Universidad de las Fuerzas Armadas-ESPE

Asignatura de Programación Orientada a Objetos

SEGUNDO PARCIAL

TRABAJO PRÁCTICO

Nombres de los estudiantes: Juliette Estefanía Flores Tanicuchi.

Alejandra Mariela Sánchez Delgado

Enma Maritza Enríquez Romero

Luis Eduardo Erazo Vallejo

Fecha: 31-01-2025

TRABAJO PRÁCTICO

Tema: Sistema de Gestión de Inventario con MVC**.**

Introducción

Actualmente, la gestión eficiente de un inventario es primordial en los negocios que manejan productos físicos para un funcionamiento exitoso.

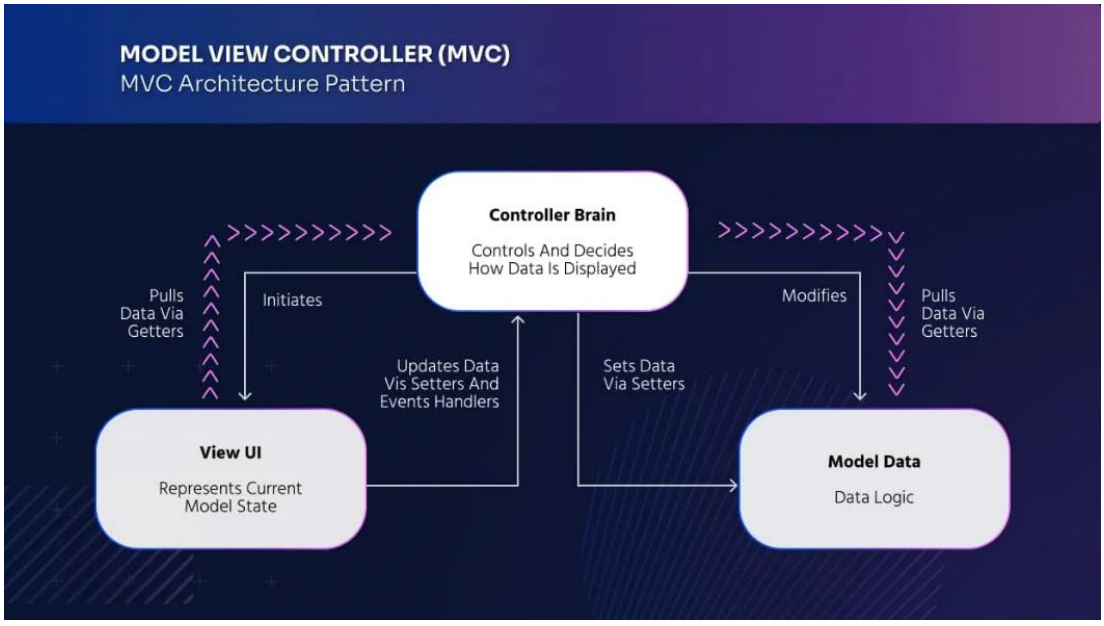
Un sistema de inventario permite controlar la disponibilidad, el valor y la cantidad de productos logrando una optimización de operaciones**.**

Objetivo General

- Desarrollar e implementar un sistema de gestión de inventario para una tienda utilizando el patrón de diseño Modelo-Vista-Controlador (MVC), permitiendo una administración eficiente y estructurada de los productos.

Marco Teórico

MVC es la abreviatura de Modelo, Vista y Controlador esto es una forma muy común para la organización de códigos en donde su función principal es que cada sección del código tenga un determinado propósito.



Patrón arquitectónico de un MVC

Dentro de las ventajas mas comunes de este patrón son:

- El aprovechamiento de ciertos componentes de la interfaz de usuario reutilizable
- Optimización del procesamiento de los datos, validación y conversación
- Permite el modularidad
- Sencillez para poder crear diferentes representaciones con los mismos datos

El patrón MVC aplicado a un sistema de gestión de inventario permite:

- Organización estructurada de la lógica de negocio, interfaz y control

- Mantenimiento eficiente del código

- Escalabilidad para añadir nuevas funcionalidades

- Testing independiente de cada componente

2. Componentes Principales

Modelo:

- Gestión de datos de productos

- Operaciones CRUD

- Reglas de negocio

- Validaciones

- Persistencia de datos

Vista:

- Interfaz de usuario

- Formularios de entrada

- Reportes

- Visualización de datos

- Mensajes al usuario

Controlador:

- Coordinación entre Modelo y Vista

- Procesamiento de solicitudes

- Gestión de flujo de datos

- Manejo de eventos

- Validaciones de entrada

Casos de Uso Típicos

1. Gestión de Productos:

   - Registro de nuevos productos

   - Actualización de existencias

   - Consulta de inventario

   - Eliminación de productos

2. Control de Stock:

   - Alertas de bajo inventario

   - Registro de movimientos

   - Control de pérdidas

   - Gestión de ubicaciones

3. Reportes:

   - Inventario actual

   - Movimientos históricos

   - Valoración del inventario

   - Productos críticos

4. Consideraciones de Diseño

Seguridad:

- Autenticación de usuarios

- Autorización de operaciones

- Registro de auditoría

- Backup de datos

Usabilidad:

- Interfaz intuitiva

- Flujos de trabajo eficientes

- Mensajes claros

- Ayuda contextual

Rendimiento:

- Optimización de consultas

- Caché de datos

- Paginación de resultados

- Procesamiento asíncrono

Tecnologías y Herramientas Comunes

Backend:

- Python (Django, Flask)

- Java (Spring)

- PHP (Laravel)

- .NET (ASP.NET MVC)

Frontend:

- HTML5/CSS3

- JavaScript

- Frameworks (React, Angular, Vue.js)

- Bootstrap/Material-UI

Base de Datos:

- MySQL

- PostgreSQL

- MongoDB

- SQLite

METODOLOGÍA

Para el desarrollo del sistema de gestión de inventario, se llevó a cabo un análisis detallado de los requerimientos funcionales y no funcionales. Los requerimientos funcionales incluyen la capacidad de agregar, modificar, eliminar y consultar productos en el inventario. Los requerimientos no funcionales abarcan la usabilidad, rendimiento y escalabilidad del sistema. Además, se identificaron los actores principales del sistema, como administradores y empleados de la tienda, quienes interactuarán con la aplicación.

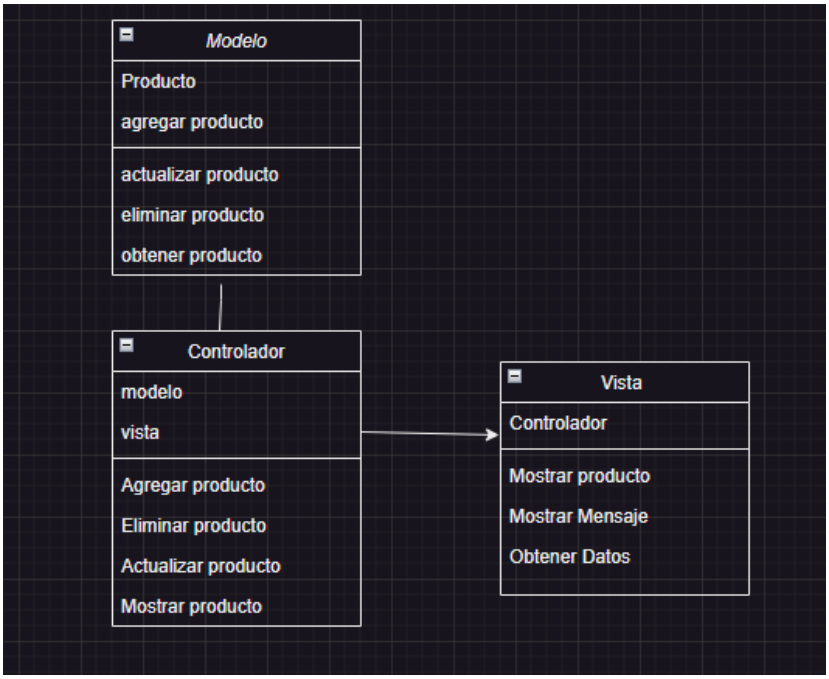
Diseño de la Arquitectura

El diseño del sistema se basa en el patrón MVC, permitiendo la separación de la lógica de negocio, la interfaz de usuario y la gestión de control. Se definieron los siguientes componentes clave:

Modelo: Encargado del manejo de datos, consultas y actualizaciones en la base de datos.

Vista: Responsable de la presentación de la información y la interacción con el usuario.

Controlador: Gestiona la lógica de aplicación, procesando las entradas del usuario y actualizando el modelo y la vista según corresponda.



Bibliografía

- Aguilar, J. M., & JMAguilar. (s/f). *¿Qué es el patrón MVC en programación y por qué es útil?* campusMVP.es. Recuperado el 3 de febrero de 2025, de https://www.campusmvp.es/recursos/post/que-es-el-patron-mvc-en-programacion-y-por-que-es-util.aspx
- *MVC: Model, view, controller*. (s/f). Codecademy. https://www.codecademy.com/article/mvc
- Bustamante, E., & Nittala, P. (2024, agosto 20). *Understanding the MVC pattern in software design*. Oshyn.com. https://www.oshyn.com/blog/mvc-pattern-software-design

Repositorio de Github

https://github.com/Esteff593/AE-Juliette-Flores-Alejandra-Sanchez-Enma-Enriquez-Luis-Erazo/upload