**CHAPTER #1**

**INTRODUCTION**

As we all know that it is very inconvenient for parents to constantly watch over their new born baby while doing their work or chores. Besides that, we know that every house is of different shape and size. Some houses are large and some other are two storied. These days most of the new born babies are kept in a separate room in their own crib. Babies cry to communicate their needs, whether they are hungry or need a diaper change. An IOT (internet of things) based baby monitoring device can alert parents and caregivers whenever their baby is disturbed or awakened from sleep while they are out of immediate hearing distance from the baby.

A sound sensor can detect the baby. A PIR sensor (passive infrared sensor) can be used to check the availability of parents/guardians at the house. If there is no one at home and the baby starts crying, a notification is sent to the parent’s phone indicating the absence of caregiver at home. Else, an alarm system also gets activated to alert the caregiver who is inside the house but out of immediate hearing distance from the baby.

* 1. **Aim**

The main objective of this IOT based project is to let parents stay free of worry and manage both their personal and professional life well. This device keeps them alert about their baby. When they get a notification about the absence of the caregiver, the parent can call the neighbors or any other guardian and ask them to check upon the child immediately.

* 1. **Existing System**

There are some devices available in the market. But does not prove to be effortless, manageable and straightforward. For example,

1. Baby Crying Detector on Samsung Galaxy S5.
2. Snowman Wireless Baby Cry Detector Monitor.
3. Foscam Baby Monitor Camera.
   * 1. **Disadvantages of existing system**

The following are the disadvantages of the existing system:

* They are very expensive.
* The device must be placed within 1 meter distance from the baby. As the mobile phone emits radiation, it could effect baby’s health.
  1. **Proposed System**

To overcome all the drawbacks of the existing system we go for an IOT based device. This system alerts the parent through a notification message and depending on the situation it activates the alarm.

* + 1. **Advantages of Proposed System**

The following are the advantages of the proposed system:

* It does not affect child’s health
* It is cost effective
* It is highly useful and
* It is easy to use

**CHAPTER #2**

**REQUIREMENTS ANALYSIS**

**2.1 Feasibility study**

The next step in analysis is to verify the feasibility of the proposed system. “All projects are feasible given unlimited resources and infinite time”. But in reality, both resources and time are scarce. Project should confirm to time bounce and should be optimal in their consumption of resources.

• Technical feasibility

• Operational feasibility

• Economical feasibility

**2.1.1 Technical Feasibility**

Evaluating the technical feasibility is the trickiest part of a feasibility study. A number of issues have to be considered while doing a technical analysis.

**i) Understand the different technologies involved in the proposed system** Before commencing the project, we have to be very clear about what are the Technologies that are to be required for the development of the new system.

**ii) Find out whether the organization currently possesses the required technologies** Is the required technology available with the organization? If so is the capacity sufficient?

For instance – “Will the current printer be able to handle the new reports and forms required for the new system?”

**2.1.2 Operational Feasibility**

This test of feasibility asks if the system will work when it is developed and installed.

Here are the following questions that will help to solve the operational feasibility of a

Project

* Is there sufficient support for the project from management
* Are the current business methods acceptable to the user?

Since the proposed system was to help reduce the hardships encountered in the existing manual system, the new system was considered to be operationally feasible.

**2.1.3 Economic Feasibility**

To decide whether a project is economically feasible, or not we have to consider various factors as

• Cost benefit analysis

• Long-term returns

• Maintenance costs

**2.2. System specification**

|  |  |
| --- | --- |
| **SOFTWARE REQUIREMENTS** |  |
| Operating System | Windows / Linux |
| Programming Language | Arduino |
| IDE(Integrated development environment) | Arduino IDE |
| **HARDWARE REQUIREMENTS** |  |
| Processor | AMD (or) Intel |
| RAM | 4GB+ |
| Microcontroller | Arduino Uno |
| Sensors and  audio signaling device | Sound , PIR |
| Module | GSM Module |

**CHAPTER #3**

**TECHNOLOGIES USED**

**3.1 INTRODUCTION TO ARDUINO**

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

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

Arduino IDE is a special software running on your system that allows you to write sketches (synonym for program in Arduino language) for different Arduino boards. The Arduino programming language is based on a very simple hardware programming language called processing, which is similar to the C language. After the sketch is written in the Arduino IDE, it should be uploaded on the Arduino board for execution.

**3.2 INTRODUCTION TO ARDUINO IDE**

The Arduino integrated development environment (IDE**)** is a cross-platform application (for Windows, macOS, Linux) that is written in the programming language Java. It is used to write and upload programs to Arduino board.

The source code for the IDE is released under the GNU General Public License, version 2. The Arduino IDE supports the languages C and C++ using special rules of code structuring. The Arduino IDE supplies a software library from the Wiring project, which provides many common input and output procedures. User-written code only requires two basic functions, for starting the sketch and the main program loop, that are compiled and linked with a program stub *main()* into an executable cyclic executive program with the GNU toolchain, also included with the IDE distribution. The Arduino IDE employs the program avrdude to convert the executable code into a text file in hexadecimal encoding that is loaded into the Arduino board by a loader program in the board's firmware

**CHAPTER #4**

**SYSTEM DESIGN**

System design is transition from a user oriented document to programmers or database personnel. The design is a solution, how to approach to the creation of a new system. This is composed of several steps. It provides the understanding and procedural details necessary for implementing the system recommended in the feasibility study. Designing goes through logical and physical stages of development, logical design reviews the present physical system, prepare input and output specification, details of implementation plan and prepare a logical design walkthrough.

**4.1 Software Design**

In designing the software following principles are followed:

• Modularity and partitioning: Software is designed such that, each system should consists hierarchy of modules and serve to partition into separate function.

• Coupling: Modules should have little dependence on other modules of a system.

• Cohesion: Modules should carry out in a single processing function.

• Shared use: Avoid duplication by allowing a single module called by other that need the function it provides

**4.2 Modules and their functionalities**

We have 3 modules

1. Detection

2. Alerting through SMS

3. Alerting through Alarm

**4.2.1 Detection**

Detection-detection is the most important task. We detect the baby cry with the help of sound sensor placed near to the baby. The value of sound sensor is passed to the arduino uno for checking if it is above the limit mentioned in the code. If so, it will consider it as the baby is crying.

**4.2.2 Alerting through SMS**

When the device detects the baby cry, we alert parents and caregiver. For that purpose, we are using GSM module. A notifications sent to the parent whenever the baby cries.. If the caregiver is not present even within the house, a notification is sent again to the parent indicating the absence of the caregiver.

**4.2.3 Alerting through Alarm**

When the device detects the baby cry, we alert parents and caregiver through alarm. For that purpose, we are using buzzer. The buzzer gets activated only when the caretaker is within the house but out of immediate hearing distance. We are using PIR sensor to check the availability of the caregiver within the house.

**4.3 UML Diagrams**

**UML Concepts**

The Unified Modeling Language (UML) is a standard language for writing software blue prints. The UML is a language for

• Visualizing

• Specifying

• Constructing

• Documenting the artifacts of a software intensive system.

The UML is a language which provides vocabulary and the rules for combining words in that vocabulary for the purpose of communication. A modeling language is a language whose vocabulary and the rules focus on the conceptual and physical representation of a system. Modeling yields an understanding of a system.

**Building Blocks of the UML**

The vocabulary of the UML encompasses three kinds of building blocks:

• Things

• Relationships

• Diagrams

Things are the abstractions that are first-class citizens in a model; relationships tie these things together, diagrams group interesting collection of things.

**Things in the UML**

There are four kinds of things in the UML

• Structural things

• Behavioral things

• Grouping things

• Annotational things

Structural things are the nouns of UML models. The structural things used in the project design are: First, a class is a description of a set of objects that share the same attributes, operations, relationships and semantics.

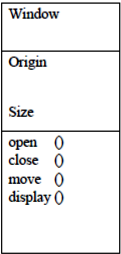


Fig: Class

Second, a use case is a description of set of sequence of actions that a system performs that yields an observable result of value to particular actor.

Fig: Use Case

Third, a node is a physical element that exists at runtime and represents a computational resource, generally having at least some memory and often processing capability.

Fig: Nodes

Behavioral things are the dynamic parts of UML models. The behavioral things used

is

* Interaction

An interaction is a behavior that comprises a set of messages exchanged

among a set of objects within a particular context to accomplish a specific

purpose. An interaction involves a number of other elements, including

messages, action sequences (the behavior invoked by a message, and links the

connection between objects).

display

Fig: Messages

• Association

An association is a structural relationship that describes a set links, a link

being a connection among objects. Aggregation is a special kind of association,

representing a structural relationship between a whole and its parts.

Fig: Association

### Relationships in the UML

There are four kinds of relationships in the UML:

* Dependency
* Association
* Generalization
* Realization

A **dependency** is a semantic relationship between two things in which a change to one thing may affect the semantics of the other thing (the dependent thing).

A **generalization** is a specialization/generalization relationship in which objects of the specialized element (the child) are substitutable for objects of the generalized element (the parent).

Fig: Generalization

**4.3.1 Class Diagram**

An object is any person, place, thing, concept, event, screen, or report applicable to your system. Objects both know things (they have attributes) and they do things (they have methods). A class is a representation of an object and, in many ways; it is simply a template from which objects are created. Classes form the main building blocks of an object-oriented application. Although thousands of students attend the university, you would only model one class, called Student, which would represent the entire collection of students.

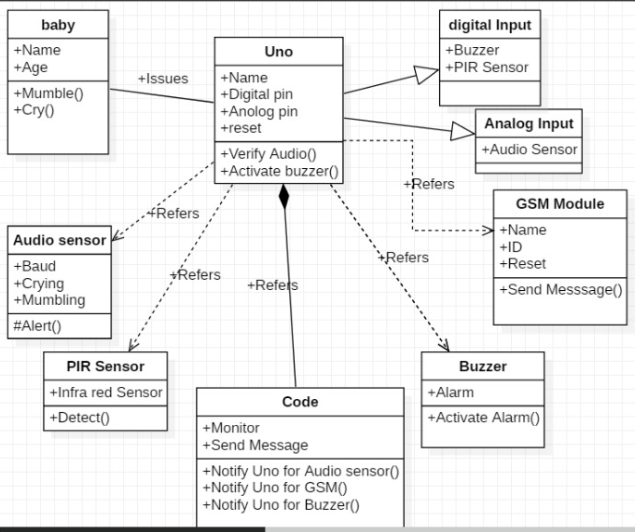


Fig 4.3.1 Class Diagram

### 4.3.2.Use Case Diagram

### A use case diagram is a graph of factors set of use cases enclosed by a system boundary, communication associations between the actors and users and generalization among use cases. The use case model defines the outside (actors) and inside (use case) of the systems behavior.

Use case diagram is quite simple in nature and depicts two types of elements: one representing the business roles and the other representing the business processes

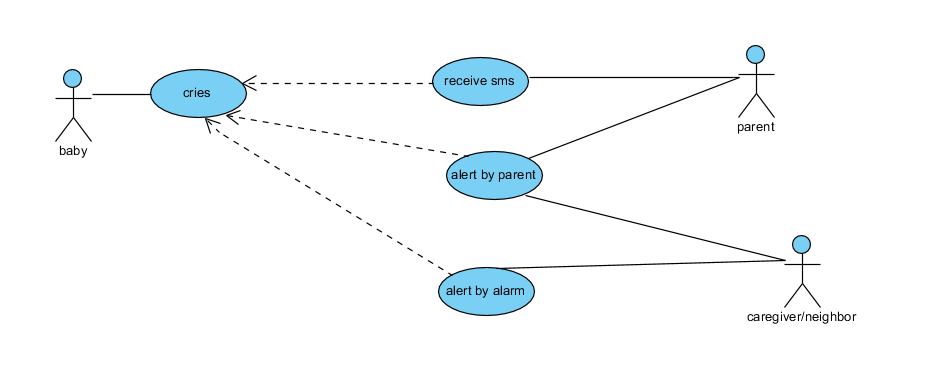


Fig 4.3.2 Use Case Diagram

**4.3.3 Sequence Diagram**

A 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. These diagrams are widely used by businessmen and software developers to document and understand requirements for new and existing systems

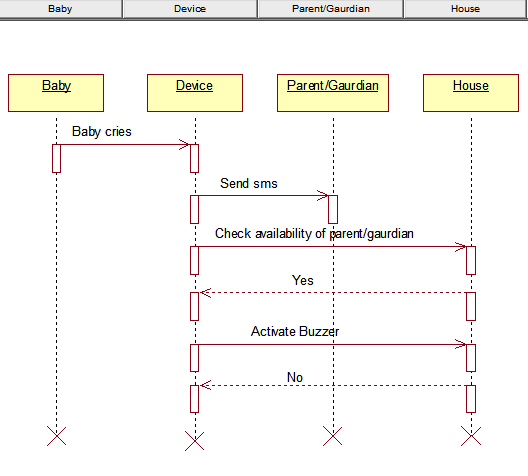


Fig 4.3.3 Sequence Diagram

**4.3.4 Collabration Diagram**

## Collaboration diagrams are used to show how objects interact to perform the behavior of a particular use case, or a part of a use case. Along with sequence diagrams, collaboration are used by designers to define and clarify the roles of the objects that perform a particular flow of events of a use case.  They are the primary source of information used to determining class responsibilities and interfaces.

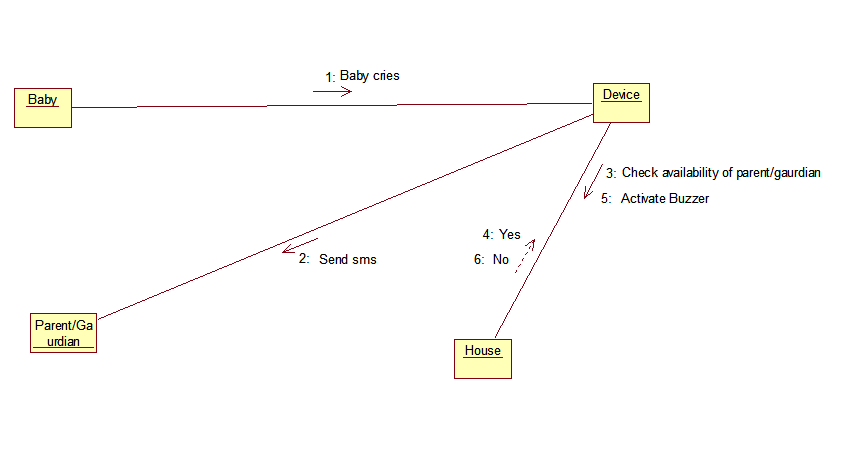


Fig 4.3.4 Collabration Diagram

**CHAPTER #5**

**IMPLEMENTATION**

**5.1 allcombinedcode.ino:**

#include <SoftwareSerial.h>

int inputPin = 2; // choose the input pin (for PIR sensor)

int buzzerPin = 9; //choose the input pin (for buzzer)

SoftwareSerial sim(10, 11); //10-Rx -> Tx ,11-Tx -> Rx used for communication

String number = "+917893421290"; // mobile number to which alert message has to be sent

namespace monitoringSystem

{

class babyMonitor {

protected:

//the volume to detect if the baby is crying or no

int isCrying;

// Sometimes the baby start mumbling or playing, specially at the age when he starts learning how to speak

int isMumbling;

//This is to configure the baby

public:

void setVoice(int voiceLevel, boolean isC)

// set voice level, isC =True to set the voice level of "isCrying" else to set the voice level of isMumbeling

{

if (isC){

isCrying= voiceLevel;

}

else{

isMumbling= voiceLevel;

}

}

void initialize()

// initialize the system

{

Serial.begin(9600);

// Beginning the band manager restarts the modem

Serial.println("Starting the service");

// This part help when you debug problems with GSM Shield connectivity

//int pin\_query = PINManager.isPIN();

//Serial.println(pin\_query);

}

void SendMessage()

{

Serial.println ("Sending Message");

sim.println("AT+CMGF=1"); //Sets the GSM Module in Text Mode

delay(1000);

//Serial.println ("Set SMS Number");

sim.println("AT+CMGS=\"" + number + "\"\r"); //Mobile phone number to send message

delay(1000);

String SMS = "Baby is crying..."; //Assigning message which is to be send

sim.println(SMS); //Send the SMS to the given mobile number

delay(100);

sim.println((char)26); // ASCII code of CTRL+Z

delay(1000);

}

void monitor() {

int val = 0; //variable to read the value from PIR sensor0

int noise; //variable to read the value from Sound sensor

noise = analogRead(A0); //read input from Sound Sensor here A0 - indicates analog pin A0

val = digitalRead(inputPin); // read input value (from pin 2)

Serial.println(noise);

Serial.println(val);

if (noise < isMumbling) {

}

if (noise > isMumbling && noise < isCrying ) {

}

if (noise > isCrying) {

SendMessage(); // calling the function

if (val > 0) // check the existence of person

{

Serial.println("Motion detected!");

tone(buzzerPin, 50, 1000); // used to activate the buzzer with frequency and time

val = 0;

}

delay(1000);

}

delay(1000);

}

};

};

monitoringSystem::babyMonitor \_babyMonitor;

void setup()

{

delay(7000); //delay for 7 seconds to make sure the modules get the signal

pinMode(inputPin, INPUT); // declare pir sensor as input

pinMode(buzzerPin, OUTPUT); // declare the pin as output mode

Serial.begin(9600); // Serial port begin (or) Setting the baud rate of Serial Monitor (Arduino)

sim.begin(9600); // Setting the baud rate of GSM Module

\_babyMonitor.initialize() ;

// set isCrying at 600

\_babyMonitor.setVoice(300,false);

// set isMumbeling at 300

\_babyMonitor.setVoice(600,true);

// personalize configuration

// delay 1 minute to prepare the setup, to not infuluence the sound detector

delay(15000);

}

void loop()

{

//start monitoring

\_babyMonitor.monitor();

}

**CHAPTER #6**

**TESTING**

**INTRODUCTION**

The development of software systems involves of a series of production activities where opportunities for injection of human fallibilities are enormous. Errors may begin to occur at the very inception of the process where the objectives may be erroneously or imperfectly specified, as well as in later design and development stages. Because of human inability to perform and communicate with perfection, software development is accompanied by a quality assurance activity. Testing Techniques:

Testing is the process of executing a program with the intention of finding errors. The various test strategies used for testing the software are as follows.

**6.1 Unit Testing**

Unit testing is a software development process in which the smallest testable parts of an application, called units are individually and independently scrutinized for proper operation. Unit testing is often automated but it can also be done manually. This testing mode is a component of Extreme Programming (XP), a pragmatic method of software development that takes a meticulous approach for building a product by means of continual testing and revision.

Unit tests are written from a programmer's perspective. They ensure that a particular method of a class successfully performs a set of specific tasks. Each test confirms that a method produces the expected output when given a known input.

**6.2 Integration Testing**

Integration testing, also known as integration and testing (I&T), is a software development process in which program units are combined and tested as groups in multiple ways. In this testing, we find errors that have occurred during the integration .The goal of testing is to detect the design errors while focusing on the testing the interconnection between modules.

**6.3 Validation Testing**

This testing concentrates on confirming that the software is error-free in all respects. All the specified validations are verified and the software is subjected to hardcore testing. It also aims at determining the degree of deviation that exists in the software designed from the specification that are listed out and are corrected.

**6.4 System Testing**

In this testing, the system is tested for the errors after coupling all the modules together. The system is tested against the specified requirements to see if all the requirements are met and the system performs as specified by the requirements.

**CHAPTER #7**

**OUTPUT SCREENS**

**7.1 IDE (Integrated Development Environment)**

This particular page serves as a home page for Aurduino IDE. This provides a basic

area to write code and execute.

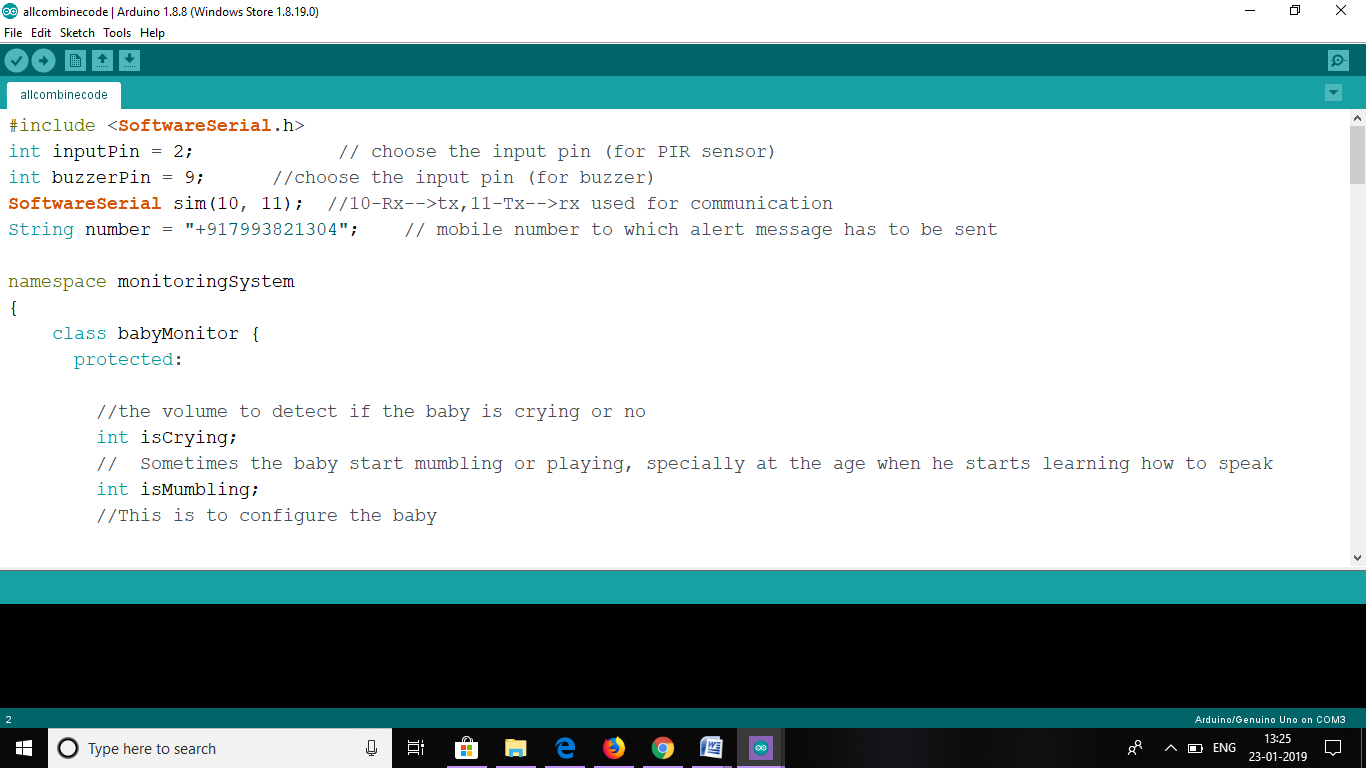


Fig: 7.1 IDE

**7.2 TOOLS**

This selection provides option to select Serial monitor, click on the 5th option to view it in toolbar in the Arduino IDE.

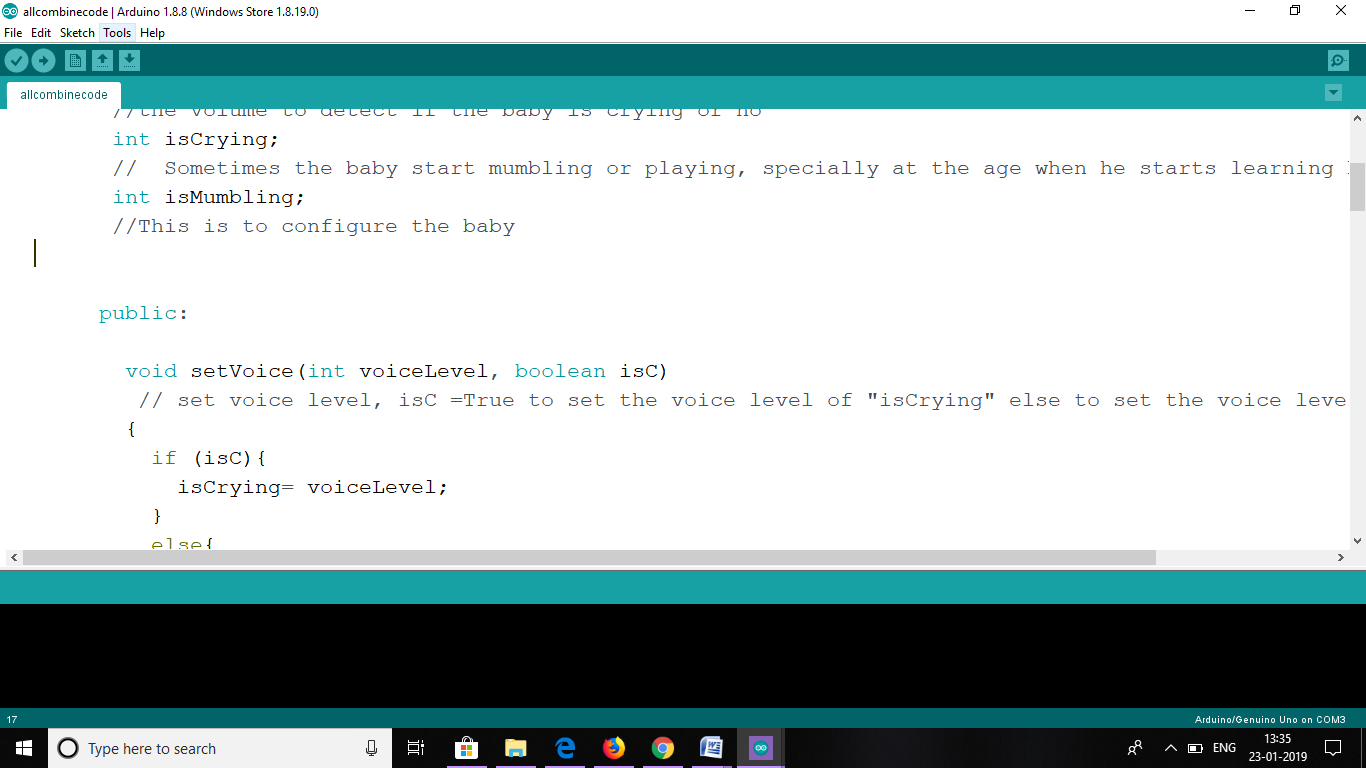


Fig: 7.2 Tools

**7.3 Serial monitor**

This window is called the Serial Monitor and it is part of the Arduino IDE software. Its job is to allow you to both send messages from your computer to an Arduino board (over USB) and also to receive messages from the Arduino.

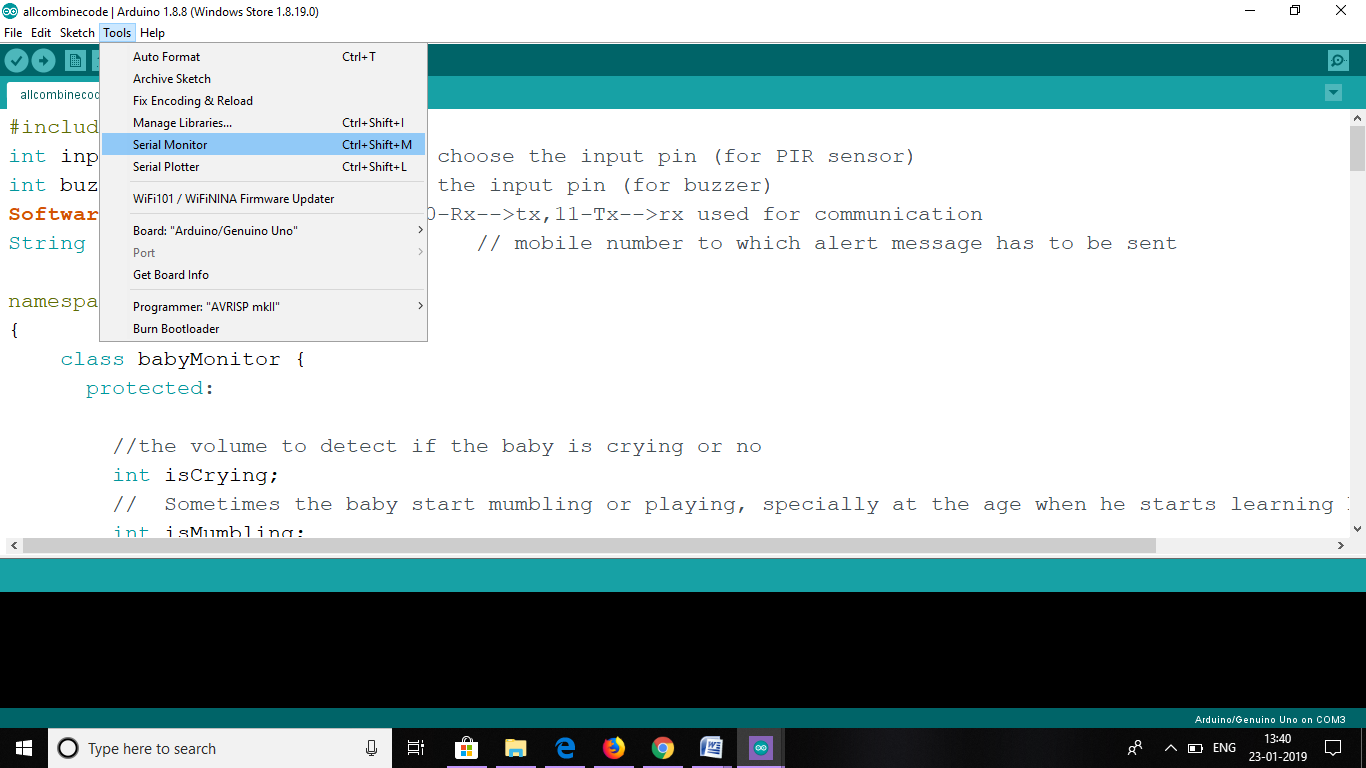


Fig: 7.3 Serial Monitor

**7.4 COM3**

Select the serial device of the board from the Tools | Serial Port menu. This is likely to be COM3 or higher. To find out, you can disconnect your board and re-open the menu; the entry that disappears should be the Arduino board.

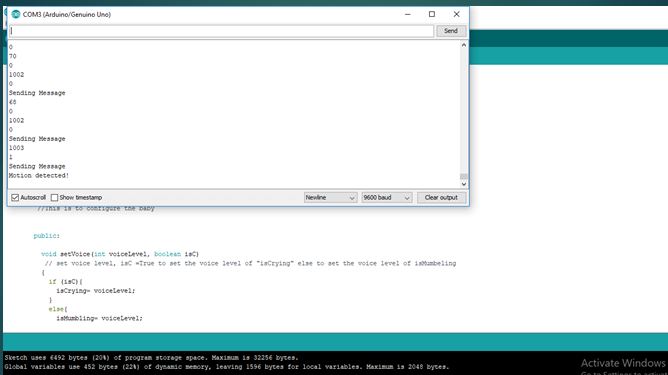


Fig: 7.4 COM3

**7.5 MESSAGE**

A written communication sent to or left for a recipient to be informed.

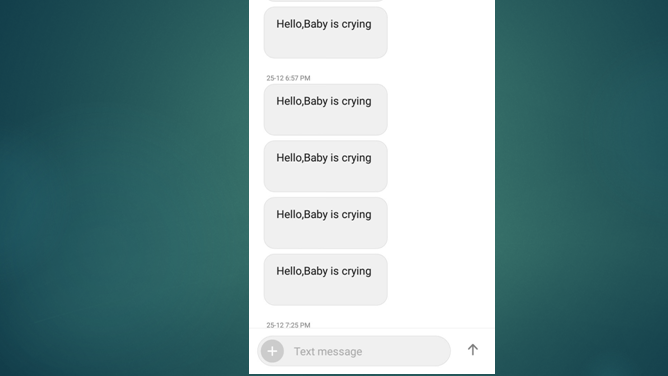


Fig: 7.5 Message

**CHAPTER #8**

**CONCLUSION**

It has been a great pleasure for me to work on this exciting and challenging project. This project proved good for me as it provided practical knowledge of not only in arduino programming, but also about all handling procedure related with “An IOT based device”. With this proposed idea we would like to enter into the market and enhance ourselves in developing and updating the device. More services like adding a camera for live monitoring and playing lullaby whenever baby cries etc. will be implemented in the upcoming days to match ourselves with current market scenarios.

**CHAPTER #9**

**REFERENCES**

**WEBSITES**

* http://www.instructables.com/id/Sim900-GSM-module-interfacing-to-arduino/
* http://www.circuitstoday.com/interface-gsm-module-with-arduino
* http://miliohm.com/sim800l-arduino-tutorial/
* https://learn.sparkfun.com/tutorials/sound-detector-hookup-guide
* http://www.instructables.com/id/PIR-Motion-Sensor-Tutorial/
* http://www.instructables.com/id/Interfacing-Buzzer-to-Arduino/