Demo

Login Service

Login service manages the registration and login of users. On registration, login, and password of the user get stored in the MongoDB database together with generated uuid. On successful login, the specific user gets redirected to the tasks microservice.

To provide redundancy and high availability, the database runs as a replica set with 3 members in it, and if any member gets disconnected, the database runs in read-only mode.

To start the service, run start_mongo.sh script and python file.

```
./mongo_start.sh
python login_microservice/authentication_service.py
```

Task Service

This service contains tasks themselves. You can add and remove tasks,
and also go to feedback service. The login page redirects to this service.

It works with hazelcast, so whenever a new task is added or deleted,
it is added/deleted from hazelcast. To run it, install hazelcast to
docker and start a couple of hazelcast members.

```
docker pull hazelcast/hazelcast:5.3.0      # pull hazelcast t
docker network create hazelcast-network    # create hazelcast

# start hazelcast members on different ports
docker run --name first-memeber -it --network hazelcast-netwo
           --rm -e HZ_CLUSTERNAME=tasks -p 5701:5701 hazelcas
docker run --name second-memeber -it --network hazelcast-netw
           --rm -e HZ_CLUSTERNAME=tasks -p 5702:5701 hazelcas
docker run --name third-memeber -it --network hazelcast-netwo
           --rm -e HZ_CLUSTERNAME=tasks -p 5703:5701 hazelcas

# open hazelcast manager to see distribution of data etc. (it
docker run --network hazelcast-network -p 8080:8080 hazelcast
```

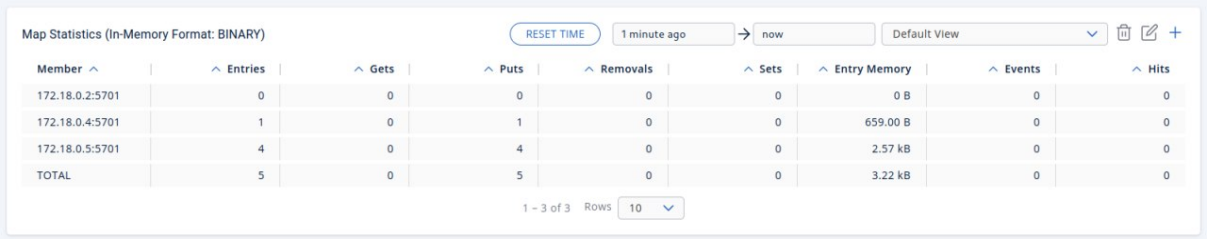
Also, note that it is important to install `hazelcast-python-client` to run code. Then you can run the service with the following line in tasks service folder:

```
python app.py
```

Examples

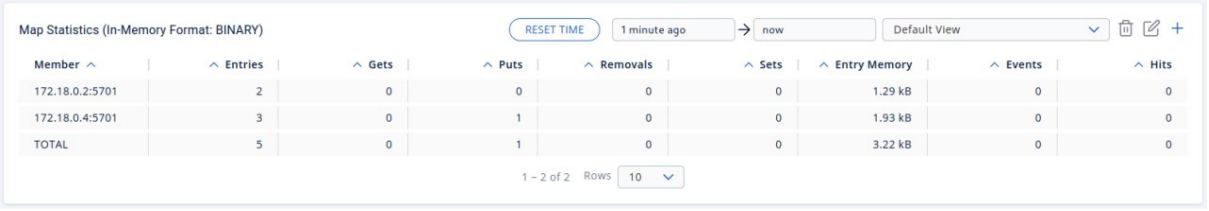
At first, we have 3 hazelcast members, and we added 5 tasks. We can see how they are distributed among members:

```
Members {size:3, ver:3} [
        Member [172.18.0.2]:5701 - 7d5bf792-aa04-45d0-b051-ed2bbd5a5f59
        Member [172.18.0.4]:5701 - c40a26fa-82b0-4dae-b69e-f6de6da4ade1 this
        Member [172.18.0.5]:5701 - fb398031-b6c5-4f70-bdbd-b751655508e6
]
```



After that we stop one member, and see that tasks are redistributed among remaining hazelcast members without loss:

```
Members {size:2, ver:4} [
        Member [172.18.0.2]:5701 - 7d5bf792-aa04-45d0-b051-ed2bbd5a5f59
        Member [172.18.0.4]:5701 - c40a26fa-82b0-4dae-b69e-f6de6da4ade1 this
]
```



Feedback Service

Displays feedback from other users and allows you to submit your own feedback. It can be found on/the feedback page.

It is implemented using cassandra database. To run this service you should download cassandra for your system and start its service, e.g. on linux you would do

```
sudo systemctl start cassandra
```

Then, run init.cql script, which will create keyspace & feedback database, and add some entries to database.

```
cqlsh -f init.cql
```

Now that cassandra database is configured, so install requirements for python application and run flask application.