```
# Import libraries
import csv
import time
import socket
import RPi.GPIO as GPIO
import requests
import gspread

# Import python files
from switch import *
from temperature import *
from light import *

# Functions
def getip():
    try:
        s = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
        s.connect(("8.8.8.8", 80))
        return s.getsockname()[0]
    except:
        return 0

def TelMess(text):
    try:
        base_url = 'https://api.telegram.org/bot5189477795:AAEVYv_V0PWOicis7RtdYsNIQZFNOMHxIJk/' \
                   'sendMessage?chat_id=-682759305&text={text}'.format(
            text=text)
        requests.get(base_url)
    except:
        return 0


# Mysql Parameters
ip = 'XXX.XXX.XXX.XXX'
port = '3306'
database = 'Bakalarka'

# Setup relay
led_y = 6
led_r = 13
heat_foil_bed = 26
heat_foil_liv = 19

# Setup switches
button_1 = 27
button_2 = 23
```

```
button_3 = 17
button_4 = 18

# Setup DHT Sensors
temp_cycle_limit = 10

# Variables
cycle = 0
run = 1
cycle_but1 = 0
cycle_but2 = 0
cycle_but3 = 0
cycle_but4 = 0
relay = [False, False, False, False]
online = [2,2,2,2,2,2,2,2,2,2,2,2,2]

# Temp setting
set_temp_bed = 20
set_temp_liv = 23
mer_dokument = 'regulace_v2.csv'
delta_temp = 0.5
time_zone = 3600  # +1 hour to time (Prague)
mess_delay = 30  # 30 sec mess delay

# Switches
bedroom_switch = Switch('switch', 'QYKPMdbKNydPTW5k', ip, database, port, '''loznice''', 0, button_1)
living_room_switch_1 = Switch('switch', 'QYKPMdbKNydPTW5k', ip, database, port, '''obyvak1''', 0, button_2)
living_room_switch_2 = Switch('switch', 'QYKPMdbKNydPTW5k', ip, database, port, '''obyvak2''', 0, button_3)

#Lights
bedroom_light = Light('light', '3yRdaB3r6by5', ip, database, port, '''obyvak''', 0, led_r)
living_room_light = Light('light', '3yRdaB3r6by5', ip, database, port, '''loznice''', 0, led_y)

GPIO.setmode(GPIO.BCM)

# Setup switch to INPUT
GPIO.setup(button_1, GPIO.IN, pull_up_down=GPIO.PUD_UP)  # Button 1 for Red led
GPIO.setup(button_2, GPIO.IN, pull_up_down=GPIO.PUD_UP)  # Button for Yellow led
GPIO.setup(button_3, GPIO.IN, pull_up_down=GPIO.PUD_UP)  # End loop button
GPIO.setup(button_4, GPIO.IN, pull_up_down=GPIO.PUD_UP)  # Button 1 for Red led

# Setup Relay to 0 by default value
GPIO.setup(heat_foil_bed, GPIO.IN)  # Heating foil 1
GPIO.setup(heat_foil_liv, GPIO.IN)  # Heating foil 2
GPIO.setup(led_r, GPIO.IN)  # Led yellow
GPIO.setup(led_y, GPIO.IN)  # Led red

DhtSensor1 = DhtSensor('TempSensor', 'Kj#7](J&haH>QYx`', ip, database, port, "Obyvak_DHT", 0)
DhtSensor2 = DhtSensor('TempSensor', 'Kj#7](J&haH>QYx`', ip, database, port, "Loznice_DHT", 0)

# Setup Google sheet API
try:
    sa = gspread.service_account(filename="../../odevzdání/service_account_google.json")
    sh = sa.open("Nastaveni")
    wks_sett = sh.worksheet("Default setting")

    set_temp_bed = float(wks_sett.acell('B5').value)
    set_temp_liv = float(wks_sett.acell('B4').value)
    delta_temp = float(wks_sett.acell('B3').value)
    mess_delay = float(wks_sett.acell('B6').value)
except:
    print("Google sheets not working")
# Connect user into database
try:
    DhtSensor1.conn_to_database()
    DhtSensor2.conn_to_database()
    bedroom_switch.conn_to_database()
    living_room_switch_1.conn_to_database()
    living_room_switch_2.conn_to_database()
    bedroom_light.conn_to_database()
    living_room_light.conn_to_database()
    online[0] = 1
except:
    online[0] = 0

seconds = time.time()
local_time = time.ctime(seconds + time_zone)  # 3600 timezone to Prague
next_measurement = time.ctime(seconds + time_zone + mess_delay)  # 30 sec delay for measurement
print("Local time:", local_time)

# print("Ip address: ",getip())
```

```python
TelMess("Aplikace Smart-Home byla spuštěna")
TelMess("Ip adresa RPI je:" + str(getip()))

# Measurement to csv file herader
header = ['TIME', 'TEMP_OB', 'TEMP_LOZ', 'REL_OB', 'REL_LOZ']
with open(mer_dokument, 'w', encoding='UTF8', newline='') as f:
    writer = csv.writer(f)
    writer.writerow(header)

# Lists

Light_list = [living_room_light,bedroom_light]
Switch_list = [bedroom_switch, living_room_switch_1, living_room_switch_2]

# Super loop
while run == 1:
    cycle = cycle + 1

    # Button repair
    if cycle_but1 > 50000:
        cycle_but1 = 50000
    if cycle_but2 > 50000:
        cycle_but2 = 50000
    if cycle_but3 > 50000:
        cycle_but3 = 50000
    if cycle_but4 > 50000:
        cycle_but4 = 50000

    # Control switches
    if GPIO.input(bedroom_switch.gpio_port) == 0:
        cycle_but1 = cycle_but1 + 1
    else:
        cycle_but1 = 0

    if GPIO.input(living_room_switch_1.gpio_port) == 0:
        cycle_but2 = cycle_but2 + 1
    else:
        cycle_but2 = 0

    if GPIO.input(living_room_switch_2.gpio_port) == 0:
        cycle_but3 = cycle_but3 + 1
    else:
        cycle_but3 = 0

    if GPIO.input(button_4) == 0:
        cycle_but4 = cycle_but4 + 1
    else:
            cycle_but4 = 0

        if cycle_but1 > 20000:
            if bedroom_switch.actual_state != 1:
                bedroom_switch.actual_state = 1
                online[1] = bedroom_switch.update_database_state()
        else:
            if bedroom_switch.actual_state != 0:
                bedroom_switch.actual_state = 0
                online[2] = bedroom_switch.update_database_state()

        if cycle_but2 > 20000:
            if living_room_switch_1.actual_state != 1:
                living_room_switch_1.actual_state = 1
                online[3] = living_room_switch_1.update_database_state()
        else:
            if living_room_switch_1.actual_state != 0:
                living_room_switch_1.actual_state = 0
                online[4] = living_room_switch_1.update_database_state()

        if cycle_but3 > 20000:
            if living_room_switch_2.actual_state != 1:
                living_room_switch_2.actual_state = 1
                online[5] = living_room_switch_2.update_database_state()
        else:
            if living_room_switch_2.actual_state != 0:
                living_room_switch_2.actual_state = 0
                online[6] = living_room_switch_2.update_database_state()

        if cycle_but4 > 20000: # Program END
            run = 0
        # Light set on/off
        if bedroom_switch.actual_state == 1:
            if not relay[0]:
                relay[0] = True
                bedroom_light.actual_state = 1
                online[7] = bedroom_light.update_datab_state()
        else:
            if relay[0]:
                relay[0] = False
                bedroom_light.actual_state = 0
                online[8] = bedroom_light.update_datab_state()

        if living_room_switch_1.actual_state != living_room_switch_2.actual_state:
            if not relay[1]:
                living_room_light.actual_state = 1
            online[9] = living_room_light.update_datab_state()
            relay[1] = True
    else:
        if relay[1]:
            living_room_light.actual_state = 0
            online[10] = living_room_light.update_datab_state()
            relay[1] = False

    if local_time >= next_measurement:

        # new time for measurement
        next_measurement = time.ctime(seconds + time_zone + mess_delay)
        print("Another measurement will be in: ", str(next_measurement))

        # Measure temperature
        DhtSensor1.mess_temperature()
        DhtSensor2.mess_temperature()
        # Living room switch
        if DhtSensor2.temperature > set_temp_liv + delta_temp:
            GPIO.setup(heat_foil_liv, GPIO.IN)
            relay[3] = False
        elif DhtSensor2.temperature < set_temp_liv - delta_temp:
            GPIO.setup(heat_foil_liv, GPIO.OUT)
            relay[3] = True
        # Bedroom switch
        if DhtSensor1.temperature > set_temp_bed + delta_temp:
            GPIO.setup(heat_foil_bed, GPIO.IN)
            relay[2] = False
        elif DhtSensor1.temperature < set_temp_bed - delta_temp:
            GPIO.setup(heat_foil_bed, GPIO.OUT)
            relay[2] = True

        TelMess("Teplota v ložnici: " + str(DhtSensor1.temperature) + "°C \n Teplota v obýváku: " +
                str(DhtSensor2.temperature) + "°C" + "\nNastavení teplot pro relé: \nLožnice: " +
                str(set_temp_bed) + "°C \nObývák: " + str(set_temp_liv) + "°C\n Delta: " + str(delta_temp) +
                "°C \nStav osvětlení: \nLožnice: " + str(relay[0]) + "\nObývák: " + str(relay[1]) +
                "\nStav topných folií: \nLožnice: " + str(relay[2]) + "\nObývák: " + str(relay[3]))

        online[11] = DhtSensor1.add_temp_database()
        online[12] = DhtSensor2.add_temp_database()

        print(online)

        data = [local_time, DhtSensor2.temperature, DhtSensor1.temperature, relay[2], relay[3]]
        with open(mer_dokument, 'a', encoding='UTF8', newline='') as f:
            writer = csv.writer(f)
            writer.writerow(data)

        # Check database connection
        if 0 in online:
            print("Databáze není připojena")
            try:
                DhtSensor1.conn_to_database()
                DhtSensor2.conn_to_database()
                bedroom_switch.conn_to_database()
                living_room_switch_1.conn_to_database()
                living_room_switch_2.conn_to_database()
                bedroom_light.conn_to_database()
                living_room_light.conn_to_database()
                print("Databáze opět připojena")
            except:
                print("Nepodařilo se databázi připojit")

        else:
            print("Databáze je připojena")

        # Update data from and to google sheets
        try:
            # update from
            set_temp_bed = float(wks_sett.acell('B5').value)
            set_temp_liv = float(wks_sett.acell('B4').value)
            delta_temp = float(wks_sett.acell('B3').value)
            mess_delay = float(wks_sett.acell('B6').value)

            # update to

            wks_sett.update('F2', '{teplota} °C'.format(teplota = DhtSensor1.temperature))
            wks_sett.update('F3', '{teplota} °C'.format(teplota=DhtSensor2.temperature))
            wks_sett.update('F4', '{stav}'.format(stav=relay[0]))
            wks_sett.update('F5', '{stav}'.format(stav=relay[1]))
            wks_sett.update('H2', '{stav}'.format(stav=relay[3]))
            wks_sett.update('H3', '{stav}'.format(stav=relay[2]))
            wks_sett.update('H5', '{time}'.format(time=local_time))
            wks_sett.update('H6', '{IP}'.format(IP=getip()))
        except:
            print("Google sheets not working")
```

```

    if cycle == 50000:  # End of Cycle
        seconds = time.time()
        local_time = time.ctime(seconds + time_zone)
        cycle = 0

#End of program
GPIO.cleanup()
TelMess("Aplikace Smart-Home byla ukončena")
try:
    DhtSensor1.close_database()
    DhtSensor2.close_database()
    bedroom_switch.close_database()
    living_room_switch_1.close_database()
    living_room_switch_2.close_database()
    living_room_light.close_database()
    bedroom_light.close_database()
except:
    print("Nebylo správně ukončeno")
```

```
from datab_con import *
import RPi.GPIO as GPIO

class Light(database):
    def __init__(self, user, password, host, name_of_database, port, location, actual_state, gpio_port):
        super().__init__(user, password, host, name_of_database, port)
        self.location = location
        self.actual_state = actual_state
        self.gpio_port = gpio_port

    def update_datab_state(self):
        if self.actual_state == 1:
            GPIO.setup(self.gpio_port, GPIO.OUT)
        else:
            GPIO.setup(self.gpio_port, GPIO.IN)

        sql = "UPDATE `Bakalarka`.`lights` SET state = {state} WHERE location={light_name}".format(state=self.actual_state,
                                                                                                  light_name=self.location)
        try:
            self.insert_to_database(sql)
            return 1
        except:
            return 0
```

```
from datab_con import *


class Switch(database):
    def __init__(self, user, password, host, name_of_database, port, location, actual_state, gpio_port):
        super().__init__(user, password, host, name_of_database, port)
        self.location = location
        self.actual_state = actual_state
        self.gpio_port = gpio_port

    def get_actual_state(self):
        sql = "SELECT * FROM akalarka.switches WHERE location={name_switch}".format(name_switch=self.location)
        online_state_switch = self.select_from_database(sql)
        self.actual_state = int(online_state_switch[0][2])
        print(self.actual_state)

    def update_database_state(self):
        sql = "UPDATE `Bakalarka`.`switches` SET state = {state} WHERE name_switch={switch_name}".format(state=self.actual_state,
                                                                                                           switch_name=self.location)
        try:
            self.insert_to_database(sql)
            return 1
        except:
            return 0
```

```python
import mysql.connector

class database:

    def __init__(self, user, password, host, name_of_database, port):
        self.user = user
        self.password = password
        self.host = host
        self.name_of_database = name_of_database
        self.port = port

    def conn_to_database(self):
        print("Attempting to connect a user: " + self.user)
        try:
            global my_database
            my_database = mysql.connector.connect(
                host=str(self.host),
                user=str(self.user),
                password=str(self.password),
                database = str(self.name_of_database),
                port = str(self.port)
                )
            print("Connect to database was successful")
        except mysql.connector.Error as e:
            print(e)

    def insert_to_database(self,sql):
        cursor = my_database.cursor()
        cursor.execute(sql)
        my_database.commit()

    def select_from_database(self,sql):
        try:
            cursor = my_database.cursor()
            cursor.execute(sql)
            return cursor.fetchall()
        except:
            return 10

    def close_database(self):
        try:
            user = self.user
            my_database.close()
            print("Connections is closed for " + user)
            return 1
        except:
            print("Connections isn't closed for " + user)
        return 0

```

```python
from datab_con import *

import board
import adafruit_dht

dhtDevice_obyvak = adafruit_dht.DHT22(board.D20, use_pulseio=False)
dhtDevice_loznice = adafruit_dht.DHT22(board.D21, use_pulseio=False)


class DhtSensor(database):

    def __init__(self, user, password, host, name_of_database, port, sensorID, temperature):
        super().__init__(user, password, host, name_of_database, port)
        self.sensorID = sensorID
        self.temperature = temperature

    def mess_temperature(self):
        temp = 0
        divide = 0
        for i in range(10):
            if self.sensorID == "Obyvak_DHT":
                try:
                    temp += round(dhtDevice_obyvak.temperature, 3)
                    divide += 1
                except:
                    temp = 0
                    divide = 1

            if self.sensorID == "Loznice_DHT":
                try:
                    temp += round(dhtDevice_loznice.temperature, 3)
                    divide += 1
                except:
                    temp = 0
                    divide = 1
        if divide != 0:
            self.temperature = round(temp / divide, 3)
        else:
            self.temperature = round(temp, 3)

    def add_temp_database(self):
        sql = "INSERT INTO `Bakalarka`.`temp_in` ( `sensor_id`, `temp`) VALUES ('{}',{})".format(self.sensorID,
                                                                                                  self.temperature)
        try:
            self.insert_to_database(sql)
            return 1
        except:
            return 0
```