BuySellAuto: Modificaciones

Miguel Sánchez Rodríguez

Miguel Sánchez

Pablo Ruiz Morán

Jaime Alvarado Fernandez

Indice

1. Modificaciones de entregas 2
1.1. Entrega 0 2
1.2. Entrega 1 2
1.3. Entrega 2 4
2. Patrones utilizados 4
2.1. Modelo Vista Controlador (MVC) 4
2.2. Patron Singleton 5
3. Repositorio Git-Hub 5
4. Reparto de Responsabilidades 5

1. Modificaciones de entregas

1.1. Entrega 0

Durante el desarrollo del sistema, se han realizado las siguientes modificaciones con respecto a lo planteado inicialmente en la Entrega 0:

- Se descartó la funcionalidad de **gestión de contratos** de los empleados.
- No se implementó la funcionalidad de **optimización de transporte**, que incluía administración de rutas y costos para envíos nacionales e internacionales, al centrarse el sistema en la operativa local del taller.
- Se eliminó la **gestión logística avanzada**, descartando el seguimiento detallado del transporte y la asignación dinámica de rutas.
- No se desarrolló un sistema específico de **asociación de productos a ubicaciones físicas** dentro de los talleres o almacenes. El control de inventario se simplificó sin geolocalización interna.
- Se eliminó la figura del **proveedor de repuestos** como actor del sistema, ya que no se implementó su gestión ni registro.
- Se descartó el perfil de **usuario invitado**, de modo que todos los usuarios que acceden al sistema deben estar registrados.
- No se implementó el **sistema de chat entre compradores y vendedores**.
- Se excluyó finalmente el módulo de **gestión de eventos promocionales**, inicialmente previsto para registrar campañas o ferias.

1.2. Entrega 1

Durante el desarrollo del sistema se realizaron los siguientes cambios respecto a los requisitos definidos inicialmente en la Especificación de Requisitos del Software (ERS):

- Se **eliminó la funcionalidad de gestión de eventos promocionales** descrita en el apartado 2.2.3. Aunque inicialmente se propuso como una extensión del sistema, finalmente se descartó.
- Se **descartó la implementación del módulo de gestión de reseñas** (apartado 2.2.8), que permitía a los usuarios calificar productos y servicios.
- No se implementó la **verificación de calidad con sello identificativo** dentro de los servicios ofertados (2.2.4), ya que requería un sistema adicional de validación y trazabilidad que excedía el alcance del proyecto.

- Se **eliminó la posibilidad de realizar envío de facturas por correo electrónico** (mencionado en el REQ 06), dado que el sistema opera en entorno offline sin conexión a Internet.
- Se **simplificó la autenticación de usuarios**, eliminando la autenticación en dos pasos (2FA) especificada en el REQ 12. Actualmente, el sistema incluye autenticación estándar con correo y contraseña.
- Se **omitió la funcionalidad de exportación automática periódica de informes** indicada en el REQ 18. Solo se permite la exportación manual.
- Algunas funcionalidades mencionadas como parte de la evolución del sistema (sección 2.6), como la integración con dispositivos móviles, asistentes inteligentes o actualización centralizada, se consideraron fuera del alcance actual y no se implementaron.
- Se elimino el uso de una red LAN para mantener interconectados los diferentes quioscos que pudiera llegar a tener un taller.

Además, se realizaron las siguientes **modificaciones en los requisitos funcionales y no funcionales** definidos originalmente:

- **REQ-02 (Gestión de transporte):**
  Aunque se mantenía la idea de transporte como parte del modelo, se descartó la implementación completa de rutas, vehículos, o transportistas. Por tanto, este requisito fue eliminado.
- **REQ-05 (Gestión de contratos laborales):**
  Eliminado, ya que la gestión de contratos individuales por empleado no aporta funcionalidad clave en el sistema.
- **REQ-07 (Control de almacenes con localización precisa):**
  Se eliminó la necesidad de asociar cada producto a una ubicación exacta física. Solo se almacena información básica del taller o almacén, sin control logístico avanzado.
- **REQ-10 (Usuarios invitados):**
  Este tipo de usuario fue descartado, y el sistema requiere autenticación obligatoria para acceder a funcionalidades.
- **REQ-13 (Proveedores externos):**
  Eliminado el rol de proveedor y la funcionalidad de abastecimiento externo de repuestos.
- **REQ-17 (Chat privado):**
  La funcionalidad de chat entre usuarios fue eliminada por simplicidad. No se implementó ningún sistema de mensajería.

- **RNF-03 (Escalabilidad con servicios en la nube):**
  Eliminado como no aplicable al modelo actual, que funciona en entornos cerrados (quioscos locales sin conexión a Internet).
- **RNF-05 (Seguridad por biometría):**
  Se sustituyó por un sistema estándar de usuario y contraseña con bloqueo temporal tras intentos fallidos. No se aplicó biometría.

1.3. Entrega 2

Durante el desarrollo del sistema se realizaron los siguientes cambios respecto a los casos de uso del sistema:

- Se eliminó el caso de uso, perteneciente al usuario, de **desactivar cuenta**.
- Se eliminó el caso de uso, perteneciente al cliente, de **seleccionar horario**.
- Se eliminó el caso de uso, perteneciente al cliente, de **entregar cobro**.
- Se eliminó el caso de uso, perteneciente al administrador, de **reportes de incidencias**.
- Se eliminó el caso de uso, perteneciente al empleado, de **redactar informes de servicios**.
- Se eliminó el caso de uso, perteneciente al empleado, de **informe técnico**.
- Se eliminó el caso de uso, perteneciente al empleado, de **registrar piezas**.
- Se eliminó el caso de uso, perteneciente al administrador, de **actualizar agenda**.

Junto con los diagramas de secuencia de los casos de usos mencionados.

2. Patrones utilizados

2.1. Modelo Vista Controlador (MVC)

¿Dónde se ha utilizado?
El patrón MVC se ha utilizado como estructura arquitectónica principal en toda la aplicación. Cada ventana o pantalla del sistema cuenta con:

- Una **Vista** desarrollada en PyQt6
- Un **Controlador** específico que gestiona la lógica de interacción con la vista y coordina acciones con el modelo
- Un **Modelo**, formado por clases VO (Value Object) y DAOs.

¿Para qué se ha utilizado?
El patrón MVC se ha empleado para:

- Separar claramente la lógica de presentación de la lógica de negocio y acceso a datos.
- Permitir una mayor reutilización de código.
- Facilitar la modificación independiente de la interfaz y del modelo.
- Garantizar una mejor organización del sistema y facilitar futuras ampliaciones.

2.2. Patron Singleton

¿Dónde se ha utilizado?
El patrón Singleton se ha utilizado en la clase encargada de la **gestión de logs del sistema**. Esta clase mantiene una única instancia a lo largo de toda la ejecución de la aplicación, lo que permite centralizar el registro de eventos, errores y operaciones.

¿Para qué se ha utilizado?
El patrón Singleton se empleó para:

- Asegurar que exista una única instancia de la clase de logging.
- Centralizar el registro de acciones del sistema desde distintas partes del código.
- Evitar duplicidades o pérdidas de información en los registros.
- Facilitar el mantenimiento y depuración del sistema durante el desarrollo y pruebas.
- Creación de registro de operaciones para que pueda ser revisado como historial.

3. Repositorio Git-Hub

Enlace al repositorio github del proyecto:

https://github.com/Mig501/IngSw

4. Reparto de Responsabilidades

David Morán Gorgojo
Ha actuado como **coordinador general y supervisor del proyecto**, participando de forma transversal en todas las áreas. Ha servido de **enlace y comunicacion**

entre las diferentes partes, contribuyendo tanto al desarrollo de la **documentación técnica y funcional**, como a la implementación de **componentes de base de datos, frontend y backend**. Su rol ha sido clave para asegurar la coherencia entre las distintas capas del sistema.

Miguel Sánchez Rodríguez
Encargado principal de la **documentación técnica y funcional** del sistema, incluyendo la ERS, **los diagramas UML** y los contratos de operación. Además, ha sido uno de los principales responsables del **diseño y desarrollo de la base de datos**.

Pablo Ruiz Morán
Responsable de la **lógica de backend**, incluyendo la **implementación de los controladores** y la **conexión con los modelos** de datos. Ha colaborado activamente en la **integración del backend** con la interfaz gráfica y también en la **construcción y validación de la base de datos**.

Jaime Alvarado Fernández

Encargado del **frontend** del sistema, desarrollando la interfaz gráfica con PyQt6. También ha trabajado en la **conexión del frontend con el backend**, para lograr una integración fluida entre capas. Ha **diseñado las vistas** del sistema y se ha ocupado de la gestión de eventos de usuario, **navegación entre pantallas** y experiencia de usuario.