Para **resolver** o problema proposto pensei em criar três Structs:

- **Material**
- **Wand**
- **Production**

Uma Wand seria composta de Productions e uma Production seria composta de Materials e um atributo Available. Essa abordagem foi utilizada para que seja possível saber exatamente quais são os materiais que compõem a varinha que estamos analisando. Sem essa estrutura, perderíamos a rastreabilidade, uma Wand teria inúmeros Materials mas não saberíamos qual Material pertence a qual varinha.

Além dessas Structs, pensei em 5 funções principais:

- addMaterial
- listMaterials
- createWand
- sellWand
- listWands

A função **addMaterial** serve para adicionar Material a ledger: Caso já exista o Material com aquela origem, o Material sofre um incremento na ledger, caso contrário é criada uma chave nova e um novo Material é adicionado a ledger.

A função **listMaterials** serve para exibir todos os Materials disponíveis na ledger.

A função **createWand** serve para criar uma nova varinha. Ela só pode ser executada pelo Sr Olivaras(org0-example-com). Primeiro ela verifica se há material disponível na ledger, faz a retirada desse material, atualiza a ledger e registra a varinha nova na ledger.

A função **sellWand** serve para que o Sr Olivaras marque uma varinha como vendida, só ele pode executar essa função. Ao vender uma varinha, wand.Quantity sofre um decremento e também o atributo Available de Production dessa varinha é modificado para false.

A função **listWands** serve para listar as varinhas disponíveis e indisponíveis . Optei por deixar as varinhas vendidas listadas para não perder rastreabilidade. O atributo quantity continua contando apenas as varinhas disponíveis. É possível **verificar** a **disponibilidade** da varinha através do atributo **Available(true or false)** da varinha.

Como executar a solução no Windows 10 (Para executar em outro sistema há uma **pequena** mudança na forma como os argumentos são passados. **Para verificar tais mudanças acessar a documentação do minifabric)**:
Pré-requisitos
[docker](https://www.docker.com/) (18.03 or newer) environment
[Minifabric]https://github.com/hyperledger-labs/minifabric) environment
—----------

Realizando o Deploy do Smart Contract
Abra o terminal em ~/mywork e utilize o comando:

minifab up

Dentro do diretório que foi instalado o Minifabric, navegar até o diretório vars/chaincode/, neste diretório salvar a pasta studio

---

Realizar o Deploy
Abra o terminal em ~/mywork para realizar o deploy

Parametro -n studio representa o nome do smart contract
Parametro -l go representa a linguagem do smart contract
Parametro -v 1.2 é a versão que está sendo realizada o deploy

minifab ccup -n studio -l go -v 1.2

Métodos Disponíveis no Smart Contract
Adicionar Material
Parametro -n studio
Parametro -p nome do método
Parametros obrigatórios: Nome do Material, Origem e Quantidade

minifab invoke -n studio -p \"addMaterial\",\"Carvalho\",\"Bruxo1\",\"10\"

Consultar Materiais Disponíveis
Parametro -n studio
Parametro -p nome do método

minifab invoke -n studio -p \"listMaterials\"

Criar Nova Varinha
Parametro -n studio
Parametro -p nome do método
Parametros obrigatórios: Nome da Varinha, Nome do Material, Origem e Quantidade
(Pode-se adicionar mais materiais em trios - Nome, origem e quantidade)

minifab invoke -n studio -p \"createWand\",\"ElderWand\",\"Carvalho\",\"Bruxo1\",\"5\"
minifab invoke -n studio -p \"createWand\",\"ElderWand\",\"Carvalho\",\"Bruxo1\",\"5\",\"Flor\",\"Bruxo2\",\"10\"

Registrar Venda de Varinha
Parametro -n studio
Parametro -p nome do método
Parametros obrigatórios: Nome da Varinha, Indice relativo ao historico de producao(0 a n-1, sendo n o total de elementos de ProductionHistory referente aquela varinha)

minifab invoke -n studio -p \"sellWand\",\"ElderWand\",\"0\"

Consultar Varinhas Disponíveis
Parametro -n studio
Parametro -p nome do método

minifab invoke -n studio -p \"listWands\"

—----------

Exemplo de execução:

```
1) minifab up
2) minifab ccup -n studio -l go -v 2.0
3) minifab invoke -n studio -p
\”addMaterial\”,\”PenaDeFenix\”,\”Harry\”,\”5\”
4) minifab invoke -n studio -p
\”addMaterial\”,\”PenaDeFenix\”,\”Ron\”,\”10\”
5) minifab invoke -n studio -p \”listMaterials\”
6) minifab invoke -n studio -p
\”addMaterial\”,\”PenaDeFenix\”,\”Harry\”,\”5\”
7) minifab invoke -n studio -p
\”addMaterial\”,\”Carvalho\”,\”Hermione\”,\”2\”
8) minifab invoke -n studio -p \”listMaterials\”
9) minifab invoke -n studio -p
\”createWand\”,\”Varinha1\”,\”PenaDeFenix\”,\”Ron\”,\”2\”
10) minifab invoke -n studio -p
\”createWand\”,\”Varinha1\”,\”PenaDeFenix\”,\”Ron\”,\”1\”,\”PenaDeFenix\”,\”
Harry\”,\”1\”
11) minifab invoke -n studio -p
\”createWand\”,\”Varinha2\”,\”PenaDeFenix\”,\”Ron\”,\”2\”,\”Carvalho\”,\”Her
mione\”,\”1\”
12) minifab invoke -n studio -p \”listWands\”
13) minifab invoke -n studio -p \”sellWand\”,\”Varinha1\”,\”0\”
14) minifab invoke -n studio -p \”listWands\”
```

Imagens:

https://drive.google.com/drive/folders/1y8equdtjbI_kb_bIGi6v6fGI5dHKJj3s?usp=drive_link