INTERVIEW QUESTION

Overview

We have collected chat statistics for each day by a unique website Id.

The file contains stats collected over 14 days.

Each entry contains:

- Website Id
- Date
- Total number of chats
- Total number of missed chats

There will only be one entry for a website per day.

For example, if there are a total of 5 websites for a period of 14 days, there will be 70 entries in the file.

The stats file can be found here :

https://bitbucket.org/!api/2.0/snippets/tawkto/aA8zqE/4f62624a75da6d1b8dd7f70e53af8d36a1603910/files/webstats.json

Task

Write a program that will retrieve raw data from provided URL and outputs the sum of chats and missed chats per website Id for a given date range.

Requirements

- Data must be loaded from provided URL
- Date filtering must be optional. If date filters are not provided, the entire dataset needs to be aggregated
- Partial date filtering must be supported. I.e. If only start date is provided - data is filtered from a given start date without end date filtering
- Output must be a JSON array or objects, each containing website ID, number of chats and number of missed chats
- Usage of **lodash** or **underscore** is not allowed

Examples

Note: This example assumes your solution entry-point is a function called `processStatistics` with 2 arguments - `startDate` and `endDate`. Example is provided in Javascript and example output data is truncated, but accurate for data-set and can be used to validate your solution.

Example **without** date range option

```
processStatistics()

output:

[[{
  websiteId: '4f8b36d00000000000000001',
  chats: 1568,
  missedChats: 11
}, {
  websiteId: '4f8b36d00000000000000002',
  chats: 70,
  missedChats: 8
}, ...]
```

Example **with** date range option

```
/**
 * processStatistics(startDate, endDate)
 */
processStatistics(new Date(2019, 3, 5), new Date(2019, 3, 12))

output:

[{
  websiteId: '4f8b36d00000000000000003',
  chats: 1814,
  missedChats: 122
},
{
  websiteId: '4f8b36d00000000000000007',
  chats: 1474,
  missedChats: 3620
}, ...]
```

Notes

Criteria Your solution will be evaluated on include but are not limited to:

- Time and space complexity (solution is expected to execute in linear time complexity)
- Adherence to programming best practices such as SOLID principles
- Readability and documentation
- Unit-tests are not required, but are considered a plus

Write code as if you were to ship it to production — the details matter. Code is expected to be well written.

Your solution must be submitted in a coderpad.io workspace provided to you. Instruction on how to use codepad.io can be found here https://coderpad.io/getting-started-candidates