Blood Bank Temperature Prediction and Alert using IoT

\

The ‘Blood Cold Chain’ is the system for storing and transporting blood and its components in regulated temperature. Any break in this chain loses blood's ability to move O_2 or CO_2 to and from tissues upon transfusion.There is also the risk of bacterial contamination if blood is exposed to warm temperatures. Conversely, if exposed to below freezing temperature blood may get hemolyzed. Hemolyzed blood if transfused can cause renal failure and fatal bleeding problems. So control of temperature is vital to the successful shelf life of blood and its components.

Blood is collected at body temperature, i.e. +37 °C. But to take care of its vital properties, it must be cooled to below +10 °C to be transported, and stored at refrigeration temperatures of around +2 °C to +6 °C until use. We can use the Internet of Things(IoT) with Machine Learning(ML) to make blood bank temperature monitoring devices to alert on current and future temperature anomalies.

Things used in this project

Hardware components:

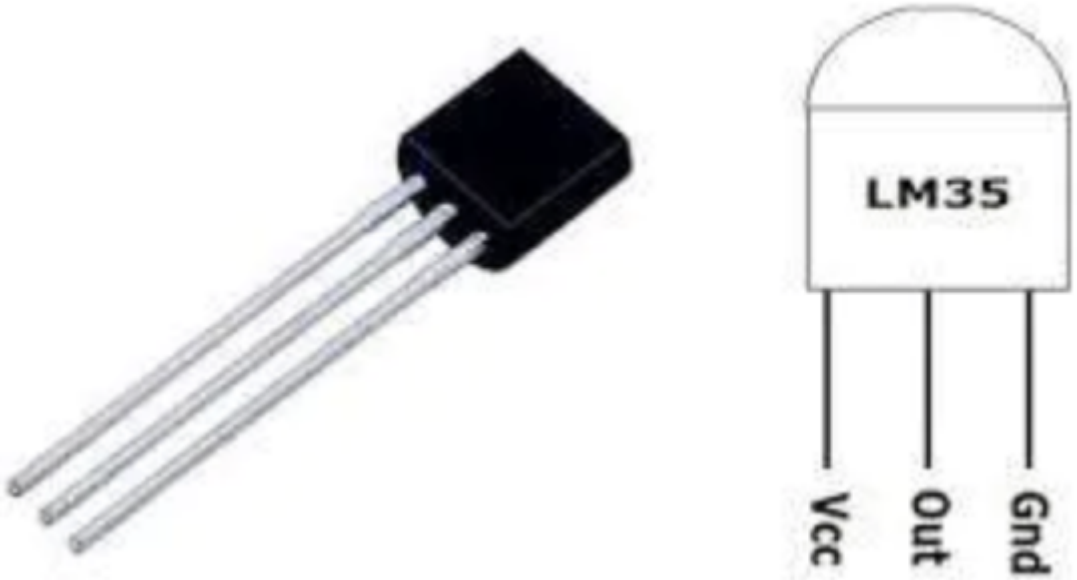
1. Bolt IoT WiFi Module
2. Temperature Sensor LM35
3. Buzzer
4. USB-A to Micro-USB Cable
5. Solder less Breadboard
6. Male/Female Jumper Wires x 5
7. Male/Male Jumper Wires. x 3

Software, Apps and online services:

1. Bolt IoT Android App
2. Bolt IoT Cloud
3. Ubuntu
4. Telegram

Hardware setup:

Step 1: Connect LM35 temperature sensor to the Bolt IoT Wi-Fi module.



Temperature Sensor, LM35*,* has the following three pins

- **VCC** - connected to 5V of Bolt Wi-Fi module.

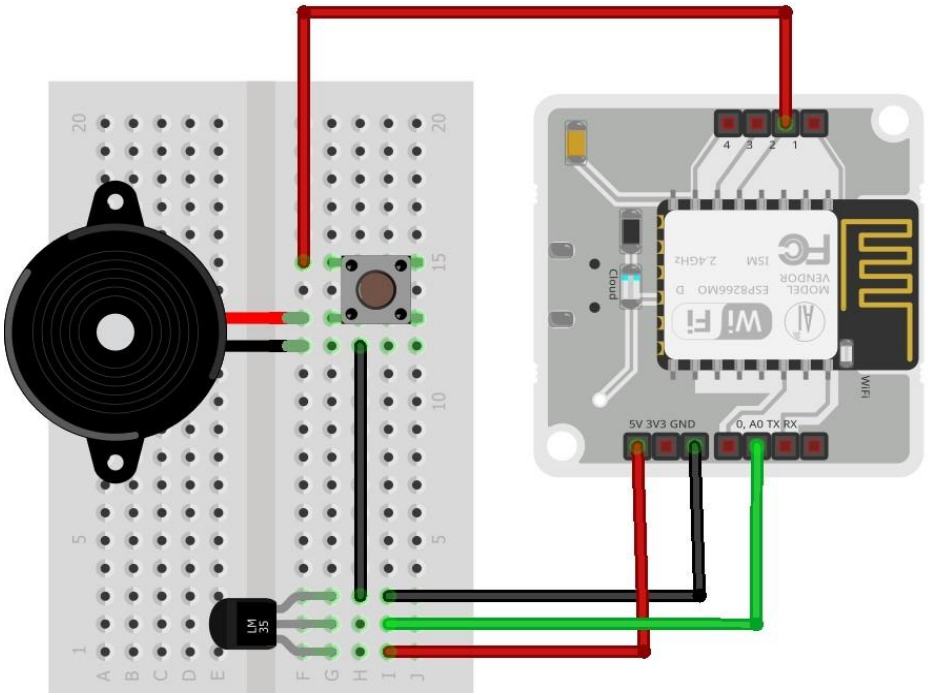
- **Output** - connected to A0 (analog input pin) of Bolt Wi-Fi module.
- **Ground** - connected with negative leg of LED on breadboard; indirectly connected to GND of Bolt Wi-Fi module.

Step 2: Connect Buzzer to the Bolt IoT Wi-Fi module.

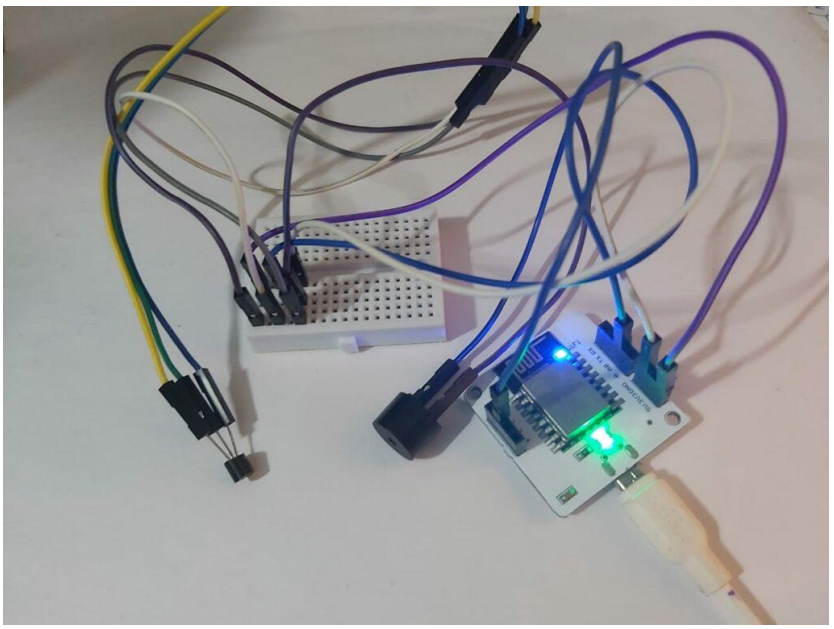
Buzzer has two pins

- **Positive(longer pin)** - connected to GPIO pin 2 of Bolt Wi-Fi module.
- **Negative(shorter pin)** - connected to GND of Bolt Wi-Fi module.

Step 3: Power up the Bolt IoT Wi-Fi module using USB cable.



*In the above schematic one extra component is used i.e. push button, one can include/ exclude it while making connections.



Circuit Connection

Software setup:

Creating telegram channel and bot:

- Open your Telegram App and swipe left open the Main Menu and tap on the feature New Channel.
- Give a suitable name to the channel and provide description to it.
- Set the channel as Public Channel and provide a permanent link to it by giving a suitable name. Note: Space between words is not allowed.
- Now we need to create a Telegram Bot.
- After creating the Telegram Channel, come back to the home page of the Telegram and search for the BotFather in the search menu.
- After opening the BotFather enter the command /start to create the bot.
- To create a new bot enter the command /newbot and give a suitable name to the bot.
- Secondly we need to give the username to the bot created it should be ending with the bot e.g. usernamebot
- Now an API Key will be generated to store that API Key as it has to be used in the coding.

- Now that we have created both Telegram Channel and Telegram Bot next we need to link them both together.
- To do that open your channel and tap on the channel name and then click on the Administrator button and search for the bot name you have created.

Software Programming:

To make and send alerting message facility via Telegram, we need to understand the skeleton of the coding. The whole program has three parts namely:

1. Configuration code: It consists of all the backend details of Bolt IoT Wi-Fi Module and the Telegram.
2. Main code: It consists of the core coding of the facility.
3. Prediction code: It consists of JavaScript code for ML Prediction.

To create above three mention files, follow below given steps:

Open the Digital Ocean Ubuntu server (For Windows /Mac Operating System) or open the terminal in the Ubuntu Operating System.

Step 1: We need to create a directory (folder) to store the first two coding files mentioned above. To create a directory named Blood_Bank_Temp , type out the following command:

```
sudo mkdir Blood_Bank_Temp
```

Step 2: Next we have to enter the directory that we just created. To enter the directory named Blood_Bank_Temp type out the following command:

```
cd Blood_Bank_Temp
```

Step 3: After entering the folder first we need to create the configuration python file which will hold all the backend details of the Bolt IoT Wi-Fi Module,Telegram and Threshold values.To create the configuration python file in the folder which was created first type out the following command for creating the file with extension .py;

```
sudo nano conf.py
```

Step 4: After the above mentioned file enter the following data into the file. Make sure that you add the updated Bolt API key, device id and Telegram details & Threshold values;

```
bolt_api_key = "XXXXXXXX"
device_id = "BOLTXXXXXXXX"
telegram_chat_id = "@XXXXXXXX"
telegram_bot_id = "botXXXXXXXX"
threshold = [20.48, 61.44, 102.4]
frame_size = 10
mul_factor = 2
```

Step 5: Save the file by clicking "ctrl+x" and press enter. Next create another file which will include the main coding of the facility.

```
sudo nano main_code.py
```

Step 6: Enter the following code into the newly created file:

```
import conf, json, time, math, statistics, requests
from boltiot import Bolt

def compute_bounds(history_data, frame_size, factor):
    if len(history_data)<frame_size:
        return None
    if len(history_data)>frame_size:
        del
history_data[0:len(history_data)-frame_size-1]
    Mn = statistics.mean(history_data)
    Variance = 0
    for data in history_data:
        Variance += math.pow((data-Mn),2)
    Zn = factor*math.sqrt(Variance/frame_size)
    High_bound = history_data[frame_size-1]+Zn
    Low_bound= history_data[frame_size - 1]- Zn
    return [High_bound, Low_bound]
```

```
mybolt = Bolt(conf.bolt_api_key, conf.device_id)
history_data = []

def send_telegram_message(message):
    url = "https://api.telegram.org/" +
conf.telegram_bot_id + "/sendMessage"
    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 occurredin sending the alert
message via Telegram")
        print(e)
        return False

def buzzer_alert():
    high_response = mybolt.digitalWrite("2", "HIGH")
    print(high_response)
    time.sleep(5)
    low_response = mybolt.digitalWrite("2", "LOW")
    print(low_response)
    time.sleep(5)


def check_temperature(value,checker):
    if value > conf.threshold[2]:
        print("The temperaure value increased the
threshold value. Sending an alert notification")
        message = "Blood Bank Refrigerator
Temperature increased the threshold value. The
current value is: " + str(int(sensor_value*0.097))
```

```
        telegram_status =
send_telegram_message(message)
        print("This is the Telegram status",
telegram_status)
        if not checker:
            return buzzer_alert()
        if checker:
            return 0

    if value < conf.threshold[0]:
        print("The temperature value  decreased below
the threshold value. Sending an alert notification")
        message = "Blood Bank Refrigerator
Temperature decreased below the threshold value. The
current value is: " + str(int(sensor_value*0.097))
        telegram_status =
send_telegram_message(message)
        print("This is the Telegram status",
telegram_status)
        if not checker:
            return buzzer_alert()
        if checker:
            return 0

    if value > conf.threshold[1] and value <
conf.threshold[2]:
        print("The temperature value is between
",str(int(conf.threshold[1]*0.097))," and
",str(int(conf.threshold[2]*0.097)), ". Sending an
alert notification")
        message = "Blood Bank Refrigerator
Temperature is between " +
str(int(conf.threshold[1]*0.097)) + " and " +
str(int(conf.threshold[2]*0.097)) + ". Check
prediction table. The current value is: " +
str(int(sensor_value*0.097))
        telegram_status =
send_telegram_message(message)
```

```
        print("This is the Telegram status",
telegram_status)
        if not checker:
            return buzzer_alert()
        if checker:
            return 0

    if value > conf.threshold[0] and value <
conf.threshold[1]:
        if not checker:
            time.sleep(10)
        if checker:
            return 0


while True:
    checker = False
    response = mybolt.analogRead('A0')
    data = json.loads(response)
    if data['success'] != 1:
        print("There was an error while retriving the
data")
        print("This is the error:" +data['value'])
        time.sleep(10)
        continue
    print("This is the value: " +data['value'])

    sensor_value = 0
    try:
        sensor_value = int(data['value'])
    except e:
        print("There was an error while parsing the
response: ",e)
        continue

    bound= compute_bounds(history_data,
conf.frame_size, conf.mul_factor)
    if not bound:
```

```
        required_data_count =
conf.frame_size-len(history_data)
        print("Not enough data to compute Z-score.
Need ",required_data_count," more data points")
        history_data.append(int(data['value']))
        check_temperature(sensor_value, checker)
        continue

    try:
        if sensor_value> bound[0]:
            print("The temperature value increased
suddenly. Sending an alert notification")
            message = "Blood Bank Refrigerator
Temperature increased suddenly. The current value is:
" + str(int(sensor_value*0.097))
            telegram_status =
send_telegram_message(message)
            print("This is the Telegram status",
telegram_status)
            buzzer_alert()
            checker = True
        elif sensor_value< bound[1]:
            print("The temperature value decreased
suddenly. Sending an alert notification")
            message = "Blood Bank Refrigerator
Temperature decreased suddenly. The current value is:
" + str(int(sensor_value*0.097))
            telegram_status =
send_telegram_message(message)
            print("This is the Telegram status",
telegram_status)
            buzzer_alert()
            checker = True
        check_temperature(sensor_value, checker)
        history_data.append(sensor_value)
    except Exception as e:
        print("error", e)
```

Step 7: Exit the code editor using "ctrl+x" and then run the code that you have written using the following command in same directory;

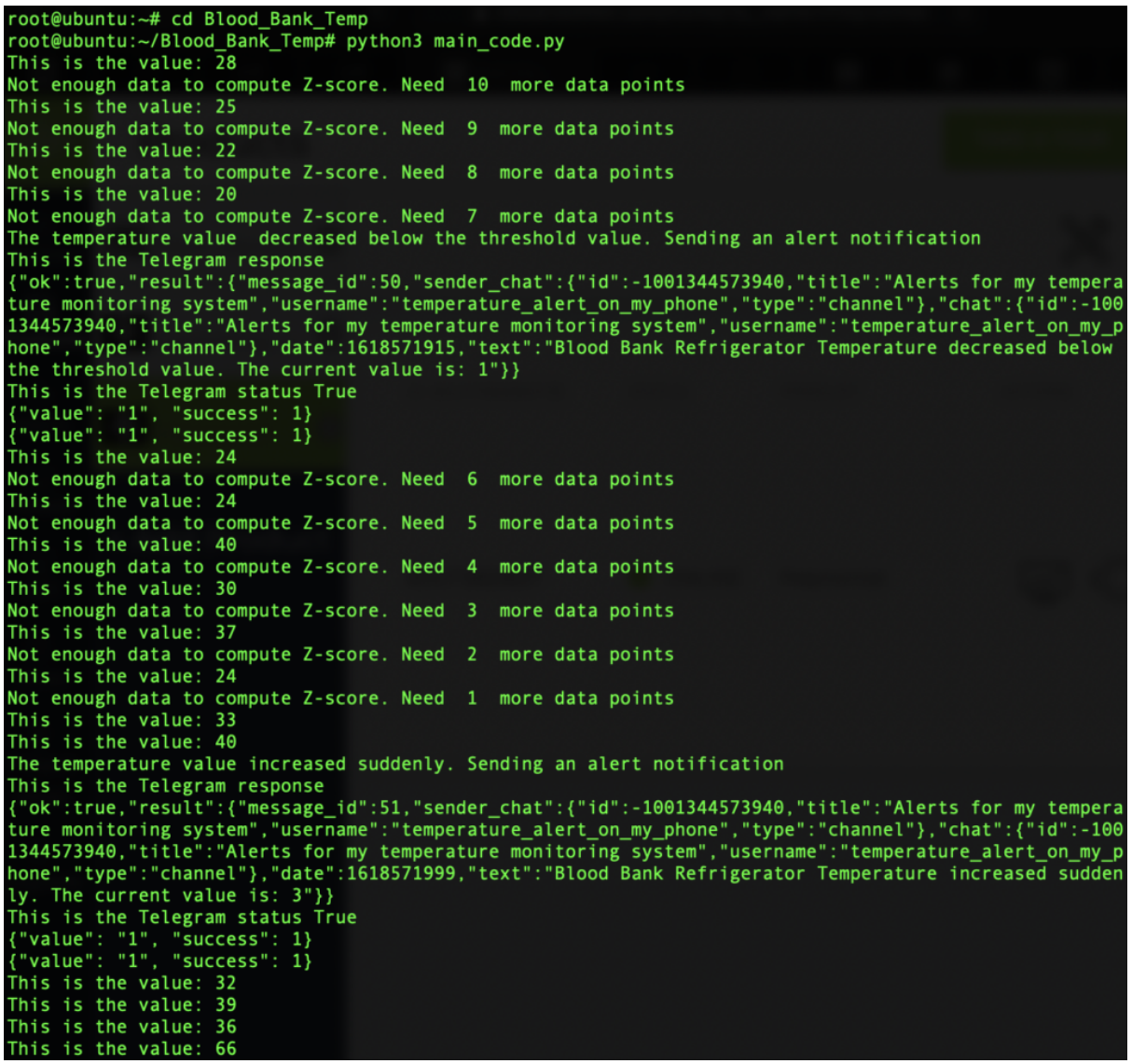
```
sudo python3 main_code.py
```

Step 8: Go to Bolt cloud https://cloud.boltiot.com/ and follow steps

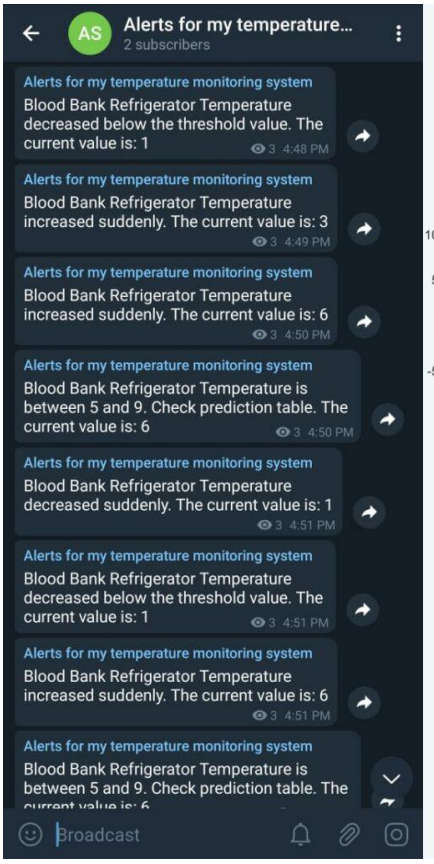
1. Click on Products and than click on + Add Product,
2. Give Product name and select Input devices and GPIO option and Click DONE
3. Click on Configure this product icon
4. In Hardware configuration select data collection rate 5 minutes, select pin to be used as Analog A0 and enter variable name 'temp'
5. In Code configuration enter file name 'predict' and choose file extension as .js
6. Write the below JavaScript code to run Polynomial Regression for prediction and click Save your configuration icon.

```
setChartLibrary('google-chart');
setChartTitle('Polynomial Regression');
setChartType('predictionGraph');
setAxisName('time_stamp','temp');
mul(0.097);
setAnimation(true);
setCrosshair(true);
plotChart('time_stamp','temp');
```

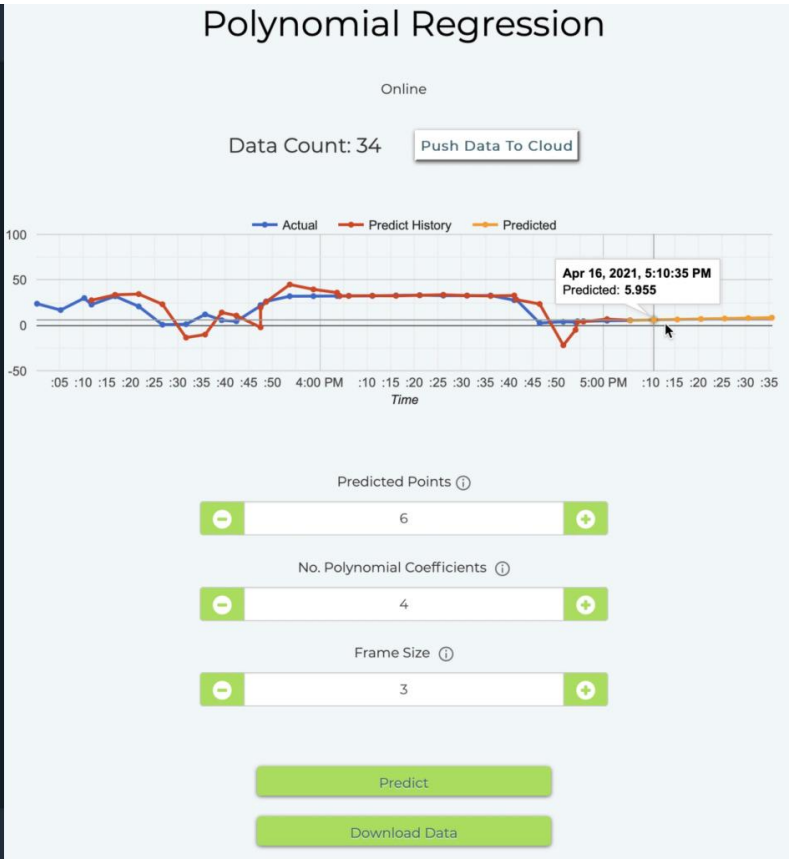
Output:



Output on terminal



Alert send on Telegram



Prediction table on Bolt Cloud/App

Conclusion:

To maintain the 'Blood Cold Chain' one needs to regulate the temperature between +2 °C to +6 °C and during transporting it should not exceed +10 °C. So we can apply some constraints to this temperature threshold and provide alerts when it breaks these constraints so that precaution is taken much earlier. Sudden change in temperature due to leakage or power down is also detected and alerted. Prediction of future 20 min temperatures makes this system more easy to regulate and prevent future anomalies. Buzzer triggers an alarm for 5 sec for every alert and checks the values at an interval of 10 sec.

Video:

https://www.youtube.com/watch?v=3O1Rn7LKCtQ