diagrama de procesos optimizado

a. Descripción de procesos

Tabla 8. Descripción de procesos

N°	Proceso	Descripción
1	Registrar ticket	Registro del ticket en el Sistema de gestión de tickets, se ingresa la información necesaria para su solución como son el registrar archivos (fotos, videos, archivos PDF) de lo que le ocurrió al cliente, tipo de registro (requerimiento, módulo de facturación, página web, firma electrónica) y una descripción.
2	Recolección de datos	Comprobación de los datos recibidos por parte del equipo de ventas (Solo en caso de que el tipo de ticket corresponde al departamento).
3	Revisión del sistema	Se verifica el sistema de un cliente internamente, puede necesitar de revisión remota junto con el cliente en sesión de AnyDesk, llamada por parte del asistente hacia el cliente para clarificar explicaciones, traspasar el soporte para solucionar un inconveniente fuera del alcance del programador.
4	Resuelve y notifica	Notifica al cliente una vez se resuelve el ticket que se registró.
5	Reabrir ticket registrado	El cliente, en el caso de que un ticket no se encuentre solventado, puede aperturar y notificar nuevamente el mismo ticket para que se repita el proceso de revisión.

3.1.2 Elección de herramienta openSource para desarrollo Frontend

La herramienta openSource que se utilizó en el desarrollo Frontend fue Sencha Ext JS debido a que múltiples proyectos de la empresa están desarrollados en ese framework, lo que facilita la implantación de la aplicación con los sistemas existentes en la empresa. También es importante mencionar que Ext JS ofrece componentes responsivos listos para ámbitos de producción, manejo de datos avanzada como son sistemas de data binding y modelos integrados.

Ext JS es un Framework integral con JavaScript que permite crear aplicaciones web y móviles multiplataforma para dispositivos modernos, ofrece un gran catálogo de componentes y herramientas las cuales pueden ser integradas sin presentar problemas de rendimiento o compatibilidad [24].

La Tabla 9 presenta el análisis comparativo de las versiones existentes del Framework con la finalidad de seleccionar la mejor versión que se acople a las necesidades del proyecto.

Tabla 9. Versiones de Sencha Ext JS

Versión	Fecha de lanzamiento	Características	Soporte para plataformas	Arquitectura	Rendimiento	Soporte para Promesas
6.x	06/2015	Fusionó Ext JS y Sencha Touch para aplicaciones móviles y web.	IE9+, Firefox, Chrome, Safari, Edge	Soporta patrón MVVM	Optimización en la carga de componentes	Implementación inicial para soporte de operaciones asíncronas
7.x	06/2019	Mejoras en herramientas de desarrollo y soporte para modern toolkit.	Firefox, Chrome, Safari, Edge	Implementación nativa de MVVM para flexibilidad de desarrollo	Reducción de tiempos de respuesta para aplicativos	Manejo de promesas y gestión de operaciones asíncronas por peticiones Ajax y Fetch
8.x	05/2023	Nuevas características como el soporte para TypeScript.	Firefox, Chrome, Safari, Edge	Mejora para gestión de datos y lógica de presentación	Tiempos de carga reducidos, posee bugs en renderización de componentes nuevos.	Optimización para manejo de promesas mediante Ajax y Fetch

Tras analizar las versiones más recientes disponibles del Framework Ext JS, se eligió la versión 7.x debido a que el rendimiento que ofrece en los navegadores web es superior a versiones pasadas gracias a la implementación nativa de la arquitectura MVVM y el manejo de promesas junto con peticiones asíncronas, optimizando los tiempos de respuesta y carga de los aplicativos realizados con esta herramienta. Además de poseer mayor documentación en comparación con la versión más actual registrada en el mercado. Si bien, la curva de aprendizaje es mayor en comparación con la versión 6.x, en la versión 7.x se ofrece un conjunto de herramientas que benefician el diseño responsive junto con mayor personalización de los componentes.

Los beneficios que otorga el Framework Ext JS para el desarrollo web se ven reflejados en la Tabla 10, donde se muestra a distintas empresas que han utilizado esta herramienta.

Tabla 10. Empresas que utilizan Sencha Ext JS para su desarrollo web

Empresa	**Descripción**	**Uso**
PlayCanvas	Motor de juegos	Interfaz para diseño de juegos
Rambox	Aplicación para comunicaciones	Interfaz de usuario
Keyhole Software	Empresa desarrolladora de software	Desarrollo de aplicaciones web empresariales complejas
OneHat	Empresa desarrolladora de software	Desarrollo de aplicaciones web empresariales complejas

3.1.3 Elección de herramienta openSource para desarrollo Backend

La herramienta openSource que se utilizó en el desarrollo Backend fue Node JS debido a que múltiples proyectos de la empresa están desarrollados en ese framework, lo que facilita la implantación de la aplicación con los sistemas existentes en la empresa.

Node JS es un entorno de ejecución JavaScript multiplataforma de código abierto que permite construir aplicaciones de red rápidas y escalables de forma asíncrona mediante

las bibliotecas nativas que posee. También es ampliamente versátil y eficiente al momento de manejar grandes volúmenes de datos y conexiones simultaneas debido a que sus bibliotecas permiten que el flujo de ejecución de código JavaScript no presente bloqueos ya que toda transferencia de datos y llamadas de funciones externas no detienen la ejecución del programa [25].

La Tabla 11 presenta el análisis comparativo de las versiones existentes del Framework con la finalidad de seleccionar la mejor versión que se acople a las necesidades del proyecto.

Tabla 11. Versiones de Node JS

Versión	Fecha de lanzamiento	Características	Tipo	Variables de entorno	Soporte para JavaScript
20	18/04/2023	Mejor manejo de funciones asíncronas y reducción del tamaño de la aplicación	LTS	Requiere uso de paquetes externos como dotenv para su configuración	Importación de módulos ESM con CommonJS mediante require()
21	17/10/2023	Estabilización de comandos fetch y WebStreams	Current	Introducción de gestión nativa de variables de entorno	Introducción de webSockets integrado
22	24/04/2024	Integración de compiladores web para mayor velocidad de carga	Current	Validación de variables sensibles	Soporte para gráficos ESM y optimización de carga para módulos
23	16/10/2024	Mejoras de rendimiento para herramientas de test	Current	Herramientas de configuración dinámica y segura para aplicaciones.	Interoperabilidad de módulos ESM

La versión seleccionada para el proyecto es Node JS 20 debido a que ofrece manejo de funciones asíncronas para manejo de datos en tiempo real agregando la buena gestión de módulos para JavaScript y el uso de variables de entorno que permiten un buen rendimiento y la documentación que posee junto con la característica de mantenimiento a largo plazo por parte de la comunidad.

Los beneficios que otorga el Framework Node JS para el desarrollo web se ven reflejados en la Tabla 12, donde se muestra a distintas empresas que han utilizado esta herramienta.

Tabla 12. Empresas que utilizan Node JS en desarrollo web

Empresa	**Descripción**	**Uso**
eBay	Marketplace	Optimización de navegación y búsqueda
Uber	Servicio de transporte en linea	Manejo de solicitudes de viaje en tiempo real
PayPal	Servicio de pagos en línea	Agilizar procesos de transacción
LinkedIn	Red Social enfocada a los negocios	Carga eficiente de contenido dinámico y mensajería en tiempo real
Netflix	Streaming de películas	Velocidad de arranque de la aplicación

3.1.4 Elección del gestor para Base de datos openSource

El gestor de base de datos openSource que se utilizó para el almacenamiento de datos fue MySQL debido a que múltiples proyectos de la empresa están desarrollados bajo la utilización de dicho gestor, lo que facilita la implantación de la aplicación con los sistemas existentes en la empresa.

MySQL es un sistema gestor de base de datos relacional de código abierto desarrollado por Oracle. Se caracteriza por su amplia capacidad de almacenar y recuperar datos de forma eficiente, siendo ampliamente utilizado en aplicaciones web empresariales dado su fiabilidad y escalabilidad [26].

Para seleccionar la versión más adecuada para el proyecto, se realizó una comparativa de versiones disponibles bajo los criterios que se consideran necesarios para el proyecto, mostrados en la Tabla 13.

Tabla 13.Versiones de MySQL

Versión	Fecha de lanzamiento	JSON y Manejo de datos	Seguridad	Particiones	Rendimiento	Replicación	Soporte
MySQL 5.6	02/2013	No posee soporte nativo para JSON, se maneja mediante TEXT	Autenticación mediante cifrado SSL/TLS	Particionamiento por rangos y hash	Índices de memoría y optimización de consultas	GTID (Global Transaction Identifiers)	No posee
MySQL 5.7	10/2015	Soporte nativo para JSON, permitiendo manipulación y almacenamiento directo.	Roles de usuario dinámicos	Soporte para tablas particionadas	índices de consulta, consultas JSON	Replicación mediante binlog grupal y menor latencia con GTID	No posee
MySQL 8.0	04/2018	Funciones para búsqueda avanzada y manipulación de datos optimizada.	Autenticación basada en SHA-256, inclusión de roles de usuario y encriptación de tablas	Soporte para consultas sobre tablas particionadas	Optimización de índices, persistencia en memoria caché y paralelización de consultas	Replicación mediante transacciones y soporte para múltiples servidores	Si posee

Tras analizar las características técnicas que ofrece las distintas versiones de MySQL, se seleccionó trabajar con la versión 8.0 debido a que ofrece excelente manejo de datos de alto volumen, en donde pueden almacenarse imágenes o documentos, además, posee soporte para lectura de datos JSON y seguridad mejorada con roles y encriptación de tablas, es beneficioso para el proyecto ya que la velocidad de lectura y escritura que posee es ampliamente superior comparado a versiones anteriores, además de poseer soporte activo.

En la Tabla 14 se reflejan empresas cuyos casos de éxito se han manifestado debido a la integración de MySQL a sus proyectos, donde también podemos visualizar las ventajas que adquirimos al momento de aplicar el gestor en el proyecto.

Tabla 14. Empresas que usan MySQL

Empresa	**Descripción**	**Uso**
Zendesk	SaaS	Manejo de alto volumen de datos (Gestión de tickets)
Facebook	Red social	Procesamiento de transacciones en línea
Twitter (X)	Red social	Mejora de escalabilidad y rendimiento junto con almacenamiento de datos críticos
Uber	Servicio de transporte en línea	Manejo de alto volumen de datos y escala de operaciones
HubSpot	SaaS	Almacenamiento e integración de datos

3.1.5 Metodologías de desarrollo para aplicaciones web

Las metodologías de desarrollo ágil son esenciales al momento de organizar proyectos, permiten implementar cambios de forma continua sin interrumpir el flujo de trabajo y reducen el tiempo de entrega en base a una serie de marcos de trabajo. Es necesario el implementar las metodologías agiles en un mercado que se encuentra en constante cambio. Es entonces que, en la **Tabla 15** se muestra un cuadro comparativo el cual ayudó en la selección de la metodología que mejor se acopló en el desarrollo del proyecto.

Tabla 15. Comparativa de metodologías de desarrollo ágil

	XP	Scrum	Lean
Tamaño del equipo	Menores a 10 personas	Múltiples equipos menores de 10 personas	Variable (Depende del proyecto)
Tamaño del proyecto	Pequeños, medianos	Pequeños, medianos, grandes	Pequeños, medianos, grandes
Complejidad	Baja	Alta	Media
Metodología	Retroalimentación continua con el cliente	Sprints	Flujo de trabajo, no posee fases
Roles	- Programador - Cliente - Tester - Gestor	- Cliente o Stakeholder - Scrum Master - Programadores - Product Owner	- Gerente del proyecto - Programadores - Equipo de soporte
Entregas	Dependiendo del criterio del tester, cada iteración es puesta en producción	Al finalizar un Sprint, se entrega el avance del proyecto al Product Owner.	Entrega frecuente de los avances del proyecto en base a métricas definidas
Fases	- Exploración - Planificación - Diseño - Codificación - Pruebas	Sprint con objetivo específico, cada sprint posee 5 fases: - Planificación del Sprint - Desarrollo - Scrum Diario (Reunión diaria) - Revisión del Sprint - Retrospectiva del Sprint	No posee fases predefinidas
Comunicación	Presencia del cliente/usuario en todo momento	Reunión diaria del equipo	Presencia del cliente/usuario continuo

La metodología que se seleccionó para el desarrollo del proyecto fue XP (Extreme Programing) debido a que al ser una metodología menos compleja de aplicar y estar enfocada hacia proyectos pequeños, fue la más adecuada de entre las metodologías más utilizadas. XP ofrece la flexibilidad necesaria para la implementación de cambios inesperados, además de, que el ciclo de desarrollo que presenta permite realizar ajustes rápidos y pruebas continuas garantizando la calidad del producto.

Fases de la metodología XP

1. **Exploración:** Se recopilan los requisitos para la realización del proyecto y se exploran múltiples soluciones, permitiendo identificar los objetivos iniciales,

definir el alcance del proyecto y las iteraciones del mismo. También define traducir los requisitos en tareas específicas y recopilar las necesidades del cliente.

2. **Planificación del proyecto:** Es la fase en la cual se identifican los requisitos del proyecto, se definen priorizaciones según las necesidades del cliente mediante las historias de usuario, lo que permite guiar el desarrollo y establecer tiempos de desarrollo aproximados.

3. **Diseño:** Es la fase en la cual se realiza los bosquejos del sistema. Se opta por realizar diseños simples que cumplan con los requisitos permitiendo una modificación fácil en caso de que estos cambien. El objetivo de esta fase es el poder identificar los riesgos y refactorizar la estructura del proyecto de ser necesario.

4. **Codificación:** Es la fase en donde los desarrolladores trabajan en pares para la escritura del código, facilitando la revisión mutua y mejorando la calidad del producto. El cliente se encuentra presente en el proceso para que al realizar las pruebas, se puedan sugerir cambios en base a lo codificado.

5. **Pruebas:** Es la fase donde se realizan los test del software donde el cliente otorga la retroalimentación de cada iteración culminada.

3.2 Desarrollo de la propuesta

3.2.1 Fase 1: Exploración

En la fase de Exploración se recopiló los datos necesarios para realizar las historias de usuario, los cuales definen los requerimientos del producto para realizar su primera entrega. También se familiariza con las herramientas, tecnologías y prácticas que se utilizaron para el desarrollo del proyecto.

Con los datos recopilados en el Capítulo II y el diagrama de procesos optimizado realizado en la Figura 13, se identificaron tres roles asociados en el proceso del soporte técnico, los cuales se muestran en la Tabla 16.

Tabla 16. Roles del sistema

Rol	**Acciones**
Administradores	- Visualizar tickets (Todos) - Crear ticket - Editar ticket - Traspasar ticket - Cerrar ticket - Visualizar tutoriales - Agregar, eliminar o editar técnicos - Agregar, eliminar o editar usuarios
Técnicos	- Visualizar tickets (Todos) - Crear ticket - Editar ticket - Traspasar ticket - Cerrar ticket - Visualizar tutoriales
Usuarios	- Visualizar tickets (Por Usuario) - Crear ticket - Reaperturar ticket - Visualizar tutoriales

a. Historias de usuario

Una historia de usuario es un requerimiento hecho por el cliente, el cual se redacta de forma corta y no técnica, sobre lo que el cliente necesita que realice el sistema. Su principal objetivo es el ayudar a establecer los test de aceptación ayudando a verificar la correcta implementación de cada entrega.

Las historias de usuario fueron realizadas con los requerimientos identificados mediante los datos obtenidos, las cuales se muestran a continuación.

Tabla 17. Historia de Usuario 1 – Acceso al sistema

HISTORIA DE USUARIO		**#1**
Usuario: Todos		
Nombre: Acceso al sistema		
Prioridad en el negocio: Alta	**Riesgo de desarrollo:** Alta	
Puntos estimados: 1	**Iteración asignada: 1**	
Programador a cargo: Patricio Moscoso		
Descripción: El usuario podrá acceder al sistema con sus credenciales.		
Observación:		

Tabla 18. Historia de Usuario 2 – Recuperar contraseña

HISTORIA DE USUARIO	**#2**
Usuario: Todos	
Nombre: Recuperar contraseña	
Prioridad en el negocio: Alta	**Riesgo de desarrollo:** Alta
Puntos estimados: 1	**Iteración asignada: 1**
Programador a cargo: Patricio Moscoso	
Descripción: Todos pueden recuperar la contraseña.	
Observación: Deben tener sus anteriores credenciales para poder solicitar su cambio de contraseña	

Tabla 19. Historia de Usuario 3 – Administración de técnicos

HISTORIA DE USUARIO	**#3**
Usuario: Administrador	
Nombre: Administración de técnicos	
Prioridad en el negocio: Alta	**Riesgo de desarrollo:** Alta
Puntos estimados: 1	**Iteración asignada: 1**
Programador a cargo: Patricio Moscoso	
Descripción: El usuario administrador podrá crear, modificar y eliminar a otros técnicos.	
Observación:	

Tabla 20. Historia de Usuario 4 – Administrador de usuarios

HISTORIA DE USUARIO	**#4**
Usuario: Administrador	
Nombre: Administración de usuarios	
Prioridad en el negocio: Alta	**Riesgo de desarrollo:** Alta
Puntos estimados: 1	**Iteración asignada: 1**
Programador a cargo: Patricio Moscoso	
Descripción: El usuario administrador podrá crear, modificar y eliminar a otros usuarios.	
Observación:	

Tabla 21. Historia de Usuario 5 – Administración de roles

HISTORIA DE USUARIO	**#5**
Usuario: Administrador	
Nombre: Administración de roles	
Prioridad en el negocio: Alta	**Riesgo de desarrollo:** Alta
Puntos estimados: 1	**Iteración asignada:** 1
Programador a cargo: Patricio Moscoso	

Descripción: El usuario administrador podrá crear, modificar y deshabilitar roles de usuarios.
Observación:

Tabla 22. Historia de Usuario 6 – Administración de departamentos

HISTORIA DE USUARIO	**#6**
Usuario: Administrador	
Nombre: Administración de departamentos	
Prioridad en el negocio: Alta	**Riesgo de desarrollo:** Alta
Puntos estimados: 1	**Iteración asignada: 2**
Programador a cargo: Patricio Moscoso	
Descripción: El usuario administrador podrá crear, modificar y deshabilitar los departamentos asignados de usuarios.	
Observación:	

Tabla 23. Historia de Usuario 7 – Administración de planes

HISTORIA DE USUARIO	**#7**
Usuario: Administrador	
Nombre: Administración de planes	
Prioridad en el negocio: Alta	**Riesgo de desarrollo:** Alta
Puntos estimados: 1	**Iteración asignada: 2**
Programador a cargo: Patricio Moscoso	
Descripción: El usuario administrador podrá crear, modificar y deshabilitar los planes asignados a sistemas.	
Observación:	

Tabla 24. Historia de Usuario 8 – Administración de sistemas

HISTORIA DE USUARIO	**#8**
Usuario: Administrador	
Nombre: Administración de sistemas	
Prioridad en el negocio: Alta	**Riesgo de desarrollo:** Alta
Puntos estimados: 1	**Iteración asignada: 2**
Programador a cargo: Patricio Moscoso	
Descripción: El usuario administrador podrá crear, modificar y eliminar los sistemas o servicios existentes en la empresa.	
Observación:	

Tabla 25. Historia de Usuario 9 – Cerrar Sesión

HISTORIA DE USUARIO	**#9**
Usuario: Todos	
Nombre: Cerrar Sesión	
Prioridad en el negocio: Alta	**Riesgo de desarrollo:** Media
Puntos estimados: 1	**Iteración asignada: 2**
Programador a cargo: Patricio Moscoso	

Descripción: Todos podrán cerrar sesión desde el menú principal.
Observación:

Tabla 26. Historia de Usuario 10 – Administración de estados para tickets

HISTORIA DE USUARIO	**#10**
Usuario: Administrador	
Nombre: Administración de estados para tickets	
Prioridad en el negocio: Alta	**Riesgo de desarrollo:** Media
Puntos estimados: 1	**Iteración asignada: 3**
Programador a cargo: Patricio Moscoso	
Descripción: El usuario administrador podrá crear, modificar y deshabilitar los estados asignables para los tickets.	
Observación:	

Tabla 27. Historia de Usuario 11 – Administración de prioridades para tickets

HISTORIA DE USUARIO	**#11**
Usuario: Administrador	
Nombre: Administración de prioridades para tickets	
Prioridad en el negocio: Alta	**Riesgo de desarrollo:** Media
Puntos estimados: 1	**Iteración asignada: 3**
Programador a cargo: Patricio Moscoso	
Descripción: El usuario administrador podrá crear, modificar y deshabilitar las prioridades asignables para los tickets.	
Observación:	

Tabla 28. Historia de Usuario 12 – Crear ticket

HISTORIA DE USUARIO	**#12**
Usuario: Todos	
Nombre: Crear ticket	
Prioridad en el negocio: Alta	**Riesgo de desarrollo:** Alta
Puntos estimados: 2	**Iteración asignada: 3**
Programador a cargo: Patricio Moscoso	
Descripción: Todos podrán crear un ticket para relatar una incidencia.	
Observación: Al momento de crear un ticket, se asigna un estado y se debe autoasignar a un miembro de la empresa automáticamente	

Tabla 29. Historia de Usuario 13 – Editar ticket

HISTORIA DE USUARIO	**#13**
Usuario: Técnicos, Administrador	
Nombre: Editar ticket	
Prioridad en el negocio: Alta	**Riesgo de desarrollo:** Alta
Puntos estimados: 2	**Iteración asignada: 3**
Programador a cargo: Patricio Moscoso	
Descripción: Todos los miembros de la empresa pueden editar un ticket.	

Observación: El ticket tiene que cambiar de estado tras realizar una interacción, ya sea por parte del técnico o del cliente.

Tabla 30. Historia de Usuario 14 – Traspasar ticket

HISTORIA DE USUARIO	**#14**
Usuario: Técnicos, Administrador	
Nombre: Traspasar ticket	
Prioridad en el negocio: Alta	**Riesgo de desarrollo:** Alta
Puntos estimados: 2	**Iteración asignada: 4**
Programador a cargo: Patricio Moscoso	
Descripción: Todos los miembros de la empresa pueden traspasar un ticket a otro miembro de la empresa.	
Observación: El ticket puede traspasarse las veces que sean necesarias, no posee un límite de traspasos.	

Tabla 31. Historia de Usuario 15 – Cerrar ticket

HISTORIA DE USUARIO	**#15**
Usuario: Técnicos, Administrador	
Nombre: Cerrar ticket	
Prioridad en el negocio: Alta	**Riesgo de desarrollo:** Alta
Puntos estimados: 2	**Iteración asignada: 4**
Programador a cargo: Patricio Moscoso	
Descripción: Todos los miembros de la empresa pueden finalizar un ticket.	
Observación:	

Tabla 32. Historia de Usuario 16 – Reaperturar ticket

HISTORIA DE USUARIO	**#16**
Usuario: Usuarios	
Nombre: Reaperturar ticket	
Prioridad en el negocio: Alta	**Riesgo de desarrollo:** Alta
Puntos estimados: 2	**Iteración asignada: 4**
Programador a cargo: Patricio Moscoso	
Descripción: Solo el usuario puede reabrir un ticket	
Observación: El ticket tiene que estar finalizado para poder ser reaperturado.	

Tabla 33. Historia de Usuario 17 – Cambio de estado del ticket

HISTORIA DE USUARIO	**#17**
Usuario: --	
Nombre: Cambio de estado del ticket	
Prioridad en el negocio: Alta	**Riesgo de desarrollo:** Alta
Puntos estimados: 2	**Iteración asignada: 4**
Programador a cargo: Patricio Moscoso	

Descripción: El ticket cambia de estado por cada respuesta que se efectué.
Observación: El ticket tiene que estar aperturado para realizar cualquier cambio.

Tabla 34. Historia de Usuario 18 – Visualización de tickets

HISTORIA DE USUARIO	**#18**
Usuario: Todos	
Nombre: Visualización de tickets	
Prioridad en el negocio: Alta	**Riesgo de desarrollo:** Alta
Puntos estimados: 1	**Iteración asignada: 5**
Programador a cargo: Patricio Moscoso	
Descripción: Pantalla para visualizar el detalle del ticket	
Observación:	

Tabla 35. Historia de Usuario 19 – Mensajería instantánea para chats

HISTORIA DE USUARIO	**#19**
Usuario: Todos	
Nombre: Mensajería instantánea para chats	
Prioridad en el negocio: Alta	**Riesgo de desarrollo:** Alta
Puntos estimados: 2	**Iteración asignada: 5**
Programador a cargo: Patricio Moscoso	
Descripción: Cada ticket tendrá su propio chat en donde podrán comunicarse cliente y técnico.	
Observación:	

Tabla 36. Historia de Usuario 20 – Notificación de tickets

HISTORIA DE USUARIO	**#20**
Usuario: --	
Nombre: Notificación de tickets	
Prioridad en el negocio: Alta	**Riesgo de desarrollo:** Alta
Puntos estimados: 5	**Iteración asignada: 5**
Programador a cargo: Patricio Moscoso	
Descripción: El sistema notificará tanto a clientes como miembros de la empresa cada que se efectúe una respuesta.	
Observación:	

Tabla 37. Historia de Usuario 21 – Tutoriales

HISTORIA DE USUARIO	**#21**
Usuario: Todos	
Nombre: Tutoriales	
Prioridad en el negocio: Alta	**Riesgo de desarrollo:** Alta
Puntos estimados: 3	**Iteración asignada: 5**
Programador a cargo: Patricio Moscoso	
Descripción: Visualización de tutoriales para capacitar a los usuarios y personal en el aplicativo.	

Observación:

Tabla 38. Historia de Usuario 22 – Reportes

HISTORIA DE USUARIO		**#22**
Usuario: Todos		
Nombre: Reportes		
Prioridad en el negocio: Alta	**Riesgo de desarrollo:** Alta	
Puntos estimados: 3	**Iteración asignada: 5**	
Programador a cargo: Patricio Moscoso		
Descripción: Todos podrán generar reportes filtrados por fechas.		
Observación:		

3.2.2 Fase 2: Planificación del proyecto

a. Estimación de Historias de Usuario

Tras la obtención de las historias de usuario y los roles definidos, se procedió a la realizar la planificación del proyecto mediante la estimación de las historias de usuario los cuales se presentan en la Tabla 32. Se definieron 8 horas de trabajo diario destinados al desarrollo del sistema, definiendo los requerimientos que más tiempo tomaron.

Tabla 39. Estimación de Historias de Usuario

N°	**Historia de usuario**	**Duración**	
		Días	**Horas**
	Base de datos	3	24
1	Acceso al sistema	1	8
2	Recuperar contraseña	1	8
3	Administración de técnicos	1	8
4	Administración de usuarios	1	8
5	Administración de roles	1	8
6	Administración de departamentos	1	8
7	Administración de planes	1	8
8	Administración de sistemas	1	8
9	Cerrar Sesión	1	8
10	Administración de estados para tickets	1	8
11	Administración de prioridades para tickets	1	8

12	Crear ticket	2	16
13	Editar ticket	2	16
14	Traspasar ticket	2	16
15	Cerrar ticket	2	16
N°	**Historia de usuario**	**Días**	**Horas**
16	Reaperturar ticket	2	16
17	Cambio de estado del ticket	2	16
18	Visualización de tickets	1	8
19	Mensajeria instantanea para chats	2	16
20	Notificación de tickets	5	40
21	Tutoriales	3	24
22	Reportes	3	24
Total		40	320

b. Plan de Entrega

Una vez definido la estimación de historias de usuario, se realiza el plan de entrega. El objetivo es definir el tiempo estimado del proyecto necesario para su desarrollo. Los tiempos varían de acuerdo con el cliente y el desarrollador. El tiempo es definido en horas por días laborables.

Tabla 40. Plan de Entrega

Iteración	**N°**	**Historia de usuario**	**Duración**		
			Semanas	**Días**	**Horas**
1		Base de datos	0,6	3	24
	1	Acceso al sistema	0,2	1	8
	2	Recuperar contraseña	0,2	1	8
	3	Administración de técnicos	0,2	1	8
	4	Administración de usuarios	0,2	1	8
	5	Administración de roles	0,2	1	8
2	6	Administración de departamentos	0,2	1	8
	7	Administración de planes	0,2	1	8
	8	Administración de sistemas	0,2	1	8
	9	Cerrar Sesión	0,2	1	8
3	10	Administración de estados para tickets	0,2	1	8
	11	Administración de prioridades para tickets	0,2	1	8
	12	Crear ticket	0,4	2	16
	13	Editar ticket	0,4	2	16
4	14	Traspasar ticket	0,4	2	16
	15	Cerrar ticket	0,4	2	16
	16	Reaperturar ticket	0,4	2	16
	17	Cambio de estado del ticket	0,4	2	16
5	18	Visualización de tickets	0,2	1	8

	19	Mensajeria instantanea para chats	0,4	2	16
	20	Notificación de tickets	1	5	40
	21	Tutoriales	0,6	3	24
	22	Reportes	0,6	3	24
Total			8	40	320

c. Equipo de trabajo

Tabla 41. Equipo de trabajo

Roles	Miembros
Programador	Patricio Moscoso
Cliente	Empleados de la empresa ConnectaServices
Tester	Ing. Patricio Constante
Coach	Ing. Franklin Mayorga
Seguimiento	Ing. Franklin Mayorga Patricio Moscoso

3.2.3 Fase 3: Diseño

a. Tarjetas CRC

Las tarjetas CRC (Clase-Responsabilidad-Colaboración) permiten organizar el diseño de software a través de representar clases, responsabilidades y colaboraciones dentro del sistema en pequeñas tarjetas [27].

Por ende, se elaboraron las tarjetas CRC en base a las historias de usuario recopiladas en la Fase 1.

Tabla 42. Acceso al sistema

ACCESO AL SISTEMA	
Responsabilidad	**Colaboradores**
- Obtener usuario y contraseña - Validar usuario y contraseña	- Capa de datos - Capa de manejo de datos - Capa de presentación
Observación:	

Tabla 43. Recuperar contraseña

RECUPERAR CONTRASEÑA

Responsabilidad	Colaboradores
- Obtener y validar usuario - Validar nueva contraseña - Guardar nueva contraseña	- Capa de datos - Capa de manejo de datos - Capa de presentación
Observación:	

Tabla 44. Administración de técnicos

ADMINISTRACION DE TÉCNICOS	
Responsabilidad	**Colaboradores**
- Obtener parámetros - Guardar parámetros	- Capa de datos - Capa de manejo de datos - Capa de presentación
Observación:	

Tabla 45. Administración de usuarios

ADMINISTRACION DE USUARIOS	
Responsabilidad	**Colaboradores**
- Obtener parámetros - Guardar parámetros	- Capa de datos - Capa de manejo de datos - Capa de presentación
Observación:	

Tabla 46. Administración de roles

ADMINISTRACION DE ROLES	
Responsabilidad	**Colaboradores**
- Obtener parámetros - Guardar parámetros	- Capa de datos - Capa de manejo de datos - Capa de presentación
Observación:	

Tabla 47. Administración de departamentos

ADMINISTRACION DE DEPARTAMENTOS	
Responsabilidad	**Colaboradores**
- Obtener parámetros - Guardar parámetros	- Capa de datos - Capa de manejo de datos - Capa de presentación
Observación:	

Tabla 48. Administración de planes

ADMINISTRACION DE PLANES	
Responsabilidad	**Colaboradores**
- Obtener parámetros - Guardar parámetros	- Capa de datos - Capa de manejo de datos - Capa de presentación
Observación:	

Tabla 49. Administración de sistemas

ADMINISTRACION DE SISTEMAS	
Responsabilidad	**Colaboradores**
- Obtener parámetros	- Capa de datos

- Guardar parámetros	- Capa de manejo de datos - Capa de presentación
Observación:	

Tabla 50. Cerrar sesión

CERRAR SESION	
Responsabilidad	**Colaboradores**
- Eliminar credenciales	- Capa de manejo de datos - Capa de presentación
Observación:	

Tabla 51. Administración de estados para tickets

ADMINISTRACIÓN DE ESTADOS PARA TICKETS	
Responsabilidad	**Colaboradores**
- Obtener parámetros - Validar parámetros - Guardar parámetros	- Capa de datos - Capa de manejo de datos - Capa de presentación
Observación:	

Tabla 52. Administración de prioridades para tickets

ADMINISTRACIÓN DE PRIORIDADES PARA TICKETS	
Responsabilidad	**Colaboradores**
- Obtener parámetros - Validar parámetros - Guardar parámetros	- Capa de datos - Capa de manejo de datos - Capa de presentación
Observación:	

Tabla 53. Crear ticket

CREAR TICKET	
Responsabilidad	**Colaboradores**
- Obtener parámetros - Validar parámetros - Guardar parámetros	- Capa de datos - Capa de manejo de datos - Capa de presentación
Observación:	

Tabla 54. Editar ticket

EDITAR TICKET	
Responsabilidad	**Colaboradores**
- Obtener parámetros - Validar parámetros - Guardar parámetros	- Capa de datos - Capa de manejo de datos - Capa de presentación
Observación:	

Tabla 55. Traspasar ticket

TRASPASAR TICKET	
Responsabilidad	**Colaboradores**
- Obtener parámetros - Guardar parámetros	- Capa de datos - Capa de manejo de datos - Capa de presentación
Observación:	

Tabla 56. Cerrar ticket

CERRAR TICKET	
Responsabilidad	**Colaboradores**
- Obtener parámetros - Guardar parámetros	- Capa de datos - Capa de manejo de datos - Capa de presentación
Observación:	

Tabla 57. Reaperturar ticket

REAPERTURAR TICKET	
Responsabilidad	**Colaboradores**
- Obtener parámetros - Validar parámetros - Guardar parámetros	- Capa de datos - Capa de manejo de datos - Capa de presentación
Observación:	

Tabla 58. Cambio de estado del ticket

CAMBIO DE ESTADO DEL TICKET	
Responsabilidad	**Colaboradores**
- Obtener parámetros - Validar parámetros - Guardar parámetros	- Capa de datos - Capa de manejo de datos
Observación:	

Tabla 59. Visualización de tickets

VISUALIZACION DE TICKETS	
Responsabilidad	**Colaboradores**
- Obtener parámetros - Mostrar parámetros	- Capa de datos - Capa de manejo de datos - Capa de presentación
Observación:	

Tabla 60. Mensajería instantánea para chats

MENSAJERÍA INSTANTÁNEA PARA CHATS	
Responsabilidad	**Colaboradores**
- Obtener parámetros - Crear mensaje - Enviar mensaje	- Capa de datos - Capa de manejo de datos - Capa de presentación
Observación:	

Tabla 61. Notificación de tickets

NOTIFICACIÓN DE TICKETS	
Responsabilidad	**Colaboradores**

- Obtener parámetros - Enviar notificación por mail	- Capa de datos - Capa de manejo de datos - Capa de presentación
Observación:	

Tabla 62. Tutoriales

TUTORIALES	
Responsabilidad	**Colaboradores**
- Mostrar manuales	- Capa de presentación
Observación:	

Tabla 63. Reportes

REPORTES	
Responsabilidad	**Colaboradores**
- Obtener parámetros - Validar parámetros - Generar reporte	- Capa de datos - Capa de manejo de datos - Capa de presentación
Observación:	

b. Diseño de la base de datos

La Figura 14 muestra la estructura de la base de datos final utilizada para el desarrollo del aplicativo web.

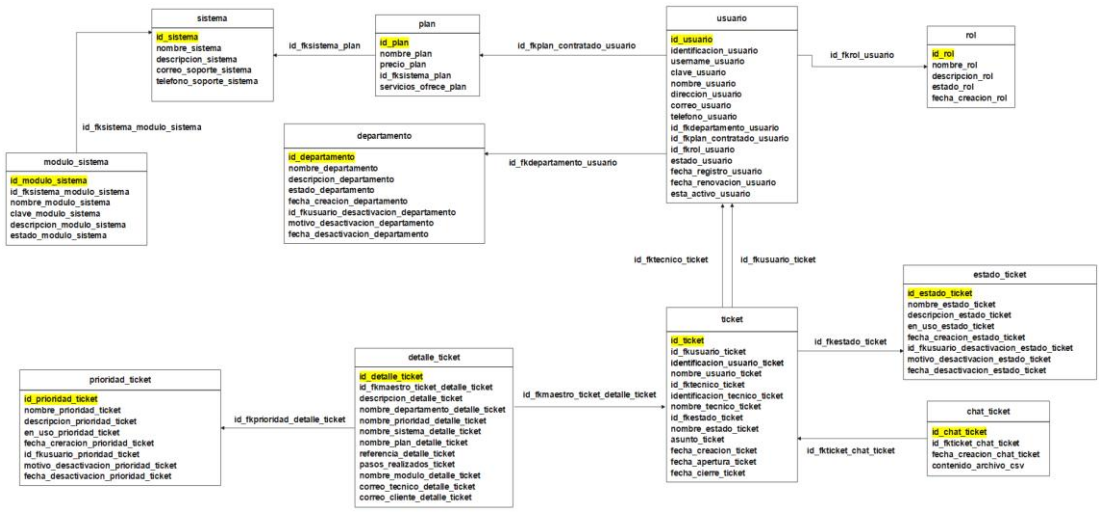


Figura 14. Modelo Entidad Relación

c. Diseño de la interacción

El diseño del prototipo se efectuó en la herramienta gratuita Wondershare EdrawMax debido a que posee una versión gratuita de acceso libre para los usuarios, permitiendo un límite de 60 figuras por ilustración, lo cual fue suficiente para realizar el bosquejo [28].

La Figura 14 muestra una visión del acceso al sistema destinado a todos los roles, mientras que la Figura 15 presenta la pantalla principal el cual tiene el objetivo de presentar los datos más relevantes del usuario en la parte superior junto con las estadísticas de los tickets que posee en la ventana llamado “CONTENIDO”.

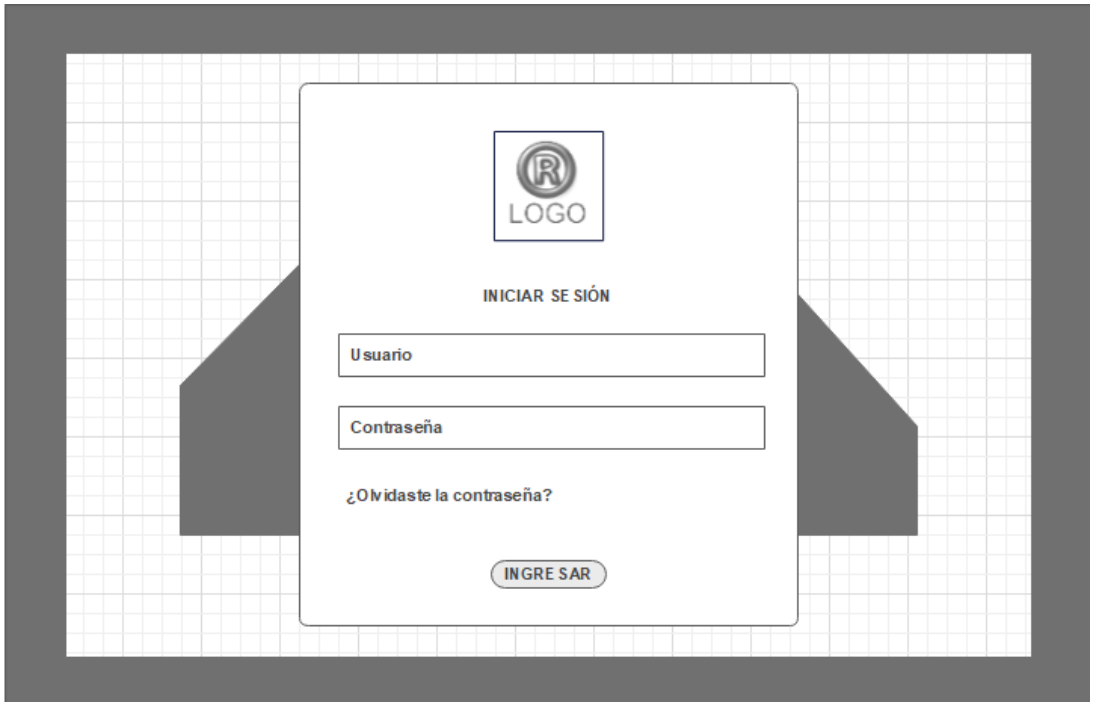


Figura 15. Acceso al sistema



Figura 16. Pantalla principal

En la ventana “CONTENIDO” se muestran las múltiples interfaces de los distintos módulos, los cuales se dividen en dos tipos: módulos administrativos y modulo inicio. Los módulos administrativos serán visibles solo para usuarios Administrativos donde podrán realizar acciones solo a usuarios y técnicos. El módulo inicio será la visualización del dashboard, donde se mostrará la información relevante de los tickets realizados y los botones para realizar acciones rápidas.

En la Figura 17 se muestra el diseño para todos los módulos administrativos, el único módulo administrativo disponible para todos los roles es el módulo de tickets.

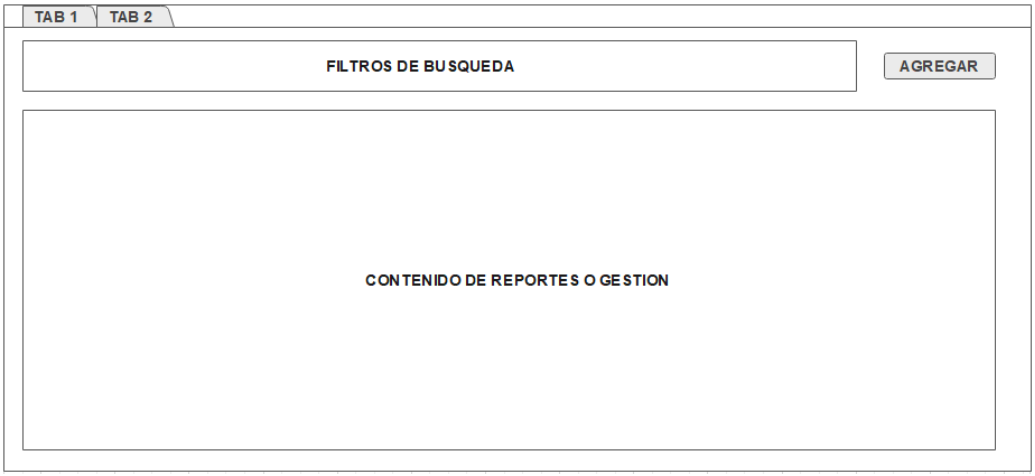


Figura 17. Módulos administrativos

La Figura 18 se muestra el diseño para el módulo de inicio, el cual será visible tras acceder al sistema exitosamente.

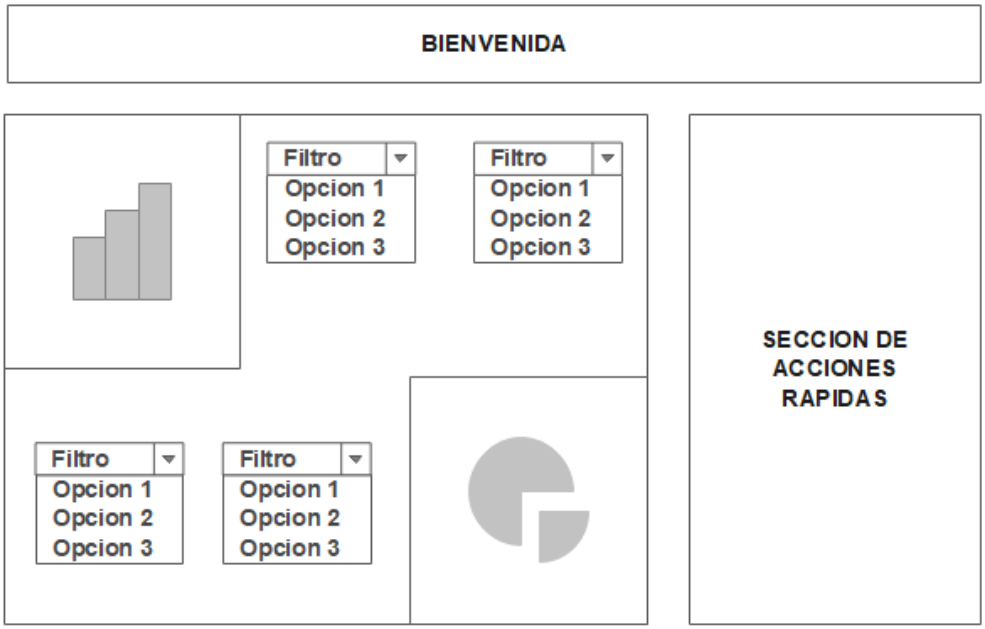


Figura 18. Módulo de inicio

La Figura 19 muestra la interfaz dentro de la ventana “CONTENEDOR” en la cual presenta los datos de un solo ticket. La interfaz se presenta a todos los roles aunque presenta algunas funciones adicionales dependiendo del rol con el cual se acceda al sistema.

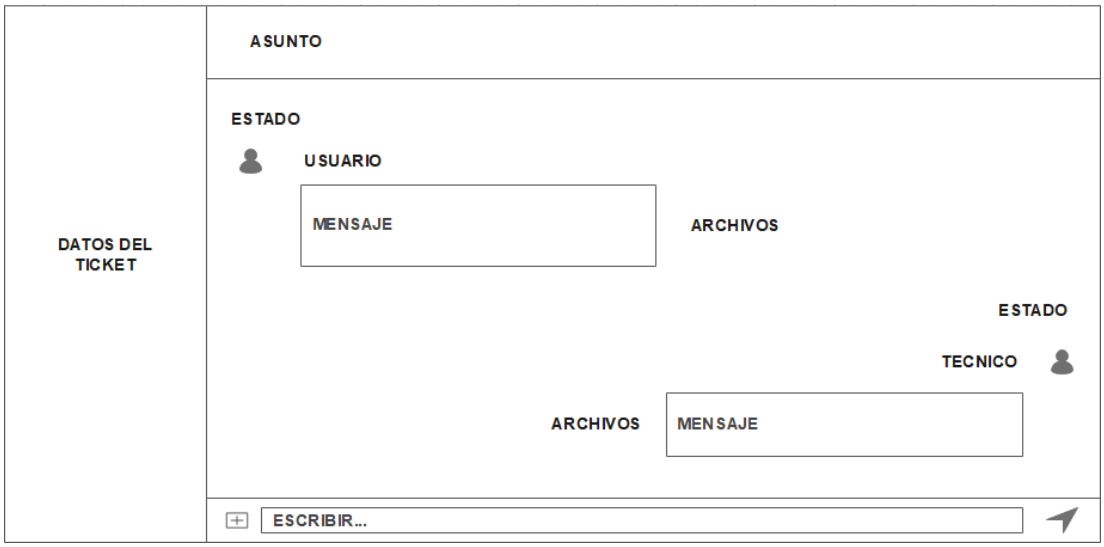


Figura 19. Contenido del ticket

3.2.4 Fase 4: Codificación

a. Iteración 1

Con las herramientas seleccionadas y las historias de usuario definidas, se procedió a la realización de la base de datos en MySQL, además de integrarse con Node JS para los servicios del sistema. Posterior a ello se desarrollaron las historias de usuario planteadas en la fase de exploración y planificación.

La primera Iteración consta del desarrollo para registro de usuarios y administración de los mismos. A continuación se muestra una tabla resumen de lo que se abarca en esta iteración.

Tabla 64. Resumen iteración 1

Iteración	N°	Historia de usuario
1		Base de datos
	1	Acceso al sistema
	2	Recuperar contraseña
	3	Administración de técnicos
	4	Administración de usuarios
	5	Administración de roles

Historia de usuario 1 – Acceso al sistema

Tabla 65. Tarea 1 – Acceso al sistema

TAREA	
Historia de usuario: #1	
Usuario: Todos	
Nombre: Acceso al sistema	
Prioridad en el negocio: Alta	**Riesgo de desarrollo:** Alta
Puntos estimados: 1	**Iteración asignada: 1**
Fecha Inicio: 01/11/2024	**Fecha Fin: 01/11/2024**
Programador a cargo: Patricio Moscoso	
Descripción: El usuario podrá acceder al sistema con sus credenciales.	

El código obtiene el nombre de usuario y la contraseña previamente creadas de un usuario, las credenciales son verificadas para luego acceder a la pantalla principal del sistema, la cual posee funciones activas dependiendo del rol de usuario que posea.

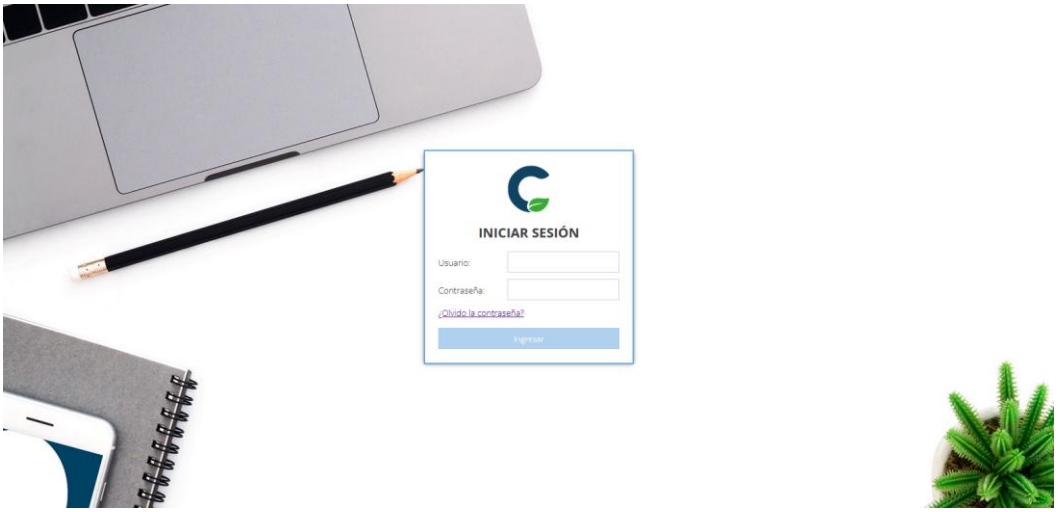


Figura 20. Inicio de sesión

```
        {
            xtype: 'container',
            layout: 'fit',
            id: 'formLogin',
            hidden: false,
            items: [
                {
                    xtype: 'image',
                    src: 'resources/logo.png',
                    alt: 'logo_emp',
                    width: 75,
```

```
                    style: {
                        'display': 'block',
                        'margin': 'auto'
                    }
                },
                {
                    xtype: 'form',
                    id: 'loginForm',
                    items: [
                        {
                            xtype: 'label',
                            text: 'INICIAR SESIÓN',
                            style: {
                                textAlign: 'center',
                                fontWeight: 'bold',
                                'font-size': '20px !important',
                                display: 'block',
                                width: '100%',
                            },
                            margin: '20 0 20 0'
                        }, {
                            xtype: 'textfield',
                            name: 'username',
                            fieldLabel: 'Usuario',
                            allowBlank: false
                        }, {
                            xtype: 'textfield',
                            name: 'password',
                            inputType: 'password',
                            fieldLabel: 'Contraseña',
                            allowBlank: false,
                        }, {
                            xtype: 'label',
                            html: '<span><a href="#">¿Olvido la
contraseña?</a></span>',
                            margin: '0 0 0 0',
                            listeners: {
                                click: {
                                    element: 'el',
                                    fn:
'onClickLabelRecuperarContraseña'
                                }
                            }
                        },
                        {
                            xtype: 'button',
                            text: 'Ingresar',
                            id: 'button_ingresar',
```

```
                            formBind: true,
                            listeners: {
                                click: 'onLoginClick'
                            },
                            style: {
                                'display': 'block',
                                'margin-top': '15px'
                            }
                        }
                    ]
                }
            ]
        },
```

```
server.get('/iniciarSesion', async (req, res) => {
    try {
        var usuario = req.query.usuario.toString().replaceAll('\'',
'')
        var clave =
calcularSHA1(req.query.clave.toString().replaceAll('\'', ''))

        var [results, fields] = await bd.execute(`select
a.id_usuario,
                a.identificacion_usuario,
                a.nombre_usuario,
                a.direccion_usuario,
                a.correo_usuario,
                a.telefono_usuario,
                a.plan_contratado_usuario,
                d.id_sistema,
                d.nombre_sistema,
                c.nombre_plan,
                a.id_fkrol_usuario,
                b.nombre_rol,
                a.fecha_registro_usuario,
                a.fecha_renovacion_usuario
            from usuario as a, rol as b, plan as c, sistema as d
            where a.plan_contratado_usuario = c.id_plan
            and a.id_fkrol_usuario = b.id_rol
            and c.id_fksistema_plan = d.id_sistema
            and a.username_usuario = ?
            and a.clave_usuario = ?
            and a.estado_usuario = 1;`,
```

```
            [usuario, clave]);

        if (results.length > 0) {

            res.cookie("id_usuario", results[0].id_usuario);
            res.cookie("identificacion_usuario",
results[0].identificacion_usuario);
            res.cookie("nombre_usuario", results[0].nombre_usuario);
            res.cookie("rol_usuario", results[0].id_fkrol_usuario);
            res.cookie("nombre_rol_usuario", results[0].nombre_rol);
            res.cookie("plan_usuario",
results[0].plan_contratado_usuario);
            res.cookie("nombre_plan_usuario",
results[0].nombre_plan);
            res.cookie("id_sistema_usuario", results[0].id_sistema);
            res.cookie("nombre_sistema_usuario",
results[0].nombre_sistema);

            return res.json({ data: results, success: true, tipo: 1
});
        } else {
            return res.json({ data: [], success: false, tipo: 2 });
        }
    } catch (error) {
        console.log(error)
        return res.json({ data: "Ha ocurrido un error, intentelo mas
tarde o contacte con soporte", success: false, tipo: 2 });
    }
})
```

Historia de usuario 2 – Recuperar contraseña

Tabla 66. Tarea 2 – Recuperar contraseña

TAREA	
Historia de usuario: #2	
Usuario: Todos	
Nombre: Recuperar contraseña	
Prioridad en el negocio: Alta	**Riesgo de desarrollo:** Alta
Puntos estimados: 1	**Iteración asignada: 1**
Fecha Inicio: 02/11/2024	**Fecha Fin: 02/11/2024**
Programador a cargo: Patricio Moscoso	
Descripción: Todos pueden recuperar la contraseña.	
Observación: Deben tener sus anteriores credenciales para poder solicitar su cambio de contraseña	

Se verifica la existencia de la cédula en el sistema junto con el correo electrónico ingresado, si existen, el sistema generara una contraseña aleatoria y la enviara al correo del usuario para que pueda ingresar al sistema.

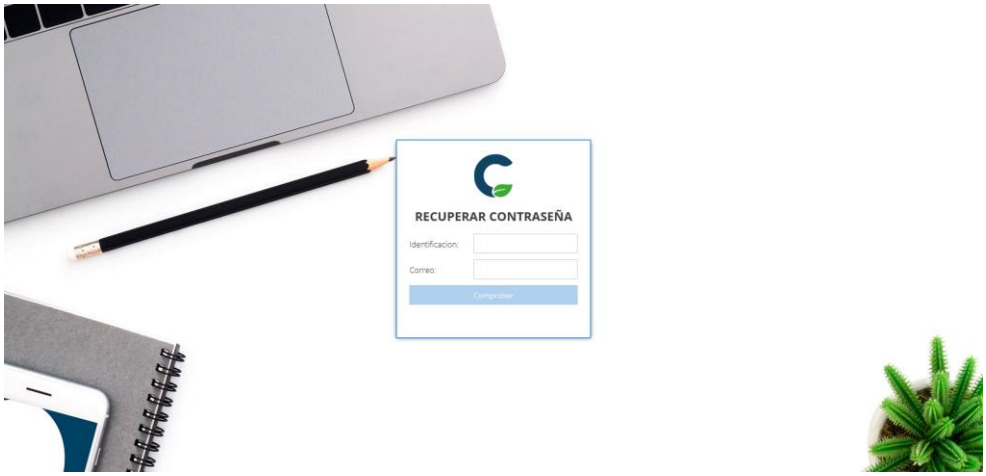


Figura 21. Recuperar contraseña

```
        {
            xtype: 'container',
            layout: 'fit',
            id: 'formRecuperarContra',
            hidden: true,
            items: [
                {
                    xtype: 'image',
                    src: 'resources/logo.png',
                    width: 75,
                    alt: 'logo_emp',
                    style: {
                        'display': 'block',
                        'margin': 'auto'
                    }
                },
                {
                    xtype: 'form',
                    id: 'recuperarClaveForm',
                    items: [
                        {
                            xtype: 'label',
                            text: 'RECUPERAR CONTRASEÑA',
                            style: {
                                textAlign: 'center',
                                fontWeight: 'bold',
```

```
                                'font-size': '20px !important',
                                display: 'block',
                                width: '100%',
                            },
                            margin: '20 0 20 0'
                        }, {
                            xtype: 'textfield',
                            name: 'identificacion',
                            fieldLabel: 'Identificacion',
                            allowBlank: false
                        }, {
                            xtype: 'textfield',
                            name: 'correo',
                            inputType: 'mail',
                            fieldLabel: 'Correo',
                            allowBlank: false
                        }, {
                            xtype: 'button',
                            text: 'Comprobar',
                            id: 'button_comprobar',
                            formBind: true,
                            listeners: {
                                click:
'onClickAccionRecuperarContraseña'
                            },
                            style: {
                                'display': 'block',
                                'margin': 'auto'
                            }
                        },
                    ]
                }
            ]
        },
```

```
const cambiarContraseña = async (req, res, correo, identificacion) =>
{
    try {
        sqlVerificar = `select id_usuario from usuario where
correo_usuario = ? and identificacion_usuario = ?;`

        var [results, fields] = await bd.execute(sqlVerificar, [
            correo, identificacion
        ]);

        if (results.length == 1) {
```

```
            //generar y guardar la contraseña
            var password = generator.generate({
                length: 15,
                numbers: true,
                strict: true
            });

            const contraEncriptada = calcularSHA1(password);
            sqlActualizar = `select cambiar_contraseña_usuario(?,?)
as cambio;`

            var id_encontrado = results[0].id_usuario;
            var [results, fields] = await bd.execute(sqlActualizar, [
                id_encontrado, contraEncriptada
            ]);

            // enviar correo con nueva contraseña
            var info = await emailHelper(
                correo,
                'Recuperar Contraseña',
                'Su clave temporal es ' + password + '. Por favor, no
responda a este correo y no comparta su contraseña.'
            );

            console.log(info.response.includes('OK'))
            return res.json({ data: "Se ha generado un acceso
temporal, revise su correo" , success: true, tipo: 1 });
        } else {
            return res.json({ data: "Usuario no existente, verifique
sus credenciales", success: false, tipo: 2 });
        }
    } catch (error) {
        return res.json({ data: error, success: false, tipo: 2 });

    }

}
```

Historia de usuario 3 – Administración de técnicos

Tabla 67. Tarea 3 – Administración de técnicos

TAREA	
Historia de usuario: #3	
Usuario: Administrador	
Nombre: Administración de técnicos	
Prioridad en el negocio: Alta	**Riesgo de desarrollo:** Alta

Puntos estimados: 1	**Iteración asignada: 1**
Fecha Inicio: 03/11/2024	**Fecha Fin: 03/11/2024**
Programador a cargo: Patricio Moscoso	
Descripción: El usuario administrador podrá crear, modificar y eliminar a otros técnicos.	

El usuario administrador posee la capacidad de crear y modificar técnicos, validando que sus nombres de usuario y cédulas no se repitan. También puede eliminar y deshabilitar a técnicos, siempre que estos no posean tickets aperturados ni tampoco tickets asignados.

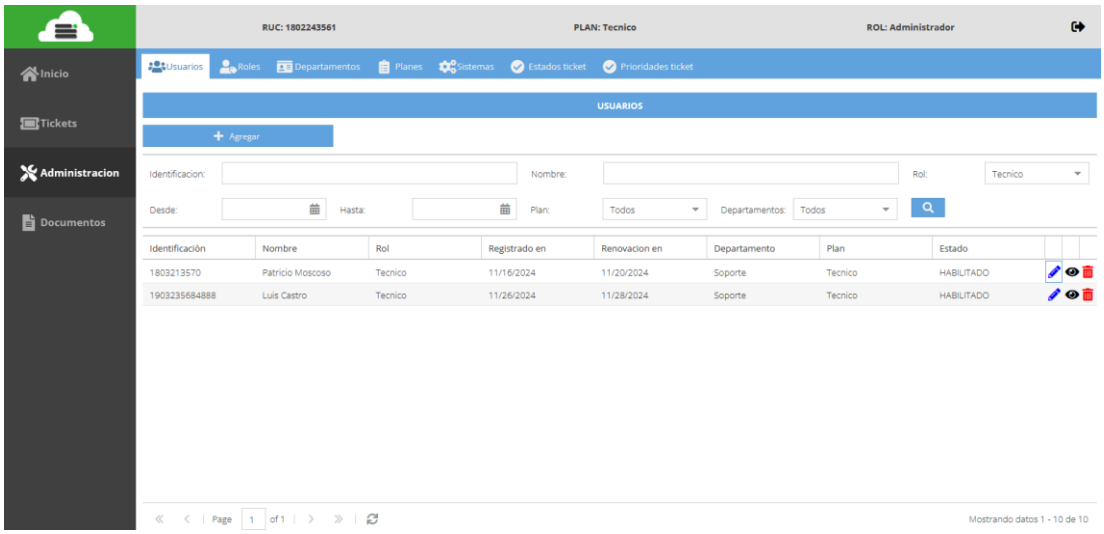


Figura 22. Administración de técnicos

```
        {
            xtype: 'label',
            text: 'USUARIOS',
            columnWidth: 1,
            margin: '10 0 10 0',
            padding: 10,
            style: 'background-color: #5fa2dd; color: white; font-
weight: bold; font-size: 30; text-align: center;'
        },
        {
            xtype: 'button',
            text: 'Agregar',
            id: 'button_agregar_usuario',
            iconCls: 'fas fa-plus',
            columnWidth: 0.2,
            listeners: {
                click: 'onClickAgregarUsuario'
            }
        },
```

```
        {
            xtype: 'panel',
            id: 'campos_buscar_usuario',
            layout: 'column',
            columnWidth: 1,
            header: false,
            margin: '10 0 10 0',
            style: 'border-top: 2px #d0d0d0 solid; border-bottom: 2px
#d0d0d0 solid;',
            items: [
                {
                    xtype: 'textfield',
                    columnWidth: 0.4,
                    padding: 10,
                    id: 'identificacion_usuario',
                    name: 'identificacion',
                    fieldLabel: 'Identificacion',
                },
                {
                    xtype: 'textfield',
                    columnWidth: 0.4,
                    padding: 10,
                    id: 'nombre_usuario',
                    fieldLabel: 'Nombre',
                    name: 'nombre',
                },
                {
                    xtype: 'combo',
                    editable: false,
                    columnWidth: 0.2,
                    padding: 10,
                    id: 'rol_usuario',
                    store:
Ext.create('helpdesk.store.Rol.RolesActivos'),
                    displayField: 'nombre_rol',
                    valueField: 'id_rol',
                    value: '0',
                    fieldLabel: 'Rol',
                    name: 'rol',
                },
                {
                    xtype: 'datefield',
                    columnWidth: 0.2,
                    padding: 10,
                    fieldLabel: 'Desde',
                    format: 'Y-m-d',
                    name: 'fecha_inicio',
                    id: 'fecha_inicio',
```

```
                    value: Date.now(),
                    listeners: {
                        change: function (field, newValue) {
                            var datefieldB =
field.up('container').down('#fecha_fin');
                            datefieldB.setMinValue(newValue);
                        }
                    }
                },
                {
                    xtype: 'datefield',
                    columnWidth: 0.2,
                    padding: 10,
                    format: 'Y-m-d',
                    fieldLabel: 'Hasta',
                    name: 'fecha_fin',
                    id: 'fecha_fin',
                },
                {
                    xtype: 'combo',
                    editable: false,
                    columnWidth: 0.2,
                    padding: 10,
                    id: 'plan',
                    store:
Ext.create('helpdesk.store.plan.PlanesActivos'),
                    displayField: 'nombre_plan',
                    valueField: 'id_plan',
                    value: '-1',
                    fieldLabel: 'Plan',
                    name: 'plan_usuario',
                },
                {
                    xtype: 'combo',
                    editable: false,
                    columnWidth: 0.2,
                    padding: 10,
                    id: 'departamentos_usuario',
                    store:
Ext.create('helpdesk.store.departamento.DepartamentosActivos'),
                    displayField: 'nombre_departamento',
                    valueField: 'id_departamento',
                    value: '0',
                    fieldLabel: 'Departamentos',
                    name: 'departamentos_usuario',
                },
                {
                    xtype: 'button',
```

```
                    columnWidth: 0.05,
                    iconCls: 'fas fa-search',
                    padding: 5,
                    margin: 10,
                    listeners: {
                        click: 'onClickBuscarPorCampos'
                    }
                },
            ]
        },
        {
            xtype: 'grid',
            id: 'grid_usuarios_admin',
            columnWidth: 1,
            header: false,
            scrollable: true,
            style: 'border-top: 2px #d0d0d0 solid; border-bottom: 2px
#d0d0d0 solid;',
            store: Ext.create('helpdesk.store.usuario.Usuarios'),
            columns: [
                { text: 'Id', dataIndex: 'id_usuario', hidden: true,
hideable: false },
                { text: 'Identificaciòn', dataIndex:
'identificacion_usuario', flex: 1 },
                { text: 'Nombre', dataIndex: 'nombre_usuario', flex:
1 },
                { text: 'Id Rol', dataIndex: 'id_fkrol_usuario',
hidden: true, hideable: false },
                { text: 'Rol', dataIndex: 'nombre_rol', flex: 1 },
                { text: 'Registrado en', dataIndex:
'fecha_registro_usuario', formatter: 'date("m/d/Y")', flex: 1 },
                { text: 'Renovacion en', dataIndex:
'fecha_renovacion_usuario', formatter: 'date("m/d/Y")', flex: 1 },
                { text: 'Id Departamento', dataIndex:
'id_fkdepartamento_usuario', hidden: true, hideable: false },
                { text: 'Departamento', dataIndex:
'nombre_departamento', flex: 1 },
                { text: 'Id Plan', dataIndex:
'plan_contratado_usuario', hidden: true, hideable: false },
                { text: 'Plan', dataIndex: 'nombre_plan', flex: 1 },
                { text: 'Valor Estado', dataIndex: 'estado_usuario',
hidden: true, hideable: false },
                { text: 'Estado', dataIndex: 'nombre_estado', flex: 1
},
                {
                    xtype: 'actioncolumn',
                    hideable: false,
                    width: 'auto',
```

```
                items: [
                    {
                        iconCls: 'fas fa-pencil-alt azul',
                        tooltip: 'Editar',
                        display: 'fit',
                        handler: 'onClickEditarUsuario'
                    },

                ]
            },
            {
                xtype: 'actioncolumn',
                hideable: false,
                width: 'auto',
                items: [
                    {
                        getClass: function (v, meta, rec) {
                            var value = rec.get('estado_usuario')
                            if (value == 1) {
                                return 'fas fa-eye negro';
                            } else {
                                return 'fas fa-eye-slash negro';
                            }
                        },
                        handler: 'onClickDesactivarUsuario'
                    }
                ],
            },
            {
                xtype: 'actioncolumn',
                hideable: false,
                width: 'auto',
                items: [
                    {
                        iconCls: 'fas fa-trash-alt rojo',
                        tooltip: 'Eliminar',
                        handler: 'onClickEliminarUsuario'
                    }
                ]
            },
        ],
        listeners: {
            select: 'onItemSelected',
            afterrender: 'calcularTamañoGrid'
        },
        bbar: Ext.create('Ext.PagingToolbar', {
            store: 'Usuarios',
            displayInfo: true,
```

```
                displayMsg: 'Mostrando datos {0} - {1} de {2}',
                emptyMsg: "No hay datos para mostrar",
                inputItemWidth: 35,
            })
        }
```

```
const insertarUsuario = async (req, res, usuarioIng) => {
    try {
        usuarioIng.clave_usuario =
calcularSHA1(usuarioIng.clave_usuario);

        sqlInsertar = `select crear_usuario(?,?,?,?,?,?,?,?,?,?,?,?) as
creacion`;

        var [results, fields] = await bd.execute(sqlInsertar, [
            usuarioIng.identificacion_usuario,
usuarioIng.username_usuario, usuarioIng.clave_usuario,
usuarioIng.nombre_usuario, usuarioIng.direccion_usuario,
usuarioIng.correo_usuario,
            usuarioIng.telefono_usuario,
usuarioIng.plan_contratado_usuario, usuarioIng.id_fkrol_usuario,
usuarioIng.fecha_renovacion_usuario,
usuarioIng.id_fkdepartamento_usuario
        ]);

        if (results[0].creacion == 1) {
            return res.json({ data: "Usuario guardado", success:
true, tipo: 1 });
        } else {
            return res.json({ data: "Error al guardar usuario,
contactese con soporte", success: false, tipo: 2 });
        }
    } catch (error) {
        console.log(error)
        return res.json({ data: error, success: false, tipo: 2 });
    }

}
const editarUsuario = async (req, res, usuarioIng) => {

    try {
        sqlInsertar = `select editar_usuario(?,?,?,?,?,?,?,?,?,?,?,?,?,?)
as edicion`;

        var [results, fields] = await bd.execute(sqlInsertar, [
            usuarioIng.id_usuario, usuarioIng.identificacion_usuario,
usuarioIng.username_usuario, usuarioIng.clave_usuario,
```

```
usuarioIng.nombre_usuario, usuarioIng.direccion_usuario,
usuarioIng.correo_usuario,
            usuarioIng.telefono_usuario,
usuarioIng.plan_contratado_usuario, usuarioIng.id_fkrol_usuario,
usuarioIng.fecha_renovacion_usuario,
usuarioIng.id_fkdepartamento_usuario
        ]);

        if (results[0].edicion == 1) {
            return res.json({ data: "Usuario guardado", success:
true, tipo: 1 });
        } else {
            return res.json({ data: "Error al guardar usuario,
contactese con soporte", success: false, tipo: 2 });
        }
    } catch (error) {
        return res.json({ data: error, success: false, tipo: 2 });

    }

}

const desactivarUsuario = async (req, res, id_usuario, estado) => {
    try {
        sqlDesactivar = `select desactivar_usuario(?,?) as
desactivar`;
        var [results, fields] = await bd.execute(sqlDesactivar,
[id_usuario, estado]);

        if (results[0].desactivar == 1) {
            if (estado == 1) {
                return res.json({ data: "Usuario habilitado",
success: true, tipo: 1 });
            } else {
                return res.json({ data: "Usuario deshabilitado",
success: true, tipo: 1 });
            }
        } else {
            return res.json({ data: "Ocurrio un error al actualizar
el usuario, contactese con soporte", success: false, tipo: 2 });
        }
    } catch (error) {
        console.log(error)
        return res.json({ data: error, success: false, tipo: 2 });
    }
}

const eliminarUsuario = async (req, res, id_usuario) => {
```

```
    try {
        sqlEliminar = `select eliminar_usuario(?) as eliminar`;
        var [results, fields] = await bd.execute(sqlEliminar,
[id_usuario]);

        if (results[0].eliminar == 1) {
            return res.json({ data: "Usuario eliminado", success:
true, tipo: 1 });
        } else {
            return res.json({ data: "Error al eliminar usuario,
contactese con soporte", success: false, tipo: 2 });
        }
    } catch (error) {
        return res.json({ data: error, success: false, tipo: 2 });

    }

}
```

Historia de usuario 4 – Administración de usuarios

TAREA	
Historia de usuario: #4	
Usuario: Administrador	
Nombre: Administración de usuarios	
Prioridad en el negocio: Alta	**Riesgo de desarrollo:** Alta
Puntos estimados: 1	**Iteración asignada: 1**
Fecha Inicio: 04/11/2024	**Fecha Fin: 04/11/2024**
Programador a cargo: Patricio Moscoso	
Descripción: El usuario administrador podrá crear, modificar y eliminar a otros usuarios.	

Tabla 68. Tarea 4 – Administración de usuarios

El usuario administrador posee las mismas acciones que posee en la administración de técnicos, con la diferencia de que en los controladores del Frontend para crear y modificar un usuario, se selecciona un rol y un departamento predeterminado para usuarios comunes, en el apartado Backend se utilizó los mismos servicios para administración de tecnicos.

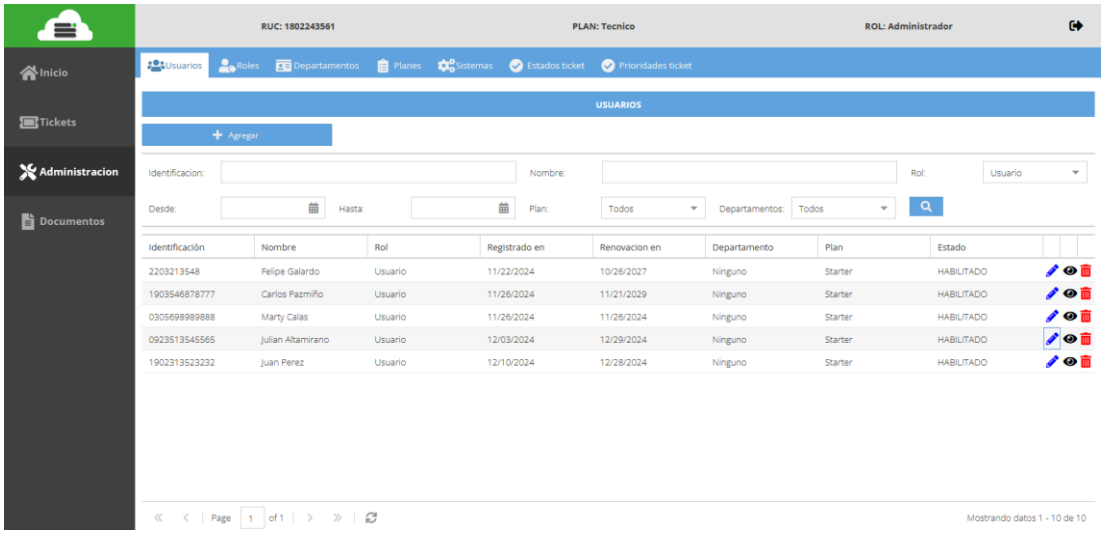


Figura 23. Administración de usuarios

```
onClickAgregarUsuario: function () {
        Ext.create('Ext.window.Window', {
            extend: 'Ext.window.Window',
            alias: 'widget.crearusuarios',
            modal: true,
            title: 'Crear Usuario',
            layout: 'fit',
            closable: false,
            resizable: false,
            bodyStyle: 'background-color: #f5f5f5; padding: 20px;',
            items: [{
                xtype: 'form',
                header: false,
                bodyPadding: 20,
                id: 'form_crear_editar_usuario',
                items: [
                    {
                        xtype: 'textfield',
                        name: 'identificacion_usuario',
                        fieldLabel: 'Identificacion',
                        maskRe: /[0-9]/,
                        maxLength: 13,
                        allowBlank: false
                    },
                    {
                        xtype: 'textfield',
                        name: 'nombre_usuario',
                        fieldLabel: 'Nombre',
                        allowBlank: false
                    },
```

```
                {
                    xtype: 'textfield',
                    name: 'telefono_usuario',
                    fieldLabel: 'Telefono',
                    maxLength: 10,
                    maskRe: /[0-9]/,
                    allowBlank: false
                },
                {
                    xtype: 'textfield',
                    name: 'direccion_usuario',
                    fieldLabel: 'Direccion',
                    allowBlank: false
                },
                {
                    xtype: 'textfield',
                    name: 'correo_usuario',
                    fieldLabel: 'Correo',
                    vtype: 'email',
                    allowBlank: false
                },
                {
                    xtype: 'textfield',
                    name: 'username_usuario',
                    fieldLabel: 'Usuario',
                    allowBlank: false
                },
                {
                    xtype: 'textfield',
                    name: 'clave_usuario',
                    fieldLabel: 'Contraseña',
                    inputType: 'password',
                    allowBlank: false
                },
                {
                    xtype: 'datefield',
                    format: 'Y-m-d',
                    fieldLabel: 'Fecha de Renovacion',
                    name: 'fecha_renovacion',
                    id: 'fecha_renovacion',
                    submitFormat: 'Y-m-d',
                },
                {
                    xtype: 'combo',
                    name: 'id_rol',
                    fieldLabel: 'Rol',
                    itemId: 'comboRol',
```

```
                        store:
Ext.create('helpdesk.store.Rol.RolesActivos'),
                        displayField: 'nombre_rol',
                        valueField: 'id_rol',
                        editable: false,
                        listeners: {
                            beforequery: function (queryPlan) {
                                var combo = queryPlan.combo;
                                var valorOculto = 0; // Valor que
deseas ocultar
                                combo.store.filterBy(function
(record) {
                                    return record.get('id_rol') !==
valorOculto;
                                });
                            },
                            change: function (field, newValue) {
                                var datefieldB =
field.up('form').down('#plan_usuario');
                                var datefieldC =
field.up('form').down('#departamento_usuario');
                                if (field.value == 3) {
                                    datefieldB.setValue('1');
                                    datefieldB.show();

                                    datefieldC.setValue('5');
                                    datefieldC.hide(); //hide
                                }

                                if (field.value == 2) {
                                    datefieldB.setValue('0');
                                    datefieldB.hide(); //hide

                                    datefieldC.getStore().filterBy(fu
nction (record) {
                                        return
record.get('id_departamento') !== 5;
                                    });

                                    datefieldC.setValue('1');
                                    datefieldC.show();
                                }

                                if (field.value == 1) {
                                    datefieldB.setValue('0');
                                    datefieldB.hide(); //hide

                                    datefieldC.setValue('2');
```

```
                                    datefieldC.hide(); //hide
                                }
                            },
                            afterrender: function (field) {
                                field.setValue(1);
                            }
                        }
                    },
                    {
                        xtype: 'combo',
                        editable: false,
                        id: 'plan_usuario',
                        store:
Ext.create('helpdesk.store.plan.PlanesActivos'),
                        displayField: 'nombre_plan',
                        valueField: 'id_plan',
                        value: 1,
                        fieldLabel: 'Plan',
                        name: 'plan',
                        listeners: {
                            beforequery: function (queryPlan) {
                                var combo = queryPlan.combo;
                                combo.store.filterBy(function
(record) {
                                    return record.get('id_plan') !==
-1 && record.get('id_plan') !== 0;
                                });
                            },
                        }
                    },
                    {
                        xtype: 'combo',
                        editable: false,
                        id: 'departamento_usuario',
                        store:
Ext.create('helpdesk.store.departamento.DepartamentosActivos'),
                        displayField: 'nombre_departamento',
                        valueField: 'id_departamento',
                        value: 1,
                        fieldLabel: 'Departamento',
                        name: 'departamento',
                        listeners: {
                            beforequery: function (queryPlan) {
                                var combo = queryPlan.combo;
                                var valorOculto = 0; // Valor que
deseas ocultar
                                combo.store.filterBy(function
(record) {
```

```
                                    return
record.get('id_departamento') !== valorOculto;
                                });
                            },
                        }
                    },
                ]
            }],
            buttons: [
                {
                    text: 'Guardar',
                    formBind: true,
                    style: 'background-color: green; color: white;',
                    handler: function (btn) {
                        var form =
Ext.getCmp('form_crear_editar_usuario').getForm();
                        let values = form.getValues();
                        if (form.isValid()) {

                            Ext.Ajax.request({
                                url:
'http://localhost:8080/usuarios/insertarusuarios',
                                withCredentials: true,
                                method: 'POST',
                                params: {
                                    id: '',
                                    identificacion:
values.identificacion_usuario,
                                    username:
values.username_usuario,
                                    clave: values.clave_usuario,
                                    nombre: values.nombre_usuario,
                                    direccion:
values.direccion_usuario,
                                    correo: values.correo_usuario,
                                    telefono:
values.telefono_usuario,
                                    plan: values.plan,
                                    id_rol: values.id_rol,
                                    fecha_renovacion:
values.fecha_renovacion,
                                    id_departamento:
values.departamento,
                                },
                                success: function (response) {
                                    var response_aux =
Ext.util.JSON.decode(response.responseText, true);
                                    if (response_aux.success) {
```

```
                                    btn.up('window').destroy();
                                    var grid =
Ext.getCmp('grid_usuarios_admin');
                                    var store = grid.getStore();
                                    store.reload();

                                    socket_general.emit('llamado_
actualizar_grids', 'grid_usuarios_admin');

                                    Ext.Msg.alert('Éxito',
'Usuario guardado');

                                } else {
                                    Ext.Msg.alert('Error', 'Ha
ocurrido un error, contactese con soporte');
                                }

                                this.close();
                            },
                            failure: function () {
                                Ext.Msg.alert('Error', 'Ha
ocurrido un error, contactese con soporte');
                            }
                        });
                    } else {
                        Ext.Msg.alert('Error', 'Llene todos los
campos');
                    }
                }
            },
            {
                text: 'Cancelar',
                style: 'background-color: red; color: white;',
                handler: function (btn) {
                    btn.up('window').close();
                }
            }
        ]
    }).show();
},

onClickEditarUsuario: function (grid, rowIndex, colIndex) {
    var record = grid.getStore().getAt(rowIndex);
    var id_usuario = record.get('id_usuario');
    var identificacion_usuario =
record.get('identificacion_usuario');
    var nombre_usuario = record.get('nombre_usuario');
    var telefono_usuario = record.get('telefono_usuario');
```

```
        var direccion = record.get('direccion_usuario');
        var correo = record.get('correo_usuario');
        var username = record.get('username_usuario');
        var clave = record.get('clave_usuario');
        var plan_selecc = record.get('plan_contratado_usuario');
        var id_fkdepartamento_selec =
!record.get('id_fkdepartamento_usuario') ? 0 :
record.get('id_fkdepartamento_usuario');
        var fecha_renovacion_obt = new
Date(record.get('fecha_renovacion_usuario'));
        var id_fkrol_usuario = record.get('id_fkrol_usuario');

        Ext.create('Ext.window.Window', {
            extend: 'Ext.window.Window',
            alias: 'widget.crearusuarios',
            modal: true,
            title: 'Editar Usuario',
            layout: 'fit',
            closable: false,
            resizable: false,
            bodyStyle: 'background-color: #f5f5f5; padding: 20px;',
            items: [{
                xtype: 'form',
                header: false,
                bodyPadding: 20,
                id: 'form_crear_editar_usuario',
                items: [
                    {
                        xtype: 'textfield',
                        name: 'identificacion_usuario',
                        fieldLabel: 'Identificacion',
                        maskRe: /[0-9]/,
                        maxLength: 13,
                        value: identificacion_usuario,
                        allowBlank: false
                    },
                    {
                        xtype: 'textfield',
                        name: 'nombre_usuario',
                        fieldLabel: 'Nombre',
                        value: nombre_usuario,
                        allowBlank: false
                    },
                    {
                        xtype: 'textfield',
                        name: 'telefono_usuario',
                        value: telefono_usuario,
                        fieldLabel: 'Telefono',
```

```
                    maxLength: 10,
                    maskRe: /[0-9]/,
                    allowBlank: false
                },
                {
                    xtype: 'textfield',
                    name: 'direccion_usuario',
                    value: direccion,
                    fieldLabel: 'Direccion',
                    allowBlank: false
                },
                {
                    xtype: 'textfield',
                    name: 'correo_usuario',
                    fieldLabel: 'Correo',
                    value: correo,
                    vtype: 'email',
                    allowBlank: false
                },
                {
                    xtype: 'textfield',
                    name: 'username_usuario',
                    value: username,
                    fieldLabel: 'Usuario',
                    allowBlank: false
                },
                {
                    xtype: 'textfield',
                    name: 'clave_usuario',
                    value: clave,
                    fieldLabel: 'Contraseña',
                    inputType: 'password',
                    allowBlank: false
                },
                {
                    xtype: 'datefield',
                    format: 'Y-m-d',
                    fieldLabel: 'Fecha de Renovacion',
                    name: 'fecha_renovacion',
                    value: fecha_renovacion_obt,
                    id: 'fecha_renovacion',
                    submitFormat: 'Y-m-d',
                },
                {
                    xtype: 'combo',
                    editable: false,
                    name: 'id_rol',
                    fieldLabel: 'Rol',
```

```
                        itemId: 'comboRol',
                        store:
Ext.create('helpdesk.store.Rol.RolesActivos'),
                        displayField: 'nombre_rol',
                        valueField: 'id_rol',
                        //value: id_fkrol_usuario,
                        listeners: {
                            beforequery: function (queryPlan) {
                                var combo = queryPlan.combo;
                                var valorOculto = 0; // Valor que
deseas ocultar
                                combo.store.filterBy(function
(record) {
                                    return record.get('id_rol') !==
valorOculto;
                                });
                            },
                            change: function (field, newValue) {
                                var datefieldB =
field.up('form').down('#plan_usuario');
                                var datefieldC =
field.up('form').down('#departamento_usuario');

                                if (field.value == 3) {
                                    datefieldB.setValue('1');
                                    datefieldB.show();

                                    datefieldC.setValue('5');
                                    datefieldC.hide(); //hide
                                }

                                if (field.value == 2) {
                                    datefieldB.setValue('0');
                                    datefieldB.hide(); //hide

                                    datefieldC.getStore().filterBy(fu
nction (record) {
                                        return
record.get('id_departamento') !== 5;
                                    });

                                    datefieldC.setValue('1');
                                    datefieldC.show();
                                }

                                if (field.value == 1) {
                                    datefieldB.setValue('0');
                                    datefieldB.hide(); //hide
```

```
                                    datefieldC.setValue('2');
                                    datefieldC.hide(); //hide
                                }
                            },
                            afterrender: function (field) {
                                field.setValue(id_fkrol_usuario);
                            }
                        }
                    },
                    {
                        xtype: 'combo',
                        editable: false,
                        id: 'plan_usuario',
                        store:
Ext.create('helpdesk.store.plan.PlanesActivos'),
                        displayField: 'nombre_plan',
                        valueField: 'id_plan',
                        value: plan_selecc,
                        fieldLabel: 'Plan',
                        name: 'plan',
                        listeners: {
                            beforequery: function (queryPlan) {
                                var combo = queryPlan.combo;
                                combo.store.filterBy(function
(record) {
                                    return record.get('id_plan') !==
-1 && record.get('id_plan') !== 0;
                                });
                            },
                        }
                    },
                    {
                        xtype: 'combo',
                        editable: false,
                        id: 'departamento_usuario',
                        store:
Ext.create('helpdesk.store.departamento.DepartamentosActivos'),
                        displayField: 'nombre_departamento',
                        valueField: 'id_departamento',
                        value: id_fkdepartamento_selec,
                        fieldLabel: 'Departamento',
                        name: 'departamento',
                        listeners: {
                            beforequery: function (queryPlan) {
                                var combo = queryPlan.combo;
                                var valorOculto = 0; // Valor que
deseas ocultar
```

```
                                combo.store.filterBy(function
(record) {
                                    return
record.get('id_departamento') !== valorOculto;
                                });
                            }
                        }
                    },
                ]
            }],
            buttons: [
                {
                    text: 'Guardar',
                    formBind: true,
                    style: 'background-color: green; color: white;',
                    handler: function (btn) {
                        var form =
Ext.getCmp('form_crear_editar_usuario').getForm();
                        let values = form.getValues();
                        if (form.isValid()) {

                            Ext.Ajax.request({
                                url:
'http://localhost:8080/usuarios/insertarusuarios',
                                withCredentials: true,
                                method: 'POST',
                                params: {
                                    id: id_usuario,
                                    identificacion:
values.identificacion_usuario,
                                    username:
values.username_usuario,
                                    clave: values.clave_usuario,
                                    nombre: values.nombre_usuario,
                                    direccion:
values.direccion_usuario,
                                    correo: values.correo_usuario,
                                    telefono:
values.telefono_usuario,
                                    plan: values.plan,
                                    id_rol: values.id_rol,
                                    fecha_renovacion:
values.fecha_renovacion,
                                    id_departamento:
values.departamento,
                                },
                                success: function (response) {
```

```
                                    var response_aux =
Ext.util.JSON.decode(response.responseText, true);
                                    if (response_aux.success) {
                                        btn.up('window').destroy();
                                        var grid =
Ext.getCmp('grid_usuarios_admin');
                                        var store = grid.getStore();
                                        store.reload();

                                        socket_general.emit('llamado_
actualizar_grids', 'grid_usuarios_admin');

                                        Ext.Msg.alert('Éxito',
'Usuario guardado');

                                    } else {
                                        Ext.Msg.alert('Error', 'Ha
ocurrido un error, contactese con soporte');
                                    }

                                    this.close();
                                },
                                failure: function () {
                                    Ext.Msg.alert('Error', 'Ha
ocurrido un error, contactese con soporte');
                                }
                            });
                        } else {
                            Ext.Msg.alert('Error', 'Llene todos los
campos');
                        }
                    }
                },
                {
                    text: 'Cancelar',
                    style: 'background-color: red; color: white;',
                    handler: function (btn) {
                        btn.up('window').close();
                    }
                }
            ]
        }).show();
    },
```

Historia de usuario 5 – Administración de roles

Tabla 69. Tarea 5 – Administración de roles

TAREA	
Historia de usuario: #5	
Usuario: Administrador	
Nombre: Administración de roles	
Prioridad en el negocio: Alta	**Riesgo de desarrollo:** Alta
Puntos estimados: 1	**Iteración asignada:** 1
Fecha Inicio: 05/11/2024	**Fecha Fin: 06/11/2024**
Programador a cargo: Patricio Moscoso	
Descripción: El usuario administrador podrá crear, modificar y deshabilitar roles de usuarios.	

El usuario administrador puede crear y modificar roles, los nombres ingresados no pueden repetirse. Solo se puede deshabilitar los roles existentes, no se pueden eliminar de ninguna forma debido a que el sistema trabaja con los tres roles por defecto (administrador, técnico, y usuario).

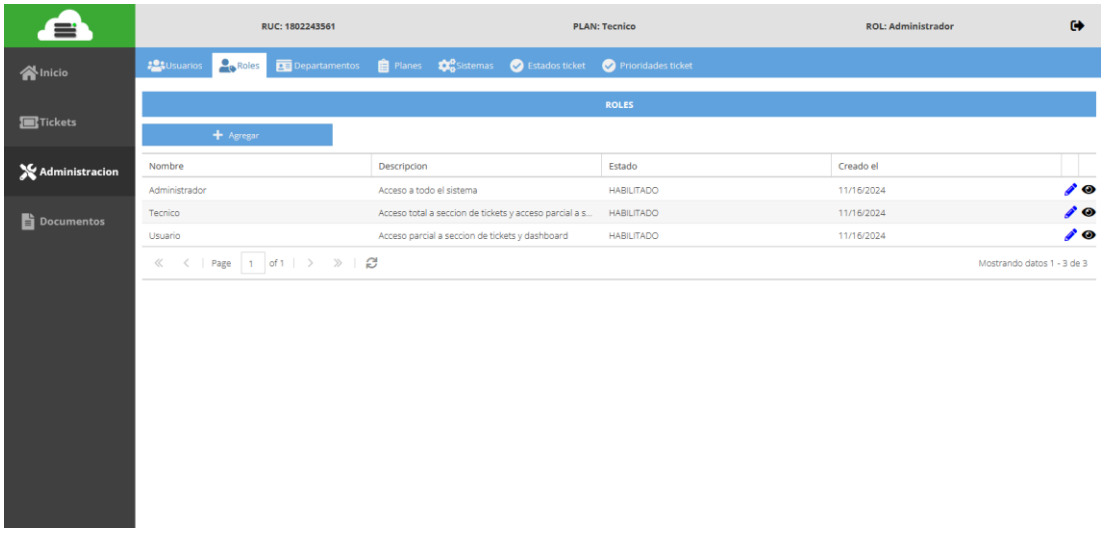


Figura 24. Administración de roles

```
        {
            xtype: 'label',
            text: 'ROLES',
            columnWidth: 1,
            margin: '10 0 10 0',
            padding: 10,
            style: 'background-color: #5fa2dd; color: white; font-
weight: bold; font-size: 30; text-align: center;'
```

```
        },
        {
            xtype: 'button',
            text: 'Agregar',
            iconCls: 'fas fa-plus',
            margin: '0 0 10 0',
            columnWidth: 0.2,
            listeners: {
                click: 'onClickAgregarRol'
            }
        },
        {
            xtype: 'grid',
            id: 'grid_roles_admin',
            columnWidth: 1,
            header: false,
            style: 'border-top: 2px #d0d0d0 solid; border-bottom: 2px
#d0d0d0 solid;',
            store: Ext.create('helpdesk.store.Rol.Rol'),
            columns: [
                { text: 'Id', dataIndex: 'id_rol', hidden: true,
hideable: false },
                { text: 'Nombre', dataIndex: 'nombre_rol', flex: 1 },
                { text: 'Descripcion', dataIndex: 'descripcion_rol',
flex: 1},
                { text: 'Estado', dataIndex: 'nombre_estado_rol',
flex: 1 },
                { text: 'Creado el', dataIndex: 'fecha_creacion_rol',
formatter: 'date("m/d/Y")', flex: 1 },
                { text: 'Id Estado', dataIndex: 'estado_rol', hidden:
true, hideable: false },
                {
                    xtype: 'actioncolumn',
                    hideable: false,
                    width: 'auto',
                    items: [
                        {
                            iconCls: 'fas fa-pencil-alt azul',
                            tooltip: 'Editar',
                            display: 'fit',
                            handler: 'onClickEditarRol'
                        },

                    ]
                },
                {
                    xtype: 'actioncolumn',
                    hideable: false,
```

```
                    width: 'auto',
                    items: [
                        {
                            getClass: function (v, meta, rec) {
                                var value = rec.get('estado_rol')
                                if (value == 1) {
                                    return 'fas fa-eye negro';
                                } else {
                                    return 'fas fa-eye-slash negro';
                                }
                            },
                            handler: 'onClickDesactivarRol'
                        }
                    ],
                },
            ],
            listeners: {
                select: 'onItemSelected'
            },
            bbar: Ext.create('Ext.PagingToolbar', {
                store: 'Rol',
                displayInfo: true,
                displayMsg: 'Mostrando datos {0} - {1} de {2}',
                emptyMsg: "No hay datos para mostrar",
                inputItemWidth: 35,
            })
        }
```

```
const insertarRoles = async (req, res, rolIngresado) => {

    try {
        sqlInsertar = `select crear_rol(?,?) as creacion`;

        var [results, fields] = await bd.execute(sqlInsertar, [
            rolIngresado.nombre_rol, rolIngresado.descripcion_rol
        ]);

        if (results[0].creacion == 1) {
            return res.json({ data: "Rol guardado", success: true,
tipo: 1 });
        } else {
            return res.json({ data: "Error al guardar rol, contactese
con soporte", success: false, tipo: 2 });
        }
    } catch (error) {
```

```
        return res.json({ data: "Error al guardar rol, contactese con
soporte", success: false, tipo: 2 });

    }

}

const editarRoles = async (req, res, rolIngresado) => {
    try {
        sqlInsertar = `select editar_rol(?,?,?) as edicion`;

        var [results, fields] = await bd.execute(sqlInsertar, [
            rolIngresado.id_rol, rolIngresado.nombre_rol,
rolIngresado.descripcion_rol
        ]);

        if (results[0].edicion == 1) {
            return res.json({ data: "Rol guardado", success: true,
tipo: 1 });
        } else {
            return res.json({ data: "Error al guardar rol, contactese
con soporte", success: false, tipo: 2 });
        }
    } catch (error) {
        return res.json({ data: "Error al guardar rol, contactese con
soporte", success: false, tipo: 2 });
    }

}

const desactivarRol = async (req, res, id_rol, estado) => {
    try {
        sqlDesactivar = `select desactivar_activar_rol(?,?) as
desactivar`;
        var [results, fields] = await bd.execute(sqlDesactivar,
[id_rol, estado]);

        if (results[0].desactivar == 1) {
            if (estado == 1) {
                return res.json({ data: "Rol habilitado", success:
true, tipo: 1 });
            } else {
                return res.json({ data: "Rol deshabilitado", success:
true, tipo: 1 });
            }
        } else {
            return res.json({ data: "Ocurrio un error al actualizar
el rol, contactese con soporte", success: false, tipo: 2 });
```

```
        }
    } catch (error) {
        console.log(error)
        return res.json({ data: "Ocurrio un error al actualizar el
rol, contactese con soporte", success: false, tipo: 2 });
    }
}
```

b. Iteración 2

La Iteración 2 consta del desarrollo para administración de servicios y sistemas que ofrece la empresa, la funcionalidad de cerrar sesión y la administración de departamentos para asignación de usuarios. A continuación se muestra una tabla resumen de lo que se abarca en esta iteración.

Tabla 70. Resumen Iteración 2

Iteración	N°	Historia de usuario
2	6	Administración de departamentos
	7	Administración de planes
	8	Administración de sistemas
	9	Cerrar Sesión

Historia de usuario 6 – Administración de departamentos

Tabla 71. Tarea 6 – Administración de departamentos

TAREA	
Historia de usuario: #6	
Usuario: Administrador	
Nombre: Administración de departamentos	
Prioridad en el negocio: Alta	**Riesgo de desarrollo:** Alta
Puntos estimados: 1	**Iteración asignada: 2**
Fecha Inicio: 06/11/2024	**Fecha Fin: 06/11/2024**
Programador a cargo: Patricio Moscoso	
Descripción: El usuario administrador podrá crear, modificar y deshabilitar los departamentos asignados de usuarios.	

El usuario administrador puede crear y modificar los departamentos, se controla que el nombre no se repita con los departamentos existentes. Solo se puede deshabilitar departamentos, no se pueden eliminar de ninguna forma ya que el sistema trabaja con los departamentos definidos por defecto (Ninguno, Soporte, Escalado).

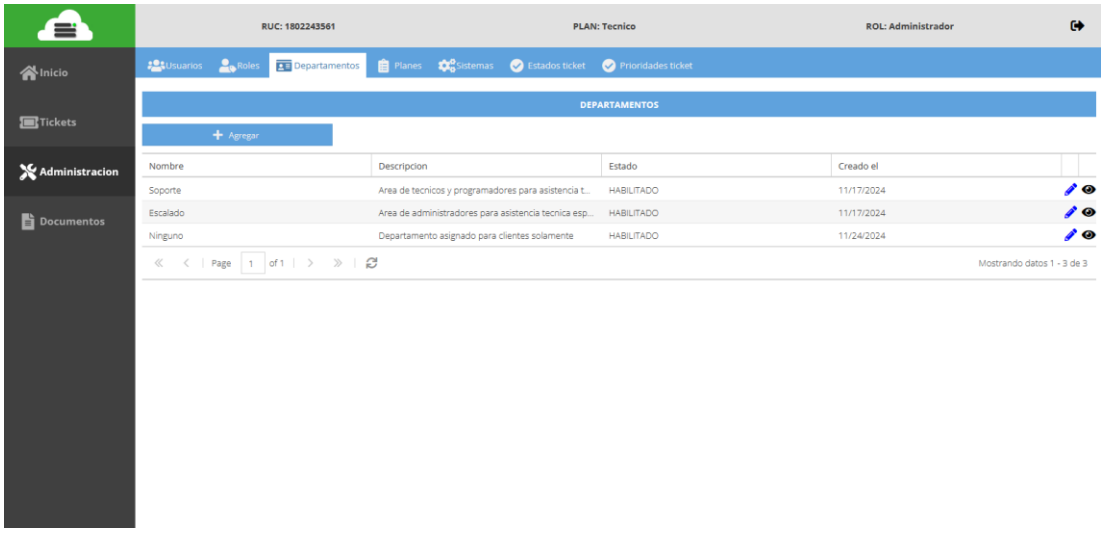


Figura 25. Administración de departamentos

```
        {
            xtype: 'label',
            text: 'DEPARTAMENTOS',
            columnWidth: 1,
            margin: '10 0 10 0',
            padding: 10,
            style: 'background-color: #5fa2dd; color: white; font-
weight: bold; font-size: 30; text-align: center;'
        },
        {
            xtype: 'button',
            text: 'Agregar',
            iconCls: 'fas fa-plus',
            margin: '0 0 10 0',
            columnWidth: 0.2,
            listeners: {
                click: 'onClickAgregarDepartamento'
            }
        },
        {
            xtype: 'grid',
            id: 'grid_departamentos_admin',
            columnWidth: 1,
            header: false,
            style: 'border-top: 2px #d0d0d0 solid; border-bottom: 2px
#d0d0d0 solid;',
            store:
Ext.create('helpdesk.store.departamento.Departamentos'),
            columns: [
```

```
                { text: 'Id', dataIndex: 'id_departamento', hidden:
true, hideable: false },
                { text: 'Nombre', dataIndex: 'nombre_departamento',
flex: 1 },
                { text: 'Descripcion', dataIndex:
'descripcion_departamento', flex: 1},
                { text: 'Estado', dataIndex:
'nombre_estado_departamento', flex: 1 },
                { text: 'Creado el', dataIndex:
'fecha_creacion_departamento', formatter: 'date("m/d/Y")', flex: 1 },
                { text: 'Id Estado', dataIndex:
'estado_departamento', hidden: true, hideable: false },
                { text: 'Id Usuario', dataIndex:
'id_fkusuario_desactivacion_departamento', hidden: true, hideable:
false },
                { text: 'Usuario', dataIndex: 'nombre_usuario',
hidden: true, hideable: false },
                { text: 'Motivo Desactivacion', dataIndex:
'motivo_desactivacion_departamento', flex: 1, hidden: true },
                { text: 'Ult. Fecha Desactivacion', dataIndex:
'fecha_desactivacion_departamento', formatter: 'date("m/d/Y")', flex:
1, hidden: true },
                {
                    xtype: 'actioncolumn',
                    hideable: false,
                    width: 'auto',
                    items: [
                        {
                            iconCls: 'fas fa-pencil-alt azul',
                            tooltip: 'Editar',
                            display: 'fit',
                            handler: 'onClickEditarDepartamento'
                        },
                    ]
                },
                {
                    xtype: 'actioncolumn',
                    hideable: false,
                    width: 'auto',
                    items: [
                        {
                            getClass: function (v, meta, rec) {
                                var value =
rec.get('estado_departamento')
                                if (value == 1) {
                                    return 'fas fa-eye negro';
                                } else {
                                    return 'fas fa-eye-slash negro';
```

```
                                }
                            },
                            handler: 'onClickDesactivarDepartamento'
                        }
                    ],
                },
            ],
            listeners: {
                select: 'onItemSelected'
            },
            bbar: Ext.create('Ext.PagingToolbar', {
                store: 'Departamento',
                displayInfo: true,
                displayMsg: 'Mostrando datos {0} - {1} de {2}',
                emptyMsg: "No hay datos para mostrar",
                inputItemWidth: 35,
            })
        }
```

```
const insertarDepartamentos = async (req, res, departamento) => {
    try {
        sqlInsertar = `select crear_departamento(?,?) as creacion`;

        var [results, fields] = await bd.execute(sqlInsertar, [
            departamento.nombre_departamento,
departamento.descripcion_departamento
        ]);

        if (results[0].creacion == 1) {
            return res.json({ data: "Departamento guardado", success:
true, tipo: 1 });
        } else {
            return res.json({ data: "Error al guardar departamento,
contactese con soporte", success: false, tipo: 2 });
        }
    } catch (error) {
        return res.json({ data: error, success: false, tipo: 2 });

    }
}

const editarDepartamentos = async (req, res, departamento) => {

    try {
        sqlInsertar = `select actualizar_departamento(?,?,?) as
edicion`;
```

```
        var [results, fields] = await bd.execute(sqlInsertar, [
            departamento.id_departamento,
departamento.nombre_departamento,
departamento.descripcion_departamento
        ]);

        if (results[0].edicion == 1) {
            return res.json({ data: "Departamento guardado", success:
true, tipo: 1 });
        } else {
            return res.json({ data: "Error al guardar departamento,
contactese con soporte", success: false, tipo: 2 });
        }
    } catch (error) {
        return res.json({ data: error, success: false, tipo: 2 });

    }
}

const activarDesactivarDepartamentos = async (req, res, departamento)
=> {
    try {
        sqlInsertar = `select
desactivar_activar_departamento(?,?,?,?) as edicion`;

        var [results, fields] = await bd.execute(sqlInsertar, [
            departamento.id_departamento,
departamento.estado_departamento,
            departamento.id_fkusuario_desactivacion_departamento,
departamento.motivo_desactivacion_departamento
        ]);

        if (results[0].edicion == 1) {
            return res.json({ data: "Departamento actualizado",
success: true, tipo: 1 });
        } else {
            return res.json({ data: "Error al actualizar el
departamento, contactese con soporte", success: false, tipo: 2 });
        }
    } catch (error) {
        return res.json({ data: error, success: false, tipo: 2 });
    }
}
```

Historia de usuario 7 – Administración de planes

Tabla 72. Tarea 7 – Administración de planes

TAREA	
Historia de usuario: #7	
Usuario: Administrador	
Nombre: Administración de planes	
Prioridad en el negocio: Alta	**Riesgo de desarrollo:** Alta
Puntos estimados: 1	**Iteración asignada: 2**
Fecha Inicio: 07/11/2024	**Fecha Fin: 07/11/2024**
Programador a cargo: Patricio Moscoso	
Descripción: El usuario administrador podrá crear, modificar y deshabilitar los planes asignados a sistemas.	

El usuario administrador puede crear, eliminar y modificar los planes de cada sistema. En caso de eliminar, se verifica que el plan no este asignado a ningún usuario. Al crear o modificar un plan, se compara el nombre del plan para evitar que se repitan en los registros.

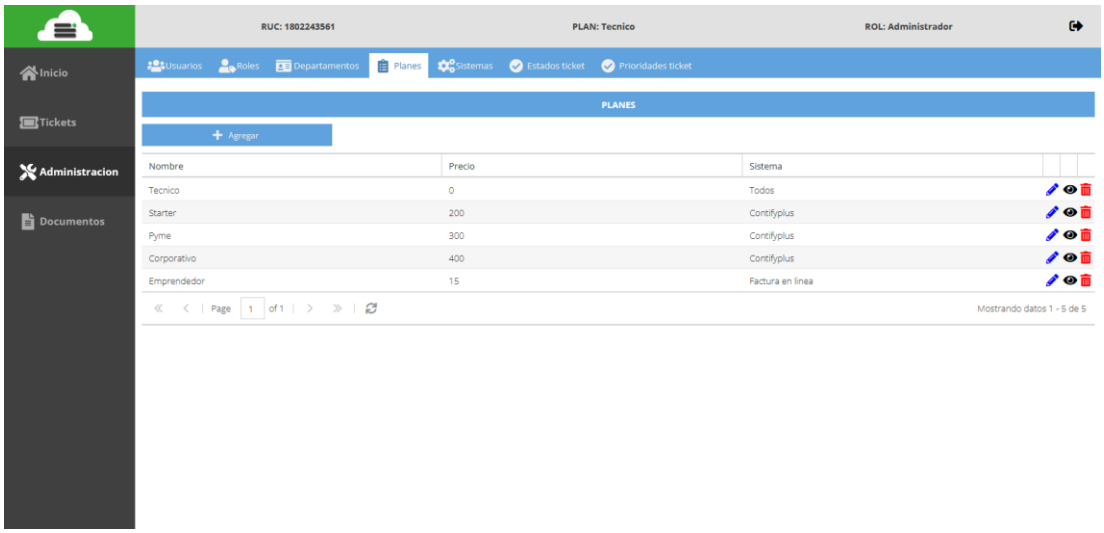


Figura 26. Administración de planes

```
        {
            xtype: 'label',
            text: 'PLANES',
            columnWidth: 1,
            margin: '10 0 10 0',
            padding: 10,
            style: 'background-color: #5fa2dd; color: white; font-
weight: bold; font-size: 30; text-align: center;'
```

```
        },
        {
            xtype: 'button',
            text: 'Agregar',
            iconCls: 'fas fa-plus',
            margin: '0 0 10 0',
            id: 'button_agregar_plan',
            columnWidth: 0.2,
            listeners: {
                click: 'onClickAgregarPlan'
            }
        },
        {
            xtype: 'grid',
            id: 'grid_planes_admin',
            columnWidth: 1,
            header: false,
            style: 'border-top: 2px #d0d0d0 solid; border-bottom: 2px
#d0d0d0 solid;',
            store: Ext.create('helpdesk.store.plan.Plan'),
            columns: [
                { text: 'Id', dataIndex: 'id_plan', hidden: true,
hideable: false },
                { text: 'Nombre', dataIndex: 'nombre_plan', flex: 1
},
                { text: 'Precio', dataIndex: 'precio_plan', flex: 1},
                { text: 'Id Sistema', dataIndex: 'id_fksistema_plan',
hidden: true, hideable: false },
                { text: 'Sistema', dataIndex: 'nombre_sistema', flex:
1 },
                { text: 'Servicios', dataIndex:
'servicios_ofrece_plan', hidden: true, hideable: false },
                {
                    xtype: 'actioncolumn',
                    hideable: false,
                    width: 'auto',
                    items: [
                        {
                            iconCls: 'fas fa-pencil-alt azul',
                            tooltip: 'Editar',
                            display: 'fit',
                            handler: 'onClickEditarPlan'
                        },
                    ]
                },
                {
                    xtype: 'actioncolumn',
                    hideable: false,
```

```
                    width: 'auto',
                    items: [
                        {
                            iconCls: 'fas fa-eye negro',
                            tooltip: 'Ver beneficios',
                            display: 'fit',
                            handler: 'onClickVerificarBeneficios'
                        }
                    ],
                },
                {
                    xtype: 'actioncolumn',
                    hideable: false,
                    width: 'auto',
                    items: [
                        {
                            iconCls: 'fas fa-trash-alt rojo',
                            tooltip: 'Eliminar',
                            handler: 'onClickEliminarPlan'
                        }
                    ]
                }
            ],
            listeners: {
                select: 'onItemSelected'
            },
            bbar: Ext.create('Ext.PagingToolbar', {
                store: 'Plan',
                displayInfo: true,
                displayMsg: 'Mostrando datos {0} - {1} de {2}',
                emptyMsg: "No hay datos para mostrar",
                inputItemWidth: 35,
            })
        }
```

```
const insertarPlanes = async (req, res, planIngresado) => {

    try {
        sqlInsertar = `select agregar_plan(?,?,?,?) as creacion`;

        var [results, fields] = await bd.execute(sqlInsertar, [
            planIngresado.nombre_plan, planIngresado.precio_plan,
            planIngresado.id_fksistema_plan,
planIngresado.servicios_ofrece_plan
        ]);

        if (results[0].creacion == 1) {
```

```
            return res.json({ data: "Plan guardado", success: true,
tipo: 1 });
        } else {
            return res.json({ data: "Error al guardar el plan,
contactese con soporte", success: false, tipo: 2 });
        }
    } catch (error) {
        return res.json({ data: error.message, success: false, tipo:
2 });
    }
}

const editarPlanes = async (req, res, planIngresado) => {
    try {
        sqlInsertar = `select editar_plan(?,?,?,?,?) as edicion`;

        var [results, fields] = await bd.execute(sqlInsertar, [
            planIngresado.id_plan, planIngresado.nombre_plan,
planIngresado.precio_plan,
            planIngresado.id_fksistema_plan,
planIngresado.servicios_ofrece_plan
        ]);

        if (results[0].edicion == 1) {
            return res.json({ data: "Plan guardado", success: true,
tipo: 1 });
        } else {
            return res.json({ data: "Error al guardar el plan,
contactese con soporte", success: false, tipo: 2 });
        }
    } catch (error) {
        return res.json({ data: error.message, success: false, tipo:
2 });
    }
}

const eliminarPlanes = async (req, res, id_plan) => {

    try {
        sqlEliminar = `select eliminar_plan(?) as eliminar`;
        var [results, fields] = await bd.execute(sqlEliminar,
[id_plan]);

        if (results[0].eliminar == 1) {
            return res.json({ data: "Plan eliminado", success: true,
tipo: 1 });
        } else {
```

```
            return res.json({ data: "Error al eliminar el plan,
contactese con soporte", success: false, tipo: 2 });
        }
    } catch (error) {
        return res.json({ data: error.message, success: false, tipo:
2 });
    }
}
```

Historia de usuario 8 – Administración de sistemas

Tabla 73. Tarea 8 – Administración de sistemas

TAREA	
Historia de usuario: #8	
Usuario: Administrador	
Nombre: Administración de sistemas	
Prioridad en el negocio: Alta	**Riesgo de desarrollo:** Alta
Puntos estimados: 1	**Iteración asignada: 2**
Fecha Inicio: 08/11/2024	**08/11/2024**
Programador a cargo: Patricio Moscoso	
Descripción: El usuario administrador podrá crear, modificar y eliminar los sistemas o servicios existentes en la empresa.	

El usuario administrador puede crear y modificar sistemas que posea la empresa, con el fin de asignar planes y módulos asociados a los mismos. También se pueden eliminar siempre que estos no posean planes, módulos y tickets asociados.

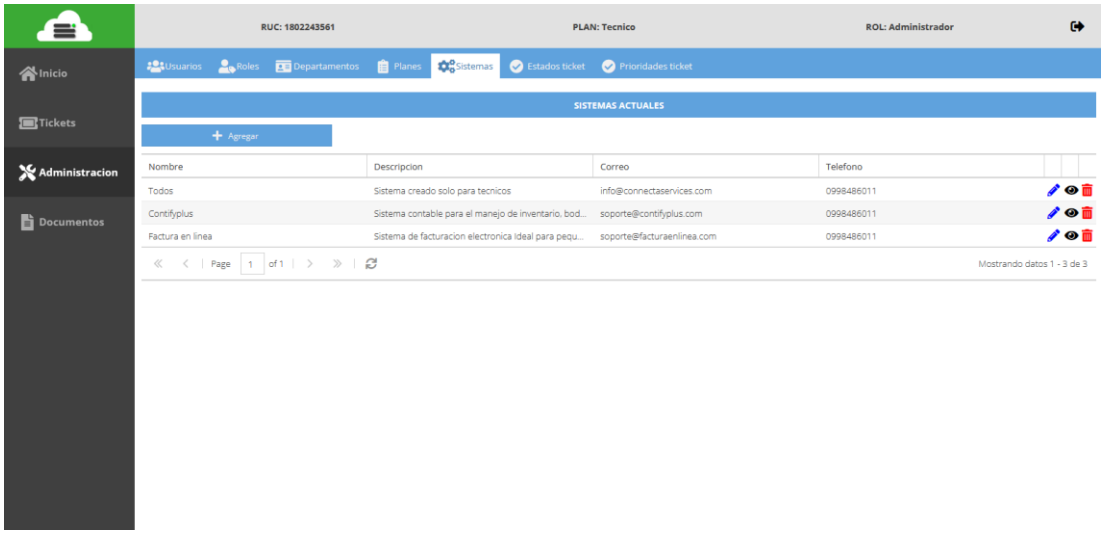


Figura 27. Administración de sistemas

```
        {
            xtype: 'panel',
            id: 'ContenedorSistemas',
            layout: 'column',
            columnWidth: 1,
            items: [
                {
                    xtype: 'label',
                    text: 'SISTEMAS ACTUALES',
                    columnWidth: 1,
                    margin: '10 0 10 0',
                    padding: 10,
                    style: 'background-color: #5fa2dd; color: white;
font-weight: bold; font-size: 30; text-align: center;'
                },
                {
                    xtype: 'button',
                    text: 'Agregar',
                    iconCls: 'fas fa-plus',
                    id: 'button_agregar_sistema',
                    margin: '0 0 10 0',
                    columnWidth: 0.2,
                    listeners: {
                        click: 'onClickAgregarSistema'
                    }
                },
                {
                    xtype: 'grid',
                    id: 'grid_sistemas_admin',
                    columnWidth: 1,
                    header: false,
                    style: 'border-top: 2px #d0d0d0 solid; border-
bottom: 2px #d0d0d0 solid;',
                    store:
Ext.create('helpdesk.store.sistema.Sistema'),
                    columns: [
                        { text: 'Id', dataIndex: 'id_sistema',
hidden: true, hideable: false },
                        { text: 'Nombre', dataIndex:
'nombre_sistema', flex: 1 },
                        { text: 'Descripcion', dataIndex:
'descripcion_sistema', flex: 1},
                        { text: 'Correo', dataIndex:
'correo_soporte_sistema', flex: 1 },
                        { text: 'Telefono', dataIndex:
'telefono_soporte_sistema', flex: 1 },
                        {
                            xtype: 'actioncolumn',
```

```
                            hideable: false,
                            width: 'auto',
                            items: [
                                {
                                    iconCls: 'fas fa-pencil-alt
azul',
                                    tooltip: 'Editar',
                                    display: 'fit',
                                    handler: 'onClickEditarSistema'
                                },

                            ]
                        },
                        {
                            xtype: 'actioncolumn',
                            hideable: false,
                            width: 'auto',
                            items: [
                                {
                                    iconCls: 'fas fa-eye negro',
                                    tooltip: 'Ver modulos',
                                    display: 'fit',
                                    handler:
'onClickVisualizarModulos'
                                },

                            ]
                        },
                        {
                            xtype: 'actioncolumn',
                            hideable: false,
                            width: 'auto',
                            items: [
                                {
                                    iconCls: 'fas fa-trash-alt rojo',
                                    tooltip: 'Eliminar',
                                    handler: 'onClickEliminarSistema'
                                }
                            ]
                        }
                    ],
                    listeners: {
                        select: 'onItemSelected'
                    },
                    bbar: Ext.create('Ext.PagingToolbar', {
                        store: 'Sistema',
                        displayInfo: true,
```

```
                        displayMsg: 'Mostrando datos {0} - {1} de
{2}',
                        emptyMsg: "No hay datos para mostrar",
                        inputItemWidth: 35,
                    })
                }
            ]
        }
```

```
const insertarSistemas = async (req, res, sistemaIngresado) => {

    try {
        sqlInsertar = `select agregar_sistema(?,?,?,?) as creacion`;

        var [results, fields] = await bd.execute(sqlInsertar, [
            sistemaIngresado.nombre_sistema,
sistemaIngresado.descripcion_sistema,
            sistemaIngresado.correo_soporte_sistema,
sistemaIngresado.telefono_soporte_sistema
        ]);

        if (results[0].creacion == 1) {
            return res.json({ data: "Sistema guardado", success:
true, tipo: 1 });
        } else {
            return res.json({ data: "Error al guardar el sistema,
contactese con soporte", success: false, tipo: 2 });
        }
    } catch (error) {
        return res.json({ data: "Error al guardar el sistema,
contactese con soporte", success: false, tipo: 2 });
    }
}

const editarSistemas = async (req, res, sistemaIngresado) => {
    try {
        sqlInsertar = `select editar_info_sistema(?,?,?,?,?) as
edicion`;

        var [results, fields] = await bd.execute(sqlInsertar, [
            sistemaIngresado.id_sistema,
sistemaIngresado.nombre_sistema,
sistemaIngresado.descripcion_sistema,
            sistemaIngresado.correo_soporte_sistema,
sistemaIngresado.telefono_soporte_sistema
        ]);
```

```
        if (results[0].edicion == 1) {
            return res.json({ data: "Rol guardado", success: true,
tipo: 1 });
        } else {
            return res.json({ data: "Error al guardar el sistema,
contactese con soporte", success: false, tipo: 2 });
        }
    } catch (error) {
        return res.json({ data: "Error al guardar el sistema,
contactese con soporte", success: false, tipo: 2 });
    }
}

const eliminarSistemas = async (req, res, id_sistema) => {

    try {
        sqlEliminar = `select eliminar_sistema(?) as eliminar`;
        var [results, fields] = await bd.execute(sqlEliminar,
[id_sistema]);

        if (results[0].eliminar == 1) {
            return res.json({ data: "Sistema eliminado", success:
true, tipo: 1 });
        } else {
            return res.json({ data: "Error al eliminar sistema,
contactese con soporte", success: true, tipo: 1 });
        }
    } catch (error) {
        return res.json({ data: "Error al eliminar sistema,
contactese con soporte", success: false, tipo: 2 });
    }
}
```

Historia de usuario 9 – Cerrar sesión

Tabla 74. Tarea 9 – Cerrar sesión

TAREA	
Historia de usuario: #9	
Usuario: Todos	
Nombre: Cerrar Sesión	
Prioridad en el negocio: Alta	**Riesgo de desarrollo:** Media
Puntos estimados: 1	**Iteración asignada: 2**
Fecha Inicio: 09/11/2024	**Fecha Final: 09/11/2024**
Programador a cargo: Patricio Moscoso	
Descripción: Todos podrán cerrar sesión desde el menú principal.	

El botón diseñado en la parte superior derecha efectúe el evento de cerrar sesión, donde se eliminan las cookies asociadas al servidor y al navegador, haciendo que se reestablezca la ventana de ingreso al sistema.

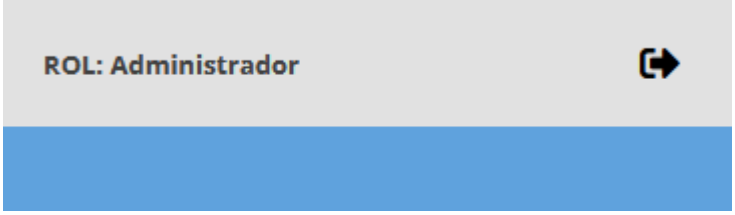


Figura 28. Botón de cierre de sesión

```
{
            xtype: 'button',
            text: '<div style="color: black; font-weight: bold; font-
size: 20px;"> <i class="fas fa-sign-out-alt"></i></div>',
            iconAlign: 'right',
            style: 'background-color: transparent; border: 0px; ',
            flex: 0,
            listeners: {
                click: {
                    fn: function (btn) {
                        Ext.Ajax.request({
                            withCredentials: true,
                            method: 'GET',
                            url: 'http://localhost:8080/salirSesion',

                            success: function () {
                                localStorage.clear();
                                Ext.util.Cookies.clear('id_usuario');
                                Ext.util.Cookies.clear('identificacio
n_usuario');
                                Ext.util.Cookies.clear('nombre_usuari
o');
                                Ext.util.Cookies.clear('rol_usuario')
;
                                Ext.util.Cookies.clear('nombre_rol_us
uario');
                                Ext.util.Cookies.clear('plan_usuario'
);
                                Ext.util.Cookies.clear('nombre_plan_u
suario');
                                Ext.util.Cookies.clear('id_sistema_us
uario');
```

```
                                Ext.util.Cookies.clear('nombre_sistem
a_usuario');
                                window.location.reload();
                            },
                            failure: function () {
                                Ext.Msg.alert('Error', 'No se pudo
cerrar sesion, contactese con soporte');
                            }
                        });

                    }
                }
            }
        },
```

```
server.get('/salirSesion', (req, res) => {
    try {
        res.clearCookie("id_usuario");
        res.clearCookie("identificacion_usuario");
        res.clearCookie("nombre_usuario");
        res.clearCookie("rol_usuario");
        res.clearCookie("nombre_rol_usuario");
        res.clearCookie("plan_usuario");
        res.clearCookie("nombre_plan_usuario");
        res.clearCookie("id_sistema_usuario");
        res.clearCookie("nombre_sistema_usuario");
        return res.json({ data: 'VARIABLES BORRADAS', success: true,
tipo: 1 });

    } catch (error) {

        return res.json({
            success: false,
            data: error.message
        })
    }
});
```

c. Iteración 3

En la Iteración 3 se procede al desarrollo para funciones administrativas de tickets las cuales son la creación y edición. Durante el desarrollo de esta iteración, se definieron dos funciones administrativas adicionales, la asignación de estados y la asignación de

prioridades, ambas para tickets. A continuación se muestra una tabla resumen de lo que se abarca en esta iteración.

Tabla 75. Resumen Iteración 3

Iteración	N°	Historia de usuario
3	10	Administración de estados para tickets
	11	Administración de prioridades para tickets
	12	Crear ticket
	13	Editar ticket

Historia de usuario 10 – Administración de estados

Tabla 76. Tarea 10 – Administración de estados para tickets

TAREA	
Historia de usuario: #10	
Usuario: Administrador	
Nombre: Administración de estados para tickets	
Prioridad en el negocio: Alta	**Riesgo de desarrollo:** Media
Puntos estimados: 1	**Iteración asignada: 3**
Fecha Inicio: 10/11/2024	**10/11/2024**
Programador a cargo: Patricio Moscoso	
Descripción: El usuario administrador podrá crear, modificar y deshabilitar los estados asignables para los tickets.	

El usuario administrador puede crear, modificar y deshabilitar los estados del sistema. Al crear o modificar un estado, se compara que el nombre del estado no se repita con los estados existentes. Se puede eliminar un estado siempre que no posea tickets asignados con ese mismo estado.

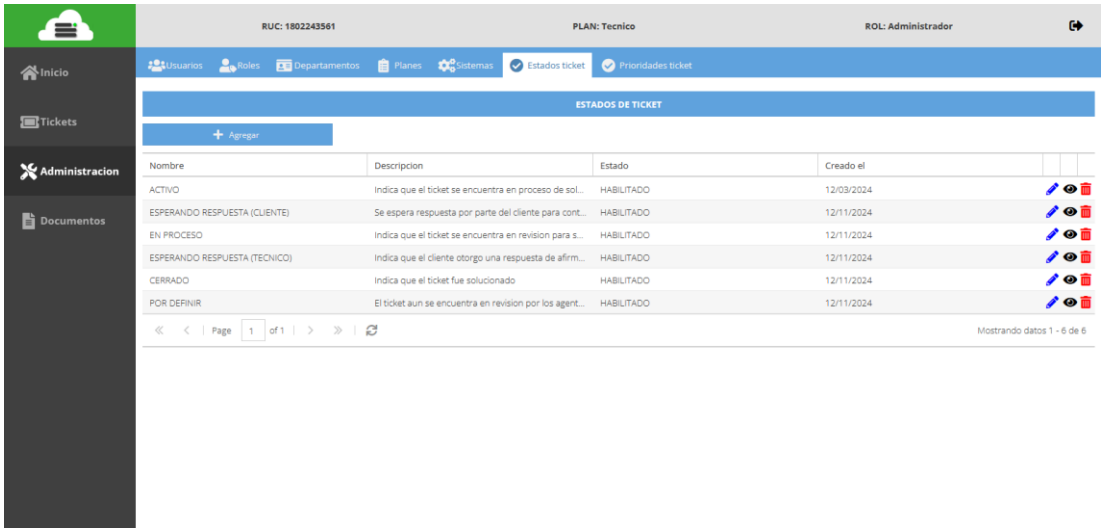


Figura 29. Administración de estados para tickets

```
        {
            xtype: 'label',
            text: 'ESTADOS DE TICKET',
            columnWidth: 1,
            margin: '10 0 10 0',
            padding: 10,
            style: 'background-color: #5fa2dd; color: white; font-
weight: bold; font-size: 30; text-align: center;'
        },
        {
            xtype: 'button',
            text: 'Agregar',
            iconCls: 'fas fa-plus',
            margin: '0 0 10 0',
            columnWidth: 0.2,
            listeners: {
                click: 'onClickAgregarEstado'
            }
        },
        {
            xtype: 'grid',
            id: 'grid_estados_admin',
            columnWidth: 1,
            header: false,
            style: 'border-top: 2px #d0d0d0 solid; border-bottom: 2px
#d0d0d0 solid;',
            store: Ext.create('helpdesk.store.estados.Estado'),
            columns: [
                { text: 'Id', dataIndex: 'id_estado_ticket', hidden:
true, hideable: false },
                { text: 'Nombre', dataIndex: 'nombre_estado_ticket',
flex: 1 },
                { text: 'Descripcion', dataIndex:
'descripcion_estado_ticket', flex: 1},
                { text: 'Id Estado', dataIndex:
'en_uso_estado_ticket', hidden: true, hideable: false },
                { text: 'Estado', dataIndex: 'nombre_estado', flex: 1
},
                { text: 'Creado el', dataIndex:
'fecha_creacion_estado_ticket', formatter: 'date("m/d/Y")', flex: 1
},
                { text: 'Id Usuario', dataIndex:
'id_fkusuario_desactivacion_estado_ticket', hidden: true, hideable:
false },
```

```
                { text: 'Usuario', dataIndex: 'nombre_usuario',
hidden: true, hideable: false },
                { text: 'Desactivado el', dataIndex:
'fecha_desactivacion_estado_ticket', hidden: true, hideable: false },
                {
                    xtype: 'actioncolumn',
                    hideable: false,
                    width: 'auto',
                    items: [
                        {
                            iconCls: 'fas fa-pencil-alt azul',
                            tooltip: 'Editar',
                            display: 'fit',
                            handler: 'onClickEditarEstado'
                        },

                    ]
                },
                {
                    xtype: 'actioncolumn',
                    hideable: false,
                    width: 'auto',
                    items: [
                        {
                            getClass: function (v, meta, rec) {
                                var value =
rec.get('en_uso_estado_ticket')
                                if (value == 1) {
                                    return 'fas fa-eye negro';
                                } else {
                                    return 'fas fa-eye-slash negro';
                                }
                            },
                            handler: 'onClickDesactivarEstado'
                        }
                    ],
                },
                {
                    xtype: 'actioncolumn',
                    hideable: false,
                    width: 'auto',
                    items: [
                        {
                            iconCls: 'fas fa-trash-alt rojo',
                            tooltip: 'Eliminar',
                            handler: 'onClickEliminarEstado'
                        }
                    ]
```

```
                }
            ],
            listeners: {
                select: 'onItemSelected'
            },
            bbar: Ext.create('Ext.PagingToolbar', {
                store: 'Estado',
                displayInfo: true,
                displayMsg: 'Mostrando datos {0} - {1} de {2}',
                emptyMsg: "No hay datos para mostrar",
                inputItemWidth: 35,
            })
        }
```

```
const insertarEstadoTicket = async (req, res, estado) => {
    try {
        sqlInsertar = `select crear_estado_ticket(?,?) as creacion`;

        var [results, fields] = await bd.execute(sqlInsertar, [
            estado.nombre_estado_ticket,
estado.descripcion_estado_ticket
        ]);

        if (results[0].creacion == 1) {
            return res.json({ data: "Estado guardado", success: true,
tipo: 1 });
        } else {
            return res.json({ data: "Error al guardar el estado,
contactese con soporte", success: false, tipo: 2 });
        }
    } catch (error) {
        return res.json({ data: error, success: false, tipo: 2 });

    }
}

const editarEstadoTicket = async (req, res, estado) => {

    try {
        sqlInsertar = `select actualizar_estado_ticket(?,?,?) as
edicion`;

        var [results, fields] = await bd.execute(sqlInsertar, [
            estado.id_estado_ticket, estado.nombre_estado_ticket,
estado.descripcion_estado_ticket
        ]);
```

```
        if (results[0].edicion == 1) {
            return res.json({ data: "Estado guardado", success: true,
tipo: 1 });
        } else {
            return res.json({ data: "Error al guardar el estado,
contactese con soporte", success: false, tipo: 2 });
        }
    } catch (error) {
        return res.json({ data: error, success: false, tipo: 2 });

    }
}

const activarDesactivarEstadoTicket = async (req, res, estado) => {
    try {
        sqlInsertar = `select
desactivar_activar_estado_ticket(?,?,?,?) as edicion`;

        var [results, fields] = await bd.execute(sqlInsertar, [
            estado.id_estado_ticket, estado.en_uso_estado_ticket,
            estado.id_fkusuario_desactivacion_estado_ticket,
estado.motivo_desactivacion_estado_ticket
        ]);

        if (results[0].edicion == 1) {
            return res.json({ data: "Estado actualizado", success:
true, tipo: 1 });
        } else {
            return res.json({ data: "Error al actualizar el estado,
contactese con soporte", success: false, tipo: 2 });
        }
    } catch (error) {
        return res.json({ data: error, success: false, tipo: 2 });

    }
}

const eliminarEstadoTicket = async (req, res, id_estado) => {
    try {
        sqlEliminar = `select eliminar_estado_ticket(?) as eliminar`;
        var [results, fields] = await bd.execute(sqlEliminar,
[id_estado]);

        if (results[0].eliminar == 1) {
            return res.json({ data: "Estado eliminado", success:
true, tipo: 1 });
        } else {
```

```
            return res.json({ data: "Error al eliminar el estado,
contactese con soporte", success: false, tipo: 1 });
        }
    } catch (error) {
        return res.json({ data: error, success: false, tipo: 1 });
    }
}
```

Historia de usuario 11 – Administración de prioridades para tickets

Tabla 77. Tarea 11 – Administración de prioridades para tickets

TAREA	
Historia de usuario: #11	
Usuario: Administrador	
Nombre: Administración de prioridades para tickets	
Prioridad en el negocio: Alta	**Riesgo de desarrollo:** Media
Puntos estimados: 1	**Iteración asignada: 3**
Fecha Inicio: 11/11/2024	**Fecha Fin: 11/11/2024**
Programador a cargo: Patricio Moscoso	
Descripción: El usuario administrador podrá crear, modificar y deshabilitar las prioridades asignables para los tickets.	

El usuario administrador puede crear, modificar, eliminar y deshabilitar una prioridad del sistema. Al crear o modificar una prioridad se controla que el nombre no se repita entre los existentes. Al eliminar se verifica que la prioridad no se encuentre asignada a un ticket.



Figura 30. Administración de prioridades para tickets

```
        {
            xtype: 'label',
            text: 'PRIODIRADES DE TICKET',
            columnWidth: 1,
            margin: '10 0 10 0',
            padding: 10,
            style: 'background-color: #5fa2dd; color: white; font-
weight: bold; font-size: 30; text-align: center;'
        },
        {
            xtype: 'button',
            text: 'Agregar',
            iconCls: 'fas fa-plus',
```

```
            margin: '0 0 10 0',
            columnWidth: 0.2,
            listeners: {
                click: 'onClickAgregarPrioridad'
            }
        },
        {
            xtype: 'grid',
            id: 'grid_prioridades_admin',
            columnWidth: 1,
            header: false,
            style: 'border-top: 2px #d0d0d0 solid; border-bottom: 2px
#d0d0d0 solid;',
            store: Ext.create('helpdesk.store.prioridad.Prioridad'),
            columns: [
                { text: 'Id', dataIndex: 'id_prioridad_ticket',
hidden: true, hideable: false },
                { text: 'Nombre', dataIndex:
'nombre_prioridad_ticket', flex: 1 },
                { text: 'Descripcion', dataIndex:
'descripcion_prioridad_ticket', flex: 1},
                { text: 'Id Prioridad', dataIndex:
'en_uso_prioridad_ticket', hidden: true, hideable: false },
                { text: 'Estado', dataIndex: 'nombre_estado', flex: 1
},
                { text: 'Creado el', dataIndex:
'fecha_creracion_prioridad_ticket', formatter: 'date("m/d/Y")', flex:
1 },
                { text: 'Id Usuario', dataIndex:
'id_fkusuario_desactivacion_prioridad_ticket', hidden: true,
hideable: false },
                { text: 'Usuario', dataIndex: 'nombre_usuario',
hidden: true, hideable: true },
                { text: 'Motivo', dataIndex:
'motivo_desactivacion_prioridad_ticket', hidden: true, hideable: true
},
                { text: 'Desactivado el', dataIndex:
'fecha_desactivacion_prioridad_ticket', hidden: true, hideable: true
},
                {
                    xtype: 'actioncolumn',
                    hideable: false,
                    width: 'auto',
                    items: [
                        {
                            iconCls: 'fas fa-pencil-alt azul',
                            tooltip: 'Editar',
                            display: 'fit',
```

```
                            handler: 'onClickEditarPrioridad'
                        },

                    ]
                },
                {
                    xtype: 'actioncolumn',
                    hideable: false,
                    width: 'auto',
                    items: [
                        {
                            getClass: function (v, meta, rec) {
                                var value =
rec.get('en_uso_prioridad_ticket')
                                if (value == 1) {
                                    return 'fas fa-eye negro';
                                } else {
                                    return 'fas fa-eye-slash negro';
                                }
                            },
                            handler: 'onClickDesactivarPrioridad'
                        }
                    ],
                },
                {
                    xtype: 'actioncolumn',
                    hideable: false,
                    width: 'auto',
                    items: [
                        {
                            iconCls: 'fas fa-trash-alt rojo',
                            tooltip: 'Eliminar',
                            handler: 'onClickEliminarPrioridad'
                        }
                    ]
                }
            ],
            listeners: {
                select: 'onItemSelected'
            },
            bbar: Ext.create('Ext.PagingToolbar', {
                store: 'Prioridad',
                displayInfo: true,
                displayMsg: 'Mostrando datos {0} - {1} de {2}',
                emptyMsg: "No hay datos para mostrar",
                inputItemWidth: 35,
            })
        }
```

```
const insertarPrioridadTicket = async (req, res, prioridad) => {
    try {
        sqlInsertar = `select crear_prioridad_ticket(?,?) as
creacion`;

        var [results, fields] = await bd.execute(sqlInsertar, [
            prioridad.nombre_prioridad_ticket,
prioridad.descripcion_prioridad_ticket
        ]);

        if (results[0].creacion == 1) {
            return res.json({ data: "Prioriad guardada", success:
true, tipo: 1 });
        } else {
            return res.json({ data: "Error al guardar la prioridad,
contactese con soporte", success: false, tipo: 2 });
        }
    } catch (error) {
        return res.json({ data: error, success: false, tipo: 2 });

    }
}

const editarPrioridadTicket = async (req, res, prioridad) => {

    try {
        sqlInsertar = `select actualizar_prioridad_ticket(?,?,?) as
edicion`;

        var [results, fields] = await bd.execute(sqlInsertar, [
            prioridad.id_prioridad_ticket,
prioridad.nombre_prioridad_ticket,
prioridad.descripcion_prioridad_ticket
        ]);

        if (results[0].edicion == 1) {
            return res.json({ data: "Prioridad guardada", success:
true, tipo: 1 });
        } else {
            return res.json({ data: "Error al guardar la prioridad,
contactese con soporte", success: false, tipo: 2 });
        }
    } catch (error) {
        return res.json({ data: error, success: false, tipo: 2 });
    }
}
```

```
const activarDesactivarPrioridadTicket = async (req, res, prioridad)
=> {
    try {
        sqlInsertar = `select
desactivar_activar_prioridad_ticket(?,?,?,?) as edicion`;

        var [results, fields] = await bd.execute(sqlInsertar, [
            prioridad.id_prioridad_ticket,
prioridad.en_uso_prioridad_ticket,
            prioridad.id_fkusuario_desactivacion_prioridad_ticket,
prioridad.motivo_desactivacion_prioridad_ticket
        ]);

        if (results[0].edicion == 1) {
            return res.json({ data: "Prioridad actualizada", success:
true, tipo: 1 });
        } else {
            return res.json({ data: "Error al actualizar la
prioridad, contactese con soporte", success: false, tipo: 2 });
        }
    } catch (error) {
        return res.json({ data: error, success: false, tipo: 2 });

    }
}

const eliminarPrioridadTicket = async (req, res, id_rol) => {
    try {
        sqlEliminar = `select eliminar_prioridad_ticket(?) as
eliminar`;
        var [results, fields] = await bd.execute(sqlEliminar,
[id_rol]);

        if (results[0].eliminar == 1) {
            return res.json({ data: "Prioridad eliminada", success:
true, tipo: 1 });
        } else {
            return res.json({ data: "Error al eliminar la prioridad,
contactese con soporte", success: false, tipo: 1 });
        }
    } catch (error) {
        return res.json({ data: error, success: false, tipo: 1 });
    }
}
```

Historia de usuario 12 – Crear ticket

Tabla 78. Tarea 12 – Crear ticket

TAREA	
Historia de usuario: #12	
Usuario: Todos	
Nombre: Crear ticket	
Prioridad en el negocio: Alta	**Riesgo de desarrollo:** Alta
Puntos estimados: 2	**Iteración asignada: 3**
Fecha Inicio: 12/11/2024	**Fecha Fin: 13/11/2024**
Programador a cargo: Patricio Moscoso	
Descripción: Todos podrán crear un ticket para relatar una incidencia.	
Observación: Al momento de crear un ticket, se asigna un estado y se debe autoasignar a un miembro de la empresa automáticamente	

Al crear un ticket, el sistema asigna automáticamente a un técnico para su respectiva solución, el técnico cambia cada 7 días. Además, se autogenera una sala de chat donde se conecta el cliente y técnico para su posterior conversación.

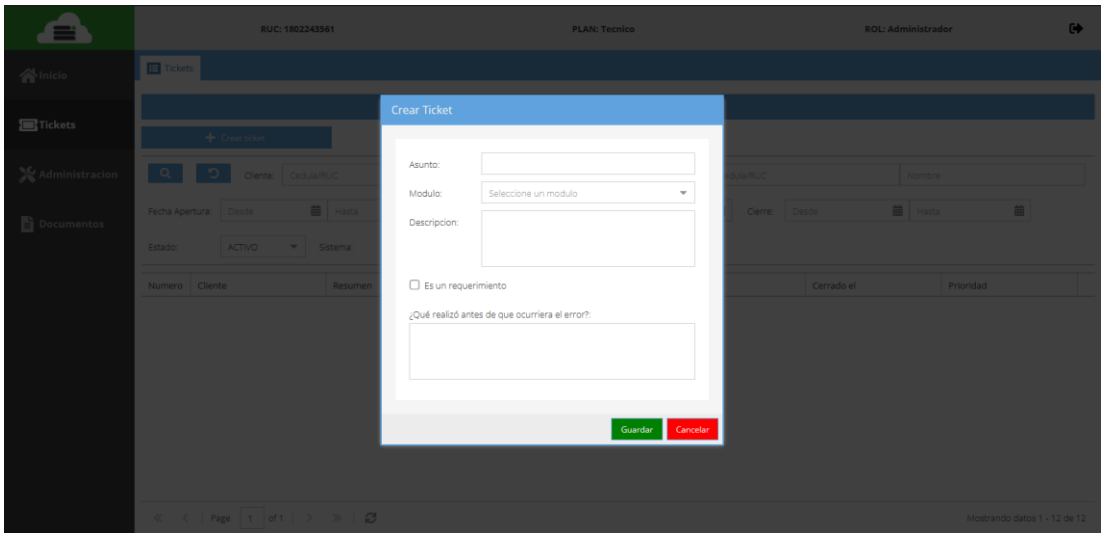


Figura 31. Creación del ticket

```
onClickCrearTicket: function () {
        Ext.create('Ext.window.Window', {
            extend: 'Ext.window.Window',
            alias: 'widget.crearticket',
            modal: true,
            title: 'Crear Ticket',
            layout: 'fit',
            width: 500,
```

```
            closable: false,
            resizable: false,
            bodyStyle: 'background-color: #f5f5f5; padding: 20px;',
            items: [{
                xtype: 'form',
                header: false,
                bodyPadding: 20,
                id: 'form_crear_editar_ticket',
                items: [
                    {
                        xtype: 'textfield',
                        name: 'asunto_ticket',
                        width: '100%',
                        fieldLabel: 'Asunto',
                        allowBlank: false
                    },
                    {
                        xtype: 'combo',
                        editable: false,
                        width: '100%',
                        id: 'modulo_ticket',
                        store:
Ext.create('helpdesk.store.modulos.ModulosActivos'),
                        displayField: 'nombre_modulo_sistema',
                        valueField: 'id_modulo_sistema',
                        fieldLabel: 'Modulo',
                        emptyText: 'Seleccione un modulo',
                        allowBlank: false,
                        name: 'modulo',
                        listeners: {
                            beforequery: function (queryPlan) {
                                var combo = queryPlan.combo;
                                combo.store.filterBy(function
(record) {
                                    return
record.get('id_modulo_sistema') !== -1 &&
record.get('id_modulo_sistema') !== 0;
                                });
                            },
                        }
                    },
                    {
                        xtype: 'textarea',
                        width: '100%',
                        name: 'descripcion_ticket',
                        fieldLabel: 'Descripcion',
                        allowBlank: false
                    },
```

```
                    {
                        xtype: 'checkbox',
                        boxLabel: 'Es un requerimiento',
                        name: 'requerimiento',
                        hideLabel: true,
                        listeners: {
                            change: function (field, newValue) {
                                var datefieldB =
field.up('form').down('#pasos_ticket');
                                if (field.value) {
                                    datefieldB.setValue('Ninguno');
                                    datefieldB.hide();
                                } else {
                                    datefieldB.setValue('');
                                    datefieldB.show();
                                }

                            }
                        }
                    },
                    {
                        xtype: 'textarea',
                        width: '100%',
                        id: 'pasos_ticket',
                        name: 'pasos_ticket',
                        fieldLabel: '¿Qué realizó antes de que
ocurriera el error?',
                        labelAlign: 'top',
                        allowBlank: false,
                    },

                ]
            }],
            buttons: [
                {
                    text: 'Guardar',
                    formBind: true,
                    style: 'background-color: green; color: white;',
                    handler: function (btn) {
                        var form =
Ext.getCmp('form_crear_editar_ticket').getForm();
                        let values = form.getValues();
                        if (form.isValid()) {
                            var id_usuario =
Ext.util.Cookies.get('id_usuario');
                            var cedula_usuario =
Ext.util.Cookies.get('identificacion_usuario');
```

```
                            var nombre_usuario =
Ext.util.Cookies.get('nombre_usuario');
                            var plan =
Ext.util.Cookies.get('plan_usuario');
                            var descripcion =
values.descripcion_ticket;
                            var asunto = values.asunto_ticket;
                            var pasos = values.pasos_ticket;
                            var modulo = values.modulo;
                            var requerimiento = values.requerimiento;
                            Ext.Ajax.request({
                                url:
'http://localhost:8080/tickets/agregarTicket',
                                withCredentials: true,
                                method: 'POST',
                                params: {
                                    id_usuario: id_usuario,
                                    cedula_usuario: cedula_usuario,
                                    nombre_usuario: nombre_usuario,
                                    plan: plan,
                                    descripcion: descripcion,
                                    asunto: asunto,
                                    pasos: pasos,
                                    modulo: modulo,
                                    requerimiento: requerimiento
                                },
                                success: function (response) {
                                    var response_aux =
Ext.util.JSON.decode(response.responseText, true);
                                    if (response_aux.success) {
                                        btn.up('window').destroy();
                                        var grid =
Ext.getCmp('grid_tickets');
                                        var store = grid.getStore();
                                        store.reload();

                                        socket_general.emit('llamado_
actualizar_grids', 'grid_tickets');

                                        construirDetalleTicket(grid,
response_aux.data);

                                    } else {
                                        Ext.Msg.alert('Error', 'Ha
ocurrido un error, contactese con soporte');
                                    }

                                    this.close();
```

```
                                },
                                failure: function () {
                                    Ext.Msg.alert('Error', 'Ha
ocurrido un error, contactese con soporte');
                                }
                            });
                        } else {
                            Ext.Msg.alert('Error', 'Llene todos los
campos');
                        }
                    }
                },
                {
                    text: 'Cancelar',
                    style: 'background-color: red; color: white;',
                    handler: function (btn) {
                        btn.up('window').close();
                    }
                }
            ]
        }).show();
    },
```

```
const crearTicket = async (req, res, ticketIngresado) => {
    try {
        sqlInsertar = `select crear_ticket(?,?,?,?,?,?,?,?,?) as
creacion`;

        var [results, fields] = await bd.execute(sqlInsertar, [
            ticketIngresado.id_fkusuario_ticket,
ticketIngresado.identificacion_usuario_ticket,
ticketIngresado.nombre_usuario_ticket,
            ticketIngresado.id_fkplan_detalle_ticket,
ticketIngresado.descripcion_detalle_ticket,
ticketIngresado.asunto_ticket,
            ticketIngresado.pasos_realizados_ticket,
ticketIngresado.id_fkmodulo_detalle_ticket,
ticketIngresado.es_requerimiento_ticket
        ]);

        if (results[0].creacion > 0) {
            return res.json({ data: results[0].creacion, success:
true, tipo: 1 });
        } else {
            return res.json({ data: "Error al guardar el ticket,
contactese con soporte", success: false, tipo: 2 });
        }
```

```
    } catch (error) {
        return res.json({ data: error, success: false, tipo: 2 });
    }
}

//cambiar tecnico cada x tiempo (5 minutos)
cron.schedule(
    '0 */5 * * * *',
    async () => {
        try {
            var sqlTecnicos = 'select id_usuario, nombre_usuario,
esta_activo_usuario from usuario where id_fkrol_usuario = 2 and
estado_usuario = 1;'
            var [results, fields] = await bd.execute(sqlTecnicos);

            var posicion = 0;
            for (let index = 0; index < results.length; index++) {
                if (results[index].esta_activo_usuario == 1) {
                    posicion = index + 1;
                    if (posicion > results.length - 1) {
                        posicion = 0;
                    }
                }
            }

            var actualizarTecnico = 'select asignar_soporte_agente(?)
as act;'
            var [results2, fields] = await
bd.execute(actualizarTecnico, [results[posicion].id_usuario]);

            if (results2[0].act == 1) {
                console.log('Soporte asignado')
            } else {
                console.log('Error al asignar personal, por favor,
realicelo manualmente.')
            }

        } catch (error) {
            console.log(error)
        }
    }
);
```

Historia de usuario 13 – Editar ticket

Tabla 79. Tarea 13 – Editar ticket

TAREA	
Historia de usuario: #13	
Usuario: Técnicos, Administrador	
Nombre: Editar ticket	
Prioridad en el negocio: Alta	**Riesgo de desarrollo:** Alta
Puntos estimados: 2	**Iteración asignada: 3**
Fecha Inicio: 14/11/2024	**Fecha Fin: 15/11/2024**
Programador a cargo: Patricio Moscoso	
Descripción: Todos los miembros de la empresa pueden editar un ticket.	
Observación: El ticket tiene que cambiar de estado tras realizar una interacción, ya sea por parte del técnico o del cliente.	

Al editar el ticket requirió de la creación de una función por parte del Frontend que cree una vista para la lectura de los mensajes que existen de ese ticket, junto con el detalle del mismo. En la parte inferior de la pantalla se colocó un botón que guarda los cambios realizados en la interfaz.

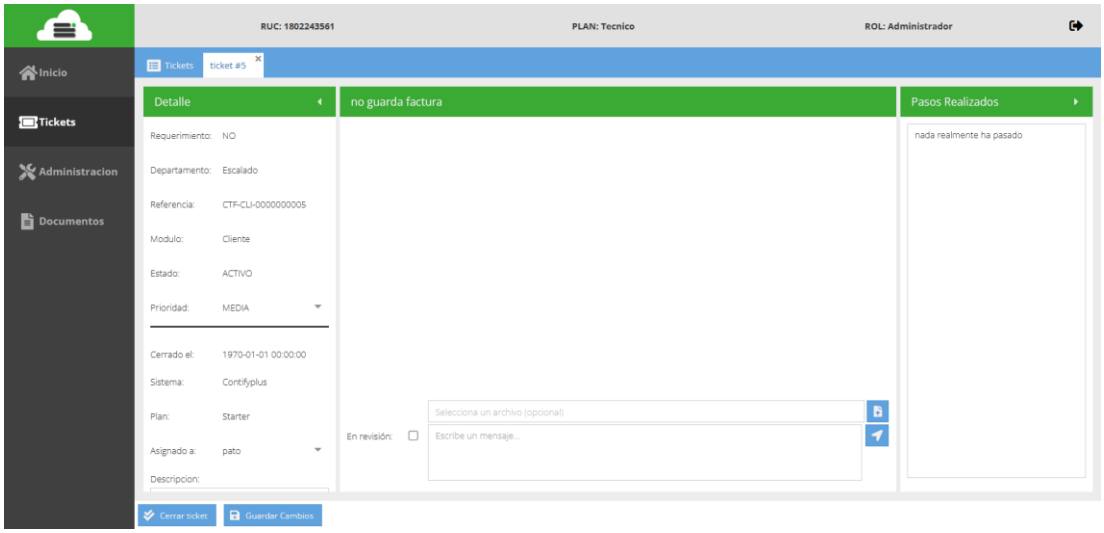


Figura 32. Detalle ticket

```
function construirDetalleTicket(objeto, id_ticket) {
    Ext.Ajax.request({
        withCredentials: true,
        method: 'GET',
        url: 'http://localhost:8080/tickets/obtenerUnTicket',
        params: {
            id: id_ticket,
```

```
        },

        success: function (response) {
            var response_aux =
Ext.util.JSON.decode(response.responseText, true);
            if (response_aux.success) {

                var socket = io('http://localhost:8080');
                var ticket = response_aux.data.tickets;
                var chat = response_aux.data.chats;
                var panel = objeto.up('tabpanel');
                var componente =
Ext.create('helpdesk.view.tickets.DetalleTicket', {
                    id_ticket: ticket.id_ticket,
                    requerimiento:
ticket.nombre_requerimiento_ticket,
                    modulo: ticket.nombre_modulo_detalle_ticket,
                    departamento:
ticket.id_fkdepartamento_detalle_ticket,
                    asunto: ticket.asunto_ticket,
                    referencia: ticket.referencia_detalle_ticket,
                    estado: ticket.id_fkestado_ticket,
                    fecha_cierre: ticket.fecha_cierre_ticket,
                    prioridad: ticket.id_fkprioridad_detalle_ticket,
                    sistema: ticket.id_fksistema_detalle_ticket,
                    plan: ticket.id_fkplan_detalle_ticket,
                    tecnico: ticket.id_fktecnico_ticket,
                    descripcion: ticket.descripcion_detalle_ticket,
                    pasos: ticket.pasos_realizados_ticket,
                    correo_cliente:
ticket.correo_cliente_detalle_ticket,
                    correo_tecnico:
ticket.correo_tecnico_detalle_ticket,
                    mensajes: chat,
                    socket: socket
                });
                panel.insert(1,
                    {
                        title: 'ticket #' + id_ticket,
                        itemId: 'ticket_' + id_ticket,
                        closable: true,
                        xtype: 'container',
                        items: [
                            {
                                xtype: componente,
                            }
                        ],
                        listeners: {
```

```
                            beforeclose: function (element) {
                                socket.emit('leave_chat', id_ticket);
                            }
                        }
                    }
                );

                var chats = response_aux.data.chats;
                var container_mensajes =
Ext.ComponentQuery.query('container[itemId=container_mensajes_' +
ticket.id_ticket + ']')[0];

                chats.forEach((data) => {
                    if (!data.usuario) {
                        return;
                    }

                    var fecha = new
Date(data.fecha).toISOString().split('T')[0] + ' ' + new
Date(data.fecha).toISOString().split('T')[1].replace(/\.\d+Z$/, '');

                    var archivos =
data.path_archivos.replace(/[\[\]]/g, '')
                    archivos = archivos.length == 0 ? [] :
archivos.split(',');

                    var nombres =
data.nombres_archivos.replace(/[\[\]]/g, '')
                    nombres = nombres.length == 0 ? [] :
nombres.split(',');

                    container_mensajes.insert(0, {
                        xtype: 'container',
                        layout: 'column',
                        width: '100%',
                        padding: 10,
                        margin: 10,
                        style: {
                            backgroundColor: data.usuario ==
Ext.util.Cookies.get('nombre_usuario') ? '#e0f1ff' : '#eaeaea'
                        },
                        items: [
                            {
                                xtype: 'button',
                                columnWidth: 0.05,
                                padding: 10,
                                iconCls: 'fas fa-user',
                                disabled: true
```

```
                            },
                            {
                                xtype: 'container',
                                columnWidth: 0.95,
                                layout: 'vbox',
                                items: [
                                    {
                                        xtype: 'label',
                                        margin: 5,
                                        text: data.id,
                                        hidden: true
                                    },
                                    {
                                        xtype: 'label',
                                        margin: 5,
                                        text: data.usuario
                                    },
                                    {
                                        xtype: 'label',
                                        margin: 5,
                                        width: '100%',
                                        text: data.mensaje,
                                        readOnly: true
                                    },
                                    data.contiene_archivo == 1 ? {
                                        xtype: 'container',
                                        margin: 5,
                                        width: '100%',
                                        items: archivos.map(function
(file, posicion) {
                                            return {
                                                xtype: 'button',
                                                margin: 5,
                                                text:
nombres[posicion], // Usa el nombre del archivo del array
                                                iconCls: 'fas fa-
file',
                                                layout: 'fit',
                                                handler: function ()
{
                                                    Ext.create('Ext.w
indow.Window', {
                                                        title:
nombres[posicion],
                                                        layout: {
                                                            align:
'middle',
```

```
                                                                pack:
'center',
                                                                type:
'hbox'
                                                            },
                                                            height: 500,
                                                            width: 800,
                                                            listeners: {
                                                                show:
function () {
                                                                    var
myFrame = Ext.create('Ext.Component', {
                                                                        h
eight: 500,
                                                                        w
idth: 800,
                                                                        a
utoEl: {

  tag: 'iframe',

  src: 'http://localhost:8080/reportes/obtenerDocumento?path=' +
archivos[posicion]
                                                                    }
                                                                });
                                                                this.
add(myFrame);
                                                            }
                                                        }
                                                    }).show();
                                                }
                                            };
                                        })
                                    } : {
                                        xtype: 'container',
                                    },
                                    {
                                        xtype: 'label',
                                        margin: 5,
                                        text: fecha
                                    },
                                ]
                            }
                        ]
                    });
                });
```

```
                container_mensajes.scrollTop =
container_mensajes.scrollHeight;


                var boton_enviar_mensaje =
Ext.ComponentQuery.query('button[itemId=enviarMensaje' +
ticket.id_ticket + ']')[0];
                boton_enviar_mensaje.fireEvent('click');


                panel.setActiveTab(1);
            } else {
                Ext.Msg.alert('Fallo', 'Error al actualizar el
usuario, contactese con soporte');
            }

        },
        failure: function () {
            Ext.Msg.alert('Error', 'Error al actualizar el usuario,
contactese con soporte');
        }
    });
};

//boton para actualizar el ticket
{
                    xtype: 'button',
                    text: 'Guardar Cambios',
                    iconCls: 'fas fa-save',
                    disabled: this.estado == 6 ? true : false,
                    margin: 5,
                    listeners: {
                        click: function (btn) {
                            var panel = btn.up('panel');
                            var container_detalle =
Ext.ComponentQuery.query('panel[itemId=panel_detalle_' +
panel.id_ticket + ']')[0];
                            var container_pasos =
Ext.ComponentQuery.query('panel[itemId=panel_pasos_' +
panel.id_ticket + ']')[0];
                            var values_detalle = {}
                            var values_pasos = {}

                            container_detalle.query('[isFormField]').
forEach(function (field) {
                                values_detalle[field.getName()] =
field.getValue();
                            });
```

```
                            container_pasos.query('[isFormField]').fo
rEach(function (field) {
                                values_pasos[field.getName()] =
field.getValue();
                            });

                            Ext.Ajax.request({
                                url:
'http://localhost:8080/tickets/modificarTicket',
                                withCredentials: true,
                                method: 'POST',
                                params: {
                                    id: panel.id_ticket,
                                    estado: panel.estado,
                                    descripcion:
values_detalle.descripcion,
                                    tecnico: values_detalle.tecnico,
                                    prioridad:
values_detalle.prioridad,
                                    pasos: values_pasos.pasos,
                                    sistema: panel.sistema
                                },
                                success: function (response) {
                                    var response_aux =
Ext.util.JSON.decode(response.responseText, true);
                                    if (response_aux.success) {
                                        var grid =
Ext.getCmp('grid_tickets');
                                        var store = grid.getStore();
                                        store.reload();

                                        var actualizar = {
                                            estado: panel.estado,
                                            prioridad:
values_detalle.prioridad,
                                            fecha: '',
                                            tecnico:
values_detalle.tecnico,
                                            descripcion:
values_detalle.descripcion,
                                            pasos:
values_pasos.pasos,
                                        }

                                        socket_general.emit('llamado_
actualizar_grids', 'grid_tickets');
```

```
                                        socket_general.emit('llamado_
actualizar_tickets', panel.id_ticket, actualizar, 0);

                                        Ext.Msg.alert('Exito',
'¡Ticket actualizado!');


                                    } else {
                                        Ext.Msg.alert('Error', 'Ha
ocurrido un error, contactese con soporte');
                                    }

                                    this.close();
                                },
                                failure: function () {
                                    Ext.Msg.alert('Error', 'Ha
ocurrido un error, contactese con soporte');
                                }
                            });
                        }
                    }
                }
```

```
const actualizarTicket = async (req, res, ticketIngresado) => {
    try {
        sqlInsertar = `select actualizar_ticket(?,?,?,?,?,?,?) as
creacion`;

        var [results, fields] = await bd.execute(sqlInsertar, [
            ticketIngresado.id_ticket,
ticketIngresado.id_fkestado_ticket,
ticketIngresado.descripcion_detalle_ticket,
            ticketIngresado.id_fkprioridad_detalle_ticket,
ticketIngresado.pasos_realizados_ticket,
ticketIngresado.id_fksistema_detalle_ticket,
ticketIngresado.id_fktecnico_ticket,
        ]);

        if (results[0].creacion == 1) {
            return res.json({ data: results[0].creacion, success:
true, tipo: 1 });
        } else {
            return res.json({ data: "Error al actualizar el ticket,
contactese con soporte", success: false, tipo: 2 });
        }
    } catch (error) {
        return res.json({ data: error, success: false, tipo: 2 });
```

```
    }
}
```

d. Iteración 4

En la Iteración 4 se desarrolló las funciones de traspasar ticket, cerrar ticket, Reaperturar ticket y el cambio de estado de tickets. Las funciones relatadas ocurren tras la creación de un ticket y fue necesario el culminar la funcionalidad de mostrar detalle para la implementación. A continuación se muestra una tabla resumen de lo que se abarca en esta iteración.

Tabla 80. Resumen Iteración 4

Iteración	N°	Historia de usuario
4	14	Traspasar ticket
	15	Cerrar ticket
	16	Reaperturar ticket
	17	Cambio de estado del ticket

Historia de usuario 14 – Traspasar ticket

Tabla 81. Tarea 14 – Traspasar ticket

TAREA	
Historia de usuario: #14	
Usuario: Técnicos, Administrador	
Nombre: Traspasar ticket	
Prioridad en el negocio: Alta	**Riesgo de desarrollo:** Alta
Puntos estimados: 2	**Iteración asignada: 4**
Fecha Inicio: 16/11/2024	**Fecha Fin: 17/11/2024**
Programador a cargo: Patricio Moscoso	
Descripción: Todos los miembros de la empresa pueden traspasar un ticket a otro miembro de la empresa.	
Observación: El ticket puede traspasarse las veces que sean necesarias, no posee un límite de traspasos.	

La función de traspaso se implementó dentro del servicio de actualizar ticket, por lo que solo se requirió el aumento del campo id_fktecnico_ticket en los parámetros a recibir.

Historia de usuario 15 – Cerrar ticket

Tabla 82. Tarea 15 – Cerrar ticket

TAREA	
Historia de usuario: #15	
Usuario: Técnicos, Administrador	
Nombre: Cerrar ticket	
Prioridad en el negocio: Alta	**Riesgo de desarrollo:** Alta
Puntos estimados: 2	**Iteración asignada: 4**
Fecha Inicio: 18/11/2024	**Fecha Fin: 19/11/2024**
Programador a cargo: Patricio Moscoso	
Descripción: Todos los miembros de la empresa pueden finalizar un ticket.	

El cerrar ticket se implementó como botón acompañante de Editar ticket en la parte inferior de la pantalla, al cerrar el ticket, tanto el formulario de enviar mensaje como los datos editables del ticket son deshabilitados, además de habilitar el botón de reaperturar ticket para los clientes.



Figura 33. Cerrar ticket

```
{
                    xtype: 'button',
                    text: 'Cerrar ticket',
                    itemId: 'btn_cerrar_ticket_' +
this.parent.id_ticket,
                    disabled: this.parent.estado == 6 ? true : false,
                    iconCls: 'fas fa-check-double',
                    margin: 5,
                    listeners: {
                        click: function (btn) {
                            var panel = btn.up('panel');
                            var container_pasos =
Ext.ComponentQuery.query('panel[itemId=panel_pasos_' +
panel.id_ticket + ']')[0];
                            var values_pasos = {}
                            container_pasos.query('[isFormField]').fo
rEach(function (field) {
                                values_pasos[field.getName()] =
field.getValue();
                            });

                            Ext.Ajax.request({
```

```
                                url:
'http://localhost:8080/tickets/finalizarTicket',
                                withCredentials: true,
                                method: 'POST',
                                params: {
                                    id: panel.id_ticket,
                                    pasos: values_pasos.pasos,
                                    correo_cliente:
panel.correo_cliente
                                },
                                success: function (response) {
                                    var response_aux =
Ext.util.JSON.decode(response.responseText, true);
                                    if (response_aux.success) {
                                        var grid =
Ext.getCmp('grid_tickets');
                                        var store = grid.getStore();
                                        store.reload();
                                        socket_general.emit('llamado_
actualizar_grids', 'grid_tickets');

                                        var actualizar = {
                                            estado: 6,
                                        }

                                        socket_general.emit('llamado_
actualizar_tickets', panel.id_ticket, actualizar, 2);

                                        Ext.Msg.alert('Exito',
'¡Ticket finalizado!');

                                    } else {
                                        Ext.Msg.alert('Error', 'Ha
ocurrido un error, contactese con soporte');
                                    }

                                    this.close();
                                },
                                failure: function () {
                                    Ext.Msg.alert('Error', 'Ha
ocurrido un error, contactese con soporte');
                                }
                            });
                        }
                    }
                },
```

```
const cerrarTicket = async (req, res, ticketIngresado,
correo_cliente) => {
    try {
        sqlInsertar = `select cerrar_ticket(?,?) as creacion`;

        var [results, fields] = await bd.execute(sqlInsertar, [
            ticketIngresado.id_ticket,
ticketIngresado.pasos_realizados_ticket
        ]);

        emailHelper(
            correo_cliente,
            'Su ticket #' + ticketIngresado.id_ticket + ' ha sido
finalizado.',
            `<span><b>Ticket #${ticketIngresado.id_ticket}</b></br>
            </br>
            Mensaje :Ticket finalizado</br>
            </br>
            ¡Si necesita mas detalles o tiene mas novedades, no dude
en reaperturar el ticket!</br>
            </br>
            </span>`
        );

        if (results[0].creacion == 1) {
            return res.json({ data: 'Ticket finalizado.', success:
true, tipo: 1 });
        } else {
            return res.json({ data: "Error al actualizar el ticket,
contactese con soporte", success: false, tipo: 2 });
        }
    } catch (error) {
        console.log(error)
        return res.json({ data: error, success: false, tipo: 2 });
    }
}
```

Historia de usuario 16 – Reaperturar ticket

Tabla 83. Tarea 16 – Reaperturar ticket

TAREA	
Historia de usuario: #16	
Usuario: Usuarios	
Nombre: Reaperturar ticket	
Prioridad en el negocio: Alta	**Riesgo de desarrollo:** Alta

Puntos estimados: 2	**Iteración asignada: 4**
Fecha Inicio: 20/11/2024	**Fecha Fin: 21/11/2024**
Programador a cargo: Patricio Moscoso	
Descripción: Solo el usuario puede reabrir un ticket	
Observación: El ticket tiene que estar finalizado para poder ser reaperturado.	

Al habilitar el botón de reaperturar ticket, al presionarse, se reactiva el formulario para enviar mensajes, los campos editables del detalle del ticket para los técnicos, además de cambiar el estado y notificando al último técnico a cargo del mismo.

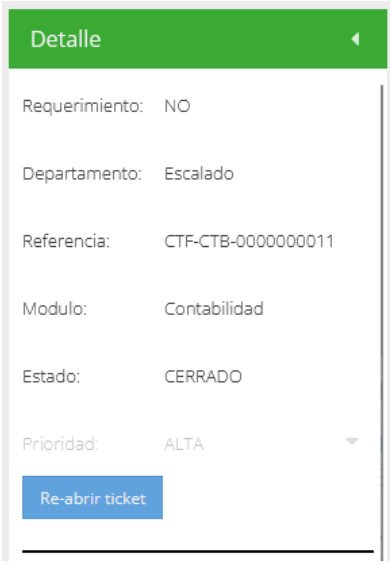


Figura 34. Reaperturar ticket

```
                        {
                            xtype: 'button',
                            layout: 'fit',
                            itemId: 'btn_detalle_reabrir_' +
componente.id_ticket,
                            text: 'Re-abrir ticket',
                            margin: '0 0 20 0',
                            listeners: {
                                click: function (btn) {
                                    Ext.create('Ext.window.Window
', {
                                        title: 'Abrir ticket',
                                        layout: 'fit',
                                        modal: true,
                                        layout: 'fit',
                                        width: 500,
```

```
                                                bodyStyle: 'background-
color: #f5f5f5; padding: 20px;',
                                                items: [{
                                                    xtype: 'textarea',
                                                    labelAlign: 'top',
                                                    fieldLabel: '¿Que
sucedió?',
                                                    emptyText:
'Cuentanos...',
                                                    height: 100,
                                                }],
                                                buttons: [
                                                    {
                                                        text: 'Aceptar',
                                                        style:
'background-color: green; color: white;',
                                                        handler: function
(btn) {
                                                            var
descripcion = btn.up('window').down('textarea');
                                                            Ext.Ajax.requ
est({
                                                                withCrede
ntials: true,
                                                                method:
'POST',
                                                                url:
'http://localhost:8080/tickets/reabrirTicket',
                                                                params: {
                                                                    id:
componente.id_ticket,
                                                                    descr
ipcion: descripcion.value
                                                                },

                                                                success:
function (response) {
                                                                    var
response_aux = Ext.util.JSON.decode(response.responseText, true);
                                                                    if
(response_aux.success) {
                                                                        b
tn.up('window').destroy();
                                                                        E
xt.Msg.alert('Éxito', 'Ticket reaperturado');
                                                                        v
ar grid = Ext.getCmp('grid_tickets');
```

```
                                                                    v
ar store = grid.getStore();
                                                                    s
tore.reload();

                                                                    s
ocket_general.emit('llamado_actualizar_grids', 'grid_tickets');

                                                                    v
ar actualizar = {

  estado: 2,
                                                                    }

                                                                    s
ocket_general.emit('llamado_actualizar_tickets',
componente.id_ticket, actualizar, 2);

                                                                    t
his.close();
                                                                }
else {
                                                                    E
xt.Msg.alert('Fallo', 'Error al actualizar el ticket, contactese con
soporte');
                                                                }

                                                            },
                                                            failure:
function () {
                                                                Ext.M
sg.alert('Error', 'Error al actualizar el ticket, contactese con
soporte');
                                                            }
                                                        });
                                                    }
                                                },
                                                {
                                                    text: 'Cancelar',
                                                    style:
'background-color: red; color: white;',
                                                    handler: function
(btn) {
                                                        btn.up('windo
w').close();
                                                    }
                                                }
                                            ]
```

```
                                        }).show();
                                    }
                                },
                                hidden: componente.estado == 6 ?
false : true,
                            },
```

```
const reaperturarTicket = async (req, res, ticketIngresado) => {
    try {
        sqlInsertar = `select reaperturar_ticket(?,?) as creacion`;

        var [results, fields] = await bd.execute(sqlInsertar, [
            ticketIngresado.id_ticket,
ticketIngresado.descripcion_detalle_ticket
        ]);

        if (results[0].creacion == 1) {
            return res.json({ data: 'Ticket Reaperturado.', success:
true, tipo: 1 });
        } else {
            return res.json({ data: "Error al actualizar el ticket,
contactese con soporte", success: false, tipo: 2 });
        }
    } catch (error) {
        console.log(error)
        return res.json({ data: error, success: false, tipo: 2 });
    }
}
```

Historia de usuario 17 – Cambio de estado del ticket

Tabla 84. Tarea 17 – Cambio de estado del ticket

TAREA	
Historia de usuario: #17	
Usuario: --	
Nombre: Cambio de estado del ticket	
Prioridad en el negocio: Alta	**Riesgo de desarrollo:** Alta
Puntos estimados: 2	**Iteración asignada: 4**
Fecha Inicio: 22/11/2024	**Fecha Fin: 22/11/2024**
Programador a cargo: Patricio Moscoso	
Descripción: El ticket cambia de estado por cada respuesta que se efectué.	
Observación: El ticket tiene que estar aperturado para realizar cualquier cambio.	

La actualización del estado fue creado conjuntamente con el servicio de mensajería instantánea, el estado del ticket cambia cada que se presionase la tecla *Enter* o se de click en el botón enviar, se controla que el textarea de mensaje no este vacío.



Figura 35. Formulario para envio de mensajes

```
                                    {
                                        xtype: 'button',
                                        width: '100%',
                                        margin: '0 0 0 5',
                                        columnWidth: 0.05,
                                        itemId: 'enviarMensaje' +
componente.id_ticket,
                                        disabled: componente.estado
== 6 ? true : false,
                                        iconCls: 'fas fa-location-
arrow',
                                        listeners: {
                                            click: function () {
                                                var containerValores
= this.up('container');
                                                var values = {};
                                                containerValores.quer
y('[isFormField]').forEach(function (field) {
                                                    values[field.getN
ame()] = field.getValue();
                                                });

                                                //verificar si existe
un archivo subido
                                                var filefield =
containerValores.down('filefield');
                                                var file = null;
                                                var contiene_archivo
= 0;
                                                var nombres_archivos
= [];
                                                var paths_archivos =
[];
```

```
                                                var bufferArchivo =
'';
                                                if (filefield.el) {
                                                    file =
filefield.el.down('input[type=file]').dom.files[0];
                                                    if (file) {
                                                        const blob =
new Blob([file]);
                                                        var reader =
new FileReader();

                                                        reader.onload
= e => {
                                                            const
dataURL = reader.result;
                                                            bufferArc
hivo = dataURL.slice(dataURL.indexOf(',') + 1);
                                                        };

                                                        reader.readAs
DataURL(blob);

                                                        nombres_archi
vos.push(filefield.el.down('input[type=file]').dom.files[0].name);
                                                        paths_archivo
s.push(filefield.el.down('input[type=file]').dom.files[0].webkitRelat
ivePath);
                                                        contiene_arch
ivo = 1;

                                                        //limpiar
filefield
                                                        filefield.res
et();
                                                        filefield.set
RawValue("");
                                                    }
                                                }

                                                //verificar que se
envie un mensaje
                                                var texto =
containerValores.down('textarea').getValue();
                                                if (texto.length == 0
&& constructor == 1) {
                                                    Ext.Msg.alert('Ad
vertencia', 'No se va a enviar ningun mensaje')
                                                    return;
```

```
                                                    }

                                                    //crear mensaje y
enviar al server
                                                    containerValores.down
('textarea').setValue('');
                                                    var fecha_actual =
new Date(Date.now()).toISOString()
                                                    var mensaje = {
                                                        id: id_generado,
                                                        usuario:
Ext.util.Cookies.get('nombre_usuario'),
                                                        texto: texto,
                                                        contiene_archivo:
contiene_archivo,
                                                        paths:
paths_archivos,
                                                        nombres_archivos:
nombres_archivos,
                                                        fecha:
fecha_actual,
                                                        id_ticket:
componente.id_ticket,
                                                        rol:
Ext.util.Cookies.get('rol_usuario'),
                                                        correo_cliente:
componente.correo_cliente,
                                                        correo_tecnico:
componente.correo_tecnico,
                                                        revision:
values['es_revision']
                                                    };
                                                    socket_chat.emit('cha
t message', mensaje, file);
                                                    id_generado++;
                                                    var grid =
Ext.getCmp('grid_tickets');
                                                    var store =
grid.getStore();
                                                    store.reload();
                                                    socket_general.emit('
llamado_actualizar_grids', 'grid_tickets');
                                                    constructor = 1;
                                                    return;
                                                }
```

```
                                        }
                                    },
```

```
var es_revision = mensaje.revision == true ? 1 : 0;
            try {
                var [results, fields] = await bd.execute('select
autocambiar_estado_ticket(?,?,?)', [mensaje.id_ticket, mensaje.rol,
es_revision]);

                insercion = results.affectedRows;
            } catch (error) {
                console.log(error)
                insercion = -2;
            }
```

e. Iteración 5

En la Iteración 5 se desarrolló la visualización de tickets junto con el servicio de mensajería instantánea para los chats de cada ticket, además de la implementación de notificaciones por correo y la elaboración de los reportes del sistema. A continuación se muestra una tabla resumen de lo que se abarca en esta iteración.

Tabla 85. Resumen Iteración 5

Iteración	N°	Historia de usuario
5	18	Visualización de tickets
	19	Mensajeria instantanea para chats
	20	Notificación de tickets
	21	Tutoriales
	22	Reportes

Historia de usuario 18 – Visualización de tickets

Tabla 86. Tarea 18 – Visualización de tickets

HISTORIA DE USUARIO		#18
Historia de usuario: #18		
Usuario: Todos		
Nombre: Visualización de tickets		
Prioridad en el negocio: Alta	**Riesgo de desarrollo:** Alta	
Puntos estimados: 1	**Iteración asignada: 5**	
Fecha Inicio: 23/11/2024	**Fecha Fin: 23/11/2024**	
Programador a cargo: Patricio Moscoso		
Descripción: Pantalla para visualizar el detalle del ticket		

La pantalla de visualización presenta los datos de todos los tickets, un resumen del cliente que lo creo y los datos más relevantes del ticket, además el botón de ojo permite ver el detalle del ticket junto con el chat que este posee.

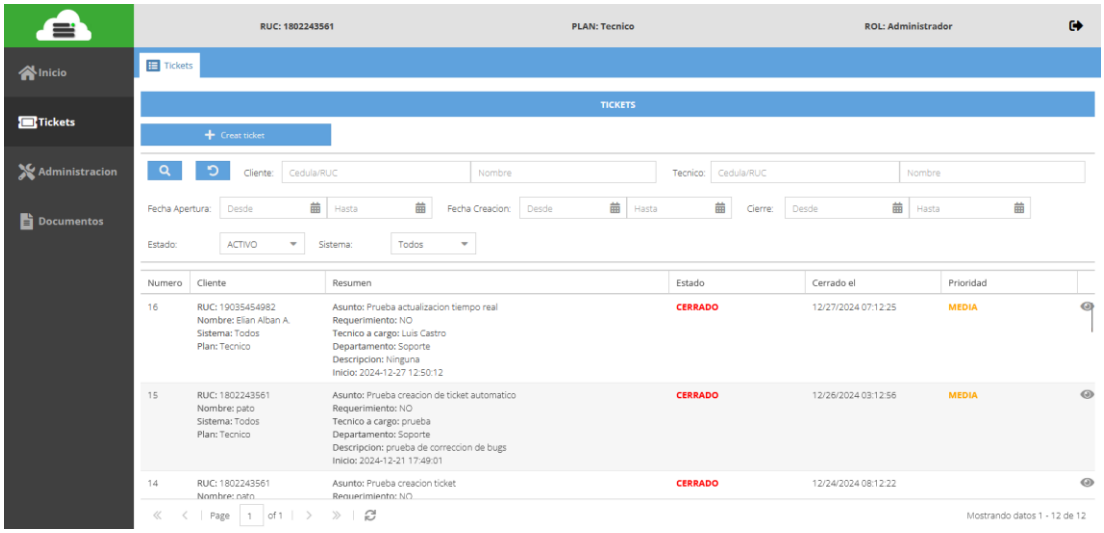


Figura 36. Visualización de tickets

```
        {
            xtype: 'label',
            text: 'TICKETS',
            columnWidth: 1,
            margin: '10 0 10 0',
            padding: 10,
            style: 'background-color: #5fa2dd; color: white; font-
weight: bold; font-size: 30; text-align: center;'
        },
        {
            xtype: 'button',
            text: 'Creat ticket',
            itemId: 'button_crear_ticket',
            iconCls: 'fas fa-plus',
            columnWidth: 0.2,
            listeners: {
                click: 'onClickCrearTicket'
            }
        },
        {
            xtype: 'panel',
            id: 'campos_buscar_ticket',
```

```
            width: '100%',
            layout: 'column',
            //title: 'Filtros',
            columnWidth: 1,
            //collapsible: true,
            margin: '10 0 10 0',
            height: Ext.util.Cookies.get('rol_usuario') != 3 ? 160 :
110,
            style: 'border-top: 2px #d0d0d0 solid; border-bottom: 2px
#d0d0d0 solid; height: auto;',
            cls: 'cabecera',
            items: [
                {
                    xtype: 'button',
                    iconCls: 'fas fa-search',
                    width: 50,
                    padding: 5,
                    margin: 10,
                    listeners: {
                        click: 'onClickBuscarPorCampos'
                    }
                },
                {
                    xtype: 'button',
                    iconCls: 'fas fa-undo-alt',
                    width: 50,
                    padding: 5,
                    margin: 10,
                    listeners: {
                        click: 'onClickLimpiarCampos'
                    }
                },
                {
                    xtype: 'fieldcontainer',
                    fieldLabel: 'Cliente',
                    columnWidth: 0.5,
                    labelWidth: 50,
                    padding: 10,
                    layout: 'hbox',
                    items: [
                        {
                            xtype: 'textfield',
                            name: 'identificacion_cliente',
                            emptyText: 'Cedula/RUC',
                            width: '50%',
                            maskRe: /[0-9]/,
                        },
                        {
```

```
                            xtype: 'textfield',
                            name: 'nombre_cliente',
                            width: '49%',
                            emptyText: 'Nombre'
                        },
                    ],
                    hidden: Ext.util.Cookies.get('rol_usuario') == 3
? true : false,
                },
                {
                    xtype: 'fieldcontainer',
                    fieldLabel: 'Tecnico',
                    columnWidth: 0.5,
                    labelWidth: 50,
                    padding: 10,
                    layout: 'hbox',
                    items: [
                        {
                            xtype: 'textfield',
                            name: 'identificacion_tecnico',
                            emptyText: 'Cedula/RUC',
                            width: '50%',
                            maskRe: /[0-9]/,
                        },
                        {
                            xtype: 'textfield',
                            name: 'nombre_tecnico',
                            width: '49%',
                            emptyText: 'Nombre'
                        },
                    ],
                    hidden: Ext.util.Cookies.get('rol_usuario') == 3
? true : false,
                },
                {
                    xtype: 'fieldcontainer',
                    fieldLabel: 'Fecha Apertura',
                    columnWidth: 0.35,
                    padding: 10,
                    layout: 'hbox',
                    items: [
                        {
                            xtype: 'datefield',
                            name: 'fecha_inicio_apertura',
                            id: 'fecha_inicio_apertura',
                            emptyText: 'Desde',
                            width: '50%',
                            listeners: {
```

```
                                change: function (field, newValue) {
                                    var datefieldB =
field.up('container').down('#fecha_fin_apertura');
                                    datefieldB.setMinValue(newValue);
                                }
                            }
                        },
                        {
                            xtype: 'datefield',
                            name: 'fecha_fin_apertura',
                            id: 'fecha_fin_apertura',
                            width: '49%',
                            emptyText: 'Hasta'
                        },
                    ]
                },
                {
                    xtype: 'fieldcontainer',
                    fieldLabel: 'Fecha Creacion',
                    columnWidth: 0.35,
                    padding: 10,
                    layout: 'hbox',
                    items: [
                        {
                            xtype: 'datefield',
                            name: 'fecha_inicio_creacion',
                            id: 'fecha_inicio_creacion',
                            emptyText: 'Desde',
                            width: '50%',
                            listeners: {
                                change: function (field, newValue) {
                                    var datefieldB =
field.up('container').down('#fecha_fin_creacion');
                                    datefieldB.setMinValue(newValue);
                                }
                            }
                        },
                        {
                            xtype: 'datefield',
                            name: 'fecha_fin_creacion',
                            id: 'fecha_fin_creacion',
                            width: '49%',
                            emptyText: 'Hasta'
                        },
                    ]
                },
                {
                    xtype: 'fieldcontainer',
```

```
                    fieldLabel: 'Cierre',
                    columnWidth: 0.35,
                    labelWidth: 50,
                    padding: 10,
                    layout: 'hbox',
                    items: [
                        {
                            xtype: 'datefield',
                            name: 'fecha_inicio_cierre',
                            id: 'fecha_inicio_cierre',
                            emptyText: 'Desde',
                            width: '50%',
                            listeners: {
                                change: function (field, newValue) {
                                    var datefieldB =
field.up('container').down('#fecha_fin_cierre');
                                    datefieldB.setMinValue(newValue);
                                }
                            }
                        },
                        {
                            xtype: 'datefield',
                            name: 'fecha_fin_cierre',
                            id: 'fecha_fin_cierre',
                            width: '49%',
                            emptyText: 'Hasta'
                        },
                    ]
                },
                {
                    xtype: 'combo',
                    editable: false,
                    columnWidth: 0.2,
                    padding: 10,
                    id: 'estado_ticket',
                    store:
Ext.create('helpdesk.store.estados.EstadosActivos'),
                    displayField: 'nombre_estado_ticket',
                    valueField: 'id_estado_ticket',
                    fieldLabel: 'Estado',
                    name: 'estado_ticket',
                    listeners: {
                        afterrender: function (combo) {
                            var value = combo.getStore().getAt(0)
                            combo.setValue(value.get('id_estado_ticke
t'))
                        }
                    }
```

```
                },
                {
                    xtype: 'combo',
                    editable: false,
                    columnWidth: 0.2,
                    padding: 10,
                    id: 'sistema_ticket',
                    store:
Ext.create('helpdesk.store.sistema.SistemaActivos'),
                    displayField: 'nombre_sistema',
                    valueField: 'id_sistema',
                    value: 1,
                    fieldLabel: 'Sistema',
                    name: 'sistema_ticket',
                    listeners: {
                        afterrender: function (combo) {
                            var value = combo.getStore().getAt(0)
                            combo.setValue(value.get('id_sistema'))
                        }
                    },
                    hidden: Ext.util.Cookies.get('rol_usuario') == 3
? true : false,
                },
            ]
        },
        {
            xtype: 'grid',
            id: 'grid_tickets',
            store: Ext.create('helpdesk.store.ticket.Ticket'),
            columnWidth: 1,
            margin: '0 0 5 0',
            scrollable: true,
            header: false,
            style: 'border-top: 2px #d0d0d0 solid; border-bottom: 2px
#d0d0d0 solid;',
            columns: [
                { text: 'Numero', dataIndex: 'id_ticket', width:
'auto' },
                { text: 'Id Usuario', dataIndex:
'id_fkusuario_ticket', hidden: true, hideable: false },
                { text: 'Id Tecnico', dataIndex:
'id_fktecnico_ticket', hidden: true, hideable: false, width: 'auto'
},
                {
                    text: 'Cliente',
                    dataIndex: 'id_fkusuario_ticket',
                    flex: 1,
                    renderer: function (value, metaData, record) {
```

```
                        value = record
                        const identificacion =
record.get('identificacion_usuario_ticket');
                        const nombre =
record.get('nombre_usuario_ticket');
                        const sistema =
record.get('nombre_sistema_detalle_ticket');
                        const plan =
record.get('nombre_plan_detalle_ticket');

                        return `
                            <div>
                                <span><b>RUC:</b>
${identificacion}</span><br>
                                <span><b>Nombre:</b>
${nombre}</span><br>
                                <span><b>Sistema:</b>
${sistema}</span><br>
                                <span><b>Plan:</b> ${plan}</span><br>
                            </div>
                        `;
                    },
                },
                {
                    text: 'Resumen',
                    dataIndex: 'id_fkusuario_ticket',
                    width: 500,
                    renderer: function (value, metaData, record) {
                        const asunto = record.get('asunto_ticket');
                        const requerimiento =
record.get('nombre_requerimiento_ticket');
                        const tecnico_a_cargo =
record.get('nombre_tecnico_ticket');
                        const descripcion =
record.get('descripcion_detalle_ticket');
                        const departamento =
record.get('nombre_departamento_detalle_ticket');
                        const fecha_inio =
record.get('fecha_apertura_ticket');

                        return `
                            <span><b>Asunto:</b> ${asunto}</span><br>
                            <span><b>Requerimiento:</b>
${requerimiento}</span><br>
                            <span><b>Tecnico a cargo:</b>
${tecnico_a_cargo}</span><br>
                            <span><b>Departamento:</b>
${departamento}</span><br>
```

```
                            <div style="white-
space:normal;"><b>Descripcion:</b> ${descripcion}</div>
                            <span><b>Inicio:</b>
${fecha_inio}</span><br>
                        `;
                    },
                },
                { text: 'Id Estado', dataIndex: 'id_fkestado_ticket',
hidden: true, hideable: false },
                {
                    text: 'Estado',
                    dataIndex: 'nombre_estado_ticket',
                    flex: 1,
                    width: 'auto',
                    renderer: function (value, metaData, record) {
                        var valor =
record.get('nombre_estado_ticket');
                        if (valor == 'CERRADO') {
                            return '<span style="color: red; font-
weight: bold;">' + valor + '</span>'
                        } else {
                            return '<span style="color: green; font-
weight: bold;">' + valor + '</span>'
                        }
                    }
                },
                { text: 'Cerrado el', dataIndex:
'fecha_cierre_ticket', formatter: 'date("m/d/Y h:m:s")', flex: 1,
width: 'auto' },
                {
                    text: 'Prioridad',
                    dataIndex: 'nombre_prioridad_detalle_ticket',
                    flex: 1,
                    width: 'auto',
                    hidden: Ext.util.Cookies.get('rol_usuario') == 3
? true : false,
                    hideable: false,
                    renderer: function (value, metaData, record) {
                        var valor =
record.get('nombre_prioridad_detalle_ticket');
                        switch (valor) {
                            case 'ALTA':
                                return '<span style="color: red;
font-weight: bold;">' + valor + '</span>';
                            case 'MEDIA':
                                return '<span style="color: orange;
font-weight: bold;">' + valor + '</span>';
                            case 'BAJA':
```

```
                                return '<span style="color: green;
font-weight: bold;">' + valor + '</span>';
                            }
                        }
                    },
                    {
                        xtype: 'actioncolumn',
                        hideable: false,
                        width: 'auto',
                        items: [
                            {
                                iconCls: 'fas fa-eye black',
                                tooltip: 'Ver detalle',
                                display: 'fit',
                                handler: 'onClickVerDetalle',
                            },

                        ]
                    },
                ],
                listeners: {
                    afterrender: 'calcularTamañoGrid'
                },
                bbar: Ext.create('Ext.PagingToolbar', {
                    store: 'Ticket',
                    displayInfo: true,
                    displayMsg: 'Mostrando datos {0} - {1} de {2}',
                    emptyMsg: "No hay datos para mostrar",
                    inputItemWidth: 35,
                }),

            }
```

```
const seleccionarTickets = async (req, res, identificacion_usuario,
nombre_usuario, identificacion_tecnico, nombre_tecnico,
    estado, sistema, fecha_creacion_inicio, fecha_creacion_final,
fecha_apertura_inicio, fecha_apertura_final, fecha_cierre_inicio,
fecha_cierre_final,
    numero_bloque, tamanio_bloque) => {
    try {
        var offset = (numero_bloque - 1) * tamanio_bloque;

        var sqltickets = `SELECT a.id_ticket,
                    a.id_fkusuario_ticket,
                    a.identificacion_usuario_ticket,
                    a.nombre_usuario_ticket,
                    a.id_fktecnico_ticket,
```

```
                    a.identificacion_tecnico_ticket,
                    a.nombre_tecnico_ticket,
                    a.id_fkestado_ticket,
                    a.nombre_estado_ticket,
                    a.asunto_ticket,
                    a.fecha_creacion_ticket,
                    a.fecha_apertura_ticket,
                    a.fecha_cierre_ticket,
                    b.id_fksistema_detalle_ticket,
                    b.nombre_sistema_detalle_ticket,
                    a.es_requerimiento_ticket,
                    b.descripcion_detalle_ticket,
                    b.nombre_departamento_detalle_ticket,
                    b.nombre_prioridad_detalle_ticket,
                    b.nombre_plan_detalle_ticket
                FROM ticket as a, detalle_ticket as b
                where 1 = 1
                and a.id_ticket =
b.id_fkmaestro_ticket_detalle_ticket
                and a.identificacion_usuario_ticket like ?
                and a.nombre_usuario_ticket like ?
                and a.identificacion_tecnico_ticket like ?
                and a.nombre_tecnico_ticket like ?`;

        var rol = req.cookies.rol_usuario || 3;
        if (rol == 3) {
            sqltickets += ` and a.id_fkusuario_ticket = ` +
req.cookies.id_usuario;
        }

        var valores = ['%' + identificacion_usuario + '%', '%' +
nombre_usuario + '%', '%' + identificacion_tecnico + '%', '%' +
nombre_tecnico + '%'];

        if (estado != 0) {
            sqltickets += ` and a.id_fkestado_ticket = ?`;
            valores.push(estado)
        }

        if (sistema != 0) {
            sqltickets += ` and b.id_fksistema_detalle_ticket = ?`;
            valores.push(sistema)
        }

        if (fecha_creacion_inicio != null && fecha_creacion_final !=
null) {
            sqltickets += ` and a.fecha_creacion_ticket between ? and
?`;
```

```
            valores.push(fecha_creacion_inicio, fecha_creacion_final)
        }

        if (fecha_apertura_inicio != null && fecha_apertura_final !=
null) {
            sqltickets += ` and a.fecha_apertura_ticket between ? and
?`;
            valores.push(fecha_apertura_inicio, fecha_apertura_final)
        }

        if (fecha_cierre_inicio != null && fecha_cierre_final !=
null) {
            sqltickets += ` and a.fecha_cierre_ticket between ? and
?`;
            valores.push(fecha_cierre_inicio, fecha_cierre_final)
        }

        var sqlconteotickets = `select count(*) as total from (` +
sqltickets + `) as subconsulta;`;

        sqltickets += `  order by a.fecha_apertura_ticket desc limit
${tamanio_bloque} offset ${offset};`;

        var [results, fields] = await bd.execute(sqltickets,
valores);
        var [results2, fields2] = await bd.execute(sqlconteotickets,
valores);

        let tickets = [];
        for (let index = 0; index < results.length; index++) {
            tickets.push(construirTicket(results[index]))
        }

        return res.json({ data: tickets, success: true, total:
results2[0].total });
    } catch (error) {
        return res.json({ data: error, success: true, tipo: 2 });
    }
}

const seleccionarUnTicket = async (req, res, id_ticket) => {
    try {
        var sqlTicket = `SELECT b.id_ticket,
            b.id_fkestado_ticket,
            b.id_fktecnico_ticket,
            b.asunto_ticket,
            b.es_requerimiento_ticket,
            b.fecha_cierre_ticket,
```

```
            a.descripcion_detalle_ticket,
            a.id_fkdepartamento_detalle_ticket,
            a.id_fkprioridad_detalle_ticket,
            a.id_fksistema_detalle_ticket,
            a.id_fkplan_detalle_ticket,
            a.id_fkmodulo_detalle_ticket,
            a.nombre_modulo_detalle_ticket,
            a.referencia_detalle_ticket,
            a.pasos_realizados_ticket,
            a.correo_tecnico_detalle_ticket,
            a.correo_cliente_detalle_ticket,
            c.contenido_archivo_csv
        FROM detalle_ticket as a, ticket as b, chat as c
        where a.id_fkmaestro_ticket_detalle_ticket = b.id_ticket
        and b.id_ticket = c.id_fkticket_chat_ticket
        and a.id_fkmaestro_ticket_detalle_ticket = ?;`;

        var [results, fields] = await bd.execute(sqlTicket,
[id_ticket]);

        let tickets = construirDetalleTicket(results[0]);

        let chats = Papa.parse(results[0].contenido_archivo_csv, {
            header: true,
        }).data;

        return res.json({ data: {tickets, chats}, success: true,
total: tickets.length });
    } catch (error) {
        console.log(error)
        return res.json({ data: [], success: true, total: 0 });
    }
}
```

Historia de usuario 19 – Mensajería instantánea para chats

Tabla 87. Tarea 19 – Mensajería instantánea para chats

TAREA	
Historia de usuario: #19	
Usuario: Todos	
Nombre: Mensajería instantánea para chats	
Prioridad en el negocio: Alta	**Riesgo de desarrollo:** Alta
Puntos estimados: 2	**Iteración asignada: 5**
Fecha Inicio: 24/11/2024	**Fecha Fin: 25/11/2024**

Programador a cargo: Patricio Moscoso
Descripción: Cada ticket tendrá su propio chat en donde podrán comunicarse cliente y técnico.

En el evento launch de la aplicación, se crea un canal con el servidor el cual permite la comunicación en tiempo real al momento de emitirse un evento. En el Backend se define el evento para recargar los datos de las aplicaciones en tiempo real, el evento para recibir y enviar mensajes, el evento para actualizar la interfaz de todos los clientes en tiempo real y por último el evento que finaliza la conexión del usuario. Debido a que es una función que se ejecuta en segundo plano, no posee diseño de interfaz.

```
launch: async function () {
        var loggedIn;
        var mainView;
        loggedIn = localStorage.getItem('isAuthenticated');

        if (loggedIn) {
            var rol = Ext.util.Cookies.get('rol_usuario');
            socket_general = io('http://localhost:8080');

            switch (rol) {
                case "1":
                    mainView = 'app-main';
                    Ext.create({
                        xtype: mainView,
                        backgroundSize: 'cover',
                        backgroundPosition: 'center',
                        backgroundRepeat: 'no-repeat',
                        width: '100vw',
                        height: '100vh',
                        renderTo: Ext.getBody(),
                    });
                    break;

                case "2":
                    mainView = 'app-main';
                    Ext.create({
                        xtype: mainView,
                        backgroundSize: 'cover',
                        backgroundPosition: 'center',
                        backgroundRepeat: 'no-repeat',
                        width: '100vw',
                        height: '100vh',
                        renderTo: Ext.getBody(),
                    });
```

```
                    Ext.getCmp('acceso_rapido_crear_usuario').hide();
                    Ext.getCmp('acceso_rapido_administracion').hide()
;
                    Ext.getCmp('acceso_rapido_planes').hide();
                    Ext.getCmp('acceso_rapido_sistemas').hide();
                    break;

                case "3":
                    mainView = 'app-main';

                    Ext.create({
                        xtype: mainView,
                        backgroundSize: 'cover',
                        backgroundPosition: 'center',
                        backgroundRepeat: 'no-repeat',
                        width: '100vw',
                        height: '100vh',
                        renderTo: Ext.getBody(),
                    });

                    Ext.getCmp('acceso_rapido_crear_usuario').hide();
                    Ext.getCmp('acceso_rapido_administracion').hide()
;
                    Ext.getCmp('acceso_rapido_planes').hide();
                    Ext.getCmp('acceso_rapido_sistemas').hide();

                    break;
            }

        } else {

            Ext.create({
                xtype: 'container',
                id: 'login_background',
                renderTo: Ext.getBody(),
                style: {
                    backgroundImage:
'url(resources/backgroundLogin.jpg)',
                    backgroundSize: 'cover',
                    backgroundPosition: 'center',
                    backgroundRepeat: 'no-repeat',
                    width: '100vw',
                    height: '100vh',
                },
                items: [
                    {xtype: Ext.create('helpdesk.view.login.Login')}
                ]
            });
```

```
        }


        if (socket_general) {
            //actualizar grids de todos los clientes (en tiempo real)
            socket_general.on('actualizar_grids',
(id_grid_actualizar) => {
                var grid = Ext.getCmp(id_grid_actualizar);
                if (grid) {
                    var store = grid.getStore();
                    store.reload();
                }
            })

            //actualizar tickets de todos los clientes (en tiempo
real)
            socket_general.on('actualizar_tickets', (id_ticket,
actualizar, es_mensaje) => {
                var panel_construido =
Ext.ComponentQuery.query('container[itemId=ticket_' + id_ticket +
']')[0];
                if (panel_construido) {
                    switch (es_mensaje) {
                        case 0:
                            //actualizar info del ticket
                            panel_construido.estado =
actualizar.estado;

                            var field_estado =
Ext.ComponentQuery.query('displayfield[itemId=field_detalle_estado_'
+ id_ticket + ']')[0];
                            field_estado.setValue(Ext.getStore("Estad
osActivos").findRecord('id_estado_ticket',
actualizar.estado).get('nombre_estado_ticket'))

                            var field_fecha =
Ext.ComponentQuery.query('displayfield[itemId=field_detalle_fechaCier
re_' + id_ticket + ']')[0];
                            field_fecha.setValue(actualizar.fecha)

                            if (Ext.util.Cookies.get('rol_usuario')
== 3) {
                                //actualizar campos de cliente
                                var field_tecnico =
Ext.ComponentQuery.query('displayfield[itemId=field_detalle_tecnico_'
+ id_ticket + ']')[0];
```

```
                                field_tecnico.setValue(Ext.getStore("
Usuarios").findRecord('id_usuario',
actualizar.tecnico).get('nombre_usuario'))

                                var displayfield_descripcion =
Ext.ComponentQuery.query('displayfield[itemId=displayfield_detalle_de
scripcion_' + id_ticket + ']')[0];
                                displayfield_descripcion.setValue(act
ualizar.descripcion)
                            } else {
                                //actualizar campos de tecnico/admin
                                var field_prioridad =
Ext.ComponentQuery.query('displayfield[itemId=field_detalle_prioridad
_' + id_ticket + ']')[0];
                                field_prioridad.setValue(actualizar.p
rioridad)

                                var combo_tecnico =
Ext.ComponentQuery.query('combo[itemId=combo_detalle_tecnico_' +
id_ticket + ']')[0];
                                combo_tecnico.setValue(actualizar.tec
nico)

                                var textarea_descripcion =
Ext.ComponentQuery.query('textarea[itemId=textarea_detalle_descripcio
n_' + id_ticket + ']')[0];
                                textarea_descripcion.setValue(actuali
zar.descripcion)

                                var textarea_pasos =
Ext.ComponentQuery.query('textarea[itemId=textarea_detalle_pasos_' +
id_ticket + ']')[0];
                                textarea_pasos.setValue(actualizar.pa
sos)
                            }
                            break;
                        case 1:
                            //actualizar solo el estado del ticket
                            panel_construido.estado =
actualizar.estado;

                            var field_estado =
Ext.ComponentQuery.query('displayfield[itemId=field_detalle_estado_'
+ id_ticket + ']')[0];
                            field_estado.setValue(Ext.getStore("Estad
osActivos").findRecord('id_estado_ticket',
actualizar.estado).get('nombre_estado_ticket'))
                            break;
```

```
                        case 2:
                            panel_construido.estado =
actualizar.estado;

                            var field_estado =
Ext.ComponentQuery.query('displayfield[itemId=field_detalle_estado_'
+ id_ticket + ']')[0];
                            field_estado.setValue(Ext.getStore("Estad
osActivos").findRecord('id_estado_ticket',
actualizar.estado).get('nombre_estado_ticket'))

                            //deshabilitar / habilitar botones
generales
                            if (actualizar.estado == 6) {
                                var btn_enviar_mensaje =
Ext.ComponentQuery.query('button[itemId=enviarMensaje' + id_ticket +
']')[0];
                                btn_enviar_mensaje.disable();

                                var btn_reabrir =
Ext.ComponentQuery.query('button[itemId=btn_detalle_reabrir_' +
id_ticket + ']')[0];
                                btn_reabrir.show();

                                var textarea_mensaje =
Ext.ComponentQuery.query('textarea[itemId=textarea_detalle_mensaje_'
+ id_ticket + ']')[0];
                                textarea_mensaje.disable();

                                var archivo_ticket =
Ext.ComponentQuery.query('filefield[itemId=filefield_detalle_archivo_
' + id_ticket + ']')[0];
                                archivo_ticket.disable();
                            } else {
                                var btn_enviar_mensaje =
Ext.ComponentQuery.query('button[itemId=enviarMensaje' + id_ticket +
']')[0];
                                btn_enviar_mensaje.enable();

                                var btn_reabrir =
Ext.ComponentQuery.query('button[itemId=btn_detalle_reabrir_' +
id_ticket + ']')[0];
                                btn_reabrir.hide();

                                var textarea_mensaje =
Ext.ComponentQuery.query('textarea[itemId=textarea_detalle_mensaje_'
+ id_ticket + ']')[0];
                                textarea_mensaje.enable();
```

```
                                var archivo_ticket =
Ext.ComponentQuery.query('filefield[itemId=filefield_detalle_archivo_
' + id_ticket + ']')[0];
                                archivo_ticket.enable();
                            }

                            //deshabilitar / habilitar botones para
tecnicos
                            if (Ext.util.Cookies.get('rol_usuario')
!= 3) {
                                if (actualizar.estado == 6) {
                                    var field_prioridad =
Ext.ComponentQuery.query('combo[itemId=combo_detalle_prioridad_' +
id_ticket + ']')[0];
                                    field_prioridad.disable();

                                    var combo_tecnico =
Ext.ComponentQuery.query('combo[itemId=combo_detalle_tecnico_' +
id_ticket + ']')[0];
                                    combo_tecnico.disable();

                                    var textarea_descripcion =
Ext.ComponentQuery.query('textarea[itemId=textarea_detalle_descripcio
n_' + id_ticket + ']')[0];
                                    textarea_descripcion.disable();

                                    var textarea_pasos =
Ext.ComponentQuery.query('textarea[itemId=textarea_detalle_pasos_' +
id_ticket + ']')[0];
                                    textarea_pasos.disable();
                                } else {
                                    var field_prioridad =
Ext.ComponentQuery.query('combo[itemId=combo_detalle_prioridad_' +
id_ticket + ']')[0];
                                    field_prioridad.enable();

                                    var combo_tecnico =
Ext.ComponentQuery.query('combo[itemId=combo_detalle_tecnico_' +
id_ticket + ']')[0];
                                    combo_tecnico.enable();

                                    var textarea_descripcion =
Ext.ComponentQuery.query('textarea[itemId=textarea_detalle_descripcio
n_' + id_ticket + ']')[0];
                                    textarea_descripcion.enable();
```

```
                                    var textarea_pasos =
Ext.ComponentQuery.query('textarea[itemId=textarea_detalle_pasos_' +
id_ticket + ']')[0];
                                    textarea_pasos.enable();
                                }

                            }

                            break;
                        }

                    }
                });
            }

    },
```

```
// sockets para manejo de mensajes en tiempo real
io.on('connection', (socket) => {

    //al enviar mensaje
    socket.on('chat message', async (mensaje, arrayBuffer) => {

        console.log('hola')
        if (mensaje.texto != '') {
            //actualiza el estado del ticket y la fecha de caducidad
(en caso de que sea estado "esperando respuesta cliente")
            var es_revision = mensaje.revision == true ? 1 : 0;
            try {
                var [results, fields] = await bd.execute('select
autocambiar_estado_ticket(?,?,?)', [mensaje.id_ticket, mensaje.rol,
es_revision]);

                insercion = results.affectedRows;
            } catch (error) {
                console.log(error)
                insercion = -2;
            }

            //guardar archivos recibidos del mensaje (en caso de
tener)
            if (arrayBuffer && mensaje.nombres_archivos.length > 0) {
                writeFile(__dirname + "/tmp/archivos/" +
mensaje.nombres_archivos[0], arrayBuffer, (err) => {
                    console.log(err)
                });
```

```
                mensaje.paths[0] = "/tmp/archivos/" +
mensaje.nombres_archivos[0]
            }

            //actualizar csv de mensajes
            var mensajeJson = JSON.stringify({
                data: [
                    mensaje.id,
                    mensaje.usuario,
                    mensaje.texto,
                    mensaje.contiene_archivo,
                    mensaje.fecha,
                    mensaje.paths,
                    mensaje.nombres_archivos,
                ]
            });

            let chats = Papa.unparse(mensajeJson);
            var insercion = 0;
            try {
                var [results, fields] = await bd.execute(`UPDATE
helpdesk.chat
                    SET
                    contenido_archivo_csv =
CONCAT(contenido_archivo_csv, '\n', ?)
                    WHERE id_fkticket_chat_ticket = ?;`, [chats,
mensaje.id_ticket]);

                insercion = results.affectedRows;
            } catch (error) {
                console.log(error)
                insercion = -1;
            }

            //notificar por correo
            var rol = mensaje.rol == 3 ? 'Cliente' : 'Tecnico';
            var correo = mensaje.rol == 3 ? mensaje.correo_tecnico :
mensaje.correo_cliente;
            emailHelper(
                correo,
                'Te han respondido a tu ticket #' +
mensaje.id_ticket,
                `<span><b>Ticket #${mensaje.id_ticket}</b></br>
                </br>
                Mensaje :${mensaje.texto}</br>
                </br>
                ¡Esperamos su pronta respuesta!</br>
                </br>
```

```
                Att: ${rol} ${mensaje.usuario}</br></span>`
            );

            //asignar estado en front
            var estado_obtenido = 0;
            if (es_revision == 1) {
                estado_obtenido = 4;
            } else {
                if (mensaje.rol != 3) {
                    estado_obtenido = 3;
                } else {
                    estado_obtenido = 5;
                }
            }

            var actualizar = {
                estado: estado_obtenido
            }

            io.sockets.emit('actualizar_tickets', mensaje.id_ticket,
actualizar, 1);
            io.to('room_ticket_' + mensaje.id_ticket).emit('chat
message', mensaje);
        } else {
            socket.join('room_ticket_' + mensaje.id_ticket);
        }
    })

    //salir del chat
    socket.on('leave_chat', (id_ticket) => {
        socket.leave('room_ticket_' + id_ticket);
        socket.disconnect();
        console.log('usuario salio de sala ' + id_ticket)
    })

    //actualizar grids
    socket.on('llamado_actualizar_grids', (id_grid_actualizar) => {
        socket.broadcast.emit('actualizar_grids',
id_grid_actualizar);
    })

    //actualizar interfaz tickets
    socket.on('llamado_actualizar_tickets', (id_ticket, actualizar,
es_mensaje) => {
        io.sockets.emit('actualizar_tickets', id_ticket, actualizar,
es_mensaje);
    })
```

```
    //desconexion
    socket.on('disconnect', () => {
        console.log('usuario desconectado')
    })
});
```

Tabla 88. Tarea 20 – Notificación de tickets

TAREA	
Historia de usuario: #20	
Usuario: --	
Nombre: Notificación de tickets	
Prioridad en el negocio: Alta	**Riesgo de desarrollo:** Alta
Puntos estimados: 5	**Iteración asignada: 5**
Fecha Inicio: 26/11/2024	**Fecha Fin: 01/12/2024**
Programador a cargo: Patricio Moscoso	
Descripción: El sistema notificará tanto a clientes como miembros de la empresa cada que se efectúe una respuesta.	

El código envía correos electrónicos mediante configuración Gmail otorgado por el autor, en el entorno de producción se configuró al correo empresarial junto con su respectiva configuración. Al ser una función que se ejecuta en segundo plano en el Backend, no posee interfaz.

```
const emailHelper = async (to, subject, text) => {
  // Crear trasporter (emisor)
  let transporter = nodemailer.createTransport({
    service: "gmail",
    auth: {
      user: "correo_autor",
      pass: "**** **** **** ****",
    },
  });

  // Configuracion
  let mailOptions = {
    from: "correo_autor",
    to: to,
    subject: subject,
    html: text,
  };

  // Enviar correo
  try {
    let info = await transporter.sendMail(mailOptions);
```

```
    console.log("Email sent: " + info.response);
    return info.response;
  } catch (error) {
    console.error("Error sending email:", error);
    throw error;
  }
};
```

Historia de usuario 21 – Tutoriales

Tabla 89. Tarea 21 – Tutoriales

TAREA	
Historia de usuario: #21	
Usuario: Todos	
Nombre: Tutoriales	
Prioridad en el negocio: Alta	**Riesgo de desarrollo:** Alta
Puntos estimados: 3	**Iteración asignada: 5**
Fecha Inicio: 02/12/2024	**Fecha Fin: 04/12/2024**
Programador a cargo: Patricio Moscoso	
Descripción: Visualización de tutoriales para capacitar a los usuarios y personal en el aplicativo.	

El código presenta la interfaz para verificar los tutoriales (manuales de usuario) para roles de técnico/administrador y usuario respectivamente. Los manuales están en formato PDF.

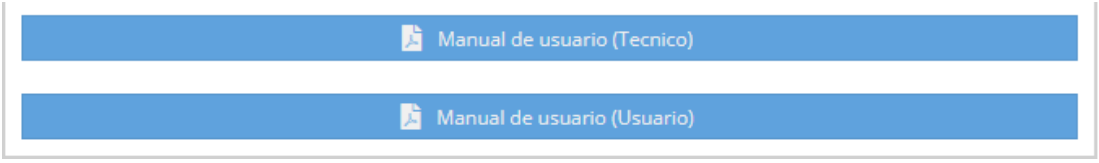


Figura 37. Manuales de usuario

```
{
                    xtype: 'button',
                    columnWidth: 1,
                    iconCls: 'fas fa-file-pdf',
                    text: 'Manual de usuario (Tecnico)',
                    cls: '.botones-reporte',
                    padding: 5,
                    margin: 10,
                    hidden: Ext.util.Cookies.get('rol_usuario') == 3
? true : false,
                    listeners: {
                        click: 'onClickManualTecnico'
                    }
```

```
                },
                {
                    xtype: 'button',
                    columnWidth: 1,
                    iconCls: 'fas fa-file-pdf',
                    text: 'Manual de usuario (Usuario)',
                    cls: '.botones-reporte',
                    padding: 5,
                    margin: 10,
                    listeners: {
                        click: 'onClickManualUsuario'
                    }
                },
```

```
router.get('/obtenerDocumento', async (req, res) => {
    try {
        var filepath = req.query.path;
        const archivo = path.join(__dirname, '..', filepath);

        res.sendFile(archivo, (err) => {
            if (err) {
                console.log(err.message)
                res.json({ data: 'Archivo no encontrado, envielo
nuevamente por favor.', success: false, tipo: 2 })
            }
        });
    } catch (error) {
        console.log(error)
        return res.json({ data: "Ha ocurrido un error, intentelo mas
tarde o contacte con soporte", success: false, tipo: 2 });
    }
});
```

Tabla 90. Tarea 22 – Reportes

TAREA	
Historia de usuario: #22	
Usuario: Todos	
Nombre: Reportes	
Prioridad en el negocio: Alta	**Riesgo de desarrollo:** Alta
Puntos estimados: 3	**Iteración asignada: 5**
Fecha Inicio: 05/12/2024	**Fecha Fin: 07/12/2024**
Programador a cargo: Patricio Moscoso	
Descripción: Todos podrán generar reportes filtrados por fechas.	

Presenta los reportes de tickets por usuario (tanto resueltos como no resueltos) en .PDF, resumen de tickets por usuario en .PDF. Los reportes de clientes con más reportes y tickets creados por mes son utilizados para gráficos estadísticos.

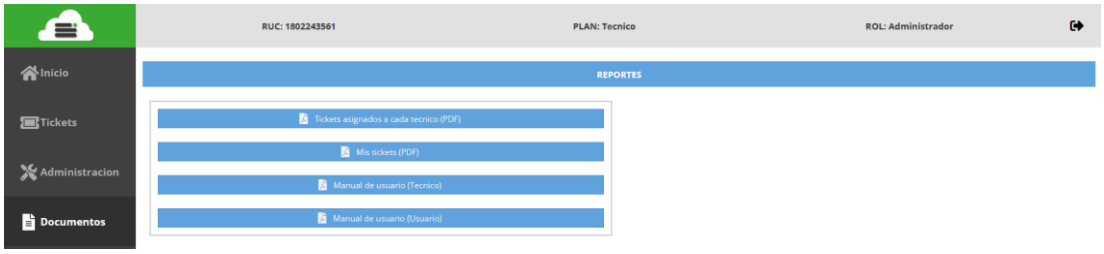


Figura 38. Reportes

```
        {
            xtype: 'label',
            text: 'REPORTES',
            columnWidth: 1,
            margin: '10 0 10 0',
            padding: 10,
            style: 'background-color: #5fa2dd; color: white; font-
weight: bold; font-size: 30; text-align: center;'
        },
        {
            xtype: 'panel',

            layout: 'column',
            columnWidth: 0.5,
            header: false,
            margin: 10,
            style: 'border: 2px #d0d0d0 solid;',
            items: [
                {
                    xtype: 'button',
                    columnWidth: 1,
                    iconCls: 'fas fa-file-pdf',
                    text: 'Tickets asignados a cada tecnico (PDF)',
                    padding: 5,
                    margin: 10,
                    cls: '.botones-reporte',
                    listeners: {
                        click: 'onClickTicketsPorTecnico'
                    },
                    hidden: Ext.util.Cookies.get('rol_usuario') == 3
? true : false,
                },
```

```
            {
                xtype: 'button',
                columnWidth: 1,
                iconCls: 'fas fa-file-pdf',
                text: 'Mis tickets (PDF)',
                padding: 5,
                cls: '.botones-reporte',
                margin: 10,
                listeners: {
                    click: 'onClickMisTickets'
                }
            },
```



REPORTES DE TICKETS
Enero a Diciembre - 2024

Identificacion: [redacted]
Nombre: [redacted]
Direccion: [redacted]
Correo: [redacted]
Telefono: [redacted]
Fecha de registro: 2024-11-16 17:38:54

Sistema: [redacted]
Plan Contratado: [redacted]
Fecha de renovacion 2024-11-21 00:00:00
Departamento: [redacted]

Generado por [redacted]
Generado el: 2024-12-29 05:43:23

# Ticket	Detalle	Usuario	Estado
5	**Asunto:** no guarda factura **Referencia:** CTF-CLI-0000000005 **Creado el:** 2024-12-08 14:11:07 **Módulo:** [redacted] **Descripción:** nada realmente	**Identificación:** [redacted] **Nombre:** [redacted] **Departamento asignado:** [redacted] **Cerrado el:**	ACTIVO
11	**Asunto:** Reporte de ventas erroneo **Referencia:** CTF-CTB-0000000011 **Creado el:** 2024-12-10 22:18:59 **Módulo:** [redacted] **Descripción:** Los valores no son los esperados	**Identificación:** [redacted] **Nombre:** [redacted] **Departamento asignado:** [redacted] **Cerrado el:** 2024-12-13 15:58:32	CERRADO

Figura 39. Estructura de documentos PDF

3.2.5 Fase 5: Pruebas

a. Pruebas de aceptación

Las pruebas de aceptación se llevaron a cabo cuando se finalizaba una iteración. Si al momento de realizar las pruebas se producía un fallo, se tomaba nota y se lo corregía para las siguientes iteraciones.

Tabla 91. Pruebas de aceptación para Iteración 1

Iteración	N°	Historia de usuario	Evaluación	Observaciones
1	1	Ingreso al sistema	Neutral	Control de datos vacíos en los campos de inicio sesión.
	2	Recuperar contraseña	Exitosa	Ninguna
	3	Administración de técnicos	Neutral	Cambiar color a los botones de interacción (editar, habilitar, eliminar)
	4	Administración de usuarios	Neutral	Cambiar color a los botones de interacción (editar, habilitar, eliminar)
	5	Administración de roles	Neutral	Cambiar color a los botones de interacción (editar, habilitar, eliminar)

Tabla 92. Pruebas de aceptación para Iteración 2

Iteración	N°	Historia de usuario	Evaluación	Observaciones
2	1	Ingreso al sistema	Exitosa	Ninguna
	3	Administración de técnicos	Exitosa	Ninguna
	4	Administración de usuarios	Exitosa	Ninguna
	5	Administración de roles	Exitosa	Ninguna
	6	Administración de departamentos	Exitosa	Ninguna
	7	Administración de planes	Exitosa	Ninguna
	8	Administración de sistemas	Exitosa	Ninguna
	9	Cerrar Sesión	Fallida	Las cookies no se eliminan correctamente, la interfaz de Login no carga adecuadamente

Tabla 93. Pruebas de aceptación para Iteración 3

Iteración	N°	Historia de usuario	Evaluación	Observaciones
3	9	Cerrar Sesión	Exitosa	Ninguna
	10	Administración de estados para tickets	Exitosa	Ninguna
	11	Administración de prioridades para tickets	Exitosa	Ninguna
	12	Crear ticket	Neutral	Añadir los pasos realizados por el técnico en el detalle
	13	Editar ticket	Exitosa	Ninguna

Tabla 94. Pruebas de aceptación para Iteración 4

Iteración	N°	Historia de usuario	Evaluación	Observaciones
4	12	Crear ticket	Exitosa	Ninguna
	14	Traspasar ticket	Exitosa	Ninguna
	15	Cerrar ticket	Neutral	Deshabilitar campos de edición y formulario mensajes al deshabilitar
	16	Reaperturar ticket	Neutral	Habilitar campos de edición y formulario mensajes al deshabilitar
	17	Cambio de estado del ticket	Exitosa	Ninguna

Tabla 95. Pruebas de aceptación para Iteración 5

Iteración	N°	Historia de usuario	Evaluación	Observaciones
5	15	Cerrar ticket	Exitosa	Ninguna
	16	Reaperturar ticket	Exitosa	Ninguna
	18	Visualización de tickets	Exitosa	Ninguna
	19	Mensajería instantánea para chats	Fallida	Los mensajes son enviados a todos los tickets abiertos en el sistema.
	20	Notificación de tickets	Neutral	Mejorar el esquema del correo electrónico
	21	Tutoriales	Exitosa	Ninguna
	22	Reportes	Fallida	Control de campos vacíos en Front antes de realizar el reporte

Tabla 96. Pruebas de aceptación para actividades fallidas de Iteración 5

Iteración	N°	Historia de usuario	Evaluación	Observaciones
5	19	Mensajería instantánea para chats	Exitosa	Ninguna
	20	Notificación de tickets	Exitosa	Ninguna
	22	Reportes	Exitosa	Ninguna

b. Pruebas de usabilidad

Para la realización de las pruebas de usabilidad, se utilizó el modelo TAM (Modelo de aceptación de tecnología) el cual permite determinar el comportamiento de un usuario al utilizar un sistema, dicho comportamiento es influenciado por la facilidad de uso percibida y la utilidad percibida. Por lo cual se establecieron como métricas de medición la utilidad percibida, la facilidad de uso y la intención de uso [29].

Las pruebas de usabilidad fueron realizadas una vez se concretó el sistema. Se aplicaron tests de usabilidad a todos los integrantes de la empresa debido a que no

exceden las cien personas, permitiendo medir el nivel de satisfacción al utilizar el sistema en la totalidad de la empresa.

Tabla 97. Prueba de usabilidad 1

Prueba de usabilidad					
Número: #1					
Valoración - 1 = Malo - 2 = Regular - 3 = Bueno - 4 = Muy Bueno - 5 = Excelente					
Usuario: Gerente de la empresa					
Preguntas	**1**	**2**	**3**	**4**	**5**
¿El sistema navega entre módulos de forma fluida?				**X**	
¿La combinación de colores seleccionados son los adecuados?					**X**
¿El sistema facilita la gestión de solicitudes de soporte?				**X**	
¿La aplicación es intuitiva y fácil de usar?				**X**	
¿Los formularios están correctamente distribuidos?				**X**	

Tabla 98. Prueba de usabilidad 2

Prueba de usabilidad					
Número: #1					
Valoración - 1 = Malo - 2 = Regular - 3 = Bueno - 4 = Muy Bueno - 5 = Excelente					
Usuario: Agente técnico 1					
Preguntas	**1**	**2**	**3**	**4**	**5**
¿El sistema navega entre módulos de forma fluida?				**X**	
¿La combinación de colores seleccionados son los adecuados?				**X**	
¿El sistema facilita la gestión de solicitudes de soporte?				**X**	
¿La aplicación es intuitiva y fácil de usar?				**X**	
¿Los formularios están correctamente distribuidos?				**X**	

Tabla 99. Prueba de usabilidad 3

Prueba de usabilidad					
Número: #1					
Valoración - 1 = Malo - 2 = Regular - 3 = Bueno - 4 = Muy Bueno - 5 = Excelente					
Usuario: Agente técnico 2					
Preguntas	**1**	**2**	**3**	**4**	**5**
¿El sistema navega entre módulos de forma fluida?				**X**	

¿La combinación de colores seleccionados son los adecuados?					**X**
¿El sistema facilita la gestión de solicitudes de soporte?			**X**		
¿La aplicación es intuitiva y fácil de usar?				**X**	
¿Los formularios están correctamente distribuidos?				**X**	

Tabla 100. Prueba de usabilidad 4

Prueba de usabilidad					
Número: #1					
Valoración - 1 = Malo - 2 = Regular - 3 = Bueno - 4 = Muy Bueno - 5 = Excelente					
Usuario: Project Manager					
Preguntas	**1**	**2**	**3**	**4**	**5**
¿El sistema navega entre módulos de forma fluida?				**X**	
¿La combinación de colores seleccionados son los adecuados?				**X**	
¿El sistema facilita la gestión de solicitudes de soporte?			**X**		
¿La aplicación es intuitiva y fácil de usar?					**X**
¿Los formularios están correctamente distribuidos?					**X**

Tabla 101. Prueba de usabilidad 5

Prueba de usabilidad					
Número: #1					
Valoración - 1 = Malo - 2 = Regular - 3 = Bueno - 4 = Muy Bueno - 5 = Excelente					
Usuario: Secretaria					
Preguntas	**1**	**2**	**3**	**4**	**5**
¿El sistema navega entre módulos de forma fluida?				**X**	
¿La combinación de colores seleccionados son los adecuados?					**X**
¿El sistema facilita la gestión de solicitudes de soporte?				**X**	
¿La aplicación es intuitiva y fácil de usar?				**X**	
¿Los formularios están correctamente distribuidos?					**X**

3.2.6 Implantación

a. Instalación del sistema web

El sistema fue implantado en el servidor Cloud que posee la empresa, donde se creó un subdominio en el cPanel propio del servidor y se procedió a subir el aplicativo

mediante la clonación de un repositorio creado en GitHub. El repositorio es privado y solo es accesible por autorización del autor y tutor empresarial.

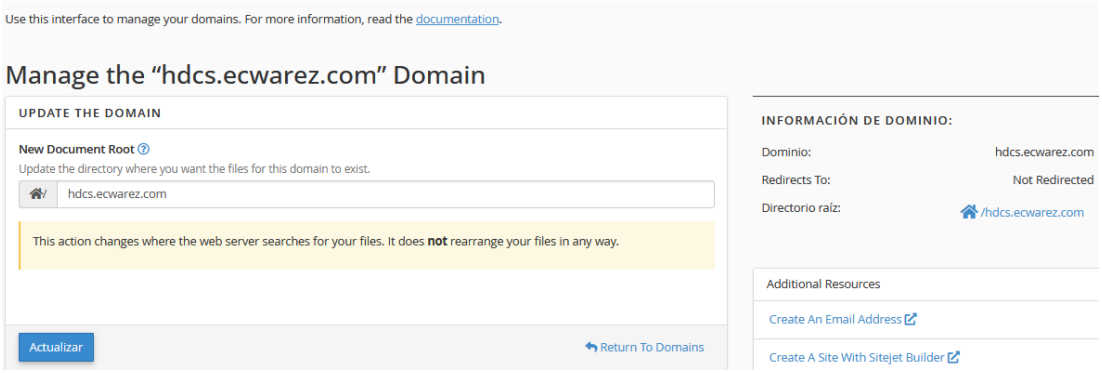


Figura 40. Subdominio creado para el aplicativo

Al clonarse los archivos, se configuró el framework Node JS dentro del servidor y se especificó la ruta del archivo index.js para ejecutar el aplicativo.

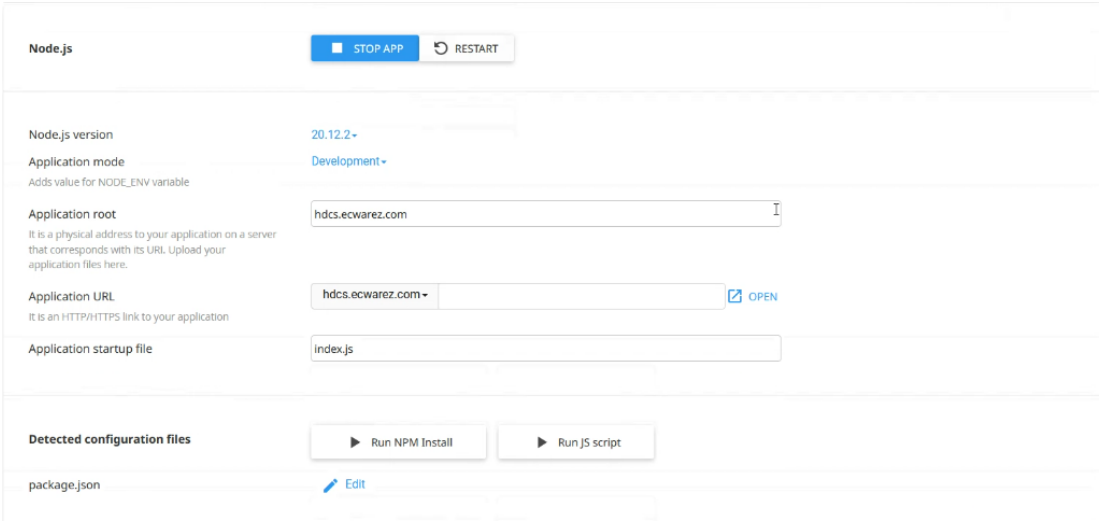


Figura 41. Creación del servicio Node

Después se creó la base de datos para el aplicativo web en el cpanel del servidor junto con el usuario respectivo y todas las tablas y funciones necesarias para su posterior configuración en el archivo de conexión de node.



Figura 42. Creación de la base de datos

Tras la configuración de la base de datos en el aplicativo, se puso en producción el dominio junto con la aplicación web.

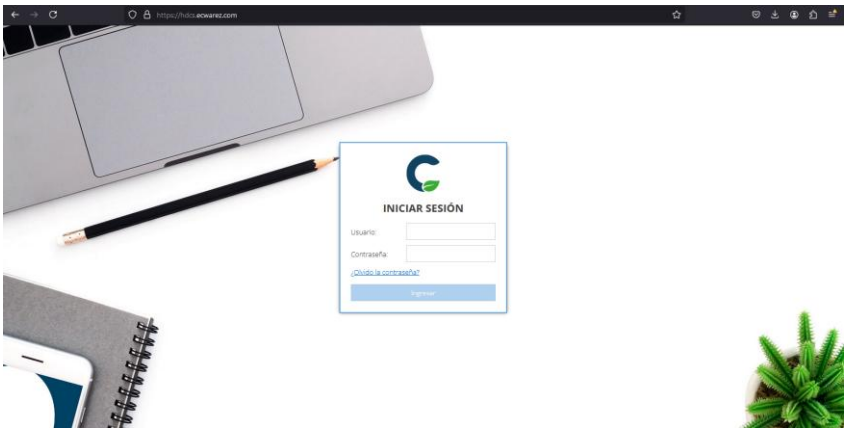


Figura 43. Sistema en producción

b. Capacitaciones

Tras la realización de las pruebas en cada una de las iteraciones de desarrollo de la aplicación y la implantación en el servidor de la empresa, se procedió a realizar las capacitaciones al personal técnico junto con la entrega del manual de usuario para la revisión de instrucciones y funcionalidades. El procedimiento se llevó a cabo en base al cronograma mostrado en la Tabla 50.

Tabla 102. Cronograma de capacitaciones

Tema	Encargado	Dirigido a	Fecha de inicio	Fecha de finalización	Realizado
Acceso al sistema	Patricio Moscoso	Técnicos, Gerente y PM	16/12/2024	16/12/2024	SI
Recuperar contraseña	Patricio Moscoso	Técnicos, Gerente y PM	16/12/2024	16/12/2024	SI
Administración de técnicos	Patricio Moscoso	Técnicos, Gerente y PM	16/12/2024	16/12/2024	SI
Administración de usuarios	Patricio Moscoso	Técnicos, Gerente y PM	16/12/2024	16/12/2024	SI
Cerrar Sesión	Patricio Moscoso	Técnicos, Gerente y PM	16/12/2024	16/12/2024	SI
Crear ticket	Patricio Moscoso	Técnicos, Gerente y PM	16/12/2024	16/12/2024	SI
Editar ticket	Patricio Moscoso	Técnicos, Gerente y PM	16/12/2024	16/12/2024	SI
Traspasar ticket	Patricio Moscoso	Técnicos, Gerente y PM	16/12/2024	16/12/2024	SI

Cerrar ticket	Patricio Moscoso	Técnicos, Gerente y PM	16/12/2024	16/12/2024	SI
Reaperturar ticket	Patricio Moscoso	Técnicos, Gerente y PM	16/12/2024	16/12/2024	SI
Cambio de estado del ticket	Patricio Moscoso	Técnicos, Gerente y PM	16/12/2024	16/12/2024	SI
Visualización de tickets	Patricio Moscoso	Técnicos, Gerente y PM	16/12/2024	16/12/2024	SI
Notificación de tickets	Patricio Moscoso	Técnicos, Gerente y PM	16/12/2024	16/12/2024	SI
Tutoriales	Patricio Moscoso	Técnicos, Gerente y PM	16/12/2024	16/12/2024	SI
Reportes	Patricio Moscoso	Técnicos, Gerente y PM	16/12/2024	16/12/2024	SI

CAPÍTULO IV. CONCLUSIONES Y RECOMENDACIONES

4.1 Conclusiones

- La recolección de datos con los instrumentos utilizados permitió la implementación óptima del proceso de atención técnica en el sistema web HelpDesk logrando identificar los cuellos de botella y optimizando el procedimiento base realizado por los agentes técnicos.

- Por medio del análisis de las distintas versiones de las herramientas openSource se logró la integración rápida y sostenible de las mismas dado al manejo de estructuras JSON para manipulación de los datos, el manejo de módulos lenguaje JavaScript y el continuo soporte que aportan sus respectivas comunidades, facilitando la solución de errores.

- La aplicación de la metodología XP optimizó el tiempo de desarrollo, debido a que ayudó a definir los entregables y organizó los cambios que se presentaron durante el progreso de las iteraciones, mejorando la lógica del sistema junto con el diseño de forma simultánea.

- El sistema web HelpDesk posee la capacidad de organizar las incidencias mediate el historial de mensajes que posee cada ticket registrado, visualización de los datos necesarios para los asistentes y los pasos realizados por técnico disponibles en tiempo real, el sistema puede adaptarse a cualquier otra empresa pequeña o mediana y ajustarse a las necesidades que requieran.

4.2 Recomendaciones

- Examinar los registros de las incidencias mediante el punto único de contacto (SPOC) en el departamento de soporte para definir bases para planificar el manejo y optimización de datos en el sistema a desarrollar.

- Analizar los patrones identificados en incidencias registradas en los chats, ya que estos permitirán plasmar acciones preventivas en la documentación del sistema a fin de ahorrar recursos a la empresa.

- Coordinar reuniones simultaneas con el stakeholder antes de definir una metodología de desarrollo debido a que afecta a la planificación, el desarrollo y la optimización del sistema.

- Realizar constantes actualizaciones al sistema en base a los requerimientos planteados por los usuarios y técnicos, permitiendo longevidad y robusticidad al manejo de datos.

REFERENCIAS BIBLIOGRÁFICAS

[1] E. AGUILAR LOZANO, “PLATAFORMA MULTICANAL EN LINEA PARA LA GESTION Y SEGUIMIENTO DE TICKETS PARA EMPRESAS EN TELECOMUNICACIONES”, TESIS DE GRADO, UNIVERSIDAD MAYOR DE SAN ANDRÉS, LA PAZ , 2021.

[2] L. A.-A. Abigail Gutierrez Ramirez, “Implementation of a Multiplatform System to Improve the Management of the Help Desk Process in the Constitutional Tribunal of Peru”, *International Journal of Engineering Trends and Technology*, nov. 2023.

[3] K. B. V. G. M. ALEKSANDRA REVINA, “IT Ticket Classification: The Simpler, the Better”, *IEE Access*, vol. 8, oct. 2020.

[4] Carbonel Guzmán Sergio Daniel, “MODELO DE MESA DE AYUDA BAJO LA METODOLOGÍA ITIL v3: CASO DE ESTUDIO EMPRESA PYME COMPUTROCK EIRL”, bachelor Thesis, Universidad Señor de Sipán, Pimentel, 2020.

[5] J. G. Lozada Acuña y C. J. Reyes Chalacan, “Desarrollo de un aplicativo web de mesa de ayuda aplicando el marco de referencia ITIL v4 para la gestión de tickets en el Departamento de Secretaría de la Universidad Técnica de Cotopaxi Extensión La Maná.”, bachelor Thesis, UNIVERSIDAD TÉCNICA DE COTOPAXI EXTENSIÓN LA MANÁ, LA MANÁ, 2022.

[6] J. K. Díaz Hidalgo, “Diseño de software de help desk en la nube basado en ITIL para la Cruz Roja Ecuatoriana”, bachelor Thesis, UNIVERSIDAD POLITÉCNICA SALESIANA SEDE QUITO, Quito, 2024.

[7] C. F. Aldás Flores y C. F. Oñate Rivera, “Desarrollo de un sistema Help Desk usando programación por capas y tecnología Asp.net Core para la Cooperativa de Ahorro y Crédito 9 de octubre Ltda.”, bachelor Thesis, Universidad Técnica de Ambato, Ambato, 2021.

[8] N. P. Reyes Velez y I. A. Supo Gurumendi, “PROTOTIPO DE UN SISTEMA PARA GESTIÓN DE INCIDENCIAS DE SERVICIO AL CLIENTE PARA LA EMPRESA DE MANTENIMIENTO Y SERVICIOS DE EQUIPOS INFORMÁTICOS REMYS S.A.”, Bachelor Thesis, Universidad de Guayaquil, Guayaquil, 2022.

[9] F. X. Ibañez Mendoza, “Implementación de servicio help desk o mesa de ayuda para la empresa ‘Machalagps’.”, bachelor Thesis, Universidad Católica de Santiago de Guayaquil, Guayaquil, 2022.

[10] C. Yashraj, S. Komal, K. Suraj, y S. Prof. Ranjana, “Maximizing Efficiency and Customer Satisfaction with Modern Helpdesk Ticketing System”, *IJRASET*, vol. 11, núm. iv, abr. 2023.

[11] K. Didi, O. Erna, D. Eko, N. Irfan, y D. Muhammad, “Web-Based IT Helpdesk Ticketing System at PT. Dayacipta Kemasindo”, *Bit-Tech*, vol. 5, dic. 2022.

[12] G. Livia, “What is Information Technology (IT)”, https://www.snhu.edu/about-us/newsroom/stem/what-is-information-technology.

[13] DAVID VÁSQUEZ COTRINA, “IMPACTO DE LA IMPLEMENTACIÓN DE UN SISTEMA HELP DESK EN LA GESTIÓN DE INCIDENTES INFORMÁTICOS EN LA UNIDAD DE INFORMÁTICA Y SISTEMAS DE LA MUNICIPALIDAD PROVINCIAL DE CAJAMARCA,» UNIVERSIDAD NACIONAL DE CAJAMARCA”, thesis, UNIVERSIDAD NACIONAL DE CAJAMARCA, Cajamarca, 2022.

[14] E. G. Maida y J. Pacienzia, “Metodologías de desarrollo de software”, Tesis de grado, UNIVERSIDAD CATÓLICA ARGENTINA, 2015.

[15] Ingeniería y Tecnología, “Framework: qué es, para qué sirve y algunos ejemplos”, UNIR. Consultado: el 23 de octubre de 2024. [En línea]. Disponible en: https://unirfp.unir.net/revista/ingenieria-y-tecnologia/framework/

[16] Jessica Morales-Carrillo, Luis Cedeño-Valarezo, Jesús Stefano Cajape Bravo, y Jonathan Geovanny Ormaza Calderón, “Metodologías de desarrollo de software y su ámbito de aplicación: Una revisión sistemática”, *risti*, pp. 29–45, jul. 2021, Consultado: el 23 de octubre de 2024. [En línea]. Disponible en: https://www.proquest.com/openview/937c0ffcf92bf2c2761d76165f34090e/1?pq-origsite=gscholar&cbl=1006393

[17] “¿Qué es una base de datos?”, AWS. Consultado: el 23 de octubre de 2024. [En línea]. Disponible en: https://aws.amazon.com/es/what-is/database/

[18] “¿Qué son los gestores de bases de datos y qué tipos existen?”, UNIR. Consultado: el 23 de octubre de 2024. [En línea]. Disponible en: https://ecuador.unir.net/actualidad-unir/gestores-bases-datos/

[19] B. O. Pesantes Jara, “Mesa de Ayuda para docentes y personal administrativo de la Facultad de Ingeniería de la UCSG”, Examen complexivo de grado, Universidad Católica de Santiago de Guayaquil, Guayaquil, 2019.

[20] F. X. Hugo Cardenas, C. E. Jimenez Rosero, M. Holovatyi, y P. E. Lara Pazos, “El impacto de las redes sociales en la administración de las empresas”, *RECIMUNDO*, vol. 4, núm. 1, pp. 173–182, dic. 2019, doi: 10.26820/recimundo/4.(1).enero.2020.173-182.

[21] G. C. Carmen Cristina, “MODELO DE GESTIÓN BASADO EN LA METODOLOGÍA ITIL V3 PARA EL SERVICIO DE HELP DESK ORIENTADA AL SOPORTE TÉCNICO DE LA UNIDAD DE TI DE LA EMPRESA ELÉCTRICA REGIONAL NORTE S.A. EMELNORTE”, bachelor Thesis, UNIVERSIDAD TÉCNICA DEL NORTE, Ibarra, 2021.

[22] A. E. Osejos Vasquez y J. L. Merino Murillo, “MODELO SERVQUAL COMO INSTRUMENTO DE EVALUACIÓN DE LA CALIDAD DEL SERVICIO AL CLIENTE, CANTÓN JIPIJAPA ”, *UNESUM*, vol. 4, núm. 4, pp. 79–92, nov. 2020.

[23] C. Ballesteros García y A. González, *APOYO NO FINANCIERO: LA ASISTENCIA TÉCNICA EN LA INVERSIÓN DE IMPACTO*, vol. 2. USA: Creative Commons, 2022.

[24] “Ext JS”, Sencha. Consultado: el 23 de octubre de 2024. [En línea]. Disponible en: https://www.sencha.com/products/extjs/

[25] “Node JS”, Node JS. Consultado: el 23 de octubre de 2024. [En línea]. Disponible en: https://nodejs.org/en

[26] “¿Qué es MySQL?”, ORACLE. Consultado: el 23 de octubre de 2024. [En línea]. Disponible en: https://www.oracle.com/co/mysql/what-is-mysql/

[27] Scott Ambler, “The Object Primer 3.ª edición: Desarrollo impulsado por modelos ágiles con UML 2”, https://scottambler-com.translate.goog/the-object-primer-3-e/?_x_tr_sl=en&_x_tr_tl=es&_x_tr_hl=es&_x_tr_pto=tc. Consultado: el 1 de enero de 2025. [En línea]. Disponible en: https://scottambler-com.translate.goog/the-object-primer-3-e/?_x_tr_sl=en&_x_tr_tl=es&_x_tr_hl=es&_x_tr_pto=tc

[28] edrawsoft, “Wondershare EdrawMax”. Consultado: el 1 de enero de 2025. [En línea]. Disponible en: https://www.edrawmax.com/es/?stay=1

[29] C. A. Manosalvas Vaca y Y. L. Paredes Andrade, “Intención de Compra de Servicios de alojamiento a través de Redes Sociales: Aplicación del Modelo de Aceptación Tecnológica”, *INNOVA Research Journal*, vol. 6, núm. 2, pp. 274–281, may 2021, doi: 10.33890/innova.v6.n2.2021.1630.

ANEXOS

Anexo A. Fichas de observación a asistentes técnicos

Anexo A1: Ficha de observación a asistente técnico.

Ficha de observación				
Fecha y hora de la observación: 20/9/2024 10:37 **Tema:** Proceso de Asistencia Técnica **Duración total del soporte:** 25 min. **Objetivo:** Determinar los procesos involucrados en la gestión de asistencias técnicas a clientes en la empresa.				
Indicadores	**Observación**		**Novedades**	
Recepción de la solicitud				
¿Cómo se recibe la solicitud? (ticket, llamada, correo, etc.)	Llamada			
¿Se solicita información adicional al cliente?	Se solicita el ruc al cliente.			
Tiempo entre recepción e inicio del proceso	**Inmediata** x	**15 min**	**30 min**	**Más de 30 min**
Duración (De la fase)	3 min.			
Identificación y diagnóstico del problema				
Acciones del asistente para identificar el problema	Revisa el sistema del cliente, verifica la factura y sus datos internos, encuentra la causa de la anomalía, traspasa el soporte al líder del departamento, el líder de realiza cambios en el sistema, el soporte retorna al asistente, el asistente notifica el cliente.			
¿Se realiza alguna verificación inicial?	Se verifica que el documento no se encuentre autorizado por el SRI, además de verificar que datos posee la factura.		El asistente, al encontrar la causa de la anomalía y no poder modificar el sistema debido a su rol de usuario (admin) redirige el soporte al jefe del departamento (superadmin) para que este pueda corregir la anomalía.	
Comunicación con el cliente para confirmar detalles del problema.	El asistente se encuentra paciente y revisa el sistema. El cliente le explica la situación al asistente.			
Duración (De la fase)	7 min.			
Resolución del problema				

Acciones realizadas por el asistente técnico.	Verifica el sistema de la clienta, investiga la causa de la anomalía, traspasa el soporte a líder del departamento, el líder del departamento resuelve la anomalía y notifica al asistente, el asistente notifica al cliente	
Uso de herramientas o software específico.	Navegador Web (Mozilla)	
Verificación final con el cliente para asegurar la resolución.	No se realiza una verificación final.	
Duración (De la fase)	10 min.	
Seguimiento y cierre		
¿Se ofrece alguna indicación adicional al cliente para prevenir futuros problemas?	El asistente propone anular el documento anterior y generar uno nuevo con los datos correctos.	La anomalía no es encontrada en su totalidad, pero se propone una solución rápida al cliente.
Registro en la plataforma de gestión de tickets (si aplica).	Se registra el ticket tras haber encontrado la anomalía, se traspasa al líder y este devuelve el ticket editado al asistente para que lo finalice.	
Tiempo total del proceso de atención	25 min.	
Duración (De la fase)	5 min.	
Interacción con el cliente		
Descripción de los intercambios con el cliente durante el soporte.	El asistente y el cliente mantienen una conversación breve. El cliente solo reporta el problema, explica lo sucedido y se retira.	
Canales utilizados para comunicarse con el cliente (llamada, correo, chat, etc.).	Llamada, chat de WhatsApp	Se utiliza debido a que el soporte se realizó a lo largo de la llamada, una vez se corta la comunicación, se procede al chat de WhatsApp.
Notas adicionales		
Notas	El soporte se traspasa al líder del departamento en caso de que los asistentes no encuentren la causa de la anomalía.	

Anexo A2. Ficha de observación a asistente técnico

Ficha de observación		
Fecha y hora de la observación: 23/9/2024 11:58 **Tema:** Proceso de Asistencia Técnica **Duración total del soporte:** 30 min. **Objetivo:** Determinar los procesos involucrados en la gestión de asistencias técnicas a clientes en la empresa.		
Indicadores	**Observación**	**Novedades**

Recepción de la solicitud				
¿Cómo se recibe la solicitud? (ticket, llamada, correo, etc.)	Se recepta por llamada		Es atendida de inmediato tras sonar el teléfono	
¿Se solicita información adicional al cliente?	No se solicita, directamente se solicita al cliente realizar AnyDesk		El asistente posee un archivo Excel con el ruc de los clientes para acceder al sistema rápidamente	
Tiempo entre recepción e inicio del proceso	Inmediata x	15 min	30 min	Más de 30 min
Duración (De la fase)	1 min.			
Identificación y diagnóstico del problema				
Acciones del asistente para identificar el problema	Realizar AnyDesk para visualizar el sistema del cliente remotamente.			
¿Se realiza alguna verificación inicial?	Al iniciar la asistencia no se realiza, pero con forme se avanza en la revisión, se verifica la velocidad de internet, al tomar control del sistema, el asistente procede a verificar el apartado red de la consola de depuración del navegador.		Tras explicar las novedades el asistente procede a subir un pdf en ambos sistemas (del cliente y localmente) para verificar por qué no se sube.	
Comunicación con el cliente para confirmar detalles del problema.	Se posee buena comunicación entre cliente y asistente, el asistente es paciente y está abierto a responder cualquier duda o inquietud por parte del cliente. El cliente le explica la situación al asistente la última acción que realizo antes de presentarse la anomalía.			
Duración (De la fase)	7 min.			
Resolución del problema				
Acciones realizadas por el asistente técnico.	Revisar la consola de depuración del navegador para verificar las peticiones y respuestas entre computador y servidor. Además, también compara el sistema del cliente junto al sistema realizado localmente para verificar los procesos de creación de PDFs			
Uso de herramientas o software específico.	Utiliza la herramienta anydesk para revisar remotamente el sistema.		Tras 2 minutos, el asistente utiliza su navegador web para revisar el sistema del cliente en su computador.	
Verificación final con el cliente para asegurar la resolución.	No se concreta la asistencia debido a que el asistente aun no logra encontrar la causa de la anomalía.			
Duración (De la fase)	3 min.			
Seguimiento y cierre				

¿Se ofrece alguna indicación adicional al cliente para prevenir futuros problemas?	Si, el asistente le sugiere al cliente el no realizar acciones con las generaciones PDF tras identificar el problema.	Tras la revisión de la velocidad de internet, el asistente encontró que la causa de la animalia es el internet lento, por ende pide al cliente que le envie los documentos por privado para subirlos.
Registro en la plataforma de gestión de tickets (si aplica).	No se registra el soporte por parte del asistente, ya que se encontró la causa de la anomalía.	
Tiempo total del proceso de atención	26 min.	
Duración (De la fase)	15 min.	
Interacción con el cliente		
Descripción de los intercambios con el cliente durante el soporte.	El asistente solicita los documentos necesarios al cliente para realizar pruebas en un entorno local para con ello informar al cliente por qué se generó la anomalía.	
Canales utilizados para comunicarse con el cliente (llamada, correo, chat, etc.).	Llamada, chat de WhatsApp	Se utiliza debido a que el soporte se realizó a lo largo de la llamada, nunca se cortó la comunicación.
Notas adicionales		
Notas		

Anexo A3. Ficha de observación a asistente técnico

Ficha de observación				
Fecha y hora de la observación: 23/9/2024 15:43 **Tema:** Proceso de Asistencia Técnica **Duración total del soporte:** 15 min. **Objetivo:** Determinar los procesos involucrados en la gestión de asistencias técnicas a clientes en la empresa.				
Indicadores	**Observación**		**Novedades**	
Recepción de la solicitud				
¿Cómo se recibe la solicitud? (ticket, llamada, correo, etc.)	Mensaje de WhatsApp			
¿Se solicita información adicional al cliente?	No fue necesaria ya que el cliente adjunto una factura con los datos de la empresa.		Siempre se necesita corroborar el ruc del cliente para atenderlo.	
Tiempo entre recepción e inicio del proceso	Inmediata x	15 min ☐	30 min ☐	Más de 30 min ☐
Duración (De la fase)	3 min			
Identificación y diagnóstico del problema				

Acciones del asistente para identificar el problema	Se revisa el sistema del cliente debido a que la factura enviada no se muestra correctamente.	
¿Se realiza alguna verificación inicial?	Se revisa que la factura no se encuentre autorizada por el SRI.	
Comunicación con el cliente para confirmar detalles del problema.	Fluida, no se necesitó de llamadas o sesiones remotas, el cliente logro explicar de forma adecuada su inconveniente en mensajes de voz.	
Duración (De la fase)	5 min.	
Resolución del problema		
Acciones realizadas por el asistente técnico.	Entrar en el sistema del cliente como administrador, revisar la factura junto a los datos que la conforman (datos de los productos, datos del cliente, totales, etc.), encontrar la anomalía, notificar la anomalía, registrar el ticket.	
Uso de herramientas o software específico.	Navegador web (Mozilla)	
Verificación final con el cliente para asegurar la resolución.	Se notifica al cliente que un producto posee caracteres especiales.	
Duración (De la fase)	5 min.	
Seguimiento y cierre		
¿Se ofrece alguna indicación adicional al cliente para prevenir futuros problemas?	Se le indica al cliente el generar pdf e imprimir nuevamente la factura.	
Registro en la plataforma de gestión de tickets (si aplica).	Si se realiza, una vez se concluye con la asistencia.	
Tiempo total del proceso de atención	15 min.	
Duración (De la fase)	2 min.	
Interacción con el cliente		
Descripción de los intercambios con el cliente durante el soporte.	Solo se intercambia un documento, el cual es una foto de una factura.	
Canales utilizados para comunicarse con el cliente (llamada, correo, chat, etc.).	Mensajes de WhatsApp	
Notas adicionales		
Notas		

Anexo A4. Ficha de observación a asistente técnico

<table>
<tr><th colspan="5">Ficha de observación</th></tr>
<tr><td colspan="5">Fecha y hora de la observación: 24/9/2024 9:47
Tema: Proceso de Asistencia Técnica
Duración total del soporte: 14 min.
Objetivo: Determinar los procesos involucrados en la gestión de asistencias técnicas a clientes en la empresa.</td></tr>
<tr><td>Indicadores</td><td colspan="2">Observación</td><td colspan="2">Novedades</td></tr>
<tr><td colspan="5">Recepción de la solicitud</td></tr>
<tr><td>¿Cómo se recibe la solicitud? (ticket, llamada, correo, etc.)</td><td colspan="2">Mensaje de WhatsApp</td><td colspan="2"></td></tr>
<tr><td>¿Se solicita información adicional al cliente?</td><td colspan="2">El cliente remite el número de ruc de su negocio para verificar si está asociado como cliente en la empresa.</td><td colspan="2">También envía una foto de una cedula.</td></tr>
<tr><td>Tiempo entre recepción e inicio del proceso</td><td>Inmediata
x</td><td>15 min</td><td>30 min</td><td>Más de 30 min</td></tr>
<tr><td>Duración (De la fase)</td><td colspan="2">9:48 (1 min.)</td><td colspan="2"></td></tr>
<tr><td colspan="5">Identificación y diagnóstico del problema</td></tr>
<tr><td>Acciones del asistente para identificar el problema</td><td colspan="2">El asistente ingresa al sistema del cliente mediante usuario administrador, revisa el apartado de personal para ingresar un nuevo personal.</td><td colspan="2">Adicional, abre la consola de depuración del navegador para revisar solicitudes y respuestas entre servidor y máquina.</td></tr>
<tr><td>¿Se realiza alguna verificación inicial?</td><td colspan="2">El asistente comprueba el número de ruc en el sistema de soporte para verificar si está asociada.</td><td colspan="2"></td></tr>
<tr><td>Comunicación con el cliente para confirmar detalles del problema.</td><td colspan="2">La comunicación se realiza de forma educada y el cliente envía mensajes de voz donde especifica su problema.</td><td colspan="2"></td></tr>
<tr><td>Duración (De la fase)</td><td colspan="2">9:50</td><td colspan="2"></td></tr>
<tr><td colspan="5">Resolución del problema</td></tr>
<tr><td>Acciones realizadas por el asistente técnico.</td><td colspan="2">Tras revisar el sistema del cliente, envía el número de ruc a la líder del departamento, para que pueda verificar el sistema a profundidad.</td><td colspan="2">El líder realiza una modificación en el sistema, resulta ser una firma electrónica invalida en el sistema, la líder lo actualizo y se trasladó el soporte con el asistente.</td></tr>
<tr><td>Uso de herramientas o software específico.</td><td colspan="2">Navegadores web (Mozilla)</td><td colspan="2"></td></tr>
<tr><td>Verificación final con el cliente para asegurar la resolución.</td><td colspan="2">Se verifica si el cliente puede emitir una factura con la firma electrónica que subió.</td><td colspan="2"></td></tr>
<tr><td>Duración (De la fase)</td><td colspan="2">9:56</td><td colspan="2"></td></tr>
<tr><td colspan="5">Seguimiento y cierre</td></tr>
</table>

¿Se ofrece alguna indicación adicional al cliente para prevenir futuros problemas?	Se le indica al cliente que refresque su sistema y con ello pueda emitir facturas nuevamente.	
Registro en la plataforma de gestión de tickets (si aplica).	Si se registró, para traspasar el soporte a la líder en el departamento, posteriormente el ticket es devuelto hacia el asistente.	
Tiempo total del proceso de atención	12 min.	
Duración (De la fase)	9:59	
Interacción con el cliente		
Descripción de los intercambios con el cliente durante el soporte.	Se solicita al cliente su número de ruc, se realiza verificación en el sistema, se identifica el problema, se traspasa ticket de ser necesario, se resuelve el ticket, se notifica al cliente.	
Canales utilizados para comunicarse con el cliente (llamada, correo, chat, etc.).	Siempre se utilizó el chat de WhatsApp.	
Notas adicionales		
Notas		

Anexo A5. Ficha de observación a asistente técnico

Ficha de observación				
Fecha y hora de la observación: 24/9/2024 15:43 **Tema:** Proceso de Asistencia Técnica **Duración total del soporte:** 16 min **Objetivo:** Determinar los procesos involucrados en la gestión de asistencias técnicas a clientes en la empresa.				
Indicadores	**Observación**		**Novedades**	
Recepción de la solicitud				
¿Cómo se recibe la solicitud? (ticket, llamada, correo, etc.)	Llamada telefónica			
¿Se solicita información adicional al cliente?	El asistente solicita el reporte de facturas cobradas para poder verificar el archivo en su totalidad.		La cliente solicita realizar un AnyDesk para poder revisar el reporte junto al asistente.	
Tiempo entre recepción e inicio del proceso	Inmediata x	15 min	30 min	Más de 30 min
Duración (De la fase)	3 min			
Identificación y diagnóstico del problema				

Acciones del asistente para identificar el problema	Se revisa el sistema del cliente remotamente para poder verificar los datos debido a que en el listado de facturas existen facturas que no se muestran en el reporte.	El asistente explica amablemente a la clienta por qué algunas facturas no se muestran correctamente.
¿Se realiza alguna verificación inicial?	Se generan los reportes del cliente para comparar junto al listado de facturas.	
Comunicación con el cliente para confirmar detalles del problema.	El cliente y el asistente dialogan sobre el motivo del porque no se muestran las facturas.	
Duración (De la fase)	5 min.	
Resolución del problema		
Acciones realizadas por el asistente técnico.	Revisar los reportes generados remotamente por parte del asistente y comparar con la lista de facturas.	
Uso de herramientas o software específico.	AnyDesk	
Verificación final con el cliente para asegurar la resolución.	Se explica a la clienta que las facturas no poseen un estado de autorizadas, las cuales se corroboran al momento de mostrarse el listado de facturas.	
Duración (De la fase)	5 min.	
Seguimiento y cierre		
¿Se ofrece alguna indicación adicional al cliente para prevenir futuros problemas?	En este caso no, dado que solo fue una confusión por parte del cliente.	
Registro en la plataforma de gestión de tickets (si aplica).	Se registra el soporte en el sistema de tickets, explicando que exigió la clienta, que se realizó, los datos de la clienta y el sistema que utiliza.	
Tiempo total del proceso de atención	8 min	
Duración (De la fase)	3 min	
Interacción con el cliente		
Descripción de los intercambios con el cliente durante el soporte.	No se realizó ningún intercambio, solo se explicó por Anydesk	
Canales utilizados para comunicarse con el cliente (llamada, correo, chat, etc.).	Llamada	
Notas adicionales		
Notas		

Anexo B.Anexo B. Fichas de observación a agentes de venta

Anexo B1: Ficha de observación a agente de ventas.

Ficha de observación			
Fecha y hora de la observación: 23/09/2024 10:56 **Tema:** Proceso de soporte por parte del departamento de ventas **Duración total del soporte:** 1 hora y 15 min. **Objetivo:** Determinar los procesos involucrados en la gestión de asistencias técnicas a clientes en la empresa.			
Indicadores	**Observación**		**Novedades**
Recepción de la solicitud			
¿Cómo se recibe la solicitud? (ticket, llamada, correo, etc.)	La solicitud se recibe por medio de ticket		
¿Cuál es la actitud del agente?	Se ha observado una mejora en la cordialidad del agente.		
¿Se solicita información adicional al cliente?	Si para poder solventar de manera rápida y eficaz su requerimiento o queja		En algunos casos llama todo el personal de la empresa y la información se distorsiona
¿El agente identifica rápidamente las necesidades del cliente?	Si x	No 	
Duración (De la fase)	10 minutos		
Presentación del producto / servicio			
¿Cómo presenta el agente el producto o servicio?	La presentación del servicio se lo hace por medio de videoconferencias en donde el cliente evalúa y decide trabajar con el producto.		
¿Se mencionan características?	Se mencionan pros y contras que tiene nuestro servicio para evitar problemas a futuro con el cliente		
¿Se realizan demostraciones?	Si, se lo hace por medio de la videoconferencia mediante cita agendada.		
¿Qué herramientas se utilizó para presentar el producto? ¿Fueron suficientes?	Google Meet		
Duración (De la fase)	30 minutos		
Objeciones			
¿El cliente presenta dudas?	Si, al ser un programa que maneja múltiples módulos como facturación, clientes, inventario y demás, el cliente requiere de repetir explicaciones para un manejo adecuado del programa		
¿El agente responde las dudas con certeza?	Si		

¿El agente se muestra positivo y proactivo ante las objeciones?	Si	
Duración (De la fase)	20 minutos	
Cierre y Seguimiento		
¿Se llego a algún acuerdo?	Si, el agente le explica el proceso de adquisición del servicio al cliente.	
¿Qué decisión toma el cliente?	El cliente decide adquirir el servicio	
¿Se ofrece alguna indicación adicional al cliente?	Cuanto tiempo toma el adquirir el servicio	
¿Se acuerda una acción futura?	Si, el comprobante de pago enviado por mensaje de texto o correo	
Tiempo total del proceso de atención	10 minutos	
Actitud del cliente durante el soporte	Positiva	
Duración (De la fase)	5 minutos	
Notas adicionales		
Notas	Ninguna	

Anexo B2. Ficha de observación a agente de ventas.

Ficha de observación		
Fecha y hora de la observación: 24/09/2024 8:34 **Tema:** Proceso de soporte por parte del departamento de ventas **Duración total del soporte:** 45 min. **Objetivo:** Determinar los procesos involucrados en la gestión de asistencias técnicas a clientes en la empresa.		
Indicadores	**Observación**	**Novedades**
Recepción de la solicitud		
¿Cómo se recibe la solicitud? (ticket, llamada, correo, etc.)	La solicitud se recibe por medio de llamada	A veces los clientes no tienen buena señal por lo que se solicita que envíen sus requerimientos por chat.
¿Cuál es la actitud del agente?	Cordial	
¿Se solicita información adicional al cliente?	Si, el requerimiento del cliente no es concreto por lo que se pide escribir sus necesidades.	
¿El agente identifica rápidamente las necesidades del cliente?	Si [x] No [ ]	
Duración (De la fase)	5 minutos	
Presentación del producto / servicio		

¿Cómo presenta el agente el producto o servicio?	La presentación del servicio se lo hace por medio de videoconferencia	
¿Se mencionan características?	Si. Se especifica lo que tiene y ofrece cada servicio	
¿Se realizan demostraciones?	Si, en la videoconferencia con el cliente	
¿Qué herramientas se utilizó para presentar el producto? ¿Fueron suficientes?	Se utilizó Google meet	Fueron suficientes para presentar el sistema.
Duración (De la fase)	20 minutos	
Objeciones		
¿El cliente presenta dudas?	Si, el cliente no posee mucha experiencia con el manejo de tecnología.	
¿El agente responde las dudas con certeza?	Si	
¿El agente se muestra positivo y proactivo ante las objeciones?	Si	
Duración (De la fase)	10 minutos	
Cierre y Seguimiento		
¿Se llego a algún acuerdo?	Si, se le brindó una solución al cliente.	
¿Qué decisión toma el cliente?	Esperar al turno en el que el problema va a hacer solucionado.	
¿Se ofrece alguna indicación adicional al cliente?	El tiempo que tarde en solucionarse su petición.	
¿Se acuerda una acción futura?	Si, una llamada para notificar la petición solicitada.	
Tiempo total del proceso de atención	5 minutos	
Actitud del cliente durante el soporte	Positiva	
Duración (De la fase)	10 minutos	
Notas adicionales		
Notas		

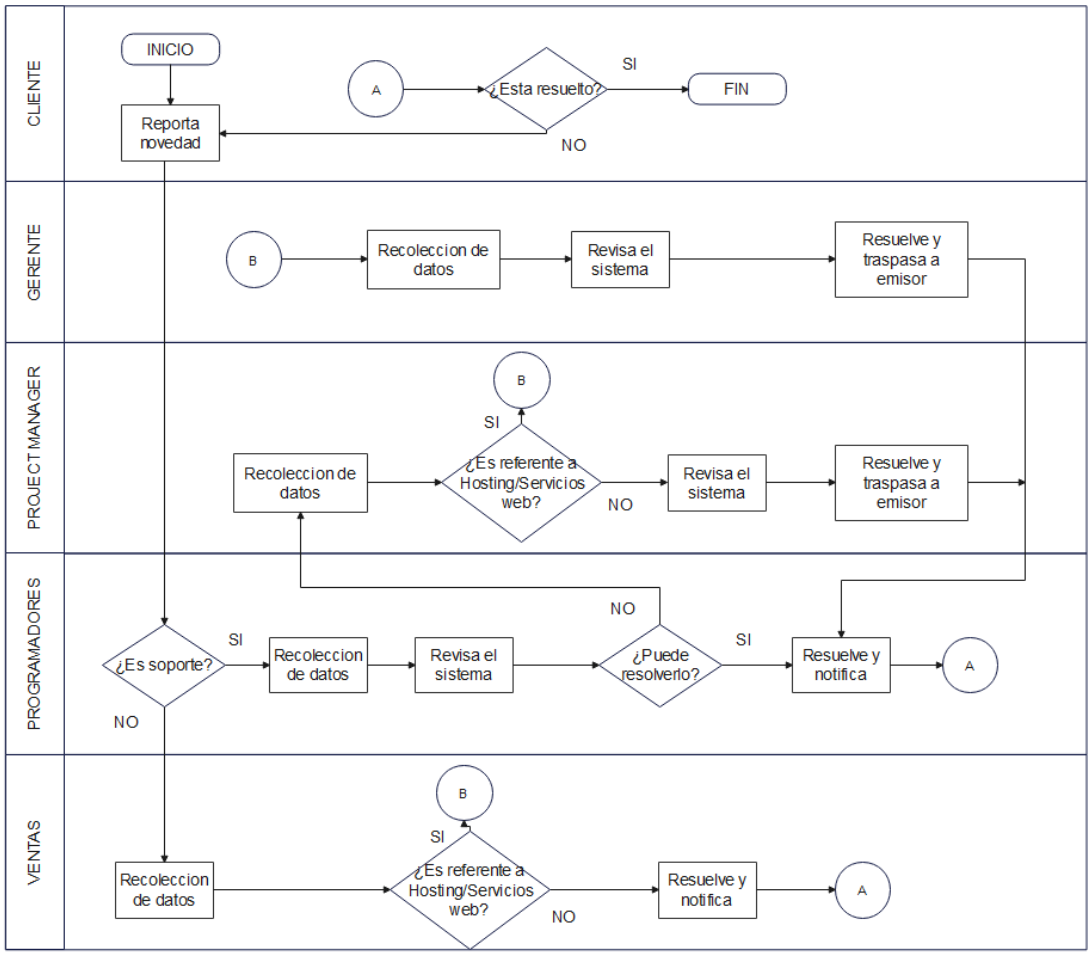
Anexo B3. Ficha de observación a agente de ventas.

Ficha de observación		
Fecha y hora de la observación: 24/09/2024 15:48 **Tema:** Proceso de soporte por parte del departamento de ventas **Duración total del soporte:** 45 min. **Objetivo:** Determinar los procesos involucrados en la gestión de asistencias técnicas a clientes en la empresa.		
Indicadores	**Observación**	**Novedades**
Recepción de la solicitud		

¿Cómo se recibe la solicitud? (ticket, llamada, correo, etc.)	La solicitud se recibe por medio de correo		
¿Cuál es la actitud del agente?	Atenta porque cuando por correo se contesta el mensaje o se le llama		
¿Se solicita información adicional al cliente?	El cliente informa su requerimiento por escrito pero no se entiende, por lo que se solicita información adicional		
¿El agente identifica rápidamente las necesidades del cliente?	Si x	No	
Duración (De la fase)	5 minutos		
Presentación del producto / servicio			
¿Cómo presenta el agente el producto o servicio?	La presentación del servicio se lo hace por medio de videoconferencias y llamadas		
¿Se mencionan características?	Si, para solventar todas las dudas de los clientes		
¿Se realizan demostraciones?	Si, se lo hace por medio de AnyDesk o llamada		
¿Qué herramientas se utilizó para presentar el producto? ¿Fueron suficientes?	Meet Google, zoom o AnyDesk		
Duración (De la fase)	15 minutos		
Objeciones			
¿El cliente presenta dudas?	Si, el cliente no sabe cómo anular o eliminar ciertas cosas del programa.		
¿El agente responde las dudas con certeza?	Si		
¿El agente se muestra positivo y proactivo ante las objeciones?	Si		
Duración (De la fase)	5 minutos		
Cierre y Seguimiento			
¿Se llego a algún acuerdo?	Si, se le da solución a su duda o requerimiento		
¿Qué decisión toma el cliente?	La decisión de agendar un turno para que el requerimiento sea atendido y solventado		
¿Se ofrece alguna indicación adicional al cliente?	No		
¿Se acuerda una acción futura?	SI		
Tiempo total del proceso de atención	30 minutos		
Actitud del cliente durante el soporte	Positiva		
Duración (De la fase)	20 minutos		

Notas adicionales	
Notas	En algunos casos los clientes dan el mando a algunas personas para que realicen modificaciones en el sistema y la información se distorsiona ya que se comunican 2 personas a la misma vez

Anexo C. Diagrama de procesos para soporte técnico



Descripción de procesos

Tabla 103. Descripción de procesos

N°	Proceso	Descripción
1	Reporta novedad	El cliente notifica una anomalía o requerimiento en un medio de comunicación (correo, WhatsApp, llamada telefónica). En caso de que no se solucione, el cliente puede volver a reportar la anomalía.

N°	Proceso	Descripción
2	Recolección de datos	Se solicita los datos necesarios al emisor, entre ellos el RUC para comprobar que servicio o sistema posee el cliente, una captura o video de donde encontró el incidente y por último cual es el proceso que se realizó (en caso de traspasar el soporte). El programador es quién clasifica el tipo de incidencia y a corde al mismo lo traspasa al encargado adecuado (ventas, PM o gerente)
3	Revisa el sistema	Se verifica el sistema de un cliente internamente, puede necesitar de revisión remota junto con el cliente en sesión de AnyDesk, llamada por parte del asistente hacia el cliente para clarificar explicaciones.
4	Resuelve y notifica	Notifica al cliente una vez se resuelve el ticket que se registró.
5	Resuelve y traspasa a emisor	En caso de que un programador no pueda solucionar el inconveniente, se pasa el inconveniente a el PM para que lo pueda resolver, es en este punto donde se crea un ticket para relatar el proceso realizado.
6	Reabrir ticket registrado	El cliente, en el caso de que un ticket no se encuentre solventado, puede aperturar y notificar nuevamente el mismo ticket para que se repita el proceso de revisión.

Anexo D. Formato para Historias de Usuario

HISTORIA DE USUARIO		**#1**
Usuario:		
Nombre:		
Prioridad en el negocio:	**Riesgo de desarrollo:**	
Puntos estimados:	**Iteración asignada:**	
Programador a cargo:		
Descripción:		
Observación:		

Anexo E. Manual de usuario

CONNECTASERVICES

Av. Rodrigo Pachano y Los Guaytambos. Ambato - Ecuador

Manual de Usuario



Helpdesk

LOGIN



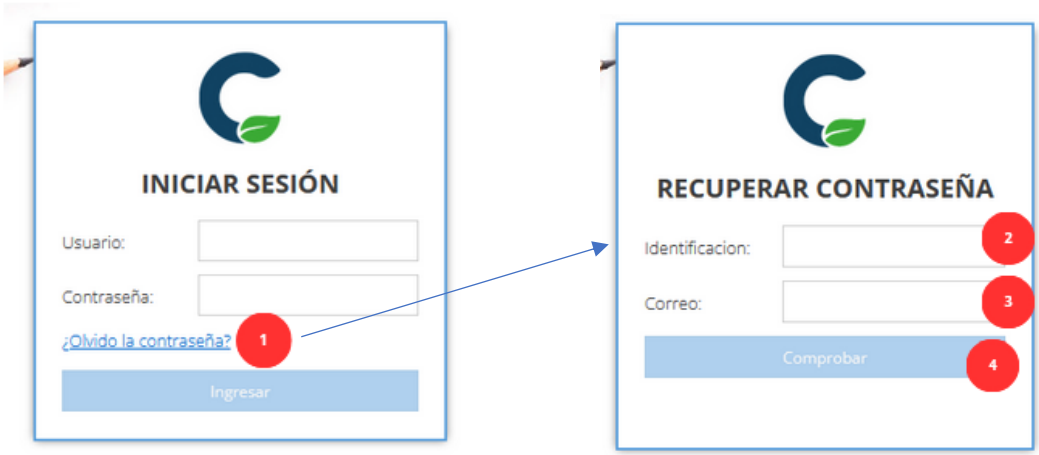
1. Ingresa tu usuario

2. Ingresa tu contraseña

3. Ingresar al sistema

RECUPERAR CONTRASEÑA

En caso de extraviar u olvidar tu contraseña:



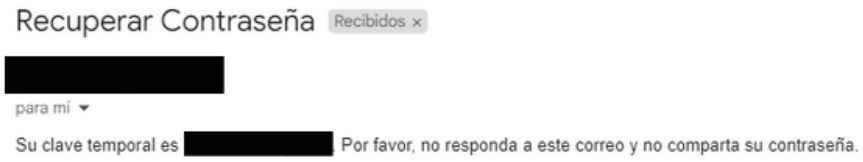
1. Dar click en el texto “¿Olvido la contraseña?”

2. Ingresa tu número de cédula

3. Ingresa tu correo con el cual te registraste a nuestros servicios

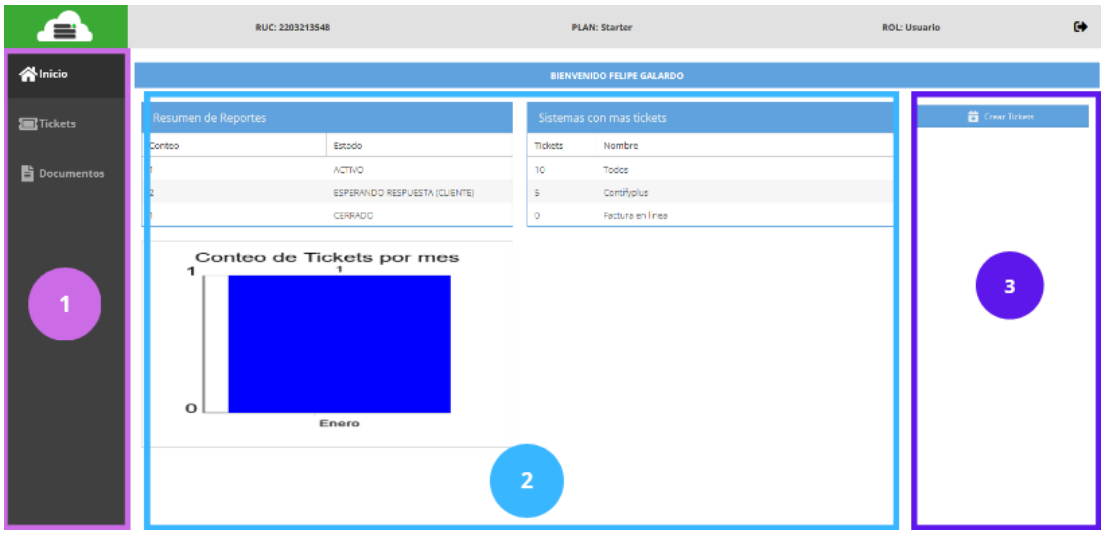
4. Dar click en el botón comprobar.

Un correo se enviará con tu nueva contraseña con la cual puedes acceder al sistema.



PANTALLA PRINCIPAL

El menú principal esta compuesto por tres secciones:



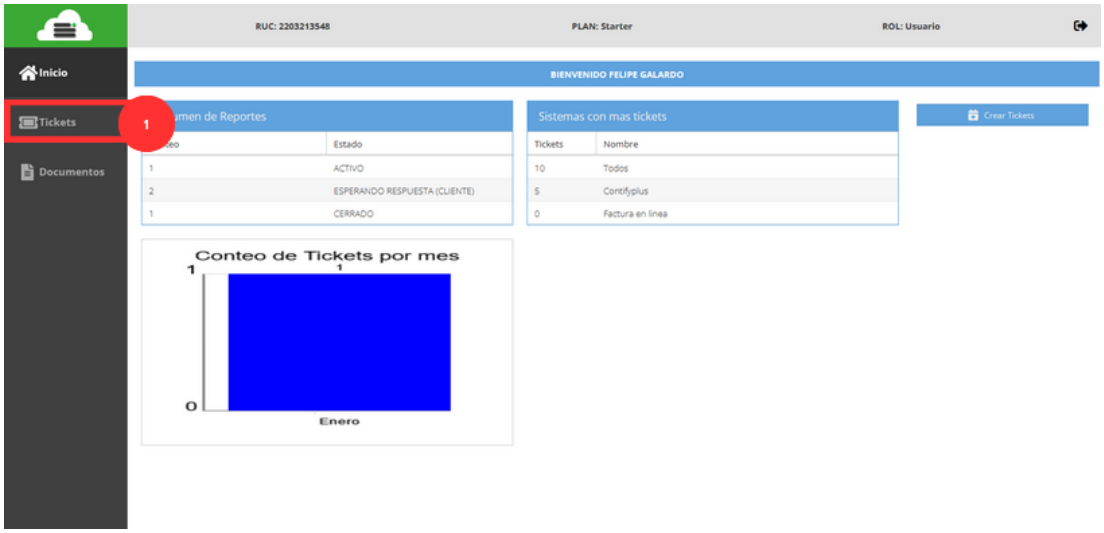
1. Menú de módulos: Donde puedes navegar a través del sistema para crear tickets, verificar usuarios y sistemas o realizar reportes

2. Dashboard: Página principal que te muestra los datos mas relevantes de tu uso en el sistema.

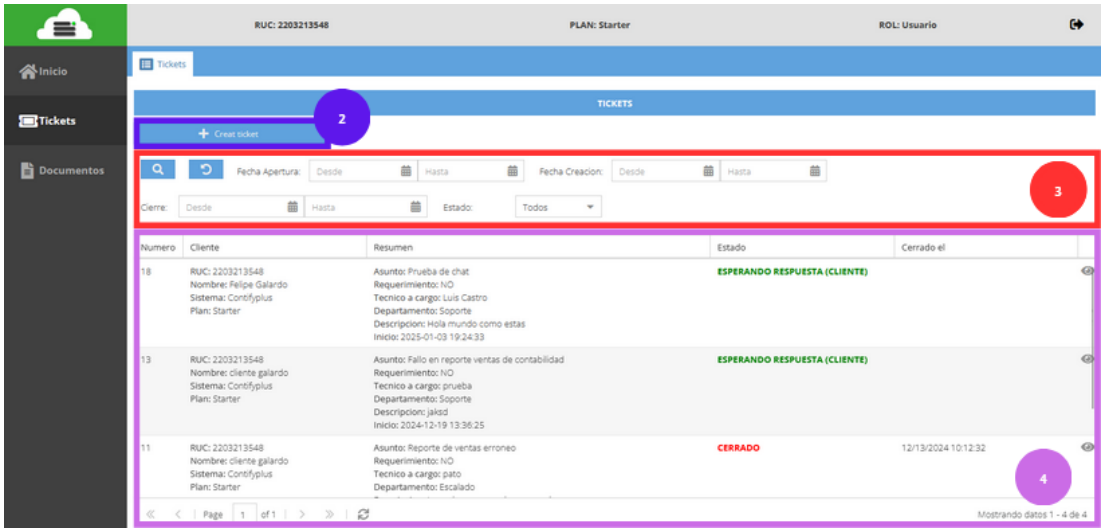
3. Sección de acciones rápidas: Botones que te ayudan a realizar acciones de forma rápida en lugar de navegar a través de los módulos.

TICKETS

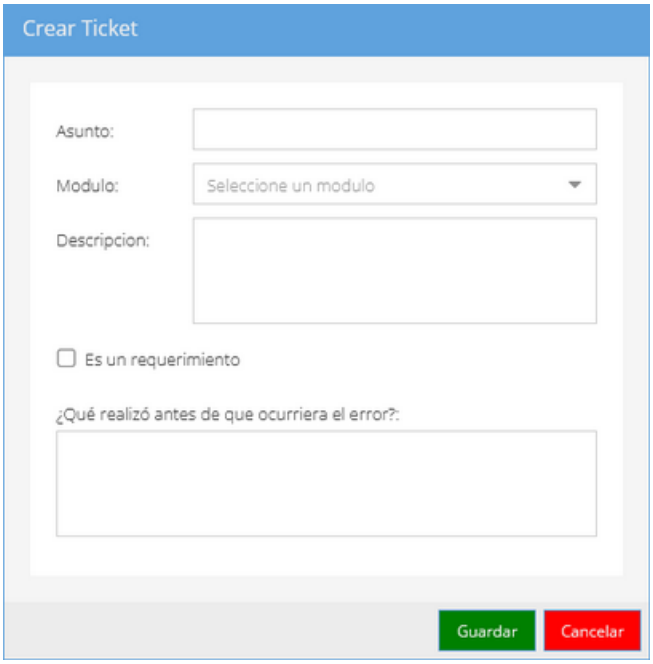
Para acceder al módulo Tickets damos click en la segunda opción del menú izquierdo.



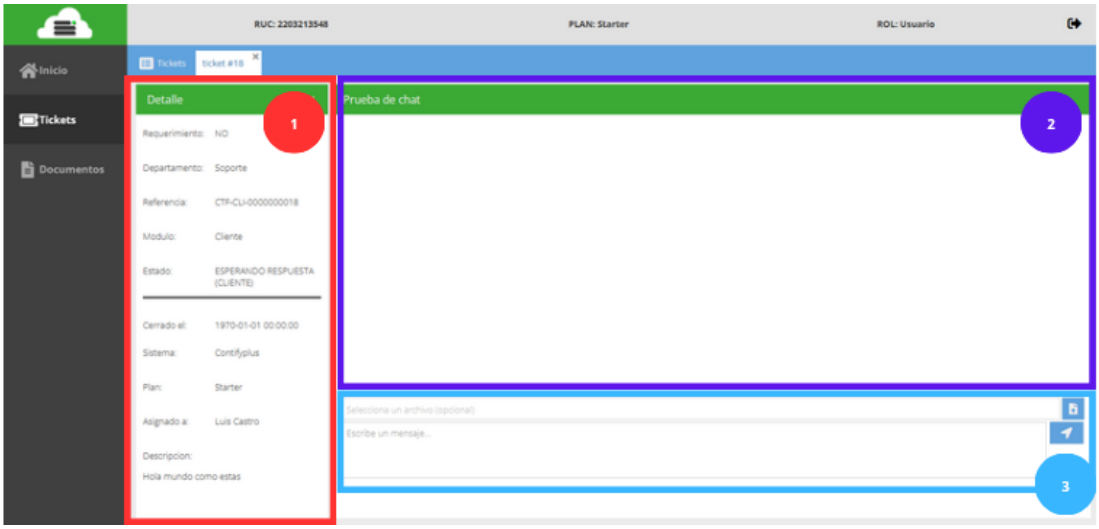
En el podremos verificar los siguientes componentes:



2. Botón para crear un ticket: Al dar click en el botón, se abrirá una ventana flotante que permitirá el registro de un nuevo ticket, todos los datos son requeridos para la creación del ticket.



- Tras llenar todos los campos, damos al botón “Guardar”, el cual mostrará el detalle del ticket creado, junto con la sala de chat asociada al ticket para enviar mensajes entre cliente y técnico.



- **La sección 1** muestra el detalle del ticket completamente.

- **La sección 2** es el espacio para mostrar los chats del ticket, utiliza la rueda del ratón para navegar en el chat.

- **La sección 3** es el formulario para enviar mensajes, es obligatorio escribir en el segundo cuadro de texto, mientras que es opcional el seleccionar un archivo para enviarlo en el mensaje.

3. Filtros aplicables a la búsqueda: Para aplicar los filtros con los datos ingresados se debe dar click en el botón buscar (botón azul con símbolo de lupa).

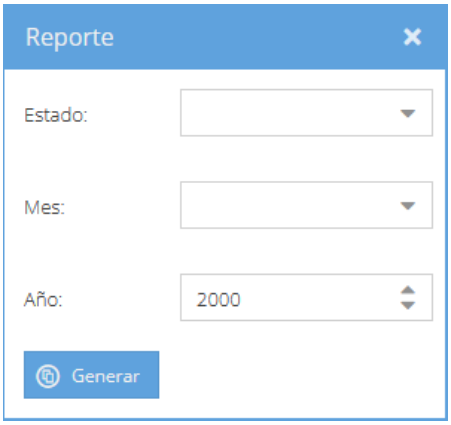
4. Lista de tickets: Muestra la información relevante de cada ticket, en el extremo derecho se encuentra el botón para visualizar el detalle del ticket.

REPORTES



1. Dar click en la viñeta “Documentos”, abrirá en la pagina principal el modulo de Reportes, en los cuales podremos verificar nuestros reportes.

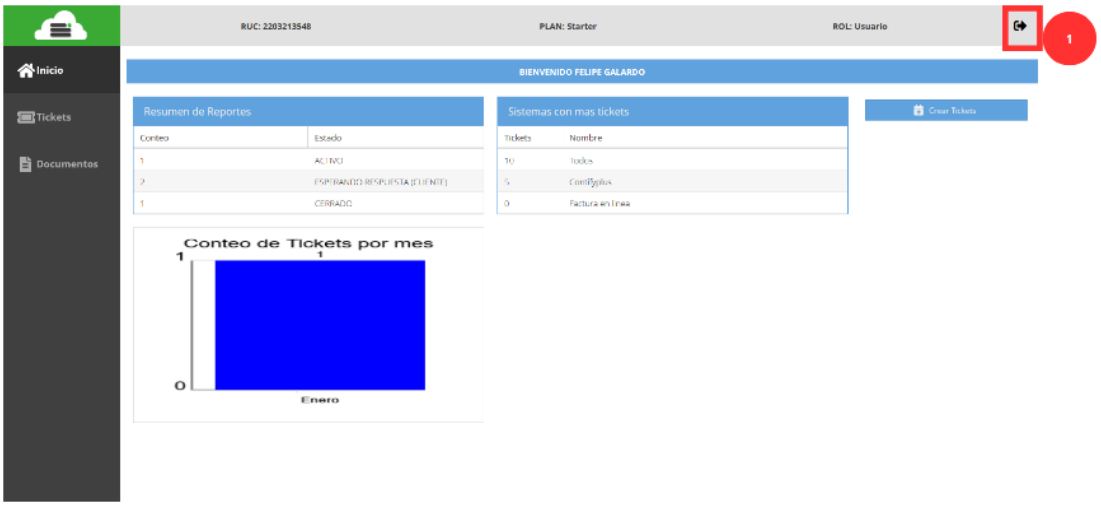
2. La sección 2 proporciona reportes por estados, meses y años, al presionar click en el primer botón, aparecerá una ventana flotante el cual permitirá ingresar los datos requeridos para la realización del reporte, los campos no son obligatorios.



Al presionar click en el botón de “Generar” se generará el reporte y se abrirá en una ventana nueva.

3. La sección 3 es el botón que muestra el presente manual de usuario.

SALIR



Para salir del sistema y cerrar la sesión del usuario solo tiene que dar click en el símbolo del botón 1 (resaltado en rojo). Al presionarlo, se redirigirá a la pantalla de Iniciar Sesión.