**SMART AUDITORIUM SYSTEM**

**A Project Report *on***

*IOT with Rasberry PI and data*

*visualization on Web*

***Submitted by***

**SOUVIK GANGULY**

**DEBASISH BISWAS**

**SUBHAJIT PRAMANIK**

**TRIRANJAN SAMANTA**

**SOUMYAJYOTI SAHA**

**ABHISEK MANNA**

**PARTHA SARATHI SARKAR**

**PRIYANGSHU NATH**

**SHREYA DAS**

**ESHITA ROY**

**UTSA PURKAYASTHA**

***of***

**Bachelor of Technology (B.Tech)**

***In***

**ELECTRONICS & COMMUNICATION ENGINEERING**

***From***

**NETAJI SUBHASH ENGINEERING COLLEGE**

**Contents-**

|  |  |  |
| --- | --- | --- |
| **SL NO.** | **PARTICULARS** | **PAGE NO.** |
| 1 | Acknowledgement  & Introduction | 3 |
| 2 | Components Used | 4 |
| 3 | Description | 4-33 |
| 4 | Steps | 34 |
| 5 | Block Diagram | 35 |
| 6 | Circuit Diagram | 35 |
| 7 | Code | 36-38 |
| 8 | Result | 39 |
| 9 | Costing | 40 |
| 10 | Purpose of our project | 40 |
| 11 | Scope | 40 |
| 12 | Conclusion | 41 |
| 13 | Reference | 41 |

**ACKNOWLEDGEMENT**

The achievement that is associated with the successful completion of any task would be incomplete without mentioning the names of those people whose endless cooperation made it possible. Their constant guidance and encouragement made all our efforts successful. We take this opportunity to express our deep gratitude towards our project mentor, **SHOUVIK SARKAR** for giving such valuable suggestions, guidance and encouragement during the development of this project work. Last but not the least we are grateful to all the faculty members of Academy of Skill Development. for their support.

**INTRODUCTION**

Automated lighting can make you more energy efficient, with lights and lighting levels only being used as necessary. By only using energy that is needed, users can reduce their electrical bills. Lighting can be combined with other automated systems for convenience, for example, with a home security system.

Automatic lighting systems can be very useful in detecting human presence and turning on/off the electrical appliances without any manual controller who has to be present 24x7 that can help with manual labor and thus is a very good example of technology helping with day to day human work.

To indicate if the room/hall/auditorium that is concerned is capable of allowing any more entries or is it full and that can be accessed worldwide through third party cloud.

**Component List-**

**Hardware :** Raspberry Pi Pico(W)

Infrared sensor x 2

LEDs

Connecting Wires

Breadboard

**Software :** Thonny (Python IDE)

OS Windows

Wokwi

Things speak cloud

**HARDWARE**

**Raspberry Pi Pico W:**

Raspberry pi pico is **Raspberry Pi'**s first wireless microcontroller board, designed especially for physical computing. It is the successor of the popular**Raspberry Pi Pico** board. Similar to the [Picoboard](https://components101.com/development-boards/raspberry-pi-pico-pinout-datasheet-specifications), which we discussed earlier, the **Pico W** board is also built around the [**Raspberry Foundation**](https://components101.com/tags/raspberry-pi) in-house ARM chip RP2040. The main improvement is the addition of Wi-Fi and Bluetooth functionality. **Raspberry Pi Pico W** incorporates an Infineon **CYW43439** wireless chip that supports IEEE 802.11 b/g/n wireless LAN, and Bluetooth 5.2.

**Raspberry Pi Pico W Pinout:**

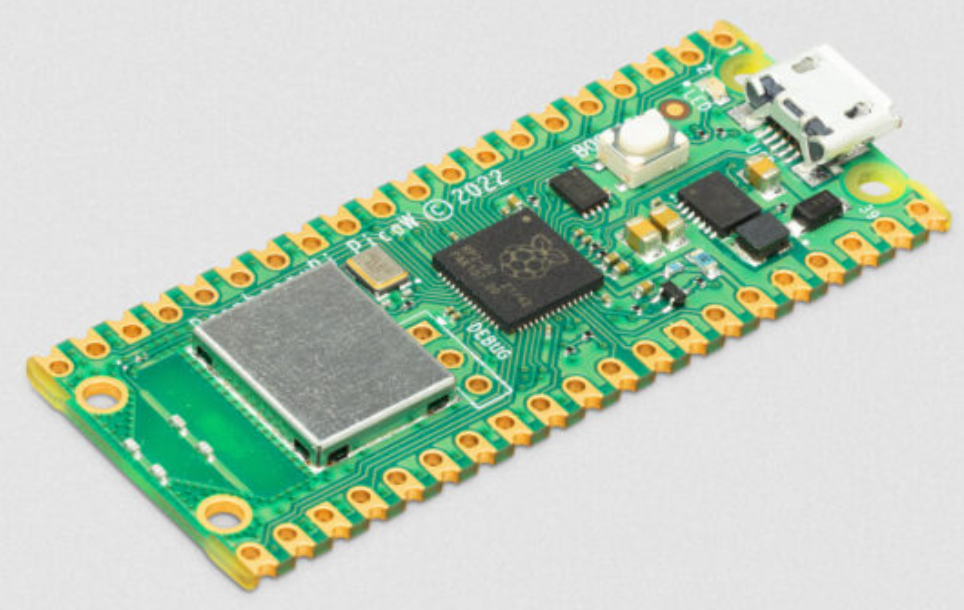
Raspberry Pi Pico W is pin-to-pin compatible with the older Pico board and has a total of 40 Pins out of which 26 are multipurpose GPIOs. These 26 digital input and output pins are labeled as GP0, GP1, and so on up to GP28. Out of these 30 pins, GP23, GP24, GP25, and GP29 are not exposed on the header, hence we only get a total of 26 GPIO pins. All GPIOs are operating at a 3.3V logic level.  Apart from these pins, Pico w also offers a 3-Pin header for debug interface which is located near the RP2040 Chip.

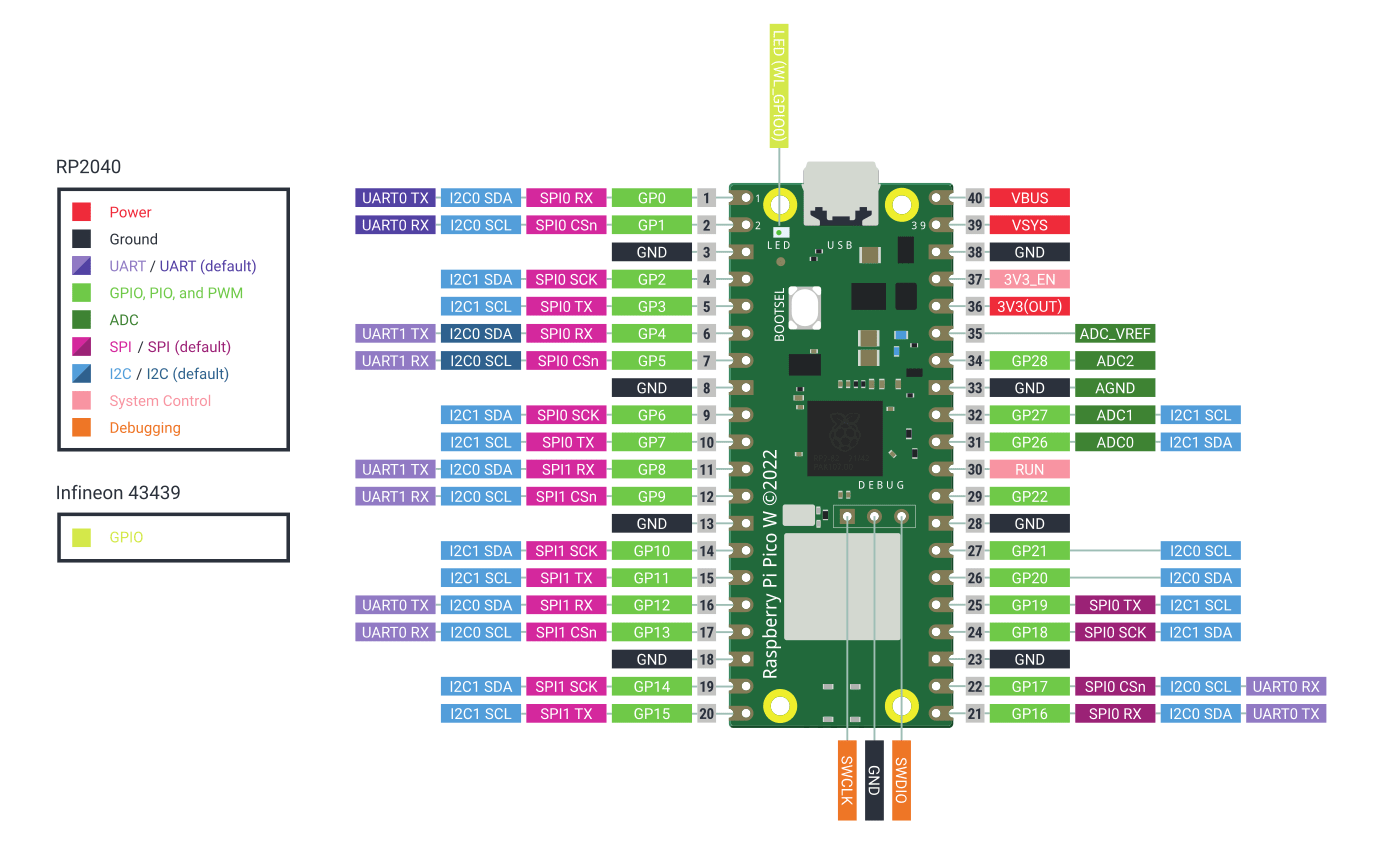
**Raspberry Pi Pico W Specifications:**

The key features of Raspberry Pi Pico W are the following:

* **RP2040**microcontroller chip designed by Raspberry Pi in
* **Dual-core ARM Cortex M0+** processor, flexible clock running up to 133 MHz
* 264kB of **SRAM**, and 2MB of onboard Flash memory
* On-board single-band 2.4GHz wireless interfaces (802.11n)
* Castellated module allows soldering directly to carrier boards
* USB 1.1 Host and Device support
* Low-power sleep and dormant modes
* Drag & drop programming using mass storage over USB
* 26 multi-function GPIO pins
* 2×SPI, 2×I2C, 2×UART, 3×12-bit ADC, 16×controllable PWM channels
* Accurate clock and timer on-chip
* Temperature sensor
* Accelerated floating point libraries on-chip
* 8×Programmable IO (PIO) state machines for custom peripheral support

**Raspberry Pi Pico W:**

****

**Raspberry Pi Pico W Pinout:**

Raspberry Pi Pico W uses an on-board buck-boost SMPS which is able to generate the required 3.3V (to power RP2040 and external circuitry) from a wide range of input voltages (~1.8 to 5.5V). This allows significant flexibility in powering the unit from various sources, such as a single lithium-ion cell, or three AA cells in series. Battery chargers can also be very easily integrated with the Pico W powerchain. Reprogramming the Pico W flash can be done using USB (simply drag and drop a file onto the Pico W, which appears as a mass storage device), or the standard serial wire debug (SWD) port can reset the system and load and run code without any button presses. The SWD port can also be used to interactively debug code running on the RP2040.

The Pico W is a single sided 51mm × 21mm × 1mm PCB with a micro USB port overhanging the top edge, and dual castellated/through-hole pins around the two long edges. The onboard wireless antenna is located on the bottom edge. To avoid detuning the antenna, no material should intrude into this space. Pico W is designed to be usable as a surfacemount module as well as presenting a dual inline package (DIP) format, with the 40 main user pins on a 2.54mm (0.1") pitch grid with 1mm holes, compatible with veroboard and breadboard. Pico W also has four 2.1mm (± 0.05mm) drilled mounting holes to provide for mechanical fixing

A few RP2040 GPIO pins are used for internal board functions:

GPIO29 OP/IP wireless SPI CLK/ADC mode (ADC3) to measure VSYS/3

GPIO25 OP wireless SPI CS - when high also enablesmGPIO29 ADC pin to read VSYS

GPIO24 OP/IP wireless SPI data/IRQ

GPIO23 OP wireless power on signal

WL\_GPIO2 IP VBUS sense - high if VBUS is present, else low

WL\_GPIO1 OP controls the on-board SMPS power save pin (Section 3.4)

WL\_GPIO0 OP connected to user LED

Apart from GPIO and ground pins, there are seven other pins on the main 40-pin interface:

PIN40 VBUS

PIN39 VSYS

PIN37 3V3\_EN

PIN36 3V3

PIN35 ADC\_VREF

PIN33 AGND

PIN30 RUN

VBUS is the micro-USB input voltage, connected to micro-USB port pin 1. This is nominally 5V (or 0V if the USB is not connected or not powered).

VSYS is the main system input voltage, which can vary in the allowed range 1.8V to 5.5V, and is used by the on-board SMPS to generate the 3.3V for the RP2040 and its GPIO.

3V3\_EN connects to the on-board SMPS enable pin, and is pulled high (to VSYS) via a 100kΩ resistor. To disable the 3.3V (which also de-powers the RP2040), short this pin low.

3V3 is the main 3.3V supply to RP2040 and its I/O, generated by the on-board SMPS. This pin can be used to power external circuitry (maximum output current will depend on RP2040 load and VSYS voltage; it is recommended to keep the load on this pin under 300mA).

ADC\_VREF is the ADC power supply (and reference) voltage, and is generated on Pico W by filtering the 3.3V supply. This pin can be used with an external reference if better ADC performance is required.

AGND is the ground reference for GPIO26-29. There is a separate analogue ground plane running under these signals and terminating at this pin. If the ADC is not used or ADC performance is not critical, this pin can be connected to digital ground.

RUN is the RP2040 enable pin, and has an internal (on-chip) pull-up resistor to 3.3V of about ~50kΩ. To reset RP2040, short this pin low.Finally, there are also six test points (TP1-TP6), which can be accessed if required, for example if using as a surfacemount module. These are:

TP1 Ground (close-coupled ground for differential USB signals)

TP2 USB DM

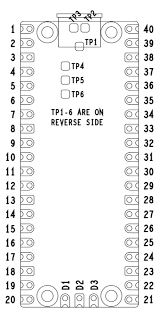
TP3 USB DP

TP4 WL\_GPIO1/SMPS PS pin (do not use)

TP5 WL\_GPIO0/LED (not recommended to be used)

TP6 BOOTSEL

TP1, TP2 and TP3 can be used to access USB signals instead of using the micro-USB port. TP6 can be used to drive the system into mass-storage USB programming mode (by shorting it low at power-up). Note that TP4 is not intended to be used externally, and TP5 is not really recommended to be used as it will only swing from 0V to the LED forward voltage (and hence can only really be used as an output with special care).



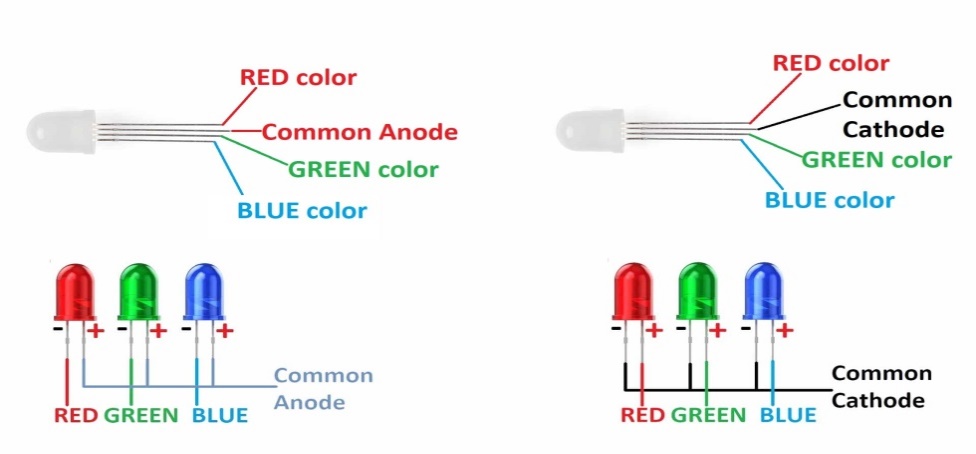
## RGB LED:

An RGB LED is basically an LED package that can produce almost any color. It can be used in different applications such as outdoor decoration lighting, stage lighting designs, home decoration lighting, LED matrix display, and more.

RGB LEDs have three internal LEDs (Red, Green, and Blue) that can be combined to produce almost any color output. In order to produce different kinds of colors, we need to set the intensity of each internal LED and combine the three color outputs. In this tutorial, we are going to use PWM to adjust the intensity of the red, green, and blue LEDs individually and the trick here is that our eyes will see the combination of the colors, instead of the individual colors because the LEDs are very close to each other inside.

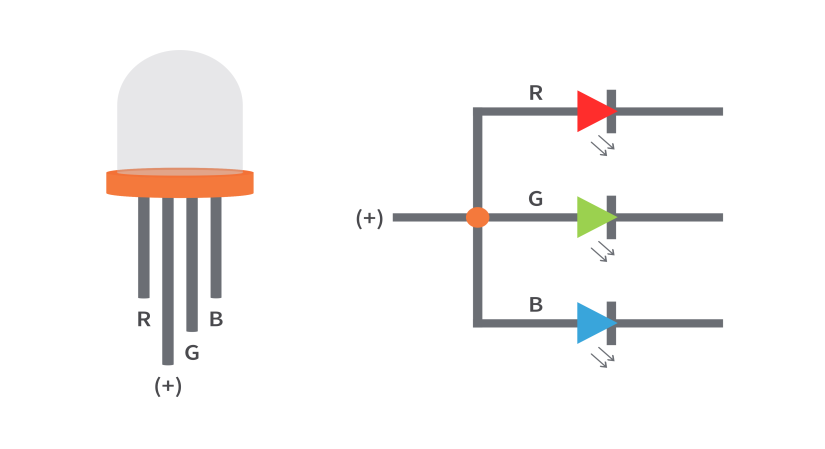
**RGB LED Types and Structure**

As mentioned earlier, RGB LEDs have three LEDs inside them and usually, these three internal LEDs share either a common anode or a common cathode especially in a through holepackage.So basically, we can categorize RGB LEDs as either common anode and common.



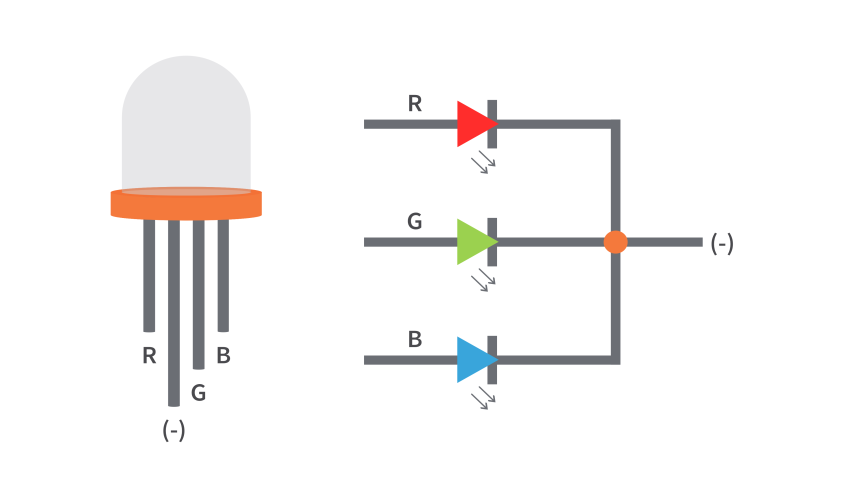
## Common Anode RGB LED

In a common anode RGB LED, the anode of the internal LEDs are all connected to the external anode lead. To control each color, you need to apply a LOW signal or ground to the red, green, and blue leads and connect the anode lead to the positive terminal of the power supply.



## Common Cathode RGB LED

In a common cathode RGB LED, the cathode of the internal LEDs are all connected to the external cathode lead. To control each color, you need to apply a HIGH signal or VCC to the red, green, and blue leads and connect the anode lead to the negative terminal of the power supply.



### Working:

The **working of RGB LED** is, it has four terminals, in which three colors red, green, blue, and one more terminal represent anode or cathode depending on its type. The emission of various colors using this LED can be achieved by changing or setting the intensity levels of internal LED’s ( red LED, the green LED, blue LED) and combining these colored outputs to display different colored outputs. Since we know that the formation of this LED can be done with three basic separate LEDs for red, green, and blue in one package.

A Constant Current Reduction method (CCR) or pulse width [modulation](https://www.watelectronics.com/what-is-analog-modulation-types-its-applications/) (PWM) is used to set the intensity levels of colored LEDs (red, green, blue) separately. Here we can observe the combination of different colored outputs when LED is blinking because the internal LEDs and terminals are very close to each other. These types of LEDs are mostly implemented in computer hardware, motherboard, RAM, etc.

## INFRARED 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. These types of sensors measure only infrared radiation, rather than emitting it that is called a [passive IR sensor](https://www.elprocus.com/passive-infrared-pir-sensor-with-applications/). Usually, in the infrared spectrum, all the objects radiate some form of thermal radiation.

Infrared Sensor

These types of radiations are invisible to our eyes, which can be detected by an infrared sensor. The emitter is simply an IR LED ([Light Emitting Diode](https://www.elprocus.com/explain-different-types-leds-working-applications-engineering-students/)) and the detector is simply an IR photodiode that is sensitive to IR light of the same wavelength as that emitted by the IR LED. When IR light falls on the photodiode, the resistances and the output voltages will change in proportion to the magnitude of the IR light received.

### Working Principle-

The working principle of an infrared sensor is similar to the object detection sensor. This sensor includes an IR LED & an IR Photodiode, so by combining these two can be formed as a photo-coupler otherwise optocoupler. The physics laws used in this sensor are planks radiation, Stephan Boltzmann & weins displacement.

IR LED is one kind of transmitter that emits IR radiations. This LED looks similar to a standard LED and the radiation which is generated by this is not visible to the human eye. Infrared receivers mainly detect the radiation using an infrared transmitter. These infrared receivers are available in photodiodes form. IR Photodiodes are dissimilar as compared with usual photodiodes because they detect simply IR radiation. Different kinds of infrared receivers mainly exist depending on the voltage, wavelength, package, etc.

Once it is used as the combination of an IR transmitter & receiver, then the receiver’s wavelength must equal the transmitter. Here, the transmitter is IR LED whereas the receiver is IR photodiode. The infrared photodiode is responsive to the infrared light that is generated through an infrared LED. The resistance of photo-diode & the change in output voltage is in proportion to the infrared light obtained. This is the IR sensor’s fundamental working principle.

Once the infrared transmitter generates emission, then it arrives at the object & some of the emission will reflect back toward the infrared receiver. The sensor output can be decided by the IR receiver depending on the intensity of the response.

### Types of Infrared Sensor

Infrared sensors are classified into two types like active IR sensor and passive IR sensor.

#### **Active IR Sensor**

This active infrared sensor includes both the transmitter as well as the receiver. In most of the applications, the light-emitting diode is used as a source. LED is used as a non-imaging infrared sensor whereas the laser diode is used as an imaging infrared sensor.

These sensors work through energy radiation, received & detected through radiation. Further, it can be processed by using the signal processor to fetch the necessary information. The best examples of this active infrared sensor are reflectance and break beam sensor.

#### **Passive IR Sensor**

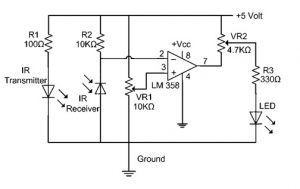
The passive infrared sensor includes detectors only but they don’t include a transmitter. These sensors use an object like a transmitter or IR source. This object emits energy and detects through infrared receivers. After that, a signal processor is used to understand the signal to obtain the required information.

The best examples of this sensor are pyroelectric detector, bolometer, thermocouple-thermopile, etc. These sensors are classified into two types like thermal IR sensor and quantum IR sensor. The thermal IR sensor doesn’t depend on wavelength. The energy source used by these sensors is heated. Thermal detectors are slow with their response and detection time. The quantum IR sensor depends on the wavelength and these sensors include high response and detection time. These sensors need regular cooling for specific measurements.

### IR Sensor Circuit Diagram-

An infrared sensor circuit is one of the basic and popular sensor modules in an [electronic device](https://www.elprocus.com/basic-components-used-electronics-electrical/). This sensor is analogous to human’s visionary senses, which can be used to detect obstacles and it is one of the common applications in real-time. This circuit comprises the following components

* [LM358 IC](https://www.elprocus.com/op-amp-ics-pin-configuration-features-working/) 2 IR transmitter and receiver pair
* Resistors of the range of kilo-ohms.
* Variable resistors.
* LED (Light Emitting Diode).



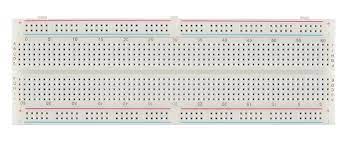
Infrared Sensor Circuit Diagram-

In this project, the transmitter section includes an IR sensor, which transmits continuous IR rays to be received by an IR receiver module. An IR output terminal of the receiver varies depending upon its receiving of IR rays. Since this variation cannot be analyzed as such, therefore this output can be fed to a comparator circuit. Here an [operational amplifier](https://www.elprocus.com/op-amp-ics-pin-configuration-features-working/) (op-amp) of LM 339 is used as a comparator circuit.

When the IR receiver does not receive a signal, the potential at the inverting input goes higher than that non-inverting input of the comparator IC (LM339). Thus the output of the comparator goes low, but the LED does not glow. When the IR receiver module receives a signal to the potential at the inverting input goes low. Thus the output of the comparator (LM 339) goes high and the LED starts glowing.

**Breadboard-**

Breadboards are temporary work boards for [electronic circuits](https://www.sciencedirect.com/topics/engineering/networks-circuits). The general shape of a breadboard is shown in Fig. 6.3. Compatible with most breadboards, 24-gauge wire is used to connect circuits; solid wire, not stranded. Sometimes, kits may be available with various colors of fixed lengths to specifically fit breadboards. These are a nice convenience.



The horizontal rows are connected throughout the row and may make a complete row with the addition of a simple jumper at the center point. These rows are noted with red and blue or black markings.

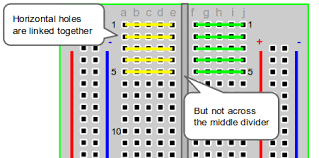
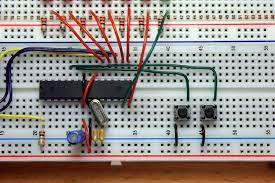


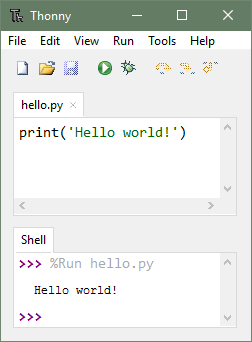
Fig. 6.4. Breadboard horizontal row and vertical column connections.

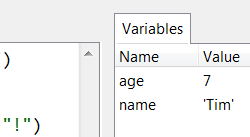
Similarly, the vertical columns are connected vertically down half the width of the board. A jumper may be inserted at the middle location to connect the full width of the breadboard. The horizontal rows are typically used for power and ground, and, with the middle NOT connected, one can apply more than one power level, such as 12 V, 3.3 V, and ground shows the orientation of the board for our purposes. As noted vertical columns are for connections to each leg of your integrated circuit (IC). Like the three integrated circuits shown in, your IC should straddle the center valley.



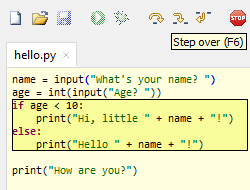
**SOFTWARES-**

**THONNY IDE:**

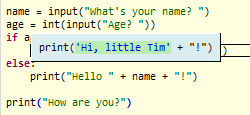
An IDE will contain all those features in just one software.The Thonny IDE will then allow you to be more productive when you write Python programs.Also, when you begin with programming, using an IDE is great because you don’t have to think about all the configuration, you can just start and learn step by step directly by programming.Thonny comes with Python 3.10 built in, so just one simple installer is needed and you're ready to learn programming. The initial user interface is stripped of all features that may distract beginners.**No-hassle variables**. Once you're done with hello-worlds, select View → Variables and see how your programs and shell commands affect Python variables.



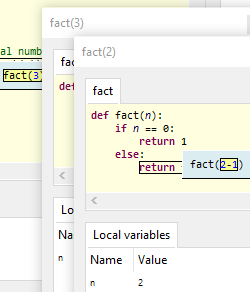
**Simple debugger.** Just press Ctrl+F5 instead of F5 and you can run your programs step-by-step, no breakpoints needed. Press F6 for a big step and F7 for a small step. Steps follow program structure, not just code lines.



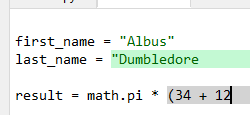
**Step through expression evaluation.** If you use small steps, then you can even see how Python evaluates your expressions. You can think of this light-blue box as a piece of paper where Python replaces subexpressions with their values, piece-by-piece.



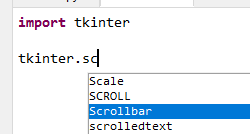
**Faithful representation of function calls.** Stepping into a function call opens a new window with separate local variables table and code pointer. Good understanding of how function calls work is especially important for understanding recursion.



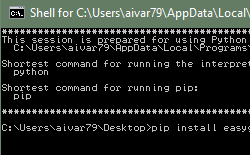
**Highlights syntax errors.** Unclosed quotes and parentheses are the most common beginners' syntax errors. Thonny's editor makes these easy to spot.



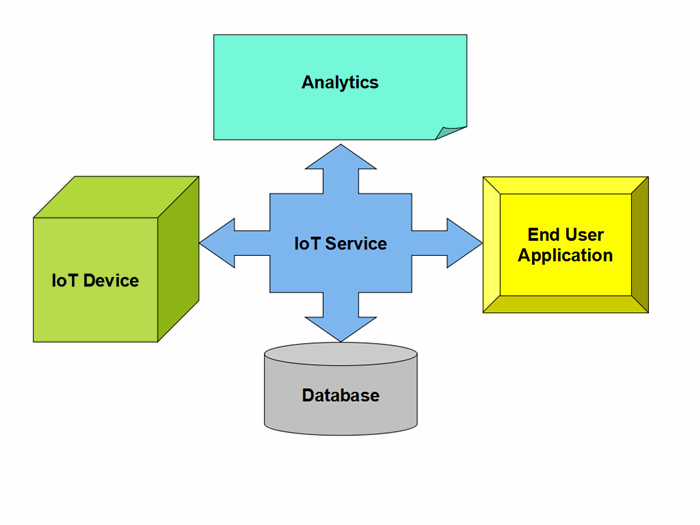
**Code completion.** Students can explore APIs with the help of code completion.



**Beginner friendly system shell**. Select Tools → Open system shell to install extra packages or learn handling Python on command line. PATH and conflicts with other Python interpreters are taken care of by Thonny.



**Things Speak**  
The Internet of Things(IoT) is a system of ‘connected things’. The things generally comprise of an embedded operating system and an ability to communicate with the internet or with the neighboring things. One of the key elements of a generic IoT system that bridges the various ‘things’ is an IoT service. An interesting implication from the ‘things’ comprising the IoT systems is that the things by themselves cannot do anything. At a bare minimum, they should have an ability to connect to other ‘things’. But the real power of IoT is harnessed when the things connect to a ‘service’ either directly or via other ‘things’. In such systems, the service plays the role of an invisible manager by providing capabilities ranging from simple data collection and monitoring to complex data analytics. The below diagram illustrates where an IoT service fits in an IoT ecosystem:



One such IoT application platform that offers a wide variety of analysis, monitoring and counter-action capabilities is ‘ThingSpeak’.ThingSpeak is a platform providing various services exclusively targeted for building IoT applications. It offers the capabilities of real-time data collection, visualizing the collected data in the form of charts, ability to create plugins and apps for collaborating with web services, social network and other APIs. We will consider each of these features in detail below.

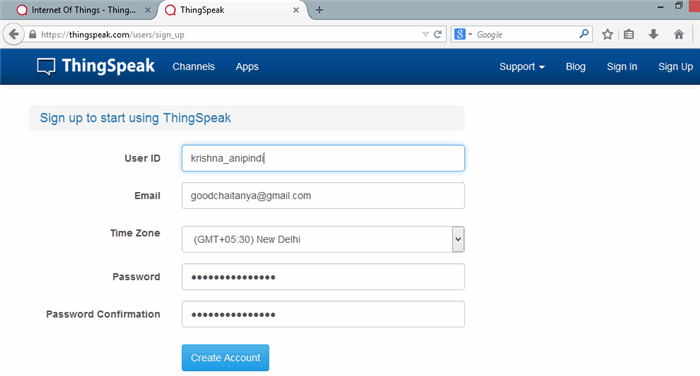
The core element of ThingSpeak is a ‘ThingSpeak Channel’. A channel stores the data that we send to ThingSpeak and comprises of the below elements:

* 8 fields for storing data of any type - These can be used to store the data from a sensor or from an embedded device.
* 3 location fields - Can be used to store the latitude, longitude and the elevation. These are very useful for tracking a moving device.
* 1 status field - A short message to describe the data stored in the channel.

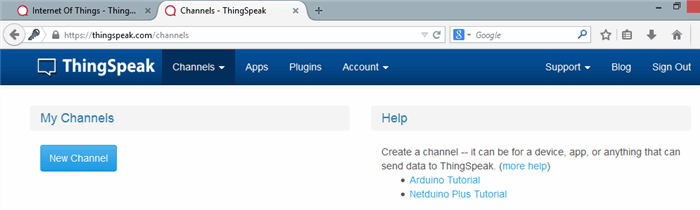
To use ThingSpeak, we need to signup and create a channel. Once we have a channel, we can send the data, allow ThingSpeak to process it and also retrieve the same. Let us start exploring ThingSpeak by signing up and setting up a channel.

***Working Process-***

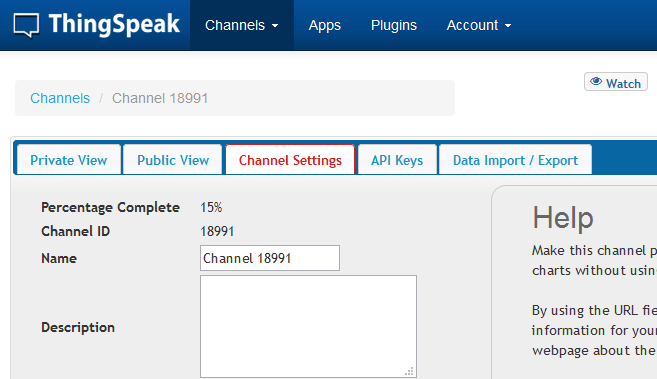
Open <https://thingspeak.com/> and click on the ‘Get Started Now’ button on the center of the page and you will be redirected to the sign-up page(you will reach the same page when you click the ‘Sign Up’ button on the extreme right). Fill out the required details and click on the ‘Create Account’ button.



Now you should see a page with a confirmation that the account was successfully created. The confirmation message disappears after a few seconds and the final page should look as in the below screen:

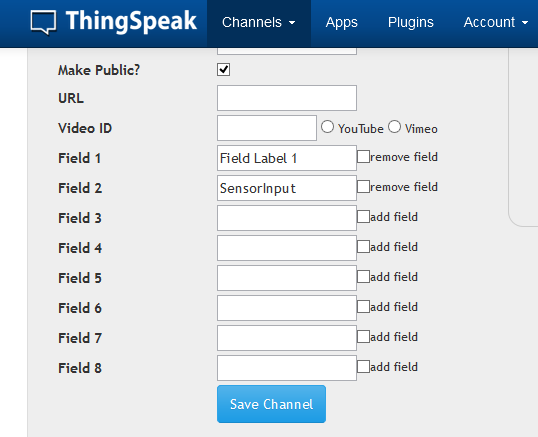


Go ahead and click on ‘New Channel’. You should see a page like the below:

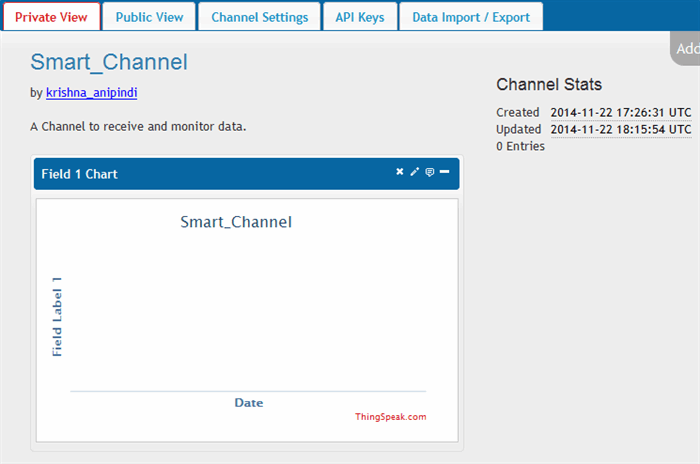


You can change the name to fit your need and you can add a description corresponding to the channel. You can add any other useful description into the metadata field. In the same page, you should see the fields for Latitude, Longitude and Elevation. Also, when you scroll down you should see a check box that says ‘Make Public?’. Let us consider the significance of the various fields and the tabs:

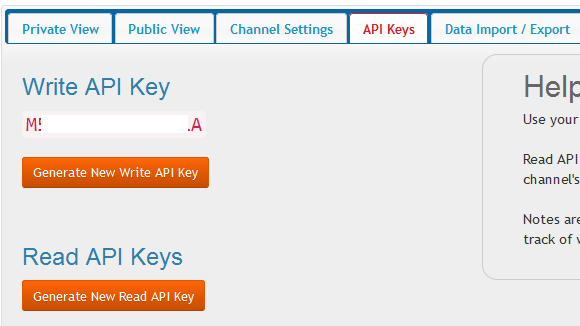
* Latitude, longitude and elevation - These fields correspond to the location of a ‘thing’ and are especially significant for moving things.
* Make Public? - If the channel is made public, anyone can view the channel's data feed and the corresponding charts. If this check box is not checked, the channel is private, which means for every read or write operation, the user has to pass a corresponding API key.
* URL - This can be the URL of your blog or website and if specified, will appear on the public view of the channel.
* Video ID - This is the ID corresponding to your YouTube or Vimeo ID. If specified, the video appears on the public view of the channel.
* Fields 1 to 8 - These are the fields which correspond to the data sent by a sensor or a ‘thing’. A field has to be added before it can be used to store data. By default, Field 1 is added. In case you try posting to fields that you have not added, your request will still be successful, but you will not be able to see the field in the charts and the corresponding data. You can click on the small box before the ‘add field’ text corresponding to each field to add it. Once you click the ‘add field’ box, a default label name appears in the text box corresponding to each field and the ‘add field’ text changes to ‘remove field’. You can edit the field text that appears by default when a field is added to make more sense. For example, in the below screen, I have modified the text for Field 2 to ‘SensorInput’. To remove a field which is added, just check on the ‘remove field’ box. Once you click this, the text ‘remove field’ changes back to ‘add field’ and the corresponding field text is cleared.



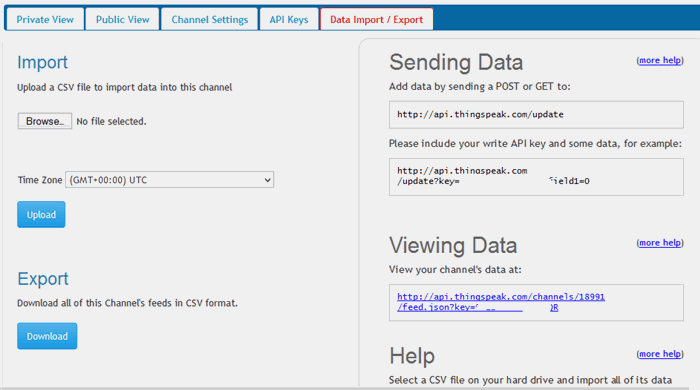
Once you have edited the fields, click on ‘Save Channel’ button. You should now see a page like the below in which the ‘Private View’ tab is defaulted:



The Private View shows a chart corresponding to each of the fields that we have added. Now click on the ‘Public View’ tab. This should look exactly similar to the what we see in the ‘Private View’ tab since our channel is public. In case your channel is not public('make public' check box not checked in the ‘channel settings’ tab), the public view tab shows a message that ‘This channel is not public’. Now click on the ‘API Keys’ tab. You should see a screen similar to the below. The write API key is used for sending data to the channel and the read API key(s) is used to read the channel data. When we create a channel, by default, a write API key is generated. We generate read API keys by clicking the ‘Generate New Read API Key’ button under this tab.You can also add a note corresponding to each of the read API keys.



Now click on the ‘Data Import/Export’ tab and you should see a screen similar to the below. This tab is used to import the ‘Comma Separated Values(CSV)’ data from a file into the channel. You can also download the channel’s feed from here in CSV format. This tab also outlines how to send and view data by providing the URIs to the send and view APIs.



Now that we have come this far, let us get into some action. Let us try updating the fields in our channel using C# and .NET. I have used Visual Studio 2013 Express for Web, but the code relatively works the same with other versions too.

1. Open Visual Studio and Create a new project. In case you are prompted for a template, choose an empty web application.
2. In the solution explorer, right click on the project, Add --> New Item and choose Web Form. Once the form gets added, add the below code inside the form tag:

<label id="lblError" runat="server" style="display:none;font-weight:bold;color:red"></label>

 This label is used to display an error message in case we run into any exceptions.

1. Go to the code behind and add the below code in the page load:

Shrink ▲

try

{

const string WRITEKEY = "YOUR\_KEY";

string strUpdateBase = "http://api.thingspeak.com/update";

string strUpdateURI = strUpdateBase + "?key=" + WRITEKEY;

string strField1 = "18";

string strField2 = "42";

HttpWebRequest ThingsSpeakReq;

HttpWebResponse ThingsSpeakResp;

strUpdateURI+= "&field1=" + strField1;

strUpdateURI+= "&field2=" + strField2;

ThingsSpeakReq = (HttpWebRequest)WebRequest.Create(strUpdateURI);

ThingsSpeakResp = (HttpWebResponse)ThingsSpeakReq.GetResponse();

if(!(string.Equals(ThingsSpeakResp.StatusDescription,"OK")))

{

Exception exData = new Exception(ThingsSpeakResp.StatusDescription);

throw exData;

}

}

catch (Exception ex)

{

lblError.InnerText = ex.Message;

lblError.Style.Add("display", "block");

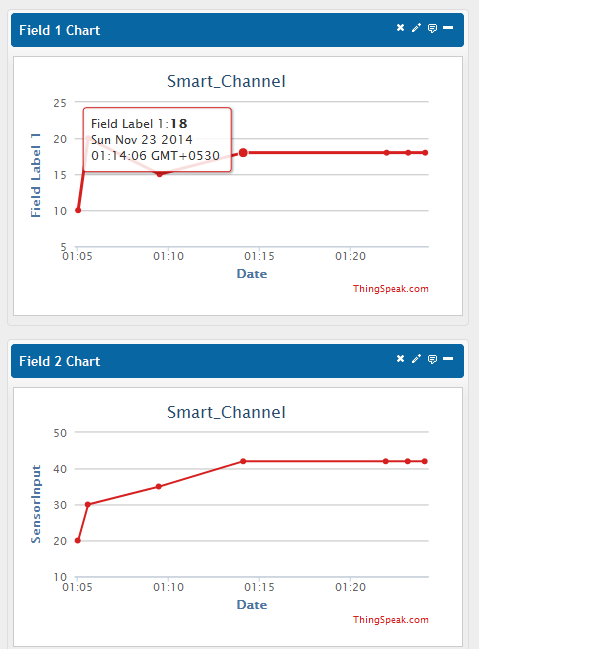
throw;

}

**Exploring the code:** I have started with building the URI for updating the fields. The initial URI is <http://api.thingspeak.com/update> to which the key has to be appended. Please note that you need to replace the string YOUR\_KEY with the actual write API key which you will get once you sign up and create a channel.

After that I have declared two constants which I will be pushing to my ThingSpeak channel. After that I am appending these fields to the URL under the parameters field1 and field2. You can push the other fields until field8 similarly. After this, I am creating a HTTPWebRequest object corresponding to this URI by using the ‘WebRequest.Create’ method. After this, I am retrieving the response using the GetResponse method and assigning the same to the response object. In case the data was successfully sent to the channel, the resulting StatusDescription will be ‘OK’. I am checking for any non-OK status descriptions and just in case of an error, I am throwing an exception and displaying the resultant message in a label.

After a series of updates, the charts in the private view tab for each of the fields will look like the below:



Each of the dots correspond to the value and the time at which the value was posted to the channel. Place the mouse over a dot to get more details on the exact date and the GMT offset from which the value was posted.

Please note that in the above example, I have sent some sample values to the channel. You can send any data here, say the periodic readings from a temperature sensor or RPM values from a motor. The Y-axis show the names that we specified to each of the labels.

**WOKWI**

Wokwi is an online Electronics simulator. You can use it to simulate Arduino, ESP32, and many other popular boards, parts and sensors.

Here are some quick examples of things we can make with Wokwi:

• Arduino Uno "Hello World"

• Blink an LED on ESP32

• Monitor the weather on ATtiny85

• Control 32 Servos with Arduino Mega

• Animate an LED Matrix with FastLED

• 7 Segment Counter with MicroPython on ESP32

Why is wokwi useful?

* No waiting for components, or downloading large software. Your browser has everything you need to start coding your next IoT project in seconds.
* We can't destroy the virtual hardware. So frying components is not an option. And unlike real hardware, you can always undo.

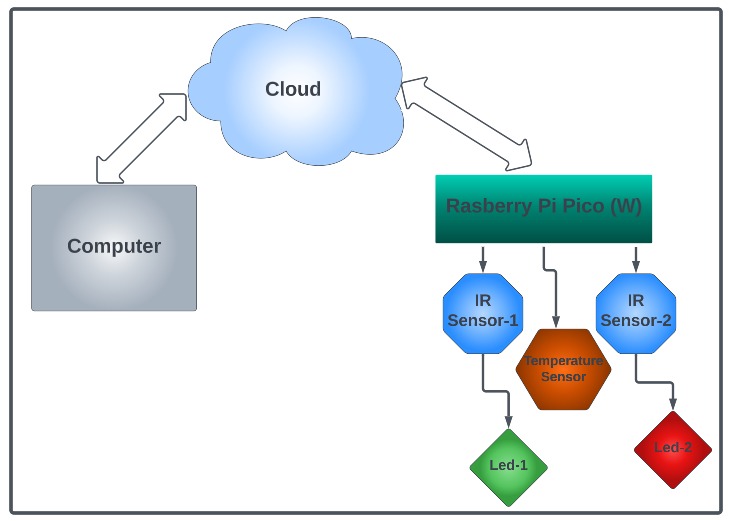
### Easy to get help and feedback.

* Separate hardware and software issues.
* No need to scavenge parts from old projects. Use as many parts as you need, without worrying about project price and stock.
* A place for you to share your projects, ask for help, and get inspiration.

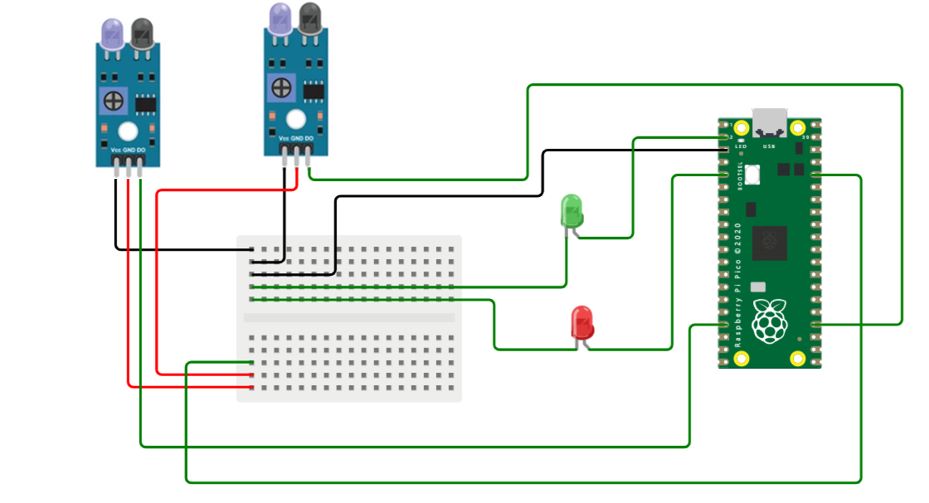
**Steps-**

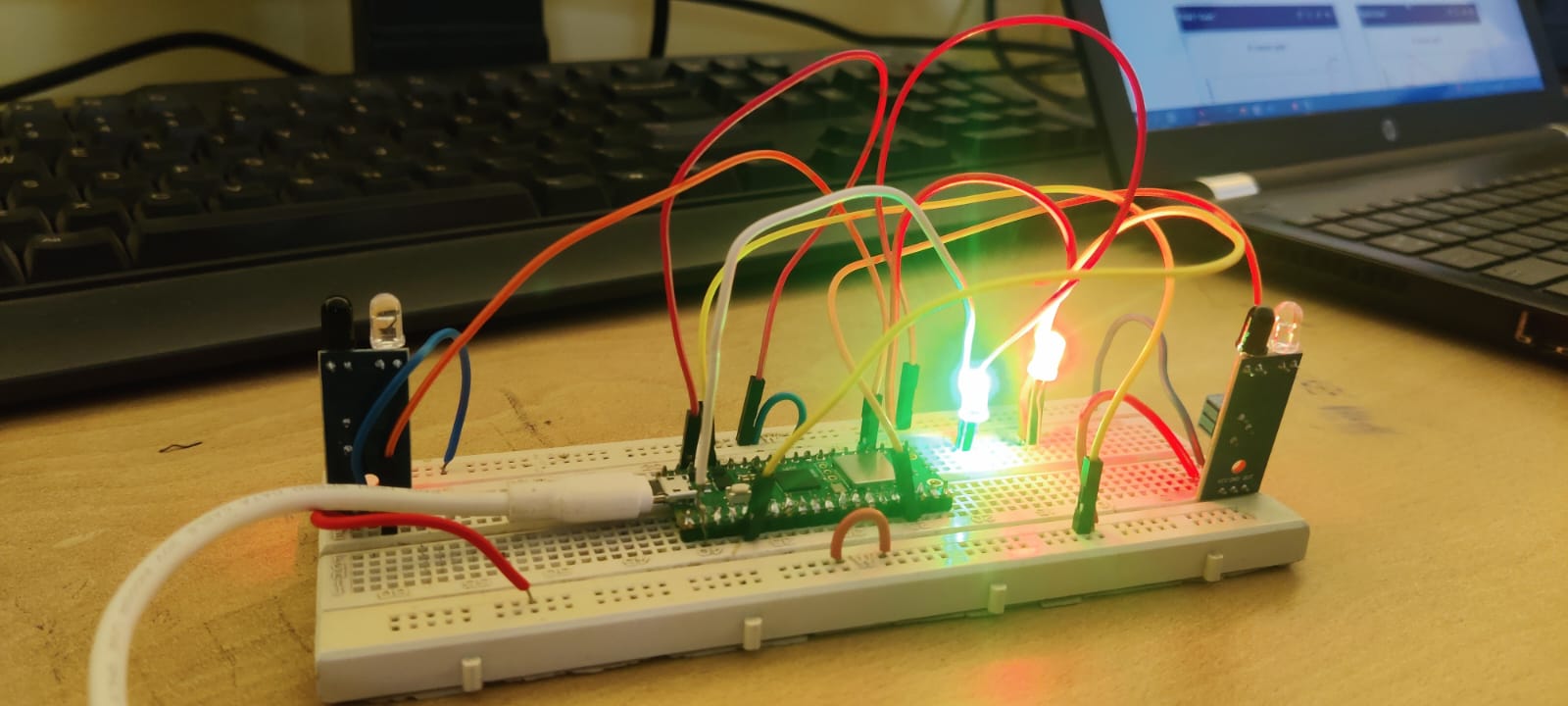
1. We create a working circuit using 2 Infrared Sensors , LEDs and Raspberry Pi Pico (W) board by connecting respective VCC, Ground and Pin connections.
2. Thonny, which is the Python IDE is used to construct the logical part of our project.
3. We run an infinite loop and check continuously if our IR sensors detect any presence of Entry or Exit.
4. The program counts any presence detected by the sensors namely IR1 and IR2 and increases ENTRY count if the former increases and increases the EXIT count if the later increases.
5. We also keep track of the total count of people present in the room by calculating Entry minus Exit, if we find there are people present in the room the green lights turn on and remain so if there are more entries, but turns off if the room becomes empty.
6. If the total count of people present in the room exceed 10 a Red Led turns on.
7. We also keep track of the temperature of the room.
8. We upload all the collected data that is Entry, Exit, Total count and the Temperature to a third party cloud which we can access world wide.

**Block Diagram-**



**Circuit Diagram-**



****

**Code-**

from machine import Pin,ADC

import urequests

from machine import Pin

import network

from time import sleep

c1 = 0

c2 = 0

total=0

led =Pin(1,Pin.OUT)

led2 =Pin(3,Pin.OUT)

ir1 = Pin(13, Pin.IN)

ir2 = Pin(18, Pin.IN)

temp=ADC(4)

HTTP\_HEADERS = {'Content-Type': 'application/json'}

THINGSPEAK\_WRITE\_API\_KEY = 'IP0WXDALZ3K6CWYK'

ssid = 'Redmi note 7 pro'

password = 'deba2211'

# Configure Pico W as Station

sta\_if=network.WLAN(network.STA\_IF)

sta\_if.active(True)

if not sta\_if.isconnected():

print('connecting to network...')

sta\_if.connect(ssid, password)

while not sta\_if.isconnected():

pass

print('network config:', sta\_if.ifconfig())

while True:

data=temp.read\_u16()

data1= data\*0.000050354

temp1=27-(data1-0.706)/0.001721

if ir1.value()==0:

c1 += 1

sleep(0.5)

print("in", c1)

else:

c1 = c1

sleep(0.5)

print("in",c1)

if ir2.value()==0:

c2 += 1

sleep(0.5)

print("out",c2)

else:

c2 = c2

sleep(0.5)

print("out",c2)

total=c1-c2

print("TOTAL", total)

if total>0:

led.value(1)

else:

led.value(0)

if total>=10:

led2.value(1)

else:

led2.value(0)

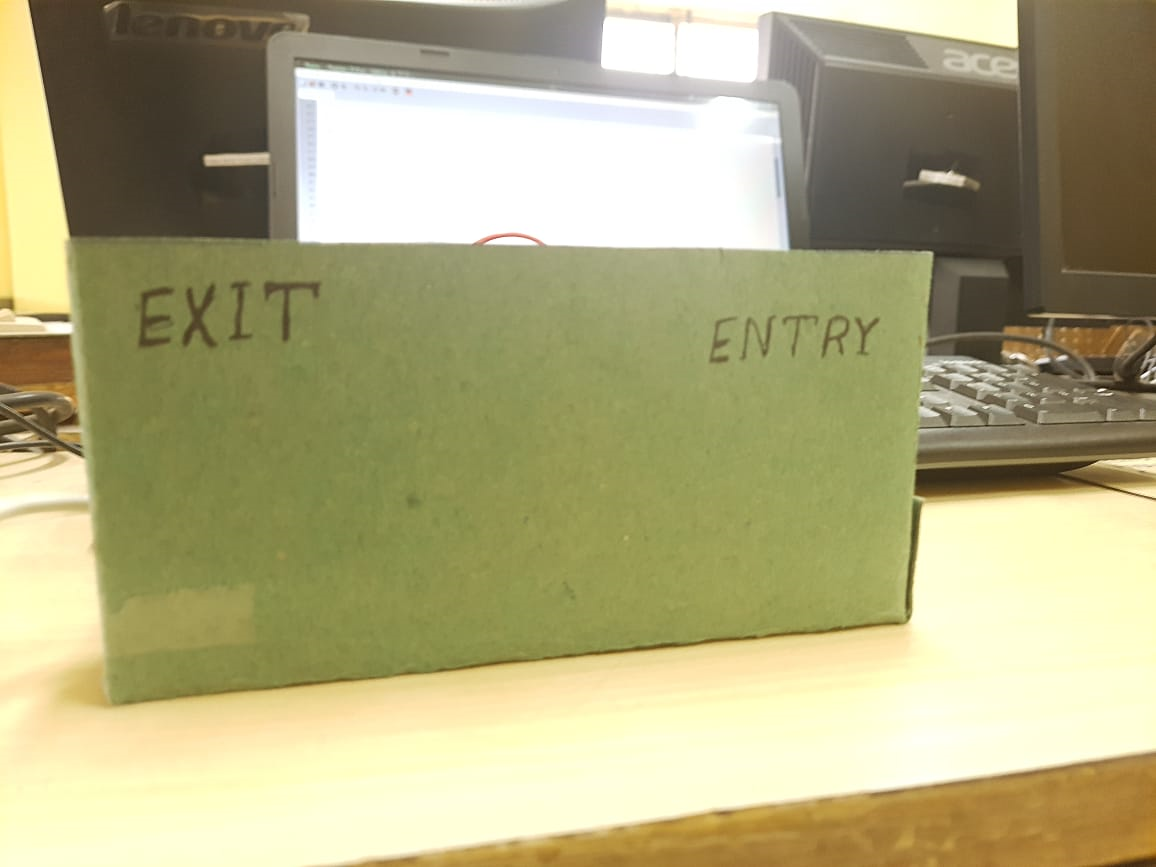
dht\_readings = {'field1':c1, 'field2':c2,'field3':total, 'field4':temp1}

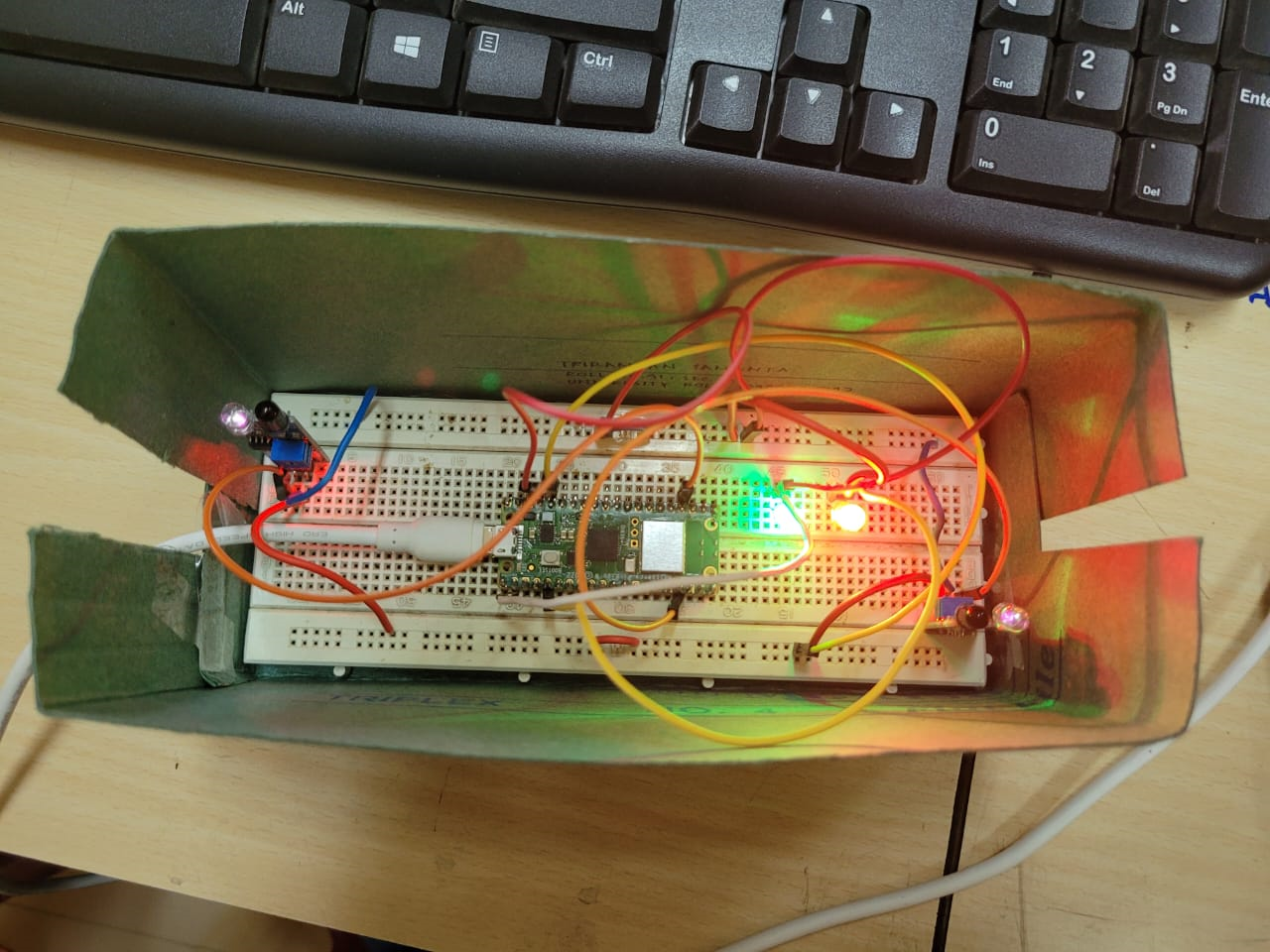
request = urequests.post( 'http://api.thingspeak.com/update?api\_key=' + THINGSPEAK\_WRITE\_API\_KEY, json = dht\_readings, headers = HTTP\_HEADERS )

request.close()

print(dht\_readings)

**Result-**

****

****

**Costing-**

* Rasberry Pi pico w
* IR Sensor
* Led
* Connecting Wires
* Breadboard

Total-Rs850

**Purpose Of Project-**

The purpose of this Automatic Auditorium System is that we can detect total head counts and also limit the entry of excess people. It also helps to detect the temperature which helps a lot in detecting excess heat which may be the case of fire, which can be avoided.

**Scope-**

* + Auditorium: Automatic lighting system in auditorium/halls that smartly detects human presence and turns its electrical facilities on and vice versa for empty auditorium or halls.
  + Stadium: Automatic lighting system in stadiums that smartly detects human presence and turns its electrical facilities on and vice versa for empty stadium and also gives information about temperature conditions around.
  + Conference room
  + Seminar hall
  + Hospital: In a similar way to halls this system can be used in medical facilities like hospitals and nursing homes where the authority can have information if total number of available space for patient entry is available or not.

**Conclusion-**

Smart Auditorium System is not just people counter but also provides with temperature measurement and maximum capacity of the room.

Automated people counters offer much more reliable and insightful data. The more information you have, the more your business decisions will benefit. However, if your business doesn’t see a lot of traffic, you might not need highly sophisticated data to know when your peaks are and how many visitors you host each day. For hourly traffic patterns, invest in an automated system.

Also the temperature sensor helps in detection of excess heat which provides a safety measure for the people.

**References-**

* https://www.geeksforgeeks.org, https://en.m.www.wikipedia.org, [www.itctech.com](http://www.itctech.com/), [www.academia.edu](http://www.academia.edu/)