

Part 2

Smart Street light

Table of Contents

Chapter 1	3
Introduction	3
1.1 Overview	3
1.2 Internet of Things (IOT)	3
1.3 Problem Statement	3
1.4 Objectives:	3
1.5 Applications	3
Chapter 2	5
Literature Review	5
Chapter 3	6
System Requirements	6
3.1 Software Requirements	6
3.2 Hardware Requirements	6
Chapter 4	11
Methodology	11
4.1 System Design and Block Diagram	11
4.2 Flow chat of the Program	12
4.3 Circuit Diagram	12
4.4 Code	13
Chapter 5	17
Result and Discussions:	17
5.1 Output	17
5.2 Limitations	17
5.3 Real Time Images	17
Chapter 6	21
Conclusion and Future Enhancement	21
6.1 Conclusion	21
6.2 Future Enhancement	21
References	22

Table of Figure

Fig 1: Micro Controller (ESP32)	6
Fig 2: LDR Module	7
Fig 3: Jumper Wires.....	7
Fig 4: PIR Sensor	8
Fig 5: Bread Board	8
Fig 6: Push Button	9
Fig 7: LED Bulb.....	9
Fig 8: LCD Module	9
Fig 9: Buzzer	10
Fig 10: Block Diagram.....	11
Fig 11: Flow Chart of Program.....	12
Fig 12: Circuit Diagram	12
Fig 13: Definitions and Includes	13
Fig 14: Pin Definition	13
Fig 15: Installation and Setup.....	13
Fig 16: Setup Function.....	14
Fig 17: Main Loop.....	14
Fig 18: Blynk Virtual button Handler	14
Fig 19: Function to Send Sensor Data	15
Fig 20: Function to send Sensor data 2	16
Fig 21: Full Setup.....	17
Fig 22: Bread Board with Buzzer and LDR Module.....	18
Fig 23: Object detected on PIR with blowing Bulb.....	18
Fig 24: LCD Screen	19
Fig 25: ESP32	19
Fig 26: Blynk Dashboard.....	20

Chapter 1

Introduction

1.1 Overview

Street lighting is an essential part of city infrastructure; it provides safety and sight at night by lighting streets and public areas. However, there is a need for creative alternatives because conventional street lighting systems frequently present problems with energy consumption and manual control. To solve these problems and advance urban lighting into the future, this project focuses on putting into practice an IoT-based Smart Street Lighting system.

This project intends to decrease maintenance costs, increase operational efficiency, and reduce energy consumption by utilizing LED lighting systems and IoT technology. The goal is to build a more intelligent, sustainable urban lighting system that enhances public safety by utilizing automation and real-time monitoring.

1.2 Internet of Things (IOT)

The "Internet of Things" (IoT) refers to a network of physical objects equipped with sensors, software, and other technologies to exchange data via the internet. These objects range from industrial machinery and streetlights to home appliances like refrigerators and thermostats. IoT enables remote monitoring and control, reducing human intervention and enhancing efficiency, accuracy, and economic benefits. In smart street lighting, IoT allows for intelligent management, leading to energy savings, improved public safety, and operational efficiency (McKinsey & Company, 2022).

1.3 Problem Statement

Street lighting systems operated manually the old-fashioned way have poor response times and inefficiencies, which raises maintenance costs and restricts flexibility to adjust to changing environmental conditions. Inefficiency and innovation are hampered by a lack of digital integration. Making the switch to IoT-powered automated street lighting systems allows for seamless integration, real-time monitoring, and adaptive control, which improves operational efficiency and urban sustainability.

1.4 Objectives:

The main objectives behind creating smart street light are as below:

- Develop remote control for street lights.
- Improve operational efficiency of street lighting to save energy.
- Integrate emergency push button for instant lighting during emergencies.

1.5 Applications

Smart street lighting systems improve illumination control in a variety of settings, public safety, and energy savings. Adaptive lighting that adjusts to movement is beneficial for suburban regions, city centers, and residential areas since it increases safety and visibility. Commercial areas with lower energy costs and better security include shopping centers and office buildings.

There is increased safety and visibility in public areas like playgrounds, parks, and sports arenas. Campus lighting may be more efficiently managed by schools and colleges to

guarantee well-lit parking lots and walkways. Healthcare facilities have the ability to improve safety in surrounding regions including parking lots. Furthermore, smart lighting systems can be used to provide effective lighting in places without traditional infrastructure, such as isolated or rural locations. All things considered, these technologies provide increased safety, operational efficiency, and energy savings.

1.6 Features:

- Remote control and monitoring
- Energy efficiency using LED lighting
- Automated lighting adjustment based on real-time conditions

Chapter 2

Literature Review

There are currently a lot of concepts for smart street light systems. The some of them which are like this are listed below:

An intelligent street light system has been suggested to reduce a city's energy expenditure by adjusting the street light's brightness in accordance with the volume of traffic during the day (Shahzad et al., 2016).

Weather-responsive smart LED street lighting systems, as outlined to reduce the occurrence of road accidents (Daely et al., 2017).

The proposed intelligent LED streetlight system operates via remote control and utilizes readily available XBee modules to transmit sensor data to a central station (Leccese, 2013).

The idea of minimizing lighting energy consumption under different road conditions was explored in (Mahoor, Salmasi and Najafabadi, 2017). The study proposed three controllers to optimize energy usage. Although fault detection was discussed, the implementation of a robust algorithm or supporting data was lacking.

Chapter 3

System Requirements

3.1 Software Requirements

For creating, deploying, and managing the hardware components of an IoT project, software requirements must be specified. These include the operating system, programming languages, development tools, and libraries. The list of software components that are used in this project are as below:

3.1.1 Aurdino IDE

The Arduino IDE, an Integrated Development Environment, is tailored for programming Arduino microcontroller boards, offering a straightforward interface for coding, compiling, and uploading. It's open-source and compatible with various operating systems like Windows, macOS, and Linux. Utilizing a simplified version of the C++ programming language, it's accessible to both novices and seasoned developers. Additionally, it features a serial monitor, enabling real-time interaction and code debugging with Arduino boards.

3.1.2 Blynk

Blynk is an intuitive Internet of Things platform that allows you to customize remote control interfaces with a drag-and-drop interface. Smartphone apps connect and manage devices like Raspberry Pi and Arduino. It suits developers of all skill levels with interactive features, documentation, and simple integration, making IoT project development easier.

3.1.3 Wokwi

Wokwi is an online platform used for electronics and embedded system simulation. Users can use their web browser to create, model, and test projects like Arduino. With features like code editing, debugging, and collaboration, it enables real-time interaction with virtual hardware and is perfect for electronics education and prototyping.

3.2 Hardware Requirements

To guarantee interoperability, functionality, performance, scalability, dependability, and affordability in Internet of Things applications, hardware requirements are required. The hardware components that are used in this project are as below:

3.2.1 Esp32



Fig 1: Micro Controller (ESP32)

The ESP32 microcontroller, developed by Espressif Systems, features a wide range of pins for various functions. Its GPIO pins (GPIO0 to GPIO39) manage digital communication, sensor interfacing, and device control. Analog pins (GPIO32 to GPIO39) interpret sensor voltage levels. Power and ground pins ensure stable connections, while UART, SPI, and I2C pins enable external device communication. Specialized pins handle bootstrapping, reset, and deep sleep mode. PWM pins generate signals with varying pulse widths, and DAC pins provide precise analog output. Touch sensor pins support capacitive touch interfaces, and RTC pins ensure accurate timekeeping. Understanding the ESP32's pinout is crucial for effective IoT design.

3.2.2 LDR Module

An LDR (Light Dependent Resistor), or photoresistor, decreases in resistance with increased light intensity. Used in street lights, alarms, and exposure meters, it functions in a voltage divider circuit. Despite slow response and temperature sensitivity, LDRs are valued for their simplicity and efficiency in light detection.



Fig 2: LDR Module

3.2.3 Jumper Wires

A jump wire is an electrical wire with connectors or tinned ends, sometimes referred to as a jumper or DuPont wire. They come in different colours and lengths for effortless organization and are available in male-to-male, male-to-female, and female-to-female combinations. Jumpers are primarily used for solderless connections between components on a breadboard or in test circuits.

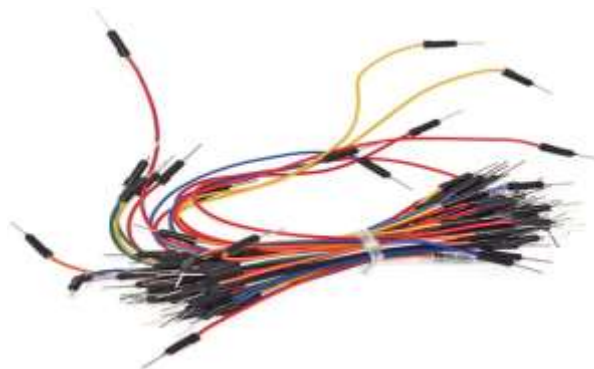


Fig 3: Jumper Wires

3.2.4 PIR Sensor

A PIR (Passive Infrared) sensor detects motion based on temperature variations without emitting radiation by measuring variations in infrared radiation from objects in its field of view (AAI Security Systems, 2015). PIR sensors, which are often found in energy-saving gadgets,

security systems, home automation, and automated doors, are made up of Fresnel lenses and pyroelectric sensors. Although they work well, they may cause pets or other nearby objects to sound a false alarm. Install and adjust them carefully for best results.



Fig 4: PIR Sensor

3.2.5 Bread Board

An electrical circuit can be constructed and tested without the need for soldering using a reusable prototype equipment called a breadboard. Its linked holes on its plastic base make it simple to install and remove components. Widely used for rapid prototyping and educational purposes, breadboards offer convenience and flexibility but are best suited for temporary setups due to their limited durability.

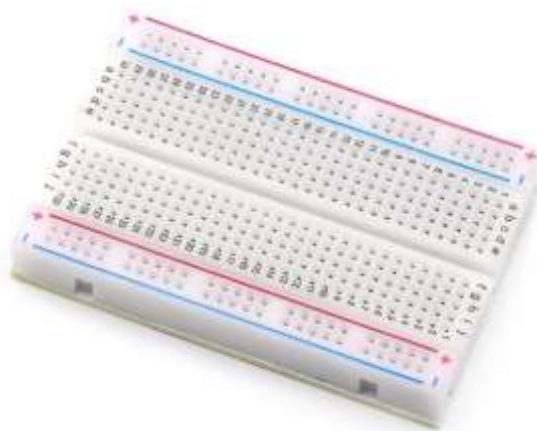


Fig 5: Bread Board

3.2.6 Push Button

A push button, also known as a momentary switch, is a fundamental electrical component that, when depressed, completes a circuit by permitting current to flow and, when released, terminates it. It is frequently utilized for manual control of electronic devices, starting things like lights on or appliances and microcontroller projects.



Fig 6: Push Button

3.2.7 LED Light

A light-emitting diode, or LED, is a semiconductor that, when an electric current flows through it, releases light. It finds widespread use in electronics for indicators, displays, and lighting owing to its energy efficiency, extended lifespan, and the ability to vary in color and size.



Fig 7: LED Bulb

3.2.8 LCD module



Fig 8: LCD Module

Liquid Crystal Display (LCD) modules are integrated into IoT devices to show settings, notifications, and statistics in real time while promoting user involvement. They improve functionality and user experience, and are frequently seen in weather stations and control panels for smart homes.

3.2.9 Buzzer



Fig 9: Buzzer

The electric buzzer generates sound through an electric current passing through its speaker. By adjusting voltage levels and pulse frequencies using digital pins, various tones can be produced. In my project, when a fire flame is detected, the buzzer will sound the alarm.

Chapter 4

Methodology

4.1 System Design and Block Diagram

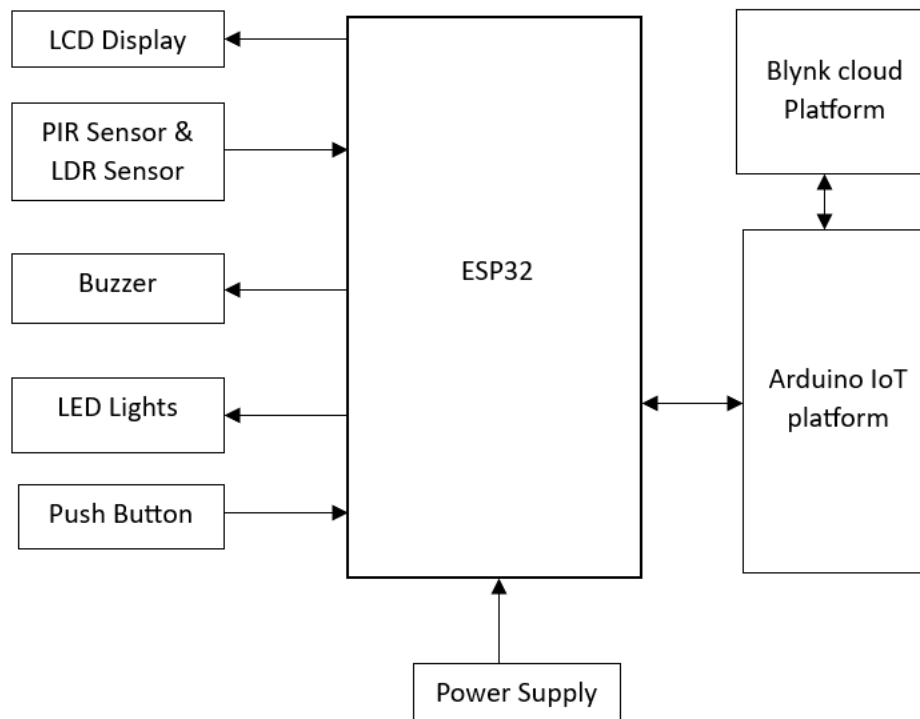


Fig 10: Block Diagram

In the above block diagram, various components such as sensors, an LCD display, an LED light, a PIR sensor, an LDR sensor, and a buzzer are present on the left side. These sensors send data to the ESP32 microcontroller, which processes this data to facilitate the operation of connected devices like LCD display, switches, LED lights, and buzzers according to the implemented logic. The ESP32 continuously sends data to the cloud, enabling real-time monitoring through Blynk app. Users can now remotely operate gadgets thanks to this connection, enhancing capability and convenience of the smart street light system (Ali et al., 2023).

The PIR sensor and LDR sensor play crucial roles in the smart street light's operation. The LDR sensor measures ambient light levels, ensuring that the street light activates only during nighttime. When the LDR sensor detects low light levels, the PIR sensor is enabled to detect motion. If motion is detected, the LED light is turned on and remains on for a preset duration unless further motion is detected. This ensures that the street light is only active when necessary, conserving energy while maintaining functionality. The status and control of these sensors and the LED light are integrated with the Blynk platform, allowing for remote monitoring and adjustments.

Additional components such as the buzzer and LCD display enhance the system's capabilities. The buzzer serves as an alert mechanism, triggered by button, specific conditions like if some accidents happen. To ensure local monitoring, the LCD display offers real-time information on the status of the system, including sensor data and device states. These components are connected to the ESP32, which sends their data to the Blynk cloud for remote monitoring and control. Utilizing the Arduino IoT platform for development and deployment, the smart street

light system combines efficient operation with advanced IoT technologies to offer a robust and user-friendly solution for smart street lighting.

4.2 Flow chat of the Program

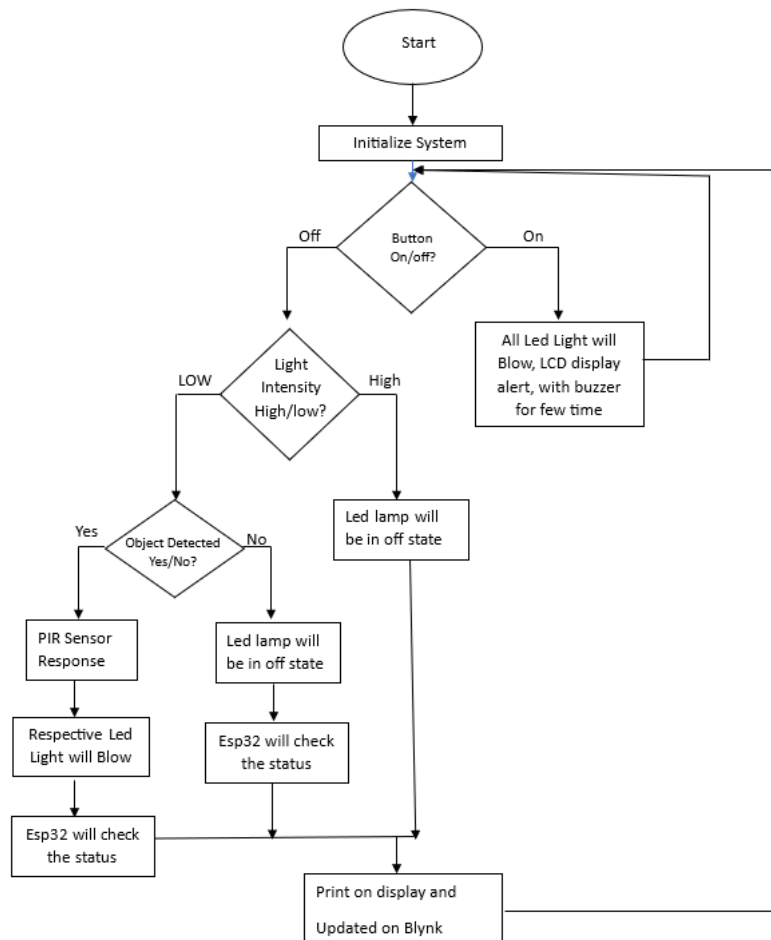


Fig 11: Flow Chart of Program

4.3 Circuit Diagram

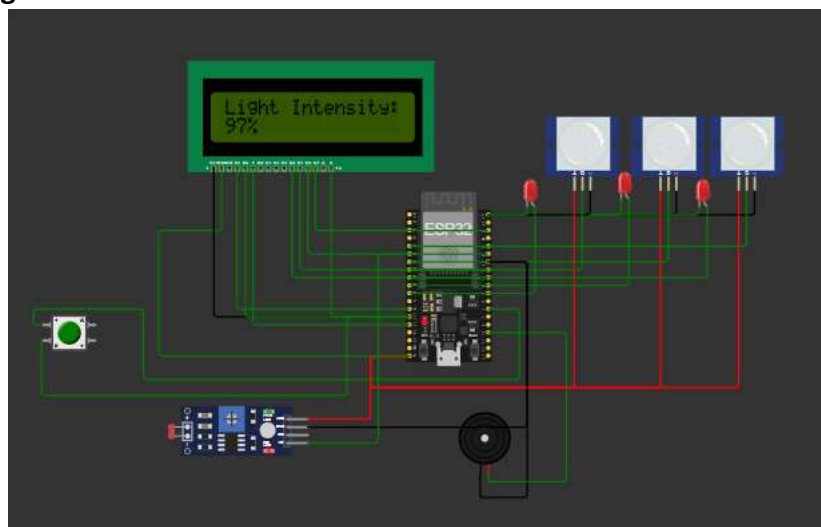


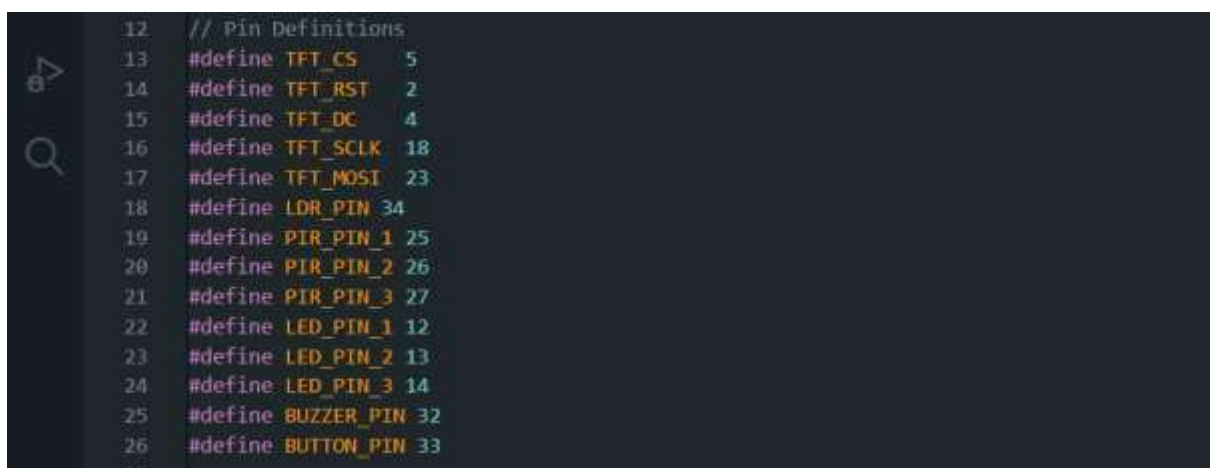
Fig 12: Circuit Diagram

4.4 Code



```
IOT.ino
1  #define BLYNK_TEMPLATE_ID "TMPL5ysohikeg"
2  #define BLYNK_TEMPLATE_NAME "Smart Street Light"
3
4  #include <Arduino.h>
5  #include <SPI.h>
6  #include <Adafruit_GFX.h>
7  #include <Adafruit_ST7789.h>
8  #include <WiFi.h>
9  #include <WiFiClient.h>
10 #include <BlynkSimpleEsp32.h>
```

Fig 13: Definitions and Includes



```
12 // Pin Definitions
13 #define TFT_CS 5
14 #define TFT_RST 2
15 #define TFT_DC 4
16 #define TFT_SCLK 18
17 #define TFT_MOSI 23
18 #define LDR_PIN 34
19 #define PIR_PIN_1 25
20 #define PIR_PIN_2 26
21 #define PIR_PIN_3 27
22 #define LED_PIN_1 12
23 #define LED_PIN_2 13
24 #define LED_PIN_3 14
25 #define BUZZER_PIN 32
26 #define BUTTON_PIN 33
```

Fig 14: Pin Definition



```
28 // Initialize ST7789 display
29 Adafruit_ST7789 tft = Adafruit_ST7789(TFT_CS, TFT_DC, TFT_RST);
30
31 // Blynk Auth Token
32 char auth[] = "PvJ9Myrxu_WIWGto7ZzhVXl3rS5lacdE";
33
34 // Your WiFi credentials
35 char ssid[] = "huwai nova j7";
36 char pass[] = "laxmi@755";
37
38 // Global variables
39 bool buzzerOn = false; // Flag to track the state of the buzzer
40 int prevLightIntensity = 0; // Variable to store the previous light intensity
41
42 BlynkTimer timer;
```

Fig 15: Installation and Setup

```

44 // Blynk virtual pin assignments
45 #define VIRTUAL_BUTTON V0
46 #define VIRTUAL_LED1 V1
47 #define VIRTUAL_LED2 V2
48 #define VIRTUAL_LED3 V3
49 #define VIRTUAL_BUZZER V4
50 #define VIRTUAL_LDR V5
51 #define VIRTUAL_PIR1 V6
52 #define VIRTUAL_PIR2 V7
53 #define VIRTUAL_PIR3 V8
54
55 void setup() {
56     // Initialize Serial communication
57     Serial.begin(115200);
58
59     // Set pin modes
60     pinMode(LDR_PIN, INPUT);
61     pinMode(PIR_PIN_1, INPUT);
62     pinMode(PIR_PIN_2, INPUT);
63     pinMode(PIR_PIN_3, INPUT);
64     pinMode(LED_PIN_1, OUTPUT);
65     pinMode(LED_PIN_2, OUTPUT);
66     pinMode(LED_PIN_3, OUTPUT);
67     pinMode(BUZZER_PIN, OUTPUT);
68
69     // Initialize ST7789 display
70     tft.init(240, 320);
71     tft.setRotation(1);
72     tft.fillScreen(ST77XX_BLACK);
73     tft.setTextSize(2);
74     tft.setTextColor(ST77XX_WHITE);
75
76     // Connect to WiFi and Blynk
77     Blynk.begin(auth, ssid, pass);
78
79     // Setup a function to be called every second
80     timer.setInterval(1000L, sendSensorData);
81 }

```

Fig 16: Setup Function

```

82
83 void loop() {
84     Blynk.run();
85     timer.run();
86 }
87

```

Fig 17: Main Loop

```

87
88 // Blynk virtual button handler
89 BLYNK_WRITE(VIRTUAL_BUTTON) {
90     int buttonState = param.asInt();
91     buzzerOn = (buttonState == 1);
92     if (buzzerOn) {
93         tone(BUZZER_PIN, 1000); // Emit a 1000 Hz tone
94         Serial.println("Buzzer ON");
95         Blynk.virtualWrite(VIRTUAL_BUZZER, 1);
96     } else {
97         noTone(BUZZER_PIN);
98         Serial.println("Buzzer OFF");
99         Blynk.virtualWrite(VIRTUAL_BUZZER, 0);
100     }
101 }

```

Fig 18: Blynk Virtual button Handler

```

103 // Function to send sensor data to Blynk
104 void sendSensorData() {
105     // Read light intensity from LDR pin
106     int lightIntensity = analogRead(LDR_PIN);
107
108     // Send light intensity to Blynk
109     Blynk.virtualWrite(VIRTUAL_LDR, lightIntensity);
110
111     // Display light intensity on LCD if it has changed
112     if (lightIntensity != prevLightIntensity) {
113         // Clear previous light intensity reading
114         tft.fillRect(0, 0, 240, 20, ST77XX_BLACK);
115         // Print new light intensity reading
116         tft.setCursor(0, 0);
117         tft.print("Light Intensity: ");
118         tft.println(lightIntensity);
119         // Update previous light intensity
120         prevLightIntensity = lightIntensity;
121     }
122
123     // Read PIR sensor values
124     bool pirStatus1 = digitalRead(PIR_PIN_1);
125     bool pirStatus2 = digitalRead(PIR_PIN_2);
126     bool pirStatus3 = digitalRead(PIR_PIN_3);
127
128     // Send PIR sensor values to Blynk
129     Blynk.virtualWrite(VIRTUAL_PIR1, pirStatus1);
130     Blynk.virtualWrite(VIRTUAL_PIR2, pirStatus2);
131     Blynk.virtualWrite(VIRTUAL_PIR3, pirStatus3);
132
133     // Display which PIR sensor detects an object
134     tft.fillRect(0, 40, 240, 60, ST77XX_BLACK); // Clear previous PIR sensor status
135     tft.setCursor(0, 40);
136     tft.print("PIR 1: ");
137     tft.println(pirStatus1 ? "DETECTED" : "NOT DETECTED");

```

Fig 19: Function to Send Sensor Data


```

135     tft.setCursor(0, 40);
136     tft.print("PIR 1: ");
137     tft.println(pirStatus1 ? "DETECTED" : "NOT DETECTED");
138     tft.setCursor(0, 60);
139     tft.print("PIR 2: ");
140     tft.println(pirStatus2 ? "DETECTED" : "NOT DETECTED");
141     tft.setCursor(0, 80);
142     tft.print("PIR 3: ");
143     tft.println(pirStatus3 ? "DETECTED" : "NOT DETECTED");
144
145     // Control LEDs through Blynk
146     Blynk.virtualWrite(VIRTUAL_LED1, pirStatus1);
147     Blynk.virtualWrite(VIRTUAL_LED2, pirStatus2);
148     Blynk.virtualWrite(VIRTUAL_LED3, pirStatus3);
149
150     if (buzzerOn) {
151         // Turn on all LEDs if the buzzer is on
152         digitalWrite(LED_PIN_1, HIGH);
153         digitalWrite(LED_PIN_2, HIGH);
154         digitalWrite(LED_PIN_3, HIGH);
155     } else {
156         // Control LEDs based on PIR sensors if the buzzer is off
157         digitalWrite(LED_PIN_1, pirStatus1);
158         digitalWrite(LED_PIN_2, pirStatus2);
159         digitalWrite(LED_PIN_3, pirStatus3);
160     }
161
162     // Clear emergency alert message only if needed
163     if (!buzzerOn) {
164         tft.fillRect(0, 20, 240, 20, ST77XX_BLACK);
165     }
166
167     // Display emergency alert if buzzer is on
168     if (buzzerOn) {
169         tft.setCursor(0, 20);
170         tft.println("EMERGENCY ALERT!");
171     }
172 }

```

Fig 20: Function to send Sensor data 2

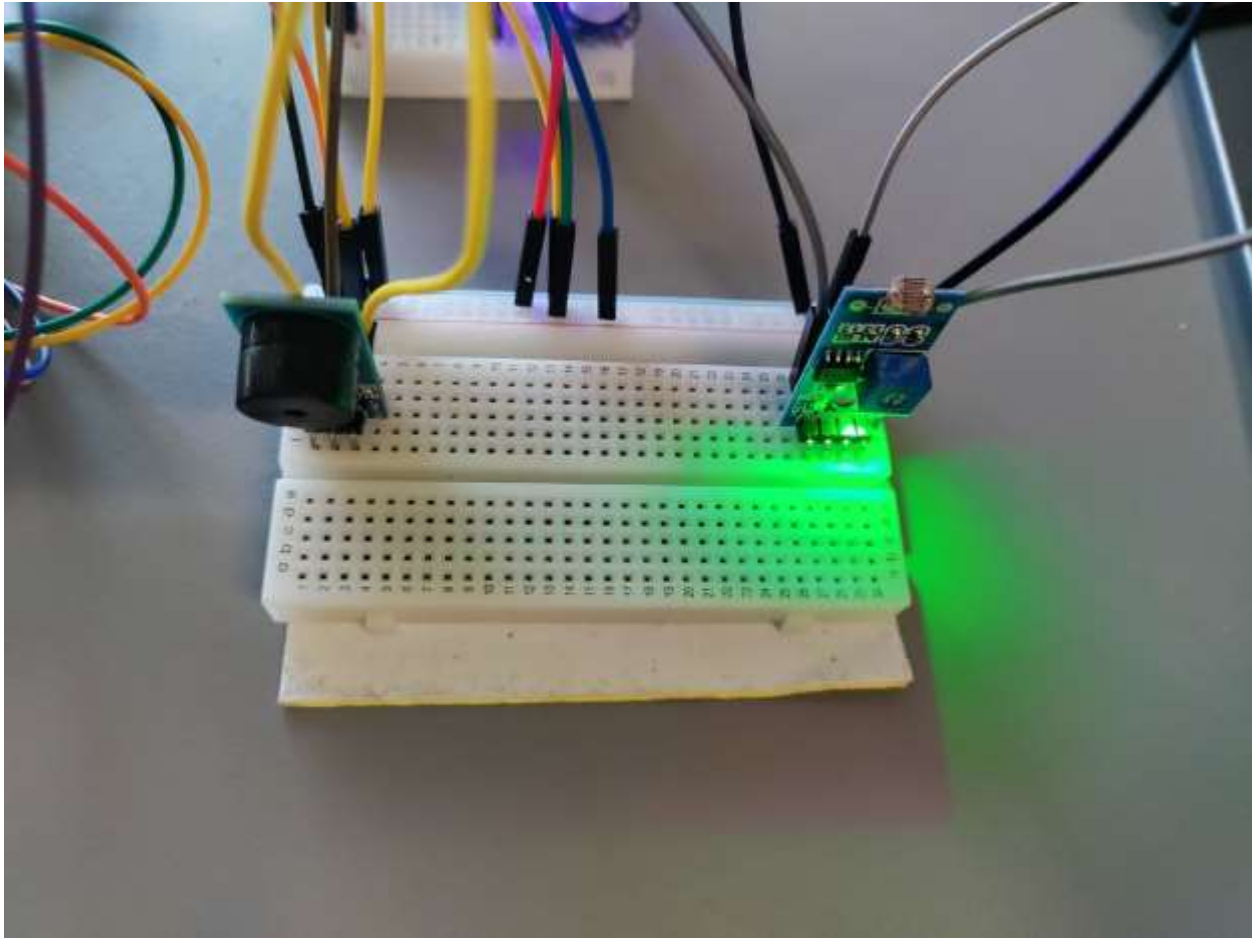


Fig 22: Bread Board with Buzzer and LDR Module

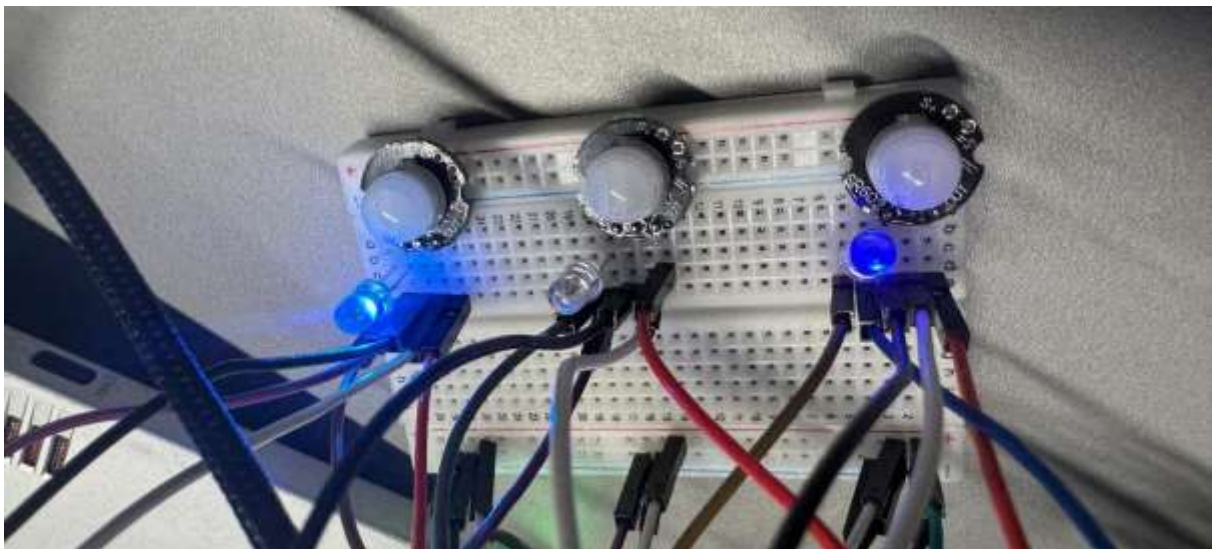


Fig 23: Object detected on PIR with blowing Bulb

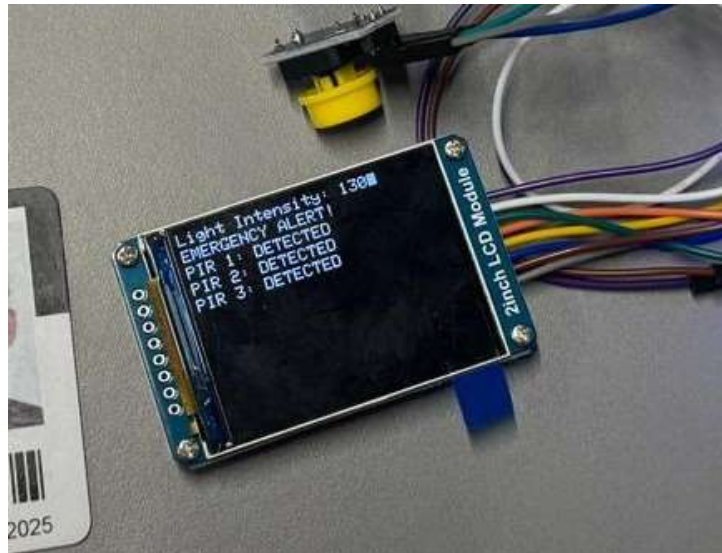


Fig 24: LCD Screen

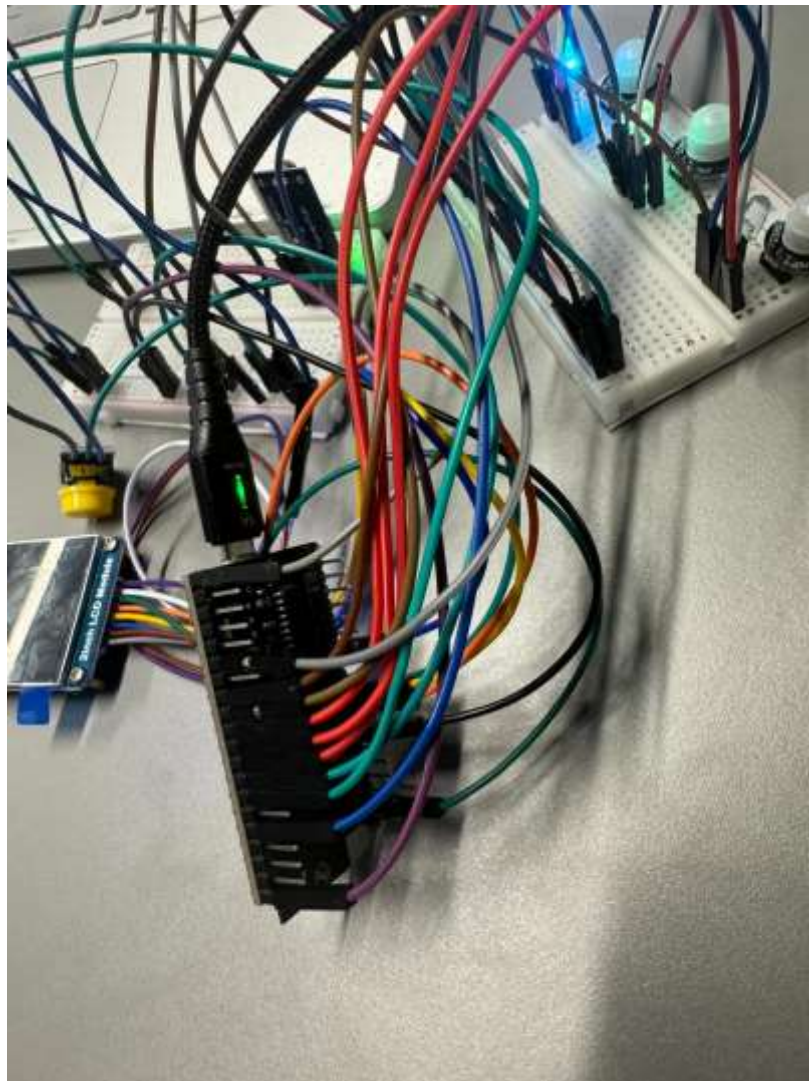


Fig 25: ESP32

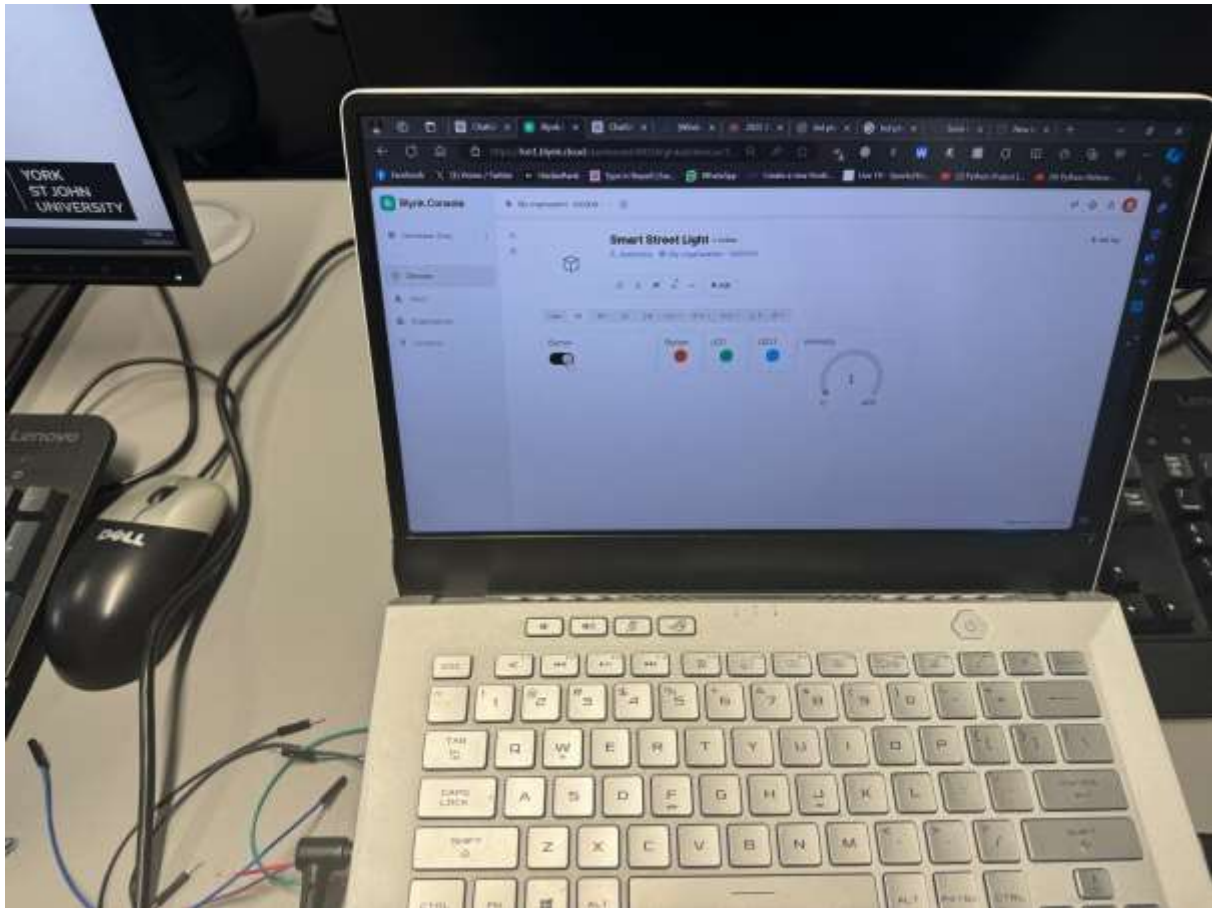


Fig 26: Blynk Dashboard

Chapter 6

Conclusion and Future Enhancement

6.1 Conclusion

The smart street light system integrates sensors, microcontrollers, and cloud connectivity for efficient monitoring and remote control via the Blynk app. While enhancements like a buzzer and LCD display improve functionality, issues such as internet reliability and sensor accuracy in diverse environments need addressing for optimal performance.

6.2 Future Enhancement

To improve energy efficiency, advanced algorithms can dynamically adjust light intensity, aided by instantaneous data examination. Combining machine learning and artificial intelligence enables predictive maintenance and adaptive control. Robust security measures, including encryption and authentication, are crucial for system and user privacy protection against cyber threats, advancing towards a smarter, energy-efficient, and secure urban infrastructure.

References

- McKinsey & Company (2022). What is IoT: The Internet of Things explained | McKinsey. [online] [www.mckinsey.com](https://www.mckinsey.com/featured-insights/mckinsey-explainers/what-is-the-internet-of-things). Available at: <https://www.mckinsey.com/featured-insights/mckinsey-explainers/what-is-the-internet-of-things>.
- Shahzad, G., Yang, H., Ahmad, A.W. and Lee, C. (2016). Energy-Efficient Intelligent Street Lighting System Using Traffic-Adaptive Control. *IEEE Sensors Journal*, 16(13), pp.5397–5405.
- Daely, P.T., Reda, H.T., Satria, G.B., Kim, J.W. and Shin, S.Y. (2017). Design of Smart LED Streetlight System for Smart City with Web-Based Management System. *IEEE Sensors Journal*, 17(18), pp.6100–6110.
- F. Leccese, "Remote-control system of high efficiency and intelligent street lighting using a ZigBee network of devices and sensors", *IEEE Trans. Power Del.*, vol. 28, no. 1, pp. 21-28, Jan. 2013.
- G. Shahzad, H. Yang, A. W. Ahmad and C. Lee, "Energy-efficient intelligent street lighting system using traffic-adaptive control", *IEEE Sensors J.*, vol. 16, no. 13, pp. 5397-5405, Jul 2016.
- AAI Security Systems. (2015). What is a PIR Sensor and How does it Work? - AAI Security. [online] Available at: <https://www.aaisecurity.co.uk/news/pir-sensor/>.
- Mahoor, M., Salmasi, F.R. and Najafabadi, T.A. (2017). A Hierarchical Smart Street Lighting System With Brute-Force Energy Optimization. *IEEE Sensors Journal*, [online] 17(9), pp.2871–2879.
- Ali, M., Scandurra, P., Moretti, F. and Blaso, L. (2023). Architecting a big data-driven software architecture for smart street lighting.