PROJETO FINAL

Universidade de Aveiro

Jorge Catarino, Francisco Martinho, Raquel Rainho, Paulo Vasconcelos

PROJETO FINAL

Departamento de Eletrónica, Telecomunicações e Informática
Laboratórios de Informática

Universidade de Aveiro

Jorge Catarino, Francisco Martinho, Raquel Rainho, Paulo Vasconcelos

11 de junho de 2017

Resumo

Apresentação de uma aplicação capaz de fazer, de editar e carregar para o servidor imagens que serão guardadas numa pasta e cujas características ficarão armazenadas numa base de dados. Para tal, foram utilizadas várias ferramentas tais como o *Cherrypy*, uma framework concebida em *python* com a funcionalidade de servidor, que serve de esqueleto a aplicação, o *SQLite*, um Database Management System (DBMS) de popularidade ascendente que não necessita de configuração e cria bases de dados relacionais, a biblioteca PIL, que permite editar imagens como o programador entender, e o *Ratchet*, uma framework para aplicações web mobile capaz de se adaptar a qualquer telemóvel.

Conteúdo

Lista de Figuras

Capítulo 1

Introdução

A crescente popularidade das redes sociais e a "epidemia"de narcisismo que tem vindo a assolar as faixas etárias mais jovens levou a que aplicações móveis como o *Instagram* tenham alcançado sucesso à escala global.

Ao atentar na página de um utilizador assíduo, podemos seguir o seu dia-a-dia como se a nossa existência nunca abandonasse o seu ombro. Vemos o que este almoçou, visitou e sentiu enquanto documenta as coisas mais banais e quotidianas.

Os memes são os novos governantes da Internet. O que começou como piadas privadas em comunidades restritas, explodiu, sendo agora parte do nosso quotidiano, alcançando sucesso mediático - ainda que nem sempre pelas melhores razões.

O objetivo desta aplicação é juntar estes dois mundos tomando como base o *Instagram*. Pretende também ser intuitiva e apelativa aos vários hipotéticos usuários.

Este documento está dividido em quatro capítulos. Depois desta introdução, no Capítulo 2 é apresentada a metodologia seguida, no Capítulo 3 são apresentados os resultados obtidos, sendo estes discutidos no Capítulo 4. Finalmente, no Capítulo 5 são apresentadas as conclusões do trabalho.

Capítulo 2

Metodologia

Neste capítulo vão ser expostas as várias ferramentas que a aplicação web desenvolvida irá usar, de forma a facilitar a leitura do resto do relatório.

2.1 Framework

Em termos de desenvolvimento de software, uma framework é um conjunto de código escrito previamente, que procura aumentar a eficiência do programador ao fornecer funcionalidades que são comuns a várias aplicações por forma a auxiliar o desenvolvedor.

Esta ferramenta pode ser vista como um esqueleto da aplicação, sendo que vai interligar os vários módulos necessários para o bom funcionamento desta, desde a base de dados até ao editor de imagem. [1]

Neste projeto, a framework usada foi o *Cherrypy*, conforme descrito na subsecção seguinte.

2.1.1 CherryPy

Sendo este projeto composto por diversas frentes com diferentes objetivos, seria impossível concretizar algo assim sem fazer a conexão entre o trabalho realizado por cada secção. O responsável por garantir essas boas ligações é o **CherryPy**.

CherryPy é uma framework para aplicações web desenvolvida em Python. Criada para ser a maior apoiante ao desenvolvimento nesta linguagem (nomeadamente, respeitando a filosofia da linguagem [2]), esta framework permite ao programador escrever uma aplicação web da mesma forma que este desenvolveria um programa para Python, o que pode resultar num código base mais compacto.

[1] Pode ler mais sobre estes módulos na Subseção 2.3.1 e Seção 2.4

[2] Que pode ser lida com o comando "import this"no Python ou aqui

Não só servindo como Framework, o CherryPy serve também como servidor web, com suporte a Web Server Gateway Interface (WSGI) [3]

2.2 Página Web

Numa aplicação web destinada a uso comercial, o Front-End tem uma importância extrema. Sendo o único módulo que o usuário comum normalmente interage, é crucial que este seja intuitivo e atrativo pois, mesmo que o Back-End esteja completamente funcional, uma aplicação difícil de usar dificilmente atrairá utilizadores. É também relevante garantir compatibilidade para o maior número de *Browsers* e aparelhos possível.

Tendo em vista esta finalidade, foi utilizado a framework *Ratchet*.

2.2.1 Ratchet

Ratchet é uma framework open-source para desenvolvimento de Front-End mobile, com HTML, CSS e JavaScript. Esta ferramenta permite criar páginas Web, que podem posteriormente ser adaptadas para vários aparelhos diferentes, desde telemóveis a tablets. Foi desenvolvida por três ex-programadores do Twitter, enquanto desenvolviam a aplicação mobile da rede social. Continua a ser desenvolvida atualmente por um dos criadores e pela comunidade.

Na figura, pode-se ver um exemplo genérico de uma aplicação que usa esta framework, a correr num IPhone.

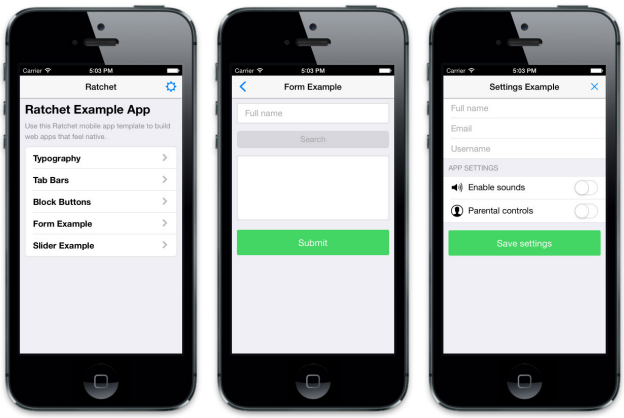

Figura 2.1: Exemplo de uma Aplicação Ratchet

[3] Pode ler mais sobre WSGI aqui

2.3 Persistência

2.3.1 Base de Dados

Uma base de dados é uma ferramenta que tem a capacidade de armazenar e organizar grande quantidade de informação, independentemente do que esta seja.

Com o uso de um DBMS , podemos aceder aos dados nela guardados, podendo pesquisar por certos elementos, adicionar novos dados, atualizar as informações mais antigas e apagar o que já não for necessário, entre outras funções.

Ao longo dos anos, foram desenvolvidos vários modelos de base de dados, começando com o modelo de navegação, popularizado nos anos 60, até as mais recentes bases de dados em XML.

O modelo que irá ser usado neste projeto é o modelo Relacional.

Base de Dados Relacional

Uma base de dados relacional é uma coleção de vários itens de informação, organizada por tabelas, estas podendo ser acedidas conforme necessário, sem ter de ser reorganizadas. Este modelo foi inventado por E.F.Codd, em 1970, na IBM.

Este modelo destaca-se pelo facto de ser possível identificar cada fila de uma tabela, com uma chave única de qualquer tipo, que se dá o nome de chave primária.

É também possível relacionar várias tabelas, através de chaves estrangeiras, isto é, pondo a chave primária de uma tabela noutra.

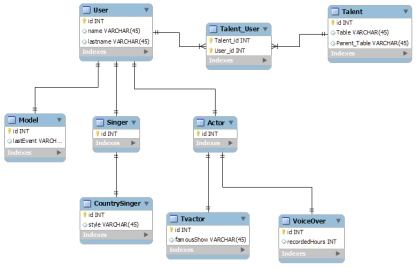

Figura 2.2: Desenho de uma base de dados relacional

Neste projeto, para aceder à base de dados, vai ser usado SQLite, uma DBMS que não necessita de configuração e que não necessita de um servidor para funcionar. Por estas e outras razões, a popularidade desta DBMS tem crescido ao longo dos anos, sendo usada em aplicações como o Facebook e o Firefox.

Este gestor tem a linguagem de programação SQL implementada nele, pelo que os acessos à base de dados vão ser feitos via queries usando esta linguagem.

2.4 Edição de Imagem

Edição de imagem é o processo de alterar uma fotografia de modo a atingir certos objetivos, seja cortar um certo elemento intrusivo, realçar cores ou inserir novos elementos. Este programa deverá ser capaz de aplicar alguns efeitos a imagens e de criar memes, sendo este processo exposto nas subsecções seguintes e demonstrado no Capítulo 3

2.4.1 Módulo de Efeitos

Nesta subsecção vão ser expostos os vários efeitos implementados. Estes foram desenvolvidos usando a Python Imaging Library (PIL), biblioteca de Python que permite a edição de imagem.

Efeito de Inversão

Este efeito, como é óbvio pelo título, inverte as cores de uma foto.

Figura 2.3: Exemplo do Efeito de Inversão

Efeito Cinza

Remove toda a informação crómatica da imagem, deixando apenas a intensidade, o que faz que a imagem fique apenas em tons de cinza.

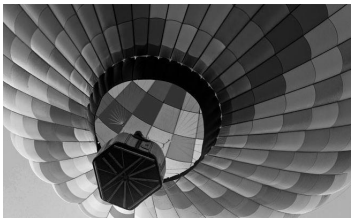

Figura 2.4: Exemplo do Efeito Cinza

Sepia

Altera a imagem, dando efeito de ser uma foto antiga.

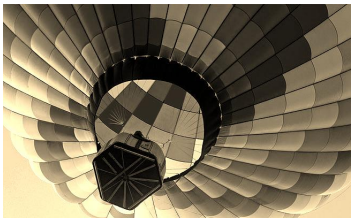

Figura 2.5: Exemplo do Efeito Sepia

Efeito Envelhecido

Pretende alterar a foto de forma a que esta pareça velha, sendo algo semelhante ao efeito da Seção 2.4.1, sendo diferente no facto em que faz parecer com que a foto esteja fisicamente velha.

Figura 2.6: Exemplo do Efeito Envelhecido

Efeito Quadruplicado

Quadriplica a imagem e a cada cópia atribui uma cor diferente.

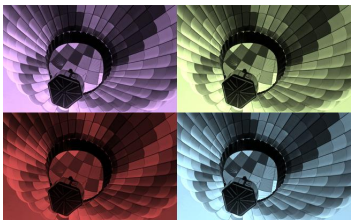

Figura 2.7: Exemplo do Efeito Quadriplicado

Mistura de Imagem

Sobrepõe uma imagem com outra, sendo que a última tem a sua opacidade reduzida.

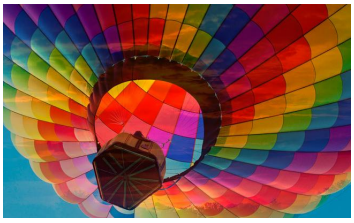

Figura 2.8: Exemplo de Mistura de Imagens

Efeito Preto/Branco

Converte a imagem para BW, deixando-a a preto e branco.

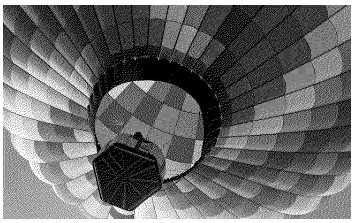

Figura 2.9: Exemplo do Efeito Preto/Branco

Efeito Saturado

Muda a saturação da imagem.

Figura 2.10: Exemplo do Efeito de Saturação

Efeito RedToGreen

Substitui os tons vermelhos da fotografia por tons verdes.

Figura 2.11: Exemplo do Efeito de RedToGreen

Efeito Add

Adiciona as formas e a cor de uma imagem base, com a fornecida pelo usuário.

Figura 2.12: Exemplo do Efeito Add

2.4.2 Módulo de memes

Meme é um termo amplo. A palavra, cunhada por Richard Dawkins no livro ***"O Gene Egoísta"*** publicado em 1976, onde o autor tentava explicar a maneira como a cultura se espalhava, era usada para descrever qualquer ideia, comportamento ou estilo que passava de pessoa a pessoa dentro de uma determinada cultura.

Na cultura da internet, memes podem vir em vários formatos, desde pequenos textos, músicas, vídeos e imagens, normalmente com fins humorísticos.

A aplicação desenvolvida deverá ser capaz de criar um formato de *internet meme* famoso, o image macro, que consiste de uma imagem com texto sobreposto, normalmente na fonte Impact.

Figura 2.13: Exemplo de um image macro que pode ser encontrado na web

À semelhança do outros efeitos, para editar a imagem, vai ser usada a PIL

Capítulo 3

Resultados

3.1 Base de Dados

Como foi anteriormente referido, as base de dados são compostas de tabelas. A base de dados da aplicação contém duas tabelas principais, a images e a users. A primeira, images, armazena todos os dados relativos às imagens enviadas para o servidor. Na users são guardados os dados dos votos de cada utilizador. Na figura abaixo é possível observar a estrutura de ambas as tabelas.

```
CREATE TABLE users(
uid TEXT,
userVotesUp TEXT,
userVotesDown TEXT
);
CREATE TABLE images(
id TEXT,
filepath TEXT,
votes_up INTEGER,
votes_down INTEGER,
category TEXT,
data INTEGER,
location TEXT,
author TEXT);
```

Figura 3.1: Esquema das tabelas da base de dados

Para inserir os dados de uma imagem na base de dados, o servidor chama a função setup(), que tem como argumentos o id, o filepath, o número de votos (que inicialmente é 0), a categoria, a timestamp, a localização e o autor de cada imagem. Com estes argumentos, é feito um *query* de SQL de INSERT. A função UPdate() adiciona um voto, negativo ou positivo, consoante o pretendido.

A função list() devolve uma lista devolve uma lista com todos os dados da base de dados, que será ntão interpretada pelo servidor.

A regVote() regista na tabela users, a imagem em que um determinado usuário votou.

Outra função importante é a VerifyVote() que verifica se o utilizador já votou ou não numa imagem, retornando True caso isto se verificar.

Por fim temos a função removeVote() que serve para remover um voto.

3.2 Edição de imagem

3.2.1 Efeito de Inversão

Para a utilização desta função, foi necessário passar-lhe, como argumentos, a imagem a ser modificada, o diretório onde se encontrava e o tamanho da mesma.

A criação de dois ciclos **for** foi usada de modo a se percorrer todos os pixeis da imagem recebida.A cada pixel foi modificado o valor de r, g e b, subtraindo, ao valor 255, o valor de cada canal, guardando, posteriormente, o valor obtido e, por fim, salvar a imagem modificada.

3.2.2 Efeito Cinza

Como se sabe, existe uma função que converte as imagens para tons de cinza. No entanto, decidiu-se criar uma função própria que gerasse uma imagem com a escala de cinzentos, mas com intensidades diferentes.

3.2.3 Sépia

Sépia é um efeito artístico que se baseia na manipulação de cores. Foi "retirado"do guião que nos foi apresentado. Para este efeito, é necessário apenas manipular os coeficientes aplicados em cada multiplicação (que são aplicados a cada canal).

3.2.4 Efeito Envelhecido

Neste módulo, foi necessário utilizar-se uma figura como base, de modo a se conseguir criar o tipo de efeito pretendido. Posteriormente, com o método **blend()**, foi criada uma nova imagem (juntando a base e a imagem original) e usando o *alpha* igual a 0.7. Para se poder implementar um efeito que permitisse colocar as bordas mais escuras em relação ao centro da imagem aproveitou-se um excerto de código fornecido no guião prático relacionado com a manipulação de imagens.

Para além disso, a imagem foi também convertida em tons de castanho e, para não parecer muito escura, foi utilizado o método point().

3.2.5 Efeito Quadruplicado

Para a função que gera uma imagem quadruplicada foi necessário utilizar uma imagem auxiliar (a original, mas redimensionada) que terá sido colocada quatro vezes por cima da original. No entanto, foram utilizados diferentes cores para cada uma. Para três delas foi utilizado o mesmo método da função sépia, apenas modificando os canais r, g e b. Para a quarta, foi utilizado um método de ImageOps para a colorir com tons avermelhados.

3.2.6 Efeito Preto/Branco

No efeito preto e branco foi apenas utilizada uma conversão de RGB para BW.

3.2.7 Efeito Saturado

Neste módulo, foi necessário converter a imagem para YCbCr

3.2.8 Efeito RedToGreen

Esta função, que recebe como argumentos a imagem e o seu caminho, separa os canais vermelho, verde e azul e, em vez de os colocar por ordem na imagem, coloca no canal vermelho a cor verde.

3.2.9 Efeito Add

O Efeito add, como algumas funções anteriores, foi concretizada utilizando-se um método (add()), que adiciona duas imagens, dividindo o resultado por escala e adicionando o deslocamento.

Capítulo 4

Análise

Ao longo do desenvolvimento deste projeto foram detetados vários problemas, nomeadamente nos tipos de dados devolvidos pelas diferentes funções desenvolvidas em Python no ficheiro *API.py*, estabelecer as ligações corretas entre o javascript do interface e o servidor e o desenvolvimento dos próprios javascript e interface (sendo que, quanto a estes últimos, ficaram por criar os botões de voto positivo e negativo para cada imagem e conseguir pôr em funcionamento alguns outros botões de efeitos).

Apesar de algumas funcionalidades estarem com problemas no Front-End, a restante estrutura do projeto está preparada para a receção, tratamento e armazenamento dos diversos dados.

Depois de alguma reflexão, o atributo de **localização** não foi utilizado. No entanto, este atributo está implementado e armazenado na coluna criada para esse efeito estando capaz de ser utilizado caso se verifique capacidade por parte do interface de envio desses dados. A não utilização deste atributo deveu-se à sua inexistência nas bases de dados dos restantes grupos (o que impossibilitaria a sua sincronização aquando da execução do método *listAll* por parte do servidor e à inexistência de uma forma de introdução desse dado no interface. Este atributo tinha sido pensado como algo a definir de uma de duas formas:

1. Com recurso ao GPS existente no próprio dispositivo móvel, depois das devidas permissões concedidas;

2. A partir da introdução da localização por parte do utilizador (uma vez mais, em semelhança à aplicação *Instagram*).

Capítulo 5

Conclusões

Tendo em consideração todas as dificuldades, superadas e não superadas, é-nos possível atestar o quão enriquecedor pode ser um projeto tão desafiante como este. Dinâmica de grupo e apoio mútuo foram conceitos sempre presentes e consideramos que só assim seria possível concluir algo desta dimensão e complexidade.

Aplicações móveis como estas são, de facto, um sucesso à escala global e o trabalho existente por trás, determinando o bom funcionamento das mesmas, é o que dita a manutenção desse mesmo sucesso. Projetos como estes levam-nos a testar os nossos limites e a excedermo-nos. Consideramos que esse objetivo foi alcançado.

No entanto, consideramos também que, caso tivessemos um pouco mais de tempo poderiamos ter desenvolvido um pouco mais o código desenvolvido pondo assim a aplicação a funcionar devidamente.

Contribuições dos autores

Jorge Catarino - ***database*** 25%

Francisco Martinho - ***Front-End*** 25%

Raquel Rainho - Tratamento de Imagens 25%

Paulo Vasconcelos - ***Back-End*** 25%

Acrónimos

DBMS Database Management System

PIL Python Imaging Library

WSGI Web Server Gateway Interface

Bibliografia

[1] Vários *Slides e guiões de LabI (Guiões 8, 9, 10, 17, 18, 20 e 22)* `elearning.ua.pt/`