DEV
CHALLENGE
12

Backend Task

Level: **Standard** (part A) and **Pro** (part A and part B). Part C – bonus task.
Duration = **6-7 hours**
Deadline: **23 June 18:00 EEST**

Shared Database without dedicated master.

A) **Task**:

Consortium of N organization want to share common data, which is useful for all of them. This common data consists of set of pairs (name, value)). But there is a small problem: they can't decide, which organization is main and who will run a master database. So, they decided, that each organization will have their own copy of the database and for each modification, all data will be replicated.

Can you help them by designing algorithm for data update:

You should implement a service, which will accept next commands:

- update(name, value) (where name and value are strings)
- get(name)
- remove(name)
- dump(fileName) (dump content of the current data to the file in human-readable format)

Via some network API (rest or some sort of IDL - doesn't matter). Authorization is omitted for the simplicity. Set of the data is relatively small and can fit in memory.

Service can use additional internal API for data update process.
Also, we have configuration, where IP address for all other servers are stored.

Provide a file, which contains:

1. Project (in any programming language: from Scala to PHP)
2. Instruction for compiling and starting/installing. (Preparing install package is not requirements, run from project root is fine).
3. Instruction, how to load and store name-value pair and how to look on dump file.
4. Way to start node a docker will be a plus.

When we update key on one computer, it should be updated on all computers in network and become visible for all nodes (i.e. get to any node should return value).

B) After consortium implemented shared database, next problems was discovered:

1. New members on consortium want to participate and connect/host own copy of shared database.. So, we need automate node unboarding and make list of nodes dynamic
2. Some node can be offline for maintenance or by failures. But we don't want shutdown whole shared database for all time

Task:

1. Describe the solutions for both problems

2. implement onboarding/restoring of node after maintenance.

C) After great success of previous implementation, consortium decided to open database to bigger audience (more than 1000 nodes) with next changes to the model:

1. For each key, instead of value, we want to keep sequence of the values with the time and the author of the last change. In external API method get return last value, and new method getHistory - queue all values for given key.
2. We don't want to wait after submitting value, why it will be propagates in all nodes, so, we will need next changes to the public API:
    a. Get method return last value one for all online nodes.
    b. We want have to ability to subscribe on changes, which will call our application, when valua (i.e. addListener call). Mechanism of the notification (RPC callback, websocket, rtc.. - not matter).

Task:

Implement methods:
getHistory
Get
addListener

With new semantics.

Result presentation format for Standard & Pro

1. Implement service of your choice: (part A, B) and optional - (C (bonus only)).
2. Provide project (in any programming language, from Scala to PHP).
3. Provide file in any human readable format (from markdown to .pdf) with description.
4. Upload your result at your Account https://devchallenge.it/user/ in ONE archive named Backend_Final.zip. Names of files inside the archive should not contain your name or last name.
5. The organizers and judges can disqualify the work of the participant if the work:
    a. contains any indication of the name, surname, e-mail, company, address or other personal data of the participant;
    b. done in a different format than specified in the task;
    c. done with the help of others, and not the participant personally.

Get more information about judges.

Criteria for Standard	**Max points you can get**
Basic Service Commands	80
Algorithm for N-master replication	200
Testsuite for N-master replication	120
General quality of code	64
Instructions for update/	28
∑ for Standard	**512 points**

Criteria for Pro	**Max points you can get**
Description of the fault tolerance technique	120
Description of the onboarding/restore algorithm	80
Implementation of fault-tolerante replication	80
Implementation of setting-up node after shutdown.	80
Implementation of dynamic set of nodes	40
Description of test-cases for fault-tolerance	28
General code quality	64
∑ for Pro	**512 points**

32 bonus points goes for part C.

Please note that:

1. The results must be uploaded at your Account https://devchallenge.it/user/ in ONE archive named Backend_Final.zip. Names of files inside the archive should not contain your name or last name. The results must be uploaded by June 23, 18:00 (EEST), after that time it will be blocked automatically.
2. You can ask questions and clarify the content of the task at DEV Challenge 12 Slack Backend chat. Before submitting a question, check out the chat. Perhaps your question has an answer already :)
3. Judges will ignore questions that are not relevant to the Championship.
4. Your support is help team at the venue, you can always ask them**.**
5. The announcement of the results will be held on June 24 at 16:30 EEST.