**AIR QUALITY MONITORING ( I O T )**

**NAME: PREETHI S**

**NM: AU721221106072**

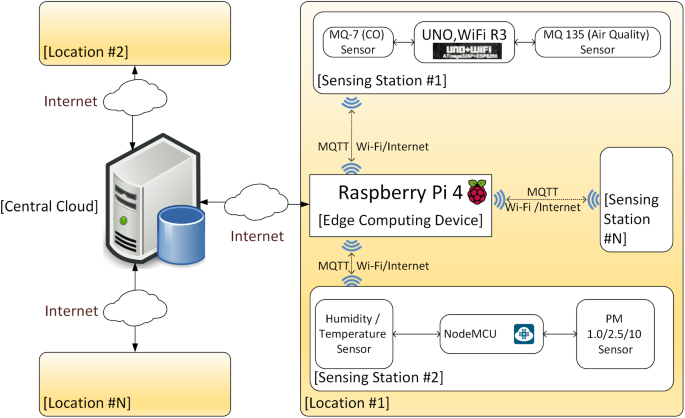
**EMAIL:** [**21ecb11@karpagamtech.ac.in**](mailto:21ecb11@karpagamtech.ac.in)

**PHASE:3**



Air quality monitoring is the systematic measurement of pollutants in the air to assess its cleanliness and safety for humans and the environment. Key pollutants monitored include particulate matter (PM2.5 and PM10), ozone (O3), nitrogen dioxide (NO2), and sulfur dioxide (SO2)

This section proposes a new system that leverages the ever-growing set of single board computers (SBC) that contain hardware powerful enough to perform a reasonable level of computation with low cost and power consumption. The following diagram illustrates the components of the proposed design (Fig. [1](https://link.springer.com/article/10.1007/s40747-021-00476-w#Fig1)).



On the edge of the system exists an instance of SBC, a Raspberry Pi 4. The Raspberry Pi 4 is responsible for controlling and collecting data from multiple sensing stations via Message Queuing Telemetry Transport (MQTT). Hence, the edge device will act as an MQTT broker for all sensing stations, MQTT clients. Each station gathers readings from the connected sensors via a multitude of inputs available in an Arduino-compatible device equipped with Wi-Fi capabilities, such as NodeMCU, Arduino Uno Wi-Fi, Uno, Wi-Fi R3, amongst others. Data could be sent to the Raspberry Pi through its General-Purpose Input Output (GPIO) pins or other inputs if Wi-Fi is unavailable. Sensors may include MQ gas sensors, humidity and temperature sensors like DHT-11 or DHT-22 and PM sensors. The stations may be placed in the same city in industrial or residential locations or distributed across the country, according to the authority’s needs.

After collecting data from the attached sensors for its configured period (mostly 24 h to 1 week) [[20](https://link.springer.com/article/10.1007/s40747-021-00476-w#ref-CR20), [22](https://link.springer.com/article/10.1007/s40747-021-00476-w#ref-CR22)], the edge device is responsible for calculating Air Quality Index (AQI) as well as predicting the next time step or steps (minutes, hours, days, real time) according to its configuration. It may also warn its local vicinity or perform other tasks as configured by the authority or its operator. Afterwards, it may compress available readings and send them to the central cloud for further processing and prediction on a large scale. A system composed of these edge devices would broadcast their raw data to the cloud, which helps in making pivotal decisions and predicting next time-steps for the whole area monitored by the system. The cloud would also help estimate and predict AQI for areas without edge devices, and it may even send corrective data to the edge devices to better predict air pollution concentration level in their local region according to data collected from other neighboring areas.

This system could be used in multiple configurations, including industrial establishments, especially those dealing with environmentally hazardous substances and other factories in general. In addition, the average consumer would benefit from such a system that could work independently from the cloud if required. Also, in governmental settings, this would give the big picture of the air quality situation nationwide. Finally, this system has a flexible configuration as it does not require fixed/static installations and can be mounted on moving vehicles with appropriate adjustments. The system has not been fully implemented yet, just the edge part was implemented using a Raspberry Pi 4 device, and the next phase of this research study is to complete the full implementation.

PROGRAM:

Creating an air quality monitoring system using a Raspberry Pi involves reading data from air quality sensors and processing that data to determine air quality levels.

from gpiozero import MCP3008

import time

# Define the analog pin connected to the MQ135 sensor

mq135\_analog\_pin = 0

# Create an MCP3008 object to read analog data from the specified pin

adc = MCP3008(channel=mq135\_analog\_pin)

# Define the threshold value for air quality classification

threshold = 500

def read\_mq135\_value():

# Read analog data from the MQ135 sensor

sensor\_value = adc.value \* 1023 # MCP3008 provides values between 0 and 1, so multiply by 1023 for the full range

return sensor\_value

def classify\_air\_quality(sensor\_value):

if sensor\_value < threshold:

return "Good"

elif sensor\_value < threshold \* 1.5:

return "Moderate"

else:

return "Poor"

while True:

try:

# Read MQ135 sensor value

mq135\_value = read\_mq135\_value()

# Classify air quality based on the sensor value

air\_quality = classify\_air\_quality(mq135\_value)

# Print air quality status

print("Air Quality: {}".format(air\_quality))

# Wait for some time before taking the next reading

time.sleep(1) # Sleep for 1 second

except KeyboardInterrupt:

# Handle keyboard interrupt (Ctrl+C)

print("Program terminated by user")

break