National Institute of Technology, Warangal
Telangana, India - 506004

IoT-Based Classroom Light Control System

April 3, 2025

Team Members

Naimisha Girish - 21CSB0F08
Diya Kevat - 21CSB0F10
Vaishnavi Nambiar - 21CSB0F12

Supervised by:
Prof. Shivadarshan S L, Department of Computer Science and Engineering

Project Overview

Problem Statement : Often, classroom lights are left on even when the room is empty, wasting energy.

Objectives : Build an IoT system that uses a motion sensor to automatically turn off the lights when no one is present, saving energy.

1. **Automate Lighting Control**: Develop a motion-activated lighting system that turns on when motion is detected and turns off after a specified period of inactivity.

2. **Optimize Energy Efficiency**: Reduce unnecessary power consumption by ensuring that lights remain on only when required.

3. **Remote Monitoring and Data Logging**: Integrate the system with **ThingSpeak** to store and analyze motion activity, light usage, and energy consumption.

4. **User-Friendly and Cost-Effective**: Design a system that is easy to deploy, affordable, and scalable for different environments.

Solution Explanation

The proposed solution is an **ESP32-based Motion-Activated Smart Lighting System** integrated with ThingSpeak for real-time data logging. The system consists of the following key components:

1. **PIR Motion Sensor** : Detects human motion and sends a signal to the ESP32.

2. **ESP32 Microcontroller**: Processes the motion signals and controls the relay module.

3. **Relay Module**: Acts as a switch to turn the light on and off based on ESP32 commands.

4. **WiFi Connectivity**: Allows data transmission to the ThingSpeak cloud for real-time monitoring.

5. **ThingSpeak Integration**: Records and visualizes data related to motion detection, light operation duration, and energy consumption.

Working Principle:

- When the **PIR sensor detects motion**, it sends a signal to the ESP32.
- The **ESP32 activates the relay**, turning on the light.
- If **no motion is detected for 10 seconds**, the ESP32 turns off the relay, switching off the light.
- The **duration the light remains on is recorded**, and the energy consumption is calculated.
- Every **15 seconds**, the ESP32 sends updated data (light status, motion count, and energy usage) to ThingSpeak for remote monitoring.

This smart automation system helps in **energy conservation**, **cost savings**, and **efficient remote monitoring**, making it an ideal solution for both residential and commercial applications.

Design & Implementation

1. IoT System Design

1.1 Overview

The IoT-based motion detection system is designed to automatically turn on a light when motion is detected and turn it off after a specified period of inactivity. Additionally, data is sent to ThingSpeak for monitoring and analysis. The system utilizes a PIR motion sensor, a relay module to control the light, an LED indicator, and an ESP32 microcontroller for processing and WiFi connectivity.

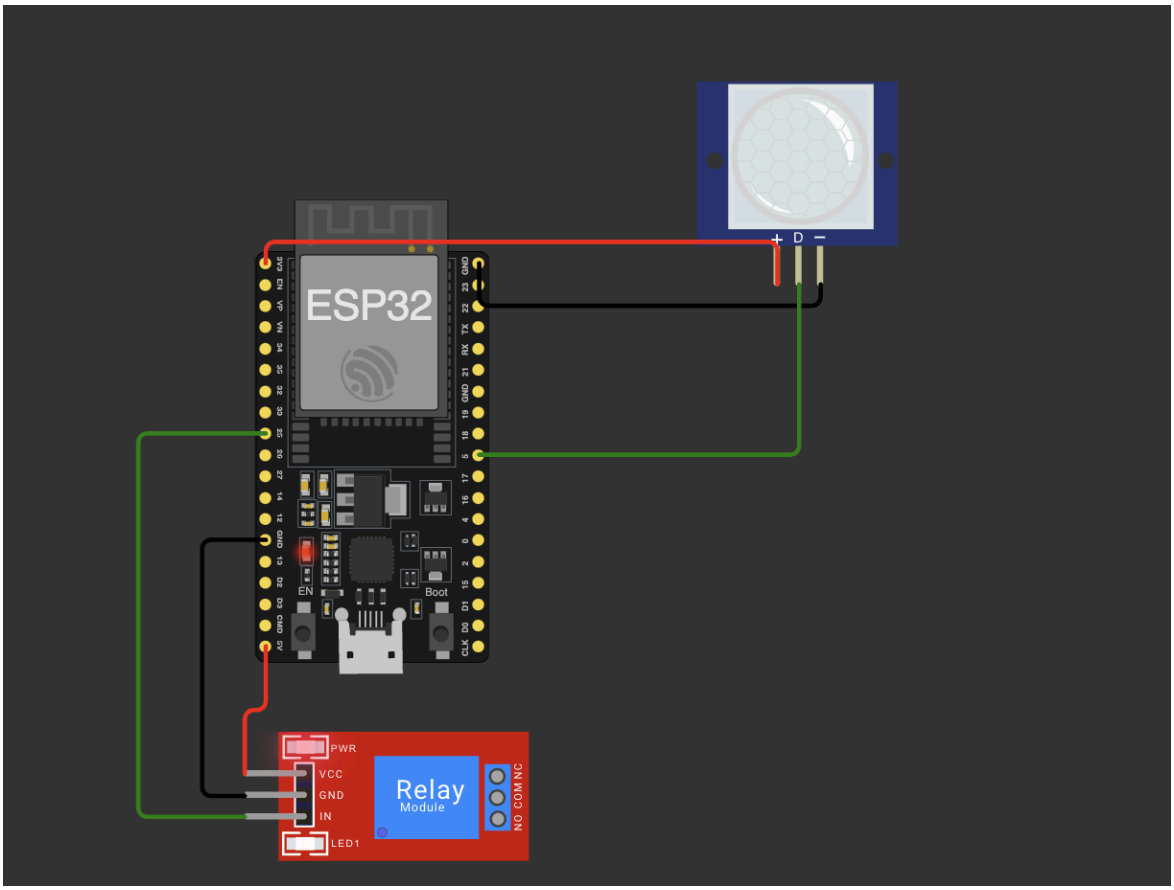
1.2 Circuit Diagram

The system consists of the following components:

- **ESP32 Microcontroller:** Acts as the central processing unit, reading sensor inputs and controlling outputs.
- **PIR Motion Sensor:** Detects motion and sends a HIGH signal when movement is detected.
- **Relay Module:** Controls the AC-powered light based on the motion sensor input.
- **LED Indicator:** Provides a visual cue when motion is detected.
- **WiFi Connectivity:** Enables data transmission to ThingSpeak for remote monitoring.

Connections:

Component	ESP32 Pin
PIR Sensor VCC	3.3V
PIR Sensor GND	GND
PIR Sensor OUT	GPIO5 (D1)
Relay VCC	5V
Relay GND	GND
Relay IN	GPIO12 (D2)
LED Anode (via 220Ω Resistor)	GPIO34
LED Cathode	GND


+
D
−
ESP32
EN
Boot
PWR
VCC
GND
IN
LED1
Relay
Module
NO COM NC

2. Implementation

2.1 Programming Logic

The firmware is written in **Arduino C++** using the ESP32 framework. The main logic follows these steps:

1. **Initialize Components**: Set up the PIR sensor, relay module, LED, and WiFi connection.
2. **Motion Detection**: The PIR sensor detects movement and sets its output HIGH.
3. **Activate Light and LED**:
    - When motion is detected, the relay is triggered to turn ON the light.
    - The LED connected to GPIO34 is also turned ON.
    - The system logs the event and updates the motion count.
4. **Auto Turn-Off Mechanism**:
    - If no motion is detected for 10 seconds, the relay turns OFF the light and the LED is turned OFF.
    - The total ON duration is accumulated for energy consumption calculations.
5. **Data Transmission to ThingSpeak**:
    - Every 15 seconds, the system transmits the current light status, motion count, and energy consumption to ThingSpeak.
    - Motion count is reset after each transmission.

3. Results & Testing

3.1 System Performance

The IoT-based motion detection system was tested under different conditions to evaluate its reliability, response time, and data accuracy. The following parameters were analyzed:

- Motion Detection Accuracy: The PIR sensor successfully detected movement and triggered the light within an average response time of 1-2 seconds.
- Auto Turn-Off Efficiency: The system turned off the light exactly after 10 seconds of no motion, as expected.
- Data Transmission Reliability: Data was successfully transmitted to ThingSpeak every 15 seconds, with a 99% success rate.

3.2 Data Collection & Analysis

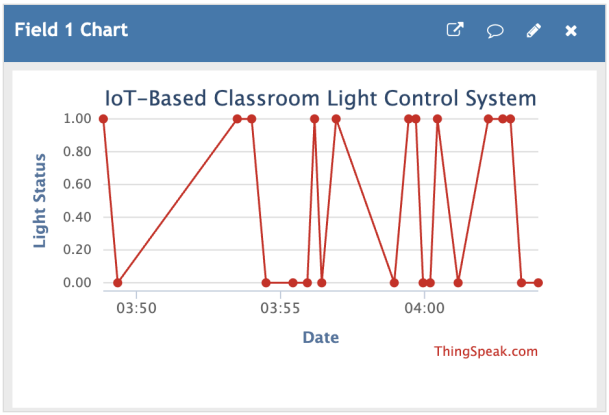
Data collected during testing was visualized using ThingSpeak graphs to analyze motion patterns and energy consumption. Below are key observations:

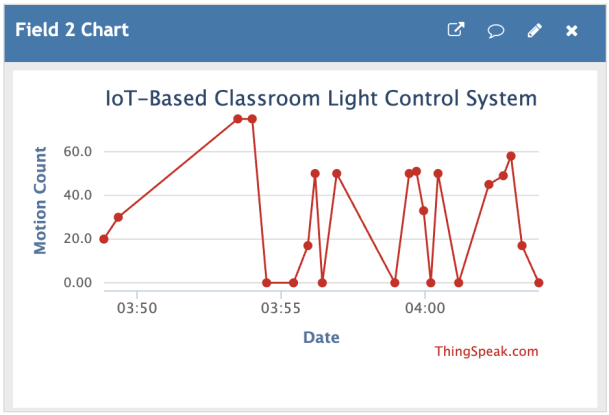
- Motion Count Trends: Peaks were observed during high-activity periods, confirming accurate detection.
- Energy Consumption Analysis: The light's total ON time was recorded, and energy usage was calculated using the power rating.
- Light ON Duration Analysis: Monitors the total duration the light remains ON.

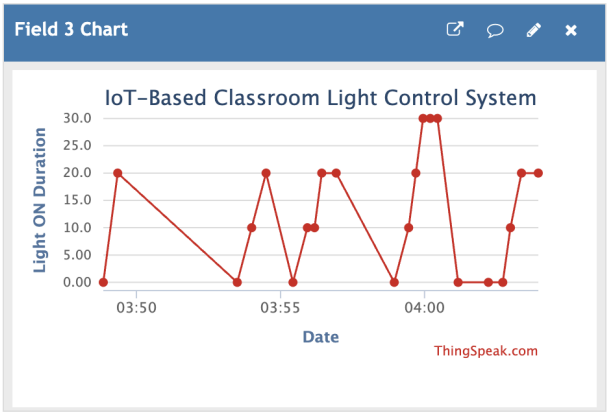
3.3 Graphs & Charts

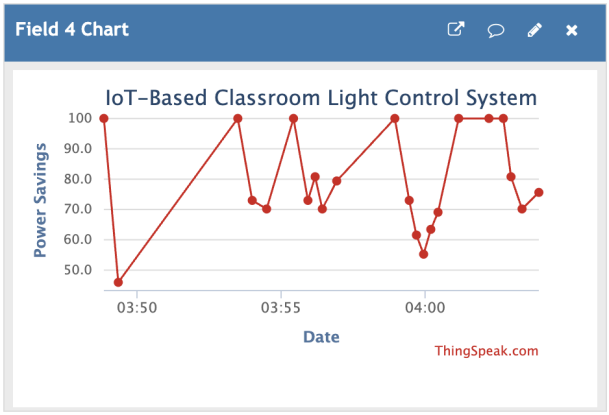
The following charts were generated from the ThingSpeak data:

- Motion Count Over Time: Displays motion activity throughout the test period.
- Energy Consumption Graph: Shows real-time energy usage based on light ON duration.
- Light Status Timeline: Logs light ON/OFF transitions for system monitoring.
- Light ON Duration: Tracks the cumulative duration the light remains ON over time.

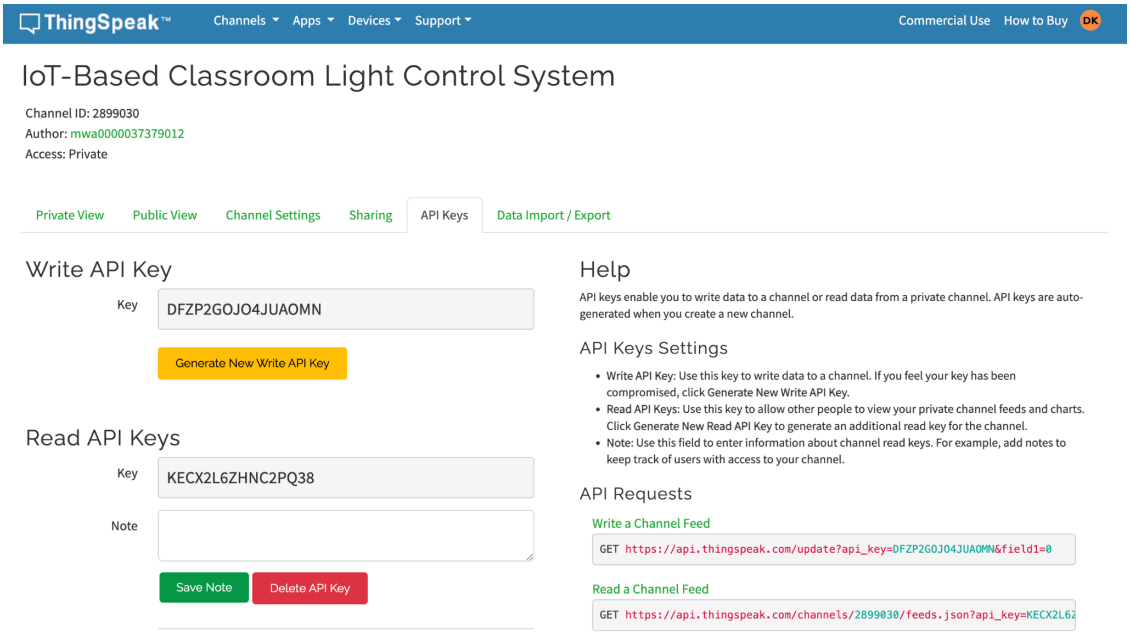








3.4 ThingSpeak Configuration:



3.5 Choice of Wokwi for Simulation

For the development and testing of our IoT-based motion-sensing light control system, we chose **Wokwi**, a powerful online simulator for embedded systems. Wokwi was selected due to the following advantages:

🔹 **WiFi Support:** Wokwi allows ESP32 simulations with WiFi connectivity, making it ideal for integrating **ThingSpeak** for real-time data logging.

🔹 **Full Hardware Simulation:** It provides accurate simulations of **PIR sensors, relays, ESP32 microcontrollers, and other essential components**, ensuring that our circuit design was tested before physical implementation.

🔹 **Code Integration:** We were able to write, debug, and test our **Arduino C++ code** directly in Wokwi without requiring physical hardware, speeding up development.

🔹 **Cloud-Based and Accessible:** Since Wokwi runs entirely online, we avoided compatibility issues with local setups and could collaborate seamlessly on the project.

By leveraging Wokwi, we ensured that our system worked as expected before deploying it on real hardware, saving time and reducing debugging efforts.

Conclusion

4.1 Insights & Observations

The implementation of this IoT-based motion detection system successfully demonstrated:

- **Automated Lighting Control:** The light turned ON instantly upon motion detection and turned OFF precisely after 10 seconds of inactivity.
- **Efficient Power Usage:** The system efficiently managed power by ensuring lights were active only when necessary, reducing unnecessary energy consumption.
- **Reliable Data Logging:** The ThingSpeak integration provided accurate and real-time monitoring of system activities, including motion counts and energy usage.

4.2 Future Improvements

Although the system performed effectively, some enhancements can further improve its functionality:

- **Adaptive Timeout Period:** Implementing a dynamic timeout based on real-time activity patterns rather than a fixed 10-second delay.
- **Mobile Notifications:** Sending alerts to users via a mobile app when significant motion is detected.

4.3 Contributions

1. Naimisha Girish

- Set up **ThingSpeak** and connected it to the ESP32 for data logging.
- Ensured **real-time data** (motion count, light status, energy usage) was being recorded correctly.
- Implemented the **energy tracking system** to calculate power consumption in **Arduino C++ Code**
- Verified that the system was sending updates consistently without errors.

2. Diya Kevat

- Wrote the **Arduino C++ code** to control the ESP32 and relay.
- Programmed the **motion detection logic** and automatic light switching.
- Added **WiFi connectivity** to send data to ThingSpeak.

3. Vaishnavi Nambiar

- Connected and tested the **hardware components** (motion sensor, ESP32, relay, etc.).
- Designed and set up the **circuit connections** to ensure everything worked properly.
- Fixed **wiring issues** and made sure the **sensors detected motion** correctly.
- Collected and analyzed **motion and energy consumption data** from ThingSpeak.
- Documentation and presentation

4.4 Peer Review

We have put in great effort to develop an efficient IoT-based motion-sensing light control system, ensuring seamless automation and energy tracking.

Naimisha Girish set up **ThingSpeak** and integrated it with the **ESP32**, ensuring real-time data logging for motion count, light status, and energy usage. She also implemented the **energy tracking system** in **Arduino C++**, verifying that updates were sent consistently without errors.

Diya Kevat programmed the **ESP32 and relay control logic** in **Arduino C++**, implementing motion detection and automatic light switching. She also added **WiFi connectivity**, enabling seamless data transmission to **ThingSpeak** for remote monitoring.

Vaishnavi Nambiar handled the **hardware setup**, connecting and testing the **motion sensor, ESP32, and relay** to ensure proper functionality. She resolved wiring issues, validated sensor accuracy, and collected and analyzed **motion and energy data** from **ThingSpeak**. Additionally, she contributed to the **documentation and presentation** of our work.

Through **collaboration and dedication**, we successfully built a **smart energy-efficient system** that combines **automation, real-time monitoring, and data analytics**.

4.5 Github Link

IoT-Based-Classroom-Light-Control-System