FACULTAD 2

Módulo de Búsqueda Forense del Sistema Integral de Seguridad “Winteli”.

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

Autor: Zenia Yisney Zerquera González

Tutor: Ing. Octavio García Iribar
Cotutor: Ing. Rafael Corpas Crehuet

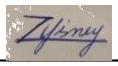
La Habana, diciembre de 2023
“Año 64 de la Revolución

DECLARACIÓN DE AUTORÍA

Se declara por este medio que Zenia Yisney Zerquera González, con carnet 99022710674, soy la autora principal del trabajo de diploma titulado Módulo de Búsqueda Forense del Sistema Integral de Seguridad Winteli y se autoriza a la Universidad de las Ciencias Informáticas a hacer uso del mismo en su beneficio, así como los derechos patrimoniales con carácter exclusivo.

Para que así conste firman la presente a los _5_ días del mes de diciembre del año 2023.

Zenia Yisney Zerquera González



_____

Firma del Autor

Ing. Octavio García Iribar

_____

Firma del Tutor

Ing. Rafael Corpas Crehuet

_____

Firma del Tutor

DATOS DE CONTACTO

Ing. Octavio García Iribar: graduado de Ingeniería en Ciencias Informáticas en la Universidad de las Ciencias Informáticas (UCI) en diciembre del 2021. Se ha desempeñado en el rol de profesor en el Departamento de Informática de la Facultad 2. Ha impartido las asignaturas de Estructuras de Datos I, Estructuras de Datos II y Estructuras de Datos y Algoritmos. No ha sido tutor después de graduado. Se preocupa por los estudiantes en tutoría, pero requiere de habilidades para hacerlo. Actualmente es profesor de programación en el departamento de Informática de la Facultad 2. **oiribar@uci.cu**

Ing. Rafael Corpas Crehuet: graduado de Ingeniería en Ciencias Informáticas en la Universidad de las Ciencias Informáticas (UCI) en julio del 2014. Se desempeña como desarrollador de software en la Empresa de Tecnologías de la Información para la Defensa (XETID), con más de 7 años de experiencia en el área de aplicaciones y tecnologías web. Ha realizado tutorías y asesorías de varios trabajos de diploma de la carrera de Ingeniería Informática. **rcorpas@xetid.cu**

AGRADECIMIENTOS

Querida familia, amigos y tutores:

Hoy quiero expresarles mi más profundo agradecimiento por todo lo que han hecho por mí. Gracias por su apoyo, su confianza, su paciencia y su amor. Gracias por estar siempre a mi lado, por creer en mí, por enseñarme y por guiarme. Gracias por compartir conmigo sus experiencias, sus consejos, sus risas y sus sueños.

Sin ustedes, no sería la persona que soy hoy. Ustedes han sido mi inspiración, mi motivación y mi fuerza. Ustedes han sido mi familia, mis amigos y mis tutores. Ustedes han sido mi luz, mi alegría y mi esperanza.

Les agradezco de todo corazón por cada momento que hemos vivido juntos, por cada lección que me han dado, por cada abrazo que me han brindado. Les agradezco por ser parte de mi vida, por enriquecerla y por hacerla más bella.

Los quiero mucho y los llevo siempre en mi corazón.

DEDICATORIA

A mi querida familia:

Quiero dedicarles mi más sincero reconocimiento y admiración. Ustedes son el regalo más grande que la vida me ha dado. Ustedes son mi apoyo, mi consuelo, mi orgullo y mi felicidad.

Gracias por estar siempre conmigo, por darme su amor incondicional, por cuidarme y protegerme. Gracias por enseñarme los valores que me han hecho crecer como persona, por darme la libertad de elegir mi camino, por respetar mis decisiones y por celebrar mis logros.

Ustedes son mi ejemplo a seguir, mi fuente de inspiración, mi razón de ser. Ustedes son mi familia, mi tesoro, mi todo.

Les dedico este texto con todo mi cariño y gratitud. Les dedico cada una de mis metas, de mis sueños, de mis proyectos. Les dedico mi vida entera, porque sin ustedes nada tendría sentido.

Los amo con toda mi alma y los llevo siempre en mi corazón.

RESUMEN

El Sistema Integral de Seguridad Winteli es una solución completa e integrada para la seguridad física de instalaciones empresariales y corporativas. Combina e integra sistemas de videovigilancia, control de acceso y detección de intrusos para brindar una seguridad completa y a medida tanto para interiores como exteriores. Las cámaras de seguridad permiten la visualización en tiempo real del local, zona, perímetro u objeto de interés mediante flujos de imágenes, lo que permite una rápida respuesta ante cualquier situación de seguridad que pueda presentarse. El sistema cuenta con un Módulo de Búsqueda Forense que permite la monitorización de la seguridad física de locales y áreas de interés a través de la emisión de alarmas por sensores, cámaras de seguridad y dispositivos de control de acceso. La solución se ha desarrollado siguiendo los pasos del proceso de desarrollo de software utilizado por la Empresa de Tecnologías de la Información para la Defensa (XETID), utilizando tecnologías como JavaScript y el framework VueJs. La solución incluye un diseño de arquitectura para la visualización del módulo de búsqueda forense en el sistema Winteli, que permite la comunicación con el servidor encargado de proveer el flujo de video de las cámaras del sistema de videovigilancia.

Palabras claves: control de acceso, integración de sistemas, módulo de búsqueda forense, monitorización, seguridad física, videovigilancia.

ABSTRACT

The Integral Security System Winteli is a complete and integrated solution for the physical security of business and corporate facilities. It combines and integrates video surveillance, access control and intrusion detection systems to provide complete and tailored security for both interiors and exteriors. Security cameras allow real-time viewing of the premises, area, perimeter or object of interest through image streams, which allows a quick response to any security situation that may arise. The system has a Forensic Search Module that allows monitoring of the physical security of premises and areas of interest through the issuance of alarms by sensors, security cameras and access control devices. The solution has been developed following the steps of the software development process used by the Defense Information Technology Company (XETID), using technologies such as JavaScript and the VueJs framework, together with the Visual Studio integrated development environment. Code. The solution includes an architectural design for the visualization of the forensic search module in the Winteli system, which allows communication with server in charge of providing the video flow from the cameras of the video surveillance system.

Keywords: access control, systems integration, forensic search module, monitoring, physical security, video surveillance

ÍNDICE

INTRODUCCIÓN 1

CAPÍTULO 1: FUNDAMENTOS TEÓRICO-METODOLÓGICOS SOBRE EL PROCESO DE BÚSQUEDA EN GRABACIONES DE CÁMARAS DE SEGURIDAD. 6

1.1 Conceptos asociados al objeto de estudio 6

1.2 Soluciones existentes de las tecnologías similares nacionales e internacionales 8

1.3 Metodología de desarrollo de software 11

1.4 Lenguajes, Tecnologías y Herramientas para el desarrollo de la propuesta solución. 16

1.4.1 Para el modelado 16

1.4.2 Para el diseño 17

1.4.3 Para la implementación 18

Conclusiones del capítulo: 22

CAPÍTULO 2: ANALISIS Y DISEÑO DE LA PROPUESTA DE SOLUCIÓN. 23

2.1 Mapa conceptual 23

2.2 Propuesta de solución 24

2.3 Planificación 25

2.3.1 Historias de Usuarios 25

2.3.2 Requisitos no funcionales 26

2.3.3 Estimación de esfuerzo por HU 27

2.3.4 Iteraciones y su plan de entrega: 27

2.4 Diseño del sistema del módulo de búsqueda forense: 29

2.4.1 Tarjetas de Clase-Responsabilidad-Colaboración (CRC) 29

2.4.2 Patrones de arquitectura de software 30

2.4.3 Patrones de diseño 30

Conclusiones del capítulo 33

CAPÍTULO 3: IMPLEMENTACIÓN DE LA SOLUCIÓN, VALIDACIÓN Y PRUEBAS SOBRE EL MÓDULO DE BÚSQUEDA FORENSE DEL SISTEMA INTEGRAL DE SEGURIDAD WINTELI. 34

3.1 Implementación de la solución 34

3.1.1 Diagrama de despliegue 34

3.1.2 Estándares de codificación 35

3.1.3 Tareas de ingeniería (TI) 36

3.2 Pruebas de software 38

3.2.1 Pruebas a nivel unitarias 38

3.2.2 Pruebas a nivel de aceptación 41

3.2.3 Prueba de integración 42

3.3 Resultados de las pruebas 43

CONCLUSIONES FINALES 45

RECOMENDACIONES 46

REFERENCIAS BIBLIOGRÁFICAS 47

ANEXO 1: HU PARA EL MÓDULO DE BÚSQUEDA FORENSE DEL SISTEMA INTEGRAL DE SEGURIDAD “WINTELI” 51

ANEXO 2: TAREAS DE INGIENERÍA (TI) PARA EL MÓDULO DE BÚSQUEDA FORENSE DEL SISTEMA INTEGRAL DE SEGURIDAD “WINTELI” 54

ANEXO 3 CASOS DE PRUEBA DE ACEPTACIÓN 57

ÍNDICE DE TABLAS

Tabla 1. 10

Tabla 2. 25

Tabla 3. 25

Tabla 4. 27

Tabla 5. 28

Tabla 6. 29

Tabla 7. 29

Tabla 8. 29

Tabla 9. 36

Tabla 10. 37

Tabla 11. 37

Tabla 12. 39

Tabla 13. 40

Tabla 14. 40

Tabla 15. 40

Tabla 16. 41

Tabla 17. 42

Tabla 18. 50

Tabla 19. 50

Tabla 20. 50

Tabla 21. 51

Tabla 22. 51

Tabla 23. 51

Tabla 24. 52

Tabla 25. 53

Tabla 26. 53

Tabla 27. 54

Tabla 28. 54

Tabla 29. 55

Tabla 30. 56

Tabla 31. 56

Tabla 32. 57

Tabla 33. 57

Tabla 34. 58

Tabla 35. 59

Tabla 36. 59

ÍNDICE DE FIGURAS

Figura 1: Fases de la metodología XP. 15

Figura 2: Mapa conceptual del Sistema “Winteli” 23

Figura 3: Representación de la propuesta de solución 25

Figura 4: Representación del patrón arquitectónico (Tomado de https://styde.net/introduccion-a-vue-js/) 30

Figura 5: Patrón GRASP Experto 31

Figura 6: Patrón GRASP Controlador 32

Figura 7: Patrón GOF Observador 33

Figura 8: Patrón GOF Estrategia 33

Figura 9: Diagrama de despliegue 34

Figura 10: Conversión de nomenclaturas 35

Figura 11: Diseño del código 36

Figura 12: Código del método pickFile 39

Figura 13: Camino básico del método pickFile 39

Figura 14: No conformidades detectadas en los casos de prueba de aceptación 43

INTRODUCCIÓN

Las formas de protección de recursos valiosos siempre han sido muy variadas y se encuentran en constante evolución, siendo la más usual y tradicional el uso exclusivo de personas que se dedican a la supervisión directa hacia el medio a proteger .De esta forma se hace necesario el empleo de mayor cantidad de fuerza laboral a medida que aumenta la zona a custodiar, además esta variante trae consigo errores de naturaleza humana los cuales pueden ser que un custodio se quede dormido o sufra algún tipo de accidente o percance. Con el surgimiento y desarrollo de nuevas tecnologías surge una nueva forma de protección llamada videovigilancia que reduce la mayoría de los errores de antiguos métodos, logrando una mayor seguridad en el ámbito donde se encuentre.

En los últimos años el desarrollo de sistemas de videovigilancia ha sido del interés de investigadores e industrias de todo el mundo. Esto ha traído consigo una constante evolución tecnológica de dichos sistemas, que además de acelerar su proliferación, ha incrementado sustancialmente las funcionalidades de los mismos prestando especial interés al análisis de video. La videovigilancia ha transcurrido por 3 diferentes generaciones de sistema en las cuales el análisis de video es llevado a cabo de diferentes maneras. La primera generación se caracteriza por ser sistemas analógicos de circuito cerrado de televisión que ofrecen escasas herramientas de análisis de video. La segunda introduce los primeros algoritmos de interpretación automática de escenas reales. Ya en la tercera generación los sistemas son completamente digitales, cuentan con cámaras IP que se instalan dentro de una red de datos, lo que ha permitido desarrollar diversas herramientas el análisis de los videos captados por cámaras de seguridad.

Entre las principales herramientas para realizar análisis de video se encuentran los video-sensores. Actualmente entre los más utilizados están los de detección de movimiento, reconocimiento facial, control vehicular, detección de objetos, entre otros. Gracias a estos se han podido desarrollar estaciones de monitoreo que interpretan sin depender del ojo humano situaciones de riesgo, planificando acciones o generando alertas al detectar eventos significativos. También han permitido desarrollar reproductores especializados para el análisis de videos almacenados. Estos permiten concentrar la atención del operador exclusivamente en información relevante, reduciendo el factor del error humano gracias a la detección automática de eventos significativos en las grabaciones. Además, con estos reproductores se pueden visualizar varios videos a la vez, realizar anotaciones en instantes específicos, capturar fotogramas relevantes, controlar parámetros como el brillo y el contraste de las grabaciones.

Las ventajas son amplias en cuanto a la reducción del personal de guardia, así como en la posibilidad de monitorizar varias áreas de forma simultánea y centralizada. Otra característica esencial es la posibilidad de grabar lo acontecido en los turnos de guardia para la posterior visualización, en busca de información relevante acerca de cualquier incidente.

En Cuba, en las últimas décadas se ha logrado un notable avance en los sistemas automáticos de vigilancia. La utilización de sistemas de seguridad avanzados ha impulsado a compañías especializadas en el desarrollo de software a elaborar productos de este tipo para el país. Son disímiles las empresas que utilizan los sistemas de videovigilancia permitiendo, sobre todo la calidad de imagen, monitoreo remoto y la seguridad. Con el uso de este sistema la cantidad de personal encargado a la custodia se reduciría considerablemente pues solo se hace necesario el empleo de personas en las estaciones de monitorización y en puntos específicos donde puedan responder rápidamente ante una alarma.

En la Universidad de las Ciencias Informáticas radica a su vez, la Empresa de Tecnologías de la Información para la Defensa (XETID), que presenta dentro de su cartera de productos y servicios un sistema de videovigilancia nombrado Winteli, desarrollado para instituciones de las fuerzas armadas y diversos sectores de la economía. Es una solución para videovigilancia inteligente con una amplia gama de analíticas de video y opciones avanzadas, que permite a las empresas que lo emplean estar alertas a su entorno, reconocer posibles amenazas y tomar medidas preventivas. El sistema constituye una herramienta de apoyo al servicio de guardia y permite a los operadores del sistema controlar el área de manera más abarcadora y precisa. Además, se encarga del monitoreo y control de varias cámaras situadas en determinadas zonas de la empresa desde un mismo punto de control.

En este sentido, se propone la siguiente **situación problemática:**

El actual sistema de videovigilancia enfrenta desafíos considerables en cuanto a la eficiencia y la resolución de incidentes delictivos pues no cuenta con elementos que permitan realizar búsquedas de eventos extraordinarios de forma automática. A pesar de recibir una gran cantidad de grabaciones de múltiples cámaras de seguridad que supervisan las diversas instalaciones, la resolución de incidentes ha resultado ser un proceso lento y laborioso. La búsqueda manual de imágenes relevantes en las grabaciones ha implicado una carga de trabajo considerable para el personal de seguridad e investigadores, afectando negativamente tanto la identificación de los

culpables como la productividad del equipo de seguridad. Entre los desafíos que presenta el Sistema Winteli están:

- Mayor riesgo de errores humanos pues al depender exclusivamente de la búsqueda manual, existe un mayor riesgo de cometer errores y pasar por alto eventos importantes o evidencias criticas en las grabaciones de video. Estos errores pueden afectar la calidad de las investigaciones y comprometer la resolución de los incidentes delictivos.
- Dificultad para identificar rápidamente a los culpables pues la falta de una búsqueda forense dificulta la identificación rápida y precisa de los culpables en los incidentes delictivos. Esto puede retrasar el proceso de investigación y limitar la capacidad de realizar un seguimiento afectivo de los sospechosos.
- Ineficiencia en la utilización de recursos pues la búsqueda manual de eventos extraordinarios y la revisión extensiva de grabaciones de video requiere una gran cantidad de recursos humanos y tiempo.
- La toma de decisiones en respuesta a incidentes delictivos se ve retrasada, la demorar en la identificación y recopilación de pruebas puede permitir que los delincuentes escapen o que se produzcan daños adicionales.
- El análisis de manual de las grabaciones de video dificulta la detección de patrones o tendencias en los incidentes delictivos.

Basándose en la problemática anterior se propone el siguiente **problema de investigación:** ¿Cómo optimizar el proceso de búsqueda de objetivos y eventos extraordinarios en las grabaciones de las cámaras de seguridad del Sistema Integral de Seguridad Winteli? Teniendo como **objeto de estudio** es el proceso de búsqueda en grabaciones de cámaras de seguridad; en tanto el **campo de acción** lo constituye el sistema de videovigilancia Winteli.

Se define como **objetivo general:** desarrollar el Módulo de Búsqueda Forense que permita optimizar y agilizar el proceso de búsqueda de eventos y objetivos en las grabaciones de las cámaras de seguridad del Sistema Integral de Seguridad Winteli.

A partir del objetivo general se determinan los **objetivos específicos** siguientes:

1. Establecer los fundamentos y referentes teórico-metodológicos para el desarrollo del Módulo de Búsqueda Forense que permita optimizar y agilizar el proceso de búsqueda de eventos y objetivos en las grabaciones de las cámaras de seguridad del Sistema Integral de Seguridad “Winteli”.
2. Diseñar la propuesta de solución que responda al desarrollo del Módulo de Búsqueda Forense que permita optimizar y agilizar el proceso de búsqueda de

eventos y objetivos en las grabaciones de las cámaras de seguridad del Sistema Integral de Seguridad “Winteli” en la Universidad de las Ciencias Informáticas.

3. Implementar el Módulo de Búsqueda Forense que permita optimizar y agilizar el proceso de búsqueda de eventos y objetivos en las grabaciones de las cámaras de seguridad del Sistema Integral de Seguridad “Winteli”.
4. Validar la solución del Módulo de Búsqueda Forense que permita optimizar y agilizar el proceso de búsqueda de eventos y objetivos en las grabaciones de las cámaras de seguridad del Sistema Integral de Seguridad “Winteli” mediante pruebas de software.

Para el desarrollo de la investigación se emplean los siguientes métodos de investigación:

Métodos teóricos

- Analítico - Sintético: se realiza un estudio de todos los aspectos relacionados a los sistemas de videovigilancia, facilitando la comprensión del tema a desarrollar y obteniendo una síntesis del análisis realizado y definiendo las características generales del sistema.
- Inductivo - Deductivo: permite integrar los métodos y funcionalidades, así como sus patrones de diseño para resolver problemas particulares de la solución en desarrollo, facilitando el análisis de los elementos generales a elementos más específicos.
- Modelación: se utiliza con el objetivo de crear modelos y diagramas que son abstracciones del producto final, permitiendo tener un dominio inicial de la información que se va a modelar, representar de forma estática los requisitos y obtener una versión del sistema original para validar los requisitos con el cliente.

Métodos Empíricos

- Entrevista: se utilizó una entrevista **no estructurada** para obtener información respecto al funcionamiento del Sistema Winteli para la gestión de monitoreo de cámaras de videovigilancia, así como requisitos funcionales y características de la propuesta de solución, a través de respuestas verbales dadas por el cliente.
- Observación: posibilitó conocer la estructura del Sistema Winteli, la monitorización de las cámaras, además de cómo funciona el sistema.

La implementación del Módulo de Búsqueda Forense para optimizar y agilizar el proceso de búsqueda en las grabaciones de cámaras de seguridad podría traer novedades significativas en términos de automatización análisis eficiente de datos, reconocimiento

de objetos y patrones, interfaz de usuario intuitiva, integración con sistemas existentes. Esto podría mejorar la capacidad de investigación forense y la eficacia de los equipos de seguridad en la búsqueda de eventos y objetivos en las grabaciones de video.

El presente trabajo está estructurado en introducción, tres capítulos, conclusiones, recomendaciones, referencias bibliográficas y anexos. La estructura capitular se describe a continuación:

Capítulo 1: Fundamentos teórico-metodológicos sobre el proceso de búsqueda en grabaciones de cámaras de seguridad: el capítulo contiene los conceptos fundamentales asociados al tema, así como un estudio del estado del arte de sistemas que permitan el monitoreo de cámaras de videovigilancia; con el fin de realizar una comparación con la solución a desarrollar. Se hace referencia a las diferentes tecnologías, lenguajes y herramientas a emplear para el desarrollo de la aplicación.

Capítulo 2: Análisis y diseño de la propuesta de solución: en este capítulo se realiza el análisis del modelo de dominio del problema y una caracterización de la solución propuesta. Se especifican los requisitos funcionales y no funcionales definidos según las necesidades de la empresa y el modelo conceptual elaborado. Se definen los conceptos fundamentales del trabajo investigativo y se diseña la arquitectura de la solución. Se definen: las historias de usuario (HU) como parte de los artefactos definidos por la metodología de desarrollo empleada. La estimación de esfuerzo de las HU permitirá elaborar el plan de entrega de las iteraciones del proyecto. Se describe la arquitectura del sistema, los patrones utilizados en el diseño y el modelo de datos como punto de partida para la implementación del sistema. Se conforman las tarjetas de clases-responsabilidad-colaboración (CRC) y se materializa la propuesta del sistema que da cumplimiento al objetivo trazado en la investigación.

Capítulo 3: Implementación de la solución, validación y pruebas sobre el Módulo de Búsqueda Forense del Sistema Integral de Seguridad Winteli: en este capítulo se aborda todo lo referente a los aspectos relacionados con la implementación de la propuesta de solución. Se documentan los principales resultados del proceso de búsqueda forense del sistema Winteli, a partir de la estrategia de prueba definida. Además, se verán los mecanismos que se utilizaron para la verificación y validación de la solución propuesta. Se observará como se llegó de lo descrito en el capítulo 1 y utilizando los métodos y técnicas científicos, al estado final de la aplicación.

CAPÍTULO 1: FUNDAMENTOS TEÓRICO-METODOLÓGICOS SOBRE EL PROCESO DE BÚSQUEDA EN GRABACIONES DE CÁMARAS DE SEGURIDAD.

En el presente capítulo se describen los conceptos asociados al proceso de búsqueda en grabaciones de cámaras de seguridad. Se documenta el resultado del estudio de soluciones existentes a nivel nacional e internacional que se corresponden con el objetivo de la investigación, para analizar las funcionalidades y características que se pueden tener en cuenta para el desarrollo de la solución propuesta. Además, se fundamenta la selección de la metodología de desarrollo de software, tecnologías y herramientas a utilizar para el desarrollo de la aplicación.

1.1 Conceptos asociados al objeto de estudio

Es amplia la bibliografía que refiere el significado de **Videovigilancia** (Requero José L. (1997) y ULL María V. (2005)). No obstante, para este estudio se asume el criterio de Klauser, (2005) que tratan el tema como un método de supervisión por imágenes para detectar y registrar hechos extraordinarios en tiempo real que pueden ser utilizadas en casos legales o de auditoría. Según Francisco Klauser (2005) *"(…) la videovigilancia es 'conservadora' y 'protectora'; conservadora porque se encarga de la conservación del orden público y de la prevención de la conducta antisocial, y protectora pues protege diferentes áreas del peligro*".

El autor Jova Rodríguez (2013) define el concepto de **Streaming**, manifestando que "(…) *es la capacidad de distribuir contenidos multimedia a través de una red digital, con la característica especial de permitir el acceso a estos contenidos según se requiera, sin necesidad de descargarlos previamente*". Las aplicaciones basadas en streaming pueden clasificarse en aquellas relacionadas con la interacción entre dos o más usuarios (como video conferencia y transmisión de voz).

Cuando la autora de esta investigación se refiere a **eventos u objetos extraordinarios** están haciendo referencia a cualquier actividad o acción que sea de carácter dudoso o de naturaleza delictiva.

La bibliografía consultada permitió resumir que hay varios autores que en este aspecto abordan el tema. No obstante, para este estudio asumimos el criterio de Sammons J, (2018) porque se ajusta a los intereses de nuestra investigación; manifestando la siguiente definición de **Búsqueda Forense** como "*(…) el uso de tecnología para buscar y analizar grabaciones de cámaras de seguridad con el fin de encontrar evidencia*

relevante para una investigación". Los productos de búsqueda forense están diseñados para acelerar las investigaciones forenses y ofrecen un control completo que permite buscar de forma eficaz objetos, personas, atributos, movimientos, incidentes y mucho más en una escena . Estos productos utilizan tecnologías como la inteligencia artificial y el aprendizaje profundo para mejorar la velocidad, precisión y usabilidad de la búsqueda .

En el ámbito informático basándose en videovigilancia existen varios tipos de búsqueda forense, entre ellos:

- **Búsqueda temporal:** se realiza para encontrar un evento específico en un período determinado de tiempo. Esta búsqueda puede realizarse utilizando herramientas de software especializadas que permiten acelerar o ralentizar la reproducción del video y saltar rápidamente a momentos específicos de la grabación.

- **Búsqueda por movimiento:** se utiliza para detectar y analizar eventos que involucren movimiento en una determinada área de vigilancia. Esta técnica es especialmente útil en situaciones en las que se desea monitorear un área extensa o cuando se requiere una vigilancia más activa de ciertas zonas de interés. La búsqueda por movimiento se realiza mediante el uso de algoritmos de detección de movimiento en el software de videovigilancia. Estos algoritmos analizan las imágenes capturadas por las cámaras de seguridad y detectan cualquier cambio en la escena, como el movimiento de personas, vehículos u objetos.

- **Búsqueda por objeto:** es una técnica avanzada que permite a los operadores de seguridad buscar y rastrear objetos específicos en las grabaciones de video. Esta herramienta utiliza algoritmos de inteligencia artificial y aprendizaje automático para identificar y seguir objetos en movimiento, como vehículos, personas o paquetes. La búsqueda por objeto es particularmente útil en situaciones en las que se necesita encontrar un objeto específico, como en el caso de robos de vehículos o paquetes perdidos. Además, esta técnica puede ayudar a los operadores de seguridad a detectar patrones de comportamiento sospechosos y prevenir incidentes antes de que ocurran.

- **Búsqueda por características físicas:** es otra técnica avanzada que permite a los operadores de seguridad buscar y rastrear personas específicas en las grabaciones de video. Esta herramienta utiliza algoritmos de reconocimiento

facial y de características físicas para identificar y seguir a personas en movimiento. Es una herramienta poderosa para la videovigilancia efectiva y la identificación temprana de individuos relevantes en tiempo real.

- **Búsqueda por patrones:** se refiere a la capacidad de un sistema de videovigilancia para detectar patrones de comportamiento anormales o sospechosos en las imágenes capturadas por las cámaras. Esto implica el uso de algoritmos de análisis de video y aprendizaje automático para identificar patrones de movimiento, actividad y comportamiento que puedan indicar una amenaza o un delito en progreso.

- **Búsqueda por análisis de comportamiento:** se refiere a la capacidad de un sistema de videovigilancia para detectar comportamientos anormales o sospechosos en las imágenes capturadas por las cámaras. Esto implica el uso de algoritmos de análisis de video y aprendizaje automático para identificar patrones de movimiento, actividad y comportamiento que puedan indicar una amenaza o un delito en progreso. La búsqueda por análisis de comportamiento es una herramienta valiosa para mejorar la eficacia de la videovigilancia y reducir el tiempo necesario para analizar grandes cantidades de imágenes de video. Además, esta técnica permite detectar comportamientos no previstos o inesperados, lo que puede ayudar a prevenir situaciones peligrosas o críticas.

Luego del análisis bibliográfico ("Python Essential Reference" de David Beazley, "Effective Java" de Joshua Bloch, "Effective C++" de Scott Meyers, "You Don't Know JS" de Kyle Simpson) podemos resumir que la **Introspección de objetos** es la capacidad de un programa para examinar las características y estructuras internas de los objetos en tiempo de ejecución. Es una capacidad que permite a un programa inspeccionar y determinar información sobre los objetos que están siendo utilizados en un momento dado.

1.2 Soluciones existentes de las tecnologías similares nacionales e internacionales

Hasta donde llega la exploración bibliográfica podemos resumir que en el ámbito internacional existen múltiples sistemas de videovigilancia que pueden ser similares a la propuesta de solución. La autora decide tomar de ejemplo el sistema VIVOTEK, HIKVISION y THALES COGENT; la información de dichos sistemas fue extraída de sus sitios web oficiales.

VIVOTEK: es un sistema que utiliza la inteligencia artificial para resolver varios problemas cotidianos de los sistemas de videovigilancia tradicionales. Utiliza conceptos como la inteligencia artificial, el aprendizaje automático, el aprendizaje profundo y las redes neuronales para mejorar la eficiencia y la eficacia de los sistemas de videovigilancia. (Tomado de web oficial de Vivotek)

Hikvision: es un sistema de videovigilancia pionero en la detección y reconocimiento automático de rostros. Este sistema utiliza tecnologías de aprendizaje profundo para ofrecer un nivel de seguridad completamente nuevo. (Tomado de web oficial de Hikvision)

Thales Cogent: es un sistema de videovigilancia que ofrece una solución de reconocimiento facial biométrico de última generación llamada FRP (Facial Recognition Platform). Su algoritmo de clase mundial basado en redes neuronales profundas garantiza la eficiencia y precisión para la detección, el seguimiento y el reconocimiento de rostros. (Tomado de web oficial de Thales Cogent)

Hasta donde llega la exploración bibliográfica podemos resumir que en el ámbito nacional existen múltiples sistemas de videovigilancia que pueden ser similares a la propuesta de solución. La autora decide tomar de ejemplo el sistema XYMA SAFE VISION, XILEMA SURIA.

Xyma Safe Vision: es un software de videovigilancia profesional basado en tecnología IP con un alto grado de modularidad, adaptable, flexible y escalable, que permite el uso de tecnologías analógicas. Administra la seguridad monitorizando y controlando en tiempo real y de forma histórica cada uno de los movimientos que ocurren en las áreas sensibles que se identifiquen. Las posibilidades del sistema están distribuidas para darle flexibilidad y su arquitectura modular lo convierte en un potente sistema que permite administrar, monitorizar, grabar, revisar, configurar alertas y alarmas y obtener reportes para cualquier solución de videovigilancia que se requiera implementar. (Tomado de web oficial de Datys)

Xilema Suria: constituye una plataforma profesional para la gestión de videovigilancia en cualquier entorno necesitado de seguridad y protección. Con un alto grado de escalabilidad y adaptabilidad representa el apoyo ideal en el resguardo de personas, inmuebles, oficinas, establecimientos públicos y otros bienes. El producto se concibe sobre la base del desarrollo de un sistema de videovigilancia soportado sobre tecnología IP, como parte de un circuito cerrado de televisión. (Tomado de web oficial de Gladiadores)

Es importante tener en cuenta que, si se requiere una búsqueda forense más avanzada, es posible que sea necesario invertir en sistemas de videovigilancia más sofisticados que incluyan módulos de búsqueda forense o utilizar software adicional para analizar y buscar el contenido de los videos capturados.

Para comprar los sistemas mencionados anteriormente se utilizarán los siguientes parámetros:

Módulo Búsqueda Forense: se refiere al uso de herramientas de análisis de datos y videos para realizar búsquedas más efectivas y certeras.
Accesibilidad: se refiere a si se puede tener acceso al sistema estando en Cuba.
Compatibilidad: se refiere a la capacidad de agregar plugins que te permitan asociar cámaras de otros fabricantes.
Multiplataforma: se refiere a la capacidad de poder ejecutarse en más de un sistema operativo.
Código abierto: se refiere a que sea un sistema sin licencia que permita acceder a su código fuente para modificarlo y así adaptarlo a las necesidades del cliente.

Tabla 1. Comparación entre los sistemas de videovigilancia

Criterios de comparación	**Módulo Búsqueda Forense**	**Accesibilidad**	**Compatibilidad**	**Multiplataforma**	**Código abierto**
VIVOTEK	Sí	Sí (pago)	No	Sí	No
Hikvision	Sí	Sí (pago)	No	Sí	No
Thales Cogent	Sí	Sí (pago)	No	Sí	No
Safe Vision	Sí	Sí	Sí	No (Windows)	No
Xilema Suria	Sí	Sí	Sí	No (Windows)	No
Winteli	Sí	Sí	Sí	Sí	Sí

Al comparar los sistemas anteriores respecto a los indicadores en la tabla1 y debido a que el tema de la videovigilancia es un tema sensible en cuanto a confidencialidad de la información no se pudo realizar un estudio a fondo de cada uno de los sistemas, ya que todos ellos son privativos de código cerrado y sus sitios web no ofrecen mucha información, se determinó que ninguno de los sistemas similares satisface la solución al problema dado en la investigación, por lo tanto, el Módulo de Búsqueda Forense será una aplicación necesaria para el proceso de optimización de las búsquedas de objetivos

y eventos extraordinarios en las grabaciones de las cámaras de seguridad. Además, el sistema que se creará al terminar la investigación, será capaz de adaptarse a las condiciones tecnológicas del país y cumplirá positivamente los indicadores expuestos anteriormente.

1.3 Metodología de desarrollo de software

Los autores Maida & Pacienzia (2015) definen una metodología como "(…) *un conjunto integrado de técnicas y métodos que permite abordar de forma homogénea y abierta cada una de las actividades del ciclo de vida de un proyecto de desarrollo. Es un proceso de software detallado y completo. Las metodologías se basan en una combinación de los modelos de proceso genéricos. Definen artefactos, roles y actividades, junto con prácticas y técnicas recomendadas*".

En la actualidad existen una gran cantidad de metodologías para el desarrollo de software, separadas en dos grandes grupos; las metodologías tradicionales o pesadas y las metodologías ágiles. Las metodologías tradicionales se basan en las buenas prácticas dentro de la ingeniería del software, siguiendo un marco de disciplina estricto y un riguroso proceso de aplicación. Las metodologías ágiles, en cambio, representan una solución a los problemas que requieren una respuesta rápida en un ambiente flexible y con cambios constantes, haciendo caso omiso de la documentación rigurosa y los métodos formales.

Metodologías tradicionales:

Según los autores Maida & Pacienzia (2015) "*(…) Las metodologías tradicionales son denominadas como metodologías pesadas. Centran su atención en llevar una documentación exhaustiva de todo el proyecto, la planificación y control del mismo, en especificaciones precisas de requisitos y modelado y en cumplir con un plan de trabajo, definido todo esto, en la fase inicial del desarrollo del proyecto*". Estas metodologías tradicionales imponen una disciplina rigurosa de trabajo sobre el proceso de desarrollo del software, con el fin de conseguir un software más eficiente. Para ello, se hace énfasis en la planificación total de todo el trabajo a realizar y una vez que está todo detallado, comienza el ciclo de desarrollo del producto software.

Las principales metodologías tradicionales son:

- Cascada
- Prototipo
- Espiral
- Incremental

Estas son algunas de las ventajas que debemos tener en cuenta:

- Usa una estructura clara con una serie de pasos definidos.
- Compromiso con el producto final establecido desde el principio del proyecto.
- Estimación de calendarios y presupuestos con mayor precisión al comenzar el proyecto.
- Prioriza la documentación de la información a lo largo de todo el ciclo de vida del proyecto.
- Requiere menos coordinación debido a que los procesos son secuenciales, con fases claramente definidas.

A continuación, se presentan las principales desventajas a las que se enfrenta con el uso de las metodologías tradicionales:

- Hay riesgo de pérdida de tiempo, esfuerzo y dinero debido a retrasos y contratiempos durante las transiciones de fase a fase.
- Retrasa las pruebas hasta después de la entrega del producto.
- El cliente o el usuario final no se integra en el proceso de producción activamente hasta la finalización del proyecto.
- Es más difícil dividir y compartir el trabajo debido a que las secuencias de fases son más estrictas y los equipos están más especializados.
- El método no es apropiado para los proyectos en los que se sabe desde inicio que hay muchas probabilidades que los requisitos cambien.

Metodologías ágiles:

Para los autores Maida & Pacienzia (2015) las metodologías ágiles "*(…) nacen como respuesta a los problemas que puedan ocasionar las metodologías tradicionales y se basa en dos aspectos fundamentales, retrasar las decisiones y la planificación adaptativa. Basan su fundamento en la adaptabilidad de los procesos de desarrollo. Un modelo de desarrollo ágil, generalmente es un proceso Incremental, también Cooperativo, Sencillo y finalmente Adaptativo. Las metodologías ágiles proporcionan una serie de pautas y principios junto a técnicas pragmáticas que hacen que la entrega del proyecto sea menos complicada y más satisfactoria tanto para los clientes como para los equipos de trabajo, evitando de esta manera los caminos burocráticos de las metodologías tradicionales, generando poca documentación y no haciendo uso de métodos formales"*.

Las principales metodologías ágiles son:

- Kanban
- Scrum
- Lean

- Programación extrema (XP)

Las ventajas a tener en cuenta para la selección de una metodología ágil son:

- Mayor rapidez y eficiencia, y menores costes.
- Mayor compromiso, motivación e implicación del equipo.
- Aumento de la productividad.
- Se establecen prioridades flexibles.
- Mejora la calidad final.

También se deben tener en cuenta las desventajas que presentan este tipo de metodologías:

- Menor planificación concreta debido a la constante repriorización de las tareas y su plazo de ejecución.
- Los equipos ágiles suelen ser pequeños, por lo que los miembros del equipo deben estar altamente capacitados en una variedad de áreas y procesos.
- Se requiere participación activa y colaboración constante de los miembros del equipo durante todo el proceso, lo que consume más tiempo que un enfoque tradicional.
- El producto final puede diferir del propuesto, ya que se pueden agregar iteraciones, los comentarios de los clientes pueden alterar los planes y los plazos pueden cambiar.
- La documentación puede descuidarse, ya que se prefieren los entregables del trabajo a la documentación completa del proyecto.

Selección de la metodología

Luego de un análisis detallado de las metodologías antes mencionadas y teniendo en cuenta las ventajas y desventajas encontradas se decide utilizar XP, pues su tasa de error es muy pequeña. También se puede aplicar a cualquier lenguaje de programación, el cliente tiene el control sobre las prioridades, durante todo el proyecto se pueden realizar diversas pruebas y, sobre todo, permite ahorrar mucho tiempo.

Hasta donde llega la exploración bibliográfica podemos resumir que para la implementación del Módulo de Búsqueda Forense se usará la metodología XP; pues esta se centra más en las buenas prácticas de desarrollo de software que promueven la excelencia técnica y la creación de un software de alta calidad. Por lo anteriormente expuesto se toma la decisión de que esta sea la metodología a utilizar ya que es un proyecto relativamente pequeño, de poca duración, se tiene un equipo de desarrollo de dos personas, y el cliente es parte de la investigación. XP es el más destacado de los

procesos ágiles de desarrollo de software, los cambios de requisitos sobre la marcha son un aspecto natural, apoya la integración del equipo de programación con el cliente o usuario, consta de una alta gama de bibliografía, sus artefactos presentan una descripción del comportamiento del sistema fácil de entender y teniendo en consideración la integración de la propuesta de solución al sistema base al que se desea añadir.

XP (Extreme Programming) es una metodología ágil de desarrollo de software que se enfoca en la entrega rápida y continua de desarrollo de software de alta calidad. XP se basa en cuatro valores fundamentales: comunicación, simplicidad, retroalimentación y coraje.

Luego de un análisis detallado podemos decir que la metodología XP es un enfoque de la ingeniería del software formulado por Kent Beck. Según el autor Intecto, (2009) cuando habla sobre esta metodología lo plantea como "(…) *el más destacado de los procesos ágiles de desarrollo de software y al igual que éstos, la programación extrema se diferencia de las metodologías tradicionales principalmente en que pone más énfasis en la adaptabilidad que en la previsibilidad*". Los defensores de XP creen, que puede ser capaz de adaptarse a los cambios de requisitos en cualquier punto de la vida del proyecto es una aproximación mejor y más realista que intentar definir todos los requisitos al comienzo del proyecto e invertir esfuerzos después en controlar los cambios en los requisitos.

A continuación, se muestran las características de la metodología XP teniendo en cuenta los autores Bustamante & Rodríguez, (2014)**:**

- Pone más énfasis en la adaptabilidad que en la previsibilidad.
- Se aplica de manera dinámica durante el ciclo de vida del software.
- Es capaz de adaptarse a los cambios de requisitos.
- Los individuos e interacciones son más importantes que los procesos y herramientas.
- Software que funcione es más importante que documentación exhaustiva.
- La colaboración con el cliente es más importante que la negociación de contratos.
- La respuesta ante el cambio es más importante que el seguimiento de un plan.

Esta metodología contiene **4 fases**: Planificación, Diseño, Desarrollo, Pruebas. Las cuales se desarrollan de la siguiente forma:

Planificación es la fase en la que se establecen los objetivos específicos del proyecto, se priorizan las tareas y se crea un plan de entrega iterativo. También se establecen las pruebas y los criterios de aceptación para cada tarea. Esta fase genera los siguientes artefactos: historia de usuario, requisitos no funcionales, plan de iteraciones, plan de entrega.

Diseño es la fase en la que se crea un diseño de alto nivel del sistema, que incluye la arquitectura y la estructura general. También se definen los casos de uso y se crean diagramas de secuencia para cada tarea. Esta fase genera los siguientes artefactos: patrón arquitectónico, tarjetas CRC y patrones de diseño: GRASP Y GOF.

Desarrollo es la fase encargada de escribe el código para cada tarea, siguiendo los principios de simplicidad y claridad. También se realizan pruebas unitarias para asegurarse de que el código funciona como se espera. Esta fase genera el artefacto: tareas de ingeniería.

Pruebas se encarga de realizar las pruebas de integración y pruebas de aceptación para asegurarse de que el software cumple con los requisitos del cliente. También se realizan pruebas de rendimiento y seguridad según sea necesario. Esta fase genera los siguientes artefactos: pruebas unitarias y pruebas de aceptación.

En el presente trabajo de diploma se desarrollan las fases de planificación, diseño, desarrollo, y pruebas además se transita por las siguientes disciplinas de la metodología: Programación en Pareja, Pruebas automatizadas, Integración continua, Diseño simple y Cliente presente.

Figura 1: Fases de la metodología XP.

1.4 Lenguajes, Tecnologías y Herramientas para el desarrollo de la propuesta solución.

En el modelado de la propuesta de solución se utilizó el siguiente lenguaje, tecnología y herramienta, descritos a continuación:

1.4.1 Para el modelado

En relación a este tema es varia la bibliografía encontrada de autores extranjeros. Los consultados manifiestan que desempeña un rol importante no solo en el desarrollo de software, sino también en los sistemas que no tienen software en muchas industrias, ya que es una forma de mostrar visualmente el comportamiento y la estructura de un sistema o proceso. El **UML** ayuda a mostrar errores potenciales en las estructuras de aplicaciones, el comportamiento del sistema y otros procesos empresariales. (*La guía sencilla para la diagramación de UML y el modelado de la base de datos*, s. f.)

Herramienta Visual Paradigm: Visual Paradigm es una herramienta CASE (*Computer Aided Software Engineering*, Ingeniería de Software Asistida por Computadoras) que emplea UML como lenguaje de modelado. Propicia un conjunto de ayudas para el desarrollo de programas informáticos a través de la representación de todo tipo de diagramas. Está concebido para soportar el ciclo de vida completo del proceso de desarrollo de software. Ofrece un lenguaje estándar común a todo el equipo de desarrollo, lo cual facilita considerablemente la comunicación. Permite la generación de código fuente y bases de datos mediante la transformación de diagramas de Entidad-Relación en tablas de base de datos. Se utilizará la versión 16.2 (Visual Paradigm International)

La autora decide seleccionar esta herramienta para realizar los diagramas porque es la que más se ajusta a las necesidades de esta investigación y por las siguientes razones:

- Presenta licencia gratuita.
- Es fácil de instalar y actualizar y compatible entre ediciones.
- Soporta todos los diagramas de UML.
- Generación de código.
- Editor de figuras.
- Disponibilidad en múltiples plataformas: Microsoft Windows, Linux, Mac OS X, Solaris o Java.

1.4.2 Para el diseño

Existen distintos tipos de lenguaje: principalmente de bajo nivel y de alto nivel. La diferencia se encuentra en lo cerca o lejos que estemos del hardware de nuestro equipo.

Esta cercanía tiene que ver con el control que tengamos sobre el dispositivo, placa o controlador.(*¿Qué es un lenguaje de programación?* | *Desarrollar Inclusión*, s. f.)

HTML5: Definiéndolo de forma sencilla, para el autor Eguiluz, (2018) "*(…) HTML es el lenguaje con el que se escriben la mayoría de páginas web de Internet. Además, es un estándar reconocido en todo el mundo y cuyas normas define un organismo sin ánimo de lucro llamado World Wide Web Consortium, más conocido como W3C. Como se trata de un estándar reconocido por todas las empresas relacionadas con el mundo de Internet, una misma página HTML se visualiza de forma muy similar en cualquier navegador de cualquier sistema operativo".*

El lenguaje HTML permite:

- Permite describir hipertexto
- Tiene un despliegue rápido
- Lo reconoce y admite cualquier tipo de explorador
- Permite archivos pequeños.

Por las razones mencionadas anteriormente, y porque es el que más se ajusta a las necesidades de esta investigación, los autores han decidido su empleo.

CSS3: Según el autor Eguiluz, (2018) *"(…) Es un lenguaje de hojas de estilos en cascadas creado para controlar el aspecto o presentación de los documentos electrónicos definidos con HTML y XHTML. CSS es la mejor forma de separar los contenidos de su presentación y es imprescindible para crear páginas web complejas. Además, mejora la accesibilidad del documento, reduce la complejidad de su mantenimiento y permite visualizar el mismo documento en infinidad de dispositivos diferentes".*

El lenguaje CSS facilita el trabajo con el diseño de las páginas web y porque es el que más se ajusta a las necesidades de esta investigación, la autora de esta investigación lo seleccionan por las siguientes razones:

- Si se necesita hacer modificaciones visuales se hacen en un sólo lugar. No es necesario editar todo el HTML en cuestión por separado.
- Se reduce la duplicación de estilos en diferentes lugares, por lo que es más fácil de organizar y hacer cambios. Además, al final la información a transmitir es considerablemente menor.

Es más fácil crear versiones diferentes de presentación para otros tipos de dispositivos como tablets, smartphones o dispositivos móviles.

Vue.js: Según el autor You, E. (2014) lo define como *"(…) un marco de trabajo (framework) JavaScript versátil que se puede utilizar tanto como una librería como un marco de*

trabajo completo, diseñado para simplificar y agilizar el desarrollo de aplicaciones web interactivas". En su capacidad como librería, Vue.js proporciona una serie de herramientas y funcionalidades que permiten la creación de componentes web reutilizables y la incorporación de interactividad en páginas web de manera modular. Esto lo convierte en una opción flexible para desarrolladores que desean mejorar la experiencia del usuario en proyectos web existentes sin tener que reconstruir por completo la aplicación. Vue.js se destaca por su capacidad de manipulación del DOM, su sistema de reactividad y su enfoque en la creación de interfaces de usuario dinámicas. Para el desarrollo de la propuesta de solución se utilizará como librería en la versión 2.

Es un modelo de reactividad, lo que significa que los cambios realizados en los datos se reflejan automáticamente en la interfaz de usuario si la necesidad de actualizar manualmente. Esto agiliza el desarrollo y mejora la experiencia del usuario.

Características de Vue.js:

- Proporciona enlaces de datos bidireccionales lo que permite sincronozar automáticamente los datos entre el modelo y la vista. Esto facilita la actualización automática de la interfaz de usuario cuando los datos cambian.
- Puedes describir cómo se debería ver tu aplicación en diferentes estados y se encargará de actualizar la interfaz de usuario según corresponda.
- Promueve la creación de componentes reutilizables. Puedes encapsular bloques de código y estilos en componentes individuales y luego reutilizarlos fácilmente en diferentes partes de tu aplicación.
- Proporciona directivas personalizadas que te permiten extender el comportamiento de HTML básico. Algunas directivas populares incluyen v-bind, v-if, v-for, etc.

Por las razones mencionadas anteriormente, y porque es el que más se ajusta a las necesidades de esta investigación, la autora ha decidido su empleo.

1.4.3 Para la implementación

Para el autor Hipp, D. R. (2000) "***SQLite*** *es un sistema de gestión de base de datos relacional que se encuentra incorporado en la mayoría de los sistemas operativos y lenguaje de programación. A diferencia de otros sistemas de gestión de base de datos, SQLite no funciona como un proceso independiente, sino que integra directamente en la aplicación que lo utiliza".* Se utilizará la versión 3

Es una base de datos liviana y el código abierto que no requiere un servidor separado para funcionar, lo que significa que puede ser utilizado en aplicaciones o embebidas en

dispositivos móviles. Es ampliamente utilizado en aplicaciones móviles, navegadores web y otros programas que necesitan almacenar y acceder a datos de forma eficiente.

En resumen, es una base de datos relacional ligera y fácil de usar que se utiliza ampliamente en aplicaciones móviles y otros programas para almacenar y acceder a datos de manera eficiente.

La autora ha decidido utilizar este gestor de bases de datos porque es el que más se ajusta a las necesidades de esta investigación y por las siguientes razones:

- Es estable, multiplataforma y compatible con versiones anteriores.
- Su código es de dominio público y gratuito.
- No requiere instalación o configuración.
- Guarda la base de datos en un solo archivo.

Node.js: El autor Dahl, R. (2009) lo define como *"(…) un entorno de ejecución de JavaScript que te permite ejecutar código JavaScript fuera del navegador. Es ideal para crear aplicaciones el lado del servidor y manejar eventos de entrada/salida de manera eficiente. Utiliza moto V8 de Google Chome y se basa en un modelo de operaciones de entrada/salida no bloqueante y orientado a eventos, lo que lo hace muy eficiente y escalable"*. Se utilizará la versión 18.14.2

Características de Node.js:

- Alta velocidad y rendimiento: Debido a que se basa en el motor V8, Node.js es muy rápido y eficiente en la ejecución de código JavaScript.
- Manejo de eventos: utiliza un sistema de eventos que permite manejar múltiples solicitudes de manera no bloqueante y así lograr un alto rendimiento y bajo consumo de recursos.
- API basado en JavaScript: al utilizar este lenguaje tanto en el lado del cliente como en el del servidor, puedes compartir código y lógica entre ambos, lo que facilita el desarrollo y manteniento de aplicaciones.
- Amplia comunidad y ecosistema: cuenta con una gran comunidad de desarrolladores y una amplia variedad de módulos y bibliotecas disponibles a través de npm (Node Package Manager).
- Escalabilidad: su arquitectura es muy adecuada para aplicaciones escalables, ya que su modelo de operaciones no bloqueantes permite manejar una gran cantidad de conexiones simultáneas sin afectar el rendimiento.

Por las razones mencionadas anteriormente, y porque es el que más se ajusta a las necesidades de esta investigación, la autora ha decidido su empleo.

Visual Studio Code: Es un editor de código fuente desarrollado por Microsoft, Es una aplicación gratuita y de código abierto que proporciona un entorno de desarrollo integrado (IDE) ligero y altamente personalizable para escribir, editar y depurar código en varios lenguajes de programación. Se utilizará en la versión 1.83.1 (Tomado del sitio oficial)

La autora decide seleccionar esta herramienta para realizar los diagramas porque es la que más se ajusta a las necesidades de esta investigación y por las siguientes razones:

- Posee una capacidad de adaptación a las necesidades y preferencias de los desarrolladores.
- Edición de código
- Depuración
- Integración con el control de versiones
- Extensibilidad
- Configuración personalizada

JavaScript: Es un lenguaje de secuencias de comandos que te permite crear contenido de actualización dinámica, controlar multimedia, animar imágenes y prácticamente todo lo demás. Es un robusto lenguaje de programación que se puede aplicar a un documento HTML y usarse para crear interactividad dinámica en los sitios web.
Según el autor Eguiluz, (2018) lo define como *"(…) un lenguaje de programación que se utiliza principalmente para crear páginas web dinámicas. Técnicamente, JavaScript es un lenguaje de programación interpretado, por lo que no es necesario compilar los programas para ejecutarlos. En otras palabras, los programas escritos con JavaScript se pueden probar directamente en cualquier navegador sin necesidad de procesos intermedios"*. A pesar de su nombre, JavaScript no guarda ninguna relación directa con el lenguaje de programación Java.

Entre las múltiples ventajas que ofrece JavaScript la autora de esta investigación lo seleccionan por las siguientes razones:

- JavaScript tiende a ser muy rápido porque a menudo se ejecuta inmediatamente en el navegador.
- La sintaxis de JavaScript está inspirada por Java y es relativamente sencillo de aprender comparado con otros lenguajes de programación
- JavaScript puede ser usado para crear características como arrastrar y soltar, y componentes tales como las diapositivas, lo cual mejora enormemente la interfaz de usuario y la experiencia del sitio.

Axios: es una librería JavaScript que se utiliza para realizar solicitudes HTTP desde el navegador o desde Node.js. Se puede utilizar para realizar solicitudes tanto de manera asíncrona como síncrona y es muy comúnmente utilizada en aplicaciones web para interactuar con servicios web y API. (Tomado de sitio web oficial)

Entre las múltiples ventajas que ofrece Axios la autora de esta investigación lo seleccionan por las siguientes razones:

- Sencillez de Uso: Axios proporciona una interfaz simple y fácil de usar para realizar solicitudes HTTP.
- Compatibilidad con Promesas: Utiliza promesas, lo que facilita la gestión de solicitudes asíncronas y el manejo de respuestas.
- Interceptores: Permite el uso de interceptores para transformar las solicitudes o respuestas antes de que sean enviadas o después de que son recibidas.
- Manejo Automático de JSON: Axios automáticamente convierte las respuestas JSON a objetos JavaScript, simplificando el manejo de datos.
- Soporte para Cancelación de Solicitudes: Proporciona una funcionalidad de cancelación para abortar solicitudes en curso cuando sea necesario.

Janus: es un servidor de transmisión en tiempo real (gateway) de código abierto. WebRTC, que significa "Web Real-Time Communication", es una tecnología que permite la comunicación en tiempo real directamente en los navegadores web sin necesidad de plugins. Se especializa en ofrecer servicios para aplicaciones que requieren transmisión de medios en tiempo real, como videoconferencias, chat de video, y otras formas de comunicación en línea. (Miniero, L, 2014)

Entre las múltiples ventajas que ofrece Janus la autora de esta investigación lo seleccionan por las siguientes razones:

- Gateway WebRTC: Janus actúa como un servidor intermediario que permite la comunicación en tiempo real entre diferentes clientes a través de WebRTC.
- Extensibilidad: Janus está diseñado de manera modular y es altamente extensible. Puedes agregar funcionalidades específicas según las necesidades de tu aplicación.
- API RESTful: Proporciona una interfaz de programación de aplicaciones (API) basada en HTTP que permite el control y la configuración del servidor.
- Soporte para Múltiples Flujo de Medios: Permite manejar múltiples flujos de medios (audio, video, datos) simultáneamente.
- Implementación de Plugins: Puedes extender las capacidades de Janus mediante la implementación de plugins específicos para tu caso de uso.

Conclusiones del capítulo:

Este capítulo se demostró la eficiencia del empleo de los métodos de investigación científica seleccionados al permitir construir el marco teórico de esta investigación. Luego de realizar un estudio del estado del arte se llega a la conclusión de que ninguno cumplía todas las necesidades que requiere el problema planteado. Se proponen herramientas, lenguajes y tecnologías para el desarrollo del Módulo de Búsqueda Forense. El uso de metodologías ágiles, en este caso XP fue la seleccionada y permite disponer de una mayor variación que pueden originarse a lo largo del proyecto, una mejora de la calidad y la posibilidad de presentar novedades sobre el producto que se encuentra en desarrollo. Además, se hizo un análisis de los sistemas homólogos existentes lo que permitió tener mayor claridad y referencia a la hora de construir la propuesta de solución.

CAPÍTULO 2: ANALISIS Y DISEÑO DE LA PROPUESTA DE SOLUCIÓN.

En este capítulo se aborda lo referente al análisis y diseño del Módulo de Búsqueda Forense del Sistema Integral de Seguridad Winteli. Se define la propuesta de solución a través de la descripción de su funcionamiento y de los principales aspectos que permiten una mejor comprensión del mismo. Se muestran las características y funcionalidades del sistema a implementar. Se transita por cada una de las fases que propone la metodología XP, exponiendo así las historias de usuarios (HU) que se encuentran en la aplicación y la estimación de su esfuerzo.

2.1 Mapa conceptual

A continuación, se presenta un mapa conceptual para explicar como funciona el Sistema Integral de Seguridad “Winteli”:

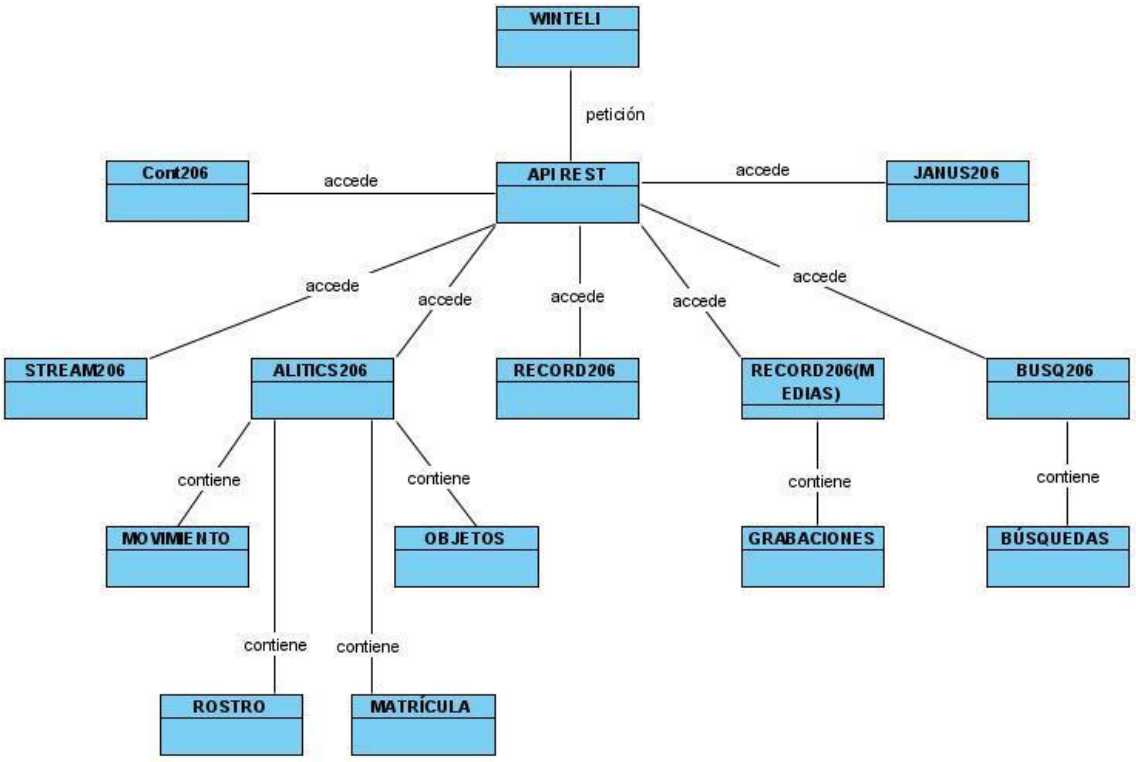


Figura 2: Mapa conceptual del Sistema “Winteli”

El Sistema Winteli realiza peticiones de búsquedas a los servidores a través del API REST, este último es el encargado de acceder a los distintos servidores (Cont206, Stream206, Alitics206, Record206, Record206(Media), busq206, janus206). Dependiendo de la petición que el usuario realiza es el acceso al flujo de datos de los servidores. El servidor **Cont206** se encarga del monitoreo y control de las cámaras, el servidor Stream206 se encarga de proporcionar vídeo en directo o a la carta a dispositivos con conexión a Internet, el servidor de **Alitics206** es el que contiene las

configuraciones de las analíticas necesarias para realizar las búsquedas(detección movimiento, reconocimiento facial(rostro), detección de matrícula, detección de objetos), el servidor de **Record206** se encarga de mostrar las grabaciones para su posterior revisión y este permite la posibilidad de descargar los videos de grabaciones, el servidor de **Record206(media)** se encarga almacenar las grabaciones para su posterior análisis y procesamiento, el servidor **Busq206** se encarga de realizar la búsqueda forense teniendo en cuenta los parámetros requeridos por cada búsqueda y la necesidad del usuario, el servidor de **Janus206** contiene las librerías necesarias para realizar la conexión y compatibilidad con los demás servidores haciendo uso de la introspección de objetos. Luego de realizada la petición al API y este haya realizado las gestiones necesarias recibe la información que posteriormente envía al sistema para que muestre el resultado al usuario.

2.2 Propuesta de solución

En respuesta a los desafíos identificados en el capítulo anterior, se propone una solución integral: la creación del módulo de búsqueda forense. Este módulo se implementará en el Sistema Integral de Seguridad Winteli, con el propósito de optimizar el proceso de búsqueda de objetivos y eventos extraordinarios en las grabaciones de las cámaras de seguridad. La finalidad de esta propuesta es permitir búsquedas específicas y precisas en las grabaciones de video, lo que se traducirá en una mejora significativa en la eficiencia global del sistema de videovigilancia; pues minimizará la posibilidad de errores humanos, facilitará la gestión de incidentes en tiempo real al permitir búsquedas rápidas de eventos sospechosos y situaciones de seguridad, los investigadores podrán definir criterios de búsqueda detallados. El cliente a través del Sistema Winteli le hace una petición al API REST y este a su vez accede al servidor de búsqueda forense que tiene configurado las analíticas de detección de movimiento, reconocimiento facial (rostro), identificación de matrícula y detección de objetos. Este servidor se conecta simultáneamente al servidor de media que almacena las grabaciones de las cámaras. Luego la información es enviada al API para que este se encargue de mostrarla en la interfaz visual del Módulo de Búsqueda Forense.

La implementación de este módulo de búsqueda forense representa una inversión en la eficiencia y la efectividad del sistema de seguridad, lo que se traducirá en una mejor protección de las instalaciones, la mitigación de riesgos y una respuesta más ágil ante incidentes.

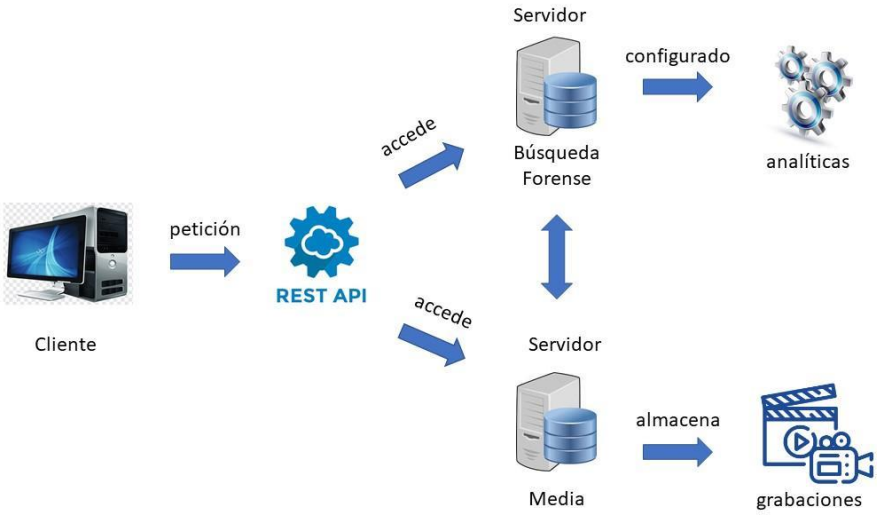


Figura 3: Representación de la propuesta de solución

2.3 Planificación

2.3.1 Historias de Usuarios

La plantilla a utilizarse para la elaboración de las historias de usuario se muestra en la tabla siguiente, con cada uno de sus componentes explicados y siendo elaborada a juicio propio de la autora de la investigación.

Tabla 2. Plantilla de elaboración de historias de usuario

Historia de Usuario	
Número: identificador de la historia de usuario	**Nombre Historia**: Nombre de la historia de usuario:
Prioridad: Nivel de prioridad en el negocio (Alto, Medio, Bajo)	**Iteración asignada**: Número de la iteración en la que será desarrollada
Estimación: aproximación del esfuerzo necesario (en tiempo ideal) para implementar la historia de usuario	
Programador: Persona encargada de programar cada historia de usuario	
Riesgo en desarrollo: riesgo técnico o funcional asociado a la implementación de la historia de usuario (Alto, Medio, Bajo)	
Descripción: síntesis de la historia de usuario.	

A continuación, se muestra la HU Listar cámaras. El resto pueden consultarse en los anexos. **Ver Anexo 1 Historia de Usuarios.**

Tabla 3. Historia de usuario "Listar cámaras"

Historia de Usuario

Número: 1	**Nombre Historia**: Listar cámaras
Prioridad: Alto	**Iteración asignada**: 1
Estimación: 1	
Programador: Zenia Zerquera González	
Riesgo en desarrollo: Alto	
Descripción: Yo quiero que me muestre una lista de todas las cámaras que tengan configurada las analíticas de la búsqueda forense, pero que sea en un panel en la parte izquierda de la pantalla.	

2.3.2 Requisitos no funcionales

RNF 1. Software

RNF 1.1. El sistema es multiplataforma, tanto Windows como Ubuntu en su versión 16.04.

RNF 1.2. Navegador con soporte de WebRTC h264, Chromium, Chrome, Opera, Yadex.

RNF 2. Hardware

RNF 2.1. Las prestaciones para el servidor de búsqueda forense serán 8GB de memoria RAM (mínimo), capacidad de almacenamiento 80GB (mínimo), capacidad de procesamiento 3.8GHZ, capacidad de transmisión de datos 100MiB/s.

RNF 2.2. Las prestaciones para la PC Cliente serán 4GB de memoria RAM (mínimo), capacidad de almacenamiento 60GB (mínimo), capacidad de procesamiento 2.6GHZ, capacidad de transmisión de datos 100MiB/s.

RNF 3. Usabilidad

RNF 3.1. Expresarse en un lenguaje de fácil comprensión.

RNF 3.2. La interfaz deberá mostrar todas sus funcionalidades de manera organizada.

RNF 4. Seguridad

RNF 4.1. El sistema debe comunicarse usando un protocolo seguro, tendrá un administrador del sistema, que otorgará los privilegios y permisos a los usuarios determinados.

RNF 4.2. Se implementarán políticas de resguardo de información, así como la realización de copias periódicas de seguridad, que permitan restaurar el sistema en caso de fallo crítico o pérdida total de la información.

RNF 4.3. El sistema debe mostrar las funcionalidades de acuerdo a los permisos del usuario que esté activo.

RNF 4.4. Solo el administrador puede cambiar las configuraciones del sistema.

RNF 5. Fiabilidad

RNF 5.1. La información estará disponible todo el tiempo para los usuarios del sistema.

RNF 6. Rendimiento

RNF 6.1. El tiempo de respuesta debe ser menor que un minuto.

RNF 6.2. Estará implementado sobre una tecnología Web, facilitando su uso a través de la red.

2.3.3 Estimación de esfuerzo por HU

Después de definir las HU, se realiza una planificación del tiempo que se necesita para implementarlas, las estimaciones de esfuerzo asociado a la implementación de las historias la establecen los programadores utilizando como medida el punto. Un punto, equivale a una semana ideal de programación. La jornada laboral será de 8 horas y se trabajaran 5 días por semana.

Tabla 4. Estimación de esfuerzo por Historia de usuario

Nombre	Puntos estimados
Listar cámaras	1
Búsqueda de movimiento (ancho_maximo, ancho_minimo, altura_maxima, altura_minima)	1
Búsqueda rostro (similitud, imagen)	1
Búsqueda matrícula (similitud, matricula)	1
Búsqueda objeto (similitud, listado_objeto)	1
Filtrar la búsqueda (fecha_inicio, fecha_fin, hora_inicio, hora_fin)	1
Mostrar video	3
Reproducir video	3
total	**12**

Sumando los puntos de estimación de esfuerzo de cada una de las HU presentes en la investigación, se calcula que el desarrollo de la aplicación será de 12 semanas.

2.3.4 Iteraciones y su plan de entrega:

Iteración 1: Durante esta iteración, se enfocará en la implementación de Historias de Usuario de alta prioridad relacionadas con la etapa inicial del proceso de reproducción del sistema. Esto establecerá la base fundamental de la arquitectura. Además, se entregará una primera versión del producto, lo que permitirá al cliente probar algunas

de las funcionalidades clave como listar cámaras y seleccionar búsqueda de movimiento y búsqueda de rostro.

Iteración 2: Durante esta iteración, se continuará en la implementación de Historias de Usuario de alta prioridad relacionadas con la etapa a la realización de la búsqueda del módulo. Se solucionarán los errores y no conformidades detectadas por el cliente relacionadas con las implementaciones de la iteración anterior. Al final de esta iteración el cliente podrá probar las funcionalidades restantes de las búsquedas (matrícula y objeto) y podrá llenar el formulario del filtrado de la búsqueda.

Iteración 3: Durante esta iteración, se continuará en la implementación de Historias de Usuario de alta prioridad relacionadas con la etapa visualización de los resultados de la búsqueda del módulo. Se solucionarán los errores y no conformidades detectadas por el cliente relacionadas con las implementaciones de la iteración anterior. Al final de esta iteración el cliente podrá mostrar video de la búsqueda realizada.

Iteración 4: Durante esta iteración, se continuará en la implementación de Historias de Usuario de prioridad alta relacionadas con la etapa reproducción del resultado de la búsqueda del módulo. Se solucionarán los errores y no conformidades detectadas por el cliente relacionadas con las implementaciones de la iteración anterior. Al final de esta iteración el cliente podrá reproducir los videos de las imágenes obtenidas posterior de la realización de la búsqueda.

Tabla 5. Plan de duración de las iteraciones

Iteración	Historia de usuario	Duración de la iteración
1	Listar cámaras	3
	Búsqueda de movimiento (ancho_maximo, ancho_minimo, altura_maxima, altura_minima)	
	Búsqueda rostro (similitud, imagen)	
2	Búsqueda matrícula (similitud, matricula)	3
	Búsqueda objeto (similitud, listado_objeto)	
	Filtrar la búsqueda (fecha_inicio, fecha_fin, hora_inicio, hora_fin)	
3	Mostrar video	3
4	Reproducir video	3

Cuando las iteraciones son realizadas completamente, deberían de estar las HU correspondientes a cada una de las iteraciones que se realizaron, cumplidas satisfactoriamente. En la tabla anterior se muestran las HU que corresponden a cada iteración, junto al total de días estimados que debe demorarse el equipo de desarrollo en realizarlas, y en la siguiente tabla, se muestra el plan de entrega de cada una de estas iteraciones.

Tabla 6. Plan de entrega de las iteraciones

Iteraciones	**1**	**2**	**3**	**4**
HU	De la 1 a la 3	De la 4 a la 6	La 7	La 8
Fecha de entrega	03-07-2023	29-08-2023	27-09-2023	26-10-2023

2.4 Diseño del sistema del módulo de búsqueda forense:

Sobre los diagramas, se es muy claro que se pueden usar siempre que no tome mucho tiempo en realizarlos. En XP se prefiere tener una descripción del sistema o parte de él, en lugar de una serie de complejos diagramas que probablemente tomen más tiempo y sean menos instructivos (Echeverry Tobón & Delgado Carmona, 2007, p. 26)

2.4.1 Tarjetas de Clase-Responsabilidad-Colaboración (CRC)

Las siguientes tablas determinan las tarjetas de clases responsabilidad colaboración del sistema a implementar.

Tabla 7. Tarjeta CRC Operaciones de Búsqueda y Selección

Clase: Camara	
Responsabilidades	**Colaboraciones**
Guardar la información de las cámaras del sistema.	

Tabla 8. Tarjeta CRC Filtros y Criterios de Búsqueda

Clase: Servers	
Responsabilidades	**Colaboraciones**
Guardar la información de los servidores del sistema.	

2.4.2 Patrones de arquitectura de software

Los patrones arquitectónicos se utilizan para expresar una estructura de organización base o esquema para un software. Proporcionando un conjunto de subsistemas predefinidos, especificando sus responsabilidades, reglas, directrices que determinan la organización, comunicación, interacción y relaciones entre ellos (Gómez Mejías, 2013).

Modelo Vista Vista-Modelo (Microsoft Ignite, 2023)

El patrón Modelo-Vista-Modelo (MVVM), es un patrón de arquitectura de software que separa la aplicación en 3 componentes: Modelo, Vista, Vista-Modelo, cada una con un propósito diferente.

Modelo: es la parte lógica del sistema, en donde se realiza el tratamiento de la información el cual interactúa directamente con la base de datos, es la parte que el usuario no ve.

Vista: es el componente que interactúa con el usuario es decir interpreta y muestra los datos que el usuario requiere.

Vista- Modelo: determina la separación de estos componentes y al mismo tiempo el acoplamiento de sus resultados, siendo así el componente necesario para que el sistema funcione.

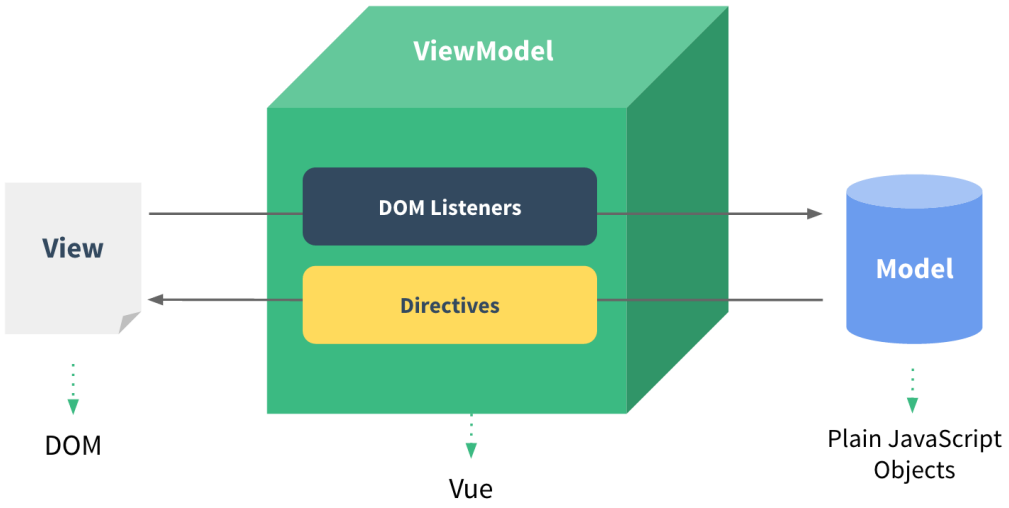


Figura 4: Representación del patrón arquitectónico (Tomado de https://styde.net/introduccion-a-vue-js/)

2.4.3 Patrones de diseño

Los patrones de diseño son la base para la búsqueda de soluciones a problemas comunes en el desarrollo de software y otros ámbitos referentes al diseño de interacción

o interfaces. Para que una solución sea considerada un patrón debe poseer ciertas características; una de ellas es su efectividad resolviendo problemas similares en diferentes ocasiones y la otra es que debe ser reusable, lo que significa que es aplicable a diferentes problemas de diseño en distintas circunstancias (Merlino, Vranic, Rodríguez, Pytel, & García Martínez, 2011).

Patrones de diseño GRASP

Los patrones GRASP (por sus siglas en inglés, General Responsibility Assignment Software Patterns) describen los principios fundamentales de la asignación de responsabilidades a objetos. El nombre se eligió para indicar la importancia de captar estos principios, si se quiere diseñar eficazmente el software orientado a objetos (Demestre & González, 2019). Los patrones que se utilizaron en la investigación fueron los siguientes:

• Patrón Experto: Es el principal responsable de la asignación de responsabilidades. Se refiere a la clase que cuenta con la información o atributos necesarios para cumplir con la responsabilidad o responsabilidades para la que fue implementada. De esta forma se conserva el encapsulamiento y se logra una mayor cohesión. En la investigación, este patrón se evidencia en la clase CameraComponent.

```
<CameraComponent
  :key="templ + '-' + r + '-' + c + ':' + ir + '-' + ic"
  :idContainer="
    templ + '-' + r + '-' + c + ':' + ir + '-' + ic
  "
  :compHeigth="template[templ].height[c - 1]"
  :ref="templ + '-' + r + '-' + c + ':' + ir + '-' + ic"
  :portJanus="null"
  :allowMenu="true"
  :allowRecord="true"
  :toolBar="true"
  :createChanelsFor="null"
  :allowSelection="true"
/>
```

Figura 5: Patrón GRASP Experto

• Patrón Controlador: Puede ser definido como el elemento intermedio entre una interfaz determinada y el algoritmo que la implementa. Es quien ayuda a decidir quién debería encargarse de un evento del sistema (Evento de alto nivel generado por un actor externo al sistema). Un controlador es un objeto de interfaz no destinada al usuario que se encarga de manejar un evento del sistema. Al llegar el evento, una petición del usuario al sistema el controlador decidirá quién se va a encargar de manejar dicho evento o petición. Define además el método de su operación. En la investigación está presente en el componente forensic.js.

```
actions: {
  Buscar({ state, commit, rootGetters }, data) {
    console.warn("parametros busueda",data.parametros);
    let remoteObject = rootGetters.getRemoteObject(
      state.selectedServer.idServer
    );
    let obj = remoteObject.filter( (i)=>{  return i.type==data.type})[0].obj;
    obj.search(data.parametros);
  },
},
```

Figura 6: Patrón GRASP Controlador

• Patrón Alta Cohesión: Una clase con alta cohesión tiene un número relativamente pequeño de métodos, con funcionalidades altamente relacionadas, y no realiza mucho trabajo. Colabora con otros objetos para compartir el esfuerzo si la tarea es extensa.

• Patrón Bajo Acoplamiento: Una clase con bajo acoplamiento asigna las responsabilidades a cada clase para mantener un bajo acoplamiento entre las mismas. El bajo acoplamiento apoya al diseño de clases más independientes, que reduce el impacto de los cambios, así como clases más reutilizables. El acoplamiento no es importante sino se busca la reutilización.

Estos dos últimos patrones se evidencian en la investigación en la implementación del propio marco de trabajo, el cual permite el uso de forma individual de los componentes, poniendo de manifiesto el bajo acoplamiento, así como la dependencia existente entre ellos o alta cohesión.

Patrones de diseño GOF

Patrones publicados por Gamma, Helm, Johnson y Vlossodes en 1995: patrones de la banda de los cuatro (del inglés, *Gang of Four*). Esta serie de patrones permiten ampliar el lenguaje, aprender nuevos estilos de diseño y además introducir más notación UML. Existen 23 patrones GoF de los cuales 15 se utilizan con frecuencia. Los patrones de diseño del grupo GoF se clasifican en tres grandes categorías basadas en su propósito: creacionales, estructurales y de comportamiento (Demestre & González, 2019). A continuación, se describe el patrón GoF utilizado en la solución propuesta:

- Observador (Observer): Define una dependencia de uno-a-muchos entre objetos, de forma que cuando un objeto cambie de estado se notifique y actualicen automáticamente todos los objetos que dependen de él. Vue.js utiliza este patrón para implementar su sistema de reactividad. Los componentes de este observan los cambios en los datos del modelo y actualizan automáticamente la vista cuando se producen cambios.

```
export default {
  namespaced: true,
  state: {
    selectedServer: null,
    selectedCamera: null,
    searchResult: null,
  },
  getters: {},
  mutations: {
    setStates(state, data){
    state[data.state]=data.info
    }
  },
```

Figura 7: Patrón GOF Observador

- Estrategia (Strategy): Permite disponer de varios métodos para resolver un problema y elegir cuál utilizar en tiempo de ejecución. Vue.js utiliza el patrón Estrategia en su sistema de directivas. Las directivas de Vue.js, como v-bind y v-on, permiten definir diferentes estrategias de manipulación del DOM y eventos, lo que brinda flexibilidad al desarrollador para adaptarse a diferentes escenarios.

```
<template v-slot:activator="{ on, attrs }">
  <v-text-field
    v-model="time_fin"
    label="Hora Hasta:"
    prepend-icon="mdi-clock-time-four-outline"
    readonly
    dense
    v-bind="attrs"
    v-on="on"
    :disabled="!valid"
  ></v-text-field>
</template>
```

Figura 8: Patrón GOF Estrategia

Conclusiones del capítulo

La participación del cliente junto a los desarrolladores como lo plantea la metodología XP, jugó un papel clave para establecer las prioridades en la correcta planeación y análisis de una solución para lograr el sistema informático para la búsqueda de objetos y eventos en las cámaras de vigilancia. Al fijar los días para las entregas de las HU, se tomaron acuerdos de entregas de estas HU y se fueron corrigiendo errores en el desarrollo de las mismas, que conllevarán más adelante a una mejor implementación y satisfacción del cliente. Se realizó la descripción de la propuesta de solución a partir de las deficiencias y necesidades identificadas, asegurando una implementación correctamente descrita y detallada. La selección del patrón arquitectónico MVVM brindó la posibilidad de optimizar la organización y la comunicación entre los diferentes componentes de la de solución propuesta.

CAPÍTULO 3: IMPLEMENTACIÓN DE LA SOLUCIÓN, VALIDACIÓN Y PRUEBAS SOBRE EL MÓDULO DE BÚSQUEDA FORENSE DEL SISTEMA INTEGRAL DE SEGURIDAD WINTELI.

El presente capítulo muestra el proceso de implementación del Módulo de Búsqueda Forense del Sistema Integral de Seguridad Winteli, así como las diferentes tareas de ingeniería (a partir de ahora TI) que propone la metodología XP a desarrollarse en las cuatro iteraciones que conforman el proyecto. También se podrá encontrar la validación de la solución propuesta mediante la aplicación de pruebas a los niveles de unidad y aceptación, utilizando los métodos de caja blanca con la técnica del camino básico y el método de caja negra con la técnica de partición equivalente.

3.1 Implementación de la solución

En esta fase, las HU presentadas en el capítulo anterior se descomponen en TI, y son asignadas a uno de los desarrolladores de la aplicación, que sería la persona responsable de su implementación. Estas tareas al ser asignadas a un programador, se describen en un lenguaje técnico y no en un lenguaje entendible para el cliente. Además, se podrán observar las diferentes descripciones de los estándares de codificación utilizados por los programadores de la aplicación.

3.1.1 Diagrama de despliegue

Los diagramas de despliegue muestran las relaciones físicas de los distintos nodos que componen un sistema y el reparto de los componentes sobre dichos nodos. La vista de despliegue representa la disposición de las instancias de componentes de ejecución en instancias de nodos conectados por enlaces de comunicación (Grau & Segura, 2008).

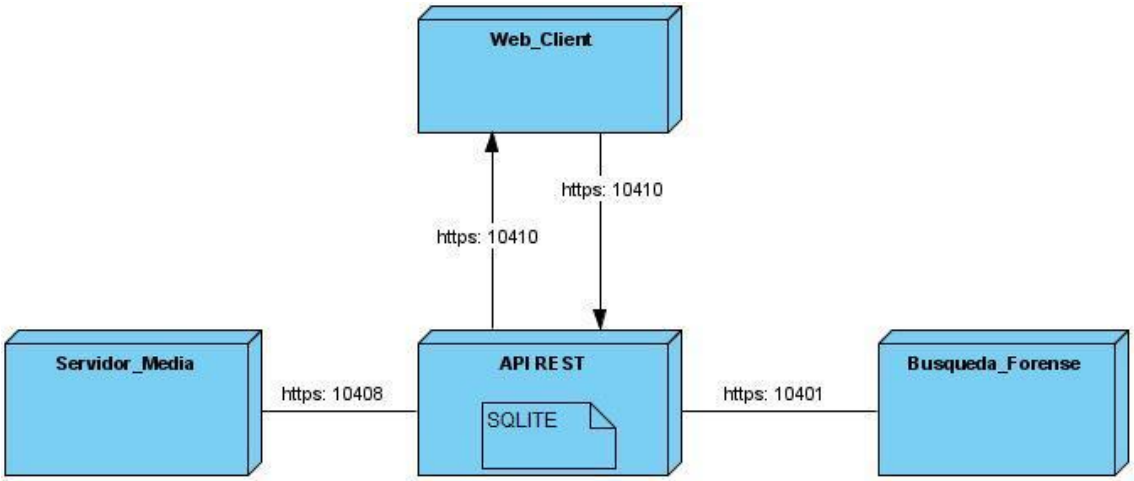


Figura 9: Diagrama de despliegue

El diagrama representa la comunicación que existe entre los nodos que interactúan en el flujo de la información. Donde el cliente web y el API REST, hacen uso del protocolo https por el puerto 10410, para el proceso de solicitud de la búsqueda y del protocolo https por el puerto 10408, para el flujo de video.

La comunicación entre el API REST y el servidor búsqueda forense se realiza a través del protocolo https por el puerto 10401. Mientras que la comunicación entre el servidor media y el servidor web para almacenar fotos y videos grabados, se realiza a través el protocolo https.

3.1.2 Estándares de codificación

Los estándares de codificación se utilizan para darle una mejor organización al código del programa, así a la hora de darle mantenimiento, este legible y bien estructurado. Un código que siga estos estándares de codificación, no será problema de mantener en el tiempo por los desarrolladores de software, ya sean los que desarrollaron ese código en específico, o cualquier otro desarrollador que tenga que darle mantenimiento. A continuación, se definen los estándares de codificación para la implementación del sistema informático para la correlación web.

1. Conversión de nomenclaturas.
    - Las variables, paquetes y módulos llevan el nombre en minúsculas y subrayado.
    - Las clases, funciones, excepciones y formularios llevan el nombre de CapWords (Caso camel, primera letra en mayúscula y demás minúsculas)
    - Los parámetros de la función llevan el nombre lower_with_under (minúsculas y subrayado).

Ejemplo:

```
479     RealizarBusqueda() {
480       let params = {
481         //parámetros genéricos
482         StartDate: `${this.date} ${this.time_ini}`, //"2023-03-15 00:05",
483         EndDate: `${this.date1} ${this.time_fin}`, //"2023-03-27 23:55",
484         getAll: false,
485         CameraId: `VideoSource${this.selectedCamera.idcamara}-${this.selectedCamera.nombre}`
486       };
```

Figura 10: Conversión de nomenclaturas

2. Diseño del código
    - Cada línea de código tiene como máximo una longitud de 80 caracteres, si es demasiado largo se debe usar barras, corchetes o llaves para doblar.
    - Las funciones o clases deben estar separadas por 2 líneas en blanco y si se usan diferentes funciones dentro de esa clase, separarlas con 1 línea en

blanco.

- En la instrucción if / for, incluso si solo hay una oración en la instrucción de ejecución, comience una nueva hilera.

Ejemplo:

```
imageToSend() {
  var dataSeparated = this.facePreview.split(",");
  var byteString = atob(dataSeparated[1]);
  var img = new Array(byteString.length);
  for (var i = 0; i < byteString.length; i++) {
    img[i] = byteString.charCodeAt(i);
  }
  return img;
},


RealizarBusqueda() {
  let params = {
    //parámetros genéricos
    StartDate: `${this.date} ${this.time_ini}`, //"2023-03-15 00:05",
    EndDate: `${this.date1} ${this.time_fin}`, //"2023-03-27 23:55",
    getAll: false,
    CameraId: `VideoSource${this.selectedCamera.idcamara}-${this.selectedCamera.nombre}`
  };

  let tipoBusq = null;
  if (this.tab == 0) {
    tipoBusq = "MotionForensicSearch";
    params.minWidth = this.minancho;
    params.minHeight = this.minaltura;
    params.maxWidth = this.maxancho;
    params.maxHeight = this.maxaltura;
  }
```

Figura 11: Diseño del código

3.1.3 Tareas de ingeniería (TI)

Las TI se establecerán por el orden de las iteraciones en que fueron desarrolladas las HU. A continuación, se observarán las TI correspondientes a la primera iteración del proceso de desarrollo, que estas incluyen las HU del 1, 2 y 3. **Las demás TI se podrán ver en el Anexo 2.**

Tabla 9. TI Listar cámaras

Tarea de Ingeniería	
Número de la tarea: 1	**Número de HU:** 1
Nombre de la tarea: Listar cámaras	
Tipo de tarea: Desarrollo	**Puntos estimados:** 1
Programador responsable: Zenia Zerquera González	
Descripción: Para listar las cámaras en el sistema, es fundamental que estas estén debidamente configuradas con las analíticas necesarias que permitan llevar a cabo búsquedas efectivas. Cuando las cámaras están configuradas de manera adecuada, el sistema desplegará una lista de cámaras organizadas en una estructura de árbol,	

proporcionando una visión estructurada de las mismas; es decir, agrupadas por áreas. Por otro lado, si las cámaras no están configuradas, el panel de cámaras aparecerá vacío, sin información disponible. Es importante destacar que, para acceder a esta funcionalidad, el usuario debe haberse autenticado previamente en el sistema y contar con los permisos necesarios para realizar esta acción.

Tabla 10. TI Búsqueda de movimiento (ancho_maximo, ancho_minimo, altura_maxima, altura_minima)

Tarea de Ingeniería	
Número de la tarea: 2	**Número de HU:** 2
Nombre de la tarea: Búsqueda de movimiento (ancho_maximo, ancho_minimo, altura_maxima, altura_minima)	
Tipo de tarea: Desarrollo	**Puntos estimados:** 1
Programador responsable: Claudia Hernández Céspedes	
Descripción: La búsqueda por detección de movimiento implica que primeramente debe estar seleccionada una cámara; sino la funcionalidad estará inhabilitada. Una vez habilitada la búsqueda el usuario podrá llenar los campos con los valores requeridos por dicha búsqueda, luego podrá realizar la búsqueda haciendo clic en el botón buscar o cancelar la búsqueda dando clic en el botón cancelar; si se realiza la búsqueda el sistema muestra las imágenes que estén guardadas en el servidor de media, en caso de cancelar la búsqueda los campos que llenó anteriormente pasan a estar vacíos nuevamente. Es importante destacar que, para acceder a esta funcionalidad, el usuario debe haberse autenticado previamente en el sistema y contar con los permisos necesarios para realizar esta acción.	

Tabla 11. TI Búsqueda rostro (similitud, imagen)

Tarea de Ingeniería	
Número de la tarea: 3	**Número de HU:** 3
Nombre de la tarea: Búsqueda rostro (similitud, imagen)	
Tipo de tarea: Desarrollo	**Puntos estimados:** 1
Programador responsable: Claudia Hernández Céspedes	
Descripción: La búsqueda por reconocimiento facial (rostro) implica que primeramente debe estar seleccionada una cámara; sino la funcionalidad estará inhabilitada. Una vez habilitada la búsqueda el usuario podrá llenar los campos con los valores requeridos por dicha búsqueda, luego podrá realizar la búsqueda haciendo	

clic en el botón buscar o cancelar la búsqueda dando clic en el botón cancelar; si se realiza la búsqueda el sistema muestra las imágenes que estén guardadas en el servidor de media, en caso de cancelar la búsqueda los campos que llenó anteriormente pasan a estar vacíos nuevamente. Es importante destacar que, para acceder a esta funcionalidad, el usuario debe haberse autenticado previamente en el sistema y contar con los permisos necesarios para realizar esta acción.

3.2 Pruebas de software

La metodología XP divide las pruebas del sistema en dos grupos pruebas a nivel unidad, encargadas de verificar el código y diseñada por los programadores, y pruebas de nivel aceptación o pruebas funcionales destinadas a evaluar si al final de una iteración se consiguió la funcionalidad requerida diseñadas por el cliente final (Beck, 1999).

Además, de las pruebas referidas en la metodología, en el proceso de probar el sistema se usaron pruebas a nivel de sistema que son según Sommerville la integración de componentes para crear una versión del sistema y, luego, poner a prueba el sistema integrado. Las pruebas de sistema demuestran que los componentes son compatibles, que interactúan correctamente y que transfieren los datos correctos en el momento adecuado a través de sus interfaces (Sommerville, 2011).

Mediante la aplicación de pruebas al módulo de búsqueda forense, se llegó a una solución óptima y dichas pruebas se mostrarán más adelante con los resultados obtenidos por cada iteración. Pues XP propone realizar la mayor cantidad de pruebas posibles al sistema, con el objetivo de detectar la mayor cantidad de problemas o fallos en las funcionalidades del sistema y poder corregirlas a tiempo.

3.2.1 Pruebas a nivel unitarias

Las pruebas a nivel de unidad o unitarias son el proceso de probar componentes del programa tales como métodos o clases de objetos. Las funciones o los métodos individuales son el tipo más simple de componente. Las pruebas deben llamarse para dichas rutinas con diferentes parámetros de entrada (Sommerville, 2011).

Utilizando el método de caja blanca el cual consiste específicamente en cómo diseñar los casos de prueba atendiendo al comportamiento interno y la estructura del programa, examinándose la lógica interna sin considerar los aspectos de rendimiento (Roger, 2002).Usando las técnicas de camino básico la escogida para realizar las pruebas pues permite obtener una medida de la complejidad lógica del diseño procedimental y usar esa medida como guía para la definición de un conjunto básico de caminos de ejecución,

garantizando con estos que durante la prueba se ejecute por lo menos una vez cada sentencia del programa.

Antes de aplicar esta técnica es necesario calcular la complejidad ciclomática del algoritmo o fragmento de código que se analice. Se escogió por los desarrolladores el método pickFile (), el cual permite seleccionar una imagen para utilizarla como referencia de búsqueda en las grabaciones.

```
pickFile() {
1   const file = document.querySelector("input[type=file]").files[0];
2   if (file) {
      const reader = new FileReader();
      let rawImg;
      reader.onloadend = () => {
3       rawImg = reader.result;
        this.facePreview = rawImg;
      };
      reader.readAsDataURL(file);
    }
4 },
```

Figura 12: Código del método pickFile

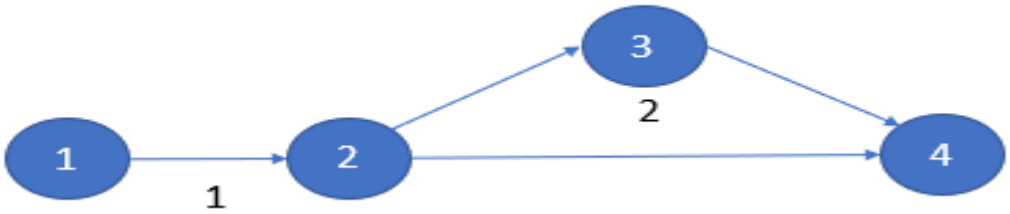


Figura 13: Camino básico del método pickFile

En la tabla que se presenta a continuación se evidencian las fórmulas para calcular la complejidad ciclomática.

Tabla 12. Fórmulas para calcular la complejidad ciclomática

Casos	Descripción
V(G) = (E - N) + 2 V(G) = 4 - 4 + 2 V(G) = 2	donde “E” es el número de aristas y “N” es el número de nodos
V(G) = P + 1 V(G) = 1 + 1 V(G) = 2	donde “P” es el número de nodos predicados (son los nodos de los cuales parten dos o más aristas)

V(G) = R V(G) = 2	donde “R” es el número de regiones en el grafo

El cálculo realizado ha dado el mismo resultado en los tres casos; se puede decir que la complejidad ciclomática del código del método pickFile () es 2, lo que significa que hay dos caminos independientes a través del código. Uno cuando el archivo se selecciona y otro cuando no se selecciona.

Tabla 13. Camino básico

Número	**Camino básico que sigue el flujo**
1	1,2,4
2	1,2,3,4

Una vez determinados los caminos se procede a realizar los casos de pruebas para cada uno de ellos. Para definir los casos de pruebas se tiene en cuenta:

- Trayectoria: Camino básico seleccionado para la prueba.
- Descripción del proceso: Se describe el caso de prueba y se tratan de forma general los aspectos fundamentales de los datos de entrada.
- Resultados Esperados: Se coloca el resultado esperado que debe devolver el sistema después del caso de prueba.

Tabla 14. Caso prueba para la trayectoria1

Caso de prueba para la trayectoria 1	
Trayectoria	1,2,4
Descripción del proceso	Verifica que no hay un archivo seleccionado
Resultados esperados	Muestra el cuadro de visor de imagen como vacío

Tabla 15. Caso de prueba para la trayectoria2

Caso de prueba para la trayectoria 2	
Trayectoria	1,2,3,4
Descripción del proceso	Verifica que hay un archivo seleccionado y lo convierte en url.
Resultados esperados	Se muestra en el visor de imagen el archivo seleccionado

3.2.2 Pruebas a nivel de aceptación

Las pruebas del nivel de aceptación constituyen el principal mecanismo para validar el sistema con el cliente y medir su nivel de satisfacción. Estas pruebas marcan el nivel de calidad del software en relación a lo esperado por el cliente y generan las posibles no conformidades del mismo. (Crispin & House, 2003)

Las pruebas de aceptación son creadas en base a las historias de usuarios, en cada ciclo de la iteración del desarrollo. El cliente es el responsable de especificar uno o diversos escenarios para comprobar que una historia de usuario ha sido correctamente implementada. En caso de que fallen varias pruebas, el cliente es el responsable de indicar el orden de prioridad de resolución.

Una historia de usuario no se puede considerar terminada hasta tanto pase correctamente todas las pruebas de aceptación. Dado que la responsabilidad es grupal, es recomendable publicar los resultados de las pruebas de aceptación, de manera que todo el equipo esté al tanto de esta información. Además, las pruebas de aceptación son consideradas como Pruebas de Caja Negra y son tan importantes como las pruebas unitarias, dado que significan la satisfacción del cliente con el producto desarrollado (Joskowicz, 2008).

Para realizar las pruebas en el nivel de aceptación correctamente se elaboraron una serie de casos de prueba de aceptación para cada HU que deben ser probadas por los usuarios y devolver no conformidades a resolver. Seguidamente se describe los casos de prueba correspondiente a la HU Listar cámaras. **Los demás deben consultarse en el Anexo 3.**

Tabla 16. Caso prueba aceptación HU1_P1

Casos de pruebas de aceptación	
Código: HU1_P1	**Historia de usuario:** Listar cámaras
Nombre: Listar cámaras	
Descripción: Yo quiero que me muestre una lista de todas las cámaras que tengan configurada las analíticas de la búsqueda forense, pero que sea en un panel en la parte izquierda de la pantalla.	
Condición de ejecución: ● El usuario debe estar autenticado en el sistema. ● Las cámaras deben tener configuradas las analíticas correspondientes y deben tener conexión con los servidores de media, streaming y búsqueda forense.	
Entrada/Pasos de ejecución:	

● El usuario accede al módulo de búsqueda forense ● El usuario selecciona “Cámara” del menú desplegable del panel mostrado en la parte izquierda de la pantalla.
Resultado de la prueba: El sistema muestra el listado de las cámaras.
Evaluación de la prueba: Satisfactoria

3.2.3 Prueba de integración

Las pruebas de integración son una técnica sistemática para construir la arquitectura del software mientras se llevan a cabo pruebas para descubrir errores asociados con la interfaz. El objetivo es tomar los componentes probados de manera individual y construir una estructura de programa que se haya dictado por diseño. (Bejerano Y, 2019)

Los objetivos de las pruebas de integración son reducir el riesgo de afectaciones de seguridad en la aplicación, Encontrar defectos en las comunicaciones, verificar que el comportamiento del software entre las distintas interfaces es el esperado y prevenir la aparición de defectos en fases posteriores de pruebas asegurar la confianza en la calidad de las interfaces. (Moreno O, 2019)

Tabla 17. Prueba de integración del módulo

Prueba de integración			
Prueba Sistema - Módulo			
Objetivo	**Descripción de la prueba**	**Resultados esperados**	**Evaluación**
Validar el enlace entre la interfaz del módulo y la pantalla de inicio del sistema	Seleccionar en la barra de navegación el Módulo de Búsqueda Forense	La aplicación muestra el formulario requerido para realizar la búsqueda	Satisfactoria
Prueba Componente - Componente			
Objetivo	**Descripción de la prueba**	**Resultados esperados**	**Evaluación**
Validar la comunicación entre el componente de mostrar cámaras y	Selecciona una cámara del listado y se habilita el formulario	Al seleccionar la cámara el sistema habilita el formulario	Satisfactoria

el componente del
formulario

3.3 Resultados de las pruebas

Las pruebas a las funcionalidades se realizaron en cuatro iteraciones detectándose no conformidades significativas y no significativas. Las no conformidades no significativas, se centraron en errores ortográficos como: omisiones de tildes, paréntesis, cambio de mayúscula por minúscula. Las significativas, en errores de validación funcionales.

En la siguiente gráfica se muestran las no conformidades encontradas a lo largo de todo el proceso de aplicación de caso de uso de pruebas de aceptación.

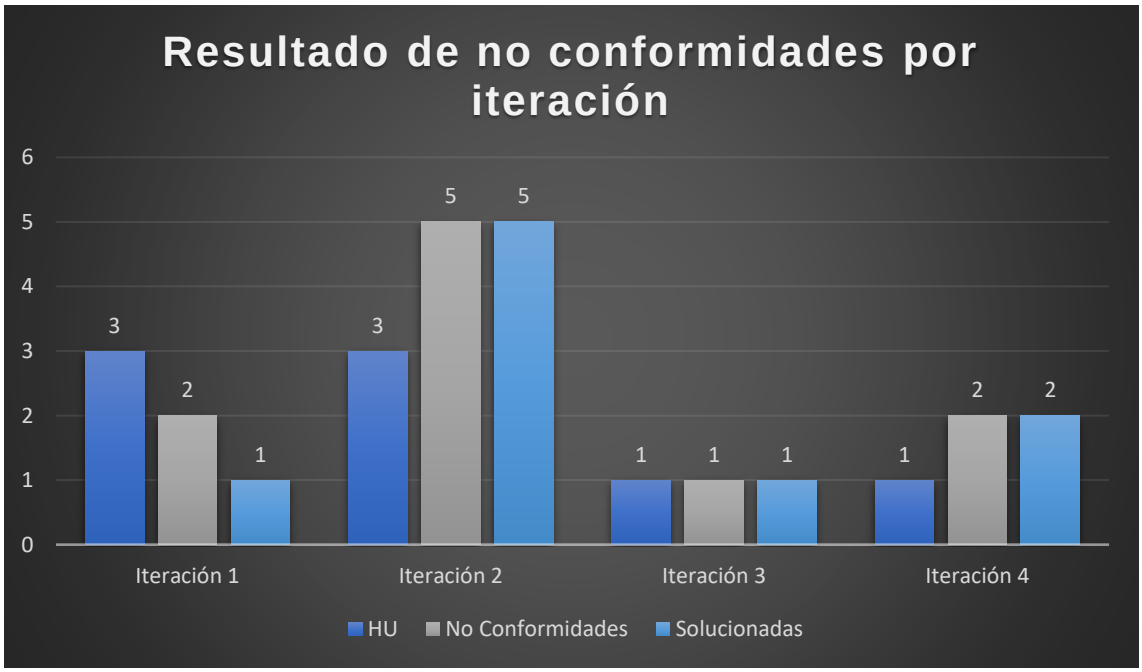


Figura 14: No conformidades detectadas en los casos de prueba de aceptación

En la primera iteración se realizaron las pruebas a 3 historias de usuario, donde se detectó 2 no conformidades que se solucionaron satisfactoriamente. En la segunda iteración se realizaron las pruebas a 3 historias de usuario, donde se detectaron 5 no conformidades, se realizaron correcciones a 3 quedando para una segunda vuelta de pruebas 2 no conformidades; en esta se corrigieron todas las no conformidades pendientes de dicha iteración. En la tercera iteración se realizaron las pruebas a 1 historia de usuario, donde se detectaron 1 no conformidad que fue resuelta satisfactoriamente. En la cuarta iteración se realizaron las pruebas a 1 historia de usuario, donde se detectaron 2 no conformidades que se solucionaron satisfactoriamente.

Las no conformidades encontradas después de concluida cada iteración de pruebas se analizaron por parte del equipo de desarrollo para corregir los errores detectados en la iteración anterior, lo que contribuyó a mejorar la calidad y funcionalidad del software. Al final de la cuarta iteración quedaron resueltas todas las no conformidades detectadas.

Conclusiones parciales

La ejecución de este capítulo permitió someter el sistema a un riguroso proceso de evaluación a través de pruebas unitarias y de aceptación, conforme a las directrices establecidas por la metodología XP. Durante estas pruebas, se identificaron ciertos aspectos que no cumplían con las expectativas, lo que llevó a la realización de iteraciones posteriores para abordar y resolver estas deficiencias. Este enfoque iterativo y colaborativo resultó en una evaluación exitosa de las funcionalidades del sistema.

CONCLUSIONES FINALES

Una vez completada la fase de desarrollo del Módulo de Búsqueda Forense del Sistema Integral de Seguridad “Winteli”, se puede llegar a la conclusión de que:

- Este estudio ha demostrado eficazmente la aplicación de métodos de investigación científica para construir el marco teórico necesario en el desarrollo del Módulo de Búsqueda Forense. Tras un exhaustivo análisis del estado del arte, se concluyó que ninguna solución existente cumplía con todas las necesidades identificadas. En respuesta, se propusieron herramientas, lenguajes y tecnologías específicos para abordar eficientemente el proceso de búsqueda de eventos y objetivos en las grabaciones de cámaras de seguridad del Sistema Integral de Seguridad Winteli.

- El enfoque metodológico ágil, en particular la metodología XP, fue seleccionado con la intención de proporcionar flexibilidad a lo largo del proyecto, mejorar la calidad y permitir la introducción de novedades durante el desarrollo. La participación activa del cliente y los desarrolladores desempeñó un papel clave en la planificación, priorización y análisis de soluciones, fundamentándose en la metodología XP para lograr un sistema informático eficaz.

- La implementación del Módulo de Búsqueda Forense se llevó a cabo, siguiendo una descripción detallada de la propuesta de solución basada en las deficiencias y necesidades identificadas. La elección del patrón arquitectónico MVVM demostró ser acertada al optimizar la organización y comunicación entre los componentes de la solución.

- La fase de validación se centró en pruebas de software, siguiendo las directrices de la metodología XP. Durante este proceso, se identificaron aspectos a mejorar, lo que condujo a iteraciones posteriores para abordar y resolver deficiencias, confirmando la efectividad del enfoque iterativo y colaborativo.

.

RECOMENDACIONES

La autora una vez terminada la investigación le recomiendan al cliente la posibilidad de incorporar nuevas funcionales al sistema como:

- Descargar un video: Integrar un botón que permita en el componente de reproducción de video la descarga del video que se está reproduciendo.
- Análisis de Sentimientos: Explorar la implementación de análisis de sentimientos en el sistema. Esto puede permitir la detección de comportamientos sospechosos a través del análisis de gestos faciales y expresiones, mejorando la capacidad de detección de eventos.
- Visualización 3D y Mapas Interactivos: Integrar capacidades de visualización 3D y mapas interactivos que muestren la ubicación de las cámaras y eventos en tiempo real. Esto proporcionará una comprensión más completa de la situación y mejorará la respuesta a incidentes.

REFERENCIAS BIBLIOGRÁFICAS

Axios sitio web oficial from https://axios-http.com/docs/intro

Beazley, D. (1999). Python Essential Reference.

Beck, K. (1999). Extreme Programming Explained.

Bejerano Y. Pérez, (2019) «Sistema de Gestión de Licencias del Personal Aeronáutico del Instituto de la Aeronáutica Civil de Cuba versión 2.0», bachelorThesis, Universidad de las Ciencias Informáticas. Facultad 1., 2019. [En línea]. Disponible en: https://repositorio.uci.cu/jspui/handle/123456789/10288

Bustamante, D., & Rodrìguez, J. C. (2014). Metodología actual- Metodología XP. Barinas

Bloch, J. (2001). Effective Java.

Crispin, L., & House, T. (2003). Testing extreme programming: Addison-Wesley Professional.

Dahl, R. (2009). Node.js. Joyent Inc. Recuperado from https://nodejs.org/

Datys (2010). [Online] DATYS le brinda un Sistema de Video Vigilancia profesional.

Demestre, D. H., & González, Y. H. (2019). Diseño de un Sistema de Gestión de Información de Recursos Humanos. 12, 31-49

Echeverry Tobón, L. M., & Delgado Carmona, L. E. (2007). Caso práctico de la metodología ágil XP al desarrollo de software

Eguiluz, J. (Producer). (2018). Libros y manuales sobre diseño y programación web. Uniwebsidad Los mejores recursos para aprender diseño y programación web. Retrieved from https://uniwebsidad.com/libros/javascript

Framework: Qué es, para qué sirve y algunos ejemplos. (2021, agosto 19). Edix España from https://unirfp.unir.net/revista/ingenieria-y-tecnologia/framework/

Gomez Mejias, M. (Producer). (2013). Ingenio DS Patrones Arquitectónicos. Ingenio DS Patrones Arquitectónicos. Retrieved from https://ingeniods.wordpress.com/2013/09/16/patrones-arquitectonicos/

Grau, X. F., & Segura, M. I. S. J. R. e. (2008). Desarrollo orientado a objetos con UML.1

Hikvision Sitio oficial from https://www.hikvision.com/es/core-technologies/deep-learning/facial-recognition/.

Hipp, D. R. (2000). SQLite. Recuperado de https://www.sqlite.org/index.html

Intecto. (2009). Ingeniería del software: Metodología y ciclos de vida In Ingeniería del software: Metodología y ciclos de vida (pp. 59). España

Jova Rodríguez, J.R. (2013). «Módulo de streaming para archivos multimedia del Sistema Integral de Análisis de Información».

Joskowicz, J. J. U. d. V. (2008). Reglas y prácticas en eXtreme Programming. 22.

Klauser, F. (2005). VideoVigilancia y Orden Público: El Caso de la Ciudad de Ginebra. Papeles del CEIC, 2005(1), 1-21

Gladiadores sitio web from https://gladiadores.uci.cu/ponencia-sistema-de-video-vigilancia-xilema-suria-para-la-proteccion-y-seguridad-de-entidades-subsistema-analisis/

La guía sencilla para la diagramación de UML y el modelado de la base de datos. (s. f.). Recuperado 20 de junio de 2022, from https://www.microsoft.com/es-ww/microsoft-365/business-insights-ideas/resources/guide-to-uml-diagramming-and-database-modeling https://www.edix.com/es/instituto/framework/

Letelier, P., & Penadés, M. C. (2006). Metodologías ágiles para el desarrollo de software: eXtreme Programming (XP). 05

Maida, E. G., & Pacienzia, J. (2015). Metodologías de desarrollo de software.

Merlino, H., Vranic, A., Rodríguez, D., Pytel, P., & García Martínez, R. (2011). Ambientes de desarrollo de software basados en patrones de usabilidad. Paper presented at the XIII Workshop de Investigadores en Ciencias de la Computación

Meyers, S. (2014). Effective C++

Microsoft (2023). Modelo-Vista-Modelo de Vista (MVVM) from https://learn.microsoft.com/es-es/dotnet/architecture/maui/mvvm

Miniero, L. (2014). Janus: servidor WebRTC de uso general from https://janus.conf.meetecho.com/docs/

Moreno O. (2019) «Pruebas de integración: validando la unión de componentes», Oscar Moreno, 1 de octubre de 2019. http://oscarmoreno.com/pruebas-de-integracion/

Pressman, R. S. (2010). Ingeniería del software un enfoque práctico (7a.ed.). McGraw-Hill

Requero, Jose L. (1997). Revista Vasca de Derecho Procesar y Arbitraje, tomo IX

Rodríguez, Eduardo (2020). MVVM- Qué es y cómo funciona? from https://inmediatum.com/blog/ingenieria/mvvm-que-es-y-como-funciona/#:~:text=MVVM%2C%20por%20sus%20siglas%20en,parte%20visual%20sea%20totalmente%20independiente.

Roger, S. P. (2002). Ingeniería de Software: Un enfoque práctico: McGraw Hill New York.

¿Qué es un lenguaje de programación? | Desarrollar Inclusión. (s. f.). Recuperado 20 de junio de 2022, from https://desarrollarinclusion.cilsa.org/tecnologia-inclusiva/que-es-un-lenguaje-de-programacion/

Sammons, J. (2018). The Basics of Digital Forensics: The Primer for Getting Started in Digital Forensics. Syngress

Simpson, K. (2014). You Don't Know JS.

Sommerville, I. (2011). Ingeniería de Software (9na. ed). México DF. In: México: Pearson Educación

Thales Cogent. Sistemas de reconocimiento facial from https://bing.com/search?q=sistemas+de+videovigilancia+con+reconocimiento+facial%2c+b%c3%basqueda+de+objetos%2c+b%c3%basqueda+de+patrones+y+b%c3%basqueda+de+caracter%c3%adsticas+f%c3%adsicas.

ULL, María V. (2005). Revista de derecho político. Num. 63

Valladarez, B. S. M. M., Gaitan, B. M. E., & Reyes, B. N. N. P. (2016). Universidad Nacional Autónoma de Nicaragua, Managua Unan-Managua recinto universitario Rubén Darío (rurd) facultad de ciencias e ingeniería departamento de computación

Visual Studio Code from https://code.visualstudio.com/

Visual Paradigm International. Visual Paradigm. Recuperado from https://www.visual-paradigm.com/

Vivotek Sitio oficial from https://www.tecnoseguro.com/analisis/cctv/deep-search-vivotek-una-alternativa-seria-busqueda-forense-videovigilancia.

You, E. (2014). Vue.js. Recuperado from https://vuejs.org/

ANEXO 1: HU PARA EL MÓDULO DE BÚSQUEDA FORENSE DEL SISTEMA INTEGRAL DE SEGURIDAD “WINTELI”

Tabla 18. HU Búsqueda de movimiento (ancho_maximo, ancho_minimo, altura_maxima, altura_minima)

Historia de Usuario

Número: 2	**Nombre Historia**: Búsqueda de movimiento (ancho_maximo, ancho_minimo, altura_maxima, altura_minima)
Prioridad: Alto	**Iteración asignada**: 1
Estimación: 1	
Programador: Zenia Zerquera González	
Riesgo en desarrollo: Alto	
Descripción: Yo quiero realizar la búsqueda de detección de movimiento con los siguientes parámetros: ancho máximo, ancho mínimo, altura máxima, altura mínima.	

Tabla 19. HU Búsqueda rostro (similitud, imagen)

Historias de usuario

Número: 3	**Nombre historia:** Búsqueda rostro (similitud, imagen)
Prioridad: Alto	**Iteración asignada:** 1
Estimación: 1	
Programador: Claudia Hernández Céspedes	
Riesgo en desarrollo: Alto	
Descripción: Yo quiero realizar la búsqueda de detección de rostro con los siguientes parámetros: porciento de similitud y subir una imagen.	

Tabla 20. HU Búsqueda matrícula (similitud, matricula)

Historia de Usuario

Número: 4	**Nombre Historia**: Búsqueda matrícula (similitud, matricula)
Prioridad: Alto	**Iteración asignada**: 2
Estimación: 1	
Programador: Claudia Hernández Céspedes	
Riesgo en desarrollo: Alto	

Descripción: Yo quiero realizar la búsqueda de detección de matrícula con los siguientes parámetros: porciento de similitud y agregar una matrícula.

Tabla 21. HU Búsqueda objeto (similitud, listado_objeto)

Historias de usuario

Número: 5	**Nombre historia:** Búsqueda objeto (similitud, listado_objeto)
Prioridad: Alto	**Iteración asignada:** 2
Estimación: 1	
Programador: Zenia Zerquera González	
Riesgo en desarrollo: Alto	
Descripción: Yo quiero realizar la búsqueda de detección de objeto con los siguientes parámetros: porciento de similitud y seleccionar uno o varios objetos.	

Tabla 22. HU Filtrar la búsqueda (fecha_inicio, fecha_fin, hora_inicio, hora_fin)

Historias de usuario

Número: 6	**Nombre historia:** Filtrar la búsqueda (fecha_inicio, fecha_fin, hora_inicio, hora_fin)
Prioridad: Medio	**Iteración asignada:** 2
Estimación: 1	
Programador: Claudia Hernández Céspedes	
Riesgo en desarrollo: Medio	
Descripción: Yo quiero que para realizar una búsqueda se filtren la fecha de inicio, la fecha de fin, la hora de inicio y la hora de fin.	

Tabla 23. HU Mostrar video

Historias de usuario

Número: 7	**Nombre historia:** Mostrar video
Prioridad: Alto	**Iteración asignada:** 3
Estimación: 3	
Programador: Zenia Zerquera González	
Riesgo en desarrollo: Alto	
Descripción: Yo quiero que se muestre el video de las imágenes que se obtuvieron en la búsqueda.	

Tabla 24. HU Reproducir video

Historias de usuario

Número: 8	**Nombre historia:** Reproducir video
Prioridad: Alto	**Iteración asignada:** 4
Estimación: 3	
Programador: Zenia Zerquera González	
Riesgo en desarrollo: Alto	
Descripción: Yo quiero que al video mostrado se pueda pausar y continuar la reproducción.	

ANEXO 2: TAREAS DE INGIENERÍA (TI) PARA EL MÓDULO DE BÚSQUEDA FORENSE DEL SISTEMA INTEGRAL DE SEGURIDAD "WINTELI"

Tabla 25. TI Búsqueda matrícula (similitud, matricula)

Tarea de Ingeniería	
Número de la tarea: 4	**Número de HU:** 4
Nombre de la tarea: Búsqueda matrícula (similitud, matricula)	
Tipo de tarea: Desarrollo	**Puntos estimados:** 1
Programador responsable: Zenia Zerquera González	
Descripción: La búsqueda por detección de matrícula implica que primeramente debe estar seleccionada una cámara; sino la funcionalidad estará inhabilitada. Una vez habilitada la búsqueda el usuario podrá llenar los campos con los valores requeridos por dicha búsqueda, luego podrá realizar la búsqueda haciendo clic en el botón buscar o cancelar la búsqueda dando clic en el botón cancelar; si se realiza la búsqueda el sistema muestra las imágenes que estén guardadas en el servidor de media, en caso de cancelar la búsqueda los campos que llenó anteriormente pasan a estar vacíos nuevamente. Es importante destacar que, para acceder a esta funcionalidad, el usuario debe haberse autenticado previamente en el sistema y contar con los permisos necesarios para realizar esta acción.	

Tabla 26. TI Búsqueda objeto (similitud, listado_objeto)

Tarea de Ingeniería	
Número de la tarea: 5	**Número de HU:** 5
Nombre de la tarea: Búsqueda objeto (similitud, listado_objeto)	
Tipo de tarea: Desarrollo	**Puntos estimados:** 1
Programador responsable: Claudia Hernández Céspedes	
Descripción: La búsqueda por detección de objeto implica que primeramente debe estar seleccionada una cámara; sino la funcionalidad estará inhabilitada. Una vez habilitada la búsqueda el usuario podrá llenar los campos con los valores requeridos por dicha búsqueda, luego podrá realizar la búsqueda haciendo clic en el botón buscar o cancelar la búsqueda dando clic en el botón cancelar; si se realiza la búsqueda el sistema muestra las imágenes que estén guardadas en el servidor de media, en caso de cancelar la búsqueda los campos que llenó anteriormente pasan a estar vacíos nuevamente. Es importante destacar que, para acceder a esta	

funcionalidad, el usuario debe haberse autenticado previamente en el sistema y contar con los permisos necesarios para realizar esta acción.

Tabla 27. TI Filtrar la búsqueda (fecha_inicio, fecha_fin, hora_inicio, hora_fin)

Tarea de Ingeniería	
Número de la tarea: 6	**Número de HU:** 6
Nombre de la tarea: Filtrar la búsqueda (fecha_inicio, fecha_fin, hora_inicio, hora_fin)	
Tipo de tarea: Desarrollo	**Puntos estimados:** 1
Programador responsable: Claudia Hernández Céspedes	
Descripción: El proceso de filtrar la búsqueda implica que primeramente debe estar seleccionada una cámara; sino la funcionalidad estará inhabilitada. Una vez habilitada esta función el usuario podrá llenar los campos con los valores requeridos por dicho filtrado; es decir, llenar los campos de fecha inicio, fecha fin, hora inicio y hora fin. Luego podrá seleccionar el tipo de búsqueda deseada y procederá a llenar los campos requeridos por dicha búsqueda, luego podrá realizar la búsqueda haciendo clic en el botón buscar o cancelar la búsqueda dando clic en el botón cancelar; si se realiza la búsqueda el sistema muestra las imágenes que estén guardadas en el servidor de media, en caso de cancelar la búsqueda los campos que llenó anteriormente pasan a estar vacíos nuevamente. Es importante destacar que, para acceder a esta funcionalidad, el usuario debe haberse autenticado previamente en el sistema y contar con los permisos necesarios para realizar esta acción.	

Tabla 28. TI Mostrar video

Tarea de Ingeniería	
Número de la tarea: 7	**Número de HU:** 7
Nombre de la tarea: Mostrar video	
Tipo de tarea: Desarrollo	**Puntos estimados:** 3
Programador responsable: Zenia Zerquera González	
Descripción: El proceso de mostrar video implica que primeramente debe estar seleccionada una cámara; sino la funcionalidad estará inhabilitada. Una vez habilitada esta función el usuario podrá llenar los campos con los valores requeridos por la funcionalidad filtrar la búsqueda; es decir, llenar los campos de fecha inicio, fecha fin, hora inicio y hora fin. Luego podrá seleccionar el tipo de búsqueda deseada y procederá a llenar los campos requeridos por dicha búsqueda, luego podrá realizar la	

búsqueda haciendo clic en el botón buscar o cancelar la búsqueda dando clic en el botón cancelar; si se realiza la búsqueda el sistema muestra las imágenes que estén guardadas en el servidor de media, en caso de cancelar la búsqueda los campos que llenó anteriormente pasan a estar vacíos nuevamente. Es importante destacar que, para acceder a esta funcionalidad, el usuario debe haberse autenticado previamente en el sistema y contar con los permisos necesarios para realizar esta acción.

Tabla 29. TI Reproducir video

Tarea de Ingeniería	
Número de la tarea: 8	**Número de HU:** 8
Nombre de la tarea: Reproducir video	
Tipo de tarea: Desarrollo	**Puntos estimados:** 3
Programador responsable: Claudia Hernández Céspedes	
Descripción: El proceso de reproducir video implica que primeramente debe estar seleccionada una cámara; sino la funcionalidad estará inhabilitada. Una vez habilitada esta función el usuario podrá llenar los campos con los valores requeridos por la funcionalidad filtrar la búsqueda; es decir, llenar los campos de fecha inicio, fecha fin, hora inicio y hora fin. Luego podrá seleccionar el tipo de búsqueda deseada y procederá a llenar los campos requeridos por dicha búsqueda, luego podrá realizar la búsqueda haciendo clic en el botón buscar o cancelar la búsqueda dando clic en el botón cancelar; si se realiza la búsqueda el sistema muestra las imágenes que estén guardadas en el servidor de media, en caso de cancelar la búsqueda los campos que llenó anteriormente pasan a estar vacíos nuevamente. Una vez mostrada las imágenes el usuario podrá seleccionar el video que desea reproducir. El sistema mostrará la reproducción de este y activará la función de pausar y continuar video. Es importante destacar que, para acceder a esta funcionalidad, el usuario debe haberse autenticado previamente en el sistema y contar con los permisos necesarios para realizar esta acción.	

ANEXO 3 CASOS DE PRUEBA DE ACEPTACIÓN

Tabla 30. Caso prueba aceptación HU2_P2

Casos de pruebas de aceptación	
Código: HU2_P2	**Historia de usuario:** Búsqueda de movimiento (ancho_maximo, ancho_minimo, altura_maxima, altura_minima)
Nombre: Búsqueda de movimiento (ancho_maximo, ancho_minimo, altura_maxima, altura_minima)	
Descripción: Yo quiero realizar la búsqueda de detección de movimiento con los siguientes parámetros: ancho máximo, ancho mínimo, altura máxima, altura mínima.	
Condición de ejecución: ● El usuario debe estar autenticado en el sistema ● Debe estar el listado de las cámaras	
Entrada/Pasos de ejecución: ● El usuario accede al módulo de búsqueda forense ● El usuario visualiza el listado de las cámaras, ubicado en la parte izquierda de la pantalla ● El usuario selecciona una cámara ● El usuario selecciona la búsqueda de movimiento ● El usuario llena el campo de ancho_maximo ● El usuario llena el campo de ancho_minimo ● El usuario llena el campo de altura_maxima ● El usuario llena el campo de altura_minima ● El usuario selecciona el botón buscar	
Resultado de la prueba: El sistema muestra la realización de la búsqueda	
Evaluación de la prueba: Satisfactoria	

Tabla 31. Caso de Prueba de aceptación HU3_P3

Casos de pruebas de aceptación	
Código: HU3_P3	**Historia de usuario:** Búsqueda rostro (similitud, imagen)
Nombre: Búsqueda rostro (similitud, imagen)	
Descripción: Yo quiero realizar la búsqueda de detección de rostro con los siguientes parámetros: porciento de similitud y subir una imagen.	
Condición de ejecución: ● El usuario debe estar autenticado en el sistema	

● Debe estar seleccionada una cámara.
Entrada/Pasos de ejecución: ● El usuario accede al módulo de búsqueda forense ● El usuario visualiza el listado de las cámaras, ubicado en la parte izquierda de la pantalla ● El usuario selecciona una cámara ● El usuario selecciona la búsqueda de rostro ● El usuario llena el campo de porciento de similitud ● El usuario llena el campo de imagen ● El usuario selecciona el botón buscar
Resultado de la prueba: El sistema muestra la realización de la búsqueda
Evaluación de la prueba: Satisfactoria

Tabla 32. Caso de prueba de aceptación HU4_P4

Casos de pruebas de aceptación	
Código: HU4_P4	**Historia de usuario:** Búsqueda matrícula (similitud, matricula)
Nombre: Búsqueda matrícula (similitud, matricula)	
Descripción: Yo quiero realizar la búsqueda de detección de matrícula con los siguientes parámetros: porciento de similitud y agregar una matrícula.	
Condición de ejecución: ● El usuario debe estar autenticado en el sistema ● Debe estar seleccionada una cámara	
Entrada/Pasos de ejecución: ● El usuario accede al módulo de búsqueda forense ● El usuario visualiza el listado de las cámaras, ubicado en la parte izquierda de la pantalla ● El usuario selecciona una cámara ● El usuario selecciona la búsqueda de matrícula ● El usuario llena el campo de porciento de similitud ● El usuario llena el campo de matrícula ● El usuario selecciona el botón buscar	
Resultado de la prueba: El sistema muestra la realización de la búsqueda	
Evaluación de la prueba: Satisfactoria	

Tabla 33. Caso de prueba de aceptación HU5_P5

Casos de pruebas de aceptación	
Código: HU5_P5	**Historia de usuario:** Búsqueda objeto (similitud, listado_objeto)
Nombre: Búsqueda objeto (similitud, listado_objeto)	
Descripción: Yo quiero realizar la búsqueda de detección de objeto con los siguientes parámetros: porciento de similitud y seleccionar uno o varios objetos.	
Condición de ejecución: ● El usuario debe estar autenticado en el sistema ● Debe estar seleccionada una cámara	
Entrada/Pasos de ejecución: ● El usuario accede al módulo de búsqueda forense ● El usuario visualiza el listado de las cámaras, ubicado en la parte izquierda de la pantalla ● El usuario selecciona una cámara ● El usuario selecciona la búsqueda de objeto ● El usuario llena el campo de porciento de similitud ● El usuario llena el campo de seleccionar objeto(s) ● El usuario selecciona el botón buscar	
Resultado de la prueba: El sistema muestra la realización de la búsqueda	
Evaluación de la prueba: Satisfactoria	

Tabla 34. Caso de prueba de aceptación HU6_P6

Casos de pruebas de aceptación	
Código: HU6_P6	**Historia de usuario:** Filtrar la búsqueda (fecha_inicio, fecha_fin, hora_inicio, hora_fin)
Nombre: Filtrar la búsqueda (fecha_inicio, fecha_fin, hora_inicio, hora_fin)	
Descripción: Yo quiero que para realizar una búsqueda se filtren la fecha de inicio, la fecha de fin, la hora de inicio y la hora de fin.	
Condición de ejecución: ● El usuario debe estar autenticado en el sistema ● Debe estar seleccionada una cámara	
Entrada/Pasos de ejecución: ● El usuario accede al módulo de búsqueda forense ● El usuario visualiza el listado de las cámaras, ubicado en la parte izquierda de la pantalla ● El usuario selecciona una cámara	

- El usuario selecciona el tipo de búsqueda
- El usuario llena los campos requeridos por la búsqueda seleccionada
- El usuario llena el campo de fecha de inicio
- El usuario llena el campo de fecha de fin
- El usuario llena el campo de hora de inicio
- El usuario llena el campo de hora de fin
- El usuario selecciona el botón buscar

Resultado de la prueba: El sistema muestra los valores de los campos del filtrado

Evaluación de la prueba: Satisfactoria

Tabla 35. Caso de prueba de aceptación HU7_P7

Casos de pruebas de aceptación	
Código: HU7_P7	**Historia de usuario:** Mostrar video
Nombre: Mostrar video	
Descripción: Yo quiero que se muestre el video de las imágenes que se obtuvieron en la búsqueda.	
Condición de ejecución: ● El usuario debe estar autenticado en el sistema ● Debe estar seleccionada una cámara	
Entrada/Pasos de ejecución: ● El usuario accede al módulo de búsqueda forense ● El usuario visualiza el listado de las cámaras, ubicado en la parte izquierda de la pantalla ● El usuario selecciona una cámara ● El usuario selecciona el tipo de búsqueda ● El usuario llena los campos requeridos por la búsqueda seleccionada ● El usuario llena el campo de fecha de inicio ● El usuario llena el campo de fecha de fin ● El usuario llena el campo de hora de inicio ● El usuario llena el campo de hora de fin ● El usuario selecciona el botón buscar	
Resultado de la prueba: El sistema muestra los videos del resultado de la vista	
Evaluación de la prueba: Satisfactoria	

Tabla 36. Caso de prueba de aceptación HU8_P8

Casos de pruebas de aceptación	
Código: HU8_P8	**Historia de usuario:** Reproducir video
Nombre: Reproducir video	
Descripción: Yo quiero que al video mostrado se pueda pausar y continuar la reproducción.	
Condición de ejecución: ● El usuario debe estar autenticado en el sistema ● Debe estar seleccionada una cámara	
Entrada/Pasos de ejecución: ● El usuario accede al módulo de búsqueda forense ● El usuario visualiza el listado de las cámaras, ubicado en la parte izquierda de la pantalla ● El usuario selecciona una cámara ● El usuario selecciona el tipo de búsqueda ● El usuario llena los campos requeridos por la búsqueda seleccionada ● El usuario llena el campo de fecha de inicio ● El usuario llena el campo de fecha de fin ● El usuario llena el campo de hora de inicio ● El usuario llena el campo de hora de fin ● El usuario selecciona el botón buscar ● El usuario selecciona un video de los obtenidos en la búsqueda ● El usuario selecciona el botón de pause si quiere pausar el video ● El usuario selecciona el botón de play si quiere continuar la reproducción del video	
Resultado de la prueba: El sistema muestra la reproducción del video	
Evaluación de la prueba: Satisfactoria	