Universidade do Minho
Mestrado Integrado em Engenharia Informática
Processamento e Representação de Conhecimento

Aplicação Web - Ontologia sobre automóveis

Tiago Fraga, A74092

14 de Junho de 2019

Conteúdo

Capítulo 1

Introdução

Com a realização deste trabalho prático pretende-se conceber e modelar uma ontologia sobre um tema à escolha do aluno de forma a guardar os dados na base de dados **Graph-DB**, com o objetivo de criar um servidor *WEB* que suporte pedidos os pedidos à base de dados e forneça a informação resultante à interface desenvolvida em **Vue JS**.

De forma a realizar este trabalho foi preciso seguir um conjunto de passos que irão ser descritos no presente relatório.

Em primeiro lugar, é necessário escolher um tema que será alvo de estudo e tratamento de dados de forma a que quando a ontologia esteja pronta e modelada, seja possivel fazer a migração dos dados e criação de *individuals* da ontologia da forma correta.

A base de dados que irá suportar os dados oriundos da ontologia será o **Graph-DB**, uma base de dados orientada a grafos que suporta corretamente a importação de ficheiros do tipo *Turtle (.ttl)*, que será o tipo da ontologia desenvolvida.

Tendo os dados prontos e sido corretamente inseridos na base de dados, é necessário proceder ao desenvolvimento de *queries* em **SPARQL** de forma a extrair a informação necessária para fornecer ao servidor *WEB*.

O servidor *WEB* foi desenvolvido utilizando a linguagem **Node JS**, e fará a ponte entre a base de dados e a interface.

Por fim, foi desenvolvida a interface utilizando a linguagem **Vue JS**, com o objetivo de apresentar os dados de uma forma mais apelativa ao utilizador, dando-lhe a capacidade de poder navegar na aplicação *WEB* de uma forma rápida, simples e eficaz.

Capítulo 2

Caso de Estudo

2.1 Escolha do Tema

A primeira etapa do trabalho envolve a escolha do tema a estudar.

Para desenvolver modelar uma ontologia é necessário ter informação prévia sobre os dados a tratar, como tal o tema que irei abordar tem de ser algo que mesmo nao tendo um *DataSet* disponível sei que tipo de dados vou tratar.

Desta forma, decidi escolher um tema sobre um domínio que, pessoalmente, sinto que tenho um vasto conhecimento. Para isso, o tema escolhido foi sobre automóveis e suas especificações mecânicas.

Ao escolher este tema, sabia à partida e sem ter acesso a nenhuma *API* de dados ou a nenhum DataSet, qual seria o tipo de informação que iria ter de tratar de forma a modelar a ontologia para o que pretendia desenvolver.

Ao desenvolver uma aplicação *WEB* dentro deste tema, pretendo criar um local onde seja possivel aceder à informação das marcas dos automóveis, às especificações dos automóveis, dos motores que são utilizados nos mesmos, bem como haver a possibilidade de fazer todo o tipo de pesquisas dentro das informações disponíveis.

2.2 Recolha dos Dados

2.2.1 Escolha do DataSet

Após a definição de qual o tema a ser estudado durante a realização do trabalho, foi necessário recolher dados de forma a fazer o povoamento da base de dados com informação robusta.

Numa primeira fase procurei *WebSites* de renome, que me pudessem fornecer a informação que necessitava, tais como : Standvirtual, OLX, entre outros internacionais que forneciam *APis* de dados. No entanto, ou não obtive resposta para aceder às *APIs* ou então, as mesmas eram pagas, por sinal valores elevados.

Deste modo, e após uma pesquisa profunda consegui obter um ficheiro *.csv* com bastante informação sobre o que pretendia.

Este ficheiro tem várias especificações de vários automóveis de várias marcas bem como dos seus motores.

2.2.2 Tratamento dos Dados

Após ter o *DataSet*, foi necessário efetuar um tratamento dos dados, de forma a ter os mesmos preparados e prontos a serem transferidos para a ontologia.

O tratamento de dados foi efetuado utilizando a plataforma *Knime* bem como alguns *scripts* em *python*.

O *dataset* original vem com os seguintes atributos:

- Marca, Modelo, Motor, Código do Motor, Torque, Potencia, Consumo médio, Emissões de CO2, Peso, Cilindrada, Material Bloco/Material dos Cilindros, Grau de compressão, Caixa de Velocidades, Pneus, Tração, Outros.

Foi necessário adicionar o combustível a cada um dos automóveis presentes, bem como fazer a limpeza dos atributos pois havia alguns com falha de valores. Além disso foi feita a generalização do formato dos atributos para facilitar a integração na ontologia.

2.3 Ontologia

2.3.1 Classes

Após a limpeza dos dados, e estes estarem prontos para serem integrados na ontologia, surgiu a fase de criação e modelação da mesma. Para o fazer, recorri ao auxilio da ferramenta **Protegé** que fornece os utensílios necessários para o desenvolvimento de uma ontologia.

A ontologia desenvolvida tem sete classe:

- Marca;
- Modelo;
- Motor;
- Veiculo;
- Pneu;
- Tracção;
- CaixaVelocidades;

Foram definidas estas classes, pois estes iam ser os objetos principais de estudo cujo objetivo era criar relações entre eles.

Cada uma destas classes irá possuir informação autónoma umas das outras, como tal, estas irão ser relacionadas através das relações que irei descrever no tópico a seguir.

2.3.2 Relações

Para interligar todas as classes, com objetivo de obter facilmente e intuitivamente a informação aquando do momento das *queries*, foram criadas várias relações.

Todas as relações criadas possuem inversa.

As relações desenvolvidas foram:

	Dominio	Range	Inversa
temMarca	Veiculo	Marca	eMarcaDe
temModelo	Veiculo	Modelo	eModeloDe
temMotor	Veiculo	Motor	eMotorDe
temPneu	Veiculo	Pneu	ePneuDe
temTracao	Veiculo	Tracao	eTracaoDe
temCaixa	Veiculo	Caixa	eCaixaDe

Tabela 2.1: Relações das Classes.

2.3.3 Atributos

Cada classe possui um conjunto de atributos. Estes atributos irão ter os valores presentes no *DataSet* dos dados.

A classe **Marca**, **Caixa** e **Tracao** irá ser definida pelos atributos nome e descrição. A classe **Veiculo** irá ser definida pelos atributos peso3p e peso5p, que indicam o peso do veiculo para uma carroçaria de 3 portas e 5 portas. A classe **Modelo** irá ser definida pelo atributo nome. A classe **Pneu** irá ser definida pelos atributos jante, largura e ratio. Por fim, a classe **Motor** irá ser definida pelos atributos nome, cilindrada, co2, combustivel, potencia, potenciarpm, torque, torquerpm, consumo, materialBloco e materialCilindros.

2.3.4 Indivíduos

De forma a terminar o modelação da ontologia, foi necessário criar os indivíduos da mesma.

Cada individuo vai pertencer a uma classe, e ter várias relações que como explicado anteriormente, irá relacionar os indivíduos das variadas classes.

Numa primeira fase e para gerar o template base, fiz a criação dos primeiros indivíduos de cada classe.

Posto isto, modifiquei os dados do ficheiro de dados no formato *.csv* para formato *json*.

Com o auxilio de um script em *Node JS* por classe, criei os indivíduos para todas as classes.

Capítulo 3

Aplicação Web

3.1 Base de Dados

Dada por terminada a modelação e criação da ontologia para o trabalho prático, surgiu a etapa de importação da mesma na base de dados *Graph-DB*.

Como a ontologia estava presente num ficheiro *Turtle* e todas a inferências de relações já tinham sido inferidas pelo *Protegé* a importação do ficheiro foi direta.

Posto isto, a base de dados estava pronta para receber as queries do servidor para responder com a informação pedida.

3.2 Servidor

O servidor web está dividido em 2 partes.

O controlador faz a ligação com a base de dados, estando encarregue de injetar as *queries* à mesma recebendo a sua resposta. Tendo a resposta, fica encarregue de a fornecer ao roteador.

Por sua vez, o roteador recebe a resposta do controlador e ficava encarregue de a fornecer à interface que apresentará os dados visualmente apelativos ao utilizador.

3.3 Controlador

O controlador é o módulo que está encarregue de injetar as *queries* à base de dados.

As queries que foram desenvolvidas foram:

- **contaMarcas**
  Devolve o numero total de marcas;

- **contaModelos**
  Devolve o numero total de modelos;

- **contaVeiculos**
  Devolve o numero total de automóveis;

- **contaMotores**
  Devolve o numero total de motores;

- **contaPneus**
  Devolve o numero total de pneus;

- **contaTracoes**
  Devolve o numero total de tipos de tracções;

- **contaCvs**
  Devolve o numero total de caixas de velocidade;

- **todasMarcas**
  Devolve a lista com todas as marcas;

- **todosVeiculos**
  Devolve a lista com todos os automóveis;

- **todosMotores**
  Devolve a lista com todos os motores;

- **todasCaixas**
  Devolve a lista com todas as caixas de velocidade;

- **todasTracoes**
  Devolve a lista com todos os tipos de tracções;

- **todasJantes**
  Devolve a lista com a medida de todas as jantes dos pneus;

- **todasLarguras**
  Devolve a lista com a medida de todas as larguras dos pneus;

- **todosRatios**
  Devolve a lista com a medida de todos os *ratios* dos pneus;

- **pesquisar**
  Dado um objeto *json* com vários parâmetros, devolve uma lista de automóveis que obedece a esses parâmetros. Os parâmetros são: id de uma marca, id de um modelo, o combustível, o id de um tipo de tração, o id de uma caixa de velocidades, e um valor minimo e máximo para os valores de cilindrada, potencia, torque, consumo e emissões de CO2;

- **infoVeiculoHead**
  Devolve a informação total de um dado automóvel. Como parâmetro é passado o seu id;

- **infoVeiculoBody**
  Devolve a lista dos automóveis que utilizam o mesmo motor de um dado automóvel. Como parâmetro é passado o seu id;

- **infoMotorHead**
  Devolve a informação total de um dado motor. Como parâmetro é passado o seu id;

- **infoMotorBodyMarca**
  Devolve a lista de todas as marcas que utilizam um dado motor. Como parâmetro é passado o seu id;

- **infoMotorBodyVeiculos**
  Devolve a lista de todas os automóveis que utilizam um dado motor. Como parâmetro é passado o seu id;

- **infoPneuHead**
  Devolve a informação total de um dado pneu. Como parâmetro é passado o seu id;

- **infoPneuBodyMarca**
  Devolve a lista de todas as marcas que utilizam um dado pneu. Como parâmetro é passado o seu id;

- **infoPneuBodyVeiculos**
  Devolve a lista de todas os automóveis que utilizam um dado pneu. Como parâmetro é passado o seu id;

- **infoCVHead**
  Devolve a informação total de uma dada caixa de velocidades. Como parâmetro é passado o seu id;

- **infoCVBodyMarca**
  Devolve a lista de todas as marcas que utilizam uma dada caixa de velocidades. Como parâmetro é passado o seu id;

- **infoCVBodyVeiculos**
  Devolve a lista de todas os automóveis que utilizam uma dada caixa de velocidades. Como parâmetro é passado o seu id;

- **infoTracaoHead**
  Devolve a informação total de um dado tipo de tração. Como parâmetro é passado o seu id;

- **infoTracaoBodyMarca**
  Devolve a lista de todas as marcas que utilizam um dado tipo de tração. Como parâmetro é

passado o seu id;

- **infoTracaoBodyVeiculos**
  Devolve a lista de todas os automóveis que utilizam um dado tipo de tração. Como parâmetro é passado o seu id;

- **infoMarcaHead**
  Devolve a informação total de uma dado marca. Como parâmetro é passado o seu id;

- **infoMarcaBodyModelos**
  Devolve a lista de todas os automóveis que utilizam uma dada marca. Como parâmetro é passado o seu id;

- **infoMarcaBodyMotores**
  Devolve a lista de todas os motores que uma dada marca utiliza. Como parâmetro é passado o seu id;

- **infoMarcaMaisPotente**
  Devolve o automóvel com mais cavalos de uma dada marca. Como parâmetro é passado o seu id;

- **infoMarcaMaisTorque**
  Devolve o automóvel com mais torque de uma dada marca. Como parâmetro é passado o seu id;

- **infoMarcaMenosConsumo**
  Devolve o automóvel com menor consumo de uma dada marca. Como parâmetro é passado o seu id;

- **infoMarcaMenosEmissoes**
  Devolve o automóvel com menor emissão de gases CO2 de uma dada marca. Como parâmetro é passado o seu id;

3.4 Roteador

O roteador possui as funções que fazem a ponte entre o controlador e a interface.

Cada função do roteador tem definida uma rota que é chamada pela interface, de forma a que esta receba a informação pretendida.

Na maior parte das funções são utilizados pedidos *GET*, tanto com algum parâmetro, como o id de uma marca ou de um automóvel, como sem parâmetros.

Na função encarregue de definir a rota para a *querie* de pesquisa, esta rota está definida segundo um pedido *POST* pois recebe um objeto *json* no corpo do pedido.

3.5 Interface

Para desenvolver a interface da aplicação *WEB* foi utilizada a tecnologia *VUE JS*. Nesta ferramenta, com o auxilio da *framework Vuetify*, foi possivel ter acesso a elementos visuais previamente definidos que auxiliaram no processo de desenvolvimento.

De destacas que foi utilizado a ferramenta **Axios** para fazer os pedidos ao roteador do servidor de forma a obter a informação desejada da base de dados.

Capítulo 4

Conclusão

Neste capítulo dou por terminado a realização do trabalho prático para a unidade curricular de processamento e representação de conhecimento inserida no perfil de processamento de linguagens e conhecimento do quarto ano do mestrado integrado em engenharia informática.

Após o termino do mesmo posso afirmar que o processo de escolha de tema foi das etapas mais difíceis do projecto uma vez que ponderei durante bastante tempo que informação podia obter para elaborar uma aplicação robusta, eficaz e apelativa.

Mesmo no momento em que decidi por estudar o domínio automobilístico, deparei me com vários problemas, entre eles, a dificuldade em obter dados. Os conjuntos de dados mais interessantes e ricos eram na sua totalidade pagos, e a preços exorbitantes. No entanto, e após uma busca intensiva consegui obter dados que me satisfaziam para elaborar esta ontologia bem como a aplicação *WEB*.

Como trabalho futuro, penso que uma das etapas a seguir seria o aumento da ontologia com novos dados, de forma a poder alargar a mesma uma enciclopédia automóvel com dados vastos e interessantes para os entusiastas do mundo automobilístico puderem usufruir.

Apêndice A

Anexo

A.1 Ontologia

A.1.1 Adicionar atributo combustivel

Script em python para adicionar ao *DataSet* o combustível de cada automóvel.

```
1  import csv
2  import re
3
4  ficheiro_read = "dados4py.csv"
5  ficheiro_write = "dados4json.csv"
6
7  lista = ['dCi', 'HDi', 'TDI']
8
9
10 with open(ficheiro_write, mode='w') as file_writer:
11     writer = csv.writer(file_writer, delimiter=',', quotechar='"', quoting=
           csv.QUOTE_MINIMAL)
12     with open(ficheiro_read) as csv_file:
13         texto = csv.reader(csv_file, delimiter=',')
14         line_count = 0
15         for row in texto:
16             if line_count == 0:
17             #Escrever as colunas
18                 writer.writerow([row[0],row[1],row[2],row[3],row[4],row[5],
                       row[6],row[7],row[8],row[9],row[10],row[11],row[12],row
                       [13],row[14],row[15],row[16],row[17],'Combustivel'])
19                 line_count += 1
20             else:
```

```
                    line_count += 1
                    verificar = row[2]
                    if 'dCi' in verificar:
                        writer.writerow([row[0],row[1],row[2],row[3],row[4],row
                            [5],row[6],row[7],row[8],row[9],row[10],row[11],row
                            [12],row[13],row[14],row[15],row[16],row[17],'gasoleo
                            '])
                    elif 'HDi' in verificar:
                        writer.writerow([row[0],row[1],row[2],row[3],row[4],row
                            [5],row[6],row[7],row[8],row[9],row[10],row[11],row
                            [12],row[13],row[14],row[15],row[16],row[17],'gasoleo
                            '])
                    elif 'TDI' in verificar:
                        writer.writerow([row[0],row[1],row[2],row[3],row[4],row
                            [5],row[6],row[7],row[8],row[9],row[10],row[11],row
                            [12],row[13],row[14],row[15],row[16],row[17],'gasoleo
                            '])
                    else:
                        writer.writerow([row[0],row[1],row[2],row[3],row[4],row
                            [5],row[6],row[7],row[8],row[9],row[10],row[11],row
                            [12],row[13],row[14],row[15],row[16],row[17],'
                            gasolina'])
```

A.1.2 Adicionar código do motor

Script em python para adicionar ao *DataSet* o os códigos dos motores em falta.

```
import csv
import re

ficheiro_read = "dados4code.csv"
ficheiro_write = "dados4json.csv"
codigos = {}
string = "PRC"
valores = 0

with open(ficheiro_read) as csv_file:
    texto = csv.reader(csv_file, delimiter=';')
    line_count = 0
    for row in texto:
        if line_count == 0:
            line_count += 1
        else:
            line_count += 1
```

```
18              engine = re.sub(r"\s\s","",row[2])
19              code = row[3]
20              if code == "":
21                  code = string + str(valores)
22                  valores += 1
23              value = codigos.get(engine)
24              if value:
25                  if value == code:
26                      print("Codigo igual")
27                  else:
28                      print("Codigo diferente: " + str(line_count))
29              else:
30                  codigos[engine] = code
31
32  with open(ficheiro_write, mode='w') as file_writer:
33      writer = csv.writer(file_writer, delimiter=',', quotechar='"', quoting=
            csv.QUOTE_MINIMAL)
34      with open(ficheiro_read) as csv_file:
35          texto = csv.reader(csv_file, delimiter=';')
36          line_count = 0
37          for row in texto:
38              if line_count == 0:
39              #Escrever as colunas
40                  writer.writerow([row[0],row[1],row[2],row[3],row[4],row[5],
                        row[6],row[7],row[8],row[9],row[10],row[11],row[12],row
                        [13],row[14],row[15],row[16],row[17],row[18],row[19],row
                        [20],row[21]])
41                  line_count += 1
42              else:
43                  line_count += 1
44                  engine = re.sub(r"\s\s","",row[2])
45                  code = row[3]
46                  if code == "":
47                      writer.writerow([row[0],row[1],engine,codigos[engine],
                            row[4],row[5],row[6],row[7],row[8],row[9],row[10],row
                            [11],row[12],row[13],row[14],row[15],row[16],row[17],
                            row[18],row[19],row[20],row[21]])
48                  else:
49                      writer.writerow([row[0],row[1],row[2],row[3],row[4],row
                            [5],row[6],row[7],row[8],row[9],row[10],row[11],row
                            [12],row[13],row[14],row[15],row[16],row[17],row[18],
                            row[19],row[20],row[21]])
```

A.1.3 Indivíduos da classe Marca

Script em *Node JS* para criar os indivíduos da classe Marca.

```
1
2  const jsonfile = require('jsonfile')
3  const file = './Dados/dados.json'
4
5  const marcas = ['Renault']
6  const modelos = ['twingo']
7  const motores = ['d7f']
8  const caixas = ['MT']
9  const tracoes = ['2wd']
10 const pneus = ['14_165_65']
11
12 valores = 2
13
14 jsonfile.readFile(file)
15     .then(obj =>{
16
17         console.log("### Marcas ###\n")
18         for(var i=0;i<obj.length;i++){
19             if(!marcas.includes(obj[i].Brand)){
20                 var marca = "";
21                 marca += ":m_" + obj[i].Brand.replace(/\W/g, '_') + " rdf:
                       type owl:NamedIndividual , :Marca; \n";
22                 marca += "\t:descricao \""+"\";\n";
23                 marca += "\t:nome \""+ obj[i].Brand +"\".\n";
24                 marcas.push(obj[i].Brand)
25                 console.log(marca)
26             }
27         }
28     })
29     .catch(error => console.error(error))
```

A.1.4 Indivíduos da classe Modelo

Script em *Node JS* para criar os indivíduos da classe Modelo.

```
1
2  const jsonfile = require('jsonfile')
3  const file = './Dados/dados.json'
4
5  const marcas = ['Renault']
```

```
const modelos = ['twingo']
const motores = ['d7f']
const caixas = ['MT']
const tracoes = ['2wd']
const pneus = ['14_165_65']

valores = 2

jsonfile.readFile(file)
    .then(obj =>{

        console.log("### Modelos ###\n")
        for(var i=0;i<obj.length;i++){
            if(!modelos.includes(obj[i].Vehicle)){
                var modelo = "";
                modelo += ":mo_" + String(obj[i].Vehicle).replace(/\W/g, '_'
                    ) + " rdf:type owl:NamedIndividual , :Modelo; \n";
                modelo += "\t:descricao \""+"\";\n";
                modelo += "\t:nome \""+ obj[i].Vehicle +"\".\n";
                modelos.push(obj[i].Vehicle)
                console.log(modelo)
            }
        }
    })
    .catch(error => console.error(error))
```

A.1.5 Indivíduos da classe Veiculo

Script em *Node JS* para criar os indivíduos da classe Veiculo.

```
const jsonfile = require('jsonfile')
const file = './Dados/dados.json'

const marcas = ['Renault']
const modelos = ['twingo']
const motores = ['d7f']
const caixas = ['MT']
const tracoes = ['2wd']
const pneus = ['14_165_65']

valores = 2

jsonfile.readFile(file)
    .then(obj =>{
```

```
15
16          console.log("### Modelos ###\n")
17          for(var i=0;i<obj.length;i++){
18              var carro = "";
19              carro += ":v_" + valores + " rdf:type owl:NamedIndividual , :
                    Veiculo; \n";
20              carro += "\t:temCaixa :cv_"+ obj[i].GearBox.replace(/\W/g, '_')
                    +" ;\n";
21              carro += "\t:temMarca :m_"+ obj[i].Brand.replace(/\W/g, '_') +"
                    ;\n";
22              carro += "\t:temModelo :mo_"+ String(obj[i].Vehicle).replace(/\W
                    /g, '_') +" ;\n";
23              carro += "\t:temMotor :mot_"+ obj[i].Engine_code.replace(/\W/g,
                    '_') +" ;\n";
24              jante = String(obj[i].Jante)
25              if(jante == "" | jante == "?"){jante = 0}
26              largura = String(obj[i].Largura)
27              if(largura == "" | largura == "?"){largura = 0}
28              ratio = String(obj[i].Ratio)
29              if(ratio == "" | ratio == "?"){ratio = 0}
30              pneu = jante + "_" + largura + "_" + ratio
31              carro += "\t:temPneu :pn_"+ pneu +" ;\n";
32              carro += "\t:temTracao :tr_"+ obj[i].Traction.replace(/\W/g, '_'
                    ) +" ;\n";
33              carro += "\t:descricao \""+"\" ;\n";
34              peso3p = obj[i].Weight3p
35              if(peso3p == ""| peso3p == "?"){peso3p = 0}
36              carro += "\t:peso3p "+ peso3p +" ;\n";
37              peso5p = obj[i].Weight5p
38              if(peso5p == ""| peso5p == "?"){peso5p = 0}
39              carro += "\t:peso5p "+ peso5p +" .\n";
40              valores += 1
41              console.log(carro)
42          }
43      })
44      .catch(error => console.error(error))
```

A.1.6 Indivíduos da classe Motor

Script em *Node JS* para criar os indivíduos da classe Motor.

```
1 const jsonfile = require('jsonfile')
2 const file = './Dados/dados.json'
3
```

```
const marcas = ['Renault']
const modelos = ['twingo']
const motores = ['D7F']
const caixas = ['MT']
const tracoes = ['2wd']
const pneus = ['14_165_65']

valores = 2

jsonfile.readFile(file)
    .then(obj =>{

        console.log("### Motores ###\n")
        for(var i=0;i<obj.length;i++){
            if(!motores.includes(obj[i].Engine_code)){
                var motor = "";
                motor += ":mot_" + String(obj[i].Engine_code).replace(/\W/g,
                    '_') + " rdf:type owl:NamedIndividual , :Motor; \n";
                motor += "\t:cilindrada "+obj[i].Displacement+" ;\n";
                co2 = obj[i].co2
                if(co2 == "" | co2 == "?"){
                    motor += "\t:co2 "+0+" ;\n";
                }else{
                    motor += "\t:co2 "+obj[i].co2+" ;\n";
                }
                motor += "\t:codigo \""+obj[i].Engine_code+"\";\n";
                motor += "\t:combustivel \""+obj[i].Combustivel+"\";\n";
                compressao = obj[i].Compression
                if(compressao == "" | compressao == "?"){
                    motor += "\t:compressao "+0+" ;\n";
                }else{
                    motor += "\t:compressao "+obj[i].Compression+" ;\n";
                }
                consumo = obj[i].fuel
                if(consumo == "" | consumo == "?"){
                    motor += "\t:consumo "+0+" ;\n";
                }else{
                    motor += "\t:consumo "+obj[i].fuel+" ;\n";
                }
                motor += "\t:descricao \""+"\";\n";
                motor += "\t:materialBloco \""+obj[i].Block_material+"\";\n"
                    ;
```

```
44                  motor += "\t:materialCilindros \""+obj[i].Cylinder_head+"
                        \";\n";
45                  motor += "\t:power "+obj[i].Power+" ;\n";
46                  motor += "\t:power_rpm "+obj[i].Power_rpm+" ;\n";
47                  motor += "\t:torque "+obj[i].Torque+" ;\n";
48                  motor += "\t:torque_rpm "+obj[i].Torque_rpm+" ;\n";
49                  motor += "\t:nome \""+obj[i].Engine+"\".\n";
50                  motores.push(obj[i].Engine_code)
51                  console.log(motor)
52              }
53          }
54      })
55      .catch(error => console.error(error))
```

A.1.7 Indivíduos da classe Pneu

Script em *Node JS* para criar os indivíduos da classe Pneu.

```
1  const jsonfile = require('jsonfile')
2  const file = './Dados/dados.json'
3
4  const marcas = ['Renault']
5  const modelos = ['twingo']
6  const motores = ['D7F']
7  const caixas = ['MT']
8  const tracoes = ['2wd']
9  const pneus = ['14_165_65']
10
11 valores = 2
12
13 jsonfile.readFile(file)
14     .then(obj =>{
15
16         console.log("### Pneus ###\n")
17         for(var i=0;i<obj.length;i++){
18             var match = String(obj[i].Jante) + "_" + String(obj[i].Largura)
                   + "_" +String(obj[i].Ratio)
19             if(!pneus.includes(match)){
20                 var pneu = "";
21                 pneu += ":pn_" + match + " rdf:type owl:NamedIndividual , :
                       Pneu; \n";
22                 pneu += "\t:descricao \""+"\";\n";
23                 pneu += "\t:jante "+obj[i].Jante+" ;\n";
24                 pneu += "\t:jante "+obj[i].Largura+" ;\n";
```

```
                    pneu += "\t:jante "+obj[i].Ratio+" .\n";

                    pneus.push(match)
                    console.log(pneu)
                }
            }
        })
        .catch(error => console.error(error))
```

A.1.8 Indivíduos da classe Caixa de Velocidades

Script em *Node JS* para criar os indivíduos da classe Caixa de Velocidades.

```
const jsonfile = require('jsonfile')
const file = './Dados/dados.json'

const marcas = ['renault']
const modelos = ['twingo']
const motores = ['d7f']
const caixas = ['MT']
const tracoes = ['2wd']
const pneus = ['14_165_65']

valores = 2

jsonfile.readFile(file)
    .then(obj =>{

        console.log("### GearBoxes ###\n")
        for(var i=0;i<obj.length;i++){
            if(!caixas.includes(obj[i].GearBox)){
                var caixa = "";
                caixa += ":cv_" + obj[i].GearBox.replace(/\W/g, '_') + " rdf
                    :type owl:NamedIndividual , :CaixaVelocidades; \n";
                caixa += "\t:descricao \""+"\";\n";
                caixa += "\t:tipo \""+ obj[i].GearBox +"\".\n";
                caixas.push(obj[i].GearBox)
                console.log(caixa)
            }
        }
    })
    .catch(error => console.error(error))
```

A.1.9 Indivíduos da classe Tracção

Script em *Node JS* para criar os indivíduos da classe Tracção.

```
const jsonfile = require('jsonfile')
const file = './Dados/dados.json'

const marcas = ['Renault']
const modelos = ['twingo']
const motores = ['D7F']
const caixas = ['MT']
const tracoes = ['2wd']
const pneus = ['14_165_65']

valores = 2

jsonfile.readFile(file)
    .then(obj =>{

        console.log("### Tracoes ###\n")
        for(var i=0;i<obj.length;i++){
            if(!tracoes.includes(obj[i].Traction)){
                var tracao = "";
                tracao += ":pn_" + obj[i].Traction.replace(/\W/g, '_') + "
                    rdf:type owl:NamedIndividual , :Tracao; \n";
                tracao += "\t:descricao \""+"\";\n";
                tracao += "\t:nome \""+obj[i].Traction+"\";\n";
                tracao += "\t:tipo \""+"\".\n";

                tracoes.push(obj[i].Traction)
                console.log(tracao)
            }
        }
    })
    .catch(error => console.error(error))
```

A.2 Servidor Web

A.2.1 Controlador

```
const axios = require('axios');
const Carro = module.exports;

```

```
4  normalize = function(response) {
5      return response.results.bindings.map(obj =>
6          Object.entries(obj)
7              .reduce((new_obj, [k,v]) => (new_obj[k] = v.value, new_obj),
8                      new Object()));
9  };
10
11 async function execQuery (query){
12     try{
13         var enconded = encodeURIComponent(query)
14         response = await axios.get("http://localhost:7200/repositories/
               automoveis" + "?query=" + enconded);
15         return(normalize(response.data));
16     }
17     catch(error){
18         return("Erro: " + error)
19     }
20 }
21
22 //------------------------------ HOME ------------------------------
23
24 Carro.contaMarcas = async () => {
25     const query = `PREFIX : <http://prc.di.uminho.pt/2019/automoveis#>
26                    select (count(?s) as ?marcas) where {
27                        ?s a :Marca.
28                    }`
29
30     var res = await execQuery(query);
31     return res;
32 };
33
34 Carro.contaModelos = async () => {
35     const query = `PREFIX : <http://prc.di.uminho.pt/2019/automoveis#>
36                    select (count(?s) as ?modelos) where {
37                        ?s a :Modelo.
38                    }`
39
40     var res = await execQuery(query);
41     return res;
42 };
43
44 Carro.contaVeiculos = async () => {
45     const query = `PREFIX : <http://prc.di.uminho.pt/2019/automoveis#>
```

```
46                      select (count(?s) as ?veiculos) where {
47                          ?s a :Veiculo.
48                      }`
49
50      var res = await execQuery(query);
51      return res;
52  };
53
54  Carro.contaMotores = async () => {
55      const query = `PREFIX : <http://prc.di.uminho.pt/2019/automoveis#>
56                      select (count(?s) as ?motores) where {
57                          ?s a :Motor.
58                      }`
59
60      var res = await execQuery(query);
61      return res;
62  };
63
64  Carro.contaPneus = async () => {
65      const query = `PREFIX : <http://prc.di.uminho.pt/2019/automoveis#>
66                      select (count(?s) as ?pneus) where {
67                          ?s a :Pneu.
68                      }`
69
70      var res = await execQuery(query);
71      return res;
72  };
73
74  Carro.contaTracoes = async () => {
75      const query = `PREFIX : <http://prc.di.uminho.pt/2019/automoveis#>
76                      select (count(?s) as ?tracoes) where {
77                          ?s a :Tracao.
78                      }`
79
80      var res = await execQuery(query);
81      return res;
82  };
83
84  Carro.contaCvs = async () => {
85      const query = `PREFIX : <http://prc.di.uminho.pt/2019/automoveis#>
86                      select (count(?s) as ?cvs) where {
87                          ?s a :CaixaVelocidades.
88                      }`
```

```
89
90      var res = await execQuery(query);
91      return res;
92  };
93
94  //------------------------------ Pagina MARCAS ------------------------------
95
96  Carro.todasMarcas = async () => {
97      const query = `PREFIX : <http://prc.di.uminho.pt/2019/automoveis#>
98                      select ?nome where {
99                          ?m a :Marca.
100                         ?m :nome ?nome
101                     }`
102
103     var res = await execQuery(query);
104     return res;
105 };
106
107
108
109 //------------------------------ Pagina Veiculos ------------------------------
110
111 Carro.todosVeiculos = async () => {
112     const query = `PREFIX : <http://prc.di.uminho.pt/2019/automoveis#>
113                     select ?v ?nomeMarca ?nomeModelo ?nomeMotor where {
114                         ?v a :Veiculo.
115                         ?v :temMarca ?m.
116                         ?m :nome ?nomeMarca.
117                         ?v :temModelo ?mod.
118                         ?mod :nome ?nomeModelo.
119                         ?v :temMotor ?mot.
120                         ?mot :nome ?nomeMotor.
121                     }`
122     var res = await execQuery(query);
123     return res;
124 };
125
126 //------------------------------ Pagina Motores ------------------------------
127
128 Carro.todosMotores = async () => {
129     const query = `PREFIX : <http://prc.di.uminho.pt/2019/automoveis#>
130                     select ?mot ?nome ?cod ?comb where {
131                         ?mot a :Motor.
```

```
132                         ?mot :nome ?nome.
133                         ?mot :codigo ?cod.
134                         ?mot :combustivel ?comb
135                     }`
136     var res = await execQuery(query);
137     return res;
138 };
139
140
141 //------------------------------ Pagina Caixas de Velocidade
    ------------------------------
142
143 Carro.todasCaixas = async () => {
144     const query = `PREFIX : <http://prc.di.uminho.pt/2019/automoveis#>
145                     select ?cvs ?nome where {
146                         ?cvs a :CaixaVelocidades.
147                         ?cvs :tipo ?nome.
148                     }`
149     var res = await execQuery(query);
150     return res;
151 };
152
153
154 //------------------------------ Pagina Tracoes ------------------------------
155
156 Carro.todasTracoes = async () => {
157     const query = `PREFIX : <http://prc.di.uminho.pt/2019/automoveis#>
158                     select ?tr ?nome where {
159                         ?tr a :Tracao.
160                         ?tr :nome ?nome.
161                     }`
162     var res = await execQuery(query);
163     return res;
164 };
165
166 //------------------------------ Pagina Pesquisa ------------------------------
167
168
169
170 Carro.todasJantes = async () => {
171     const query = `PREFIX : <http://prc.di.uminho.pt/2019/automoveis#>
172                     select distinct ?jante where {
173                         ?pn a :Pneu.
```

```
174                         ?pn :jante ?jante
175                     }`
176     var res = await execQuery(query);
177     return res;
178 };
179
180 Carro.todasLarguras = async () => {
181     const query = `PREFIX : <http://prc.di.uminho.pt/2019/automoveis#>
182                     select distinct ?largura where {
183                         ?pn a :Pneu.
184                         ?pn :largura ?largura
185                     }`
186     var res = await execQuery(query);
187     return res;
188 };
189
190 Carro.todosRatios = async () => {
191     const query = `PREFIX : <http://prc.di.uminho.pt/2019/automoveis#>
192                     select distinct ?ratio where {
193                         ?pn a :Pneu.
194                         ?pn :ratio ?ratio
195                     }`
196     var res = await execQuery(query);
197     return res;
198 };
199
200
201
202 //------------------------------ Pagina Pesquisa ------------------------------
203
204
205
206 Carro.pesquisar = async (pesquisa) => {
207     const query = `PREFIX : <http://prc.di.uminho.pt/2019/automoveis#>
208     select ?v ?nomeMarca ?nomeModelo ?nomeMotor where {
209         ?v a :Veiculo.
210         ?v :temMarca ${pesquisa.marca}.
211         ${pesquisa.marca} :nome ?nomeMarca.
212         ?v :temModelo ${pesquisa.modelo}.
213         ${pesquisa.modelo} :nome ?nomeModelo.
214         ?v :temMotor ?motor.
215         ?motor :nome ?nomeMotor.
216         ?motor :combustivel ${pesquisa.combustivel}.
```

```
217         ?v :temTracao ${pesquisa.tracao}.
218         ?v :temCaixa ${pesquisa.caixa}.
219         ?motor :materialBloco ${pesquisa.mb}.
220         ?motor :cilindrada ?cilindrada
221         FILTER(?cilindrada >= ${pesquisa.minCilindrada})
222         FILTER(?cilindrada < ${pesquisa.maxCilindrada})
223         ?motor :power ?potencia
224         FILTER(?potencia >= ${pesquisa.minPotencia})
225         FILTER(?potencia < ${pesquisa.maxPotencia})
226         ?motor :torque ?torque
227         FILTER(?torque >= ${pesquisa.minTorque})
228         FILTER(?torque < ${pesquisa.maxTorque})
229         ?motor :consumo ?consumo
230         FILTER(?consumo >= ${pesquisa.minConsumo})
231         FILTER(?consumo < ${pesquisa.maxConsumo})
232         ?motor :co2 ?co2
233         FILTER(?co2 >= ${pesquisa.minEmissoes})
234         FILTER(?co2 < ${pesquisa.maxEmissoes})
235     }`
236     var res = await execQuery(query);
237     return res;
238 };
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
```

```
260
261
262
263
264
265
266
267
268 //------------------------------ Pagina P1 -> Info Veiculo
        ------------------------------
269
270
271 Carro.infoVeiculoHead = async (idVeiculo) => {
272     const query = `PREFIX : <http://prc.di.uminho.pt/2019/automoveis#>
273     select ?nomeMarca ?nomeModelo ?motor ?nomeMotor ?codigo ?cilindrada ?
            combustivel ?co2 ?compressao ?consumo ?mb ?mc ?power ?prpm ?torque ?
            trpm ?peso3p ?peso5p ?caixa ?nomeCaixa ?tracao ?nomeTracao ?pneu ?
            jante ?largura ?ratio where {
274         :${idVeiculo} :temMarca ?marca.
275         ?marca :nome ?nomeMarca.
276         :${idVeiculo} :temModelo ?modelo.
277         ?modelo :nome ?nomeModelo.
278         :${idVeiculo} :temMotor ?motor.
279         ?motor :nome ?nomeMotor.
280         ?motor :codigo ?codigo.
281         ?motor :cilindrada ?cilindrada.
282         ?motor :combustivel ?combustivel.
283         ?motor :co2 ?co2.
284         ?motor :compressao ?compressao.
285         ?motor :consumo ?consumo.
286         ?motor :materialBloco ?mb.
287         ?motor :materialCilindros ?mc.
288         ?motor :power ?power.
289         ?motor :power_rpm ?prpm.
290         ?motor :torque ?torque.
291         ?motor :torque_rpm ?trpm.
292         :${idVeiculo} :peso3p ?peso3p.
293         :${idVeiculo} :peso5p ?peso5p.
294         :${idVeiculo} :temCaixa ?caixa.
295         ?caixa :tipo ?nomeCaixa.
296         :${idVeiculo} :temTracao ?tracao.
297         ?tracao :nome ?nomeTracao.
298         :${idVeiculo} :temPneu ?pneu.
```

```
299         ?pneu :jante ?jante.
300         ?pneu :largura ?largura.
301         ?pneu :ratio ?ratio.
302     }`
303     var res = await execQuery(query);
304     return res;
305 };
306 
307 
308 Carro.infoVeiculoBody = async (idVeiculo) => {
309     const query = `PREFIX : <http://prc.di.uminho.pt/2019/automoveis#>
310                         select ?vs ?nomeMarca ?nomeModelo where {
311                             :${idVeiculo} :temMotor ?motor.
312                             ?motor :eMotorDe ?vs.
313                             ?vs :temMarca ?marca.
314                             ?vs :temModelo ?modelo.
315                             ?marca :nome ?nomeMarca.
316                             ?modelo :nome ?nomeModelo.
317                         }`
318     var res = await execQuery(query);
319     return res;
320 };
321 
322 
323 //------------------------------ Pagina P2 -> Info Motor
    ------------------------------
324 
325 Carro.infoMotorHead = async (idMotor) => {
326     const query = `PREFIX : <http://prc.di.uminho.pt/2019/automoveis#>
327     select * where {
328         :${idMotor} :nome ?nomeMotor.
329         :${idMotor} :codigo ?codigo.
330         :${idMotor} :cilindrada ?cilindrada.
331         :${idMotor} :combustivel ?combustivel.
332         :${idMotor} :co2 ?co2.
333         :${idMotor} :compressao ?compressao.
334         :${idMotor} :consumo ?consumo.
335         :${idMotor} :materialBloco ?mb.
336         :${idMotor} :materialCilindros ?mc.
337         :${idMotor} :power ?power.
338         :${idMotor} :power_rpm ?prpm.
339         :${idMotor} :torque ?torque.
340         :${idMotor} :torque_rpm ?trpm.
```

```
341     }`
342     var res = await execQuery(query);
343     return res;
344 };
345 
346 
347 Carro.infoMotorBodyMarca = async (idMotor) => {
348     const query = `PREFIX : <http://prc.di.uminho.pt/2019/automoveis#>
349                     select distinct ?marca ?nomeMarca where {
350                         :${idMotor} :eMotorDe ?v.
351                         ?v :temMarca ?marca.
352                         ?marca :nome ?nomeMarca.
353                     }`
354     var res = await execQuery(query);
355     return res;
356 };
357 
358 Carro.infoMotorBodyVeiculos = async (idMotor) => {
359     const query = `PREFIX : <http://prc.di.uminho.pt/2019/automoveis#>
360                     select ?v ?nomeMarca ?nomeModelo where {
361                         :${idMotor} :eMotorDe ?v.
362                         ?v :temMarca ?marca.
363                         ?v :temModelo ?modelo.
364                         ?marca :nome ?nomeMarca.
365                         ?modelo :nome ?nomeModelo.
366                     }`
367     var res = await execQuery(query);
368     return res;
369 };
370 
371 
372 //---------------------------- Pagina P3 -> Info Pneu
    ----------------------------
373 
374 Carro.infoPneuHead = async (idPneu) => {
375     const query = `PREFIX : <http://prc.di.uminho.pt/2019/automoveis#>
376                     select * where {
377                         :${idPneu} :jante ?jante.
378                         :${idPneu} :largura ?largura.
379                         :${idPneu} :ratio ?ratio.
380                         :${idPneu} :descricao ?desc.
381                     }`
382     var res = await execQuery(query);
```

```
383     return res;
384 };
385 
386 Carro.infoPneuBodyMarca = async (idPneu) => {
387     const query = `PREFIX : <http://prc.di.uminho.pt/2019/automoveis#>
388                    select distinct ?m ?nome where {
389                        :${idPneu} :ePneuDe ?v.
390                        ?v :temMarca ?m.
391                        ?m :nome ?nome.
392                    }`
393     var res = await execQuery(query);
394     return res;
395 };
396 
397 Carro.infoPneuBodyVeiculos = async (idPneu) => {
398     const query = `PREFIX : <http://prc.di.uminho.pt/2019/automoveis#>
399                    select ?v ?nomeMarca ?nomeModelo ?nomeMotor where {
400                        :${idPneu} :ePneuDe ?v.
401                        ?v :temMarca ?m.
402                        ?m :nome ?nomeMarca.
403                        ?v :temModelo ?modelo.
404                        ?modelo :nome ?nomeModelo.
405                        ?v :temMotor ?motor.
406                        ?motor :nome ?nomeMotor.
407                    }`
408     var res = await execQuery(query);
409     return res;
410 };
411 
412 
413 //------------------------------ Pagina P4 -> Info Caixa de Velocidades
        ------------------------------
414 
415 Carro.infoCVHead = async (idCV) => {
416     const query = `PREFIX : <http://prc.di.uminho.pt/2019/automoveis#>
417                    select ?tipo ?desc where {
418                        :${idCV} :tipo ?tipo.
419                        :${idCV} :descricao ?desc.
420                    }`
421     var res = await execQuery(query);
422     return res;
423 };
424 
```

```
425 Carro.infoCVBodyMarca = async (idCV) => {
426     const query = `PREFIX : <http://prc.di.uminho.pt/2019/automoveis#>
427                     select distinct ?marca ?nomeMarca where {
428                         :${idCV} :eCaixaDe ?vs.
429                         ?vs :temMarca ?marca.
430                         ?marca :nome ?nomeMarca.
431                     }`
432     var res = await execQuery(query);
433     return res;
434 };
435
436 Carro.infoCVBodyVeiculos = async (idCV) => {
437     const query = `PREFIX : <http://prc.di.uminho.pt/2019/automoveis#>
438                     select ?vs ?nomeMarca ?nomeModelo ?nomeMotor where {
439                         :${idCV} :eCaixaDe ?vs.
440                         ?vs :temMarca ?marca.
441                         ?marca :nome ?nomeMarca.
442                         ?vs :temModelo ?modelo.
443                         ?modelo :nome ?nomeModelo.
444                         ?vs :temMotor ?motor.
445                         ?motor :nome ?nomeMotor.
446                     }`
447     var res = await execQuery(query);
448     return res;
449 };
450
451
452
453
454 //---------------------------- Pagina P5 -> Info Tracao
    ----------------------------
455
456 Carro.infoTracaoHead = async (idTr) => {
457     const query = `PREFIX : <http://prc.di.uminho.pt/2019/automoveis#>
458                     select ?nome ?tipo ?desc where {
459                         :${idTr} :nome ?nome.
460                         :${idTr} :tipo ?tipo.
461                         :${idTr} :descricao ?desc.
462                     }`
463     var res = await execQuery(query);
464     return res;
465 };
466
```

```
467 Carro.infoTracaoBodyMarca = async (idTr) => {
468     const query = `PREFIX : <http://prc.di.uminho.pt/2019/automoveis#>
469                    select distinct ?marca ?nomeMarca where {
470                        :${idTr} :eTracaoDe ?v.
471                        ?v :temMarca ?marca.
472                        ?marca :nome ?nomeMarca.
473                    }`
474     var res = await execQuery(query);
475     return res;
476 };
477
478 Carro.infoTracaoBodyVeiculos = async (idTr) => {
479     const query = `PREFIX : <http://prc.di.uminho.pt/2019/automoveis#>
480                    select ?v ?nomeMarca ?nomeModelo ?nomeMotor where {
481                        :${idTr} :eTracaoDe ?v.
482                        ?v :temMarca ?marca.
483                        ?marca :nome ?nomeMarca.
484                        ?v :temModelo ?modelo.
485                        ?modelo :nome ?nomeModelo.
486                        ?v :temMotor ?motor.
487                        ?motor :nome ?nomeMotor.
488                    }`
489     var res = await execQuery(query);
490     return res;
491 };
492
493
494 //------------------------------ Pagina P6 -> Info Marca
    ------------------------------
495
496 Carro.infoMarcaHead = async (idMarca) => {
497     const query = `PREFIX : <http://prc.di.uminho.pt/2019/automoveis#>
498                    select * where {
499                        :${idMarca} :nome ?nome.
500                        :${idMarca} :descricao ?desc.
501                    }`
502     var res = await execQuery(query);
503     return res;
504 };
505
506 Carro.infoMarcaBodyModelos = async (idMarca) => {
507     const query = `PREFIX : <http://prc.di.uminho.pt/2019/automoveis#>
508                    select ?v ?modelo ?nomeModelo ?nomeMotor where {
```

```
509                         ?v :temMarca :${idMarca}.
510                         ?v :temModelo ?modelo.
511                         ?modelo :nome ?nomeModelo.
512                         ?v :temMotor ?mot.
513                         ?mot :nome ?nomeMotor.
514     }`
515
516     var res = await execQuery(query);
517     return res;
518 };
519
520
521 Carro.infoMarcaBodyMotores = async (idMarca) => {
522     const query = `PREFIX : <http://prc.di.uminho.pt/2019/automoveis#>
523                     select distinct ?mot ?nome ?cod where {
524                         ?v a :Veiculo.
525                         ?v :temMarca :${idMarca}.
526                         ?v :temMotor ?mot.
527                         ?mot :nome ?nome.
528                         ?mot :codigo ?cod
529                     }`
530
531     var res = await execQuery(query);
532     return res;
533 };
534
535 Carro.infoMarcaMaisPotente = async (idMarca) => {
536     const query = `PREFIX : <http://prc.di.uminho.pt/2019/automoveis#>
537                     select ?v ?nomeModelo ?nomeMotor ?potencia where {
538                         ?v :temMarca :${idMarca}.
539                         ?v :temModelo ?modelo.
540                         ?modelo :nome ?nomeModelo.
541                         ?v :temMotor ?motor.
542                         ?motor :nome ?nomeMotor.
543                         ?motor :power ?potencia
544                     }ORDER BY DESC(?potencia)
545                     LIMIT 1`
546     var res = await execQuery(query);
547     return res;
548 };
549
550 Carro.infoMarcaMaisTorque = async (idMarca) => {
551     const query = `PREFIX : <http://prc.di.uminho.pt/2019/automoveis#>
```

```
552                     select ?v ?nomeModelo ?nomeMotor ?torque where {
553                         ?v :temMarca :${idMarca}.
554                         ?v :temModelo ?modelo.
555                         ?modelo :nome ?nomeModelo.
556                         ?v :temMotor ?motor.
557                         ?motor :nome ?nomeMotor.
558                         ?motor :torque ?torque.
559                     }ORDER BY DESC(?torque)
560                     LIMIT 1`
561     var res = await execQuery(query);
562     return res;
563 };
564
565 Carro.infoMarcaMenosConsumo = async (idMarca) => {
566     const query = `PREFIX : <http://prc.di.uminho.pt/2019/automoveis#>
567                     select ?v ?nomeModelo ?nomeMotor ?consumo where {
568                         ?v :temMarca :${idMarca}.
569                         ?v :temModelo ?modelo.
570                         ?modelo :nome ?nomeModelo.
571                         ?v :temMotor ?motor.
572                         ?motor :nome ?nomeMotor.
573                         ?motor :consumo ?consumo.
574                         FILTER(?consumo > 0).
575                     }ORDER BY(?consumo)
576                     LIMIT 1`
577     var res = await execQuery(query);
578     return res;
579 };
580
581 Carro.infoMarcaMenosEmissoes = async (idMarca) => {
582     const query = `PREFIX : <http://prc.di.uminho.pt/2019/automoveis#>
583                     select ?v ?nomeModelo ?nomeMotor ?emissoes where {
584                         ?v :temMarca :${idMarca}.
585                         ?v :temModelo ?modelo.
586                         ?modelo :nome ?nomeModelo.
587                         ?v :temMotor ?motor.
588                         ?motor :nome ?nomeMotor.
589                         ?motor :co2 ?emissoes.
590                         FILTER(?emissoes > 0).
591                     }ORDER BY(?emissoes)
592                     LIMIT 1`
593     var res = await execQuery(query);
594     return res;
```

```
595 };
```

A.2.2 Roteador

```
1  var express = require('express');
2  var router = express.Router();
3
4  var Carro = require('../controllers/automoveis')
5
6
7  //------------------------------ HOME ------------------------------
8
9  router.get('/home/contaMarcas', async function(req, res, next) {
10     var dados = await Carro.contaMarcas()
11     res.jsonp(dados)
12 });
13
14 router.get('/home/contaModelos', async function(req, res, next) {
15     var dados = await Carro.contaModelos()
16     res.jsonp(dados)
17 });
18
19 router.get('/home/contaVeiculos', async function(req, res, next) {
20     var dados = await Carro.contaVeiculos()
21     res.jsonp(dados)
22 });
23
24 router.get('/home/contaMotores', async function(req, res, next) {
25     var dados = await Carro.contaMotores()
26     res.jsonp(dados)
27 });
28
29 router.get('/home/contaPneus', async function(req, res, next) {
30     var dados = await Carro.contaPneus()
31     res.jsonp(dados)
32 });
33
34 router.get('/home/contaTracoes', async function(req, res, next) {
35     var dados = await Carro.contaTracoes()
36     res.jsonp(dados)
37 });
38
39 router.get('/home/contaCvs', async function(req, res, next) {
40     var dados = await Carro.contaCvs()
```

```
41      res.jsonp(dados)
42  });
43
44  //------------------------------ Pagina MARCAS ------------------------------
45
46  router.get('/marcas/todasMarcas', async function(req, res, next) {
47      var dados = await Carro.todasMarcas()
48      res.jsonp(dados)
49  });
50
51
52  //------------------------------ Pagina Veiculos ------------------------------
53
54  router.get('/veiculos/todosVeiculos', async function(req, res, next) {
55      var dados = await Carro.todosVeiculos()
56      res.jsonp(dados)
57  });
58
59  //------------------------------ Pagina Motores ------------------------------
60
61  router.get('/motores/todosMotores', async function(req, res, next) {
62      var dados = await Carro.todosMotores()
63      res.jsonp(dados)
64  });
65
66
67  //------------------------------ Todas Caixas ------------------------------
68
69  router.get('/caixas/todasCaixas', async function(req, res, next) {
70      var dados = await Carro.todasCaixas()
71      res.jsonp(dados)
72  });
73
74
75  //------------------------------ Pagina Tracoes ------------------------------
76
77  router.get('/tracoes/todasTracoes', async function(req, res, next) {
78      var dados = await Carro.todasTracoes()
79      res.jsonp(dados)
80  });
81
82
83  //------------------------------ Pagina Pneus ------------------------------
```

```
84
85
86  router.get('/pneus/todasJantes', async function(req, res, next) {
87      var dados = await Carro.todasJantes()
88      res.jsonp(dados)
89  });
90
91  router.get('/pneus/todasLarguras', async function(req, res, next) {
92      var dados = await Carro.todasLarguras()
93      res.jsonp(dados)
94  });
95
96  router.get('/pneus/todosRatios', async function(req, res, next) {
97      var dados = await Carro.todosRatios()
98      res.jsonp(dados)
99  });
100
101
102 //------------------------------ Pagina Pesquisa ------------------------------
103
104
105 router.post('/pesquisar', async function(req, res, next) {
106     var dados = await Carro.pesquisar(req.body)
107     res.jsonp(dados)
108 });
109
110
111
112 //------------------------------ Pagina P1 -> Info Veiculo
        ------------------------------
113
114 router.get('/p1/infoVeiculoHead/:id', async function(req, res, next) {
115     var dados = await Carro.infoVeiculoHead(req.params.id)
116     res.jsonp(dados)
117 });
118
119 router.get('/p1/infoVeiculoBody/:id', async function(req, res, next) {
120     var dados = await Carro.infoVeiculoBody(req.params.id)
121     res.jsonp(dados)
122 });
123
124 //------------------------------ Pagina P2 -> Info Motor
        ------------------------------
```

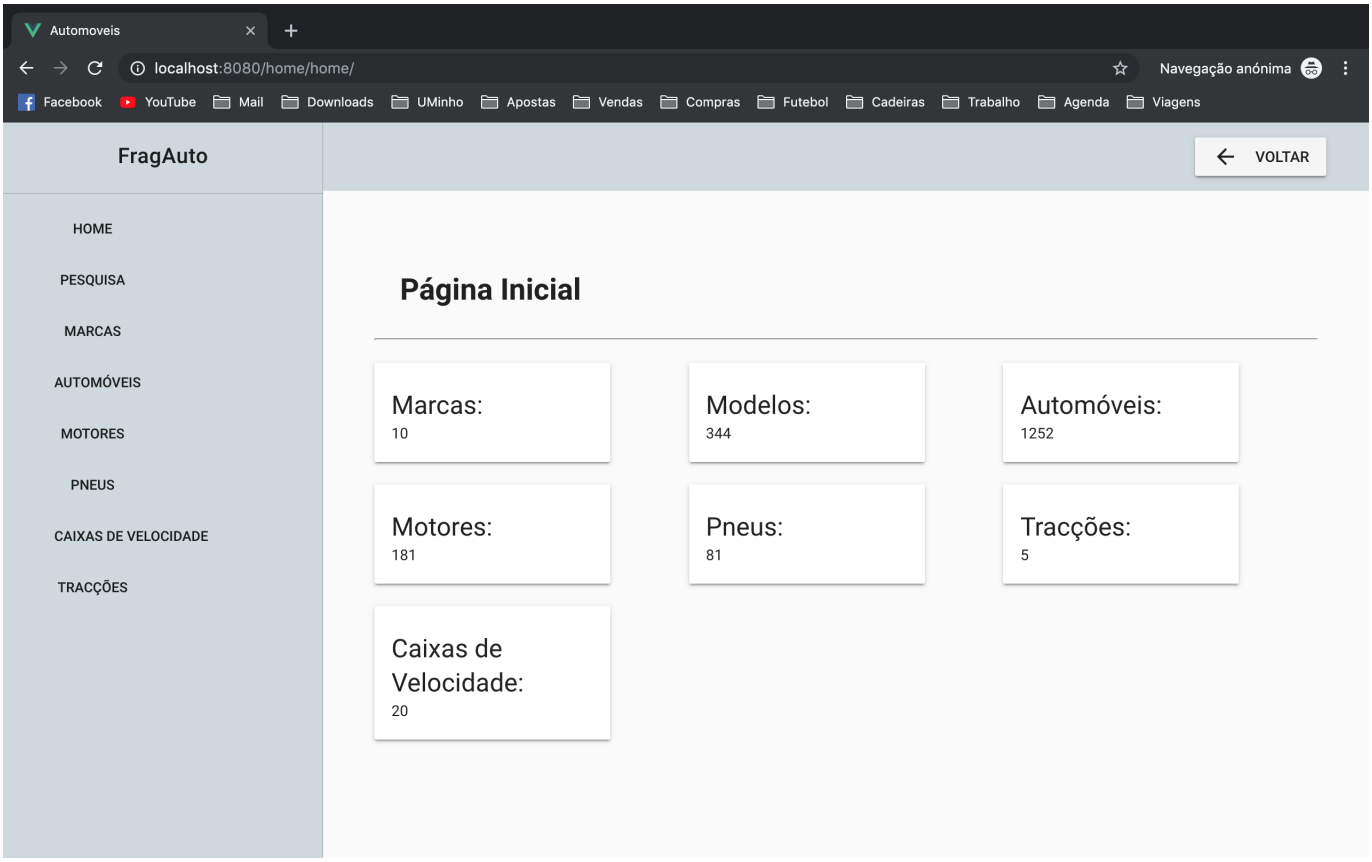
```
125
126
127 router.get('/p2/infoMotorHead/:id', async function(req, res, next) {
128     var dados = await Carro.infoMotorHead(req.params.id)
129     res.jsonp(dados)
130 });
131
132 router.get('/p2/infoMotorBodyMarca/:id', async function(req, res, next) {
133     var dados = await Carro.infoMotorBodyMarca(req.params.id)
134     res.jsonp(dados)
135 });
136
137 router.get('/p2/infoMotorBodyVeiculos/:id', async function(req, res, next) {
138     var dados = await Carro.infoMotorBodyVeiculos(req.params.id)
139     res.jsonp(dados)
140 });
141
142 //------------------------------ Pagina P3 -> Info Pneu
        ------------------------------
143
144
145 router.get('/p3/infoPneuHead/:id', async function(req, res, next) {
146     var dados = await Carro.infoPneuHead(req.params.id)
147     res.jsonp(dados)
148 });
149
150 router.get('/p3/infoPneuBodyMarca/:id', async function(req, res, next) {
151     var dados = await Carro.infoPneuBodyMarca(req.params.id)
152     res.jsonp(dados)
153 });
154
155 router.get('/p3/infoPneuBodyVeiculos/:id', async function(req, res, next) {
156     var dados = await Carro.infoPneuBodyVeiculos(req.params.id)
157     res.jsonp(dados)
158 });
159
160 //------------------------------ Pagina P4 -> Info CV
        ------------------------------
161
162
163 router.get('/p4/infoCVHead/:id', async function(req, res, next) {
164     var dados = await Carro.infoCVHead(req.params.id)
165     res.jsonp(dados)
```

```
166 });
167
168 router.get('/p4/infoCVBodyMarca/:id', async function(req, res, next) {
169     var dados = await Carro.infoCVBodyMarca(req.params.id)
170     res.jsonp(dados)
171 });
172
173 router.get('/p4/infoCVBodyVeiculos/:id', async function(req, res, next) {
174     var dados = await Carro.infoCVBodyVeiculos(req.params.id)
175     res.jsonp(dados)
176 });
177
178 //------------------------------ Pagina P5 -> Info Tracao
    ------------------------------
179
180
181 router.get('/p5/infoTracaoHead/:id', async function(req, res, next) {
182     var dados = await Carro.infoTracaoHead(req.params.id)
183     res.jsonp(dados)
184 });
185
186 router.get('/p5/infoTracaoBodyMarca/:id', async function(req, res, next) {
187     var dados = await Carro.infoTracaoBodyMarca(req.params.id)
188     res.jsonp(dados)
189 });
190
191 router.get('/p5/infoTracaoBodyVeiculos/:id', async function(req, res, next)
    {
192     var dados = await Carro.infoTracaoBodyVeiculos(req.params.id)
193     res.jsonp(dados)
194 });
195
196 //------------------------------ Pagina P6 -> Info Marca
    ------------------------------
197
198
199 router.get('/p6/infoMarcaHead/:id', async function(req, res, next) {
200     var dados = await Carro.infoMarcaHead(req.params.id)
201     res.jsonp(dados)
202 });
203
204 router.get('/p6/infoMarcaBodyModelos/:id', async function(req, res, next) {
205     var dados = await Carro.infoMarcaBodyModelos(req.params.id)
```

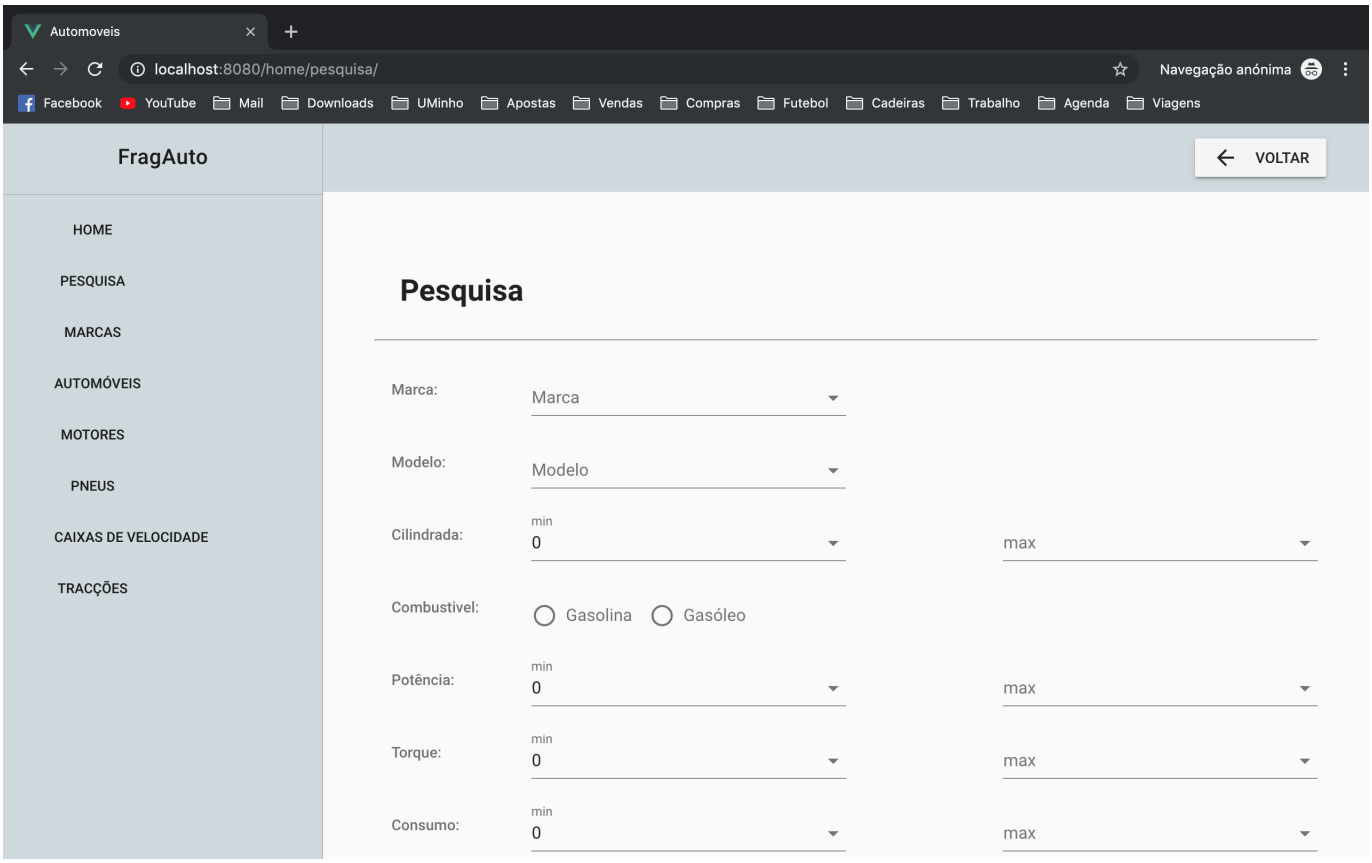
```
206     res.jsonp(dados)
207 });
208
209 router.get('/p6/infoMarcaBodyMotores/:id', async function(req, res, next) {
210     var dados = await Carro.infoMarcaBodyMotores(req.params.id)
211     res.jsonp(dados)
212 });
213
214 router.get('/p6/infoMarcaMaisPotente/:id', async function(req, res, next) {
215     var dados = await Carro.infoMarcaMaisPotente(req.params.id)
216     res.jsonp(dados)
217 });
218
219 router.get('/p6/infoMarcaMaisTorque/:id', async function(req, res, next) {
220     var dados = await Carro.infoMarcaMaisTorque(req.params.id)
221     res.jsonp(dados)
222 });
223
224 router.get('/p6/infoMarcaMenosConsumo/:id', async function(req, res, next) {
225     var dados = await Carro.infoMarcaMenosConsumo(req.params.id)
226     res.jsonp(dados)
227 });
228
229 router.get('/p6/infoMarcaMenosEmissoes/:id', async function(req, res, next)
        {
230     var dados = await Carro.infoMarcaMenosEmissoes(req.params.id)
231     res.jsonp(dados)
232 });
233
234
235 module.exports = router;
```

A.3 Interface

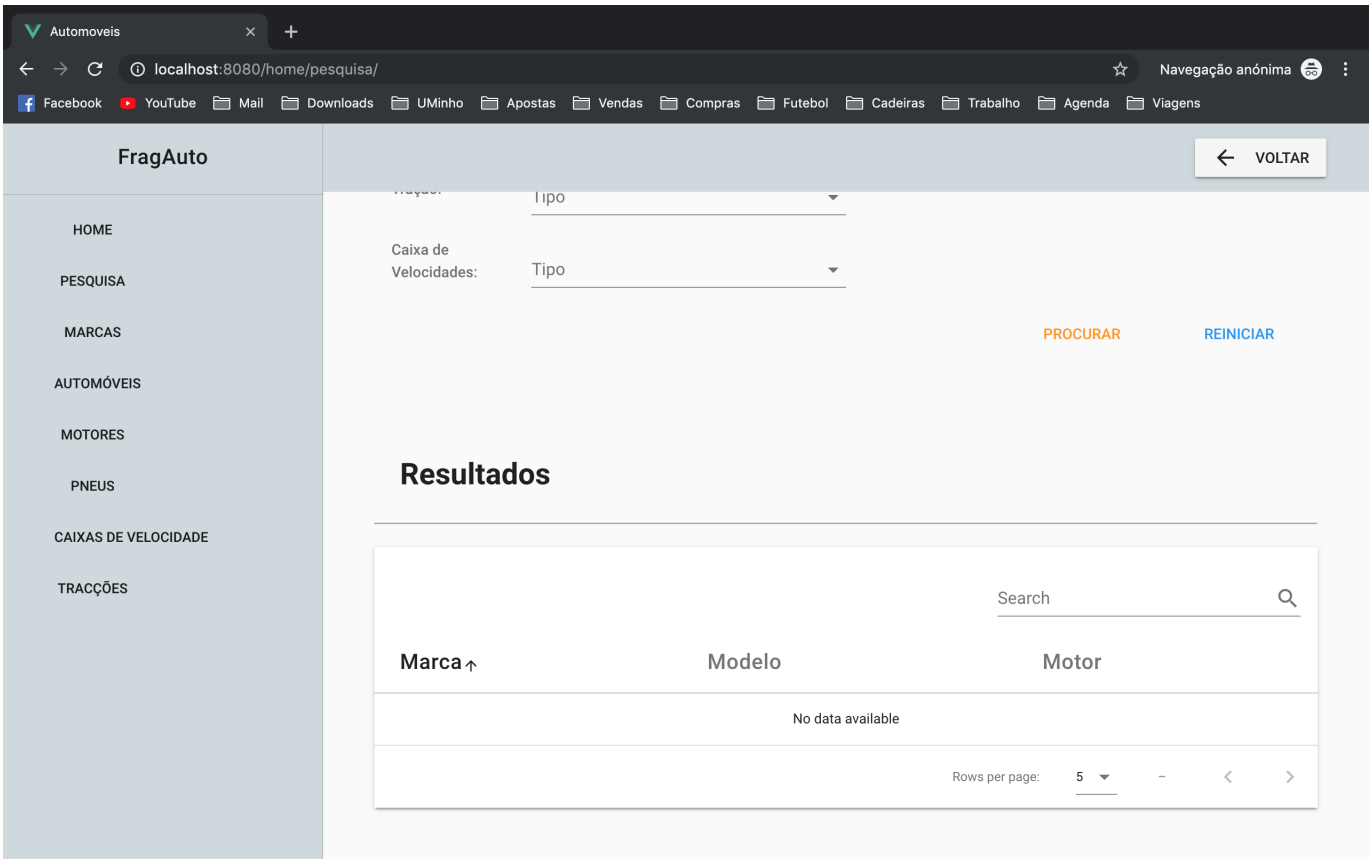
A.3.1 Pagina inicial



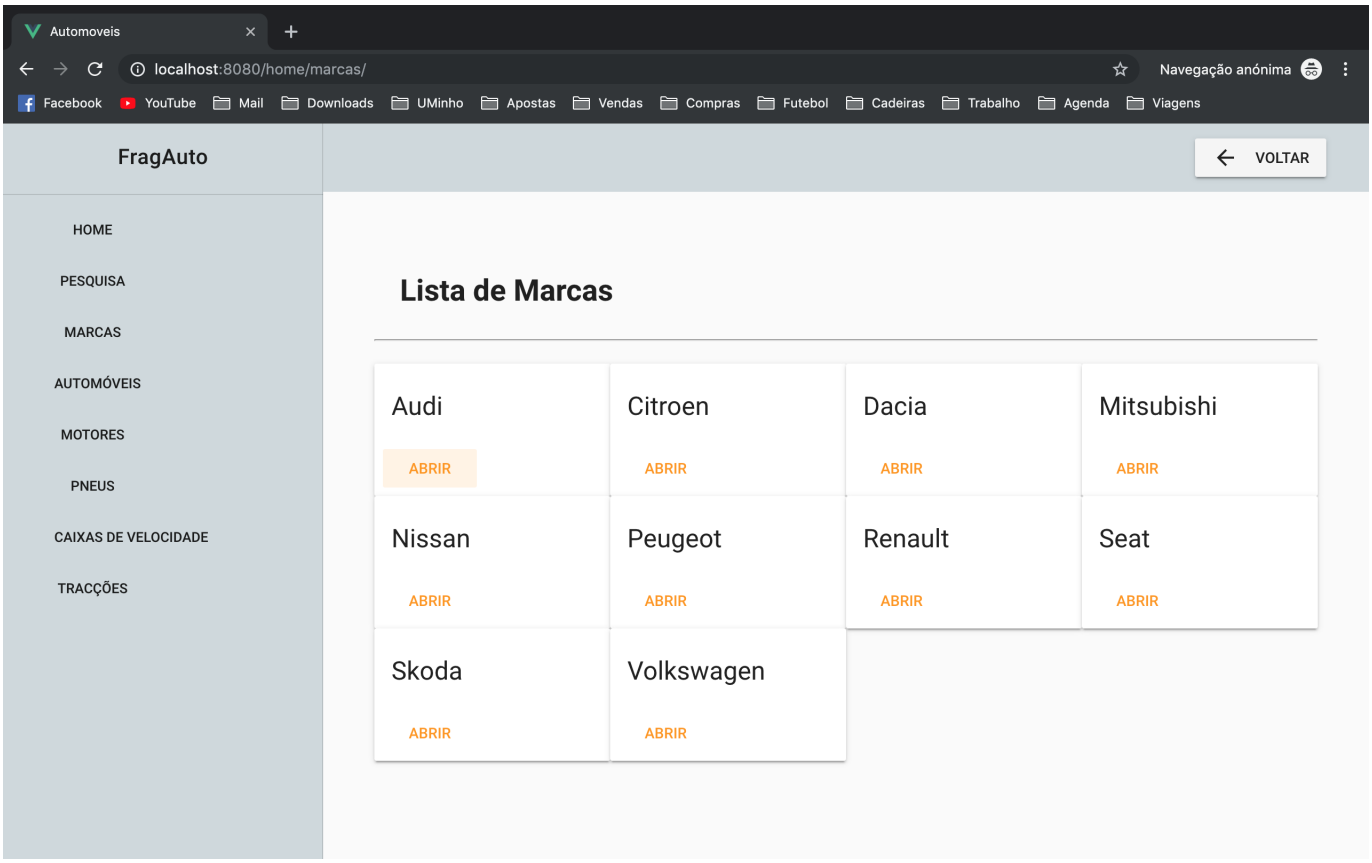
A.3.2 Pagina de pesquisa



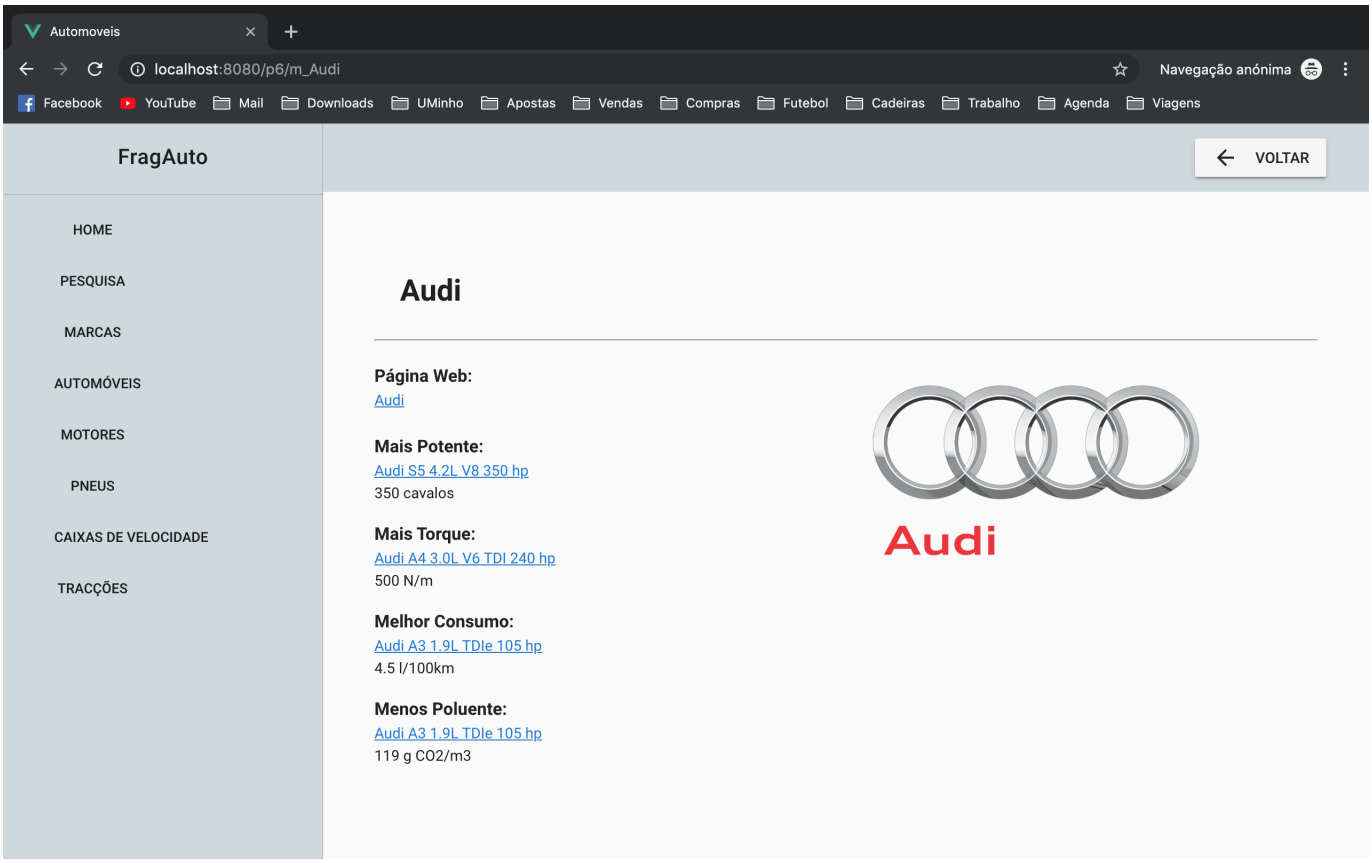
A.3.3 Pagina de pesquisa com apresentação de resultados.



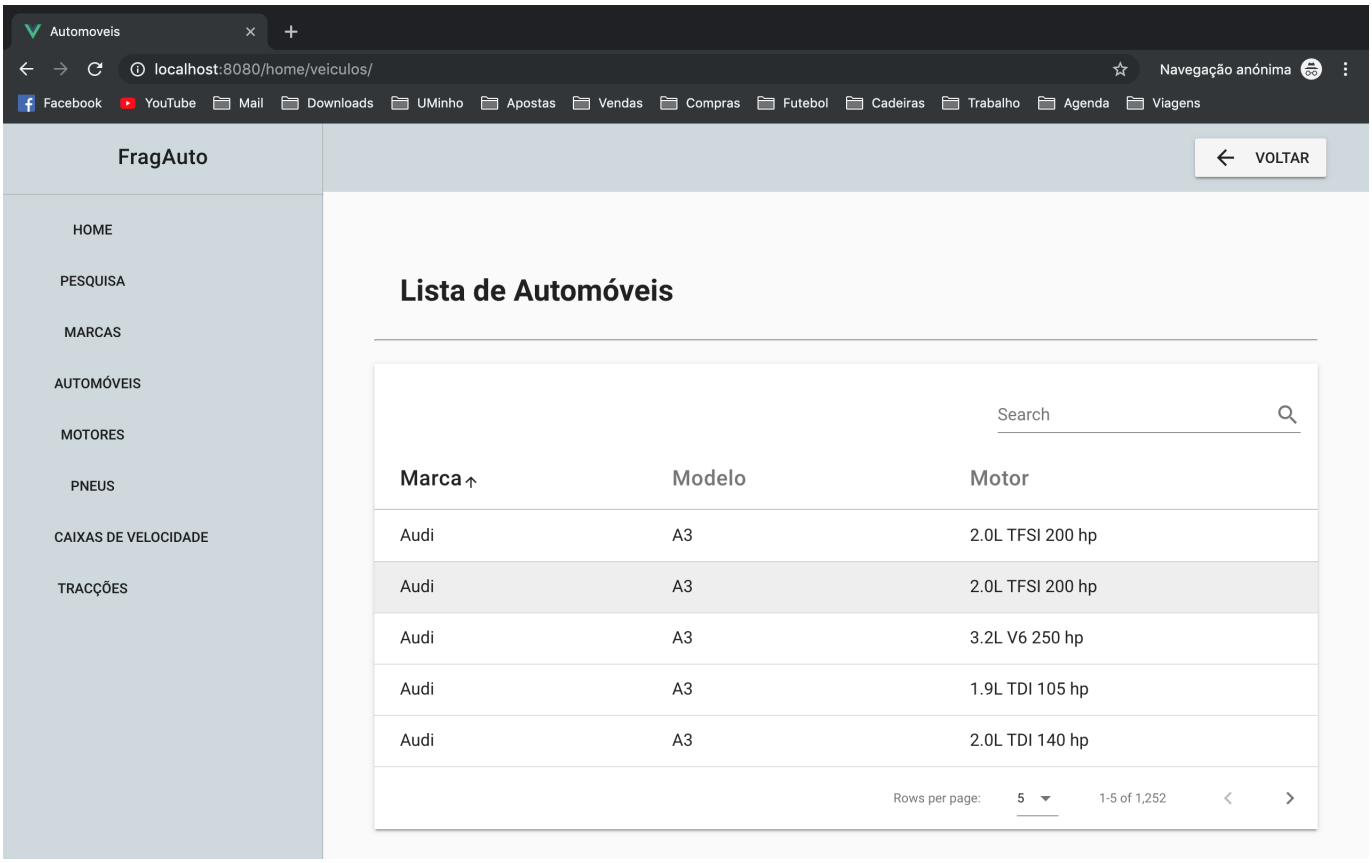
A.3.4 Pagina de listagem de marcas.



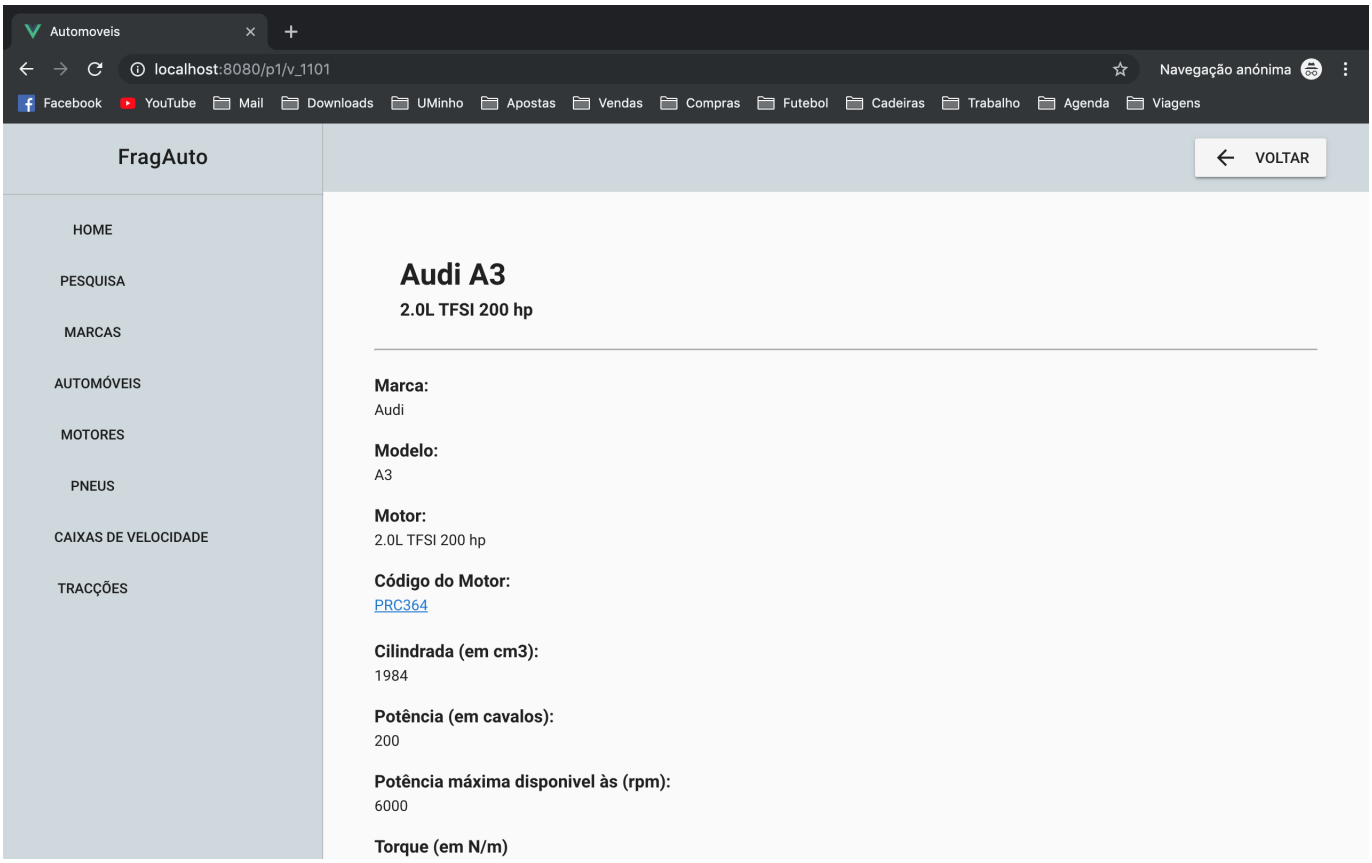
A.3.5 Pagina de informação de uma marca.



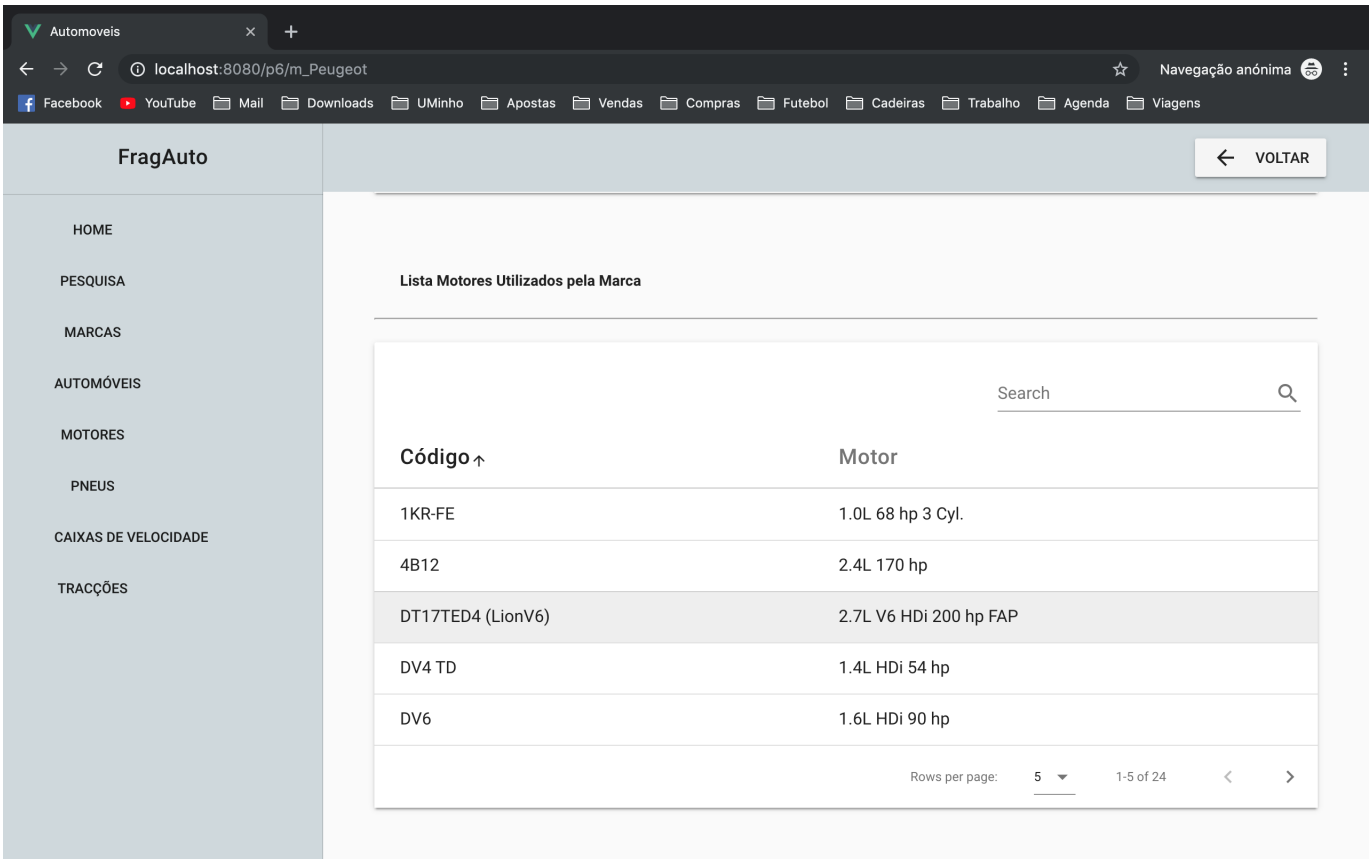
A.3.6 Pagina de automóveis.



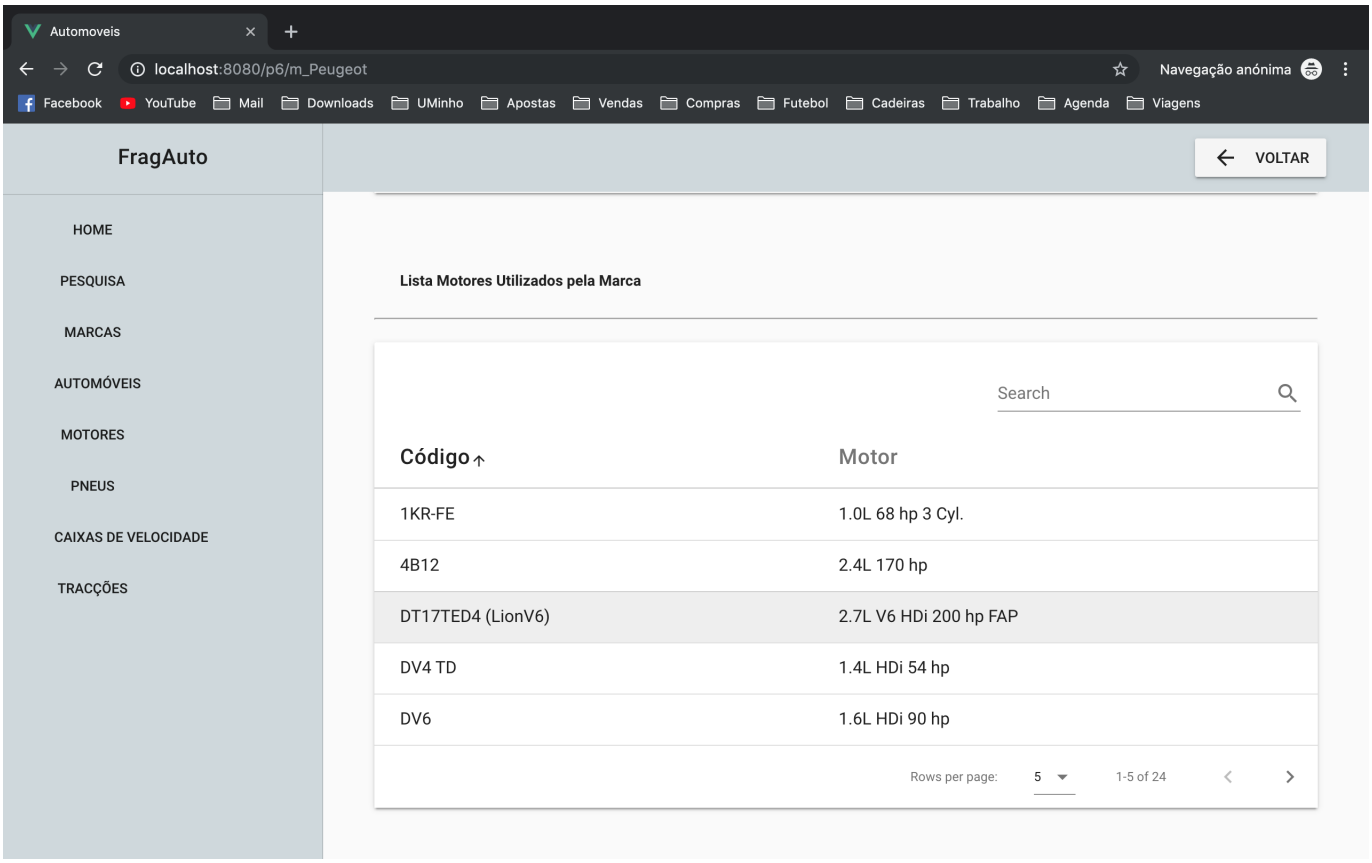
A.3.7 Pagina de informação de um automóvel.



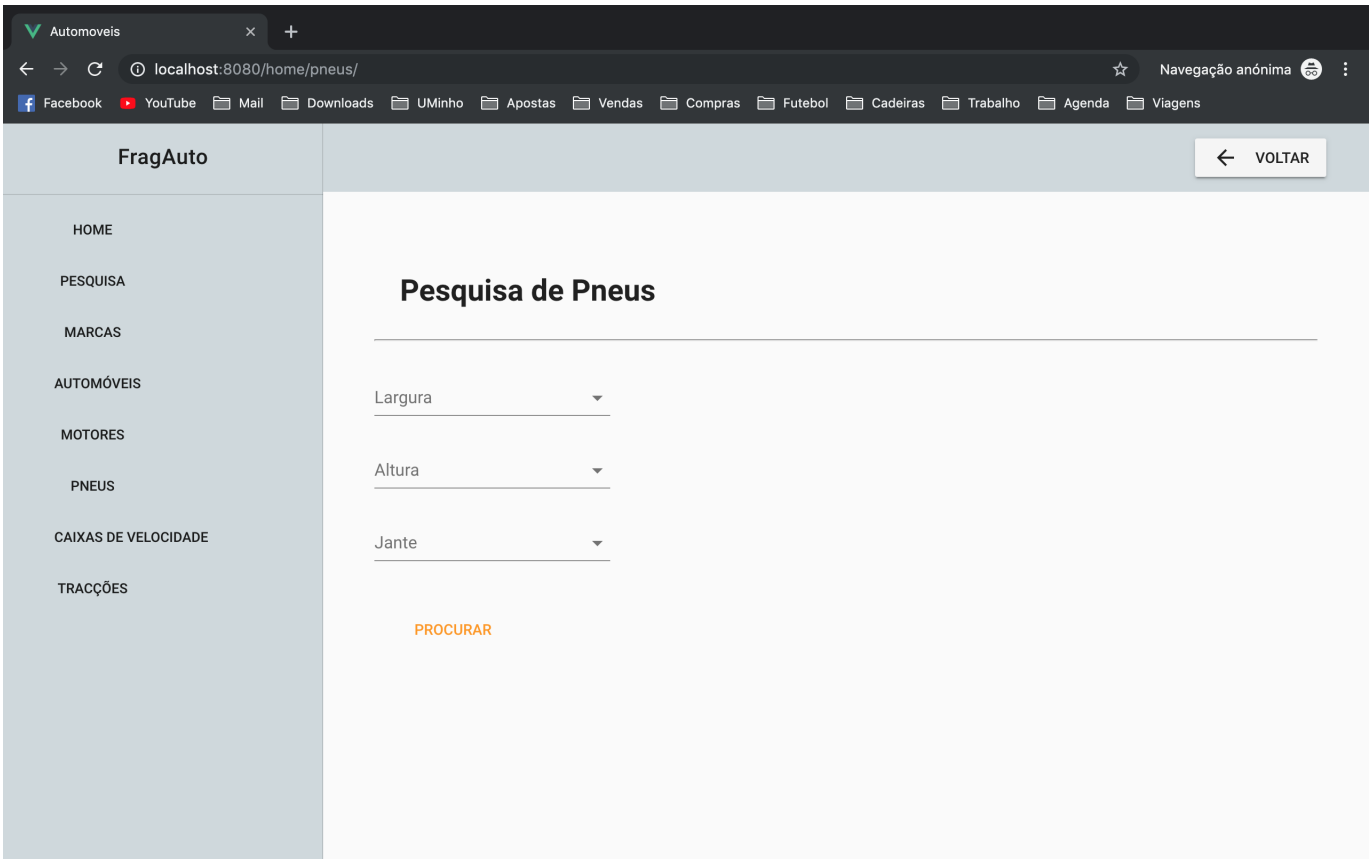
A.3.8 Pagina de motores



A.3.9 Pagina de uma marca - Listagem dos motores



A.3.10 Pagina de pesquisa de pneus



A.3.11 Pagina de informação de um pneu