|  |
| --- |
| A Case Study Report  Submitted in partial fulfilment of the  Requirements for the Course of  THEME BASED LAB  IN  BE 3/4 (IT) VI-SEMESTER  By  **Kabir Varsha(1602-16-737-055)**  **Devi kadampalli(1602-16-737-311)**  vlogo.gif  **Department Of Information Technology**  **Vasavi College of Engineering**  **(Affliated to Osmania University)**  **Ibrahimbagh,Hyderabad-31**  **2019**    **Vasavi College of Engineering (Autonomous)**  **Ibrahimbagh, Hyderabad-31**  **Department of Information Technology**  **vlogo.gif**  **DECLARATION BY THE CANDIDATE**  We, **Devi kadampalli and Kabir varsha** bearing hall ticket numbers, **1602-16-737-311** and **1602-16-737-055** hereby declare that the Case study report entitled **“Smart Door bell system”** under the guidance of **Mr.Rajashekar G**., Associate Professor, Department of **Information Technology**, VCE, Hyderabad is submitted in partial fulfilment of the requirement for the course of **Theme Based Lab** in BE ¾(IT), VI-Semester. This is a record of bonafide work carried out by me and the Design embodied in this project report has not been submitted by any other.  Devi kadampalli(1602-16-737-311)  Kabir varsha(1602-16-737-088)  **Vasavi College of Engineering (Autonomous)**  **Ibrahimbagh, Hyderabad-31**  **Department of Information Technology**    **BONAFIDE CERTIFICATE**  This is to certify that the project entitled  **SMART DOORBELL SYSTEM** being submitted by Devi and varsha**,** bearinghall ticket numbers **1602-16-737-311** and **1602-16-737-055,** in partial fulfilment of the requirement for the course of **Theme based Lab** in BE ¾ (IT) VI-Semester is a record of bonafide work carried out by him/her under my guidance.  **Mr. Krishna Kishore Dr.K.Rammohan rao Assistant professor Professor & HOD Internal Guide External Examiner Dept. of IT**  **ACKNOWLEDGEMENT**  We take this opportunity with pride and enormous gratitude, to express the deeply embedded feeling and gratefulness to our respectable guide **Mr. Rajashekar G.,** Department of Information Technology , whose guidance was unforgettable and innovative ideas as well as his constructive suggestions has made the presentation of my project a grand success.  We are thankful to **Dr. K. Ram Mohan Rao,**  Head of Department (IT), **Vasavi College of Engineering** for their help during our course work.  Finally at last but not least express our heart full thanks to the management of our college, **Vasavi College of Engineering** for providing the necessary arrangements and support to complete my project work successively.  **Table of Contents**  Abstract  1.Introduction………………………………………………………………………………….1  1.1 Project ………………………………………………………3  1.2 Purpose of Project……………………………………………………………………………4  2.Specific Requirements………………………………………………………………….6  2.1 Hardware Requirements………………………………………………………6  2.2 Software Requirements……………………………………………………….6  3.Proposed Work…………………………………………………………………………...8  3.1 Use Cases………………………………………………………………………..…..8  3.2 UI Prototypes……………………………………………………………….…..…9  3.3Architecture and Technology used……………………….…….……..…10  3.4 Design…………………………………………………………………..…………….11  i.Use-Case Diagram……………………………………………...…..………….11  ii.UML Diagram…………………………………………………………..………..12  3.5 Implementation…………………………………………………………………..17  3.6 Testing………………………………………………………………..……………….22  4.Results…………………………………………………………………………………………23  5.Github Links…………………………………………………………………………………23  6.References…………………………………………………………………………………..25 Abstract In a busy world where people often get busy with their daily schedules, it is difficult for the employed people to remember schedules outside their profession. The security is the common risk faced by the busy people because they leave their home with simple lock system. Our aim is to target for security and convenience of them who are absorbed with work life. In this process we are going to use arduino, GPRS shield, PIR sensor, RGB LED etc., here  **INTRODUCTION**  **What is the project?**  The project is one of those IOT projects that includes the use of sensors and notifies the user regarding the activity that has been detected. **Main purpose of doing a project:**The main aim of doing this project is to ensure the security and convenience of people who are absorbed with work life. In other words make the people aware of the presence of any intruder at their home.To alert the people with a message sent to their mail as well as to their mobile. By this message they can verify whether the visitor is a intruder or not. If they find the visitor as an intruder, then an immediate action will be taken by them against the intruder. |
| 1. **Specific Requirements**    1. **Hardware Requirements**   Arduino UNO Board  RGB LED  GSM Modem  Jumper Wires  PIR Sensor  Piezo Buzzer  Bread Board   * 1. **Software Requirements**   Arduino IDE  SMTP2GO  **PROPOSED WORK**  **OBJECTIVES-**  **USE CASES OR STORIES :**   1. **Human Presence Sensing:**   Senses the presence of a human using an PIR sensor.   1. **Notifying by alarm:**   Whenever a human activity is detected, the doorbell will switch on automatically and alert the residents regarding it.   1. **Notifying the user:**   It can send a notification to user at any part of the world via SMS or mail.   1. **Authenticating the person:**   The user can authenticate the entry of that person via an application. |
| Hardware requirements are:   1. GSM MODULE :   we are going to see how to interface GSM Module to Arduino. There are different kinds of GSM modules available in market. We are using the most popular module based on Simcom SIM900 and Arduino Uno for this tutorial. Interfacing a GSM module to Arduino is pretty simple. You only need to make 3 connections between the gsm module and arduino. So lets get to business!  A GSM Module is basically a GSM Modem Â (like SIM 900) connected to a PCB with different types of output taken from the board – say TTL Output (for Arduino, 8051 and other microcontrollers) and RS232 Output to interface directly with a PC (personal computer). The board will also have pins or provisions to attach mic and speaker, to take out +5V or other values of power and ground connections. These type of provisions vary with different modules.  Lots of varieties of GSM modem and GSM Modules are available in the market to choose from. For our project of connecting a gsm modem or module to arduino and hence send and receive sms using arduino – its always good to choose an arduino compatible GSM Module – that is a GSM module with TTL Output provisions. Notes on GSM Module 1. We use SIM900 GSM Module – This means the module supports communication in 900MHz band. We are from India and most of the mobile network providers in this country operate in the 900Mhz band. If you are from another country, you have to check the mobile network band in your area. A majority of United States mobile networks operate in 850Mhz band (the band is either 850Mhz or 1900Mhz). Canada operates primarily on 1900 Mhz band. Please read this wiki entry on [GSM Frequency Bands around the World](http://en.wikipedia.org/wiki/GSM_frequency_bands" \l "GSM_frequency_usage_around_the_world).  2. Check the power requirements of GSM module – GSM modules are manufactured by different companies. They all have different input power supply specs. You need to double check your GSM modules power requirements. In this tutorial, our gsm module requires a 12 volts input. So we feed it using a 12V,1A DC power supply. I have seen gsm modules which require 15 volts and some other types which needs only 5 volts input. They differ with manufacturers. If you are having a 5V module, you can power it directly from Arduino’s 5V out.  Note:- GSM Modules are manufactured by connecting a particular GSM modem to a PCB and then giving provisions for RS232 outputs, TTL outputs, Mic and Speaker interfacing provisions etc. The most popular modem under use is SIM 900 gsm modem from manufacturer SIMCom. They also manufacture GSM Modems in bands 850, 300 and other frequency bands.  3. Check for TTL Output Pins in the module – You can feed the data from gsm module directly to Arduino only if the module is enabled with TTL output pins. Otherwise you have to convert the [RS232 data to TTL](http://en.wikipedia.org/wiki/MAX232) using MAX232 IC and feed it to Arduino. Most of the gsm modules in market are equipped with TTL output pins. Just ensure you are buying the right one.So that’s all about the gsm module basics. Now lets Gsm_Module_Arduino_Interface1 Booting the GSM Module! 1. Insert the SIM card to GSM module and lock it.  2. Connect the adapter to GSM module and turn it ON!  3. Now wait for some time (say 1 minute) and see the blinking rate of ‘status LED’ Â or ‘network LED’ (GSM module will take some time to establish connection with mobile network)  4. Once the connection is established successfully, the status/network LED will blink continuously every 3 seconds. You may try making a call to the mobile number of the sim card inside GSM module. If you hear a ring back, the gsm module has successfully established network connection.  Okay! Now let’s see how to connect a gsm module to Arduino! Connecting GSM Module to Arduino There are two ways of connecting GSM module to arduino. In any case, the communication between Arduino and GSM module is serial. So we are supposed to use serial pins of Arduino (Rx and Tx). So if you are going with this method, you may connect the Tx pin of GSM module to Rx pin of Arduino and Rx pin of GSM module to Tx pin of Arduino. You read it right ? GSM Tx –> Arduino Rx and GSM Rx –> Arduino Tx. Now connect the ground pin of arduino to ground pin of gsm module! So that’s all! You made 3 connections and the wiring is over! Now you can load different programs to communicate with gsm module and make it work.  Note:- The problem with this connection is that, while programming Arduino uses serial ports to load program from the Arduino IDE. If these pins are used in wiring,Â  the program will not be loaded successfully to Arduino. So you have to disconnect wiring in Rx and Tx each time you burn the program to arduino. Once the program is loaded successfully, you can reconnect these pins and have the system working!  To avoid this difficulty, I am using an alternate method in which two digital pins of arduino are used for serial communication. We need to select two PWM enabled pins of arduino for this method. So I choose pins 9 and 10 (which are PWM enabled pins). This method is made possible with the [SoftwareSerial Library](http://arduino.cc/en/Reference/softwareSerial) of Ardunio. SoftwareSerial is a library of Arduino which enables serial data communication through other digital pins of Arduino. The library replicates hardware functions and handles the task of serial communication.  I hope you understood so far! Â Lets get to the circuit diagram! So given below is the circuit diagram to connect gsm module to arduino – and hence use the circuit to send sms and receive sms using arduino and gsm modem.    Interfacing-GSM-Module-and-Arduino  Make the connections as shown! Now lets get to the coding part. The program has two objectives as described below:-  1) Send SMS using Arduino and GSM Module – to a specified mobile number inside the program  2) Receive SMS using Arduino and GSM Module – to the SIM card loaded in the GSM Module.   1. Arduino uno:   The Arduino UNO is an open-source microcontroller board based on the [Microchip](https://en.wikipedia.org/wiki/Microchip_Technology" \o "Microchip Technology) [ATmega328P](https://en.wikipedia.org/wiki/ATmega328P" \o "ATmega328P)microcontroller and developed by arduino. The board is equipped with sets of digital and analog input/output (I/O) pins that may be interfaced to various expansion boards (shields) and other circuits. The board has 14 Digital pins, 6 Analog pins, and programmable with the [Arduino IDE](https://en.wikipedia.org/wiki/Arduino" \l "Software" \o "Arduino) (Integrated Development Environment) via a type B USB cable. It can be powered by a USB cable or by an external 9 volt battery, though it accepts voltages between 7 and 20 volts. It is also similar to the Arduino Nano and Leonardo. The hardware reference design is distributed under a [Creative Commons](https://en.wikipedia.org/wiki/Creative_Commons" \o "Creative Commons) Attribution Share-Alike 2.5 license and is available on the Arduino website. Layout and production files for some versions of the hardware are also available. "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. The ATmega328 on the Arduino Uno comes preprogrammed with a bootloader that allows uploading new code to it without the use of an external hardware programmer.[]](https://en.wikipedia.org/wiki/Arduino_Uno" \l "cite_note-What_is_Arduino?-3) It communicates using the original STK500 protocol. The Uno also differs from all preceding boards in that it does not use the FTDI USB-to-serial driver chip. Instead, it uses the Atmega16U2 (Atmega8U2 up to version R2) programmed as a USB-to-serial converter.  220px-Arduino_uno General Pin function :LED: 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. 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 - 20V), the USB connector (5V), or the VIN pin of the board (7-20V). Supplying voltage via the 5V or 3.3V pins bypasses the regulator, and can damage the board.  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.  Reset: Typically used to add a reset button to shields which block the one on the board. Special Pin Functions : Each of the 14 digital pins and 6 Analog pins on the Uno 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 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.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.  In addition, some pins have specialized functions:   * Serial / [UART](https://en.wikipedia.org/wiki/UART" \o "UART): pins 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: pins 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. * [PWM](https://en.wikipedia.org/wiki/Pulse-width_modulation" \o "Pulse-width modulation) (Pulse Width Modulation): 3, 5, 6, 9, 10, and 11 Can provide 8-bit PWM output with the analogWrite() function. * [SPI](https://en.wikipedia.org/wiki/Serial_Peripheral_Interface" \o "Serial Peripheral Interface) (Serial Peripheral Interface): 10 (SS), 11 (MOSI), 12 (MISO), 13 (SCK). These pins support SPI communication using the SPI library. * TWI (Two Wire Interface) / [I²C](https://en.wikipedia.org/wiki/I%C2%B2C" \o "I²C): A4 or SDA pin and A5 or SCL pin. Support TWI communication using the Wire library. * AREF (Analog REFerence): Reference voltage for the analog inputs.[[7]](https://en.wikipedia.org/wiki/Arduino_Uno" \l "cite_note-website-7)  Communication :The 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. 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 allows serial communication on any of the Uno's digital pins.  1. **pizeo buzzer**     A **buzzer** or **beeper** is an [audio](https://en.wikipedia.org/wiki/Sound" \o "Sound) signalling device, which may be [mechanical](https://en.wikipedia.org/wiki/Machine" \o "Machine), [electromechanical](https://en.wikipedia.org/wiki/Electromechanics" \o "Electromechanics), or [piezoelectric](https://en.wikipedia.org/wiki/Piezoelectricity" \o ") (*piezo* for short). Typical uses of buzzers and beepers include [alarm devices](https://en.wikipedia.org/wiki/Alarm_devices" \o "Alarm devices), [timers](https://en.wikipedia.org/wiki/Timer" \o "Timer), and confirmation of user input such as a mouse click or keystroke  FII0D11IAP6UG5A.LARGE. **Electromechanical** The electric buzzer was invented in 1831 by [Joseph Henry](https://en.wikipedia.org/wiki/Joseph_Henry" \o "Joseph Henry). They were mainly used in early [doorbells](https://en.wikipedia.org/wiki/Doorbell" \o "Doorbell) until they were phased out in the early 1930s in favor of musical chimes, which had a softer tone. **Piezoelectric**[Piezoelectric](https://en.wikipedia.org/wiki/Piezoelectricity" \o "Piezoelectricity) buzzers, or piezo buzzers, as they are sometimes called, were invented by Japanese manufacturers and fitted into a wide array of products during the 1970s to 1980s. This advancement mainly came about because of cooperative efforts by Japanese manufacturing companies. In 1951, they established the Barium Titanate Application Research Committee, which allowed the companies to be "competitively cooperative" and bring about several piezoelectric innovations and inventions.**Electromechanical** Early devices were based on an electromechanical system identical to an [electric bell](https://en.wikipedia.org/wiki/Electric_bell" \o "Electric bell) without the metal gong. Similarly, a [relay](https://en.wikipedia.org/wiki/Relay" \o "Relay) may be connected to interrupt its own actuating [current](https://en.wikipedia.org/wiki/Electric_current" \o "Electric current), causing the [contacts](https://en.wikipedia.org/wiki/Switch" \o "Switch) to buzz. Often these units were anchored to a wall or ceiling to use it as a sounding board. The word "buzzer" comes from the rasping noise that electromechanical buzzers made. **Mechanical** A [joy buzzer](https://en.wikipedia.org/wiki/Joy_buzzer" \o "Joy buzzer) is an example of a purely mechanical buzzer and they require drivers. Other examples of them are doorbells. **Piezoelectric** [IMG_256](https://en.wikipedia.org/wiki/File:2007-07-24_Piezoelectric_buzzer.jpg)  Piezoelectric disk beeper  A [piezoelectric](https://en.wikipedia.org/wiki/Piezoelectric" \o "Piezoelectric) element may be driven by an [oscillating](https://en.wikipedia.org/wiki/Oscillation" \o "Oscillation) electronic circuit or other [audio signal](https://en.wikipedia.org/wiki/Audio_signal" \o "Audio signal)source, driven with a [piezoelectric audio amplifier](https://en.wikipedia.org/wiki/Piezoelectric_audio_amplifier" \o "Piezoelectric audio amplifier). Sounds commonly used to indicate that a button has been pressed are a click, a ring or a beep. **Modern applications**While technological advancements have caused buzzers to be impractical and undesirable[*[citation needed](https://en.wikipedia.org/wiki/Wikipedia:Citation_needed" \o "Wikipedia:Citation needed)*], there are still instances in which buzzers and similar circuits may be used. Present day applications include:  * Judging panels * Educational purposes * [Annunciator panels](https://en.wikipedia.org/wiki/Annunciator_panel" \o "Annunciator panel) * Electronic [metronomes](https://en.wikipedia.org/wiki/Metronome" \o "Metronome) * [Game show](https://en.wikipedia.org/wiki/Game_show" \o "Game show) [lock-out device](https://en.wikipedia.org/wiki/Lock-out_device" \o "Lock-out device) * [Microwave ovens](https://en.wikipedia.org/wiki/Microwave_oven" \o "Microwave oven) and other [household appliances](https://en.wikipedia.org/wiki/Major_appliance" \o "Major appliance) * [Sporting](https://en.wikipedia.org/wiki/Sport" \o "Sport) events such as [basketball](https://en.wikipedia.org/wiki/Basketball" \o "Basketball) games * Electrical [alarms](https://en.wikipedia.org/wiki/Alarms" \o "Alarms) * [Joy buzzer](https://en.wikipedia.org/wiki/Joy_buzzer" \o "Joy buzzer) (mechanical buzzer)   **Interfacing of buzzer With Arduino:**  **F3DDJQ4IAP6UGTG.LARGE**  In this circuit, the positive lead of the buzzer is connected to pin 13 of the Arduino, the negative lead of the buzzer is connected to GND.  IMG_256   1. **PIR Sensor:**   PIR sensors are more complicated than many of the other sensors explained in these tutorials (like photocells, FSRs and tilt switches) because there are multiple variables that affect the sensors input and output. To begin explaining how a basic sensor works, we'll use this rather nice diagram  The PIR sensor itself has two slots in it, each slot is made of a special material that is sensitive to IR. The lens used here is not really doing much and so we see that the two slots can 'see' out past some distance (basically the sensitivity of the sensor). When the sensor is idle, both slots detect the same amount of IR, the ambient amount radiated from the room or walls or outdoors. When a warm body like a human or animal passes by, it first intercepts one half of the PIR sensor, which causes a positive differential change between the two halves. When the warm body leaves the sensing area, the reverse happens, whereby the sensor generates a negative differential change. These change pulses are what is detected.    IMG_256 The PIR Sensor The IR sensor itself is housed in a hermetically sealed metal can to improve noise/temperature/humidity immunity. There is a window made of IR-transmissive material (typically coated silicon since that is very easy to come by) that protects the sensing element. Behind the window are the two balanced sensors.  [IMG_256](https://learn.adafruit.com/assets/512)  [Left image from Murata datasheet](http://learn.adafruit.com/system/assets/assets/000/010/137/original/pyroelectrics21e.pdf)  [IMG_257](https://learn.adafruit.com/assets/513)  [Image from RE200B datasheet](http://learn.adafruit.com/system/assets/assets/000/010/134/original/RE200B.pdf)  You can see above the diagram showing the element window, the two pieces of sensing material  [IMG_258](https://learn.adafruit.com/assets/514)  [Image from RE200B datasheet](http://learn.adafruit.com/system/assets/assets/000/010/134/original/RE200B.pdf)  This image shows the internal schematic. There is actually a JFET inside (a type of transistor) which is very low-noise and buffers the extremely high impedence of the sensors into something a low-cost chip (like the BIS0001) can sense. Lenses PIR sensors are rather generic and for the most part vary only in price and sensitivity. Most of the real magic happens with the optics. This is a pretty good idea for manufacturing: the PIR sensor and circuitry is fixed and costs a few dollars. The lens costs only a few cents and can change the breadth, range, sensing pattern, very easily.  In the diagram up top, the lens is just a piece of plastic, but that means that the detection area is just two rectangles. Usually we'd like to have a detection area that is much larger. To do that, we use [a simple lens](http://en.wikipedia.org/wiki/Lens_(optics)) such as those found in a camera: they condenses a large area (such as a landscape) into a small one (on film or a CCD sensor). For reasons that will be apparent soon, we would like to make the PIR lenses small and thin and moldable from cheap plastic, even though it may add distortion. For this reason the sensors are actually [Fresnel lenses](http://en.wikipedia.org/wiki/Fresnel_lens):  [IMG_259](https://learn.adafruit.com/assets/515)  [Image from Sensors Magazine](http://www.sensorsmag.com/articles/0403/35/main.shtml)  The Fresnel lens condenses light, providing a larger range of IR to the sensor.  [IMG_260](https://learn.adafruit.com/assets/516)  [Image from BHlens.com](http://www.bhlens.com/linear_fresnel_lens.aspx)  [IMG_261](https://learn.adafruit.com/assets/517)  [Image from Cypress appnote 2105](http://learn.adafruit.com/system/assets/assets/000/010/138/original/an2105.pdf)  OK, so now we have a much larger range. However, remember that we actually have two sensors, and more importantly we dont want two really big sensing-area rectangles, but rather a scattering of multiple small areas. So what we do is split up the lens into multiple section, each section of which is a fresnel lens.  **Interfacing PIR Sensor to Arduino –Interfacing PIR Sensor to Arduino – Circuit Diagram**  IMG_256   1. **RGB LED :**   The RGB LED can emit different colors by mixing the 3 basic colors red, green and blue. So it actually consists of 3 separate LEDs red, green and blue packed in a single case. That’s why it has 4 leads, one lead for each of the 3 colors and one common cathode or anode depending of the RGB LED type. In this tutorial I will be using a common cathode one.  IMG_256    **UML DIAGRAM:**  IMG_256 |
| **IMPLEMENTATION CODE:**  int8\_t answer;  int onModulePin= 2;  char aux\_str[129];  //Write here you server and account data  const char smtp\_server[ ] = "[mail.smtp2go.com](http://mail.smtp2go.com/" \t "https://mail.google.com/mail/u/0/" \l "inbox/_blank)";      // SMTP server  const char smtp\_user\_name[ ] = "[smartdoorbell1141@gmail.com](mailto:smartdoorbell1141@gmail.com" \t "https://mail.google.com/mail/u/0/" \l "inbox/_blank)";   // SMTP user name  const char smtp\_password[ ] = "lasthurrah1141";    // SMTP password  const char smtp\_port[ ] = "2525";              // SMTP server port  //Change here your data  const char pin[]="\*\*";  const char phone\_number[]="9550016474";                 //phone number to which message or voice call is to be originated to  const char phone\_number1[]="9550016474";                 //phone number to which message or voice call is to be originated to  const char sms\_text[]="Hey ! Please let me know if you recieve a message or a call.";     //message to be sent as the short message.  //Write here you SIM card data  const char pin\_number[] = "0000";  const char apn[] = "Internet";  const char user\_name[] = "airtel";  const char password[] = "40449";  //Write here your information about sender, direcctions and names  const char sender\_address[ ] = "[smartdoorbell1141@gmail.com](mailto:smartdoorbell1141@gmail.com" \t "https://mail.google.com/mail/u/0/" \l "inbox/_blank)";    // Sender address  const char sender\_name[ ] = "Arduino Project";       // Sender name  const char to\_address[ ] = "[kabirvarsha15@gmail.com](mailto:kabirvarsha15@gmail.com" \t "https://mail.google.com/mail/u/0/" \l "inbox/_blank)";        // Recipient address  const char to\_name[ ] = "Varsha";           // Recipient name  //Write here the subject and body of the email  char subject[ ] = "INTRUDER ALERT";  const char body[ ] = "Hey ! Please let me know if you recieve a message or a call.";   //message to be sent as email.  //Variables initialization and decleration  int PIRpin = 3;  int blueled = 4;  int greenled = 5;  int redled = 6;  int Buzzerpin = 9;  void setup(){    pinMode(onModulePin, OUTPUT);    Serial.begin(115200);     // make the PIRpin's pin an input:    pinMode(PIRpin, INPUT);    pinMode(blueled, OUTPUT);    pinMode(greenled, OUTPUT);    pinMode(redled, OUTPUT);    pinMode(Buzzerpin, OUTPUT);    }  void loop(){      delay(1000);      digitalWrite(blueled, HIGH);      digitalWrite(greenled, LOW);      digitalWrite(redled, LOW);      digitalWrite(Buzzerpin, LOW);      // read the input pin:      int PIRstate = digitalRead(PIRpin);      // print out the state of the button:      Serial.println(PIRstate);      delay(100);      digitalWrite(blueled, LOW);        while(PIRstate == HIGH){       int PIRstate = digitalRead(PIRpin);      // print out the state of the button:      Serial.println(PIRstate);      delay(200);      digitalWrite(blueled, LOW);      digitalWrite(greenled, LOW);      digitalWrite(redled, HIGH);      digitalWrite(Buzzerpin, HIGH);      sendsms();      delay(10000);      sendsms1();      delay(10000);      makevoicecall();      delay(10000);      makevoicecall1();      delay(10000);      sendemail();      delay(20000);      PIRstate = digitalRead(PIRpin);      Serial.println(PIRstate);      delay(200);      Serial.println(PIRstate);      delay(200);        if(PIRstate == LOW){      digitalWrite(redled, LOW);      digitalWrite(Buzzerpin, LOW);      break;      }    }    }  ///////////////////////////////////////////// FUNCTION USED /////////////////////////////////////////  void power\_on(){    uint8\_t answer=0;    // checks if the module is started    answer = sendATcommand("AT", "OK", 2000);    if (answer == 0)    {      // power on pulse      digitalWrite(onModulePin,HIGH);      delay(3000);      digitalWrite(onModulePin,LOW);      // waits for an answer from the module      while(answer == 0){     // Send AT every two seconds and wait for the answer        answer = sendATcommand("AT", "OK", 2000);      }    }  }  int8\_t sendATcommand(char\* ATcommand, char\* expected\_answer, unsigned int timeout){    uint8\_t x=0,  answer=0;    char response[100];    unsigned long previous;    memset(response, '\0', 100);    // Initialice the string    delay(100);    while( Serial.available() > 0) Serial.read();    // Clean the input buffer    Serial.println(ATcommand);    // Send the AT command      x = 0;    previous = millis();    // this loop waits for the answer    do{      // if there are data in the UART input buffer, reads it and checks for the answer      if(Serial.available() != 0){        response[x] = Serial.read();        x++;        // check if the desired answer is in the response of the module        if (strstr(response, expected\_answer) != NULL)        {          answer = 1;        }      }      // Waits for the answer with time out    }    while((answer == 0) && ((millis() - previous) < timeout));    return answer;  }    void sendsms(){      Serial.println("Starting Shield to send sms...");    power\_on();    delay(3000);    //sets the PIN code    sprintf(aux\_str, "AT+CPIN=%s", pin);    sendATcommand(aux\_str, "OK", 2000);    delay(3000);    Serial.println("Connecting to the network...");    while( (sendATcommand("AT+CREG?", "+CREG: 0,1", 500) ||      sendATcommand("AT+CREG?", "+CREG: 0,5", 500)) == 0 );    Serial.print("Setting SMS mode...");    digitalWrite(greenled, HIGH);    digitalWrite(redled, LOW);    sendATcommand("AT+CMGF=1", "OK", 1000);    // sets the SMS mode to text    Serial.println("Sending SMS");      sprintf(aux\_str,"AT+CMGS=\"%s\"", phone\_number);    answer = sendATcommand(aux\_str, ">", 2000);    // send the SMS number    if (answer == 1)    {      Serial.println(sms\_text);      Serial.write(0x1A);      answer = sendATcommand("", "OK", 20000);      if (answer == 1)      {        Serial.print("Sent ");        digitalWrite(greenled, LOW);        digitalWrite(redled, HIGH);      }      else      {        Serial.print("error ");      }    }    else    {      Serial.print("error ");      Serial.println(answer, DEC);    }  }  void sendsms1(){      Serial.println("Starting Shield to send sms...");    power\_on();    delay(3000);    //sets the PIN code    sprintf(aux\_str, "AT+CPIN=%s", pin);    sendATcommand(aux\_str, "OK", 2000);    delay(3000);    Serial.println("Connecting to the network...");    while( (sendATcommand("AT+CREG?", "+CREG: 0,1", 500) ||      sendATcommand("AT+CREG?", "+CREG: 0,5", 500)) == 0 );    Serial.print("Setting SMS mode...");    digitalWrite(greenled, HIGH);    digitalWrite(redled, LOW);    sendATcommand("AT+CMGF=1", "OK", 1000);    // sets the SMS mode to text    Serial.println("Sending SMS");      sprintf(aux\_str,"AT+CMGS=\"%s\"", phone\_number1);    answer = sendATcommand(aux\_str, ">", 2000);    // send the SMS number    if (answer == 1)    {      Serial.println(sms\_text);      Serial.write(0x1A);      answer = sendATcommand("", "OK", 20000);      if (answer == 1)      {        Serial.print("Sent ");        digitalWrite(greenled, LOW);        digitalWrite(redled, HIGH);      }      else      {        Serial.print("error ");      }    }    else    {      Serial.print("error ");      Serial.println(answer, DEC);    }  }   void makevoicecall(){         Serial.println("Starting Shield to make a voice call...");    power\_on();    delay(3000);    //sets the PIN code    sprintf(aux\_str, "AT+CPIN=%s", pin);    sendATcommand(aux\_str, "OK", 2000);    delay(3000);    Serial.println("Connecting to the network...");    //Enables the use of command ATH    sendATcommand("AT+CVHU=0", "OK", 10000);    while ( (sendATcommand("AT+CREG?", "+CREG: 0,1", 500) ||             sendATcommand("AT+CREG?", "+CREG: 0,5", 500)) == 0 );      Serial.print("Calling to ");    digitalWrite(greenled, HIGH);    digitalWrite(redled, LOW);    Serial.print(phone\_number);    //Make the phone call    sprintf(aux\_str, "ATD%s;", phone\_number);    sendATcommand(aux\_str, "OK", 10000);    delay(25000);  //after this time elapse .....    // disconnects the existing call    digitalWrite(greenled, LOW);    digitalWrite(redled, HIGH);    Serial.println("ATH");    Serial.println("Call disconnected");   }  void makevoicecall1(){         Serial.println("Starting Shield to make a voice call...");    power\_on();    delay(3000);    //sets the PIN code    sprintf(aux\_str, "AT+CPIN=%s", pin);    sendATcommand(aux\_str, "OK", 2000);    delay(3000);    Serial.println("Connecting to the network...");    //Enables the use of command ATH    sendATcommand("AT+CVHU=0", "OK", 10000);    while ( (sendATcommand("AT+CREG?", "+CREG: 0,1", 500) ||             sendATcommand("AT+CREG?", "+CREG: 0,5", 500)) == 0 );      Serial.print("Calling to ");    digitalWrite(greenled, HIGH);    digitalWrite(redled, LOW);    Serial.print(phone\_number);    //Make the phone call    sprintf(aux\_str, "ATD%s;", phone\_number1);    sendATcommand(aux\_str, "OK", 10000);    delay(25000);  //after this time elapse .....    // disconnects the existing call    digitalWrite(greenled, LOW);    digitalWrite(redled, HIGH);    Serial.println("ATH");    Serial.println("Call disconnected");   }       void sendemail(){         Serial.println("Starting Shield to send an email...");    power\_on();    delay(3000);    //sets the PIN code    digitalWrite(greenled, HIGH);    digitalWrite(redled, LOW);    sprintf(aux\_str, "AT+CPIN=%s", pin\_number);    sendATcommand(aux\_str, "OK", 2000);    delay(3000);    while( (sendATcommand("AT+CREG?", "+CREG: 0,1", 500) ||      sendATcommand("AT+CREG?", "+CREG: 0,5", 500)) == 0 );    // sets the SMTP server and port    sprintf(aux\_str, "AT+SMTPSRV=\"%s\",%s", smtp\_server, smtp\_port);    sendATcommand(aux\_str, "OK", 2000);    // sets user name and password    sprintf(aux\_str, "AT+SMTPAUTH=1,\"%s\",\"%s\"", smtp\_user\_name, smtp\_password);    sendATcommand(aux\_str, "OK", 2000);    // sets sender adress and name    sprintf(aux\_str, "AT+SMTPFROM=\"%s\",\"%s\"", sender\_address, sender\_name);    sendATcommand(aux\_str, "OK", 2000);    // sets sender adress and name    sprintf(aux\_str, "AT+SMTPRCPT=1,0,\"%s\",\"%s\"", to\_address, to\_name);    sendATcommand(aux\_str, "OK", 2000);    // subjet of the email    sprintf(aux\_str, "AT+SMTPSUB=\"%s\"", subject);    sendATcommand(aux\_str, "OK", 2000);    // body of the email    sprintf(aux\_str, "AT+SMTPBODY=\"%s\"", body);    sendATcommand(aux\_str, "OK", 2000);    // sets APN, user name and password    sprintf(aux\_str, "AT+CGSOCKCONT=1,\"IP\",\"%s\"", apn);    sendATcommand(aux\_str, "OK", 2000);    sprintf(aux\_str, "AT+CSOCKAUTH=1,1,\"%s\",\"%s\"", user\_name, password);    sendATcommand(aux\_str, "OK", 2000);    delay(2000);    Serial.println("Sending email...");    // sends the email and waits the answer of the module    answer = sendATcommand("AT+SMTPSEND", "+SMTP: SUCCESS", 60000);    if (answer == 1)    {      digitalWrite(greenled, LOW);       digitalWrite(redled, HIGH);      Serial.println("Done!");    }    else    {      Serial.println("Error");    }   }  **GITHUB Links:**  <https://github.com/Devikadampalli/wtlab--311>  <https://github.com/Devikadampalli/wtlab--311>  **References:**  [https://app.smtp2go.com/settings/ipauth/](https://app.smtp2go.com/settings/ipauth/" \t "https://mail.google.com/mail/u/0/" \l "inbox/_blank)  [https://www.arduino.cc/en/Guide/ArduinoGSMShield](https://www.arduino.cc/en/Guide/ArduinoGSMShield" \t "https://mail.google.com/mail/u/0/" \l "inbox/_blank)  [https://create.arduino.cc/projecthub/eani/diy-how-to-use-the-arduino-uno-to-send-an-email-or-sms-28ac4d](https://create.arduino.cc/projecthub/eani/diy-how-to-use-the-arduino-uno-to-send-an-email-or-sms-28ac4d" \t "https://mail.google.com/mail/u/0/" \l "inbox/_blank)  [https://mechatrofice.com/arduino/gsm-send-sms](https://mechatrofice.com/arduino/gsm-send-sms" \t "https://mail.google.com/mail/u/0/" \l "inbox/_blank)  [https://randomnerdtutorials.com/arduino-with-pir-motion-sensor/](https://randomnerdtutorials.com/arduino-with-pir-motion-sensor/" \t "https://mail.google.com/mail/u/0/" \l "inbox/_blank) |