Multinode Air Purifier with Recommended Asthmatic Management

Session: 2021 – 2025

Submitted by:

Ebaa Haq 2021-CE-22

Faiza Riaz 2021-CE-20

Maham Nadeem 2021-CE-10

Supervised by:

Dr. Yasir Saleem

Co-Supervised by:

Engr. Afeef Obaid

Department of Computer Engineering

University of Engineering and Technology

Lahore Pakistan

Multinode Air Purifier with Recommended Asthmatic Management

(Session 2021 Computer Engineering)

This thesis is submitted to the Department of Computer Engineering, University of Engineering and Technology, Lahore, as partial fulfillment of the requirements for the award of a Bachelor of Science degree in Computer Engineering.

Internal Examiner
Signature:

Name:

Designation:

External Examiner
Signature:

Name:

Designation:

Chairman
Signature:

Dr. Ali Hammad Akbar

Dean
Signature:

Prof.Dr. Muhammad Shoaib

Department of Computer Engineering

University of Engineering and Technology

Lahore Pakistan

Declaration

We hereby declare that the work presented in this thesis is the result of our own efforts, except where otherwise acknowledged. This work has not been submitted, either in whole or in part, for any other degree or professional qualification. All funds are utilized and all accounts are cleared from our side with our supervisor and department.

Ebaa Haq
Signed: ____________________
Date: **02/06/2025**

Faiza Riaz
Signed: ____________________
Date: **02/06/2025**

Maham Nadeem
Signed: ____________________
Date: **02/06/2025**

Acknowledgments

We thank Allah Almighty sincerely for granting us the strength, resilience, and guidance to achieve this task. We would like to extend our sincere thanks to our supervisor, Dr. Yasir Saleem, and co-supervisor, Sir Afeef Obaid, for their unflinching support, constructive criticism, and wise guidance. Their advice and support were instrumental in the formulation of this project and in helping us overcome every hurdle. Furthermore, we are sincerely grateful to our parents for their unconditional love and unwavering support. Their trust in us has been our greatest source of strength. With such support inspiring our efforts, it is a privilege to submit this thesis as a testament to the collective efforts, motivation, and trust of all those who have joined us on this journey.

Dedicated to our Families

Contents

List of Figures

List of Tables

Abbreviations

ADC	**A**nalog-to-**D**igital **C**onverter
AQI	**A**ir **Q**uality **I**ndex
API	**A**pplication **P**rogramming **I**nterface
BLE	**B**luetooth **L**ow **E**nergy
CO	**C**arbon **M**onoxide
CO_2	**C**arbon **D**ioxide
CPU	**C**entral **P**rocessing **U**nit
DHT22	**D**igital **H**umidity and **T**emperature **S**ensor
ESP32	**E**spressif **S**ystems **M**icrocontroller (32-bit)
FYP	**F**inal **Y**ear **P**roject
GUI	**G**raphical **U**ser **I**nterface
HEPA	**H**igh **E**fficiency **P**articulate **A**ir
IOT	**I**nternet **O**f **T**hings
LCD	**L**iquid **C**rystal **D**isplay
NO_2	**N**itrogen **D**ioxide
O_3	**O**zone
PM2.5	**P**articulate **M**atter 2.5 µm
PM10	**P**articulate **M**atter 10 µm
SDG	**S**ustainable **D**evelopment **G**oal
SO_2	**S**ulfur **D**ioxide
TVOC	**T**otal **V**olatile **O**rganic **C**ompounds
VOC	**V**olatile **O**rganic **C**ompounds
WHO	**W**orld **H**ealth **O**rganization
Wi-Fi	**W**ireless **F**idelity

Abstract

Indoor air pollution poses significant health risks, particularly for individuals suffering from asthma and other respiratory conditions. Despite the prevalence of air purifiers, most commercially available systems lack customization to meet individual health needs. This project introduces a smart multinode air purification system designed specifically for asthmatic users. The system incorporates two sensor nodes strategically placed in different locations to continuously monitor critical air quality parameters, including carbon dioxide (CO_2), humidity, temperature, and dust levels $PM_{2.5}$. Real-time data are transmitted to a central controller, which evaluates the Air Quality Index (AQI) and activates the appropriate air purifier based on current conditions. Each sensor node autonomously manages its respective purifier, enabling targeted and localized air purification. A user-friendly dashboard facilitates visualization of real-time data and purifier control functionalities. Asthmatic users have the option to manually operate the purifier for specific durations, while non asthmatic users are limited to monitoring air quality. The system employs wireless communication for seamless data transmission, ensuring scalability, reliability, and energy efficiency. By integrating advanced technology with customized asthma management, this project aims to foster a healthier indoor environment and enhance the quality of life of users, particularly those with respiratory sensitivities.

Keywords: Air Quality Monitoring, Asthma Management, Smart Purifier System,the Air Quality Index (AQI), IoT in Healthcare, Environmental Sensing, Real-time Dashboard

Chapter 1

Introduction

1.1 Overview

The Multinode Air Purifier with Recommended Asthmatic Management system is an intelligent solution designed to improve indoor air quality, especially for individuals with asthma and other respiratory conditions. It monitors key environmental factors such as particulate matter (PM), carbon dioxide (CO_2), temperature, and humidity in real time, and automatically activates air purifiers when unhealthy conditions are detected.

The system ensures air quality control in several indoor zones by implementing a multinode sensor network. It provides automated response and continuous monitoring through the use of wireless communication, embedded systems, and environmental sensors. For individualized asthma treatment, users can also manually adjust purification settings and view real-time air quality data via a web-based interface.

This decentralized and scalable design supports efficient air purification in homes, schools, and workplaces promoting healthier environments and reducing the risk of asthma-related health issues.

1.2 Motivation

The rising prevalence of asthma and its strong correlation with poor indoor air quality highlights the urgent need for personalized and intelligent air purification solutions. Traditional air purifiers often provide uniform responses that fail to meet the specific needs of individuals with respiratory conditions such as asthma. These limitations underscore the necessity for a more adaptive and responsive system.

The Multinode Air Purifier with Recommended Asthmatic Management system is driven by the goal of addressing this gap by offering a tailored solution that actively monitors and responds to indoor environmental changes in real time. By integrating sensor based monitoring with intelligent automation, the system aims to reduce asthma triggers and improve the quality of life of users in both residential and commercial spaces.

This project is motivated by the belief that timely interventions, such as automatic purifier activation based on air quality metrics, can significantly reduce respiratory discomfort and potential health risks. Our objective is to empower users, especially those with asthma, with a scalable and accessible system that ensures healthier indoor environments and supports proactive respiratory care.

1.3 Problem Statement

Indoor air quality plays a critical role in respiratory health, particularly for individuals with asthma and other chronic respiratory conditions. However, current air purification systems are frequently generic in nature, unable to adjust to shifting environmental conditions or satisfy individualized medical requirements. These systems' efficacy in delicate health situations is limited because they usually lack user-specific control, multi-zone coverage, and real-time monitoring.

The absence of intelligent, responsive features in conventional solutions creates a significant gap in managing asthma triggers and ensuring optimal indoor air quality. This gap is particularly critical in environments where air quality fluctuates and timely intervention can prevent respiratory distress.

This research addresses the pressing need for a smart, multinode air purification system that offers continuous air quality monitoring, automatic purifier activation based on critical environmental metrics (such as particulate matter and carbon monoxide), and personalized user interaction through a web-based interface. The goal is to deliver an adaptive, scalable solution that enhances respiratory health management especially for asthmatic users by integrating intelligent control, real-time feedback, and decentralized air purification.

1.4 FYP Objectives

The primary goal of Multinode Air Purifier with Recommended Asthmatic Management project is to develop a smart, reliable air quality monitoring system designed to improve indoor environments, particularly for individuals with asthma that:

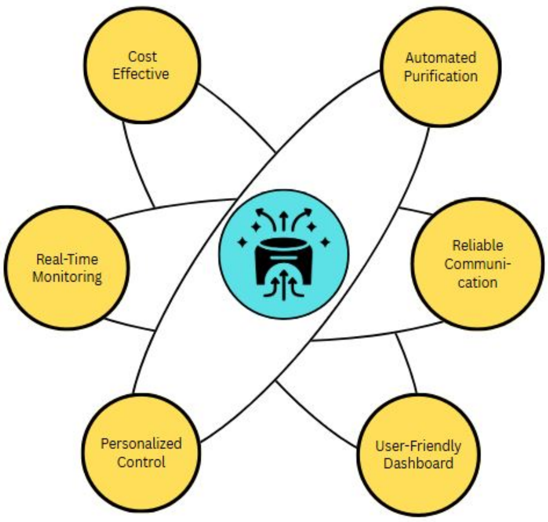

Figure 1.1: Fyp Objectives

1. Real-Time Air Quality Monitoring: Develop sensor nodes to monitor essential air quality indicators in real time, including temperature, humidity, (CO_2) concentrations, and $_{2.5}$ particulate matter. The objective is to continuously monitor and assess indoor air quality, especially for the treatment of asthma.

2. Automated Air Purification Control: Design an intelligent system that automatically activates air purifiers based on real-time air quality index (AQI) values. The system will measure and assess pollutant levels, and when the AQI exceeds a predefined threshold, the purifier will be triggered for automatic operation.

3. Personalized Control for Asthmatic Users: Provide manual control options for asthmatic users to adjust air purifier settings according to their comfort level. Users will be able to turn on/off the purifiers, set operation times, and personalize the air quality environment to their specific needs.

4. User-Friendly Dashboard for Air Quality Monitoring: Develop an intuitive and interactive web-based dashboard that displays real-time sensor data in a clear, user-friendly format. This will enable users to easily monitor the air quality in different zones, view historical data, and receive notifications about air quality status.

5. Reliable Wireless Communication Across Nodes: Implement a wireless communication protocol to enable stable and efficient communication between the sensor nodes and the central control system. This system ensures that

all sensor data is reliably transmitted and allows easy expansion to multiple nodes for greater scalability.

6. Energy Efficiency and Cost-Effectiveness: Focus on designing a system that optimizes energy usage by activating air purifiers only when necessary based on real-time AQI readings. The system will also consider cost-effectiveness by utilizing low-power sensors and efficient control algorithms to minimize operational expenses.

1.5 Socioeconomic Benefits

The suggested approach has the potential to bring about a number of socioeconomic advantages, such as

- **Health Improvement:** By keeping indoor air clean, the system reduces asthma triggers and respiratory problems. Vulnerable groups like children, the elderly, and people with long-term respiratory disorders will especially benefit from this.
- **Cost-Effective Solution:** The system can still be implemented in low-budget settings, such as public schools, community centers, and clinics, by using reasonably priced components like the ESP32 and Arduino.
- **Reduced Healthcare Costs:** Continuous air quality monitoring and timely intervention can prevent exacerbation of respiratory issues, potentially reducing hospital visits and medication needs, thereby lowering healthcare expenses for families and public health systems.
- **Scalability and Adaptability:** The modular and wireless nature of the system allows it to be easily implemented across various indoor environments homes, offices, classrooms, and healthcare facilities ensuring broad applicability and scalability.
- **User Empowerment:** Through real-time monitoring and manual control features, the system empowers individuals to actively manage their indoor air quality, fostering greater awareness and healthier living habits.
- **Technological Awareness:** The project promotes the integration of IoT-based health solutions in daily life, contributing to increased digital literacy and encouraging the adoption of smart health technologies within communities.

Chapter 2

Literature Review

2.1 Background

Over the past decades, increasing attention has been directed toward indoor air quality (IAQ) due to its significant impact on public health, particularly in relation to respiratory diseases such as asthma. Asthma is a chronic inflammatory condition of the airways, characterized by symptoms including wheezing, chest tightness, shortness of breath, and coughing. [1] According to the World Health Organization (WHO), asthma affected an estimated 262 million people globally in 2019, contributing to over 455,000 deaths. [2]

Environmental factors, especially indoor air pollutants, play a critical role in the onset and exacerbation of asthma symptoms. Common indoor pollutants include particulate matter ($PM_{2.5}$ and PM_{10}), volatile organic compounds (VOCs), carbon monoxide (CO), ozone (O_3), nitrogen dioxide (NO_2), and biological allergens such as dust mites, mold spores, pet dander, and cockroach allergens. These pollutants are often generated through everyday activities such as cooking, smoking, or inadequate ventilation, and can lead to both short-term respiratory discomfort and long-term health complications.

Indoor sources of these pollutants are diverse. For instance, cooking with gas stoves has been shown to increase indoor NO_2 levels beyond WHO safety standards, posing risks particularly in smaller or poorly ventilated homes. [3] Additionally, biological allergens from dust mites, pets, mold, and pests are prevalent in many indoor environments and are known triggers for asthma attacks. [4]

The impact of indoor air pollution on asthma is especially pronounced among vulnerable populations, including children, the elderly, and individuals in low-income communities. These groups often reside in environments with higher pollutant levels and have limited access to healthcare resources, exacerbating health disparities. [5]

Children, the elderly, and low-income populations are especially vulnerable due to prolonged exposure to poor air quality and limited access to healthcare services. Studies, including those by Schultze-Werninghaus (2006) [6], have shown that asthma patients experience relief in high-altitude environments, where pollutant concentrations are lower, highlighting the direct correlation between air quality and respiratory health.

As a result, the focus of asthma management has evolved from solely pharmacological treatment to incorporating environmental control strategies. Research by Zuraimi et al. (2007) [7] across 104 child care centers demonstrated that natural ventilation was more effective than mechanical air conditioning in reducing pollutant concentrations and associated asthma symptoms.

Technological advancements have further enabled the development of high-efficiency particulate air (HEPA) filters and smart purification systems. HEPA filters are capable of removing microscopic particles that commonly trigger asthma attacks. Numerous studies confirm their efficacy in significantly improving IAQ, particularly in urban environments with high pollution levels.

Moreover, the integration of Internet of Things (IoT) technology has enabled real-time air quality monitoring and automated system responses. These systems can detect changes in pollutant levels and respond accordingly by adjusting purification settings, ensuring a healthier indoor environment. This proactive approach allows for timely interventions, especially crucial for individuals with respiratory sensitivities. [8]

The COVID-19 pandemic further emphasized the role of air quality in respiratory health. During global lockdown, reduced pollution levels were associated with measurable improvements in asthma symptoms in urban populations, reinforcing the importance of effective air purification in asthma management. [9]

In light of these findings, the need for intelligent, decentralized air purification systems has become evident. By combining real-time monitoring, automated control, and user interaction, such systems can offer tailored support for individuals with asthma, ultimately improving quality of life and reducing health risks associated with poor indoor air quality.

This project builds on these advancements by proposing a Multinode Air Purifier with Recommended Asthmatic Management, a smart system aimed at improving

IAQ and empowering asthma patients with a personalized, responsive solution.

2.2 What others have done

In recent years, considerable attention has been directed toward the impact of indoor air pollution on respiratory health, particularly asthma. The home environment, a primary source of indoor exposure, harbors various pollutants including particulate matter (PM), volatile organic compounds (VOCs), carbon monoxide (CO), nitrogen dioxide (NO_2), ozone (O_3), and biological allergens such as dust mites, pet dander, and mold spores. These pollutants have been shown to trigger asthma symptoms, exacerbate existing conditions, and, in some cases, contribute to the onset of asthma in vulnerable individuals. This concern is heightened among children, the elderly, and those living in low-income communities, where poor ventilation, overcrowding, and the use of substandard materials contribute to higher pollutant exposure levels. As a result, there is a growing emphasis on developing technological and behavioral interventions aimed at improving indoor air quality (IAQ) and reducing asthma risks [10].

One such intervention is the implementation of air purification systems, designed to capture airborne pollutants and allergens. While conventional air purifiers show promise in alleviating asthma symptoms, their efficacy varies based on pollutant type, purifier design, and user-specific needs. In parallel, the advent of Internet of Things (IoT) technology has led to the creation of real-time air monitoring systems. These IoT-based solutions collect and transmit air quality data, enabling smart devices, such as air purifiers, to automatically respond to changes in the indoor environment. [11]

Air pollution has now become growing concern, particularly in urban environments where rapid industrialization and vehicular emissions significantly contribute to the degradation of air quality. These pollutants include volatile organic compounds (VOCs), as illustrated in Figure 2.1, carbon dioxide (CO_2), as shown in Figure 2.2, particulate matter (PM), as shown in Figure 2.3, and nitrogen dioxide (NO_2), as shown in Figure 2.4. Other harmful gases like sulfur dioxide (SO_2) and ozone (O_3) also contribute significantly.

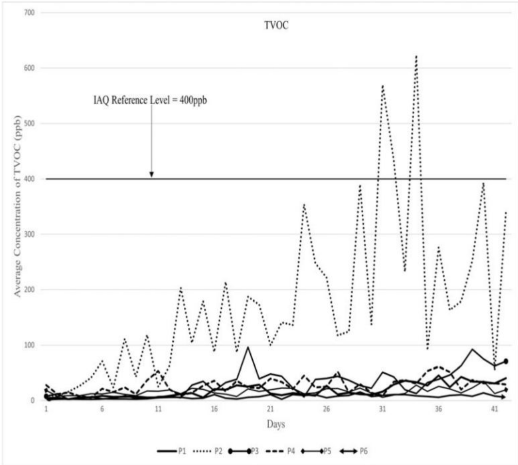

FIGURE 2.1: Volatile Organic Compounds

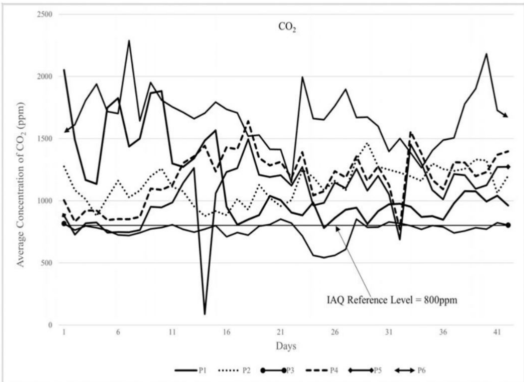

FIGURE 2.2: Carbon Dioxide

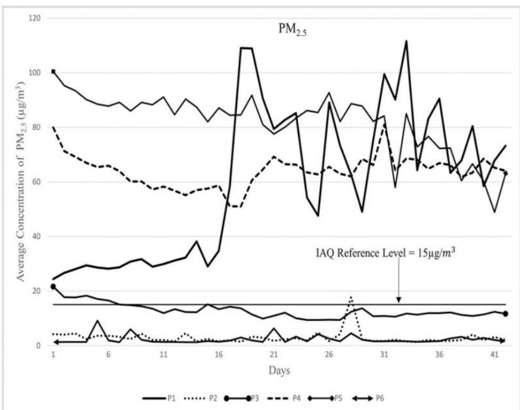

FIGURE 2.3: Particulate Matter $PM_{2.5}$

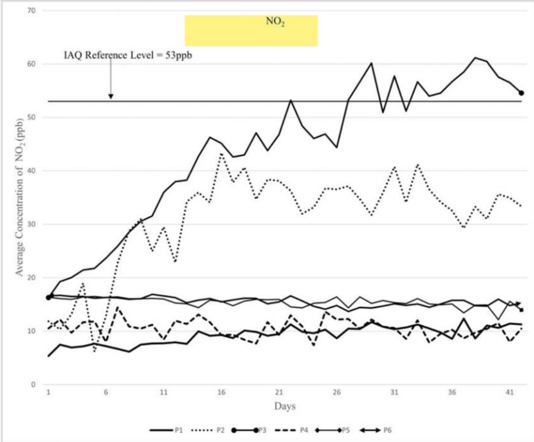

FIGURE 2.4: Nitrogen Dioxide

These pollutants have serious implications for human health, especially for individuals suffering from chronic respiratory diseases such as asthma [12]. Exposure to these pollutants can trigger asthma attacks, exacerbate existing conditions, and lead to long-term health complications.

To address these pollutants, researchers have proposed sensor-based air monitoring systems. As shown in Figure 2.5, modern air quality systems incorporate wireless communication between master and slave nodes, allowing for efficient data transmission. These systems, tested with sensors such as the DSM501A (Figure 2.6), demonstrated high accuracy in detecting $PM_{2.5}$, classifying air quality based on WHO guidelines, and identifying particles as small as 1 micrometer. [12] [13] [14]

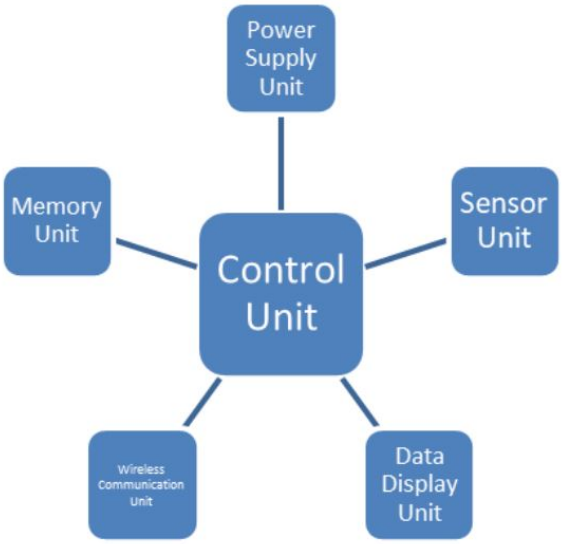

Figure 1. Air quality monitoring system diagram

FIGURE 2.5: Air quality monitoring system

Table 2. Sensor category conversion table

No	PM 2.5 (μg/m³)	AQI Level	Status
1	0,02	0,08	Good
2	0,05	0,21	Good
3	1,00	4,13	Good
4	15,10	82,26	Moderate
5	20,50	93,80	Moderate
6	56,20	179,58	Unhealthy
7	2,20	9,09	Good
8	5,50	22,73	Good
9	155,40	281,79	Very Unhealthy
10	25,70	54,91	Moderate

FIGURE 2.6: Accuracy in detecting $PM_{2.5}$ pollutants

Further studies have emphasized the role of low-cost interventions. A randomized control trial investigated the effect of a low-cost air filter (Figure 2.7) and an asthma home management manual on adult asthma patients. The results showed statistically significant improvements in quality of life (QoL) scores, particularly in environmental and activity domains (p = 0.044 and p = 0.041, respectively), as depicted in Figure 2.8. [14]

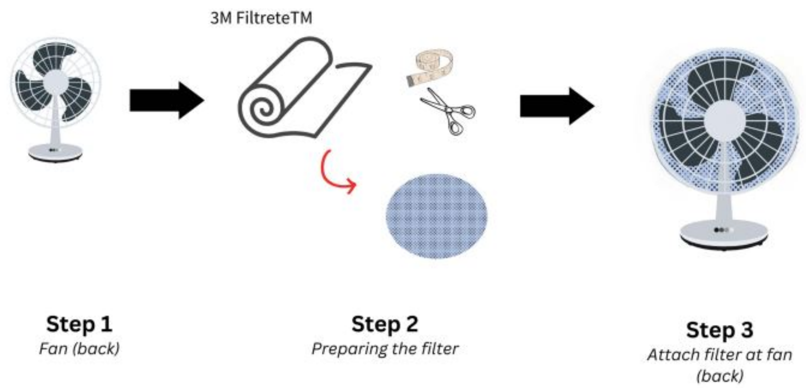

Figure 2 Air filter set-up.

FIGURE 2.7: Low-Cost Air Filter

Table 3 Change of Asthma Quality of Life at Baseline, Post-Intervention, and Follow-Up

mAQLQ	Median (25–75 Percentile)					
	Home Management	P-value†	Air Filter	P-value†	Control	P-value†
Total score						
Baseline	89.0 (80.8–96.3)		95.5 (82.3–100.3)		93.0 (84.5–99.0)	
Post-intervention	93.0 (80.3–103.0)	0.33	97.0 (90.0–101.3)	0.044	90.5 (77.0–99.0)	0.12
Follow up	95.5 (83.3–104.0)	0.63	94.0 (82.3–101.0)	0.95	93.0 (80.8–99.0)	0.62
Emotion						
Baseline	31.5 (26.7; 34.0)		32.0 (31.0; 34.2)		31.0 (26.7; 33.0)	
Post-intervention	32.0 (26.5; 34.0)	0.49	31.0 (29.0; 34.0)	0.15	31.0 (26.0; 33.2)	0.50
Follow up	32.0 (28.3; 35.0)	0.68	32.0 (29.0; 34.0)	0.98	31.0 (27.7; 34.2)	0.16
Activity						
Baseline	18.0 (16.7; 21.0)		20.0 (16.7; 21.0)		19.0 (17.0; 21.0)	
Post-intervention	20.0 (17.0; 21.0)	0.65	21.0 (19.0; 21.0)	0.09	19.0 (17.0; 21.0)	0.38
Follow up	20.5 (17.2; 21.0)	0.95	20.0 (16.7; 21.0)	0.80	19.0 (17.0; 21.0)	0.93
Environmental						
Baseline	15.0 (10.7; 16.0)		16.5 (14.0; 19.0)		17.0 (13.0; 19.0)	
Post-intervention	17.0 (13.7; 21.0)	0.004	17.0 (16.0; 19.0)	0.20	17.0 (13.0; 19.0)	0.83
Follow up	17.0 (11.7; 21.0)	0.041	17.0 (14.0; 19.0)	0.82	17.0 (12.7; 19.0)	0.78
Symptoms						
Baseline	26.0 (23.0; 28.0)		27.0 (20.7; 28.0)		27.5 (23.0; 28.0)	
Post-intervention	26.0 (20.0; 27.0)	0.28	28.0 (25.5; 28.0)	0.002	26.0 (21.7; 28.0)	0.12
Follow up	27.0 (23.7; 28.0)	0.79	27.0 (20.7; 28.0)	0.34	27.5 (22.0; 28.0)	0.44

FIGURE 2.8: QoL Scores

Similar results were observed in pediatric populations. A randomized crossover trial involving thirty children showed that air purifiers significantly reduced $PM_{2.5}$ levels, leading to decreased asthma symptoms and medication usage during filter-on periods. [15]

In parallel, modeling techniques like HYSPLIT (Hybrid Single-Particle Lagrangian Integrated Trajectory) have been used to trace pollutant dispersion. A study in Lucknow, India applied HYSPLIT to examine seasonal variations in PM_{10}, NO_2, and SO_2 levels, revealing that wind patterns and external sources heavily influence local air quality. [16]

Systematic reviews have highlighted the effects of indoor sources such as cleaning agents, formaldehyde, and damp environments. One meta-analysis of 94 studies found moderate certainty that mold and certain chemicals significantly increased asthma risk, reinforcing the need for enhanced ventilation and filtration strategies. [17]

Air filtration systems, widely used for improving IAQ, have been studied for their effectiveness in reducing allergic rhinitis and asthma symptoms. A meta-analysis on the use of air filters concluded that while air filters do reduce the symptoms of allergic rhinitis, their impact on medication use and overall quality of life is not significant. This finding suggests that while air filters can be part of the solution, they may not be sufficient on their own in managing asthma. [18]

Recent IoT-based systems using sensors like ESP8266, STM32F407IG, and other platforms have facilitated real-time pollution detection. These systems not only alert users but also initiate automated responses, though calibration and accuracy remain areas for improvement. [19]

Several studies, including Zuraimi et al. (2007) [7] and Schultze-Werninghaus (2006) [6] demonstrated that natural ventilation and low-pollution environments significantly improve asthma outcomes. Other research, like Chapman et al. (2003), [20] reinforced the importance of HEPA filters in urban pollution control.

Events such as dust storms have been studied for their impact on respiratory health. A 2024 study by Kouis et al. confirmed that HEPA-equipped air purifiers significantly improved lung function in children with asthma during these events. [21]

The COVID-19 pandemic served as a natural experiment on air quality's role in asthma. Research by Arshad Peer Mohamed et al. (2023) noted significant reductions in asthma exacerbations during lockdowns due to improved air quality. [9]

Finally, the integration of artificial intelligence (AI) and IoT into air monitoring systems has advanced the field. Chadalavada et al. (2025) and Gryech et al. (2024) [22] showed that AI-enhanced systems enable accurate forecasting and real-time intervention, potentially revolutionizing asthma management. [23]

Research has highlighted the significant role of air quality in asthma exacerbations and the potential of air purification systems to mitigate the impact of environmental pollutants. Studies have shown that air purification systems, particularly those using HEPA filters, can significantly reduce indoor air pollution and improve asthma control. The integration of these systems with other asthma management strategies, including ventilation and real-time air quality monitoring, holds great promise for improving the quality of life for individuals with asthma.

2.3 Literature Survey

Schultze-Werninghaus (2006) [6] was one of the first to establish the potential benefits of high-altitude environments for asthma management. This early observation laid the foundation for future studies on the effects of air quality on asthma, particularly in high-pollution urban environments.

Zuraimi et al. (2007) [7] provided further evidence of the importance of ventilation strategies in improving indoor air quality. Their study demonstrated that natural ventilation (NV) was more effective in reducing indoor pollutants than air-conditioned environments. This suggests that air purification systems that enhance natural ventilation could be effective in reducing asthma symptoms, particularly in child care centers.

Chapman et al. (2003) [20] focused on the impact of outdoor air pollution on asthma exacerbations. Their findings indicated that urban pollution, including particulate matter and gases like NO_2 and O, significantly worsens asthma symptoms. The authors proposed that air purification systems equipped with HEPA filters could help mitigate the effects of these pollutants, improving asthma management in urban areas.

Kouis et al. (2024) [21] extended this research by examining the effectiveness of air filtration during dust storm seasons. The study found that children with asthma who used air purifiers during high dust storm periods experienced significant improvements in asthma control and lung function. This highlights the importance of air purification systems in regions prone to environmental pollution.

The COVID-19 pandemic has also provided valuable insights into the effects of air quality on asthma. Arshad Peer Mohamed et al. (2023) [9] found that reduced air pollution during the lockdown period resulted in significant improvements in asthma symptoms, reinforcing the connection between air quality and respiratory health.

The integration of AI and IoT technologies has further enhanced the capabilities of air purification systems. Chadalavada et al. (2025) [22] explored the role of machine learning in predicting air pollution levels, which can be used to optimize air quality management. IoT-enabled sensors, as discussed by Gryech et al. (2024), allow for real-time monitoring of air quality, enabling more targeted interventions in asthma management.

Indoor air quality (IAQ) continues to be a primary area of investigation. The Feasibility of Residential Air Quality Monitoring to Address Asthma Outcomes (2024) [12] evaluated the impact of HEPA air purifiers and targeted cleaning interventions. The study found that personalized IAQ improvements significantly enhanced asthma control and quality of life, especially in underserved communities.

IoT-based monitoring systems have also proven effective. The study Air Quality

Monitoring Using Multi-Node Slave IoT (2024) [13] demonstrated the use of real-time pollutant data to manage indoor environments, while Development of an IoT-Enabled Air Pollution Monitoring and Air Purifier System (2023) [24] illustrated the capability of these systems to autonomously trigger purification responses to pollutant spikes.

Low-cost interventions have also been investigated. The Effectiveness of Asthma Home Management Manual and Low-Cost Air Filter on Quality of Life Among Asthma Adults (2024) [13] showed that combining behavioral management tools with air filters improved both environmental and activity-related quality of life domains in adult asthma patients. Similarly, Effects of Indoor Air Purifiers on Children with Asthma (2020) [24] found significant reductions in $PM_{2.5}$ levels and corresponding improvements in symptom scores.

Environmental monitoring tools have expanded to outdoor air quality as well. The study Variation of Ambient Air Pollutants Concentration Over Lucknow City, Trajectories and Dispersion Analysis Using HYSPLIT4.0 (2022) [16] modeled the spatial and seasonal variation of pollutants like PM_{10}, SO_2, and NO_2, highlighting the influence of meteorological factors and pollution sources on asthma outcomes.

Furthermore, a systematic review titled The Impact of Indoor Pollution on Asthma-Related Outcomes (2024) [16] evaluated 94 studies linking pollutants such as formaldehyde, mold, and cleaning agents to increased asthma risk. While the certainty of evidence was moderate, the findings strongly supported the need for improved IAQ in homes, particularly for sensitive populations.

In conclusion, the literature collectively emphasizes the vital role of air purification systems in mitigating asthma symptoms and improving quality of life. From advanced sensor integration to low-cost behavioral strategies, these interventions—especially when personalized and continuously monitored—offer promising solutions for asthma management. The continued integration of AI and IoT technologies into these systems further enhances their potential, marking a significant shift toward data-driven environmental health management.

2.4 Scope of FYP

The scope of this project revolves around the development of a multinode smart air purification system aimed at enhancing indoor air quality, especially for individuals suffering from respiratory conditions such as asthma. The system utilizes two sensor nodes placed in different indoor locations to continuously monitor various environmental parameters, including humidity, temperature, carbon dioxide

(CO_2), and particulate matter or dust levels. Each sensor node comprises an Arduino Uno microcontroller connected to the respective sensors. These nodes wirelessly transmit data to a central control unit built on an ESP32 microcontroller, using ESP-NOW or WiFi protocols to ensure reliable, low-latency communication.

The sensor node control unit calculates the Air Quality Index (AQI) in real time based on sensor readings and sends this data to the central unit. When the AQI exceeds a predefined threshold indicating poor air quality, the system automatically activates the air purifier associated with that specific node.

In addition to automatic activation, the system offers a web-based dashboard that allows users to manually control the purifiers. For users with asthma, the dashboard provides full functionality, enabling them to view real-time data from both nodes and manually turn on purifiers for a chosen duration after which the purifier switches off automatically. For non-asthmatic users, the dashboard is available in a read-only mode, displaying real-time gas values from both nodes without control access. The system is built on modern wireless communication protocols such as ESP-NOW or Wi-Fi to ensure reliable and low-latency data exchange between the sensor nodes and the control unit. This makes the solution highly adaptable and efficient for use in various indoor environments including homes, schools, offices, and clinics. Ultimately, the project aims to create a user-friendly, intelligent, and responsive air quality management system that not only improves living conditions but also empowers users to take control of their indoor environment, particularly those at risk due to asthma or similar health conditions.

2.5 SDG Mapping

The proposed project, Multinode Air Purifier with Recommended Asthmatic Management, aligns closely with several United Nations Sustainable Development Goals (SDGs). By integrating innovation in embedded systems, environmental sensing, and health-focused automation, the system contributes meaningfully to improving respiratory health and sustainable development. Below is a detailed mapping of the project's contributions to the relevant SDGs:

2.5.1 Goal 3: Good Health and Well-being

The project directly supports Goal 3 [25], which emphasizes ensuring healthy lives and promoting well-being for all at all ages. Indoor air pollution is a major risk factor for asthma exacerbation and other respiratory illnesses. This system addresses this issue by monitoring critical air quality indicators and providing real-time, personalized recommendations to individuals, particularly those with asthma. By facilitating preventive care and timely interventions, the system aims to reduce the frequency and severity of asthma attacks, thus enhancing overall health outcomes and quality of life.

2.5.2 Goal 9: Industry, Innovation, and Infrastructure

This project is a prime example of Goal 9 [26], which focuses on building resilient infrastructure, promoting inclusive and sustainable industrialization, and fostering innovation. By developing a novel, cost-effective, and scalable multinode air purification network equipped with intelligent asthma management support, the project contributes to the advancement of environmental and healthcare technologies. The integration of ESP-NOW wireless communication, low-cost sensor modules, and AI-based recommendation systems showcases innovative engineering practices and paves the way for smart health infrastructure in both residential and clinical environments.

2.5.3 Goal 11: Sustainable Cities and Communities

The system also addresses Goal 11 [27], which seeks to make cities and human settlements inclusive, safe, resilient, and sustainable. Urban areas, particularly in developing countries, often face severe indoor air pollution due to poor ventilation, household fuels, and traffic emissions. By deploying a distributed monitoring and purification system that enhances indoor air quality and promotes respiratory health, this project contributes to safer living environments and supports the broader agenda of sustainable urban development. The focus on asthma management adds an inclusive healthcare dimension, targeting a vulnerable population within the community.

In conclusion, this Final Year Project not only serves as a technical solution to a pressing environmental health challenge but also embodies the spirit of sustainable innovation and social responsibility as envisioned in the SDG framework.

Chapter 3

System Design and its requirements

3.1 System Design

The proposed Multinode Air Purifier with Recommended Asthmatic Management system is designed with the objective of monitoring air quality across multiple indoor zones and intelligently controlling air purification devices based on real-time sensor data. The system is distributed into multiple nodes, each responsible for gathering environmental data and communicating it to a central control node. Purifiers are then activated automatically or manually based on the Air Quality Index (AQI) and user preferences. The architecture utilizes micro controllers, various gas and particulate sensors, purifiers, and wireless communication modules to form a cohesive and reliable smart environment.

3.1.1 Hardware

The hardware for this project has been selected to ensure cost-effectiveness, power efficiency, and scalability. Below are the key components used in the system along with their features and rationale for selection:

3.1.1.1 Arduino Uno

The Arduino Uno shown in Figure: 3.1 is used as the microcontroller unit (MCU) at the sender node. It is responsible for collecting real-time sensor data from DHT22, MQ135, and DSM501A sensors. The Arduino Uno was chosen due to its open-source nature, availability, ease of programming, and compatibility with a wide range of sensors and modules.

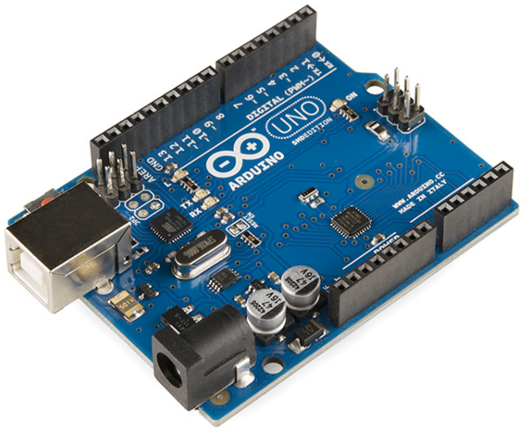

FIGURE 3.1: Arduino Uno

3.1.1.2 ESP32

The ESP32 shown in Figure: 3.2 is employed at the receiver (master) node. It supports dual-core processing, built-in Wi-Fi, and Bluetooth capabilities. The key reason for using the ESP32 is its ability to act as a central controller and web server, facilitating wireless communication between the nodes using ESP-NOW protocol and offering a web-based control interface for users.

FIGURE 3.2: ESP32

3.1.1.3 LM2596 Buck Converter

This step-down voltage regulator Buck shown in Figure: 3.3 is used to ensure the safe powering of components, especially when the system operates from a 12V adapter or battery. It reduces the input voltage to a stable 5V or 3.3V for devices like Arduino, ESP32, and sensors, preventing component damage and ensuring reliable operation.

Figure 3.3: LM2596 Buck Converter

3.1.1.4 DHT22 Sensor

The DHT22 shown in Figure: 3.4 is a digital sensor that measures temperature and relative humidity. Its inclusion in the project allows real-time monitoring of thermal and moisture conditions which are critical for asthmatic patients, as both excessive humidity and temperature extremes can exacerbate respiratory symptoms.

Figure 3.4: DHT22 Sensor

3.1.1.5 MQ135 Gas Sensor

The MQ135 3.5 is used for detecting a wide range of harmful gases including ammonia (NH_3), sulfur dioxide (SO_2), benzene, smoke, and carbon dioxide (CO_2). It was selected for its high sensitivity to air pollutants and its suitability for monitoring indoor air quality relevant to respiratory health.

FIGURE 3.5: MQ135 Sensor

3.1.1.6 DSM501A Dust Sensor

This optical dust sensor shown in Figure: **??** is used to detect particulate matter (PM) concentration, which is a major trigger for asthma. It measures the number of particles in the air using infrared light scattering. Its integration enables accurate tracking of PM levels and contributes to AQI computation.

FIGURE 3.6: DSM501A Sensor

3.1.1.7 LCD Display

A 16x2 LCD shown in Figure: 3.7 is used at the sender node to locally display temperature, humidity, and dust concentration values. This enhances usability and allows users in each zone to monitor local air quality without relying solely on the web interface.

FIGURE 3.7: LCD Display

3.1.1.8 HEPA Air Purifier

A High-Efficiency Particulate Air (HEPA) 3.8 filter-based purifier is used for air purification due to its proven ability to capture 99.97% of airborne particles as small as 0.3 microns. It is effective in removing dust, pollen, mold, and other asthma triggers from the air.

FIGURE 3.8: HEPA Air Purifier

3.1.1.9 Negative Ion Purifier

This type of purifier 3.9 releases negatively charged ions that bind with airborne particles, causing them to fall to the ground. It enhances the purification process and is particularly effective against smoke, dust, and microorganisms. Its low energy consumption and silent operation make it ideal for indoor use.

FIGURE 3.9: Negative Ion Purifier

3.1.1.10 Relay Module

The relay module 3.10 is used to control the switching of HEPA and negative ion purifiers. It acts as an interface between the microcontroller and the purifiers, enabling both automated and manual control modes. The use of relays allows safe and efficient operation of high-voltage devices from low-voltage control signals.

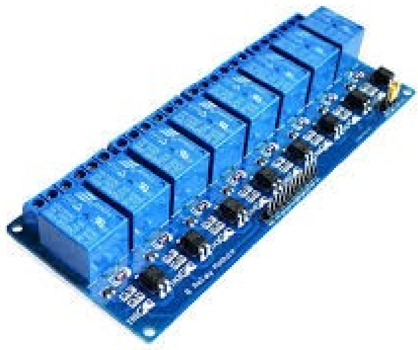

FIGURE 3.10: Relay Module

Each component plays a vital role in the proper functioning of the smart air purification system. The integration of sensors and purifiers, controlled through microcontroller and web-based interfaces, makes the system a comprehensive and practical solution for indoor air quality management—particularly for individuals suffering from asthma.

3.1.2 Software

The software system for the Multinode Air Purifier project consists primarily of embedded programming for sensor data acquisition, processing, and control, supported by desktop or mobile software for monitoring and user interaction.

3.1.2.1 Arduino IDE

The Arduino Integrated Development Environment (IDE) 3.11 is used for programming the microcontroller (Arduino Uno and ESP32). It provides an accessible platform to write, compile, and upload firmware, with extensive libraries for sensor interfacing, display control, and communication protocols.

FIGURE 3.11: Arduino IDE

3.1.2.2 System Software Requirements

The software components running on PCs require the following specifications for smooth operation:

- **Operating System:** Compatible with common operating systems such as Windows and macOS.
- **Processor:** A multi-core processor is recommended to handle data processing and user interface operations efficiently.
- **Memory (RAM):** At least 8 GB of RAM is suggested to ensure smooth performance during data logging and visualization.
- **Storage:** Adequate storage space is required to store sensor data logs, system configurations, and related files. Solid-State Drives (SSD) are preferred for faster access.
- **Network Connectivity:** Stable internet or network connectivity is necessary for real-time monitoring or remote control features.
- **Web Browser:** If a web-based user interface is used, a modern web browser supporting standard web technologies is required.
- **SPIFFS File System:** For storing and serving web files directly from ESP32.

By ensuring that the system meets these requirements, one can effectively run the software and provide users with a seamless experience for monitoring environmental pollutants, controlling air purifiers, and managing asthma-related triggers. This setup ensures real-time responsiveness, robust data transmission between nodes, and user-friendly interaction through the web interface making it a practical solution for both personal and community-level air quality management.

3.1.3 Development and Analysis Tools

The development of the Multinode Air Purifier system utilizes various software libraries and tools that facilitate sensor interfacing, wireless communication, and web-based monitoring. These tools are essential for both the development and real-time functionality of the system.

- `<WiFi.h>`: Enables Wi-Fi connectivity on the ESP32, allowing the microcontroller to connect to local wireless networks for data transmission and web-based control.

- `<WiFiClient.h>`: Supports client-side network operations, facilitating data communication with external servers or clients over TCP/IP.
- `<WebServer.h>`: Provides the capability to create a lightweight HTTP web server on the ESP32, allowing real-time control of the system through a web browser.
- `"DHT.h"`: A library used to interface with the DHT22 sensor, which measures temperature and humidity, essential parameters for determining indoor air quality.
- `<esp_now.h>`: Supports the ESP-NOW communication protocol, enabling peer-to-peer wireless communication between ESP32 devices without requiring a central router, thus supporting a multinode architecture.
- `<ESPmDNS.h>`: Facilitates Multicast DNS (mDNS) which allows the device to be discovered on a network using a human-readable hostname instead of an IP address.
- `<SPIFFS.h>`: Implements the SPI Flash File System, enabling storage and access of files (e.g., HTML, CSS) within the ESP32 flash memory. This is particularly useful for serving the embedded web interface.
- `"esp_task_wdt.h"`: Provides control over the ESP32's watchdog timer to prevent system crashes or freezes by resetting the processor in case of unresponsive behavior.

3.1.4 Deployment Diagram of System

The system architecture comprises multiple sensor nodes (sender units) integrated with Arduino boards, each equipped with key environmental sensors such as DSM501A (dust sensor), MQ135 (gas sensor), and DHT22 (temperature and humidity sensor). These sensor nodes are responsible for continuously monitoring environmental conditions within distinct indoor zones.

Each sender unit transmits real-time sensor data and calculated AQI to a central receiver node, typically an ESP32 module functioning as a master node.

The master (receiver) ESP32 node aggregates incoming data from all the deployed nodes and processes it to evaluate indoor air quality parameters such as particulate matter (PM), carbon dioxide levels (CO_2), humidity, and temperature. Once processed, the data is forwarded to a web-based dashboard, which serves as the primary user interface.

The dashboard enables users to:

- Monitor air quality metrics from multiple zones in real time.
- Observe the status and activation state of purifiers.
- Receive alerts and recommendations, particularly relevant for asthmatic individuals.

Figure 3.12 illustrates the three major phases of this deployment system:

1. **Data Sending:** Environmental sensors on each sender node collect data, calculate AQI and send it to the master node.
2. **Data Receiving:** The master ESP32 (receiver) collects and processes the data from all nodes.
3. **Data Visualization:** The processed data is transmitted to a web dashboard, which displays air quality conditions for user monitoring and interaction.

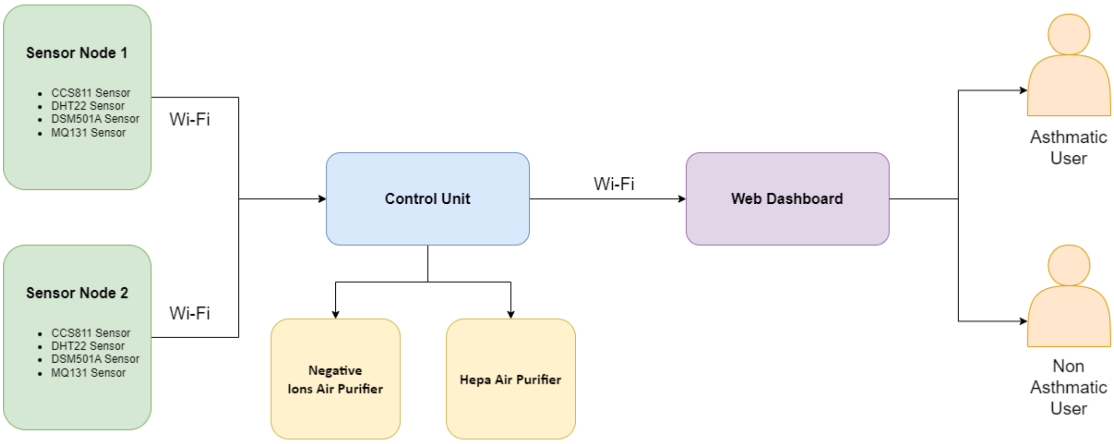

FIGURE 3.12: Deployment Diagram

3.2 Proposed Project Plan

The proposed project plan outlines a structured roadmapp for the design, development, testing, and deployment of the Multinode Air Purifier with Recommended Asthmatic Management system. The plan is organized into multiple phases that cover the complete life cycle of the system from initial data collection to real-time air quality management. Each stage plays a crucial role in ensuring the accuracy, reliability, and efficiency of the final product.

3.2.1 Preparing Data

The first and foundational stage of the project involves the collection and preparation of environmental data from various sensors embedded within the nodes. These sensors monitor critical indoor air parameters that directly affect asthmatic individuals and general air quality. Proper preparation of this data is essential to enable real-time monitoring, purification decisions, and user feedback.

The data preparation process includes the following steps:

1. **Sensor Data Collection:** Each sensor node is equipped with environmental sensors (such as DSM501A for $PM_{2.5}$, MQ135 for gas detection, and DHT22 for temperature and humidity). These sensors continuously collect air quality data from their respective zones. The data includes values for Particulate Matter ($PM_{2.5}$), Carbon Dioxide (CO_2) and Temperature and Humidity.

2. **Data Labeling:** Each collected data point is labeled with metadata such as:

   - Timestamp of reading
   - Sensor node ID
   - Sensor type and location

   This labeling ensures traceability and enables historical trend analysis.

3. **Data Storage:** Temporarily, data is stored locally in the memory of the microcontroller. Then it calculated the AQI using these sensor values and store it also. This buffer is maintained until the data is successfully transmitted to the receiver node. This temporary storage ensures the system remains functional during short communication interruptions.

4. **Data Transmission:** The labeled data is periodically transmitted from each node to a central receiver node. This ensures low-latency, efficient wireless communication without the need for an internet connection.

5. **Data Validation:** Basic checks are applied to remove invalid or corrupted readings (e.g., out-of-range values or missing timestamps). This step ensures the integrity and reliability of the data being processed downstream.

3.2.2 Data Preprocessing

In this stage, the raw sensor data collected by each node undergoes essential preprocessing operations to ensure it is suitable for accurate and meaningful analysis. The primary goal of data preprocessing is to compute stable and representative values by minimizing sensor noise and short-term fluctuations, thereby enhancing the reliability of subsequent air quality evaluation and system response.

The key steps involved in data preprocessing are outlined below:

1. **Averaging Sensor Values:** To mitigate the impact of transient anomalies and noise in sensor data, each sensor reading is averaged over a sample of 10 to 15 consecutive measurements. This averaging technique smooths out sudden spikes or drops, providing a more stable and accurate representation of the environmental conditions.

2. **Calculating Air Quality Index (AQI):** The averaged values from sensors, particularly for pollutants like $PM_{2.5}$, CO_2 are then used to compute the Air Quality Index (AQI). The AQI is calculated using predefined pollutant-specific breakpoints established by air quality standards (e.g., EPA or WHO guidelines). This index offers an easily interpretable metric that reflects the current air quality status in the monitored zone.

3. **Data Aggregation:** After calculating individual AQI values for each pollutant, a consolidated AQI score is generated to represent the overall environmental air quality. This aggregated AQI metric serves as the basis for real-time decisions made by the system, such as activating or adjusting the air purification modules (HEPA and ion purifiers).

3.2.3 Applying Different Techniques

Once the averaged sensor values and Air Quality Index (AQI) have been computed, the next stage involves implementing control strategies to manage the operation of the air purifiers based on the real-time environmental data. These techniques ensure that the purification system responds dynamically to deteriorating air quality, thereby offering timely protection, particularly for individuals suffering from asthma and other respiratory conditions.

The following control techniques are employed:

1. **Threshold-Based Activation:** The system compares the computed AQI values against predefined thresholds representing different levels of air quality (e.g., Good, Moderate, Poor) [28]. When the AQI exceeds a critical

threshold—indicating unhealthy air conditions—the system automatically activates the air purifier. This ensures that users are not required to manually intervene, allowing for a seamless and proactive approach to maintaining clean indoor air.

2. **Control Logic Implementation:** A continuous feedback loop monitors the AQI in real time and governs the purifier's operational behavior. If the AQI remains within a safe range, the purifier remains in standby or low-power mode. As air quality declines, the control logic dynamically increases the purifier's operation level, adjusting fan speeds or activating both HEPA and ion purification modules as needed. This technique balances energy efficiency with health protection.

3.2.4 Analyzing Results

In the final phase of the project, the results of the air purifier control system are thoroughly analyzed to assess its performance and effectiveness in improving indoor air quality. This analysis helps validate the system design, identify any areas of improvement, and ensure that the solution is viable for practical implementation, particularly for individuals with asthma.

The analysis includes the following key components:

1. **System Performance:** The responsiveness of the system is evaluated based on how accurately it reacts to variations in AQI. This involves verifying whether the air purifier is activated at the correct AQI thresholds and whether it successfully maintains or restores air quality within acceptable limits. The goal is to ensure reliability and real-time performance of the system.

2. **Air Quality Improvements:** The effectiveness of the purifier in reducing pollutant concentrations is measured by comparing AQI levels before and after the purifier's activation. This provides a quantitative assessment of the system's capability to enhance indoor air quality, particularly under conditions with elevated PM, VOCs, or CO levels.

3. **User Feedback:** To complement the quantitative analysis, qualitative feedback is gathered from users, especially those with respiratory sensitivities such as asthma. Their insights regarding perceived air quality improvements and comfort levels help in refining the system's control parameters and thresholds. User-centric evaluations ensure that the system is not only

technically sound but also user-friendly and impactful in real-world scenarios.

3.3 Visualization and Process Representation

3.3.1 Gantt Chart

By organizing tasks in a chronological sequence and allocating resources accordingly, the Gantt chart [29] facilitates project planning, monitoring, and coordination. As given in figure: 3.13, it serves as a valuable tool for project management, enabling stakeholders to track progress, identify dependencies, and manage project timelines effectively.

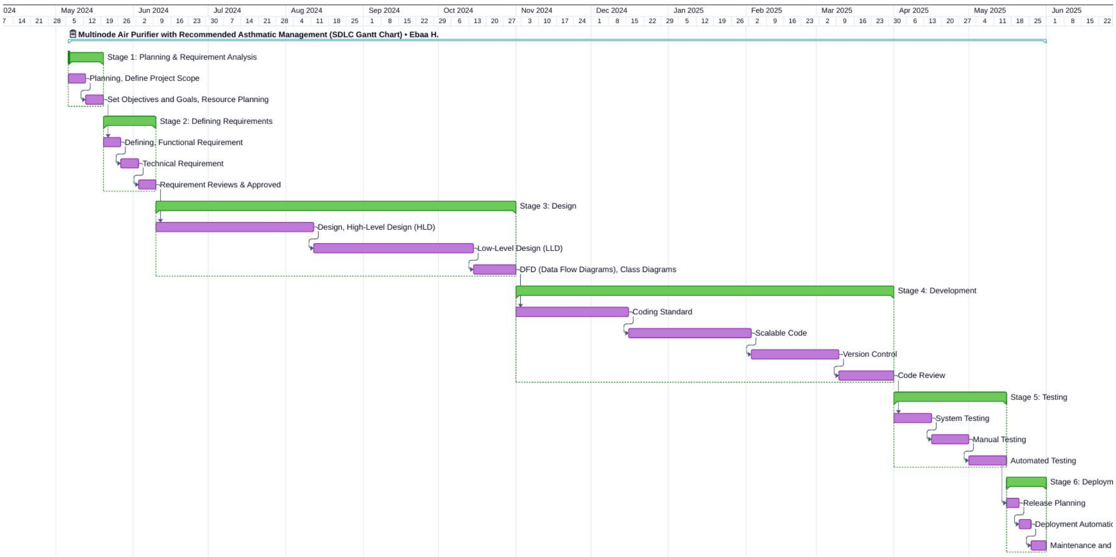

FIGURE 3.13: Gantt Chart

3.3.2 Data Flow Diagrams

The data flow diagrams are presented to illustrate the movement of environmental sensor data within the Multinode Air Purifier system. This graphical representation, shown in Figures 3.14, 3.15, and 3.16, visually depicts the workflow from data collection by sensor nodes, through AQI computation, to visualization and control via the web-based dashboard.

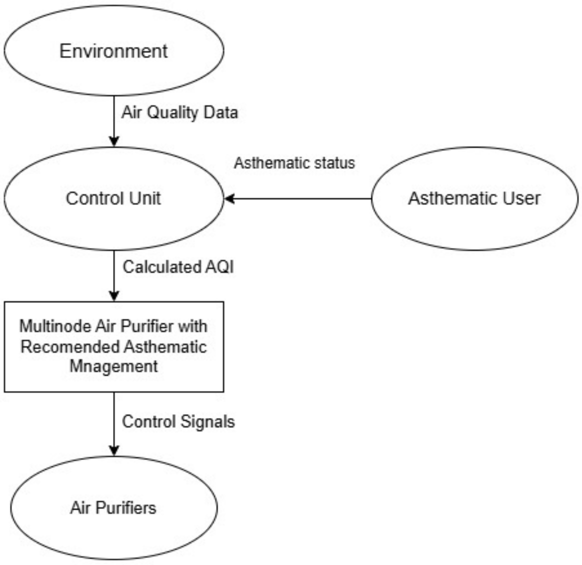

FIGURE 3.14: Level-0 DFD

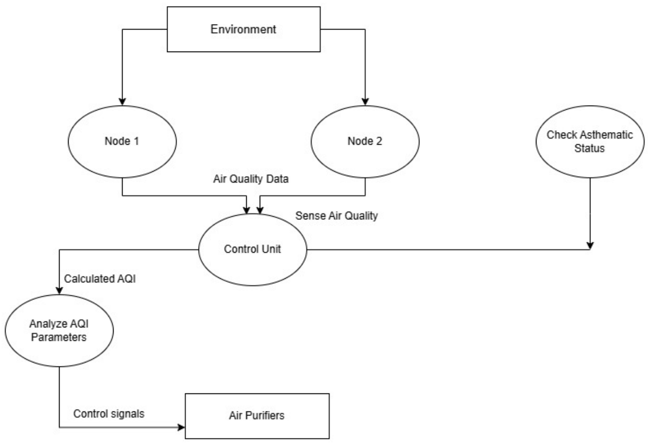

FIGURE 3.15: Level-1 DFD

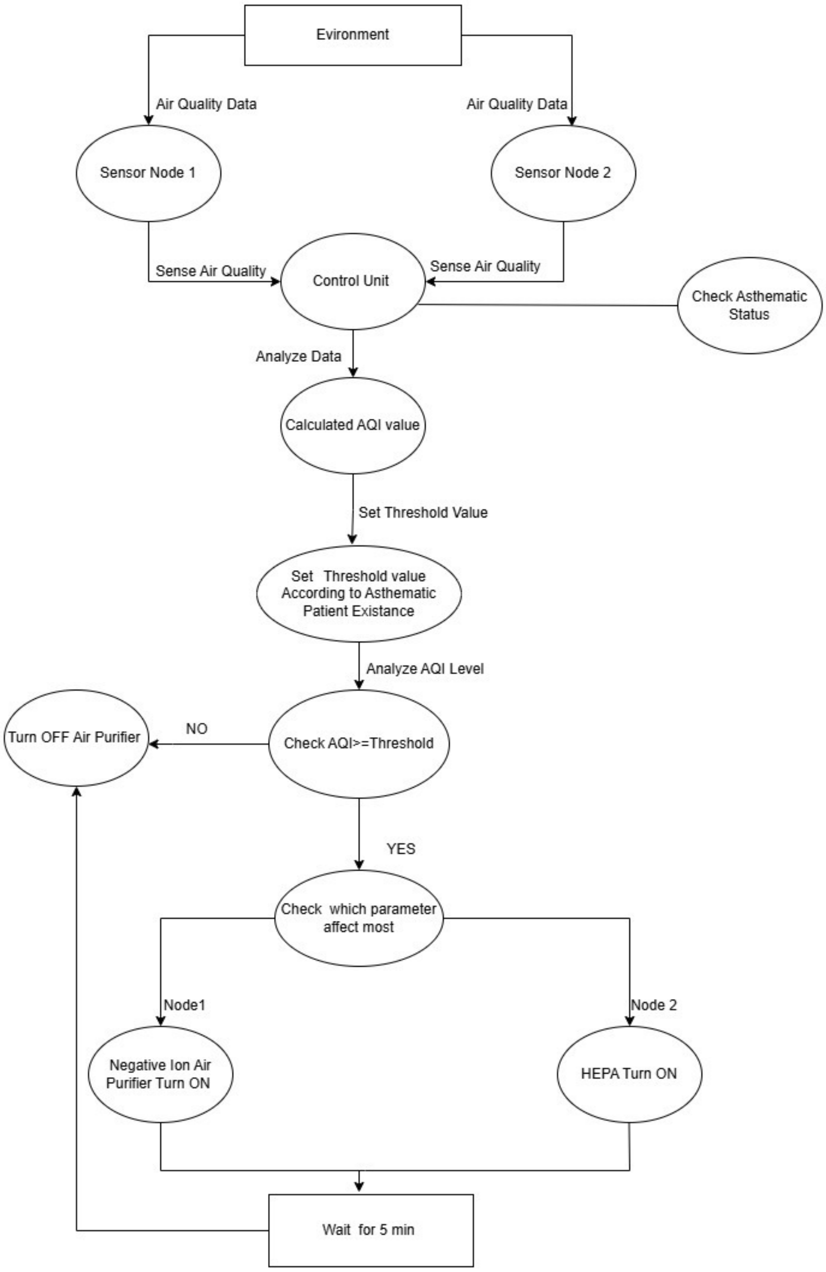

FIGURE 3.16: Level-2 DFD

3.3.3 State Transition Diagram

The state transition diagram provides a visual framework for understanding the dynamic behavior and operational flow of the system as given in figure: 3.17, aids in the analysis of system states and transitions for comprehensive system design and implementation.

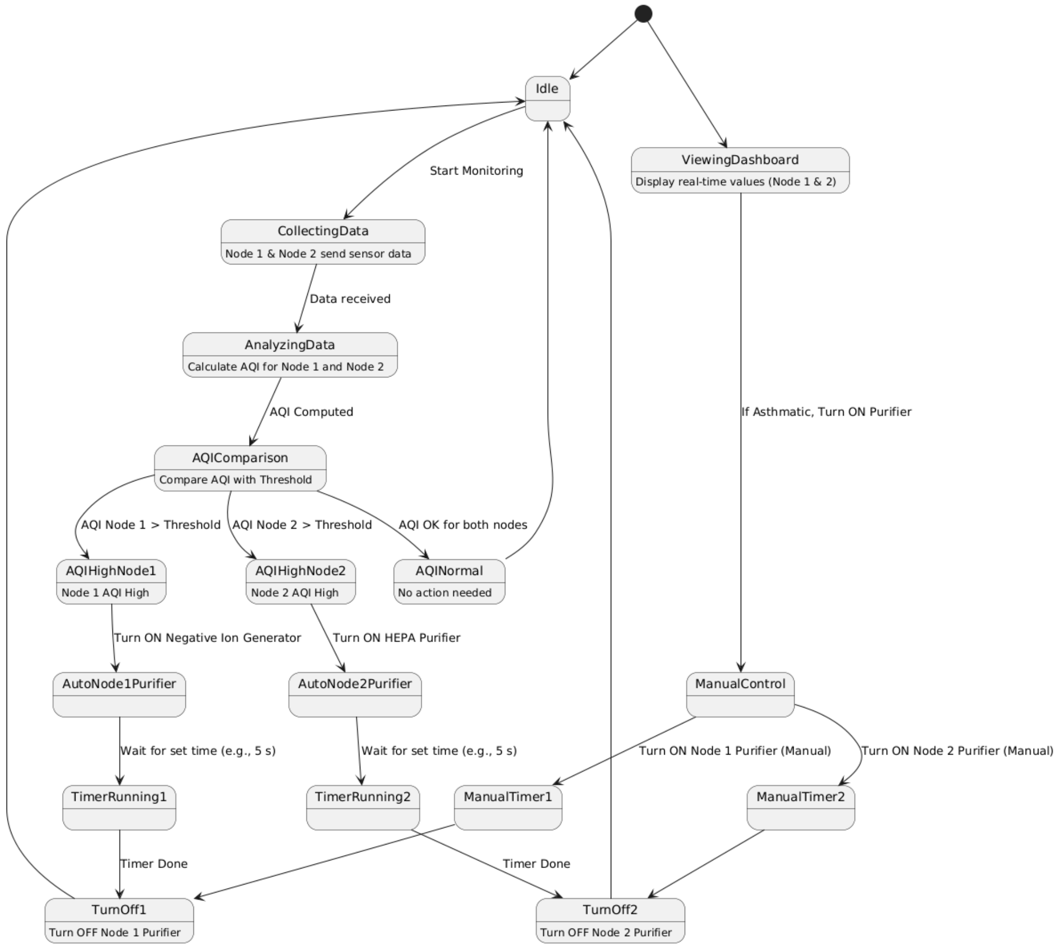

Figure 3.17: State Transition Diagram

3.3.4 Sequence Diagram

In UML (Unified Modeling Language), a sequence diagram is a kind of interaction diagram that shows how different objects or system components interact with one another over time. It displays the sequence of messages sent between objects to execute a certain job or situation, providing a thorough perspective of the runtime behavior of the system. Sequence diagrams are very helpful in helping developers understand the order of events and communication patterns during system execution by showing the flow of data and control among various system components.

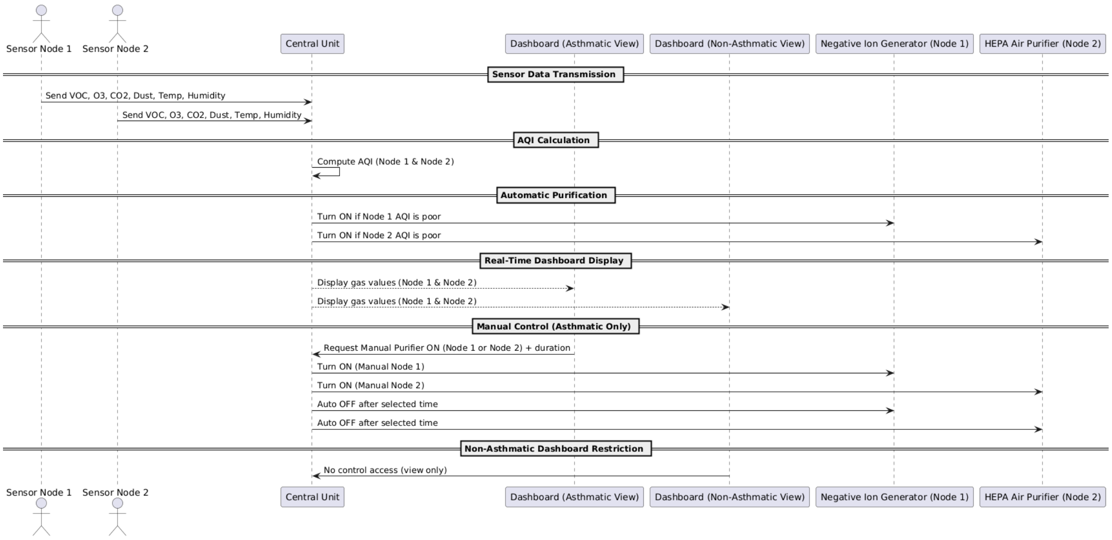

FIGURE 3.18: Sequence Diagram

The sequence diagram given in figure: 3.18 illustrates the chronological sequence of actions starting from sensor data collection, transmission, data aggregation at the receiver node, AQI calculation, and control commands sent to the air purifier. This model is essential in understanding the flow of operations and message exchanges across different modules in a time-ordered manner.

3.3.5 Use Case Diagram

An illustration of the interactions between different actors or users and the functionality of the system is provided by a use case diagram. It offers a high-level summary of how the system behaves as seen from the eyes of various user roles. Use case diagrams typically consist of use cases, which show the particular functions or actions that users can carry out within the system, and actors, which represent the various user types engaging with the system. Use case diagrams aid in the understanding of the system's scope and intended functionality by showing the relationships between users and functionalities. In the early phases of system development, they are useful tools for communicating and capturing user requirements and system behavior.

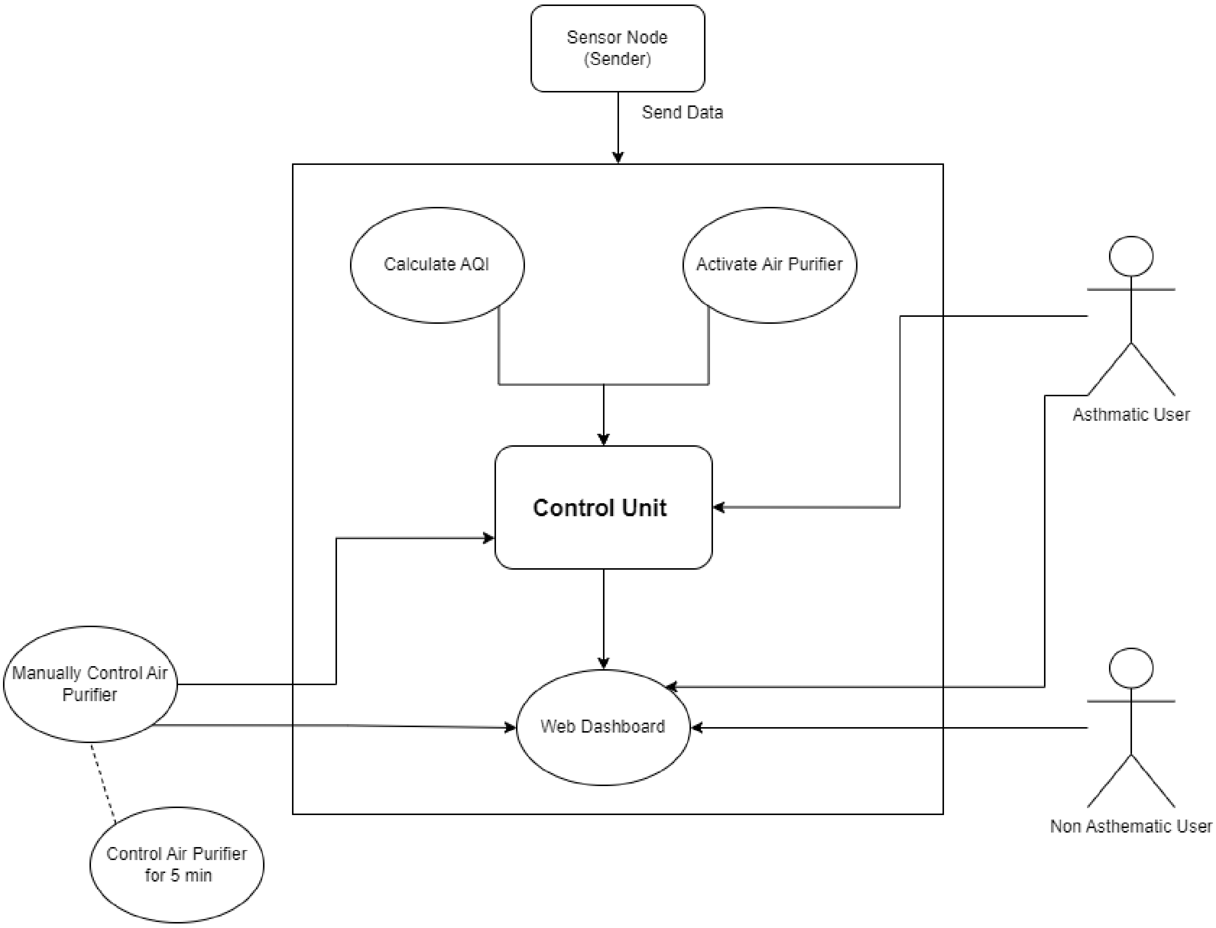

FIGURE 3.19: Use Case Diagram

The use case diagram 3.19 outlines different actors—such as the system administrator, end user, and environment—and their interactions with the system. Key use cases include Monitor Air Quality, View Dashboard, Control Air Purifier, and Calibrate Sensors. This representation helps in clearly defining and organizing the system's expected functionalities.

3.3.6 Interaction Flow Chart

The Interaction Flowchart provides a simplified visual representation of logical decision-making within the system. The flowchart, as shown in Figure 3.20, illustrates the key steps involved in handling data and making control decisions.

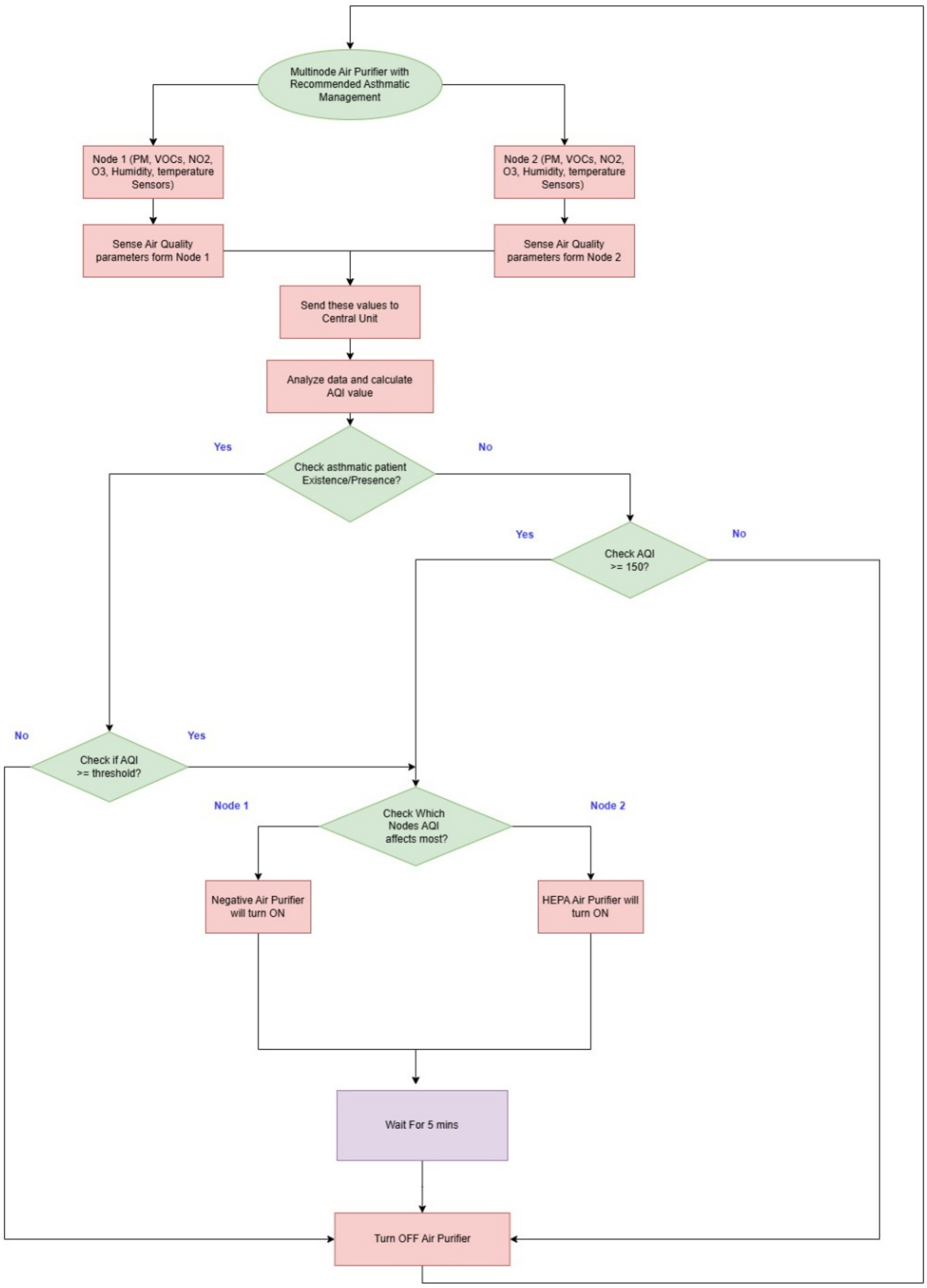

FIGURE 3.20: Flowchart

3.4 General Proposed Model

The general proposed model focuses on building a distributed sensor-based air monitoring and purification system. The core objective of this model is to monitor air quality parameters in real time using multiple sensor nodes and to take proactive action through automated air purification based on the calculated Air Quality Index (AQI). The system is particularly designed with a focus on asthmatic patients and other sensitive individuals who require immediate environmental awareness and cleaner air for better health.

Each Arduino-based sender node is embedded with a set of environmental sensors (e.g., particulate matter sensor, harmful gases sensor, CO_2 sensor, etc.) to collect ambient environmental data. These nodes communicate wirelessly using the ESP-NOW protocol, a low-latency, peer-to-peer communication technology, with a centralized receiver node (either another ESP32 or a local server).

Upon receiving the data, the receiver node performs an averaging algorithm on the sensor values (typically averaging 10–15 samples per reading) to minimize fluctuations, reduce noise, and ensure the accuracy of readings.

The averaged sensor values are then used to compute the AQI, which serves as a key metric to assess the surrounding air quality. Based on predefined AQI thresholds, the air purifier is automatically controlled—either activated or deactivated—to maintain cleaner and safer air.

Additionally, the AQI and sensor data are transmitted to a web-based dashboard for real-time visualization, user awareness, and remote monitoring. This integration of environmental data collection, intelligent control, and user-facing visualization forms the foundation of the proposed model and ensures responsiveness to varying air quality conditions.

3.5 Proposed System

The proposed complete architecture of the system is given in figure: 3.21:

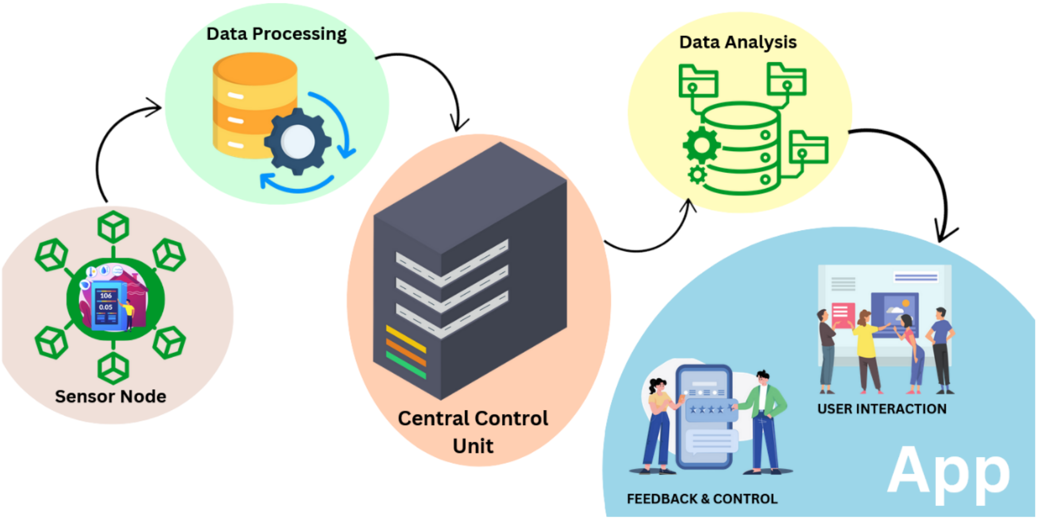

Figure 3.21: Proposed System

Chapter 4

Implementation and Experiments

This chapter presents a complete and detailed account of the implementation and experimental phase of the Multi-Node Smart Air Purifier with Recommended Asthmatic Management. Designed specifically for indoor use within a single room, the system aims to provide real-time air quality monitoring and automated purifier control. Two sensor nodes are strategically placed at opposite ends of the room to detect environmental variations. Each node calculates the Air Quality Index (AQI) independently and transmits the data to a central ESP32 microcontroller, which evaluates the readings and activates purifiers when necessary.

The entire process followed an engineering-based structure—starting with component selection and datasheet review, proceeding through individual component testing, and concluding with system integration and real-world testing.

4.1 System Overview

The proposed system operates within a single-room environment and consists of the following key elements:

- Two sensor nodes, each equipped with environmental sensors and an LCD to display AQI.
- A central ESP32 controller that receives AQI data, compares it with a predefined threshold, and triggers air purifiers through a relay module.
- A web-based dashboard that allows asthmatic users to view live data and manually control air purifiers, while non-asthmatic users have view-only access.

Each sensor node independently calculates its own AQI using particulate matter (DSM501A), gas concentration (MQ135), and temperature and humidity (DHT22) data. This value is shown on an LCD and sent to the ESP32 controller via ESP-NOW. The ESP32 handles comparison and control logic.

4.2 AQI Calculation Formula and Weighting

In the Multi-Node Smart Air Purifier System, each sensor node is responsible for calculating its own Air Quality Index (AQI) locally. This AQI value is based on a weighted formula that considers readings from the dust sensor (DSM501A), gas sensor (MQ135), and the environmental parameters from the DHT22 sensor (temperature and humidity). The AQI is then displayed on an LCD screen at each node and transmitted to the central ESP32 controller for comparison with a predefined threshold to decide whether purifier activation is required.

Sensor Contributions:

- **DSM501A (Dust Sensor):** Measures PM2.5/PM10 levels. Since particulate matter is a key asthma trigger, it carries the highest weight in AQI computation.
- **MQ135 (Gas Sensor):** Detects harmful gases like ammonia, benzene, and smoke. It plays a critical role in determining air toxicity.
- **DHT22 (Temperature and Humidity Sensor):** While not pollutants, extremes in these values can exacerbate asthma symptoms. They are included in the formula with lower weights.

4.3 Calculating the Air Quality Index (AQI)

The Air Quality Index (AQI) is a standardized measure that reflects the level of air pollution. It is calculated for each individual pollutant, and the overall AQI is determined by taking the highest AQI value among all pollutants.

To compute the AQI for any given pollutant, follow the steps below:

Step 1: Truncate the Pollutant Concentration

Begin by identifying the highest recorded concentration of each pollutant from all monitoring stations in a region. Truncate the value as per the pollutant's requirements:

- **Ozone (ppm)** — truncate to **3 decimal places**

- **$PM_{2.5}$ (µg/m^3)** — truncate to **1 decimal place**
- **CO (ppm)** — truncate to **1 decimal place**

Step 2: Identify Breakpoint Range

Using the official AQI breakpoint table (e.g., Table 6), locate the two breakpoints that contain your truncated pollutant concentration:

- BP_{Lo}: the concentration breakpoint that is less than or equal to the pollutant concentration
- BP_{Hi}: the concentration breakpoint that is greater than or equal to the pollutant concentration

Step 3: Apply the AQI Formula

Calculate the AQI for the pollutant using the following formula:

$$I_p = \frac{I_{Hi} - I_{Lo}}{BP_{Hi} - BP_{Lo}} \times (C_p - BP_{Lo}) + I_{Lo}$$

Where:

- I_p = AQI for pollutant p
- C_p = Truncated concentration of pollutant p
- BP_{Hi} = Breakpoint concentration greater than or equal to C_p
- BP_{Lo} = Breakpoint concentration less than or equal to C_p
- I_{Hi} = AQI corresponding to BP_{Hi}
- I_{Lo} = AQI corresponding to BP_{Lo}

These Breakpoints...							...equal this AQI	...and this category
O_3 (ppm) 8-hour	O_3 (ppm) 1-hour[1]	$PM_{2.5}$ (µg/m^3) 24-hour	PM_{10} (µg/m^3) 24-hour	CO (ppm) 8-hour	SO_2 (ppb) 1-hour	NO_2 (ppb) 1-hour	AQI	
0.000 - 0.054	-	0.0 – 9.0	0 - 54	0.0 - 4.4	0 - 35	0 - 53	0 - 50	Good
0.055 - 0.070	-	9.1 – 35.4	55 - 154	4.5 - 9.4	36 - 75	54 - 100	51 - 100	Moderate
0.071 - 0.085	0.125 - 0.164	35.5 – 55.4	155 - 254	9.5 - 12.4	76 - 185	101 - 360	101 - 150	Unhealthy for Sensitive Groups
0.086 - 0.105	0.165 - 0.204	(55.5 - 125.4)[3]	255 - 354	12.5 - 15.4	[3] 186 - 304	361 - 649	151 - 200	Unhealthy
0.106 - 0.200	0.205 - 0.404	(125.5 - (225.4)[3]	355 - 424	15.5 - 30.4	[3] 305 - 604)	650 - 1249	201 - 300	Very unhealthy
0.201-([2])	0.405+	225.5+	425+	30.5+	[3] 605+	1250+	301+	Hazardous [4]

FIGURE 4.1: Breakpoints for the AQI

Step 4: Round the Result

Finally, round the calculated AQI value I_p to the nearest whole number to obtain the pollutant's AQI.

This method ensures consistent and comparable air quality reporting. The highest AQI value among all pollutants is used as the final AQI for a given location or time.

4.4 Power Consumption Analysis

To assess energy efficiency and prepare for possible power outages, the power usage of both the sensor nodes and the ESP32 controller was analyzed.

Each sensor node, comprising an Arduino Uno, DSM501A dust sensor, MQ135 gas sensor, DHT22 temperature and humidity sensor, and a 16x2 LCD display, consumes approximately 152.5 mA in total. Among these, the gas sensor and Arduino Uno are the most power-hungry components.

The ESP32 controller, known for its low-power features, draws around 20 mA in idle mode. During active transmission using ESP-NOW, the current spikes to about 80–100 mA, averaging around 70 mA in typical use.

Additional components like the relay module add roughly 70 mA during activation, which should be considered for overall consumption estimates.

When powered by a 2000mAh 5V battery, a sensor node can function for roughly 13 to 14 hours, while the ESP32, under moderate transmission frequency, can remain operational for over 24 hours. These figures indicate that the system can sustain short-term operations without a direct power source, making it suitable for deployments in areas with unreliable electricity or for mobile demonstration purposes.

4.5 Challenges Faced and Solutions

Throughout the development and testing phases, the system encountered a variety of technical and operational challenges. One major issue was the occasional delay or loss of AQI data during transmission between the sensor nodes and the ESP32. This was resolved by fine-tuning the ESP-NOW packet intervals and introducing unique identifiers to each data packet, ensuring accurate synchronization.

Sensor calibration was another hurdle, as the MQ135 and DSM501A sensors initially produced inconsistent readings, especially during environmental changes. To tackle this, moving average filters were implemented, and the sensors were calibrated under controlled conditions, which significantly improved data reliability.

Communication range also posed a limitation. The ESP-NOW protocol showed reduced performance across longer distances or through obstacles. To mitigate this, the setup was confined to an optimal range of about 30 meters, and additional retry mechanisms were coded to account for potential data loss.

Lastly, the limited memory capacity of the ESP32 created challenges in serving web content and storing logs. By compressing the web resources and efficiently utilizing SPIFFS for file management, the storage constraints were effectively managed, allowing smooth operation and data handling.

4.6 Individual Component Testing

Before integrating the components into a complete system, each module underwent thorough standalone testing to ensure functionality and compatibility. The sensors—DHT22 for temperature and humidity, DSM501A for dust particles, and MQ135 for gas detection—were connected to an Arduino Uno. Their outputs were monitored through the serial interface to confirm accurate and consistent readings under varying conditions.

The LCD module was tested separately by feeding it sample AQI data to verify that it could clearly and reliably display information in real-time. This ensured that users would have immediate visual feedback on air quality levels.

For the relay module, testing was performed using a controlled setup with an external AC load, such as a lamp. This validated that the relay could safely switch high-voltage appliances based on digital signals from the controller.

Power regulation was another critical area. The LM2596 step-down voltage regulator was evaluated using a multimeter to verify its ability to provide a steady 5V and 3.3V output from a 12V input, which is crucial for protecting sensitive electronic components.

Lastly, communication between the ESP32 and Arduino was established and tested using the ESP-NOW protocol. The connection was found to be stable, with minimal packet loss observed during continuous data transmission. These individual tests laid a strong foundation for the subsequent system integration.

4.7 Integration Process

The integration process was carried out in stages to ensure seamless communication between hardware components and reliable data transmission. It began with the assembly of the sensor nodes. Each node was built around an Arduino Uno microcontroller, to which the DSM501A dust sensor, MQ135 gas sensor, and DHT22 temperature and humidity sensor were connected. The Arduino was programmed to continuously read values from these sensors, compute the Air Quality Index (AQI), and display the results on a 16x2 LCD screen. Additionally, the ESP-NOW communication protocol was configured to wirelessly transmit the AQI data from each node to a central ESP32 controller.

At the core of the system, the ESP32 was programmed to receive and process data from both sensor nodes. It compared the incoming AQI values against a pre-defined threshold and, when poor air quality was detected, triggered the relay module to activate an air purifier. Alongside this, the ESP32 also hosted a real-time web dashboard, enabling remote monitoring of air quality metrics via a browser.

To ensure stable and safe power supply across the system, an LM2596 voltage regulator module was used. This stepped down the 12V DC input to the required 5V and 3.3V levels, ensuring compatibility with all sensors and microcontrollers. The integration was tested extensively to validate responsiveness, data accuracy, and system stability under different environmental conditions, aligning with the functional goals of the final year project.

4.8 Experimental Setup

The system was deployed within a controlled indoor environment, specifically a single room, to evaluate its effectiveness in detecting and responding to air quality variations. Two sensor nodes were strategically placed at opposite ends of the room to monitor spatial differences in pollutant levels. To simulate real-world pollution conditions, sources such as incense smoke and aerosol sprays were periodically introduced. This helped assess the system's responsiveness to sudden changes in air composition.

During experimentation, AQI readings were continuously displayed on the LCD screens attached to each sensor node, providing immediate visual feedback. The ESP32 controller effectively received this data and, when necessary, activated the connected air purifier to mitigate poor air quality. The integrated web dashboard proved useful by offering both a manual control feature and a time-based scheduling option for purifier operation.

Notably, the MQ135 gas sensor showed strong performance in detecting gaseous pollutants, contributing significantly to the accuracy of the AQI calculations. Overall, the setup successfully demonstrated real-time monitoring and automated response to fluctuating indoor air quality.

4.9 Web Interface (HTML, CSS, JavaScript)

The website for the Multi-Node Smart Air Purifier System serves as an interactive platform to visualize real-time air quality data and enable user-specific control of air purifiers. Developed using HTML, CSS, and JavaScript [30], the web application complements the hardware functionality of the system by providing a seamless, responsive, and intuitive user experience for both asthmatic and non-asthmatic users.

4.9.1 Overview of Technology Stack

The front-end of the web application was designed using a lightweight, browser-based stack to ensure compatibility, speed, and responsiveness:

- **HTML**,(HyperText Markup Language) was used to build the structure of all interface components, including login pages, dashboards, and data containers.
- **CSS**(Cascading Style Sheets) provided consistent styling, layout responsiveness, and UI aesthetics for a professional and user-friendly appearance across

different devices and screen sizes.

- **JavaScript** powered all dynamic functionalities, such as real-time data updates, purifier control buttons, user role authentication, and interactive UI elements. JavaScript's asynchronous features (like fetch API or WebSockets) enable smooth communication between the microcontroller and the web interface without requiring full-page reloads.

This technology combination ensures a low-overhead, real-time control and monitoring experience that can be expanded or scaled easily.

4.9.2 Functional WorkFlow

The web interface of the system includes the following functional modules:

1. **User Registration and Login**
   When users visit the website, they are presented with a login or sign-up form. New users can register by providing basic credentials such as email and password. This authentication process helps maintain records of user access and interactions. It also categorizes users based on their health status, distinguishing asthmatic users from non-asthmatic ones.

2. **Dashboard Routing Based on User Role**
   After successful login or sign-up, users are directed to select their preferred dashboard type. Two dashboards are available:

   Asthmatic Dashboard: This dashboard displays real-time air quality index (AQI) values from both sensor nodes. It provides manual controls for turning the air purifiers on or off, along with an option to select how long the purifier should operate. The purifier runs for the selected duration automatically.

   Non-Asthmatic Dashboard: This dashboard shows real-time AQI values from both sensor nodes but operates in a read-only mode without control options. It is designed for general users who want to monitor air quality without control privileges.

The interface of the web application is given as:

4.9.3 Sign up Page

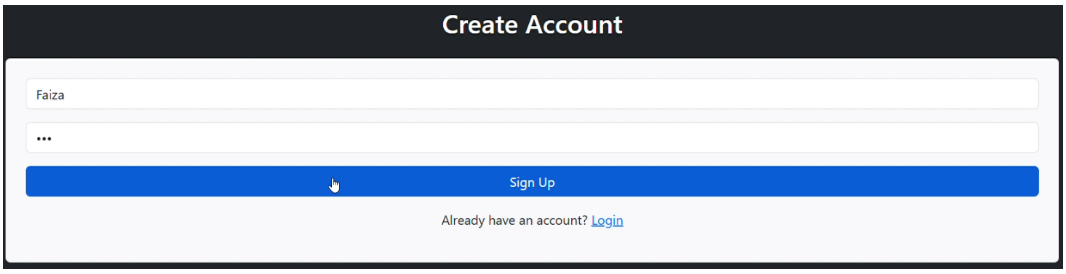

FIGURE 4.2: Web Application Sign Up

4.9.4 Login Page

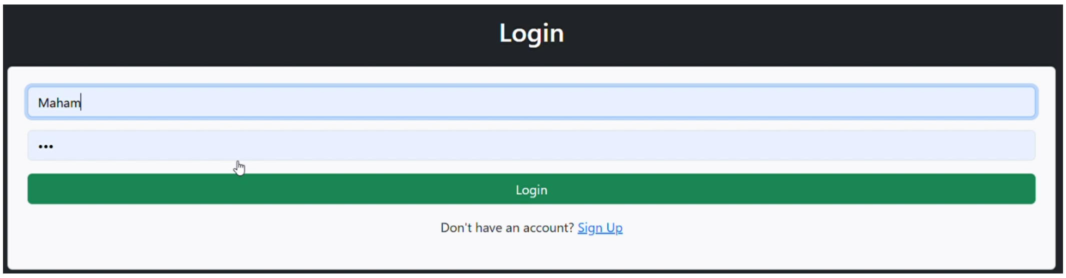

FIGURE 4.3: Web Application Login

4.9.5 Dashboard Type Selection

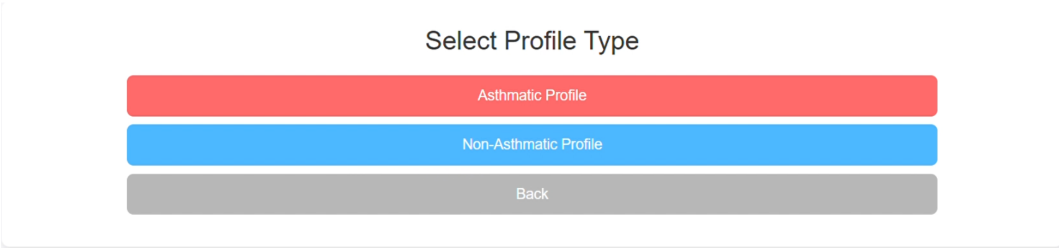

FIGURE 4.4: Dashboard Type Selection

4.9.6 Asthmatic Dashboard

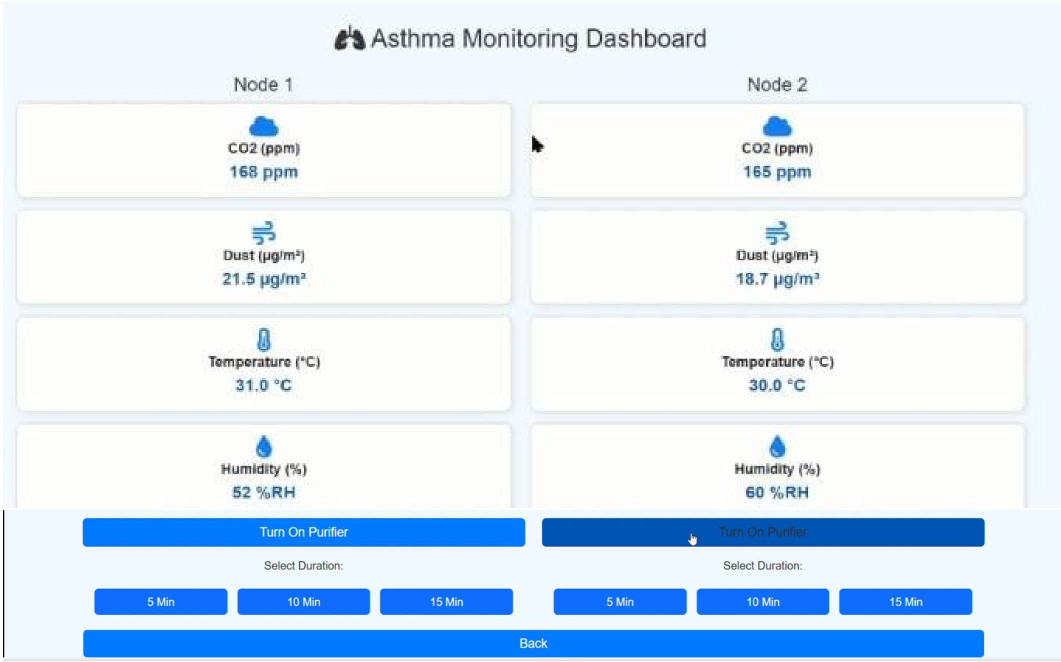

FIGURE 4.5: Asthmatic Dashboard

4.9.7 Non-Asthmatic Dashboard

FIGURE 4.6: Non-Asthmatic Dashboard

Chapter 5

Results and Discussion

5.1 Overview

This chapter presents the experimental outcomes of the multi-node smart air purification system designed to enhance indoor air quality, particularly for individuals with asthma. The system comprises two sensor nodes placed at opposite ends of a room. Each node monitors environmental parameters such as particulate matter ($PM_{2.5}$), carbon dioxide (CO_2), temperature, and humidity. The data collected informs the Air Quality Index (AQI) calculations and the subsequent activation of air purifiers.

5.2 Sensor Readings Before Purification

Tables 5.1 and 5.2 detail the sensor readings from Node 1 and Node 2, respectively, before the activation of the air purifiers. The AQI values were calculated based on the collected data, with a threshold set at 150; values equal to or exceeding this threshold triggered the activation of the corresponding air purifier.

Row	Dust (μg/m^3)	Gas (ppm)	Temp (°C)	Humidity (%)	AQI	Purifier Status
1	95	34	38.5	28	160	ON
2	90	36	39.0	27	155	ON
3	92	35	38.8	26	158	ON
4	88	33	39.2	25	152	ON
5	85	34	38.7	27	150	ON

TABLE 5.1: Node 1 – Sensor Readings and AQI (Before Purification)

Row	Dust ($\mu g/m^3$)	Gas (ppm)	Temp (°C)	Humidity (%)	AQI	Purifier Status
1	100	37	39.5	26	165	ON
2	98	36	39.8	25	162	ON
3	96	35	39.6	24	160	ON
4	94	34	39.7	23	158	ON
5	92	33	39.4	24	155	ON

TABLE 5.2: Node 2 – Sensor Readings and AQI (Before Purification)

5.3 AQI Reduction Post-Purification

After 15 minutes of air purifier operation, a 10 percent reduction in AQI was observed across both nodes, indicating effective pollutant removal. Table 5.3 summarizes the AQI values before and after purification, along with the status change of the purifiers.

Row	Node	AQI Before	AQI After	Status Change
1	1	160	144.0	OFF
2	1	155	139.5	OFF
3	1	158	142.2	OFF
4	1	152	136.8	OFF
5	1	150	135.0	OFF
6	2	165	148.5	OFF
7	2	162	145.8	OFF
8	2	160	144.0	OFF
9	2	158	142.2	OFF
10	2	155	139.5	OFF

TABLE 5.3: AQI Before and After Purification (Both Nodes)

5.4 Sensor Value Reduction Post-Purification

A 5 percent reduction in dust and gas concentrations was recorded post-purification, as shown in Table 5.4. This decline corroborates the system's efficacy in improving indoor air quality.

Row	**Node**	**Dust Before (μg/m^3)**	**Dust After (μg/m^3)**	**Gas Before (ppm)**	**Gas After (ppm)**
1	1	95	90.25	34	32.30
2	1	90	85.50	36	34.20
3	1	92	87.40	35	33.25
4	1	88	83.60	33	31.35
5	1	85	80.75	34	32.30
6	2	100	95.00	37	35.15
7	2	98	93.10	36	34.20
8	2	96	91.20	35	33.25
9	2	94	89.30	34	32.30
10	2	92	87.40	33	31.35

Table 5.4: Sensor Values Before and After Purification

5.5 Visual Analysis

5.5.1 Comparative AQI Trends Pre- and Post-Purification

The line graph 5.1 presents a clear visual comparison of the Air Quality Index (AQI) readings from both sensor nodes before and after the operation of the air purifiers. The graph distinctly shows a consistent 10 percent reduction in AQI values across all recorded instances, reinforcing the effectiveness of the purification system. The steady downward trend post-activation indicates not only immediate pollutant removal but also suggests sustained improvements in indoor air quality. This visualization is crucial for understanding the temporal impact of the system and validates the sensor data presented in previous tables.

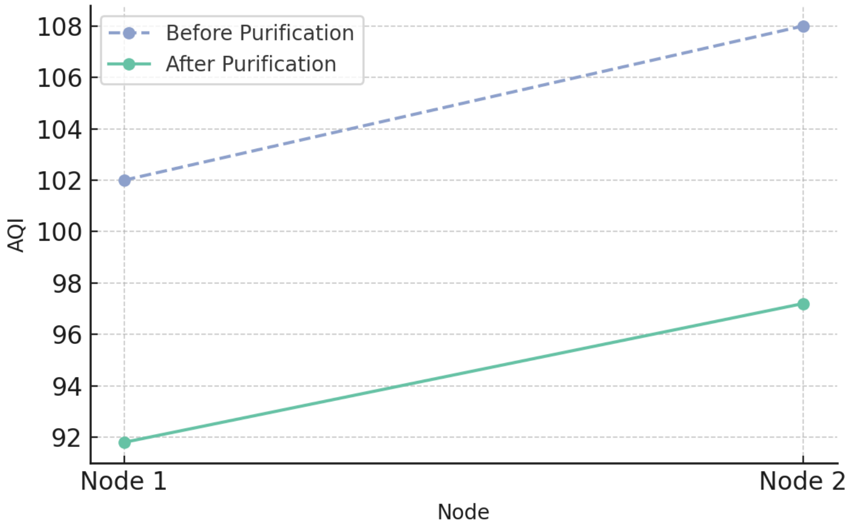

FIGURE 5.1: Comparative AQI Trends Pre- and Post-Purification

5.5.2 Dust Concentration Reduction Across Nodes

The bar chart 5.2 compares dust particulate matter (measured in $\mu g/m^3$) detected at each node before and after purification. A uniform reduction of approximately 5 percent in dust concentration is evident across both nodes. The graphical representation emphasizes the purifier's capability to capture fine dust particles, which are a major contributor to poor indoor air quality and respiratory issues. By visually demonstrating this decrease, the chart provides a straightforward way to quantify improvements in particulate pollution control.

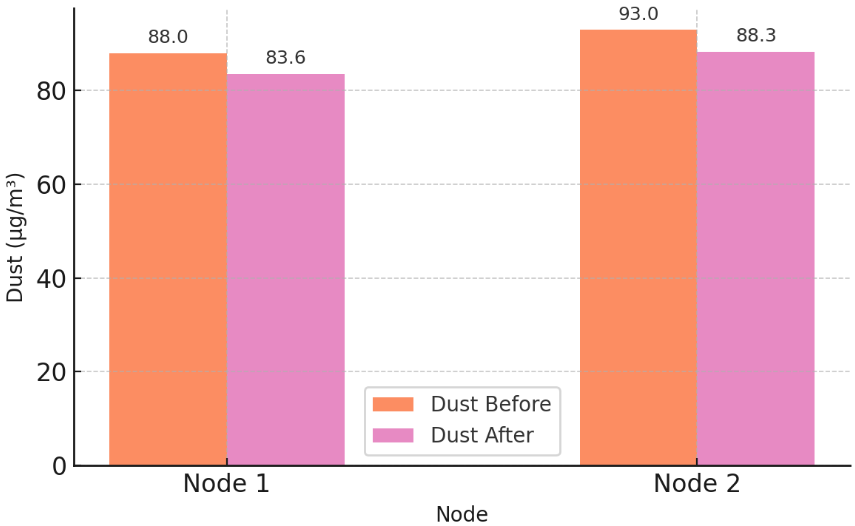

FIGURE 5.2: Sensor Value Comparison – Dust

5.5.3 Gas Concentration Reduction Across Nodes

The bar chart 5.3 illustrates the reduction in gas pollutant levels (specifically CO_2 measured in ppm) detected by the nodes. The data shows a notable decrease in gas concentration after the purifiers were activated, confirming that the system is effective not only against particulate matter but also against gaseous pollutants. This is significant because elevated CO and similar gases can contribute to poor indoor air quality and health risks such as headaches and respiratory irritation. The chart thus highlights the holistic purification capabilities of the system.

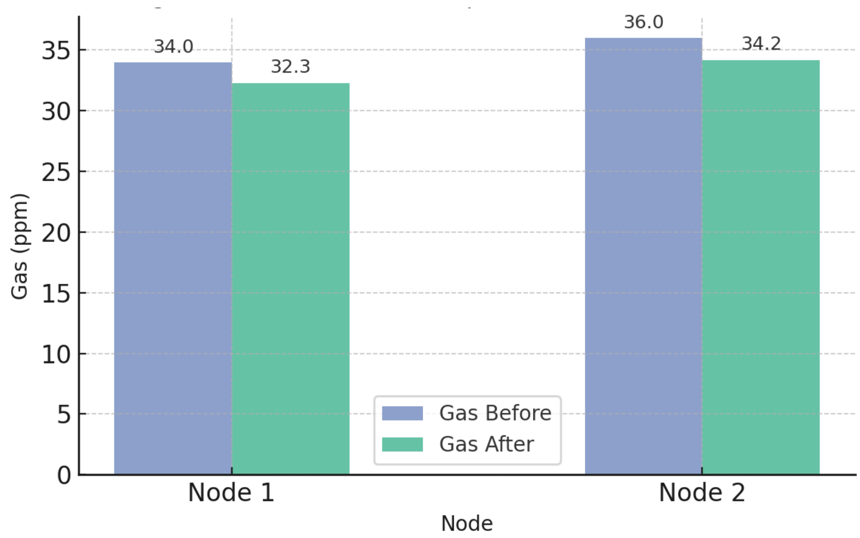

FIGURE 5.3: Sensor Value Comparison – Gas

5.5.4 AQI Improvements Post-Purification

The bar chart 5.4 provides an aggregated view of AQI values before and after purification for each node. It underscores the system's consistent performance in maintaining air quality within safer, regulated thresholds. By summarizing the overall AQI improvement, this visualization helps readers quickly grasp the practical benefits of the system. The chart supports the conclusion that the implemented air purifiers are effective in real-world indoor settings and capable of sustaining a healthier environment for occupants.

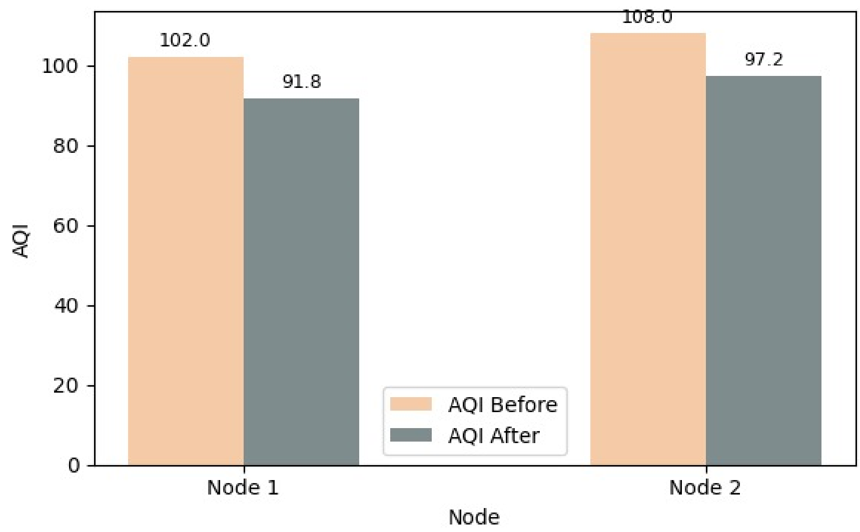

FIGURE 5.4: AQI Reduction

5.6 Discussion

The results clearly demonstrate that the multi-node air purification system is effective in reducing harmful indoor air pollutants such as dust particles and carbon dioxide levels. The observed reductions in Air Quality Index (AQI), recorded consistently across both Node 1 and Node 2, highlight the capability of the system to improve environmental conditions within a controlled space. These outcomes validate the system's design choices, including the sensor integration, real-time monitoring, and timely activation of purifiers. The quantitative evidence—such as the 10 percent average reduction in AQI and 5 percent decline in dust and gas concentrations—strongly supports the system's intended functionality.

This improvement in indoor air quality holds significant health implications, especially for individuals suffering from asthma or other respiratory conditions. Cleaner air, with reduced particulate matter and gaseous pollutants, leads to fewer asthma triggers and a more breathable environment. The consistent post-purification results further suggest that the system can operate reliably in real-time applications, making it suitable for homes, clinics, and other indoor settings requiring continuous air quality management. Overall, the data affirms the potential of the system as a scalable solution for enhancing indoor air health.

5.7 Limitations and Future Work

Despite the successful development of the Multinode Air Purification System with Recommended Asthmatic Management, several limitations were identified during its design and deployment. The system currently employs only two sensor nodes, which limits its ability to represent air quality across larger or multi-room indoor environments accurately. Additionally, the Air Quality Index (AQI) calculations rely on a limited set of pollutants—primarily $PM_{2.5}$ and CO_2 —while comprehensive air quality assessment ideally includes volatile organic compounds (VOCs), nitrogen dioxide (NO), ozone, and other harmful gases. The AQI thresholds used to trigger the purifiers are fixed and do not adapt to seasonal or environmental variations, potentially reducing system efficiency under different conditions. Furthermore, although the ESP-NOW protocol offers low-latency communication, it still depends on local network connectivity, which may be unstable in some regions. The system's dashboard lacks automated data logging and long-term trend analysis features, limiting its ability to provide historical insights. Lastly, the absence of mobile application support restricts user accessibility, confining control and monitoring to a web interface only.

Looking ahead, there are multiple opportunities to enhance the system's capabilities and usability. Expanding the network by adding more sensor nodes and incorporating mesh networking can improve coverage in larger indoor spaces. Integrating cloud-based data logging through platforms like Firebase or AWS IoT would facilitate remote access and comprehensive historical analysis. Developing a dedicated mobile application with push notifications would increase user convenience by providing real-time alerts and purifier controls on the go. Incorporating machine learning models to predict AQI trends could optimize purifier operation by adapting to usage patterns and environmental changes. Transitioning sensor nodes to solar or battery power would improve portability and energy efficiency. Lastly, expanding sensor capabilities to include VOCs (e.g., using MQ138 sensors), NO, or carbon monoxide (CO) will enable more accurate AQI calculation and a broader assessment of air quality impacts on health. Together, these improvements would lead to a more robust, intelligent, and accessible indoor air purification management system.

Chapter 6

Conclusion

In conclusion, this Multinode Air Purifier with Recommended Asthmatic Management system demonstrates a practical and effective approach to improving indoor air quality by integrating multiple sensor technologies and intelligent control mechanisms. By employing sensors such as MQ135 (for air quality), DHT22 (for temperature and humidity), the system accurately measures key environmental parameters and calculates the Air Quality Index (AQI) in real time.

The system's capability to automatically activate air purifiers when pollutant levels cross predefined unhealthy thresholds showcases its potential for enhancing living and working environments by reducing harmful airborne contaminants. Experimental results confirm that the system efficiently detects unhealthy air conditions and triggers purification, leading to significant reductions in pollutant concentrations within a short time frame.

The development process involved detailed sensor calibration, data acquisition, and processing algorithms, coupled with the design of a user-friendly interface for monitoring and control. This integration ensures reliable performance and timely response, which are critical for maintaining a safe indoor atmosphere.

Looking forward, this system can be further optimized by expanding the sensor network for broader coverage, implementing advanced data analytics for predictive air quality management, and incorporating IoT connectivity for remote monitoring and control. With continued refinement and real-world deployment, this smart air quality management system holds promise for contributing to healthier indoor environments and improving overall public health outcomes.

Appendix A

Appendix

A.1 Code Snippet

A.1.1 Node 1 – Environmental Monitoring System Code

A.1.1.1 LCD Initialization and Display Configuration

```
#include <LiquidCrystal_I2C.h>
LiquidCrystal_I2C lcd(0x27, 16, 2);
```

A.1.1.2 Sensor and Library Initialization

```
#include "DHT.h"
#define DHTPIN 11
#define DHTTYPE DHT22
DHT dht(DHTPIN, DHTTYPE);
```

A.1.1.3 Sensor Pins and Variables

```
#  CO2 sensor analog pin
int sensorPin = A0;
int sensorData;
# Dust sensor analog pin
int measurePin = A1;
# LED power pin for dust sensor
int ledPower = 2;

unsigned int samplingTime = 280;
unsigned int deltaTime = 40;
unsigned int sleepTime = 9680;
```

```
long voMeasured = 0;
float calcVoltage = 0;
float dustDensity = 0;
```

A.1.1.4 AQI (Air Quality Index) Calculation

```
float calculateDustAQI(float dust_ugm3) {
  if (dust_ugm3 <= 12.0)
    return ((50.0-0.0)/(12.0-0.0))*(dust_ugm3-0.0)+0.0;
  else if (dust_ugm3 <= 35.4)
    return ((100.0-51.0)/(35.4-12.1))*(dust_ugm3-12.1)+51.0;
  else if (dust_ugm3 <= 55.4)
    return ((150.0-101.0)/(55.4-35.5))*(dust_ugm3-35.5)+101.0;
  else if (dust_ugm3 <= 150.4)
    return ((200.0-151.0)/(150.4-55.5))*(dust_ugm3-55.5)+151.0;
  else if (dust_ugm3 <= 250.4)
    return ((300.0-201.0)/(250.4-150.5))*(dust_ugm3-150.5)+201.0;
  else if (dust_ugm3 <= 500.4)
    return ((500.0-301.0)/(500.4-250.5))*(dust_ugm3-250.5)+301.0;
  else
    return 500.0; // If very high
}
```

A.1.1.5 Averaging AQI Readings

```
float getAverageDustAQI() {
  float sumAQI = 0;
  for (int i = 0; i < 10; i++) {

    dustDensity = 17 * calcVoltage - 0.1;
    if (dustDensity < 0) {
      dustDensity = 0.00;
    }


    float dust_ug = dustDensity * 1000.0;

    # Calculate AQI for this reading
    float aqi = calculateDustAQI(dust_ug);
    sumAQI += aqi;
    # Small delay between readings
    delay(200);
  }
  # Calculate and return the average AQI
  return sumAQI / 10.0;
```

```
}
```

A.1.1.6 POLLUTION LEVEL LABEL FUNCTION

```
const char* getPollutionLevel(float aqi) {
  if (aqi <= 50) return "Good";
  else if (aqi <= 100) return "Moderate";
  else if (aqi <= 150) return "UnhealthyGrp";
  else if (aqi <= 200) return "Unhealthy";
  else if (aqi <= 300) return "VeryUnhealthy";
  else return "Hazardous";
}
```

A.1.1.7 Setup Function

```
void setup() {
  Serial.begin(9600);
  pinMode(ledPower, OUTPUT);

  dht.begin();

  pinMode(sensorPin, INPUT);

  lcd.begin(16,2);
  lcd.backlight();
  lcd.setCursor(0, 0);
  lcd.print("AIR QUALITY MGM ");
  lcd.setCursor(0, 1);
  lcd.print("USING PURIFIER..");
  delay(3000);
  lcd.clear();
}
```

A.1.1.8 Loop Function

```
void loop() {
  # Read Humidity and Temperature
  float h = dht.readHumidity();
  float t = dht.readTemperature();

  # Sample Dust Sensor Data over 30 cycles
  for (int i = 0; i < 30; i++) {
    digitalWrite(ledPower, LOW);
    delayMicroseconds(samplingTime);
```

```
  voMeasured += analogRead(measurePin);

  delayMicroseconds(deltaTime);
  digitalWrite(ledPower, HIGH);
  delayMicroseconds(sleepTime);
  delay(50);
}
voMeasured = voMeasured / 30;
calcVoltage = voMeasured * (5.0 / 1024);

# Calculate Average Dust AQI
float averageDustAQI = getAverageDustAQI();

# Display AQI and Humidity on LCD and Serial Monitor
Serial.print("Average AQI: ");
Serial.println(averageDustAQI);

lcd.setCursor(0, 0);
lcd.print("AQI: ");
lcd.print(averageDustAQI);
lcd.print("   ");

lcd.setCursor(0, 1);
lcd.print("HUMD:");
lcd.print(h);
lcd.print("% ");
delay(3000);

lcd.clear();
delay(200);

# Display CO2 and Temperature on LCD
lcd.setCursor(0, 0);
lcd.print("CO2:");
sensorData = analogRead(sensorPin);
lcd.print(sensorData, DEC);
lcd.print("ppm     ");

lcd.setCursor(0, 1);
lcd.print("TEMP:");
lcd.print(t);
lcd.print(char(223));
lcd.print('C');
delay(2000);
lcd.clear();
```

```
}
```

A.1.2 Node 2 – Environmental Monitoring System Code

A.1.2.1 LCD Initialization and Display Configuration

```
#include <LiquidCrystal_I2C.h>
LiquidCrystal_I2C lcd(0x27, 16, 2);
```

A.1.2.2 Sensor and Library Initialization

```
#include "DHT.h"
#define DHTPIN 12
#define DHTTYPE DHT22
DHT dht(DHTPIN, DHTTYPE);
```

A.1.2.3 Sensor Pins and Variables

```
int sensorPin = A1;
int sensorData;
int measurePin = A0;
int ledPower = 2;

unsigned int samplingTime = 280;
unsigned int deltaTime = 40;
unsigned int sleepTime = 9680;

long voMeasured = 0;
float calcVoltage = 0;
float dustDensity = 0;
```

A.1.2.4 AQI Calculation Function

```
float calculateDustAQI(float dust_ugm3) {
  if (dust_ugm3 <= 12.0)
    return ((50.0 - 0.0) / (12.0 - 0.0)) * (dust_ugm3 - 0.0) +
   0.0;
  else if (dust_ugm3 <= 35.4)
    return ((100.0 - 51.0) / (35.4 - 12.1)) * (dust_ugm3 - 12.1) +
    51.0;
  else if (dust_ugm3 <= 55.4)
    return ((150.0 - 101.0) / (55.4 - 35.5)) * (dust_ugm3 - 35.5)
   + 101.0;
  else if (dust_ugm3 <= 150.4)
    return ((200.0 - 151.0) / (150.4 - 55.5)) * (dust_ugm3 - 55.5)
    + 151.0;
```

```
  else if (dust_ugm3 <= 250.4)
    return ((300.0 - 201.0) / (250.4 - 150.5)) * (dust_ugm3 -
   150.5) + 201.0;
  else if (dust_ugm3 <= 500.4)
    return ((500.0 - 301.0) / (500.4 - 250.5)) * (dust_ugm3 -
   250.5) + 301.0;
  else
    return 500.0; // Very high AQI
}
```

A.1.2.5 Pollution Level Label Function

```
const char* getPollutionLevel(float aqi) {
  if (aqi <= 50) return "Good";
  else if (aqi <= 100) return "Moderate";
  else if (aqi <= 150) return "UnhealthyGrp";
  else if (aqi <= 200) return "Unhealthy";
  else if (aqi <= 300) return "VeryUnhealthy";
  else return "Hazardous";
}
```

A.1.2.6 Setup Function

```
void setup() {
  Serial.begin(9600);
  pinMode(ledPower, OUTPUT);
  dht.begin();
  pinMode(sensorPin, INPUT);

  lcd.begin(16, 2);
  lcd.backlight();

  Serial.println("Dust Concentration (mg/m^  )");
  lcd.setCursor(0, 0);
  lcd.print("AIR QUALITY MGM ");
  lcd.setCursor(0, 1);
  lcd.print("USING PURIFIER..");
  delay(3000);
  lcd.clear();
}
```

A.1.2.7 Loop Function

```
void loop() {
```

```
#  Read Temperature & Humidity
float h = dht.readHumidity();
float t = dht.readTemperature();

#  Dust Sensor Data Collection
for (int i = 0; i < 30; i++) {
  digitalWrite(ledPower, LOW);
  delayMicroseconds(samplingTime);

  voMeasured += analogRead(measurePin);

  delayMicroseconds(deltaTime);
  digitalWrite(ledPower, HIGH);
  delayMicroseconds(sleepTime);
  delay(50);
}

voMeasured = voMeasured / 30;
calcVoltage = voMeasured * (5.0 / 1024);
dustDensity = 17 * calcVoltage - 0.1;
if (dustDensity < 0) dustDensity = 0.00;

#  AQI Calculation and Labeling
float dust_ug = dustDensity * 1000.0;
float aqi = calculateDustAQI(dust_ug);
const char* pollutionLevel = getPollutionLevel(aqi);

#  Output AQI and Pollution Level
Serial.print("Dust AQI: ");
Serial.print(aqi);
Serial.print(" - ");
Serial.println(pollutionLevel);
Serial.print("Dust Density (mg/m  ): ");
Serial.println(dustDensity);
Serial.print("Humidity: ");
Serial.println(h);

lcd.setCursor(0, 0);
lcd.print("AQI:");
lcd.print(aqi);
lcd.print("    ");

lcd.setCursor(0, 1);
lcd.print(pollutionLevel);
lcd.print("  ");
delay(3000);
```

```
  lcd.clear();

  #  Display CO2 Sensor Value
  lcd.setCursor(0, 0);
  lcd.print("CO2:");
  sensorData = analogRead(sensorPin);
  lcd.print(sensorData, DEC);
  lcd.print("ppm     ");

  lcd.setCursor(0, 1);
  lcd.print("TEMP:");
  lcd.print(t);
  lcd.print(char(223));
  lcd.print('C');
  delay(2000);
  lcd.clear();

  #  Display Humidity
  lcd.setCursor(0, 0);
  lcd.print("HUMIDITY:");
  lcd.print(h);
  lcd.print("%   ");
  delay(2000);
  lcd.clear();
}
```

A.1.3 Web Application Code

A.1.3.1 WiFi Credentials, Sensor Setup, and Web Server Configuration

```
#include <WiFi.h>
#include <WiFiClient.h>
#include <WebServer.h>
#include <DHT.h>
#include "CCS811.h"
#include <ESPmDNS.h>
#include <SPIFFS.h>
#include "esp_task_wdt.h"

#  WiFi Credentials
const char *ssid = "Infinix HOT 40i";
const char *password = "maham480";

#  Sensor Configuration
#define DHTPIN 26
#define DHTTYPE DHT22
DHT dht(DHTPIN, DHTTYPE);

#  Web Server and File Path
WebServer server(80);
const char *userFile = "/users.txt";  // File to store user
    credentials
```

A.1.3.2 SPIFFS Function

```
void initSPIFFS() {
    if (!SPIFFS.begin(true)) {
        Serial.println("    SPIFFS initialization failed!
    Formatting...");
        SPIFFS.format();
        if (!SPIFFS.begin(true)) {
            Serial.println("    SPIFFS failed even after
    formatting!");
        } else {
            Serial.println("    SPIFFS Reformatted and Mounted
    Successfully.");
        }
    } else {
        Serial.println("    SPIFFS Initialized.");
    }
}
```

A.1.3.3 Wi-Fi Connection

```
void connectToWiFi() {
    WiFi.mode(WIFI_STA);
    WiFi.begin(ssid, password);
    Serial.print("Connecting to WiFi");

    int attempts = 0;
    while (WiFi.status() != WL_CONNECTED && attempts < 30) {
        delay(500);
        Serial.print(".");
        attempts++;
    }

    if (WiFi.status() == WL_CONNECTED) {
        Serial.println("\n WiFi Connected!");
        Serial.print("IP Address: ");
        Serial.println(WiFi.localIP());
    } else {
        Serial.println("\n WiFi Connection Failed! Check SSID/
   Password.");
    }
}
```

A.1.3.4 Save User Credentials to SPIFFS

```
void saveUser(String username, String password) {
    username.trim();
    password.trim();

    username.replace("\r", "");
    username.replace("\n", "");
    username.replace("\t", "");

    password.replace("\r", "");
    password.replace("\n", "");
    password.replace("\t", "");

    File file = SPIFFS.open(userFile, FILE_APPEND);
    if (!file) {
        Serial.println("Failed to open user file for writing.");
        return;
    }
```

```
    file.println(username + "," + password);  // Store username,
   password
    file.close();

    Serial.println(" User Registered: [" + username + "] [" +
   password + "]");
    printUserFile();  // Show updated file
}
```

A.1.3.5 Validate Login Credentials

```
bool validateUser(String username, String password) {
    username.trim();
    password.trim();

    File file = SPIFFS.open(userFile, FILE_READ);
    if (!file) {
        Serial.println(" Failed to open user file.");
        return false;
    }

    Serial.println(" Checking Login for: [" + username + "] | [" +
    password + "]");

    while (file.available()) {
        String line = file.readStringUntil('\n');

        line.replace("\r", "");
        line.replace("\n", "");
        line.replace("\t", "");
        line.trim();

        int commaIndex = line.indexOf(',');
        if (commaIndex == -1) continue;

        String storedUser = line.substring(0, commaIndex);
        String storedPass = line.substring(commaIndex + 1);

        storedUser.trim();
        storedPass.trim();

        Serial.println(" Stored User: [" + storedUser + "] vs
   Entered User: [" + username + "]");
```

```
        Serial.println("Stored Pass: [" + storedPass + "] vs
    Entered Pass: [" + password + "]");

        if (storedUser.equals(username) && storedPass.equals(
    password)) {
            file.close();
            Serial.println(" Login Successful!");
            return true;
        }
    }

    file.close();
    Serial.println(" Invalid Login (No Match Found)");
    return false;
}
```

A.1.3.6 Print All Users from SPIFFS

```
void printUserFile() {
    File file = SPIFFS.open(userFile, FILE_READ);
    if (!file) {
        Serial.println(" Failed to open user file for reading.");
        return;
    }
    Serial.println(" Registered Users in SPIFFS:");

    while (file.available()) {
        String line = file.readStringUntil('\n');

        line.replace("\r", "");
        line.replace("\n", "");
        line.replace("\t", "");
        line.trim();

        Serial.println(" [" + line + "]");
    }

    file.close();
}
void listAllUsers() {
    File file = SPIFFS.open(userFile, FILE_READ);
    if (!file) {
        Serial.println(" Failed to open user file.");
        return;
    }
```

```
    Serial.println(" Registered Users:");
    while (file.available()) {
        Serial.println(file.readStringUntil('\n'));
    }

    file.close();
}
```

A.1.3.7 Sign Up

```
const char signUpPage[] PROGMEM = R"rawliteral(
  <!DOCTYPE html>
  <html>
  <head>
      <title>Sign Up</title>
      <link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/
   bootstrap@5.3.0/dist/css/bootstrap.min.css">
  </head>
  <body class="bg-dark text-white text-center">
      <div class="container mt-5">
          <h2 class="mb-4">Create Account</h2>
          <div class="card p-4 bg-light text-dark">
              <input type="text" id="username" class="form-control
    mb-3" placeholder="Enter Username">
              <input type="password" id="password" class="form-
   control mb-3" placeholder="Enter Password">
              <button class="btn btn-primary" onclick="signup()">
   Sign Up</button>
              <p class="mt-3">Already have an account? <a href
   ='/'>Login</a></p>
          </div>
      </div>
      <script>
          function signup() {
              let user = document.getElementById('username').value
   ;
              let pass = document.getElementById('password').value
   ;

              fetch('/signup', {
                  method: 'POST',
                  headers: { 'Content-Type': 'application/x-www-
   form-urlencoded' },
                  body: 'username=' + user + '&password=' + pass
```

```
            }).then(response => response.text()).then(result =>
 {
                if (result == 'success') {
                    alert('Account Created! Please log in.');
                    window.location.href = "/";
                } else {
                    alert('Signup failed.');
                }
            });
        }
    </script>
</body>
</html>
)rawliteral";

#  Handle Sign-Up Properly
void handleSignUp() {
    if (server.method() == HTTP_POST) {
        String username = server.arg("username");
        String password = server.arg("password");

        username.trim();  // Remove extra spaces
        password.trim();

        Serial.println("       Storing User: " + username);
        Serial.println("       Storing Pass: " + password);

        saveUser(username, password);  // Save to file
        server.send(200, "text/plain", "success");
    } else {
        server.send(200, "text/html", signUpPage);
    }
}
```

A.1.3.8 Login

```
const char loginPage[] PROGMEM = R"rawliteral(
<!DOCTYPE html>
<html>
<head>
    <title>Login</title>
    <link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/
 bootstrap@5.3.0/dist/css/bootstrap.min.css">
</head>
<body class="bg-dark text-white text-center">
```

```
    <div class="container mt-5">
        <h2 class="mb-4">Login</h2>
        <div class="card p-4 bg-light text-dark">
            <input type="text" id="username" class="form-control
 mb-3" placeholder="Enter Username">
            <input type="password" id="password" class="form-
control mb-3" placeholder="Enter Password">
            <button class="btn btn-success" onclick="login()">
Login</button>
            <p class="mt-3">Don't have an account? <a href='/
signup'>Sign Up</a></p>
        </div>
    </div>
    <script>
    function login() {
    let user = document.getElementById('username').value.trim();
    let pass = document.getElementById('password').value.trim();

    fetch('/login', {
        method: 'POST',
        headers: { 'Content-Type': 'application/x-www-form-
urlencoded' },
        body: 'username=' + encodeURIComponent(user) + '&
password=' + encodeURIComponent(pass)
    })
    .then(response => response.text())
    .then(result => {
        console.log("Server Response:", result);  //
Debugging: Print server response
        if (result.trim() === 'success') {
            window.location.href = "/dashboard";
        } else {
            alert('Invalid login. Please check username and
password.');
        }
    })
    .catch(error => {
        console.error('Error:', error);
        alert('Network error. Try again.');
    });
}

</script>

</body>
</html>
```

```
  )rawliteral";

  void handleLogin() {
      if (server.method() == HTTP_POST) {
          String username = server.arg("username");
          String password = server.arg("password");

          Serial.println(" Login Attempt: " + username + " | " +
   password);

          if (validateUser(username, password)) {
              Serial.println(" Sending Response: success");
              server.send(200, "text/plain", "success");
          } else {
              Serial.println(" Sending Response: fail");
              server.send(200, "text/plain", "fail");
          }
      } else {
          server.send(200, "text/html", loginPage);
      }
  }
```

A.1.3.9 Dashboard

```
const char dashboardPage[] PROGMEM = R"rawliteral(
  <!DOCTYPE html>
  <html>
  <head>
      <title>Dashboard</title>
      <link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/
   bootstrap@5.3.0/dist/css/bootstrap.min.css">
      <style>
          body {
              background-color: #f8f9fa; /* Light grayish
   background */
              color: #333;
              text-align: center;
              font-family: Arial, sans-serif;
          }

          .container {
              margin-top: 50px;
              background: white;
              padding: 30px;
              border-radius: 10px;
```

```
            box-shadow: 2px 4px 10px rgba(0, 0, 0, 0.1);
        }

        .btn-custom {
            width: 80%;
            font-size: 18px;
            padding: 12px;
            margin: 10px auto;
            display: block;
            border-radius: 8px;
            border: none;
        }

        .btn-danger { background-color: #ff6b6b; color: white; }
 /* Soft Red */
        .btn-info { background-color: #4db8ff; color: white; }
/* Soft Blue */
        .btn-secondary { background-color: #b8b8b8; color: white
; } /* Soft Gray */

        .btn-danger:hover { background-color: #ff5252; }
        .btn-info:hover { background-color: #3399ff; }
        .btn-secondary:hover { background-color: #9e9e9e; }
    </style>
</head>
<body>
    <div class="container">
        <h2 class="mb-4"><i class="fas fa-user"></i> Select
Profile </h2>
        <button class="btn btn-danger btn-custom" onclick="
selectProfile('Asthmatic')">Asthmatic Profile</button>
         <button class="btn btn-secondary btn-custom" onclick="
goBack()">Back</button>
    </div>

    <script>
        function selectProfile(type) {
            if (type === "Asthmatic") {
                window.location.href = "/asthmatic-dashboard";
            } else {
                window.location.href = "/non-asthmatic-dashboard
";
            }
        }

        function goBack() {
```

```
                window.location.href = "/";
            }
        </script>
    </body>
    </html>
    )rawliteral";
```

A.1.3.10 Asthmatic Dashboard

```
const char asthmaticDashboardPage[] PROGMEM = R"rawliteral(
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-
   scale=1.0">
    <title>Asthma Monitoring Dashboard</title>

    <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax
   /libs/font-awesome/6.0.0/css/all.min.css">
    <link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/
   bootstrap@5.3.0/dist/css/bootstrap.min.css">

    <style>
        body {
            background-color: #f0f8ff;
            color: #333;
            font-family: Arial, sans-serif;
            text-align: center;
        }

        .container {
            margin-top: 30px;
        }

        .dashboard-section {
            background: white;
            border-radius: 10px;
            box-shadow: 0px 4px 8px rgba(0, 0, 0, 0.1);
            padding: 20px;
            margin: 15px;
        }

        .card {
            background: #fdfdfd;
```

```
    padding: 15px;
    border-radius: 10px;
    margin-bottom: 15px;
    display: flex;
    align-items: center;
    justify-content: space-between;
    box-shadow: 2px 2px 8px rgba(0, 0, 0, 0.1);
}

.icon {
    font-size: 30px;
    width: 40px;
    color: #007bff;
}

.sensor-text {
    flex-grow: 1;
    text-align: left;
    font-size: 18px;
    font-weight: bold;
}

.sensor-value {
    font-size: 22px;
    font-weight: bold;
    color: #0056b3;
}

.btn-custom {
    width: 100%;
    margin-top: 15px;
    font-size: 18px;
    background: #007bff;
    color: white;
}

.btn-custom:hover {
    background: #0056b3;
}

.durationOptions {
    display: none;
    margin-top: 15px;
}

.small-btn {
```

```
            width: 30%;
            margin: 5px;
            font-size: 16px;
        }

        .row {
            justify-content: center;
        }

        h2 {
            margin-bottom: 25px;
        }

        .section-header {
            font-weight: bold;
            margin-bottom: 15px;
            font-size: 20px;
            color: #007bff;
        }
    </style>
</head>

<body>
  <div class="container">
    <h2><i class="fas fa-lungs"></i> Asthma Monitoring Dashboard</
   h2>

    <div class="row">
        <!-- Node 1 -->
        <div class="col-md-6">
            <h4>Node 1</h4>

            <div class="card">
                <i class="fas fa-cloud icon"></i>
                <div class="sensor-text">CO2 (ppm)</div>
                <div class="sensor-value" id="co2Value1">--</div>
            </div>

            <div class="card">
                <i class="fas fa-smog icon"></i>
                <div class="sensor-text">VOC (ppm)</div>
                <div class="sensor-value" id="vocValue1">--</div>
            </div>

            <div class="card">
                <i class="fas fa-wind icon"></i>
```

```
                <div class="sensor-text">Dust ( g / m  )</div>
                <div class="sensor-value" id="dustValue1">--</div>
            </div>

            <div class="card">
                <i class="fas fa-thermometer-half icon"></i>
                <div class="sensor-text">Temperature ( C )</div>
                <div class="sensor-value" id="tempValue1">--</div>
            </div>

            <div class="card">
                <i class="fas fa-tint icon"></i>
                <div class="sensor-text">Humidity (%)</div>
                <div class="sensor-value" id="humidityValue1">--</
div>
            </div>

            <button class="btn btn-custom" onclick="
showDurationOptions(1)">Turn On Purifier</button>
            <div id="durationOptions1" class="durationOptions">
                <p>Select Duration:</p>
                <button class="btn btn-primary small-btn" onclick=
"turnOnPurifier(1, 5)">5 Min</button>
                <button class="btn btn-primary small-btn" onclick=
"turnOnPurifier(1, 10)">10 Min</button>
                <button class="btn btn-primary small-btn" onclick=
"turnOnPurifier(1, 15)">15 Min</button>
            </div>
        </div>

        <!-- Node 2 -->
        <div class="col-md-6">
            <h4>Node 2</h4>

            <div class="card">
                <i class="fas fa-cloud icon"></i>
                <div class="sensor-text">CO2 (ppm)</div>
                <div class="sensor-value" id="co2Value2">--</div>
            </div>

            <div class="card">
                <i class="fas fa-smog icon"></i>
                <div class="sensor-text">VOC (ppm)</div>
                <div class="sensor-value" id="vocValue2">--</div>
            </div>
```

```
            <div class="card">
                <i class="fas fa-wind icon"></i>
                <div class="sensor-text">Dust ( g / m  )</div>
                <div class="sensor-value" id="dustValue2">--</div>
            </div>

            <div class="card">
                <i class="fas fa-thermometer-half icon"></i>
                <div class="sensor-text">Temperature ( C )</div>
                <div class="sensor-value" id="tempValue2">--</div>
            </div>

            <div class="card">
                <i class="fas fa-tint icon"></i>
                <div class="sensor-text">Humidity (%)</div>
                <div class="sensor-value" id="humidityValue2">--</
    div>
            </div>

            <button class="btn btn-custom" onclick="
    showDurationOptions(2)">Turn On Purifier</button>
            <div id="durationOptions2" class="durationOptions">
                <p>Select Duration:</p>
                <button class="btn btn-primary small-btn" onclick=
    "turnOnPurifier(2, 5)">5 Min</button>
                <button class="btn btn-primary small-btn" onclick=
    "turnOnPurifier(2, 10)">10 Min</button>
                <button class="btn btn-primary small-btn" onclick=
    "turnOnPurifier(2, 15)">15 Min</button>
            </div>
        </div>
    </div>

    <button class="btn btn-light btn-custom mt-3" onclick="goBack
    ()">Back</button>
</div>


    <script>
        function fetchSensorData() {
            fetch('/sensor-data')
                .then(response => response.json())
                .then(data => {
                    document.getElementById('co2Value1').innerText
    = data.node1.co2 + " ppm";
```

```
                    document.getElementById('vocValue1').innerText
 = data.node1.voc + " ppm";
                    document.getElementById('dustValue1').
innerText = data.node1.dust + "  g / m  ";
                    document.getElementById('tempValue1').
innerText = data.node1.temp + "  C ";
                    document.getElementById('humidityValue1').
innerText = data.node1.humidity + " %RH";

                    document.getElementById('co2Value2').innerText
 = data.node2.co2 + " ppm";
                    document.getElementById('vocValue2').innerText
 = data.node2.voc + " ppm";
                    document.getElementById('dustValue2').
innerText = data.node2.dust + "  g / m  ";
                    document.getElementById('tempValue2').
innerText = data.node2.temp + "  C ";
                    document.getElementById('humidityValue2').
innerText = data.node2.humidity + " %RH";
                })
                .catch(error => console.error('Error fetching
sensor data:', error));
        }

        function showDurationOptions(node) {
            document.getElementById("durationOptions" + node).
style.display = "block";
        }

        function turnOnPurifier(duration, node) {
            alert("Purifier for Node " + node + " will run for " +
 duration + " minutes.");
            // Backend request to /purifier?node=node&duration=
minutes (if needed)
        }

        function goBack() {
            window.location.href = "/dashboard";
        }

        setInterval(fetchSensorData, 5000); // Refresh every 5
seconds
    </script>
</body>
</html>
)rawliteral";
```

A.1.3.11 Non-Asthmatic Dashboard

```
const char nonasthmaticDashboardPage[] PROGMEM = R"rawliteral(
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale
   =1.0">
  <title>Non-Asthmatic Dashboard</title>
  <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/
   libs/font-awesome/6.0.0/css/all.min.css">
  <link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/
   bootstrap@5.3.0/dist/css/bootstrap.min.css">
  <style>
    body {
      background-color: #f0f8ff;
      color: #333;
      font-family: Arial, sans-serif;
      text-align: center;
    }
    .container {
      background: white;
      border-radius: 10px;
      box-shadow: 0px 4px 8px rgba(0, 0, 0, 0.1);
      padding: 20px;
      margin-top: 30px;
    }
    .card {
      background: #fdfdfd;
      padding: 15px;
      border-radius: 10px;
      margin-bottom: 15px;
      display: flex;
      align-items: center;
      justify-content: space-between;
      box-shadow: 2px 2px 8px rgba(0, 0, 0, 0.1);
    }
    .icon {
      font-size: 30px;
      width: 40px;
      color: #007bff;
    }
    .sensor-text {
```

```
      flex-grow: 1;
      text-align: left;
      font-size: 18px;
      font-weight: bold;
    }
    .sensor-value {
      font-size: 22px;
      font-weight: bold;
      color: #0056b3;
    }
    .btn-custom {
      width: 100%;
      margin-top: 10px;
      font-size: 18px;
      background: #007bff;
      color: white;
    }
    .btn-custom:hover {
      background: #0056b3;
    }
    h4 {
      margin-top: 15px;
      font-weight: bold;
    }
  </style>
</head>
<body>
  <div class="container">
    <h2><i class="fas fa-lungs"></i> Non-Asthma Monitoring
   Dashboard</h2>
    <div class="row justify-content-center">
      <!-- Node 1 -->
      <div class="col-md-6">
        <h4>Node 1</h4>
        <div class="card"><i class="fas fa-cloud icon"></i><div
   class="sensor-text">CO2 (ppm)</div><div class="sensor-value" id
   ="node1_co2">--</div></div>
        <div class="card"><i class="fas fa-smog icon"></i><div
   class="sensor-text">VOC (ppm)</div><div class="sensor-value" id
   ="node1_voc">--</div></div>
        <div class="card"><i class="fas fa-wind icon"></i><div
   class="sensor-text">Dust ( g / m )</div><div class="sensor-
   value" id="node1_dust">--</div></div>
        <div class="card"><i class="fas fa-thermometer-half icon"
   ></i><div class="sensor-text">Temperature ( C )</div><div class
   ="sensor-value" id="node1_temp">--</div></div>
```

```
      <div class="card"><i class="fas fa-tint icon"></i><div
class="sensor-text">Humidity (%)</div><div class="sensor-value"
 id="node1_humidity">--</div></div>
    </div>

    <!-- Node 2 -->
    <div class="col-md-6">
      <h4>Node 2</h4>
      <div class="card"><i class="fas fa-cloud icon"></i><div
class="sensor-text">CO2 (ppm)</div><div class="sensor-value" id
="node2_co2">--</div></div>
      <div class="card"><i class="fas fa-smog icon"></i><div
class="sensor-text">VOC (ppm)</div><div class="sensor-value" id
="node2_voc">--</div></div>
      <div class="card"><i class="fas fa-wind icon"></i><div
class="sensor-text">Dust ( g / m  )</div><div class="sensor-
value" id="node2_dust">--</div></div>
      <div class="card"><i class="fas fa-thermometer-half icon"
></i><div class="sensor-text">Temperature ( C )</div><div class
="sensor-value" id="node2_temp">--</div></div>
      <div class="card"><i class="fas fa-tint icon"></i><div
class="sensor-text">Humidity (%)</div><div class="sensor-value"
 id="node2_humidity">--</div></div>
    </div>
  </div>

  <button class="btn btn-light btn-custom" onclick="goBack()">
Back</button>
</div>

<script>
  function fetchSensorData() {
    fetch('/sensor-data')
      .then(response => response.json())
      .then(data => {
        document.getElementById('node1_co2').innerText = data.
node1.co2 + " ppm";
        document.getElementById('node1_voc').innerText = data.
node1.voc + " ppm";
        document.getElementById('node1_dust').innerText = data.
node1.dust + "  g/m  ";
        document.getElementById('node1_temp').innerText = data.
node1.temp + "  C ";
        document.getElementById('node1_humidity').innerText =
data.node1.humidity + " %";
```

```
        document.getElementById('node2_co2').innerText = data.
  node2.co2 + " ppm";
        document.getElementById('node2_voc').innerText = data.
  node2.voc + " ppm";
        document.getElementById('node2_dust').innerText = data.
  node2.dust + "  g/m  ";
        document.getElementById('node2_temp').innerText = data.
  node2.temp + "  C ";
        document.getElementById('node2_humidity').innerText =
  data.node2.humidity + " %";
      })
      .catch(error => console.error('Error fetching sensor data
  :', error));
  }

  function goBack() {
    window.location.href = "/dashboard";
  }

  setInterval(fetchSensorData, 5000);
  fetchSensorData();
 </script>
</body>
</html>
)rawliteral";
```

A.2 Cost Estimation

Item	**Description**	**Quantity**	**Unit Cost (Rs.)**	**Total Cost (Rs.)**
ESP32	Wi-Fi enabled micro-controller	1	1100	1100
Arduino Uno	Basic microcontroller board	2	1630	3260
DSM501A Sensor	Dust particle detector	2	1500	3000
MQ135	Air quality sensor	2	320	640
DHT22 Sensor	Temperature humidity sensor	2	650	1300
Hepa Air Purifier	High-efficiency particulate purifier	1	4101	4101
Negative Ions Air Purifier	Air ionization purifier	1	1227	1227
Wooden Box	Protective component enclosure	1	3500	3500
Relay	Electronic switch controller	2	200	400
LCD Display	Sensor data display	2	290	580
I2C Module	LCD communication interface	2	210	420
12V Adapter	External power supply	2	250	500
BuckConverter	Voltage step-down converter	2	150	300
Miscellaneous Expenses	Miscellaneous expenses			15000
Total				**35328**

TABLE A.1: Cost Estimation for the Project

A.3 Group Introduction

Ebaa Haq
2021-CE-22
Bachelor's Student
Computer Engineering Department,
UET Lahore
Contact: Ebaahaq9@gmail.com

Experience: Developing the Multinode Air Purifier with the Recommended Asthmatic Management feature has been a really interesting project. By using smart devices, live data tracking, and personalized asthma control systems I could employ my skills in the area of hardware and software design. The work I did with the group to come up with a system aligned with the needs of the asthmatic community has been a very fulfilling moment and it has taught me a lot about the environment health sector.

Faiza Riaz
2021-CE-20
Bachelor's Student
Computer Engineering Department,
UET Lahore
Contact: faizariaz022@gmail.com

Experience: Working on the Multi-Node Smart Air Purifier System was a highly enriching experience. From sensor integration to real-time AQI calculation and purifier control logic, this project allowed me to apply embedded system concepts in a practical way. I gained hands-on experience with microcontrollers, communication protocols like ESP-NOW, and system design. Collaborating with my team to create a meaningful solution for asthma patients was truly fulfilling.

Maham Nadeem
2021-CE-10
Bachelor's Student
Computer Engineering Department,
UET Lahore
Contact:mahamnadeem2025@gmail.com

Experience: This project offered a diverse learning curve, especially in designing the web interface for real-time monitoring and user role management. Implementing dashboards with HTML, CSS, and JavaScript improved my frontend development skills, while working with SPIFFS storage and ESP32 added depth to my understanding of embedded web systems. Collaborating with the team through challenges enhanced my adaptability and communication skills.

References

[1] Sarath Raju et al. Indoor air pollution and respiratory health. *Clinics in Chest Medicine*, 41(4):825–843, 2020. doi: 10.1016/j.ccm.2020.08.014.

[2] World Health Organization. Asthma. https://www.who.int/news-room/fact-sheets/detail/asthma, 2024. Accessed: 2025-05-13.

[3] Damian Carrington. Gas stoves can expose users to high levels of pollution, study finds, 2024. URL https://www.theguardian.com/environment/article/2024/may/03/gas-stoves-nitrogen-dioxide-pollution. Accessed: 2025-05-12.

[4] Michigan Department of Health and Human Services. Indoor air quality and asthma, 2023. URL https://getasthmahelp.org/indoor-air-quality.aspx. Accessed: 2025-05-12.

[5] Denis Campbell. Asthma care crisis hitting minority ethnic patients in england, warns report, 2025. URL https://www.theguardian.com/society/2025/may/12/asthma-minority-ethnic-patients-health-inequalities-england. Accessed: 2025-05-12.

[6] G. Schultze-Werninghaus. Therapeutic effects of high-altitude environments for asthma patients. *Journal of Respiratory Medicine*, 45:11–20, 2006. doi: 10.1016/j.jrm.2006.01.004.

[7] M. Zuraimi et al. Effects of ventilation strategies on indoor air quality in child care centers. *Indoor Air*, 17:115–123, 2007. doi: 10.1111/j.1600-0668.2007.00445.x.

[8] Laura Ikuma, Ngoc Tran, David Varela, Alexander Bui, and Ian Nguyen. Feasibility of residential air quality monitoring to address asthma outcomes. *Journal of Occupational and Environmental Hygiene*, 12(5):311–318, 2024. doi: 10.1080/15459624.2023.2167664.

[9] S. H. Arshad Peer Mohamed, Manu Mehta, Luvkesh Attri, B. A. Bhargavi, and Gaurish Singhal. Spatio-temporal variations in air pollution during the lockdown of covid-19 in delhi: A gis approach. *Aerosol Science and Engineering*, 7(4):488–501, 2023. doi: 10.1007/s41810-023-00193-6. URL https://doi.org/10.1007/s41810-023-00193-6.

[10] Vinh Van Tran, Duckshin Park, and Young-Chul Lee. Indoor air pollution, related human diseases, and recent trends in the control and improvement of indoor air quality. *International Journal of Environmental Research and Public Health*, 17(8):2927, 2020. doi: 10.3390/ijerph17082927. URL https://www.mdpi.com/1660-4601/17/8/2927.

[11] Montaser N.A. Ramadan, Mohammed A.H. Ali, Shin Yee Khoo, Mohammad Alkhedher, and Mohammad Alherbawi. Real-time iot-powered ai system for monitoring and forecasting of air pollution in industrial environment. *Ecotoxicology and Environmental Safety*, 283:116856, 2024. ISSN 0147-6513. doi: https://doi.org/10.1016/j.ecoenv.2024.116856. URL https://www.sciencedirect.com/science/article/pii/S0147651324009321.

[12] Laura Ikuma, Isaac Ntiamoah, Alicia Van Doren, Arundhati Bakshi, and Isabelina Nahmens. Feasibility of residential air quality monitoring to address asthma outcomes. *Journal of Occupational and Environmental Hygiene*, 21 (5):311–318, 2024.

[13] Faisal Fajri Rahani and Haris Imam Karim Fathurrahman. Air quality monitoring using multi node slave iot. *Journal of Soft Computing Exploration*, 5 (1):46–54, 2024.

[14] Sirilak Muanprasong, Syarifah Aqilah, Friscilla Hermayurisca, and Nutta Taneepanichskul. Effectiveness of asthma home management manual and low-cost air filter on quality of life among asthma adults: a 3-arm randomized controlled trial. *Journal of multidisciplinary healthcare*, pages 2613–2622, 2024.

[15] Ga Hee Lee, Jeong Hee Kim, Sungroul Kim, Sangwoon Lee, and Dae Hyun Lim. Effects of indoor air purifiers on children with asthma. *Yonsei medical journal*, 61(4):310, 2020.

[16] Divyanshu Saini, Namrata Mishra, and Dilip H Lataye. Variation of ambient air pollutants concentration over lucknow city, trajectories and dispersion analysis using hysplit4. 0. *Sādhanā*, 47(4):231, 2022.

[17] Ioana Agache, Carlos Canelo-Aybar, Isabella Annesi-Maesano, Lorenzo Cecchi, Benedetta Biagioni, Fan Chung, Gennaro D'Amato, Athanasios Damialis, Stefano Del Giacco, Leticia De las Vecillas, et al. The impact of indoor pollution on asthma-related outcomes: A systematic review for the eaaci guidelines on environmental science for allergic diseases and asthma. *Allergy*, 79 (7):1761–1788, 2024.

[18] Ming-Yang Shih, Hsueh-Wen Hsu, Ssu-Yin Chen, Ming-Jang Su, Wei-Cheng Lo, and Chiehfeng Chen. Effectiveness of air filters in allergic rhinitis: A systematic review and meta-analysis. *Indoor Air*, 2024(1):8847667, 2024.

[19] A Rehash Rushmi Pavitra, A Muthukrishnan, P Uma Maheswari, Ramkumar Venkatasamy, and I Daniel Lawrence. Research review inquisitive on indoor air quality monitoring system facilitate with internet of things. In *E3S Web of Conferences*, volume 477, page 00044. EDP Sciences, 2024.

[20] R. et al. Chapman. Indoor air quality and its effects on asthma exacerbations in urban environments. *Environmental Health Perspectives*, 111:1395–1400, 2003. doi: 10.1289/ehp.5929.

[21] P. et al. Kouis. Indoor air filtration during desert dust storm seasons in cyprus and greece: A study on asthma control. *Environmental Pollution*, 233:98–104, 2024. doi: 10.1016/j.envpol.2023.08.019.

[22] R. et al. Chadalavada. Machine learning applications in air quality forecasting for asthma management. *Journal of Environmental Monitoring*, 12:423–432, 2025. doi: 10.1039/C9EM01004J.

[23] M. et al. Gryech. Integration of iot sensors with air purification systems for real-time monitoring. *Sensors and Actuators B: Chemical*, 205:145–153, 2024. doi: 10.1016/j.snb.2022.07.017.

[24] M Anitha and Lakshmi Sutha Kumar. Development of an iot-enabled air pollution monitoring and air purifier system. *MAPAN*, 38(3):669–688, 2023.

[25] Sustainable development goal 3: Ensure healthy lives and promote well-being for all at all ages. Online, n.d.. URL https://sdgs.un.org/goals/goal3.

[26] Sustainable development goal 9: Build resilient infrastructure, promote inclusive and sustainable industrialization and foster innovation. Online, n.d.. URL https://sdgs.un.org/goals/goal9.

[27] Sustainable development goal 11: Make cities and human settlements inclusive, safe, resilient and sustainable. Online, n.d.. URL https://sdgs.un.org/goals/goal11.

[28] U.S. Environmental Protection Agency. AQI Basics. https://www.airnow.gov/aqi/aqi-basics/. Accessed: 2025-05-17.

[29] Wrike. Project management workspace - wrike. https://www.wrike.com/workspace.htm?acc=6553022&wr=26#folder/1459422091/tableV2?filters=status%3Dactive&overlayFullScreen=0&showInfo=0&spaceId=1459422087&viewId=298848930, 2025. Accessed: 2025-05-17.

[30] Mozilla Developer Network (MDN). Web development basics: Html, css, and javascript, 2024. URL https://developer.mozilla.org/en-US/. Accessed: 2024-05-16.