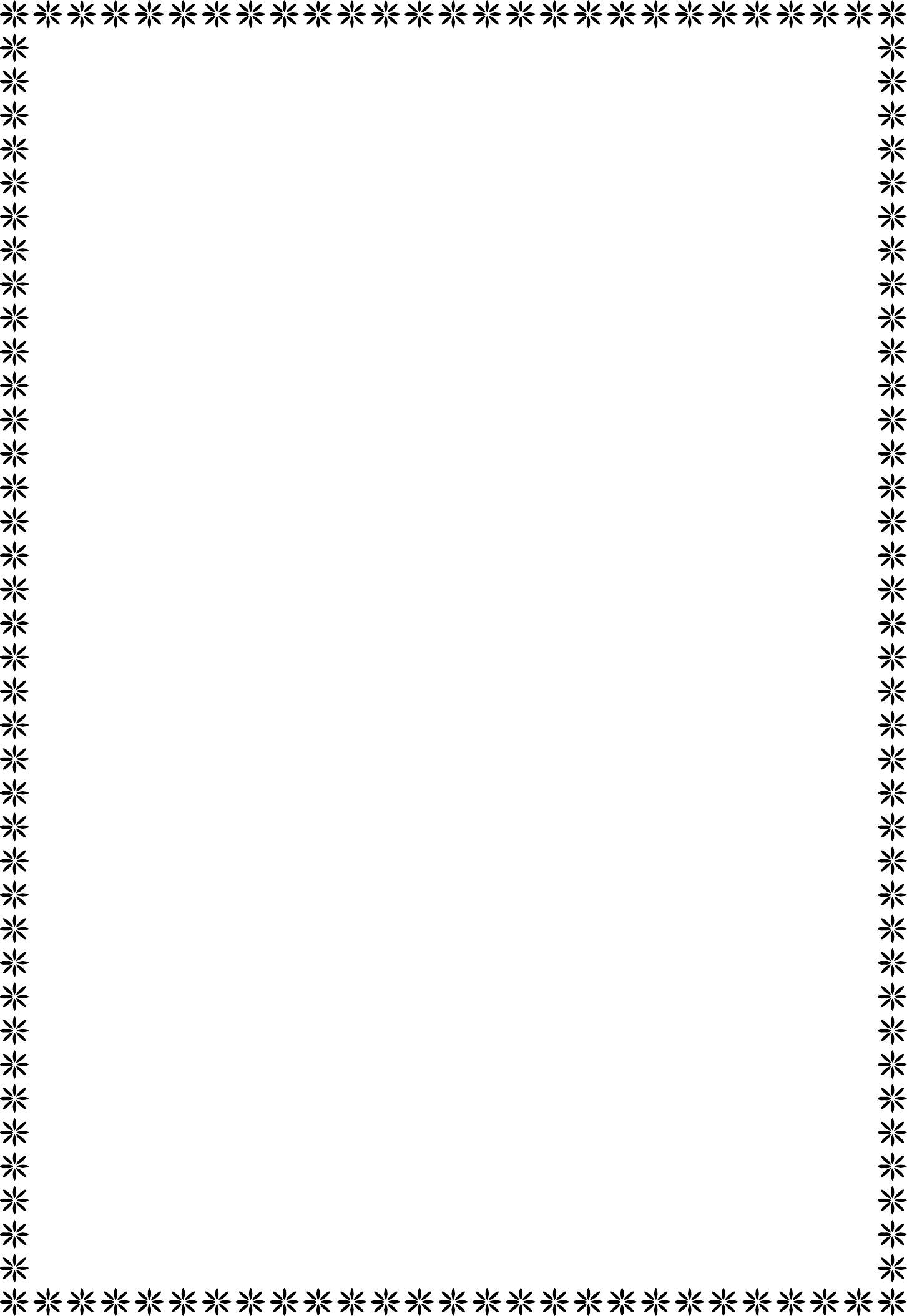
SONA COLLEGE OF TECHNOLOGY



Junction Main Road,Sona Nagar, Suramangalam(P.O) SALEM-636005

Name: TAMILSELVAN K

Branch: CSE

Semester: VI

Register No: 1518102145

Certified that this is the Bona-fide record of work done by above student in U15CS604R INTERNET OF THINGS LABORATORY During the year 2020- 2021

Lab-in-charge Head of the department

Submitted for the University Practical Examination held on

Internal Examiner External Examiner

# LIST OF EXPERIMENTS

1. Turn ON and OFF the LEDs.

2. Identify the objects using IR and PIR sensor.

3. Measure the moisture level of soil using soil moisture sensor.

4. Measure the distance between the ultrasonic sensor and the obstacle.

5. Identify the leakage of gas/smoke in the environment.

6. Measure the humidity and moisture value of the environment.

7. Control a LED using relay switch.

**MINI PROJECT**

Build an IoT system for the following suggested titles but not limited to:

8. Line follower robot

9. Smart weather monitoring system

10. Smart lighting system

11. Smart waste management system

12. Smart parking system

# LIST OF EXERCISES

|  |  |  |
| --- | --- | --- |
| **S.NO** | **DATE** | **Name of an Experiment** |
| 1 | 11-02-2021 | Turn ON and OFF the LEDs. |
| 2 | 25-02-2021 | Identify the objects using IR and PIR sensor. |
| 3 | 04-03-2021 | Measure the moisture level of soil using soil moisture sensor. |
| 4 | 18-03-2021 | Measure the distance between the ultrasonic sensor and the obstacle. |
| 5 | 24-03-2021 | Built-in Functions, set operations and various Join operation |
| 6 | 07-04-2021 | Identify the leakage of gas/smoke in the environment. |
| 7 | 14-04-2021 | Control a LED using relay switch. |
| **Mini Project** | | |
| 8 | 05-05-2021 | Smart Building Automation Management System |

**Study and Install Arduino IDE**

**Arduino:**

Arduino is a prototype platform (open-source) based on an easy-to-use hardware and software. It consists of a circuit board, which can be programmed (referred to as a microcontroller) and ready-made software called Arduino IDE (Integrated Development Environment), which is used to write and upload the computer code to the physical board.

Arduino provides a standard form factor that breaks the functions of the micro-controller into a more accessible package.

Arduino consists of various features that can be useful for the development of various projects.

The key features are −

* Arduino boards are able to read analog or digital input signals from different sensors and turn it into an output such as activating a motor, turning LED on/off, connect to the cloud and many other actions.
* The board functions can be controlled by sending a set of instructions to the microcontroller on the board via Arduino IDE (referred to as uploading software).
* Unlike most previous programmable circuit boards, Arduino does not need an extra piece of hardware (called a programmer) in order to load a new code onto the board. You can simply use a USB cable.
* Additionally, the Arduino IDE uses a simplified version of C++, making it easier to learn programming.

**Download the Arduino Software (IDE)**

* Get the latest version from the arduino.cc web site. Choice canbe between the Installer (.exe) and the Zip packages. Use of the first ones that installs directly is needed to use the Arduino Software (IDE), including the drivers. Zip package need to install the drivers manually. The Zip file is also useful to create a portable installation.
* When the download finishes, proceed with the installation and allow the driver installation process when a warning comes from the operating system.
* Choose the components to install



* Choose the installation directory (we suggest to keep the default one)





* The process will extract and install all the required files to execute the Arduino Software (IDE) properly.
* Proceed with board specific instructions
* The Arduino Software (IDE) will be installed.

**Different Arduino Boards:**

**1. Arduinouno**

This is the latest revision of the basic Arduino USB board. It connects to the computer with a standard USB cable and contains everything else needed to program and to use the board.

**2. Arduino Bluetooth**

The Arduino BT is a microcontroller board originally based on the ATmega168, but now is supplied with the 328, and the Bluegiga WT11 bluetooth module. It supports wireless serial communication over bluetooth.

**3. Arduino Mega**

The original Arduino Mega has an ATmega1280 and an FTDI USB-to-serial chip.

**4. Arduino NANO**

The Arduino Nano 3.0 has an ATmega328 and a two-layer PCB and the power LED moved to the top of the board.

**Study and configure Raspberry Pi.**

**Raspberry Pi:**

The **Raspberry Pi** is a series of small single-board computers developed in the United Kingdom by the Raspberry Pi Foundation to promote the teaching of basic computer science in schools and in developing countries. The original model became far more popular than anticipated, selling outside of its target market for uses such as robotics. Peripherals (including keyboards, mice and cases) are not included with the Raspberry Pi. Some accessories however have been included in several official and unofficial bundles.

According to the Raspberry Pi Foundation, over 5 million Raspberry Pi’s have been sold before February 2015, making it the best-selling British computer.

By November 2016 they had sold 11 million units, reaching 12.5m in March 2017, making it the third best-selling "general purpose computer" ever.

To get started, NOOBS (New Out Of Box Software) is an easy operating system install manager for the Raspberry Pi.

**How to get and install NOOBS:**

* ***DOWNLOAD NOOBS OS***

Use an SD card with a minimum capacity of 8GB.

1. GO to the https://www.raspberrypi.org/downloads/
2. Click on **NOOBS**, then click on the Download ZIP button under ‘NOOBS (offline and network install)’, and select a folder to save it to.
3. Extract the files from the zip.

* ***FORMAT YOUR SD CARD***

It is best to format the SD card before copying the NOOBS files onto it. To do this:

1. Download SD Formatter 4.0 for either Windows or Mac.
2. Follow the instructions to install the software.
3. Insert the SD card into the computer or laptop’s SD card reader and make a note of the drive letter allocated to it, e.g. G:/
4. In SD Formatter, select the drive letter for the SD card and format it.

* ***DRAG AND DROP NOOBS FILES***

1. Once the SD card has been formatted, drag all the files in the extracted NOOBS folder and drop them onto the SD card drive.
2. The necessary files will then be transferred to the SD card.
3. When this process has finished, safely remove the SD card and insert it into the Raspberry Pi.

* ***FIRST BOOT***

1. Plug in the keyboard, mouse, and monitor cables.
2. Now plug the USB power cable into the Pi.
3. The Raspberry Pi will boot, and a window will appear with a list of different operating systems that can beinstalled. Raspbianwill be recommended– tick the box next to Raspbian and click on Install.
4. Raspbian will then run through its installation process. Note that this can take a while.
5. When the install process has completed, the Raspberry Pi configuration menu (raspi-config) will load. Here you are able to set the time and date for your region, enable a Raspberry Pi camera board, or even create users. Then exit this menu by using **Tab** on the keyboard to move to Finish.

* ***LOGGING IN AND ACCESSING THE GRAPHICAL USER INTERFACE***

1. The default login for Raspbian is username pi with the password raspberry. **Note that no writings appear when the password is typed.** This is a security feature in Linux.

2. To load the graphical user interface, type startx and press **Enter**.

**Ex. No: 1 Turn ON and OFF the LEDs**

**Date:**

**Aim**: To implement a simple application using Arduino for Blink LED.

**Hardware Requirements:**

* 1x Breadboard
* 1x Arduino Uno R3/Raspberry Pi
* 1x LED
* 1x 330Ω Resistor
* 2x Jumper Wires

**Circuit connection for Arduino:**

Arduino pin D13 - 330 Ω Resistor

330 Ω Resistor – +ve leg of LED

Gnd - ve leg of LED



**Fig: 1.1 Circuit diagram for LED blink using Arduino**

Use the Figure: 1.1 to connect the Arduino with resistors and LED. Open the Arduino IDE software on the computer. Program in the Arduino language will control the circuit. Open a new sketch File by clicking New.

Now upload the 'Blink' example sketch into Arduino. Both the built-in 'L' LED and the external LED will be blinking.

**Program:**

/\*

Blink

Turns on an LED for one second, then off for one second, repeatedly.

This example code is in the public domain.

\*/

//pin 13 has an LED connected on most Arduino boards.

// give it a name:

int led = 13;

//the setup routine runs once when you press reset:

void setup() {

// initialize the digital pin as an output.

pinMode(led, OUTPUT);

}

// the loop routine runs over and over again forever:

Void loop() {

digitalWrite(led, HIGH); // turn the LED on( HIGH is the voltage level)

delay(1000); // wait for a second

digitalWrite(led, LOW) //turn the LED off by making the voltage LOW

delay(1000) //wait for a second

}

//same output can be obtained using different pin of the Arduino – say D7.

Move the red jumper lead from pin D13 to pin D7 and modify the following line near the top of the sketch:

*int led = 13;*

So that it reads.

*int led = 7;*

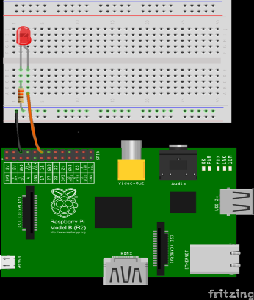
Upload the modified sketch to your Arduino board and the LED should still be blinking, but this time using pin D7.

**Circuit connection for Raspberry pi:**

GPIO 10 - 330 Ω Resistor

330 Ω Resistor – +ve leg of LED

Gnd - \_ve leg of LED

**

**Fig: 1.2 Circuit diagram for LED blink using Raspberry pi**

importRPi.GPIO as GPIO # import GPIO

GPIO.setmode(GPIO.BOARD) # set Board mode

GPIO.setup(10, GPIO.OUT) #set the pin 10 as output

GPIO.output(10, True) #LED glow

time.sleep(2) # delay for 2 seconds

GPIO.output(10, False) #LED off

**Result:**

A simple Arduino based application for blinking LEDs has been successfully implemented and verified.

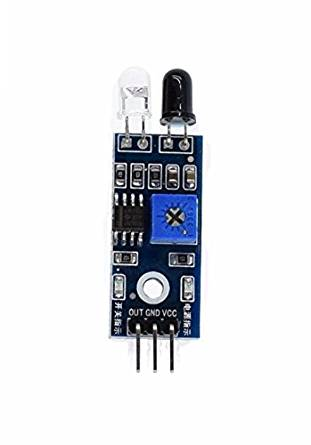
**Ex.No: 2 Identify the objects using IR and PIR sensor**

**Date:**

**Aim**: To identify the objects using IR (Infrared) and PIR sensors.

**IR sensors:**

An infrared sensor which is shown in fig 2.1 is an electronic instrument which is used to sense certain characteristics of its surroundings by either emitting (through transmitter) and/or detecting (through receiver) infrared radiation. Infrared sensors are also capable of measuring the heat being emitted by an object and detecting its motion.



**Fig: 2.1 IR sensor**

**PIR sensors:**

PIR sensor which is shown in fig 2.2 allows to sense motion. They are used to detect whether a human has moved in or out of the sensor's range. They are commonly found in appliances and gadgets used at home or for businesses. They are often referred to as PIR, "**Passive Infrared**", "Pyroelectric", or "IR motion" sensors.



**Fig:2.2 PIR sensor**

PIRs are made of pyroelectric sensors; a round metal can with a rectangular crystal in the center, which can detect levels of infrared radiation. Everything emits low-level radiation, and the hotter something is, the more radiation is emitted. The sensor in a motion detector is split in two halves. This is to detect motion (change) and not average IR levels. The two halves are connected so that they cancel out each other. If one-half sees more or less IR radiation than the other, the output will swing high or low.

PIRs have adjustable settings and have a header installed in the 3-pin ground/out/power pads. For many basic projects or products that need to detect when a person has left or entered the area, PIR sensors are great. Note that PIRs do not tell you the number of people around or their closeness to the sensor. The lens is often fixed to a certain sweep at a distance and they are sometimes set off by the pets in the house.

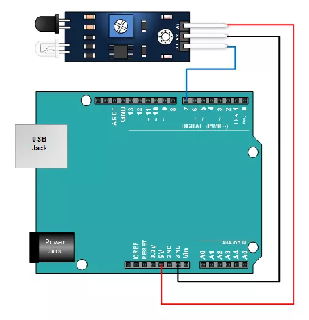
## Hardware Requirements:

* 1 × Breadboard
* 1 × Arduino Uno R3
* 1 × IR sensor/PIR Sensor

**Circuit connection for IR sensor:**

IR sensor has three terminals - Vcc, OUT and GND. Connect the sensor as specified in fig: 2.1

* Connect the +Vcc to +5v on Arduino board.
* Connect OUT to digital pin 7 on Arduino board.
* Connect GND with GND on Arduino.



**Fig: 2.3 Circuit connection for object detection using IR sensor**

IR sensor:

// the setup routine runs once when you press reset:

int IRpin = 2;

void setup() {

// initialize serial communication at 9600 bits per second:

Serial.begin(9600);

pinMode(IRpin, INPUT);

}

// the loop routine runs over and over again forever:

void loop() {

// read the input on analog pin 0:

int IRsensorValue = digitalRead(IRpin);

// print out the value you read:

Serial.println(IRsensorValue);

if(IRsensorValue==HIGH)

Serial.println(“Object detected”);

else

Serial.println(“Object not detected”);

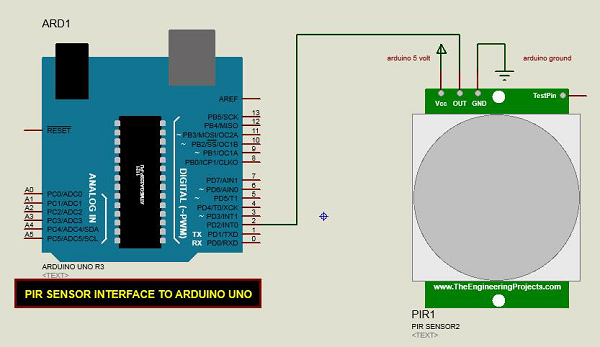
delay(1); // delay in between reads for stability

}

**Circuit connection for PIR sensor:**

PIR sensor has three terminals - Vcc, OUT and GND. Connect the sensor as specified in fig 2.2

* Connect the +Vcc to +5v on Arduino board.
* Connect OUT to digital pin 2 on Arduino board.
* Connect GND with GND on Arduino.



**Fig:2.4Circuit connection for object detection using PIR sensor**

**Program:**

PIR sensor:

// the setup routine runs once when you press reset:

int PIRpin = 2;

void setup() {

// initialize serial communication at 9600 bits per second:

Serial.begin(9600);

pinMode(PIRpin, INPUT);

}

// the loop routine runs over and over again forever:

void loop() {

// read the input on analog pin 0:

int PIRsensorValue = digitalRead(PIRpin);

// print out the value you read:

Serial.println(PIRsensorValue);

if(PIRsensorValue==HIGH)

Serial.println(“Object detected”);

else

Serial.println(“Object not detected”);

delay(1); // delay in between reads for stability

}

**Result:**

A simple Arduino-based application that uses IR (infrared) and PIR sensors to identify objects has been successfully implemented and verified.

**Ex.no: 3 Measure the moisture level of soil using soil moisture sensors**

**Date:**

**Aim**:To measure the moisture level of soil using Arduino soil moisture sensor.

This sensor measures the volumetric content of water inside the soil and gives the moisture level as output. The sensor is equipped with both analog and digital output, so it can be used in both analog and digital mode. In this article, we are going to interface the sensor in both modes.

**Hardware Requirements**:

**Hardware Requirements:**

* 1x Breadboard
* 1x Arduino Uno R3
* 1x Soil moisture sensor
* Jumper Wires

The specifications of the soil moisture sensor FC-28 are as follows

* Input voltage 3.3-5V
* Output voltage 0-4.2V
* Input current 35mA
* Output signal Both Analog and Digital
* Pin Out – Soil Moisture Sensor

The soil Moisture sensor FC-28 has the following four pins:

* VCC: For power
* A0: Analog output
* D0: Digital output
* GND: Ground

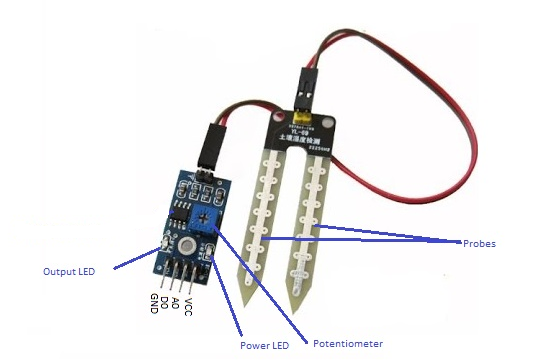
The module also contains a potentiometer which will set the threshold value.

**Working of Sensor:**

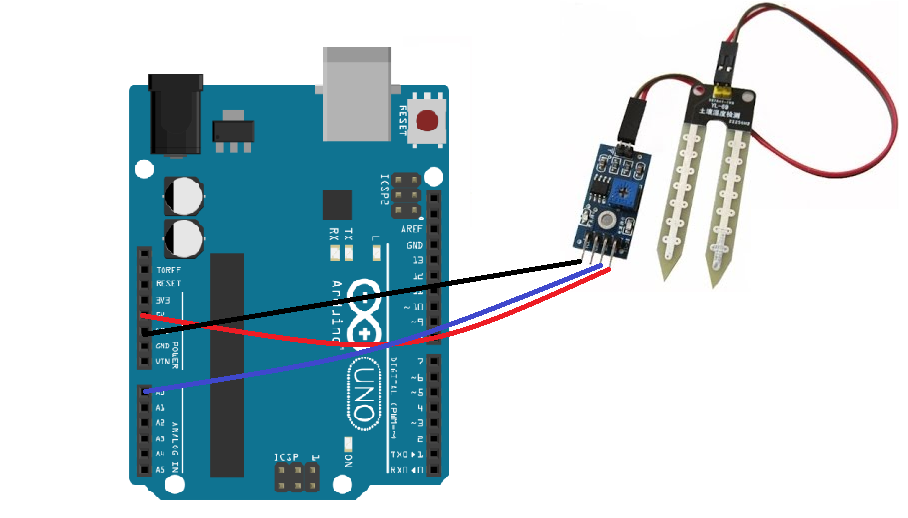
The soil moisture sensor consists of two probes which are used to measure the volumetric content of water. The two probes allow the current to pass through the soil and then it gets the resistance value to measure the moisture value.

When there is more water, the soil will conduct more electricity which means that there will be less resistance. Therefore, the moisture level will be higher. Dry soil conducts electricity poorly, so when there will be less water, then the soil will conduct less electricity which means that there will be more resistance. Therefore, the moisture level will be lower.

This sensor can be connected in two modes; Analog mode orDigital mode. First, connect it in Analog mode and then use it in Digital mode.



**Fig: 3.1 Soil moisture sensor**



**Fig:3.2 Circuit connection for Soil moisture sensor using Arduino**

**Program**:

intsensor\_pin = A0; //Soil Sensor input at Analog PIN A0

void setup() {

   Serial.begin(9600);

   Serial.println("Reading From the Sensor ...");

   delay(2000);}

void loop() {

   output\_value= analogRead(sensor\_pin);

   Serial.print("Mositure : ");

     Serial.println("%");

   delay(1000);

}

**Code Explanation**

In the setup function, the “Serial.begin(9600)” command will help in communication between the Arduino and serial monitor. Then, print the “Reading from the Sensor …” on the serial monitor.

In the loop function, read from the sensor analog pin and will store the values in the “output\_ value” variable. Then, map the output values to 0-100, because the moisture is measured in percentage. When talking the readings from the dry soil, then the sensor value will be 550 and in the wet soil, the sensor value was 10. So, the mapped these values to get the moisture. After that, print these values on the serial monitor.

**Result:**

A simple Arduino based application for measuring the moisture level of soil using soil moisture sensor has been successfully implemented and verified.

**Ex.no: 4 Measure the distance between the ultrasonic sensor and the obstacle**

**Date:**

**Aim:** To measure the distance between the ultrasonic sensor and the obstacle.

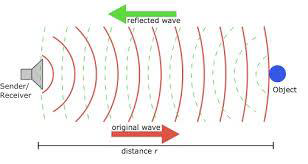
The HC-SR04 ultrasonic sensor uses ultrasonic sound waves to determine the distance of an object. It offers excellent non-contact range detection with high accuracy and stable readings in an easy-to-use package from 2 cm to 400 cm or 1 foot to 13 feet.

The operation is not affected by sunlight or black material, although acoustically, soft materials like cloth can be difficult to detect. It comes complete with ultrasonic transmitter and receiver module.



**Fig: 4.1 Ultra sonic sensor**

The sensor head emits an ultrasonic wave and receives the wave reflected back from the target. Ultrasonic Sensors measure the distance to the target by measuring the time between the emission and reception. (Refer fig 4.1)



**Fig:4.2Working principle of HC-SR04**

**Outline and detection principle:**

An optical sensor has a transmitter and receiver, whereas an ultrasonic sensor uses a single ultrasonic element for both emission and reception. In a reflective model ultrasonic sensor, a single oscillator emits and receives ultrasonic waves alternately. This enables miniaturization of the sensor head.

**Distance calculation:**

The distance can be calculated with the following formula:

Distance L = 1/2 × T × C, where L is the distance, T is the time between the emission and reception, and C is the sonic speed. (The value is multiplied by 1/2 because T is the time for go-and-return distance.)

**Features:**

The following list shows typical characteristics enabled by the detection system.

[Transparent object detectable]

Since ultrasonic waves can reflect off a glass or liquid surface and return to the sensor head, even transparent targets can be detected.

[Resistant to mist and dirt]

Detection is not affected by accumulation of dust or dirt.

[Complex shaped objects detectable]

Presence detection is stable even for targets such as mesh trays or springs.

## Hardware Requirements:

* 1 × Breadboard
* 1 × Arduino Uno R3/Raspberry pi
* 1 × ULTRASONIC Sensor (HC-SR04)

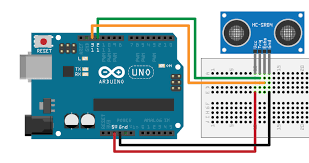
In our program, we have displayed the distance measured by the sensor in inches and cm via the serial port.

## Circuit connection for Arduino:

Follow the circuit diagram and make the connections as shown in the image given below.

The Ultrasonic sensor has four terminals - +5V, Trigger, Echo, and GND connected as in fig 4.2

* Connect the +5V pin to +5v on your Arduino board.
* Connect Trigger to digital pin 7 on your Arduino board.
* Connect Echo to digital pin 6 on your Arduino board.
* Connect GND with GND on Arduino.



**Fig:4.2 Circuit connection for HC-SR04 using Arduino**

**Program for Arduino**:

/\*

\* Ultrasonic Sensor HC-SR04 interfacing with Arduino.

\*/

// defining the pins

constinttrigPin = 9;

constintechoPin = 10;

// defining variables

long duration;

int distance;

void setup() {

pinMode(trigPin, OUTPUT); // Sets the trigPin as an Output

pinMode(echoPin, INPUT); // Sets the echoPin as an Input

Serial.begin(9600); // Starts the serial communication

}

void loop() {

// Clears the trigPin

digitalWrite(trigPin, LOW);

delayMicroseconds(2);

// Sets the trigPin on HIGH state for 10 micro seconds

digitalWrite(trigPin, HIGH);

delayMicroseconds(10);

digitalWrite(trigPin, LOW);

// Reads the echoPin, returns the sound wave travel time in microseconds

duration = pulseIn(echoPin, HIGH);

// Calculating the distance

distance= duration\*0.034/2;

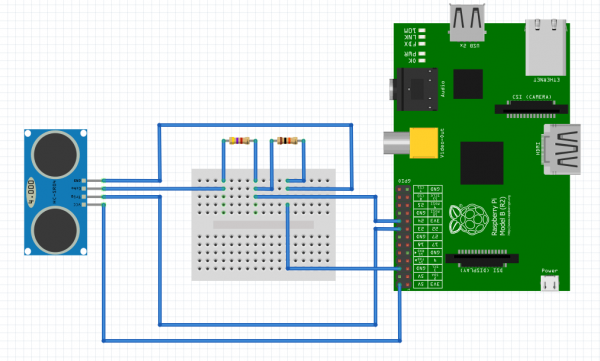
// Prints the distance on the Serial Monitor

Serial.print("Distance: ");

Serial.println(distance);

}

**Connection diagram for Raspberry pi:**



**Fig:4.3 Circuit connection for HC-SR04 using Raspberry pi**

**Program for Raspberry pi:**

import RPI.GPIO as GPIO #Import GPIO library

import time #Import time library

GPIO.setmode(GPIO.BCM) #Set GPIO pin numbering

TRIG = 23 #Associate pin 23 to TRIG

ECHO = 24 #Associate pin 24 to ECHO

print "Distance measurement in progress"

GPIO.setup(TRIG,GPIO.OUT) #Set pin as GPIO out

GPIO.setup(ECHO,GPIO.IN) #Set pin as GPIO in

while True:

GPIO.output(TRIG, False) #Set TRIG as LOW

print "Waitng For Sensor To Settle"

time.sleep(2) #Delay of 2 seconds

GPIO.output(TRIG, True) #Set TRIG as HIGH

time.sleep(0.00001) #Delay of 0.00001 seconds

GPIO.output(TRIG, False) #Set TRIG as LOW

whileGPIO.input(ECHO)==0: #Check whether the ECHO is LOW

pulse\_start = time.time() #Saves the last known time of LOW pulse

whileGPIO.input(ECHO)==1: #Check whether the ECHO is HIGH

pulse\_end = time.time() #Saves the last known time of HIGH pulse

pulse\_duration = pulse\_end - pulse\_start #Get pulse duration to a variable

distance = pulse\_duration \* 0.034/2 #Multiply pulse duration by 17150 to get distance

distance = round(distance, 2) #Round to two decimal points

print "Distance:",distance #Print distance

***Result:***

A simple Arduino based application for measuring the distance between the ultrasonic sensor and the obstacle has been successfully implemented and verified.

**Ex.no: 5 Identify the leakage of gas/smoke in the environment**

**Date:**

**Aim:**To identify the leakage of gas/smoke in the environment using smoke sensor.

**MQ- 6 Smoke Sensor:**

The smoke sensor is the MQ-6. This is a sensor that is not only sensitive to smoke, but also to flammable gas.The MQ-6 smoke sensor reports smoke by the voltage level that it outputs. The more smoke there is, the greater the voltage that it outputs. Conversely, the less smoke that it is exposed to, the less voltage it outputs.



**Fig:5.1 MQ- 6 smoke sensor**

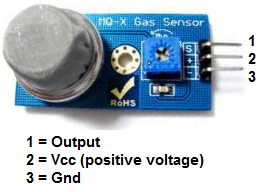
The MQ-6 also has a built-in potentiometer to adjust the sensitivity to smoke. By adjusting the potentiometer, it is possible to change how sensitive it is to smoke, so it is a form of calibrating it to adjust how much voltage it will put out in relation to the smoke it is exposed to.

Wire the MQ-6 to an Arduino so that the Arduino can read the amount of voltage output by the sensor and sound a buzzer if the sensor outputs a voltage above a certain threshold. This way, the sensor is detecting smoke and sounds a buzzer alerting a person such as a homeowner to this fact.

**Hardware Requirements**

* 1x MQ-6 Smoke Sensor
* 1x Arduino board
* 1x Buzzer

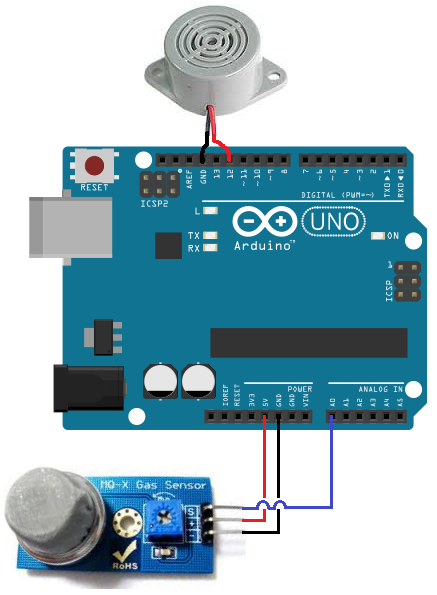
The complete boardwith 3 leads shown in fig.5.1



**Fig:5.2 MQ 6 with three leads**  
  
The 3 leads are Output, Vcc, and GND.

The gas sensor needs about 5 volts of power in order to operate. This is done by connecting 5 volts to Vcc. The Output pin gives out the voltage reading, which is proportional to the amount of smoke that the sensor is exposed to. Again, a high voltage output means the sensor is exposed to a lot of smoke. A low or 0 voltage output means the sensor is exposed to either little or no smoke.

**Circuit diagram:**



**Fig: 5.3 Circuit connection for MQ6 using Arduino**

So to power the smoke sensor, connect pin 2 of the smoke sensor to the 5V terminal of the Arduino and terminal 3 to the GND terminal of the Arduino. This gives the smoke sensor the 5 volts it needs to be powered. The output of the sensor goes into analog pin A0 of the Arduino. Through this connection, the Arduino can read the analog voltage output from the sensor. The Arduino board has a built-in analog-to-digital converter, so it is able to read analog values without any external ADC chip.

Depending on the value that the Arduino reads, it determines the action that will occur with the circuit. If the sensor outputs a voltage above a certain threshold, the buzzer will go off, alerting a user that smoke has been detected.

These are all the physical connections in order for the circuit to work.

**Code for the Arduino MQ-6 Smoke Sensor Circuit**

The code upload is shown below.

/\*Code for MQ-6 Smoke Sensor Circuit Built with an Arduino Board\*/  
  
constintsensorPin= 0;  
constintbuzzerPin= 12;  
intsmoke\_level;  
  
void setup() {  
Serial.begin(115200); //sets the baud rate for data transfer in bits/second  
pinMode(sensorPin, INPUT);//the smoke sensor will be an input to the arduino  
pinMode(buzzerPin, OUTPUT);//the buzzer serves an output in the circuit  
}  
  
void loop() {  
smoke\_level= analogRead(sensorPin); //arduino reads the value from the smoke sensor  
Serial.println(smoke\_level);//prints just for debugging purposes, to see what values the sensor is picking up  
if(smoke\_level> 200){ //if smoke level is greater than 200, the buzzer will go off  
digitalWrite(buzzerPin, HIGH);  
}  
else{  
digitalWrite(buzzerPin, LOW);  
}  
}

**Code Explanation:**

The first block of code declares and initializes 3 variables. The sensor Pin represents the smoke sensor. It is initialized to 0, because it will be connected to analog pin A0 of the Arduino board. The next variable, buzzer Pin, represents the pin that the anode of the buzzer will be connected to; it is initialized to 12 because it will be connected to digital pin D12 of the Arduino board. And the variable, smoke\_level, represents the amount of smoke that the smoke sensor picks up.

The next block of code defines the baud rate and the input and output of the circuit. The sensorPin, which is the smoke sensor pin, serves as the input of the circuit. This sensor is input into the Arduino so that the Arduino can read and process the value. The buzzerPin serves as the output. If the smoke level is above a certain threshold, the output of the circuit, the buzzer, will go off.

The next block of code uses the analogRead() function to read the value from the sensorPin (the smoke sensor). This will be a numerical value from 0 to 1023. 0 represents no smoke, while 1023 represents smoke at the absolute maximum highest level. So the variable, smoke\_level, represents the smoke level that can range from 0 to 1023. In the code, if the smoke level rises above 200, triggering the buzzer to sound is done by sending the digital pin D12 high. So 200 is the threshold level. If the smoke level is below this value, then the buzzer does not go off.

This last block of code was the loop() function. This is the part of code that repeats over and over in an infinite loop. This means that the code always checks to see what the smoke\_level is, so that it can know whether to trigger the buzzer or not.

And this is how a smoke sensor works with an MQ-6 and an Arduino.

**Result:**

A simple Arduino based application for identifying the leakage of gas/smoke in the environment using smoke sensor has been successfully implemented and verified.

**Ex.no: 6 Measure the humidity and moisture value of the environment**

**Date:**

**Aim:**

To measure the humidity and moisture value of the environment using DHT11 sensor and Raspberry pi.

**DHT 11 sensor:**

The DHT11 is a low-cost temperature and humidity sensor. It isn’t the fastest sensor around but its cheap price makes it useful for experimenting or projects where you don’t require new readings multiple times a second. The device only requires three connections to the Pi. +3.3v, ground and one GPIO pin.



**Fig: 6.1 DHT 11 sensor**

The 4-pin device will require a resistor (4.7K-10K) to be placed between Pin 1 (3.3V) and Pin 2 (Data).The 3-pin modules will usually have this resistor included which makes the wiring a bit easier. The 3 pins should be connected to the Pi as shown in the table below:

| DHT Pin | Signal | Pi Pin |
| --- | --- | --- |
| 1 | 3.3V | 1 |
| 2 | Data/Out | 11 (GPIO17) |
| 3 | not used | – |
| 4 | Ground | 6 or 9 |

## Python Library

The DHT11 requires a specific protocol to be applied to the data pin. In order to save time trying to implement this yourself it’s far easier to use the Adafruit DHT library.

The library deals with the data that needs to be exchanged with the sensor but it is sensitive to timing issues. The Pi’s operating system may get in the way while performing other tasks so to compensate for this the library requests a number of readings from the device until it gets one that is valid.

## Software Setup

To start with update your package lists and install a few Python libraries :

**sudo apt-get update**

**sudo apt-get install build-essential python-dev**

Then clone the Adafruitlibrary from their repository :

**git clone https://github.com/adafruit/Adafruit\_Python\_DHT.git**

**cdAdafruit\_Python\_DHT**

Then install the library for Python 2 and Python 3 :

**sudo python setup.py install**

**sudo python3 setup.py install**

Hopefully at this point the library is installed and ready to be used within a Python script.

## Adafruit Example Python Script

Adafruit provide an example script that you can use to check your sensor is operating correctly.

**cd ~**

**cdAdafruit\_Python\_DHT**

**cd examples**

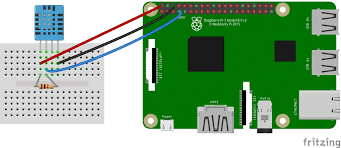
Then :

**python AdafruitDHT.py 11 17**

The example script takes two parameters. The first is the sensor type so is set to “11” to represent the DHT11. The second is the GPIO number so for my example I am using “17” for GPIO17. You can change this if you are using a different GPIO pin for your data/out wire.

Sample output:

Temp=22.0\* Humidity=68.0%



**Fig:6.2 Circuit connection for DHT 11 using Raspberry pi**

**Circuit connection:**

VCC of DHT11 -> 5v Pin of Raspberry Pi 3

GND of DHT11 -> GND Pin of Raspberry Pi 3

Signal pin of DHT11 -> GPIO 4 Pin of Raspberry Pi3

Program:

#!/usr/bin/python

import sys

importAdafruit\_DHT

while True:

humidity, temperature = Adafruit\_DHT.read\_retry(11, 4)

print 'Temp: {0:0.1f} C Humidity: {1:0.1f} %'.format(temperature, humidity)

**Result:**

A simple application for measuring the humidity and moisture value of the environment using DHT11 sensor and Raspberry pi has been successfully implemented and verified.

**Ex.No:7 Control a LED using relay switch**

**Date**:

**Aim:** To control a LED using a relay switch and Bluetooth.

**Relay:**

Relay is an electromagnetic switch, which is controlled by small current, and used to switch ON and OFF relatively much larger current. Means by applying small current we can switch ON the relay which allows much larger current to flow. A relay is a good example of controlling the AC (alternate current) devices, using a much smaller DC current.  Commonly used Relay is Single Pole Double Throw (SPDT) Relay, it has five terminals as in fig 7.1

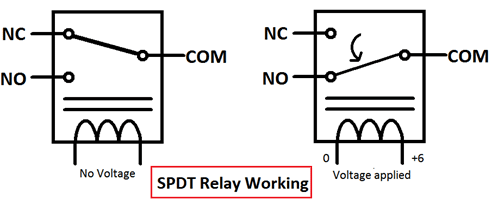


Fig:7.1

When there is no voltage applied to the coil, COM (common) is connected to NC (normally closed contact). When there is some voltage applied to the coil, the electromagnetic field produced, which attracts the Armature (lever connected to spring), and COM and NO (normally open contact) gets connected, which allow a larger current to flow. Relays are available in many ratings, here we used 6V operating voltage relay, which allows 7A-250VAC current to flow.

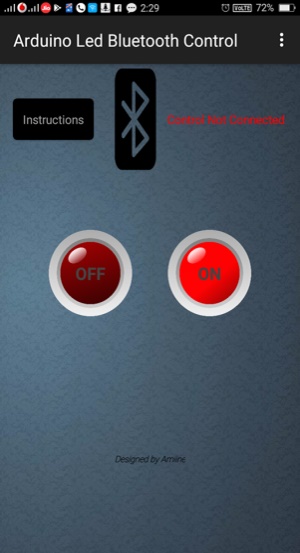
The relay is always configured by using a small Driver circuit which consists a Transistor, Diode and a resistor. Transistor is used to amplify the current so that full current (from the DC source – 9v battery) can flow through a coil to fully energies it. The resistor is used to provide biasing to the transistor. And Diode is used to prevent reverse current flow, when the transistor is switched OFF. Every Inductor coil produces equal and opposite EMF when switched OFF suddenly, this may cause permanent damage to components, so Diode must be used to prevent reverse current. Here a 6V Relay module is used.



Fig:7.2 Relay switch

**Mobile app:**

The mobile app named, “**Arduino LED Bluetooth Control**”, available in the Google play store is to be downloaded and installed in the mobile. This mobile app can be used to switch on/off the LED.



**Fig: 7.3 Arduino LED Bluetooth Control app**

**Hardware Requirements:**

* 1x Breadboard
* 1x Arduino Uno R3
* 1x LED
* 1x 330Ω Resistor
* 1x 5v or 6v relay
* Jumper Wires

**Circuit connection:**

Arduino pin7 – +ve Relay switch

Arduino Gnd – -ve Relay switch

COM of Relay switch – Arduino Gnd

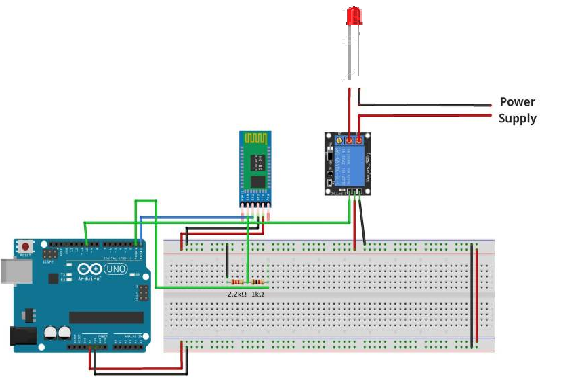
NO of relay switch – +ve of LED

-ve of LED – Arduino Gnd

Arduino pin Rx – Bluetooth pin Tx

Arduino pin Tx – Bluetooth pin Rx

**Circuit diagram:**



**Fig: 7.4 Circuit connection for LED control**

**Program:**

char data; //Variable for storing received data

void setup()

{

Serial.begin(9600); //Sets the data rate in bits per second (baud) for serial data transmission

pinMode(7, OUTPUT); //Sets digital pin 7 as output pin

}

void loop()

{

if(Serial.available() > 0) // Send data only when you receive data:

{

data = Serial.read(); //Read the incoming data and store it into variable data

Serial.print(data); //Print Value inside data in Serial monitor

Serial.print("\n"); //New line

if(data == '1') //Checks whether value of data is equal to 1

digitalWrite(7, HIGH); //If value is 1 then LED turns ON

else if(data == '0') //Checks whether value of data is equal to 0

digitalWrite(7, LOW); //If value is 0 then LED turns OFF

}

**Result:**

A simple application for controlling a LED using a relay switch and Bluetooth has been successfully implemented and verified

**Date: Mini Project**

**Smart Building Automation Management System**

This project presents the details of the installation of an energy management system in the buildings of a typical college campus in Salem and the consequent reduction in the electrical energy consumption. The electrical load demand in the universities is getting increased recently at an accelerated pace because of the rapid expansion of the campus infrastructure due to the increase in the intake of students, starting of new courses, research centres and laboratories. Internet of Things (IoT), i.e. the intercommunication of machines for an effective delivery of services with minimal human intervention, can form the basis of such systems. IoT based systems are gaining popularity due to the availability of highspeed mobile networks coupled with cost effective, accessible and fast embedded hardware. The system architecture takes care of the scenario where the devices are turned off when they are not in use. Being an entirely open source framework, a more deterministic control of security features can be attained by the user and can reduce the consecutive increase in CO2 emission from generation plant. Safety and security of any living or working place is one of the most primary concerns. One of the reasons for the rise of the smart building is the increasing risk of burglary and robbery and the busy lifestyle. We propose a methodology where the wireless sensor inputs the parameters to the Arduino Uno microcontroller like human detection, obstacle detection and keypad for the security. The microcontroller then uploads these values onto the cloud with the help of IOT. These values can be accessed by the concerned person through the android smart phone. The SBEMS is designed with an algorithm that is based on Hidden Markov Model (HMM) in order to estimate the probability of the building being in each of the above states.

**1. Introduction**

**1.1 Motivation of the project:**

So much of electricity is being wasted due to the unwanted usage of electrical appliances when they were not in use. Reduction in the consumption of electrical power by using Smart Building Automation Management System can reduce the usage of natural resource which ultimately reduces the pollution.

**1.2 Objective of the Project:**

* The objective of the project is to design a system with internet of things which controls the electrical appliances in a building through an android application
* The system consists of a wireless sensor network based on arduino, a mobile phone and a personal computer with internet connectivity.

The aim of the proposed work is to monitor and control the electrical devices used in buildings. This system allows authorized persons to remotely monitor and control devices in the building environment.

**2. System Analysis**

**2.1 Problem Statement**

Most of electrical energies are being wasted by the unwanted usage of electrical appliances when they are not in use (i.e) glowing of lights and working of Air conditionars when people were not in the building.

**2.2 Proposed System**

The paper offers a smart setting by connecting numerous sensors to the electrical devices which can be accessed remotely through an android application or over a web server and which can also provide real-time data visualization. The IOT layers used in this project are mentioned in the below figure 1..

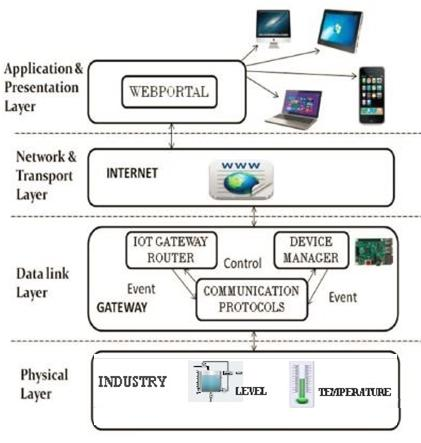


Fig. 1 Layers of IOT for the proposed system

**3. SYSTEM REQUIREMENTS**

**3.1 HARDWARE REQUIREMENTS**

* Arduino UNO
* PIR Sensor
* ESP-8266 WIFI Module
* Relay
* LCD Display

**3.1.1 HARDWARE DESCRIPTION**

**ARDUINO UNO:**

Arduino/Genuino Uno is a microcontroller board based on the [ATmega328P](http://www.atmel.com/Images/doc8161.pdf). It has 14 digital input/output pins (of which 6 can be used as PWM outputs), 6 analog inputs, a 16 MHz quartz crystal, a USB connection, a power jack, an ICSP header and a reset button. It contains everything needed to support the microcontroller; simply connect it to a computer with a USB cable or power it with a AC-to-DC adapter or battery to get started.. You can tinker with your UNO without worrying too much about doing something wrong, worst case scenario you can replace the chip for a few dollars and start over again.

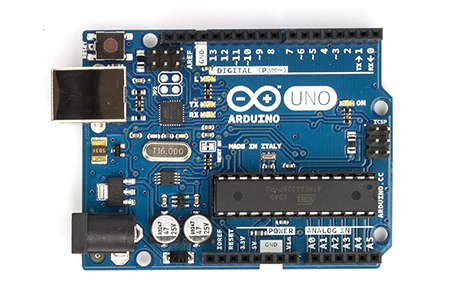
 

Fig. 2 Arduino UNO

"Uno" means one in Italian and was chosen to mark the release of Arduino Software (IDE) 1.0. The Uno board and version 1.0 of Arduino Software (IDE) were the reference versions of Arduino, now evolved to newer releases. The Uno board is the first in a series of USB Arduino boards, and the reference model for the Arduino platform; for an extensive list of current, past or outdated boards see the Arduino index of boards.

**Technical specs**

|  |  |
| --- | --- |
| Microcontroller | [ATmega328P](http://www.atmel.com/Images/doc8161.pdf) |
| Operating Voltage | 5V |
| Input Voltage (recommended) | 7-12V |
| Input Voltage (limit) | 6-20V |
| Digital I/O Pins | 14 (of which 6 provide PWM output) |
| PWM Digital I/O Pins | 6 |
| Analog Input Pins | 6 |
| DC Current per I/O Pin | 20 mA |
| DC Current for 3.3V Pin | 50 mA |
| Flash Memory | 32 KB (ATmega328P) of which 0.5 KB used by bootloader |
| SRAM | 2 KB (ATmega328P) |
| EEPROM | 1 KB (ATmega328P) |
| Clock Speed | 16 MHz |
| Length | 68.6 mm |
| Width | 53.4 mm |
| Weight | 25 g |

**Power**

The Arduino/Genuino Uno board can be powered via the USB connection or with an external power supply. The power source is selected automatically.

External (non-USB) power can come either from an AC-to-DC adapter (wall-wart) or battery. The adapter can be connected by plugging a 2.1mm center-positive plug into the board's power jack. Leads from a battery can be inserted in the GND and Vin pin headers of the POWER connector.

The board can operate on an external supply from 6 to 20 volts. If supplied with less than 7V, however, the 5V pin may supply less than five volts and the board may become unstable. If using more than 12V, the voltage regulator may overheat and damage the board. The recommended range is 7 to 12 volts.

The power pins are as follows:

* Vin. The input voltage to the Arduino/Genuino board when it's using an external power source (as opposed to 5 volts from the USB connection or other regulated power source). You can supply voltage through this pin, or, if supplying voltage via the power jack, access it through this pin.
* 5V. This pin outputs a regulated 5V from the regulator on the board. The board can be supplied with power either from the DC power jack (7 - 12V), the USB connector (5V), or the VIN pin of the board (7-12V). Supplying voltage via the 5V or 3.3V pins bypasses the regulator, and can damage your board. We don't advise it.
* 3V3 A 3.3 volt supply generated by the on-board regulator. Maximum current draw is 50 mA.
* GND. Ground pins.
* IOREF. This pin on the Arduino/Genuino board provides the voltage reference with which the microcontroller operates. A properly configured shield can read the IOREF pin voltage and select the appropriate power source or enable voltage translators on the outputs to work with the 5V or 3.3V.

**Memory**

The ATmega328 has 32 KB (with 0.5 KB occupied by the bootloader). It also has 2 KB of SRAM and 1 KB of EEPROM (which can be read and written with the [EEPROM library](https://www.arduino.cc/en/Reference/EEPROM)).

**Input and Output**

See the mapping between Arduino pins and ATmega328P ports. The mapping for the Atmega8, 168, and 328 is identical. Each of the 14 digital pins on the Uno can be used as an input or output, using [pinMode()](https://www.arduino.cc/en/Reference/PinMode),[digitalWrite()](https://www.arduino.cc/en/Reference/DigitalWrite), and [digitalRead()](https://www.arduino.cc/en/Reference/DigitalRead) functions. They operate at 5 volts. Each pin can provide or receive 20 mA as recommended operating condition and has an internal pull-up resistor (disconnected by default) of 20-50k ohm. A maximum of 40mA is the value that must not be exceeded on any I/O pin to avoid permanent damage to the microcontroller.

In addition, some pins have specialized functions:

* Serial: 0 (RX) and 1 (TX). Used to receive (RX) and transmit (TX) TTL serial data. These pins are connected to the corresponding pins of the ATmega8U2 USB-to-TTL Serial chip.
* External Interrupts: 2 and 3. These pins can be configured to trigger an interrupt on a low value, a rising or falling edge, or a change in value. See the attachInterrupt() function for details.
* PWM: 3, 5, 6, 9, 10, and 11. Provide 8-bit PWM output with the analogWrite() function.
* SPI: 10 (SS), 11 (MOSI), 12 (MISO), 13 (SCK). These pins support SPI communication using the SPI library.
* LED: 13. There is a built-in LED driven by digital pin 13. When the pin is HIGH value, the LED is on, when the pin is LOW, it's off.
* TWI: A4 or SDA pin and A5 or SCL pin. Support TWI communication using the Wire library.

The Uno has 6 analog inputs, labeled A0 through A5, each of which provide 10 bits of resolution (i.e. 1024 different values). By default they measure from ground to 5 volts, though is it possible to change the upper end of their range using the AREF pin and the analogReference() function. There are a couple of other pins on the board:

* AREF. Reference voltage for the analog inputs. Used with analogReference().
* Reset. Bring this line LOW to reset the microcontroller. Typically used to add a reset button to shields which block the one on the board.

**Communication**

Arduino/Genuino Uno has a number of facilities for communicating with a computer, another Arduino/Genuino board, or other microcontrollers. The ATmega328 provides UART TTL (5V) serial communication, which is available on digital pins 0 (RX) and 1 (TX). An ATmega16U2 on the board channels this serial communication over USB and appears as a virtual com port to software on the computer. The 16U2 firmware uses the standard USB COM drivers, and no external driver is needed. However,[on Windows, a .inf file is required](https://www.arduino.cc/en/Guide/Windows#toc4). The Arduino Software (IDE) includes a serial monitor which allows simple textual data to be sent to and from the board. The RX and TX LEDs on the board will flash when data is being transmitted via the USB-to-serial chip and USB connection to the computer (but not for serial communication on pins 0 and 1). A [SoftwareSerial library](https://www.arduino.cc/en/Reference/SoftwareSerial) allows serial communication on any of the Uno's digital pins.

The ATmega328 also supports I2C (TWI) and SPI communication. The Arduino Software (IDE) includes a Wire library to simplify use of the I2C bus; see the [documentation](https://www.arduino.cc/en/Reference/Wire) for details. For SPI communication, use the [SPI library](https://www.arduino.cc/en/Reference/SPI). Automatic (Software) Reset Rather than requiring a physical press of the reset button before an upload, the Arduino/Genuino Uno board is designed in a way that allows it to be reset by software running on a connected computer. One of the hardware flow control lines (DTR) of the ATmega8U2/16U2 is connected to the reset line of the ATmega328 via a 100 nanofarad capacitor. When this line is asserted (taken low), the reset line drops long enough to reset the chip. The Arduino Software (IDE) uses this capability to allow you to upload code by simply pressing the upload button in the interface toolbar. This means that the bootloader can have a shorter timeout, as the lowering of DTR can be well-coordinated with the start of the upload.

This setup has other implications. When the Uno is connected to either a computer running Mac OS X or Linux, it resets each time a connection is made to it from software (via USB). For the following half-second or so, the bootloader is running on the Uno. While it is programmed to ignore malformed data (i.e. anything besides an upload of new code), it will intercept the first few bytes of data sent to the board after a connection is opened. If a sketch running on the board receives one-time configuration or other data when it first starts, make sure that the software with which it communicates waits a second after opening the connection and before sending this data.

The Uno board contains a trace that can be cut to disable the auto-reset. The pads on either side of the trace can be soldered together to re-enable it. It's labeled "RESET-EN".

**LCD DISPLAY:**

Liquid crystal displays (LCDs) have materials which combine the properties of both liquids and crystals. Rather than having a melting point, they have a temperature range within which the molecules are almost as mobile as they would be in a liquid, but are grouped together in an ordered form similar to a crystal.

An LCD consists of two glass panels, with the liquid crystal material sand witched in between them. The inner surface of the glass plates are coated with transparent electrodes which define the character, symbols or patterns to be displayed polymeric layers are present in between the electrodes and the liquid crystal, which makes the liquid crystal molecules to maintain a defined orientation angle.

One each polarisers are pasted outside the two glass panels. These polarisers would rotate the light rays passing through them to a definite angle, in a particular direction.

When the LCD is in the off state, light rays are rotated by the two polarisers and the liquid crystal, such that the light rays come out of the LCD without any orientation, and hence the LCD appears transparent. When sufficient voltage is applied to the electrodes, the liquid crystal molecules would be aligned in a specific direction. The light rays passing through the LCD would be rotated by the polarisers, which would result in activating / highlighting the desired characters.

The LCD’s are lightweight with only a few millimeters thickness. Since the LCD’s consume less power, they are compatible with low power electronic circuits, and can be powered for long durations. The LCD’s don’t generate light and so light is needed to read the display. By using backlighting, reading is possible in the dark. The LCD’s have long life and a wide operating temperature range. Changing the display size or the layout size is relatively simple which makes the LCD’s more customer friendly. The LCDs used exclusively in watches, calculators and measuring instruments are the simple seven-segment displays, having a limited amount of numeric data. The recent advances in technology have resulted in better legibility, more information displaying capability and a wider temperature range. These have resulted in the LCDs being extensively used in telecommunications and entertainment electronics. The LCDs have even started replacing the cathode ray tubes (CRTs) used for the display of text and graphics, and also in small TV applications.

**powersupply:**

The power supply should be of +5V, with maximum allowable transients of 10mv. To achieve a better / suitable contrast for the display, the voltage (VL) at pin 3 should be adjusted properly.

A module should not be inserted or removed from a live circuit. The ground terminal of the power supply must be isolated properly so that no voltage is induced in it. The module should be isolated from the other circuits, so that stray voltages are not induced, which could cause a flickering display.

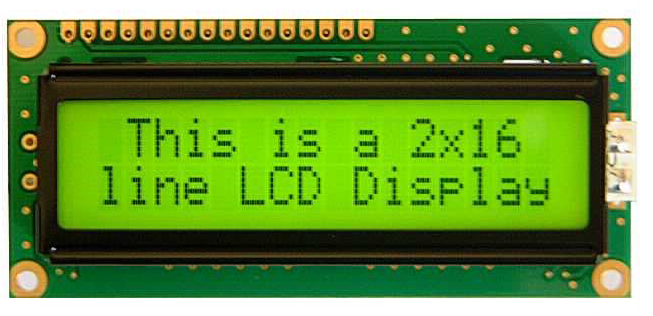


Fig. 3 LCD Display

A **liquid crystal display** (**LCD**) is a thin, flat [electronic visual display](http://en.wikipedia.org/wiki/Electronic_visual_display) that uses the light modulating properties of [liquid crystals](http://en.wikipedia.org/wiki/Liquid_Crystals) (LCs). LCs do not emit light directly. They are used in a wide range of applications including: [computer monitors](http://en.wikipedia.org/wiki/Computer_monitor), [television](http://en.wikipedia.org/wiki/Television), instrument panels, [aircraft cockpit displays](http://en.wikipedia.org/wiki/Flight_instruments), [signage](http://en.wikipedia.org/wiki/Signage), etc. They are common in consumer devices such as video players, gaming devices, [clocks](http://en.wikipedia.org/wiki/Clock), watches, [calculators](http://en.wikipedia.org/wiki/Calculator), and [telephones](http://en.wikipedia.org/wiki/Telephone). LCDs have displaced [cathode ray tube](http://en.wikipedia.org/wiki/Cathode_ray_tube)(CRT) displays in most applications. They are usually more compact, lightweight, portable, less expensive, more reliable, and easier on the eyes. They are available in a wider range of screen sizes than CRT and plasma displays, and since they do not use phosphors, they cannot suffer image burn-in.LCDs are more energy efficient and offer safer disposal than CRTs. Its low electrical power consumption enables it to be used in [battery](http://en.wikipedia.org/wiki/Battery_(electricity))-powered [electronic](http://en.wikipedia.org/wiki/Electronics) equipment. It is an [electronically-modulated optical device](http://en.wikipedia.org/wiki/Electro-optic_modulator) made up of any number of [pixels](http://en.wikipedia.org/wiki/Pixel) filled with [liquid crystals](http://en.wikipedia.org/wiki/Liquid_crystal) and arrayed in front of a [light source](file:///F:\SONA%20IOT%20LAB%202021\LAB%20RECORD\Light) ([backlight](http://en.wikipedia.org/wiki/Backlight)) or [reflector](http://en.wikipedia.org/wiki/Reflector_(photography)) to produce images in colour or [monochrome](http://en.wikipedia.org/wiki/Monochrome). The earliest discovery leading to the development of LCD technology, the discovery of liquid crystals, dates from 1888. By 2008, worldwide sales of televisions with LCD screens had surpassed the sale of CRT units

**ESP8266 WIFI MODULE:**

The ESP8266 is a low-cost wi-fi microchip, with a full TCP/IP stack and microcontroller capability. 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, the chip, and the software on it, as well as to translate the Chinese documentation.

The **ESP8285** is an ESP8266 with 1 MiB of built-in flash, allowing the building of single-chip devices capable of connecting to Wi-Fi. The successor to these microcontroller chips is the ESP32, released in 2016.

FEATURES:

* **Processor**: L106 32-bit RISC microprocessor core based on the Tensilica Xtensa Diamond Standard 106Micro running at 80 Mhz
* **Memory:**
  + 32 KiB instruction RAM
  + 32 KiB instruction cache RAM
  + 80 KiB user-data RAM
  + 16 KiB ETS system-data RAM
* External QSPI flash: up to 16 MiB is supported (512 KiB to 4 MiB typically included)
* IEEE 802.11 b/g/n Wi-Fi
* Integrated TR switch,Balum,LNA,Power amplifier and matching network
* 16 GPIO pins

**RELAY:**

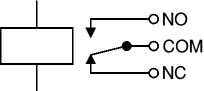
A relay is an electrically operated switch. Current flowing through the coil of the relay creates a magnetic field which attracts a lever and changes the switch contacts. The coil current can be on or off so relays have two switch positions and they are double throw (changeover) switches. Relays allow one circuit to switch a second circuit which can be completely separate from the first. For example a low voltage battery circuit can use a relay to switch a 230V AC mains circuit. There is no electrical connection inside the relay between the two circuits; the link is magnetic and mechanical.

The coil of a relay passes a relatively large current, typically 30mA for a 12V relay, but it can be as much as 100mA for relays designed to operate from lower voltages. Most ICs (chips) cannot provide this current and a transistor is usually used to amplify the small IC current to the larger value required for the relay coil. The maximum output current for the popular 555 timer IC is 200mA so these devices can supply relay coils directly without amplification.



Fig. 4 Relay

Relays are usually SPDT or DPDT but they can have many more sets of switch contacts, for example relays with 4 sets of changeover contacts are readily available. Most relays are designed for PCB mounting but you can solder wires directly to the pins providing you take care to avoid melting the plastic case of the relay. The animated picture shows a working relay with its coil and switch contacts. You can see a lever on the left being attracted by magnetism when the coil is switched on. This lever moves the switch contacts. There is one set of contacts (SPDT) in the foreground and another behind them, making the relay DPDT.



The relay's switch connections are usually labeled COM, NC and NO:

* **COM** = Common, always connect to this, it is the moving part of the switch.
* **NC** = Normally Closed, COM is connected to this when the relay coil is **off**.
* **NO** = Normally Open, COM is connected to this when the relay coil is **on**.

**PIR SENSOR:**

A passive infrared sensor (PIR sensor) is an electronic sensor that measures infrared (IR) light radiating from objects in its field of view. They are most often used in PIR-based [motion detectors](https://en.wikipedia.org/wiki/Motion_detector). PIR sensors are commonly used in security alarms and automatic lighting applications. PIR sensors detect general movement, but do not give information on who or what moved. For that purpose, an active IR sensor is required.

PIR sensors are commonly called simply "PIR", or sometimes "PID", for "passive infrared detector". The term *passive* refers to the fact that PIR devices do not radiate energy for detection purposes. They work entirely by detecting infrared radiation (radiant heat) emitted by or reflected from objects.



Fig. 5 PIR Sensor

**Operating principle and operation:**

All objects with a temperature above absolute zero emit heat energy in the form of radiation. Usually this radiation isn't visible to the human eye because it radiates at infrared wavelengths, but it can be detected by electronic devices designed for such a purpose.

A PIR-based [motion detector](https://en.wikipedia.org/wiki/Motion_detector) is used to sense movement of people, animals, or other objects. They are commonly used in bu[rglar alarms](https://en.wikipedia.org/wiki/Burglar_alarm) and automatically-activated [lighting](https://en.wikipedia.org/wiki/Lighting) systems. A PIR sensor can detect changes in the amount of infrared radiation impinging upon it, which varies depending on the temperature and surface characteristics of the objects in front of the sensor. When an object, such as a person, passes in front of the background, such as a wall, the temperature at that point in the sensor's field of view will rise from [room temperature](https://en.wikipedia.org/wiki/Room_temperature) to [body temperature](https://en.wikipedia.org/wiki/Body_temperature), and then back again. The sensor converts the resulting change in the incoming infrared radiation into a change in the output voltage, and this triggers the detection. Objects of similar temperature but different surface characteristics may also have a different infrared emission pattern, and thus moving them with respect to the background may trigger the detector as well.

PIRs come in many configurations for a wide variety of applications. The most common models have numerous [Fresnel lenses](https://en.wikipedia.org/wiki/Fresnel_lens) or mirror segments, an effective range of about 10 meters (30 feet), and a field of view less than 180. Models with wider fields of view, including 360°, are available, typically designed to mount on a ceiling. Some larger PIRs are made with single segment mirrors and can sense changes in infrared energy over 30 meters (100 feet) from the PIR. There are also PIRs designed with reversible orientation mirrors which allow either broad coverage (110° wide) or very narrow "curtain" coverage, or with individually selectable segments to "shape" the coverage.

**3.2 SOFTWARE REQUIREMENTS**

* Arduino IDE
* Ardroid Studio

**3.2.1 SOFTWARE DESCRIPTION**

**Arduino IDE:**

The [**Arduino**](https://en.wikipedia.org/wiki/Arduino) **Integrated Development Environment (**[**IDE**](https://en.wikipedia.org/wiki/Integrated_development_environment)**)** is a [cross-platform](https://en.wikipedia.org/wiki/Cross-platform) application (for [Windows](https://en.wikipedia.org/wiki/Windows), [macOS](https://en.wikipedia.org/wiki/MacOS), [Linux](https://en.wikipedia.org/wiki/Linux)) that is written in functions from [C](https://en.wikipedia.org/wiki/C_(programming_language)) and C++. It is used to write and upload programs to Arduino compatible boards, but also, with the help of 3rd party cores, other vendor development boards.

The source code for the IDE is released under the [GNU General Public License](https://en.wikipedia.org/wiki/GNU_General_Public_License), version 2.The Arduino IDE supports the languages [C](https://en.wikipedia.org/wiki/C_(programming_language)) and [C++](https://en.wikipedia.org/wiki/C%2B%2B) using special rules of code structuring. The Arduino IDE supplies a [software library](https://en.wikipedia.org/wiki/Software_library) from the [Wiring](https://en.wikipedia.org/wiki/Wiring_(development_platform)) project, which provides many common input and output procedures. User-written code only requires two basic functions, for starting the sketch and the main program loop, that are compiled and linked with a program stub *main()* into an executable [cyclic executive](https://en.wikipedia.org/wiki/Cyclic_executive) program with the [GNU toolchain](https://en.wikipedia.org/wiki/GNU_toolchain), also included with the IDE distribution.The Arduino IDE employs the program *avrdude* to convert the executable code into a text file in hexadecimal encoding that is loaded into the Arduino board by a loader program in the board's firmware. By default, avrdude is used as the uploading tool to flash the user code onto official Arduino boards.

**Android Studio:**

Android Studio is the official integrated developmentenvironment(IDE) for Google’s Android operating system, built on JetBrains'IntelliJ IDEA software and designed specifically for Android Development. It is available for download on Windows, mac and Linux based operating systems. It is a replacement for the Eclipse Android Development Tools(ADT) as the primary IDE for native Android application development.

Android Studio was announced on May 16, 2013 at the [Google I/O](https://en.wikipedia.org/wiki/Google_I/O) conference. It was in early access preview stage starting from version 0.1 in May 2013, then entered beta stage starting from version 0.8 which was released in June 2014. The first stable build was released in December 2014, starting from version 1.0. On May 7, 2019, [Kotlin](https://en.wikipedia.org/wiki/Kotlin_(programming_language)) replaced [Java](https://en.wikipedia.org/wiki/Java_(programming_language)) as Google’s preferred language for Android app development. Java is still supported, as is [C++](https://en.wikipedia.org/wiki/C%2B%2B).

**Features:**

* Android-specific [refactoring](https://en.wikipedia.org/wiki/Code_refactoring) and quick fixes
* Lint tools to catch performance, usability, version compatibility and other problems
* ProGuard integration and app-signing capabilities
* Template-based wizards to create common Android designs and components
* A rich [layout editor](https://en.wikipedia.org/wiki/Graphical_user_interface_builder) that allows users to drag-and-drop UI components, option to [preview layouts](https://en.wikipedia.org/wiki/WYSIWYG) on multiple screen configurations
* Support for building [Android Wear](https://en.wikipedia.org/wiki/Android_Wear) apps
* Built-in support for Google Cloud Platform, enabling integration with Firebase Cloud Messaging (Earlier 'Google Cloud Messaging') and Google App Engine
* Android Virtual Device (Emulator) to run and debug apps in the Android studio.

**4. SYSTEM DESIGN**

**4.1 Architecture:**

The proposed system is implemented with microcontroller board (Arduino uno) and Mobile application developed on android studio, server. The server is used to communicate with mobile application and microcontroller. On microcontroller side local decision are made based on mode selected by user from mobile application. Other major decision based on external environment is taken by server and then conveyed to microcontroller.

Description: Description: A screenshot of a cell phone

Description automatically generated

Fig. 6 Block diagram of Building Automation

On microcontroller side local decision are made based on mode selected by user from mobile application. Other major decision based on external environment is taken by server and then conveyed to microcontroller. External environment is accessed through IoT kit and Data visualization is done. After visualizing the data, the electrical appliances can be controlled using the mobile application(i.e it can be switched on when needed and switched off when not in use remotely using the android application).

Description: Description: A close up of a map

Description automatically generated

Fig. 7 Architecture of the system

**5. SYSTEM IMPLEMENTATION**

**5.1. Module Description**

**5.1.1 Sensors and Microcontrollers**

Light sensitive sensor works on the principle of photoconductivity [11]. A light dependent resistor works on the principle of photoconductivity. Photoconductivity is an optical phenomenon in which the materials conductivity is increased when light is absorbed by the material. When light falls i.e. when the photons fall on the device, the electrons in the valence band of the semiconductor material are excited to the conduction band. As more and more current starts flowing through the device when the circuit is closed and it is said that the resistance of the device has been decreased. A temperature sensor measures the hotness or coolness of an object. The sensors working base is the voltage that’s read across the diode. The temperature rises whenever the voltage increases. The sensor records any voltage drop between the transistor base and emitter. A popular thermal measuring method is thermocouple, which is composed of two different metal alloy wires.

Now and again alluded to as an embedded controller or microcontroller unit (MCU), microcontrollers are compact incorporated circuits intended to oversee a particular operation in an installed framework. Microcontroller incorporates a processor, memory and input /output(I/O) peripherals on a solitary chip. Microcontrollers are utilized as a part of numerous ventures and applications, incorpo- rating into the home and undertaking, building mechani- sation, producing, mechanical autonomy, car, lighting, brilliant vitality, modern computerization, interchanges and web of things (IoT) arrangements.

Devices like fans and lights are associated with the sensors and microcontroller. These sensors are in charge of acquiring the information from the encompassing and the microcontroller will control the working of these gadgets as per the information that is processed.



Fig. 8 Hardware Setup

**5.1.2 Web server and database**

A web server is a framework that conveys substance or administrations to end clients over the web. A web server comprises of a physical server, server working framework (OS) and software used to facilitate HTTP correspondence. A database server is a PC framework that furnishes different PCs with administrations identified with getting to and recovering information from a database. A server is a machine where the application code dwells and database is a machine which goes about as an information storehouse for the application. Database additionally dwells on a server and we call it as database server. Values acquired from the sensors would be stored in the database.

The images include interactive capabilities, enabling users to manipulate them or drill into the data for querying and analysis. Information perception is a general term that depicts any effort to enable individuals to comprehend the importance of information by setting it in a visual setting. Examples, patterns and connections that may go undetected in text-based information can be uncovered and perceived simpler with information representation programming. Nowadays it is shown in the form of infographics, dials and gauges, geographic maps, sparklines, warm maps, and point by point bar, pie and fever graphs.

**5.1.3 Mobile application**

Portable applications are composed with thought for the requests and requirements of the gadgets and furthermore to exploit any specific capacities they have. A gaming application, for instance, may exploit the iPhone’s sensor. Similarly, the mobile application in this architecture would be responsible for giving control conditions for the functioning of the devices.

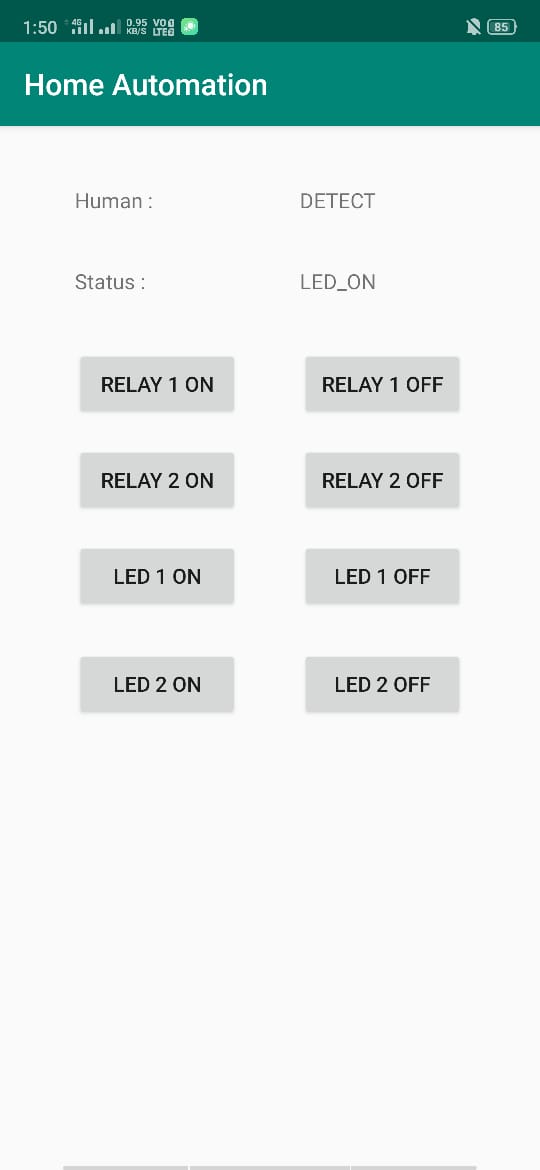


Fig. 9 Application UI

**6. CONCLUSION**

The important aim of this paper is that to reduce the power consumption and to save the energy without any waste. In this prototype we have created both automated on/off system and manual controlling system using the android mobile application. This prototype can be used in places like classrooms, labs, seminor halls in colleges. This system needs initial cost for the installation but the maintenance is very simple, thus such system is very helpful in reducing the conventional power. This system can be implemented in large scale to bring significant reduction in electrical energy and to reduce the human work. IR sensor (presence sensor) which has the task of identifying the passage of any objects causing the light glows dim to maximum intensity of led light. This feature permits to activate lamps solely when necessary, avoiding wastage of energy. This work aims to reduce the drawbacks of the current power saving, and find a solution to save power. This is done by replacing sodium vapor lamps by LED and adding dimming technology to it. It also provides an effective measure to save energy by preventing unnecessary wastage of electricity, caused due to manual switching of electrical appliances when it is not required.