**DOCUMENTO DE DISEÑO DETALLADO PARA SISTEMA DE GALERÍA Y CASA DE SUBASTAS**

**1. Introducción**

Este documento detalla el diseño de un sistema de gestión para una galería y casa de subastas. El sistema manejará operaciones como la gestión de inventario de piezas de arte, usuarios con diferentes roles (compradores, administradores, cajeros, operadores), transacciones de compras, y administración de subastas.

**2. Descripción General del Sistema**

El sistema permitirá:

- Registrar y gestionar piezas de arte: Añadir, actualizar, consultar y devolver piezas.

- Administrar usuarios: Incluir diferentes tipos de usuarios con roles específicos y permisos.

- Realizar ventas y subastas: Manejar ventas directas a precio fijo y subastas dinámicas de piezas.

**3. Archivos de Datos**

**- Piezas:** La plataforma contiene una amplia gama de detalles sobre las piezas, que incluyen información sobre su tipo, propietario, condiciones de subasta/venta y otros aspectos relevantes. Para gestionar estos datos de manera efectiva, hemos optado por utilizar el formato JSON debido a su capacidad para manejar objetos de manera eficiente y estructurada, lo que facilita el proceso de persistencia.

**- Usuarios:** Nuestra plataforma almacena información detallada de los usuarios, asignándoles roles específicos tales como administrador, cajero y comprador. Optamos por utilizar el formato JSON para gestionar estos datos debido a su eficiencia en cuanto a persistencia, ya que nos permite manejar los objetos de manera ágil y estructurada.

**4. Diseño de Clases según en el UML**

EL diagrama Uml lo encontramos en el github por cuestioes de resolución de la imagen.

**Explicación de sus clases principales:**

**Usuario**

**- Atributos:**

- `login`: String

- `password`: String

- `nombre`: String

- `rol`: String

- `telefono`: int

- `verificado`: boolean

**- Métodos:**

- Constructores, getters y setters estándar para todos los atributos.

- `isVerificado()`: Verifica el estado de autenticación del usuario.

**Operador** (Extiende Usuario)

**- Métodos:**

- `getSubastas()`, `setSubastas(List<Subasta>)`: Gestiona las subastas asociadas al operador.

- `registrarOferta(Oferta)`: Registra una nueva oferta en una subasta.

- `agregarSubastas(Subasta)`: Añade una subasta a la lista de subastas del operador.

- `mejoresOfertas(List<Subasta>, List<Oferta>)`: Devuelve las mejores ofertas de una lista de subastas, según la pieza.

**Cajero** (Extiende de Usuario)

**- Métodos**:

- `registrarCompraSubasta(Oferta)`: Registra la compra de una pieza en una subasta.

- `registrarCompraPrecioFijo(Comprador, Pieza)`: Registra la compra de una pieza a precio fijo.

**Administrador** (Extiende de Usuario)

**- Métodos:**

- `ingresarPieza()`: Añade una nueva pieza al sistema.

- `aprobarVentaPrecioFijo(Comprador, Pieza)`, `aprobarVentaSubasta(Operador)`: Aprueba ventas tanto en precio fijo como en subastas.

-verificarCliente(Comprador):Verifica al usuario si no esta en mora.

-verificarCajero(Cajero): verificar al cajero si entra con una clase especifica.

-verificarOperador(Operador): verifica al operador si entra con una clase específica.

-`devolverPieza(Pieza, Propietario)`: Devuelve una pieza a su propietario.

- aprobarVentaSubasta(Operador)

**Comprador** (Extiende de Usuario)

**- Atributos:**

- `estadoCuenta`: float

- `maxCompras`: float

- `seguridadFinanciera`: boolean

- `mora`: boolean

- `metodoPago`: String

**- Métodos:**

- `getEstadoCuenta()`, `setMaxCompras(float)`: Gestiona el límite y el estado de la cuenta del comprador.

Clases de Entidades

Pieza

**- Atributos:**

- Detalles como `titulo`, `anoCreacion`, `autor`, `disponible`, `tiempoConsignacion`, `subasta`, `valores`, `propietario`, `bodega`.

- Métodos:

- Completo set de getters y setters.

- `isDisponible()`, `setDisponible(boolean)`: Gestiona la disponibilidad de la pieza.

**Propietario**

**- Métodos:**

- `getHistorialPiezas()`, `setHistorialPiezas(List<Pieza>)`: Accede y modifica el historial de piezas del propietario.

**Subasta**

**- Métodos:**

- `getPiezasSubastadas()`, `getOfertas()`: Obtiene las piezas y ofertas en la subasta.

**Oferta**

**- Atributos:**

- `piezaSubastada`: Pieza

- `comprador`: Comprador

- `valorOfertado`: int

- `subasta`: Subasta

**- Métodos:**

- Completo set de getters y setters para manejar los detalles de la oferta.

**5. Relaciones Entre Clases**

- Operador y Subasta: Un operador puede manejar múltiples subastas.

- Cajero y Compra: Un cajero procesa múltiples compras.

- Administrador y Pieza: El administrador tiene la autoridad para ingresar y devolver piezas.

- Comprador y Oferta: Compradores realizan ofertas en subastas.

**6. Implementación y Tecnología**

- Backend: Java para lógica de negocio y manejo de datos.

- Frontend: JavaFX para la interfaz de usuario o tecnologías web.

- Base de Datos: JSON para almacenamiento de datos, manipulados con librerías.

**7. Seguridad**

- Autenticación de usuarios y encriptación de contraseñas para garantizar la seguridad en el acceso al sistema.

- Validaciones en el lado del servidor para todas las entradas para prevenir inyecciones SQL y otros ataques comunes.

**8. Conclusión**

Este documento proporciona un diseño cohesivo y detallado para el sistema de gestión de una galería y casa de subastas, asegurando la funcionalidad, la seguridad y la eficiencia del sistema. Con este diseño, el sistema estará bien equipado para manejar las operaciones diarias, proporcionar una excelente experiencia de usuario, y garantizar transacciones seguras y confiables. Este diseño será la base sobre la cual el sistema será desarrollado, desplegado, y mantenido.