Backend engineer take home assignment

The Task

Design a REST API that calculates the cashback for some transactions based on the rulesets provided. Award the highest available cashback for each transaction.

Transaction

A transaction is any financial transaction carried out between a seller and a customer.

Cashback

A cashback is a percentage of a transaction that is offered back to the customer as a an incentive.

Ruleset

Rule define ways and conditions for transaction/customer to earn cashback. Rule-set consists of one or more rules that are applied to the transaction to determine if the transaction/customer is eligible for any cashback or not.

Ruleset example

Each property in the ruleset is a rule. Ruleset may or may not have some of the properties. If the rule property is not defined, then the rule should not be applied.

```
{
    "startDate": "YYYY-mm-dd",
    "endDate": "YYYY-mm-dd",
    "cashback": 2.00,
    "redemptionLimit": 10
}
```

Rules for cashback

- If the transaction is between `startDate` and `endDate` then it is applicable for cashback. Otherwise, it's not applicable.
- Given cashback can not be applied more times than specified in `redemptionLimit`.

Rulesets will be provided through `POST /ruleset` endpoint. This endpoint can be used multiple times to create multiple rulesets. All of the provided rulesets should be applied to the transaction when calculating cashback:

Request:

`POST /ruleset`

```
{
    "startDate": "YYYY-mm-dd",
    "endDate": "YYYY-mm-dd",
    "cashback": 2.00,
    "redemptionLimit": 10
}
```

Response:

201 OK

Transaction example

```
{
  "date": "YYYY-mm-dd",
  "customerId": 1,
  "id": 1
}
```

Transactions will be provided through `POST /transaction` endpoint. This endpoint can be used multiple times.

Request:

`POST /transaction`

```
{
  "date": "YYYY-mm-dd",
  "id": 1
}
```

Response:

201 OK

Cashback example

```
{
  "transactionId": 1,
  "amount": 2.00
}
```

Cashbacks should be returned from `GET /cashback`. Cashbacks should be calculated for each transaction using each of the rulesets provided previously.

Request:

`GET /cashback`

Response:

200 OK

```
[{
  "transactionId": 1,
  "amount": 2.00
}]
```

Optional feature - bud

If you wish, you may also opt to implement the following features

Add rules for budget and minimum transactions

- budget
- minTransactions

```
{
    "startDate": "YYYY-mm-dd",
    "endDate": "YYYY-mm-dd",
    "cashback": 2.00,
    "redemptionLimit": 10,
    "minTransactions": 2,
    "budget": 100.00
}
```

- Each transaction cashback amount should be deducted from the ruleset `budget`. If the ruleset budget is exhausted then further transaction are not applicable for that ruleset.
- Cashback should be only awarded when transaction count for the given ruleset exceeds `minTransactions`.

Apply rulesets for transactions based on `customerId`

Add additional field `customerId` to transaction. Apply `minTransactions` rule based on `customerId` . Meaning if the customer has not done enough transactions cashback should not be returned.

```
{
  "date": "YYYY-mm-dd",
  "customerId": 1,
  "id": 1
}
```

Example - cashback based on date

POST /ruleset

```
{
    "startDate": "2020-04-10",
    "endDate": "2020-05-10",
    "amount": 2.00
}
```

POST /cashback

```
{
    "startDate": "2020-01-10",
    "endDate": "2020-02-10",
    "amount": 1.00
}
```


POST /transaction


```
{
  "date": "2020-03-01",
  "id": 1
}
```


POST /transaction


```
{
  "date": "2020-02-01",
  "id": 2
}
```


POST /transaction


```
{
  "date": "2020-05-01",
  "id": 3
}
```


GET /cashback


```
[{
  "transactionId": 2,
  "amount": 1.00
}, {
  "transactionId": 3,
  "amount": 2.00
}]
```