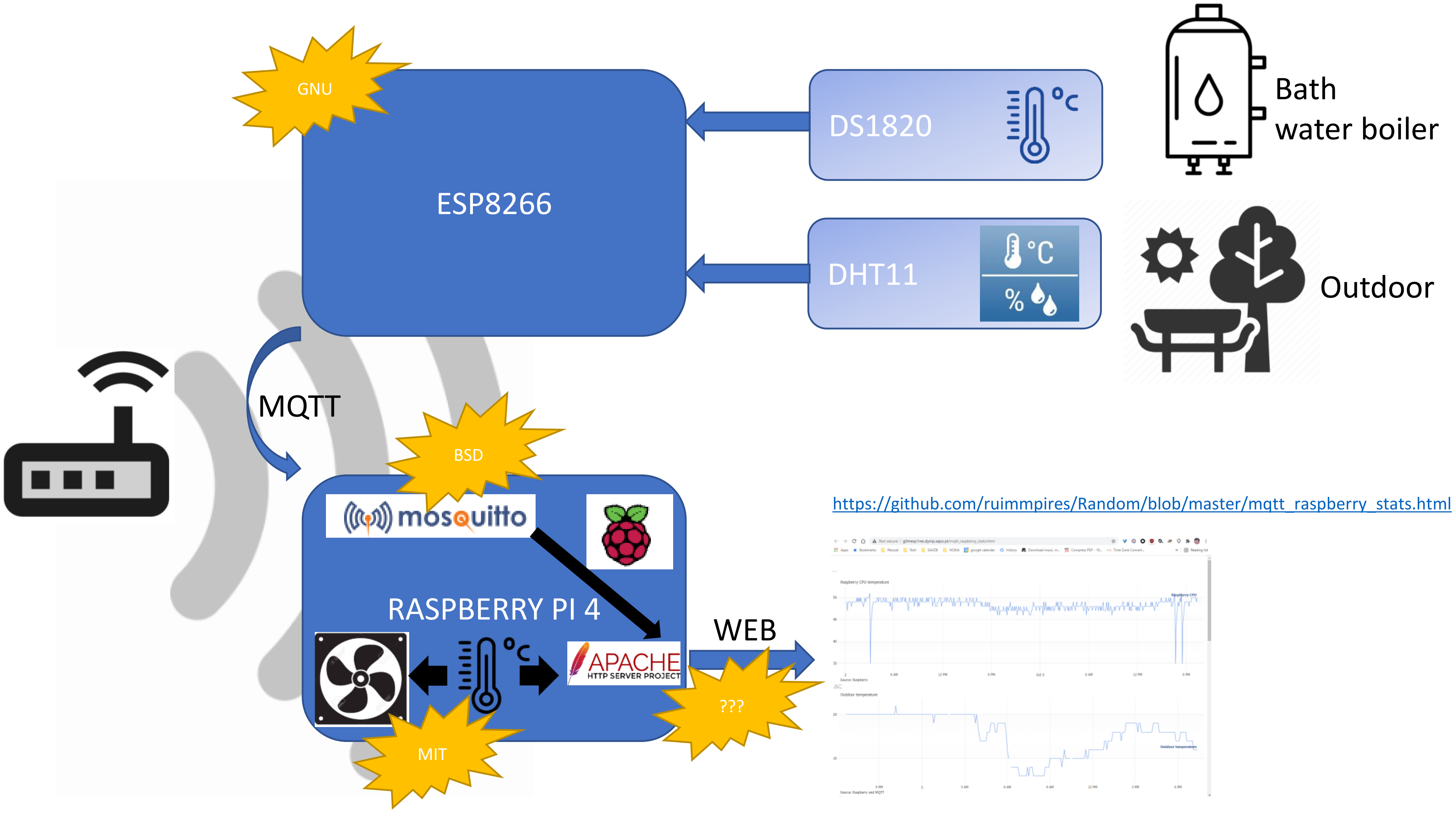
mqtt_raspberry_stats

- Architecture
- Licenses
- code



https://github.com/ruimmpires/Random/blob/master/mqtt_raspberry_stats.html

Code in ESP8266

https://github.com/ruimmpires/Random/blob/master/Hum_Temp__DS18B20_and_DHT11_mqtt_v3.ino

Libraries:

All licensed as GNU Lesser General Public License as published by the Free Software Foundation:

- ESP8266WiFi.h
- PubSubClient.h
- time.h
- DS18B20.h, including DS18B20.cpp

Code:

Sends via MQTT 3 strings:

- Outdoor humidity in %
- Outdoor temperature in ºC
  - Includes a trailing 0 for temperatures below 10.
- Water temperature in ºC
- Reads data and sends via MQTT each 10s
- Every 100 cycles does a self WiFi reconnect for improved stability

Code in Raspberry for MQTT

https://github.com/ruimmpires/Random/blob/master/mqttmon.sh

Libraries:

Mosquitto: Eclipse Mosquitto is an open source (EPL/EDL licensed) message broker that implements the MQTT protocol. The receiver of EPL-licensed programs can use, modify, copy and distribute the work and modified versions, in some cases being obligated to release their own changes.

mosquitto_sub: BSD license.

Code (bash script):

- Every 15mins
- subscribes to the 3 topics (mosquitto_sub), one for each sensor.
- upon each subscription outputs the values to a temporary file
    - strips and cleans the temporary file
    - gets the date in the desired format
    - outputs to a new csv with ,,,
- kills mosquito subscriber process (does not look good, but works fine)
    - Example: 2020/10/23-19:45:16,13,91,38.3

Code in Raspberry Pi for Raspberry CPU Temperature

https://github.com/ruimmpires/Random/blob/master/tempmon.sh

Libraries:
vcgencmd: command line tool designed by Broadcom used on the Raspberry Pi. Apache License 2.0. A permissive license whose main conditions require preservation of copyright and license notices.
RPi.GPIO: to control the CPU fan, MIT license?

Code (bash script):

- Gets the CPU temperature with rhw *vcgencmd* command
- Cleans-up the output
- Creates a timestamp
- If temperature over 55ºC, turn on the fan for one minute
- outputs to a new csv
  - Example: 2021/10/03-19:25:02,49

/home/pi/Desktop/fanon.py

```
try:
    import RPi.GPIO as GPIO
    import time
    import subprocess
    import os
except RuntimeError:
    print("Error importing RPi.GPIO!  This is probably because
you need superuser privileges. You can achieve this by using
'sudo' to run your script")

GPIO.setwarnings(False)
GPIO.setmode(GPIO.BOARD)
GPIO.setup(5, GPIO.OUT)
GPIO.output(5, True)
#subprocess.call(['sh', './print_temp.sh'])
print ("fan on")
```

/home/pi/Desktop/fanoff.py

```
try:
    import RPi.GPIO as GPIO
    import time
    import subprocess
    import os
except RuntimeError:
    print("Error importing RPi.GPIO!  This is
probably because you need superuser privileges. You
can achieve this by using 'sudo' to run your script")

GPIO.setwarnings(False)
GPIO.setmode(GPIO.BOARD)
GPIO.setup(5, GPIO.OUT)
GPIO.output(5, False)
#subprocess.call(['sh', './print_temp.sh'])
```

Code in Raspberry Pi for final tweaks

Libraries:
none

CRONTAB:

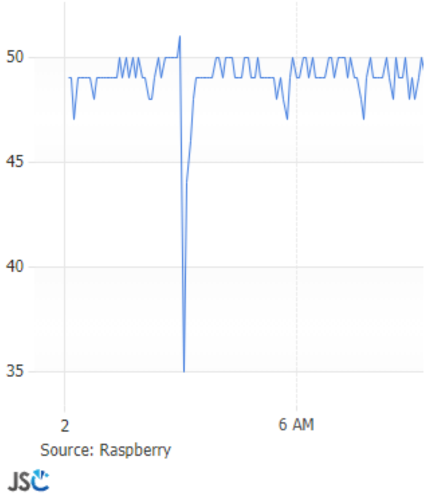
```
*/5 * * * * /home/pi/Desktop/tempmon.sh >> /home/pi/crontab.log  2>&1
*/5 * * * * /home/pi/Desktop/mqttmon.sh >> /home/pi/crontab.log  2>&1
1 0 * * * /home/pi/Desktop/trimcsv.sh
```

TRIMCSV

```
#!/bin/bash
printf "Trimming the CSVs older than "date -d "2 days ago" "+20%y/%m/%d-%T"
cp /var/www/html/mqttstats.csv /var/www/html/mqttstats.csv.back
cp /var/www/html/rtemp.csv /var/www/html/rtemp.csv.back
awk -v d="$(date -d "1 days ago" "+20%y/%m/%d-%T")" '$1 $2 > d' /var/www/html/mqttstats.csv.back >
/var/www/html/mqttstats.csv
awk -v d="$(date -d "1 days ago" "+20%y/%m/%d-%T")" '$1 $2 > d' /var/www/html/rtemp.csv.back >
/var/www/html/rtemp.csv
```

Code in Raspberry Pi for the web server

https://github.com/ruimmpires/Random/blob/master/mqtt_raspberry_stats.html



Running an apache webserver:
apache2 -v
Server version: Apache/2.4.38 (Raspbian)
Server built: 2020-08-25T20:08:29

Licenses:

- jscharting.js. Free licensed? Brand in the homepage?

The webpage runs three scripts:
<script src="https://code.jscharting.com/2.9.0/jscharting.js"></script>
<script src="js/rasp.js"></script>
<script src="js/mqtt.js"></script>

- download the csv (yes, this was not needed as the files are local, but it works fine)
- transform the csv in a series
- render the series in charts

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
<html>
<head>
 <title>Statistics</title>
</head>
<body lang="en-US" link="#000080" vlink="#800000" dir="ltr"><p><br/><br/>
			<div id="chartDiv_raspTemp" style="width:100%; height:400px; margin:0 auto;"></div>
			<div id="chartDiv_outdoorTemp" style="width:100%; height:400px; margin:0 auto;"></div>
			<div id="chartDiv_outdoorHum" style="width:100%; height:400px; margin:0 auto;"></div>
			<div id="chartDiv_waterTemp" style="width:100%; height:400px; margin:0 auto;"></div>
			<div id="chartDiv_power" style="width:100%; height:400px; margin:0 auto;"></div>
			<div id="chartDiv_potential" style="width:100%; height:400px; margin:0 auto;"></div>
</body>
<script src="https://code.jscharting.com/2.9.0/jscharting.js"></script>
<script src="js/mqtt.js"></script>
<script src="js/rasp.js"></script>
</html>
```