Missão Prática - Nível 2

//Disciplina: Descobrindo o JavaScript – 1º Procedimento

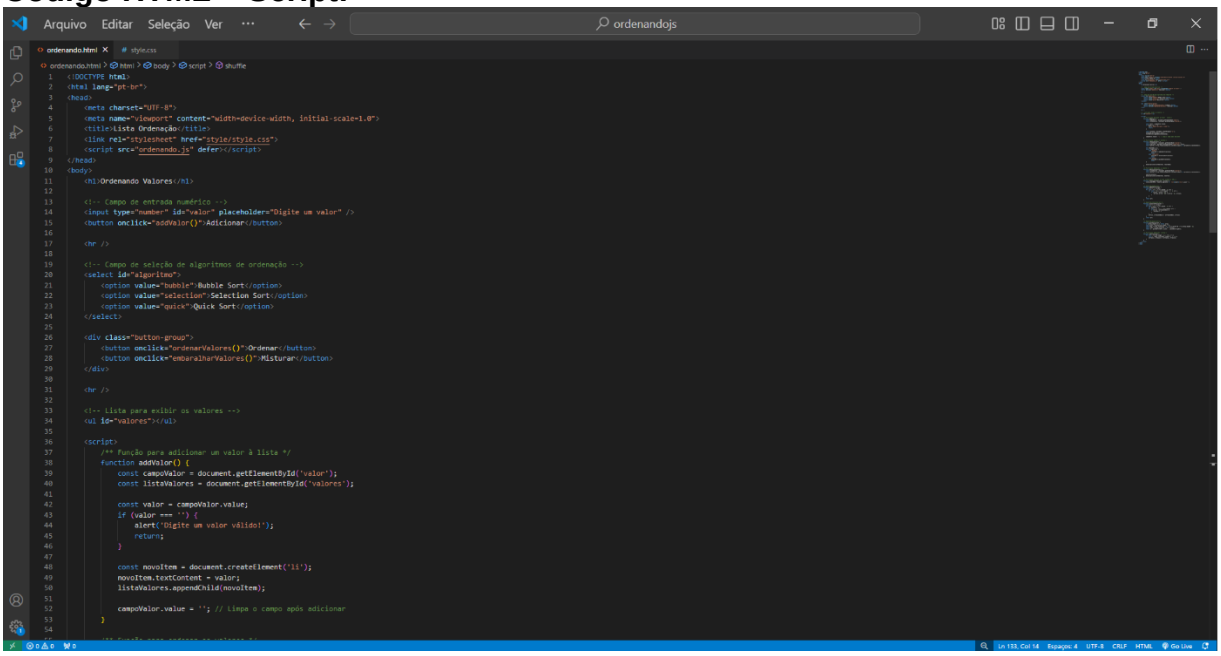
//Professor: Maria Manso

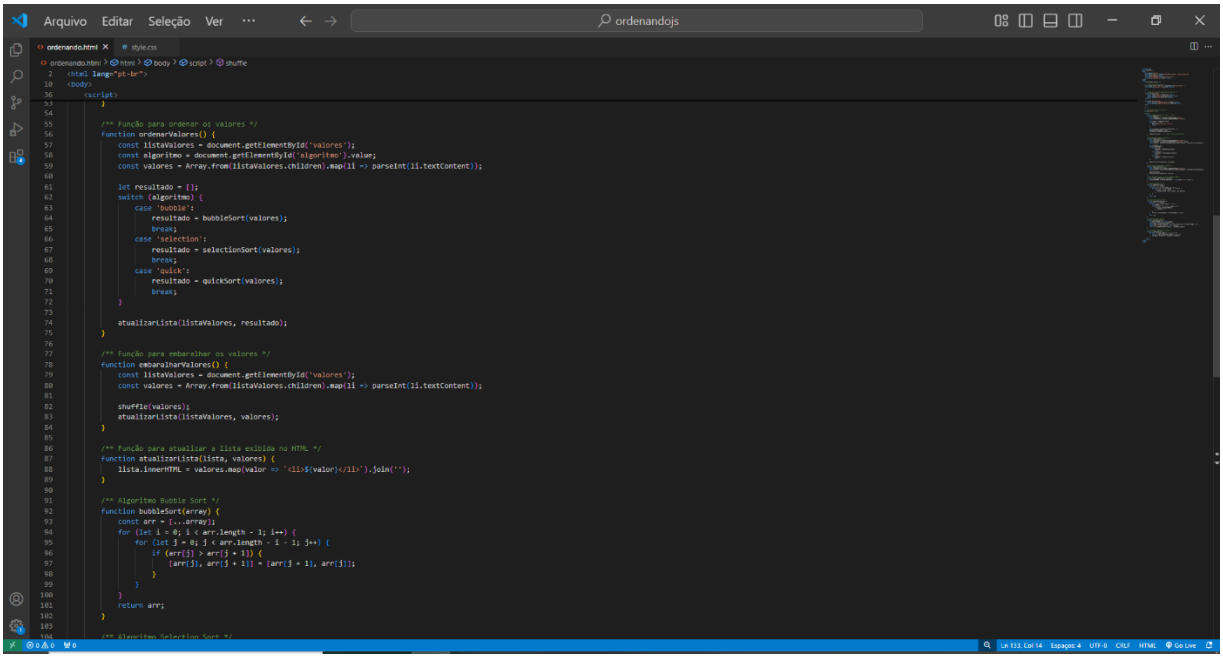
//Aluno: Rafael Lima de Medeiros

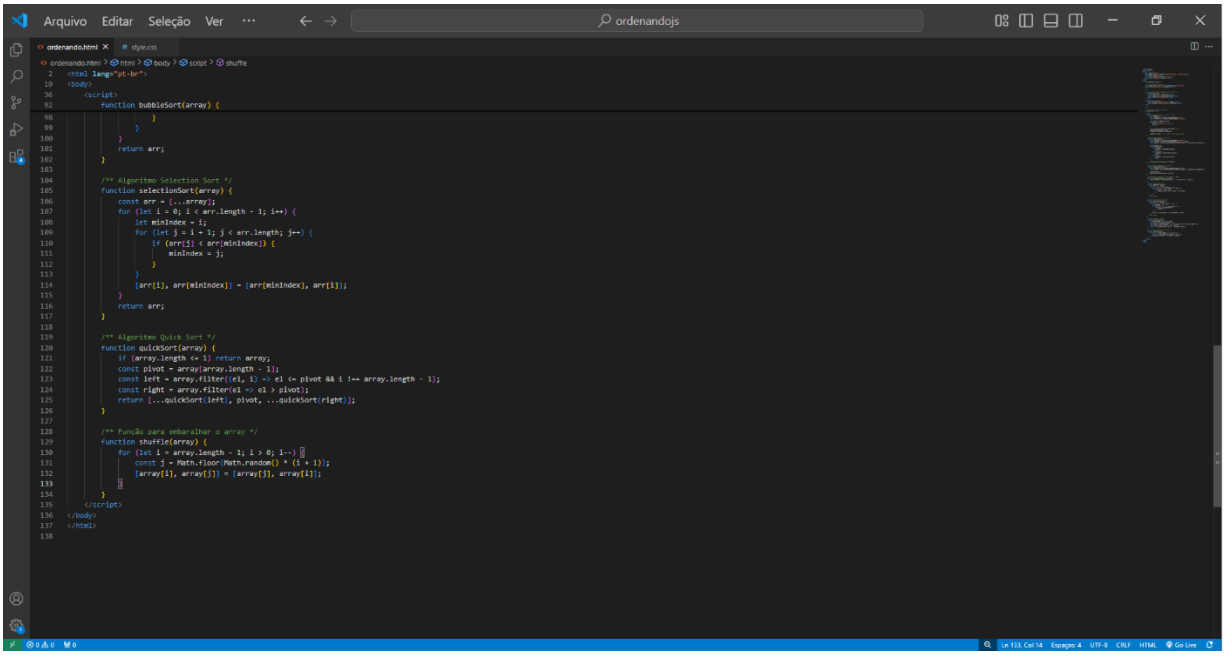
//Turma: 2024.2

//Data atual: 04/01/2025

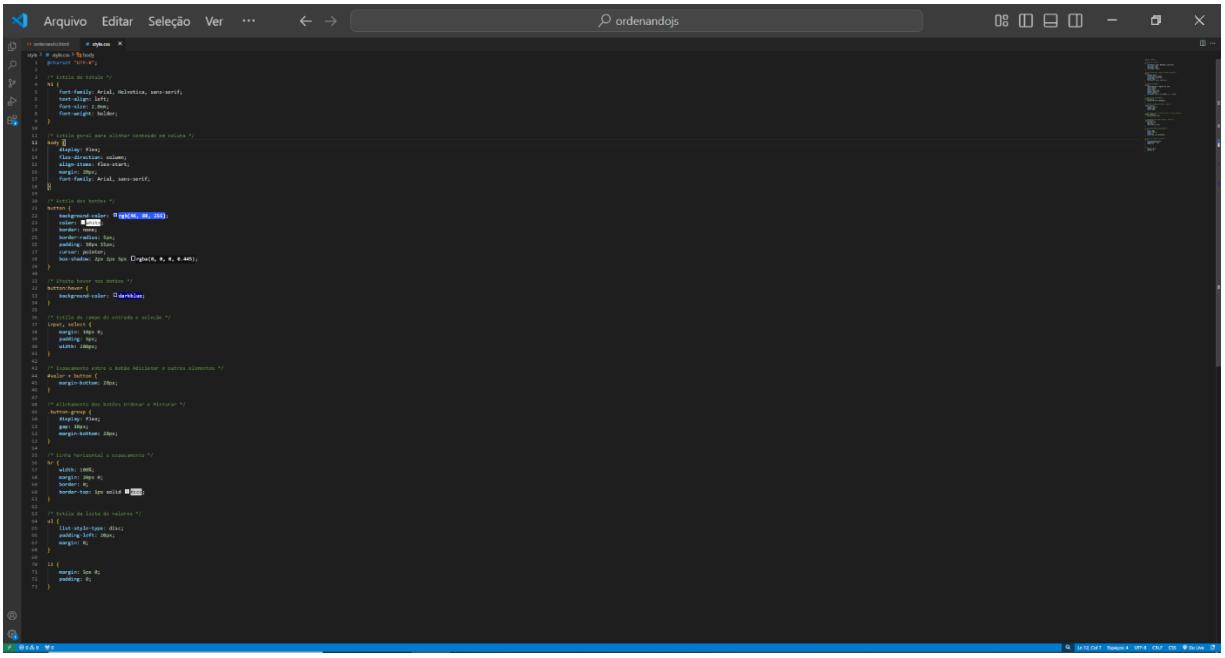
Código HTML + Script:



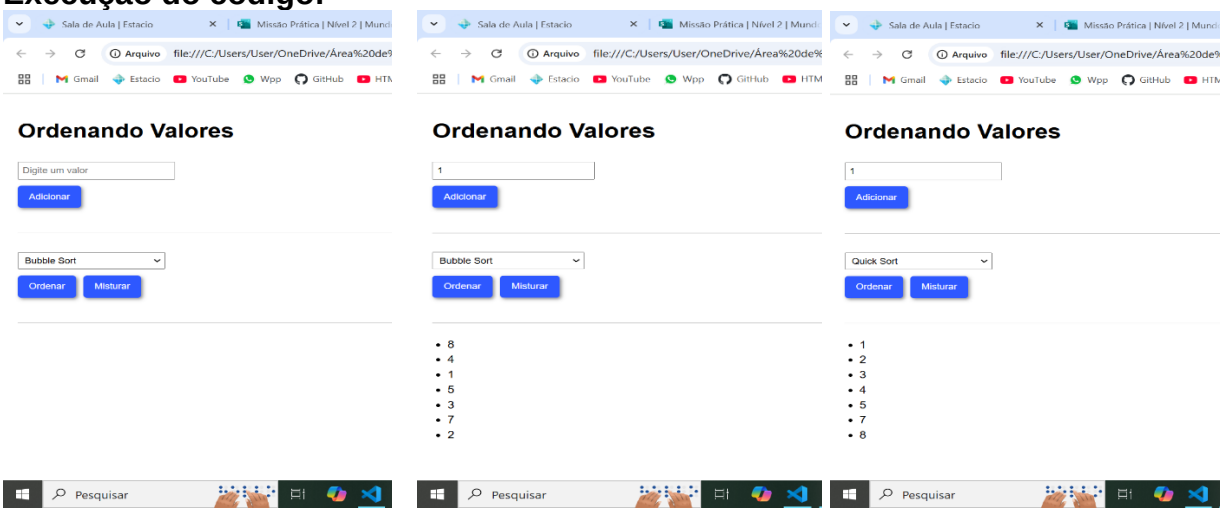




Estilo:



Execução do código:



Código digitado:

Ordenando.html

```
<!DOCTYPE html>
<html lang="pt-br">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Lista Ordenação</title>
    <link rel="stylesheet" href="style/style.css">
    <script src="ordenando.js" defer></script>
</head>
<body>
    <h1>Ordenando Valores</h1>

    <!-- Campo de entrada numérico -->
    <input type="number" id="valor" placeholder="Digite um valor" />
    <button onclick="addValor()">Adicionar</button>

    <hr />

    <!-- Campo de seleção de algoritmos de ordenação -->
    <select id="algoritmo">
        <option value="bubble">Bubble Sort</option>
        <option value="selection">Selection Sort</option>
        <option value="quick">Quick Sort</option>
    </select>

    <div class="button-group">
        <button onclick="ordenarValores()">Ordenar</button>
        <button onclick="embaralharValores()">Misturar</button>
    </div>

    <hr />

    <!-- Lista para exibir os valores -->
    <ul id="valores"></ul>

    <script>
        /** Função para adicionar um valor à lista */
        function addValor() {
            const campoValor = document.getElementById('valor');
            const listaValores = document.getElementById('valores');

            const valor = campoValor.value;
            if (valor === '') {
                alert('Digite um valor válido!');
                return;
            }

            const novoItem = document.createElement('li');
            novoItem.textContent = valor;
            listaValores.appendChild(novoItem);

            campoValor.value = ''; // Limpa o campo após adicionar
        }

        /** Função para ordenar os valores */
        function ordenarValores() {
            const listaValores = document.getElementById('valores');
            const algoritmo = document.getElementById('algoritmo').value;
            const valores = Array.from(listaValores.children).map(li => parseInt(li.textContent));

            let resultado = [];
            switch (algoritmo) {
                case 'bubble':
                    resultado = bubbleSort(valores);
                    break;
                case 'selection':
                    resultado = selectionSort(valores);
                    break;
                case 'quick':
                    resultado = quickSort(valores);
                    break;
            }

            atualizarLista(listaValores, resultado);
        }

        /** Função para embaralhar os valores */
```

```
    function embaralharValores() {
        const listaValores = document.getElementById('valores');
        const valores = Array.from(listaValores.children).map(li => parseInt(li.textContent));

        shuffle(valores);
        atualizarLista(listaValores, valores);
    }

    /** Função para atualizar a lista exibida no HTML */
    function atualizarLista(lista, valores) {
        lista.innerHTML = valores.map(valor => `<li>${valor}</li>`).join('');
    }

    /** Algoritmo Bubble Sort */
    function bubbleSort(array) {
        const arr = [...array];
        for (let i = 0; i < arr.length - 1; i++) {
            for (let j = 0; j < arr.length - i - 1; j++) {
                if (arr[j] > arr[j + 1]) {
                    [arr[j], arr[j + 1]] = [arr[j + 1], arr[j]];
                }
            }
        }
        return arr;
    }

    /** Algoritmo Selection Sort */
    function selectionSort(array) {
        const arr = [...array];
        for (let i = 0; i < arr.length - 1; i++) {
            let minIndex = i;
            for (let j = i + 1; j < arr.length; j++) {
                if (arr[j] < arr[minIndex]) {
                    minIndex = j;
                }
            }
            [arr[i], arr[minIndex]] = [arr[minIndex], arr[i]];
        }
        return arr;
    }

    /** Algoritmo Quick Sort */
    function quickSort(array) {
        if (array.length <= 1) return array;
        const pivot = array[array.length - 1];
        const left = array.filter((el, i) => el <= pivot && i !== array.length - 1);
        const right = array.filter(el => el > pivot);
        return [...quickSort(left), pivot, ...quickSort(right)];
    }

    /** Função para embaralhar o array */
    function shuffle(array) {
        for (let i = array.length - 1; i > 0; i--) {
            const j = Math.floor(Math.random() * (i + 1));
            [array[i], array[j]] = [array[j], array[i]];
        }
    }
  </script>
</body>
</html>
```

Style.css

```
@charset "UTF-8";

/* Estilo do título */
h1 {
    font-family: Arial, Helvetica, sans-serif;
    text-align: left;
    font-size: 2.0em;
    font-weight: bolder;
}

/* Estilo geral para alinhar conteúdo em coluna */
body {
    display: flex;
    flex-direction: column;
    align-items: flex-start;
    margin: 20px;
    font-family: Arial, sans-serif;
}

/* Estilo dos botões */
button {
    background-color: rgb(46, 88, 255);
    color: white;
    border: none;
    border-radius: 5px;
    padding: 10px 15px;
    cursor: pointer;
    box-shadow: 2px 2px 5px rgba(0, 0, 0, 0.445);
}

/* Efeito hover nos botões */
button:hover {
    background-color: darkblue;
}

/* Estilo do campo de entrada e seleção */
input, select {
    margin: 10px 0;
    padding: 5px;
    width: 200px;
}

/* Espaçamento entre o botão Adicionar e outros elementos */
#valor + button {
    margin-bottom: 20px;
}

/* Alinhamento dos botões Ordenar e Misturar */
.button-group {
    display: flex;
    gap: 10px;
    margin-bottom: 20px;
}

/* Linha horizontal e espaçamento */
hr {
    width: 100%;
    margin: 20px 0;
    border: 0;
    border-top: 1px solid #ccc;
}

/* Estilo da lista de valores */
ul {
    list-style-type: disc;
    padding-left: 20px;
    margin: 0;
}

li {
    margin: 5px 0;
    padding: 0;
}
```

Missão Prática - Nível 2

//Disciplina: Descobrindo o JavaScript – 2º Procedimento

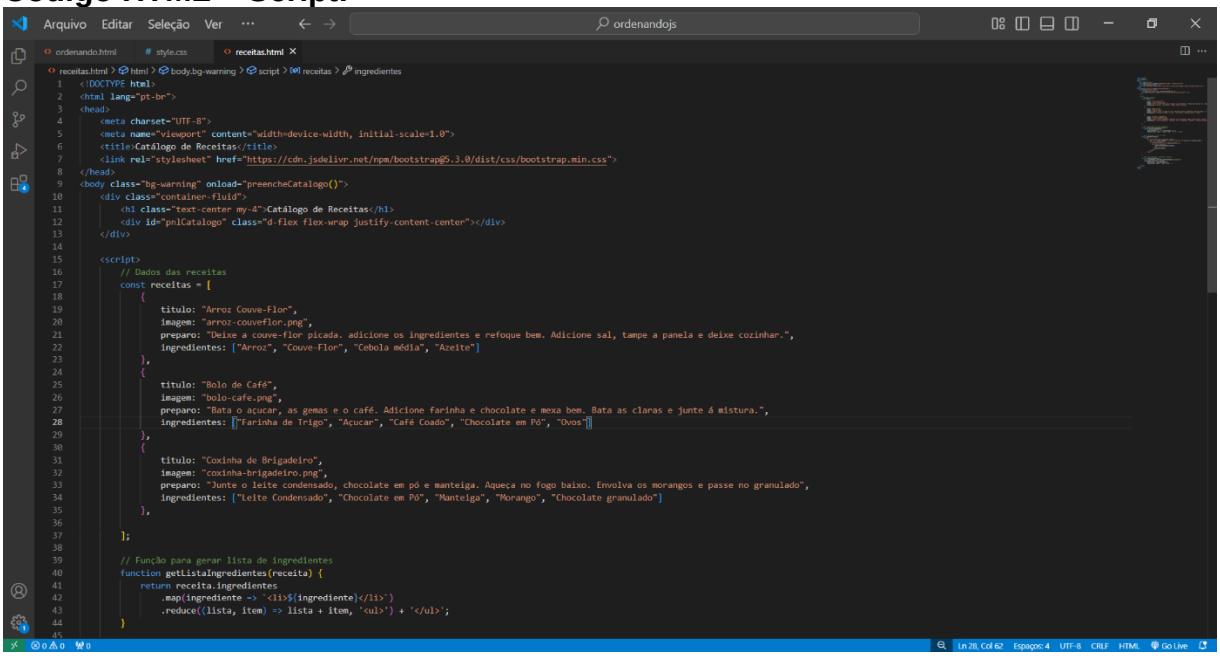
//Professor: Maria Manso

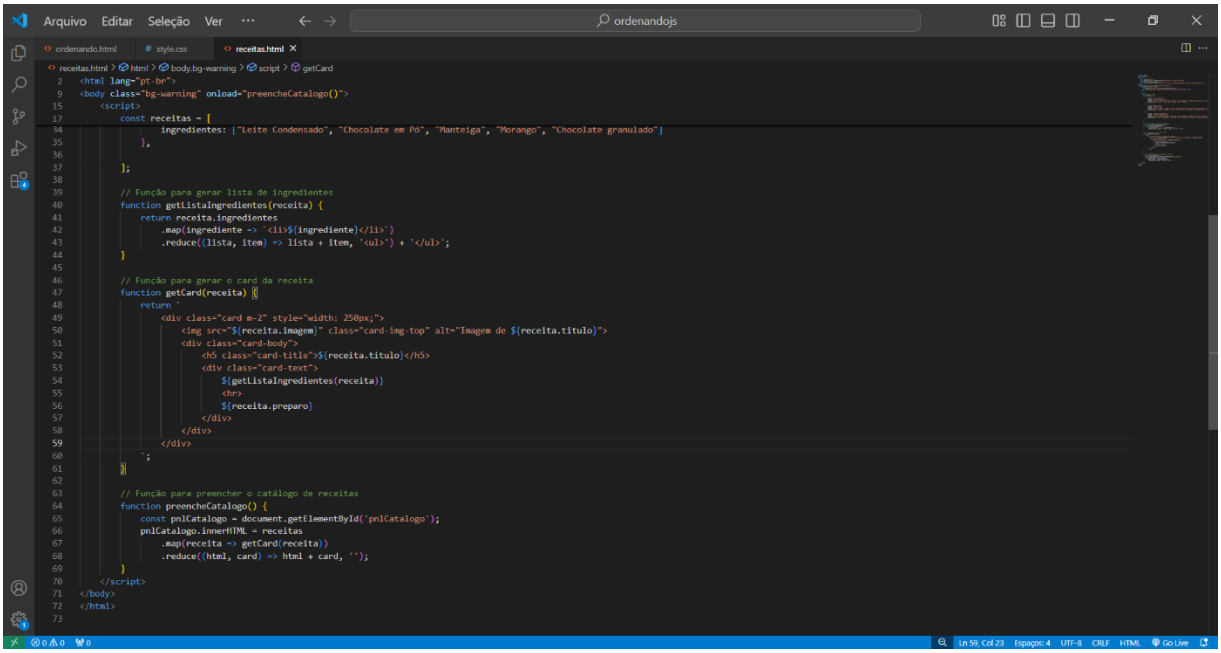
//Aluno: Rafael Lima de Medeiros

//Turma: 2024.2

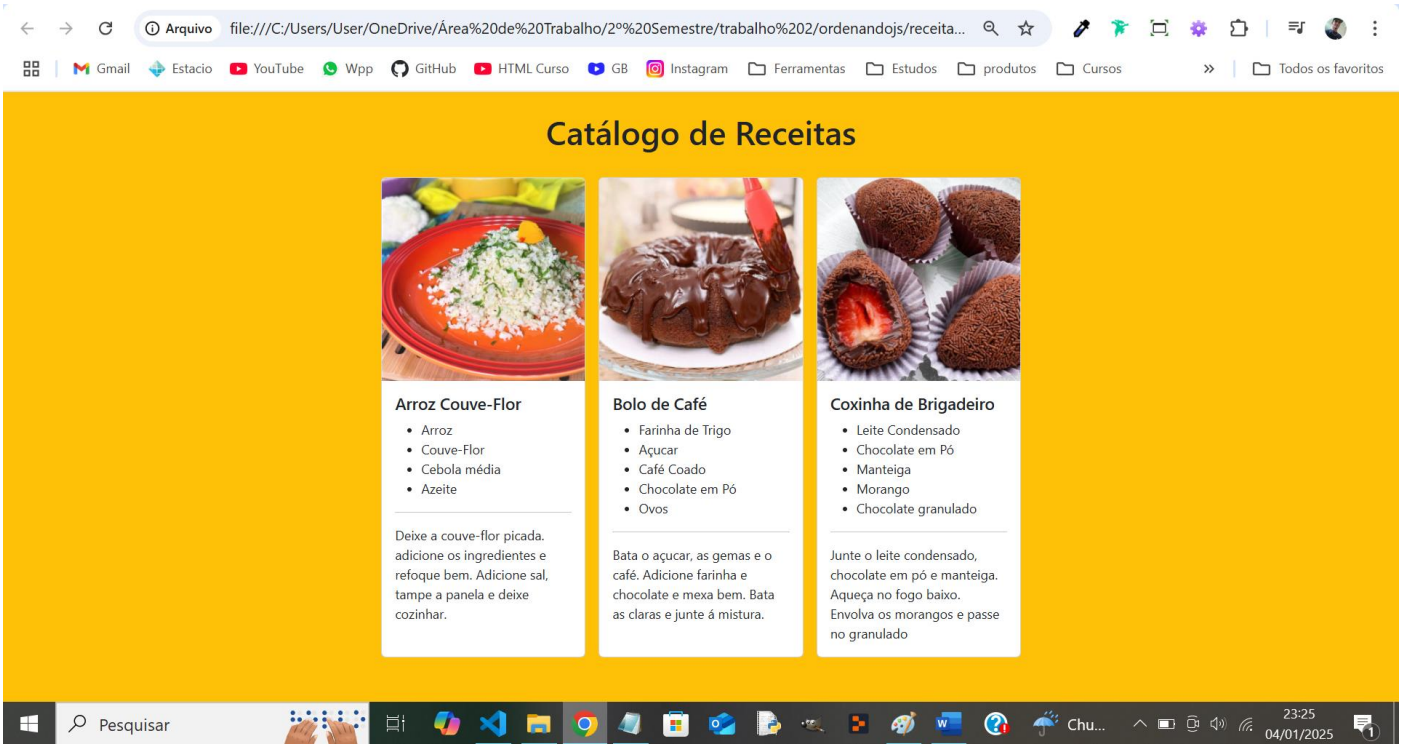
//Data atual: 04/01/2025

Código HTML + Script:





Execução do código:



Código digitado:

receitas.html

```
<!DOCTYPE html>
<html lang="pt-br">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Catálogo de Receitas</title>
    <link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/css/bootstrap.min.css">
</head>
<body class="bg-warning" onload="preencheCatalogo()">
    <div class="container-fluid">
        <h1 class="text-center my-4">Catálogo de Receitas</h1>
        <div id="pnlCatalogo" class="d-flex flex-wrap justify-content-center"></div>
    </div>

    <script>
        // Dados das receitas
        const receitas = [
            {
                titulo: "Arroz Couve-Flor",
                imagem: "arroz-couveflor.png",
                preparo: "Deixe a couve-flor picada. adicione os ingredientes e refoque bem. Adicione sal, tampe a panela e deixe cozinhar.",
                ingredientes: ["Arroz", "Couve-Flor", "Cebola média", "Azeite"]
            },
```

```
        {
            titulo: "Bolo de Café",
            imagem: "bolo-cafe.png",
            preparo: "Bata o açucar, as gemas e o café. Adicione farinha e chocolate e mexa bem. Bata as claras e junte á mistura.",
            ingredientes: ["Farinha de Trigo", "Açucar", "Café Coado", "Chocolate em Pó", "Ovos"]
        },
        {
            titulo: "Coxinha de Brigadeiro",
            imagem: "coxinha-brigadeiro.png",
            preparo: "Junte o leite condensado, chocolate em pó e manteiga. Aqueça no fogo baixo. Envolva os morangos e passe no granulado",
            ingredientes: ["Leite Condensado", "Chocolate em Pó", "Manteiga", "Morango", "Chocolate granulado"]
        },

    ];

    // Função para gerar lista de ingredientes
    function getListaIngredientes(receita) {
        return receita.ingredientes
            .map(ingrediente => `<li>${ingrediente}</li>`)
            .reduce((lista, item) => lista + item, '<ul>') + '</ul>';
    }

    // Função para gerar o card da receita
    function getCard(receita) {
        return `
            <div class="card m-2" style="width: 250px;">
                <img src="${receita.imagem}" class="card-img-top" alt="Imagem de ${receita.titulo}">
                <div class="card-body">
                    <h5 class="card-title">${receita.titulo}</h5>
                    <div class="card-text">
                        ${getListaIngredientes(receita)}
                        <hr>
                        ${receita.preparo}
                    </div>
                </div>
            </div>
        `;
    }

    // Função para preencher o catálogo de receitas
    function preencheCatalogo() {
        const pnlCatalogo = document.getElementById('pnlCatalogo');
        pnlCatalogo.innerHTML = receitas
            .map(receita => getCard(receita))
            .reduce((html, card) => html + card, '');
    }
  </script>
</body>
</html>
```

Missão Prática - Nível 2

//Disciplina: Descobrindo o JavaScript – 3º Procedimento

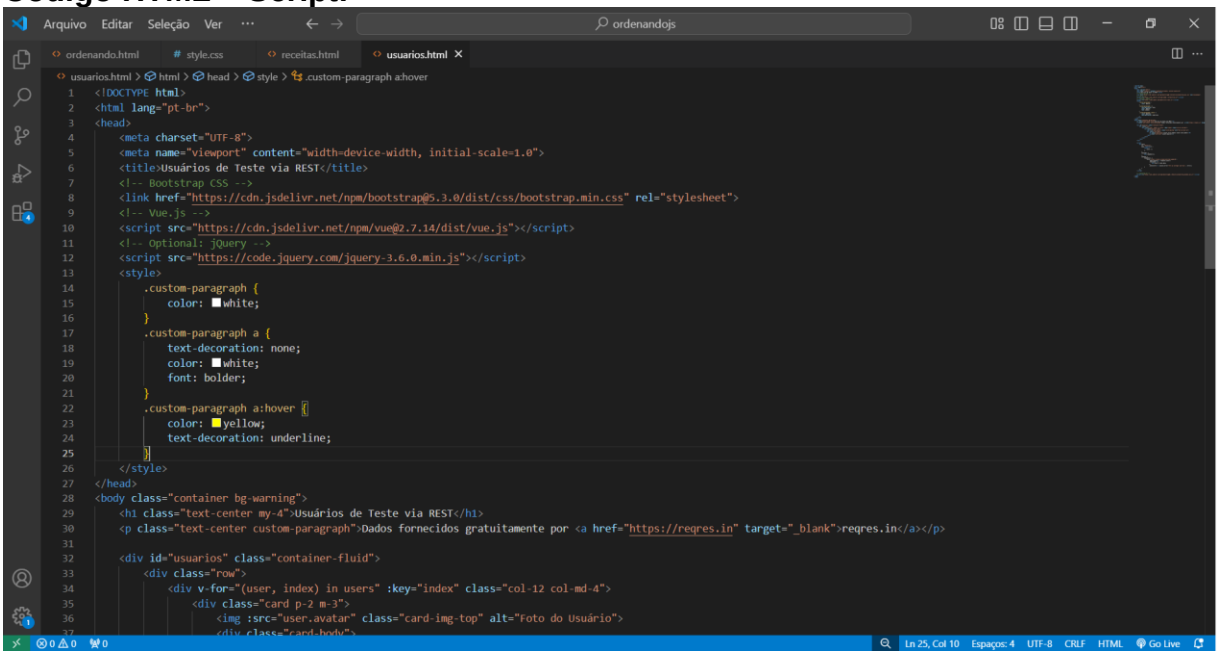
//Professor: Maria Manso

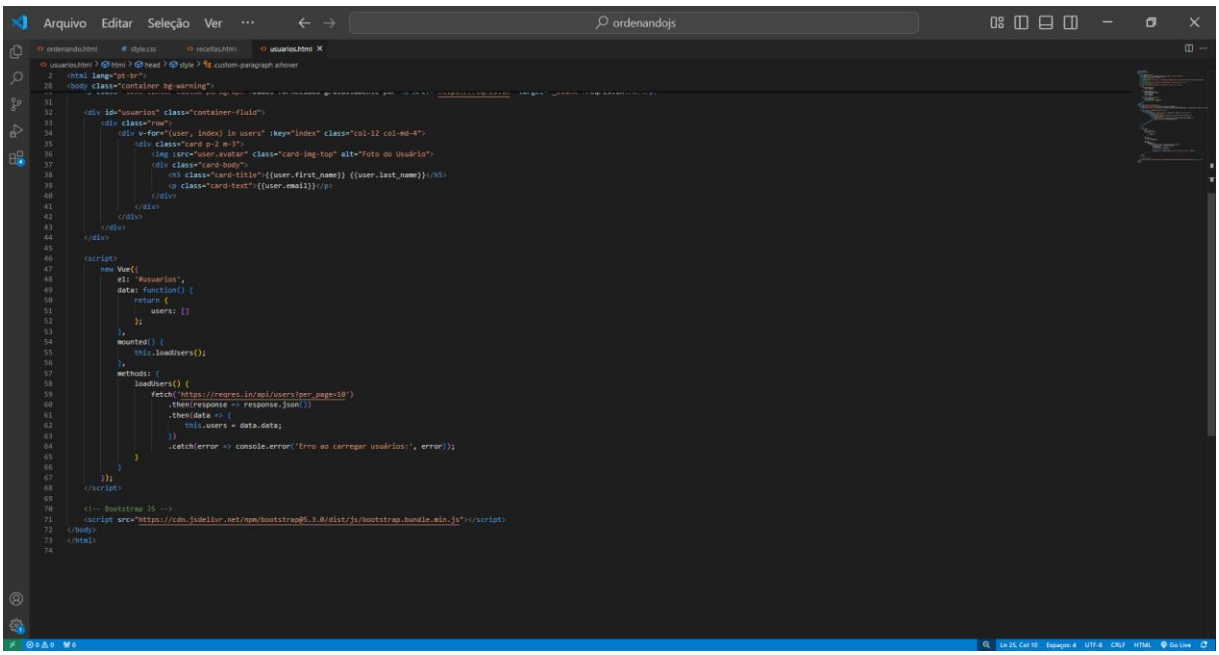
//Aluno: Rafael Lima de Medeiros

//Turma: 2024.2

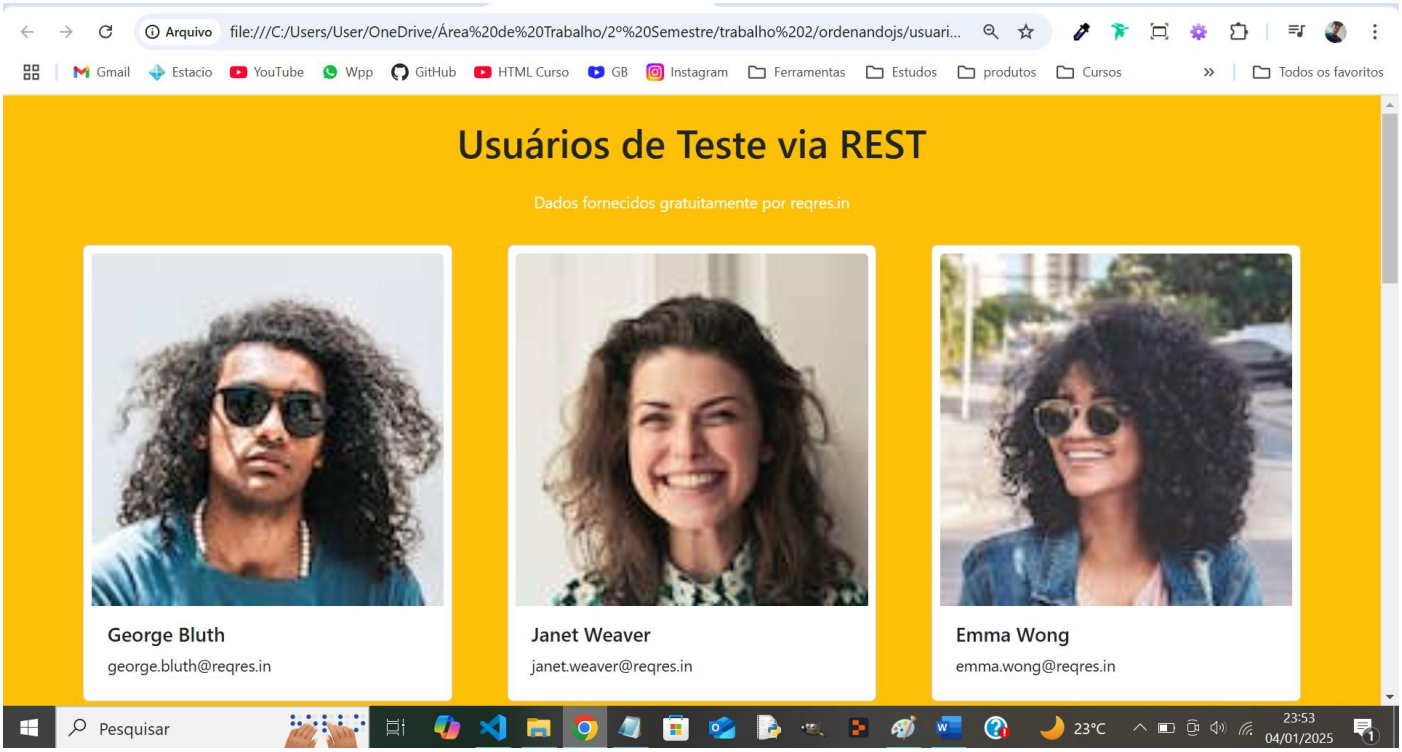
//Data atual: 04/01/2025

Código HTML + Script:





Execução do código:



Código digitado:

receitas.html

```
<!DOCTYPE html>
<html lang="pt-br">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Usuários de Teste via REST</title>
    <!-- Bootstrap CSS -->
    <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/css/bootstrap.min.css" rel="stylesheet">
    <!-- Vue.js -->
    <script src="https://cdn.jsdelivr.net/npm/vue@2.7.14/dist/vue.js"></script>
    <!-- Optional: jQuery -->
    <script src="https://code.jquery.com/jquery-3.6.0.min.js"></script>
    <style>
        .custom-paragraph {
            color: white;
        }
        .custom-paragraph a {
            text-decoration: none;
            color: white;
            font: bolder;
        }
        .custom-paragraph a:hover {
            color: yellow;
            text-decoration: underline;
        }
    </style>
</head>
<body class="container bg-warning">
    <h1 class="text-center my-4">Usuários de Teste via REST</h1>
```

```
    <p class="text-center custom-paragraph">Dados fornecidos gratuitamente por <a href="https://reqres.in" target="_blank">reqres.in</a></p>

    <div id="usuarios" class="container-fluid">
        <div class="row">
            <div v-for="(user, index) in users" :key="index" class="col-12 col-md-4">
                <div class="card p-2 m-3">
                    <img :src="user.avatar" class="card-img-top" alt="Foto do Usuário">
                    <div class="card-body">
                        <h5 class="card-title">{{user.first_name}} {{user.last_name}}</h5>
                        <p class="card-text">{{user.email}}</p>
                    </div>
                </div>
            </div>
        </div>
    </div>

    <script>
        new Vue({
            el: '#usuarios',
            data: function() {
                return {
                    users: []
                };
            },
            mounted() {
                this.loadUsers();
            },
            methods: {
                loadUsers() {
                    fetch('https://reqres.in/api/users?per_page=10')
                        .then(response => response.json())
                        .then(data => {
                            this.users = data.data;
                        })
                        .catch(error => console.error('Erro ao carregar usuários:', error));
                }
            }
        });
    </script>

    <!-- Bootstrap JS -->
    <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/js/bootstrap.bundle.min.js"></script>
</body>
</html>
```