UNIVERSITY OF VOCATIONAL TECHNOLOGY

Faculty of Engineering Technology

Department of Electro-Mechanical Technology

EE402040

Internet of Things (IoT)

Smart Home Lighting Control System Using ESP32 IoT Modules.

Project Report

Group Members

Name	**Reg. no**
R.D.B.D MUNASINGHE	: MEC/22/B1/14
N.P.R VITHANAGE	: MEC/22/B1/25

Program	: B.Tech in Mechatronics Technology
Submission Date	: 31st July 2025

Instructed by: Mr. Janith Kasun

Contents

Smart Home Lighting Control System Using ESP32 IoT Modules.

1.Introduction

This project aims to develop a smart home lighting control system using ESP32 IoT modules. Traditional lighting systems are often inefficient and inconvenient, lacking automation and remote control features. By integrating motion and light sensors with ESP32 microcontrollers, this system allows users to monitor and control their home lighting through a Python-based web application. It uses Wi-Fi communication with HTTP protocols to ensure real-time data exchange. The solution is designed to be cost-effective, scalable, and energy-efficient, making smart lighting accessible for everyday use.

2. Objectives & Aim

The goal of this project is to build an IoT-based smart lighting control system that allows users to monitor and control home lighting remotely using a Python-based web application. It enhances energy efficiency and convenience through automation, scheduling, and real-time feedback using ESP32 modules.

- Establishing reliable Wi-Fi communication between ESP32 devices and the web app.
- Utilizing HTTP protocols for efficient, real-time data exchange.
- Structuring control and status data using JSON format for seamless integration.
- Programming the ESP32 modules in C++ to handle sensor inputs, control relays, and communicate with the web app.
- Developing a user-friendly Python web dashboard that allows users to turn lights on/off, schedule lighting, and view real-time status updates.

3.Hardware Compornent

Component	**Purpose & Justification**
ESP32 DevKitC (x2)	Central microcontrollers with Wi-Fi, Bluetooth, GPIOs, ADCs, and PWM support; ideal for IoT applications. ESP32 supports C++ programming and multiple communication protocols
Relay Module (e.g., 4 or 8-channel)	To switch high-voltage house lights on/off safely via ESP32 GPIOs, enabling control of standard AC lighting circuits
PIR Motion Sensor	Detects human presence to automate lighting based on occupancy, improving energy efficiency
LDR (Light Dependent Resistor)	Measures ambient light intensity to enable adaptive lighting control (turn lights off when natural light is sufficient)
LED Indicators	Provide visual feedback on system status and manual controls
Power Supply	5V/3.3V regulated supply for ESP32 and sensors; ensures stable operation
Breadboard & Jumper Wires	For prototyping and connecting components without soldering

3.1 Bill of Materials

Component	Quantity	Unit Price (Rs.)	Total Cost (Rs.)
ESP32 DevKitC (x2)	2	1,100	2,200
Relay Module (e.g., 4 or 8-channel)	1	1,000	1,000
PIR Motion Sensor	1	900	900
LDR (Light Dependent Resistor)	1	250	250
LED Indicators	**8**	10	80
Breadboard & Jumper Wires	One Set	800	800

Estimated Total: Rs. 5,230.00

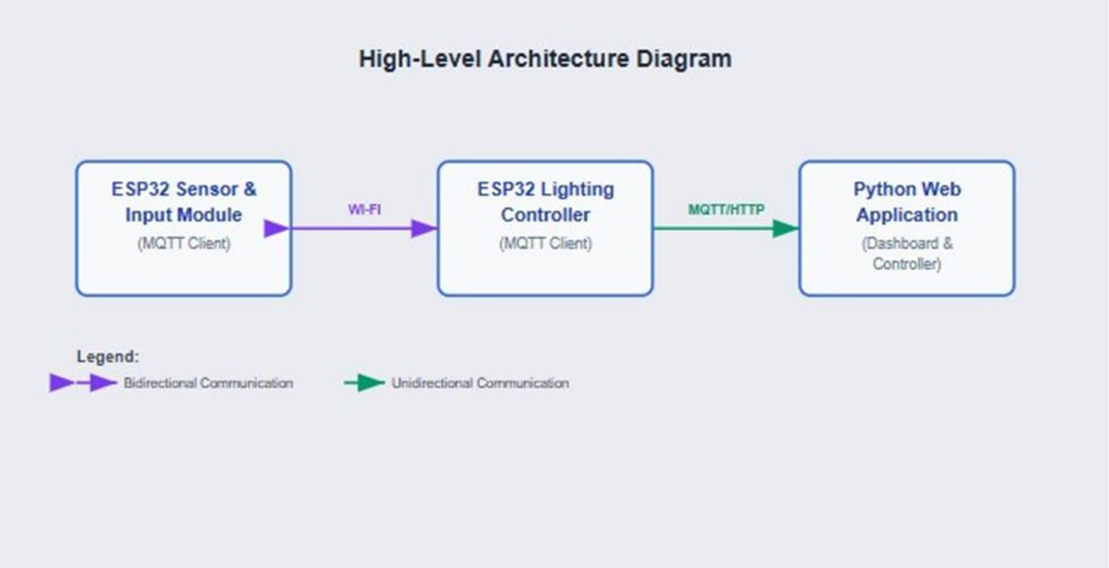
4.Software Design

Functionality:

- Connect ESP32 to Wi-Fi network.
- Initialize sensors (PIR, LDR) and relay outputs.
- Read sensor data periodically.
- Publish sensor data/status via HTTP in JSON format.
- Subscribe to HTTP topics for control commands.
- Parse incoming JSON commands to toggle relays or adjust lighting.
- Optionally, provide HTTP REST endpoints for direct control.

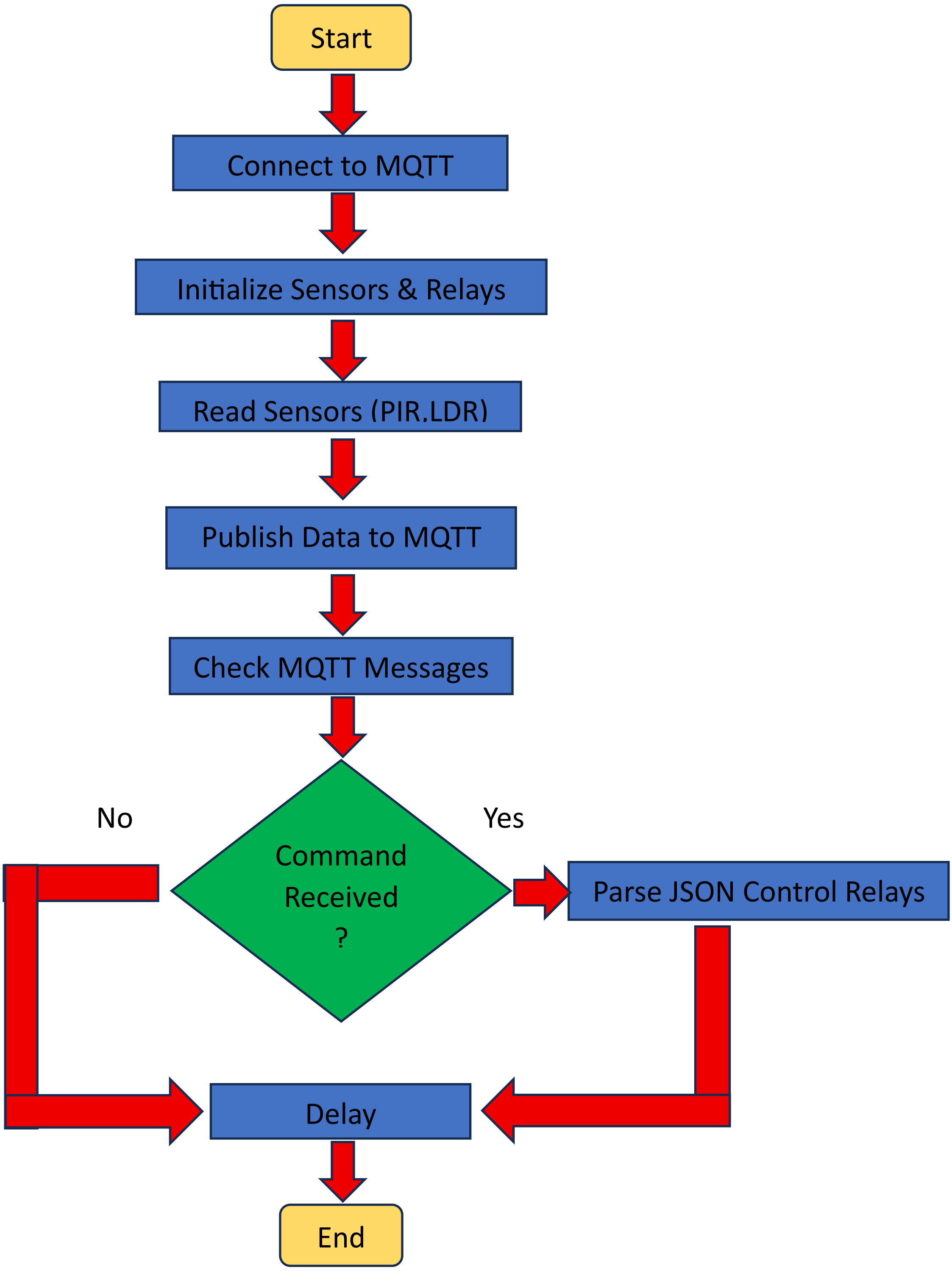
5.System Architecture

The proposed IoT smart lighting control system consists of two ESP32 modules communicating over Wi-Fi using HTTP protocols. One ESP32 acts as the Controller Unit connected to the lighting hardware (relays or LED strips), while the other serves as a Sensor & Input Unit or a secondary controller for distributed control. Both modules connect to a local Wi-Fi network and communicate with a Python-based web application hosted on a server or PC, which provides a user-friendly dashboard for remote monitoring and control.


High-Level Architecture Diagram
ESP32 Sensor & Input Module
(MQTT Client)
WI-FI
ESP32 Lighting Controller
(MQTT Client)
MQTT/HTTP
Python Web Application
(Dashboard & Controller)
Legend:
Bidirectional Communication
Unidirectional Communication

- The ESP32 Sensor & Input Module gathers inputs (e.g., motion, light sensors, buttons).
- The ESP32 Lighting Controller manages the physical lights via relays or LED drivers.
- The Python Web App communicates via MQTT broker or HTTP REST API to send commands and receive status updates.

6.Flow Chart


Start
Connect to MQTT
Initialize Sensors & Relays
Read Sensors (PIR.LDR)
Publish Data to MQTT
Check MQTT Messages
No
Command Received ?
Yes
Parse JSON Control Relays
Delay
End

7.How It Works

Two ESP32 boards are connected to a local Wi-Fi network:

•One acts as the Sensor Module, reading motion and light data using PIR and LDR sensors.
•The other acts as the Lighting Controller, switching relays to turn lights on/off based on commands.

Both communicate with a Python Flask web app using MQTT protocols.
Sensor data is published as JSON via MQTT, and the web dashboard allows real-time light control, scheduling, and feedback.

8.Challenges Faced & Solutions

Challenge	Sollution/Future Plan
Sensor accuracy under varying conditions	Implemented filtering logi and adjustable sensitivity thresholds
Network disconnections	Added retry logic and optional MQTT fallback for critical commands
MQTT integration with Python	Used Paho-MQTT library with error handling and QoS level 1
Ul complexity and responsiveness	Used Bootstrap for responsive design and AJAX for dynamic update
Future Plan	Add voice control, smartphone app, and expand to other appliances

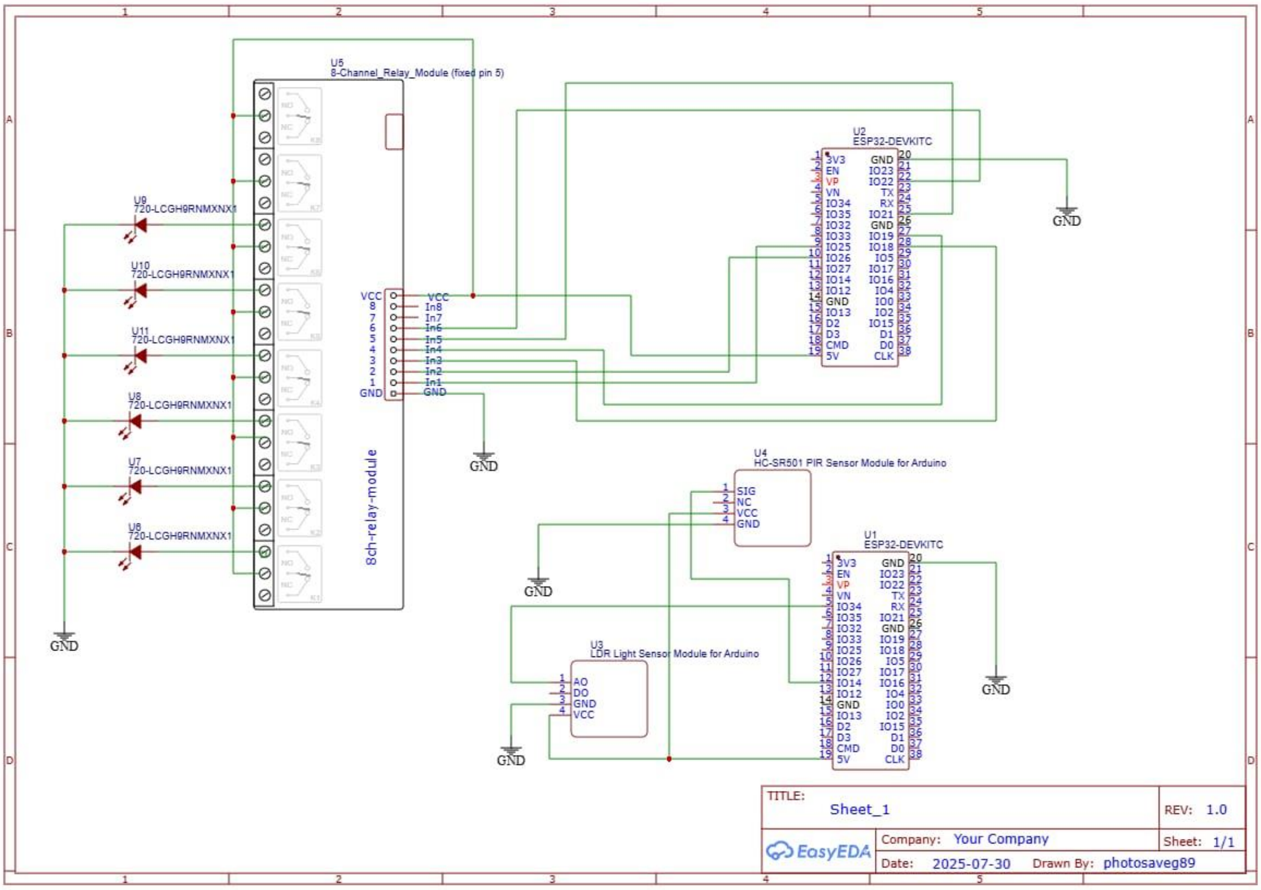
9. YouTube Demo Link

https://youtu.be/flOiRhBpzEk

10.GitHub Link

https://github.com/IOTPROJECT1234/SMART-HOME-LIGHTING-CONTROL-SYSTEM.git

11. Wiring Diagram



12.Reference

1. Project Proposal
2. https://www.youtube.com/
3. https://www.espressif.com/en/products/socs/esp32
4. https://robocraze.com/blogs/post/what-is-the-ldr-sensor?srsltid=AfmBOopFAXXdsNWzbPiK4xeTuolkeCkVZGfFLMkAFuGWVIem6ZxM6Z7i
5. https://learn.adafruit.com/pir-passive-infrared-proximity-motion-sensor/overview