



Department of Digital Business

Journal of Artificial Intelligence and Digital Business (RIGGS)

Homepage: <https://journal.ilmudata.co.id/index.php/RIGGS>

Vol. 5 No. 2 (2026) pp: 15044-15062

P-ISSN: 2963-9298, e-ISSN: 2963-914X

Implementation of MQTT Broker and Gemini API in an Internet of Things Based Indoor Air Pollution Monitoring System

I Putu Tude Rama Prasatya, Ketut Agus Seputra, Kadek Yota Ernanda Aryanto

Department of Informatics, Universitas Pendidikan Ganesha

tude@student.undiksha.ac.id, agus.seputra@undiksha.ac.id, yota.ernanda@undiksha.ac.id

Abstract

Indoor air pollution from pollutants such as Carbon Monoxide (CO) and Particulate Matter (PM_{2.5}) poses significant health risks to room occupants, particularly in enclosed spaces with poor ventilation where pollutants can accumulate to hazardous concentrations. To address this challenge, this study designed and implemented an IoT-based indoor air quality monitoring system integrating an MQTT broker for real-time data transmission and the Gemini API for intelligent data interpretation. The system adopts a three-layer architecture spanning hardware, backend, and application layers. The hardware layer utilizes an ESP32 as a wireless gateway and an Arduino Nano for sensor acquisition, employing the Sharp GP2Y1010AU0F sensor for particulate matter and the MQ-7 sensor for carbon monoxide. The backend, built with Laravel, manages data through a dual-database approach, where MySQL handles structured user data and InfluxDB stores continuous sensor readings. A Flutter mobile application, built with the MVVM pattern, serves as the user interface, delivering real-time air quality information. The Gemini API further enhances the system by automatically generating air quality classifications and actionable health recommendations. Calibration testing demonstrated an average error of 6.45% for the MQ-7 sensor and a notably low 0.44% error for the Sharp GP2Y1010AU0F sensor, indicating high measurement accuracy. A 24-hour continuous stress test revealed a system uptime of 83.3%, confirming reasonable operational reliability. Finally, a usability evaluation using the SUS method involving 30 respondents yielded an average score of 70.67, placing the system in Grade B with a "Good" interpretation, confirming the system is functional and easy to use.

Keywords: Internet of Things; MQTT; Gemini API; Indoor Air Pollution; Carbon Monoxide; Particulate Matter 2.5;

1. Introduction

Air quality is a critical factor that directly affects human health and environmental comfort, particularly in urban areas [1]. Air pollution arises from both natural processes and anthropogenic activities, defined as the introduction of harmful substances into the environment that pose risks to humans and other living organisms. The impacts of air pollution extend beyond climate change, contributing to a wide range of short and long-term health complications [2].

Indoor environmental conditions are equally concerning, as individuals spend approximately 90% of their time indoors, whether at home or in the workplace. According to the World Health Organization (WHO), indoor air pollution (IAP) is responsible for approximately 3.8 million deaths annually [3]. Primary sources of IAP include cooking activities, tobacco smoking, electronic device emissions, consumer products, and building material off gassing. Common pollutants include Particulate Matter PM_{2.5} and Carbon Monoxide (CO), which are associated with respiratory disorders, cardiovascular disease, pregnancy complications, and chronic conditions such as Chronic Obstructive Pulmonary Disease (COPD) and lung cancer. Vulnerable populations, including children and the elderly, face disproportionately higher risks from prolonged exposure [4].

In response to these challenges, the rapid advancement of information technology has introduced the Internet of Things (IoT) paradigm, which integrates physical devices with the internet through hardware and software enhancements, enabling machine-to-machine communication and transforming conventional systems into smarter, more efficient solutions [5]. The application of IoT has expanded widely; for example, it is actively used to monitor household electrical power consumption to encourage energy conservation [6]. Similarly, IoT facilitates the remote control of household electrical appliances, allowing users to efficiently manage devices over the internet via

microcomputers [7]. This energy efficiency can be further optimized by implementing IoT-based smart contact boxes that specifically track the power usage of electronic devices within a smart home [8]. Beyond smart homes, IoT is highly beneficial in agriculture, where it is used to monitor soil moisture and light intensity to prevent the spread of *Fusarium oxysporum* in vanilla plants [9]. It has also been effectively deployed in solar hydroponic systems to automatically monitor essential water quality parameters like pH and Total Dissolved Solids (TDS) [10]. In the health and sports domain, IoT technology enables the creation of prototypes capable of monitoring an athlete's heartbeat and body temperature in real time [11].

Communication protocols play a fundamental role within the IoT ecosystem. Based on the 2023 Eclipse IoT & Edge Developer Survey, Message Queue Telemetry Transport (MQTT) and Hypertext Transfer Protocol (HTTP) are the two most widely adopted protocols in IoT deployments, with MQTT demonstrating higher adoption rates. MQTT operates on a publish-subscribe architecture designed to minimize network bandwidth consumption and optimize device resource utilization, ensuring reliable data transmission. In contrast, HTTP follows a client-server model that, while prevalent in web communications, introduces significant network overhead due to large packet headers, which can degrade performance in resource-constrained IoT environments. Despite these advancements in connectivity, interconnecting networks in IoT remain vulnerable to network-based disruptions and attacks, such as Denial of Service (DoS), necessitating the use of classification algorithms like Naive Bayes for effective intrusion detection [12].

As the volume of data generated by IoT devices continues to grow, the need for intelligent data analysis has become increasingly important. Artificial Intelligence (AI) addresses this need by enabling pattern recognition, predictive analysis, and autonomous decision-making based on available data. The integration of AI with IoT gives rise to the Artificial Intelligence of Things (AIoT) paradigm, in which devices not only interconnect but also adapt intelligently to their environment, enhancing communication, analysis, and decision-making capabilities within IoT systems [13].

Based on the aforementioned challenges and technological advancements, this study proposes an IoT-based air quality monitoring system utilizing the MQTT protocol to facilitate real-time monitoring of CO and PM2.5 levels via a mobile application. The MQTT protocol enables efficient data transmission, supports effective remote monitoring, and optimizes bandwidth and power consumption. Furthermore, the system is integrated with the Gemini API as an AIoT component, enabling intelligent interpretation of air quality data and providing actionable recommendations to users based on real-time sensor readings

2. Research Method

2.1. Research Process



Gambar 1. Research Process

As illustrated in Figure 1, the research methodology follows six interconnected stages. The first stage, problem identification, establishes that prolonged exposure to indoor air pollutants such as CO and PM2.5 poses significant health risks, ranging from respiratory irritation to chronic conditions including asthma and pulmonary disease. The design stage then translates these findings into a comprehensive system architecture covering hardware selection, sensor parameter determination, and software interface planning. The development stage materializes the design through hardware assembly, microcontroller firmware programming, and integration of a real-time dashboard with threshold-based notifications.

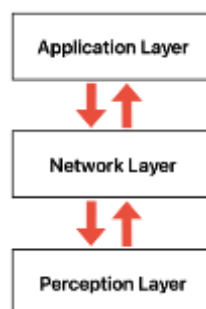
System evaluation was carried out in the testing stage using two complementary methods: black-box testing to verify functional correctness, and the System Usability Scale (SUS) to quantitatively assess user acceptability. Both methods confirmed that the system operates without significant functional errors and achieves an acceptable usability score. Finally, the expert validation stage subjected the system to evaluation by domain experts in air quality, environmental health, and embedded systems engineering, whose assessments regarding sensor accuracy, parameter relevance, and system reliability served as authoritative confirmation of the system's academic and practical feasibility.

2.2. Indoor Air Pollution

Indoor air contains several hazardous pollutants, including PM2.5, CO, NO₂, and radon, all of which pose significant health risks with prolonged exposure [14]. PM2.5 and CO primarily originate from indoor activities such as cooking, smoking, and the use of heating appliances, while PM2.5 can also come from outdoor environments and domestic activities like burning candles and using fireplaces. Exposure to PM2.5 is linked to respiratory disorders, decreased lung function, and increased risk of premature death in individuals with heart and lung disease, whereas CO exposure disrupts the blood's oxygen-carrying capacity, leading to fatigue, chest pain, and impaired brain function [15]. To address these risks, exposure thresholds have been established: PM2.5 is limited to 35 µg/m³ based on Indonesian health regulations [16], while the safe CO limit is set at 9 ppm according to ANSI/ASHRAE standards [17].

2.3. Internet of Things

The Internet of Things (IoT) is a rapidly evolving concept that describes a network of interconnected devices capable of automatically collecting, processing, and sharing data via the internet without direct human intervention. Within the IoT ecosystem, various architectural models have been proposed to define the structure and interaction between components. In this study, a three-layer architecture is adopted as illustrated in figure 2.



Gambar 2. Three-layer architecture

The perception layer is responsible for collecting data through sensors installed on physical devices. The network layer connects these devices to a communication network so that data can be forwarded to the processing system. Meanwhile, the application layer provides specific services to users based on the results of data processing from the previous layer.

Sensors play a critical role in the IoT ecosystem as devices that detect changes in physical or environmental conditions and convert them into electronic signals for further analysis. In this study, two sensors are utilized for indoor air quality measurement.

The first is the Sharp GP2Y1010AU0F sensor, shown in figure 3



Gambar 3. Sharp GP2Y1010AU0F sensor

Which is designed to detect and measure the concentration of fine particulate matter with a diameter of less than 2.5 micrometers (PM_{2.5}) in the air.

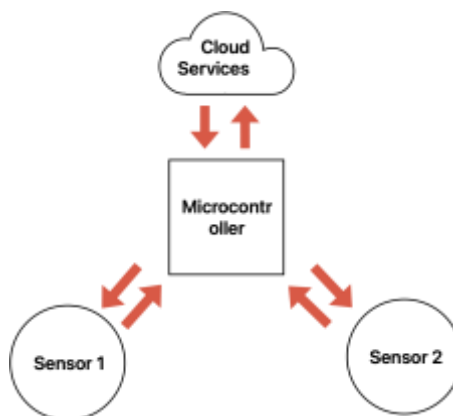
The second is the MQ-7 sensor, shown in figure 4



Gambar 4. MQ-7 sensor

Which detects the concentration of carbon monoxide (CO) in the air, a colorless and odorless toxic gas that poses serious health risks.

Communication between devices in IoT can be carried out using various models, including device-to-device, device-to-cloud, device-to-gateway, and back-end data-sharing models. In this study, the device-to-cloud communication model is employed as depicted in figure 5.



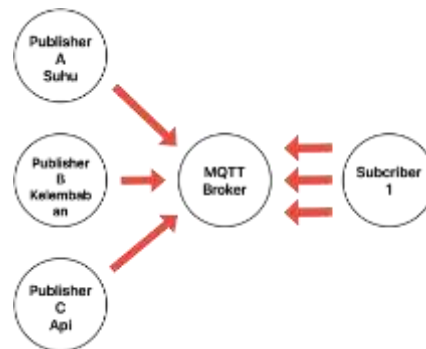
Gambar 5. Device-to-cloud communication model

In which IoT devices transmit sensor data directly to cloud services in real time. The data is subsequently stored and processed in the cloud, enabling remote monitoring and control of devices from any location. This model was selected due to its ability to facilitate seamless integration with cloud platforms, accelerate data processing, and enable efficient access to information. The system implementation utilizes a cloud VPS as the server infrastructure

to run Laravel, an MQTT broker, and InfluxDB, ensuring that all IoT device data is stored and managed centrally within an integrated platform [18].

2.4. MQTT

MQTT (Message Queuing Telemetry Transport) is a lightweight network protocol developed in 1999 that runs on TCP/IP and is widely used in IoT applications for device-to-device communication. Unlike HTTP which adopts a request-response approach, MQTT implements a publish/subscribe architecture in which publishers send data to a broker, which then filters and forwards messages to the appropriate subscribers based on hierarchical topics. In this architecture, MQTT topics are organized hierarchically, similar to a folder structure separated by slashes (/), for example `"/sensor/temperature/client_id"`, which describes the hierarchy of a device with a temperature sensor. This architecture allows for persistent connections between the broker and clients, minimizing message delivery latency and power consumption. The publish/subscribe architecture of MQTT is illustrated in figure 6 [19].



Gambar 6. Publish/subscribe architecture

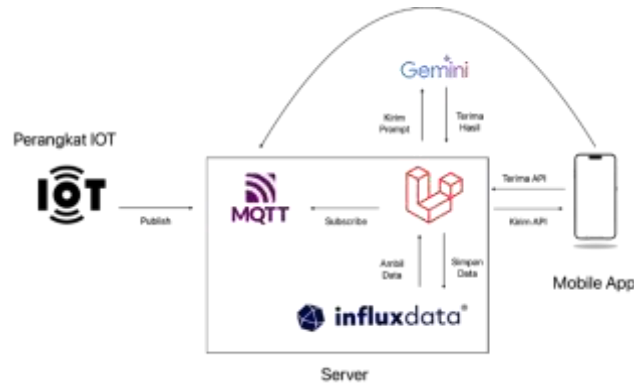
In terms of performance, MQTT demonstrates significantly superior efficiency compared to HTTP. Tests conducted with 50 clients showed that MQTT consumed only 22.63% of CPU resources, whereas HTTP consumed up to 87%. Furthermore, in a scenario involving the transmission of 10,000 payloads, MQTT utilized only 28.35% of CPU compared to HTTP's 94%. Additionally, HTTP shows limitations in CPU usage as the number of clients increases, while an increase in the number of payloads does not significantly impact CPU usage in MQTT. In terms of data transfer speed, MQTT also proved superior, completing the delivery of the same dataset in only 3.24 seconds, significantly faster than HTTP which required 177.344 seconds [20].

2.5. LLM Gemini API

Large Language Models (LLMs) are a form of artificial intelligence designed to understand and generate text across various languages, trained on massive text corpora using deep learning techniques. LLMs are capable of performing a wide range of language-based tasks, including text generation, machine translation, information summarization, question answering, code generation, and sentiment analysis [21]. LLMs operate through transformer architecture, where input text is tokenized and represented as numerical embeddings, enabling the model to predict subsequent words using an attention mechanism that captures contextual relationships within a text sequence. Model accuracy and reliability are further enhanced through fine-tuning, prompt engineering, and Reinforcement Learning from Human Feedback (RLHF) to minimize bias and information errors arising from large-scale unstructured training data [22].

In this study, Google's Gemini API is employed as the primary intelligence component for classifying indoor air quality based on CO and PM2.5 sensor parameters, as well as generating actionable health recommendations appropriate to the detected air conditions. Through a specifically designed prompt, the Gemini API interprets sensor measurement results and produces structured output comprising air quality classifications (e.g., good, moderate, poor) along with practical recommendations such as opening windows, activating ventilation systems, or avoiding outdoor activities. This integration enhances the system's responsiveness and provides meaningful, context-aware guidance to users for maintaining healthy indoor environmental conditions.

2.6. Architecture



Gambar 7. Publish/subscribe architecture

As illustrated in figure 7, the proposed system architecture integrates several interconnected components to enable real-time air quality monitoring, where IoT devices act as both publishers and subscribers to an MQTT broker, enabling bidirectional communication for transmitting sensor data and receiving control commands. Laravel, serving as the backend server, subscribes to MQTT topics published by IoT devices and persists the incoming data into InfluxDB, a time-series database optimized for sensor trend storage and visualization, while also exposing a RESTful API that queries InfluxDB and delivers processed sensor readings to the mobile application. The mobile application can additionally communicate directly with the MQTT broker to issue real-time device control commands, bypassing the backend where low-latency interaction is required. Furthermore, the system incorporates the Gemini API to provide AI-based analysis, whereby upon receiving a user request with a specified time range, Laravel retrieves the corresponding sensor data from InfluxDB and constructs a contextual prompt that is subsequently forwarded to the Gemini API, with the generated response returned to the mobile application via the Laravel API, enabling users to obtain automated insights and recommendations based on historical air quality data.

3. Result

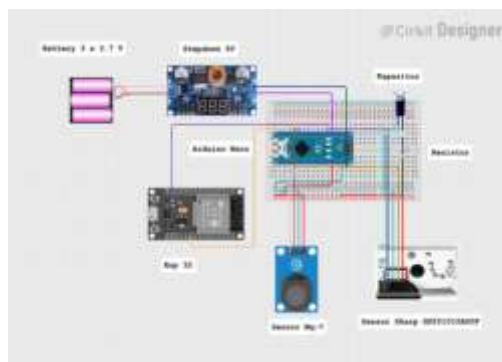
The results represent the implementation of the system design described in the previous chapter. This system is designed to monitor indoor air quality, specifically focusing on the pollutants carbon monoxide and fine particulate matter 2.5.

3.1. System Design

The system design is broadly divided into three main phases: IoT hardware design, backend server design for data processing, and mobile application design as the user monitoring interface.

3.1.1. Internet of Things Design

Prior to physical assembly, the IoT circuit was first designed virtually using Cirkuit Designer, as illustrated in figure 8.



Gambar 8. Internet Of Things Design with Cirkuit Designer

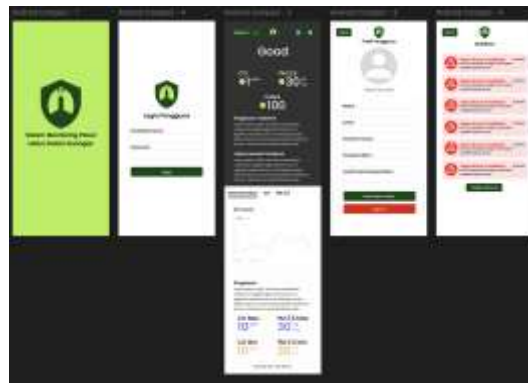
This approach enables comprehensive visualization of all component connections including the step-down module, Arduino Nano, ESP32, MQ-7 sensor, Sharp GP2Y1010AU0F sensor, capacitors, and resistors both schematically and visually. Virtual design was conducted to minimize potential wiring errors, identify conflicts in power and signal paths early on, and ensure alignment between the conceptual design and the overall technical requirements of the system

3.1.2. Backend Design

The backend system was developed with a RESTful API consisting of 11 endpoints, documented using the OpenAPI Specification Swagger 2.0 as a standardized contract between the backend and mobile application. These endpoints are grouped into four functional categories: Auth, which handles user authentication via Login and Logout; User, which manages account operations including Update Profile, Update Profile Image, and Get User Login; Data, which provides access to air quality readings through Get Daily CO Data, Get Daily PM2.5 Data, and Get Device Data; and Notifications, which manages alerts via Get FCM Token, Get User Notification, and Delete All Notifications. All data exchanges are structured in JSON format, with each response containing a status parameter, HTTP status code, descriptive message, and a data object as the main information payload.

3.1.3. Mobile Application Design

The mobile application interface was designed using Figma with a green visual theme representing health and the environment, as illustrated in figure 9.



Gambar 9. Mobile Application Design with Figma

The application consists of five main pages: a Splash Screen displaying the system branding, a Login Page for email and password authentication, a Homepage (Dashboard) presenting real-time air quality status, numerical CO and PM2.5 indicators, daily pollution trend graphs, along with AI-generated summaries and action recommendations, a Profile Page for account management including profile updates and password changes, and a Notifications Page displaying a real-time history of early warnings when air quality is detected as poor or hazardous

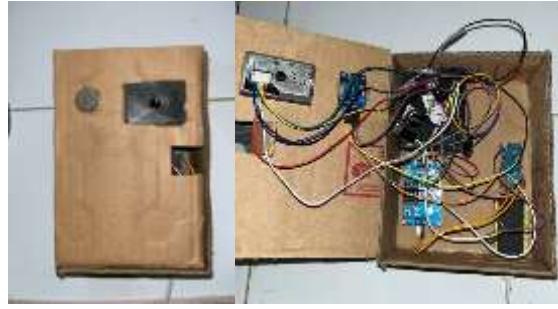
3.2. System Implementation

The development of this system is divided into three main components: IoT development on the device side, backend development on the server side, and a mobile app for user access.

3.2.1. Internet of Things Development

a. Internet of Things Assembly

The physical prototype of the indoor air pollution monitoring system is shown in figure 10.



Gambar 10. External and internal view of the internet of Things Prototype

Physical implementation of the IoT hardware circuit, assembled according to the wiring diagram previously designed. All main components including the ESP32 and Arduino Nano as the processing and wireless communication units, the Sharp GP2Y1010AU0F sensor, and the MQ-7 gas sensor have been successfully interconnected using jumper wires in accordance with the specified connection paths.

b. Calibration and Validation MQ-7

The sensor calibration process is performed in a clean air environment to determine the sensor's baseline resistance (R_0). Under these conditions, the sensor's output voltage was measured at 1.10 volts at a concentration of 0 ppm using a calibration device. The sensor's resistance (R_s) is calculated using the voltage divider equation as follows:

$$r_s = \frac{(V_c - V_{rl}) \cdot R_l}{V_{rl}}$$

By substituting the values $V_c=5$ V, $V_{RL}=1,10$ V, and $R_L=10.000$ obtained:

$$r_s = \frac{(5 - 1.10) \cdot 10000}{1.10}$$

$$r_s = \frac{3.90 \cdot 10000}{1.10}$$

$$r_s = \frac{390000}{1.10} = 35454$$

$$R_0 = 35454$$

Since the measurements were taken under clean air conditions, the value R_0 the same as R_s , so that we obtain $R_0 = 35.454 \Omega$.

The gas concentration is calculated using the sensor characteristic curve equation in the form of a power function, namely:

$$y = 10^{m \cdot \text{ratio} + b}$$

Where y is the gas concentration (ppm) and the ratio is the proportion R_s/R_0 . To determine the constant m and b , two reference points obtained from the sensor datasheet are used, namely:

$$\text{Point 1: } x_1 = 0.25, y_1 = 10$$

$$\text{Point 2: } x_2 = 0.02, y_2 = 1000$$

Both points are converted to a base-10 logarithmic scale, resulting in:

$$\text{Point 1: } x_1 = (0.25) = -0.602, y_1 = (10) = 1$$

$$\text{Point 2: } x_2 = (0.02) = -1.699, y_2 = (1000) = 3$$

Gradient value m calculated using the formula for the slope of a line between two points:

$$m = \frac{y_2 - y_1}{x_2 - x_1} = \frac{3 - 1}{-1.699 - (-0.602)} = \frac{2}{-1.097} = -1.823$$

Next, the value of the constant b is calculated by substituting the value of m and the coordinates of Point 1 into the equation of the line $y=mx+b$:

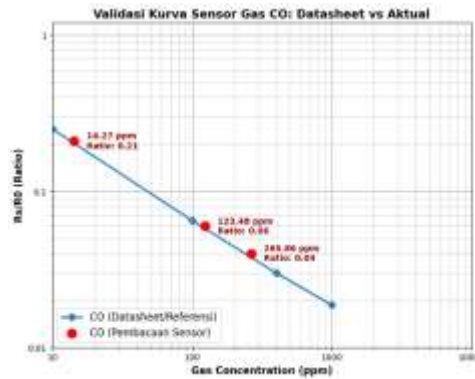
$$y = mx + b$$

$$1 = -1.823 x (-0.602) + b$$

$$1 = 1.097 + b$$

$$b = 1 - 1.097 = -0.097$$

Thus, the sensor characteristic curve constant is obtained with a value of $m = -1,823$ dan $b = -0,097$, which is then used to calculate the gas concentration in the system



Gambar 11. Comparison of Actual ppm with Dataset ppm MQ-7 Sensor

Figure 11 presents the validation results of the MQ-7 CO sensor by comparing the characteristic curve from the sensor datasheet with actual calibrated readings. The graph demonstrates that the sensor's actual readings closely follow the reference curve, confirming consistent sensor behavior across the tested concentration range. Validation was conducted at three concentration points, as summarized in Table I

Table 1. Validation results for the MQ-7 sensor

Ratio	PPM actual	PPM dataset	Difference	Error
0.21	14.27	14.08	0.51 ppm	3.71 %
0.06	123.48	135.00	-11.52 ppm	9.33 %
0.04	265.86	282.60	-16.74ppm	6.30%

At a ratio of 0.21, the sensor recorded 14.27 ppm against a reference of 14.08 ppm (error: 3.71%); at a ratio of 0.06, the reading was 123.48 ppm compared to 135.00 ppm (error: 9.33%); and at a ratio of 0.04, the reading was 265.86 ppm against a reference of 282.60 ppm (error: 6.30%). The average error across all three test points was 6.45%, calculated using the following equation:

$$\text{error} = \frac{\text{ppm aktual} - \text{ppm dataset}}{\text{ppm dataset}} \times 100$$

DOI: <https://doi.org/10.31004/riggs.v5i2.10874>

Lisensi: Creative Commons Attribution 4.0 International (CC BY 4.0)

This result indicates that the calibrated MQ-7 sensor achieves a sufficiently acceptable level of accuracy for use in indoor CO gas concentration monitoring

c. Calibration and Validation Sharp GP2Y1010AU0F

The conversion of the sensor's output voltage into a gas concentration value is performed using a linear equation approach, namely:

$$y = mx + b$$

Where y is the gas concentration (ppm), x is the sensor output voltage (V), and m and b are the slope and intercept constants, respectively, which are determined through the calibration process. In this method, the reference voltage (V_0) does not need to be determined manually because it is already available in the sensor dataset used.

To determine the values of the constants m and b, two reference points obtained from the sensor dataset are used, namely

$$\text{Point 1: } x_1 = 0.6, y_1 = 0$$

$$\text{Point 2: } x_2 = 3.5, y_2 = 0.5$$

The gradient is calculated using the formula for the slope of a line between two points as follows:

$$a = \frac{y_2 - y_1}{x_2 - x_1} = \frac{0.5 - 0}{3.5 - 0.6} = \frac{0.5}{2.9} = 0.172$$

Next, the value of the constant b is calculated by substituting the value of m and the coordinates of Point 1 into the equation $y = mx + b$, resulting in:

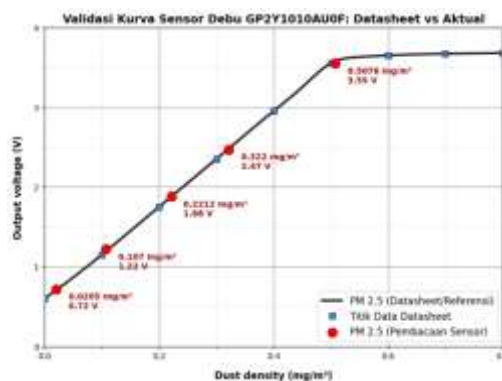
$$y = mx + b$$

$$0 = 0.172 \times 0.6 + b$$

$$0 = 0.1032 + b$$

$$b = -0.103$$

Thus, the coefficients of the linear sensor equation were determined to be $m = 0.172$ and $b = -0.103$, which were then used to convert the sensor's output voltage into a gas concentration value



Gambar 12. Comparison of Actual mg/m3 with Dataset mg/m3 Sharp GP2Y1010AU0F Sensor

Figure 12 presents the validation results of the Sharp GP2Y1010AU0F PM2.5 sensor by comparing the characteristic curve from the sensor datasheet with actual calibrated readings. The graph demonstrates that all

DOI: <https://doi.org/10.31004/riggs.v5i2.10874>

Lisensi: Creative Commons Attribution 4.0 International (CC BY 4.0)

actual sensor readings closely follow the reference curve across all test points, confirming that as dust concentration increases, the sensor output voltage increases proportionally. Validation was conducted at five voltage points, as summarized in Table 2.

Table 2. Validation results for the Sharp GP2Y1010AU0F sensor

Volt	Mg/m3 actual	Mg/m3 dataset	Difference	Error
0.74	0.0205	0.02084	-0.00034	1.63%
1.22	0.1070	0.10684	0.00016	0.15%
1.88	0.2212	0.22036	0.00084	0.38%
2.47	0.3220	0.32184	0.00016	0.05%
3.55	0.5076	0.50760	0	0%

At 0.74 V, the sensor recorded 0.0205 mg/m³ against a reference of 0.02084 mg/m³ (error: 1.63%); at 1.22 V, the reading was 0.1070 mg/m³ compared to 0.10684 mg/m³ (error: 0.15%); at 1.88 V, the reading was 0.2212 mg/m³ against 0.22036 mg/m³ (error: 0.38%); at 2.47 V, the reading was 0.3220 mg/m³ compared to 0.32184 mg/m³ (error: 0.05%); and at 3.55 V, the actual reading was identical to the reference value of 0.5076 mg/m³ (error: 0%). This result indicates that the calibrated Sharp GP2Y1010AU0F sensor achieves a very high level of accuracy, making it highly reliable for use in indoor PM2.5 concentration monitoring.

3.2.2. Backend Development

The backend system is built using the Laravel framework and employs a dual-storage strategy to manage data efficiently. MySQL serves as the relational database for user data management, while InfluxDB is utilized as a time-series database to handle high-frequency sensor data streams. Artificial intelligence capabilities are integrated via the Gemini API, which processes historical data from InfluxDB to generate air quality summaries and actionable recommendations for users.

Data communication between IoT devices and the backend is facilitated through the MQTT protocol. To handle this persistent data stream in real time, Laravel runs a dedicated background service using a console command-based approach, which subscribes to the sensor/air topic via the `mqtt_client` package. Incoming JSON-formatted sensor data is parsed and stored in InfluxDB using an air measurement structure with a `device_id` tag and fields for CO and PM2.5 values.

For security, the system implements token-based authentication using Laravel Sanctum. Upon successful login, each user receives a unique Bearer Token that must be included in the header of every subsequent API request, ensuring secure data exchange between the mobile application and the server

a. Authentication

The authentication system consists of two endpoints. The Login API (`/api/login`, POST) validates user credentials via email or username, verifies the encrypted password against the database, and upon successful authentication generates a Bearer Token via Laravel Sanctum, returning a success response (HTTP 200) containing user data and the session token. Error responses (HTTP 404 or 400) are returned if the user is not found or the password is incorrect. The Logout API (`/api/logout`, POST) terminates the user session by invalidating and deleting the current access token from the database, ensuring it can no longer be used to access system resources, and returns a success response (HTTP 200) upon completion

b. User Management

The User Management category consists of three endpoints. First, Get User Login (GET `/api/user/profile`) retrieves the authenticated user's profile data based on the current session, returning a 200 response with the complete user data on success, or 401 if the session is invalid. Second, Update Profile Image (POST `/api/user/profil-image`) handles profile photo updates with strict validation, accepting only jpg, jpeg, and png formats with a maximum size of 2MB; the system automatically deletes the existing photo before storing the new image in the public directory and updating its reference in the database, returning 200 on success or 400 if validation fails. Third, Update Profile (PUT `/api/user/profile`) allows updates to sensitive user data such as name, email, and password through a multi-layered validation process that ensures email uniqueness, password confirmation consistency, and

verification of the old password against the stored hash; once validation succeeds, the new password is hashed before being saved to the database, returning 200 on success or 400 if any validation check fails.

c. Notification

The Notification management category consists of three endpoints. The Save Token API (/api/update-token, POST) manages FCM token updates for real-time push notification delivery. The system validates the user_id and fcm_token input, and implements smart logic that only performs a database write operation if the new token differs from the stored one, returning HTTP 200 on success or HTTP 400 if validation fails. The Get User Notifications API (/api/notifications/{user_id}, GET) retrieves the complete notification history for a specific user, sorted in descending chronological order. The API consistently returns HTTP 200 regardless of whether notifications exist, returning either a complete notification list or an empty data array with an informative message. The Delete All Notifications API (/api/notifications/delete-all/{user_id}, DELETE) clears a user's entire notification history in a single bulk delete operation. The system validates the operation's success by counting deleted rows, returning HTTP 200 if records were successfully deleted, or an informative response indicating no data was found if no records existed

d. Sensor Data

In the Data category, the Get Daily CO Data API (/api/air/co, GET) and Get Daily PM2.5 Data API (/api/air/pm25, GET) retrieve time-bound sensor data from InfluxDB to generate descriptive statistics, which are then evaluated by the Gemini AI service to produce comprehensive insights and health recommendations returned in a structured JSON response, alongside the Get Device Data API (/api/air/device, GET) that serves as a registry gateway to provide unique device IDs for client-side filtering. Complementing these historical data endpoints, the AI Recommendation API (/api/air/recommendation, POST) acts as an intelligent advisory tool that processes a client-calculated Indoor Air Quality Index (IAQI). This index is derived from CO and PM2.5 readings using the standard linear interpolation formula

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

I_p = index for pollutant p

C_p = the truncated concentration of pollutant p

BP_{Hi} = the concentration of breakpoint that is greater than or equal to C_p

BP_{Lo} = the concentration of breakpoint that is less than or equal to C_p

I_{Hi} = the IAQI value corresponding to BP_{Hi}

I_{Lo} = the IAQI value corresponding to BP_{Lo}

Where each pollutant concentration is mapped to its corresponding breakpoint range

Label	IAQI	CO2 (ppm)	VOC (Index)	NOx (Index)	CH2O (ppm)	PM1 (µg/m3)	PM2.5 (µg/m3)	PM10 (µg/m3)	CO (ppm)	O3 (ppm)
Good	100	400	1	1	0	0	0	0	0	0
	=	=	=	=	=	=	=	=	=	=
	81	599	199	49	0.05	14	20	30	1.7	0.025
Moderate	80	600	200	50	0.06	15	21	31	1.8	0.026
	=	=	=	=	=	=	=	=	=	=
	61	999	249	99	0.1	34	50	75	8.7	0.06
Polluted	60	1000	250	100	0.11	35	51	76	8.8	0.061
	=	=	=	=	=	=	=	=	=	=
	41	1499	349	299	0.3	81	90	125	30	0.075
Very Polluted	40	1500	350	300	0.31	82	91	126	30.1	0.076
	=	=	=	=	=	=	=	=	=	=
	21	2499	399	349	0.75	95	140	200	75	0.1
Severely Polluted	20	2500	400	350	0.76	96	141	201	75.1	0.101
	=	=	=	=	=	=	=	=	=	=
	0	4000	500	500	1	150	200	300	30	0.3

Gambar 13. IAQI breakpoint and category table [24]

To produce an individual index value. The final IAQI is then determined by taking the minimum value among all individual pollutant indices, following the principle that the worst-performing pollutant governs the overall air quality assessment. To optimize resources, this endpoint returns a predefined static response for "Good" IAQI levels (81–100) requiring no intervention, but forwards poorer index values (0–80) to the Gemini AI, instructing it to act as an indoor air quality expert to generate a structured, parsable string containing an Indonesian air quality category, an analytical summary, and practical indoor mitigation strategies like improving ventilation [24]. Furthermore, to prevent excessive API invocations and ensure system efficiency, the mobile application employs a label-based cooldown mechanism that only triggers the AI service when the IAQI label changes or a specific cooldown period elapse, seamlessly utilizing cached fallback responses if the AI service is temporarily unavailable.

3.2.2. Mobile Application Development

The mobile application was developed using the Flutter framework, enabling a single codebase to be compiled natively for both Android and iOS platforms. Flutter's Skia graphics rendering engine ensures high performance and a consistent user interface across various devices. The application adopts the MVVM (Model-View-ViewModel) architectural pattern, which clearly separates business logic, data management, and the user interface into three layers: the Model layer (HomeModel, LoginModel, NotificationModel, UserModel), the View layer (HomePage, NotificationPage, ProfilePage, LoginPage), and the ViewModel layer (HomeViewModel, NotificationViewModel, LoginViewModel, ProfileViewModel).

For data communication, the application integrates an MQTT Service for real-time device connectivity updates and the HTTP package for handling RESTful API requests asynchronously. Air quality trend visualization is implemented using the FL Chart library, while push notifications are delivered via Firebase Cloud Messaging (FCM), ensuring users receive early warnings even when the application is not actively open.

The implemented user interface consists of six main screens. The Splash Screen, as shown in figure 14.



Gambar 14. Splash Screen Implementation

Establishes the visual identity with a green theme and a shield-lung logo representing indoor respiratory health protection.

The Login Screen, illustrated in figure 15.



Gambar 15. Login Screen Implementation

Implements client-side validation before sending credentials to the authentication API, storing the access token upon successful authentication.

The CO Monitoring Screen and PM2.5 Monitoring Screen, as illustrated in figures 16.



Gambar 16. Home Screen Implementation

Respectively, each provide real-time and daily trend charts, AI-generated narrative summaries, health recommendations, and statistical panels displaying minimum, maximum, and average values.

The Notifications Screen, shown in figure 17



Gambar 17. Notification Screen Implementation

Serves as a central repository for air quality early warnings, displaying parameter type, measured value, and timestamp for each alert event.

Finally, the Profile Screen, shown in figure 18.



Gambar 18. Profile Screen Implementation

Enables users to manage their account identity and security, including profile photo updates, personal information edits, and a three-step password change mechanism.

3.3. System Testing

At this stage, the developed system is tested to ensure that all functionalities operate in accordance with the design requirements and objectives

3.3.1. Black Box Testing

Black-box testing was conducted on the fully integrated IoT-based air quality monitoring system in a controlled environment, covering all system components including the ESP32 microcontroller, MQ-7 and Sharp GP2Y1010AU0F sensors, MQTT broker, Laravel backend, InfluxDB database, and mobile application. Each test scenario was executed sequentially and systematically, with the results compared against predefined expected outcomes to determine the success status of each scenario. The complete black-box testing results are presented in Table 3.

Table 3. Black box testing

No	Test Scenarios	Input	Expected Results	Status
1	User login	Registered email address and password	The user successfully accessed the app's home screen	Valid
2	User logout	Click the logout button	The user has successfully logged out, the session has been terminated, and they have been redirected to the login page	Valid
3	Sending sensor data from the ESP32 to an MQTT broker	CO and PM2.5 sensor data	The data was successfully published to the MQTT broker on the specified topic	Valid
4	Receiving sensor data from an MQTT broker in a mobile app	-	The mobile app successfully receives sensor data in real time via an MQTT subscription	Valid
5	Receiving sensor data from an MQTT broker in a Laravel backend	-	The Laravel backend successfully received sensor data via MQTT subscription	Valid

3.3.2. Usability Scale

a. Preparation of The Questionnaire

Usability evaluation was conducted using the System Usability Scale (SUS) method, a widely recognized evaluation instrument in human-computer interaction research. The SUS questionnaire consists of 10 statements measuring users' perceptions of ease of use, efficiency, and satisfaction, answered using a 5-point Likert scale ranging from 1 (Strongly Disagree) to 5 (Strongly Agree). The questionnaire was distributed online via Google Forms to 30 respondents comprising students living in dormitories, selected as representatives of real users potentially exposed to indoor air pollution in their daily lives.

b. Results of the System Usability Scale Calculation

Usability testing was conducted on 30 respondents using the System Usability Scale (SUS) method, with the complete scoring results presented in Table 4.

Table 4. System usability scale calculation

	Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8	Q9	Q10	Skor
1	4	3	3	4	4	3	3	3	4	4	52,5
2	5	3	5	3	5	5	5	5	5	5	60
3	5	1	5	1	5	1	5	1	5	1	100
4	5	2	5	2	5	2	5	2	5	2	87,5
5	4	3	3	2	3	1	4	3	2	2	62,5
6	5	1	4	3	4	2	3	1	4	4	72,5
7	4	2	4	1	5	2	4	1	4	1	85
8	4	4	3	1	4	1	4	1	1	1	70
9	3	3	4	2	3	3	2	2	4	3	57,5
10	4	2	4	1	4	2	2	1	4	2	75
11	3	2	4	2	4	3	4	2	4	3	67,5
12	2	2	4	5	5	2	4	2	4	4	60
13	4	3	4	3	4	3	4	3	4	4	60
14	4	1	5	1	4	1	4	1	4	4	82,5
15	4	4	4	4	4	4	4	2	4	2	60
16	4	2	4	3	4	2	3	1	4	3	70
17	5	2	3	4	5	2	4	2	4	5	65
18	5	5	5	5	5	5	5	5	5	5	50
19	5	1	5	2	5	4	4	1	5	2	85
20	4	3	3	1	4	3	4	2	4	2	70
21	4	2	4	1	5	1	4	2	4	3	80
22	4	2	4	1	4	2	3	3	4	5	65
23	3	1	5	1	5	1	5	1	5	1	95
24	4	4	3	4	4	4	3	3	4	4	47,5
25	4	3	3	3	4	3	3	3	3	4	52,5
26	4	4	4	1	4	2	4	2	4	2	72,5
27	5	1	5	1	5	1	5	2	5	1	97,5
28	5	2	5	1	4	2	5	1	5	2	90

	Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8	Q9	Q10	Skor
29	5	5	5	5	5	5	5	5	5	5	50
30	4	2	4	3	5	1	5	2	3	2	77,5
	Total Score										70,67

The total SUS score obtained from all respondents was 2,120, yielding an average score of 70.67. Based on the SUS rating scale proposed by Brandy Klug (2017), this score falls within the range of 68 to 80.3, categorized as Grade B with an interpretation of "Good" [16]. This result indicates that the developed IoT-based indoor air quality monitoring system, encompassing IoT devices, MQTT communication, and the mobile application, is perceived by users as easy to use, functionally adequate, and requiring minimal adaptation time for operation.

3.3.3. Expert Validation

Testing of the /api/air/recommendation endpoint was conducted by submitting IAQI values representing five air quality categories: Good, Moderate, Polluted, Very Polluted, and Severely Polluted. Each category was tested three times using varying PM2.5 and CO concentration values calculated through the standard interpolation formula, which were then transmitted to the endpoint via the POST method.

Table 5. Response Gemini API

No	Category	CO	PM 2.5	Response
1	Good	5 µg/m ³ IAQ value = 86	0.5 ppm IAQ value = 87	kategori:baik,ringkasan: Kualitas udara dalam ruangan dalam kondisi baik.,saran: Tidak ada saran khusus, kondisi udara aman.
2	Moderate	25 µg/m ³ IAQ value = 64	2.5 ppm IAQ value = 63	kategori:sedang,ringkasan:Nilai IAQI Anda adalah 63, yang masuk dalam kategori \"Moderate\" atau sedang. Kualitas udara dalam ruangan ini masih tergolong aman, namun dapat mulai menimbulkan ketidaknyamanan bagi individu yang sensitif terhadap polutan.,saran:Buka jendela dan pintu untuk sirkulasi udara segar; Gunakan penyaring udara HEPA jika tersedia; Hindari aktivitas yang menghasilkan asap atau debu di dalam ruangan.
3	Polluted	55 µg/m ³ IAQ value = 43	9 ppm IAQ value = 44	kategori:tercemar,ringkasan:Nilai IAQI sebesar 43 menunjukkan kualitas udara dalam ruangan berada pada kategori \"Polluted\" atau tercemar, yang dapat berdampak negatif pada kesehatan penghuni seperti iritasi mata, hidung, tenggorokan, dan memicu gejala alergi atau asma.,saran:Buka jendela dan pintu untuk ventilasi silang; Gunakan pembersih udara (air purifier) dengan filter HEPA; Hindari aktivitas yang menghasilkan polutan di dalam ruangan seperti merokok atau membakar dupa.
4	Very Polluted	95 µg/m ³ IAQ value = 23	10.5 ppm IAQ value = 23	kategori:sangat tercemar,ringkasan:Nilai IAQI Anda adalah 23, yang menunjukkan kondisi \"Sangat Tercemar\" di dalam ruangan. Kualitas udara yang buruk ini dapat memicu atau memperburuk masalah pernapasan seperti batuk, iritasi tenggorokan, dan kesulitan bernapas bagi penghuni.,saran:Buka jendela dan pintu untuk ventilasi silang;Nyalakan kipas angin atau AC dengan mode ventilasi;Hindari aktivitas yang menghasilkan polusi di dalam ruangan seperti merokok atau membakar lilin.
5	Severely Polluted	145 µg/m ³ IAQ value = 1	16 ppm IAQ value = 1	kategori:berbahaya,ringkasan:Nilai IAQI Anda adalah 1, yang menunjukkan kondisi \"Berbahaya\" di dalam ruangan. Kualitas udara yang sangat buruk ini dapat menimbulkan risiko kesehatan serius bagi penghuni.,saran:Segera buka semua jendela dan pintu untuk ventilasi maksimal; nyalakan kipas angin atau AC dengan mode sirkulasi udara luar jika tersedia; identifikasi dan hilangkan sumber polusi di dalam ruangan seperti asap rokok atau produk pembersih beraroma kuat.

As presented in Table 5, results demonstrated that the system consistently produced contextually appropriate responses for each category. The Good category returned brief responses with no specific recommendations, while the Moderate category introduced preventive advisories. As pollution levels increased through the Polluted and Very Polluted categories, recommendations became more specific, including the use of HEPA-filter air purifiers. The Severely Polluted category generated high-urgency responses with immediate instructions to maximize ventilation and identify pollutant sources.

Response content was validated by an environmental engineering expert, Mr. Ahmad Daudsyah Imami, S.T., M.T., a lecturer at the Environmental Engineering Study Program of Institut Teknologi Sumatera. The validation confirmed that all system-generated responses aligned with scientific standards in the field of indoor air quality, establishing their suitability as a basis for health recommendations delivered to application users.

4. Conclusion

This study successfully designed and implemented an IoT-based indoor air quality monitoring system integrating an MQTT broker and the Gemini API. The system adopts a three-layer IoT architecture comprising an ESP32 and Arduino Nano as the hardware layer, a Laravel backend with InfluxDB and MySQL as the server layer, and a Flutter-based mobile application as the user interface layer. Sensor calibration results demonstrated that the MQ-7 CO sensor achieved an average error of 6.45%, while the Sharp GP2Y1010AU0F PM2.5 sensor achieved a remarkably low average error of 0.44%, confirming that both sensors are sufficiently accurate for indoor air quality monitoring. The MQTT protocol proved effective as the primary communication mechanism, enabling real-time and efficient sensor data transmission from IoT devices to the backend server. The integration of the Gemini API successfully enhanced the system's intelligence by automatically generating air quality classifications, condition summaries, and actionable health recommendations based on historical sensor data. System testing yielded positive results across all evaluation aspects. Black-box testing confirmed that all 13 functional test scenarios were executed successfully. Durability testing over a continuous 24-hour period demonstrated a system uptime of 83.3%, with the MQTT connection restoring itself automatically upon network recovery. Battery life testing showed that the IoT device operated for 3 hours and 5 minutes on a 3×3.7 V battery, indicating that an external power source is recommended for long-term deployment. Usability evaluation using the System Usability Scale (SUS) involving 30 respondents yielded an average score of 70.67, categorized as Grade B with a "Good" interpretation, confirming that the system is easy to use and meets user needs without requiring a steep learning curve. The accuracy and relevance of the AI-generated health recommendations were further validated by an environmental engineering expert, Mr. Ahmad Daudsyah Imami, S.T., M.T., a lecturer at the Environmental Engineering Study Program of Institut Teknologi Sumatera. The validation confirmed that the system-generated responses, encompassing air quality classifications, condition summaries, and health advisories, are in accordance with scientific standards in the field of indoor air quality, thereby establishing the system's credibility and suitability as a reliable health recommendation tool for application users.

References

1. B. P. Pratiwi et al., "Air Quality and Its Impact on Urban Health," *Journal of Environmental Health*, vol. 12, no. 3, pp. 45–52, 2020, doi: <https://doi.org/10.26877/jiu.v6i2.6552>
2. S. Laksono and N. Afiyani, "Air Pollution Sources and Health Impacts: A Review," *Indonesian Journal of Public Health*, vol. 18, no. 1, pp. 10–19, 2023, doi: <https://doi.org/10.31869/mm.v6i1.4696>
3. V. Tran et al., "Indoor Air Pollution and Health Risk Assessment," *Journal of Indoor Environment*, vol. 9, no. 2, pp. 100–110, 2020, doi: <https://doi.org/10.3390/ijerph17082927>
4. I. W. R. Aryanta and S. E. Maharani, "Health Impacts of Air Pollution on Vulnerable Populations," *Bali Medical Journal*, vol. 12, no. 4, pp. 310–318, 2023, doi: <https://doi.org/10.36733/jeco.v3i2.7035>
5. R. A. Mouha, "Internet of Things (IoT): Overview and Future Directions," *Journal of Data Analysis and Information Processing*, vol. 9, no. 2, pp. 77–101, 2021, doi: <https://doi.org/10.4236/jdaip.2021.92006>
6. F. Saraya, G. S. Santyadiputra, and I. M. D. Maysanjaya, "SIMOD: Sistem Monitoring Konsumsi Daya Peralatan Listrik Rumah Berbasis Internet of Things," *INSERT: Information System and Emerging Technology Journal*, vol. 3, no. 2, pp. 62–74, Dec. 2022, doi: <https://doi.org/10.23887/insert.v3i2.50910>
7. I. K. Y. Muliawan, I. G. B. Subawa, and I. B. N. Pascima, "SIMOBI: Sistem Monitoring Solar Hydroponic Berbasis Internet of Things (IoT) Studi Kasus: PT. Dago Engineering," *KARMAPATI (Kumpulan Artikel Mahasiswa Pendidikan Teknik Informatika)*, vol. 14, no. 1, pp. 56–64, 2025, doi: <https://doi.org/10.23887/karmapati.v14i1.93382>
8. I. G. L. A. R. Putra, I. G. L. Wiratma, and I. M. E. Listartha, "Implementasi Teknologi Tepat Guna Berbasis Internet of Things: V-Growz Sebagai Upaya Meningkatkan Hasil Panen Petani Vanili Desa Ambengan," *Proceeding Senadimas Undiksha*, vol. 9, pp. 2039–2046, Nov. 2024, doi: <https://eproceeding.undiksha.ac.id/index.php/SENADIMAS/article/view/730/336>
9. K. U. Ariawan, "Penerapan IoT Untuk Sistem Kendali Jarak Jauh Peralatan Listrik Rumah Tangga Berbasis Raspberry PI," *JANAPATI (Jurnal Nasional Pendidikan Teknik Informatika)*, vol. 9, no. 3, pp. 292–303, Dec. 2020, doi: <https://doi.org/10.23887/janapati.v9i3.23264>
10. T. Akbar, G. Suweken, G. Indrawan, and K. Y. E. Aryanto, "Kotak-Kontak Pintar Pada Rumah Cerdas Berbasis Teknologi Internet of Things," *Jurnal Pendidikan Teknologi dan Kejuruan*, vol. 16, no. 2, pp. 278–286, Jul. 2019, doi: <https://ejournal.undiksha.ac.id/index.php/JPTK/article/view/18937>
11. I. M. A. Wirawan et al., "Heartbeats measurement prototype development based on internet of things," *J. Phys.: Conf. Ser.*, vol. 1165, p. 012001, 2019, doi: <https://iopscience.iop.org/article/10.1088/1742-6596/1165/1/012001>
12. F. F. Setiadi, M. W. A. Kesiman, and K. Y. E. Aryanto, "Detection of dos attacks using naive bayes method based on internet of things (iot)," *J. Phys.: Conf. Ser.*, vol. 1810, p. 012013, 2021, doi: <https://iopscience.iop.org/article/10.1088/1742-6596/1810/1/012013>
13. S. Gitakarma and L. P. A. S. Tjahyanti, "Peranan Internet of Things dan Kecerdasan Buatan dalam Teknologi Saat Ini," *Jurnal Komputer dan Teknologi Sains (KOMTEKS)*, vol. 1, no. 1, 2022, doi: <https://doi.org/10.37637/komteks.v1i1.1060>

DOI: <https://doi.org/10.31004/riggs.v5i2.10874>

Lisensi: Creative Commons Attribution 4.0 International (CC BY 4.0)

14. G. Adamkiewicz et al., World Health Organization Regional Office for Europe: Selected Pollutants. Copenhagen: WHO Regional Office for Europe, 2010. Available: <https://www.who.int/publications/i/item/9789289002134>
15. V. Van Tran, D. Park, and Y. C. 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*, vol. 17, no. 8, 2020, doi: 10.3390/ijerph17082927
16. Kementerian Kesehatan Republik Indonesia, "Peraturan Menteri Kesehatan Republik Indonesia Nomor 7 Tahun 2019," 2019, Available: <https://peraturan.bpk.go.id/Details/111721/permenkes-no-7-tahun-2019>
17. ASHRAE, "ANSI/ASHRAE Addendum e to ANSI/ASHRAE Standard 62.1-2016," Atlanta, GA: American Society of Heating, Refrigerating and Air-Conditioning Engineers, 2018,. Available: https://www.ashrae.org/File%20Library/Technical%20Resources/Standards%20and%20Guidelines/Standards%20Addenda/62.1-2016/62_1_2016_e_20180126.pdf
18. R. A. Mouha, "Internet of Things (IoT): Overview and Future Directions," *Journal of Data Analysis and Information Processing*, vol. 9, no. 2, pp. 77–101, 2021, doi: <https://doi.org/10.4236/jdaip.2021.92006>.
19. H. J. Jara Ochoa, R. Peña, Y. Ledo Mezquita, E. Gonzalez, and S. Camacho-Leon, "Comparative analysis of power consumption between MQTT and HTTP protocols in an IoT platform designed and implemented for remote real-time monitoring of long-term cold chain transport operations," *Sensors*, vol. 23, no. 10, 2023, doi: <https://doi.org/10.3390/s23104896>.
20. I. R. Nugraha, W. H. N. Putra, and E. Setiawan, "A comparative study of HTTP and MQTT for IoT applications in hydroponics," *Jurnal RESTI*, vol. 8, no. 1, pp. 119–126, 2024, doi: <https://doi.org/10.29207/resti.v8i1.5561>.
21. N. Rachmat and D. P. Kesuma, "Implementasi large language models Gemini pada pengembangan aplikasi chatbot berbasis Android," *Jurnal Universitas Islam Ogan Komering Ilir*, vol. 4, no. 1, 2024, doi <http://dx.doi.org/10.31314/juik.v4i1.2831>.
22. IBM, "Apa itu large language models (LLM)?," IBM, 2023, Available: <https://www.ibm.com/id-id/think/topics/large-language-models>
23. B. Klug, "An overview of the system usability scale in library website and system usability testing," *Weave: Journal of Library User Experience*, vol. 1, no. 6, 2017, doi: <https://doi.org/10.3998/weave.12535642.0001.602>.
24. ATMO Team, "Indoor Air Quality Index (IAQI): Standards and Metrics," *Atmotube*, Sep. 5, 2024. [Online]. Available: <https://atmotube.com/blog/indoor-air-quality-index-iaqi>.