ESCOLA
SUPERIOR
DE MEDIA
ARTES
E DESIGN
POLITÉCNICO
DO PORTO

ALP

FICHA DE EXERCÍCIOS
ATIVIDADE LETIVA

Programação Web I
Ficha 02 - Introdução ao Vue.js

Abra o VSC e crie projetos Vue para cada exercício com o nome **VueExN** onde N é o número do exercício.

Exercício 01 – Hierarquia de componentes

1. Crie um componente **UserProfile** que exiba informações de um utilizador numa tabela com 1 linha e 2 colunas. Dentro da primeira célula invoque um componente **UserAvatar** com a foto do utilizador e, na segunda, um componente chamado **UserDetails** deve mostrar o nome e o email.

2. Crie um componente **ShoppingCart**, que invoca um componente **CartItem** repetidamente (três vezes) para exibir cada item. Cada **CartItem** deve ser uma tabela com três colunas. As duas primeiras serão usadas para o nome e o preço do produto alimentadas pelo componente **ItemDetails**. A última coluna deve invocar um componente **RemoveButton** para apresentar um botão (não funcional) para remover o item.

Exercício 02 – Interpolação (texto, HTML, atributos e expressões JS)

3. Crie um componente chamado **UserData** para exibir informação de um determinado utilizador num template HTML.

a. Inclua as seguintes propriedades na sua função **data**:

```
data() {
    return {
        userName: "João",
        userAge: 17,
        userBio: "É um programador <em>frontend</em> que adora criar aplicações em Vue.js."
    };
}
```

b. Exiba numa ** **, em parágrafos independentes, as seguintes mensagens:

> Bem-vindo, João!
> João tem 17 anos e, por isso, é menor de idade.
> É um programador *frontend* que adora criar aplicações em Vue.js.

Dica: use os seguintes tipos de interpolação:

- 1ºparágrafo: interpolação textual
- 2ºparágrafo: interpolação através de expressão JS
- 3ºparágrafo: interpolação HTML

4. Crie um componente **CityImage** que usa interpolação para definir atributos de um elemento **img**. A URL da imagem e o texto alternativo devem ser interpolados a partir de dados do componente.
    a. Inclua os seguintes dados na função data do componente:
        i. Texto alternativo: "Vila do Conde"
        ii. Link da imagem: https://i.ytimg.com/vi/1rv77537JaM/maxresdefault.jpg
    b. Exiba a imagem das seguintes formas:
        i. Use interpolação por atributo
        ii. Por referência local a partir da pasta **public**
        iii. Por referência local a partir da pasta **src/assets**

Exercício 03 – Diretivas v-bind e v-on

5. Crie um componente **ProductData** para listar detalhes de um produto
    a. Use a diretiva **v-bind** no template HTML para apresentar os seguintes dados dinamicamente:

```
data() {
    return {
        name: "Desktop Dell i5 3.20GHz 8GB RAM 500GB",
        image: "https://i.ebayimg.com/images/g/MbUAAOSwI7ZimiI5/s-l1200.jpg",
        link: "https://www.ebay.com/itm/284416443932",
        isDisabled: true
    }
},
```

Dica: use os seguintes tipos de vinculação:

- 1ºparágrafo: uso da tag **<h2>** com interpolação textual (sem **v-bind**)
- 2ºparágrafo: uso da tag <**img src>** com vinculação dinâmica
- 3ºparágrafo: uso da tag **<a href>** com vinculação dinâmica
- 4ºparágrafo: uso da tag **<button disabled>** com vinculação dinâmica

   b. Crie uma propriedade **attr** com o valor "href" e altere a vinculação do link para que este apresente o argumento de forma dinâmica
   c. Altere a sintaxe da vinculação da imagem pela sua abreviação

6. Crie um componente **DynamicNumber** para modelar uma pequena aplicação que incrementa/ decrementa um número.
   a. Defina a propriedade **num** com o valor 0
   b. Apresente a propriedade **num** no template
   c. Crie dois botões no template:
      i. Botão "-": decrementa a propriedade **num** de uma unidade
      ii. Botão "+": incrementa a propriedade **num** de uma unidade

Exercício 04 – Métodos e propriedades computadas

7. Crie um componente que tenha uma lista de números e uma propriedade computada que calcule a soma dos números.
8. Crie um componente com propriedade **school** com o valor "esmad". Use um método para transformar o texto em maiúsculas e uma propriedade computada para contar o número de caracteres.
9. Crie um componente chamado **PersonData** que tem como objetivo familiarizar o aluno com as propriedades calculadas. Implemente os seguintes passos:
   a. Defina o objeto person na fundação data:
      `person: {firstName: "Rui", lastName: "Silva", age: 23}`
   b. Mostre o objeto **person** no template, numa tag **<ul>**
   c. Crie 3 funções de ciclo de vida nos momentos da criação, montagem e atualização. As mensagens devem surgir na consola no form: "EVENTO: E" onde E pode ser CRIAÇÃO, MONTAGEM ou ATUALIZAÇÃO
   d. Crie um método chamado **printDataPerson** que deve imprimir na consola:
      "METHOD--> NOME: N e IDADE: I"
      onde N e I são o primeiro nome e a idade da pessoa

e. Adicione a invocação ao método no template

f. Crie uma propriedade calculada chamado **printDataPersonComputed** que deve imprimir na consola a seguinte string:

"COMPUTED--> NOME: N e IDADE: I"

onde N e I são o primeiro nome e a idade da pessoa

g. Invoque 2 vezes o método e 2 vezes a propriedade calculada no template;

h. Altere através da DevTools a idade da pessoa e verifique a consola,

nomeadamente o nº de vezes que o método e a PC são chamadas;

i. Crie uma PC chamada **getFullNameComputed** para devolver o nome completo da pessoa;

j. Crie um watcher para observar qualquer alteração da idade da pessoa. Nesse caso, deve imprimir na consola: "Idade alterada!";

k. Melhore o watcher anterior para que exiba a seguinte mensagem:

"Idade alterada de I1 para I2"

onde I1 e I2 são as idades antiga e nova da pessoa, respetivamente

Exercício 05 – Vinculação de classe e estilos

10. Crie um componente **Card** onde o utilizador pode clicar em botões para alterar o tema de um cartão de informação. Dependendo do tema selecionado, a cor e o estilo do cartão mudam.

    a. Crie a propriedade **theme** com o valor "light"

    b. Crie no template um cartão que exiba um título e uma descrição.

```
<div class="card">
 <h2>Título do Cartão</h2>
 <p>Este é um exemplo de cartão.</p>
</div>
```

    c. Adicione três botões para selecionar entre os temas "Claro", "Escuro" e "Colorido".

```
<button>Claro</button>
<button>Escuro</button>
<button>Colorido</button>
```

    d. Defina as regras CSS

```
<style>
.card {
  width: 300px;
  padding: 20px;
  border-radius: 10px;
  text-align: center;
  margin-top: 20px;
```

```
  transition: background-color 0.3s ease, color 0.3s ease;
}
.light {
  background-color: white;
  color: black;
  border: 1px solid #ccc;
}
.dark {
  background-color: black;
  color: white;
  border: 1px solid #333;
}
.colorful {
  background-color: orange;
  color: blue;
  border: 1px solid green;
}
```

11. Crie um componente **DynamicFontSize** onde o utilizador pode ajustar o tamanho do texto.
    a. Crie uma propriedade **fontSize** com o valor 20
    b. Adicione o texto "Olá Mundo!" numa ** **:
    c. Use a diretiva **v-bind** no atributo **style** da ** ** para aplicar dinamicamente o estilo **fontSize**
    d. Crie dois botões "+" e "-" que incrementam e decrementam, respetivamente o tamanho da fonte

Exercício 06 – Renderização condicional

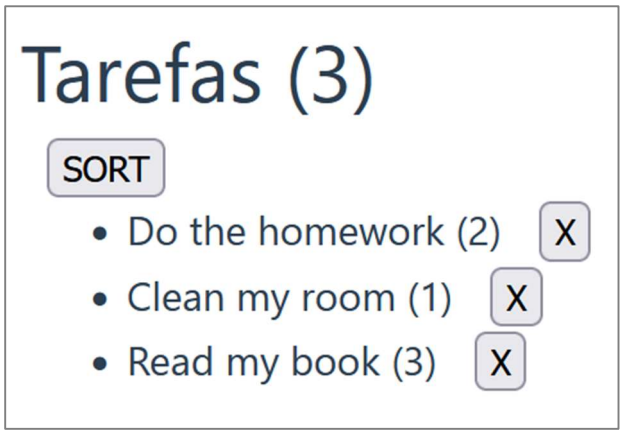
12. Crie um componente **UserCard** que exibe ou esconde os detalhes de um utilizador (nome, idade e localidade) com base numa propriedade booleana **showDetails**. A variável deve alternar o seu valor após ser premido um botão de alternância com o texto MOSTRA ou ESCONDE. Use **v-if** para controlar a renderização.
13. Crie um componente **Grade** que exibe a situação de um aluno mediante a sua nota. Os possíveis estados são:
    a. Reprovado: [0-7]
    b. Oral:]7-10[
    c. Aprovado: [10-20]
    d. Nota inválida: qualquer outro valor

    Use **v-if** e **v-else-if** para controlar a mensagem exibida.

Exercício 07 – Renderização cíclica

14. Crie um componente **SchoolsList** para apresentar as escolas do P. PORTO
    a. Defina uma propriedade **schools** com o nome de todas as escolas do P. PORTO.
    b. Exiba cada escola em linhas independentes numa tabela

15. Crie um componente **TodoList** para listar as tarefas de um utilizador



    a. Defina uma propriedade **tasks** com o seguinte valor:

```
[
    {desc: "Do the homework", priority: 2},
    {desc: "Clean my room", priority: 1},
    {desc: "Read my book", priority: 3},
]
```

    b. Crie um **<h1>** com o texto "Tarefas (T)" onde T é o nº de tarefas atual
    c. Exiba cada descrição de uma tarefa num item de lista (**li**).
    d. Adicione a prioridade colocando-a após a descrição entre parêntesis
    e. Crie um botão "SORT" no topo que ordene as tarefas por prioridade (1 – mais importante)
    f. Adicione um botão "X" a cada tarefa que quando premido deve remover a tarefa
    g. Por fim, quando não existirem tarefas, remova o botão SORT