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:

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.

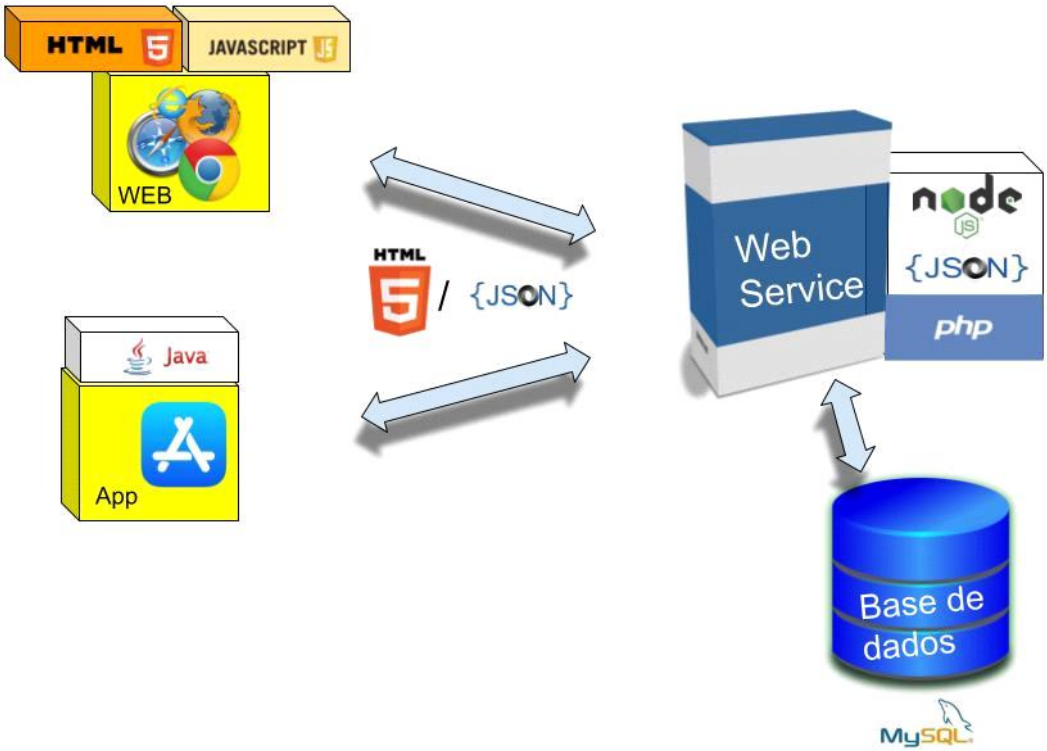
O resultado final deve ser enviado para o moodle incluindo o Word da prova e PDF da prova (gravar como PDF) e os ficheiros HTML e JS desenvolvidos. Deve ser anexado o link para Github no tópico Avaliação.

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)

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



Há sempre um web Service/Servidor a servir como intermediario entre o que vemos no browser/aplicação e a base de dados, em que o cliente(browser/aplicação) envia pedidos ao servidor, neste caso pedidos de dados, o servidor processa o pedido do cliente e interage com a base de dados aguardando uma resposta, a base de dados retorna o pedido feito ao servidor e o servidor retorna os dados ao cliente.

Figura 1 - Estrutura do documento

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

1. Aluno chega e identifica-se
2. Há uma fila na qual o aluno deve de esperar
3. Há a escolha do aluno para a sua refeição
4. O aluno paga a sua refeição

5. O aluno consome aquilo que adquiriu

6. O aluno descarta e limpa o seu lixo

7. O aluno sai do refeitorio

Um protocolo serve como base fundamental para uma comunicação eficiente e confiavel, fornecendo um conjunto de regras e padrões que orientam essa comunicação.

Parte II (25 valores)

- 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

Enquanto o primeiro retorna apenas um único elemento uma vez que o ID é único dentro do html, os outros dois podem retornar vários elementos já que podem haver varios com a mesma tagName e o mesmo className.

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

Ambos mostram resultados identicos, mas enquanto o json é aplicado mais num contexto de programação web, o xml é usado onde a validação de estrutura é crítica, devido à sua complexidade .

Parte III (20 valores)

- Qual a diferença entre <p> e <pre>

Enquanto a tag <p> define um parágrafo de texto automáticamente formatado para aquela página, a tag <pre> define texto pré-formatado, usado para que se mantenha a formatação precisa, por exemplo, código fonte ou ASCII.

- Para que server

  <meta charset="utf-8">

A tag <meta charset="utf-8"> é essencial para garantir que o conteúdo de uma página HTML seja exibido corretamente, independentemente dos caracteres usados. Ao escolher "utf-8" estamos a garantir a compatívilidade uma ampla gama de dispositivos e browsers.

Parte IV (30 valores)

- 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.

Parte V (50 valores)

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

Parte VI (50 valores)

Considere as imagens seguintes.

routes > JS 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 > JS 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

1.1 - Complete o ficheiro de rotas dos produtos.

1 - productsRouter.get("/", controller.getAll);

2 - productsRouter.get("/:id", controller.getById);

3 - productsRouter.post("/", authMiddleware, controller.create);

4 - productsRouter.update("/id", authMiddleware, controller.update);

5 - productsRouter.delete("/:id", authMiddleware, controller.delete);

1.2 - Explique cada uma das linhas do ficheiro anterior

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

https://github.com/SakanaPT/PW/tree/main/teste

Bom trabalho!

António Lira Fernandes