DISEÑO ARQUITECTURA DE SOFTWARE PARA LA GESTIÓN ACADEMICA EN COLEGIO INTEGRADO ATALAYA

INGENIERO:
EMMANUEL HERNANDO GAMBOA ROJAS

UNIVERSIDAD SIMÓN BOLÍVAR EXT-BARRANQUILLA
FACULTAD DE INGENIERIA DE SISTEMAS
SAN JOSÉ DE CÚCUTA
2021

DISEÑO ARQUITECTURA DE SOFTWARE PARA LA GESTIÓN ACADEMICA EN COLEGIO INTEGRADO ATALAYA

INGENIERO:
EMMANUEL HERNANDO GAMBOA ROJAS

UNIVERSIDAD SIMÓN BOLÍVAR EXT-BARRANQUILLA
FACULTAD DE INGENIERIA DE SISTEMAS
SAN JOSÉ DE CÚCUTA
2021

Nota de aceptación:

_____

Firma del presidente del jurado

_____

Firma del jurado

_____

Firma del jurado

San José de Cúcuta, 03 de Marzo de 2022

CONTENIDO

LISTA DE ANEXOS

Pág.

LISTA DE TABLAS

LISTA DE FIGURAS

RESUMEN

El presente trabajo tiene como propósito diseñar una Arquitectura de Software Académico que se adapte a las necesidades del colegio Integrado Atalaya en la ciudad de Cúcuta y listar las ventajas y desventajas que tendría implementarla. El proyecto se basó en la investigación de las diferentes falencias con las que cuenta el actual software académico que utiliza el colegio Integrado Atalaya y los costos de utilizar el software como servicio pagando una anualidad, también los costos de las diferentes plataformas que utilizan el resto de colegios en la ciudad de Cúcuta.

El proyecto busca brindar una solución tecnológica propia que permita una reducción de costos significativa y la cual automatice el proceso académico del colegio dentro de las actividades a tener en cuenta están la gestión de estudiantes, docentes, padres de familia, notas del estudiante, asignaturas, cursos, boletines, inscripciones, logros, etc. El proyecto conto con 3 fases, en la primera fase se identificó la metodología y el método de investigación que se utilizara para este proyecto, también se realizó una identificación inicial en donde se logró evidenciar las funcionalidades y falencias con las que cuenta el software actual y los costos que tiene dicho servicio anual, en la segunda fase se procedió a investigar las diferentes plataformas académicas que prestan el servicio en la ciudad de Cúcuta, los costos que manejan, las funcionalidades con las que cuentan y los servicios que brindan al colegio durante el año lectivo, en la tercera y última fase se hizo la toma y análisis de requerimientos para el diseño de la Arquitectura de Software, por último se realizó un presupuesto estimado del desarrollo el cual permitió determinar que si es rentable para el colegio.

Palabras Clave: Arquitectura, Plataforma, Software, Colegio

ABSTRACT

The purpose of this work is to design an Academic Software Architecture that adapts to the needs of the Juan Atalaya Integrated School in the city of Cúcuta and to list the advantages and disadvantages that could be implemented. The project was based on the investigation of the different shortcomings of the current academic software used by the Juan Atalaya Integrated School and the costs of using the software as a service paying an annual fee, as well as the costs of the different platforms used by the rest. of schools in the city of Cúcuta.

The project seeks to provide its own technological solution that allows a significant cost reduction and which automates the academic process of the school within the activities to be taken into account are the management of students, teachers, parents, student notes, subjects, courses, newsletters, registrations, achievements, etc. The project had 3 phases, in the first phase the methodology and the research method used for this project were identified, an initial identification was also carried out where it was possible to demonstrate the functionalities and shortcomings of the current software and the costs that this annual service has, in the second phase we proceeded to investigate the different academic platforms that provide the service in the city of Cúcuta, the costs they handle, the functionalities they have and the services they provide to the school during the school year, in the third and final phase, the taking and analysis of requirements for the design of the Software Architecture was made, finally an estimated budget of the development was made, which allowed determining if it is profitable for the school.

Keywords: Architecture, Platform, Software, College

INTRODUCCIÓN

El colegio Integrado Atalaya en la ciudad de Cúcuta apoyándose en la tecnología actualmente provee a los alumnos, padres de familia, docentes y directivos de un software académico el cual les ha permitido automatizar el proceso de las diferentes actividades del día a día del colegio, por dicha plataforma el colegio debe pagar una anualidad a una empresa externa la cual es la dueña del software y si desean realizar ajustes a la plataforma estos generan un costo extra a parte de la anualidad que pagan, por tal motivo el colegio con el fin de reducir costos desea tener un software propio el cual le permita seguir teniendo automatizado el proceso de las actividades del día a día del colegio y tener un software que se adapte mucho más a las necesidades del colegio. Por tal motivo se planteó al colegio el diseño de una Arquitectura de Software Académico la cual se utilizará como la base para realizar el presupuesto en tiempo y dinero del desarrollo del software, esto con el fin de analizar si es rentable para el colegio invertir en un software propio y que ventajas y desventajas tiene.

El diseño de la Arquitectura del Software Académico se realizó basándose en las necesidades del colegio y las diferentes funcionalidades y falencias con las que cuenta el software académico que actualmente están usando esto con el fin de diseñar una Arquitectura de Software que cumple con las funcionalidades críticas que suplen todo el proceso de actividades del día a día del Colegio.

1. PROBLEMA

1.1. PLANTEAMIENTO DEL PROBLEMA

El colegio Integrado Atalaya está ubicado en la ciudad de Cúcuta, es una institución educativa que presta educación preescolar básica, primaria y secundaria en las jornadas de la mañana y tarde para los diferentes niños que viven en los alrededores de las diferentes sedes educativas del colegio.

El colegio con el pasar del tiempo ha estado incrementando la cantidad de alumnos que tienen inscritos en sus diferentes sedes educativas esto gracias a la calidad de educación que brindan sus docentes y directivos en las aulas de clases a los diferentes alumnos. Dado el incremento que ha tenido el colegio desde hace algunos años se vio en la necesidad de automatizar el proceso de actividades del día a día pues llevaban el registro de alumnos, notas, inscripciones, etc. de forma manual y no tenían control de la información ni copias de seguridad, por tal motivo se vieron en la necesidad de contratar con una empresa el servicio de plataforma académica y pagar una anualidad por dicho software. El costo de la anualidad varía según la cantidad de alumnos que tiene inscritos en el año lectivo el colegio y los ajustes que desee realizar el colegio al aplicativo tienen un costo adicional. Como los costos de utilizar dicha plataforma para el colegio son altos por tener un volumen grande de estudiantes inscritos, se le ofreció al colegio el diseño de una Arquitectura de Software Académico que cumpla con sus necesidades y la cual permita utilizar como la base para realizar el presupuesto en tiempo y dinero del desarrollo del software, esto con el fin de analizar si es rentable para el colegio invertir en un software propio y que ventajas y desventajas tiene.

1.2. FORMULACIÓN DEL PROBLEMA

¿Es rentable para el colegio Integrado Atalaya desarrollar e implementar la Arquitectura de Software Académico planteada?

1.3. SISTEMATIZACIÓN DEL PROBLEMA

¿El desarrollo, mantenibilidad y soporte del Software Académico planteado a través de la Arquitectura diseñada son costosos?

¿La Arquitectura del Software académico planteada cumple con las necesidades del colegio Integrado Atalaya?

1.4. DELIMITACIÓN

1.4.1. Espacial

El proyecto de investigación se llevará a cabo en el colegio Integrado Atalaya ubicado en la ciudad de Cúcuta.

1.4.2. Temporal

El desarrollo de este proyecto se llevará a cabo desde el mes de Octubre del año 2021 mes en que se dio inicio al 2 Semestre de la especialización, hasta marzo del año 2022 mes en que se acaba el 2 semestre de la especialización, debido a que es el tiempo estipulado por la Universidad Simón Bolívar.

1.4.3. Conceptual

El desarrollo de este proyecto está delimitado por los conocimientos adquiridos en mi formación como Especialista en Ingeniería de Software específicamente en lo referente a Gestión de Proyectos y Arquitectura de Software.

1.4. JUSTIFICACIÓN

Ante el aumento de alumnos inscritos en el colegio Integrado Atalaya el cual está ubicado en la ciudad de Cúcuta y cuenta con varias sedes, el colegio ha visto la necesidad de tener su propio software académico pues el costo que debe pagar por utilizar dicho software depende de la cantidad de alumnos que tienen inscritos en el año lectivo y las modificaciones que requieran hacer tiene un costo adicional, es por eso que se planteó al colegio el diseño de una Arquitectura de Software Académico que cumpla con sus necesidades y de la cual se pueda utilizar como la base para realizar el presupuesto en tiempo y dinero del desarrollo del software, esto con el fin de analizar si es rentable para el colegio invertir en un software propio y que ventajas y desventajas tendría.

1.6. OBJETIVOS

1.6.1. Objetivo General.

Diseñar una Arquitectura de Software en el colegio Integrado Atalaya por medio del Modelo de “4+1” Vistas de la Arquitectura del Software para la gestión académica.

1.6.2. Objetivos Específicos.

- Identificar los requisitos funcionales y no funcionales que componen el proceso de gestión académica.
- Diseñar la Arquitectura de Software tomando como base los requisitos funcionales y no funcionales utilizando el Modelo de “4+1” Vistas de la Arquitectura del Software.
- Validar la Arquitectura de Software utilizando la metodología Architecture Tradeoff Analysis Method (ATAM)

1.6.3. Actividades de los Objetivos Específicos.

1. Identificar los requisitos funcionales y no funcionales que componen el proceso de gestión académica.
    - Listar los requisitos funcionales
    - Listar los requisitos no funcionales
2. Diseñar la Arquitectura de Software tomando como base los requisitos funcionales y no funcionales utilizando el Modelo de “4+1” Vistas de la Arquitectura del Software
    - Investigar sobre el Modelo de “4+1” Vistas de la Arquitectura del Software
    - Definir los requisitos funcionales
    - Definir los requisitos no funcionales
    - Definir los casos de uso
    - Definir la vista de desarrollo
    - Definir la vista de procesos
    - Definir la vista lógica.
    - Definir la vista de despliegue
3. Validar la Arquitectura de Software utilizando la metodología Architecture Tradeoff Analysis Method (ATAM).
    - Investigar la metodologia Architecture Tradeoff Analysis Method (ATAM)
    - Presentación del ATAM
    - Presentación de las metas del negocio
    - Presentación de la Arquitectura
    - Identificación de los enfoques arquitectónicos
    - Generar el árbol de utilidad de los atributos de calidad

- Análisis de las propuestas arquitectónicas
- Lluvia de ideas y priorización de escenarios
- Análisis de las propuestas arquitectónicas
- Presentación de los resultados

1.6.4. Metas de la Investigación.

- ✓ Ofrecerle al colegio Integrado Atalaya una Arquitectura de Software que permita gestionar la información académica.
- ✓ Presupuesto estimado sobre desarrollo, mantenibilidad y soporte del Software Académico utilizando la Arquitectura de Software planteada con sus ventajas y desventajas.

1.6.5. Producto.

Diseño de la Arquitectura del Software Académico.

2. MARCO REFERENCIAL

2.1 ANTECEDENTES

A medida que avanza la tecnología la forma en cómo se desarrolla software a estado cambiando y apareciendo nuevas tecnologías las cuales permiten desarrollar software robusto, estable, reutilizable, mantenible y escalable, en definitiva, software de calidad.

Todo esto es gracias a que se ha utilizado la Ingeniería de Software la cual permite toda la gestión del proyecto, el análisis previo de la situación, el diseño del proyecto, el desarrollo del software, las pruebas necesarias para confirmar su correcto funcionamiento y la implementación del sistema.

La Arquitectura de Software está directamente relacionado con la Ingeniería de Software pues consiste en determinar y esquematizar la estructura general del proyecto, diagramando su esqueleto con un grado relativamente alto de especificidad y señalando los distintos componentes que serán necesarios para llevar a cabo el desarrollo, tales como aplicaciones complementarias y bases de datos. Se trata de un punto fundamental del proceso, y es muchas veces la clave del éxito de un producto informático.

En años anteriores los colegios gestionaban la información académica de forma manual por medio de libros y era un poco tedioso, con los avances en el desarrollo de software los colegios han empezado a utilizar plataformas académicas las cuales agilizan la gestión académica y permiten tener copias de seguridad de la información almacenada.

2.1.1. Internacionales.

- Roberto Martín Tuesta Pereyra en su tesis “Diseño De Una Arquitectura Para El Sistema De Gestión Académica En La Universidad Nacional De La Amazonía Peruana” para obtener el grado de Maestro En Ingeniería De Sistemas Con Mención En Gerencia De La Información Y Gestión De Software, habla sobre: Diseñar una arquitectura de referencia basada en el marco de trabajo TOGAF para el Sistema de Gestión Académica de la Universidad Nacional de la Amazonía Peruana. Esta investigación contaba con dos variables: La Variable Independiente (X): Arquitectura de Software y la Variable Dependiente (Y): Sistema de Gestión Académica. La investigación fue de tipo Cualitativo, Descriptivo y Explicativo, perteneciente al diseño no experimental, mientras que el diseño específico fue Transeccional Descriptivo. La población estuvo conformada por los 06 trabajadores de la Universidad Nacional de la Amazonía Peruana (UNAP). La muestra estuvo conformada por el total de la población, es decir 6 personas, La técnica que se empleó en la recolección de los datos fue la encuesta y el instrumento fue el cuestionario.
- Juan Tahuiton Mora en su tesis “Arquitectura de software para aplicaciones Web” para obtener el grado de Maestro en Ciencias en Computación, habla sobre: Una arquitectura de software para aplicaciones Web en donde se sigue un proceso de ingeniería de software. En este desarrollo, la arquitectura se descompone mediante distintas vistas o enfoques tales como, la vista lógica, la vista de procesos, la vista de desarrollo, la vista física y la vista de seguridad. Cada vista, en esta tesis, se desarrolla mediante el lenguaje de modelado unificado UML. Además de presentar la

arquitectura genérica de un sistema en Web, en esta tesis presentamos un caso de estudio en el cual se utiliza la arquitectura genérica desarrollada para modelar la aplicación.

- Luis Miguel Ruiz Hernández en su tesis “Arquitectura De Software Para La Gestión Y Proyección Financiera De Planes De Negocios” para obtener el grado de Ingeniero de Sistemas, habla sobre: Diseñar la arquitectura de un sistema que permita gestionar la información generada por los proyectos de planes de negocio que realizan los estudiantes en los Programas de Administración en la Fundación Universitaria Tecnológico Comfenalco. Entendiéndose por “Gestión de la Información” en el contexto de las organizaciones, todos los procesos llevados a cabo, relacionados con la obtención de la información adecuada, en la forma correcta, para la persona indicada, al coste adecuado, en el momento oportuno, en el lugar apropiado y articulando todas estas operaciones para el desarrollo de una acción correcta (Pérez-Montoro Gutiérrez & Golkhosravi, 2010).

2.1.2. Nacionales.

- Oscar Rolando Sánchez Rueda en su tesis “Diseño e implementación de una arquitectura de software con soporte para dispositivos móviles que apoye la enseñanza musical en niños” para obtener el grado de Maestría En Ingeniería De Sistemas Y Computación habla sobre: El proceso de desarrollo del sistema MUSIKAMI, este sistema, es una herramienta de apoyo a la educación musical en niños enfocada principalmente en entrenamiento auditivo y rítmico. Este proyecto se divide en 4 fases donde se describe en detalle el proceso de construcción del sistema de información: 1) Análisis. 2) diseño. 3) Diseño y validación de la arquitectura. 4) Implementación y

validación con usuarios finales. Como resultado de este proyecto obtenemos esta memoria de grado, los anexos técnicos correspondientes a cada fase, el código final del sistema junto con sus documentos de diseño respectivos y la evaluación realizada por los usuarios finales.

2.1.3. Locales.

- Londoño Maya, Mauricio, Salcedo Barnes, Osvaldo y Salguero Silvera, Arnulfo en su tesis "Diseño y desarrollo de un software que permita el registro de notas del Colegio Jesús Maestro" para obtener el grado de Ingeniería de Sistemas, hablan sobre: Actualmente la mayoría de los colegios se encuentran en un proceso de desarrollo y ésta no es la excepción; pues se pretende desarrollar un software para el proceso de notas de los estudiantes del Colegio Jesús Maestro de Soledad, esto con el fin de cambiar la actual situación que se presenta en este colegio. En estos momentos, el colegio no tiene un sistema de información adecuado a las circunstancias, pues todo tipo de información se ésta manejando de forma manual y esto va en contra de un buen ambiente entre los miembros del instituto y los usuarios que se acercan a éste; es por eso que la razón de este proyecto es la creación de un software que no sólo será útil al instituto, si no a los estudiantes, docentes, padres de familia y toda la comunidad interesada en hacer parte de este plantel educativo. Este proceso se está desarrollando bajo las técnicas basadas en: encuestas, asesorías, estudios y consultas, sabemos lo oportuno que es contar con un software de esta magnitud, porque la ayuda que ofrecerá será inmensa tanto en espacio, como en tiempo y eficiencia.

2.2. MARCO TEORICO

2.2.1. Software.

Según Sommerville, I. (2005). “Muchas personas asocian el termino software con los programas de computadora. Sin embargo, yo prefiero una definición más amplia donde el software no son solo programas, si no todos los documentos asociados y la configuración de datos que se necesitan para hacer que estos programas operen de manera correcta. Por lo general, un sistema de software consiste en diversos programas independientes, archivos de configuración que se utilizan para ejecutar estos programas, un sistema de documentación que describe la estructura del sistema, la documentación para el usuario que explica cómo utilizar el sistema y sitios web que permitan a los usuarios descargar información de productos recientes.”

2.2.2. Arquitectura de Software

Según Bass, Clements y Kazman, 2012. “La arquitectura de software de un sistema es el conjunto de estructuras necesarias para razonar sobre el sistema. Comprende elementos de software, relaciones entre ellos, y propiedades de ambos”.

Según P. Clements, 2003. “La AS es, a grandes rasgos, una vista del sistema que incluye los componentes principales del mismo, la conducta de esos componentes según se la percibe desde el resto del sistema y las formas en que los componentes interactúan y se coordinan para alcanzar la misión del sistema. La vista arquitectónica es una vista abstracta, aportando el más alto nivel de comprensión y la supresión o diferimiento del detalle inherente a la mayor parte de las abstracciones”.

2.2.3. Ingeniería de Software

Según Campderrich Falgueras, B. (2003). "Un sistema de software, denominado también aplicación o simplemente software, es un conjunto integrado de programas que en su forma definitiva se pueden ejecutar, pero comprende también las definiciones de estructuras de datos (por ejemplo, definiciones de bases de datos) que utilizan estos programas y también la documentación referente a todo ello (tanto la documentación de ayuda en el uso del software para sus usuarios como la documentación generada durante su construcción, parte de la cual también servirá para su mantenimiento posterior)"

2.2.4. El Lenguaje Unificado de Modelado (UML)

Según C. Martin, R. (2004). "El Lenguaje Unificado de Modelado (UML) es una notación grafica para dibujar diagramas de conceptos de software. Se puede utilizar para dibujar diagramas de un dominio del problema, un diseño del software propuesto o una implementación de un software ya completado". Fowler describe estos tres niveles diferentes como Conceptual, de Especificación y de Implementación"

Según Pressman, R. S. (2010). "Es un lenguaje estándar para escribir diseños de software. El UML puede usarse para visualizar, especificar, construir y documentar los artefactos de un sistema de software intensivo, En otras palabras, tal como los arquitectos de edificios crean planos para que los use una compañía constructora, los arquitectos de software crean diagramas de UML para ayudar a los desarrolladores de software a construir el software".

2.2.5. Método de Análisis de Arquitecturas de Software (SAAM)

El método de análisis de arquitectura de software (SAAM) es un método utilizado en la arquitectura de software para evaluar la arquitectura de un sistema. Fue el primer método de análisis de arquitectura de software documentado y se desarrolló a mediados de la década de 1990 para analizar la modificabilidad de un sistema, pero es útil para probar cualquier aspecto no funcional.

2.2.6. Modelo de "4+1" Vistas de la Arquitectura del Software

Según Kruchten, P. (1995). "El modelo 4+1 describe la arquitectura del software usando cinco vistas concurrentes. Tal como se muestra en la Figura 1, cada vista se refiere a un conjunto de intereses de diferentes stakeholders del sistema"

- **La vista lógica** describe el modelo de objetos del diseño cuando se usa un método de diseño orientado a objetos. Para diseñar una aplicación muy orientada a los datos, se puede usar un enfoque alternativo para desarrollar algún otro tipo de vista lógica, tal como diagramas de entidad-relación.
- **La vista de procesos** describe los aspectos de concurrencia y sincronización del diseño.
- **La vista física** describe el mapeo del software en el hardware y refleja los aspectos de distribución.
- **La vista de desarrollo** describe la organización estática del software en su ambiente de desarrollo.

Los diseñadores de software pueden organizar la descripción de sus decisiones de arquitectura en estas cuatro vistas, y luego ilustrarlas con un conjunto reducido de casos de uso

o escenarios, los cuales constituyen la quinta vista. La arquitectura evoluciona parcialmente a partir de estos escenarios.

En Rational, aplicamos la fórmula de Dwayne Perry y Alexander Wolf [9] de manera independiente para cada vista:

Arquitectura del software = {Elementos, Formas, Motivación/Restricciones}

Para cada vista definimos un conjunto de elementos (componentes, contenedores y conectores), captamos la forma y los patrones con que trabajan, y captamos la justificación y las restricciones, relacionando la arquitectura con algunos de sus requisitos.

Cada vista se describe en lo que llamamos “diagrama” (blueprint) que usa su notación particular. Los arquitectos también pueden usar estilos de arquitectura para cada vista, y por lo tanto hacer que coexistan distintos estilos en un mismo sistema.

El modelo de 4+1 vistas es bastante genérico: se puede usar otra notación y herramientas que las aquí descritas, así como también otros métodos de diseño, especialmente para las descomposiciones lógica y de procesos.

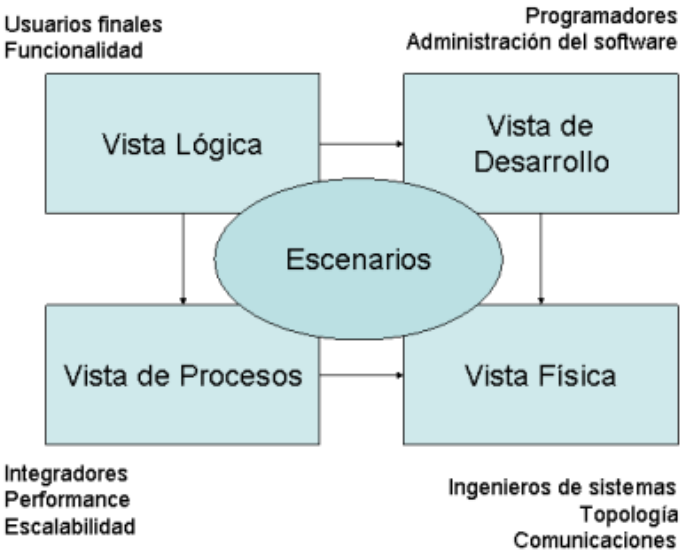

Figura 1. Modelo de "4+1" vistas

2.2.7. Casos de Uso

Según Jacobson et al., 1993. "Los casos de uso son una técnica de descubrimiento de requerimientos que se introdujo por primera vez en el método Objectory"

Según Sommerville, I. (2005). "Ahora se ha convertido en una característica fundamental del modelado de lenguaje unificado. En su forma más sencilla, un caso de uso identifica a los actores implicados en una interacción, y nombra el tipo de interacción. Entonces, esto se complementa con información adicional que describe la interacción con el sistema. La información adicional puede ser una descripción textual, o bien, uno o más modelos gráficos como una secuencia UML o un gráfico de estado".

Según Sommerville, I. (2005). "Los casos de uso se documentan con el empleo de un diagrama de caso de uso de alto nivel. El conjunto de casos de uso representa todas las interacciones posibles que se describirán en los requerimientos del sistema. Los actores en el proceso, que pueden ser individuos u otros sistemas, se representan como figuras sencillas".

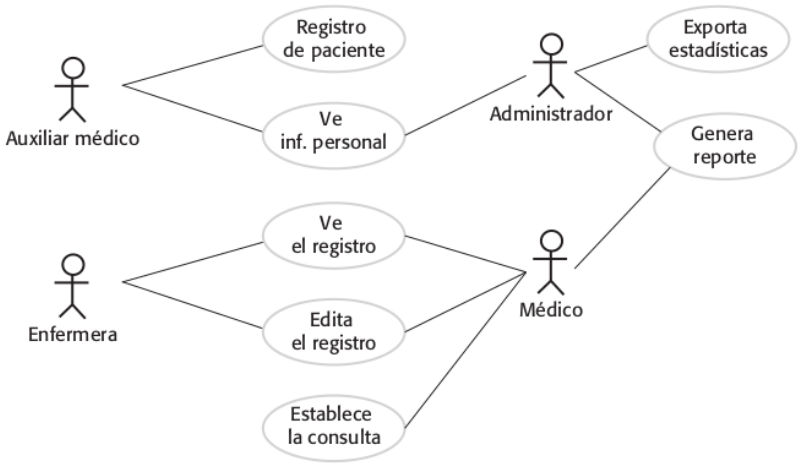

Figura 2. Casos de uso para el MHC-PMS

2.2.8. Diagrama de Secuencias

Según Sommerville, I. (2011). "Los diagramas de secuencia en el UML se usan principalmente para modelar las interacciones entre los actores y los objetos en un sistema, así como las interacciones entre los objetos en sí. El UML tiene una amplia sintaxis para diagramas de secuencia, lo cual permite muchos tipos diferentes de interacción a modelar. Como aquí no hay espacio para cubrir todas las posibilidades, sólo nos enfocaremos en lo básico de este tipo de diagrama."

Según Sommerville, I. (2011). "Como sugiere el nombre, un diagrama de secuencia muestra la sucesión de interacciones que ocurre durante un caso de uso particular o una instancia de caso de uso".

Según Sommerville, I. (2011). "Los objetos y actores que intervienen se mencionan a lo largo de la parte superior del diagrama, con una línea punteada que se dibuja verticalmente a partir de éstos. Las interacciones entre los objetos se indican con flechas dirigidas. El

rectángulo sobre las líneas punteadas indica la línea de vida del objeto tratado (es decir, el tiempo que la instancia del objeto está involucrada en la computación). La secuencia de interacciones se lee de arriba abajo. Las anotaciones sobre las flechas señalan las llamadas a los objetos, sus parámetros y los valores que regresan. En este ejemplo, también se muestra la notación empleada para exponer alternativas. Un recuadro marcado con "alt" se usa con las condiciones indicadas entre corchetes".

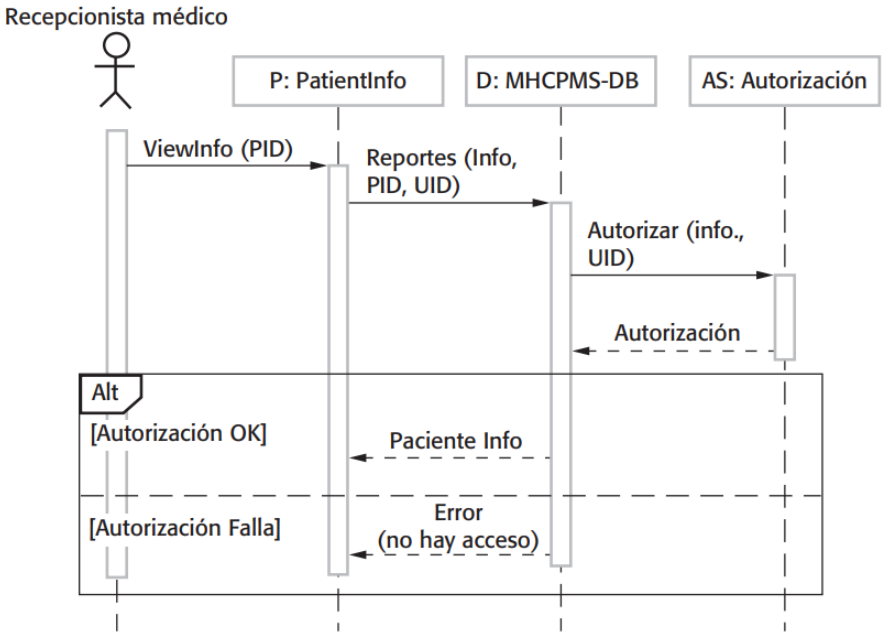

Figura 3. Diagrama de secuencia para "ver información del paciente"

2.2.9. Diagrama de Clase

Según Pressman, R. S. (2001). "Un modelo de clases es una descripción de las clases en un sistema y sus relaciones. No describe el comportamiento dinámico del sistema, por ejemplo, el comportamiento de objetos individuales. El primer elemento de un diagrama de clases es una descripción de clases individuales"

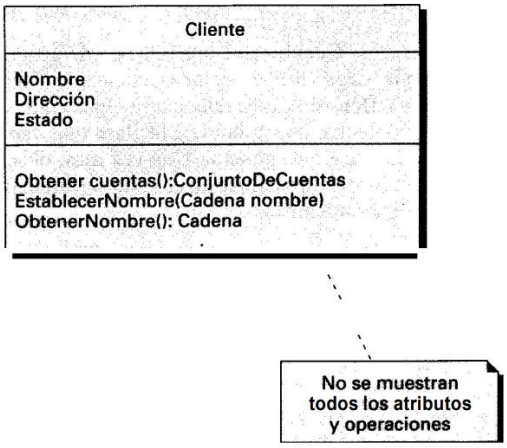

Figura 4. Diagrama de Clase

2.2.10. Diagrama de Componentes

Según Sommerville, I. (2011). "Un modelo de componentes es una definición de estándares para implementación, documentación y despliegue de componentes. Estos estándares se establecen con la finalidad de que los desarrolladores de componentes se aseguren de que éstos pueden interoperar".

Según Lau y Wang, 2007. "También funcionan para proveedores de infraestructuras de ejecución de componentes que ofrecen middleware para apoyar la ejecución de componentes. Se han propuesto muchos modelos de componentes, pero ahora los modelos más importantes son el modelo WebServices, el modelo Enterprise Java Beans (EJB) de Sun, y el modelo .NET de Microsoft".

Según Weinreich y Sametinger (2001). "analizan los elementos básicos de un modelo ideal de componentes. Este diagrama muestra que los elementos de un modelo de componentes definen las interfaces de componentes, la información que necesita usar el componente en un programa y cómo debe implementarse un componente".

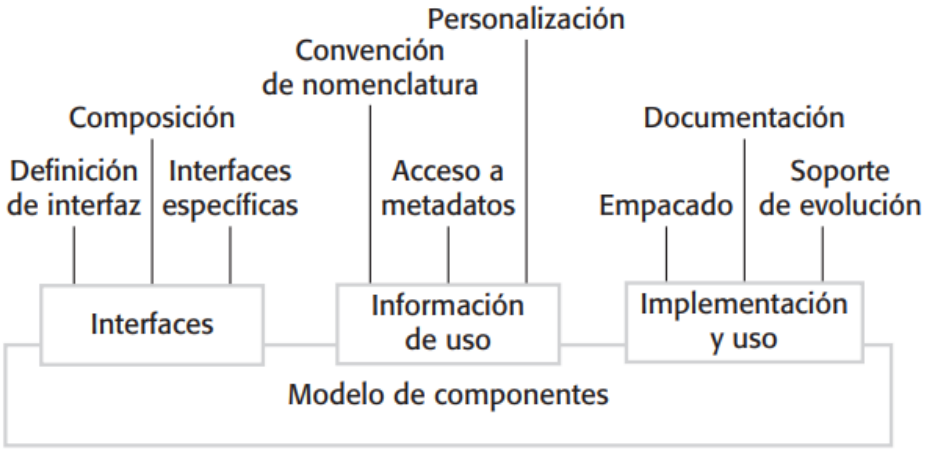

Figura 5. Elementos básicos de un modelo de componentes

2.2.11. Diagrama de Despliegue

Según Sommerville, I. (2011). "Los diagramas de despliegue UML muestran cómo los componentes de software se despliegan físicamente en los procesadores; es decir, el diagrama de despliegue muestra el hardware y el software en el sistema, así como el middleware usado para conectar los diferentes componentes en el sistema. En esencia, los diagramas de despliegue se pueden considerar como una forma de definir y documentar el entorno objetivo".

Según Pressman, R. S. (2010). "El diagrama de despliegue UML se utiliza en situaciones en las que deben considerarse arquitecturas de configuración compleja".

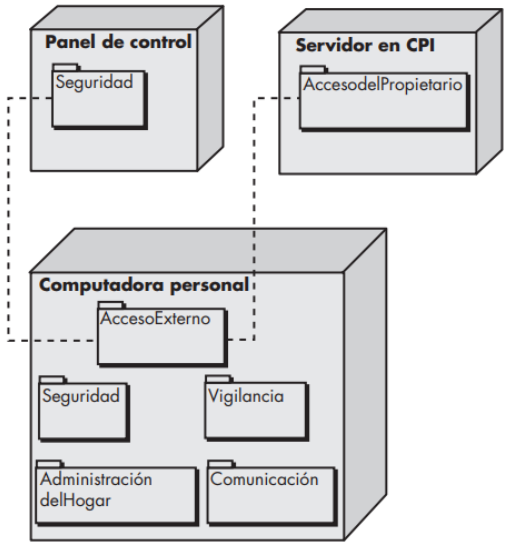

Figura 6. Diagrama de despliegue UML.

2.2.12. Diagrama de Paquetes

Un diagrama de paquetes es un diagrama de estructura cuyo contenido es, principalmente, paquetes y sus relaciones. La distinción entre los diversos tipos de diagramas de estructura (clases, objetos y paquetes) es relativa. Todos pueden incluir:

- Como nodos del grafo: Clases, Interfaces, Instancias o Paquetes.
- Y como arcos (relaciones): Agregaciones, asociaciones, composiciones, dependencias, generalizaciones, realizaciones, dependencias de uso, y fusiones, importaciones y accesos entre paquetes.

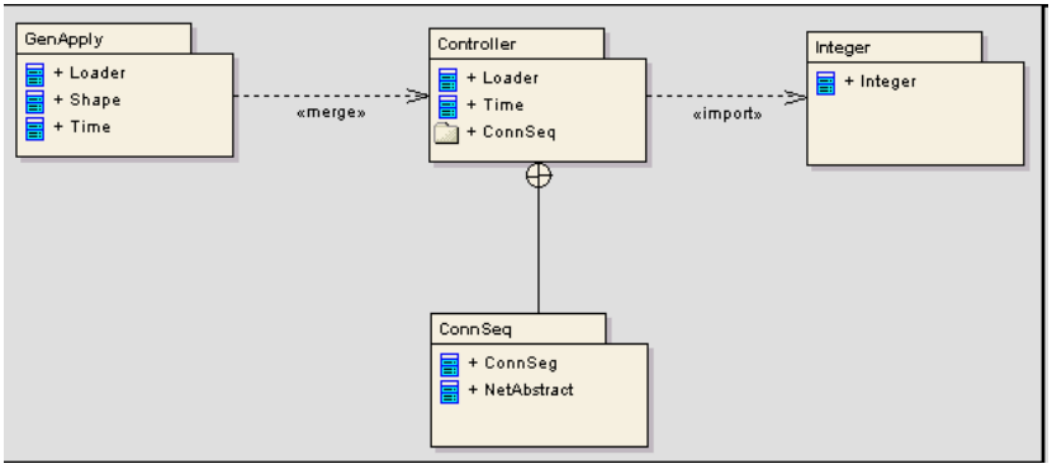

Figura 7. Diagrama de Paquetes

2.2.13. Diagrama de Actividades

Según Pressman, R. S. (2010). "El diagrama de actividad UML enriquece el caso de uso al proporcionar una representación gráfica del flujo de interacción dentro de un escenario específico. Un diagrama de actividades es similar a uno de flujo, y utiliza rectángulos redondeados para denotar una función específica del sistema, flechas para representar flujo a través de éste, rombos de decisión para ilustrar una ramificación de las decisiones (cada flecha que salga del rombo se etiqueta) y líneas continuas para indicar que están ocurriendo actividades en paralelo".

Según Pressman, R. S. (2010). "El diagrama de actividades permite representar la secuencia, condición y repetición todos los elementos de que consta la programación estructurada y es descendiente de un diseño gráfico anterior (que todavía se utiliza mucho) llamado diagrama de flujo. Como cualquier diagrama de actividades, el de flujo es muy simple"

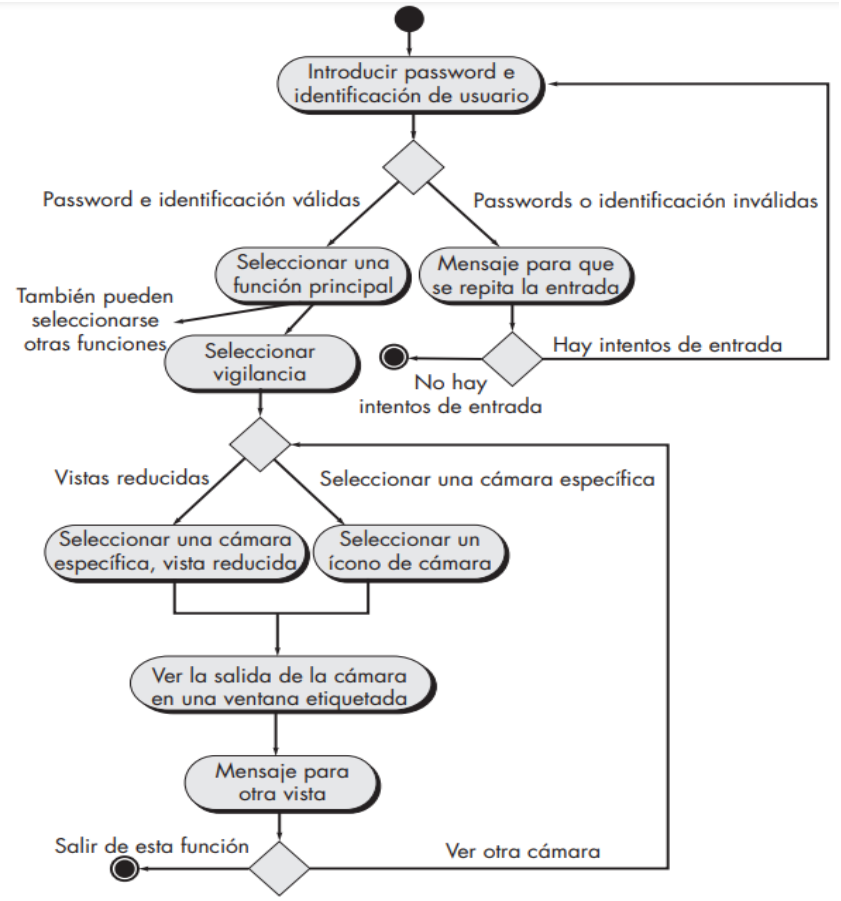

Figura 8. Diagrama de Actividades

2.2.14. Modelo Entidad Relación

Propuesto por Peter Chen en 1976, en modelo Entidad-Relación (también conocido como modelo E-R) es el modelo conceptual de alto nivel más extendido en la actualidad. Presenta un alto nivel de abstracción y se fundamenta en percibir la realidad como una serie de entidades (objetos que existen en la realidad) y de relaciones entre esos objetos. Tanto las entidades como las relaciones contienen atributos.

Según Martinez López, F. J., & Gallegoz Ruiz, A. (2017). "Se trata del modelo de datos más extendido en el mundo. Presenta un alto nivel de abstracción y es un modelo basado en percibir la realidad como una serie de entidades (objetos que existen en la realidad) y de

relaciones entre esos objetos. Tanto las entidades como las relaciones contienen atributos (información que los definen), que nos servirán para diferenciar cada entidad de otras similares".

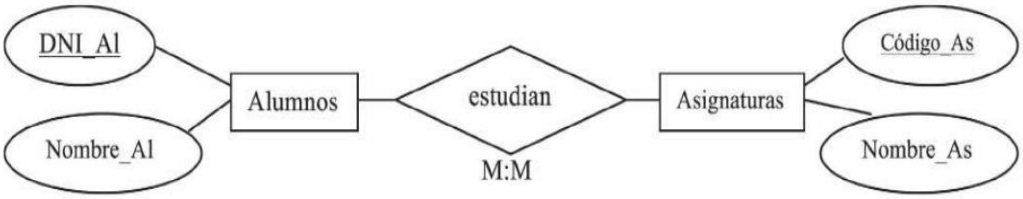

Figura 9. Modelo Entidad Relación

2.2.15. Modelo Relacional

La segunda generación de las bases de datos surgió de la mano de Edgar Frank Codd (23/08/1923 – 18/04/2003), ingeniero de la empresa IBM, con su artículo "Un modelo relacional de datos para grandes bancos de datos compartidos", en 1970, en el cual definió el modelo relacional y las famosas 12 reglas de Codd para los sistemas de datos relacionales. Por tal motivo, en la literatura, Codd está considerado como el padre de las bases de datos relacionales. De este modo, también surgirían las bases de datos comerciales. A partir de los trabajos de Codd se desarrollaron los primeros Sistemas de Gestión de Bases de Datos Relacionales (DBMS relacionales), destacando el proyecto denominado System R, de IBM, el cual mejoró notablemente el rendimiento del modelo relacional, haciéndolo más competitivo que los anteriores modelos jerárquicos y de red. Otro DBMS relacional importante es el de Oracle Corporation (empresa que toma como nombre el de dicho software, Oracle).

Según Martinez López, F. J., & Gallegoz Ruiz, A. (2017). "En este modelo se emplea una colección de tablas para expresar tanto los datos, como sus relaciones. Cada tabla contiene

varias columnas (con nombre único) y las relaciones se representan mediante columnas comunes en dichas tablas".

Tabla "Alumnos"

DNI_Al	Nombre_Al	Edad_Al	Teléfono_Al	Email_Al
11111111A	Francisco Javier	12	666111111	fran@gmail.com
22222222B	Amalia	10	666222222	amalia@gmail.com
33333333C	Manuel	15	666333333	manuel@gmail.com
44444444D	Mª del Mar	12	666444444	mmar@gmail.com

Tabla "Asignaturas"

Código_As	Nombre_As	NumCréditos_As	Curso_As
001	Bases de Datos	6	3
002	Redes	9	3
003	Programación	6	1
004	Sistemas Operativos	4,5	2

Tabla "Estudian"

DNI_Al	Código_As
11111111A	003
22222222B	004
33333333C	001
33333333C	002
44444444D	003

Figura 10. Modelo Relacional

2.2.16. Patrones arquitectónicos

Según Garlan y Shaw, 1993. "La arquitectura de un sistema de software puede basarse en un patrón o un estilo arquitectónico particular. Un patrón arquitectónico es una descripción de una organización del sistema**"**., tal como una organización cliente-servidor o una arquitectura por capas. Los patrones arquitectónicos captan la esencia de una arquitectura que se usó en diferentes sistemas de software. Usted tiene que conocer tanto los patrones comunes, en que éstos se usen, como sus fortalezas y debilidades cuando se tomen decisiones sobre la arquitectura de un sistema.

La idea de los patrones como una forma de presentar, compartir y reutilizar el conocimiento sobre los sistemas de software se usa ahora ampliamente. El origen de esto fue la publicación de un libro acerca de patrones de diseño orientados a objetos (Gamma et al., 1995), que incitó el desarrollo de otros tipos de patrón, como los patrones para el diseño organizacional (Coplien y Harrison, 2004), patrones de usabilidad (Usability Group, 1998), interacción (Martin y Sommerville, 2004), administración de la configuración (Berczuk y Appleton, 2002), etcétera.

Los patrones arquitectónicos se propusieron en la década de 1990, con el nombre de "estilos arquitectónicos" (Shaw y Garlan, 1996), en una serie de cinco volúmenes de manuales sobre arquitectura de software orientada a patrones, publicados entre 1996 y 2007 (Buschmann et al., 1996; Buschmann et al., 2007a; Buschmann et al., 2007b; Kircher y Jain, 2004; Schmidt et al., 2000).

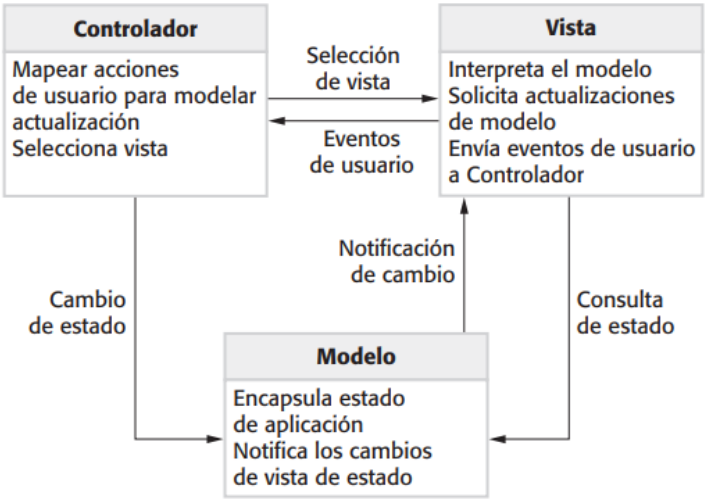

Figura 11. La organización del MVC

2.2.17. Requerimientos funcionales

Según Sommerville, I. (2011). “Son enunciados acerca de servicios que el sistema debe proveer, de cómo debería reaccionar el sistema a entradas particulares y de cómo debería comportarse el sistema en situaciones específicas. En algunos casos, los requerimientos funcionales también explican lo que no debe hacer el sistema”.

Los requerimientos funcionales para un sistema refieren lo que el sistema debe hacer. Tales requerimientos dependen del tipo de software que se esté desarrollando, de los usuarios esperados del software y del enfoque general que adopta la organización cuando se escriben los requerimientos. Al expresarse como requerimientos del usuario, los requerimientos funcionales se describen por lo general de forma abstracta que entiendan los usuarios del sistema. Sin embargo, requerimientos funcionales más específicos del sistema detallan las funciones del sistema, sus entradas y salidas, sus excepciones, etcétera.

2.2.18. Requerimientos no funcionales

Según Sommerville, I. (2011). “Son limitaciones sobre servicios o funciones que ofrece el sistema. Incluyen restricciones tanto de temporización y del proceso de desarrollo, como impuestas por los estándares. Los requerimientos no funcionales se suelen aplicar al sistema como un todo, más que a características o a servicios individuales del sistema”.

Los requerimientos no funcionales, como indica su nombre, son requerimientos que no se relacionan directamente con los servicios específicos que el sistema entrega a sus usuarios. Pueden relacionarse con propiedades emergentes del sistema, como fiabilidad, tiempo de respuesta y uso de almacenamiento. De forma alternativa, pueden definir restricciones sobre la implementación del sistema, como las capacidades de los dispositivos I/O o las representaciones de datos usados en las interfaces con otros sistemas. Los requerimientos no

funcionales, como el rendimiento, la seguridad o la disponibilidad, especifican o restringen por lo general características del sistema como un todo. Los requerimientos no funcionales a menudo son más significativos que los requerimientos funcionales individuales. Es común que los usuarios del sistema encuentren formas para trabajar en torno a una función del sistema que realmente no cubre sus necesidades. No obstante, el fracaso para cubrir los requerimientos no funcionales haría que todo el sistema fuera inútil. Por ejemplo, si un sistema de aeronave no cubre sus requerimientos de fiabilidad, no será certificado para su operación como dispositivo seguro; si un sistema de control embebido fracasa para cubrir sus requerimientos de rendimiento, no operarán correctamente las funciones de control.

2.2.19. Metodología Architecture Tradeoff Analysis Method (ATAM).

Según Delgado, A., Castro, A., & Germán, M. (2017). "El Architecture Tradeoff Analysis Method (ATAM) es una metodología para evaluar Arquitecturas de Software que principalmente evalúa la adecuación de la Arquitectura de Software definida con respecto a los atributos de calidad especificados para el sistema. Surge confluyendo ideas y técnicas de tres áreas: la noción de estilos o patrones de arquitectura, el análisis de atributos de calidad y el método Software Architecture Analysis Method (SAAM) que es el predecesor del ATAM. Obtiene su nombre del concepto de que a veces no será posible cumplir con todos los atributos de calidad definidos y se deberán realizar concesiones mutuas (o tradeoffs) entre estos, para obtener el balance adecuado".

2.4. MARCO CONTEXTUAL

El desarrollo de este proyecto tendrá una duración 1 semestre en el transcurso académico de la especialización en Ingeniería de Software, se llevará a cabo en el colegio Integrado Atalaya ubicado en la ciudad de Cúcuta. Este colegio tiene la necesidad de establecer si es viable el desarrollo de su propio software académico el cual estará basado en la Arquitectura de Software planteada.

3. DISEÑO METODOLOGICO

3.1. METODOLOGIA DE INVESTIGACIÓN

Según Hurtado De Barrera, J. (2010). “La Investigación proyectiva tiene como objetivo diseñar o crear propuestas dirigidas a resolver determinadas situaciones. Los provectos de arquitectura e ingeniería, el diseño de maquinarias, la creación de programas de intervención social, el diseño de programas de estudio, los inventos, la elaboración de programas informáticos, entre otros, siempre que estén sustentados en un proceso de investigación, son ejemplos de investigación proyectiva. Este tipo de investigación potencia el desarrollo tecnológico.”

La metodología de investigación usada para el proyecto es la Investigación proyectiva para elaborar una propuesta de implementación tecnológica la cual consta del diseño de una Arquitectura de Software para una plataforma de Gestión Académica. Para la realización del proyecto se realizó la recolección de requerimientos, se definieron los requerimientos funcionales y no funcionales, se realizó una búsqueda de plataformas de gestión académica para la cotización de dichas plataformas, se realizó el diseño de la arquitectura de software, se realizó un presupuesto sobre el costo del desarrollo e implementación teniendo en cuenta el valor por hora que se cobra en el mercado y los costos anuales de los servicios en la nube.

3.2. METODO DE INVESTIGACIÓN

El método sistémico es un proceso mediante el cual se relacionan hechos aparentemente aislados y se formula una teoría que unifica los diversos elementos. Consiste en la reunión racional de varios elementos dispersos en una nueva totalidad, este se presenta más en el planteamiento de la hipótesis. El investigador sintetiza las superaciones en la imaginación para establecer una explicación tentativa que someterá a prueba.

El método sistémico está dirigido a modelar el objeto mediante la determinación de sus componentes, así como las relaciones entre ellos. Esas relaciones determinan por un lado la estructura del objeto y por otro su dinámica.

3.2. DETERMINACIÓN DE FUENTES

En la búsqueda de una fuente para identificar la metodología de investigación que usaría para el proyecto encontré el libro "Metodología de la Investigación – guía para la comprensión holística de la ciencia" escrito por Jacqueline Hurtado de Barrera el cual está completo, especifico y actualizado y cuenta con los conceptos de los diferentes tipos de investigación entre esas la Investigación Proyectiva por tal motivo decidí elegirlo como fuente de investigación para la realización de mi proyecto.

3.3. FASES DEL PROYECTO

INICIAL	Identificar metodología y método de investigación. Identificar el modelo de Arquitectura. Identificar el método de evaluación de Arquitecturas	Se identificó la metodología y el método de investigación que se utilizara para este proyecto. Se identifico el modelo y método de evaluación de la Arquitectura.
INTERMEDIA	Investigar plataformas académicas con costos y funcionalidades.	Se procedió a investigar las diferentes plataformas académicas que prestan el servicio en la ciudad de Cúcuta, los costos que manejan, las funcionalidades con las que cuentan y los

		servicios que brindan al colegio
FINAL	Diseñar la Arquitectura de Software según los requerimientos. Validar la Arquitectura de Software Elaborar el documento final	Se hizo la toma y análisis de requerimientos para el diseño de la Arquitectura de Software utilizando el Modelo de “4+1” Vistas de laArquitectura del Software. Se valido la Arquitectura de Software utilizando la metodología Architecture Tradeoff Analysis Method (ATAM). Se realizó un presupuesto estimado del desarrollo. Se realizo el documento final.

Tabla 1. Fases del Proyecto

3.4.PLATAFORMAS ACADEMICAS Y SUS FUNCIONALIDADES

3.4.1. Phidias

Phidias es una plataforma de gestión académica que fue desarrollada por los hermanos Santiago y Juan Sebastián Cortés en la ciudad de Bogotá y se utiliza en 7 países los cuales son Colombia, España, Panamá, México, Cuba, Francia y Bélgica, es una plataforma muy Completa que cuenta con 4 módulos los cuales son: Módulo Académico, Módulo Tesorería, Módulo Administración y Módulo Comunicación.

3.5. DISEÑO ARQUITECTURA DE SOFTWARE

El objetivo es el diseño de la Arquitectura de Software utilizando el Modelo de "4+1" Vistas de la Arquitectura del Software el cual plantea utilizar 4 vistas las cuales son Vista Lógica, Vista de Desarrollo, Vista de Procesos y Vista Física, por último, se plantea el escenario que hace parte de este modelo 4+1. En las diferentes vistas del modelo se trabajan con los Casos de Uso, Diagrama de Secuencias, Diagrama de Clase, Diagrama de Componentes, Diagrama de Despliegue, Diagrama de Paquetes, Diagrama de Actividades, Modelo Entidad Relación, Modelo Relacional, Requerimientos funcionales y Requerimientos no funcionales.

3.5.1. Lista de Características

- **Roles de Usuarios:** La aplicación web de gestión académica cuenta con diferentes roles los cuales tienen restricciones a ciertas funcionalidades de la aplicación, para acceder al sistema se necesita de un usuario con contraseña y un rol definido, los roles de usuario que se plantean para la aplicación son: Administrador Sistema, Administrador Colegio, Coordinador, Docente, Estudiante, Padre de Familia.

    - ✓ **Rol Administrador Sistema:** Este rol de usuario cuenta con acceso total a las diferentes funcionalidades del sistema las cuales incluyen la de gestión académica y la gestión del sistema.
    - ✓ **Rol Administrador Colegio:** Este rol de usuario cuenta con acceso total al grupo de funcionalidades de gestión académica las cuales

incluyen estudiantes, docentes, usuarios, asignaturas, grados, cursos, niveles académicos, calificaciones, boletines, etc.

- ✓ **Rol secretaria:** Este rol de usuario cuenta con acceso a la gestión de estudiantes.
- ✓ **Coordinador:** Este rol de usuario cuenta con acceso a las funcionalidades de asignaturas, grados, cursos, niveles académicos, calificaciones, etc.
- ✓ **Docente:** Este rol de usuario cuenta con acceso a las funcionalidades que permiten al docente registrar las calificaciones del estudiante y los logros.
- ✓ **Estudiante:** Este rol de usuario cuenta con acceso a las funcionalidades de ver las notas, horario y asignaturas del estudiante.
- ✓ **Acudiente:** Este rol de usuario cuenta con acceso a las funcionalidades de ver las notas, horario y asignaturas del estudiante.

- **Diseño Responsive Interface Usuario:** La resolución de la interfaz de usuario debe ser responsive lo cual permite que se ajuste a las diferentes resoluciones de dispositivos móviles.

3.5.2. Recursos Financieros

Para implementar el desarrollo del software de gestión Académica de la Arquitectura diseñada el colegio Integrado Atalaya debe realizar una inversión que cubra los costos de contratar los Desarrolladores y el servicio en la nube un costo que es anual.

3.5.3. Recursos Humanos

Se debe contar con desarrolladores de software que tengan los siguientes conocimientos:

1. Lenguaje de programación Javascript
2. NodeJs
3. Framework Angular
4. Framework NestJS
5. Base de Datos PostgreSQL
6. Servicio en la Nube Azure.

3.5.4. Recursos Tecnológicos

En recursos tecnológicos debe contar con los siguientes recursos tecnológicos:

1. Computadores para los Desarrolladores de Software
2. Servicio en la Nube Azure
3. Celular para las pruebas que realizaran los Desarrolladores de Software

3.5.5. Requisitos

Se identifican los requisitos funcionales y no funcionales con los que contara la Arquitectura de Software para la Gestión Académica del colegio, estos requisitos son el resultado de la información recolectada en las reuniones que se realizaron con el personal que labora en el colegio.

3.5.5.1. Requisitos Funcionales

Id Requisito	Requerimiento	Actor(es) Involucrados	Descripción
RF1	Inicio de Sesión	Administrador, Estudiante, Coordinador, Docente, Secretaria y Acudiente	El sistema carga el formulario de iniciar sesión y el usuario digita sus credenciales los cuales el sistema validara dicha información para verificar que sus credenciales son válidas y permitir el acceso a la plataforma.
RF2	Gestión Año Lectivo	Administrador	Se registra, actualiza y elimina la información correspondiente al año lectivo.
RF3	Gestión Estudiante	Administrador, Secretaria	Se registra, actualiza y elimina la información de estudiantes.
RF4	Gestión Docente	Administrador	Se registra, actualiza y elimina la información de docentes.
RF5	Gestión Curso	Administrador	Se registra, actualiza y elimina la información de los cursos.
RF6	Gestión Asignatura	Administrador	Se registra, actualiza y elimina la información de asignaturas.
RF7	Gestión Área	Administrador	Se registra, actualiza y elimina la información de áreas.
RF8	Matricular Alumno	Administrador, Secretaria	Se realiza la búsqueda del usuario para matricularlo en el curso y año lectivo.
RF9	Gestión Usuarios	Administrador	Se registra, actualiza y elimina la información de usuarios.
RF10	Gestión Grados	Administrador	Se registra, actualiza y elimina la información de grados.

RF11	Gestión Horarios	Administrador, Coordinador	Se registra, actualiza y elimina los horarios del año lectivo
RF12	Gestión Periodos Académicos	Administrador	Se registra, actualiza y elimina los periodos académicos.
RF13	Gestión Carga Académica	Administrador, Coordinador, Docente	Se registran las notas de cada estudiante por materia
RF14	Gestión de Logros Académicos	Administrador	Se registra, actualiza y elimina los logros académicos.
RF15	Notas Académicas de estudiantes por materia	Estudiante, Acudiente	El estudiante podrá visualizar sus notas académicas obtenidas por materia que tiene asignada.
RF16	Carga Académica Estudiante	Estudiante, Acudiente	El usuario podrá visualizar la carga académica que tiene asignada para el año lectivo.
RF17	Horario Estudiante	Estudiante, Acudiente	El usuario podrá visualizar el horario asignado por semana en el año lectivo.
RF18	Carga Académica Docente	Docente	El docente podrá visualizar la carga académica que tiene asignada por grados para el año lectivo.
RF19	Horario Docente	Docente	El docente podrá visualizar el horario que tiene asignado por semana de las horas que debe estar presente por grado.
RF20	Gestión Roles	Administrador	El usuario con permisos de administrador podrá gestionar los diferentes roles de la plataforma académica.
RF21	Cierre de Sesión	Administrador, Estudiante, Coordinador, Docente y Acudiente	El usuario solicita al sistema el cierre de sesión.

Tabla 2. Requisitos Funcionales

3.5.5.2.Requisitos no Funcionales

Id Requisito	Requerimiento	Descripción
RNF1	Framework Angular	Se utilizará el framework Angular desde la capa del cliente el cual maneja el lenguaje de programación Javascript.

RNF2	Framework NestJS	Se utilizará el framework NestJS desde la capa del servidor el cual maneja el lenguaje de programación Javascript.
RNF3	Disponibilidad 24/7	Se requiere la disponibilidad 24/7 para el aplicativo de gestión académica.
RNF4	Restricción Funcionalidades	Se requiere la restricción de usuarios a las funcionalidades por el rol de usuario.
RNF5	Mensajes de error	Se requiere que el aplicativo presente mensajes de errores de fácil comprensión para el usuario cuando se presentan inconvenientes en el sistema.
RNF6	Interfaz de Usuario	Se requiere que el sistema presente una interfaz de usuario amigable, responsive y la paleta de colores de la institución académica.
RNF7	Ambiente WEB	Se requiere que el sistema funcione en ambiente web y sea accesible desde cualquier navegador web.

Tabla 3. Requisitos no Funcionales

3.5.5.3. Restricciones

- El rol de estudiante y acudiente solo tienen acceso a las funcionalidades que permiten visualizar horario, visualizar notas, visualizar carga académica.
- El rol de docente y coordinador solo tienen acceso a las funcionalidades que permiten la gestión académica.
- El rol de secretaria solo tiene acceso a las funcionalidades de matricular estudiante, gestión estudiante.
- Se requiere acceso a internet desde los diferentes dispositivos en los cuales se utilizará la plataforma.
- Compatibilidad con el navegador web Google Chrome y Firefox.

3.5.5.4. Modelo del Dominio

El modelo de dominio es el artefacto clave del análisis orientado a objetos y nos muestra las clases conceptuales significativas en un dominio del problema. La siguiente figura representa el modelo de dominio del sistema gestión académica.

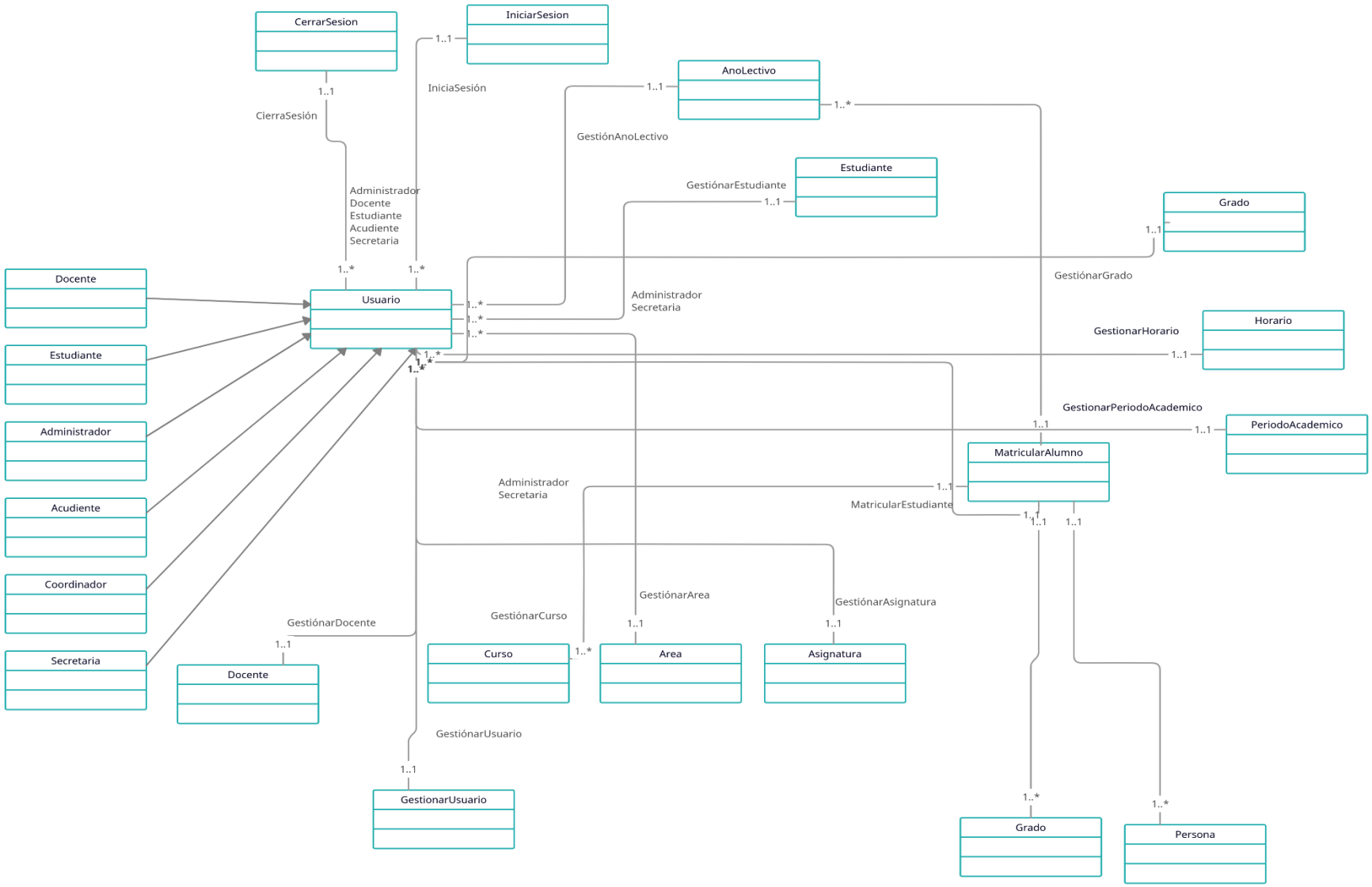

Figura 12. Modelo de Dominio

A continuación, se explica el modelo de dominio planteado para mayor comprensión:

- Se plantea para el sistema 6 tipos de usuario los cuales son administrador, secretaria, estudiante, acudiente, docente y coordinador. Estos usuarios para poder tener acceso al sistema deben primero iniciar sesión.
- Los usuarios tienen la función de cerrar sesión al terminar de utilizar la plataforma.
- El administrador tiene acceso total a las funcionalidades del sistema.
- El administrador y la secretaria tienen acceso a la funcionalidad de matricular estudiante.

3.5.6. Escenarios

Esta vista va a ser representada por los casos de uso los cuales fueron establecidos como resultado de la recolección de información, en esta vista se identifican los usuarios o actores que van a interactuar con el sistema y los diferentes escenarios en los que participa cada uno. Esta vista va a tener la función de unir y relacionar las otras 4 vistas, esto quiere decir que desde un caso de uso podemos ver cómo se van ligando las otras 4 vistas, con lo que tendremos una trazabilidad de componentes, clases, equipos, paquetes, etc.

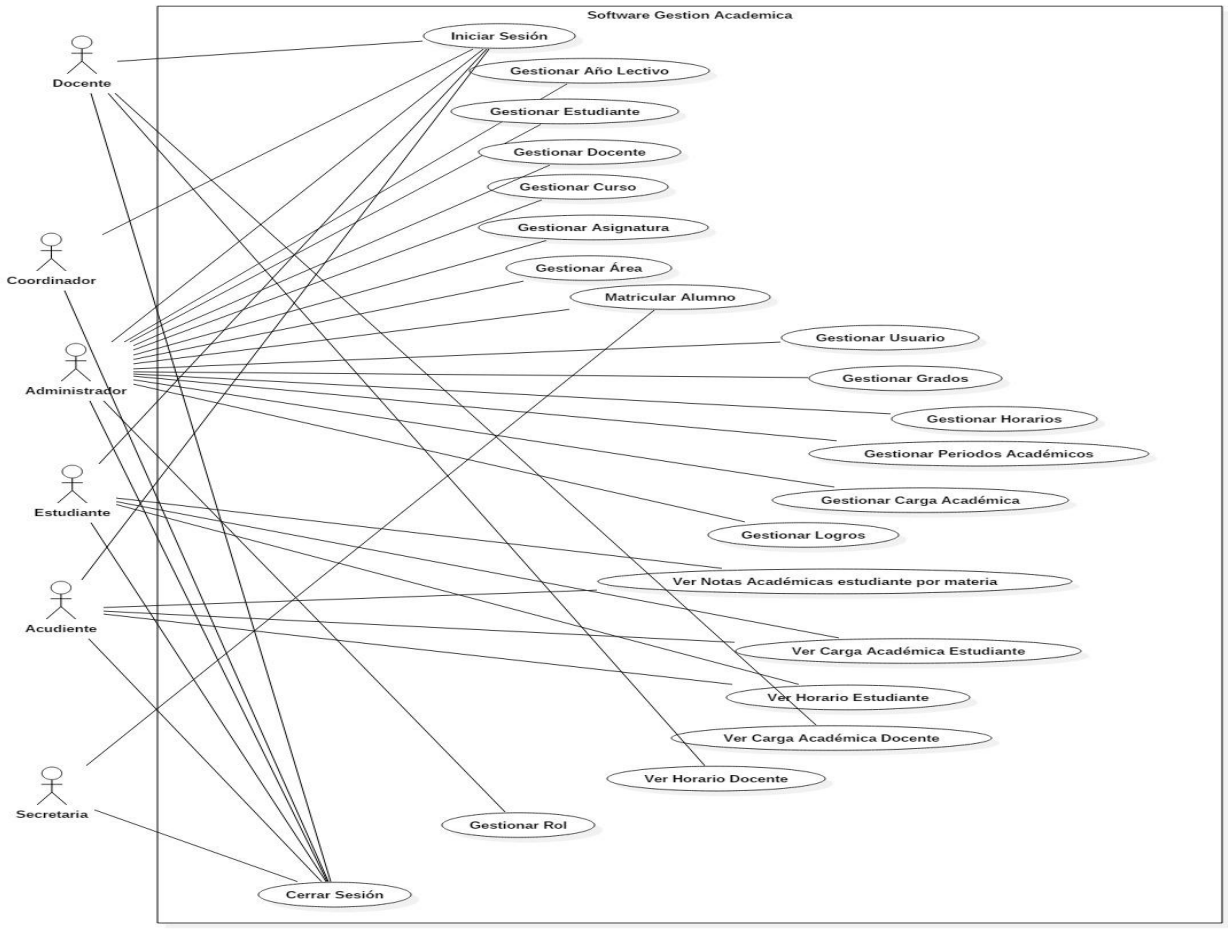

Figura 13. Diagrama Caso de Uso General

3.5.6.1. Casos de Uso

DETALLE DE CASO DE USO	
Numero	**RF1**
Nombre	**Iniciar Sesión**
Descripción	El sistema carga el formulario de iniciar sesión y el usuario digita sus credenciales los cuales el sistema validara dicha información para verificar que sus credenciales son válidas y permitir el acceso a la plataforma.
Actores	- Administrador, Estudiante, Coordinador, Docente, Secretaria y Acudiente

Precondiciones	- El usuario debe digitar el usuario y contraseña.
Flujo Normal	- Se digita el usuario y contraseña - Clic en el botón Iniciar Sesión - El sistema valida los datos recibidos y devuelve un mensaje de éxito o error. - El usuario recibe la respuesta del sistema - Los datos son correctos es redireccionado a la plataforma
Flujos Alternativos	- Si los datos no son correctos se muestra un mensaje de error.
Postcondiciones	- Se carga el Dashboard de la plataforma.
	-

Tabla 4. Iniciar Sesión

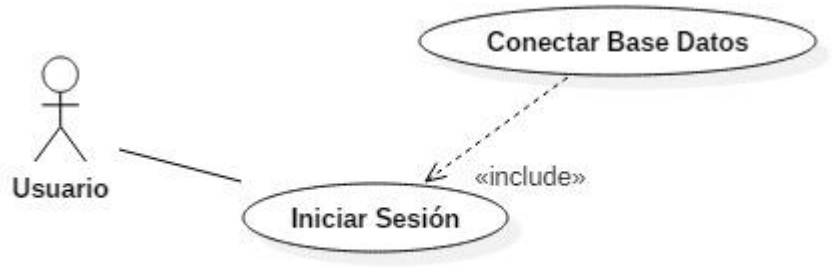

Figura 14. Diagrama Caso de Uso Iniciar Sesión

DETALLE DE CASO DE USO	
Numero	**RF2**
Nombre	**Gestionar Año Lectivo**
Descripción	El sistema carga la vista de Gestión Año Lectivo y permite al usuario la lista de registros, registrar, actualizar y eliminar la información correspondiente al año lectivo.
Actores	- Administrador
Precondiciones	- El usuario debe tener una sesión iniciada.

Flujo Normal	- El sistema carga la vista de Gestión Año Lectivo - El usuario selecciona la opción que desea realizar - Si ¿es para registrar? el usuario debe digitar la información del formulario que es obligatoria y luego hacer clic en el botón de registrar. - Si ¿es para actualizar? el usuario debe seleccionar el año lectivo a modificar y el sistema cargará el formulario con la información del año lectivo seleccionado y el usuario deberá modificar la información y luego clic en el botón de actualizar. - Si ¿es para eliminar? el usuario debe hacer clic en el botón de eliminar y luego el sistema abre un modal de confirmación el cual el usuario debe aceptar.
Flujos Alternativos	- Si surgió un error al tratar de registrar, actualizar o eliminar el sistema mostrara un mensaje de error.
Postcondiciones	- Se ejecuta la opción que el usuario realizo.

Tabla 5. Gestionar Año Lectivo

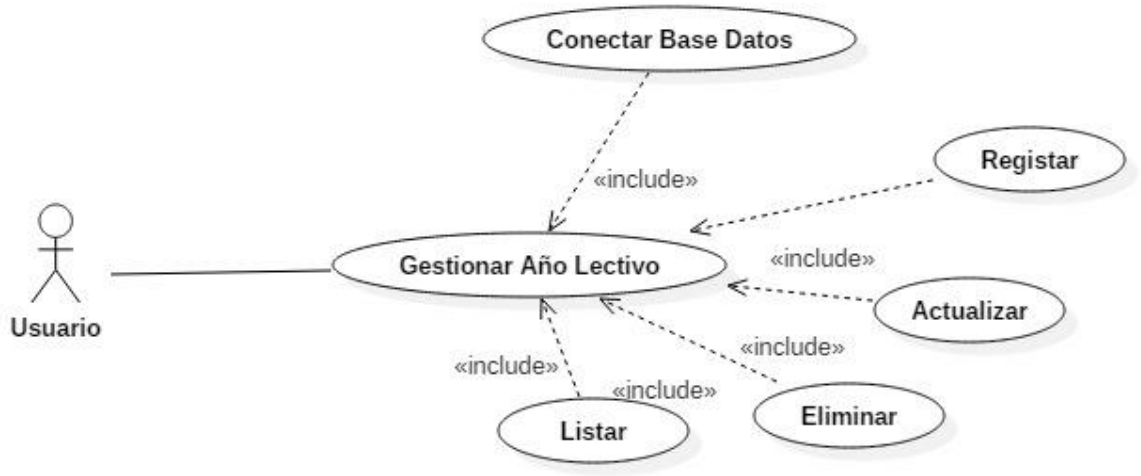

Figura 15. Diagrama Caso de Uso Gestionar Año Lectivo

DETALLE DE CASO DE USO	
Numero	**RF3**
Nombre	**Gestionar Estudiante**

Descripción	El sistema carga la vista de Gestión Estudiante y permite al usuario la lista de registros, registrar, actualizar y eliminar la información correspondiente al estudiante.
Actores	- Administrador
Precondiciones	- El usuario debe tener una sesión iniciada.
Flujo Normal	- El sistema carga la vista de Gestión Estudiante - El usuario selecciona la opción que desea realizar - Si ¿es para registrar? el usuario debe digitar la información del formulario que es obligatoria y luego hacer clic en el botón de registrar. - Si ¿es para actualizar? el usuario debe seleccionar el estudiante a modificar y el sistema cargará el formulario con la información del estudiante seleccionado y el usuario deberá modificar la información y luego clic en el botón de actualizar. - Si ¿es para eliminar? el usuario debe hacer clic en el botón de eliminar y luego el sistema abre un modal de confirmación el cual el usuario debe aceptar.
Flujos Alternativos	- Si surgió un error al tratar de registrar, actualizar o eliminar el sistema mostrara un mensaje de error.
Postcondiciones	- Se ejecuta la opción que el usuario realizo.

Tabla 6. Gestionar Estudiante

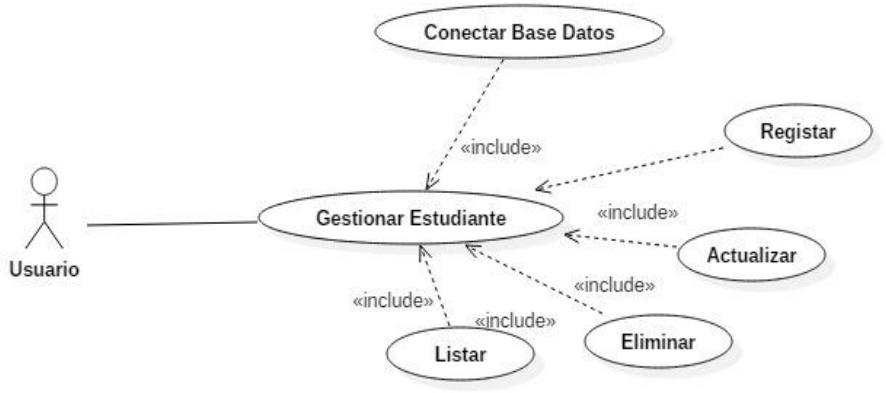

Figura 16. Diagrama Caso de Uso Gestionar Estudiante

DETALLE DE CASO DE USO	
Numero	**RF4**
Nombre	**Gestionar Docente**
Descripción	El sistema carga la vista de Gestión Docente y permite al usuario la lista de registros, registrar, actualizar y eliminar la información correspondiente al docente.
Actores	- Administrador
Precondiciones	- El usuario debe tener una sesión iniciada.
Flujo Normal	- El sistema carga la vista de Gestión Docente - El usuario selecciona la opción que desea realizar - Si ¿es para registrar? el usuario debe digitar la información del formulario que es obligatoria y luego hacer clic en el botón de registrar. - Si ¿es para actualizar? el usuario debe seleccionar el docente a modificar y el sistema cargará el formulario con la información del docente seleccionado y el usuario deberá modificar la información y luego clic en el botón de actualizar. - Si ¿es para eliminar? el usuario debe hacer clic en el botón de eliminar y luego el sistema abre un modal de confirmación el cual el usuario debe aceptar.
Flujos Alternativos	- Si surgió un error al tratar de registrar, actualizar o eliminar el sistema mostrara un mensaje de error.
Postcondiciones	- Se ejecuta la opción que el usuario realizo.

Tabla 7. Gestionar Docente

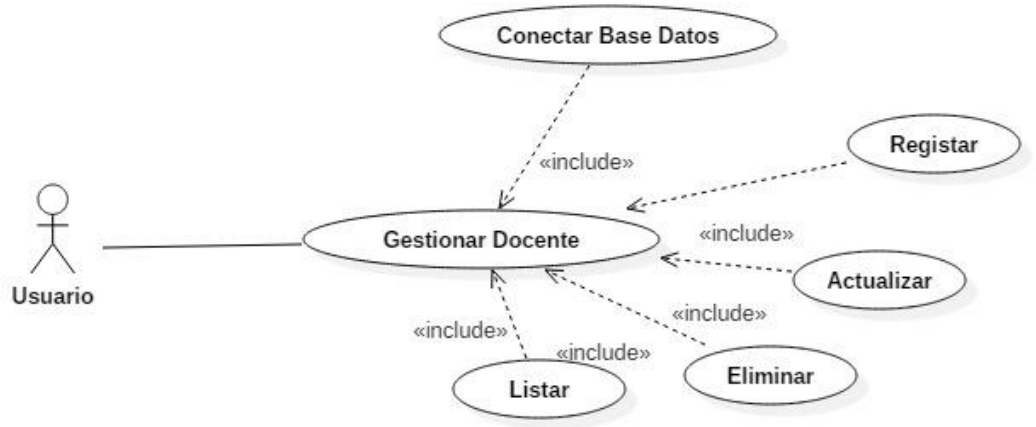

Figura 177. Diagrama Caso de Uso Gestionar Docente

DETALLE DE CASO DE USO	
Numero	**RF5**
Nombre	**Gestionar Curso**
Descripción	El sistema carga la vista de Gestión Curso y permite al usuario la lista de registros, registrar, actualizar y eliminar la información correspondiente al curso.
Actores	- Administrador
Precondiciones	- El usuario debe tener una sesión iniciada.
Flujo Normal	- El sistema carga la vista de Gestión Curso - El usuario selecciona la opción que desea realizar - Si ¿es para registrar? el usuario debe digitar la información del formulario que es obligatoria y luego hacer clic en el botón de registrar. - Si ¿es para actualizar? el usuario debe seleccionar el curso a modificar y el sistema cargará el formulario con la información del curso seleccionado y el usuario deberá modificar la información y luego clic en el botón de actualizar. - Si ¿es para eliminar? el usuario debe hacer clic en el botón de eliminar y luego el sistema abre un modal de confirmación el cual el usuario debe aceptar.

Flujos Alternativos	- Si surgió un error al tratar de registrar, actualizar o eliminar el sistema mostrara un mensaje de error.
Postcondiciones	- Se ejecuta la opción que el usuario realizo.

Tabla 8. Gestionar Curso

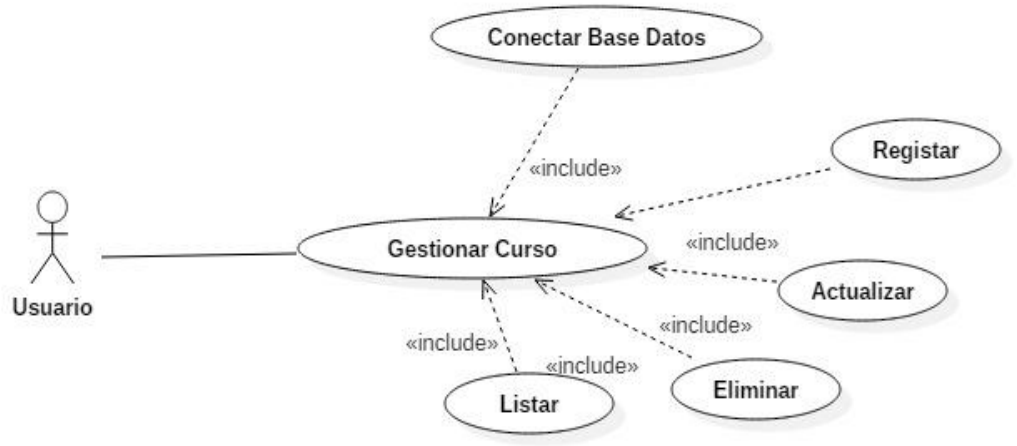

Figura 18. Diagrama Caso de Uso Gestionar Curso

DETALLE DE CASO DE USO	
Numero	**RF6**
Nombre	**Gestionar Asignatura**
Descripción	El sistema carga la vista de Gestión Asignatura y permite al usuario la lista de registros, registrar, actualizar y eliminar la información correspondiente a la asignatura.
Actores	- Administrador
Precondiciones	- El usuario debe tener una sesión iniciada.

Flujo Normal	- El sistema carga la vista de Gestión Asignatura - El usuario selecciona la opción que desea realizar - Si ¿es para registrar? el usuario debe digitar la información del formulario que es obligatoria y luego hacer clic en el botón de registrar. - Si ¿es para actualizar? el usuario debe seleccionar la asignatura a modificar y el sistema cargará el formulario con la información de la asignatura seleccionada y el usuario deberá modificar la información y luego clic en el botón de actualizar. - Si ¿es para eliminar? el usuario debe hacer clic en el botón de eliminar y luego el sistema abre un modal de confirmación el cual el usuario debe aceptar.
Flujos Alternativos	- Si surgió un error al tratar de registrar, actualizar o eliminar el sistema mostrara un mensaje de error.
Postcondiciones	- Se ejecuta la opción que el usuario realizo.

Tabla 9. Gestionar Asignatura

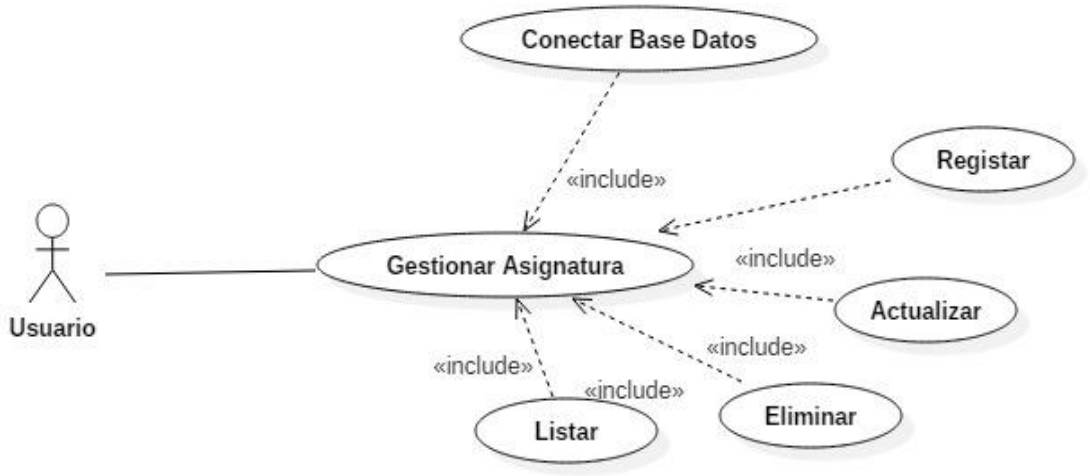

Figura 19. Diagrama Caso Uso Gestionar Asignatura

DETALLE DE CASO DE USO	
Numero	**RF7**
Nombre	**Gestionar Área**

Descripción	El sistema carga la vista de Gestión Área y permite al usuario la lista de registros, registrar, actualizar y eliminar la información correspondiente al área.
Actores	- Administrador
Precondiciones	- El usuario debe tener una sesión iniciada.
Flujo Normal	- El sistema carga la vista de Gestión Área - El usuario selecciona la opción que desea realizar - Si ¿es para registrar? el usuario debe digitar la información del formulario que es obligatoria y luego hacer clic en el botón de registrar. - Si ¿es para actualizar? el usuario debe seleccionar el área a modificar y el sistema cargará el formulario con la información del área seleccionada y el usuario deberá modificar la información y luego clic en el botón de actualizar. - Si ¿es para eliminar? el usuario debe hacer clic en el botón de eliminar y luego el sistema abre un modal de confirmación el cual el usuario debe aceptar.
Flujos Alternativos	- Si surgió un error al tratar de registrar, actualizar o eliminar el sistema mostrara un mensaje de error.
Postcondiciones	- Se ejecuta la opción que el usuario realizo.

Tabla 10. Gestionar Área

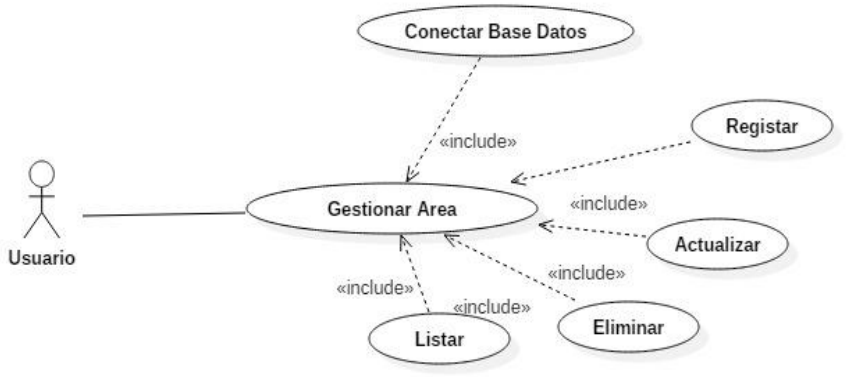

Figura 20. Diagrama Caso de Uso Gestionar Área

DETALLE DE CASO DE USO	
Numero	**RF8**
Nombre	**Matricular Alumno**
Descripción	El sistema carga la vista de Matriculas con la lista de alumnos los cuales puede filtrar por nombres y apellidos o número de documento, permite al usuario matricular el estudiante que ha seleccionado curso y año lectivo.
Actores	- Administrador, Secretaria
Precondiciones	- El usuario debe tener una sesión iniciada.
Flujo Normal	- El sistema carga la vista de Matriculas con la lista de alumnos los cuales puede filtrar por nombres y apellidos o número de documento - Si el estudiante no existe el usuario debe proceder a registrar el estudiante, si el estudiante existe lo selecciona de la lista. - El sistema carga un modal con la información de matrícula de estudiante. - El usuario digita la información del formulario de matrícula. - El sistema procede a matricular al estudiante con la información recibida.
Flujos Alternativos	- Si surgió un error al tratar de matricular al estudiante el sistema mostrara un mensaje de error.
Postcondiciones	- El estudiante queda matriculado.

Tabla 11. Matricular Alumno

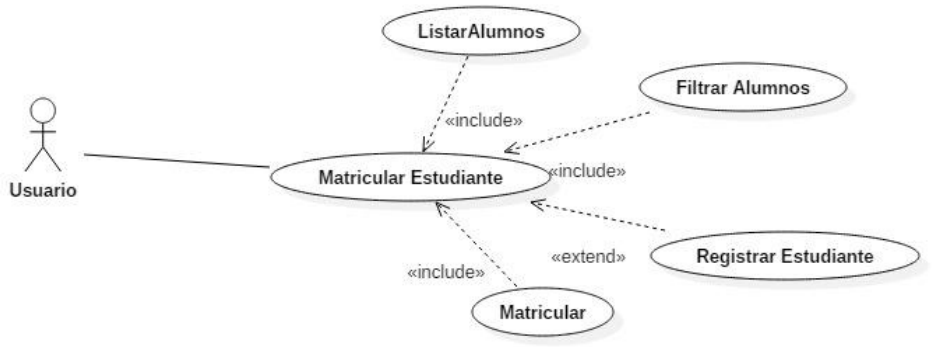

Figura 21. Diagrama Caso de Uso Matricular Estudiante

DETALLE DE CASO DE USO	
Numero	**RF9**
Nombre	**Gestionar Usuario**
Descripción	El sistema carga la vista de Gestión Usuario y permite al usuario la lista de registros, registrar, actualizar y eliminar la información correspondiente al usuario.
Actores	- Administrador
Precondiciones	- El usuario debe tener una sesión iniciada.
Flujo Normal	- El sistema carga la vista de Gestión Usuario - El usuario selecciona la opción que desea realizar - Si ¿es para registrar? el usuario debe digitar la información del formulario que es obligatoria y luego hacer clic en el botón de registrar. - Si ¿es para actualizar? el usuario debe seleccionar el usuario a modificar y el sistema cargará el formulario con la información del usuario seleccionado y el usuario deberá modificar la información y luego clic en el botón de actualizar. - Si ¿es para eliminar? el usuario debe hacer clic en el botón de eliminar y luego el sistema abre un modal de confirmación el cual el usuario debe aceptar.
Flujos Alternativos	- Si surgió un error al tratar de registrar, actualizar o eliminar el sistema mostrara un mensaje de error.

Postcondiciones	- Se ejecuta la opción que el usuario realizo.

Tabla 12. Gestionar Usuario

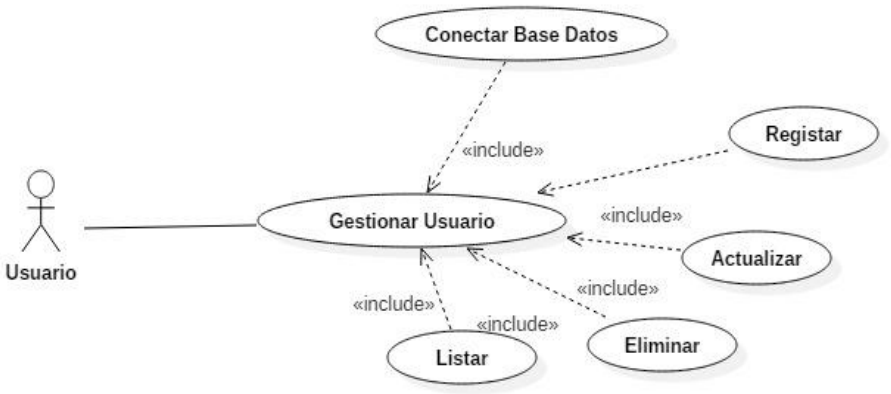

Figura 22. Diagrama de Caso de Uso Gestionar Usuario

DETALLE DE CASO DE USO	
Numero	**RF10**
Nombre	**Gestionar Grados**
Descripción	El sistema carga la vista de Gestión Grado y permite al usuario la lista de registros, usuario registrar, actualizar y eliminar la información correspondiente al grado.
Actores	- Administrador
Precondiciones	- El usuario debe tener una sesión iniciada.

Flujo Normal	- El sistema carga la vista de Gestión Grado - El usuario selecciona la opción que desea realizar - Si ¿es para registrar? el usuario debe digitar la información del formulario que es obligatoria y luego hacer clic en el botón de registrar. - Si ¿es para actualizar? el usuario debe seleccionar el grado a modificar y el sistema cargará el formulario con la información del grado seleccionado y el usuario deberá modificar la información y luego clic en el botón de actualizar. - Si ¿es para eliminar? el usuario debe hacer clic en el botón de eliminar y luego el sistema abre un modal de confirmación el cual el usuario debe aceptar.
Flujos Alternativos	- Si surgió un error al tratar de registrar, actualizar o eliminar el sistema mostrara un mensaje de error.
Postcondiciones	- Se ejecuta la opción que el usuario realizo.

Tabla 13. Gestionar Grados

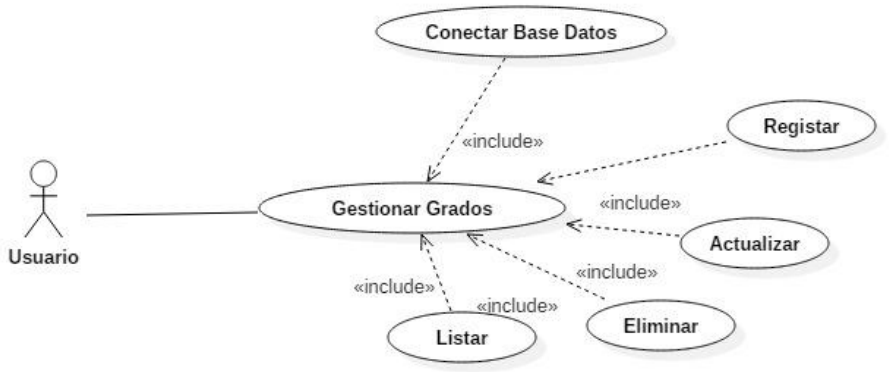

Figura 23. Diagrama de Caso de Uso Gestionar Grado

DETALLE DE CASO DE USO	
Numero	**RF11**
Nombre	**Gestionar Horarios**

Descripción	El sistema carga la vista de Gestión Horario y permite al usuario la lista de registros, registrar, actualizar y eliminar la información correspondiente al horario.
Actores	- Administrador
Precondiciones	- El usuario debe tener una sesión iniciada.
Flujo Normal	- El sistema carga la vista de Gestión Horario - El usuario selecciona la opción que desea realizar - Si ¿es para registrar? el usuario debe digitar la información del formulario que es obligatoria y luego hacer clic en el botón de registrar. - Si ¿es para actualizar? el usuario debe seleccionar el usuario a modificar y el sistema cargará el formulario con la información del horario seleccionado y el usuario deberá modificar la información y luego clic en el botón de actualizar. - Si ¿es para eliminar? el usuario debe hacer clic en el botón de eliminar y luego el sistema abre un modal de confirmación el cual el usuario debe aceptar.
Flujos Alternativos	- Si surgió un error al tratar de registrar, actualizar o eliminar el sistema mostrara un mensaje de error.
Postcondiciones	- Se ejecuta la opción que el usuario realizo.

Tabla 14. Gestionar Horarios

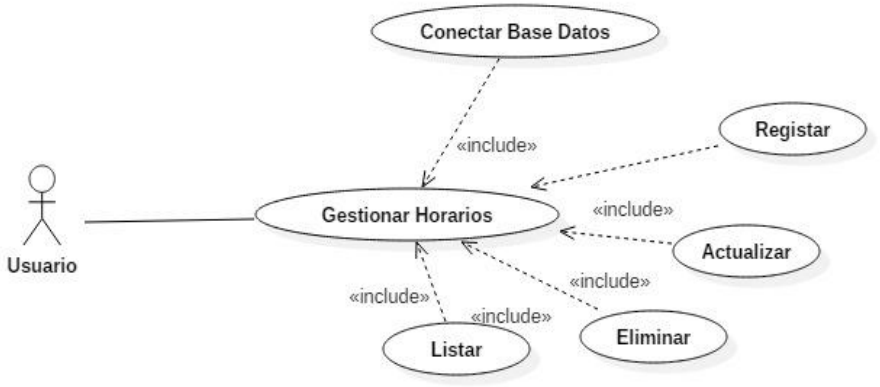

Figura 24. Diagrama de Caso de Uso Gestionar Horario

DETALLE DE CASO DE USO	
Numero	**RF12**
Nombre	**Gestionar Periodos Académicos**
Descripción	El sistema carga la vista de Gestión Periodo Académico y permite al usuario la lista de registros, registrar, actualizar y eliminar la información correspondiente al periodo académico.
Actores	- Administrador
Precondiciones	- El usuario debe tener una sesión iniciada.
Flujo Normal	- El sistema carga la vista de Gestión Usuario - El usuario selecciona la opción que desea realizar - Si ¿es para registrar? el usuario debe digitar la información del formulario que es obligatoria y luego hacer clic en el botón de registrar. - Si ¿es para actualizar? el usuario debe seleccionar el periodo académico a modificar y el sistema cargará el formulario con la información del periodo académico seleccionado y el usuario deberá modificar la información y luego clic en el botón de actualizar. - Si ¿es para eliminar? el usuario debe hacer clic en el botón de eliminar y luego el sistema abre un modal de confirmación el cual el usuario debe aceptar.
Flujos Alternativos	- Si surgió un error al tratar de registrar, actualizar o eliminar el sistema mostrara un mensaje de error.
Postcondiciones	- Se ejecuta la opción que el usuario realizo.

Tabla 15. Gestionar Periodos Académicos

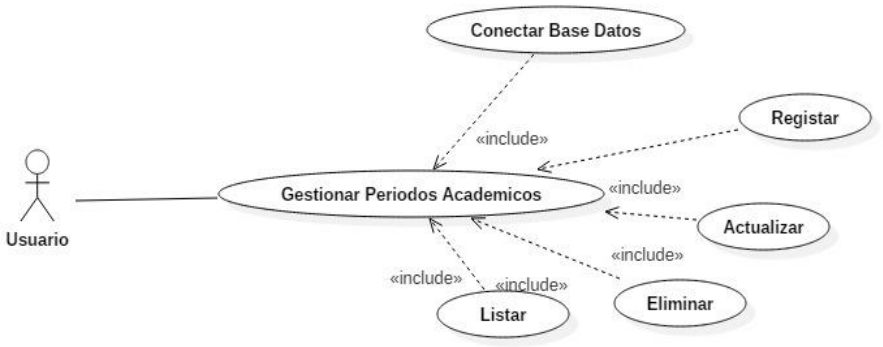

Figura 25. Diagrama de Caso de Uso Gestionar Periodo Académico

DETALLE DE CASO DE USO	
Numero	**RF13**
Nombre	**Gestionar Carga Académica**
Descripción	El sistema carga la vista de Gestión Carga Académica y permite al usuario el registro de notas de los estudiantes por materia.
Actores	- Administrador, Docente
Precondiciones	- El usuario debe tener una sesión iniciada.
Flujo Normal	- El sistema carga la vista de Gestión Carga Académica. - El usuario busca la materia a la que desea registrar las notas de los estudiantes. - El sistema valida el id del estudiante y registra las notas del estudiante.
Flujos Alternativos	- Si surgió un error al tratar de registrar las notas el sistema mostrara un mensaje de error.
Postcondiciones	- Se ejecuta el registro de notas que el usuario realizo.

Tabla 16. Gestionar Carga Académica

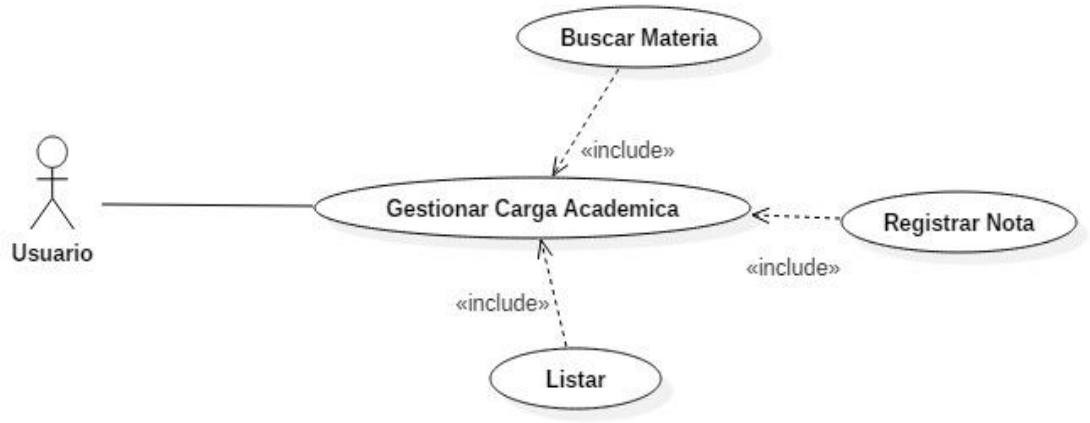

Figura 26. Diagrama de Caso de Uso Gestionar Carga Academica

DETALLE DE CASO DE USO	
Numero	**RF14**
Nombre	**Gestionar Logros**
Descripción	El sistema carga la vista de Gestión Logro y permite al usuario registrar, actualizar y eliminar la información correspondiente al logro.
Actores	- Administrador, Coordinador, Docente
Precondiciones	- El usuario debe tener una sesión iniciada.
Flujo Normal	- El sistema carga la vista de Gestión Logro - El usuario selecciona la opción que desea realizar - Si ¿es para registrar? el usuario debe digitar la información del formulario que es obligatoria y luego hacer clic en el botón de registrar. - Si ¿es para actualizar? el usuario debe seleccionar el logro a modificar y el sistema cargará el formulario con la información del logro seleccionado y el usuario deberá modificar la información y luego clic en el botón de actualizar. - Si ¿es para eliminar? el usuario debe hacer clic en el botón de eliminar y luego el sistema abre un modal de confirmación el cual el usuario debe aceptar.

Flujos Alternativos	- Si surgió un error al tratar de registrar, actualizar o eliminar el sistema mostrara un mensaje de error.
Postcondiciones	- Se ejecuta la opción que el usuario realizo.

Tabla 17. Gestionar Logros

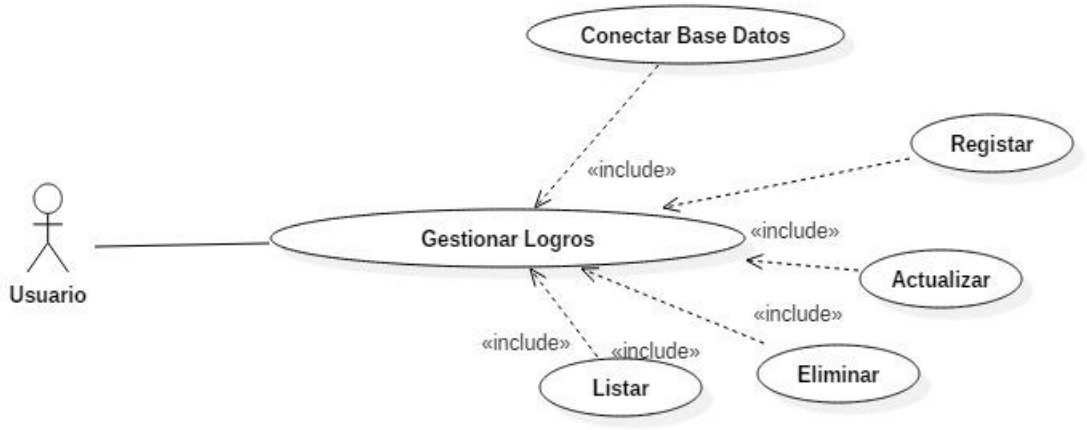

Figura 27. Diagrama de caso de Uso Gestionar Logro

DETALLE DE CASO DE USO	
Numero	**RF15**
Nombre	**Notas Académicas de estudiante por materia**
Descripción	El sistema carga la vista de notas académicas de estudiante por materia.
Actores	- Estudiante, Acudiente
Precondiciones	- El usuario debe tener una sesión iniciada.
Flujo Normal	- El sistema carga la vista de notas académicas por materia. - El usuario visualiza la información de sus notas.
Flujos Alternativos	- Si surgió un error al tratar de visualizar la vista se mostrará un mensaje de error.
Postcondiciones	- Se visualiza las notas del estudiante por materia.

Tabla 18. Notas Académicas de estudiante por materia

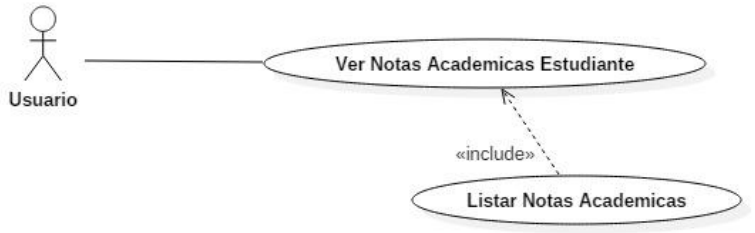

Figura 28. Diagrama de Caso de Uso Notas Académicas Estudiante

DETALLE DE CASO DE USO	
Numero	**RF16**
Nombre	**Carga Académica Estudiante**
Descripción	El sistema carga la vista de carga académica que tiene asignada el usuario para el año lectivo.
Actores	- Estudiante, Acudiente
Precondiciones	- El usuario debe tener una sesión iniciada.
Flujo Normal	- El sistema carga la vista de carga académica de estudiante. - El usuario visualiza la información de su carga académica para el año lectivo.
Flujos Alternativos	- Si surgió un error al tratar de visualizar la vista se mostrará un mensaje de error.
Postcondiciones	- Se visualiza la carga académica del estudiante en el año lectivo.

Tabla 19. Carga Académica Estudiante

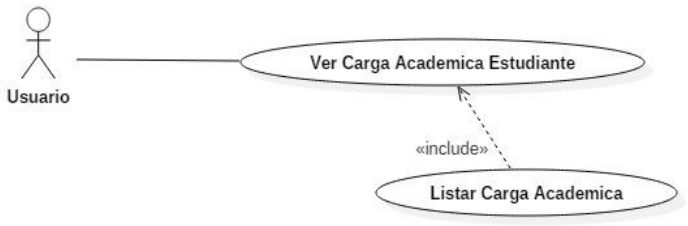

Figura 28. Diagrama de caso de Uso Ver Carga Académica Estudiante

DETALLE DE CASO DE USO	
Numero	**RF17**
Nombre	**Horario Estudiante**
Descripción	El sistema carga la vista del horario semanal de estudiante para el año lectivo.
Actores	- Estudiante, Acudiente
Precondiciones	- El usuario debe tener una sesión iniciada.
Flujo Normal	- El sistema carga la vista del horario semanal de estudiante para el año lectivo. - El usuario visualiza la información de su horario académico semanal.
Flujos Alternativos	- Si surgió un error al tratar de visualizar la vista se mostrará un mensaje de error.
Postcondiciones	- Se visualiza la información de su horario académico semanal.

Tabla 20. Horario Estudiante

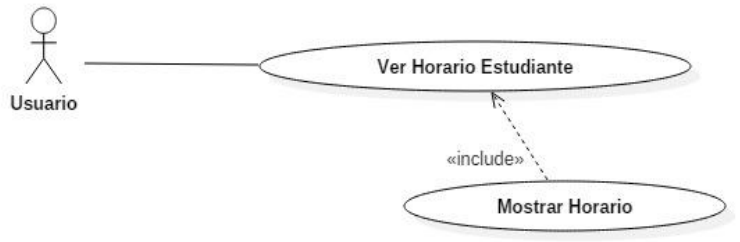

Figura 30. Diagrama de caso de Uso Horario Estudiante

DETALLE DE CASO DE USO	
Numero	**RF18**
Nombre	**Carga Académica Docente**
Descripción	El sistema carga la vista de carga académica que tiene asignada el usuario para el año lectivo.

Actores	- Docente
Precondiciones	- El usuario debe tener una sesión iniciada.
Flujo Normal	- El sistema carga la vista de carga académica de docente. - El usuario visualiza la información de su carga académica para el año lectivo.
Flujos Alternativos	- Si surgió un error al tratar de visualizar la vista se mostrará un mensaje de error.
Postcondiciones	- Se visualiza la carga académica del docente en el año lectivo.

Tabla 21. Carga Académica Docente

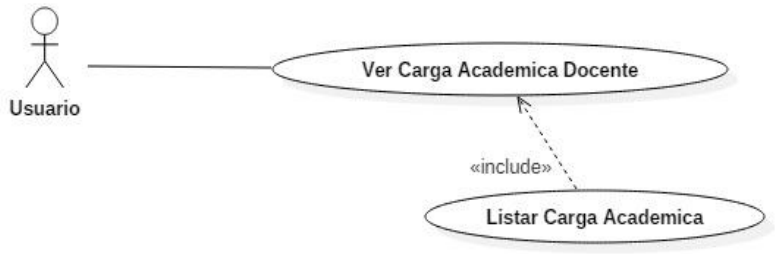

Figura 29. Diagrama de Caso de Uso Carga Académica Docente

DETALLE DE CASO DE USO	
Numero	**RF19**
Nombre	**Horario Docente**
Descripción	El sistema carga la vista del horario semanal de docente para el año lectivo.
Actores	- Docente
Precondiciones	- El usuario debe tener una sesión iniciada.
Flujo Normal	- El sistema carga la vista del horario semanal de docente para el año lectivo. - El usuario visualiza la información de su horario académico semanal.

Flujos Alternativos	- Si surgió un error al tratar de visualizar la vista se mostrará un mensaje de error.
Postcondiciones	- Se visualiza la información de su horario académico semanal.

Tabla 22. Horario Docente

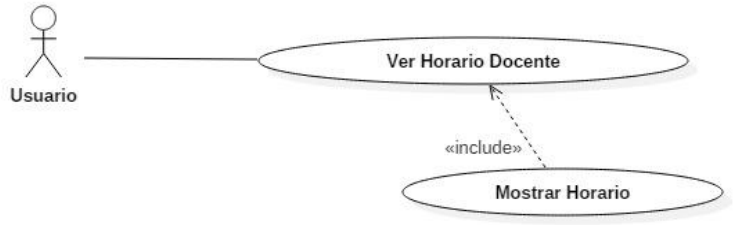

Figura 30. Diagrama de Caso de Uso Horario Docente

DETALLE DE CASO DE USO	
Numero	**RF20**
Nombre	**Gestionar Rol**
Descripción	El sistema carga la vista de Gestión Rol y permite al usuario registrar, actualizar y eliminar la información correspondiente al rol.
Actores	- Administrador
Precondiciones	- El usuario debe tener una sesión iniciada.

Flujo Normal	- El sistema carga la vista de Gestión Rol - El usuario selecciona la opción que desea realizar - Si ¿es para registrar? el usuario debe digitar la información del formulario que es obligatoria y luego hacer clic en el botón de registrar. - Si ¿es para actualizar? el usuario debe seleccionar el rol a modificar y el sistema cargará el formulario con la información del rol seleccionado y el usuario deberá modificar la información y luego clic en el botón de actualizar. - Si ¿es para eliminar? el usuario debe hacer clic en el botón de eliminar y luego el sistema abre un modal de confirmación el cual el usuario debe aceptar.
Flujos Alternativos	- Si surgió un error al tratar de registrar, actualizar o eliminar el sistema mostrara un mensaje de error.
Postcondiciones	- Se ejecuta la opción que el usuario realizo.

Tabla 23. Gestionar Rol

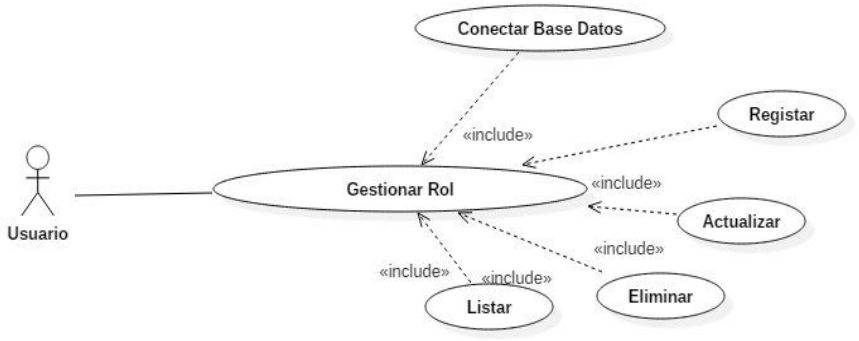

Figura 31. Diagrama de Caso de Uso Gestionar Rol

DETALLE DE CASO DE USO	
Numero	**RF21**
Nombre	**Cerrar Sesión**
Descripción	El sistema carga el enlace para cerrar sesión en el menú principal y el usuario hace clic en el enlace para cerrar la sesión.

Actores	- Administrador, Estudiante, Coordinador, Docente, Secretaria y Acudiente
Precondiciones	- El usuario debe hacer clic en el enlace
Flujo Normal	- El sistema carga en el menú principal el enlace para cerrar sesión. - El usuario hace clic en el enlace. - El sistema cierra la sesión.
Flujos Alternativos	- Si no se logró cerrar la sesión se muestra un mensaje de error.
Postcondiciones	- Se carga el formulario de Iniciar Sesión.

Tabla 24. Cerrar Sesión

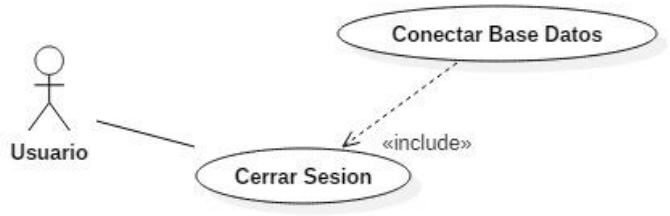

Figura 32. Diagrama de caso de Uso Cerrar Sesión

3.5.7. Vista Lógica

En la vista lógica se definió la arquitectura que se utilizara en el Fronted la cual es Modelo Vista Controlador MVC y en el Backend Modelo Controlador (MC) del sistema las cuales tiene muchos beneficios entre ellos separar la lógica del negocio por medio de controladores y la representación de la información que maneja la aplicación. Optimizando la capacidad de la organización, la vista lógica apoya principalmente los requisitos funcionales lo que el sistema debe brindar en términos de servicios a sus usuarios. Se uso el enfoque de Booch/Rational para representar la arquitectura lógica, mediante diagramas de clases.

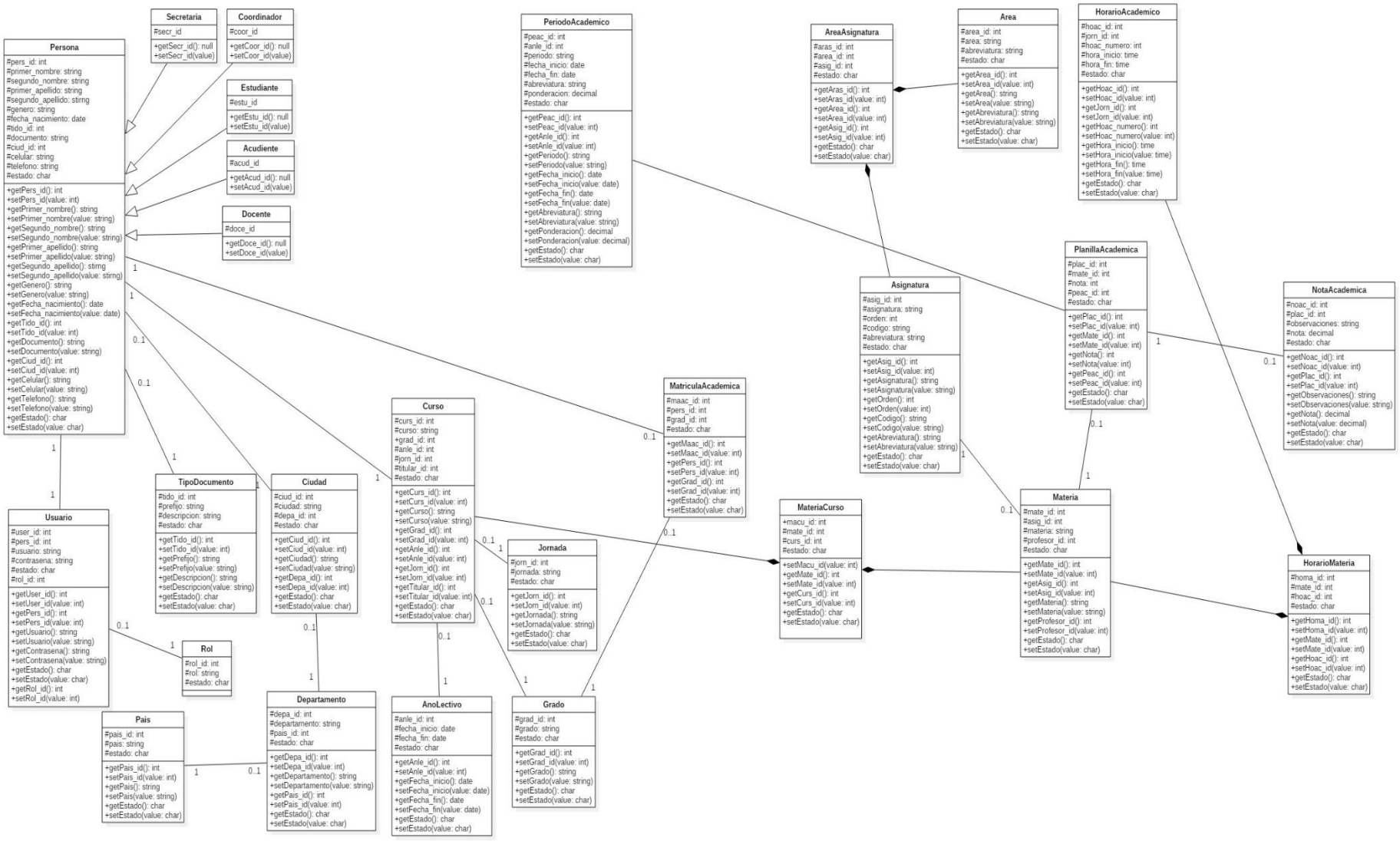

Figura 33. Diagrama de Clase

3.5.8. Vista de Desarrollo

Se muestra el sistema desde la perspectiva de un programador y se ocupa de la gestión del software; o, en otras palabras, se va a mostrar cómo está dividido el sistema software en componentes y las dependencias que hay entre esos componentes.

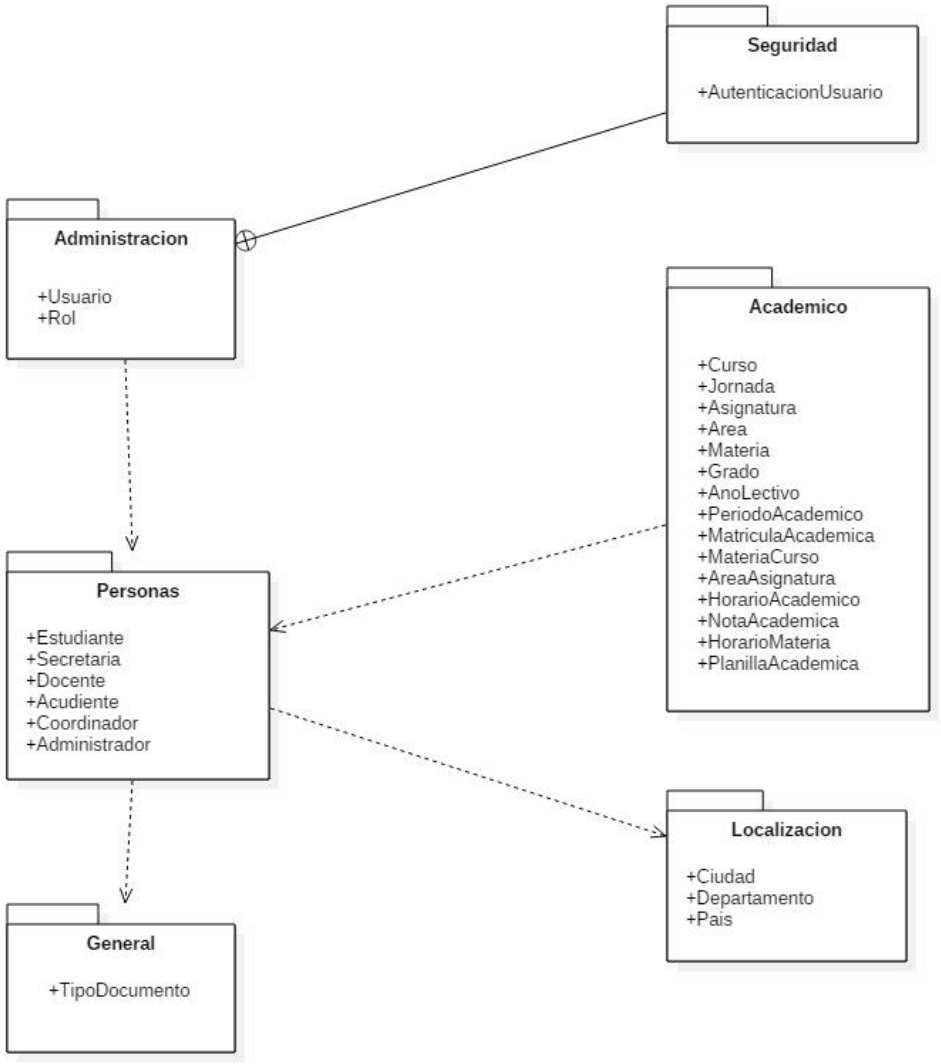

Figura 34. Diagrama de Paquetes

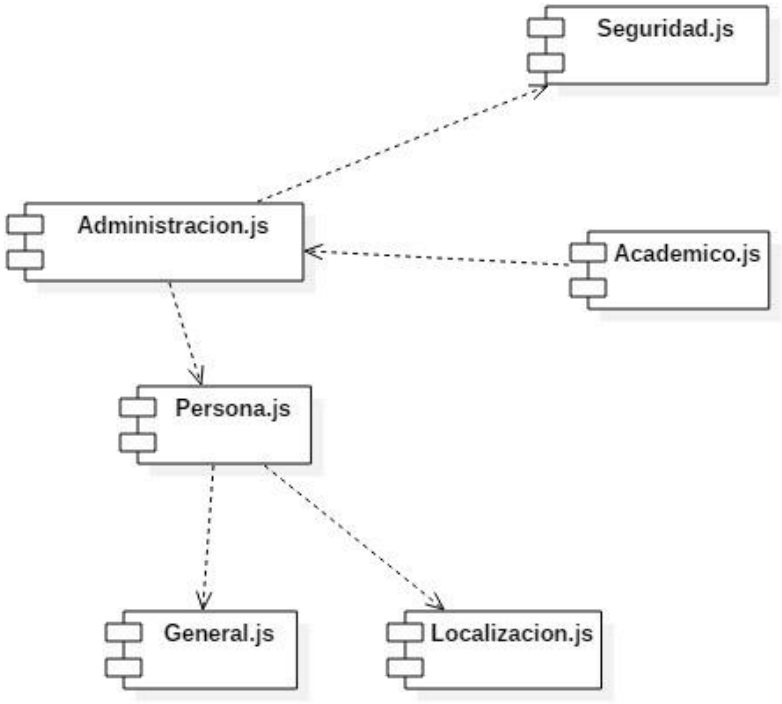

Figura 35. Diagrama de Componentes

3.5.9. Vista Física

En la vista física se muestra desde la perspectiva de un ingeniero de sistemas todos los componentes físicos del sistema, así como las conexiones físicas entre esos componentes que conforman la solución (incluyendo los servicios).

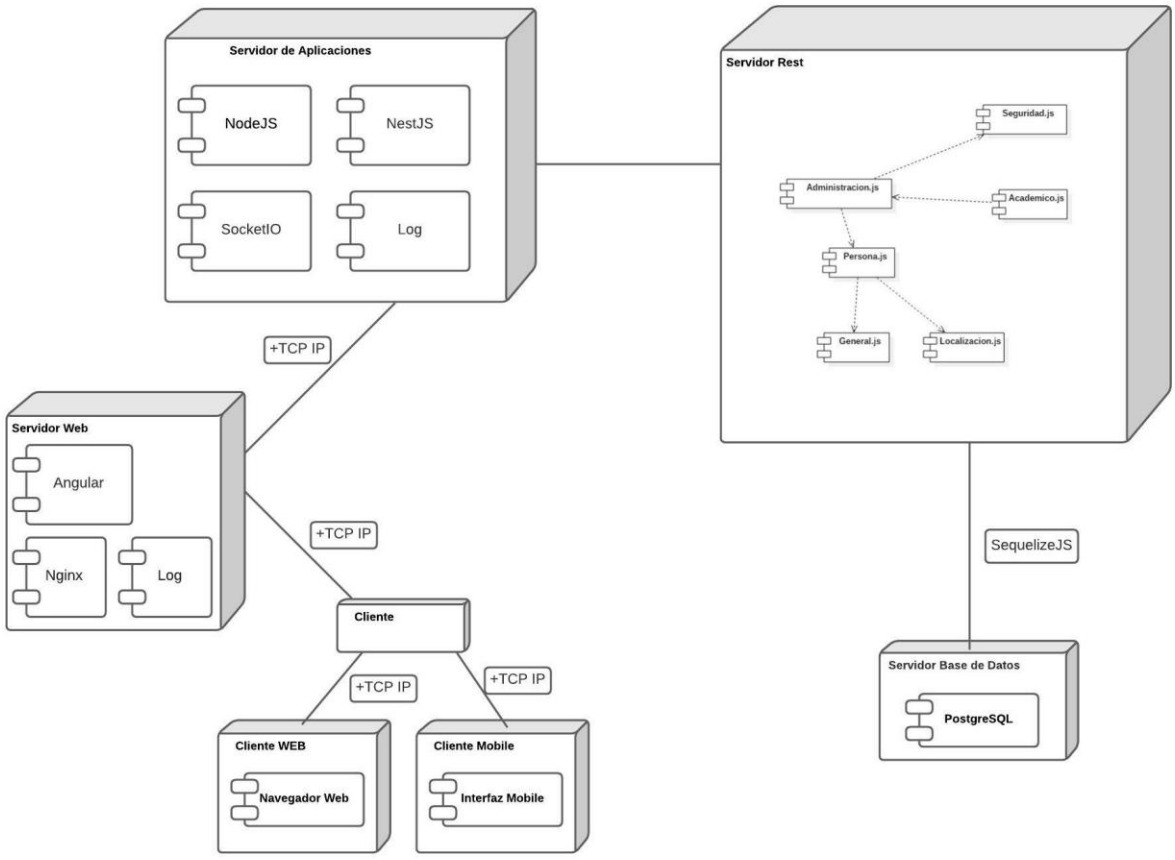

Figura 36. Diagrama de Despliegue

A continuación, se explica la vista de despliegue planteada para mayor comprensión:

- Como se muestra en la figura 13 se utilizó una arquitectura de 5 capas para la vista de despliegue, la capa de cliente cuenta con dos interfaces de usuario que permiten el acceso a la aplicación desde el navegador web en computadores de escritorio y dispositivos móviles esta capa se comunica con la capa del servidor web que cuenta con el Fronted desarrollado en Angular y ejecutando un servidor Ngnix, esta capa a su vez se comunica con la capa del servidor de aplicaciones el cual se ejecuta bajo NodeJS, dicho Backend está desarrollado con el Framework NestJS y utiliza Socket.IO, esta capa a su vez se comunica con el servidor Rest y esta con la capa de

persistencia que cuenta con el servidor de base de datos la cual es PostgreSQL. Los clientes se comunicarán a través de http con el servidor web.

3.5.10. Vista de Procesos

En esta vista se muestran los procesos que hay en el sistema y la forma en la que se comunican estos procesos; es decir, se representa desde la perspectiva de un integrador de sistemas, el flujo de trabajo paso a paso de negocio y operacionales de los componentes que conforman el sistema.

3.5.10.1. Iniciar Sesión

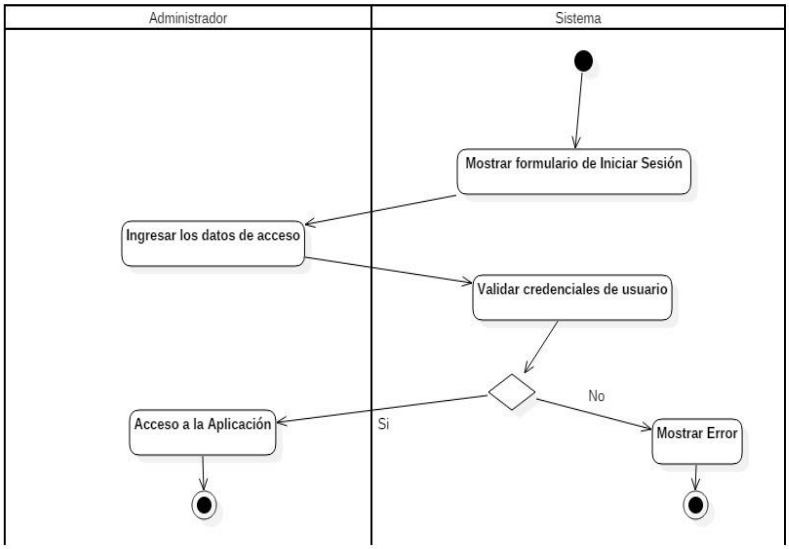

Figura 37. Diagrama de Actividad Iniciar Sesión

3.5.10.2. Gestión Año Lectivo

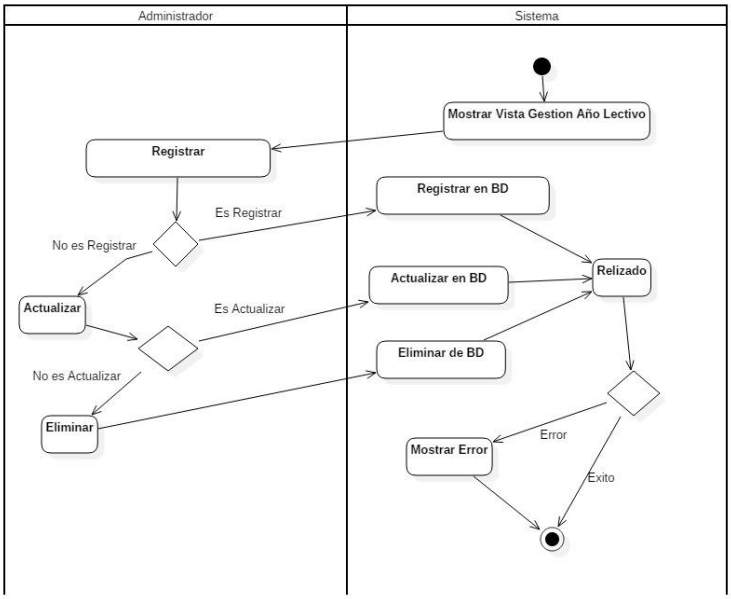

Figura 38. Diagrama de Actividad Gestión Año Lectivo

3.5.10.3. Gestión Estudiante

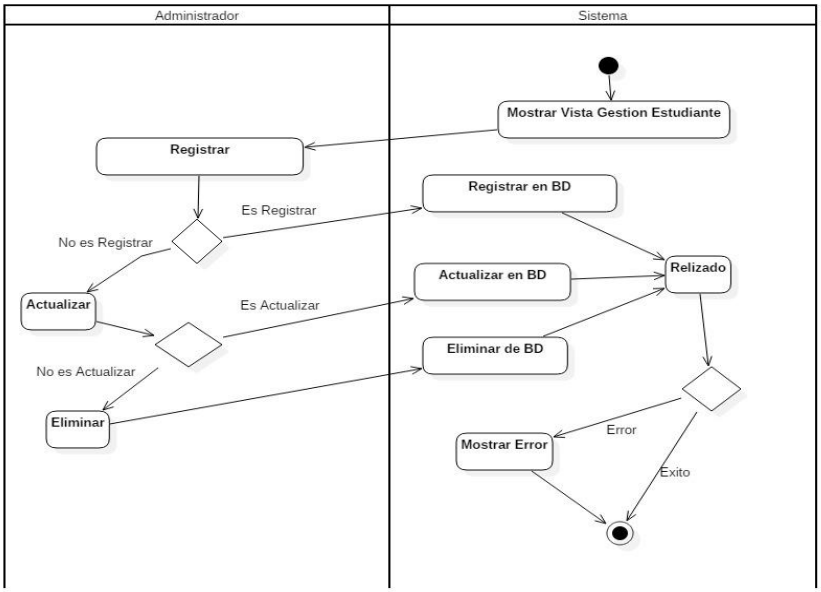

Figura 39. Diagrama de Actividades Gestión Estudiante

3.5.10.4. Gestión Docente

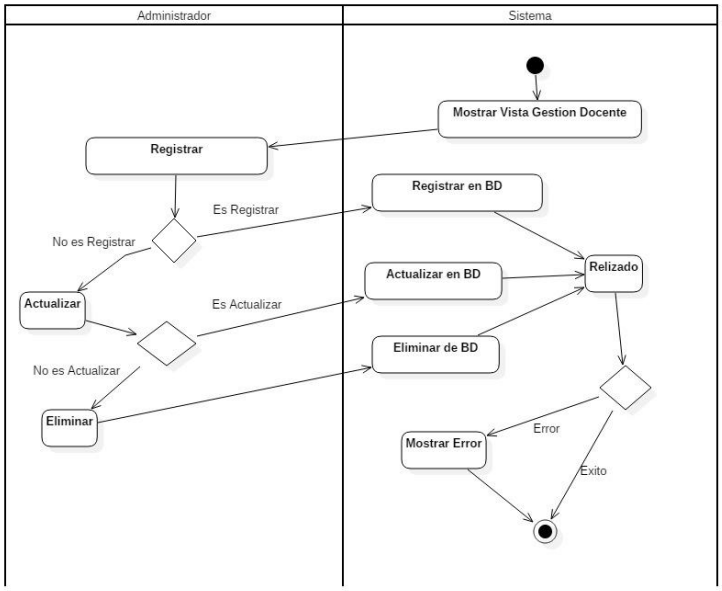

Figura 40. Diagrama de Actividades Gestión Docente

3.5.10.5. Gestión Curso

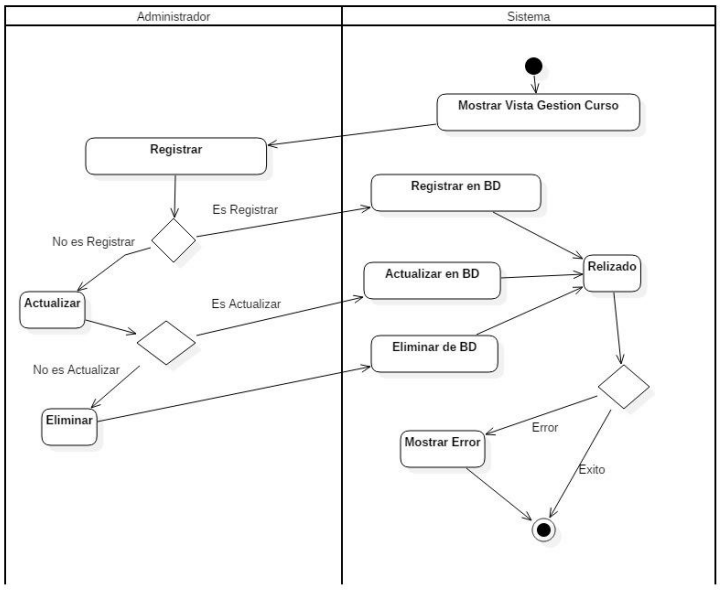

Figura 41. Diagrama de Actividades Gestión Curso

3.5.10.6. Gestión Asignatura

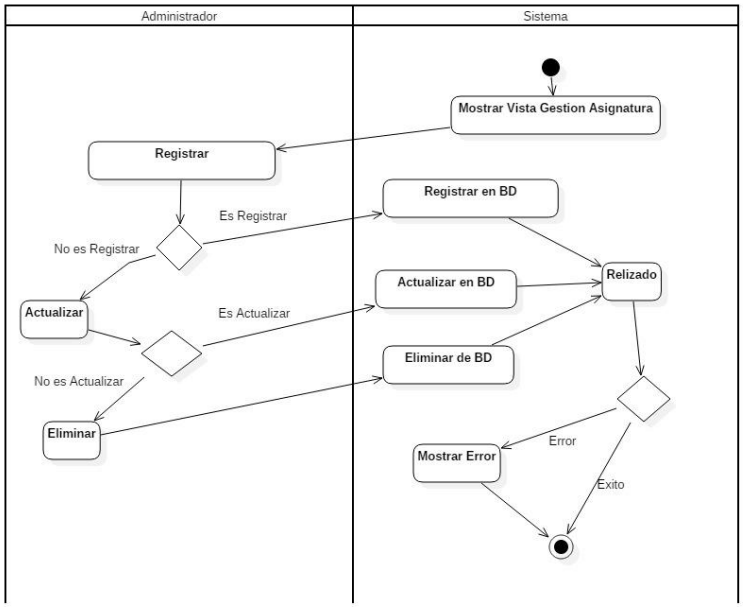

Figura 42. Diagrama de Actividades Gestión Asignatura

3.5.10.7. Gestión Área

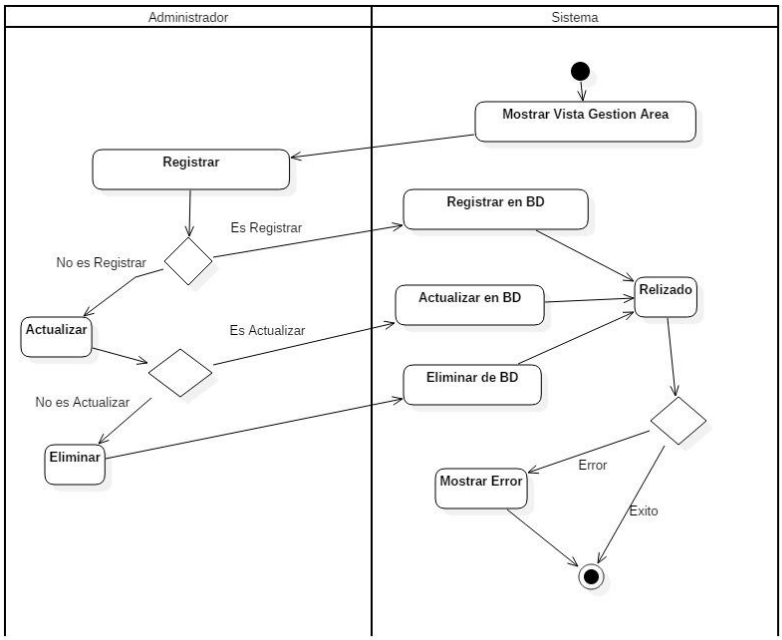

Figura 43. Diagrama de Actividades Gestión Área

3.5.10.8. Matricular Alumno

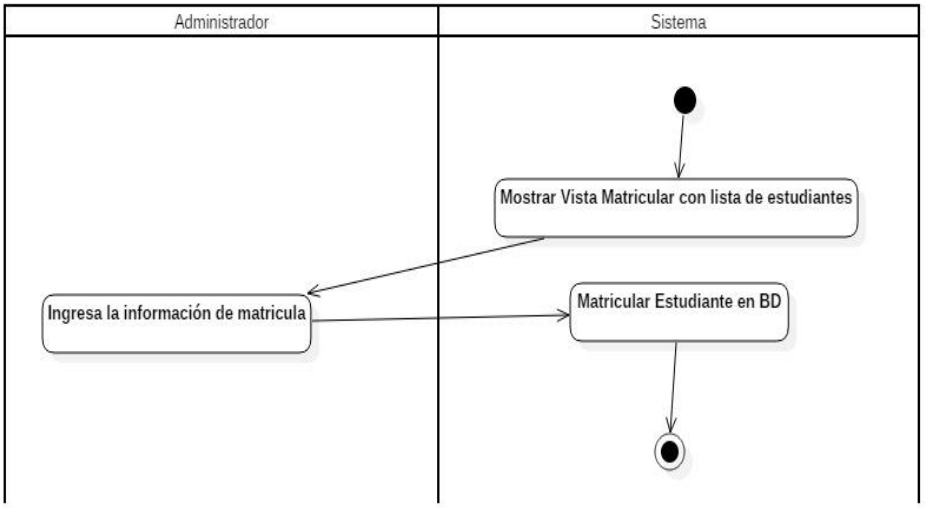

Figura 44. Diagrama de Actividades Matricular Estudiante

3.5.10.9. Gestión Usuarios

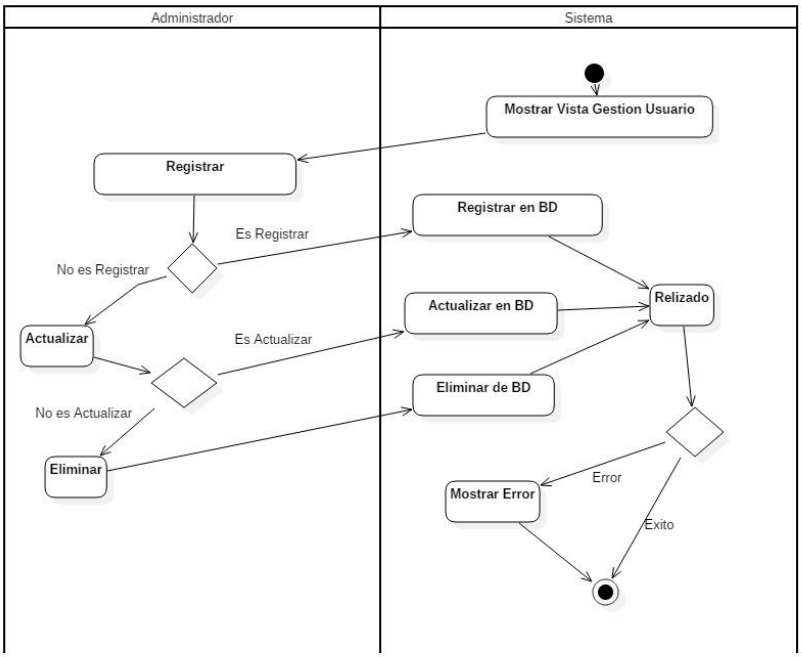

Figura 45. Diagrama de Actividades Gestión Usuario

3.5.10.10. Gestión Grados

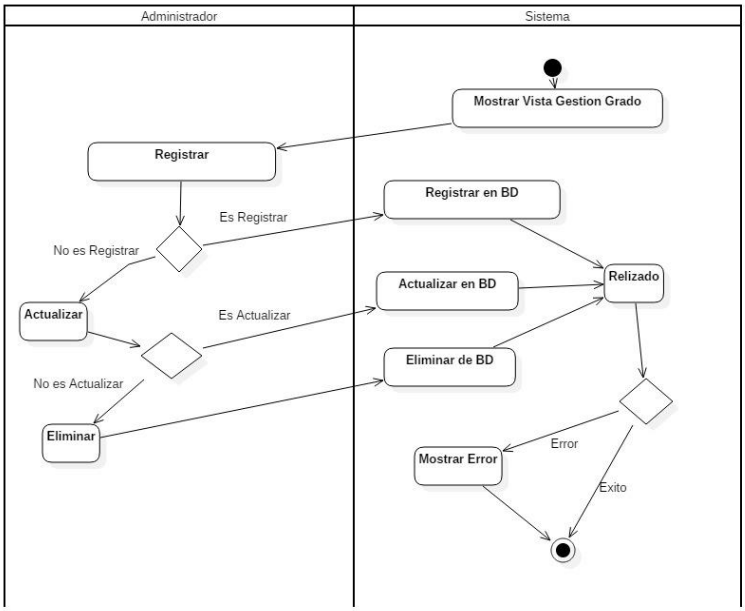

Figura 46. Diagrama de Actividades Gestión Grado

3.5.10.11. Gestión Horarios

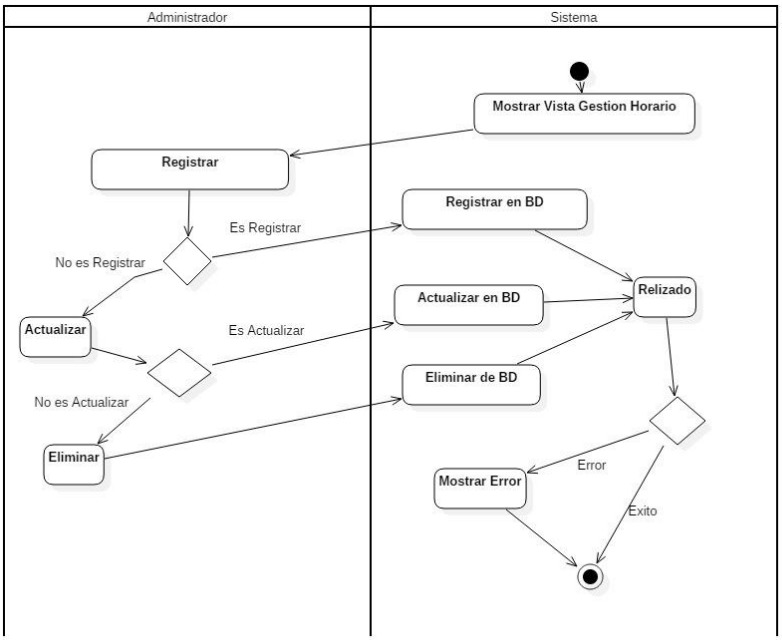

Figura 47. Diagrama de Actividades Gestión Horario

3.5.10.12. Gestión Periodos Académicos

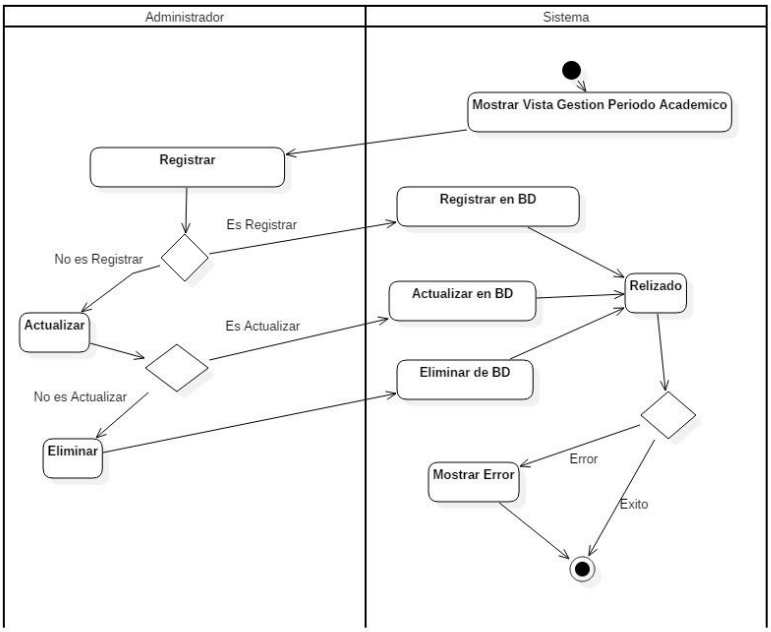

Figura 48. Diagrama de Actividades Gestión Periodo Académico

3.5.10.13. Gestión Carga Académica

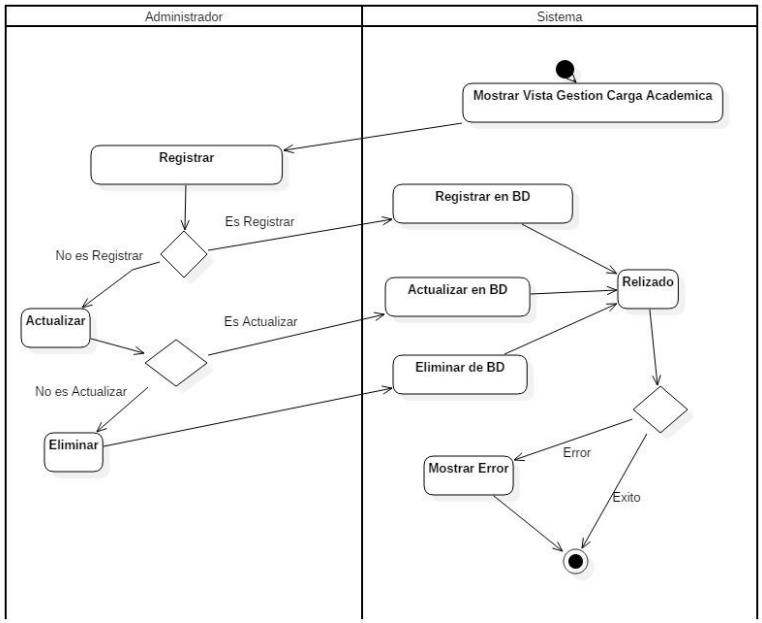

Figura 49. Diagrama de Actividades Gestión Carga Académica

3.5.10.14. Gestión de Logros Académicos

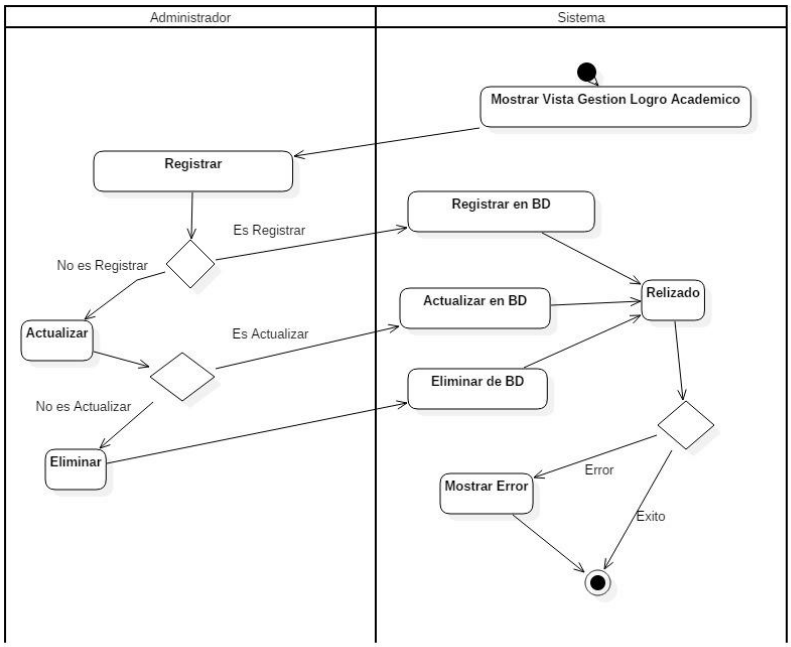

Figura 50. Diagrama de Actividades Gestión Logro Académico

3.5.10.15. Notas Académicas de estudiantes por materia

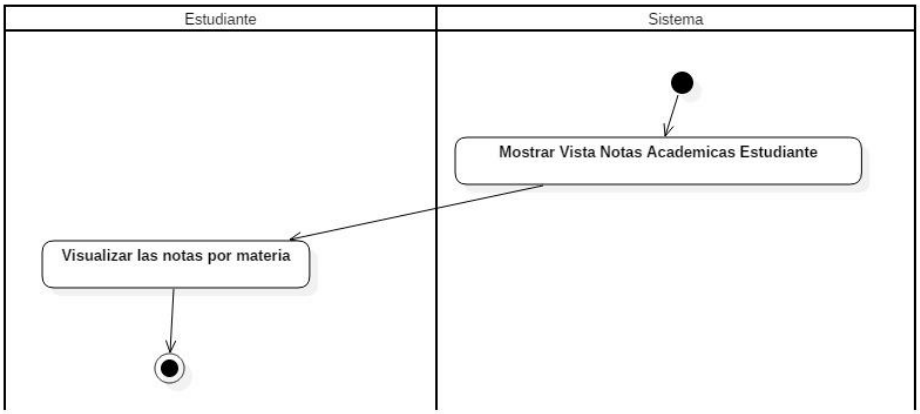

Figura 51. Diagrama de Actividades Notas Académicas Estudiante

3.5.10.16. Carga Académica Estudiante

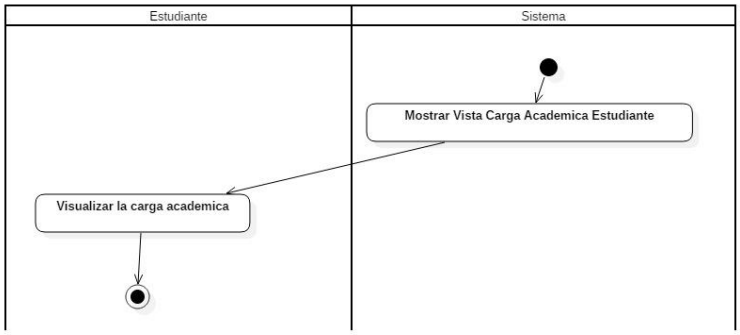

Figura 52. Diagrama de Actividades Carga Académica Estudiante

3.5.10.17. Horario Estudiante

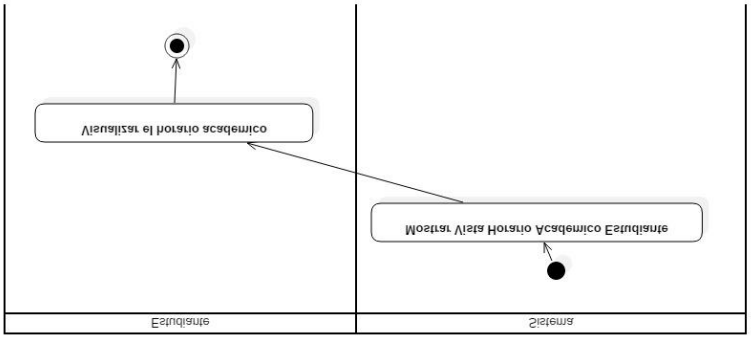

Figura 53. Diagrama de Actividades Horario Estudiante

3.5.10.18. Carga Académica Docente

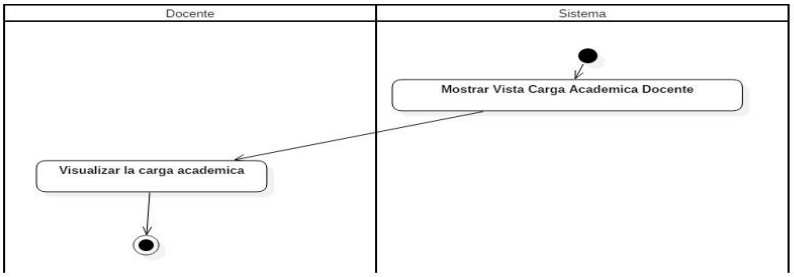

Figura 54. Diagrama de Actividades Carga Académica Docente

3.5.10.19. Horario Docente

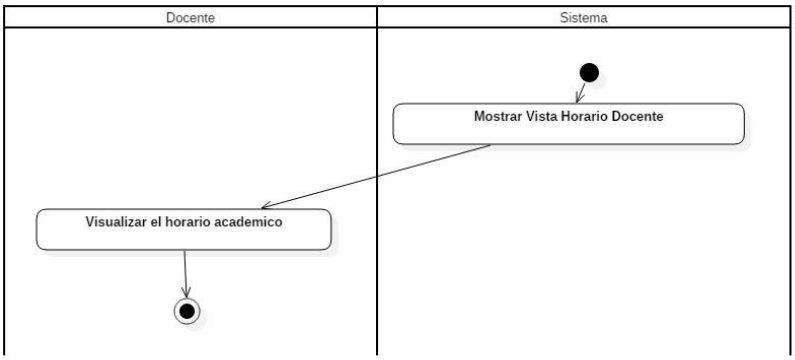

Figura 55. Diagrama de Actividades Horario Docente

3.5.10.20. Gestión Roles

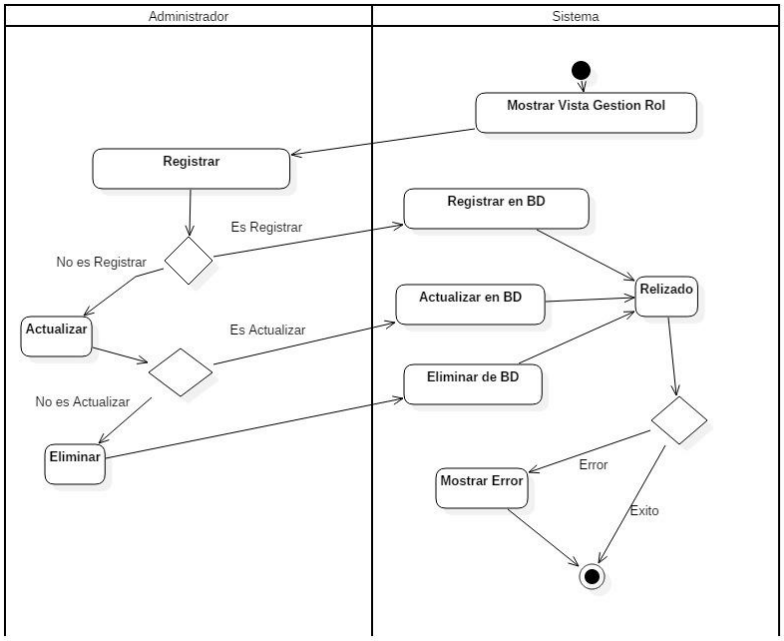

Figura 56. Diagrama de Actividades Gestión Rol

3.5.10.21. Cierre de Sesión

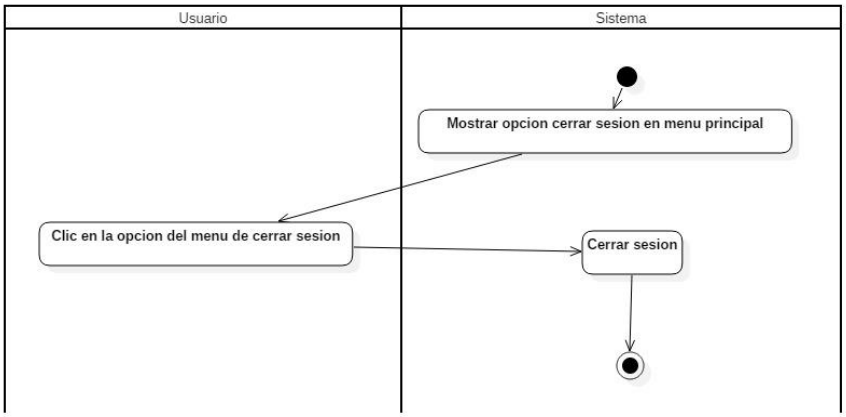

Figura 57. Diagrama de Actividades Cerrar Sesión

3.5.11. Diagrama Modelo Entidad Relación.

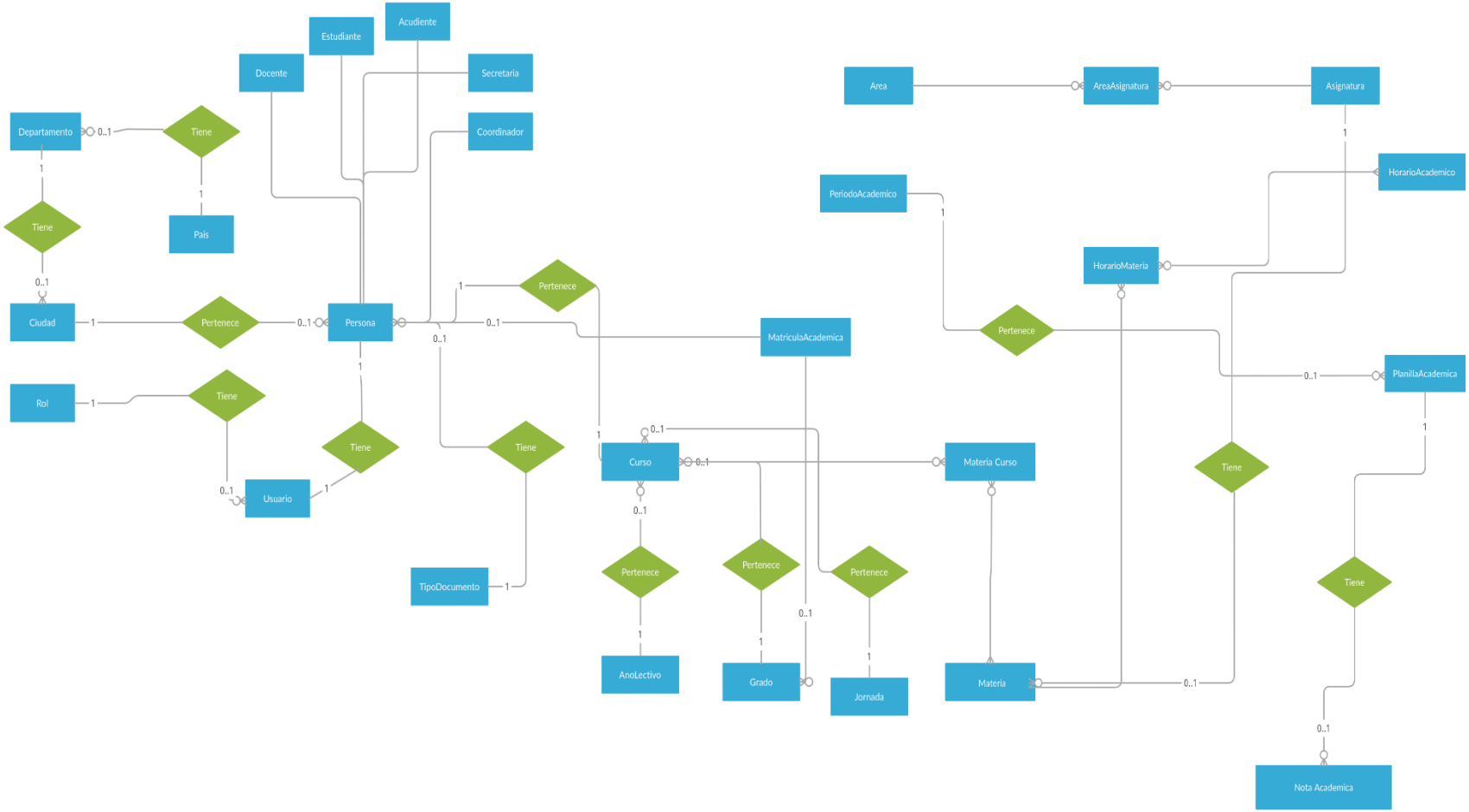

Figura 58. Diagrama Modelo Entidad Relación

3.5.12. Modelo Relacional.

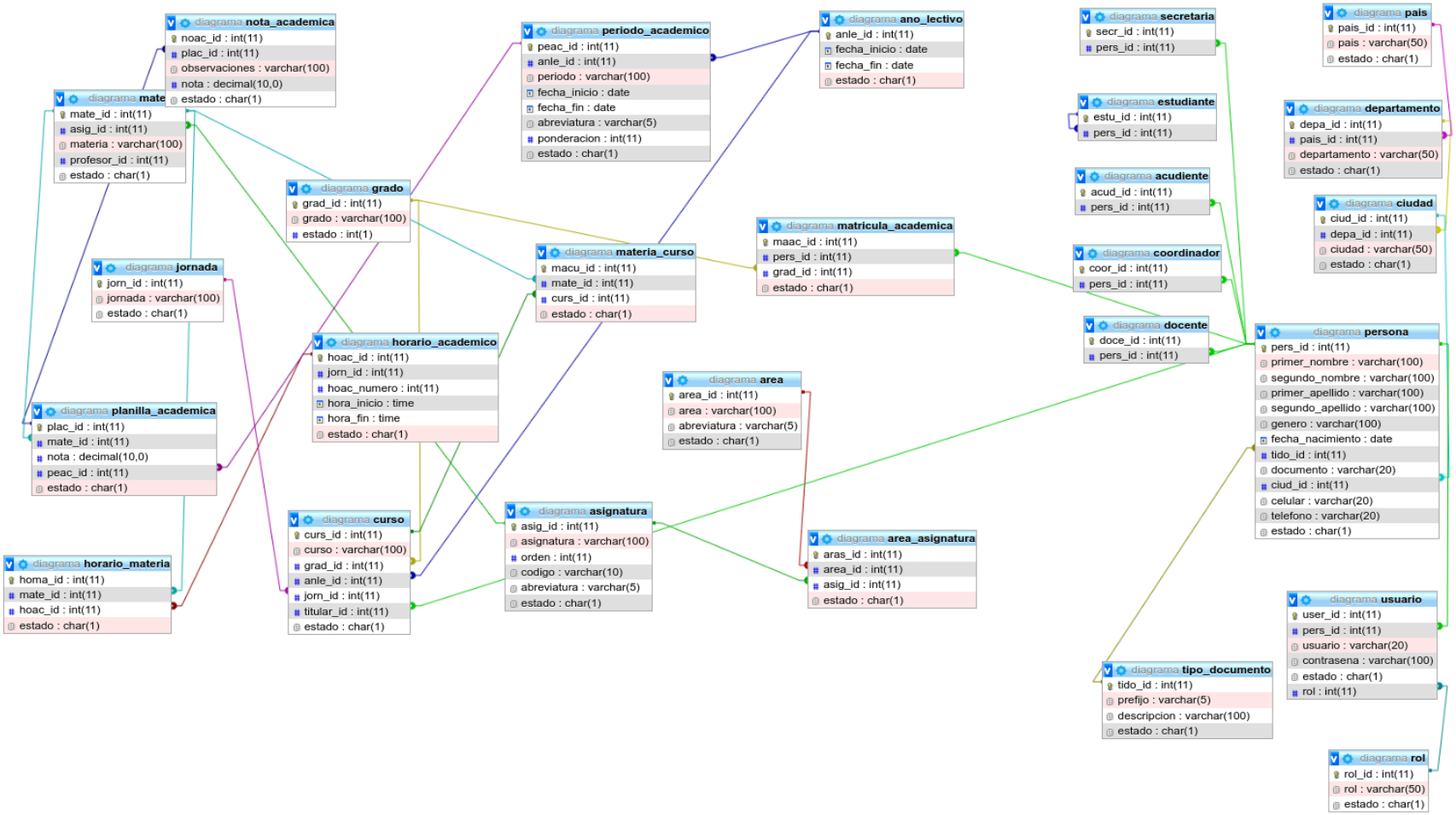

Figura 59. Modelo Relacional

3.6. VALIDAR ARQUITECTURA DE SOFTWARE CON ATAM

3.6.1. Investigar la metodología Architecture Tradeoff Analysis Method (ATAM)

3.6.2. Fase 1. Presentación

En la fase 1 de **ATAM** se realizan las actividades que corresponden al grupo de presentación el cual contiene 3 pasos los cuales son Presentar ATAM, Presentación de las metas del negocio y Presentación de la Arquitectura.

3.6.2.1. Presentar ATAM

Se realizo la presentación de la metodología de **ATAM** a los stakeholders para tratar de establecer expectativas y responder las preguntas propuestas, a continuación, se mostrará la tabla con los pasos que tiene la metodología ATAM y los cuales fueron presentados a los stakeholders.

Paso	ACTIVIDAD
1	Presentar ATAM
2	Presentación de las metas del negocio
3	Presentación de la Arquitectura
4	Identificación de los enfoques arquitectónicos
5	Generar el árbol de utilidad de los atributos de calidad
6	Análisis de las propuestas arquitectónicas
7	Lluvia de ideas y priorización de escenarios
8	Análisis de las propuestas arquitectónicas

9	Presentación de los resultados

Tabla 25. Presentación Metodología ATAM

3.6.2.2. Presentación de las metas del negocio

Se realiza la descripción de las metas del negocio que motivan el esfuerzo y se identificaron los atributos de calidad en los requerimientos a partir de los casos de uso que hacen parte de la arquitectura de software diseñada, a continuación, se muestra en una tabla dichos atributos de calidad.

Caso de uso	Atributo de calidad	Descripción
Inicio de Sesión	Disponibilidad	Cuando el usuario requiera la autenticación el servicio de autenticación debe estar funcionando.
	Usabilidad	La interfaz de usuario del formulario de iniciar sesión debe ser amigable e intuitiva.
	Integridad	El acceso a la plataforma puede ser autorizado con la validez exitosa del servicio de autenticación de usuario.
Gestión Año Lectivo	Integridad	Se debe validar el acceso a la aplicación para evitar el acceso no autorizado a la información de la funcionalidad.
	Usabilidad	Las interfaces de usuario que hacen parte de esa funcionalidad deben ser amigables e intuitivas para un correcto uso del sistema.
	Mantenimiento	Se debe diseñar la funcionalidad de tal manera que sea rápido y fácil hacer modificaciones.
Gestión Estudiante	Rendimiento	Los tiempos de respuesta de las peticiones al servidor deben ser bajo.
	Usabilidad	Las interfaces de usuario que hacen parte

		de esa funcionalidad deben ser amigables e intuitivas para un correcto uso del sistema.
	Eficiencia	Los recursos de hardware y software utilizados por esta funcionalidad deben ser los adecuados para evitar rendimientos bajos del sistema.
Gestión Docente	Usabilidad	Las interfaces de usuario que hacen parte de esa funcionalidad deben ser amigables e intuitivas para un correcto uso del sistema.
	Mantenimiento	Se debe diseñar la funcionalidad de tal manera que sea rápido y fácil hacer modificaciones.
	Confiabilidad	La información cargada desde la base de datos y mostrada al usuario por esta aplicación no debe tener errores y ser precisa.
Gestión Curso	Eficiencia	Los recursos de hardware y software utilizados por esta funcionalidad deben ser los adecuados para evitar rendimientos bajos del sistema.
	Usabilidad	Las interfaces de usuario que hacen parte de esa funcionalidad deben ser amigables e intuitivas para un correcto uso del sistema.
	Integridad	Se debe validar el acceso a la aplicación para evitar el acceso no autorizado a la información de la funcionalidad.
	Confiabilidad	La información cargada desde la base de datos y mostrada al usuario por esta aplicación no debe tener errores y ser precisa.
	Rendimiento	La capacidad de respuesta del sistema al

Gestión Asignatura		momento de ejecutar una petición debe estar dentro del intervalo de tiempo optimo.
	Seguridad	El sistema debe estar protegido de perder, suministrar información o modificación de la información a fuentes sin autorización.
Gestión Área	Usabilidad	Las interfaces de usuario que hacen parte de esa funcionalidad deben ser amigables e intuitivas para un correcto uso del sistema.
	Seguridad	El sistema debe estar protegido de perder, suministrar información o modificación de la información a fuentes sin autorización.
	Mantenimiento	Se debe diseñar la funcionalidad de tal manera que sea rápido y fácil hacer modificaciones.
Matricular Alumno	Eficiencia	Los recursos de hardware y software utilizados por esta funcionalidad deben ser los adecuados para evitar rendimientos bajos del sistema.
	Usabilidad	Las interfaces de usuario que hacen parte de esa funcionalidad deben ser amigables e intuitivas para un correcto uso del sistema.
	Integridad	Se debe validar el acceso a la aplicación para evitar el acceso no autorizado a la información de la funcionalidad y evitar acciones que afecten el sistema.
Gestión Usuarios	Mantenimiento	Se debe diseñar la funcionalidad de tal manera que sea rápido y fácil hacer modificaciones.
	Confiabilidad	La información cargada desde la base de datos y mostrada al usuario por esta aplicación no debe tener errores y ser

		precisa.
	Rendimiento	La capacidad de respuesta del sistema al momento de ejecutar una petición debe estar dentro del intervalo de tiempo optimo.
Gestión Grados	Usabilidad	Las interfaces de usuario que hacen parte de esa funcionalidad deben ser amigables e intuitivas para un correcto uso del sistema.
	Eficiencia	Los recursos de hardware y software utilizados por esta funcionalidad deben ser los adecuados para evitar rendimientos bajos del sistema.
	Mantenimiento	Se debe diseñar la funcionalidad de tal manera que sea rápido y fácil hacer modificaciones.
Gestión Horarios	Confiabilidad	La información cargada desde la base de datos y mostrada al usuario por esta aplicación no debe tener errores y ser precisa.
	Rendimiento	La capacidad de respuesta del sistema al momento de ejecutar una petición debe estar dentro del intervalo de tiempo optimo.
	Seguridad	El sistema debe estar protegido de perder, suministrar información o modificación de la información a fuentes sin autorización.
Gestión Periodos Académicos	Funcionalidad	La funcionalidad debe realizar las funciones para las que fue desarrollada.
	Madurez	La funcionalidad debe permitir las excepciones para impedir el mal correcto funcionamiento de la funcionalidad ante fallas.
	Usabilidad	La funcionalidad debe permitir la facilidad

		de aprendizaje al usuario al momento de utilizarla.
Gestión Carga Académica	Seguridad	El sistema debe estar protegido de perder, suministrar información o modificación de la información a fuentes sin autorización.
	Eficiencia	Los recursos de hardware y software utilizados por esta funcionalidad deben ser los adecuados para evitar rendimientos bajos del sistema.
	Usabilidad	El sistema debe tener la capacidad de permitirle al usuario operarlo y controlarlo.
Gestión de Logros Académicos	Confiabilidad	La información cargada desde la base de datos y mostrada al usuario por esta aplicación no debe tener errores y ser precisa.
	Rendimiento	La capacidad de respuesta del sistema al momento de ejecutar una petición debe estar dentro del intervalo de tiempo optimo.
	Seguridad	El sistema debe estar protegido de perder, suministrar información o modificación de la información a fuentes sin autorización.
Carga Académica Estudiante	Usabilidad	Las interfaces de usuario que hacen parte de esa funcionalidad deben ser amigables e intuitivas para un correcto uso del sistema.
	Eficiencia	Los recursos de hardware y software utilizados por esta funcionalidad deben ser los adecuados para evitar rendimientos bajos del sistema.
	Mantenimiento	Se debe diseñar la funcionalidad de tal manera que sea rápido y fácil hacer modificaciones.

Horario Estudiante	Confiabilidad	La información cargada desde la base de datos y mostrada al usuario por esta aplicación no debe tener errores y ser precisa.
	Rendimiento	La capacidad de respuesta del sistema al momento de ejecutar una petición debe estar dentro del intervalo de tiempo optimo.
	Seguridad	El sistema debe estar protegido de perder, suministrar información o modificación de la información a fuentes sin autorización.
Carga Académica Docente	Usabilidad	Las interfaces de usuario que hacen parte de esa funcionalidad deben ser amigables e intuitivas para un correcto uso del sistema.
	Mantenimiento	Se debe diseñar la funcionalidad de tal manera que sea rápido y fácil hacer modificaciones.
	Confiabilidad	La información cargada desde la base de datos y mostrada al usuario por esta aplicación no debe tener errores y ser precisa.
Horario Docente	Mantenibilidad	La funcionalidad debe permitir al usuario el menor esfuerzo y tiempo al mantenerlo o mejorarlo.
	Usabilidad	Las interfaces de usuario que hacen parte de esa funcionalidad deben ser amigables e intuitivas para un correcto uso del sistema.
	Rendimiento	La capacidad de respuesta del sistema al momento de ejecutar una petición debe estar dentro del intervalo de tiempo optimo.
	Mantenibilidad	La capacidad de la funcionalidad debe evitar efectos inesperados debido a

Gestión Roles		modificaciones realizadas en dicha funcionalidad.
	Confiabilidad	La información cargada desde la base de datos y mostrada al usuario por esta aplicación no debe tener errores y ser precisa.
	Usabilidad	Las interfaces de usuario que hacen parte de esa funcionalidad deben ser amigables e intuitivas para un correcto uso del sistema.
Cierre de Sesión	Rendimiento	La capacidad de respuesta del sistema al momento de ejecutar una petición debe estar dentro del intervalo de tiempo optimo.
	Eficiencia	Los recursos de hardware y software utilizados por esta funcionalidad deben ser los adecuados para evitar rendimientos bajos del sistema.
	Madurez	La funcionalidad debe permitir las excepciones para impedir el mal correcto funcionamiento de la funcionalidad ante fallas.

Tabla 26. Metas del Negocio

3.6.2.3. Presentación de la Arquitectura

Se realizo la presentación de la arquitectura diseñada a los stakeholders la cual se encuentra en el punto **3.5. DISEÑO ARQUITECTURA DE SOFTWARE**, dicha arquitectura se diseñó abordando el modelo de negocio del colegio Integrado Atalaya. En la

presentación de la arquitectura se mostró el patrón arquitectónico utilizado el cual es el modelo de “4+1” Vistas de la Arquitectura de Software, dicho modelo consta de 4 vistas y un escenario. Las vistas del modelo son la vista lógica, la vista de procesos, la vista física, la vista de desarrollo y por último el escenario, se mostraron los requerimientos funcionales y no funcionales, los recursos humanos, los recursos tecnológicos, las restricciones y los diferentes diagramas que se definieron en las vistas del modelo “4+1”, en el escenario se definieron los casos de uso.

3.6.3. Fase 2. Investigación y análisis

En la fase 2 de **ATAM** se realizan las actividades que corresponden al grupo de investigación y análisis el cual contiene 3 pasos los cuales son Identificación de los enfoques arquitectónicos, Generar el árbol de utilidad de los atributos de calidad y Análisis de las propuestas arquitectónicas.

3.6.3.1. Identificación de los enfoques arquitectónicos

En la identificación de la propuesta arquitectónica la decisión que se tomó sobre los estilos o patrones utilizados se llevó a cabo estudiando la arquitectura presentada la cual permitió tomar la decisión en base a los requisitos funcionales que se identificaron en la arquitectura propuesta y en las tecnologías con las que se plantea trabajar, se observó en el modelo de dominio la descomposición del sistema en diferentes componentes lo cual permite tener una programación modular en la aplicación.

Se plantearon dos patrones arquitectónicos los cuales trabajaran desde el fronted o el servidor, desde el fronted se trabajará con el patrón Modelo Vista Controlador (MVC) el cual permite tener separada la vista de usuario de la lógica del modelo de negocio en el controlador, este a

su vez se encarga de recibir y direccionar al modelo para que se encargue de la persistencia de la información. Para el servidor se trabajará con el patrón Modelo Controlador (MC) dado que se propone trabajar el backend por medio de servicios esto permite tener una integración continua y probar nuevas funcionalidades y revertir si el sistema presenta algún fallo crítico, dicho patrón solo cuenta con el Modelo y Controlador en donde el controlador recibe la petición del usuario y se encarga de direccionar al modelo para la persistencia de la información.

3.6.3.2. Generar el árbol de utilidad de los atributos de calidad

Se realizo el árbol de utilidad de los atributos de calidad del sistema el cual consta de disponibilidad, usabilidad, modularidad, integridad, Confidencialidad, mantenimiento y seguridad los cuales fueron especificados en forma de escenarios y necesidades.

ARBOL DE UTILIDAD		
Atributo de Calidad	**Necesidad**	**Escenario**
Integridad	El sistema debe permitir el acceso a las funcionalidades que tiene el rol del usuario.	Los usuarios solo podrán tener acceso a las funcionalidades del sistema que tiene asignado dicho rol (A, A).
Usabilidad	El sistema debe ser intuitivo y amigable al usuario.	El sistema tiene una interfaz de usuario amigable e intuitiva la cual permite facilidad al usar (M, A).
Confidencialidad	El sistema debe permitir la protección de la información contra el acceso de datos no	El sistema cuenta con una protección de la información por medio de autenticación

	autorizados.	(M,A).
Disponibilidad	El sistema debe estar operativo y accesible para cuando el usuario lo requiera.	El sistema está disponible 24/7 para cuando un usuario lo requiera (M, A).
Mantenimiento	El sistema debe permitir al programador realizar mejoras de forma rápida y fácil.	El sistema debe estar desarrollado de tal manera que las mejoras se puedan realizar fácil y rápido sin inconvenientes (M, A)
Modularidad	El sistema debe estar diseñado de forma modular para facilitar el acceso a las funcionalidades.	El sistema cuenta con módulos los cuales tienen funcionalidades (M, M)

Tabla 27. Árbol de Utilidad

3.6.3.3. Análisis de las propuestas arquitectónicas

Se realizo la asociación de los diferentes escenarios que hacen parte del árbol de utilidad con los estilos utilizados en la arquitectura diseñada para identificar los atributos de calidad requeridos. La Integridad hace parte de los atributos de calidad el cual en esta arquitectura está asociado con los diagramas de actividades del sistema que muestran el comportamiento de los procesos que se encargan de gestionar la información del sistema y validar la autenticación de acceso del usuario si posee los privilegios o permisos. La Usabilidad hace parte de los atributos de calidad el cual en esta arquitectura se asocia con los diagramas de actividades y casos de uso, en los cuales se muestran los diferentes procesos que interactúan con el usuario. La disponibilidad hace parte de los atributos de calidad el cual en esta arquitectura está asociado con el diagrama de despliegue del sistema, este diagrama está

compuesto por la infraestructura física y de software muy necesarias para el correcto funcionamiento del sistema y su disponibilidad, se propuso una infraestructura web porque garantiza la disponibilidad a los usuarios en el momento que lo necesiten y tengan una conexión de internet. La arquitectura diseñada está compuesta por diferentes modelos y servicios los cuales están asociados al atributo de calidad de modularidad lo que permite que el sistema se pueda adaptar en futuras necesidades con facilidad y no será necesario implantar un sistema de gestión completo dado que será flexible y fácil de mantener.

3.6.4. Fase 3. Pruebas

En la fase 3 de ATAM se realizan las actividades que corresponden al grupo de investigación y análisis el cual contiene 2 pasos los cuales son Lluvia de ideas y priorización de escenarios y Análisis de las propuestas arquitectónicas.

3.6.4.1. Lluvia de ideas y priorización de escenarios

Como resultado del análisis de las propuestas arquitectónicas que se realizó se logró encontrar nuevos escenarios que no se especificaron en el árbol de utilidad de los atributos de calidad, por tal motivo se vio la necesidad de volver a generarlo para tenerlos en cuenta.

ARBOL DE UTILIDAD		
Atributo de Calidad	**Necesidad**	**Escenario**
Robustez	El sistema debe ser modificado sin agregar errores y opere correctamente.	El sistema se modifica y se realizan las respectivas pruebas para evitar agregar errores. (A, A).
Tolerancia a fallos	El sistema debe operar según lo previsto en presencia de	El sistema sigue operando ante fallas inesperadas tanto

	fallos hardware o software.	en el hardware como en el software (M, A).
Consistencia	En las operaciones del usuario debe existir coherencia.	Las operaciones que realiza el usuario son coherentes con las funcionalidades que tiene el sistema (M, A).
Escalabilidad	El sistema debe trabajar con diferentes cantidades de trabajo como lo son el volumen de datos.	El sistema esta diseñado para que soporte grandes cantidades de volumen de datos lo cual lo hace escalable (M, A).
Integridad	El sistema debe permitir el rol de usuarios para la restricción de acceso a funcionalidades del sistema en roles específicos.	El sistema valida el acceso a la funcionalidad que se desea cargar el usuario con el fin de no restringir el acceso a roles que no tienen asignada la funcionalidad.

Tabla 28. Árbol de Utilidad Actualizado

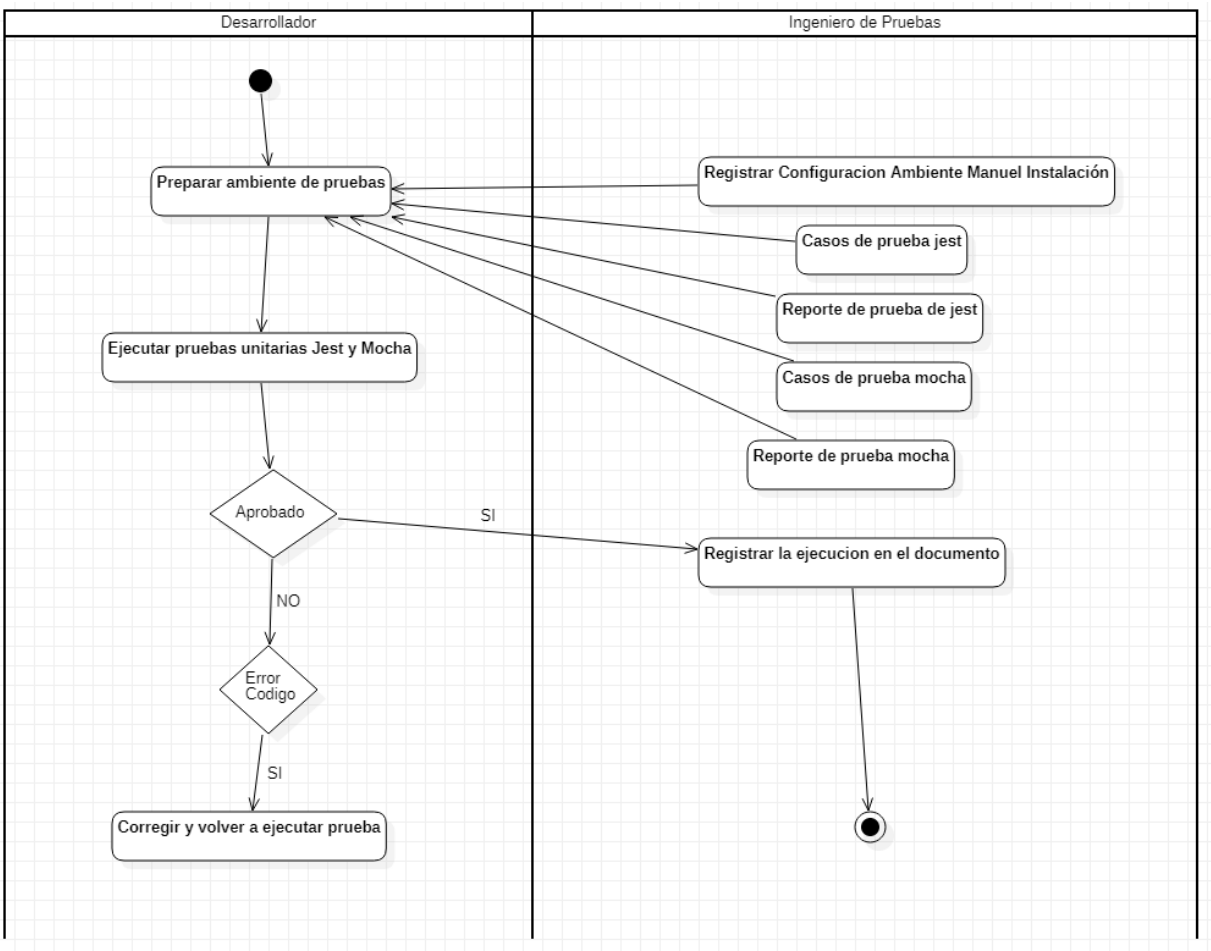

Figura 60. Diagrama de Pruebas Unitarias

3.6.4.2. Análisis de las propuestas arquitectónicas

Se realizo nuevamente un análisis de las propuestas arquitectónicas en donde se consideró los escenarios con más alta prioridad esto para decidir aquellos cambios necesarios que se deben realizar en la arquitectura diseñada. Los escenarios con más alta prioridad son los atributos de calidad que se especifican en el árbol de utilidad los cuales son robustez, tolerancia a fallos, consistencia, escalabilidad e integridad, se analizaron estos atributos y como están especificados en la arquitectura diseñada, de los 5 atributos de calidad mencionados en el árbol de utilidad los que se deben soportar de otra forma son la robustez, tolerancia a fallos, dado que no hicieron parte de las primeras fases del diseño. La robustez

está relacionada con los diferentes procesos que se establecieron a través de los casos de uso de la arquitectura diseñada, ya que los cambios que se realicen en el sistema no deben generar errores y funcionar correctamente. La tolerancia a fallos se relaciona con el diseño de la arquitectura dado que el sistema puede llegar a presentar fallas en algunos de sus servicios, pero seguirá funcionando porque el fallo de un servicio no afecta el resto.

3.6.5. Fase 4. Reportes

En la fase 4 que es la última de ATAM se realiza la actividad que corresponden al grupo de Reportes y contiene un paso el cual es Presentación de los resultados.

3.6.5.1. Presentación de los resultados

Se presentaron los resultados a los stakeholders basado en la información recolectada a lo largo de la evaluación en donde se explicaron cada uno de los pasos de la metodología de ATAM los cuales fueron las metas del negocio, presentación arquitectura, enfoques arquitectónicos, árbol de utilidad de atributos de calidad, análisis de las propuestas arquitectónicas, lluvias de ideas y priorización de escenarios y por último el análisis de las propuestas arquitectónicas.

3.7. PRESUPUESTO DESARROLLO SOFTWARE ACADEMICO

Se realiza un presupuesto para el desarrollo del software de gestión académica en el colegio Integrado Atalaya la cual consta de las funcionalidades, equipo de desarrollo, tecnologías a utilizar, herramientas de trabajo, características de servidor, soporte, mantenimiento, tiempo estimado de desarrollo y el costo por hora de mano de obra.

3.7.1. Funcionalidades del Sistema

Para empezar, se especificará las funcionalidades con las que contara el software que se planteó en el diseño de la Arquitectura de Software, a continuación, se listaran las funcionalidades.

- Inicio de Sesión
- Gestión Año Lectivo
- Gestión Estudiante
- Gestión Docente
- Gestión Curso
- Gestión Asignatura
- Gestión Área
- Matricular Alumno
- Gestión Usuarios
- Gestión Grados
- Gestión Horarios
- Gestión Periodos Académicos
- Gestión Carga Académica
- Gestión de Logros Académicos
- Notas Académicas de estudiantes por materia
- Carga Académica Estudiante
- Horario Estudiante
- Carga Académica Docente

- Horario Docente
- Gestión Roles
- Cierre de Sesión

3.7.2. Equipo de Desarrollo

El equipo de desarrollo con el que debe contar el colegio es de dos Desarrolladores de Software que tengan mínimo un año de experiencia y tengan conocimiento en el lenguaje Javascript, NodeJS, NestJS, SequelizeJS, Angular, base de datos PostgreSQL, administración de VPS CON Cpanel WHM.

3.7.3. Tecnologías

Las tecnologías que se utilizaran para el desarrollo del software de gestión académica son las siguientes: por parte del cliente es Angular y del lado del servidor NodeJS, NestJS y SequelizeJS, para la persistencia de la información se utilizara la base de datos PostgreSQL.

3.7.4. Herramientas

Las herramientas que se utilizaran son un editor de código como lo es Visual Studio Code, para la gestión de la base de datos se utilizara pgAdmin4 y DBeaver Community, es necesario la instalación en el equipo de NodeJS, Angular, Base de Datos PostgreSQL

3.7.5. Características del Servidor

Para la ejecución de la plataforma en producción es necesario de un servidor el cual debe cumplir con ciertas características que permitan la confiabilidad y disponibilidad del servicio 24/7, a continuación, se especifican las características con las que debe cumplir el servidor.

- 4 Core CPU
- 6GB de memoria RAM DDR4
- 180 GB de Almacenamiento SSD
- Panel de control WHM
- 2 IPs
- SSL
- Servicio Nginx
- Servicio PostgreSQL

3.7.6. Soporte

Se pueden ofrecer dos tipos de soporte, el soporte a los usuarios que utilizan la plataforma cuando tienen algunas inquietudes con respecto a funcionalidades o necesitan realizar ciertas acciones las cuales no están autorizados. El otro soporte seria con respecto a fallas que se puedan presentar en la plataforma académica cuando se está utilizando. Se aconseja tener un desarrollador de planta que tenga conocimiento en el sistema y en los lenguajes que se utilizaron para prestar dichos soportes que se pueden presentar en la utilización de la plataforma académica.

3.7.7. Mantenimiento

Se aconseja realizar mantenimiento al sistema, la base de datos y desarrollar nuevas funcionalidades que sean a la medida del colegio.

3.7.8. Tiempo de Desarrollo

Se estima un tiempo estimado de 4 meses para desarrollar la plataforma académica con 3 desarrolladores y un líder de proyecto dado que son 21 requerimientos los que se deben cumplir, se propone realizar 8 sprint los cuales se trabajara de la siguiente forma: 7 sprint que contemplan el desarrollo completo de los requerimientos y en el 8 sprint las pruebas pertinentes y correcciones encontradas. Cabe mencionar que cada Sprint será de máximo 2 semanas.

3.7.9. Costos

Se mostrará en una tabla los diferentes costos para el desarrollo del software académico. Para cada desarrollador se le puede asignar un sueldo de $2.500.000 de pesos y el líder técnico $3.000.000 de pesos ambos sueldos con los parafiscales que rigen en Colombia, estos salarios hacen parte del promedio en el mercado actual de desarrolladores juniors y lideres técnicos en Colombia. El servidor que se requiere para el despliegue y funcionamiento de la aplicación tiene un costo de $1.640.064 de pesos por año. Los equipos para los desarrolladores son con características altas pues se necesitan tener varias aplicaciones en ejecución que son un poco pesadas y se necesita de una buena máquina.

COMPONENTE	CANTIDAD	COSTO UNITARIO	COSTO TOTAL
MANO DE OBRA			

Desarrollador	4 meses	$7.500.000	$30.000.000
Líder Técnico	4 meses	$3.000.000	$12.000.000
HARDWARE			
Servidor WEB	1 año	$1.640.064	$1.640.064
Equipo desarrollador	3	$2.856.000	$8.568.000
SOFTWARE			
Visual Studio Code	1	0	0
pgAdmin4	1	0	0
DBeaver Community	1	0	0
NodeJS	1	0	0
Angular	1	0	0
PostgreSQL	1	0	0
		Total	$52.208.064

CONCLUSIONES

Este trabajo me ha permitido diseñar una arquitectura de software para la gestión académica a partir de las necesidades de un colegio el cual desea tener su propia plataforma académica, en el desarrollo de este proyecto se logró trabajar con el tipo de Investigación proyectiva, el método sistémico, el Modelo 4+1 y sus diferentes vistas las cuales son Vista Lógica, Vista de Desarrollo, Vista Física, Vista de Procesos y por último Escenarios.

BIBLIOGRAFIA

Sampieri, R.H & Collado, C & Baptista, M.P. (2010). Metodología de la Investigación Quinta Edición. México D.F: McGRAW-HILL / INTERAMERICANA EDITORES, S.A. DE C.V.

Sommerville, I. (2005). Ingenieria del Software (7.a ed.). Pearson Education.

L. Bass, P. Clements, R. Kazman, Software Architecture in Practice, 2nd Edition, Addison Wesley, 2003

Campderrich Falgueras, B. (2003). Ingeniería del software (Primera edición). Editorial UOC.

C. Martin, R. (2004). UML para Programadores Java: Vol. Primera Edición. Pearson Education.

Pressman, R. S. (2010). Ingenieria del Software un enfoque práctico (Septima Edición). Mc Graw Hill.

Kruchten, P. (1995). Planos Arquitectónicos: El Modelo de 4+1 Vistas de la Arquitectura del Software. IEEE Software 12.

Sommerville, I. (2011). Ingenieria de Software (Novena Edición). Pearson Education.

Pressman, R. S. (2001). Ingenieria de Software un enfoque practico (Quinta edición). Mc Graw Hill.

Martinez López, F. J., & Gallegoz Ruiz, A. (2017). *Programación de Bases de Datos Relacionales*. Ra-Ma.

Hurtado De Barrera, J. (2010). *Metodología de la Investigación Guía para la comprensión holística de la ciencia* (Cuarta Edición). Quirón Ediciones.

Delgado, A., Castro, A., & Germán, M. (2017). Evaluación de Arquitecturas de Software con ATAM (Architecture Tradeoff Analysis Method): un caso de estudio. *Instituto de Computación – Facultad de Ingeniería*, *1*(1).

ANEXOS

ANEXO A

COTIZACIÓN PLATAFORMA ACADEMICA AULAKVA

Contacto: +593 96 769 6597
Email: rpsolutions26@gmail.com
Demo: www.demosisedcol.com
Aula virtual: www.aulakva.com

Plataforma Virtual de Gestión Académico

Somos R&P Solutions ponemos a su disposición nuestra plataforma de Gestión Académica y Aula virtual con las siguientes características

CARACTERÍSTICAS DE NUESTRA PLATAFORMA

- Interfaz amigable
- Gestión de cursos
- Gestión de paralelos/secciones
- Gestión de matrículas online
- Exámenes
- Tareas
- Foros
- Material de estudio
- Reportes administrativos
- Video conferencia integrada sin límite de usuarios
- Registro de cobros de pensiones
- Chat entre usuarios
- Registro de asistencias
- Registro de notas
- Biblioteca virtual
- Portafolio de archivos
- Reportes académicos

Accesos al demo

Administrador
Usuario: admin
Contraseña: 4321
Docente
Usuario: apucaf
Contraseña: 4321
Estudiante
Usuario: vrosado
Contraseña: holaF

Entregamos su plataforma en 48 horas

Para contribuir con la educación por motivos de la situación grave que estamos viviendo hemos cambiado nuestros costos adecuándolos para que todos tengan acceso.

Plan básico máximo 100 estudiantes al año
$180
Incluye:

- Hosting dedicado
- Domininio .com que el cliente elija
- Certificado SSL
- Almacenamiento 40GB
- Manuales de usuario (video y PDF)
- Soporte técnico permanente las 24 horas

Plan estándar hasta 250 estudiantes al año
$400
Incluye:

- Hosting dedicado
- Domininio .com que el cliente elija
- Certificado SSL
- Almacenamiento 80GB
- Manuales de usuario (video y PDF)
- Página web auto administrable
- Soporte técnico permanente las 24 horas

Plan Premium hasta 400 estudiantes al año
$850
Incluye:

- Hosting dedicado
- Domininio .com que el cliente elija
- Certificado SSL
- Almacenamiento 140GB
- 5 Correos institucionales
- Manuales de usuario (video y PDF)
- Página web auto administrable
- Soporte técnico permanente las 24 horas

Plan Premium sin límite, estudiantes ilimitados por año
$1800
Incluye:

- Hosting dedicado cloud
- Domininio .edu.ec que el cliente elija
- Certificado SSL
- Almacenamiento 200GB
- 15 Correos institucionales
- Manuales de usuario (video y PDF)
- Página web auto administrable
- Soporte técnico permanente las 24 horas

Planes Mensuales

	BÁSICO	INTERMEDIO	COMPLETO
Planes	**$120 Por mes**	**$300 Por mes**	**$480 Por mes**
Cantidad de usuarios	Hasta 400 estudiantes	Hasta 1000 estudiantes	Hasta 1500 estudiantes
- Matriculación Online	✓	✓	✓
- usuarios para Administradores, Docentes, Estudiantes, Padres de familia	✓	✓	✓
- Aula virtual	✓	✓	✓
- Vídeo conferencia sin límite de tiempo	✓	✓	✓
- Registro de asistencia	✓	✓	✓
- Registro de calificaciones	✓	✓	✓
- Registro de pagos de pensiones	✓	✓	✓
- Reportes administrativos	✓	✓	✓
- Reportes Académicos	✓	✓	✓
- Chat integrado	✓	✓	✓
- Biblioteca virtual	✓	✓	✓

-Página web administrable	✓	✓	✓
- 10 correos institucionales		✓	✓
Soporte técnico las 24 horas del día	✓	✓	✓
Certificado SSL	✓	✓	✓
Hosting dedicado	✓	✓	✓
Almacenamiento de 140GB	✓	✓	✓

PLAN POR SUSCRIPCIÓN (SOLO AULA VIRTUAL)

	Docentes	**Estudiantes**	**Padres**
Plan aula Virtual	$3 Por mes	$2,50 Por mes	$2 Por mes
Examen online			
Tareas online			
Foros online			
Material de estudio			
Chat entre usuarios			
Video conferencia			
Calificaciones			
Asistencias			
Horario de clases			
Asistencia técnica las 24 horas			
Manuales de usuarios en video y pdf			
Capacitación a los docentes por video conferencia			

Si los planes mencionados no se aplican para su institución indicar la situación actual de la institución y le indicaremos valores acordes de cada institución

Formas de pago

- Depósito mediante cuenta Bancaria
- Transferencia
- Tarjeta de crédito
- Paypal
- Efectivo
- Cheque

Se emitirá Documento de contrato antes de realizar los cobros los cuales serán cancelados de la siguiente manera:

- 40% al firmar el contrato
- 60% al entregar las credenciales del administrador y levantado todos los servicios mencionados.

Garantías

- Ofrecemos asesorías técnicas las 24 horas del día, en caso de fallos en la plataforma contactar con nuestro servicio técnico el cual estará disponible siempre.
- Respaldos diarios de la base de datos.
- Seguridad de los datos de usuarios encriptados en la base de datos.
- Libre de ataques de DDOS.
- Actualizaciones permanentes de nuestra plataforma sin costos adicionales por nuevos módulos que se vayan agregando.

Tiempo de vida de la plataforma

Nuestra plataforma se encuentra habilitada para funcionamiento permanente firmamos los contratos con los clientes quienes deciden si seguir usando nuestra plataforma o no, los cuales tienen total libertad de terminar el contrato en el cual se determina que tiempo tiene de validez.

ANEXO B

CALCULO DE RETORNO DE INVERSIÓN (ROI) DE SOFTWARE ACADEMICO

Cálculo de Retorno de Inversión (ROI)

Cálculo de ROI para Software Academico

Se plantea un total de 4000 estudiantes por año, por estudiante el costo es de $7,000 mil pesos al año.

Proyecto	Software
Inversión Inicial	$ 52.208.064,00
Ingresos proyectados	
Año 2022	$ 28.000.000,00
Año 2023	$ 28.000.000,00
Año 2024	$ 28.000.000,00
Tasa de descuento	5%

Resultados:	Software
ROI	46%

ANEXO C

CRONOGRAMA DE ACTIVIDADES

UNIVERSIDAD SIMÓN BOLÍVAR Barranquilla y Cúcuta - Colombia	CRONOGRAMA DEL PROYECTO	Cód:	2,02111E+12
		Mod:	
		Ver	
NOMBRE DEL PROYECTO	DESARROLLO SOFTWARE DE GESTION ACADEMICA		
DURACIÓN DE LA EJECUCIÓN DEL PROYECTO EN SPRINT	2 SEMANAS SON 1 SPRINT		
GRUPO DE TRABAJO	3 DESARROLLADORES Y 1 LIDER TECNICO		

	ACTIVIDADES	SPRINT															
		1		2		3		4		5		6		7		8	
1	Inicio de Sesión																
2	Gestión Año Lectivo																
3	Gestión Estudiante																
4	Gestión Docente																
5	Gestión Curso																
6	Gestión Asignatura																
7	Gestión Área																
8	Matricular Alumno																
9	Gestión Usuarios																
10	Gestión Grados																
11	Gestión Horarios																
12	Gestión Periodos Académicos																
13	Gestión Carga Académica																
14	Gestión de Logros Académicos																
15	Notas Académicas de estudiantes por materia																
16	Carga Académica Estudiante																
17	Horario Estudiante																
18	Carga Académica Docente																
19	Horario Docente																
20	Gestión Roles																
21	Pruebas y Correciones del software																
19	Horario Docente																
20	Gestión Roles																
21	Pruebas y Correciones del software																

ANEXO D

FUNCIONALIDADES DE PLATAFORMA PHIDEAS

Phidias

Phidias es una plataforma de gestión académica que fue desarrollada por los hermanos Santiago y Juan Sebastián Cortés en la ciudad de Bogotá y se utiliza en 7 países los cuales son Colombia, España, Panamá, México, Cuba, Francia y Bélgica, es una plataforma muy completa que cuenta con 4 módulos los cuales son: Módulo Académico, Módulo Tesorería, Módulo Administración y Módulo Comunicación.

Las funcionalidades de cada módulo son tomadas directamente desde la página web de la plataforma www.phidias.co.

Módulo Académico

Evalúa y consulta tus materias de acuerdo al esquema de calificación que manejes (rúbricas, competencias, sistema IB, entre otros). Phidias facilita el registro y el análisis de resultados académicos, ofreciendo reportes de desempeño, información en tiempo real para familiares y estudiantes, además de otras funcionalidades.

- Administra y publica el plan estratégico y curricular.
- Comparte archivos y planes académicos.
- Crea deberes y material de apoyo.
- Crea exámenes en línea con calificación automática.
- Envía notificaciones de tareas y exámenes.
- Personaliza boletines, libros de notas y otros formatos.
- Genera y envía boletines e informes.
- Registra menciones y reconocimientos.
- Controla las fechas de modificación y consulta de notas.
- Programa eventos en el calendario.
- Asigna múltiples docentes y directores de curso.
- Realiza comentarios personalizados.
- Gestiona observaciones.

- Crea informes por materias, sección, indicadores, ranking, entre otros.
- Integra otras plataformas con Phidias.

Módulo Tesorería

Mediante una propuesta innovadora de planeación de documentos, podrás generar eficientemente facturas para todos los usuarios reutilizando la información disponible en otros módulos (administrativo, académico, extraescolares, transporte). Los usuarios pueden, inclusive desde la app, ver y pagar sus obligaciones.

- Gestiona cobros y genera estados de cuenta.
- Registra notas de crédito y débito.
- Genera facturas electrónicas, sin intermediarios: Phidias es un Proveedor Tecnológico (PT) autorizado por la DIAN.
- Crea justificantes de pago y certificados.
- Planifica la facturación con conceptos y descuentos ilimitados, con espacios para especificar el PUC.
- Configura recargos o intereses en facturas vencidas.
- Administra y genera reportes dinámicos de facturas, pagos y cartera.
- Produce informes de situación clasificados por conceptos, cursos y edades.
- Crea informes personalizados a través de Análisis de datos. (Saber +)
- Intégrate con sistemas estándares de pago, domiciliación, contabilidad, etc.

Módulo Administración

Gestiona el perfil del alumno y su núcleo familiar. Permite analizar las estadísticas de tu centro para la toma de decisiones a nivel administrativo.

- Crea perfiles y usuarios de manera ilimitada.
- Completa fichas de datos personalizables con la información de los usuarios.
- Gestiona documentos pendientes.
- Gestiona los permisos para cada tipo de usuario.
- Comprueba estadísticas de uso del sistema.
- Consulta la información de cada familia.

- Lleva el control de los responsables de cada estudiante y sus datos de contacto.
- Accede a los logs de actividad para las auditorías.
- Conserva copias de seguridad diarias.
- Integra otras plataformas con Phidias.

Módulo Comunicación

Gestión centralizada de la comunicación del colegio. Divulga información a través de varios canales (e-mail, app móvil y web), siendo siempre respetuosos con los datos personales.

- Envía mensajes internos, masivos o individuales.
- Envía comunicados y circulares.
- Adjunta archivos desde tu dispositivo y desde Google Drive.
- Incrusta formularios y vídeos directamente en tus mensajes.
- Programa el envío en la fecha y hora que desees.
- Verifica quién ha leído los mensajes.
- Agenda citaciones a reuniones y tutorías.
- Envía notificaciones (push al móvil, alertas SMS, e-mail, etc.)
- Permite o desactiva las respuestas de otros usuarios.
- Integra otras plataformas con Phidias.