| **Job Sheet** |
| --- |

| **Module** | **:** | IoT Fundamentals | **Module Code** | **:** | EC33105FP | |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  | |
| **Job No** | **:** | 10 | **Duration** | **:** | F/T: | 7 hours |
|  |  |  |  |  | P/T: |  |

| **Job Title** | **:** | Visualize Data and Setup Alerts on IoT Platform |
| --- | --- | --- |
|  |  |  |
| **Objectives** | **:** | 1. Prepare RPi and Upswift 2. Connect Ultrasonic and LED to RPi Pins 3. Perform Data Extraction and Visualization 4. Setup Alerts **- OPTIONAL!** |

**Tools, Equipment and Materials**

| PC |
| --- |
| RPi with microSD |
| RPi Power Adaptor  HC-SR04+ x 1  1 kohm x 3  220 ohm x 1  LEDs(red ) x 1  Breadboard  4G WiFi Router |
|  |

**Number of Tasks to Complete**: 8

**Part A: Prepare RPi and Upswift (0.5 hour)**

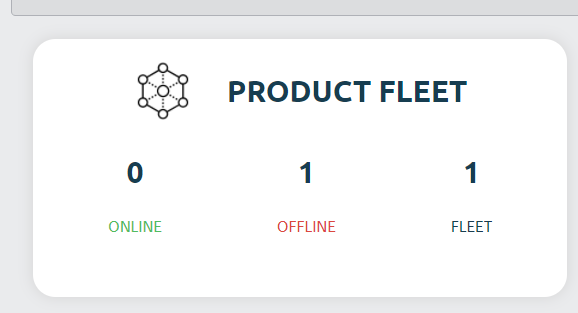
Before performing any hardware configurations, we need to ensure that the RPi is accessible remotely.

YOU MUST HAVE COMPLETED JOB 1 BEFORE DOING THIS

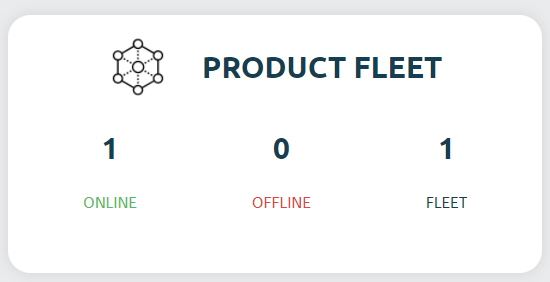
**Task 1 – Prepare RPi and UpSwift**

Perform the following steps to setup RPi:

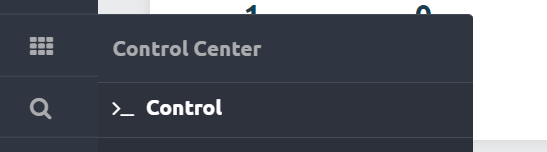
1. Connect only the power adaptor to RPi.
2. Turn on the power
3. Go to <https://dashboard.upswift.io/dashboard/> and login. The device would be **offline**.



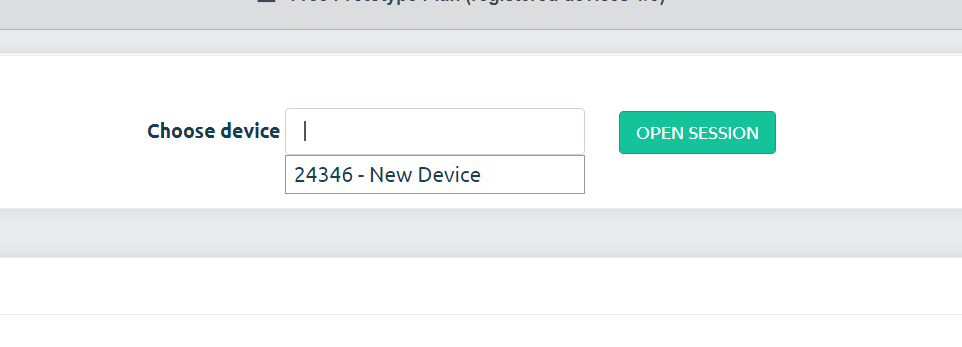
1. Wait for a while and refresh the web page by clicking . Device should appear online after 1 - 2 mins.



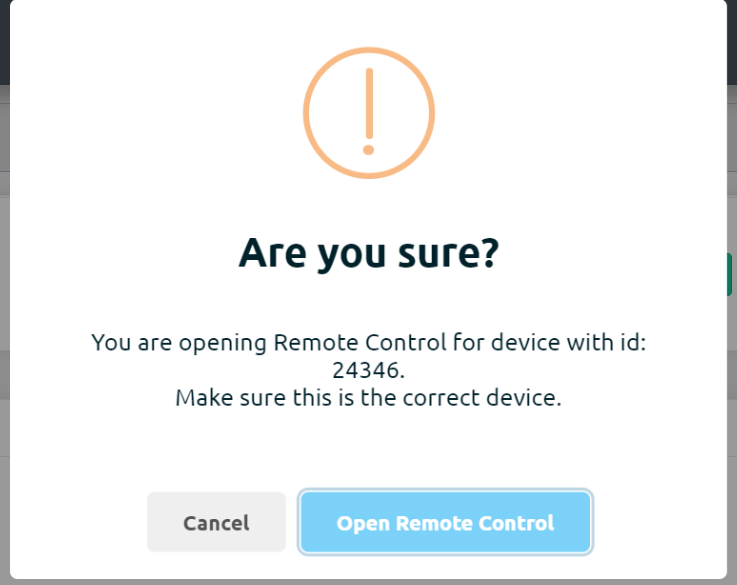
1. On the left side of your menu, click on the **Control** option under Control center.



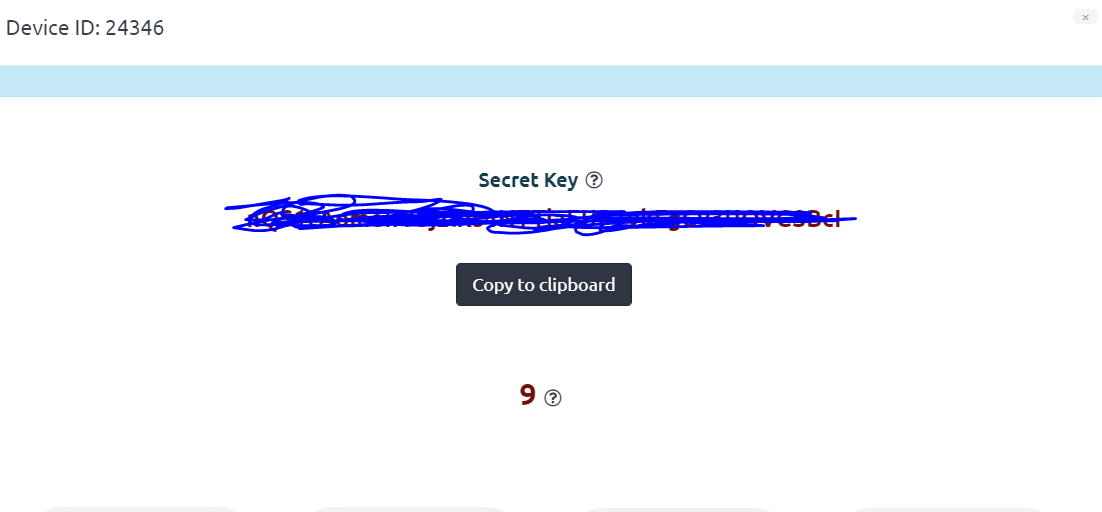
1. On the page center, select your device and click open session.



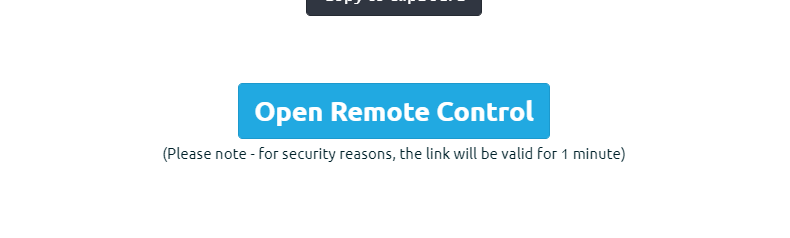
1. For the prompt, click **Open Remote Control.**



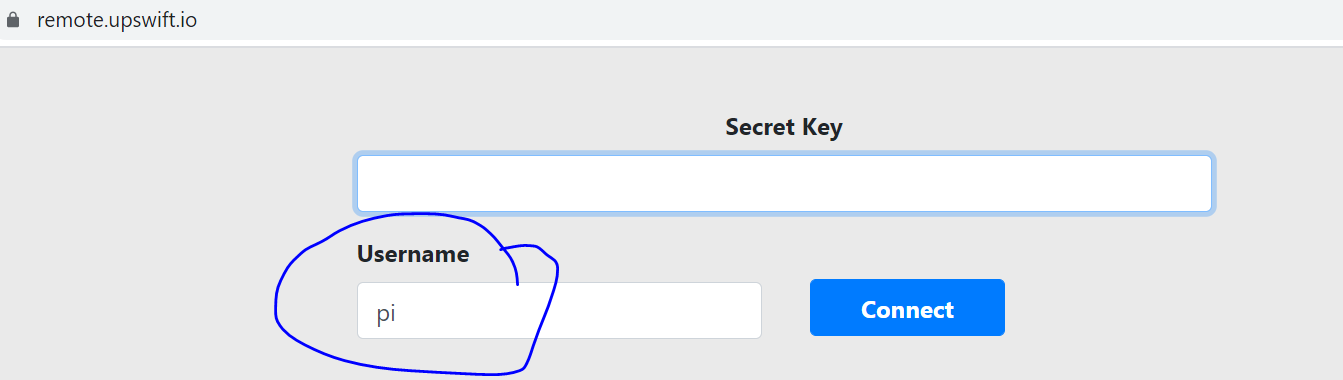
1. You will be shown a secret key. Click on **Copy to clipboard** button

****

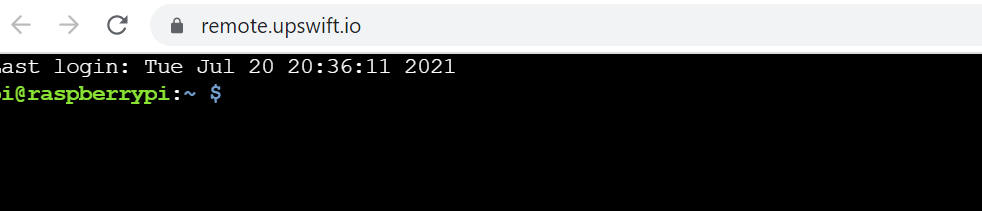
1. Once the timer is up, click on the **Open Remote Control** button.

****

1. **Paste** the secret key in the input text box and change username to pi. Click on **Connect**.

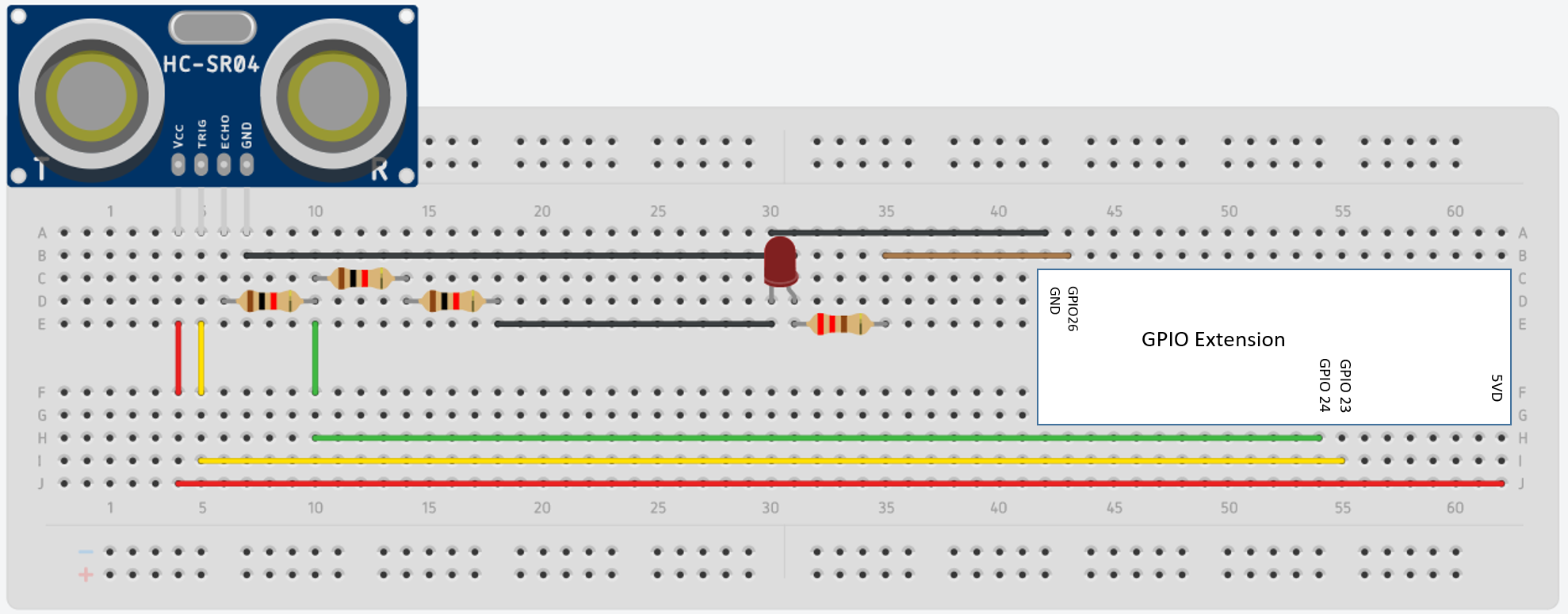
****

1. You will be presented with a browser based RPi terminal.

****

**Part B: Connect Ultrasonic and LED**

Connect the ultrasonic sensor (HC SR04) to the RPi’s GPIOs following the schematic below :



**Task 2 – Connect Ultrasonic and LED**

Perform the following steps

1. Connect Ultrasonic Vcc to GPIO 5V (Pin 1).
2. Connect Ultrasonic TRIG to GPIO 23 (Pin 16).
3. Connect Ultrasonic ECHO to 1k resistor.
4. Use two 1 k resistors in series to form a 2 k resistor (1k + 1k = 2k).
5. Connect 2k resistor to 1 k resistor and GPIO 24 (Green wire).
6. Connect 2k resistor to Ultrasonic GND and Ultrasonic GND to RPi GPIO GND.
7. For LED, connect GPIO 26 to 220 ohm resistor
8. Connect other end of 220 ohm resistor to Anode(+) of LED
9. Connect Cathode(-) of LED to GND

**Part C: Perform Data Extraction and Visualization**

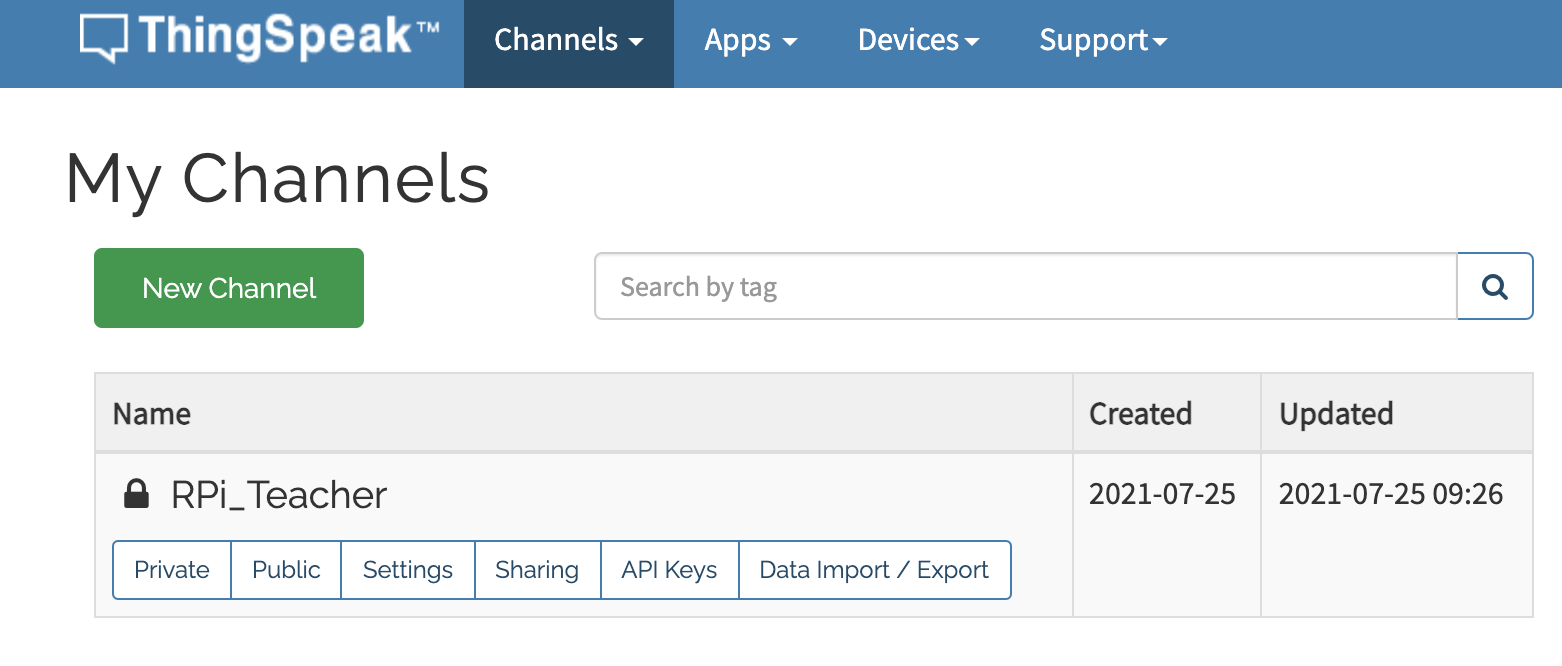
We will be using a HTTP GET request to send the distance and LED status data to the ThingSpeak Channel. From there, we will customize the visualization to show the data.

In order to do that, first we need to update the channel fields, get a write key, execute the program, and add widgets. The following tasks cover this.

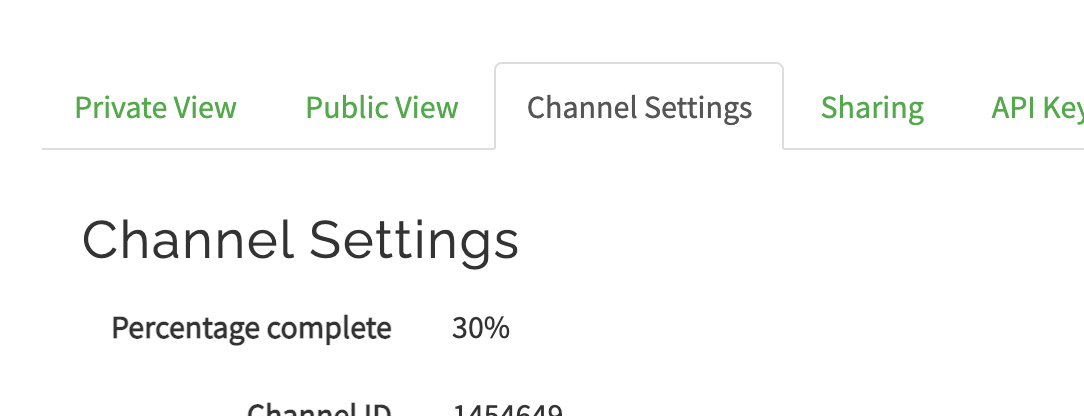
**Task 3 – Update Channel for Ultrasonic and LED**

Perform the following steps.

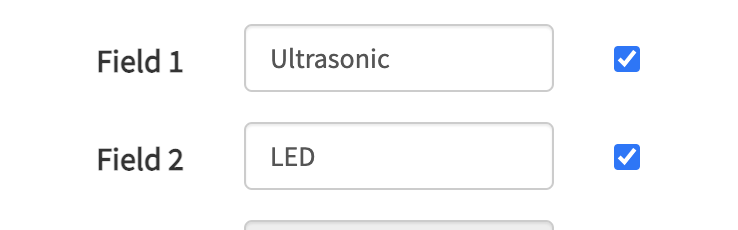
1. Login to <https://thingspeak.com/>
2. Look for your Channel created in Jobsheet 8.



1. Click on **Settings / Channel Settings.**

****

1. Update the fields to **Ultrasonic** and **LED**.

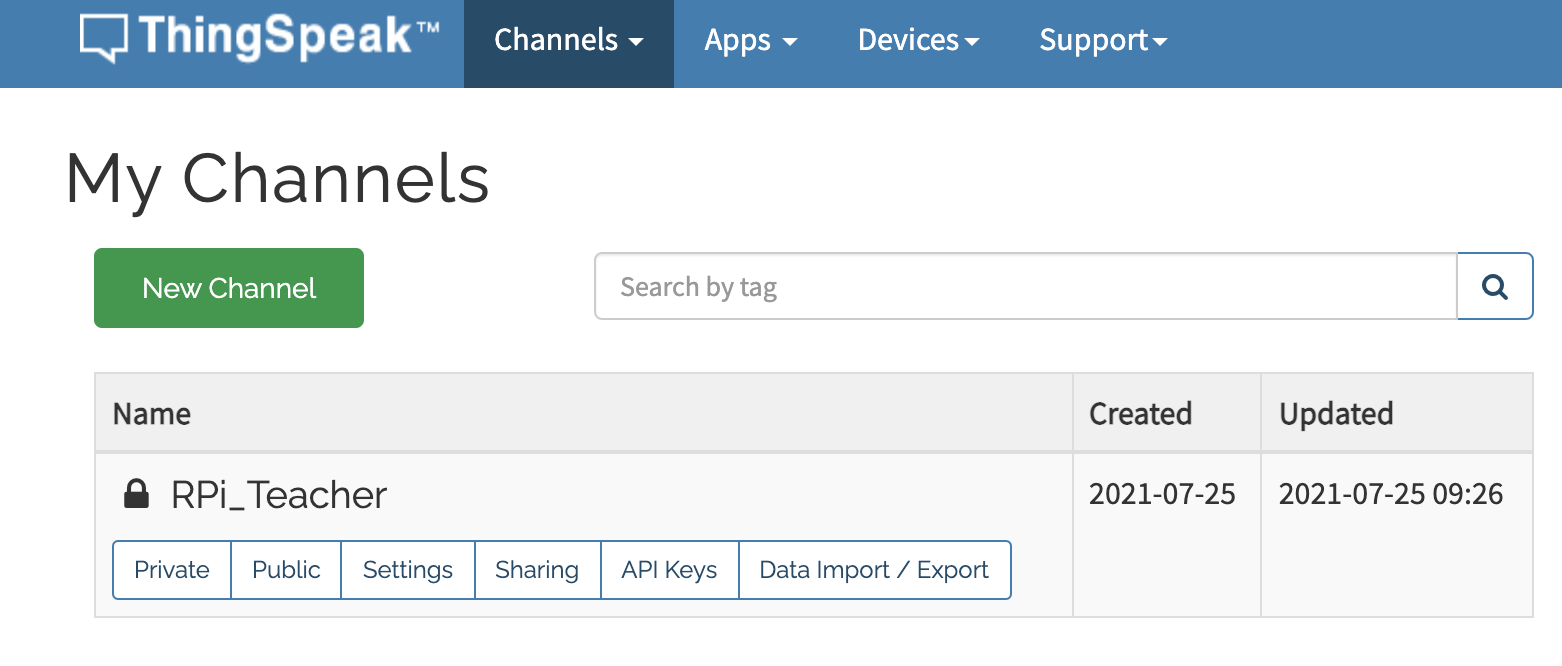


1. Scroll down, and click on 

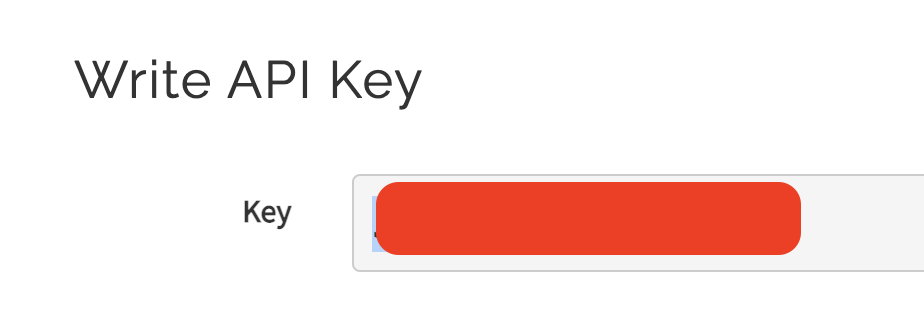
**Task 4 – Get Write Key from ThingSpeak**

Perform the following steps :

1. Login to <https://thingspeak.com/>
2. Look for your Channel created in Jobsheet 8.



1. Click on **API Keys.**
2. Copy the **Write API Key.** We need to use it later in the python code on the RPi.

****

**Task 5 – Install Requests Library (skip this if requests has been installed)**

You can skip this step if the requests library has already been installed. If unsure, you can still perform the following steps.

In order to send data by making a HTTP request, we need to use a python library to help us. The requests library makes sending HTTP requests simple, with one line of code.

<https://pypi.org/project/requests/>

Perform the following steps.

1. In the RPi, run **pip install requests**

****

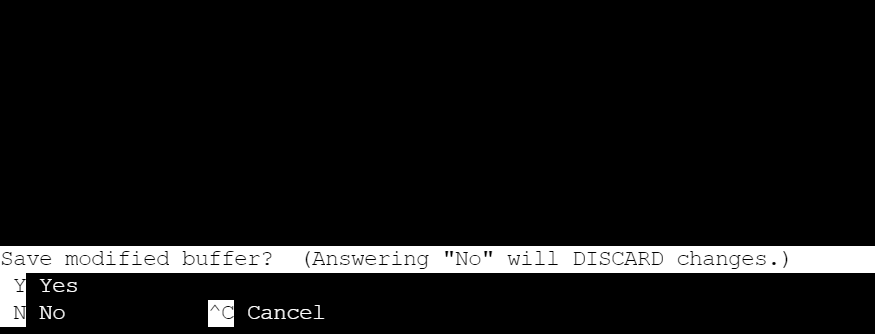
**Task 6 – Send data to ThingSpeak**

Perform the following steps.

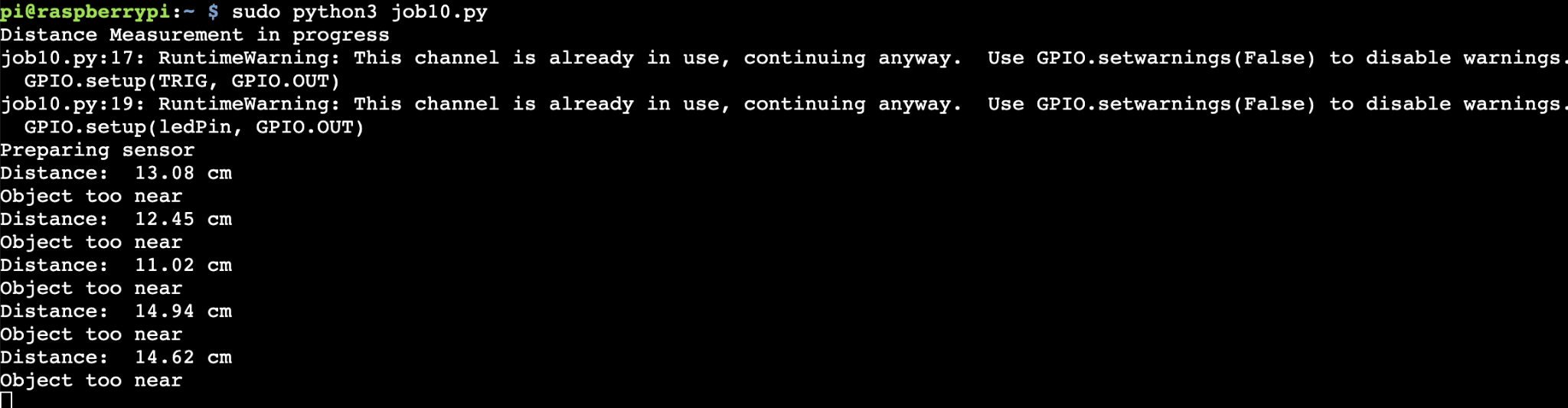
1. Run **sudo nano job10.py.**
2. Copy the code found at <https://github.com/Prakashash18/iot-fundamentals/blob/main/job10.py>
3. Paste it into the nano editor by right clicking and pasting. Replace the YOUR\_API\_KEY with the **Write API Key** copied earlier.



1. [Keyboard] Press CTRL + X to save your work. You will be prompted with the following message.



1. [Keyboard] Press **y** for yes and hit **Enter**
2. On the terminal, run **sudo python3 job10.py** to execute the code. Note that we are using python version 3.
3. You should see the output on the terminal

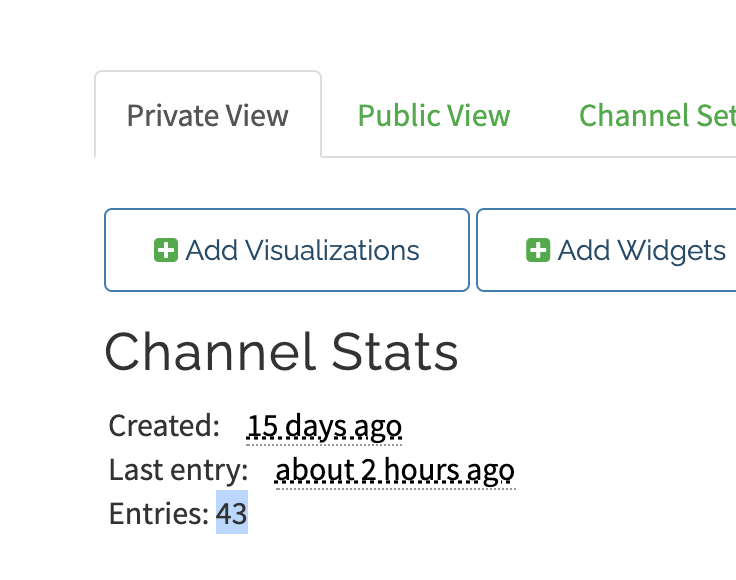


1. Back onThingSpeak dashboard under **Private View**, every 15 seconds, you’d see data being shown as a line chart.

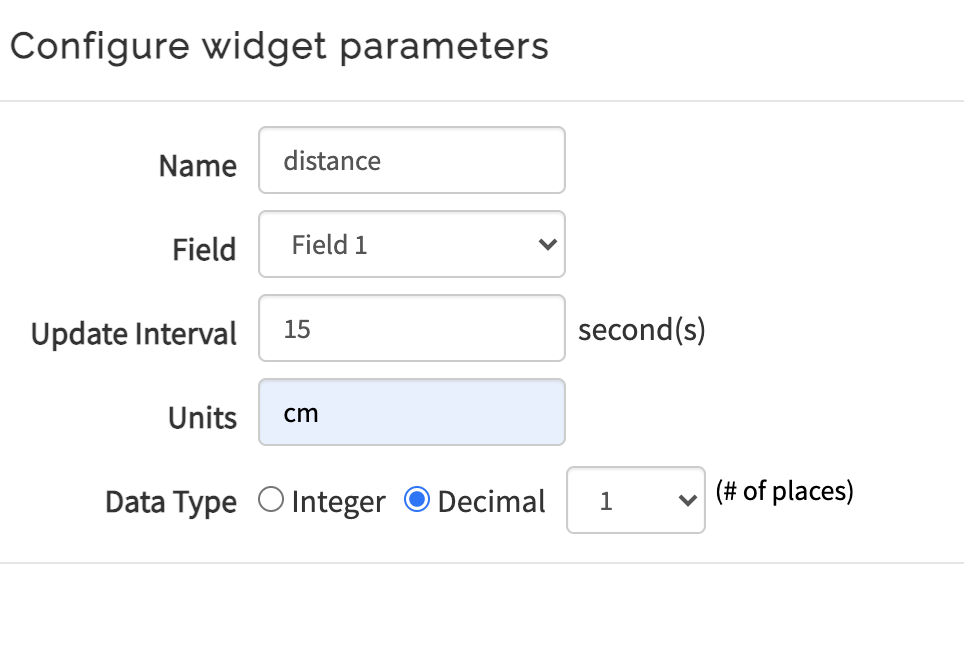
**Task 7 – Add Visualizations**

Line charts can’t represent the distance and LED status data well as line charts are used to establish trends. In the following part, we are going to customize the visualizations in ThingSpeak to better represent the distance and LED status data.

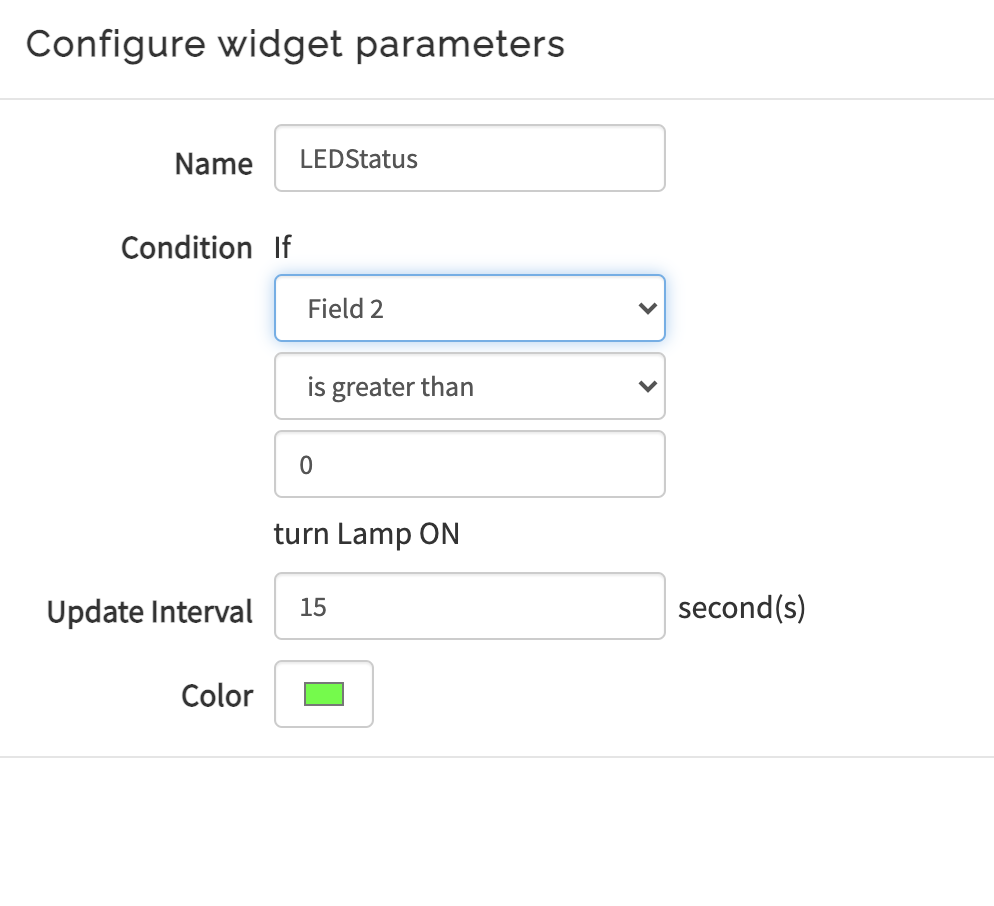
1. Click on **Private View**

****

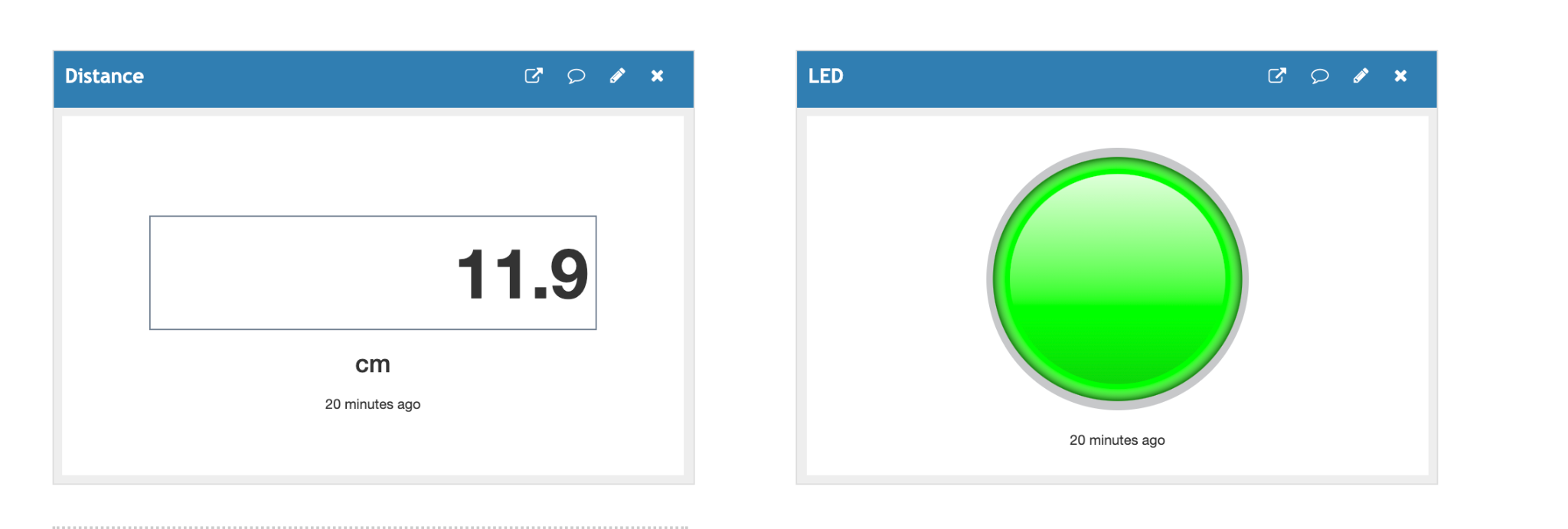
1. Remove all visualizations present on the Private View by clicking on the 
2. Click on .
3. Select the **Numeric Display** and click **Next.**
4. Configure the widget parameters as such



1. Click on **Create**
2. Click on .
3. Select the **Lamp Indicator** and click **Next.**
4. Configure the widget parameters as such



1. Click on **Create**
2. The following should be presented to you. Test if the distance reading and lamp status are accurately shown (Note: there is a 15 sec delay, so you might need to wait 15 secs for the data to be captured).



**Part D: Setup alerts (Optional)**

Alerts are useful to notify users when events occur. In ThingSpeak, there are many ways to create alerts. Some ways are using Twitter to send a tweet and sending emails. In the following tasks, we are going to send an email anytime an object is detected by the ultrasonic sensor.

To do this, we need to create a MATLAB code App that will be used to send emails.

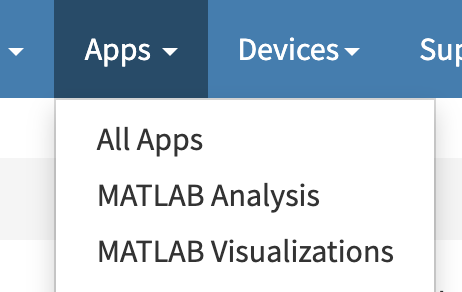
**Task 8 – Create App**

Perform the following steps.

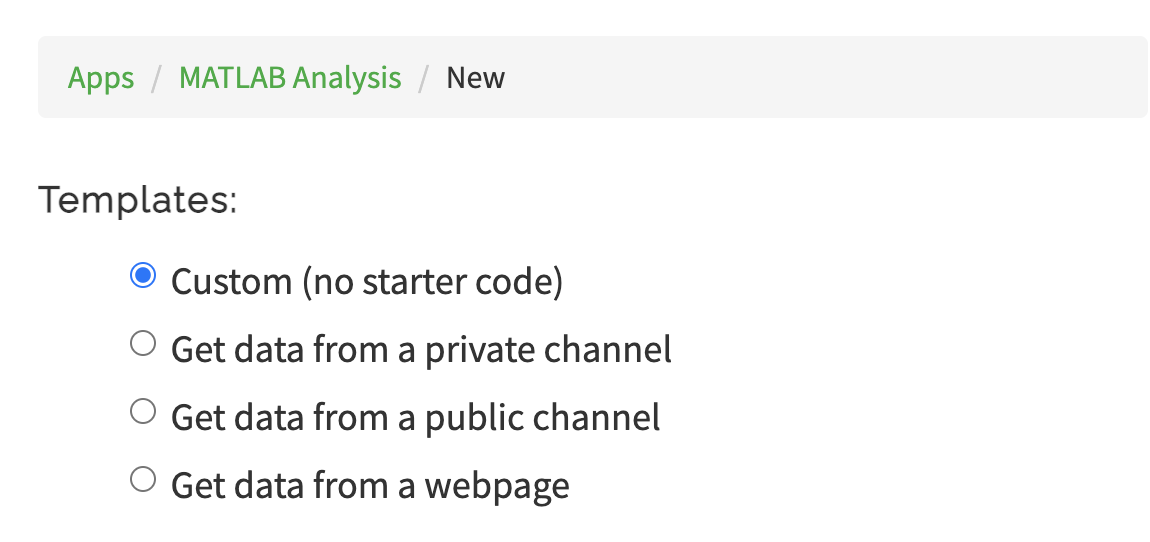
1. Click on **Apps**.

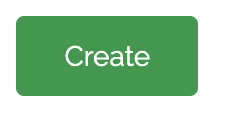


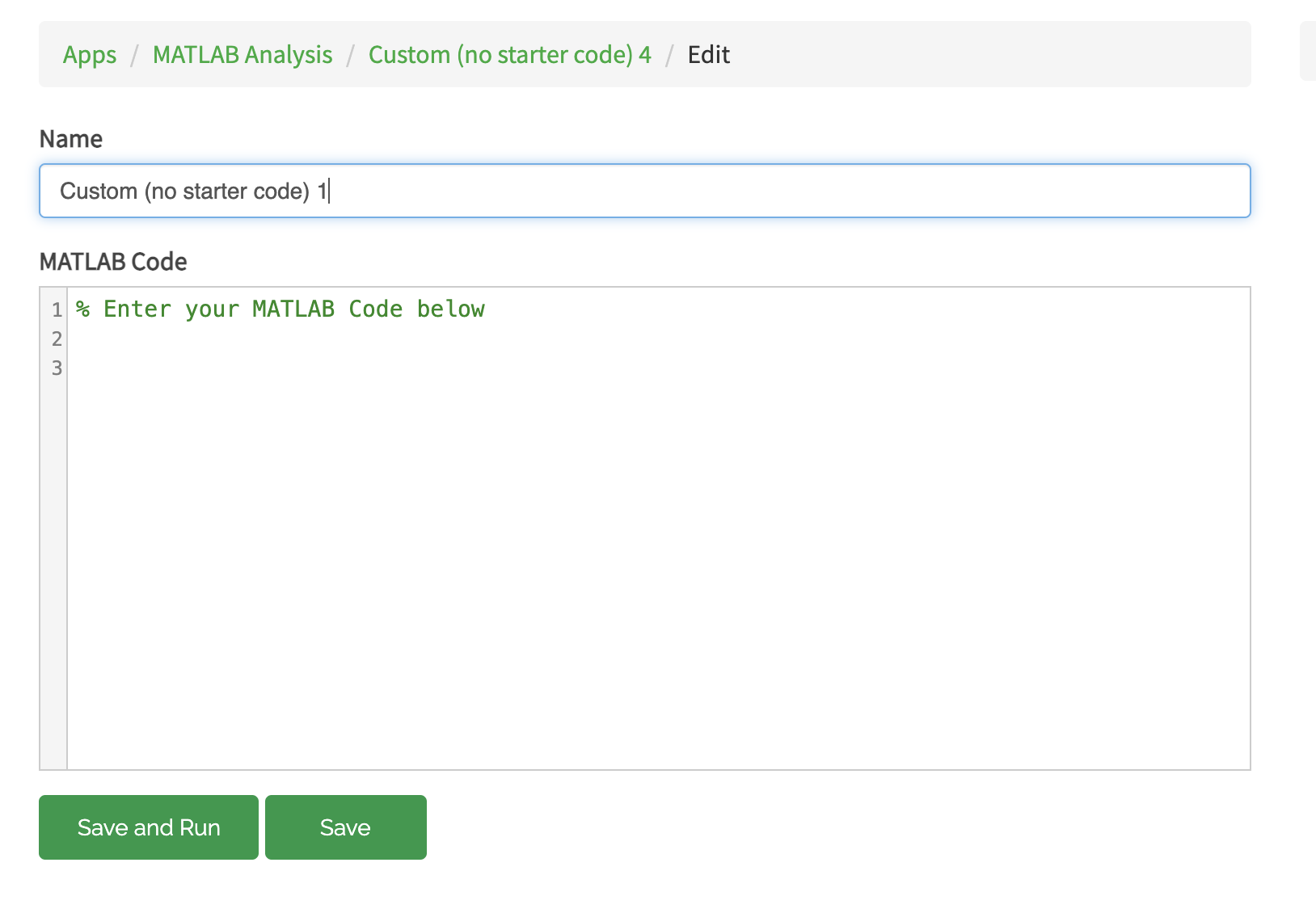
1. Select **MATLAB Analysis**.



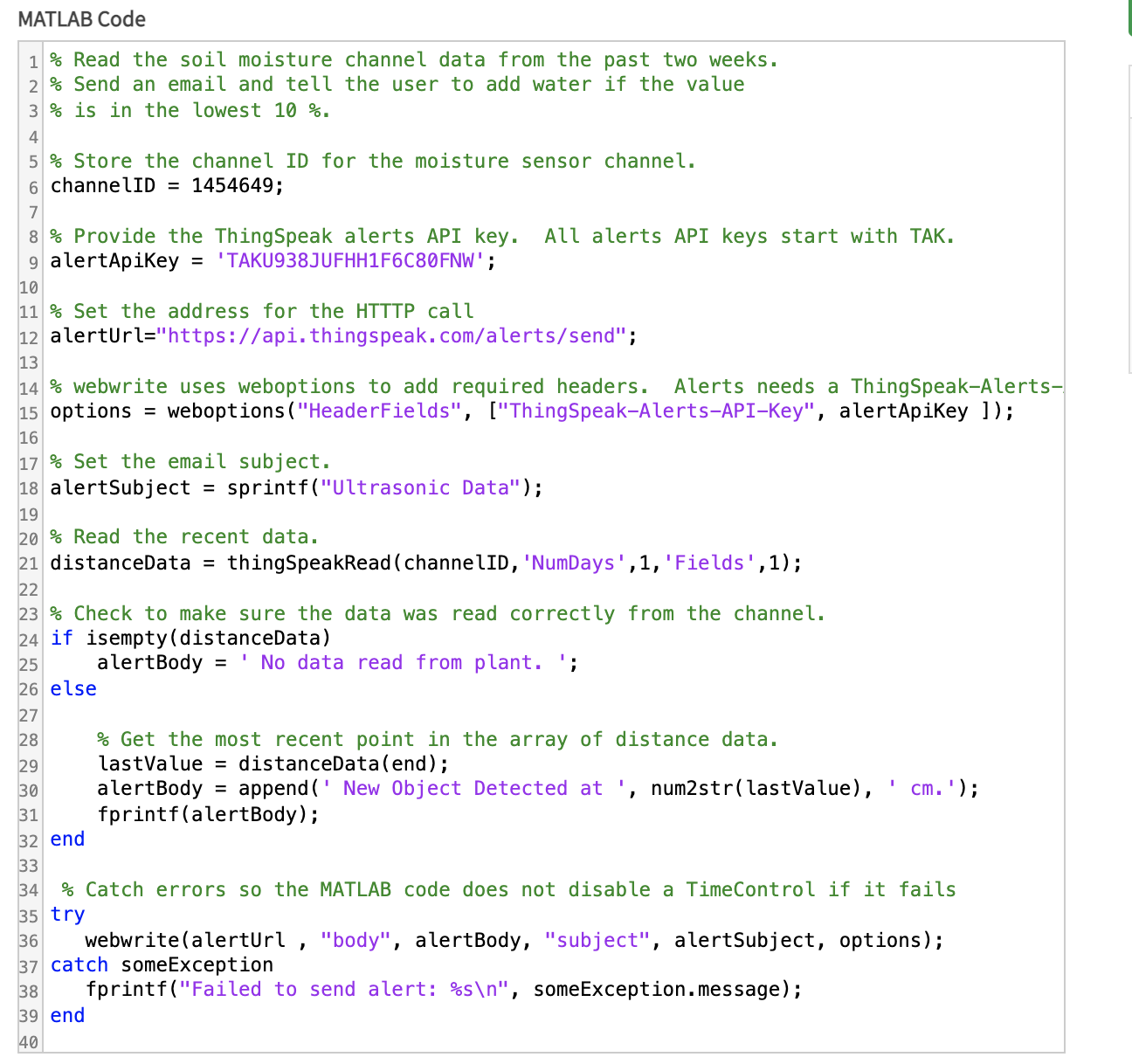
1. Click on ****
2. Under Templates, select **Custom(no starter code)**



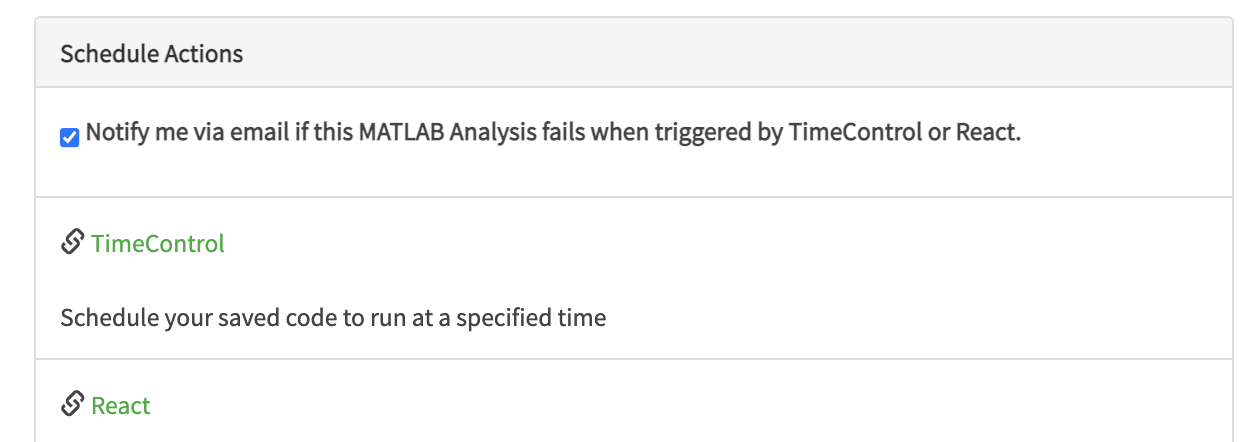
1. Click 
2. The empty MATLAB code will be shown here.



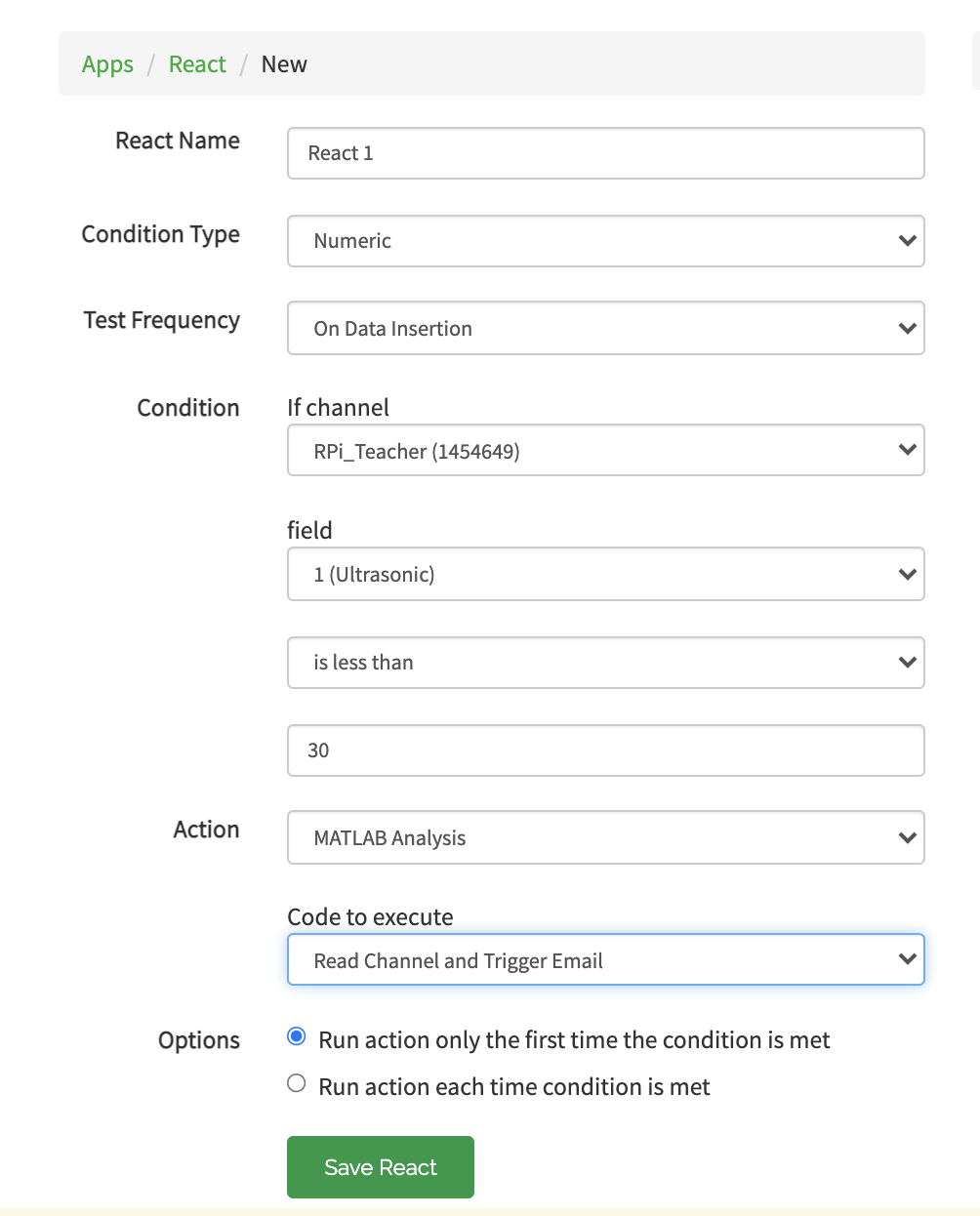
1. Update the MATLAB code as such. You would need to replace the Channel ID and alert API key

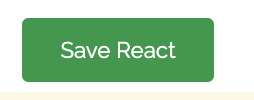


1. Click 
2. Schedule this email by clicking on **React**.



1. Fill up the React Parameters as shown



1. Click 
2. You should receive an email with the distance reading as such when an object is placed less than 30 cm from the sensor.

