USER GUIDE

Project: Smart Temperature Monitoring System

Designed by Muhammad Usman | usman.pak@outlook.com

Features implemented.

1. Continuous temperature monitoring.
2. REST API implementation for remote access of device.
3. Remote access of current temperature (Read only)
4. User configurable alarm threshold of temperature. (Read/Write)
5. Remote access of red LED. (Read/Write)
6. Alarm Indication via Blue LED, when temperature goes higher than threshold.
7. Temperature threshold storage in “temperature-sensor-config.txt”

[NOTE: Temperature is in Celsius only]

RESTful API for Remote Access

Server address for current implementation : http://localhost:8080

Use GET method for reading/getting information and POST method for setting the value of sensor or operate the led

1. Read current temperature:
   /sensor/temperature/current
2. Read alarm threshold of temperature:
   /sensor/temperature/threshold
3. Set alarm threshold of temperature:
   /sensor/temperature/threshold
   Set value in body of message “*value=val*”
4. Turn red led on
   /led/red
   Set value in body of message “*value=1*”
5. Turn red led off
   /led/red
   Set value in body of message “*value=0*”

Test Procedure

Server runs on localhost at port 8080

Executable

Git location: https://github.com/usman-pak1991/usman/
System Location: /usman/
Files: Fib_Task

Executation procedure

cd /usman/
sudo git pull https://github.com/usman-pak1991/usman/
sudo chmod 777 Fib_Task
sudo ./Fib_Task

```
pi@raspberrypi:/usman $ sudo chmod 777 Fib_Task
pi@raspberrypi:/usman $ ./Fib_Task
/usman/2021-04-
01 15:48:05  I mongoose.c:3154:mg_listen 1 accepting on http://localhost:8080
```

Snapshot of Server

CURL command to test the REST API

Note: Pleases type the commands in remote terminal and do not copy/past them.

sudo curl -d "value=26.6" -X POST http://localhost:8080/sensor/temperature/threshold
sudo curl -d "value=22.5" -X POST http://localhost:8080/sensor/temperature/threshold
sudo curl -d "value=1" -X POST http://localhost:8080/led/red
sudo curl -d "value=0" -X POST http://localhost:8080/led/red
sudo curl -i -X GET http://localhost:8080/sensor/temperature/threshold
sudo curl -i -X GET http://localhost:8080/sensor/temperature/current

CURL commands and HTTP Server Responses

```
pi@raspberrypi:/ $ sudo curl -i -
X GET http://localhost:8080/sensor/temperature/threshold
HTTP/1.1 200 OK
Content-Type: text/html
Content-Length: 4

10.3

pi@raspberrypi:/ $ sudo curl -i -
X GET http://localhost:8080/sensor/temperature/current
HTTP/1.1 200 OK
```

```
Content-Type: text/html
Content-Length: 4

23.0

pi@raspberrypi:/ $ sudo curl -i -d "value=26.6" -
X POST http://localhost:8080/sensor/temperature/threshold
HTTP/1.1 200 OK
Content-Type: text/html
Content-Length: 1

pi@raspberrypi:/ $ sudo curl -i -
X GET http://localhost:8080/sensor/temperature/threshold
HTTP/1.1 200 OK
Content-Type: text/html
Content-Length: 4

26.6

pi@raspberrypi:/ $ sudo curl -i -d "value=22.5" -
X POST http://localhost:8080/sensor/temperature/threshold
HTTP/1.1 200 OK
Content-Type: text/html
Content-Length: 1

pi@raspberrypi:/ $ sudo curl -i -
X GET http://localhost:8080/sensor/temperature/threshold
HTTP/1.1 200 OK
Content-Type: text/html
Content-Length: 4

22.5

pi@raspberrypi:/ $ sudo curl -i -d "value=0" -
X POST http://localhost:8080/led/red
HTTP/1.1 200 OK - RED LED-OFF
Content-Type: text/html
Content-Length: 1

pi@raspberrypi:/ $ sudo curl -i -d "value=1" -
X POST http://localhost:8080/led/red
HTTP/1.1 200 OK - RED LED-ON
Content-Type: text/html
Content-Length: 1
```