Cloud, APIs and Alerts > Social Media of Things

Alert messaging using Telegram

What is 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.

Some Telegram terminologies -
What is a Telegram channel?
A channel is to Telegram what Groups are to Whatsapp. Channels are a tool for broadcasting your messages to large audiences. They can have an unlimited number of subscribers, they can be public with a permanent URL and each post in a channel has its own view counter.

What is a Bot?
Bots are third-party applications that run inside Telegram. Users can interact with bots by sending them messages, commands and requests.

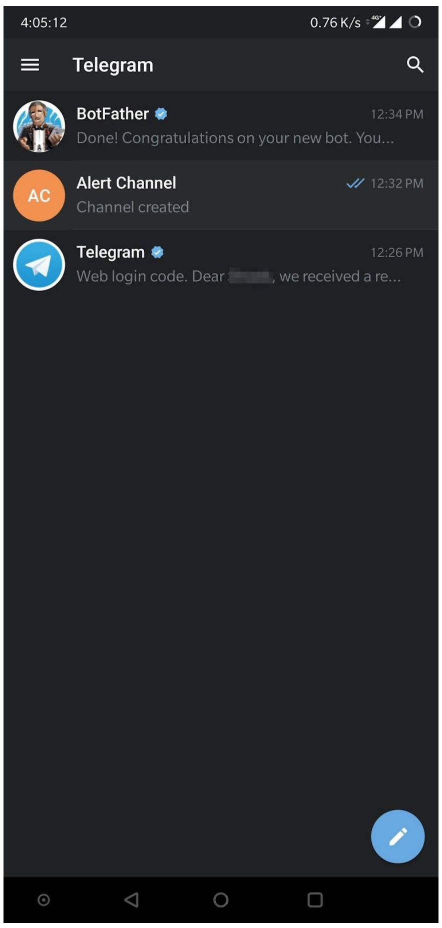
We will be using Bots to send alerts on our channel.

What will we learn in this project?
In this project, you will be able to send alerts via Telegram to your phone when the sensor value exceeds your set threshold.

What do I require for this project?
To send alerts and messages via Telegram, you will require,
- An Android/iOS/Windows phone with Internet connectivity.
- A Telegram account. You will require a mobile number for this.
- Access to your Ubuntu OS via puTTY or Virtual Machine.

Setup Telegram
Steps to sign-up/sign-in to Telegram

1. Go to the Playstore or App Store on your phone and search for Telegram.
2. Download and Install the latest version of the Telegram app.
3. Signup for a new account or sign in to Telegram by providing your mobile number.
4. Telegram will call you OR send you an SMS to verify your mobile number.
5. Get your account verified and approve any permissions if required.
6. You will be logged in to Telegram and will be shown a screen similar to the one below. (It's okay if it differs from the screenshot below)


4:05:12 0.76 K/s

Telegram

BotFather 12:34 PM
Done! Congratulations on your new bot. You...

Alert Channel 12:32 PM
Channel created

Telegram 12:26 PM
Web login code. Dear [redacted], we received a re...

You have successfully installed Telegram and set up your account. In the next lessons, we will be learning more about Channels and Bots.

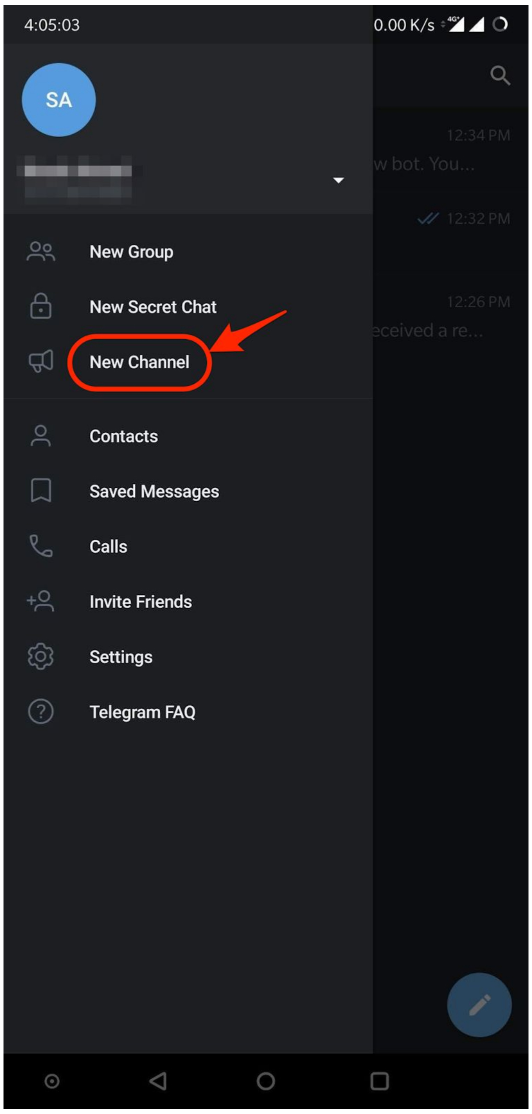
Creating a Channel

What is a Telegram channel?

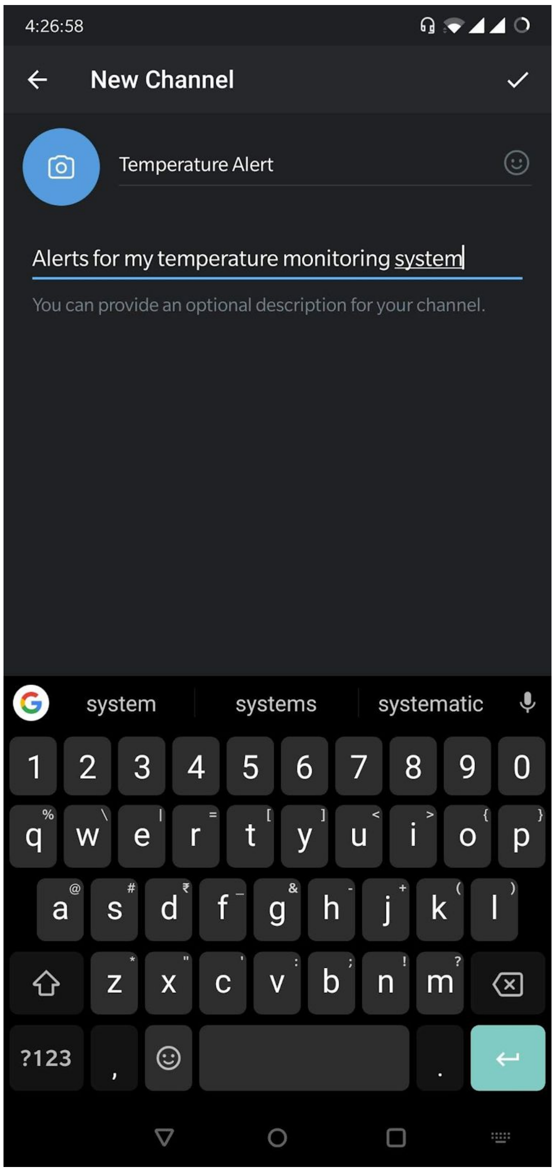
A channel is to Telegram what Groups are to Whatsapp. Channels are a tool for broadcasting your messages to large audiences. They can have an unlimited number of subscribers, they can be public with a permanent URL and each post in a channel has its own view counter.

Steps to create a channel

1. Go to the home screen of the Telegram app.
2. Swipe from the left side to reveal the menu.
3. Click on "New Channel".



4. It will ask you for a Name and Description for your channel. Give a suitable name and description. Adding a photo is optional.



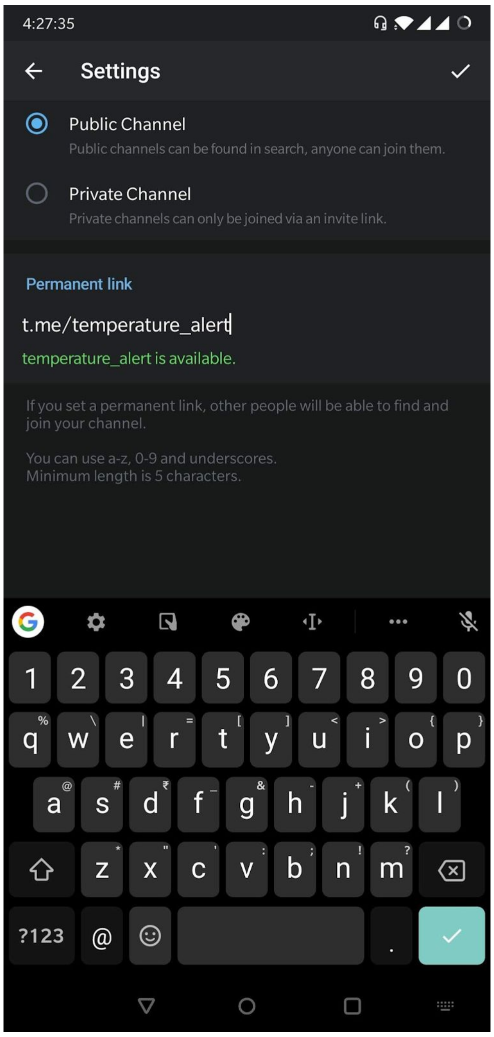
Temperature Alert

Alerts for my temperature monitoring system

You can provide an optional description for your channel.

5. In the next screen set the channel as Public.
6. On the same screen, it will ask you to enter a permanent link for your channel. You can use lowercase letters and numbers 0-9 to create the channel link.
7. <u>Please note that the channel link name is global and you will be able to create a channel link only if the link name is available. The channel link name is something similar to an email address, i.e. only one unique email ID can exist at one time.</u>

8. You can keep the channel link name as temperature_alert_ followed by your email ID. Example, if my email ID is developer@boltiot.com, I can keep the channel link name as temperature_alert_developer_boltiot_com(I have removed the @ and . symbol since it was not allowed and replaced it with an underscore _). This is just a suggestion. You can keep the channel link name as you like as long as it is valid.
9. **Keep a note of this Channel permanent link name.** It will be required later on to send messages to this channel. *For example, the channel link name in the screenshot below is "temperature_alert".*


4:27:35

Settings

● Public Channel
Public channels can be found in search, anyone can join them.

○ Private Channel
Private channels can only be joined via an invite link.

Permanent link

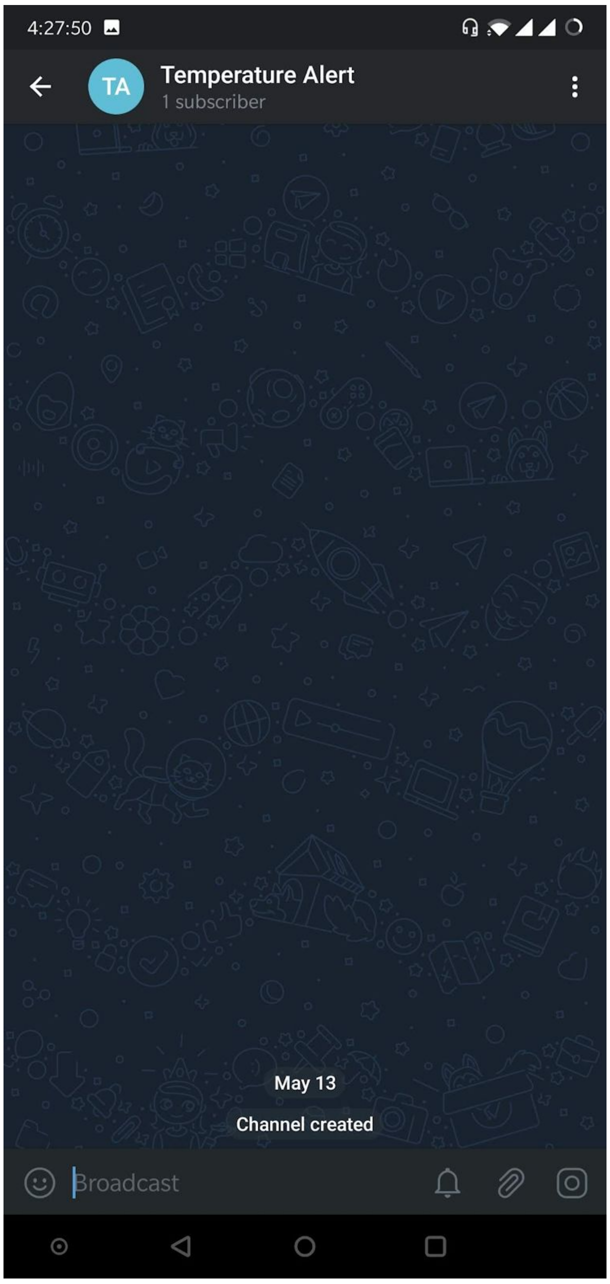
t.me/temperature_alert

temperature_alert is available.

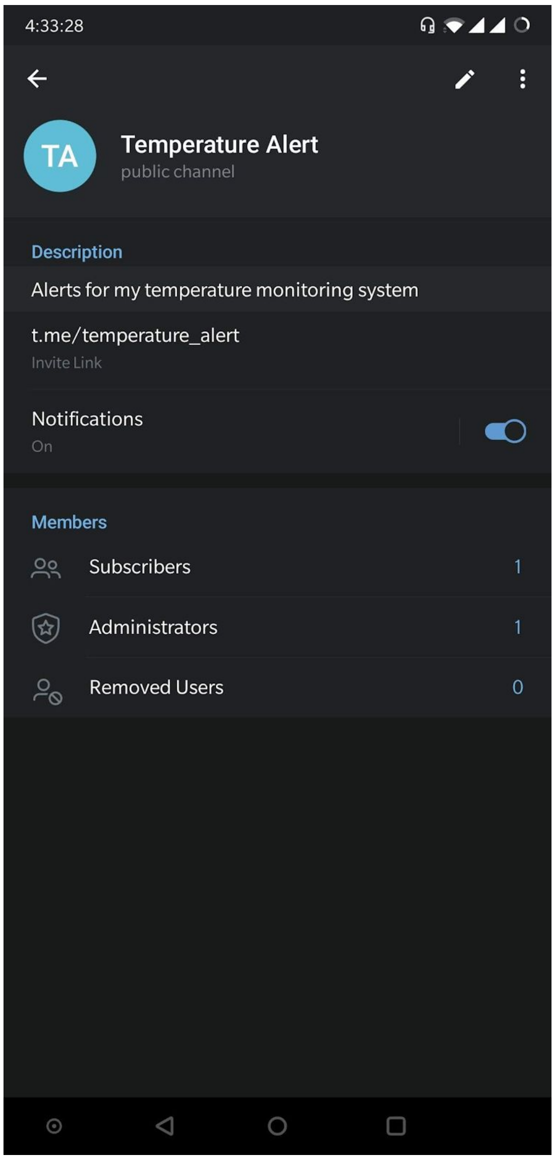
If you set a permanent link, other people will be able to find and join your channel.

You can use a-z, 0-9 and underscores.
Minimum length is 5 characters.

10. Congratulations! You have created a new channel. You can also view this channel from your app home screen.



11. You can click on the channel name at the top to view more details about it.


4:33:28

Temperature Alert
public channel

Description
Alerts for my temperature monitoring system

t.me/temperature_alert
Invite Link

Notifications
On

Members
Subscribers 1
Administrators 1
Removed Users 0

12. **Text after t.me/ is your Telegram chat ID.** For example. If my invite link is t.me/temperature_alert then my chat id would be temperature_alert and in your python code, you have to add @ before the chat id. For example.

    telegram_chat_id = "@temperature_alert"
13. The telegram_chat_id will be used later on in the python code to send the alerts. The python code will be taught in the later lessons.
14. Next, we will need to create and add a Bot to the channel so that it can post alerts for us on this channel.

Create a Bot

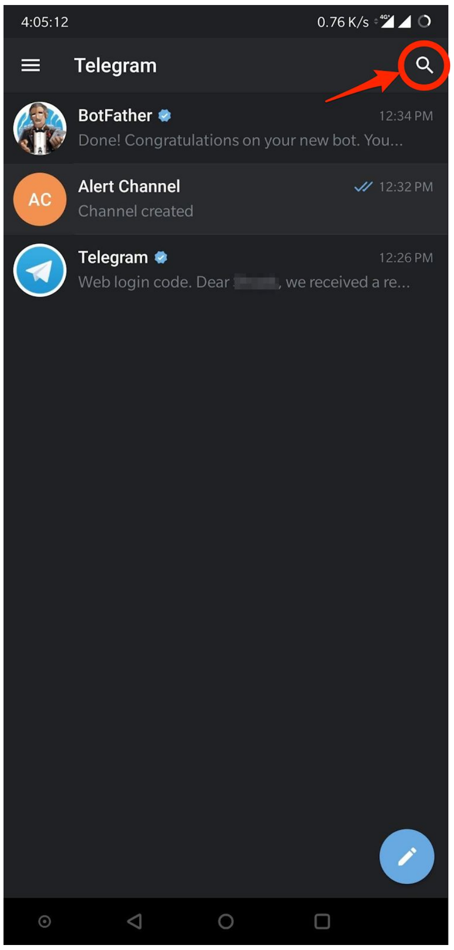
What is a Bot?

Bots are third-party applications that run inside Telegram. Users can interact with bots by sending them messages, commands and requests.
We will be using Bots to send alerts on our channel.

Create a new Bot

Telegram allows us to create a new bot with the help of a bot called "BotFather". It is a single bot to create and manage other bots.

1. On the Home screen of the app, click on the search icon on the top right and type in "botfather".

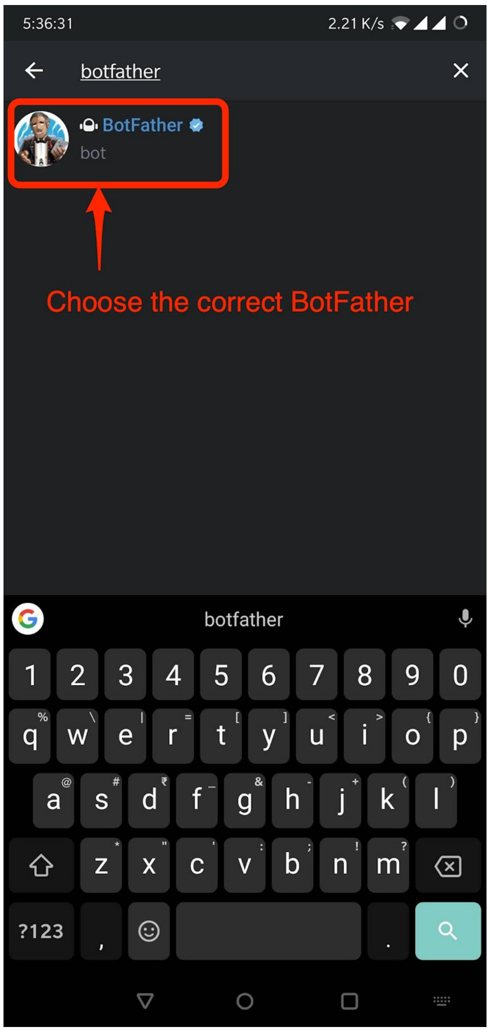


BotFather 12:34 PM
Done! Congratulations on your new bot. You...

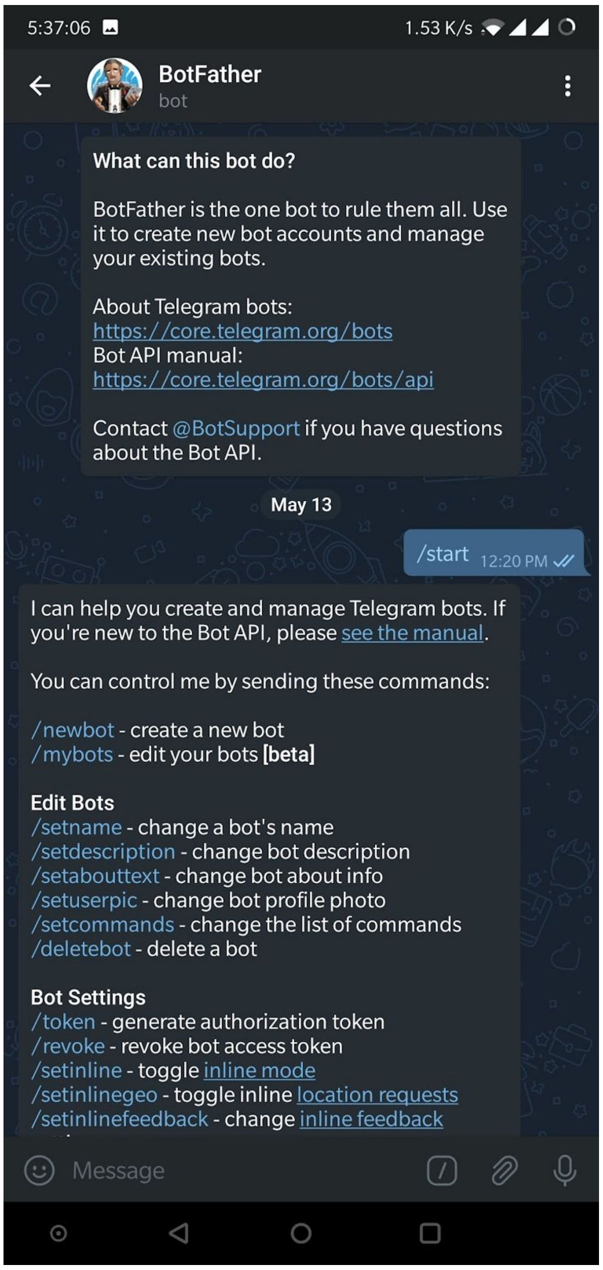
Alert Channel 12:32 PM
Channel created

Telegram 12:26 PM
Web login code. Dear [redacted], we received a re...

2. In the search results, click on the correct result for "BotFather" as shown below. The correct "BotFather" will have a blue tick mark next to its name. This will take you to a chat window with the "BotFather".



3. The chat will have a few items already and will display you a list of commands to get started.



5:37:06

1.53 K/s

BotFather

bot

What can this bot do?

BotFather is the one bot to rule them all. Use it to create new bot accounts and manage your existing bots.

About Telegram bots:
https://core.telegram.org/bots
Bot API manual:
https://core.telegram.org/bots/api

Contact @BotSupport if you have questions about the Bot API.

May 13

/start 12:20 PM

I can help you create and manage Telegram bots. If you're new to the Bot API, please see the manual.

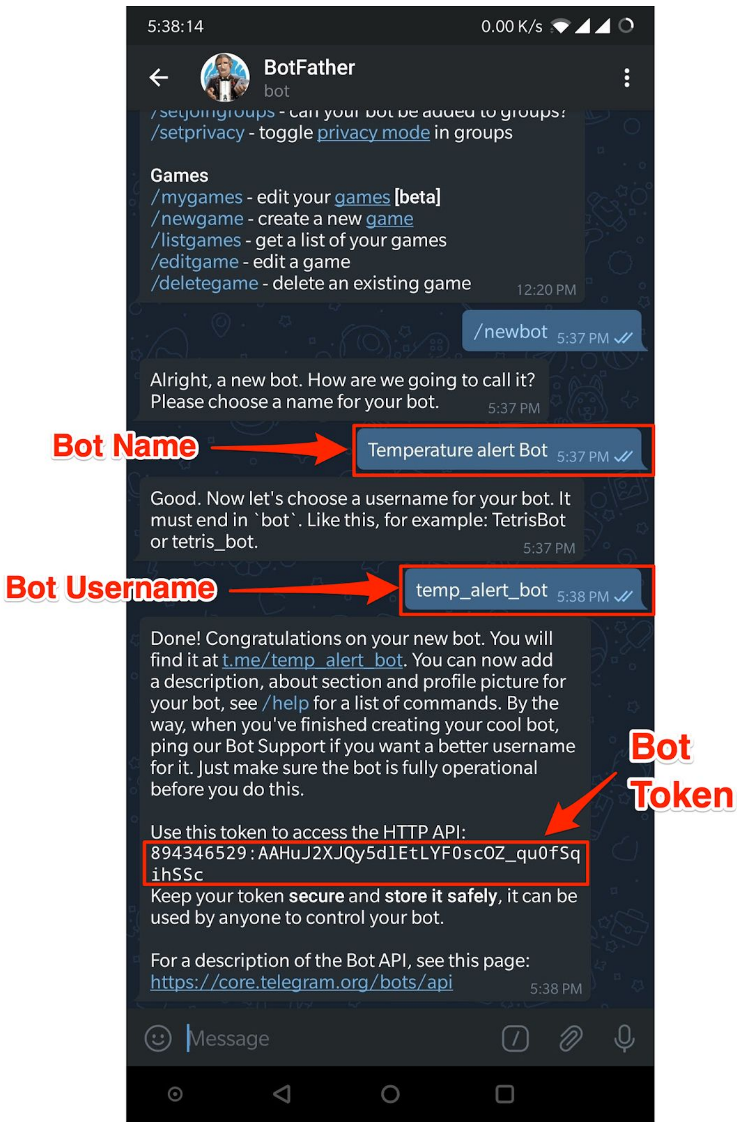
You can control me by sending these commands:

/newbot - create a new bot
/mybots - edit your bots **[beta]**

Edit Bots
/setname - change a bot's name
/setdescription - change bot description
/setabouttext - change bot about info
/setuserpic - change bot profile photo
/setcommands - change the list of commands
/deletebot - delete a bot

Bot Settings
/token - generate authorization token
/revoke - revoke bot access token
/setinline - toggle inline mode
/setinlinegeo - toggle inline location requests
/setinlinefeedback - change inline feedback

4. Since we need to create a new Bot, go ahead and type in "/newbot" in the window.
5. It will ask you to type in a few more details like Bot name and Bot username.
6. When your bot is created successfully, you will be shown a screen similar to the one below and will contain the Bot Token. This token is used to control your Bot as well as send messages on behalf of it. Please keep this token secure as it will allow anyone to access your Bot.


5:38:14 0.00 K/s

BotFather
bot

/setjoingroups - can your bot be added to groups?
/setprivacy - toggle privacy mode in groups

Games
/mygames - edit your games [beta]
/newgame - create a new game
/listgames - get a list of your games
/editgame - edit a game
/deletegame - delete an existing game 12:20 PM

/newbot 5:37 PM

Alright, a new bot. How are we going to call it? Please choose a name for your bot. 5:37 PM

Temperature alert Bot 5:37 PM

Good. Now let's choose a username for your bot. It must end in `bot`. Like this, for example: TetrisBot or tetris_bot. 5:37 PM

temp_alert_bot 5:38 PM

Done! Congratulations on your new bot. You will find it at t.me/temp_alert_bot. You can now add a description, about section and profile picture for your bot, see /help for a list of commands. By the way, when you've finished creating your cool bot, ping our Bot Support if you want a better username for it. Just make sure the bot is fully operational before you do this.

Use this token to access the HTTP API:
894346529:AAHuJ2XJQy5dlEtLYF0sc0Z_qu0fSqihSSc
Keep your token **secure** and **store it safely**, it can be used by anyone to control your bot.

For a description of the Bot API, see this page:
https://core.telegram.org/bots/api 5:38 PM

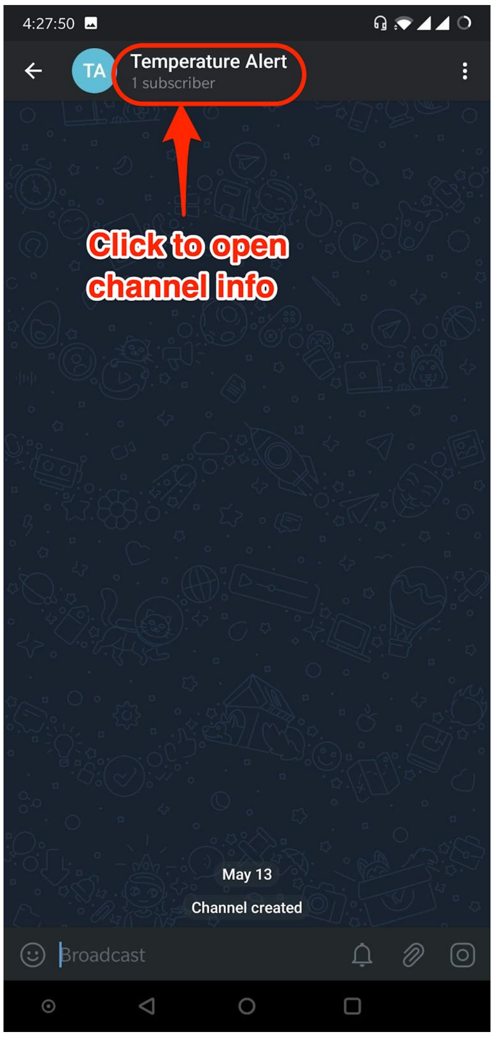
In the screenshot above, the Bot token is "894346529:AAhuJ2XJQy5dlEtLYF0sc0Z_qu0fSqihSSc". Please save it as telegram_bot_id. The telegram_bot_id is saved as "bot" followed by the bot token. So, in this example, the telegram_bot_id will be "bot894346529:AAhuJ2XJQy5dlEtLYF0sc0Z_qu0fSqihSSc".

7. The telegram_bot_id will be used in the python code to send messages. The python code will be taught in the later lessons.
   Also, please be careful when saving the Bot ID. You may get confused between 0 and O, I and 1 etc. as they are similar looking.

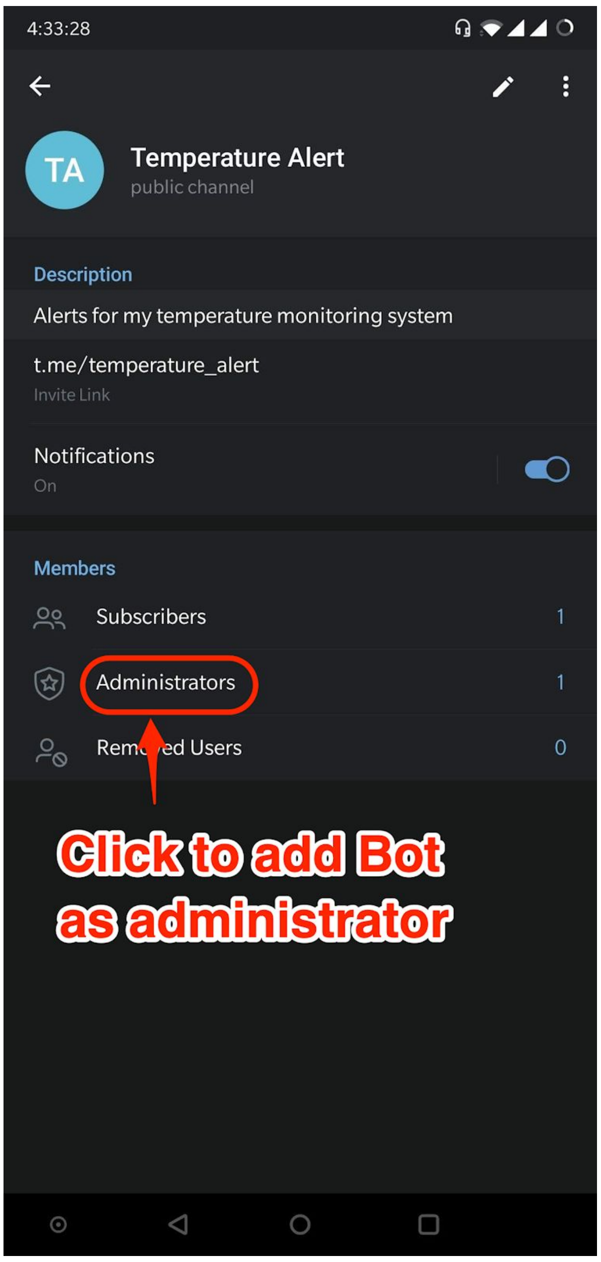
8. Congratulations, you have created a new Bot. Now, we will need to add it to the channel we have created previously so that we can send alerts.

Add the bot to a channel

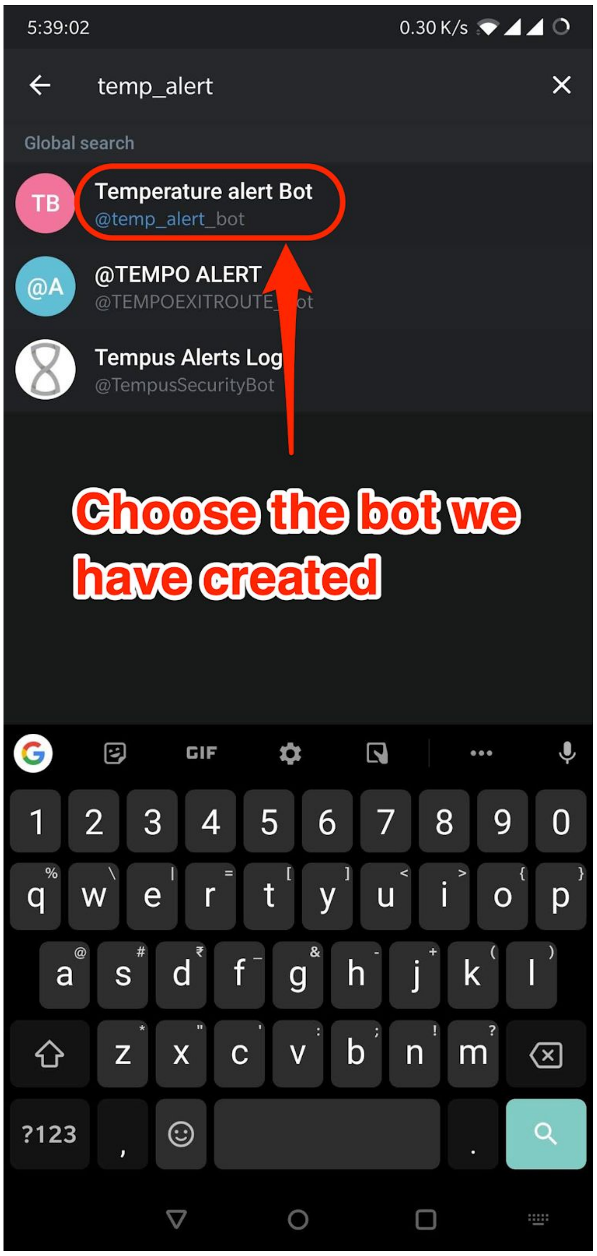
1. From the App home screen, open the channel we have created earlier.
2. In the screen, click on the channel name on the top to open the information for the Channel.



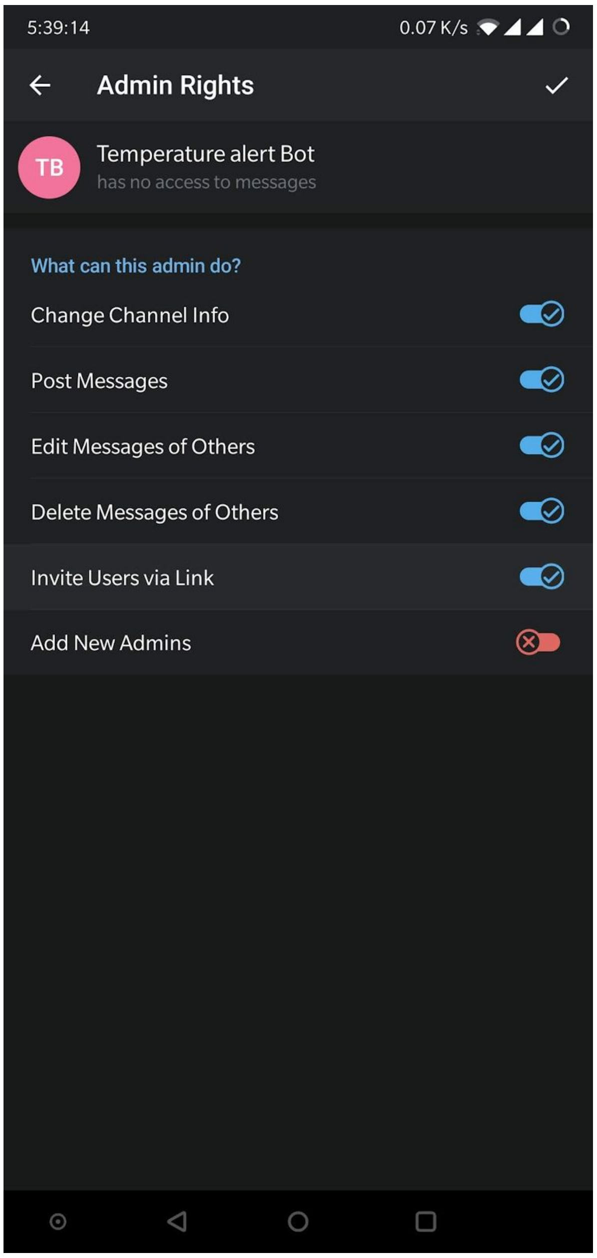
3. Click on the Administrators button so that we can add the newly created bot to the channel.



4. Search for the bot that we have created using the Bot's username. Once you have found the correct Bot click on it to add it to the channel. Please make sure you have clicked on the Administrators button in the previous step as this will allow our bot to post to the channel.



5. You will be asked to confirm the rights for the bot. Press on the tick mark on the top right to continue adding the bot to the channel. Make sure that the bot has the rights to "Post Messages".



5:39:14 0.07 K/s

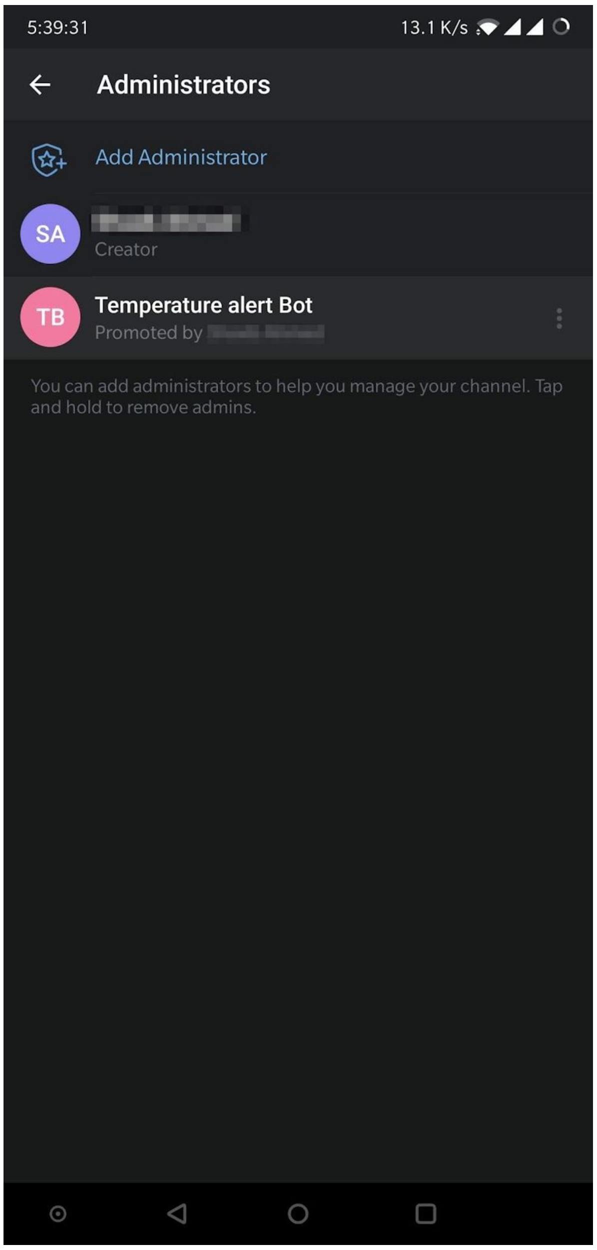
Admin Rights

Temperature alert Bot
has no access to messages

What can this admin do?

Change Channel Info ✓
Post Messages ✓
Edit Messages of Others ✓
Delete Messages of Others ✓
Invite Users via Link ✓
Add New Admins ✗

6. You will now see the newly created Bot in the list of administrators for the channel.



5:39:31 13.1 K/s

Administrators

Add Administrator

SA [redacted]
Creator

TB Temperature alert Bot
Promoted by [redacted]

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

7. Now we will code for sending messages to the channel via the Bot.

Sending Telegram message when temperature crosses threshold

Circuit connections

For this project, set up the circuit as we have done for the Temperature monitoring system as we have done earlier and power on the Bolt device.

Code

The code can be basically divided into three major parts. They are,

1. Reading values from the Bolt device
2. Comparing the sensor value with the threshold.
3. Sending a message via Telegram if the threshold has been breached.

The above steps are put in a while loop so that it reads the sensor value every 10 seconds and sends the alert if required.

We will be building the system functionally as,

1. Reading the sensor value from the device.
2. Sending a Telegram message.
3. Checking if sensor value is valid and sending a message if it exceeds the threshold.

How this works

Telegram is a third-party messaging service which is similar to WhatsApp. It is free of cost and simple to use.

Telegram has a feature called as Bots which are just customized applications which you can build in order to send messages to a particular channel or group.

So, if we want to send a message to a channel, we will need to send a request to Telegram instructing the bot to send a message to a particular group.

It's that simple. Lets learn how to do that in the next few lessons.

Reading sensor values from the Bolt device

Following the last part of the previous lecture, the code is divided into three major functions. We will be discussing and creating a function to read the sensor value from the Bolt device.

We will start by creating a new Python file to hold our configurations. Connect to your Ubuntu OS and type in the command `sudo mkdir temp_alert` in your shell to create a new folder for our project.

Then type `cd temp_alert` to move into that folder.

Create a config file to hold the configuration variables for Boltiot and Telegram. Type the command `sudo nano conf.py` to create a new file and edit it in the shell.

Paste the following lines of code into the conf file and press CTRL + X followed by ENTER key to save it.

```
"""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 similar to BOLTXXXX
where XXXX is some numbers
telegram_chat_id = "@XXXX"            # This is the channel ID of the created Telegram
channel. Paste after @ symbol.
telegram_bot_id = "botXXXX"           # This is the bot ID of the created Telegram Bot. Paste
after bot text.
threshold = 250                       # Threshold beyond which the alert should be sent
```

- bolt_api_key -> This is your API key for the Bolt Cloud. Visit cloud.boltiot.com to find your API key. Paste the API key inside the double inverted commas.
- device_id -> This is the ID of your Bolt device. It will be something like BOLT12345. Paste the device ID inside the double inverted commas.
- telegram_chat_id -> This is the channel ID which we have got when creating the channel on Telegram. It will be under the permanent link name of the channel. Paste the channel id after the @ symbol. Do not keep any spaces.
- telegram_bot_id -> This is the ID of the Telegram bot we have got when creating the Bot. It will be mentioned under the "BotFather" channel as “token”. Paste the token after the "bot" text. Do not keep any spaces.

- threshold -> This is the threshold beyond which the alert should be sent. Configure this value to receive alerts accordingly.

Add the configurations to your conf.py file and save it.

Now we will need to create a new python file using the command `sudo nano telegram_alert.py`.

Add the following code and save it using CTRL + X followed by ENTER key.
Remember, this is not the final code. It is only a part for sending alert messages via Telegram.

```
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
```

In the above code, we have declared a function `get_sensor_value_from_pin()` which returns the sensor value from the selected pin. In case of any error, it returns -999 which we have declared as an error response.

It takes one parameter(pin) and makes a request to the Bolt Cloud to fetch the latest sensor value from the mentioned pin. The line `mybolt.analogRead()` returns the value for that pin.

The response returned by the Bolt Cloud needs to be converted into a JSON object for ease of use. The line `data = json.loads(response)` converts the response from the Bolt Cloud into a JSON object.

Before returning the value of the response, we need to check if the request was made successfully. The Bolt Cloud returns a status of 1 if the request was made successfully and anything else apart from 1 means that the request has failed. The line `if data["success"] != "1":` checks for a valid response.

We can finally get the sensor value from the "value" field inside the response, convert it to an integer and return it.

The function is encased inside a try-except block to handle any exceptions and errors. In case of any error or exception inside the try block, the except block is executed. This will return a response of -999 in case of any error. For more on exception handling in python, please visit https://www.programiz.com/python-programming/exception-handling.

In the next lesson, we will learn how to send messages via Telegram.

Send message via Telegram

Previously we have learnt how to get the sensor value from the Bolt device. In this part, we will write the code to send a Telegram message.

We will need to write a function to send a request to Telegram to send a message to the Telegram channel we have created earlier.

Add the function below inside the file "telegram_alert.py" and save it.
Remember that this not the final code. It is only a part of the final program.

```
def send_telegram_message(message):
    """Sends message via Telegram"""
    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 occurred in sending the alert message via Telegram")
        print(e)
        return False
```

In the above code, we have created a function to send a message via Telegram. The function `send_telegram_message()` takes one parameter i.e. the message to send.

In the first line, we are building the URL so that Telegram knows which bot it has to send the message to. The telegram_bot_id is required for this purpose.

The data variable is a dictionary which holds the chat ID(Channel ID) so that the Bot knows which channel it has to post the message to. It also contains the text message to be sent as a message.

In the next step, we need to make an HTTP request to the Telegram servers using the URL we have built earlier. The request is a "POST" request which contains all the relevant data like URL and the data to be contained in the request.

Next, we are just printing the response from the request as a text form. We also need to convert the text data to JSON data so that we can find the status of the request to Telegram.

The line `telegram_data = json.loads(response.text)` converts the text response to a JSON object and stores it in telegram_data.

The status of the request is stored in the "ok" field of the telegram_data variable and we are returning it. The "ok" field will always contain a boolean value i.e. True/False and is True if the message has been sent.

The function is encased in a try-except block so that any errors are caught and it returns a False if any error is present in the try block.

To reiterate, the function takes a message parameter to be sent and returns a True/False status depending on the status of the request to Telegram.

Now, we will simply need to call this function inside the block where we will detect that the sensor value has exceeded the threshold.

Head over to the next lesson to actually send messages when the temperature exceeds the threshold.

Code explanation - Part 3

In the previous lessons, we have written a function to return us the value of the sensor from the Bolt device and a function to send a Telegram message. In this final lesson, we will get started for coding the final part of the program i.e. "Comparing sensor value with the threshold and sending alert message".

Open the telegram_alert.py file using the command `sudo nano telegram_alert.py`. Add the code below and save it.

In this code, we will be comparing the sensor value we have received against our set threshold value to check if the sensor value is a valid one or not and if an alert needs to be sent.

This can be broken down into the following steps -
1. Get the sensor value from the Bolt device.
2. Check if the sensor value is valid or not.
3. Check if the sensor value exceeds the threshold and if a message needs to be sent.
4. Wait for 10 seconds.
5. Repeat from step 1.

Remember, this is not the final code. It is only a part of the code for sending alerts via Telegram.

```
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)
```

In the code above, we have declared a While loop to run infinitely. A while loop will run as long as the condition is true. Since we have declared the condition as True, the value of True will never change. Hence the loop will never exit.

Inside the while loop, we have used the function `get_sensor_value_from_pin()` which we have written in the last lesson to get the sensor value from the pin A0 of the Bolt device.

Now, we need to first check if the response is valid by checking if the response is equal to -999 which we have defined as our error response in the function `get_sensor_value_from_pin()`.

If the response is exactly -999, we know an error has occurred. The line "if sensor_value == -999" check for this condition. At this point, we can print a message and skip this reading. The program waits for 10 seconds and then continues to the next iteration of the while loop. The continue statement skips all the code after its invocation and goes into the next iteration of the loop.

Now, we know that the response is valid, we need to check if the sensor value has exceeded the threshold that we have defined in the conf.py file.

A simple conditional statement checks if the sensor value has exceeded the threshold. The line "if sensor_value >= conf.threshold:" does this. Inside, we are printing a message saying that the threshold has been breached, constructing the message to be sent and invoking the function to send the alert message via Telegram which we have created in the previous lesson.

The program waits for 10 seconds and then continues to the next iteration of the while loop.

This completes the three part process of sending alert messages when the temperature exceeds the threshold.

Try building the complete program on your own before proceeding to the next lesson. Don't worry, the entire code will be given in the next lesson if you are facing any difficulty along with screenshots.

NOTE - To exit from a While loop, press CTRL + C on your keyboard.

Complete Code

Code for conf.py

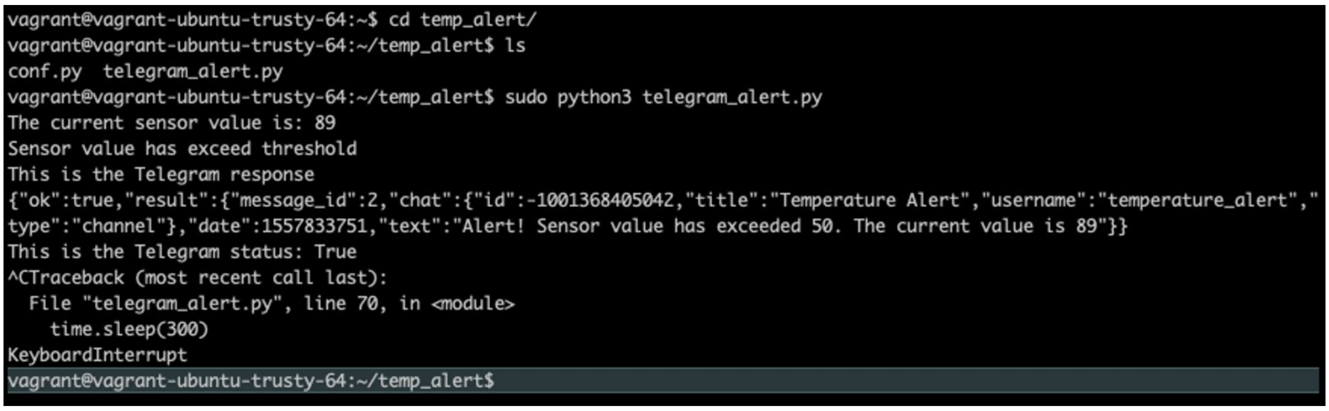
```
"""Configurations for telegram_alert.py"""
bolt_api_key = "XXXX"              # This is your Bolt Cloud API Key
device_id = "XXXX"                 # This is the device ID and will be similar to BOLTXXXX where
XXXX is some numbers
telegram_chat_id = "@XXXX"         # This is the channel ID of the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

Code for telegram_alert.py
NOTE - To exit from a While loop, press CTRL + C on your keyboard.

```
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 + "/sendMessage"
    data = {
        "chat_id": conf.telegram_chat_id,
        "text": message
    }
    try:
        response = requests.request(
            "GET",
            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)
```

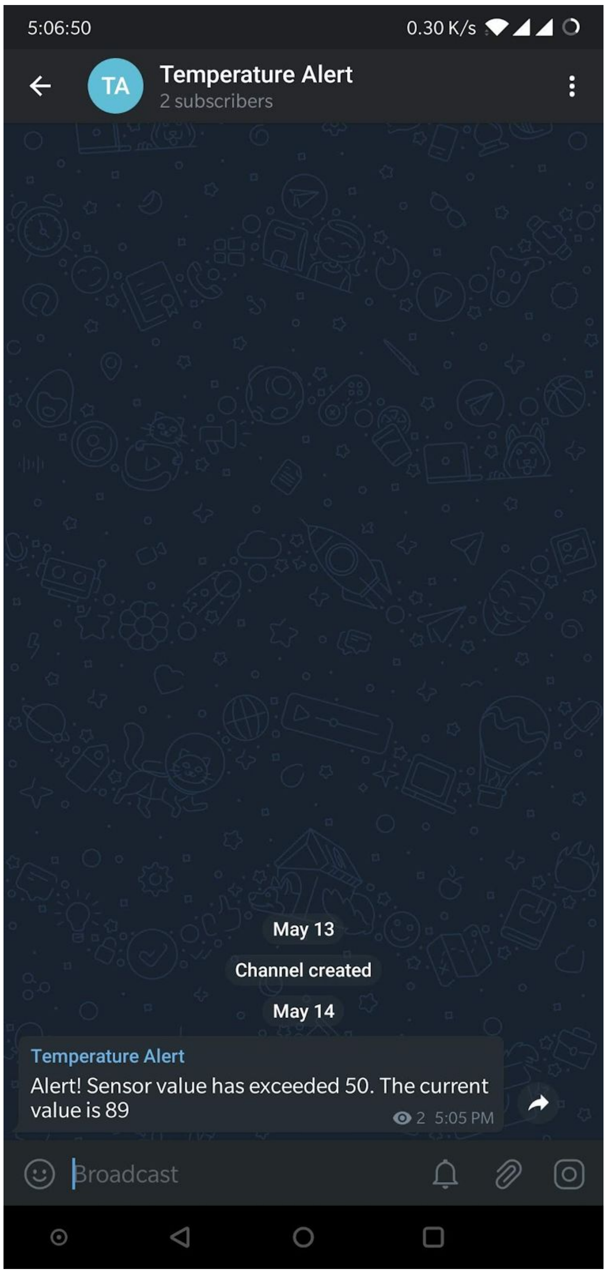
Sample Output

Running the script -



```
vagrant@vagrant-ubuntu-trusty-64:~$ cd temp_alert/
vagrant@vagrant-ubuntu-trusty-64:~/temp_alert$ ls
conf.py  telegram_alert.py
vagrant@vagrant-ubuntu-trusty-64:~/temp_alert$ sudo python3 telegram_alert.py
The current sensor value is: 89
Sensor value has exceed threshold
This is the Telegram response
{"ok":true,"result":{"message_id":2,"chat":{"id":-1001368405042,"title":"Temperature Alert","username":"temperature_alert","
type":"channel"},"date":1557833751,"text":"Alert! Sensor value has exceeded 50. The current value is 89"}}
This is the Telegram status: True
^CTraceback (most recent call last):
  File "telegram_alert.py", line 70, in <module>
    time.sleep(300)
KeyboardInterrupt
vagrant@vagrant-ubuntu-trusty-64:~/temp_alert$
```

Getting the output on the app -



5:06:50 0.30 K/s

Temperature Alert
2 subscribers

May 13
Channel created
May 14

Temperature Alert
Alert! Sensor value has exceeded 50. The current value is 89
2 5:05 PM