Smart Environmental Management System for Energy Optimization in Malls

Dr. N. Sree Divya[1], Bhavana Deyyam[2], Ch. Laxmi Pravalika[3]
1 Assistant Professor, Mahatma Gandhi Institute of Technology, India
2, 3 UG Student, Mahatma Gandhi Institute of Technology, India

Abstract— A flexible and intelligent environmental management system aimed at enhancing energy efficiency in shopping malls. The system continuously monitors environmental and occupancy conditions using real-time sensor inputs and automates device control accordingly. It employs an Arduino-based microcontroller integrated with IR sensors for people counting, a DHT11 sensor for temperature and humidity monitoring, and an LDR for ambient light detection. LED indicators adjust lighting levels based on occupancy, while fan speed is regulated according to temperature. Additionally, an ESP8266 module enables wireless, real-time monitoring via a web-based dashboard. The system improves overall energy efficiency and promotes sustainable operation in commercial settings by dynamically adjusting resources in response to human activity and environmental factors.

Keywords—*Energy optimization, IoT-based automation, Arduino platform, occupancy-driven control, ESP8266 module, real-time monitoring, smart mall energy management, environmental automation.*

I. INTRODUCTION

Commercial spaces such as shopping malls consume substantial amounts of electricity, primarily due to their extensive requirements for lighting, HVAC systems, and ventilation. Often, this energy is wasted in unoccupied or underutilized zones, driving up operational costs and reducing efficiency. In the era of sustainability and smart infrastructure, there is a growing demand for automated solutions that dynamically manage resource consumption based on real-time occupancy and environmental factors. In this context, we propose an IoT-based automated environmental control system that monitors both human presence and ambient conditions to intelligently manage lighting and ventilation. The system leverages multiple sensors—IR for occupancy detection, DHT11 for temperature and humidity, and an LDR for ambient light levels—processed through an Arduino Uno microcontroller. Device control logic dynamically adjusts lighting (represented by LEDs) and ventilation (simulated by a fan), with system status accessible through a web-based interface provided by an ESP8266 module. This approach aims to optimize energy usage by increasing resource allocation where necessary and conserving energy in unused areas.

II. EASE OF USE

A. Deployment Simplicity

The system is engineered for seamless implementation in both prototype and real-world environments. The sensor modules—IR, DHT11, and LDR—are straightforward to install and require minimal calibration. All components interface with the Arduino Uno via jumper wires on a breadboard, promoting modular assembly and allowing easy component replacement or rearrangement. The modular circuit design ensures that even non-technical personnel can assemble or replicate the setup following basic instructions.

B. Dashboard Accessibility

One of the system's strengths is its user-friendly, web-based dashboard, hosted locally by the ESP8266 module. Any device connected to the same Wi-Fi network can access the dashboard via the module's IP address. Real-time monitoring capabilities include:

- Current occupancy

- Temperature and humidity
- Light intensity
- Status of lighting and fan control

C. Maintenance and Expansion

Component replacement is simple thanks to the plug-and-play assembly. The Arduino's code can be easily modified through the Arduino IDE, allowing the system to evolve with new sensors or integration with cloud services. This future-proof approach ensures long-term usability without requiring significant redesign.

III. SYSTEM DESIGN

A. Functional Overview

The system functions as an intelligent environmental controller that dynamically adjusts lighting and ventilation based on sensor input. The Arduino Uno serves as the core processing unit, receiving data from IR sensors (occupancy), DHT11 (temperature and humidity), and an LDR (light intensity). The system controls LEDs (as lighting indicators) and a DC fan to simulate real-world energy-consuming appliances. An ESP8266 module provides a local dashboard to wirelessly monitor system status in real time [2].

B. Block Diagram

The system is composed of multiple input and output components, connected through a central Arduino Uno. The block diagram below illustrates the interaction between modules:

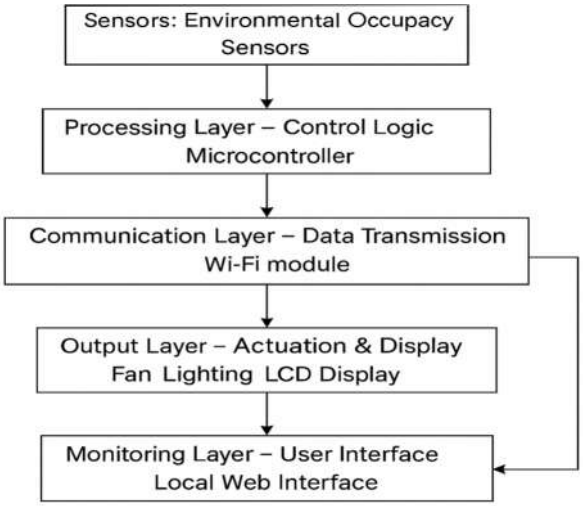

Fig. 1. Block diagram of the automated environmental control system.

The following operational flow is followed:

- IR sensors detect entry and exit to update occupancy count.
- LDR monitors natural light conditions.
- DHT11 tracks room temperature and humidity.
- Arduino Uno processes sensor input to:
  - Control LED count based on occupancy (up to 4 LEDs).
  - Manage fan operation with delayed logic based on temperature.
- ESP8266 wirelessly transmits data to a local dashboard for visualization.

C. Dashboard Visualization

Through the ESP8266's web server, users can view:

- Occupancy count
- Real-time temperature and humidity
- Light intensity levels
- Status of lighting and fan controls

IV. HARDWARE COMPONENTS USED

A. Component List and Description

The system utilizes affordable, easily sourced electronic components for its intelligent environmental control. Each component plays a key role in data acquisition, processing, or actuation. The table below summarizes the components used and their respective purposes.

TABLE I. LIST OF HARDWARE COMPONENTS

Component	***Quantity***	***Purpose***
Arduino Uno	1	Core controller for processing sensor data and controlling outputs
ESP8266 Wi-Fi	1	Provides a local web-based dashboard via Wi-Fi
IR Sensors	2	Track entry/exit and update occupancy count
DHT11 Sensor	1	Measures temperature and humidity
LDR (Light Sensor)	1	Measures ambient light levels.
LEDs	4	Simulate lighting control based on occupancy
DC Fan	1	Simulates ventilation control based on temperature
Relay Module	2	Provides safe switching for LEDs and fan.
Jumper Wires	Multiple	Connects components electrically.
Power Supply (5V)	1	Provides power to modules and sensors

B. Integration Method

All components are mounted on a breadboard for ease of assembly and flexibility. The Arduino Uno interfaces with sensor inputs through its digital and analog pins, while LEDs and fan are controlled via relay modules. Data is communicated to the ESP8266 through serial communication [3]. Inputs from sensors are connected to the digital and analog pins of the Arduino Uno, while outputs (LEDs and fan) are controlled via relays. The ESP8266 communicates with the Arduino using serial communication for data transmission to the dashboard.

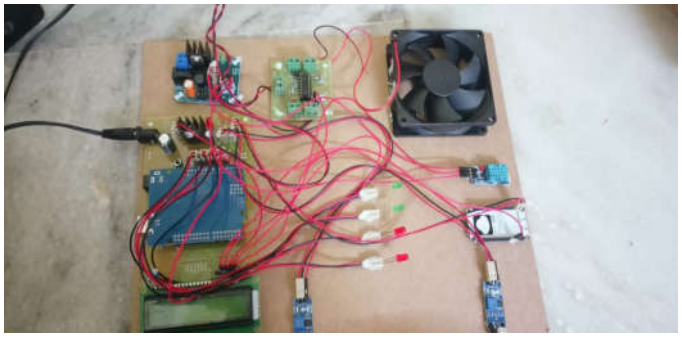

Fig. 2. Physical setup of the system using Arduino and sensors.

V. SOFTWARE TOOLS USED

A. Arduino IDE

The system's software is developed using the Arduino Integrated Development Environment (IDE), which allows for programming, compiling, and uploading embedded C/C++ code to the Arduino Uno. The IDE offers built-in libraries and tools for managing sensor input and output operations, such as reading from the IR, DHT11, and LDR sensors. The serial monitor feature aids in debugging and verifying real-time sensor data during development [4].

B. ESP8266 Firmware and Web Interface

The ESP8266 module is programmed via the Arduino IDE with its corresponding board support package. A lightweight HTML/CSS-based web page is embedded into the ESP8266 code to serve a local dashboard. This dashboard enables real-time monitoring of occupancy, temperature, humidity, light levels, and device status, accessible via any browser on the local network [5].

C. Serial Communication

Communication between the Arduino Uno and the ESP8266 module is established using serial communication protocols (implemented with the SoftwareSerial library). This enables sensor data collected by the Arduino to be transmitted seamlessly to the ESP8266 for real-time display on the web interface [6].

D. Additional Libraries Used

The following libraries were used to enable functionality and simplify development:

- DHT.h for DHT11 temperature and humidity sensor
- SoftwareSerial.h for ESP8266 communication
- ESP8266WiFi.h and ESP8266WebServer.h for dashboard hosting
- Adafruit_Sensor.h for sensor calibration and integration [7].

VI. EXECUTION

A. Sensor Calibration and Data Acquisition

At system startup, all sensors connected to the Arduino are initialized. IR sensors at both entry and exit points detect movement, updating the occupancy count accordingly. The DHT11 sensor gathers temperature and humidity readings at regular intervals, while the LDR continuously monitors ambient light levels.

B. Logic for Lighting Control

The lighting control logic dynamically adjusts LED activation based on the current occupancy count:

- 1 person → 1 LED
- 2 persons → 2 LEDs
- 3 persons → 3 LEDs
- 4 or more persons → 4 LEDs

This logic models real-world adaptive lighting based on occupancy patterns.

C. Fan Control Logic with Delay

Fan operation is governed by a temperature-based control algorithm with built-in delay to prevent rapid toggling:

- Temperature > 30°C → Fan ON (after a 5-second delay)
- Temperature < 28°C → Fan OFF (with delay)

This ensures both energy efficiency and system longevity.

D. Dashboard Display via ESP8266

The ESP8266 acts as a web server, transmitting sensor data from the Arduino and providing a live dashboard. The dashboard displays:

- Current occupancy count
- Temperature and humidity
- Ambient light levels
- Current status of lighting and fan operation

E. Testing

Various test scenarios were conducted:

- Simulated occupancy changes using hand gestures over IR sensors
- Temperature manually varied to test fan control
- Dashboard responsiveness observed to confirm real-time accuracy

VII. RESULTS

A. Lighting Control

The LED lighting system accurately responded to changes in occupancy detected by IR sensors. The dynamic adjustment of lighting based on real-time foot traffic successfully demonstrated efficient energy management.

TABLE II. LED ACTIVATION BASED ON PEOPLE COUNT

People Count	***Number of LEDs ON***
0	0
1	1
2	2
3	3
4 or more	4

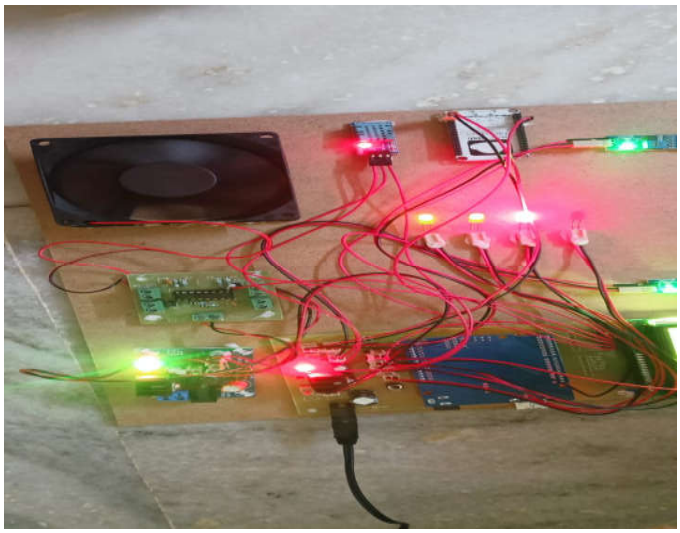

Fig. 3. Hardware implementation of the automated environmental control system.

B. Fan Speed Control

Fan operation followed a multi-level control strategy based on occupancy and temperature:

TABLE III. FAN SPEED LOGIC

People Count	*Temperature (°C)*	*Fan Speed (%)*
1-2	< 28	30
2-3	28-30	60
3-4	30-32	80
>=4	> 32	99

Testing verified smooth and stable fan control transitions based on the system's logic.

C. Dashboard Monitoring

The web dashboard provided real-time visualization of:

- Occupancy
- Temperature and humidity
- Light levels
- LED and fan status

D. Performance

- **System Latency:** < 1 second for sensor-to-output response
- **Occupancy Counting Accuracy:** ~95% under controlled conditions
- **Fan Response:** Stable and jitter-free transitions
- **Dashboard:** Responsive with minimal lag on the local network

VIII. CHALLENGES FACED

A. IR Sensor Alignment and Accuracy

Maintaining accurate occupancy counting was a key challenge, particularly due to occasional misalignment of IR sensors. Variations in positioning or rapid consecutive movements sometimes resulted in incorrect counts. Fine-tuning sensor alignment and testing under various lighting and motion conditions helped improve reliability.

B. Fan Speed Calibration

Implementing multi-level fan speed control via PWM (Pulse Width Modulation) required careful calibration. Smoothing the transition between speed levels (30%, 60%, 80%, 99%) while responding accurately to temperature and occupancy added complexity to the control logic.

C. ESP8266 Stability

During early testing, the ESP8266 module sometimes experienced connectivity issues, particularly when switching between serial data exchange and dashboard hosting. Optimizing baud rates and refining software routines improved stability.

D. Power Supply Constraints

Powering the entire system—including the Arduino, sensors, relays, and ESP8266—introduced occasional voltage drops, causing resets. The issue was resolved by providing a regulated 5V power supply and isolating power lines for sensitive modules.

IX. CONCLUSION

An intelligent, IoT-based environmental control system that dynamically adjusts lighting and ventilation based on real-time occupancy and environmental conditions. Using low-cost components such as Arduino, IR sensors, DHT11, LDR, and ESP8266, the system achieved responsive and scalable energy management for a simulated shopping mall environment. The multi-level fan control, dynamic lighting adjustments, and real-time dashboard provide a strong foundation for sustainable and efficient operation in commercial spaces. The system is easily adaptable and requires minimal manual intervention, making it a promising solution for smart building automation.

X. FUTURE SCOPE

A. Multi-Zone Expansion

Extend system coverage to multiple zones within a mall, with independent control and centralized dashboard monitoring.

B. Integration with Real Fixtures

Adapt system to control actual appliances such as lights, air conditioners, and ventilation fans through relay-based automation.

C. Cloud Based Monitoring

Integrate cloud services for remote monitoring, historical data logging, and predictive analytics.

D. Mobile App Development

Develop a dedicated mobile app for facility managers to provide real-time control and alerts.

E. Machine Learning Integration

Incorporate predictive algorithms to enhance responsiveness and optimize energy consumption based on historical patterns.

ACKNOWLEDGMENT

This Project Survey would be incomplete without the introduction of the people who made it possible and whose guidance, encouragement crowns all efforts with success.

We are thankful to our honorable Prof. G. Chandramohan Reddy Sir, Principal of MGIT and Dr. D. Vijaya Lakshmi, Professor and HOD, Department of IT, MGIT, for providing excellent infrastructure and a conductive atmosphere for completing the project successfully.

We are deeply indebted to our project guide, Dr. N. Sree Divya, Assistant Professor, Department of IT, MGIT, for her constant support, valuable suggestions, immense patience, and expertise throughout the course of our project.

We are profoundly grateful to our project coordinator, Dr. N Sree Divya, Assistant Professor, Department of IT, MGIT, for her unwavering support, and valuable suggestions throughout the course of our project.

REFERENCES

[1] R. K. Rajput, *Basic Electrical and Electronics Engineering*, Laxmi Publications, 2015.

[2] M. Ali, "Internet of Things (IoT): Architecture and Applications," *IEEE Explore*, 2020.

[3] Arduino, "Arduino Uno Board," [Online]. Available: https://www.arduino.cc/en/Main/ArduinoBoardUno.

[4] ESP8266 Community, "ESP8266WiFi and Web Server Library," [Online]. Available: https://arduino-esp8266.readthedocs.io/.

[5] Adafruit, "DHT11 Sensor Library," [Online]. Available: https://github.com/adafruit/DHT-sensor-library.

[6] Random Nerd Tutorials, "ESP8266 Web Server with HTML," [Online]. Available: https://randomnerdtutorials.com/esp8266-web-server/.

[7] N. Sree Divya, Bhavana Deyyam, and CH. Laxmi Pravalika, "Automated Environmental Control for Energy Efficiency in Malls," *International Journal of Research Publication and Reviews*, vol. 6, no. 4, pp. 4074–4086, Apr. 2025. DOI: 10.55248/gengpi.6.0425.1460.