Design and Construction of a Monitoring and Control System for Public Street Lighting Based on the Internet of Things

Reza Fahlevi[1], Andi Ahmad Dahlan[2], Ummul Khair[3]

[1,2,3] Telecommunication Engineering Study Program, Department of Electrical Engineering, State Polytechnic of Padang, Indonesia

[1]zavifahlevi1801@gmail.com, [2]aadfuty@pnp.ac.id, [3]ummul@pnp.ac.id

Abstract—**This research discusses a Public Street Lighting system based on the Internet of Things (IoT) to improve energy efficiency and remote monitoring. The background problem is the need for increased operational efficiency and energy savings in the PJU system. This final project aims to design and test an IoT-based PJU system that transmits real-time data between nodes and gateways using LoRa technology and the MQTT protocol. The research process involves hardware and software design, as well as system testing under various conditions. Testing measures the data transmission time and analyzes delays using LED indicators on gateway devices and dashboards. The test results show significant variations in data transmission times compared to the programmed times. The programmed transmission times are 10 seconds for node 1 and 20 seconds for node 2, but the test results show an average time of around 15 seconds for node 1 and 21.89 to 36.02 seconds for node 2. These variations are caused by factors such as network communication delays, processor loads, and the efficiency of the LoRa system.**

Keywords— IoT, LoRa, MQTT, Public street lighting.

I. INTRODUCTION

Roads are essential elements in both urban and rural contexts, serving as networks that connect various locations for daily activities. Street lighting at night is integral to these routes, providing safety for both road users and pedestrians. Insufficient street lighting contributes to high accident rates, as well as crime and theft, leading to public discomfort and fear of using these roads, especially at night, due to feelings of insecurity and unease [1].

To enhance traffic safety and comfort, street lights or Public Street Lighting (PJU) are used to provide illumination at night, allowing drivers to see the road better [2]. The primary goal of PJU is to provide adequate lighting for drivers to travel safely and comfortably during nighttime [3]. Proper lighting can reduce the number of accidents as well as criminal activities that occur at night. However, efficient management of PJU often poses challenges, particularly in monitoring and controlling street lights that are distributed across various locations.

Currently, the use of Public Street Lighting (PJU) relies on conventional lamps for energy savings [4]. However, due to the large number of installed PJU lights that remain on throughout the night at maximum brightness, this still leads to wastefulness [5]-[9]. One of the main issues is suboptimal energy efficiency. Conventional PJU lights are often left on all night at maximum brightness without any adaptive control mechanisms based on environmental conditions or actual needs. This results in significant energy waste [10]-15]

This study aims to design and develop an Internet of Things (IoT)-based PJU system that uses LoRa technology and the MQTT protocol to monitor and control PJU in real time. This system is expected to improve energy efficiency and accelerate detection and repair when there is a disturbance in PJU.

II. METHOD

A. System Design

The design of the Public Street Lighting (PJU) system based on LoRa and Internet of Things (IoT) technology aims to create a more efficient, energy-saving, and intelligent lighting solution. This system includes hardware and software components working in sync. The hardware involves nodes installed on streetlights to collect environmental data using light and motion sensors. The collected data is sent to a gateway, which connects the nodes to the control center and transmits data to the server. The software uses Node-RED and its dashboard with the MQTT protocol. Node-RED manages logic and integrates sensor data processing, streetlight control, and communication across system components. The Node-RED Dashboard serves as a graphical interface, enabling operators to monitor streetlight status, receive notifications, and control lights via a web interface. The MQTT protocol ensures efficient data and command exchange among system components.

B. System Block Diagram

In the block diagram of this system, a detailed explanation of the block diagram for the PJU system will be provided, which is divided into two main parts: the block diagram of the node system as the data sender and the block diagram of the gateway system as the data receiver. These two diagrams will illustrate the workflow, key components, and interactions between the node and gateway within the PJU system, as well as how data is transmitted, received, and processed to ensure the system functions effectively and efficiently, as shown in Fig. 1.

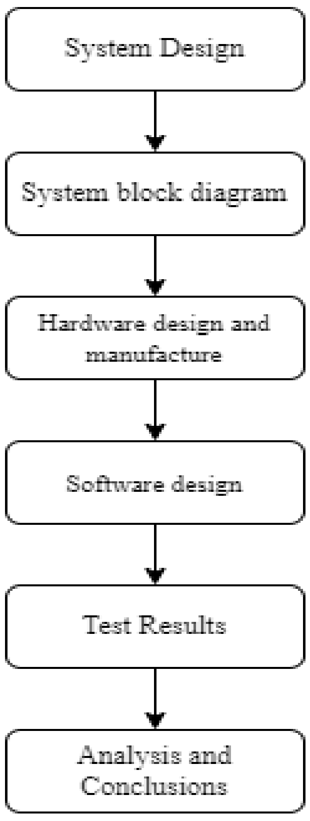


Figure 1. System Design

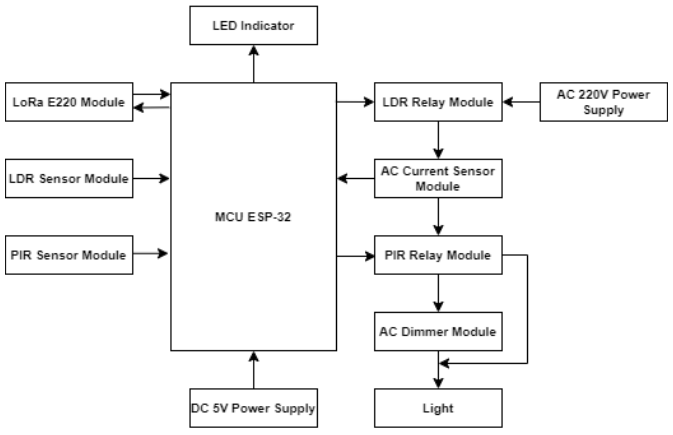


Figure 2. Block Diagram Node

In Figure 2, the Public Street Lighting (PJU) node system operates through three main components: input, process, and output. In the input component, modules like the LoRa E220, LDR sensor, and PIR sensor function to collect environmental data, such as light intensity and motion detection. This data is then processed by the main component, the MCU ESP32, which acts as the central processing unit. The MCU ESP32 gathers and analyzes data from various sensors to determine appropriate actions, such as adjusting the PJU lights' brightness or turning them on or off. In the output component, devices like relays and dimmers control the light operation based on instructions from the ESP32, enabling the lights to automatically turn on, off, or dim according to environmental conditions. Additionally, system data can be transmitted to other nodes through the LoRa E220 module for further monitoring or actions, creating an efficient PJU system that is responsive to its surroundings, as shown in Fig. 3.

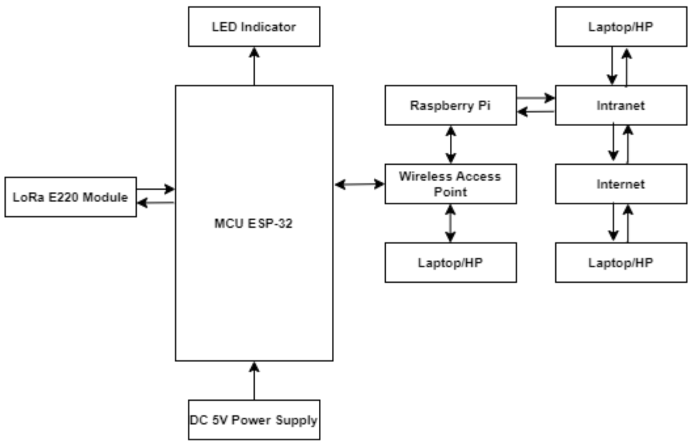


Figure 3. Block Diagram Gateway

In the PJU gateway system block diagram, the architecture enables both local and remote control and monitoring of street lighting. The LoRa E220 module is used for long-range wireless communication, while the MCU ESP32 serves as the main control center, processing data from various devices and managing system output. The Raspberry Pi acts as a mini-server, connecting the system to the local network (intranet) and the internet, facilitating data transmission to the cloud or central server. Through the wireless access point, devices like smartphones or laptops can connect to the system, monitor status, and send commands either locally via intranet or remotely over the internet. The LED indicator provides visual feedback on system status, allowing operators to quickly detect and respond to issues when necessary. This combination of components creates an efficient PJU system, enabling responsive monitoring and control along with flexible access for operators.

C. System Flowchart

In this section there is a flowchart of the node and gateway, which describes in detail the workflow between the two components. This section discusses the flowchart of the node system, which describes the workflow from data collection by the sensor to sending it to the gateway via LoRa. The process begins with initialization, then the data is processed and prepared to be sent if the power supply is available. On data reception (Rx LoRa), if there is data, the system processes and saves the lamp control status, then resets the sending time. On data sending (Tx LoRa), data is sent if the sending time interval is reached, and if not, the process ends without sending.

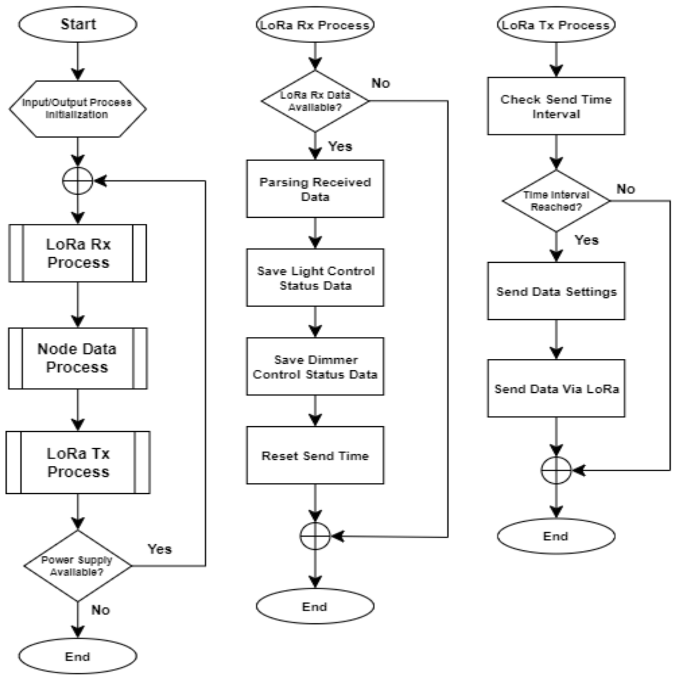


Figure 4. Flowchart Node

Based on the node data processing flowchart, the diagram outlines the lamp control process managed by a node. The process begins by checking the lamp control status. If the automatic mode is activated, the system checks the LDR sensor status. If it is dark, the LDR relay is activated, and the system then checks the AC current sensor status. If AC current is detected, the next step is to check the PIR sensor status. If movement is detected, the PIR relay is activated along with the AC dimmer module to adjust the lamp's brightness. The lamp status is set to ON (bright) with a white LED, indicating that the lamp is fully lit. If no motion is detected, the PIR relay remains off, and the dimmer is set to OFF, indicating the lamp is not illuminated, as shown in Fig. 4.

In another scenario, if manual mode is activated, the LDR relay is activated, and the system checks the AC current sensor status. If AC current is detected, the system examines the dimmer status. If the dimmer is OFF, the PIR relay remains inactive, the lamp status is set to ON (bright), and the LED lights up white. However, if the dimmer is not OFF, the PIR relay and AC dimmer are activated, setting the lamp to ON (dim) with a yellow LED, indicating a dim lighting condition. If AC current is not detected in either scenario, the lamp status is set to disconnected, the LED lights up blue, and the lamp remains OFF. This process shows that the system can operate in two modes, automatic and manual, and is capable of detecting environmental conditions such as light and motion to efficiently control the lamp, as shown in Fig. 5.

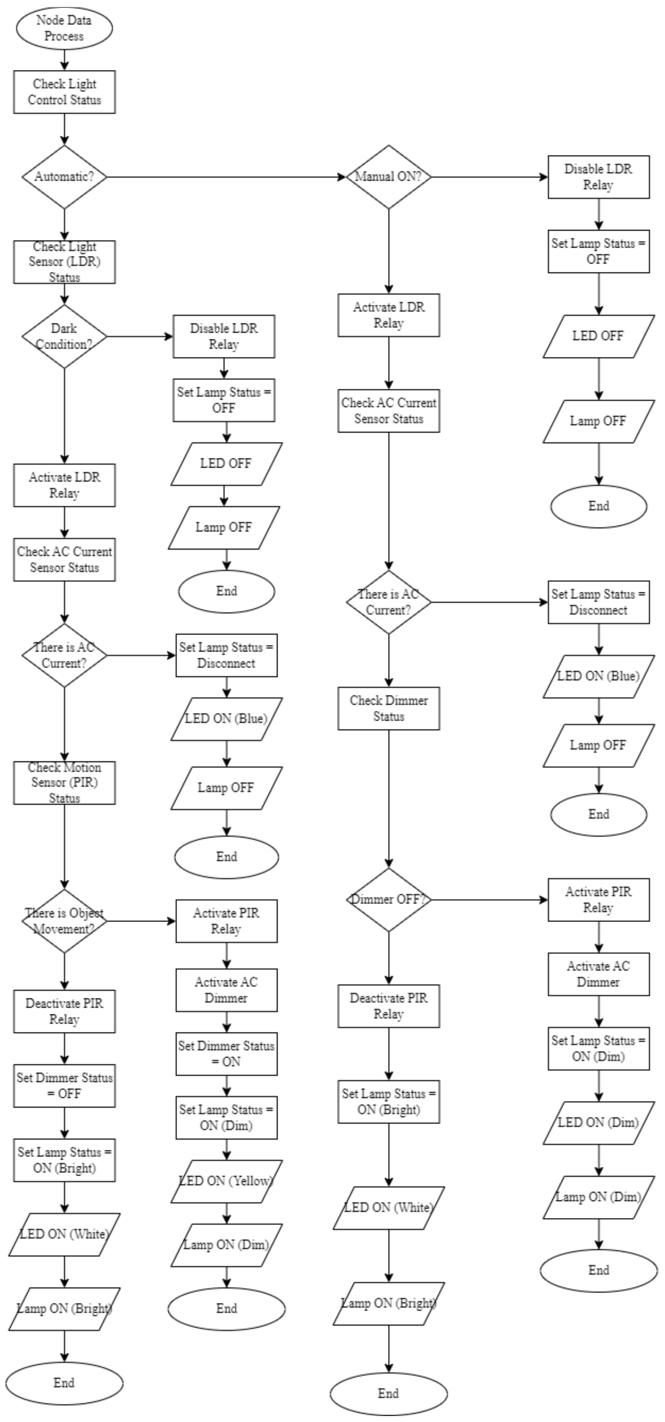


Figure 5. Flowchart Node Data Process

The gateway system flowchart outlines the communication and data management process in operating the PJU, utilizing MQTT and LoRa protocols. The workflow begins with initialization to ensure the readiness of hardware and software, followed by the system subscribing to relevant MQTT topics, enabling the reception of control commands and data from nodes or other gateways, as shown in Fig. 6.

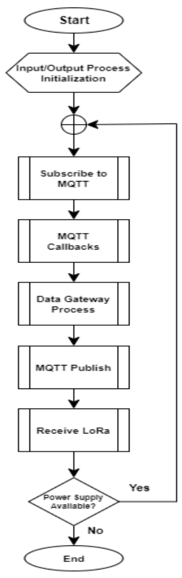


Figure 6. Flowchart Gateway

After initialization, the system subscribes to relevant MQTT topics, allowing nodes to listen to messages sent through the MQTT broker. This subscription is essential for receiving control commands and data from the gateway or other nodes. Figure 7 presents a flowchart detailing the process of subscribing to MQTT topics.

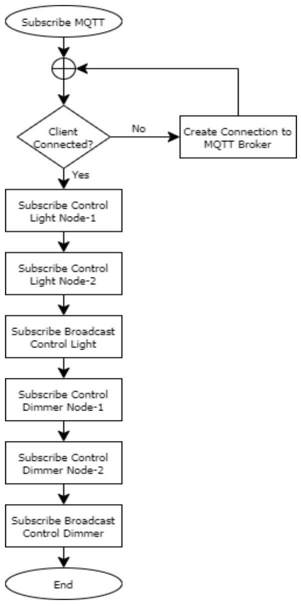

Subscribe MQTT
Client Connected?
No
Create Connection to MQTT Broker
Yes
Subscribe Control Light Node-1
Subscribe Control Light Node-2
Subscribe Broadcast Control Light
Subscribe Control Dimmer Node-1
Subscribe Control Dimmer Node-2
Subscribe Broadcast Control Dimmer
End

Figure 7. Flowchart Subscibe MQTT

The configuration process, as shown in Figure 7, involves setting up system communication with an MQTT broker to control multiple device nodes via subscribed messages. The system initializes by subscribing to relevant MQTT topics. If not connected, it first establishes a connection to the MQTT broker. Once connected, the system subscribes to topics for controlling lights and dimmers on nodes 1 and 2, as well as a broadcast topic for general controls. After subscribing to all topics, the system is ready to receive and process control messages, enabling it to operate lights and dimmers across the network. Figure 8 presents a flowchart illustrating the MQTT callback function that handles incoming messages.

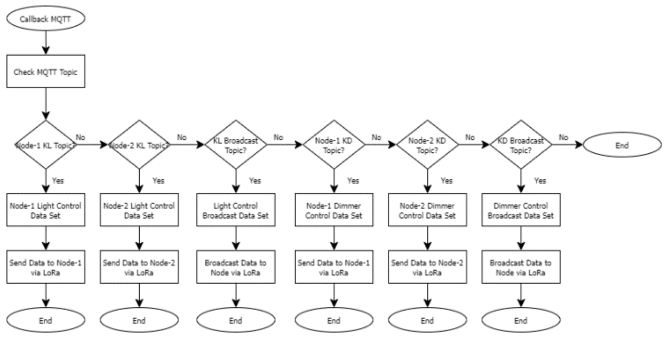

Callback MQTT
Check MQTT Topic
Node-1 KL Topic?
Node-2 KL Topic?
KL Broadcast Topic?
Node-1 KD Topic?
Node-2 KD Topic?
KD Broadcast Topic?
No
Yes
End
Node-1 Light Control Data Set
Node-2 Light Control Data Set
Light Control Broadcast Data Set
Node-1 Dimmer Control Data Set
Node-2 Dimmer Control Data Set
Dimmer Control Broadcast Data Set
Send Data to Node-1 via LoRa
Send Data to Node-2 via LoRa
Broadcast Data to Node via LoRa
Send Data to Node-1 via LoRa
Send Data to Node-2 via LoRa
Broadcast Data to Node via LoRa
End

Figure 8. Flowchart Callback MQTT

The flowchart in Figure 7 outlines a system process that uses MQTT and LoRa for controlling PJU nodes. The process begins by receiving an MQTT callback and identifying the topic to determine the specific control type, whether for individual node lights or dimmers or as a broadcast to all nodes. If the topic is "KL Node-1" or "KL Node-2," the system sets the light control data and sends it via LoRa to the corresponding node. For the topic "Broadcast KL," light control data is broadcast to all nodes via LoRa. Similarly, if the topic is "KD Node-1" or "KD Node-2," the dimmer control data is set and transmitted to the designated node, while "Broadcast KD" sends dimmer control data to all nodes. After completing each process, the system resets, ready to receive a new MQTT callback. The subsequent gateway data processing is illustrated in Figure 9.

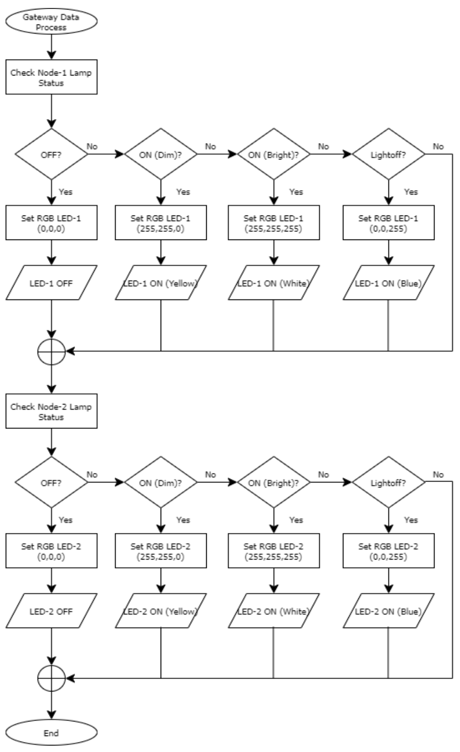


Figure 9. Flowchart Gateway Data Process

The flowchart in Figure 8 illustrates the process of checking the status of lights on two nodes (node 1 and node 2) and setting the RGB LED color according to the light conditions. The process begins with checking the light status on node 1, where, if the light is off, the RGB LED is set to black ([0,0,0]); if dim, it is set to yellow ([255,255,0]); if bright, it is set to white ([255,255,255]); and if there is a disconnection, it is set to blue ([0,0,255]). The same process is applied to node 2. Once the checks on both nodes are complete, the system processes data from the gateway to control the lights and dimmers and publishes the results to the appropriate MQTT topics. Additionally, communication is conducted via LoRa in areas with limited network coverage, as depicted in the flowchart in Figure 10.

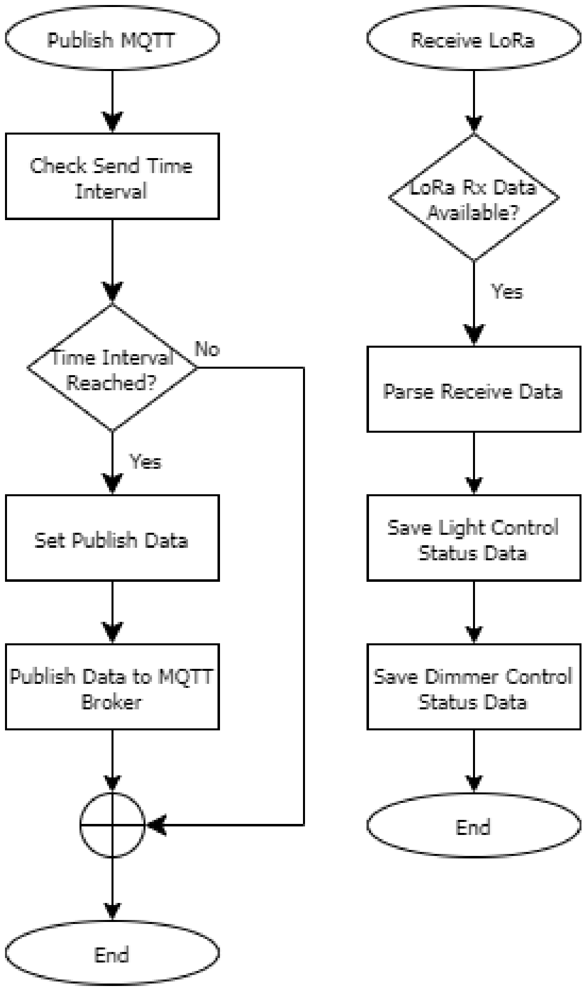


Figure 10. Flowchart Publish MQTT & Receive LoRa

The flowchart in Figure 10 outlines two main processes in the system: data transmission via MQTT and data reception via LoRa. For MQTT transmission, the process begins by publishing data, where the system checks the transmission interval and, once reached, prepares and sends data to the MQTT Broker. For data reception with LoRa, the system first checks for data availability; if data is present, it parses and stores control data for lights or dimmers based on the type. LoRa communication is used as a backup in areas with limited network coverage. Before proceeding, the system checks the power supply; if available, the operation continues, and if not, it stops to prevent damage. This process cycles continuously to maintain responsive operations for control commands and received data.

D. PJU Hardware Design

The hardware design for the LoRa and IoT-based PJU system includes key components such as the PIR sensor, LDR sensor, ESP32, dimmer, and lamp to ensure efficient streetlight control. With a structured design, data from the sensors is sent to the ESP32 for processing, followed by dimmer adjustments to control lamp brightness, saving energy and extending lamp lifespan. This system enables remote monitoring and control, enhancing energy efficiency and safety for road users.

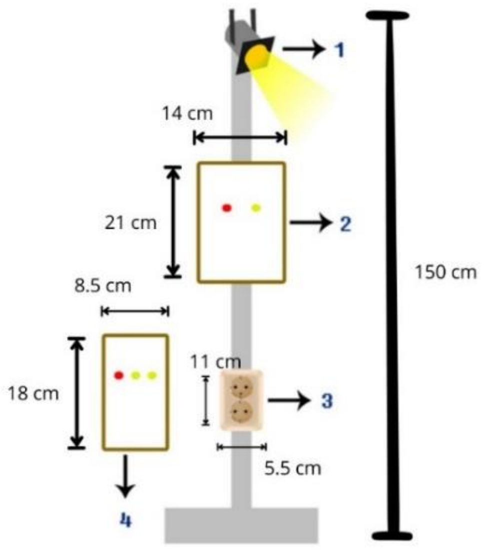


Figure 11. Hardware Design

Figure 11 illustrates the mechanical design of a 150 cm PJU pole and the strategic placement of key components to ensure optimal functionality. The setup includes a well-positioned light fitting for effective illumination, a node box containing essential components (ESP-32 module, LoRa E220-900T30D, 2000W dimmer module, PIR sensor, ZMCT103C current sensor, photosensitive light intensity module, relay, LED indicators, and interconnecting cables) organized for maximum functionality, and a power outlet for the PJU devices. Additionally, a gateway box houses components such as the ESP-32 module, LoRa E220-900T30D, LED indicator, and jumper cables. This arrangement optimizes efficiency, ease of maintenance, and system reliability.

E. Creating a Node System Box and Gateway

In constructing the PJU node system box, strawboard was used as the material of choice. Strawboard was selected for its lightweight yet sturdy nature, providing adequate protection for the electronic components inside. Additionally, this material is easy to shape and adjust according to the available space requirements. The box serves as a housing for the developed sensor system, where sensors are neatly arranged inside. The box dimensions for the data transmission system are 21 x 14 x 7.5 cm, providing sufficient space to accommodate all necessary components. Figure 12 shows a box containing the complete PJU node circuit, illustrating how all components are placed and organized within.

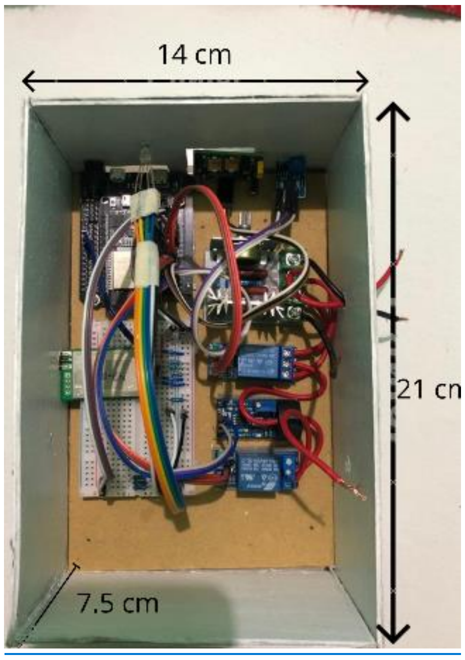


Figure 12. Node System Box

Figure 13 displays a PJU node device specifically designed for public street lighting systems. This device includes essential components such as a light fitting, a node system box, and a stable support pole. All parts are connected through a wiring system that allows integration with power sources and other control systems, ensuring optimal functionality in operation.

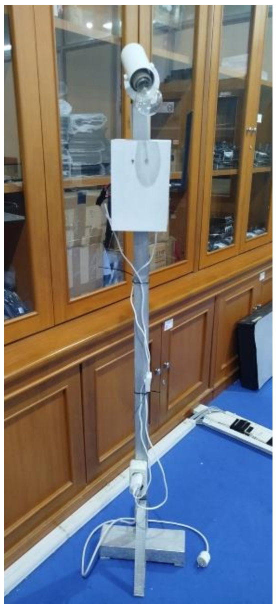


Figure 13. Node System

The gateway system box was constructed using strawboard, with the sensor system and related components neatly arranged inside. Figure 14 illustrates the box containing the complete gateway assembly, with components optimally organized to ensure space efficiency and ease of access. The box dimensions are adjusted to fit the size and number of components used, and the interior design ensures each component has a secure position, preventing movement during operation and reducing the risk of physical damage that could impact system performance. This careful arrangement within the box supports the overall operation of the system, as visualized in Figure 14.

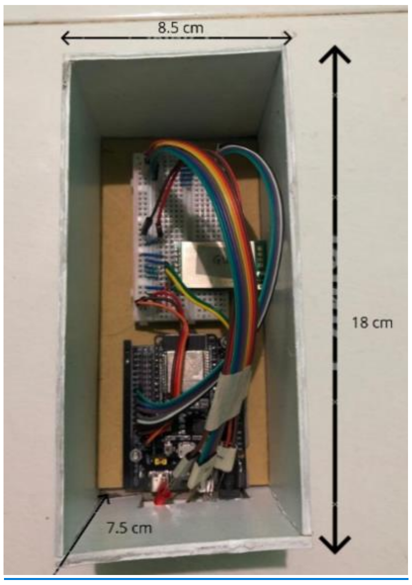


Figure 14. Gateway system device box

Figure 15 shows an illustration of a gateway device designed to connect and manage communication between various nodes in a network. This device is essential for efficiently collecting, processing, and forwarding data, and it is equipped with advanced technology components and features that ensure high performance in complex and integrated systems.

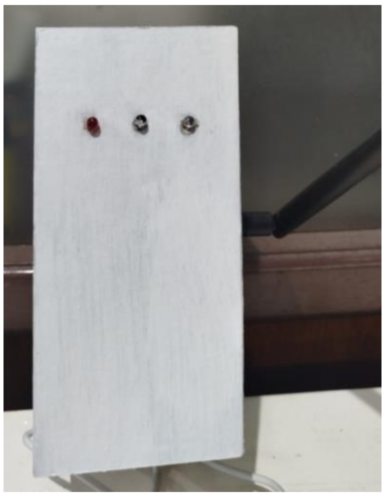


Figure 15. Gateway System Device Box

F. *Flow Node-RED Design*

This system uses Node-RED to manage and control street lights (PJU) by connecting hardware, APIs, and online services, and communicates via the MQTT protocol with a server on a Raspberry Pi. The system consists of several key components, including a PJU monitor debug that monitors data from the PJU nodes, and a node acting as an MQTT client connected to the broker on the Raspberry Pi. Other components process MQTT messages and convert JSON data for use by Node-RED. Control nodes manage the status and brightness of the lights, while notification and indicator nodes display real-time status updates, as shown in Fig. 16.

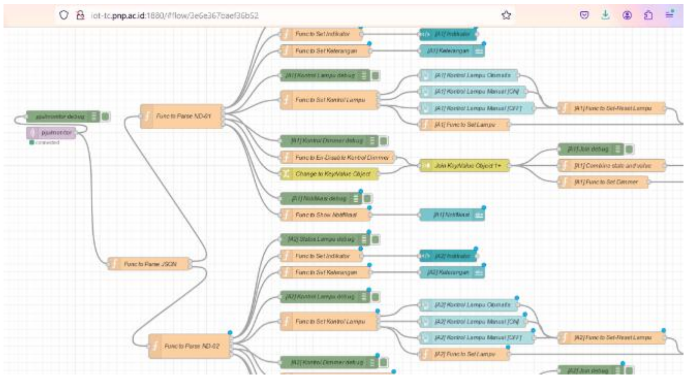


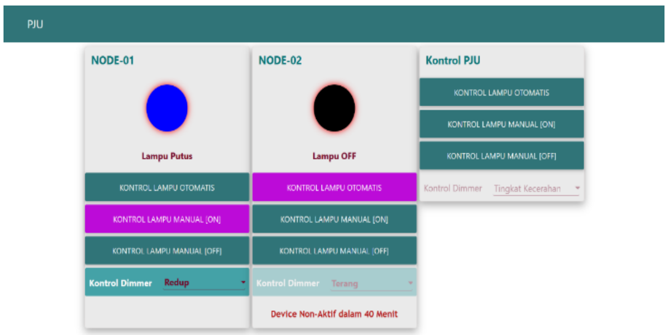
Figure 16. Flow Node-RED Design

The PJU monitoring system, as shown in Figure 10, includes a debug monitor to oversee data from the PJU node, connected to multiple nodes for data processing. The PJU monitor node acts as an MQTT client linked to an MQTT broker on Raspberry Pi, and the "func to parse ND-01 and ND-02" nodes handle parsed MQTT messages. Additionally, a JSON parsing function node converts data to a Node-RED-compatible format. Control nodes for lights and dimmers receive MQTT commands to manage the on/off status and brightness of the PJU lamps, based on sensor data or manual commands. The notification node relays status updates from the broker, while an indicator node shows real-time lamp and brightness levels on the dashboard.

The Node-RED dashboard enables real-time monitoring and control of the PJU system, allowing users to issue commands that are sent as MQTT messages to relevant nodes. This dashboard features widgets like control buttons, graphs, and status indicators, providing visual feedback on the system's condition. The system operates in either automatic mode, adjusting lights based on environmental conditions (light, current, motion), or manual mode, where users control the lights directly. Any status change triggers notifications and real-time indicators on the dashboard, ensuring efficient and energy-saving street lighting operation.

G. *Dashboard Design*

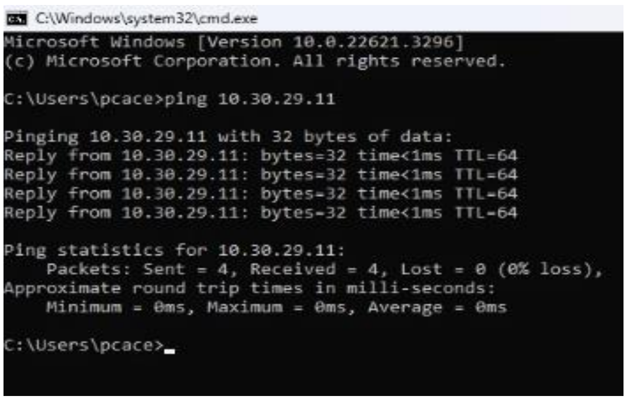
This dashboard is designed to manage and monitor various connected nodes, primarily related to light control and notifications. Each node has similar controls, including indicators, descriptions, automatic and manual light controls, as well as dimmer controls to adjust brightness, as shown in Fig. 17.


PJU
NODE-01
Lampu Putus
KONTROL LAMPU OTOMATIS
KONTROL LAMPU MANUAL [ON]
KONTROL LAMPU MANUAL [OFF]
Kontrol Dimmer
Redup
NODE-02
Lampu OFF
KONTROL LAMPU OTOMATIS
KONTROL LAMPU MANUAL [ON]
KONTROL LAMPU MANUAL [OFF]
Kontrol Dimmer
Terang
Device Non-Aktif dalam 40 Menit
Kontrol PJU
KONTROL LAMPU OTOMATIS
KONTROL LAMPU MANUAL [ON]
KONTROL LAMPU MANUAL [OFF]
Kontrol Dimmer
Tingkat Kecerahan
Figure 17. Dashboard Design

III. RESULTS AND DISCUSSION

A. *Testing Network Connection to Raspberry using LAN*

The connection testing to the Raspberry Pi through the LAN network with IP 10.30.29.11 aims to evaluate the stability and speed of the local connection. The ping results indicate that the payload size of each ICMP packet is 32 bytes, and the packet still has 64 hops remaining (TTL=64), suggesting that it has not passed through many routers. This testing is crucial to ensure the Raspberry Pi functions well as the system's control center, with results covering data transmission speed, packet loss, and overall connection quality, which are essential for the system's daily operations, as shown in Fig. 18.



```
C:\Windows\system32\cmd.exe
Microsoft Windows [Version 10.0.22621.3296]
(c) Microsoft Corporation. All rights reserved.

C:\Users\pcace>ping 10.30.29.11

Pinging 10.30.29.11 with 32 bytes of data:
Reply from 10.30.29.11: bytes=32 time<1ms TTL=64
Reply from 10.30.29.11: bytes=32 time<1ms TTL=64
Reply from 10.30.29.11: bytes=32 time<1ms TTL=64
Reply from 10.30.29.11: bytes=32 time<1ms TTL=64

Ping statistics for 10.30.29.11:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum = 0ms, Average = 0ms

C:\Users\pcace>_
```

Figure 18. Testing Network Connection to Raspberry using LAN

B. *Testing Data Delivery on Dashboard Through LAN Network*

The testing was conducted to measure the data transmission time displayed on the dashboard via the LAN network. It involved observing changes in values on the dashboard, including the status of node devices such as light status, light control, and dimmer control. The tests were performed three times on node 1 and node 2.

1) Node 1

The testing of data transmission time from node 1 to the dashboard over the LAN network in the PJU system yielded significant results, as shown in Table I. This test aimed to measure the efficiency of the time taken for data to be displayed on the dashboard, which is crucial for monitoring and controlling the system. The results indicated that the time required for the transition of PJU node conditions ranged from 20 to 40 seconds, even though the program was set to send data every 10 seconds using LoRa.

An additional 2 seconds was needed for data transmission to the dashboard via the gateway and Raspberry Pi using the MQTT protocol. This protocol facilitates machine-to-machine (M2M) and Internet of Things (IoT) communication through a publish-subscribe mechanism. The test results showed that the recorded transmission time varied between 20.93 seconds and 42.75 seconds, indicating delays in data transmission, which could be attributed to factors such as network speed, network capacity, and the efficiency of the LoRa system. This variation in time signifies that the data transmission did not align with the programmed schedule.

TABLE I
NODE 1 DATA DELIVERY TIME ON DASHBOARD

Node 1 Data Delivery Time on Dashboard	
25.19 s	24.61 s
27.01 s	41.38 s
20.93 s	42.75

2) Node 2

The following is the documentation of data transmission times recorded when data was sent to the dashboard over the LAN network from node 2. According to Table II, although the programmed transmission time was set at 20 seconds, the recorded transmission times varied between 21.89 seconds and 36.02 seconds. While some delays were closer to the programmed time, significant variations indicate that there were delays in data transmission.

TABLE II
NODE 2 DATA DELIVERY TIME ON DASHBOARD

Node 2 Data Delivery Time on Dashboard	
31.86 s	21.89 s
36.02 s	22.05 s
33.31 s	21.57 s

IV. CONCLUSION

The connection testing to the Raspberry Pi through the LAN network for the Public Street Lighting (PJU) system with IP 10.30.29.11 shows significant results in evaluating the stability and speed of the local connection. The ping test results indicate that the ICMP packets have a payload size of 32 bytes with a TTL of 64, suggesting that the connection does not pass through many routers, which is important for the system's performance. The data transmission time from node 1 to the dashboard ranges from 20 to 40 seconds, even though the program is set to send data every 10 seconds. An additional 2 seconds for data transmission through the gateway and Raspberry Pi using the MQTT protocol indicates delays. The variation in recorded transmission times, ranging from 20.93 to 42.75 seconds, reflects issues with alignment to the programmed schedule.

REFERENCES

[1] Bachri, A. (2019) ‘Rancang Bangun Smart Kontrol Lampu Penerangan Jalan Umum Berbasis SMS Gateway’, Jurnal JE-Unisla, Vol 4(2), pp. 1–10.

[2] Tansri, Angga Budi, Subianto, Mohamad, Widodo, Rinto B., Giovanno, Yosua, dan Randi, Ovi Iswanto. 2020. Rancang Bangun Prototipe Sistem Pemantauan dan Pemetaan Lampu Penerangan Jalan Umum (PJU) Berbasis Arduino UNO. Smatika Jurnal, 10(01), 19–25. J. U. Duncombe, “Infrared navigation—Part I: An assessment of feasibility,” IEEE Trans. Electron Devices, vol. ED-11, no. 1, pp. 34–39, Jan. 1959, 10.1109/TED.2016.2628402.

[3] E. Hidayatullah, Sulaeman, Andang, Andang, dan Maulana, Fachruddin. 2022. Penerangan Jalan Umum Pintar dengan Kendali Power Line Carrier berbasis Internet of Things. JITEL (Jurnal Ilmiah Telekomunikasi, Elektronika, dan Listrik Tenaga), 2(1), 47–56.

[4] G. A. Putra, A. A. N. Amrita, dan I. M. Suyadnya, 2018, “Rancang Bangun Alat Monitoring Kerusakan Lampu Penerangan Jalan Umum Berbasis Mikrokontroler dengan Notifikasi SMS,” J. Comput. Sci.

InformaticsEng., vol. 2, pp.90–99, doi :10.29303/jcosine.v2i2.141

[5] H. Amri, J., Lianda, J. Custer, 2018, “Sistem Pengaturan Energi Penerangan Jalan Umum Berbasis Arduino Uno,” Pros. Semin. Nas. Fis., Univ. Riau, no. 3, pp. 31–35.

[6] Perdana, Ridho Hendra Yoga, et al. "Jig detection using scanning method base on internet of things for smart learning factory." 2020 IEEE International IOT, Electronics and Mechatronics Conference (IEMTRONICS). IEEE, 2020.

[7] Kopetz, Hermann, and Wilfried Steiner. "Internet of things." Real-time systems: design principles for distributed embedded applications. Cham: Springer International Publishing, 2022. 325-341.

[8] Suprianto, Dodit, et al. "Hybrid Multi-Servo Motor Controller Within an IoT-Enabled Smart Mechatronics Framework." Inform: Jurnal Ilmiah Bidang Teknologi Informasi dan Komunikasi 10.2 (2025): 121-128.

[9] Idhalama, Ogagaoghene Uzezi, and John Otieno Oredo. "Exploring the next generation Internet of Things (IoT) requirements and applications: A comprehensive overview." Information Development (2024): 02666669241267852.

[10] Hussain, Iqram. "Secure, sustainable smart cities and the Internet of Things: Perspectives, challenges, and future directions." Sustainability 16.4 (2024): 1390.

[11] Rath, Kali Charan, Alex Khang, and Debanik Roy. "The role of Internet of Things (IoT) technology in Industry 4.0 economy." Advanced IoT technologies and applications in the industry 4.0 digital economy. CRC Press, 2024. 1-28.

[12] Yalli, Jameel Shehu, Mohd Hilmi Hasan, and Aisha Abubakar Badawi. "Internet of Things (IoT): origins, embedded technologies, smart applications, and its growth in the last decade." IEEE access 12 (2024): 91357-91382.

[13] Putrada, Aji Gautama, et al. "Machine learning methods in smart lighting toward achieving user comfort: A survey." IEEE access 10 (2022): 45137-45178.

[14] Soheilian, Moe, Géza Fischl, and Myriam Aries. "Smart lighting application for energy saving and user well-being in the residential environment." Sustainability 13.11 (2021): 6198.

[15] Widartha, Vandha Pradwiyasma, et al. "Advancing smart lighting: a developmental approach to energy efficiency through brightness adjustment strategies." Journal of Low Power Electronics and Applications 14.1 (2024): 6.