Client – Server – DB

In this assignment, you'll need to create 3 different services that take part in a data ETL process + a bit more.

1. Client – extracts data to a file and sends it to a server
2. Server – receives data from the client and save it to a local file. The server also serves data to outside clients
3. Data Processor – processes data saved by the server and loads it to a database.

Do the following steps to complete the assignment:

1. Read it through the end and understand what you need to do.
2. Install Node.JS and Postgres DB on your machine and make sure they all work properly.
3. Start coding.
4. You need to send 6 files:
   a. Server code in 1 js (javascript – node.JS) file called `server.js`
   b. Client code in 1 js (javascript – node.JS) file called `client.js`
   c. A file with script to create database table called `db.sql`
   d. Data processor code in 1 js (javascript – node.js) file called `data_processor.js`
   e. `README.md` with running instructions written as markdown (make it clear)
   f. `package.json` with relevant packages to install
5. Open a new Github project and upload your work. Make sure to have your project open for public read.

The Server

The server will have 2 endpoints:

1. **http://localhost:8000/liveEvent** (HTTP method: POST)
   This method will receive a single event from the client and **append it to a local file** (you can select any file name you want for this file)
   The server will authenticate the client's calls using the 'Authorization' HTTP header so when you get a call from the client, find that header in the headers and make sure it equals the word 'secret'. (more about that header here: https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/Authorization)
2. **http:// localhost:8000/userEvents/<userid>** (HTTP method: GET)
   This method will return all data for a given user **from the database table**. The <userid> means that it will be replaced with a user id (for example 'abc123'). Data should be returned as JSON.

- Server must be written in Node.JS (you can use express https://expressjs.com/ or any other framework you want).

The Client

The client will send events to the server 'liveEvent' endpoint. Events will be in the JSON format and will have the fields:

- userId – string (for example: 'abc123')
- name – string - we'll support 2 events: 'add_revenue' and 'subtract_revenue'

- value – integer (for example: 432)

When you send an event from the client to the server, add the secret 'secret' to every call. The secret should be added to the 'Authorization' header of the HTTP call.
The client will read the JSON events from a local file called 'events.jsonl' (read about the jsonl format here: https://jsonlines.org/). The file will have 1 event per line. For example, `events.jsonl` file can look like this:

{ "userId": "user1", "name": "add_revenue", "value": 98 }
{ "userId": "user1", "name": "subtract_revenue", "value": 72 }
{ "userId": "user2", "name": "add_revenue", "value": 70 }
{ "userId": "user1", "name": "add_revenue", "value": 1 }
{ "userId": "user2", "name": "subtract_revenue", "value": 12 }

- Client must be written in node.js, read events from the `events.jsonl` file and send them to the server.

The Database
The DB will have one table called 'users_revenue' that will have 2 columns: 'user_id' and 'revenue'.

- The database should be Postgres (https://www.postgresql.org/download/)

Data Processor
Write a script that reads the events file on the server side, calculates the user's revenue and saves it to the database. Data Processor script should be able to run on different event files in different times and update the database accordingly.

Follow the following guidelines for the data processor:

1. You can choose any one of the following implementation options:
   a. Calculate revenue per user in memory and initiate an UPDATE query to the database one time for every user.
   b. Initiate an UPDATE query for each event you read from the server side events file.
2. In your implementation, take into account:
   a. Very large server side events file (even 100MB of data with tons of users)
   b. Multiple data processors running at once and updating the same users in the database.
3. revenue should be added/subtracted from the current revenue in the database. So, if the user had $100 until now and he got -32 then his new revenue should be $68 in the database.

<u>**MOST IMPORTANT NOTE FOR THE WHOLE ASSIGNMENT:**</u> Keep It Simple! Just do what you're asked to do AND Don't hand over any more or less files than the 6 files requested above.

ENJOY ☺