20-21 International Project

REPORT: SPRINT 4

Jyri Jacobson, Kalle Kaukola, Timi Partala, Arvo Cant & Casper De Keyser

Group 2

Table of contents

1 Overview

In this sprint we focused on connecting everything together and adding the last modifications so everything was ready to use. We also prepared the presentation and demonstration of our project, so we could show off everything we have done.

2 Sprint review

2.1 Hardware

In this week we focused to get the BPM calculation code to work, after few trial and error we found out that Arduino does not support "^" as math sign. After some more time we got the calculation fixed so it works..

We also set everything up on Jyri's local network with Casper's help.

The last thing we did was film the demo video and prepare the presentation by writing the slides we were going to discuss.

2.2 Software

2.2.1 Frontend

This last sprint we wrote a http client so the app can get data form the NodeJS server. Afterwards, Casper had a meeting with Jyri to configure the app further for his setup.

2.2.2 Backend

During this last sprint I mostly did some refining of the backend. I changed the server files a bit so it was easier for Jyri to configure it on his system. I also configured the extra containers I wanted to use so I was able to monitor the system accurately. I will explain both parts in more detail:

- **Updating server files**

In the server files, I added some fields at the top so it was easier to use the files on other systems. In the following screenshot you can see what that looks like:

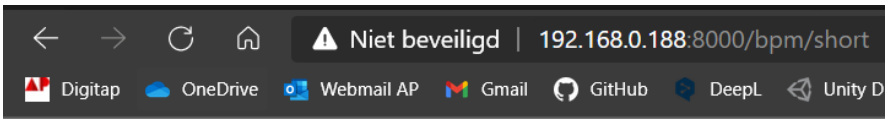
```
const url = "http://192.168.1.101:8086"; // change this to the IP of the InfluxDB instance (usually it's http://localhost:8086)
const org = "intproject"; // change this to the name of your organisation
const token =
  "lpdGK1n0Mdce9t9auJ9nvPH4YSDSlnFLGTUESkxe6HosLjHGIsadzakVhHgTtPHOGqRMwHqdgqHWHv0SFPblpg=="; // change this to the token of your bucket

const queryApi = new InfluxDB({ url, token }).getQueryApi(org);

const bucket = "ecgdb";
const measurementName = "heart_rate";
const fieldVoltageName = "ecgValue";
const fieldTimestampName = "timeValue";
const fieldBpmName = "bpm";

const shortQuery = "30s";
const mediumQuery = "1m";
const longQuery = "2m";
```

I also added some other routes to the webserver so it was possible to also query for the calculated heartbeats, because the Finnish guys were able to calculate this. It would have been unfortunate if they managed to do this calculation and it wasn't used in the further flow. This route is visible in the next screenshot:




```
[
  {
    "result": "_result",
    "table": 0,
    "_field_bpm": "bpm",
    "_field_timestamp": "timestamp",
    "_measurement_bpm": "ecg",
    "_measurement_timestamp": "ecg",
    "_start_bpm": "2021-04-30T08:42:14.637821211Z",
    "_start_timestamp": "2021-04-30T08:42:14.637821211Z",
    "_stop_bpm": "2021-04-30T08:42:44.637821211Z",
    "_stop_timestamp": "2021-04-30T08:42:44.637821211Z",
    "_time": "2021-04-30T08:42:15.45712Z",
    "_value_bpm": 94,
    "_value_timestamp": 1619772135453
  },
  {
    "result": "_result",
    "table": 0,
    "_field_bpm": "bpm",
    "_field_timestamp": "timestamp",
    "_measurement_bpm": "ecg",
    "_measurement_timestamp": "ecg",
    "_start_bpm": "2021-04-30T08:42:14.637821211Z",
    "_start_timestamp": "2021-04-30T08:42:14.637821211Z",
    "_stop_bpm": "2021-04-30T08:42:44.637821211Z",
    "_stop_timestamp": "2021-04-30T08:42:44.637821211Z",
    "_time": "2021-04-30T08:42:16.45795Z",
    "_value_bpm": 91,
    "_value_timestamp": 1619772136454
  },
  {
    "result": "_result",
    "table": 0,
    "_field_bpm": "bpm",
    "_field_timestamp": "timestamp",
    "_measurement_bpm": "ecg",
    "_measurement_timestamp": "ecg",
    "_start_bpm": "2021-04-30T08:42:14.637821211Z",
    "_start_timestamp": "2021-04-30T08:42:14.637821211Z",
    "_stop_bpm": "2021-04-30T08:42:44.637821211Z",
    "_stop_timestamp": "2021-04-30T08:42:44.637821211Z",
    "_time": "2021-04-30T08:42:17.458844Z",
    "_value_bpm": 105,
    "_value_timestamp": 1619772137455
  },
  {
    "result": "_result",
    "table": 0,
    "_field_bpm": "bpm",
    "_field_timestamp": "timestamp",
    "_measurement_bpm": "ecg",
    "_measurement_timestamp": "ecg",
    "_start_bpm": "2021-04-30T08:42:14.637821211Z",
    "_start_timestamp": "2021-04-30T08:42:14.637821211Z",
    "_stop_bpm": "2021-04-30T08:42:44.637821211Z",
    "_stop_timestamp": "2021-04-30T08:42:44.637821211Z",
    "_time": "2021-04-30T08:42:18.459384Z",
    "_value_bpm": 96,
    "_value_timestamp": 1619772138455
  },
```

- **Monitoring containers**

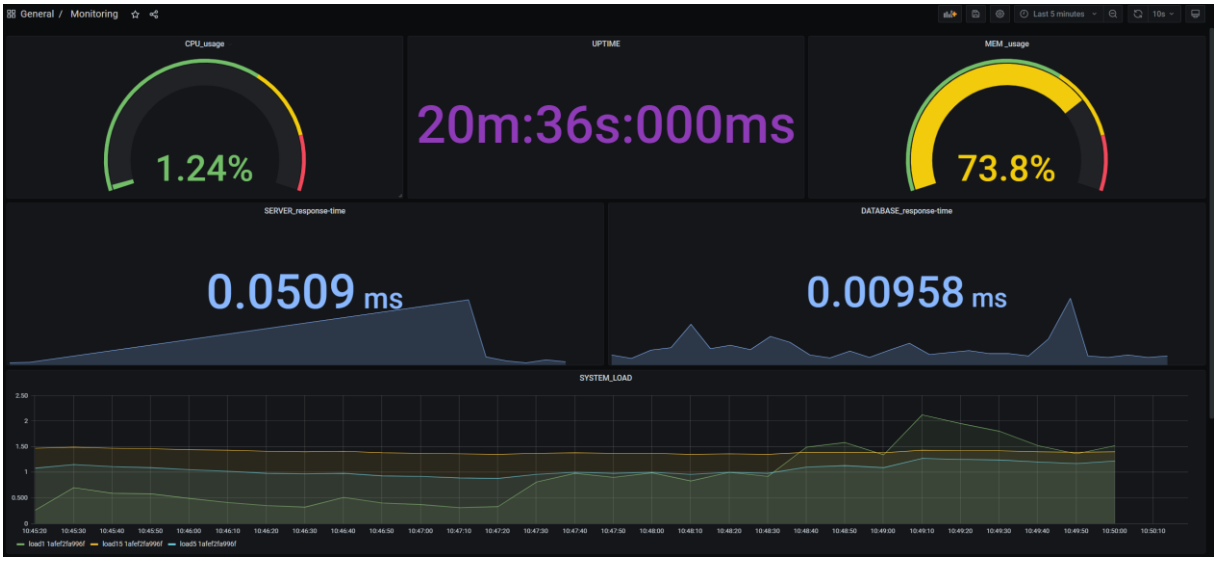
The second thing I did was adding the monitoring containers to the setup. This way, I was able to record, send and visualize the metrics of the RPI server. These are now the containers running on the server:

```
casper@ubuntu:~/server$ docker ps
CONTAINER ID   IMAGE              COMMAND                  CREATED       STATUS                    PORTS                              NAMES
1afef2fa996f   telegraf           "/entrypoint.sh tele…"   8 days ago    Up 13 minutes             8092/udp, 8125/udp, 8094/tcp       telegraf
d1e66e51ca2c   grafana/grafana    "/run.sh"                9 days ago    Up 13 minutes             0.0.0.0:3000->3000/tcp             grafana
a45ce5075aeb   nodered/node-red   "npm --no-update-not…"   5 weeks ago   Up 13 minutes (healthy)   0.0.0.0:1880->1880/tcp             nodered
6fdb252d6105   influxdb:2.0.4     "/entrypoint.sh infl…"   5 weeks ago   Up 13 minutes             0.0.0.0:8086->8086/tcp             influxdb
casper@ubuntu:~/server$
```

I added the Telegraf container to gather and send metrics to the database. The Grafana container is used to visualize everything, as visible on the next screenshots:

(overview of the generated ECG data)

(overview of the metrics of the server)



These extra containers were not added to the final setup.

3 Sprint retrospective

The second part of this report contains our retrospective sprint. Similar to last retro, we used a tool called Miro where you can setup a board and everyone can fill in what they would like to say.

In the screenshot below you can see our board. If the text is too small, be sure to go this our repo where we've also included an image of this board. (see docs/sprint4)