Projeto 2

Universidade de Aveiro (UA)

Vasco Sousa, Miguel Cabral,
Tiago Rainho, Francisco Monteiro

Projeto 2

Departamento de Eletrónica, Telecomunicações e Informática

UA

Vasco Sousa, Miguel Cabral,
Tiago Rainho, Francisco Monteiro
(93049) jvcs@ua.pt, (93091) miguel.f.cabral@ua.pt,
(92984) tiago.rainho@ua.pt, (93105) francisco.monteiro@ua.pt

15 de Junho de 2019

Resumo

O nosso projeto consiste numa aplicação web que recebe imagens, envia-as para um endereço que as analisa e devolve possíveis objetos detetados. É suportada por uma interface web que é muito simples de se trabalhar. Depois de enviada a imagem são recortadas as imagens dos objetos e armazenadas para futura pesquisa. A pesquisa pode ser feita através do nome do objeto detetado e também através da cor predominante. Podemos também alterar o nível de confiança de cada objeto entre 0 e 100, em que 0 mostrará todos os objetos detetados de acordo com a pesquisa e 100 só mostrará aqueles que o sistema detetou com 100% de certeza do que eram. O código é feito maioritariamente em python, mas também utilizámos o Sqlite3 para criar a base de dados. Já a interface web utiliza Hyper Text Markup Language (HTML), Cascading Style Sheets (CSS), JavaScript (JS).

Agradecimentos

Queremos agradecer a todos os docentes que lecionam a Unidade Curricular (UC) de Laboratórios de Informática (LABI) que no conjunto destes dois semestres nos deram os conhecimentos para a realização deste projeto.

Foi usado o tema **argon** *(clicar para ir para o site)* nas paginas html e tambem **Bootstrap** *(clicar para ir para o site)* .

Conteúdo

Lista de Figuras

Capítulo 1

Introdução

Este relatório foi elaborado com o propósito de descrever e analisar o segundo projeto realizado no âmbito da UC de LABI.

Este documento está dividido em oito capítulos. Depois desta introdução no Capítulo 2 é apresentada a interface web do sistema, no Capítulo 3 é explicitado como funciona a aplicação web , no Capítulo 4 é abordada a base de dados. No Capítulo 5 é abordado o funcionamento do processador de imagens. No Capítulo 6 são explicitadas as novas funcionalidades desenvolvidas e no Capítulo 7 são explicitados os resultados. Finalmente, no Capítulo 8 são apresentadas as conclusões do trabalho.

Capítulo 2

Interface web

Esta componente é composta por 5 páginas desenvolvidas através de HTML, JS e CSS fornecendo a interface para a interação com o sistema. A aplicação web é simples e fácil de entender para qualquer utilizador, pelo que foi desenhada para ser fácil de trabalhar, funcional e organizada.

2.1 HTML, CSS e Javascript

Neste projeto, utilizámos estes recursos como é normal na criação de websites. O achtml é a principal base da aplicação web, sendo que é onde se implementam as várias divisões da aplicação. No accss é onde se implementam os estilos das letras e das imagens. Isto ajuda a simplificar e a tornar apelativa a aplicação. O acjs ajuda a implementar certas funcionalidades e dinâmicas na interface web. Este ajuda a tornar a página mais interativa para o utilizador. Estes são os principais aspetos da interface web. Utilizámos também o Bootstrap para nos ajudar a tornar a página mais simples e interativa para o utilizador.

2.2 Funcionalidades de cada página web

A interface web está dividida em cinco páginas distintas. A primeira página da interface web é composta pela listagem de todos os objetos detetados nas imagens enviadas e a quantidade de vezes que já foi detetado.

Na segunda página, é implementada a funcionalidade de procura de imagens pelo nome. As imagens que aparecem são as que foram detetadas e recortadas. Pode também ajustar o nível de confiança de deteção, entre 0 e 100, sendo que predefinidamente estará a 50.

Na terceira página, é onde pode enviar novas imagens para o sistema, para a sua deteção.

Na quarta, irá restringir a pesquisa, pois é possível pesquisar por nome e por cor. As únicas cores disponíveis são o vermelho, o verde e o azul, visto que estamos a utilizar o modo de cores RGB. Assim, as imagens que aparecerão

nesta página, serão as que tiverem predominantemente a cor escolhida para pesquisa.

Já a quinta página, é apenas abordada a informação sobre os autores do projeto.

2.3 Responsividade

A aplicação web tem várias caraterísticas. Uma delas é a responsividade, isto é, é possível visualizar a aplicação em qualquer dispositivo de qualquer tamanho sem alterar a integridade da mesma. Utilizámos também uma *navbar* para facilitar a mudança de funcionalidade pelo utilizador, clicando na aba que quer utilizar.

Capítulo 3

Aplicação Web

A aplicação web é composta por um programa em python que apresenta métodos que permitem o fluxo de informação entre as diversas componentes.

A primeira página é composta por uma lista de todos os objetos já detetados, de todas as imagens inseridas. A listagem é conseguida através da chamada de ***/list?type=names*** que devolve um objeto JSON com um array de todos os objetos detetados no sistema. Por exemplo:

[{"id": 27, "class": "truck", "confidence": 22}, {"id": 26, "class": "person", "confidence": 22}, {"id": 25, "class": "person", "confidence": 29}, {"id": 24, "class": "car", "confidence": 30}, {"id": 23, "class": "handbag", "confidence": 38}, {"id": 22, "class": "person", "confidence": 46}, {"id": 21, "class": "car", "confidence": 58}, {"id": 20, "class": "truck", "confidence": 58}, {"id": 19, "class": "handbag", "confidence": 68}, {"id": 18, "class": "person", "confidence": 68}, {"id": 17, "class": "person", "confidence": 71}, {"id": 16, "class": "car", "confidence": 79}, {"id": 15, "class": "traffic light", "confidence": 87}, {"id": 14, "class": "truck", "confidence": 92}, {"id": 13, "class": "truck", "confidence": 94}, {"id": 12, "class": "person", "confidence": 95}, {"id": 11, "class": "car", "confidence": 96}, {"id": 10, "class": "person", "confidence": 98}, {"id": 9, "class": "car", "confidence": 98}, {"id": 8, "class": "car", "confidence": 98}, {"id": 7, "class": "person", "confidence": 99}, {"id": 6, "class": "person", "confidence": 100}, {"id": 5, "class": "tie", "confidence": 26}, {"id": 4, "class": "person", "confidence": 100}, {"id": 3, "class": "traffic light", "confidence": 94}, {"id": 2, "class": "traffic light", "confidence": 99}, {"id": 1, "class": "traffic light", "confidence": 100}]

Figura 3.1: Objeto JSON

A função ***/list?type=detected*** devolve um objeto JSON com uma lista das pequenas imagens contendo os objetos extraídos e o nível de confiança e a imagem original. O exemplo abaixo mostra isso :

[{"id": 27, "class": "truck", "name": "croped_21_NYC_14th_Street_looking_west_12_2005.jpg", "image": "917265eb71a32cccf4b073e4b91ecf66", "confidence": 22, "original": "e0ed6c9e6d4fbc7f5680126c116ffae8"}, {"id": 26, "class": "person", "name": "croped_20_NYC_14th_Street_looking_west_12_2005.jpg", "image": "1e57543c58eb4aca934ebecbb4262f0c", "confidence": 22, "original": "e0ed6c9e6d4fbc7f5680126c116ffae8"}, {"id": 25, "class": "person", "name": "croped_19_NYC_14th_Street_looking_west_12_2005.jpg", "image": "c82dd7a747a58c457f1dd64858c4542e", "confidence": 29, "original": "e0ed6c9e6d4fbc7f5680126c116ffae8"}, {"id": 24, "class": "car", "name": "croped_18_NYC_14th_Street_looking_west_12_2005.jpg", "image": "1b7241cbd43627dd45cb523e17fcabf6", "confidence": 30, "original": "e0ed6c9e6d4fbc7f5680126c116ffae8"}, {"id": 23, "class": "handbag", "name":

Figura 3.2: Objeto JSON

A função ***/list?type=detected&name=NAME*** devolve um objeto JSON com uma lista de pequenas imagens contendo os objetos extraídos que sejam relativos ao objeto referenciado pelo filtro NAME. Por exemplo quando *NAME* corresponde a *person*:

[{"id": 40, "class": "person", "name": "croped_6_images.jpeg", "image": "e1dcecf16e8ebb3e2f1beaa1af4ea246", "confidence": 56, "original": "864ed1f95b3beb0534de627dd7f21160"}, {
"person", "name": "croped_3_images.jpeg", "image": "b792d7a6b9a8f30e5e3a0a87770dc3c6", "confidence": 97, "original": "864ed1f95b3beb0534de627dd7f21160"}, {"id": 36, "class":
"croped_2_images.jpeg", "image": "d56974d217ec7d8dd153788cc6f61f73", "confidence": 97, "original": "864ed1f95b3beb0534de627dd7f21160"}, {"id": 35, "class": "person", "name":

Figura 3.3: Objeto JSON

A função */**list?type=detected&name=NAME&color=COLOR*** devolve um objeto JSON tem a mesma função que a anterior mas com um filtro adicional (COLOR). Por exemplo quando *COLOR* corresponde a *green*:

[{"id": 26, "class": "person", "name": "croped_25_DTAH_MarketStreet_Toronto_LandscapeArchitecture_UrbanDesign.jpg", "image": "1436a2a95bb6e5dd546850e4f8b1d6b4", "confidence": 26, "original":
"2e61088931053569ef72ce43de392ee3"}, {"id": 8, "class": "person", "name": "croped_7_DTAH_MarketStreet_Toronto_LandscapeArchitecture_UrbanDesign.jpg", "image":
"9325d20e5590a185e91a0023a6afa5e1", "confidence": 92, "original": "2e61088931053569ef72ce43de392ee3"}]

Figura 3.4: Objeto JSON

A função */**put:*** permite enviar uma imagem para que seja processada e armazenada.

A função */**get?id=IDENTIFIER*** permite obter uma imagem (completa ou extraída) com um objeto através do id dado. Por exemplo:

[{"id": 11, "class": "traffic", "name": "croped_10_0Q8A5141V2.jpg", "image": "87d6758305cb933580061e14dafb9101", "confidence": 99, "original": "b8ffb592e595b84817cce74f0d967bb1"}]

Figura 3.5: Objeto JSON

Capítulo 4

Persistência

Para ajuda à regulação do armazenamento de imagens, criámos uma base de dados onde se encontra o caminho para as imagens. Desta forma, facilita a procura das imagens na pasta. Usámos o sqlite3 como linguagem para a base de dados. Nas imagens seguintes podemos visualizar o armazenamento do caminho para as imagens e a separação entre as imagens originais e as imagens recortadas. As imagens originais encontram-se na pasta original e as imagens recortadas encontram-se na pasta objects.

id	class	name	image	confidence	original	created_at
1	person	croped_0_images.jpeg	fcf4dec4ba6a3a5d229baefaf9abe6d7	100	7b7303e10dab6ea69d2ad87cae16fc0e	2019-06-06 17:23:35
2	person	croped_1_images.jpeg	9ece1021823c882f27247fdf847d49747	100	7b7303e10dab6ea69d2ad87cae16fc0e	2019-06-06 17:23:35
3	person	croped_2_images.jpeg	6280842fee40f2aab49efb9491e2a601	98	7b7303e10dab6ea69d2ad87cae16fc0e	2019-06-06 17:23:35
4	traffic light	croped_3_images.jpeg	5cd1f40d67d005189ca3b5a12e59408b	98	7b7303e10dab6ea69d2ad87cae16fc0e	2019-06-06 17:23:35
5	person	croped_4_images.jpeg	81ae5b79307b27594acb877465ebaded	96	7b7303e10dab6ea69d2ad87cae16fc0e	2019-06-06 17:23:35
6	person	croped_5_images.jpeg	701b90a5f35face7fddb64383b66a6d1	88	7b7303e10dab6ea69d2ad87cae16fc0e	2019-06-06 17:23:35
7	person	croped_6_images.jpeg	36ccc7013ede367222711a2267a5491e	83	7b7303e10dab6ea69d2ad87cae16fc0e	2019-06-06 17:23:35
8	person	croped_7_images.jpeg	c9b8717803707fdf64e87c5b08925f7b	72	7b7303e10dab6ea69d2ad87cae16fc0e	2019-06-06 17:23:35
9	person	croped_8_images.jpeg	a797f23400ebac70a5ba742bfa39ed69	65	7b7303e10dab6ea69d2ad87cae16fc0e	2019-06-06 17:23:35
10	suitcase	croped_9_images.jpeg	947f1650bb56958dfbe4c76d01b89aa5	31	7b7303e10dab6ea69d2ad87cae16fc0e	2019-06-06 17:23:35
11	traffic light	croped_10_images.jpeg	005a36b712b5e0a16e32120f797ab7c3	30	7b7303e10dab6ea69d2ad87cae16fc0e	2019-06-06 17:23:35
12	traffic light	croped_11_images.jpeg	abc29d6f04f3097545a369b25749cd8c	28	7b7303e10dab6ea69d2ad87cae16fc0e	2019-06-06 17:23:35
13	backpack	croped_12_images.jpeg	1ca2643802e5112d426a4a32df915254	24	7b7303e10dab6ea69d2ad87cae16fc0e	2019-06-06 17:23:35
14	person	croped_13_images.jpeg	85bd7859b2dbfe31db364be8e9847957	20	7b7303e10dab6ea69d2ad87cae16fc0e	2019-06-06 17:23:35
15	traffic light	croped_14_images.jpeg	afd20b3efdcab188391f3a909dcc3445	20	7b7303e10dab6ea69d2ad87cae16fc0e	2019-06-06 17:23:35

Figura 4.1: Tabela dos Objetos detetados

Na tabela anterior, podemos visualizar que há várias colunas, cada uma correspondente a uma caraterística das imagens. A coluna *class* é referente ao objeto em si, por exemplo, uma pessoa. A coluna *name* é o nome do ficheiro da imagem. A coluna *image* é a encriptação do nome do ficheiro da imagem recortada e a *original* é a encriptação do nome do ficheiro da imagem original. A coluna *confidence* é referente à confiança da imagem, em que quanto maior a confiança, maior a certeza de que a imagem é o objeto indicado, da cor indicada e a última coluna é a data em que a imagem foi enviada.

id	name	checksum	created_at
1	images.jpeg	/b/303e10dab6ea69d2ad87cae16fc0e	2019-06-06 17:23:29

Figura 4.2: Tabela das Imagens Enviadas

Nesta tabela, podemos visualizar as imagens enviadas, sendo que na primeira coluna temos o nome da imagem original e na coluna seguinte a sua encriptação. É aqui que ficam armazenadas as imagens originais enviadas pelo utilizador.

Capítulo 5

Processador de Imagens

Esta componente é o responsável pelo processamento das imagens. Fizémo-lo de modo a que a imagem seja enviada automaticamente para o endereço disponibilizado no enunciado do projeto, e que devolve um objeto acjson que é interpretado nesta componente. Utilizámos também o algoritmo fornecido pelos docentes da UC para enviar automaticamente as imagens para esse endereço.

Capítulo 6

Novas Funcionalidades

Como foi proposto pelos docentes decidimos implementar uma funcionalidade para além das pedidas. Essa funcionalidade consiste em apagar objetos da base de dados e também alterar a sua classe e o nível de confiança que o servidor nos manda. Foram criadas duas funções para o sucedido a função *delect_object* que apaga o objeto e a função *edit_object*. De seguida temos alguns exemplos da funcionalidade descrita em cima:

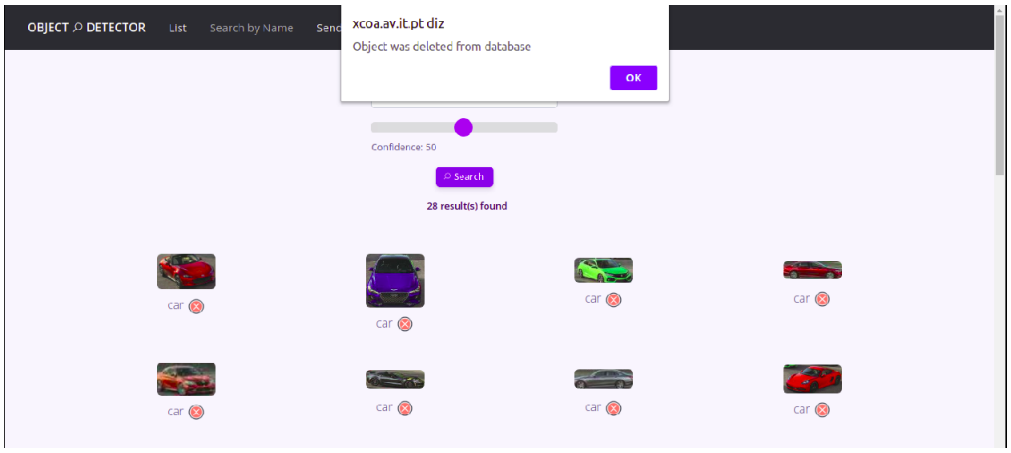

Figura 6.1: Apagar objetos da base de dados

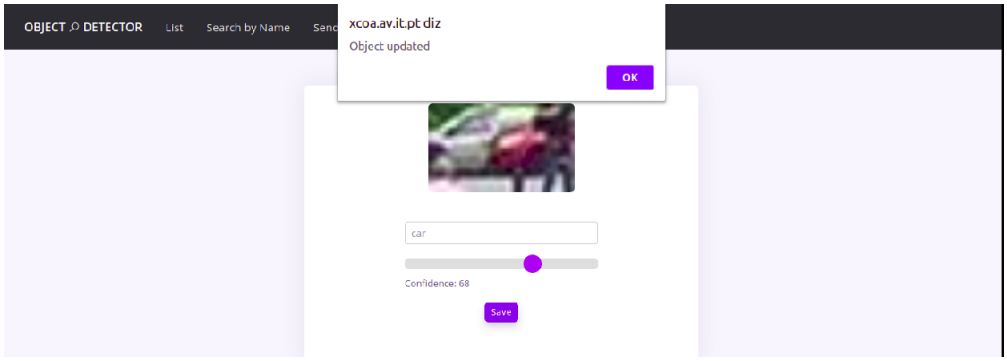

Figura 6.2: Alterar a classe e o nível de confiança

Capítulo 7

Resultados

O projeto foi concluído estando todas as funcionalidades a funcionar como se poderá ver de seguida com o exemplo da seguinte imagem:

Figura 7.1: Imagem original

De seguida é analisada pelo servidor e retorna todos os objetos encontrados nessa imagem.
Na página Search by name é possível pesquisar o objeto pretendido e modificar o nível de confiança.
Na página Search by Type and Color é possível pesquisar não só o objeto como também filtar os objetos com uma determinada cor.

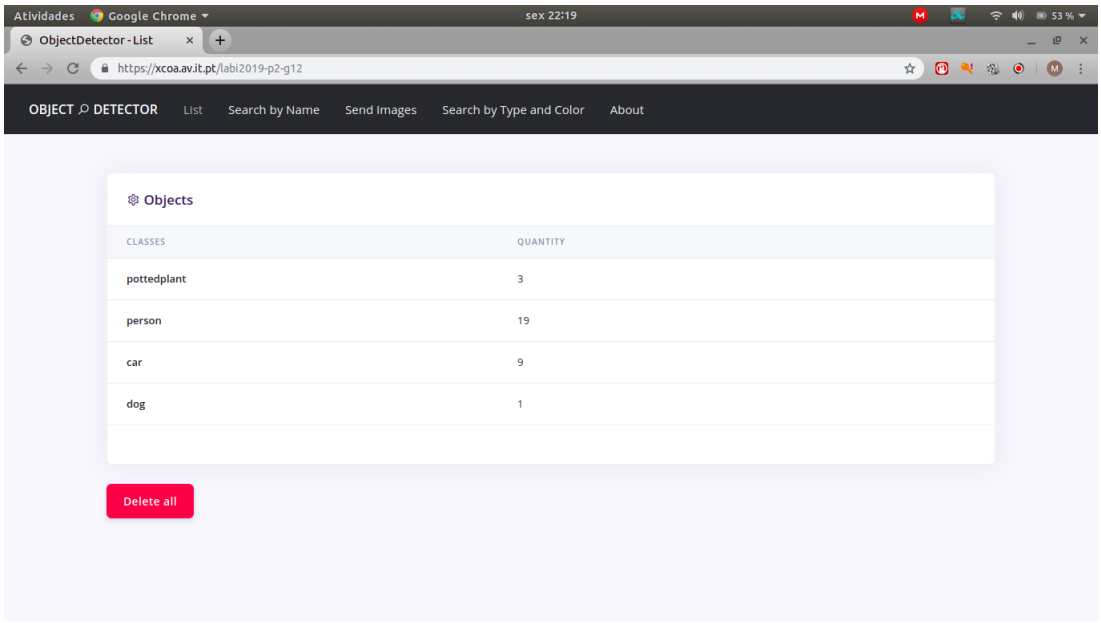

Figura 7.2: Objetos encontrados

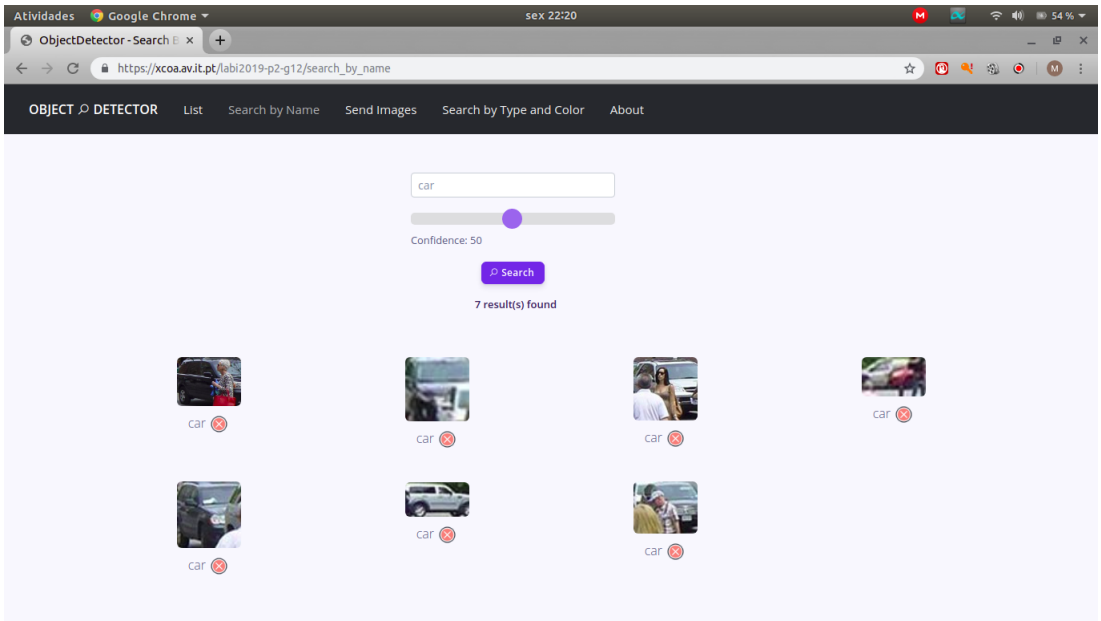

Figura 7.3: Procurar por carro

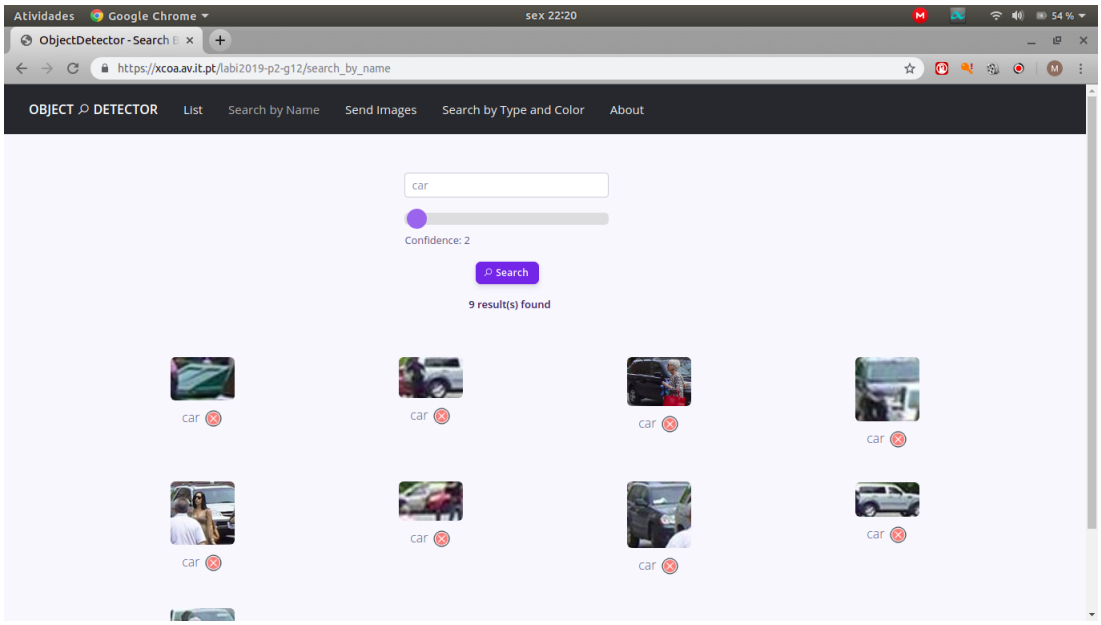

Figura 7.4: Alterar nível de confiança

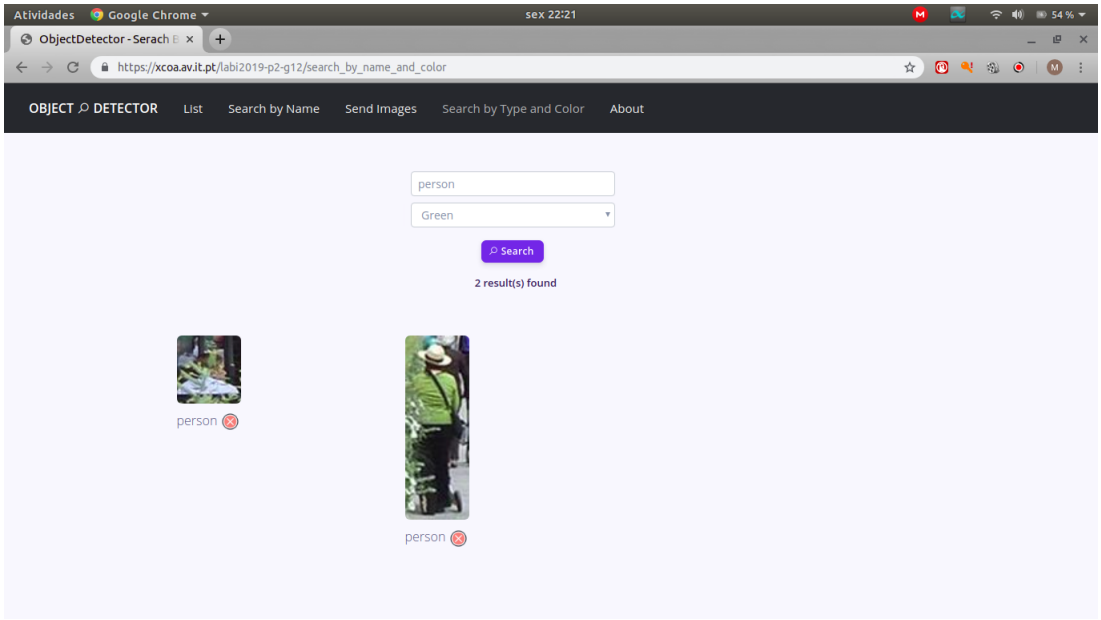

Figura 7.5: Procurar por pessoa e verde

Capítulo 8

Conclusões

Concluímos assim que este projeto foi um sucesso, pois concluímos todas as etapas pedidas. Conseguimos outro objetivo nosso que era deixar tudo simples e fácil de perceber e de trabalhar para o utilizador. Depois de tudo feito, ainda acrescentámos outra funcionalidade, pois achámos que poderíamos ainda melhorar o nosso projeto. Com isto, podemos concluir que acabámos o projeto da melhor maneira e que cumprimos todos os requisitos do enunciado do projeto.

Contribuições dos autores

O autor **Vasco Sousa** fez um total de 25% :

- Python
- HTML

O autor **Francisco Monteiro** fez um total de 25%:

- Relatório
- JS
- Testes e Depuração

O autor **Miguel Cabral** fez um total de 25%:

- Relatório
- CSS
- Base de Dados
- Testes e Depuração

O autor **Tiago Rainho** fez um total de 25%:

- Python
- HTML

Code UA

XCOA

Acrónimos

LABI Laboratórios de Informática

UA Universidade de Aveiro

UC Unidade Curricular

HTML Hyper Text Markup Language

JS JavaScript

CSS Cascading Style Sheets

JSON JavaScript Object Notation