Oligarch Rating Exercise

Purpose

The purpose of this exercise is to create a **Java microservice** that exposes **REST API** according to **industry standards**. This MS is called '***oligarch-rating***' and its functionality is to manage the *world oligarch rating*.

The following MSs with its API is given (you are not supposed to implement it):

`assets-valuation` MS
- **`GET assets-valuation/cash/evaluate?amount={amount}&localCurrency={localCurrency}`**
- **`GET assets-valuation/bitcoin/value`**

oligarch-*helper* MS
- `GET oligarch-helper/oligarch-threshold`

The Flow

- The **client** of the ***oligarch-rating*** MS sends person's data to the ***oligarch-rating*** service:

```
{
  "id": 123456789,
  "personInformation": {
    "firstName": "Bill",
    "lastName": "Gates"
  },
  "financialAssets": {
    "cashAmount": 16000000000,
    "currency":ILS,
    "bitcoinAmount": 50
  }
}
```

- The ***oligarch-rating*** analyses the person's financial status (evaluates his/her **`assets value`**) using the ***assets-valuation & oligarch-helper*** APIs:
  - the service will call the ***assets-valuation*** APIs:
    `GET assets-valuation/cash/evaluate?amount={amount}&localCurrency={localCurrency}`
    `GET assets-valuation/bitcoin/value`
  - the ***assets-valuation*** will return the asset value in USD.
  - Then it will call another API:
    `GET oligarch-helper/oligarch-threshold`
    to get the threshold value to be considered as an oligarch.
  - The person's fortune is calculated as follows:
    `assetsValue = cash + numberOfBitcoins * bitcoinValue`
- If the person's assets value is greater than the threshold, meaning the person is considered as an **Oligarch**, it will be persisted to the DB by the '***oligarch-rating'*** service with the following fields:
  - ID
  - firstName
  - lastName
  - assetsValue

Requirements

1. Implement the oligarch-rating microservice in Java (use Java 17) with Spring Boot, with the following end points:
    a. Endpoint to handle the oligarch-rating's client request with the person's information (as described above).
    This endpoint will return a proper response according to the person's financial status.
    b. Get all Oligarchs persisted in the DB.
    c. Get Oligarch information by ID.
    d. Bonus: Get the oligarch rank by ID (E.g. Person ID XXX is ranked 2nd in the world's oligarch list).
2. Add unit tests.
3. Submit your solution within 24 hours of receiving it.

Submission Notes

Submit your project via one of these options:

1. Google Drive
2. Mail
    a. Delete the project "build" folder
    b. Compress the project to a zip file
    c. Email the project zip file back

Note: Do not upload the project into public repositories (such as GitHub)

Tips

1. **Spring Boot** makes it easy to create stand-alone, production-grade Spring based Applications that you can "just run". Many commonly used capabilities are implemented by Spring and can be used with minimal configuration. Such as
    a. Creating a microservice from scratch (Spring initializer will be very helpful)
    b. Exposing REST API easily
    c. Accessing data in DB
2. An **in-memory database** can be used for this exercise (consider using h2 in memory DB)