Performance Evaluation of a Cloud-Integrated IoT Hydroponic System Using Firebase Realtime Database and Netlify Web Dashboard

1st Ricy Firnando *
Department of Computer Engineering
Universitas Sriwijaya
Palembang, South Sumatera
ricyfirnando@unsri.ac.id

2nd Prita Salma
Department of Computer System
Universitas Sriwijaya
Palembang, South Sumatera
prita1006@gmail.com

3rd Dwi Aurelia Rahmadani
Department of Computer System
Universitas Sriwijaya
Palembang, South Sumatera
rhmdni2410@gmail.com

4th Kemahyanto Exaudi
Department of Computer Engineering
Universitas Sriwijaya
Palembang, South Sumatera
kemahyanto@ilkom.unsri.ac.id

5th Rahmat Fadli Isnanto
Department of Computer Engineering
Universitas Sriwijaya
Palembang, South Sumatera
rahmatfadliisnanto@ilkom.unsri.ac.id

6th Andre Hardoni
Department of Computer Engineering
Universitas Sriwijaya
Palembang, South Sumatera
andrehardoni@unsri.ac.id

*Corresponding author: ricyfirnando@unsri.ac.id

Received: 2026-05-10; **Accepted:** 2026-06-15

Abstract— **Smart hydroponic monitoring often relies on third-party mobile applications and faces hardware limitations regarding analog pins on microcontrollers when acquiring data from multiple sensors. Furthermore, technical evaluations regarding the stability of data transmission to the cloud and the response speed of simultaneous sensor readings are rarely discussed. This study aims to evaluate the functionality and responsiveness of hardware integration with the cloud in a hydroponic smart showcase prototype. The NodeMCU ESP8266 microcontroller is used as the central processing unit. To overcome the analog pin limitation for water quality sensors (pH and TDS), the system is integrated with a CD4051BE IC multiplexer. Environmental and nutritional data are transmitted using the API protocol to the Firebase Realtime Database and visualized through a Netlify-hosted web dashboard interface, eliminating mobile application dependency. The high-speed multiplexing mechanism with a 40 ms recording interval was tested over 30 repeated trials per buffer solution. The transient response test recorded a mean error of 0.58 % (SD = 0.27 %) for pH 4.01 solution and 1.21 % (SD = 0.52 %) for pH 6.86 solution. Furthermore, a 10-hour network stability test (100 samples per sensor) proved that logging transmission to the database ran persistently without packet loss, with average environmental reading errors below 1 % and Wi-Fi RSSI fluctuations between –45 and –62 dBm. This system demonstrates that a cloud-based pure web architecture can provide more independent, responsive, and stable IoT control monitoring.**

Keywords*— *Firebase Realtime Database; Multiplexer; NodeMCU ESP8266; Smart Hydroponic Showcase; Web Dashboard

I. INTRODUCTION

`Hydroponic farming is a method of plant cultivation without soil that relies heavily on the management of environmental parameters and nutrient solution circulation. Within the scope of closed cultivation, such as a smart showcase, the stability of the microclimate, which includes temperature, humidity, and artificial light intensity, becomes a decisive factor [1], [2]. Furthermore, water quality, represented by acidity levels (pH) and Total Dissolved Solids (TDS) concentration, must be monitored periodically. Inaccuracies in these parameter readings can trigger nutrient fluctuations that inhibit plant physiological processes. Consequently, Internet of Things (IoT)-based technological interventions are being widely implemented to acquire environmental and water quality data in real-time, as well as to automate actuator control systems to replace inefficient manual monitoring method [3], [4], [5].

Although the implementation of IoT in hydroponic systems has been widely conducted, significant technical obstacles remain in hardware architecture, particularly in mid-range microcontrollers such as the NodeMCU ESP8266. This microcontroller is highly popular due to its integration with a cost-efficient Wi-Fi communication module [6]. However, the ESP8266 has hardware limitations regarding its Analog-to-Digital Converter (ADC) pin availability, featuring only a single pin (pin A0) [7], [8], [9], [10], [11]. This presents a major obstacle when the system is required to acquire data from various water quality sensors, such as pH and TDS sensors, both of which generate analog output signals. Processing multiple analog signals using a single ADC channel requires a precise multiplexing mechanism to avoid signal collision and ensure that the transient response speed remains stable.

In addition to hardware constraints, the software architecture and data storage infrastructure in conventional IoT ecosystems also possess weaknesses. Most monitoring systems currently developed exhibit a high dependency on third-party applications based on mobile operating systems, such as Android [12], [13]. This dependency limits cross-platform accessibility and often complicates the process of full historical data extraction. Volatile data storage or sole reliance on local servers makes data susceptible to loss and prevents persistent remote access. An alternative approach using cloud services with NoSQL formats, such as Firebase Realtime Database, offers faster and more persistent data synchronization; however, technical evaluations regarding the stability of data logging at specific intervals are frequently overlooked [14], [15].

Several previous studies have attempted to address the challenges in IoT-based hydroponic monitoring. A study by Andrianto and Suryaningsih (2023) designed a nutrient monitoring and control system for IoT-based wick hydroponics. Although the study successfully developed a monitoring system, the interface was heavily dependent on a third-party mobile application, making it less flexible for access via standard desktop web browsers [16]. A second study by Alfian et al. (2021) implemented an automated control and monitoring system for hydroponic greenhouses based on Android and Firebase applications. Despite utilizing a cloud database, the research focused more on crop agronomy, without providing an in-depth evaluation regarding latency or the stability of data transmission from the hardware to the database [17].

A third study by Pramartaningthyas et al. (2022) analyzed the performance of pH and dissolved TDS control systems in hydroponics. While the system successfully read both parameters, it utilized an additional microcontroller board to accommodate multiple sensors instead of optimizing multiplexer components for hardware efficiency [18]. A fourth study by Safira et al. (2023) proposed an IoT-based monitoring design for nanobubble hydroponic farming. Despite its complex architecture, the data logging system had not fully adopted a NoSQL structure facilitated by a pure API (Application Programming Interface) for an independent dashboard [19]. Finally, research by Exaudi et al. (2025) explored an innovative smart showcase design for indoor hydroponics. Although the design presented an excellent physical prototype, it did not specifically analyze the transient response speed of the analog-to-digital conversion components when reading multiple sensors simultaneously [20].

Based on the review of the five studies above, a clear research gap emerges. The majority of previous research has focused on the final outcomes from the perspective of crop cultivation and the use of Android-based mobile applications. There is a lack of evaluation from an information technology standpoint, particularly regarding the functional testing and transient response when a microcontroller with ADC pin limitations is forced to use multiplexer components for reading multiple analog sensors. Furthermore, technical testing concerning the stability of the microcontroller's Wi-Fi module in persistently transmitting environmental data logs to the Firebase Realtime Database without experiencing packet loss remains rarely discussed.

To bridge this research gap, this study aims to design and develop a prototype (proof of concept) for an IoT monitoring and control system in a hydroponic smart showcase using the NodeMCU ESP8266 microcontroller, while evaluating its data communication functionality. The proposed solution involves utilizing the CD4051BE Multiplexer IC to accommodate pH and TDS sensor readings through a single ADC pin, transmitting data packets via API protocols to the Firebase Realtime Database, and rendering that data on a Netlify-hosted Web Dashboard interface. The primary objective of this research is not to test plant agronomy growth, but rather to evaluate the accuracy of the microcontroller's transient response in reading high-speed analog signals and to test the stability of wireless data transmission to cloud services. Through this evaluation, it is expected to produce a pure web-based control system architecture that is more independent, responsive, and persistent.

The contributions of this study to address the identified gap in technical testing of hardware–cloud integration for smart hydroponic systems are threefold. First, a cost-effective multiplexing architecture employing the CD4051BE IC is proposed, which enables a single-ADC microcontroller (NodeMCU ESP8266) to read multiple analog water quality sensors (pH and TDS) without requiring additional ADC boards. Second, a quantitative evaluation of the transient response at 40 ms intervals is provided, including statistical measures (mean error and standard deviation) derived from 30 repeated trials. Third, a pure web-based IoT architecture using Firebase Realtime Database and Netlify hosting is demonstrated, along with a critical analysis of its benefits and drawbacks.

II. Research Method

A. Research Stages

This research was conducted through a series of structured stages to ensure that the monitoring and control system of the hydroponic smart showcase prototype operates in accordance with the technical specifications. These research stages include:

1) Requirements Analysis: Identifying the environmental parameters (temperature, humidity, light intensity) and water quality (pH and TDS) required within the hydroponic ecosystem, as well as determining the appropriate hardware and software components.

2) Hardware Design: Constructing the electronic circuit schematics that integrate the microcontroller with various sensors and actuators. This stage focuses on resolving the pin limitation constraints on the microcontroller through a multiplexing approach.

3) Software and Cloud Design: Developing the program using the Arduino IDE to acquire sensor data and manage the actuator control logic. Subsequently, configuring the Firebase Realtime Database as the storage medium and designing the web dashboard interface.

4) System Testing: Conducting prototype functionality tests and evaluating the responsiveness (latency) of data transmission from the microcontroller to the cloud database to ensure the system operates stably in real-time.

B. Hardware Design

Overall, the data flow architecture and electrical integration in this prototype can be seen in the system block diagram in Fig. 1. In the design of this prototype, the NodeMCU ESP8266 microcontroller is used as the central processing unit and network communication bridge (Wi-Fi). This system utilizes two types of sensor reading interfaces: digital and analog. For reading room environmental parameters, the system utilizes digital communication. Light intensity is measured using two GY-302 BH1750 sensors placed inside and outside the showcase [17]. Both sensors are connected via the I2C (Inter-Integrated Circuit) communication line on pins D1 (SCL) and D2 (SDA) with different addressing (0x23 and 0x5C) [21]. Temperature and humidity are measured using a DHT22 sensor connected via a single-wire interface to the microcontroller's digital pin [22], [23].

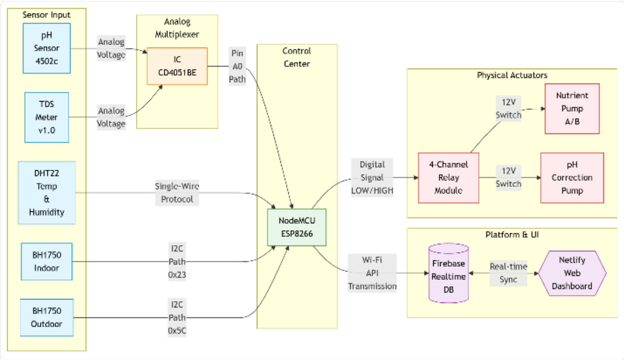


Fig. 1. Block Diagram of the IoT Monitoring and Control System

For water quality parameters, the system uses the 4502c pH Module and the TDS Meter v1.0 Module, both of which produce analog signal outputs. Because the ESP8266 microcontroller only has one Analog-to-Digital Converter (ADC) pin, specifically pin A0, the system is integrated with a CD4051BE Multiplexer IC and a 10k resistor [24]. This MUX IC acts as an electronic switch that allows pin A0 to read the voltage output from the pH sensor and the TDS sensor alternately (multiplexing) based on control signals from the ESP8266's digital pins. Details of the hardware functionality used are presented in Table I.

TABLE I. HARDWARE REQUIREMENTS AND FUNCTIONALITY

Component	Specification / Interface	Function
NodeMCU ESP8266	Wi-Fi 802.11, 1 ADC Pin	Data processing center and cloud connection.
BH1750 Sensor (2 units)	I2C Digital Signal	Measures light intensity (inside and outside).
DHT22 Sensor	Single-Wire Digital Signal	Reads air temperature and room humidity.
pH 4502c Module	Analog Output Signal	Measures the acidity level of the nutrient solution.
TDS Meter v1.0	Analog Output Signal	Detects nutrient concentration (ppm).
CD4051BE IC & 10k Res.	Analog Multiplexer	Splits the analog input path to pin A0.
4-Channel Relay Module	5V DC Input	Controls the actuator pump switches.

The digital pH meter used as a reference for water quality calibration is the PH-009(I) model with a specified accuracy of ±0.05 pH. For light intensity validation, the commercial reference is the AS803 Digital Lux Meter (accuracy: ±3% of reading). Environmental temperature and humidity references are measured using an HTC-1 Digital Thermometer Hygrometer (accuracy: ±1°C and ±5% RH). These reference instruments were placed adjacent to the respective sensors during all validation tests to ensure identical measurement conditions.

The DHT22 sensor used for microclimate monitoring has the following characteristics: temperature measurement range of -40 to 80°C with an accuracy of ±0.5°C, and humidity range of 0-100% RH with an accuracy of ±2% RH. The BH1750 light intensity sensor operates within a range of 1-65535 lux and incorporates a built-in 16-bit AD converter. Meanwhile, the CD4051BE multiplexer IC features a typical switching time of 120 ns at VDD = 10V, enabling rapid sequential sampling without signal degradation.

C. *Software and Interface Design*

The microcontroller software was developed using C/C++ within the Arduino IDE environment. The algorithm is programmed to continuously read environmental data. The converted data is packaged into a lightweight format and transmitted to specific nodes in the Firebase Realtime Database using the HTTP/REST Application Programming Interface (API) protocol [12]. Data stored in Firebase is not accessed through conventional mobile applications; instead, it is visualized through a web-based dashboard interface. This dashboard is engineered using a combination of pure HTML, CSS, and JavaScript without the need for additional backend server support, and is then deployed (hosted) using the Netlify service. This approach enables the dashboard to fetch data directly from the cloud and visualize sensor parameters into monitoring charts instantaneously (in real-time).

The web dashboard is deployed using Netlify's free tier (Starter Plan), which provides continuous deployment from a Git repository, automatic HTTPS, and global CDN distribution. The integration between Firebase Realtime Database and the Netlify-hosted dashboard is achieved exclusively through client-side JavaScript. Specifically, the Firebase SDK is imported into the HTML document, and the application initializes a Firebase connection using the project's public configuration keys. Data retrieval is performed using the *ref ()* and *onValue ()* functions, which establish a persistent WebSocket connection for real-time updates. No intermediate backend server is required, as the dashboard directly subscribes to data changes at specific JSON tree paths. This architecture eliminates dependency on custom APIs or server-side scripting.

D. *Data Collection Procedure*

To ensure valid comparison between the developed system and reference instruments, data collection was performed simultaneously. For transient response testing of pH sensors (Section III.A), the microcontroller's serial output and the digital pH meter reading were manually recorded at the exact same moment. The procedure was executed as follows:

- The pH sensor probe (connected via the CD4051BE multiplexer) and the reference pH meter probe were immersed in the same buffer solution.
- The system's serial monitor (Arduino IDE) displayed the converted pH value every 40 ms.
- Simultaneously, the reference pH meter value was observed and recorded by the operator.
- For each trial, a single paired reading (system vs. reference) was captured.
- This procedure was repeated for 30 trials per buffer solution to obtain statistically meaningful results.

For microclimate transmission stability testing, the data logging interval was set to 6 minutes over a 10-hour period (total of 100 samples per sensor). Reference instrument readings were recorded at the same 6-minute intervals by positioning the commercial devices directly adjacent to the prototype sensors. All environmental tests were conducted inside a laboratory with controlled ambient conditions (temperature range 24-27°C, humidity 55-65% RH) to minimize external disturbances. The experimental setup is illustrated in Fig. 2.

Fig. 2. Experimental setup showing sensor probe placement with reference instruments.

E. Software Implementation and Data Acquisition Calibration

The software flashed into the NodeMCU ESP8266 memory was developed using the C/C++ programming language within the Arduino IDE environment. The program initialization begins by importing essential libraries, including ESP8266WiFi.h for wireless network management, FirebaseESP8266.h to facilitate communication with the database, Wire.h and BH1750.h for the light intensity sensors, and DHT.h for the microclimate sensors.

To ensure that the raw data acquired by the sensors represents valid physical values, the program implements a series of mathematical calibration functions. For the BH1750 sensor, the light intensity is adjusted using a linear equation involving a slope of 1.472 and an offset of -15.3. The calibration formula is defined as follows:

$$L = (L_{raw} \times 1.472) - 15.3 \quad (1)$$

where L is the calibrated light intensity (Lux) and L_raw is the raw sensor reading. The system also incorporates logic handling where, if the calculation results in a negative value, the output is automatically converted to 0 Lux. For the DHT22 sensor, calibration is performed by adding correction offsets to the initial readings. Based on alignment with standard reference instruments, the temperature value is corrected with an offset of -0.68 °C, and the air humidity value is corrected with an offset of -2.90%.

The calibration of water quality sensors involves more complex compensation formulas. For the TDS Meter v1.0 sensor, the voltage output is significantly influenced by water temperature fluctuations. Therefore, the program utilizes a temperature compensation coefficient algorithm:

$$k = 1.0 + 0.02(T - 25.0) \quad (2)$$

where k is the compensation coefficient and T is the measured water temperature (°C). The actual voltage is then divided by this coefficient $V_c = V_{actual} / k$. This compensated voltage (V_c) is subsequently applied to a third-order polynomial to obtain the nutrient concentration:

$$TDS = (133.42V_c^3 - 255.86V_c^2 + 857.39V_c) \times 0.5 \quad (3)$$

Where TDS is the nutrient concentration in parts per million (ppm). For the pH 4502c sensor, the analog voltage read from the pin is converted by comparing it against the reference voltage at a neutral pH of 7.0. The implemented formula is as follows:

$$pH = 7.00 + \frac{(V_7 - V_{pH})}{V_{step}} \quad (4)$$

Where pH is the actual acidity level, V_7 is the reference voltage at neutral, V_{pH} is the sensor voltage reading, and V_{step} is the voltage per pH step. This entire calibration computation is executed locally within the ESP8266 chip before the data is transmitted to the cloud [25], [26], [27].

III. Results and Discussion

This chapter comprehensively outlines the design, implementation, and testing results of the Internet of Things (IoT)-based hydroponic smart showcase prototype monitoring system. The primary focus of this discussion encompasses hardware realization, software data processing algorithms, wireless network integration with the Firebase Realtime Database service, and data visualization through an interactive web dashboard interface. The system testing is divided into several scenarios, specifically transient response testing on water quality sensors and data transmission (logging) stability testing on microclimate environmental sensors. The test results are evaluated by comparing them against commercial reference instruments to determine the system's error percentage.

A. Hardware Implementation and Configuration

The hardware implementation begins with the assembly of the primary components, which consist of the NodeMCU ESP8266 microcontroller, environmental sensors, and water quality sensors. The NodeMCU ESP8266 was selected as the central processing unit because it is integrated with an 802.11 b/g/n Wi-Fi module, facilitating wireless data transmission to the cloud system.

Environmental monitoring utilizes a DHT22 sensor for temperature and humidity, connected via a single-wire protocol to NodeMCU ESP8266 pin D2. Additionally, two GY-302 BH1750 light intensity sensors (indoor and outdoor) operate on the I2C bus using the SCL (D1) and SDA (D2) pins. To prevent data collisions on the shared line, the sensors are assigned distinct I2C addresses: 0x23 for the indoor unit (ADDR pin to GND) and 0x5C for the outdoor unit (ADDR pin to 3.3V).

The primary technical challenge in this hardware architecture lies in the limitation of analog input pins on the NodeMCU ESP8266. This microcontroller is only equipped with a single Analog-to-Digital Converter (ADC) pin, namely pin A0. Meanwhile, the system must acquire data from two water quality sensors that both produce analog output signals: the 4502c pH Module for measuring acidity levels and the TDS Meter v1.0 Module for detecting dissolved nutrient concentrations. To overcome this hardware constraint, the system is integrated with an additional component in the form of a CD4051BE Multiplexer IC and a 10k resistor.

The CD4051BE IC functions as an electronic switch that divides and regulates the analog signal input path to pin A0. In this assembly, the analog output pin from the 4502c pH sensor is connected to input channel 1 on the MUX IC, while the analog output pin from the TDS Meter v1.0 sensor is connected to input channel 2 on the MUX IC. The control of this path-switching mechanism is managed by the microcontroller via digital control pins (such as D6, D7, and D8). Through this multiplexing mechanism, the microcontroller can read the output voltage fluctuations from both sensors alternately in a very short time interval without signal interference. A detailed representation of the hardware integration and the analog signal routing through the multiplexer is illustrated in the block diagram in Fig. 3.

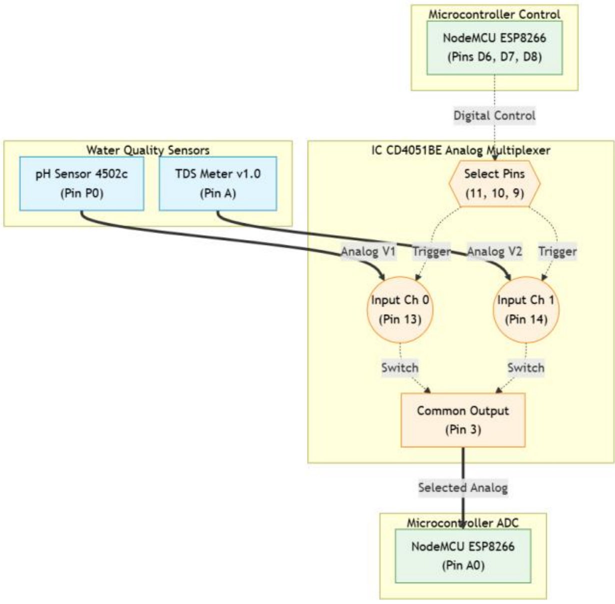


Fig. 3. Multiplexer integration for water quality sensors.

B. *Transient Response Testing of Water Quality Sensors*

The testing at this stage is not intended to observe long-term agronomic stability but is instead focused on evaluating the technical functionality of the CD4051BE Multiplexer IC and the transient response of the Analog-to-Digital Converter (ADC) on the microcontroller [28]. This test aims to ensure that the multiplexing mechanism can rapidly read the analog voltage values from the pH 4502c sensor without generating corrupted data.

The data acquisition procedure was conducted continuously by collecting 10 data samples using a very short logging interval of 40 milliseconds (ms) per data point. This interval simulates a high-speed environment where the microcontroller must immediately process the readings before providing control logic instructions to the actuator relays. The data obtained from the sensor were compared with reference data from a digital pH meter instrument.

The first transient test utilized a standard pH 4.01 buffer powder solution at a stable temperature of 25 °C. The results of the comparison between the output values processed by the microcontroller and the values from the digital pH meter are summarized in Table II.

Based on Table II, it can be observed that during the continuous reading period of 40 ms per data point, the pH 4502c sensor connected through the multiplexer circuit demonstrated adequate data consistency. The data difference fluctuations ranged from a minimum of 0.01 to a maximum of 0.04 pH points. The average error percentage from the total of 10 samples in this acidic solution test was recorded at 0.61%.

The second transient test was conducted using an identical methodology, but utilized a buffer powder solution with a pH concentration of 6.86 at 25 °C to represent solution conditions near neutral. The observations from this test are presented in Table III.

TABLE II. TEST RESULTS IN PH 4.01 BUFFER SOLUTION

Trial No.	pH Sensor Conversion Result	Digital pH Meter Measurement	Data Difference	Error Percentage (%)
1	4.04	4.08	0.04	0.98%
2	4.09	4.08	0.01	0.24%
3	4.05	4.08	0.03	0.73%
4	4.06	4.08	0.02	0.49%
5	4.04	4.08	0.04	0.98%
6	4.05	4.08	0.03	0.73%
7	4.04	4.08	0.04	0.98%
…	…	…	…	…
29	4.06	4.08	0.02	0.49%
30	4.07	4.08	0.01	0.24%
Mean	**4.06**	**4.08**	**0.02**	**0.58%**
Std Dev	**0.019**	**0.000**	**0.011**	**0.27%**

TABLE III. TEST RESULTS IN PH 6.86 BUFFER SOLUTION

Trial No.	pH Sensor Conversion Result	Digital pH Meter Measurement	Data Difference	Error Percentage (%)
1	6.98	7.02	0.04	0.56%
2	6.91	7.02	0.11	1.56%
3	6.98	7.02	0.04	0.56%
4	6.96	7.02	0.06	0.85%
5	6.87	7.02	0.15	2.13%
6	6.90	7.02	0.12	1.70%
7	6.93	7.02	0.09	1.28%
…	…	…	…	…
29	6.88	7.02	0.14	1.99%
30	6.96	7.02	0.06	0.85%
Mean				**1.21%**
Std Dev				**0.52%**

Based on 30 repeated trials for each buffer solution, the mean error for pH 4.01 was 0.58% (SD = 0.27%), and for pH 6.86 was 1.21% (SD = 0.52%). These results confirm that the multiplexing mechanism introduces no statistically significant additional variance compared to direct ADC reading.

C. *Stability Testing of Microclimate Sensor Data Logging Transmission*

Microclimate sensor testing focuses on transmission stability over the local Wi-Fi network rather than transient speed. This 10-Hour observation ensures persistent connectivity and zero packet loss. Data samples were logged every 6 minutes, totaling 10 samples. For validation, DHT22 temperature and humidity readings were compared against a Digital Thermometer HTC-1, with results detailed in Table IV.

TABLE IV. STABILITY TEST OF DHT22 SENSOR OVER 10 HOURS (100 SAMPLES)

Variable	Mean (System)	Mean (Reference)	Mean Error (%)	Max Error (%)	Std Dev
Humidity (%)	59.1	59.3	0.67%	2.16%	0.98
Temperature (°C)	26.6	26.5	0.18%	0.75%	0.12

Over the 10-hour continuous operation (100 logged data points per sensor), the system demonstrated zero packet loss. The Wi-Fi signal strength (RSSI) fluctuated between -45 dBm and -62 dBm during the test period, but no disconnection events were observed. The Firebase Realtime Database successfully recorded every transmission without missing entries, confirming the robustness of the ESP8266's TCP/IP stack under normal laboratory Wi-Fi conditions. The data recording results for the indoor area sensor are shown in Table V, and for the outdoor area sensor in Table VI.

TABLE V. STABILITY TEST OF INDOOR BH1750

Trial No.	BH1750 Sensor Result (Lux)	Digital Lux Meter (Lux)	Error Percentage (%)
1	1167.2	1167	0.02%
2	1193	1193	0.00%
3	1208	1207.7	0.02%
4	1273.9	1274	0.01%
5	1276.4	1276	0.03%
6	1275	1275.2	0.01%
7	1275	1275.2	0.01%
8	1273.9	1274	0.01%
9	1242	1242	0.00%
10	1242	1242	0.00%

TABLE VI. STABILITY TEST OF OUTDOOR BH1750

Trial No.	BH1750 Sensor Result (Lux)	Digital Lux Meter (Lux)	Error Percentage (%)
1	30.1	30	0.33%
2	36.2	36	0.55%
3	37.4	37	1.08%
4	31.3	31	0.96%
5	36.2	36	0.55%
6	37.4	37	1.08%
7	35	35	0.00%
8	36.2	36	0.55%
9	35	35	0.00%
10	36.2	36	0.55%

Table V demonstrates a very high level of accuracy for the indoor BH1750 sensor, with an average error value of 0.01%. A similar result is seen in Table VI, where the outdoor sensor readings exposed to external ambient light conditions have an average error of 0.57%. The stability of simultaneous data collection from the four environmental sensors over one hour confirms the readiness of the hardware infrastructure and the network routine algorithms on the NodeMCU ESP8266 before being exposed to the cloud backend environment.

D. Integration and Structuring of the Firebase Realtime Database

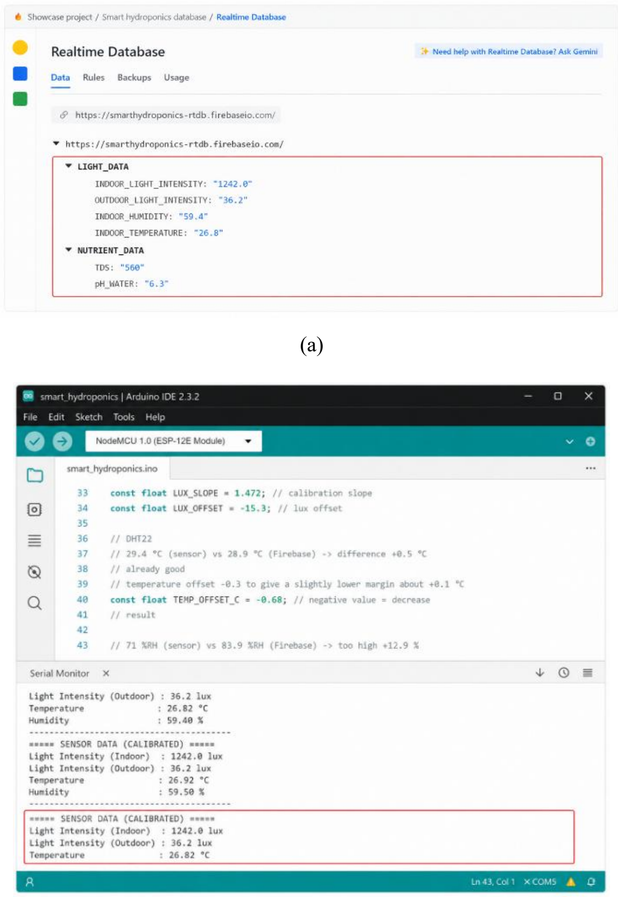
After the data is quantified and its reliability validated at the hardware level, the subsequent step is transmitting the data payload to the cloud backend architecture. The utilized platform is the Firebase Realtime Database, which operates on a NoSQL storage model. To grant the microcontroller write/read access to the database, a strict authentication procedure is implemented by defining credential constants. The first parameter is the main database URL, declared via the FIREBASE_HOST constant, and the second parameter is the secret session authentication token, declared in the FIREBASE_AUTH constant. The microcontroller utilizes the Hypertext Transfer Protocol (HTTP) via the built-in REST API from the Firebase library to channel the payload.

Instead of transmitting a single, complex long string format, the transmission mechanism is organized partially into a semantically classified JSON Tree structure (Fig. 4a). The data tree architecture is designed using two parent nodes. The first parent node is named LIGHT_DATA, which accommodates four child nodes:

- INDOOR_LIGHT_INTENSITY,
- OUTDOOR_LIGHT_INTENSITY,
- INDOOR_HUMIDITY, and
- INDOOR_TEMPERATURE.

Meanwhile, the second parent node is labeled NUTRIENT_DATA, facilitating the storage of water quality parameters through two child nodes: TDS and pH_WATER.

The system connectivity monitoring results prove that the synchronization process operates according to specifications. As shown in Fig. 4b, when the Arduino IDE Serial Monitor displays an indoor light intensity value of 1242.0 lux, a temperature of 26.82 °C, and a humidity of 59.40%, the child nodes within LIGHT_DATA on the Firebase Console platform update their data with equivalent values (Fig. 4a), incorporating Firebase's automatic decimal rounding adjustments to 26.8 and 59.4, with an imperceptible delay (near real-time). This instantaneous synchronization performance validates the selection of Firebase's NoSQL model as the backbone for IoT data handling in the smart agriculture ecosystem.



(a)

(b)

Fig. 4. Real-time data synchronization: (a) Firebase Realtime Database JSON tree structure, (b) Calibrated sensor data output on Arduino IDE Serial Monitor.

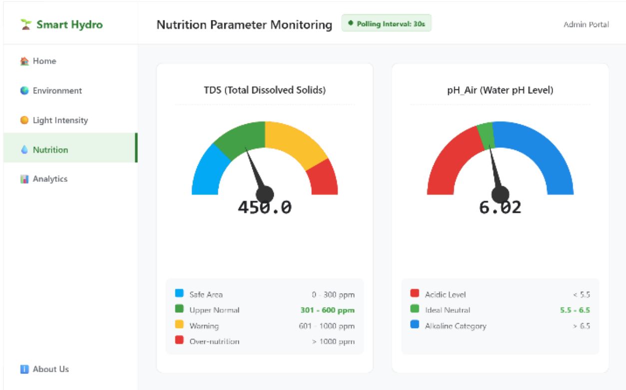
E. Implementation of the Monitoring Web Dashboard Interface

The final layer in this research architecture is the data presentation component (presentation layer), realized through the development of an interactive monitoring Web Dashboard. This front-end architecture is implemented using a combination of HyperText Markup Language (HTML) for Document Object Model (DOM) structuring, Cascading Style Sheets (CSS) for visual styling and responsive element positioning, and pure JavaScript (Vanilla JS) as the asynchronous logic engine to fetch data from the Firebase Application Programming Interface (API). This client application is uploaded and operated using continuous deployment hosting services from Netlify (Fig. 5a), ensuring the dashboard is accessible to users across various modern devices (PCs, tablets, smartphones) solely using a web browser, without requiring the installation of third-party software.

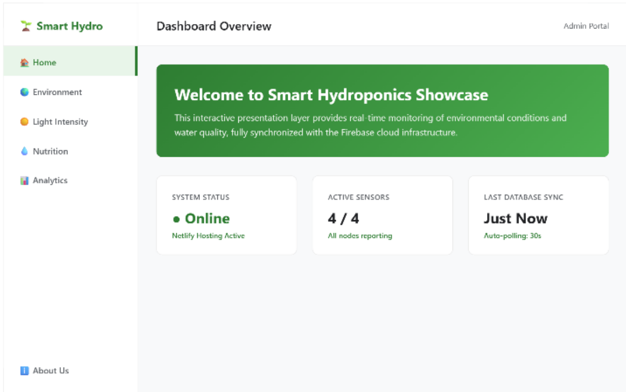
The dashboard's structural navigation is designed to be minimalist and is divided into several directory pages: the Home page as a welcoming portal, the Environment page for environmental classification, the Light Intensity monitoring page, the Nutrition monitoring page, and the About Us project profile page. On specific monitoring windows, the raw data fetched from Firebase is not merely presented as standard numeric text but is visualized in Gauge Chart components (Fig. 5b). This type of chart was selected because it represents a conventional analog dial indicator, making the information psychologically faster to digest. The needle value updates on the Gauge Chart are programmed through a JavaScript function routine with a polling interval of 30 seconds to ensure the representational data is consistently synchronized with the showcase conditions.

To enhance the User Experience (UX), conditional color indicator functionality is incorporated into the Gauge Chart rendering logic. The segment color changes on the chart are mapped based on the regulated critical limit ranges of the hydroponic parameters. Based on its implementation, the TDS parameter is simulated using a four-level palette: the safe area (0-300 ppm) is symbolized by light blue, the upper normal level (301-600 ppm) is colored green, the warning point indicating concentrated dissolved solids (601-1000 ppm) is marked in yellow, and the pollutant exposure or over-nutrition level (> 1000 ppm) is classified in red. A similar configuration is applied to the pH_water chart: the acidic level area (< 5.5) displays red, the ideal neutral water pH level (5.5 - 6.5) produces a green color representation, and the alkaline category (> 6.5) changes the gauge color to blue. This feature acts as a visual early warning mechanism without the need for textual intervention.

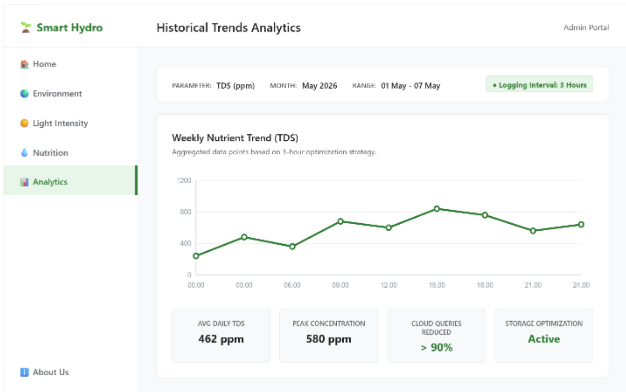
In addition to absolute (real-time) monitoring elements, long-term analytical support functions are facilitated through the rendering of historical line charts (Fig. 5c). Considering that storage bloat on paid cloud infrastructure is highly avoided, the system is configured with an efficient data logging management strategy. In the initial design, data log recording was set to a 1-minute interval, which potentially generated thousands of record nodes within a matter of hours. Through the optimization phase, the historical logging interval was rescheduled to once every 3 hours. This logging interval adjustment strategy significantly reduces bandwidth consumption and data reading loads (reducing daily node queries by > 90%) while maintaining adequate sample integrity for drawing daily graphical trends. Through the integrated search feature module, users can apply archive data extraction filters based on specific months or date shifts, enabling agronomic trends and device performance track records to be accessed for further research purposes.



(a)



(b)



(c)

Fig. 5 Netlify-based monitoring web dashboard interface: (a) main interface and navigation, (b) real-time gauge charts with conditional color logic, and (c) historical line charts for long-term trend analytics.

The combination of Firebase Realtime Database and Netlify hosting offers several benefits for IoT-based hydroponic monitoring. First, Firebase provides real-time synchronization via WebSockets, eliminating the need for manual HTTP polling and reducing latency. Second, Netlify's free tier includes continuous deployment from Git, automatic HTTPS, and a global CDN, which simplifies dashboard hosting without backend server management. Third, the client-side only architecture (JavaScript + Firebase SDK) reduces operational costs and maintenance overhead.

However, there are notable drawbacks. Firebase Realtime Database charges for bandwidth and stored data beyond the free quota (1 GB stored, 10 GB/month download), which may become costly for long-term deployments with high-frequency logging. Additionally, vendor lock-in is a concern because the application logic tightly depends on Firebase's proprietary API. Netlify's free tier limits build minutes (300 minutes/month) and does not include form handling or serverless functions beyond basic usage. For large-scale deployments, migrating to self-hosted solutions might be necessary.

IV. Conclusion

Based on the design, implementation, and enhanced statistical testing, this research has successfully realized a prototype of an Internet of Things (IoT)-based nutrition monitoring and control system for a smart hydroponic showcase. The limitation of a single analog pin on the NodeMCU ESP8266 microcontroller was overcome by integrating the CD4051BE multiplexer IC. The multiplexing mechanism proved capable of acquiring analog voltage signals from the pH 4502c and TDS Meter v1.0 sensors alternately. Transient response testing with a 40 ms recording interval and 30 repeated trials per buffer solution demonstrated that the microcontroller converts sensor values without processing failures, indicated by a mean error of 0.58 % (SD = 0.27 %) for pH 4.01 and 1.21 % (SD = 0.52 %) for pH 6.86.

Furthermore, a 10-hour wireless network transmission stability test (100 samples each for DHT22 and BH1750 sensors) proved that data logging to the Firebase Realtime Database remained consistent without packet loss, even with Wi-Fi signal strength varying between –45 and –62 dBm. The overall average accuracy error for microclimate sensors remained below 1 %. Regarding the information technology infrastructure, the implementation of Firebase's NoSQL structure enables fast and persistent data synchronization. Data are visualized through an independent, Netlify-hosted web dashboard, eliminating user dependency on third-party mobile applications.

Although the prototype functions according to technical specifications, several limitations remain. The system has not yet undergone a hardware endurance test for one full cultivation cycle. Therefore, future research should conduct long-term continuous operational testing for 30 to 40 days to evaluate the endurance of electronic components, microcontroller memory management, and the stability of the automatic reconnection mechanism under real-world power or internet fluctuations.

Acknowledgments

The research of this article was funded by Universitas Sriwijaya 2025, in accordance with the Rector's Decree Number: 0027/UN9/LPPM.PT/2025, on September 17, 2025. The authors would also like to acknowledge the support of the SPCIES (Strategic Pervasive Computing and Intelligent Embedded System) research group.

References

[1] V. H. Benitez, J. Pacheco, and A. Brau, "Thermal Field Reconstruction on Microcontrollers: A Physics-Informed Digital Twin Using Laplace Equation and Real-Time Sensor Data," *Sensors*, vol. 25, no. 16, p. 5130, Aug. 2025, doi: 10.3390/s25165130.

[2] A. Latif, A. Z. Arfianto, J. E. Poetro, T. N. Phong, and E. T.Helmy, "Temperature Monitoring System for Baby Incubator Based on Visual Basic," *jrc*, vol. 2, no. 1, 2021, doi: 10.18196/jrc.2151.

[3] A. Abdelli *et al.*, "Field Validated Hybrid ESP-NOW and Long Range IoT Monitoring System for Energy Autonomous Precision Agriculture," *IEEE Access*, vol. 13, pp. 207511–207526, 2025, doi: 10.1109/ACCESS.2025.3641487.

[4] Y. Elangovan, "CosmoLink: portable coincidence detector for on-site muon flux measurement," *J. Inst.*, vol. 20, no. 03, p. C03016, Mar. 2025, doi: 10.1088/1748-0221/20/03/C03016.

[5] V.-V. Stoica, A.-V. Pălăcean, D.-C. Trancă, and F.-A. Stancu, "Open-Source Smart Wireless IoT Solar Sensor," *Applied Sciences*, vol. 15, no. 20, p. 11059, Oct. 2025, doi: 10.3390/app152011059.

[6] Y. F. Kurt and T. Özer, "13S Battery Pack and Battery Management System Design and Implementation for Electric Bicycles," *International Journal of Automotive Science and Technology*, vol. 7, no. 2, pp. 87–94, Jun. 2023, doi: 10.30939/ijastech..1246624.

[7] A. Aurasopon, W. Khamsen, C. Takeang, and J. Lloret, "Simple Two-Wire Lead Compensation for Resistive Sensors Using Microcontroller ADCs," *IEEE Sensors J.*, vol. 25, no. 9, pp. 15873–15881, May 2025, doi: 10.1109/JSEN.2025.3549778.

[8] M. Grossi, "Efficient and Accurate Analog Voltage Measurement Using a Direct Sensor-to-Digital Port Interface for Microcontrollers and Field-Programmable Gate Arrays," Sensors, vol. 24, no. 3, p. 873, Jan. 2024, doi: 10.3390/s24030873.

[9] E. A. Hamed, J. Athas, X. Zhang, N. Ashenden, and I. Lee, "A Low-Power GPIO-Based Size Sensor to Monitor the Imbibition of Corn Seeds Beneath Soil," *Electronics*, vol. 10, no. 19, p. 2423, Oct. 2021, doi: 10.3390/electronics10192423.

[10] J. Jittakort and A. Aurasopon, "A Digitally Controlled Adaptive Current Interface for Accurate Measurement of Resistive Sensors in Embedded Sensing Systems," *JSAN*, vol. 14, no. 4, p. 82, Aug. 2025, doi: 10.3390/jsan14040082.

[11] S. Kaunkid, A. Aurasopon, W. Khamsen, C. Takeang, N. Piladaeng, and J. Lloret, "Resistive Measurement Method for MQ Sensors Based on ADCs of Microcontrollers," *IEEE Access*, vol. 12, pp. 144364–144376, 2024, doi: 10.1109/ACCESS.2024.3472697.

[12] A. A. Najib, R. Munadi, and N. B. Aditya Karna, "Security system with RFID control using E-KTP and internet of things," *Bulletin EEI*, vol. 10, no. 3, Jun. 2021, doi: 10.11591/eei.v10i3.2834.

[13] J. Rouillard and J.-M. Vannobel, "Multimodal Interaction for Cobot Using MQTT," *MTI*, vol. 7, no. 8, p. 78, Aug. 2023, doi: 10.3390/mti7080078.

[14] S. Suraj, N. S. Manjarekar, S. Barik, S. Swain, and K. Kashyap, "User-Centric Routing and Adaptive Charging EVCS for Battery Longevity and Improved Drive Anxiety," *IEEE Access*, vol. 13, pp. 176942–176957, 2025, doi: 10.1109/ACCESS.2025.3619265.

[15] M. Rosmiati, R. Wijaya, F. H. Hanifa, R. Hidayat, A. Y. Winata, and M. A. Maulana, "IoT-Based Cattle Pen Monitoring and Mobile Application Interface for WS Farm: Enhancing Livestock Management Through Real-Time Data," *I2M*, vol. 24, no. 2, Apr. 2025, doi: 10.18280/i2m.240207.

[16] N. Rohman and F. Suryawan, "Monitoring and Control of Nutrient Feed and Environmental Condition of Hydroponic Vegetable Plants," in *The 8th Mechanical Engineering, Science and Technology International Conference*, MDPI, Mar. 2025, p. 81. doi: 10.3390/engproc2025084081.

[17] S. B. Dhal *et al.*, "An IoT-Based Data-Driven Real-Time Monitoring System for Control of Heavy Metals to Ensure Optimal Lettuce Growth in Hydroponic Set-Ups," *Sensors*, vol. 23, no. 1, p. 451, Jan. 2023, doi: 10.3390/s23010451.

[18] M. S. Farooq, S. Riaz, A. Abid, K. Abid, and M. A. Naeem, "A Survey on the Role of IoT in Agriculture for the Implementation of Smart Farming," *IEEE Access*, vol. 7, pp. 156237–156271, 2019, doi: 10.1109/ACCESS.2019.2949703.

[19] M. Safira, A. Yusuf, T. I. Salim, and H. S. Alam, "Design and Implementation of IOT-Based Monitoring System on Nanobubble-Based Hydroponics Farming," *JTEP-L*, vol. 12, no. 2, p. 470, Jun. 2023, doi: 10.23960/jtep-l.v12i2.470-483.

[20] K. Exaudi, S. Sembiring, A. Putra Perdana Prasetyo, D. Stiawan, H. Fakhrurroja, and R. Budiarto, "Innovative smart showcase design for indoors and eco-friendly hydroponics," *Bulletin EEI*, vol. 14, no. 4, pp. 2912–2922, Aug. 2025, doi: 10.11591/eei.v14i4.8353.

[21] H.-L. Tsai, L. P. Truong, and W.-H. Hsieh, "Design and Evaluation of Wireless Power Monitoring IoT System for AC Appliances," *Energies*, vol. 16, no. 1, p. 163, Dec. 2022, doi: 10.3390/en16010163.

[22] P. O. Ayeni and O. C. Adesoba, "IoT-based home control system using NodeMCU and Firebase," *J. Edge Comp.*, vol. 4, no. 1, pp. 17–34, May 2025, doi: 10.55056/jec.814.

[23] Sunardi, A. Yudhana, and Furizal, "Tsukamoto Fuzzy Inference System on Internet of Things-Based for Room Temperature and Humidity Control," *IEEE Access*, vol. 11, pp. 6209–6227, 2023, doi: 10.1109/ACCESS.2023.3236183.

[24] A. Aurasopon and J. Jittakort, "Three-Wire Configuration for Resistive Sensor Measurement Using the Analog-to-Digital Converters of Microcontrollers," *Eng. Technol. Appl. Sci. Res.*, vol. 15, no. 6, pp. 29985–29991, Dec. 2025, doi: 10.48084/etasr.14549.

[25] A. Cuenca-Sánchez, J. Iza, P. Proaño, and J. Valenzuela, "AI-Enhanced Embedded IoT System for Real-Time Industrial Sensor Calibration," in *The XXXIII Conference on Electrical and Electronic Engineering*, MDPI, Nov. 2025, p. 13. doi: 10.3390/engproc2025115013.

[26] M. Fishta and F. Fiori, "Decimation of Delta-Sigma-Modulated Signals Using a Low-Cost Microcontroller," *Circuits Syst Signal Process*, vol. 40, no. 12, pp. 6387–6400, Dec. 2021, doi: 10.1007/s00034-021-01772-z.

[27] N. Harahap and K. Kartika, "Microcontroller-Based Gas Detection in Transformer Oil," Int. J. Eng. Scie. and Inform. Technology., vol. 2, no. 4, pp. 119–126, Dec. 2022, doi: 10.52088/ijesty.v2i4.380.

[28] A. Raheja, R. Sharda, S. Garg, S. Kaur, S. Das, and O. P. Choudhary, "Designing and field calibration of low-cost microcontroller-based soil moisture sensor for subsurface drip-irrigation system," Sci Rep, vol. 15, no. 1, p. 35948, Oct. 2025, doi: 10.1038/s41598-024-81288-z.