**Documentación de la API de Wheelz**

Bienvenido a la documentación de la API para el proyecto de alquiler de vehículos Wheelz. Esta API gestiona el backend de una plataforma de alquiler de vehículos, proporcionando funcionalidades para ver la disponibilidad de vehículos, realizar reservas y calcular el costo del alquiler.

**Herramientas y Tecnologías Utilizadas**

* **Java Spring Framework**: La API está desarrollada utilizando el framework Java Spring, que proporciona soluciones robustas y escalables para aplicaciones web.
* **Base de Datos MySql**: Los datos se almacenan en una base de datos MySql, gestionada y accedida mediante consultas SQL.
* **IntelliJ IDEA**: Utilizado como el entorno de desarrollo integrado (IDE) para el desarrollo del proyecto.

**Controladores**

Los controladores descritos a continuación muestran los procesos internos y el manejo de datos dentro del api.

**URL General:** [**https://devps-production.up.railway.app/swagger-ui/index.html#/**](https://devps-production.up.railway.app/swagger-ui/index.html#/)

1. **Controladores de Usuario:**

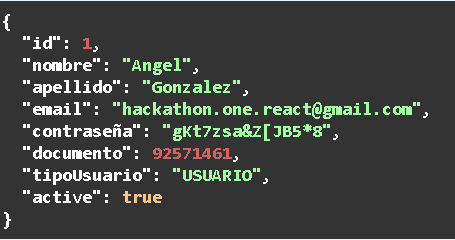
**Metodos Get:**

1. **Obtener Usuario por id:**

* **Ruta**:



* **Descripción**: Nos da toda la información de un usuario en base a su id.
* **Respuesta de la Solicitud**:



1. **Obtener todos los Usuario:**

* **Ruta**:



* **Descripción**: Retorna todos los usuarios en la base de datos.
* **Respuesta de la Solicitud**:



1. **Obtener Usuarios Activos:**

* **Ruta**:



* **Descripción**: N retorna todos los usuarios que estén activos en la plataforma.
* **Respuesta de la Solicitud**:



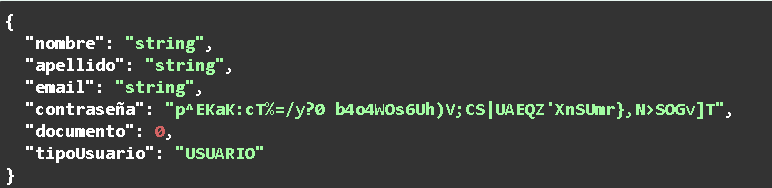
**Metodos Post:**

1. **Registro de Usuarios:**

* **Ruta**:



* **Descripción**: Crea un usuario nuevo. Nos permite validar que ni la contraseña ni el documento estén repetidos, también pide unas especificaciones especiales para la contraseña.
* **Datos Requeridos**:

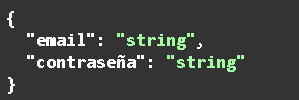


1. **Login de Usuario:**

* **Ruta**:



* **Descripción**: En este endpoint verificamos las credenciales tanto de usuario como del admin para permitirle acceso a los recursos que no son públicos y necesitan autorización.
* **Datos Requeridos**:



**Metodos Patch:**

1. **Desactivar Usuario:**

* **Ruta**:



* **Descripción**: Desactiva un usuario
* **Datos Requeridos**:



1. **Activar Usuario:**

* **Ruta**:



* **Descripción**: Activa un usuario.
* **Datos Requeridos**:



**Metodos Put:**

1. **Modificar Usuario:**

* **Ruta**:



* **Descripción**: Modifica los datos necesarios de un usuario
* **Datos Requeridos**:

No son necesarios todos los datos del Json, solo los que el usuario desee cambiar o actualizar



1. **Controladores de Tipo Cobertura:**

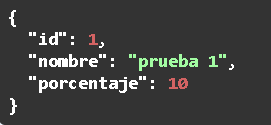
**Metodos Get:**

1. **Obtener Tipo de Cobertura por id:**

* **Ruta**:



* **Descripción**: Nos da toda la información de un tipo de cobertura en base a su id.
* **Respuesta de la Solicitud**:



1. **Obtener todos los Tipos de Cobertura:**

* **Ruta**:



* **Descripción**: Retorna todos los tipos de cobertura almacenados
* **Respuesta de la Solicitud**:



**Metodos Post:**

1. **Crear Tipos de Cobertura:**

* **Ruta**:



* **Descripción**: Crea un tipo de cobertura
* **Datos Solicitados**:



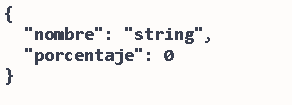
**Metodos Put:**

1. **Modificar Tipos de Cobertura:**

* **Ruta**:



* **Descripción**: Modifica un tipo de cobertura
* **Datos Solicitados**:



**Metodos Delete:**

1. **Eliminar Tipos de Cobertura:**

* **Ruta**:



* **Descripción**: Elimina un tipo de cobertura
* **Datos Solicitados**:



1. **Controladores de Reserva:**

**Metodos Get:**

1. **Obtener Reserva por id:**

* **Ruta**:



* **Descripción**: Nos da toda la información de una reserva en base a su id.
* **Respuesta de la Solicitud**:

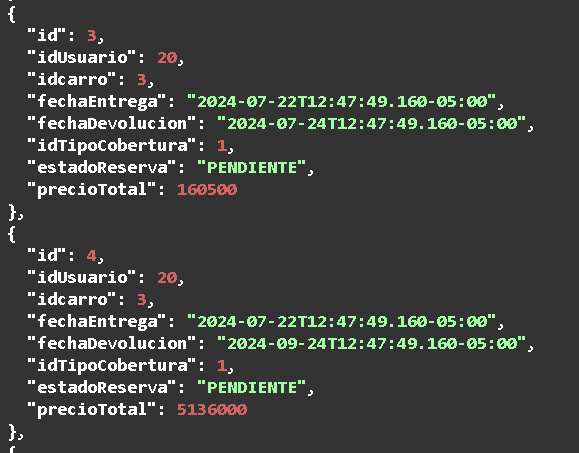
****

1. **Obtener Todas las Reservas:**

* **Ruta**:



* **Descripción**: Nos trae todas las reservas en la base de datos.
* **Respuesta de la Solicitud**:



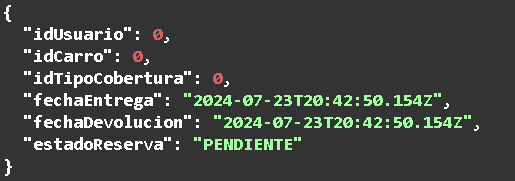
**Metodos Post:**

1. **Crear Reserva:**

* **Ruta**:



* **Descripción**: Nos permite crear una reserva, aca se calcula el valor total de forma automática, tomando el numero de días de la reserva y multiplicándolo por el precio del valor de alquiler diario de un vehiculo, luego
* **Datos Solicitados**:

****

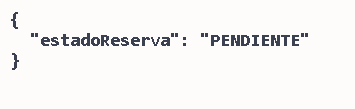
**Metodos Put:**

1. **Modificar Reserva:**

* **Ruta**:



* **Descripción**: Nos permite cambiar el estado de una reserva
* **Datos Solicitados**:



1. **Controladores de Carro:**

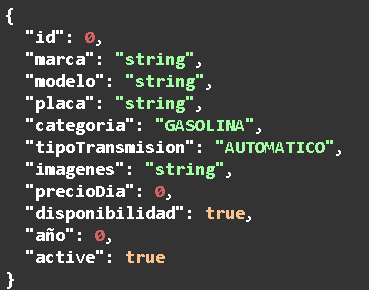
**Metodos Get:**

1. **Obtener Carro por id:**

* **Ruta**:



* **Descripción**: Nos da toda la información de un carro en base a su id.
* **Respuesta de la Solicitud**:

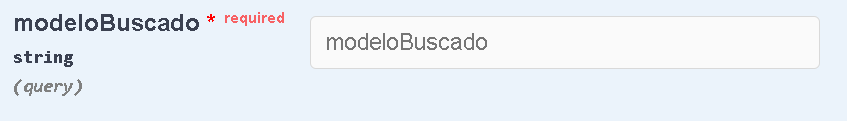


1. **Buscar carro por modelo:**

* **Ruta**:



* **Descripción**: Este endpoint utiliza un stored procedure para retornar los carros por modelo
* **Datos Requeridos**:

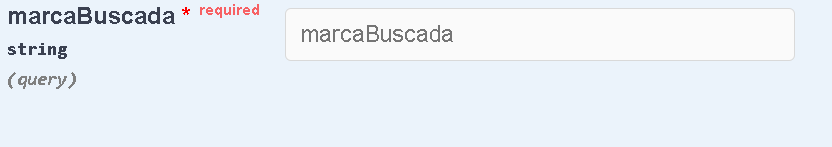


1. **Buscar carro por marca:**

* **Ruta**:



* **Descripción**: Este endpoint utiliza un stored procedure para retornar los carros por marca
* **Datos Requeridos**:

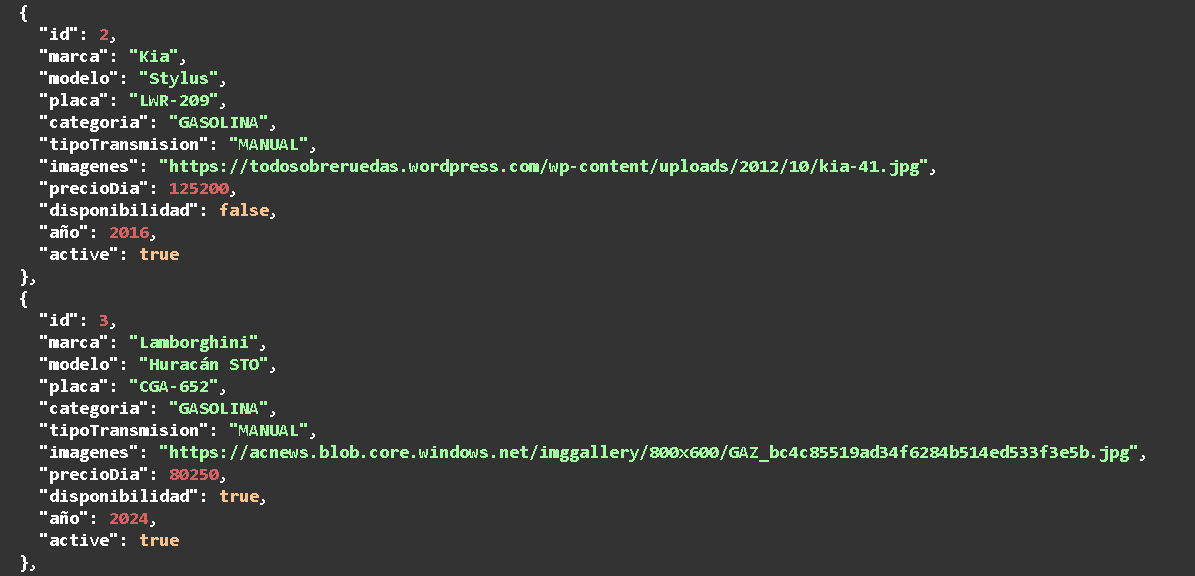


1. **Obtener todos los carros:**

* **Ruta**:



* **Descripción**: Retorna todos los carros en la base de datos
* **Respuesta de la Solicitud**:

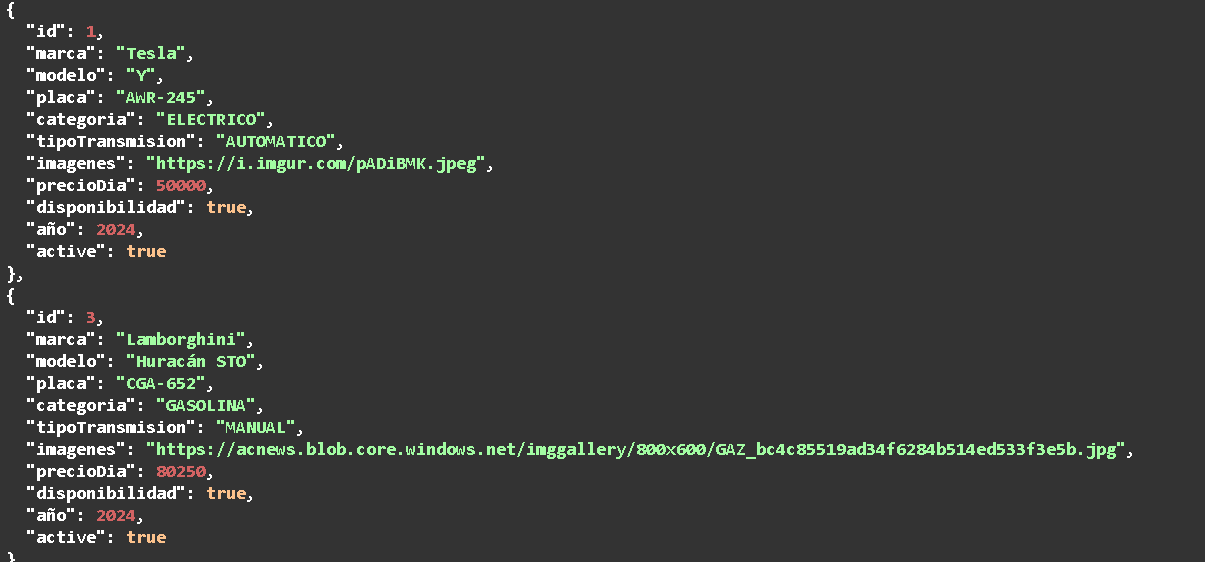


1. **Obtener todos los carros activos:**

* **Ruta**:



* **Descripción**: Retorna todos los carros en la base de datos que esten disponibles y activos
* **Respuesta de la Solicitud**:



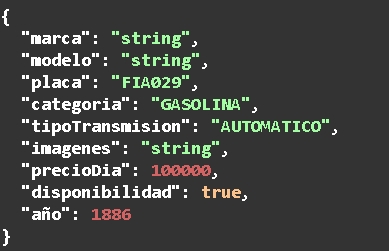
**Metodos Post:**

1. **Crear Carro:**

* **Ruta**:



* **Descripción**: Crea un nuevo carro
* **Datos Requeridos**:



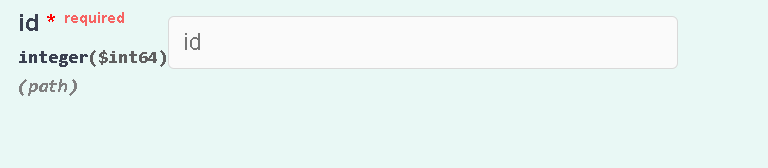
**Metodos Patch:**

1. **Desactivar Carro:**

* **Ruta**:



* **Descripción**: Desactiva un carro
* **Datos Requeridos**:

****

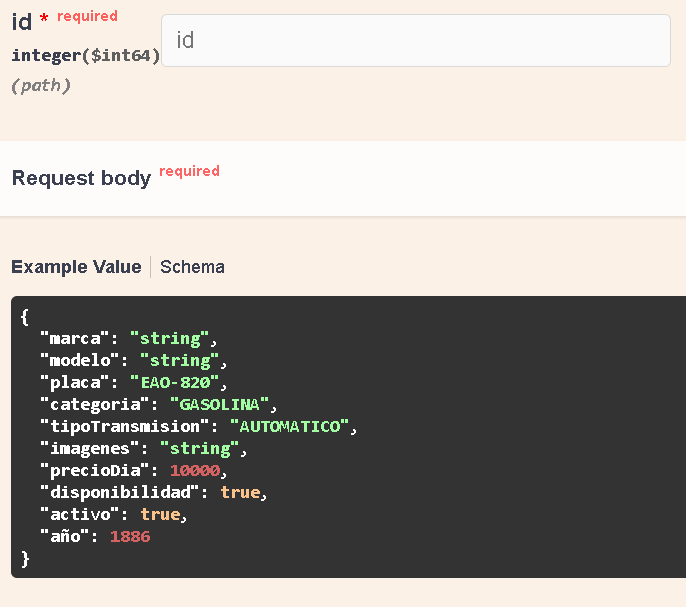
**Metodos Put:**

1. **Modifica un Carro:**

* **Ruta**:



* **Descripción**: Modifica en base a un id los campos necesarios de un carro
* **Datos Requeridos**:



1. **Controladores de Pago:**

*En implementación*

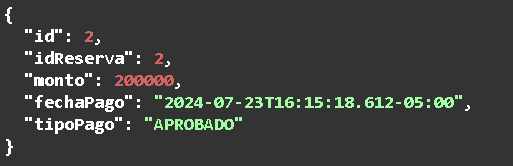
**Metodos Get:**

1. **Obtener Pago por id:**

* **Ruta**:



* **Descripción**: Nos trae un pago por id.
* **Respuesta de la Solicitud**:

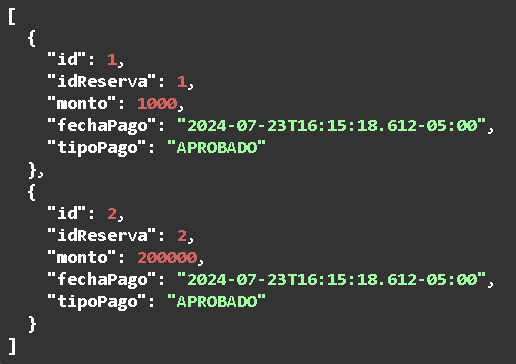


1. **Obtener Todos los pagos:**

* **Ruta**:



* **Descripción**: Nos trae todos los pagos.
* **Respuesta de la Solicitud**:



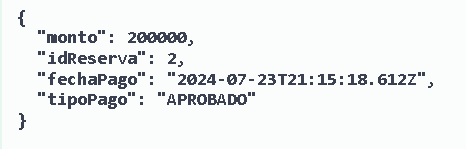
**Metodos Post:**

1. **Obtener Pago por id:**

* **Ruta**:



* **Descripción**: Genera un pago
* **Datos necesarios:**



Pasos para Ejecutar el Proyecto

Clonar el repositorio:

<https://github.com/EdBigpun/wheelz-app.git>

 **Importar el proyecto en IntelliJ IDEA:**

* Abrir IntelliJ IDEA.
* Seleccionar "Open or Import" y elegir la carpeta del proyecto clonado.
* Seguir las instrucciones del asistente para configurar Maven.

 **Configurar la base de datos:**

* Crear una base de datos MySql llamada wheelz\_db.
* Configurar las credenciales en application.properties

 **Iniciar el servidor MySql**.

 **Compilar y ejecutar la aplicación:**

mvn clean install

mvn spring-boot:run

1. **Probar los endpoints con Postman o cualquier herramienta similar.**

**Autenticación**

Para poder acceder a los endpoints, debes autenticarte primero y obtener un token. En Postman, agrega el JSON en el cuerpo de la solicitud, luego en Authorization selecciona la opción de Bearer Token y pega el token para poder tener el acceso.

Esta documentación proporciona una guía completa para configurar, entender y utilizar la API de alquiler de vehículos Wheelz. Para más detalles, revisa el código fuente y los comentarios en el código.