UACM

Universidad Autónoma de la Ciudad de México

Nada humano me es ajeno

Documento de diseño

Integrantes:

Martínez Zarate Yasbeth Mariana

García de Jesús Héctor

Flores Trejo Víctor Rubén

Robles Vásquez Daniela Judith

Autor	Fecha	Versión
Martinez Zarate Yasbeth Mariana	28/05/23	5.0

INDICE

Autor	Fecha	Versión
Martinez Zarate Yasbeth Mariana	28/05/23	5.0

TABLA DE ILUSTRACIONES

Autor	Fecha	Versión
Martinez Zarate Yasbeth Mariana	03/12/23	6.0

1. INTRODUCCIÓN

Dentro del siguiente apartado se definirán varios puntos de vista de diseño, se ilustran estos puntos en términos de selecciones de lenguaje de diseño, relacionando preocupaciones de diseño con varias perspectivas, así como establecer nombres neutrales de lenguaje.

Para cada punto de vista se proporcionan breves descripciones que relacionan un conjunto mínimo de entidades de diseño, relaciones de diseño, atributos de entidad relación, así como restricciones de diseño.

El documento deberá incluir información sobre la arquitectura del sistema, la estructura de datos, diagramas de flujo y otros detalles más importantes, también incluirá información sobre los requisitos del sistema, las pruebas y los casos de uso.

Este documento es una parte esencial del proceso de desarrollo de software, ya que proporciona una guía detallada para los desarrolladores sobre cómo construir el sistema y cómo espera que funcione. Es por eso que cada sección de nuestro documento se explicará por medio de diferentes diagramas que realiza cada sección, dentro de esos diagramas contaremos con los diagramas UML, como lo es el diagrama de secuencia, diagrama de clases, diagrama de componentes, los diagramas de paquetes estos siendo de vital importancia para la realización de diseño del sistema que se va a realizar.

Autor	Fecha	Versión
Martinez Zarate Yasbeth Mariana	03/12/23	6.0

2. PROPÓSITO

El propósito de este documento es proporcionar una descripción detallada de cómo se construirá el sistema y cómo funcionará. El documento es una herramienta esencial para los desarrolladores, ya que proporciona una guía detallada de cómo construir el sistema y con todas las especificaciones requeridas.

El documento deberá proporcionar una descripción clara de la arquitectura del sistema, la estructura de datos, los diagramas, así como asegurarse de que el sistema final cumpla con los requisitos del cliente y el usuario, también deberá facilitar la comunicación entre los miembros del equipo de desarrollo, cliente y usuario.

Autor	Fecha	Versión
Martinez Zarate Yasbeth Mariana	03/12/23	6.0

3. ALCANCE

El alcance de esta documentación es describir con detalle las metas y los objetivos generales de la solución, al tiempo que captura los procesos de automatización necesarios para completar la solución a nuestro problema a resolver.

Lograremos definir los requisitos del sistema, incluyendo el objetivo del software, los recursos necesarios y las restricciones que se encuentran dentro del sistema. Describiremos la arquitectura del sistema, aquí se describe la estructura general del software, incluyendo los módulos, subsistemas y la relación que existe entre cada una y la comunicación que puede existir con otros sistemas.

Se describe el diseño de la interfaz, diseño de paquetes, diseño de la vista lógica, entre otras, para así facilitar la manera de entender y comprender el sistema a realizar. Cada uno de los puntos de vista describirá con problemas, elementos y atributos en que consiste cada uno de ellos.

Autor	Fecha	Versión
Martinez Zarate Yasbeth Mariana	03/12/23	5.0

4. PUNTO DE VISTA DE CONTEXTO

Aquí nos referiremos a la manera en que se interpreta y se comprende una situación, evento o mensaje a través del análisis de su contexto.

El punto de vista del contexto representa los servicios proporcionados por un sujeto de diseño con referencias a un contexto explícito, este contexto se define por referencias a actores que incluyen usuarios y otras partes interesadas, que interactúan con el sujeto de diseño en su entorno.

4.1. PROBLEMAS DE DISEÑO

Aquí nos enfocaremos en identificar y abordar las interacciones entre el software y el contexto en el que se utiliza. Estos problemas pueden surgir cuando el software no está adecuadamente adaptado a las características específicas del contexto de uso, lo que puede dificultar su uso efectivo por parte de los usuarios finales.

El problema que se puede presentar en el diseño en este caso es que todo el sistema se presente como una caja negra, es decir que se puede conocer el sistema solo con lo que se le da al usuario o comprador, pero no verá más a fondo el sistema, como el funcionamiento interno (algoritmo) del sistema y esto puede delimitar el conocimiento de ciertas funciones, se podrá conocer las entradas y salidas del sistema, pero difícilmente cómo funciona internamente.

4.2. ELEMENTOS DE DISEÑO

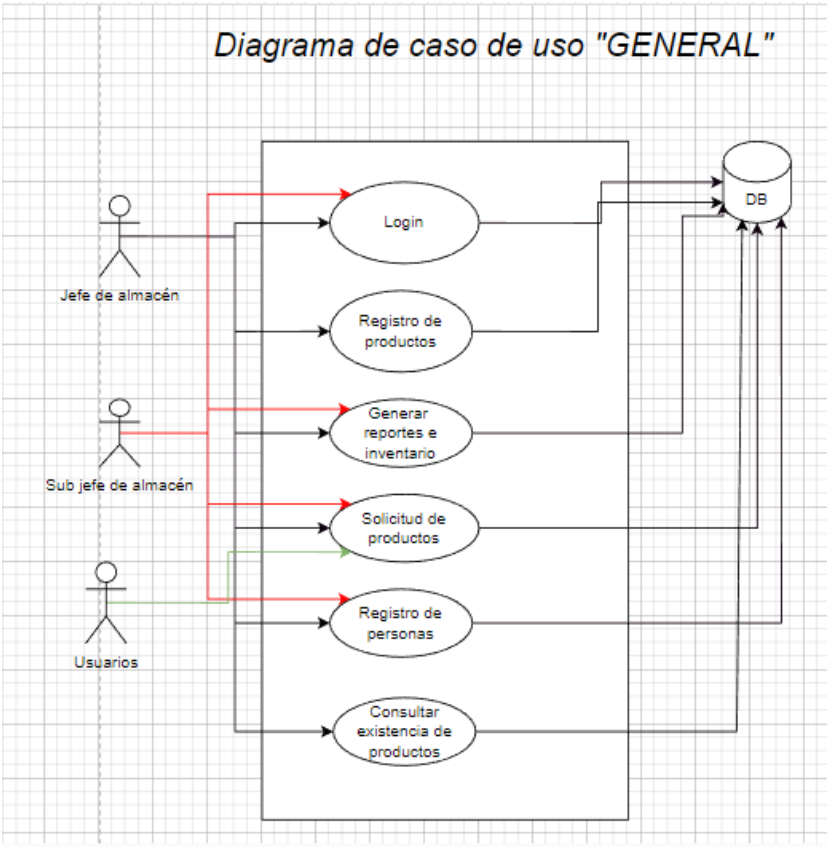
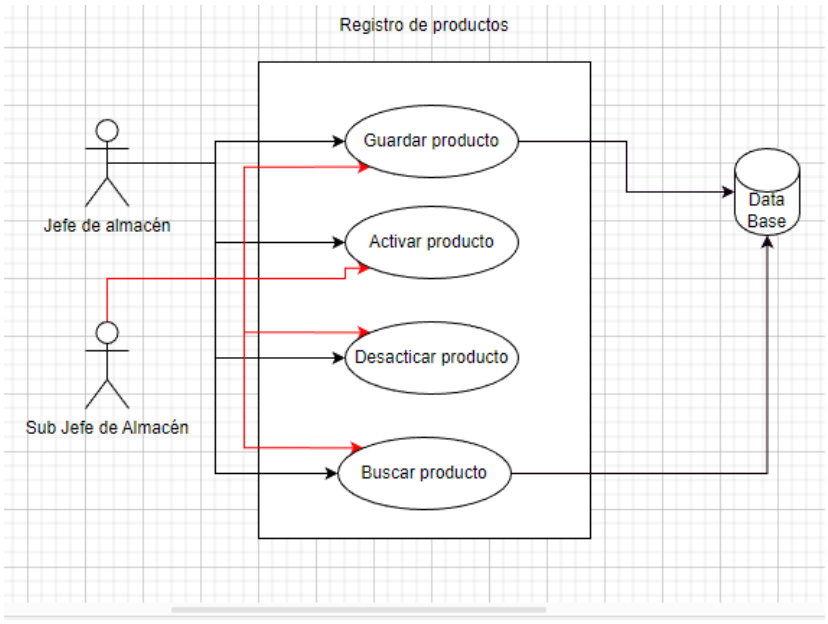


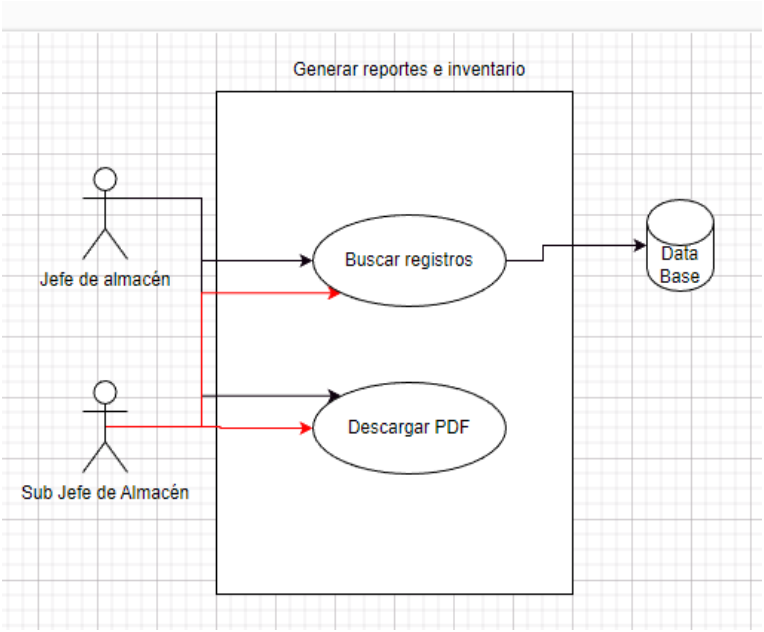
Diagrama de caso de uso "general" 1

Aquí se visualiza el diagrama de caso de uso general, en el cual se encontrarán los actores y las funciones que estos van a realizar y los roles que tendrá cada uno.



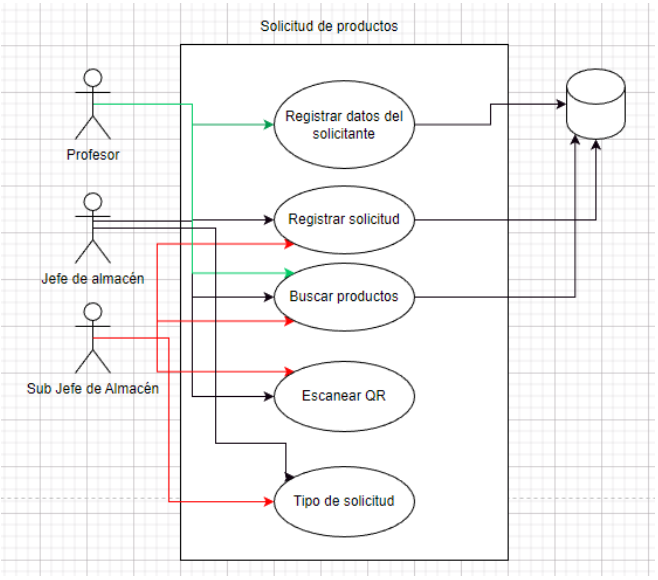
Caso de uso "Registro de productos" 2

En este diagrama de caso de uso se visualizará lo que harán los actores (jefe de almacén y Sub jefe de Almacén) los cuales estarán utilizando los casos de uso (funciones del sistema). Entre las funciones que realizarán será guardar producto, aquí cualquiera de los actores podrá ingresar y guardar los datos en una Data Base, así como activar y desactivar los productos dentro del mismo caso de uso, buscará productos también en el registro de productos.



Caso de uso "Generar reportes e inventario" 3

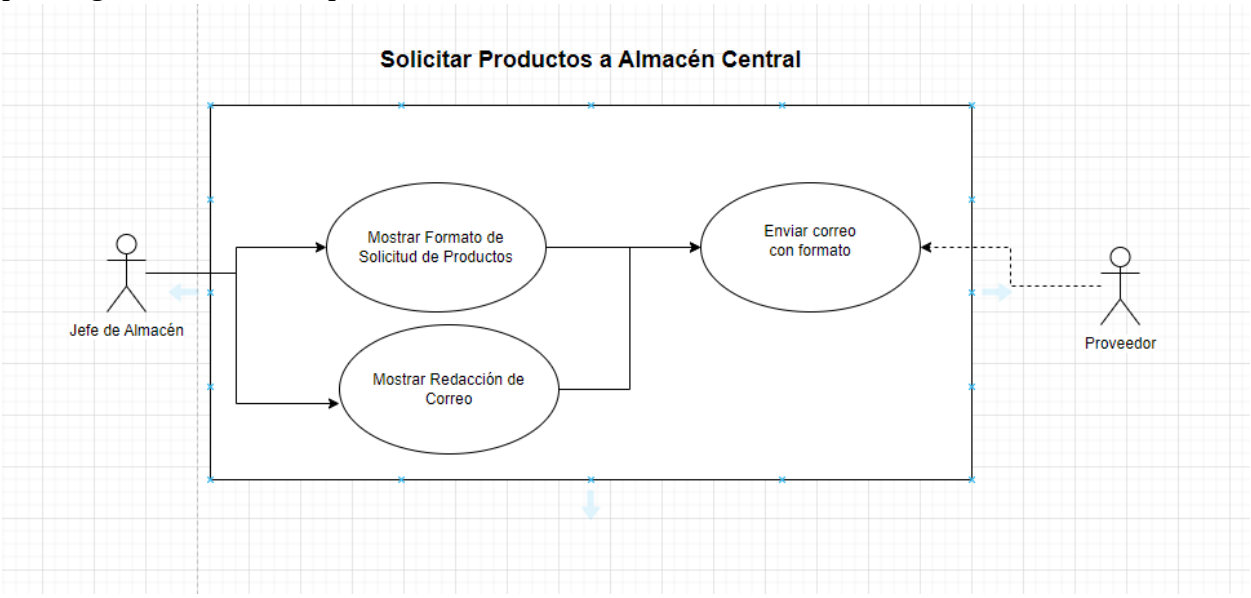
En este caso de uso "Generar reportes e inventarios" los actores podrán realizar dos funciones, las cuales constarán de buscar registro, esto por medio de la Data Base, así como podrán descargar dicho reporte en PDF y posteriormente se podrá imprimir.



Caso de uso "Solicitud de productos" 4

En este caso de uso interactúan tres actores con el sistema (jefe de almacén, subjefe de almacén y los usuarios), en este caso los usuarios serán los profesores y personal de limpieza del plantel. Las

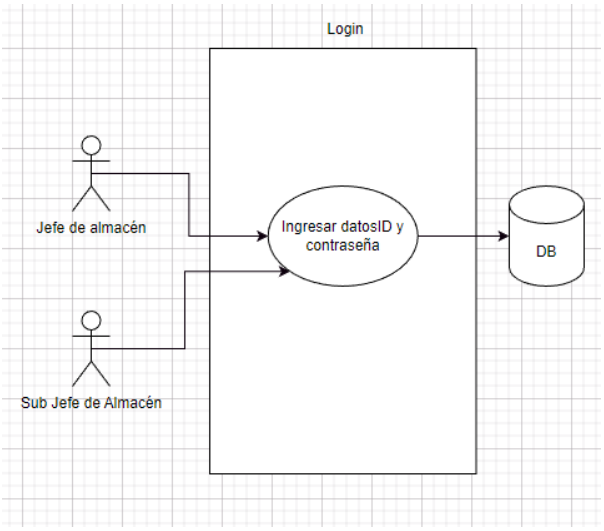
funciones que podrán realizar en el sistema serán cuatro, el registrar datos del solicitante interactúa el usuario registrando sus propios datos, esto será guardado en la Data Base. El apartado de registrar solicitud solo lo podrán utilizar dos actores, este también será manejado por la Data Base. El buscar productos se realizará también por medio de los datos ya registrados en la Data Base. La función de escanear QR será utilizada por dos actores y estos son los encargados de registrar el QR de los usuarios para registrar sus datos o productos.



Caso de uso "Solicitar productos a almacén central" 5

En este caso de uso de "solicitar productos a almacén central" solo se encontrará a un actor y este realizará una sola función que es el llenar el formato de solicitud de productos, este mismo será enviado por medio de un correo que se hará en el momento de hacer la solicitud.

Este apartado será integrado con el apartado de solicitud de productos, ahí se seleccionará si requieren solicitar productos del almacén o solicitar productos al almacén central.



Caso de uso "Login" 6

En el caso de uso (Ilustración 6) de "Login" se encontrarán dos actores (jefe de almacén y sub jefe de almacén) estos ingresarán sus datos, ID y contraseña para poder ser buscados en una Data Base y este les permita acceso si sus datos son correctos.

4.3. IDIOMAS DE EJEMPLO

Algunos de los idiomas de lenguaje de diseño para nuestro sistema serían:

- IDEF0: Es un lenguaje de modelado y diseño de proceso que se utiliza para representar gráficamente las funciones y actividades de un sistema o proceso. Este lenguaje se utiliza para modelar los sistemas en una perspectiva funcional y para representar la secuencia lógica de actividades en un proceso.
- Diagrama de contexto de análisis estructurado: Este diagrama muestra la interacción del sistema con su entorno, identificando las entradas y salidas que tiene el sistema.

Autor	Fecha	Versión
Martínez Zarate Yasbeth Mariana	27/11/23	4.0

5. PUNTOS DE VISTA DE LA COMPOSICIÓN

En este punto se expondrá de cómo se compone el sistema.

5.1. PROBLEMAS DE DISEÑO

Los problemas del diseño se establecen al generar la estructura del software y la asignación de roles para localizar y asignar funciones principales y responsabilidades.

Se utilizará el mantenimiento de software para tener un análisis profundo de impacto y entender los esfuerzos de cada cambio.

Se va abordar la reutilización de software, como se muestre en el diagrama. Hay funcionalidades o responsabilidades en las cuales se repite por los roles, esto ayudara a un mejor plan de trabajo, planificación y seguimiento del proyecto del software.

Adquiriendo esta información podemos gestionar un menor costo y menor esfuerzo en el calendario de organización del desarrollo del software.

Considerando lo anterior los problemas que podríamos experimentar respecto a como se compone el sistema se indican en el siguiente ejemplo:

- Problema de vinculación de datos en almacenamiento de base de datos.
- Mala interacción o interrumpida de las interfaces que lo componen.
- Sin acceso a sistema web por cuestiones de estructura.

Extracción incorrecta de la información capturada por código QR.

5.2. ELEMENTOS DE DISEÑO

Los elementos del diseño estarán presentes en cada momento del sistema siendo parte los tipos de conforman un sistema: subsistemas, componentes, módulos; puertos e interfaces (proporcionadas y requeridas); también bibliotecas, frameworks, repositorios de software, catálogos y plantillas.

Para la composición o creación del código estará compuesto de GitHub como ayuda de repositorio de los códigos y actualizaciones que se estén realizando en el sistema.

Uno de los elementos importantes es el lenguaje que se utilizara en al diseño en este caso será Python, de tal forma se utilizara Abstract Base Class. Y las bibliotecas correspondientes.

Es importante implementar una estructura con interfaces de diseño el cual esta determinara el diferente proceso que se va a realizar.

Frameworks:

Django es un framework de desarrollo web de código abierto, escrito en Python, que respeta el patrón de diseño conocido como modelo–vista–controlador.

XAMPP es un servidor independiente de plataforma, software libre, que consiste principalmente en la base de datos MySQL, el servidor web Apache y los intérpretes para lenguajes de script:

En el siguiente diagrama se especificará las fases o las interfaces de cada interacción con respecto a sus sistemas y componentes de cada interfaz.

En el siguiente diagrama UML se mostrar las actividades relacionadas y como se componen dependiendo de su actividad principal, donde finalmente se genera un almacenamiento en una base de datos.

5.2.1 ATRIBUTO DE FUNCIÓN

En esta entidad se validará los datos de entrada por cada función de atributos respectivamente por los roles que se establecerán en cada interfaz:

Usuario (afectación de datos de entrada):

- Usuario podrá tener acceso a métodos de datos de entrada relacionados a la solicitud de materiales.

Jefe y/o Subjefe (afectación de datos de entrada):

- Generar productos
- Activar producto.

- Desactivar producto.

5.2.2. ATRIBUTO DE SUBORDINADOS

La identificación de todas las entidades que componen esta entidad. El atributo subordinado identifica la relación "compuesto por" para una entidad. Esta información se utiliza para rastrear los requisitos hasta las entidades de diseño y para identificar las relaciones estructurales padre/hijo a través de un tema de diseño.

Es importante definir que los componentes se dividirán por procesos e interfaces diferentes, siendo así las operaciones distintas.

Jefe o Subjefe tendrán funciones diferentes y operaciones que afectan a la base de datos de forma diferente que se define a continuación:

Registro de productos a base de datos está compuesto por:

- Generar productos
- Activar producto.
- Desactivar producto.

Solicitar productos al almacén este compuesto:

- Solicitar productos a proveedor.
- Emitir reportes.
- Registro de datos del solicitante.

Emisión de reportes este compuesto por:

- Descargar PDF
- Buscar registros.
- Enviar correo de formato.

5.3. IDIOMAS DE EJEMPLO

Para describir los puntos de composición se tomaron como ejemplo un diagrama UML package diagram.

Autor	Fecha	Versión
Rubén Trejo Víctor	27/11/23	5.0

6. PUNTO DE VISTA LÓGICO

El propósito del punto de vista lógico será diseñar las implementaciones correctas de clases e interfaces con la relación correcta para cada estructura con los roles que participan entre sí. Por ejemplo.

1. Clases de productos: Se utilizarán varias clases en el sistema, como lo son los apartados de productos.
2. Clases de usuario: En este apartado se encontrarán los usuarios que tendrán acceso al sistema, estos tendrán ciertos permisos y restricciones en cuando al sistema.
3. Módulos de gestión de inventario: Los módulos que se encontrarán dentro del sistema serán cinco y contaremos como con la gestión de usuarios, la realización de inventarios, el registro de productos, etc.

Son ideas de clases de diseño que laboran y están relacionadas.

6.1. PROBLEMAS DE DISEÑO

Los problemas de diseño orientado en el punto de vista lógico es implementar un diseño que pueda ser reutilizable y manejable. El verdadero problema es seleccionar los métodos correctos para su utilización nuevamente.

Esto podría considerarse por el tipo de procesos en los cuales se repite o genera el mismo proceso, así poder utilizar mismas clases para reutilización.

6.2. ELEMENTOS DE DISEÑO

Entidades de diseño: AdministracionProductos, JefeAlmacen, tipo de datos: int, atributo, métodos: RegistroProductos, PedidoDeProductos, clase de asociación: AdministracionProductos.

Relaciones de diseño: Clase de productos y clases de usuario están relacionados, en las cuales dependen de las clases principales, el cual se implementará para que sea un diseño estratégico que pueda ser fundamental para aplicar la herencia y relación entre clases.

Atributos de diseño: Se generan roles como JefeAlmacen, SubJefe y usuario. El cual dependerá la visibilidad, cardinalidad, tipo, para cada rol.

Restricciones de diseño: restricciones de valor estarán establecidos por los roles esto para no tener afectación en la base de datos o el sistema, como las restricciones de exclusividad de relaciones.

En el siguiente diagrama se crea un diseño con interfaces y actividades de cada rol en un diagrama de clases para su entendimiento.

6.3. IDIOMAS DE EJEMPLO

Para el punto de vista lógico se utilizó un Diagramas de clases UML para su mayor entendimiento y comprensión del funcionamiento del sistema.

Autor	Fecha	Versión
Flores Trejo Víctor Rubén	30/11/23	4.0

7. PUNTOS DE VISTA DE LA DEPENDENCIA

Descripción de la arquitectura del software la cual debe de proporcionar una vista general de la estructura del sistema. Vemos como un sistema depende de otros componentes del software para su funcionamiento a esto le podemos denominar interdependencia de los diferentes componentes utilizados en el desarrollo del software.

Para nuestro proyecto implementare un diagrama de bloques el cual nos dará una representación visual de los principales componentes y conexiones entre ellos, este diagrama nos permite comprender la arquitectura del software.

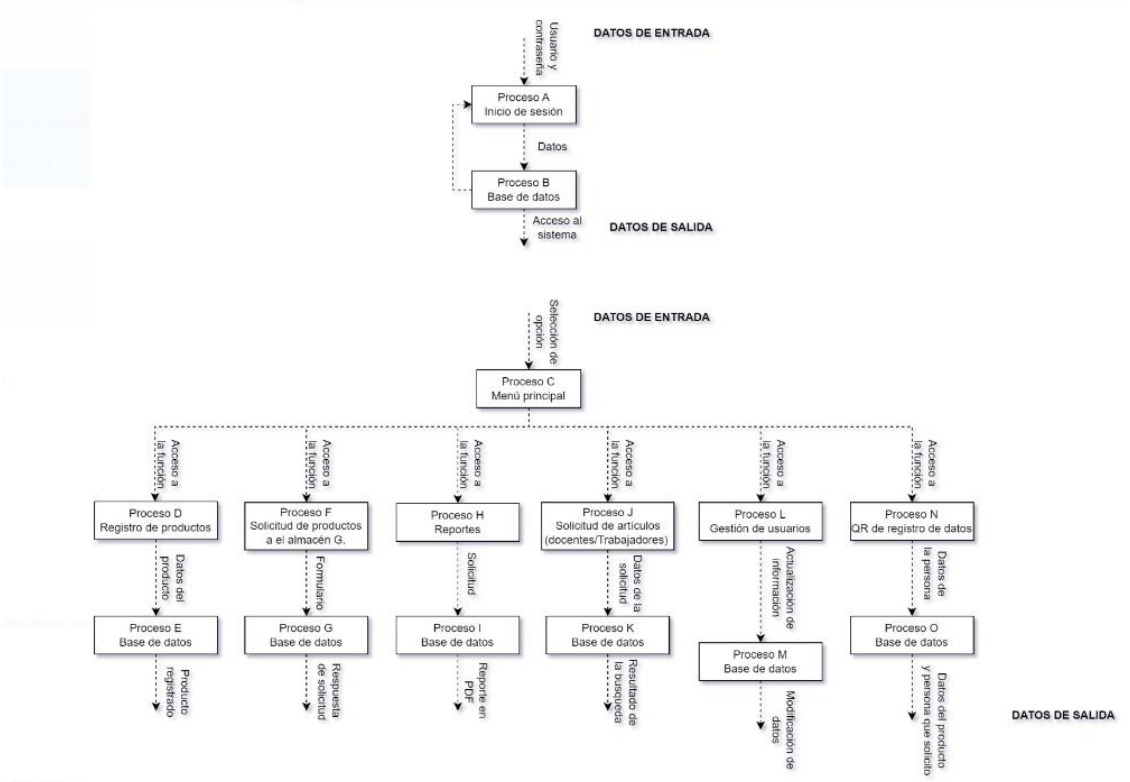


Diagrama de bloques general 7

El diagrama muestra como el

usuario interactúa con el sistema y este procesa la información para arrojarnos el resultado esperado. Vemos en parte la relación que existe en cada función.

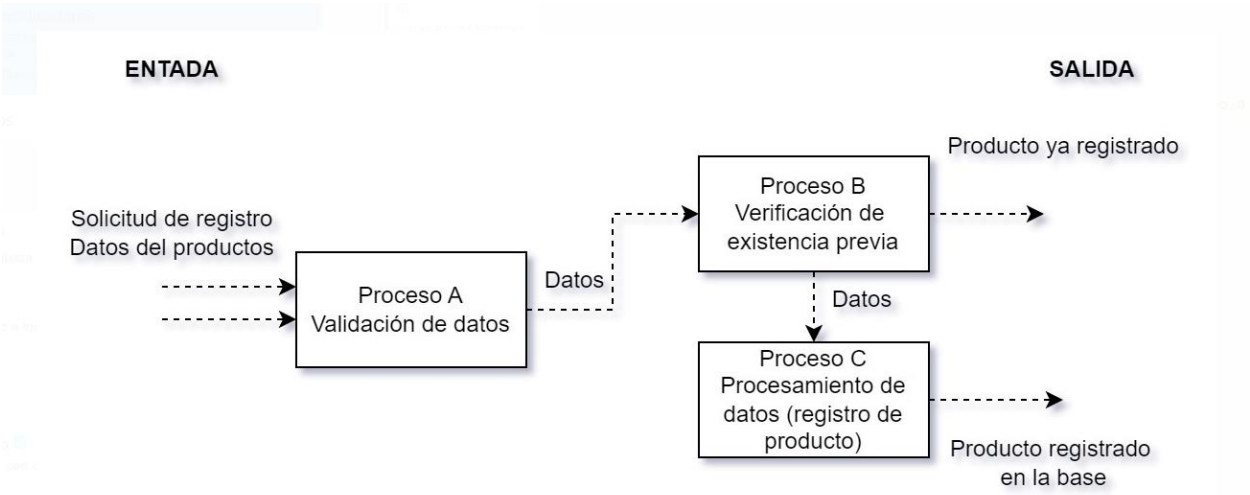


Diagrama de bloques "Solicitud de registro de productos" 8

Diagrama de bloques (Ilustración 8) de registro de productos donde vemos sus 2 entradas y como es que estas entradas se moldean en procesos o funciones y su respectiva conexión entre estas, las cuales nos llevan a un resultado (2 resultados finales en este caso).

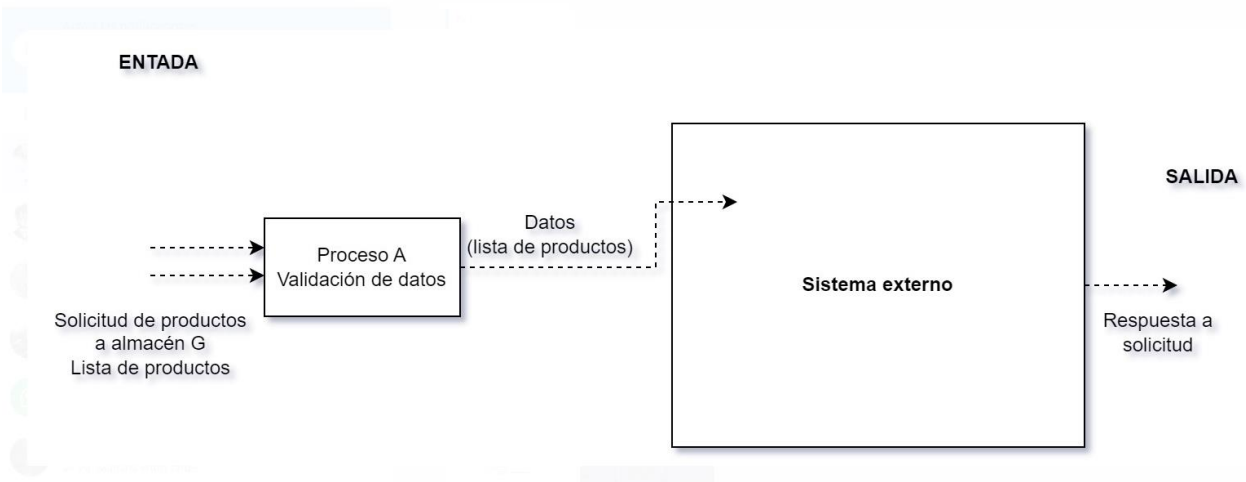


Diagrama de bloques "Solicitud de productos a almacén central" 9

Diagrama de bloques de solicitud de productos a almacén general (Ilustración 9) vemos de nuevo sus principales entradas que tiene y sus métodos o funciones por donde se procesa la información, en este caso vemos que tenemos un interfaz externo con bloques adicionales, el cual se sale de nuestro sistema principal y toma nuestros datos y los procesa para darnos el resultado de una salida.

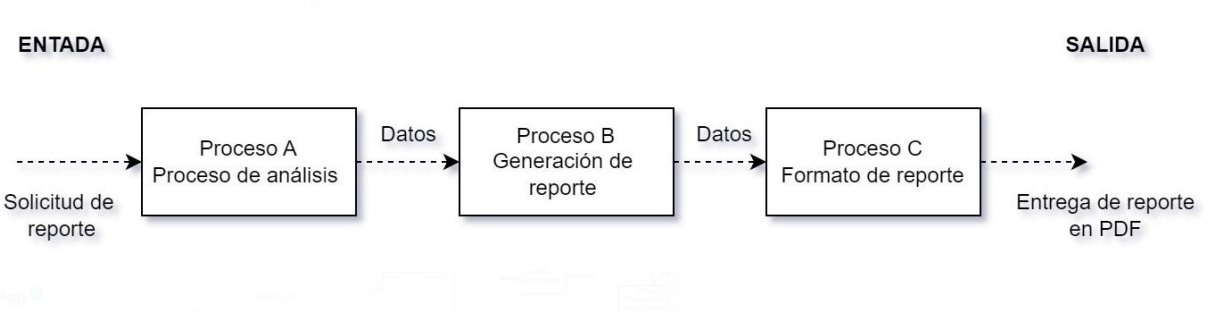


Diagrama de bloques "Generar reportes e inventario" 10

Diagrama de bloques de generación de reportes de inventario (Ilustración 10), en la cual solo necesitamos hacer una solicitud la cual sería nuestros datos de entrada, esta solicitud se procesaría en un análisis para hacer la consulta a nuestra base de datos con el informe, pasando a una generación de reporte en la cual se generan estadísticas en concreto de nuestros datos y por último la creación de este archivo en el formato que se estableció, dando como resultado de salida un informe en PDF.

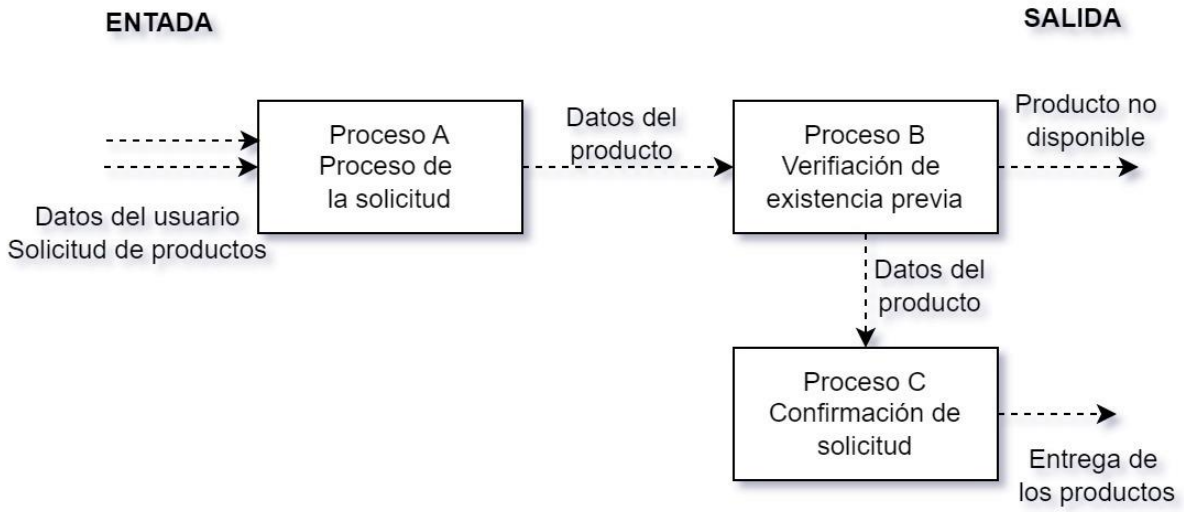


Diagrama de bloques "Solicitud de artículos" 11

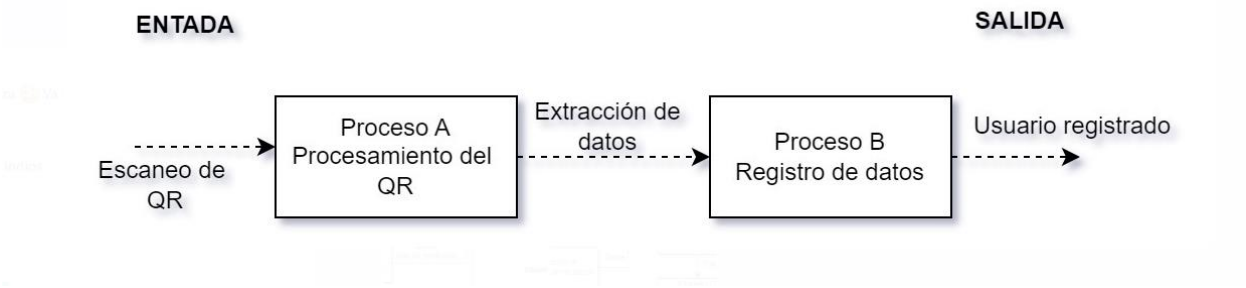


Diagrama de bloques "Solicitud de artículos" 12

Diagrama de bloques de solicitud de artículos (Ilustración 12), esto por parte de los alumnos y profesores el cual la entrada es la solicitud y los artículos, la cual se procesa para validar la información, se verifica la existencia de los productos y se hace la confirmación de la solicitud para la respectiva entrega de los artículos.

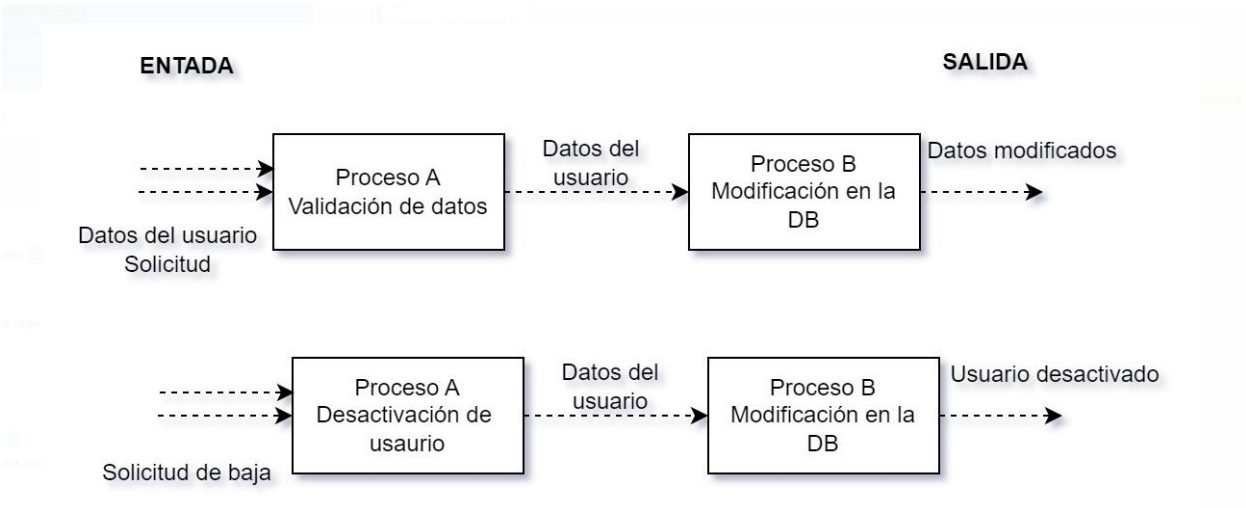


Diagrama de bloques "Gestión de usuarios" 13

Diagrama de bloques de gestión de usuario (Ilustración 13), en donde se divide en dos partes que sería la modificación de datos en la cual validamos esos datos y modificamos estos en la base y la desactivación de un usuario, que sería similar los módulos al de modificación solo que en este recibimos de entrada la solicitud.

7.1 PROBLEMAS DE DISEÑO

Es posible encontrar como problemas en el desarrollo de software en este caso falta de claridad debido a una complejidad excesiva con la utilización bloques y conexiones, es necesario eliminar y simplificar bloques innecesarios, ausencia de etiquetas claras la cuales pueden hacer difícil la comprensión del mismo y conexiones confusas que pueden llevar a un flujo erróneo y datos perdidos, esto son algunos de los problemas de diseño en este apartado.

7.2. ELEMENTOS DE DISEÑO

Proporcionamos una visión general de alto nivel del diseño del software, la arquitectura que nos indica cómo está organizada en los diferentes componentes, describiendo los módulos, también se debe de incluir el diseño de la base de datos y las interfaces de usuarios, esto con el fin de facilitar un diseño eficiente y efectivo.

7.2.1 ATRIBUTO DEPENDENCIAS

Describimos todas las dependencias identificadas y como estas afectan al diseño y funcionamiento del software, tales dependencias como (de hardware, software y entre módulos) son dependencias importantes ya que ayudan a mejorar la eficiencia y fiabilidad del software.

7.3 IDIOMAS DE EJEMPLO

En los idiomas de ejemplos se pueden incluir documentación como:

- Ejemplos de entrada y salida: donde se muestra como procesa los diferentes tipos de datos.
- Ejemplos de casos de uso: es una forma de demostrar cómo se utiliza el software en las diferentes situaciones.
- Diagramas de flujo: útiles para demostrar el flujo de datos en los procesos que se llevan a cabo.

Para este punto creamos un diagrama de comunicación.

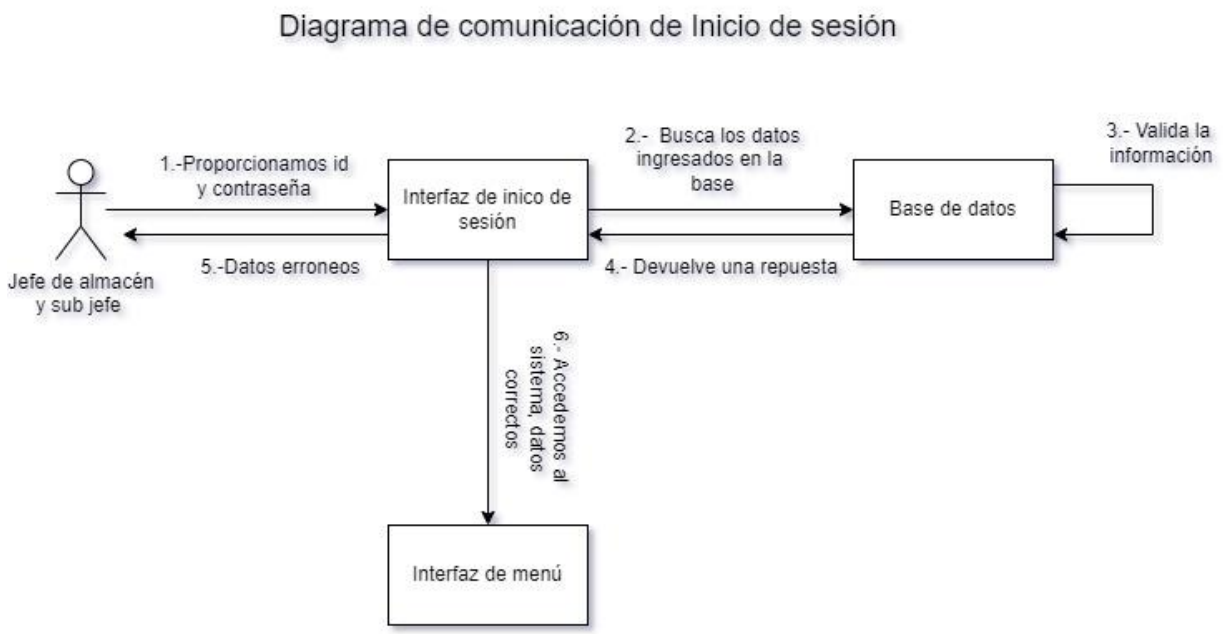


Diagrama de comunicación "Gestión de usuarios" 14

Representación del inicio de sesión (Ilustración 14), donde se ve la interacción de los usuarios con el sistema para poder acceder al sistema, se ejemplifica como se manda el mensaje y la secuencia de eventos que ocurren durante el inicio de sesión.

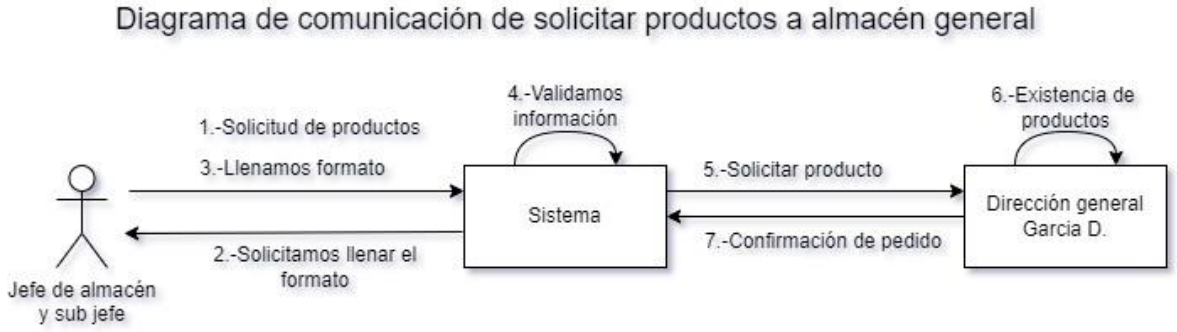


Diagrama de comunicación "Solicitud de productos a almacén central" 15

EL diagrama (Ilustración 15) proporciona la visualización de una solicitud de productos al almacén general en el cual se ve todo el proceso que conlleva la solicitud hasta el punto de respuesta de la misma y la entrega del producto.

Diagrama de comunicación "Registro de productos" 16

En este diagrama (Ilustración 16), se brinda la representación visual de la interacción entre el jefe de almacén y el sistema durante el proceso de registro de los productos.

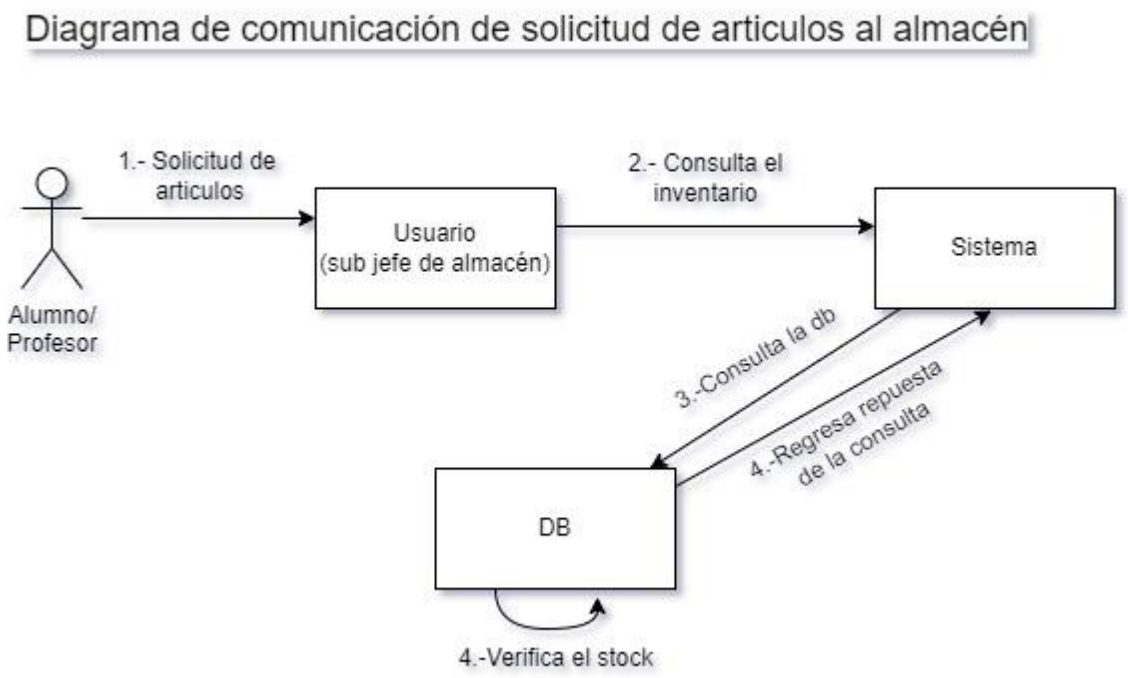


Diagrama de comunicación "Solicitud de artículos al almacén" 17

El diagrama (Ilustración 17) en este caso el usuario solo hace una petición a los encargados del almacén los cuales serán quienes sean responsables de hacer todo el proceso de verificación del producto, la entrega de dicho para responder a la solicitud de estos usuarios.

Diagrama de comunicación "Generar reportes e inventario" 18

El siguiente diagrama (Ilustración 18) es una representación de la interacción de un usuario con el sistema para hacer la consulta o generación de reporte, la cual se encargará el sistema por completo de todo el flujo hasta su generación.

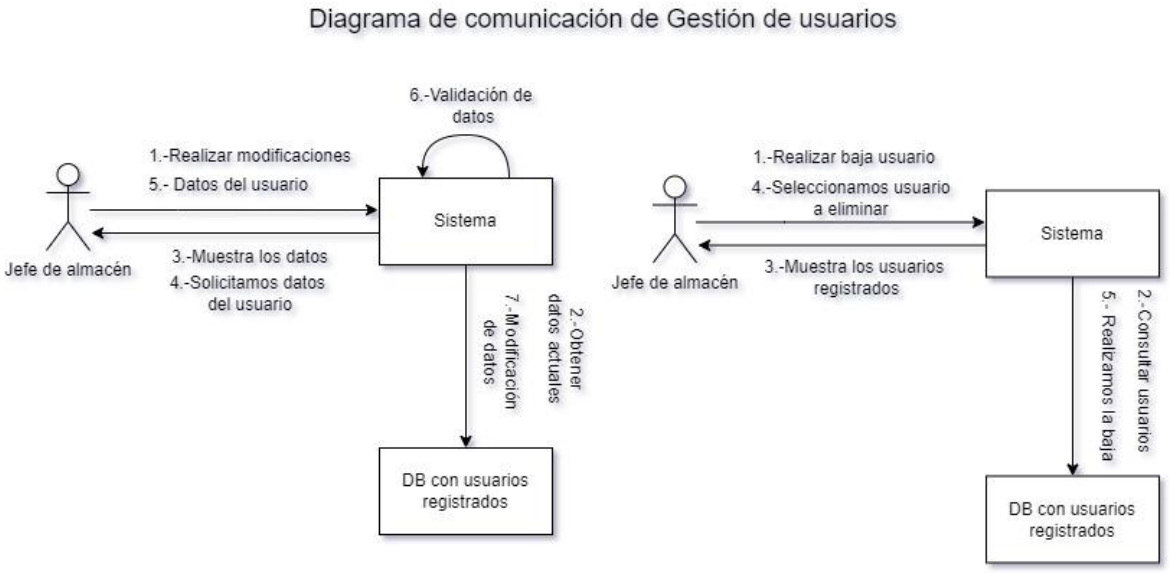


Diagrama de comunicación "Gestión de usuarios" 19

Diagrama de comunicación de la gestión de usuarios (Ilustración 19), en donde el jefe de almacén será la única persona que podrá modificar y/o actualizar los datos del usuario.

Autor	Fecha	Versión
Flores Trejo Víctor Rubén	30/05/23	4.0

8. PUNTOS DE VISTA DE LA INFORMACIÓN

En el cual se definen varios puntos de vista, los cuales nos describen información del sistema, uno de estos puntos es el (Punto de vista de la información). El cual se centra en el manejo y almacenamiento. Como se procesa la información en el sistema y como es que esta se asegura (integridad, confidencialidad y disponibilidad) lo cual para nuestro almacén es de vital importancia, ya que no cualquier persona puede tener acceso a los datos de los productos o de lo que nos llega.

8.1. PROBLEMAS DE DISEÑO

Pueden existir algunos problemas en el diseño como (falta de adecuación al contexto, dificultad para una documentación actualizada, exceso de documentación), duplicación de la información, relaciones inadecuadas de los datos, etc. Ojo el estándar no proporciona como diseñar la arquitectura de un almacén, como seleccionar el software de gestión de datos, su diseño y construcción, pero es importante que el diseñador del software tenga en cuenta estas limitaciones y complemente su diseño por su cuenta.

8.2. ELEMENTOS DE DISEÑO

Algunos elementos importantes del diseño son clave y deben de incluir (requisitos del sistema, descripción, arquitectura, diseño de módulos, base de datos, interfaces, etc.). En el software de almacén necesitamos un diseño detallado, su implementación, pruebas y conclusiones, esto para tener un acuerdo al estándar IEEE con respecto a nuestro almacén.

8.2.1. ATRIBUTO DE DATOS

Aspectos importantes que se deben de considerar, estos son: funcionalidad, fiabilidad, usabilidad, eficiencia, mantenimiento, etc. Esto garantiza a la hora de desarrollo de nuestro software un almacén que cumpla con los requisitos del cliente y funcionalidad eficaz y eficiente tanto a corto como a largo plazo, estos atributos son de vital importancia.

8.3. IDIOMAS DE EJEMPLO

Modelo de software que son comúnmente utilizado por los ingenieros para la documentación y diseño de este, estos pueden ser (UML, Diagrama de entidad relación, entre otros). Aquí desarrollaremos el diagrama de entidad relación.

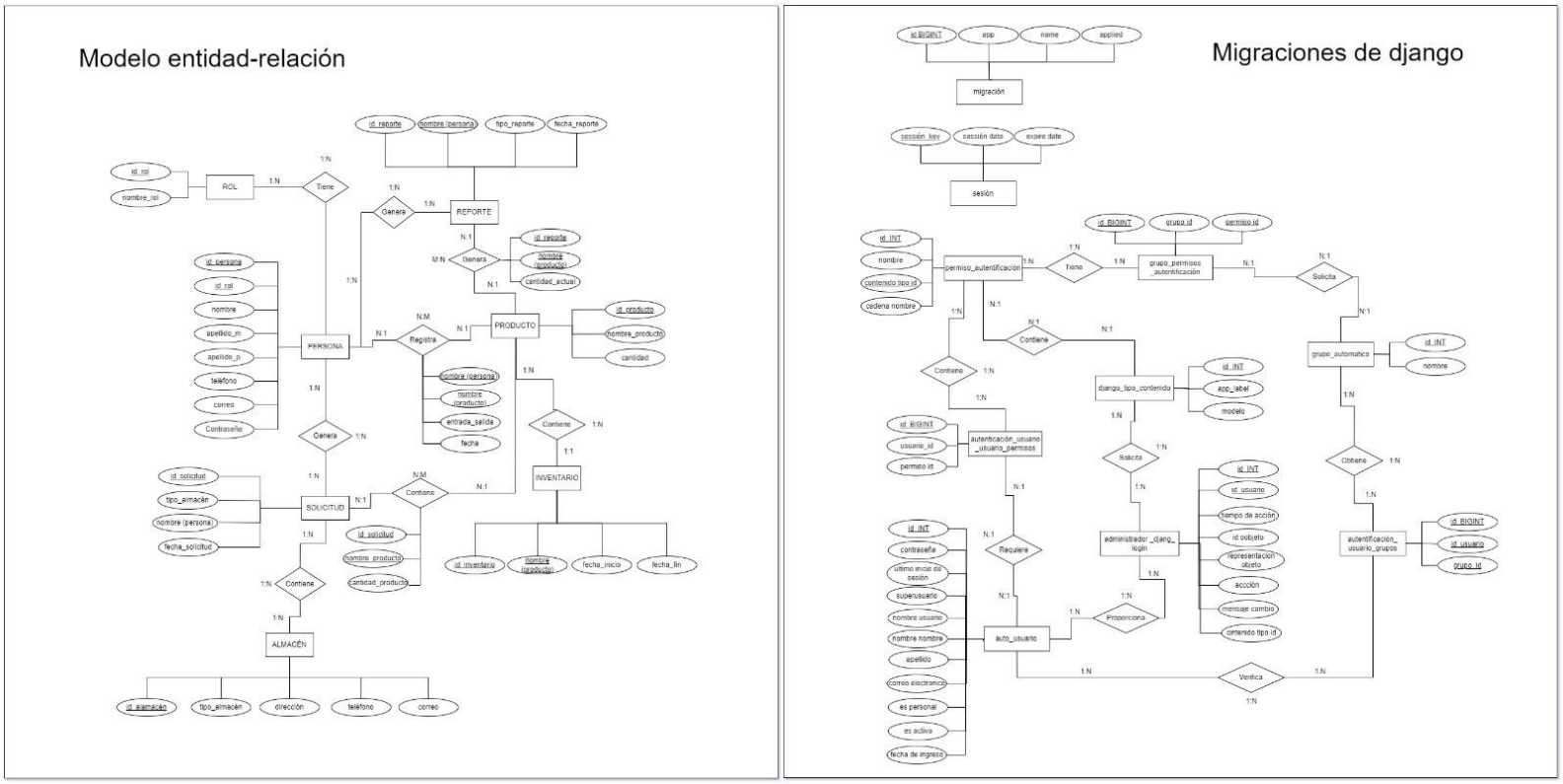


Diagrama "Entidad Relación" 20

En el siguiente diagrama (Ilustración 20) podemos ver las entidades y sus respectivos atributos, así como la relación que existe entre ellos para la gestión de nuestro almacén. Tenemos una visión más clara de cómo es que está estructurado y organizado, la cual nos sirve de guía para el diseño y la implementación de nuestra base.

Autor	Fecha	Versión
García de Jesús Héctor	30/11/23	4.0

9. PUNTO DE VISTA DEL USO DE PATRONES

En esta sección, se presentarán los patrones que se implementarán durante el funcionamiento del sistema. Estos patrones son soluciones probadas y efectivas para los problemas más comunes que pueden surgir durante las operaciones cotidianas. La implementación de estos patrones nos permitirá mejorar la calidad y eficiencia del software que estamos desarrollando. Al utilizar soluciones ya conocidas para ciertos problemas específicos, podemos evitar la necesidad de desarrollar soluciones desde cero, lo que nos permite centrarnos en la mejora continua de nuestro software.

Es de este modo, que los patrones que lleguemos a utilizar serán de tipo creacional y diseño que en esta sección nos proporcionará las herramientas necesarias para enfrentar los desafíos que puedan surgir durante el desarrollo y la operación de nuestro software. Con este conocimiento, podremos desarrollar un software robusto, eficiente y fácil de mantener. Así que, sin más preámbulos, comencemos con nuestra exploración de estos patrones y cómo pueden ayudarnos a alcanzar nuestros objetivos.

9.1. PROBLEMAS DE DISEÑO

Durante el desarrollo y la creación de este sistema se ha visto desde diferentes puntos de vista problemas que han podido retrasar o generar cuestiones que nos impiden resolver de una manera clara las situaciones que se nos presentan, es por ello que se busca la mejor manera de poder enfrentarse a dichas formas en este caso los patrones son siempre una buena opción para realizarlo, pero también se debe elegir aquellos que cumplan con la necesidad del sistemas por lo que a continuación se mencionan algunas cuestiones para una buena elección de estos:

- **Aplicación innecesaria de patrones**: Los patrones son muy útiles a la hora de poder realizar las tareas que se encarguen dentro del sistema, sin embargo, se deben de tomar en cuenta los cuales

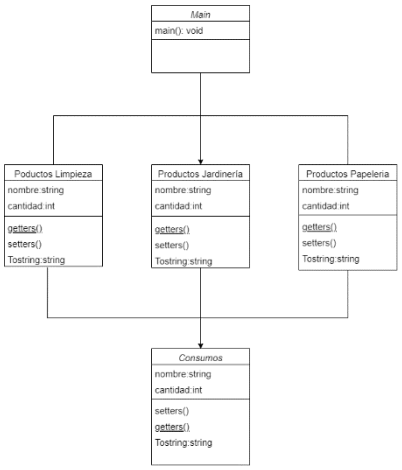
pueden aplicarse de manera correcta y en otro caso donde se aplican innecesariamente a un diseño, lo que puede hacer que el diseño sea más complejo de lo necesario, esto hará que el software sea más difícil de entender y mantener. Por ejemplo, dentro de este sistema lo que se buscara será la eficiencia del sistema y su seguridad, es por ello que se implementaran patrones como Singleton y Factory Methon. Que nos ayudaran precisamente a la organización adecuada de los datos.

- **Selección incorrecta de patrones:** La selección incorrecta de patrones puede hacer que el diseño sea menos eficiente o efectivo de lo que podría ser, por lo que podría a llegar a convertirse en una carga en lugar de una ayuda. Por eso siempre es importante tomar en cuenta a donde se aplica dicho caso, en este caso nuestro sistema va dirigido a un sistema educativo por lo que los patrones como Confirmshaming, son innecesarios.
- **Patrones mal implementados**: Si un patrón no se implementa correctamente, puede causar problemas de diseño, es por ello que debemos evitar cualquier nuevo patrón que no se conozca de manera adecuada, además de siempre revisar de manera adecuada la implementación y uso del mismo. En nuestro sistema a pesar de tener una funcionalidad útil, si el patrón Singleton no es abordado de manera adecuada puede llegar a realizar numerosas fallas al sistema.

9.2. ELEMENTOS DE DISEÑO

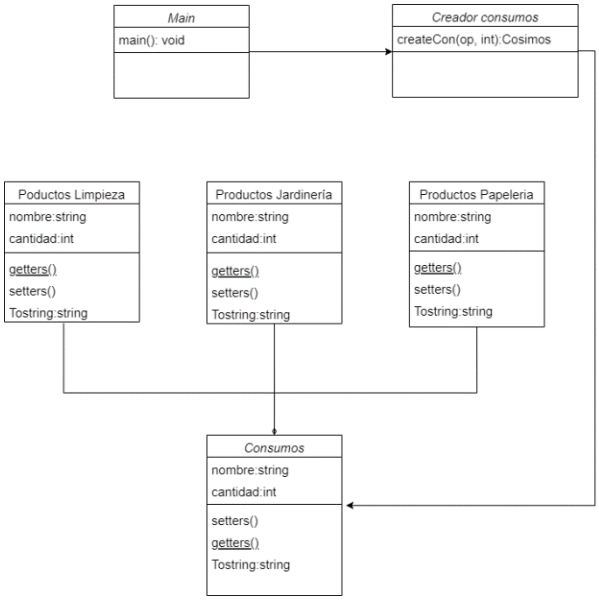
En este apartado se incluirán y se explicara el uso de los componentes del software que se diseñarán utilizando patrones. Es importante resaltar que estos patrones son aquellos que nos permiten un mejor manejo de los datos, además de ellos nos facilitan encontrar posibles fallas si es que el código tiende a volverse muy complejo.

1. Clases de productos: Dentro del sistema de gestión de los productos es importante mencionar que a la hora de una recepción de mercancía es importante separar y catalogar cada uno de ellos, por lo que se utilizarán varias clases en el sistema, como lo son los apartados de productos. Es de este modo que para esto se utilizara el patrón Factory Method, que nos ayudara crear una única clase, de las cuales se desplegaran otras que ayude a un mejor funcionamiento y ordenamiento, por ejemplo:

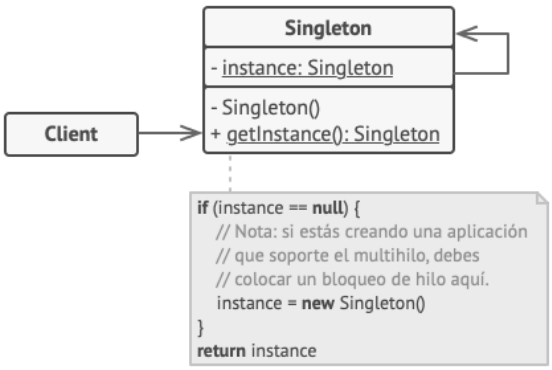


Vemos que la clase Main depende de tres clases, por lo que tenemos el problema de que nuestro núcleo de la aplicación, el componente Main, va a ser cada vez más complicado de escalar a medida que avanza la vida de la aplicación. Creando otro componente de este modo la entrada y salida al usuario de los datos y la creación de los objetos, podemos separar las funciones en funciones más concretas, pero esta entrada no pretende tratar sobre arquitectura de software sino sobre el patrón Factory, que consiste en crear un componente cuya función es la de la instanciar objetos.

Ahora tenemos una clase llamada CreadorConsumos que se encarga de instanciar las clases concretas en función de un parámetro que le pasa la clase Main.y de este modo esta ya no tiene que saber cómo crear el objeto, no es su función, sólo sabe que puede llamar al método de un componente que sí le devuelve un objeto.

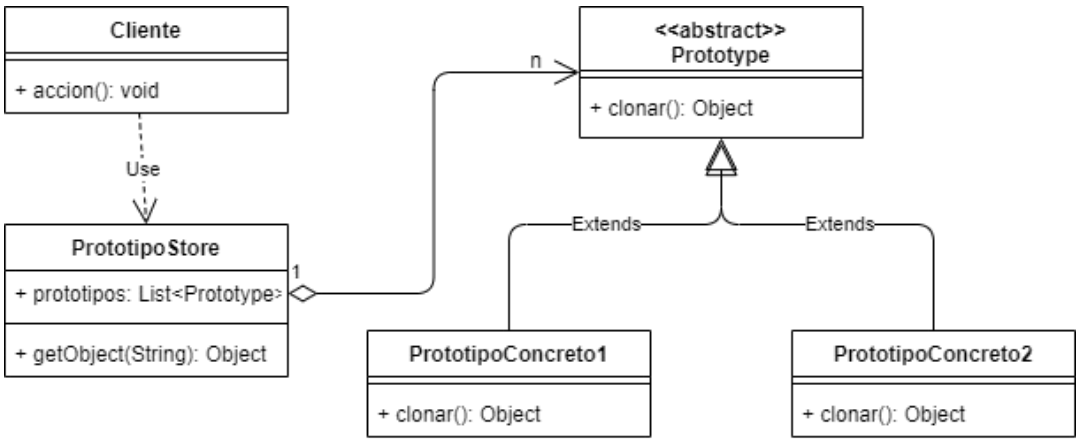


2. Clases de usuario: En este apartado se encontrarán los usuarios que tendrán acceso al sistema, estos tendrán ciertos permisos y restricciones en cuando al sistema. Por lo que de acuerdo a ello se optó por utilizar el patrón Singleton que nos ayuda a que en una única instancia se pueda conectar un usuario así como también la posibilidad mejorar la seguridad junto con el patrón Command, por ejemplo.



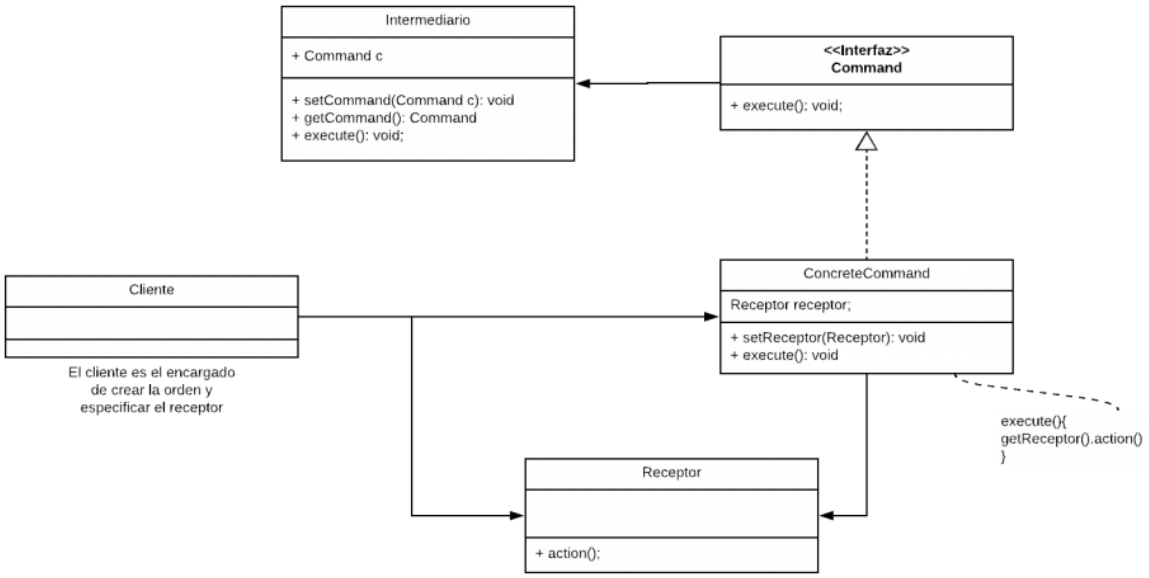
El diagrama muestra dos componentes principales: Cliente: Esta es la entidad que necesita acceder a la instancia del Singleton. Singleton: Esta es la clase que implementa el patrón Singleton. Tiene los siguientes elementos: -instance: Singleton: Este es un atributo privado que mantiene la única instancia de la clase. -Singleton(): Este es el constructor de la clase, que es privado para evitar que se creen nuevas instancias directamente. +getInstance(): Singleton: Este es un método estático que controla el acceso a la única instancia. Crea una nueva instancia si aún no existe y luego la devuelve.

3. Módulos de gestión de inventario: Los módulos que se encontrarán dentro del sistema serán cinco y contaremos como con el registro de personal, la realización de inventarios, el registro de productos, solicitud de productos almacén central y solicitud de productos al alancen. Bien para estos dos último se ha decido utilizar el patrón Command y el patrón Prototype que nos ayudaran a una mejor gestión en cuanto a los recursos que tenemos dentro del sistema, por ejemplo.



En el patrón Prototype tenemos:

Clase Cliente: Esta clase contiene un método llamado acción(). Esta es la clase que utiliza el patrón de prototipo. Clase PrototipoStore: Esta clase contiene una lista de prototipos y un método getObject(String). Este método se utiliza para clonar un prototipo existente en la lista. Clase Prototype: Esta es una clase abstracta que contiene un método clonar(). Este método se utiliza para clonar un objeto. Clases PrototipoConector1 y PrototipoConector2: Estas son clases concretas que extienden la clase abstracta Prototype y sobrescriben el método clonar(). El patrón de diseño de Prototipo se utiliza cuando la creación de un nuevo objeto es costosa (en términos de recursos y tiempo), y es más eficiente clonar un objeto existente lo cual nos ayudara mucho en el la reutilización de nuestro código a la hora de crear clases y diferentes instancias.



Por otro lado, tenemos el patrón command que nos ayudara en las solicitudes de peticiones tan como en el almacén central como en el local. Ejemplo:

La implementación de un código que permite la creación de peticiones por parte del usuario, su almacenamiento y posterior ejecución sobre un receptor. Nos permite desacoplar el objeto que dispara una acción del objeto que la realiza. Podemos deducir que existen 4 entidades que interactúan entre sí. El cliente que realiza la petición. La propia petición. Un elemento intermediario entre el cliente y el receptor. El receptor que se encarga de ejecutar la petición. El cliente es el sujeto que crea la petición en base a dos cuestiones: qué quiere hacer y sobre quién quiere hacerlo.

La petición es el elemento central del patrón. Contiene el código a ejecutar sobre el receptor y por ende, una referencia a este. El elemento intermediario es el encargado de almacenar las peticiones y provocar su ejecución cuando el cliente lo necesite, o cuando se considere oportuno por cualquier otro motivo. El receptor simplemente es el sujeto pasivo de la ejecución del comando disparado por el intermediario.

Diagramas de paquetes

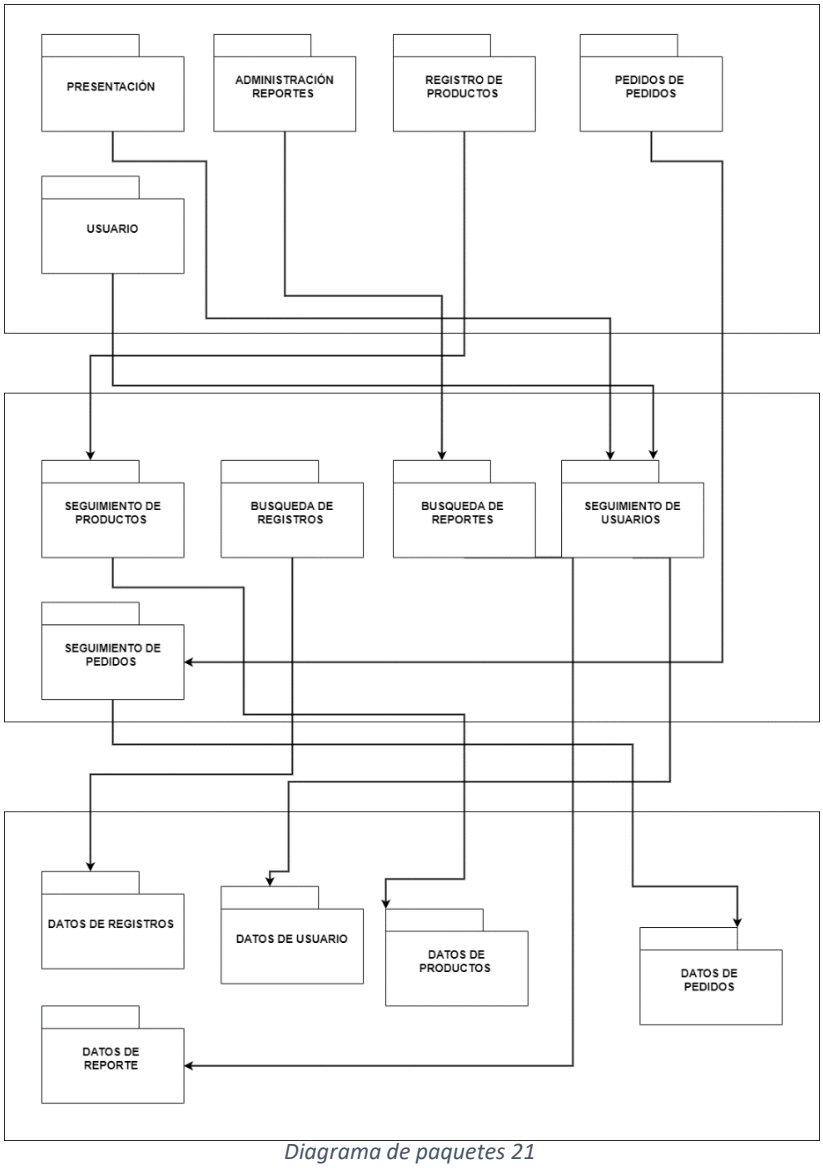


Diagrama de paquetes 21

DIAGRAMA DE PAQUETES ESPECIFICOS

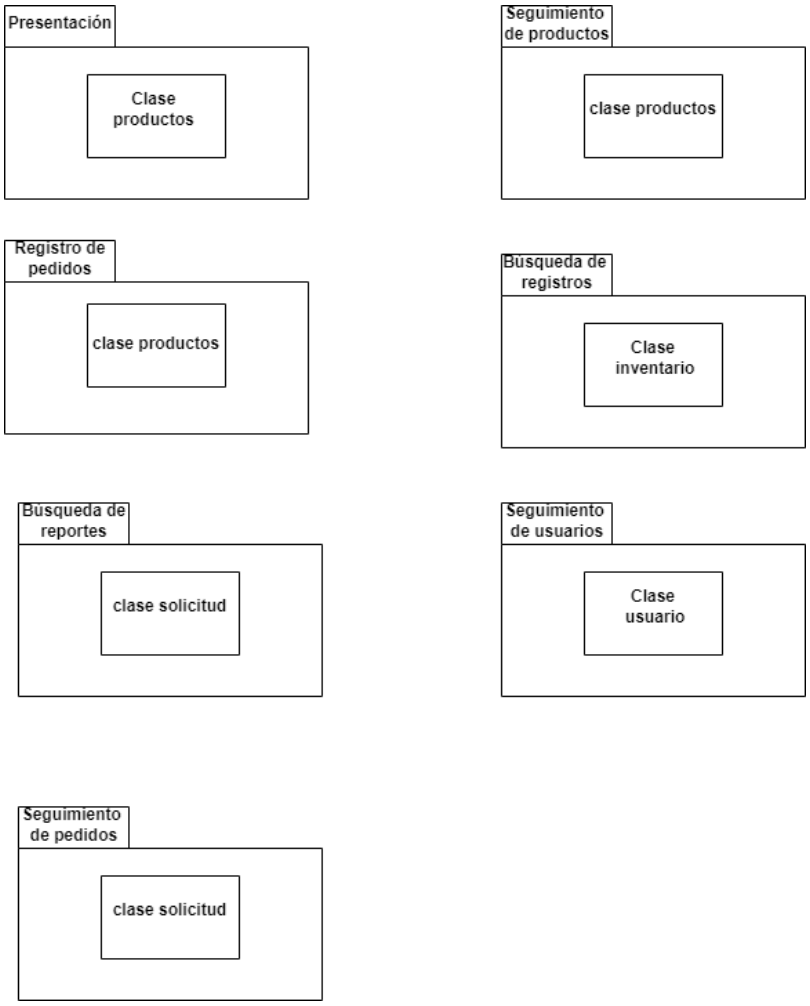


Ilustración 22

9.3. IDIOMAS EJEMPLO

Dentro de los idiomas que se pueden utilizar en este apartado se encuentran:

Diagrama de estructura compuesta: Un diagrama de estructura es un tipo de diagrama en el Lenguaje de Modelado Unificado, que muestra la estructura interna de una clase y las colaboraciones que esta estructura hace posibles.

Autor	Fecha	Versión
Martínez Zarate Yasbeth Mariana	28/11/23	5.0

10.PUNTO DE VISTA DE LA INTERFAZ

A continuación, se presenta una explicación para cada punto que contendrá el sistema de acuerdo a las características establecidas anteriormente con el cliente es por ello que se enfocará en las principales problemáticas de diseño.

1. *Interfaz de registro de usuarios:* Dentro de este apartado se dará la posibilidad de agregar de forma manual a los encargados dentro del almacén de la universidad, por lo cual se creará un ID de usuario, así como una contraseña. Es de este modo, que el usuario accederá al sistema únicamente con estas credenciales. La forma en que se realizará este proceso será dentro de una base de datos conectada a través de la aplicación que dará la consulta de los datos al ingresar dichas credenciales por el usuario.

2. *Interfaz de registro de productos:* Dentro de este apartado se encontrará una sección dedicada al registro de los productos que lleguen del almacén central, es así que dicha tarea estará a cargo de los responsables dentro del almacén de la universidad, por lo cual se creará una ID para cada producto. Facilitando la tarea de registro, y dando la posibilidad de agregarlos mediante un código QR. La forma en que se realizará este proceso será dentro de una base de datos (creada en My SQL) conectada a través de la aplicación que dará el almacenamiento de los datos al ingresar los códigos por el usuario.

3. *Interfaz de generador de reportes de inventario:* Dentro de este apartado se encontrará una sección dedicada a la realización de los inventarios dentro del almacén. Es decir que, por cada inventario, se dispondrá de un documento donde se refleje la información registrada con anterioridad por el encargado que cumplió dicha función. Con esto se podrá obtener un mejor manejo de los productos en stock, es de este modo el registro se realizará mediante el ingreso de los ides al sistema. La forma en que se realizará este proceso será dentro de una base de datos

(creada en My SQL) conectada a través de la aplicación que dará el almacenamiento de los datos al ingresar los códigos por el usuario.

4. *Interfaz de solicitud de artículos del almacén:* Dentro de este apartado se encontrará una sección dedicada a la petición de los productos existentes dentro del almacén. Dichos productos serán solicitados por los empleados (maestros, personal de limpieza, etc.). Es decir que al momento de requerir un suministro este se descontará del stock de los productos. Con esto se podrá obtener un mejor manejo de los productos como una repartición más ágil. La forma en que se realizará este proceso será dentro de una base de datos (creada en My SQL) conectada a través de la aplicación que dará el almacenamiento de los datos al ingresar los códigos por el usuario a través de códigos QR.

10.1. PROBLEMAS DE DISEÑO

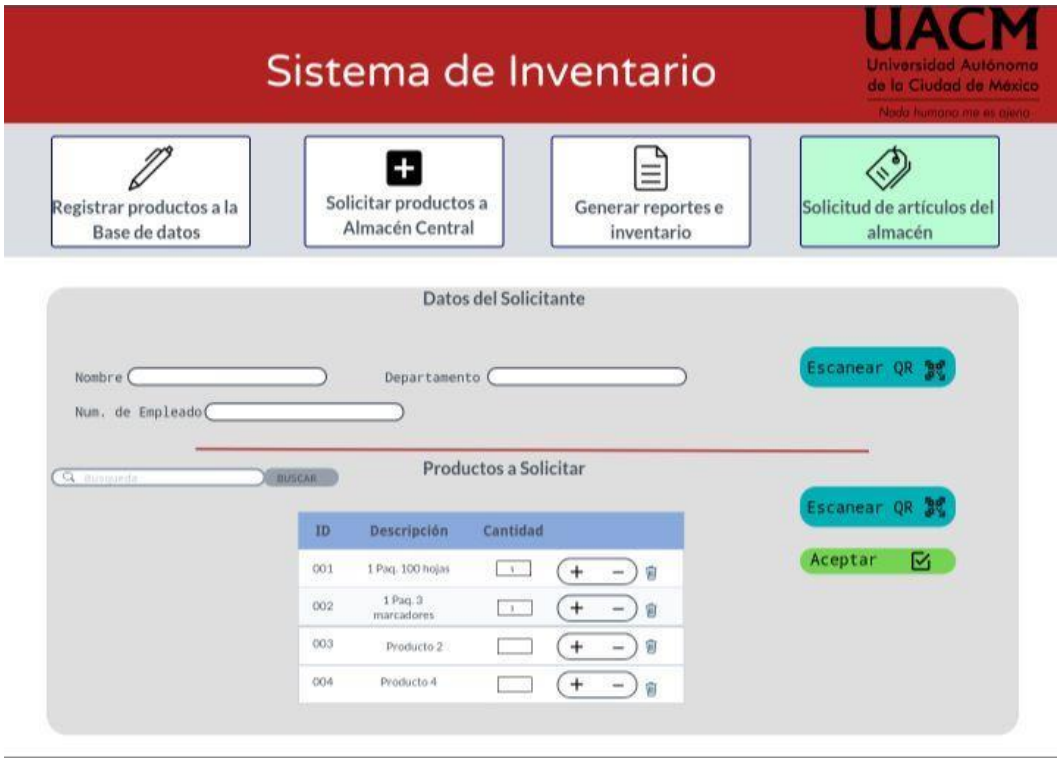
Primeramente, dentro de la realización de este proyecto se determinará la eficiencia de las interfaces y funcionamientos que componen a dicho sistema por lo cual se buscará un mejor manejo y gestión de los atributos requeridos. Es así que dado lo anterior el sistema puede que no contenga algunos requerimientos establecidos con el cliente.

Como es el caso de los métodos especiales, para el compartimiento de datos a través del sistema de base de datos, por ello se tomarán medidas que no lleguen a perjudicar al sistema y así tenga un buen funcionamiento.

10.2. ELEMENTOS DE DISEÑO

Interfaz general de sistema 23

En la imagen (Ilustración 24) se muestra la interfaz general del sistema de inventario, con cada uno de los módulos que se han propuesto en el proyecto.



Interfaz "Solicitud de artículos del almacén" 24

En este apartado (Ilustración 25) se encontrará como está conformado la interfaz de "Solicitud de artículos a almacén"

Interfaz "Generar reportes e inventario" 25

Se presenta el interfaz (Ilustración 26) del apartado de "Generar reportes e inventario.

Interfaz de "Registrar productos a la Base de datos" 26

En la imagen (Ilustración 27) se mostrará la interfaz de "Registrar productos a la Base de datos", esto como ya se mencionó se hará por medio del ID y la cantidad del producto a registra.

Interfaz "Solicitar productos a Almacén Central" 27

Dentro de esta imagen (Ilustración 28) se mostrará como está conformada la interfaz de "Solicitud de productos a Almacén Central".

10.2.1 ATRIBUTOS DE LA INTERFAZ

Los atributos que se caracterizan en este proceso de desarrollo, serán las conexiones que realizarán al sistema a través de la base de datos, con ello se dará las interacciones necesarias que se soliciten dentro de los distintos apartados del sistema.

Por ejemplo, en el caso de los apartados que requieran una repuesta del almacenamiento, es decir que en la situación de que algún encargado requiera una información de un producto o un personal este accederá a través del sistema conectando con la base de datos establecida. Además de la implementación de los códigos QR que nos ayudaran a mejorar los procesos administrativos realizados por los encargados.

10.3. IDIOMAS DE EJEMPLO

1. Diagrama de componentes: Es un diagrama tipo del Lenguaje Unificado de Modelado. Un diagrama de componentes representa cómo un sistema de software es dividido en componentes y muestra las dependencias entre estos componentes.

2. Lenguajes de definición de interfaz (IDL): El lenguaje de descripción de interfaz o también lenguaje de definición de interfaz es un lenguaje informático utilizado para describir la interfaz de componentes de software.

Autor	Fecha	Versión
García de Jesús Héctor	27/11/23	4.0

11.PUNTO DE VISTA DE ESTRUCTURA

En el desarrollo de este sistema, se utilizarán diversas herramientas para facilitar su creación y funcionamiento. Un ejemplo destacado es Python, un lenguaje de programación versátil y potente. Para el desarrollo de las interfaces de usuario, se utilizará Django, una extensión de Python que proporciona un marco de trabajo robusto para el desarrollo web. Además, se utilizará JavaScript para mejorar la interactividad y funcionalidad de las páginas web.

En este apartado, se describirán los elementos clave que se desarrollarán para este sistema. Estos incluyen:

- **Botones de entrada y salida**: Permitirán al usuario interactuar con el sistema, ingresando y saliendo de diferentes secciones.
- **Botones de acción**: Facilitarán la realización de acciones específicas dentro del sistema.
- **Colores significativos:** Se utilizarán para mejorar la experiencia del usuario, proporcionando retroalimentación visual y haciendo que la interfaz sea más intuitiva.
- **Botones de desplazamiento**: Permitirán al usuario navegar a través de diferentes secciones del sistema.
- **Interfaces interactivas:** Mejorarán la experiencia del usuario, haciendo que el sistema sea más atractivo y fácil de usar.
- **Campos de texto**: Permitirán al usuario introducir información en el sistema.
- Información sobre los apartados: Proporcionará detalles adicionales sobre las diferentes secciones del sistema.
- **Herramientas de ayuda:** Asistirán al usuario, proporcionando orientación y soporte cuando sea necesario.

11.1. PROBLEMAS DE DISEÑO

El sistema de gestión de inventarios, como componente de grano grueso, se puede dividir en componentes más pequeños, como la interfaz de usuario, la base de datos de productos y la lógica de negocios. La reutilización de estos componentes de grano fino en diferentes partes del sistema

permite la creación de un sistema de gestión de inventarios eficiente y escalable, sin tener que desarrollarlos desde cero.

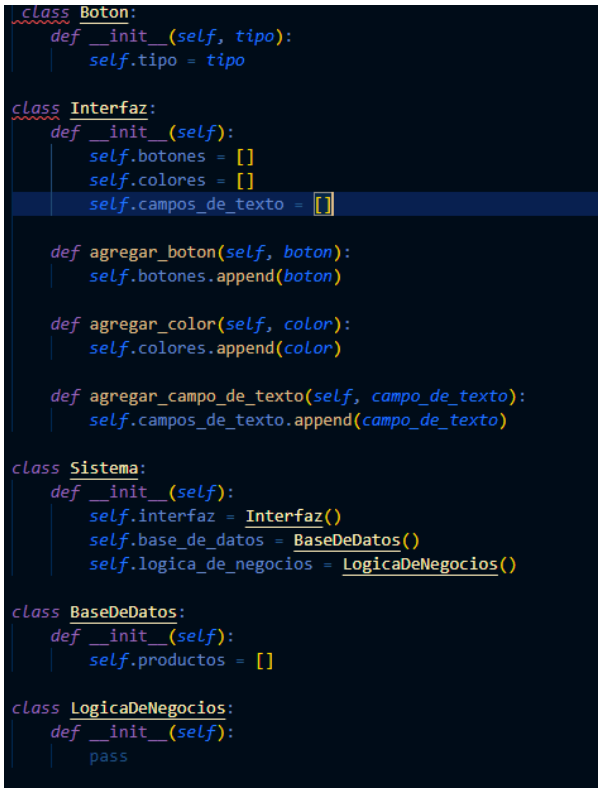
11.2. ELEMENTOS DE DISEÑO

Los elementos de diseño incluyen entidades, relaciones y atributos. Las entidades de diseño son los componentes individuales del sistema, como los puertos, conectores y clases. Las relaciones de diseño describen cómo se conectan estas entidades entre sí. Los atributos de diseño proporcionan detalles adicionales sobre cada entidad, como su nombre, tipo, propósito y definición.

11.3. IDIOMAS DE EJEMPLO

Los idiomas de ejemplo incluyen el Diagrama de componentes UML y el Diagrama de clases UML. Estos son lenguajes de modelado visual que se utilizan para diseñar software, describiendo los componentes de un sistema y cómo se relacionan entre sí.

Por ejemplo, aquí tenemos un pequeño ejemplo sobre el código de las clases.



```python
class Boton:
    def __init__(self, tipo):
        self.tipo = tipo

class Interfaz:
    def __init__(self):
        self.botones = []
        self.colores = []
        self.campos_de_texto = []

    def agregar_boton(self, boton):
        self.botones.append(boton)

    def agregar_color(self, color):
        self.colores.append(color)

    def agregar_campo_de_texto(self, campo_de_texto):
        self.campos_de_texto.append(campo_de_texto)

class Sistema:
    def __init__(self):
        self.interfaz = Interfaz()
        self.base_de_datos = BaseDeDatos()
        self.logica_de_negocios = LogicaDeNegocios()

class BaseDeDatos:
    def __init__(self):
        self.productos = []

class LogicaDeNegocios:
    def __init__(self):
        pass
```

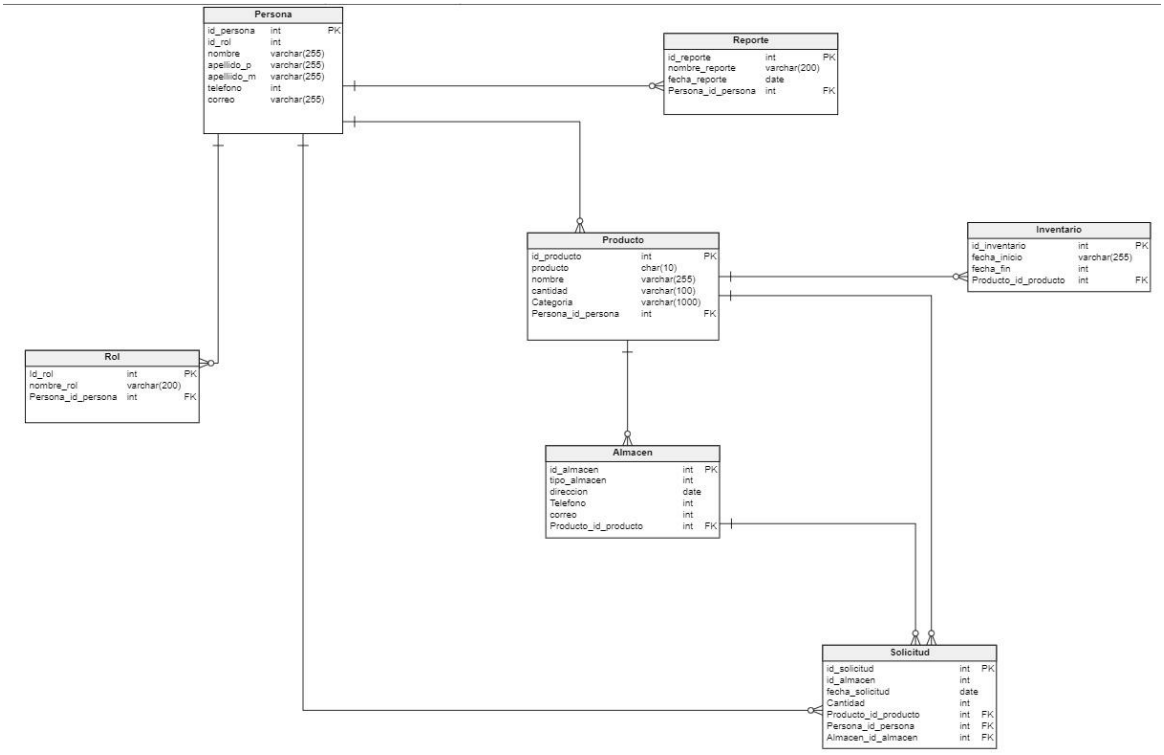


Diagrama de clases 28

Autor	Fecha	Versión
Martinez Zarate Yasbeth Mariana	28/05/23	3.0

12.PUNTO DE VISTA DE INTERACCIÓN

¿Por qué?: Los usuarios podrían utilizar el sistema para mejorar el ordenamiento de los recursos dentro del almacén de la universidad, de manera que sea más conveniente y eficiente para ellos.

¿Dónde?: La interacción tendría lugar en una plataforma en línea, como un sitio web.

¿Cómo?: los usuarios podrían interactuar a través de dicha plataforma ofreciéndoles una mayor comodidad en cuanto a los registros que se realizan en las diferentes áreas de trabajo del almacén.

En qué nivel: la interacción ocurriría en varios niveles, incluyendo la interacción entre el usuario y la interfaz gráfica, la interacción entre el usuario y el sistema de búsqueda de productos, y la interacción entre el sistema de realización de inventarios, además de los sistemas de administración.

12.1. PROBLEMAS DE DISEÑO

Los diseñadores tendrán que evaluar la asignación de responsabilidades entre las diferentes entidades involucradas en el sistema, como los pedidos de transporte, los sensores de inventario y los sistemas de gestión de pedidos. Finalmente, los diseñadores tendrán que considerar la lógica de transición de estado y concurrencia para garantizar que el sistema pueda manejar situaciones en tiempo real, como la detección de nuevos pedidos o la reasignación de tareas en caso de fallos o retrasos en alguna parte del sistema.

12.2. ELEMENTOS DE DISEÑO

Clases

Producto: representa un producto en el inventario del almacén y tiene atributos como nombre, descripción y cantidad disponible.

Métodos

AgregarProducto(): permite agregar un nuevo producto al inventario del almacén. Real

EliminarProducto(): permite eliminar un producto del inventario.

Estados: Abierto: el almacén está abierto y los clientes pueden pedir productos. Cerrado: el almacén está cerrado y no se pueden realizar pedidos.

Eventos: Asignación de producto: un usuario solicita un producto del inventario.

Jerarquía: encargados del almacén: es responsable de mantener un inventario preciso y asegurarse de que los usuarios reciban de manera adecuada los productos solicitados.

Temporización: El almacén puede estar abierto durante horas específicas del día, como de 9am a 7pm. Las tareas de inventario pueden programarse para llevarse a cabo en horarios específicos para evitar interrupciones en los pedidos.

Sincronización: Los cambios en el inventario deben sincronizarse en tiempo real para garantizar que los usuarios no soliciten productos que ya no están disponibles.

DIAGRAMAS DE SECUENCIA

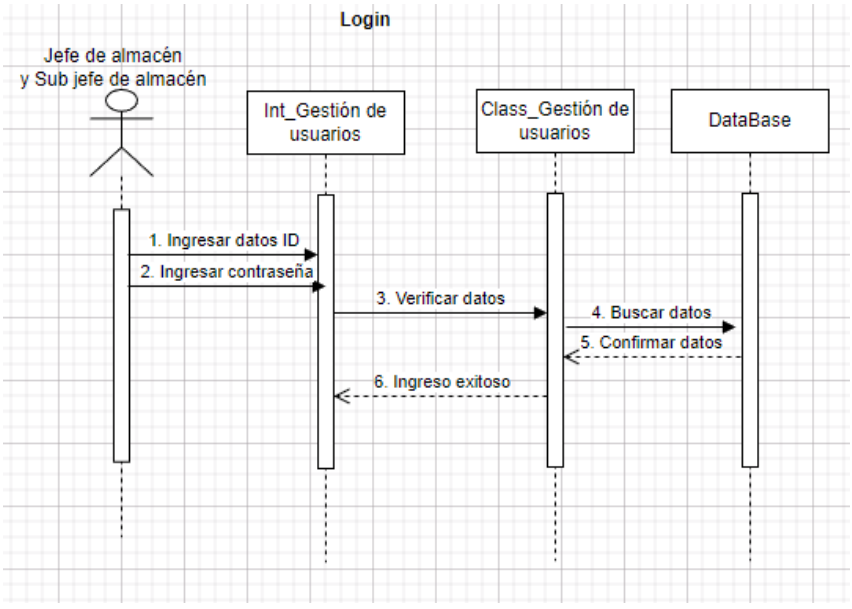


Diagrama de secuencia "Login" 29

En la ilustración 29 se nos muestra el proceso para la función de "Login" el cual constará de dos actores quienes serán encargados de validar al sistema su ingreso por medio de un ID (se le asigno

antes) junto con su contraseña. Constará de una secuencia de pasos para poder realizar lo pedido, en este caso se ingresará el ID y contraseña en la pantalla de iniciar sesión, después se verificarán los datos en la Data Base (donde se almaceno el ID y contraseña), una vez verificada devolverá la confirmación de datos y así se autenticarán para poder ingresar al sistema.

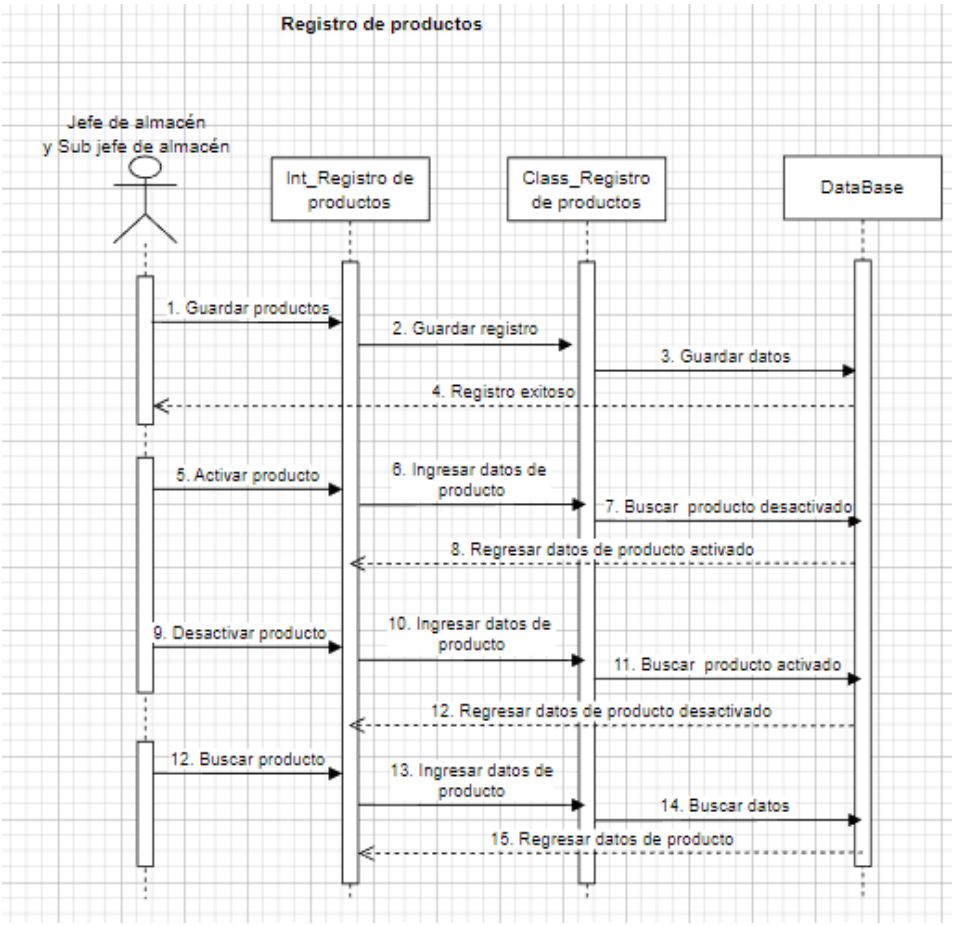


Diagrama de secuencia "Registro de productos" 30

En la ilustración 30 se nos mostrara la secuencia de pasos que deben de intervenir para el registro de productos. Aquí encontraremos a dos actores que serán encargados de que esta secuencia se haga de la forma correcta, vemos que estarán cuatro procesos (guardar producto, activar producto, desactivar producto y buscar producto) acciones que se podrán realizar por medio de varias

secuencias. Todas las acciones a realizar estarán conectadas a una base de datos para que su información pueda ser verificada como se muestra en la imagen.

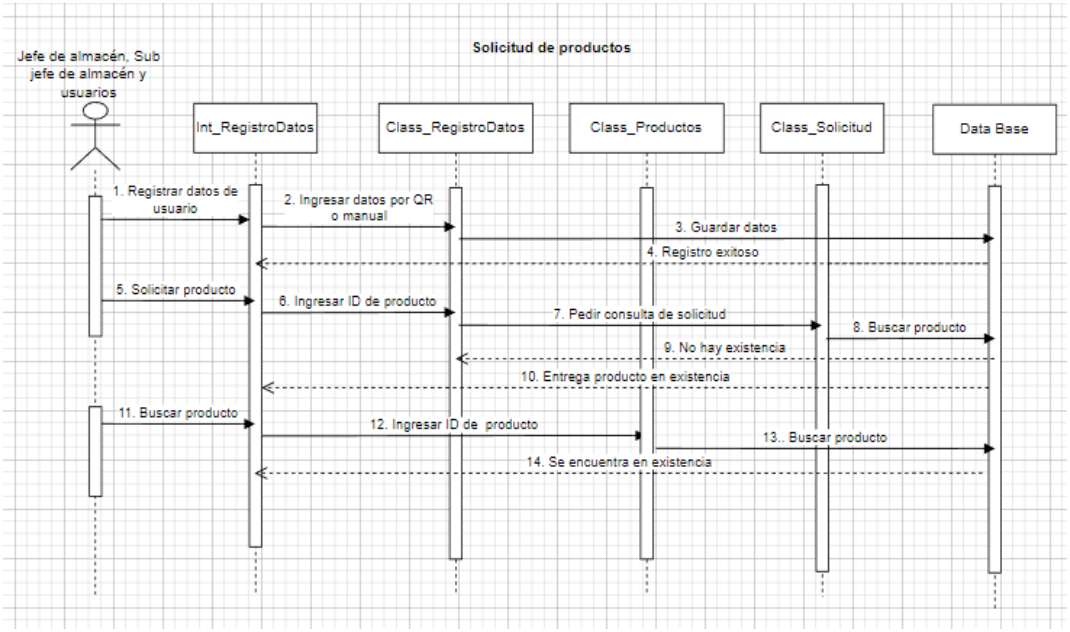


Diagrama de secuencia "Solicitud de artículos del almacén" 31

En la ilustración 31 esta nuestro diagrama de secuencia "Solicitud de artículos del almacén" en este apartado se encontrarán a tres actores, los cuales serán encargados de que la secuencia se haga de forma correcta, como en la imagen pasada aquí también contaremos con un base de datos para poder verificar algunos datos, en este caso será el registro de los datos del "usuario" los cuales constaran de los profesores y personal de limpieza, otra de las funciones serán el solicitar producto y buscar dicho producto, para esta secuencia se deberá ingresar primero el ID del producto y consultar en la base de datos si se encuentra disponible o no.

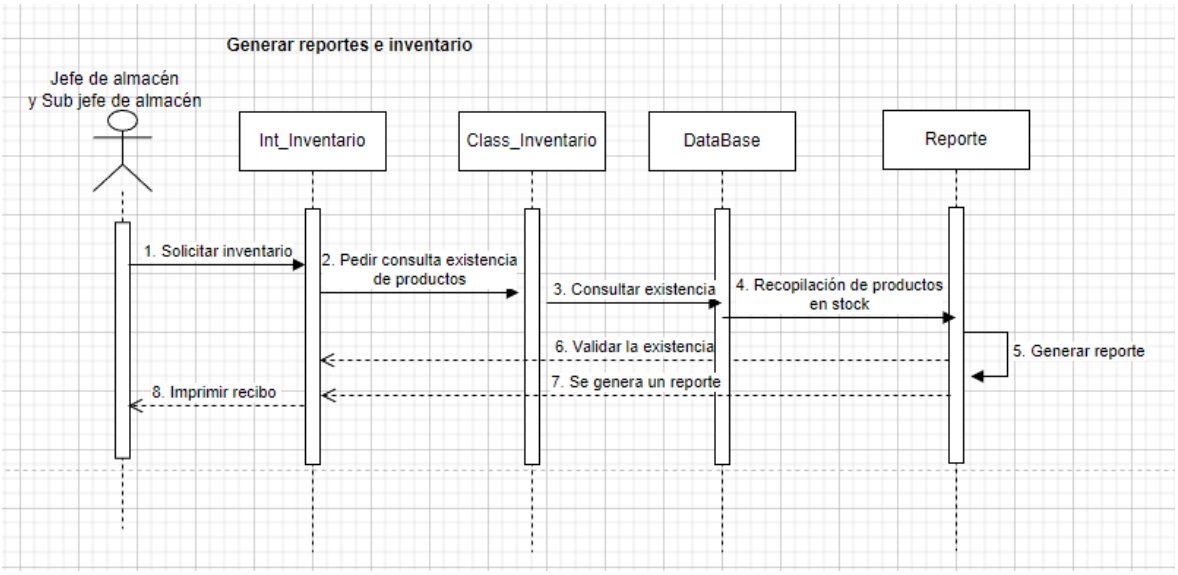


Diagrama de secuencia "Generar reportes e inventario" 32

En el diagrama (Ilustración 32) se muestra el módulo "Generar reportes e inventario" el cual constara de dos actores quienes podrán tener interacción con la interfaz de inventario y reporte, este en mayor parte funcionara por medio de la base de datos que es la que guardará toda la información de los pedidos, todo esto será por medio de un intervalo de tiempo que ingresara el actor para saber o imprimir el reporte o inventario de dichas fechas.

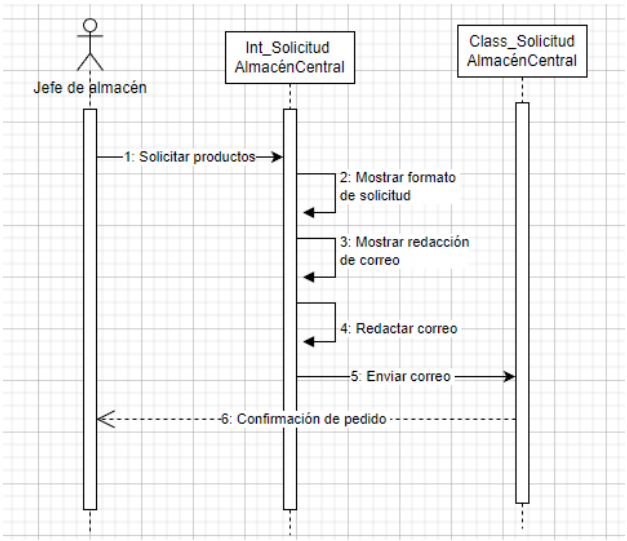


Diagrama de secuencia "Solicitar productos almacén central" 33

En la ilustración 33 se muestra el diagrama de secuencia del módulo de "Solicitar productos a almacén central" el cual constará de solo un actor, este será encargado de mandar la solicitud de

productos por medio de un formato que será asignado y luego enviado por correo electrónico al almacén central de la UACM (proveedor).

Autor	Fecha	Versión
Daniela Judith Robles Vásquez	28/11/23	3.0

13.PUNTO DE VISTA DE LA DIMANICA DE ESTADOS

En este apartado nos referiremos a un enfoque para modelar el comportamiento de sistemas complejos. Esta técnica se basa en la idea de que los sistemas pueden ser descritos como una colección de estados, donde cada estado representa una condición o situación particular en el sistema.

Se identificarán los diferentes estados posibles del sistema y se describen las transiciones entre ellos. Las transiciones pueden ser causadas por eventos externos o internos y pueden llevar al sistema a un nuevo estado.

13.1. PROBLEMAS DE DISEÑO

Los diagramas de estado son herramientas útiles usadas para modelar el comportamiento dinámico de un procedimiento o caso de uso haciendo énfasis en el proceso que se lleva a cabo. Estos diagramas son uno de los mejores comprendidos, ya que son herederos directos de los diagramas de flujo, de la misma forma heredan características de los diagramas de estado, diagrama de flujo de datos y de las redes de Petri. Aunque estos diagramas son comprensibles también conllevan algunos problemas en el tiempo del diseño, por ejemplo:

- ✓ Ambigüedad: Pueden ser ambiguos si no se especifica claramente el significado de los estados y las transiciones. Esto puede llevar a diferentes interpretaciones del modelo y, por lo tanto, a errores en el diseño o implementación del sistema.
- ✓ Ambigüedad: los diagramas de estado pueden ser ambiguos si no se especifica claramente el significado de los estados y las transiciones. Esto puede llevar a diferentes interpretaciones del modelo y, por lo tanto, a errores en el diseño o implementación del sistema.

- ✓ Dificultad para mantener la consistencia: los diagramas de estado pueden ser difíciles de mantener si hay cambios en los requisitos o en el comportamiento del sistema.
- ✓ Complejidad: los diagramas de estado pueden volverse muy complejos cuando se modelan sistemas grandes y complejos. Esto puede hacer que sea difícil entender el comportamiento del sistema y realizar cambios en el modelo.

13.2. ELEMENTOS DE DISEÑO

DIAGRAMAS DE ESTADOS

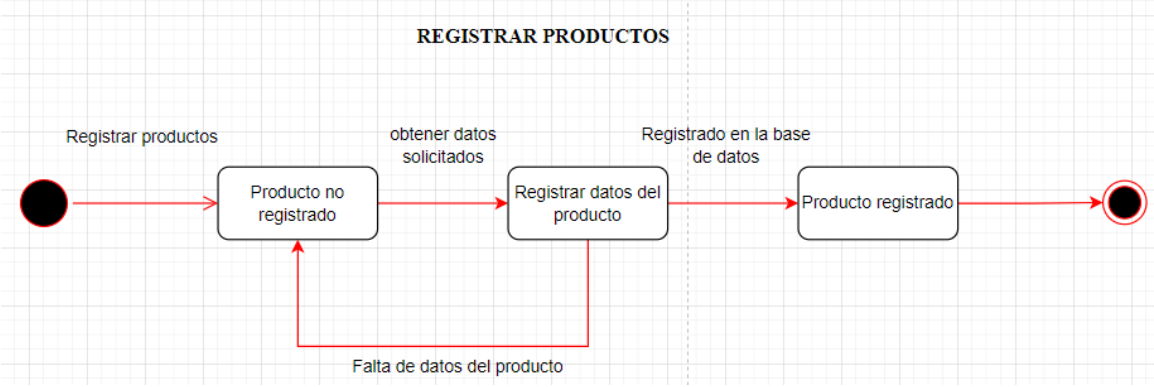


Diagrama de estado "Registrar productos a la base de datos" 34

En el diagrama de estado (Ilustración 34) se muestra el comportamiento de nuestro objeto (Registrar productos a la Base de Datos), con el cual llevaremos a un producto no registrado a pertenecer a nuestra base de datos y estar registrado, en caso de que este ya esté registrado se cancelaria la operación.

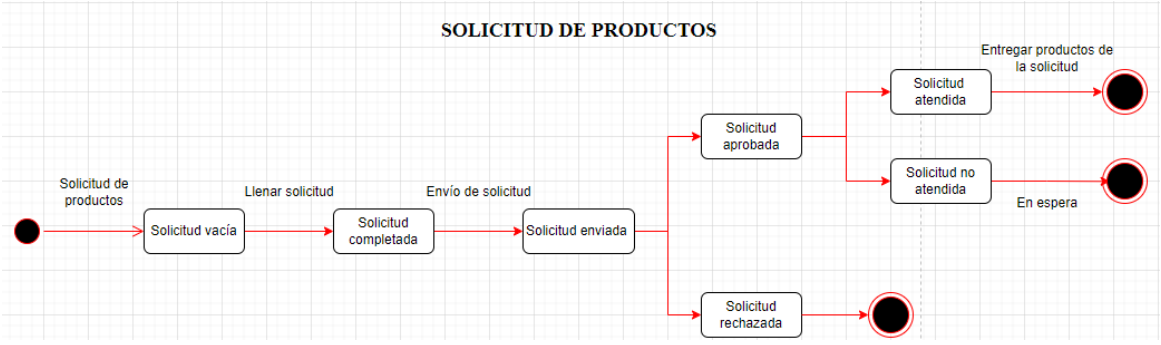


Diagrama de estado "Solicitud de productos" 35

En el diagrama de estado (Ilustración 35) se muestra el comportamiento de nuestro objeto (Solicitud de artículos del almacén), con el cual se muestra el proceso que conlleva realizar una solicitud para que los productos que se requieren puedan ser validados.

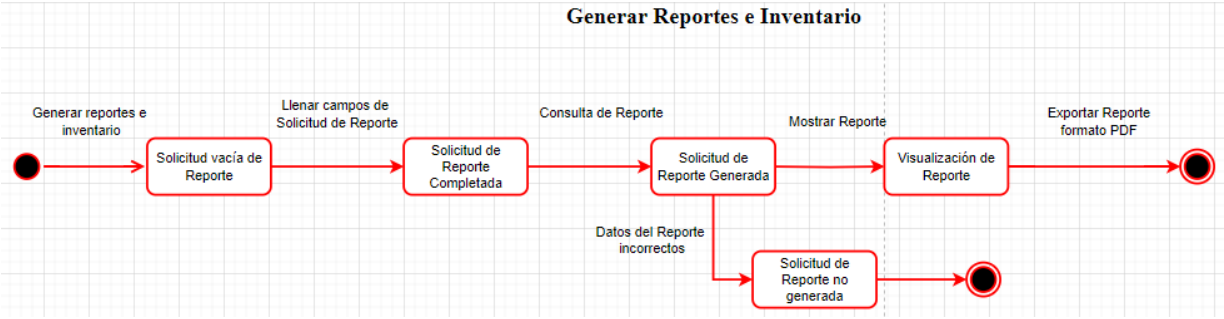


Diagrama de estado "Generar reportes e inventario" 36

En el diagrama de estado (Ilustración 36) se muestra el comportamiento de nuestro objeto (Generar Reportes e Inventario), con el cual se muestra el proceso que conlleva generar reportes del inventario para que un reporte pueda ser validado y exportado correctamente.

13.3. IDIOMAS DE EJEMPLO

Los idiomas de ejemplo de diagramas de estados son lenguajes formales que se utilizan para describir los estados y transiciones de un sistema en un diagrama de estado. Algunos de los idiomas de ejemplo:

- ✓ Lenguaje de Modelado Unificado (UML): es un lenguaje de modelado estándar que se utiliza para describir sistemas de software. Incluye un conjunto de diagramas, entre ellos el diagrama de estado, que se utiliza para modelar el comportamiento del sistema.

- ✓ Diagrama de estado de Harel: es un lenguaje gráfico de modelado de sistemas de eventos discretos desarrollado por David Harel. Es una extensión del diagrama de estado tradicional y permite modelar estados complejos y procesos concurrentes.
- ✓ La tabla de transición de estados: También conocida como matriz de transición de estados, es una herramienta que se utiliza para modelar sistemas de estado finito. La tabla muestra una lista de los posibles estados del sistema y las transiciones entre ellos, en función de los eventos de entrada que se reciben.

Autor	Fecha	Versión
Martinez Zarate Yasbeth Mariana	05/11/23	3.0

14. PUNTO DE VISTA DEL ALGORITMO

En este apartado se describirá de manera detallada el diseño de las operaciones (métodos y funciones), los detalles internos y la lógica de cada entidad de diseño. Todo esto dependerá de la tarea específica que se realizará en el sistema.

14.1. PROBLEMAS DE DISEÑO

Los problemas de diseño pueden incluir varios desafíos que se deben considerar y ser resueltos durante la implementación del algoritmo para resolver los problemas en específico, estos problemas podrían ser:

- Eficiencia: El algoritmo podría ser un poco lento a la hora de resolver el problema, esto solo en caso de que sean datos de mayor magnitud y de cantidades exorbitantes.

- Escalabilidad: El algoritmo debe estar preparado para poder ajustarse a futuras mejoras, ya sea en cuanto al aumento de datos que se pueden llegar a generar o incluso un módulo que se puede llegar a implementar.

- Complejidad: El algoritmo debe ser lo suficiente simple de entender y ser modificable cuando se tenga que hacer cambios, pero lo suficientemente complejo para manjar problemas completos y producir los resultados que queremos.

14.2. ELEMENTOS DE DISEÑO

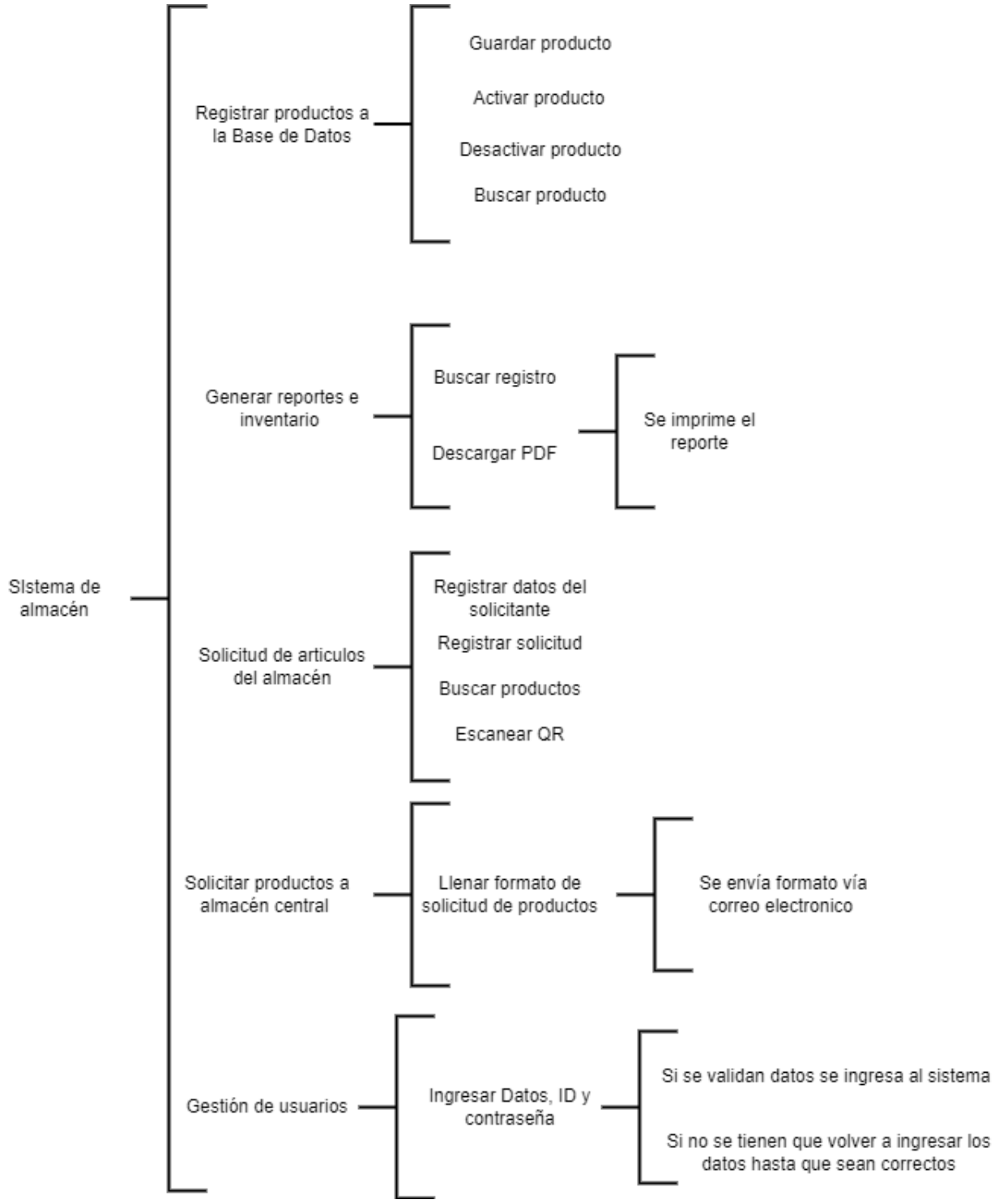


Diagrama jerárquico 37

Este diagrama (ilustración 38) nos muestra una descomposición jerárquica que consiste en dividir los procesos en componentes más pequeños y manejables, estos son representaciones de los módulos.

14.3. IDIOMAS DE EJEMPLOS

Para visualizar de una manera más sencilla el punto de vista de algoritmo se pueden utilizar los siguientes idiomas de ejemplo:

1. Tablas de decisiones: Una tabla de decisiones es una entrada de lógica de reglas planificadas, en formato de tabla, que se compone de condiciones, representadas en las cabeceras de columna y fila, y acciones, representadas como puntos de intersección de los casos condicionales de la tabla.
2. Diagramas de flujo: El diagrama de flujo o flujograma o diagrama de actividades es la representación gráfica de un algoritmo o proceso.
3. Pseudocódigos: Es una forma de representar código, como algoritmos, funciones y otros procesos, utilizando una combinación de lenguaje natural y elementos similares al lenguaje de programación.

Autor	Fecha	Versión
Martinez Zarate Yasbeth Mariana	20/11/23	3.0

15.PUNTO DE VISTA DE RECURSOS

Dentro de los puntos de vista de recursos haremos referencia a la gestión de recursos necesarios para llevar a cabo el desarrollo del software. Estos recursos ayudaran a modelar las características y a hacer un buen manejo de ellas.

12.1. PROBLEMAS DE DISEÑO

Los problemas de diseño en cuando al punto de vista de los recursos es amplia, dado que las preocupaciones clave incluirán la mala utilización de los recursos, la disponibilidad que estos tengan al momento de desarrollar el sistema o el mismo rendimiento.

- Utilización de recursos: En cuanto a una mala utilización de recursos, cabe mencionar que el tiempo es importante y está dentro de nuestros límites y recursos más importantes, al igual que el personal que nos apoyará en el desarrollo y las condiciones que nos den para el trabajo.

- Disponibilidad de recursos: La disponibilidad de los recursos es importante, ya que si no estuviesen disponibles nos veríamos afectados en el proceso de desarrollo, aumentando el tiempo de entrega y haciendo más trabajo del debido o estipulado.

15.2.1. ATRIBUTO DE RECURSOS

Se podrá interactuar con el sistema mediante conexión a internet la cual se podrá ejecutar en dispositivos móviles (smartphone, tabletas, computadoras portátiles, computadoras de escritorio). *(Se podrá interactuar con el sistema solo desde una computadora con las especificaciones requeridas)*

Para su eficaz funcionamiento con la interfaz del usuario se consideran los siguientes dispositivos de hardware importantes:

- Cámara.
- Teclado, mouse, monitor, CPU.
- Pantalla táctil.

Software.

- Lector de códigos QR.
- Acceso a internet.
- Navegador Chrome (de preferencia), explorador, safari, opera, etc.
- Link de acceso.
- Driver Mysql
- Python Django

Interfaces de software-hardware.

La interacción de ambas interfaces genera proceso en conjunto para las distintas operaciones con el sistema, como el registro de los productos por código QR, se requiere la cámara, la introducción de productos manualmente, una pantalla táctil o un teclado con la ayuda del servicio web se podrá manejar de forma correcta.

15.3. IDIOMAS DE EJEMPLOS

1. Diagrama de clases de UML: un diagrama de clases en Lenguaje Unificado de Modelado es un tipo de diagrama de estructura estática que describe la estructura de un sistema mostrando las clases del sistema, sus atributos, operaciones, y las relaciones entre los objetos.
2. Restricción de objetos de UML idioma OCL: OCL es un lenguaje notacional, subconjunto del UML estándar, industrial, que permite a los desarrolladores de software escribir

restricciones sobre modelos de objetos (pre y pos condiciones, invariantes, valores derivados y restricciones sobre operaciones).