Financial app

Intro

The goal of this task is to implement a function which generates a simple financial report based on transactions (like bank or credit card transactions).

The report will be an object whose keys are expense categories (like vacation, public transportation etc) and values are the total amount spent on those categories in the relevant timeframe.

Each transaction looks like this:

```
{
 "amount": 166,
 "description": "Star taxis",
 "date": "12/04/2017"
}
```

The report you will generate looks like this:

```
{
 "EATING_OUT": 4325,
 "GROCERIES": 0,
 "VACATION": 228,
 "MEDICAL": 780,
 "PUBLIC_TRANSPORTATION": 0,
 "CAR_MAINTENANCE": 2000,
 "SAVINGS": 350,
 "BILLS": 0,
 "ENTERTAINMENT": 0,
 "NO_CATEGORY": 780
}
```

Each transaction will be categorized to one of the expense categories based on the transaction's description text.

For example, the transaction in the example above, will be categorized as PUBLIC_TRANSPORTATION based on the description Star taxis.

If the classifier isn't able to categorize a given transaction description, it will return null. In this case this transaction
should appear in the report under NO_CATEGORY.

Server

We have implemented a server that has two endpoints you need to use in order to generate the report. This code is already implemented, and you need to use it without changing it.

To start the server run npm install && npm run startServer.

This will start a server on http://localhost:4000 with the following endpoints:

1. Graphql

http://localhost:4000/graphql

A graphql endpoint that exposes one query called transactions that fetches transactions by username and transaction dates.
When the server is running you can go to the url in the browser and see the schema and run queries in the graphql playground.

(A good source for graphql is [this intro (https://graphql.org/learn/)](https://graphql.org/learn/), the relevant parts are queries, fields, arguments and variables)

2. Transaction classification

POST http://localhost:4000/transaction/classification

This endpoint receives a description of a transaction and returns the category of that transaction.

Example of the request post body:

```
{
 "transactionDescription": "Star taxies"
}
```

Example of the response body:

```
{
    "transactionCategory": "PUBLIC_TRANSPORTATION"
}
```

Your code

What you need to do, is implement the generateReport function in test-code/solution/expenseReportGenerator.js

The function receives a username, startDate and endDate parameters which you should use to:

- Fetch the relevant transactions from graphql
- For each of those transactions call the transaction classification endpoint to get the expense category
- Process the data and return the report

Bonus

If you finish the requirements above and still have time, here is a bonus task:

Assume that the classification endpoint has a rate limit of at most 10 concurrent request and will block you if you exceed this rate.

Write your code in a way that limits the concurrent calls you are making to this endpoint to 10.

Notes

- The date format in the test is always DD/MM/YYYY (for example 01/10/2017 or 15/08/2018)
- All transactions are in the date range between 01/01/2017 to 31/12/2018
- All the code you write should be under the solution directory. The rest of the code outside that

directory should be considered an api that you use without changing it.

The exercise is not supposed to be difficult to implement, but do pay attention to:

- code structure
- code readability
- tests
- correct and running solution
  ... and finally code that you are proud of!