# Acknowledgement

*Apart from the efforts of the team, the success of any project depends largely on the encouragement and guidelines of many others. We take this opportunity to express our gratitude to the people who have been instrumental in the successful completion of this project.*

*The completion of any inter-disciplinary project depends upon cooperation, coordination and combined efforts of several sources of knowledge.*

*We are eternally grateful to our project supervisor* ***Mr.Shuvro Roy*** *for his even willingness to give us valuable advice and direction; under whom we executed this project. His constant guidance and willingness to share his vast knowledge made us understand this project and its manifestations in great depths and helped us to complete the assigned tasks. We are highly thankful to our project guide, whose invaluable guidance helped us understands the project better. Although there may be many who remain unacknowledged in this humble.*

**Aanandita Diwan Vishal Sangwan Shubham Aswal Nishant Sisodia**

02314504419 01214504419 03714504419 03014504419

# TABLE OF CONTENTS

#### Software Requirement Specification

1. Introduction................................................................................................................... 1

1.1 Purpose................................................................................................................ 1

1.2 Methodology & Modules.................................................................................... 1

1.3 Document Conventions....................................................................................... 2

1.4 Intended Audience and Reading Suggestions..................................................... 3

1.5 Project Scope....................................................................................................... 3

2. Overall Description....................................................................................................... 4

2.1 Product Perspective............................................................................................. 4

2.2 Product Features.................................................................................................. 4

2.2.1 Core Features............................................................................................... 4

2.2.1.1 Dew Point Detection...................................................................... 4

2.2.1.2 Light Intensity Calculation............................................................ 5

2.2.1.3 Final Result.................................................................................... 5

2.2.2 Additional Features.................................................................................... 5

2.3 User Classes and Characteristics......................................................................... 5

2.4 Operating Environment....................................................................................... 6

2.5 Design and Implementation Constraints ............................................................ 6

2.6 User Documentation ........................................................................................... 7

2.7 Assumptions and Dependencies ......................................................................... 7

3. Software Interface Requirements ................................................................................. 7

3.1 User Interfaces..................................................................................................... 7

3.2 Hardware Interfaces ............................................................................................ 8

3.3 Software Interfaces ............................................................................................. 9

3.4 Communications Interfaces.................................................................................. 9

4. Other Non-functional Requirements .............................................................................. 12

4.1 Performance Requirements................................................................................... 12

4.2 Safety Requirements.............................................................................................. 12

4.3 Security Requirements........................................................................................... 13

4.4 Software Quality Attributes...................................................................................  13

5. Diagrams.......................................................................................................................... 14

6. Risk Management............................................................................................................. 22

7. Implementation................................................................................................................. 26

8. Conclusion........................................................................................................................ 29

**List of Figures**

Figure 1- Use case of fog..................................................................................................... 14

Figure 2-Use case of distance detection............................................................................... 15

Figure 3-Class Diagram…………………………………………………………………… 16

Figure 4-Sequence Diagram………………………………………………………………. 17

Figure 5-Activity Diagram………………………………………………………………… 18

Figure 3-DFD level 0............................................................................................................ 19

Figure 4-DFD level 1 for fog detection................................................................................ 19

Figure 5- DFD level 2 for fog detection............................................................................... 20

Figure 6- DFD level 1 for obstacle detection....................................................................... 20

Figure 7- DFD level 2 for obstacle detection....................................................................... 21

Figure 8-ER diagram............................................................................................................ 22

### **SRS(Software Requirement Specification)**

**1. Introduction**

Road accidents are common in low visibility, many road accidents occur in winter weather due to fog or in urban areas due to smog, especially on highways. Police being aware of these fact issues special notices to follow several traffic rules, it has led to a low fatality rate, but still many collisions occur due to low visibility.

Data from the ministry of road transport and highways shows that as many as 12,678 people lost their lives in fog-related road accidents in 2018. In 2017, it was 11,090. Delhi traffic police say the problem is more rampant on highways and expressways around the city. These fatalities increase by 8%-10% when visibility becomes zero.

North India, which is engulfed in thick fog in December and January, has the most number of deaths during these months. Our main focus in this project is to detect dense fog so that we can avoid vehicle collision and reduce fatalities.

**1.1 Purpose**

To date, no work has been done on a sensor to determine visibility, so we are trying to rectify that, by using various sensors (like humidity, temperature, Light sensor etc.) together so that we can detect fog and reduce collisions. This project is an effort to further reduce the fatality rate by determining the visibility and limiting the speed of the vehicle by alerting the driver.

Every year approximately 10,000 lives are lost due to low visibility in fog in North India. The bad news is that they are getting deadlier with every passing year. Fog related road deaths accounting for 16% of the total road crashes in 2017. So, the main purpose of our project is to avoid fog-related fatalities and crashes.

As we know working efficiency of trains and flights reduces when visibility is below 100 meters and 550 meters respectively. We hope that our prototype could be the base to develop technology for better visibility so that flights and trains do not get affected in low visibility areas, particularly in hilly areas.

**1.2 Methodology & Modules**

We will have two modules First detects fog , Firstly, we take a humidity sensor.

In this it will take moisture level from the atmosphere and if moisture is above 80%.

below this, there will be not enough water droplet or moisture to create fog.

It will go to our next sensor where we can see the temperature.

If the temperature is 24**°** Celsius or 75 **°**Fahrenheit or less than this, there will be a chance of fog to appear. If the temperature is higher, it will evaporate the moisture which is present in the atmosphere. After these 2 tests, we will go to our 3rd sensor which is a dew point sensor , dew point is the temperature to which air must be cooled to become saturated with water vapour. The threshold is and will be near to the temperature in the atmosphere. Then it will compare or check the difference between the temperature and dew point. If the difference between the temperature and dew point is 2.5 c or 4.5f the fog will be formed or we can say the fog will be detected. As water droplets and evaporated water stay at the same temperature, they can form a fog.

The lesser the difference in temperature and dew point, the more the fog or we can say the denser the fog is. If the difference is higher then the fog can’t be formed as there is temperature difference in water droplet and evaporated water will not mix and the water settles down and gets evaporated.

Example:

Temperature = 70.0 **°**F

Dew Point = 65.5 **°**F.

Humidity = 83%

In this example, the fog begins forming at 83%.

Mist== Mist forms when the relative humidity is greater than about 70%.

After sufficient indications for fog formation is given by temperature and humidity sensors we will do a third and final test to confirm fog existence. That is light intensity test.

We will have two seperate pairs of led and LDR light detectors. The LED will be thrown on the LDRs. In one pair we will keep the led and LDR in a transparent box where light can permeate the environment but no fog can enter the box. In second pair we will keep the led and LDR in oven air conditions where the fog and air is free to flow . Once the led is turned on the LDR's will detect the light intensity of the led. If the intensity of light in pair 2 , the one which is not in the transparent box is less than the one in transparent box then we will give the signal that fog is there. Because in case of fog pair 2 led light will scatter when light will hit fog particles and hence its intensity will be less than the pair whose light path was uninterrupted by fog particles

Second module detects, Using ultrasonic sensor distancewill bedetermined*.* If something comes in the path the car will turn on the LED display to indicate Stop . indicating to driver something is in path. how far the sensor can detect the distance if driver goes beyong the speed where it is in trouble of being in accident the led will display again for the driver to slow down

**1.3 Document Conventions**

*This document features some terminology that readers may be unfamiliar with*

*SRS: Software Requirement Specification*

***Humidity****: This refers to Relative humidity.****Relative humidity****tells us how much water vapour is in the air.*

***Vehicle:****Transport entity used for transportation of passengers.*

***Dew Point:****The****dew point****is the temperature to which air must be cooled to become saturated with water vapour. When cooled further, the airborne water vapour will condense to form liquid water (****dew****).*

***LDR:****A photoresistor or*light-dependent resistor*is an electronic component that is sensitive to light. When light falls upon it, then the resistance changes.*

**1.4 Intended Audience and Reading Suggestions**

The intended readers of this document are current and future developers working on “Fog Detection and Obstacle avoidance System” and the sponsors of the project.

**1.5 Project Scope**

*The “Fog Detection and Obstacle avoidance System” is a hardware application, which helps people to avoid obstacles in the vehicle’s path when visibility is low due to fog conditions. The Fog Detection and Obstacle avoidance System is composed of two main components: a fog detection system and an obstacle avoidance system which will support and interact with the client-side to alert them in case of danger.*

*Currently, this project is only a prototype as we can’t make changes in the actual car's accelerator and braking system. But with future collaboration with car designers, we can insert this module into actual cars.*

Advantages & Applications

Advantages: This project will contribute greatly to saving lives.

It will make travel easier and safer even in low visibility. Emergency cases like ambulance, fire brigade, or police assistance can arrive even in harsh weather conditions.

Lesser number of accidents

It will make driving safer in fog conditions where visibility is very low resulting in loss of lives,

Possible application: It will also be beneficial in the future with possible applications extending to make low-cost automatic cars, and even trains or aeroplanes operating in a foggy environment with low visibility conditions

Limitations: Currently this project is only a prototype as we can’t make changes in the actual car's accelerator and braking system. But with future collaboration with car designers, we can insert this module in actual cars.

**2. Overall Description**

**2.1 Product Perspective**

Our project is divided into two different modules first is the detection of fog and the second is avoidance of obstacles if and when the fog is detected. In the fog detection module, we first get the input from the temperature and humidity sensor, the temperature and humidity are used to calculate the dew point, the dew point is then used to determine whether the fog is present or not, as there are various types of fog-like radiation fog or dense fog. We use a second module to determine whether the fog affects our visibility or not. In this module we use two LED’s and two light detecting diode, we input the led to the light detecting diodes and we calculate the intensity. The difference between the two intensity of the LDR’s is then used to determine whether our visibility is affected or not. If the visibility is affected then we will move on to obstacles avoidance, in the obstacle avoidance module we first get the speed of the car. The speed is input from the optometer using rotations per second, using this speed we calculate the braking distance. The braking distance also uses the coefficient of fractions therefore we use humidity and temperature to detect what the coefficient of friction might be, in case of rain, snow or dry environment. The ultrasonic sensor constantly detects the distance from the car, if any obstacle comes within the braking distance, the LED light will be turned on and the driver will be alerted.

**2.2 Product Features**

The following list offers a brief outline and description of the main features and functionalities of the Fog Detection and Obstacle avoidance System. The features are split into two major categories: core features and additional features. Core features are essential to the application’s operation, whereas additional features simply add new functionalities.

**2.2.1 CORE FEATURES**

**2.2.1.1. DEW Point DETECTION**

Gets temperature and humidity from DHT!! temperature and humidity sensory calculates the dew point using the formula

Td = T - ((100 - RH)/5.) where Td is dew point temperature (in **degrees Celsius**), T is observed temperature (in **degrees Celsius**), and RH is relative humidity (in per cent).

This relationship is fairly accurate for relative humidity values above 50%.

**2.2.1.2. LIGHT INTENSITY CALCULATION**

*LED light is allowed to fall upon LDR*.

*One LED-LDR pair is cased within an insulated box where ambient light is allowed to enter but fog is not*whereas *Another LED-LDR pair is out in open where both ambient light and fog is allowed to pass through freely.*

*The light intensity values from both pairs are compared if there is variation between the two values that indicated fog/pollution is present.*

**2.2.1.3 FINAL RESULT**

If the difference between the dew point and the current temperature is 2 degree Celsius and if fog is detected in 2.2.1.2 then Fog is present in eh environment and the visibility condition is low.

**2.2.2 ADDITIONAL FEATURES**

***Obstacle AVOIDANCE***

The speed is input from the optometer using rotations per second, using this speed we calculate the braking distance. The braking distance also uses the coefficient of friction therefore we use humidity and temperature to detect what the coefficient of friction might be, in case of rain, snow or dry environment. The ultrasonic sensor constantly detects the distance from the car, if any obstacle comes within the braking distance, the LED light will be turned on and the driver will be alerted.

**2.3 User Classes and Characteristics**

Users that interact with the system: users of the Fog Detection and Obstacle avoidance System, and administrators. Each of these types of users has a different use of the system so each of them has its requirements.

The application users can only use the application to be alerted through LED light. This is usually the driver who will get alerted through a small Led light blinking on indicating danger in the path when visibility is low.

This means that the user is expected to know how to drive and can have enough awareness about how to operate the car.

The administrators only interact when setting up the Fog Detection and Obstacle avoidance System. This is set up only once. This means administrators are expected to be hardware literate and to be able to work with setting up the laser and other hardware system requirements.

It is also important that the application is as user- friendly as possible, most importantly, the application must be reliable.

**2.4 Operating Environment**

The main components of the Fog Detection and Obstacle avoidance System is the software application as well as the Hardware, which will be limited to the Arduino system (specifically Arduino UNO or Arduino NANO). The application is not resource- or graphics-intensive, so there are no viewing constraints.

The app will rely on several functionalities built into Arduino’s Application Programming Interface (IDE), Additional libraries also need to be added in the IDE which is: adafruit and its dependencies, TimerOne, DHT11 and NewPing.

Beyond that, the application is a self-contained unit and will not rely on any other software components.

The application will, however, frequently interact with the Hardware through UNO.

**2.5 Design and Implementation Constraints**

***Software code***

*The primary design constraint is uploading the code to Arduino UNO. Since the application is designated for Arduino, library inclusion will be a major design consideration.*

*Other constraints such as processing power are also worth considering. Fog Detection and Obstacle avoidance System are meant to be quick and responsive, so each feature must be designed and implemented with efficiency in mind.*

***Hardware Configuration***

 The system shall be built using a standard component available easily in the market and over the internet. Components should be in proper working order and condition and checked beforehand.

*Wires should not have loose connections power supply should be able to support the proper requirements and not be under or over the limit of safe voltage.*

*The laser should point head on the LDR’s and move after installation. The initial configuration of the intensities should be noted with care during installation.*

*The product must be stored in such a way that no damage occurs to it in future.*

*General knowledge of basic computer and hardware skills is required to install the product.*

**2.6 User Documentation**

Online User Documentation and Help System Requirements

*As the product’s components are available in markets or over the internet easily. The code is simple enough and once uploaded to the Arduino will work without re-configuration.*

*Also, a tutorial CD for a visual walkthrough as well as a pdf document explaining how to use will be provided to a user at deployment time.*

**2.7 Assumptions and Dependencies**

*The System will run on Arduino Uno or Nano. No Internet Connectivity is required after Arduino IDE and libraries are installed. The product shall be based on hardware and has to be run from an Arduino (uno or nano).*

*The product shall take an initial load time when the code is uploaded to Arduino. Afterwards, no connectivity to the internet or laptop/desktop is required and Arduino will be able to run and manage all the components. The system will be standalone with no external help required.*

*The performance shall depend upon the hardware components of the client/customer.*

System Features *The System continuously checks information about the weather conditions. Of course, this project has a high priority because when visibility is a low chance of accidents increases.*

**3.Software Interface Requirements**

This section provides a detailed description of all inputs into and outputs from the system. It also describes the hardware, software and communication interfaces and provides basic prototypes of the user interface.

The IDE used is Arduino IDE.

The Arduino used is Arduino Uno.

**3.1 User Interfaces**

A Led will be used to indicate to the user whether there is an obstacle in the vehicle’s path or not. The LED will be turned on if there is an obstacle and a break should be pressed with the current speed. The Led will automatically turn off once the obstacle is no longer in the way.

**3.2 Hardware Interfaces**

The system is standalone once the code is uploaded onto the Arduino. From then on the Arduino will be able to handle the hardware on its own.

HARDWARE : Actual Cost

| S.No | Components and supplies | | **Qty.** | **COSTS** |
| --- | --- | --- | --- | --- |
| 1 | Arduino UNO R3 development board ans USB | | 1 | 400 |
| 2 | Laser | | 2 | 100 |
| 3 | Light Dependent Resistor LDR (Photoresistor) | | 2 | 20 |
| 4 | Ultrasonic Sensor - HC-SR04 (Generic) | | 1 | 90 |
| 5 | Jumper wires (generic) | | - | 30 |
| 6 | Breadboard | | 3 | 225 |
| 7 | DHT11 - Temperature and Humidity Sensor Module | | 1 | 110 |
| 8 | RGB LED 5mm | | 1 | 1 |
| 9 | Resistors (2x100OHM, 1x120OHM) | | 3 | 4 |
| 10 | Potentiometer | | 1 | 20 |
| TOTAL COST | | RS . 1000 | | | |

Additional Cost for simulating the project:

In real time we can directly take the speed of car from speedometer

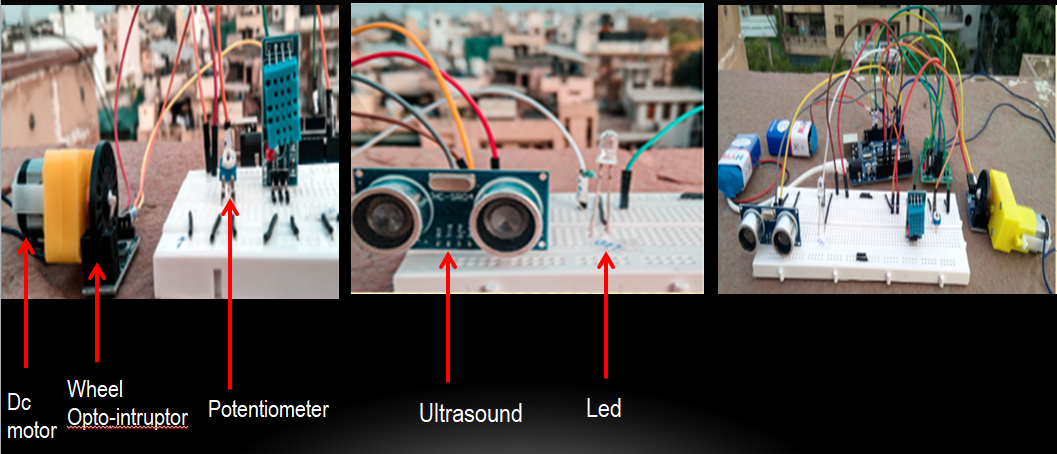
| **Total Cost** (approximate) | | | | Rs.885 |
| --- | --- | --- | --- | --- |
| 1 | 6-9V TT gear DC motor | 1 | 100 | | |
| 2 | L293D DC motor driver 12V | 1 | 130 | | |
| 3 | Speed encoder disk | 1 | 50 | | |
| 4 | Photo Interruptor | 1 | 150 | | |
| 5 | 9V Battery | 2 | 50 | | |
| 6 | Battery Connector clips | 2 | 20 | | |
| TOTAL COST | | RS. 500 | | | |

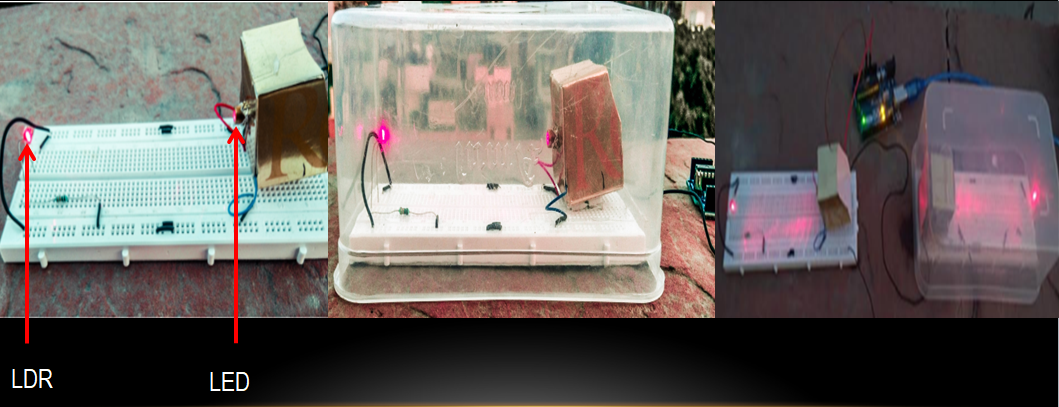
TOTAL COST OF PROJECT : Rs.1500

**3.3 Software Interfaces**

The communication between the driver and the hardware will be monitored through Arduino Uno.

**3.4 Communications Interfaces**

****



** **

** **



**** 

** **



** **

** ** 



** **

** **

**4. Other Non**-**functional Requirements**

**4.1 Performance Requirements**

Performance should not be an issue because all of our program codes involve small pieces of data which will require very little computation and thus will occur very quickly.

updates should only take a few microseconds as the computation seep of Arduino is very fast calculation algorithms used in the application will be highly efficient, taking only a fraction of a second to compute.

**4.2 Safety Requirements**

Hardware will not affect the data car or its interior design as it's stored outside of the original car model. It cannot cause any damage to the car or its internal components. The only potential safety concern associated with this application applies when an accident occurs and the components get damaged.

**4.3 Security Requirements**

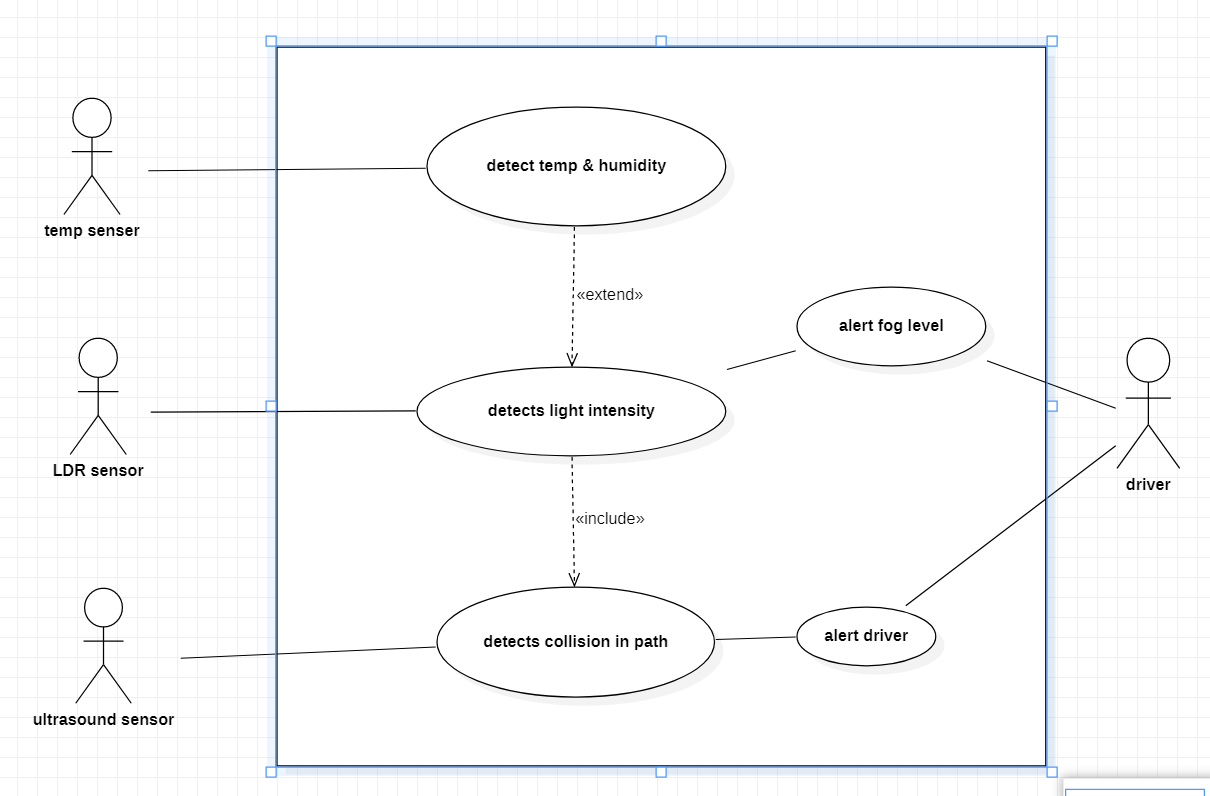
This application assumes that only the user or whoever he/she allows will have access to his/her car. With that being said, anyone in the car will be able to use this as this is a safety application used to increase the safety of anyone and everyone in the car. The system shall use secure connections in installation to ensure the proper working of the project.

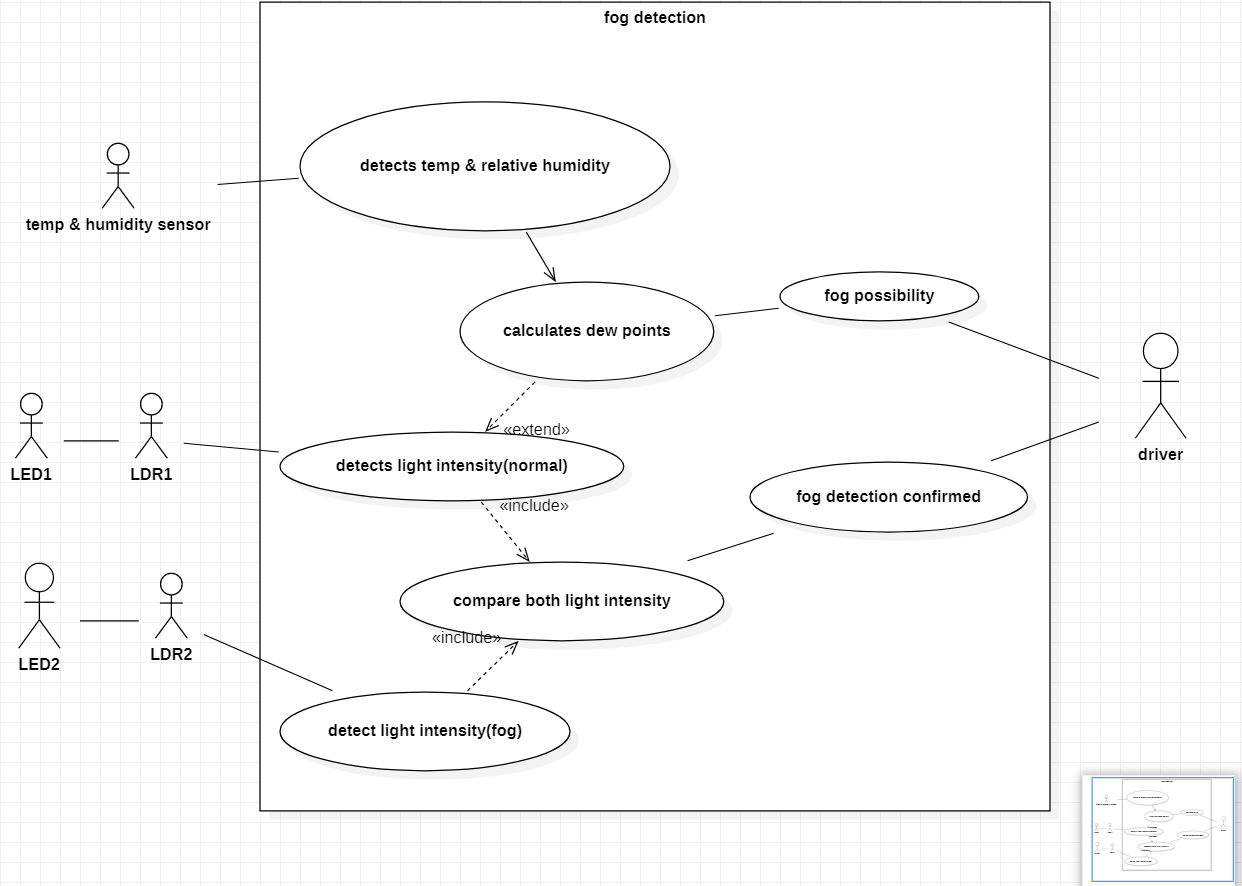
**4.4 Software Quality Attributes**

| TASK NAME | 24 SEP 2020 | OCT 2020 | 5 NOV 2020 | 9 NOV 2020 | 10 NOV 2020 | 11 NOV 2020 | 1 FEB 2021 | 28  FEB  2021 | 1  MAR  2021 | 8  MAR  2021 | 30  MAR  2021 | 8  APR  2021 | 9  APR  2021 | 30  APR  2021 |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Planning  &  Research |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Finalization  of  project |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Synopsis |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Implementation |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Testing |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Documentation |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Project evaluation |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| Implementing  changes |  |  |  |  |  |  |  |  |  |  |  |  |  |  |

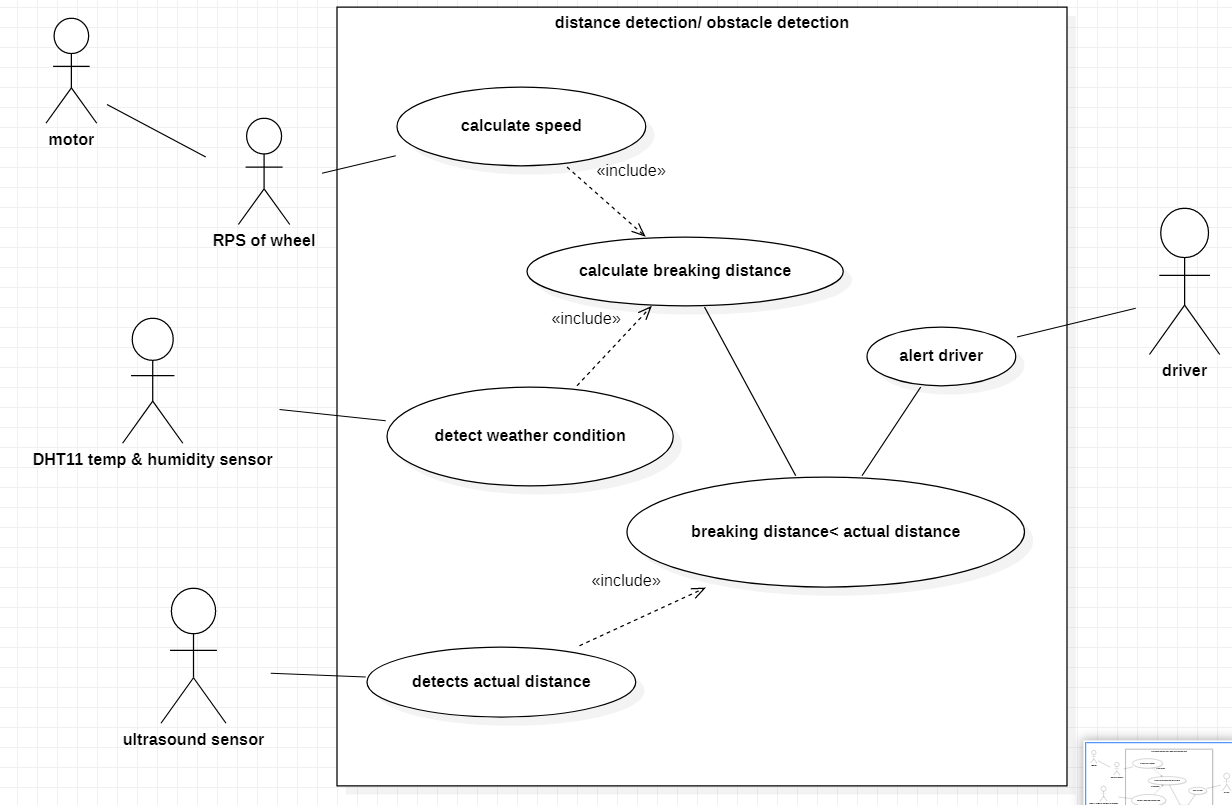
**5.DIAGRAMS**

**5.1 Use Case Diagram**

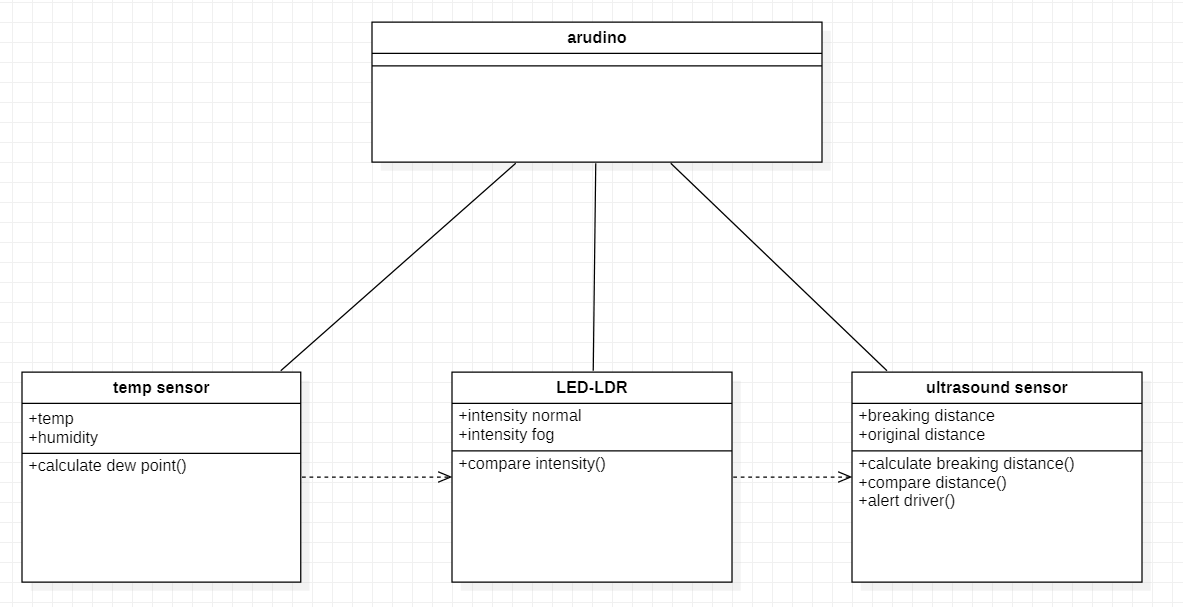
****



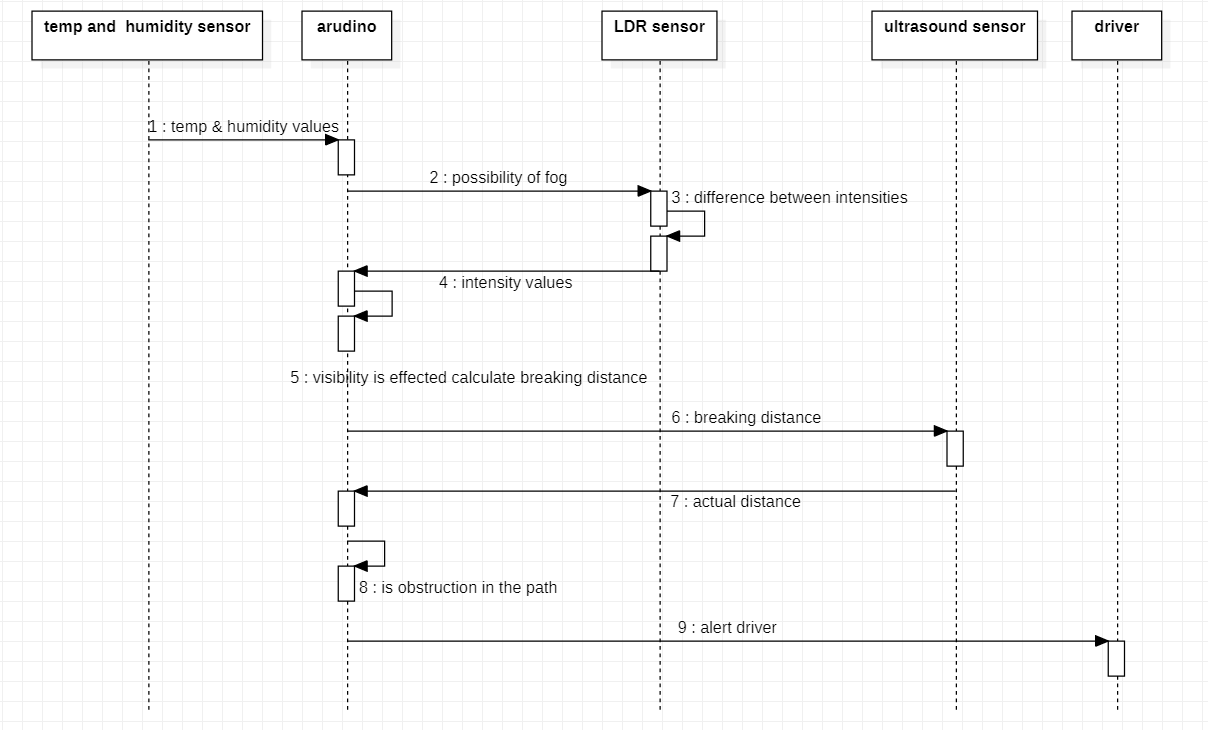




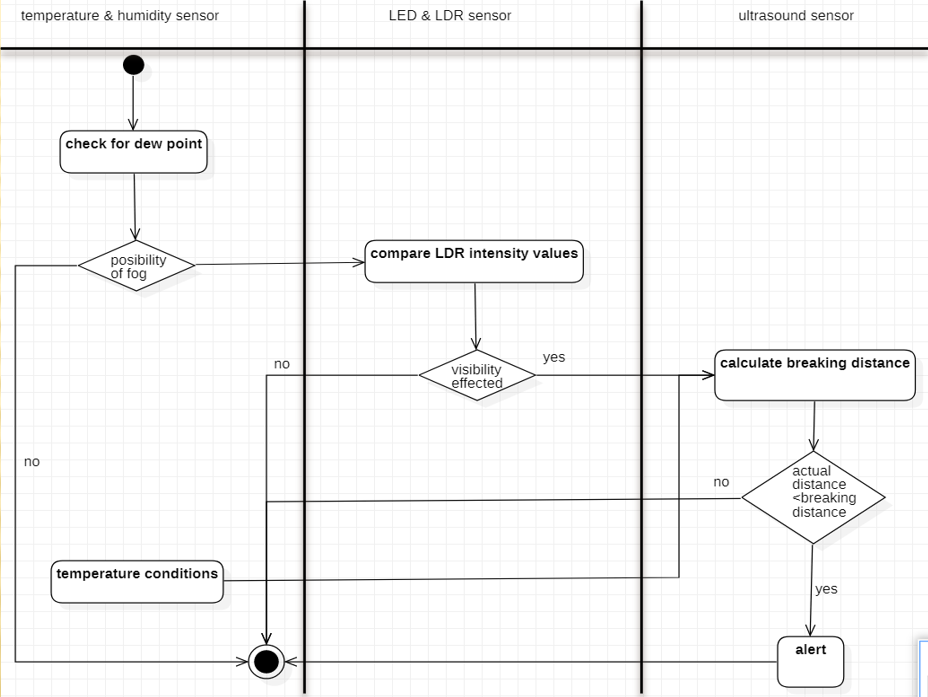
**5.2 Class Diagram**



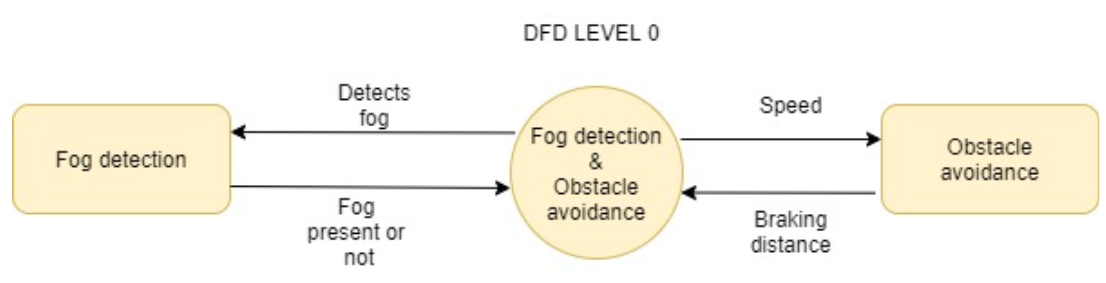
**5.3 Sequence Diagram**

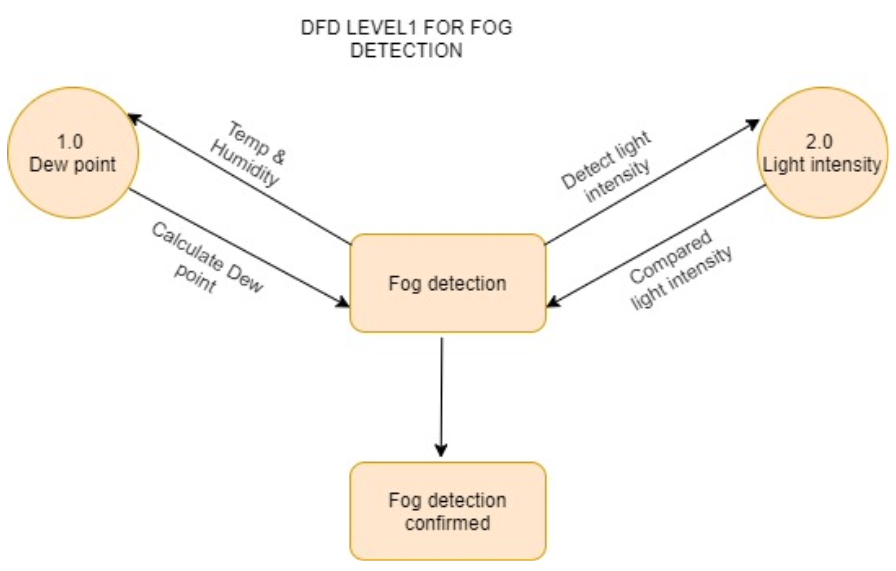


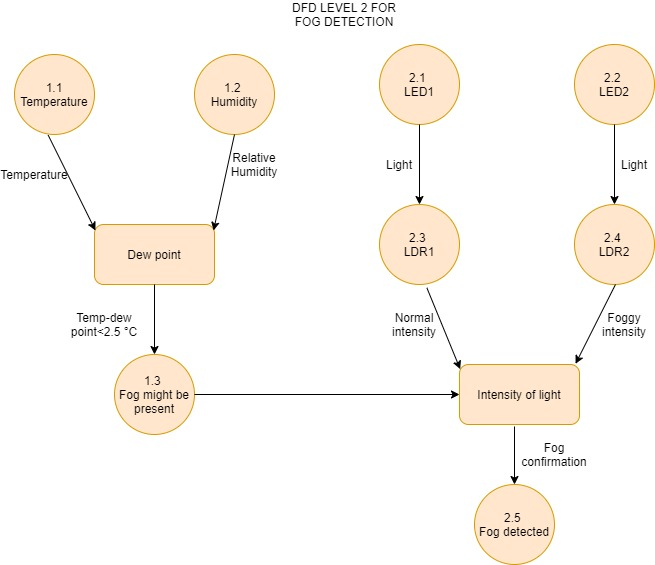
**5.4 Activity Diagram**

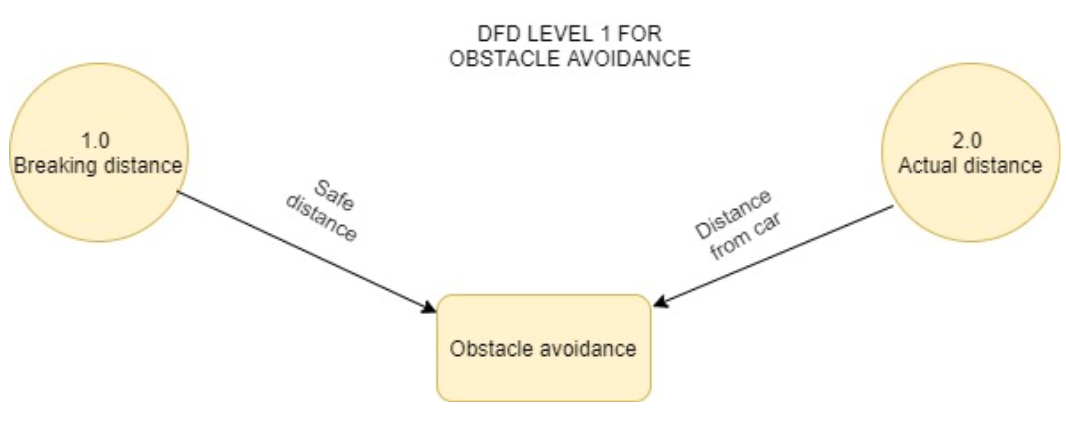


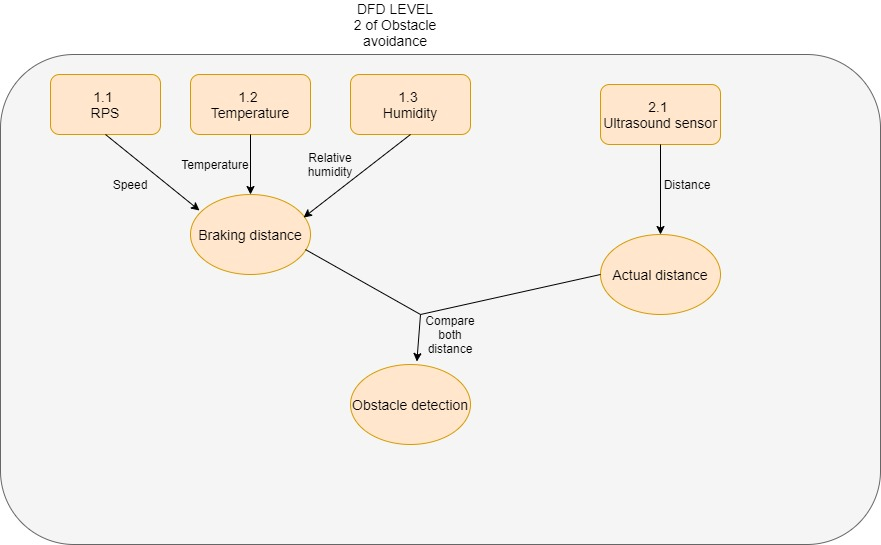
**5.5 Data Flow Diagram**

****

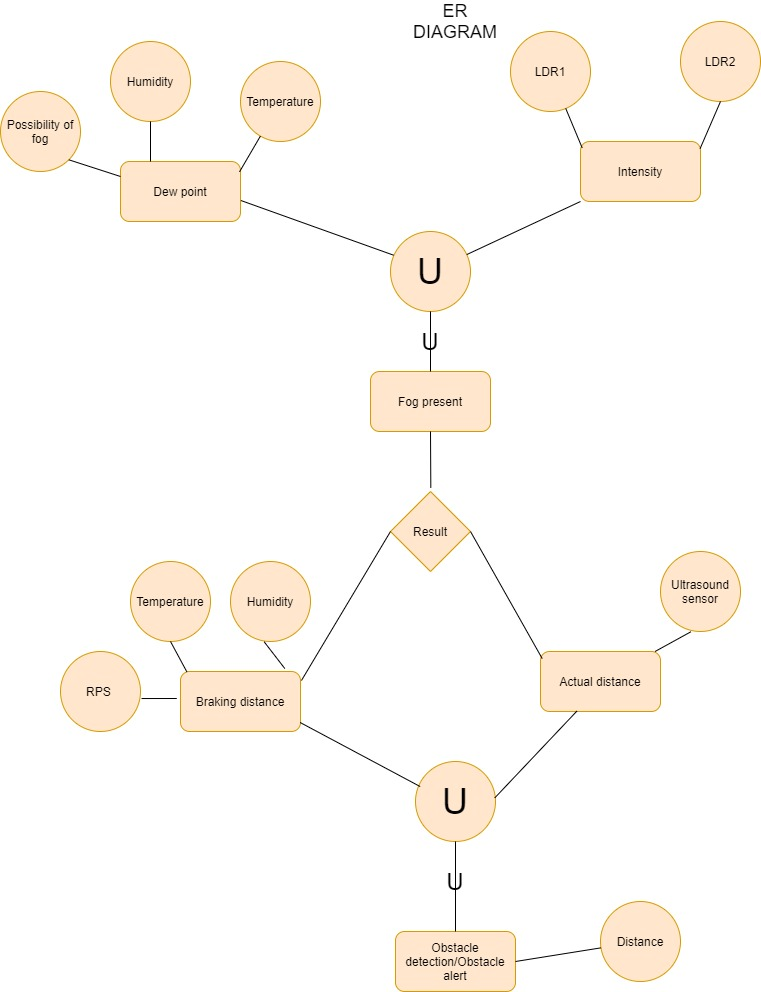
****

****

****

****

**5.6 ER Diagram**

****

**6.Risk Management**

| **Sno.** | **RISK** | **CATEGORY** | **PROBABILITY** | **IMPACT** | **RMMM PLAN** |
| --- | --- | --- | --- | --- | --- |
| **1** | Some team members leave the project development in between. | Technical risk | 30% | 2 | Use backup staffs which knows what was going on in the project. |
| **2** | Delivery deadline tightened. | Project risk | 30% | 1 | Team may use extra members to complete the task on schedule time. |
| **3** | Losing of all the project data. This may be caused by a hard disk being wiped out by a virus, hard disk failure etc. | Project risk | 20% | 2 | Carry out necessary backup of database data, source code and documentation. |
| **4** | Team  dissension/lack of cohesion. | Project risk | 10% | 3 | We could set some guide lines and rules how we deal with each other. |

**7. Implementation**

* **Implementing Module for FOG DETECTION**

#include <DHT.h>

#define Type DHT11

intsensePin=8;

DHT HT(sensePin,Type);

float hum;

float temp;

float td;

//===========================================

intLDRbox = A5;

intLDRopen = A4 ;

floatReadingBox ;

floatReadingOpen ;

int foggy; // fog not present

intdt=3000;

void setup() {

  // put your setup code here, to run once:

  Serial.begin(9600);

  HT.begin();

  delay(dt);

  Serial.println("Fog forms when the difference between air Temperature and dew point is less than 2.5 ° C");

  pinMode(LDRbox, INPUT);

  pinMode(LDRopen, INPUT);

}

void loop() {

  hum=HT.readHumidity();

  temp=HT.readTemperature();

  td = temp-( (100.-hum)/5. );

  Serial.print("Humidity : ");

  Serial.print(hum);

  Serial.print(", Temperature : ");

  Serial.println(temp);

  Serial.print("Dew point : ");

  Serial.println(td);

// if(temp-td>2.5)

// {

// Serial.println("No Fog present");

// }

// else

// {

// Serial.println("Possibility of Fog Formation");

// }

  Serial.println();

//============================================================================================

  foggy = 0;

  ReadingBox = analogRead(LDRbox);

  ReadingOpen = analogRead(LDRopen);

  Serial.print("Normal Environment : ");

  Serial.print(ReadingBox);

  Serial.print(" , Foggy Environment : ");

  Serial.println(ReadingOpen);

  if(ReadingBox-ReadingOpen> 3 || ReadingOpen-ReadingBox> 3)

  {

    foggy = 1;

  }

  if(foggy == 0)

  {

     Serial.println("Visibilty is not affected ");

  }

  if(foggy == 1)

  {

     Serial.println("Visibilty Is affected ");

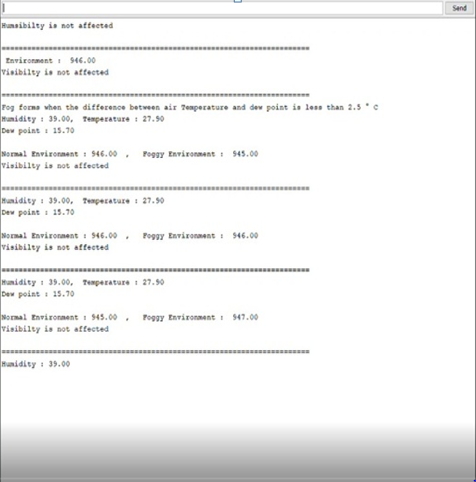
  }

  Serial.println();

  Serial.println("========================================================================");

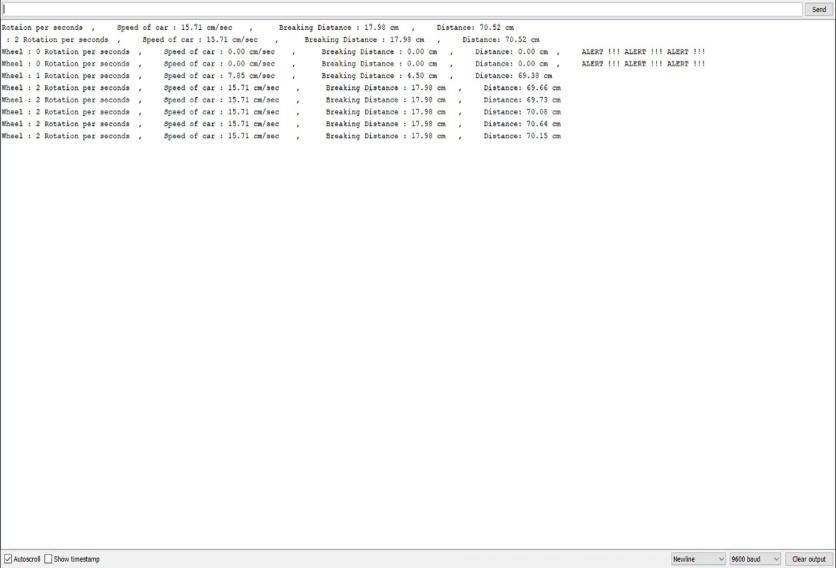
  delay(dt);

}



* **Implementing Module for OBSTACLE AVOIDANCE**

| #include "TimerOne.h"  #include "NewPing.h"  #include <DHT.h>  #define Type DHT11  #define MAX\_DISTANCE 400  #define TRIGGER\_PIN 3  #define ECHO\_PIN 4  int led = 6;  int M1a = 11 ; // motor pin  int M1b = 12 ; // motor pin  intpotPin = A0; // potentiometer to vary moter speed  intsensePin = 8;  float humidity;  float temperature;  DHT HT(sensePin,Type);  float g=9.8;  intpotvalue;  intmotorspeed ;  int counter=0;  intrps;  float velocity;  float cof ; // coefficient of friction for road and tire  float breakingDistance ;  float duration; // Stores HC-SR04 pulse duration value  float distance; // Stores calculated distance in cm  float soundsp; // Stores calculated speed of sound in M/S  float soundcm; // Stores calculated speed of sound in cm/ms  int iterations = 5;  NewPing sonar(TRIGGER\_PIN, ECHO\_PIN, MAX\_DISTANCE);  void docount() // counts from the speed sensor  {    counter++; // increase +1 the counter value  }  void timerIsr()  {    Timer1.detachInterrupt(); //stop the timer    Serial.print("Wheel : ");    rps = (counter / 20); // divide by number of holes in Disc    Serial.print(rps,DEC);    Serial.print(" Rotation per seconds , ");    velocity = rps \*2 \* 3.14159 \* 1.25 ; // v=wr(cm/sec) , r of wheel = 1.25cm    Serial.print(" Speed of car : ");    Serial.print(velocity);    Serial.print(" cm/sec , ");    breakingDistance = (velocity \* velocity ) / ( 2 \* 9.8 \* cof );    Serial.print(" Breaking Distance : ");    Serial.print(breakingDistance);    Serial.print(" cm ");      Serial.print(" , Distance: ");    Serial.print(distance);    Serial.println(" cm");    if(distance <= breakingDistance)    {      Serial.print(" , ALERT !!! ALERT !!!ALERT !!! ");      digitalWrite(led, HIGH) ;      delay(500);    }    else    {      digitalWrite(led, LOW) ;    }    Serial.println();      counter=0; // reset counter to zero    Timer1.attachInterrupt( timerIsr ); //enable the timer  }  void setup() {    // put your setup code here, to run once:    Serial.begin(9600);    HT.begin();      pinMode(M1a, OUTPUT);    pinMode(M1b, OUTPUT);    pinMode(potPin , INPUT);    pinMode(led, OUTPUT) ;    delay(2000); // Delay so DHT-22 sensor can stabalize    humidity = HT.readHumidity();    temperature = HT.readTemperature();    if( humidity == 100 ) //coefficient of friction depending on temperature and humidity    {        if( temperature < 0)        {            cof = 0.25 ; //snow        }        else        {            cof = 0.5 ; //rain        }    }    else    {        cof = 0.7 ;    }    Timer1.initialize(1000000); // set timer for 1sec    attachInterrupt(0, docount, RISING); // increase counter when speed sensor pin goes High    Timer1.attachInterrupt( timerIsr ); // enable the timer  }  void loop() {    // put your main code here, to run repeatedly:    potvalue= analogRead(potPin);    motorspeed = //(255./1023.)\*potvalue;    map(potvalue, 0, 1023,80, 255);    analogWrite(M1a, motorspeed); // set speed of motor (0-255)    digitalWrite(M1b, 0);    soundsp = 331.4 + (0.606 \* temperature) + (0.0124 \* humidity); // Calculate the Speed of Sound in M/S    soundcm = soundsp / 10000; // Convert to cm/ms    duration = sonar.ping\_median(iterations);    distance = (duration / 2) \* soundcm; // Calculate the distance  } |  |  |
| --- | --- | --- |



**8. Conclusion**

The entire project has been designed and developed as per the requirements of the user. It is found to be bug free as per the testing standards that are implemented.

The whole system’s activities are divided into two major parts. For implementing the system Arduino IDE is used.

The system comprises of following features:

* fog detection
* obstacle avoidance

There are also few features which can be integrated with this system to make it more flexible. Below list shows the future points to be consider:

* Getting to collaborate with future car designers to integrate this system into cars basic model.
* This can further be used in developing automatic cars.

Finally, we like to conclude that we put all our efforts throughout the development of our project and tried to fulfil most of the requirements of the user.

*References*

* 1. IEEE Software Engineering Standards Committee, “IEEE Std 830-1998,

IEEE Recommended Practice for Software Requirements Specifications”, October 20, 1998. [IEEE] The applicable IEEE standards are published in “IEEE Standards Collection,” 2001 edition.

* 1. The principle source of text book material is “Software Engineering APRACTITIONER’ SAPPROACH” by Roger S. Pressman*.*
  2. The principle source of text book material is “A Concise Introduction to Software Engineering” by PankajJalote*.*





5.



6.