

**GESTIÓN DE ALMACENES DAM**

Una caricatura de una persona

Descripción generada automáticamente con confianza media

**Realizado por:**

Azahara Blanco Rodríguez

Daniel Rodríguez Fernández

Daniel Alejandro Carmona Rodríguez

Jeremy Ramos Segura

Jan Carlos Ramos Martínez

**04/05/2022**

|  |  |  |  |
| --- | --- | --- | --- |
| **Proyecto** | GDAM | | |
| **Entregable** | Especificación de Requisitos | | |
| **Autor** | GDAM GROUP S.L. | | |
| **Versión/Edición** | V1.1 | **Fecha Versión** | 27/04/2022 |
| **Aprobado por** |  | **Fecha Aprobación** | DD/MM/AAAA |
|  |  | **Nº Total de Páginas** | 20 |

|  |  |
| --- | --- |
| **PROJECT MANAGER** | |
| Azahara Blanco Rodríguez |

|  |  |
| --- | --- |
| **PROJECT OWNER AND PLANNING MANAGER** | |
| Daniel Rodríguez Fernández |

|  |  |
| --- | --- |
| **QUALITY ASSURANCE MANAGER** | |
| Daniel Alejandro Carmona Rodríguez |

|  |  |
| --- | --- |
| **DATA BASE ADMINISTRATOR MANAGER** | |
| Jeremy Ramos Segura |

|  |  |
| --- | --- |
| **UI DESIGN MANAGER** | |
| Jan Carlos Ramos Martínez |

Tabla de contenido

[1. Introducción 4](#_Toc102064040)

[2. Problema Propuesto: 5](#_Toc102064041)

[3. Requisitos 8](#_Toc102064042)

[3.1. Requisitos Funcionales 8](#_Toc102064043)

[3.2. Requisitos No Funcionales 9](#_Toc102064044)

[3.3. Requisitos de Información 11](#_Toc102064045)

[4. Diseño e Implementación 13](#_Toc102064046)

[4.1. Diagramas 13](#_Toc102064047)

[4.1.1. Diagrama de Clases 14](#_Toc102064048)

[4.1.2. Diagramas de Casos de Uso 14](#_Toc102064049)

[4.2. Diseño Vista User 15](#_Toc102064050)

[4.3. Implementación 16](#_Toc102064051)

[5. Evaluación y Análisis 17](#_Toc102064052)

[6. Conclusiones 18](#_Toc102064053)

[7. Futuras Implementaciones 19](#_Toc102064054)

[8. Bibliografía 20](#_Toc102064055)

# Introducción

# Problema Propuesto:

Hemos sido contratados para realizar un sistema de gestión de almacenes, vamos a realizarlo para gestionar de forma genérica los Supplier, Reception, Article, Order, User y Store.

Comienzo describiendo cada clase y la funcionalidad de cada clase:

**User:**

Son aquellos que pueden acceder al sistema, solo los vamos a utilizar para loggearnos, tenemos dos perfiles:

*Atributos de la clase: userName(String), password(String), name(String), surnames(String), photo(String), email(String), dateCreatedAt(Local Date), typeUser(Manager – Employer).*

**Manager:**

Sus funciones son las siguientes: crear Article, actualizar un article, consultar todos los Article, consultar un article por código de article, crear User, eliminar User, consulta de todos los User, consulta de un User por su nombre de User. Crear Supplier, modificar Supplier, listar todos los Supplier, crea una nueva Reception, eliminar una Reception, consultar todas las Receptions, crear Order, borrar Order, consultar todos los Order, consultar un Order por su código de Order, crear store, consultar una store por código de store, consultar todas las Store, imprimir ordenes de Order o de Receptions en HTML, exportar e importar los datos del programa, copia de seguridad del programa.

**Empleado:**

Sus funcione serán más limitadas, sus funciones serán las siguientes: crear un Order, borrar un Order, buscar un Order por código de Order, mostrar todos los Order, listar todos los Article, buscar un article por código de article, crear una Reception, consultar Receptions, generar la orden de Order o de Reception en HTML.

**Article:**

Son todos los elementos físicos que tenemos almacenados en nuestro almacén, y podemos dar salida a través de los Order o de entrada cuando no haya existencias disponibles o deseemos aumentar el stock.

*Atributos de la clase: PIC(UUID), article(String), description(String), location(String), price(double), stock(int), isActive(Boolean).*

*\*\* PIC 🡪 Product Identification Code \*\**

**Supplier**

Son aquellos que nos permiten aumentar la cantidad de stock de un article. Pueden suministrar cualquier tipo de article, es decir que un article puede ser suministrado por Supplier diferentes.

*Atributos de la clase: SIC(UUID), nameSuplier(String), direction(String), telephoneNumber(String), email(String)*

*\*\* SIC 🡪 Suplier Identification Code \*\**

**Store**

Son aquellos los cuales hacen un Order de Article. Son propias de la empresa, por lo tanto, no se les aplica facturación.

*Atributos de la clase: STIC(UUID), name(String), cif(String), direction(String), telephoneNumber(String), email(String).*

*\*\* STIC 🡪 Store Identification Code \*\**

**Reception**

A través de la Reception podemos proveer nuestro almacén de existencias de Article. Las Receptions pueden crearlas tanto los Manager como los Empleados. Se puede hacer cuando necesitemos más cantidad de un article o falta de existencias. El Supplier de una Reception puede suministrar cualquier tipo de article.

Las Receptions no se pueden modificar, en caso de que este mal se tiene que borrar y se crea de nuevo la Reception, esta opción solo está disponible para el perfil de Manager.

Las Receptions están formadas de líneas de Reception.

Una vez que se finaliza la Reception se introduce el método de pago y nos devolverá la Orden de Reception en un archivo HTML.

*Atributos de la Clase Reception: RIC(UUID), SICSuplier(UUID), Carrier(String), Cost(double).*

|  |  |  |  |
| --- | --- | --- | --- |
| RIC | Supplier | Carrier | Cost |

*\*\* RIC 🡪 Reception Identification Code \*\**

*Atributos de la Clase Line Reception: RLIC(UUID), pic(UUID), articleName(String), load(int), unitPrice(double), totalPrice(Double)*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| RLIC | pic | articleName | load | unitPrice | totalPrice |

*\*\* RLIC 🡪 Reception Line Identification Code \*\**

**Order**

Cuando un cliente no hace una solicitud de Order, podremos crear un Order tanto el Manager como el Empleado podrán crear Order. Una vez que se realiza un Order se descuenta la cantidad del stock del article.

Si hacemos un Order y no hay existencias de un article tenemos varias opciones, o se salga de crear Order, o nos informe de que no hay cantidades existentes y modifiquemos el Order.

Un Order está formado por el Order y las líneas de Order que lo componen.

Una vez se finaliza el Order se genera una Orden de Order en HTML.

*Atributos de la Clase Order: OIC(UUID), customer(Lista de Clientes), Price(double).*

*\*\* OIC 🡪 Order Identification Code \*\**

*Atributos de la Clase Línea Order: OLIC(UUID), article(Lista de Article), load(int), unitPrice(double), totalPrice(double).*

*\*\* OLIC 🡪 Order Line Identification Code \*\**

**Tool**

El manager tiene la opción de exportar e importar los datos del programa en un JSON. Dentro del menú crearemos estas dos opciones.

# Requisitos

## Requisitos Funcionales

|  |  |  |  |
| --- | --- | --- | --- |
| **Cod** | **Módulo** | **Sección** | **Check** |
| RF1 | Supplier | Añadir Supplier. | ✔ |
| RF2 | Modificar Supplier. | ✔ |
| RF3 | Lista todos los Supplier | ✔ |
| RF4 | Reception | Añadir Reception. | ✔ |
| RF5 | Eliminar Reception. | ✔ |
| RF6 | Lista todas las Receptions. | ✔ |
| RF7 | Article | Añadir Article. | ✔ |
| RF8 | Actualizar Article. | ✔ |
| RF9 | Lista todos los Article. | ✔ |
| RF10 | Consulta por código de article. | ✔ |
| RF11 | Order | Añadir Order. | ✔ |
| RF12 | Eliminar Order. | ✔ |
| RF13 | Lista todos los Order. | ✔ |
| RF14 | Consulta por código de Order. | ✔ |
| RF15 | User | Añadir User. | ✔ |
| RF16 | Eliminar User. | ✔ |
| RF17 | Lista de todos los User. | ✔ |
| RF18 | Consulta por código de User. | ✔ |
| RF19 | Store | Añadir store. | ✔ |
| RF20 | Lista todas las Store. | ✔ |
| RF21 | Consulta por código de store. | ✔ |
| RF22 | Tool | Exportar datos. | ✔ |
| RF23 | Importar Datos. | ✔ |

## Requisitos No Funcionales

|  |  |  |
| --- | --- | --- |
| **Cod** | **Requisito No Funcional** | **Check** |
| RNF1 | Los colores del programa deben ser los acordados por el equipo de desarrollo. | ✔ |
| RNF2 | El programa se podrá visualizar por parte del User en tema claro o en tema oscuro, siempre con un botón deslizante en la esquina inferior derecha que permita cambiar el tema. | ✔ |
| RNF3 | El código de identificación de article estará formado por las siglas “PIC” seguido de un UUID aleatorio de 5 caracteres. | ✔ |
| RNF4 | Todos los User recogidos dentro del sistema tienen que tener asociados a ellos un User y contraseña. | ✔ |
| RNF5 | Todos los precios deben ser de tipo double, con dos decimales y moneda europea. | ✔ |
| RNF6 | El programa tendrá asociada una base de datos migrada en SQLlite. | ✔ |
| RNF7 | El código de identificación de Supplier estará formado por las siglas “SIC” seguido de un UUID aleatorio de 5 caracteres. | ✔ |
| RNF8 | El código de identificación de store estará formado por las siglas “STIC” seguido de un UUID aleatorio de 5 caracteres. | ✔ |
| RNF9 | El código de identificación de una Reception estará formado por las siglas “RIC” seguido de un UUID aleatorio de 5 caracteres. | ✔ |
| RNF10 | El programa cuenta con una copia de seguridad del mismo, la cual se exportará en un archivo JSON | ✔ |
| RNF11 | El código de identificación de línea de una Reception estará formado por las siglas “RLIC” seguido de un UUID aleatorio de 5 caracteres. | ✔ |
| RNF12 | El código de identificación de un Order estará formado por las siglas “OIC” seguido de un UUID aleatorio de 5 caracteres. | ✔ |
| RNF13 | El código de identificación de línea de un Order estará formado por las siglas “OLIC” seguido de un UUID aleatorio de 5 caracteres. | ✔ |
| RNF14 | Todas las fechas que maneje el programa deben tener el siguiente formato: dd-mm-yyyy | ✔ |
| RNF15 | El formato de los emails que se almacenen tiene que tener el siguiente formato: x@x.x | ✔ |
| RNF16 | El formato de los números telefónicos tiene que ser el siguiente:   * Han de comenzar con dígitos entre 6 y 9. * Seguido de, tendrán que disponer de 8 dígitos del 0 al 9. | ✔ |
| RNF17 | Para poder realizar el Order de un article la cantidad solicitada tiene que ser menor a la existente. | ✔ |
| RNF18 | La localización de un article tendrá el siguiente formato:  (F-C-A).   * F será Fila que será entre la A y Z * C será Columna número entre 0 y 9 * A será Altura número entre 0 y 9 | ✔ |
| RNF19 | Sí un article no está activo, no se podrá realizar el Order de uno de ellos. | ✔ |
| RNF20 | El fichero de orden de Order debe ser en formato HTML | ✔ |
| RNF21 | El fichero de orden de Order debe contener dirección de origen y destino, Article adquiridos con su cantidad precio unitario y precio de la línea de Order. En el pie de página indicará fecha y hora de generación. | ✔ |
| RNF22 | El programa tiene que ser 100% legible para todos los User que accedan al mismo. | ✔ |
| RNF23 | El programa tiene que ser portátil con lo cual podremos transportarlo a cualquier instancia. | ✔ |
| RNF24 | El fichero de orden de Reception debe ser en formato HTML | ✔ |
| RNF25 | El fichero de orden de Reception debe contener dirección de origen y destino, Article adquiridos con su cantidad precio unitario y precio de la línea de Order. En el pie de página indicará fecha y hora de generación. | ✔ |
| RNF26 | El programa cuenta con una pestaña de Acerca de donde irá información complementaria del programa tales como: versión, desarrolladores etc... | ✔ |
| RNF27 | Cada User tendrá un perfil con las opciones disponibles según su perfil, puede ser Manager o Employer.  Opciones de Manager:   * Añadir Supplier. * Modificar Supplier. * Lista todos los Supplier * Añadir Reception. * Eliminar Reception. * Lista todas las Receptions. * Añadir Article. * Actualizar Article. * Lista todos los Article. * Consulta por código de article. * Añadir Order. * Eliminar Order. * Lista todos los Order. * Consulta por código de Order. * Añadir User. * Eliminar User. * Lista de todos los User. * Consulta por código de User. * Añadir store. * Lista todas las Store. * Consulta por código de store. * Exportar datos. * Importar Datos.   Opciones de Employer:   * Lista todos los Supplier * Añadir Reception. * Lista todas las Receptions. * Lista todos los Article. * Consulta por código de article. * Añadir Order. * Eliminar Order. * Lista todos los Order. * Consulta por código de Order. * Lista todas las Store. * Consulta por código de store. | ✔ |
| RNF28 | El programa contará con un sistema de despliegue de ventanas según la opción que se elija dentro del menú de opciones de cada perfil | ✔ |
| RNF29 | El campo password de los User debe tener al menos una longitud de 4 caracteres. | ✔ |
| RNF30 | El formato del nombre de User será:  Nombre (PrimeraInicialprimerApellido)(SegundaInicialSegundoApellido), junto con su campo seguido de barra baja.  Ej.: DanielRF\_Empleado  Esto se aplicará sobre la vista del programa. El formato de entrada será Nombre + Apellidos. | ✔ |
| RNF31 | El campo de pago de una Reception puede ser transferencia o tarjeta | ✔ |
| RNF32 | Nuestro programa tiene que haber cumplido con los test específicos realizados por el conjunto de desarrolladores antes de su, distribución. Debe estar testeado a nivel de repositorio con (Mockito) y a nivel unitario con JUNIT. | ✔ |

## Requisitos de Información

|  |  |  |
| --- | --- | --- |
| **Cod** | **Requisito de Información** | **Check** |
| RI1 | User:   * userName * password * name * surnames * photo * email * dateCreatedAt * typeUser | ✔ |
| RI2 | Article:   * PIC * article * description * location * price * stock * isActive | ✔ |
| RI3 | Supplier:   * SIC * nameSuplier * direction * telephoneNumber * email | ✔ |
| RI4 | Store:   * STIC * name * cif * direction * telephoneNumber * email | ✔ |
| RI5 | Reception:   * RIC * SICSuplier * Carrier * Cost | ✔ |
| RI6 | Line Reception:   * RLIC * pic * articleName * load * unitPrice * totalPrice | ✔ |
| RI7 | Order:   * OIC * customer * Price | ✔ |
| RI8 | Línea Order:   * OLIC * article * load * unitPrice * totalPrice | ✔ |

# Diseño e Implementación

## Diagramas

En este apartado nos encontraremos distintos tipos de Diagramas, los diagramas nos ayudarán a interpretar mucho mejor la estructura y composición de nuestro programa, así como a la hora de programar tener un diseño de la arquitectura que va a seguir.

1. **Diagrama de clases:**
2. **Diagrama de casos de uso:**

## Diagrama de Clases

## Diagramas de Casos de Uso

## Diseño Vista User

## Implementación

# Evaluación y Análisis

# Conclusiones

# Futuras Implementaciones

# 

# Bibliografía

Proyectos GitHub:

<https://github.com/joseluisgs/Programacion-09-2021-2022>

<https://github.com/joseluisgs/Programacion-08-2021-2022>

<https://github.com/joseluisgs/EntornosDesarrollo-05-2021-2022>

<https://github.com/joseluisgs/EntornosDesarrollo-06-2021-2022>

<https://github.com/idanirf/ProgramacionUD8>

<https://github.com/idanirf/BaseDeDatosUD4>

Consultas Solución de errores:

<https://www.tabnine.com/code/java/methods/java.util.Optional/orElseThrow>

Referencias:

[https://es.myservername.com/features-functional requirements#:~:text=Por%20lo%20tanto%2C%20los%20requisitos,componente%20y%20no%20en%20funci%C3%B3n](https://es.myservername.com/features-functional%20requirements#:~:text=Por%20lo%20tanto%2C%20los%20requisitos,componente%20y%20no%20en%20funci%C3%B3n).

<https://grados.ugr.es/informatica/pages/reacredita/e21tfgs/_doc/memoriatfgnot87>

Entornos de Desarrollo y Tool:

<https://www.jetbrains.com/es-es/idea/>

<https://www.mysql.com/>