Exercício

criar CRUD de usuário

Precisamos que os seguintes dados do usuário sejam salvos no banco de dados

**USUARIO**

* nome
* UUID único
* Email
* Nome da mãe
* senha
* Telefone
* data de nascimento
* endereço completo (Rua, cep, número, complemento)
* CPF
* RG
* Pessoa exposta politicamente (sim/não)
* Renda mensal em reais
* Valor do patrimônio
* Data de cadastro
* Data de atualização
* lista de produtos que gostaria de usar.
* (PIX, TED, pagamento de contas, recarga de celular, Empréstimo)

**PRODUTO**

* nome do produto
* UUID do produto
* Data de cadastro
* Data de atualização
* produto habilitado(sim/nao)

Regras funcionais

não pode ter mesmo EMAIL cadastrado para dois usuários diferentes

um usuário com o mesmo CPF só pode cadastrar 3 contas (todas com e-mails diferentes)

A lista de produtos é dinâmica,

ou seja, eu posso cadastrar um produto ou remover um produto)

o usuário só pode colocar na lista de produtos um produto caso esteja habilitado previamente

um usuário pode cancelar sua conta, mas os dados nunca podem ser apagados do banco de dados

O usuário ao se cadastrar deve indicar quais produtos quer cadastrar

Um produto ao ser cancelado não pode ser excluído da base.

Regras não funcionais

CPF só pode ser salvo em um formato valido

Email só pode ser salvo em um formato valido

Toda regra de negócio deve ficar obrigatoriamente na camada Service

Toda validação de parâmetros de entrada deve ficar no controller.

A aplicação deve ser em java com spring boot e spring jpa com banco oracle e usar gradle

A api deve ser rest enviar e receber json

Checkpoint 2

**Entregar um Readme.md contendo**

* **Versão do java,**
* **versão do gradle.**
* **Se usou oracle ou h2.**

**Usar o cadastro de usuário já criado.**

**Criar Swagger. # não desconta nota**

Criar uma entidade Chave Pix

Um usuário pode ter de 0 a 5 chaves pix cadastradas.

Chave pix possui os seguintes dados  
  
tipo (email,telefone,cpf,EVP)  
ativa (false;true)

Usuário

Valor da chave (Ex: se for email o valor vai ser um email valido [usuario@provedor.com.br](mailto:usuario@provedor.com.br) se for cpf vai ser um cpf valido)

chaves EVP (Chave com valor aleatório com um valor UUID)

Só pode criar chave pix usuário que for ativo na base.

Criar entidade Saldo.

Todo usuário possui um saldo. Ao criar uma conta nova o usuário recebe 100 reais de saldo onde ele pode gastar onde quiser. **O saldo nunca pode ficar negativo.**

Saldo possui os seguintes dados

Valor

Usuário

Caso de uso:  
Fazer um pix:

Ao fazer um pix o valor deve ser subtraído do usuário que está enviando e somado ao saldo do usuário que está recebendo.

O saldo do usuário de debito deve ser positivo.

Ambos os usuários devem estar ativos na base.

O usuário informa a chave pix do destinatário. E o valor enviado deve ser somado no saldo do usuário dono dessa chave pix.

Não funcionais.

Toda transação de pix deve ser armazenada no banco de dados, com o status

Sucesso ou falha. Se houve falha deve ser salvo em uma coluna de descrição o motivo.

Deve ser salvo o horário da transação.

**Um usuário pode buscar todas as transações que ele fez nos últimos 7 dias.**

**Toda transação tem um usuário debitor (quem fez o envio do dinheiro) e um usuário creditor (que recebeu o dinheiro)**

**Um usuário pode buscar todas as transações que ele recebeu nos últimos 7 dias.**

Deve obrigatoriamente usar  
gradle

Spring boot

hibernate

Entrega dia até 23/04 23:59

Caso entregue duas pessoas entreguem mesmo código, as notas de quem entregou mesmo código será 0.

Se o código não buildar ou não subir a aplicação a nota será zerada.

Forma de entrega:  
Criar repositório no github e me enviar o link.  
Será considerado somente commits até esse dia e horário 22/04 23:59 demais commits serão descartados.

Entrega em grupo (mesmo grupo do challenge)