1. Objetivo

Este documento tem o objetivo de orientar usuários e desenvolvedores que porventura venham acessar ou implementar os serviços dispostos por essa API.

2. Contextualizacão

A API Carros-Store fornece endpoints para serviços de CRUD referente ao cadastro de carros. Existem 3 serviços, sendo eles, cadastrar um novo carro, retornar um carro pelo seu id e retornar um lista paginada de carros.

2.1. Entidades Base H2

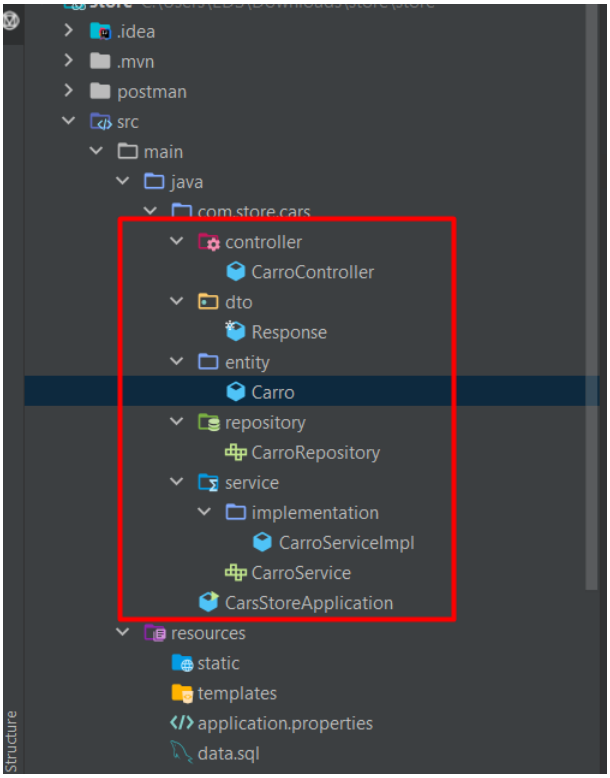
O serviço integra com um banco H2 fazendo uso da tabela Carro. Essa tabela é modelada na aplicação através da classe entidade carro cujo o modelo de dados é exposto abaixo.

Entidade Carro

Nome do parâmetro	Atributo no banco	Descrição	Tipo	Formato	Requerido?
Id	carro.id	Identificador interno único	INTEIRO	Caracteres numéricos	sim
Marca	carro.marca	Marca do carro.	STRING	Texto livre	sim
Modelo	carro.modelo	Modelo do carro.	STRING	Texto livre	sim
Data de cadastro	carro.data_cadastro	Data que indica o dia que o carro foi cadastrado no sistema	DATE	YYYY-MM-DD	sim
Valor	carro.valor	Data e hora do início do atendimento automático na URA.	Decimal(10,2)	Caracteres numéricos	sim

3. Estrutura do projeto

No desenvolvimento desta aplicação foi utilizado a estrutura **Package by Layers**, de modo que possamos agrupar as classes e interfaces de acordo com suas respectivas camadas de atuação.



Pacote	Responsabilidade
controller	Pacote a classe que lida com as requisições vindas do usuário.
dto	Pacote que contém apenas uma classe (Reponse) responsável por definir um padrão de resposta às requisições feitas..
entity	Pacote que contém a classe que define uma representação da tabela carro da base de dados.
repository	Pacote que contém a interface que enviam e retornam dados à base em objetos de domínio na aplicação.
service	Pacote que contém classe e interface com regras de negócios e posterior redirecionamento aos repositories para acesso a base.

4. Serviços

A seguir estão listados os serviços que irão auxiliar no cadastro das informações dos carros em questão:

Operação	Endpoint	Parâmetros Requerimentos	Retorno	Objetivo
POST	/carros	Entidade carro - Dados do carro passados no corpo da requisição	Objeto response contendo no atributo 'data' as informações do carro cadastrado.	Registrar um carro na base.
GET	/carros/{id}	Id - Id do carro passado no path da requisição	Objeto response contendo no atributo 'data' o carro referente ao id passado na requisição.	Retornar o carro referente ao id passado na requisição.
GET	/carros	start e limit- parâmetros da requisição referentes a paginação da lista que será buscada na base base de dados. Se não forem passados, os atributos assumiram os seguintes valores: start = 0 e limit = 5	Lista página de carros.	Atualizar um atendimento finalizado por um cidadão na URA atravéz do id(idOrigem) do atendimento e da data de criação do registro no AWS S3.

4.1. Modelos JSON

- ```
  {
      "marca":"Nissan",
      "modelo":"Sentra",
      "dataCadastro":"2021-01-10",
      "valor":44000
  }
  ```

- ```
  {
  ```

```
    "status": 201,
    "errorCode": null,
    "stacktrace": null,
    "messages": [
        "Created"
    ],
    "data": [
        {
            "id": 8,
            "marca": "Nissan",
            "modelo": "Sentra",
            "dataCadastro": "2021-01-10",
            "valor": 44000
        }
    ]
}
```

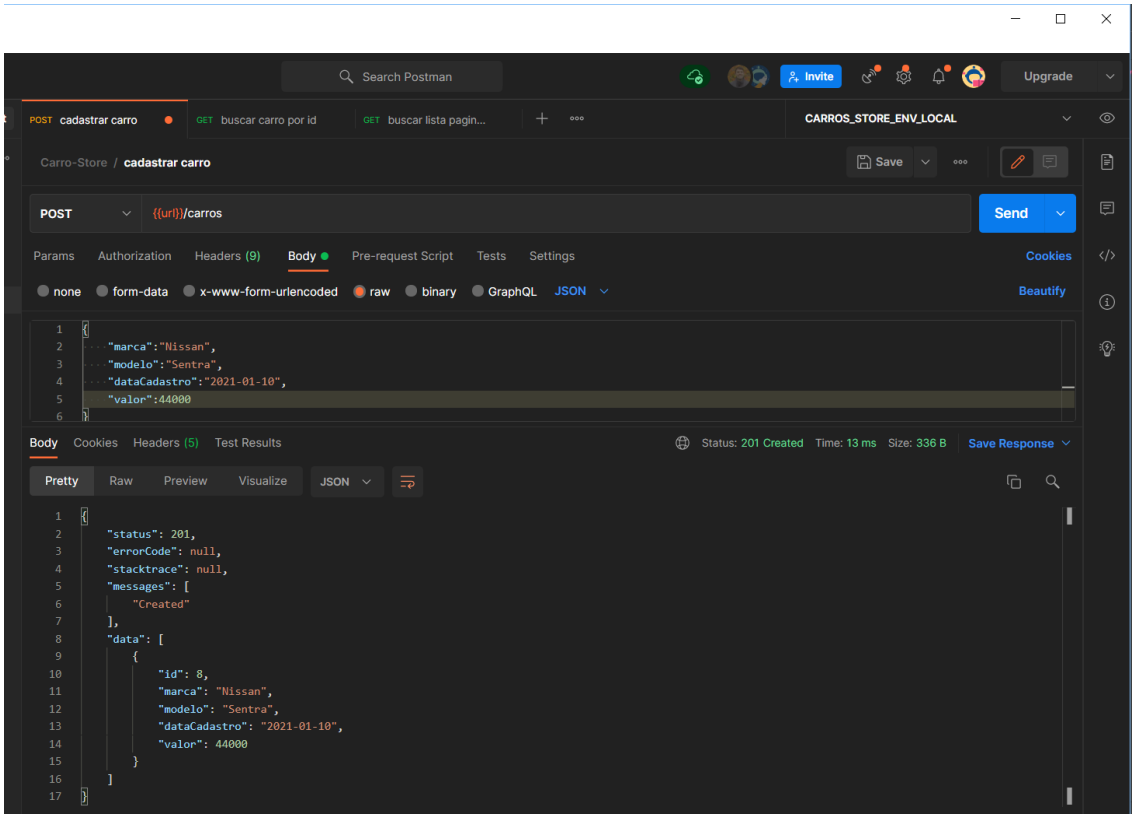

4.2. Tipos de retorno da API

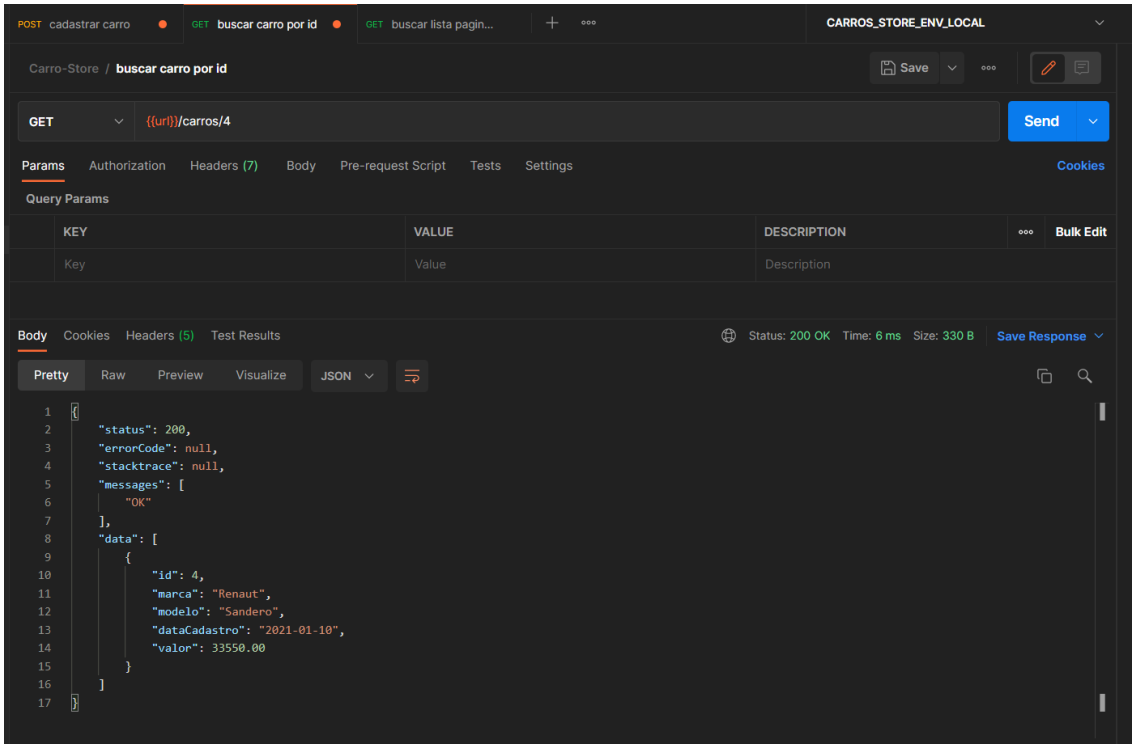
Status Code	Mensagem	Causa
200	"OK" ou "Objeto alterado com sucesso"	- Operação de busca bem sucedida.
201	"Objeto criado com sucesso"	- Dados cadastrado na base.
400	"Invalid request body"	- O JSON no corpo da requisição não se encontra no formato esperado. Possivelmente algum campo obrigatório não está sendo passado ou está mal formatado.
404	"Recurso não encontrado"	- Id origem não foi encontrado na base.
500	"Internal erro"	- Algum erro interno e inesperado.

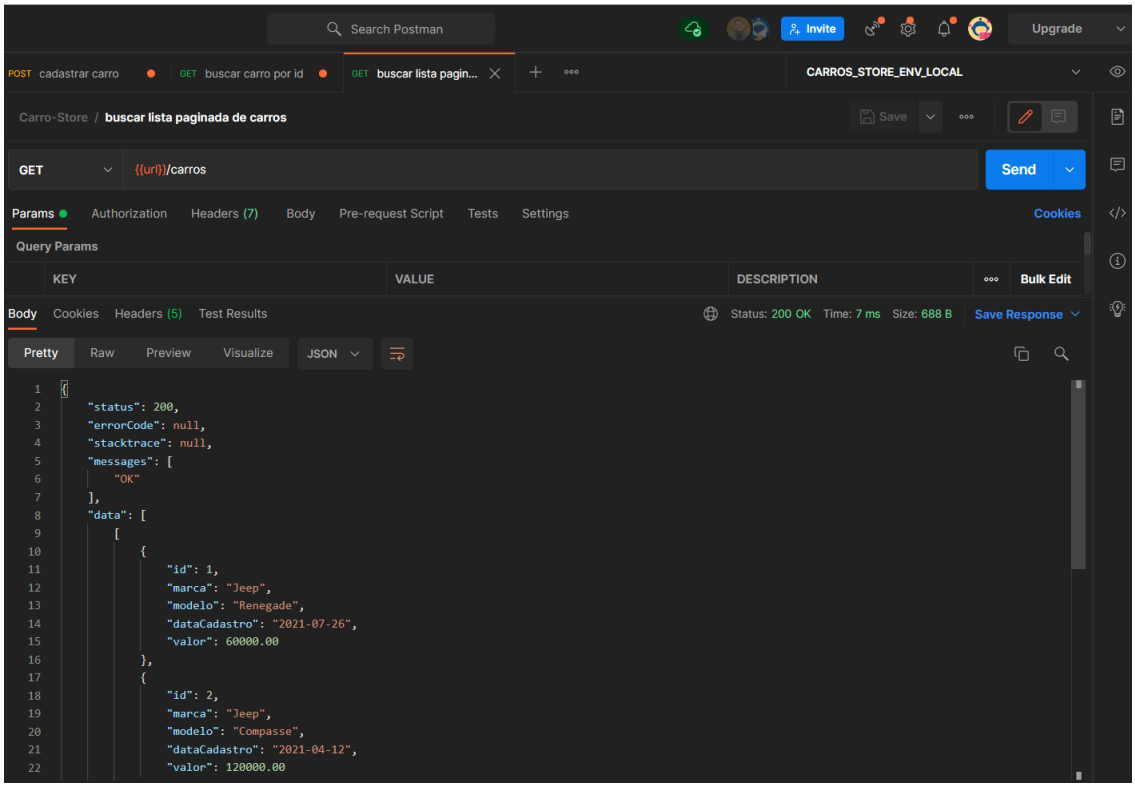
5. Exemplos de Consumo do Serviços

Para facilitar o teste dos recursos desenvolvidos, está disponível na raiz do projeto uma pasta(/postman) contento dois arquivos referente a coleção postman que apontam para os endpoints da aplicação e também um arquivo de variáveis de ambiente. Ambos deverão ser importados na ferramenta Postman para que os testes possam ser efetuados.

5.1. Evidências de execução no postman

- 

- 


Search Postman
Invite
Upgrade
POST cadastrar carro
GET buscar carro por id
GET buscar lista pagin...
CARROS_STORE_ENV_LOCAL
Carro-Store / buscar lista paginada de carros
Save
GET
{{url}}/carros
Send
Params
Authorization
Headers (7)
Body
Pre-request Script
Tests
Settings
Cookies
Query Params
KEY
VALUE
DESCRIPTION
Bulk Edit
Body
Cookies
Headers (5)
Test Results
Status: 200 OK
Time: 7 ms
Size: 688 B
Save Response
Pretty
Raw
Preview
Visualize
JSON
{
"status": 200,
"errorCode": null,
"stacktrace": null,
"messages": [
"OK"
],
"data": [
[
{
"id": 1,
"marca": "Jeep",
"modelo": "Renegade",
"dataCadastro": "2021-07-26",
"valor": 60000.00
},
{
"id": 2,
"marca": "Jeep",
"modelo": "Compasse",
"dataCadastro": "2021-04-12",
"valor": 120000.00