UNIVERSIDAD AUTÓNOMA GABRIEL RENÉ MORENO



“Smart Condominium”

**Aplicación Móvil y Web con Inteligencia Artificial para Administración de Condominios**

**Estudiantes:**

-

-

**Materia:** Sistemas de información 2

**Docente:** Ing. Angelica Garzón Cuellar

**Gestión: 2**-**2025**

Contenido

[**CAPITULO 1 : FUNDAMENTOS TEORICOS** 3](#_Toc138193292)

[**1.1** **Introducción** 4](#_Toc138193293)

[**1.2** **Objetivo General** 4](#_Toc138193294)

[**1.3** **Objetivos Específicos** 5](#_Toc138193295)

[**1.4** **Alcance** 6](#_Toc138193296)

[**1.5** **Marco Teórico** 8](#_Toc138193297)

[**CAPITULO 2 : HERRAMIENTAS TECNOLOGICAS PARA EL DESARROLLO** 10](#_Toc138193298)

[**CAPITULO 3:TECNOLOGÍA PARA EL DESARROLLO DEL SOFTWARE** 13](#_Toc138193299)

[**3.1 ESTRATEGIA PARA EL DESARROLLO DEL SOFTWARE** 13](#_Toc138193300)

[**3.2 METODOLOGÍA PARA EL DESARROLLO DEL SOFTWARE** 14](#_Toc138193301)

[**3.2.1 CARACTERÍSTICAS DEL PUDS** 14](#_Toc138193302)

[**3.2.2 CARACTERÍSTICAS DE UML** 16](#_Toc138193303)

[**CAPITULO 4 : FLUJO DE TRABAJO: CAPTURA DE REQUISITOS** 19](#_Toc138193304)

[**4.1 Actores** 20](#_Toc138193305)

[**4.2 Casos de Uso** 20](#_Toc138193306)

[**4.3 Priorizar Caso de Uso** 20](#_Toc138193307)

[**4.4 Detalle de Caso de uso** 21](#_Toc138193308)

[**4.5 Estructurar Modelo de Caso de Uso** 33](#_Toc138193309)

[**CAPÍTULO 5: FLUJO DE TRABAJO: ANÁLISIS** 34](#_Toc138193310)

[**5.1 ANÁLISIS DE LA ARQUITECTURA** 35](#_Toc138193311)

[**5.1.1 IDENTIFICAR PAQUETE** 35](#_Toc138193312)

[**5.1.2 RELACIÓN PAQUETE – CASO DE USO** 36](#_Toc138193313)

[**5.2 Análisis de Casos de Uso** 38](#_Toc138193314)

[**5. 3 ANÁLISIS DE CLASES** 39](#_Toc138193315)

[**CAPÍTULO 6 FLUJO DE TRABAJO: DISEÑO** 42](#_Toc138193316)

[**6.1 DISEÑO DE ARQUITECTURA** 43](#_Toc138193317)

[**6.1.1 Diagrama de Despliegue** 43](#_Toc138193318)

[**6.1.2 Diagrama Organizado en Capas** 44](#_Toc138193319)

[**6.2 DISEÑO DE DATOS** 45](#_Toc138193320)

[**6.2.1 DISEÑO DE DATOS LÓGICOS** 45](#_Toc138193321)

[**6.3 Diseño de caso de uso** 46](#_Toc138193322)

[**6.3.1 Diagrama de secuencia.** 46](#_Toc138193323)

[**CAPÍTULO 7 : Implementación** 52](#_Toc138193324)

[**7.1. Implementación  de arquitectura** 53](#_Toc138193325)

# **CAPITULO 1: FUNDAMENTOS TEÓRICOS**

## **Introducción**

## **Objetivo General**

## **Objetivos Específicos**

## **Alcance**

## **Marco Teórico**

# **CAPITULO 2: HERRAMIENTAS TECNOLÓGICAS PARA EL DESARROLLO**

**REACT**

**PYTHON (DJANGO)**

**FLUTTER**



Flutter es un framework que permite el [desarrollo de un proyecto de programación](https://talently.tech/blog/que-es-un-framework-en-programacion/). Es gratuito y de código abierto, y fue creado por Google en mayo de 2017.

Básicamente, permite crear una aplicación móvil nativa con una sola base de código. ¿Qué significa esto? Que puede usar un lenguaje de programación y una base de código para crear dos aplicaciones diferentes (para iOS y Android). Esta es, quizás, la principal ventaja de lo que es Flutter y lo que lo hace súper valioso.

**POSTGRE SQL**

PostgreSQL, también llamado Postgres, es un [sistema de gestión de bases de datos](https://es.wikipedia.org/wiki/Sistema_de_gesti%C3%B3n_de_bases_de_datos) [relacional](https://es.wikipedia.org/wiki/Modelo_relacional) [orientado a objetos](https://es.wikipedia.org/wiki/Base_de_datos_orientada_a_objetos) y de [código abierto](https://es.wikipedia.org/wiki/Software_de_c%C3%B3digo_abierto), publicado bajo la [licencia PostgreSQL](https://es.wikipedia.org/w/index.php?title=Licencia_PostgreSQL&action=edit&redlink=1), similar a la [BSD](https://es.wikipedia.org/wiki/Licencia_BSD) o la [MIT](https://es.wikipedia.org/wiki/Licencia_MIT).

Como muchos otros proyectos de [código abierto](https://es.wikipedia.org/wiki/Software_de_c%C3%B3digo_abierto), el desarrollo de PostgreSQL no es manejado por una empresa o persona, sino que es dirigido por una comunidad de desarrolladores que trabajan de forma desinteresada, [altruista](https://es.wikipedia.org/wiki/Altruismo), libre o apoyados por [organizaciones comerciales](https://es.wikipedia.org/wiki/Empresas). Dicha comunidad es denominada el [PGDG](https://es.wikipedia.org/w/index.php?title=PostgreSQL_Global_Development_Group&action=edit&redlink=1) (*PostgreSQL Global Development Group*).

PostgreSQL no tiene un gestor de errores (bugs), haciendo muy difícil conocer el estado de corrección de los mismos.

# **CAPITULO 3: TECNOLOGÍA PARA EL DESARROLLO DEL SOFTWARE**

## **3.1 ESTRATEGIA PARA EL DESARROLLO DEL SOFTWARE**

El proyecto se desarrollará en base a dos estrategias, el Proceso Unificado de Desarrollo de Software (PUDS) y el Lenguaje Unificado de Modelado (UML).

-El PUDS y UML son necesarios para la estrategia del desarrollo de software debido a que el lenguaje unificado de modelado es un requerimiento para el proceso unificado de desarrollo de software para representar las exigencias de sistemas en diagramas y generar una mejor comprensión de la utilidad del mismo.

-El PUDS es un conjunto de actividades necesarias para transformar los requisitos de un trabajo genérico que puede utilizarse para una gran variedad de sistemas software, como también para diferentes áreas de aplicación, diferentes tipos de organizaciones, diferentes niveles de aptitud y diferentes tamaños de proyectos.

Así también es necesario para garantizar la calidad en el resultado final, gracias a que el proceso tiene una variedad de aplicación y utilidad en diferentes tipos de organizaciones. Además, este proceso integra el control necesario para mantener en orden el proceso de desarrollo.

## **3.2 METODOLOGÍA PARA EL DESARROLLO DEL SOFTWARE**

### **3.2.1 CARACTERÍSTICAS DEL PUDS**

El Proceso Unificado es un proceso de desarrollo de software: “conjunto de actividades necesarias para transformar los requisitos del usuario en un sistema software”.

Es un proceso orientado a objetos dirigido por casos de uso, centrado en la arquitectura con un ciclo de vida iterativo e incremental.

Está basado en el lenguaje unificado modelado (UML), este proceso puede organizarse en cuatro fases: inicio, elaboración, construcción y transición.

Las fases se dividen en sus conjuntos de iteraciones, en las que se desarrollan los flujos de trabajo: requerimientos, análisis, diseño, complementación y pruebas.

**Características:**

* **Dirigido por Casos de Uso:**

-El Proceso Unificado es un proceso de desarrollo de software: “conjunto de actividades necesarias para transformar los requisitos del usuario en un sistema software”.

Los casos de uso también guían el proceso de desarrollo (diseño, implementación, y prueba).

-Basándose en los casos de uso los desarrolladores crean una serie de modelos de diseño e implementación que llevan a cabo los casos de uso. De este modo los casos de uso no solo inician el proceso de desarrollo sino que le proporcionan un hilo conductor, avanza a través de una serie de flujos de trabajo que parten de los casos de uso.

-La idea es que cada iteración tome un conjunto de casos de uso o escenarios y desarrolle todo el camino a través de las distintas disciplinas. ·

* **Centrado en la Arquitectura:**

-La arquitectura de un sistema software se describe mediante diferentes vistas del sistema en construcción.

-Los casos de uso deben encajar en la arquitectura, y a su vez la arquitectura debe permitir el desarrollo de todos los casos de uso requeridos, actualmente y a futuro.

-A medida que los casos de uso se especifican y maduran, se descubre más de la arquitectura, y esto a su vez lleva a la maduración de más casos de uso. Este proceso continúa hasta que se considere que la arquitectura es estable

Se necesita una arquitectura para:

* Comprender el sistema
* Organizar el desarrollo
* Fomentar la reutilización
* Hacer evolucionar el sistema

● **Iterativo e Incremental:**

-El Proceso Unificado es un marco de desarrollo iterativo e incremental compuesto de cuatro fases denominadas El Ciclo de Vida del Proceso Unificado, estas fases son: Inicio, Elaboración, Construcción y Transición

-Cada una de estas fases es a su vez dividida en una serie de iteraciones (la de inicio puede incluir varias iteraciones en proyectos grandes).En cada iteración se desarrolla en secuencia un conjunto de disciplinas o flujos de trabajos.

-Estas iteraciones ofrecen como resultado un incremento del producto desarrollado que añade o mejora las funcionalidades del sistema en desarrollo.

-Los desarrolladores basan la selección de lo que implementarán en cada iteración en dos cosas: el conjunto de casos de uso que amplían la funcionalidad, y en los riesgos más importantes que deben mitigarse

-En cada iteración los desarrolladores identifican y especifican los casos de uso relevantes, crean un diseño utilizando la arquitectura seleccionada como guía, para implementar dichos casos de uso. Si la iteración cumple sus objetivos, se continúa con la próxima. Sino deben revisarse las decisiones previas y probar un nuevo enfoque.

**Fases del PUDS:**

**Inicio. -** El objetivo de esta fase es ayudar al equipo de proyecto a decidir cuáles son los verdaderos objetivos del proyecto. Las iteraciones exploran diferentes soluciones posibles, y diferentes arquitecturas posibles. Puede que todo el trabajo físico realizado en esta fase sea descartado. Lo único que normalmente sobrevive a la fase de inicio es el incremento del conocimiento en el equipo.

**Elaboración. -** Aquí se especifican en detalle la mayoría de los casos de uso del producto y se diseña la arquitectura del sistema.

**Construcción. -** Durante la fase de construcción se crea el producto. La línea base de la arquitectura crece hasta convertirse en el sistema completo. Al final de esta fase, el producto contiene todos los casos de uso implementados, sin embargo, puede que no esté libre de defectos.

**Transición. –** Se accede a lo que se conoce como la fase Beta, el programa está funcional pero constantemente siendo actualizado para mejoras, corrección de errores y actualizaciones.

### **3.2.2 CARACTERÍSTICAS DE UML**

-El Lenguaje Unificado de Modelado (UML), es un lenguaje de modelado visual de propósito general que se utiliza para especificar, visualizar, construir y documentar los artefactos de un sistema software.

-UML permite describir un sistema en diferentes niveles de abstracción, simplificando la complejidad sin perder información, para que tanto usuarios, líderes y desarrolladores puedan comprender claramente las características de la aplicación

-Está pensado para ser utilizado con todos los métodos de desarrollo, etapas del ciclo de vida, dominios de aplicación y medios.

-Pueden ser utilizados para generar código y casos de prueba, esto exige un perfil UML adecuado, el uso de herramientas que se correspondan con la plataforma destino y elegir entre varios componentes de implementación.

-UML necesita ser lo suficientemente expresivo para manejar todos los conceptos que surgen en un sistema moderno, como la concurrencia y la distribución.

-Divide cada proyecto en un número de diagramas que representan las diferentes vistas del proyecto. Estos diagramas juntos son los que representan la arquitectura del proyecto.

-UML presenta una variedad de tipos de diagrama, donde las 2 principales son:

-**Estructurales**: Diagramas de clase, componentes, despliegue, objetos, paquetes, perfiles y estructura compuesta.

-**De comportamiento**: Diagrama de actividades, casos de uso, máquina de estados e interacción.

● **Diagramas UML**

Los diagramas de Clases de UML se pueden usar para modelar la base de pueden usar para modelar la base de datos relacional en la que un sistema está basado, sin embargo, los esté basado, sin embargo, los diagramas tradicionales de modelado capturan más información capturan más información sobre la base de datos relacional y son más adecuados para modelarla.

**Diagrama de Clases:** Modela la estructura estática de las clases en el sistema.

**Diagrama de Componentes:** Modela los componentes que componen una aplicación, sistema o empresa.

**Diagrama de Estructura de Composición:** Representa la estructura interna de un clasificador.

**Diagrama de Despliegue Físico:** Muestra cómo y dónde se despliega el sistema.

**Diagrama de Objetos:** Modela la estructura estática de los objetos en el sistema.

**Diagrama de Actividades:** Modela el comportamiento de los casos de uso, objetos y operaciones.

**Diagrama de Comunicaciones:** Modela interacciones entre objetos

**Diagrama de Secuencias:** Representa una interacción, poniendo el foco en la secuencia de los mensajes que se intercambian.

**Diagrama de Máquinas de Estado:** Ilustra como un elemento se puede mover entre estados que clasifican su comportamiento.

**Diagrama de Tiempos:** Muestra los cambios en el estado o la condición de una línea de vida a lo largo del tiempo lineal.

**Diagrama de Casos de Uso:** Muestra las relaciones entre los actores y el sujeto (sistema) y los casos de uso.

# **CAPITULO 4: FLUJO DE TRABAJO: CAPTURA DE REQUISITOS**

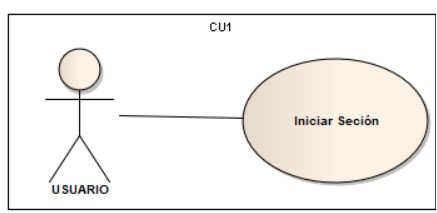
## **4.1 Actores**

## **4.2 Casos de Uso**

## **4.3 Priorizar Caso de Uso**

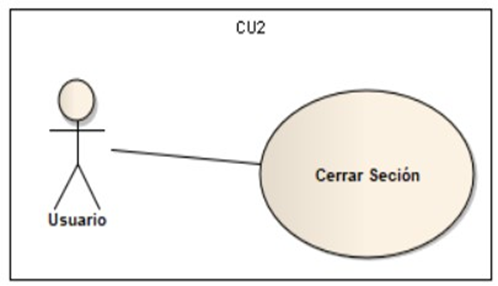
|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **NRO** | **CASO DE USO** | **ESTADO** | **PRIORIDAD** | **RIESGO** | **ACTOR** |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

## **4.4 Detalle de Caso de uso**

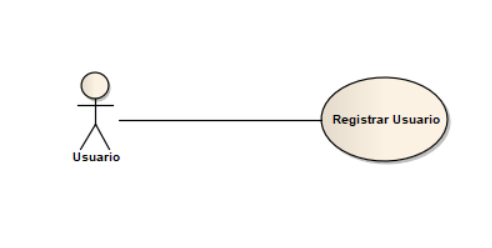
CU1:Iniciar sesión

|  |  |
| --- | --- |
| **NOMBRE** | INICIAR SESIÓN |
| **PROPÓSITO** | Iniciar sesión de un usuario |
| **FLUJO DE SUCESOS** | 1. **Ingresar a la app**     1. Ingresar el nombre de usuario y contraseña en los campos donde se requiere.    2. Dar click en el botón “Iniciar Sesión”, para ingresar a la app.    3. Se verificará si los datos coinciden con los registrados.    4. Si los datos ingresados son válidos, ingresa a la app caso contrario, se retendrá en el inicio de sesión. |
| **ACTORES** |  |
| **INICIADOR** | Usuario |
| **PRECONDICIÓN** | Ninguna |
| **POSTCONDICIÓN** | Ninguna |
| **EXCEPCIONES** | Datos ingresados erróneamente deberán volverse a ingresar los datos que sean correctos |

CU2:Cerrar sesión



|  |  |
| --- | --- |
| **NOMBRE** | CERRAR SESIÓN |
| **PROPÓSITO** | Cerrar sesión de un usuario |
| **FLUJO DE SUCESOS** | 1. **Salir de la app**    1. Seleccionar el botón de opciones de perfil (icono Imagen del usuario).    2. Seleccionar opción de cerrar sesión.    3. Confirmar cerrar sesión    4. Se redirige al punto de iniciar sesión. |
| **ACTORES** |  |
| **INICIADOR** | USUARIO |
| **PRECONDICIÓN** | Ninguna |
| **POSTCONDICIÓN** | Ninguna |
| **EXCEPCIONES** | Que no haya iniciado sesión |

CU3: Registrar Usuario 

|  |  |
| --- | --- |
| **NOMBRE** | REGISTRAR USUARIOS |
| **PROPÓSITO** | Que los estudiantes de registren para poder utilizar la app ya sea como choferes o usuarios |
| **FLUJO DE SUCESOS** | **1.REGISTRARSE**  1.1 Introducir datos del  1.2 Registrar datos del |
| **ACTORES** | Usuario |
| **INICIADOR** | Usuario |
| **PRECONDICIÓN** |  |
| **POST-CONDICIÓN** | Ninguna |
| **EXCEPCIONES** | Datos ingresados ya existen, datos erróneos al registrar |

CU4:

|  |  |
| --- | --- |
| **NOMBRE** |  |
| **PROPÓSITO** |  |
| **FLUJO DE SUCESOS** |  |
| **ACTORES** |  |
| **INICIADOR** |  |
| **PRECONDICIÓN** |  |
| **POST-CONDICIÓN** | Ninguna |
| **EXCEPCIONES** | Datos ingresados ya existen, datos erróneos al registrar |

CU5:

|  |  |
| --- | --- |
| **NOMBRE** |  |
| **PROPÓSITO** |  |
| **FLUJO DE SUCESOS** |  |
| **ACTORES** |  |
| **INICIADOR** | Usuario |
| **PRECONDICIÓN** | Ninguna |
| **POST-CONDICIÓN** | Ninguna |
| **EXCEPCIONES** |  |

CU6:

|  |  |
| --- | --- |
| **NOMBRE** |  |
| **PROPÓSITO** |  |
| **FLUJO DE SUCESOS** |  |
| **ACTORES** |  |
| **INICIADOR** | Usuario |
| **PRECONDICIÓN** | Ninguna |
| **POST-CONDICIÓN** | Ninguna |
| **EXCEPCIONES** | Ninguna |

CU7:

|  |  |
| --- | --- |
| **NOMBRE** |  |
| **PROPÓSITO** |  |
| **FLUJO DE SUCESOS** |  |
| **ACTORES** |  |
| **INICIADOR** | Usuario |
| **PRECONDICIÓN** | Ninguno |
| **POST-CONDICIÓN** | Ninguna |
| **EXCEPCIONES** | Ninguna |

CU8:

|  |  |
| --- | --- |
| **NOMBRE** |  |
| **PROPÓSITO** |  |
| **FLUJO DE SUCESOS** |  |
| **ACTORES** |  |
| **INICIADOR** |  |
| **PRECONDICIÓN** | Ninguno |
| **POST-CONDICIÓN** | Ninguna |
| **EXCEPCIONES** | Ninguna |

CU9:

|  |  |
| --- | --- |
| **NOMBRE** |  |
| **PROPÓSITO** |  |
| **FLUJO DE SUCESOS** |  |
| **ACTORES** |  |
| **INICIADOR** |  |
| **PRECONDICIÓN** |  |
| **POST-CONDICIÓN** | Ninguna |
| **EXCEPCIONES** |  |

CU10:

|  |  |
| --- | --- |
| **NOMBRE** |  |
| **PROPÓSITO** |  |
| **FLUJO DE SUCESOS** |  |
| **ACTORES** |  |
| **INICIADOR** | Usuario |
| **PRECONDICIÓN** | Ninguno |
| **POST-CONDICIÓN** | Ninguna |
| **EXCEPCIONES** | Ninguna |

CU11: Validar Usuario

ITERACION 1:

CU1: Iniciar sesión

CU2: Cerrar sesión

CU3: Registrar Usuario

CU4: Registrar Vehículo

ITERACION 2:

CU5: Ver rutas

CU6: Solicitudes de viaje

CU7: Confirmación de viaje

ITERACION 3:

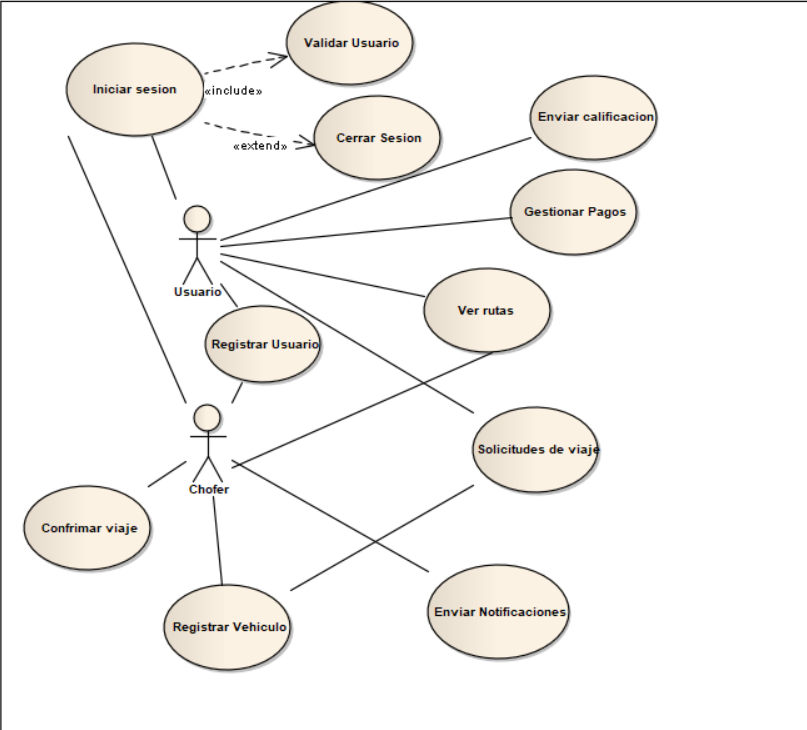
CU8: Enviar calificación

CU9: Gestionar Pago

CU10: Enviar notificaciones

CU11: Validar Usuario

## **4.5 Estructurar Modelo de Caso de Uso**



# **CAPÍTULO 5: FLUJO DE TRABAJO: ANÁLISIS**

## **5.1 ANÁLISIS DE LA ARQUITECTURA**

### **5.1.1 IDENTIFICAR PAQUETE**

### **5.1.2 RELACIÓN PAQUETE – CASO DE USO**

**P1**

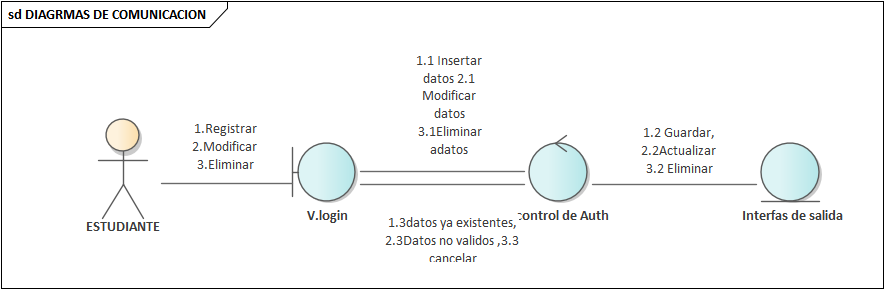
**P2**

**P3**

**P4**

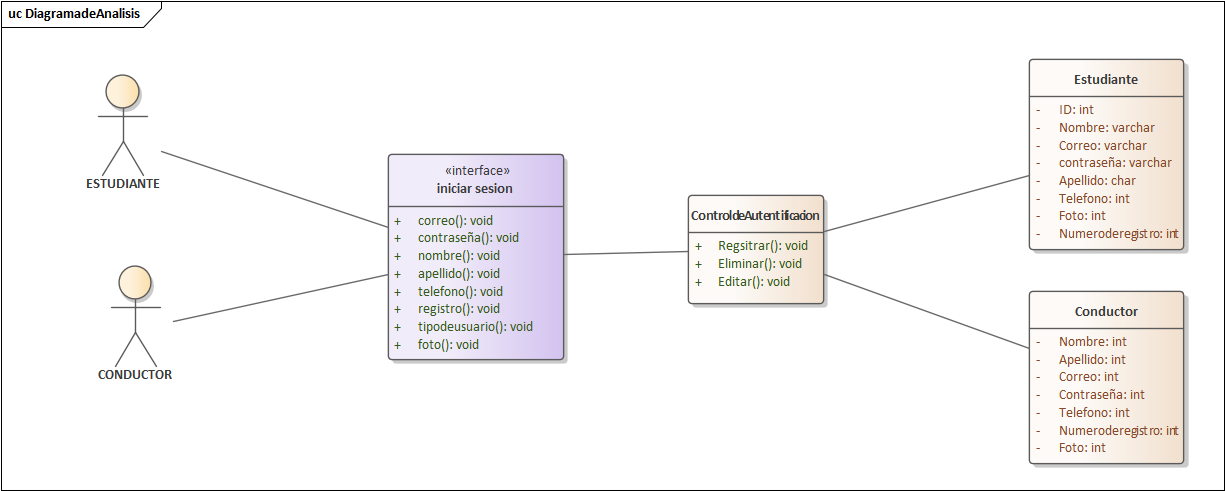
## **5.2 Análisis de Casos de Uso**

**CU3 REGISTRAR USUARIO**

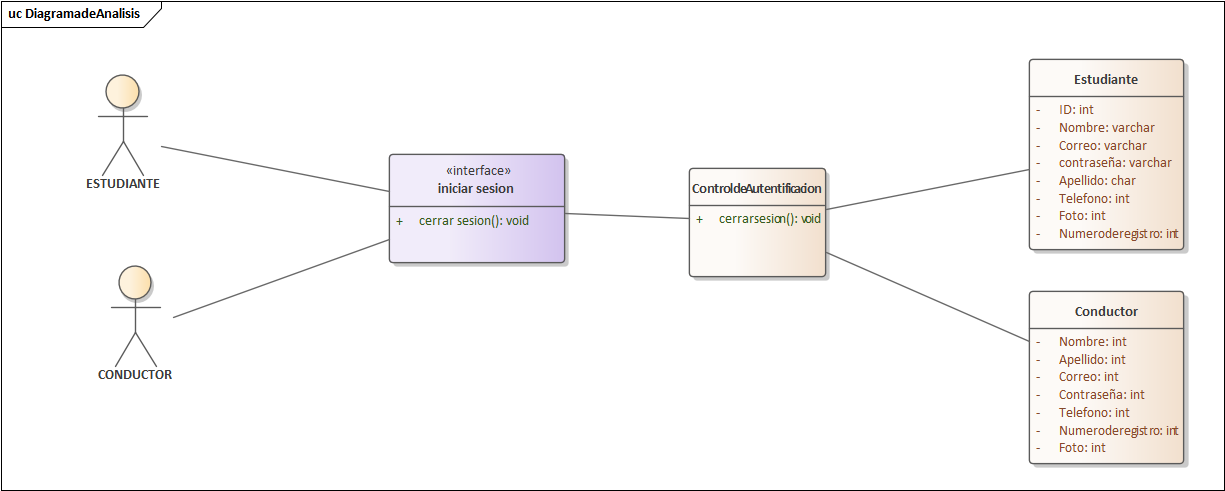
****

## **5. 3 ANÁLISIS DE CLASES**

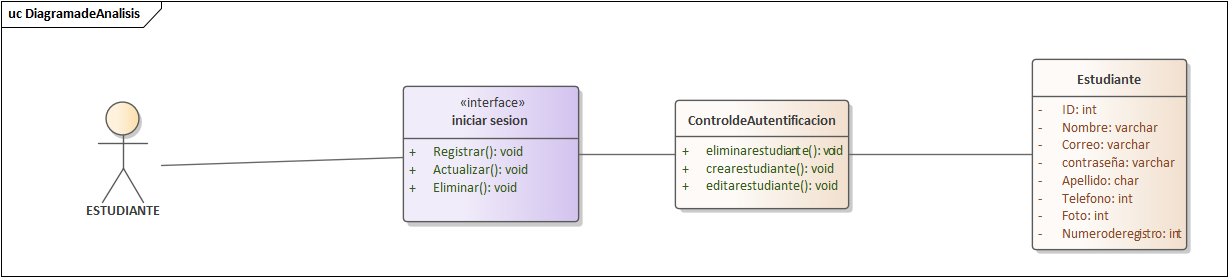
CU1 Iniciar sesión



CU2 Cerrar sesión



CU3 Gestionar usuario



CU4

CU5

CU6

CU7

CU8

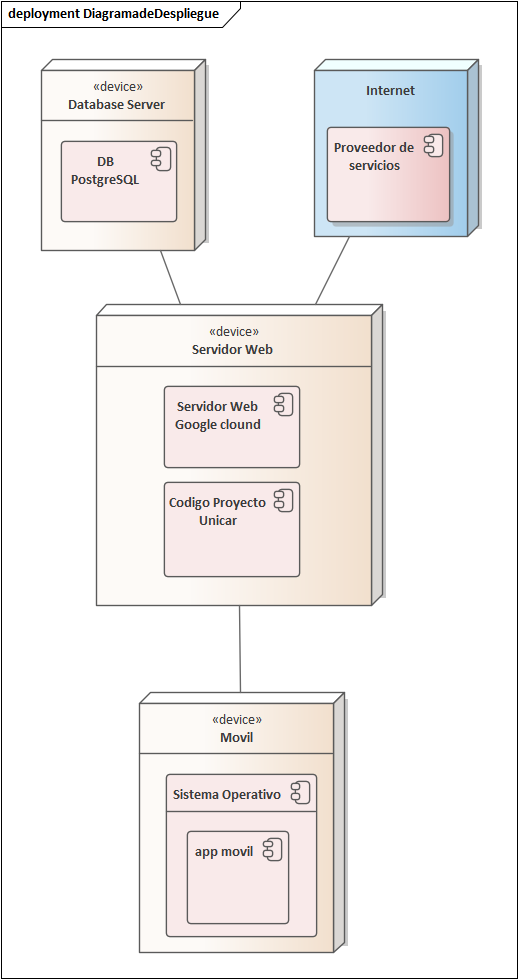
CU9

CU10

# **CAPÍTULO 6 FLUJO DE TRABAJO: DISEÑO**

## **6.1 DISEÑO DE ARQUITECTURA**

### **6.1.1 Diagrama de Despliegue**



### **6.1.2 Diagrama Organizado en Capas**

## **6.2 DISEÑO DE DATOS**

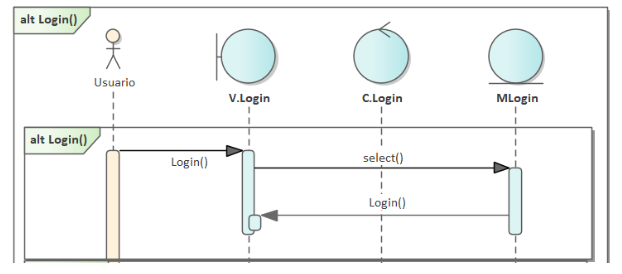
### **6.2.1 DISEÑO DE DATOS LÓGICOS**

#### **6.2.1.1 DIAGRAMA DE CLASES**

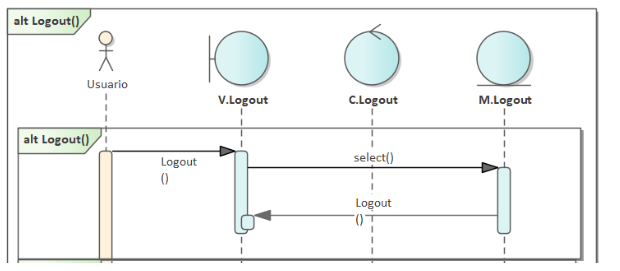
## **6.3 Diseño de casos de uso**

### **6.3.1 Diagrama de secuencia.**

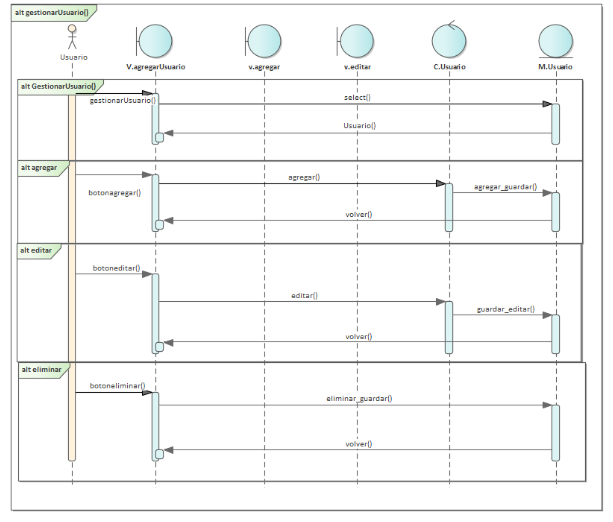
CU1 Iniciar sesión



CU2 Cerrar sesión



CU3 Gestionar usuario



CU4

CU6

CU11

CU10

**CAPÍTULO 7: Implementación**

**7.1. Implementación de arquitectura**