IIETA International Information and Engineering Technology Association
Advancing the World of Information and Engineering

Ingénierie des Systèmes d'Information
Vol. 30, No. 2, February, 2025, pp. 335-347
Journal homepage: http://iieta.org/journals/isi

Implementation of an Intelligent Power Management System for Building Using Machine Learning Model

Marwa Mushtaq Talib[1*], Muayad Sadik Croock[2]

[1] Department of Computer Engineering, University of Mosul, Mosul 00964, Iraq
[2] Department of Control and Systems Engineering, University of Technology, Baghdad 00964, Iraq

Corresponding Author Email: marwa.21enp1@student.uomosul.edu.iq

https://doi.org/10.18280/isi.300205

Received: 18 September 2024
Revised: 13 December 2024
Accepted: 14 January 2025
Available online: 27 February 2025

Keywords:
power management system, smart buildings, machine learning, WSN, MQTT protocol, Node-RED

ABSTRACT

This paper discusses the design and implementation of an intelligent power management system (IPMS) with the goal of optimizing energy consumption within buildings. The system incorporates wireless sensor networks (WSNs), a machine learning (ML) model, and embedded hardware for monitoring and controlling the power consumption of Heating, Ventilation, and Air Conditioning (HVAC) systems, which are the predominant energy consumers in buildings. The IPMS architecture comprises of local units (LUs) which are equipped with ESP32 microcontrollers and a range of environmental sensors. Additionally, there is a central unit (CU) which is built on a Raspberry Pi4. The CU utilizes a Random Forest machine learning model to analyze real-time sensor data and ascertain the optimal operational mode for each room. This includes transitioning between Shutdown, Select, and Full modes based on factors such as occupancy and environmental conditions. The system is effectively managed and closely monitored using a Node-RED dashboard, which offers a user-friendly interface for seamless control and comprehensive data visualization in real time. The proposed system shows a remarkable level of accuracy in forecasting operational modes and attains substantial energy conservation, as confirmed by different case studies.

1. INTRODUCTION

From residential environments to production facilities, power management systems are quite crucial for maximizing efficiency and energy allocation in many different fields. Given the global need for energy, these technologies are vitally necessary in promoting sustainability and the energy economy. Efficient energy management lowers wasteful power usage and carbon emissions, therefore cutting running costs and environmental damage. One of the main and rising causes of climate change is the building and construction sector as over forty percent of world energy consumption and one-third of annual greenhouse gas emissions come from it [1]. An improved method of tracking and regulating a building's energy consumption is building energy management systems or BEMS. Apart from controlling energy, the system might control a broad spectrum of other building components, whether it is industrial, commercial, or private buildings [2]. Therefore, intelligent control and monitoring systems offer appealing energy savings and a return on investment for newly built and retrofitted buildings when employed at scale.

Integrating cutting-edge energy and technical technology, the smart building idea offers a transforming method for controlling energy use within buildings. Embedded systems integration with WSNs and AI technologies completes the concept of smart buildings. embedded systems along with WSNs create enormous volumes of data. Quickly and powerfully processing this data, artificial intelligence systems can generate insightful analysis and support real-time decision-making [3]. In line with what was already discussed, our study's goal is to present an IPMS that seeks to maximize power usage by carefully controlling the running of HVAC systems, as they are considered the main source of energy consumption in buildings [4].

This study implements a system to enhance power consumption in buildings, the developed system incorporates real-time sensor data with AI-driven decision-making analysis. The system contains several local units (LUs) and a main central unit (CU). Each local unit with ESP32 microcontroller function attached to sensors is located in a room as sensor nodes gathering environmental data including temperature, humidity, light intensity, and occupancy status. On the other hand, the central unit driven by a Raspberry Pi4, acts as a processing center and a gateway. It incorporates a machine learning (ML) model to examine local unit incoming data to make decisions depending on predefined criteria for three classification decisions: Full, Select, and Shutdown. To control and monitor the system, node-red; which is a cloud-based programming tool is used for creatively wiring physical turns on devices, APIs, and online services in innovative combinations.

The paper is structured as follows: Section 2 specifies related works in energy optimization solutions. Section 3 indicates the suggested intelligent power management system. Section 4 discusses the results, whereas Section 5

concludes the findings and suggests future studies.

2. RELATED WORK

Examining studies on how to enhance power management shows that, particularly in recent years, power management has been thoroughly studied in the literature. Many research papers have addressed the idea of Building Energy Management Systems (BEMS) and the degree of the influence of new technology in enhancing this notion. By allowing better energy solutions, embedded systems, WSN, and AI are considered a transformation in power management in buildings. The literature lacks systematic approaches for sensor-actuator integration, which led Fernandes et al. [5] to present the SmartLVEnergy framework, intended to upgrade outdated low-voltage systems. This framework employs a retrofitting strategy that integrates decentralized sensing, distributed computation, and predictive analytics. Following Artificial Intelligence of Things (AIoT) and small machine learning (TinyML) concepts, the framework was used to upgrade the energy distribution panel of a legacy manufacturing facility with sensor devices supporting remote monitoring and decentralized statistical analysis using fifteen-min energy demand forecasting models based on two-layer long-short-term memory (LSTM) networks. This AIoT architecture represented a breakthrough in sensor-actuator retrofitting—that is, combining digital technologies with older systems for smart energy management and low-cost end-to-end IoT system design for smart building energy monitoring and control.

One of the crucial elements for energy control and monitoring in smart buildings includes developing a low-cost end-to-end system as represented by Kök et al. [6] where edge computing enabled ML models were created to forecast building energy use. Based on edge computing, they suggested a four-layer IoT architecture to estimate energy usage in smart buildings, therefore enabling real-time monitoring of energy use. Four-layer IoT system architecture provided comprising physical device layer, edge layer, fog layer, and cloud layer. With the use of artificial intelligence and IoT, test findings reveal that the suggested method may be applied in rapid, efficient, and interpretable building energy management.

In the same context, Essa et al. [7] discussed using IoT technology and artificial intelligence to create a smart system for an educational lab managing energy use. Different controllers in the real system such as Arduino Mega 2560 and Siemens S7-1200 PLC—help to increase system dependability and efficiency. Node MCU ESP8266 sends and receives data through Wi-Fi from and to PLCs and Arduinos while Bylink and the ThingSpeak platform create the IoT facilities. The suggested prototype of building management system (BMS) was tested in a laboratory coupled with several kinds of sensors. The collected data is Fed to an Artificial Neural Network (ANN), enabling the BMS to regulate air conditioning, ventilation, firefighting, lighting, and temperature forecasting.

The importance of real-time power monitoring and adaptive management for smart homes and smart grid-based infrastructure was addressed by Saluja et al. [8]. The suggested solution was a smart energy control system using smart home appliances for infrastructure. The presented solution allowed modern automation and energy usage management by combining real-time sensor data across smart devices with sophisticated AI-driven analysis. The solution guarantees flawless connection between the client appliances (Arduino) and the centralized server (Raspberry Pi) through a ZigBee-based personal area network in a star topology. This enabled quick energy-saving techniques to be applied always. Using defense-in-depth with several levels of protection, the suggested method also emphasizes a safe system to guard data around the smart grid. Implementing real-time monitoring and control energy systems have been investigated thoroughly as revised in references [9-17].

3. PROPOSED IPMS

In this section hardware and software implementations are discussed. Emphasizing the design, functioning, and integration of the system will help grasp the technical elements and data processing in real time.

3.1 System structure

Using a network of microcontrollers and sensors, the intelligent power management system is meant to effectively monitor and control energy use inside a specific area, such as a building or room.

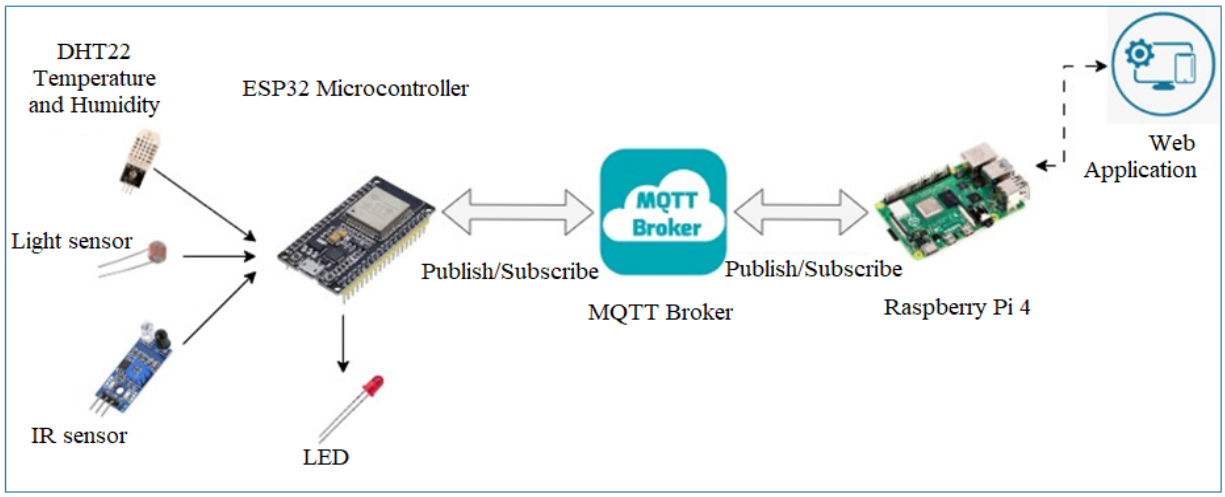

Figure 1. System prototype overview

Two main components—the Central Unit (Raspberry Pi4) and the Local Units (ESP32 microcontrollers)—form the architecture as indicated in Figure 1. The CU acts as the system's brain. It receives data from the LUs, runs ML algorithms on it, and decides how best to control energy use. Using the Message Queuing Telemetry Transport (MQTT) protocol, the Raspberry Pi4 also manages the local units' and its own communications. It processes data using a pre-trained machine learning model, follows particular MQTT topics to receive sensor data and control requests, and subsequently generates control actions back to the local units.

As mentioned before, local units consist of sensors and actuators attached to ESP32 microcontroller. Every ESP32 is linked to three sensors: light sensors (LDR), infrared (IR) sensors for occupancy detection, and temperature and humidity sensors (DHT22). Real-time environmental data collected by these sensors is localized by the ESP32 and then sent to the Raspberry Pi via MQTT. The ESP32 units also follow Raspberry Pi control commands, which they utilize to run actuators such as LEDs that symbolize three modes of operations.

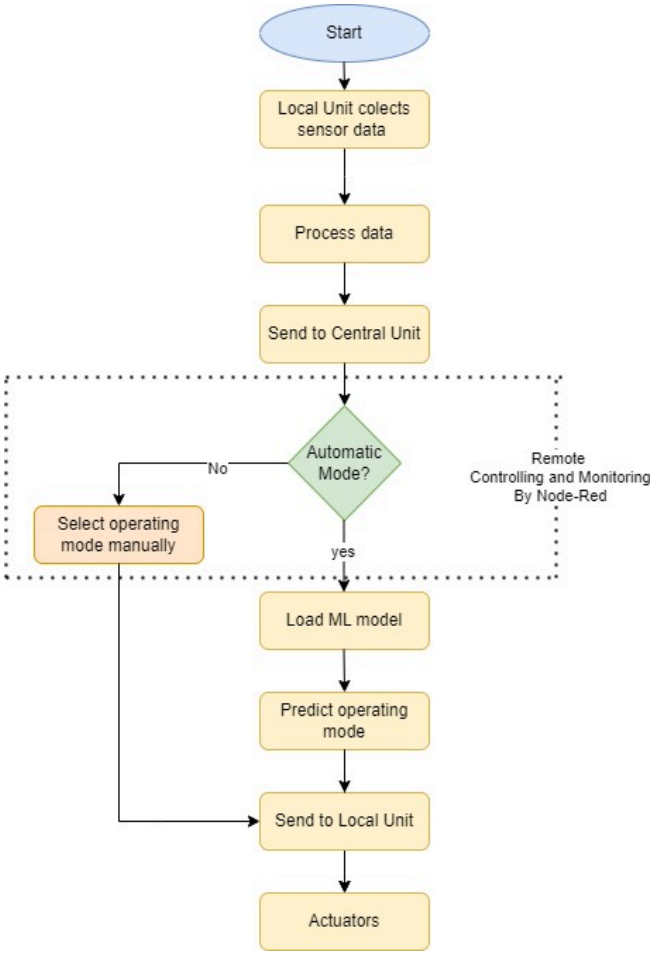

Figure 2. Flow chart of the proposed system

To enable the management and control of this system remotely Node-red is implemented which is a web-based tool to visualize the integration of hardware devices, API's and online facilities. Leveraging Node-RED, one can remotely operate and manage the system by means of real-time data graphical representation, control interfaces, and autonomous decision-making capability. A flow chart for the proposed system is presented in Figure 2.

3.2 ML model

ML model plays a major role in this system by processing real-time sensor data and making smart decisions. Based on the results from our previous work [18] Random Forest RF is best suited to be adopted as the ML model for this system. RF incorporates several decision trees as shown in Figure 3 to improve prediction accuracy with the final forecast derived by averaging the findings of all the trees, this method produces reduced bias and variation as shown in Eq. (1) [19]:

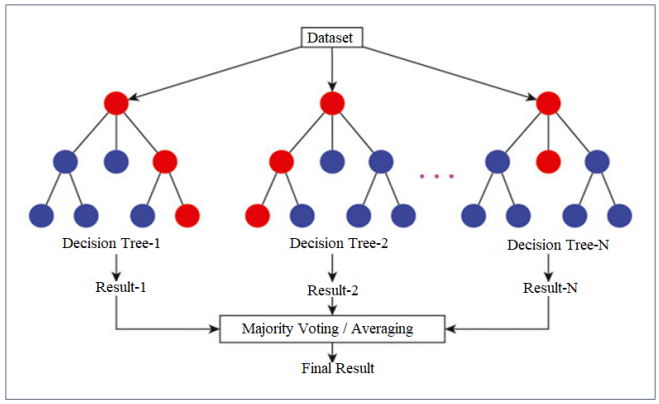

Figure 3. RF classifier [20]

$$X^* = \pi\left(O(c)\middle|\beta = \frac{1}{k\sum_{k=1}^{k} x_{lk}^*\left(f(c)\right)}\right), k^* \qquad (1)$$

where, X^* is the ideal points; $O(c)$ is the observation; β is the learning rate variable; K is the number of decision trees; f is feature transform and lk is a leaf node of the decision tree.

In order to guarantee reliability, the RF approach utilized in this study was configured with particular hyperparameters which employ a total of 1000 estimators (trees) and the maximum depth is set to five to prevent overfitting. Other parameters include a bootstrapping sample technique to increase stability and the Gini impurity being criteria for node splitting.

The CU initially handles and scales real-time sensor data to fit the demands of the model. The ML model then receives the processed data and generates an input-based prediction. After that, the label encoder decodes this prediction back into a legible action, say "Shutdown," and sent to the pertinent topic to guide the LU on the necessary action.

3.3 Dataset structure

ML model is pre-trained by a measured data set taken from a floor office in a university building over one year [21] where both indoor and outdoor measures were acquired. Occupancy-related models can be developed and validated among other uses for the obtained data as a result, the development of the model in this study considers indoor elements such as temperature, humidity, occupant presence, and light conditions. Depending on occupancy status and temperature, a new feature called Mode was included in the dataset to classify it based on the three described modes (Shutdown, Select and Full). As shown in Figure 4 which indicates the distribution of operational modes class Shutdown dominates other classes, to address this limitation of imbalanced data, upscaling measurements are implemented to even out all classes and prevent bias issues. After this point, data is labeled

by tagging them such that the model may recognize the classes by use of tags then features are scaled to improve algorithm performance by balancing the influence of all the elements on distance computation. Feature scaling is crucial since the scale of these features may exceed their values; this can be solved by scaling the features into a designated range [22]. The final step is dividing data set into 80% training sets and 20% testing sets making it ready for the training and testing phase.

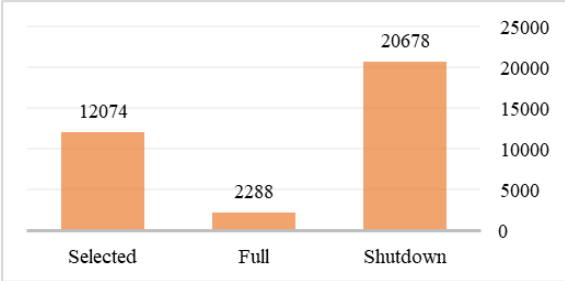

Figure 4. Distribution of operational modes

3.4 Hardware implementation

Three LUs and one CU together implement the hardware. Specifications for each component are discussed here in detail.

3.4.1 Central unit implementation

As mentioned before, CU (Raspberry Pi4) as shown in Figure 5, acts as the core of the system and it takes several steps to accomplish the whole process as discussed below:

Figure 5. CU (Raspberry Pi4)

(1) MQTT communications

MQTT is a lightweight protocol, which makes it perfect for IoT applications, making it efficient for the communication between CU and LUs. Raspberry Pi acts as an MQTT broker and MQTT client which subscribes to particular topics to get data from the LUs. For example, it follows topics like room 1, room 2, and room 3 where every topic relates to a specific room. Sensor data including temperature, humidity, occupancy, and light level abound in these subjects. The CU, being subscribed to a certain topic, gets the data payload right away when LU gathers real-time sensor data. Apart from gathering sensor data, the Raspberry Pi also follows topics like manual_mode_room1 that influence the operational mode of the system. These topics switch from automatic to manual modes depending on commands sent via a user interface, which is in our case a Node-RED dashboard.

CU not only receives data from LUs but also processes them through the ML model and publishes the actions (e.g., Shutdown, Select, or Full) to topics like action_room1, action_room2, and action_room3 to which the LUs already subscribe to those topics. This will make the LUs switch from one mode to another and that's represented in this prototype system by turning ON/OFF LEDs which correspond to each mode of operation.

The MQTT protocol is utilized for allowing light-weight and effective communication between the CU and the LUs as a result Quality of Service measurements (QoS) is set to zero, this ensures the fastest message delivery without acknowledgment to lower latency and resource usage which is suitable for real-time applications. To overcome this limitation, the system checks the MQTT connection between CU and LUs constantly. In case the connection is lost, LUs intended to automatically re-connect with the MQTT broker and continue Publish/Subscribe with topics. Besides that, to reduce the possibility of missed communications, the LUs transmit regular updates to the CU at five-second intervals. If a message is not received, it shall be substituted by the subsequent update to ensure dependable real-time performance for critical decision-making processes.

(2) Mode handling

The CU is responsible for running two separate modes: Manual Mode and Automatic Mode. The default mode, Automatic Mode, uses the ML model and autonomous decision-making to continuously process sensor input. The CU handles received sensor data, forecasts required actions, and transmits commands to the ESP32 nodes to drive actuators like LEDs. Manual Mode lets the system wait for user commands instead of making automatic decisions based on sensor data. To switch from Automatic Mode to Manual Mode a message is received from a subscribed topic like manual_mode_room1 which is controlled by Node-Red web application. In this mode, the user can directly send commands to LUs from the application dashboard.

(3) Multi-room coordination

In order to guarantee efficient power utilization in a multi-room situation, the CU combines and controls data among several LUs. Every LU is given a distinct MQTT client ID to set it apart from others, therefore allowing the CU to access and handle data from individual nodes. Every LU sends its real-time sensor data that is: temperature, humidity, occupancy and light to distinct MQTT topics including room 1, room 2, and room 3. The CU functions as both the MQTT broker as well as client, subscribing to the selected topics and at the same time processing the incoming data. Upon processing, the CU employs the RF machine learning model to predict the suitable mode for each room. Then the decisions made from CU are sent back to LUs through topics like action_room1, action_room2 and action_room3. Each LU defines its related control message through its distinct MQTT identifier and as a result, reacts accordingly.

3.4.2 Local unit implementation

LU which consists of an ESP32 microcontroller and (temperature, humidity, light, and occupancy status) sensors attached to it responsible for the data acquisition process as shown in Figure 6. ESP32 is a strong and flexible microcontroller well-known for its integrated Wi-Fi and Bluetooth capabilities. Its dual-core CPU and plenty of GPIO pins support a broad spectrum of sensors and peripherals helping it to effectively manage functions including data collecting, processing, and communication. Three types of sensors are connected to ESP32 which are DHT22 for temperature and humidity, light sensors represented by LDR sensor which measures light intensity, and Occupancy status sensor which is an IR sensor that can detect motions. LU also

handles three LEDs each one corresponding to one of the three modes of operation (Shutdown, Select, and Full), LU is subscribed to a topic such as action_room1, and upon receiving the command from the topic, it updates the status of the LEDs according to that command.

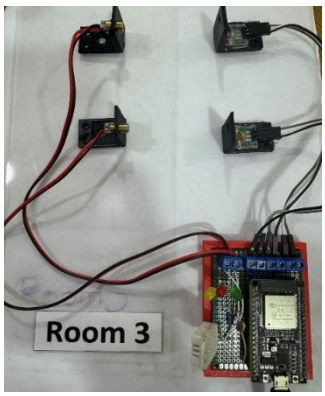

Figure 6. Local unit

The ESP32 gathers regular interval data from the DHT22, LDR, and IR sensors. The sensor readings, temperature and humidity, for the DHT22 are numerical values that are not needed for further processing. The IR sensors accumulate binary outputs, either motion or non-motion, that are used to determine occupancy levels over time. LDR sensor measures light intensity and based on a specific threshold it indicates whether there is light or not. After that, the gathered data is arranged into a structured payload. This payload is ready for MQTT transfer to the central unit, therefore guaranteeing that all pertinent environmental data is sent for centralized processing.

It is worth mentioning that LU subscribes to related topics (e.g., action_room1 and manual_mode_room1) and publishes sensor data to topics like room1. Automatic reconnection is also handled if the MQTT connection is lost. The loop operation of the ESP32 constantly monitors the MQTT connection and, in case of need, reconnects to guarantee continuous data flow.

3.4.3 Local unit fault tolerance handling mechanisms

To guarantee the reliability and robustness of the system, a specialized mechanism has been developed to identify and manage LU failures. Failures in the LU may originate from two principal sources: the sensors failing to supply data or a failure of the ESP32 microcontroller due to missing connectivity or hardware failure. A central unit was developed to identify the two types of errors by monitoring for absent sensor data or a lack of connectivity from the LU. Upon the identification of such issues, the CU isolates the malfunctioning LU, and stops the automatic decision processes associated with that unit. After this point, CU publishes the status of malfunctioning LU to a topic named (node_red/fault_status_room1), which the node-RED already subscribed to, to initiate an alert in the controlling dashboard to allow user intervention

3.5 Cloud central site

One cloud-based programming tool acknowledged as helping IoT systems and services to be developed and integrated is Node-RED [23]. Node-RED offers a scalable and adaptable environment for users to build intricate processes by linking physical devices, APIs, and many internet services as a cloud-native platform. Node-RED's cloud-based architecture lets several users build, run, and administer apps from anywhere with internet access, therefore facilitating real-time access and teamwork. In current distributed systems, where flawless integration between many devices and services is crucial, this feature is especially helpful. Utilizing Node-RED's cloud deployment, scalability is also supported, therefore enabling applications to expand and change with system demands. Node-RED uses the cloud to streamline the creation and upkeep of IoT solutions, thereby providing an easily available and strong tool for field practitioners as well as researchers.

4. RESULTS OF THE PROPOSED SYSTEM

The proposed IPMS prototype is implemented as shown in Figure 7 where three nodes of LUs correspond to a room, so they represent room1, room2, and room3 respectively. Three cases were discussed to verify the real-time data processing for the prototype system as clarified below:

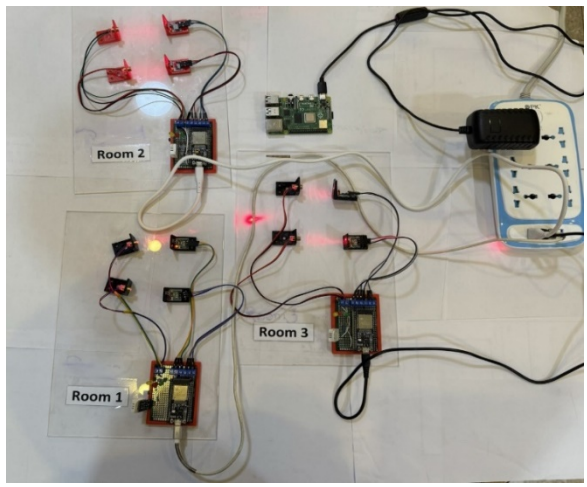

Figure 7. IPMS prototype implementation

4.1 First case study

The first case study represents when there is no occupancy in the room while maintaining the rest of the features.

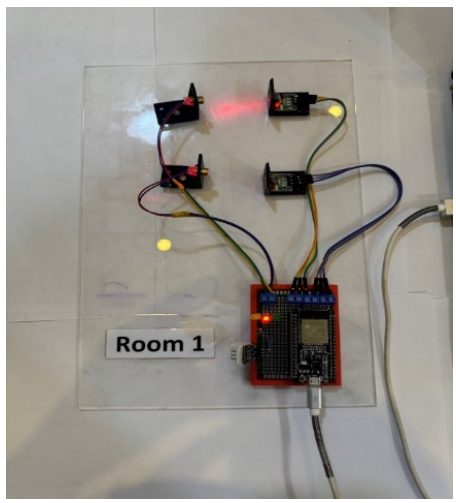

Figure 8. Implementation for LU in room1

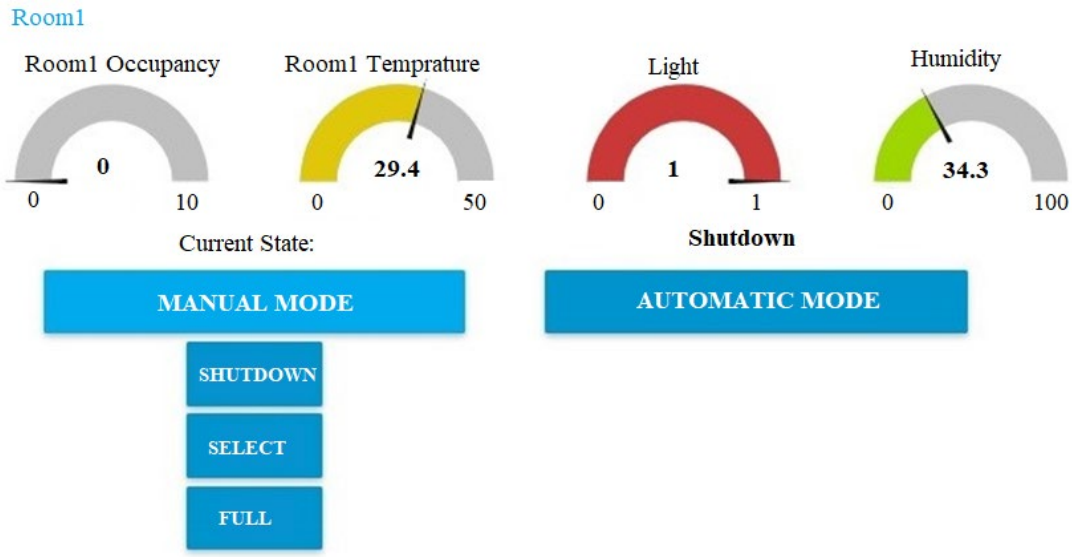

Figure 9. Web application monitoring for room1

Table 1. Serial monitor readings from room 1

Sensor Timestamp	Occupancy	Temperature (℃)	Humidity (%)	Light	Action Timestamp	Action
02:46:34.609	0.00	29.50	49.30	1	02:46:34.609	Shutdown
02:46:39.629	0.00	29.80	59.30	1	02:46:39.583	Shutdown
02:46:44.624	0.00	30.10	65.30	1	02:46:44.624	Shutdown
02:46:49.623	0.00	30.40	68.40	1	02:46:49.623	Shutdown
02:46:54.630	0.00	30.70	68.00	1	02:46:54.630	Shutdown
02:46:59.657	0.00	31.00	67.00	1	02:46:59.610	Shutdown
02:47:04.646	0.00	31.20	64.70	1	02:47:04.646	Shutdown
02:47:09.665	0.00	31.30	61.00	1	02:47:09.665	Shutdown
02:47:14.673	0.00	31.50	52.30	1	02:47:14.673	Shutdown
02:47:19.686	0.00	31.50	47.90	1	02:47:19.686	Shutdown
02:47:24.688	0.00	31.50	46.60	1	02:47:24.688	Shutdown
02:47:29.687	0.00	31.50	44.60	0	02:47:29.687	Shutdown

In this case no matter the parameters acquired from the LUs, the IPMS shut down all the HVAC devices since there is no occupancy at all. This case will be represented as a red LED turned on which indicates the Shutdown mode as shown in Figure 8. Figure 9 shows the Web application monitoring process in Automatic Mode for this room.

Readings are taken every five seconds and we take 120 readings representing 10 minutes. The system accurately predicts the Shutdown mode in all readings. Table 1 implies a window of one-minute serial monitor readings in LU for room 1.

4.2 Second case study

The second case study shows the implementation of the proposed system in the case of Select Mode as Figure 10 implies. LUs will measure the occupancy levels and all parameters and send them to CU for processing. It specifies when the room is partially occupied within the specified temperature range. Hence, there is no need to operate all HVAC devices, and it is represented by a yellow LED. Since the room is not fully occupied, this mode represents an energy-saving approach in the smart system.

Figure 11 indicates the Automatic mode monitoring through the web application while Table 2 shows the serial monitor readings for room 2. From the readings, it could be seen that the system accurately predicts the mode of operation depending on real-time sensor readings but when the temperature drops from 25℃ to 24.80℃ in the time stamp (16:25:14.312) the mode status did not change to shut down in the upcoming reading (16:25:19.314), instead, it cahanges in the second upcoming reading (16:25:24.322) that means it takes approximately ten seconds to transit from one mode to another when specific changes happen. That could be justified due to the predefined delay of 5 seconds between sensor reading intervals.

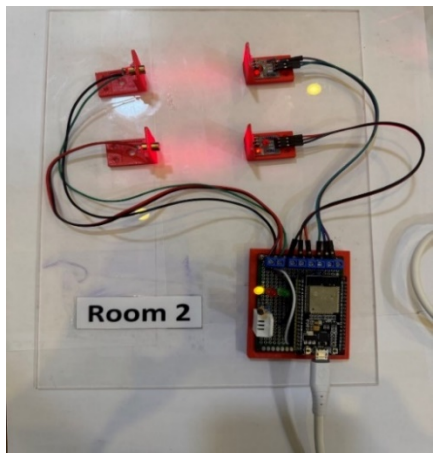

Figure 10. Implementation of room 2

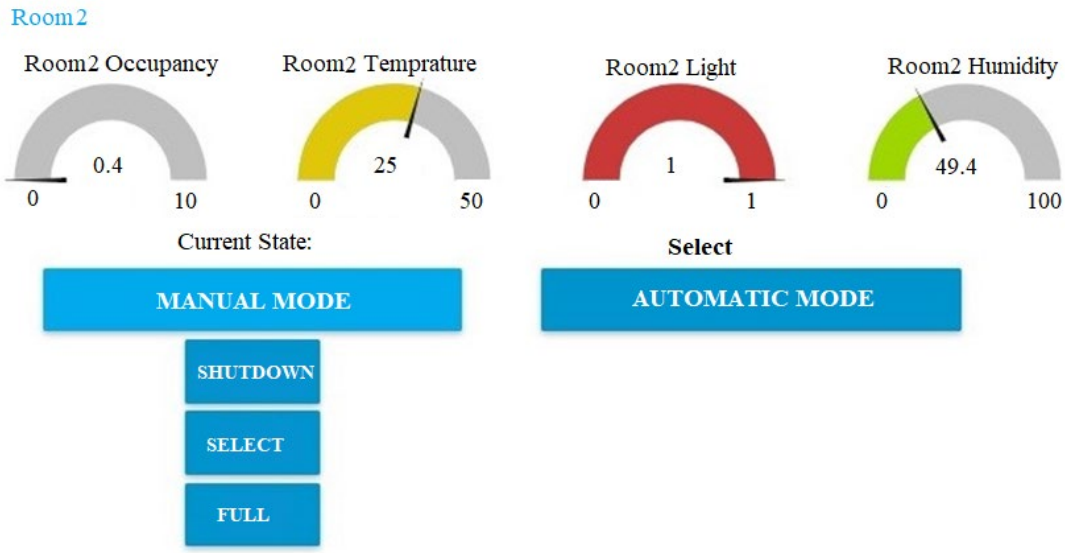

Figure 11. Web application monitoring of room 2

Table 2. Serial monitor readings from room 2

Sensor Timestamp	Occupancy	Temperature (℃)	Humidity (%)	Light	Action Timestamp	Action
16:24:39.247	0.40	26.50	36.30	1	16:24:44.256	Select
16:24:44.256	0.40	26.20	32.40	1	16:24:49.263	Select
16:24:49.263	0.40	25.90	29.80	1	16:24:54.262	Select
16:24:54.262	0.40	25.50	28.20	1	16:24:59.270	Select
16:24:59.270	0.40	25.30	27.30	1	16:25:04.299	Select
16:25:04.299	0.40	25.20	26.70	1	16:25:09.287	Select
16:25:09.287	0.40	25.00	26.60	1	16:25:14.312	Select
16:25:14.312	0.40	24.80	26.40	1	16:25:19.314	Select
16:25:19.314	0.40	24.80	26.30	1	16:25:24.322	Shutdown
16:25:24.322	0.40	24.60	27.00	1	16:25:29.329	Shutdown
16:25:29.329	0.40	24.30	29.70	1	16:25:34.345	Shutdown
16:25:34.345	0.40	24.00	33.50	1	16:25:39.355	Shutdown

Table 3. Serial monitor readings from room 3

Sensor Timestamp	Occupancy	Temperature (℃)	Humidity (%)	Light	Action Timestamp	Action
16:40:58.945	0.50	25.90	41.60	1	16:41:03.949	Full
16:41:03.949	0.50	25.80	41.20	1	16:41:08.955	Full
16:41:08.955	0.50	25.70	41.00	1	16:41:13.976	Full
16:41:13.976	0.50	25.30	39.50	1	16:41:18.996	Full
16:41:18.949	0.50	24.90	38.80	1	16:41:23.978	Full
16:41:23.978	0.50	24.50	38.70	1	16:41:28.966	Shutdown
16:41:28.966	0.50	24.20	42.30	1	16:41:29.005	Shutdown
16:41:29.005	0.50	23.90	46.70	1	16:41:34.009	Shutdown
16:41:34.009	0.50	23.60	50.10	1	16:41:39.013	Shutdown
16:41:39.013	0.50	23.50	52.80	1	16:41:44.020	Shutdown
16:41:44.020	0.50	23.40	54.70	1	16:41:49.035	Shutdown
16:41:49.035	0.50	23.20	55.70	1	16:41:54.026	Shutdown

4.3 Third case study

This case study indicates the operation in Full mode. LUs will measure the occupancy levels and all parameters and send them to CU for processing. When the occupancy and temperature exceed the threshold levels, IPMS will operate all the HVAC devices in this room and a green LED turned on to represent this mode. The implementation for this case and the web application monitoring are shown in Figures 12 and 13 respectively.

Table 3 shows the serial monitor readings of room 3, It shows that the mode will be Full if the occupancy status is beyond the threshold which is more than or equal to 50% of the total room capacity and the temperature is more than or equal to 25C. As we noticed in case study 2, it takes one reading to switch from one mode to another when sensor readings change.

Manual mode could be activated at any time to manually control the system from the node-red web application. As previously mentioned, CU subscribes to topics named (manual_mode_room1, manual_mode_room2, and manual_mode_room3) which correspond to each room respectively where the user can active this mode and operate the LUs remotely to operate manually in one of the three operation modes as shown in Figure 14 which indicates the manual control in room 3.

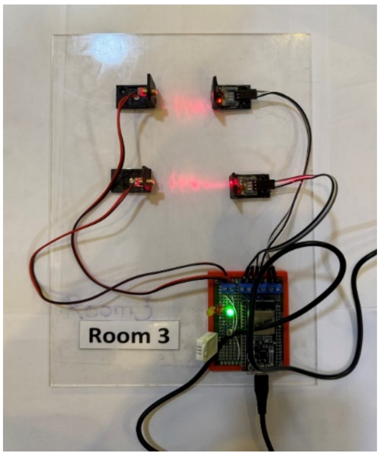

Figure 12. Implementation of room3

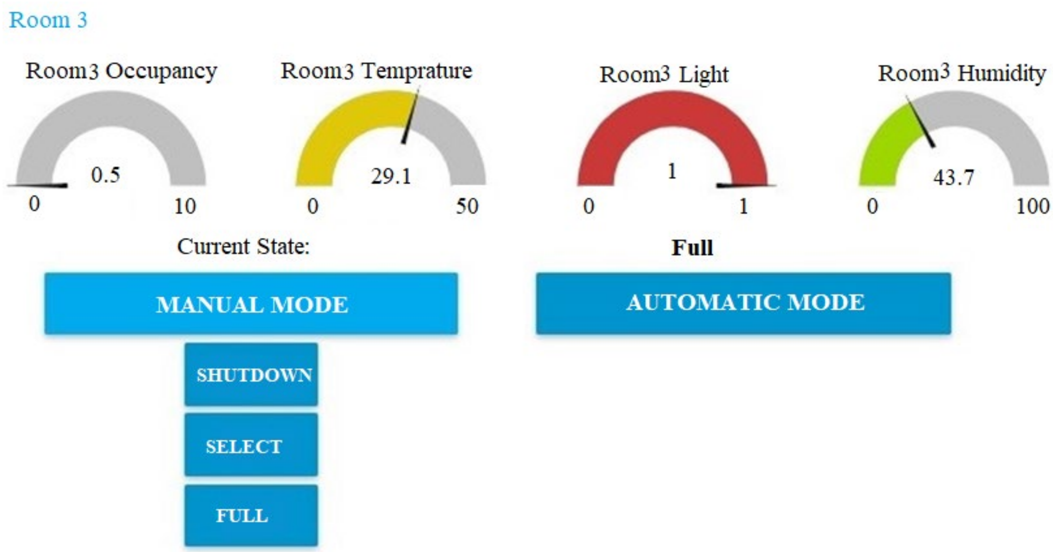

Figure 13. Web application monitoring for room3

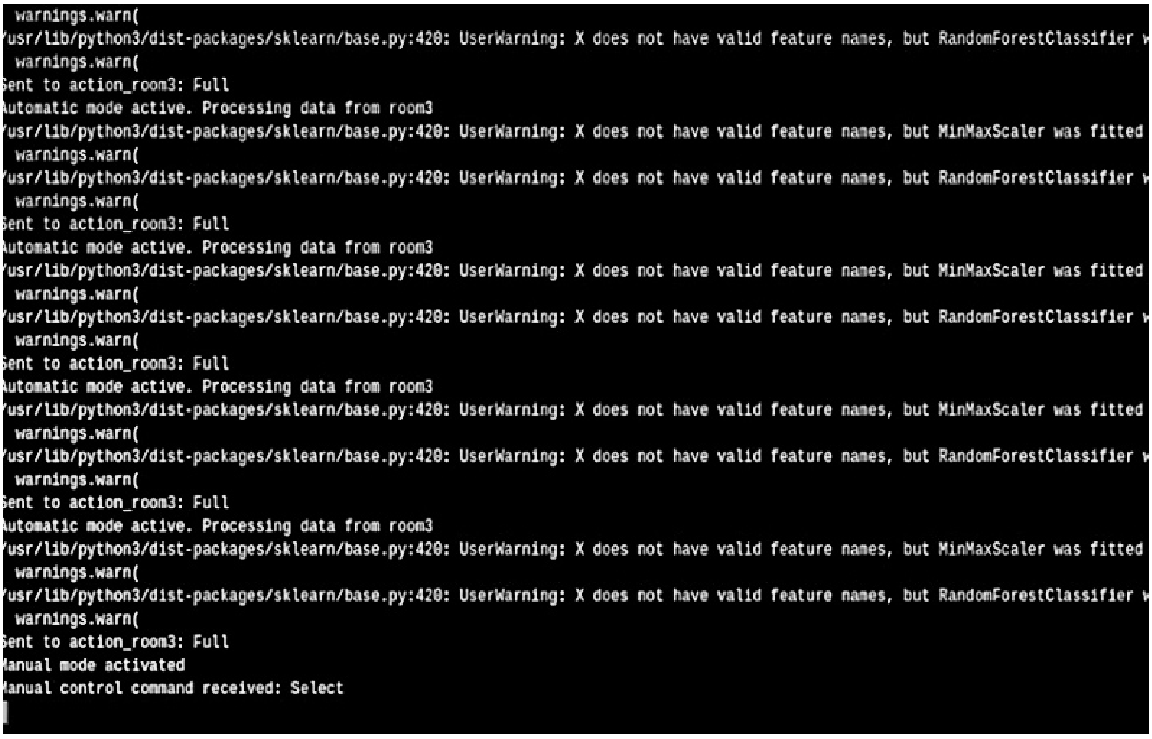

Figure 14. Manual mode

4.4 Fourth case study

Using a mix of hardware, cloud monitoring, and centralized control, three LUs are used concurrently in this case study to monitor and control various rooms within a building. LU1 represents room 1 which is in Shutdown mode LU2 signifies room 2 in Select mode and LU3 denotes room3 in Full mode as shown in Figure 15 which depicts the hardware configuration for the three rooms, thereby highlighting how each room has sensors and controllers linked to an ESP32 microcontroller (LUs), which is in charge of interacting with the CU (Raspberry Pi), it could be seen that each LU operates in different modes as implied by the various LEDs.

Real-time data from every room is presented in Figure 16 which shows the cloud monitoring and control interface using Node-RED. Every room's occupancy, temperature, light, and humidity levels may be viewed by users who can also change between manual and automatic management modes.

Showing the run windows in the CU, Figure 17 shows the execution of scripts matching every room. These scripts find the suitable control action to be transmitted back to the LUs by processing the sensor data using the pre-defined ML model, from the figure it could be noticed that room 1 is in Shutdown mode, room 2 is in Select mode while room 3 is in Full mode.

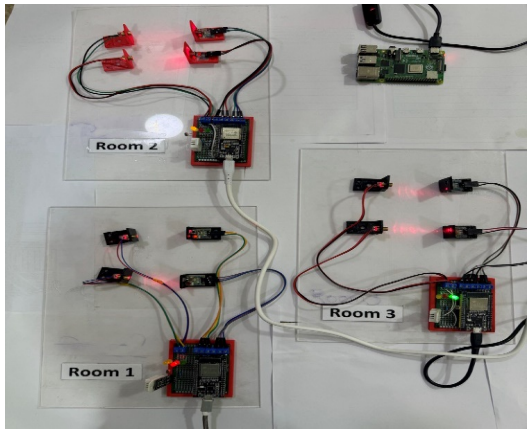

Figure 15. Hardware configuration for the three rooms

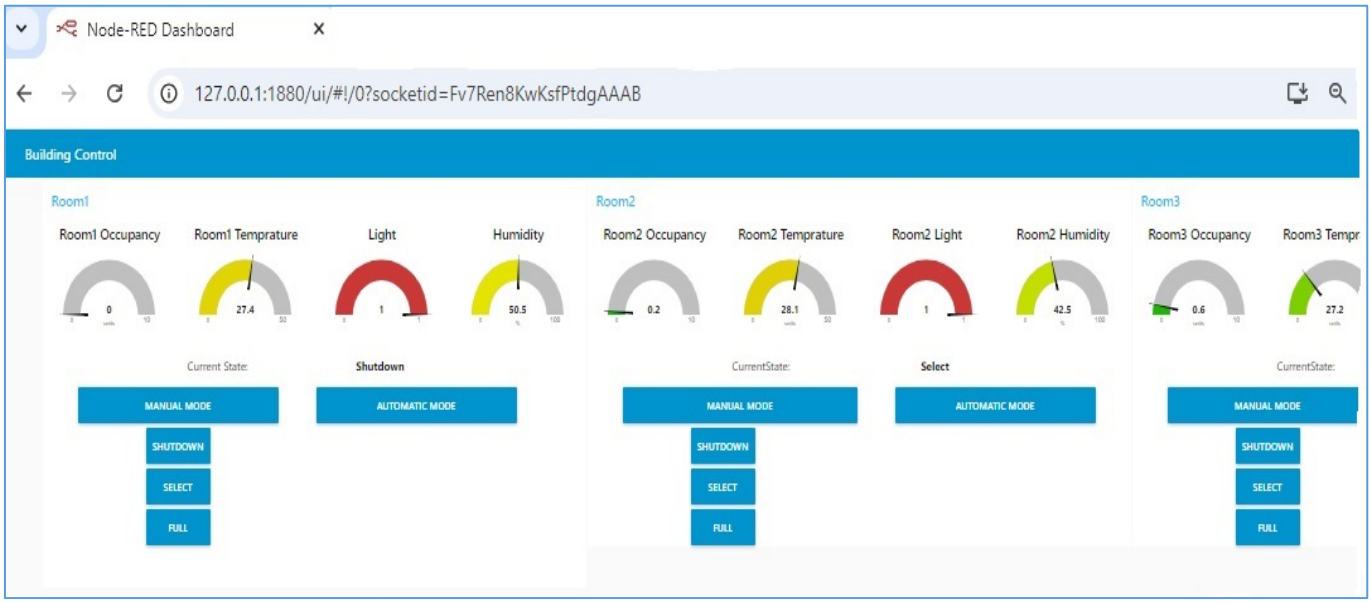

Figure 16. Node-red cloud monitoring and control

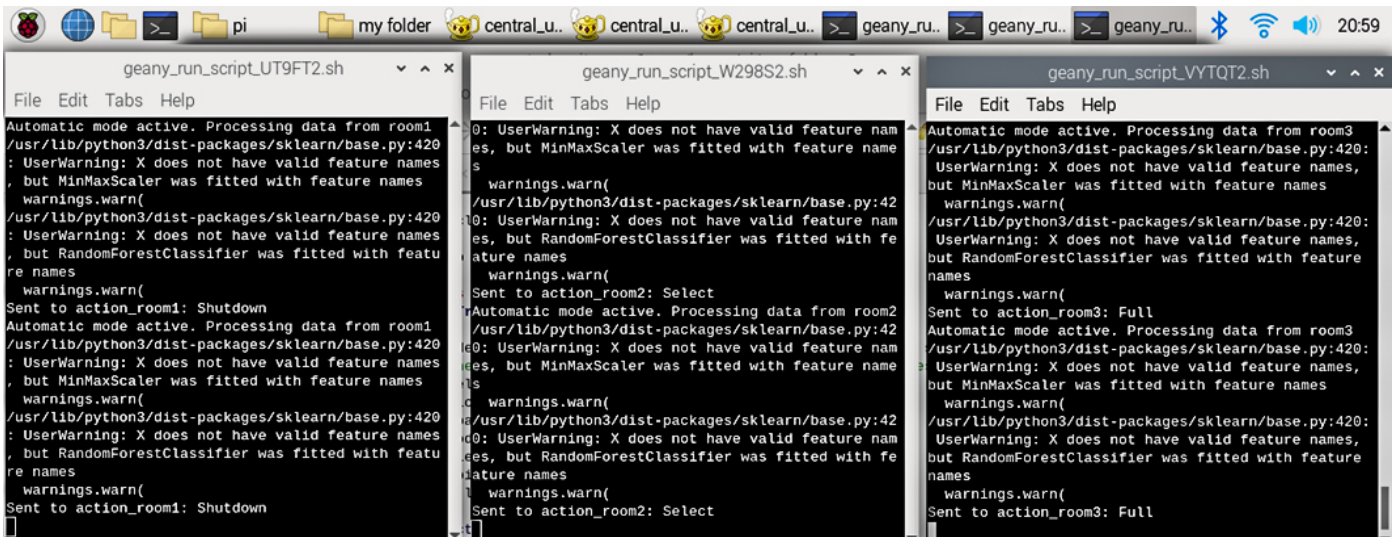

Figure 17. CU run window

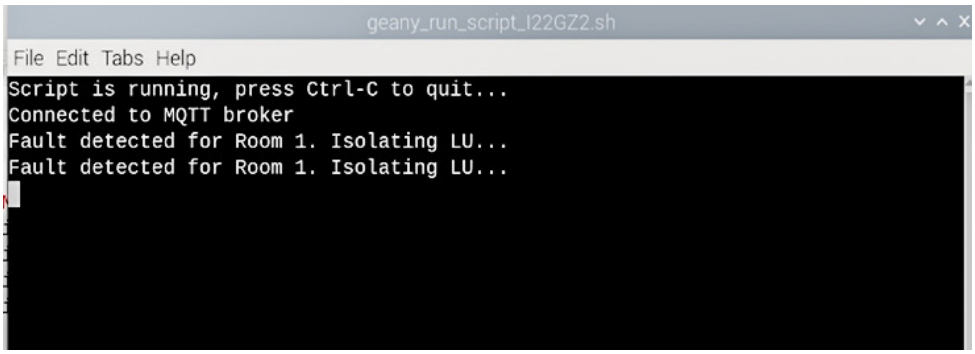

Figure 18. Fault detection in CU

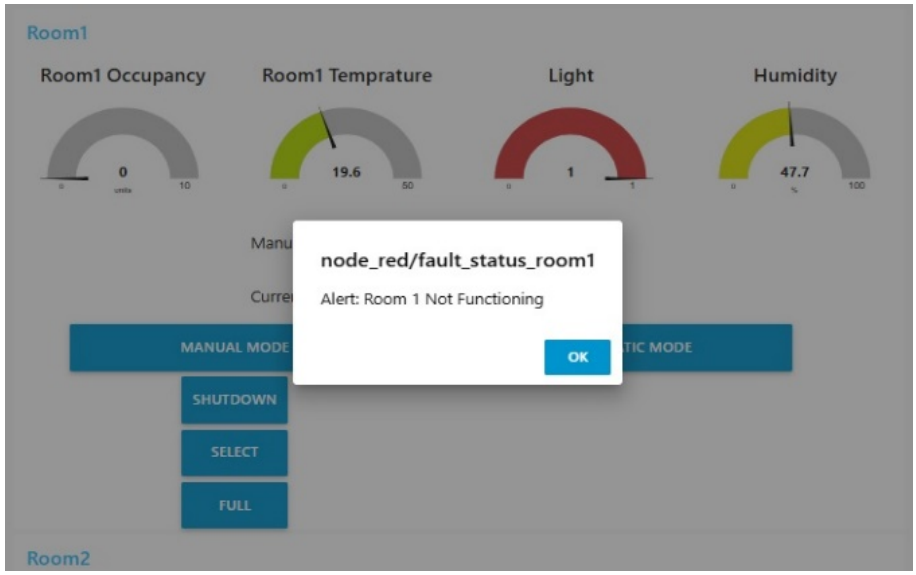

Figure 19. Notification for malfunctioning LU notification for malfunctioning LU

4.5 Fifth case study

The implementation effectively illustrates the system's capability to identify and isolate faults within a Local Unit. Figure 18 illustrates that the CU is responsible for monitoring the status of LUs. Upon detection of a fault, the CU takes action to isolate the faulty LU by stopping decision-making processes associated with that particular unit. Simultaneously, the system provides notifications to the user via the Node-RED dashboard. As illustrated through Figure 19, this notification serves to quickly notify the user that one of the LU is not responding.

4.6 Case studies analysis

For the above-mentioned case studies, a comparison is made between the results of this study and the results obtained from our previous study which investigated the optimal ML method in terms of accuracy [18]. In this work, the system achieved a 100% accuracy rate by showing great accuracy in predicting modes of operation (Shutdown, Select, and Full) in real-time settings. This performance is in line with our previous research, in which the Random Forest (RF) classifier remained robust and also attained 100% accuracy as shown in Figure 20. Although this work mostly concentrates on real-time hardware implementation with efficient processing of sensor data every 5 seconds, the original study provides a more general comparison of machine learning and deep learning models; RF emerges as the most efficient in terms of both accuracy and classification time.

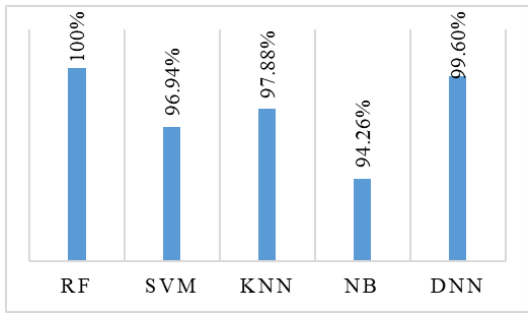

Figure 20. The accuracy results in reference [18]

Moreover, it could be seen from the confusion matrix in Figure 21 that RF 100% accurately predicts all the classes and this is the exact situation in our study where it always gives accurate predictions in real-time measurements. With the RF classifier being a common and ideal option in both systems, both use machine learning to show great efficiency in controlling energy in buildings. This guarantees that, especially in using RF for precise and effective power management, our system is in line with the best practices and results offered in the original study.

To measure the efficiency of the proposed approaches, power consumption is calculated for both Select mode and Shutdown mode, lets first assume that each room has ten 2-ton Air Conditioners (ACs) so the following calculations are done.

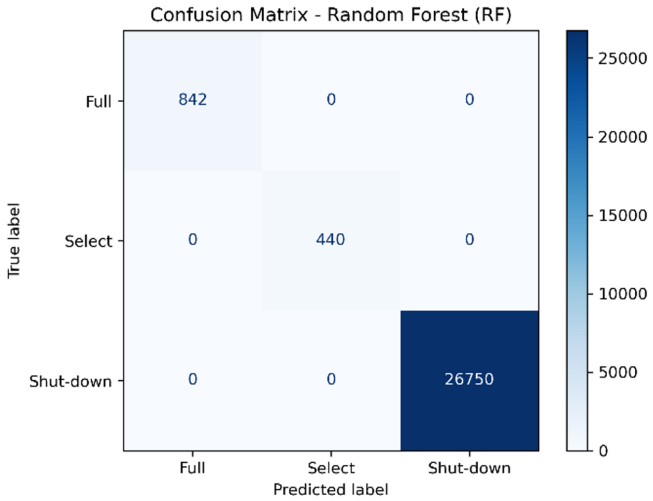

Figure 21. Confusion matrix for RF in reference [11]

Convert tons to BTUs:

The cooling capacity Q_{AC} of an AC unit is measured in tons where 1 ton is equal to 12,000 British Thermal Units per hour (BTU/hr). Hence, for a 2-ton AC unit:

$$Q_{AC=2*12{,}000\ BTU/hr=24{,}000\ BTU/hr}$$

Convert BTUs to watts:

The power consumption P in watts (W) can be calculated by converting BTUs to watts where each 1 BTU / hr = 0.29307107 W. Thus, for a 2-ton AC unit:

$$P_{AC=}Q_{AC} * 0.29307107\text{W} = 24{,}000 * 0.29307107\text{W} \approx 7{,}033.7\text{W}$$

Thus, for a room equipped with n = 10 ACs total power consumption P_{total} is:

$$P_{total} = n * P_{AC} = 10 * 7{,}033.7\text{W} = 70{,}337\text{W} \approx 70.34\text{kW}$$

That means each room needs approximately 70.34 kW, now let's assume when "Select" mode is activated half of the ACs ($n_{Select} = 5$) will operate so the power consumption P_{Select} will be:

$$P_{Select} = n_{Select} * P_{AC=}5 * 7{,}033.7\text{W} = 35{,}168\text{W} \approx 35.17\text{kW}$$

When Shutdown mode is activated none of the ACs will be operated resulting in:

$$P_{Shutdown} = 0\text{W}$$

As a result, whereas the "Shutdown" mode yields a 100% power savings, the "Select" mode reduces power consumption by 50%. This notable lower in energy consumption emphasizes the effectiveness and energy-saving possibilities of using various running modes.

Let's assume the working hours in a building are 8 hr/day, and the AC system runs in Full mode for four hours, in which case all ten of the units are active requiring 70.34 kW of power. The system turns to Select mode for three hours, in which case just half of the AC units are operational, lowering power usage to 35.17 kW. The system is in shutdown mode throughout the last hour, in which case 0% power consumption results from none of the AC units being active. The complete power consumption for every mode for the eight-hour period is broken out in Table 4.

Table 4. Energy consumption

Mode	Duration (hours)	Power Consumption (kW)	Energy Consumption (kW/h)
Full	4	70.34	4 × 70.34 = 281.36
Select	3	35.17	3 × 35.17 = 105.51
Shutdown	1	0	1 × 0 = 0
Total	8	105.51	386.87

From Table 4 it could be inferred if the room runs in Full mode for eight hours, the overall energy consumption may be computed as follows:

$$E_1 = 8 \times 70.34\text{kW} = 562.72\text{kWh}$$

By contrast, the total energy usage in the scenario whereby the system runs four hours in Full mode, three hours in Select mode, and one hour in Shutdown mode is:

$$E_2 = 386.8\text{kWh}$$

We can apply the following calculation for system efficiency under the scenario as opposed to running in Full mode for eight hours:

$$Efficiency\% = \left(\frac{E_1 - E_2}{E_1}\right) \times 100\%$$

Substituting the values:

$$Efficiency\% = \left(\frac{562.72 - 386.87}{562.72}\right) \times 100\% \approx 31.27\%$$

Operating with the mixed mode scenario (Full, Select, Shutdown) generates an energy savings of roughly 31.27% compared to running in Full mode for the whole 8-hour period. This shows that strategic usage of Full, Select, and Shutdown modes can considerably increase energy efficiency, hence lowering the power consumption by 31.27%. This efficiency helps to use more sustainable energy as well as lowers running expenses.

5. CONCLUSION

The deployment of the proposed IPMS shows the potential for substantial energy conservation in building management. Through the utilization of Internet of Things (IoT) technologies and a sophisticated machine learning model, the system effectively monitors and controls the functioning of HVAC systems by analyzing sensor data in real time. The findings from the case studies demonstrate that the IPMS possesses the capability to effectively forecast and transition between operational modes, thus enhancing energy efficiency and minimizing unnecessary power consumption. The incorporation of Node-RED into the system's framework significantly improves its usability and scalability for remote control and monitoring purposes. This study not only offers a pragmatic resolution for the management of energy in intelligent buildings but also establishes the foundation for

upcoming improvements, such as the integration of supplementary sensors for indoor settings to provide exact monitoring and energy economy. Future research will examine how exterior sensors may be integrated to consider the impact of outside environmental variables, thus enhancing the adaptability of the system.

Although the present model makes use of data from one building, future studies will try to include multi-building datasets and the implementation of more sophisticated machine learning models to increase generality and flexibility among several building kinds. These attempts will help to improve the IPMS design so assuring its adaptability and efficiency under many operational environments.

REFERENCES

[1] Ceballos-Fuentealba, I., Álvarez-Miranda, E., Torres-Fuchslocher, C., del Campo-Hitschfeld, M. L., Díaz-Guerrero, J. (2019). A simulation and optimisation methodology for choosing energy efficiency measures in non-residential buildings. Applied Energy, 256: 113953. https://doi.org/10.1016/j.apenergy.2019.113953

[2] Swathika, O.G., Karthikeyan, K., Padmanaban, S. (Eds.). (2022). Smart Buildings Digitalization, Two Volume Set. CRC Press.

[3] Damodaram, D., Godi, R.K., Rao, D.D., Glory, K.B., Somu, K. (2023). Power control management system model using wireless sensor network. Measurement: Sensors, 25: 100639. https://doi.org/10.1016/j.measen.2022.100639

[4] Raed, A.A., Walid, A.A.H., Zubo, R. (2022). IMDC-IST 2021: Proceedings of 2nd International Multi-Disciplinary Conference Theme: Integrated Sciences and Technologies, IMDC-IST 2021, 7-9 September 2021, Sakarya, Turkey. EAI Publishing. https://books.google.iq/books?id=YNVzEAAAQBAJ.

[5] Fernandes, R., Costa, C., Gomes, R., Vilaça, N. (2024). SmartLVEnergy: An AIoT framework for energy management through distributed processing and sensor-actuator integration in legacy low-voltage systems. IEEE Sensors Journal, 24(13): 20726-20741. https://doi.org/10.1109/JSEN.2024.3403484

[6] Kök, İ., Ergun, Y., Uğur, N. (2023). Explainable AI-powered edge computing solution for smart building energy management in green IoT. In Low-Cost Digital Solutions for Industrial Automation (LoDiSA 2023), Cambridge, UK, pp. 150-157. https://doi.org/10.1049/icp.2023.1747

[7] Essa, M.E.S.M., El-shafeey, A.M., Omar, A.H., Fathi, A.E., Maref, A.S.A.E., Lotfy, J.V.W., El-Sayed, M.S. (2023). Reliable Integration of Neural Network and Internet of Things for Forecasting, Controlling, and Monitoring of Experimental Building Management System. Sustainability, 15(3): 2168. https://doi.org/10.3390/su15032168

[8] Saluja, S., Prochnow, C., Couper, G., Zivko, F., Yu, X., Gromov, M., Saniie, J. (2024). Smart multi-building energy monitoring system. In2024 IEEE International Conference on Electro Information Technology (eIT), Eau Claire, WI, USA, pp. 342-347. https://doi.org/10.1109/eIT60633.2024.10609877

[9] Sadeeq, M.A., Zeebaree, S.R. (2023). Design and implementation of an energy management system based on distributed IoT. Computers and Electrical Engineering, 109: 108775. https://doi.org/10.1016/j.compeleceng.2023.108775

[10] Ahmed, S.T., Al-Hamdani, R., Croock, M.S. (2019). Developed third iterative dichotomizer based on feature decisive values for educational data mining. Indonesian Journal of Electrical Engineering and Computer Science, 18(1): 209-217. https://doi.org/10.11591/ijeecs.v18.i1.pp209-217

[11] Espejel-Blanco, D.F., Hoyo-Montano, J.A., Chavez, J.M., Hernandez-Aguirre, F.A., Cruz-Flores, I.A., Valenzuela-Soriano, F.J. (2024). Design of HVAC control system for building energy management systems. In 2024 IEEE Conference on Technologies for Sustainability (SusTech), Portland, OR, USA, pp. 1-5. https://doi.org/10.1109/SusTech60925.2024.10553504

[12] Khudhur, D.D., Croock, M.S. (2021). Physical cyber-security algorithm for wireless sensor networks. TELKOMNIKA (Telecommunication Computing Electronics and Control), 19(4), 1177-1184. https://doi.org/10.12928/telkomnika.v19i4.18464

[13] Gözüoğlu, A., Ozgonenel, O., Gezegin, C. (2024). Modeling and implementation of demand-side energy management system. Sigma Journal of Engineering and Natural Sciences, 42(5): 1628-1645. https://doi.org/10.14744/sigma.2023.00106

[14] Dhaou, I.B. (2023). Design and implementation of an internet-of-things-enabled smart meter and smart plug for home-energy-management system. Electronics, 12(19): 4041. https://doi.org/10.3390/electronics12194041

[15] Khudhur, D.D., Croock, M.S. (2021). Developed security and privacy algorithms for cyber physical system. International Journal of Electrical and Computer Engineering, 11(6): 5379. https://doi.org/10.11591/ijece.v11i6.pp5379-5389

[16] Elkholy, M.H., Elymany, M., Yona, A., Senjyu, T., Takahashi, H., Lotfy, M.E. (2023). Experimental validation of an AI-embedded FPGA-based real-time smart energy management system using multi-objective reptile search algorithm and gorilla troops optimizer. Energy Conversion and Management, 282: 116860. https://doi.org/10.1016/j.enconman.2023.116860

[17] Gozuoglu, A., Ozgonenel, O., Gezegin, C. (2024). CNN-LSTM based deep learning application on jetson nano: estimating electrical energy consumption for future smart homes. Internet of Things, 26: 101148. https://doi.org/10.1016/j.iot.2024.101148

[18] Talib, M.M., Croock, M.S. (2024). Intelligent power management models for buildings: A comparative analysis. Journal Européen des Systèmes Automatisés, 57(1): 95-103. https://doi.org/10.18280/jesa.570110

[19] Dangeti, P. (2017). Statistics for Machine Learning. Packt Publishing Ltd.

[20] Khan, M.Y., Qayoom, A., Nizami, M.S., Siddiqui, M.S., Wasi, S., Raazi, S.M.K.U.R. (2021). Automated Prediction of Good Dictionary EXamples (GDEX): A comprehensive experiment with distant supervision, machine learning, and word embedding-based deep learning techniques. Complexity, 2021(1): 2553199. https://doi.org/10.1155/2021/2553199

[21] Mahdavi, A., Berger, C., Tahmasebi, F., Schuss, M. (2019). Monitored data on occupants' presence and actions in an office building. Scientific Data, 6(1): 290.

https://doi.org/10.1038/s41597-019-0271-7
[22] Ozdemir, S., Susarla, D. (2018). Feature Engineering Made Easy: Identify unique features from your dataset in order to build powerful machine learning systems. Packt Publishing Ltd.
[23] Japón, B.R. (2022). Learn IoT Programming Using Node-RED: Begin to Code Full Stack IoT Apps and Edge Devices with Raspberry Pi, NodeJS, and Grafana. Bpb Publications.