



FULL DIGITAL PERFORMANCE

DOCUMENTAÇÃO DO PROJETO CLASSIFICATÓRIO

AUTOR: AUGUSTO ESTEVÃO MONTE

RESUMO

Em primeira análise, esta documentação irá abordar o projeto de software proposto pela Rocky Full Digital performance.

O problema do projeto é consertar uma implementação feita no banco de dados buscando corrigir alguns caracteres trocados do mesmo, sendo eles “a por æ, c por ç, o por ø, b por ß”. O banco de dados usado é um NoSQL, “um banco de dados não relacional que permite maior velocidade, flexibilidade e escalabilidade ao armazenar e acessar dados não estruturados” (CÓDIGO FONTE TV, 2018).

Todavia, as informações a serem corrigidas estão armazenadas em um objeto em formato JSON para corrigir estes dados será utilizada a linguagem Javascript.

Palavras-chave: NoSQL, objeto JSON, Javascript, consertar.


1 INTRODUÇÃO

Em primeiro lugar, o trabalho a seguir irá apresentar uma a solução para o problema apresentado no teste prático da Rocky.

Antes de mais nada é necessário abordar o que seria um objeto JSON “JavaScript Object Notation (JSON) é um formato baseado em texto padrão para representar dados estruturados com base na sintaxe do objeto JavaScript” (MOZILLA, 2021) com essa definição em mente a linguagem mais adequada para trabalhar com essa extensão de arquivo seria o Javascript já que o mesmo contém uma série de funções utilitárias para o desenvolvimento rápido usando este arquivo.

1.1 FUNCIONALIDADES

O código abaixo faz uma requisição no sistema de arquivos do Node, usando a constante “fs” usaremos ela para ler os arquivos.



```
const fs = require('fs');
```

O código abaixo lê o arquivo que contém os dados alterados e os constrói armazenando na constante chamada “*json*”, e em seguida retorna a constante.

```
function readCorruptedJson() {  
  const data = fs.readFileSync('broken-database.json');  
  const json = JSON.parse(data);  
  return json;  
}
```

A seguinte função “*fixAllJsonNames*” recebe por parâmetro o objeto JSON que foi lido do arquivo “*broken-database.json*” acima, e em seguida procura no mesmo todas as ocorrências de caracteres inválidos como “*æ,ç,ø,ß*” e os substitui respectivamente por “*a,c,o,b*” os caracteres corretos e em seguida retorna o objeto JSON com os valores corrigidos .

```
function fixAllJsonNames(json) {  
  for (let i = 0; i < json.length; i++) {  
    json[i].name = json[i].name.replace(/ø/g, 'o');  
    json[i].name = json[i].name.replace(/ç/g, 'c');  
    json[i].name = json[i].name.replace(/æ/g, 'a');  
    json[i].name = json[i].name.replace(/ß/g, 'b');  
  }  
  return json;  
}
```

A seguinte função transforma todos os valores de *json.price* para float, pois no arquivo algumas partes estão em string.

```
function fixAllJsonPrices(json) {  
  for (let i = 0; i < json.length; i++) {  
    json[i].price = parseFloat(json[i].price);  
  }  
  return json;  
}
```

Corrige a quantidade de produtos em todos os campos do objeto JSON, pois alguns produtos estavam sem “*quantity*”, aqui também foi adicionado uma validação para que não fosse atribuído NULL a “*quantity*”

```
function fixAllJsonQuantity(json) {  
  for (let i = 0; i < json.length; i++) {  
    if (json[i].quantity == null) {  
      json[i].quantity = 0;  
    }  
    json[i].quantity = parseInt(json[i].quantity);  
  }  
  return json;  
}
```

Faz as 3 correções no objeto JSON, troca os caracteres inválidos nos nomes, transforma os preços em float, e coloca *quantity* em objetos sem esse atributo.

```
function fixJson(json) {  
  json = fixAllJsonNames(json);  
  json = fixAllJsonPrices(json);  
  json = fixAllJsonQuantity(json);  
  return json;  
}
```

Escreve no arquivo “saida.json” todo o JSON corrigido.

```
function writeJson(json) {  
  fs.writeFileSync('saida.json', JSON.stringify(json));  
}
```

Ordena todos os nomes do objeto JSON em ordem alfabética, sendo primeiro as frases convertidas para letras minúsculas e em seguida é comparado pela função *sort* se os elementos são iguais ou diferentes.

```
function orderJsonInAlphabetical(json) {  
  json.sort(function (a, b) {  
    if (a.name.toLowerCase() < b.name.toLowerCase()) {  
      return -1;  
    }  
    if (a.name.toLowerCase() > b.name.toLowerCase()) {  
      return 1;  
    }  
    return 0;  
  });  
  return json;  
}
```

Ordena o objeto JSON por ID de forma crescente, segue a mesma lógica da ordenação feita em ordem alfabética só que com números.

```
function orderJsonByID(json) {  
  json.sort(function (a, b) {  
    return a.id - b.id;  
  });  
  return json;  
}
```

Imprime no console do desenvolvedor todo o objeto JSON que passou pelas 2 ordenações, sendo elas: Alfabéticas e numéricas.

```
function printAllOrderedJson(orderJson) {  
  for (let i = 0; i < orderJson.length; i++) {  
    console.log(orderJson[i]);  
  }  
}
```

Essa função calcula o valor total dos itens por categoria multiplicando a quantidade do item pelo preço, e em seguida adicionando em um Array chamado valorTotal.

```
function totalValueInInventory(json) {  
  const valorTotal = [];  
  for (let i = 0; i < json.length; i++) {  
    let valor = json[i].quantity * json[i].price;  
    valorTotal.push(valor);  
  }  
  return valorTotal;  
}
```


Neste ponto, todo o código é colocado dentro de uma única função para ser executado de maneira mais organizada.

```
function main() {  
  const json = readCorruptedJson();  
  let totalPrices = new Array();  
  const fixedJson = fixJson(json);  
  writeJson(fixedJson);  
  const orderJson = orderJsonById(orderJsonInAlphabetical(fixedJson));  
  printAllOrderedJson(orderJson);  
  totalPrices = totalValueInInventory(fixedJson);  
}
```

Em seguida a função `main()` é chamada executando todos os métodos acima, e gerando o objeto JSON corrigido chamado de *"saida.json"*

```
main();
```

Já no arquivo “saida.json” teremos o seguinte objeto JSON corrigido e ordenado pelo ID.

```
[
  {
    "id": 1316334,
    "name": "Refrigerador bottom Freezer Electrolux de 02 Portas Frost Free com 598 Litros",
    "quantity": 12,
    "price": 3880.23,
    "category": "Eletrodomésticos"
  },
  {
    "id": 1911864,
    "name": "Mouse Gamer Predator cestus 510 Fox Preto",
    "price": 699,
    "category": "Acessórios",
    "quantity": 0
  },
  {
    "id": 2162952,
    "name": "Kit Gamer acer - Notebook + Headset + Mouse",
    "price": 25599,
    "category": "Eletrônicos",
    "quantity": 0
  },
  {
    "id": 3500957,
    "name": "Monitor 29 LG FHD Ultrawide com 1000:1 de contraste",
    "quantity": 18,
    "price": 1559.4,
    "category": "Eletrônicos"
  },
  {
    "id": 5677240,
    "name": "Conjunto de Panelas antiaderentes com 05 Peças Paris",
    "quantity": 21,
    "price": 192.84,
    "category": "Panelas"
  },
  {
    "id": 6502394,
    "name": "Fogão de Piso Electrolux de 04 bocas, Mesa de Vidro Prata",
    "quantity": 37,
    "price": 1419,
    "category": "Eletrodomésticos"
  },
  {
    "id": 8875900,
    "name": "Smart TV 4K Sony LED 65\" 4K X-Reality Pro, UpScalling, Motionflow XR 240 e Wi-F",
    "quantity": 0,
    "price": 5799.42,
    "category": "Eletrônicos"
  },
  {
    "id": 9576720,
    "name": "Forno Micro-ondas Panasonic com capacidade de 21 Litros branco",
    "quantity": 13,
    "price": 358.77,
    "category": "Eletrodomésticos"
  },
  {
    "id": 9628920,
    "name": "Lava & Seca 10,2 Kg Samsung Eco bubble branca com 09 Programas de Lavagem",
    "quantity": 57,
    "price": 3719.7,
    "category": "Eletrodomésticos"
  },
  {
    "id": 9746439,
    "name": "Home Theater LG com blu-ray 3D, 5.1 canais e 1000W",
    "quantity": 80,
    "price": 2199,
    "category": "Eletrônicos"
  }
]
```

REFERÊNCIAS

Mozilla Firefox 19 de agosto de 2021. Disponível em:

<<https://developer.mozilla.org/pt-BR/docs/Learn/JavaScript/Objects/JSON>>. Acesso em 07 de Abril de 2022.