DIAGRAMA DE CLASES A PARTIR DEL INFORME DE REQUERIMIENTOS

GA4-220501095-AA2-EV01

PRESENTADO A:

INST. ANDRES FERNANDO SANCHEZ SOLARTE

PRESENTADO POR:

GRUPO NUMERO 3

CARLOS JOSE DELGADO GONZALEZ

DURLEY SANDRITH GALVAN JIMENEZ

JORGE ANDERSON CORTES TORRES

YASSER LEONARDO PACHECO CAÑIZARES

PROGRAMA:

PROGRAMACION DE APLICACIONES PARA DISPOSITIVOS MOVILES (2977832)

SERVICIO NACIONAL DE APRENDIZAJE

SENA

2024

INTRODUCCIÓN

Este documento presenta un diagrama de clases elaborado en una herramienta CASE de un software diseñado para gestionar internamente una joyería artesanal. El diagrama de clases es una representación visual de las entidades principales del sistema, sus atributos, métodos y las relaciones entre ellas. Se ha utilizado una herramienta CASE (STAR UML) para crear este diagrama, asegurando la precisión y claridad en la representación de la estructura del software.

Las herramientas CASE (Computer-Aided Software Engineering) son aplicaciones de software utilizadas para ayudar en el desarrollo y mantenimiento de software.

DESARROLLO

En este caso el sistema propuesto está diseñado para manejar los aspectos clave de una joyería artesanal, incluyendo la gestión de inventario, clientes, pedidos y catálogo.

Este diagrama servirá como base para el desarrollo del software, proporcionando una visión clara de la arquitectura del sistema a los desarrolladores y stakeholders.

Este diagrama incluye las siguientes clases principales:

- Articulo: representa los productos de la joyería.
- Cliente: almacena información de los clientes.
- Pedido: gestiona los pedidos realizados por los clientes.
- Catalogo: visualiza y modifica catálogo de artículos disponibles en la joyería.
- Inventario: controla el stock de materiales y joyas.
- Proveedor: gestiona la información de los proveedores de materiales.

Cada clase tiene atributos y métodos relevantes, y las relaciones entre clases están representadas con líneas y multiplicidades.

Análisis de las relaciones:

Pedido -- Articulo: Un pedido puede incluir muchos artículos, pero los artículos existen independientemente de los pedidos.

Catalogo -- Articulo: Un catálogo puede contener varios artículos, pero los artículos existen independientemente de que haya un catálogo o no y un catalogo puede cambiar sin afectar a los artículos ya creadas.

Cliente – Catalogo: el cliente puede revisar el catalogo y puede buscar un artículo, obtener una copia del catalogo en PDF y que la aplicación le muestre a través del catálogo las novedades de artículos.

Inventario -- Articulo: El inventario lleva el control de los artículos, pero los artículos pueden existir fuera del sistema de inventario.

Proveedor -- Inventario -- Articulo: El proveedor suministra los materiales que forman parte del inventario, pero ni el inventario ni los artículos dependen de un único proveedor para existir.

Catalogo – Cliente: Un cliente puede acceder a un catálogo, y el catálogo es el mismo para todos los clientes.

Articulo – Catalogo: Un catálogo contiene múltiples artículos, y un artículo pertenece a un solo catálogo en este contexto

DIAGRAMA DE CLASES (TEXTO)

[Articulo]
-id: int
-nombre: String
-descripcion: String
-precio: double
-material: String
-stock: int

_____
+crearJoya(): void
+actualizarStock(): void
+calcularPrecio(): double
+listarArticulo(): void
+eliminarArticulo(): void

[Cliente]
-id: int
-nombre: String
-email: String
-telefono: String

_____
+registrarCliente(): void
+actualizarInformacion(): void
+verCatalogo(): void
+ realizarPedidos(): boolean
+ verHistorialpedidos(): List<pedidos>

[Pedido]
-id: int
-fecha: Date
-estado: String
-total: double

_____
+crearPedido(): void
+actualizarEstado(): void
+calcularTotal(): double
-id: int
-nombre: String
-especialidad: String
+asignarTrabajo(): void
+actualizarDisponibilidad(): void

[Inventario]
-id: int
-nombre: String
-cantidad: int
-umbralMinimo: int

_____
+actualizarCantidad(): void
+verificarStock(): boolean

[Proveedor]
-id: int
-nombre: String
-contacto: String

_____
+realizarPedido(): void
+actualizarInformacion(): void

[Catalogo]
-articulo: List<articulo>
- nombre: String
- fechaActualizacion: Date

_____
-agregarArticulo(): void
- eliminarArticulo(): boolean
+ buscarArticulo(): Articulo
- actualizarPrecio(): boolean
+obtenerCatalogoPDF(): File
+ mostrarNovedades(): List<Articulo>

//////////////////////////////
RELACIONES
//////////////////////////////
Cliente "1" -- "N" Pedido
Pedido "N" -- "N" Articulo
Articulo "1" -- "N" Inventario
Proveedor “1” -- “N” Inventario “1” -- “N” Articulo
Catalogo “1”-- “N” Articulo
Cliente “1”-- “1” Catalogo

DIAGRAMA DE CLASES UTILIZANDO HERRAMIENTA CASE (STAR UML)