



**Ciencias y Tecnologías Computacionales**  
**Sistema Gestión de Reportes de Incidencias Dinámico para**  
**los servicios de soportes en la Universidad de las Ciencias**  
**Informáticas**

**Trabajo de diploma para optar por el título de**  
**Ingeniero en Ciencias Informáticas**

Autor(es): Yaneysi Velázquez Silva  
Tutor(es): MsC. Neybis Lago Clara  
Co-tutor: Ing. Víctor E. Cabrera Sanz

La Habana, mayo del 2022

“Año 64 de la Revolución”

## **DECLARACIÓN DE AUTORÍA**

El autor Yaneyssi Velázquez Silva con el carné de identidad 99081117859, declara ser el autor de la presente tesis que tiene por título: Sistema de gestión de reportes para los servicios de soporte y reconocemos a la Universidad de las Ciencias Informáticas los derechos patrimoniales de la misma, con carácter exclusivo.

Yaneyssi Velázquez Silva

---

Firma del Autor

MSc. Neybis Lago Clara

---

Firma del Tutor

Ing. Víctor E Cabrera Sanz

---

Firma del Tutor

## **DATOS DE CONTACTO**

Nombre: MsC. Neybis Lago Clara.

Correo: [nlago@uci.cu](mailto:nlago@uci.cu)

Categoría Científica: Master en Ciencias.

Lugar de Trabajo: Centro Soporte

Nombre: Ing. Víctor E. Cabrera Sanz.

Correo: [vecabrera@uci.cu](mailto:vecabrera@uci.cu)

Categoría Científica: Ingeniero.

Lugar de Trabajo: Centro Soporte

## **AGRADECIMIENTOS**

### **A MI FAMILIA:**

“Ustedes han sido siempre el motor que impulsa mis sueños y esperanzas, quienes estuvieron siempre a mi lado en los días y noches más difíciles. Siempre han sido mis mejores guías de vida. Hoy cuando concluyo mis estudios, les dedico a ustedes este logro porque cada uno de ustedes formaron un rol muy importante durante estos años de estudios. Orgullosa de tenerlos como mi familia, que estén a mi lado en este momento tan importante lo es todo para mí. Gracias por ser quienes son y por creer en mí”

### **A MIS TUTORES:**

“Neybis Lago Clara y Víctor E Cabrera Sanz. Sin ustedes y sus virtudes, su paciencia y constancia este trabajo no lo hubiese logrado tan fácil. Sus consejos fueron siempre útiles cuando no salían de mi pensamiento las ideas para escribir lo que hoy he logrado. Ustedes formaron parte importante de esta historia con sus aportes profesionales que lo caracterizan. Muchas gracias por sus múltiples palabras de aliento, cuando más las necesite; por estar allí cuando mis horas de trabajo se hacían confusas. Gracias.

### **A MIS COMPAÑEROS:**

“Mis amigos y compañeros de escuela, ya culminan esta maravillosa aventura y no puedo dejar de recordar cuantas tardes y horas de estudio nos juntamos a lo largo de nuestra formación. Ya nos toca cerrar un capítulo maravilloso en esta historia de vida y no puedo dejar de agradecerles por su apoyo y constancia, al estar en las horas más difíciles, por compartir horas de estudio. Gracias por estar siempre allí.”

### **A CADA PERSONA:**

Quiero agradecer a cada persona que no es familia mía y tampoco amigo y sin pensarlo me ayudaron a lograr esta meta en mi vida, quizás para ellos fue insignificante, pero para mí lo fue todo. No voy a poner nombres porque no me alcanzaría la hoja, pero cada uno de ellos lo saben y quiero que sepan que estoy muy agradecida. Gracias.

## RESUMEN

En Cuba existe la Universidad de las Ciencias Informáticas (UCI), la cual en su estructura cuenta con centros dedicados al desarrollo de software. Entre ellos se encuentra el Centro de Soporte Tecnológico, dedicado al mantenimiento de los productos que son desarrollados por la Red de Centros de la universidad. El centro de soporte tiene como misión: Brindar el servicio de soporte técnico a las aplicaciones y servicios informáticos desarrollados por la UCI con calidad y eficiencia a partir de una correcta gestión de los mismos. Entre los servicios que brinda el centro se encuentra el reporte y resolución de incidencias. El sistema utilizado Zammad, actualmente permite la generación de reportes, pero el número de filtros que posee son limitados, el formato de exportación permitido es solo Excel y los campos que refleja tienen como inconveniente que, los nombres de los campos salen solo en inglés, no incluye la descripción de las incidencias, no es posible modificar el formato de las fechas asociadas a las incidencias y no permite seleccionar que campos visibilizar. Para el desarrollo de la presente aplicación informática se realiza una propuesta de solución basada en la gestión de reportes dinámicos, la cual tiene como objetivo general desarrollar un sistema de gestión de reportes dinámicos que permita seleccionar y mostrar las incidencias de los servicios de soporte. Para el desarrollo de la solución se empleó como framework readJS, metodología AUP\_UCI y como gestor de base de datos PostgreSQL y para validar la solución se aplicaron pruebas funcionales y pruebas de aceptación. Como resultado se obtuvo un sistema de gestión de reportes de incidencias para los servicios de soportes.

**PALABRAS CLAVES:** sistema, reportes, incidencias, software, gestión de reportes.

**ABSTRACT**

In Cuba there is the University of Informatics Sciences (UCI), which in its structure has centers dedicated to software development. Among them is the Technological Support Center, dedicated to the maintenance of the products that are developed by the Network of Centers of the university. The support center's mission is to: Provide technical support services to computer applications and services developed by the UCI with quality and efficiency based on their correct management. Among the services provided by the center is the reporting and resolution of incidents. The system used by Zammad currently allows the generation of reports, but the number of filters it has are limited, the export format allowed is only Excel and the fields it reflects have the drawback that the names of the fields appear only in English, it does not include the description of the incidents, it is not possible to modify the format of the dates associated with the incidents and it does not allow selecting which fields to display. For the development of this computer application, a solution proposal is made based on the management of dynamic reports, which has the general objective of developing a dynamic report management system that allows selecting and displaying the incidents of the support services. . For the development of the solution, ReactJS framework, AUP\_UCI methodology and PostgreSQL database manager were used as a framework and functional tests and acceptance tests were applied to validate the solution. As a result, an incident report management system for support services was obtained.

**KEY WORDS:** system, reports, incidents, software, report management.

**Índice**

|  |    |
|--|----|
| INTRODUCCIÓN.....  | 1  |
| CAPÍTULO I: FUNDAMENTOS Y REFERENTES TEÓRICO-METODOLÓGICOS DEL SISTEMA GESTIÓN DE REPORTES DE INCIDENCIAS PARA LOS SERVICIOS DE SOPORTES EN LA UNIVERSIDAD DE LAS CIENCIAS INFORMÁTICAS..... | 6  |
| 1.1 Introducción.....  | 6  |
| 1.2 Conceptos asociados.....   | 6  |
| 1.3 Soluciones existentes.....   | 8  |
| 1.4 Metodología utilizada en la solución propuesta.....  | 10 |
| 1.5 Tecnologías y Herramientas asociadas al desarrollo del sistema.....  | 11 |
| 1.5.1- Tecnologías.....  | 12 |
| 1.5.2- Tecnologías de desarrollo.....  | 13 |
| Conclusiones del capítulo:.....  | 14 |
| CAPÍTULO II: ANÁLISIS, DISEÑO E IMPLEMENTACIÓN DEL SISTEMA DE GESTIÓN DE REPORTES DE INCIDENCIAS PARA LOS SERVICIOS DE SOPORTES.....   | 16 |
| 2.1 Introducción.....  | 16 |
| 2.2 Modelo del dominio.....  | 16 |
| 2.3 Solución Propuesta.....  | 17 |
| 2.4 Concepción del sistema.....  | 18 |
| 2.4.1 Visión y alcance del sistema.....  | 18 |
| 2.4.2 Planificación del proyecto por roles.....  | 18 |
| 2.5 Especificación de requisitos.....  | 21 |
| 2.5.1 Lista de reserva del producto.....   | 21 |
| 2.5.2 Historias de usuario.....  | 24 |
| 2.6 Arquitectura del sistema.....  | 28 |
| 2.7 Patrones del Diseño.....   | 29 |
| 2.7.1 Patrones GRASP.....  | 30 |

|   |           |
|---|-----------|
| 2.7.2 Patrones GOF.....   | 33        |
| 2.8 Diagrama Entidad Relación.....  | 34        |
| 2.9 Conclusiones del Capítulo.....  | 35        |
| <b>CAPÍTULO III: PRUEBAS REALIZADAS AL SISTEMA DE GESTIÓN DE REPORTES DE<br/>INCIDENCIAS PARA LOS SERVICIOS DE SOPORTE.....</b> | <b>36</b> |
| 3.1 Introducción.....   | 36        |
| 3.2 Diagrama de despliegue.....   | 36        |
| 3.4.1 Prueba de Caja Blanca.....  | 38        |
| 3.4.2 Prueba de Aceptación.....   | 40        |
| 3.5 Interfaces principales del Sistema de gestión de reportes de incidencias.....   | 40        |
| 3.6 Conclusiones del capítulo.....  | 43        |
| <b>CONCLUSIONES FINALES.....</b>  | <b>44</b> |
| <b>RECOMENDACIONES.....</b>   | <b>45</b> |
| <b>BIBLIOGRAPHY.....</b>  | <b>46</b> |
| <b>ANEXOS.....</b>  | <b>49</b> |
| <b>ANEXO #1 HISTORIAS DE USUARIOS.....</b>  | <b>49</b> |
| <b>ANEXO #2 ACTA DE ACEPTACIÓN.....</b>   | <b>55</b> |



**ÍNDICE DE TABLAS**

|  |    |
|--|----|
| <i>Tabla 1:Roles y responsabilidades en el desarrollo del sistema.....</i> | 19 |
| <i>Tabla 2: Requisitos funcionales del sistema.....</i>                    | 21 |
| <i>Tabla 3: Requisitos no funcionales del sistema.....</i>                 | 22 |
| <i>Tabla 4: Historia de usuario 1.....</i>                                 | 23 |
| <i>Tabla 5: Historia de usuario 2.....</i>                                 | 24 |
| <i>Tabla 6: Historia de usuario 3.....</i>                                 | 25 |
| <i>Tabla 7:Historia de usuario 4.....</i>                                  | 26 |
| <i>Tabla 8: Historia de usuario 5.....</i>                                 | 27 |
| <i>Tabla 9:Clasificación de los patrones GOF.....</i>                      | 33 |

**ÍNDICE DE FIGURAS**

|   |    |
|---|----|
| <i>Ilustración 1: Diagrama de dominio del sistema.....</i>                            | 16 |
| <i>Ilustración 2:Diagrama de clases y arquitectura de desarrollo del sistema.....</i> | 29 |
| <i>Ilustración 3: Ejemplo de patrón experto.....</i>                                  | 30 |
| <i>Ilustración 4: Ejemplo de patrón creador.....</i>                                  | 31 |
| <i>Ilustración 5:Ejemplo del patrón bajo acoplamiento.....</i>                        | 32 |
| <i>Ilustración 6:Ejemplo del patrón de alta cohesión.....</i>                         | 32 |
| <i>Ilustración 7:Ejemplo de patrón controlador.....</i>                               | 33 |
| <i>Ilustración 8:Ejemplo de Singleton.....</i>  | 34 |
| <i>Ilustración 9:Ejemplo de Decorador.....</i>  | 35 |
| <i>Ilustración 10: Diagrama Entidad Relación.....</i>                                 | 35 |
| <i>Ilustración 11:Diagrama de despliegue del sistema (elaboración propia).....</i>    | 38 |
| <i>Ilustración 12: Captura de pantalla de la plataforma SonarQube.....</i>            | 40 |
| <i>Ilustración 13: Captura de pantalla de la plataforma SonarQube.....</i>            | 40 |
| <i>Ilustración 14:Captura de pantalla del sistema (Autenticar Usuario).....</i>       | 42 |
| <i>Ilustración 15:Captura de pantalla del sistema (Generar reportes).....</i>         | 43 |

**OPINIÓN DEL(OS) TUTOR(ES)**

<Contenido de la opinión de los tutores>

**AVAL DEL CLIENTE**

<Contenido del aval del cliente sobre la solución desarrollada>

# INTRODUCCIÓN

Las empresas en el mundo, hoy manejan un cúmulo de información alto y varios procesos a la misma vez. “La información es necesaria e importante para cualquier proceso que se realice, pues permite que se evite cometer errores, juzgar a los demás, sufrir un accidente, entre otras. Con una información actualizada es posible tener un impacto en las relaciones o la toma de decisiones de una persona, un grupo de estos o de una empresa” (EUROINNOVA, 2022).

Teniendo en cuenta toda la información que se acumula, cada día es más necesario obtener una estadística oportuna para la toma de decisiones, una manera de realizar este proceso es a partir de la generación de reportes con los datos que se utilizan. En el campo de la informática, “los reportes son informes que organizan y exhiben la información contenida en una base de datos. Su función es aplicar un formato determinado a los datos para mostrarlos por medio de un diseño atractivo y que sea fácil de interpretar por los usuarios” (Julián, et al., 2021).

Los reportes no sólo permiten conocer el estado de las empresas, sino que también ayudan a tomar mejores decisiones, e incluso cambiar el plan de trabajo si se requiere. “Todo sistema de gestión empresarial debe contar con funciones que faciliten disponer de los reportes en las compañías, en el momento que sean necesarios. Por este motivo los reportes juegan un papel fundamental, ya que con ellos se mide el rendimiento eficiente de los procesos operativos previamente establecidos o requieren de cambios para encaminar a la compañía a un crecimiento óptimo” (InterFuerza, 2021).

En la actualidad con el desarrollo del software, la generación de reporte se ha logrado automatizar a partir de sistemas de gestión. Este tipo de sistema se conoce como “una herramienta que permite controlar, planificar, organizar, y hasta cierto punto, automatizar las tareas de una empresa. Su objetivo es unificar en un único software todas las operaciones de la compañía con el fin de facilitar la toma de decisiones y el análisis de los datos. En los últimos años, los sistemas de gestión han ido ganando popularidad como herramienta fundamental para las organizaciones gracias a sus prestaciones cada vez más compleja, especialmente en un proceso de transformación digital como el que se está viviendo actualmente” (Ekon, 2021).

El desarrollo de los sistemas de gestión se ha visto potenciado por el avance significativo de las TIC, debido a que se han creado nuevos canales de información accesibles tanto a las personas como a las organizaciones. Las nuevas tecnologías permiten a una empresa competir y obtener la información disponible en el lugar y momento en el que se necesite por lo que éstas se han

## Introducción

convertido en arma esencial para ofrecer productos y servicios con mayor calidad” (Yohannia López Vargas, 2016).

En Cuba existen varias entidades desarrolladoras de software, enfocadas a resolver a través de la informatización, problemas de la sociedad actual; la Universidad de las Ciencias Informáticas (UCI) se destaca, al contar con una estructura de centros de producción de software. Entre ellos se encuentra el Centro de Soporte Tecnológico, dedicado al mantenimiento de los productos que son desarrollados por la Red de Centros de la universidad. Este centro tiene como misión: “Brindar el servicio de soporte técnico a las aplicaciones y servicios informáticos desarrollados por la UCI con calidad y eficiencia a partir de una correcta gestión de los mismos” [CITATION Cen \l 3082 ]. Entre los servicios que brinda el centro se encuentra el reporte y resolución de incidencias.

La gestión de incidencias es un proceso representado en la metodología de gestión de servicios ITIL<sup>1</sup>, que guía el desarrollo de las actividades del Centro de Soporte en la UCI. Las incidencias se asocian a solicitudes de información, problemas o peticiones de cambio que realizan los clientes contratados, y son registradas, mediante tickets, en la plataforma de gestión Zammad. La plataforma permite el monitoreo, seguimiento y control de los tickets y cuenta con un módulo de reportes para el análisis.

Mensualmente el centro emite un informe de conciliación con el cliente, que refleja las incidencias trabajadas en un período específico, habitualmente comprendido entre tres y seis meses. Además, se envían con frecuencia los reportes con el estado de sus sistemas. Estos análisis se realizan mediante la generación de reportes en Zammad, utilizando filtros asociados que permitan acotar la información necesaria.

El número de filtros que posee el módulo de reportes de Zammad, es limitado. Solo permite para la exportación el formato Excel, los nombres de los campos que reflejan, se muestran siempre en idioma inglés y no incluye la descripción de las incidencias. Por otra parte, no es posible modificar el formato de las fechas, ni seleccionar que campos se desea visibilizar.

Otra desventaja es que el módulo no incluye gráficos para resumir la situación de las incidencias según clasificaciones y necesidades, tanto del centro como del cliente. Además, no es dinámico, y la inclusión de nuevos filtros, funcionalidades o utilidades debe ser siempre desde el código fuente.

A partir de la situación antes descrita se ha identificado el siguiente **problema a resolver**:

<sup>1</sup>Las siglas ITIL significan Information Technology Infrastructure Library, se traduce como Biblioteca de Infraestructura de Tecnologías de Información.

## Introducción

¿Cómo mejorar la gestión de reportes de incidencias dinámicos en el centro de soporte de la Universidad de las Ciencias Informáticas?

Se define como **objeto estudio** de la presente investigación: Sistema de gestión de reportes dinámicos, enmarcando como **campo de acción**: Sistema de gestión de reportes de incidencias dinámicos en los servicios de soporte.

Para la presente investigación se define como **objetivo general**: Desarrollar un sistema de gestión de reportes de incidencias dinámicos que permita administrar las incidencias de los servicios de soporte de la Universidad de las Ciencias Informáticas.

Para dar cumplimiento al objetivo trazado y guiar la presente investigación se definen las siguientes **tareas de la investigación**:

1. Estudio de los antecedentes de la situación problemática en el centro de soporte para diagnosticar las necesidades existentes.
2. Investigación de los conceptos asociados a la gestión de reportes dinámicos, las incidencias y los servicios de soporte.
3. Elaboración del marco teórico de la investigación relacionado con los elementos asociados al objeto de estudio y el campo de acción.
4. Elaboración del estado del arte asociado a las herramientas de generación de reportes.
5. Investigación y análisis de las herramientas y tecnologías de desarrollo a utilizar en la implementación del sistema propuesto.
6. Diseño del sistema Gestión de reportes de incidencias para los servicios de soporte, según la metodología definida.
7. Implementación de la herramienta propuesta, utilizando las tecnologías de desarrollo seleccionadas.
8. Realización de las pruebas de software necesarias para verificar el cumplimiento del objetivo planteado.

**Para desarrollar el presente trabajo de diploma se propone utilizar los siguientes Métodos de investigación científica:**

**Métodos científicos a emplear:**

### **Métodos Teóricos**

**Analítico-sintético:** Este método se aplicó para organizar el objeto de estudio, extrayendo los elementos esenciales que debían ser abordados en la investigación. Para ello se formularon los conceptos y definiciones que permitieron su comprensión y la búsqueda de la solución al problema existente. Además, se realizó la síntesis de los elementos necesarios para la utilización de las tecnologías y metodologías adecuadas para el desarrollo del sistema propuesto.

### **Métodos Empíricos:**

**Observación:** Este método se aplicó para determinar los problemas que existían con la gestión de reportes y el alcance de la problemática. Permitted establecer las actividades necesarias y el funcionamiento del proceso para la construcción del sistema informático. Ver guía de Observación ([Anexo #4](#)).

**Entrevista:** Este método se aplicó para determinar qué elementos debía poseer el sistema propuesto como solución. Ver entrevista en los ([Anexo #5](#)).

El contenido del presente trabajo está distribuido, en resumen, introducción y tres capítulos y conclusiones generales. A continuación, se explica la información que se recoge en cada uno de ellos:

### **Capítulo 1: Fundamentación teórica del Sistema Gestión de Reportes de Incidencias para los servicios de soportes en la Universidad de las Ciencias Informáticas.**

En este capítulo se realiza la elaboración del marco teórico donde se exponen los conceptos asociados a la solución de la problemática y las diferentes soluciones existentes a nivel mundial. También se describen y caracterizan la metodología, herramientas y tecnologías a utilizar para el desarrollo del sistema de gestión.

### **Capítulo 2: Análisis, diseño e implementación del Sistema Gestión de Reportes de Incidencias para los servicios de soportes en la Universidad de las Ciencias Informáticas.**

Se especifica el Modelo Conceptual para identificar los principales conceptos del negocio. Se elabora los artefactos generados por la metodología seleccionada. Se modelan y detallan los diagramas que representan las funcionalidades del sistema, a partir de los patrones de diseño identificados.

### **Capítulo 3: Pruebas realizadas al Sistema Gestión de Reportes de Incidencias para los servicios de soportes en la Universidad de las Ciencias Informáticas.**

## Introducción

Se muestra la distribución física de los distintos componentes lógicos desarrollados, a través del modelo de despliegue y la organización del sistema mediante el modelo de componentes. Se explican los estilos de programación y estándares de codificación empleados y por último se valida el sistema desarrollado aplicándole las pruebas necesarias para demostrar que la solución es correcta.



# **CAPÍTULO I: FUNDAMENTOS Y REFERENTES TEÓRICO-METODOLÓGICOS DEL SISTEMA GESTIÓN DE REPORTES DE INCIDENCIAS PARA LOS SERVICIOS DE SOPORTES EN LA UNIVERSIDAD DE LAS CIENCIAS INFORMÁTICAS.**

## **1.1 Introducción**

En el presente capítulo se abordan las principales definiciones y conceptos asociados a la investigación, obteniéndose la información necesaria para la comprensión del tema, relacionado con la solución que se propone. Además, se realizó un estudio de varios sistemas que forman parte del dominio de la investigación, buscando una guía para los autores en el desarrollo del resultado y la comprensión del tema específico. De esta forma se puntualizarán las metodologías, así como las tecnologías que se utilizaron en el avance del proyecto. De manera general se obtendrá una base teórica completa para poder iniciar el proceso de desarrollo de la solución propuesta.

## **1.2 Conceptos asociados**

Comenzando por el objeto de estudio y el campo de acción de la investigación, y apoyando la comprensión del objetivo de la investigación, es fundamental definir tres conceptos importantes, como son: sistema, incidencias y servicios de soporte.

### **Sistema:**

Sistema es “un conjunto de elementos organizados que se encuentran en interacción, que buscan alguna meta o metas comunes, operando para ello sobre datos, información, sobre energía, materia u organismos, en una referencia temporal para producir como salida, información, energía, u organismos. Un sistema puede ser además un agregado de personas, cosas e información agrupados en conjunto de acuerdo con un objetivo”. [CITATION EUR21 \l 1033 ]

Otros de sus conceptos, es que sistema “es una serie de elementos que están interconectados entre sí y cuyo funcionamiento es como un todo. Los elementos que lo conforman pueden ser diversos, como un grupo de lineamientos o principios perfectamente estructurados acerca de una teoría, ciencia o materia” [ CITATION Mar22 \l 3082 ].

Para la presente investigación se toma que un sistema es un conjunto de elementos organizados que se encuentran interrelacionados operando para ellos sobre datos e información, que genera una referencia temporal a producir como salida, en este caso los reportes.

### **Incidencias:**

Según Pedro González “la incidencia es un acontecimiento que pasa muy rápidamente en un negocio y tendrá consecuencias en el mismo. Una buena gestión de incidencias es de vital importancia para todas las empresas, ya que su cometido es solucionar cualquier problema que ocurra en una empresa de manera rápida y eficaz”[ CITATION Ped21 \l 1033 ].

Según la medicina, “incidencia no es más que la cantidad de casos nuevos de una enfermedad, un síntoma, muerte o lesión que se presenta durante un período de tiempo específico, como un año. La incidencia muestra la probabilidad de que una persona de una cierta población resulte afectada por dicha enfermedad”[ CITATION Med21 \l 1033 ].

Para la investigación el concepto más apropiado de incidencia es que, incidencia no es más que “una interrupción no planificada de un servicio de la tecnología de la información (TI) o una reducción de la calidad de un servicio de TI, es decir, un incidente es cualquier interrupción de servicios de Tecnología de la Información que afecta desde un solo usuario hasta toda la empresa”[ CITATION Sil21 \l 1033 ].

### **Servicios de Soporte:**

El soporte técnico es el “servicio que provee una empresa a sus clientes, con el fin de que reciban ayuda acerca del uso de sus productos, ya sean físicos o digitales. Puede estar proporcionado directamente por la compañía o estar a manos de agentes especializados externos”. [CITATION Hub21 \l 1033 ]

Los autores de la investigación concuerdan con que el concepto más adecuado es que, los servicios de soporte no son más que un personal especializado que se encarga de solucionar averías de hardware o software de PC y sistemas. Es responsable de la TI.

### **Reporte:**

“Un reporte es un [informe](#) o una [noticia](#). Este tipo de [documento](#) (que puede ser impreso, digital, audiovisual, etc.) pretende transmitir una [información](#), aunque puede tener diversos objetivos. Existen reportes divulgativos, persuasivos y de otros tipos”[ CITATION Pér22 \l 1033 ].

En materia de informática, “un reporte es un informe en el cual se exhibe y se organiza la información compilándose en una base de datos. Su objetivo principal es imponer un formato en específico a los datos para que sean mostrados a través de un diseño llamativo y atractivo, que al mismo tiempo sea de fácil interpretación por los usuarios”[ CITATION Equ22 \l 1033 ].

### 1.3 Soluciones existentes

En la actualidad, las nuevas tecnologías han avanzado a pasos agigantados, la prueba de ello es con los sistemas de Reportes de Incidencias. Ya que, “estos permiten hacer un seguimiento adecuado a los incidentes reportados por los clientes para tratarlos de forma eficiente y minimizar su impacto negativo”[ CITATION Dou21 \l 1033 ]. Implementar un software de gestión de reportes es una gran opción de productividad para las organizaciones: este permite prevenir o restaurar en el menor tiempo posible cualquier interrupción o retraso que afecte a la calidad del servicio (no planificada) y minimizar el impacto de las operaciones comerciales de cualquier empresa u organización.

Después de una amplia búsqueda e investigación de soluciones existentes, semejantes al problema de investigación, se encontraron algunos software y herramientas para el desarrollo de reportes como: Megalytic, Service Desk y Help Desk.

#### **Megalytic:**

“Megalytic es una herramienta de panel e informes de clientes utilizada por las agencias de marketing digital y las empresas de alojamiento de sitios web para conectar fácilmente sus datos y crear informes atractivos que les encantan a los clientes. Megalytic para Google Data Studio automatiza el trabajo tedioso y repetitivo de combinar datos de Facebook, LinkedIn, Microsoft y más, para que pueda dedicar más tiempo a entregar informes útiles a los clientes”[ CITATION MEG \l 1033 ].

Algunas de las ventajas y desventajas [ CITATION Sam20 \l 1033 ]:

#### Ventajas:

- La generación de informes enlatados es rápida y sencilla.
- La vinculación con Google Analytics es fácil. También puede enviar automáticamente correos electrónicos de varios tipos de informes a clientes directamente cada mes.

#### Desventajas:

- Las características son muy limitadas.
- Poco mejoramiento en las funciones del sistema.

- Los widgets<sup>2</sup> de informe son muy rígidos y no tienen mucho que pueda personalizar. Si se busca informes muy específicos y busca la perfección de los informes, encontrará muchas limitaciones con esta herramienta.

### **Service Desk:**

Una plataforma de Service Desk “permite gestionar un enorme volumen de incidentes y brindar respuestas de manera inmediata y eficiente a los clientes, la Mesa de Servicio o Service Desk engloba procesos más amplios, ya que permite planificar, estructurar y proveer una mayor variedad de servicios”[ CITATION Mar19 \l 1033 ].

Algunas ventajas de Service Desk:

- La plataforma de Service Desk permite simplificar el manejo de un gran volumen de incidencias en un mismo entorno de trabajo.
- Tiene como objetivo resolver los problemas de la forma más rápida posible.
- Usar una solución de soporte ágil y bien ordenada reduce los tiempos de capacitación y permite un proceso más automatizado de los nuevos agentes.

### **Help Desk:**

“Help desk se traduce del inglés como mesa de ayuda, y precisamente busca prestar soluciones integrales como si fuera la ventanilla donde irías a preguntar una duda en una oficina. El servicio soporte de nivel 1 también conocido como soporte help desk, soporte front-end o asistencia de primera línea de soporte proporciona información y apoyo relacionados con los productos y servicios de la empresa” [ CITATION Car22 \l 1033 ].

El servicio puede realizarse por el equipo internamente desde la compañía, pero también puede subcontratarse a proveedores externos o freelancers<sup>3</sup>. Los técnicos de soporte de nivel 1 tratan generalmente problemas de fácil resolución. “Estos pueden incluir problemas como la verificación de

---

<sup>2</sup>Esta es una herramienta que, apuntando a una aplicación o servicio del propio sistema operativo, muestra información interactiva en la pantalla para que dé un simple vistazo se puede conocer información o utilizar la aplicación de forma directa sin necesidad de abrir esta.

<sup>3</sup>Un freelancer es una persona que trabaja por cuenta propia, es decir sin una relación de dependencia laboral, y puede hacerlo para varios clientes que contratan sus servicios profesionales.

incidencias en las líneas de comunicación, resolución de problemas de usuario y contraseña, instalación/reinstalación básica de aplicaciones software, verificación de la configuración apropiada de hardware y software, y asistencia mediante la navegación de menús de aplicación”. [ CITATION Car22 \l 1033 ]

El estudio y análisis de las soluciones existentes, refleja que estos sistemas presentan peculiaridades que no se adaptan a los requerimientos del centro de soporte. A pesar de que se identificaron funcionalidades útiles, al ser privativos no pueden ser utilizados, siendo necesario el desarrollo de un nuevo software. Los elementos identificados servirán de base para el diseño e implementación de la nueva propuesta.

### **1.4 Metodología utilizada en la solución propuesta.**

Como metodología se denomina la serie de métodos y técnicas de rigor científico que se aplican sistemáticamente durante un proceso de investigación para alcanzar un resultado teóricamente válido. En este sentido, la metodología funciona como el soporte conceptual que rige la manera en que aplicamos los procedimientos en una investigación [ CITATION Coe22 \l 3082 ].

Creada por Scott Ambler, la metodología AUP es considerada una versión simplificada del Proceso Unificado Racional (RUP). AUP describe, de una manera simple y fácil de entender, la forma de desarrollar aplicaciones de software de negocio usando técnicas ágiles y conceptos que aún se mantienen válidos en RUP. Entre las técnicas ágiles que incluye se encuentra el Desarrollo Dirigido por Pruebas, el Modelado ágil, la Gestión de Cambios y la Refactorización de Base de Datos para mejorar la productividad. La estructura de esta metodología se organiza en cuatro fases y define 7 disciplinas, de ellas cuatro a los procesos ingenieriles y tres dedicados a la Gestión de Proyectos.

Al no existir una metodología de software universal, ya que toda metodología debe ser adaptada a las características de cada proyecto (equipo de desarrollo, recursos, etc.) exigiéndose así que el proceso sea configurable; decide hacer una variación de la metodología AUP, de forma tal que se adapte al ciclo de vida definido para la actividad productiva de la UCI.

Por tanto, la metodología utilizada como guía para el desarrollo de la presente investigación fue la AUP-UCI en el escenario 4 este es un escenario ágil, los clientes estarán disponibles y acompañando al desarrollador durante toda la etapa de educación de requisitos, diseño e implementación del sistema, se tiene bien definido el negocio, el proyecto no será extenso y están bien definidos los elemen-

tos a implementar, esta metodología es por la que se rigen los procesos productivos en el centro de soporte de tecnologías.

De las 4 fases que propone AUP (Inicio, Elaboración, Construcción, Transición) se decide para el ciclo de vida de los proyectos de la UCI mantener la fase de Inicio, pero modificando el objetivo de la misma, se unifican las restantes 3 fases de AUP en una sola, a la que llamaremos Ejecución y se agrega la fase de Cierre.

AUP propone 9 roles (Administrador de proyecto, Ingeniero de procesos, Desarrollador, Administrador de BD, Modelador ágil, Administrador de la configuración, Stakeholder, Administrador de pruebas, Probador), se decide para el ciclo de vida de los proyectos de la UCI tener 11 roles, manteniendo algunos de los propuestos por AUP y unificando o agregando otros. Con la adaptación de AUP que se propone para la actividad productiva de la UCI lograr estandarizar el proceso de desarrollo de software[CITATION Tam15 \l 1033 ].

### **1.5 Tecnologías y Herramientas asociadas al desarrollo del sistema.**

La selección de las herramientas y tecnologías en las que se apoya el desarrollo del sistema de gestión de reportes de incidencias para los servicios de soportes está en correspondencia con las solicitudes del cliente. Estas especificaciones se realizaron teniendo en cuenta la aplicación de ellas al contexto de trabajo del sistema, la vigencia en las comunidades de desarrollo y las facilidades funcionales que aportan al desarrollo del producto.

#### **1.5.1- Tecnologías**

##### **Base de datos PostgreSQL**

“[PostgreSQL](#), o simplemente Postgres para darle un nombre más pintoresco, es un sistema de código abierto de administración de bases de datos del tipo relacional, aunque también es posible ejecutar consultas que sean no relacionales. Dos detalles a destacar de PostgreSQL es que posee data types (tipos de datos) avanzados y permite ejecutar optimizaciones de rendimiento avanzadas, que son características que por lo general solo se ven en sistemas de bases de datos comerciales, como por ejemplo SQL Server de Microsoft u Oracle de la compañía homónima”[ CITATION San19 \l 3082 ].

##### **ReactJS versión 18.1.0**

“ReactJS es una librería JavaScript de código libre para construir interfaces web, desarrollada por Facebook. React junta HTML, CSS y JavaScript en un solo paquete llamado componente, para poder

reutilizar el código en diferentes lugares de tu aplicación. React tiene un virtual DOM para comparar con el DOM real e identificar únicamente las partes que deben cambiar en pantalla y así optimizar el rendimiento, además usa JSX para escribir los componentes en JavaScript, pero con la misma sintaxis de HTML”[CITATION EDc22 \l 1033 ].

### **NestJS versión 8.4.2**

“NestJS es un framework progresivo de NodeJS desarrollado en TypeScript diseñado para facilitar el desarrollo de aplicaciones backend, aportando a los programadores una buena estructura y metodología inicial. Aunque es compatible con TypeScript, NestJS también permite desarrollar aplicaciones en código JavaScript”[CITATION Epi22 \l 1033 ].

### **Material UI versión 5.10.5**

“Material UI es una biblioteca de componentes React de código abierto que implementa Material Design de Google. Incluye una colección integral de componentes pre construidos que están listos para usar en producción desde el primer momento. Material UI tiene un diseño hermoso y presenta un conjunto de opciones de personalización que facilitan la implementación de su propio sistema de diseño personalizado sobre nuestros componentes”[ CITATION Mat \l 1033 ].

### **Nginx versión 1.22.0**

“NGINX es un servidor web de código abierto, de alto rendimiento. Cuenta con una arquitectura avanzada basada en eventos EBA en sus siglas en inglés, o Arquitectura Basada en Eventos. Esta característica permite numerosas conexiones simultáneas, lo que proporciona más velocidad y escalabilidad. NGINX entrega el contenido estático del sitio web rápidamente, es fácil de configurar y tiene un bajo consumo de recursos. Debido a todas estas características, el servidor es utilizado por grandes compañías como Microsoft, IBM, Google, [WordPress.org](https://www.wordpress.org), entre otras”[ CITATION Hig20 \l 1033 ].

## **1.5.2- Tecnologías de desarrollo**

### **HTML5**

“HTML5 es un estándar que sirve como referencia del software que conecta con la elaboración de páginas web en sus diferentes versiones, define una estructura básica y un código (denominado HTML) para la definición de contenido de una página web, como texto, imágenes, vídeos, juegos, entre otros”.[ CITATION Jua19 \l 1033 ]

### **CSS**

## Capítulo 1

“CSS son las siglas en inglés para hojas de estilo en cascada (Cascading Style Sheets). Básicamente, es un lenguaje que maneja el diseño y presentación de las páginas web, es decir, cómo lucen cuando un usuario las visita. Funciona junto con el lenguaje HTML que se encarga del contenido básico de las páginas. Se les denomina hojas de estilo en cascada porque puedes tener varias hojas y una de ellas con las propiedades heredadas (o en cascada) de otras”[ CITATION Die21 \l 1033 ].

### **JavaScript**

“JavaScript es el lenguaje de programación que debes usar para añadir características interactivas a tu sitio web, (por ejemplo, juegos, eventos que ocurren cuando los botones son presionados o los datos son introducidos en los formularios, efectos de estilo dinámicos, animación, y mucho más). [JavaScript](#) es un robusto lenguaje de programación que se puede aplicar a un documento [HTML](#) y usarse para crear interactividad dinámica en los sitios web”[ CITATION MDN22 \l 1033 ].

### **1.5.3- Herramientas de desarrollo:**

Una de las herramientas que desempeña un papel importante en el desarrollo de soluciones informáticas son los Entornos de Desarrollo Integrado (IDE). Estos ofrecen facilidades al equipo de desarrollo cuando se implementan las aplicaciones debido a que permite la identificación de errores comunes que se comenten a diario.

### **Visual Studio Code versión 1.68.0**

“Visual Estudio Code es una versión reducida del entorno de desarrollo oficial de Microsoft centrado exclusivamente en el editor de código. Es multiplataforma y soporta la sintaxis de una gran cantidad de lenguajes de programación. La herramienta proporciona soporte y asistencia a lenguajes de diversos ámbitos: HTML, CSS, JavaScript, diversas variantes de C, Java, SQL, PHP, Ruby, Visual Basic o JSON entre otros, soportando resultado, snippets y autocompletado”[ CITATION Mic22 \l 1033 ].

### **PgAdmin4**

“PgAdmin 4 es la principal herramienta de gestión de código abierto para Postgres. PgAdmin está diseñado para monitorear y administrar múltiples servidores de bases de datos PostgreSQL, tanto locales como remotos, a través de una sola interfaz gráfica que permite la fácil creación y administración de objetos de bases de datos, así como una serie de otras herramientas para administrar sus bases de datos”[CITATION Yog22 \l 3082 ].

### **Visual Paradigm 8.0**



## Capítulo 1

“Visual Paradigm es una herramienta de diseño UML y herramienta CASE del inglés (Computer Aided Software Engineering, Ingeniería de Software Asistida por Computadora) diseñada para ayudar al desarrollo de software. Soporta los principales estándares de la industria tales como el Lenguaje de Modelado Unificado (UML), y la notación de Modelado de Procesos de Negocio (BPMN). Ofrece un completo conjunto de herramientas de equipos de desarrollo de software necesario para la captura de requisitos, planificación de software, planificación de controles, modelado de clases y modelado de datos. Además, permite modelar todos los tipos de diagramas de clases, código inverso y generar código desde los diagramas”[ CITATION 1li22 \l 1033 ].

### **Conclusiones del capítulo:**

A partir del análisis teórico de la investigación fue posible comprender el objetivo planteado, teniendo como partida el campo de acción donde se aplica. El estudio de los conceptos asociados permitió comprender términos usualmente tratados, facilitó la búsqueda bibliográfica y aportó vocablos concretos que acercaran al autor a la temática. De esta manera puede decirse que el sistema de gestión de reportes de incidencias para los servicios de soportes, se define como: Un grupo de elementos interrelacionados que utiliza el centro de soporte para gestión y la administración de las incidencias, con el objetivo de desarrollar un sistema de gestión de reportes dinámicos que permita seleccionar y mostrar las incidencias de los servicios de soporte.

El análisis de distintos sistemas permitió reafirmar que es necesario continuar con el desarrollo de una nueva solución, pues Megalytic y Help Desk tienen características muy limitadas y tratan problemas de fácil resolución. Por su parte Service Desk si cumple con lo establecido por el cliente, pero al ser privativo no se puede utilizar.

Como parte del análisis de las tecnologías de desarrollo fueron usadas las definidas por el cliente, las cuales los autores pudieron corroborar su utilidad, las facilidades que aportan al desarrollo, además de ser las más usadas en el mercado como HTML5, CSS, entre otras. Finalmente, para guiar el desarrollo de la presente investigación se utilizó la metodología AUP en su variación UCI por su adecuación a los procesos productivos de la universidad. Todo esto permitió un desarrollo sólido para obtener un Sistema Gestión de Reportes de Incidencias para los servicios de soportes que se ajuste a las necesidades del centro.

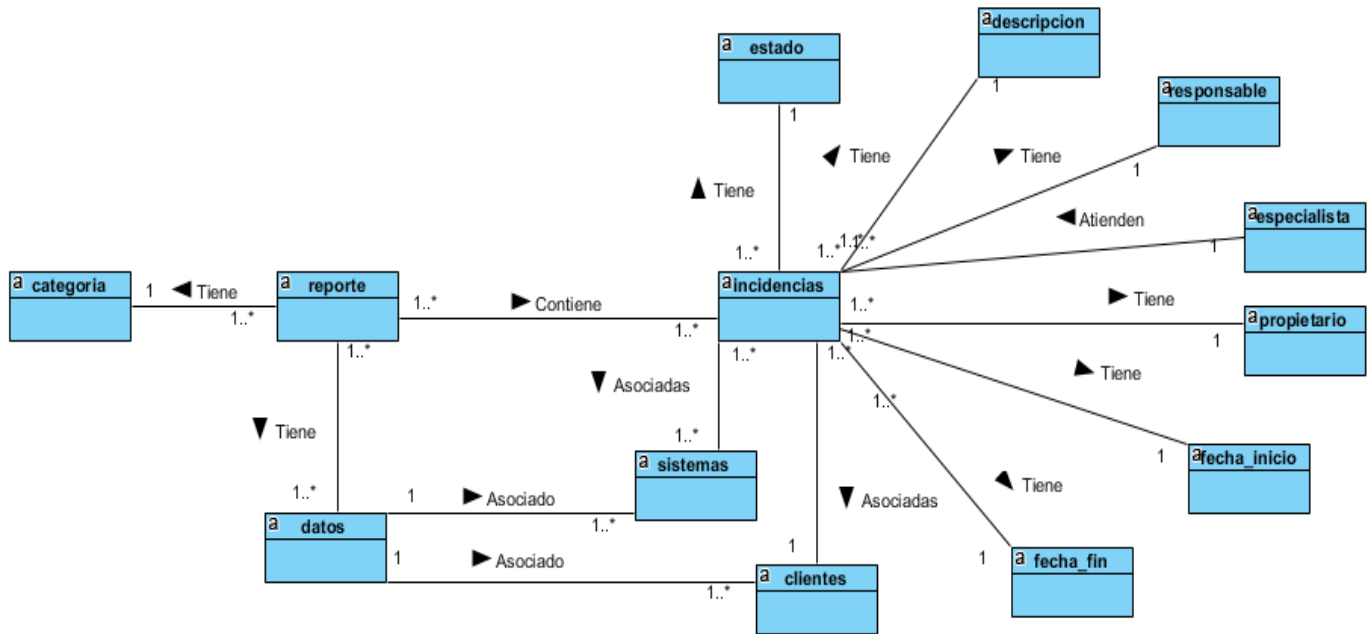
## **CAPÍTULO II: ANÁLISIS, DISEÑO E IMPLEMENTACIÓN DEL SISTEMA DE GESTIÓN DE REPORTES DE INCIDENCIAS PARA LOS SERVICIOS DE SOPORTES.**

### **2.1 Introducción**

El presente capítulo abarcará en su contenido, la información relevante obtenida mediante el proceso de negocio, como fase inicial de la ingeniería de software, bajo la cual se organiza la solución propuesta. De acuerdo a la metodología AUP en su variación UCI y utilizando prácticas ágiles en consecuencia con la solución propuesta, la descripción de los requisitos funcionales y no funcionales se realiza mediante la lista de reserva del producto. Los requisitos funcionales son ampliados mediante las historias de usuario, como funcionalidades del sistema, siendo organizados en la línea de tiempo del producto. Se diseñan las clases y relaciones más importantes del software y se identifican los principales patrones de diseño mediante los cuales se garantiza la aplicación de buenas prácticas de programación.

### **2.2 Modelo del dominio**

Su utilidad radica en ser una forma de inspiración para el diseño del software, es entrada para muchos de los artefactos que se construyen en un proceso de software. “Un modelo de dominio muestra las clases conceptuales significativas en un dominio del problema, se centra en las abstracciones relevantes, vocabulario de él dominio e información del dominio. Es el artefacto clave del análisis orientado a objetos”[ CITATION Fra18 \l 3082 ].



**Ilustración 1: Diagrama de dominio del sistema**

Un reporte tiene unos datos asociados a sistemas y a clientes y contiene incidencias, que las incidencias están asociadas a sistemas y a clientes y las atienden los especialistas y que, además las incidencias contienen estado, descripción, propietario, responsable, especialista, fecha de inicio y fecha fin y los reportes además tienen categorías.

### 2.3 Solución Propuesta

Se propone crear un sistema de gestión de reportes de incidencias para los servicios de soporte en la Universidad de las Ciencias Informáticas teniendo como cliente al centro de soporte y como objetivo: la gestión de reportes dinámicos. La propuesta de solución usará la arquitectura cliente-servidor, en el servidor se encontrará una API<sup>4</sup> que es la encargada de la conexión con la base de datos de Zammad, permitiendo así intercambiar información entre la plataforma y el sistema de reporte. El cliente requerirá autenticación para trabajar con los reportes.

De esta forma el sistema va a tener un procedimiento, que solo permitirá a los trabajadores administrativos del centro autenticarse; una vez autenticado, el usuario podrá generar reportes a partir de las incidencias registradas en la plataforma. Para facilitar el proceso se muestra un listado histórico de las incidencias con las opciones de filtrado por los campos asociados a estas,

<sup>4</sup>Las API (interfaz de programación de aplicaciones) son mecanismos que permiten a dos componentes de software comunicarse entre sí mediante un conjunto de definiciones y protocolos.

permitiendo seleccionar los campos que sean de interés para generar el reporte.

Al filtrar los datos y seleccionar las columnas a visualizar en el reporte, se debe presionar el botón GENERAR REPORTE, mostrándose una pantalla con el resumen de los filtros aplicados y las opciones de exportar, en formato PDF o Excel, o graficar, en gráficos de barras o de pastel. Por otra parte, el usuario puede guardar el reporte para futuras consultas de información, en caso de no guardarlo, el reporte no quedara registrado en el sistema. En caso de guardar el reporte, podrá ser consultado en un listado de reportes guardados, donde podrá modificar los filtros asociados a ese reporte, consultar sus datos o eliminarlo.

Dentro de las funcionalidades se encuentra Crear Tarea, que permite generar tareas automáticas para el envío de reportes mediante correo electrónico, cada cierto periodo de tiempo, por ejemplo, mensual, trimestral o anual. Cuando se crea la tarea se selecciona el tipo de reporte, el sistema, la entidad y el cliente al cual se le enviará el correo electrónico con el reporte correspondiente. Una vez creada el sistema verifica los tiempos de entrega del periodo seleccionado y automáticamente hace el envío del reporte al cliente. El histórico de los reportes enviados podrá consultarse en el listado de correos, donde aparecerá el nombre y la fecha en la que se realizó el envío.

También está disponible la función gestionar usuario, donde se podrán crear los usuarios con acceso al sistema, que tendrán un rol asignado y en dependencia de este serán las funcionalidades que se mostrarán.

## **2.4 Concepción del sistema**

La estructura, el diseño y las funcionalidades que debe ejecutar el sistema para alcanzar su objetivo, son vitales para entender el curso de la investigación. Es asociado a esta idea que se hace preciso la gestión documental de la concepción del sistema, mostrando los principales detalles del producto de software que se propone como solución. Esto permitirá un mayor entendimiento del proyecto, no solo para su desarrollo, sino también para el cliente y la creación de futuras versiones.

### **2.4.1 Visión y alcance del sistema.**

El sistema de gestión de reportes de incidencias tendrá como objetivos fundamentales:

- Organizar las incidencias de acuerdo a campos específicos.
- Visualizar la información de manera organizada.
- Obtener reportes personalizados.
- Mejorar la distribución de los servicios del centro de soporte.

- Graficar los reportes utilizando gráficas de barra o pastel.
- Exportar los reportes en formato excel o pdf.

Para el cumplimiento del sistema se implementan funcionalidades óptimas desde el punto de vista del código fuente, que permitan graficar y exportar los reportes.

### 2.4.2 Planificación del proyecto por roles.

Para cumplir con el objetivo de la solución en un período de tiempo establecido es preciso determinar y organizar el trabajo guiado por la metodología de software elegida. Además, es importante definir quien realizará cada tarea y darle un nivel de importancia a cada una, pues de su cumplimiento en tiempo depende el éxito del producto final. Para el desarrollo de la solución y acorde a las necesidades del sistema a implementar, siguiendo las líneas de la metodología de software AUP, se definen los siguientes roles:

- **Jefe de proyecto:** Es la persona que representa al equipo de desarrollo. Sirve de guía y organizador durante el proceso de construcción del software. Debe dominar correctamente lo establecido en el proceso de negocio y conocer las tecnologías que se utilizan.
- **Analista:** Es el encargado de identificar las necesidades del cliente, para definir un diseño previo del software. Los analistas deben determinar si cada uno de los requisitos especificados son o no esenciales y, además, determinar de ser necesario información adicional requerida. Debe considerar todos los recursos especiales requeridos, las estimaciones del cliente y sus tiempos límites, así como factores extras de desarrollo que puedan ser de interés. Es el encargado junto al cliente, de identificar los requisitos funcionales y no funcionales del sistema. El éxito del proyecto dependerá de una buena especificación de requisitos, los cuales permitirán conformar una arquitectura idónea que evite fallas en la estructura del proyecto y pueda dar origen al colapso del mismo en forma parcial o total.
- **Desarrollador:** Por medio del mismo es posible que las especificaciones del sistema se conviertan en código fuente ejecutable. El éxito del software depende en gran medida del conocimiento del lenguaje de programación seleccionado para el desarrollo, una buena definición de requisitos y un correcto entendimiento entre el analista y programador.
- **Administrador de Base de Datos:** Es la persona que administra las tecnologías de la información relacionadas a los modelos de datos, para ser almacenados en Bases de datos. Provee al desarrollador con un modelo de datos de calidad para una correcta implementación

del sistema y garantiza el diseño, implementación y mantenimiento del sistema de base de datos. Así como el establecimiento de políticas y procedimientos relativos a la gestión, la seguridad, el mantenimiento y el uso del sistema de gestión de base de datos.

- **Stakeholder(cliente):** En la metodología AUP-UCI el cliente forma parte del equipo de desarrollo. En ese caso este rol es responsable de, junto al analista, establecer los requisitos del sistema. Acompaña a cada rol dentro del equipo de desarrollo en todas las fases del software, apoyando el trabajo que se realiza y dando su criterio acerca de los resultados que se obtienen en cada etapa. Participa en el diseño y ejecución de las pruebas de software.
- **Probador:** La posibilidad de que aparezcan errores humanos durante el proceso de desarrollo es el motivo por el cual existe este rol. Por ejemplo, los requisitos del sistema pueden ser especificados en forma errónea o imperfecta, y es por medio de este que se detectan y solucionan estas no conformidades. Es por ello que junto al cliente o al programador, según el tipo de prueba que se realiza al sistema, debe diseñar los casos de pruebas y aplicárselo al software. Es responsable de lograr los resultados correctos en la fase de prueba.

En la siguiente tabla se pueden observar los roles definidos por personas para el desarrollo del sistema de gestión de reportes de incidencias para los servicios de soporte para la Universidad de las Ciencias Informáticas.

**Tabla 1:Roles y responsabilidades en el desarrollo del sistema**

| Rol                     | Responsabilidad  | Nombre                  |
|-------------------------|--|-------------------------|
| <b>Jefe de Proyecto</b> | <ul style="list-style-type: none"> <li>• Representar al equipo de desarrollo.</li> <li>• Guiar y organizar el proceso de desarrollo del software.</li> <li>• Conocer el proceso de negocio establecido.</li> <li>• Conocer las tecnologías a utilizar.</li> </ul>                          | Neybis Lago Clara       |
| <b>Analista</b>         | <ul style="list-style-type: none"> <li>• Es el encargado junto al cliente, de identificar los requisitos funcionales y no funcionales del sistema.</li> <li>• Generar apoyo documental.</li> <li>• Diseñar los diagramas del diseño.</li> <li>• Realizar el proceso de negocio.</li> </ul> | Yaneysi Velázquez Silva |

|  |  |  |
|--|--|--|
| <b>Desarrollador</b>                   | <ul style="list-style-type: none"> <li>• Conocer las tecnologías a utilizar.</li> <li>• Implementar las funcionalidades del sistema.</li> </ul>  | Yaneysi Velázquez<br>Silva   |
| <b>Administrador de bases de datos</b> | <ul style="list-style-type: none"> <li>• Elaborar el modelo de datos de la base de datos del sistema.</li> <li>• Crea las funcionalidades en la base de datos para un funcionamiento óptimo del sistema.</li> </ul>  | Yaneysi Velázquez<br>Silva   |
| <b>Stakeholder(cliente )</b>           | <ul style="list-style-type: none"> <li>• Establecer el proceso de negocio.</li> <li>• Definir los requisitos funcionales del sistema.</li> <li>• Apoyar y participar en cada fase del desarrollo del software.</li> <li>• Participar en el proceso de pruebas del software.</li> </ul> | Neybis Lago Clara<br>Víctor Cabrera Sanz                               |
| <b>Probador</b>                        | <ul style="list-style-type: none"> <li>• Diseñar casos de pruebas.</li> <li>• Ejecutar casos de pruebas.</li> </ul>  | Neybis Lago Clara<br>Víctor Cabrera Sanz<br>Yaneysi Velázquez<br>Silva |

## 2.5 Especificación de requisitos

La especificación de requisitos tiene como objetivo fundamental servir como medio de comunicación entre cliente, usuario, ingeniero de requisitos y desarrolladores. Estos deben recoger tanto las necesidades de clientes y usuarios como los requisitos que debe cumplir el sistema a desarrollar para satisfacer dichas necesidades.

### 2.5.1 Lista de reserva del producto.

La estimación para el costo de tiempo que implicará cada requisito estará dada en días, para un total aproximado de 65 días.

**Tabla 2: Requisitos funcionales del sistema.**

| Prioridad | Ítem* | Descripción   | Estimación | Estimado por |
|-----------|-------|---|------------|--------------|
| Media     | RF1   | Eliminar usuario.   | 2          | Analista     |
| Media     | RF2   | Modificar usuario.  | 2          | Analista     |
| Media     | RF3   | Insertar usuario.   | 2          | Analista     |
| Media     | RF4   | Listar usuario.   | 2          | Analista     |
| Alta      | RF5   | Graficar reporte.   | 5          | Analista     |
| Alta      | RF6   | Generar reporte.  | 4          | Analista     |
| Media     | RF7   | Exportar reporte a pdf.                                   | 2          | Analista     |
| Alta      | RF8   | Exportar reporte a Excel.                                 | 2          | Analista     |
| Media     | RF9   | Exportar gráfico como imagen.                             | 3          | Analista     |
| Alta      | RF10  | Generar contenedor de la aplicación.                      | 5          | Analista     |
| Media     | RF11  | Integrar el sistema de reportes con la plataforma Zammad. | 8          | Analista     |
| Alta      | RF12  | Insertar reporte.   | 3          | Analista     |
| Alta      | RF13  | Listar Reporte.   | 2          | Analista     |
| Alta      | RF14  | Eliminar reporte.   | 2          | Analista     |
| Alta      | RF15  | Modificar reporte.  | 4          | Analista     |
| Alta      | RF16  | Crear tipo de reporte.                                    | 4          | Analista     |
| Alta      | RF17  | Eliminar tipo de reporte.                                 | 4          | Analista     |
| Alta      | RF18  | Modificar tipo de reporte.                                | 4          | Analista     |
| Baja      | RF19  | Crear tareas programadas (Enviar reporte).                | 2          | Analista     |



|      |      |                                   |   |          |
|------|------|-----------------------------------|---|----------|
| Baja | RF20 | Enviar notificaciones por Correo. | 3 | Analista |
|------|------|-----------------------------------|---|----------|

**Tabla 3: Requisitos no funcionales del sistema.**

| Requisitos no funcionales |   |
|---------------------------|---|
| Usabilidad                | <ul style="list-style-type: none"> <li>- El sistema será utilizado por todo el personal autorizado.</li> <li>- La propuesta de solución debe brindar un acceso fácil y rápido.</li> </ul>   |
| Fiabilidad                | <ul style="list-style-type: none"> <li>-El sistema brinda la posibilidad de establecer permisos sobre acciones, garantizando que solo acceda quien esté autorizado.</li> <li>-El sistema muestra las funcionalidades de acuerdo a quien esté autenticado en el mismo.</li> <li>-El sistema debe garantizar la protección ante acciones no autorizadas.</li> </ul> |
| Rendimiento               | <ul style="list-style-type: none"> <li>- El sistema debe responder en un tiempo menor de los 10 segundos.</li> <li>- El sistema deberá soportar una conexión simultánea de al menos 10 usuarios.</li> </ul>   |
| Disponibilidad            | -Los usuarios deben tener acceso a la información desde cualquier dispositivo sin que los mecanismos utilizados para la seguridad de los datos retrasen la obtención de los mismos.   |
| Restricciones de diseño   | <ul style="list-style-type: none"> <li>- Lenguajes de desarrollo: ReactJS 18.1.0, CSS 3, HTML5.</li> <li>- El sistema gestor de base datos PostgreSQL 11 con PgAdmin4.</li> </ul>   |
| Software                  | <ul style="list-style-type: none"> <li>-Para el despliegue del sistema se debe contar en el servidor de bases de datos con PostgreSQL.</li> <li>-La comunicación entre el cliente y el servidor de aplicaciones se realiza a través del protocolo HTTPS.</li> </ul>   |
| Hardware                  | <ul style="list-style-type: none"> <li>- Monitor VGA o superior.</li> <li>-Para el cliente se recomienda como mínimo un dual Core a 2.8 GHZ con 2 GB de RAM.</li> </ul>   |

|                  |   |
|------------------|---|
| Confidencialidad | -Debe mantenerse la consistencia de los datos en correspondencia con la realidad, mediante la actualización frecuente con las fuentes correspondientes. |
| Portabilidad     | -El sistema debe ser multiplataforma.   |

## 2.5.2 Historias de usuario.

Luego de establecer los requisitos y estimar su tiempo de ejecución es necesario describirlos minuciosamente para facilitar así su desarrollo. Utilizando el escenario 4 de la metodología AUP-UCI las metodologías ágiles, generalmente, proponen las Historias de Usuarios (HU) como el mecanismo para gestionar los requisitos, aplicándose de manera similar en AUP UCI en su versión ágil. Las HU “se escriben desde la perspectiva del cliente, aunque los desarrolladores pueden brindar también su ayuda en la identificación de las mismas. El contenido de estas debe ser concreto y sencillo. El tratamiento de las HU es muy dinámico y flexible, en cualquier momento pueden romperse, reemplazarse por otras más específicas o generales, añadirse nuevas o ser modificadas. Cada HU es lo suficientemente comprensible y delimitada para que los programadores puedan implementarla en unas pocas semanas”[CITATION Wen \l 1033 ].

A continuación, se muestran cinco Historias de Usuario. El resto de ellas se encuentran en la sección de los Anexos ([ver Anexo#1](#)).

**Tabla 4: Historia de usuario 1**

|   |  |
|---|--|
|   |  |
| Número: HU_1  | Requisito: Integrar el sistema de reportes con la plataforma Zammad. |
| Programador: Yaneysi Velázquez Silva  | Iteración Asignada: 1  |
| Prioridad: Alta   | Tiempo Estimado: 2 días  |
| Riesgo en Desarrollo: Alto  | Tiempo Real: 0.4   |
| Descripción:<br>Para la conexión con el sistema Zammad, se debe generar un contenedor para la aplicación de reportes, la cual contendrá una API que es la que se va a conectar al otro contenedor del Zammad para obtener los datos.<br>Crear BD. |  |

Observaciones:

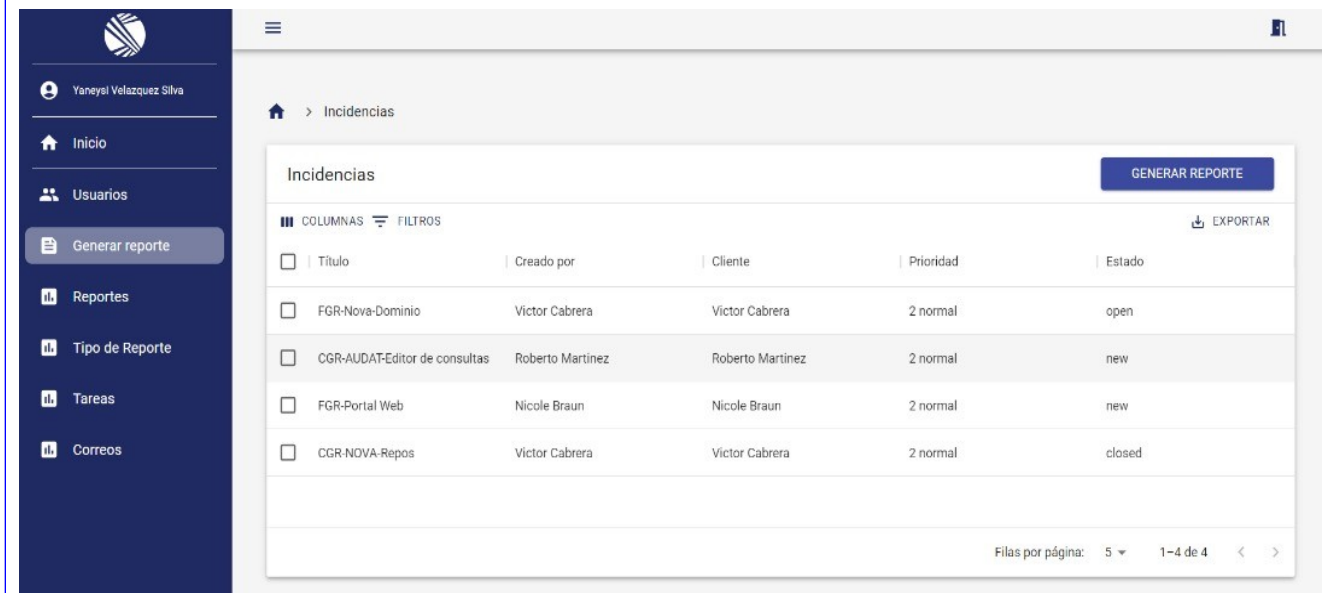
Es necesario que el contenedor de Zammad y el de la aplicación estén disponibles.

Prototipo elemental de interfaz gráfica de usuario:

**Tabla 5: Historia de usuario 2**

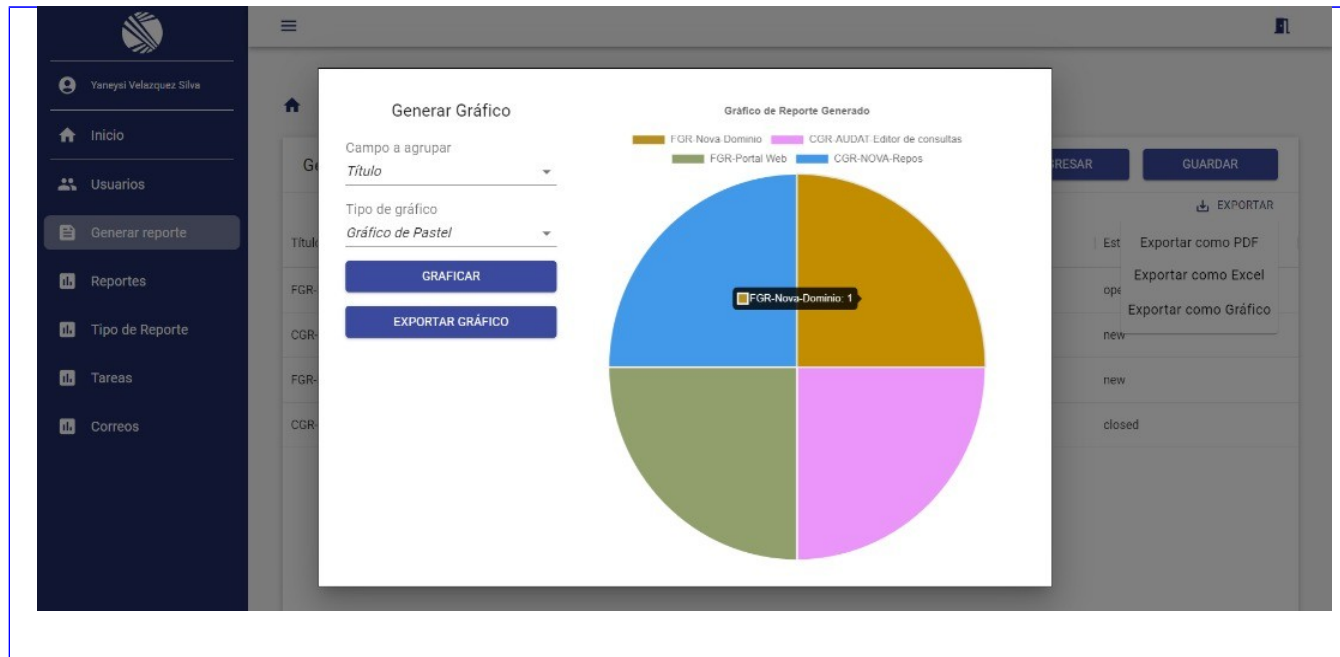
| Número: HU_6  | Requisito: Generar reporte. |
|---|-----------------------------|
| Programador: Yaneysi Velázquez Silva  | Iteración Asignada: 1       |
| Prioridad: Alta   | Tiempo Estimado: 4 días     |
| Riesgo en Desarrollo: Media   | Tiempo Real: 0.8            |
| <p>Descripción:</p> <p>Solo el administrador avanzado se le permite generar reporte. Permite insertar, listar, eliminar y modificar reporte.</p> <p>Da la posibilidad de realizar alguna de las siguientes acciones:</p> <p>Si decide insertar un reporte, se debe ir al listado de ticket, hacer un filtro y presionar botón “Generar Reporte”.</p> <p>Si decide modificar los datos de un reporte, ir al listado de reportes, presionar el botón de “Modificar” y se guarda el reporte modificado.</p> <p>Si decide listar reporte(s), ir a la sección “Listar reporte”.</p> <p>Si decide eliminar uno(s) reporte(s), ir a la sección “Eliminar reporte”.</p> |                             |
| <p>Observaciones:</p> <p>El administrador tiene que estar autenticado.</p>  |                             |

Prototipo elemental de interfaz gráfica de usuario:



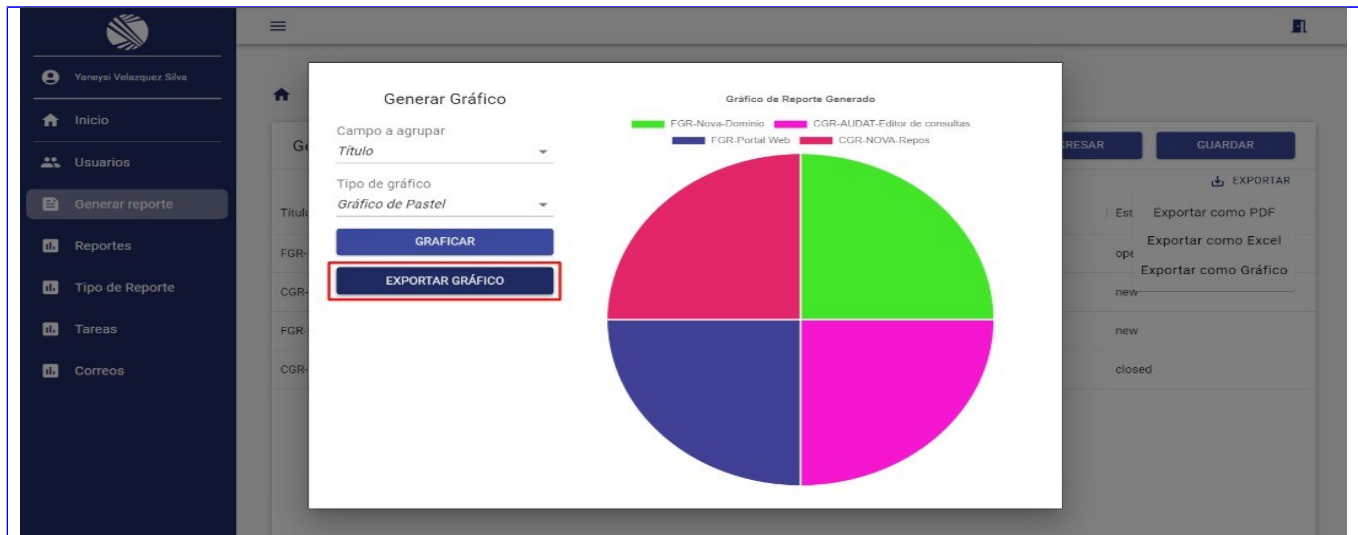
**Tabla 6: Historia de usuario 3**

|  |  |                              |
|--|--|------------------------------|
| Número: HU_5   |  | Requisito: Graficar Reportes |
| Programador: Yaneyssi Velázquez Silva                  |  | Iteración Asignada: 3        |
| Prioridad: Alta  |  | Tiempo Estimado: 2 días      |
| Riesgo en Desarrollo: Baja                             |  | Tiempo Real: 0.4             |
| Descripción:   |  |                              |
| Para graficar un reporte es necesario crearlo primero. |  |                              |
| Observaciones:   |  |                              |
| Para graficar se debe generar primero un reporte.      |  |                              |
| Prototipo elemental de interfaz gráfica de usuario:    |  |                              |



**Tabla 7: Historia de usuario 4**

|   |  |
|---|--|
|   |  |
| Número: HU_4  | Requisito: Seleccionar tipo de gráficos. |
| Programador: Yaneysi Velázquez Silva  | Iteración Asignada: 3                    |
| Prioridad: Baja   | Tiempo Estimado: 1 día                   |
| Riesgo en Desarrollo: Bajo  | Tiempo Real: 0.2                         |
| Descripción:<br>Se selecciona el tipo de gráfico con el que desea graficar. |  |
| Observaciones:<br>Prototipo elemental de interfaz gráfica de usuario:       |  |



**Tabla 8: Historia de usuario 5**

|  |                             |
|--|-----------------------------|
|  |                             |
| Número: HU_5   | Requisito: Insertar Usuario |
| Programador: Yaneyssi Velázquez Silva  | Iteración Asignada: 3       |
| Prioridad: Baja  | Tiempo Estimado: 2días      |
| Riesgo en Desarrollo: Baja   | Tiempo Real: 0.5            |
| Descripción:<br>Se permite insertar usuario que tendrá permiso para gestionar el sistema.    |                             |
| Observaciones:<br>El dispositivo por el cual se va a estar conectado tiene q tener conexión. |                             |

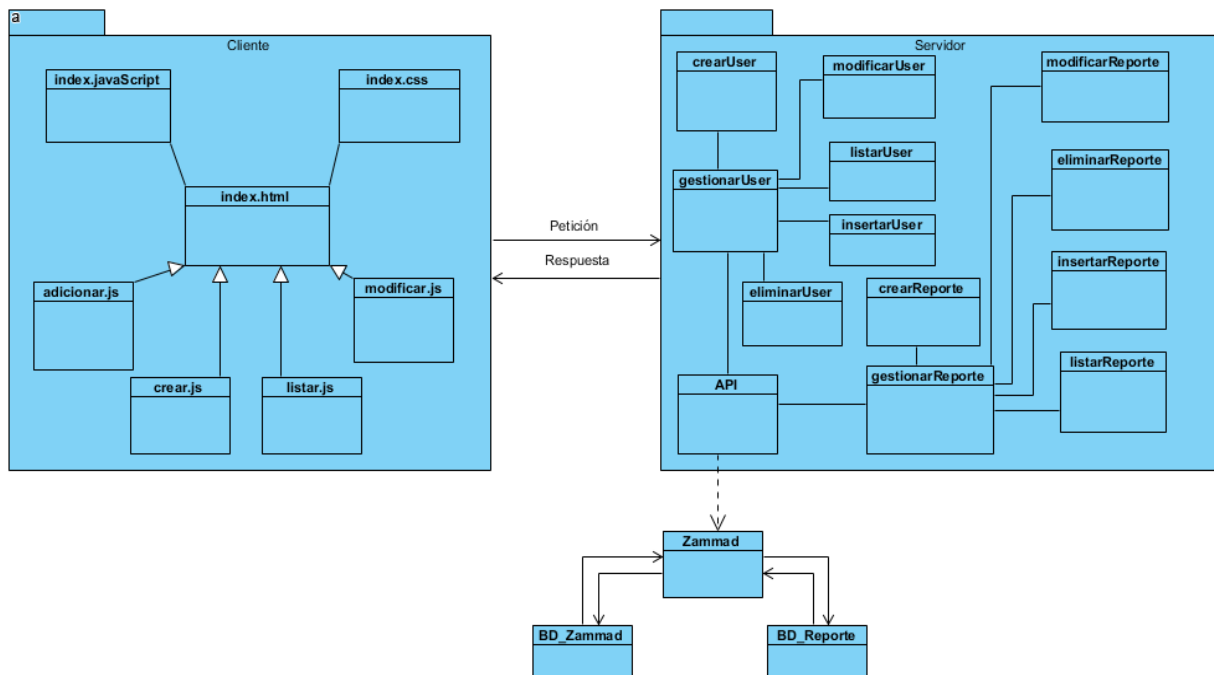
Prototipo elemental de interfaz gráfica de usuario:

The screenshot shows a web application interface. On the left is a dark blue sidebar with a logo at the top and a user profile for 'Yaneysi Velazquez Silva'. Below the profile are menu items: 'Inicio', 'Usuarios', 'Generar reporte', 'Reportes', 'Tipo de Reporte', 'Tareas', and 'Correos'. The main content area has a breadcrumb trail 'Inicio > Usuarios > Adicionar'. A modal window titled 'Crear Usuario' is open, containing a form with the following fields: 'Nombre', 'Correo', 'Contraseña', 'Confirmar contraseña', and a dropdown menu for 'Rol' with 'Usuario' selected. A dark blue 'ACEPTAR' button is located at the bottom right of the modal.

## 2.6 Arquitectura del sistema.

Al usar ReactJS como framework de desarrollo, se utiliza la arquitectura Cliente-Servidor, por ser sobre el cual se encuentra desarrollado el marco de trabajo.

Cliente-Servidor es uno de los estilos arquitectónicos distribuidos más conocidos, “el cual está compuesto por dos componentes, el proveedor y el consumidor. El proveedor es un servidor que brinda una serie de servicios o recursos los cuales son consumido por el Cliente. En una arquitectura Cliente-Servidor existe un servidor y múltiples clientes que se conectan al servidor para recuperar todos los recursos necesarios para funcionar, en este sentido, el cliente solo es una capa para representar los datos y se detonan acciones para modificar el estado del servidor, mientras que el servidor es el que hace todo el trabajo pesado”[ CITATION Osc20 \l 1033 ].



**Ilustración 2: Diagrama de clases y arquitectura de desarrollo del sistema**

## 2.7 Patrones del Diseño.

“Un patrón es una descripción de un problema y su solución que recibe un nombre y que puede emplearse en otros contextos; en teoría, indica la manera de utilizarlo en circunstancias diversas. Muchos patrones ofrecen orientación sobre cómo asignar las responsabilidades a los objetos ante determinada categoría de problemas. Los patrones intentan codificar el conocimiento, las expresiones y los principios ya existentes: cuanto más trillados y generalizados, tanto mejor. En consecuencia, los patrones GRASP no introducen ideas novedosas; son una mera codificación de los principios básicos más usados” (Lerman, 1999).

Los patrones del diseño son un conjunto de buenas prácticas que se evidencia en la implementación de los productos software de manera reiterada, e incluso inherentemente. Su utilización permite obtener un código fuente organizado, una comunicación entre clases bien delimitada y la asignación de responsabilidades a los objetos que pueden cumplir el objetivo necesario. Durante el desarrollo del sistema de reportes de incidencias para los servicios de soporte, se tuvieron en cuenta alguno de estos patrones, los cuales se enuncian a continuación.



## 2.7.1 Patrones GRASP.

“GRASP es el acrónimo de General Responsibility Assignment Software Patterns (patrones generales de software para asignar responsabilidades). El nombre se eligió para indicar la importancia de captar estos principios, si se quiere diseñar eficazmente un software. Los patrones no se proponen descubrir ni expresar nuevos principios de la ingeniería del software. Todo lo contrario: intentan codificar el conocimiento, las expresiones y los principios ya existentes. En consecuencia, los patrones GRASP no introducen ideas novedosas; son una mera codificación de los principios básicos más usados”[ CITATION Wen \l 1033 ].

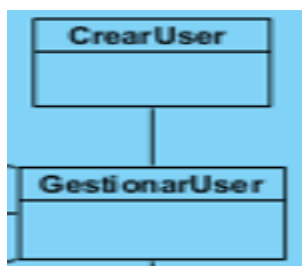
### Experto.

Solución: Asignar una responsabilidad a la clase que tiene la información necesaria para cumplirla.

Experto es un patrón que se usa más que cualquier otro al asignar responsabilidades; es un principio básico que suele utilizarse en el diseño orientado a objetos. Con él no se pretende designar una idea oscura ni extraña; expresa simplemente la “intuición” de que los objetos hacen cosas relacionadas con la información que poseen.

Beneficios: Se conserva el encapsulamiento, ya que los objetos se valen de su propia información para hacer lo que se les pide. Esto soporta un bajo acoplamiento, lo que favorece al hecho de tener sistemas más robustos y de fácil mantenimiento.

El comportamiento se distribuye entre las clases que cuentan con la información requerida, alentando con ello definiciones de clase “sencillas” y más cohesivas que son más fáciles de comprender y de mantener. Así se brinda soporte a una alta cohesión. Ejemplo de ello es la clase CrearUser.



**Ilustración 3: Ejemplo de patrón experto.**

Creador

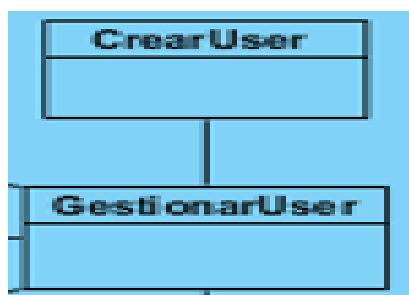
El patrón Creador guía la asignación de responsabilidades relacionadas con la creación de objetos, tarea muy frecuente en los sistemas orientados a objetos. El propósito fundamental de este patrón es encontrar un creador que debemos conectar con el objeto producido en cualquier evento. Al escogerlo como creador, se da soporte al bajo acoplamiento, lo cual supone menos dependencias respecto al mantenimiento y mejores oportunidades de reutilización. Es probable que el acoplamiento no aumente, pues la clase creada tiende a ser visible a la clase creador, debido a las asociaciones actuales que nos llevaron a elegirla como el parámetro adecuado.



**Ilustración 4: Ejemplo de patrón creador**

### Bajo Acoplamiento

Cuando se habla de acoplamiento entre objetos, se hace referencia a la "fuerza" con la que ciertos objetos están relacionados, o dependen unos de otros. Mientras más dependencias tenga un objeto de otros para llevar a cabo sus tareas, más fuerte será el acoplamiento. Cuando una clase de objetos puede realizar sus tareas, sin depender de ninguna otra clase de objetos (o de un número muy reducido de ellas) se dice que hay bajo acoplamiento. El grado de acoplamiento es importante debido a que está relacionado íntimamente con la reutilización de clases de objetos. Una clase que depende de muchas otras es menos reutilizable.

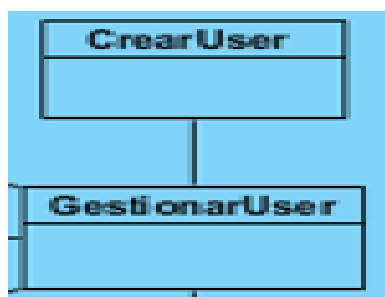


**Ilustración 5: Ejemplo del patrón bajo acoplamiento**

### Alta Cohesión

“En la perspectiva del diseño orientado a objetos, la cohesión (o, más exactamente, la cohesión funcional) es una medida de cuán relacionadas y enfocadas están las responsabilidades de una clase. Una alta cohesión caracteriza a las clases con responsabilidades estrechamente relacionadas que no realicen un trabajo enorme. Una clase con baja cohesión hace muchas cosas no afines o un trabajo excesivo” [CITATION Ler99 \ 1033 ].

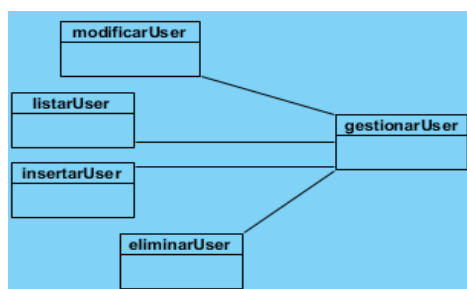
No convienen este tipo de clases pues son difíciles de comprender, de reutilizar, de conservar y afectan constantemente los cambios. En su mayoría las clases con baja cohesión son muy abstractas y delegan funcionalidades a objetos que no le corresponden.



**Ilustración 6: Ejemplo del patrón de alta cohesión**

### Controlador

El patrón "controlador" establece una clara separación entre la interfaz de usuario (una interfaz gráfica, por ejemplo) y el corazón o núcleo de procesamiento de la aplicación, donde se halla la lógica de negocios.



**Ilustración 7: Ejemplo de patrón controlador**

## 2.7.2 Patrones GOF

Los patrones GOF (Banda de los Cuatros) no son más que problemas recurrentes en el desarrollo de software que fueron clasificado y agrupados a partir de dos criterios, su propósito y alcance. Según

este grupo los patrones “describen soluciones simples y elegantes a problemas específicos en el diseño de software orientado a objetos” (Guerrero, y otros, 2013).

**Tabla 9: Clasificación de los patrones GOF**

| Categoría GOF     | Patrón de diseño   |
|-------------------|--|
| Creacionales      | <ul style="list-style-type: none"> <li>➤ Builder.</li> <li>➤ Factory Method.</li> <li>➤ Singleton</li> </ul>   |
| Estructurales     | <ul style="list-style-type: none"> <li>➤ Decorator.</li> <li>➤ Facade.</li> </ul>                              |
| De Comportamiento | <ul style="list-style-type: none"> <li>➤ Iterator.</li> <li>➤ Strategy.</li> <li>➤ Template Method.</li> </ul> |

**Singleton:**

“Este patrón garantiza que solamente se cree una instancia de la clase y provee un punto de acceso global a él. Todos los objetos que utilizan una instancia de esa clase usan la misma instancia” [ CITATION Mai11 \l 3082 ].

Un ejemplo del patrón singleton se refleja en el backend de la aplicación, todos los servicios implementados en la API encargados de obtener y guardar datos que son usados en los controladores son instancias únicas, estos pueden usarse en cualquier parte de la aplicación. Este comportamiento se debe a que cuando se definen estos servicios son marcados como proveedores con el decorador @Injectable(), permitiendo así poder inyectar en el constructor de otras clases la instancia declarada de ese servicio.

```

@ApiTags('report')
@Controller('report')
export class ReportController {
  constructor(private readonly reportService: ReportService) {}

  @CheckPolicies((ability: AppAbility) => ...)
  @Post(['create'])
  createUser(...)

  @Public()
  @CheckPolicies((ability: AppAbility) => ...)
  @Get('filter')
  filter(...)

  @Public()
  @CheckPolicies((ability: AppAbility) => ...)
  @Get('findById')
  findOne(@Query() idReportInput: IdReportInput): Promise<ReportEntity> {...}

  @CheckPolicies((ability: AppAbility) => ...)
  @Post('update')
  updateUser(...)

  @CheckPolicies((ability: AppAbility) => ...)
  @Delete('remove')
  removeUser(@Body('ids') ids: string[]): Promise<ReportEntity[]> {...}
}

```

**Ilustración 8: Ejemplo de Singleton**

**Decorador:**

En el sistema donde se implementa es en la API en la implementación de los permisos a las rutas, cuando el usuario hace una petición se verifica si tiene los permisos para acceder a esa ruta.

**Ilustración 9: Ejemplo de Decorador**

```

@Injectables()
export class PermissionsGuard implements CanActivate {
  constructor(
    private reflector: Reflector,
    private caslAbilityFactory: CaslAbilityFactory,
    private userService: UsersService
  ) {}

  async canActivate(context: ExecutionContext): Promise<boolean> {
    const req = context.switchToHttp().getRequest();
    const policyHandlers =
      this.reflector.get<PolicyHandler[]>(
        CHECK_POLICIES_KEY,
        context.getHandler()
      ) || [];

    const user = await this.userService.findOneByEmail(req.user.email);
    const ability = this.caslAbilityFactory.createForUser(user);

    return policyHandlers.every((handler) =>
      PermissionsGuard.execPolicyHandler(handler, ability)
    );
  }

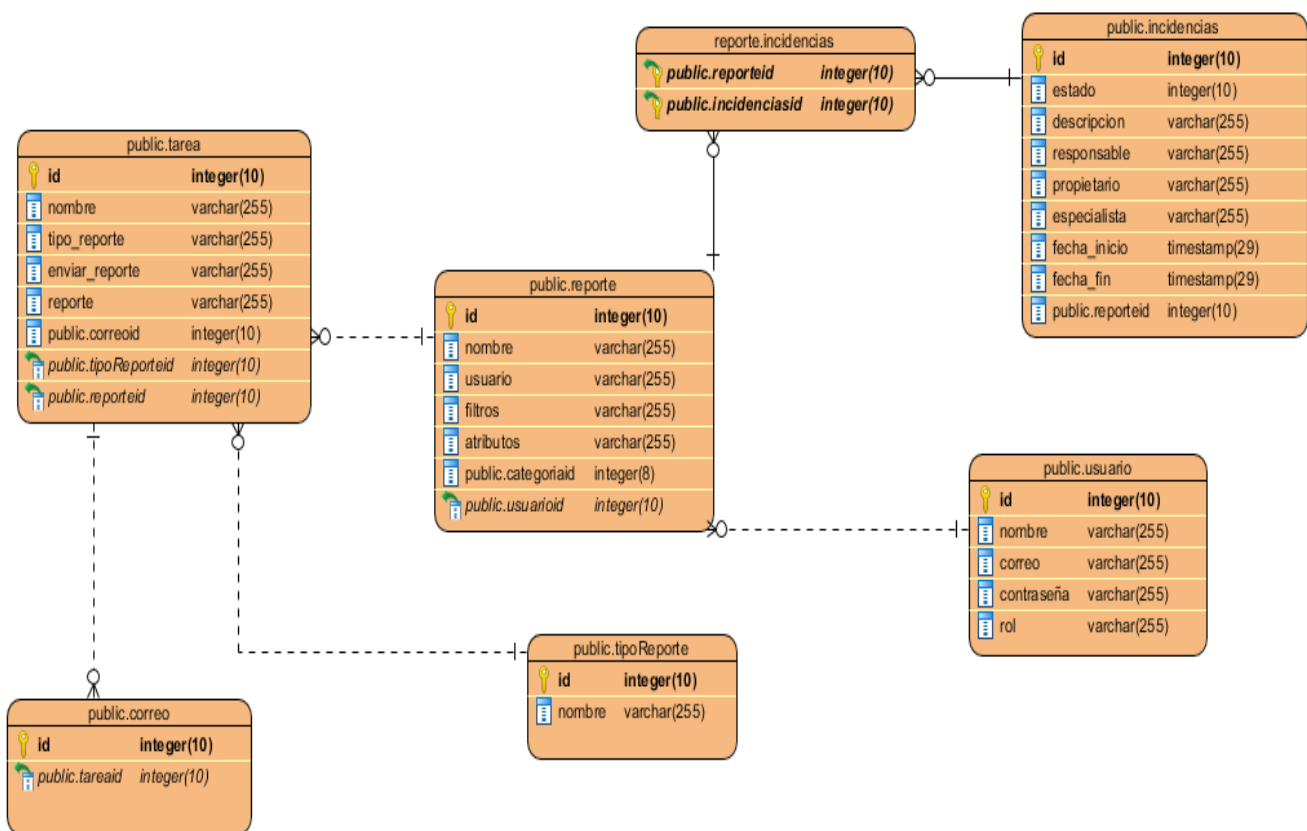
  private static execPolicyHandler(
    handler: PolicyHandler,
    ability: AppAbility
  ) {
    if (typeof handler === 'function') {
      return handler(ability);
    }
    return handler.handle(ability);
  }
}

```

## 2.8 Diagrama Entidad Relación

La solución propuesta está basada en el desarrollo sobre tecnología web, es por ello que es preciso tener una base de datos que permita almacenar toda la información referente al sistema de gestión de reportes de incidencias para los servicios de soporte. El diseño de esta base de datos es el proceso en el cual se toman los elementos conceptuales y se materializan. Estos elementos son representados mediante un diagrama de entidad-relación, definido como “una herramienta para el modelado de datos que facilita la representación de entidades de una base de datos”. Este modelo está basado en una percepción del mundo real que consta de una colección de objetos básicos, llamados entidades, y de relaciones entre esos objetos amorfos[ CITATION Wen \l 3082 ].

A continuación, se muestra el modelo de datos del sistema de gestión de reportes de incidencias para los servicios de Soporte, que cuenta con un total de 8 entidades.



**Ilustración 10: Diagrama Entidad Relación**

## 2.9 Conclusiones del Capítulo.

La arquitectura Cliente Servidor sirvió para la organización del desarrollo del software. La identificación de los roles y actores que intervienen en la propuesta de solución proporcionó la restricción del acceso a las funcionalidades del sistema. La elaboración de las historias de

usuarios permitió la descripción de los requerimientos del software y sus niveles de prioridad. Se concibió el plan de iteraciones y el orden de implementación de las historias de usuarios, dado sus niveles de prioridad, lo que permitió establecer un compromiso básico con el cliente.

Los patrones de diseño utilizados simplificaron el desarrollo de la propuesta de solución. El modelo conceptual de la propuesta de solución favoreció una mejor comprensión del software a implementar, y el diagrama entidad relación, ilustró la estructura de su base de datos.



# **CAPÍTULO III: VALIDACIÓN DEL SISTEMA DE GESTIÓN DE REPORTES DE INCIDENCIAS PARA LOS SERVICIOS DE SOPORTE.**

## **3.1 Introducción**

Las pruebas de software consisten en la dinámica de la verificación del comportamiento de un programa en un conjunto finito de casos de prueba, debidamente seleccionados de por lo general infinitas ejecuciones de dominio, contra la del comportamiento esperado. Son una serie de actividades que se realizan con el propósito de encontrar los posibles fallos de implementación, calidad o usabilidad de un programa u ordenador; probando el comportamiento del mismo. Lo cual no puede asegurar la ausencia de defectos; sólo puede demostrar que existen defectos en el software. Debido a la importancia que posee esta fase, el siguiente capítulo estará orientado a la realización de pruebas a las distintas funcionalidades del software para verificar que funcionen correctamente.

## **3.2 Diagrama de despliegue.**

“El objetivo de estos diagramas es mostrar la disposición de las particiones físicas del sistema de información y la asignación de los componentes software a estas particiones. Es decir, las relaciones físicas entre los componentes software y hardware en el sistema a entregar. En estos diagramas se representan dos tipos de elementos, nodos y conexiones, así como la distribución de componentes del sistema de información con respecto a la partición física del sistema”[ CITATION man22 \l 3082 ].

Nodo

Se representa con la figura de un cubo. El nodo se etiqueta con un nombre representativo de la partición física que simboliza. Se pueden asociar a los nodos subsistemas de construcción.

Conexión

Las conexiones se representan con una línea continua que une ambos nodos y pueden tener una etiqueta que indique el tipo de conexión. (ejemplo: canal, red, protocolo, etc.)

### **Descripción de elementos e interfaces de comunicación:**

**Dispositivo de Cliente:** La estación de trabajo necesita un navegador web para conectarse al sistema hospedado en el servidor de aplicaciones utilizando el protocolo de comunicación HTTPS.

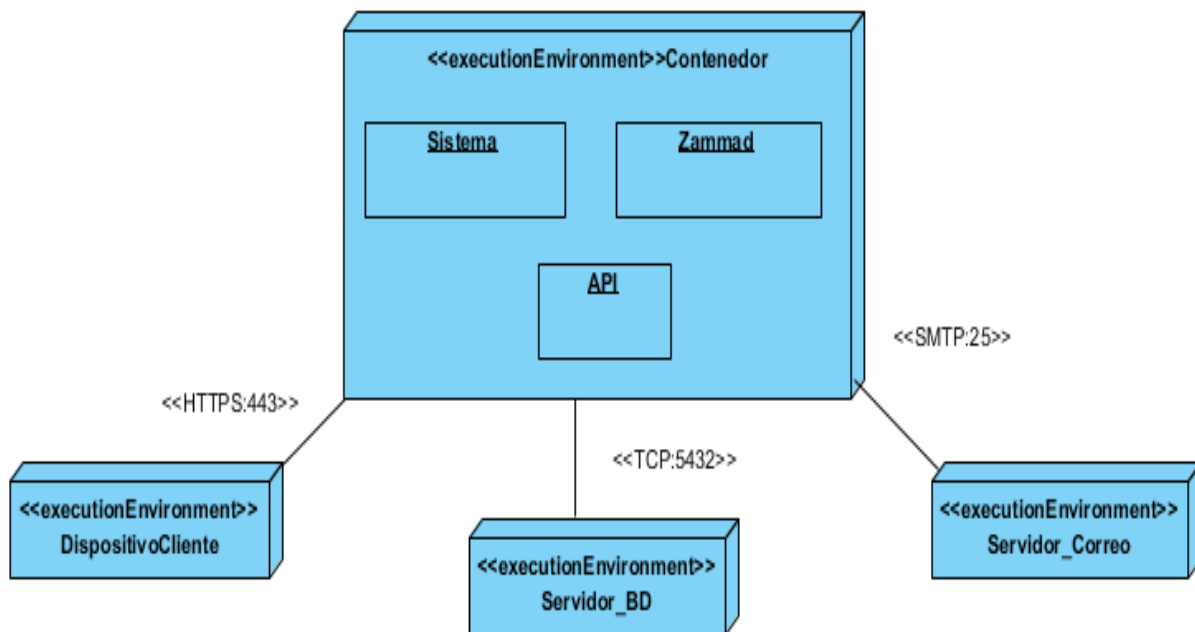
**Servidor de aplicaciones:** Es la estación de trabajo que hospeda el código fuente de la aplicación y que le brinda al usuario las interfaces para realizar los procesos del sistema. Es un servidor de aplicaciones web NGINX y se comunica con el servidor de base de datos donde se almacenan los datos de la aplicación realizando la comunicación mediante el protocolo TCP.

**Servidor de BD:** Servidor de bases de datos PostgreSQL, es el encargado del almacenamiento de

los datos del sistema. Se comunica con el servidor de aplicaciones del sistema, posibilitando el acceso mediante el usuario con privilegios para las operaciones determinadas a realizarse en el mismo.

**Servidor de correo:** Este servidor es el encargado del envío de correos electrónicos comunicándose a través de protocolo SMTP.

A continuación, se presenta al modelo de despliegue correspondiente a la propuesta de solución:



**Ilustración 11: Diagrama de despliegue del sistema (elaboración propia)**

## **3.4 Validación del Sistema de gestión de reportes de incidencias para los servicios de soporte en la UCI.**

### **3.4.1 Prueba de Caja Blanca**

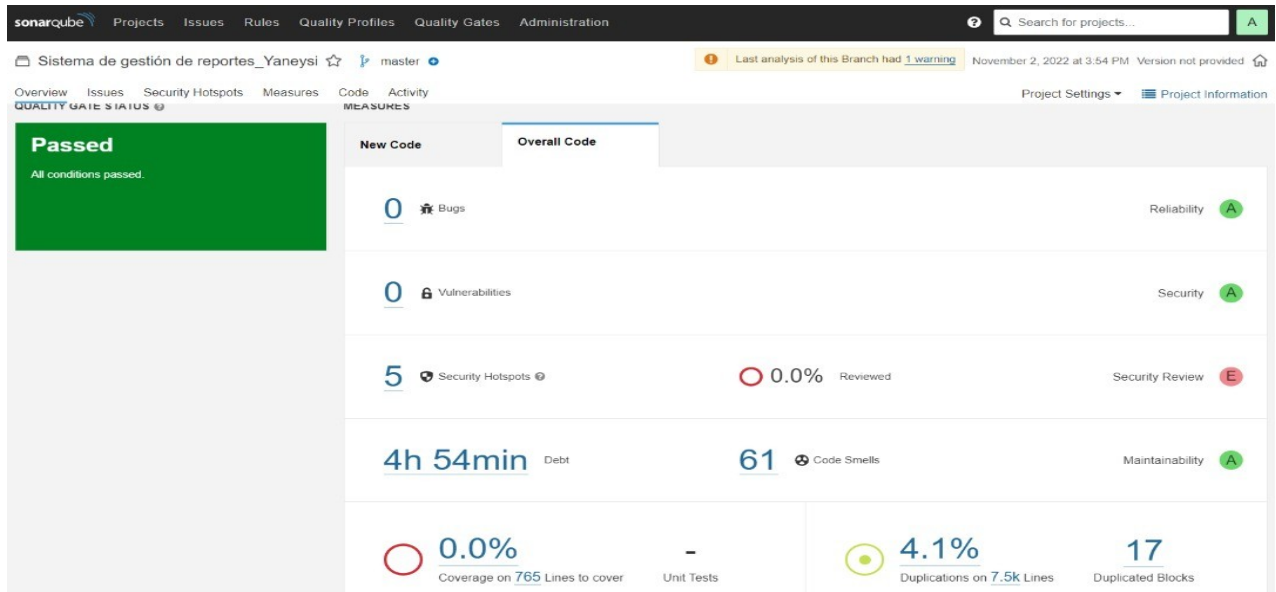
Dentro de la prueba de funcionalidad se realizó el método de Caja Blanca, “este puede definirse como una técnica de monitorización o prueba de [software](#) en la se analiza el diseño, código y estructura interna, con el objetivo de mejorar propiedades como la seguridad y el uso eficiente del sistema. Estas pruebas se caracterizan principalmente porque son los propios sistemas y aplicaciones quienes exponen sus métricas para que el usuario pueda leerlas, analizarlas y tomar decisiones y acciones en función de la obtención de un resultado u otro”[ CITATION Kee22 \l 1033 ].

El método de caja blanca se realizó mediante una plataforma llamada SonarQube “que es una plataforma de código abierto para la inspección continua de la calidad del código a través de diferentes herramientas de análisis estático de código fuente. Proporciona métricas que ayudan a mejorar la [calidad del código](#) de un programa permitiendo a los equipos de desarrollo hacer seguimiento y detectar errores y vulnerabilidades de seguridad para mantener el código limpio. Es una herramienta esencial para la fase de testing<sup>5</sup> y auditoría de código dentro del ciclo de desarrollo de una aplicación y se considera perfecta para guiar a los equipos de desarrollo durante las revisiones de código. Soporta una etapa de inspección continua”[ CITATION Sen21 \l 1033 ].

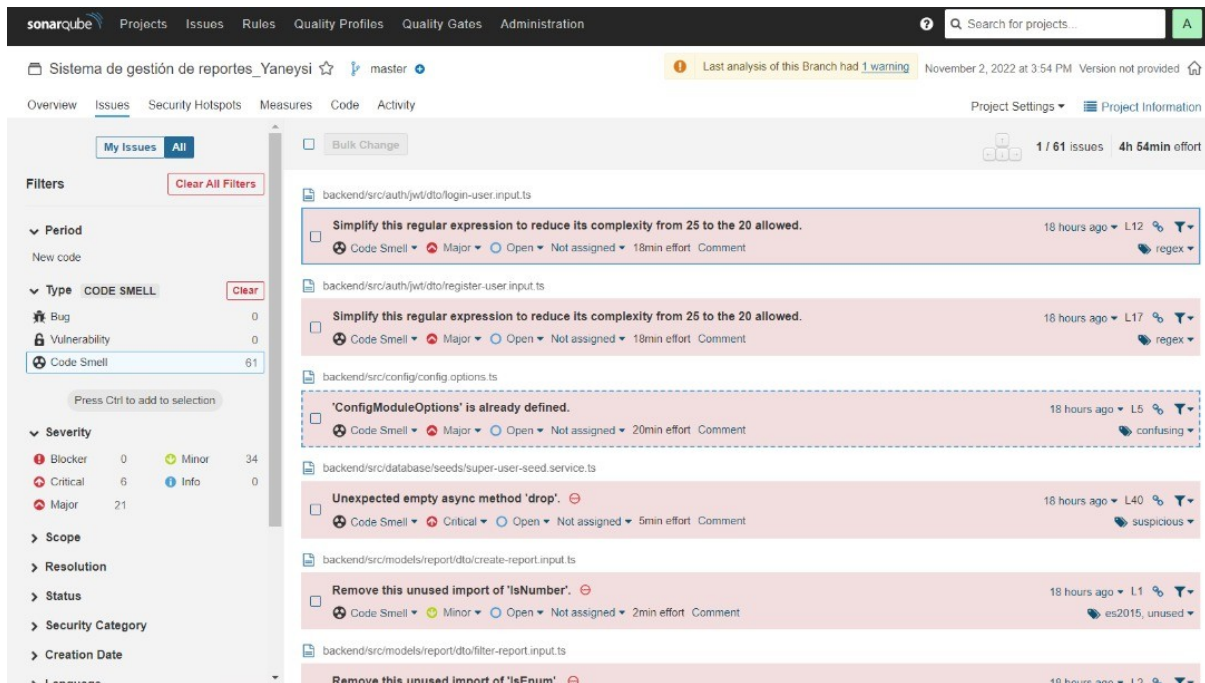
Donde el sistema no presenta ningún error en la confiabilidad, ninguna vulnerabilidad de seguridad y con solo 5 puntos de acceso en consultas de seguridad, esto no es más que código sensible que requiere una revisión manual para evaluar si existe o no una vulnerabilidad.

---

<sup>5</sup>es un proceso para verificar y validar la funcionalidad de un programa o una aplicación de software



**Ilustración 12: Captura de pantalla de la plataforma SonarQube.**



**Ilustración 13: Captura de pantalla de la plataforma SonarQube.**

### 3.4.2 Prueba de Aceptación

El objetivo de las pruebas de aceptación es validar que un sistema cumple con el funcionamiento esperado y permitir al usuario de dicho sistema que determine su aceptación, desde el punto de vista de su funcionalidad y rendimiento.

“En la ingeniería de software, las pruebas de aceptación (PA) se realizan para establecer el grado de confianza en un sistema, partes del mismo o en sus características no funcionales. La confianza en el sistema estará determinada por su grado de adherencia a las necesidades, requerimientos y procesos de negocio solicitados por el usuario o cliente. Es en función a estos que el usuario debe decidir si acepta o no el sistema que le está siendo entregado”[ CITATION Wen \l 3082 ].

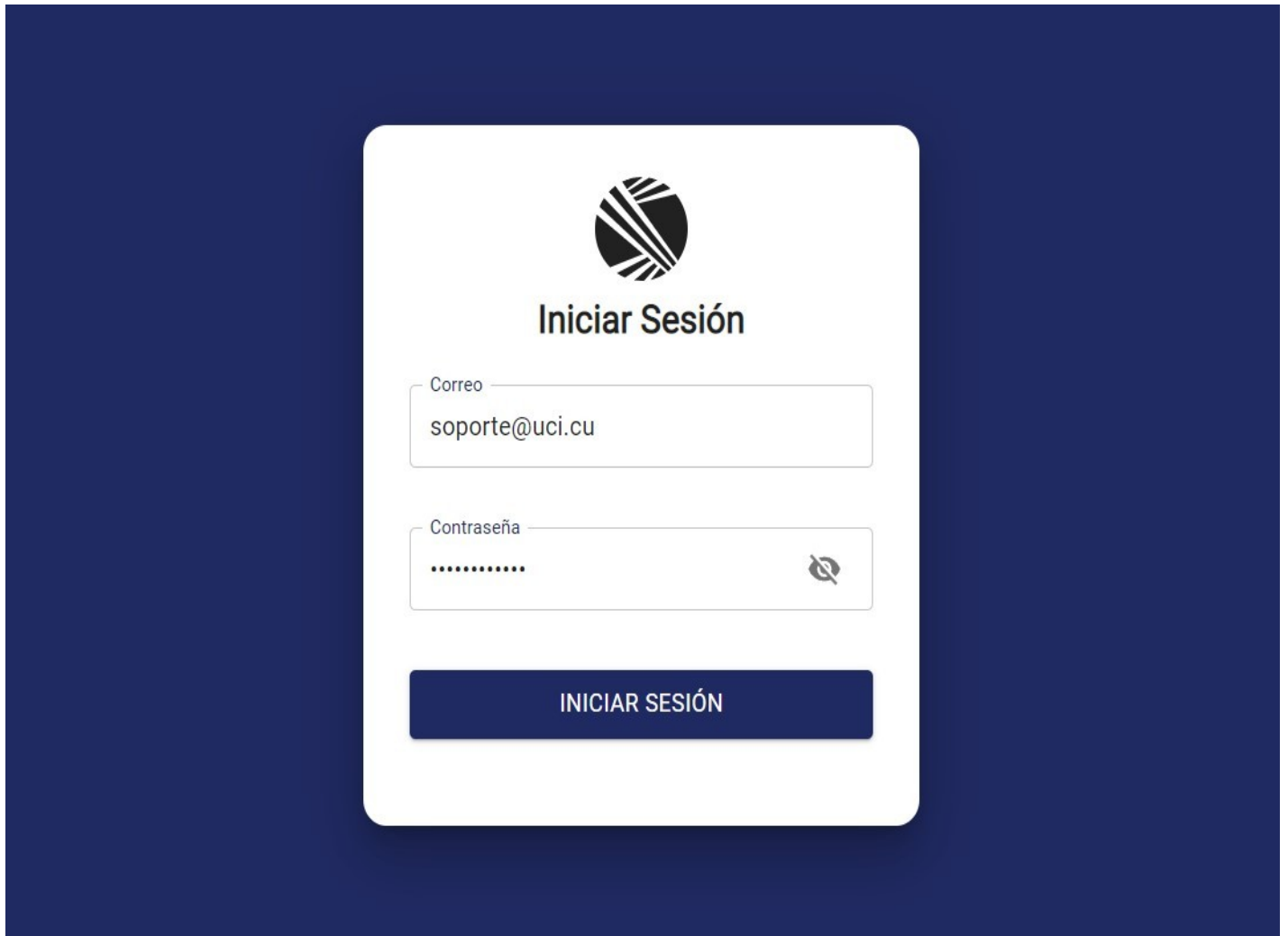
Por lo tanto, estas pruebas suelen ser responsabilidad de los clientes o usuarios del sistema.

Para realizar las PA se presentó la aplicación informática al cliente. La revisión se realizó teniendo en cuenta los requisitos definidos en la Especificación de Requisitos. Como resultado de la conformidad del cliente con la aplicación informática se emitió el acta de aceptación.

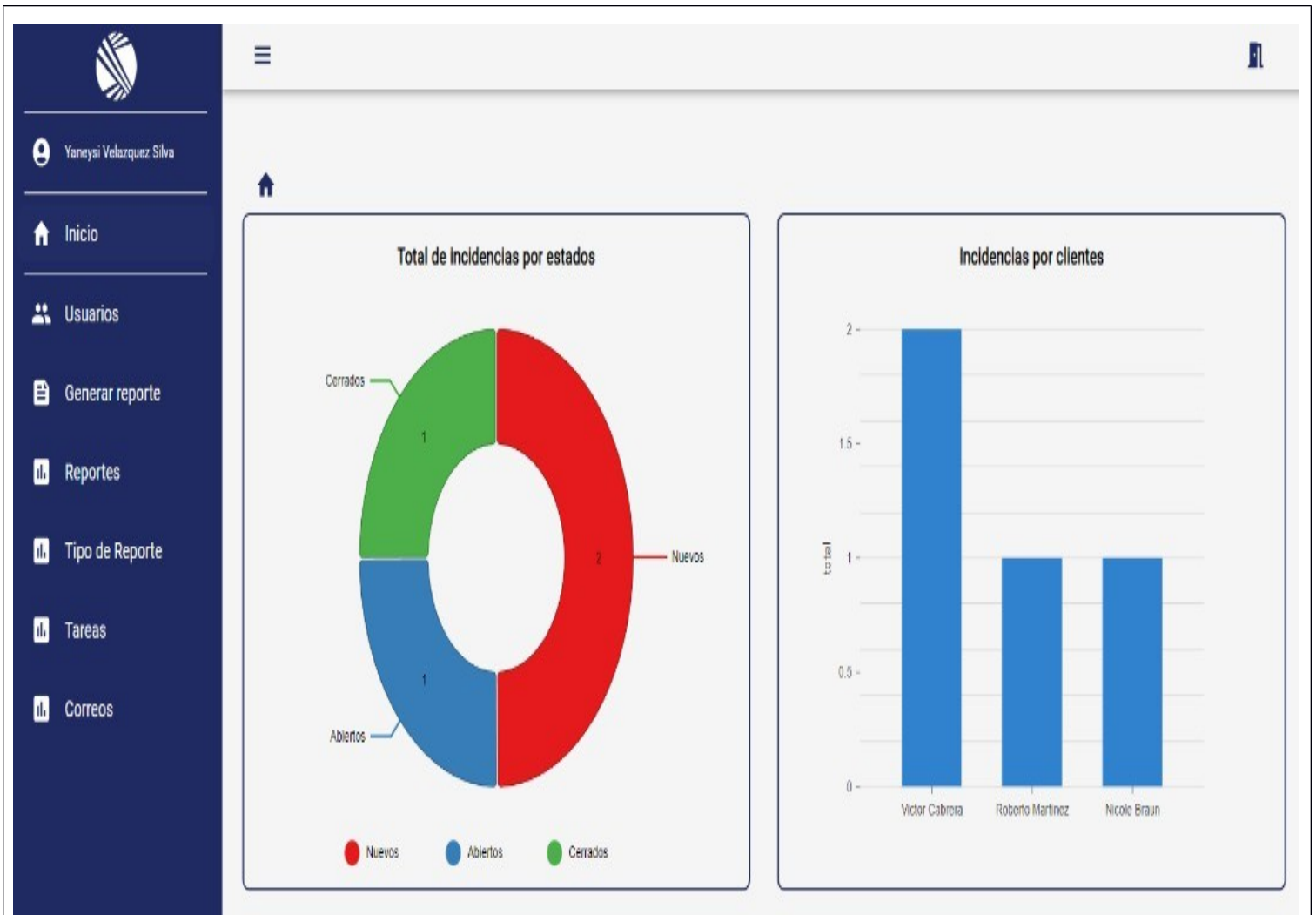
El acta de aceptación se encuentra en la sección de los Anexos. ([verAnexo #2](#))

### **3.5 Interfaces principales del Sistema de gestión de reportes de incidencias**

Una vez desarrollado el Sistema de gestión de reportes de incidencias para los servicios de soporte, es posible visualizar las pantallas principales del mismo, donde se observa el resultado obtenido durante la implementación de los requisitos funcionales descritos en el capítulo 2.



***Ilustración 14: Captura de pantalla del sistema (Autenticar Usuario)***



**Ilustración 15: Captura de pantalla del sistema (Generar reportes)**

## **3.6 Conclusiones del capítulo**

En este capítulo se han abordado los elementos de la implementación de la propuesta de solución, así como las pruebas realizadas a la misma, lo cual permite arribar a las siguientes conclusiones:

1. La elaboración del diagrama de despliegue representa de forma visual las relaciones físicas que existen entre los componentes de software y hardware en el sistema.
2. El correcto uso de los estándares de codificación, permite que el código del sistema desarrollado sea legible para lograr una fácil y mejor comprensión del mismo, lo cual es de utilidad para el mantenimiento del sistema.
3. El proceso de validación de la propuesta de solución, a través de la estrategia de pruebas especificada, arroja como resultado que el sistema implementado responde a los requerimientos definidos por el cliente.



## **CONCLUSIONES FINALES**

De manera general, la presente investigación concluyó con el desarrollo del Sistema de gestión de reportes de incidencias para los servicios de soportes de la Universidad de las Ciencias Informáticas, el cual sirve de apoyo al proceso de gestión de reportes de incidencias de la UCI.

1. El análisis y la fundamentación teórica de los principales conceptos asociados a la investigación, permitió lograr una mayor comprensión del alcance de la investigación y esclarecer su objeto de estudio.
2. La sistematización del marco teórico de la investigación científica y del estado actual de las herramientas informáticas para la gestión de reportes de incidencias, posibilitó la definición del ambiente de desarrollo para la implementación de la solución propuesta.
3. La integración de diversas áreas del conocimiento como son la ingeniería y gestión de software, base de datos, programación, entre otras, permitió el análisis, diseño e implementación del sistema.
4. La solución fue validada a partir de la definición correcta de una estrategia de pruebas, que permitió comprobar el correcto funcionamiento del Sistema de gestión de reportes de incidencias para los servicios de soportes de la Universidad de las Ciencias Informáticas, a partir de los requerimientos definidos por el cliente.

## **RECOMENDACIONES**

Para el desarrollo de futuras investigaciones relacionadas con la presente, se propone:

-Estudiar estructura de la entidad ticket y entidades relacionadas en las nuevas versiones de la plataforma Zammad para integrar el sistema implementado con las nuevas versiones de la plataforma.

## **BIBLIOGRAPHY**

**1library.co. 2022.** 1library.co. [En línea] 2022. <https://1library.co/article/herramienta-ca-se-visual-paradigm->

**Alina. 2020.** userlike.com. [En línea] 21 de agosto de 2020. <https://www.userlike.com/es/blog/ventajas-desventajas-chat-en-vivo>.

**Azaustre, Carlos. 2018.** Carlos Azaustre. [En línea] 2018. <https://carlosazaustre.es/como-funciona-flux>.

**Blancarte, Oscar. 2020.** *Introduccion a la arquitectura de Software*. 2020.

**Borges, Santiago. 2019.** blog.infranetworking.com. [En línea] 19 de noviembre de 2019. <https://blog.infranetworking.com/servidor-postgresql/>.

**Bruno. 2018.** [En línea] 10 de abril de 2018. <https://codearmy.co/comparaci%C3%B3n-react-contra-angular-2402f761b14e>.

**Coelho, Fabián. 22.** Significados.com. [En línea] 6 de junio de 22. <https://www.significados.com/metodologia/>.

**ComparaSoftware. 2022.** ComparaSoftware. [En línea] 2022. <https://blog.comparasoftware.com/apps-help-desk-gratis/>.

**docs, MDN web. 2022.** MDN web docs. [En línea] 5 de septiembre de 2022.

**EDteam. 2022.** [En línea] 2022. <https://ed.team/cursos/react-state>.

**Ekon. 2021.** Ekon. [En línea] 28 de julio de 2021. [www.ekon.es/blog/sistemas-de-gestion-integral-para-el-funcionamiento-optimo-de-la-empresa/](http://www.ekon.es/blog/sistemas-de-gestion-integral-para-el-funcionamiento-optimo-de-la-empresa/)

**Epitech. 2022.** Epitech.com. [En línea] 19 de febrero de 2022. <https://www.epitech-it.es/nestjs-que-es/>.

**es?, ¿Qué. 2021.** ¿Qué es? [En línea] abril de 2021. <https://www.quees.pro/reporte/>.

**EUROINNOVA. 2022.** EUROINNOVA. [En línea] 2022. <https://www.euroinnova.edu.es/blog/porque-es-importante-la-informacion/>

**Francisco José García Peñalvo, Alicia García Holgado. 2018.** *Departamento de Informática y Automática*. Salamanca : s.n., 2018.

**GetApp.** GetApp. [En línea] <https://www.getapp.es/software/102433/megalytic>.

**González, Pedro. 2021.** Billin. [En línea] 2021. <https://www.billin.net/glosario/definicion-incidencia/>.

**helppeople.** helppeoplecloud.com. [En línea] <https://helppeoplecloud.com/2021/08/20/help-desk-open-source-software-ventajas-y-desventajas/>

**InterFuerza. 2021.** InterFuerza.com. [En línea] 7 de junio de 2021. <https://www.interfuerza.com/importancia-del-manejo-de-reportes/>.

**Jiménez, Juan Diego Pérez. 2019.** OpenWebinars. [En línea] 20 de enero de 2019. <https://openwebinars.net/blog/que-es-html5/>

**Julián, Pérez Porto y María , Merino. 2021.** Definición.de. [En línea] 2021. <https://definicion.de/reporte/>.

**KeepCoding. 2022.** KeepCoding Tech School. [En línea] 3 de junio de 2022. <https://keepcoding.io/blog/que-son-pruebas-de-caja-blanca/>.

**Kord, Maia. 2011.** TUXNOTS. [En línea] 3 de diciembre de 2011. <https://sites.google.com/site/tuxnots/materias-de-la-facu/metodologia-de-sistemas/>

**L, Samantha. 2020.** Software Advice. [En línea] 25 de noviembre de 2020. <https://www.softwareadvice.es/reviews/79731/megalytic>.

**Lauren.** [En línea] <https://www.softwareadvice.es/software/79731/megalytic>.

**Lerman, Craig. 1999.** *UML y Patrones*. Mexico : s.n., 1999.

**Mahajan, Yogesh. 2022.** pgadmin.org. [En línea] 27 de mayo de 2022. <https://www.pgadmin.org/>.

**Mamalova, Reynaldo Columbie. 2019.** *Sistema para la gestión de reportes de incidencias de mantenimiento en la Universidad de las Ciencias Informáticas*. La Habana : s.n., 2019.

**2022.** manuel.cillero.es. [En línea] 2022. <https://manuel.cillero.es/doc/metodologia/metri-ca-3/tecnicas/diagrama-de-despliegue/>.

**MEGALYTIC.** megalytic.com. [En línea] <https://www.megalytic.com/>.

**Microsoft. 2022.** [En línea] 23 de mayo de 2022. <https://visual-studio-code.uptodown.com/windows>.

**Moreno, Juanita. 2021.** Hubspot. [En línea] 30 de 11 de 2021. <https://blog.hubspot.es/service/que-es-soporte-tecnico/>

**Reactjs.org. 2019.** Reactjs.org. [En línea] 21 de febrero de 2019. <https://es.reactjs.org/>.

**Sánchez, Tamara Rodríguez. 2015.** Metodología de desarrollo para la Actividad productiva de la UCI. [En línea] 6 de marzo de 2015.

**Santos, Diego. 2021.** HubSpot. [En línea] 9 de diciembre de 2021. <https://blog.hubspot.es/website/que-es-css>.

**Silva, Douglas da. 2020.** Blog de Zendesk. [En línea] 24 de junio de 2020. <https://www.zendesk.com.mx/blog/sistema-help-desk/>.

**Soporte, Centro de. 2019.** SOPORTE. [En línea] 2019. <https://soporte.uci.cu/inicio>.

**V, Vasilina. 2017.** capterra.es. [En línea] 2017. <https://www.capterra.es/software/143200/brand-embassy>.

**Vargas, Yohannia López.** *Revista Cubana de Ciencias Informáticas*.

**Vilela, Carlota. 2022.** [En línea] 4 de 5 de 2022.

<https://www.freelancermap.com/blog/es/que-hace-soporte-tecnico-nivel-1/>.

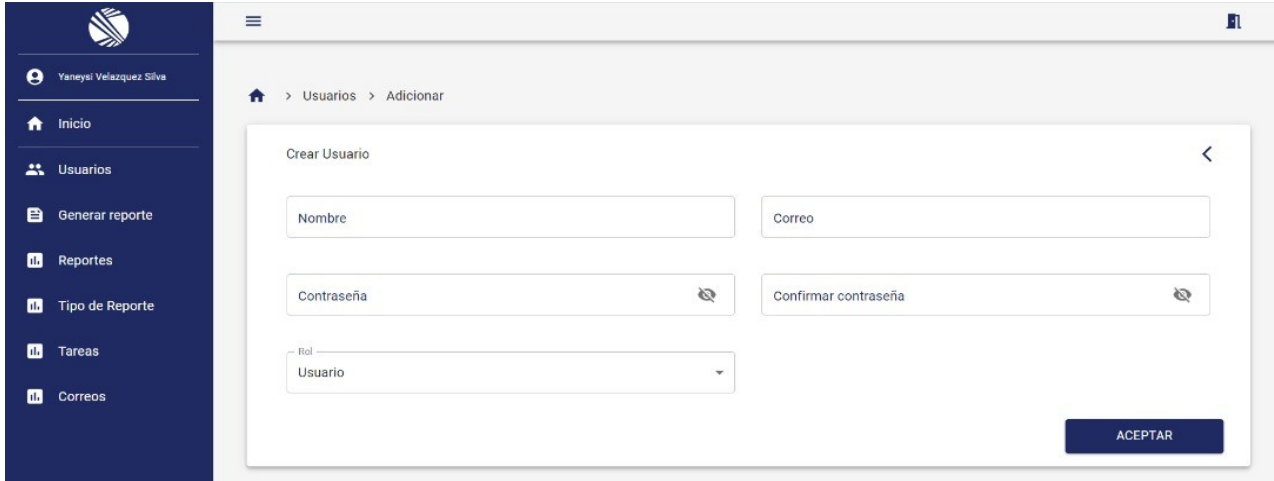
**VisualParadigm.** visual-paradigm.com. [En línea] <https://www.visual-paradigm.com/>.

**Wendy , Llanes San Martín y Leandro Ernesto , Fundichely Velazco. 2018.** Sistema de selección basada en habilidades para la gestión de recursos humanos. 2018.

**Yohannia López Vargas, Alejandro Vázquez Chavez. 2016.** Scielo. *Scielo*. [En línea] 15 de abril de 2016. <http://scielo.sld.cu/>

## ANEXOS

### ANEXO #1 HISTORIAS DE USUARIOS

|   |                          |
|---|--------------------------|
|   |                          |
| Número: HU_1  | Requisito: Crear usuario |
| Programador: Yaneysi Velázquez Silva  | Iteración Asignada: 2    |
| Prioridad: Media  | Tiempo Estimado: 2 día   |
| Riesgo en Desarrollo: Bajo  | Tiempo Real: 0.4         |
| <p>Descripción:</p> <p>Se permite crear un usuario</p>  |                          |
| <p>Observaciones:</p> <p>El sistema tiene que tener conexión.</p>   |                          |
| <p>Prototipo elemental de interfaz gráfica de usuario:</p>  |                          |

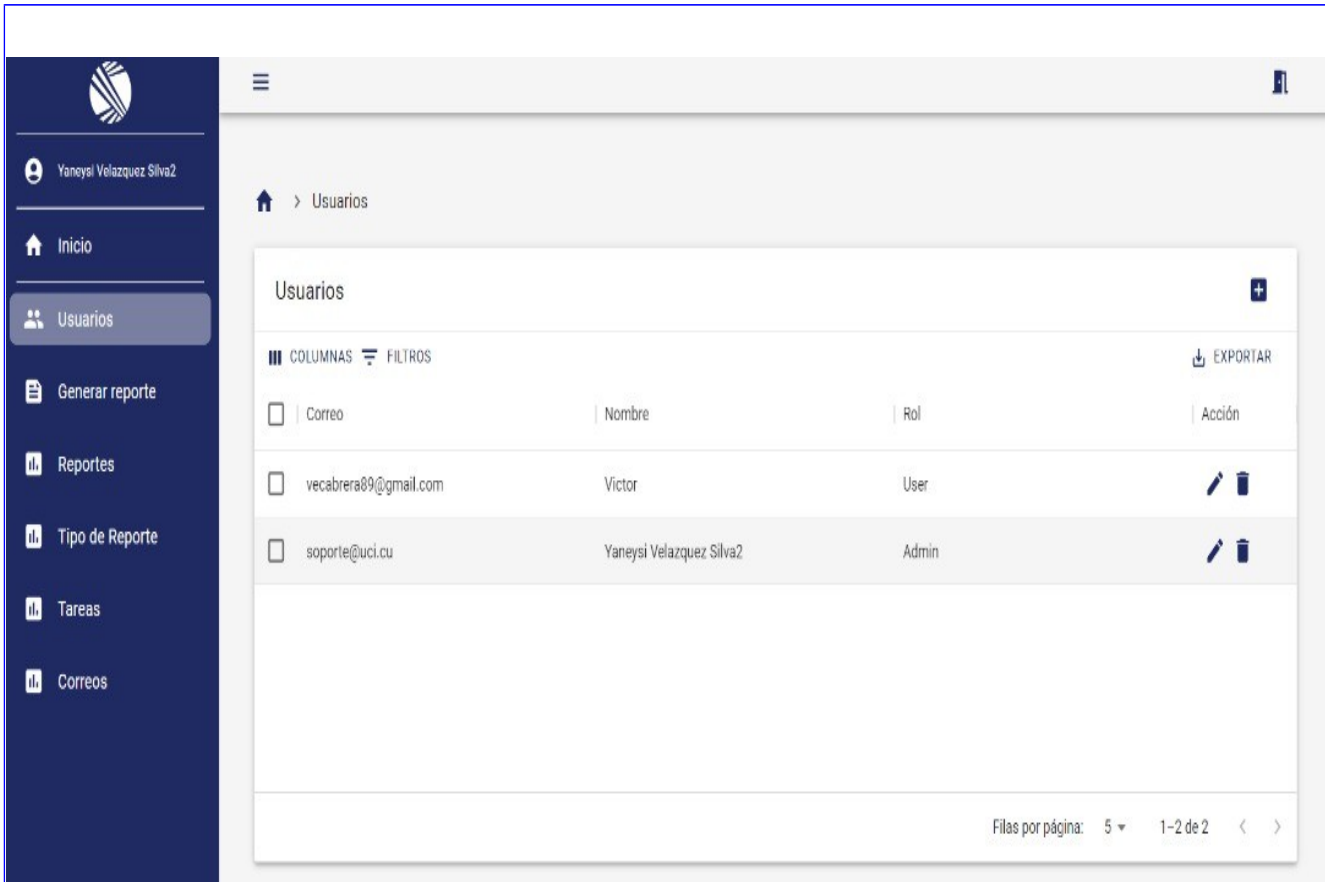
|                                      |                              |
|--------------------------------------|------------------------------|
|                                      |                              |
| Número: HU_2                         | Requisito: Adicionar usuario |
| Programador: Yaneysi Velázquez Silva | Iteración Asignada: 2        |

|  |                         |
|--|-------------------------|
| Prioridad: Media   | Tiempo Estimado: 2 días |
| Riesgo en Desarrollo: Baja                               | Tiempo Real: 0.4        |
| Descripción:<br>Registrar los datos de un usuario nuevo. |                         |

Observaciones:

Prototipo elemental de interfaz gráfica de usuario:

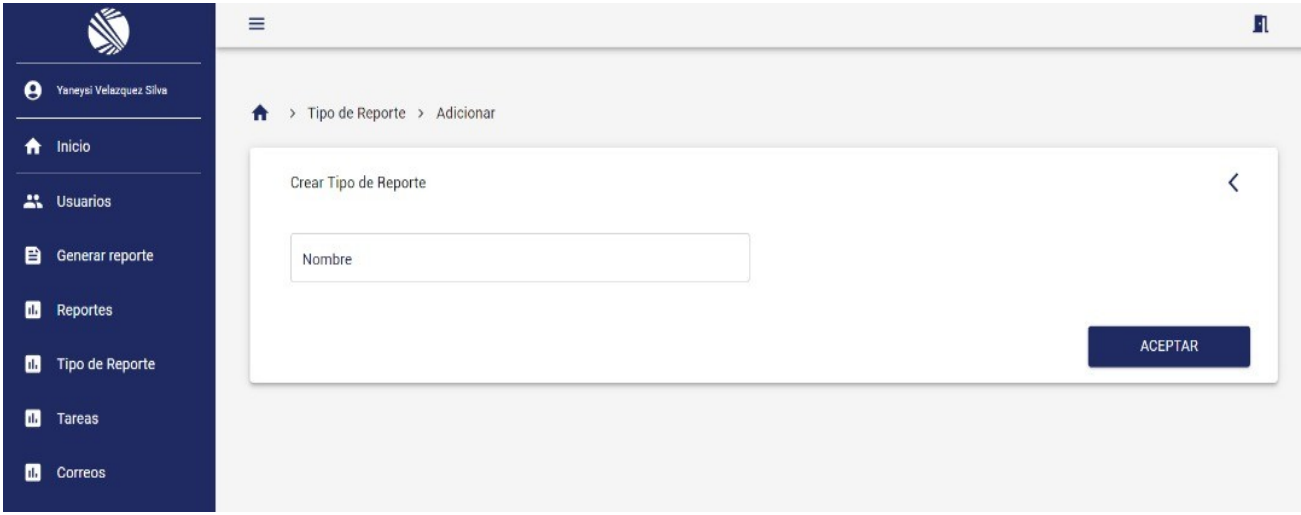
|  |                           |
|--|---------------------------|
| Número: HU_4   |                           |
| Número: HU_4   | Requisito: Listar usuario |
| Programador: Yaneyssi Velázquez Silva                    | Iteración Asignada: 2     |
| Prioridad: Alta  | Tiempo Estimado: 2 días   |
| Riesgo en Desarrollo: Alto                               | Tiempo Real: 0.4          |
| Descripción:<br>Permite listar los usuarios del sistema. |                           |
| Observaciones:<br>El sistema tiene que tener conexión.   |                           |
| Prototipo elemental de interfaz gráfica de usuario:      |                           |



|  |   |
|--|---|
|  |   |
| Número: HU_10  | Requisito: Generar contenedor de la aplicación. |
| Programador: Yaneysi Velázquez Silva   | Iteración Asignada:1                            |
| Prioridad: Alta  | Tiempo Estimado: 2 días                         |
| Riesgo en Desarrollo: Alto   | Tiempo Real: 0.4                                |
| Descripción:   |   |
| Para generar el contenedor de la aplicación se debe configurar un Docker Compose van a estar las imágenes que hacen falta para levantar la aplicación, en ese caso es Postgres en Nginx. |   |
| Observaciones:   |   |
| Prototipo elemental de interfaz gráfica de usuario:  |   |

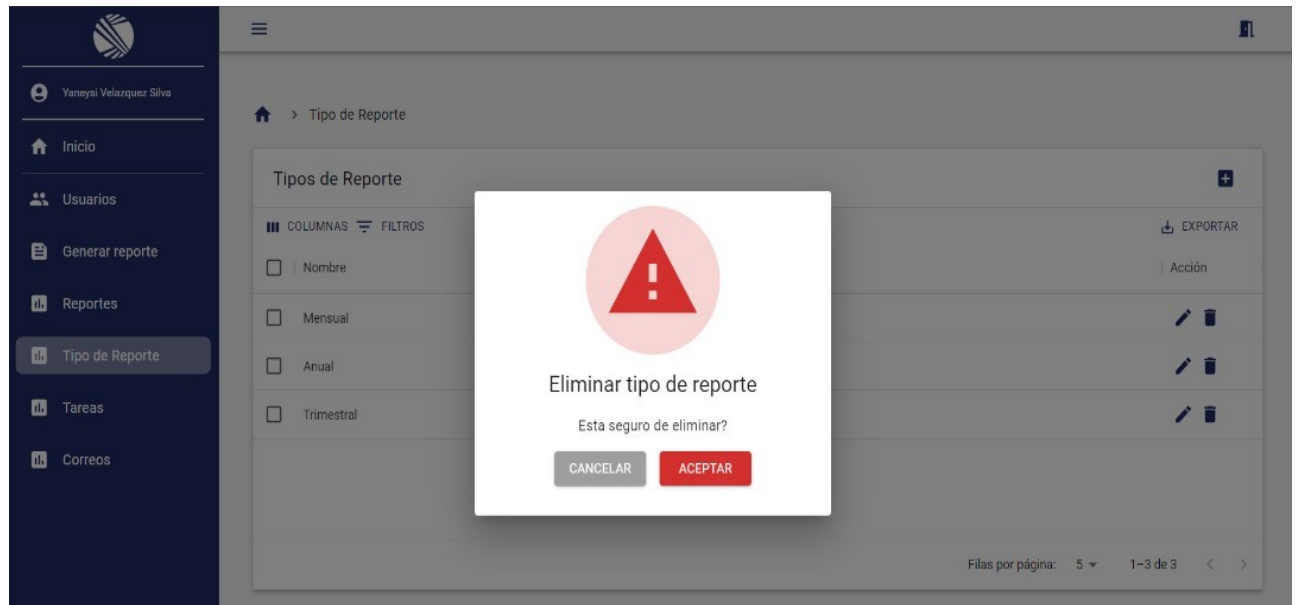
|               |                                  |
|---------------|----------------------------------|
|               |                                  |
| Número: HU_16 | Requisito: Crear tipo de reporte |



|   |                         |
|---|-------------------------|
| Programador: Yaneysi Velázquez Silva  | Iteración Asignada: 1   |
| Prioridad: Alta   | Tiempo Estimado: 4 días |
| Riesgo en Desarrollo: Alto  | Tiempo Real: 0.8        |
| <b>Descripción:</b><br>Permite crear un tipo de reporte como, por ejemplo: Mensual, anual y trimestral. |                         |
| <b>Observaciones:</b><br>El usuario tiene que estar autenticado.  |                         |
| <b>Prototipo elemental de interfaz gráfica de usuario:</b>  |                         |
|                      |                         |

|  |                                  |
|--|----------------------------------|
| Número: HU_17  | Requisito: Eliminar tipo reporte |
| Programador: Yaneysi Velázquez Silva                             | Iteración Asignada: 2            |
| Prioridad: Alta  | Tiempo Estimado: 4 días          |
| Riesgo en Desarrollo: Alto                                       | Tiempo Real: 0.8                 |
| <b>Descripción:</b><br>Permite eliminar un reporte.              |                                  |
| <b>Observaciones:</b><br>El usuario tiene que estar autenticado. |                                  |

## Prototipo elemental de interfaz gráfica de usuario:



|  |                                   |
|--|-----------------------------------|
| Número: HU_18  | Requisito: Modificar tipo reporte |
| Programador: Yaneysi Velázquez Silva                       | Iteración Asignada: 2             |
| Prioridad: Alto  | Tiempo Estimado: 4 día            |
| Riesgo en Desarrollo: medio                                | Tiempo Real: 0.8                  |
| Descripción:<br>Permite al usuario modificar los reportes. |                                   |
| Observaciones:<br>El usuario tiene que estar autenticado.  |                                   |
| Prototipo elemental de interfaz gráfica de usuario:        |                                   |

The screenshot shows a mobile application interface. On the left is a dark blue sidebar with a logo at the top and a list of menu items: 'Inicio', 'Usuarios', 'Generar reporte', 'Reportes', 'Tipo de Reporte', 'Tareas', and 'Correos'. The user's name 'Yanaysi Velazquez Silva' is displayed at the top of the sidebar. The main content area has a breadcrumb trail: 'Inicio > Tipo de Reporte > Editar'. Below this is a white modal window titled 'Editar Tipo de Reporte' with a back arrow on the right. Inside the modal is a text input field labeled 'Nombre' with the text 'Trimestral' entered. At the bottom right of the modal is a dark blue button labeled 'ACEPTAR'.

|  |   |
|--|---|
| Número: HU_19  | Requisito: Crear tareas programadas (Enviar reporte). |
| Programador: Yanaysi Velázquez Silva   | Iteración Asignada: 3                                 |
| Prioridad: Bajo  | Tiempo Estimado: 2 día                                |
| Riesgo en Desarrollo: Bajo   | Tiempo Real: 0.4                                      |
| <p>Descripción:</p> <p>Una vez creado los tipos de reporte se pueden crear las tareas programadas para enviarles ya sea mensual, trimestral o anual los reportes a los clientes en una fecha programada.</p> |   |
| <p>Observaciones:</p> <p>El sistema tiene que tener conexión.</p>  |   |
| <p>Prototipo elemental de interfaz gráfica de usuario:</p>   |   |

|   |  |
|---|--|
|   |  |
| Número: HU_20                                       | Requisito: Enviar notificaciones por Correo. |
| Programador: Yaneysi Velázquez Silva                | Iteración Asignada: 3                        |
| Prioridad: Bajo                                     | Tiempo Estimado: 2 día                       |
| Riesgo en Desarrollo: Bajo                          | Tiempo Real: 0.4                             |
| Descripción:  |  |
| Enviar el reporte creado por correo a un cliente.   |  |
| Observaciones:                                      |  |
| El sistema tiene que tener conexión.                |  |
| Prototipo elemental de interfaz gráfica de usuario: |  |

## ANEXO #2 ACTA DE ACEPTACIÓN.

## ACTA DE ACEPTACIÓN

En cumplimiento del Convenio de colaboración con el centro de Soporte Tecnológico y en función de la ejecución del proyecto: Sistema de gestión de reportes de incidencias para los servicios de soporte en la Universidad de las Ciencias Informáticas, se hace entrega de los productos que se relacionan a continuación: aplicación informática.

La Parte Cliente, luego de haber revisado los productos de trabajo determina que: Acepta

| Entrega                                     | Recibe                                    |
|---|---|
| Nombre y apellidos: Yaneysi Velázquez Silva | Nombre y apellidos: Neybis Lago Clara     |
| Cargo: Jefe de proyecto                     | Cargo: Subdirectora del centro de Soporte |

### Representante Parte Suministradora

Nombre y Apellidos: Neybis Lago Clara

Cargo: Subdirectora del centro de Soporte Tecnológico

Firma:



Neybis Lago Clara  
 C=CU, OU=Centro de Soporte  
 Tecnológico, O=UCI,  
 CN=Neybis Lago Clara,  
 E=nlago@uci.cu  
 Soy el autor de este documento  
 2022.11.02 10:00:02-05'00'

Observador independiente

Nombre y Apellidos: Víctor E Cabrera Sanz

Cargo: Jefe de línea

Firma:

Fecha: 27/10/2022

## ANEXO #3 CAPTURA DE PANTALLA SONARQUBE

The screenshot shows the SonarQube web interface. At the top, there are navigation tabs for Projects, Issues, Rules, Quality Profiles, Quality Gates, and Administration. A search bar is present on the right. Below the navigation, the page title is 'Sistema de gestión de reportes\_Yaneysi' with a 'master' branch indicator. A notification states 'Last analysis of this Branch had 1 warning' on November 2, 2022. The main content area is titled 'Security Hotspots' and shows a list of hotspots. The selected hotspot is 'Denial of Service (DoS)' with a severity of 'MEDIUM'. The description reads: 'Make sure the regex used here, which is vulnerable to super-linear runtime due to backtracking, cannot lead to denial of service.' The code snippet shows a TypeScript class 'LoginUserInput' with a '@Matches' decorator. The regex pattern is '^(?=[^\\d])(?=[^\\w+])(?=[^\\s])(?=[^A-Z])(?=[^a-z]).\*\$'. A comment below the code says 'message: 'password too weak','. The interface also shows a 'Change status' button and an 'Assignee' field set to 'Not assigned'.

## ANEXO #4 GUÍA DE OBSERVACIÓN.

## Guía de Observación

**Objetivo:** Determinar problemas en la gestión de reportes en la plataforma Zammad.

**Sujeto de estudio:** Centro de Soporte

**Area de observacion:** Plataforma de gestión de incidencias

**Observadores:** Yaneysi Velazquez Silva

**Tiempo de Observación:** Dos horas diarias por tres días

**Aspectos a evaluar/observar:** Gestión de los reportes en la Plataforma Zammad



| Aspectos a evaluar/observar                            | Si | No | Observaciones |
|--|----|----|---------------|
| Permite filtrar por múltiples campos                   |    |    |               |
| Permite exportar datos a PDF                           |    |    |               |
| Permite exportar datos como gráficas                   |    |    |               |
| Muestran la descripción de las incidencias             |    |    |               |
| Permite seleccionar los campos a mostrar en el reporte |    |    |               |
| Los nombres de los campos se muestran en español       |    |    |               |



### ANEXOS #5 ENTREVISTA AL CLIENTE.

Entrevista

Fecha: 20/5/22

Razón social: La generación de reportes dinámicos.

Nombre y Apellidos: Neybis Lago Clara

1. ¿De qué se trata tu negocio?

---

---

---

2. ¿Qué funcionalidades te gustaría agregar en tu software?

---

---

---

3. ¿Tienes algún diseño o estilo existente?

---

---

---

4. ¿Qué problemas resuelve el proyecto?

---

---

---

5. ¿Cuál es el propósito del software?

---

---

---

6. ¿Qué herramientas se van a utilizar para el desarrollo del software?

---

---

---