

**Puscalau Robert** - 1309B [robert-george.puscalau@student.tuiasi.ro](mailto:robert-george.puscalau@student.tuiasi.ro)

(poza).

**Mihaluca Sergiu Adrian** - 1305A [mihaluca.sergiu@yahoo.com](mailto:mihaluca.sergiu@yahoo.com)



**Ciobanasu Gabriel Viorel** - 1305A [ciobanasugabriel@yahoo.com](mailto:ciobanasugabriel@yahoo.com)



**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 își propune dezvoltarea unui sistem care permite controlul temperaturii unui imobil de la distanță, printr-o interfață web.

**Cover Image:**



### **Story:**

Din lipsa unui termostat in apartament, am decis să dezvoltăm singuri sistemul acesta deoarece am considerat ca acest proiect este accesibil pe bază de Raspberry Pi și poate fi extins foarte mult prin software.

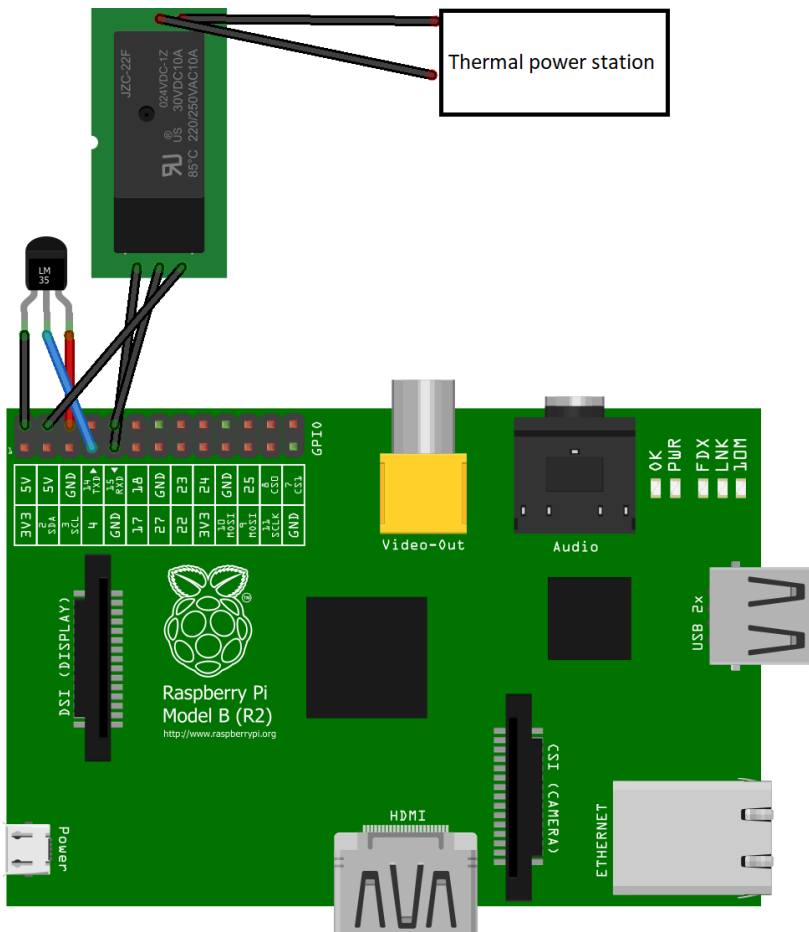
Acest proiect este destinat persoanelor care doresc să își automatizeze locuința folosind propriile cunoștințe după bunul plac.

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.

### **Components and apps:**

- > Rpi4 Model B
- > Senzor de temperatura ds18b20
- > Modul releu SRD 5v ce acționează 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		NAME Pin#
01	3.3v DC Power		DC Power 5v 02
03	GPIO02 (SDA1, I <sup>2</sup> C)		DC Power 5v 04
05	GPIO03 (SCL1, I <sup>2</sup> C)		Ground 06
07	GPIO04 (GPCLK0)		(TXD0, UART) GPIO14 08
09	Ground		(RXD0, UART) GPIO15 10
11	GPIO17		(PWM0) GPIO18 12
13	GPIO27		Ground 14
15	GPIO22		GPIO23 16
17	3.3v DC Power		GPIO24 18
19	GPIO10 (SPI0_MOSI)		Ground 20
21	GPIO09 (SPI0_MISO)		GPIO25 22
23	GPIO11 (SPI0_CLK)		(SPI0_CE0_N) GPIO08 24
25	Ground		(SPI0_CE1_N) GPIO07 26
27	GPIO00 (SDA0, I <sup>2</sup> C)		(SCL0, I <sup>2</sup> C) GPIO01 28
29	GPIO05		Ground 30
31	GPIO06		(PWM0) GPIO12 32
33	GPIO13 (PWM1)		Ground 34
35	GPIO19		GPIO16 36
37	GPIO26		GPIO20 38
39	Ground		GPIO21 40

Raspberry Pi 4 B J14 PoE Header			
Pin#	NAME		NAME Pin#
01	TR01		TR00 02
03	TR03		TR02 04

**Pinout Grouping Legend**

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

Rev 2  
13/06/2019 ccs  
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>

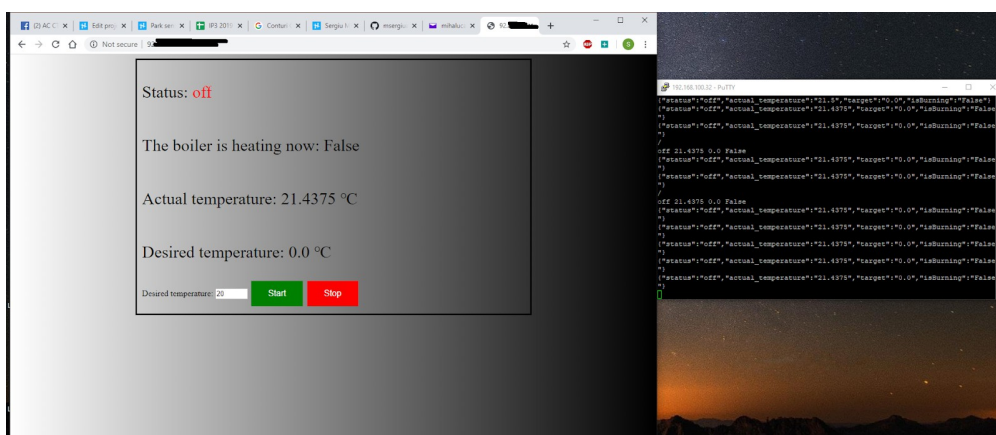
Cod aferent serverului Python:

```
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_senor.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/project-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, centrata porneste - isBurning -- iar interfata web este actualizata). Micile fluctuatii de temperatura sunt din cauza senzorului ieftin(achizitionat acum din motive de test).

