OpenStack: Timeseries as a service

Gnocchi is an open-source, multi-tenant timeseries, metrics and resources database. It provides an HTTP REST interface to create and manipulate the data. It is designed to store metrics at a very large scale on CEPH while providing access to metrics and resources information and history.
Install in an OpenStack instance the Gnocchi database and create two simple applications, one consumer and one producer, that exploit the REST interface exposed by Gnocchi to store/retrieve the data. The producer must mimic the periodic production of measurement data (random data is OK) to be stored in the timeseries database, the consumer must retrieve periodically some aggregated values like average, max, min, etc.

References:

- Gnocchi installation: https://jaas.ai/gnocchi/37
- Gnocchi REST interface: https://gnocchi.xyz/rest.html

Workbook

Gnocchi installation can be performed in a similar manner as the other OpenStack services: Install the components.
On the architecture manager:

```
juju deploy --to lxd:0 gnocchi
juju deploy --to lxd:0 memcached
juju add-relation gnocchi mysql
juju add-relation gnocchi memcached
juju add-relation gnocchi keystone
juju add-relation gnocchi ceph-mon
```

Install the OpenStack command line tools required to retrieve a token to be used to interact with Gnocchi.

On the controller:

```
snap install --classic openstackclients
```

Gnocchi exposes a REST interface to create metrics, push the data and retrieve the data. Gnocchi REST interface requires a token that can be retrieved through the following command:

```
source admin.sh
openstack token issue
```

The REST interface exposed by Gnocchi is available on the IP address of the container in which Gnocchi runs (it can be retrieved from juju status).
Some operations:

- **Retrieve Gnocchi current status**
  ```
  curl -H "X-AUTH-TOKEN: AUTH_TOKEN" http://252.3.26.90:8041/v1/status?details=False
  ```
- **Create a new metric**
  ```
  curl -d "{\"archive_policy_name\": \"name_metric\"}" -X POST -H "Content-Type: application/json" -H "X-AUTH-TOKEN: AUTH_TOKEN" http://252.3.26.90:8041/v1/metric
  ```
- **Retrieve the status of a metric and its ID**
  ```
  curl -H "X-AUTH-TOKEN: AUTH_TOKEN" http://252.3.26.90:8041/v1/metric
  ```
- **Push some measurements**
  ```
  curl -d "[ { \"timestamp\": \"2014-10-06T14:33:57\", \"value\": 43.1 }] " -X POST -H "Content-Type: application/json" -H "X-AUTH-TOKEN: AUTH_TOKEN" http://252.3.26.90:8041/v1/metric/ID_METRIC/measures
  ```
- **Read measurements**
  ```
  curl -H "X-AUTH-TOKEN: AUTH_TOKEN" http://252.3.26.90:8041/v1/metric/ID_METRIC/measures
  ```