Ano letivo: 2022/2023

Curso: Lic. Engenharia De Redes E Sistemas De Computadores

Unidade Curricular	Programação Web

Lic.	Ano do curso	2º ano	2º semestre	ECTS	

NOME do ALUNO: Nuno Miguel Lopes Faria

Prova Escrita

Versão: B

Duração: 100 minutos

Leia atentamente toda a prova antes de iniciar.

A prova é individual, não sendo permitido consultar os seus colegas. No entanto, pode consultar os apontamentos das aulas e a Internet.

O resultado final deve ser enviado para o moodle incluindo o Word da prova e PDF da prova (gravar como PDF) e os ficheiros HTML e JS desenvolvidos. Deve ser anexado o link para Github no tópico Avaliação.

No documento de resposta deve ser incluída a versão da prova.

Durante a resolução deve ir gravando o trabalho para salvaguardar as alterações.

Parte I (25 valores)

1. À luz do que aprendeu na UC, comente a seguinte imagem.

Cofinanciado por:

POCH PORTUGAL 2020 UNIÃO EUROPEIA Fundo Social Europeu

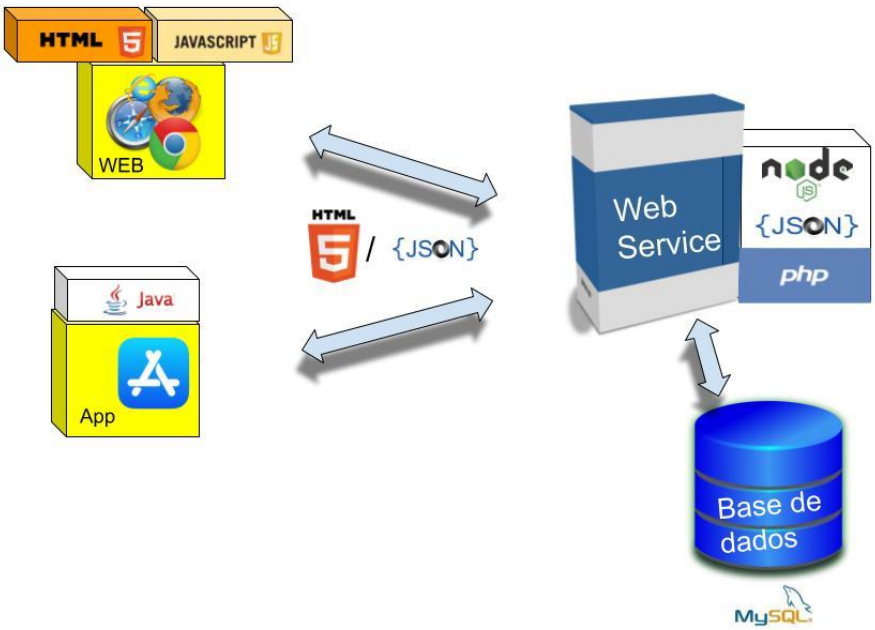

Figura 1 - Estrutura do documento

O cliente (web ou app) envia um pedido para o serviço web.

O serviço web processa esse pedido, possivelmente interagindo com a base de dados para obter ou armazenar informações.

O serviço web retorna uma resposta no formato JSON ao cliente.

O cliente processa essa resposta e apresenta os dados ao utilizador final.

Cofinanciado por:

POCH PORTUGAL 2020 UNIÃO EUROPEIA Fundo Social Europeu

2. Crie um protocolo para os alunos do IPVC para almoçar na cantina. Para que servem os protocolo e dê um exemplo

A existência de um protocolo permite uma melhor organização e eficiência no funcionamento da cantina do IPVC.

Horários de Atendimento:

Almoço: 12h30 às 14h30.

Reserva de Refeições:

Os alunos devem reservar as suas refeições até as 10h do mesmo dia no site associal.

Cada aluno pode escolher diferentes opções de menu (normal, vegetariano, grill).

Entrada na Cantina:

Os alunos devem passar o seu cartão de estudante na cantina quando estiverem a recolher a comida.

A entrada será permitida apenas no horário reservado.

Fila e Atendimento:

Os alunos devem formar filas ordenadas para recolher a comida.

O atendimento será feito por ordem de chegada, respeitando as filas.

Serviço de Refeições:

As refeições são servidas em bandejas individuais.

Cada aluno deve pegar sua bandeja e dirigir-se ao local para pegar a sua comida.

Consumo de Refeições:

Os alunos sentam-se na mesa que estiver livre para comer a sua refeição.

Após terminar a refeição, os alunos têm de colocar a sua bandeja no local indicado.

Exemplo: O Nuno, um aluno do IPVC, reserva a sua refeição normal para o dia seguinte. No dia marcado, ele chega à cantina às 12h25, pega na bandeja e nos talheres e vai pegar a sua comida, antes de pegar a comida ele passa o cartão que demonstra a refeição que ele comprou. Apos

Cofinanciado por:

pegar na comida o aluno senta-se, come e depois de comer ele levanta-se e coloca a bandeja no local indicado e vai embora.

Parte II (25 valores)

1. Considera os seguintes exemplos de objetos DOM.
   - document.getElementById(id)
   - document.getElementsByTagName(tagName)
   - document.getElementsByClassName(className)

   Porque no primeiro caso temos getElement e nos dois seguintes getElements? Dê um exemplo de utilização para cada exemplo

   No primeiro caso, document.getElementById(id) retorna um único elemento (singular) porque id tem de ser único no documento HTML.

   Nos outros 2 casos, document.getElementsByTagName(tagName) e document.getElementsByClassName(className) retornam um conjunto de elementos (plural) porque podem haver múltiplos elementos que compartilham o mesmo tagName ou a mesma classe.

2. Cria uma estrutura em JSON para registar Atores e Filmes. Faz um XML para a mesma estrutura. Comenta os resultados

   Os ficheiros Json e XML encontram-se no GitHub.

São ambos boas formas de representar dados estruturados, porem são escolhidos dependendo da necessidade do projeto, o JSON é mais utilizado em aplicações web modernas por causa da sua simplicidade e integração direta com JavaScript, enquanto o XML é útil em contextos onde a complexidade dos dados são mais importantes.

Parte III (20 valores)

1. Qual a diferença entre <p> e <pre>

   O <p> é utilizado para texto que não necessita de formatação especial, fazendo com que o navegador cuide das quebras de linha e dos espaços.

Cofinanciado por:

O <pre> é utilizado para texto onde a formatação exata de espaços e quebras de linha deve ser preservada.

2. Para que serve <meta charset="utf-8">

   A utilização do <meta charset="utf-8"> serve para que o navegador interprete corretamente os caracteres do documento HTML, fazendo com que todo o texto seja mostrado corretamente, independentemente do idioma ou dos caracteres especiais usados.

Parte IV (30 valores)

1. Prepara uma página com uma tabela 2x2 com estilos CSS que permitam apresentar 4 marcas de produtos de rede. Usa cores de fundo e cores de escrita e o logotipo de cada marca.

   A resposta esta no GitHub.

Parte V (50 valores)

1. Usando o Bootstrap, construa uma página com cards que mostre 6 monumentos e atrações turísticas do seu local de residência.
2. Cada card tem de ter um botão “ver mais” para ver mais detalhes.

   As respostas estão no GitHub.

Parte VI (50 valores)

Considere as imagens seguintes.

Cofinanciado por:

routes > products.js > ...

```
1   const productsRouter = require('express').Router();
2   const controller = require('../controllers/products');
3   const authMiddleware = require('../middlewares/auth/auth');
4
5
6
7
8
9
10
11
12  module.exports = productsRouter;
```

Figura 2 - Rotas

controllers > products.js > ...

```
1   const apiResponse = require('../utils/response/apiResponse');
2   const Products = require('../data/entities/products');
3
4   exports.getAll = async (req, res) => {···
15  }
16
17  exports.getById = async (req, res) => {···
30  }
31
32  exports.create = async (req, res) => {···
49  }
50
51  exports.update = async (req, res) => {···
72  }
73
74  exports.delete = async (req, res) => {···
92  }
```

Figura 3 - Controller Produtos

Cofinanciado por:

1.1 - Complete o ficheiro de rotas dos produtos.
A resposta esta no GitHub (products.js)

1.2 - Explique cada uma das linhas do ficheiro anterior

```
const productsRouter = require('express').Router();
```

A linha acima serve para manipular as rotas relacionadas com os produtos.

```
const controller = require('../controllers/products');
```

A linha acima serve importar o modulo de controladores.

```
const authMiddleware = require('../middlewares/auth/auth');
```

A linha acima serve para ver se o utilizador esta autenticado para aceder as rotas privadas.

```
productsRouter.use((err, req, res, next) => {
  console.error(err.stack);
  res.status(500).send('Erro');
});
```

As linhas acima servem para deteção de erro

```
productsRouter.get('/', authMiddleware, controller.getAllProducts);
productsRouter.get('/:id', authMiddleware, controller.getProductById);
productsRouter.post('/', authMiddleware, controller.createProduct);
productsRouter.put('/:id', authMiddleware, controller.updateProduct);
productsRouter.delete('/:id', authMiddleware, controller.deleteProduct);
```

As 5 linhas acima definem as rotas para os produtos, incluindo obter todos os produtos, produto por ID, criar um produto, atualizar um produto e eliminar um produto.

```
module.exports = productsRouter;
```

A linha acima serve para exportar para que possa ser usado em outras partes de uma aplicação.

1.3 - Desenvolva um ficheiro JSON que permita guardar a informação dos produtos e escreva o código para cada um dos métodos do controller products.

2. O Resultado final da prova escrita deve ser colocada no github sendo partilhado o link como resposta à prova

Bom trabalho!

António Lira Fernandes

Cofinanciado por: