INTELLIGENT AND WEATHER ADAPTIVE STREET LIGHTING SYSTEM

## ABSTRACT-

This project will be used to transform lighting from a simple illumination source into a smart infrastructure of the project. IOT will help in connecting the internet with the physical entity(lighting system).Helps in sensing the environment and providing the lighting accordingly leading to efficient use of electricity and saving energy. The project helps in controlling the street lights wirelessly in a centralized manner. The lights can be switched ON and OFF anytime anywhere. Also, the intensity of the light can also be controlled according to the level of darkness in the surrounding environment. The smart street lighting also controls the luminosity of light and performs automatic light dimming which is an aspect that serves to reduce energy consumption. The intensity of light can be controlled based on illumination and the weather conditions. Logically, this system may save a large amount of the electrical power.

## PROBLEM STATEMENT

This project’s target is to optimize the street lighting system. The electricity used for the street lights is placed at anywhere between twenty to forty percent of that produced in India. An IoT based solution will keep control of which lights are working, and how much intensity will work the best in a particular scenario.This project aims at designing smart street lighting system for energy saving of street lights. It controls the street lights based on detection of vehicles or any other obstacles on the street. We can control the lights wirelessly and also vary its intensity depending on the darkness level. Whenever the obstacle is detected on the street within the specified time the light will get automatically ON/OFF according to the obstacle detection.

# PROJECT WORKING PROCESS

In this project we use aurdino software.It consists of two parts, the circuit board, used for making connections (hardware part) and the arduino IDE the software part of the arduino in which the programming is done. Smart lighting provides remote adaptive lighting control by identifying opportunities for improved power usage and optimizing operational efficiency dynamically through the sensors and microcontrollers based on the weather conditions, traffic density, and even naturalistic features. IOT streetlight systems use a combination of long- medium- and short-range networks communication technologies. We send the command for switching ON or OFF of the lights through our phone or the laptop. This command is received by the arduino through the Wi-Fi module, which then controls the light according to the command.

# COMPONENTS USED

1)MIT App Inventor-It is an intuitive, visual programming environment that allows everyone – even children – to build fully functional apps for smartphones and tablets. Those new to MIT App Inventor can have a simple first app up and running in less than 30 minutes. And what's more, our blocks-based tool facilitates the creation of complex, high-impact apps in significantly less time than traditional programming environments. The MIT App Inventor project seeks to democratize software development by empowering all people, especially young people, to move from technology consumption to technology creation

2)ESP8266-The ESP8266 is a low-cost Wi-Fi module produced by manufacturer Espressif Systems in Shanghai, China.

The chip first came to the attention of western makers in August 2014 with the ESP-01 module, made by a third-party manufacturer Ai-Thinker. This small module allows microcontrollers to connect to a Wi-Fi network and make simple TCP/IP connections using Hayes style commands. However, at first there was almost no English-language documentation on the chip and the commands it accepted.The very low price and the fact that there were very few external components on the module, which suggested that it could eventually be very inexpensive in volume, attracted many hackers to explore the module, chip, and the software on it, as well as to translate the Chinese documentation

* 3)Basic Sheild-It contains 8 LEDs (5 mm)
* 4 Push Button
* 2 Potentiometer (102k)
* LDR (light detection resistor)
* Buzzer

4)Aurdino Software- It is open source software which can be used for projects requiring to control or program objects. It is basically used for electronics based projects. It consists of two parts, the circuit board, used for making connections (hardware part) and the arduino IDE the software part of the arduino in which the programming is done.

5)NRF Modules-The NRF24L01 is a wireless transceiver module, meaning each module can both send as well as receive data. They operate in the frequency of 2.4GHz, which falls under the ISM band and hence it is legal to use in almost all countries for engineering applications. The modules when operated efficiently can cover a distance of 100 meters (200 feet) which makes it a great choice for all wireless remote controlled projects.

The module operates at 3.3V hence can be easily used with 3.2V systems or 5V systems. Each module has an address range of 125 and each module can communicate with 6 other modules hence it is possible to have multiple wireless units communicating with each other in a particular area. Hence mesh networks or other types of networks are possible using this module.

6)Node Red Flow Editor-RED is a programming tool for wiring together hardware devices, APIs and online services in new and interesting ways. It provides a browser-based editor that makes it easy to wire together flowsusing the wide range of nodes in the palette that can be deployed to its runtime in a single-click.

# The Final Code-

Transmitter Code-

#include <ESP8266WiFi.h>  
#include <PubSubClient.h>  
const char\* ssid = "SB-IOT1";  
const char\* password = "sb@iot11";  
String command;  
   
#define ORG "nrdaz1"  
#define DEVICE\_TYPE "nodemcu"  
#define DEVICE\_ID "1000"  
#define TOKEN "2210416109"  
char server[] = ORG ".[messaging.internetofthings.ibmcloud.com](http://messaging.internetofthings.ibmcloud.com/)";  
char topic[] = "iot-2/evt/data/fmt/json";  
char topic1[] = "iot-2/cmd/home/fmt/String";  
char authMethod[] = "use-token-auth";  
char authMethod1[] = "use-token-auth";  
char token[] = TOKEN;  
char token1[] = TOKEN;  
char clientId[] = "d:" ORG ":" DEVICE\_TYPE ":" DEVICE\_ID;  
#include <SPI.h>  
#include <nRF24L01.h>  
#include <RF24.h>   
void callback(char\* topic1, byte\* payload, unsigned int payloadLength);  
WiFiClient wifiClient;  
//PubSubClient client(server, 1883,wifiClient);  
PubSubClient client(server, 1883, callback, wifiClient);  
  
RF24 radio(D4, D8); // CE, CSN  
  
const byte address[6] = "VENKY";//address is a array and give the address name as numbers or string  
  
void setup() {  
  pinMode(D2,OUTPUT);  
  radio.begin();//radio is a instance    
  radio.openWritingPipe(address);// giving address to the nRF  
  radio.setPALevel(RF24\_PA\_MIN);//setting amplification level minimum  
  radio.stopListening();//  
  Serial.begin(9600);  
 Serial.println();  
 Serial.print("Connecting to ");  
 Serial.print(ssid);  
 WiFi.begin(ssid, password);  
 while (WiFi.status() != WL\_CONNECTED) {  
 delay(500);  
 Serial.print(".");  
 }   
 Serial.println("");  
   
 Serial.print("WiFi connected, IP address: ");  
 Serial.println(WiFi.localIP());  
   wifiConnect();  
  mqttConnect();  
}  
  
void loop() {  
  int ldr = analogRead(A0);  
  if (isnan(ldr))  
{  
Serial.println("Failed to read from DHT sensor!");  
delay(1000);  
return;  
}  
PublishData(ldr);  
 if (!client.loop()) {  
    mqttConnect();  
  }  
delay(100);  
if(ldr <= 500)  
{  
  digitalWrite(D2 ,LOW);  
//  const char text[] = "lightoff";//transmiting hello world as data we can change it as our message   
  //radio.write(&text, sizeof(text));//it will transmit the data and size of it  
 // Serial.print("lightoff");  
  delay(1000);  
}  
else  
{  
  analogWrite(D2 ,ldr);  
  const char text[] = "on";//transmiting hello world as data we can change it as our message   
  radio.write(&text, sizeof(text));//it will transmit the data and size of it  
 // Serial.print("lighton");  
  delay(1000);  
}  
}  
void wifiConnect() {  
  Serial.print("Connecting to "); Serial.print(ssid);  
  WiFi.begin(ssid, password);  
  while (WiFi.status() != WL\_CONNECTED) {  
    delay(500);  
    Serial.print(".");  
  }  
  Serial.print("nWiFi connected, IP address: "); Serial.println(WiFi.localIP());  
}  
void mqttConnect() {  
  if (!client.connected()) {  
    Serial.print("Reconnecting MQTT client to "); Serial.println(server);  
    while (!client.connect(clientId, authMethod, token)) {  
      Serial.print(".");  
      delay(500);  
    }  
    initManagedDevice();  
    initManagedDevice1();  
    Serial.println();  
  }  
}  
void initManagedDevice() {  
  if (client.subscribe(topic)) {  
    Serial.println("subscribe to cmd OK");  
  } else {  
    Serial.println("subscribe to cmd FAILED");  
  }  
}  
void initManagedDevice1() {  
  if (client.subscribe(topic1)) {  
    Serial.println("subscribe to cmd OK");  
  } else {  
    Serial.println("subscribe to cmd FAILED");  
  }  
}  
  
void callback(char\* topic1, byte\* payload, unsigned int payloadLength) {  
  Serial.print("callback invoked for topic: "); Serial.println(topic);  
  
  for (int i = 0; i < payloadLength; i++) {  
   // Serial.println((char)payload[i]);  
    command += (char)payload[i];  
  }  
Serial.println(command);  
if(command == "lightoff"){  
  digitalWrite(D2,LOW);  
  Serial.println("Light is Switched Off");  
}  
else if(command == "lighton"){  
  digitalWrite(D2,HIGH);  
  Serial.println("Light is Switched On");  
}  
command ="";  
}  
  
void PublishData(float ldr){  
  if (!!!client.connected()) {  
 Serial.print("Reconnecting client to ");  
 Serial.println(server);  
 while (!!!client.connect(clientId, authMethod, token)) {  
 Serial.print(".");  
 delay(500);  
 }  
 Serial.println();  
 }  
 String payload = "{\"d\":{\"ldr\":";  
 payload += ldr;  
  payload += "}}";  
  Serial.print("Sending payload: ");  
 Serial.println(payload);   
 if (client.publish(topic, (char\*) payload.c\_str())) {  
 Serial.println("Publish ok");  
 } else {  
 Serial.println("Publish failed");  
 }  
}

THE RECEIVER CODE-

#include <SPI.h>  
#include <nRF24L01.h>  
#include <RF24.h>  
RF24 radio(D4,D8); // CE, CSN  
const byte address[6] = "VENKY";  
void setup() {  
  Serial.begin(9600);  
  radio.begin();  
  radio.openReadingPipe(0, address);  
  radio.setPALevel(RF24\_PA\_MIN);  
  radio.startListening();  
  pinMode(D1,OUTPUT);  
}  
void loop() {  
  char k[32]="on";  
  if (radio.available()) {  
    char text[32] = "";  
    radio.read(&text, sizeof(text));  
     Serial.println(text);  
    //Serial.println("KKKJK");  
    digitalWrite(D1,HIGH);  
   //delay(400);  
   }  
   else  
   {Serial.println("shfsjgv");  
     digitalWrite(D1,LOW);  
     }  
     delay(1100);  
}