PUCRS/DELL-Reliability 2022

HENRIQUE FEIJÓ PAIM

Relatório do processo seletivo Reliability 2022

PORTO ALEGRE

2022

Enunciado proposto

O desafio proposto consiste em desenvolver uma aplicação para servir a um depósito de mercado sobre a administração de itens, de forma que seja possível realizar 7 atividades obrigatórias:

- incluir novos itens ao estoque.
- excluir itens do estoque.
- alterar itens do estoque.
- consultar itens.
- consultar item pelo setor.
- consultar item por prateleira.
- consultar prateleira por setor.

O desafio também inclui mais 4 atividades não obrigatórias:

- Não permitir que o peso máximo da prateleira seja atingido.
- Não permitir que o volume máximo da prateleira seja atingido.
- Alocar um novo item automaticamente na prateleira, atentando as capacidades. máximas dela.
- Front-end que permite realizar as atividades acima citadas.

O enunciado estabelece as propriedades obrigatórias que cada item deve ter: nome, tipo, peso, volume, quantidade, setor e localização na prateleira. Além das propriedades obrigatórias de cada prateleira: setor, capacidade máxima de peso e capacidade máxima de volume.

Ao compreender todos os requisitos, nota-se que é procurado que o participante tenha habilidade em manipular estruturas de dados e realizar as operações básicas nos mesmos, como criar, ler, atualizar e deletar. Nota-se também que está sendo requisitados conhecimentos em programação orientada a objetos, visto que envolve a manipulação de objetos e suas propriedades que neste caso seriam :setor, nome, tipo etc.

O enunciado não especifica amplamente como funciona a organização do depósito, ficando a cargo do desenvolvedor definir a melhor forma para categorizar os itens e prateleiras de acordo com suas propriedades, sendo apenas destacado que sejam agrupados preferencialmente itens de um mesmo tipo na prateleira em questão.

Ambiente utilizado

O ambiente utilizado para o desenvolvimento completo da solução foi o Visual Studio Code utilizando a linguagem de programação Java.

Foram desenvolvidos alguns protótipos de tela utilizando React, Route 6 e BootStrap, a qual será explicado ao longo deste relatório o porquê de não terem sido utilizados.

A solução final apresenta as informações e interações por terminal utilizando de comandos simples de entrada e saída, atentando aos possíveis erros de entradas inválidas e visando experiência de usuário para que seja possível reparar os erros. Foi utilizado de princípios de programação orientada a objetos e de encapsulamento para realizar as atividades solicitadas de forma mais prática e legível possível.

Lógica desenvolvida

A solução final deste programa utiliza-se da manipulação de duas classes principais: Prateleira e Item, foi definido como identificador único de cada classe a propriedade *nome* e *nomePrateleira* delas, a qual sendo este o argumento requisitado durante os processos de alteração/exclusão de itens.

A concepção de depósito tomada para esta implementação se baseia nos seguintes tópicos:

- Cada prateleira possui um nome que é composto por um número e uma letra, o número representa a enumeração física da prateleira, enquanto a letra subsequente diz respeito a quais tipos de produtos são postos nesta prateleira, as letras existentes são:

  - C (Carnes)
  - P (Peixes)
  - B (Bebidas)
  - I (Congelados)
  - E (Enlatados)
  - F (Frios)
  - H (Higiene)
  - L (Limpeza)
  - S (Salgadinhos)
  - O (Outros)

- Cada prateleira possui um conjunto de itens
- As medidas de peso e volume foram adotadas na unidade de medida de kg para peso e m^3 para volume
- Foi tomado como localização na prateleira o andar em que este item se encontra na prateleira, como: 1-andar,2-andar...
- Os valores para tipo foram definidos como :carnes, peixes, bebidas, congelados, enlatados, frios, higiene, limpeza, salgadinhos, outros.

Os métodos de alteração e exclusão necessitam que seja manipulado apenas um Item por vez por questão de integridade, neste desafio proposto não foi definido qual o identificador único para cada Item ou prateleira, sendo então tomado por escolha do autor o atributo *nome* da classe *item* e *nomePrateleira* da classe *Prateleira*

A operação de incluir item primeiramente consiste em obter do usuário os dados para popular um objeto do tipo *Item* e, caso algum destes dados estejam em um formato errado, é mostrado em tela uma mensagem de erro e o usuário pode

repetir o processo de inclusão, desta forma é permitido que o usuário corrija seu erro sem ter que reiniciar o programa, visando uma melhor experiencia de usuário. Após obter os dados é validado a propriedade *tipo* e com base nesta é enviado para uma prateleira em específico, caso não for encontrada é mostrado em tela uma mensagem de erro.

A relação entre o tipo do item e a prateleira a qual deve receber este tipo foram definidos através da opinião do autor com base em raciocínio lógico obtido observando elementos do dia a dia dele.

A cada processo de inclusão e remoção realizado, é validado se as capacidades físicas da prateleira não foram alcançadas, além de calcular o quanto de peso e volume serão acrescidos sobre a prateleira após a adição, contando a quantidade em que estes itens estarão presentes na prateleira.

A operação de alteração foi desenvolvida de forma que através do identificador único seja obtido o item a ser alterado, e então é criado outro objeto do tipo Item, este novo objeto é adicionado e então após este processo, o item anterior é deletado.

Este processo foi planejado visando que ao mudar o tipo do item, este pode ocorrer de ser remanejado para outra prateleira, então preferiu-se utilizar das estruturas já desenvolvidas com os tratamentos já aplicados, a criar estruturas resultando em um código menos limpo e mais sujeito a erros e manutenções.

Para que o procedimento acima citado ocorra sem erros, é necessário primeiro encontrar o item pelo identificador, adicionar o que seria a sua versão atualizada e após a confirmação da inclusão, deletar o item antigo, tendo estas medidas de controle evita-se que seja afetado a integridade dos dados que estão sendo manipulados.

A operação de consultar itens foi definida como listar todos os itens existentes em todas as prateleiras. A de consultar por setor verifica todas as prateleiras e os itens que tiverem setor igual ao fornecido do teclado, são mostrados em tela. já a consulta de itens por prateleira capta do usuário o nome da prateleira (seu identificador único):como ‘1A,2C’ e imprime em tela todos os itens que estão nesta prateleira.

A atividade de consultar prateleira por setor segue lógica semelhante a citada acima, porém é filtrado pelo setor das prateleiras em vez de pelo setor dos itens e então impresso estas prateleiras em tela.

Implementação na linguagem Java

Como apresentado anteriormente, a solução foi implementada utilizando de programação orientada a objetos, tendo sido criado as classes *Item* e *Prateleira*, ambas as classes possuem as propriedades especificadas no enunciado deste desafio, todas estas propriedades foram declaradas nas classes e com o acréscimo dos campos *nomePrateleira, itens,pesoOcupado e volumeOcupado* na classe *Prateleira*. ambas as classes possuem os métodos *construtores*, *getters* e *setters* padrões além de métodos auxiliares para obter ou converter elementos.

A classe Item possui um método denominado *imprimeItem* cuja função é imprimir as propriedades da classe em tela, também contém um *setter* que converte

um inteiro para o valor correspondente ao tipo do item, previamente será demonstrado essa questão.

A classe Prateleira é a classe mais complexa fora a classe *App,* nesta classe ocorre o controle se um item pode ser adicionado ou não, atentando-se aos limites de peso e volume, onde a cada inclusão é verificado se é possível inserir um item ou não e se for possível, incrementar os valores de peso e volume ocupados e a cada remoção é decrementado estes valores, atentando-se ao fato da quantidade do item sendo tratado em questão.

A classe app é a responsável por apresentar o menu ao usuário e listar as opções em loop até que o usuário deseje parar de consultar e digite 0, além de realizar estas operações. o papel do banco de dados é feito utilizando um arquivo no formato CSV delimitado por ponto e vírgula, de nome “Estoque”, este arquivo é lido antes de iniciar o loop e posteriormente é sobrescrito com os novos valores após todos os processos terminarem, os dados que preenchem este arquivo são as propriedades de todos os itens de todas as prateleiras, de forma que ao serem lidos eles são adicionados as devidas prateleiras também seguindo os processos previamente apresentados.

Os objetos da classe *Prateleira* foram instanciados diretamente no método main, visto que estes objetos não precisam passar pelos processos de inclusão, alteração e exclusão.

O autor deste relatório optou por deixar instanciado uma prateleira para cada tipo respectivo, mas não significando que o programa não funcione com mais prateleiras para um mesmo tipo, caso testado.

Após ser instanciado as prateleiras é lido o arquivo e atribuído os itens as prateleiras especificas.

Por se tratar de um programa desenvolvido por terminal, foi percebível a necessidade de controlar os valores fornecidos para o tipo do item, pois o mesmo é utilizado para distribuir os itens as prateleiras devidas e este processo não pode estar sujeito a imprevisibilidade vindas da entrada do usuário, sendo para isto criado um *Enum* denominado Tipo e apresentado um submenu com as enumerações possíveis de serem escolhidas pelo usuário, demais campos preferiu-se não desenvolver esta estrutura para evitar ter um código muito poluído.

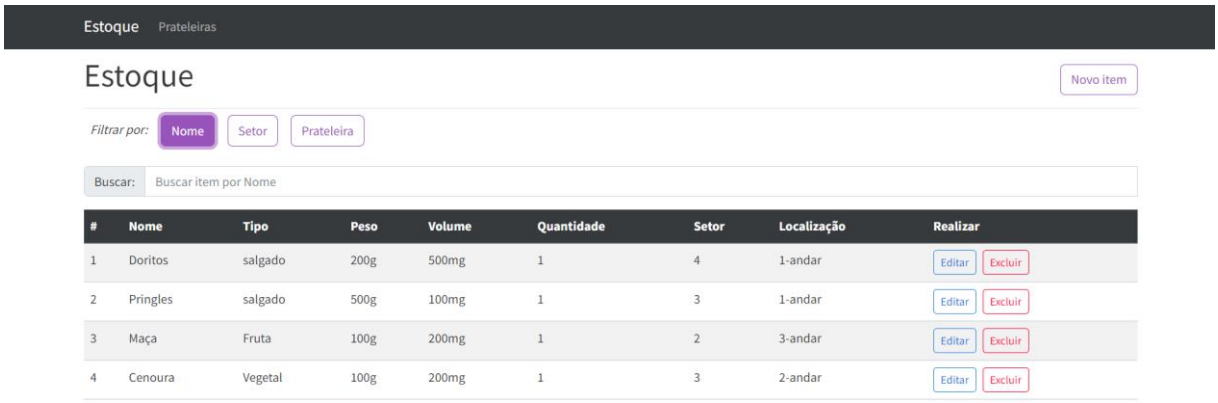
Caso o arquivo não possa ser lido ou futuramente não possa ser gravado, é apresentado mensagens de erro ao usuário

A classe App apresente o método *popularCampo* que é responsável por capturar e tratar os dados vindos da entrada do usuário, atentando-se a casos como valores inválidos para peso e volume, tipos inexistentes etc.

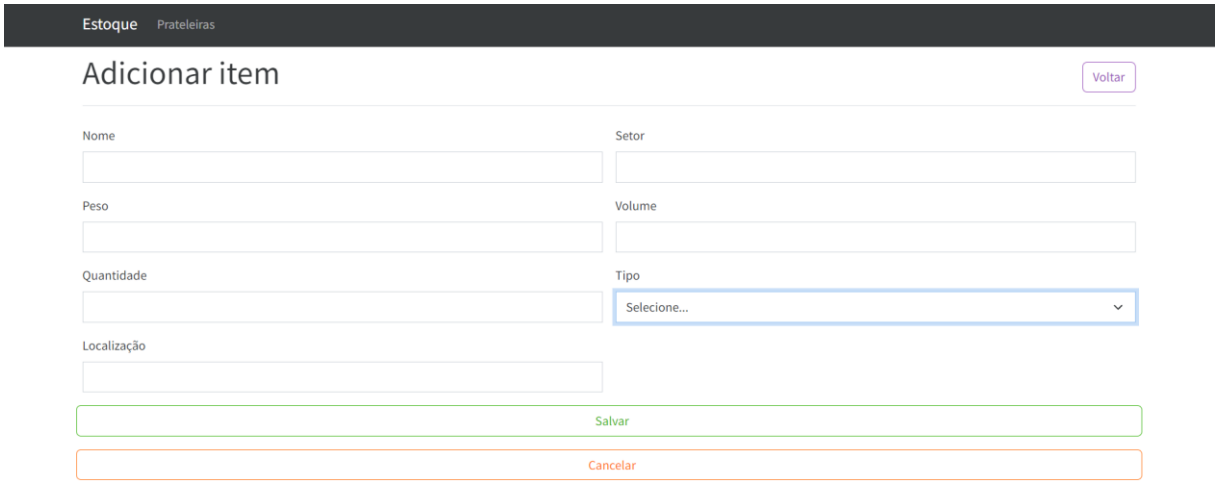
Este método é o responsável por validar o que seriam as chamadas regras de negócio, foi implementado como único método para que possa ser reaproveitado em outras ocasiões.

Front-end

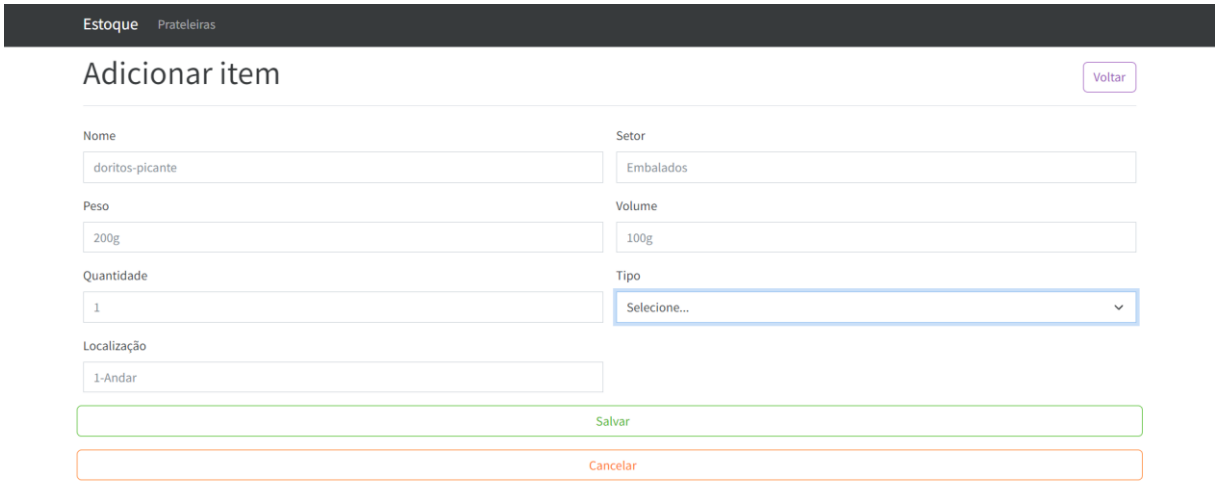
O projeto inicial para o desenvolvimento deste desafio era uma aplicação full stack sobre camadas utilizando React.js e BootStrap para o front-end, mas por questões de saúde por parte do autor que ficou incapacitado por alguns dias de atuar no desafio e a falta de fluência em desenvolver aplicações full-stack, foi optado por desenvolver o programa via terminal, mas contendo alguns protótipos de telas, abaixo esta o que seria a tela para consultar os itens:



O formulário para adicionar um item teria o seguinte aspecto:



Ao clicar em “Editar” na tela de consulta, seria apresentado a seguinte tela com os dados preenchidos como exemplo:



Dificuldades encontradas

A maior dificuldade encontrada foi quanto as limitações ao utilizar terminal como interface gráfica, a qual a falta de intuitividade dificulta que seja transmitido de forma clara as devidas informações e que seja obtido os valores devidamente formatados.

A outra dificuldade encontrada foi em abstrair um modelo de depósito de mercado, vindo de uma falta de experiência nestes aspectos por parte do autor.

De demais casos, o desenvolvimento ocorreu de forma fluida e sem muitos percalços, visto que o desafio proposto consiste em exercer conceitos básicos da programação como “CRUD” e orientação a objetos.

Testes desenvolvidos

Os testes desenvolvidos foram todos feitos de forma manual, inicialmente atribuindo os valores diretamente na classe App e posteriormente utilizando de leitura de arquivo CSV, foram testados entradas inválidas e casos em que ocorrem erros de exceções na entrada de dados.

A maioria dos cenários de erros possíveis ocorrem no método *PopulaCampos,* a qual pode ocorrer exceções de tipos de entrada inválidos ou entrada fora dos padrões definidos.

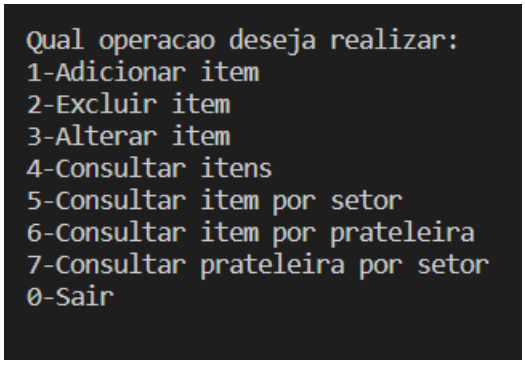
No caso de alguma entrada invalida, como tipos foras do limite definido ou valores de peso e volumes menores ou igual a zero, é impresso a seguinte mensagem:” A entrada nao corresponde com os limites possiveis de valores para o campo selecionado”.

Caso ocorra algum erro de formato de entrada, cai em um catch e apresenta a seguinte mensagem: “entrada inválida, tente novamente com outro tipo de entrada”.

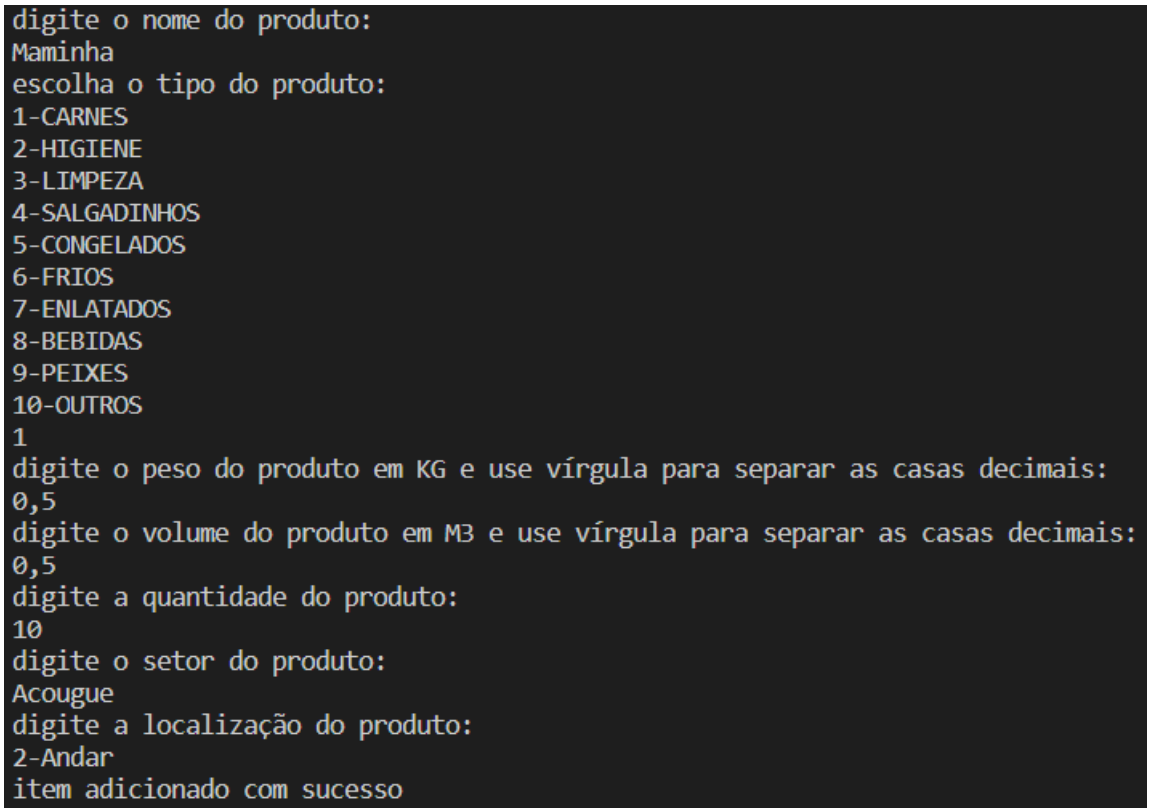
No caso de não conter prateleiras disponíveis para ser adicionado o item, é impresso a seguinte mensagem:” não foram encontradas prateleiras disponiveis para depositar este item”

Apresentação dos dados

Inicialmente o programa apresenta um menu contendo as operações que podem ser realizadas pelo usuário, lembrando que o arquivo csv presente já foi lido e salvo os contidos no mesmo. este menu é apresentado a cada operação realizada, tendo sido bem-sucedida ou não, para que desta forma seja possível o usuário corrigir seus erros, assim não comprometendo a experiencia de usuário, abaixo a imagem apresenta este menu:



Ao selecionar a opção 1 é feita uma seria de perguntas ao usuário relacionado as atribuições Necessárias para adicionar um novo item, abaixo a imagem demonstra este processo:



Caso a opção selecionada for para edição, então é demonstrado o menu da seguinte forma:

```
digite o nome do produto que deseja atualizar:
Maminha
item Selecionado:Maminha
digite abaixo os campos com os valores atualizados
digite o nome do produto:
```

E os demais campos são os mesmos apresentados na opção de inclusão
O método de impressão de itens tem o seguinte aspecto:

```
Nome:Jack Daniels
Tipo:BEBIDAS
Peso:0.5KG
Volume:0.4M3
Quantidade:100
Setor:Consumiveis
Localização na prateleira:2-andar
----------------------------------------------------------------
Nome:Pringles
Tipo:SALGADINHOS
Peso:0.2KG
Volume:0.4M3
Quantidade:5
Setor:Gulouseimas
Localização na prateleira:3-andar
----------------------------------------------------------------
Nome:Ruffles
Tipo:SALGADINHOS
Peso:0.24KG
Volume:0.3M3
Quantidade:2
Setor:Gulouseimas
Localização na prateleira:2-andar
Setor:Casa
Localização na prateleira:1-andar
```

O método de impressão de prateleiras tem o seguinte aspecto:

```
Nome da prateleira:1C
Setor:Acougue
Quantidade de itens:5
Capacidade maxima de peso:10000.0KG
Capacidade maxima de volume:10000.0M3
Capacidade ocupada de peso:481.5KG
Capacidade ocupada de volume:479.0M3
----------------------------------------------------------------
```