UNIVERSIDAD DE LAS CIENCIAS INFORMÁTICAS

Facultad de Ciencias y Tecnologías Computacionales

Ttulo: Sistema de Gestión de Activos Fijos Tangibles (AFT) para la emisora Radio Reloj.

TRABAJO DE DIPLOMA PARA OPTAR POR EL TÍTULO DE INGENIERO EN CIENCIAS INFORMÁTICAS

Autor: Rubert Marcos Maceo Garay

Tutores: MSc Yordanka Fuentes Castillo

Ing José Carlos Milanés Anaya

Universidad de las Ciencias Informáticas, Carretera a San Antonio Km 21⁄2. Torrens, Boyeros, La Habana. Cuba

Mayo 2023

"El exito es la capacidad de ir de un fracaso a otro sin perder el entusiasmo".

Winston Churchill.

Declaración de Autoría

Declaro por este medio que yo Rubert Marcos Maceo Garay, con carné de identidad 97072501881, autor principal del trabajo titulado “Sistema de Gestión de Activos Fijos Tangibles (AFT) para la emisora Radio Reloj.” y autorizo a la Universidad de las Ciencias Informáticas a hacer uso de la misma en su beneficio, así como los derechos patrimoniales con carácter exclusivo.

Para que así conste firmamos la presente a los_____días del mes de noviembre de 2023.

MSc Yordanka Fuentes Castillo Herrera

Ing José Carlos Milanés Anaya

Autor: Rubert Marcos Maceo Garay

Dedicatoria

A las grandes e inmensas estrellas de mi universo, mi madre y mi abuela, una por estar en el comienzo y largo proceso que se llama vida y la otra por saber suplir el lugar tan vacío que dejo mi madre, gracias por apoyar cada uno de mis pasos y mis deciciones, su entrega cariño y dedicación, pero sobre todo gracias por ayudarme a convertirme en una mejor persona.

De Rubert:

Gracias por darme la vida,

Mamá.

Agradecimientos

Quiero agradecer de forma especial primeramente a todos los presentes en el dia de hoy. Tribunal, oponentes y en especial a mi tutora la profe Yordanka que pese a todo lo cargada que tuvo en este proceso supo hacerse espacio para ayudar a este caso jjj, le agradezco que me tendio la mano cuando lo necesitaba que hasta mi facultad me había abandonado con el proceso de la tesis, realmente aprecio todo lo que hizo por mi.

Agradecer a unos grandes protagonistas que ya muchos no se encuentran aquí, ellos son los primeros y hasta día de hoy los mas grandes amigos, más que amigos hermanos que tuve. A Marquetti que ya podemos decir hermano que somos ingenieros la persistencia ganó, muchos momentos vividos entre los juegos de dominó a los de futbol hasta tarde en la noche, pero lo que , más resalto creo que fue aquel jonrón de Despaigne en el clásico que momento y sobre todo el !!RUBERT ACABA DE SUBIR!! Jajajaj.

AL Robe quien no quiere a ese hombre que siempre nos recibió en el 105, aquellas reuniones del grupo de amigos, el sagrado café, los clásicos Madrid vs Barca vividos y celebrados, el 9 a 1 en el FIFA pero siempre en partidos oficiales sacabas la cara jaja, nuestros partidos de futbol, mi gran utility del equipo que cuando el equipo se nos caía el asumía el rol que le otorgaran, siempre presente ante cualquier necesidad que prentara cualquiera de nuestro círculo de amistad realmente hizo agadable esta etapa de estudiante en la UCI.

A Ernesto el siempre rútico defensor de los demás, otro que era infaltable en las noches de futbol en esas que se reiniciaba a uno los pensamientos y era feliz, al mítico "Yo soy Pivooot jajaja", al que tanto jodi cada vez que le ganaba el Madrid a su barca, al que estuvo año y medio si ganarme un partido de FIFA y cuando lo logró pa que contarlo creo que hasta en el periódio salió, a él también le debo mucho en este trayecto.

A Daniel David Luna mi gran amigo y hermano creo que desde pequeños nos conocemos y quien diría que viviríamos tantas cosas en esta universidad fuiste y eres de las cosas mas grandes que me pasaron, otro intelectual de las noches de futbol, de los que siempre se preocupaba por mi y corría con aquellos insoportables dolores de estómago, el mas sentimental de todos pero al que todos más

queremos creo que los días junto a ti aquí fueron los mejores solo faltas tu por cumplir el objetivo de ser ingeniero espero que no desistas de ese sueño, hay mi negro donde esta mi negro jajaja mi rutina al saber que no estarías en el 105 el centro de altas reuniones jjajaja.

Al Lachy que siempre me acojió con el cuál también viví muchas cosas y en la cual puedo presumir que en los juegos MELLA no hubo en la pelota mejor equipo que esa CITEC y presumir de unos ORO dando la cara y siendo el faro de esa facultad, a ti hermano te agradezco mucho también.

Agradecer a Alani que me ayudó muchísimo tanto que era más fácil entender de ella que de cualquier otro profesor que me impartió alguna materia, al colorao gran amigo con el también se vivió grandes cosas en esta universidad.

Agradecer también a los que se unieron a lo largo de este proceso de ellos están Yensys, José Luis a Lissandra a la cuál le brinde ayuda cuando iba a comenzar sus pasos en el UCI y quién se iba a imaginar que terminaríamos graduándonos juntos e incluso hasta con el mismo tribunal cosas de la vida, resaltar que se convirtió en una de los amigos más sinceros y de máxima confianza.

Agraderle también a Heydis Vargas, a Grether, a Frank que fue de los últimos en incorporarse pero de los mas impotantes a él le agradezco mucho realmente mucho y sabe porqué eternamente agradecido, a Fabián otro de los que le debo mucho y en él que deposité mucha confianza, fue de también de los últimos en llegar pero de los pocos que se mantuvieron, de los que me ayudó cantidad y al que también estoy eternamente agradecido.

No quiero dejar de mencionar a ninguno pero son unos cuantos, a todos ellos agradecido estoy porque también compartieron muchas vivencias conmigo.

Agradecer a una persona que a día de hoy a dejado un poco la añoranza por la uci pero era de los que aún ya graduado siempre andaba por acá, una persona que empezó siendo un compañero más del aula pero terminó como un hermano de otra madre Omar Mario agradecerte a tí porque rápidamente llegaste a ser de los pocos que se mantuvo y de los que más aprecio y quiero, al que le peleaba por tener pocos minutos en los partidos pero de los que se que me hablaba con franqueza y creía en mí, sinceramente te considero para mí la persona mas importante de mi paso por la UCI después de mi novia claro jajaj, al que también estaré eternamente agradecido y ojalá que nuestra amistad no acabe nunca, al que el en su momento de agradecimientos me dejó pensar por algún momento que me

quedaría fuera pero tranquilo que no hice lo mismo contigo. A ti que compartimos bellos momentos, al que ya hoy puedo llamar colega, al que siempre me brindó una mano cuando lo necesité y gracias a la vida por permitir que una persona como tuyo tenga el placer de llamarle hermano.

Bueno ahora le toca a la persona principal la cual cuando me conoció encontraba y vida un poco descarrilada y me ayudó a que ese camino se arreglara, esa persona es mi novia Keylin no nos alcanzaría el tiempo para hablar de lo que significa ella para mí. Pero se les puede decir que me ayudó a superar los problemas que tuve, a cuando me caía la que me ayudó a levantarme y seguir luchando por el sueño que alcanzo hoy, la que no me permitió rendirme pese a todas las situaciones por las que pase, la que me hizo creer que si se podía y si se pudo, agradecerle por convertirse en el faro que me guía, agradecerle por soportarme, por siempre estar por y para mi disponible, por acogerme, por ayudarme a ser mejor persona, por convertirse en mi familia, a ella de todo corazón creo que le debo la mitad de este logro.

Quiero agradecer a mi papá también por depositar la confianza en mi y demostrarle que si podía alcanzar este logo y qu ella puede saber que tiene un hijo ingeniero.

Agradecerle a mi mamá que a dia de hoy no se encuentra con nosotros pero se que le hubiese gustado disfrutar este momento, un sueño que siempre me cultivo, cada momento pasado por acá siempre fue gracias a ella, todo mi esfuerzo de superación fue pensado por ella, agradecerle a ella porque se convirtió en mi modelo a seguir y estes donde estes te agradezco por haberme dado la vida y el cariño que tuve de ti esto es para ti, espero que desde alla arriba estes viendo este logro que he alcanzo y espero que te sientas orgullosa mamá, hoy tu hijo te extraña y siente mas aún tu ausencia pero se siente en alma tranquilo porque sabe que cumplió con el objetivo de hacerse alguien en esta vida, hoy hubiese querido poder compartir este logro contigo pero los motivos de la vida no me lo permitieron pero desde acá te mando un beso al cielo.

Y por último pero no por ello menos importante quiero agradecerme a mi por creer en mí, quiero agradecerme por hacer todo este trabajo duro, quiero agradecerme por no descansar ni un solo dia, quiero agradecerme por nunca renunciar, quiero agradecerme por dar siempre, intentando dar mas de lo que recibo, quiero agradecerme por intentar hacer lo correcto mas que lo incorrecto, quiero agradecerme por ser yo mismo todo el tiempo.

Gracias Dios, lo aprecio siempre, muchas gracias.

Resumen

El auge alcanzado por las Tecnologías de la Información y las Comunicaciones, ha contribuido al desarrollo en la mayoría de los sectores de la sociedad, aún más si se trata de gestionar, organizar y controlar las tareas que se realizan de forma cotidiana. A pesar de ello, en la emisora Radio Reloj la gestión de los activos fijos tangibles ubicados en sus diferentes áreas se realiza de forma manual. Esto conlleva a una gran cantidad de tiempo y esfuerzo en la realización del proceso, que resulta en errores, pérdida de información y dificulta la toma de decisiones. La presente investigación tiene como objetivo desarrollar una aplicación web para la gestión de los activos fijos tangibles en la emisora a fin de contribuir con la organización y control de la información de los activos fijos tangibles para la toma de decisiones. Para el desarrollo de la aplicación se utilizaron como ambiente de desarrollo los lenguajes UML, HTML5, CSS3, Java Script5, y las herramientas Visual Paradigm, Node JS, Mongo DB y Compass, Acunetix y Jmeter, todo bajo la guía de la metodología de desarrollo AUPvUCI. Para la validación se realizaron pruebas funcionales y no funcionales de seguridad, rendimiento y usabilidad a nivel de componente, sistema y aceptación, obteniéndose además un aval de satisfacción del cliente. Como resultado se obtuvo una aplicación web que permite al usuario autenticado, dependiendo de sus roles y permisos, gestionar los activos fijos tangibles de la emisora, moverlos entre departamentos y generar reportes para la toma de decisiones.

Palabras claves : activos fijos tangibles, emisora Radio Reloj, sistema de gestión, aplicación web

Abstract

The rise of Information and Communication Technologies has contributed to the development in most sectors of society, especially in managing, organizing, and controlling daily tasks. However, at Radio Reloj, the management of tangible fixed assets in its different areas is done manually. This leads to a significant amount of time and effort in the process, resulting in errors, loss of information, and hindering decision-making. This research aims to develop a web application for managing tangible fixed assets at the station to contribute to the organization and control of asset information for decision-making. The application was developed using UML, HTML5, CSS3, JavaScript, and tools such as Visual Paradigm, Node JS, MongoDB, Compass, Acunetix, and Jmeter, guided by the AUPvUCI development methodology. Functional and non-functional security, performance, and usability tests were conducted for validation at the component, system, and acceptance levels, along with obtaining customer satisfaction approval. The result is a web application that allows authenticated users, depending on their roles and permissions, to manage tangible fixed assets of the station, move them between departments, and generate reports for decision-making.

Keywords: tangible fixed assets, Radio Reloj station, management system, web application.

Índice de Figuras

Índice de tablas

Introducción:

Las grandes transformaciones económicas y sociales de los últimos tiempos han conformado un nuevo modelo socio-técnico, impulsado principalmente por un sorprendente desarrollo tecnológico. Estos eventos han provocado un obligatorio cambio en el propio contenido de la organización de la información en la gestión empresarial y su control.

El control es un plan donde se preveen todas las medidas administrativas dentro de la entidad, para el logro de los objetivos; de ahí que el control interno sea un plan de organización donde se establecen las políticas y procedimientos que persigue la entidad con el fin de salvaguardar los recursos con que cuenta. La efectividad del control interno depende en gran medida de la integridad y de los valores éticos del personal que diseña, administra y vigila este proceso en la entidad.

Es de suma importancia controlar ya que es la única manera de proteger los recursos contra pérdida, fraude o ineficiencia. Además de medir el cumplimiento de las tareas cerciorándose si ha habido adhesión a las políticas generales de la organización. De igual forma promover la exactitud o confiabilidad de los informes contables y administrativos. Es necesario establecer mecanismos de control para verificar la existencia real de los datos que se manejan y que dichos datos se puedan verificar o cotejar con el aspecto contable en cualquier momento o periodo con la satisfacción de que todo está en orden y al día.

Radio Reloj, una emisora de renombre en Cuba, que ha desempeñado un papel fundamental al proporcionar a los cubanos actualizaciones rápidas y precisas sobre los acontecimientos nacionales e internacionales. Con su característico "tic tac", Radio Reloj se ha convertido en un símbolo de la puntualidad y la fiabilidad informativa en Cuba. La emisora cuenta con una cadena nacional que abarca 22 transmisores en onda media y 1 en FM, y se dedica a informar sobre noticias, cultura, deportes y entretenimiento. En la misma que cuenta con 17 departamentos y una cierta cantidad de trabajadores los cuales en su mayoría periodistas. Dentro de esta institución se llevan acabo varios procesos en los cuales entra el control de los activos fijos tangibles, para lo cual existe un aparato administrativo, dirigido por un Director, un Subdirector, jefes de departamentos y Técnicos Generales. El área administrativa de la emisiora se dedica fundamentalmente al control del inventario de los AFT y aunque esto parezca sencillo y práctico es de gran dificultad y tiempo de trabajo para las personas que trabajan con dichos medios.

En la emisora Radio Reloj, existe un número elevado de AFT lo que trae como consecuencia que el proceso de búsqueda de las características y propiedades de los medios sea complejo y tedioso por lo que actualmente en esta área el control de estos medios se realiza de forma manual, con el uso de hojas de cálculo y el proceso de actualización cuando se realizan movimientos, se torna lento y engorroso. Además, debido a los constantes movimientos de medios entre departamentos, resulta difícil llevar un control de la información actualizada referida a las existencias de AFT. Estas características traen como consecuencia poca protección contra pérdida de información, lo que trae

consigo situaciones tales como: el fraude o la ineficiencia laboral del equipo de trabajo.

Todo esto trae como consecuencia demora y pérdida de datos a la hora de recuperar información dificultando la generación de reportes y las tomas de decisiones.

Por lo anterior planteado surge como problema de investigación: ¿Cómo contribuir a la gestión de los activos fijos tangibles de la emisora Radio Reloj?, definiéndose como objeto de estudio: La gestión de los activos fijos tangibles y campo de acción: La gestión de activos fijos tangibles en la emisora Radio Reloj mediante una aplicación web.

Para dar respuesta al problema se define como objetivo general: Desarrollar una aplicación web que contribuya a la gestión de los activos fijos tangibles en la emisora Radio Reloj, desglosado en los siguientes objetivos específicos.

- Elaborar el marco teórico de la investigación asociada al desarrollo de una aplicación web para la gestión de activos fijos tangibles para la emisora Radio Reloj.
- Realizar el análisis y diseño de la aplicación web para la gestión de los activos fijos tangibles para la emisora Radio Reloj.
- Implementar la solución para la gestión de los activos fijos tangibles para la emisora Radio Reloj.
- Evaluar la aplicación web para la gestión de los activos fijos tangibles para la emisora Radio Reloj a través de la estrategia de pruebas definidas para la gestión de los activos fijos tangibles para la emisora Radio Reloj.

Para dar cumplimiento a los objetivos se definen las siguientes tareas de investigación:

- Estudio de los referentes teóricos asociados a la gestión de activos fijos tangibles.
- Análisis de sistemas de gestión de activos fijos tangibles similares existentes a nivel mundial y nacional.
- Estudio y selección del ambiente tecnológico para el desarrollo del sistema de gestión de activos fijos tangibles.
- Análisis desde el punto de vista ingenieril de la aplicación web para la gestión de activos fijos tangibles para la emisora Radio Reloj.
- Diseño desde el punto de vista ingenieril de la aplicación web para la gestión de activos fijos tangibles para la emisora Radio Reloj.
- Implementación de la aplicación web para la gestión de activos fijos tangibles para la emisora Radio Reloj.
- Definición de la estrategia para validar la aplicación web para la gestión de activos fijos tangibles para la emisora Radio Reloj.
- Validación de la aplicación web para la gestión de activos fijos tangibles de la emisora Radio Reloj.

Métodos de Investigación Científicas:

Teóricos:

Histórico –Lógico:

Se utilizó para estudiar, de manera lógica, la evolución del proceso de gestión de activos fijos tangibles a través de sus principales elementos.

Analítico-sintético: A partir del mismo se estudiaron y sintetizaron los conceptos asociados al objeto de estudio de la investigación.

Modelación: Se utilizó para realizar los modelos y diagramas que sustentan el análisis y diseño de la aplicación web desarrollada.

Empíricos:

Entrevista:

Se utilizó para identificar las necesidades de la Dirección de la emisora Radio Reloj, que atiende y gestiona los medios básicos para obtener las funcionalidades con que contará el sistema informático.

Observación:

Permite constatar el estado de la gestión del proceso a partir de las prácticas cotidianas y la percepción directa en el panorama real.

El documento está estructurado de la siguiente manera: introducción, 3 capítulos, conclusión, recomendaciones, referencia bibliográficas y anexos. El desglose capitular queda de la siguiente manera:

Capítulo 1: “Fundamentación teórica de la aplicación web para la gestión de los activos fijos tangibles para la emisora Radio Reloj”: se muestra el resultado de la investigación bibliográfica sobre el objeto de estudio. Se estudian las tendencias en diseño y desarrollo actuales para las aplicaciones web. Se realiza un estudio de aplicaciones de gestion de activos fijos tangibles existentes y se detallan las tecnologías, metodologías y herramientas utilizadas, analizando sus características, desventajas y ventajas.

Capítulo 2: ”Análisis y diseño de la gestión de la aplicación web para la gestión de activos fijos tangibles de la Emisora Radio Reloj”: Se describe la propuesta de solución. Se presentan los elementos ingenieriles correspondientes al análisis y diseño de la aplicación web a desarrollar definiendo entre otros los requisitos funcionales y no funcionales, la arquitectura y la base de datos.

Capítulo 3: “Implementación y prueba de la aplicación web para la gestión de activos fijos tangibles de la emisora Radio Reloj”: Se presentan los elementos ingenieriles asociados a la solución implementada. Se define la estrategia de prueba para validar la solución y se muestran los resultados obtenidos de dicha validación.

CAPÍTULO 1: “Fundamentación teórica de la aplicación web para la gestión de los activos fijos tangibles para la emisora Radio Reloj”:

1.1 Introducción

En el presente capitulo se abordan los conceptos y elementos teoricos asociados a la gestión de activos fijos tangibles. Se realiza un estudio de sistemas de gestiones de activos fijos tangibles a nivel internacional y nacional se realiza un estudio y selección de los lenguajes y herramnientas que conforman el ambiente para desarrollar la aplicación. Finalmente se presentan las conclusiones del capítulo.

1.1.2 Fundamentación Teórica

En la llamada «Sociedad de la Información y el Conocimiento», internet ha generado un enorme interés en todos los ámbitos de nuestra sociedad. A finales de siglo XX y principios del XXI autores como Allepuz, Bueno, Bustelo, Castro, Fernández, García, Gil y Huang profundizan en la gestión del capital intelectual: nuevos parámetros de análisis para la economía de la información en la metodología del trabajo en equipo durante el proceso de gestión de información, en el tránsito de la sociedad de la información a la del conocimiento y la calidad de la información en los grupos de trabajo.

De acuerdo con varias fuentes consultadas sobre el tema de la gestión de la información prevalece en la teoría que la misma es un conjunto de procesos de las diferentes fases de la información, desde la obtención hasta el accionar en la toma de decisión, también se alude que son premisas que deben ser cumplidas por el capital humano que atiende su control en el área administrativa.

Entre los principales contenidos de la gestión de la información aparecen diferentes conceptos, tipos, se alude a su carácter sistémico englobador, dado por los diferentes elementos que comprende: procesos, clientes, agentes externos, proveedores, estrategia, propiedad intelectual y sistema; comprende entre otros contenidos la importancia en equipos y grupos de trabajo, al decir de (Bueno, 2010 & Sánchez, 2021). Además, se refieren varios principios, entre los que se relacionan con indicadores: identificar la información que se quiere medir con el objetivo, la utilidad para el destinatario, con qué variables medir y criterios de comparación; ventajas e inconvenientes que aparecen durante la gestión de la información, así como las implicaciones y el valor en su aplicación práctica. Del significado extraído de una información, cada individuo evalúa las consecuencias posibles y adecúa sus actitudes y acciones de manera acorde a las consecuencias previsibles que se deducen del significado de la información. Esto se refiere a qué reglas debe seguir el individuo o el sistema experto para modificar sus expectativas futuras sobre cada posible alternativa.

Otro concepto de información es el que la concibe como un conjunto organizado de datos procesados, que constituyen un mensaje que cambia el estado de conocimiento del sujeto o sistema que recibe dicho mensaje, como una medida de la complejidad de un conjunto de datos.

Los datos sensoriales una vez percibidos y procesados constituyen una información que cambia el estado de conocimiento, eso permite a los individuos o sistemas que poseen dicho estado nuevo de conocimiento tomar decisiones pertinentes acordes a dicho conocimiento.

A partir de que el dato es la unidad elemental de la información, de que cuando se anexa la estructura a este es que estamos en presencia de la información, es válido apuntar que esta información más la experiencia y el juicio es lo que permite situarse en el conocimiento en sí. Esta teoría lleva a los autores a identificarse con la gestión de ella y considerarla como un conjunto de actividades realizadas con el fin de controlar, almacenar y, posteriormente, recuperar adecuadamente la información producida, recibida o retenida por cualquier organización en el desarrollo de sus actividades. En el centro de la gestión de la información se encuentra la gestión de la documentación (la información que queda plasmada en documentos).

Relacionado con la gestión de la información en esta investigación existe un conjunto de conceptos, definiciones y teorías que se han considerado fuente importante para el desarrollo de la propuesta de solución que se presenta y a continuación serán considerados.

¿Qué son los sistemas de gestion de AFT?

Los sistemas de gestión de activos fijos tangibles son soluciones de software que automatizan el seguimiento, la recopilación y la gestión de los datos de activos, lo que permite una mayor eficiencia y precisión en la gestión de los activos (Iglesias, 2016).

Estos sistemas deben cumplir con ciertos requisitos para ser considerados efectivos, algunos de los cuales son:

- Deben permitir la identificación y seguimiento de los activos fijos tangibles.
- Deben permitir la gestión de la información de los activos, incluyendo su ubicación, estado, valor y vida útil.
- Deben permitir la programación y seguimiento de las actividades de mantenimiento y reparación de los activos.
- Deben permitir la generación de informes y análisis de los datos de los activos, lo que permite una mejor toma de decisiones y una mayor eficiencia en la gestión de los activos.

Además, los sistemas de gestión de activos fijos tangibles deben ser fáciles de usar y estar disponibles en múltiples plataformas, lo que permite una mayor accesibilidad y flexibilidad en la gestión de los activos.

¿Qué es un activo fijo?

Un activo fijo es un bien de la empresa que no puede convertirse en liquidez a corto plazo (Torres, 2020).

¿Qué son los AFT?

AFT (Activos Fijos Tangibles):

Representan propiedades físicamente tangibles que han de utilizarse por un período largo en las

operaciones regulares de la entidad y que normalmente no se destinan a la venta. Estos activos, con excepción del terreno y los animales productivos del ganado mayor, trasladan su valor paulatinamente, durante su vida útil, a la producción de bienes y a la prestación de servicios. Se les calcula una depreciación a su valor inicial a medida que disminuye su vida útil (Torres, 2020).
Se registran por su valor de adquisición y los gastos de transportación y montaje. En los casos de los adquiridos y en los ejecutados con medios propios, se valoran de acuerdo con las disposiciones vigentes (González, 2019).

La información es la base del trabajo que realiza el Área Administrativa de la emisora ya que según los datos de cada medio básico se decide su ubicación y destino como resultado de la información transmitida por esos datos. En la Dirección de Residencia, los AFT se manejan con las categorías de muebles y enseres. Se valoran al precio de adquisición, o a su costo real de elaboración o de producción o, en su caso, por un valor equivalente cuando se reciban bienes de uso sin contraprestación monetaria. El precio de adquisición incluye el precio neto pagado por los bienes, representado por el monto de efectivo entregado o su equivalente, más todos los gastos necesarios para colocarlos en el lugar y condiciones de uso, tales como fletes, seguros, derechos y gastos de importación y gastos de instalación, hasta su puesta en marcha, o momento de su alta (Mojena & González Díaz, 2016).
El valor de los bienes se actualiza contablemente por el registro de la depreciación, cuando corresponda, incluyéndose el importe de la misma en el resultado del ejercicio económico. Los bienes recibidos en donación sin que exista un valor de origen, deben ser contabilizados a un valor estimado que represente el desembolso que hubiera sido necesario efectuar para adquirirlos en las condiciones en que se reciben. Los bienes adquiridos en moneda extranjera se registrarán en la moneda nacional, aplicándose el tipo de cambio vigente en la fecha de adquisición. Los activos recibidos sin costo alguno o a un costo inadecuado se valoran al precio vigente de adquisición y de no existir este, por estimación efectuada por peritos. Esta regla de valoración también es aplicable a los bienes recibidos como aportaciones de capital (Mojena & González Díaz, 2016).

1.1.3 Proceso de planificación

Cabe recalcar, que la manera en la cual se realiza la gestión de los activos fijos tangibles en la emisora Radio Reloj es llamada por los encargados de ejercer esta actividad como proceso de planificación. El mismo comienza cuando el administrador recibe el listado de todos los activos fijos tangibles que posee la emisora, cuya información está almacenada en Excel. Una vez que el administrador tiene esta información, los activos fijos tangibles, su control y distribución quedan en sus manos. Los jefes de cada local tienen la responsabilidad de velar por estos activos, pero cualquier acción que se realice sobre ellos debe ser realizada por el administrador de la emisora, ya sea traslado, cambio o baja. Es importante destacar que la planificación y control de los activos

fijos tangibles es una tarea crítica para la emisora, ya que estos activos son esenciales para el funcionamiento de la misma. Por lo tanto, es fundamental que el administrador tenga un control riguroso sobre ellos y que los jefes de cada local colaboren en su mantenimiento y cuidado. En resumen, el proceso de planificación y control de los activos fijos tangibles en la emisora Radio Reloj es una tarea crítica que requiere una gestión rigurosa y eficiente por parte del administrador. La separación de responsabilidades entre el administrador y los jefes de cada local, así como la utilización de herramientas como Excel para el almacenamiento de información, son elementos clave en este proceso.

1.2 Análisis de soluciones existentes

Existen diversos sistemas de Gestión de activos fijos tangibles que cumplen funciones específicas en las entidades a las que se asocian. A pesar de que cada entidad funciona de manera particular, el funcionamiento de sus sistemas de gestión de información sirve de apoyo teórico para el cumplimiento del objetivo general de la presente investigación. A continuación de presenta el análisis de soluciones similares exitentes a nivel nacional e internacional.

1.2.1 Soluciones similares a nivel internacional

Universitas XXI – Económico

Es un sistema informático modular e integrado, antes llamado SOROLLA. Fue desarrollado por la Oficina de Cooperación Universitaria (OCU), con la ayuda de las universidades. El sistema está orientado a suministrar la información económica, financiera, tributaria, patrimonial, presupuestaria y analítica requerida tanto por los órganos de gobierno, gestión y control, como por los demás miembros de la comunidad universitaria de Zaragoza, España. Además del mantenimiento de las aplicaciones y el área de Gestión, cuenta con servicios de explotación de los datos para la obtención de indicadores y estadísticas, tanto los definidos para el sistema universitario como los requeridos con carácter interno para la toma de decisiones o la auditoria de procedimientos y funcionamiento administrativo (González, 2019).

Los módulos que utilizan las unidades de gasto descentralizadas son los siguientes:

- A través del módulo Gestión de Inventario, se realiza la gestión y el control completo de los activos fijos de la Universidad, muebles e inmuebles; así como el registro de bienes históricos e informáticos.
- Licencia privativa: El sistema cuenta con una licencia privativa, lo que significa que su uso, redistribución o modificación están prohibidos o requieren autorización.
- Dominio de aplicación web: El sistema se encuentra en un dominio de aplicación web, lo que sugiere que es accesible a través de un navegador web y puede ser utilizado en línea
- Acceso por roles para los permisos de usuario: Universitas XXI – Económico plantea un acceso por roles, lo que implica que los usuarios tienen diferentes permisos y funcionalidades

según su rol dentro del sistema.

- No es open source: El sistema no es de código abierto, lo que significa que no se rige por una licencia de software de código abierto que permita la modificación y redistribución libre del software.

Sistema de Control de Bienes Web- UG

Es un módulo que forma parte del Sistema Integral de Información Administrativa (SIIA) creado en la Universidad de Guanajuato (México), desarrollado con el propósito de apoyar a las unidades académicas y administrativas de la Universidad en el control de los bienes muebles e inmuebles bajo su cuidado. Es un sistema administrativo y sistemático que permite el registro oportuno, control e inventarios con información veraz, para poder garantizar el resguardo de los bienes y la identificación, registro y etiquetado de los bienes con eficacia, además de la custodia de los documentos que amparan la propiedad de los mismos (UG, 2023).

El sistema permite entre otras funcionalidades:

- Capturar y consultar los movimientos de los bienes y accesorios. Generar reportes en formato PDF con la información relevante de inventarios, movimientos y resguardos. Registrar los datos de los bienes y eliminar bienes en lotes. Imprimir Inventario General.
- Licencia privativa: El sistema cuenta con una licencia privativa, lo que significa que su uso, redistribución o modificación están prohibidos o requieren autorización.
- Dominio de aplicación web: El sistema se encuentra en un dominio de aplicación web, lo que sugiere que es accesible a través de un navegador web y puede ser utilizado en línea
- Acceso por roles para los permisos de usuario: No presenta un acceso por roles para los permisos de usuario. A diferencia de algunos sistemas, Sistema de Control de Bienes Web-UG no plantea un acceso por roles, lo que significa que los usuarios no tienen diferentes permisos y funcionalidades según su rol dentro del sistema.
- No es open source: El sistema no es de código abierto, lo que significa que no se rige por una licencia de software de código abierto que permita la modificación y redistribución libre del software.

Fundacite-Mérida

El sistema de Inventario y Notas de Pedidos, es un sistema de interés a la administración pública del Gobierno Bolivariano de Venezuela para la automatización en el manejo de Notas de Pedidos, Órdenes de Compra y el Inventario de Materiales, Mobiliario y Equipos.

El sistema permite entre otras funcionalidades:

- Generar Listas de Materiales y Suministros: Esta opción muestra un listado de todos los Materiales y suministros que se han ingresado en el inventario de Almacén, con el código presupuestario, nombre, existencia, unidad y el precio de cada artículo.

- Ingresar Nuevo Artículo al Inventario: permite ingresar artículos nuevos al inventario, cada artículo ingresado pertenece a un grupo de artículo por código presupuestario específico. Eliminar un Artículo del Inventario y notas de pedidos, modificar Inventario y reporte histórico en el cuál se muestra un listado de Órdenes de Compra realizadas.
- Licencia privativa: Opera bajo una licencia privativa, lo que implica restricciones en su uso, redistribución o modificación sin autorización.
- Dominio de aplicación web: Se encuentra en un dominio de aplicación web, lo que sugiere que es accesible a través de un navegador web y puede ser utilizado en línea.
- Acceso por roles para los permisos de usuario: Plantea un acceso por roles, lo que implica que los usuarios tienen diferentes permisos y funcionalidades según su rol dentro del sistema.
- No es open source: No es de código abierto, lo que limita la posibilidad de modificación y redistribución libre del software.

1.2.2 Soluciones similares existentes a nivel nacional

Sistema de Inventario del Patrimonio Cultural y Natural (SIP)

El SIP fue desarrollado desde 1988 por el Consejo Nacional de Patrimonio Cultural del Ministerio de Cultura de la República de Cuba, con el objetivo de normalizar los términos a emplear en el inventario automatizado del patrimonio cultural, con diversos fines:

- Cumplimentar la legislación nacional e internacional en lo que respecta al registro de los bienes patrimoniales. Ejercer el control y priorizar la conservación de los bienes patrimoniales más valiosos del país.
- Licencia privativa: El sistema opera bajo una licencia privativa, lo que implica restricciones en su uso, redistribución o modificación sin autorización.
- Dominio de aplicación web: Se encuentra en un dominio de aplicación web, lo que sugiere que es accesible a través de un navegador web y puede ser utilizado en línea.
- No plantea un acceso por roles para los permisos de usuario: A diferencia de algunos sistemas, el SIP no plantea un acceso por roles, lo que significa que los usuarios no tienen diferentes permisos y funcionalidades según su rol dentro del sistema.
- No es open source: El sistema no es de código abierto, lo que limita la posibilidad de modificación y redistribución libre del software.

Sistema Informativo para Ejecutivos (SIEWEB)

Fue desarrollado en el Instituto Superior Politécnico “José Antonio Echeverría”, CUJAE, con el objetivo principal de mantener informados a los diferentes directivos de la institución sobre la información almacenada por el sistema económico ASSETS.

Esta aplicación Web, implementada en el lenguaje PHP, sirve de interfaz entre el usuario y el sistema ASSETS, realizando consulta directamente a la base de datos del sistema económico y de esta forma permite:

- Mostrar los reportes que brinda el ASSETS mediante interfaz Web, facilitando la toma de decisiones de los directivos de la institución, visualizar la ubicación de los medios a nivel de área y mantener a los Jefes de Áreas y otros directivos informados de los medios existentes en la CUJAE.
- Licencia privativa: El sistema opera bajo una licencia privativa, lo que implica restricciones en su uso, redistribución o modificación sin autorización.
- Dominio de aplicación web: Se encuentra en un dominio de aplicación web, lo que sugiere que es accesible a través de un navegador web y puede ser utilizado en línea.
- No plantea un acceso por roles para los permisos de usuario: A diferencia de algunos sistemas, el SIEWEB no plantea un acceso por roles, lo que significa que los usuarios no tienen diferentes permisos y funcionalidades según su rol dentro del sistema.
- No es open source: El sistema no es de código abierto, lo que limita la posibilidad de modificación y redistribución libre del software.

Las aplicaciones homólogas de Cuba con respecto al sistema a desarrollar también se encuentran en la misma UCI. A continuación, se presentan los sistemas implementados en la UCI que más se asemejan al sistema propuesto.

Sistema de Gestión Integral (ASSETS)

Actualmente en la UCI se trabaja con el Sistema de Gestión Integral (ASSETS). Es un sistema multiusuario montado en una plataforma de servidores SQL, dividido en módulos económicos que trabajan en conjuntos para el control de las actividades económica, financiera y contable sobre los medios materiales y financieros.

ASSETS, es una aplicación cliente-servidor programada en Visual Basic 6.0 y Microsoft SQL Server 2000, utilizando adicionalmente Crystal Reports 7.0 para la generación de los reportes de salida. Al estar en plataforma SQL, garantiza mayor seguridad y consistencia en los datos, se obliga que sea ilimitado el número de usuarios conectados y hace posible la utilización de servidores remotos.

ASSETS, es un Sistema de Gestión Integral estándar y parametrizado que permite controlar, realizar y contabilizar todas las transacciones comunes relacionadas con el proceso de compra-venta, e incluye así mismo los procedimientos necesarios para registrar los movimientos de los activos fijos y de los útiles y herramientas; permite, además, administrar todos los medios de la empresa de los cuales preocupa tener información detallada, así como su ubicación física por centros de costo, áreas de responsabilidad y empleados, para evitar extravíos o facilitar su ubicación cuando se necesite.

Facilita, además, el ingreso al inventario y también la administración de los códigos asignados y todo el manejo internamente requerido.

Este sistema, aunque es eficiente no fue concebido en la UCI, no es cubano, aunque tiene licencia, no es gratuita y se debe pagar regularmente.

- Dominio de aplicación web: Se encuentra en un dominio de aplicación web, lo que sugiere que es accesible a través de un navegador web y puede ser utilizado en línea.
- Acceso por roles para los permisos de usuario: Plantea un acceso por roles, lo que implica que los usuarios tienen diferentes permisos y funcionalidades según su rol dentro del sistema.
- Open source: El sistema es de código abierto.

Resultado del Estudio Realizado

Luego del estudio de los sistemas homólogos se puede concluir que estos sistemas de gestión comparten ciertas características y objetivos comunes, cada una tiene su propia estructura, enfoque y énfasis.

Este estudio aportó:

- Saber que funcionalidades tienen esas aplicaciones y esos sistemas.
- La forma y el procedimiento que realizaban para efectuar estas funcionalidades.
- Que arquitecturas y tecnologías usaron.
- Como se lleva el proceso de gestión de los activos fijos en ellas.

Pero no se logró la incorporación de estas, porque respondían a modelos de negocio y maneras de realizar el proceso específico de cada institución por lo q no cumplian con la problemática planteada.

Los aspectos que se utilizaron para el estudio de sistemas homólogos en la gestión de activos fijos tangibles son los siguientes:

Licencia: se refiere a la autorización legal para utilizar un software. Es importante considerar la licencia del software utilizado para la gestión de activos fijos tangibles, ya que esto puede afectar la propiedad intelectual y la seguridad de los datos.

Dominio de aplicación: se refiere al ámbito en el que se utiliza el software. Es importante que el software utilizado para la gestión de activos fijos tangibles esté diseñado específicamente para este propósito y que tenga las funcionalidades necesarias para satisfacer las necesidades de la organización.

Open source: se refiere al software que se distribuye con su código fuente y que puede ser modificado y mejorado por la comunidad de desarrolladores. El uso de software de código abierto para la gestión de activos fijos tangibles puede ser beneficioso, ya que permite una mayor personalización y adaptación a las necesidades específicas de la organización.

Acceso por roles: se refiere a la capacidad de asignar diferentes niveles de acceso y permisos a

los usuarios del software. Es importante que el software utilizado para la gestión de activos fijos tangibles tenga esta funcionalidad para garantizar la seguridad y privacidad de los datos.

Generación de reportes: se refiere a la capacidad del software para generar informes y análisis de los datos de los activos fijos tangibles. Es importante que el software utilizado para la gestión de activos fijos tangibles tenga esta funcionalidad para permitir una toma de decisiones informada y eficiente.

Uno de los aspectos a tomar en cuenta para el desarrollo de la propuesta de solución es que el sistema debe de ser.

- Efectivo
- Seguro
- Personalizable para las necesidades específicas de la organización.

En la siguiente tabla se plasman aspectos esenciales que no deberían faltar en la propuesta de solución, donde el cliente también participó en el proceso de selección y evaluación de los criterios, y aprobó que los criterios seleccionados cumplían con sus requisitos y expectativas.

Tabla 1 Estudio de Homólogos

Sistemas usados		Análisis de tendencias	Generación de reportes de información de activos	Gestión de activos fijos tangibles
		Información que brindan		
Internacionales				
	Universitas XXI-Económico	transformación digital, soluciones en la nube, blockchain, metodologías ágiles y user experience.	**No**	A través del módulo Gestión de Inventario, se realiza la gestión y el control completo de los activos fijos de la Universidad
	Sistema de Control de Bienes Web-UG	mejora de la eficiencia y la productividad, tecnologías web y metodologías de análisis y diseño de sistemas	**Si**	Capturar y consultar los movimientos de los bienes y accesorios, movimientos y resguardos. Registrar los datos de los bienes y eliminar bienes en lotes.

	Fundacite-Mérida	enfoque en la promoción de la innovación, la ciencia y la tecnología en la región, el apoyo a la creatividad, la integración de actividades científicas y tecnológicas	**Si**	Esta opción muestra un listado de todos los Materiales y suministros que se han ingresado en el inventario de Almacén, con el código presupuestario, nombre, existencia, unidad y el precio de cada artículo. Ingresar Nuevo Artículo al Inventario: permite ingresar artículos nuevos al inventario, cada artículo ingresado pertenece a un grupo de artículo por código presupuestario específico. Eliminar un Artículo del Inventario y notas de pedidos, modificar Inventario
Nacionales				
	Sistema de Inventario del Patrimonio Cultural y Natural(SIP)	mejora de la eficiencia y la productividad	**No**	Cumplimentar la legislación nacional e internacional en lo que respecta al registro de los bienes patrimoniales. Ejercer el control y priorizar la conservación de los bienes patrimoniales más valiosos del país.
	Sistema Informativo para Ejecutivos (SIEWEB)	automatización de procesos y la mejora de la eficiencia y la productividad	**No**	mediante interfaz Web, facilitando la toma de decisiones de los directivos de la institución, visualizar la ubicación de los medios a nivel de área y mantener a los Jefes de Áreas y otros directivos informados de los medios existentes

				en la CUJAE.
	Sistema de Gestión Integral (ASSET)	mejora de la eficiencia y la productividad, integración de tecnologías avanzadas de análisis y en la implementación de una estrategia sólida de datos	**Si**	necesarios para registrar los movimientos de los activos fijos y de los útiles y herramientas; permite, además, administrar todos los medios de la empresa de los cuales preocupa tener información detallada, así como su ubicación física por centros de costo, áreas de responsabilidad y empleados, para evitar extravíos o facilitar su ubicación cuando se necesite.

Conclusión de los sistemas similares existentes:

Tras el estudio exhaustivo realizado, se ha llegado a la conclusión de que ninguno de los sistemas analizados cumple con todos los requerimientos definidos para la gestión de activos fijos tangibles, ya que no abarcan los procesos definidos en el marco teórico de la investigación. Sin embargo, algunas de las funcionalidades ofrecidas por estos sistemas, como el hecho de ser aplicaciones de dominio web, la forma en que gestionan los activos fijos tangibles, sus tendencias y la manera tal en que realizan los reportes de la información de dichos activos podrían ser aprovechadas. Los sistemas internacionales estudiados ofrecen funcionalidades que podrían ser útiles en el desarrollo de la solución; sin embargo, restringen el acceso a su código fuente y niegan el derecho a copiar la interfaz de usuario para su posterior modificación, lo que no cumple con el paradigma de independencia tecnológica que se busca promover en el país. En vista de lo anterior, ninguno de los sistemas analizados se considera como una solución viable, ya que no cumplen con los parámetros necesarios para ser considerados como tal. Esto resalta la necesidad de desarrollar una aplicación web que permita reducir el esfuerzo y el tiempo invertidos en la ejecución del proceso de gestión de activos fijos tangibles de la Radio Reloj. Además, cabe resaltar que, aunque el sistema ASSET cumple con los aspectos planteados, la emisora no cuenta con un presupuesto para adquirir los servicios de este sistema, también no se utliza para realizar este sistema basándome en utilizar o modificar el código de este sistema porque la relación tiempo-esfuerzo es muy corta, además a la hora de entender el código y modificar tomaría un extenso tiempo de

trabajo, entonces se opta por comenzar a realizar un nuevo sistema desde cero ya que sería más rápido y factible. Por lo tanto, se ha sugerido a la Universidad de Ciencias Informáticas que considere la creación de un sistema propio, con el fin de satisfacer las necesidades específicas de la emisora.

1.2 Entorno de desarrollo de la propuesta de solución

1.2.1 Estudio de las metodologías y estándares del desarrollo de software para el diseño del sistema informático a implementar

El desarrollo de los sistemas tradicionales de ciclo de vida se originó en la década de los años 60 del siglo XX para desarrollar a gran escala funcional de sistemas de negocio en una época de grandes conglomerados empresariales. La idea principal era continuar el desarrollo de los sistemas de información de una manera deliberada, estructurada y metódica, reiterando cada una de las etapas del ciclo de vida.

La Metodología de Desarrollo de Software tiene como objetivo presentar un conjunto de técnicas tradicionales y modernas de modelado de sistemas que permitan desarrollar software de calidad, incluyendo heurísticas de construcción y criterios de comparación de modelos de sistemas.

Para tal fin se describen, fundamentalmente, Herramientas de Análisis y Diseño Orientado a Objetos (UML), sus diagramas, especificación, y criterios de aplicación de las mismas. Como complemento se describirán las metodologías de desarrollo de software que utilizan dichas Herramientas, ciclos de vida asociados y discusión sobre el proceso de desarrollo de software más adecuado para las diferentes aplicaciones ejemplos que se presentarán. Principalmente, se presentará el Proceso Unificado, el cual utiliza un ciclo de vida iterativo e incremental.

Las metodologías de desarrollo de software denominadas ágiles se sustentan sobre la base de los valores y principios fijados en el Manifiesto Ágil por la Alianza Ágil. La propuesta de solución a desarrollar posee características que hacen que se ajuste a un proceso de desarrollo ágil:

1. La dimensión de la propuesta de solución es pequeña.
2. El equipo de desarrollo posee un tamaño pequeño.
3. El cliente forma parte del equipo de desarrollo.
4. Los requisitos pueden sufrir cambios constantemente a lo largo del proceso de desarrollo.
5. Se cuenta con un corto período de tiempo para el desarrollo.

Un ejemplo importante de metodología ágil es Agile Unified Process (AUP), en español traducido como Proceso Unificado Ágil de Scott Ambler. Esta metodología es una versión simplificada del Proceso Unificado de Rational (RUP), el cual describe de una manera simplificada la forma de desarrollar aplicaciones de software de negocio usando técnicas ágiles y conceptos que aún se mantienen válidos en RUP. AUP aplica técnicas ágiles incluyendo Desarrollo Dirigido por Pruebas.

La metodología AUP tiene varias ventajas como simplicidad, agilidad, centralización en tareas de alto valor, independencia de Herramienta fácil adaptación. Esta metodología se encuentra en plena ejecución en la UCI, debido a que debe existir una uniformidad en la ejecución de metodologías en los proyectos de la Universidad. El entorno del negocio exige que los cambios de requisitos sobre la marcha sean un aspecto natural, inevitable e incluso deseable en el desarrollo del sistema, y en el caso de AUP-UCI no plantea nada acerca de la variabilidad de los requisitos en el escenario No.2 (es el escenario que se ajusta al negocio) . Este escenario se emplea mayormente para proyectos donde el objetivo primario es la gestión y presentación de información.

Apunta a una alta integración con el usuario. La idea es ofrecer muchas entregas del sistema agregando funcionalidades paulatinamente en lapsos cortos de tiempo, esto permite que el usuario tenga más claro si el sistema hace lo que él quiere, además de comprometerlo en el desarrollo (Mojena & González Díaz, 2016).

1.2.2 Metodología AUP-UCI

Las metodologías de desarrollo de software son un marco de trabajo eficiente que surgió en la década de los años 70, pues ofrecían una respuesta a los problemas que surgían con los antiguos métodos de desarrollo.

Con la implementación de estas, se ha logrado mejorar de manera significativa las aplicaciones web, debido a que proveen una guía compuesta por etapas y procesos efectivos que permiten obtener mejores resultados y de calidad. Son muchas las metodologías que existen como RUP, XP, SCRUM, AUP entre otras como OPEN UP. La metodología AUP-UCI es una variante de AUP adaptada a lo que la Universidad UCI ha estado proponiendo como ciclo de vida de los proyectos. La investigación actual se ajusta a del escenario No.2 de esta metodología. Esta Aplica a los proyectos que hayan evaluado el negocio a informatizar y como resultado obtengan que no es necesario incluir las responsabilidades de las personas que ejecutan las actividades(Rivera, 2022).

Se escoge esta metodología ya que:

- Se adapta al ciclo de vida definido para la actividad productiva de la UCI.
- El cliente estará siempre acompañando al equipo de desarrollo para convenir los detalles de los requisitos y así poder implementarlos, probarlos y validarlos.
- Describe de manera simple la forma de desarrollar el sistema.
- Su uso permite una descripción concisa utilizando poca documentación.

Herramientas y tecnologías para el desarrollo:

Las herramientas informáticas son programas, aplicaciones o simplemente instrucciones usadas para efectuar tareas de modo más sencillo (González, 2019). Con el objetivo de minimizar los costos, se propone utilizar tecnologías y herramientas que permitan su uso sin necesidad de pago de licencias. A continuación, se hace un estudio y se comparan las herramientas para seleccionar

las que se usarán en la solución propuesta.

1.2.3 Lenguaje de modelado

UML (Unified Modeling Language) fue adoptado como estándar del Object Management Group (Grupo Gestor de Objetos) en 1997 debido a que representa una colección de las mejores prácticas de ingeniería que han sido probadas con éxito en el modelado de sistemas. Es un lenguaje para la especificación, visualización, construcción y documentación de sistemas, no solo de software. Su utilización es independiente del lenguaje de programación y de las características de los proyectos, pues ha sido diseñado para modelar cualquier tipo de soluciones informáticas, arquitectura o cualquier otra rama (González, 2019). En resumen, UML puede contribuir a una mejor investigación, una mejor comunicación, una mejor comprensión del sistema, un mejor diseño y desarrollo eficiente. Además de facilitar una documentación efectiva y promover la estandarización.

1.3 Herramienta de modelado

Visual Paradigm for UML es una herramienta CASE que soporta el ciclo de vida completo del desarrollo de software: análisis y diseño orientados a objetos, implementación y pruebas. Ayuda a una rápida construcción de aplicaciones de calidad, mejores y a un menor coste. Permite construir diagramas de diversos tipos, código inverso, generar código desde diagramas y generar documentación. La herramienta UML CASE también proporciona abundantes tutoriales de UML, demostraciones interactivas de UML y proyectos UML (Visual Paradigm., 2015). Por las razones antes expuestas se emplea esta herramienta de modelado para la propuesta de solución en su versión 8 .0 (Hernández, 2016).

En resumen, Visual Paradigm puede ser una herramienta valiosa para el campo de la informática, ya que puede ayudar a mejorar la eficiencia y la calidad del proceso de investigación y desarrollo de software. Aportando una mejora de la comunicación y comprensión del sistema, documentación efectiva y una personalización del proceso de acuerdo a las necesidades del usuario.

1.4 Tendencias y tecnologías actuales. Selección de las Herramientas y lenguajes de desarrollo

En la actualidad contar con un sitio web es de suma importancia, independientemente del objetivo del sitio, cada minuto que pasa nuestra sociedad está actualizando información de lo que sucede en diferentes lugares del mundo, y en este siglo XXI es de vital importancia estar informado de todos los acontecimientos que nos rodean.

Se determinó desarrollar una **aplicación web** debido a las necesidades del cliente, las cuales requerían versatilidad en el movimiento de datos, y las aplicaciónes web tienen las siguientes ventajas que se acoplan a las características de la Dirección de Residencia:

La Compatibilidad multiplataforma que comprenden las aplicaciones web tienen un camino mucho más sencillo para la compatibilidad multiplataforma que las aplicaciones de software descargables. Varias tecnologías incluyendo Java, Flash, ASP y Ajax permiten un desarrollo efectivo de programas soportando todos los sistemas operativos principales.

Las aplicaciones web exigen menos requerimientos de memoria, tienen muchas más razonables demandas de memoria RAM de parte del usuario final que los programas instalados localmente. Al residir y correr en los servidores del proveedor, a esas aplicaciones basadas en web donde se usa en muchos casos la memoria de las computadoras que ellos corren, dejando más espacio para correr múltiples aplicaciones al mismo tiempo sin incurrir en frustrantes deterioros en el rendimiento.

Estas aplicaciones presentan menos Bugs y deberían ser menos propensas a colgarse y crear problemas técnicos debido a software o conflictos de hardware con otras aplicaciones existentes, protocolos o software personal interno. Con aplicaciones web, todos utilizan la misma versión, y todos los bugs pueden ser corregidos tan pronto como son descubiertos. Esta es la razón por la cual las aplicaciones basadas en web deberían tener mucho menos bugs que el software de escritorio descargable tradicional.

Las páginas web tienen como características importantes las siguientes:

- Presentan contenidos de utilidad ya que la temática y las búsquedas deben ir acordes con el usuario final, si no, lo más seguro es que el visitante abandone la página rápidamente.
- Debe ser intuitivo haciendo fácil la navegación generando que las visitas consigan fácilmente su objetivo. Además, evita abandonos en tu página web.
- Realización un diseño atractivo. En una web, una primera impresión cuenta, ya que representa nuestro negocio, e incluso a nosotros mismos. Si a un usuario no le gusta lo que ve, o no le genera confianza, lo más probable es que abandone el sitio web, en muchos casos sin tan siquiera tener en cuenta el contenido. Por lo cual un buen diseño genera confianza, seriedad y muy buena impresión.
- Creación de un contenido bien estructurado con un desglose del contenido claro y sencillo, bien explicado. La importancia de una estructura clara y objetiva es lo que mantendrá a tu usuario conectado (Mojena & González Díaz, 2016).

1.4.1 Estudio y selección del framework a utilizar

Para el desarrollo de la aplicación es necesario el uso de los frameworks, y primero hay que entender su significado y para qué sirve. En el desarrollo de software, un framework o infraestructura digital, es una estructura conceptual y tecnológica de soporte definido, normalmente con artefactos o módulos concretos de software, que puede servir de base para la organización y desarrollo de software. Típicamente, puede incluir soporte de programas, bibliotecas, y un lenguaje interpretado, entre otras Herramientas, para así ayudar a desarrollar y unir los diferentes

componentes de un proyecto.

Los frameworks tienen como objetivo principal ofrecer una funcionalidad definida, auto contenido, siendo construidos usando patrones de diseño, y su característica principal es su alta cohesión y bajo acoplamiento. Para acceder a esa funcionalidad, se construyen piezas, objetos, llamados objetos calientes, que vinculan las necesidades del sistema con la funcionalidad que este presta. Esta funcionalidad, está constituida por objetos llamados fríos, que sufren poco o ningún cambio en la vida del framework, permitiendo la portabilidad entre distintos sistemas (Mojena & González Díaz, 2016). Según la información anterior se puede hacer un bosquejo de algunos de los frameworks más conocidos en el mundo:

Framework Express.JS

Express.js es un framework web mínimo y flexible para Node.js (siendo el estándar de facto para servidores web de Node.js) que proporciona un conjunto robusto de características para aplicaciones web y móviles. Express proporciona una capa delgada de características fundamentales de aplicaciones web. Facilita la creación de API's gracias la gran variedad de métodos HTTP y middleware que proporciona. Además que nos aporta la creación de aplicaciones web de manera rápida y sencilla, ser muy ligero y flexible, lo que lo hace ideal para proyectos pequeños y medianos, ofrecer una sintaxis clara y concisa que facilita la creación de aplicaciones web y ser muy rápido y eficiente, lo que lo hace ideal para aplicaciones web de alto rendimiento. (Carreño, 2016).

Koa es un marco web diseñado para ser una base más pequeña, más expresiva y más robusta para aplicaciones web y APIs. Es minimalista y flexible, lo que permite a los desarrolladores crear middleware personalizado y aplicaciones escalables. Koa utiliza funciones asíncronas, lo que permite a los desarrolladores prescindir de devoluciones de llamada y mejorar significativamente el manejo de errores. A diferencia de otros marcos, Koa no incluye ningún middleware en su núcleo, pero proporciona un conjunto elegante de métodos que facilitan la escritura rápida y placentera de servidores.

Algunas ventajas de usar Koa incluyen:

Simplicidad: Koa sigue el principio de "hacer una cosa y hacerla bien". Proporciona una API simple y elegante para manejar solicitudes HTTP y middleware, lo que facilita la construcción de aplicaciones web pequeñas y rápidas.

Middleware basado en generadores: Koa utiliza generadores de JavaScript para manejar el middleware, lo que permite un flujo de control más claro y conciso, mejorando la legibilidad del código.

Extensibilidad y personalización: Koa es altamente extensible y personalizable. Los desarrolladores pueden agregar middleware de terceros para añadir funcionalidades adicionales a

su aplicación, y elegir qué funcionalidades específicas de Koa quieren utilizar, lo que les brinda un mayor control sobre el tamaño y la configuración de su proyecto (Demashov & Gosudarev, 2019).

Sails.js es un framework web de Node.js que se basa en el patrón de arquitectura MVC (Modelo-Vista-Controlador) y está diseñado para simplificar el proceso de desarrollo de aplicaciones web y APIs. Sails.js proporciona una amplia gama de características, como enrutamiento, controladores, modelos, políticas y servicios, lo que permite a los desarrolladores crear aplicaciones escalables y mantenibles de manera rápida y eficiente.
Algunas ventajas de usar Sails.js incluyen:
Productividad: Sails.js es un framework muy productivo que permite a los desarrolladores crear aplicaciones web y APIs de manera rápida y eficiente. Proporciona una amplia gama de características y herramientas que simplifican el proceso de desarrollo
Escalabilidad: Sails.js es altamente escalable y puede manejar grandes volúmenes de tráfico y datos. Utiliza una arquitectura basada en eventos que permite a los desarrolladores crear aplicaciones escalables y de alto rendimiento
Flexibilidad: Sails.js es un framework muy flexible que permite a los desarrolladores personalizar y extender su funcionalidad. Proporciona una amplia gama de plugins y módulos que pueden ser utilizados para agregar funcionalidades adicionales a la aplicación (Demashov & Gosudarev, 2019).

Meteor es un framework web de JavaScript de pila completa para desarrollar aplicaciones web y móviles modernas. Incluye un conjunto clave de tecnologías para construir aplicaciones reactivas conectadas, una herramienta de compilación y un conjunto seleccionado de paquetes de la comunidad de Node.js y JavaScript en general. Meteor permite desarrollar en un solo lenguaje, JavaScript, en todos los entornos: servidor de aplicaciones, navegador web y dispositivo móvil. Utiliza "datos en el cable", lo que significa que el servidor envía datos, no HTML, y el cliente los renderiza. Meteor abraza el ecosistema, llevando las mejores partes de la comunidad JavaScript extremadamente activa de manera cuidadosa y considerada. Proporciona reactividad de pila completa, lo que permite que la interfaz de usuario refleje sin problemas el verdadero estado del mundo con un esfuerzo de desarrollo mínimo Algunas ventajas de usar Meteor incluyen:
Productividad: Meteor es altamente productivo, ya que permite a los desarrolladores crear aplicaciones web y móviles de manera rápida y eficiente. Al proporcionar un conjunto clave de tecnologías y una herramienta de compilación, simplifica el proceso de desarrollo.
Reactividad de pila completa: Meteor proporciona reactividad de pila completa, lo que permite que la interfaz de usuario refleje sin problemas el verdadero estado del mundo con un esfuerzo de desarrollo mínimo. Esto facilita la creación de aplicaciones reactivas y conectadas.
JavaScript en todos los entornos: Meteor permite desarrollar en un solo lenguaje, JavaScript, en

todos los entornos, lo que simplifica el desarrollo y la mantenibilidad de las aplicaciones (Ramírez, 2018).

NestJS es un framework de aplicaciones web de Node.js que utiliza TypeScript y está diseñado para crear aplicaciones escalables y eficientes. Se basa en los principios de Angular y proporciona una arquitectura modular y flexible que permite a los desarrolladores construir aplicaciones robustas y mantenibles. Algunas de las ventajas de usar NestJS incluyen su capacidad para aprovechar las características de TypeScript, su enfoque en la modularidad y la reutilización de código, y su soporte para la creación de APIs RESTful y GraphQL (Desamsetti & Dekkati, 2021).

Algunas de las ventajas del uso de NestJS incluyen:

Arquitectura modular y flexible: Gracias a su arquitectura modular, NestJS es flexible y permite utilizar otras bibliotecas y herramientas, lo que facilita la extensibilidad y la reutilización de código.

Productividad y mantenibilidad: NestJS proporciona una arquitectura de proyecto simple pero efectiva, lo que facilita tanto la escalabilidad del proyecto como su mantenimiento. Esto agiliza el desarrollo de aplicaciones backend y API, permitiendo un mantenimiento más sencillo a largo plazo.

Compatibilidad con TypeScript: NestJS está completamente compatible con TypeScript, lo que permite a los desarrolladores aprovechar las características de este lenguaje para crear aplicaciones backend eficientes, confiables y escalables.

Documentación exhaustiva: NestJS cuenta con una documentación detallada y fácil de entender, lo que ahorra tiempo de depuración al proporcionar soluciones claras a los problemas.

Después del estudio de algunos de los frameworks más utilizados por el entorno Node JS, ver sus cualidades y algunas de sus ventajas se muestra a continuación el por qué se escoge el framework Express JS por delante del resto de los frameworks estudiados.

Por qué utilizar Express JS y no otro:

Según los datos recopilados, Express.js destaca sobre Koa, Nest.js, Meteor y Sails.js (Heredia, J. S., & Sailema), por varias razones:

Popularidad y comunidad activa: Express.js es uno de los frameworks más populares y utilizados de Node.js, con una amplia comunidad de desarrolladores y una gran cantidad de recursos disponibles. Esto significa que es más probable encontrar soluciones a problemas comunes y recibir soporte de la comunidad.

Madurez y estabilidad: Express.js es un framework maduro y estable, con un largo historial de uso en aplicaciones de producción. Esto garantiza que es una opción confiable para el desarrollo de proyectos a largo plazo.

Amplia gama de características: Express.js proporciona una amplia gama de características para la construcción de aplicaciones web, incluyendo enrutamiento, middleware y plantillas. Su versatilidad y flexibilidad lo hacen adecuado para una variedad de aplicaciones y casos de uso.

Facilidad de uso y escalabilidad: Express.js es conocido por su simplicidad, flexibilidad y escalabilidad. Estas características lo hacen ideal tanto para proyectos pequeños como para aplicaciones web más complejas y de gran escala.

En resumen, Express.js destaca por su popularidad, madurez, estabilidad, amplia gama de características, facilidad de uso y escalabilidad, lo que lo convierte en una opción sólida para el desarrollo de proyectos web y API en comparación con Koa, Nest.js, Meteor y Sails.js.

1.4.2 Lenguaje de programación

Lenguaje del lado del servidor

Node.js:

La elección de Node.js como lenguaje de programación para el desarrollo del proyecto no es sólo el utilizar un solo lenguaje para programar tanto el cliente como el servidor, no es más que una ventaja. El hecho definitivo es la naturaleza propia de este entorno de ejecución, Node.js es concebido como un entorno de ejecución de JavaScript orientado a eventos asíncronos, diseñado para construir aplicaciones en red escalables y de rápida comunicación y ejecución de las entradas y salidas.

Según la web oficial: Node.js® es un entorno de ejecución para JavaScript construido con el motor de JavaScript V8 de Chrome. Node.js usa un modelo de operaciones E/S sin bloqueo y orientado a eventos, que lo hace liviano y eficiente. El ecosistema de paquetes de Node.js, npm, es el ecosistema más grande de librerías de código abierto en el mundo. La característica más importante de NodeJS, y que ahora otra serie de lenguajes están aplicando, es la de no ser bloqueante, es decir, si durante la ejecución de un programa hay peticiones que requieren un tiempo para producirse la respuesta, Node no detiene el hilo de ejecución del programa esperando que esa parte acabe, sino que continúa procesando las siguientes instrucciones que no requieren los resultados del proceso lento. Cuando este termina, realízalas instrucciones que fueran definidas para realizar con los resultados recibidos. El repositorio de módulos NPM de Node.js no solo es grande, también es de fácil uso. Este puede ser instalado conjuntamente con Node, y la instalación de los módulos es automatizada, como se verá más adelante, con un archivo de configuración incluido en el proyecto (Rodríguez, 2017).

Node.js se destaca por su ligereza, flexibilidad y eficiencia, lo que lo hace ideal para proyectos pequeños y medianos que requieren aplicaciones de red y del lado del servidor. Además, cuenta con una gran comunidad de desarrolladores y es muy fácil de aprender y utilizar. En comparación con otros entornos de tiempo de ejecución, Node.js se diferencia por su capacidad para manejar aplicaciones en tiempo real y su arquitectura de E/S basada en eventos y sin bloqueos.

Lenguajes del lado del cliente

JavaScript

JavaScript es un lenguaje de programación (de los denominados lenguajes de scripting (script se traduce como guión, literalmente)) que se utiliza principalmente para crear páginas web dinámicas, con algunos efectos realmente interesantes y que mejoren considerablemente su aspecto. Está diseñado para ser usado en conjunción con HTML, 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 (EGUILUZ, 2020). Es sencillo y rápido, los resultados pueden ser muy satisfactorios y aunque el lenguaje tenga algunas limitaciones, permite al programador controlar lo que ocurre en la página. Se utiliza JavaScript para la implementación de la lógica del negocio. Permite escribir código fuente que será analizado por un ordenador.

JavaScript es un lenguaje de programación que se utiliza para añadir características interactivas a los sitios web. Algunas de las razones por las que se recomienda su uso son:

- Es un lenguaje interpretado, lo que significa que no necesita ser compilado antes de ser ejecutado.
- Es un lenguaje de programación orientado a objetos, lo que lo hace muy flexible y escalable.
- Es un lenguaje muy popular y ampliamente utilizado, lo que significa que hay una gran cantidad de recursos disponibles para los desarrolladores que lo utilizan.
- Es un lenguaje que se ejecuta en el navegador, lo que significa que no es necesario instalar ningún software adicional para utilizarlo.
- Es un lenguaje que se puede utilizar tanto en el lado del cliente como en el lado del servidor, lo que lo hace muy versátil.

En resumen, JavaScript es un lenguaje de programación muy útil para añadir interactividad y dinamismo a los sitios web. Es fácil de aprender y utilizar, y cuenta con una gran cantidad de recursos disponibles para los desarrolladores. Además, se puede utilizar tanto en el lado del cliente como en el lado del servidor, lo que lo hace muy versátil y escalable (EGUÍLIZ, 2020).

1.4.3 Selección del framework CSS

Para la realización del diseño de una aplicación es necesario utilizar **un framework de diseño o CSS**, se realizó una investigación con algunos de los frameworks CSS más usados en el mundo (Mojena & González, 2016).

El framework Gumby fue desarrollado a la base de Saas, que es un pre-procesador potente de CSS. Actualmente Gumby está disponible en la versión 2.6 con muchas características Útiles, estilos predefinidos y elementos de interfaz comunes, tales como palancas, menús desplegables, botones, pestañas etc. Con la ayuda de Gumby, se puede personalizar y formar una cuadrícula (grid) para su diseño. Por otra parte, se puede adaptar a cualquier tamaño y resolución de pantalla(Mojena & González, 2016).

En el caso de **Kube**, es adaptable y compatible con todo lo necesario para el diseño del sitio web. Presenta la mayoría de las características de Gumby, pero a diferencia del anterior tiene una cuadrícula (grid) flexible revolucionaria con asombrosa tipografía. (Mojena & González, 2016).

La biblioteca **YAML** es ideal para los sitios web flexibles, accesibles y adaptables (responsive). El framework ofrece un conjunto de bloques para la construcción de sitios web. Los elementos incluyen la funcionalidad de tipografía, la navegación, los formularios etc. Este framework contiene un buen Ajax constructor que es una herramienta muy útil para el desarrollo visual del diseño CSS (Mojena & González, 2016). A pesar de las buenas propiedades de los frameworks anteriormente mencionados se ha elegido como el indicado para el desarrollo de la aplicación el siguiente:

El framework o conjunto de Herramientas de software libre Bootstrap en su versión 3.3.6 es para el diseño de sitios y aplicaciones web. Contiene plantillas de diseño con tipografía, formularios, botones, cuadros, menús de navegación y otros elementos de diseño basado en HTML y CSS, así como, extensiones de JavaScript opcionales adicionales (Mojena & González, 2016). Tiene las siguientes ventajas:

- Ofrece un paquete de elementos web personalizables: con Bootstrap se puede diseñar una web jugando con sus elementos compuestos por diferentes combinaciones de HTML, CSS y Javascript, de manera que las piezas siempre encajan.
- Utiliza componentes vitales para los desarrolladores: como HTML5, CSS3, jQuery o GitHub, entre otros.
- Sus plantillas son de sencilla adaptación. Se desarrolló con la idea de facilitar el proceso de adaptación web a todo tipo de dispositivos.
- Incluye Grid system: muy útil para maquetar por columnas.
- Se integra con librerías JavaScript.
- Usa Less: un lenguaje de las hojas de estilo CSS preparado para enriquecer los estilos de la web.
- Es una herramienta de uso ágil y sencillo: facilita enormemente el diseño de interfaces y además incluye por defecto una plantilla bastante optimizada.
- De manera general Bootstrap es perfecto para los diseñadores de todos los niveles, los proyectos de todos los tamaños, así como dispositivos de todas las formas (Mojena & González, 2016).

HTML5

HTML5 (HyperText Markup Language, versión 5) es la quinta revisión del lenguaje HTML. Esta nueva versión y enconjunto con CSS3, define los nuevos estándares de desarrollo web, al rediseñar el código para resolver problemas y actualizándolo así a nuevas necesidades. El uso del mismo para el desarrollo de aplicaciones virtuales está cada vez más extendido. HTML5 es una

revisión del lenguaje de marcado HTML y regulado por el Consorcio W3C7. HTML5 presenta una serie desventajas con respecto a lenguajes de marcado previos y otras herramientas para el desarrollo de aplicaciones virtuales, entre las que destacan las siguientes: No requiere del uso de plugins8 ni de APIs (Application Program Interfaces) de terceros. Tiene incorporada nuevas características que permite diseñar aplicaciones adaptables a diferentes dispositivos móviles, tales como webs, móviles y tabletas. Incluye nuevas etiquetas de video, audio y cambas. Esta última en particular proporciona más efectos visuales. El código de programación es más simple. Compatible con todos los navegadores.

HTML (HyperText Markup Language) es un lenguaje de marcado que se utiliza para estructurar y dar significado al contenido de una página web. Algunas de las razones por las que se recomienda su uso son:

Permite estructurar la información y los contenidos de una página web de manera clara y organizada.

Ofrece una gran cantidad de etiquetas y atributos predefinidos, lo que permite crear páginas web con diferentes tipos de contenido.

Es compatible con todos los navegadores web modernos, lo que garantiza que el sitio web se verá igual en todos los dispositivos.

Es fácil de aprender y utilizar, lo que lo hace ideal para desarrolladores principiantes y experimentados por igual.

Cuenta con una gran cantidad de recursos disponibles, como tutoriales y ejemplos, que facilitan su uso y aprendizaje.

En resumen, HTML es un lenguaje de marcado muy útil para estructurar y dar significado al contenido de una página web. Es fácil de aprender y utilizar, y cuenta con una gran cantidad de recursos. Además, permite crear páginas web personalizadas y compatibles con todos los navegadores web modernos (González, (2019).

1.4.4 Entorno de desarrollo integrado

Un ambiente de desarrollo integrado o entorno de desarrollo interactivo, IDE27, es una aplicación informática que proporciona servicios integrales para facilitarle al desarrollador o programador el desarrollo de software. Normalmente, un IDE consiste de un editor de código fuente, Herramientas de construcción automáticas y un depurador. La mayoría de los IDE tienen auto-completado inteligente de código (IntelliSense). Algunos IDE contienen un compilador, un intérprete, o ambos (Mojena & González, 2016).

El límite entre un IDE y otras partes del entorno de desarrollo de software más amplio no está bien definido. Muchas veces, a los efectos de simplificar la construcción de la interfaz gráfica de usuario (GUI, por sus siglas en inglés) se integran un sistema controlador de versión y varias Herramientas. Muchos IDE modernos también cuentan con un navegador de clases, un buscador de objetos y un

diagrama de jerarquía de clases, para su uso con el desarrollo de software orientado a objetos (Mojena & González, 2016). Existen muchos IDE usados en el mundo, entre los mejores se encuentran los siguientes:

El IDE **Eclipse** es uno de los más usados, es multiplataforma y multilenguaje. Por muchos programadores es muy aclamado, sin embargo, hay que tomar en cuenta que en Eclipse es necesario agregar varios plugins para que funcione al cien por ciento (dependiendo de lo que se necesite hacer), si se comienza en el mundo de la programación se toma en cuenta. Eclipse dispone de un Editor de texto con un analizador sintáctico. La compilación es en tiempo real (Mojena & González, 2016).

El IDE **Netbeans** en su versión 8.1 es un entorno de desarrollo integrado libre, hecho principalmente para el lenguaje de programación Java. Existe además un número importante de módulos para extenderlo. NetBeans IDEes un producto libre y gratuito sin restricciones de uso. La plataforma NetBeans permite que las aplicaciones sean desarrolladas a partir de un conjunto de componentes de software llamados módulos. Un funcionalidad es un archivo Java que contiene clases de java escritas para interactuar con las APIs de NetBeans y un archivo especial (manifest file) que lo identifica como funcionalidad. Las aplicaciones construidas a partir de módulos pueden ser extendidas agregándole nuevos módulos. Debido a que las funcionalidades pueden ser desarrollados (Mojena & González, 2016).

Existen otros IDEs muy conocidos, pero el más adecuado para la realización de la aplicación según sus desarrolladores debido a su usabilidad y funcionalidades es:

Como lo que se desea es desarrollar para Windows y sus versiones anteriores utilizando tecnología de Microsoft, la mejor opción es **Visual Studio**. Dentro de este IDE se podrá desarrollar utilizando el entorno .net. También tiene la opción de convertirse en un desarrollador para la tienda de Windows. Visual Studio permite a los desarrolladores crear sitios y aplicaciones web, así como servicios web en cualquier entorno que soporte la plataforma .NET (a partir de la versión .NET 2002). Así se pueden crear aplicaciones que se comuniquen entre estaciones de trabajo, páginas web, dispositivos móviles, dispositivos embebidos, consolas, entre otras tecnologías (Mojena & González, 2016).

1.4.5 Sistema gestor de base de datos

Los Servidores de Bases de Datos. También conocidos como RDBM (Relational Data Base Management System, en español Servidor de Bases de Datos), son programas que permiten organizar datos en una o más tablas relacionadas. Los servidores de Bases de Datos se utilizan en todo el mundo en una amplia variedad de aplicaciones. Para bases de datos con múltiples usuarios sirve un servidor de base de datos. Las bases de datos están situadas en un servidor y se puede acceder a ellas desde terminales o equipos con un programa llamado cliente- que permita el acceso a la base o bases de datos (Mojena & González, 2016). Los gestores de base de datos de

este tipo permiten que varios usuarios hagan operaciones sobre ella al mismo tiempo. Dentro de los RDBM más conocidos se encuentran los siguientes:

El RDBM **PostgreSQL** en su versión 9.3.10, es un sistema de gestión de bases de datos relacional orientado a objetos y libre, publicado bajo la licencia PosgreSQL. Como muchos otros proyectos de código abierto, el desarrollo de PostgreSQL no es manejado por una empresa y/o persona, sino que es dirigido por una comunidad de desarrolladores que trabajan de forma desinteresada, libre y apoyada por organizaciones comerciales (Mojena & González, 2016).

El RDBM **MySQL** es una base de datos muy rápida en la lectura cuando utiliza el motor no transaccional MyISAM31, pero puede provocar problemas de integridad en entornos de alta concurrencia en la modificación. En aplicaciones web hay baja concurrencia en la modificación de datos y en cambio el entorno es intensivo en lectura de datos, lo que hace a MySQL ideal para este tipo de aplicaciones. Sea cual sea el entorno en el que va a utilizar MySQL, es importante monitorizar de antemano el rendimiento para detectar y corregir errores tanto de SQL32 como de programación.

Como la base de datos líder del mercado se encuentra **Oracle Database** ya que soporta todos los tipos de datos relacionales estándares, así como también datos nativos como XML, texto, imágenes, documentos, audio, y datos espaciales. El acceso a la información es realizado a través de interfaces estándares. Se considera a Oracle Database como uno de los sistemas de bases de datos más completos, destacando:

soporte de transacciones, estabilidad, escalabilidad, y soporte multiplataforma (Mojena & González, 2016).

A pesar del profesionalismo Oracle, no es la escogida para ejercer de Servidor de Bases de Datos, debido a que es un Servidor muy robusto destinado a proyectos de gran envergadura. El servidor seleccionado por la familiarización del equipo de desarrollo y por su sistema de gestión simplificada es:

MongoDB:

El NRDBM Mongo (Non Relational Data Base Management System, en español Servidor de Bases de Datos No relacionales) es una base de datos open-source desarrollada por 10gen en C++, fué lanzada en el año 2009 y pertenece a la categoría de las bases de datos documentales, esta base de datos surge como una nueva tendencia en el desarrollo de bases de datos y se refieren en general, a las bases de datos sin un esquema fijo. Además, suelen tener una seguridad de las transacciones a un nivel más bajo, pero son más rápidos en el acceso a los datos y escalan mejor que las bases de datos relacionales .La base de datos MongoDB consiste en un conjunto de bases de datos en la que cada base de datos contiene varias colecciones y cada colección puede contener diferentes tipos de objetos y cada objeto también llamado documento se representa como una estructura JSON que es una lista de pares de clave-valor (esto se debe a que MongoDB

trabaja con esquemas dinámicos). El valor puede ser de tres tipos: un valor primitivo, un array de documentos, o de nuevo una lista de pares de clave-valor o documentos

MongoDB es una base de datos NoSQL diseñada para facilitar el desarrollo y escalabilidad de aplicaciones. Algunas de las razones por las que se recomienda su uso son: .

- Ofrece una gran escalabilidad horizontal, lo que permite manejar grandes volúmenes de datos y tráfico.
- Es muy flexible y permite agregar campos y colecciones sin necesidad de modificar la estructura de la base de datos.
- Cuenta con una gran cantidad de recursos disponibles, como tutoriales y ejemplos, que facilitan su uso y aprendizaje.

En resumen, MongoDB es una base de datos NoSQL muy útil para el desarrollo de aplicaciones web escalables y flexibles. Es fácil de aprender y utilizar. Además, permite almacenar datos en formato JSON y manejar grandes volúmenes de datos y tráfico (Zhingri, 2016).

1.4.6 Servidor Web

Un servidor web o servidor HTTP es un programa informático que procesa una aplicación del lado del servidor, realizando conexiones bidireccionales y/o unidireccionales y síncronas o asíncronas con el cliente y generando o cediendo una respuesta en cualquier lenguaje o Aplicación del lado del cliente. El código recibido por el cliente suele ser compilado y ejecutado por un navegador web. Para la transmisión de todos estos datos suele utilizarse algún protocolo. Generalmente se usa el protocolo HTTP para estas comunicaciones, perteneciente a la capa de aplicación del modelo OSI. El término también se emplea para referirse al ordenador que ejecuta el programa multiplataforma (Mojena & González, 2016).

El servidor web que se usa para desplegar las aplicaciones convenientes es:

Apache en su versión 2.0 es un servidor de aplicaciones web con licencia de software libre creada por la Apache Software Foundation (ASF). La licencia Apache requiere la conservación del aviso de copyright, no requiere la redistribución del código fuente cuando se distribuyen versiones modificadas. Todo el software producido por la ASF o cualquiera de sus proyectos está desarrollado bajo los términos de esta licencia. Además, algunos proyectos que no pertenecen a la ASF también siguen la licencia Apache.

Herramienta de interfaz gráfica de usuario (GUI)

MongoDB llama la atención por su simple interfaz (en Compass) y su genial desempeño en consultas y operaciones de tipo INSERT, UPDATE y DELETE, que marcan una gran diferencia con el clásico de SQL. Finalmente, como pauta a la hora de elegir un SGBD para nuestra BD, hemos de tener en cuenta las necesidades y características de nuestra BD y, sobre todo, el uso que se le quiere dar. De esta forma podremos maximizar el rendimiento que obtendremos de nuestra base de

datos.(Jiménez, 2021).

Acunetix

Acunetix es una herramienta de seguridad de aplicaciones web automatizada. Es capaz de escanear cualquier sitio Web o aplicación Web que es accesible a través del protocolo HTTP / HTTPS (HiperText Transfer Protocol).Acunetix también proporciona herramientas de pruebas de penetración manuales que aumentan y contribuyen a las pruebas automatizadas, así como ayudar con la prueba de vulnerabilidades lógicas.

Acunetix es una herramienta de seguridad web que se utiliza para detectar vulnerabilidades en aplicaciones web. Algunas de las razones por las que se recomienda su uso son:

- Permite detectar vulnerabilidades en aplicaciones web de manera rápida y sencilla.
- Ofrece una gran cantidad de opciones de escaneo, lo que permite personalizar el proceso de detección de vulnerabilidades.
- Es compatible con una amplia variedad de tecnologías web, lo que garantiza que se pueden detectar vulnerabilidades en diferentes tipos de aplicaciones web.
- Cuenta con una gran cantidad de recursos disponibles, como tutoriales y ejemplos, que facilitan su uso y aprendizaje.

En resumen, Acunetix es una herramienta de seguridad web muy útil para detectar vulnerabilidades en aplicaciones web. Es fácil de aprender y utilizar. Además, permite personalizar el proceso de detección de vulnerabilidades y es compatible con una amplia variedad de tecnologías web (González, 2019).

Jmeter

La aplicación JMeter de Apache es un software de código abierto, una aplicación Java diseñada para cargar el comportamiento funcional y medir el rendimiento. Originalmente fue diseñado para probar aplicaciones web, pero se ha expandido a otras funciones de prueba. Apache JMeter puede emplearse para probar el rendimiento en recursos estáticos y dinámicos, aplicaciones dinámicas en la Web. Se puede utilizar para simular una carga pesada en un servidor, grupo de servidores, red u objeto para probar su resistencia o para analizar el rendimiento general bajo diferentes tipos de carga. Las características de Apache JMeter incluyen: Capacidad para cargar y realizar pruebas de rendimiento en diferentes tipos de aplicaciones / servidor / protocolo. Fácil correlación mediante la capacidad de extraer datos de formatos de respuesta más populares, HTML, JSON (JavaScript Object Notation), XML (Extensible Markup Languajes) o cualquier formato textual, portabilidad completa, almacenamiento en caché y análisis offline/ reproducción de los resultados de las pruebas, núcleo extensible

JMeter es una herramienta de prueba de carga y rendimiento que se utiliza para medir el rendimiento de aplicaciones web. Algunas de las razones por las que se recomienda su uso son:

- Permite simular el comportamiento de múltiples usuarios en una aplicación web, lo que

permite medir su rendimiento en situaciones de alta carga.

- Ofrece una gran cantidad de opciones de prueba, lo que permite personalizar el proceso de prueba de acuerdo a las necesidades del usuario.
- Es compatible con una amplia variedad de tecnologías web, lo que garantiza que se pueden realizar pruebas en diferentes tipos de aplicaciones web.
- Cuenta con una gran cantidad de recursos disponibles, como tutoriales y ejemplos, que facilitan su uso y aprendizaje.

En resumen, JMeter es una herramienta muy útil para medir el rendimiento de aplicaciones web en situaciones de alta carga. Además, permite personalizar el proceso de prueba y es compatible con una amplia variedad de tecnologías web (González, 2019).

1.5 Conclusiones del capítulo

La definición del marco teórico facilitó los elementos teóricos que sustentan la solución del problema. El análisis de los sistemas homólogos, permitió identificar las tendencias en cuanto al desarrollo de los sistemas de gestión de activos fijos tangibles. Establecer la metodología AUP-UCI en su escenario 2 como guía para el proceso de desarrollo y la base tecnológica a emplear en la solución que se propone permitió el análisis de herramientas y tecnologías existentes para el desarrollo de la solución propuesta.

CAPÍTULO 2: Análisis y diseño del Sistema para la Gestión de Activos Fijos Tangibles en la Emisora Radio Reloj.

2.1 Introducción

En el proceso de creación de un software es importante el entendimiento del negocio para implementar una solución adecuada. En el presente capítulo se describirá el sistema a desarrollar, a partir del estudio de los procesos actuales del negocio, la descripción y el modelado de los mismos, con el objetivo de comprender el funcionamiento y llevar a cabo una correcta implementación de dicho sistema. Además, se especifican los aspectos funcionales del sistema que se propone construir, así como los diferentes artefactos asociados a la metodología de desarrollo.

2.2 Descripción de la propuesta de solución

La necesidad de automatizar el proceso de gestión y manejo de los activos fijos tangibles (AFT) en la emisora Radio Reloj ha impulsado el desarrollo de una solución integral. Esta solución debe permitir la organización de los inventarios por departamentos, reemplazando la búsqueda manual en extensas hojas de cálculo por simples ordenamientos en la Base de Datos. Además, se requiere la implementación de un proceso de búsqueda rápida en la base de datos, que permita a los usuarios obtener información detallada de un AFT específico con solo ingresar un valor o condición de búsqueda. La gestión administrativa de los usuarios, incluyendo la asignación de responsabilidades y roles en la aplicación, es una prioridad para automatizar. Esto garantizará un acceso seguro al sistema y promoverá la seguridad, disponibilidad y confiabilidad de los datos almacenados. La solución propuesta también abarca la gestión de la información relacionada con los AFT, incluyendo Actas de Entrega de Medios, Modelos de Movimiento (como el Modelo 1-01) y Reportes Estadísticos. La aplicación web propuesta se basará en el uso de software libre y estará respaldada por una base de datos en Mongo DB. Esta aplicación permitirá la automatización de procesos específicos de los departamentos de la emisora Radio Reloj. Una vez implementada, la solución brindará a los usuarios la capacidad de acceder al sistema para consultar la información de manera eficiente. Además, se contempla la posibilidad de que el administrador agilice el trabajo mediante la importación de un documento previamente elaborado que contenga los datos de cada activo, ya sea utilizado o no, presente en la emisora.

2.2.1 Fuentes de obtención de requisitos

Las fuentes de obtención de requisitos, fuentes de información o fuentes de requisitos, son las vías a través de las cuales el equipo de desarrollo obtiene las necesidades del cliente que más tarde se convertirán en requisitos del sistema.

En un software típico, los requerimientos tiene muchas fuentes y es esencial que todas las fuentes potencial sean identificadas y evaluadas (Fuentes Castillo, 2023). Las fuentes de información durante la fase del descubrimiento de requerimientos incluyen la documentación, los stakeholders del sistema y la especificación de sistemas similares (Sommerville, 2016c). Definen en el SWEBOK las fuentes de obtención de requisitos como se muestran a continuación:

◆Metas (“preocupación de negocio” o “factor crítico del éxito”): se refiere a los objetivos generales de alto nivel del software. Proveen la motivación por el software, pero a menudo son formuladas vagamente. Los ingenieros de software necesitan prestar particular atención a determinar el valor (concerniente a prioridad) y costo de las metas (estudio de viabilidad).

◆Conocimiento del dominio: Los ingenieros de software necesitan adquirir o tener conocimiento disponible sobre el dominio de la aplicación. El conocimiento del dominio provee el trasfondo contra el cual se debe establecer todo el conocimiento de requisitos obtenido para comprenderlos. Es una práctica emular un enfoque ontológico en el conocimiento del dominio. La relación entre conceptos relevantes y el dominio de aplicación deben ser identificadas.

◆ Stakeholders: Personas o grupos que se verán afectados por el sistema, directa o indirectamente. Pueden ser usuarios finales, ingenieros que desarrollan o dan mantenimiento a otros sistemas relacionados, gerentes del negocio, expertos en el dominio, representantes de los trabajadores entre otros.

◆Reglas del negocio: Son sentencias que definen o restringen algunos aspectos de la estructura o del comportamiento del negocio en sí.

◆ Entorno operacional: Los requisitos serán derivados del entorno en el cual el software será ejecutado. Deben ser buscados activamente, porque pueden afectar la viabilidad y el coste del software. Además, restringen las opciones de diseño.

◆ Entorno de la organización: La construcción del software se realiza para dar soporte a determinados procesos de negocio. La selección puede estar condicionada por la estructura, cultura, y política interna de la organización. Generalmente el nuevo software no debe forzar cambios imprevistos en el proceso de negocio (Fuentes Castillo, 2023).

2.2.2 Técnicas de recopilación de información (TRI)

(Pressman & Maxim, 2019b) refieren que en la actividad de indagación de los requerimientos pueden aplicarse técnicas como entrevistas con los clientes, observación, encuestas, escenarios de uso y estudio de datos históricos (por ejemplo, reportes de problemas) como materia prima para la actividad de recabación. Sommerville (2016a) por su parte, plantea el uso de técnicas de entrevistas y observaciones, que pueden usarse combinados y junto con escenarios, casos de uso, etnografías y prototipos para ayudar a los participantes a entender cómo será el sistema.

CMMI (E. del P. CMMI, 2010) plantea que algunos ejemplos de técnicas para identificar las necesidades son: demostraciones de tecnología, grupos de trabajo de control de la interfaz, grupos de trabajo de control técnico, revisiones intermedias del proyecto, cuestionarios, entrevistas y escenarios (operacional, soporte y desarrollo) obtenidos de los usuarios finales, walthroughs de soporte, desarrollo y de operación, y análisis de tareas de usuario final, workshops con las partes interesadas para la educción de los atributos de calidad, prototipos y modelos, tormenta de ideas. Además, se listan despliegue de la función de calidad, estudios de mercado, pruebas beta, extracción de fuentes como documentos, estándares o especificaciones, observación de productos, entornos y patrones de flujo de trabajo existentes, casos de uso, historias de usuario, pequeñas entregas incrementales “por rodajas” de la funcionalidad del producto, análisis de casos de negocio, ingeniería inversa (para productos heredados) y encuestas de satisfacción del cliente.En el Swebok (Bourque & Fairley, 2014) se plantea que las principales técnicas existentes para la obtención de requisitos son: entrevistas, escenarios, prototipos, reuniones facilitadoras,observación, historias de usuario, técnicas de minería de datos entre otras. En la 2da edición de libro Fundamentos de la Ingeniería de Requisitos (Pohl & Rupp, 2015), guía de estudio de la Junta Internacional de Ingeniería de Requisitos (IREB) para el examen de la certificación profesional de Ingeniería de Requisitos nivel elementan, se definen y clasifican técnicas para la obtención de requisitos.

Técnicas de encuesta: Pretenden obtener conocimiento explícito. Se centran en obtener declaraciones tan precisas e imparciales como sea posible de los interesados con respecto a sus requisitos. Suponen que el encuestado es capaz de expresar explícitamente su conocimiento y que está comprometido a invertir tiempo y esfuerzo. Al ser impulsadas por el ingeniero de requisitos, podría suceder que las preocupaciones de los stakeholders se olviden, reemplacen o se ignoren.

Entrevista: Es una conversación dirigida, con un propósito específico y que usa un formato de preguntas y respuestas. Se establece así un diálogo asimétrico, donde una parte busca recoger información y la otra, actúa como fuente. Es la técnica más usual aplicada en la captura de requisitos. Garantiza el contacto directo con los interesados lo que ayuda a la validación inmediata a través de procesos de comunicación. Resulta vital preparar correctamente la entrevista. Asegura la completitud de las respuestas.Logra retroalimentación inmediata. Consume mucho tiempo.

Cuestionario: Método empírico que se utiliza para recopilar información de o sobre personas para describir, comparar o explicar su conocimiento, sus actitudes o su comportamiento. Si existiera un gran número de personas a ser encuestadas, un cuestionario online es una opción viable. Se aplica cuando los stakeholders no pueden expresar explícitamente su conocimiento (no están disponibles). Consume poco tiempo y costo, pero son difíciles de diseñar. Permite hacer afirmaciones relevantes con respecto a los requisitos. No se tiene retroalimentación de inmediato.

Técnicas centradas en documentos: Reutilizan soluciones y experiencias hechas con sistemas existentes. Cuando se reemplaza un sistema heredado, garantiza que se pueda identificar toda la funcionalidad del sistema heredado. Deben combinarse con otras técnicas para validar los requisitos obtenidos e identificar nuevos requisitos. Estudian estándares, leyes, regulaciones que rigen o se aplican en el negocio en cuestión.

Arqueología del sistema: Extrae la información a partir de la documentación o implementación (código) de un sistema heredado, o del sistema de un competidor. Es muy laboriosa, pero genera una gran cantidad de requisitos bien detallados. No se pierde ninguna funcionalidad del sistema heredado.

Lectura basada en perspectiva: Los documentos deben leerse teniendo en cuenta una perspectiva particular, por ejemplo, la perspectiva del implementador o del probador.

Reutilización: Permite reutilizar los requisitos que se han compilado previamente y se han actualizado siguiendo un determinado estándar de calidad.

Técnicas de observación: Es la adquisición activa de información a partir de los sentidos. El ingeniero de requisitos observa a las partes interesadas mientras realizan su trabajo. Se utiliza para profundizar en lo que realmente se está haciendo, viendo de primera mano la relación del usuario con la organización y su medio ambiente físico. Se usan cuando los especialistas de dominio no pueden pasar el tiempo necesario para compartir su experiencia con el ingeniero de requisitos, o no pueden expresar su conocimiento.

Observación de campo: El ingeniero de requisitos está en el lugar con el especialista o los usuarios del sistema y observa y documenta los procesos y procedimientos operativos que llevan a cabo. Es adecuada para procedimientos operativos que son difíciles de expresar verbalmente y que son visibles físicamente.

Aprendizaje: El ingeniero de requisitos debe aprender activamente y realizar los procedimientos de las partes interesadas. Cuestiona procedimientos operativos poco claros y complejos para que pueda obtener experiencia en el dominio. Es bueno que durante un tiempo cambien de rol Ingeniero de requisitos (alumno) y los interesados (maestros).

Técnicas de creatividad: Establecen innovaciones. Sirven para desarrollar requisitos innovadores, delineando una visión inicial del sistema y factores emocionantes. Usualmente no son adecuadas para establecer requisitos precisos sobre el comportamiento del sistema.

Tormenta de ideas: Reunión creativa para recopilar ideas (5 a 10 personas). Las ideas son documentadas por un moderador sin discutirlas, juzgarlas o comentarlas. Especialmente efectiva cuando las personas pertenecen a diferentes grupos de stakeholders.

Paradoja de tormenta de ideas: Se recopilan eventos que no deben ocurrir. Posteriormente, el grupo desarrolla medidas para evitar que ocurran los eventos recopilados anteriormente. Ayuda a identificar riesgos y contramedidas. Son buenas cuando hay un grupo grande personas, se recopilan muchas ideas en poco tiempo y las personas colaboran entre si. Son poco efectivas cuando las personas no tienen el mismo nivel de conocimiento y la dinámica de grupo no es buena.

Grabaciones de audio y video: Muy adecuadas para obtener requisitos cuando los stakeholders no siempre están disponibles. Apoya entrevistas y observaciones.

Modelado de secuencias de acción: Los casos de uso documentan la vista externa del sistema a desarrollar. La interacción entre usuarios y sistemas.

Prototipos: Ayudan a contrastar los requisitos identificados y a obtenerlos cuando no están muy claros de lo que quieren.

Para la presente investigación, la autora tendrá en cuenta todas las TRI antes mencionadas, siguiendo la clasificación dada por el IREB, dejando la elección de las más idóneas para usar, a los miembros de los equipos de proyectos según las características y necesidades de los mismos.

Necesidades del cliente:

- Identificación y registro de los activos fijos tangibles de la emisora.
- Control del estado y uso de los activos fijos tangibles.
- Generación de informes sobre los activos fijos tangibles para la toma de decisiones y la rendición de cuentas.

2.3 Modelado del dominio

Un modelo conceptual es una representación visual de las clases conceptuales u objetos del mundo real en un dominio de interés (LARMAN, 2017). Para lograr un mejor entendimiento de los procesos que requieren informatización, se realizó un modelo conceptual (ver Ilustración 1), con un total de seis (13) clases y ocho (16) relaciones, el cual representa y relaciona los conceptos más importantes dentro del contexto del negocio.

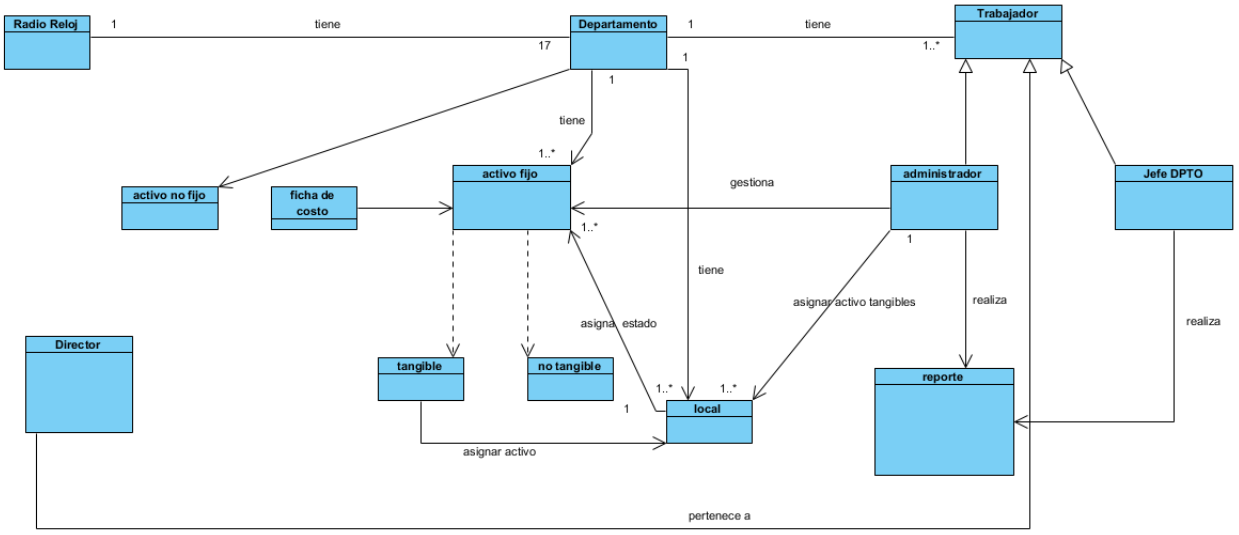


Ilustración 1 Modelo Conceptual (elaboración propia)

Descripción de las clases del modelo del conceptual

Ficha de costo: Es el documento donde se refleja la información relacionada con los componentes del costo unitario.

Local: Establecimiento donde se ubican los medios básicos.

Administrador: Es el encargado de gestionar los medios básicos y generaar los reportes.

Activos no fijos: Los activos fijos son un tipo de activos no corrientes e incluyen los bienes adquiridos por sus aspectos productivos, como edificios, vehículos, equipos.

Activos fijos: Son los medios básicos que pueden ser susceptibles de convertirse en dinero.

Activos fijos tangibles: Representan las propiedades físicamente tangibles con que cuentan los locales.

Activos fijos intangibles: Representan los medios conocidos como capital intelectual sobre los que no existe una regulación contable.

Radio Reloj: Establecimientocentral donde se ubican y se encuentran tanto los trabajadores, como los medios.

Director:Encargado del manejo de la emisora persona principal.

Trabajadores: Clase que representa todos los empleados de la emisora la cual divide entre el administrador y el resto de los empleados.

Jefe DPTO: Clase que representa a los jefes de cada departamento que no tienen los permisos requeridos en la emisora que solo interactuan con el sitio web y solo pueden tener la opción de generar reportes.

Reglas del Negocio que se deben considerar:

Las reglas del negocio que a continuación se muestran, constituyen políticas a cumplir o condiciones a satisfacer para regular algún aspecto del negocio necesario para tener un mayor control de los procesos a automatizar. Las reglas son los fundamentos definitorios de lo que el negocio sabe de sí mismo; es decir, son conocimiento básico de negocio.

- El administrador de la emisora es el único autorizado para realizar todos los movimientos que tienen lugar durante la gestión de Medios en cualquier departamento.
- Los Jefes de cada departamento solo podrán visualizar la existencia de cada activo y la única acción que podrán realizar es la de poder realizar reportes de estos.
- .El resto de los trabajadores solo podrán visualizar la existencia de los activos.

2.4 Requisitos de la propuesta de solución

Los requerimientos para un software son las descripciones de lo que el sistema debe hacer, el servicio que ofrece y las restricciones en su operación. Tales requerimientos reflejan las necesidades de los clientes por un sistema que atiende cierto propósito (Sommerville, 2011) .

2.4.1 Requisitos funcionales

Los requisitos funcionales son declaraciones de las funcionalidades que debe cumplir el sistema, de la manera en que éste debe reaccionar a entradas particulares y de cómo se debe comportar en situaciones particulares (PRESSMAN, 2002).

Para cumplir los objetivos propuestos, se necesitará que el sistema tenga las siguientes funcionalidades:

Tabla 2 Requisitos funcionales:

Requisitos funcionales	Nombre del requisito funcional	Descripción	Prioridad	Complejidad
RF 1	Iniciar sesión	Los usuarios registrados con anterioridad se pueden autenticar en el sistema, para ello deben llenar los siguientes campos: **Nombre de usuario:** Obligatorio campo de texto. Permite todos los caracteres. Longitud máxima 255 caracteres. **Contraseña:** Obligatorio. Campo de texto. Permite todos los caracteres. Longitud máxima 255 caracteres	alta	alta
RF 2	Cerrar sesión	El sistema permite al usuario autenticado cerrar la sesión del usuario. Para ello debe dirigirse al botón correspondiente en el menú. Si pasan 10 minutos de inactividad, o el usuario intenta autenticarse más de una vez,	baja	media

		el sistema cierra la sesión automáticamente y en el caso de la autenticación múltiple autentica la nueva sesión.		
RF 3	Registrar usuario.	Los usuarios no registrados se pueden registrar en el sistema, para ello deben llenar los siguientes campos: • **Nombre(s):** Obligatorio. Campo de texto. Longitud máxima 60 caracteres. Permite solo letras y espacio. Debe comenzar con letra. • **Apellidos:** Obligatorio. Campo de texto. Longitud mínima 5, máxima 60 caracteres. Permite caracteres alfabéticos, espacio, guión y comilla simple. Debe comenzar con letra. • **Dirección de correo electrónico:** Obligatorio. Campo de texto. Longitud máxima 255 caracteres. Permite direcciones de correo estructuralmente válidas (Estructura: usuario@subdominios.dominio). • **Contraseña:** Obligatorio. Campo de texto. Longitud mínima 6, máxima 128 caracteres. Permite cualquier caracterer.	media	baja
RF 4	Modificar usuario.	Los usuarios con los permisos de administración correspondientes, podrán modificar las cuentas de usuarios existentes en el sistema, editando todos sus campos. • **Nombre(s):** Obligatorio. Campo de texto. Longitud máxima 60 caracteres. Permite solo letras y espacio. Debe comenzar con letra. • **Apellidos:** Obligatorio. Campo de texto. Longitud mínima 5, máxima 60 caracteres. Permite caracteres alfabéticos, espacio, guión y comilla simple. Debe comenzar con letra. • **Nombre de usuario:** Obligatorio. Campo de texto. Longitud mínima 5, máxima 60 caracteres. Permite caracteres alfanuméricos, punto y guión bajo. Debe comenzar con letra. • **Dirección de correo electrónico:** Obligatorio. Campo de texto. Longitud máxima 255 caracteres. Permite direcciones de correo estructuralmente válidas (Estructura: usuario@subdominios.dominio). • **Contraseña:** Obligatorio. Campo de texto. Longitud mínima 6, máxima 128 caracteres. Permite cualquier caracterer. Si se edita este campo, una vez enviado el formulario, el sistema manda automáticamente un correo al usuario correspondiente, notificando el cambio de contraseña. • **Confirmar contraseña:** Obligatorio. Debe ser la misma contraseña escrita en el campo Contraseña.	media	baja

		• **Rol:** Obligatorio. Campo de selección múltiple. Contiene los roles insertados en el sistema. Valor por defecto invitado		
RF 5	Eliminar usuario.	El sistema muestra una pestaña la cual muestra la opción de aceptar o cancelar la opción de eliminar el usuario	baja	baja
RF 6	Mostrar Usuario	El sistema permite ver los datos de un usuario, dando clic en el nombre del usuario deseado.	baja	baja
RF 7	Listar cuentas de usuarios.	El sistema permite ver el listado de los usuarios registrados, accediendo a la pestaña Usuarios, de estos se muestran el nombre de usuario, el rol. Además, las opciones de modificar y eliminar.	alta	media
RF 8	Registrar rol	El sistema permite a los usuarios con rol de administración, registrar roles en el sistema. Para ello lleven llenar los siguientes datos: • Rol: Obligatorio. Campo de texto. Admite solo letras y espacio. Longitud máxima 255 caracteres. Permisos: Obligatorio. Lista de selección múltiple. Contiene un listado con todos los permisos de las acciones que se pueden realizar en el sistema	alta	media
RF 9	Editar rol	• El sistema permite a los usuarios con rol de administración, registrar roles en el sistema. Para ello lleven llenar los siguientes datos: • Rol: Obligatorio. Campo de texto. Admite solo letras y espacio. Longitud máxima 255 caracteres. • Permisos: Obligatorio. Lista de selección múltiple. Contiene un listado con todos los permisos de las acciones que se pueden realizar en el sistema	media	baja
RF 10	Listar roles	Muestra la lista de roles existentes	alta	media
RF 11	Eliminar roles	Permite eliminar uno o más roles de los existentes en el sistema.	alta	media
RF 12	Cambiar asignación de permisos individuales	El sistema permite a los usuarios con rol de administración, asignar o retirar permisos individuales a los usuarios. Para ello el administrador accede a la pantalla correspondiente y selecciona o desmarca los permisos según desee.	alta	media
RF 13	Listar permisos	Muestra la lista de permisos existentes.	alta	media
RF 14	Insertar activo.	Los usuarios autenticados con privilegios de administración pueden crear un activo. Para ello se debe llenar los siguientes campos: • **Código:** Obligatorio. Campo de texto. Permite solo caracteres numéricos. Longitud máxima 10 caracteres. • **Descripción:** Obligatorio. Campo de texto. Permite todos los caracteres.	alta	alta

		Longitud máxima 255 caracteres. • **Cantidad:** Obligatorio. Permite solo los caracteres numéricos. . Longitud máxima 1. • **Precio:** Obligatorio. Permite los caracteres numéricos • **Observación:** Obligatorio. Campo de texto. Permite todos los caracteres. Longitud máxima 255. • **Áreas:** Obligatorio. Lista de selección. Muestra un listado con todas las áreas de le emisora. . Modelos: Opcional. Lista de selección múltiple. Contiene un listado de los modelos existentes en el sistema.		
RF 15	Modificar activo	Los usuarios autenticados con privilegios de administración pueden modificar un activo. Para ello se debe llenar los siguientes campos: • **Código:** Obligatorio. Campo de texto. Permite solo caracteres numéricos. Longitud máxima 10 caracteres. • **Descripción:** Obligatorio. Campo de texto. Permite todos los caracteres. Longitud máxima 255 caracteres. • **Cantidad:** Obligatorio. Permite solo los caracteres numéricos. . Longitud máxima 1. • **Precio:** Obligatorio. Permite los caracteres numéricos • **Observación:** Obligatorio. Campo de texto. Permite todos los caracteres. Longitud máxima 255. • **Áreas:** Obligatorio. Lista de selección. Muestra un listado con todas las áreas de le emisora. • **Seleccionar Estado:** Obligatorio. Campo de texto. Permite todos los caracteres. Longitud máxima 255. . Modelos: Opcional. Lista de selección múltiple. Contiene un listado de los modelos existentes en el sistema.	alta	alta
RF 16	Eliminar activo	El sistema permite eliminar un activo de la base de datos. Para ello los usuarios con los permisos correspondientes, se dirigen a la pestaña Activos por Áreas del menú principal y en el listado que se muestra seleccionan la opción Eliminar de la Área deseada el activo a eliminar.	baja	baja
RF 17	Mostrar activo	El sistema permite visualizar la información de un activo. Para ello los usuarios con los permisos correspondientes, acceden al	alta	alta

		módulo Activos por Áreas del menú principal.		
RF 18	Buscar activo por código	El sistema permite buscar determinado activo registrado, accediendo a la pestaña de buscar. Para ello se deben llenar alguno de los siguientes campos: • **Código:** Opcional. Campo de texto. Permite solo caracteres numéricos. Longitud máxima 10 caracteres.	alta	alta
RF 19	Buscar activo por descripción	El sistema permite buscar determinado activo registrado, accediendo a la pestaña de buscar. Para ello se deben llenar alguno de los siguientes campos: • **Descripción:** Opcional. Campo de texto. Permite todos los caracteres. Longitud máxima 255 caracteres.	alta	alta
RF20	Cambiar Estado del Activo	El sistema permite buscar determinado activo registrado, accediendo a la pestaña Activos por Áreas. Para ello se deben llenar alguno de los siguientes campos: • **Seleccionar Estado:** Obligatorio. Campo de texto. Permite todos los caracteres. Longitud máxima 255.	baja	baja
RF21	Mover Activo	El sistema permite buscar determinado activo registrado, accediendo al módulo Activos por Áreas. Para ello se deben llenar alguno de los siguientes campos: • **Áreas:** Obligatorio. Lista de selección. Muestra un listado con todas las áreas de le emisora.	media	baja
RF22	Notificar Modificación de Activo	Los usuarios autenticados con privilegios de administración pueden modificar un activo. Para ello se debe llenar los siguientes campos: • **Código:** Obligatorio. Campo de texto. Permite solo caracteres numéricos. Longitud máxima 10 caracteres. • **Descripción:** Obligatorio. Campo de texto. Permite todos los caracteres. Longitud máxima 255 caracteres. • **Cantidad:** Obligatorio. Permite solo los caracteres numéricos. . Longitud máxima 1. • **Precio:** Obligatorio. Permite los caracteres numéricos • **Observación:** Obligatorio. Campo de texto. Permite todos los caracteres. Longitud máxima 255. • **Áreas:** Obligatorio. Lista de	alta	alta

		selección. Muestra un listado con todas las áreas de le emisora. • **Seleccionar Estado:** Obligatorio. Campo de texto. Permite todos los caracteres. Longitud máxima 255. . Luego se muestra una pestaña que notifica la modificación satisfactoria del activo.		
RF 23	Mostar locales	El sistema permite visualizar la información de los activos. Para ello todos los usuarios correspondientes, acceden al menú principal. Donde se muestra todos los locales y la cantidad de activos almacenados en cada área.	alta	alta
RF 24	Notificar Eliminación de Activo	El sistema permite buscar determinado activo registrado, accediendo a la pestaña Activos por Áreas. Para ello se deben llenar alguno de los siguientes campos: **Áreas:** Obligatorio. Lista de selección. Muestra un listado con todas las áreas de le emisora, Luego dar clic en el botón de eliminar activo y se muestra una pestaña que notifica la eliminación satisfactoria del activo.	alta	media
RF 25	Actualizar lista de activos	Luego de realizar cualquier acción de la lista de los activos se vuelve a seleccionar dicha área El sistema permite ver el listado de los activos ya actualizados registrados, accediendo a la pestaña Activos por Áreas. De estos se muestran el nombre de activo, el estado, la descripción. Además, las opciones de modificar y eliminar.	baja	baja
RF26	Mostrar lista de activos	El sistema permite ver el listado de los activos registrados, accediendo a la pestaña Activos por Áreas. De estos se muestran el nombre de activo, el estado, la descripción. Además, las opciones de modificar y eliminar.	alta	media
RF27	Mostrar Reporte de activos por Departamentos	El sistema permite ver el listado de los activos registrados, accediendo a la pestaña Activos por Áreas. De estos se muestran el nombre de activo, el estado, la descripción.	media	baja
RF28	Imprimir Reporte por Departamentos	El sistema permite ver el listado de los activos registrados, accediendo a la pestaña Activos por Áreas. De estos se muestran el nombre de activo, el estado, la descripción. Además muestra la opción de imprimir al dar clic.	baja	baja

2.4.2 Requisitos no funcionales

Los requisitos no funcionales representan características generales y restricciones de la aplicación o sistema que se esté desarrollando. Son aquellos que no se refieren directamente a las funciones específicas que proporciona el sistema, sino a las propiedades emergentes de éste como la fiabilidad, el tiempo de respuesta y la capacidad de almacenamiento. Suelen expresarse de una manera general y sin hacer referencia a algún modulo, transacción o característica del sistema (Hernández, 2019).

RnF Usabilidad

RnF1.1: El sistema para la gestión de activos fijos tangibles de Radio Reloj debe ser una aplicación web.

RnF 1.2: La aplicación debe presentar una interfaz agradable e intuitiva, lo que significa que los usuarios puedan comprender y utilizar la aplicación web sin necesidad de instrucciones complicadas además de un diseño limpio y organizado.

RnF Requisitos de eficiencia

RnF2.1: El sistema debe permitir que los usuarios interactúen con él de manera simultánea, 50 usuarios a lo máximo.

RnF2.2: El tiempo de demora de una petición al servidor debe ser menor de cinco (5) segundos aproximadamente.

RnF Requisitos de soporte

RnF3.1: El sistema debe contar con toda la documentación definida en el expediente de proyecto asociada a su proceso de desarrollo para las actividades de soporte.

RnF Requisitos de software

RnF4.1: Para el uso del sistema se requiere un dispositivo cliente tener instalados navegador web, dígase Windows, Linux y GNU/Linux Nova que se pueda instalar navegadores web como Chrome, Firefox, Safari, Opera para el uso de la aplicación.

RnF4.2: Servidor web NPM (Node Package Manager)

RnF4.3: Servidor de base de datos MongoDB en su versión 6.0 o superior.

RnF Requisitos de hardware

RnF5.1: El servidor de base de datos debe poseer una capacidad mínima de 20 GB.

RnF5.2: El servidor de aplicaciones web debe poseer una capacidad mínima de 80 GB.

RnF5.3: Los servidores web y de base de datos deben poseer como mínimo 1 GB de memoria

RAM.

RnF Requisitos de Seguridad

RnF6.1: En caso de que el sistema presente alguna falla, los errores deben mostrar la menor cantidad de detalles posible, de forma tal, que se evite dar información que comprometa la seguridad e integridad del sistema. Sólo se mostrarán detalles ampliados del error a usuarios con privilegios de administración.

RnF6.2: Se garantizará la integridad de la información mediante mecanismos de control de acceso utilizando usuarios, contraseñas y niveles de accesos para cada usuario, de manera que cada uno pueda tener disponible solamente las opciones que se encuentran en correspondencia con su actividad.

RnF6.3: La comunicación entre la PC cliente y el servidor de aplicaciones web se realiza a través del protocolo HTTPS.

RnF Requisitos de Portabilidad

RnF7.1: El sistema debe tener visibilidad en los principales navegadores web como Chrome,Firefox ,Safari ,Opera.

RnF7.2: El sistema podrá ser visualizado en dispositivos desde las resoluciones 320x480, 768x1024, 1024x980 y 1325x980.

2.5 Diagrama de caso de uso del sistema:

Un caso de uso captura un contrato que describe el comportamiento del sistema en diferentes condiciones mientras este responde a la petición de uno de sus usuarios.

Ilustración 2 Diagrana Caso Uso del Sistema

2.6 Descripción textual caso de uso Gestionar Activos

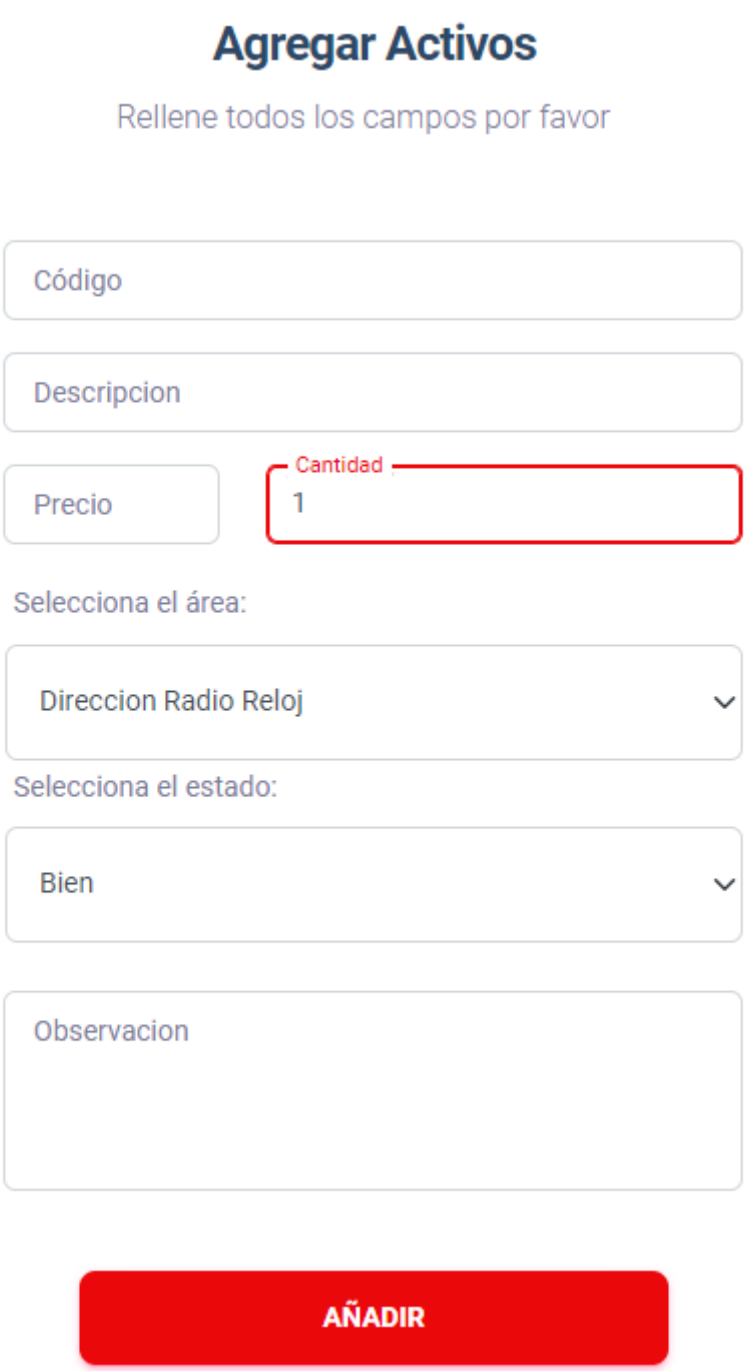
Una descripción textual de un caso de uso es una narración que describe el rol desempeñado por los actores en su interacción con el sistema. La descripción textual de un caso de uso es una especificación simple y consistente de cómo interactúan los actores y los casos de uso, y se utiliza para modelar el comportamiento del sistema y para identificar los requisitos funcionales del sistema. En general, un caso de uso consta de un actor (el usuario), una descripción del flujo de eventos y un resultado esperado(Gutierrez, 2011).

En la siguiente tabla se realiza la descripcion del caso de uso gestionar activos ya que es de todos el más importante ya que es el grueso y la función más importante y vital del sistema, es en el que se ve reflejado todo el poceso que se realiza con los activos fijos tangibles.

Tabla 3 Descripción textual Caso Uso Gestionar Activos:

Objetivo		Permitir registrar, modificar o eliminar datos acerca de un activo fijo tangible.
Actores		Administrador
Resumen		El proceso se inicia cuando el administrador decide registrar, modificar o eliminar datos acerca de un activo fijo tangible.
Complejidad		Alta.
Prioridad		Alto.
Precondiciones		Administrador ya autenticado.
Post condiciones		Se mostró, actualizó o se creó usuario(s).
Flujo de eventos		
Flujo básico "Gestionar activo"		
	Acción del actor	**Sistema**
1.	Selecciona de la página principal la opción "Gestionar activo".	
2.		Muestra una pantalla con un listado de activos y permite Registrar, Modificar o Eliminar activo(s).
3.	Desea registrar, modificar o eliminar activo(s).	
4.		Da la posibilidad de realizar alguna de las siguientes acciones: **1.** Si decide registrar un activo, ir a la sección "Insertar activo". **2.** Si decide modificar los datos de un activo, ir a la sección "Modificar activo". **3.** Si decide eliminar uno(s) activo(s), Muestra un mensaje.
Flujos Alternos		
2a. Listado de activos vacío		
		Sistema

Acción del actor		
1.		Carga una pantalla en blanco sin activos registrados.
3a. Desea realizar una búsqueda.		
Acción del actor		**Sistema**
1.		Ver el CU Buscar activo.
Sección 1: “Registrar Activo”		
Flujo básico Registrar Activo		
Acción del actor		**Sistema**
1.	Presiona el botón “Agregar”.	
2.		Muestra una ventana con los siguientes campos a introducir: • Descripción • Estado • Área • Código • Precio • Cantidad • Observación Y los botones “Aceptar” y “Cancelar.
3		Verifica que todos los campos estén llenos.
4		Verifica que los datos introducidos estén correctos.
5		Verifica que el activo no exista.
6		Registra el activo y se emite un mensaje “Registro satisfactorio”. *Finalizando así el caso de uso.*
7	Introduce los datos y presiona el botón “Aceptar”.	
Prototipo elemental de interfaz gráfica		

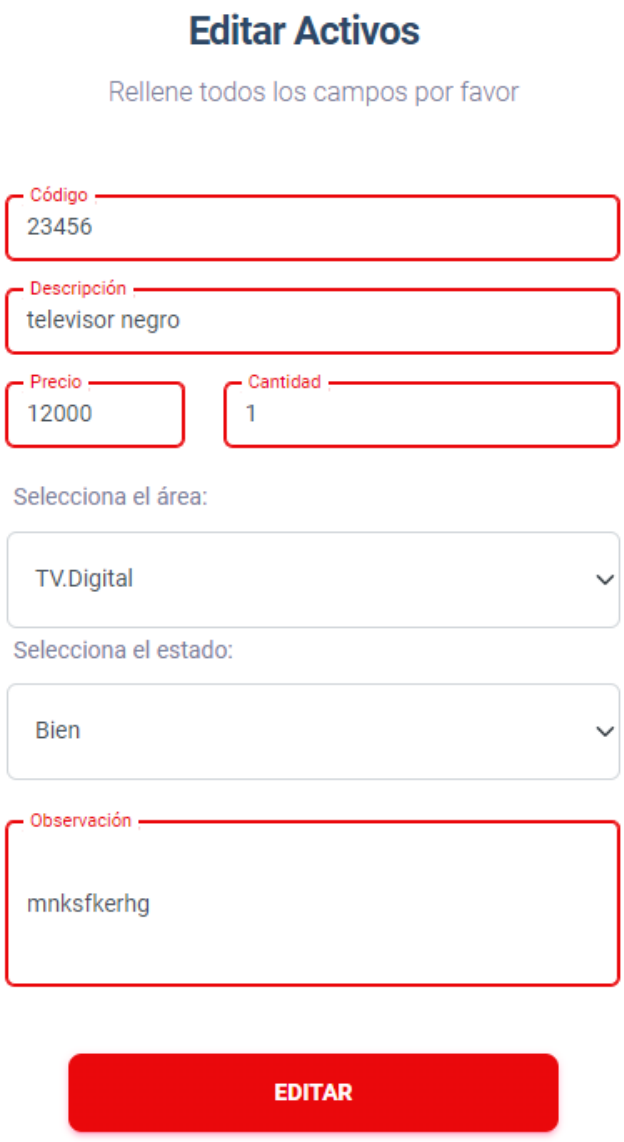


Flujos Alternos		
4a. Campos vacíos		
Acción del actor		**Sistema**
1.		Muestra el mensaje “Campos vacíos”.
5a. Datos incorrectos		

Acción del actor		Sistema
1.		Muestra el mensaje “Datos incorrectos”.
6a. Activo existente		
Acción del actor		**Sistema**
1.		Muestra el mensaje “Activo existente”.
7a.Selecciona “Cancelar”		
Acción del actor		**Sistema**
1.	Selecciona la opción “Cancelar”	
2.		Cierra la ventana de Agregar activo y vuelve a la ventana Gestionar activo.
Sección 2: “Modificar activo”		
Flujo básico Modificar activo		
Acción del actor		**Sistema**
1.	Marca uno de los activos mostrado y presiona el botón “Modificar”.	
2.		Muestra una ventana con los siguientes campos a introducir: • Descripción • Estado • Área • Código • Precio • Cantidad • Observación Y los botones “Aceptar” y “Cancelar.
3.	Realiza las actualizaciones deseadas y presiona el botón “Aceptar”.	

4.		Verifica que todos los campos estén llenos.
5.		Verifica que los datos introducidos estén correctos.
6.		Actualiza la información incorporada al activo y se emite un mensaje “Modificación satisfactoria”. *Finalizando así el caso de uso.*

Prototipo elemental de interfaz gráfica



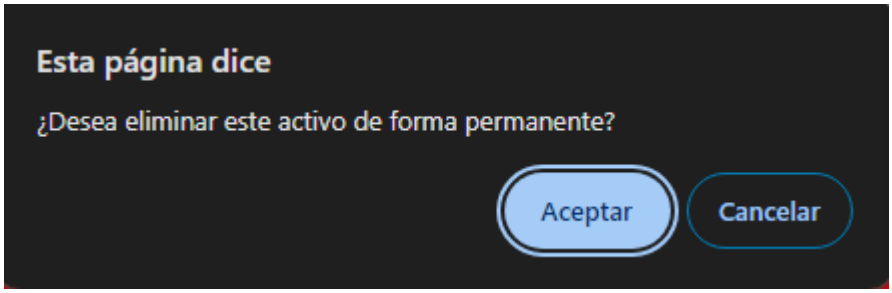
Flujos Alternos

4a. Campos vacíos

Acción del actor		Sistema
1.		Muestra el mensaje “Campos vacíos”.

5a. Datos incorrectos		
Acción del actor		**Sistema**
1.		Muestra el mensaje “Datos incorrectos”.
7a.Selecciona “Cancelar”		
Acción del actor		**Sistema**
1.	Selecciona la opción “Cancelar”	
2.		Cierra la ventana de Agregar activo y vuelve a la ventana Gestionar activo.

Sección 3: “Eliminar activo”		
Flujo básico Eliminar activo		
Acción del actor		**Sistema**
1.	Marca un(os) activo(s) mostrado(s) y presiona la opción “Eliminar”.	
		Muestra el mensaje de confirmación “¿Está seguro que desea eliminar el(los) elemento(s) seleccionado(s)?”. Y los botones “Aceptar” y “Cancelar”.
2.	Presiona el botón “Aceptar”.	
		Elimina el(los) activo(s) seleccionado(s) y emite el mensaje “Eliminación satisfactoria”. *Finalizando así el caso de uso.*
Prototipo elemental de interfaz gráfica		



Flujo Alterno		
2a. Cancelar eliminación de activo		
Acción del actor		**Sistema**

1.	Presiona el botón “Cancelar”.	
2.		Vuelve al paso 2 del flujo básico “Gestionar activo”.

2.7 Arquitectura del software

Descripción de la arquitectura de software y los patrones de diseño:

Arquitectura de software:

La arquitectura de software se refiere a la estructura y organización de un sistema de software, incluyendo la relación entre sus diferentes componentes y sus propiedades visibles externas. La arquitectura de software es una planificación fundamentada en patrones, modelos y abstracciones teóricas que se realiza antes de la implementación del software. La arquitectura de software es esencial para el desarrollo de sistemas de software eficientes y mantenibles, ya que proporciona una estructura y organización para los componentes del sistema, establece principios orientadores y promueve la reutilización. La arquitectura de software se utiliza para planificar con detalle el funcionamiento de una aplicación antes de iniciar con el proceso de desarrollo, establecer tiempos, así como determinar los recursos económicos y humanos que se necesitarán durante todo el proceso(Castro, 2015). La construcción de aplicaciones web mediante el uso de framework constituye una buena práctica, debido a que reducen el tiempo de desarrollo y hace mantenible el código. Dentro de la amplia gama de framework web se utiliza como una solución recurrente el patrón de diseño Modelo-Vista-Controlador (MVC). Express JS no es la excepción y aunque se plantee que solo garantiza los elementos del controlador y la vista, se puede integrar fácilmente un ORM para la gestión de esta capa. El patrón de diseño MVC organiza el código en base a su función. Este patrón separa el código en tres capas:

El modelo: representa la información relacionada con el dominio de la aplicación, es decir, su lógica del negocio. Abstrae la lógica relacionada con los datos, haciendo que la vista y las acciones sean independientes del tipo de gestor de base de datos utilizado. Para el componente una de las tareas más comunes es la persistencia y lectura de la base de datos. En la solución se utiliza Mongoose, una biblioteca cuyo objetivo es brindar poderosas Herramientas para el acceso a una base de datos no relacional. El ORM de Mongoose permite asociar objetos a una base de datos tal como Mongo DB. El componente gracias a las funcionalidades de Mongoose, puede persistir y recuperar objetos completos de la base de datos. Esto funciona asociando la clase TbActivoFijo.js a la tabla tb_activo_fijo de la base de datos y las propiedades de esa clase Java Script a las columnas de la tabla. Además del uso de un repositorio ActivosFijosRepositorio.js asociado a la entidad que contiene las consultas a la base de datos.

La vista: En el contexto de Node.js y Express.js, la vista se encarga de transformar el modelo en interfaces de usuario permitiendo la interacción con el sistema, se pueden utilizar otros motores de

plantillas para Node.js, como Pug o EJS. Estos motores de plantillas permiten definir la estructura de las vistas y renderizarlas con los datos del modelo. En cuanto al componente, se puede utilizar la herencia a tres niveles para reutilizar el máximo código posible, como en el caso de una plantilla base de la aplicación, una plantilla para la administración del componente y plantillas específicas para cada parte del componente. se ha diseñado para que las plantillas sean concisas y muy fáciles de leer y de escribir. En el componente se utiliza la herencia a tres niveles para reutilizar el máximo código posible. En el primer nivel se encuentra una sola plantilla base de la aplicación llamada base.html, en el segundo nivel una plantilla para la administración del componente llamada layout.html y en el tercer nivel, plantillas específicas para cada parte del componente por ejemplo index.html para listar los usuarios.

El controlador: se encarga de procesar las peticiones realizadas desde el navegador realizando los cambios necesarios en el modelo o en la vista. Este se encarga de aislar al modelo y a la vista de los detalles del protocolo utilizado para las peticiones (HTTP, consola de comando, etc.). En el componente los controladores son funciones js que toman información de la petición HTTP y construyen una respuesta HTTP. Los controladores tienen la lógica que el componente necesita para reproducir el contenido de las páginas. La clase controladora TbActivoFijoController.js tiene varias acciones, por ejemplo, indexAction que se encarga de mostrar todos los activos fijos de la base de datos (Mojena & González, 2016).

En la siguiente figura se ejemplifica de manera resumida el funcionamiento de la arquitectura MVC:

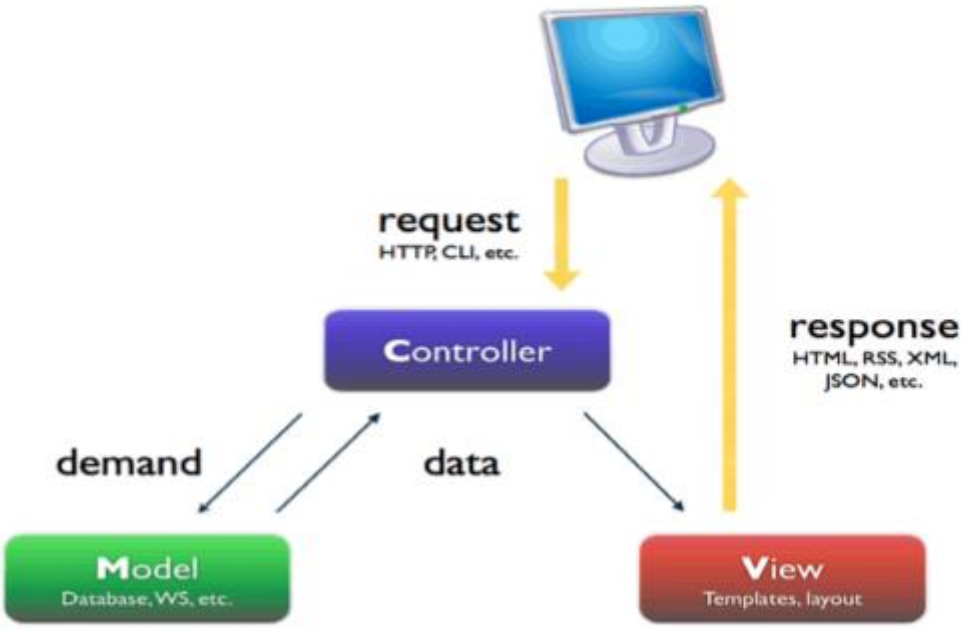


Ilustración 3 Arquitectura MVC

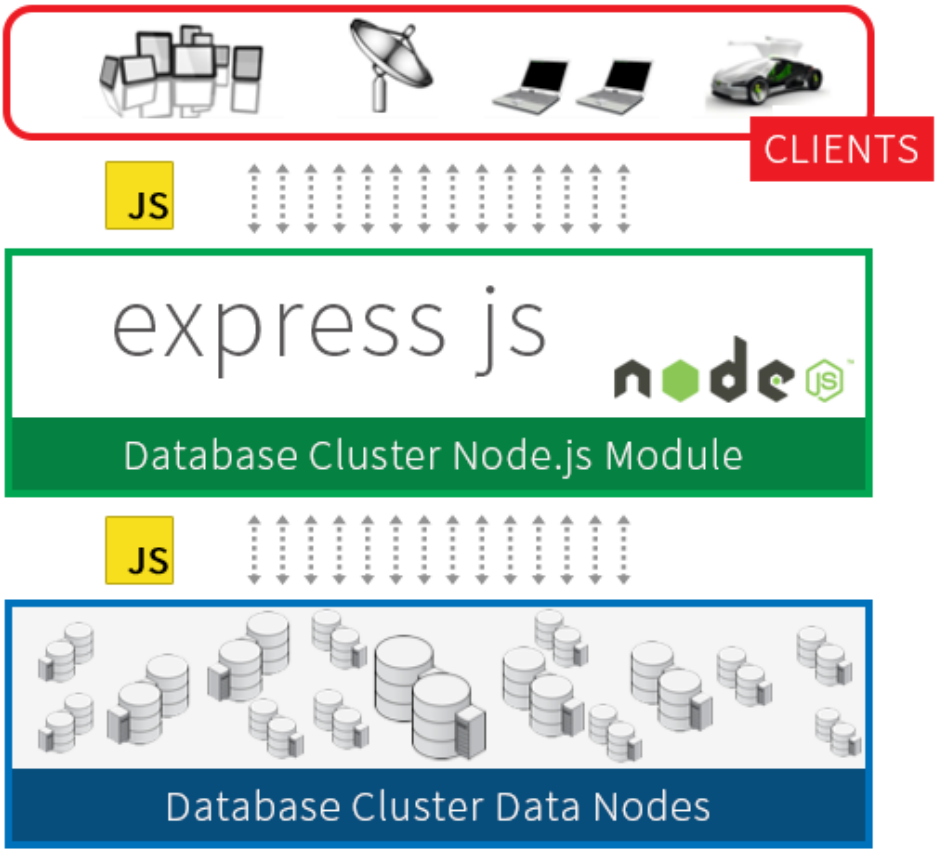


Ilustración 4 Arquitectura MVC enfocada a Express js

Cuando el usuario solicita ver una página de la herramienta, internamente sucede lo siguiente:

- El sistema de enrutamiento determina qué controlador está asociado con la página solicitada.
- Express js ejecuta el controlador asociado. Un controlador no es más que una clase de Java Script en la que puedes ejecutar cualquier código que quieras.
- El controlador solicita al modelo los datos necesarios para mostrar. El modelo no es más que una clase especializada en obtener información, normalmente de una base de datos.
- Con los datos devueltos por el modelo, el controlador solicita a la vista que cree una página mediante una plantilla y que inserte los datos del modelo. El controlador entrega al servidor la página creada por la vista.

2.8 Diagrama Entidad Relación (DER)

A continuación, se representa el DER, el cual representa el modelo físico de la base de datos del sistema. Este artefacto fue generado por la herramienta CASE Visual Paradingm for UML elegida por el equipo de desarrollo.

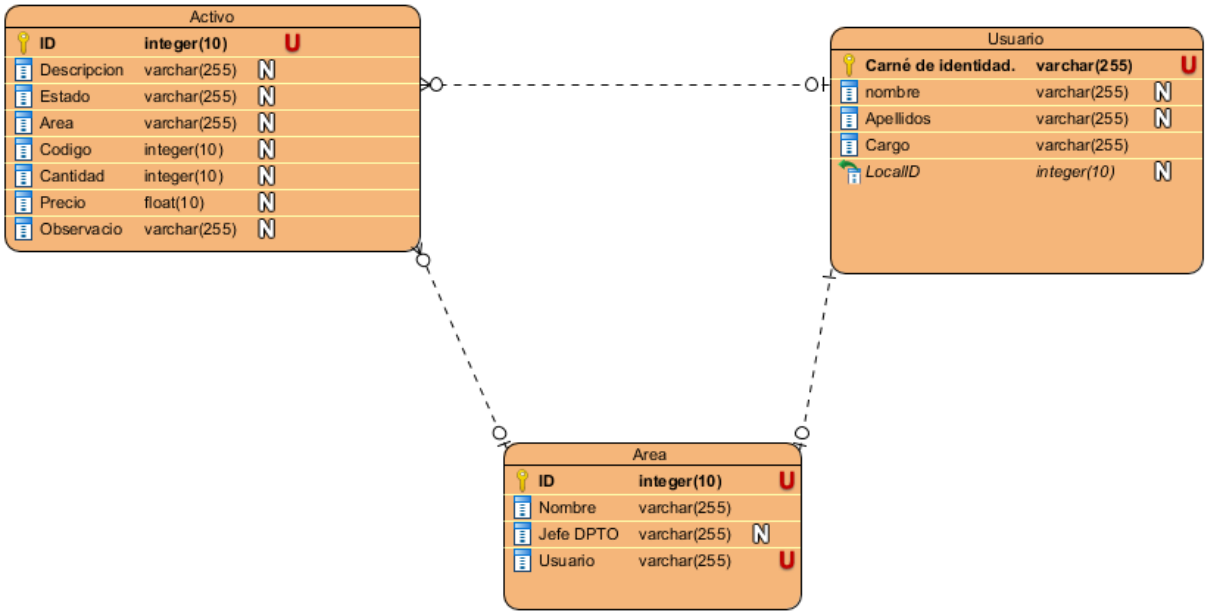


Ilustración 5 Diagrama Entidad Relación

Cada tabla de la base de datos tiene un objetivo específico que la aplicación utiliza para hacer persistir la información que gestiona. A continuación, se describe cada una de ellas:

tb_area: La tabla destinada a almacenar la información de las áreas que conforman las direcciones.
tb_tipo_activo_fijo: La tabla destinada a almacenar la información de los tipos de activos fijos tangibles.
tb_usuario: La tabla destinada a almacenar la información de los usuarios.

2.9 Patrones de diseño

Los patrones de diseño expresan esquemas para definir estructuras de diseño (o sus relaciones) con las que construir sistemas de software. Estos permiten diseñar sistemas seguros y que a su vez cumplan con los estándares de diseño establecidos por normas internacionales para el desarrollo de aplicaciones web (Buytaert, 2016).

Patrones Gang of Four (GOF)

El catálogo de patrones más famoso es el contenido en el libro “Design Patterns: Elements of Reusable Object-Oriented Software”, también conocido como: El libro GOF (Gang-Of-Four Book).

Según este documento (Gamma, y otros, 1997), estos patrones se clasifican por su propósito en creacionales, estructurales y de composición, mientras que respecto a su ámbito se clasifican en clases y objetos.

Instancia única:

Este patrón está diseñado para restringir la creación de objetos pertenecientes a una clase o el valor de un objeto único. Su intención consiste en garantizar que una clase sólo tenga una instancia y proporcionar un punto de acceso global a ella. Un ejemplo que evidencia este patrón en el Sistema de gestión de Activos es el proceso de creación de los propios activos , al cual se le asigna un identificador propio que evita la existencia en la base datos de elementos repetidos constituyendo así una instancia única.

Decorador:

Este patrón de diseño permite añadir responsabilidades extra a objetos concretos de manera dinámica. Proporciona una alternativa flexible para extender funcionalidades. Brinda la flexibilidad de que nuevos módulos puedan modificar el comportamiento del núcleo en cuanto al tratamiento de los datos y en cada uno de los eventos del sistema (Buytaert, 2016). Se encontrará evidenciado en el sistema en el módulo que permite la gestión de Activos Fijos .

Cadena de responsabilidades (Chain of Responsibility):

En cada solicitud de la página, el menú del sistema determina si hay un módulo para gestionar la solicitud y si el usuario tiene acceso a los recursos solicitados. Para ello, el mensaje se pasa a la opción del menú correspondiente a la vía de la solicitud. Si el elemento de menú no puede manejar la petición, se pasa a otro. Esto continúa hasta que un módulo se encarga de la petición, un módulo niega el acceso para el usuario, o la cadena se ha agotado.

Patrones GRASP (object-oriented design General Responsibility Assignment Software Patterns)

En diseño orientado a objetos, GRASP son patrones generales de software para asignación de responsabilidades, es el acrónimo de "GRASP (object-oriented design General Responsibility Assignment Software Patterns)". Aunque se considera que más que patrones propiamente dichos, son una serie de "buenas prácticas" de aplicación recomendable en el diseño de software.

Patrones GRASP presentes en este trabajo de diploma son los siguientes:

Controlador:

```
const mongoose = require("mongoose");
const app = require("./app");
const config = require("./config")

mongoose.connect(config.db, (err, res) => {
  if (err) throw err;
  console.log("conexion exitosa");

  app.listen(config.port, () => {
    console.log("API REST corriendooooooooo");
  });
});
```

Ilustración 6 Patrón GRASP Controlador

El patrón controlador es un patrón que sirve como intermediario entre una determinada interfaz y el algoritmo que la implementa, de tal forma que es la que recibe los datos del usuario y la que los envía a las distintas clases según el método llamado. Este patrón sugiere que la lógica de negocios debe estar separada de la capa de presentación, esto para aumentar la reutilización de código y a la vez tener un mayor control. Se recomienda dividir los eventos del sistema en el mayor número de controladores para poder aumentar la cohesión y disminuir el acoplamiento.

Alta cohesión y bajo acoplamiento:

Los conceptos de cohesión y acoplamiento no están íntimamente relacionados, sin embargo se recomienda tener un mayor grado de cohesión con un menor grado de acoplamiento. De esta forma se tiene menor dependencia y se especifican los propósitos de cada objeto en el sistema.

-Alta cohesión:

Nos dice que la información que almacena una clase debe ser coherente y debe estar (en la medida de lo posible) relacionada con la clase.

-Bajo acoplamiento:

Es la idea de tener las clases lo menos ligadas entre sí que se pueda. De tal forma que en caso de producirse una modificación en alguna de ellas, se tenga la mínima repercusión posible en el resto de clases, potenciando la reutilización, y disminuyendo la dependencia entre las clases.

```
const moment = require("moment");
const config = require("../config");
const jwt = require("jwt-simple");

function createToken(user) {
  const payload = {
    sub: user._id,
    iat: moment().unix(),
    exp: moment().add(14, "days").unix(),
  };
  return jwt.encode(payload, config.SECRET_TOKEN);
}

function decodeToken(token) {
  const decode = new Promise((resolve, reject) => {
    try {
      const payload = jwt.decode(token, config.SECRET_TOKEN);

      if (payload.exp <= moment().unix()) {
        reject({
          satatus: 401,
          sms: " el token a expirado",
        });
      }

      resolve(payload.sub);
    } catch (err) {
      reject({
        status: 500,
        sms: " invalid token ",
      });
    }
  });

  return decode;
}
```

Ilustración 7 Patrón GRASP Alta cohesión y Bajo acoplamiento

Experto en información:

El GRASP de experto en información es el principio básico de asignación de responsabilidades. Se indica, por ejemplo, que la responsabilidad de la creación de un objeto o la implementación de un método, debe recaer sobre la clase que conoce toda la información necesaria para crearlo. De este modo se obtiene un diseño con mayor cohesión y así la información se mantiene encapsulada (disminución del acoplamiento).

```
const mongoose = require("mongoose");
const Schema = mongoose.Schema;

const activoScheme = Schema({
  codigo: {
    type: String,
    unique: true // Establece que este campo debe ser único
  },
  descripcion:String,
  precio: Number,
  cantidad: Number,
  estado : String,
  Observacion: String,
  area: String,
  });

module.exports = mongoose.model("Activos", activoScheme);
```

Ilustración 8 Patrón GRASP Experto en información

2.10 Diagramas de clases del diseño

Un diagrama de clases de diseño (DCD) representa las especificaciones de las clases e interfaces software en una aplicación. A diferencia de las clases conceptuales del Modelo del Dominio, las clases de diseño de los DCD muestran las definiciones de las clases software en lugar de los conceptos del mundo real (Sommerville, 2011). A continuación, se muestra el DCD para el CU gestionar activos. El DCD está compuesto por 5 server_page (página controladora/servidora) que se corresponde con la clase principal de gestión de activos 4 client_page (página cliente/vista) que muestran las funcionalidades añadir y listar contenido, añadir, mostrar y editar planificación, y la página index correspondiente a la página principal del sistema, 4 formularios que contienen los campos botones y filtros asociados a los contenidos, la clase controladora que contiene todas las funcionalidades de la server_page (página controladora/servidora) , la clase que permite la conexión con la base de datos y la clase de la base de datos con los campos y funciones asociados al contenido.

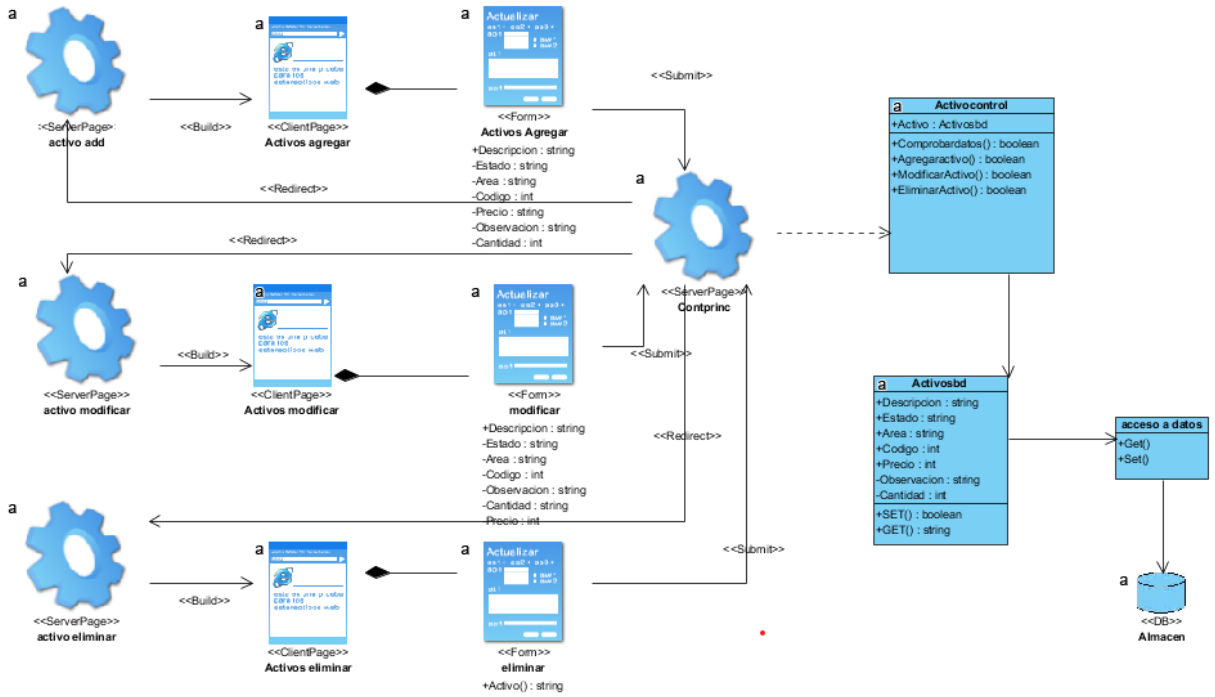


Ilustración 9 DCD con estereotipos web Gestionar Activos

(Ver el resto de los DCD con estereotipos en el anexo 4).

2.10.1 Diagrama de secuencia

Los diagramas de secuencia (DS) en el UML se usan principalmente para modelar las interacciones entre los actores y los objetos en un sistema, así como las interacciones entre los objetos en sí (Sommerville, 2011). Para la presente investigación se generó el diagrama de secuencia correspondiente al requisito funcional añadir planificación como se muestra a continuación.

El flujo representado en el DS comienza cuando el actor(administrador) del sistema selecciona en el mismo la opción de añadir activo. La página cliente intermediaria hace la solicitud a la server_page (página controladora/servidora). La página controladora construye la client_page (página cliente/vista) que permite añadir el activo, esta hace submit a la controladora que crea la página cliente con el formulario con los campos asociados a la planificación. El actor introduce los datos que se envían a la server_page (página controladora/servidora) y automáticamente se adiciona la planificación en la clase controladora donde se validan que los datos estén correctos. En caso de que los datos no hayan sido insertados de forma correcta se muestra un mensaje de error en la página cliente desde donde se creó el activo, en el caso contrario se muestra un mensaje de confirmación en la clase desde donde el actor realizo la solicitud inicial. Para el caso de modificar el activo comienza con el actor seleccionando el activo enviando la solicitud a la

client_page envía un submit al formulario con sus campos asociados, luego le envía la petición de modificar a la server_page (página controladora/servidora)automáticamente se adiciona la planificación en la clase controladora donde se validan que los datos estén correctos. En caso de que los datos no hayan sido insertados de forma correcta se muestra un mensaje de error en la página cliente desde donde se creó el activo, en el caso

contrario se muestra un mensaje de confirmación en la clase desde donde el actor realizo la solicitud inicial.

Y en el caso de eliminar activo comienza con el actor seleccionando el activo enviando la solicitud a la client_page envía un submit al formulario con sus campos asociados, luego le envía la petición de eliminar a la server_page (página controladora/servidora) automáticamente se adiciona la planificación en la clase controladora donde se validan que los datos estén correctos. En caso de que los datos no hayan sido insertados de forma correcta se muestra un mensaje de error en la página cliente desde donde se creó el activo,luego redirige esta peticion a la server_page que controla donde se adicionan o agregan los activos y esta construye la client_page.

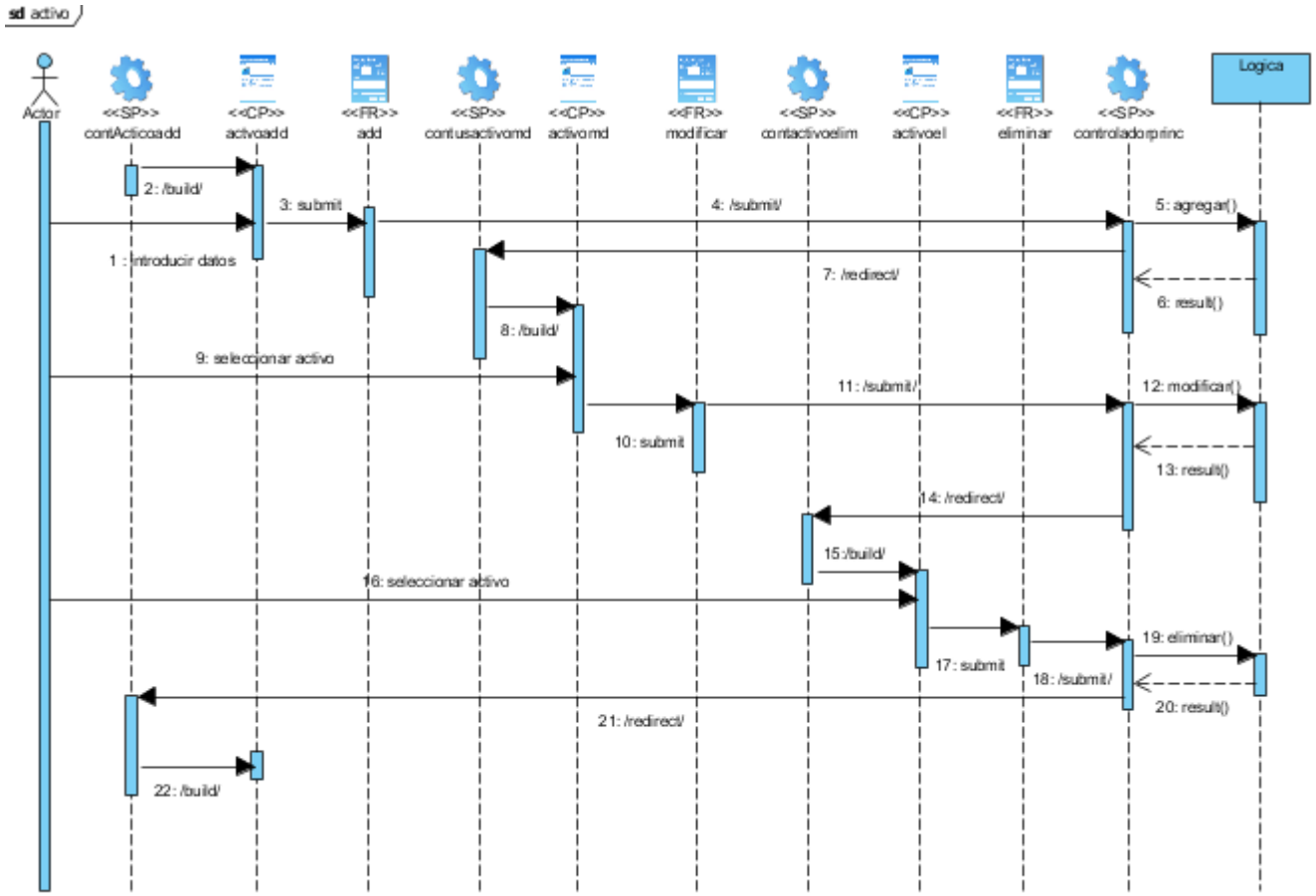


Ilustración 10 DS con estereotipos web Caso Uso Gestionar Activos

(Ver el resto de los DS con estereotipos web en el anexo 5).

2.11 Conclusiones del capítulo 2

Como parte del desarrollo del presente capítulo se determinan las siguientes conclusiones parciales:

1. El análisis de las características del sistema y la modelación del dominio permitió identificar los principales requisitos funcionales y no funcionales del sistema de gestión de activos , los cuales fueron agrupados y categorizados por casos de uso.
2. El diseño de los diagramas de clases y de secuencia facilitó la visión en cuanto a composición física y lógica del sistema.
3. La generación de todos los artefactos requeridos por el modelo de desarrollo documentaron la solución propuesta, lo cual facilitó su posterior mantenimiento (actualización o adición de funcionalidades).
4. La construcción del modelo de despliegue y la descripción del mismo brindó una mayor comprensión del sistema a desarrollar y de los elementos que lo componen.

CAPÍTULO 3: Implementación y prueba de la aplicación web para la gestión de activos fijos tangibles de la emisora Radio Reloj.

3.1 Introducción

Una vez que se sabe qué funciones debe desempeñar el sistema y se ha decidido cómo organizar sus distintos componentes (diseño), es el momento de pasar a la etapa de implementación. En esta fase, se toma como punto de partida el modelo de la fase anterior y se procede a programar los diseños especificados, los cuales son implementados en términos de componentes, ficheros de código fuente y ejecutables. Por otra parte, el desarrollo de un software es algo complejo y son innumerables las posibilidades de cometer errores. Por esta razón todo proceso de implementación debe ir acompañado de alguna actividad que garantice la calidad. Las pruebas de validación constituyen una base para garantizar la aceptación favorable de una aplicación informática por parte del usuario. Con la realización de las mismas se pretende encontrar y documentar los errores que tiene un sistema, validar los requisitos y comprobar que estos fueron implementados correctamente. Este capítulo tiene como objetivo, documentar los resultados de las fases de implementación del sistema y de la estrategia de pruebas desarrollada.

3.2 Modelo de Despliegue

Un modelo de despliegue consiste en una representación estructural de la arquitectura del sistema desde el punto de vista de la distribución de los artefactos del software en los destinos de despliegue; definiendo a los artefactos como representaciones de elementos concretos en el mundo físico que son el resultado de un proceso de desarrollo (Sarmiento, 2016).

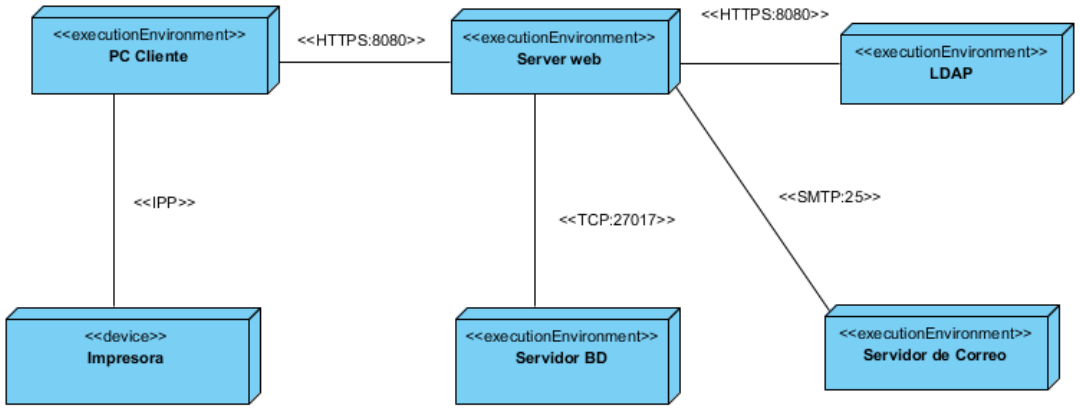


Ilustración 11 Modelo de despliegue

Descripción de elementos e interfaces de comunicación:

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

Servidor de aplicaciones: Es la estación de trabajo que hospeda el código fuente de la aplicación y que le brinda al usuario las interfaces para realizar los procesos del sistema. Esta estación se comunica con el servidor de base de datos donde se almacenan los datos de la aplicación realizando la comunicación mediante el protocolo TCP/IP (Transmission Control Protocol/Internet Protocol).

Servidor de BD: Este servidor es el encargado del almacenamiento de los datos del sistema. Se comunica con el servidor de aplicaciones del sistema, posibilitando el acceso mediante el usuario con privilegios para las operaciones determinadas a realizarse en el mismo.

Servidor de correo: Este servidor es el encargado del envío de correos electrónicos comunicándose a través del protocolo SMTP (Simple Mail Transfer Protocol).

LDAP: (Protocolo Ligero/Simplificado de Acceso a Directorio) es un protocolo de tipo cliente-servidor para acceder a un servicio de directorio. Los servicios de directorio facilitan el acceso a información organizada a una gran variedad de aplicaciones. Estos servicios le permiten a los usuarios y aplicaciones autorizadas buscar información de personas, computadoras, aplicaciones y dispositivos red.

Impresora: Es un dispositivo periférico del ordenador que permite producir una gama permanente de textos o gráficos de documentos almacenados en un formato electrónico, imprimiéndolos en medios físicos.

3.3 Diagrama de componentes

El diagrama de componentes muestra los componentes de un sistema de software conectados por las relaciones de dependencias lógicas entre cada uno de ellos. Provee una vista arquitectónica de alto nivel del sistema, ayudando a los desarrolladores a visualizar el camino de la implementación. Cada componente representa una unidad del código (fuente, binario o ejecutable), que permite mostrar las dependencias en tiempo de compilación y ejecución. La realización del diagrama posibilita tomar decisiones respecto a las tareas de implementación y los requisitos (RIVERA ALVA, 2017).

Diagrama de Componentes General:

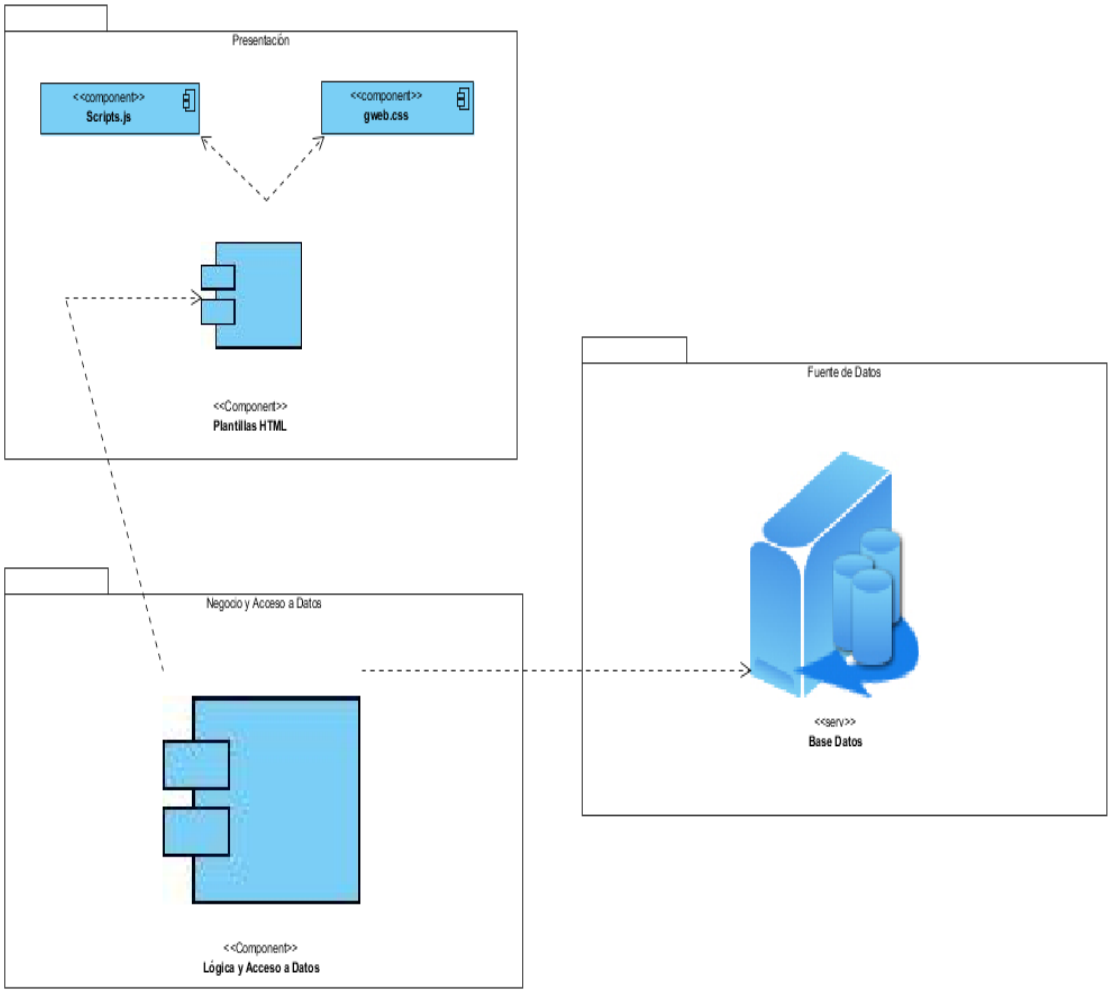


Ilustración 12 Diagrama de Componentes General

Diagrama de Componentes correspondiente al Paquete Capa de Presentación:

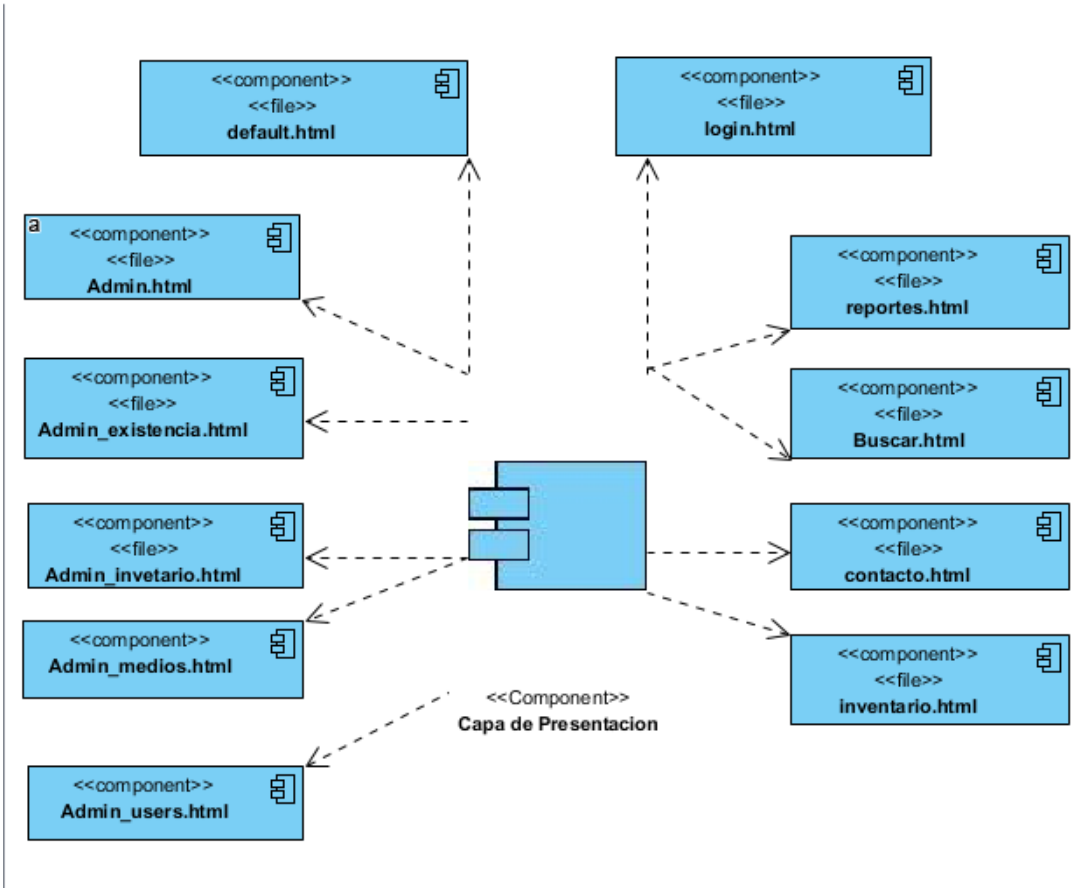


Ilustración 12 Diagrama de Componentes correspondiente al Paquete Capa de Presentación

Diagrama de Componentes correspondiente al Paquete Capa de Negocio y Acceso a Datos:

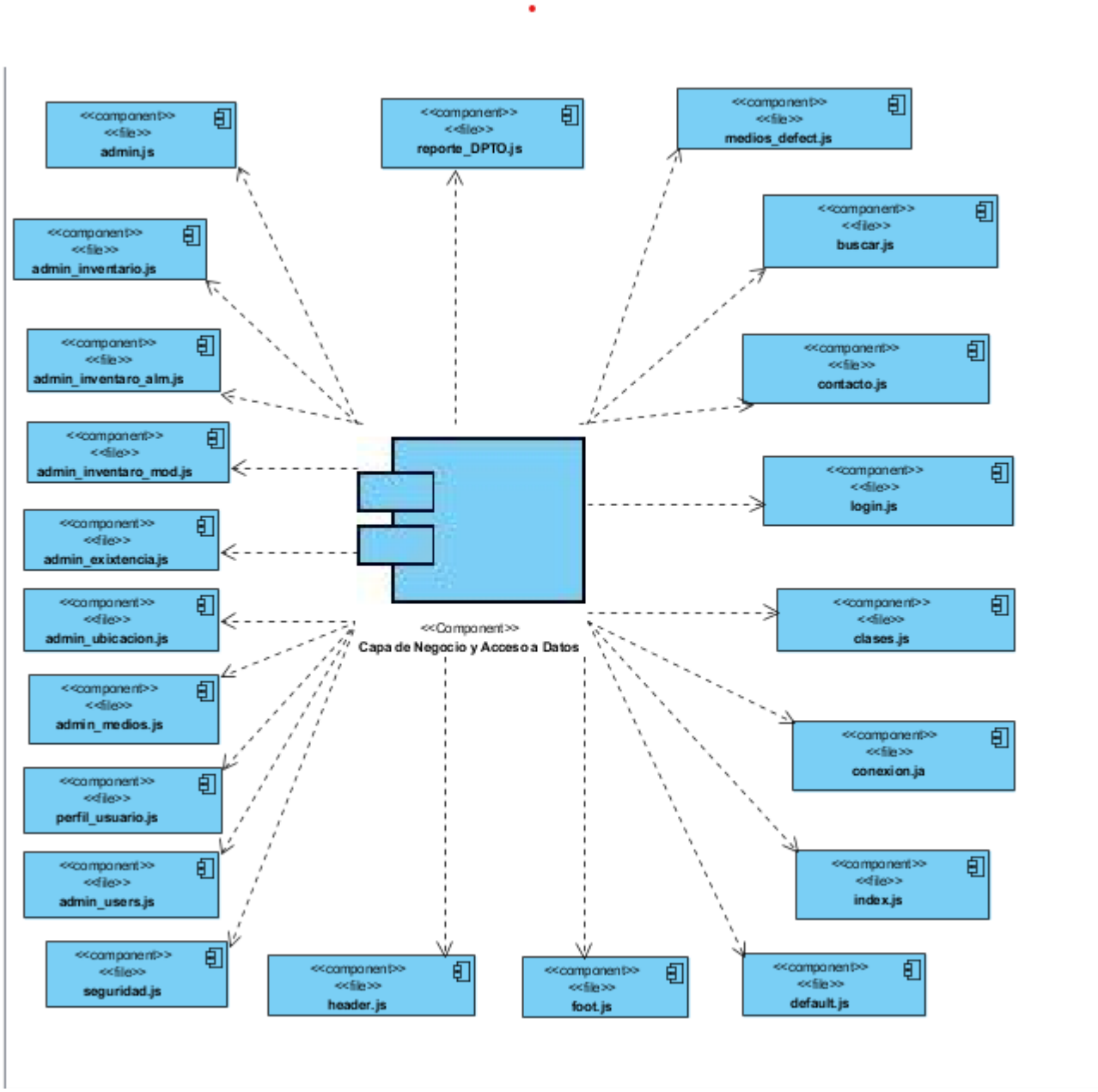


Ilustración 13 Diagrama de Componentes correspondiente al Paquete Capa de Negocio y Acceso a Datos

3.4 Estándares de codificación

Los estándares de codificación constituyen un principio esencial en el desarrollo de software. Garantizan que el código obtenido sea fácil de leer, entendido y modificado independientemente de quien haya sido el desarrollador del producto. Son una guía para el equipo de desarrollo, permiten asegurar que el código presente calidad y no contenga errores. Actualmente, se encuentran estándares de codificación para la mayoría de los lenguajes existentes. El uso de los mismos, partiendo de las convenciones definidas, permite una mejor comunicación entre los programadores, creando las condiciones para la reusabilidad y el mantenimiento de los sistemas. Para definir el estilo de codificación a seguir en la aplicación (Castillo, 2019).

.

Declaración de variables:

Declarar variables es una de las cosas más importantes a la hora de codificar, por eso es necesario tener en cuenta las siguientes premisas:

Nombres de variables:

Los nombres de variables deben ser escritos con notación camelCase.

```
var test;  // Nombre de variable correcto
var i_am_bad; //  Nombre de variable incorrecto
var iAmFine; //  Nombre de variable correcto
```

Valores de variables:

Los espacios entre el nombre y el valor de una variable son muy importantes, puesto que mejoran la legibilidad del código.

```
var test = 1;  // Nombre de variable correcto
var iAmFine = true; //  Nombre de variable correcto
```

Declaración de multiples variables:

Cuando se declaran múltiples variables deben declararse todas con la misma sentencia var, una variable por línea y con coma (,) al final a menos que sea la última variable, en cuyo caso se usará punto y coma (;) al final de la línea. A partir de la segunda línea debe usarse 4 espacios antes del nombre de variable para mejorar la legibilidad del código fuente. El orden de las variables debe ser alfabético de ser posible.

- Una declaración errónea es de la siguiente manera:

```
// declaración errónea
var bad1 = 1;
var iAmFine = false;
var sampleVariable = "wrong";
```

- Una declaración correcta es de la siguiente manera:

```
// declaración correcta
var iAmFine = true, // usa coma
    greatDay = 1, // usa coma
        sampleVariable = true; // usa punto y coma por qué es la última variable
```

Nota: Es aconsejable que cada variable tenga un comentario acerca de su objetivo, de manera que aumente la comprensibilidad del código.

Variables de tipo colección:

Cuando la variable a declarar contiene un elemento de tipo colección (arreglo y objeto) y su contenido excede los 70 caracteres o tiene más de 2 elementos debe tener cada ítem de la colección en una línea diferente para mejorar la legibilidad del código.

```
// declaración correcta
var elements = [], // colección vacía
    gender = ["male", "female"], // colección con 2 elementos
    users = {
        "Hugo": "Duck 1",
        "Paco": "Duck 2",
        "Luis": "Duck 3"
    }, // colección con más de 2 elementos
    attributes = [
        "fuerza",
        "velocidad",
        "inteligencia",
        "carisma",
    ],
    base64_logo_parts = [
        'da-
ta:image/png;base64,iVBORw0KGgoAAAANSUhEUgAAABgAAAAYCAMAAADXqc3KAAAB+',
        'FBM-
VEUAAAA/mUPidDHiLi5Cn0XkNTPmeUrkdUg/m0Q0pEfcpSbwaVdKskg+lUP4zA/iLi3m',
        'sSHkOjVAmE-
TdJSjtYFE/lkPnRj3sWUs8kkLeqCVIq0fxvhXqUkbVmSjwa1n1yBLepyX1xxP',

'0xRXqUkboST9KukpHpUbuvRrzrhF/ljbwaljuZFM4jELaoSdLtElJrUj1xxP6zwzfqSU4i0',
        // ...
        // more lines
```

```
        // ...

'/0rbNvHVxiJywa8yS2KDfV1dfbu31H8jF1RHiTKtWYeHxUvq3bn0pyjCRaiRU6aDO+gb3aE',
        'fEeVNs-
Dgm8zzLy9egPa7Qt8TSJdwhjplk06HH43ZNJ3s91KKCHQ5x4sw1fRGYDZ0n1L4FKb',
        '9/BP5JLYxToheoFCVxz57PPS8UhhEpLBVeAAAAAElFTkSuQmCC'
    ]; // colección con más de 70 caracteres
```

Declaración de funciones:

Uno de los elementos más importantes de javascript son las funciones, éstas son conocidas como ciudadanos de primer tipo. Para la declaración de funciones es necesario tener en cuenta las siguientes premisas:

Nombres de funciones:

Los nombres de funciones deben ser escritos con notación camelCase.

```
function testFunction() {
    // code here
}
```

Llaves y espaciado de funciones:

Las llaves de las funciones deben empezar en la misma linea que se declara la función y debe haber un espacio entre el parentesis de cierre de argumentos y la llave de inicio de cuerpo de función, así:

```
// A. Declaración de función
// B. Parentesis de argumentos
// C. Espacio
// D. Llave de inicio de cuerpo de función.

// ------- A ------- --- B --- C D
function testFunction(username) {
    console.log('Hello ' + username + '...');
}
```

Tipos de funciones:

En javascript las funciones pueden ser funciones como tal o funciones como variables. En caso de que sean funciones como variables es necesario usar punto y coma (;) despues del cuerpo de la función para finalizar la sentencia.

```
function testFunction() {
    console.log('Hello Test...');
```

```
}

var anotherTestFunction = function() {
    console.log('Hello Test...');
}; // ; para fin de sentencia.
```

Argumentos de una función:

Para mejorar la legibilidad del código javascript los argumentos de funciones deben ser nombres con notación camelCase. Si son varios argumentos deben estar separados por coma (,) y un espacio, así:

```
function testFunction(arg1, arg2, arg3) {
    console.log('Hello Test...');
}

var testFunction = function(arg1, arg2) {
    console.log('Hello Test...');
}; // ; para fin de sentencia.
```

Si unicamente hay un argumento no hay espacios a ningún lado, así:

```
function testFunction(arg1){
    console.log('Hello Test...');
}
```

Construcción de cadenas:

Deben ser usadas comilas simples (') puesto de ésta manera se reducen los problemas a la hora de escapar las comillas dobles (") usadas cuando se genera HTML y mejora la legibilidad.

```
// Incorrecto
var container0 = "<div class=\"another-container\" data-type=\"html\"></div>";
var container1 = "<div class="+'"'+"another-container"+'"'+
    " data-type="+'"'+"html"+'"'+"></div>";
// Correcto
var containerString = '<div class="container" data-type="html"></div>';
```

Nota: Para los casos en que no se estan construyendo elementos del DOM también deben ser usadas las comillas simples por uniformidad en el código.

Operaciones:

Todas las operaciones deben tener espacios entre los operadores y los operandos para mejorar la legibilidad del código. Si se están usando parentesis no es necesario usar espacio entre los parentesis y los operandos.

```
var v1 = b * h / 2,
    v2 = (location.toLowerCase() === 'medellin') ? 'frijoles' : 'lentejas';
```

Operaciones lógicas de igualdad y desigualdad:

Deben usarse los comparadores estrictos (aquellos que comparan valor y tipo ===, !==) para todas las operaciones lógicas de comparación.

```
function factorial(x) {
    if(x === 0){
        return 1;
    } else if(x !== 0) {
        return factorial(x - 1) * x;
    }
}

factorial(10);
```

3.5 Estrategia de pruebas

El primer enfoque de prueba se denomina “Prueba de caja negra” y el segundo “Prueba de caja blanca (Vera, 2020).

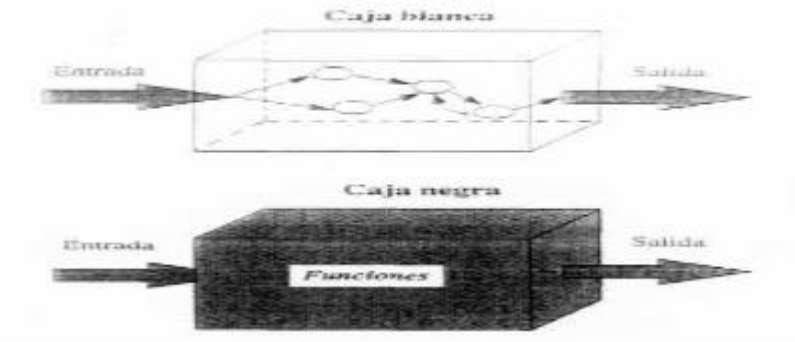


Ilustración 14 Tipos de Prueba

Tipos de prueba:

- En los módulos, Pruebas de unidad.
- En la unión de los módulos, Pruebas de integración.
- Cuando tenemos todos unidos, Prueba de Aceptación.
- Cuando el sistema está funcionando, Prueba de sistema.

La estrategia que se llevará acabo para realizar las pruebas de este sistema de gestión de activos fijos tangibles será la siguiente:

Para la prueba a nivel de unidad se utilizará la prueba de caja blanca, haciendo el uso de la **técnica de camino básico**. A nivel de sistema se aplicará la prueba de caja negra, haciendo el uso de las herramientas **Apache Jmeter** para evaluar y medir el rendimiento, carga y estrés de la aplicación web y con respecto a la seguridad de la aplicación web se hará uso de la herramienta **Acunetix WVS**. Para concluir este proceso de pruebas se efectuará un encuentro con el cliente, donde el mismo evaluará y emitirá su criterio sobre la aplicación web donde se buscará llegar al acuerdo y una **aceptación** del resultado obtenido.

3.5 Aplicación de la estrategia de prueba

Para garantizar el completo funcionamiento de toda aplicación informática es necesario realizar pruebas en todas sus categorías y de esta forma evitar cualquier problema o insatisfacción por parte de los clientes.

Estas pruebas se trazan después de estrategias que permitan evaluar todos los aspectos de determinado producto teniendo en cuenta su característica y así de una forma u otra estar completamente confiados y poder garantizar el éxito del mismo. La estrategia de pruebas realizadas está dirigida a componentes del software, analizando la usabilidad, disponibilidad, respuesta, las pruebas de seguridad y la aceptación por parte del cliente (Diaz, 2020).

Pruebas unitarias:

Las pruebas unitarias están enfocadas al código fuente de los componentes, realizado para verificar todos los flujos de control pasando primero por la revisión del programador. Verifican el funcionamiento aislado de piezas de software que pueden ser probadas de forma separada. (Pressman, 2010).

Para aplicar las pruebas unitarias el autor de la presente investigación define utilizar el método de Caja blanca. Con el empleo de este método es posible desarrollar casos de prueba que garanticen la ejecución, al menos una vez, de los caminos independientes. Para aplicar este método se define la técnica de camino básico.

Técnica de camino básico:

La técnica de camino básico es empleada en el método de Caja blanca este se realiza sobre el código, en él se comprueban los caminos lógicos del software y se puede examinar el estado del programa en varios puntos para determinar si el estado real coincide con el esperado. La misma ruta tiene como objetivo comprobar que cada camino se ejecute independiente de un componente o programa, obteniéndose una medida de la complejidad lógica del diseño (Pressman, 2010).

Pressman propone como estrategia para aplicar la ruta básica, realizar un análisis de la complejidad ciclomática de cada procedimiento que componen las clases del sistema; una vez concluido este paso se selecciona el método con valor de contener errores, además de que ofrece una medida del número de pruebas que deben diseñarse para validar la correcta implementación de una determinada función. Esta métrica se calcula sobre un grafo y se puede realizar mediante tres formas distintas:

1. V(G) = R
2. V(G) = E – N + 2
3. V (G) = P + 1

Conociendo que:

- G: Grafo de flujo (grafo).
- R: El número de regiones contribuye a estimar el valor de la complejidad ciclomática.
- E: Número de aristas.
- V (G): Complejidad ciclomática.
- N: Número de nodos del grafo.
- P: Número de nodos predicados incluidos en el grafo

Una vez calculada la complejidad ciclomática, el valor obtenido representa el límite superior de pruebas que deberán aplicarse (Pressman, 2010).

En la siguiente imagen se muestra el método async function loginUser() des caso de prueba Autenticar Usuario.

```
async function loginUser() {
  let email = document.getElementById("correo").value;
  let password = document.getElementById("password").value;

  if (email == "" || password == "") {
    alert("Rellene todos los campos"); //  para que no haya campos vacios
  } else if (/^[a-z0-9!#$%&'*+/=?^_`{|}~-]+(?:\.[a-z0-9!#$%&'*+/=?^_`{|}~-]+)*@(?:[a-z0-9](?:[a-z0-9-]*[a-z0-9])?\.)+[a-z0-9](?:[a-
    alert("Por favor escriba un correo válido");
  } else {
    let response = await fetch("http://localhost:3001/api/signIn", {
      method: "POST",
      headers: {
        "Content-Type": "application/json",
      },
      body: JSON.stringify({
        email,
        password,
      }),
    })
      .then((res) => {
        return res.json();
      })
```

1 2 3 4 5 6

Ilustración 15 método async function loginUser()

A continuación, se muestra el grafo de flujo para el método.

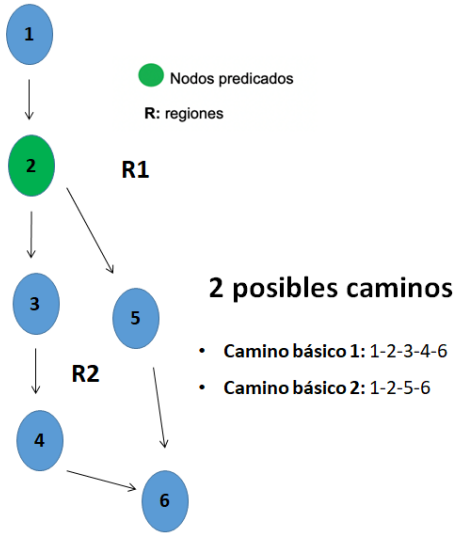


Ilustración 16 Grafo de flujo para el método

Tabla 4 Camino Básico Caso de prueba Autenticar Usuario

Trayectoria	1-2-3-4-6
Entrada	Verifica si el estado de la operación de autenticar usuario revisando que no estén los campos vacíos de ser así no te deja avanzar hasta que completes la operación, luego verifica que el correo este bien escrito de no ser así no te permitirá continuar hasta completada la operación luego verificado que todo este correcto autentica el usuario y te envía hacia la página principal del sistema.
Trayectoria	**1-2-5-6**
Entrada	Verifica si el estado de la operación de autenticar usuario es correcta y ha tenido éxito y lo envía a la página principal del sitio

Pruebas de Sistema:

Tradicionalmente, las pruebas del sistema se realizan cuando el producto software está completado. El objetivo es evaluar si un producto software cumple con los requisitos que han sido especificados. Un ciclo de vida iterativo permite probar el sistema mucho más tempranamente tan pronto como los subconjuntos bien formados de requisitos funcionales se han construido (Pressman, 2010). Para llevar a cabo estas pruebas se tuvo en cuenta el método de Caja negra que a continuación se describe.

Pruebas funcionales:

Se denominan pruebas funcionales a las pruebas de software que tienen por objetivo probar que los sistemas desarrollados, cumplan con las funciones específicas para los cuales han sido creados(González, 2019).

Método de Caja negra:

Esta prueba se lleva a cabo sobre la interfaz del software. Tiene como objetivo demostrar que las funciones del software son operativas, que las entradas se aceptan de forma adecuada y se produce un resultado correcto, y que la integridad de la información externa se mantiene. Estas pruebas permiten encontrar (Pressman, 2010):

- Funciones incorrectas o ausentes.
- Errores de interfaz.
- Errores en estructuras de datos o en accesos a las bases de datos externas.
- Errores de rendimiento.
- Errores de inicialización y terminación.

Para llevar a cabo el método de Caja negra se utiliza la técnica de Partición de equivalencia que a continuación se describe:

Técnica de prueba:

Partición de equivalencia:

Esta técnica divide el dominio de entrada de un programa en clases de datos a partir de las cuales pueden derivarse casos de prueba. El método se esfuerza por definir un caso de prueba que descubra ciertas clases de errores, reduciendo así el número total de casos de prueba que deben desarrollarse. Una clase de equivalencia representa un conjunto de estados válidos y no válidos para las condiciones de entrada (Pressman, 2010).

Casos de pruebas:

Un ejemplo de los escenarios de los casos de pruebas generados se puede observar en la tabla:
A continuación, se muestra un caso de prueba (Caso de prueba “Autenticar Usuario”)

Tabla 5 Equivalencias válidas e inválidas

No	Nombre de campo	Clasificación	Valor Nulo	Descripción
1	Correo electrónico	• **Dirección de correo electrónico:** Obligatorio.	No	El usuario in-

			Campo de texto. Longitud máxima 255 caracteres. Permite direcciones de correo estructuralmente válidas (Estructura: usuario@subdominios.dominio).		gresa su correo electrónico en el sistema.
2	Contraseña		• **Contraseña:** Obligatorio. Campo de texto. Longitud mínima 6, máxima 128 caracteres. Permite cualquier caracterer.	No	El usuario ingresa su contraseña con la que desee ser autenticado en el sistema.

Tabla 6 Descripción del caso de prueba del requisito Autenticar usuario

Escenario	Descripción	Variable 1	Variable 2	Respuesta del sistema	Flujo central
EC 1.1 Autenticar correo electrónic y contraseña, en el sistema.	Identifica satisfactoria mente el usuario en el sistema.	V rrr@gmail .com	V 123456	Autenticación realizada de forma satisfactoria con su correo electrónico y contraseña.	Selecciona la opción autenticar.
EC 1.2	Introduce datos incorrectos.	V rrr@gmail .com	I 123@*	Correo correcto pero contraseña no válida	Selecciona la opción autenticar.
EC 1.3 Los datos son incorrectos…	Introduce datos incorrectos.	I rrr@gmai 12.34	V 123456	Correo incorrecto pero contraseña	Selecciona la opción autenticar.

				válida.	
EC 1.4 Existen campos obligator ios vacíos.	Deja campos en blanco	I	I	Notifica que existen campos obligatorios vacíos.	Selecci ona la opción autenticar.

Tabla 7 Registro de defectos y dificultades detectados

Elemento	No	No conformidad	Aspecto correspondiente	Significativa	No Significativa	Recomendación	Estado NC	Respuesta del equipo de desarrollo
CP Gestionar usuario SC1 Autenticar usuario EC Elementos con datos incorrectos	1	Permite la autenticación del usuario con datos incorrectos.	Permite la autenticación con el Correo mal redactado.	X			PD: 25-10-23 RA: 29-10-23	El equipo de desarrollo añadió nuevas funcionalidades.
CP Gestionar usuario SC3 Agregar usuario EC Permite Agregar usuarios con elementos vacíos.	2	Permite agregar usuarios con elementos vacíos.	Permite agregar usuarios con elementos vacíos.	X			PD: 25-10-23 RA: 29-10-23	El equipo de desarrollo añadió nuevas funcionalidades.
CP Gestionar usuario SC2 Modificar	3	El botón de aceptación no tiene el texto de identificación	El botón de aceptación no tiene el texto		X		PD: 28-10-23 RA: 29-10-23	El equipo de desarrollo añadió nuevas

usuario EC botón de aceptación			correcto					funcionalidades.

Pruebas de rendimiento (carga y estrés)

Las pruebas de rendimiento se diseñan para asegurar que el sistema pueda procesar su carga esperada.

Éstas se ocupan tanto de demostrar que el sistema satisface sus requerimientos, como de descubrir problemas y defectos en el sistema (Sommerville, 2011).

Las pruebas de carga consisten en simular una carga de trabajo similar y superior a la que tendrá cuando el sistema esté funcionando, con el fin de detectar si el software instalado (programas y aplicaciones) cumple con los requerimientos de muchos usuarios simultáneos y también si el hardware (servidor y el equipamiento computacional de redes y enlace que lo conecta a Internet) es capaz de soportar la cantidad de visitas esperadas (PRESSMAN, 2002).

Las pruebas de estrés evalúan la robustez y la confiabilidad del software sometiéndolo a condiciones de uso extremas. Entre estas condiciones se incluyen el envío excesivo de peticiones y la ejecución en condiciones de hardware limitadas. El objetivo es saturar el programa hasta un punto de quiebre donde aparezcan defectos potencialmente peligrosos (PRESSMAN, 2002).

Resultados de las pruebas de rendimiento:

Para las pruebas de rendimiento se utiliza el software Apache Jmeter v5.5 Para ello se definen las propiedades de las PC implicadas.

Hardware de prueba (PC servidor)

• Sistema Operativo: Windows 10
• Microprocesador: Intel(R) Core(TM) i3-7100U CPU @2.40GHz 2.40GHz
• Memoria RAM: 8.00 GB
• Disco Duro: 1024 GB

Software instalado:

- Tipo de servidor web: Apache.

- Plataforma: SO Windows.
- Servidor de BD: MongoDB.

Luego de definido el hardware se configuran los parámetros del Apache JMeter logrando un ambiente de simulación con un total de 50 usuarios conectados concurrentemente, se realizan peticiones a diferentes páginas del Sistema para la Gestión de activos fijos tangibles de Radio Reloj. En la (tabla 3) se pueden observar los resultados obtenidos por el sistema.
A continuación, se describen las variables que miden el resultado de las pruebas de carga y estrés realizadas al caso de prueba **Gestionar Activos**:

• **Usuarios:** total de usuarios.
• **# Muestras:** El número de peticiones.
• **Media:** El tiempo medio transcurrido en milisegundos para un conjunto de resultados.
• **Mín:** El mínimo tiempo transcurrido en milisegundos para las muestras de la URL dada.
• **Máx:** El máximo tiempo transcurrido en un milisegundo para las muestras de la URL dada.
• **% Error:** Porcentaje de las peticiones con errores.
• **Rendimiento:** Rendimiento medido en base a peticiones por segundo/minuto/hora.
• **Kb/s Recibidos:** Rendimiento medido en Kbytes por segundos.

Tabla 8 Resultados de las pruebas de rendimiento

Usuarios	# Muestras	Media	Mín	Máx	% Error	Rendimiento (peticio-nes/segundos)	Kb/s Recibidos
50	100	29964	2902	31866	0 %	1.6	1.7

El sistema desarrollado, para un total de 50 usuarios conectados de forma concurrente respondió 100 peticiones al servidor en un promedio de 29.964 segundos, lo que equivale a 1.6 peticiones por segundo. Atendiendo a la cantidad de peticiones por cada segundo que se enviaron y las prestaciones del hardware donde se realizaron las pruebas se considera que constituye un resultado satisfactorio.

Pruebas de seguridad:

Las pruebas de seguridad se realizan para comprobar que los mecanismos de protección integrados en el sistema realmente lo protejan de irrupciones inapropiadas (PRESSMAN, 2002). Además, se encargan de certificar que los datos y las funciones del sistema solo son accesibles por los actores debidamente autorizados(Bautista, 2021).

Resultados de las pruebas de seguridad:

Con el objetivo de evaluar la seguridad de la solución propuesta al caso de prueba Gestionar Usuario se emplea la herramienta Acunetix WVS la cual arrojó los siguientes resultados.

Tabla 9 Resultados de la prueba de seguridad

Categorías de vulnerabilidades	Cantidad de errores
Vínculos rotos	1
Ataque de adivinación de contraseñas en la página de inicio de sesión	1
Las credenciales de usuario se envían en texto claro	5
Secuencias de comandos entre sitios	1
Total	8

La prueba realizada mediante la herramienta Acunetix WVS al caso de prueba Autenticar Usuario , arrojó que en la primera iteración se detectan 8 no conformidades, de ellas 2 alertas de riesgo alto, 5 alertas de riesgo medio y 1 de riesgo bajo. Para proporcionar una mayor seguridad se configuró el servidor de forma tal que solo pudieran acceder al sistema los usuarios con permiso, así como la instalación de módulos para mayor seguridad tales como SSL (Secure Sockets Layer) y TLS (Transport Layer Security)

Pruebas de usabilidad:

Según diversos estándares de la Ingeniería de Software, se puede definir la usabilidad como el grado en el que un producto puede ser utilizado por usuarios para conseguir objetivos específicos con efectividad, eficiencia y satisfacción, en un determinado contexto de uso. Como se puede

apreciar, la usabilidad de un sistema está ligada a usuarios, necesidades y condiciones específicas (Carcasés, 2016). De manera general, el término usabilidad es empleado para referirse a la capacidad que posee un producto de ser utilizado por los usuarios de forma fácil, eficiente y con satisfacción, en un determinado contexto de uso.

Se puede decir que el proceso de prueba de usabilidad se enfoca en satisfacer las necesidades de los usuarios finales basados en métricas definidas durante la planeación. Para la realización de las pruebas de usabilidad, se hace uso de la “Lista de Chequeo de Usabilidad para sitios web”, desarrollada por los especialistas del grupo de Seguridad del Departamento de Evaluación de Productos de Software (DEPSW), perteneciente al Centro Nacional de Calidad de Software (CALISOFT). A continuación, se muestran los resultados de dichas pruebas.

Tabla 10 Resultados de la prueba de usabilidad

Categoría de los indicadores	Indicadores	Proceden	Correctos	Incorrectos
Visibilidad del sistema	17	14	12	2
Lenguaje común sistema/usuario	12	10	9	1
Libertad y control por parte del usuario	29	24	18	6
Consistencia y estándares	33	27	22	5
Estética y diseño minimalista	17	15	15	0
Prevención de errores	8	7	6	1
Ayuda y documentación	11	11	7	4
Flexibilidad y eficiencia	6	4	3	1
Ayuda a los usuarios a reconocer, diagnosticar y recuperarse de los errores	11	11	8	3
Total	144	123	100	23

Como se observa en la tabla, de los 144 parámetros originales de la lista de chequeo, solo

proceden 123. En la primera iteración se evaluaron como correctos 100 parámetros, identificando 23 no conformidades, para un 81.30 % de usabilidad. Algunos de los problemas detectados en primeras iteraciones de las pruebas de usabilidad fue que el sistema luego de una acción relevante relacionada con los contenidos del mismo no había vuelta atrás a la página donde se encontraba inicialmente el administrador. Siguiendo el principio de la realización de un sistema intuitivo para todo usuario habiendo o no trabajado antes con la tecnología se validó luego de la modificación del contenido desde las páginas internas del mismo al realizar una acción relevante el sistema volvía a la vista principal de dicho contenido, otra de las no conformidades detectadas y resueltas estuvo relacionada con las resoluciones de pantalla, en un inicio el sistema solo era visible de forma correcta para determinadas resoluciones y luego de reajustado lo cumple para todas.

Pruebas de aceptación

En ingeniería de software y pruebas de software, las pruebas de aceptación pertenecen a las últimas etapas previas a la liberación en firme de versiones nuevas a fin de determinar si cumplen con las necesidades y/o requerimientos de las empresas y sus usuarios (PRESSMAN, 2002).

Por su parte, la Junta Internacional de Cualificaciones de Pruebas de Software (*ISTQB* por sus siglas en inglés) define la “Aceptación” aplicado a la rama de la informática como: Pruebas formales que se realizan atendiendo las necesidades del cliente, requerimientos y procesos de negocio, realizadas para determinar si un sistema satisface los requerimientos de aceptación que permitan que el cliente pueda determinar si acepta o no el sistema (González, 2019).

Para realizar la prueba de aceptación, se entregó la aplicación al cliente, el cual emitió su criterio a través de una carta de aceptación, a partir de sus consideraciones respecto a las ventajas que ofrecen el sistema y las necesidades que resuelve.

3.6 Interfaces principales del Sistema de gestión de activos fijos tangibles para Radio Reloj

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

La pantalla actual es la de activos por áreas. Dicha vista solo es accesible para el administrador del sistema que es el encargado de gestionar las mismas mediante los botones que contiene la vista que le permiten trabajar con las con los activos y las áreas o departamentos a los cuales pertenecen.

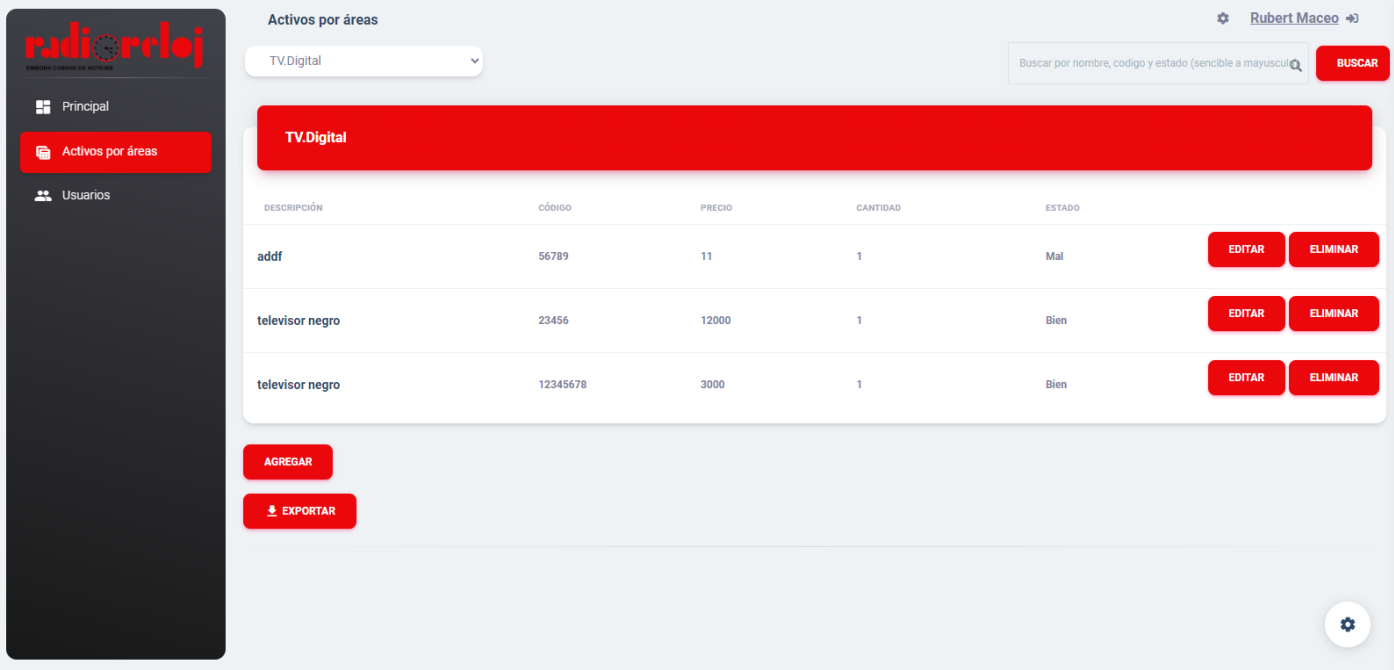


Ilustración 17 Captura de pantalla del sistema Activos por Áreas

La pantalla actual es la vista principal del sistema relacionada con el contenido donde muestras todas lao departamentos de Radio Reloj en la cual se interactuar con la información precisa de cada área haciendo clic sobre el departamento que elijas en la cual accederá a toda información asequible de dicho departamento.

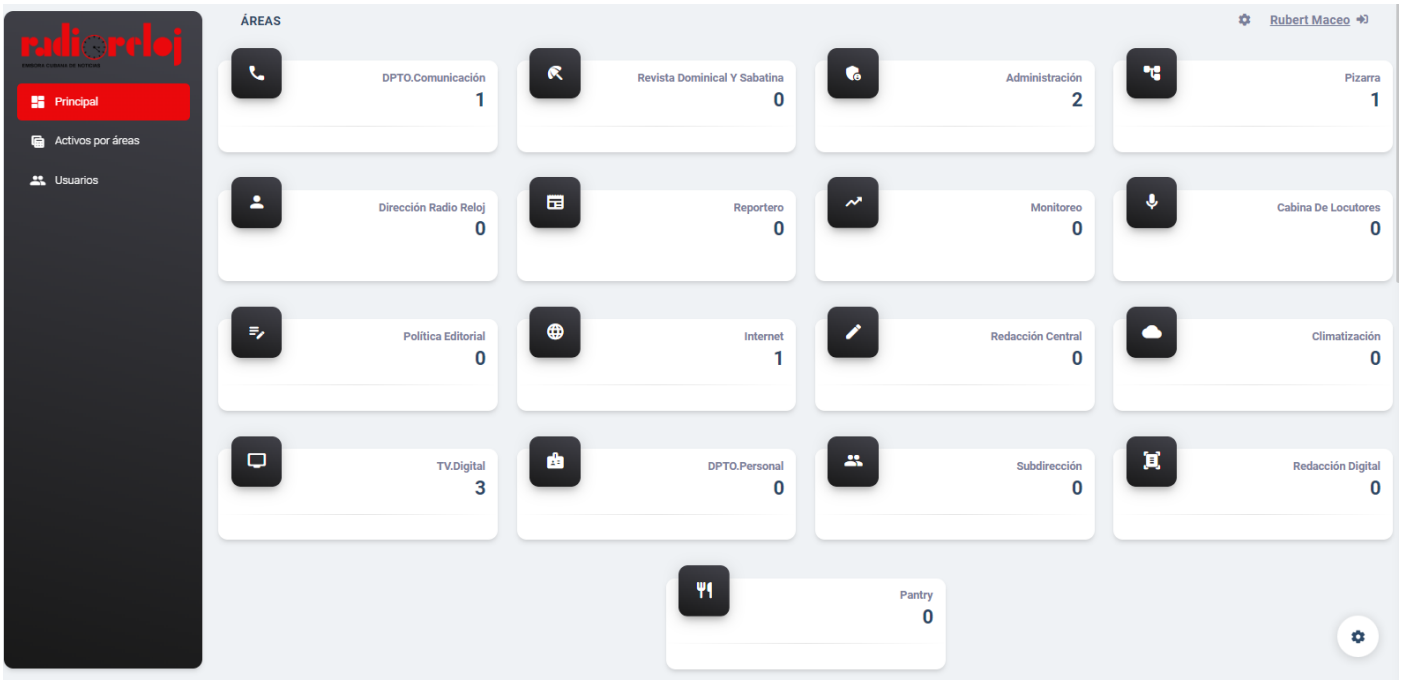


Ilustración 18 Captura de pantalla del sistema Mostrar departamentos

3.6 Conclusiones parciales

Como parte del desarrollo del presente capítulo se determinan las siguientes conclusiones parciales:

- La confección del diagrama de componentes permitió observar la integración de los componentes de software.
- Aplicar los estándares de codificación permitió obtener en el sistema un código legible, mantenible, estándar y fácil de comprender lo que asegura la calidad y facilita un futuro mantenimiento.
- El proceso de validación de la solución propuesta a través de las pruebas de carga y estrés, funcionalidad, seguridad, usabilidad y aceptación permitió identificar y corregir las no conformidades detectadas para obtener un producto de mayor calidad.

Conclusiones Generales

- El estudio de los referentes teóricos y el análisis de las diferentes herramientas y tendencias para la gestión de contenidos permitió determinar la no existencia de un sistema informático que responda a las necesidades requeridas por el cliente.
- El diseño de la propuesta de solución permitió generar los artefactos más significativos de acuerdo con la metodología de desarrollo de software AUP-UCI tomándose como referencia los requisitos detectados. La implementación del sistema a través de las herramientas y lenguajes seleccionados permitió obtener una aplicación web capaz de manejar datos referentes a los activos fijos tangibles y la gestión o manejo que realizan a los mismos.
- Las técnicas de validación aplicadas a la propuesta de solución permitieron la detección y corrección de las no conformidades detectadas y evidenciaron que el sistema constituye una solución funcional.
- La validación del problema de investigación mediante la carta de aceptación emitida por el cliente demostró que el Sistema para la gestión de activos fijos tangibles contribuye a informatizar el proceso de control y manejo de estos mismos en la emisora Radio Reloj.

Recomendaciones

Al concluir el presente trabajo investigativo, se recomienda para investigaciones futuras:

- Desarrollar una versión que aproveche las principales tendencias del desarrollo de software actual, como son el desarrollo de software para dispositivos móviles y la computación en la nube
- Se continúe el estudio y análisis más profundo en las áreas de la institución con vistas a la adecuación e implementación del sistema informático que se propone, por ser un proceso que está presente en cada una de ellas.
- A partir de la propuesta de solución que se muestran en esta investigación, que otros profesionales estudien la adaptabilidad del sistema informático presentado.

Bibliografía

Alfonso Benítez, D. (2017). Herramienta para generar productos de trabajos de la metodología variación AUP-UCI (Bachelor's thesis, Universidad de las Ciencias Informáticas. Facultad 3.).

Álvarez, Miguel Ángel. Maestros del Web. Sitio Web Maestros del Web. [Online] Noviembre 3, 2003. [Cited: Abril 26, 2007.] Encargado del sitio de Desarrolloweb.com. http://www.maestrosdelweb.com/editorial/zendstudio/

http://repositorio.ug.edu.ec/handle/redug/27304

Antigua, M. (2021, marzo 12). Assets y sus Generalidades (página 2). (Antigua, 2021)

Anchundia Medrano, L. A. (2022). Análisis comparativo de tecnologías Front End Angular Js Vs React Js, en el modelo de procesos para el desarrollo de aplicaciones web (Bachelor's thesis, Babahoyo: UTB-FAFI. 2022).

Ana Isabel Gómez-Varela, N. B.-V. (2016). Diseño de Aplicaciones Web Educativas conHTML5: el Efecto Fotoeléctrico Valencia.

Ballamudi, V. K. R., Lal, K., Desamsetti, H., & Dekkati, S. (2021). Getting Started Modern Web Development with Next. js: An Indispensable React Framework. *Digitalization & Sustainability Review*, *1*(1),

Buytaert, Dries. 2016. Drupal.org. [Online] 2016. [Citado em: 13 de enero de 2019.]

Bircher, C. (2017). Arquitectura Cliente-Servidor/Cliente-Servidor en Aplicaciones de Misión Crítica.

Carreño Villalba, R. (2016). Aplicación de administración con Angular, Node y Express para una aplicación Django.

Castro, L. (2015). Arquitectura del Software. Cengage Learning Editores.

Castillo Quiel, Y., Saavedra, A., & Villarreal, V. (2019). Estándares de codificación e interoperabilidad en Salud: evaluación del proyecto AmIHEALTH. Revista Cubana de Información en Ciencias de la Salud, 30(3).

EGUÍLIZ, Javier. Introducción a JavaScript. [En línea]. 2020. [Citado el: 17 junio 2021]. Disponible en: [https://www.jesusda.com/docs/ebooks/introduccion_javascript.pdf].

Demashov, D., & Gosudarev, I. (2019). Efficiency Evaluation of Node. js Web-Server Frameworks. In *MICSECS*

Ferrer Carcasés, A., Sabina Diez, A., Peña Montero, A. M., & Martínez Pérez, M. M. (2016). Interfaz web para la administración y monitorización de la herramienta de réplica de datos SymmetricDS (Bachelor's thesis).

FERREIRA-MEDINA, Heberto, et al. Utilización del modelo de referencia Web para el desarrollo de aplicaciones basadas en un framework; evaluación establecida con indicadores Using the Web reference model for developing applications based on a framework; evaluation based.

Fuentes Castillo, Y (2023). FUNDAMENTACIÓN TEÓRICA DEL PIR PARA EL DESARROLLO DE SOFTWARE

Fuentes, L., & Vallecillo, A. (2004). Una introducción a los perfiles UML. Novática, 168, 6-11.

Gil Jiménez, I., & Fernández Liguori, L. (2021). SQL vs NoSQL: Benchmarking de MySQL y MongoDB.

Gimenez, C., Braun, G., Cecchi, L., & Fillottrani, P. R. (2016, May). Una arquitectura cliente-servidor para modelado conceptual asistido por razonamiento automático. In XVIII Workshop de Investigadores enCiencias de la Computación (WICC 2016, Entre Ríos, Argentina).

González, H., & de la Caridad, D. (2019). Sistema para la gestión de la guardia obrera en la Facultad 1 de la Universidad de las Ciencias Informáticas (Bachelor's thesis, Universidad de las Ciencias Informáticas. Facultad 1.).

González, H., & de la Caridad, D. (2019). Sistema para la gestión de la guardia obrera en la Facultad 1 de la Universidad de las Ciencias Informáticas (Bachelor's thesis, Universidad de las Ciencias Informáticas. Facultad 1.).

Gonzales Castillo, J. V., & Reyes Jaramillo, L. E. (2014). Desarrollo e implementación de un sistema web contable a nivel de prototipo utilizando software libre para la Empresa Comercial AL&CE SRL.

GONZÁLEZ, Bernardo Hernández. Sistema para el control del proceso de impresiones en la Facultad 6 de la Univesidad de las Ciencias Informáticas. Serie Científica de la Universidad de las Ciencias Informáticas, 2015, vol. 8, no 2, p. 1-1.

GUANAJUATO, U. D. Sistema de Control de Bienes Web- UG Universidad de Guanajuato, 2023. [Disponible en: www.siia.ugto.mx/bienes]

Gutierrez, D. (2011). Casos de uso Diagramas de Casos de Uso. Gutierrez, Demián, 1, 45.

HARO, Edward, et al. Desarrollo backend para aplicaciones web, servicios web restful: Node. js vs spring boot. Revista Ibérica de Sistemas e Tecnologias de Informação, 2019, no E17, p. 309-321.

Hernández González, B., Ramírez Ramírez, T., & Mar Cornelio, O. (2019). Sistema para la auditoría y control de los activos fijos tangibles. Revista Universidad y Sociedad, 11(1), 128-134.

Hernández, L. R. B., Peña, D. M., Valdés, O. R., & Cornelio, O. M. (2016). Extensión de la herramienta Visual Paradigm for UML para la evaluación y corrección de Diagramas de Casos de Uso. Serie Científica de la Universidad de las Ciencias Informáticas, 9(7), 7-20.

HERNÁNDEZ GONZÁLEZ, Bernardo; RAMÍREZ RAMÍREZ, Tania; MAR.

Herrero Arnanz, J. (2015). Adelantamientos seguros mediante un sistema de transmisión de vídeo entre vehículos.

Heredia, J. S., & Sailema, G. C. Comparative Analysis for Web Applications Based on REST Services: MEAN Stack and Java EE Stack Análisis comparativo para aplicaciones web basados en servicios REST: stack MEAN y.

Heurtel, O. (2015). Oracle 12c: administración. Ediciones ENI.

Horsford Rivera, J. (2022). Sistema web para la gestión de Información en el Departamento de Tecnología de la Facultad de Ciencias y Tecnologías Computacionales (Bachelor's thesis, Universidad de las Ciencias Informáticas. Facultad 6. Facultad de Ciencias y Tecnologías Computacionales (CITEC))

HUARAC MORALES, Yenssy Milagros. El activo fijo tangible y su depreciación en las empresas comerciales del distrito de Barranca. 2021.

Lizama, O., Kindley, G., Morales, J. J., & Gonzales, A. (2016). Redes de Computadores: Arquitectura Cliente-Servidor. Universidad Tecnica Federico Santa Maria, 1-8.

LIZAMA, Oscar, et al. Redes de Computadores: Arquitectura Cliente-Servidor. Universidad Tecnica Federico Santa Maria, 2016, p. 1-8.

Marin Diaz, A., Trujillo Casañola, Y., & Buedo Hidalgo, D. (2020). Estrategia de pruebas para organizaciones desarrolladoras de software. Revista Cubana de Ciencias Informáticas, 14(3), 83-104.

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

Moya Chaviano, M., Sarmiento Ávila, E. E., Martínez Jera, E., & Nuñez Broch, Y. (2016). Desarrollo de funcionalidades para la conducción de estudios en el Sistema de Gestión de Ensayos Clínicos (Bachelor's thesis).

Molina Hernández, Y., Granda Dihigo, A., & Velázquez Cintra, A. (2019). Los requisitos no funcionales de software. Una estrategia para su desarrollo en el Centro de Informática Médica. Revista Cubana de Ciencias Informáticas, 13(2), 77-90

Martín de las Pueblas Encinas, G. (2019). Definición de Requisitos Funcionales bajo Especificación IEEE para un Sistema de Ingeniería.

ORTEGA, Gilberto Andrés Vargas. Lineamientos para el diseño de aplicaciones web soportados en patrones GRASP. Ciencia e Ingenieria, 2021, vol. 8, no 2, p. e5716304-e5716304.

Ortí, C. B. (2011). Las tecnologías de la información y comunicación (TIC). Univ. Val., Unidad Tecnol. Educ,(951), 1-7.

PACHECO-CASADIEGO, Jorge Manuel. Metodología para elaborar el modelo conceptual de datos. 2017.

Padrón Mojena, Y., & González Díaz, L. D. (2016). Sistema informático para la gestión de la información de los Activos Fijos Tangibles, los Útiles y Herramientas de la residencia de la Universidad de las Ciencias Informáticas (Bachelor's thesis, Universidad de las Ciencias Informáticas. Facultad 4).

PENADÉS GRAMAGE, María Carmen. Diagrama de casos de uso. 2016.

PUCIARELLI, Luciano. Node JS-Vol. 1: Instalación-Arquitectura-node y npm. RedUsers, 2020.

Portal del Consejo nacional de patrimonio cultural—Inicio. (s. f.). Recuperado 30 de abril de 2023, de http://www.cnpc.cult.cu/

(Portal del Consejo nacional de patrimonio cultural - Inicio, s. f.).

Pérez Ibarra, S. G., Quispe, J. R., Mullicundo, F. F., & Lamas, D. A. (2021). Herramientas y tecnologías para el desarrollo web desde el FrontEnd al BackEnd. In XXIII Workshop de Investigadores en Ciencias de la Computación (WICC 2021, Chilecito, La Rioja).

Piza Burgos, N. D., Amaiquema Márquez, F. A., & Beltrán Baquerizo, G. E. (2019). Métodos y técnicas en la investigación cualitativa. Algunas precisiones necesarias. Conrado, 15(70), 455-459.

Pressman, R. S. (2002). Ingeniería delsoftware. Unenfoque práctico.(5: edición).

Ramírez Cevallos, W. R. (2018). Estudio del Framework Meteor para aplicaciones web. desarrollo del sistema de planificación de proyectos para el GAD parroquial La Esperanza (Bachelor's thesis).

Rodríguez Flores, G. (2017). Desarrollo de una aplicación web con Node.js para la monitorización en tiempo real de un electrocardiograma. (Trabajo Fin de Grado Inédito). Universidad de Sevilla, Sevilla.

Romero Iglesias, D. R. (2016). GESTIÓN DEL ACTIVO FIJO EN LA UNIVERSIDAD CIENTIFICA DEL SUR 2016.

RIVERA ALVA, E. 2017. Arquitectura-de-Software-II-Diagrama-de-Componentes-y-Despliegue. 2017.

Robayo-Bautista, E. C. (2021). Guía de principios y buenas prácticas para pruebas de seguridad de software en aplicaciones web para una empresa del sector privado.

Sanz, M. L., Mesa, J. M. V., Domingo, F. J. S., & Pérez, Á. M. (2016). MF0492_3 Programación Web en el Entorno Servidor. Ra-Ma Editorial.

Sánchez, I. R. A. (2021). La Sociedad de la Información, Sociedad del Conocimiento y Sociedad del Aprendizaje. Referentes en torno a su formación. *Bibliotecas. Anales de investigación*, *12*(2)

Sommerville, I. (2011). Ingeniería de Requerimientos. Ingeniería de Software, 82-88.

Sommerville, Ian. 2011. INGENIERIA DEL SOFTWARE. UN ENFOQUE PRACTICO. 2011.

Solorza, F., Braun, G., Cecchi, L., & Fillottrani, P. R. (2018). Hacia la Formalizacion de un Lenguaje Visual Unificador de UML, EER y ORM 2. In XX Workshop de Investigadores en Ciencias de la Computación (WICC 2018, Universidad Nacional del Nordeste).

Septién, L. A. D., Collazo, Y. N., & Toledo, R. Y. (2017). Sistema informático para el apoyo a la gestión estratégica en la Universidad de Ciego de Ávila. Universidad & ciencia, 6, 292-307.

Torres, F. G. L., Quintanilla, D. C., & Andrade, J. E. O. (2020). Control y contabilización de activos fijos y su incidencia en la toma de decisiones administrativas. Revista Arbitrada Interdisciplinaria Koinonía, 5(4), 443-472.

Universitas XXI - Económico | Vicegerencia económica, s. f.).

Universidad de Murcia. (2016). Obtenido de https://www.um.es.

Universitas XXI - Económico | Vicegerencia económica. (s. f.). Recuperado 30 de abril de 2023, de https://vgeconomica.unizar.es/es/secciones/universitas-xxi-economico.

VISUALPARADIGM. [En línea] [Citado el: 29 junio 2021]. Disponible en: [https://www.visual-paradigm.com-aboutus/newsreleases/].

VISUALStudioCod. [En línea] enero de 2020. [Citado el: 17 de febrero de 2020.] Disponible en: [https://code.visualstudio.com/updates/v1_42].

Vera, Y. P., Valdivia, J. J. G., Quentasi, S. M. Z., Yana, D. M. C., & Apaza, R. E. C. (2020). Design

Thinking en la Planificación de Pruebas de Software. Innovación y Software, 1(2).

Vele Zhingri, C. A. (2016). Análisis de rendimiento entre la base de datos relacional: MySQL y una base de datos no relacional: MongoDB (Bachelor's thesis, Universidad del Azuay).

Wallis, G. (2017). CSS3 For Newbies.

Anexo 1

Entrevista realizada a los especialistas de Radio Reloj para conocer cómo funciona el centro

1. ¿Cómo maneja la radio la privacidad y seguridad de la información almacenada en sus activos fijos tangibles?
2. ¿Cuántos activos tienen actualmente en su radio y cuáles son estos activos?
3. ¿Con qué frecuencia se hacen mantenimientos a los activos y cómo se lleva un registro de estos procesos?
4. ¿Cómo controla actualmente la radio los activos fijos tangibles?

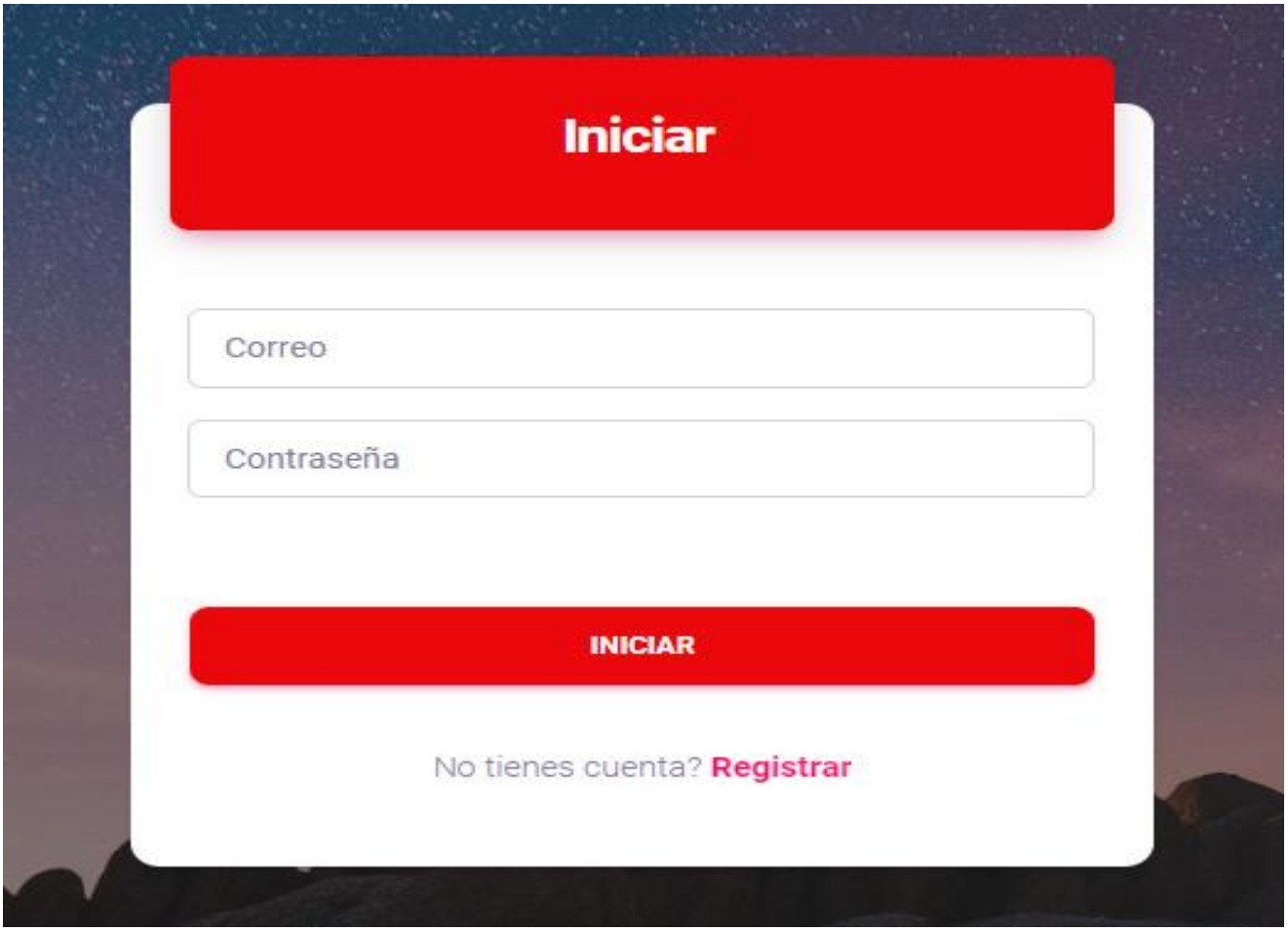
Anexo 2

Entrevista realizada a los especialistas de Radio Reloj para conocer que contenido y funcionalidades desean ver en el Sistema de Gestión de Activos.

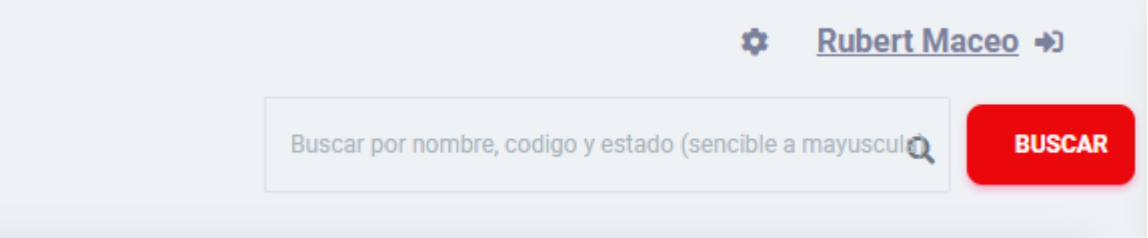
1. ¿Hay algún miembro del personal asignado específicamente para el mantenimiento y la gestión de los activos fijos tangibles?
2. ¿Qué papel podrían tener los trabajadores en una implementación exitosa de un sistema de gestión de activos fijos tangibles?
3. ¿Qué tan a menudo realiza inventarios de sus activos fijos tangibles?
4. ¿Cree que un sistema de gestión de activos fijos tangibles sería beneficioso para la radio? Si es así, ¿qué características le gustaría ver en un sistema de este tipo?
5. ¿Cómo registran actualmente los movimientos de activos dentro de la empresa? (por ejemplo, traslados, cambios de responsable, mantenimiento?.
6. ¿Cómo actualmente se realiza el seguimiento y gestión de los activos fijos tangibles de la estación de radio?.

Anexo 3

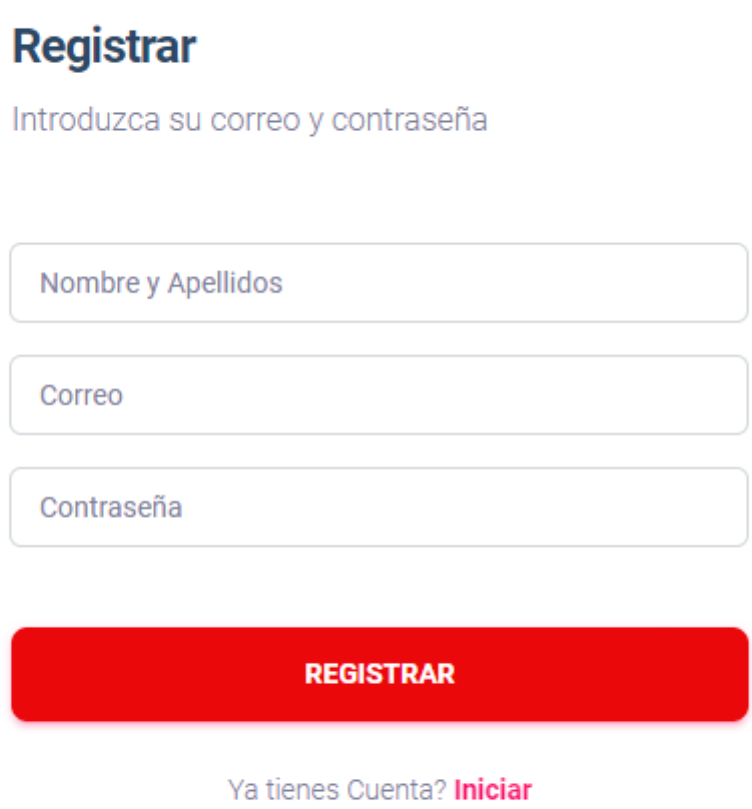
Interfaz: Autenticar



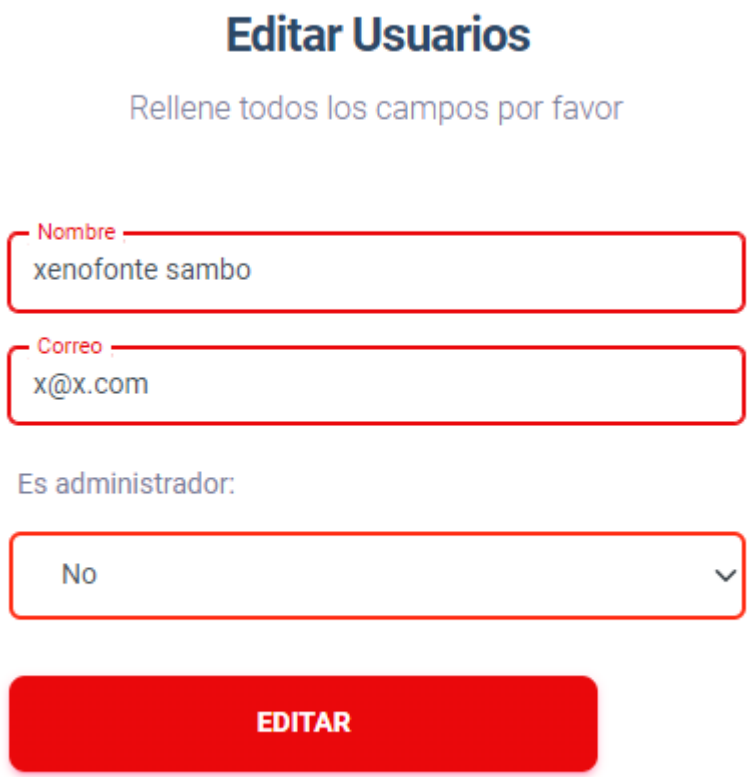
Interfaz: Buscar Medios



Interfaz: Registrar Usuario

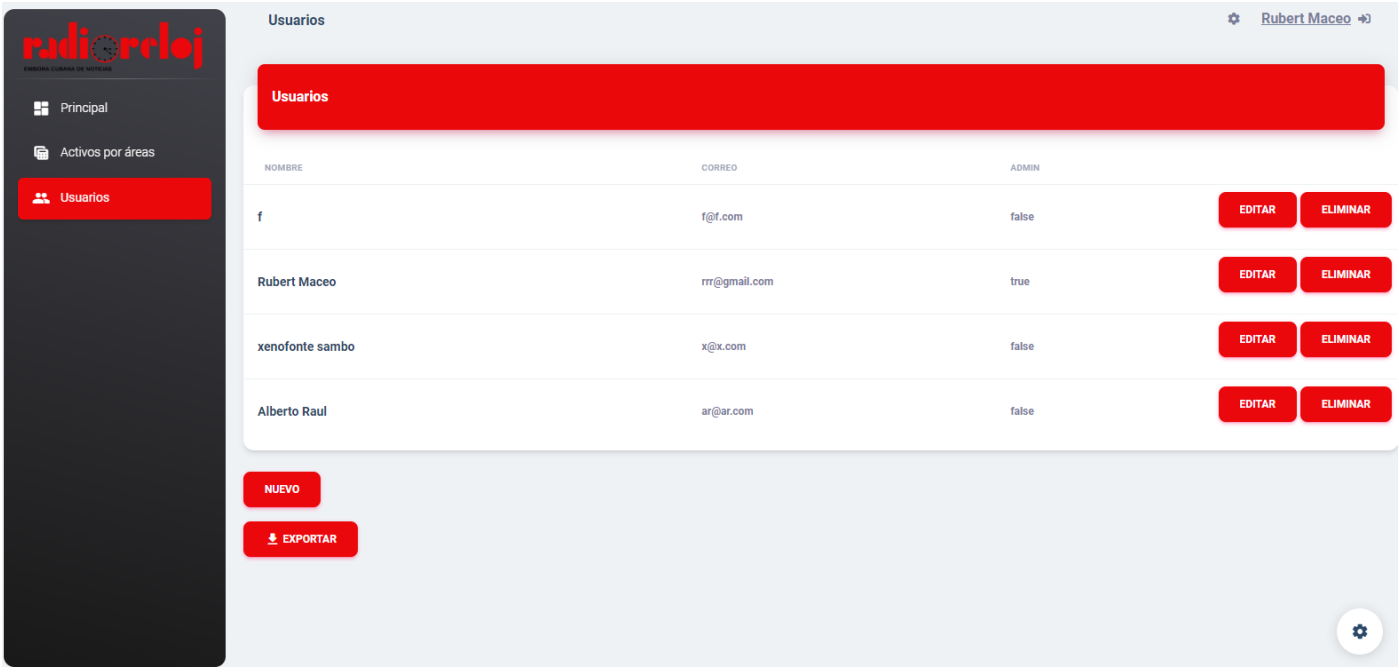


Interfaz:Modificar Usuarios



Interfaz: Listar Activos

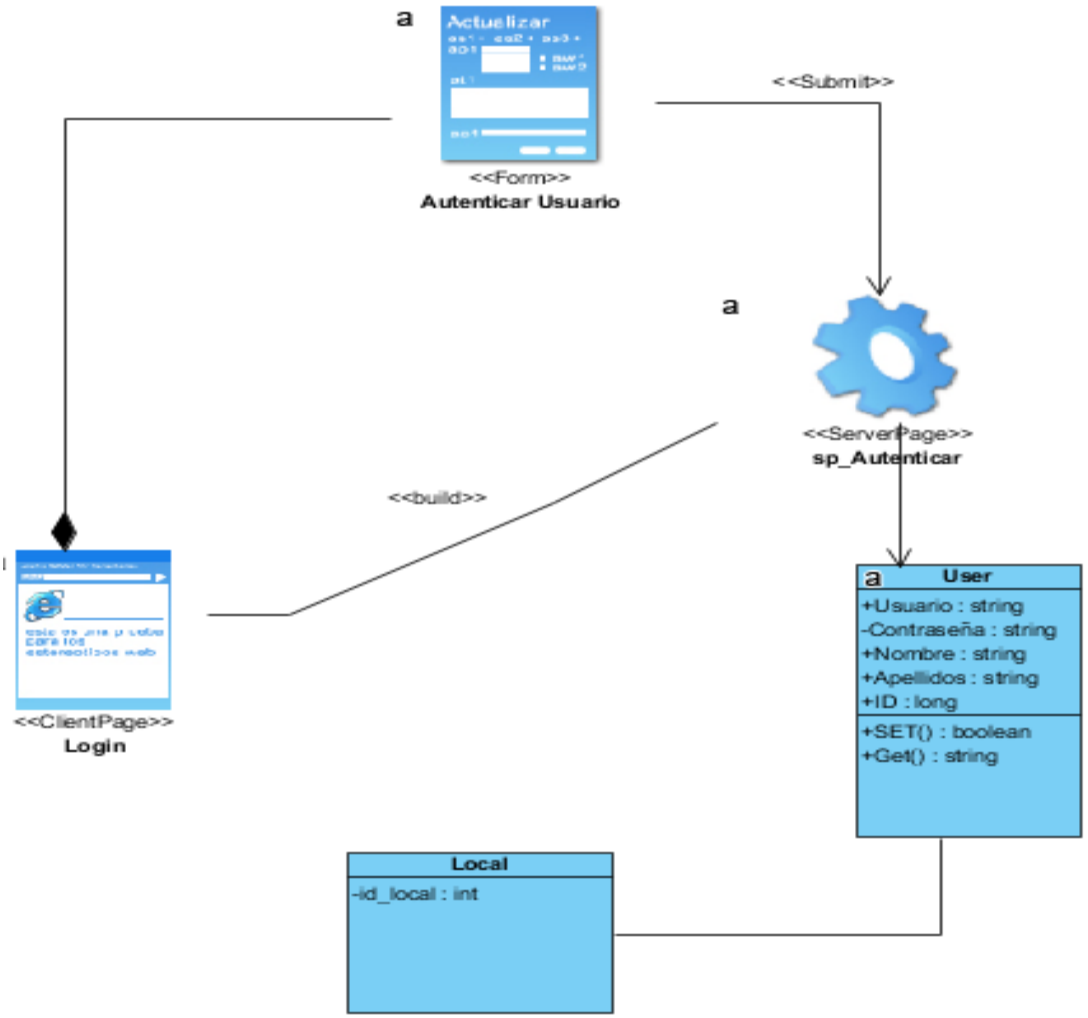
Interfaz: Usuarios



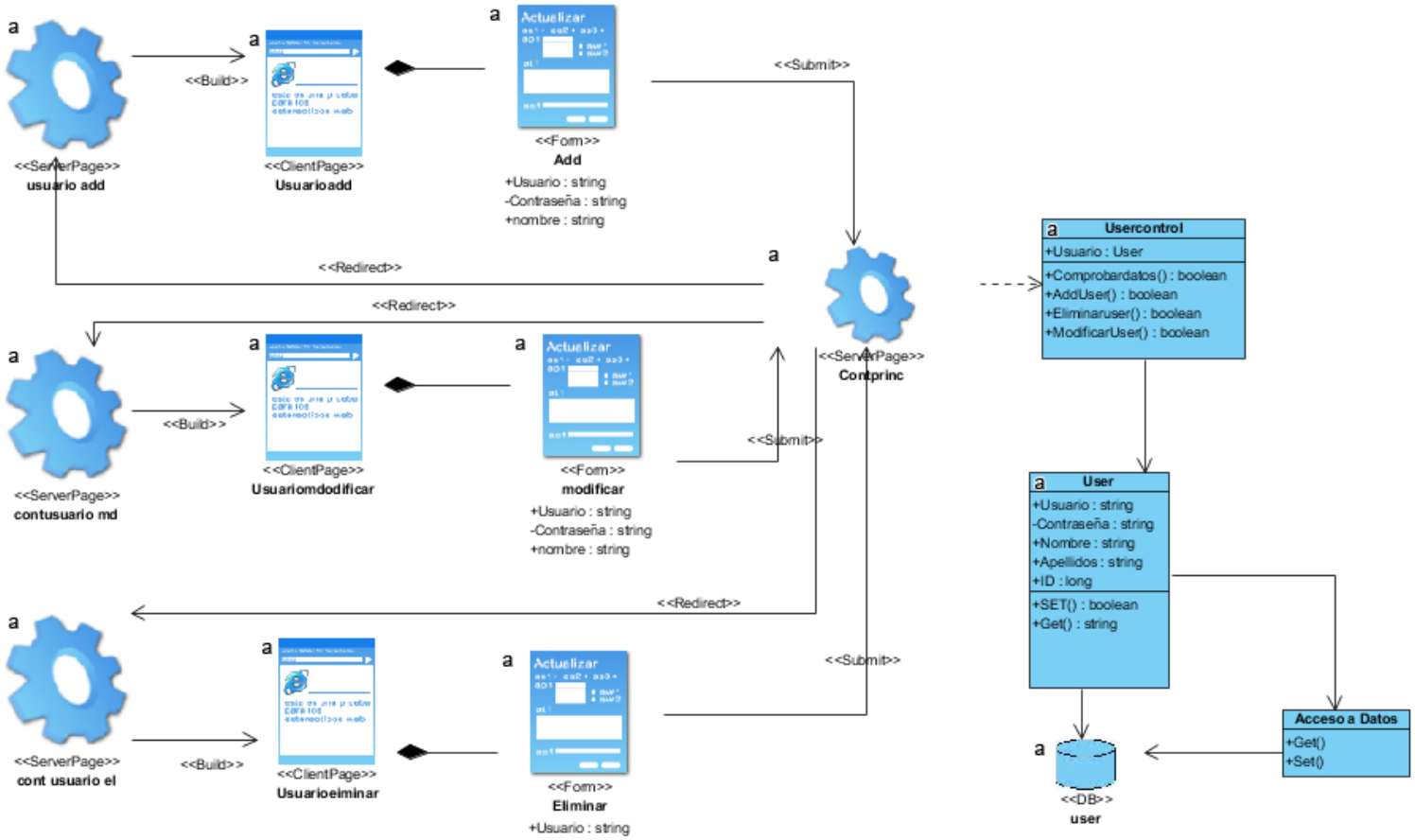
Anexo 4

Diagramas de Clases del Diseño

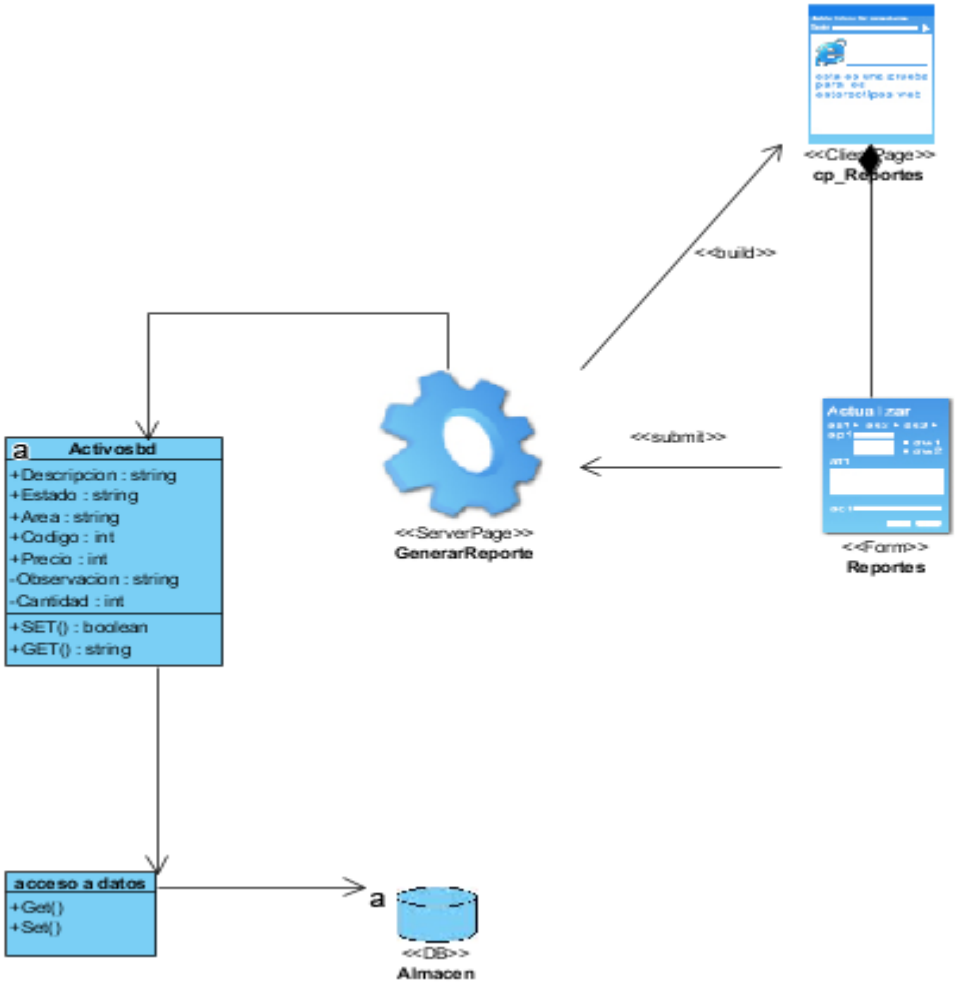
CU: Autenticar



CU: Gestionar Usuarios



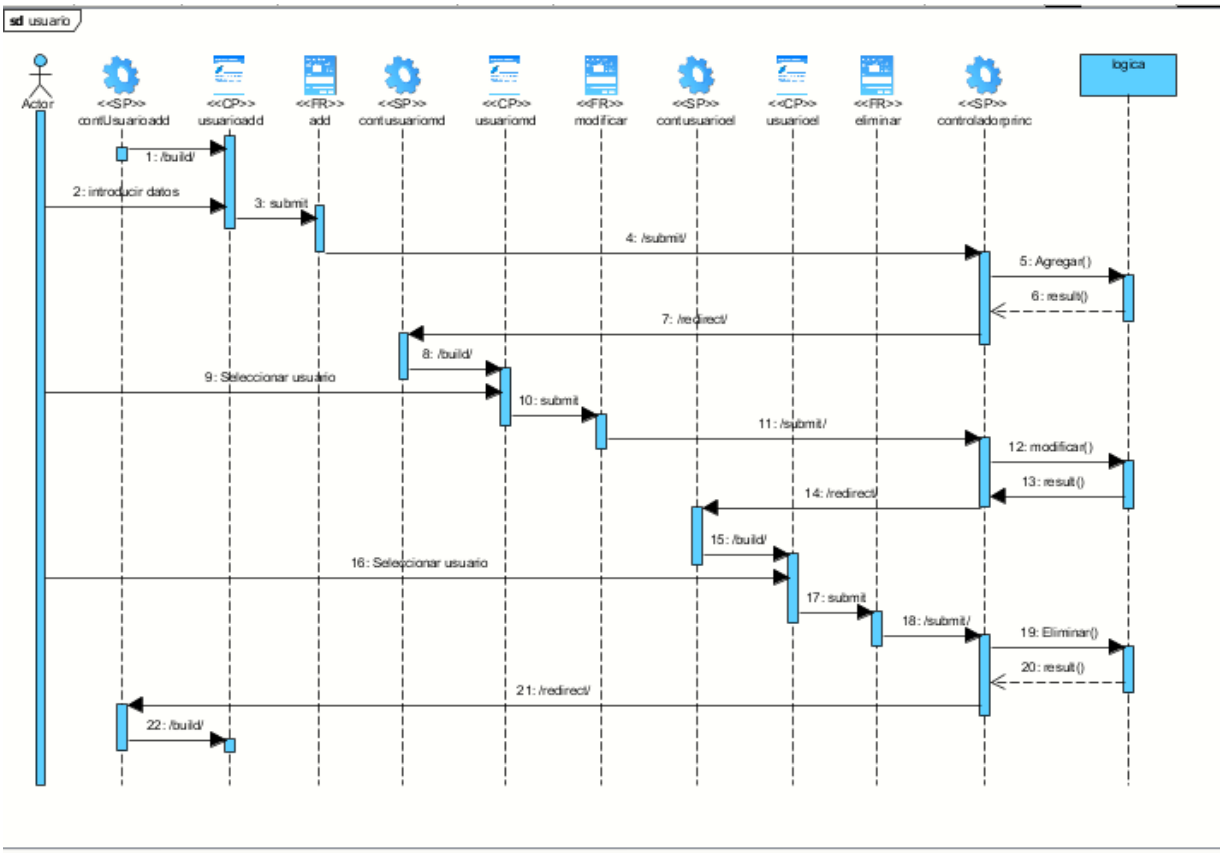
CU: Generar Reportes



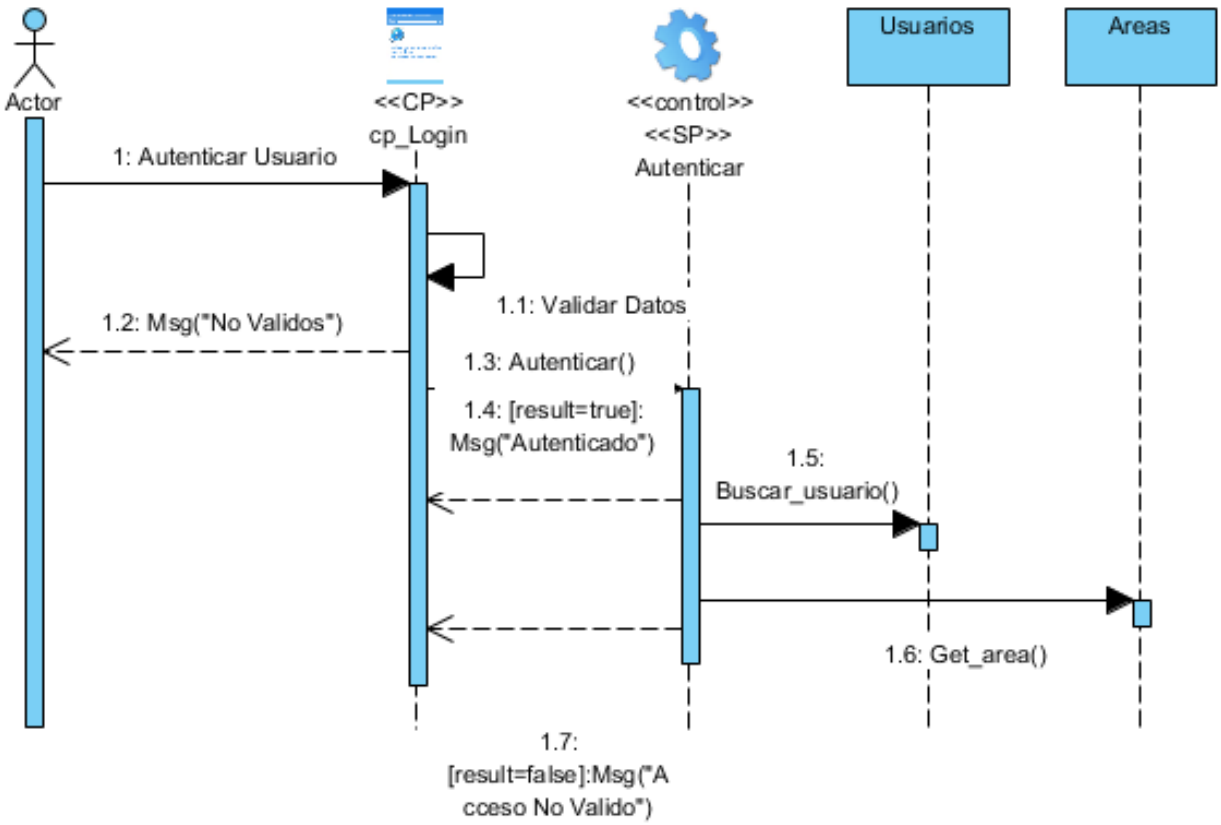
Anexo 5

Diagramas de Secuencia de los Casos de Uso

CU: Gestionar Usarios



CU: Autenticar



CU: Generar Reportes