Epidemic Tracker | Home Assignment

Synamedia - Olympium Team

Exercise pre-requisites:
Setup your environment to support the following requirements:

- Internet connectivity
- NodeJS 14 (LTS version) or greater installed
- Git installed (including git bash console)
- Github user/account created (Send us our git username in order to approve you as a collaborator)
- Implement your solution only within the epidemic-tracker folder
- Home assignment can be implemented in one of the languages:
  Python (Flask/Django), NodeJS (Express), Java (Spring)

1. Simple Hello <message> response

Create a directory by the name: epidemic-tracker, there you will implement your solution.

Implement the REST API call: /epidemic-tracker/hello/<message>
API response should be in the following format (<message> is the message string sent in the request):

```
"hello from epidemic tracker, message: <message>"
```

2. Initial git commit

Make sure that you have your own github user/account and ssh keys are setup.
Initialize a local git repo and add/stage your files using git add command.
Setup the remote repo and push your implementation so far, by executing:

```
git remote add origin git@github.com:demouser8/epidemic-tracker-rp-<your name initials>.git
git push origin master
```

3. Estimating COVID-19 infection risk

Implement the REST API PUT call /epidemic-tracker/covid19/infectionRisk
API is required to estimate the individual risk of infection with COVID-19 based on the countries he visited in, and the duration of his visit in each given country.

PUT body example:

```
{
  "visited": [{"countryName": "Italy", "dateOfArriaval": "2021-01-01", "durationDays": 7},
    {"countryName": "Netherlands", "dateOfArriaval": "2021-01-08", "durationDays": 11},
    {"countryName": "Norway", "dateOfArriaval": "2021-01-19","durationDays": 3}]
}
```

Risk Estimation:
For each visited country that appears in the request's body - fetch COVID-19 metadata from the following 3rd party API (unless data already have been fetched and appear in server's cache):
https://api.covid19api.com/country/<country-name>?from=<date>&to=<date>
Example:
https://api.covid19api.com/country/israel?from=2021-01-17T00:00:00Z&to=2021-01-18T00:00:00Z

The estimated infection risk for day X will be calculated in this manner:

```
dayXNewCases = (day X active cases - day before X active cases)
```

```
sickPopulationRatioFactor = (day x active cases / day x confirmed cases * 100)
estimatedRisk = (dayXNewCases / day X active cases * 100) * sickPopulationRatioFactor * 0.5
```

Required response:

```
{
  "estimatedRisk": <value>
}
```

Note: The program is required to cache COVID-19 data per country at certain dates to avoid redundant network I/O calls. Data shall survive server restarts.
Note: each day requires previous day data, hence this should be considered when fetching data from api.covid19api.com (or fetching data from cache)

4. Subscribe API

Pre-requisite:
Clone DDS repo

```
git clone git@github.com:demouser8/dds.git dds
```

Cd into the dds directory and run the disease-data-source (dds) server by executing:

```
npm install && npm start
```

Implement the subscription REST API PUT call: /epidemic-tracker/covid19/subscribe/route
API allows clients to subscribe to notifications related to the COVID-19 virus developments

PUT body example:

```
{
  "country": "IL",
  "email": subscriber1@notexist.com
}
```

Required response:

201 in case data saved successfully

Fetch new notifications/events from disease-data-source server:
Regardless supporting REST API calls, your server is required to poll the following API, for new COVID-19 cases, every 120 seconds:
http://localhost:3030/disease-data-source/covid19/notifications

DDS server response body example:

```
[{
    “notification”: “Country in lockdown”
    “countryCode: “IL”,
    "country": "israel"
}, {
    “notification”: “Masks are optional”,
    “countryCode”: “ES”,
    country: “spain”
}, {}, …]
```

You are required to write a notification line in the notifications.json file In this format:

```
[
  {"email": subscriber1@notexist.com, country: “israel”, notification: “Masks are optional”}
]
```

For each qualified subscriber the program is required send the proper notification (e.g. IL subscribers should get only notifications relevant to Israel)

5. Publish notification to subscribers (Bonus question #1)

Pre-requisite:
Clone DDS repo

```
git clone git@github.com:demouser8/dds.git dds
```

Cd into the dds directory and run the disease-data-source (dds) server by executing:

```
npm install && npm start
```

Fetch confirmed cases data from disease-data-source server:
Regardless supporting REST API calls, your server is required to poll the following API, for new COVID-19 cases, every 120 seconds:
http://localhost:3030/disease-data-source/covid19/cases/confirmed

Response body example:

```
[{
    "route": [{"lat": 23.54545, "long": 24.44234}, {"lat": 23.54545, "long": 24.44234}],
    "date": "2021-01-14",
    "country": "IL"
}, {}, {}, …]
```

If a new confirmed case has been fetched successfully, the program is required to validate whether any subscriber has been within the infection zone, which is **True** if:
Any point of the confirmed case route has been less than 2 meters from any of the subscribers' route points.

For each qualified subscriber the program is required to write a notification line in the notifications.json file in the following format (example):

```
{"notify": "subscriber1@notexist.com", "date": "2021-01-14", "interactionPoint":
  {"lat": 23.54545, "long": 24.44234}}
```

Note: The data sent by the subscriber using the subscribe API described in task #4, should be saved in a way that will be most efficient for validating against a new COVID19 confirmed cases.
Any 3rd party library can be used. (e.g. don't validate against subscribers that holds a different date value)

6. Docker container (Bonus question #2)

Build a docker file for your server and add it to git source control.

Additional details

API information: https://documenter.getpostman.com/view/10808728/SzS8rjbc
Email for questions: guylevy@synamedia.com

Good luck and Enjoy