Project: Temperature Monitoring System

Interfacing LM35 Sensor to Measure Temperature

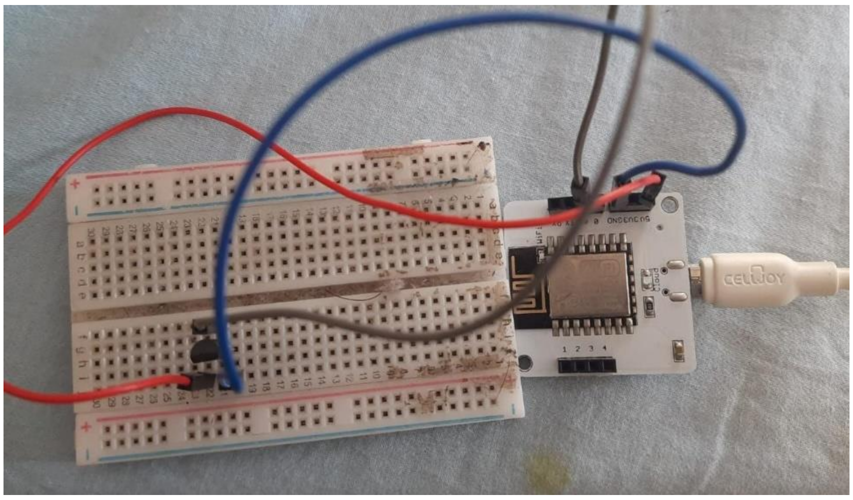
Outcomes

- ➢ Learn how to interface the LM35 sensor with Wi-Fi based Microcontroller module
- ➢ Recording the temperature using this sensor.
- ➢ Learn how to send alerts via E-mail, SMS and Telegram Channel when the temperature value crosses a threshold.

Hardware required

- The Bolt Wi-Fi module
- Jumper Wires
- Breadboard
- Temperature Sensor: LM35 sensor

Circuit Diagram



Key Points:

- Holding the sensor with flat portion facing, the pins of the sensor are designated as VCC, Output and Gnd from your left to right
- Using jumper wire connect:
    - VCC pin of the LM35 connects to 5v of the Bolt Wifi module.
    - Output pin of the LM35 connects to A0 (Analog input pin) of the Bolt Wifi module
    - Output pin of the LM35 connects to A0 (Analog input pin) of the Bolt Wifi module

Third-party Services used:

- Digital Ocean Droplet
    - It provides developers cloud services that help to deploy and scale applications that run simultaneously on multiple computers. DigitalOcean calls its cloud servers Droplets which accounts to the same thing as Instances by AWS(Amazon Web services).
    - To-Do:
        - Create an account and do necessary steps.
        - Click on Create button and choose Droplet from the drop-down menu
        - Choosing a Size($5/month plan is enough for this tasks)
        - Choosing a Datacenter Region
        - Make a note of login credentials and server IP
        - If you are a Windows OS, the most popular software for SSH in windows is Putty. Putty is a free Telnet and SSH terminal software for Windows and Unix platforms that enable users to remotely access computers over the Internet.
        - Once PuTTY is installed, start the program, enter the IP Address of your droplet, username and password.
- Twilio
    - It is a third-party SMS functionality provider. It is a cloud communications platform as a service (PaaS) company. Twilio allows software developers to programmatically make and receive phone calls and also send and receive text messages using its web service APIs.
    - To-Do:
        - SIGN UP
        - Choose Product and Language as per your expertise(For this task,I have used, Product→SMS, Building→ Alert & Notification, Language→Python)
        - Take a note of Account SID, Auth token and a Twilio phone number.
            - Steps to follow:
                - Connect the temperature monitoring circuit
                - Login into the putty by entering the IP address of your digital ocean droplet.

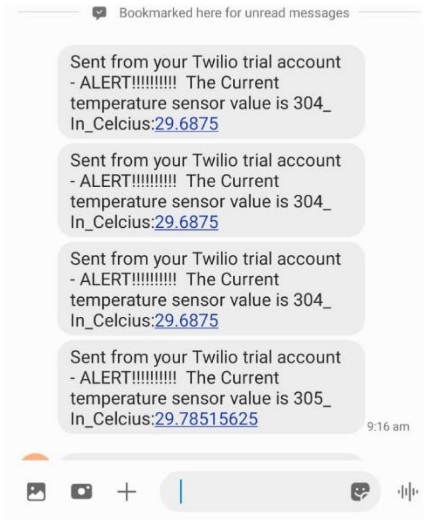
- After successful login, create a file named `conf.py`
  - `Code:`

```
SID = 'You can find SID in your Twilio Dashboard'

AUTH_TOKEN = 'You can find  on your Twilio Dashboard'

FROM_NUMBER = 'This is the no. generated by Twilio. You can find this on your Twilio Dashboard'

TO_NUMBER = 'This is your number. Make sure you are adding +91 in beginning'
API_KEY = 'This is your Bolt Cloud accout API key'
DEVICE_ID = 'This is the ID of your Bolt device'
```

- Now create another file named `temp_sms.py`
  - `Code:`

```
import conf, json, time
from boltiot import Sms, Bolt
import json, time

minimum_limit = 300
maximum_limit = 600


mybolt = Bolt(conf.API_KEY, conf.DEVICE_ID)
sms = Sms(conf.SID, conf.AUTH_TOKEN, conf.TO_NUMBER, conf.FROM_NUMBER)


while True:
    print ("Reading sensor value")
    response = mybolt.analogRead('A0')
    data = json.loads(response)
    print("Sensor value is: " + str(data['value']))
    try:
        sensor_value = int(data['value'])
        if sensor_value > maximum_limit or sensor_value < minimum_limit:
            print("Making request to Twilio to send a SMS")
            response = sms.send_sms("The Current temperature sensor value is " +str(sensor_value))
            print("Response received from Twilio is: " + str(response))
            print("Status of SMS at Twilio is :" + str(response.status))
    except Exception as e:
        print ("Error occured: Below are the details")
        print (e)
    time.sleep(10)
```

Output:



Sent from your Twilio trial account - ALERT!!!!!!!!!! The Current temperature sensor value is 304_ In_Celcius:29.6875

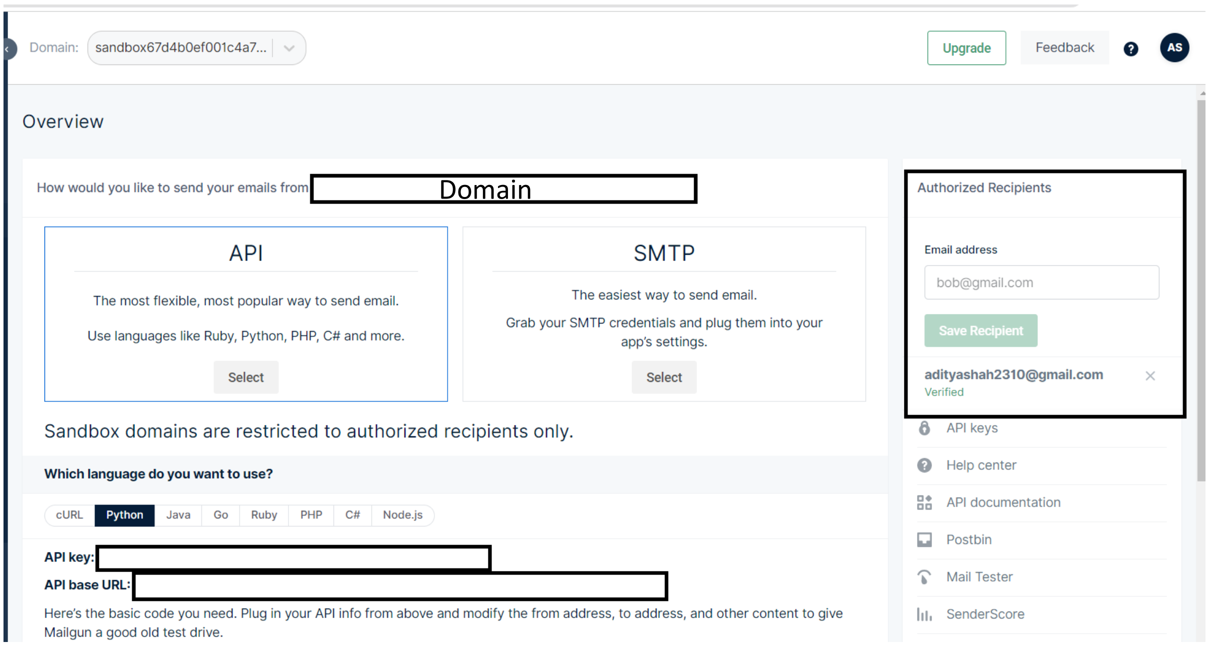
Sent from your Twilio trial account - ALERT!!!!!!!!!! The Current temperature sensor value is 304_ In_Celcius:29.6875

Sent from your Twilio trial account - ALERT!!!!!!!!!! The Current temperature sensor value is 304_ In_Celcius:29.6875

Sent from your Twilio trial account - ALERT!!!!!!!!!! The Current temperature sensor value is 305_ In_Celcius:29.78515625

9:16 am

- Mailgun
  - It is an Email automation service. It has a very powerful set of inbuilt functions for sending emails. Developers can process their email with the help of Mailgun API
  - To-DO:
    - Sign Up
    - `Add Recipient and Verify the recipient`
    - Click on the ID of the newly generated sandbox
    - Make a note of sandbox URL(Omit https://api.mailgun.net/v3/) , API key and sender email id(include mailgun@ before your domain



Overview

How would you like to send your emails from Domain

API

The most flexible, most popular way to send email.

Use languages like Ruby, Python, PHP, C# and more.

Select

SMTP

The easiest way to send email.

Grab your SMTP credentials and plug them into your app's settings.

Select

Sandbox domains are restricted to authorized recipients only.

Which language do you want to use?

API key: [redacted]

API base URL: [redacted]

Here's the basic code you need. Plug in your API info from above and modify the from address, to address, and other content to give Mailgun a good old test drive.

Authorized Recipients

Email address

bob@gmail.com

Save Recipient

adityashah2310@gmail.com
Verified

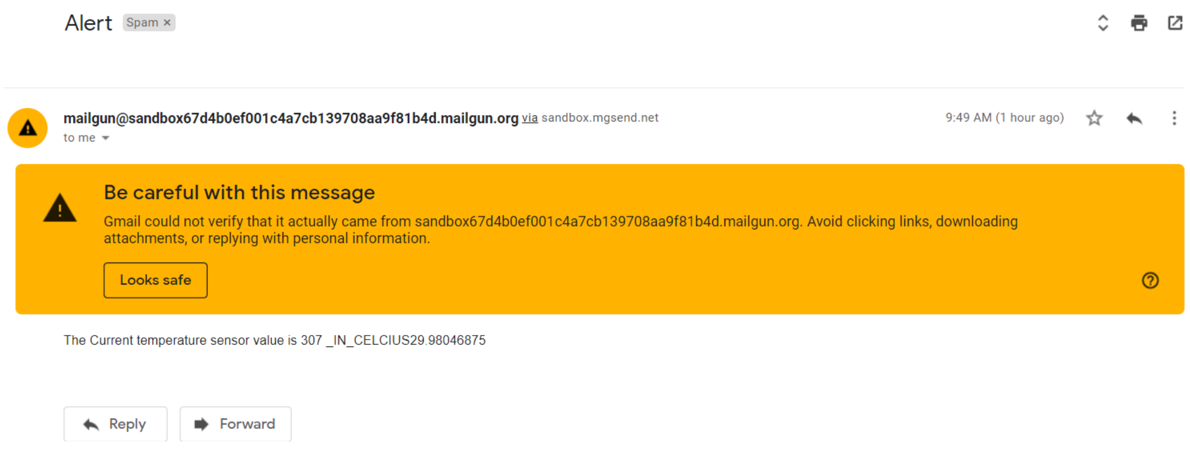
- Steps:
  - Connect the temperature monitoring circuit
  - Login into the putty by entering the IP address of your digital ocean droplet
  - Create a file named `email_conf.py`
    - Code

```
MAILGUN_API_KEY = 'This is the private API key which you can find on your Mailgun Dashboard'

SANDBOX_URL= 'You can find this on your Mailgun Dashboard'

SENDER_EMAIL = 'This would be test@your SANDBOX_URL'

RECIPIENT_EMAIL = 'Enter your Email ID Here'
API_KEY = 'This is your Bolt Cloud accout API key'
DEVICE_ID = 'This is the ID of your Bolt device'
```

  - Create one more file named `temp_email.py`
    - Code

```
import email_conf, json, time
from boltiot import Email, Bolt

minimum_limit = 300 #the minimum threshold of light value
maximum_limit = 600 #the maximum threshold of light value


mybolt = Bolt(email_conf.API_KEY, email_conf.DEVICE_ID)
mailer = Email(email_conf.MAILGUN_API_KEY, email_conf.SANDBOX_URL, email_conf.SENDER_EMAIL, email_conf.RECIPIENT_EMAIL)


while True:
    print ("Reading sensor value")
    response = mybolt.analogRead('A0')
    data = json.loads(response)
    print ("Sensor value is: " + str(data['value']))
    try:
        sensor_value = int(data['value'])
        if sensor_value > maximum_limit or sensor_value < minimum_limit:
            print("Making request to Mailgun to send an email")
            response = mailer.send_email("Alert", "The Current temperature sensor value is " +str(sensor_value))
            response_text = json.loads(response.text)
            print("Response received from Mailgun is: " + str(response_text['message']))
    except Exception as e:
        print ("Error occured: Below are the details")
        print (e)
    time.sleep(10)
```

Output:


Alert Spam ×

mailgun@sandbox67d4b0ef001c4a7cb139708aa9f81b4d.mailgun.org via sandbox.mgsend.net
to me

9:49 AM (1 hour ago)

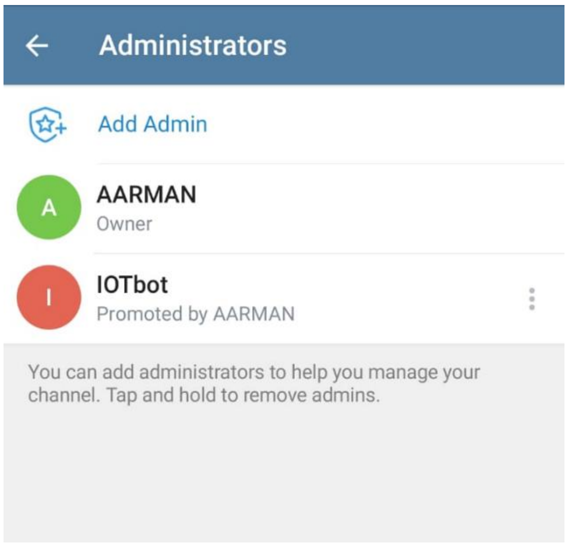
Be careful with this message

Gmail could not verify that it actually came from sandbox67d4b0ef001c4a7cb139708aa9f81b4d.mailgun.org. Avoid clicking links, downloading attachments, or replying with personal information.

Looks safe

The Current temperature sensor value is 307 _IN_CELCIUS29.98046875

- Telegram
  - Telegram is a messaging app similar to Whatsapp. You can send and receive messages along with files also. It is FREE to use. You can access the platform via your Android/iOS/Windows phone and also your PC or Mac.
  - To-Do:
    - Download and install the latest version of Telegram app
    - Create a new channel
    - Set the channel as Public
    - Keep a note of this Channel permanent link name
    - Click on the channel name at the top and make a note of Text after t.me/ as your Telegram chat ID
    - Create a new Bot
      - BOT
        - These are third-party applications that run inside Telegram. Users can interact with bots by sending them messages, commands and requests.
        - On the Home screen of the app, click on the search icon on the top right and type in "botfather" with a blue tick mark
        - type in "/newbot" and then type in few more details like Bot name and Bot username.
        - When your bot is created successfully, make a note of the token(Omit HTTP API: and add the keywork “bot” before the token) as it will allow you to access your Bot. For ex: "bot894346529:AAhuJ2XJQy5dlEtLYF0sc0Z_qu0fSqihSSc"
    - Add the bot to channel
      - Click on the channel name and go to Administrators
      - Add the newly created bot to the channel(Search for the bot using the Bot's username)


Administrators

Add Admin

AARMAN
Owner

IOTbot
Promoted by AARMAN

You can add administrators to help you manage your channel. Tap and hold to remove admins.

- Code:
  - conf.py

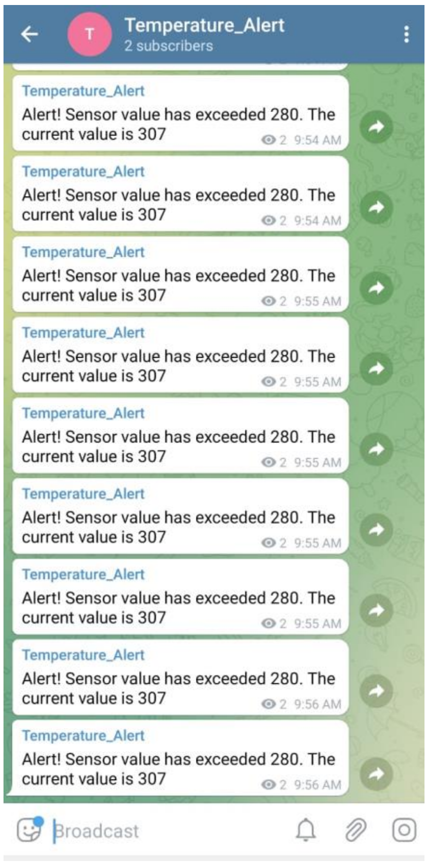
```
➢ """Configurations for telegram_alert.py"""
➢ bolt_api_key = "XXXX"                 # This is your Bolt Cloud API Key
➢ device_id = "XXXX"                    # This is the device ID and will be sim
  ilar to BOLTXXXX where XXXX is some numbers
➢ telegram_chat_id = "@XXXX"            # This is the channel ID of the created
  Telegram channel. Paste after @
➢ telegram_bot_id = "botXXXX"           # This is the bot ID of the created Tel
  egram Bot. Paste after bot
➢ threshold = 250                       # Threshold beyond which the alert shou
  ld be sent
```

  - telegram_alert.py

```
➢ import requests                   # for making HTTP requests
➢ import json                       # library for handling JSON data
➢ import time                       # module for sleep operation
➢
➢ from boltiot import Bolt          # importing Bolt from boltiot module
➢ import conf                       # config file
➢
➢ mybolt = Bolt(conf.bolt_api_key, conf.device_id)
➢
➢ def get_sensor_value_from_pin(pin):
➢     """Returns the sensor value. Returns -999 if request fails"""
➢     try:
➢         response = mybolt.analogRead(pin)
➢         data = json.loads(response)
➢         if data["success"] != 1:
➢             print("Request not successfull")
➢             print("This is the response->", data)
➢             return -999
➢         sensor_value = int(data["value"])
```

```
        return sensor_value
    except Exception as e:
        print("Something went wrong when returning the sensor value")
        print(e)
        return -999


def send_telegram_message(message):
    """Sends message via Telegram"""
    url = "https://api.telegram.org/" + conf.telegram_bot_id + "/sendMessag
e"
    data = {
        "chat_id": conf.telegram_chat_id,
        "text": message
    }
    try:
        response = requests.request(
            "POST",
            url,
            params=data
        )
        print("This is the Telegram response")
        print(response.text)
        telegram_data = json.loads(response.text)
        return telegram_data["ok"]
    except Exception as e:
        print("An error occurred in sending the alert message via Telegram"
)
        print(e)
        return False


while True:
    # Step 1
    sensor_value = get_sensor_value_from_pin("A0")
    print("The current sensor value is:", sensor_value)

    # Step 2
    if sensor_value == -999:
        print("Request was unsuccessfull. Skipping.")
        time.sleep(10)
        continue

    # Step 3
    if sensor_value >= conf.threshold:
        print("Sensor value has exceeded threshold")
        message = "Alert! Sensor value has exceeded " + str(conf.threshold)
+ \
                  ". The current value is " + str(sensor_value)
        telegram_status = send_telegram_message(message)
        print("This is the Telegram status:", telegram_status)

    # Step 4
    time.sleep(10)
```

Output:



Temperature_Alert
Alert! Sensor value has exceeded 280. The current value is 307
2 9:54 AM

Temperature_Alert
Alert! Sensor value has exceeded 280. The current value is 307
2 9:54 AM

Temperature_Alert
Alert! Sensor value has exceeded 280. The current value is 307
2 9:55 AM

Temperature_Alert
Alert! Sensor value has exceeded 280. The current value is 307
2 9:55 AM

Temperature_Alert
Alert! Sensor value has exceeded 280. The current value is 307
2 9:55 AM

Temperature_Alert
Alert! Sensor value has exceeded 280. The current value is 307
2 9:55 AM

Temperature_Alert
Alert! Sensor value has exceeded 280. The current value is 307
2 9:55 AM

Temperature_Alert
Alert! Sensor value has exceeded 280. The current value is 307
2 9:56 AM

Temperature_Alert
Alert! Sensor value has exceeded 280. The current value is 307
2 9:56 AM

[Demonstration Video]