**M111 - Big Data Management**
**Spring 2021 Programming Assignment**

*Andreas Paterakis*
*Department of Information and Communication Technologies*

## **Introduction**:

This is a project of a simple version of a distributed, fault-tolerant, Key-Value database. These records are stored and retrieved using a key that uniquely identifies the record, and is used to find data within the database.
This assignment consists of three parts: The **Generator,** the **Broker** and the **Server**. All of them are written in Java and use the Apache maven building tool to create the executable jars.

### 1) Generator

The generator is responsible for the creation of a .txt file that includes data to process by the broker and server.

It starts with the following command:

```
java -jar target/Generator-1.0-SNAPSHOT.jar -k keyFile.txt
-n 50 -d 3 -l 4 -m 5
```

where

- -n indicates the number of lines to generate
- -d the maximum level of nesting a value can have in a set of key-value
- -m the maximum number of keys inside each value
- -l the maximum length of string value.

All the above parameters should be included in the command that initiates the generator.

The appropriate format of the keyFile file is this one:

```
"name",string
"age",int
"address",string
"number",int
"zipCode",int
"height",float
"email",string
"profession",string
...
```

and the outcome of the generator is a file with this format:

```
"person1" : {"zipCode" : 33 ; "profession" : {"number" : {}} ; "age" : 39}
"person2" : {}
"person3" : {"zipCode" : 43}
"person4" : {"profession" : "GrGp" ; "number" : 27}
...
```

The generator takes into consideration the number of elements the 'keyFile' contains, so no duplicate data will be generated per line. The total number of keys (-m) should be less than the number of keys inside the 'keyFile'.

**2) Broker**

The broker starts with this command:

```
java -jar target/kvBroker-1.0-SNAPSHOT.jar -i
dataToIndex.txt -s serverFile.txt -k 2
```

The broker in order to start expects the file that contains data in the appropriate form, as the generator creates (-i parameter represents the file name). Furthermore, it expects a file that contains the ports of the servers, which it will connect to(-s parameter). The data in the second file must be of this format:

*localhost: 5000*
*localhost: 5001*
*localhost: 5002*

The broker should be deployed **after** the servers' deployment. Otherwise there will be no connection at all. *'localhost'* is a constant for each line. The only variable part is ports one (*i.e 5001, 5002 etc...*).

Lastly, -k parameter is not necessary to be specified and it will be set to ‘1’.

Once the kvBroker starts, it connects to all the servers, and for each line dataToIndex.txt it randomly picks k servers where it sends a request of the form PUT data.

After the data input, the broker is ready to take requests from the user of the following format:

- Any kind of quotes (“”) are not allowed in the queries.

a) **GET key**

This request should have the following format:

Request:
*GET person25*
Response:
*person25 : {"zipCode" : 39 , "profession" : "eheZ" , "age" : 19}*

b) **QUERY keyPath**

This request could have the following formats:

Request:
QUERY person25
Response:
*person25 : {"zipCode" : 39 , "profession" : "eheZ" , "age" : 19}*

Request:
QUERY person25.profession
Response:
*person25.profession : "eheZ"*

c) **DELETE key**

Delete request deletes the specified high-level key, so the only acceptable format is the following:

Request:

*DELETE person25*

Response:

*person25 : DELETED*

In all the above requests, if the data asked from the user is **not found** then the broker response is this:

*person25 : NOT FOUND*

In case a server goes down, the broker informs the user about the server failure and the remaining number of the operating ones.
Furthermore, if the number of servers that are still connected to the broker and operating is equal or less than the replication factor (-k) the user asked, the broker types the following message:

*WARNING: It seems that there are less than 2 servers still running. Can not guarantee the correct output*

**3)Server**

The Key-Value server starts with the following command:

```
-jar target/kvServer-1.0-SNAPSHOT.jar -p {port}
```

The server maintains a data structure called **‘trie’** for both key and value.
It uses the trie structure for indexing and searching all of the keys (both top-level and within the values and the nesting) and no other data structure.

The server uses the *jackson libraries* for PUT functionality.

General information

All the files(i.e .txt) that each part of the assignment asks for its instantiation, must be inside the corresponding project file, otherwise it will not be able to start. For your information, they all already include the files the need to operate.

In case the broker is terminated, servers that are connected to it are terminated as well.