# STREET LIGHT AUTOMATION:

AIM:

To develop the application of IR Sensor and power saving.

Components required:

1. IR Sensor

2. Atmega32 Micro controller

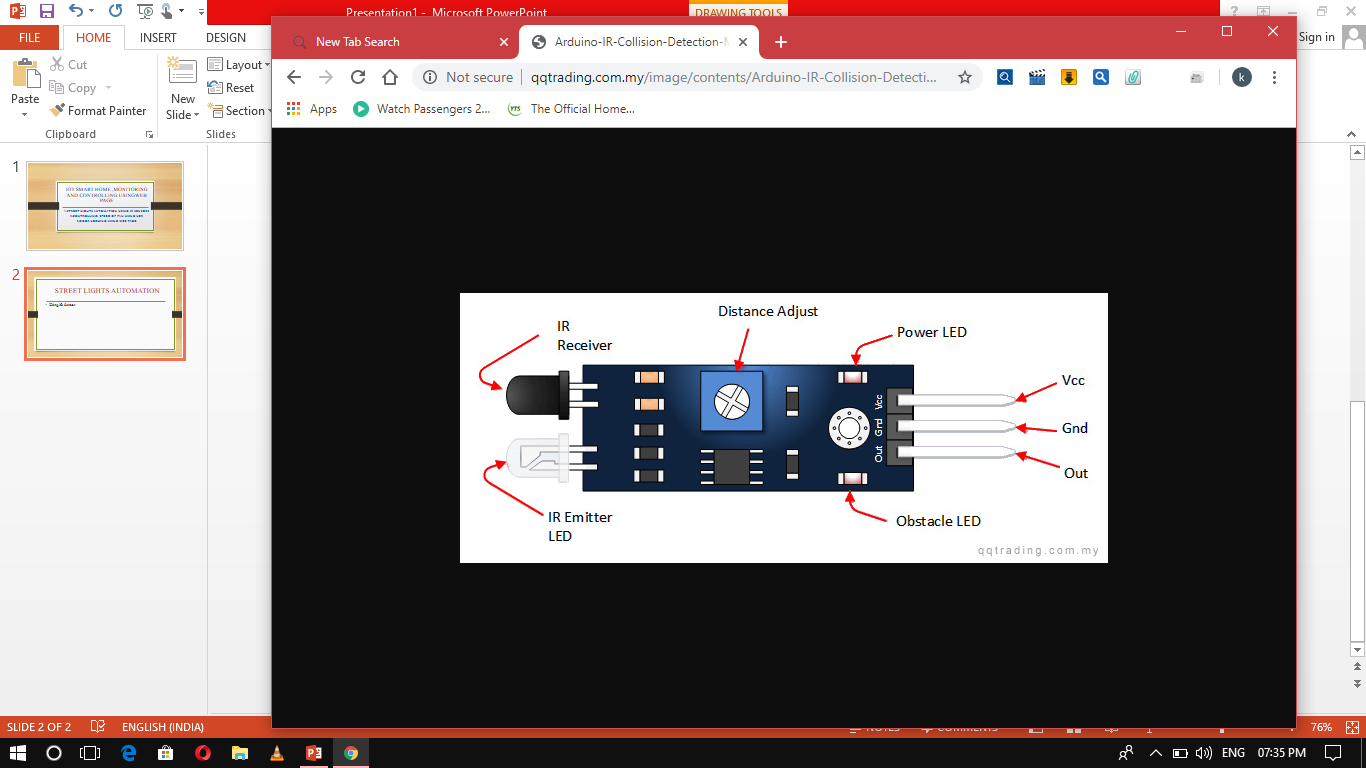
3. Basic Shield

4. Connecting wires

5. USB cable

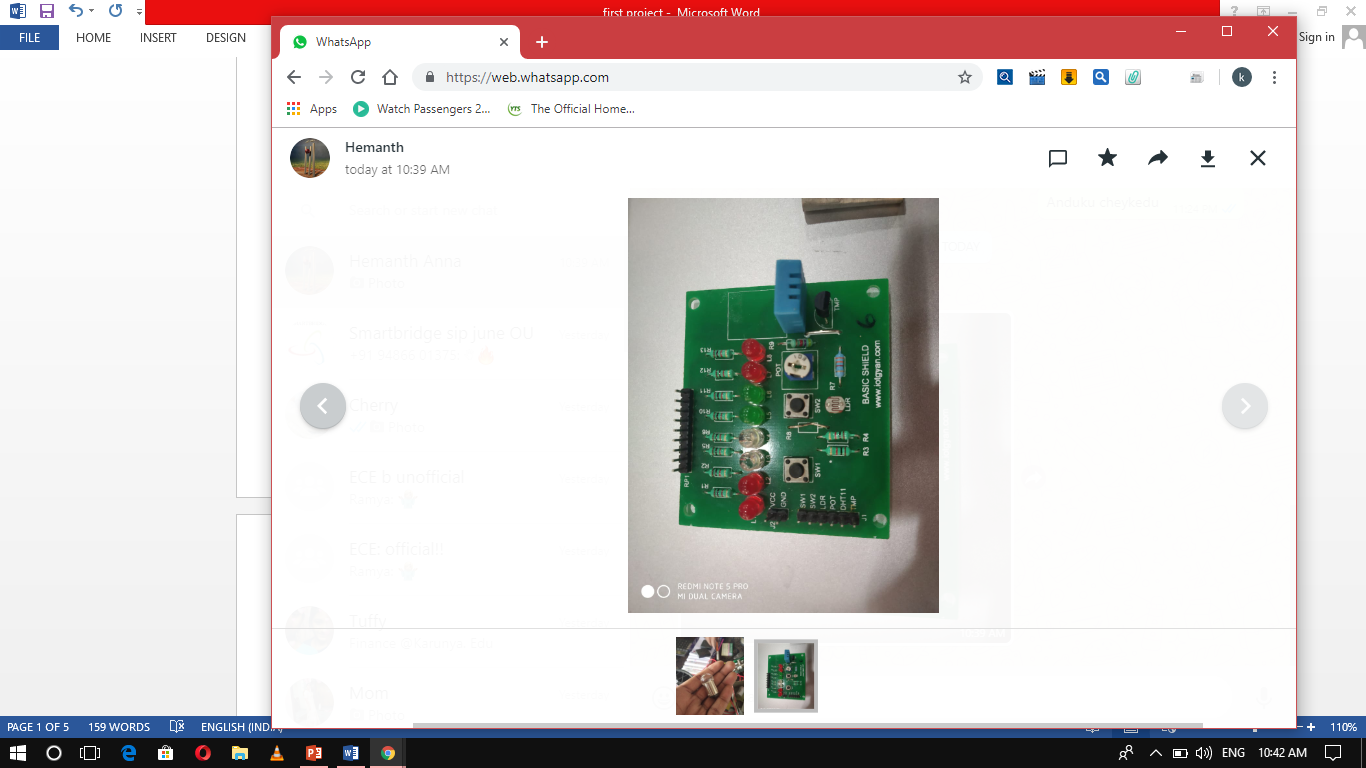
Introduction:

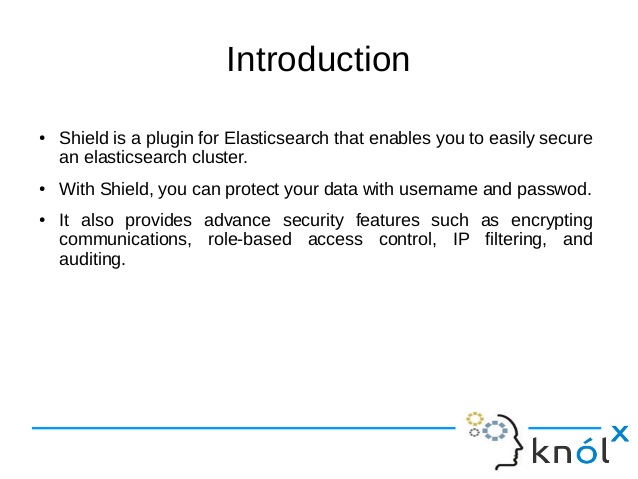
IR Sensor:



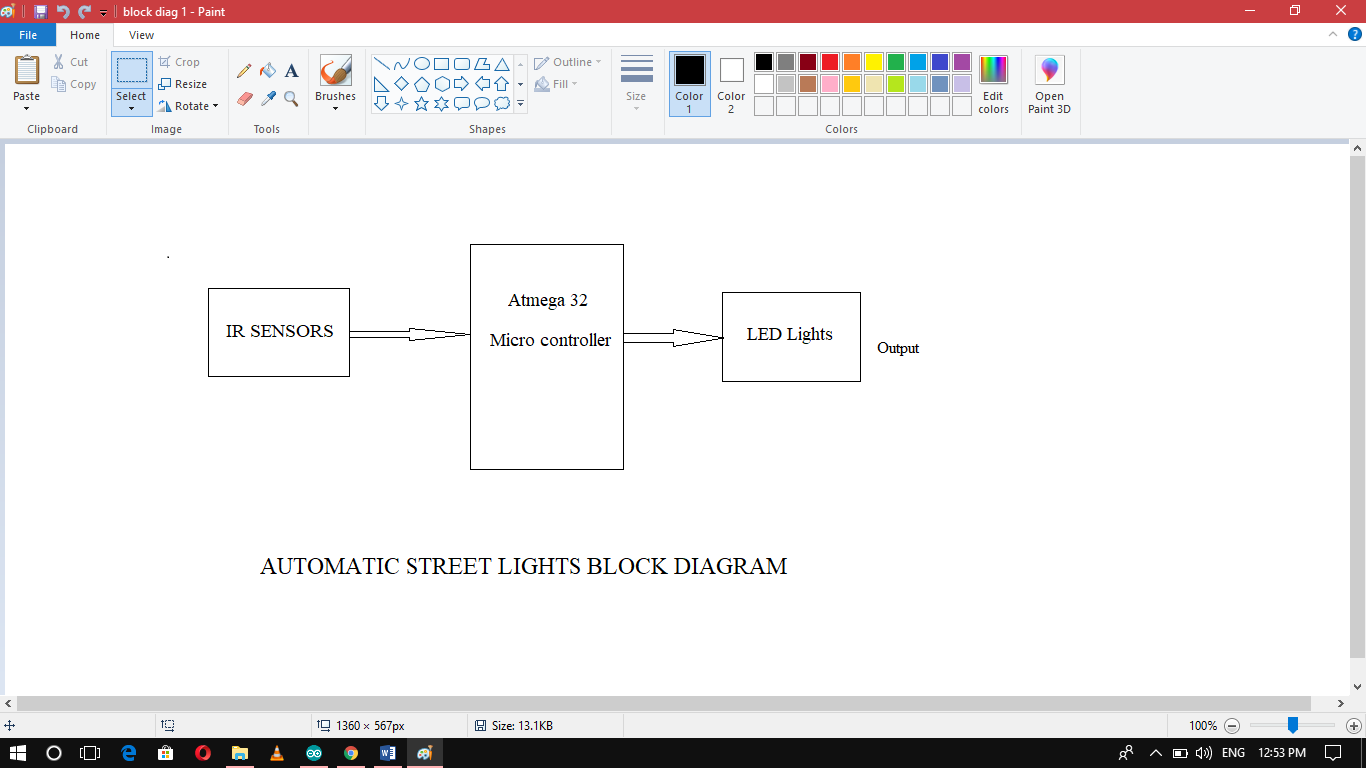
* An infrared sensor is an electronic device, that emits in order to sense some aspects of the surroundings.
* An IR sensor can measure the heat of an object as well as detects the motion

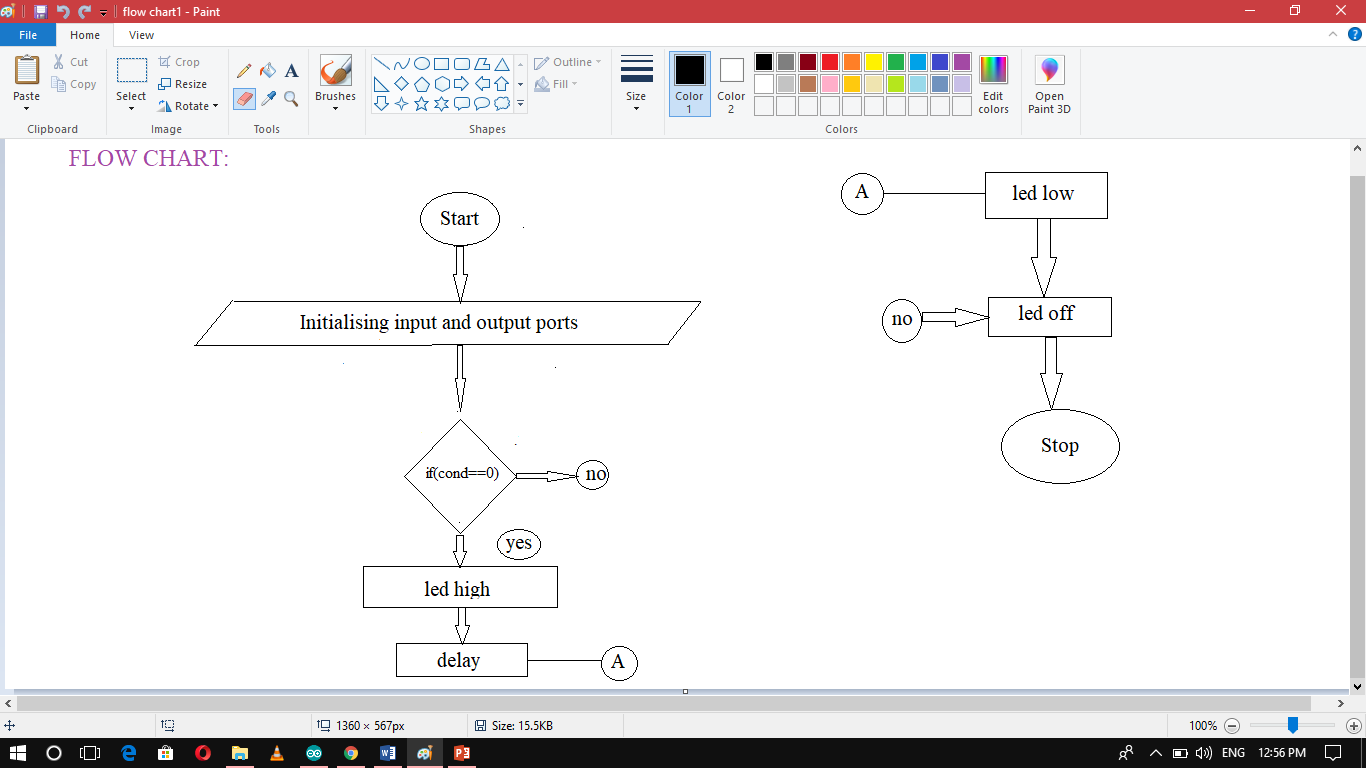
Basic Shield:





# BLOCK DIAGRAM:





# Algorithm:

1.Start the program.

2.Initialize the input and output ports.

3.Set a switch condition ,if the condition is true then on the led and delay for 500 milli seconds and off the led.

4.Else off the led.

5. Repeat it for several ports which were included.

6.Stop the program.

# Source Code:

#ifndef *F\_CPU*

#define *F\_CPU* 16000000UL

#endif

#include<avr/io.h>

#include<util/delay.h>

int main(void)

{

DDRA|=(1<<PA0);

DDRB&=~(1<<PB0);

DDRA|=(1<<PA1);

DDRB&=~(1<<PB1);

if((PINB&(1<<PB0))==0)

{

PORTA|=(1<<PA0);

*\_delay\_ms*(500);

PORTA&=~(1<<PA0);

}

else if((PINB&(1<<PB1))==0)

{

PORTA|=(1<<PA1);

*\_delay\_ms*(500);

PORTA&=~(1<<PA1);

}

else

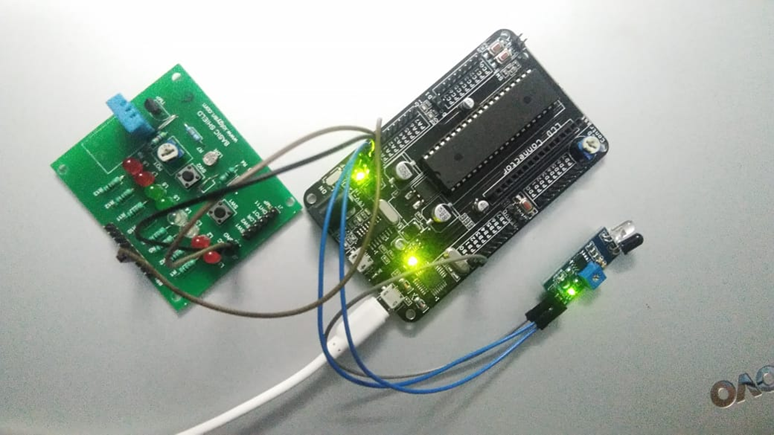
{

PORTA=0x00;

}

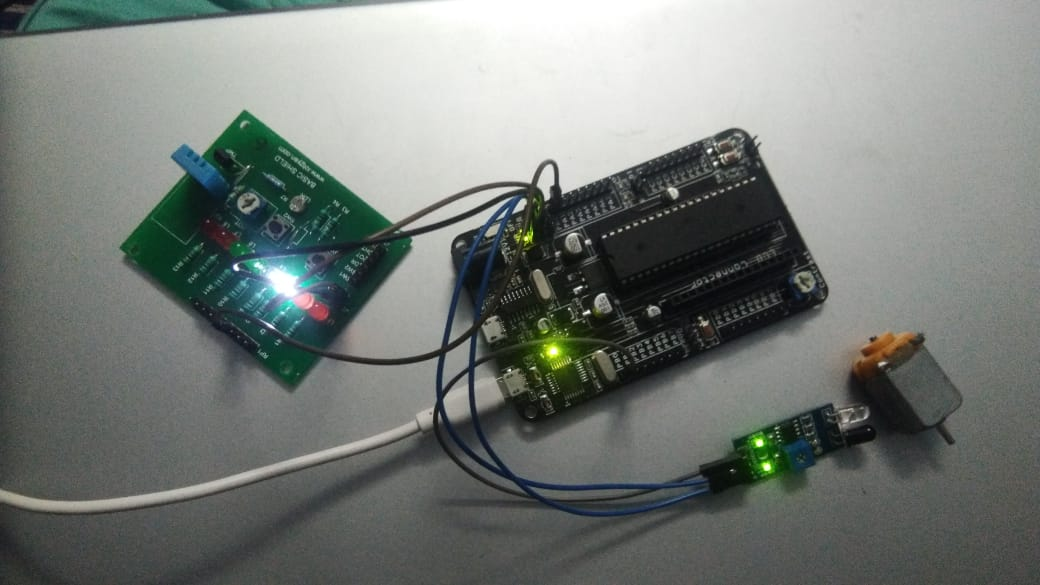
}

Working before the object passing through IR Sensor:



* Connections are set as per the picture shown.
* IR Sensor has transmitter and receiver , transmitter is used to emit radiation of required wave length.
* This radiation reaches the object and is reflected back , the reflected radiation is detected by the IR receiver.
* The IR Receiver detected radiation is then further processed based on its intensity.

Working after the object passing through IR Sensor



* When an object passes through IR Sensor , it results to turn “ON” led when an object goes away then led results to turn “OFF”.
* When an object is placed is placed infront if IR Sensor it automatically receives the signal and transmitter emits the signal and makes led to turn “ON”

When an object goes away no signal can be received and transmitted by IR Sensor then led turned “OFF” automatically.

Conclusion:

By using this we can save power. There are lower chances of the automatic street light system over heating and risk of accidents are also minimized. Cost of operating automatic street lights is far less when comparing to the conventional street lights .Automatic street lights system s are eco-friendly and helps in reducing the carbon footprint.

DOOR CONTROLLING USING WEB PAGE:

AIM:

To lock and unlock the door using web page and servo motor.

Components required:

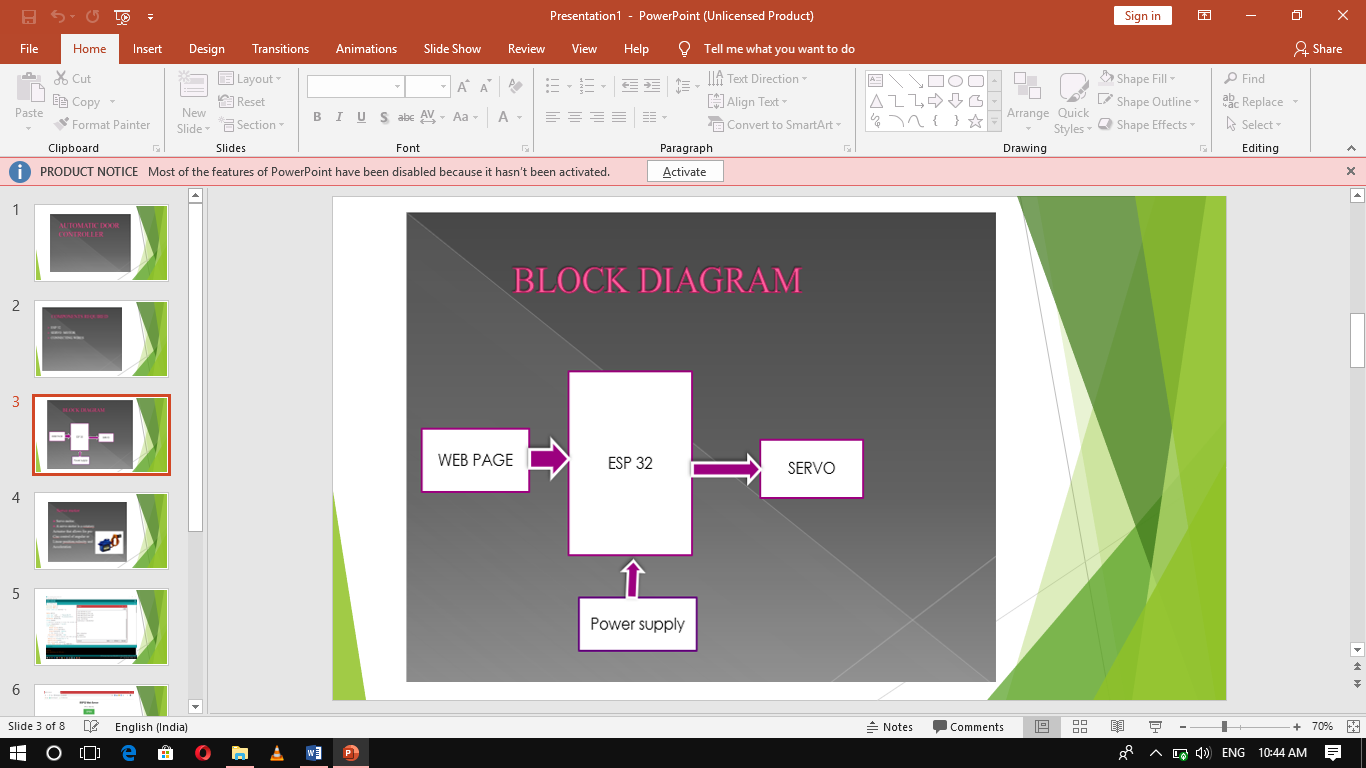
1. Servo Motor

2. ESP32 Micro controller

3. Connecting wires

Servo Motor:

A servo motor is a rotatory actuator that allows for precise control of angular or linear position , velocity and acceleration.



Source Code:

#include <Servo.h>

#include <WiFi.h>

static const int servoPin = 2;

Servo servo1;

const char\* ssid = "cherry";

const char\* password = "12345678";

WiFiServer server(80);

String header;

// Auxiliar variables to store the current output state

String output2State = "close";

void setup() {

Serial.begin(115200);

servo1.attach(servoPin);

pinMode(servoPin, OUTPUT);

// Set outputs to LOW

digitalWrite(servoPin, LOW);

// Connect to Wi-Fi network with SSID and password

Serial.print("Connecting to ");

Serial.println(ssid);

WiFi.begin(ssid, password);

while (WiFi.status() != WL\_CONNECTED) {

delay(500);

Serial.println(".");

}

// Print local IP address and start web server

Serial.println("");

Serial.println("WiFi connected.");

Serial.println("IP address: ");

Serial.println(WiFi.localIP());

server.begin();

}

void loop(){

WiFiClient client = server.available();

if (client) {

Serial.println("New Client."); // print a message out in the serial port

String currentLine = ""; // make a String to hold incoming data from the client

while (client.connected()) { // loop while the client's connected

if (client.available()) { // if there's bytes to read from the client,

char c = client.read(); // read a byte, then

Serial.write(c); // print it out the serial monitor

header += c;

if (c == '\n') { // if the byte is a newline character

// if the current line is blank, you got two newline characters in a row.

// that's the end of the client HTTP request, so send a response:

if (currentLine.length() == 0) {

// HTTP headers always start with a response code (e.g. HTTP/1.1 200 OK)

// and a content-type so the client knows what's coming, then a blank line:

client.println("HTTP/1.1 200 OK");

client.println("Content-type:text/html");

client.println("Connection: close");

client.println();

// turns the GPIOs on and off

if (header.indexOf("GET /2/open") >= 0) {

Serial.println("GPIO 2 open");

output2State = "open";

digitalWrite(servoPin, HIGH);

} else if (header.indexOf("GET /2/close") >= 0) {

Serial.println("GPIO 2 close");

output2State = "close";

digitalWrite(servoPin, LOW);

}

// Display the HTML web page

client.println("<!DOCTYPE html><html>");

client.println("<head><meta name=\"viewport\" content=\"width=device-width, initial-scale=1\">");

client.println("<link rel=\"icon\" href=\"data:,\">");

// CSS to style the on/off buttons

// Feel free to change the background-color and font-size attributes to fit your preferences

client.println("<style>html { font-family: Helvetica; display: inline-block; margin: 0px auto; text-align: center;}");

client.println(".button { background-color: #4CAF50; border: none; color: white; padding: 16px 40px;");

client.println("text-decoration: none; font-size: 30px; margin: 2px; cursor: pointer;}");

client.println(".button2 {background-color: #555555;}</style></head>");

// Web Page Heading

client.println("<body><h1>ESP32 Web Server</h1>");

// Display current state, and ON/OFF buttons for GPIO 2

client.println("<p>GPIO 2 - State " + output2State + "</p>");

// If the output26State is off, it displays the ON button

if (output2State=="close") {

for(int posDegrees = 0; posDegrees <= 180; posDegrees++) {

servo1.write(posDegrees);

Serial.println(posDegrees);

delay(20);

}

client.println("<p><a href=\"/2/open\"><button class=\"button\">OPEN</button></a></p>");

} else {

for(int posDegrees = 180; posDegrees >= 0; posDegrees--) {

servo1.write(posDegrees);

Serial.println(posDegrees);

delay(20);

}

client.println("<p><a href=\"/2/close\"><button class=\"button2\">CLOSE</button></a></p>");

}

// The HTTP response ends with another blank line

client.println();

// Break out of the while loop

break;

} else { // if you got a newline, then clear currentLine

currentLine = "";

}

} else if (c != '\r') { // if you got anything else but a carriage return character,

currentLine += c; // add it to the end of the currentLine

}

}

}

// Clear the header variable

header = "";

// Close the connection

client.stop();

Serial.println("Client disconnected.");

Serial.println("");

}

}

Conclusion

The main concept of this program is to control the door automatically (i.e locking and unlocking)we can control this by using servo motor and web page.

SPEED OF FAN CONTROLLING USING LDR:

AIM:

To control speed of fan using LDR.

Components required:

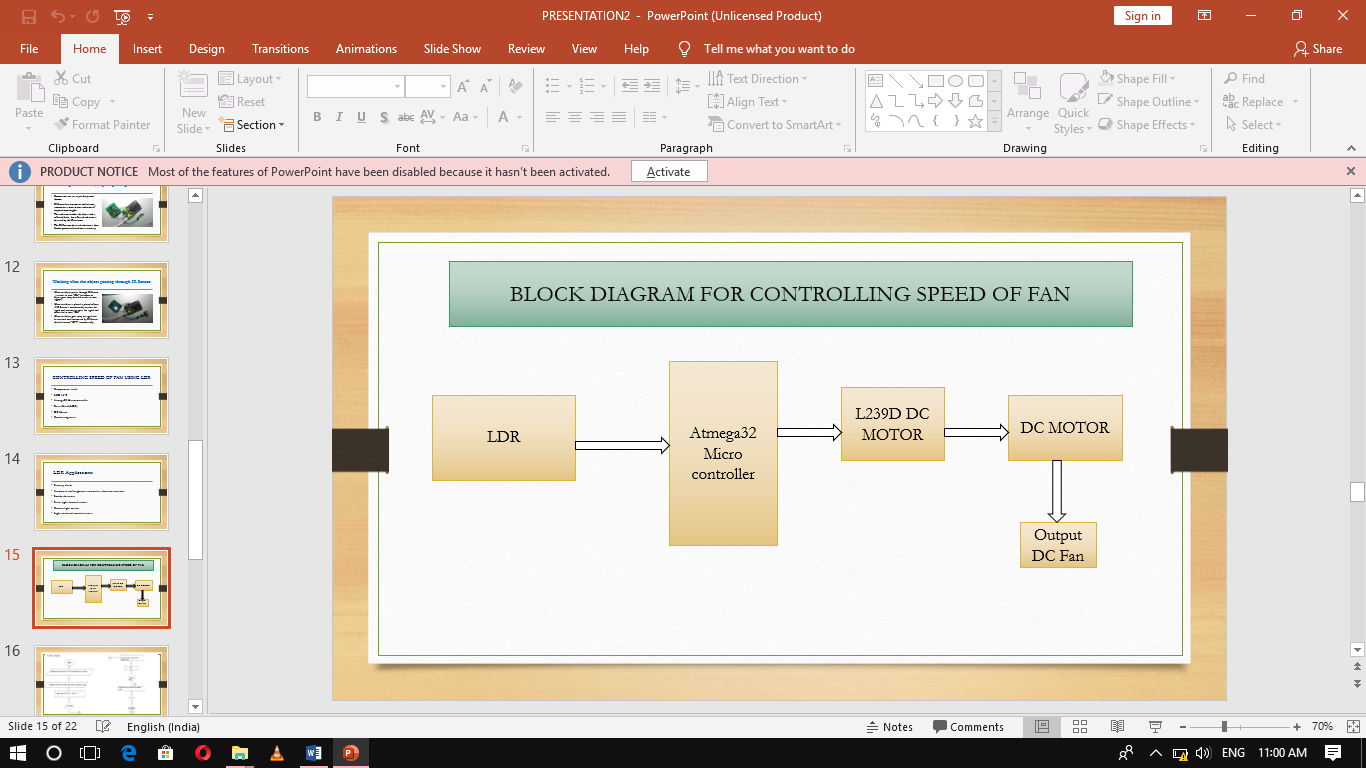
1. LCD 16\*2 display

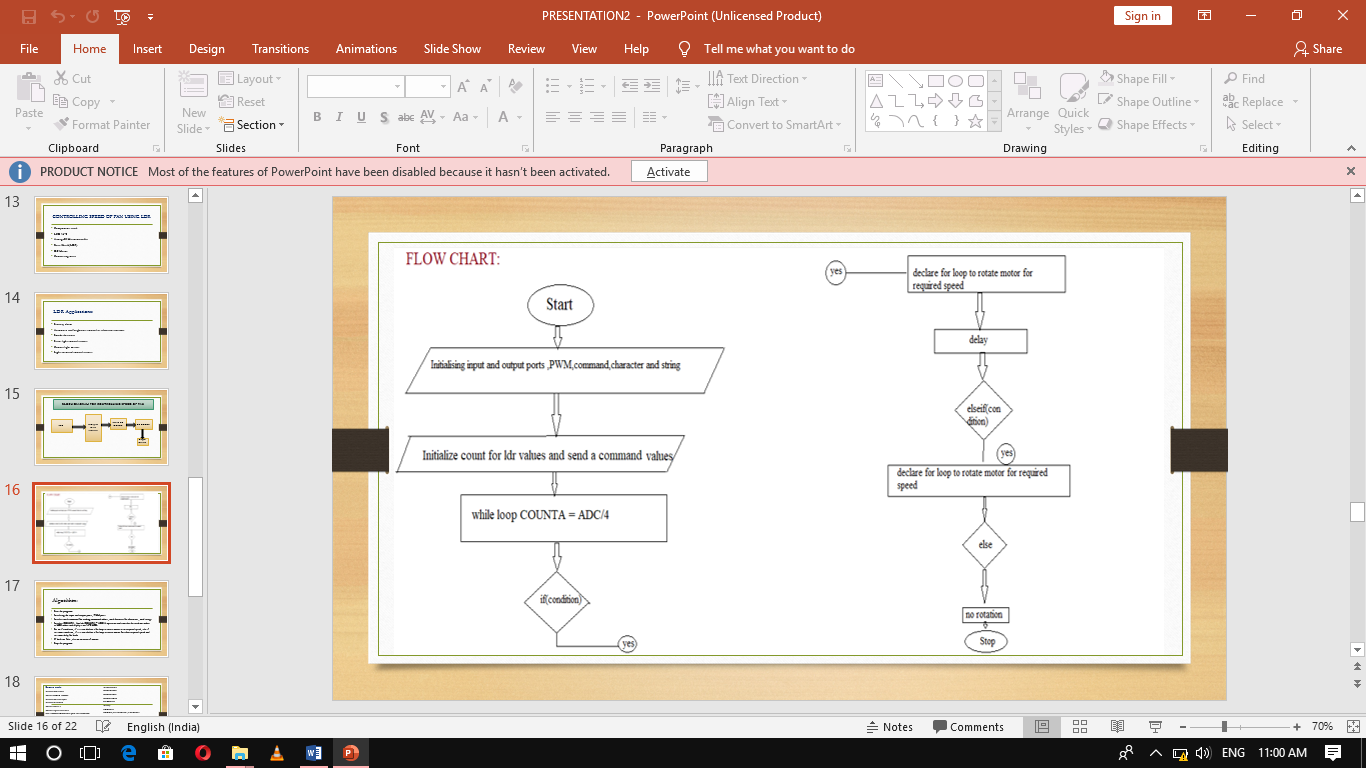
2. Atmega32 Micro controller

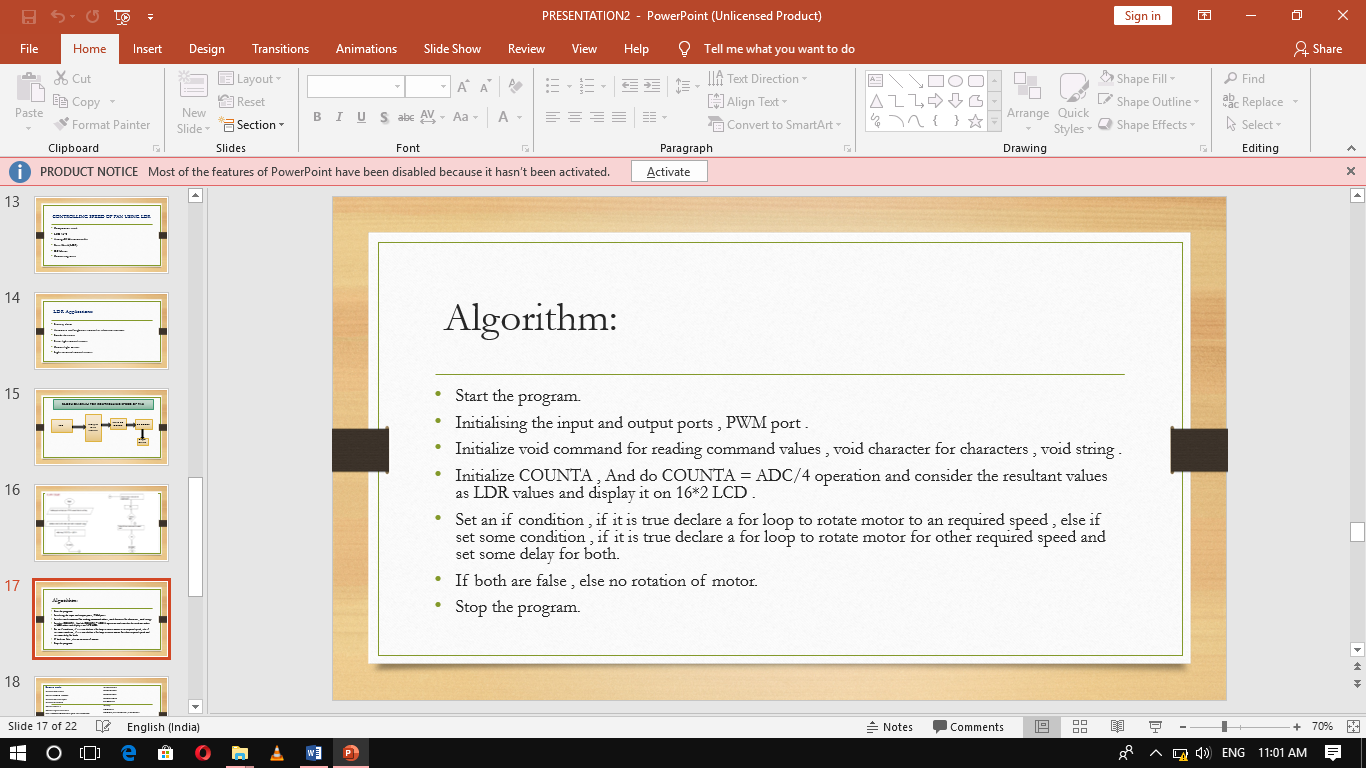
3. L239D motor driver

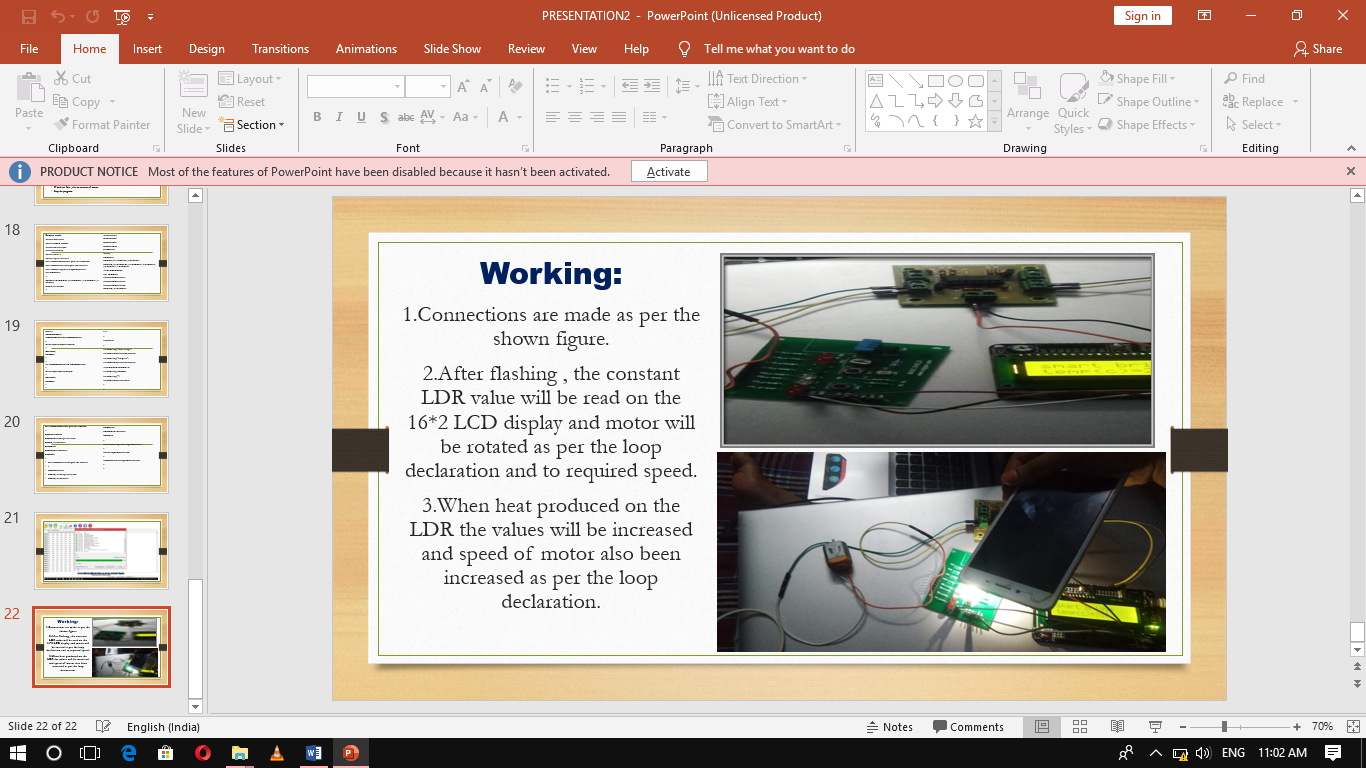
4. DC Motor

5. Connecting wires









Source Code:

#include<avr/io.h>

#define *F\_CPU* 1000000

#include<util/delay.h>

#include<stdlib.h>

#define enable 5

#define registerselection 7

void send\_a\_command(unsigned char command);

void send\_a\_character(unsigned char character);

void send\_a\_string(char\*string\_of\_character);

void PWM\_init()

{

TCCR0=(1<<WGM00)|(1<<WGM01)|(1<<COM01)|(1<<CS00);

DDRB|=(1<<PB3);

}

int main(void)

{

DDRC=0XFF;

DDRA=0X00;

DDRD=0XFF;

*\_delay\_ms*(50);

int duty;

PWM\_init();

ADMUX|=(1<<REFS0)|(1<<REFS1);

ADCSRA|=(1<<ADEN)|(1<<ADATE)|(1<<ADPS0)|(1<<ADPS1)|(1<<ADPS2);

*int16\_t* COUNTA=0;

char SHOWA[3];

send\_a\_command(0x01);

send\_a\_command(0x38);

send\_a\_command(0x0E);

ADCSRA|=(1<<ADSC);

while(1)

{

COUNTA=ADC/4;

if((COUNTA>=255) && (COUNTA<=399))

{

for( int duty=100;duty<255;duty++)

{

OCR0=duty;

*\_delay\_ms*(8);

}

}

else if((COUNTA>=400) && (COUNTA<=500))

{

for( int duty=0;duty<255;duty++)

{

OCR0=duty;

*\_delay\_ms*(8);

}

}

else

{

duty=0x00;

}

send\_a\_string("smart bridge");

send\_a\_command(0x80+0x40+0);

send\_a\_string("temp(c)=");

send\_a\_command(0x80+0x40+8);

*itoa*(COUNTA,SHOWA,10);

send\_a\_string(SHOWA);

send\_a\_string("");

send\_a\_command(0x80+0);

}

}

void send\_a\_command(unsigned char command)

{

PORTC=command;

PORTD&=~(1<<registerselection);

PORTD|=(1<<enable);

*\_delay\_ms*(50);

PORTD&=~(1<<enable);

PORTC=0;

}

void send\_a\_character(unsigned char character)

{

PORTC=character;

PORTD|=(1<<registerselection);

PORTD|=(1<<enable);

*\_delay\_ms*(50);

PORTD&=~(1<<enable);

PORTC=0;

}

void send\_a\_string(char\*string\_of\_characters)

{

while(\*string\_of\_characters>0)

{

send\_a\_character(\*string\_of\_characters++);

}

}

Conclusion:

Hence the conclusion is that we can control the fan of the speed by using LDR.if the temperature increases the speed of the fan automatically increases . if temperature decreases automatically the speed of fan decreases.