

```

1  # Import libraries
2  import csv
3  import time
4  import socket
5  import RPi.GPIO as GPIO
6  import requests
7  import gspread
8
9  # Import python files
10 from switch import *
11 from temperature import *
12 from light import *
13
14 # Functions
15 def getip():
16     try:
17         s = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
18         s.connect(("8.8.8.8", 80))
19         return s.getsockname()[0]
20     except:
21         return 0
22
23 def TelMess(text):
24     try:
25         base_url = 'https://api.telegram.org/bot5189477795:
AAEVYv_V0PWOicis7RtdYsNIQZFNOMHxIJk/' \
26             'sendMessage?chat_id=-682759305&text={text}'.format(
27                 text=text)
28         requests.get(base_url)
29     except:
30         return 0
31
32
33 # Mysql Parameters
34 ip = 'XXX.XXX.XXX.XXX'
35 port = '3306'
36 database = 'Bakalarka'
37
38 # Setup relay
39 led_y = 6
40 led_r = 13
41 heat_foil_bed = 26
42 heat_foil_liv = 19
43
44 # Setup switches
45 button_1 = 27
46 button_2 = 23

```

```

47 button_3 = 17
48 button_4 = 18
49
50 # Setup DHT Sensors
51 temp_cycle_limit = 10
52
53 # Variables
54 cycle = 0
55 run = 1
56 cycle_but1 = 0
57 cycle_but2 = 0
58 cycle_but3 = 0
59 cycle_but4 = 0
60 relay = [False, False, False, False]
61 online = [2,2,2,2,2,2,2,2,2,2,2,2]
62
63 # Temp setting
64 set_temp_bed = 20
65 set_temp_liv = 23
66 mer_dokument = 'regulace_v2.csv'
67 delta_temp = 0.5
68 time_zone = 3600 # +1 hour to time (Prague)
69 mess_delay = 30 # 30 sec mess delay
70
71 # Switches
72 bedroom_switch = Switch('switch', 'QYKPMdbKNydPTW5k', ip, database,
    port, "loznice", 0, button_1)
73 living_room_switch_1 = Switch('switch', 'QYKPMdbKNydPTW5k', ip,
    database, port, "obyvak1", 0, button_2)
74 living_room_switch_2 = Switch('switch', 'QYKPMdbKNydPTW5k', ip,
    database, port, "obyvak2", 0, button_3)
75
76 #Lights
77 bedroom_light = Light('light', '3yRdaB3r6by5', ip, database, port, "obyvak"
    , 0, led_r)
78 living_room_light = Light('light', '3yRdaB3r6by5', ip, database, port, "loznice"
    , 0, led_y)
79
80 GPIO.setmode(GPIO.BCM)
81
82 # Setup switch to INPUT
83 GPIO.setup(button_1, GPIO.IN, pull_up_down=GPIO.PUD_UP) # Button 1
    for Red led
84 GPIO.setup(button_2, GPIO.IN, pull_up_down=GPIO.PUD_UP) # Button for
    Yellow led
85 GPIO.setup(button_3, GPIO.IN, pull_up_down=GPIO.PUD_UP) # End loop
    button

```

```

86 GPIO.setup(button_4, GPIO.IN, pull_up_down=GPIO.PUD_UP) # Button 1
   for Red led
87
88 # Setup Relay to 0 by default value
89 GPIO.setup(heat_foil_bed, GPIO.IN) # Heating foil 1
90 GPIO.setup(heat_foil_liv, GPIO.IN) # Heating foil 2
91 GPIO.setup(led_r, GPIO.IN) # Led yellow
92 GPIO.setup(led_y, GPIO.IN) # Led red
93
94 DhtSensor1 = DhtSensor('TempSensor', 'Kj#7](J&haH>QYx`', ip, database
   , port, "Obyvak_DHT", 0)
95 DhtSensor2 = DhtSensor('TempSensor', 'Kj#7](J&haH>QYx`', ip, database
   , port, "Loznice_DHT", 0)
96
97 # Setup Google sheet API
98 try:
99     sa = gspread.service_account(filename=".../odevzdání/
service_account_google.json")
100     sh = sa.open("Nastaveni")
101     wks_sett = sh.worksheet("Default setting")
102
103     set_temp_bed = float(wks_sett.acell('B5').value)
104     set_temp_liv = float(wks_sett.acell('B4').value)
105     delta_temp = float(wks_sett.acell('B3').value)
106     mess_delay = float(wks_sett.acell('B6').value)
107 except:
108     print("Google sheets not working")
109 # Connect user into database
110 try:
111     DhtSensor1.conn_to_database()
112     DhtSensor2.conn_to_database()
113     bedroom_switch.conn_to_database()
114     living_room_switch_1.conn_to_database()
115     living_room_switch_2.conn_to_database()
116     bedroom_light.conn_to_database()
117     living_room_light.conn_to_database()
118     online[0] = 1
119 except:
120     online[0] = 0
121
122 seconds = time.time()
123 local_time = time.ctime(seconds + time_zone) # 3600 timezone to Prague
124 next_measurement = time.ctime(seconds + time_zone + mess_delay) # 30
   sec delay for measurement
125 print("Local time:", local_time)
126
127 # print("Ip address: ",getip())

```

```

128 TelMess("Aplikace Smart-Home byla spuštěna")
129 TelMess("Ip adresa RPI je:" + str(getip()))
130
131 # Measurement to csv file herader
132 header = ['TIME', 'TEMP_OB', 'TEMP_LOZ', 'REL_OB', 'REL_LOZ']
133 with open(mer_dokument, 'w', encoding='UTF8', newline='') as f:
134     writer = csv.writer(f)
135     writer.writerow(header)
136
137 # Lists
138
139 Light_list = [living_room_light, bedroom_light]
140 Switch_list = [bedroom_switch, living_room_switch_1, living_room_switch_2]
141
142 # Super loop
143 while run == 1:
144     cycle = cycle + 1
145
146     # Button repair
147     if cycle_but1 > 50000:
148         cycle_but1 = 50000
149     if cycle_but2 > 50000:
150         cycle_but2 = 50000
151     if cycle_but3 > 50000:
152         cycle_but3 = 50000
153     if cycle_but4 > 50000:
154         cycle_but4 = 50000
155
156     # Control switches
157     if GPIO.input(bedroom_switch.gpio_port) == 0:
158         cycle_but1 = cycle_but1 + 1
159     else:
160         cycle_but1 = 0
161
162     if GPIO.input(living_room_switch_1.gpio_port) == 0:
163         cycle_but2 = cycle_but2 + 1
164     else:
165         cycle_but2 = 0
166
167     if GPIO.input(living_room_switch_2.gpio_port) == 0:
168         cycle_but3 = cycle_but3 + 1
169     else:
170         cycle_but3 = 0
171
172     if GPIO.input(button_4) == 0:
173         cycle_but4 = cycle_but4 + 1
174     else:

```

```

175     cycle_but4 = 0
176
177     if cycle_but1 > 20000:
178         if bedroom_switch.actual_state != 1:
179             bedroom_switch.actual_state = 1
180             online[1] = bedroom_switch.update_database_state()
181     else:
182         if bedroom_switch.actual_state != 0:
183             bedroom_switch.actual_state = 0
184             online[2] = bedroom_switch.update_database_state()
185
186     if cycle_but2 > 20000:
187         if living_room_switch_1.actual_state != 1:
188             living_room_switch_1.actual_state = 1
189             online[3] = living_room_switch_1.update_database_state()
190     else:
191         if living_room_switch_1.actual_state != 0:
192             living_room_switch_1.actual_state = 0
193             online[4] = living_room_switch_1.update_database_state()
194
195     if cycle_but3 > 20000:
196         if living_room_switch_2.actual_state != 1:
197             living_room_switch_2.actual_state = 1
198             online[5] = living_room_switch_2.update_database_state()
199     else:
200         if living_room_switch_2.actual_state != 0:
201             living_room_switch_2.actual_state = 0
202             online[6] = living_room_switch_2.update_database_state()
203
204     if cycle_but4 > 20000: # Program END
205         run = 0
206     # Light set on/off
207     if bedroom_switch.actual_state == 1:
208         if not relay[0]:
209             relay[0] = True
210             bedroom_light.actual_state = 1
211             online[7] = bedroom_light.update_datab_state()
212     else:
213         if relay[0]:
214             relay[0] = False
215             bedroom_light.actual_state = 0
216             online[8] = bedroom_light.update_datab_state()
217
218     if living_room_switch_1.actual_state != living_room_switch_2.actual_state
219     :
219         if not relay[1]:
220             living_room_light.actual_state = 1

```

```

221         online[9] = living_room_light.update_datab_state()
222         relay[1] = True
223     else:
224         if relay[1]:
225             living_room_light.actual_state = 0
226             online[10] = living_room_light.update_datab_state()
227             relay[1] = False
228
229     if local_time >= next_measurement:
230
231         # new time for measurement
232         next_measurement = time.ctime(seconds + time_zone + mess_delay)
233         print("Another measurement will be in: ", str(next_measurement))
234
235         # Measure temperature
236         DhtSensor1.mess_temperature()
237         DhtSensor2.mess_temperature()
238         # Living room switch
239         if DhtSensor2.temperature > set_temp_liv + delta_temp:
240             GPIO.setup(heat_foil_liv, GPIO.IN)
241             relay[3] = False
242         elif DhtSensor2.temperature < set_temp_liv - delta_temp:
243             GPIO.setup(heat_foil_liv, GPIO.OUT)
244             relay[3] = True
245         # Bedroom switch
246         if DhtSensor1.temperature > set_temp_bed + delta_temp:
247             GPIO.setup(heat_foil_bed, GPIO.IN)
248             relay[2] = False
249         elif DhtSensor1.temperature < set_temp_bed - delta_temp:
250             GPIO.setup(heat_foil_bed, GPIO.OUT)
251             relay[2] = True
252
253         TelMess("Teplota v ložnici: " + str(DhtSensor1.temperature) + "°C \n
Teplota v obýváku: " +
254             str(DhtSensor2.temperature) + "°C" + "\nNastavení teplot pro
relé: \nLožnice: " +
255             str(set_temp_bed) + "°C \nObývánk: " + str(set_temp_liv) + "°C\n
Delta: " + str(delta_temp) +
256             "°C \nStav osvětlení: \nLožnice: " + str(relay[0]) + "\nObývánk: "
+ str(relay[1]) +
257             "\nStav topných folií: \nLožnice: " + str(relay[2]) + "\nObývánk: "
+ str(relay[3]))
258
259         online[11] = DhtSensor1.add_temp_database()
260         online[12] = DhtSensor2.add_temp_database()
261
262         print(online)

```

```

263
264     data = [local_time, DhtSensor2.temperature, DhtSensor1.temperature,
    relay[2], relay[3]]
265     with open(mer_dokument, 'a', encoding='UTF8', newline='') as f:
266         writer = csv.writer(f)
267         writer.writerow(data)
268
269     # Check database connection
270     if 0 in online:
271         print("Databáze není připojena")
272         try:
273             DhtSensor1.conn_to_database()
274             DhtSensor2.conn_to_database()
275             bedroom_switch.conn_to_database()
276             living_room_switch_1.conn_to_database()
277             living_room_switch_2.conn_to_database()
278             bedroom_light.conn_to_database()
279             living_room_light.conn_to_database()
280             print("Databáze opět připojena")
281         except:
282             print("Nepodařilo se databázi připojit")
283
284     else:
285         print("Databáze je připojena")
286
287     # Update data from and to google sheets
288     try:
289         # update from
290         set_temp_bed = float(wks_sett.acell('B5').value)
291         set_temp_liv = float(wks_sett.acell('B4').value)
292         delta_temp = float(wks_sett.acell('B3').value)
293         mess_delay = float(wks_sett.acell('B6').value)
294
295         # update to
296
297         wks_sett.update('F2', '{teplota} °C'.format(teplota = DhtSensor1.
    temperature))
298         wks_sett.update('F3', '{teplota} °C'.format(teplota=DhtSensor2.
    temperature))
299         wks_sett.update('F4', '{stav}'.format(stav=relay[0]))
300         wks_sett.update('F5', '{stav}'.format(stav=relay[1]))
301         wks_sett.update('H2', '{stav}'.format(stav=relay[3]))
302         wks_sett.update('H3', '{stav}'.format(stav=relay[2]))
303         wks_sett.update('H5', '{time}'.format(time=local_time))
304         wks_sett.update('H6', '{IP}'.format(IP=getip()))
305     except:
306         print("Google sheets not working")

```

```
307
308     if cycle == 50000: # End of Cycle
309         seconds = time.time()
310         local_time = time.ctime(seconds + time_zone)
311         cycle = 0
312
313 #End of program
314 GPIO.cleanup()
315 TelMess("Aplikace Smart-Home byla ukončena")
316 try:
317     DhtSensor1.close_database()
318     DhtSensor2.close_database()
319     bedroom_switch.close_database()
320     living_room_switch_1.close_database()
321     living_room_switch_2.close_database()
322     living_room_light.close_database()
323     bedroom_light.close_database()
324 except:
325     print("Nebylo správně ukončeno")
```



```
1 from datab_con import *
2 import RPi.GPIO as GPIO
3
4 class Light(database):
5     def __init__(self, user, password, host, name_of_database, port, location,
        actual_state, gpio_port):
6         super().__init__(user, password, host, name_of_database, port)
7         self.location = location
8         self.actual_state = actual_state
9         self.gpio_port = gpio_port
10
11     def update_datab_state(self):
12         if self.actual_state == 1:
13             GPIO.setup(self.gpio_port, GPIO.OUT)
14         else:
15             GPIO.setup(self.gpio_port, GPIO.IN)
16
17         sql = "UPDATE `Bakalarka`.`lights` SET state = {state} WHERE
        location={light_name}".format(state=self.actual_state,
18                                     light_name=self.
        location)
19         try:
20             self.insert_to_database(sql)
21             return 1
22         except:
23             return 0
```

```
1  from datab_con import *
2
3
4  class Switch(database):
5      def __init__(self, user, password, host, name_of_database, port, location,
        actual_state, gpio_port):
6          super().__init__(user, password, host, name_of_database, port)
7          self.location = location
8          self.actual_state = actual_state
9          self.gpio_port = gpio_port
10
11     def get_actual_state(self):
12         sql = "SELECT * FROM akalarka.switches WHERE location={
        name_switch}".format(name_switch=self.location)
13         online_state_switch = self.select_from_database(sql)
14         self.actual_state = int(online_state_switch[0][2])
15         print(self.actual_state)
16
17     def update_database_state(self):
18         sql = "UPDATE `Bakalarka`.`switches` SET state = {state} WHERE
        name_switch={switch_name}".format(state=self.actual_state,
19                                         switch_name=self.
        location)
20         try:
21             self.insert_to_database(sql)
22             return 1
23         except:
24             return 0
```

```
1  import mysql.connector
2
3  class database:
4
5      def __init__(self, user, password, host, name_of_database, port):
6          self.user = user
7          self.password = password
8          self.host = host
9          self.name_of_database = name_of_database
10         self.port = port
11
12     def conn_to_database(self):
13         print("Attempting to connect a user: " + self.user)
14         try:
15             global my_database
16             my_database = mysql.connector.connect(
17                 host=str(self.host),
18                 user=str(self.user),
19                 password=str(self.password),
20                 database = str(self.name_of_database),
21                 port = str(self.port)
22             )
23             print("Connect to database was successful")
24         except mysql.connector.Error as e:
25             print(e)
26
27     def insert_to_database(self,sql):
28         cursor = my_database.cursor()
29         cursor.execute(sql)
30         my_database.commit()
31
32     def select_from_database(self,sql):
33         try:
34             cursor = my_database.cursor()
35             cursor.execute(sql)
36             return cursor.fetchall()
37         except:
38             return 10
39
40     def close_database(self):
41         try:
42             user = self.user
43             my_database.close()
44             print("Connections is closed for " + user)
45             return 1
46         except:
47             print("Connections isn't closed for " + user)
```

48       **return 0**

49

```

1  from datab_con import *
2
3  import board
4  import adafruit_dht
5
6  dhtDevice_obyvak = adafruit_dht.DHT22(board.D20, use_pulseio=False)
7  dhtDevice_loznice = adafruit_dht.DHT22(board.D21, use_pulseio=False)
8
9
10 class DhtSensor(database):
11
12     def __init__(self, user, password, host, name_of_database, port, sensorID
, temperature):
13         super().__init__(user, password, host, name_of_database, port)
14         self.sensorID = sensorID
15         self.temperature = temperature
16
17     def mess_temperature(self):
18         temp = 0
19         divide = 0
20         for i in range(10):
21             if self.sensorID == "Obyvak_DHT":
22                 try:
23                     temp += round(dhtDevice_obyvak.temperature, 3)
24                     divide += 1
25                 except:
26                     temp = 0
27                     divide = 1
28
29             if self.sensorID == "Loznice_DHT":
30                 try:
31                     temp += round(dhtDevice_loznice.temperature, 3)
32                     divide += 1
33                 except:
34                     temp = 0
35                     divide = 1
36             if divide != 0:
37                 self.temperature = round(temp / divide, 3)
38             else:
39                 self.temperature = round(temp, 3)
40
41     def add_temp_database(self):
42         sql = "INSERT INTO `Bakalarka`.`temp_in` ( `sensor_id`, `temp`)
VALUES ('{}',{})".format(self.sensorID,
43                                     self.temperature
44         )
45     try:

```

```
45         self.insert_to_database(sql)
46         return 1
47     except:
48         return 0
```