UNIVERSIDAD DE LAS FUERZAS ARMADAS ESPE

- Eduardo Anibal Mejía Catillo
- Bryan Roger Camacho Ramírez
- Israel Fernando Portilla Santamaria
- Jenifer Andrea Castro Cuevas
- Julio Emerson Coro Pineida
- Alexis Sebastián Murminacho Cabascango

TEMA:
Aplicación de Gestión de Tareas con Interfaz Gráfica y Base de Datos NoSQL

NRC:1323

QUITO-ECUADOR

Link repositorio:
https://github.com/Andre17C/DeberesAndrea2/tree/c9e25cac0325625ff07e56e460e1844d1de17f4b/GRUPO%209/PROYECTO

Link del video:
https://drive.google.com/file/d/14cHPlvwhTYPKH2dEOAc6a1euQfsNmYNV/view?usp=drivesdk

Diseño de una Aplicación para la Gestión de Tareas con Interfaz Gráfica y Base de Datos NoSQL

Objetivo General:

Desarrollar una aplicación que permita gestionar tareas mediante una interfaz gráfica y almacenamiento en una base de datos NoSQL, aplicando principios SOLID.

Objetivos Específicos:

- Diseñar e implementar una interfaz gráfica intuitiva para la gestión de tareas.

-Utilizar una base de datos oSQL para almacenar y consultar las tareas.

-Aplicar principios SOLID para mejorar la mantenibilidad y escalabilidad del código.

Marco Teórico:

Principios SOLID

Single Responsibility Principle (SRP): Cada clase debe tener una única responsabilidad.

Open/Closed Principle (OCP): El código debe ser abierto a extensiones, pero cerrado a modificaciones.

Liskov Substitution Principle (LSP): Las subclases deben poder sustituir a sus superclases sin alterar el funcionamiento.

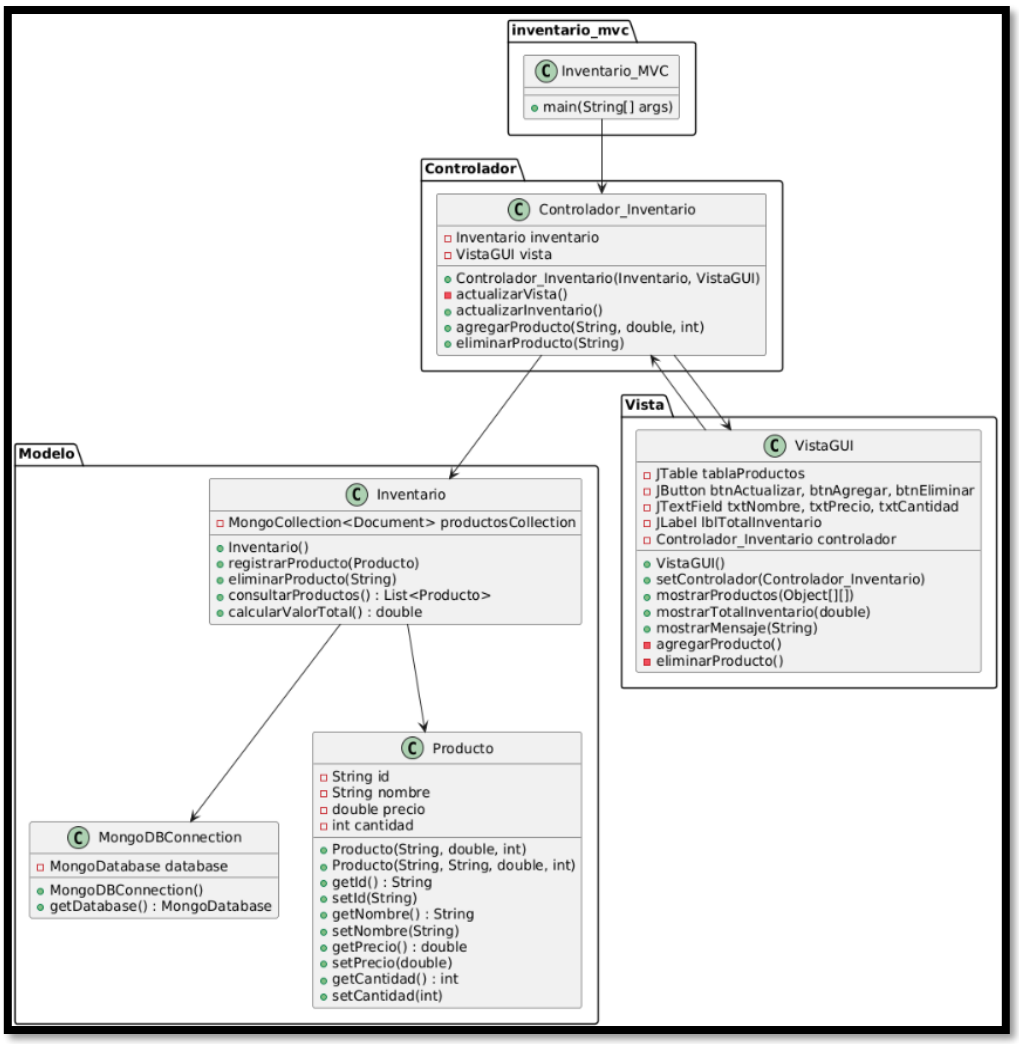
Interface Segregation Principle (ISP): Se deben definir interfaces específicas en lugar de generales.

Dependency Inversion Principle (DIP): Las dependencias deben basarse en abstracciones en lugar de clases concretas.

Base de Datos NoSQL

MongoDB se utiliza para almacenar las tareas de forma flexible y escalable, permitiendo consultas eficientes sin esquema fijo.

Diagrama de Flujo:



Diseño del Programa:

Modelo: Maneja la lógica y el acceso a la base de datos.

Vista: Presenta la interfaz gráfica al usuario.

Controlador: Gestiona la comunicación entre la vista y el modelo.

Estructura del Proyecto

inventario_MVC

|— Controlador

| ├— Controlador_Inventario.java

|— Modelo

| ├— Inventario.java

| ├— MongoDBConnection.java

| ├— Producto.java

|— Vista

| ├— VistaGUI.java

|— inventario_mvc

| ├— Inventario_MVC.java

|— Librerías

| ├— mongo-java-driver-3.12.14.jar

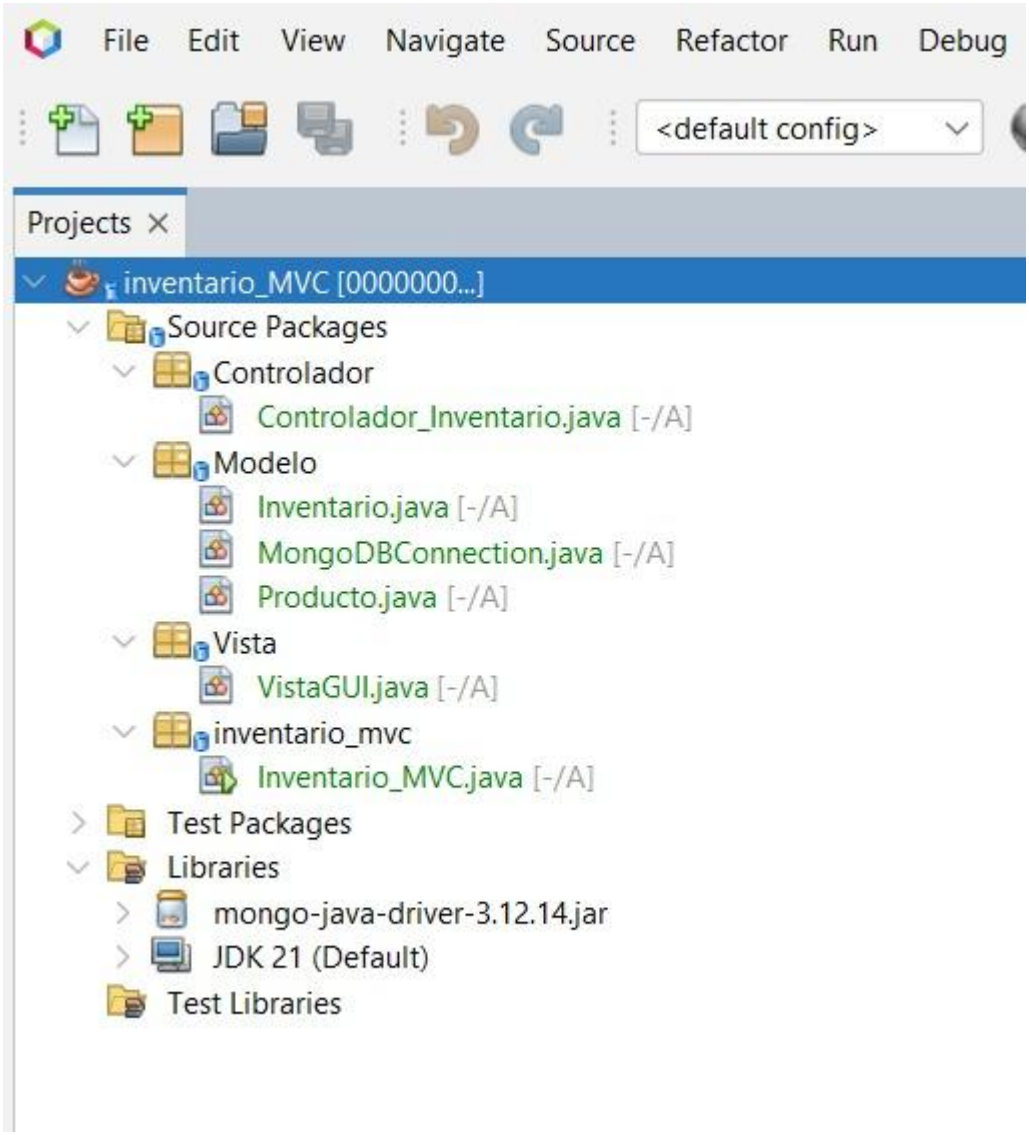
Código Fuente

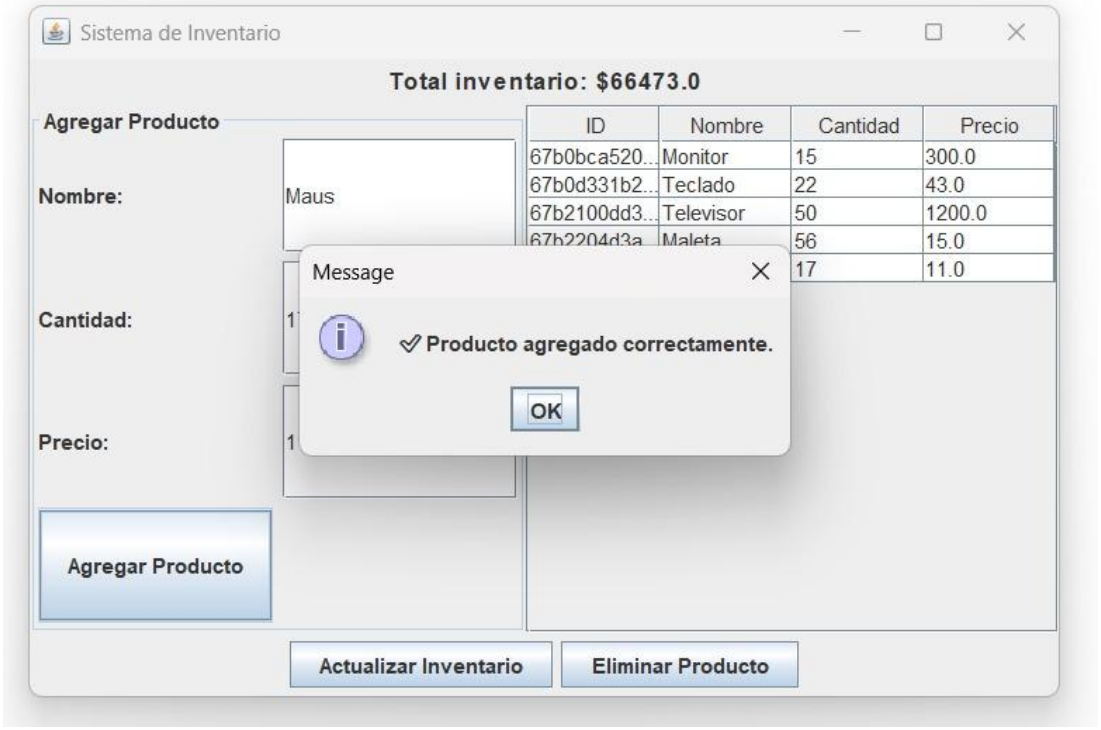
```
 package Controlador;
import Modelo.Inventario;
import Modelo.Producto;
import Vista.VistaGUI;
import java.util.List;
public class Controlador_Inventario {
    private final Inventario inventario;
    private final VistaGUI vista;
    public Controlador_Inventario(Inventario inventario, VistaGUI vista) {
        this.inventario = inventario;
        this.vista = vista;
        this.vista.setControlador(this);
        actualizarVista();
    }

    private void actualizarVista() {
        List<Producto> productos = inventario.consultarProductos();
        Object[][] data = new Object[productos.size()][4];
        for (int i = 0; i < productos.size(); i++) {
            Producto p = productos.get(i);
            data[i][0] = p.getId();       // ID (sin cambios)
            data[i][1] = p.getNombre();   // Nombre (sin cambios)
            data[i][2] = p.getCantidad(); // Ahora la cantidad esta antes del precio
            data[i][3] = p.getPrecio();   // Ahora el precio esta despues de la cantidad
        }
        vista.mostrarProductos(data);
        vista.mostrarTotalInventario(inventario.calcularValorTotal());
    }
    public void actualizarInventario() {
```

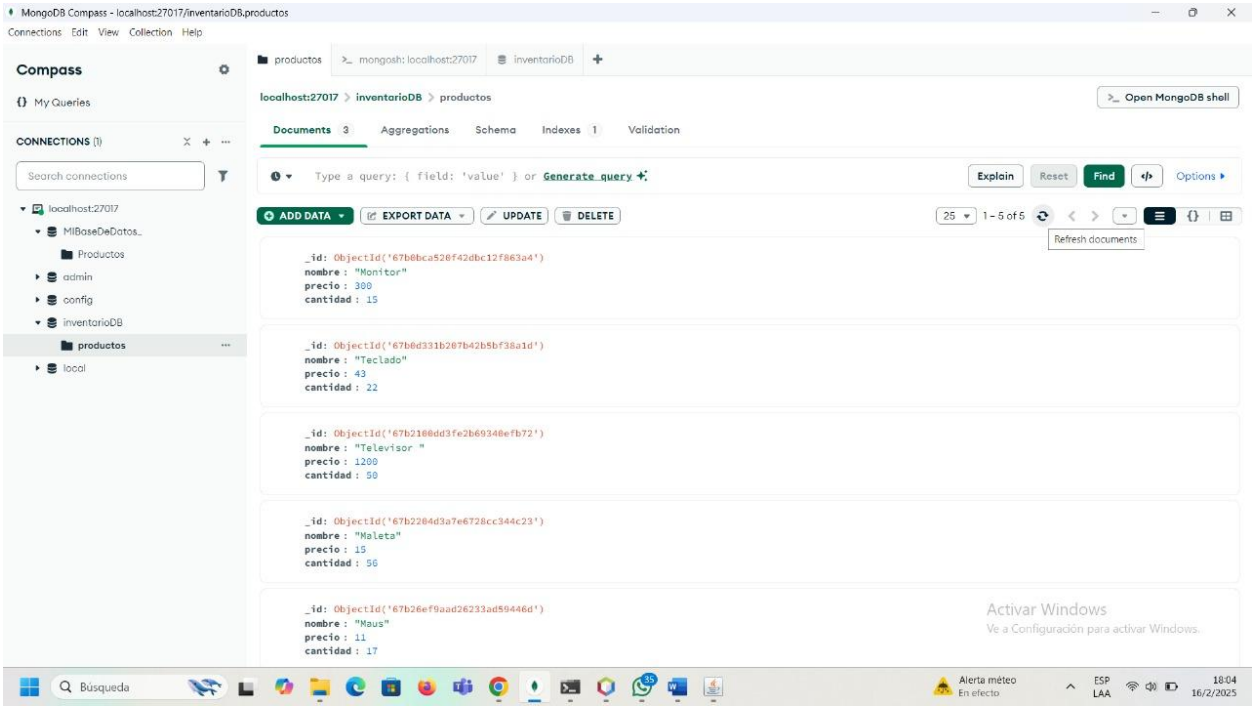
```
        try {
            actualizarVista();
            vista.mostrarMensaje("Inventario actualizado correctamente.");
        } catch (Exception e) {
            vista.mostrarMensaje("Error al actualizar el inventario: " + e.getMessage());
        }
    }
    public void agregarProducto(String nombre, double precio, int cantidad) {
        try {
            inventario.registrarProducto(new Producto(nombre, precio, cantidad));
            actualizarVista();
            vista.mostrarMensaje("Producto agregado correctamente.");
        } catch (Exception e) {
            vista.mostrarMensaje("Error al agregar el producto: " + e.getMessage());
        }
    }

    public void eliminarProducto(String id) {
        try {
            inventario.eliminarProducto(id);
            actualizarVista();
            vista.mostrarMensaje("Producto eliminado correctamente.");
        } catch (Exception e) {
            vista.mostrarMensaje("Error al eliminar el producto: " + e.getMessage());
        }
    }
}

package Modelo;
import com.mongodb.client.*;
```

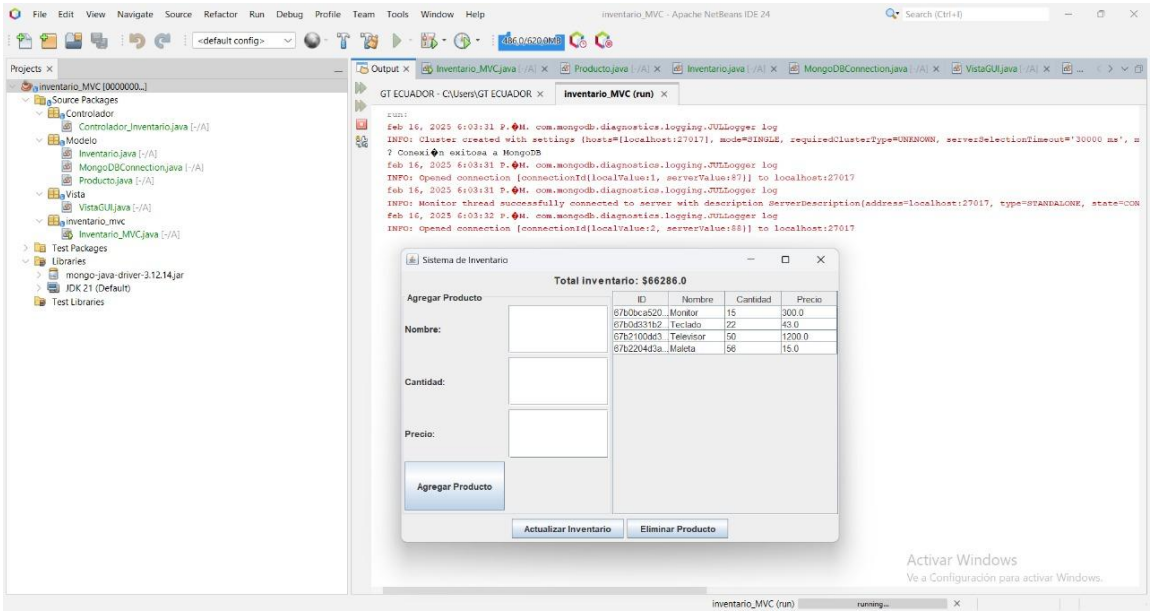
```
import org.bson.Document;
import org.bson.types.ObjectId;
import java.util.ArrayList;
import java.util.List;
public class Inventario {
    private final MongoCollection<Document> productosCollection;
    public Inventario() {
        MongoDBConnection conexion = new MongoDBConnection();
        productosCollection = conexion.getDatabase().getCollection("productos");
    }
    public void registrarProducto(Producto producto) {
        Document doc = new Document("nombre", producto.getNombre())
            .append("precio", producto.getPrecio())
            .append("cantidad", producto.getCantidad());
        productosCollection.insertOne(doc);
    }
   public void eliminarProducto(String id) {
        product…



package Modelo;
import com.mongodb.client.MongoClient;
import com.mongodb.client.MongoClients;
import com.mongodb.client.MongoDatabase;
public class MongoDBConnection {
    private MongoDatabase database;
    public MongoDBConnection() {
        try {
            // Conectar con MongoDB en localhost en el puerto 27017
            MongoClient mongoClient = MongoClients.create("mongodb://localhost:27017");
```

```
            database = mongoClient.getDatabase("inventarioDB"); // Nombre de la base de datos
            System.out.println("Conexion exitosa a MongoDB");
        } catch (Exception e) {
            System.out.println("Error de conexión a MongoDB: " + e.getMessage());
            e.printStackTrace();
        }
    }
    public MongoDatabase getDatabase() {
        return database;
    }
}
```


File
Edit
View
Navigate
Source
Refactor
Run
Debug
<default config>
Projects
inventario_MVC [0000000...]
Source Packages
Controlador
Controlador_Inventario.java [-/A]
Modelo
Inventario.java [-/A]
MongoDBConnection.java [-/A]
Producto.java [-/A]
Vista
VistaGUI.java [-/A]
inventario_mvc
Inventario_MVC.java [-/A]
Test Packages
Libraries
mongo-java-driver-3.12.14.jar
JDK 21 (Default)
Test Libraries


Sistema de Inventario
Total inventario: $66473.0
Agregar Producto
Nombre:
Maus
Cantidad:
Precio:
Agregar Producto
ID
Nombre
Cantidad
Precio
67b0bca520...
Monitor
15
300.0
67b0d331b2...
Teclado
22
43.0
67b2100dd3...
Televisor
50
1200.0
67b2204d3a...
Maleta
56
15.0
17
11.0
Message
Producto agregado correctamente.
OK
Actualizar Inventario
Eliminar Producto


MongoDB Compass - localhost:27017/inventarioDB.productos
Connections Edit View Collection Help
Compass
My Queries
CONNECTIONS (1)
Search connections
localhost:27017
MiBaseDeDatos..
Productos
admin
config
inventarioDB
productos
local
productos
mongosh: localhost:27017
inventarioDB
localhost:27017 > inventarioDB > productos
Open MongoDB shell
Documents 3
Aggregations
Schema
Indexes 1
Validation
Type a query: { field: 'value' } or Generate query
Explain
Reset
Find
Options
ADD DATA
EXPORT DATA
UPDATE
DELETE
25
1 - 5 of 5
Refresh documents
_id: ObjectId('67b0bca520f42dbc12f863a4')
nombre : "Monitor"
precio : 300
cantidad : 15
_id: ObjectId('67b0d331b207b42b5bf38a1d')
nombre : "Teclado"
precio : 43
cantidad : 22
_id: ObjectId('67b2100dd3fe2b69340efb72')
nombre : "Televisor "
precio : 1200
cantidad : 50
_id: ObjectId('67b2204d3a7e6728cc344c23')
nombre : "Maleta"
precio : 15
cantidad : 56
_id: ObjectId('67b26ef9aad26233ad59446d')
nombre : "Maus"
precio : 11
cantidad : 17
Activar Windows
Ve a Configuración para activar Windows.
Búsqueda
Alerta météo
En efecto
ESP
LAA
18:04
16/2/2025



Conclusiones:

- Se logró desarrollar una aplicación funcional para la gestión de tareas con interfaz gráfica y base de datos NoSQL.

-La aplicación sigue los principios SOLID, mejorando la organización y mantenibilidad del código.

Bibliografía:

- dev.xcheko51x (27 jul 2020) Instalación MongoDB en Windows. https://www.youtube.com/watch?v=quIDR_tLenw&list=PLJ5vS_GmZftjGpVzvSpmC0MNYn7CAVJg2

- Sin Rueda Tecnológica(4 ago 2021) ¿Cómo conectar Java(Apache Netbeans) con MongoDB?https://www.youtube.com/watch?v=TK2S32209cI

Justificación

Eduardo Anibal Mejia Catillo y Alexis Sebastian Murminacho Cabascango no participaron del video dado que tuvieron motivos de trabajo que atender, pero si participaron activamente en el desarrollo del programa y su implementación.