Online Round Task
Backend | DEV Challenge XIX

Content:

1. Task description

Trust network is a community of people who know each other and can assign each other a trust level. We usually use trust networks in real life when donating money, searching for healthcare professionals or expertise: we search between friends, friends of friends, etc.

Let's model information flow in our computer. The main entities are

1. Person, which has the next attributes:
   a. Id.
   b. Set of expertise topics [strings]
   c. Set of relations to other peoples: pairs
      i. Id - contact id,
      ii. trustLevel - number between 1 or 10

We can build a model of trust network using straightforward REST API:

2. Description of input data

You should implements server with the next endpoints:

Request POST /api/people

```
{
    "id": "Garry",
    “topics”: [“books”, “magic”, “movies”]
}
```

Response 201

```
{
  “id”: “Garry”,
   “topics”: [“books”, “magic”, “movies”]
}
```

Also we can update or create trust connections:

Request POST /api/people/Garry/trust_connections
Hash pair with person_id - trust level

```
{
  “Ron”: 10,
  “Hermione”: 10,
}
```

Response 201

Request POST /api/people/Garry/trust_connections

```
{
    “Snape”: 4,
   “Voldemort”: 1
}
```

Response 201

Should add contacts 8 and 12 with trust level 1 and 5 accordingly

The main work is a sending messages (question, search for expertise, etc), which should have form:

Request POST api/messages
Request:

```
{
        “text”: “Voldemort is alive!”,
        “topics”: [“magic”],
        “from_person_id”: “Garry”,
        “min_trust_level”: 5
}
```

*all fields are required

Response should trace message delivery through the network based on people topics and trust connection levels. Each person should receive this message only one time and not be spammed. All persons who receive a message must have appropriate topics.

Note, that message is send broadcasted to all

Response 201

```
{
    "Garry": ["Hermione", “Rone”]
}
```

Bonus – implement delivery of non-broadcast message, where

Receiver should have topics listed in requests,
Intermediate nodes can not have topics, listed in request

Request POST api/path

```
  Request: {
     "text": "need to find an expertise in magic",
     "topics":  ["books",",magic"],
     “from_person_id”: “Garry”,
     “min_trust_level”: 5
  }
```

Response 201

This message should find an receiver, which have appropriate topics in attributes. All participants in the path should be connected with a trust level of 5 or more.

As a result, we should receive back:

```
{
  from: "Garry"
  path: ["Hermione"]
}
```

- the path from the message sender to the message receiver, including all intermediate agents. When we have more than one variant, we should return a shorter variant.

3. Format of presentation of results

Downloads the decision in the Participant's account on the platform in **one file archive**.

☝️ Please note that the .git directory should not be present in the archive

☝ Please note that the name of the archive and file names inside the archive should not contain your first or last name. The size of the solution archive should not exceed **10 MB**.

The Organizers and Judges reserve the right to disqualify the Participant's work if the work:

- contains any reference to the Participant's name, surname, e-mail address, company, address, or other personal data;
- completed in a different format than specified in the task;
- performed with the help of third parties, and not by the Participant personally.

The archive should contain

- a 'docker-compose' file in the root, which starts a server with the given endpoints on '/api' URI on port 8080, available from localhost as 'http://localhost:8080/'.
- README.md, where you wrote instructions on how to start service and tests and some thoughts about your choices during performing this task and the next steps to make your service better.

4. Deadline for submission

October 3, 2022 — after the time runs out, the possibility of uploading works to the platform will be automatically blocked. The participants who have moved on to the Final will be announced on October 10, 2022.

5. Evaluation criteria

Criteria	Points
Technical assessment	**166**
Result Correctness	90
Following API Format	38
Performance	38
Expert assessment	**90**
Code quality	38
Test	52
Bonus task	**128**
Result Correctness	90
Following API Format	38

6. Contacts

Questions and clarifications regarding the content of tasks:
Slack channel: #nomination-backend.
☝ Judges will ignore questions that do not relate to the tasks of the Championship.

Organizational questions:
Contact us via e-mail hello@devchallenge.it or Slack channel #02-ask-the-organizers.