**CHAPTER 1**

**INTRODUCTION**

The necessity for some means by which someone on the outside of a door could notify someone on the inside of his presence has been recognized for centuries. The earliest solution to this problem consisted of the simple expedient of knocking on the door with one's fist. As the human race grew in wisdom and technical sophistication, new and subtler methods were invented. The first of these was the mechanical door knocker, which saved man untold pain from bruised knuckles.

Then came the mechanical switch method requires that a mat containing many such switches be placed in front of the door in such a way that anyone approaching the door must step on it. Not only are such mats highly unesthetic, but it must be connected to the sound producing unit inside the house, requiring that a hole be drilled through a wall. This sort of installation is beyond what most homeowners have the time or skill to attempt, and is therefore usually done by professionals, greatly increasing total cost.

The photoelectric method requires that a light source and photodetector be mounted on either side of the path leading to the door. Here again, installation is usually done by professionals, and unless it is possible to hide the units in shrubbery, the light source and detector can be even more unesthetic than a mat.

Installation of a doorbell based on a capacitance proximity sensor is somewhat simpler than that of a mechanical switch or photoelectric unit in that the entire device can be mounted on the inside of the door. However, in order to obtain even the minimal detection range of two feet, metal sensor plates of several square feet must be used, and while these cannot be seen from the outside, they are painfully obvious from the inside.

In order to overcome the disadvantages of the methods discussed, it is clear that the ideal automatic doorbell would consist of a single small battery-operated unit, requiring no electrical connections at all. Installation would consist of driving a nail into the door and hanging the unit from it. The automatic doorbell herein described seeks to meet these criteria.

In the year 2020, Covid-19 virus has infected mankind on this earth and in this situation to stay safe we have to follow various precautions. Washing our hands regularly, wearing masks in public, and avoiding touching surfaces.

We can’t have the same old habits of eating, travelling, buying or even doing our routine works. Today homes have become offices and the internet is only saviour.

During lockdown the trend of ordering online and home delivery of various things groceries, electronics, books has increased. Though e-commerce companies claim that their employees take all safety measures from sanitizing to social distancing but when they come to house, they have to touch the doorbell and their temperature needs to be scanned. Also, our Corona warriors that is doctors, government officials have to go door to door for testing the local communities. So, touching surfaces is one we should avoid. As per studies, corona viruses stay on surfaces for nearly 24-48 hours.

Even when our family member or maybe a guest comes from outside, we need to check their temperature to see if it is less than 99.6’ F. If the temperature is more, we should tell them to seek a doctor or maybe live in isolated space and shall be allowed to visit the house only after proper sanitization is done.

**CHAPTER 2**

**LITERATURE SURVEY**

**2.1** **SMART BELL USING IOT**

Ambika, Baswaraj Gadgey, Veeresh Pujari, Pallavi B V, International Journal for Research in Applied Science & Engineering Technology (IJRASET) (2017).

This paper gives basic idea of how to remotely monitor and control door. It will work as and when bell rings at the door or any motion is sensed at the door, it will act as a trigger to the camera and the camera will capture the image of the person standing in front of the door, that will be shown to the registered user who is away from home and then he will identify the person and through the web server he can control the door lock.

Smart home security control system has become indispensable in daily life.

The design and development of a home security system, based on human face recognition technology and remotely monitoring technology, to confirm visitor identity and to control door accessibility has been reported in this paper.

**2.2 IOT SMART BELL NOTIFICATION SYSTEM:DESIGN AND IMPLEMENTATION**

Woo-hyuk Park and Yun-gyung cheong, 19th International Conference on Advanced Communication Technology (ICACT) (2017).

In this paper, they provide a security system that combines the functions of smart phone and home network system. It enables the users to monitor visitors in real-time, remotely via the IoT-based doorbell installed near the entrance door to a house.

If an outsider breaks into the house, the system can help identify the trespasser by acquiring CCTV evidence. Furthermore, this system can be used to report to the police or home security service provider immediately when a trespass occurs.

The design and development of a home security system combined with the functions of smart phone and home network system to confirm visitor identity has been reported in this paper.

**2.3 RASPBERRY PI BASED SMART DOORBELL**

Jie-Ci Yang, Chin-Lun Lai, Hsin-Teng Sheu and Jiann-Jone Chen, MDPI journal (2016).

The proposed system will allow to communicate between visitors and owners of the house. Video camera system (the photo of visitors will be sent to the owner of the house), instant message notification, SMS / MMS notification and dual audio / single sided process will be provided with videophone.

In addition, cloud storage of image data with a high resolution will be provided using the system with increasing safety and security is-sues, the use of smart door system increased consistently with the advent of security related electronics, such as digital door locks, advanced video conversation devices, and wire-less home security networks.

The design and development of a home security system combined with the functions of video camera system along with cloud storage to ensure security has been reported in this paper.

**2.4** **ARDUINO BASED WIRELESS DOOR BELL**

Anusha (Electronics Hub)

The aim of this project is to design a simple and cost-efficient wireless doorbell. This project is designed an Arduino based Wireless Doorbell using simple hardware.

The project is implemented using RF Module for wireless communication and also an Arduino UNO board to analyze the data

In order to ring the bell (or buzzer in this case), we need to push the button on the transmitter side of the circuit. When the button is pushed on the transmitter side, a logic ‘0’ will be detected by the Encoder IC. The Encoder IC will transmit this data serially through the RF Transmitter Module.

The transmitted data will be received by the RF Receiver Module and is given to the Decoder IC. The Decoder IC, then decodes the serial data to parallel data and transmits the Logic ‘0’ to Arduino. In the Arduino UNO’s, it is programmed such that, whenever a Logic ‘0’ is detected by the Arduino, the buzzer is turned on. Hence, whenever the button is pressed, the buzzer is turned on wirelessly.

The design and development of a home security system combined with the functions of RFID based system to ensure security has been reported in this paper.

**CHAPTER 3**

**SCOPE AND OBJECTIVES OF THE PROJECT**

**3.1 OBJECTIVES OF THE PROJECT**

Protecting our loved ones and ourselves from the deadly coronavirus by restricting entry to our house to only people with a normal body temperature. So, our objective is to build a smart anti-corona doorbell using an Arduino Nano.

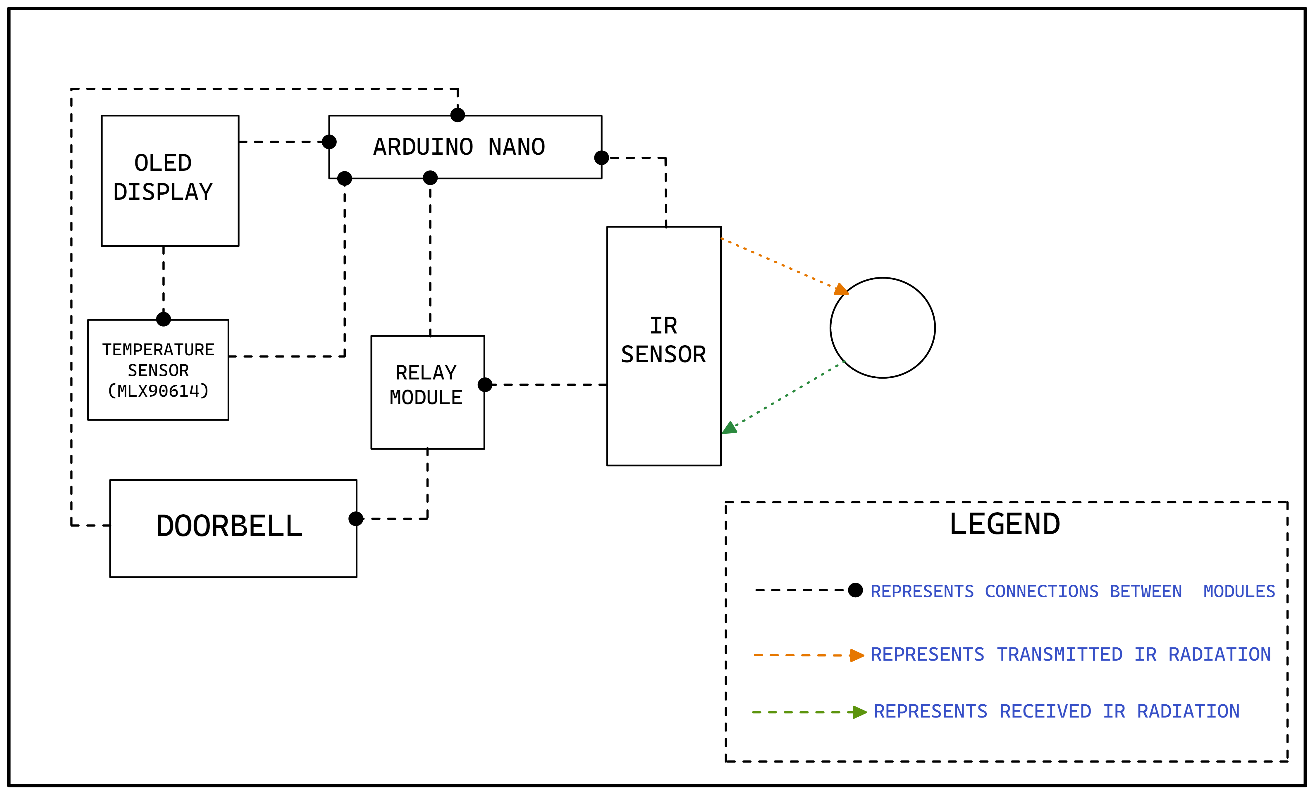
As COVID-19 is becoming worse day by day it is important for to be safe ourselves and ensure safety to our family. As we cannot measure a person’s temperature at home. So, for the safety concerns, we install this advanced doorbell at the entrance of our house to measure the visitor’s temperature and then by using it we can decide whether to let them in or not which ensures safety.

Thus, the main idea of the project is to ensure safety through technology.

**3.2 SCOPE OF THE PROJECT**

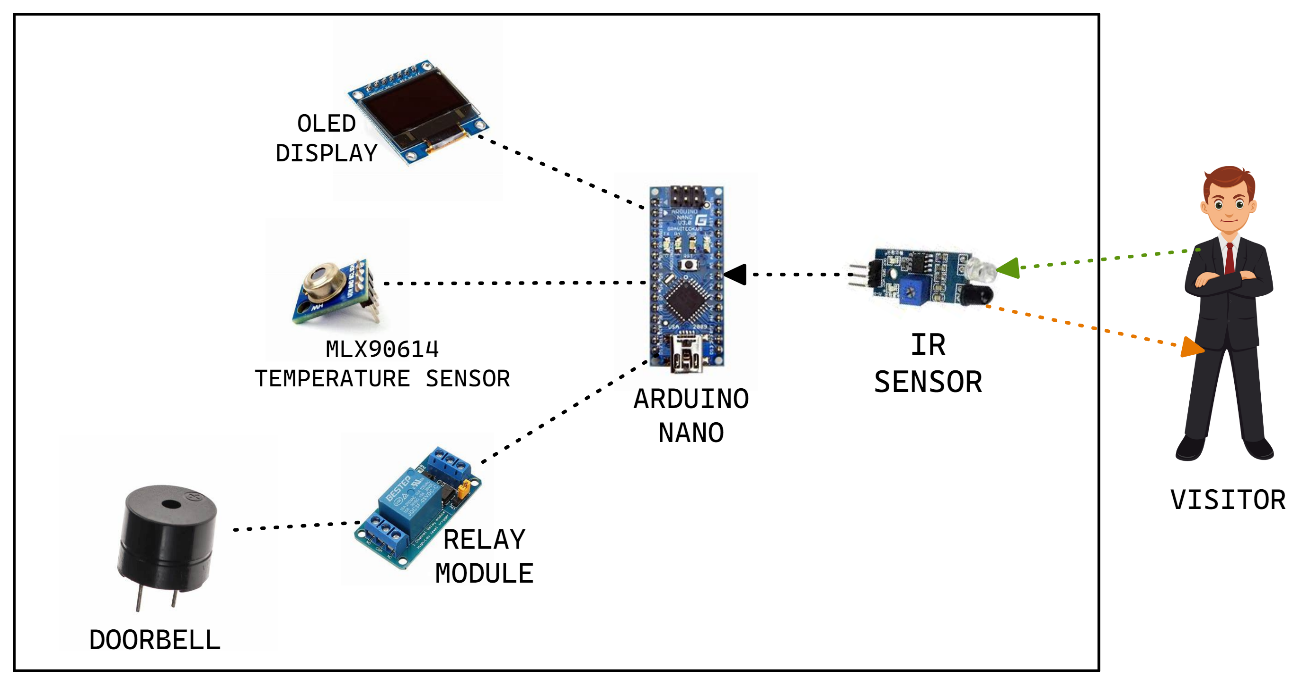
The ARDUINO is a trivial and compact processor which is having high computing power contrary to its size. ARDUINO has its own impact for its computing, processing speed and multiple functions which can be used to program and to interface manifold devices in the real time and it is précised by the C++ programming language which is simple, highly efficient, powerful and optimal. Subsequently ARDUINO can be appropriate for the high compatible, reliable and scalable purpose in numerous applications. This type of doorbell system has an immense impact in the busy human lifestyle to simplify their efforts to keep them and their away from a deadly disease like COVID-19 through this optimal solution.

**3.3 BLOCK DIAGRAM OF THE PROJECT**



**Fig:3.1 Block Diagram of the Project**

**3.4 PICTURE REPRESENTATION**



**Fig:3.2 Picture Representation**

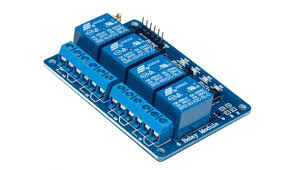
**CHAPTER 4**

**HARDWARE DETAILS OF THE PROJECT**

The Hardware components used in this project are

* OLED Display
* Bread Board
* ARDUINO NANO
* IR sensor
* Temperature Sensor (MLX90614)
* Door Bell/ Buzzer
* Relay Module

**4.1 RELAY MODULE**

****

**Fig:4.1 Relay Module**

• It is also able to control various appliances and other types of equipment with large current. Relay output maximum contact is AC250V 10A and DC5V 10A. One can connect a microcontroller with standard interface directly to it.

• Red working status indicator lights are conducive to the safe use. It has a wide range of applications such as all MCU control, industrial sector, PLC control, smart home control.

• This neat relay module features 4 x 5V relays rated at 10A/250V each. It is designed to switch up to 4 high current (10A) or high voltage (250V) loads with the help of microcontroller.

• Each relay can individually switch on/off by an opto-isolated digital input, which that can connect directly to a microcontroller output pin. It only requires a voltage of approximately 1.0V to switch the inputs on but can handle input voltages up to 5V. This makes it ideal for 1.0V to 5V devices.

**4.1.1 SPECIFICATIONS AND FEATURES**

• One normally closed contact and one normally open contact

• Channel: 4 channels

• Relay Operating Voltage: 3.3V to 5V

• Triode drive, increasing relay coil

• High impedance controller pin

• Pull-down circuit for avoidance of malfunction

• Power supply indicator and Control indicator lamp

• Power supply and relay instructions, lit, disconnect is off;

• Input signal, signal, common Terminal and start conducting;

• Useful for appliance control;

• DC or AC signal, control, you can control the 220V AC load;

• There is a normally open and one normally close contact;

• The module is compliant with international safety standards, control and load areas isolation trenches.

**4.2 TEMPERATURE SENSOR (MLX90614)**

****

**Fig:4.2 Temperature Sensor**

The MLX90614 is an infrared thermometer for non-contact temperature measurements. Both the IR sensitive thermopile detector chip and the signal conditioning ASIC are integrated in the same TO-39 can. Integrated into the MLX90614 are a low noise amplifier, 17-bit ADC and powerful DSP unit thus achieving high accuracy and resolution of the thermometer.

The thermometer comes factory calibrated with a digital SMBus output giving full access to the measured temperature in the complete temperature range(s) with a resolution of 0.02°C.

The user can configure the digital output to be pulse width modulation (PWM). As a standard, the 10-bit PWM is configured to continuously transmit the measured temperature in range of -20 to 120°C, with an output resolution of 0.14°C.

**4.3 IR SENSOR**

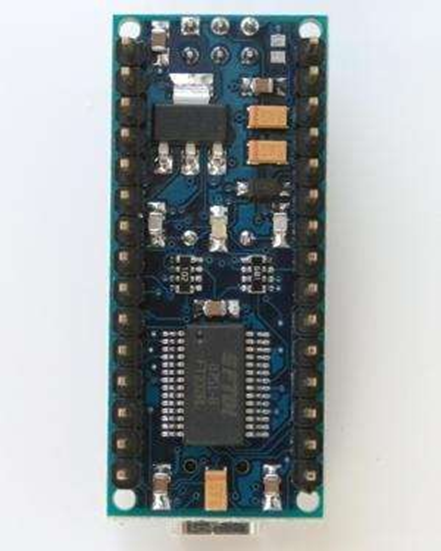
An infrared sensor is an electronic device, that emits in order to sense some aspects of the surroundings. An IR sensor can measure the heat of an object as well as detects the motion. These types of sensors measure only infrared radiation, rather than emitting it that is called a passive IR sensor. Usually, in the infrared spectrum, all the objects radiate some form of thermal radiation.



**Fig:4.3 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) 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.

**4.4 ARCHITECTURE OF ARDUINO**

  
**Fig:4.4 Arduino Nano Front** **Fig:4.5 Arduino Nano Rear**

**Overview:**

The Arduino Nano is a small, complete, and breadboard-friendly board based on the ATmega328 (Arduino Nano 3.o) or ATmegai68 (Arduino Nano 2.x). It has more or less the same functionality of the Arduino Duemilanove, but in a different package. It lacks only a DC power jack, and works with a Mini-B USB cable instead of a standard one. The Nano was designed and is being produced by Gravitech.

  
 **Fig:4.6 Pin Configuration Front Fig:4.7 Pin Configuration Back**

**Schematic and Design:**

Arduino Nano 3.0 (ATmega328): schematic, Eagle files.

Arduino Nano 2.3 (ATmegai68): manual (pdf), Eagle files.

Note: since the free version of Eagle does not handle more than 2 layers, and this version of the Nano is 4 layers, it is published here unrouted, so users can open and use it in the free version of Eagle.

**Power:**

The Arduino Nano can be powered via the Mini-B USB connection, 6-2oV unregulated external power supply (pin 30), or 5V regulated external power supply (pin 27). The power source is automatically selected to the highest voltage source.

The FTDI FT232RL chip on the Nano is only powered if the board is being powered over USB. As a result, when running on external (non-USB) power, the 3.3V output (which is supplied by the FTDI chip) is not available and the RX and TX LEDs will flicker if digital pins o or 1 are high.

**Specifications:**

Microcontroller Atmel ATmegai68 or ATmega32

Operating Voltage (logic level) 5V

Input Voltage (recommended) -12V

Input Voltage (limits) 6-20V

Digital I/O Pins 14 (of which 6 provide PWM output)

Analog Input Pins 8

DC Current per I/O Pin 40mA

Clock Speed 16 MHz

**Memory:**

The ATmegai68 has 16 KB of flash memory for storing code (of which **2** KB is used for the bootloader); the ATmega328 has 32 KB, (also with **2** KB used for the bootloader). The ATmegai68 has 1 KB of SRAM and 512 bytes of EEPROM (which can be read and written with the EEPROM library); the ATmega328 has **2** KB of SRAM and 1 KB of EEPROM.

**Input and Output:**

Each of the 14 digital pins on the Nano can be used as an input or output, using pinMode(), digitalWrite(), and digitalRead() functions. They operate at 5 volts. Each pin can provide or receive a maximum of 4o mA and has an internal pull-up resistor (disconnected by default) of 20-5o kOhms.

In addition, some pins have specialized functions:

* **Serial: o (RX) and 1 (TX).** Used to receive (RX) and transmit (TX) TTL serial data. These pins are connected to the corresponding pins of the FTDI USB-to-TTL Serial chip.
* **External Interrupts: 2 and 3.** These pins can be configured to trigger an interrupt on a low value, arising 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), ii (MOSI), 12 (MISO), 13 (SCK).** These pins support SPI communication, which, although provided by the underlying hardware, is not currently included in the Arduino language.
* **LED: 13.** There is a built-in LED connected to digital pin 13. When the pin is HIGH value, the LED is on, when the pin is LOW, ifs off.

The Nano has 8 analog inputs, 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 analogReference() function. Additionally, some pins have specialized functionality:

**I2C: 4 (SDA) and 5 (SCL).** Support I2C (TWI) communication using the Wire library (documentation on the Wiring website).

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:**

The Arduino Nano has a number of facilities for communicating with a computer, another Arduino, or other microcontrollers. The ATmegai68 and ATmega328 provide UART TTL (V) serial communication, which is available on digital pins o (RX) and 1 (TX). An FTDI FT232RL on the board channels this serial communication over USB and the FTDI drivers (included with the Arduino software) provide a virtual com port to software on the computer. The Arduino software includes a serial monitor which allows simple textual data to be sent to and from the Arduino board. The RX and TX LEDs on the board will flash when data is being transmitted via the FTDI chip and USB connection to the computer (but not for serial communication on pins o and 1).

A SoftwareSerial library allows for serial communication on any of the Nano's digital pins.

The ATmega168 and ATmega328 also support I2C (TWI) and SPI communication. The Arduino software includes a Wire library to simplify use of the I2C bus; see the documentation for details. To use the SPI communication, please see the ATmegai68 or ATmega328 datasheet.

**Programming:**

The Arduino Nano can be programmed with the Arduino software (download). Select "Arduino Diecimila, Duemilanove, or Nano w/ ATmegai68" or "Arduino Duemilanove or Nano w/ ATmega328" from the Tools> Board menu (according to the microcontroller on your board). For details, see the reference and tutorials.

The ATmegai68 or ATmega328 on the Arduino Nano comes preburned with a bootloader that allows you to upload new code to it without the use of an external hardware programmer. It communicates using the original STK5oo protocol (reference, C header files).

You can also bypass the bootloader and program the microcontroller through the ICSP (In-Circuit Serial Programming) header; see these instructions for details.

**Automatic (Software) Reset:**

Rather then requiring a physical press of the reset button before an upload, the Arduino Nano 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 FT232RL is connected to the reset line of the ATmegai68 or ATmega328 via a loo nano-farad capacitor. When this line is asserted (taken low), the reset line drops long enough to reset the chip. The Arduino software uses this capability to allow you to upload code by simply pressing the upload button in the Arduino environment. 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 Nano 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 Nano. 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.

**CHAPTER 5**

**SOFTWARE DETAILS OF THE PROJECT**

**5.1 ARDUINO IDE**

The Arduino Integrated Development Environment - or Arduino Software (IDE) - contains a text editor for writing code, a message area, a text console, a toolbar with buttons for common functions and a series of menus. It connects to the Arduino hardware to upload programs and communicate with them.

The Arduino IDE is incredibly minimalistic, yet it provides a near-complete environment for most Arduino-based projects. The top menu bar has the standard options, including “File” (new, load save, etc.), “Edit” (font, copy, paste, etc.), “Sketch” (for compiling and programming), “Tools” (useful options for testing projects), and “Help”. The middle section of the IDE is a simple text editor that where you can enter the program code. The bottom section of the IDE is dedicated to an output window that is used to see the status of the compilation, how much memory has been used, any errors that were found in the program, and various other useful messages.

Projects made using the Arduino are called sketches, and such sketches are usually written in a cut-down version of C++ (a number of C++ features are not included). Because programming a microcontroller is somewhat different from programming a computer, there are a number of device-specific libraries (e.g., changing pin modes, output data on pins, reading analog values, and timers). This sometimes confuses users who think Arduino is programmed in an “Arduino language.” However, the Arduino is, in fact, programmed in C++. It just uses unique libraries for the device.



**Fig:5.1 The Arduino IDE in its default state**

### The 6 Buttons

### While more advanced projects will take advantage of the built-in tools in the IDE, most projects will rely on the six buttons found below the menu bar.

### Image of The button bar

### Fig:5.2 The button bar

1. The **check mark** is used to verify your code. Click this once you have written your code.
2. The **arrow** uploads your code to the Arduino to run.
3. The **dotted paper** will create a new file.
4. The **upward arrow** is used to open an existing Arduino project.
5. The **downward arrow** is used to save the current file.
6. The far-right button is a **serial monitor**, which is useful for sending data from the Arduino to the PC for debugging purposes.

### There are plenty of other features available to consider on the IDE. But, having used many different types of microcontrollers and having been involved in multiple programming environments, it is shocking how simple the Arduino and its IDE is! In less than two minutes, you can get a simple C++ program uploaded onto the Arduino and have it running.

### 5.2 C++ PROGRAMMING LANGUAGE

### C++ is a general-purpose programming language created by Bjarne Stroustrup as an extension of the C programming language, or "C with Classes". The language has expanded significantly over time, and modern C++ now has object-oriented, generic, and functional features in addition to facilities for low-level memory manipulation.

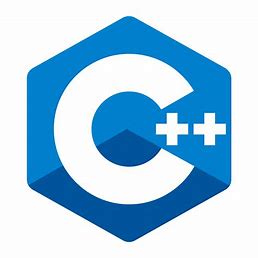
### It is almost always implemented as a compiled language, and many vendors provide C++ compilers, including the Free Software Foundation, LLVM, Microsoft, Intel, Oracle, and IBM, so it is available on many platforms.

### C++ was designed with an orientation toward systems programming and embedded, resource-constrained software and large systems, with performance, efficiency, and flexibility of use as its design highlights.

### C++ has also been found useful in many other contexts, with key strengths being software infrastructure and resource-constrained applications, including desktop applications, video games, servers (e.g. e-commerce, web search, or databases), and performance-critical applications (e.g. telephone switches or space probes).

### C++ is standardized by the International Organization for Standardization (ISO), with the latest standard version ratified and published by ISO in December 2020 as ISO/IEC 14882:2020 (informally known as C++20). The C++ programming language was initially standardized in 1998 as ISO/IEC 14882:1998, which was then amended by the C++03, C++11, C++14, and C++17 standards.

### The current C++20 standard supersedes these with new features and an enlarged standard library. Before the initial standardization in 1998, C++ was developed by Danish computer scientist Bjarne Stroustrup at Bell Labs since 1979 as an extension of the C language; he wanted an efficient and flexible language similar to C that also provided high-level features for program organization. Since 2012, C++ has been on a three-year release schedule with C++23 as the next planned standard.



**Fig:5.3 The C++ PROGRAMMING LANGUAGE**

### About C++ Programming

### •Multi-paradigm Language - C++ supports at least seven different styles of programming. Developers can choose any of the styles.

### •General Purpose Language - You can use C++ to develop games, desktop apps, operating systems, and so on.

### •Speed - Like C programming, the performance of optimized C++ code is exceptional.

### •Object-oriented - C++ allows you to divide complex problems into smaller sets by using objects.

### WHY C++?

### •C++ is used to develop games, desktop apps, operating systems, browsers, and so on because of its performance.

### •After learning C++, it will be much easier to learn other programming languages like Java, Python, etc.

### •C++ helps you to understand the internal architecture of a computer, how computer stores and retrieves information.

### •C++ has a large community that supports it by providing online courses and lectures, both paid and unpaid. Statistically speaking, C++ is the 6th most used and followed tag on Stack Overflow and GitHub.

### •Scalability refers to the ability of a program to scale. It means that the C++ program is capable of running on a small scale as well as a large scale of data. We can also build applications that are resource intensive.

### 5.3 FLOW CHART

### 

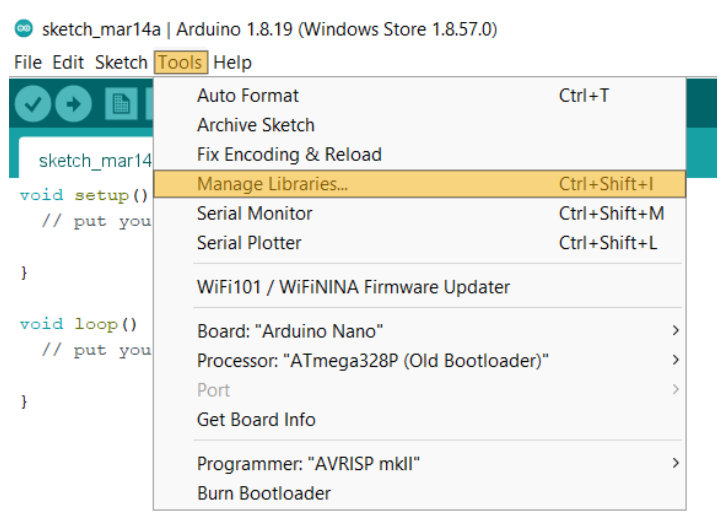
### Fig:5.4 Flow chart

**5.4 CODING OF THE PROJECT**

To continue with the coding part of the project, we need to install a library called as **ADAFRUIT** library on ARDUINO IDEto use **MLX90614** temperature sensor and **SSD1306** OLED display.

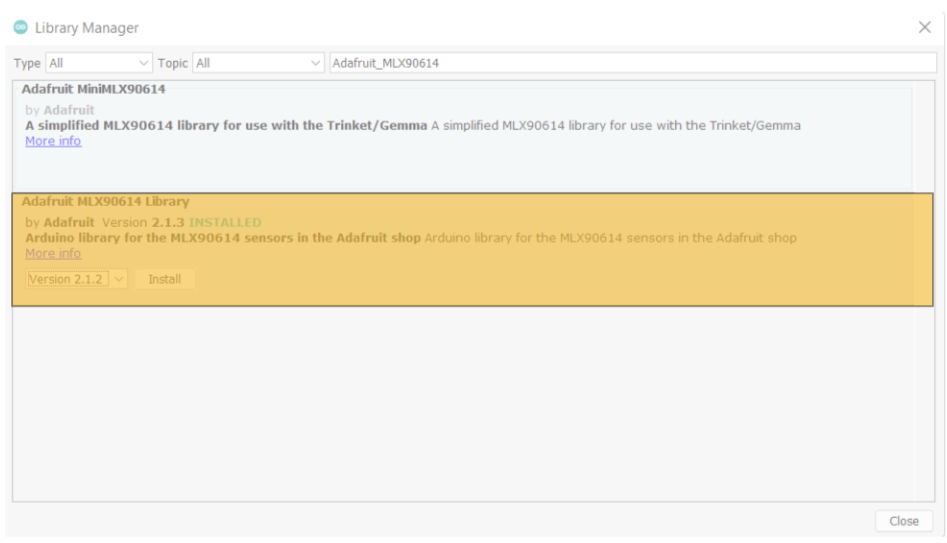
# **How to install ADAFRIUT library on ARDUINO IDE**

1. First of all, you need to **ARUDINO IDE** from the official website or from the Microsoft store.
2. After finishing installation, open up the IDE. A blank document will be opened. Give it a name and save in your desired folder.
3. Then, go to “**Tools”** section and find option called as “**Manage libraries”.**



**Fig:5.5 Tools Section**

1. Now a dialog box will appear. In the search bar type “Adafruit\_MLX90614”.
2. Select the latest version(2.1.2) and click on **install** button.
3. Now, install library for OLED display by following same method but type “**Adafruit\_SSD1306”.**



**Fig:5.6 Library Manager**

1. After installing the requried libraries, now head over to the “**Tools”** section again and make sure you’ve selected **ARDUINO NANO** from the listin the “**Board”** section and **“ATmega 328P(Old Bootloader)**  in the **“Processor”** section.

### 

### Fig:5.7 Board Section

1. Now, type the following code:

#include <Adafruit\_MLX90614.h>//Library for MLX90614 temperature sensor

#include <Adafruit\_SSD1306.h> //Library for SSD1306 OLED display

#include <SPI.h> //Library used for communicating with peripheral devices

#include <Wire.h> //Library used for communicating with I2C devices

#include <Adafruit\_GFX.h>

#define SCREEN\_WIDTH 128

#define SCREEN\_HEIGHT 32

#define OLED\_RESET 4

Adafruit\_SSD1306

display(SCREEN\_WIDTH, SCREEN\_HEIGHT, &Wire, OLED\_RESET); //Initialising the OLED display

Adafruit\_MLX90614 mlx = Adafruit\_MLX90614(); //Initialising the temperature sensor

#define ir 8

#define relay 3

void setup(){

pinMode(ir,INPUT);

pinMode(relay, OUTPUT);

mlx.begin();

display.begin(SSD1306\_SWITCHCAPVCC, 0x3C); //Initialising the OLED display

display.clearDisplay(); //clears the OLED display

display.setTextSize(2); //Sets text size

display.setTextColor(WHITE); //sets text color

display.setCursor(0,0); //Sets cursor to first row first column

display.print(" SMART DOORBELL");

display.display(); //this function is used for the changes to make effect

}

void loop(){

delay(500); //delay of 500 milliseconds

if(digitalRead(ir) == LOW){ //IR sensor will output 0 or LOW if it detects any object

float temp = mlx.readObjectTempC() + 4; //temp variable stores temperature

if (temp <= 38) {

display.clearDisplay();

display.setCursor(0,0);

display.print("Temp:");display.print(temp);display.print(" !WELCOME!"); //Prints temperature and welcomes user if the perso has normal body temperature

display.display();

digitalWrite(relay,HIGH); //Triggers relay to ring door bell

} else{

digitalWrite(relay, LOW);

display.clearDisplay();

display.setCursor(0,0);

display.print("HIGH TEMP! GO AWAY!"); //Turns off the relay which does not ring the bell and tells person to go away due to high temperature

display.display();

}

} else { //If the IR sensor does not detect anything

digitalWrite(relay,LOW);

display.clearDisplay();

display.setCursor(0,0);

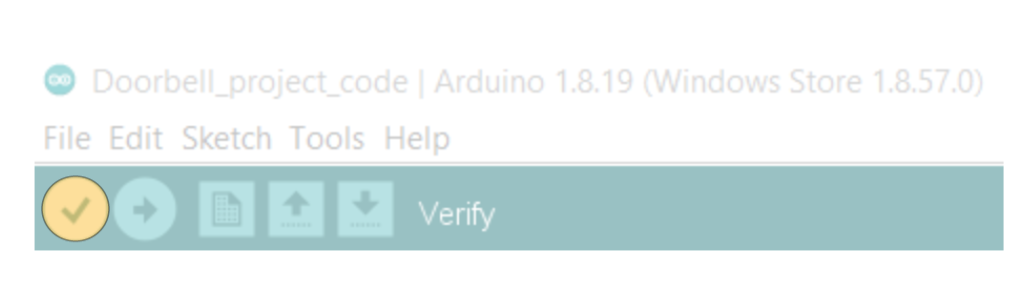
display.print(" SMART DOORBELL"); //Just prints

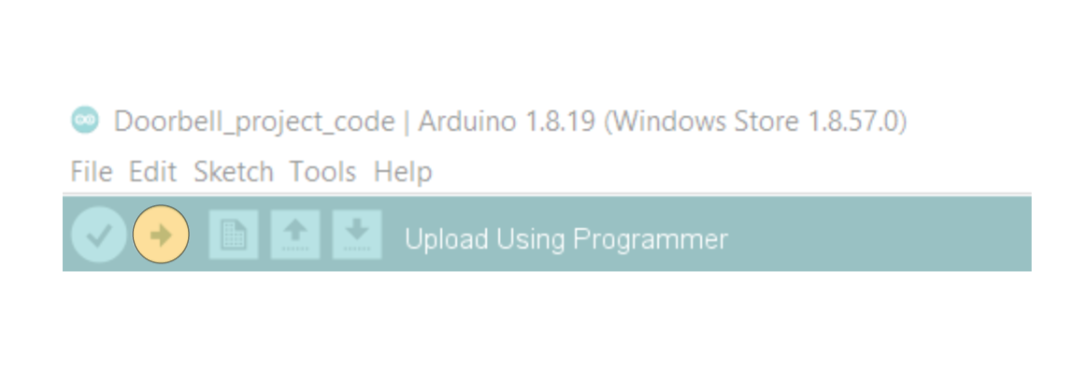
display.display();

}

}

1. Now, click on **Verify** and after successful verification, click on **Upload**.

 **Fig:5.8 Verify**



**Fig:5.9 Upload**

1. After uploading is done, you are now ready to test doorbell and check results.

**CHAPTER 6**

**RESULTS AND DISCUSSION**

**6.1 Case 1:**

If the measured temperature of the object is less than or equal to the defined temperature, then the following message will be printed on the screen.

**A watch on a person's wrist

Description automatically generated**

**6.2 Case 2:**

If the measured temperature of the object is greater to the defined temperature, then the following message will be printed on the screen.

**A picture containing person

Description automatically generated**

**CHAPTER 7**

**CONCLUSION**

This research can be further extended by linking it to devices through IOT. This will help in reducing the manpower to check the temperature. In the future aspect it can connect to the emergency medical services.

In general, this product will provide an extra layer of safety from COVID-19 infection by making automatic decisions through the developed system as there is potential chance of new variants of COVID may arise in the coming days, the process of primary detection of temperature is very important. This can be easily done by this automated doorbell.

**REFERENCES**

1. Ambika, Baswaraj Gadgey, Veeresh Pujari, Pallavi B V “Smart Bell Using IOT” International Journal for Research in Applied Science & Engineering Technology (IJRASET) (2017).
2. Woo-hyuk Park and Yun-Gyung cheong “IoT smart bell notification system: Design and implementation” 19th International Conference on Advanced Communication Technology (ICACT) (2017).
3. Jie-Ci Yang, Chin-Lun Lai, Hsin-Teng Sheu and Jiann-Jone Chen “An Intelligent Automated Door Control System Based on a Smart Camera” http://www.mdpi.com/journal/sensors (2016).
4. Arduino based Wireless Doorbell: www.electronicshub.org/wireless-door-bell

**PUBLICATIONS / CONFERENCE CERTIFICATES**