Senior Software Engineer (Backend) Coding Exercise

Overview

This coding exercise involves designing and implementing an API-only application that returns the closest car parks to a user, together with each parking lot's availability.

Please use Java for this exercise. The choice of any frameworks or tools are up to you. You should also include the use of a relational database, such as `MySQL` or `PostgreSQL` (if there is a good reason not to use a relational database, tell us!)

Data sources

- **Car park information**

  Dataset: https://beta.data.gov.sg/datasets/d_23f946fa557947f93a8043bbef41dd09/view

  This dataset provides detailed information about each car park. This can be treated as static and is to be loaded with a task from the CSV file provided in the link above. You can check it into the repository and do not have to automate the downloading and updating of this data.

  Hint: The coordinates provided are in a `SVY21` format. You may have to do some conversion to a more widely used format. An option is to use a converter (https://www.onemap.gov.sg/apidocs/apidocs/#coordinateConverters) when importing or a similarly implemented library.

- **Car park availability**

  Dataset: https://beta.data.gov.sg/collections/85/view

  The API endpoint in that link provides live updates on the parking lot availability for the car parks. Create a task to call the API and save this data in the database. Scheduling the task is out of the scope of this project, but if manually run, the same task should be able to update existing data with newer data from the API if run periodically.

Business Requirements

- The endpoint should take the url parameters `latitude` and `longitude`, and return a JSON array of car parks sorted by distance ascending with the total and available parking lots.
- Requests without required parameters `latitude` and `longitude` should return HTTP status code `400`.
- Pagination should be implemented with `page` and `per_page` to iterate through a subset of results. Only car parks with available parking lots should be returned.
- Input validation should be implemented where you see fit.

Expected output

Request

```
GET /carparks/nearest?latitude=1.37326&longitude=103.897&page=1&per_page=3
```

Response

```
[
    {
        "address": "BLK 401-413, 460-463 HOUGANG AVENUE 10",
        "latitude": 1.37429,
        "longitude": 103.896,
        "total_lots": 693,
        "available_lots": 182
    },
    {
        "address": "BLK 351-357 HOUGANG AVENUE 7",
        "latitude": 1.37234,
        "longitude": 103.899,
        "total_lots": 249,
        "available_lots": 143
    },
    {
        "address": "BLK 364 / 365 UPP SERANGOON RD",
        "latitude": 1.37011,
        "longitude": 103.897,
        "total_lots": 471,
        "available_lots": 324
    }
]
```

Some rules

- Use version control (preferably Git but any will do), and use it as you would for your regular work.
- Your application should be in a state that is runnable by our reviewers.
- Include documentation in a top level `README.md`, explaining the approach, instructions on how to run your application, and any other details as you see fit.

For bonus points

- We love clean, readable code that comes with tests.
- Keep in mind scalability, performance and maintainability concerns.
- Package your development environment with something like Docker Compose.
- We love to hear other people’s thought process and the tradeoffs they consider, so write those down.

Guidelines

We understand that your time is precious, so please aim to spend between no more than 8 hours on this exercise in total. As a guideline, consider spending between 3-8 hours and then calling a hard stop.

Please try to make a submission within 1 week, but if you need more time due to personal or work commitments, just drop your recruitment coordinator a note.

Submission

Please host the code for your submission on GitHub or a similar service like Gitlab as a private repository, and then invite Cuong (https://github.com/Cuongle-Wego) as collaborator.

Drop Cuong (cuongle@wego.com) an email when you’re done.