Projeto 2

Objetivos:

- Planeamento e preparação do projeto 2.

Este guião apresenta as regras do segundo projeto da disciplina de Laboratórios de Informática.

1.1 Regras

O trabalho deve ser realizado por um grupo de 4 alunos e entregue via a plataforma `http://elearning.ua.pt`, dentro do prazo lá indicado. A entrega deverá ser feita por **apenas um dos membros** do grupo e deve consistir de **um único ficheiro zip!** com o código e o relatório final (em PDF).

Na elaboração do relatório recomenda-se a adoção do estilo e estrutura de relatório descrito nas aulas teórico-práticas e a utilização de recursos de escrita como: referências a fontes externas, referências a figuras e tabelas, tabela de conteúdo, resumo, conclusões, etc.

O relatório tem como objetivos descrever o enquadramento das tecnologias usadas, descrever a implementação, apresentar provas que demonstrem o seu funcionamento correto (p.ex, capturas de ecrã) e analisar os resultados obtidos (p.ex, analisar as capturas).

É obrigatório incluir uma secção "Contribuições dos autores" onde se descrevem resumidamente as contribuições de cada elemento do grupo e se avalia a percentagem de trabalho de cada um. Esta auto-avaliação poderá afetar a ponderação da nota a atribuir a cada elemento.

É obrigatório identificar claramente os autores, indicando também o seu número mecanográfico.

É obrigatório identificar claramente a área utilizada no servidor `XCOA` e o projeto na plataforma `Code.UA`.

1.2 Avaliação

A avaliação irá incidir sobre:

1. **cumprimento dos objetivos através das funções implementadas**,
2. **qualidade do código produzido e dos comentários**,
3. **testes práticos, funcionais e unitários realizados**,
4. **estrutura e conteúdo do relatório**,
5. **utilização das funcionalidades de tarefas do code.ua e git**,
6. **apresentação e discussão do trabalho**

Relatórios meramente descritivos sem qualquer descrição da aplicação, apresentação dos resultados obtidos, testes efetuados, ou discussão serão fracamente avaliados.

Só serão avaliados trabalhos enviados via a plataforma `http://elearning.ua.pt`. Ficheiros corrompidos ou inválidos não serão posteriormente avaliados e não será permitido o reenvio.

Deve ser utilizado um projeto na plataforma Code.UA com um identificador no formato `labi2020-p2-gX`. Substitua o caractere `X` pelo valor 1. Se não for possível criar este projeto, incremente o número até que seja aceite. **Não salte números!**

1.3 Tema Proposto

O objetivo do projeto consiste na criação de um sistema que permita inserir, visualizar e comentar imagens enviadas para uma aplicação, replicando funcionalidades encontradas noutros serviços como o Instagram. Pretende-se que para isto os alunos implementem uma aplicação Web, respetivos módulos de processamento e persistência e depois um conjunto de páginas que permitam o acesso através de um navegador.

Existirá valor na integração dos componentes e na disponibilização de um serviço completo, mas assume-se que os componentes poderão funcionar de forma isolada. Este aspeto também é vital para que seja possível testar o funcionamento isolado de partes do sistema.

1.4 Objetivos

O projeto deve consistir numa aplicação Web, com base de dados e código para processamento das imagens, que deverá ficar a funcionar no servidor XCOA.

O sistema proposto tem de ser composto pelos seguintes componentes:

- **Interface Web**
- **Aplicação Web**
- **Persistência**
- **Processador de Imagens**

1.4.1 Interface Web

Este componente deve ser composto por um mínimo de 5 páginas (funcionalidades), desenvolvidas através de HyperText Markup Language (HTML)[1], JavaScript (JS)[2] e Cascading Style Sheets (CSS)[3], fornecendo o único interface para interação com o sistema. Desde que as funcionalidades aqui pedidas sejam mantidas, os alunos poderão adicionar outras páginas e outras funcionalidades.

A primeira página irá apresentar uma listagem de algumas imagens enviadas para o sistema. A lista de imagens a apresentar pode ser escolhida de acordo com um qualquer algoritmo, devendo ser possível escolher imagens por por data, por número de votos, por palavra chave ou por autor. Através de um elemento de navegação (ex, um botão) ou navegando até ao final da página, será possível aceder às restantes imagens. Cada imagem deverá ser apresentada com a indicação do seu autor, data e palavras chave.

Ao clicar na imagem, o utilizador deverá ir para a página 3. Ao clicar numa palavra chave, a listagem deve ser atualizada com as imagens relativas a essa palavra chave. Ao clicar no nome do autor o utilizador deverá ir para a página 4.

A segunda página irá permitir o envio de uma imagem para o servidor. Juntamente com a imagem o autor deverá indicar o seu nome e um comentário. Poderá igualmente indicar palavras chave relacionadas com a imagem enviada. Não existem noções de autenticação ou se sessões de utilizadores, pelo que os autores indicam o nome a cada inserção.

A terceira página irá permitir aceder a uma imagem. Aqui será apresentada a imagem, o comentário inicial da mesma, o autor, a data da imagem, a localização, e as palavras chave. Deverá ser igualmente apresentado um contador com o número de visualizações de imagem, de votos positivos e votos negativos.

Finalmente, deverão ser apresentados outros comentários associados a esta imagem.

Os utilizadores que acedam à página poderão adicionar um comentário composto por um nome do autor e um texto, assim como votar na imagem (pontos positivos ou negativos).

A quarta página deverá apresentar o perfil de um utilizador, contendo todas as imagens enviadas, comentários efetuados e votos atribuídos.

A quinta página deverá conter informação sobre os alunos e o repositório utilizado.

Todas as páginas deverão obter informação através de pedidos à aplicação Web, que deverá devolver um objeto JavaScript Object Notation (JSON)[4].

1.4.2 Aplicação Web

A aplicação Web consiste num programa **`python`** que serve conteúdos estáticos (`html`, `css`, `js`), apresentando métodos que permitem a navegação entre os diversos componentes.

Irá também fornecer uma interface programática que permita obter informação, ou inserir informação relativa às imagens, autores e comentários efetuados. Esta interface irá responder sempre com objetos JSON.

Assume-se que as imagens são identificadas por um identificador criado de uma síntese do conteúdo da imagem.

Deverão ser considerados pelo menos os seguintes métodos:

- **`/api/list?start=X&filter=Y&order=spec`**: Devolve um objeto contendo uma lista de 10 identificadores de imagens a apresentar. Aceita como argumentos opcionais um filtro a aplicar à listagem, num formato a definir pelos alunos (ex, `author:labiuser1`), um critério de ordenação (ex. `date` ou `views`), assim como o número inicial da imagem a devolver. Este último argumento aplica-se caso existam mais de 10 imagens, permitindo obter imagens além das 10 primeiras. Se o valor for 20, deverão ser devolvidas 10 imagens, com início na vigésima imagem da lista. Caso o campo `secure` esteja definido como verdadeiro, os campos de informação da imagem deverão vir em branco, mantendo-se apenas a data da imagem, votos e as palavras chave. O resultado deverá consistir num objeto com o seguinte formato:

```
{
    "filter": "filtro aplicado",
    "start": "número inicial das imagens",
    "images": [ {
            "id": "identificador da imagem",
            "author": "autor da imagem",
            "date": "data da imagem",
            "location": "coordenadas da imagem",
            "keywords": "palavras chave",
            "votes_up": "votos positivos",
            "votes_down": "votos negativos",
            "views": "visualizações",
            "secure": true ou false
        },...]
}
```

- **`/api/image/add`**: Permite enviar uma nova imagem para ser processada e armazenada, juntamente com o comentário do autor, nome do autor, uma lista de efeitos a aplicar e outros dados que sejam relevantes. Este método deverá aceitar um pedido HyperText Transfer Protocol (HTTP)[5] do tipo `POST` resultante de um formulário com a informação a enviar. O formulário deverá possuir pelo menos os campos `author`, `comment`, `effects` e `image`. Poderão ser também enviadas palavras chave num campo denominado de `keywords`. Se o utilizador assim o desejar, pode indicar uma senha (`password`). Esta senha irá ser utilizada para cifrar o comentário e a imagem armazenada. Caso a imagem possua meta informação, com é exemplo a localização, alguns destes elementos deverão ser guardados. Alguns elementos como a distancia focal, o telemóvel ou câmara utilizados podem ser considerados como palavras chave.

- **`/api/image/details?id=IDENTIFIER&password=senha`**: Permite obter informação de uma imagem. Se a imagem for protegida, é necessário fornecer uma senha no campo `password` para que os campos de detalhe da imagem possuam informação.

  A resposta deverá possuir o seguinte formato:

```
{
    "id": "identificador da imagem",
    "author": "autor da imagem",
    "date": "data da imagem",
    "location": "coordenadas da imagem",
    "keywords": "palavras chave",
    "votes_up": "votos positivos",
    "votes_down": "votos negativos",
    "secure": true ou false,
    "comments": [
        {
            "date": "data do comentário",
            "author": "autor do comentário",
            "texto": "texto do comentário"
        },...
    ]
}
```

- **`/api/comment/add`**: Permite enviar um novo comentário para uma imagem usando um método HTTP `POST`. Este pedido deverá aceitar o envio do autor (`author`) e do comentário (`comment`). Se a imagem for protegida com uma senha, os comentários também deverão ser armazenados de forma cifrada.

- **`/api/vote/add`**: Permite enviar um novo voto para uma imagem usando um método HTTP `POST`. Este pedido deverá aceitar pelo menos o identificador da imagem (`identifier`) e o voto (`vote`). O voto deverá ter uma unidade, podendo ser negativo ou positivo.

- `/api/image/get?id=IDENTIFIER&password=TEXT`: Permite obter uma imagem completa. O método deverá devolver uma imagem ou um redirecionamento (ex. HTTP 301) que leve o navegador ao ficheiro da imagem. O campo `password` é opcional e deve ser indicado caso a imagem necessite de uma senha.

1.4.3 Persistência

Este componente deverá ser composto por métodos que permitem o registo de informação numa base de dados relacional e a obtenção de informação da mesma. Os métodos serão utilizados pela Aplicação Web para registar uma referência para as imagens enviadas, os comentários e os votos.

Deverá ser utilizada uma base de dados `SQLite3`, localizada no mesmo diretório da aplicação.

Os ficheiros contendo as imagens deverão ser armazenados no sistema de ficheiros, sendo que a base de dados relacional apenas terá informação sobre as imagens. Isto é, a base de dados terá informação sobre a data de inserção, comentários e outras propriedades, mas o ficheiro em si não deverá ser armazenado na base de dados. Recomenda-se que as imagens sejam armazenadas com base no seu identificador.

É obrigatório que o acesso à base de dados seja realizada através de métodos desenvolvidos pelos alunos, correspondendo a todas as ações previstas pela aplicação Web (obter listagem de imagens, obter perfil de utilizador, obter informação de uma imagem, votar, inserir imagem, obter nome do ficheiro de uma imagem)

Caso a imagem seja segura, valores como os comentários deverão ser cifrados na inserção e decifrados na obtenção.

1.4.4 Processador de imagens

Este componente, que executa como parte da Aplicação Web, terá métodos para lidar com imagens enviadas para o sistema, ou a obtenção das mesmas. Em particular ele deverá suportar o armazenamento e obtenção de uma imagem, cifrando-a ou decifrando-a oportunamente se assim indicado pelo utilizador.

Deverá igualmente suportar a aplicação de efeitos indicados pelo utilizador no momento do envio. Deverão ser implementados vários efeitos, pelo menos 3 dos quais não contemplados durante a disciplina.

Finalmente, deverá permitir obter informação dos metadados das imagens, em particular a localização indicada na imagem, ou outros textos que possam ser interessantes para a criação de palavras chave.

1.4.5 Testes

Todos os métodos com caráter público dos módulos de persistência, processamento de imagens devem ser cobertos com testes unitários. Estes testes devem no mínimo verificar o funcionamento correto de cada método. Deverão igualmente ser consideradas situações de erro, tais como o envio de argumentos em formatos incorretos ou com valores incorretos.

Os métodos expostos pela Aplicação Web deverão ser igualmente testados, através da realização de pedidos HTTP. Recomenda-se a utilização da biblioteca **`python requests`**. Esta biblioteca permite o envio de pedidos específicos e a obtenção do objeto JSON com o resultado, que pode depois ser validado.

1.5 Bónus

Cada projeto poderá beneficiar de até 2 valores de bónus, caso implemente funcionalidades adicionais de relevo, sendo a sua atribuição definida pelos docentes. Recomenda-se que os alunos discutam potenciais bonificações antes da entrega final.

1.6 Notas Relevantes

1.6.1 Portas TCP

Como estudado, o sistema Linux apenas permite criar um socket em cada porta da mesma família. Não será possível executar múltiplas aplicações, num mesmo servidor, sem que se tomem precauções adequadas. Desta forma, é obrigatório que cada grupo execute a sua aplicação numa porta distinta. Neste caso estarão atribuídas as portas $10000 + n$ em que n é o número do grupo.

Depois será possivel aceder ao `XCOA` de forma a que o servidor redirecione os pedidos HTTP para a aplicação `cherrypy` correta.
Como exemplo, no caso do grupo 8, será possível aceder a
`http://xcoa.av.it.pt/labi2020-p2-g8`, sendo que estes pedidos serão autenticados e reenviados para uma aplicação à escuta na porta 10008. Um acesso à listagem de imagens deste grupo deverá ser feita acedendo ao URL
`https://xcoa.av.it.pt/labi2020-p2-g8/api/list`, sendo que o servidor `XCOA` irá re-enviar a informação para uma aplicação no endereço
`http://127.0.0.1:10008/api/list`.

De forma a iniciar uma aplicação numa porta alternativa, o seguinte excerto pode ser utilizado como base:

```
import cherrypy
cherrypy.config.update({'server.socket_port': 10008,})

class HelloWorld(object):
    def index(self):
            return "Hello World!"

    index.exposed = True

cherrypy.quickstart(HeloWorld())
```

1.6.2 Javascript

Uma componente relevante do projeto assenta na utilização de Javascript e no acesso aos métodos expostos pela Aplicação Web. Recomenda-se que os alunos consultem a documentação de módulos como o **JQuery**, em particular as funcionalidades relacionadas como o acesso a recursos externos (ex. ajax get e post).

De acordo com a documentação presente em `https://api.jquery.com/jQuery.get/`, o exemplo seguinte é relevante para a obtenção de objetos fornecidos pela Aplicação Web.

```
$.get( "api/list?start=0&filter=cats,dogs&order=date")
  .done(function(data) {
    // Got a JSON object from the server.
    console.log(data)
  })
  .fail(function() {
    console.log( "error getting the image list" );
  });
```

Da mesma forma existe um método que permite enviar informação através de um pedido HTTP `POST`: `https://api.jquery.com/jQuery.post/`.

```
$.post( "/api/comment/add", { author: "John", comment: "great pic!" })
  .done(function( data ) {
    // Got answer from server
    console.log(data);
  });
```

Referências

[1] W3C. (1999). HTML 4.01 Specification, URL: `http://www.w3.org/TR/1999/REC-html401-19991224/`.

[2] ECMA International, *Standard ECMA-262 – ECMAScript Language Specification*, Padrão, dez. de 1999. URL: `http://www.ecma-international.org/publications/standards/Ecma-262.htm`.

[3] W3C. (2001). Cascading Style Sheets Level 2 Revision 1 (CSS 2.1) Specification, URL: `http://www.w3.org/TR/2011/REC-CSS2-20110607/`.

[4] E. T. Bray, *The JavaScript Object Notation (JSON) Data Interchange Format*, RFC 7159, Internet Engineering Task Force, mar. de 2014.

[5] R. Fielding, J. Gettys, J. Mogul, H. Frystyk, L. Masinter, P. Leach e T. Berners-Lee, *Hypertext Transfer Protocol – HTTP/1.1*, RFC 2616 (Draft Standard), Updated by RFCs 2817, 5785, 6266, Internet Engineering Task Force, jun. de 1999.

Glossário

CSS	Cascading Style Sheets
HTML	HyperText Markup Language
HTTP	HyperText Transfer Protocol
JS	JavaScript
JSON	JavaScript Object Notation
URL	Uniform Resource Locator