MEASURING LIGHT INTENSITY USING ESP8266 MODULE

An

Internet of Things Course Project Report

in partial fulfilment of the degree

Bachelor of Technology
in
Computer Science & Engineering

By

B. SAI CHARAN	**2203A51L72**
P. CHANDHANA	**2203A51L47**
B. SUSHMITHA	**2203A51L82**
MD. SHAHNAWAZ	**2203A51L87**

Under the guidance of

Dr. Minakshmi Shaw

Assistant Professor

Submitted to

SR UNIVERSITY

DECLARATION

We, B. SAI CHARAN, P. CHANDHANA, B. SUSHMITHA, MD. SHAHNAWAZ hereby declares that the project entitled "MEASURING LIGHT INTENSITY USING ESP8266 MODULE" submitted to SR University, Warangal for the partial fulfillment of the course Internet of Things (21CS116) is a record of bonafide work carried out by me under the supervision of Dr. Minakshmi Shaw, Assistant Professor, School of Computer Science & Artificial Intelligence, SR University, Warangal.

Place: Warangal

Date: 30-04-2024

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

CERTIFICATE

This is to certify that the **Internet of Things** - Course Project Report entitle "**MEASURING LIGHT INTENSITY USING ESP8266 MODULE**" is a record of bonafide work carried out by student B. SAI CHARAN, P. CHANDHANA, B. SUSHMITHA, MD. SHAHNAWAZ bearing Roll No(s) 2203A51L72, 2203A51L47, 2203A51L82, 2203A51L87 during the academic year 2023-24 in partial fulfillment of the award of the degree of ***Bachelor of Technology*** in **Computer Science & Engineering** by the SR University, Ananthsagar, Hasanparthy, Warangal.

Guide	**Head of the Department**
Dr. Minakshmi Shaw	Dr. Sheshikala Martha
Assistant Professor,	Professor & Head,
SR University	SR University
Ananthasagar,	Ananthasagar.
Warangal.	Warangal.

ACKNOWLEDGEMENT

We express our thanks to Course co-coordinator **Dr. Minakshmi Shaw**, Asst. Prof. for guiding us from the beginning through the end of the Course Project. We express our gratitude to Head of the department CS&AI, **Dr. M.Sheshikala**, Professor and Head(CS&AI)for encouragement, support, and insightful suggestions. We truly value their consistent feedback on our progress, which was always constructive and encouraging and ultimately drove us to the right direction.

We wish to take this opportunity to express our sincere gratitude and deep sense of respect to our beloved Dean, School of Computer Science and Artificial Intelligence, **Dr Indrajeet Gupta Sir**, for his continuous support and guidance in completing this project in the institute.

Finally, we express our thanks to all the teaching and non-teaching staff of the department for their suggestions and timely support.

B. SAI CHARAN

P. CHANDHANA

B. SUSHMITHA

MD. SHAHNAWAZ

TABLE OF CONTENTS

ABSTRACT

In this project, we present the development and deployment of a light detection system utilizing a photoresistor integrated with NodeMCU (ESP8266) microcontroller. The primary objective is to accurately measure the intensity of light in the surrounding environment and provide real-time feedback based on the detected light level. The system is designed to offer a cost-effective and efficient solution for various applications including smart home automation, security systems, and environmental monitoring. The hardware architecture comprises a photoresistor as the light sensor and NodeMCU (ESP8266) as the main processing unit. The photoresistor acts as a variable resistor whose resistance changes in response to incident light intensity. This analog signal is then converted to digital data through the ADC (Analog to Digital Converter) module of NodeMCU, allowing precise measurement of light levels. The firmware for NodeMCU is developed using Arduino IDE, facilitating easy programming and integration with the sensor. The firmware includes algorithms for calibrating and processing the sensor data to obtain accurate light intensity readings. Moreover, the system is equipped with Wi-Fi connectivity, enabling remote monitoring and control via a web interface or mobile application. To evaluate the performance of the light detection system, extensive testing is conducted under various lighting conditions and environmental scenarios. The system demonstrates reliable and responsive operation, providing prompt feedback based on the detected light levels. Additionally, the system's versatility allows for customization and integration with other IoT (Internet of Things) devices, enhancing its applicability in diverse settings. Overall, the developed light detection system offers a practical solution for monitoring ambient light levels with precision and efficiency. Its simplicity, affordability, and scalability make it suitable for deployment in both residential and commercial environments, contributing to enhanced automation, security, and environmental awareness.

OBJECTIVE

The objective of the project titled "Measuring Light Intensity Using ESP8266 Wi-Fi Module" is to create a system capable of accurately measuring the intensity of light in its surrounding environment using a light sensor, and transmitting this data wirelessly using an ESP8266 Wi-Fi module. The primary goals of this project may include:

- System Integration: The project aims to integrate a light sensor with an ESP8266 Wi-Fi module into a cohesive system. This involves selecting appropriate components, such as a photodiode or a phototransistor, and interfacing it with the ESP8266 microcontroller board.
- Sensor Calibration: Calibration of the light sensor is crucial to ensure accurate measurement of light intensity levels. The project involves developing methods to calibrate the sensor to various light conditions and ensure consistent and reliable readings.
- Firmware Development: Developing firmware for the ESP8266 module is necessary to enable it to interface with the light sensor, read data from it, and process it for transmission. This may involve programming in Arduino IDE or using other suitable development environments.
- Wireless Communication: The primary objective of the project is to enable wireless communication of the measured light intensity data. This entails configuring the ESP8266 module to establish a Wi-Fi connection and transmit the data to a designated receiver or server.
- Data Transmission: Efficient transmission of data is essential for real-time monitoring or logging of light intensity levels. The project aims to implement robust data transmission protocols to ensure the reliable delivery of information over the wireless network.
- Remote Monitoring: Optionally, the project may include developing a user interface for remote monitoring and visualization of the light intensity data. This could involve creating a web-based dashboard or a mobile application to display real-time or historical data.
- Reliability and Efficiency: Ensuring the system's reliability and efficiency is paramount, particularly if it's intended for continuous monitoring over extended periods. This involves optimizing power consumption, handling network disruptions gracefully, and implementing error detection and correction mechanisms.
- xDocumentation and Reporting: Proper documentation of the project's design, implementation, and testing procedures is essential for future reference and replication. This includes documenting circuit diagrams, firmware code, calibration procedures, and any troubleshooting steps undertaken during the project's development.

DEFINITIONS OF THE ELEMENTS

In Measuring Light Intensity Using ESP8266 module many Hardware Components includes. They are as follows:

1. ESP8266 Wi-Fi module (NodeMCU)
2. Photoresistor or LDR
3. Bread Board
4. LCD Display (16 * 2)
5. LED'S
6. Buzzer
7. Jumper Wires
8. Comparator

Let's start with each module in detail:

1.ESP8266 Wi-Fi module:

The ESP8266 is a highly popular and versatile Wi-Fi module commonly used in IoT (Internet of Things) projects and embedded systems. Here's a brief overview of its key features and capabilities:

- **Wireless Connectivity:** The ESP8266 module provides Wi-Fi connectivity, allowing devices to connect to local wireless networks and the internet. It supports both station (client) and access point (AP) modes, enabling it to connect to existing networks or create its own network.
- **Microcontroller:** In addition to Wi-Fi functionality, the ESP8266 integrates a powerful 32-bit Tensilica microcontroller unit (MCU). This MCU runs at speeds up to 80 MHz and features a low-power sleep mode, making it suitable for battery-powered applications.
- **Memory:** The ESP8266 typically comes with onboard flash memory for storing program code and data. Various models offer different amounts of flash memory, ranging from a few hundred kilobytes to several megabytes.
- **GPIO Pins:** The module features General Purpose Input/Output (GPIO) pins, which can be used to interface with external sensors, actuators, displays, and other peripherals. These pins support digital input/output, analog input (in some variants), and various communication protocols like I2C, SPI, and UART.
- **Programming**: The ESP8266 can be programmed using various development environments, including the Arduino IDE, Espressif's native SDK (ESP-IDF), MicroPython, and others. It's widely supported by a large community of developers and enthusiasts, with plenty of resources, tutorials, and libraries available.

- **Low Cost:** One of the key advantages of the ESP8266 is its affordability. The module is available at relatively low cost, making it an attractive option for hobbyists, makers, and professionals working on IoT projects with budget constraints.
- **Versatility:** Due to its small form factor, low power consumption, and integrated WiFi capabilities, the ESP8266 can be used in a wide range of applications, including home automation, sensor networks, smart devices, industrial monitoring, and more.

What is NodeMCU?

The NodeMCU (Node MicroController Unit) is an open-source software and hardware development environment built around an inexpensive System-on-a-Chip (SoC) called the ESP8266. The ESP8266, designed and manufactured by Espressif Systems, contains the crucial elements of a computer: CPU, RAM, networking (WiFi), and even a modern operating system and SDK. That makes it an excellent choice for Internet of Things (IoT) projects of all kinds.

However, as a chip, the ESP8266 is also hard to access and use. You must solder wires, with the appropriate analog voltage, to its pins for the simplest tasks such as powering it on or sending a keystroke to the “computer” on the chip. You also have to program it in low-level machine instructions that can be interpreted by the chip hardware. This level of integration is not a problem using the ESP8266 as an embedded controller chip in mass-produced electronics. It is a huge burden for hobbyists, hackers, or students who want to experiment with it in their own IoT projects.

But, what about Arduino? The Arduino project created an open-source hardware design and software SDK for their versatile IoT controller. Similar to NodeMCU, the Arduino hardware is a microcontroller board with a USB connector, LED lights, and standard data pins. It also defines standard interfaces to interact with sensors or other boards. But unlike NodeMCU, the Arduino board can have different types of CPU chips (typically an ARM or Intel x86 chip) with memory chips, and a variety of programming environments. There is an Arduino reference design for the ESP8266 chip as well. However, the flexibility of Arduino also means significant variations across different vendors. For example, most Arduino boards do not have WiFi capabilities, and some even have a serial data port instead of a USB port.

NodeMCU is an open source firmware for which open source prototyping board designs are available. The name "NodeMCU" combines "node" and "MCU" (micro-controller unit). Strictly speaking, the term "NodeMCU" refers to the firmware rather than the associated development kits.[citation needed]

Both the firmware and prototyping board designs are open source.
The firmware uses the Lua scripting language. The firmware is based on the eLua project, and built on the Espressif Non-OS SDK for ESP8266. It uses many open source projects, such as lua-cjson and SPIFFS. Due to resource constraints, users need to select the modules relevant for their project and build a firmware tailored to their needs. Support for the 32-bit ESP32 has also been implemented.

The prototyping hardware typically used is a circuit board functioning as a dual in-line package (DIP) which integrates a USB controller with a smaller surface-mounted board containing the MCU and antenna. The choice of the DIP format allows for easy prototyping on breadboards. The design was initially based on the ESP-12 module of the ESP8266, which is a Wi-Fi SoC integrated with a Tensilica Xtensa LX106 core, widely used in IoT applications (see related projects).

Types

There are two available versions of NodeMCU as version 0.9 & 1.0 where the version 0.9 contains ESP-12 and version 1.0 contains ESP-12E where E stands for "Enhanced".

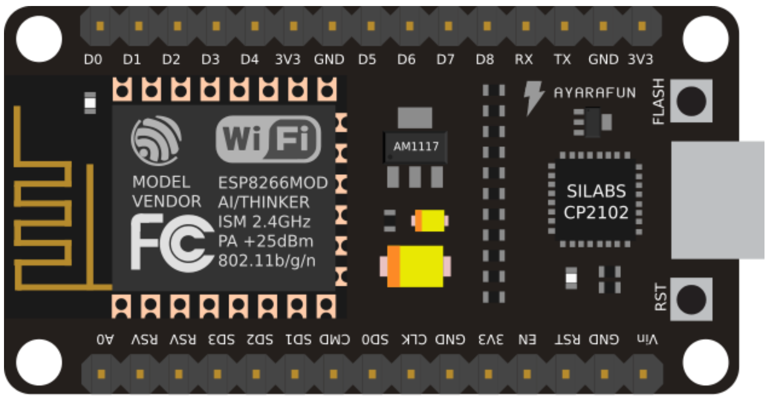
History

NodeMCU was created shortly after the ESP8266 came out. On December 30, 2013, Espressif Systems began production of the ESP8266. NodeMCU started on 13 Oct 2014, when Hong committed the first file of nodemcu-firmware to GitHub. Two months later, the project expanded to include an open-hardware platform when developer Huang R committed the gerber file of an ESP8266 board, named devkit v0.9. Later that month, Tuan PM ported MQTT client library from Contiki to the ESP8266 SoC platform, and committed to NodeMCU project, then NodeMCU was able to support the MQTT IoT protocol, using Lua to access the MQTT broker. Another important update was made on 30 Jan 2015, when Devsaurus ported the u8glib to the NodeMCU project, enabling NodeMCU to easily drive LCD, Screen, OLED, even VGA displays.

NodeMCU Specifications:

The NodeMCU is available in various package styles. Common to all the designs is the base ESP8266 core. Designs based on the architecture have maintained the standard 30-pin layout. Some designs use the more common narrow (0.9″) footprint, while others use a wide (1.1″) footprint – an important consideration to be aware of.

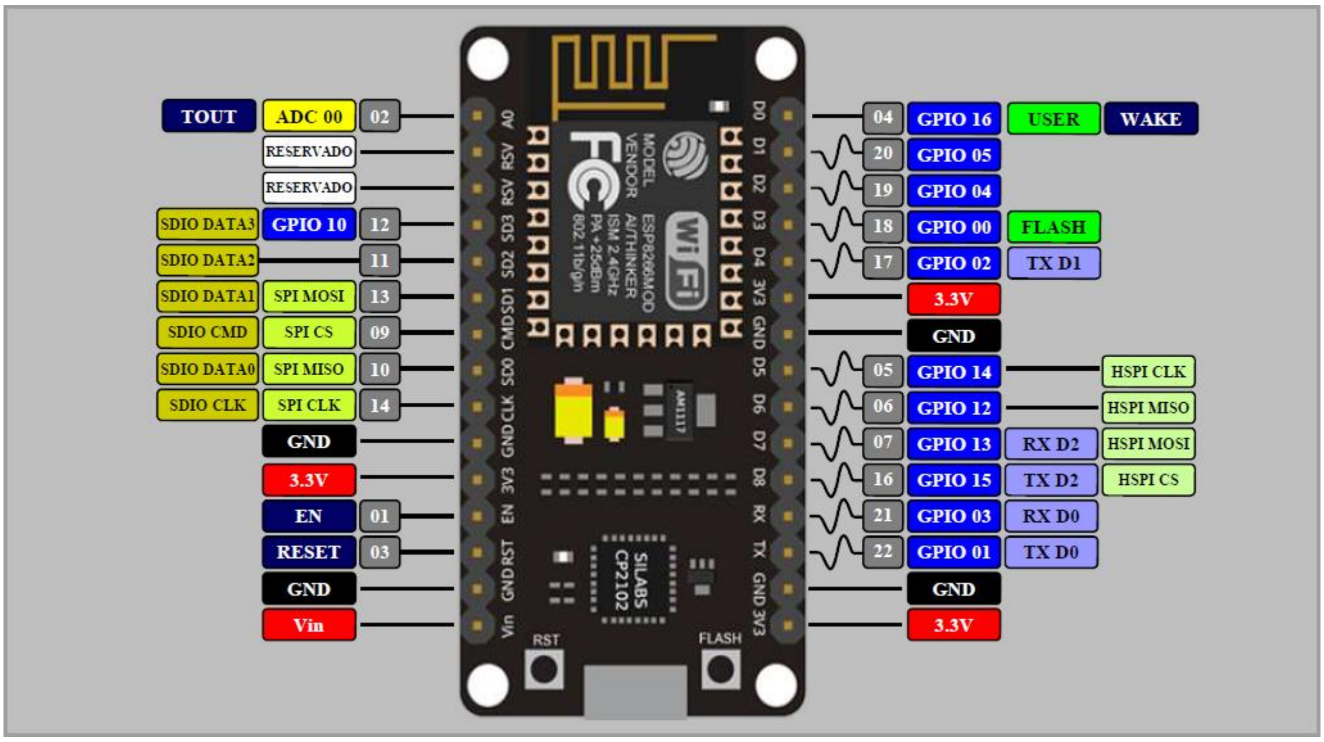
The most common models of the NodeMCU are the Amica (based on the standard narrow pin-spacing) and the LoLin which has the wider pin spacing and larger board. The open-source design of the base ESP8266 enables the market to design new variants of the NodeMCU continually.


D0 D1 D2 D3 D4 3V3 GND D5 D6 D7 D8 RX TX GND 3V3
MODEL ESP8266MOD
VENDOR AI/THINKER
ISM 2.4GHz
PA +25dBm
802.11b/g/n
AM1117
AYARAFUN
FLASH
SILABS CP2102
RST
A0 RSV RSV SD3 SD2 SD1 CMD SD0 CLK GND 3V3 EN RST GND Vin

- Model: ESP8266-12E
- Wireless Standard: 802.11 b/g/n
- Frequency range: 2.4 GHz - 2.5 GHz (2400M-2483.5M)
- Wi-Fi mode: Station / SoftAP / SoftAP+station
- Stack: Integrated TCP/IP
- Output power: 19.5dBm in 802.11b mode
- Data interface: UART / HSPI / I2C / I2S / Ir
- Remote Control GPIO / PWM
- Supports protection mode: WPA / WPA2
- Encryption: WEP / TKIP / AES
- Power supply: from 4.5 VDC to 9 VDC (VIN) or via micro USB connector
- Consumption: with continuous Wi-Fi transmission about 70 mA (200 mA MAX) - in standby < 200µA
- Operating temperature: from -40°C to +125°C
- Dimensions (mm): 58×31.20×13
- Weight: 10 grams

NodeMCU is a low-cost open source IoT platform. It initially included firmware which runs on the ESP8266 Wi-Fi SoC from Espressif Systems, and hardware which was based on the ESP-12 module. Later, support for the ESP32 32-bit MCU was added.

NodeMCU Pinout and Functions Explained:



Power Pins

- ➢ There are four power pins. VIN pin and three 3.3V pins.
- ➢ VIN can be used to directly supply the NodeMCU/ESP8266 and its peripherals.
- ➢ Power delivered on VIN is regulated through the onboard regulator on the NodeMCU module you can also supply 5V regulated to the VIN pin.
- ➢ 3.3V pins are the output of the onboard voltage regulator and can be used to supply power to external components.

GND are the ground pins of NodeMCU/ESP8266

I2C Pins are used to connect I2C sensors and peripherals.

- ➢ Both I2C Master and I2C Slave are supported.
- ➢ I2C interface functionality can be realized programmatically, and the clock frequency is 100 kHz at a maximum.
- ➢ It should be noted that I2C clock frequency should be higher than the slowest clock frequency of the slave device.

GPIO Pins

- NodeMCU/ESP8266 has 17 GPIO pins which can be assigned to functions such as I2C, I2S, UART, PWM, IR Remote Control, LED Light and Button programmatically.
- Each digital enabled GPIO can be configured to internal pull-up or pull-down, or set to high impedance.
- When configured as an input, it can also be set to edge-trigger or level-trigger to generate CPU interrupts.

ADC Channel

- The NodeMCU is embedded with a 10-bit precision SAR ADC.
- The two functions can be implemented using ADC.
- Testing power supply voltage of VDD3P3 pin and testing input voltage of TOUT pin.
- However, they cannot be implemented at the same time.

UART Pins

- NodeMCU/ESP8266 has 2 UART interfaces (UART0 and UART1) which provide asynchronous communication (RS232 and RS485), and can communicate at up to 4.5 Mbps.
- UART0 (TXD0, RXD0, RST0 & CTS0 pins) can be used for communication. However,
- UART1 (TXD1 pin) features only data transmit signal so, it is usually used for printing log.

SPI Pins

- NodeMCU/ESP8266 features two SPIs (SPI and HSPI) in slave and master modes.
- These SPIs also support the following general-purpose SPI features:
    - 4 timing modes of the SPI format transfer
    - Up to 80 MHz and the divided clocks of 80 MHz
    - Up to 64-Byte FIFO

SDIO Pins

- NodeMCU/ESP8266 features Secure Digital Input/Output Interface (SDIO) which is used to directly interface SD cards.
- 4-bit 25 MHz SDIO v1.1 and 4-bit 50 MHz SDIO v2.0 are supported.

PWM Pins

- The board has 4 channels of Pulse Width Modulation (PWM).
- The PWM output can be implemented programmatically and used for driving digital motors and LEDs.
- PWM frequency range is adjustable from 1000 μs to 10000 μs (100 Hz and 1 kHz).
- All of the ESP8266's GPIO pins, from GPIO0 to GPIO15, can be programmed to generate pulse width modulated (PWM) output.

Control Pins are used to control the NodeMCU/ESP8266.
These pins include Chip Enable pin (EN), Reset pin (RST) and WAKE pin.

EN: The ESP8266 chip is enabled when EN pin is pulled HIGH. When pulled LOW the chip works at minimum power.
RST: RST pin is used to reset the ESP8266 chip.
WAKE: Wake pin is used to wake the chip from deep-sleep.

2. LED'S:

- A light-emitting diode is a semiconductor device that emits light when current flows through it.
- Electrons in the semiconductor recombine with electron holes, releasing energy in the form of photons.
- The color of the light is determined by the energy required for electrons to cross the band gap of the semiconductor.

3. BUZZER:

- A buzzer or beeper is an audio signaling device, which may be mechanical, electromechanical, or piezoelectric (piezo for short).
- Typical uses of buzzers and beepers include alarm devices, timers, train and confirmation of user input such as a mouse click or keystroke.
- Typical uses of buzzers and beepers include alarm devices, timers, train and confirmation of user input such as a mouse click or keystroke.
- Buzzers, also known as beepers, are audio signaling devices that convert audio signals into sound.

4. LCD DISPLAY:

- A 16×2 LCD display is a liquid crystal display that can show 16 characters in each of its two rows, providing a total of 32 characters of information.
- It's commonly used to display alphanumeric information in various electronic devices.
- The operating voltage of this LCD is 4.7V-5.3V
- It includes two rows where each row can produce 16-characters.

5. JUMPER WIRES:

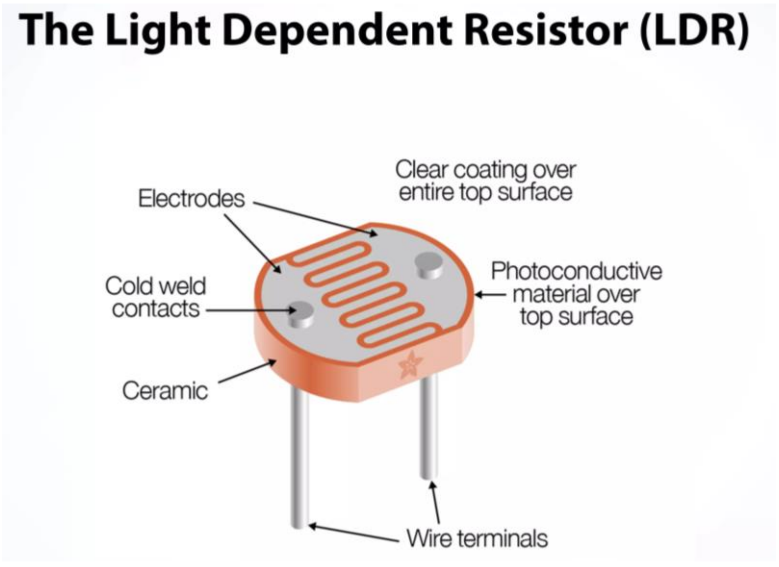
- A jump wire is an electrical wire, or group of them in a cable, with a connector or pin at each end (or sometimes without them – simply "tinned")
- which is normally used to interconnect the components of a breadboard or other prototype or test circuit, internally.

6. PHOTO RESISTOR (LDR) SENSOR:

- A Light Dependent Resistor (LDR) is a type of passive electronic sensor used to detect light.
- It's made up of two conductors separated by an insulator which becomes more conducting when exposed to high levels of light intensity, forming a variable resistor in the circuit.

WORKING PRINCIPLE OF LDR:

The photo conductivity theory underlies the operation of this resistor. It is nothing more than the fact that when light strikes its surface, the material's conductivity decreases and the electrons in the device's valence band are stimulated to the conduction band.


The Light Dependent Resistor (LDR)
Electrodes
Clear coating over entire top surface
Cold weld contacts
Photoconductive material over top surface
Ceramic
Wire terminals

SOFTWARE USED

1. ARDUINO IDE:

- The Arduino IDE (Integrated Development Environment) is used to write the computer code and upload this code to the physical board.
- The Arduino IDE is very simple and this simplicity is probably one of the main reason Arduino became so popular.
- We can certainly state that being compatible with the Arduino IDE is now one of the main requirements for a new microcontroller board.
- Over the years, many useful features have been added to the Arduino IDE and you can now managed third-party libraries and boards from the IDE, and still keep the simplicity of programming the board.

Here's some information about it:

1. **Development Environment**: The Arduino IDE provides a user-friendly interface for writing code in the Arduino programming language, which is a simplified version of C/C.
2. **Cross-Platform**: Arduino IDE is available for multiple operating systems, including Windows, macOS, and Linux, making it accessible to a wide range of users.
3. **Code Editor**: The IDE features a code editor with syntax highlighting, auto-indentation, and code completion, making it easier to write and read code.
4. **Library Support**: Arduino IDE comes with a built-in library manager that allows users to easily install and manage libraries, which are collections of pre-written code that extend the functionality of Arduino boards.
5. **Board Manager**: It includes a board manager feature that enables users to add support for different Arduino-compatible boards, such as Arduino Uno, Arduino Mega, Arduino Nano, etc. This feature allows developers to switch between different boards without needing to install separate software.
6. **Serial Monitor**: Arduino IDE includes a serial monitor tool that allows users to communicate with their Arduino boards in real-time, which is useful for debugging and testing.

2. THINGSPEAK:

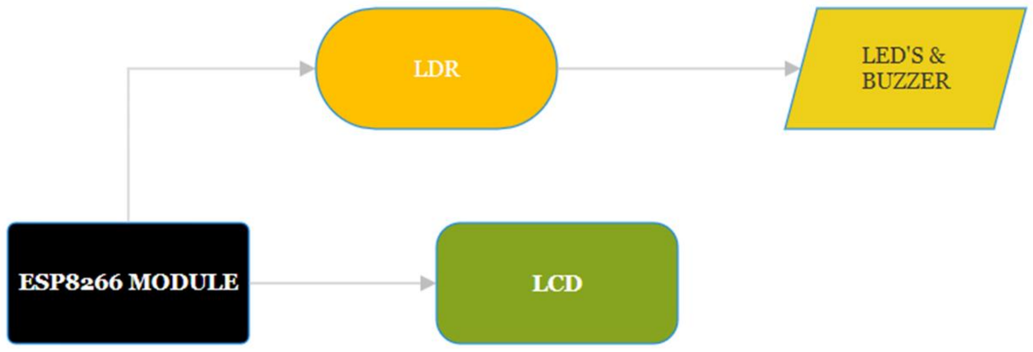
- ThingSpeak is an open-source software written in Ruby which allows users to communicate with internet enabled devices.
- It facilitates data access, retrieval and logging of data by providing an API to both the devices and social network websites.
- ThingSpeak allows you to aggregate, visualize, and analyze live data streams in the cloud.
- ThingSpeak provides instant visualizations of data posted by your devices or equipment.
- Execute MATLAB code in ThingSpeak, and perform online analysis and processing of the data as it comes in.

- ThingSpeak is IoT Cloud platform where you can send sensor data to the cloud. You can also analyze and visualize your data with MATLAB or other software, including making your own applications.
- The ThingSpeak service is operated by MathWorks. In order to sign up for ThingSpeak, you must create a new MathWorks Account or log in to your existing MathWorks Account.
- ThingSpeak is free for small non-commercial projects.
- ThingSpeak includes a Web Service (REST API) that lets you collect and store sensor data in the cloud and develop Internet of Things applications. It works with Arduino, Raspberry Pi and MATLAB (premade libraries and APIs exists) But it should work with all kind of Programming Languages, since it uses a REST API and HTTP.
- ThingSpeak is a platform providing various services exclusively targeted for building IoT applications. It offers the capabilities of real-time data collection, visualizing the collected data in the form of charts, ability to create plugins and apps for collaborating with web services, social network and other APIs.

HARDWARE ARCHITECTURE / CONNECTIVITY

To implement a system using the ESP8266 module to measure light intensity and display it on an LCD, along with additional indicators like a buzzer and LEDs, you would follow these general steps:

1. **Hardware Setup**:
    1. Connect the ESP8266 module to the LCD display, buzzer, and LEDs according to their pin configurations.
    2. Connect a light sensor (such as an LDR - Light Dependent Resistor) to one of the analog pins of the ESP8266 to measure light intensity.



2. **Programming**:

- Initialize the ESP8266 module and configure it to connect to your Wi-Fi network.
- Initialize the LCD display and configure it to show the light intensity and any additional information.
- Set up the buzzer and LEDs as output pins.
- Continuously read the light intensity from the light sensor.
- Based on the light intensity readings:
    - If the intensity is high, display a message on the LCD, turn on the green LED, and trigger the buzzer.
    - If the intensity is low, display a different message on the LCD, turn on the red LED, and turn off the buzzer.
- Update the LCD display with the current light intensity value and any additional information.

TESTING

1. Connect the Micro USB to the Laptop or Power Bank.
2. Make sure the LCD is on.
3. Turn on the Mobile’s Hotspot.
4. Wait until the module connects to the Wi-Fi.
5. After successful connection the LCD shows the Intensity and LED glows indicating LOW/HIGH.
6. Buzzer will make beep sound when intensity is High that is above 50.

CODE

```
#include <ThingSpeak.h>  //To connect our thingspeak channel with the esp8266 through this code.
#include <ESP8266WiFi.h>  // To connect the esp with internet
#include <Wire.h>
#include <LiquidCrystal_I2C.h>
const char* ssid     = "saicharan";    //Name of your wifi network
const char* password = "123456789";  // Wifi password
const long channelID = 2462879; // Channel ID
const char *writeAPIKey = "WASOYXTZBDOJOPBO"; // Your write API Key
WiFiClient client;
unsigned long delayStart = millis();   // start delay
LiquidCrystal_I2C lcd(0x27, 16, 2)
void setup()
{
  lcd.init();
  lcd.backlight();
   pinMode(14, OUTPUT);
   pinMode(12, OUTPUT);
   Serial.begin(115200);
   Serial.println();
   Serial.print("Connecting to ");
   Serial.println(ssid);
   lcd.setCursor(0,0);
   lcd.print("IOT BASED LIGHT" );
   lcd.setCursor(0,1);
   lcd.print("INTENSITY SYSTEM" );
   delay(5000);
   lcd.clear();
   WiFi.begin(ssid, password);  // Connecting esp to wifi
   while (WiFi.status() != WL_CONNECTED)  //This loop will work untill esp connects to the wifi
   {
   delay(500);
   Serial.print(".");
   }
   ThingSpeak.begin(client);       //The client here is esp8266
   delay(1000);
}
void loop() {
   int intensity = analogRead(A0);
   intensity = map(intensity, 0, 1024, 100, 0);
   lcd.setCursor(0, 0);
   lcd.print("INTENSITY : ");
   lcd.print(intensity);
   if (intensity < 50) {
      digitalWrite(12, HIGH);
      digitalWrite(14, LOW);
   } else  {
      digitalWrite(14, HIGH);
      digitalWrite(12, LOW);
   }
   if ((millis() - delayStart) >= 15000) {
      ThingSpeak.writeField(channelID, 1, intensity, writeAPIKey);
      delayStart = millis(); // Reset the delay start time
   }
   delay(500);               // Take a reading every  half a second for testing
   Serial.println();         // Normally you should take reading perhaps once or twice a day
}
```

RESULTS

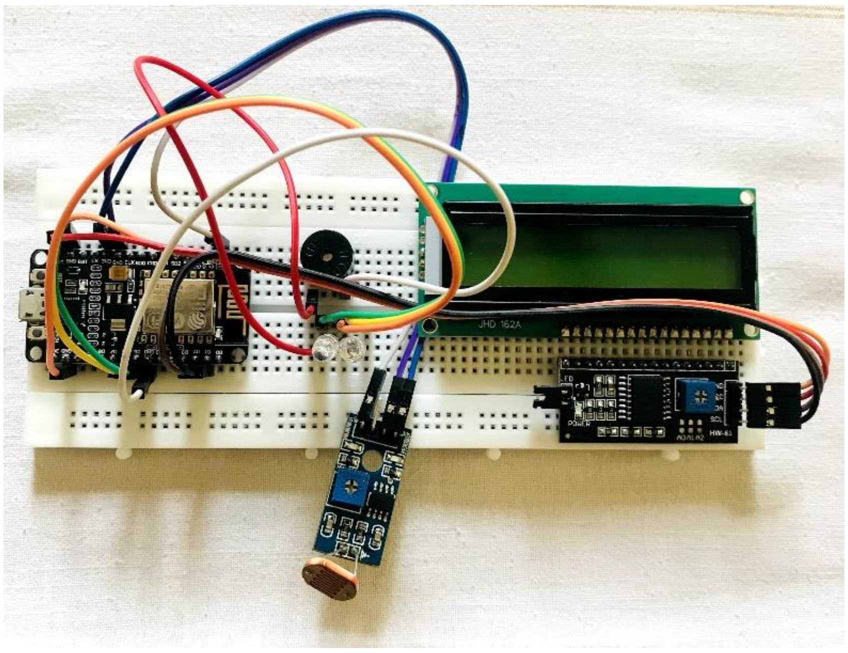


Fig – **FINAL OUTCOME OF HAREDWARE ARCHITECTURE**

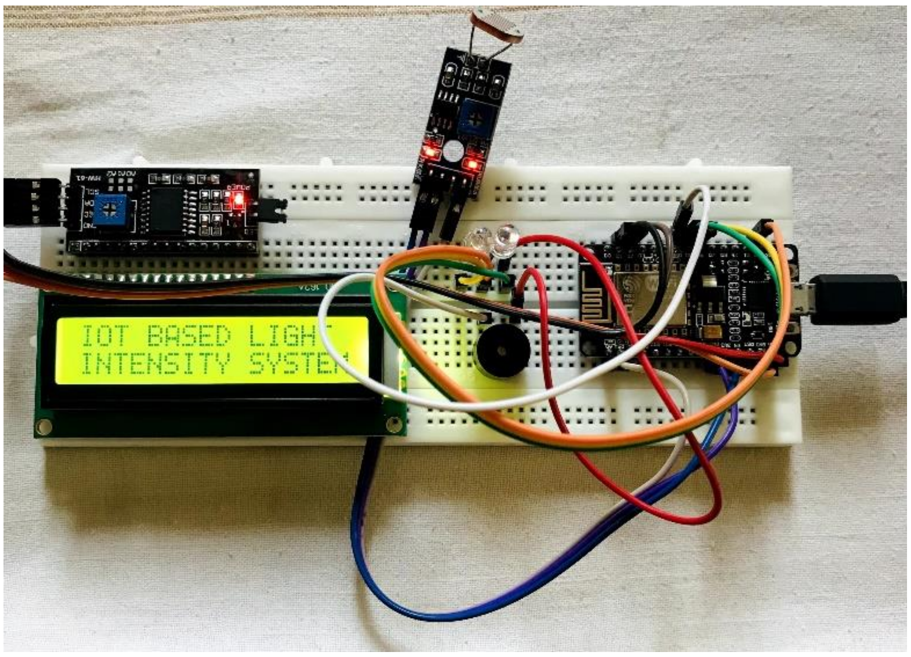


Fig – **POWER SUPPLY IS ON**

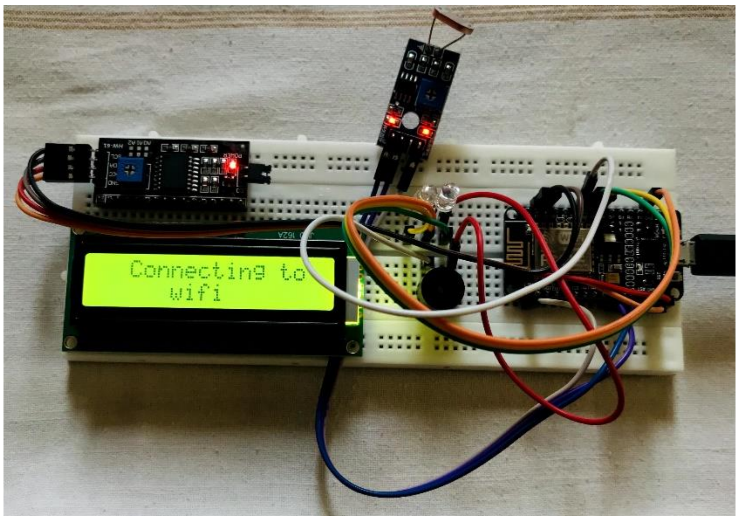


Fig – **ESP8266 Connecting to Wi-Fi**

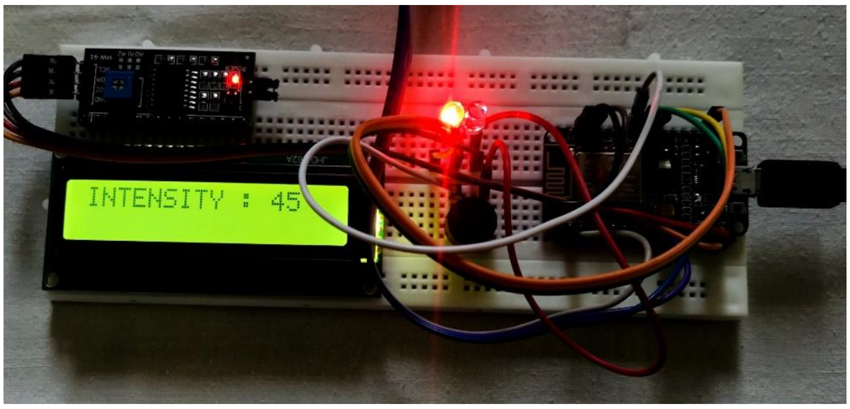


Fig - **LOW INTENSITY (RED LED GLOWS)**

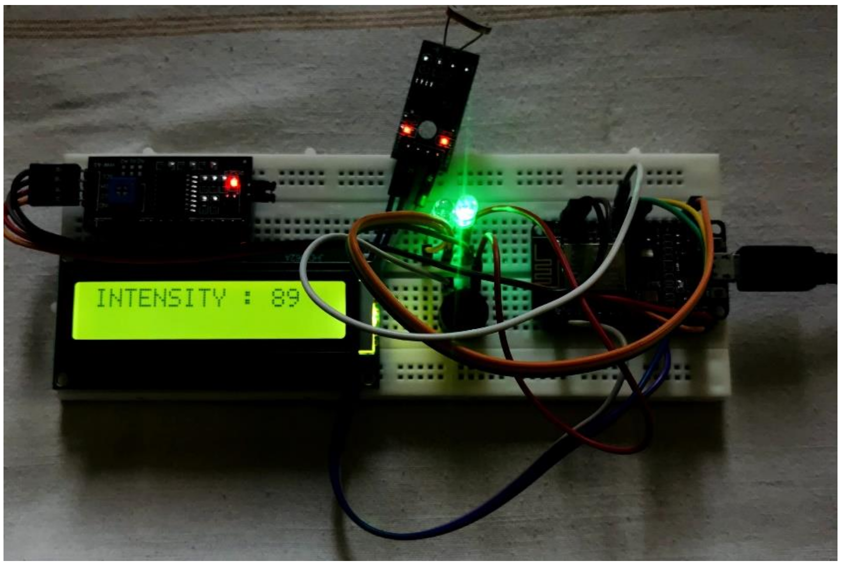
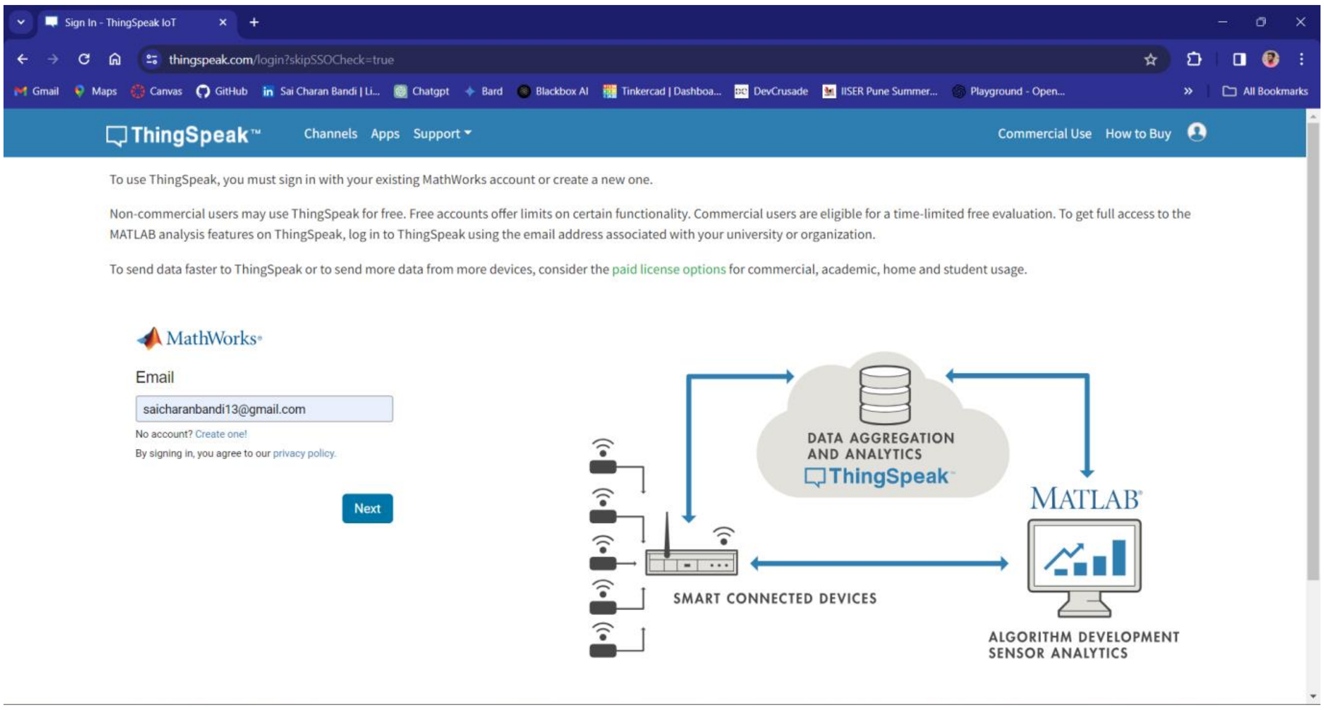


Fig - **HIGH INTENSITY (GREEN LED GLOWS)**

TESTING THINGSPEAK RESULTS:

Creating a channel in ThingSpeak, an IoT analytics platform, involves several steps. Here's a general guide:

- Sign up/login to ThingSpeak.
- Navigate to the "Channels" tab.
- Click on "New Channel."
- Fill in channel details (name, description, field names, etc.).
- Set channel privacy.
- Adjust advanced settings if needed.
- Save your channel.
- Get API keys.
- Start sending data to your channel.
- View and analyze data.



Sign In - ThingSpeak IoT

thingspeak.com/login?skipSSOCheck=true

To use ThingSpeak, you must sign in with your existing MathWorks account or create a new one.

Non-commercial users may use ThingSpeak for free. Free accounts offer limits on certain functionality. Commercial users are eligible for a time-limited free evaluation. To get full access to the MATLAB analysis features on ThingSpeak, log in to ThingSpeak using the email address associated with your university or organization.

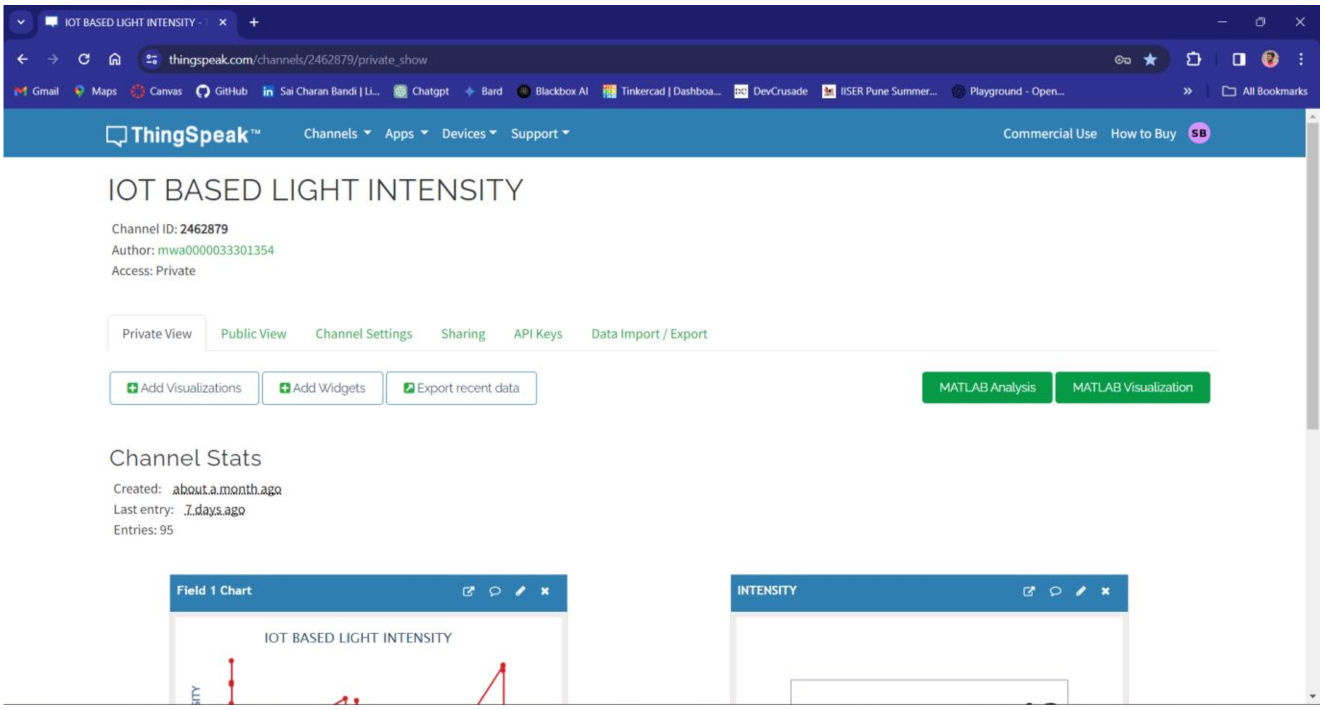
To send data faster to ThingSpeak or to send more data from more devices, consider the paid license options for commercial, academic, home and student usage.

Email: saicharanbandi13@gmail.com

No account? Create one!

By signing in, you agree to our privacy policy.

Fig – Login to ThingSpeak



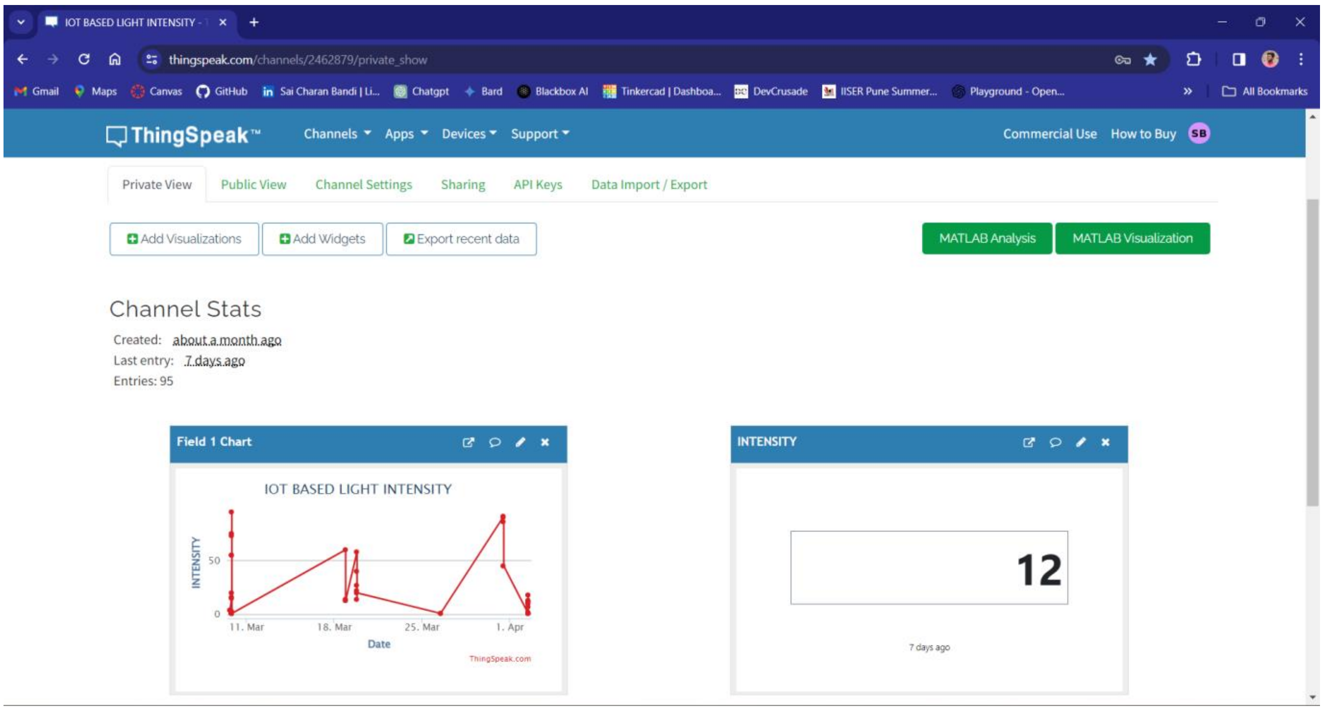
IOT BASED LIGHT INTENSITY

Channel ID: 2462879
Author: mwa0000033301354
Access: Private

Channel Stats

Created: about a month ago
Last entry: 7 days ago
Entries: 95

Fig – Channel and its settings

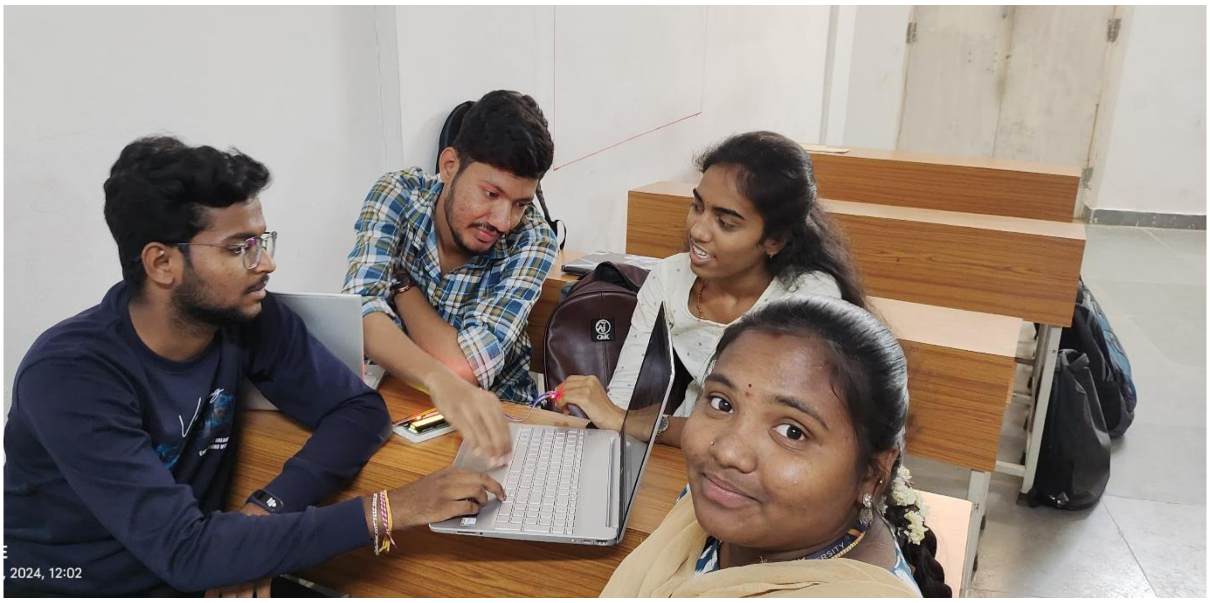


Channel Stats

Created: about a month ago
Last entry: 7 days ago
Entries: 95

Fig – Output in the LCD and Visualization of Output

TEAM



CONCLUSION

- In conclusion, the project successfully demonstrated the practical application of ESP8266 module coupled with LCD, LEDs, and a buzzer to measure light intensity.
- By integrating these components, we achieved a versatile system capable of accurately monitoring ambient light levels and providing visual and auditory feedback accordingly.
- This project not only showcases the capabilities of ESP8266 in IoT applications but also highlights the importance of sensor integration for real-time environmental monitoring.
- Moving forward, further enhancements such as remote data logging, wireless connectivity, and integration with other smart devices can be explored to extend the functionality and utility of the system.
- Overall, the project lays a solid foundation for future developments in the field of light intensity measurement and IoT-based solutions.

REFERENCES

[1] Z. Wan, Y. Song, and Z. Cao, AUTOMATED STREET LIGHT SYSTEM with ESP8266 NodeMCU," in 2019 IEEE 3rd Information Technology, Networking, Electronic and Automation Control Conference (ITNEC), Chengdu, China, 2019, pp. 377–382.

[2] K. Sangsanit and C. Techapanupreeda, "NodeMCU Choreography Automation by CoAP," in 2019 International Conference on Information Networking (ICOIN), Kuala Lumpur, Malaysia, 2019, pp. 350–353.

[3] L. C. S. Paavan, T. G. Sai, and M. K. Naga, "An IoT based Smart Garbage Alert System," in 2019 3rd International Conference on Trends in Electronics and Informatics (ICOEI), Tirunelveli, India, 2019, pp. 425–430.

[4] "Espruino on ESP8266 WiFi." [Online]. Available: https://www.espruino.com/EspruinoESP8266.

[5]"Mongoose OS - an IoT Operating System." [Online]. Available: https://mongoose-os.com/mos.html.

[6] https://www.espressif.com/en/products/software/esp-sdk/overview. [Accessed: 20-Nov-2019].

[7] "Arduino - Software." [Online]. Available: https://www.arduino.cc/en/main/software.