Smart Home System

Internet of Things and Big Data Analytics – IT4021
GROUP ID: 2024-06

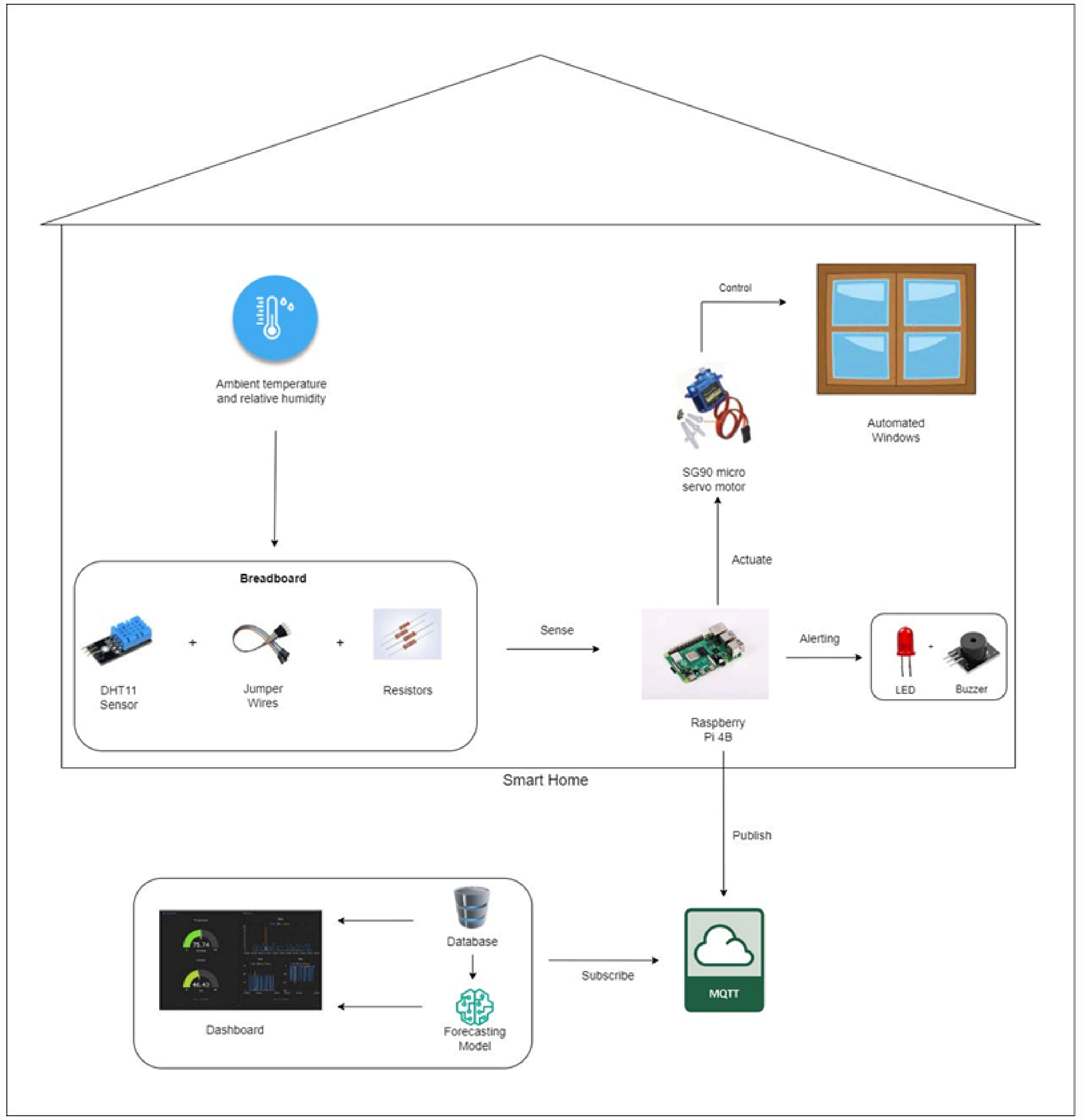
Team Members

Member Name	Student Number
Vidanage D.S.D	IT21128868
Tennekoon V.L.K	IT21015212
Ratnayake B.R.M.P	IT21066870
Maddumage P.W	IT21007538

Introduction

Our project aims to design and implement a Smart Home system that automatically manages indoor temperature based on the Heat Index (HI) by manipulating windows.

Developed a Node-RED dashboard to visualize the current Heat Index and the predicted Heat Index data up to 12 months ahead from the current date and 12 months before from the current date.



Hardware

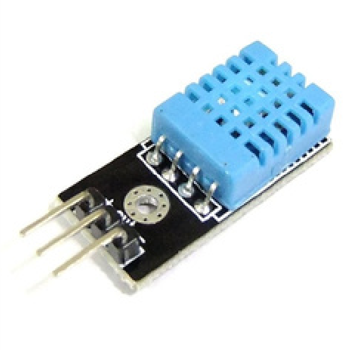
Sensor - Digital Temperature and Humidity Sensor Module (DHT11)

Actuator - Servo Motor SG90 (Actuator for Window Control)

Microcomputer - Raspberry Pi 4B Model

Micro SD Card

Alert related Components:
LED Bulb - A visual indicator to alert occupants of extreme Heat Index levels.

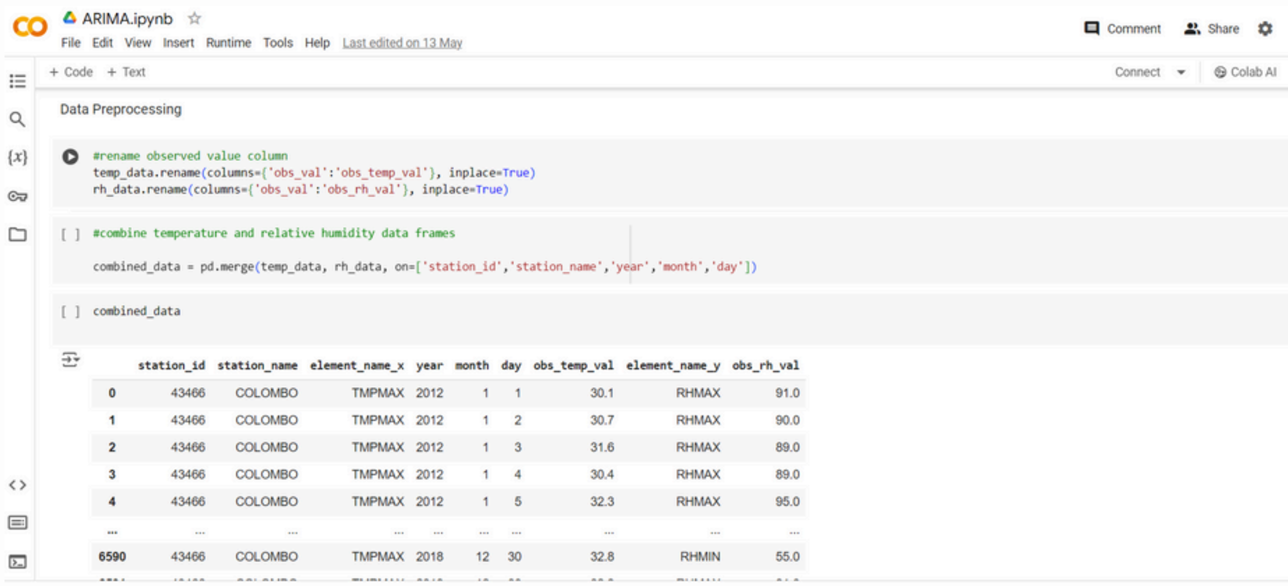


6. 3.3 to 5V Active Buzzer Alarm Module Sensor - To provide an audible alert in case of emergency situations, such as fire or gas leaks.

Model Creation and Prediction

•Preprocessed the dataset by cleaning, transforming and checking for stationarity.

•Predicted the past 12 months and next 12 months' readings using ARIMA model.


ARIMA.ipynb
File Edit View Insert Runtime Tools Help Last edited on 13 May
Comment Share
+ Code + Text
Connect
Colab AI

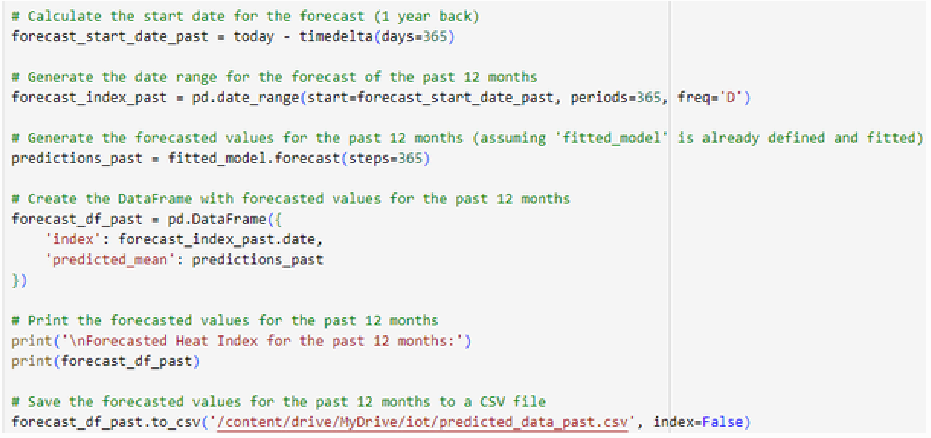
Data Preprocessing

```
#rename observed value column
temp_data.rename(columns={'obs_val':'obs_temp_val'}, inplace=True)
rh_data.rename(columns={'obs_val':'obs_rh_val'}, inplace=True)
```

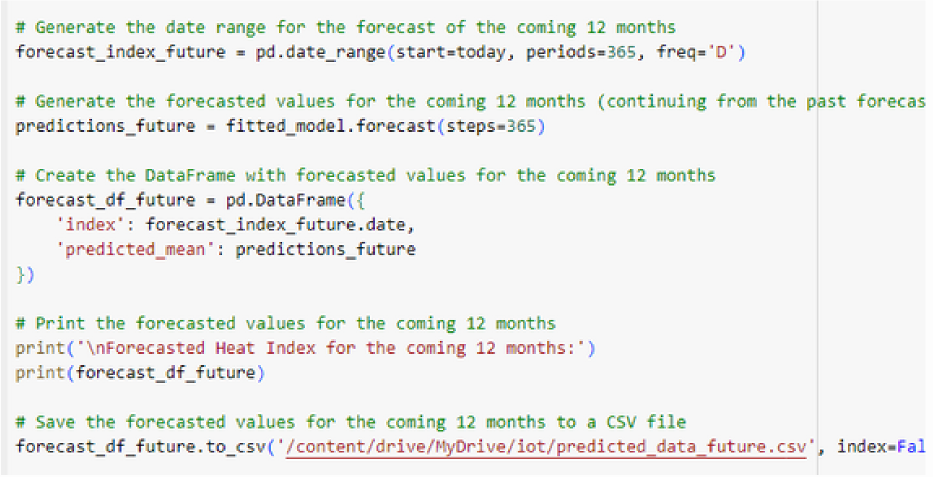
```
#combine temperature and relative humidity data frames

combined_data = pd.merge(temp_data, rh_data, on=['station_id','station_name','year','month','day'])
```

```
combined_data
```

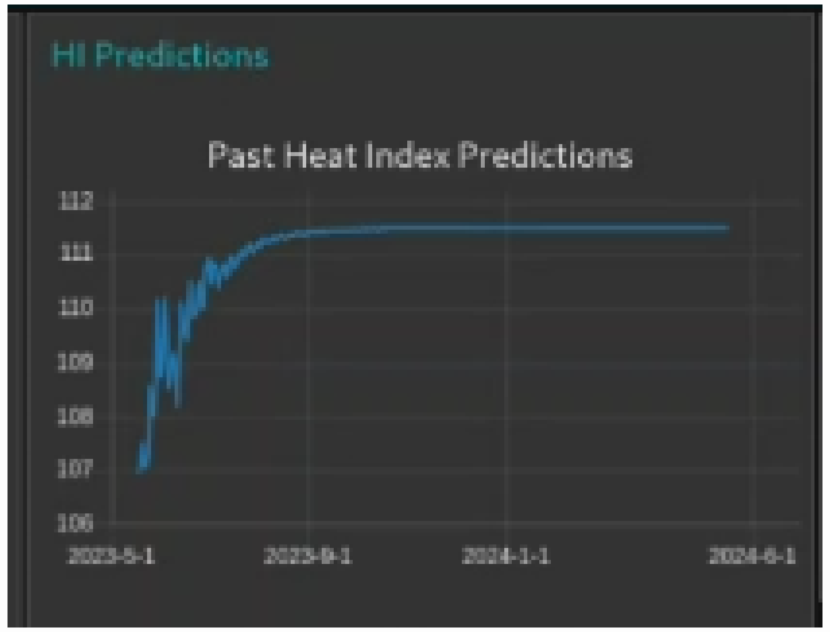
	station_id	station_name	element_name_x	year	month	day	obs_temp_val	element_name_y	obs_rh_val
0	43466	COLOMBO	TMPMAX	2012	1	1	30.1	RHMAX	91.0
1	43466	COLOMBO	TMPMAX	2012	1	2	30.7	RHMAX	90.0
2	43466	COLOMBO	TMPMAX	2012	1	3	31.6	RHMAX	89.0
3	43466	COLOMBO	TMPMAX	2012	1	4	30.4	RHMAX	89.0
4	43466	COLOMBO	TMPMAX	2012	1	5	32.3	RHMAX	95.0
...	...	...	...	...	...	...	...	...	...
6590	43466	COLOMBO	TMPMAX	2018	12	30	32.8	RHMIN	55.0

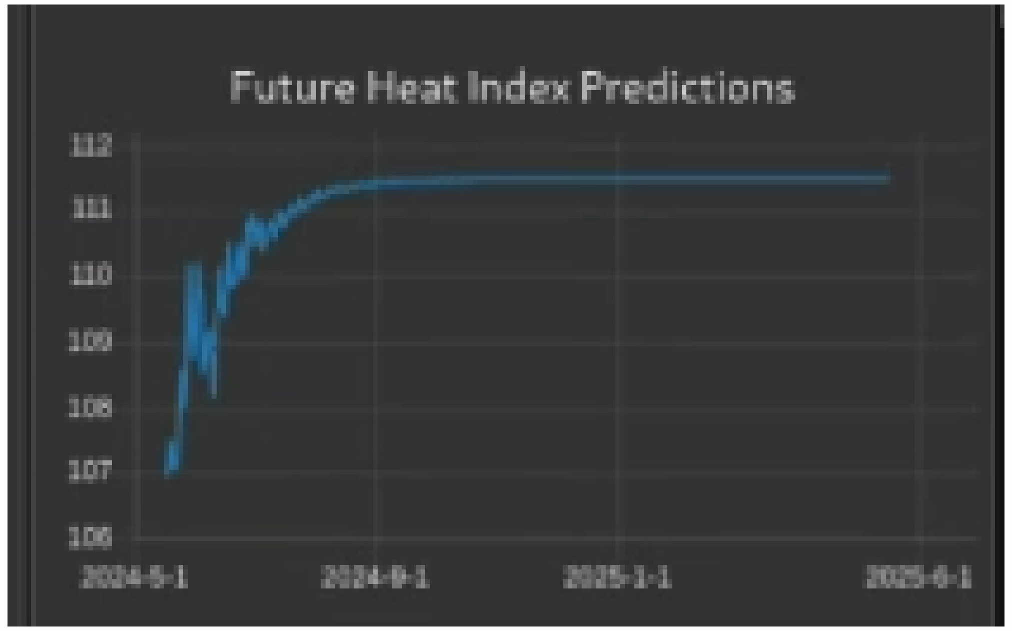


```
# Calculate the start date for the forecast (1 year back)
forecast_start_date_past = today - timedelta(days=365)

# Generate the date range for the forecast of the past 12 months
forecast_index_past = pd.date_range(start=forecast_start_date_past, periods=365, freq='D')

# Generate the forecasted values for the past 12 months (assuming 'fitted_model' is already defined and fitted)
predictions_past = fitted_model.forecast(steps=365)

# Create the DataFrame with forecasted values for the past 12 months
forecast_df_past = pd.DataFrame({
    'index': forecast_index_past.date,
    'predicted_mean': predictions_past
})

# Print the forecasted values for the past 12 months
print('\nForecasted Heat Index for the past 12 months:')
print(forecast_df_past)

# Save the forecasted values for the past 12 months to a CSV file
forecast_df_past.to_csv('/content/drive/MyDrive/iot/predicted_data_past.csv', index=False)
```



```
# Generate the date range for the forecast of the coming 12 months
forecast_index_future = pd.date_range(start=today, periods=365, freq='D')

# Generate the forecasted values for the coming 12 months (continuing from the past forecas
predictions_future = fitted_model.forecast(steps=365)

# Create the DataFrame with forecasted values for the coming 12 months
forecast_df_future = pd.DataFrame({
    'index': forecast_index_future.date,
    'predicted_mean': predictions_future
})

# Print the forecasted values for the coming 12 months
print('\nForecasted Heat Index for the coming 12 months:')
print(forecast_df_future)

# Save the forecasted values for the coming 12 months to a CSV file
forecast_df_future.to_csv('/content/drive/MyDrive/iot/predicted_data_future.csv', index=Fal
```





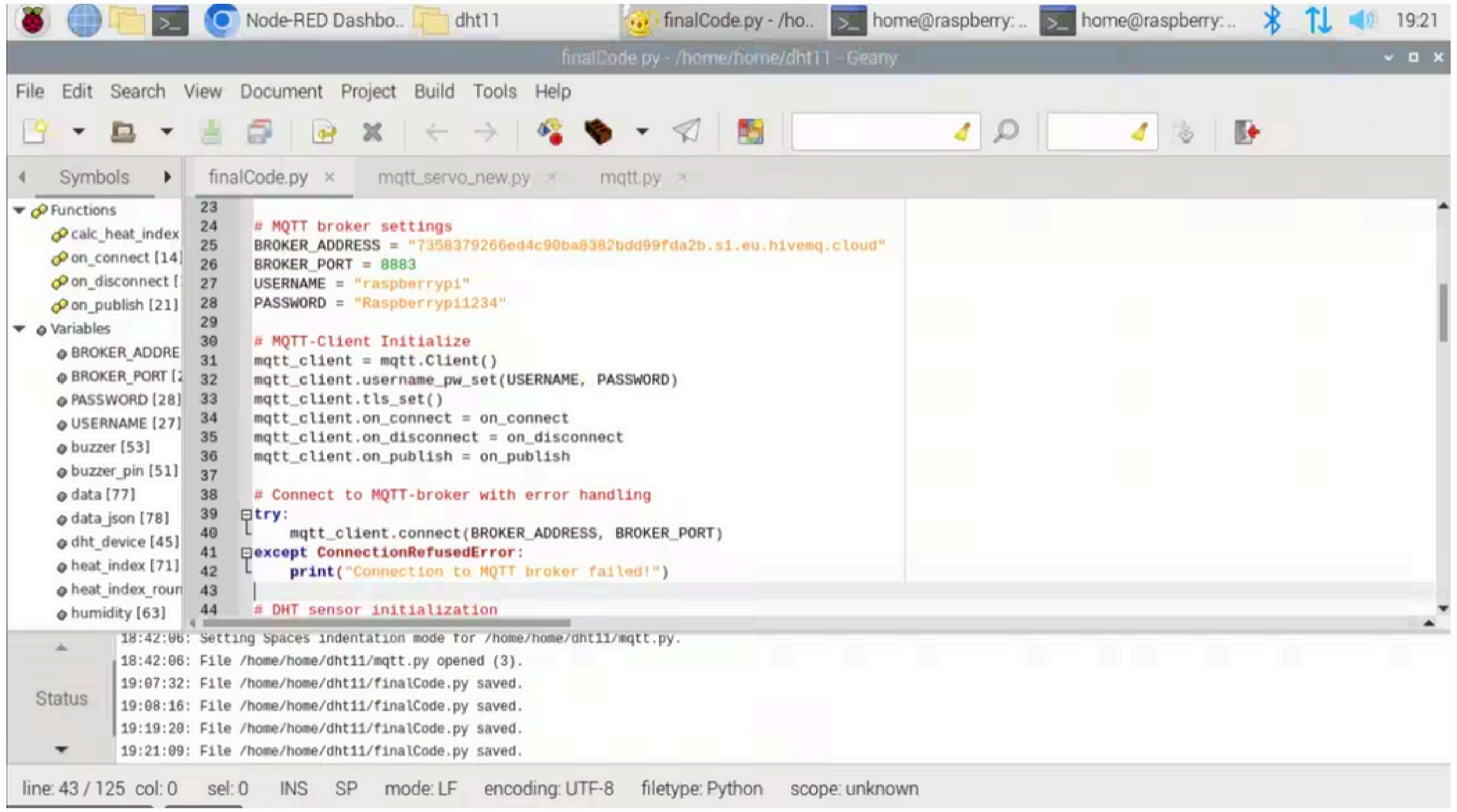
MQTT Implementation

Sensor readings are sent to the Hive MQ cloud via MQTT (Message Queuing Telemetry Transport).

Node-RED used the MQTT protocol to interface with the Hive MQ cloud, retrieving data and displaying it on a dashboard.

A simple publish-subscribe network protocol that works well for Internet of Things applications is MQTT.

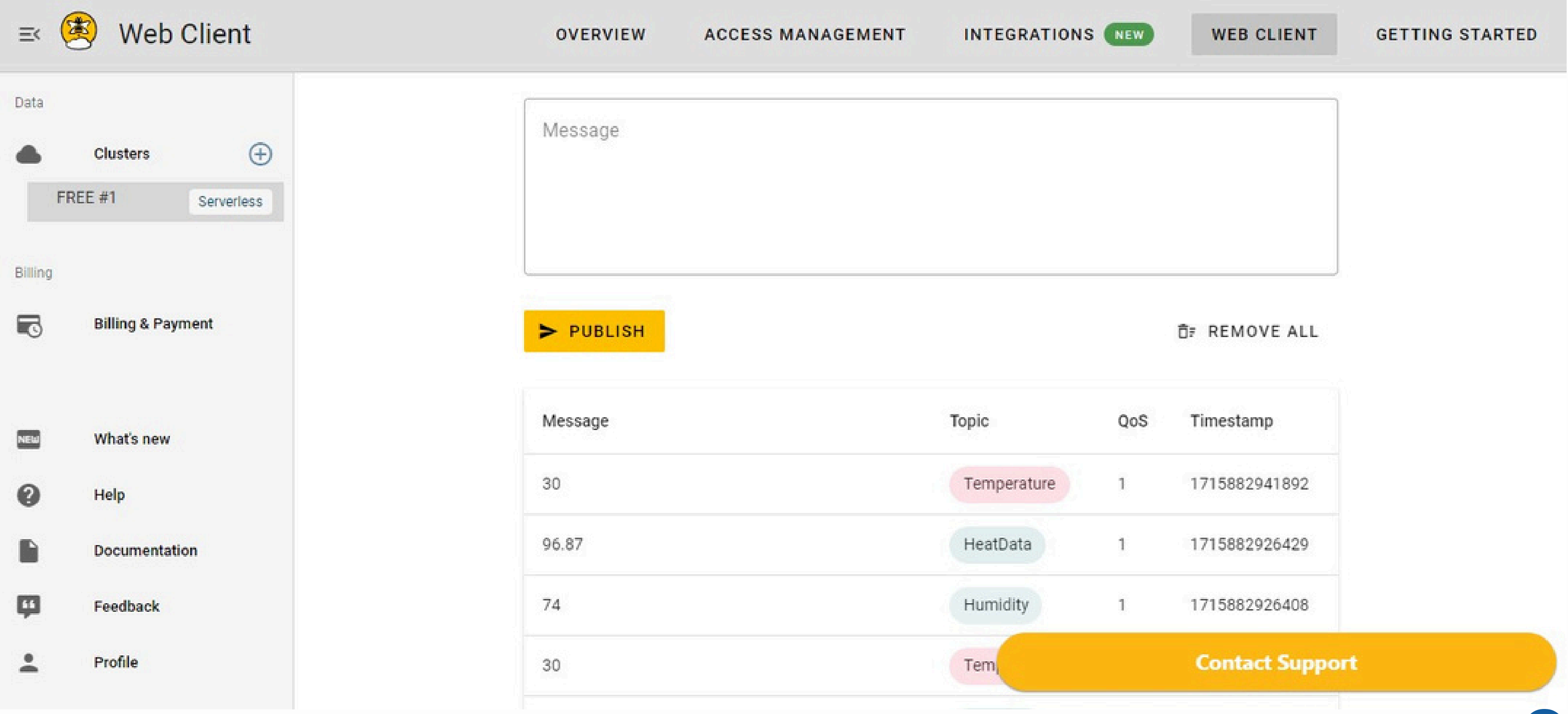
HiveMQ was used as the cloud broker to implement the MQTT protocol.



```
# MQTT broker settings
BROKER_ADDRESS = "7358379266ed4c90ba8382bdd99fda2b.s1.eu.hivemq.cloud"
BROKER_PORT = 8883
USERNAME = "raspberrypi"
PASSWORD = "Raspberrypi1234"

# MQTT-Client Initialize
mqtt_client = mqtt.Client()
mqtt_client.username_pw_set(USERNAME, PASSWORD)
mqtt_client.tls_set()
mqtt_client.on_connect = on_connect
mqtt_client.on_disconnect = on_disconnect
mqtt_client.on_publish = on_publish

# Connect to MQTT-broker with error handling
try:
    mqtt_client.connect(BROKER_ADDRESS, BROKER_PORT)
except ConnectionRefusedError:
    print("Connection to MQTT broker failed!")

# DHT sensor initialization
```

```
18:42:06: Setting Spaces indentation mode for /home/home/dht11/mqtt.py.
18:42:06: File /home/home/dht11/mqtt.py opened (3).
19:07:32: File /home/home/dht11/finalCode.py saved.
19:08:16: File /home/home/dht11/finalCode.py saved.
19:19:20: File /home/home/dht11/finalCode.py saved.
19:21:09: File /home/home/dht11/finalCode.py saved.
```

line: 43 / 125 col: 0 sel: 0 INS SP mode: LF encoding: UTF-8 filetype: Python scope: unknown



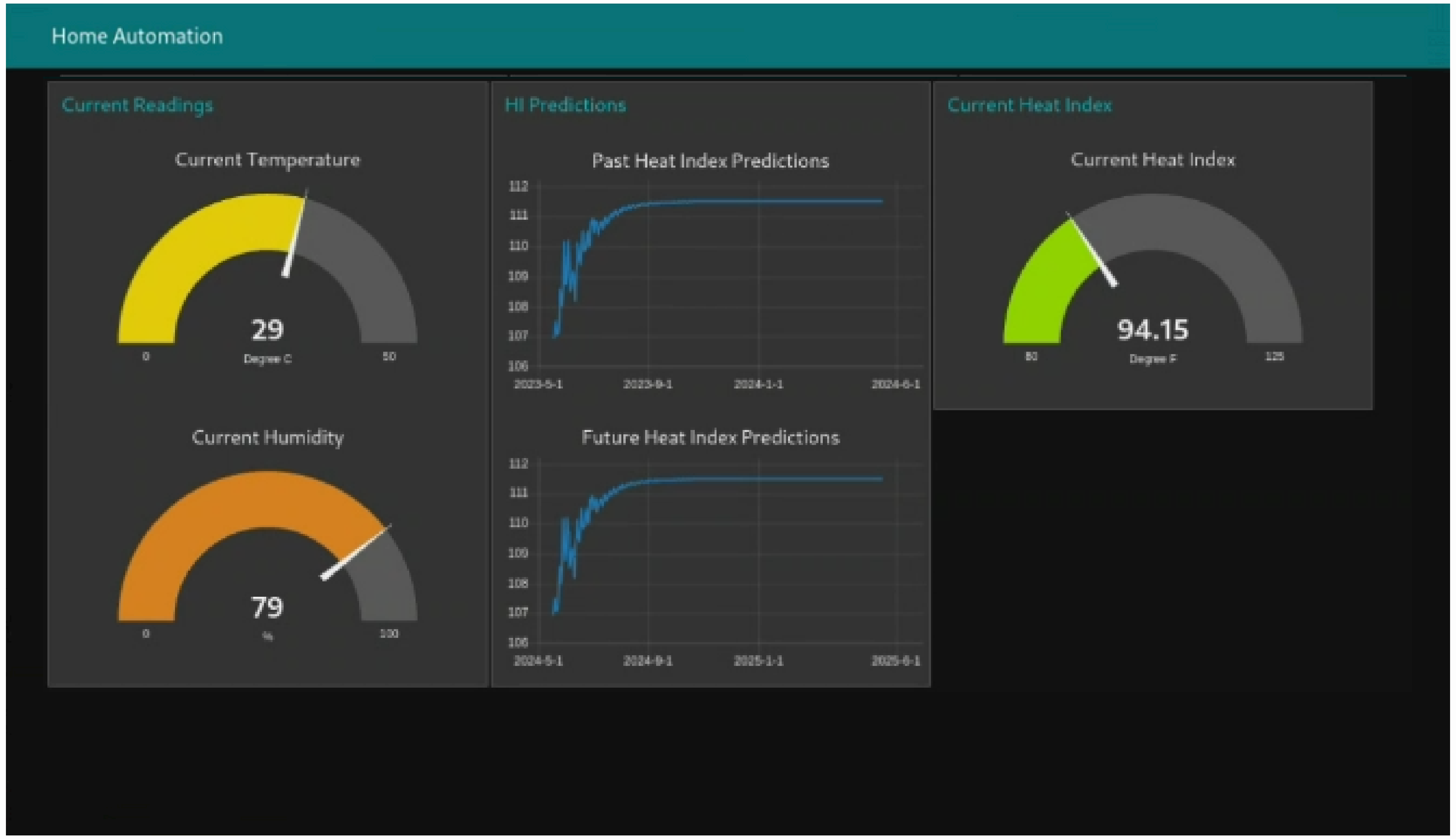
Message	Topic	QoS	Timestamp
30	Temperature	1	1715882941892
96.87	HeatData	1	1715882926429
74	Humidity	1	1715882926408
30	Tem		

Dashboard Implementation

For the purpose of showing the current, future, and previous prediction values on the dashboard, three flows were constructed in Node-RED.

The Node-red Dashboard then showed the current temperature, relative humidity, and heat index on a gauge.

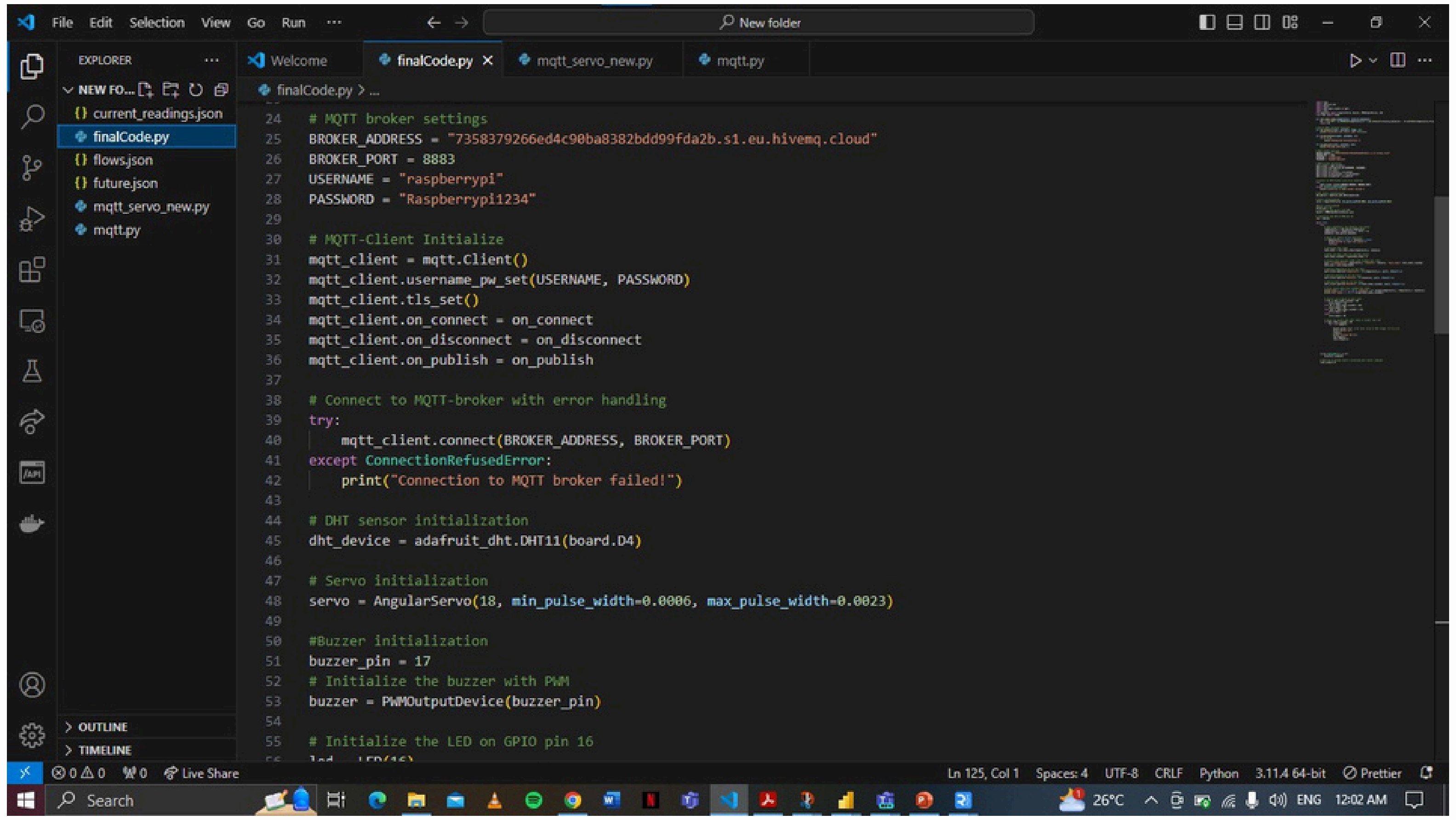
A linear graph is used to show the projections for the past and the future.



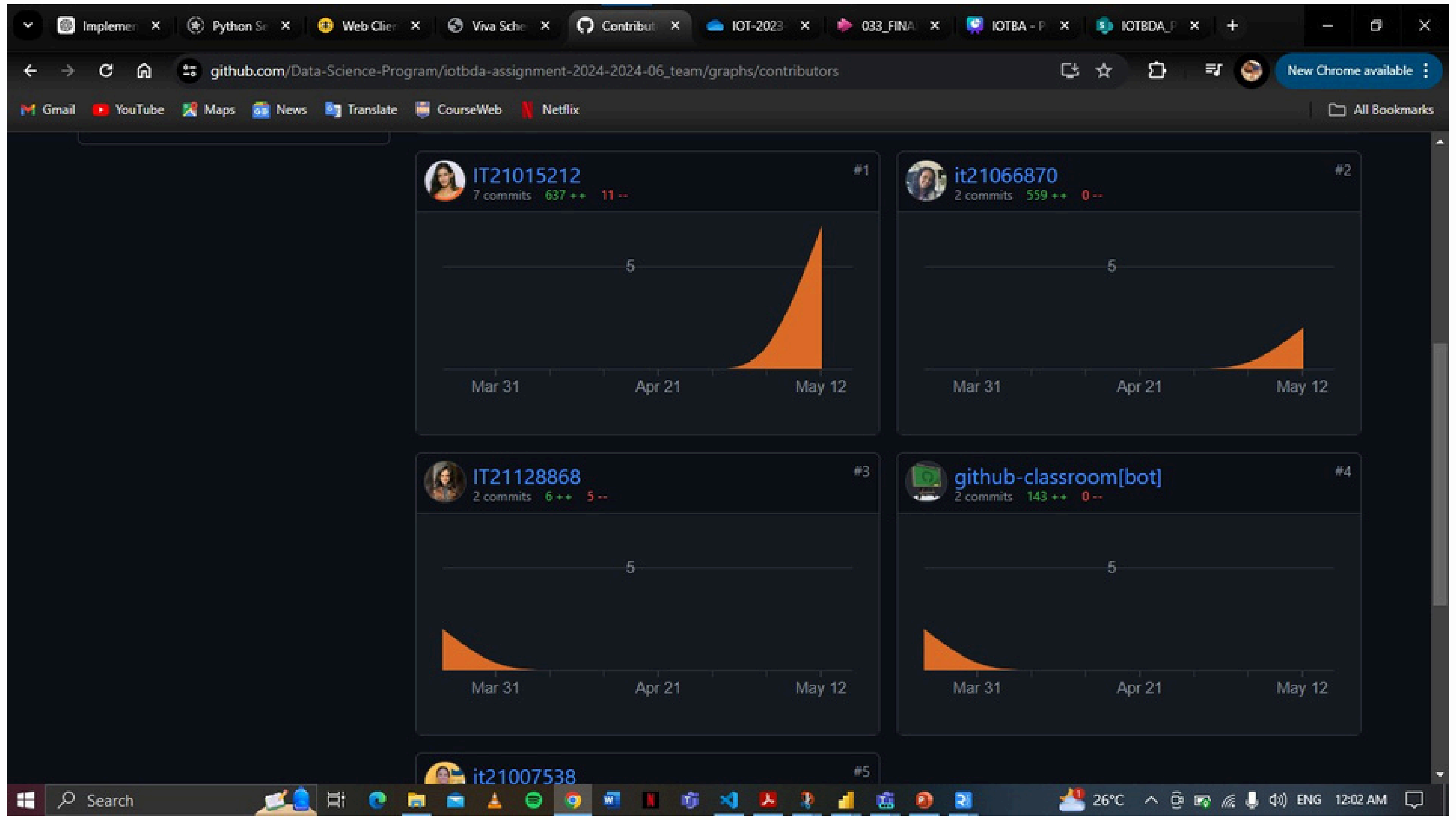
Code Quality & Version Controlling

Version controlling : All members contributed to the git repository.

Code quality: VS Code was utilized. Proper comments, indentations and code quality maintained



```python
# MQTT broker settings
BROKER_ADDRESS = "7358379266ed4c90ba8382bdd99fda2b.s1.eu.hivemq.cloud"
BROKER_PORT = 8883
USERNAME = "raspberrypi"
PASSWORD = "Raspberrypi1234"

# MQTT-Client Initialize
mqtt_client = mqtt.Client()
mqtt_client.username_pw_set(USERNAME, PASSWORD)
mqtt_client.tls_set()
mqtt_client.on_connect = on_connect
mqtt_client.on_disconnect = on_disconnect
mqtt_client.on_publish = on_publish

# Connect to MQTT-broker with error handling
try:
    mqtt_client.connect(BROKER_ADDRESS, BROKER_PORT)
except ConnectionRefusedError:
    print("Connection to MQTT broker failed!")

# DHT sensor initialization
dht_device = adafruit_dht.DHT11(board.D4)

# Servo initialization
servo = AngularServo(18, min_pulse_width=0.0006, max_pulse_width=0.0023)

#Buzzer initialization
buzzer_pin = 17
# Initialize the buzzer with PWM
buzzer = PWMOutputDevice(buzzer_pin)

# Initialize the LED on GPIO pin 16
```



IT21015212 #1
7 commits 637 ++ 11 --

it21066870 #2
2 commits 559 ++ 0 --

IT21128868 #3
2 commits 6 ++ 5 --

github-classroom[bot] #4
2 commits 143 ++ 0 --

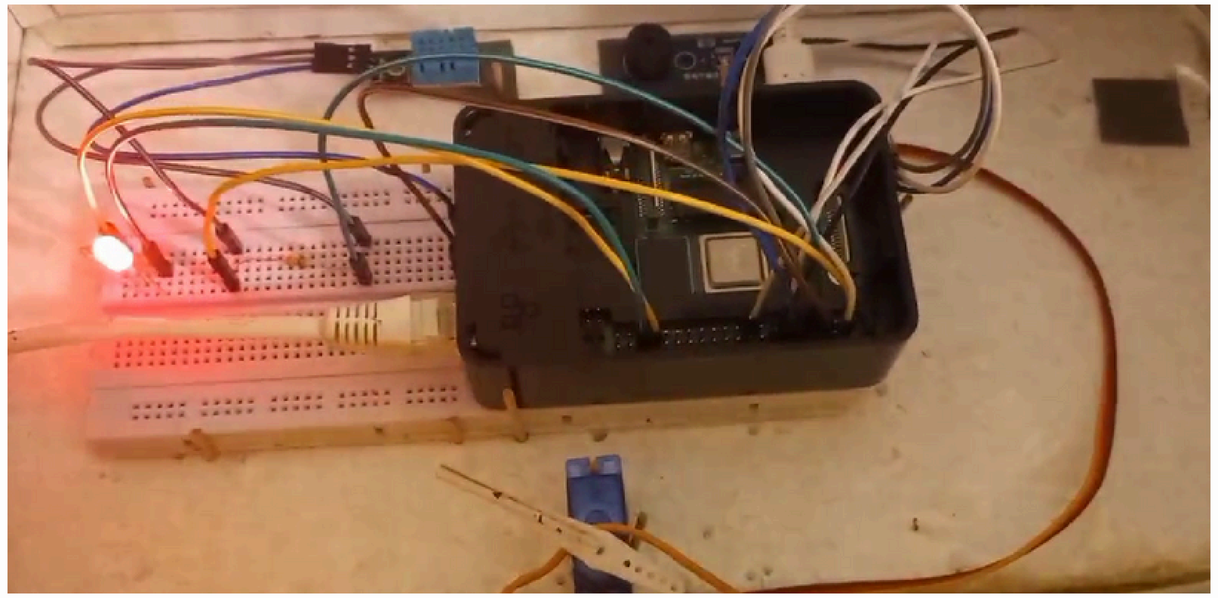
it21007538 #5

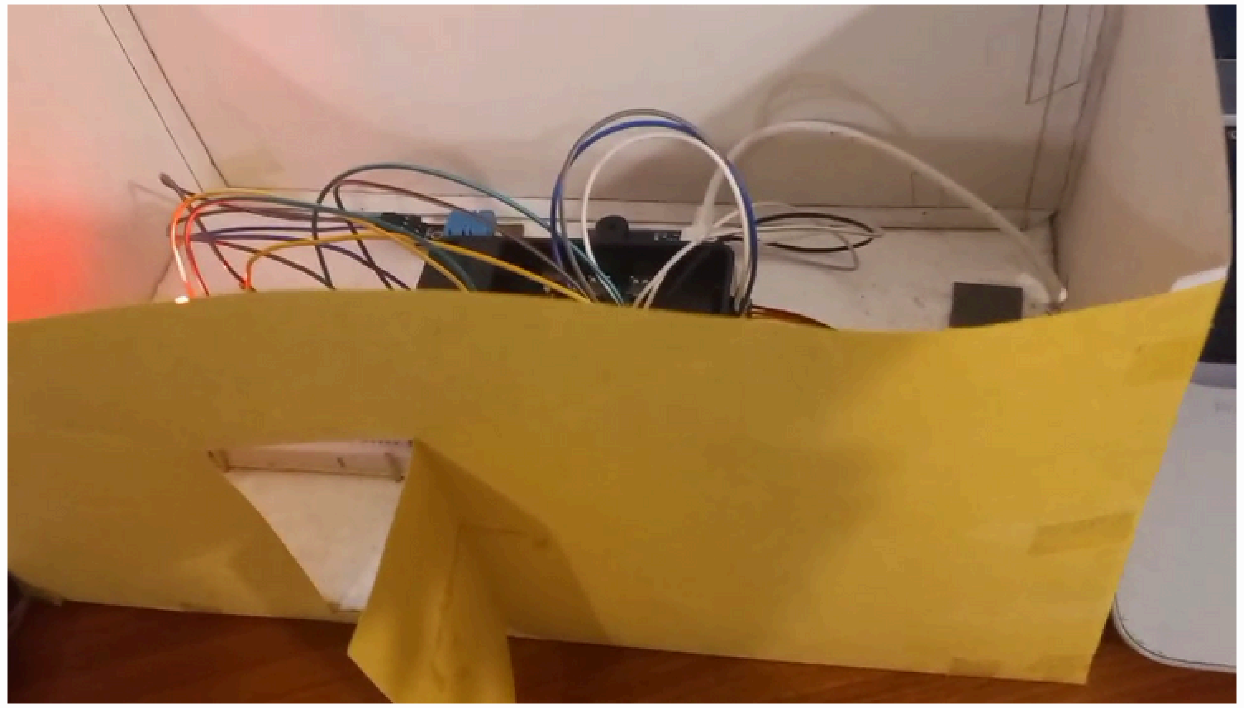
Overall Creativity

The system was developed with practicality in mind.

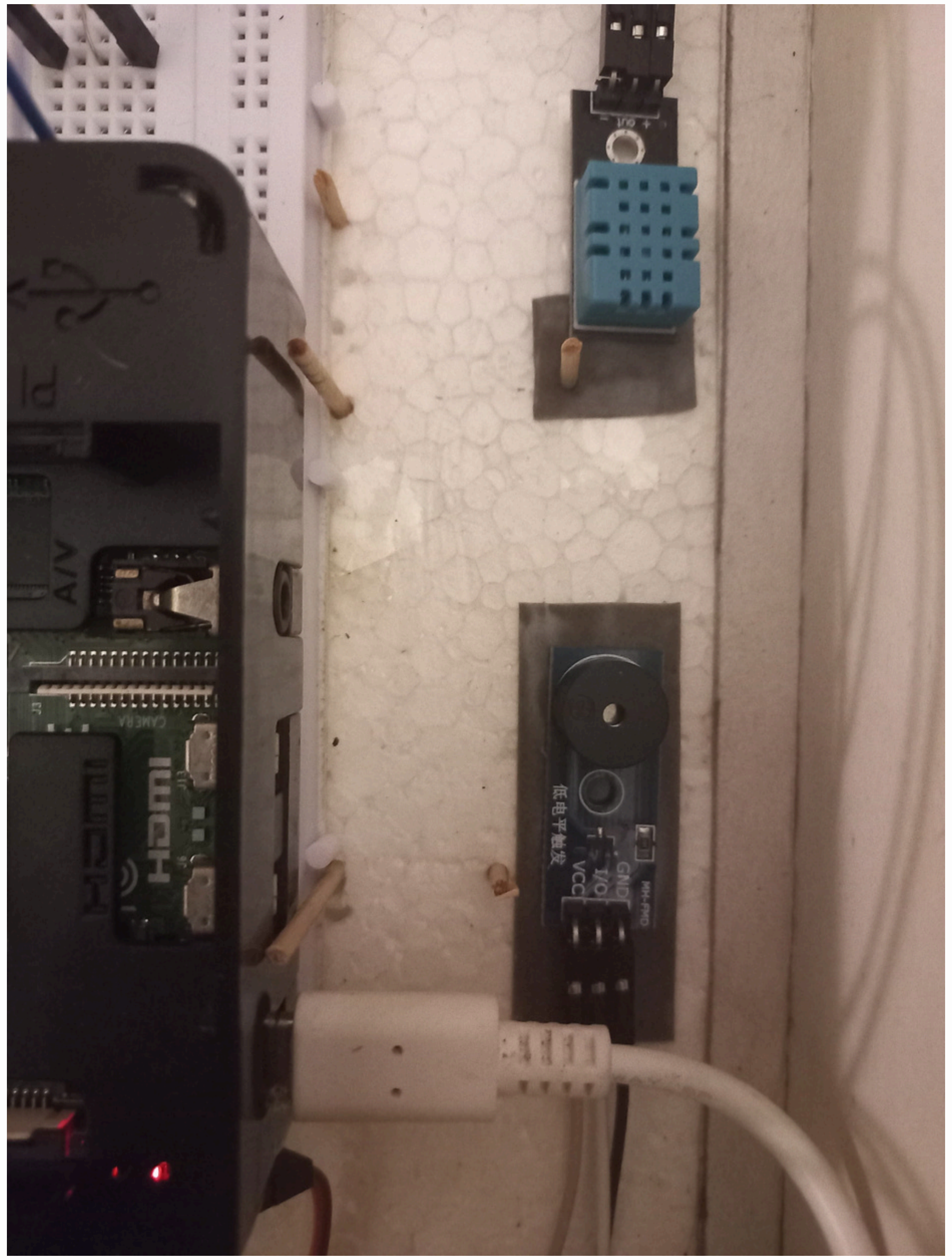
The device's design, development, and creation was done from scratch.

The device targets the commercial markets of **Firefighting Departments, Large Public Buildings, Educational Institutions, Hospitals and Healthcare Facilities and Commercial and Office Buildings.**

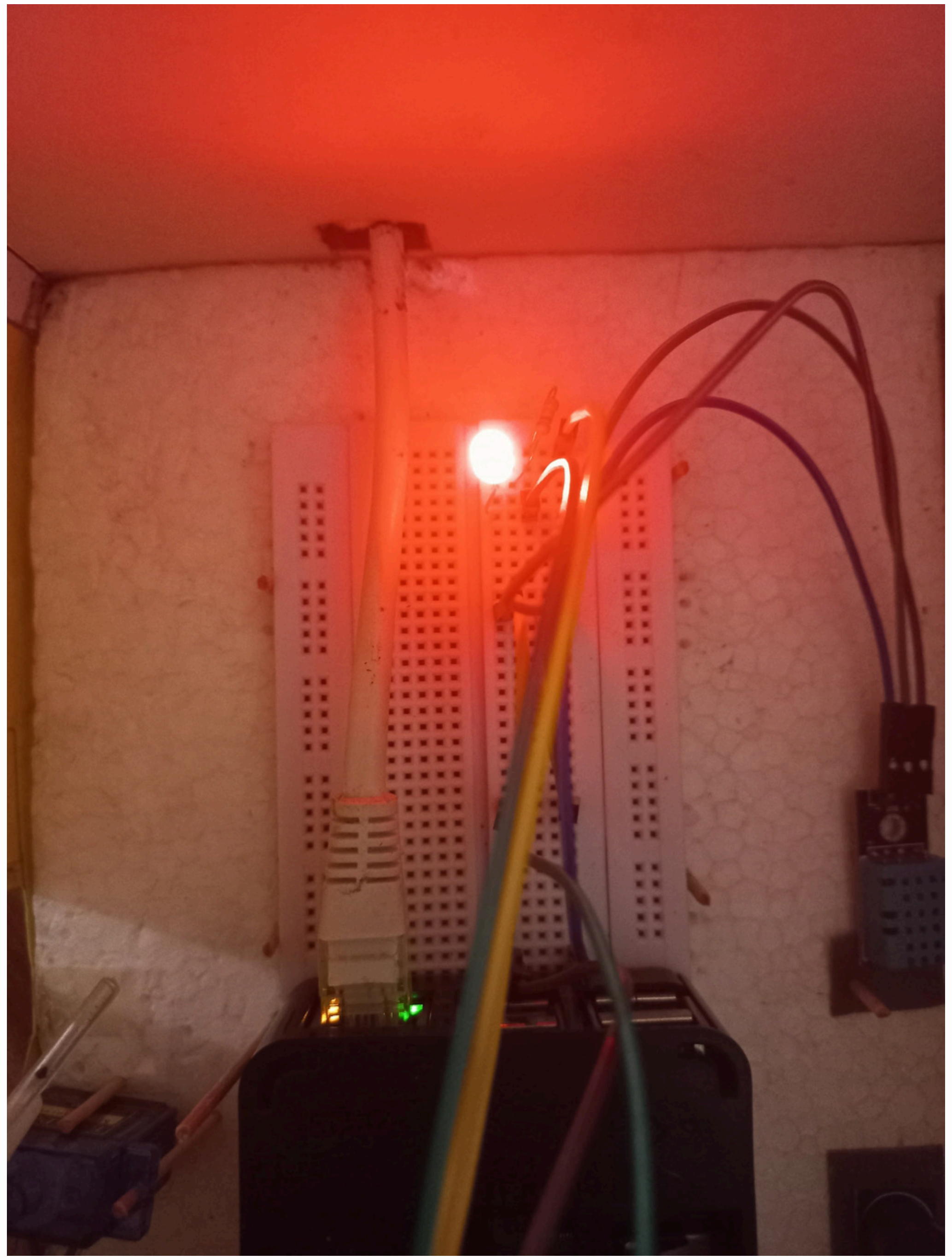




Additionally, the system alerts users in a situation of extreme heat index levels by a visual alert from LED bulb lighting and a audible alert from a buzzer.



Buzzer



LED

THANK YOU!