Chapter 1

INTRODUCTION

A **smoke detector** is a device that senses smoke, typically as an indicator of fire. Commercial security devices issue a signal to a fire alarm control panel as part of a fire alarm system, while household smoke detectors, also known as **smoke alarms**, generally issue a local audible or visual alarm from the detector itself.

This Smoke detection alarm will not only indicate the fire by buzzing the buzzer but also send an SMS alert to the REGISTERED OWNER to alert that SMOKE has been detected by the Smoke detector.

This Smoke Detector will not only detect SMOKE but will also detect PROPHANE, HYDROGEN, ALCOHOL, METHANE, CARBON MONOXIDE, so any type of GAS leakage will also get detected and will trigger the SMOKE DETECTOR.

OVERVIEW

A smoke detector is a device that senses smoke, typically as an INDICATION of fire. Commercial security devices issue a signal to an afire alarm control panel as part of a fire alarm system, while household smoke detectors, also known as smoke alarms, generally issue a local audible alarm from detector itself.

This Smoke detection alarm will not only indicate the fire by buzzing the buzzer but also send an SMS alert to the REGISTERED OWNER to alert that SMOKE has been detected by the Smoke detector.

In this project, mq2 gas sensor is used. An **ionization smoke detector** uses a radioisotope, typically americium-241, to ionize air; a difference due to smoke is detected and an alarm is generated. Ionization detectors are more sensitive to the flaming stage of fires than optical detectors, while optical detectors are more sensitive to fires in the early smoldering stage.

Arduino is the main PROCESSOR of this Project. Using Arduino and GSM module SIM 900A the Smoke Detector senses the smoke and sends the ALERT MESSAGE to the REGISTERED USER i.e. using the GSM MODULE SIM 900A.

The GSM MODULE uses a SIM CARD to send the alert message to the REGISTED USER. The GSM MODULE also requires a 12 V – 2A power supply to work.

PROBLEM DEFINITION

A smoke detector is a device that senses smoke, typically as an indicator of fire. Commercial security devices issue a signal to a fire alarm control panel as part of a fire alarm system, while household smoke detectors, also known as smoke alarms, generally issue a local audible or visual alarm from the detector itself.

In case of fire or Gas leakages, there must be alert for everyone to aware that smoke or gas leakage has been detected at the exact same time.

One of the aims of the building act is to improve household fire safety. For this reason, the building act requires automatic smoke detection in areas that people sleep.

SCOPE

The area of utility determines the scope of smoke detectors. A small area, like a household, does not need many smoke detectors and the setting off of alarms can be easily identified and mitigated. Now, consider the case of a corporate building which has many floors. One cannot simply know the location of any alarm going off on some floor while sitting in ground floor. It is evident that a large number of smoke detectors cannot be allowed to perform independently and need some sort of central monitoring.

Chapter 2

COMPONENTS

There are basically 6 major components in this SMOKE DETECTOR, that are

ARDUINO UNO R3
GSM MODULE SIM 900A
MQ 2 GAS SENSOR
BUZZER
LED’s (Green and Red)
BREADBOARD
POWER ADAPTER

1) ARDUINO UNO R3

The Arduino Uno R3 is a microcontroller board based on a removable, dual-inline-package (DIP) ATmega328 AVR microcontroller. It has 20 digital input/output pins (of which 6 can be used as PWM outputs and 6 can be used as analog inputs). Programs can be loaded on to it from the easy-to-use Arduino computer program. The Arduino has an extensive support community, which makes it a very easy way to get started working with embedded electronics. The R3 is the third, and latest, revision of the Arduino Uno.

The Arduino Uno is a microcontroller board based on the ATmega328. It has 20 digital input/output pins (of which 6 can be used as PWM outputs and 6 can be used as analog inputs), a 16 MHz resonator, a USB connection, a power jack, an in-circuit system programming (ICSP) header, and a reset button. It contains everything needed to support the microcontroller; simply connect it to a computer (or appropriate wall power adapter) with a USB cable or power it with an AC-to-DC adapter or battery to get started.

The Uno differs from all preceding boards in that it does not use the FTDI USB-to-serial driver chip. Instead, it features an ATmega16U2 programmed as a USB-to-serial converter. This auxiliary microcontroller has its own USB bootloader, which allows advanced users to reprogram it.

Microcontroller	ATmega328P
Operating Voltage	5V
Input Voltage (recommended)	7-12V
Input Voltage (limit)	6-20V
Digital I/O Pins	14 (of which 6 provide PWM output)
PWM Digital I/O Pins	6
Analog Input Pins	6
DC Current per I/O Pin	20 mA
DC Current for 3.3V Pin	50 mA
Flash Memory	32 KB (ATmega328P) of which 0.5 KB used by bootloader
SRAM	2 KB (ATmega328P)
EEPROM	1 KB (ATmega328P)
Clock Speed	16 MHz
LED_BUILTIN	13
Length	68.6 mm
Width	53.4 mm

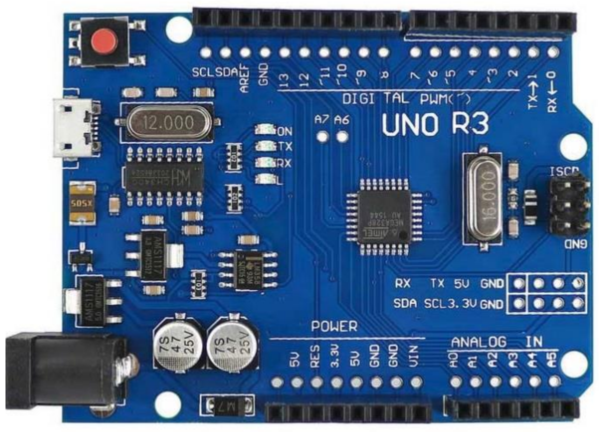


IMAGE OF ARDUINO UNO R3

2) GSM MODULE SIM 900A

OVERVIEW

GSM/GPRS Modem-RS232 is built with Dual-Band GSM/GPRS engine- SIM900A works on frequencies 900/ 1800 MHz The Modem is coming with RS232 interface, which allows you connect PC as well as a microcontroller with RS232 Chip (MAX232). The baud rate is configurable from 9600-115200 through AT command. The **GSM/GPRS Modem** is having internal TCP/IP stack to enable you to connect with internet via GPRS. It is suitable for SMS, Voice as well as DATA transfer application in M2M interface. The onboard Regulated Power supply allows you to connect wide range unregulated power supply. Using this modem, you can make audio calls, SMS, Read SMS, attend the incoming calls and internet etc. through simple AT commands.

FEATURES

- Quad-Band 850/ 900/ 1800/ 1900 MHz
- Dual-Band 900/ 1900 MHz
- GPRS multi-slot class 10/8GPRS mobile station class B
- Compliant to GSM phase 2/2+Class 4 (2 W @850/ 900 MHz)
- Class 1 (1 W @ 1800/1900MHz)
- Control via AT commands (GSM 07.07 ,07.05 and SIMCOM enhanced AT Commands)
- Low power consumption: 1.5mA (sleep mode)
- Operation temperature: -40°C to +85 °C

IMAGE OF GSM MODULE SIM900A

3) MQ-2 GAS SENSOR

MQ2 gas sensor can be used to detect the presence of LPG, Propane and Hydrogen, also could be used to detect Methane and other combustible steam, it is with low cost and suitable for different application. Sensor is sensitive to flammable gas and smoke. Smoke sensor is given 5 volts to power it. Smoke sensor indicate smoke by the voltage that it outputs. More smoke more output. A potentiometer is provided to adjust the sensitivity. Sn02 is the sensor used which is of low conductivity when the air is clean. But when smoke exist, an analog output is produced based on the concentration of smoke. The circuit has a heater. Power is given to heater by VCC and GND from power supply. The circuit has a variable resistor. The resistance across the pin depends on the smoke in air in the sensor. The resistance will be lowered if the content is more. And voltage is increased between the sensor and load resistor.

Specifications

- Wide range sensitivity to combustible gas.
- Better sensitivity to Propane, LPG and Hydrogen
- Low cost and better life
- Drive circuit is simple.
- Sensor Type: Semiconductor
- Concentration: 300-10000ppm (Combustible gas)
- Supply voltage =5v

Working with MQ2 Gas Sensor. The MQ-2 Gas Sensor module detects gas leakage in home and industry. The MQ series of gas sensors use a small heater inside with an electrochemical sensor. They are sensitive to a range of gasses and are used indoors at room temperature

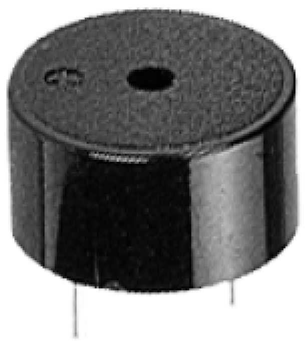
IMAGES OF MQ-2 SENSOR

4) BUZZER

A buzzer or beeper is an audio signaling device, which may be mechanical, electromechanical, or piezoelectric (*piezo* for short). Typical uses of buzzers and beepers include alarm devices,

When the mq 2 sensor senses the smoke then the RED LED will glow and along with that to alert people surrounded around the SMOKE DETECTOR buzzer is used

IMAGE OF BUZZER



5) LED's

Two different types of LED's are used. One is GREEN COLORED LED which will glow when NO SMOKE is detected.

The second LED which is RED COLOURED LED, which will glow when SMOKE is detected by the Sensor. Along with the RED LED the buzzer will also get buzzed.

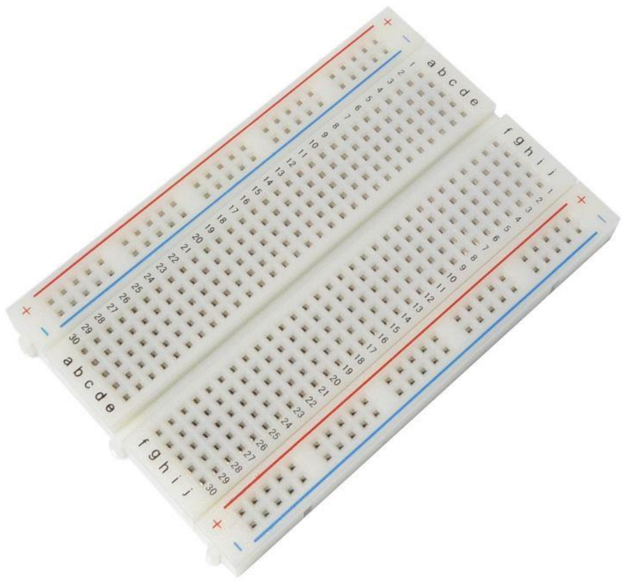
IMAGE OF LED

6) BREADBOARD

An electronics breadboard (as opposed to the type on which sandwiches are made) is actually referring to a solderless breadboard. These are great units for making temporary circuits and prototyping, and they require absolutely no soldering.

All the components of this project are connected with each other using BREADBOARD.

IMAGE OF BREADBOARD



7) POWER SUPPLY

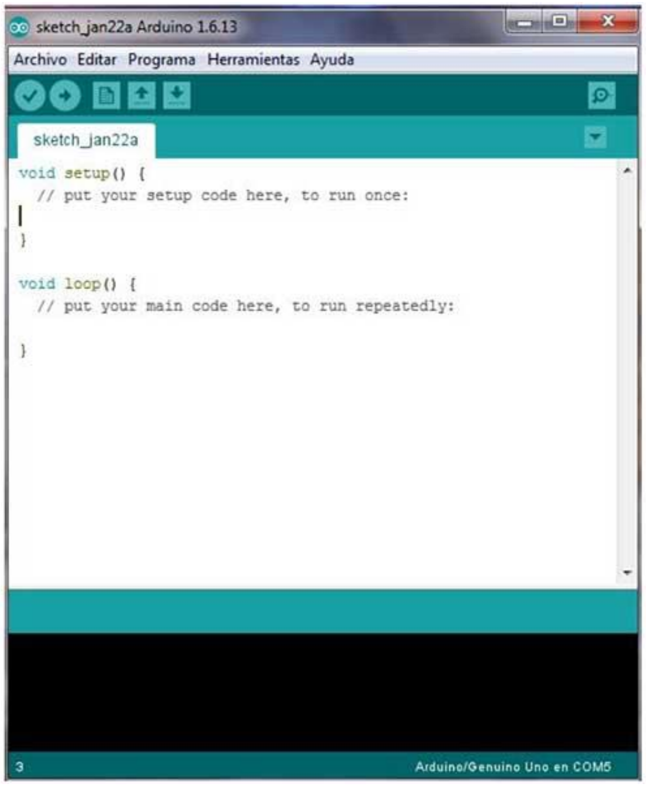
To use the GSM MODULE SIM 900A, there must a POWER SUPPLY which should power the GSM module.

To work the GSM module there is a recommended power INPUT given which is 12 V and 2A

So, the power adapter must have an OUTPUT of 12V – 2A

8) ARDUINO SOFTWARE

The Arduino integrated development environment is a cross-platform application that is written in the programming language Java. It is used to write and upload programs to Arduino compatible boards, but also, with the help of 3rd party cores, other vendor development boards.


sketch_jan22a Arduino 1.6.13
Archivo Editar Programa Herramientas Ayuda
sketch_jan22a
3
Arduino/Genuino Uno en COM5

```
void setup() {
  // put your setup code here, to run once:

}

void loop() {
  // put your main code here, to run repeatedly:

}
```

Chapter 3

DESIGN

DESIGN

Design or the Architecture of the Smoke Detector is simple but a little complicated. The connections of the PINS of the ARDUINO UNO, GSM MODULE SIM900A, MQ-2 SENSOR, BUZZER and LED's (green and red)

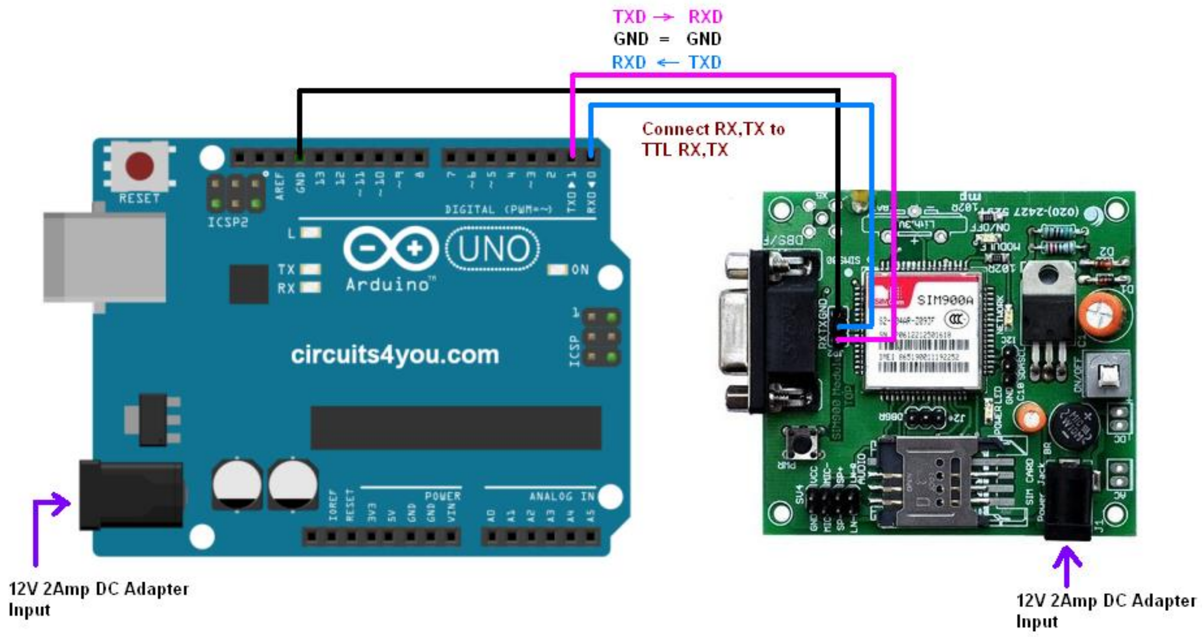
The connections are as Follows

Connect rx of GSM module to 9 Arduino ,tx to 10of Arduino and gnd to gnd

Connect red led to digital pin 13 of arduino , greenled to pin no 11

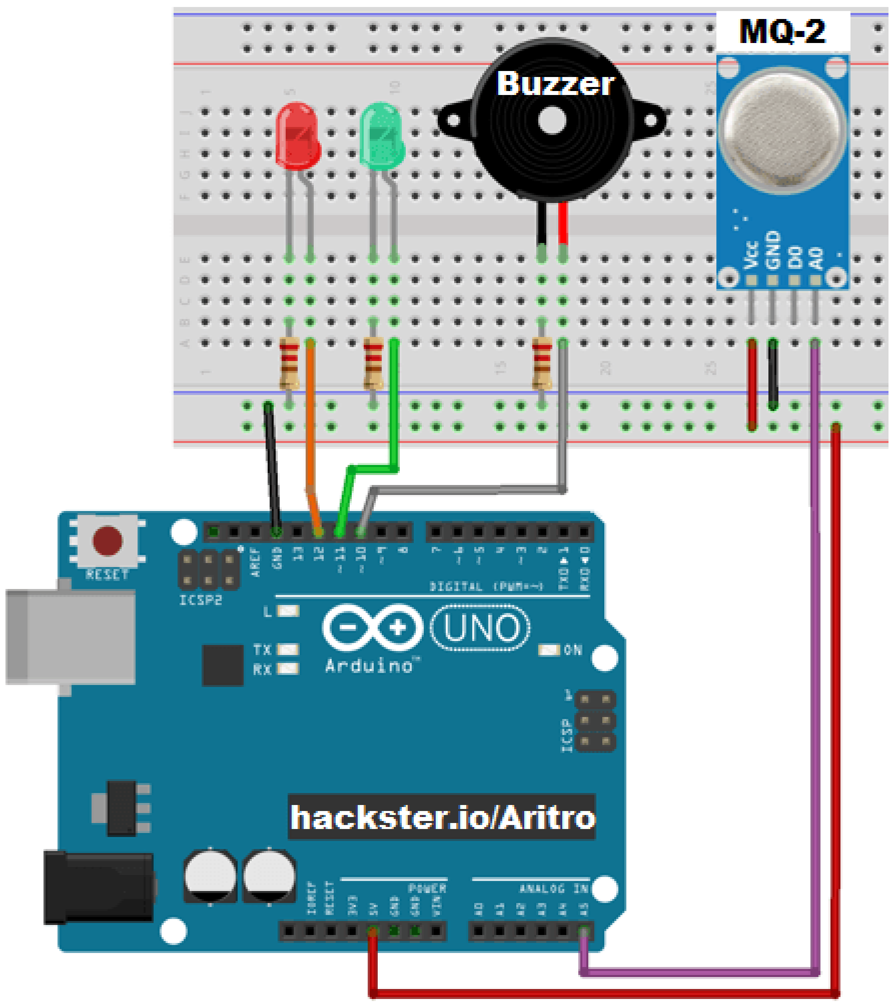
Connect buzzer to pin no 8

and input mq2 sensor to A0 of Arduino



CONNECTING ARDUINO UNO WITH GSM MODULE

CONNECTING THE MQ-2 SENSOR, BUZZER AND LED's WITH ARDUINO



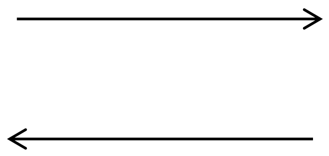
Data flow diagrams

A data flow diagram (DFD) is a graphical representation of the 'flow' of data through an information system, modeling its process aspects. Often, they are a preliminary step used to create an overview of the system which can later be elaborated. DFDs can also be used for the visualization of data processing.

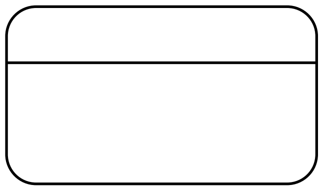
DFD Symbols:

1) A **SQUARE** defines a source or destination of system data.

2) An **ARROW** identifies data flow or data in action in motion. It is a pipeline through which information flows.

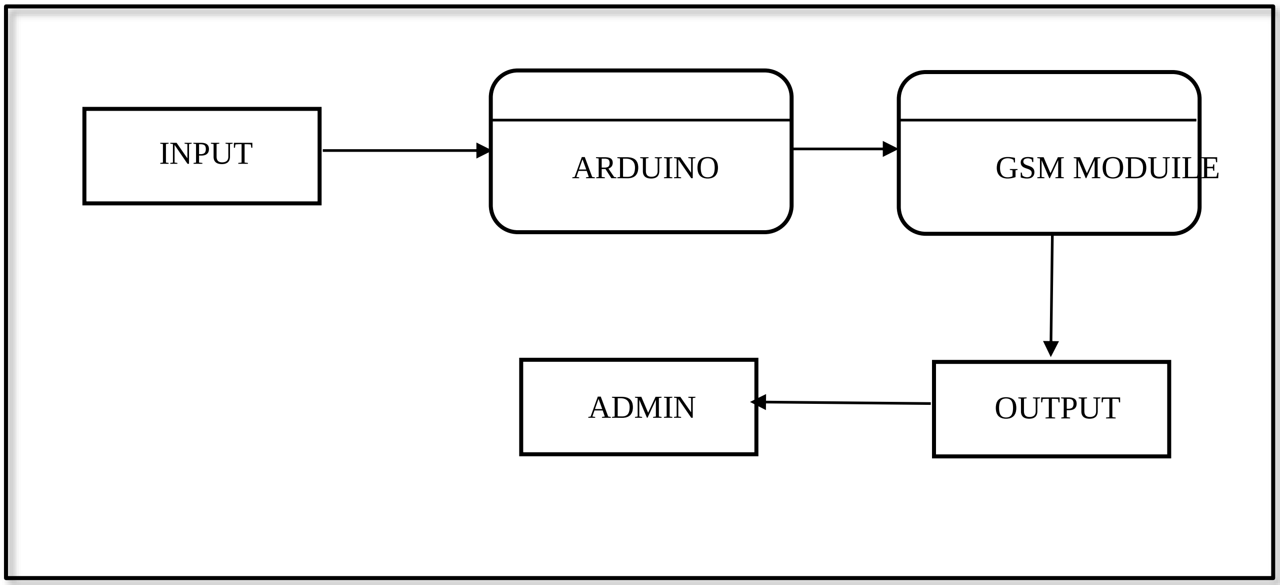


3) A **ROUNDED RECTANGLE** represents a process transforms in coming data flow into outgoing dataflows.



4) An **OPEN RECTANGLE** is a data store or data at rest or a temporary rest repository of data.

DATA FLOW DIAGRAM



DATA FLOW DIAGRAM for SMOKE DETECTOR

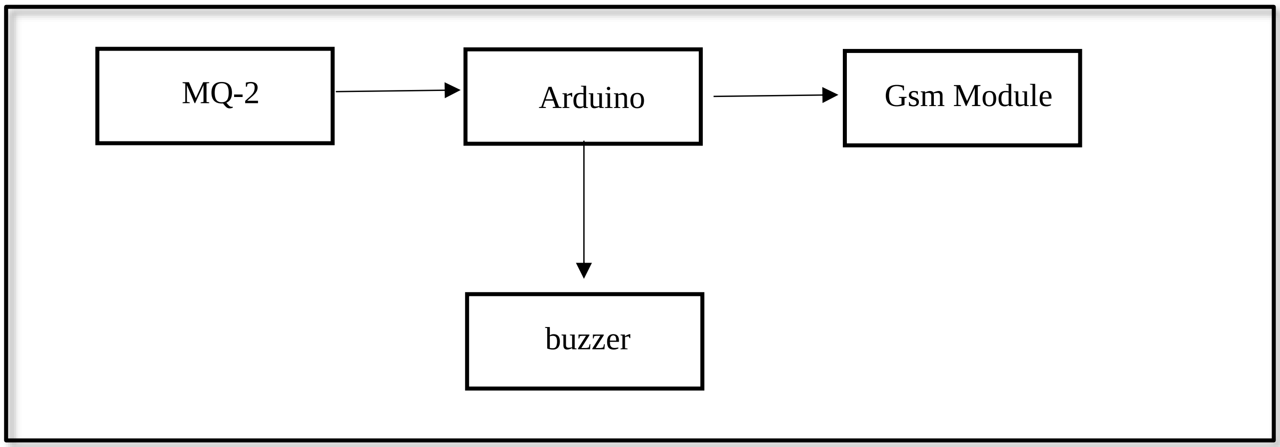
UML DIAGRAMS

The Unified Modeling Language (UML) is a general-purpose, developmental, modeling language in the field of software engineering, that is intended to provide a standard way to visualize the design of a system. The Unified Modeling Language (UML) was created to forge a common, semantically and syntactically rich visual modeling language for the architecture, design, and implementation of complex software systems both structurally and behaviorally. UML has applications beyond software development, such as process flow in manufacturing. UML is not a programming language but there are tools that can be used to generate code in various languages using UML diagrams. UML has a direct relation with object-oriented analysis and design.

OBJECT DIAGRAM

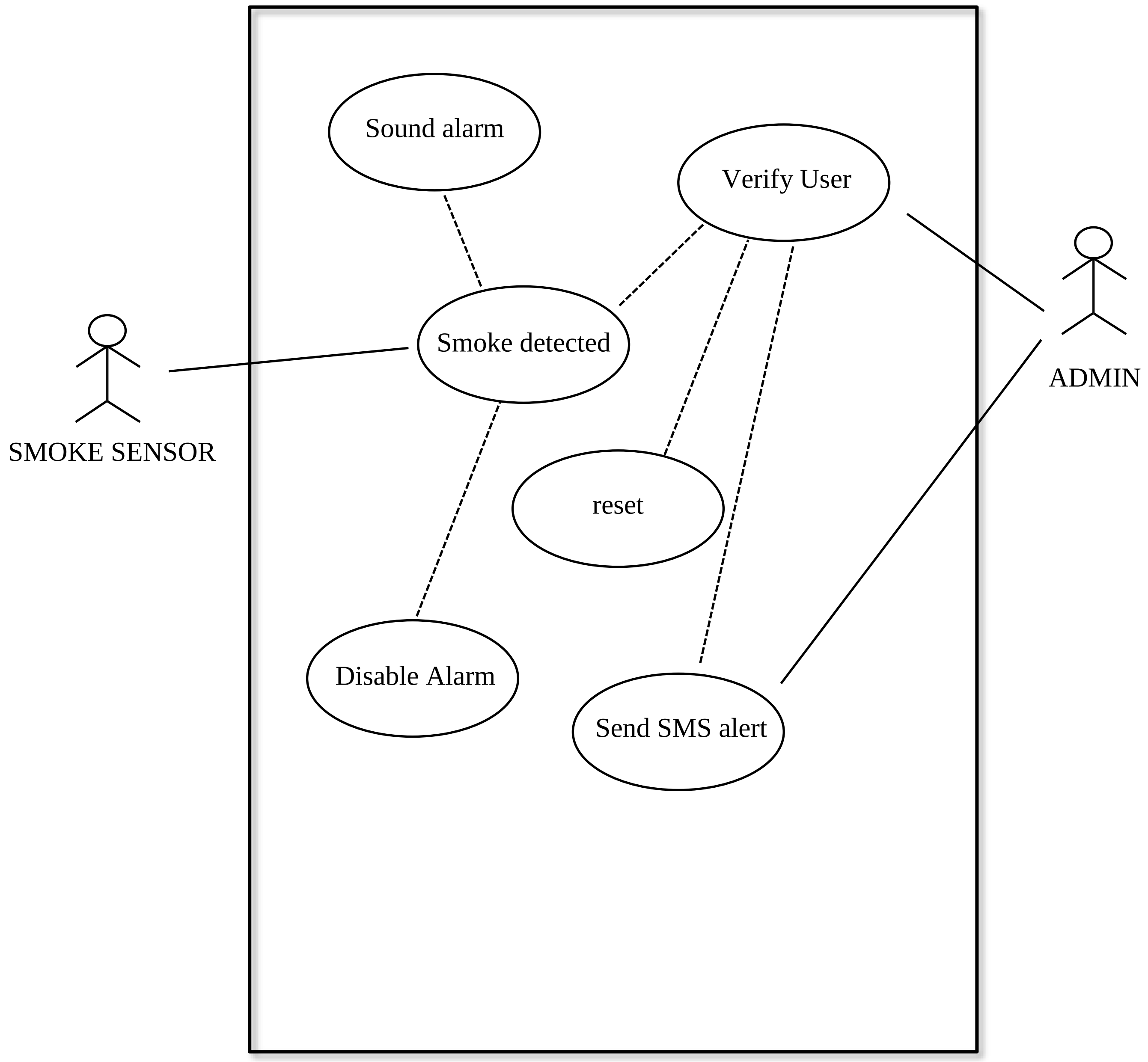
Object diagrams are also closely linked to class diagrams. Just as an object is an instance of a class, an object diagram could be viewed as an instance of a class diagram. Object diagrams describe the static structure of a system at a particular time. Object Diagram shows the relationship between objects using real world examples and illustrates how a system will look at any given time. Because data is available within objects, they can be used to clarify relationships between objects. Each object and link on an object diagram are represented by an Instance Specification. This can show an object's classifier (e.g. an abstract or concrete class) and instance name, as well as attributes and other structural features using slots. Each slot corresponds to a single attribute or feature, and may include a value for that entity.

The name on an instance specification optionally shows an instance name, a ':' separator, and optionally one or more classifier names separated by commas. The contents of slots, if any, are included below the names, in a separate attribute compartment. A link is shown as a solid line, and represents an instance of an association



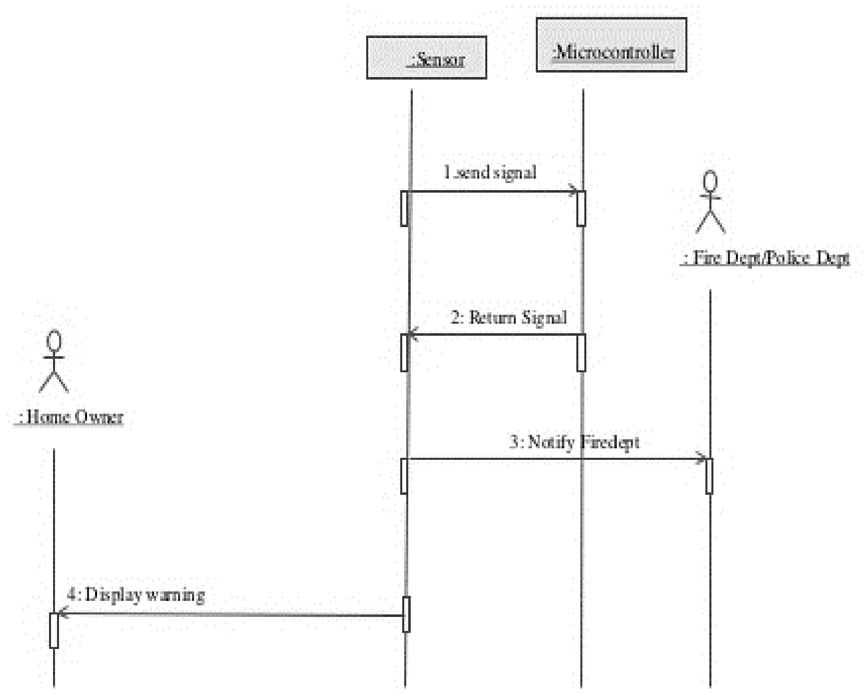
USE CASE DIAGRAM

In software and systems engineering, a use case is a list of actions or event steps typically defining the interactions between a role (known in the Unified Modeling Language (UML) as an actor) and a system to achieve a goal. The actor can be a human or other external system. In systems engineering, use cases are used at a higher level than within software engineering, often representing missions or stakeholder goals.



SEQUENCE DIAGRAM

Sequence diagram simply depicts interaction between objects in a sequential order i.e. the order in which these interactions take place. We can also use the terms event diagrams or event scenarios to refer to a sequence diagram. Sequence diagrams describe how and in what order the objects in a system function. Sequence Diagram Shows how objects interact with each other and the order of occurrence. They represent interactions for a particular scenario.



Chapter 4

CODING

Since ARDUINO UNO R3 is used the program there is a dedicated SOFTWARE available to implement code/Program into the ARDUINO UNO. The Name of the program is ARDUINO itself.

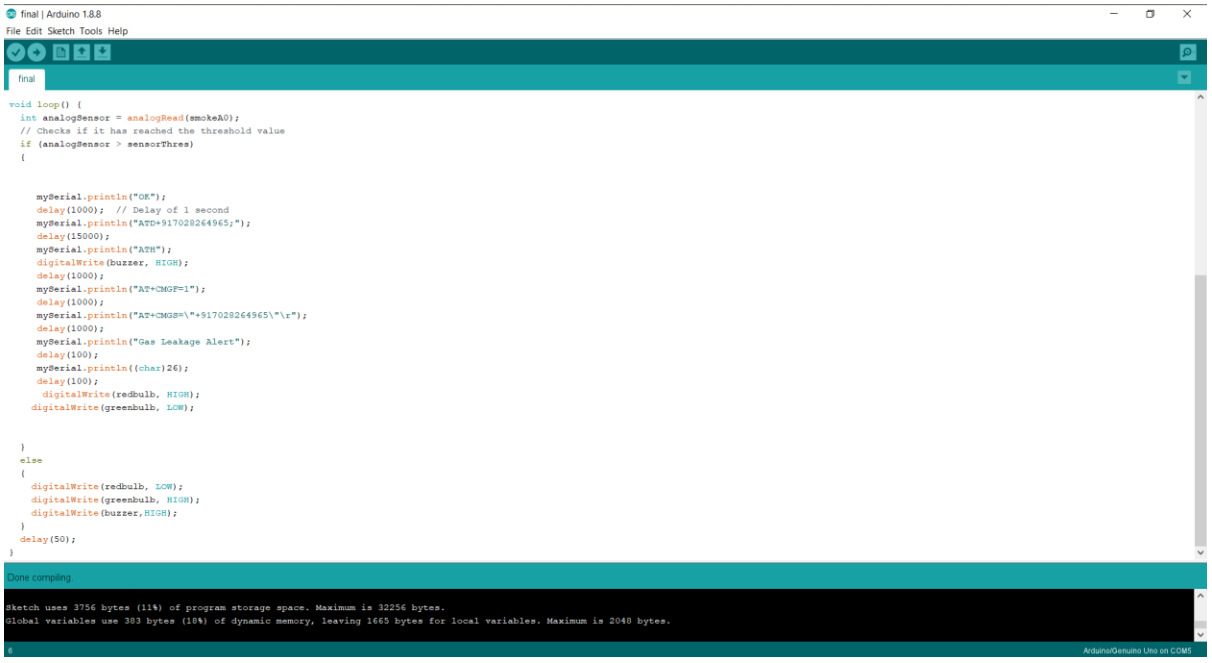
Arduino Software supports C++ Compiler Programming language.

CODE OF SMOKE DETECTOR

```
int redbulb = 13;
int greenbulb = 11;
int buzzer = 8;
int smokeA0 = A5;
// My threshold value
int sensorThres = 90;

  #include <SoftwareSerial.h>

SoftwareSerial mySerial(9, 10);



void setup() {
  pinMode(redbulb, OUTPUT);
  pinMode(greenbulb, OUTPUT);
  pinMode(buzzer, OUTPUT);
  pinMode(smokeA0, INPUT);

  mySerial.begin(9600);
  Serial.begin(9600);
  delay(100);
}
```

```
void loop() {
  int analogSensor = analogRead(smokeA0);
  // Checks if it has reached the threshold value
  if (analogSensor > sensorThres)
  {


     mySerial.println("OK");
     delay(1000);  // Delay of 1 second
     mySerial.println("ATD+91xxxxxxxxxxx;");
     delay(15000);
     mySerial.println("ATH");
     digitalWrite(buzzer, HIGH);
     delay(1000);
     mySerial.println("AT+CMGF=1");
     delay(1000);
     mySerial.println("AT+CMGS=\"+91xxxxxxxxxx\"\r");
     delay(1000);
     mySerial.println("Gas Leakage Alert");
     delay(100);
     mySerial.println((char)26);
     delay(100);
      digitalWrite(redbulb, HIGH);
    digitalWrite(greenbulb, LOW);


  }
  else
  {
    digitalWrite(redbulb, LOW);
    digitalWrite(greenbulb, HIGH);
    digitalWrite(buzzer,HIGH);
  }
  delay(50);
}
```

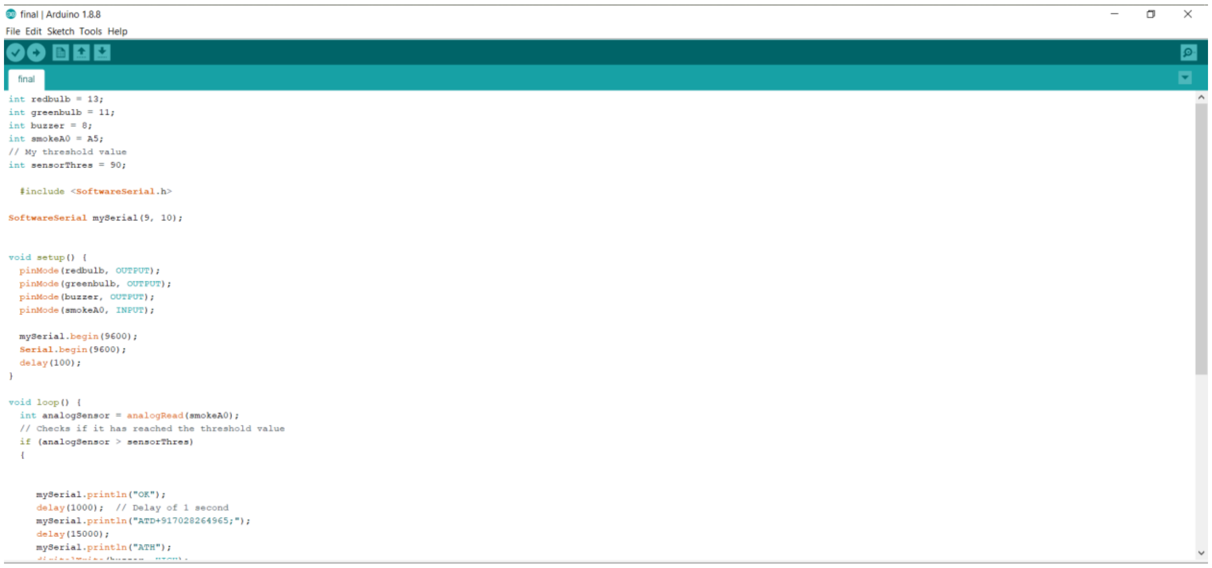
SNAPSHOT OF PROGRAM IN ARDUINO IDE



```
void loop() {
  int analogSensor = analogRead(smokeA0);
  // Checks if it has reached the threshold value
  if (analogSensor > sensorThres)
  {


     mySerial.println("OK");
     delay(1000);  // Delay of 1 second
     mySerial.println("ATD+917028264965;");
     delay(15000);
     mySerial.println("ATH");
     digitalWrite(buzzer, HIGH);
     delay(1000);
     mySerial.println("AT+CMGF=1");
     delay(1000);
     mySerial.println("AT+CMGS=\"+917028264965\"\r");
     delay(1000);
     mySerial.println("Gas Leakage Alert");
     delay(100);
     mySerial.println((char)26);
     delay(100);
      digitalWrite(redbulb, HIGH);
    digitalWrite(greenbulb, LOW);


  }
  else
  {
    digitalWrite(redbulb, LOW);
    digitalWrite(greenbulb, HIGH);
    digitalWrite(buzzer,HIGH);
  }
  delay(50);
}
```

Done compiling.

```
Sketch uses 3756 bytes (11%) of program storage space. Maximum is 32256 bytes.
Global variables use 383 bytes (18%) of dynamic memory, leaving 1665 bytes for local variables. Maximum is 2048 bytes.
```



```
int redbulb = 13;
int greenbulb = 11;
int buzzer = 8;
int smokeA0 = A5;
// My threshold value
int sensorThres = 90;

  #include <SoftwareSerial.h>

SoftwareSerial mySerial(9, 10);


void setup() {
  pinMode(redbulb, OUTPUT);
  pinMode(greenbulb, OUTPUT);
  pinMode(buzzer, OUTPUT);
  pinMode(smokeA0, INPUT);

  mySerial.begin(9600);
  Serial.begin(9600);
  delay(100);
}

void loop() {
  int analogSensor = analogRead(smokeA0);
  // Checks if it has reached the threshold value
  if (analogSensor > sensorThres)
  {


     mySerial.println("OK");
     delay(1000);  // Delay of 1 second
     mySerial.println("ATD+917028264965;");
     delay(15000);
     mySerial.println("ATH");
```

Change Setting of the SMOKE DETECTOR

The sensors sensitivity can be set in the 5th Line of the Program.

int sensorThres = **90**

By changing the value of int sensorThres the sensitivity of the MQ 2 sensor can be adjusted as needed.

To Register mobile number with the program so that the Arduino could send message to the REGISTERED PERSON could be done by adding the mobile number in line 31

mySerial.println("AT+CMGS=\"+91xxxxxxxxxx\"\r");

Just by replacing the values of xxxxxxxxxx with the persons phone number who you want the Arduino board to send the *GAS LEAKAGE ALERT* message.

The message is sent using SMS service

The message *GAS LEAKAGE ALERT* can be also modified at line 33

mySerial.println("Gas Leakage Alert");

By changing the sentence with in double quotation marks, the message sent by the GSM MODULE to the registered person will be changes to the SENTENCE within double quotation marks.

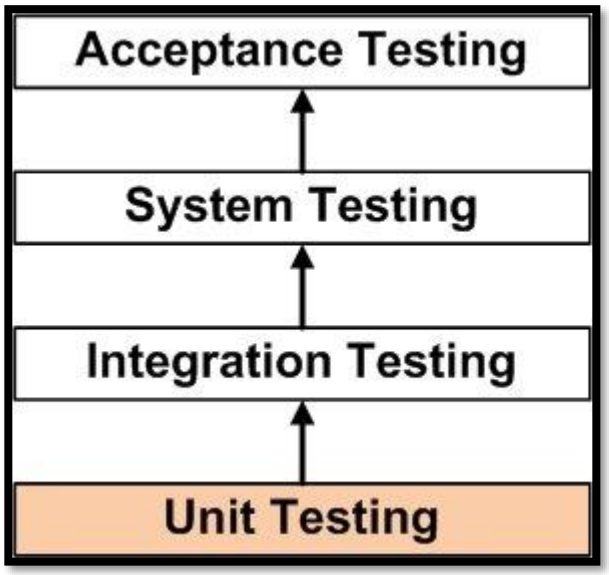
Chapter 5

TESTING

UNIT TESTING

Unit testing is a procedure used to validate that individual units of source code are working properly. A unit is a smallest testable part of an application. Utilizing a set of unit tests can dramatically reduce the number of bugs and the risks with untested code. In our project we have performed unit testing to isolate each part of the program and show that the individual parts of the program work properly. We divided each part into the group members and tested it successfully. The goal of unit testing is to isolate each part of the program and show that the individual parts are correct. A unit test provides a strict, written contract that the piece of code must satisfy. As a result, it affords several benefits.

Unit testing finds problems early in the development cycle. This includes both bugs in the programmer's implementation and flaws or missing parts of the specification for the unit. The process of writing a thorough set of tests forces the author to think through inputs, outputs, and error conditions, and thus more crisply define the unit's desired behavior.



Unit Testing

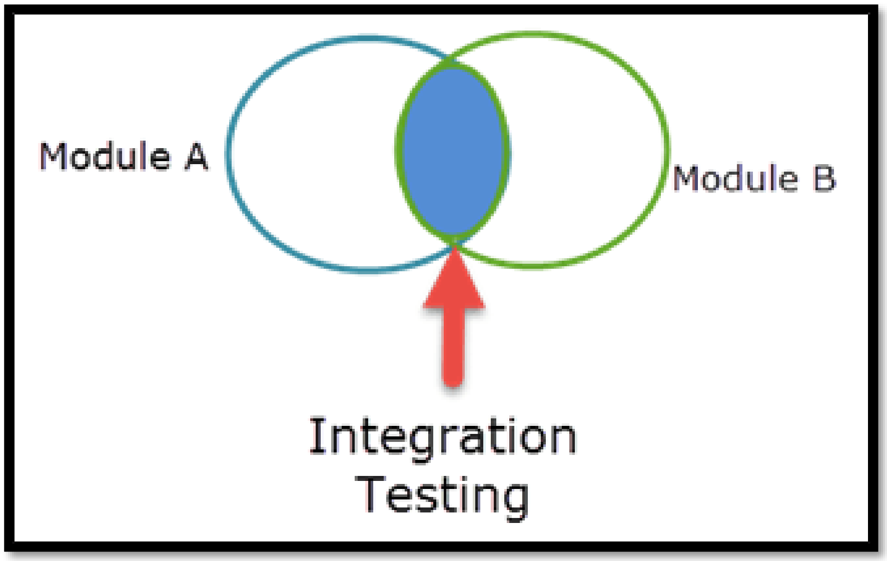
INTEGRATION TESTING

Integration testing is a logical extension of unit testing. In its simplest form, two units that have already been tested are combined into a component and the interface between them is tested. It is that phase of software testing in which individual software modules are combined and tested as a group. In our case the Arduino was tested and the GSM module was tested too. Both the components were working correctly.

But as we connect the ARDUINO with the GSM MODULE the ARDUINO IDE was showing an error with ERROR 7 SYNC.

The error was created because the RX terminal and TX terminal of GSM MODULE was unable to link with the TX and RX of Arduino.

To fix this issue we redirected the RX and TX with PORT 9 and 10 of Arduino, the test was completed and both the components were working correctly.



Integration Testing

SYSTEM TESTING

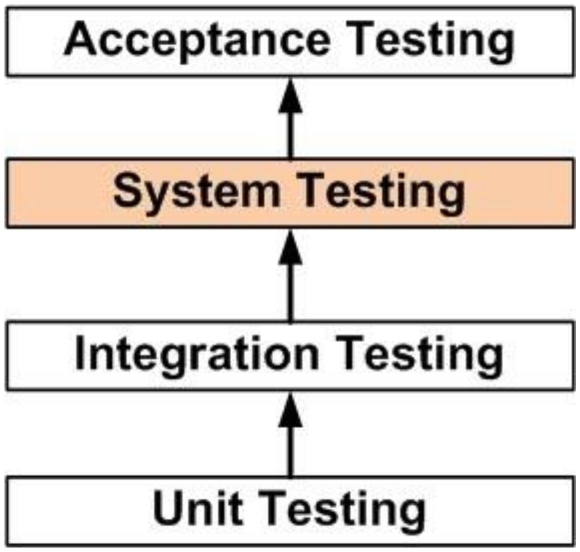
System testing of software or hardware is testing conducted on a complete, integrated system to evaluate the system's compliance with its specified requirements. System testing falls within the scope of black box testing, and as such, should require no knowledge of the inner design of the code or logic. In our case after the code is implemented into the Arduino we have to check whether all the Components are working as they were expected to work. All the components were given different PINS from the ARDUINO.

As soon as the smoke gets detected whether the Arduino is sending positive signals to GSM module and the buzzer with the RED LED to glow as soon as the SMOKE is detected.

This all things were tested in System Testing.

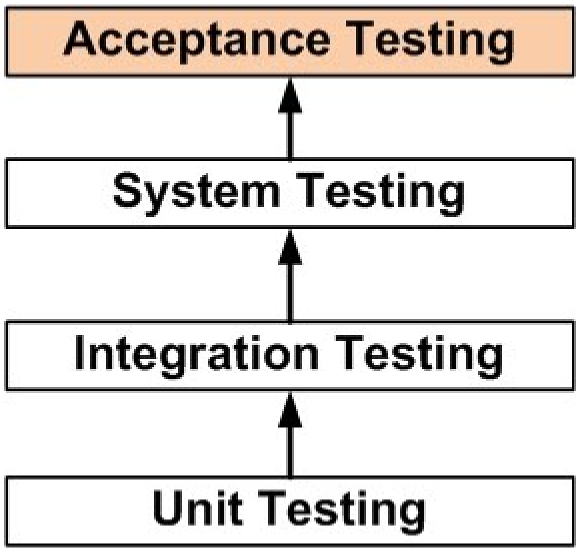
Additionally, each individual type of system test reports relevant metrics of a piece of software, including:

- Performance testing: speed, average, stability and peak response times;
- Load testing: delay of the sensor, buzzing delay



ACCEPTANCE TESTING

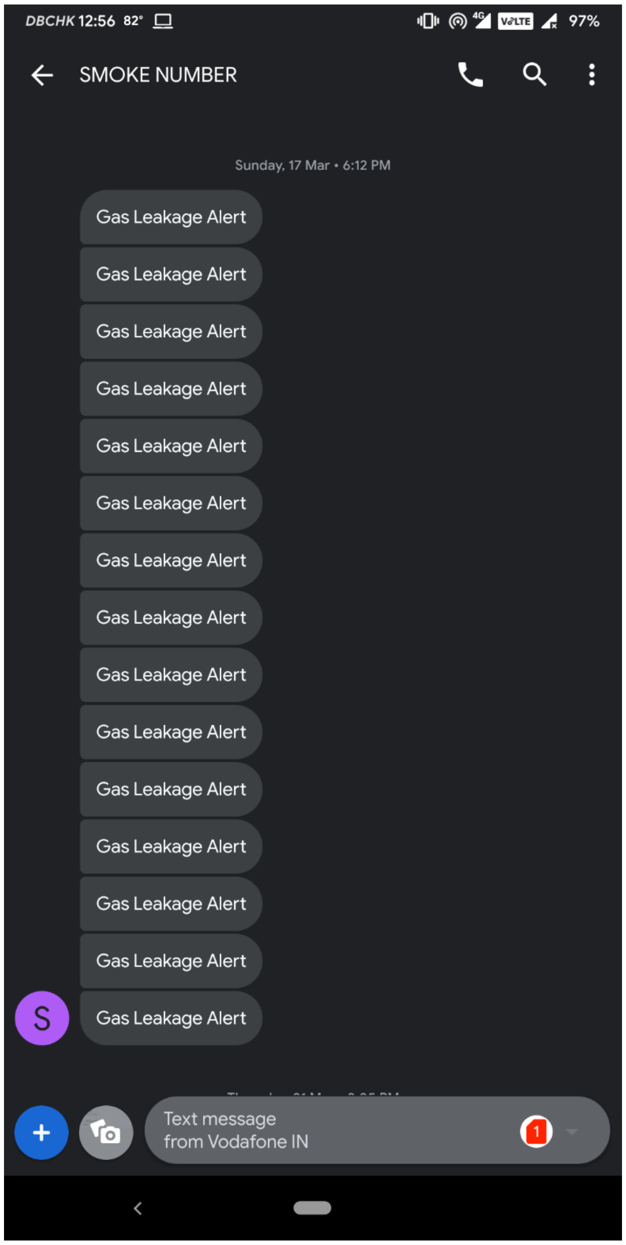
Acceptance testing: Formal testing with respect to user needs, requirements, and business processes conducted to determine whether or not a system satisfies the acceptance criteria and to enable the user, customers or other authorized entity to determine whether or not to accept the system. In our case we know that todays smoke detector just activates the buzzer to notify people around that smoke has been detected. So we embedded SMS alert feature so that as soon as smoke gets detected not only the people surrounded the person registered with the smoke detector will also get notified in there mobile phone, no matter were they are anywhere in this world.



Chapter 6

SNAPSHOTS

Since there is no app, the message is sent using SMS service to the registered phone number.



DBCHK 12:56 82° 97%

SMOKE NUMBER

Sunday, 17 Mar • 6:12 PM

Gas Leakage Alert

Gas Leakage Alert

Gas Leakage Alert

Gas Leakage Alert

Gas Leakage Alert

Gas Leakage Alert

Gas Leakage Alert

Gas Leakage Alert

Gas Leakage Alert

Gas Leakage Alert

Gas Leakage Alert

Gas Leakage Alert

Gas Leakage Alert

Gas Leakage Alert

Gas Leakage Alert

Chapter 7

Project Planning

Project time line table

Project milestones can be shown in a simple time line table. While the table doesn't look complicated, it can provide good amount of information on project progress in a simple and understandable chart.

WEEKWISE PLANNING	TASK
WEEK 1	Information and Components gathering
WEEK 2	Creating Synopsis
WEEK 3	Designing and developing the Physical Model
WEEK 4	Designing and developing the Code for The model
WEEK 5	Testing the Project
WEEK 6	Reporting and Solving the bugs

Timeline Table

Task distribution

Every person was involved in every part of development stage of the Project. Right from purchasing components till the day of testing.

Chapter 8

Conclusion & Future work

CONCLUSION

Smoke detectors are great because they save lives. You should place a smoke detector at least 6 to 12 inches away from a wall. Smoke detectors should always be in a house or an apartment. There are different shapes of smoke detectors, but the ones that are a circle shape are those that are in most homes. There are also smoke detectors shaped as noses, to smell for smoke. There should be at least 2 or 3 smoke detectors in your home. You should install a smoke detector on every floor of a house. Always have a smoke detector in your home for your own safety.

Every house, Factories, Colleges should have smoke detectors, because when it comes to safety one should always take proper precautions. A smoke detector with MOBILE ALERT system is also very useful since when it will send ALERT in the REGISTERED mobile number when the smoke detector detects smoke.

FUTURE SCOPE

Fire accidents can be controlled to a great extent in a place such as forests, homes, colleges industries, trains and some other public places

Fire accidents leads to death of excess of people, by using this technique we can save those life’s easily

To detect the chain smokers (which are hazardous to health)

REFERENCE

https://www.explainthatstuff.com/smokedetector.html

https://en.wikipedia.org/wiki/Smoke_detector

http://www.create .arduino.cc/

http://www.electronicforu.com/