Description and Requirements

We want to build a **new application that will process metering data (meter reading).**

Concepts in the domain:

- Client – client residing on a specific address with a smart meter installed at home
- Address – the address where the client resides
- Meter – smart meter capable of measuring electricity consumption and keeping historical data about the energy consumption

Each client resides at a specific address.
There cannot be two clients at the same address (pay attention how this check will be implemented, HINT: object equality).
Each client can have only one meter.

The project build has to be a Spring Boot application.
For the build tools either Maven or Gradle have to be used.
Java version cannot be older then 8 (make sure the build tools enforce this)

Functional requirements

All the functional requirements must be available through a REST endpoint.

- We should be able to get the aggregate meter reading for a year
  The response should contain the year and the aggregate value for that year (example: **year = 2018, total = 30**)
- We should be able to get the meter readings per year
  The response should contain the year and the meter readings for each month including the month name (example, **year = 2018, January = 2, February = 5, March = 7 etc.**)
- We should be able to get the meter reading per month per year
  The response should include the year, the month and the value for the requested month (example: **year = 2017, September = 21**)

The examples above do not dictate the data format to be used, they merely give an insight to avoid confusion.

Provide initial data to support the above-mentioned functionalities. This data does not have to be in a database it can be modelled in the application (look at the bonus section as well).

NOTES

- Pay attention to domain modelling (keep in mind the principles for abstraction from the basics of OOP)
- Use proper data types for model fields
- Pay attention how the application is structured
- The build, starting and usage of the application must be well documented
- The end result must be a project which can be easily build, started and tested (projects which do not build according to the provided documentation will not be taken in consideration)

Bonus

- Additional REST endpoint which will allow to add additional meter readings to the meter. We should be able to add a specific meter reading value for a specific year and specific month (if the month already has a meter reading value an appropriate error should be raised)
- Data persistency with an **in memory** relational data storage

Timing

We estimate that this exercise could take you **4-8 hours**. If you don't have time enough you can always time-box yourself and send us a partial implementation with some comments, we are flexible in that sense. In this case just make sure of explaining the shortcuts you make or the things you miss, and what would have been the next steps in case you would have more time.

How to send us the Use Case

After the solution is created please upload the code to a bitbucket **private** repository.

Once the code is uploaded share with us the repository (in bitbucket under **User and group access** add the **theitsolutions** user to have read access to the repository, it will resolve to name Zolt Egete). Also send a notification mail once the repository has been shared.