

Connected Devices

Semester Project

Name and Course

- Name: Paavan Gopala Reddy
- Course: Connected Devices
- Semester and Year: Spring 2019

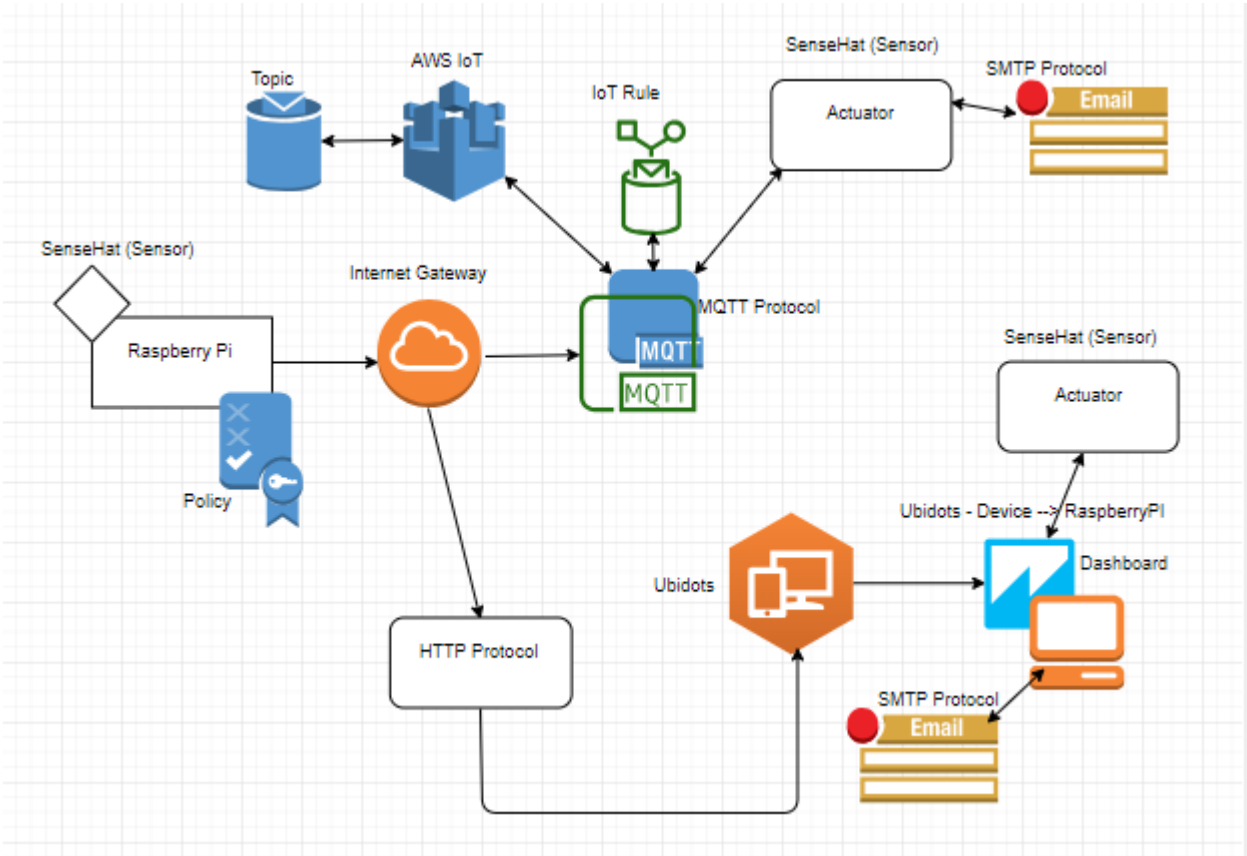
Description

- Description: **Streaming Sensor Data (Raspberry Pi and SenseHat) to AWS IoT and Ubidots**

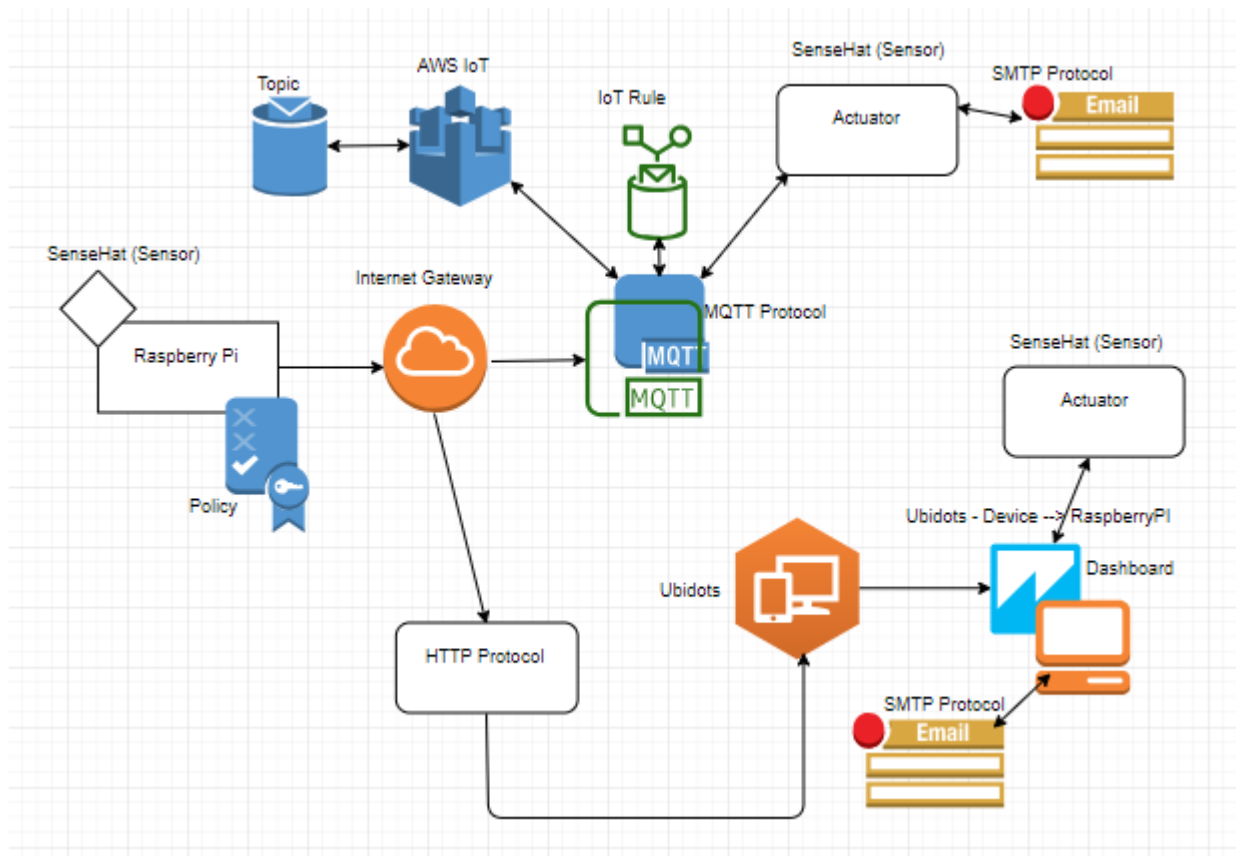
Using Raspberry Pi as the hardware and SenseHat which has Temperature, Humidity, Pressure sensors and stream this sensor related information to AWS IoT (Internet of Things) and Ubidots continuously.

The goal of this project is to build two simple applications which can help us continuously stream all the SenseHat related sensor information continuously to (AWS IoT, Ubidots) cloud services so that we can control it remotely.

System Design Block Diagram



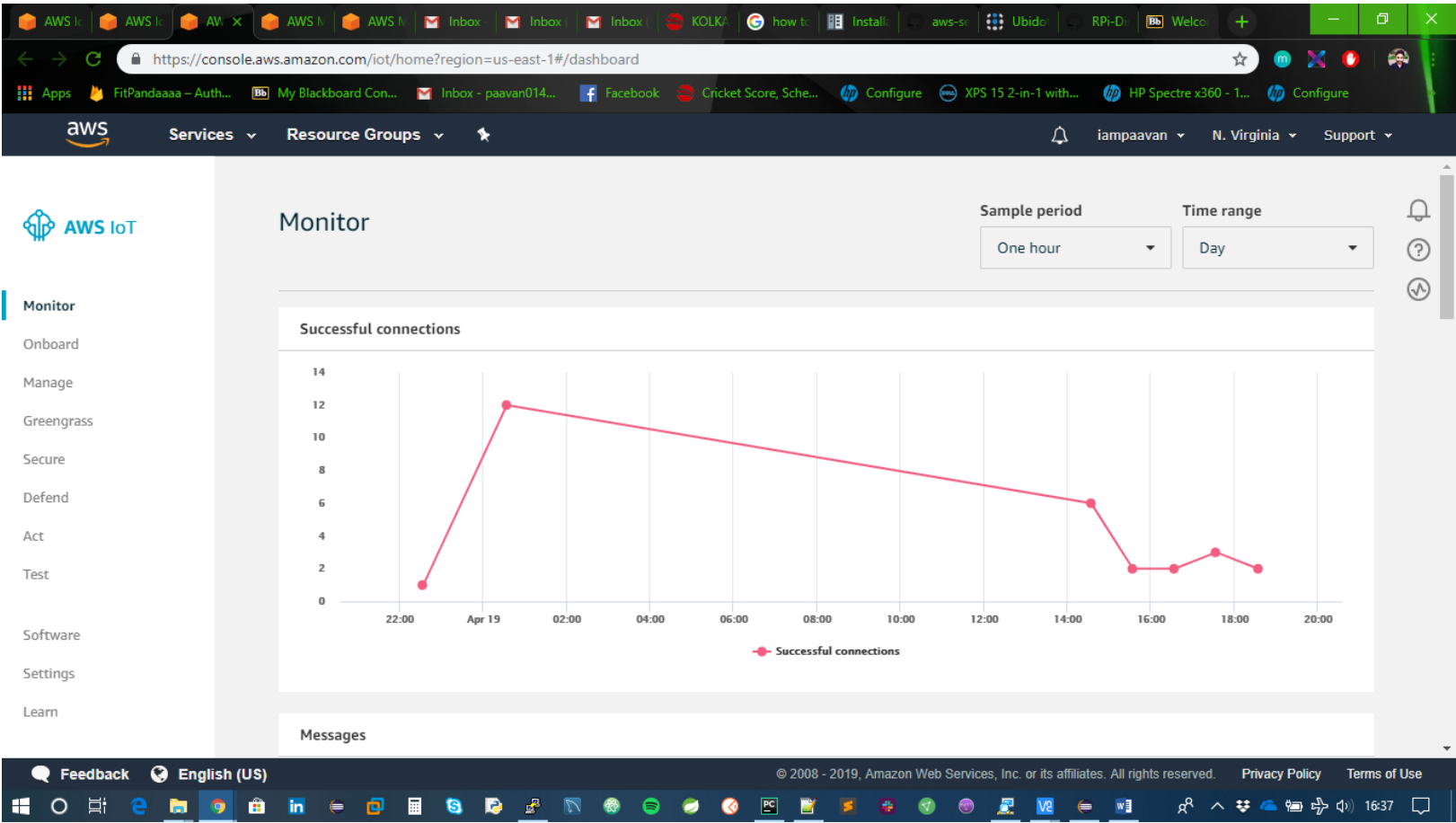
- What problem are we trying to solve?
 - ➔ Continuously read all the SenseHat related sensors information which is residing at a particular residence and send those set of data to the cloud to analyze and act accordingly remotely. If there is any unusual increase in temperature or humidity beyond the threshold limit etc., then we can get the information and set the temperature or pressure back to its normal value.
- High Level Design Diagram:



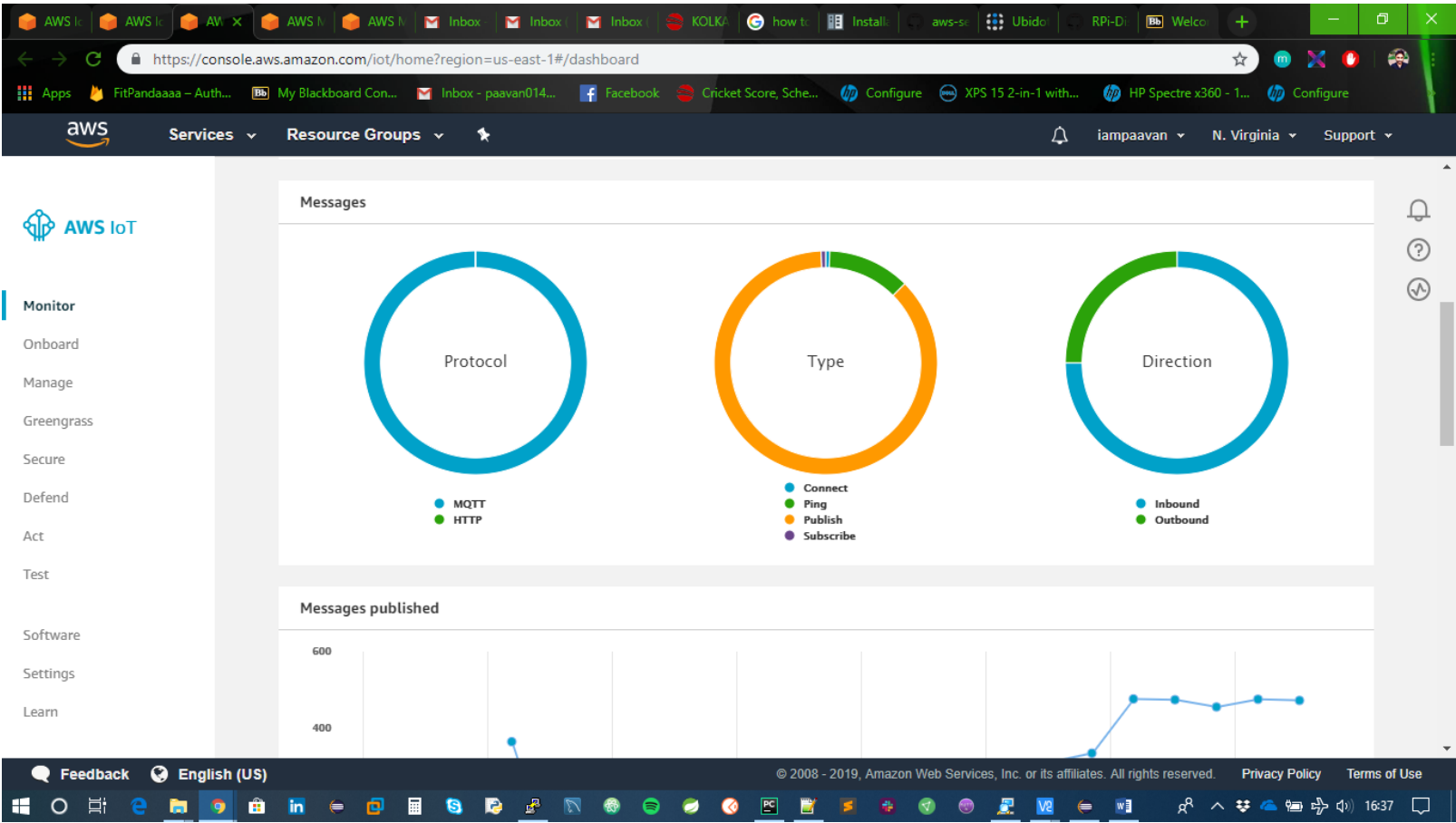
- List the cloud services and capabilities you think you'll utilize.
 - ➔ AWS IoT, Ubidots Cloud Service, MQTT, SMTP and HTTP Protocols etc.
- If your project is successful, what outcome do you expect (e.g. what will happen if everything works)?
 - ➔ We'll be able to successfully control and make necessary changes to our home automated devices if there are any unusual changes in the temperature or pressure or humidity for that matter. We can analyze the data remotely which is continuously being streamed to AWS IoT, Ubidots and Ubidots dashboard tells us all the information.

AWS IoT Dashboard:

Monitor the connections made from constrained device (Raspberry Pi)

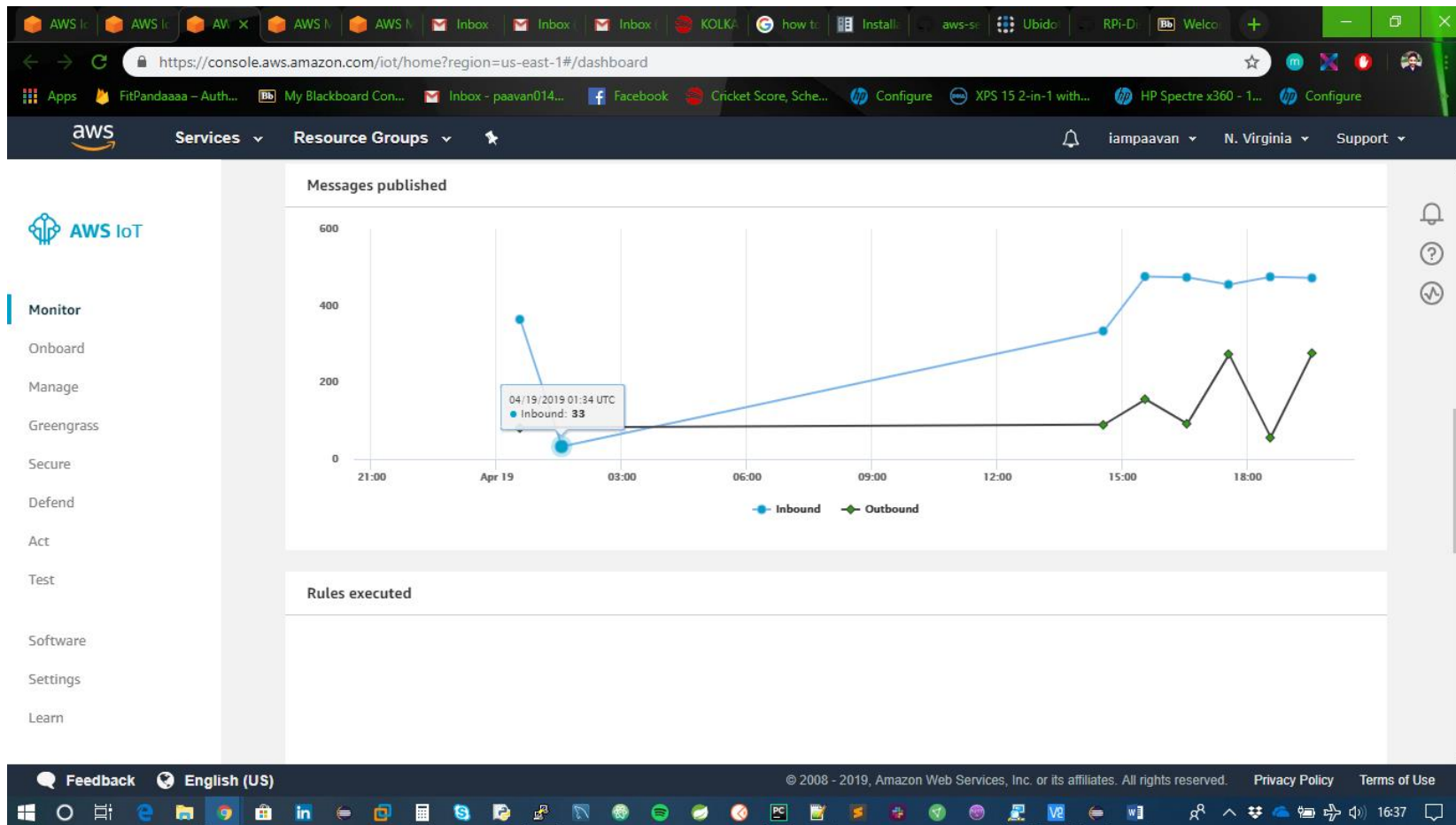


Messages Details: Protocol Used, Type of Messages and Direction of Messages



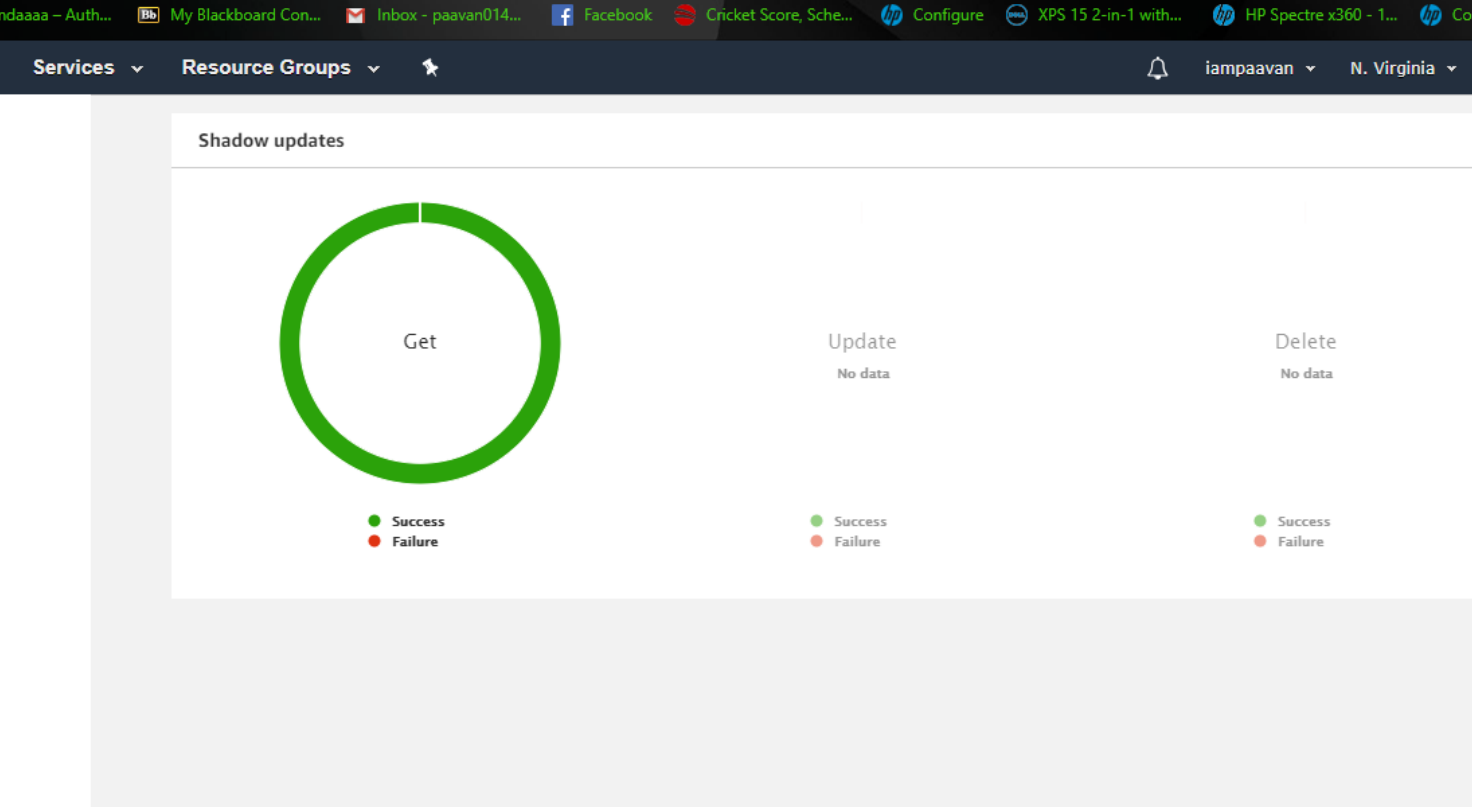
Messages Published: Sensor Data published from Constrained Device to AWS IoT

Also shows the time interval for a brief period of time. Streamed data for more than 6 to 8 hours.



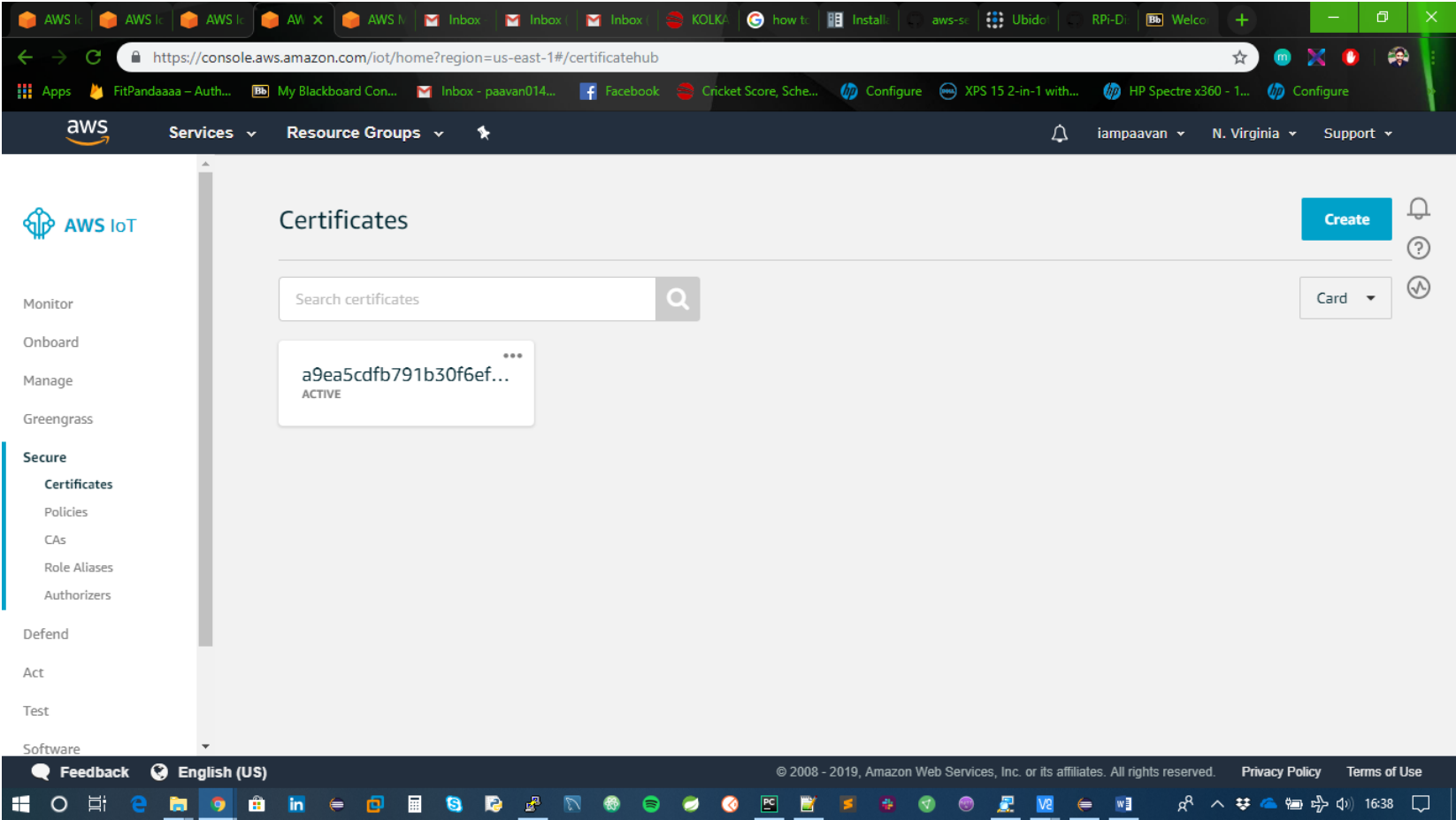
AWS IoT → Monitor → Shadow Updates

The screenshot displays the AWS IoT console interface. At the top, a navigation bar includes the AWS logo, a 'Services' dropdown, and a 'Resource Groups' dropdown. Below this, a sidebar on the left lists various IoT services: Monitor, Onboard, Manage, Greengrass, Secure, Defend, Act, Test, Software, Settings, and Learn. The main content area is titled 'Shadow updates' and features three donut charts representing the status of 'Get', 'Update', and 'Delete' operations. The 'Get' chart is a solid green circle, indicating 100% success. The 'Update' and 'Delete' charts are empty, indicating 'No data'. A legend at the bottom of each chart shows a green dot for 'Success' and a red dot for 'Failure'. The bottom of the image shows a Windows taskbar with various application icons and a system clock displaying 16:38.



Operation	Success	Failure
Get	100%	0%
Update	No data	No data
Delete	No data	No data

AWS Certificates:



A screenshot of the AWS IoT console showing the details of an IoT Certificate. The browser address bar displays the URL: `https://console.aws.amazon.com/iot/home?region=us-east-1#/certificate/a9ea5cdfb791b30f6effc5e2f06fbb324b5cc59e0776d0dcd88e8c1a09302e2`.

The certificate is in an **ACTIVE** state. The **Certificate ARN** is `arn:aws:iot:us-east-1:123456789012:certificate/a9ea5cdfb791b30f6effc5e2f06fbb324b5cc59e0776d0dcd88e8c1a09302e2`. A description states: "A certificate Amazon Resource Name (ARN) uniquely identifies this certificate. [Learn more](#)".

The **Details** section includes the following information:

- Issuer:** OU=Amazon Web Services O\=Amazon.com Inc. L\=Seattle ST\=Washington C\=US
- Subject:** CN=AWS IoT Certificate
- Create date:** Apr 1, 2019 4:22:07 PM -0400
- Effective date:** Apr 1, 2019 4:20:07 PM -0400
- Expiration date:** Dec 31, 2049 6:59:59 PM -0500

The bottom of the page shows the footer with "© 2008 - 2019, Amazon Web Services, Inc. or its affiliates. All rights reserved.", "Privacy Policy", and "Terms of Use".

AWS Policy: To allow constrained device to send data to AWS.

The screenshot displays the AWS IoT console interface. The browser address bar shows the URL: `https://console.aws.amazon.com/iot/home?region=us-east-1#/policy/thing01_policy`. The console header includes the AWS logo, navigation tabs for 'Services' and 'Resource Groups', and user information for 'iampaavan' in the 'N. Virginia' region. The left sidebar contains a navigation menu with 'Overview' (selected), 'Certificates', 'Versions', 'Groups', and 'Non-compliance'. The main content area is titled 'Policy ARN' and includes a description: 'A policy ARN uniquely identifies this policy. [Learn more](#)'. Below this, the 'Policy ARN' is displayed as `arn:aws:iot:us-east-1:838513538885:policy/thing01_policy`, which is partially obscured by a red line. The 'Policy document' section follows, with a description: 'The policy document defines the privileges of the request. [Learn more](#)'. Below the description, it states 'Version 1 updated Apr 18, 2019 12:05:10 PM -0400' and provides an 'Edit policy document' link. The policy document is shown in a code editor with the following JSON content:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "iot:*",
      "Resource": "*"
    }
  ]
}
```

The footer of the console shows 'Feedback', 'English (US)', copyright information '© 2008 - 2019, Amazon Web Services, Inc. or its affiliates. All rights reserved.', and links to 'Privacy Policy' and 'Terms of Use'. The Windows taskbar at the bottom displays various application icons and the system clock showing '16:39'.

AWS Policy details attached to the certificate.

The screenshot shows the AWS IoT console interface for managing policies. The breadcrumb navigation at the top indicates the path: Certificates > a9ea5cdfb791b30f6eff... > thing01_policy. The left sidebar contains a navigation menu with the following items: Overview, Certificates, Versions (which is currently selected), Groups, and Non-compliance. The main content area for the 'thing01_policy' page is divided into two sections. The 'Active version' section shows a single entry: 'Version 1' created on 'Apr 18, 2019 12:05:10 PM -0400'. The 'Versions' section contains a table with the following data:

Version	Date created	Usage
Version 1	Apr 18, 2019 12:05:10 PM -0400	Active

At the bottom of the console, there is a footer with 'Feedback', 'English (US)', and copyright information for 2008-2019 Amazon Web Services, Inc. The Windows taskbar at the very bottom shows various application icons and the system clock indicating 16:39.

AWS IoT → MQTT Client

awsiot.py python script will publish the sensor data through a topic called sensor/data and through AWS IoT dashboard, we can subscribe to the same topic sensor/data.

The screenshot shows the AWS IoT console interface. The left sidebar contains navigation links: Monitor, Onboard, Manage, Greengrass, Secure, Defend, Act, Test (highlighted), Software, Settings, and Learn. The main content area is titled 'Subscriptions' and includes a 'Subscribe to a topic' link. The 'Subscribe' section explains that devices publish MQTT messages on topics and that this client can subscribe to receive them. The 'Subscription topic' field is set to 'sensor/data'. The 'Max message capture' is set to 10000. The 'Quality of Service' is set to 0, with a note that this means the client will not acknowledge to the Device Gateway that messages are received. The 'MQTT payload display' is set to 'Auto-format JSON payloads (improves readability)'. The bottom of the console shows a footer with 'Feedback', 'English (US)', copyright information, and links to 'Privacy Policy' and 'Terms of Use'.

Subscriptions

[Subscribe to a topic](#)

[Publish to a topic](#)

Subscribe
Devices publish MQTT messages on topics. You can use this client to subscribe to a topic and receive these messages.

Subscription topic
sensor/data **Subscribe to topic**

Max message capture ?
10000

Quality of Service ?
☒ 0 - This client will not acknowledge to the Device Gateway that messages are received
☐ 1 - This client will acknowledge to the Device Gateway that messages are received

MQTT payload display
☒ Auto-format JSON payloads (improves readability)
☐ Display payloads as strings (more accurate)
☐ Display raw payloads (in hexadecimal)

Feedback English (US) © 2008 - 2019, Amazon Web Services, Inc. or its affiliates. All rights reserved. Privacy Policy Terms of Use

Payload received from constrained device to AWS IoT:

MQTT client

Connected as **iotconsole-1555694507798-0**

Subscriptions

Subscribe to a topic

Publish to a topic

sensor/data

Publish

Specify a topic and a message to publish with a QoS of 0.

sensor/data

Publish to topic

1 {
2 "message": "Hello from AWS IoT console"
3 }

sensor/data Apr 19, 2019 4:08:44 PM -0400

Export Hide

{ "timestamp": "2019-04-19T20:08:44Z", "temperature": 33.4899291992, "pressure": 1011.35571289, "humidity": 37.8846740723 }

MQTT client

Connected as **iotconsole-1555694507798-0**

Subscriptions

Subscribe to a topic

Publish to a topic

sensor/data

1 {
2 "message": "Hello from AWS IoT console"
3 }

sensor/data Apr 19, 2019 4:08:44 PM -0400

Export Hide

{ "timestamp": "2019-04-19T20:08:44Z", "temperature": 33.4899291992, "pressure": 1011.35571289, "humidity": 37.8846740723 }

sensor/data Apr 19, 2019 4:08:36 PM -0400

Export Hide

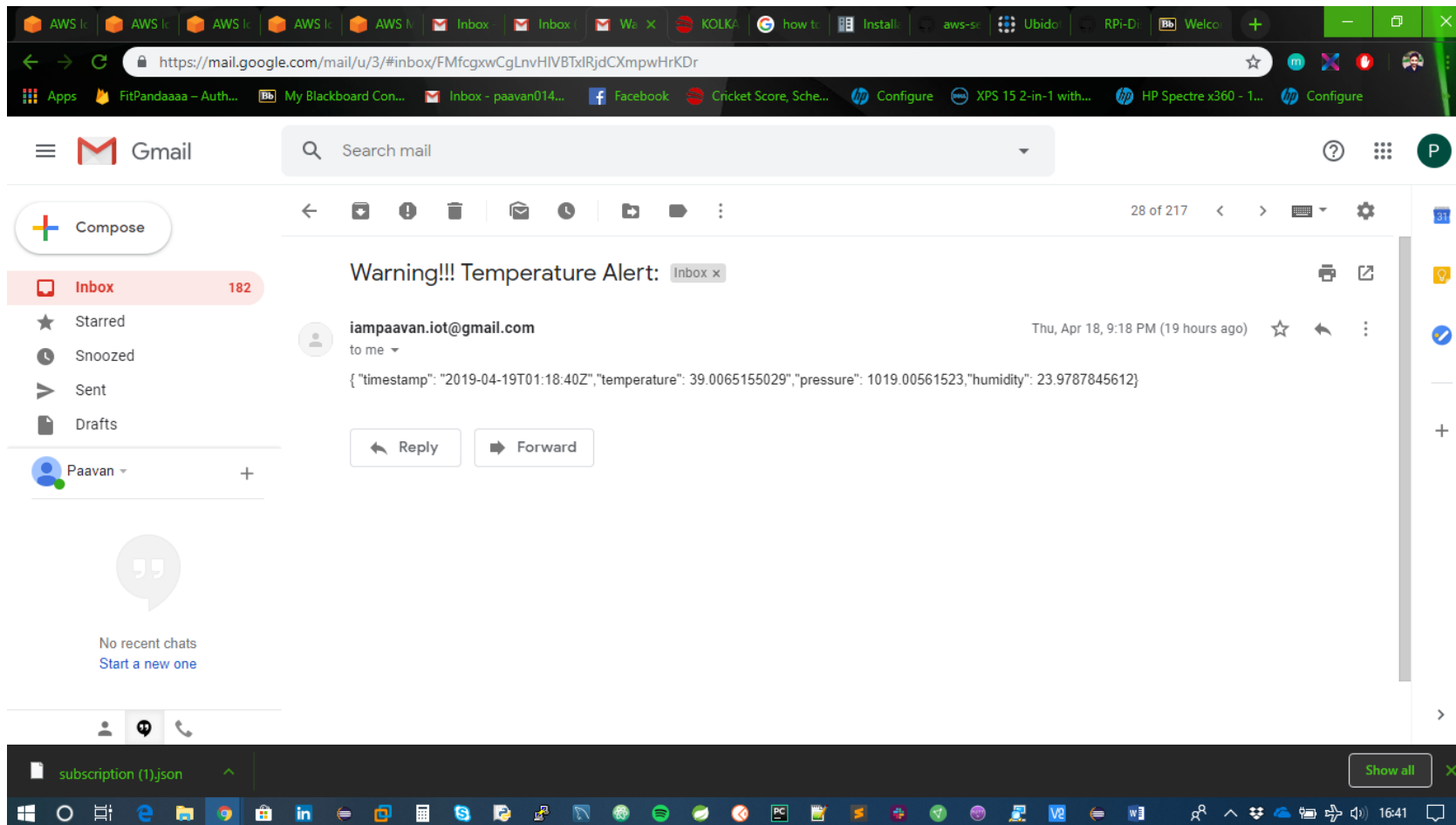
{ "timestamp": "2019-04-19T20:08:37Z", "temperature": 33.5090827942, "pressure": 1011.3125, "humidity": 38.9762496948 }

sensor/data Apr 19, 2019 4:08:29 PM -0400

Export Hide

{ "timestamp": "2019-04-19T20:08:29Z", "temperature": 33.4133110046, "pressure": 1011.29541016, "humidity": 37.9170951843 }

SMTP Protocol: Email Triggered from awsiot.py pythin script when the conditions are greater than the set threshold limit.

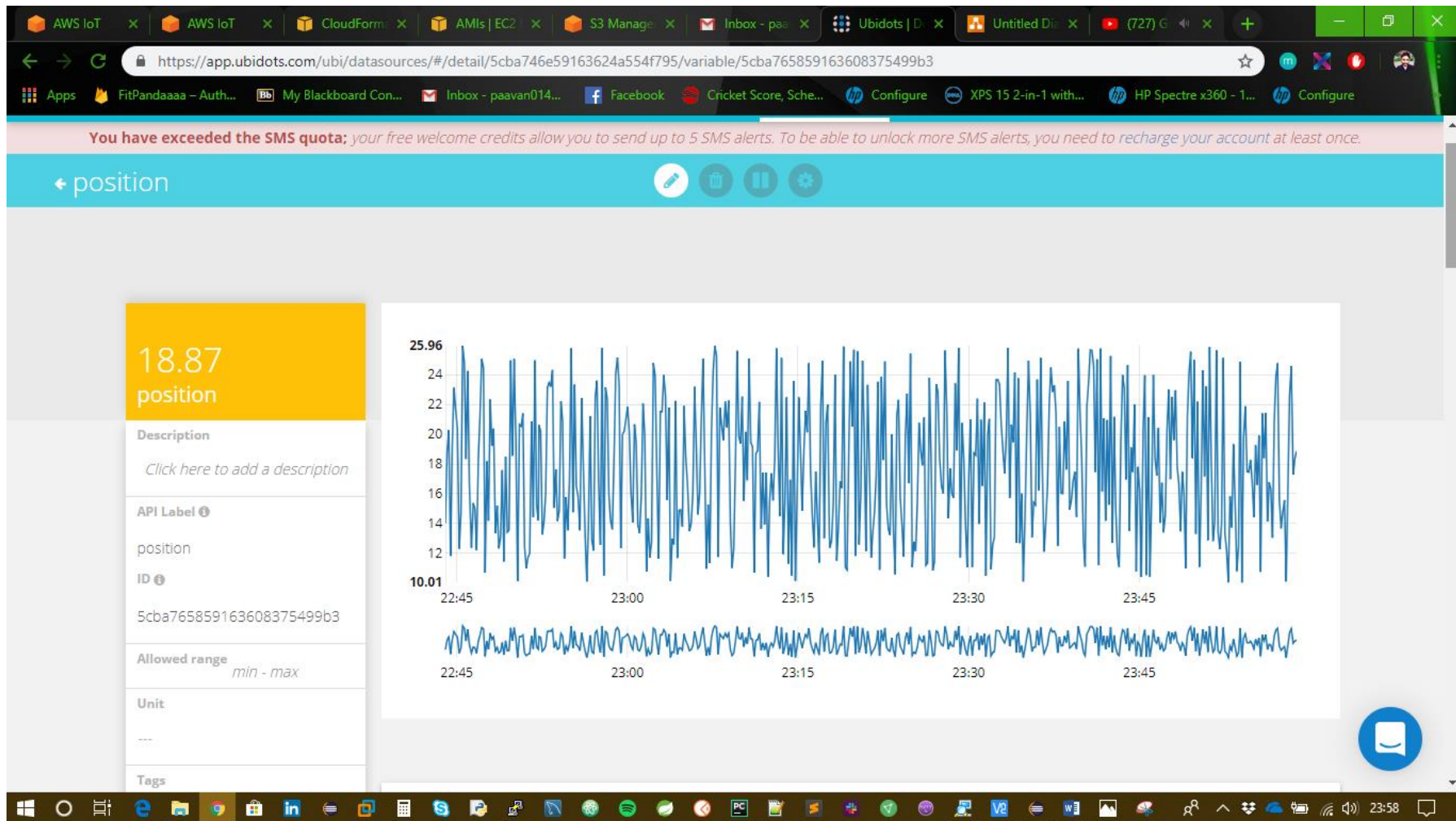


UBIDOTS: ubisots.py another python script which implements HTTP and SMTP protocol. Sends stream of sensor information to ubidots cloud

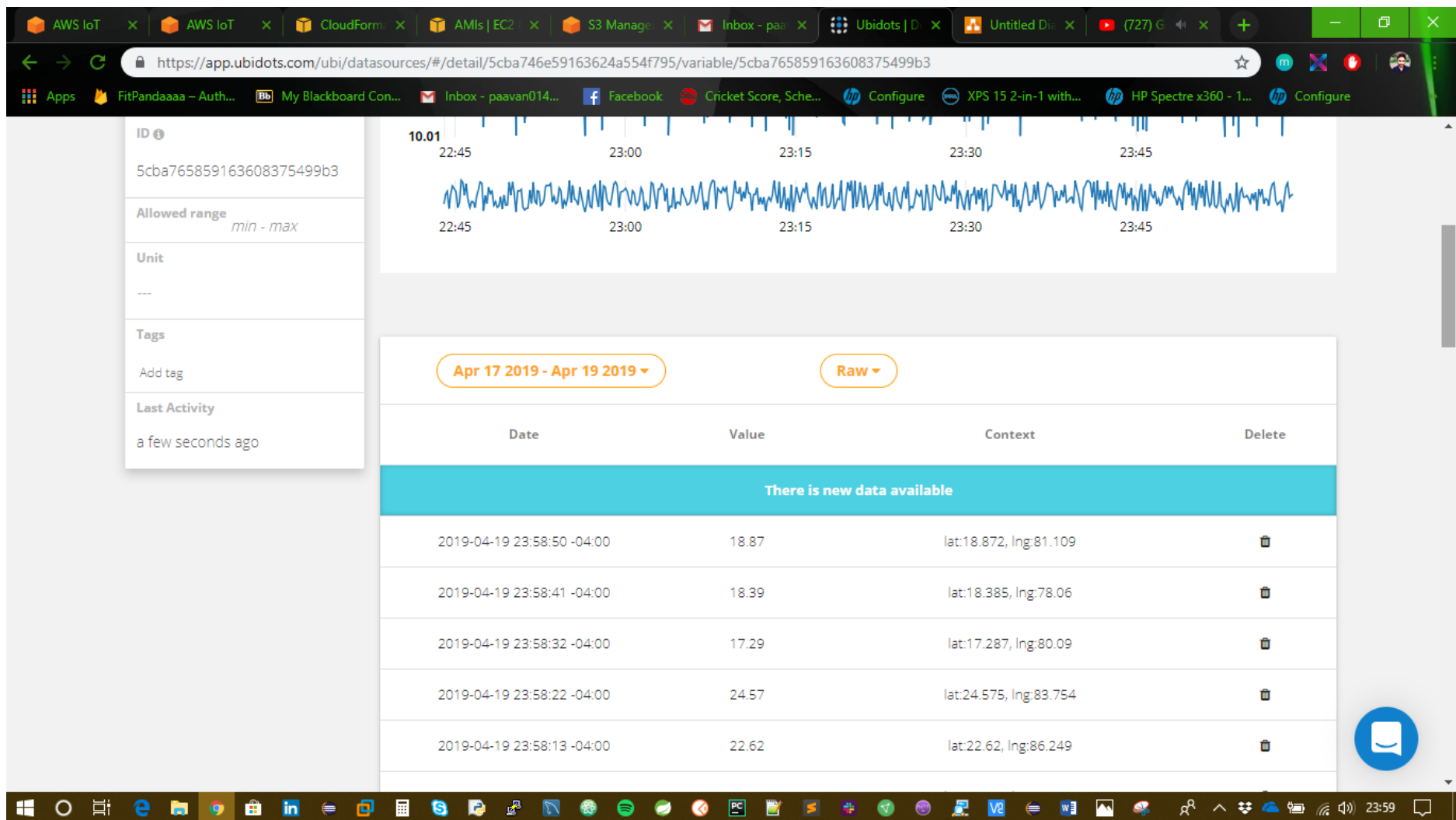
The screenshot displays the Ubidots web application interface. At the top, a browser window shows the URL <https://app.ubidots.com/ubi/datasources/#/detail/5cba746e59163624a554f795>. The main content area features a map of a region in India with a blue dot indicating the sensor's location. A sidebar on the left provides details for the 'raspberrypi' data source, including its description, API label, ID, and last activity. The main dashboard displays four live sensor readings in yellow boxes: position (24.58), humidity (47.97), temperature (30.56), and pressure (1,012.93). Each reading includes a 'Last activity' status of 'in a few seconds'. A dashed box with a plus sign and the text 'Add Variable' is visible at the bottom center. The Windows taskbar at the bottom shows various application icons and the system clock at 23:58.

Sensor	Value	Last activity
position	24.58	in a few seconds
humidity	47.97	in a few seconds
temperature	30.56	in a few seconds
pressure	1,012.93	in a few seconds

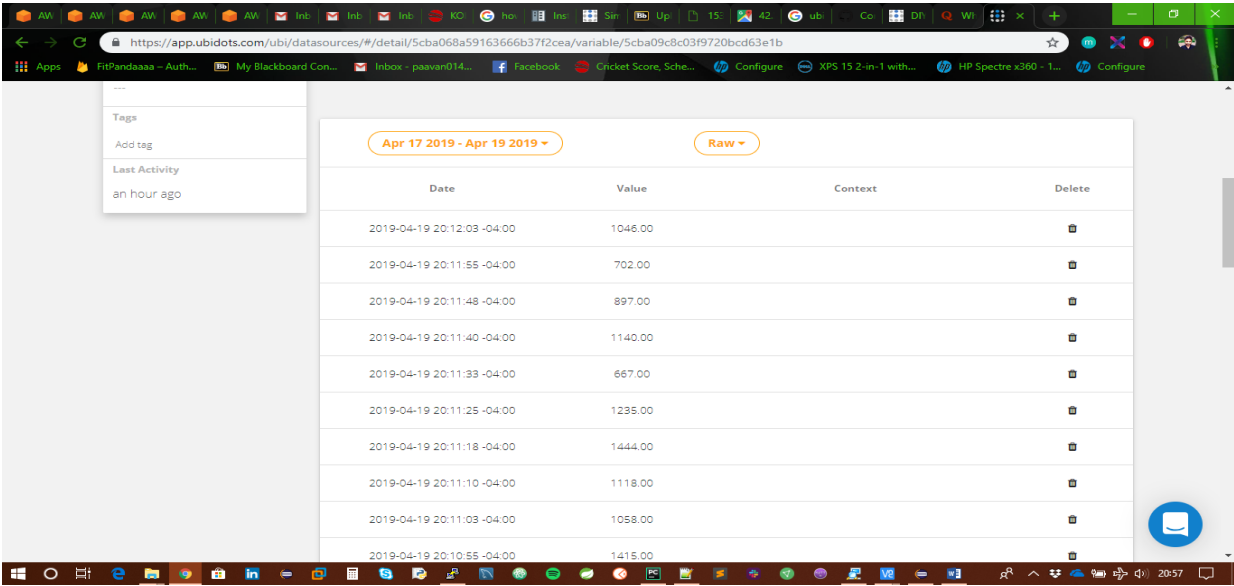
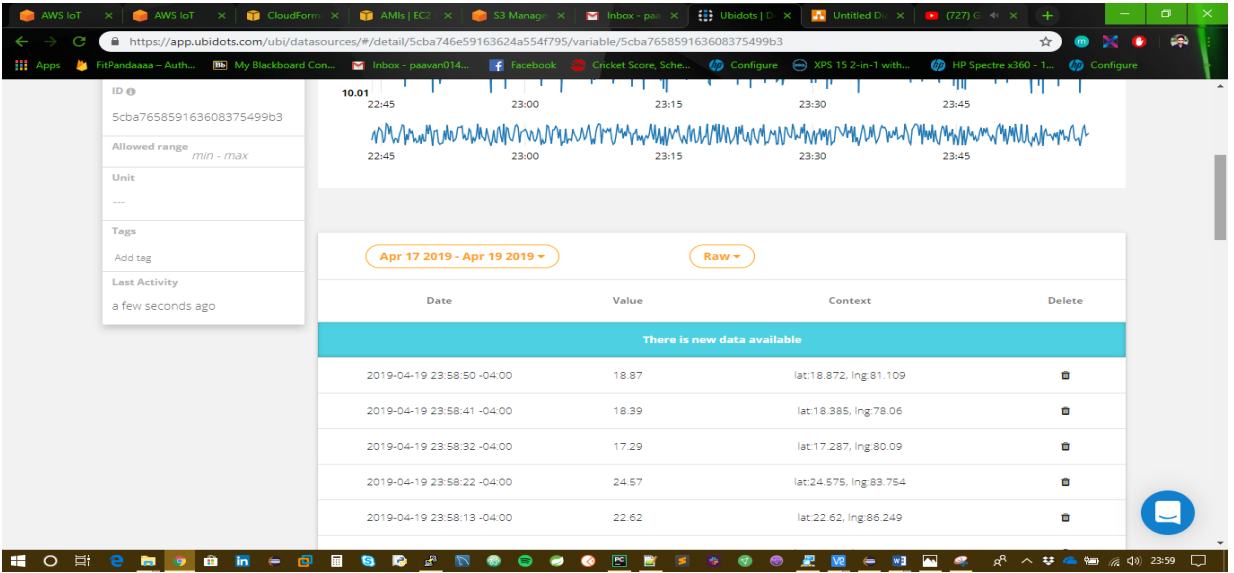
Position Details: Tells us the co-ordinates in terms of Latitude and Longitude. (In our case tells us the surroundings across India as a country)



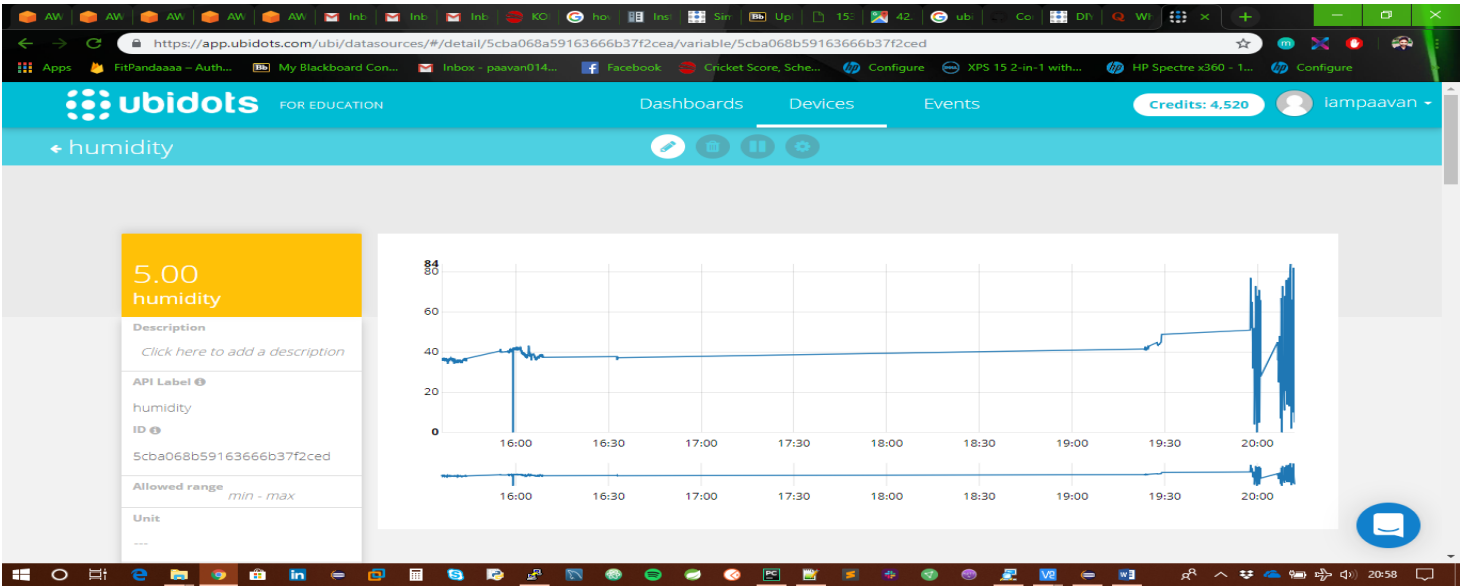
RAW Data along with the timestamps.



Pressure Details: Senses the pressure directly from sensehat of Raspberry Pi and publish the data as a payload to Ubidots.



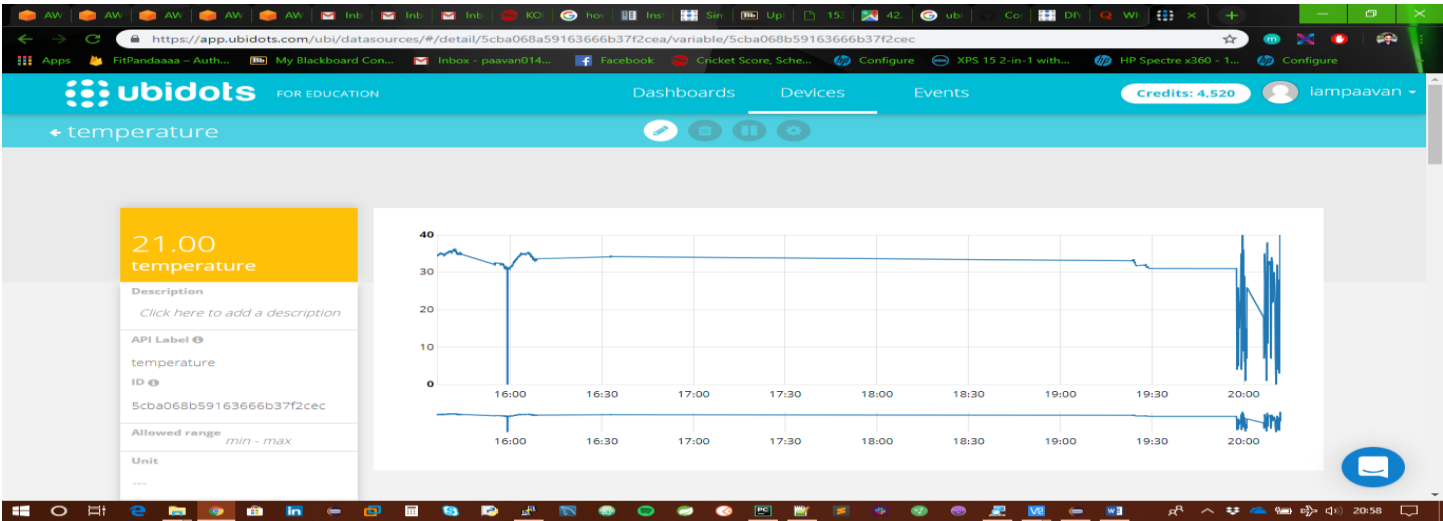
Humidity Details: Senses the pressure directly from sensehat of Raspberry Pi and publish the data as a payload to Ubidots.



This screenshot shows the 'Raw' data view for the 'humidity' variable. It displays a table of data points for the period of April 17, 2019, to April 19, 2019. The table includes columns for Date, Value, Context, and a Delete button. The data points show a sharp increase in humidity from 10.00 to 82.00 between 20:11:33 and 20:11:48 on April 19, 2019.

Date	Value	Context	Delete
2019-04-19 20:12:03 -04:00	5.00		
2019-04-19 20:11:55 -04:00	18.00		
2019-04-19 20:11:48 -04:00	33.00		
2019-04-19 20:11:40 -04:00	82.00		
2019-04-19 20:11:33 -04:00	10.00		
2019-04-19 20:11:25 -04:00	66.00		
2019-04-19 20:11:18 -04:00	66.00		
2019-04-19 20:11:10 -04:00	48.00		

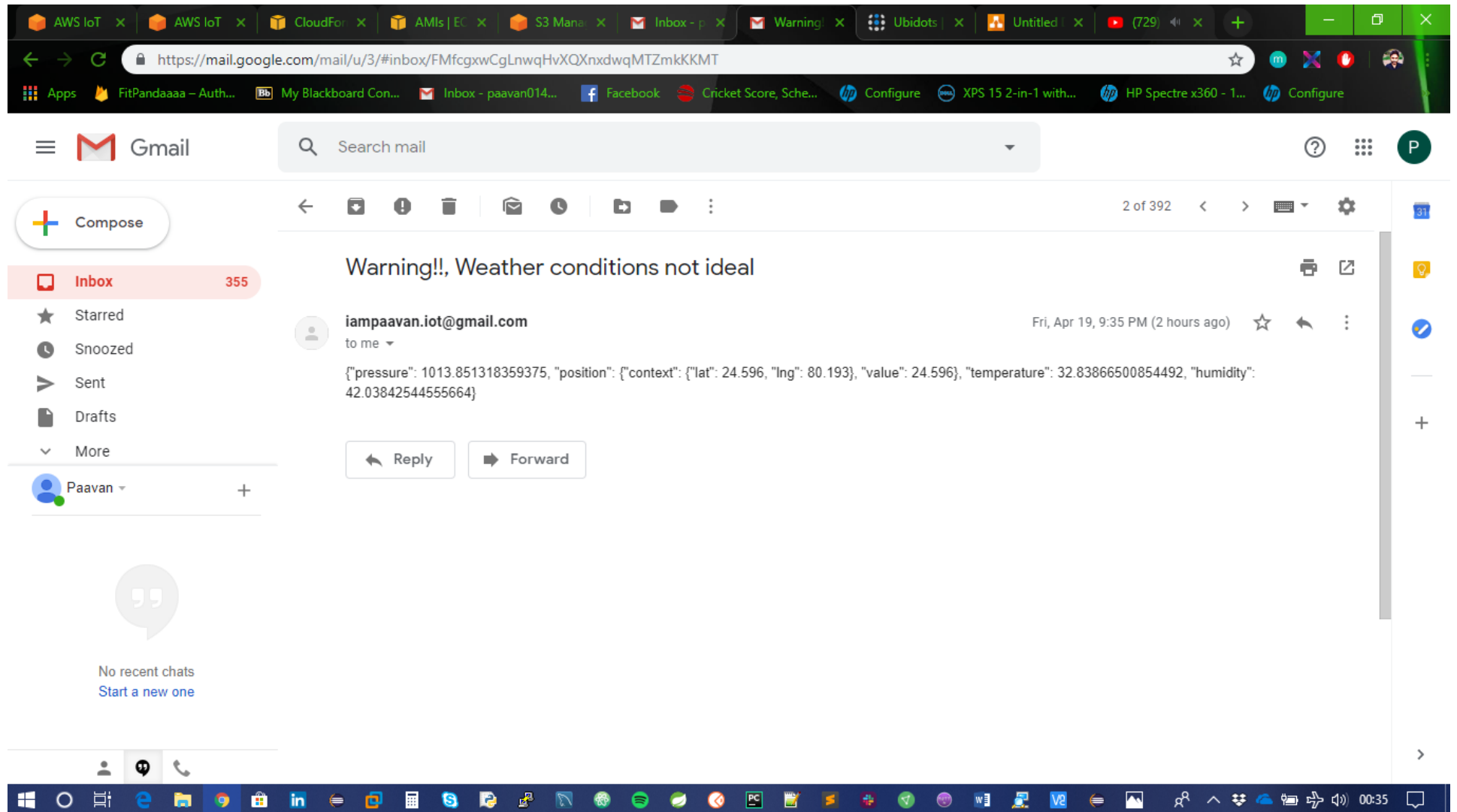
Temperature Details: Senses the pressure directly from sensehat of Raspberry Pi and publish the data as a payload to Ubidots.



The screenshot shows the Ubidots web interface for a device named 'temperature'. The dashboard includes a sidebar with the device name and a main area with a table of raw data points. The table shows temperature data from 16:00 to 20:00. The temperature starts at approximately 35, drops sharply to around 3, and then fluctuates between 3 and 40. A blue chat bubble icon is visible in the bottom right corner of the dashboard.

Date	Value	Context	Delete
2019-04-19 20:12:03 -04:00	21.00		
2019-04-19 20:11:55 -04:00	40.00		
2019-04-19 20:11:48 -04:00	18.00		
2019-04-19 20:11:40 -04:00	3.00		
2019-04-19 20:11:33 -04:00	11.00		
2019-04-19 20:11:25 -04:00	28.00		
2019-04-19 20:11:18 -04:00	25.00		
2019-04-19 20:11:10 -04:00	4.00		
2019-04-19 20:11:03 -04:00	16.00		
2019-04-19 20:10:55 -04:00	32.00		

Email Notification: When the conditions are triggered for more than the threshold limit.



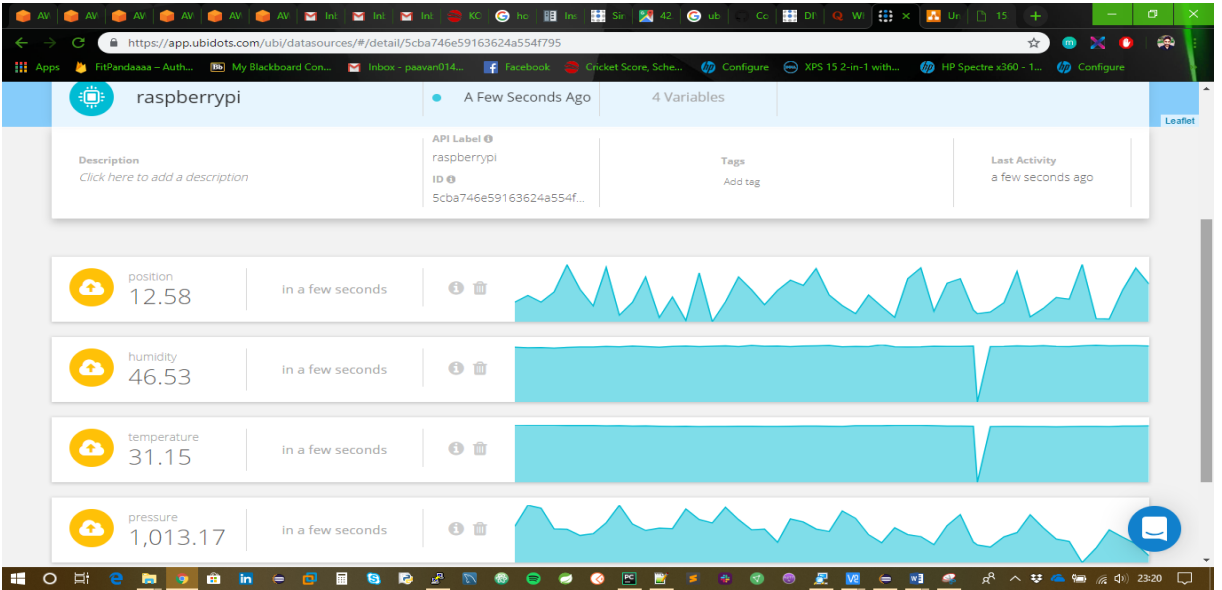
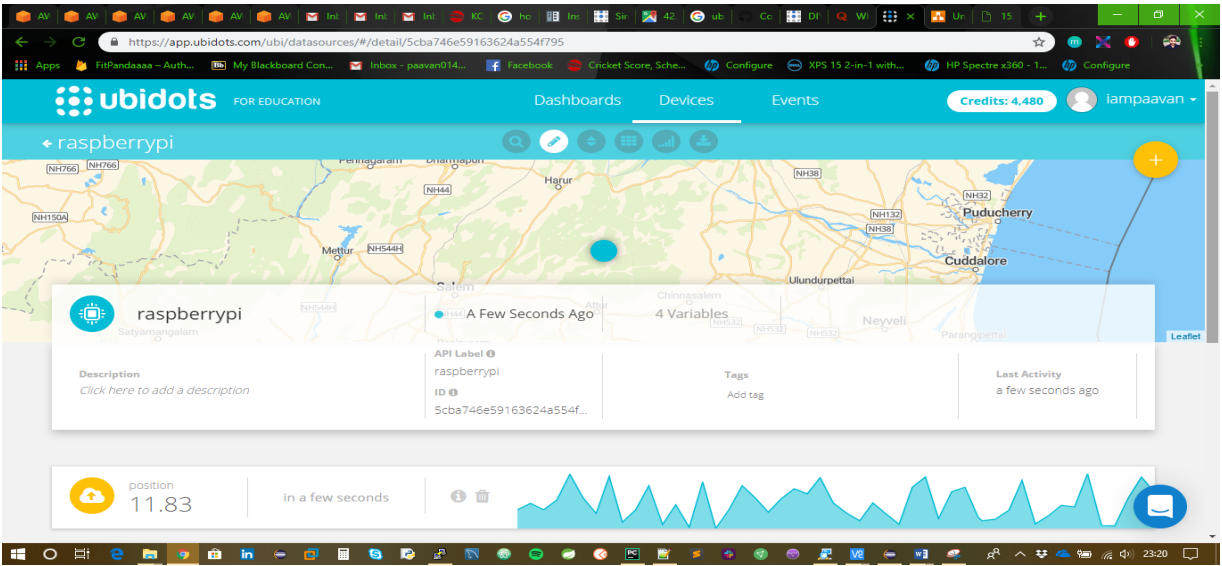
Application Console: ubidots.py

```
pi@raspberrypi: ~/workspace/iot-d...e/connected-devices-python/apps - □ ×
File Edit Tabs Help
[INFO] HTTP Request Successful. Your device is now connected and updated with se
{'pressure': 1013.304443359375, 'position': {'context': {'lat': 10.144, 'lng': 8
temperature': 32.110782623291016, 'humidity': 44.63947677612305}
[INFO] Payload sent successfully.
[INFO] Finished
Email Triggered.
[INFO] Attempting to send data
[INFO] HTTP Request Successful. Your device is now connected and updated with se
nsor details.
{'pressure': 1013.3056640625, 'position': {'context': {'lat': 12.965, 'lng': 87.
188}, 'value': 12.965}, 'temperature': 32.187400817871094, 'humidity': 45.702232
360839844}
[INFO] Payload sent successfully.
[INFO] Finished
Email Triggered.
[INFO] Attempting to send data
[INFO] HTTP Request Successful. Your device is now connected and updated with se
nsor details.
{'pressure': 1013.308349609375, 'position': {'context': {'lat': 12.596, 'lng': 8
1.112}, 'value': 12.596}, 'temperature': 31.995853424072266, 'humidity': 45.3347
7020263672}
[INFO] Payload sent successfully.
[INFO] Finished
```

Application Console: awsiot.py

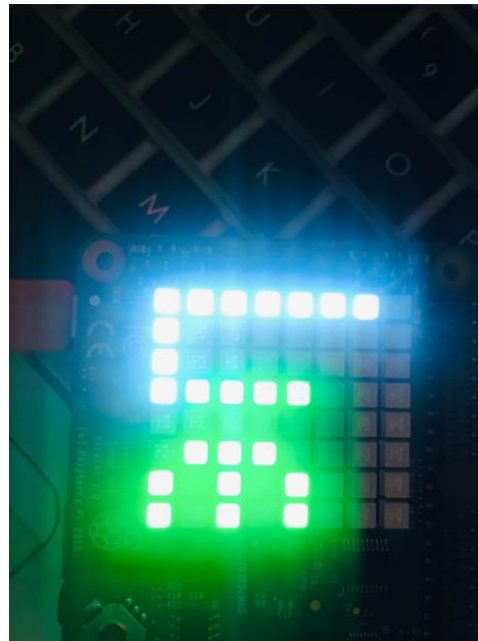
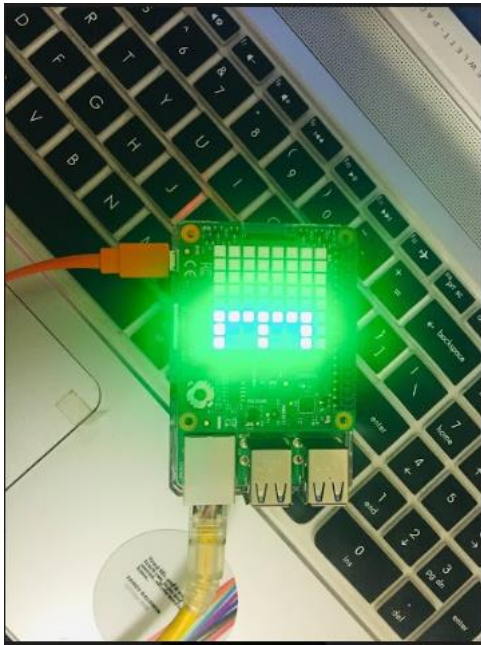
```
pi@raspberrypi: ~/workspace/iot-device/connected-devices-python/apps
File Edit Tabs Help
1013.32910156,"humidity": 45.6842193604}
No Email Triggered.
{ "timestamp": "2019-04-20T03:04:57Z","temperature": 32.1299362183,"pressure":
1013.30566406,"humidity": 45.3671951294}
No Email Triggered.
{ "timestamp": "2019-04-20T03:05:04Z","temperature": 32.2640228271,"pressure":
1013.37890625,"humidity": 45.4716682434}
No Email Triggered.
{ "timestamp": "2019-04-20T03:05:12Z","temperature": 32.1682472229,"pressure":
1013.33862305,"humidity": 45.8571395874}
No Email Triggered.
{ "timestamp": "2019-04-20T03:05:20Z","temperature": 32.3406410217,"pressure":
1013.33422852,"humidity": 45.3852081299}
No Email Triggered.
{ "timestamp": "2019-04-20T03:05:27Z","temperature": 32.2640228271,"pressure":
1013.31518555,"humidity": 45.450050354}
No Email Triggered.
{ "timestamp": "2019-04-20T03:05:35Z","temperature": 32.3023300171,"pressure": 1013.3295
8984,"humidity": 45.8427314758}
No Email Triggered.
{ "timestamp": "2019-04-20T03:05:42Z","temperature": 32.2065582275,"pressure": 1013.30517578,"humidity":
45.6193733215}
No Email Triggered.
```


UBIDOTS DASHBOARD:



Live Demo → Link → <https://drive.google.com/drive/u/0/my-drive>

SenseHat Pictures:



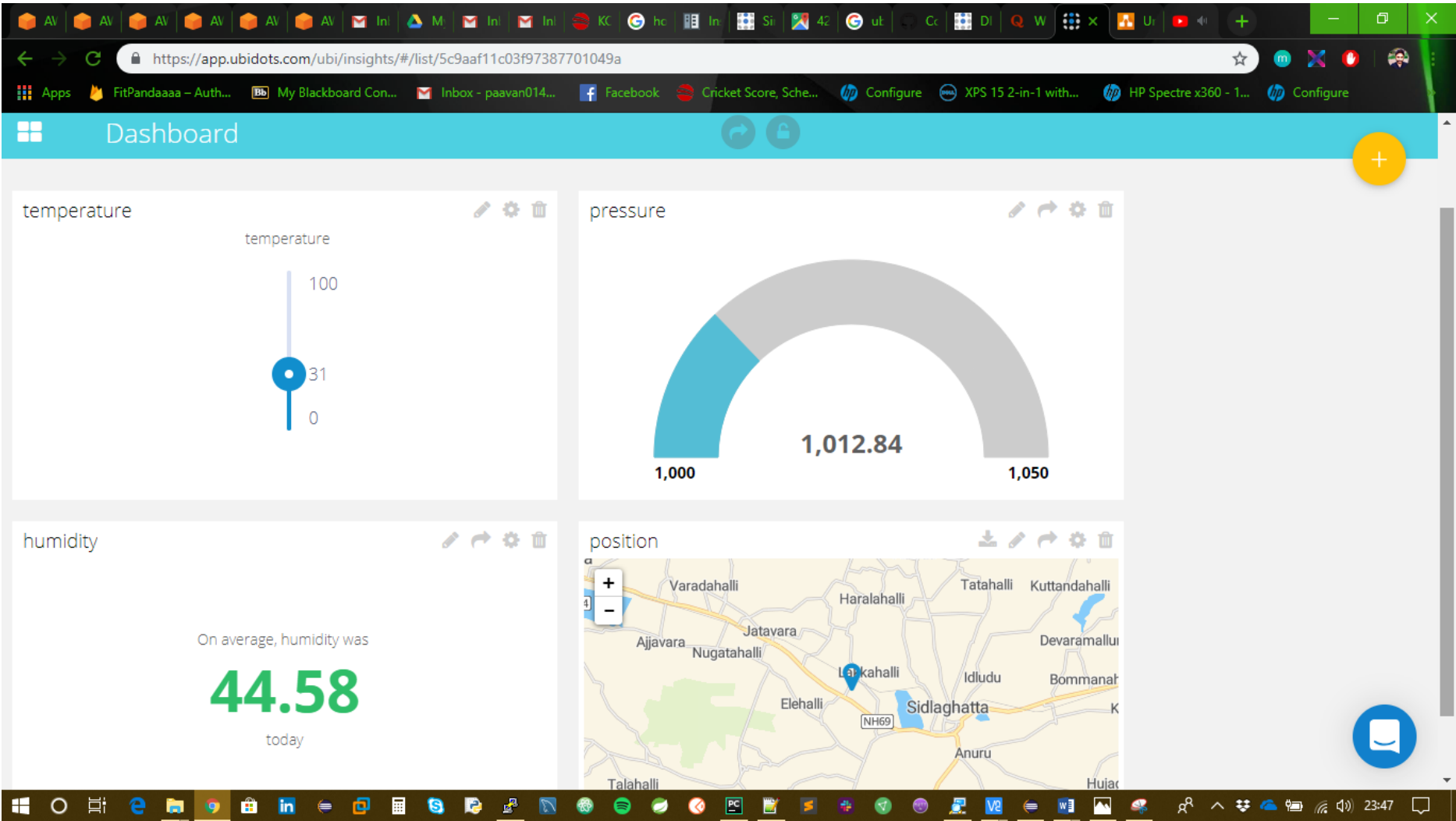
Events Created in Ubidots:

The screenshot displays the Ubidots Events page in a web browser. The URL is <https://app.ubidots.com/ubi/events/#/list>. The page shows four events in a grid:

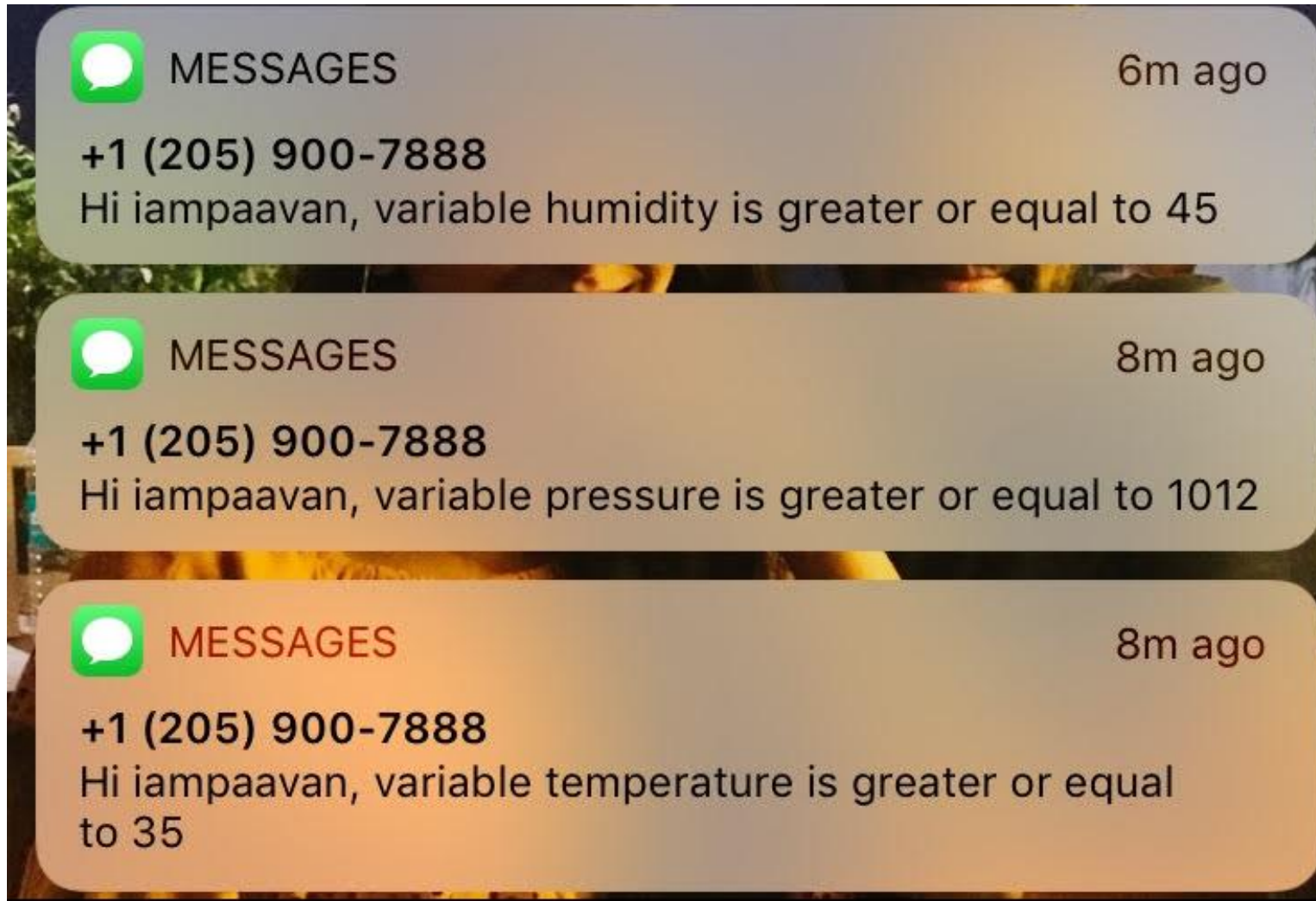
- Event 1:** Variable temperature is greater or equal to 35.0. Trigger: if temperature value is \geq than 35.0. Action: send sms to +18572308401.
- Event 2:** Variable pressure is greater or equal to 1012.0. Trigger: if pressure value is \geq than 1012.0. Action: send sms to +18572308401.
- Event 3:** Variable humidity is greater or equal to 45.0. Trigger: if humidity value is \geq than 45.0. Action: send sms to +18572308401.
- Event 4:** Variable temperature is greater or equal to 35.0. Trigger: if temperature value is \geq than 35.0. Action: Set temperature to 25.0.

Each event card includes a cloud icon with an upward arrow, a right-pointing arrow, and a smartphone icon. The bottom right corner of the page features a blue circular chat icon with a white speech bubble.

Ubidots Dashboard:



SMS Notifications:



Project Files and Documents:



values_device_rasp
berry_pi_pot4t.csv


subscription
(2).json


aws.pem


a9ea5cdfb7-public.
pem.key


a9ea5cdfb7-private.
pem.key


a9ea5cdfb7-certific
ate.pem.crt