Manual Técnico de AutoGest Pro

1. Introducción

Este manual técnico describe la arquitectura, el diseño y la implementación del sistema AutoGest Pro, una aplicación de gestión para talleres de reparación de vehículos. Este documento está dirigido a desarrolladores, arquitectos de software y otros técnicos involucrados en el desarrollo y mantenimiento del sistema.

2. Arquitectura del Sistema

AutoGest Pro sigue una arquitectura modular, con los siguientes componentes principales:

- **Interfaz de Usuario:** Desarrollada con GTK, proporciona una interfaz gráfica para la interacción del usuario.
- **Lógica de Negocio:** Implementa las funcionalidades principales del sistema, como la gestión de usuarios, vehículos, repuestos, servicios y facturas.
- **Almacenamiento de Datos:** Utiliza estructuras de datos en memoria (listas, pilas, colas, matrices dispersas) para almacenar y gestionar la información.
- **Generación de Reportes:** Utiliza Graphviz para generar reportes visuales de las estructuras de datos.
-

3. Diseño de la Base de Datos

El sistema utiliza las siguientes entidades y relaciones:

- **Usuarios:** Almacenados en una lista simplemente enlazada. Atributos: ID, Nombres, Apellidos, Correo, Contraseña.
- **Vehículos:** Almacenados en una lista doblemente enlazada. Atributos: ID, ID_Usuario, Marca, Modelo, Placa.
- **Repuestos:** Almacenados en una lista circular. Atributos: ID, Repuesto, Detalles, Costo.
- **Servicios:** Administrados en una cola. Atributos: ID, Id_Repuesto, Id_Vehiculo, Detalles, Costo.
- **Facturas:** Almacenadas en una pila. Atributos: ID, ID_Orden, Total.
- **Bitácora:** Administrada mediante una matriz dispersa. Registra la relación entre vehículos y repuestos utilizados en los servicios.

4. Diseño de la Interfaz de Usuario

La interfaz de usuario se compone de las siguientes ventanas principales:

- **Inicio de Sesión:** Permite a los usuarios acceder al sistema.
- **Menú Principal:** Proporciona acceso a las funcionalidades principales del sistema.
- **Carga Masiva:** Permite cargar datos desde archivos JSON.
- **Ingreso Manual:** Permite ingresar datos manualmente.
- **Gestión de Usuarios:** Permite ver, editar y eliminar usuarios.
- **Generación de Servicios y Facturas:** Permite crear servicios y generar facturas.
- **Reportes:** Permite generar y visualizar reportes.

5. Implementación

- **Lenguaje de Programación:** C#
- **Estructuras de Datos:** Listas enlazadas, pilas, colas, matrices dispersas.
- **Punteros:** Se utiliza unsafe code para la gestión de punteros en el registro de vehículos.
- **Carga Masiva:** Los datos se cargan desde archivos JSON y se validan antes de ser almacenados en las estructuras de datos.
- **Generación de Reportes:** Se utiliza la librería Graphviz para generar reportes visuales de las estructuras de datos.
- **Bitácora:** Se implementa una matriz dispersa para registrar la relación entre vehículos y repuestos. Se actualiza automáticamente con cada servicio generado.

6. Pruebas

Se realizan pruebas unitarias, de integración y de sistema para garantizar la calidad del software. Se utilizan pruebas automatizadas y manuales para verificar la funcionalidad y el rendimiento del sistema.

7. Despliegue

La aplicación se despliega en un entorno Linux. Se proporcionan instrucciones detalladas para la compilación, empaquetado y despliegue de la aplicación.

8. Mantenimiento

Se proporcionan instrucciones para realizar copias de seguridad y restaurar la base de datos. Se describen los procedimientos para actualizar y mantener la aplicación.