



Pușcalău Robert George

Name: Control Centrală termică de la distanță

Elevator Pitch: Acest proiect își propune dezvoltarea unui sistem ce permite controlul temperaturii unui imobil de la distanță, printr-o interfață web.



Cover Image:

Story: Personal, am nevoie de un dispozitiv care să joace rol de termostat în apartamentul în care locuiesc. Am decis să dezvolt singur sistemul acesta deoarece mi se pare un proiect accesibil pe bază de Raspberry Pi, ce poate fi extins foarte mult prin software. Cred că aplicațiile din ziua de azi sunt limitate mult mai mult din punct de vedere software decât hardware.

Acest proiect este destinat persoanelor ce doresc să își automatizeze locuința folosind propriile cunoștințe după bunul plac.

Dispozitivul își propune să implementeze o interfață web prin care locuitorii unui imobil pot controla temperatura locuinței pe baza unor senzori de temperatură plasați în încăperi și raportul energiei termice generate de centrala termică și sistemul de încălzire, afișând informații relevante în timp real.

Name: Control Centrală termică de la distanță

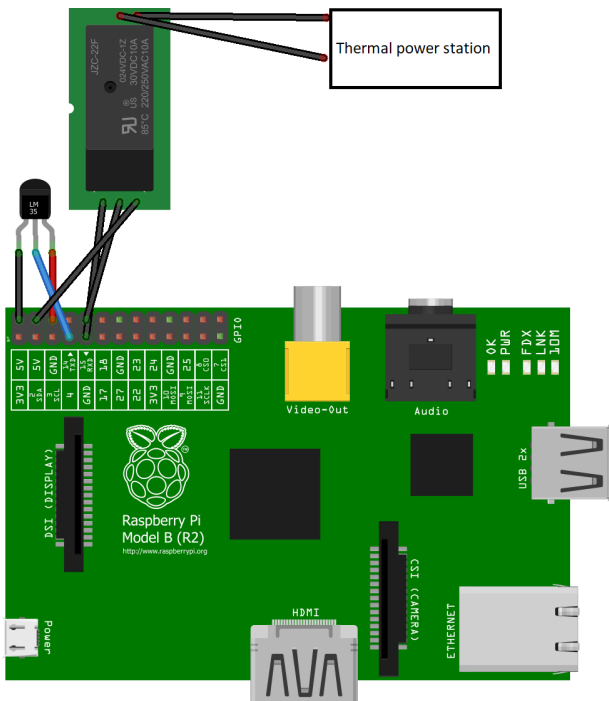
Link proiect hackster.io <https://www.hackster.io/347580/home-made-thermostat-for-thermal-plant-7a96e3>

Elevator Pitch: Acest proiect îți propune dezvoltarea unui sistem care permite controlul temperaturii unui imobil de la distanță, printr-o interfață web.

Components and apps:

- > Rpi4 Model B
- > Senzor de temperatura ds18b20
- > Modul releu SRD 5v ce actioneaza ca switch pentru centrala
- > Server Python(logica programului)
- > Server web(NodeJS)

Schematics:



fritzing

(Din pacate, layout-ul pinilor din imaginea de mai sus nu sunt cei de la rpi4, imaginea avand doar scop orientativ, explicatiile exacte urmand mai jos.)

Senzorul de temperatura ds18b20 l-am conectat la Raspberry Pi cu cei trei pini ai sai:

Ground - Pin06

VCC - Pin02

Data - Pin07 (GPIO14)

Modulul releu l-am conectat la Raspberry Pi cu cei trei pini ai sai:

Ground - Pin09

VCC - Pin04

Data - Pin40 (GPIO21)

Centrala a fost legata la modulul releu pe modul normal deschis, releul functionand ca si switch.

Raspberry Pi 4 B J8 GPIO Header			
Pin#	NAME	Pin#	NAME
01	3.3v DC Power	02	DC Power: 5v
03	GPIO-2 (SDA1, I2C)	04	DC Power: 5v
05	GPIO-3 (SCL1, I2C)	06	Ground
07	GPIO-4 (GPCLK0)	08	(TXD0, UART) GPIO14
09	Ground	10	(RXD0, UART) GPIO15
11	GPIO17	12	(PWM0) GPIO18
13	GPIO27	14	Ground
15	GPIO22	16	GPIO23
17	3.3v DC Power	18	GPIO24
19	GPIO10 (SPI0_MOSI)	20	Ground
21	GPIO-9 (SPI0_MISO)	22	GPIO25
23	GPIO11 (SPI0_CLK)	24	(SPI0_CEO_N) GPIO-8
25	Ground	26	(SPI0_CEI_N) GPIO-7
27	GPIO-8 (SDA0, I2C)	28	(SCL0, I2C) GPIO-11
29	GPIO-5	30	Ground
31	GPIO-6	32	(PWM0) GPIO12
33	GPIO13 (PWM1)	34	Ground
35	GPIO19	36	GPIO16
37	GPIO26	38	GPIO20
39	Ground	40	GPIO21

Raspberry Pi 4 B J14 PoE Header			
Pin#	NAME	Pin#	NAME
01	TR+1	02	TR-0
03	TR+3	04	TR-2

Pinout Grouping Legend

- Inter-Integrated Circuit Serial Bus
- Unassigned/Un-Allocated GPIO
- Reserved for EEPROM
- Serial Peripheral Interface Bus
- Universal Asynchronous Receiver/Transmitter

Rev. 2
10/16/2018 ccx
www.element14.com/RaspberryPi

Sunt doua aplicatii:

Serverul Python primeste comenzi dintr-o baza de date, prelucreaza ultima comanda si in functie de comanda primita(on-desired temperature / off) si temperatura din camera(achizitionata de senzorul de temperatura), acesta ia deciziile necesare pentru centrala si trimite statusul actual server-ului web(nodejs) printr-un fisier json.

Serverul web are rol ca sistemul sa fie disponibil de la distanta pentru locuitorii apartamentului, asigurand cu usurinta acces la acesta si posibilitatea de a trimite comenzi si a verifica statusul actual al centralei.

Cod aferent serverului Nodejs(din cauza dimensiunii mari l-am urcat pe github):

<https://github.com/gabrielviorelciobanasu/proiect-sm>

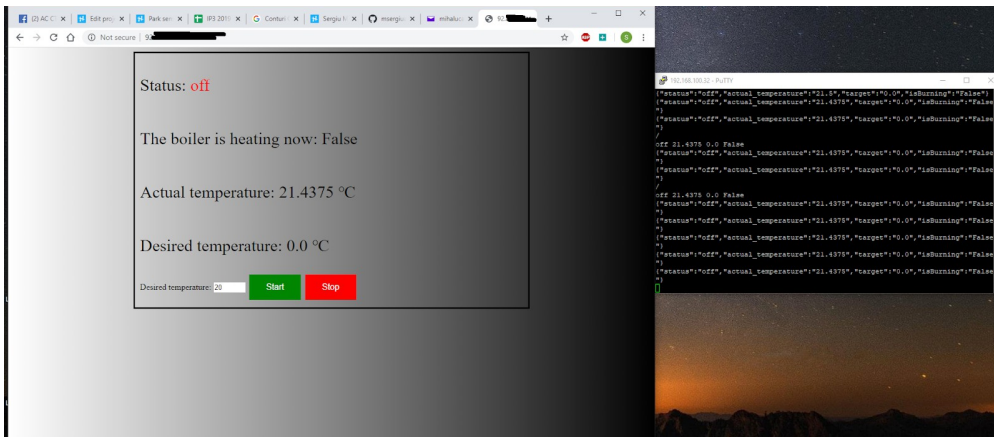
Cod aferent serverului Python:

```
main.py x
1  #import RPi.GPIO as GPIO
2  #from w1thermsensor import W1ThermSensor
3  import ...
4
5
6
7
8  # ----- INIT -----
9  channel=21
10 #GPIO.setmode(GPIO.BCM)
11 #GPIO.setup(channel,GPIO.OUT)
12 #sensor=W1ThermSensor()
13 isBurning = False
14 pin = 21
15
16 con=MySQLdb.connect(user='auto',passwd='',host='localhost',db='sm')
17 c=con.cursor()
18
19
20
21 # ----- CLASSES -----
22 class Command:
23     def __init__(self, id, data, tip, target):
24         self.id = id
25         self.data = data
26         self.tip = tip
27         self.target = target
28
29
30
31 # ----- FUNCTIONS -----
32 def getCommand():
33     con.commit()
34     query=("SELECT id, data, tip,comanda, target FROM commands where (select max(id) from commands) = id")
35     c.execute(query)
36     for(id,data,tip_comanda,target) in c:
37         command=Command(id,data,tip_comanda,target)
38
39     return command
40 def readTemp():
41     #return sensor.get_temperature()-5
42     return 27
```

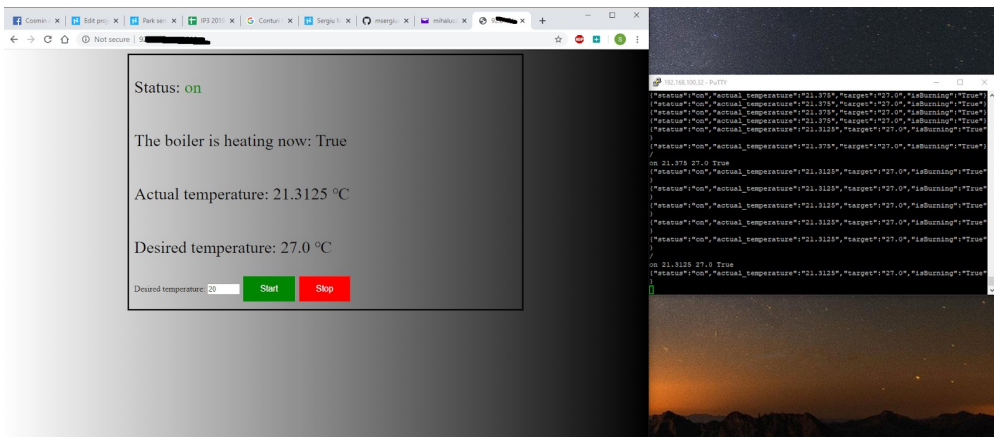
```
main.py x
41 #return_sensor.get_temperature()-5
42 return 27
43
44 def maintainCurrentTemp():
45     global pin
46     global currentCommand
47     global isBurning
48
49     if(currentCommand.tip == "on"):
50         if(readTemp() > currentCommand.target+1):
51             #GPIO.output(pin, GPIO.LOW)
52             isBurning = False
53         if(readTemp() <= currentCommand.target-1):
54             #GPIO.output(pin, GPIO.HIGH)
55             isBurning = True
56     if(currentCommand.tip == "off"):
57         #GPIO.output(pin, GPIO.LOW)
58         isBurning = False
59
60 # ----- MAIN -----
61 def main():
62     global currentCommand
63     currentCommand = getCommand()
64     while(True):
65         # ----- MAIN LOOP -----
66         # Rutina daca avem comanda noua
67         command = getCommand()
68         if(command.id > currentCommand.id):
69             currentCommand = command
70
71         # Rutina de pastrat temperatura
72
73         maintainCurrentTemp()
74         aux = readTemp()
75         data = '{"status":"' + currentCommand.tip + '", "actual_temperature":"' + str(aux) + '" \
76             | "target":"' + str(currentCommand.target) + '", "isBurning":"' + str(isBurning) + '"}'
77         with open('/home/pi/projects/sm_nodejs/proiect-sm/information.json', 'w') as outJson:
78             outJson.write(data)
79         print(data)
80
81
82
83 if __name__ == "__main__":
84     main()
85
86
87 con.close()
```

Demo Functionalitate:

Intru de unde doresc la adresa ip specifica si pe portul asociat aplicatii. In acest caz centrala nu incalzeste deoarece(este vara :).



Deoarece imi este frig, trimit comanda de incalzire.(Se observa cum serverul Python primeste comanda, centrala porneste - isBurning -- iar interfata web este actualizata). Micile fluctuatii de temperatura sunt din cauza senzorului ieftin(achizitionat acum din motive de test).



Sisteme cu microprocesoare Examen 24 iunie

Nume student: Puscalau Robert-George

Grupa: 1309B

Raspundeti personalizat la urmatoarele intrebari:

1. Ce componente hardware-software ati utilizat pentru implementarea proiectului la SM cu titlul termostat de centrala actionat prin interfata web ?

Am utilizat o placuta raspberry pi ca principala componenta. La rpi am conectat un senzor de temperatura DS18B20 si un releu de 5V.

Ca software, am utilizat python pentru logica programului de pornire oprire centrala automat, iar pentru interfata web am folosit node.js.

De asemenea, comenzile date de pe interfata web se salveaza intr-o baza de date MariaDB(mysql).

2. In ce consta cea mai importanta realizare din proiect?

Faptul ca dispozitivul embeded inchide centrala singur in momentul in care s-a atins temperatura

dorita si o porneste inapoi cand temperatura a scazut sub un anumit prag.

3. Ce a fost cel mai dificil de facut la proiect?

Interactiunea dintre serverul node.js si procesul python. Am utilizat o baza de date pentru comenzi si un fisier json pentru salvarea statusului camerei, centralei etc.

4. Ce cunostinte si abilitati trebuie sa aiba cineva care vrea sa replice/refaca/evalueze proiectul?

Cunostinte de javascript si interfatare web medii. Cunostinte minime de python si wiringPi.

Cunostinte minime de baze de date.

Cunostinte minime de electronica si electrotehnica. Rabdare, grija si atentie cand dezassembleaza centrala. Abilitatea de a citi datasheeturi simple, dar si complexe(centrala pentru legarea la ea a releului)

5. In lumea reala , cu eventuale completari, la ce s-ar putea utiliza proiectul?

Se poate extinde proiectul in python pentru a primi comenzi de la sistemele de smart home(Alexa, Google Home). De asemenea, trebuie gasita o solutie pentru a avea mai multi senzori de temperatura

in mai multe camere care sa comunice wireless si independent cu rpi de comanda pentru ca uniformiza caldura in locuinta.
