PLAN DE TRABAJO DEL ESTUDIANTE

SENATI

TRABAJO FINAL DEL CURSO

1. INFORMACIÓN GENERAL

Apellidos y Nombres:	CCACCA APAZA JHON YEFERSON	ID: 001461162
Dirección Zonal/CFP:	AREQUIPA – PUNO / UCP JULIACA ETI	
Carrera:	INGENIERIA DE SOFTWARE CON INTELIGENCIA ARTIFICIAL	Semestre: V
Curso/ Mód. Formativo	INGENIERIA DEL SOFTWARE	
Tema del Trabajo:	Realizar un análisis y propuesta de desarrollo de software para una futuraimplementación.	

2. PLANIFICACIÓN DEL TRABAJO

N°	ACTIVIDADES/ ENTREGABLES	CRONOGRAMA/ FECHA DE ENTREGA						
1	Investigación del trabajo final del curso	21/052024						
2	Respondemos las preguntas propuestas en el documento	21/05/2024						
3	Realizamos el diagrama de contexto	21/05/2024						
4	Análisis de requerimientos (Divididos entre Funcionales y No Funcionales)	21/05/2024						
5	Cronograma de actividades(actividades y carta gannt)		28/05/2024					
6	Realizamos los diagramas		28/05/2024					
7	Entrega el entregable 1		29/05/2024					
8	Avance de la segunda entrega del tr			31/05/2024				
9	Segunda Entrega Diseño ✓ Detalles de los casos de uso ✓ Diseño de la interfaz			31/05/2024				
10	Implementación ✓ Implementación de las clases ✓ Modelado de la Bases de datos			31/05/2024				
11	Entrega del entregable 2							
12								

13								

3. PREGUNTAS GUIA

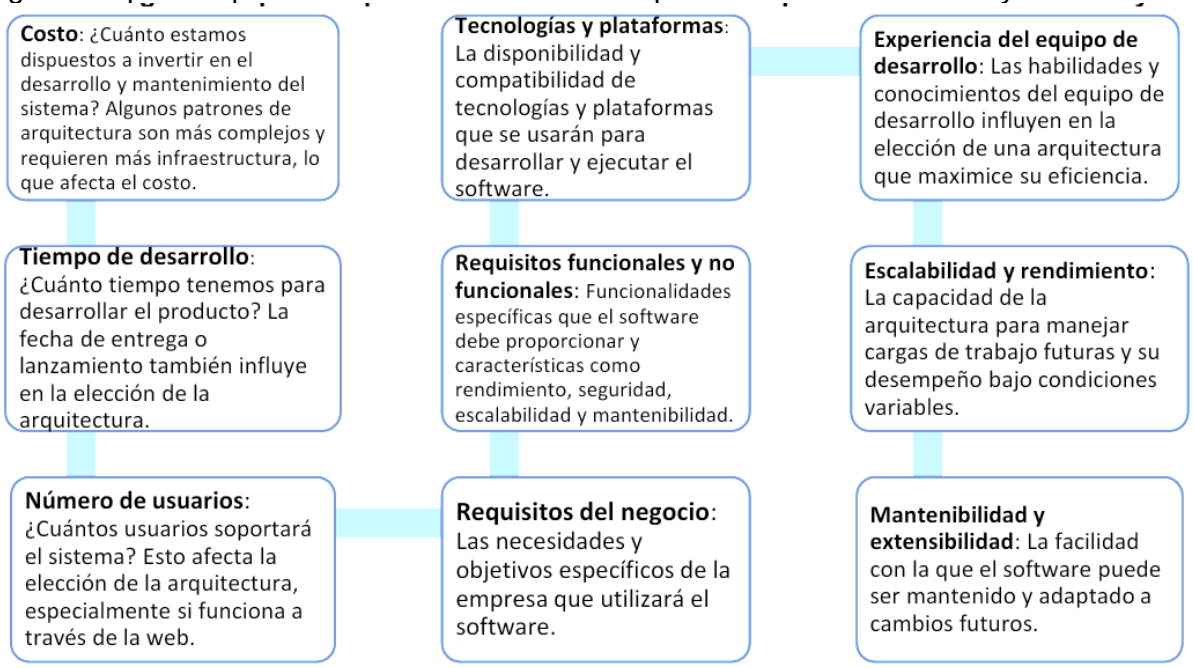
Durante la investigación de estudio, debes obtener las respuestas a las siguientes interrogantes:

Nº	PREGUNTAS
1	¿Que determina el uso de una arquitectura de software?
2	¿Que son los Diagramas de casos de uso?
3	¿Que son los Diagramas de clases?
4	¿Que son los Diagramas de secuencia?

HOJA DE RESPUESTAS A LAS PREGUNTAS GUÍA

1.	¿Que determina el uso de una arquitectura de software?

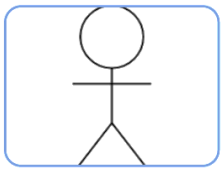
La arquitectura de software se refiere al diseño y planificación a un nivel superior de una estructura a un nivel abstracto antes de pasar a su realización. En el contexto del desarrollo de software, la arquitectura se basa en modelos, patrones y abstracciones teóricas. Sirve como una guía teórica detallada para entender cómo encajarán las piezas de un producto o servicio. Algunos aspectos que determinan el uso de una arquitectura de software incluyen:



Estos nueve aspectos deben ser considerados en conjunto para seleccionar la arquitectura de software más adecuada para un proyecto específico.

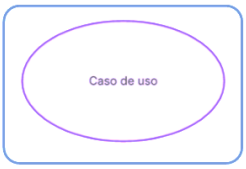
2.	¿Que son los Diagramas de casos de uso?

Los diagramas de casos de uso son una herramienta visual utilizada en el análisis y diseño de sistemas de software para representar las interacciones entre los actores externos y el sistema en sí mismo. En términos simples, muestran cómo los usuarios interactúan con un sistema y qué funcionalidades ofrece el sistema en respuesta a esas interacciones. Estos diagramas son especialmente útiles en las primeras etapas del desarrollo de software, ya que ayudan a comprender los requisitos del sistema desde la perspectiva del usuario. Algunos puntos clave sobre los diagramas de casos de uso son:



Actores

- Representan a los usuarios o entidades externas que interactúan con el sistema. Los actores pueden ser personas, otros sistemas informáticos o dispositivos externos.



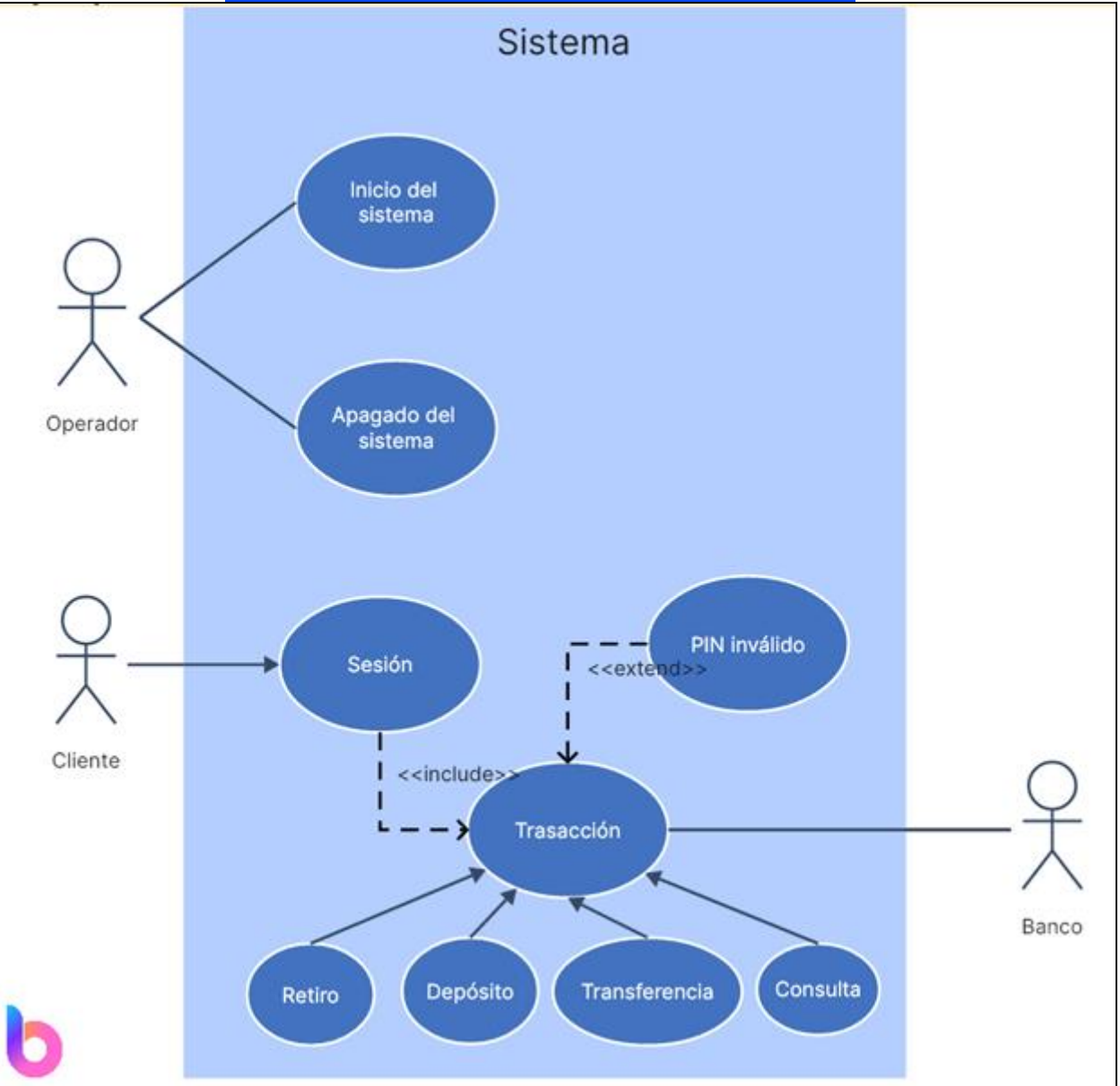
Casos de uso:

- Representan las diferentes formas en que los actores interactúan con el sistema para lograr ciertos objetivos. Cada caso de uso describe una función o característica específica del sistema desde la perspectiva del usuario.

Relaciones

1. Las relaciones de inclusión y extensión en los diagramas de casos de uso son dos tipos de conexiones que representan diferentes tipos de interacciones entre los casos de uso. La relación de inclusión indica que un caso de uso está compuesto por otro caso de uso más pequeño y reutilizable, mientras que la relación de extensión indica que un caso de uso opcional puede ampliar la funcionalidad de otro caso de uso en ciertos puntos.

A continuación, un ejemplo:



El siguiente ejemplo muestra un sistema de cajero automático. En este caso los actores son tres, que son el operador, el cliente y el banco. El operador tiene solo dos casos de uso que son el “inicio del sistema” y el “apagado del sistema”. El banco es el actor cuyo principal caso de uso es la “transacción”. Como puede ver, el caso de uso de transacción incluye varios otros casos de uso. Estos son el “retiro”, el “depósito”, la “transferencia” y la “consulta”. El cliente también tiene el caso de uso de "transacción", con todas las demás inclusiones. Además de esto, también existe una extensión de caso de uso para el “PIN inválido”.

3.	¿Que son los Diagramas de clases?

Los diagramas de clases son una herramienta visual utilizada en el diseño y modelado de sistemas orientados a objetos en el desarrollo de software. En esencia, representan la
estructura estática de un sistema, mostrando las clases del sistema, sus atributos, métodos y las relaciones entre ellas. Algunos puntos clave sobre los diagramas de clases son:

- ✓ Clases: Representan los elementos fundamentales del sistema, describiendo los objetos que serán creados en el sistema. Cada clase tiene

atributos que representan características de los objetos y métodos que representan acciones que los objetos pueden realizar.

- ✓ Atributos: Son las propiedades o características de una clase que describen el estado de los objetos de esa clase. Por ejemplo, una clase "Persona" puede tener atributos como "nombre", "edad" y "género".

- ✓ Métodos: Son las acciones o comportamientos que pueden realizar los objetos de una clase. Los métodos representan las operaciones que los objetos pueden realizar, como "calcular salario" en una clase "Empleado".

- ✓ Relaciones: Representan las asociaciones entre las clases en el sistema. Las relaciones pueden ser de diferentes tipos, como asociación, composición, agregación, herencia, entre otras, y ayudan a definir cómo las clases interactúan entre sí.

- ✓ Asociaciones: Representan las conexiones entre dos clases y muestran que los objetos de una clase están relacionados con los objetos de otra clase. Por ejemplo,
  - ➢ una asociación entre las clases "Estudiante" y "Curso" puede indicar que un estudiante está inscrito en uno o varios cursos.

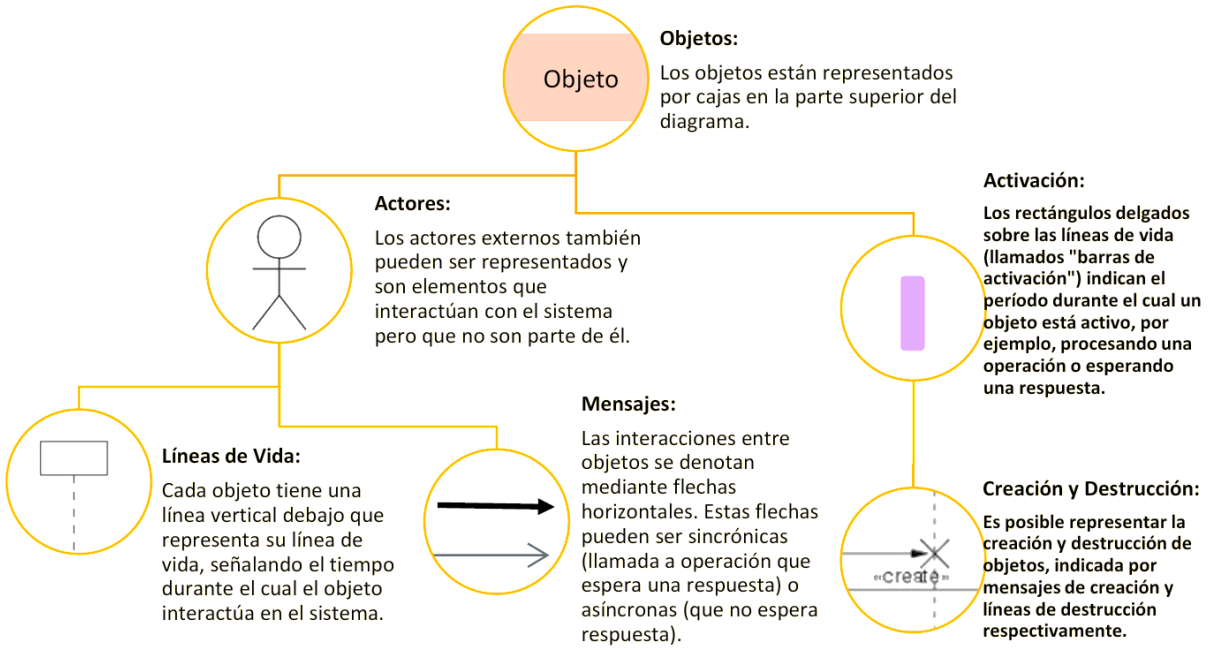
- ✓ Herencia: Es una relación en la que una clase (subclase) hereda atributos y métodos de otra clase (superclase). La subclase puede agregar nuevos atributos y métodos o modificar los existentes.

- ✓ Encapsulamiento, Abstracción y Modularidad: Los diagramas de clases promueven estos principios fundamentales de la programación orientada a objetos. El encapsulamiento se refiere a ocultar la implementación interna de una clase, la abstracción implica mostrar solo la información esencial de una clase y la modularidad se refiere a dividir el sistema en componentes independientes y reutilizables.

4.	¿Que son los Diagramas de secuencia?

Los diagramas de secuencia son una herramienta importante dentro de la ingeniería de software, utilizada principalmente para modelar la interacción entre objetos en un sistema de acuerdo con un tiempo secuencial específico. Estos diagramas son parte de la familia de diagramas de comportamiento utilizados en el modelado de sistemas orientados a objetos y proporcionan una
visualización clara de cómo los objetos interactúan y en qué secuencia lo hacen, lo que es fundamental para entender el flujo lógico de la aplicación. Aquí detallo sus características y usos principales:

Características Principales de los Diagramas de Secuencia



Usos de los Diagramas de Secuencia

- Análisis de Requerimientos:
  Ayudan a comprender los requisitos funcionales del sistema mostrando cómo se supone que los objetos interactúan.
- Diseño de Software:
  Facilitan la visualización del flujo de operaciones, ayudando en el diseño de la lógica del sistema.
- Depuración y Optimización:
  Permiten identificar redundancias o posibles puntos de mejora en las interacciones entre los objetos.
- Documentación:
  Ofrecen una manera clara y metódica para documentar los procesos del sistema, facilitando la comprensión y el mantenimiento del código por otros desarrolladores y stakeholders.

HOJA DE PLANIFICACIÓN

PROCESO DE EJECUCIÓN

OPERACIONES / PASOS /SUBPASOS	SEGURIDAD / MEDIO AMBIENTE / NORMAS -ESTANDARES

INSTRUCCIONES: debes ser lo más explícito posible. Los gráficos ayudan a transmitir mejor las ideas. No olvides los aspectos de calidad, medio ambiente y SHI.

TEMA: Estudio de Desarrollo de Software

OBJETIVO DEL TRABAJO

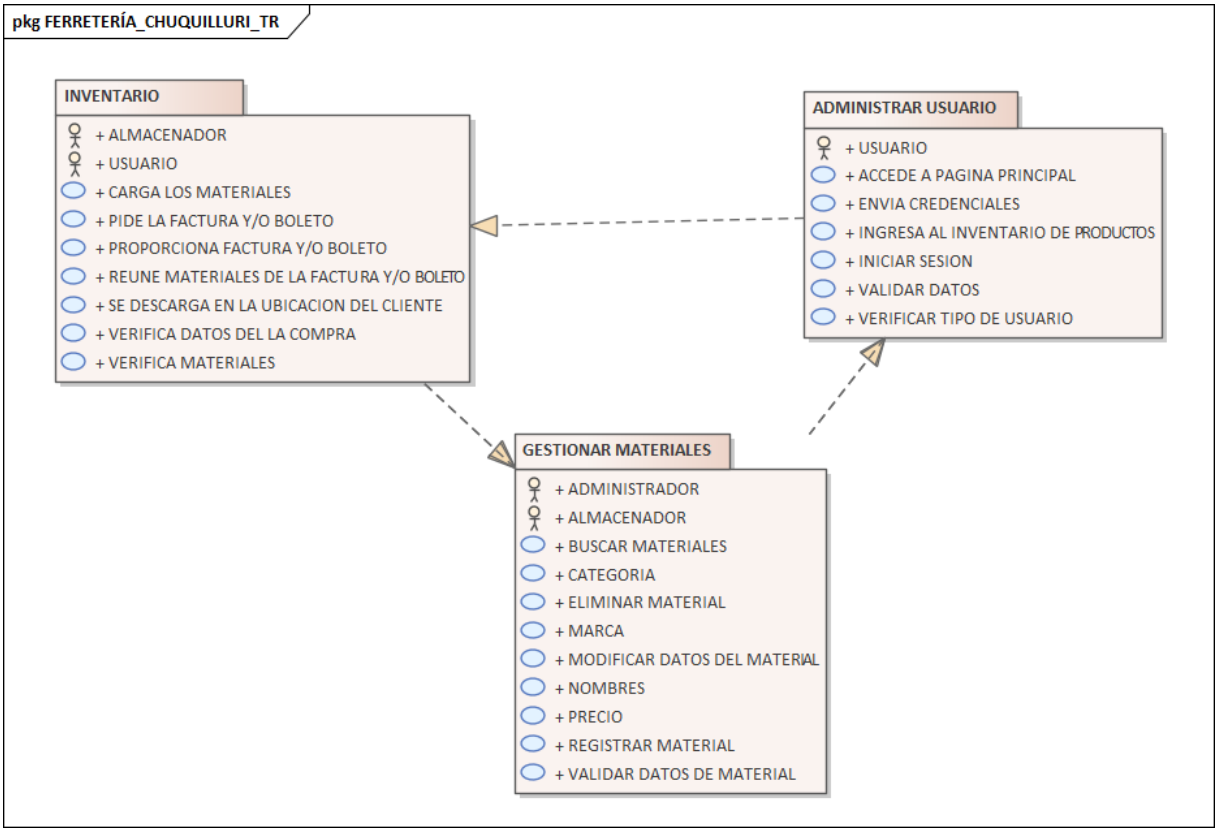
Realizar un análisis y propuesta de desarrollo de software para una futura implementación.

PLANTEAMIENTO DEL TRABAJO

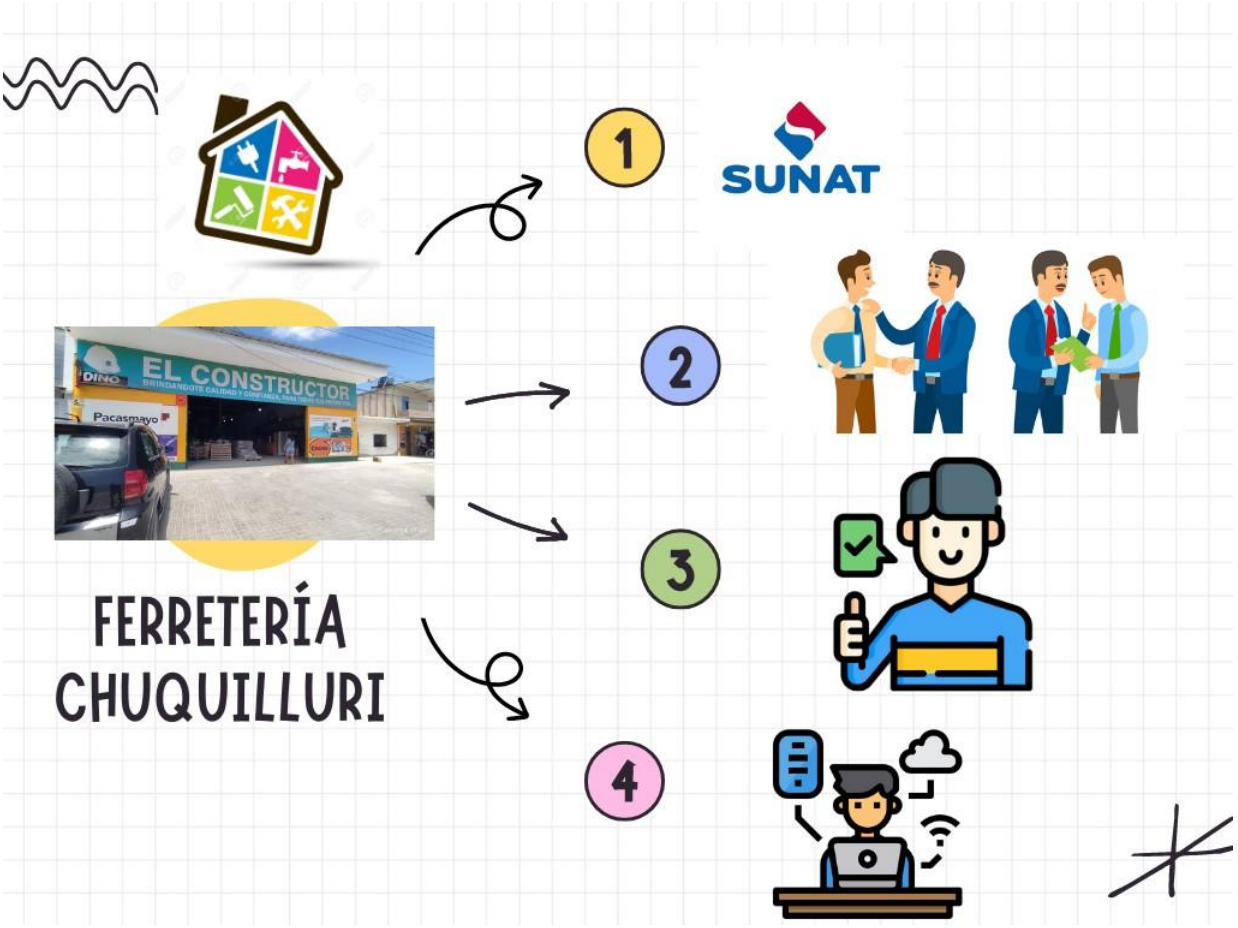
Ferretería CHUQUILLURI es una empresa dedicada a la comercialización de productos para la construcción, tiene un gran problema que sus procesos de ventas/inventario no están sistematizados y por ende su facturación lo llevan manualmente.

Se pide realizar un estudio de desarrollo de software para el caso propuesto bajo el siguiente esquema de trabajo.

Primera Entrega (Que se debe usar 3 paquetes)



1. Objetivo general (Mas Diagrama de Contexto)



2. Análisis de requerimientos(Divididos entre Funcionales y No Funcionales)

Requisitos Funcionales

RF01:	El sistema debe permitir la gestión de ventas, incluyendo la creación de facturas, el registro de ventas y la actualización automática del inventario.
Prioridad:	Alta
Nivel de Necesidad:	No Negociable
Estabilidad en el tiempo de desarrollo:	Estable

RF02:	El sistema debe permitir la gestión de inventarios, incluyendo el registro de productos, la actualización de stock, y la generación de reportes de inventario.
Prioridad:	Alta
Nivel de Necesidad:	No Negociable
Estabilidad en el tiempo de desarrollo:	Estable

No Funcionales

RNF01:	Los dispositivos y servidores deben operar de manera continua sin fallas.
Prioridad:	Alta
Nivel de Necesidad:	negociable
Estabilidad en el tiempo de desarrollo:	**Estable**

RNF02:	El sistema debe asegurar la protección de datos sensibles contra accesos no autorizados y pérdidas.
Prioridad:	Alta
Nivel de Necesidad:	No negociable
Estabilidad en el tiempo de desarrollo:	**estable**

RF0:	El sistema debe permitir la gestión de ventas, incluyendo la creación de facturas, el registro de ventas y la actualización automática del inventario.
Prioridad:	Alta
Nivel de Necesidad:	No Negociable
Estabilidad en el tiempo de desarrollo:	Estable

RF0:	El sistema debe permitir la gestión de inventarios, incluyendo el registro de productos, la actualización de stock, y la generación de reportes de inventario.
Prioridad:	Alta
Nivel de Necesidad:	No Negociable
Estabilidad en el tiempo de desarrollo:	Estable

a. Hardware

RF0:	Los dispositivos y servidores deben operar de manera continua sin fallas.
Prioridad:	Alta
Nivel de Necesidad:	No Negociable
Estabilidad en el tiempo de desarrollo:	Estable

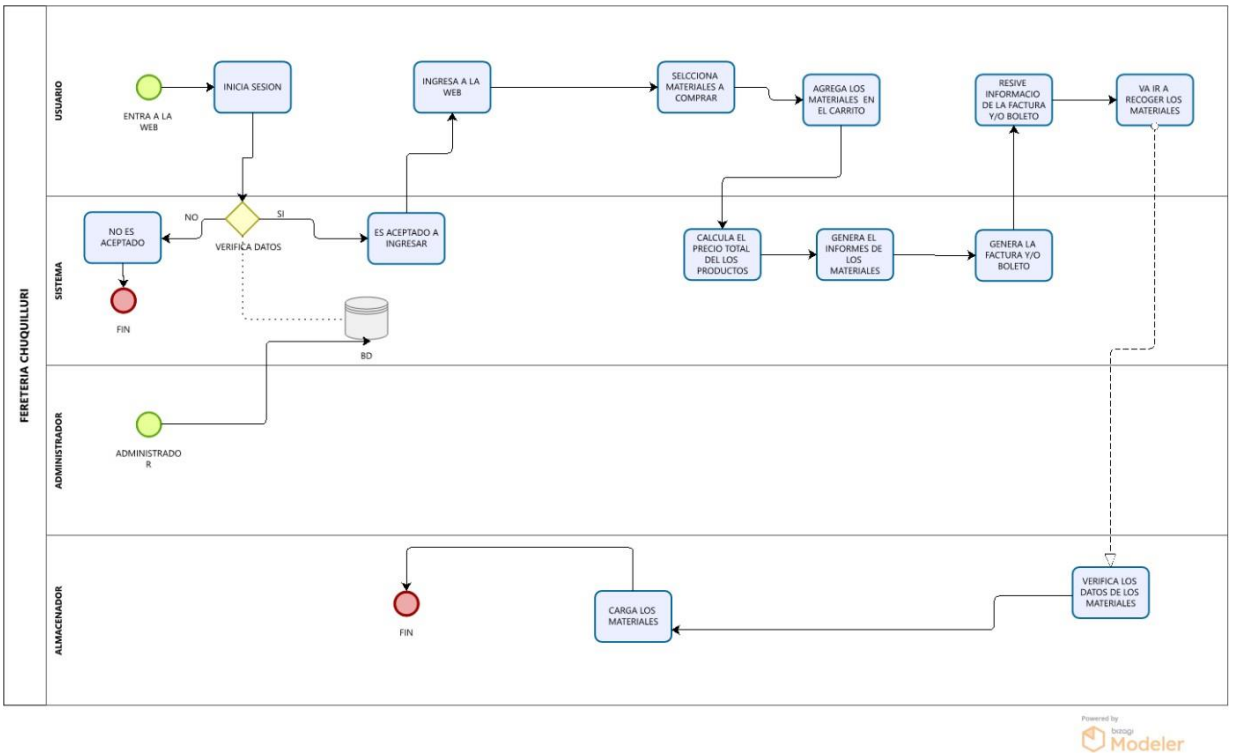
RF0:	El sistema debe asegurar la protección de datos sensibles contra accesos no autorizados y pérdidas.
Prioridad:	Alta
Nivel de Necesidad:	No Negociable
Estabilidad en el tiempo de desarrollo:	Estable

b. Software

RF0:	Los dispositivos y servidores deben operar de manera continua sin fallas.
Prioridad:	Alta
Nivel de Necesidad:	No Negociable
Estabilidad en el tiempo de desarrollo:	Estable

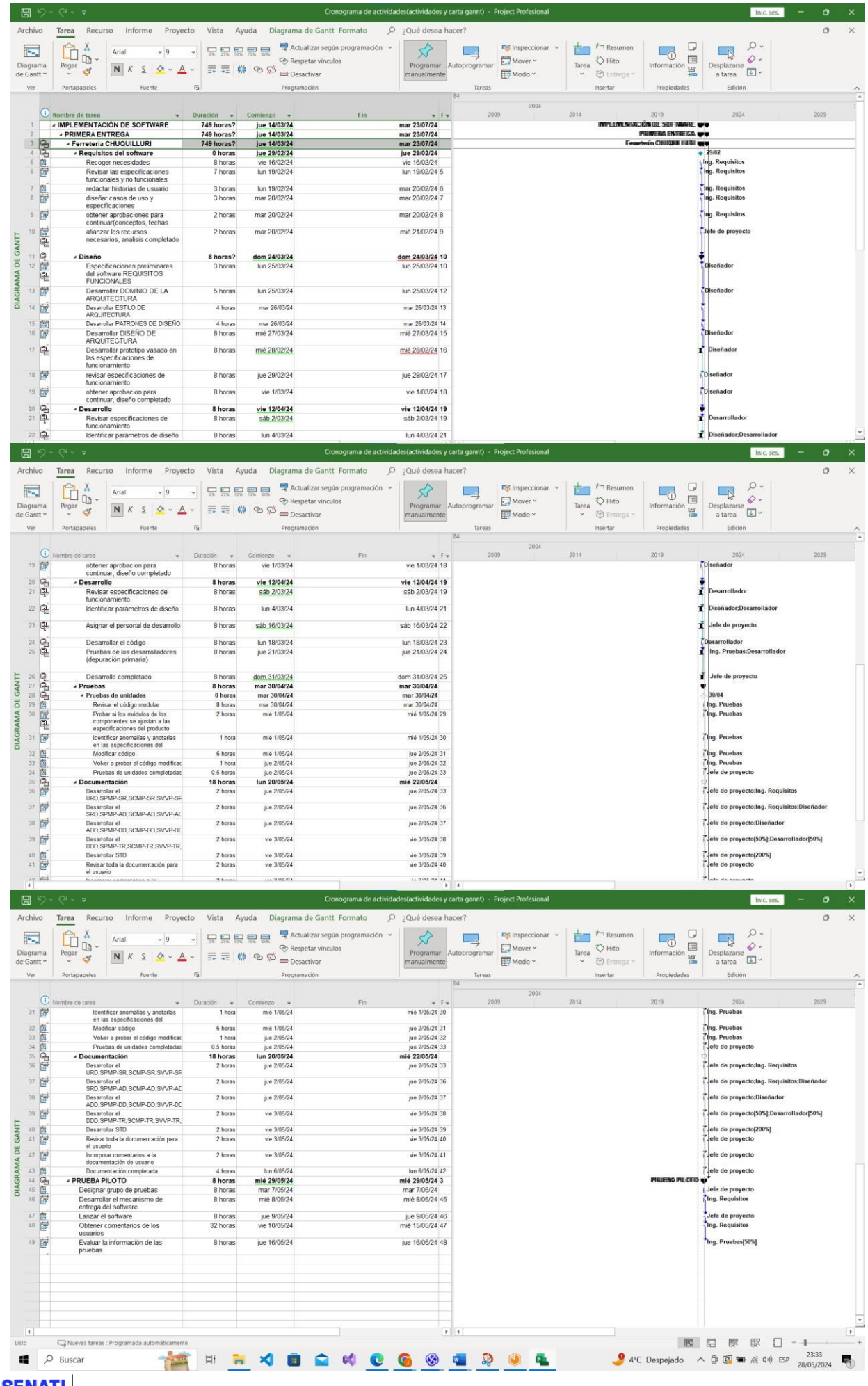
RF0:	El sistema debe asegurar la protección de datos sensibles contra accesos no autorizados y pérdidas.
Prioridad:	Alta
Nivel de Necesidad:	No Negociable
Estabilidad en el tiempo de desarrollo:	Estable

c. Diagrama BPMN

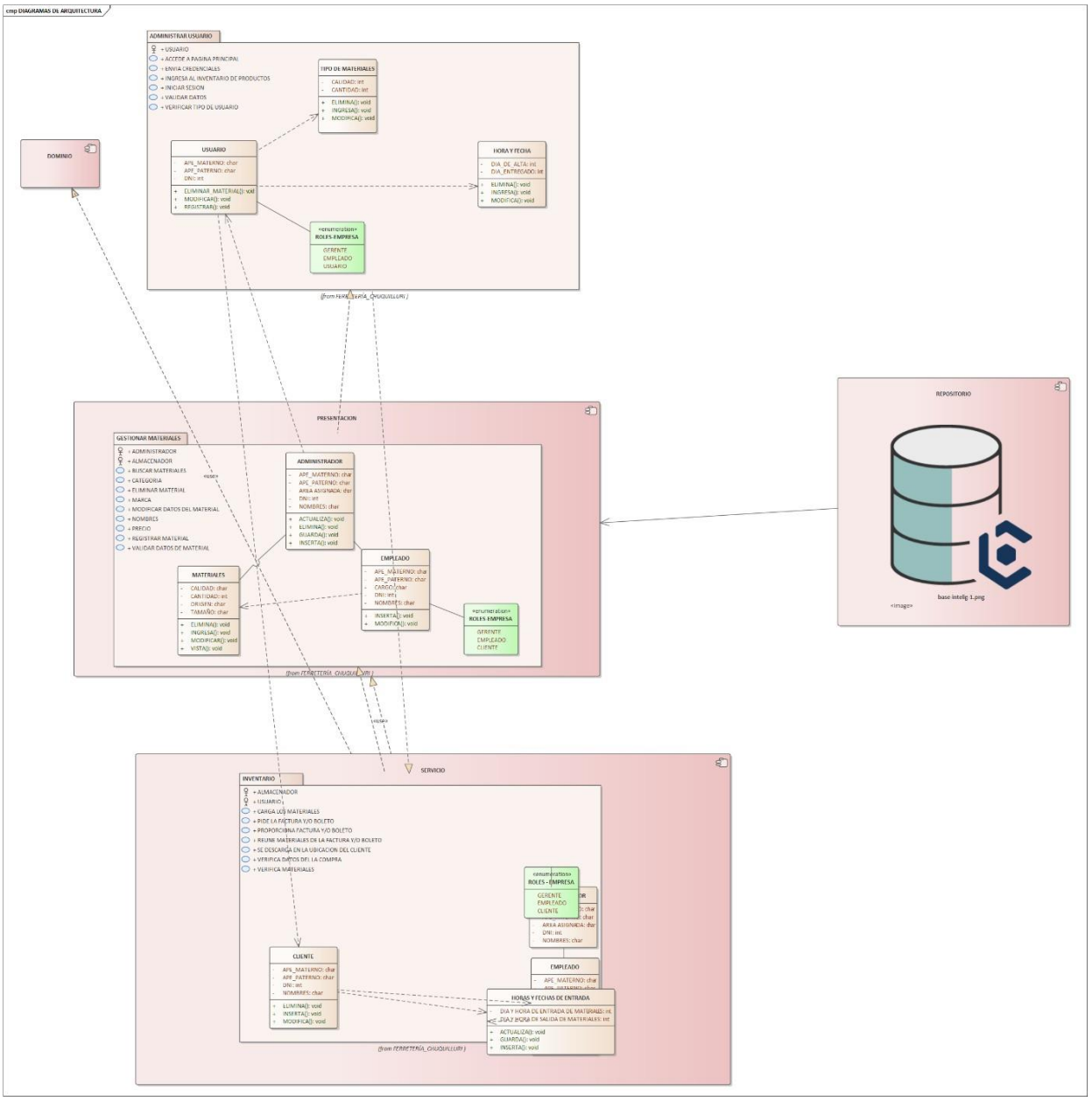


3. Cronograma de actividades(actividades y carta gannt)

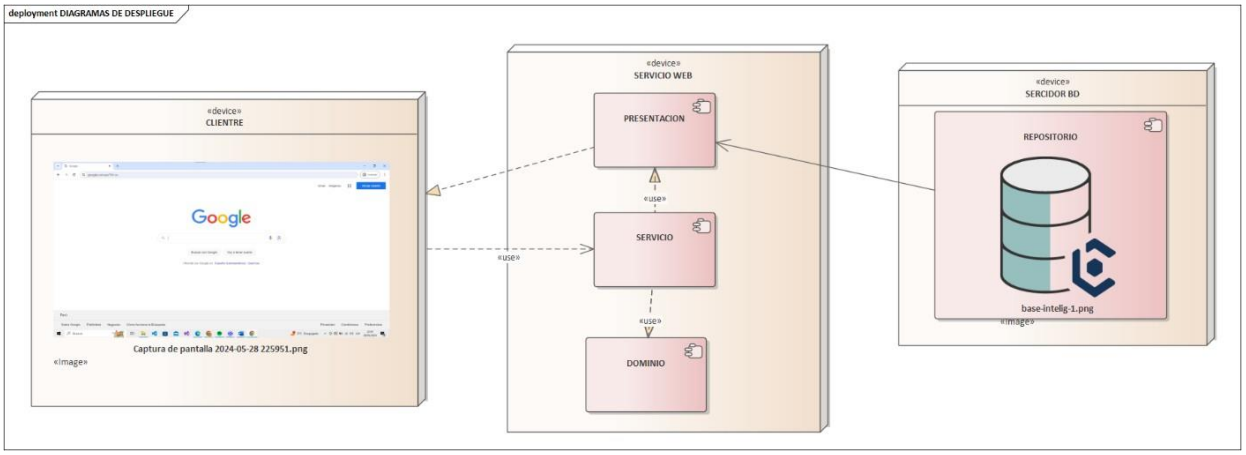
Trabajo Final del Curso



4. Desarrollo del Trabajo
    a. Arquitectura del sistema
        i. Diagrama componentes Arquitectura



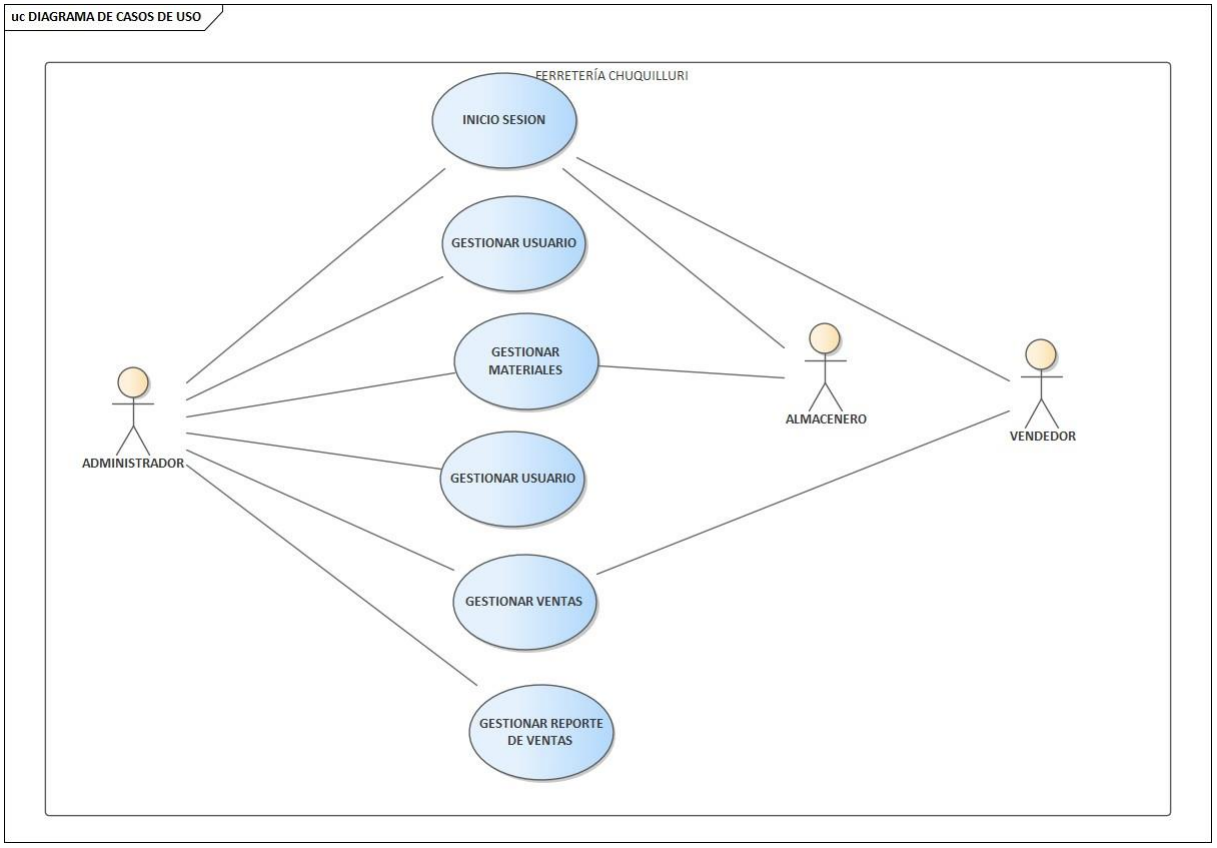
        ii. Diagrama de despliegue Arquitectura

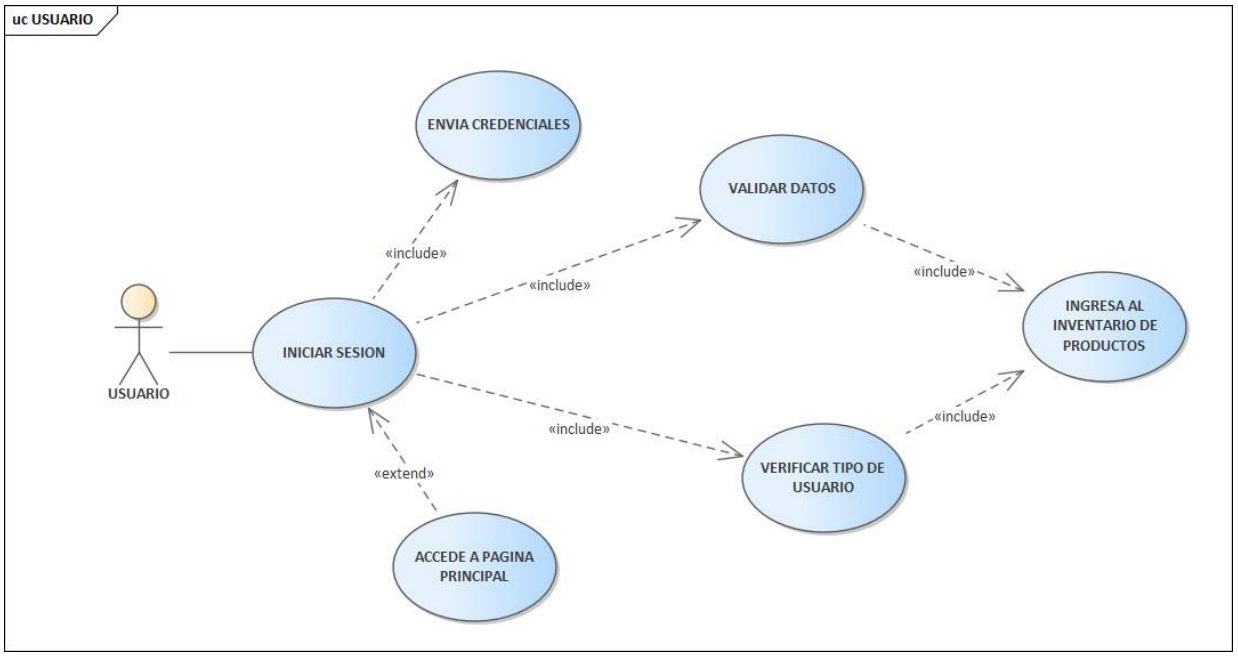


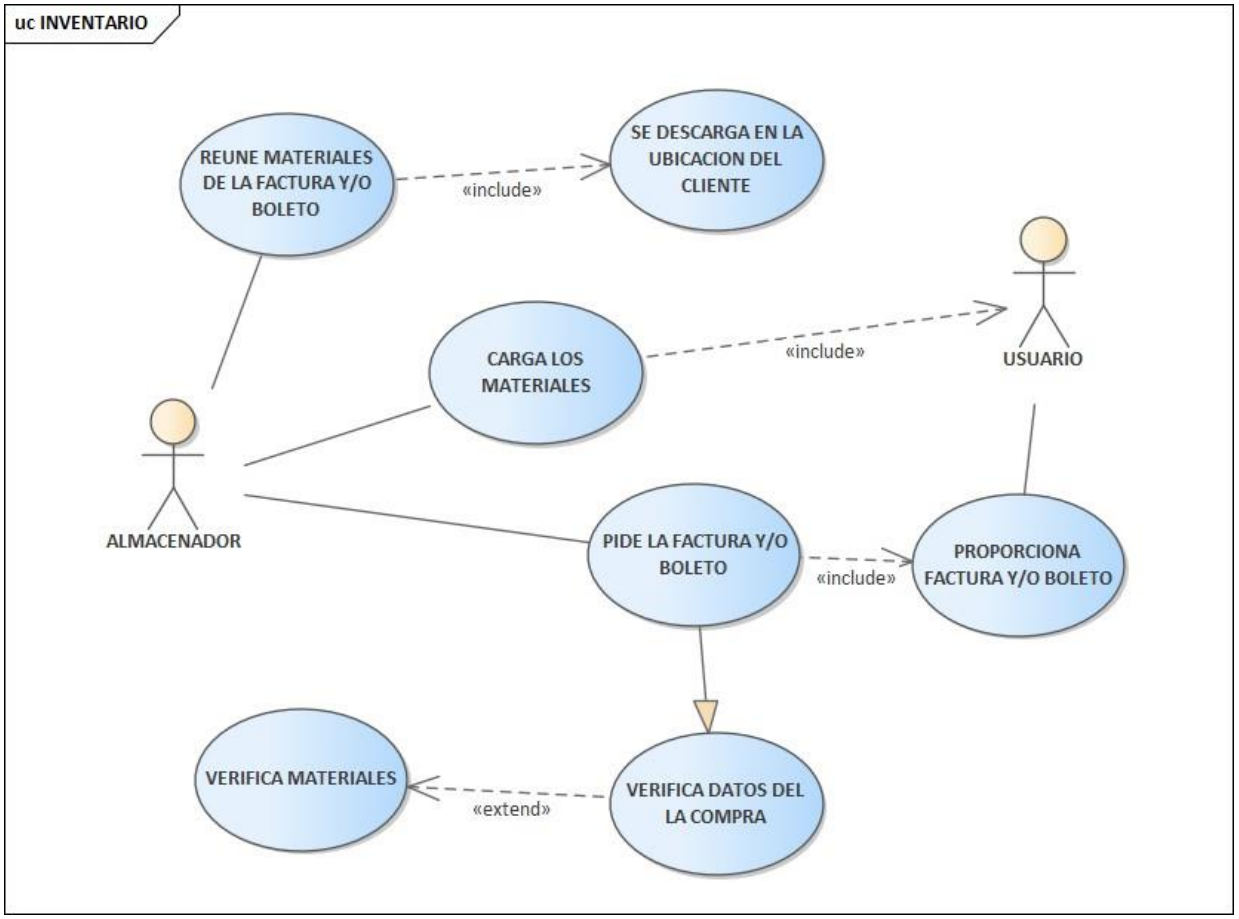
iii. Diagramas de despliegue por capa presentación y/o los que corresponda(Donde haya artefactos)

b. Diagramas

i. Diagrama de casos de uso

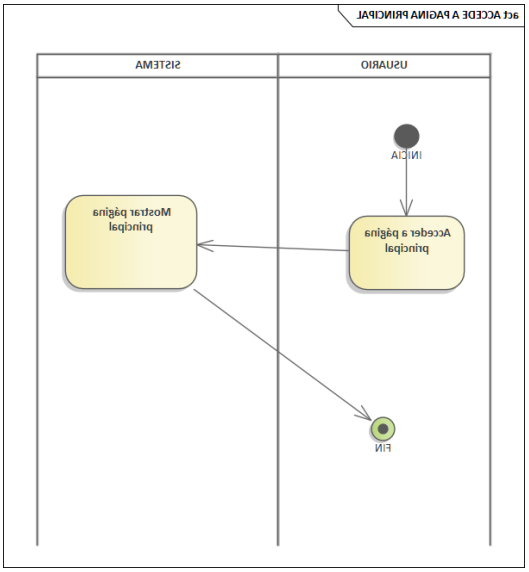


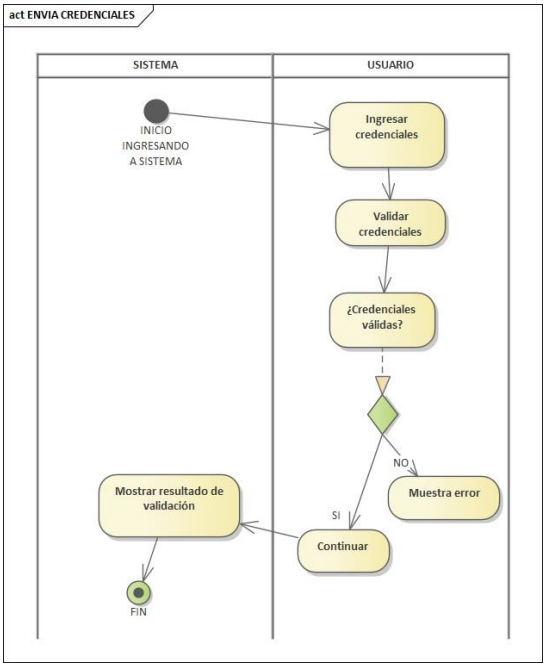


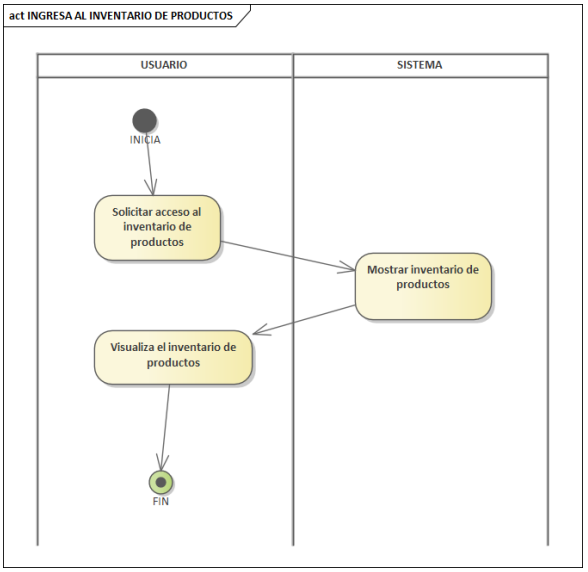


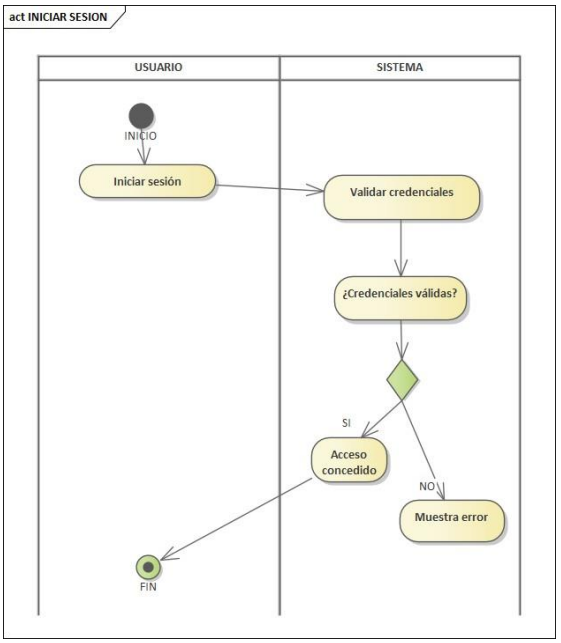
ii. Diagrama de Actividades x casos de uso

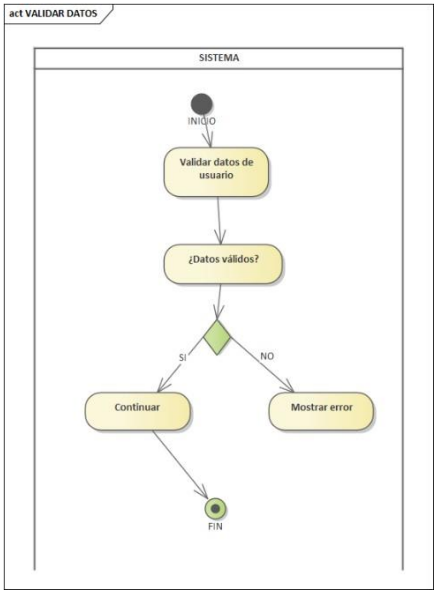
Trabajo Final del Curso

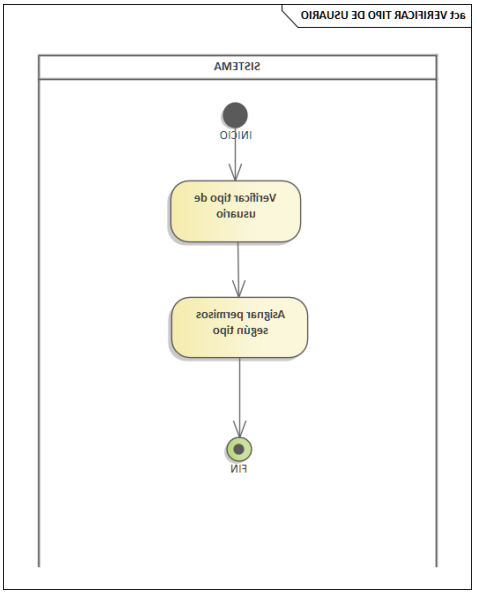




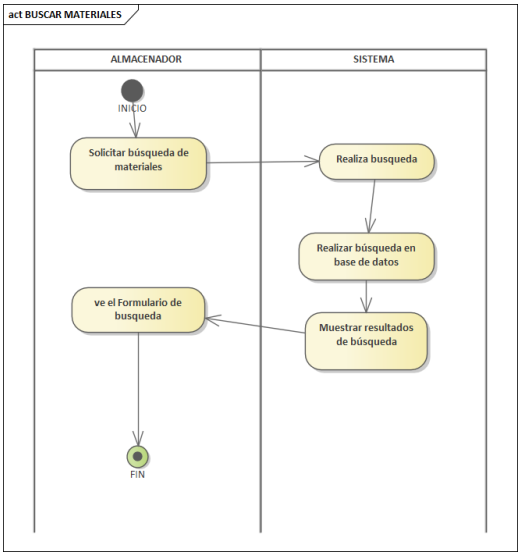


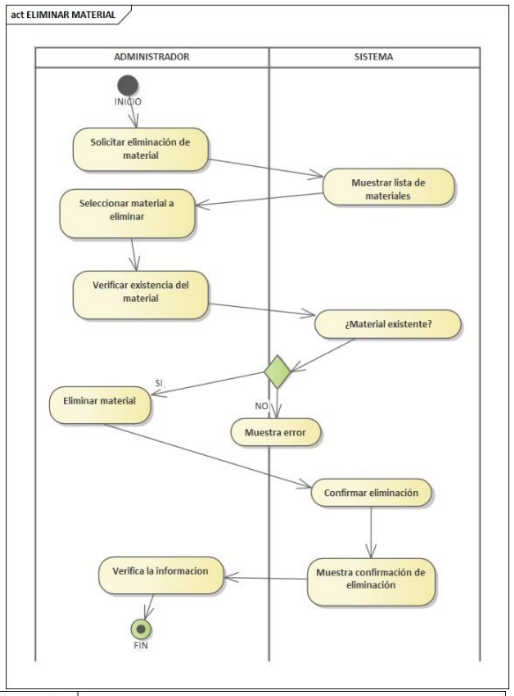


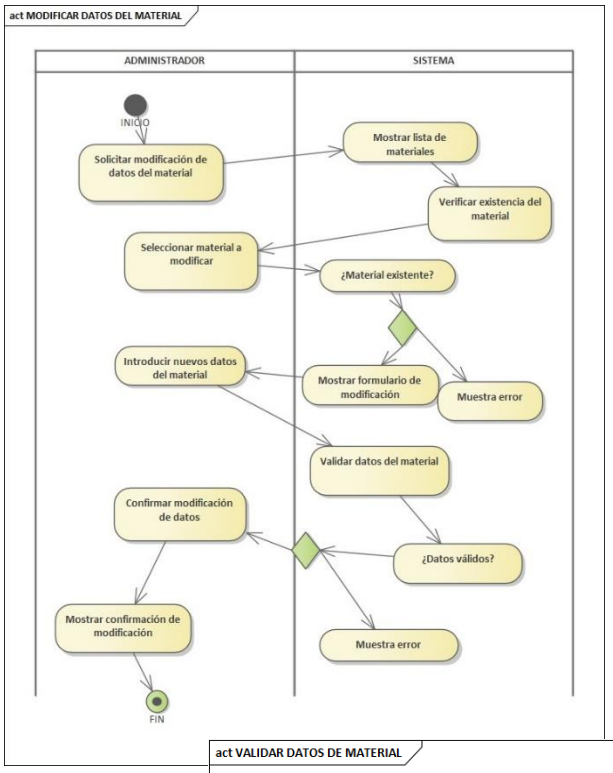


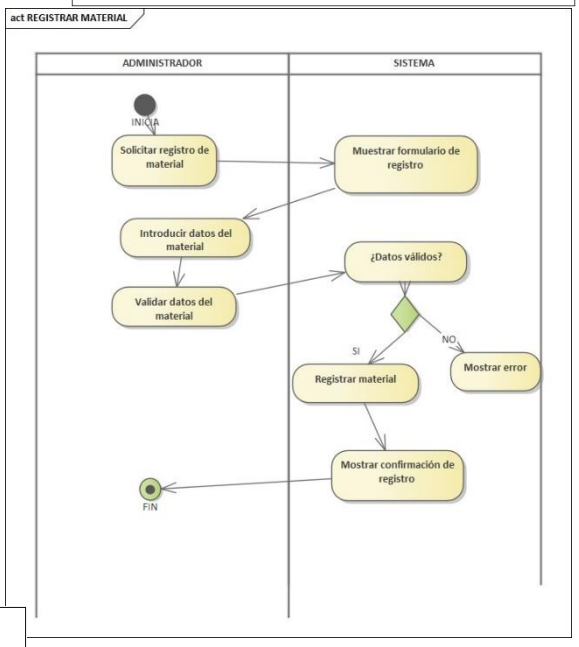


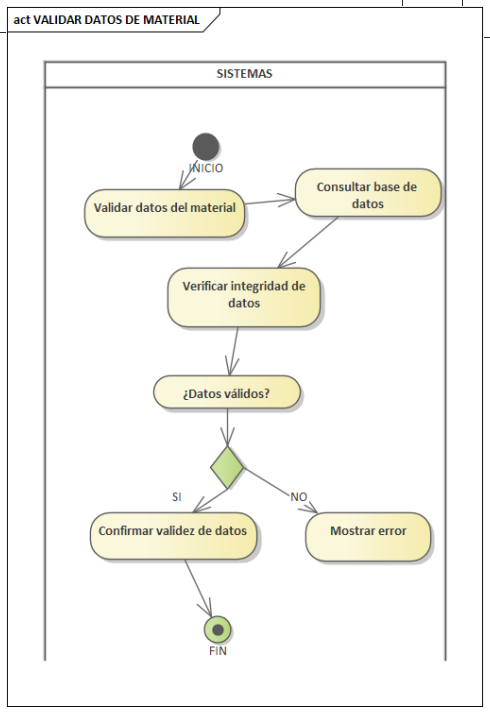
Trabajo Final del Curso



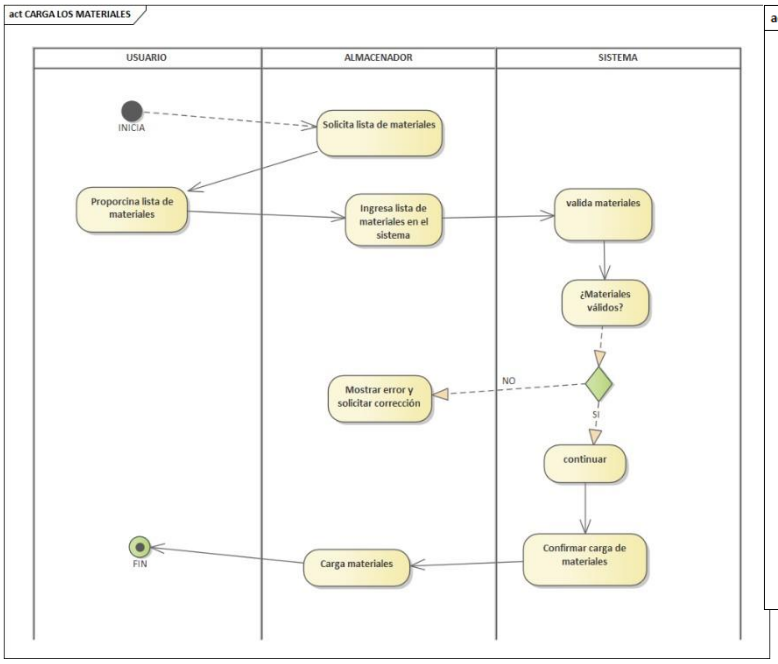


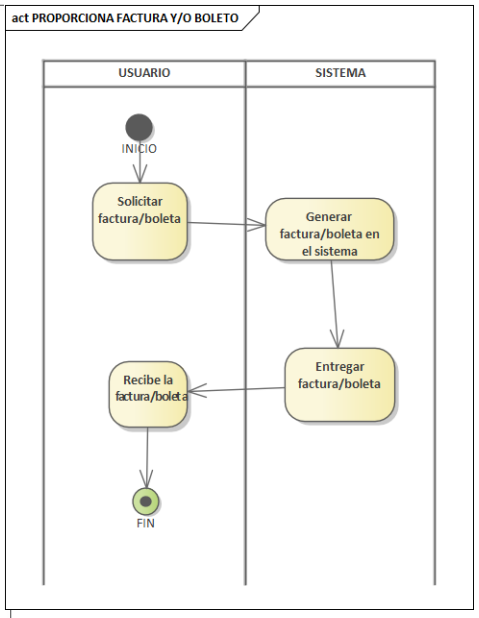


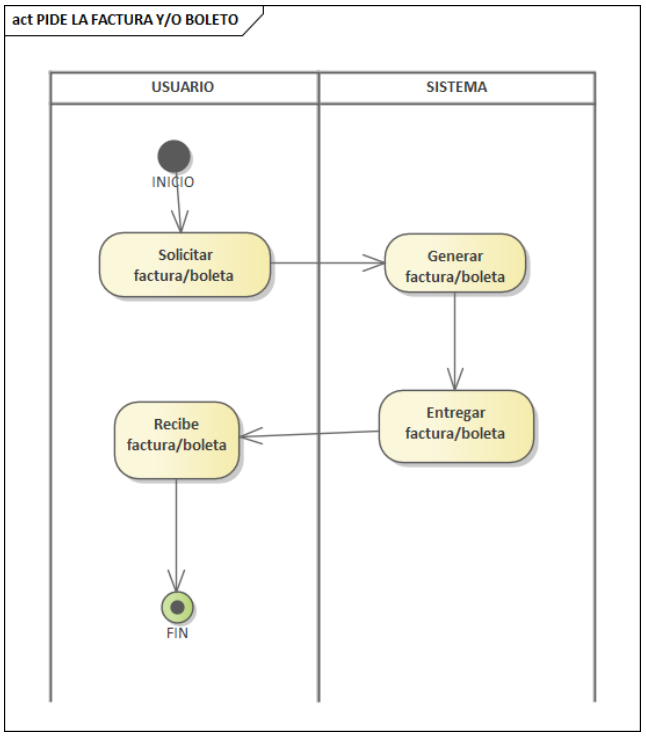


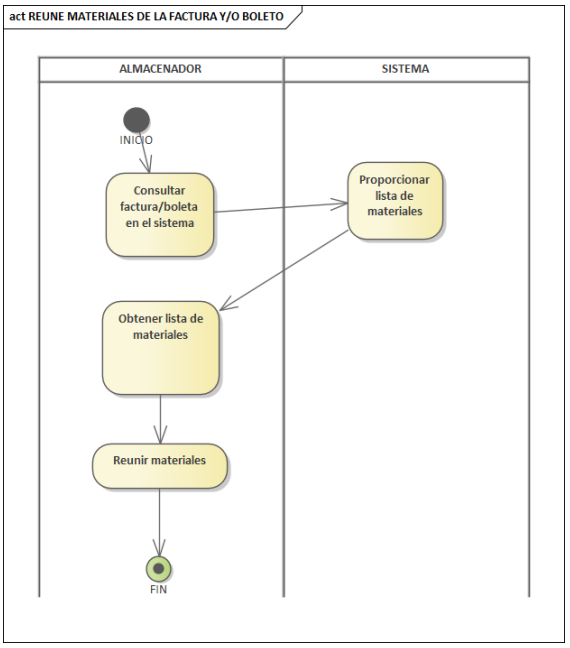


Trabajo Final del Curso

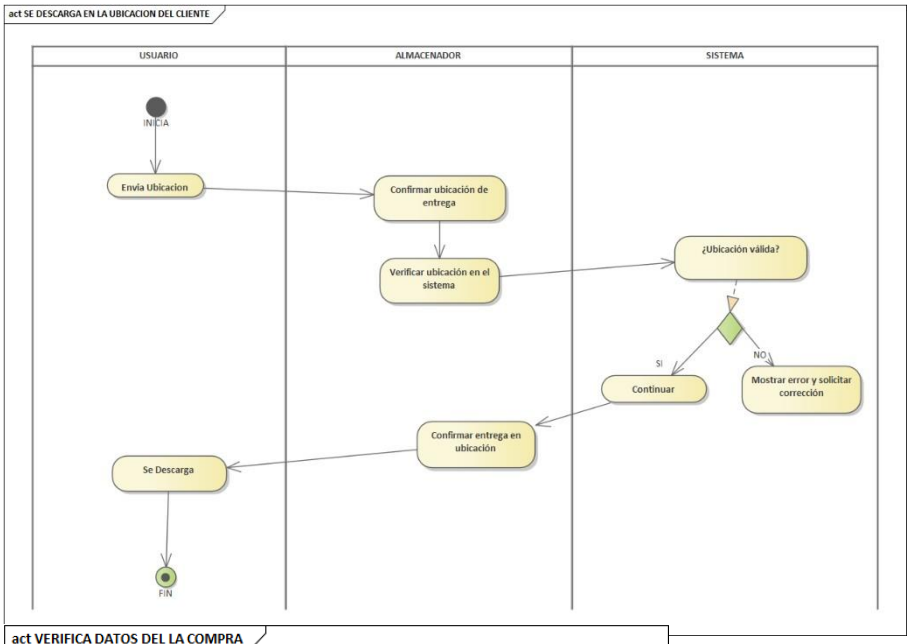


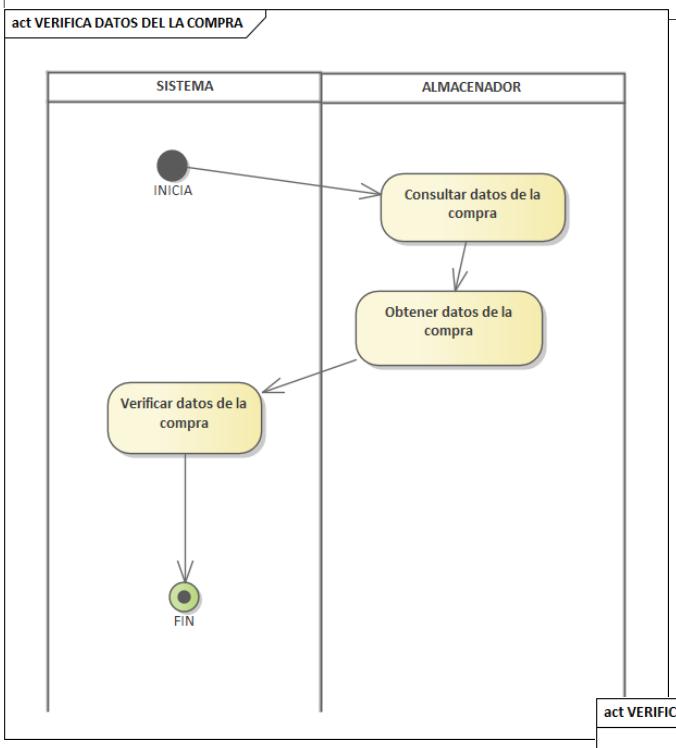


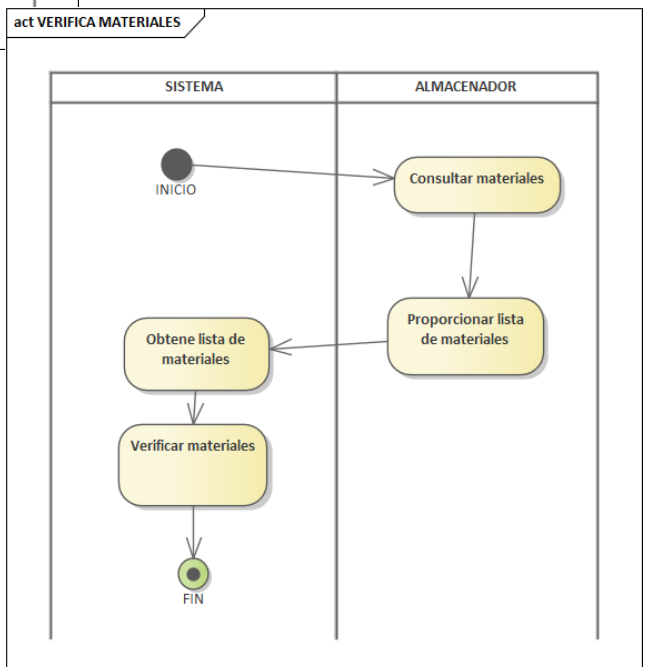




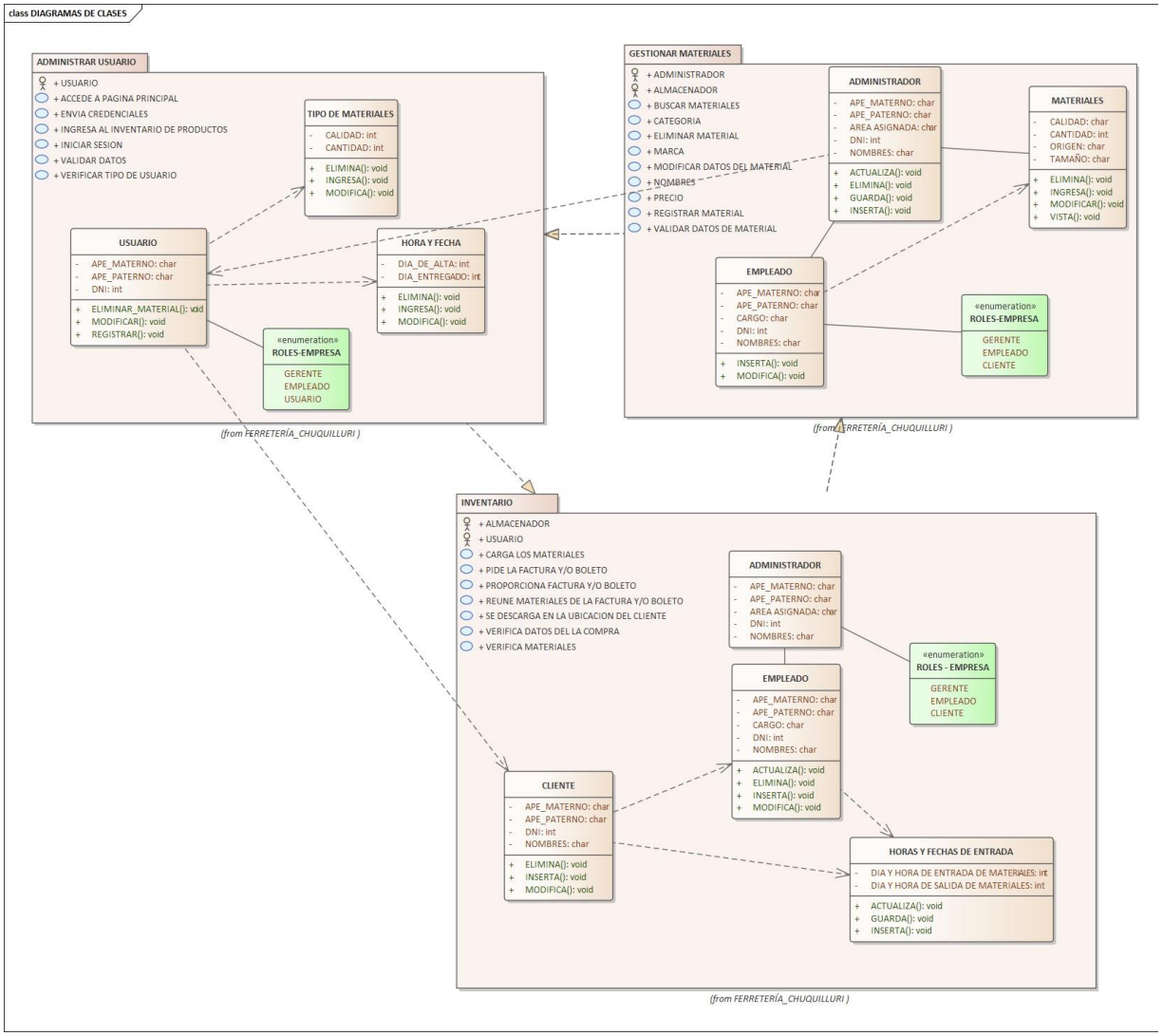
Trabajo Final del Curso



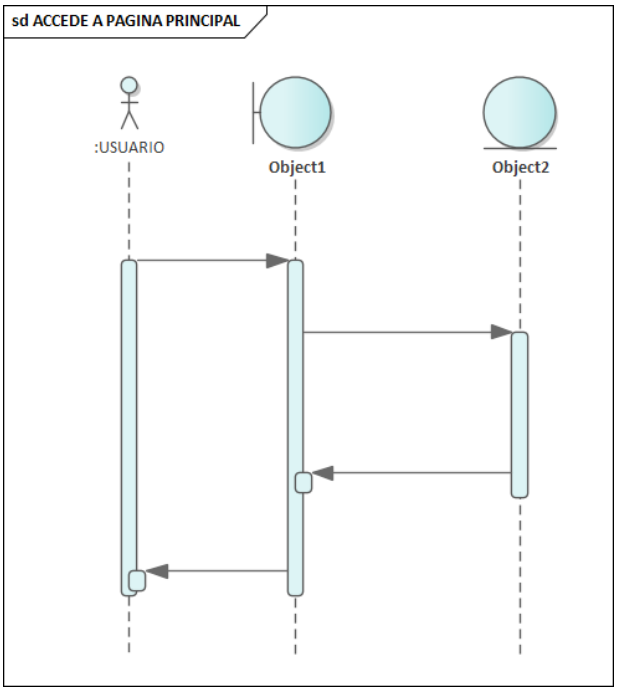


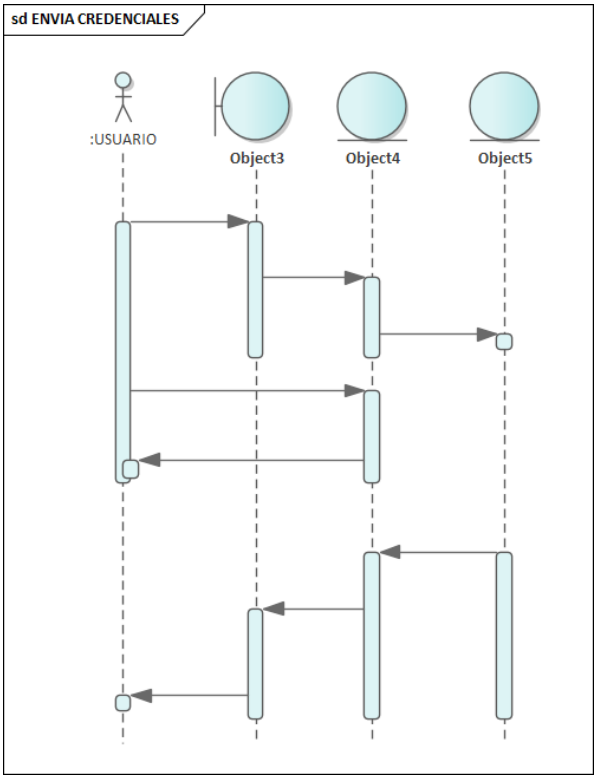


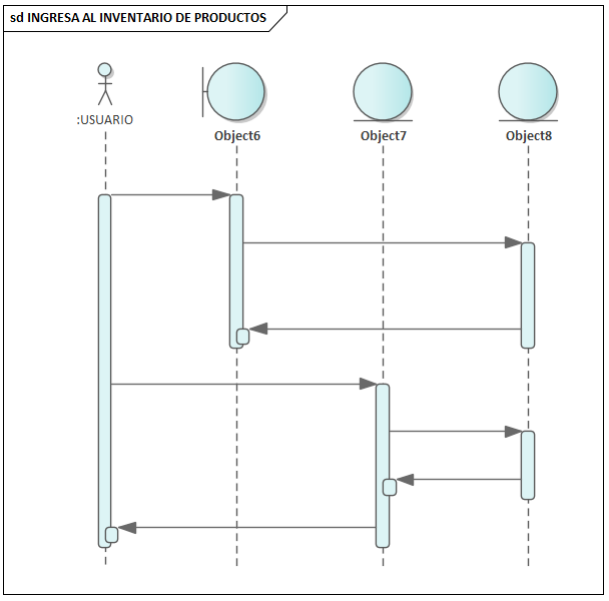
iii. Diagrama de clases

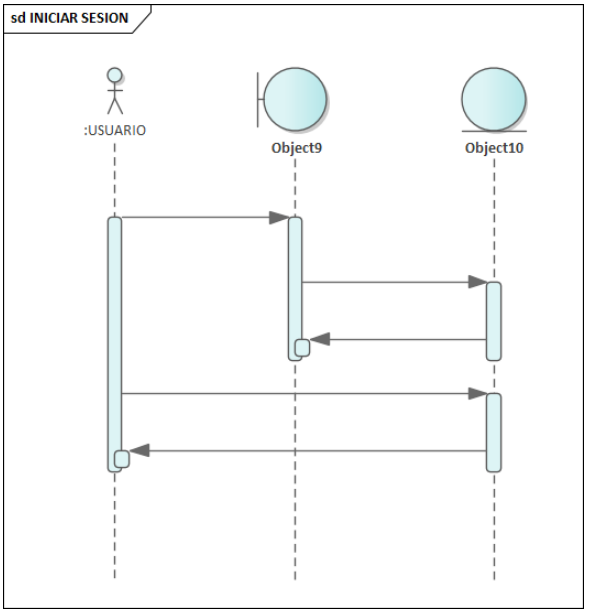


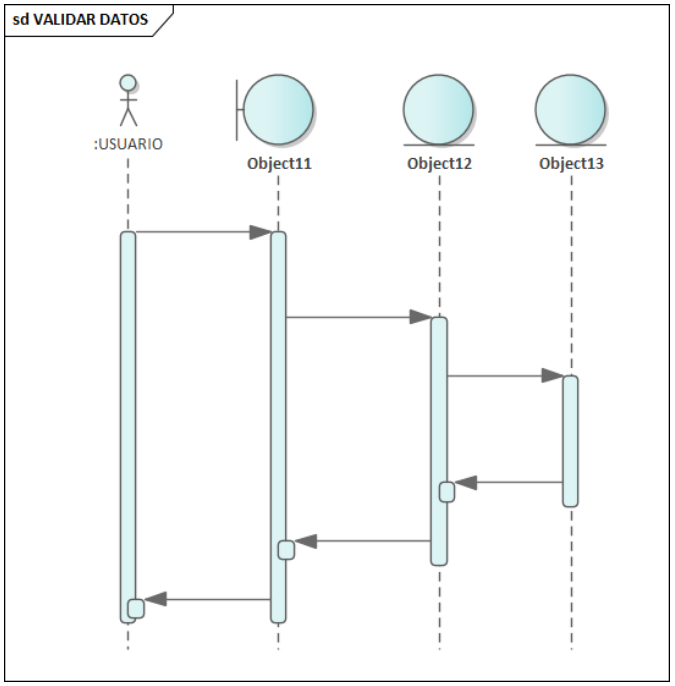
iv. Diagramas de secuencia

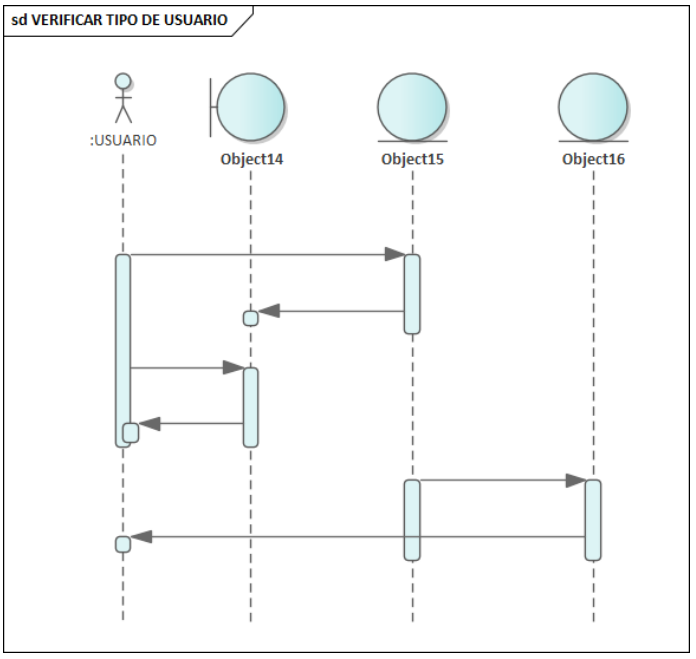


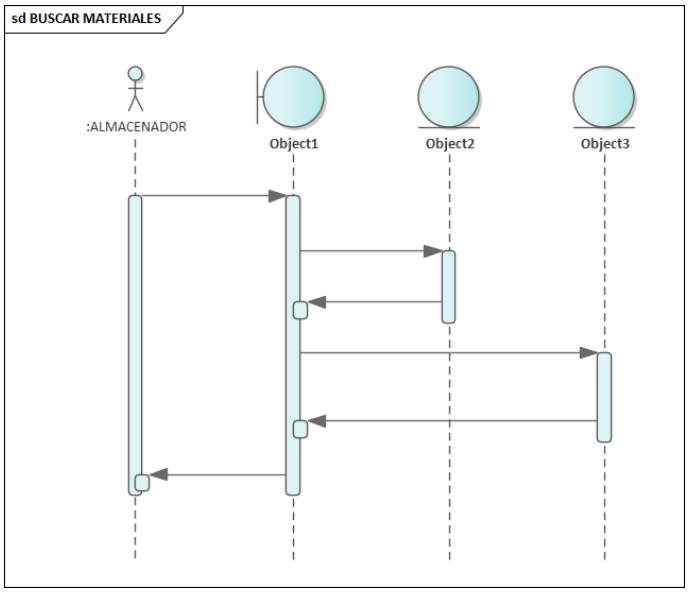


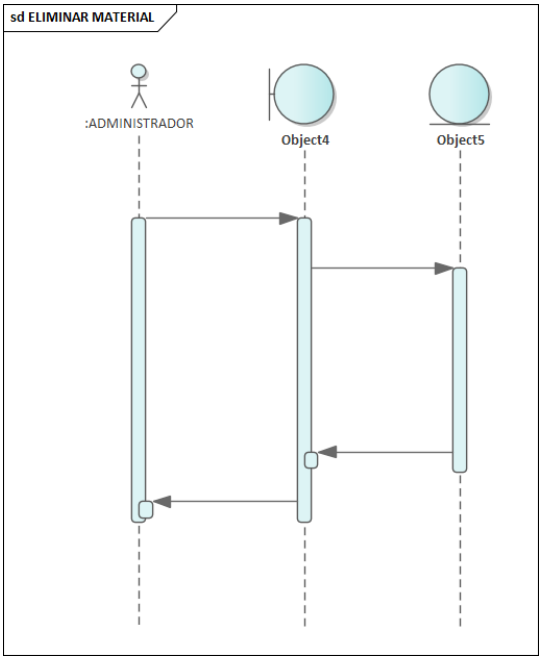


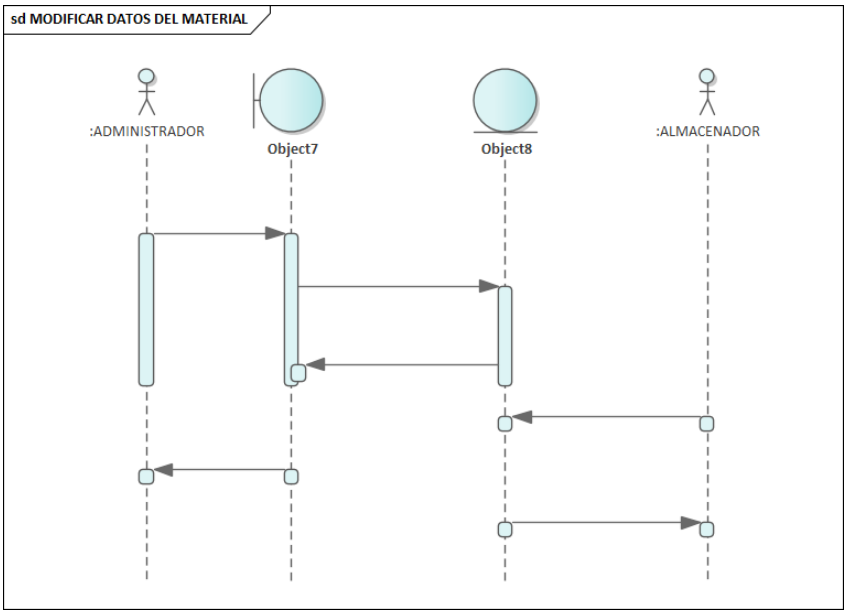


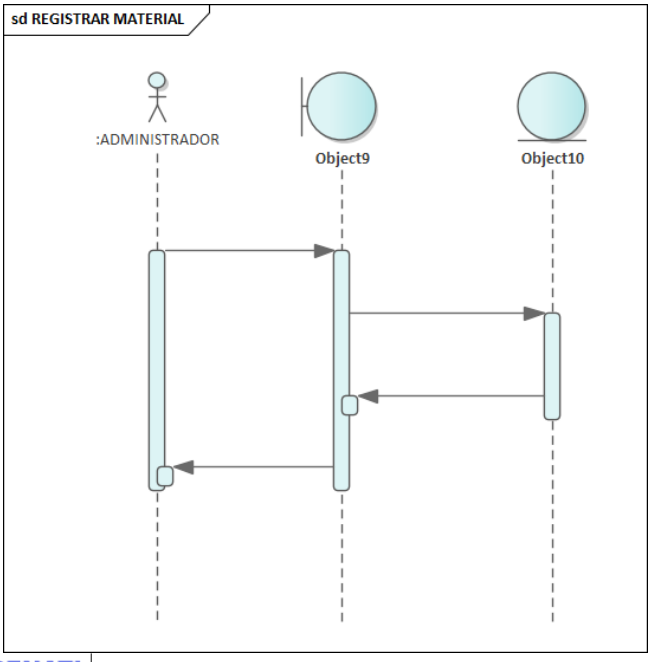

sd VALIDAR DATOS
:USUARIO
Object11
Object12
Object13

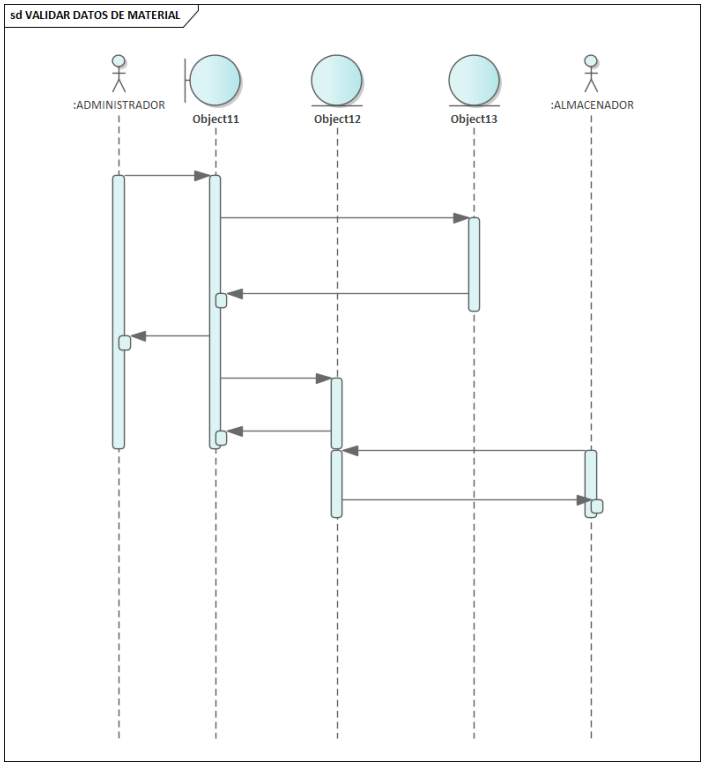

sd VERIFICAR TIPO DE USUARIO
:USUARIO
Object14
Object15
Object16

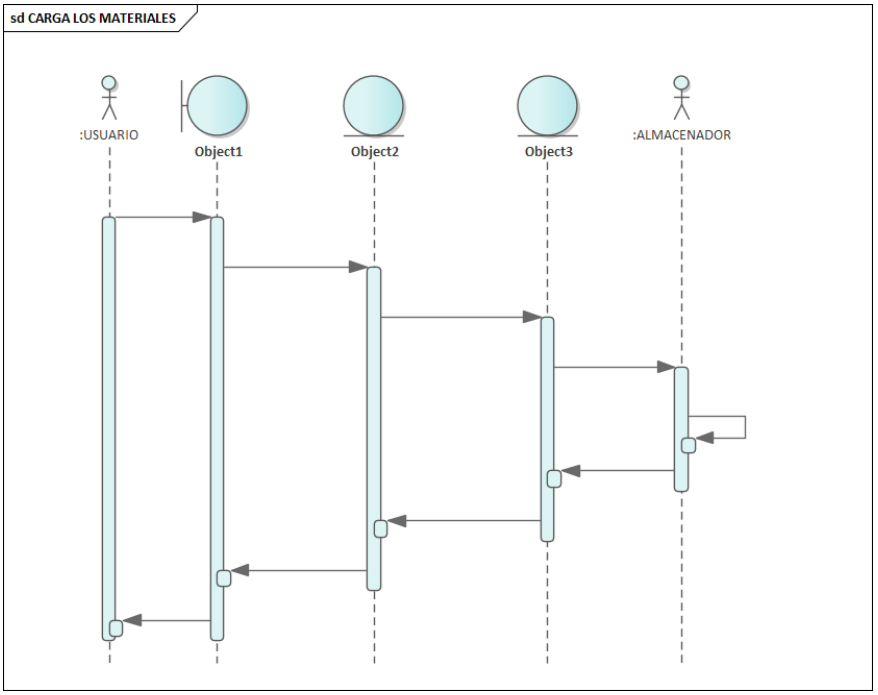

sd BUSCAR MATERIALES
:ALMACENADOR
Object1
Object2
Object3

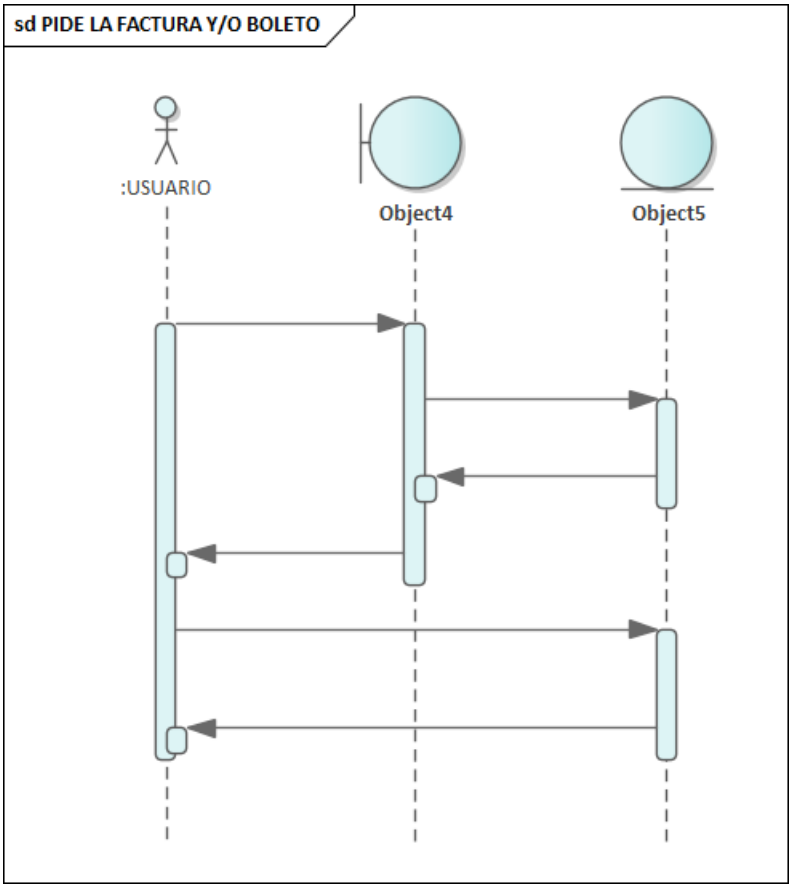

sd ELIMINAR MATERIAL
:ADMINISTRADOR
Object4
Object5

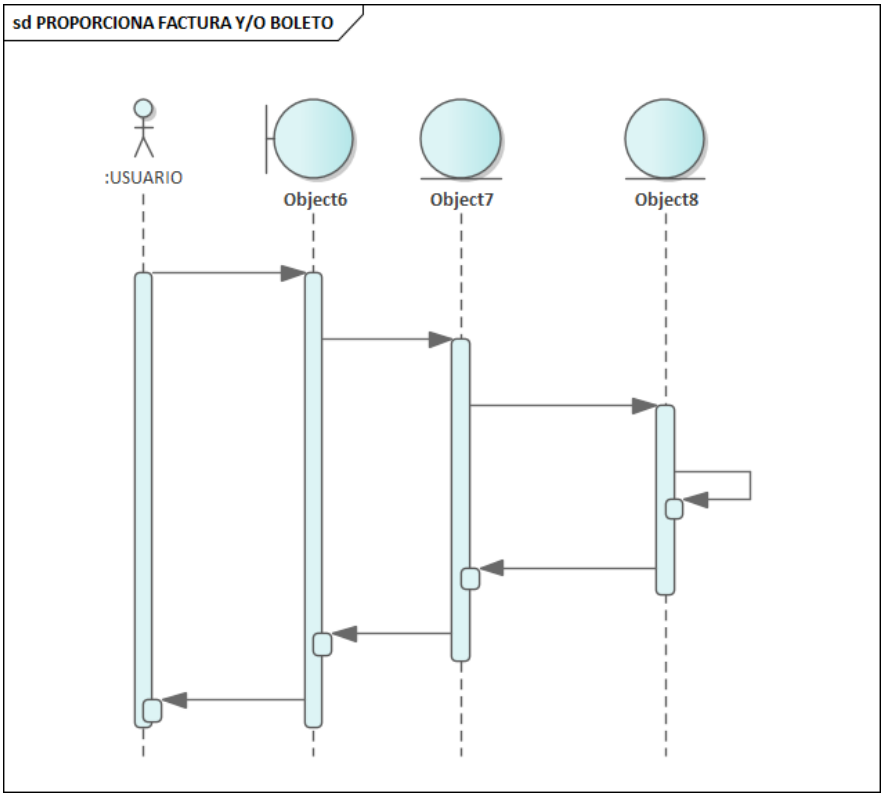

sd MODIFICAR DATOS DEL MATERIAL
:ADMINISTRADOR
Object7
Object8
:ALMACENADOR

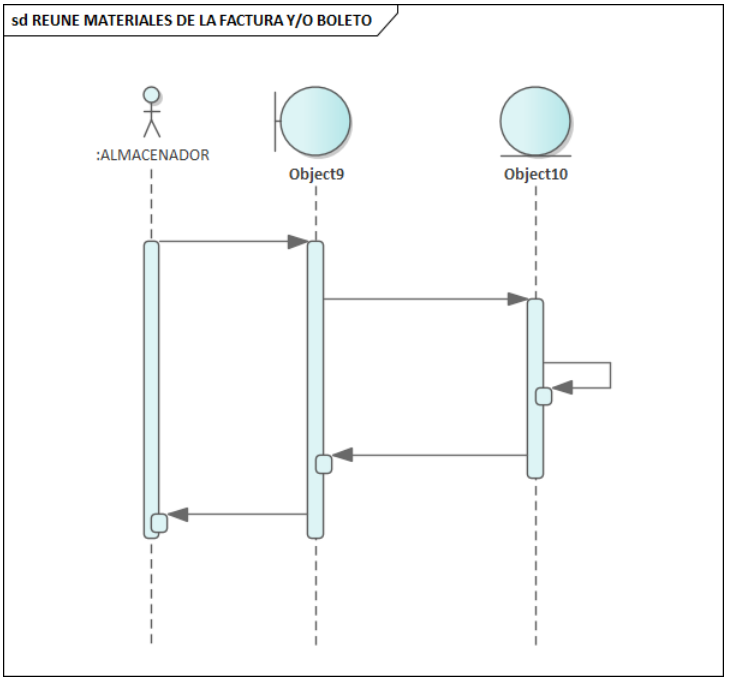

sd REGISTRAR MATERIAL
:ADMINISTRADOR
Object9
Object10

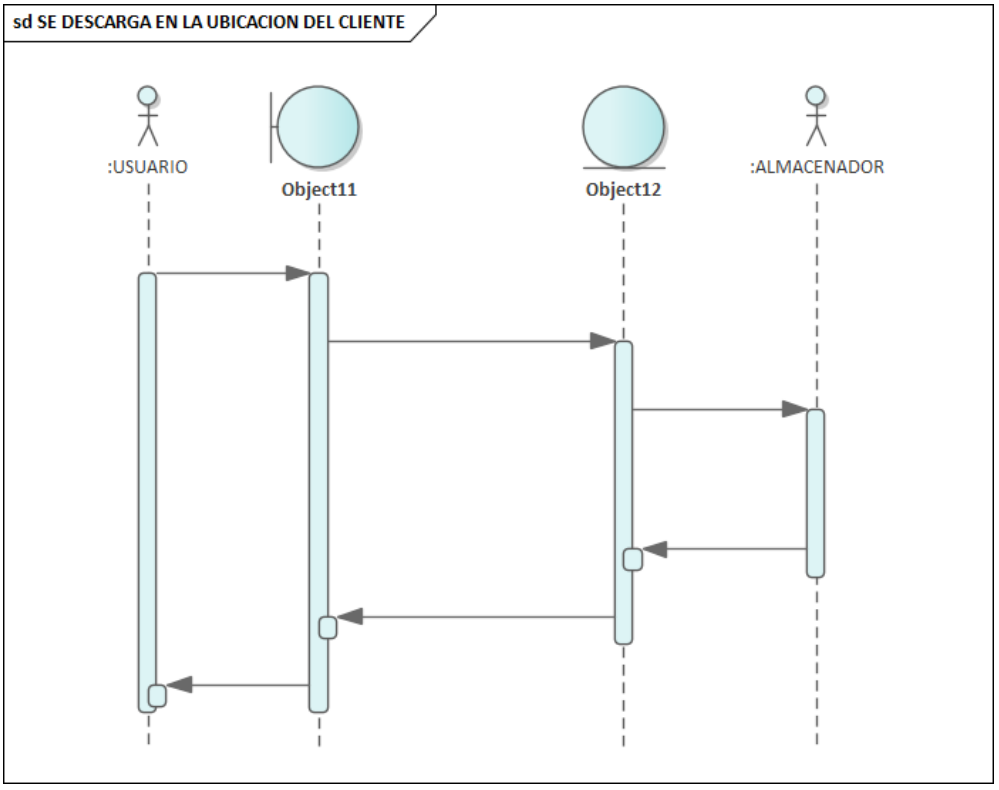

sd VALIDAR DATOS DE MATERIAL
:ADMINISTRADOR
Object11
Object12
Object13
:ALMACENADOR

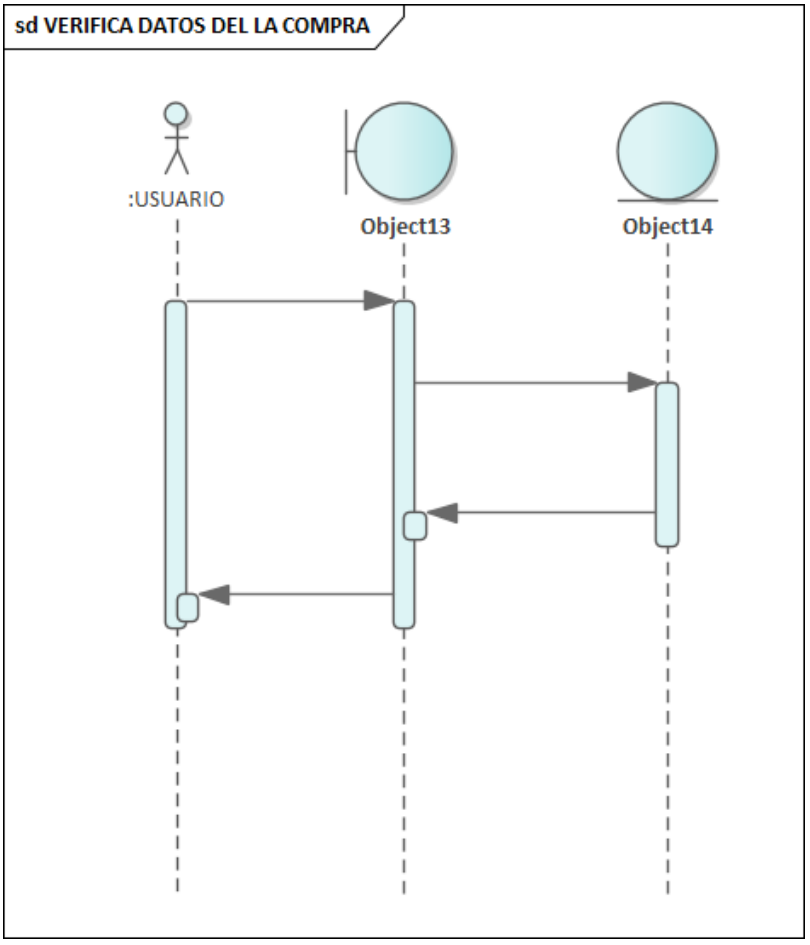

sd CARGA LOS MATERIALES
:USUARIO
Object1
Object2
Object3
:ALMACENADOR

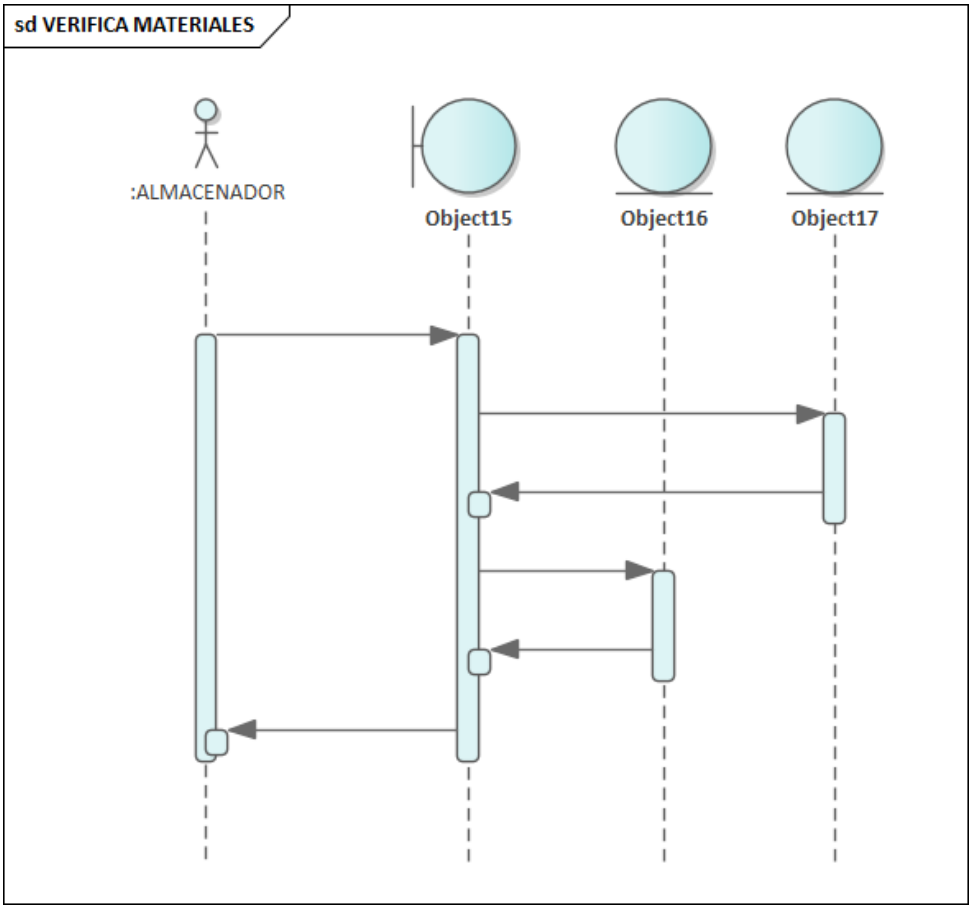

sd PIDE LA FACTURA Y/O BOLETO
:USUARIO
Object4
Object5


sd PROPORCIONA FACTURA Y/O BOLETO
:USUARIO
Object6
Object7
Object8


sd REUNE MATERIALES DE LA FACTURA Y/O BOLETO
:ALMACENADOR
Object9
Object10


sd SE DESCARGA EN LA UBICACION DEL CLIENTE
:USUARIO
Object11
Object12
:ALMACENADOR


sd VERIFICA DATOS DEL LA COMPRA
:USUARIO
Object13
Object14


sd VERIFICA MATERIALES
:ALMACENADOR
Object15
Object16
Object17

Segunda Entrega

- c. Diseño
  - i. Detalles de los casos de uso
  - ii. Diseño de la interfaz
- d. Implementación
  - i. Implementación de las clases
  - ii. Modelado de la Bases de datos

5. Conclusiones
6. Bibliografía

PREGUNTAS GUÍA

1.- ¿Que determina el uso de una arquitectura de software?

2.- ¿Que son los Diagramas de casos de uso?

3.- ¿Que son los Diagramas de clases?

4.- ¿Que son los Diagramas de secuencia?