Code Challenge

We would like to have a restful API for our statistics. The main use case for our API is to calculate realtime statistic from the last 60 seconds. There will be two APIs, one of them is called every time a transaction is made. It is also the sole input of this rest API. The other one returns the statistic based of the transactions of the last 60 seconds.

Specs

`POST /transactions`

Every Time a new transaction happened, this endpoint will be called.

Body:

```
{
  "amount": 12.3,
   "timestamp": 1478192204000
}
```

Where:

- `amount` - transaction amount
- `timestamp` - transaction time in epoch in millis in UTC time zone (this is not current timestamp)

Returns: Empty body with either 201 or 204.

- 201 - in case of success
- 204 - if transaction is older than 60 seconds

Where:

- `amount` is a double specifying the amount
- `time` is a long specifying unix time format in milliseconds

GET /statistics

This is the main endpoint of this task, this endpoint have to execute in constant time and memory (O(1)). It returns the statistic based on the transactions which happened in the last 60 seconds.

Returns:

```
{
	"sum": 1000,
	"avg": 100,
	"max": 200,
	"min": 50,
	"count": 10
}
```

Where:

- `sum` is a double specifying the total sum of transaction value in the last 60 seconds
- `avg` is a double specifying the average amount of transaction value in the last 60 seconds
- `max` is a double specifying single highest transaction value in the last 60 seconds
- `min` is a double specifying single lowest transaction value in the last 60 seconds
- `count` is a long specifying the total number of transactions happened in the last 60 seconds

Requirements

For the rest api, the biggest and maybe hardest requirement is to make the `GET /statistics` execute in constant time and space. The best solution would be O(1). It is very recommended to tackle the O(1) requirement as the last thing to do as it is not the only thing which will be rated in the code challenge.

Other requirements, which are obvious, but also listed here explicitly:

- The API have to be threadsafe with concurrent requests
- The API have to function properly, with proper result
- The project should be buildable, and tests should also complete successfully. e.g. If maven is used, then mvn clean install should complete successfully.
- The API should be able to deal with time discrepancy, which means, at any point of time, we could receive a transaction which have a timestamp of the past
- Make sure to send the case in memory solution without database (including in-memory database)
- Endpoints have to execute in constant time and memory (O(1))
- **Please complete the challenge using Java**