Problema

O sistema de cobrança de passagens, utilizando o cartão BHBUS, dos ônibus da cidade de Belo Horizonte será reformulado. Na nova versão, o passageiro poderá acompanhar seus débitos em tempo real através de um aplicativo disponível para Android e IOS e por um portal. Para que esse controle seja possível será necessário modernizar também o sistema de cobrança de cada ônibus. A máquina onde é feito os pagamentos agora chamará uma api para registrar cada débito feito através dela.

Definições do projeto

Para resolver este problema três empresas foram contratadas. A primeira será responsável pelo desenvolvimento da interface web e aplicativos que serão disponibilizados para o usuário final. A segunda empresa irá atualizar as máquinas de cobrança de cada ônibus para fazer uma requisição ao backend sempre que uma passagem for paga. Já a terceira empresa (você) fará o desenvolvimento do backend, uma api rest responsável por fornecer recursos para armazenar cada requisição feita pelas máquinas e os dados para os app e o portal.

Em uma reunião inicial as três empresas discutiram detalhes do projeto e definiram um contrato para cada endpoint do backend. Dessa forma cada uma poderia trabalhar separadamente e apenas em uma data futura agendariam uma homologação integrada.

Os contratos para a api foram:

Cadastro do passageiro (post /users)

Este endpoint deverá receber um usuário com os seguintes campos: nome, email, id do cartão e senha. Exemplo:

```
{ "name": "string", "email": "string", "cardId": "string", "password": "string" }
```

Obs: Caso o e-mail já exista, deverá retornar erro com a mensagem "E-mail já existente".

Criação de débitos (post /debits)

Este endpoint deverá receber um débito com os seguintes campos: id do cartão, código da linha de transporte e valor. Exemplo:

```
{ "cardId": "string", "code": "string", "value": "number" }
```

Obs: Caso o cardId não tiver sido cadastrado no **post /users** deve retornar "Requisição inválida"

Consulta de débitos (get /debits)

Este endpoint deverá retornar os débitos do passageiro conforme filtro passado por query string.
ex: get /debits?cardId=12345
get /debits?initialDate={initialDate}&finalDate={finalDate}

Em caso de sucesso irá retornar um array de débitos com os seguintes campos:

- id: id do débito
- debitedAt: data de criação do débito
- value : valor do débito

Exemplo:

```
[{ "id": "string", "debitedAt": "date", "value" : "number" }]
```

Orientações

- Utilize a tecnologia que você domina.
- Utilize o banco de dados que você preferir.
- Publicar o código-fonte em um repositório na internet (Bitbucket ou Github).

- Faça uma documentação mínima que você acha relevante do seu projeto.

Desejável

- Para o endpoint **post /debits** implementar uma autenticação básica para que os débitos não sejam criados por qualquer serviço.