

# 2024

## SIGMA

Sistema de Información para la Gestión de Mantenimiento de Automóviles



Jose Miguel Cendan Cabanilles

1 DAM

23-5-2024

## Contenido

Introducción .....	3
Herramientas y Métodos .....	3
Perspectiva Estática .....	4
E/R (Entidad-Relación) .....	4
Paso a Colecciones .....	5
DDL (Data Definition Language) .....	5
DML (Data Manipulation Language) .....	5
DQL (Data Query Language) .....	6
DCL (Data Control Language) .....	7
Perspectiva Dinámica .....	7
Sketch .....	7
Casos de Uso (Métodos) .....	8
Conclusiones .....	9
Bibliografía y Webgrafía .....	10

## Introducción

Las necesidades básicas de un taller automovilístico para controlar y revisar los mantenimientos realizados nos han llevado a realizar una base de datos. En ella le podremos insertar los datos esenciales sobre los vehículos y también sobre cada mantenimiento o reparación que se le realice a cada automóvil.

Esta base de datos nos permitirá buscar por matrícula, ofreciéndonos así información sobre si está dado de alta en nuestra base de datos y cuáles han sido las revisiones de dichos vehículos, si es que las tienen.

## Herramientas y Métodos

Oracle VM VirtualBox, el software de virtualización multiplataforma de código abierto más popular del mundo, permite a los desarrolladores entregar código más rápido, ya que pueden ejecutar múltiples sistemas operativos en un solo dispositivo. Los equipos de TI y los proveedores de soluciones usan VirtualBox para reducir los costos operativos y acortar el tiempo necesario para implementar aplicaciones de forma segura en entornos locales y en la nube. Los sistemas operativos de host compatibles incluyen Windows, Linux y macOS. Los equipos de desarrollo pueden consolidar cargas de trabajo usando VirtualBox para admitir cargas de trabajo masivas de hasta 32 CPU virtuales.

MongoDB ([enlace externo a IBM](#)) es un sistema de gestión de bases de datos (DBMS, por sus siglas en inglés) no relacionales y de código abierto, que utiliza documentos flexibles en lugar de tablas y filas para procesar y almacenar varias formas de datos. Al ser una solución de base de datos NoSQL, MongoDB no requiere un sistema de gestión de bases de datos relacionales (RDBMS), por lo que proporciona un modelo de almacenamiento de datos elástico que permite a los usuarios almacenar y consultar fácilmente tipos de datos multivariados. Esto no solo simplifica la gestión de la base de datos para los desarrolladores, sino que también crea un entorno altamente escalable para aplicaciones y servicios multiplataforma.

Mongo DB Compass es una aplicación multiplataforma que permite explorar la estructura de documentos de las distintas colecciones que componen una base de datos MongoDB, de manera fácil e intuitiva.

Los documentos o colecciones de documentos de MongoDB son las unidades básicas de datos. Estos documentos, con formato JSON (Java Script Object Notation) binario, pueden almacenar varios tipos de datos y distribuirse en varios sistemas. El diseño de esquema dinámico de MongoDB brinda a los usuarios una flexibilidad sin igual para crear registros de datos, consultar colecciones de documentos a través de la agregación de MongoDB y analizar grandes cantidades de información.

Python es un lenguaje de programación ampliamente utilizado en las aplicaciones web, el desarrollo de software, la ciencia de datos y el machine learning (ML). Además de que se puede ejecutar en muchas plataformas diferentes. Python cuenta con una gran biblioteca estándar que contiene códigos reutilizables para casi cualquier tarea.

Con esta herramienta podremos ejecutar el programa principal y gracias a su librería podremos mostrar los resultados que hemos explicado en la introducción.

Tkinter es una librería del lenguaje de programación Python y funciona para la creación y el desarrollo de aplicaciones de escritorio. Esta librería facilita el posicionamiento y desarrollo de una interfaz gráfica de escritorio con Python. Tkinter es el paquete estándar de Python para interactuar con Tk.

Dia es un programa para dibujar diagramas de estructura. Hemos utilizado el programa para hacer un diagrama que nos mostrará la estructura de las entidades y sus relaciones.

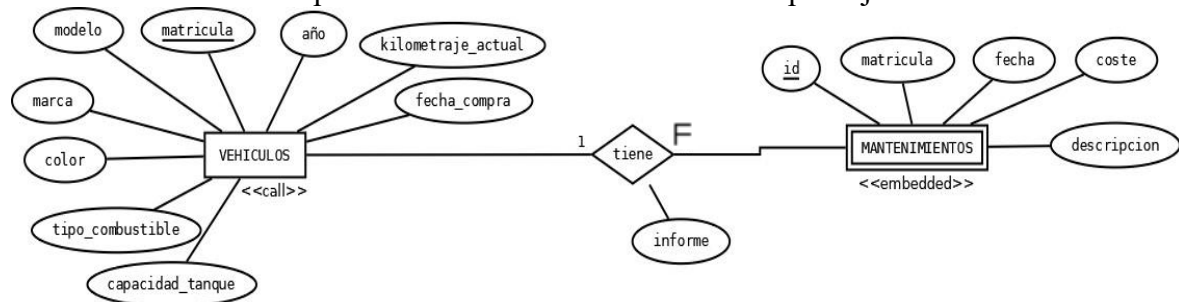
Paint es una app de edición gráfica versátil y fácil de usar en Windows para que puedas crear, editar y manipular imágenes y dibujos. Paint es ideal para el recorte, cambio de tamaño, dibujos sencillos, así como la adición de formas básicas y texto a las imágenes. Ofrece una interfaz sencilla con una variedad de herramientas y características para cualquier persona que necesite llevar a cabo tareas básicas de edición gráfica. Su simplicidad y facilidad de uso convierten a Paint en una herramienta valiosa para la edición rápida y fácil de imágenes.

La hemos utilizado para hacer un boceto y poder tener una idea básica de cómo queríamos que fuera la estética y estructura de las ventanas.

## Perspectiva Estática

### E/R (Entidad-Relación)

Hemos dibujado esta entidad relación porque unos vehículos pueden tener muchos mantenimientos y los mantenimientos solo se le hace a un vehículo. El vehículo llama a mantenimientos para sacar información de cuando se produjo.



En la foto anterior se puede ver la entidad principal de la relación incorpora el contenido de entidad secundaria.

Relaciones 1:1, 1: F (uno a poquitos) porque no saben cuántos informes gastaremos en a lo largo del programa.

## Paso a Colecciones

Para tener los datos y las relaciones claras haremos un paso para tener claro las colecciones, nos mostraría unas colecciones así:

VEHICULOS=matricula+año+kilometraje\_actual+fecha\_compra+capacidad\_tanque+ tipo\_combustible+color+marca+modelo.

MANTENIMIENTOS=id+matricula+fecha+coste+descripción

C. Ali: matricula VEHICULOS (matricula)

R.I: nul(matricula) nul (informe)

## DDL (Data Definition Language)

En una base de datos, podemos organizar los datos en colecciones que representan diferentes entidades y sus relaciones. A continuación, se muestran ejemplos de cómo podrían ser estas colecciones para una base de datos de una librería, usando MongoDB como referencia.

```
vehiculo = {  
  "_id": matricula,  
  "marca": marca,  
  "modelo": modelo,  
  "año": año,  
  "color": color,  
  "tipo_combustible": tipo_combustible,  
  "capacidad_tanque": capacidad_tanque,  
  "fecha_compra": fecha_compra,  
  "kilometraje_actual": kilometraje_actual  
}
```

```
mantenimiento = {  
  "matricula": matricula,  
  "fecha": fecha,  
  "descripcion": descripcion,  
  "costo": costo  
}
```

## DML (Data Manipulation Language)

El DML se utiliza para insertar, actualizar, eliminar y consultar datos en la base de datos. Aquí tienes ejemplos de cómo realizar estas operaciones en MongoDB para las colecciones Vehiculos y Mantenimiento.

## 1. Insertar datos

### Vehiculos

```
vehiculo = {  
  "matricula": "ABC123",  
  "marca": "Toyota",  
  "modelo": "Corolla",  
  "año": 2020,  
  "color": "Rojo",  
  "tipo_combustible": "Gasolina",  
  "capacidad_tanque": 50.0,  
  "fecha_compra": "2022-01-15",  
  "kilometraje_actual": 15000.0  
}  
db.Vehiculos.insert_one(vehiculo)
```

### Mantenimiento

```
mantenimiento = {  
  "matricula": "ABC123",  
  "fecha": "2023-05-20",  
  "descripcion": "Cambio de aceite y filtro",  
  "costo": 200.0  
}  
db.Mantenimientos.insert_one(mantenimiento)
```

## 2. Actualizar datos

### Actualizar datos

```
db.Vehiculos.update_one(  
  {"matricula": "ABC123"},  
  {"$set": {"color": "Azul"}}  
)
```

## DQL (Data Query Language)

El DQL en MongoDB se utiliza para consultar datos de las colecciones. A continuación, se presentan varios ejemplos de consultas DQL para las colecciones Vehiculos y Mantenimientos.

```

mantenimientos = db['Mantenimiento']
resultados = mantenimiento.find({"matricula": matricula})
mensaje = f'Mantenimientos para la matrícula {matricula}: \n'
for mantenimiento in resultados:
    mensaje+=f'Fecha: {mantenimiento["fecha"]}, Descripción: {mantenimiento["descripcion"]},
{mantenimiento["costo"]}\n'
if mensaje == f('Mantenimientos para la matrícula {matricula}: \n'):
    mensaje = f('No se encontraron mantenimientos para la matrícula {matricula}')
messagebox.showinfo('Mantenimientos', mensaje)

```

En este bucle for, iteramos sobre cada documento en los resultados de la consulta. Para cada documento, agregamos una línea al mensaje con los detalles del mantenimiento: fecha, descripción y costo.

Si después de iterar el mensaje no ha cambiado (lo que significa que no se encontró ningún mantenimiento), actualizamos el mensaje para indicar que no se encontraron mantenimientos para la matrícula proporcionada.

### DCL (Data Control Language)

El lenguaje DCL (Data Control Language) se usa para controlar el acceso a los datos en una base de datos. En MongoDB, esto se relaciona principalmente con la gestión de usuarios y roles, controlando quién puede hacer qué dentro de la base de datos.

```

def verificar_admin():
    if entry_usuario.get() == "admin" and entry_contraseña.get() == "admin":
        ventana_login.destroy() # Cerrar la ventana de inicio de sesión
        abrir_ventana_principal() # Abrir la ventana principal de la aplicación
    else:
        messagebox.showerror("Error", "Nombre de usuario o contraseña incorrectos")

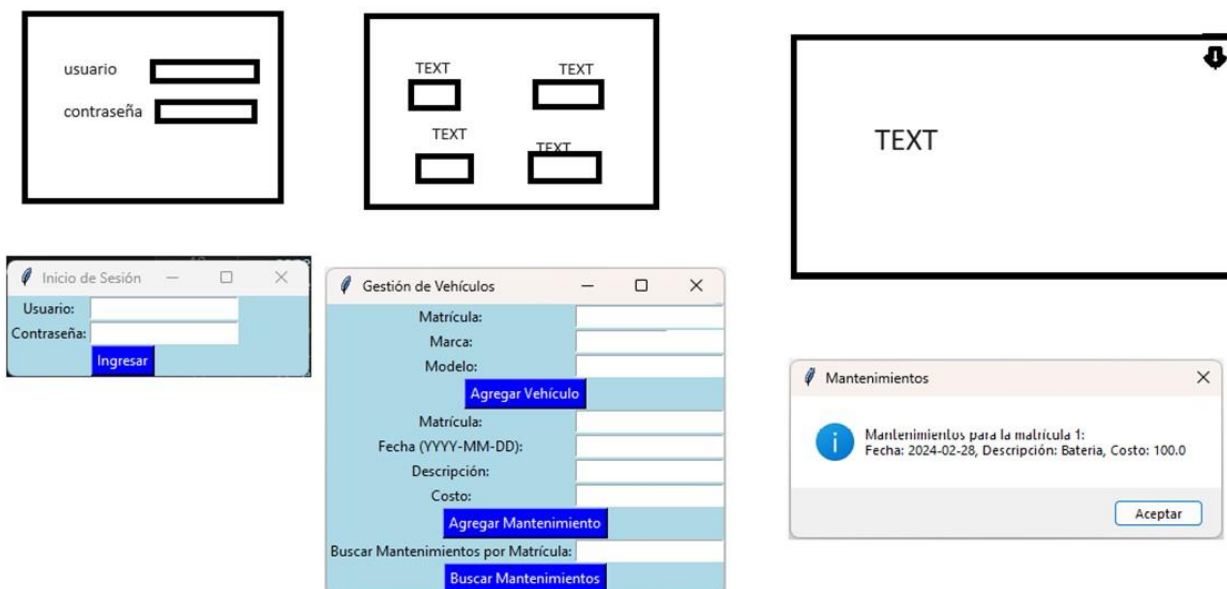
```

## Perspectiva Dinámica

### Sketch

Un sketch es un dibujo o boceto rápido que se realiza para representar una idea, concepto o imagen de manera simplificada. Es una forma rápida y básica de representar algo visualmente, a menudo utilizado como punto de partida para proyectos artísticos más grandes o como medio para comunicar una idea de forma rápida y efectiva.

Como veremos en la siguiente imagen podremos ver como hemos hecho un boceto con el cliente a través del Paint porque teníamos las reuniones online, después de varios dibujos quedamos en tener esta idea principal.



Después de acordar con el cliente sobre el concepto del sketch y si están satisfechos con la dirección que está tomando, podrías proceder a desarrollar una versión más detallada o final del diseño. Esto implica refinar los detalles, agregar color si es necesario, y asegurarte de que el diseño cumpla con los requisitos y expectativas del cliente. Una vez que el cliente esté completamente satisfecho con el diseño final, podrías proceder a cualquier paso adicional necesario, como la producción física del diseño o su implementación digital, dependiendo del proyecto en cuestión. Es importante mantener una comunicación abierta con el cliente durante todo el proceso para garantizar que sus necesidades y visiones se cumplan de manera efectiva.

### Casos de Uso (Métodos)

Caso de Uso	Descripción
Registro de Vehículo	El usuario registra un nuevo vehículo en el sistema, ingresando todos los detalles relevantes del vehículo, como la marca, modelo,
Registro de Mantenimiento	El usuario registra un nuevo mantenimiento realizado en un vehículo.
Visualización del Historial de Mantenimiento	El usuario visualiza el historial de mantenimiento de un vehículo en particular.

Registro de Vehículo:

Descripción: El usuario puede ingresar los detalles de un nuevo vehículo en el sistema. Esto incluye información relevante como la marca, modelo, año, número de serie, y cualquier otra información pertinente sobre el vehículo.



Objetivo: Mantener un registro completo y detallado de todos los vehículos que están siendo gestionados por el sistema.

Registro de Mantenimiento:

Descripción: El usuario puede registrar cualquier mantenimiento que se realice en un vehículo específico. Esto puede incluir cambios de aceite, reparaciones, inspecciones programadas, etc.

Objetivo: Registrar y rastrear todas las actividades de mantenimiento realizadas en cada vehículo para garantizar su correcto funcionamiento y prolongar su vida útil.

Visualización del Historial de Mantenimiento:

Descripción: El usuario puede acceder al historial completo de mantenimiento de un vehículo en particular. Esto mostrará todos los registros de mantenimiento previamente ingresados para ese vehículo, incluidas las fechas de servicio, los tipos de mantenimiento realizados y cualquier nota o detalle relevante asociado.

Objetivo: Proporcionar una visión completa y transparente del historial de mantenimiento de cada vehículo, lo que permite una mejor planificación de futuros servicios y una gestión más eficiente del mantenimiento.

## Conclusiones

Después de realizar la aplicación SIGMA, son varias las reflexiones que podemos obtener, tanto sobre su elaboración como de los resultados obtenidos. A continuación, se ponen en exposición cuáles son estas conclusiones.

Para ello, primero debemos entender el completo funcionamiento de la aplicación. En ella, se ha diseñado una base de datos relacional con dos tablas principales: "Vehículos" y "Mantenimiento".

La tabla "Vehículos" almacena información detallada sobre los vehículos, como su matrícula, marca, modelo, año, color, tipo de combustible, etc.

La tabla "Mantenimiento" registra los mantenimientos realizados en los vehículos, con detalles como la fecha, descripción y costo del mantenimiento, así como una referencia a la matrícula del vehículo al que corresponde.

Estos resultados proporcionan una base sólida para el desarrollo de un sistema de gestión del mantenimiento de vehículos, permitiendo a los usuarios registrar vehículos y realizar un seguimiento de los mantenimientos realizados.

## Reflexiones sobre el proceso

Identificar claramente los requisitos del sistema desde el principio fue crucial. Esto permitió diseñar una estructura de base de datos y casos de uso que satisficieran las necesidades del sistema de manera efectiva.

Mantener un enfoque interactivo y recibir retroalimentación durante el proceso ayudó a refinar y mejorar el diseño.

### Posibles mejoras futuras

Agregar restricciones y validaciones adicionales en la base de datos y la interfaz de usuario puede mejorar la integridad de los datos y la experiencia del usuario.

Explorar la posibilidad de agregar más funciones al sistema, como poder insertar en la base de datos las facturas generadas a los vehículos.

Continuar mejorando la interfaz de usuario para que sea intuitiva y fácil de usar.

### Bibliografía y Webgrafía

Youtube

ChatGPT

[IntroMongo.pdf](#)

Repositorio: [Github](#)