

Universidad de Cienfuegos “Carlos Rafael Rodríguez”

Facultad de Ingeniería

Carrera Ingeniería Informática



**Microservicio para la gestión de la información en el ASSETS
de la Universidad de Cienfuegos**

**Trabajo de diploma para optar por el título de Ingeniero
Informático**

Autor:

Yohans Alejandro Alfonso Linares

Tutor:

MSc. Manuel Cortés Iglesias. (DTE) UCF

Cienfuegos, Cuba

Curso 2022

Declaración de autoría

Declaro que soy el único autor de este trabajo y autorizo al Departamento de Tecnología Educativa e Informatización de la Universidad de Cienfuegos “Carlos Rafael Rodríguez” y al Departamento de Informática de la Facultad de Ingeniería, para que hagan el uso que estimen pertinente con el trabajo de diploma.

Para que así conste firmo la presente a los ____ días del mes de ____ del ____.

Yohans Alejandro Alfonso Linares

Nombre completo del primer autor

Manuel Cortés Iglesias

Nombre completo del primer tutor

Los abajo firmantes certificamos que el presente trabajo ha sido revisado según acuerdo de la dirección de nuestro centro y el mismo cumple los requisitos que debe tener un trabajo de esta envergadura referente a la temática señalada.

Firma Tutor

Firma ICT

Firma Vicedecano

Opinión del usuario

El Trabajo de Diploma, titulado “Microservicio para la gestión de la información en el ASSETS de la Universidad de Cienfuegos”, fue realizado en nuestra entidad Universidad de Cienfuegos “Carlos Rafael Rodríguez” en el Departamento de Tecnología Educativa e Informatización. Se considera que, en correspondencia con los objetivos trazados, el trabajo realizado nos satisface:

☐ Totalmente

☐ Parcialmente en un ____ %

Los resultados de este Trabajo de Diploma le reportan a nuestra entidad los beneficios siguientes:

Como resultado de la implantación de este trabajo se reporta un efecto económico que asciende a <valor> MN y/o <valor> CUC. (Este valor debe ser REAL, no indica lo que se reportará, sino lo que reporta a la entidad. Puede desglosarse por conceptos, tales como: cuánto cuesta un software análogo en el mercado internacional, valor de los materiales que se ahorran por la existencia del software, valor anual del (de los) salario(s) equivalente al tiempo que se ahorra por la existencia del software).

Y para que así conste, se firma la presente a los ____ días del mes de ____ del año ____.

Nombre del representante de la entidad

Cargo

Firma

Cuño

Opinión del tutor

Título: <Título del trabajo de diploma>

Autor(es): <Nombres y apellidos del autor o los autores>

El(Los) tutor(es) del presente Trabajo de Diploma considera(mos) que durante su ejecución el(los) estudiante(s) mostró(aron) las cualidades que a continuación se detallan.

<El tutor debe expresar cualitativamente su opinión y medir (usando la escala: muy alta, alta, adecuada) entre otras las cualidades siguientes: Independencia, Originalidad, Creatividad, Laboriosidad y Responsabilidad>

<Además, debe evaluar la calidad científico-técnica del trabajo realizado (resultados y documento) y expresar su opinión sobre el valor de los resultados obtenidos (aplicación y beneficios)>.

Por todo lo anteriormente expresado considero que el estudiante está (no) apto para ejercer como Ingeniero Informático; y propongo que se le otorgue al Trabajo de Diploma la calificación de <2 – Desaprobado, 3 – Aprobado, 4 – Bien, 5 – Excelente>.

<Si considera que los resultados poseen valor para ser publicados, debe expresarlo también>

Y para que así conste, se firma la presente a los ____ días del mes de _____ del año ____.

(Si procede)

Nombre completo del primer tutor

Nombre completo del segundo tutor

<Grado científico, Categoría docente
y/o investigativa>

<Grado científico, Categoría docente
y/o investigativa>

Fecha: _____

Agradecimientos

Al único Dios verdadero del cual; son todas las cosas, por Él y para Él; que en su soberanía me permitió dedicar estos años de mi vida a aprender un poco de este mundo que es la informática; a Él debo todo.

A mi familia por ayudarme en esta etapa y estar presente en mis días desde el principio, a mi pareja por tenerme paciencia mientras terminaba la tesis y por darme su compañía y apoyo

Agradezco a mis padres, hermanos, abuelos, amigos, tutores y profesores, no esperaré ser grande o estar lejos para agradecer aquellos quienes fueron motor impulsor para llegar hasta el final, a los que decidieron humildemente aportar sus conocimientos para que yo adquiriera los míos.

“La gratitud, como ciertas flores, no se da en la altura y mejor reverdece en la tierra buena de los humildes”

José Martí.

Dedicatoria

A todos los que hacen que mis días sean mejores y me motivan a ser diferente, a mi Padre Eterno, familia, pareja y amigos, a ellos va dedicado este logro.

Resumen

La presente investigación, desarrollada en la Universidad de Cienfuegos (UCf), lleva como título “Microservicio para la gestión de la información en el ASSETS de la Universidad de Cienfuegos”. Surge como respuesta a la falta de integración de sistemas existentes en la institución, como es el caso del ASSETS; lo cual hace difícil poder recuperar información y realizar tareas conjuntas entre varios sistemas. Para esto se desarrolla un sistema con arquitectura basada en microservicios para integrar la información generada por el ASSETS permitiendo el uso de esta por otras aplicaciones que se implementen en el futuro. A través del microservicio del ASSETS, los empleados del centro podrán acceder a su información desde otras interfaces. Para su desarrollo, se utiliza la metodología de desarrollo de software, Proceso Unificado de Rational (RUP), Visual Paradigm Enterprise Edition v.8.0 como herramienta de modelado, Javascript como lenguaje de programación, Node como entorno de ejecución de Javascript en el servidor, como framework web Express, así como Visual Studio Code como editor de código, Quasar como framework para crear la interfaz visual y SQLite como gestor de base de datos para la administración microservicio.

Palabras clave: sistema, microservicios, ASSETS, Node, Javascript.

Summary

The present investigation, developed at the University of Cienfuegos (UCf), is entitled "Microservice for information management in the ASSETS of the University of Cienfuegos". It arises as a response to the lack of integration of existing systems in the institution, as is the case of ASSETS; which makes it difficult to retrieve information and perform joint tasks between various systems. For this, a system with architecture based on microservices is developed to integrate the information generated by the ASSETS, allowing its use by other applications that are implemented in the future. Through the ASSETS microservice, the center's employees will be able to access their information from other interfaces. For its development, the Rational Unified Process (RUP) software development methodology is used, Visual Paradigm Enterprise Edition v.8.0 is used as a modeling tool, Javascript as a programming language, Node as a Javascript execution environment on the server, as a web framework Express, as well as Visual Studio Code as a code editor, Quasar as a framework to create the visual interface and SQLite as a database manager for the microservice administration.

Keywords: system, microservices, ASSETS, Node, Javascript.

Índice

Introducción.....	1
1 – Capítulo I: Fundamentos teóricos de los sistemas de gestión integral y la arquitectura de microservicios	9
1.1 – Introducción	9
1.2 – Descripción del dominio del problema.....	9
1.2.1 Gestión Empresarial.....	9
1.2.2 La Gestión de la Información en la Gestión Empresarial y los Sistemas de Información	10
1.2.3 ASSETS	11
1.2.4 Integración entre sistemas.....	12
1.3 – Descripción del objeto de estudio	13
1.3.1 - Objetivos estratégicos de la organización	13
1.3.2 - Flujo actual de los procesos y análisis crítico de la ejecución de estos.....	14
1.4 – Descripción de los sistemas existentes	15
1.4.1 Amazon Web Services.....	15
1.4.2 Google Analytics	16
1.4.3 Servicios Web para la integración de la información de los sistemas informáticos SIGENU y ASSETS en la Universidad de Cienfuegos (UCf).....	17
1.5 – Tendencias, metodologías y/o tecnologías actuales	18
1.5.1 Metodologías de desarrollo de software	18
1.5.2 Tendencias en la integración de sistemas	19
1.5.3 Lenguajes y Herramientas Utilizadas	24
1.6 – Conclusiones.....	31
2 – Capítulo II: Análisis y diseño del microservicio para la gestión de la información en el ASSETS de la Universidad de Cienfuegos	32
2.1 – Introducción	32
2.2 – Modelo del dominio.....	32
2.2.1 – Definición de las entidades y los conceptos principales.....	32
2.2.2 – Reglas del negocio a considerar.....	34
2.2.3 – Representación del modelo del dominio.....	35
2.3 – Requisitos.....	35
2.3.1 – Descripción del sistema propuesto	35
2.3.1.1 – Concepción general del sistema.....	35
2.3.1.2 – Requerimientos funcionales.....	36
2.3.1.3 – Requerimientos no funcionales.....	37
2.3.2 – Modelo de casos de uso del sistema	40
2.3.2.1 – Actores del sistema	40
2.3.2.2 – Paquetes y sus relaciones.....	41
2.3.2.3 – Diagramas de casos de uso del sistema.....	41
2.3.2.4 – Descripción de los casos de uso del sistema.....	42
2.4 – Construcción de la solución propuesta	49
2.4.1 – Diagrama de clases del diseño	49
2.4.2 – Diseño de la base de datos	50

2.4.2.1 – Modelo lógico de datos	50
2.4.2.2 – Modelo físico de datos	51
2.4.3 – Diagrama de implementación	53
2.4.4 – Principios de diseño	53
2.4.4.1 – Estándares en la interfaz de la aplicación	54
2.4.4.2 – Tratamiento de errores	54
2.4.4.3 – Concepción General de la ayuda.....	54
2.5 – Conclusiones	54
3 – Estudio de factibilidad y validación de la solución	56
3.1 – Introducción	56
3.2 – Estudio de factibilidad	56
3.2.1 – Planificación por casos de usos.....	56
3.2.2 – Determinación de los costos	65
3.2.3– Beneficios tangibles e intangibles.....	66
3.2.4 – Análisis de costos y beneficios	67
3.3 – Validación de la solución.....	67
3.3.1 Diseño de Pruebas Funcionales	68
3.4 – Conclusiones	78
Conclusiones	80
Recomendaciones	81
Referencias bibliográficas	82
Glosario de términos	87
Anexos.....	88
Anexo 1 – Metodología RUP	88
Anexo 1.1 – Fases de RUP	88
Anexo 2 – Prototipos de interfaces	89
Anexo 2.1 – Prototipo de la Interfaz del caso de uso Autenticar.....	89
Anexo 2.2 – Prototipo de la Interfaz del caso de uso Cerrar Sección.....	90
Anexo 2.3 – Prototipo de la Interfaz del caso de uso Gestionar Usuarios.....	90
Anexo 2.4 – Prototipo de la Interfaz del caso de uso Mostrar todos los trabajadores	96
Anexo 2.5 – Prototipo de la Interfaz del caso de uso Buscar Información de un Trabajador	96
Anexo 2.6 – Prototipo de la Interfaz del caso de uso Filtrar Información de los Trabajadores	97
Anexo 3 – Prototipos de interfaces	98
Anexo 3.1 – Diagrama de clases del diseño CU <Autenticar>.....	98
Anexo 3.2 – Diagrama de clases del diseño CU <Cerrar Sección>.....	98
Anexo 3.3 – Diagrama de clases del diseño CU <Gestionar Usuarios>.....	99
Anexo 3.4 – Diagrama de clases del diseño CU <Mostrar todos los trabajadores>	99
Anexo 3.5 – Diagrama de clases del diseño CU <Buscar Información de un trabajador>... ..	100
Anexo 3.6 – Diagrama de clases del diseño CU <Filtrar información de los trabajadores>	100

Índice de tablas

Tabla 1. Descripción de los actores del sistema	40
Tabla 2. Descripción del caso de uso de sistema <Autenticar>	43
Tabla 3. Descripción del caso de uso de sistema <Cerrar Sección>	44
Tabla 4. Descripción del caso de uso de sistema <Gestionar Usuarios>	45
Tabla 5. Descripción del caso de uso de sistema <Mostrar todos los trabajadores>.....	46
Tabla 6. Descripción del caso de uso de sistema <Buscar Información de un Trabajador>.....	47
Tabla 7. Descripción del caso de uso de sistema <Filtrar Información de los Trabajadores>	49
Tabla 8. Diagrama de Clases del Diseño	50
Tabla 9. Factor de peso de los actores del sistema.....	57
Tabla 10. Factor de peso de los Casos de Uso.....	58
Tabla 11. Factor de Complejidad Técnica.....	61
Tabla 12. Factor Ambiente	63
Tabla 13. Estimación de Esfuerzo	65

Índice de figuras

Figura 1. Diagramas de clases del modelo de objetos del dominio	35
Figura 2. Diagramas de paquetes.....	41
Figura 3. Diagramas de casos de uso del sistema	42
Figura 4. Diagramas de clases persistentes de la parte de la base de datos del ASSETS utilizada	51
Figura 5. Diagramas de clases persistentes de la base de datos del microservicio.....	51
Figura 6. Diagramas del modelo físico de datos de la parte de la base de datos del ASSETS utilizada	52
Figura 7. Diagramas del modelo físico de datos de la base de datos del microservicio.....	53
Figura 8. Diagrama de implementación	53
Figura 9. Caso de Prueba <<Autenticar>> mensaje de alerta en el campo correo.....	69
Figura 10. Caso de Prueba <<Autenticar>> mensaje de error	70
Figura 11. Caso de prueba <<Registrar Usuario>> Vista de Agregar Usuarios	72
Figura 12. Caso de prueba <<Registrar Usuario>> mensaje de error	72
Figura 13. Caso de Prueba << Modificar Estado de la Cuenta >> Modal	73
Figura 14. Caso de Prueba << Modificar Estado de la Cuenta >> Mensaje de error.....	74
Figura 15. Caso de Prueba << Modificar Api Key >> Modal	75
Figura 16. Caso de Prueba << Modificar Api Key >> Mensaje de error	76
Figura 17. Caso de Prueba << Eliminar Usuario >> Modal	77
Figura 18. Caso de Prueba << Eliminar Usuario >> Mensaje de error	78

Introducción

La información juega un papel fundamental en la evolución de la humanidad; gracias al desarrollo tecno-científico que ha traído consigo ha transformado la sociedad, la cultura y la ciencia [1], desde la antigüedad se ha visto como signo de poder y de progreso. Las sociedades avanzadas se distinguen de las demás en que disponen de más información accesible a un mayor número de personas mediante medios muy diversos. La información es capaz de cambiar la forma en que el receptor percibe algo, es capaz de impactar sobre sus juicios de valor y comportamientos [2]. El diccionario de la lengua española provee múltiples definiciones para el vocablo, como: Acción y efecto de informar, comunicación o adquisición de conocimientos que permiten ampliar o precisar los que se poseen sobre una materia determinada y conocimientos comunicados o adquiridos mediante una información.[3]

El crecimiento de las sociedades de la información y del conocimiento, apoyadas en el aumento de las Tecnologías de la Información y la Comunicación; ha dado como resultado una explosión de información sin precedentes [4]; se estima que cada cuatro años se duplica la información existente en el mundo y que en el último siglo se ha producido más información que en toda la historia anterior de la humanidad [5]. Este desarrollo exponencial ha llevado a que en cualquier organización donde la información sea la base para la toma de decisiones apoyadas por las tecnologías de información se hable de cultura de la información.[6]

La importancia que tiene la información y el inmenso cúmulo que se genera a diario nos lleva a buscar formas de gestionarla, lo que abre paso a la gestión de la información. Una de las definiciones más completas dada por Woodman (1985) habla de la gestión de la información como: todo lo relacionado con la obtención de la información adecuada, en la forma correcta, para la persona indicada, al costo conveniente, en el tiempo oportuno y en el lugar apropiado para tomar la acción correcta [7]. Según Manso Rodríguez (2008), la gestión de la información es “el conjunto de las actividades que se realizan con el propósito de adquirir, procesar, almacenar y finalmente recuperar, de manera adecuada, la información que se produce o se recibe en una organización y que permite el desarrollo

de su actividad”. En el contexto empresarial constituye un recurso determinante que puede llevar al éxito de una organización siempre y cuando se gestione de manera eficiente.[8]

La Gestión garantiza un óptimo uso y manejo de la información y contribuye a la mejora del desempeño organizacional. Su propósito está asociado a disminuir los riesgos organizacionales y aprovechar oportunidades [9], en este sentido, las teorías de la información y el conocimiento constituyen un factor de importancia para alcanzar su eficiencia[10]. En las organizaciones actuales son numerosos los datos que se ignoran o no adquieren valor por la falta de procesos y mecanismos eficaces o eficientes de gestión de información. Un ejemplo de ello son las Instituciones de Educación Superior, las cuales deben responder a procesos asociados a sus funciones de docencia, investigación y proyección social, que les exigen una pronta respuesta a diferentes solicitudes de información, como indicadores de desempeño [11].

Por ello, las autoridades académicas necesitan acceder a información útil, que les permita monitorear el progreso de los procesos institucionales con datos precisos, confiables y oportunos, de tal manera que puedan prevenir y/o predecir acontecimientos negativos en el quehacer académico [12]. Resulta imprescindible que estas gestionen el modo en que generan, manipulan y transmiten información, convirtiéndose esto en conjunto con el uso de las Tecnologías de Información y Comunicación (TIC) en la razón de ser de los sistemas de gestión de la información [13] las cuales ofrecen la posibilidad de agilizar los procesos, obtener información veraz, oportuna y confiable. [14]

Según Andreu, Ricart y Valor (1991), un sistema de información queda definido como: “conjunto formal de procesos que, operando sobre una colección de datos estructurada de acuerdo a las necesidades de la empresa, recopila, elabora y distribuyen selectivamente la información necesaria para la operación de dicha empresa y para las actividades de dirección y control correspondientes, apoyando, al menos en parte, los procesos de toma de decisiones necesarios para desempeñar funciones de negocio de la empresa de acuerdo con su estrategia” [15]. Los sistemas de información son uno de los componentes más relevantes del entorno empresarial actual. Ofrecen grandes oportunidades de éxito para las empresas debido a su capacidad de recopilar, procesar,

distribuir y compartir datos de manera integrada y oportuna, lo que se refleja en una mejora de los procesos, la administración y el manejo de la información, repercutiendo positivamente en la productividad y competitividad de las empresas permitiendo a las instituciones mejorar su eficiencia. [16]

El uso de estos sistemas a nivel global es vertiginoso; globalización, mayor calidad, reputación internacional, rendición de cuentas y transformación digital son algunos de los desafíos que enfrentan las universidades en Latinoamérica. Los Sistemas de Información juegan un papel importante para alcanzar estos objetivos, debido a que aseguran la administración de los procesos académicos, de gestión e investigación. Estudios han demostrado que la incorporación de las universidades de estas tecnologías ha traído consigo un aumento en su productividad y en los rankings internacionales. [17]

En el contexto chileno, se tiene el caso de la Universidad del Bío-Bío (UBB), la institución desarrolló a partir del año 2011 un sistema que permitió integrar diversas perspectivas e instrumentos de la gestión universitaria, denominado Sistema de Información de Gestión Estratégica (SIGEUBB). La verificación del cumplimiento de los propósitos y metas institucionales, se desarrollan por medio de este sistema.[18]

En nuestro país debido al llamado a la informatización de la sociedad, cobra aún más importancia la implementación de estos sistemas. La dirección del gobierno y del Partido Comunista de Cuba (PCC), mediante la aprobación de la política de lineamientos aprobados en VI Congreso del PCC, han priorizado determinados sectores, como es el caso de la educación y la salud, que por sus particularidades y potencialidades, deben convertirse en los pilares en que descansa la estrategia de supervivencia y desarrollo de la nación. Dichos lineamientos reconocen la importancia del perfeccionamiento de los sistemas de información y se vislumbran como un desafío para los directivos en aras de tomar decisiones y lograr efectividad en la gestión. [19]

En el Centro de Informática Médica (CESIM) se desarrolla el Sistema de Información Hospitalaria XAVIA HIS. El sistema está compuesto por módulos que aseguran la informatización de los procesos de las áreas de la institución hospitalaria. Los módulos interconectan las diferentes áreas de una institución hospitalaria como son Admisión,

Emergencias, Epidemiología, Banco de Sangre, Consulta Externa, Hospitalización, entre otros. [20]

La industria biotecnológica emplea sistemas de gestión de ensayos clínicos para lograr un manejo eficiente de los datos recopilados durante el proceso. El Centro de Inmunología Molecular (CIM) utiliza el Sistema de Gestión de Ensayos Clínicos XAVIA Clínicas, desarrollado por la Universidad de Ciencias Informáticas (UCI), para soportar el 100% de los ensayos clínicos desde el 2012. A la par desarrolla el Sistema para el manejo de datos de Ensayos Clínicos XAVIA SIDEC, este facilita y flexibiliza el diseño y aprobación de los Cuadernos de recogida de información (CRD), estandariza la información manejada, valida los datos recopilados a través de reglas, mejora el flujo de información entre las entidades que conducen el ensayo y optimiza el proceso de monitoreo. [21]

La provincia de Pinar del Río se ha sumado a este inminente desarrollo informático en el sector de la salud, pues tres de sus principales hospitales trabajan con el sistema informático Galen Clínica. Como parte de las investigaciones que se desarrollan en la maestría Informática en Salud, se ejecutan un conjunto de proyectos de informatización que complementarán al Galen Clínica. Uno de estos proyectos se desarrolla en el Hospital Clínico Quirúrgico Docente “León Cuervo Rubio” que cuenta con servicios de Medicina General y Cirugía y tiene accionar casi absoluto en especialidades como Geriátrica, Hematología, Urología, Medicina Natural y Tradicional Hospitalaria, Maxilofacial, Toxicología Clínica, Reumatología, Fisioterapia y Rehabilitación con camas para hospitalización y Medicina Legal. [22]

Existen muchos sistemas para la automatización de los procesos relacionados con las actividades docentes en el mercado, sin embargo, implican una inversión que, algunas instituciones, no están en condiciones de enfrentar. El Sistema de Gestión de la Nueva Universidad (SIGENU), es un ejemplo de sistema nacional diseñado para satisfacer necesidades específicas de la educación superior en Cuba [23]. Este se concibe de manera que brinda gran seguridad e integridad de la información, y a la vez, es tan flexible que permita ser adaptado a todos los centros de educación superior del país con sus diversas particularidades y distintas maneras de realizar determinados procedimientos y

[24] posibilita la gestión de los datos de los estudiantes en secretaría del Curso Regular Diurno (CRD) y Curso por Encuentros (CPE). [25]

La Universidad de Cienfuegos (UCf) cuenta con varios sistemas de gestión para el cumplimiento de sus actividades diarias. Algunos de estos como el SIGENU se encarga de gestionar la información de los estudiantes de la universidad. Otro sistema, que se emplea para cuestiones de gestión institucional, es el ASSETS, el sistema de gestión que ocupa el centro de atención en este documento.

El ASSETS, es un Sistema de Gestión Integral sobre plataforma mixta. Este sistema es empleado por los departamentos de Contabilidad y Finanzas, Recursos Humanos, Logística y Auditoría para poder desempeñar sus tareas diarias de gestión. ASSETS permite mantener una Contabilidad fiable y actualizada, garantizando poder realizar los cierres de períodos contables en tiempo y con la calidad requerida. En la base de datos del ASSETS se almacena la información de todos los trabajadores del centro, desde información personal hasta su categoría, grado científico, puesto de trabajo, salario, etc. El uso del sistema ha elevado la agilidad y productividad de los procesos de gestión de contabilidad y recursos humanos en la universidad, permite llevar de manera precisa cada activo y brindar un registro general de procesos claves e información en el centro. Los métodos de administración y planificación de inventarios, análisis y consultas que el sistema pone a disposición permiten conocer la situación actual y proyectarse en la toma de decisiones a futuro lo cual genera un impacto en la universidad.

Situación Problemática

El uso del ASSETS para la realización de las tareas de gestión que se llevan a cabo en los distintos departamentos es de suma importancia. El empleo de este sistema aumenta la eficiencia de las actividades de gestión de la información en la universidad, pero el proceso de acceder a información almacenada en el sistema se torna difícil para personal que no tiene acceso directo a este. Hacer consultas a la información personal para un trabajador que no tiene permiso de usar el sistema, se hace difícil, tiene que ir a distintos departamentos de la universidad pues toda la información se encuentra dispersa en los distintos módulos del sistema. El usuario debe ir por cada departamento en

correspondencia del módulo del sistema que contenga la información, si quiere información de recursos humanos debe ir a recursos humanos y si quiere ver su nómina debe ir a contabilidad y finanzas en vez de poder conseguir toda la información por una única vía.

Otra situación se presenta a raíz de que toda la información está centralizada en la base de datos del sistema y el ASSETS no cuenta con un método para hacer visible esta información para que otros sistemas la puedan consumir, esto impide que otras aplicaciones construidas o que estén por implementarse y necesitan de dicha información para realizar sus procesos tengan que buscar otras vías para ingresar los datos. El sistema al no proveer una interfaz para la integración con otros sistemas, no hace sencilla la coordinación de la información entre otros sistemas en la UCf. Por esto se hace necesario crear un canal de comunicación secundario con la base de datos que no interfiera con las operaciones del ASSETS, pero permita poder acceder a la información para que pueda ser utilizada por otras aplicaciones de software de forma general.

Lo que lleva al **Problema Científico**: ¿Cómo acceder mediante aplicaciones externas a los datos del ASSETS de la Universidad de Cienfuegos?

Se define como **objeto de estudio** los microservicios para la gestión de la información. El **campo de acción** es el acceso a los datos en el ASSET de la Universidad de Cienfuegos. Para dar solución al problema planteado, se define como **Objetivo General** de este trabajo: Aplicación de un microservicio para la gestión de la información en el ASSETS de la Universidad de Cienfuegos.

Del objetivo general se desprenden los siguientes **Objetivos Específicos**:

1. Diseñar un microservicio REST para obtener la información de los trabajadores en el ASSETS, protegido bajo un esquema de autenticación JSONWebToken y una aplicación web que permita la gestión de los usuarios del microservicio.
2. Implementar el microservicio con REST, el esquema de autenticación JSONWebToken y la aplicación web para gestionar los usuarios.

3. Validar el microservicio, el esquema de autenticación y la aplicación web.

Se plantea como **Idea a Defender**: Con la aplicación de un microservicio para el acceso a la información del ASSETS de la Universidad de Cienfuegos se contribuirá a acceder de manera sencilla y rápida a la información por parte de otras aplicaciones para la gestión de los datos almacenados en el ASSETS.

Como **aporte práctico** se tiene que con la implementación del microservicio web para el acceso a la información del ASSETS de la Universidad de Cienfuegos se podrá acceder de manera sencilla y segura a los datos pertenecientes a los trabajadores, así como permitir a otras aplicaciones hacer uso de estos datos.

Los métodos de investigación que se aplican para alcanzar los objetivos de la investigación son los siguientes:

Métodos teóricos:

Método Histórico y Lógico: Se utiliza con el objetivo de profundizar en los antecedentes de las teorías correspondientes a la gestión de información e integración de sistemas revelando las características, tendencias e invariantes que se pudieran expresar en el modelo diseñado.

Método de Análisis y Síntesis: Utilizado durante todas las etapas de la investigación, permitiendo hacer los análisis e inferencias de la bibliografía consultada, además, en la búsqueda de información, datos que conducen a la selección de los aspectos significativos que conforman el núcleo básico del informe.

Modelación: Su uso permite hacer una representación teórica, metodológica y práctica de la propuesta de sistema. Las herramientas informáticas que se utilizan son SQL Server 2012 como sistema gestor de bases de datos, Nodejs como entorno de ejecución, Express como servidor web, Javascript como lenguaje de programación, Quasar como framework para desarrollar la interfaz de usuario, Swagger como el generador de la documentación y Visual Studio Code como editor de código.

Métodos de nivel empírico:

Entrevista: Se utiliza la entrevista como una conversación planificada con los clientes, para obtener información acerca del problema en cuestión. Su uso constituye un medio para el conocimiento cualitativo de las características particulares del proceso investigado y puede influir en el posterior análisis y diseño del producto de software.

El presente documento está estructurado en introducción, 3 capítulos, conclusiones, recomendaciones, referencias bibliográficas, y anexos donde se presenta la siguiente información:

Capítulo I: Fundamentos teóricos de los sistemas de gestión integral y la arquitectura de microservicios. Se exponen los aspectos teóricos de la investigación y los conceptos asociados al dominio del problema, se determinan los antecedentes y el estado actual de los sistemas informáticos de la gestión integral, así como la integración de sistemas y la arquitectura de microservicios, al igual que se describen metodologías, lenguajes y tecnologías utilizadas.

Capítulo II: Análisis y diseño del microservicio para la gestión de la información en el ASSETS de la Universidad de Cienfuegos. Se definen las características y conceptos fundamentales del modelo de dominio, así como se enuncian los requisitos funcionales y los no funcionales con el objetivo de entender la estructura del programa. Se describe y analiza el modelo de casos de uso del sistema, se lleva a cabo la implementación de la solución propuesta, los diagramas de clases del diseño con el fin de brindar una visión detallada del sistema por construir.

Capítulo III: Estudio de factibilidad y validación del microservicio para la gestión de la información en el ASSETS de la Universidad de Cienfuegos. En este capítulo, se describe el proceso de planificación del estudio de factibilidad, se lleva cabo la determinación de los costos, así como, los beneficios tangibles e intangibles asociados al proyecto. Además, se presenta un análisis de los beneficios vinculados al desarrollo de la aplicación. Finalmente se realizan los casos de pruebas funcionales para validar las posibles salidas del software ante fallos por parte del usuario.

1 – Capítulo I: Fundamentos teóricos de los sistemas de gestión integral y la arquitectura de microservicios

1.1 – Introducción

En este capítulo se abordarán aspectos teóricos del tema investigado, exponiendo los principales conceptos asociados al dominio del problema. Se describe el contexto donde se enmarca el mismo y las características y dificultades que este presenta. Se analizan sistemas existentes que emplean diferentes arquitecturas viendo sus ventajas y desventajas, así como se describen las metodologías, herramientas, tecnologías, lenguajes y herramientas empleadas para la construcción del sistema.

1.2 – Descripción del dominio del problema

1.2.1 Gestión Empresarial

En la Gestión Empresarial existen diversos enfoques como el tradicional, que estudia la estructura de la organización y define los papeles de las personas en la misma. El enfoque de las relaciones humanas es el resultado de la investigación de lo que realmente acontece en la organización, como son las personas que trabajan en ella. [26] El enfoque sistemático es más amplio y más dinámico que los anteriores. En este se observa toda la interrelación e interdependencia de los distintos elementos que la componen. Las relaciones humanas y la tradición son parte esencial de este enfoque, tenerlas en cuenta son pasos necesarios al analizar y comprender la conducta de la organización. [27]

Dentro de la gestión empresarial la gestión de los recursos humanos juega un papel importante. El capital humano es la columna vertebral de cualquier organización. Es así como el objetivo primario de la gestión de recursos humanos es mejorar los resultados de la misma. Dentro del área de Recursos Humanos se deben tratar todas las variables que afectan los resultados de los trabajadores, es decir, todas las situaciones que causan

que los trabajadores o colaboradores tengan un desequilibrio, el cual crea un decremento en sus labores normales.[28]

1.2.2 La Gestión de la Información en la Gestión Empresarial y los Sistemas de Información

Dada la gran cantidad de información que se genera a diario y se mantiene en las organizaciones, es de vital importancia que esta se gestione correctamente. Al hacer esto se logra disponer de la información necesaria en el momento oportuno. Colocar a disposición de todos los miembros de la empresa recursos de información generados dentro de la propia empresa, necesarios para el desarrollo de las tareas cotidianas. Integrar y administrar la información interna y externa como un todo para facilitar la toma de decisiones más eficientes. Reducir los costos de la administración de documentos. [29]

Esta necesidad de gestionar la información ha traído consigo crecimiento exponencial en el uso y la implantación de sistemas y tecnologías que mejoran dichas tareas de gestión dentro de la organización, facilitando su difusión y la conexión entre departamentos y organizaciones. De esta forma, muchas compañías están generando soluciones tecnológicas para mejorar los elementos y tareas mal realizadas dentro de sus operaciones y para facilitar la administración de las mismas. [30]

1.2.2.1 Herramientas de gestión empresarial

Una herramienta de gestión es esencialmente una aplicación, solución, metodología, paradigma, método, modelo, algoritmo, procedimiento, protocolo, sistema, indicador o instrumento específico que permite y facilita la administración del negocio y la organización de manera profesional. Las herramientas informáticas en la gestión empresarial son todas aquellas que ayudan a mejorar la planificación, coordinación y gestión de la empresa mediante el uso de software o aplicaciones informáticas. Pueden ser utilizadas en cualquier departamento de la organización, controlan y mejoran las actividades de las empresas e integran todos los datos, provocando que sea mejor la toma de decisiones.

La principal ventaja de estos sistemas es la gran ayuda aportada a los directivos para tomar decisiones porque recoge toda la información relevante. Al recogerse en plataformas informáticas se consigue conservar la información a pesar de que se produzca rotación del personal. [31]

1.2.2.2 Sistema de Gestión Integral:

En el actual entorno competitivo de negocios, la racionalización de las operaciones es esencial para mantener los costes generales bajos. Los principios clave de una gestión básica incluyen eliminar redundancias y consolidar sistemas. La implementación de un Sistema de Gestión Integral o Empresarial (ERP) proporciona un marco para hacer esto. Es un sistema único diseñado para gestionar múltiples aspectos de las operaciones de una organización. Los ERP ofrecen mayores beneficios que la ejecución de sistemas de gestión separados en paralelo.[32]

Un ERP nos permite reunir en una única aplicación todos los procesos de negocio, cubriendo de esta manera las necesidades tácticas, operativas y estratégicas de una compañía en todas sus áreas de actividad. Es integrado, modular y multiplataforma, se integra todo en una única aplicación, lo que implica la centralización de tareas. Un ERP permite gestionar los procesos de gestión de la calidad de forma integrada, a través de un solo sistema.[33]

1.2.3 ASSETS

ASSETS Ultimate es un Sistema de Gestión Integral sobre plataforma mixta. El núcleo central de la aplicación se encuentra bajo la concepción Cliente-Servidor, actualizada para SQL 2000, 2008 y 2012 y un conjunto de módulos WEB que interactúa con él. Permite el control de los procesos de Compras, Ventas, Producción, Taller, Inventario, Finanzas, Contabilidad, Presupuesto, Activos Fijos, Útiles y Herramientas y Recursos Humanos, en ambiente Cliente-Servidor. Como Sistema Integral, todos sus Módulos trabajan en estrecha relación, generando automáticamente al Módulo de Contabilidad, los Comprobantes de Operaciones por cada una de las transacciones efectuadas. Esto permite que se pueda trabajar bajo el principio de Contabilidad al Día.

Dispone de métodos novedosos para Administración y Planificación de Inventarios, así como, una amplia gama de Análisis y Consultas que le permitirán no únicamente conocer exactamente la situación actual, sino, proyectar decisiones futuras. Es un sistema flexible, amigable, con ayuda en línea, que puede ser instalado en sólo una microcomputadora o sobre varias, funcionando en ambiente multiusuario, incluidas estaciones remotas. Como sistema modular, usted tendrá la opción de instalar sólo los módulos que desee. Los datos estarán almacenados en el servidor de base de datos y disponibles para aquellas aplicaciones que accedan al servidor desde las estaciones de trabajo clientes. Permite mantener una Contabilidad fiable y actualizada, garantizando poder realizar los cierres de períodos contables en tiempo y con la calidad requerida.[34]

1.2.4 Integración entre sistemas

Los sistemas de información permiten el procesamiento de entradas, el almacenamiento de datos relacionados con la entidad y la producción de reportes y otro tipo de instrumentos de resumen de datos. La integración de sistemas de información, supone una disminución significativa de costos y esfuerzos de comunicación entre trabajadores y departamentos [35]. La integración de sistemas facilita la compartición de información con otras instituciones, hacer públicos los datos contenidos en los sistemas de información a integrar y al mismo tiempo ocultar su heterogeneidad [36]. La integración efectiva de la información de todos los sistemas de una organización es de gran utilidad en la toma de decisiones. [37]

La integración es necesaria cuando los datos se encuentran altamente fragmentados y distribuidos en múltiples sistemas, imposibilitando la comunicación y el intercambio de información entre ellos. Un sistema es interoperable gracias a su capacidad de transferir información de un sistema a otro, esto se logra manteniendo los datos en un formato que permita intercambiar información y transferirla de un sistema a otro a través de interfaces específicas, adaptadas o personalizadas [38]. La integración de sistemas es una herramienta que optimiza el trabajo de los equipos. Esta puede definirse como la conexión entre varios sistemas, de forma que las distintas áreas de la empresa trabajen de forma conjunta y automatizada. Esta integración permite una mayor interacción entre las

herramientas digitales y las diferentes plataformas tecnológicas. El uso de sistemas integrados ayuda a mejorar la productividad, el servicio al cliente, el uso de nuevas tecnologías, los procesos de trabajo internos.[39]

Existen varios tipos de integración como el intercambio de datos electrónicos, la integración de base de datos; en esta opción, se comparte una única base de datos entre diferentes sistemas con la ayuda de una plataforma de extracción de datos con una Interfaz de Programación de Aplicaciones (API por sus siglas en inglés). Es la solución más utilizada por las empresas porque ofrece un excelente rendimiento y es fácil de usar. En general, funciona como una especie de puente que conecta los sistemas, transmitiendo los datos de un lado a otro de forma cifrada, segura y ágil. [39]

1.3 – Descripción del objeto de estudio

1.3.1 - Objetivos estratégicos de la organización

Misión:

La Universidad de Cienfuegos «Carlos Rafael Rodríguez», dedicada a la formación integral continua de profesionales competentes y comprometidos con la Patria Socialista contribuye mediante el conocimiento y la innovación a la dinamización del desarrollo socioeconómico del territorio de Cienfuegos y de la sociedad cubana.

Visión:

Somos una Universidad de Excelencia que:

- Promueve una cultura general integral acorde con los valores de la sociedad cubana.
- Ofrece profesionales líderes comprometidos con la transformación de la sociedad cubana.
- Exhibe una alta visibilidad de sus resultados científicos y de innovación.

- Impacta en el desarrollo económico y social del territorio y la sociedad, satisfaciendo las necesidades de superación profesional y la implementación de resultados de investigación y la innovación.

1.3.2 - Flujo actual de los procesos y análisis crítico de la ejecución de estos

El ASSETS en la UCf se utiliza en la gestión económica y contable, es empleado por los departamentos de Contabilidad y Finanzas, Recursos Humanos, Logística y Auditoría para poder desempeñar sus tareas diarias de gestión. Permite mantener una Contabilidad fiable y actualizada, garantizando poder realizar los cierres de períodos contables en tiempo y con la calidad requerida. En la base de datos del ASSETS se almacena la información de todos los trabajadores del centro, desde información personal hasta su categoría, grado científico, puesto de trabajo, salario, etc. En cada departamento que emplea el ASSETS para la gestión de sus tareas diarias se usan diferentes módulos, en dependencia del departamento. El departamento de contabilidad y finanzas hay un personal encargado de llevar las finanzas, otro los útiles y herramientas, en dependencia de los módulos hay distintos trabajadores que se encargan de utilizarlos. Dentro de un mismo departamento hay personal calificado que puede usar de uno a varios módulos. En ATM (Los almacenes de logística) alrededor de 4 trabajadores interactúan con el sistema de diversas maneras.

En cada departamento los trabajadores tienen actividades distintas. La persona que se encarga de llevar útiles y herramientas trabaja con el módulo de útiles y herramientas, así como la que tiene que ver con activos fijos trabaja con el módulo correspondiente, en caso de que alguien quiera buscar un activo según su código debe ir a ver al personal en el sistema que realiza esa función el cual le hace el seguimiento en el sistema y devuelve la información del mismo. Hay personas que llevan inventario en el departamento de contabilidad y finanzas y otras trabajan con ese módulo en logística. Hay otro personal que lleva el combustible que es un activo de los almacenes. Para un trabajador recuperar su información del ASSETS debe ir a RH o contabilidad y finanzas en dependencia del tipo de información que desee. Si un trabajador tiene alguna duda con el pago debe ir al

especialista en recursos humanos que es el que hace las nóminas, en caso de duda con las tarjetas debe ir a contabilidad, como se puede ver en caso de tener cualquier duda debe irse al lugar específico.

El ASSETS cuenta con un administrador cuyo rol principal es cerciorarse que el sistema esté funcionando sin ningún problema para que los demás usuarios puedan hacer las tareas de su día a día, ya sea tareas de economía, recursos humanos o logística. El administrador debe estar al tanto de que la web esté funcionando, en caso de que a un usuario se le olvidara la contraseña, el administrador tiene que entrar al sistema en modo administrador y restablecerla. Cada vez que el usuario tenía un problema, estos se podían resolver de forma remota, en última instancia debe ir a arreglarlo personalmente.

El ASSETS es un sistema imprescindible para la realización de las tareas de gestión en la universidad, el uso de este programa aumenta en gran manera la productividad de los departamentos que lo emplean a diario, pero el proceso de obtener información almacenada en el sistema se hace engorroso para personal que no tiene acceso a este. Recuperar toda la información en general sobre un activo o trabajador, dado que se encuentra disperso en distintos módulos se hace difícil; si un trabajador quisiera acceder a toda su información, debe ir por cada departamento en correspondencia del módulo del sistema que contenga la información, si quiere información de recursos humanos debe ir a recursos humanos y si quiere ver su nómina debe ir a contabilidad y finanzas en vez de poder conseguir toda la información por una única vía. Otra situación se presenta cuando otro sistema informático necesita emplear información que está almacenada en el ASSETS para llevar a cabo algunos de sus procesos; esta tarea sería imposible porque El ASSETS no provee de un mecanismo de integración para que otras aplicaciones externas puedan hacer uso de la información en él almacenada.

1.4 – Descripción de los sistemas existentes

1.4.1 Amazon Web Services

Amazon Web Services (AWS) es la plataforma en la nube más adoptada y completa en el mundo, ofrece más de 200 servicios integrales de centros de datos a nivel global.

Millones de empresas están usando AWS para reducir los costos, aumentar su agilidad e innovar de forma más rápida. AWS cuenta con una cantidad de servicios y de características incluidas en ellos, ofreciendo desde tecnologías de infraestructura como cómputo, almacenamiento y bases de datos hasta tecnologías emergentes como aprendizaje automático e inteligencia artificial. [40]

Sus ingresos de Public Cloud SaaS alcanzaron los 145 500 millones de USD en 2021. Algunos de los servicios de AWS más populares son Amazon EC2, Amazon S3 y Amazon Aurora. Amazon EC2 es uno de los servicios de AWS de computación en la nube de más rápido crecimiento, que ofrece servidores virtuales para administrar cualquier tipo de carga de trabajo. Amazon S3, que es un servicio de almacenamiento de objetos de AWS, que es altamente escalable y ayuda a los usuarios a acceder a cualquier cantidad de datos desde cualquier lugar. Amazon Aurora es una base de datos relacional compatible con MySQL y PostgreSQL de alto rendimiento. Permite automatizar tareas cruciales, como el aprovisionamiento de hardware, la configuración y las copias de seguridad de la base de datos, y la aplicación de parches. [41]

1.4.2 Google Analytics

Google Analytics es una herramienta de analítica web de la empresa Google lanzada el 14 de noviembre de 2005. Ofrece información del tráfico que llega a los sitios web según la audiencia, la adquisición, el comportamiento y las conversiones que se llevan a cabo en el sitio. Se pueden obtener informes como el seguimiento de usuarios exclusivos, el rendimiento del segmento de usuarios, los resultados de las diferentes campañas de marketing en línea, las sesiones por fuentes de tráfico, tasas de rebote, duración de las sesiones, contenidos visitados y conversiones para e-commerce. El 14 de octubre de 2020, Google anuncio el lanzamiento de Google Analytics 4 (GA4). La API de informes de Google Analytics, versión 4 es el método programático más avanzado para acceder a los datos de informes en Google Analytics y permite crear paneles personalizados para mostrar datos de Google Analytics, automatizar las tareas complejas de creación de informes para ahorrar tiempo e integrar los datos de Google Analytics con otras

aplicaciones empresariales. La API permite acceder a todas las ventajas que ofrece la plataforma de Google Analytics. [42]

1.4.3 Servicios Web para la integración de la información de los sistemas informáticos SIGENU y ASSETS en la Universidad de Cienfuegos (UCf).

En la UCf se llevó a cabo un proyecto como base en un trabajo de diploma para la implementación de un servicio web. El proyecto surge con la necesidad de acceder a información que se encuentra dispersa entre dos sistemas distribuidos incomunicados dentro de la UCf. Para implementar los servicios web se empleó como gestor de bases de datos PostgreSQL y SQL Server, para el desarrollo del programa principal se utilizó el lenguaje de programación PHP. Aprovechando las potencialidades que nos ofrece el componente RESTful del Framework Yii 2 se implementaron un conjunto de servicios web, los cuales funcionan como mediadores entre los sistemas SIGENU, ASSETS y aplicaciones de terceros. Los servicios anteriormente mencionados ejecutan consultas a las respectivas bases de datos, poniendo a disposición la información requerida de manera rápida y confiable. Cada uno de ellos ofrece la posibilidad de obtener datos específicos en dependencia de cuál de ellos sea el que se ejecute. Estos servicios web no han podido ser puestos en funcionamiento en la entidad. [43]

La arquitectura basada en microservicios es un enfoque novedoso que está cobrando fuerzas y empresas que están en la punta de la lanza en innovación y desarrollo tecnológico la emplean para dar solución a sus requerimientos de negocio. Los servicios web nos permitirán comunicar la capa de la base de datos con aplicaciones de terceros, dándonos las ventajas que esta arquitectura promueve.

La propuesta de solución presente va más allá que otras propuestas de solución en la UCf dado que provee un mecanismo de autenticación para poder supervisar el acceso al microservicio, la capa de seguridad es muy importante en las aplicaciones y hay información en el ASSETS que solo personal autorizado puede acceder por lo que no cualquier nodo en la red debería poder acceder a ella, solo los sistemas o usuarios

permitidos. Con la implementación de un panel de administración, esta tarea se podrá desarrollar de forma sencilla. También se provee una documentación de como poder utilizar los distintos recursos que brinda el sistema, permitiendo a los sistemas que quieran hacer uso de estos recursos los medios para poder consumir los mismos de forma correcta y segura.

1.5 – Tendencias, metodologías y/o tecnologías actuales

1.5.1 Metodologías de desarrollo de software

1.5.1.1 Metodologías robustas, RUP

El proceso unificado de Rational (RUP), es un modelo de software que permite el desarrollo de software a gran escala, mediante un proceso continuo de pruebas y retroalimentación, garantizando el cumplimiento de ciertos estándares de calidad. Aunque con el inconveniente de generar mayor complejidad en los controles de administración del mismo. Sin embargo, los beneficios obtenidos recompensan el esfuerzo invertido en este aspecto.[44]

RUP es Dirigido por casos de uso, centrado en la arquitectura, iterativo e incremental, este genera una amplia documentación y consta de 4 fases: inicio, elaboración, construcción y transición (anexo 1.2). RUP se basa en componentes, lo que significa que el sistema en construcción está hecho de componentes de software interconectados por medio de interfaces bien definidas. Usa el Lenguaje de Modelado Unificado (UML) en la preparación de todos los planos del sistema. De hecho, UML es una parte integral de RUP, fueron desarrollados a la par.[44]

1.5.1.2 UML

El Lenguaje de Modelado Unificado (UML) es un lenguaje que permite modelar, construir y documentar los elementos que forman un sistema software orientado a objetos, convirtiéndose en el estándar de facto de la industria. Este lenguaje tiene una notación gráfica muy expresiva que permite representar en mayor o menor medida todas las fases de un proyecto informático: desde el análisis con los casos de uso, el diseño con los

diagramas de clases, objetos, etc., hasta la implementación y configuración con los diagramas de despliegue. UML es ante todo un lenguaje que proporciona un vocabulario y unas reglas para permitir una comunicación y se centra en la representación gráfica de un sistema. Entre sus objetivos fundamentales se encuentran: [45]

- Ser tan simple como sea posible, pero manteniendo la capacidad de modelar toda la gama de sistemas que se necesita construir.
- Necesita ser lo suficientemente expresivo para manejar todos los conceptos que se originan en un sistema moderno, tales como la concurrencia y distribución, así como también los mecanismos de la ingeniería de software, como son el encapsulamiento y el uso de componentes.
- Debe ser un lenguaje universal, como cualquier lenguaje de propósito general. Es escogido como lenguaje del modelado para el sistema informático.

1.5.2 Tendencias en la integración de sistemas

1.5.2.1 Arquitectura Orientada a Servicios

La Arquitectura Orientada a Servicios (SOA) es una filosofía de construcción de sistemas computacionales para crear procesos de negocios empaquetados como servicios. Define el intercambio de datos mediante aplicaciones a través de servicios web y así participar de los procesos de negocio. Estas funciones son independientes de los sistemas operativos y lenguajes en que están desarrollados. SOA separa las funcionalidades en distintas unidades (Servicios) que pueden ser distribuidos en una red, pudiendo ser combinados y reutilizados para crear nuevas aplicaciones de negocio[46]. SOA es adecuada para el desarrollo de aplicaciones distribuidas, sobre todo a gran escala, puesto que facilitan la interacción entre proveedores de servicios y consumidores de los servicios, así los recursos se harán accesibles a través de servicios independientes, a los que se podrá acceder sin conocer los detalles de la implementación. [46]

Muchas empresas de software desarrollan sus productos con SOA siguiendo el enfoque de servicios Web. Los servicios Web son una tecnología que permite la reutilización de

los recursos y agilidad en los sistemas. Según la organización W3C “Un servicio web es una aplicación de software identificada por una Uniform Resource Identifier (URI), cuyas interfaces y relaciones son posibles de definir y descubrir mediante artefactos XML, y que soporta interacciones directas con otras aplicaciones utilizando mensajes basados en XML mediante protocolos de internet.” [47]

Los servicios Web son componentes de software ligeramente acoplados entregados a través de tecnologías estándar de Internet. Es decir, los servicios Web son aplicaciones de negocio autodescriptivas y modulares que exponen la lógica empresarial como servicios a través de Internet a través de la interfaz programable y donde el protocolo de Internet (IP) puede ser utilizado para encontrar e invocar esos servicios. Un servicio web es el elemento clave en la integración de diferentes sistemas de información, ya que los sistemas de información pueden basarse en diferentes plataformas, lenguajes de programación y tecnologías. [48]

La ventaja de separar la lógica de negocio en una fase con tecnología de servicio web, radica en que la lógica de negocio pasa a ser un módulo, lo que permite ser invocado por múltiples interfaces y distintas aplicaciones, ya sea de ventana, móviles o web, aprovechando la fortaleza de los servicios web en cuanto a interoperabilidad e independencia de plataformas[49]. Los servicios web permiten establecer conexiones de cualquier tipo entre dos herramientas de forma casi transparente. Si un servicio web deja de funcionar o cambia, el ecosistema continuará funcionando.[50]

1.5.2.2 Arquitectura de Microservicios

La arquitectura basada en Microservicios es una variante de SOA, cada servicio debe estar finamente separados y se comunican entre ellos utilizando métodos de comunicación ligeros como puede ser un recurso HTTP a través de una API [51]. Se considera que esta arquitectura ha evolucionado a partir de otras existentes como: aplicaciones monolíticas desarrolladas utilizando el patrón de arquitectura en capas y aplicaciones distribuidas desarrolladas a través del patrón SOA, las aplicaciones desarrolladas bajo el patrón SOA ofrece niveles de abstracción, conectividad heterogénea, orquestación de servicios y el alineamiento de TI con los objetivos de

negocios; sin embargo, es complejo, costoso, ubicuo, difícil de entender e implementar para la mayoría de las aplicaciones es en estos casos en donde la arquitectura de microservicios brilla. [52]

Entre las características de esta Arquitectura se encuentran:

- Distribución. Las funcionalidades de la arquitectura se ofrecen como servicios que pueden estar ubicados en diferentes lugares.
- Heterogeneidad. Los servicios son independientes de la tecnología subyacente; se puede ejecutar en máquinas diferentes, sobre sistemas operativos diferentes y puede estar implementada con lenguajes diferentes.
- Interoperabilidad. La especificación de los servicios sigue estándares. Servicios implementados con tecnologías diferentes puedan interoperar porque la interfaz es común.
- Desacoplamiento. Los servicios exponen funcionalidades que se pueden ofrecer de forma remota y distribuida. Esto permite desacoplar el servicio.
- Flexibilidad. Un servicio puede apoyar o ser usado por diferentes aplicaciones, lo que permite más flexibilidad en el desarrollo de nuevas aplicaciones.
- Escalabilidad. Los servicios se pueden replicar y distribuir de forma transparente al usuario, que los ve como una única fachada.

Los microservicios son un enfoque arquitectónico y organizativo del desarrollo de software en el que las aplicaciones están compuestas por pequeños servicios independientes que se comunican a través de un API bien definida y protocolos ligeros cada uno ejecutándose en su propio proceso. La arquitectura de microservicios ayuda a reducir la complejidad manejar las pruebas automatizadas, integración y despliegue continuo, seguridad y tolerancia a fallas[53]. Los servicios se construyen en torno a funcionalidades de negocio y se pueden desplegar de forma individual, permitiendo automatizar completamente los mecanismos de despliegue. El manejo de los datos se

realiza de forma descentralizada, separando el modelo de datos de forma análoga a la separación en microservicios. [54]

El proceso de desarrollo de software se vuelve más flexible bajo el enfoque de microservicios, cada microservicio está compuesto por su propia lógica de negocios, interfaz de usuario, base de datos, funcionalidad de comunicación con otros servicios y se despliega por separado. Esto hace posible cambiar y extender gradualmente con nuevas características sin afectar el resto del sistema, lo cual difiere de la forma tradicional o monolítica. En una aplicación monolítica toda la lógica se ejecuta en un único servidor de aplicaciones. Las aplicaciones monolíticas típicas son grandes y construidas por múltiples equipos, requiriendo una orquestación cuidadosa del despliegue para cada cambio. Una base de código única para toda la aplicación. Típicamente, totalmente desarrollado en un lenguaje de programación. Requiere escalar la aplicación entera, aunque los cuellos de botella estén localizados. [55]

1.5.2.3 Implementaciones en SOAP o en RESTful

En la arquitectura de microservicios se usa un marco de mensajería común, como SOAP o REST. Los Servicios web SOAP utilizan, para intercomunicarse, mensajes en lenguaje XML que siguen su estándar. La implementación de servicios web por protocolo simple de acceso a objetos (SOAP) se desarrolló como una alternativa al estándar CORBA (Common Object Request Broker Architecture). Para garantizar el transporte de datos en SOAP, se utilizan protocolos como HTTP, SMTP, entre otros, en formato XML. En este enfoque, un proveedor de servicios publica una descripción del servicio o una interfaz para el registro de servicios, por lo que el solicitante del servicio puede encontrar una instancia de servicio correcta y utilizarla. Algunos problemas de rendimiento en SOAP se producen al formar el mensaje SOAP porque agrega un encabezado adicional y partes al cuerpo al mensaje. Los servicios Web basados en SOAP incluyen una variedad de estándares estas normas fueron desarrolladas por organizaciones de normalización, como W3C. [56]

Los Servicios web RESTful (Representational State Transfer) son adecuados para escenarios de integración básicos. Se suelen integrar mejor con el protocolo HTTP que

los servicios basados en SOAP, debido a que no requieren mensajes en lenguaje XML o de las definiciones del servicio. Estos servicios utilizan estándares como HTTP, URI, SML, MIME, y tienen una infraestructura que permite la construcción de los servicios con el mínimo uso de herramientas; así, se consigue que el desarrollo de estos servicios sea de bajo costo y de fácil adopción. Esta arquitectura es más simple y ligera que la arquitectura SOAP. Usar las API de RESTful permite una distribución más rápida de nuevas funciones y actualizaciones. Esta arquitectura liviana optimiza los recursos distribuidos o en la nube y admite la escalabilidad dinámica de los servicios individuales. [56]

1.5.2.4 RESTful

REST es un estilo de arquitectura software para sistemas hipermedia distribuidos como la World Wide Web, también conocido como servicios web RESTful, estos permiten realizar solicitudes a sistemas que exponen estos servicios, para acceder y manipular un recurso web usando un conjunto de acciones predefinidas. Su característica principal es que no tiene estado, lo que significa que entre dos llamadas el servicio pierde todos los datos porque el estado lo tiene que mantener el cliente, de aquí su nombre. Si se necesita que un servicio REST recuerde cierta información, es necesario identificarse en cada llamada, ya sea por un usuario y contraseña, un token u otro tipo de credenciales.

En la actualidad existe una gran cantidad de proyectos o aplicaciones que disponen de una API REST; Twitter, YouTube, los sistemas de identificación con Facebook, y cientos de empresas generan negocio gracias a REST y las API REST. En los últimos años, logró un gran impacto en la web que prácticamente logró desplazar a SOAP por tener un estilo bastante más simple de utilizar y sobre todo por su eficiencia. Este nuevo estilo ha supuesto una nueva opción de estilo de uso de los Servicios Web. [57]

Para la implementación del microservicio se tomó como estilo la arquitectura REST dado los beneficios que aporta a la realización del sistema, permite utilizar el protocolo http con los métodos que este pone a disposición, logrando simplificar el intercambio de recursos entre las aplicaciones y el microservicio al no tener que instalarse una capa más gruesa como SOAP. Dada a la alta integración que posee el framework Express con esta

arquitectura al permitir que sus rutas se comporten como una API REST por defecto, el tiempo de desarrollo disminuirá grandemente.

Este protocolo destaca por su escalabilidad. Gracias a la separación entre el cliente y el servidor, el producto se puede escalar sin que ello represente muchas dificultades. Por su flexibilidad y portabilidad es posible realizar una migración de un servidor a otro o practicar cambios en la base de datos en todo momento. De esta forma, el frontend y el backend se pueden alojar en servidores diferentes, lo que supone una enorme ventaja de manejo. Debido a la separación entre el cliente y el servidor, el protocolo facilita que los desarrollos de las diferentes partes de un proyecto se puedan dar de manera independiente. Además de ello, la API REST se adapta en todo momento al tipo de sintaxis o plataformas de trabajo. Esto brinda la oportunidad de probar varios entornos dentro del desarrollo.

1.5.3 Lenguajes y Herramientas Utilizadas

1.5.3.1 HTML

HTML es el lenguaje con el cual que desarrollan las páginas web, es el estándar usado por los navegadores para mostrar las páginas web al usuario. Nos permite agrupar textos, sonidos e imágenes y combinarlos a nuestro gusto. Además, nos permite la introducción de referencias a otras páginas por medio de los enlaces hipertexto. HTML 5 Es la quinta revisión del lenguaje HTML, define los nuevos estándares de desarrollo web, redefiniendo el código para solucionar problemas y actualizándolo así a nuevas necesidades. No se limita solo a crear nuevas etiquetas o atributos, sino que incorpora características nuevas y proporciona una plataforma de desarrollo de complicadas aplicaciones web (mediante la interfaz de programación y aplicaciones API). El lenguaje HMTL5 se encarga de desarrollar una representación sobre los contenidos utilizados como textos, fotografías, animaciones para definir la estructura básica de una página web.[58]

1.5.3.2 CSS

CSS es un lenguaje para definir el estilo o la apariencia de las páginas web, escritas con HTML o de los documentos XML. CSS se creó para separar el contenido de la forma, a

la vez que permite a los diseñadores mantener un control mucho más preciso sobre la apariencia de las páginas. CSS es el acrónimo de Cascading Style Sheets, o en español Hojas de Estilo en Cascada; se define basándose en un estándar publicado por la organización W3C, que también se encarga de estandarizar el propio lenguaje HTML. El motivo que ha hecho necesaria la creación de CSS ha sido la separación del contenido de la presentación. CSS se ideó para aplicar el formato en las páginas, de una manera mucho más detallada, con nuevas posibilidades que no estaban al alcance de HTML. Al mismo tiempo, gracias a la posibilidad de aplicar el estilo de manera externa al propio documento HTML, se consiguió que el mantenimiento de las páginas fuese mucho más sencillo.[59]

1.5.3.3 JavaScript

JavaScript es un lenguaje de programación que se utiliza principalmente para crear páginas web dinámicas. JavaScript es un lenguaje de programación interpretado, por lo que no es necesario compilar los programas para ejecutarlos; los programas escritos con este lenguaje se pueden probar directamente en cualquier navegador sin necesidad de procesos intermedios. La sintaxis de JavaScript es muy similar a la de otros lenguajes de programación como Java y C. JavaScript fue diseñado de forma que se ejecutara en un entorno muy limitado que permitiera a los usuarios confiar en la ejecución de los scripts. Los scripts de JavaScript no pueden comunicarse con recursos que no pertenezcan al mismo dominio desde el que se descargó el script. Los scripts tampoco pueden cerrar ventanas que no hayan abierto esos mismos scripts. Además, los scripts no pueden acceder a los archivos del ordenador del usuario (ni en modo lectura ni en modo escritura) y tampoco pueden leer o modificar las preferencias del navegador. Por último, si la ejecución de un script dura demasiado tiempo, el navegador informa al usuario de que un script está consumiendo demasiados recursos y le da la posibilidad de detener su ejecución. [60]

1.5.3.4 Nodejs

Node.js, una plataforma de ejecución la cual permite ejecutar JavaScript de parte del servidor. Node está orientado a eventos asíncronos de JavaScript diseñado para construir

aplicaciones de red escalables. Si no hay trabajo por hacer Node.js permanece dormido hasta que haya una conexión para que se active una llamada al servicio, todo esto es posible, ya que se usa JavaScript del lado de servidor. Node.js se ejecuta sobre la máquina virtual V8 de Chrome. Puede generar páginas con contenido dinámico. Puede crear, abrir, leer, escribir, eliminar y cerrar archivos en el servidor. Puede agregar, eliminar, modificar registros de la base de datos. Los archivos .js contienen métodos que se ejecutaran de acuerdo a ciertos eventos. Fácil y sencillo de aprender e implementar. En la actualidad es considerado lo mejor para el desarrollo de REST API. Poco código para mantener. Fácil de testear con aplicaciones con plugins como Mocha/Chai. Empaquetamiento a través de NPM. Modelo de no bloqueo de I/O. Trabaja con casos síncronos y asíncronos. Fácil integración con bases de datos No SQL. [61]

1.5.3.5 Express

Express es un entorno de trabajo para aplicaciones web que corre sobre Node.js, de código abierto y con licencia MIT. Se utiliza para desarrollar aplicaciones web y API. El autor original es TJ Holowaychuk y la primera versión se lanzó el 2010. Express es una infraestructura de aplicaciones web Node.js mínima y flexible que proporciona un conjunto sólido de características para las aplicaciones web y móviles. Con miles de métodos de programa de utilidad HTTP y middleware a su disposición, la creación de una API sólida es rápida y sencilla. Express proporciona una delgada capa de características de aplicación web básicas, que no ocultan las características de Node.js. [62]

1.5.3.6 JSON Web Token

JSON Web Token (JWT) es un estándar abierto basado en JSON propuesto por IETF (RFC 7519) para la creación de tokens de acceso que permiten la propagación de identidad y privilegios. El token está firmado por la clave del servidor, así que el cliente y el servidor son ambos capaces de verificar que el token es legítimo. Los JSON Web Tokens están diseñados para ser compactos, los privilegios pueden ser utilizados para propagar la identidad de usuarios como parte del proceso de autenticación entre un proveedor de identidad y un proveedor de servicio, o cualquiera otro tipo de privilegios requeridos por procesos empresariales. Los JWT generalmente están formados por tres

partes: un encabezado o header, un contenido o payload, y una firma o signature. El encabezado identifica qué algoritmo fue usado para generar la firma. El token puede ser fácilmente transmitido en entornos HTML y HTTP, siendo similar a estándares basados en XML como SAML.

Generalmente, los algoritmos de cifrado utilizados son HMAC con SHA-256 (HS256) y firma digital con SHA-256 (RS256). La autorización se logra cuando el usuario ingresa sus credenciales con éxito, entonces se genera un JSON Web Token que es retornado al cliente, quien tiene que guardarlo localmente, en vez del modelo tradicional de crear una sesión en el servidor y retornar una cookie. Siempre que el usuario quiere acceder a una ruta protegida o recurso, el cliente tiene que enviar el JWT, generalmente en el encabezado de Authorization utilizando el esquema Bearer. Este es un mecanismo de autenticación sin estado - stateless-, ya que la sesión del usuario nunca se guarda en el proveedor de identidad o en el proveedor del servicio. Los recursos protegidos siempre comprobarán si existe un JWT válido en cada pedido de acceso. Si el token está presente y es válido, el proveedor del servicio otorga accesos a los recursos protegidos. Como los JWT contienen toda la información necesaria en sí mismos, se reduce la necesidad de consultar la base de datos u otras fuentes de información múltiples veces. [63]

1.5.3.7 SQLite

SQLite es una base de datos relacional integrada de código abierto que utiliza SQL para realizar las consultas. Lanzado originalmente en 2000, fue diseñado para proporcionar una forma conveniente para que las aplicaciones administren datos sin los gastos generales que a menudo conllevan los sistemas de administración de bases de datos relacionales dedicados. SQLite tiene la reputación de ser altamente portátil, fácil de usar, compacto, eficiente y confiable. En lugar de ejecutarse de forma independiente como un proceso independiente, coexiste simbióticamente dentro de la aplicación a la que sirve, dentro de su espacio de proceso. Su código está entrelazado, o incrustado, como parte del programa que lo aloja. SQLite es bastante versátil; en un entorno web puede ayudar a administrar información de sesión compleja. En lugar de serializar los datos de la sesión en un gran blob, las piezas individuales se pueden escribir y leer de forma selectiva en bases de datos de sesiones individuales. SQLite también puede servir como caché,

contener datos de configuración o, debido a su compatibilidad binaria entre plataformas, incluso funcionar como un formato de archivo de aplicación.[64]

1.5.3.8 Microsoft SQL Server

Microsoft SQL Server es un sistema de gestión de bases de datos relacionales (RDBMS) que admite una amplia variedad de aplicaciones de procesamiento de transacciones, inteligencia empresarial y análisis en entornos informáticos corporativos. Microsoft SQL Server es una de las tres tecnologías de bases de datos líderes del mercado, junto con Oracle Database y DB2 de IBM. Microsoft SQL Server se basa en SQL, se construye principalmente en torno a una estructura de tablas basada en filas que conecta los elementos de datos relacionados en diferentes tablas entre sí, evitando la necesidad de almacenar datos de forma redundante en varios lugares dentro de una base de datos. El modelo relacional también proporciona integridad referencial y otras restricciones de integridad para mantener la exactitud de los datos. [65]

1.5.3.9 Vue

Vue es un marco progresivo para la construcción de interfaces de usuario, diseñado desde cero para ser adaptable, es capaz de impulsar aplicaciones sofisticadas de una sola página, combinando consigo mismo, herramientas modernas y bibliotecas de soporte. Los componentes son una de las características más poderosas de Vue.js. Ellos te ayudan a extender elementos básicos de HTML para encapsular código reusable. En un nivel alto, los componentes son elementos personalizados a los que el compilador de Vue.js asignará un comportamiento específico. Los componentes de archivo único (SFC) son archivos con una extensión .Vue que contienen la definición completa de un solo componente y se pueden importar a su aplicación Vuejs. Los SFC simplifican la creación y el uso de componentes. [66]

1.5.3.10 Quasar

Quasar es un marco de trabajo basado en Vue.js de código abierto, que permite crear sitios y aplicaciones web en diferentes entornos tales como aplicación de escritorio, móviles, webs, entre otras. Entre las bondades que ofrece Quasar ante otros framework

está que es fácil de personalizar y ampliar. Quasar está basado en Vue.js, ofrece una interfaz de usuario de última generación que sigue las pautas de material. Es un marco de trabajo centrado en el rendimiento. Posee una comunidad en crecimiento y activa, y una documentación extensa y de calidad. Posee modo de compilación para SPA, SSR, PWA, aplicación móvil, aplicación de escritorio y extensión del navegador y ofrece una experiencia de desarrollador automatizada a través de una estrecha integración con su propia CLI. [67]

1.5.3.11 Swagger

Swagger es un conjunto de herramientas de software de código abierto para diseñar, construir, documentar, y utilizar servicios web RESTful. Fue desarrollado por SmartBear Software e incluye documentación automatizada, generación de código, y generación de casos de prueba. Al crear API, las herramientas Swagger se pueden usar para generar automáticamente un documento en el código mismo usando el estandar OpenAPI. Alternativamente, utilizando Swagger Codegen, los desarrolladores pueden desacoplar el código fuente del documento Open API y generar código de cliente y servidor directamente desde el diseño. Esto hace posible diferir el aspecto de codificación. Cuando se describe en un documento de OpenAPI, las herramientas de código abierto de Swagger pueden usarse para interactuar directamente con la API a través de la interfaz de usuario de Swagger. Este proyecto permite conexiones directamente a API en vivo a través de una interfaz de usuario interactiva basada en HTML. Las solicitudes se pueden realizar directamente desde la interfaz de usuario y las opciones exploradas por el usuario de la interfaz. [68]

1.5.3.12 Visual Studio Code

Visual Studio Code es un editor de código fuente ligero, pero potente que está disponible para Windows, macOS y Linux. Viene con soporte integrado para JavaScript, TypeScript y Node.js y tiene un rico ecosistema de extensiones para otros lenguajes y tiempos de ejecución como C++, C#, Java, Python, PHP, Go, .NET. Permite trabajar de forma remota utilizando la extensión gratuita LiveShare y editar o depurar el código en tiempo real. Visual Studio Code resalta las palabras clave en el código en diferentes colores para

ayudarlo a identificar fácilmente los patrones de codificación. También se pueden aprovechar características como IntelliSense y Peek Definition, que ayudan a comprender cómo se pueden usar las funciones y cómo se relacionan entre sí. A medida que codifica, Visual Studio Code brinda sugerencias para completar líneas de código y soluciones rápidas para errores comunes. También al usar el depurador se recorre cada línea de código y se puede comprender lo que está sucediendo. [69]

1.5.3.13 Visual Paradigm

Visual Paradigm es una herramienta CASE empleada para el modelado de los procesos de desarrollo de software. Es una herramienta profesional que soporta el ciclo de vida completo del desarrollo de software: análisis y diseño orientados a objetos, construcción, pruebas y despliegue. Entre sus principales características se encuentran que es multiplataforma, posee interoperabilidad, facilita la colaboración en equipo y brinda apoyo al ciclo de vida completo del desarrollo de software. Permite dibujar todos los tipos de diagramas de clases, código inverso, generar código desde diagramas y generar documentación. Agiliza la construcción de aplicaciones con calidad y a un menor coste. Posibilita la generación de bases de datos, transformación de diagramas de Entidad-Relación en tablas de base de datos, así como ingeniería inversa de bases de datos. [70]

1.5.3.14 Zotero

Zotero es una extensión libre para el navegador Firefox, que permite a los usuarios recolectar, administrar y citar investigaciones de todo tipo. Importa datos directamente desde las páginas web visualizadas en el momento. Detecta automáticamente cuándo estamos consultando una fuente de información: libros, artículos, revistas, bases de datos, catálogos de bibliotecas, e incluso otros recursos como Google Scholar, Amazon, Flickr o Youtube. Disponible inicialmente como complemento para el navegador Firefox, desde la versión 3.0 también funciona como programa independiente. Funciona como aplicación de administración de referencias, usada para administrar bibliografías y referencias al escribir ensayos y artículos. [71]

1.6 – Conclusiones

En este capítulo se realizó un estudio de los conceptos asociados al dominio del problema, así como un análisis de las tecnologías Web, las metodologías, los lenguajes de programación y gestor de bases de datos. Seleccionándose la metodología RUP como guía para el desarrollo y documentación del software propuesto. El sistema se construyó con base en la arquitectura de microservicios. Para el desarrollo de la aplicación se seleccionó Javascript como lenguaje de programación del lado del servidor, se utilizó NodeJs como entorno de ejecución de Javascript y como framework ExpressJS. Se usaron HTML, CSS, y Javascript como lenguajes del lado del cliente y el framework Quasar para desarrollar la interfaz de usuario.

2 – Capítulo II: Análisis y diseño del microservicio para la gestión de la información en el ASSETS de la Universidad de Cienfuegos

2.1 – Introducción

En este capítulo se aborda la Modelación de los Procesos del Negocio, según lo define y lo denomina la Metodología RUP y se obtienen artefactos que permiten modelar estos procesos utilizando el UML como lenguaje de modelado. Además, se analiza y describe el modelo del sistema. Se identifican los requerimientos funcionales y los requerimientos no funcionales, así como los actores, organizándolos por casos de uso del sistema y se representa la relación entre ellos a través del diagrama de casos de uso del sistema.

2.2 – Modelo del dominio

Un Modelo de Dominio es un artefacto de la disciplina de análisis. El modelo de dominio se crea con el fin de representar el vocabulario y los conceptos clave del dominio del problema. El modelo de dominio también identifica las relaciones entre todas las entidades comprendidas en el ámbito del dominio del problema. Los modelos de dominio pueden utilizarse para capturar y expresar el entendimiento ganado en un área bajo análisis como paso previo al diseño de un sistema. El modelo de dominio sirve para identificar y representar conceptos del dominio de problema, establecer y entender las relaciones entre los conceptos e identificar atributos de cada concepto.

2.2.1 – Definición de las entidades y los conceptos principales

ASSETS: es un Sistema de Gestión Integral sobre plataforma mixta. El núcleo central de la aplicación se encuentra bajo la concepción Cliente-Servidor, actualizada para SQL 2000, 2008 y 2012 y un conjunto de módulos WEB que interactúa con él. Es un sistema estándar y parametrizado, que permite el control de los procesos de Compras, Ventas,

Producción, Taller, Inventario, Finanzas, Contabilidad, Presupuesto, Activos Fijos, Útiles y Herramientas y Recursos Humanos, en ambiente Cliente-Servidor,

La base de datos del ASSETS es una base de datos relacional y está construida en SQL Server 2012. Los distintos módulos del ASSETS se comunican con ella para almacenar, modificar y recuperar la información. En esta base de datos se generó un usuario de solo lectura, que es el que usara el microservicio para comunicarse con la base de datos.

Administrador del ASSETS: Es un trabajador de la entidad cuya principal tarea es cerciorarse que el ASSETS esté funcionando sin ningún problema para que los demás usuarios puedan utilizarlo correctamente. Debe asegurar de que la web esté funcionando, así como gestionar que las credenciales de los usuarios estén al día y operativas y asignar o quitar permisos a los mismos. El administrador interactúa directamente con la base de datos para crear nuevas consultas o asegurarse que la información se almacena correctamente. También gestiona los backups de la base de datos y las salvallas del sistema.

Usuario Especializado: es aquel que tiene práctica en el uso del sistema y se desenvuelve sin dificultad, interactúa con el sistema a través de la interfaz visual y realiza las operaciones diarias de control y administración del flujo de información en el sistema. El ASSETS es utilizado por el personal de los departamentos de Contabilidad y Finanzas, Recursos Humanos, Logística y Auditoría en la Universidad. Según el departamento, cada trabajador utiliza uno o varios módulos y para poder hacer uso de los mismos debe tener permisos del sistema para hacer las operaciones.

Microservicio: software ubicado en la red de la universidad, construido sobre Nodejs y basado en la arquitectura Rest para la creación de API. Provee de una vía de conexión a la información de la base de datos del ASSETS, logrando así poder compartir información que es necesaria para la realización de otros procesos en la universidad y sirviendo como medio de acceso a otras aplicaciones que lo necesiten. Este habilita un panel de administración para la gestión de las aplicaciones que pueden consumir la información, así como la documentación de como poder acceder a los recursos. La gestión de las aplicaciones es llevada por el administrador del microservicio.

Administrador del Microservicio: Se encarga de que el microservicio esté funcional y operativo en la red de la UCf, así como; que la conexión a la base de datos se realice de forma exitosa. Es el encargado de llevar la gestión de las aplicaciones que pueden hacer uso del microservicio, crearle un usuario y darle su api key. Ante alguna solicitud de los usuarios, uso indebido de estos del sistema, dar de baja a usuarios, o modificar el api key de uno de ellos, el administrador hace uso del panel de administración para realizar estas tareas.

Aplicaciones: este tipo de sistemas son los que se comunicarán con el microservicio y podrán consumir la API del mismo. Se le proveerá de un usuario y api key vinculada a ese usuario para acceder a los datos de forma segura, para poder obtener el usuario debe hacer la solicitud con el administrador del microservicio. Este usuario pueden ser clientes de frontend de cualquier arquitectura, así como otros backend que quieran emplear los datos para una gestión más avanzada.

2.2.2 – Reglas del negocio a considerar

- Solo el ASSETS puede modificar los datos que se gestionan en su base de datos, solo los mecanismos de modificación definidos por el ASSETS pueden realizar dicha tarea.
- El microservicio debe brindar una API de solo lectura (Métodos HTTP GET) de la información en la base de datos del ASSETS, la información de la base de datos no puede ser modificada por los usuarios del microservicio.
- El microservicio debe implementar un protocolo de seguridad para que solo usuarios asignados puedan acceder a la API
- El microservicio contara con un administrador que se encargara de asignar las credenciales y gestionar a los usuarios para que estos puedan consumir la API

2.2.3 – Representación del modelo del dominio

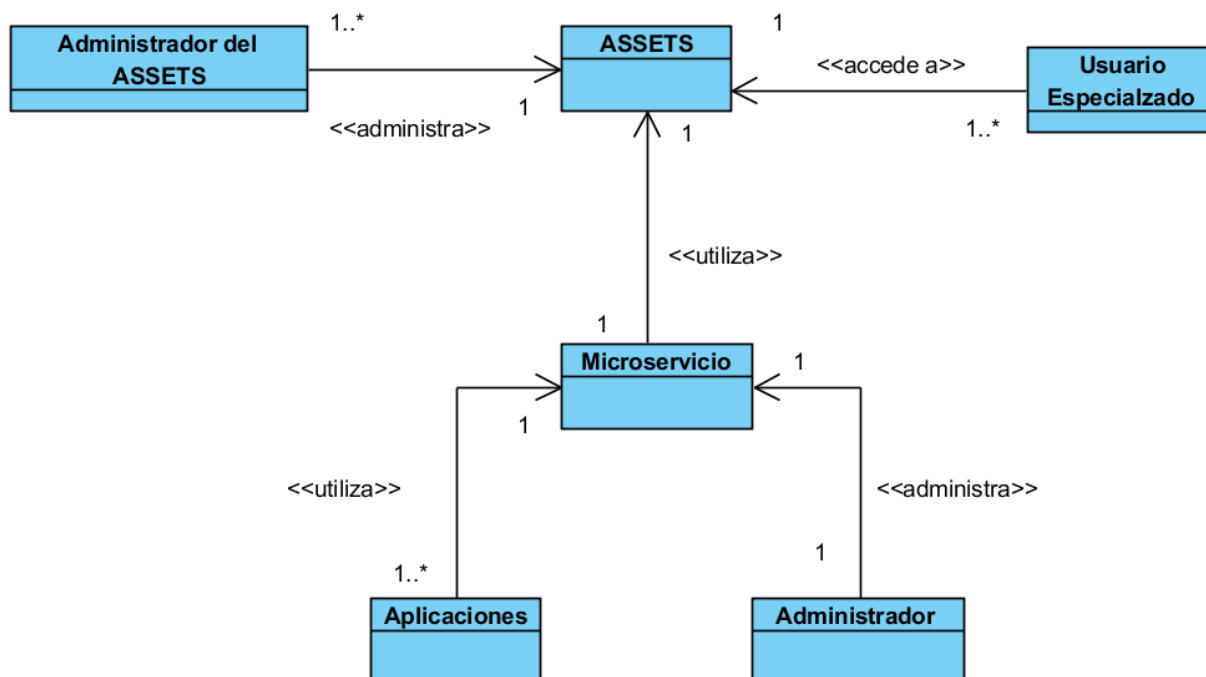


Figura 1. Diagramas de clases del modelo de objetos del dominio

2.3 – Requisitos

2.3.1 – Descripción del sistema propuesto

2.3.1.1 – Concepción general del sistema

El Sistema a construir constará de un microservicio que permitirá integrar la base de datos del ASSETS con aplicaciones que en un futuro se construyan en la UCf. También contará con una aplicación web para administrar las aplicaciones que puedan consumir los recursos y la documentación de cómo utilizar correctamente los mismos. El microservicio estará construido sobre NodeJs versión 17. El framework de desarrollo para el microservicio es ExpressJs, el cual es un marco de aplicación web de backend para Node.js, este se utilizará para implementar la capa de ruteo, el servidor y la configuración de la API.

TediousJs es la dependencia de producción que se encargara de conectarse a la base de datos y realizar las consultas, PassportJS es el módulo que ejecutara la autenticación a través de la estrategia JsonWebToken y Swagger se encargara de servir la documentación a través de la especificación dada en el archivo Yaml. Por último, la aplicación web será servida desde el servidor de archivos estáticos de Express y se construirá usando el framework Quasar el cual está desarrollado sobre Vuejs.

2.3.1.2 – Requerimientos funcionales

Los requerimientos funcionales permiten expresar una especificación más detallada de las responsabilidades del sistema que se propone. Ellos permiten determinar, de una manera clara, lo que debe hacer el mismo, a continuación, se listan los requisitos funcionales del sistema:

1. Mostrar información de todos los trabajadores
2. Buscar información de un trabajador
3. Filtrar búsqueda de trabajadores por expediente
4. Filtrar búsqueda de trabajadores por carnet de identidad
5. Filtrar búsqueda de trabajadores por nombre
6. Filtrar búsqueda de trabajadores por sus apellidos
7. Filtrar búsqueda de trabajadores por cuenta en moneda nacional
8. Filtrar búsqueda de trabajadores por color de la piel
9. Filtrar búsqueda de trabajadores por sexo
10. Filtrar búsqueda de trabajadores por municipio
11. Filtrar búsqueda de trabajadores por provincia
12. Filtrar búsqueda de trabajadores por nivel de escolaridad

13. Filtrar búsqueda de trabajadores por categoría
14. Filtrar búsqueda de trabajadores por grado científico
15. Filtrar búsqueda de trabajadores por tipo de contrato
16. Filtrar búsqueda de trabajadores por cargo
17. Autenticar
18. Cerrar sesión
19. Mostrar Usuario
20. Insertar Usuario
21. Modificar Usuario
22. Eliminar Usuario
23. Listar Usuarios

2.3.1.3 – Requerimientos no funcionales

Los requerimientos no funcionales son propiedades o cualidades que el producto debe tener, pero no pueden asociarse a ningún caso de uso, pero puede influir de cierta manera en alguno de ellos y su cumplimiento es desconocida por el usuario. En la herramienta propuesta, estos requerimientos se identifican principalmente con:

- **Apariencia o interfaz externa:** El diseño de la interfaz debe maximizar el contraste entre elementos para una correcta lectura. El sistema debe brindar información al usuario en todo momento del estado de los procesos que lleva a cabo.
- **Usabilidad:** El uso del producto debe resultar intuitivo, a la vez que debe brindar valor al usuario. Debe seguir las reglas de accesibilidad para brindarle una

experiencia grata a los usuarios. La presentación de los campos y formularios debe ser clara para su correcto empleo.

- **Rendimiento:** El sistema debe poder realizar las respuestas a las peticiones en el tiempo normal estándar, así como poder ejecutar de forma estable por largo periodo de tiempo.
- **Soporte:** El producto debe ser testeado para asegurar su calidad. Los desarrolladores deben estar disponibles para dar resolución a los errores que puedan aparecer en producción.
- **Portabilidad:** El microservicio debe poder ejecutarse sin problemas en cualquier máquina con sistema operativos Windows o distros de Linux, así como en servidores del mismo, el sitio web servido desde el microservicio debe poder ser ejecutado en la mayoría de navegadores sin importar la plataforma.
- **Políticos-culturales:** El sistema debe cumplir con las regulaciones políticas de la seguridad del departamento de tecnología educativa de la Universidad de Cienfuegos, así como asegurarse de no violar ninguna política de seguridad del software ASSETS.
- **Legales:** El producto no viola ninguna de las leyes legales de la entidad y no incumple las normas vigentes de la misma, además cumple todas las regulaciones dictadas por el país.
- **Confiability:** Se deberán de tomar las precauciones necesarias para prevenir fallos de cualquier tipo, ya sean por parte del sistema o del usuario, previniéndolos mediante validaciones y definiciones de reglas a seguir por el sistema en primera instancia.
- **Ayuda y documentación en línea:** El producto poseerá un archivo env.example en el repositorio para servir de ayuda en la configuración del servidor. Tendrá los enunciados de los derechos de autor, logos corporativos, iconos y otros elementos de la interfaz de usuario que deban mantenerse consistente con el resto de la

documentación, además de contar con un manual de usuario para el uso de la aplicación.

- **Software:**

- **Microservicio:** El servidor que ejecutara el sistema, debe tener Sistema Operativo Windows o Linux y contar con Node.js v.14 o superior.
- **Aplicación web:** cualquier navegador con soporte para ECMAScript 6, CSS 3 y HTML 5.

- **Hardware:**

- **Microservicio:** La PC donde será instalada, debe tener puerto Ethernet para conexión a la red. Las PC servidor deben tener, como mínimo, un procesador Dual Core a 2.4 GHz y 2 Gb de memoria RAM. En cuanto al espacio requerido en el disco duro, se deben reservar al menos 1 Gb para la instalación del microservicio.
- **Aplicación web:** los dispositivos que accedan al sitio, deben tener conexión a la red de la UCF; debe tener, como mínimo, las características para poder correr un navegador web.

- **Restricciones en el diseño y la implementación:** El lenguaje que va a ser usado en la implementación del microservicio va a ser JavaScript, utilizando el framework ExpressJs. Se va a usar TediousJS para la conexión con la base de datos y PassportJs para las estrategias de autenticación. La aplicación web se programará empleando el framework Quasar.
- **Seguridad:** La aplicación utilizará el protocolo de autenticación JsonWebToken para que así el acceso al microservicio solo lo puedan realizar usuarios permitidos.

2.3.2 – Modelo de casos de uso del sistema

El modelo de caso de uso identifica los requisitos del sistema en términos de la funcionalidad que debe existir para conseguir los objetivos establecidos por el usuario o para resolver un problema identificado por el usuario. En esta sección se pasa a describir los actores del sistema, se presenta el diagrama de caso de uso al igual que se describen todos los casos de uso del sistema.

2.3.2.1 – Actores del sistema

Nombre del actor	Descripción
Usuario	Aplicaciones de Terceros que quieran consumir los datos del microservicio como son la obtención de la información de un número de trabajadores, obtener información de un trabajador en específico y filtrar la información de los trabajadores por grado científico, cargo, sueldo, etc. Los requerimientos funcionales asociados a este actor son: 1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16
Administrador	Realiza la misma función que un usuario y además es el encargado de la gestión de los usuarios, agregar los mismos, cambiar el estado de sus cuentas, el api key de cada uno en el caso de pérdida y eliminar las cuentas cuando ya no se estén utilizando. Los requerimientos funcionales asociados a este actor son: 1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23

Tabla 1. Descripción de los actores del sistema

2.3.2.2 – Paquetes y sus relaciones

Los diagramas de paquetes son diagramas estructurales que se emplean para mostrar la organización y disposición de diversos elementos de un modelo en forma de paquetes. Un paquete es una agrupación de elementos UML relacionados, como diagramas, documentos, clases o, incluso, otros paquetes.

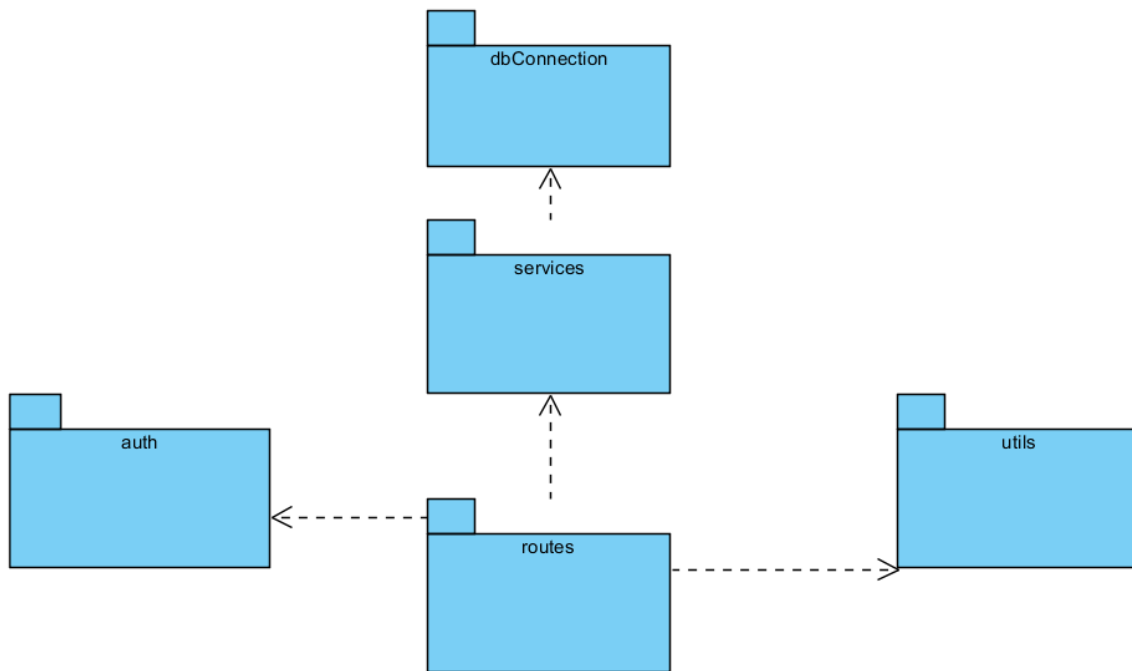


Figura 2. Diagramas de paquetes

2.3.2.3 – Diagramas de casos de uso del sistema

Un diagrama de casos de uso es una representación visual de las diferentes maneras y posibles escenarios de usar un sistema. Un caso de uso es la descripción de una acción o actividad, mientras que los diagramas de caso de uso es una descripción de las actividades que deberá realizar alguien o algo para llevar a cabo algún proceso. Un

diagrama de caso de uso puede incluir varios casos de uso y las relaciones entre casos de uso y las personas.

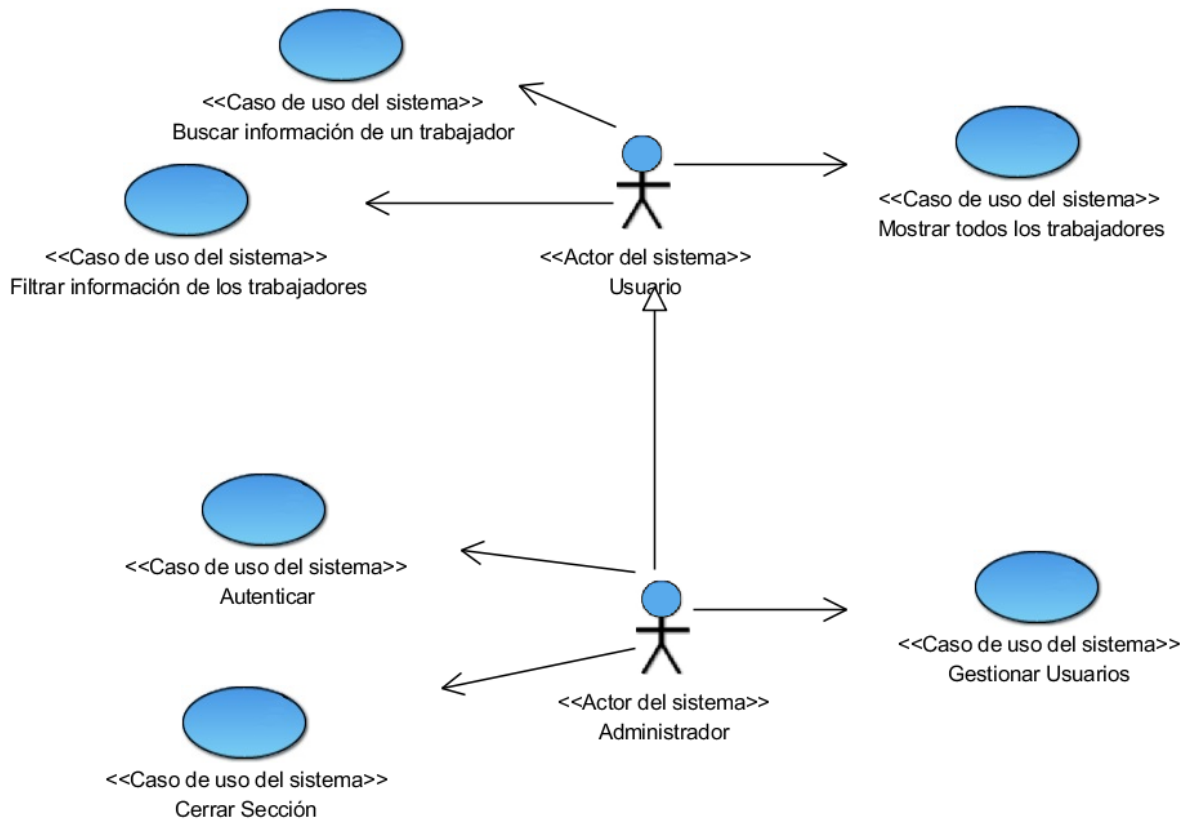


Figura 3. Diagramas de casos de uso del sistema

2.3.2.4 – Descripción de los casos de uso del sistema

Caso de uso	Autenticar
Actores	Administrador
Propósito	Permitir el acceso a las funcionalidades del sistema, teniendo en cuenta su rol como usuario.

Resumen	
El caso de uso se inicia cuando el Administrador desea ingresar al panel de administración, ingresa su correo y contraseña en el formulario de ingreso de la aplicación, a continuación, se verifica si los datos son correctos y en caso de serlo se da acceso al sistema, en caso contrario se mostrará un mensaje que notificará que algo va mal, denegando el acceso. Terminando así el caso de uso.	
Referencias	RF17
Precondiciones	El usuario administrador debe estar creado, para eso el microservicio debe ejecutarse la primera vez, lo cual crea al usuario administrador por defecto.
Post-condiciones	El administrador accede a la información de los usuarios que tienen acceso al microservicio.
Prototipo	(Ver Anexo 2.1)

Tabla 2. Descripción del caso de uso de sistema <Autenticar>

Caso de uso	Cerrar Sección
Actores	Administrador
Propósito	Cerrar la sesión del administrador para salir del sistema.
Resumen	

El caso de uso se inicia cuando el Administrador desea salir del panel de administración. Para cerrar sección debe pulsar en dicho botón en el menú, lo cual lo llevara a la página de inicio. Terminando así el caso de uso.	
Referencias	RF18
Precondiciones	El administrador debe haber iniciado sección en el sistema
Post-condiciones	El administrador no tiene acceso al panel de administración
Prototipo	(Ver Anexo 2.2)

Tabla 3. Descripción del caso de uso de sistema <Cerrar Sección>

Caso de uso	Gestionar Usuario
Actores	Administrador
Propósito	Permite gestionar los usuarios que acceden al microservicio y el estado de las cuentas.
Resumen El caso de uso se inicia cuando el Administrador necesita mostrar, insertar, modificar o eliminar un usuario del sistema. Se listan todos los usuarios en la página de inicio del panel de administración, si desea mostrar la información de uno de los usuarios debe dar click sobre la fila del usuario que quiera mostrar. Para mostrar se redirecciona a una vista de detalles donde podrá ver el nombre, correo, estado de la cuenta, tipo de	

cuenta y api key del usuario. El administrador puede modificar estado de la cuenta o cambiar api key, se muestra un formulario en dependencia de cada caso en donde se podrán modificar dichos campos. Para eliminar el administrador da en la opción correspondiente en la página de vista de detalles, lo cual abre un modal que al confirmar se elimina la cuenta. En el caso de insertar, se muestra el botón Crear Usuario, que abre un formulario en un modal, solicitando nombre, correo y el rol que va a tomar en el microservicio ya sea consumidor o administrador. Después de llenado, se verifica que el correo no se encuentre en la base de datos, de existir se muestra un mensaje de error, y en caso contrario se inserta. El caso de uso culmina con la actualización de los datos.

Referencias	RF19, RF20, RF21, RF22, RF23
Precondiciones	En caso de mostrar, modificar o eliminar debe existir el usuario.
Post-condiciones	Se ha o insertado, mostrado, modificado o eliminado un usuario
Prototipo	(Ver Anexo 2.3)

Tabla 4. Descripción del caso de uso de sistema <Gestionar Usuarios>

Caso de uso	Mostrar todos los Trabajadores
Actores	Usuario Administrador

Propósito	Permite mostrar la información de los trabajadores almacenada en la base de datos del ASSETS
Resumen El caso de uso se inicia cuando el usuario o el administrador a través de una interfaz con acceso a métodos http hace una llamada al microservicio en la url de trabajadores usando el api key que el administrador le dio en el momento de crearse la cuenta y pasando por query a la clave page un valor numérico que significa desde que número a hasta que otro número de trabajadores se van a solicitar. De estar la cuenta activada y el api key ser el correcto se devuelve la información de los trabajadores en formato JSON, en caso de haber un error se devuelve una petición dando los detalles del mismo en referencia con el estándar http	
Referencias	RF1
Precondiciones	El usuario debe estar registrado en el microservicio y tener una cuenta activa
Post-condiciones	Se ha accedido a la información de los trabajadores
Prototipo	(Ver Anexo 2.4)

Tabla 5. Descripción del caso de uso de sistema <Mostrar todos los trabajadores>

Caso de uso	Buscar Información de un Trabajador
--------------------	--

Actores	Usuario Administrador
Propósito	Permite mostrar la información de un trabajador en específico almacenado en la base de datos del ASSETS
Resumen El caso de uso se inicia cuando el usuario o el administrador a través de una interfaz con acceso a métodos http hace una llamada al microservicio en la url de trabajadores y a través de los parámetros de la url especifica el identificador del trabajador que requiere buscar. De estar la cuenta activada, el api key ser el correcto y existir el identificador se devuelve la información del trabajador en formato JSON, en caso de haber un error se devuelve una petición dando los detalles del mismo en referencia con el estándar http	
Referencias	RF2
Precondiciones	El usuario debe estar registrado en el microservicio y tener una cuenta activa
Post-condiciones	Se ha accedido a la información del trabajador con el identificador buscado
Prototipo	(Ver Anexo 2.5)

Tabla 6. Descripción del caso de uso de sistema <Buscar Información de un Trabajador>

Caso de uso		Filtrar Información de los Trabajadores
Actores	Usuario Administrador	
Propósito	Permite mostrar la información de un trabajador o varios trabajadores que cumplan un criterio en específico	
Resumen El caso de uso se inicia cuando el usuario o el administrador a través de una interfaz con acceso a métodos HTTP hace una llamada al microservicio en la url de filtros para trabajadores y a través de la query de la url especifica los criterios por el que quiere filtrar con los valores que desea buscar. De estar la cuenta activada, el api key ser el correcto se devuelve la información del trabajador en formato JSON, en caso de haber un error se devuelve una petición dando los detalles del mismo en referencia con el estándar http		
Referencias	RF3, RF4, RF5, RF6, RF7, RF8, RF9, RF10, RF11, RF12, RF13, RF14, RF15, RF16	
Precondiciones	El usuario debe estar registrado en el microservicio y tener una cuenta activa	
Post-condiciones	Se ha accedido a la información del trabajador o trabajadores que cumplen con los criterios de filtrado	

Prototipo	(Ver Anexo 2.6)
------------------	------------------

Tabla 7. Descripción del caso de uso de sistema <Filtrar Información de los Trabajadores>

2.4 – Construcción de la solución propuesta

En el presente epígrafe se realiza una descripción de la construcción de la solución propuesta. En esta descripción se ha utilizado el Diagrama de Clases del Diseño como artefacto propuesto por la Metodología RUP. Se plantean los diagramas del modelo lógico y físico de datos para una mayor comprensión del funcionamiento de la base de datos. Se describen los principios de diseño utilizados, mostrando ejemplos de cómo se presentan estos principios al usuario y la concepción general de la ayuda. También son descritas las consideraciones de codificación que se tuvieron en cuenta en la implementación de este sistema. Para describir los elementos fundamentales de la implementación se muestra el Diagrama de Implementación.

2.4.1 – Diagrama de clases del diseño

Un Diagrama de Clases de Diseño muestra la especificación para las clases software de una aplicación. Incluye la información de las clases con sus asociaciones y atributos, las interfaces, con sus operaciones y constantes, métodos, navegabilidad y dependencias.

A diferencia del Modelo Conceptual, un Diagrama de Clases de Diseño muestra definiciones de entidades software más que conceptos del mundo real. En la siguiente figura se muestra un ejemplo de Diagrama de Clases de Diseño sencillo.

Caso de Uso	Diagrama de clases del diseño
Autenticar	Ver Anexo 3.1
Cerrar sesión	Ver Anexo 3.2
Gestionar Usuarios	Ver Anexo 3.3
Mostrar todos los trabajadores	Ver Anexo 3.4

Buscar Información de un trabajador	Ver Anexo 3.5
Filtrar información de los trabajadores	Ver Anexo 3.6

Tabla 8. Diagrama de Clases del Diseño

2.4.2 – Diseño de la base de datos

2.4.2.1 – Modelo lógico de datos

El modelo lógico de datos es un modelo que no es específico de una base de datos, describe aspectos relacionados con las necesidades de una organización. Un modelo de datos lógico establece la estructura de los elementos de datos y las relaciones entre ellos. También describe los datos con el mayor detalle posible, independientemente de cómo se implementarán físicamente en la base de datos.

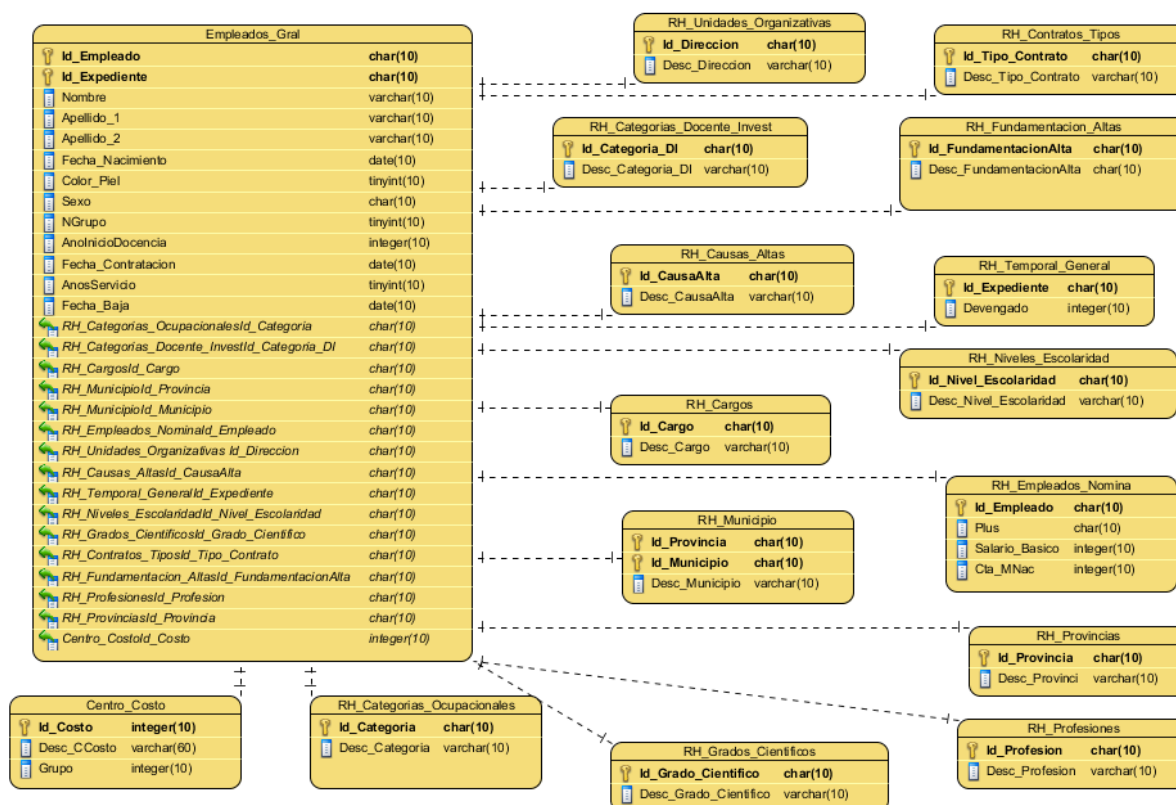


Figura 4. Diagramas de clases persistentes de la parte de la base de datos del ASSETS utilizada

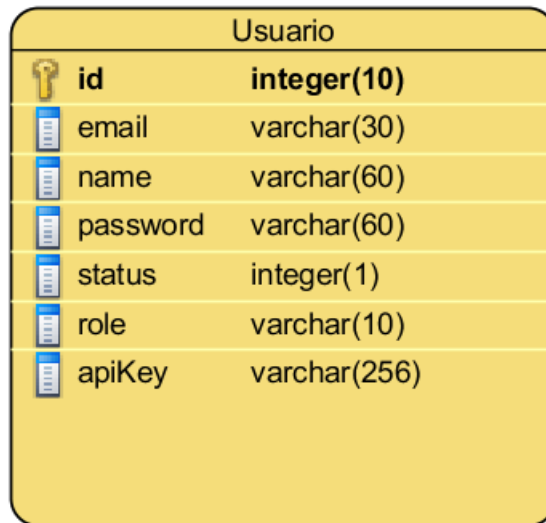


Figura 5. Diagramas de clases persistentes de la base de datos del microservicio

2.4.2.2 – Modelo físico de datos

El modelo físico de datos es un modelo específico de bases de datos que representa objetos de datos relacionales (por ejemplo, tablas, columnas, claves principales y claves externas) y sus relaciones. Un modelo físico de datos se define como una representación del modelo que se usará para la construcción de la base de datos.

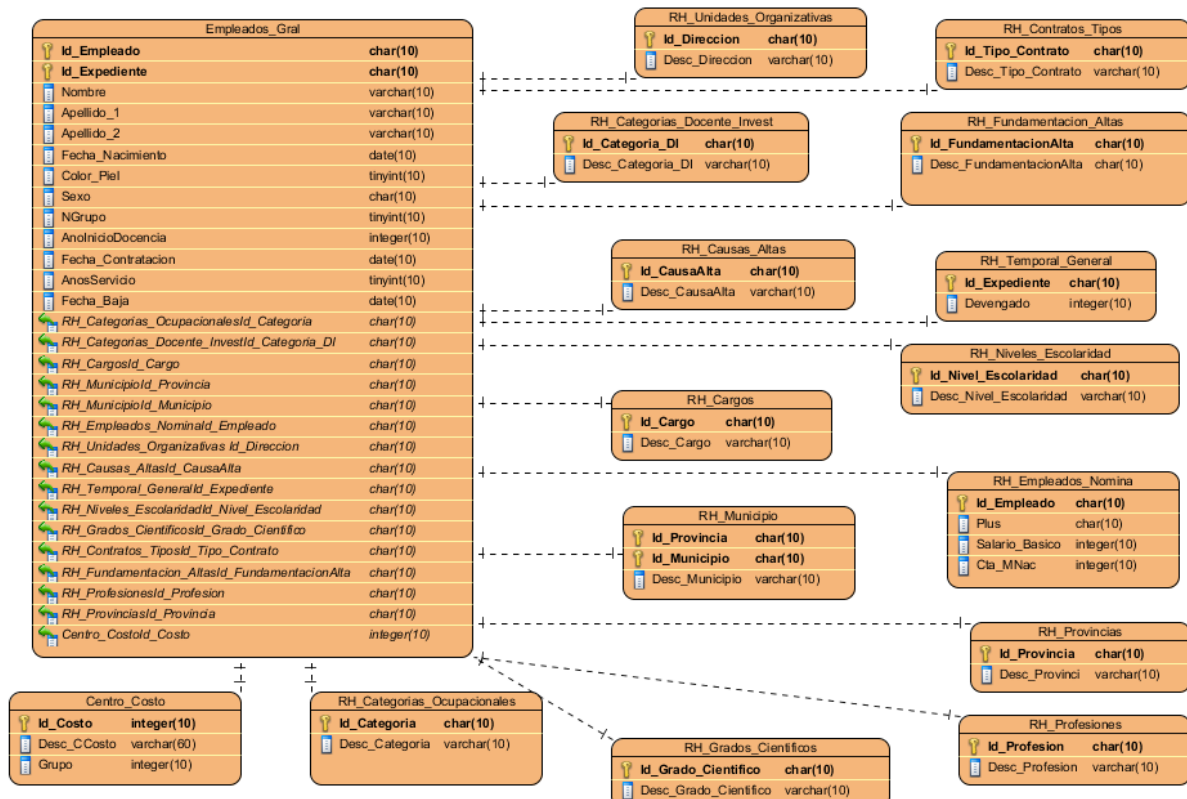


Figura 6. Diagramas del modelo físico de datos de la parte de la base de datos del ASSETS utilizada

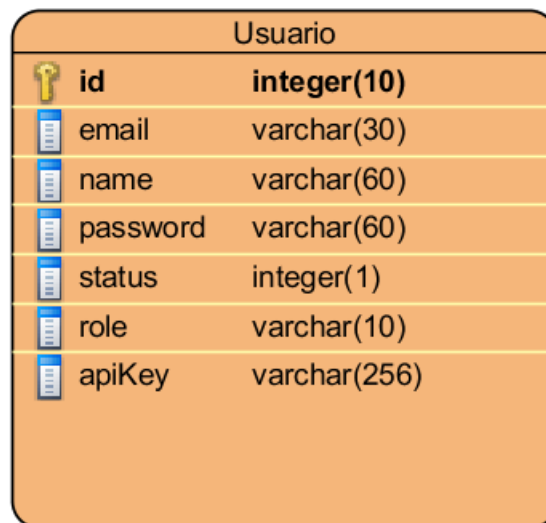


Figura 7. Diagramas del modelo físico de datos de la base de datos del microservicio

2.4.3 – Diagrama de implementación

Los diagramas de implementación muestran los aspectos de la implementación de un sistema, en los cuales se incluye la estructura del código fuente y su implementación en tiempo real con la estructura física del sistema. Es la etapa del desarrollo que describe la configuración del sistema para su ejecución en un ambiente del mundo real.

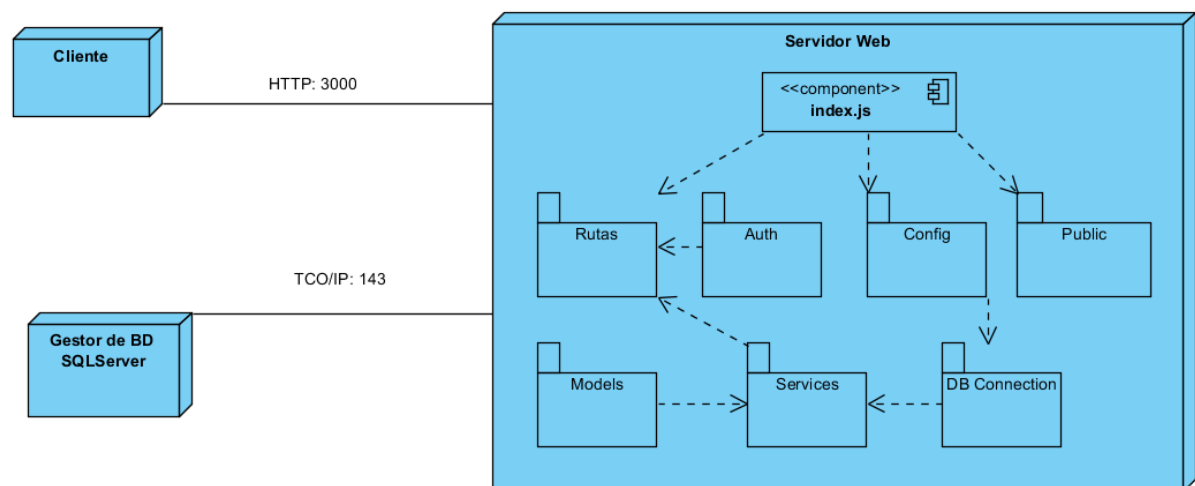


Figura 8. Diagrama de implementación

2.4.4 – Principios de diseño

El Software debe cumplir ciertos requisitos de calidad que solo se consiguen con un diseño consistente, el objetivo de esto es poder llegar a tener una comprensión total del sistema a través de los modelos y poder reconocer todos los requerimientos. La interfaz de la aplicación también debe seguir estándares, al igual que el tratamiento de errores.

2.4.4.1 – Estándares en la interfaz de la aplicación

Las interfaces del programa siguen varios de los principios del sistema de diseño de Google (Material Design) a la hora de la aplicación de los márgenes y pautas en el maquetado de las vistas. Los paneles, botones y elementos de navegación son consistentes en toda la aplicación y la interacción del usuario se hace mediante componentes de formularios simples. La fuente usada para texto y etiquetas fue Roboto y el tamaño cambia de acuerdo con el uso en cada momento. Los colores empleados en la aplicación están en consonancia con los colores del logo de la universidad, manteniendo una consistencia en el diseño.

2.4.4.2 – Tratamiento de errores

La aplicación limita sus opciones de entrada de datos, con listas desplegables que poseen valores ya definidos y checkboxes disminuyendo el riesgo de que el usuario introduzca información errónea. En los casos en que este tiene que ingresar información, los campos se validan. Cuando ocurren errores se muestran mediante mensajes de error que ayudan al usuario a estar consiente en todo momento del estado en que se encuentra el sistema.

2.4.4.3 – Concepción General de la ayuda

El usuario final contará con un manual que brindará información acerca del uso del panel de administración y de la documentación. El microservicio cuenta con una documentación web en donde se podrá acceder a toda la información y vías de uso de los recursos del microservicio y podrá probar las funcionalidades del mismo.

2.5 – Conclusiones

La identificación de las distintas entidades que conforman la problemática nos brinda la información inicial necesaria para pasar al siguiente paso de la ingeniería de software. En este capítulo se definieron las reglas que deben cumplirse para el correcto funcionamiento del sistema, y se presentó el modelo del Dominio, el cual nos ayuda en

la comprensión del sistema que se desea desarrollar. También se realizó la captura de requisitos definiendo los requerimientos funcionales y no funcionales.

A partir de estos requerimientos se pudo dar paso al modelado del sistema, identificando los actores del sistema y los casos de usos existentes, además se hizo una descripción detallada de cada caso de uso del sistema. Ya una vez analizados los distintos diagramas de clases del diseño, visto la interacción entre estos, se tiene una idea exacta de cómo es el flujo de la información en la aplicación, El modelo lógico de la base de datos nos ayuda a comprender la sección de la base de datos, la cual estamos accediendo, dado que la base de datos del el ASSETS es más extensa, pero los requerimientos del sistema nos han delimitado su uso.

3 – Estudio de factibilidad y validación de la solución

3.1 – Introducción

En este capítulo veremos la planificación por casos de usos en donde se analizará el factor de peso de los actores del sistema as como el factor de peso de los casos de uso. Se calculará los puntos de Casos de Uso Ajustados, el Factor de Complejidad Técnica, se determinarán los costos al igual que se podrán observar los beneficios tangibles e intangibles.

3.2 – Estudio de factibilidad

El estudio de factibilidad es el análisis que lleva a cabo una empresa para determinar si el negocio que se propone será bueno o malo, y cuáles serán las estrategias que se deben desarrollar para que sea exitoso. En el desarrollo de software este proceso permite tener una noción del beneficio que se alcanzara y el tiempo a dedicar en la implementación del sistema.

3.2.1 – Planificación por casos de usos

3.2.1.1 Factor de peso de los actores sin ajustar

Este valor se calcula mediante un análisis de la cantidad de Actores presentes en el sistema y la complejidad de cada uno de ellos. La complejidad de los Actores se establece teniendo en cuenta, en primer lugar, si se trata de una persona o de otro sistema y, en segundo lugar, la forma en la que el actor interactúa con el sistema.

Actor	Tipo de Actor	Factor de Peso
Administrador	Complejo	3
Usuario	Complejo	3

Tabla 9. Factor de peso de los actores del sistema

Se tienen 2 actores del sistema con clasificación Complejo, por lo que se le aplica como factor de peso 3.

$$UAW = (Cantidad\ de\ actores) * Peso$$

$$UAW = 2 * 3 = 6$$

UAW: Factor de peso de los actores sin ajustar

3.2.1.2. Factor de peso de los casos de uso

Este valor se calcula mediante un análisis de la cantidad de Casos de Uso presentes en el sistema y la complejidad de cada uno de ellos. La complejidad de los Casos de Uso se establece teniendo en cuenta la cantidad de transacciones efectuadas en el mismo, donde una transacción se entiende como una secuencia de actividades atómica, es decir, se efectúa la secuencia de actividades completa, o no se efectúa ninguna de las actividades de la secuencia.

Caso de uso	Transacciones	Clasificación
Gestionar Usuarios	5	Medio
Autenticar	1	Simple
Cerrar Sección	1	Simple
Filtrar información de los trabajadores	14	Complejo
Mostrar todos los trabajadores	1	Simple
Buscar información de un trabajador	1	Simple

Tabla 10. Factor de peso de los Casos de Uso

6 casos de uso, 4 simples, 1 medios, 1 complejo

$$UUCW = 4 * 5 + 1 * 10 + 1 * 15 = 45$$

UUCW: Factor de peso de los casos de uso sin ajustar

Por tanto:

$$UUCP = UAW + UUCW = 6 + 45 = 51$$

UUCP: Puntos de casos de uso sin ajustar

3.2.1.3 Calcular los puntos de Casos de Uso Ajustados

Una vez que se tienen los Puntos de Casos de Uso sin ajustar, se debe ajustar este valor mediante la siguiente ecuación:

$$UCP = UUCP * TCF * EF$$

UCP: Puntos de Casos de Usos Ajustados

UUCP: Puntos de Casos de Uso sin Ajustar

TCF: Factor de Complejidad Técnica

EF: Factor de Ambiente

3.2.1.4. Factor de Complejidad Técnica

Este coeficiente se calcula mediante la cuantificación de un conjunto de factores que determinan la complejidad técnica del sistema. Cada uno de los factores se cuantifica con un valor de 0 a 5, donde 0 significa un aporte irrelevante y 5 un aporte muy importante.

Factor	Descripción	Peso	Valor	Comentario	(Peso _i * Valor _i)
T1	Sistema distribuido	2	4	El sistema es distribuido	8
T2	Objetivos de performance o tiempo de respuesta	1	2	La velocidad es limitada por las entradas provistas por el usuario	2
T3	Eficiencia del usuario final	1	2	Escasas restricciones de eficiencia	2
T4	Procesamiento interno complejo	1	0	No hay cálculos Complejos	0

T5	El código debe ser reutilizable	1	2	Se implementa algo de código reutilizable para el aprovechamiento del mismo en toda la aplicación.	2
T6	Facilidad de instalación	0.5	1	Se requiere conocimientos de consola de comandos	0.5
T7	Facilidad de uso	0.5	4	El sistema es fácil de usar.	2
T8	Portabilidad	2	4	El sistema es portable	8
T9	Facilidad de cambio	1	2	El sistema ha sido concebido pensando en la incorporación de nuevas funcionalidades	2
T10	Concurrencia	1	2	Hay buena concurrencia.	2
T11	Incluye objetivos	1	3	Seguridad normal	3

	especiales de seguridad				
T12	Provee acceso directo a terceras partes	1	0	No hay acceso	0
T13	Se requieren facilidades especiales de entrenamiento a los usuarios	1	1	Sistema fácil de usar.	1
Total					32.5

Tabla 11. Factor de Complejidad Técnica

$$TCF = 0.6 + 0.01 * \sum (Peso * Valor)$$

$$TCF = 0.6 + 0.01 * 32.5 = 0.925$$

TCF: Factor de Complejidad Técnica

3.2.1.5. Factor Ambiente

Las habilidades y el entrenamiento del grupo involucrado en el desarrollo tienen un gran impacto en las estimaciones de tiempo. Estos factores son los que se contemplan en el cálculo del Factor de ambiente. El cálculo del mismo es similar al cálculo del Factor de complejidad técnica, es decir, se trata de un conjunto de factores que se cuantifican con valores de 0 a 5.

Factor	Descripción	Peso	Valor	Comentario	(Peso_i * Valor_i)
E1	Familiaridad con el modelo de proyecto utilizado	1.5	3	Se está bastante familiarizado con el modelo, pero faltan algunos detalles.	4.5
E2	Experiencia en la aplicación	0.5	2	Se ha trabajado en ello por lo que existe cierta experiencia	1
E3	Experiencia en orientación a objetos	1	4	Hay experiencia en programación orientado a objetos	4
E4	Capacidad del analista líder	0.5	3	El analista está capacitado	1.5
E5	Motivación	1	4	Hay motivación	4
E6	Estabilidad de los requerimientos	2	3	Requerimientos estables	6
E7	Personal a medio tiempo.	-1	1	El personal no es a tiempo completo	-1

E8	Dificultad del lenguaje de programación	-1	5	Se usará lenguaje JavaScript del cual se tiene un nivel medio	-5
Total					15

Tabla 12. Factor Ambiente

$$EF = 1.4 - 0.03 * \sum (Peso * Valor)$$

$$EF = 1.4 - 0.03 * 15 = 0.95$$

Por tanto:

$$UUCP = 51$$

$$TCF = 0.925$$

$$EF = 0.95$$

$$UCP = UUCP * TCF * EF$$

$$UCP = 51 * 0.925 * 0.95 = 44.82$$

3.2.1.6. Esfuerzo de desarrollo

Este cálculo se realiza con el fin de tener una aproximación del esfuerzo, pensando solo en el desarrollo según las funcionalidades de los casos de uso.

$$E = UCP * FC$$

E: Esfuerzo estimado en horas/hombre

FC: Factor de conversión

Factor de conversión

Total EF = Cantidad EF < 3 (entre E1 –E6) + Cantidad EF > 3 (entre E7, E8)

Total EF = 2 + 0

Total EF = 2

Por tanto:

$FC = 20 \text{ horas-persona}$ (porque Total EF <= 2)

$E = 44.82 * 20$

$E = 896.4 \text{ horas-persona}$

3.2.1.7. Distribución y Estimación de esfuerzo

Finalmente, para una estimación más completa de la duración total del proyecto, hay que agregar a la estimación del esfuerzo obtenida por los Puntos de Casos de Uso, las estimaciones de esfuerzo de las demás actividades relacionadas con el desarrollo de software.

Para ello se puede tener en cuenta el siguiente criterio, que estadísticamente se considera aceptable. El criterio plantea la distribución del esfuerzo entre las diferentes actividades de un proyecto, según la siguiente aproximación.

ACTIVIDAD	ESFUERZO (%)	VALOR (HORAS-HOMBRE)
Análisis	10	224.1
Diseño	20	448.2
Implementación	40	<i>896.4</i>

Prueba	15	336.15
Sobrecarga	15	336.15
Total	100	2241

Tabla 13. Estimación de Esfuerzo

3.2.1.8. Tiempo de desarrollo

Como se calculó anteriormente el esfuerzo realizado es de 6996.6 Horas-Hombre, de aquí que se considere que un trabajador emplea en su trabajo 24 días de cada mes y que *labora diariamente 8 horas*.

$$TD(meses) = \frac{E}{CH * CDL * CHL}$$

$$TD(meses) = \frac{2241}{1 * 24 * 8}$$

$$TD(meses) = 11.67 \approx 12$$

TD: Tiempo de desarrollo

CH: Cantidad de hombres

CDL: Cantidad de días laborales por mes

CHL: Cantidad de horas laborables por día

3.2.2 – Determinación de los costos

Tomando como salario promedio mensual \$4200.00 (Salario básico de un especialista B en Ciencias Informáticas).

$$C = E * CHH + \text{Otros Gastos}$$

C: Costo del proyecto

CHH: Costo por Horas-Hombre

$$CHH = K * THP$$

K: Coeficiente que tiene en cuenta los costos indirectos (1.5 y 2.0)

THP: Tarifa Horario Promedio. El salario promedio de las personas que trabajan en el proyecto dividido entre 160 Horas-Hombre.

Entonces:

$$K = 2.0$$

$$THP = \$ 4200 / 160 = \$ 26.25 / \text{horas-persona}$$

$$CHH = 2.0 * \$ 26.25 / \text{Horas-Hombre} = \$ 52.5 / \text{horas-persona}$$

$$C = 2241 \text{ horas-persona} * \$ 52.5 = \$ 117\,652.5$$

Por todos los cálculos anteriores se determinó que el costo de la realización de este proyecto es de \$ 117 652.5 MN.

3.2.3– Beneficios tangibles e intangibles

Un beneficio tangible es aquel que afecta directamente al resultado de la empresa. Los beneficios tangibles son las ventajas económicas cuantificables que obtiene la empresa a través del uso del software y se pueden estimar en pesos, recursos y tiempo ahorrado.

Como beneficio tangible de este proyecto se encuentra:

- El desarrollo de la aplicación tiene un costo de \$ 117 652.5 MN, lo que contribuye a la reducción de gastos de la empresa por concepto de no pagar por un software extranjero.

Un beneficio intangible es cualquier cosa que sea difícil de medir. Los beneficios intangibles son beneficios difíciles de cuantificar, pero por ello no dejan de ser menos importantes. Dentro de los beneficios intangibles del trabajo se destacan:

- El software permite crear una vía de comunicación entre las aplicaciones que necesiten acceder a la información, posibilidad que antes no poseían.
- Disminuye el tiempo de espera para poder acceder a información de interés sobre los trabajadores del centro.
- Garantiza un acceso fácil a información crucial para el desarrollo de distintos procesos de gestión.
- Garantiza la seguridad y acceso a la información.

3.2.4 – Análisis de costos y beneficios

El proyecto no presenta costo para el departamento de tecnología educativa, pero su desarrollo está valorado en \$ 117 652.5MN, por lo que el software ha reportado un ahorro económico a la universidad, no solo por sus prestaciones, sino también por su desarrollo. Un producto con las mismas prestaciones, de encargarse la tarea a una empresa de desarrollo de software, implicaría un costo para la entidad.

La utilización de este nuevo sistema permitirá a otras aplicaciones de la universidad, acceder a los recursos para utilizarlos en sus procesos, implica un ahorro considerable del tiempo invertido en la gestión y control de esta información. Para la realización de este sistema no fue necesaria una inversión en los medios técnicos. Por esto y lo anteriormente planteado, se concluye que el Microservicio para la gestión de la información en el ASSETS de la Universidad de Cienfuegos es factible.

3.3 – Validación de la solución

Las pruebas de software tienen por objetivo probar que los sistemas desarrollados, cumplan con las funciones específicas para las cuales han sido creados. A este tipo de

pruebas se les denomina también pruebas de comportamiento o pruebas de caja negra, básicamente el enfoque de este tipo de prueba se basa en el análisis de los datos de entrada y en los de salida, esto generalmente se define en los casos de prueba preparados antes del inicio de las pruebas. A continuación, se aplica el método para generar casos de prueba funcional a partir de los casos de uso del sistema del software propuesto.

3.3.1 Diseño de Pruebas Funcionales

Un aspecto crucial en el control de calidad del desarrollo de software son las pruebas y, dentro de estas, las pruebas funcionales, en las cuales se hace una verificación dinámica del comportamiento de un sistema, basada en la observación de un conjunto seleccionado de ejecuciones controladas o casos de prueba. Para hacer pruebas funcionales se requiere una planificación que consiste en definir los aspectos a chequear y la forma de verificar su correcto funcionamiento, punto en el cual adquieren sentido los casos de prueba.

Caso de Prueba	Autenticar
Descripción	El administrador del microservicio puede autenticarse en el panel de administracion, para hacer uso de las funcionalidades que brinda el mismo.
Validación Al hacer click sobre el botón “Ingresar”: • Los campos Correo y Contraseña no pueden estar vacíos. • El campo Correo debe referenciar a un esquema de correo válido. • El campo Correo debe existir en el sistema.	

- El campo Contraseña debe coincidir con el del Usuario especificado.
- Debe mantenerse una conexión al servidor

Si la validación no tuvo éxito, en el caso del campo correo mostrara una alerta como la de la Figura 15 y en el caso de haber un error en la información o en el procesamiento de la petición mostrara un mensaje como el de la Figura 16.

The image shows a web interface for 'Microservicio ASSETS' with the subtitle 'Panel de Administración'. It features a login form with two input fields: 'Correo' (Email) and 'Contraseña' (Password). The 'Correo' field contains the text 'admingmail.com'. A red error message box is displayed over the form, stating: 'Incluye un signo "@" en la dirección de correo electrónico. La dirección "admingmail.com" no incluye el signo "@".' (Include an '@' symbol in the email address. The address 'admingmail.com' does not include the '@' symbol.). The 'Contraseña' field is masked with dots and has a toggle icon. Below the fields is a blue button labeled 'INGRESAR'.

Figura 9. Caso de Prueba <<Autenticar>> mensaje de alerta en el campo correo

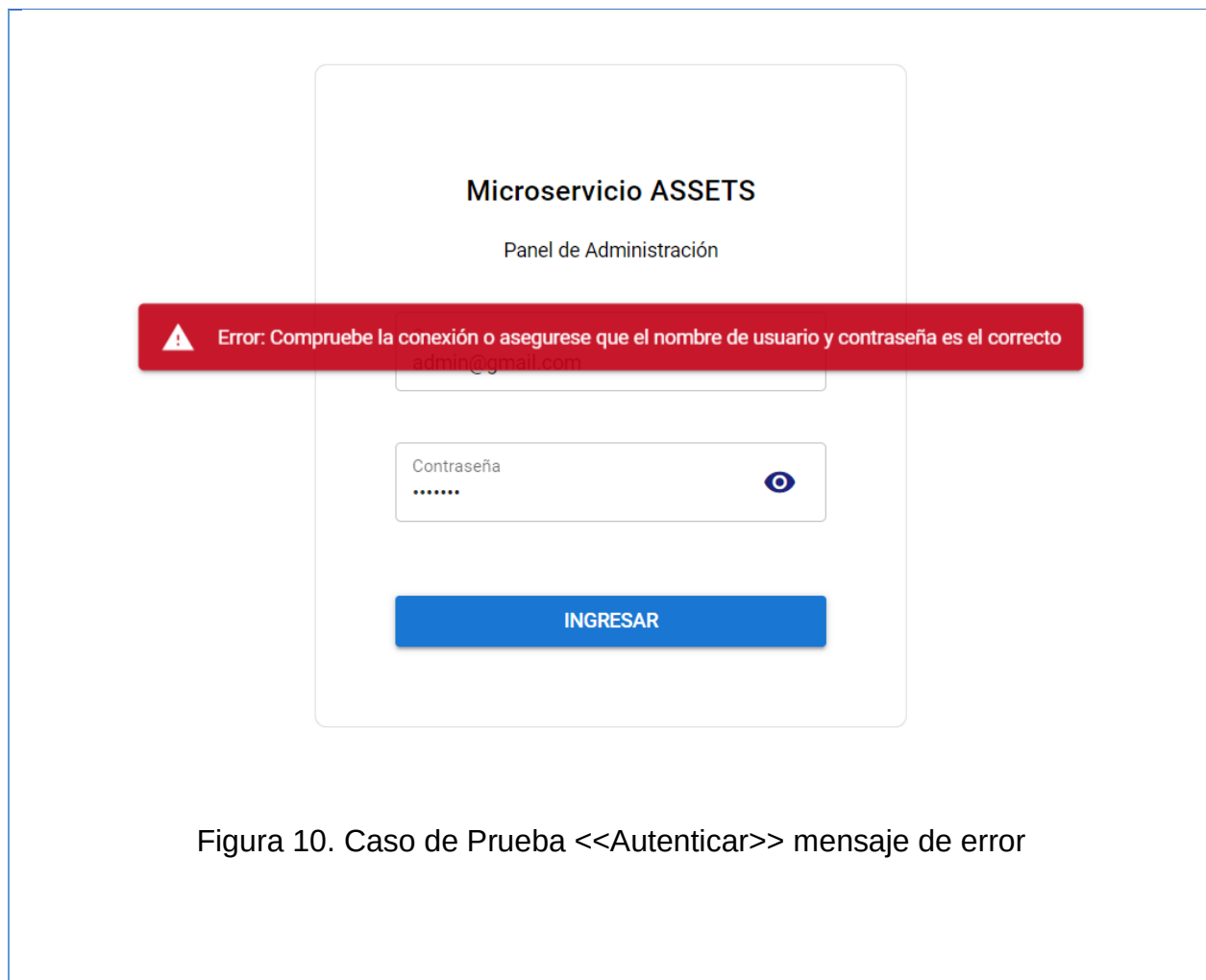


Figura 10. Caso de Prueba <<Autenticar>> mensaje de error

Caso de Prueba Registrar Usuario	
Descripción	El administrador puede registrar nuevos usuarios que tendrán acceso a los recursos que el microservicio pone a disposición.
Validación	
Al hacer click sobre el botón “Crear Usuario”:	

- Los campos Nombre, Correo, no pueden estar vacíos.
- El campo Correo debe referenciar a un esquema de correo válido.
- El campo Correo no puede coincidir con alguno existente.
- Debe mantenerse una conexión al servidor

Si la validación no tuvo éxito, mostrara un mensaje de error como el de la Figura 18.

The screenshot displays the 'Microservicio ASSETS' user management interface. A modal titled 'Panel de Creación de Usuario' is centered on the screen, prompting the user to 'Llene los campos para crear un nuevo usuario'. The modal contains the following fields and controls:

- Nombre:** A text input field.
- Correo:** A text input field with an information icon (i) to its right.
- Rol:** A dropdown menu currently showing 'Usuario'.
- CREAR USUARIO:** A dark blue button with a user icon.
- CANCELAR:** A text link below the create button.

The background interface is dimmed and shows a 'Gestión de Usuarios' section with a table of users and an 'AGREGAR USUARIO +' button.

Nombre	Correo	Rol	Api Key
apiDocsUser	testg...	...	eyJhbGciOi...
Yaal	Yaal@...	...	eyJhbGciOi...
a		...	eyJhbGciOi...
Palmero	Palmer...	...	eyJhbGciOi...
admin	admin@...	...	eyJhbGciOi...

Figura 11. Caso de prueba <<Registrar Usuario>> Vista de Agregar Usuarios

The screenshot displays a web form titled "Panel de Creación de Usuario". Below the title is the instruction "Llene los campos para crear un nuevo usuario". The form contains three input fields: "Nombre" with the value "Yaal", "Correo" with the value "Yaal@gmail.com", and "Rol" with the value "Usuario". A red error banner is overlaid on the form, displaying a warning icon and the text "Error: Revise su conexión o asegurese que toda la inforamción esta correcta". At the bottom of the form are two buttons: "CREAR USUARIO" (with a user icon) and "CANCELAR".

Panel de Creación de Usuario

Llene los campos para crear un nuevo usuario

Nombre
Yaal

Correo
Yaal@gmail.com

Rol
Usuario

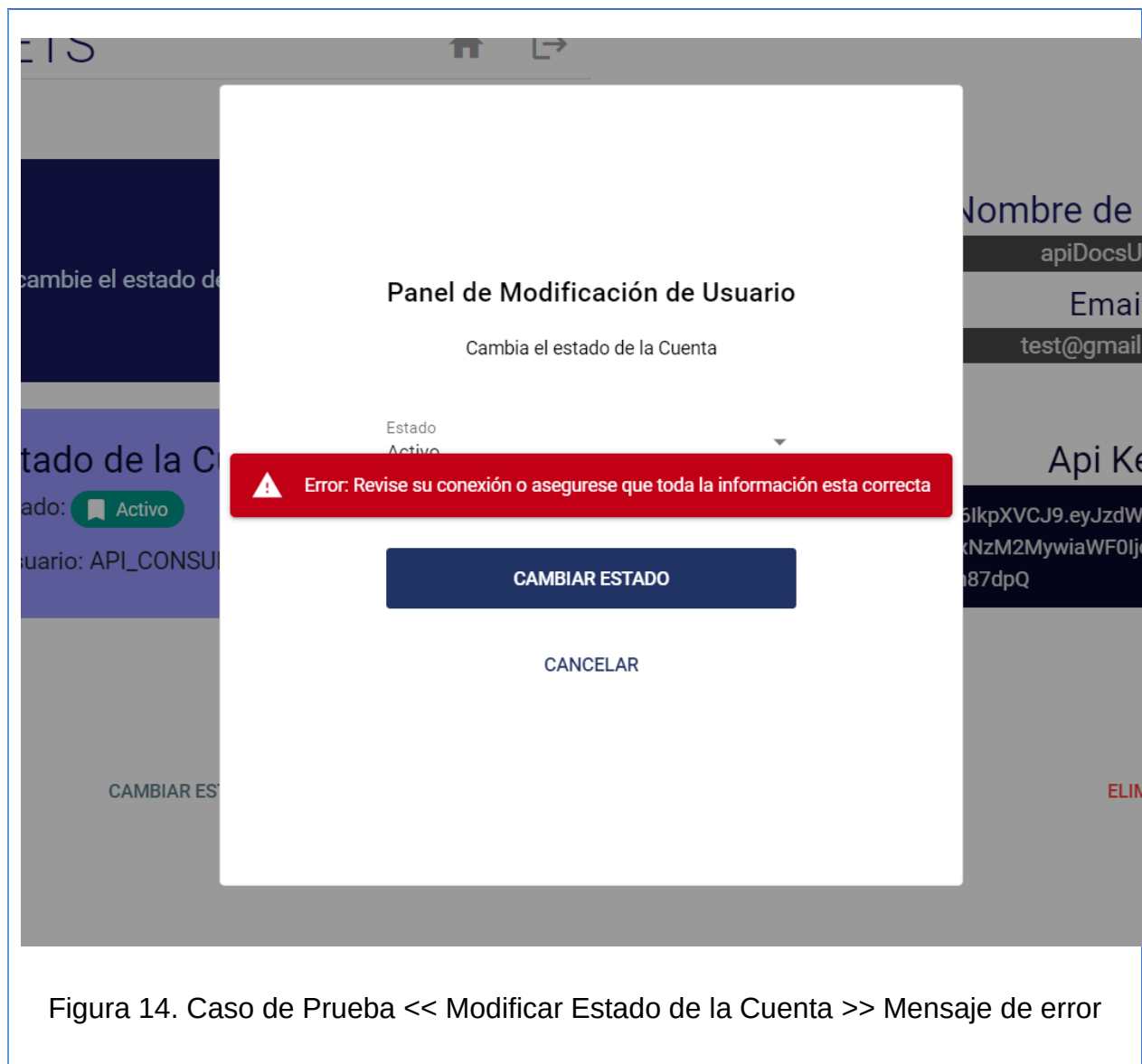
Error: Revise su conexión o asegurese que toda la inforamción esta correcta

CREAR USUARIO

CANCELAR

Figura 12. Caso de prueba <<Registrar Usuario>> mensaje de error

Caso de Prueba	Modificar Estado de la Cuenta
Descripción	El administrador puede desactivar o activar la cuenta de un usuario.
Validación Al hacer click sobre el botón “Cambiar Estado”: • Debe de haber un estado seleccionado en la casilla del select • Debe mantenerse una conexión al servidor Si la validación no tuvo éxito, se mostrará un mensaje de error (Figura 20).	
	
Figura 13. Caso de Prueba << Modificar Estado de la Cuenta >> Modal	



Caso de Prueba	Modificar Api Key
Descripción	El administrador puede cambiar el api key de un usuario.
Validación	

Al hacer click sobre el botón “Cambiar Estado”:

- Debe mantenerse una conexión al servidor

Si la validación no tuvo éxito, se mostrará un mensaje de error (Figura 22).

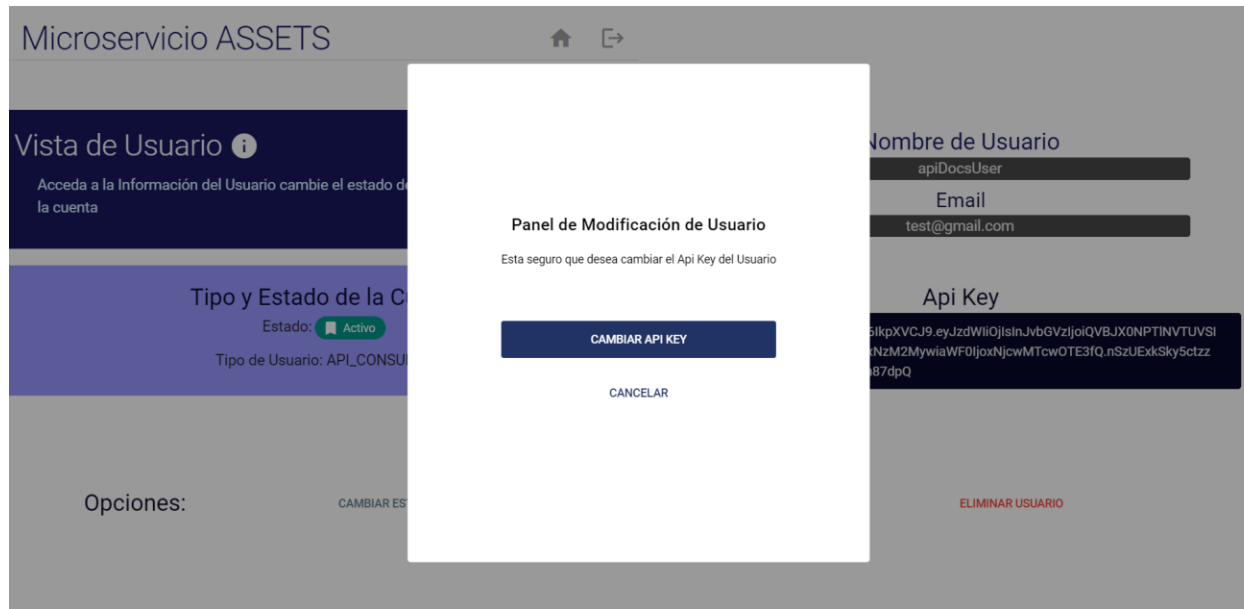


Figura 15. Caso de Prueba << Modificar Api Key >> Modal

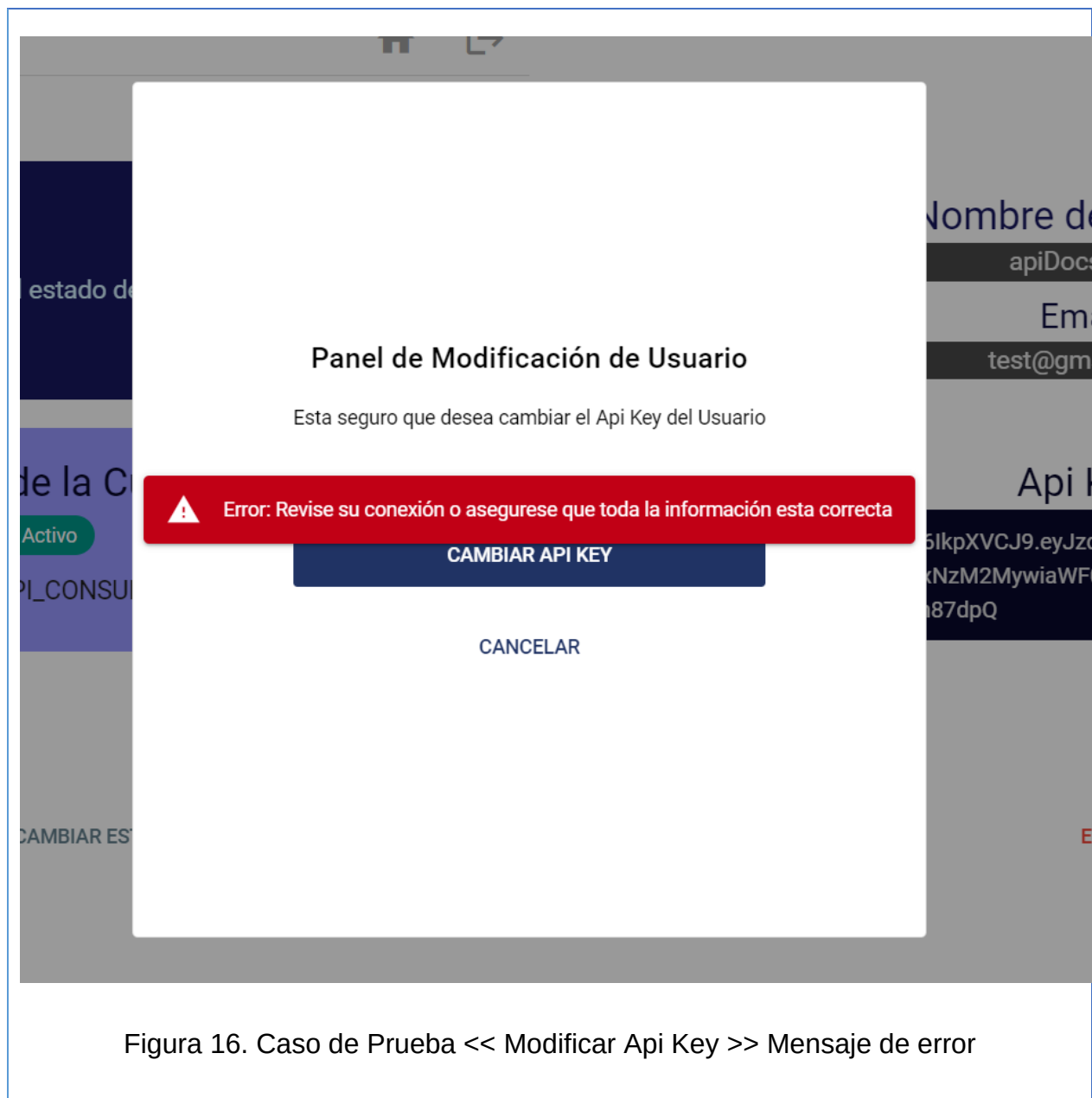


Figura 16. Caso de Prueba << Modificar Api Key >> Mensaje de error

Caso de Prueba	Eliminar Usuario
Descripción	El administrador puede eliminar la cuenta de un usuario.

Validación

Al hacer click sobre el botón “Eliminar Cuenta”:

- Debe mantenerse una conexión al servidor

Si la validación no tuvo éxito, se mostrará un mensaje de error (Figura 24).

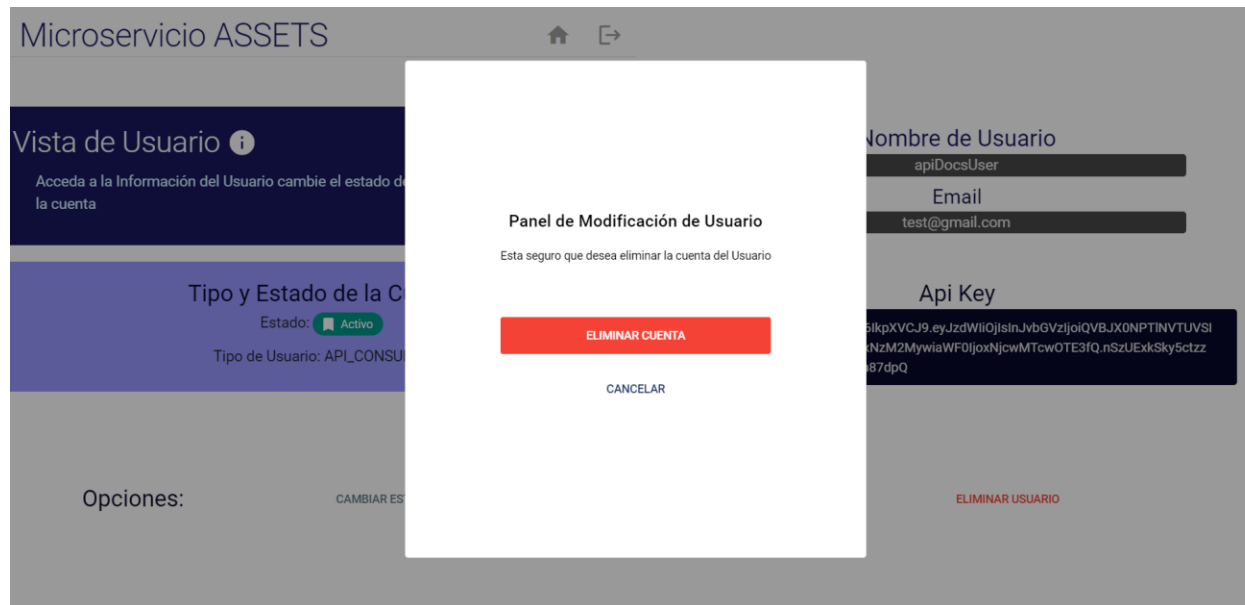


Figura 17. Caso de Prueba << Eliminar Usuario >> Modal



3.4 – Conclusiones

En este capítulo se realizó el estudio de factibilidad basado en los puntos de caso de uso al sistema arrojando el tiempo de duración del sistema y el costo del mismo además de realizarle las pruebas funcionales, esto permitió determinar la calidad al sistema. La realización del cálculo de factibilidad económica para el sistema informático propuesto

estimó un tiempo de 12 meses para su desarrollo y un costo de \$ 117 652.5 MN aproximadamente. A partir de los costos y beneficios que trajo consigo la implementación del sistema para la entidad, se pudo concluir que la ejecución del proyecto es factible. La realización de pruebas funcionales redujo los errores en las entradas de datos, logrando validar el sistema ante este tipo de errores y garantizando una mayor confiabilidad de la información.

Conclusiones

Durante el análisis del objeto de estudio de esta investigación se pudo constatar que el proceso de gestión de la información en el ASSETS de la UCf es complejo y este sistema no provee un método de acceso a la información que el mismo maneja para aplicaciones de otras partes. El desarrollo del microservicio trae solución a la problemática planteada, permitiendo que otras aplicaciones puedan integrarse con el ASSETS obteniendo acceso a la información almacenado en este sistema.

Tomando en consideración los objetivos trazados al inicio del trabajo, Al término de este se concluye:

- A partir de las entrevistas al administrador del ASSETS, así como del estudio de la documentación y manuales de usuario de este sistema y otros utilizados en la UCf, se pudo identificar la falta de un mecanismo de integración del sistema con otras aplicaciones.
- La elaboración y puesta en explotación del microservicio que brinda información del sistema ASSETS, permitió una mejor integración de esta información a disposición de terceros, lo cual contribuirá a impulsar el proceso de transformación digital en la Universidad de Cienfuegos.
- El diseño de los casos de prueba funcional permitió una verificación del comportamiento del sistema, garantizando una adecuada validación del software.
- Se utilizó el método de estimación de Puntos de Casos de Usos para determinar la factibilidad, Los resultados confirmaron la factibilidad de la construcción del sistema; se estimó un tiempo de 12 meses para su desarrollo y un costo de \$ 117 652.5MN aproximadamente.

Recomendaciones

A pesar de haberse cumplido los objetivos trazados para la realización del trabajo de diploma, esta propuesta es la primera etapa de un proyecto más abarcador. Se recomienda:

- Poner a prueba todas las funcionalidades del sistema propuesto durante un período prolongado de tiempo, permitiendo esto comprobar dichas funcionalidades de forma práctica y poder entonces detectar posibles mejoras a realizar en el futuro.
- Continuar el estudio de los procesos que se desarrollan en el ASSETS, con el objetivo de ampliar las funcionalidades de la aplicación según las nuevas necesidades que puedan surgir.
- Implementar nuevas funcionalidades a la capa de gestión del microservicio como puede ser la agregación de EndPoint dinámicos.

Referencias bibliográficas

- [1] J. Ríos Ortega y J. Ríos Ortega, «El concepto de información: dimensiones bibliotecológica, sociológica y cognoscitiva», *Investigación bibliotecológica*, vol. 28, n.º 62, Art. n.º 62, abr. 2014.
- [2] J. Carrión, «Diferencia entre dato información y conocimiento», abr. 2017, Accedido: 7 de noviembre de 2022. [En línea]. Disponible en: <http://148.202.167.116:8080/xmlui/handle/123456789/869>
- [3] R.- ASALE y RAE, «información | Diccionario de la lengua española», «*Diccionario de la lengua española*» - Edición del Tricentenario. <https://dle.rae.es/información> (accedido 7 de noviembre de 2022).
- [4] Y. C. Montejo y H. P. Sousa, «Gestión documental, Gestión de información y Gestión del conocimiento: nociones e interrelaciones», *Bibliotecas. Anales de investigación*, n.º 0, Art. n.º 0, abr. 2021.
- [5] E. R. Larrocha, *Nuevas tendencias en los sistemas de información*. Editorial Centro de Estudios Ramon Areces SA, 2017.
- [6] D. Caudillo et al., «Cultura de la información en el contexto educativo universitario: aportes teóricos», *Investigación bibliotecológica*, vol. 36, n.º 90, Art. n.º 90, mar. 2022, doi: 10.22201/iibi.24488321xe.2022.90.58480.
- [7] M. D. Pérez, «Procedimiento para el diseño de Sistemas de Gestión de Información en Cooperativas de Producción», *Cooperativismo y Desarrollo: COODES*, vol. 6, n.º 1, pp. 26-40, 2018.
- [8] A. Suárez Alfonso, I. Cruz Rodríguez, y Y. Pérez Macías, «La gestión de la información: Herramienta esencial para el desarrollo de habilidades en la comunidad estudiantil universitaria», *Revista Universidad y Sociedad*, vol. 7, n.º 2, pp. 72-79, ago. 2015.
- [9] Y. R. Cruz, «Gestión de Información y del Conocimiento para la toma de decisiones organizacionales», *Bibliotecas. Anales de investigación*, vol. 11, n.º 4, Art. n.º 4, mar. 2021.
- [10] O. S. Barzaga-Sablón, H. J. J. V. Pincay, J. V. Nevárez-Barberán, y M. V. A. Cobeña, «Gestión de la información y toma de decisiones en organizaciones educativas», *Revista de ciencias sociales*, vol. 25, n.º 2, pp. 120-130, 2019.
- [11] F. E. Vasquez-Rizo, J. V. Rodríguez Muñoz, y J.-A. Gómez-Hernández, «La gestión de información para medir la capacidad investigadora de una institución de educación superior», *Information management to measure the research capacity of a higher education institution*, vol. 40, n.º 8, mar. 2019, Accedido: 7 de noviembre de 2022. [En línea]. Disponible en: <http://hdl.handle.net/10614/11515>
- [12] M. R. Gallegos Macías, J. Galarza López, J. L. Almuiñas Rivero, M. R. Gallegos Macías, J. Galarza López, y J. L. Almuiñas Rivero, «Los sistemas de información como sustento a la gestión de la calidad en las Instituciones de Educación Superior», *Revista San Gregorio*, vol. 1, n.º 49, pp. 137-149, may 2022, doi: 10.36097/rsan.v0i49.1866.
- [13] J. P. Torres Fernández, J. G. Gallo Mendoza, R. F. Hallo Alvear, J. J. Abcarius, M. H. Muriel Páez, y A. Fernández Lorenzo, «Gestión de la información como herramienta para la toma de decisiones en salud: escenarios más probables», *Revista Cubana de Investigaciones Biomédicas*, vol. 36, n.º 3, pp. 0-0, sep. 2017.
- [14] Z. R. Arriaga, V. G. D. L. O. Burrola, y M. R. Corrales, «Modelo de gestión del conocimiento apoyado en las tecnologías de información y comunicación», *Criterio Libre*, vol. 16, n.º 28, Art. n.º 28, ago. 2018, doi: 10.18041/1900-0642/criteriolibre.2018v16n28.2132.
- [15] J. R. Moreno-Cevallos y B. L. Dueñas-Holguín, «Sistemas de información empresarial: la información como recurso estratégico», *Dominio de las Ciencias*, vol. 4, n.º 1, pp. 141-154, ene. 2018, doi: 10.23857/dc.v4i1.728.
- [16] D. Abrego Almazán, Y. Sánchez Tovar, y J. M. Medina Quintero, «Influencia de los sistemas de información en los resultados organizacionales», *Contaduría y administración*, vol. 62, n.º 2, pp. 303-320, jun. 2017, doi: 10.1016/j.cya.2016.07.005.

- [17] G. Gerón-Piñón *et al.*, «Sistemas de información en las universidades latinoamericanas: su impacto en los rankings internacionales», *Revista de la educación superior*, vol. 50, n.º 198, pp. 23-35, jun. 2021, doi: 10.36857/resu.2021.198.1699.
- [18] P. C. Huerta-Riveros, H. G. Gaete-Feres, L. M. Pedraja-Rejas, P. C. Huerta-Riveros, H. G. Gaete-Feres, y L. M. Pedraja-Rejas, «Dirección estratégica, sistema de información y calidad. El caso de una universidad estatal chilena», *Información tecnológica*, vol. 31, n.º 2, pp. 253-266, abr. 2020, doi: 10.4067/S0718-07642020000200253.
- [19] O. D. Medina Martínez *et al.*, «Sistema de información para la gestión de la producción científica de la Universidad de Ciencias Médicas de Cienfuegos», *MediSur*, vol. 17, n.º 6, Art. n.º 6, dic. 2019.
- [20] N. Milan Cristo *et al.*, «Módulo Medios diagnósticos para el Sistema de Información Hospitalaria XAVIA HIS», *Revista Cubana de Informática Médica*, vol. 13, n.º 1, jun. 2021, Accedido: 5 de diciembre de 2022. [En línea]. Disponible en: http://scielo.sld.cu/scielo.php?script=sci_abstract&pid=S1684-18592021000100006&lng=es&nrm=iso&tlng=es
- [21] L. Vega Izaguirre *et al.*, «Sistema para el manejo de datos de Ensayos Clínicos XAVIA SIDEC», *Revista Cubana de Informática Médica*, vol. 13, n.º 1, Art. n.º 1, jun. 2021, Accedido: 20 de junio de 2022. [En línea]. Disponible en: http://scielo.sld.cu/scielo.php?script=sci_abstract&pid=S1684-18592021000100005&lng=es&nrm=iso&tlng=es
- [22] M. Gómez Cruz, M. de los A. Valladares Mendoza, M. Gómez Cruz, y M. de los A. Valladares Mendoza, «Sistema de información Clínico-Hospitalario. Prototipo para indicaciones médicas de volumen y dieta», *Revista de Ciencias Médicas de Pinar del Río*, vol. 25, n.º 3, jun. 2021, Accedido: 5 de diciembre de 2022. [En línea]. Disponible en: http://scielo.sld.cu/scielo.php?script=sci_abstract&pid=S1561-31942021000300006&lng=es&nrm=iso&tlng=es
- [23] M. B. Rodríguez, M. A. C. Peraza, G. Z. González, y D. Á. Gainza, «Sistema Web para la gestión de los programas de maestrías del Instituto “Pedro Kourí”», p. 17.
- [24] R. A. Saavedra y G. M. G. Mora, «Sistema de gestión de la Nueva Universidad (SIGENU v4.0.0) como herramienta para la automatización de los procesos docentes y estadísticos de la Universidad de Granma (Revisión)», *Redel. Revista Granmense de Desarrollo Local*, vol. 2, n.º 4, Art. n.º 4, dic. 2018.
- [25] C. A. Rajme, G. A. P. Guerra, N. M. García, y G. P. Díaz, «Prototipo funcional del módulo para educación a distancia del Sistema de Gestión de la Nueva Universidad», *Anuario Ciencia en la UNAH*, vol. 18, n.º 3, Art. n.º 3, ene. 2022, Accedido: 8 de noviembre de 2022. [En línea]. Disponible en: <https://rcta.unah.edu.cu/index.php/ACUNAH/article/view/1528>
- [26] M. J. Suárez-Espinar, «Gestión Empresarial: una paradigma del siglo XXI», *Revista Científica FIPCAEC (Fomento de la investigación y publicación científico-técnica multidisciplinaria)*. ISSN: 2588-090X. *Polo de Capacitación, Investigación y Publicación (POCAIP)*, vol. 3, n.º 8, pp. 44-64, doi: 10.23857/fipcaec.v3i8.57.
- [27] P. Rubio Domínguez, *Introducción a la gestión empresarial*: Instituto Europeo de Gestión Empresarial, 2006. Accedido: 8 de noviembre de 2022. [En línea]. Disponible en: <http://up-rid2.up.ac.pa:8080/xmlui/handle/123456789/1893>
- [28] E. Correa y E. Correa, «Evolución del concepto de recursos humanos, desde el punto de vista de la psicología y la administración: discusiones y aciertos», *Suma de Negocios*, vol. 4, n.º 1, pp. 109-119, jun. 2013.
- [29] Y. Rojas Mesa, «Organización de la información: un factor determinante en la gestión empresarial», *ACIMED*, vol. 12, n.º 2, pp. 1-1, abr. 2004.
- [30] J. A. Zapata Cortés, M. D. Arango Serna, y W. Adarme Jaimes, «Herramientas tecnológicas al servicio de la gestión empresarial», 2010, Accedido: 8 de noviembre de 2022. [En línea]. Disponible en: <https://repositorio.unal.edu.co/handle/unal/38703>
- [31] «¿Qué es un sistema de gestión integrado? | NQA». <https://www.nqa.com/es-ca/certification/systems/integrated-management-systems> (accedido 8 de noviembre de 2022).

- [32] A. G. Zamalloa y I. M. S. Catalina, «Implantar un Sistema de Gestión Integral en Software libre», p. 100.
- [33] A. F. León Mendoza y E. R. López Ramirez, «Sistema de Gestión Integral de la Calidad de Software para el Centro de Gobierno Electrónico», bachelorThesis, Universidad de las Ciencias Informáticas. Facultad 3, 2015. Accedido: 8 de noviembre de 2022. [En línea]. Disponible en: <https://repositorio.uci.cu/jspui/handle/123456789/7230>
- [34] M. Deluca, «Assets Ultimate Versión 4.0».
- [35] M. F. Proaño, S. Y. Orellana, y I. O. Martillo, «Los sistemas de información y su importancia en la transformación digital de la empresa actual», *Revista ESPACIOS*, vol. 39, n.º 45, nov. 2018, Accedido: 8 de noviembre de 2022. [En línea]. Disponible en: <http://es.revistaespacios.com/a18v39n45/18394503.html>
- [36] J. Maldonado, M. Robles, y P. Crespo, «Utilización de arquetipos para la integración de sistemas de información hospitalarios», nov. 2022.
- [37] M. M. G. Ochoa, B. E. C. Arias, L. Siguenza-Guzman, y L. Segarra, «Integración de información de costos para la toma de decisiones en industrias de ensamblaje», *Revista Economía y Política*, pp. 100-117, ene. 2020, doi: 10.25097/rep.n31.2020.07.
- [38] M. Demichelis, M. S. Correa, y R. A. Bertone, «Interoperabilidad e integración entre sistemas de información en salud», presentado en XXIV Congreso Argentino de Ciencias de la Computación (La Plata, 2018)., 2018. Accedido: 8 de noviembre de 2022. [En línea]. Disponible en: <http://sedici.unlp.edu.ar/handle/10915/73476>
- [39] «Integración de sistemas: conoce su importancia, sus tipos y sus retos», *SYDLE*, 14 de septiembre de 2021. <https://www.sydle.com/br/blog/integracao-de-sistemas-6140d39a84679b13bf127a93/> (accedido 8 de noviembre de 2022).
- [40] «¿Qué es AWS?», *Amazon Web Services, Inc.* <https://aws.amazon.com/es/what-is-aws/> (accedido 1 de diciembre de 2022).
- [41] M. Technologies, «Top 30 AWS Services List to Choose in * 2022 - AWS Products», *Mindmajix*, 22 de abril de 2021. <https://mindmajix.com/top-aws-services> (accedido 1 de diciembre de 2022).
- [42] «Descripción general | Versión 4 de la API de informes de Analytics», *Google Developers*. <https://developers.google.com/analytics/devguides/reporting/core/v4?hl=es> (accedido 1 de diciembre de 2022).
- [43] D. Curbelo Reyes, «Servicios Web para Aplicaciones en la Universidad de Cienfuegos», Trabajo de diploma, Universidad de Cienfuegos Carlos Rafael Rodríguez, 2019.
- [44] M. García-Nieto *et al.*, «Aplicación del proceso unificado en el desarrollo de un software que estima el inventario y el crecimiento-rendimiento maderable en plantaciones de eucalipto», *Madera y bosques*, vol. 23, n.º 1, pp. 163-178, 2017, doi: 10.21829/myb.2017.2311557.
- [45] E. H. Orallo, «El Lenguaje Unificado de Modelado (UML)», p. 6.
- [46] J. L. Cabrera Chiappe, «Integración de sistemas orientada a servicios», *Service oriented system integration*, 2010, Accedido: 8 de noviembre de 2022. [En línea]. Disponible en: <https://repositorio.ulima.edu.pe/handle/20.500.12724/15587>
- [47] X. V. Guillén y L. N. Moldes, «Arquitectura de aplicaciones web», p. 46.
- [48] J. D. López Hinojosa, «Arquitectura de software basada en microservicios para desarrollo de aplicaciones web de la Asamblea Nacional», masterThesis, 2017. Accedido: 7 de noviembre de 2022. [En línea]. Disponible en: <http://repositorio.utn.edu.ec/handle/123456789/7603>
- [49] T. Rodríguez, R. D. Santos, y J. Aguilar, «Metodología para el Desarrollo de Aplicaciones Web Utilizando Datos Enlazados», p. 9, 2017.
- [50] A. García-Holgado y F. J. García-Peñalvo, «Definición de ecosistemas de aprendizaje independientes de plataforma», Servicio de Publicaciones Universidad de Zaragoza, 2017. Accedido: 7 de noviembre de 2022. [En línea]. Disponible en: <https://repositorio.grial.eu/handle/grial/1024>
- [51] A. Cols Fermín y M. Salcedo Real, «Arquitectura de microservicios con RESTful», masters, E.T.S.I de Sistemas Informáticos (UPM), 2017. Accedido: 1 de diciembre de 2022. [En línea]. Disponible en: <https://oa.upm.es/48230/>

- [52] Z. E. Mamani Rodríguez, L. Del Pino Rodríguez, y J. C. Gonzales Suarez, «Arquitectura basada en Microservicios y DevOps para una ingeniería de software continua», *idata*, vol. 23, n.º 2, pp. 141-149, dic. 2020, doi: 10.15381/idata.v23i2.17278.
- [53] F. H. Vera-Rivera, H. Astudillo, y C. Gaona, «Desarrollo de aplicaciones basadas en microservicios: tendencias y desafíos de investigación», *RISTI - Revista Ibérica de Sistemas e Tecnologías de Informação*, n.º E23(2019), pp. 107-120, oct. 2019.
- [54] A. Nebel, «Arquitectura de microservicios para plataformas de integración», 2019, Accedido: 7 de noviembre de 2022. [En línea]. Disponible en: <https://www.colibri.udelar.edu.uy/jspui/handle/20.500.12008/20586>
- [55] M. P. Ascue y J. L. M. Aroni, «Integración de procesos de negocio aplicando la arquitectura orientada a servicios (SOA)», *Interfases*, n.º 10, pp. 93-121, 2017.
- [56] L. M. A. Hernández, V. A. P. Romero, S. A. S. González, y J. A. V. Rodríguez, «Arquitectura REST para el desarrollo de aplicaciones web empresariales», *Revista Electrónica sobre Tecnología, Educación y Sociedad*, vol. 8, n.º 15, Art. n.º 15, ene. 2021, Accedido: 1 de diciembre de 2022. [En línea]. Disponible en: <https://www.ctes.org.mx/index.php/ctes/article/view/748>
- [57] A. Arsaute, F. A. Zorzán, M. Daniele, A. González, y M. Frutos, «Generación automática de API REST a partir de API Java, basada en transformación de Modelos (MDD)», presentado en XX Workshop de Investigadores en Ciencias de la Computación (WICC 2018, Universidad Nacional del Nordeste), 2018. Accedido: 1 de diciembre de 2022. [En línea]. Disponible en: <http://sedici.unlp.edu.ar/handle/10915/67777>
- [58] M. J. Guapi Auquilla, «Diseño metodológico para el desarrollo de interfaces gráficas en páginas web utilizando los lenguajes HTML 5 y CSS 3», bachelorThesis, Riobamba, 2018. Accedido: 2 de diciembre de 2022. [En línea]. Disponible en: <http://dspace.unach.edu.ec/handle/51000/5163>
- [59] M. A. Álvarez, *Manual de CSS3*. [En línea]. Disponible en: <http://www.desarrolloweb.com/manuales/css3.html>
- [60] J. Eguíluz Pérez, *Introducción a JavaScript*. [En línea]. Disponible en: http://190.57.147.202:90/jspui/bitstream/123456789/430/1/introduccion_javascript%20%281%29.pdf
- [61] E. Haro, T. Guarda, A. O. Z. Peñaherrera, y N. Quiña, «Desarrollo backend para aplicaciones web, Servicios Web Restful: Node.js vs Spring Boot», p. 14.
- [62] E. Brown, *Web Development with Node and Express: Leveraging the JavaScript Stack*. O'Reilly Media, Inc., 2019.
- [63] M. Jones, J. Bradley, y N. Sakimura, «JSON Web Token (JWT)», Internet Engineering Task Force, Request for Comments RFC 7519, may 2015. doi: 10.17487/RFC7519.
- [64] M. Owens, Ed., «Introducing SQLite», en *The Definitive Guide to SQLite*, Berkeley, CA: Apress, 2006, pp. 1-16. doi: 10.1007/978-1-4302-0172-4_1.
- [65] «¿Qué es Microsoft SQL Server? - Definición en WhatIs.com», *ComputerWeekly.es*. <https://www.computerweekly.com/es/definicion/Microsoft-SQL-Server> (accedido 2 de diciembre de 2022).
- [66] J. O. Flores Castillejo, «Sistema web open source vue Js para el proceso de pruebas de calidad de software en la empresa NextPerience», *Repositorio Institucional - UCV*, 2020, Accedido: 2 de diciembre de 2022. [En línea]. Disponible en: <https://repositorio.ucv.edu.pe/handle/20.500.12692/59504>
- [67] M. J. Pérez Pin y A. J. Tejada Solórzano, «Desarrollo de un sistema Web de gestión de productos y servicios a empresas de comercio electrónico empleando las herramientas Elk y el Framework Quasar para el análisis del comportamiento e interés del consumidor.», bachelorThesis, Universidad de Guayaquil. Facultad de Ciencias Matemáticas y Físicas. Carrera de Ingeniería en Sistemas Computacionales., 2021. Accedido: 2 de diciembre de 2022. [En línea]. Disponible en: <http://repositorio.ug.edu.ec/handle/redug/52479>
- [68] «API Documentation & Design Tools for Teams | Swagger». <https://swagger.io/> (accedido 2 de diciembre de 2022).

- [69] «Documentation for Visual Studio Code». <https://code.visualstudio.com/docs> (accedido 2 de diciembre de 2022).
- [70] L. R. B. Hernández, D. M. Peña, O. R. Valdés, y O. M. Cornelio, «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*, vol. 9, n.º 7, pp. 7-20, 2016.
- [71] «Zotero | Bibliotecas Universidad de Salamanca». <https://bibliotecas.usal.es/zotero> (accedido 2 de diciembre de 2022).

Glosario de términos

API: interfaz de programación de aplicaciones, es un conjunto de subrutinas, funciones y procedimientos que ofrece cierta biblioteca para ser utilizada por otro software como una capa de abstracción. En el caso del microservicio se conoce como al conjunto de recursos que pone a disposición al usuario y que debe ser accedido a través de una interfaz

Backend: es la parte del desarrollo web que se encarga de que toda la lógica de una página web funcione. Se trata del conjunto de acciones que pasan en una web pero que no vemos como, por ejemplo, la comunicación con el servidor.

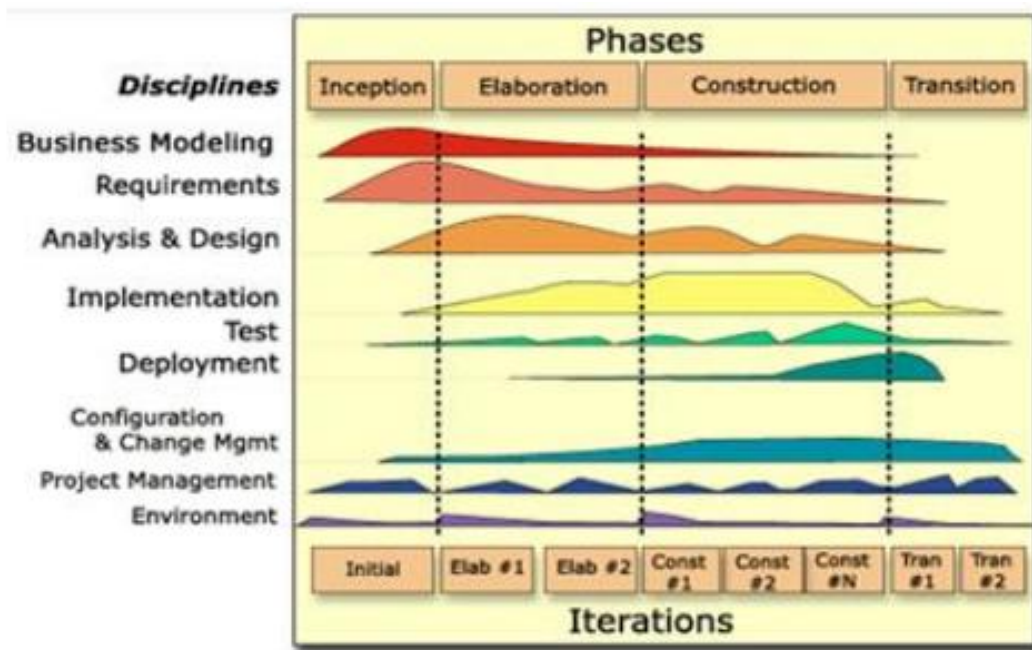
EndPoint: Hace referencia a una URL a la cual el usuario puede acceder para consumir un recurso a través de un método http determinado

Frontend: es la parte del desarrollo web que se dedica a la parte frontal de un sitio web, en pocas palabras del diseño de un sitio web, desde la estructura del sitio hasta los estilos como colores, fondos, tamaños hasta llegar a las animaciones y efectos.

Anexos

Anexo 1 – Metodología RUP

Anexo 1.1 – Fases de RUP



Anexo 2 – Prototipos de interfaces

Anexo 2.1 – Prototipo de la Interfaz del caso de uso Autenticar

Microservicio ASSETS

Panel de Administración

Correo

admin@gmail.com

Contraseña

.....



INGRESAR

Anexo 2.2 – Prototipo de la Interfaz del caso de uso Cerrar Sección

Microservicio ASSETS



Gestión de Usuarios

Usuarios

AGREGAR USUARIO +

Nombre	Email	Estado	Rol	Api Key
apiDocsUser	test@gmail.com	Activo	API_CONSUMER	eyJhbGciOi...
Yaal	Yaal@gmail.com	Activo	API_CONSUMER	eyJhbGciOi...
a	a@a	Activo	API_CONSUMER	eyJhbGciOi...
Palmero	Palmero@gmail.com	Activo	API_CONSUMER	eyJhbGciOi...
admin	admin@gmail.com	Activo	ADMIN	eyJhbGciOi...

Anexo 2.3 – Prototipo de la Interfaz del caso de uso Gestionar Usuarios

Microservicio ASSETS



Gestión de Usuarios

Usuarios

AGREGAR USUARIO +

Nombre	Email	Estado	Rol	Api Key
apiDocsUser	test@gmail.com	Activo	API_CONSUMER	eyJhbGciOi...
Yaal	Yaal@gmail.com	Activo	API_CONSUMER	eyJhbGciOi...
a	a@a	Activo	API_CONSUMER	eyJhbGciOi...
Palmero	Palmero@gmail.com	Activo	API_CONSUMER	eyJhbGciOi...
admin	admin@gmail.com	Activo	ADMIN	eyJhbGciOi...

Acceda a la Información del Usuario cambie el estado de la cuenta, su Api Key o elimine la cuenta

Estado: Activo

Tipo de Usuario: API_CONSUMER

apiDocsUser

test@gmail.com

eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJyZWRlIjoiJlNjbGVzIjoibXBjX0NPTlNlbnVzIiwiaWF0IjoiMTYyOTg2NTUzMjUzNCwiaWF0IjoxNjY5ODM1NTcyfQ.7X8kFhHkS3HNqfVvzF6BBZeUG-d0xcTtZ9D_TbGW0s

[CAMBIAR ESTADO DE LA CUENTA](#)[CAMBIAR LA API KEY](#)

ELIMINAR USUARIO

Panel de Creación de Usuario

Llene los campos para crear un nuevo usuario



Rol
Usuario



CREAR USUARIO



CANCELAR

Panel de Modificación de Usuario

Esta seguro que desea cambiar el Api Key del Usuario

CAMBIAR API KEY

CANCELAR

Panel de Modificación de Usuario

Cambia el estado de la Cuenta

Estado
Activo



CAMBIAR ESTADO

CANCELAR

Panel de Modificación de Usuario

Esta seguro que desea eliminar la cuenta del Usuario

ELIMINAR CUENTA

CANCELAR

Anexo 2.4 – Prototipo de la Interfaz del caso de uso Mostrar todos los trabajadores

Workers

GET

/workers?page=n

Regresa los n-100 trabajadores.

Al indicar el número de la página regresa los trabajadores entre (n-1)100 y (n)100

Parameters

Try it out

Name	Description
page required integer (query)	El número de la página de trabajadores a devolver
page	
api_key required string (header)	
api_key	

Anexo 2.5 – Prototipo de la Interfaz del caso de uso Buscar Información de un Trabajador

GET

/workers/{id}

Regresa el n trabajador.

Regresa el trabajador que ocupa la fila que hace referencia el id

Parameters

Try it out

Name	Description
id required integer (path)	Id del trabajador
id	
api_key required string (header)	
api_key	

Anexo 2.6 – Prototipo de la Interfaz del caso de uso Filtrar Información de los Trabajadores

GET

/workers/filter

Regresa el trabajador o los trabajadores filtrados.

Regresa el trabajador o los trabajadores que cumplen con las condiciones pasadas por query. Los campos por los que se pueden filtrar son todos los referenciados en el esquema en la documentacion solo se representan algunos para las pruebas

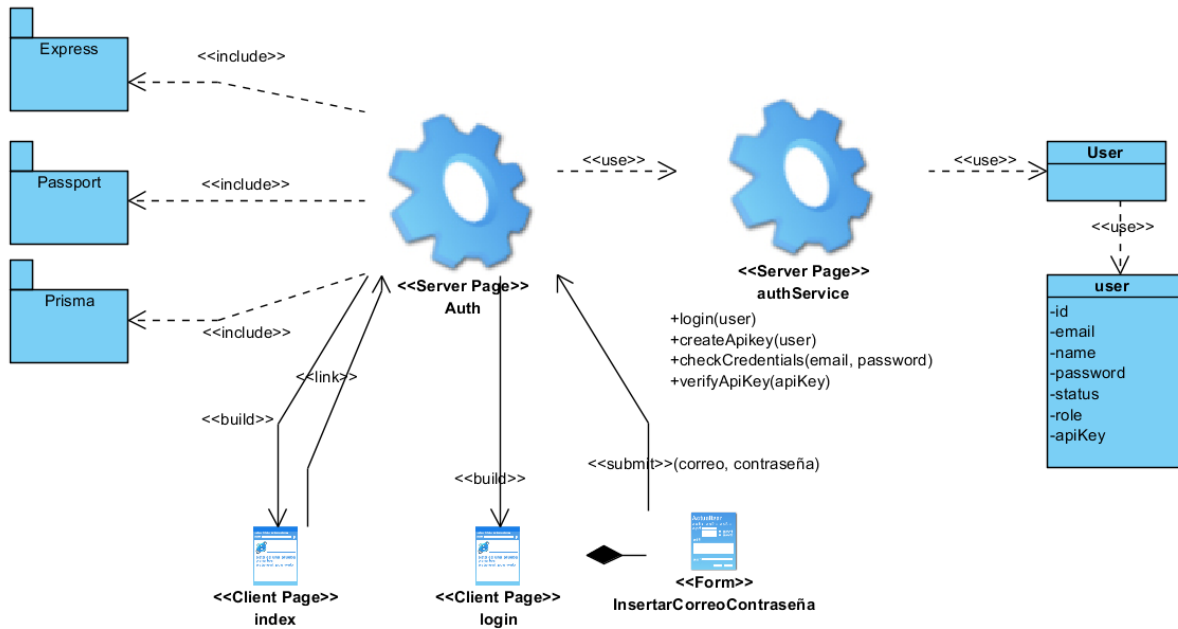
Parameters

Try it out

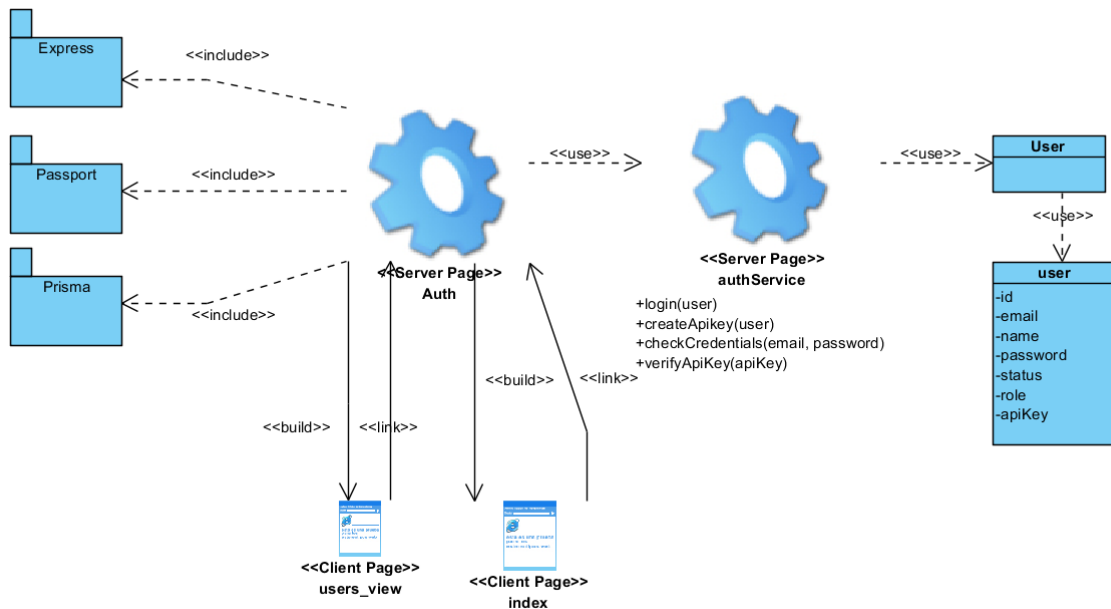
Name	Description
Id_Expediente	El Identificador del expediente del trabajador
string	
(query)	
	Default value :
	Id_Expediente
No_CI	El Carnet de Identidad del trabajador
string	
(query)	
	Default value :
	No_CI
Nombre	El Nombre del trabajador
string	
(query)	
	Default value :
	Nombre

Anexo 3 – Prototipos de interfaces

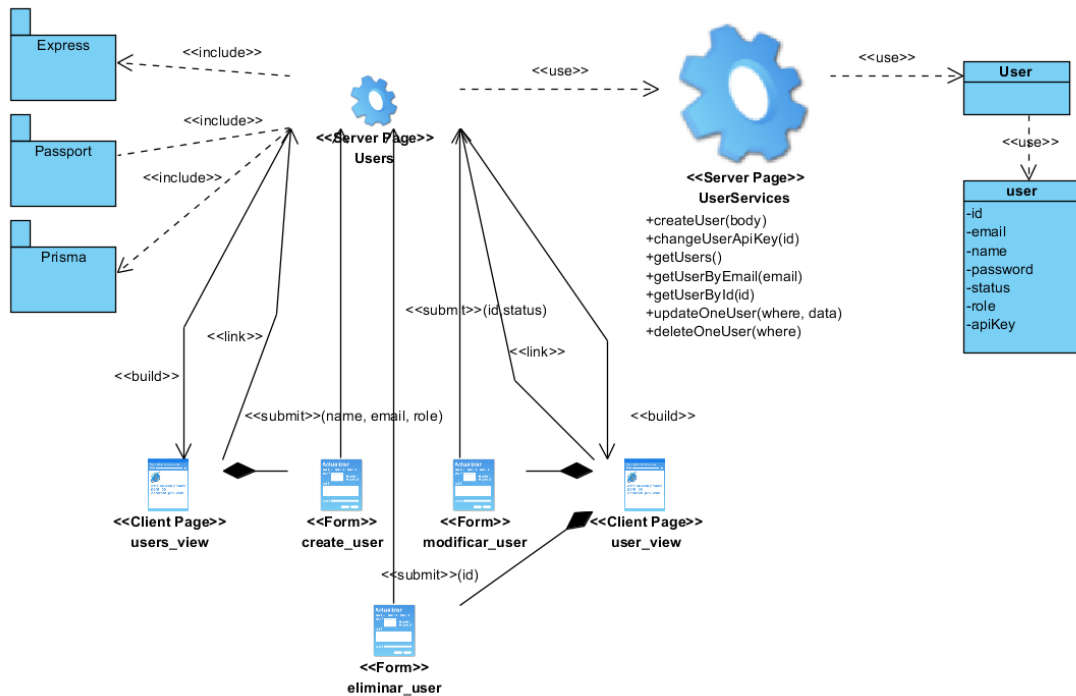
Anexo 3.1 – Diagrama de clases del diseño CU <Autenticar>



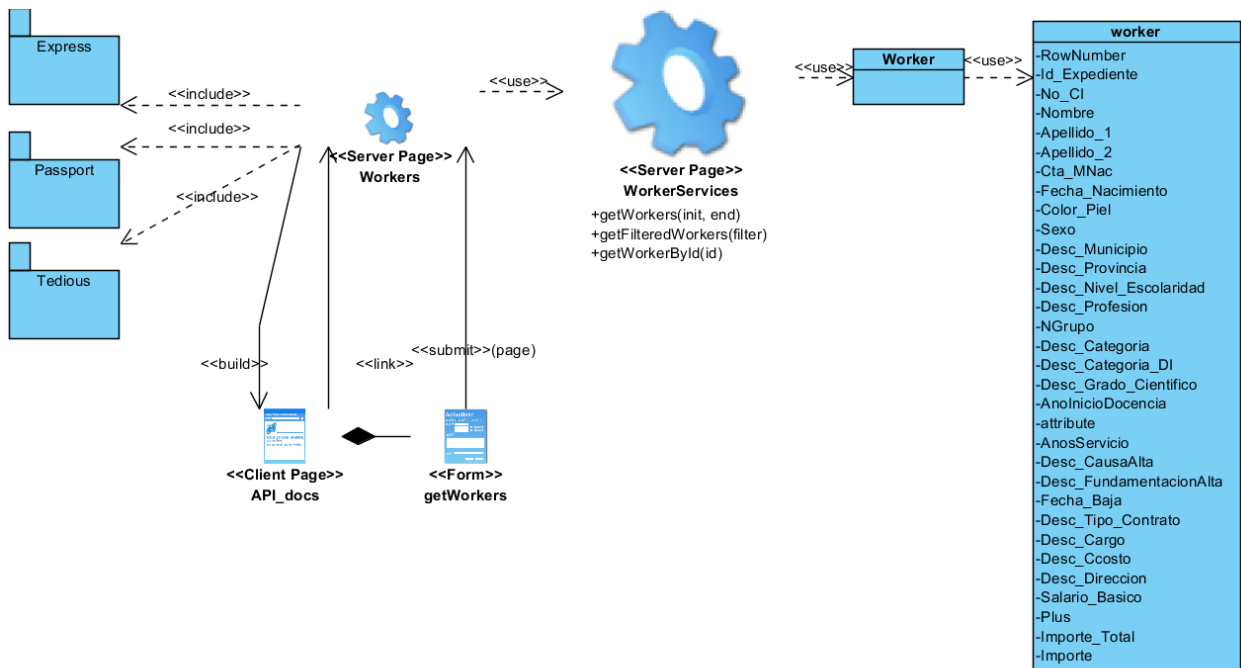
Anexo 3.2 – Diagrama de clases del diseño CU <Cerrar Sección>



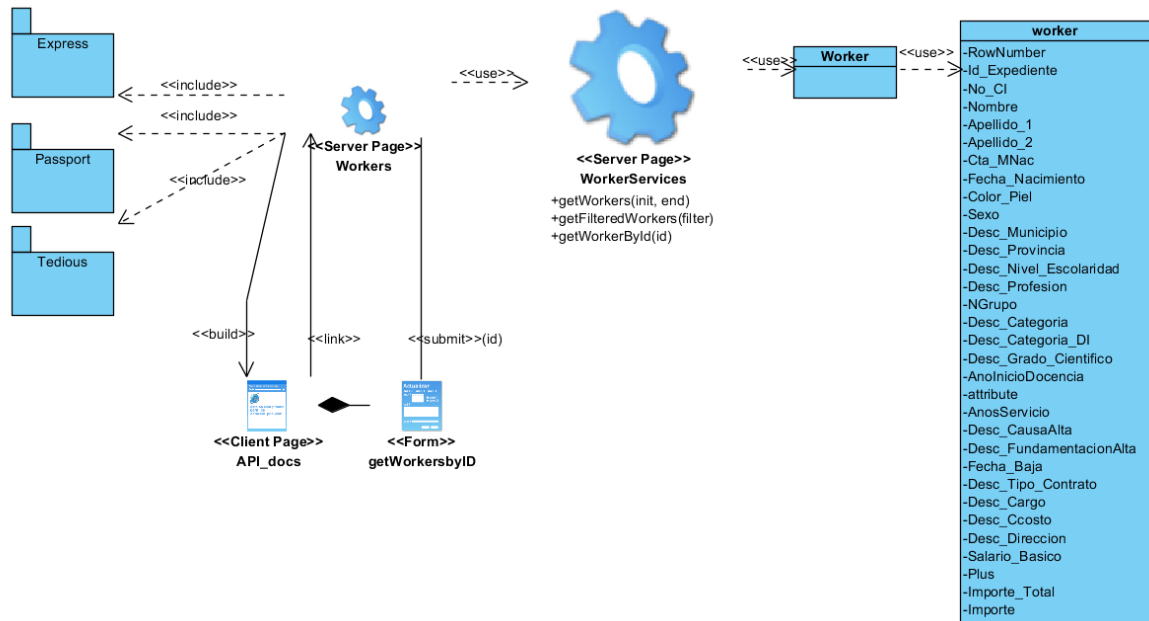
Anexo 3.3 – Diagrama de clases del diseño CU <Gestionar Usuarios>



Anexo 3.4 – Diagrama de clases del diseño CU <Mostrar todos los trabajadores>



Anexo 3.5 – Diagrama de clases del diseño CU <Buscar Información de un trabajador>



Anexo 3.6 – Diagrama de clases del diseño CU <Filtrar información de los trabajadores>

