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)

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

Cliente: O cliente é um aplicativo que solicita dados ou serviços do serviço web. O cliente pode ser um aplicativo web, um aplicativo móvel ou qualquer outro tipo de aplicativo.

Serviço web: O serviço web é um programa que fornece dados ou serviços para o cliente. O serviço web é geralmente hospedado em um servidor web.

Protocolo HTTP: O protocolo HTTP é usado para comunicação entre o cliente e o serviço web. O protocolo HTTP define como o cliente deve solicitar dados ou serviços do serviço web e como o serviço web deve responder às solicitações do cliente.

Formato de dados JSON: O formato de dados JSON é usado para codificar e decodificar dados trocados entre o cliente e o serviço web. O formato JSON é um formato de dados leve e fácil de usar que é amplamente suportado por aplicativos e serviços web.

Base de dados: O base de dados armazena os dados que são fornecidos pelo serviço web. O base de dados pode ser um base de dados relacional, como MySQL, ou um base de dados não relacional, como NoSQL.

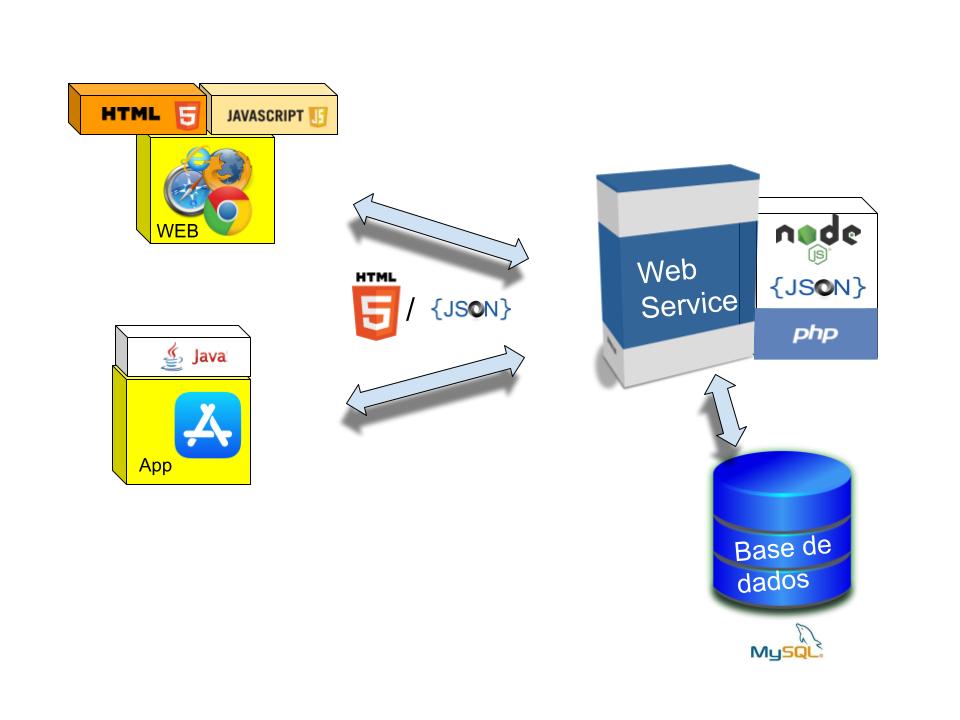


Figura 1 - Estrutura do documento

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

Importância dos Protocolos

Organização: Protocolos ajudam a organizar o fluxo de alunos, evitando filas longas e aglomerações.

Segurança: Garantem que medidas de segurança e higiene sejam seguidas, especialmente importante em tempos de pandemia.

Eficácia: Facilitam a operação diária, permitindo que o serviço de cantina funcione de maneira suave e eficiente.

Satisfação: Contribuem para a satisfação dos alunos, proporcionando uma experiência agradável e sem complicações.

Exemplo de Implementação:

Entrada na Cantina

O aluno chega à cantina e apresenta o cartão de estudante na entrada.

O sistema li o cartão e regista a entrada do aluno.

Escolha da Refeição

O aluno consulta o menu disponível online ou no painel da cantina.

Faz sua escolha e se dirige à área correspondente para retirada.

Pagamento

O aluno realiza o pagamento utilizando o método escolhido (cartão de estudante, cartão de crédito/débito, ou dinheiro).

O sistema debita o valor correspondente e emite um recibo.

Higienização e Condução

O aluno higieniza as mãos nas estações disponíveis.

Utiliza máscara até iniciar a refeição e segue as filas organizadas.

Descarte de Resíduos

Após a refeição, o aluno separa os resíduos em orgânicos e recicláveis.

Descarte os resíduos nas estações apropriadas antes de sair da cantina.

Parte II (25 valores)

1. Considera os seguintes exemplos de objetos DOM.

* document.getElementById(id)

<!DOCTYPE html>

<html>

<head>

<title>Exemplo getElementById</title>

</head>

<body>

<h1 id="titulo">Hello World</h1>

<script>

var titulo = document.getElementById("titulo");

titulo.style.color = "blue";

</script>

</body>

</html>

* document.getElementsByTagName(tagName)

<!DOCTYPE html>

<html>

<head>

<title>Exemplo getElementsByTagName</title>

</head>

<body>

<p>Parágrafo 1</p>

<p>Parágrafo 2</p>

<p>Parágrafo 3</p>

<script>

var paragrafos = document.getElementsByTagName("p");

for (var i = 0; i < paragrafos.length; i++) {

paragrafos[i].style.color = "green";

}

</script>

</body>

</html>

* document.getElementsByClassName(className)

<!DOCTYPE html>

<html>

<head>

<title>Exemplo getElementsByClassName</title>

</head>

<body>

<div class="caixa">Caixa 1</div>

<div class="caixa">Caixa 2</div>

<div class="caixa">Caixa 3</div>

<script>

var caixas = document.getElementsByClassName("caixa");

for (var i = 0; i < caixas.length; i++) {

caixas[i].style.border = "2px solid red";

}

</script>

</body>

</html>

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

Os métodos getElementsByTagName e getElementsByClassName retornam coleções de elementos porque é comum haver múltiplos elementos que compartilham a mesma tag ou classe, enquanto getElementById retorna um único elemento porque um ID deve ser único dentro de um documento HTML.

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

Ambos têm suas vantagens e desvantagens, e a escolha entre JSON e XML depende do contexto e dos requisitos específicos do projeto. JSON é geralmente preferido para comunicação rápida e leve em aplicações web, enquanto XML pode ser preferido para documentos mais complexos e onde a validação e a estrutura rígida são necessárias.

Parte III (20 valores)

1. Qual a diferença entre <p> e <pre>

<p> (parágrafo):

Define um parágrafo de texto.

O texto dentro da tag <p> é formatado como um parágrafo normal, com espaçamento automático antes e depois.

É usado para texto geral, como artigos, notícias ou posts em blogs.

<pre> (texto pré-formatado):

Define texto pré-formatado.

O texto dentro da tag <pre> é preservado exatamente como está escrito, incluindo espaços, tabulações e quebras de linha.

É usado para exibir código, scripts, logs ou qualquer outro texto que precise manter sua formatação original.

1. Para que server <meta charset="utf-8">

A tag <meta charset="utf-8"> serve para definir a codificação de caracteres usada em uma página HTML.

Imagine que o computador precisa transformar o texto que você escreve (por exemplo, "palavra") em um formato que ele possa entender. A codificação de caracteres define como isso é feito.

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.

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.

Parte VI (50 valores)

Considere as imagens seguintes.



Figura 2 - Rotas



Figura 3 - Controller Produtos

1.1 - Complete o ficheiro de rotas dos produtos.

1.2 - Explique cada uma das linhas do ficheiro anterior

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

Cria uma nova instância do roteador do Express, que é usada para definir as rotas relacionadas aos produtos.

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

Importa o módulo do controlador de produtos, que contém as funções de lógica de negócios para gerenciar produtos.

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

Importa o middleware de autenticação, que será usado para proteger as rotas e garantir que apenas usuários autenticados possam aceder a essas rotas.

productsRouter.get('/', authMiddleware, controller.getAll);

Define uma rota GET na raiz do roteador de produtos. Quando um cliente faz uma solicitação GET para esta rota, o middleware de autenticação é executado primeiro e, em seguida, a função getAll do controlador é chamada para listar todos os produtos.

productsRouter.get('/:id', authMiddleware, controller.getById);

Define uma rota GET para obter um produto específico pelo seu ID. O middleware de autenticação é executado primeiro, seguido pela função getById do controlador.

productsRouter.post('/', authMiddleware, controller.create);

Define uma rota POST para criar um novo produto. O middleware de autenticação é executado primeiro, seguido pela função create do controlador.

productsRouter.put('/:id', authMiddleware, controller.update);

Define uma rota PUT para atualizar um produto existente pelo seu ID. O middleware de autenticação é executado primeiro, seguido pela função update do controlador.

productsRouter.delete('/:id', authMiddleware, controller.delete);

Define uma rota DELETE para deletar um produto específico pelo seu ID. O middleware de autenticação é executado primeiro, seguido pela função delete do controlador.

module.exports = productsRouter;

Exporta o roteador de produtos para que ele possa ser usado em outras partes da aplicação, geralmente no arquivo principal da aplicação, onde todas as rotas são combinadas e configuradas.

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