**Logotipo

Descripción generada automáticamente**

**Informe Desarrollo aplicaciones Web 1**

**Gestión de auto lote**

**Alumnos:**Wilmer Aguilar – T32411065

José Martínez - T32311436

**Índice**

Contenido

[Informe Desarrollo Aplicaciones Web 1 3](#_Toc193914356)

[Resumen general 3](#_Toc193914357)

[Propósito principal 5](#_Toc193914358)

[Características principales 5](#_Toc193914359)

[Glosario 6](#_Toc193914360)

[Retos encontrados 6](#_Toc193914361)

[Funcionalidades principales 8](#_Toc193914362)

[Tecnología 9](#_Toc193914363)

[Desarrollo 9](#_Toc193914364)

[Modelo Entidad Relación 10](#_Toc193914365)

[Anexo 11](#_Toc193914366)

[Metodología GitHub / ramas 11](#_Toc193914367)

[Metodología GitHub / gitflow 11](#_Toc193914368)

[Metodología CRUD / get 12](#_Toc193914369)

[Metodología CRUD / put 12](#_Toc193914370)

[Metodología CRUD / post 13](#_Toc193914371)

[Metodología CRUD / delete 13](#_Toc193914372)

[Base de datos 14](#_Toc193914373)

[Código 14](#_Toc193914374)

# Informe Desarrollo Aplicaciones Web 1

Descripción de la solución

## Resumen general

Utilizando la herramienta Express.js, desarrollamos el backend de una aplicación de gestión de auto lote. Este sistema cuenta con múltiples dependencias, cada una representada como una tabla en nuestra base de datos. Para garantizar un acceso seguro a dichas dependencias, implementamos JSON Web Tokens (JWT) junto con bcrypt, lo cual permite restringir el acceso mediante un sistema de autenticación basado en login.

A cada dependencia se le implementó un CRUD completo (Crear, Leer, Actualizar y Eliminar), y el acceso a estas operaciones está protegido por medio de tokens generados al iniciar sesión correctamente.

Instancias del sistema y su finalidad:

1. **Usuarios**

Finalidad: Almacena los datos del personal del sistema (vendedores, administradores, etc.).

**Campos clave:**

* id\_usuario: Identificador único.
* nombre, correo, contraseña: Datos de acceso.
* rol: Rol del usuario (ej. vendedor, admin).
* fecha\_creacion: Fecha de alta en el sistema.

1. **Clientes**

Finalidad: Guarda información de los clientes del autolote.

**Campos clave:**

* id\_cliente: Identificador único.
* nombre, correo, telefono: Datos de contacto.
* fecha\_registro: Fecha de incorporación al sistema.

1. **Vehículos**

Finalidad: Contiene los datos de los vehículos disponibles para la venta.

**Campos clave:**

* id\_vehiculo: Identificador único.
* marca, modelo, año: Especificaciones del vehículo.
* precio, disponibilidad, descripcion: Detalles para la venta.
* fecha\_publicacion: Fecha de publicación en el sistema.

1. **Ventas**

Finalidad: Registra las transacciones de compra-venta realizadas.

**Campos clave:**

* id\_venta: Identificador único.
* id\_vehiculo, id\_cliente, id\_vendedor: Relaciones con otras entidades.
* precio\_final, impuestos, total: Información financiera.
* fecha\_venta: Fecha en la que se concretó la venta.

1. **Consultas**

Finalidad: Almacena mensajes o solicitudes realizadas por los clientes.

**Campos clave:**

* id\_consulta: Identificador único.
* id\_cliente, id\_vehiculo: Referencias cruzadas.
* mensaje: Contenido de la consulta.
* fecha\_consulta: Momento en que fue realizada.

1. **Imágenes de Vehículos**

Finalidad: Gestiona las imágenes asociadas a los vehículos publicados.

**Campos clave:**

* id\_imagen: Identificador único.
* id\_vehiculo: Relación con el vehículo correspondiente.
* url\_imagen: Ruta o enlace de la imagen almacenada.

**Tasas de Cambio**

Finalidad: Guarda las tasas de conversión entre distintas monedas.

**Campos clave:**

* id\_tasa: Identificador único.
* moneda\_origen, moneda\_destino: Códigos de moneda (ej. USD, EUR).
* tasa: Valor de conversión.
* fecha\_actualizacion: Última actualización de la tasa.

Finalmente, realizamos pruebas de todas las rutas y funcionalidades usando Postman, para verificar el correcto funcionamiento de cada operación CRUD, así como la autenticación y autorización mediante JWT.

## Propósito principal

El propósito principal del sistema de gestión de auto lote es optimizar y automatizar la administración de vehículos, clientes, vendedores y ventas mediante una API RESTFULL y una base de datos estructurada.

**Este sistema permite:**

* Mejorar la eficiencia en la venta de vehículos, asegurando un control preciso del inventario.
* Facilitar la gestión de clientes y vendedores, registrando sus datos y relaciones comerciales.
* Automatizar el proceso de ventas, calculando precios, impuestos y generando reportes.
* Garantizar la seguridad y trazabilidad de la información con autenticación y validaciones.

## Características principales

* Arquitectura y Tecnologías:

1. Backend: Node.js con Express
2. Base de datos: MySQL
3. Autenticación: JWT para sesiones seguras

* Funcionalidades:

1. Gestión de Vehículos: Consulta, modificación y eliminación de vehículos
2. Gestión de ventas: Registro y seguimiento de ventas de vehículos

## Glosario

|  |  |
| --- | --- |
| **Término** | **Definición** |
| **import** | Permite traer módulos, funciones o componentes desde otros archivos o bibliotecas. |
| **module.export** | Función en Node.js que permite exportar funciones, objetos o variables desde un archivo para que puedan ser usados en otros archivos. |
| **authmiddleware** | Función intermedia (middleware) que verifica si una solicitud está autenticada antes de permitir el acceso a ciertas rutas o recursos. |
| **request** | Objeto que representa la solicitud HTTP hecha por el cliente. Contiene datos como parámetros, cabeceras, cuerpo, etc.. |
| **response** | Objeto que representa la respuesta que el servidor enviará al cliente. Se usa para enviar datos, mensajes o estados HTTP. |
| **PORT** | Número que representa el puerto en el cual el servidor escucha y responde solicitudes (por ejemplo, 3000, 8080, etc.). |
| **get** | Método HTTP utilizado para obtener datos del servidor. |
| **post** | Método HTTP usado para enviar datos al servidor, comúnmente para crear nuevos recursos. |
| **put** | Método HTTP utilizado para actualizar por completo un recurso existente en el servidor. |
| **delete** | Método HTTP que sirve para eliminar un recurso del servidor. |
| **token** | Cadena de texto (generalmente cifrada) usada para identificar y autenticar a un usuario en solicitudes protegidas (como un JWT) |

## Retos encontrados

1. A lo largo de este proyecto enfrentamos diversas complicaciones que fuimos superando progresivamente. Entre los principales desafíos se encuentran:
2. Adaptarnos al uso de Git y la metodología GitFlow para el control de versiones.
3. Acostumbrarnos a la modalidad de trabajo en equipo, coordinando tareas y responsabilidades.
4. Resolver errores desconocidos durante la implementación de distintos protocolos en Node.js y sus diversas aplicaciones.
5. Abarcar una amplia cantidad de temas teóricos y prácticos, e implementarlos correctamente dentro del proyecto.
6. Enfrentar la diferencia en los tiempos disponibles entre los miembros del equipo, lo cual requirió una buena organización y comunicación constante.

## Funcionalidades principales

|  |  |  |
| --- | --- | --- |
| Nombre del uso | Descripción de la funcionalidad | Ruta de uso |
| Gestión de vehículos | Permite al usuario registrar nuevos vehículos al sistema, consultar el inventario de vehículos, actualizar la información del vehículo (precio, estado etc.…) | http://localhost:3000/api/vehiculos |
| Gestión de clientes | Permite registrar nuevos clientes y consultar información de clientes | http://localhost:3000/api/clientes |
| Control de vehículos | Permite a los administradores registrar ventas, asociar la venta a un cliente y o empleado, y consultar el historial de vehículos. | http://localhost:3000/api/vehiculos |
| Gestión de imágenes | Permite a los administradores poder subir y eliminar imágenes de los vehículos. | http://localhost:3000/api/imagenes |

# Tecnología

## Desarrollo

**Node.js:** Entorno de ejecución de JavaScript: en el servidor, basado en el motor V8 de Chrome. Permite desarrollar aplicaciones escalables y de alto rendimiento, especialmente para backend y APIs.

**Express.js**: Framework minimalista para Node.js que facilita la construcción de APIs RESTful. Usado para manejar las rutas y peticiones del servidor de manera eficiente.

**Express.Router:** Módulo de Express que permite crear rutas de forma modular y organizada. Ayuda a dividir las rutas en distintos archivos para mantener el código más limpio y escalable.

**Dotenv:** Paquete que permite cargar variables de entorno desde un archivo.env al entorno de Node.js, facilitando la configuración sin exponer datos sensibles como contraseñas o claves API.

**MySQL:** Sistema de gestión de bases de datos relacional (RDBMS) basado en SQL. Es ampliamente utilizado para almacenar y administrar datos en aplicaciones web y empresariales.

**Nodemon:** Herramienta que reinicia automáticamente el servidor de Node.js cada vez que detecta cambios en el código, agilizando el desarrollo.

**JSON Web Tokens (JWT)**: Estándar de autenticación que permite intercambiar información de manera segura mediante un token firmado, usado comúnmente para verificar la identidad del usuario en aplicaciones web.

**Postman**: Herramienta utilizada para probar las APIs del backend y asegurarse de que las rutas y las respuestas sean correctas antes de ser implementadas en la aplicación.

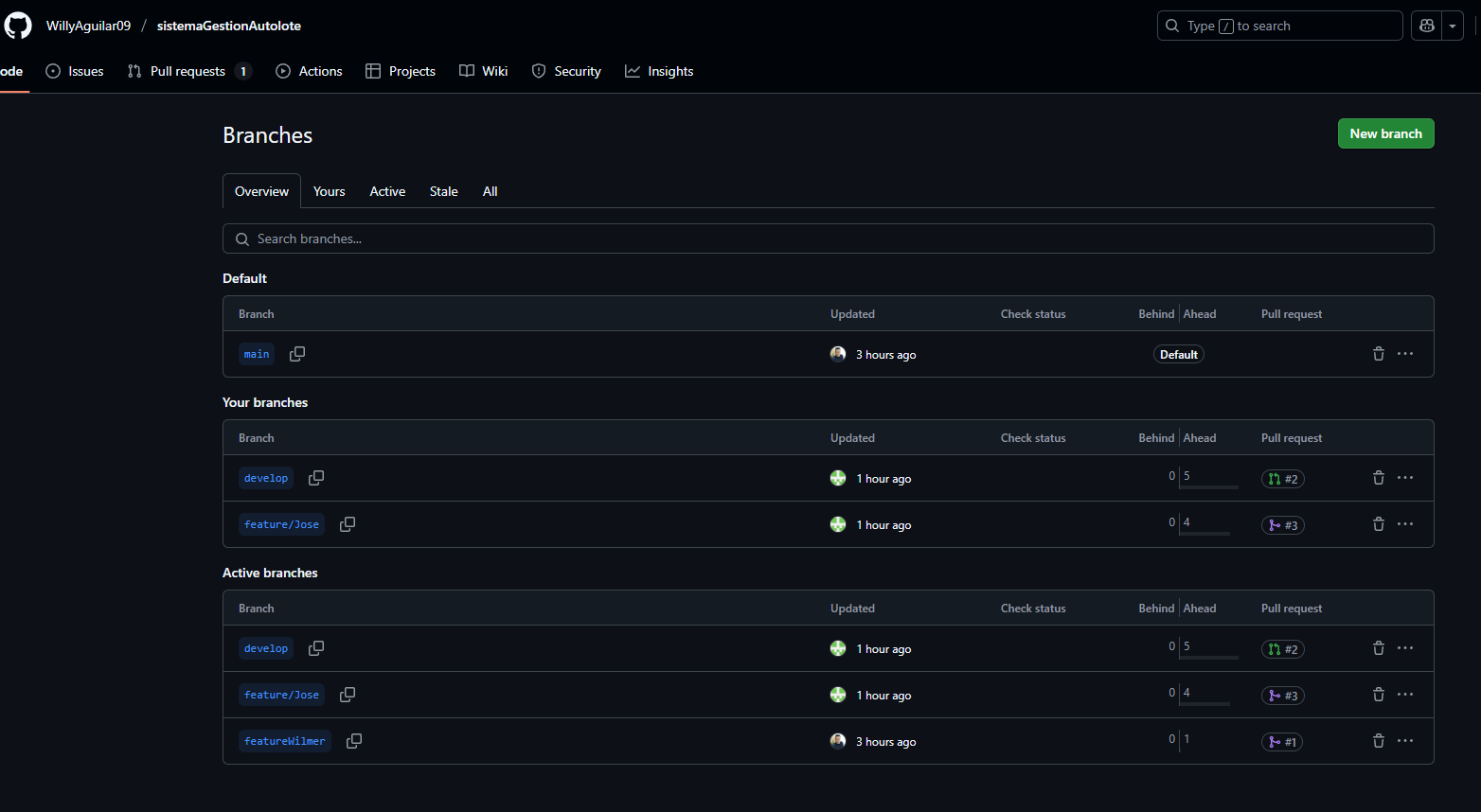
## Modelo Entidad Relación

Diagrama

Descripción generada automáticamente

# Anexo

### Metodología GitHub / ramas



### Metodología GitHub / gitflow

Captura de pantalla de un celular

Descripción generada automáticamente

### Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico Descripción generada automáticamenteMetodología CRUD / get

### Metodología CRUD / put

Interfaz de usuario gráfica, Texto

Descripción generada automáticamente

### Metodología CRUD / post

Interfaz de usuario gráfica, Texto, Correo electrónico

Descripción generada automáticamente

### Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico Descripción generada automáticamenteMetodología CRUD / delete

### Base de datos

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente

### Código

Captura de pantalla de computadora

Descripción generada automáticamente