Univesidad Autonoma Metropolitana

Unidad Cuajimalpa

Tecnologias y Sistemas de la Informacion

Gestor de Inventario

Proyecto Final

Autores:
Oscar Martinez Barrales
Oswaldo Mejia Garcia
Aaron Rodrigo Ramos Reyes

Introducción

En la actualidad, las tiendas de refacciones para motocicletas enfrentan un desafío crítico en la gestión de sus inventarios, lo que impacta directamente en su eficiencia operativa y rentabilidad. La falta de un sistema organizado y automatizado para el control de stock genera problemas como desabastecimiento, exceso de mercancía obsoleta y pérdidas económicas significativas.

Problemática Principal

Uno de los principales obstáculos que enfrentan los propietarios de estos negocios es la ausencia de herramientas eficientes para administrar su inventario. Muchas tiendas aún dependen de métodos manuales, como registros en hojas de cálculo o anotaciones en papel, lo que aumenta el riesgo de errores humanos y dificulta la visibilidad en tiempo real de las existencias. Esto puede llevar a dos escenarios perjudiciales:

- **Falta de productos clave:** Cuando no hay un seguimiento adecuado, se agotan refacciones de alta demanda, generando pérdidas de ventas y clientes insatisfechos.
- **Acumulación de stock innecesario:** La falta de análisis de ventas lleva a sobrecompra de piezas con poca rotación, ocupando espacio y generando costos adicionales.

Impacto en los Clientes

Los clientes suelen enfrentar dificultades para encontrar las refacciones que necesitan debido a:

- Desorganización en el almacén.
- Falta de personal capacitado.

Esto no solo genera frustración, sino que también afecta la reputación del negocio, reduciendo oportunidades de fidelización.

Propuestas de solución

Para mitigar estos problemas, se recomienda implementar un **sistema de gestión de inventario especializado** que permita:

- Registrar entradas y salidas de productos de forma automatizada.
- Analizar tendencias de ventas para optimizar compras.
- Agilizar la búsqueda de productos.

Mision

Desarrollar un **sistema de gestión de inventario especializado en piezas de motocicleta** que permita a talleres, distribuidores o usuarios mantener un control preciso y eficiente de su stock, facilitando la organización, búsqueda y actualización de repuestos mediante una plataforma intuitiva y confiable.

Vision

Convertirnos en una herramienta esencial para la gestión de inventarios en el sector motociclista, ideal para talleres pequeños y medianos, con capacidad de adaptarse a futuras necesidades como integración con sistemas de ventas, alertas de stock bajo y análisis de demanda.

Aunque buscamos un objetivo complejo el proyecto tiene algunas limitaciones:

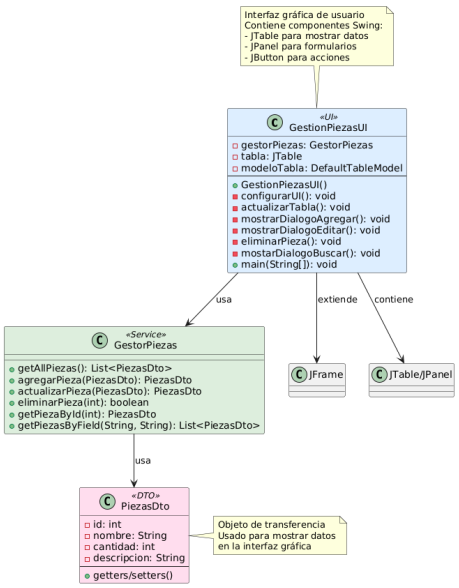
- **Uso simultaneo:** El proyecto solo permitira el uso de un usario a la vez.
- **Almacenamiento:** El proyecto solo permitira el almacenamiento de 1000 refacciones.
- **Busueda:** La interfaz permitira la busqueda de refacciones por nombre o por ID.
- **Sistema Operativo:** El proyecto solo funcionara en sistemas operativos windows.

Arquitectura del Proyecto

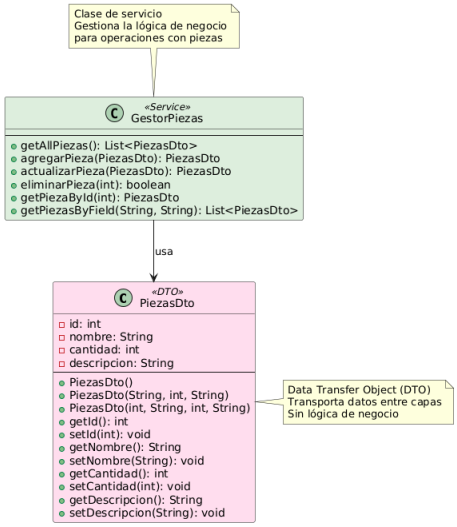
Vista Lógica

La arquitectura del proyecto se basa en el modelo de tres capas:

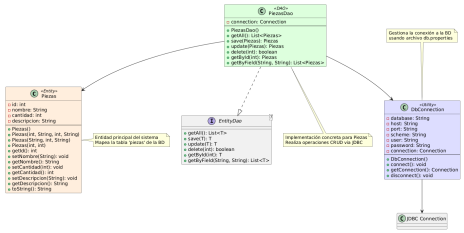
- **Capa de Presentación:** Esta capa es la encargada de mostrar la información al usuario. En esta capa se encuentra la interfaz gráfica con la que interactuara el usuario.



- **Capa de Negocio:** Esta capa es la encargada de procesar la información. En esta capa se realizaran todas las funciones del sistema a desarrollar.



- **Capa de Acceso a Datos:** Esta capa es la encargada de interactuar con la base de datos. En esta capa se encuentran las consultas a la base de datos.



Vista de Casos de Uso

La vista de casos de uso se basa en los siguientes casos de uso:

- **Registrar Refacción:** Este caso de uso permite al usuario registrar una refacción.
- **Modificar Refacción:** Este caso de uso permite al usuario modificar una refacción.
- **Eliminar Refacción:** Este caso de uso permite al usuario eliminar una refacción.
- **Buscar Refacción:** Este caso de uso permite al usuario buscar una refacción.

Resumen de la Arquitectura

Flujo Principal

- **Capa de Presentación** (`GestionPiezasUI`):
  - Interfaz gráfica Swing que captura interacciones del usuario
  - Envía solicitudes al `GestorPiezas` (capa de negocio)
- **Capa de Negocio** (`GestorPiezas`):
  - Recibe DTOs (`PiezasDto`) de la UI
  - Convierte DTOs a entidades (`Piezas`) para persistencia
  - Delega operaciones CRUD al `PiezasDao`
- **Capa de Datos**:
  - `PiezasDao` implementa `EntityDao` para operaciones SQL
  - Utiliza `DbConnection` para gestionar conexiones JDBC
  - Persiste/recupera entidades `Piezas` en la base de datos

Componentes Clave

Componente	Función
`PiezasDto`	Transporte de datos entre capas (sin lógica)
`PiezasDao`	Implementa CRUD usando el patrón DAO
`Piezas`	Entidad que mapea la tabla de base de datos
`EntityDao`	Interfaz genérica para operaciones de persistencia
`DbConnection`	Gestiona conexiones JDBC con parámetros de configuración

Relaciones Críticas

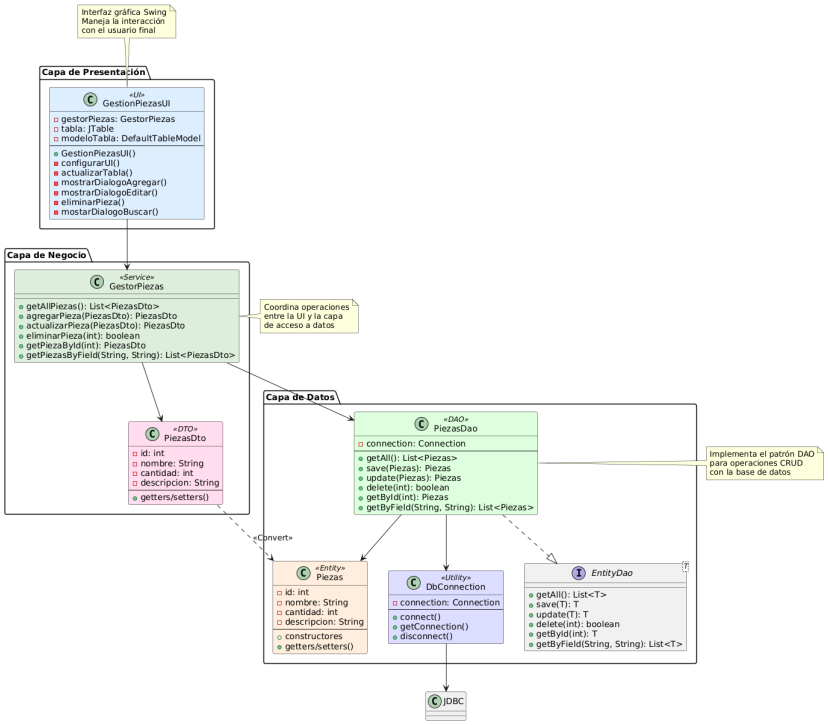
- **Dependecias:**
  - UI → Gestor: La interfaz depende del gestor para lógica de negocio
  - Gestor → (DAO + DTO): Coordina transformación y persistencia
  - DAO → (Entidad + BD): Opera sobre la base de datos
- **Conversión**: Transformación bidireccional entre `PiezasDto` (capa BL) y `Piezas` (capa DAL)

Patrones de Diseño

DTO (Data Transfer Object): Separa la estructura de datos de la lógica de negocio

DAO (Data Access Object): Abstracce el acceso a la base de datos

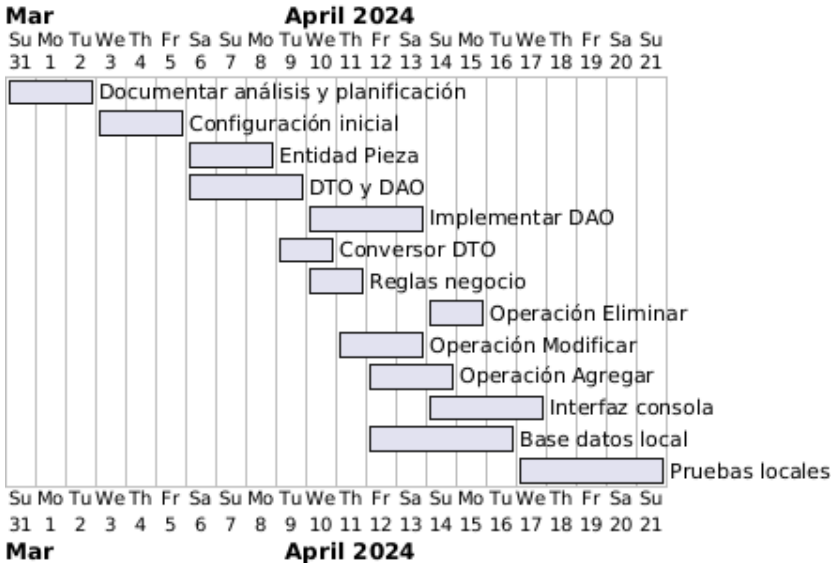
Separación de Capas: Arquitectura en 3 niveles (UI, BL, DAL) con responsabilidades claras



Cronograma

Integrantes

- Oscar Martinez Barrales
- Oswaldo Mejia Garcia
- Aaron Rodrigo Ramos Reyes



Descripción de actividades del Cronograma

- **Documentar análisis y planificación**: Crear documentación sobre los objetivos, requerimientos y planificación del proyecto.
- **Configuración inicial**: Preparar el entorno de desarrollo y las herramientas necesarias.
- **Entidad Pieza**: Modelar la entidad "Pieza.en el sistema.
- **DTO y DAO**: Diseñar los Data Transfer Objects (DTO) y Data Access Objects (DAO).
- **Implementar DAO**: Programar la capa de acceso a datos.
- **Conversor DTO**: Implementar conversores para transformar datos entre diferentes capas del sistema.
- **Reglas de negocio**: Definir e implementar las reglas que rigen la lógica del negocio.

- **Operación Eliminar**: Implementar la funcionalidad para eliminar datos.
- **Operación Modificar**: Implementar la funcionalidad para modificar datos.
- **Operación Agregar**: Implementar la funcionalidad para agregar nuevos datos.
- **Interfaz consola**: Diseñar y programar la interfaz basada en consola.
- **Base de datos local**: Configurar una base de datos local para el proyecto.
- **Pruebas locales**: Realizar pruebas para verificar la correcta funcionalidad del sistema.

Asignación de Tareas

Responsable	**Tareas Asignadas**
Aaron	Documentar análisis y planificación, Implementar DAO, Operación Eliminar, Base de datos local
Oscar	Configuración inicial, Conversor DTO, Operación Modificar, Pruebas locales
Oswaldo	Entidad Pieza, DTO y DAO, Reglas de negocio, Operación Agregar, Interfaz consola

Cuadro 1: Distribución de tareas entre el equipo de desarrollo

Conclusiones

El proyecto fue desafiante debido a que nunca habíamos trabajado con una arquitectura o estructura tan definida y "formal". Conceptos como los *DTO* (Data Transfer Object) y *DAO* (Data Access Object) eran completamente nuevos para nosotros, por lo que fue todo un reto comprenderlos e implementarlos.

Sin embargo, una vez que comenzamos a trabajar con ellos, nos dimos cuenta de todas las bondades que ofrecen. De igual modo, una interfaz gráfica era algo que hasta ahora no habíamos implementado, así que fue un desafío entender cómo hacer una y cómo conectarla con las funciones que ya teníamos desarrolladas.

Como conclusión general, nos quedamos con una sensación de crecimiento tras superar todos los desafíos que implicó desarrollar este sistema. Asimismo, el uso de nuevas herramientas como LaTeX y Git nos pareció muy enriquecedor y útil para futuros proyectos..